МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Маслянко П. П.,
Ковальчук -Химюк Л. О.

ПРАКТИКА ЗДОБУВАЧІВ СТУПЕНЯ БАКАЛАВР

Навчальний посібник

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського як
навчальний посібник для здобувачів ступеня бакалавра за освітньою програмою
«Наука про дані та математичне моделювання»
спеціальності 113 Прикладна математика

Електронне мережне навчальне видання

Київ
«ПТФ Просвіта»
2025

УДК 519.2+004.62
М-31

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 7 від 09 .05 .2024 р.)
за поданням Вченої ради факультету прикладної математики
(протокол №10 від 29. 04. 2024 р.)

Відповідальний редактор: *Тарасенко-Клятченко О.В.*, кандидат техн. наук, доцент

Рецензенти: *Малежик Михайло Павлович*, докт. фіз.-мат. наук, професор, професор кафедри комп'ютерної та програмної інженерії УДУ імені Михайла Драгоманова

Данилов Валерій Якович, докт. фіз.-мат. наук, професор, професор кафедри штучного інтелекту НН ІПСА КПІ імені Ігоря Сікорського

Головний редактор *Сирота С. В.* канд. техн. наук, доц., доцент кафедри прикладної математики ФПМ

Маслянко Павло

М-31 Практика здобувачів ступеня бакалавр [Електронний ресурс] : навч. посіб. для здобувачів ступеня бакалавра/ за освіт. програмою «Наука про дані та математичне моделювання» спец. 113 / П. П. Маслянко, Л. О. Ковальчук-Химюк; КПІ ім. Ігоря Сікорського. – Мережеве електронне ви¬дання (1 файл: 24,7 Мбайт). Київ : ПТФ Просвіта, 2025. – 315 с.
ISBN 978-617-7010-35-6

Автори позиціонують навчальний посібник як системний, теоретичний і прикладний інструмент остаточної систематизації накопичених студентами знань, умінь та інформаційних ресурсів за темою бакалаврської атестаційної роботи, як дорожню карту виконання завдань такої систематизації, як один із прикладів результатів систематизації у вигляді структурних елементів і окремих частин бакалаврської атестаційної роботи. Навчальний посібник розроблений для студентів-бакалаврів третього та четвертого року навчання, які навчаються за спеціальністю 113 «Прикладна математика» за освітньою програмою «Наука про дані та математичне моделювання» факультету прикладної математики КПІ ім. Ігоря Сікорського, як науковий і методичний інструмент проходження переддипломної практики.

УДК 519.2+004.62

ISBN 978-617-7010-35-6

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ЗВО – заклад вищої освіти

БКР – бакалаврська кваліфікаційна робота

ДК – дорожня карта

МД – магістерська дисертація

МС УМ ПП БКР – Матриця Сутностей уніфікованої моделі Проходження Практики Бакалаврської Кваліфікаційної Роботи

Орг.С – Організаційна Система

УМ ПП – Уніфікована Модель Проходження Практики

ФПЗ БКР – формалізації постановки задачі бакалаврської кваліфікаційної роботи

ВСТУП

Практика є ключовою складовою у підготовці фахівців вищої освіти, при цьому переддипломна практика стає кульмінацією навчального процесу для бакалавра освітньо-професійного напряму підготовки у закладах вищої освіти (ЗВО) України. Вона має важливе практичне значення для оцінки навчальних досягнень студентів, особливо у технічних спеціальностях. Практика дозволяє перевірити рівень теоретичної та практичної підготовки, засвоєних знань, умінь та навичок студентів, а також їхню здатність застосовувати їх на практиці. Освітні програми підготовки бакалавра передбачають проведення п'яти тижнів практики для завершення бакалаврських кваліфікаційних робіт студентами.

Положенням про екзаменаційну комісію та атестацію здобувачів вищої освіти в КПІ ім. Ігоря Сікорського [https://document.kpi.ua/files/2020_7-178.pdf], встановлено два види бакалаврських кваліфікаційних робіт (БКР): дипломний проєкт і дипломна робота.

Цитата - «Дипломний проєкт є завершеною інженерною розробкою об’єкта проєктування (системи, пристрою, технологічного процесу, програмного застосунку, тощо) і передбачає синтез об’єкта проєктування, який відповідає вимогам завдання на дипломний проєкт із докладною розробкою певної функціональної частини, (елемента, вузла, підсистеми, технологічної операції, тощо) з урахуванням рівня розвитку відповідної галузі, досягнень науки і техніки».

Цитата - «Дипломна робота передбачає систематизацію, закріплення, розширення теоретичних і практичних знань зі спеціальності та застосування їх для вирішення конкретних наукових, економічних, виробничих та інших завдань, розвиток досвіду самостійної роботи, оволодіння методами

моделювання, дослідження процесів, об’єктів, систем у певній галузі економіки»

Таким чином підготовка БКР в сучасних умовах та організація і зміст переддипломної практики потребують враховувати можливості та досвід студентів у роботі за спеціальністю, набутого ще на молодших курсах університету.

Крім цього, вимоги ринку до кваліфікаційного рівня бакалавра, зокрема випускників технічних вищих навчальних закладів України, включають потребу в теоретичних, прикладних, технологічних та інших компетенціях, які є критично важливими для виробництва товарів та послуг у сучасному високотехнологічному середовищі, умовах жорсткої конкуренції, обмежених ресурсів, ризиків та невизначеності на ринку праці. Таким чином, для підготовки конкурентоспроможних на цьому ринку бакалаврів потрібно приділити особливу увагу на технологічну, дослідну, теоретичну та прикладну складові переддипломної практики.

Досвід провідних українських та закордонних вищих навчальних закладів показує, що неможливо повністю оволодіти науковими, технологічними та інженерними аспектами для успішного виконання бакалаврських кваліфікаційних робіт та розвитку професійних навичок протягом переддипломної практики, яка передбачена освітніми програмами. Тому важливо почати підготовку до практики та написання бакалаврської кваліфікаційної роботи задовго до початку самої переддипломної практики.

Ми тут акцентуємо увагу, насамперед, на завершенні роботи над бакалаврськими кваліфікаційними роботами на стадії переддипломної практики. Для цього організація навчального процесу має бути побудована таким чином, щоб закласти основи професійної підготовки майбутніх бакалаврів упродовж усіх курсів, починаючи з першого а, власне, сама практика виступатиме у ролі заключного етапу навчального процесу за бакалаврськими програмами.

Ринок сучасної освіти пропонує студентам різноманітні можливості для отримання теоретичних знань та практичних навичок на різних курсах у корпоративних університетах провідних ІТ-компаній, на великій кількості відкритих освітніх онлайн-платформ, результати проходження яких зараховуються при навчанні в ЗВО.

Зростаюча популярність отримання практичного досвіду і навичок на практиці в університетах стає все більш поширеною. Заклади вищої освіти впроваджують нові форми навчання, зокрема дуальну систему, яка комбінує теоретичні знання, отримані в аудиторії, самостійну роботу та роботу на підприємствах. Цей підхід дозволяє студентам використовувати свої знання не лише для вирішення теоретичних завдань, але й для практичних задач, що важливо в сучасному світі.

Набуває все більшого поширення і участь студентів, одночасно з навчанням, у стартапах різного рівня складності, вітчизняних та міжнародних конкурсах, корпоративних та університетських олімпіадах.

На нашу думку, ця тенденція, коли студенти активно здобувають знання та навички з усіх можливих джерел і ресурсів, ймовірно, продовжить розвиватись, удосконалюватись та набирати системний характер у сфері конкурентного ринку освітніх послуг. У таких умовах важливо не лише передавати студентам теоретичні знання, але й пропонувати науково обґрунтовані, практичні системні моделі для застосування їх набутих знань і навичок у практиці, зокрема в реалізації інженерних проєктів та розробок.

Автори навчального посібника зосереджують увагу студентів на переліку та змісті всіх фундаментальних та прикладних навчальних дисциплін, які вивчалися та можуть бути використані для розробки індивідуальних інженерних проєктів та завдань бакалаврської кваліфікаційної роботи. При цьому потрібно уникнути повторення вмісту цих дисциплін. Мета полягає в тому, щоб продемонструвати їх сутність і можливості практичного використання для вирішення конкретних завдань бакалаврської кваліфікаційної роботи. На нашу думку, набуття студентами практичних

навичок для проведення індивідуальних інженерних розробок і проєктів за темою бакалаврської кваліфікаційної роботи, має починатися вже з виконання практичних завдань і лабораторних робіт, курсових робіт і проектів з дисциплін, перш за все, професійного спрямування.

Такий підхід, на нашу думку, забезпечує максимально продуктивну і ґрунтовну професійну підготовку майбутніх бакалаврів.

У навчальному посібнику запропонована авторська уніфікована модель дисципліни «Практика», її складові та процес проходження переддипломної практики студентами здобувачами ступеня бакалавра .

Практична підготовка, розвиток навичок і умінь - це аспекти, які автори стараються показати студентам через конкретні приклади виконання завдань практики. Вони пропонують студентам певну дорожню карту для проведення практики, а також надають науково-методичні поради щодо того, як зібрати необхідні інформаційні ресурси для написання бакалаврської кваліфікаційної роботи.

Такий підхід дозволяє студентам систематично застосовувати набуті теоретичні знання і практичні навички у своїй майбутній професійній діяльності. Вони мають можливість збирати необхідні знання та ресурси, ознайомлюватись з кращими практиками відомих індустрій та виробляти власні креативні підходи до реалізації проектів.

Крім того, в рамках своїх курсів викладачі повинні підтримувати кожного студента у пошуку власного наукового напряму та створенні індивідуальних інженерних проєктів, що відповідають темі їх бакалаврської роботи.

Тому тут, і надалі у тексті, ми будемо оперувати терміном «практика», маючи на увазі інтеграцію процесів вивчення, розуміння, засвоєння та застосування на виробництві студентами бакалаврату необхідних науково-дослідних, технологічних та інженерних робіт для проходження переддипломної практики.

Важливо розглядати та втілювати практику як творчий та інноваційний процес, який сформує в якості результату - бакалаврську кваліфікаційну роботу (БКР), її публічної презентації та опублікування наукових, теоретичних та прикладних досягнень з дотриманням норм академічної доброчесності. Хоча творчий зміст бакалаврської кваліфікаційної роботи вимагає креативності, важливо також строго дотримуватися законодавчих вимог щодо загальних рамок формування структури, змісту та оформлення бакалаврської кваліфікаційної роботи. Ця вимога у повному обсязі стосується і цього навчального посібника.

Структура і зміст навчального посібника базується на нормах Закону України «Про вищу освіту» [https://zakon.rada.gov.ua/laws/show/1556-18#Text], Положення про проведення практики студентами вищих навчальних закладів України, затвердженого наказом Міністерства освіти України 08.04.1993, № 93 (Із змінами, внесеними згідно з Наказом Міносвіти № 351(v0351281-94) від 20.12.94) [https://zakon.rada.gov.ua/laws/show/z0035-93#Text], Положення про організацію освітнього процесу в КПІ імені Ігоря Сікорського, затвердженому наказом № 7-124 від 20.07.2020 [https://kpi.ua/regulations], Положення про екзаменаційну комісію та атестацію здобувачів вищої освіти в КПІ ім. Ігоря Сікорського [https://document.kpi.ua/files/2020_7-178.pdf], Положення Про проведення практики здобувачів вищої освіти в КПІ імені Ігоря Сікорського затвердженого 24.09.2020 Наказом № 7/172 [https://osvita.kpi.ua/node/184],

Порядку надання грифів навчальним матеріалам в КПІ ім. Ігоря Сікорського (затвердженого наказом ректора № НОН/276/2022 від 29.09.2022 р.) [https://osvita.kpi.ua/sites/default/files/downloads/Poriadok-nadannia-gryfiv_2022.pdf]

При написанні цього навчального посібника з проходження практики студентами бакалаврату, автори спирались на набутий власний досвід і критичні зауваження колег на написаний нами навчальний посібник Практика здобувачів ступеня магістра [Маслянко, П. П. Практика здобувачів

ступеня магістра [Електронний ресурс] : навч. посібн. для для здобувачів ступеня магістра за освітньою програмою «Наука про дані та математичне моделювання» спеціальності 113 Прикладна математика / П. П. Маслянко, Л. О. Ковальчук-Химюк ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 3,62 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 229 с. – Назва з екрана]. [https://ela.kpi.ua/handle/123456789/48625]

Таким чином цей навчальний посібник, розроблений на основі нового, авторського, наукового та методичного підходів [1], не дублює існуючі методичні рекомендації щодо проходження практики студентами бакалаврату, відповідає вимогам освітньо-професійних програм та освітньо-кваліфікаційних характеристик підготовки бакалаврів за освітньо-професійною програмою «Наука про дані та математичне моделювання» спеціальності 113 Прикладна математика і забезпечує єдиний комплексний підхід до організації заключної практичної підготовки на основі системності, безперервності і наступності навчання студентів.

Автори позиціонують навчальний посібник як системний, теоретичний і прикладний інструмент остаточної систематизації накопичених студентами знань, умінь та інформаційних ресурсів за темою БКР, як дорожню карту виконання завдань такої систематизації, як один із варіантів прикладу результатів систематизації у вигляді структурних елементів і окремих частин БКР.

Автори опираються на важливі та актуальні сьогодні основні принципи дисципліни "переддипломна практика". Зміст і форма навчального посібника базуються на наступних ключових аспектах [1]:

- різноманітність освіти в університетах та поза ними для здобуття знань та навичок;

- потребу високотехнологічних галузей у міждисциплінарних, креативних, наукових та інженерних рішеннях;

- значне посилення ролі інженерних вмінь поряд з академічними знаннями та кращими практиками у виробництві інженерних навичок;

- використання дистанційної, колективної форми для виконання проектів та постійного звітування про результати роботи;

- використання власного досвіду та найкращих практик колег у ІТ-галузі щодо системної інженерії проектів інформатизації організаційних систем.

Навчальний посібник призначений, перш за все, для студентів бакалаврату третього і четвертого року навчання як навчальний посібник та науково-методичний інструмент наскрізної підготовки бакалаврської кваліфікаційної роботи і її остаточне доопрацювання на стадії практики.

Для зручності використання, назви розділів навчального посібника співпадають з назвами встановлених обов'язкових розділів програми переддипломної практики, а зміст розділів показано на реальних прикладах. Принагідно зазначимо, що назви розділів і підрозділів наведених прикладів виділено курсивом.

Навчальний посібник може бути корисним аспірантам, які проходять педагогічну практику і приймають участь у організації і проведенні практики студентів бакалаврату за наскрізними освітніми програмами «бакалавр – магістр – доктор філософії».

Автори навчального посібника мають на меті висловити свою думку щодо принципових відправних положень дисципліни «переддипломна практика», організації процесів її виконання, своє бачення теоретичних засад дисципліни, структури та змісту практики і поділитись власним досвідом з колегами, науково-педагогічними працівниками, а особливо з керівниками бакалаврських кваліфікаційних робіт, а також ознайомитись з їх досвідом організації та виконання бакалаврських кваліфікаційних робіт та проведення практики.

Авторський вклад у зміст навчального посібника розподілений таким чином.

Ідея, форма представлення, зміст, вступ, розділи 1- 6 навчального посібника розроблені і написані П.П. Маслянко.

Розділи 7, 9, 10, підрозділи 6.2 – 6.4 написані Л.О. Ковальчук – Химюк.

Вступ, Розділ 8 і Додатки 7,8 (силабуси) написані спільно П.П. Маслянко та Л.О. Ковальчук – Химюк.

Приклади фрагментів виконання завдань практики ми взяли із реальних результатів проходження практики з підготовки реальних бакалаврських кваліфікаційних робіт та магістерських дисертацій, започаткованих і виконаних у співавторстві з науковим керівником при вивченні дисциплін «Системи Data Science», «Основи наукових досліджень» та «НДР за темою МД» студентів Катерини Павловської, Івана Савчука та Євгена Будзінського

Викладач дисциплін і науковий керівник бакалаврських кваліфікаційних робіт та магістерських дисертацій П.П. Маслянко.

Список посилань до вступу

1. Маслянко, П. П. Практика здобувачів ступеня магістра [Електронний ресурс] : навч. посібн. для для здобувачів ступеня магістра за освітньою програмою «Наука про дані та математичне моделювання» спеціальності 113 Прикладна математика / П. П. Маслянко, Л. О. Ковальчук-Химюк ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 3,62 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 229 с. – Назва з екрана].

[https://ela.kpi.ua/handle/123456789/48625]

1. ЗАГАЛЬНІ ПОЛОЖЕННЯ НОРМАТИВНИХ ДОКУМЕНТІВ ЩОДО ОРГАНІЗАЦІЇ ТА ЗМІСТУ ПЕРЕДДИПЛОМНОЇ ПРАКТИКИ ЗДОБУВАЧІВ СТУПЕНЯ БАКАЛАВР

Проходження практики – творчий процес, тим не менше, він визначається і проводиться на основі певних законодавчих норм і правил.

Метою практичної підготовки є оволодіння здобувачами вищої освіти сучасними методами, формами організації й знаряддями праці в галузі їх майбутньої професійної діяльності, формування на базі одержаних знань, професійних умінь та набуття досвіду прийняття самостійних рішень під час конкретної роботи в реальних ринкових та виробничих умовах, виховання потреби систематично оновлювати свої знання й творчо їх застосовувати в практичній діяльності [1].

Основним нормативним організаційно-методичним документом, що регламентує структуру, зміст і процес проведення практики є програма практики [2].

Програма практики повинна:

- базуватись на освітніх програмах підготовки бакалаврів;
- пропонувати студентам дорожню карту і регламентувати стадії засвоєння знань і набуття умінь у процесі проходження практики;
- надавати чіткі критерії і процедури оцінювання рівня набутих знань, умінь і компетенцій, набутих у процесі проходження практики.

Програма практики складається з таких розділів [2, 3]:

- мета і завдання практики;
- організація проведення практики;
- зміст практики;
- індивідуальне завдання практики;
- вимоги до звіту про практику;
- підсумки практики.

Основним організаційно-методичним документом, що регламентує діяльність студентів, наукових керівників і керівників практики, є робоча програма практики, яка розробляється у відповідності до методичних рекомендації з питань організації практики студентів та складання робочих програм практики Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» [2].

Робоча програма практики (силабус) складається у відповідності до освітньо-професійної програми підготовки бакалаврів за освітньо-професійною програмою «Наука про дані та математичне моделювання» за спеціальності 113 Прикладна математика.

Терміни проведення переддипломної практики визначаються навчальним планом [1] і становлять 5 тижнів. Під час проходження практики студент збирає додаткові інформаційні ресурси за місцем проходження практики, вивчає та систематизує матеріали накопичені та розроблені ним на третьому та четвертому курсі бакалаврату в рамках курсів «Основи машинного навчання», «Математичне моделювання», «Системи глибинного навчання» та інших дисциплін професійного спрямування, потрібних для завершення написання бакалаврської атестаційної роботи.

Практика може проходити на підприємстві, в організації або в навчальному закладі. Керівництво практикою з боку ЗВО здійснює викладач кафедри, відповідальний за проходження практики та керівник випускної кваліфікаційної роботи, а з боку підприємства — призначений керівник практики із числа фахівців за профілем спеціальності.

Форма проведення практики визначається нормативними документами університету і може бути очна, дистанційна або змішана.

В залежності від форми проведення практики відповідальний за проходження практики, керівник випускної кваліфікаційної роботи, призначений керівник практики з боку підприємства визначають і реалізують організаційні заходи роботи і проведення комунікацій зі студентами для забезпечення виконання завдань практики кожного студента з урахуванням

всіх вимог до бакалаврської кваліфікаційної роботи і підготовки звіту з практики.

Тим не менше, яка б не була форма практики, результати переддипломної практики у будь якому випадку полягають у тому, щоб в результаті застосування набутих знань і навичок кожен студент повинен отримати конкретний науковий та/або науково-прикладний результат. Закон України Про наукову і науково технічну діяльність дає вичерпне означення і пояснення наукового та науково-прикладного результату [https://zakon.rada.gov.ua/laws/show/848-19#Text].

«Науковий результат – нове наукове знання, одержане в процесі фундаментальних або прикладних наукових досліджень та зафіксоване на носіях інформації. Науковий результат може бути у формі звіту, опублікованої наукової статті, наукової доповіді, наукового повідомлення про науково-дослідну роботу, монографічного дослідження, наукового відкриття, проекту нормативно-правового акту, нормативного документа або науково-методичних документів, підготовка яких потребує проведення відповідних наукових досліджень або містить наукову складову, тощо» [4].

Тобто науковий результат може бути представлений здобувачами наукового ступеня «бакалавр» у формі фундаментального та/або науково-технічного (прикладного) результату.

Бакалаврська кваліфікаційна робота (БКР), курсові роботи і проекти, презентації і виступи на конференціях, участь у написанні звітної документації, та інші офіційно зафіксовані і оприлюднені наукові результати, - це все форми публічного представлення і публікації отриманих студентами наукових результатів.

Наукові результати, отримані за темою БКР, мають бути обов’язково використані студентами у власних БКР з відповідними посиланнями у переліку використаної літератури до БКР.

На наукові результати, що належать іншим авторам і були використані при написання бакалаврської роботи, в обов’язковому порядку повинні бути посилання у тексті магістерських дисертацій, а самі наукові результати повинні бути подані у переліку використаної літератури.

Правила запобігання академічному плагіату у повному обсязі наводяться за посиланням https://document.kpi.ua/files/2020_1-76.pdf і сформульовані у Положенні про систему запобігання академічному плагіату у КПІ ім. Ігоря Сікорського у відповідності до наказу № 1/76 від 25.02.20.

Список посилань до розділу

1. Положення про організацію освітнього процесу в КПІ ім.Ігоря Сікорського: затверджено наказом Ректора, НАКАЗ № 7-124 від 20.07.2020. [електронний ресурс] . – Режим доступу – https://document.kpi.ua/regulations . – Назва з екрану.- Мова укр.

2. Положення про порядок проведення практики здобувачів вищої освіти Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» затверджено наказом Ректора, НАКАЗ № 7/172 від 24.09.2020. [електронний ресурс] . – Режим доступу – https://document.kpi.ua/files/2020_7-172.pdf . – Назва з екрану.- Мова укр.

3. Методичні рекомендації з питань організації практики студентів та складання робочих програм практики Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» [Текст] / Уклад.: Н. М. Лапенко, І.Л. Співак, І.В. Федоренко, О.М. Шаповалова; за заг. Ред. П.М. Яблонського. – К.: КПІ ім. Ігоря Сікорського, 2018. – 29 с.

4. Закон України Про наукову і науково технічну діяльність, [електронний ресурс]. – Режим доступу – https://zakon.rada.gov.ua/laws/show/848-19#Text

–Назва з екрану.- Мова укр.

2. РОБОЧА ПРОГРАМА ПРАКТИКИ, СТРУКТУРА ТА ЗМІСТ РОЗДІЛІВ

Робоча програма практики (силабус) для бакалаврів освітньо-професійного напрямку (Додаток 7), це навчально методичний документ, що упорядковує процес проходження практики, організовує роботу студента, навчає формувати структуру, зміст розділів і порядок проходження практики, показує студентам як виконувати окремі завдання практики для написання бакалаврської атестаційної роботи.

Робоча програма практики складається з таких розділів [2]:

- мета і завдання практики бакалаврів освітньо-професійного напрямку для спеціальності 113 Прикладна математика за спеціалізацією Наука про дані та математичне моделювання;
- організація проведення практики ;
- зміст практики бакалаврів;
- індивідуальне завдання практики бакалаврів;
- вимоги до звіту про практику бакалаврів;
- підсумки практики;
- форми і методи контролю проходження практики;
- критерії оцінювання виконання завдань та звіту з практики;
- рекомендована література для виконання індивідуальних завдань практики.

Зміст розділів висвітлює і загальні положення, і індивідуальні завдання практики для кожного студента.

Далі зупинимось на структурі і детальному змісті цих розділів та, на основі ринкових категорій визначених у вступі до навчального посібника, покажемо власне, авторське бачення їх реалізації для студентів та колег-викладачів.

Список посилань до розділу

1. Положення про організацію освітнього процесу в КПІ ім. Ігоря Сікорського: затверджено наказом Ректора, НАКАЗ № 7-124 від 20.07.2020. [електронний ресурс] . – Режим доступу – https://document.kpi.ua/regulations – Назва з екрану. - Мова укр.

2. Положення про порядок проведення практики здобувачів вищої освіти Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» затверджено наказом Ректора, НАКАЗ № 7/172 від 24.09.2020. [електронний ресурс] . – Режим доступу – https://document.kpi.ua/files/2020_7-172.pdf. – Назва з екрану.- Мова укр.

3. МЕТА І ЗАВДАННЯ ПЕРЕДДИПЛОМНОЇ ПРАКТИКИ БАКАЛАВРІВ ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ

Розділ «Мета і завдання практики» є одним з основних розділів робочої програми, що визначає професійну спрямованість діяльності студентів та викладачів під час проведення практики. Завдання практики полягають у формуванні структури та змісту практичних заходів за напрямком майбутньої професії, систематизації набутих у процесі навчання знань і умінь для виконання бакалаврської атестаційної роботи.

Мета переддипломної практики – систематизація накопичених студентами знань і умінь для продукування наукового/прикладного результату та вирішення практичних задач виробництва.

Зокрема метою практики є:

- систематизація, отриманих у процесі навчання, теоретичних знань і практичних навичок та їх застосування для вирішення наукових і прикладних досліджень та розробок;

- набуття практичних навичок реалізації реальних наукових та прикладних розробок для підприємств, установ і організацій;

- уміння формалізувати постановку задачі та проводити науково обґрунтовуваний вибір методів і засобів для вирішення реальних наукових і прикладних досліджень та інженерних розробок;

- уміння формувати структуру і зміст бакалаврських атестаційних робіт за індивідуальними темами, оприлюднення результатів наукових і прикладних досліджень у наукових працях, доповідях, презентаціях, тощо;

- набуття навичок продукування наукових та науково-прикладних результатів;

- набуття професійних компетентностей за спеціальністю і здатності позиціонувати себе на ринку праці.

Досягнення мети переддипломної практики, виконання завдань індивідуального завдання практики та формування наукового/прикладного

результату передбачає можливість реалізувати і продемонструвати наявність дослідницької та інженерної складової у бакалаврських кваліфікаційних роботах. Тобто, магістрант, окрім реалізації і демонстрації прикладного, інженерного результату у вигляді рішення конкретної інженерної задачі, має змогу реалізувати та продемонструвати ще й наукову новизну та/або практичну цінність своєї роботи.

Пояснимо, що у такому випадку однією з основних вимог до наукових/прикладних результатів, і до БКР у цілому, є наявність наукової новизни та/або практичної цінності отриманих результатів інженерних розробок. Наукова новизна і практична цінність, це дві споріднені категорії що характеризують належний науковий рівень та можливості, чи конкретні результати, застосування результатів БКР для вирішення конкретних інженерних задач задекларованих у бакалаврській атестаційній роботі.

Далі ми пропонуємо авторську інтерпретацію цих категорій для теоретичного розуміння і практичного застосування студентами при виконанні індивідуальних завдань практики і сформульовану у одному семантичному просторі понять [5].

Наукова новизна. Наразі у літературі можна зустріти декілька варіантів означення поняття «наукова новизна». Тим не менше практично всі вони зводяться до розуміння і пояснення того, що ті, чи інші наукові, або науково-прикладні результати були отримані вперше, але ці публікації не дають якогось академічного означення терміну «наукова новизна».

На думку авторів навчального посібника ці поняття можуть бути сформульовані таким чином [5].

Означення 1. Наукова новизна – це належним чином формалізована категорія, що характеризує нову, або суттєву відмінність нової властивості, ознаки, або будь якого іншого атрибуту результатів наукового дослідження або інженерної розробки від вже існуючих.

Уточнимо це узагальнене означення 1.

Означення 2. Наукова новизна – це належним чином формалізована категорія про нові дані, інформацію або знання, отримані про предмет досліджень, що змінюють, доповнюють або вдосконалюють наявні дані, інформацію або знання про цей предмет досліджень.

Зауважимо при цьому, що наукова новизна може стосуватись як теоретичних так і прикладних результатів наукових досліджень.

Деталізуємо і поширимо це означення на основні класи результатів наукових і прикладних досліджень, що можуть бути отримані в результаті виконання завдань індивідуальної програми практики.

Означення 3. Наукова новизна результатів теоретичних і прикладних досліджень – це належним чином формалізована категорія про науково обґрунтовані і доведені нові дані, інформацію або знання про теоретичні і прикладні результати наукових досліджень (*методи, методології, моделі, процеси, алгоритми, способи, системи, підсистеми, програмні модулі, програмні продукти, математичне та програмне забезпечення, логістичні, функціональні, динамічні, та інші моделі, класифікатори, підходи, методи застосування, фреймворки тощо, та інші інструменти*), що змінюють, доповнюють або вдосконалюють наявні дані, інформацію або знання про предмет теоретичних або прикладних досліджень.

Висновки. Наявність формалізованої, науково обґрунтованої і доведеної наукової новизни результатів наукових досліджень надає авторам, які встановили цю новизну, право стверджувати, що такі результати були отримані вперше і до цього ніде не були опубліковані і оприлюднені.

Далі розглянемо категорію «практична цінність». Поряд з науковою новизною не менш важливою характеристикою наукового результату, є практична цінність результатів наукових досліджень.

Означення 4. Практична цінність - це міра оцінювання науково-технічних, економічних, соціально-політичних, та інших потрібних характеристик корисності, ефективності, продуктивності, необхідності, тощо, застосування результатів дослідження.

Деталізуємо і поширимо це означення 4 на основні класи результатів наукових досліджень або інженерних розробок, що можуть бути отримані в результаті виконання індивідуального завдання програми практики і застосовані для вирішення завдань, задекларованих у бакалаврській атестаційної роботі.

Означення 5. Практична цінність результатів теоретичних і прикладних досліджень – це належним чином формалізована категорія про науково обґрунтовані і доведені при застосуванні на практиці нові дані, інформацію або знання про міру корисності, ефективності, продуктивності, тощо теоретичних і прикладних результатів наукових досліджень (*методів, методологій, моделей, процесів, алгоритмів, способів, систем, підсистем, програмних модулів, програмних продуктів, математичного та програмного забезпечення, логістичних, функціональних, динамічних, та інших моделей, класифікаторів, підходів, методів застосування, фреймворків тощо, та інших інструментів*), що змінюють, доповнюють або вдосконалюють наявні дані, інформацію або знання про міру корисності, ефективності, продуктивності, необхідності впровадження результатів теоретичних або прикладних досліджень.

Висновки. Таким чином при формуванні робочої програми та індивідуальних завдань практики потрібно обов’язково включати до них не тільки інженерні завдання і реалізацію лише прикладних інженерних задач, а і можливості для студентів отримати і наукову новизну та практичну цінність результатів своїх робіт. У цьому випадку до БКР потрібно включати наукову, науково-технічну, конструкторську, технологічну та інші дослідницькі складові необхідні для отримання наукового результату, що матиме і наукову новизну, і практичну цінність.

Для досягнення мети практики потрібно окреслити і перелік необхідних завдань та визначити процес - дорожню карту досягнення мети.

3.1. Конкретні завдання практики

У навчальному посібнику [5] всі завдання практики об’єднані у чотири класи за критеріями змісту та порядку їх виконання. Виконання кожного класу завдань закінчується проведенням поточного/календарного контролю, а виконання завершального, четвертого класу завдань – семестровим контролем (заліком).

Клас завдань 1. Завдання формування і систематизації ресурсів для проходження практики:

1) Написати заяву встановленого зразка на ім’я завідувача кафедри з попередньою назвою теми БКР і проханням призначити наукового керівника;
2) Формалізувати постановку задачі БКР: об’єкт, предмет і мета дослідження/роботи, кінцевий результат виконання БКР;
3) Сформувати матрицю інформаційних ресурсів (МІР) необхідних для виконання БКР;
4) Розробити шаблон БКР – першої версії БКР за форматом і вимогами Положення про державну атестацію бакалаврів КПІ ім. Ігоря Сікорського [2] з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами.

   За результатами виконання завдань передбачити проведення поточного/календарного контролю №1.

Клас завдань 2. Завдання продукування результатів практики:

1) Затвердити остаточну тему, формалізацію постановки задачі і завдання БКР. У разі необхідності уточнити зміст заяви;
2) Сформувати і захистити першу версію шаблону БКР з виконаними всіма розділами та додатками. Додаток А. Демонстраційний матеріал – презентація БКР, Додаток Б. Текст програмного коду;

Поточний контроль/ календарний контроль №2 проведення практики - за результатами виконання завдань і змісту документів для класу завдань 2.

Клас завдань 3. Завдання апробації результатів практики:

1) Провести верифікацію і валідацію результатів практики.
2) Формалізувати наукову новизну, практичну цінність та висновки за результатами виконання БКР.
3) Підготувати, та затвердити у н/керівника, презентацію результатів Звіту з переддипломної практики для заліку з практики;

Поточний контроль/календарний контроль № 3 проведення практики - за результатами виконання завдань і змісту документів для класу завдань 3.

Клас завдань 4. Завдання звітування (семестровий контроль):

1) Підготувати Звіт з практики;
2) Подати до комісії з прийому результатів переддипломної практики остаточну заяву з темою БКР, заповнений і підписаний щоденник з практики, звіт з практики, презентацію звіту з практики;
3) Захистити результати практики;
4) Ознайомитись з порядком попереднього захисту БКР.

Семестровий контроль (залік) проведення практики - за результатами звіту з практики, виконання завдань і змісту документів для класу завдань 4.

3.2. Дорожня карта переддипломної практики бакалаврів

Дорожня карта проведення переддипломної практики бакалаврів, це навчально-методичний документ що регламентує перелік, зміст завдань і порядок виконання робочої програми проведення переддипломної практики.

Дорожня карта повинна враховувати необхідні складові навчально-методичної та науково-дослідної роботи за темою БКР. У дорожній карті стадія навчально-методичної та науково-дослідної роботи (таблиця 3.1) за темою бакалаврської кваліфікаційної роботи в обов'язковому порядку проходить одночасно зі стадією переддипломна практика (таблиця 3.2).

Таблиця 3.1 Стадія навчально-методичної та науково-дослідної роботи

Завдання підготовки бакалаврів	Термін виконання /результат виконання	Відповідальні
1	2	3
1. Студент разом зі своїми керівниками бакалаврської роботи, уточнити та відредагувати теми БКР, постановку задачі й перелік завдань для досягнення мети дослідження у відповідності з нормативними документами «КПІ ім. Ігоря Сікорського».	вересень-жовтень поточного року	студенти, керівники БКР, ПІБ відповідальних по кафедрі
2. Уточнити розподіл педагогічного навантаження у відповідності з поданими заявами студентів та науково-педагогічним складом кафедри.	вересень поточного року /інше	студенти, керівники БКР, ПІБ відповідальних по кафедрі
3. Студентам, станом на визначену дату поточного року, подати заяви з остаточними темами своєї бакалаврської кваліфікаційної роботи, підписані студентами та науковими керівниками. Уточнити спосіб подачі заяви	жовтень поточного року /інше	студенти, керівники БКР, ПІБ відповідальних по кафедрі

відповідальній особі (електрона пошта, корпоративний месенджер та інше).		
4. На засіданні кафедри розглянути і затвердити остаточні теми бакалаврських кваліфікаційних робіт для формування наказу по «КПІ ім. Ігоря Сікорського».	листопад поточного року /інше	Керівники БКР, ПІБ відповідальних по кафедрі
5. Студентам, разом із керівниками БКР, підготувати структуру, завдання та зміст своїх бакалаврських кваліфікаційних робіт у відповідності до затверджених тем і нормативних документів щодо формування бакалаврських кваліфікаційних робіт.	вересень - жовтень поточного року /інше	студенти, керівники БКР,

Таблиця 3.2 Стадія переддипломної практики

Завдання проходження переддипломної практики	Термін виконання /результат виконання	Відповідальні
1	2	3
Навчально-методичні заходи переддипломної практики: Підготувати і провести загальні збори студентів разом з керівниками БКР і керівниками практики. Узгодити дату і час проведення зборів з науковими керівниками і керівниками практики [2]. Керівники БКР формують групи своїх студентів, уточнюють і редагують наявну документацію, ставлять індивідуальні завдання на практику за темою БКР, встановлюють індивідуальні графіки занять. Для підготовки і проведення загальних зборів студентів підготувати і відправити на адресу студентів електронні версії	Інформація щодо місця і часу проведення зборів, розклад консультацій з керівниками БКР, результати виконання навчально-методичних заходів.	студенти, керівники практики, керівники БКР ПІБ відповідальних по кафедрі

Продовження Таблиці 3.2

1	2	3
всіх необхідних для проходження практики шаблонів документів, зокрема: - щоденник з переддипломної практики; - Положення про екзаменаційну комісію та атестацію здобувачів вищої освіти в КПІ ім. Ігоря Сікорського [3]; - Дорожню карту проходження практики, та ін.		студенти, керівники практики, керівники БКР ПІБ відповідальних по кафедрі
Провести консультацію щодо заповнення всіх необхідних, для проходження практики, документів.		студенти, керівники практики, керівники БКР ПІБ відповідальних по кафедрі
Рекомендувати студентам-обговорити зі своїми керівниками БКР всі напрацьовані матеріали за темою БКР, а саме: - заяву встановленого зразка на ім’я завідувача кафедри ПМА з попередньою назвою теми БКР і проханням призначити керівника (Додаток 3);		

Продовження Таблиці 3.2

1	2	3
- формалізацію постановки задачі БКР (об'єкт, предмет і мета дослідження, кінцевий результат виконання БКР); - шаблон – першу версію БКР за форматом і вимогами Положення про екзаменаційну комісію та атестацію здобувачів вищої освіти в КПІ ім. Ігоря Сікорського [3] з прописаними структурою, завданнями на БКР і виконаним першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР» В разі зміни теми БКР, тут і надалі в процесі проходження переддипломної практики, перелік і зміст документів подається за зміненою темою.		
Науково-дослідні заходи переддипломної практики : - заняття з керівником БКР, уточнити тему, формалізацію постановки задачі і завдання БКР;		студенти, керівники практики, керівники БКР

Продовження Таблиці 3.2

1	2	3
- формування і захист першої версії БКР; - підготовка презентації першої версії БКР.	Структура і зміст першої версії БКР	студенти, керівники БКР, керівники практики
Конкретні завдання переддипломної практики. ***Клас завдань 1. Завдання формування і систематизації ресурсів для проходження практики*** 1.1 Написати заяву встановленого зразка на ім'я завідувача кафедри з попередньою назвою теми БКР і проханням призначити керівника БКР (Додаток 3); 1.2 Формалізувати постановку задачі БКР: об'єкт, предмет і мета дослідження/роботи, кінцевий результат виконання БКР; 1.3 Сформувати матрицю інформаційних ресурсів (МІР) необхідних для виконання БКР; 1.4 Розробити шаблон БКР – першої версії БКР за форматом	Перший–четвертий тижні практики	студенти, керівники практики, керівники БКР

і вимогами Положення про екзаменаційну комісію та атестацію здобувачів вищої освіти в КПІ ім. Ігоря Сікорського [3] з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами.		
Клас завдань 2. Завдання продукування результатів практики 2.1 Затвердити остаточну тему, формалізацію постановки задачі і завдання БКР. У разі необхідності уточнити зміст заяви; 2.2 Сформувати і захистити першу версію шаблону БКР з виконаними всіма розділами та додатками. Додаток А. Демонстраційний матеріал – презентація БКР, Додаток Б. Текст програмного коду;	Четвертий - п’ятий тиждень практики	
Клас завдань 3. Завдання апробації результатів практики:	Четвертий - п’ятий тиждень практики	студенти, керівники практики,

3.1 Провести верифікацію і валідацію результатів практики. 3.2 Формалізувати наукову новизну (за наявності), практичну цінність та висновки за результатами виконання БКР. 3.3 Підготувати, та затвердити у керівника БКР звіт з практики та презентацію результатів Звіту з переддипломної практики для заліку з практики;		керівники БКР
Клас завдань 4. Завдання звітування (семестровий контроль): 4.1 Підготувати звіт з практики; 4.2 Подати до комісії з прийому результатів переддипломної практики остаточну заяву з темою БКР, заповнений і підписаний щоденник з практики, звіт з практики, презентацію звіту з практики; 4.2 Захистити результати практики; 4.3. Ознайомитись з порядком попереднього захисту БКР.	П’ятий тиждень практики	

Продовження Таблиці 3.2

1	2	3
Вимоги до звіту з переддипломної практики 1. На захист результатів практики виноситься звіт з практики. 2. Структура і зміст розділів звіту з практики складається з таких розділів або документів: 2.1. Титульна сторінка звіту (за зразком Додаток 5); 2.2. Зміст звіту з практики; 2.3. Перелік умовних позначень, скорочень і термінів (за необхідності); 2.4. Вступ, з коротким оглядом проблемної області за якою виконується БКР, короткою аргументацією актуальності задекларованих досліджень /роботи, коротка анотація розділів звіту; 2.5. Формалізація постановки задачі БКР (об'єкт, предмет і мета дослідження/роботи, кінцевий результат виконання БКР); 2.6. Перша повна версія БКР з	Організація та порядок проведення захисту з практики, комісія для захисту практики, результати подання документації для захисту практики, силабус з практики, рекомендована структура і зміст презентації захисту звіту з практики.	студенти, керівники практики, керівники БКР

виконаними всіма розділами і додатками;		
2.7. Додаток А до БКР - Демонстраційна версія програмного забезпечення (архітектура, короткий опис функціональності, мова програмування, результати роботи програмного забезпечення, тощо); 2.8. Додаток Б до БКР - Презентація першої повної версії БКР з виконаними всіма розділами (окремі слайди можуть бути ще не розроблені); 2.9. Висновки до звіту; 2.10. Презентація звіту з практики.		студенти, керівники практики, керівники БКР

Звертаємо увагу читачів, що наведений приклад дорожньої карти переддипломної практики сформований на основі вимог освітньо-професійної програми підготовки бакалаврів [4], що можна знайти за посиланням

https://osvita.kpi.ua/113_OPPB_NDMM .

Таким чином кінцевий результат практики здобувачів ступеня бакалавр спеціальності 113 прикладна математика за спеціалізацією Наука про дані та математичне моделювання передбачає, організовує і забезпечує виконання заключного етапу підготовки фахівця, здатного формалізувати постановку

задачі і знаходити методи та засоби її вирішення в галузі науки про дані та здійснювати інноваційну професійну діяльність для комплексного виконання робіт з машинного навчання, інтелектуального аналізу даних та математичного моделювання об’єктів, процесів і явищ різного характеру, у тому числі тих, що пов’язані з обробкою великих обсягів даних (Big Data) [4].

Конкретні і обґрунтовані постановка задачі робочої програми практики, мета, завдання та дорожня карта проведення практики, дозволяють легко контролювати і виконувати завдання програми практики та своєчасно вносити необхідні зміни.

Список посилань до розділу

1. Положення про організацію освітнього процесу в КПІ ім. Ігоря Сікорського: затверджено наказом Ректора, НАКАЗ № 7-124 від 20.07.2020. [електронний ресурс] . - Режим доступу - https://document.kpi.ua/regulations - Назва з екрану.- Мова укр.

2. Положення про порядок проведення практики здобувачів вищої освіти Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» затверджено наказом Ректора, НАКАЗ № 7/172 від 24.09.2020. [електронний ресурс] . - Режим доступу - https://document.kpi.ua/files/2020_7-172.pdf. - Назва з екрану.- Мова укр.

3. Положення про екзаменаційну комісію та атестацію здобувачів вищої освіти в КПІ ім. Ігоря Сікорського (уведено в дію наказом №7/178 від 01.10.2020 р., зі змінами, внесеними наказом №НУ/71/2021 від 19.04.2021 р.) [електронний ресурс]. - Режим доступу- https://osvita.kpi.ua/sites/default/files/downloads/Pol_EK_atestaciia.pdf - Назва з екрану.- Мова укр.

4. Освітньо-професійна програма підготовки бакалаврів за спеціальністю 113 Прикладна математика за спеціалізацією Наука про дані та математичне моделювання [електронний ресурс]. - Режим доступу - https://osvita.kpi.ua/113_OPPB_NDMM- Назва з екрану.- Мова укр.

5. 1. Маслянко, П. П. Практика здобувачів ступеня магістра [Електронний ресурс] : навч. посібн. для для здобувачів ступеня магістра за освітньою програмою «Наука про дані та математичне моделювання» спеціальності 113 Прикладна математика / П. П. Маслянко, Л. О. Ковальчук-Химюк ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 3,62 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 229 с. – Назва з екрана].
[https://ela.kpi.ua/handle/123456789/48625]

4. ОРГАНІЗАЦІЯ ПРОВЕДЕННЯ ПЕРЕДДИПЛОМНОЇ ПРАКТИКИ

Керівник практики від кафедри забезпечує здійснення всіх організаційних заходів перед початком практики у відповідності до дорожньої карти проходження практики:

- проведення зборів і інструктаж про порядок проходження практики;
- надання студентам потрібних документів (направлення на практику, щоденник практики);
- проведення поточного та календарного контролю;
- надання студентам консультацій щодо виконання індивідуальних завдань практики.

Керівник випускної кваліфікаційної роботи та консультанти окремих розділів БКР надають студентам консультації з питань, пов’язаних із формуванням структури та змісту БКР.

В разі необхідності, кожен студент самостійно, або через завідувача кафедри, може звернутись до викладача за консультацією з будь якої необхідної йому дисципліни щодо змісту, або презентації результатів своєї БКР.

Для складання заліку з практики студент повинен представити керівнику практики від кафедри:

1) звіт з практики;
2) щоденник практики;
3) залікову книжку.

Студенту, який не виконав програму переддипломної практики без поважних причин, може бути надано право проходження практики повторно при виконанні умов, визначених університетом [1]. Студент, який при другому перескладанні отримав незадовільну оцінку з практики при складанні заліку комісії, відраховується з університету [2].

Список посилань до розділу

1. Положення про організацію освітнього процесу в КПІ ім.Ігоря Сікорського: затверджено наказом Ректора, НАКАЗ № 7-124 від 20.07.2020. [електроний ресурс] . - Режим доступу - https://document.kpi.ua/regulations.
- Назва з екрану.- Мова укр.

2. Положення про порядок проведення практики здобувачів вищої освіти Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» затверджено наказом Ректора, НАКАЗ № 7/172 від 24.09.2020. [електроний ресурс] . - Режим доступу - https://document.kpi.ua/files/2020_7-172.pdf.
- Назва з екрану. - Мова укр.

5. ЗМІСТ ПРАКТИКИ БАКАЛАВРІВ ЗА ОСВІТНЬО - ПРОФЕСІЙНОЮ ПРОГРАМОЮ

Процес навчання, і проходження переддипломної практики зокрема, це навчальна діяльність спрямована на отримання, встановлених нормативними документами, результатів з підготовки БКР. Переддипломна практика, як і будь який інший вид діяльності, може бути здійснена лише за наявності необхідного рівня підготовки студента, наявних ресурсів та встановленого чіткого регламенту її реалізації.

Ми викладаємо зміст цього розділу як відповідь на питання: - А з чого починати, власне, практику? Що є, так би мовити, теоретичною основою проходження практики, яке її абстрактне представлення і яким чином це представлення може бути реалізоване на практиці кожним студентом?

При цьому необхідно пам'ятати і обов'язково враховувати вже набуті практичні навички студентів, їх теоретичну підготовку та професійну орієнтацію.

Така наглядна уніфікована модель проходження практики, на нашу думку, може бути побудована на основі теорії систем і системного підходу [1].

Продемонструємо зміст практики бакалаврів за освітньо-професійною програмою покажемо у форматі Бізнес-профіль Еріксона-Пенкера (рис. 5.1).

Бізнес-профіль Еріксона-Пенкера [7] відображає основні сутності складових переддипломної практики та відношення між ними. Такий профіль дозволяє створити узагальнене представлення організації виконання практики. Для створення конкретних моделей виконання практики за індивідуальними програмами практики на основі такого профілю необхідно цей профіль розширити, наприклад за рахунок специфікації ресурсів [8].

Підкреслимо, що бізнес-профіль представляє узагальнену, уніфіковану модель організації практики бакалаврів. Відкритим залишається питання

побудови моделей окремих класів сутностей з яких складається бізне-профіль.

Таким чином уніфікована модель організації і змісту переддипломної практики бакалаврів на основі бізнес-профіля Еріксона – Пенкера є множиною класів сутностей практики і відношень між ними необхідних і достатніх для практичної реалізації конкретної переддипломної практики бакалаврів на основі індивідуальної програми практики.

У контексті змісту бізнес-профіля Еріксона – Пенкера, для його практичного застосування, **уніфікована модель організації і змісту переддипломної практики бакалаврів** складається з таких основних класів сутностей, як *Проблема, Мета, Процес, Зміна стану, Ресурс, Подія і Бінес правило* проходження практики,

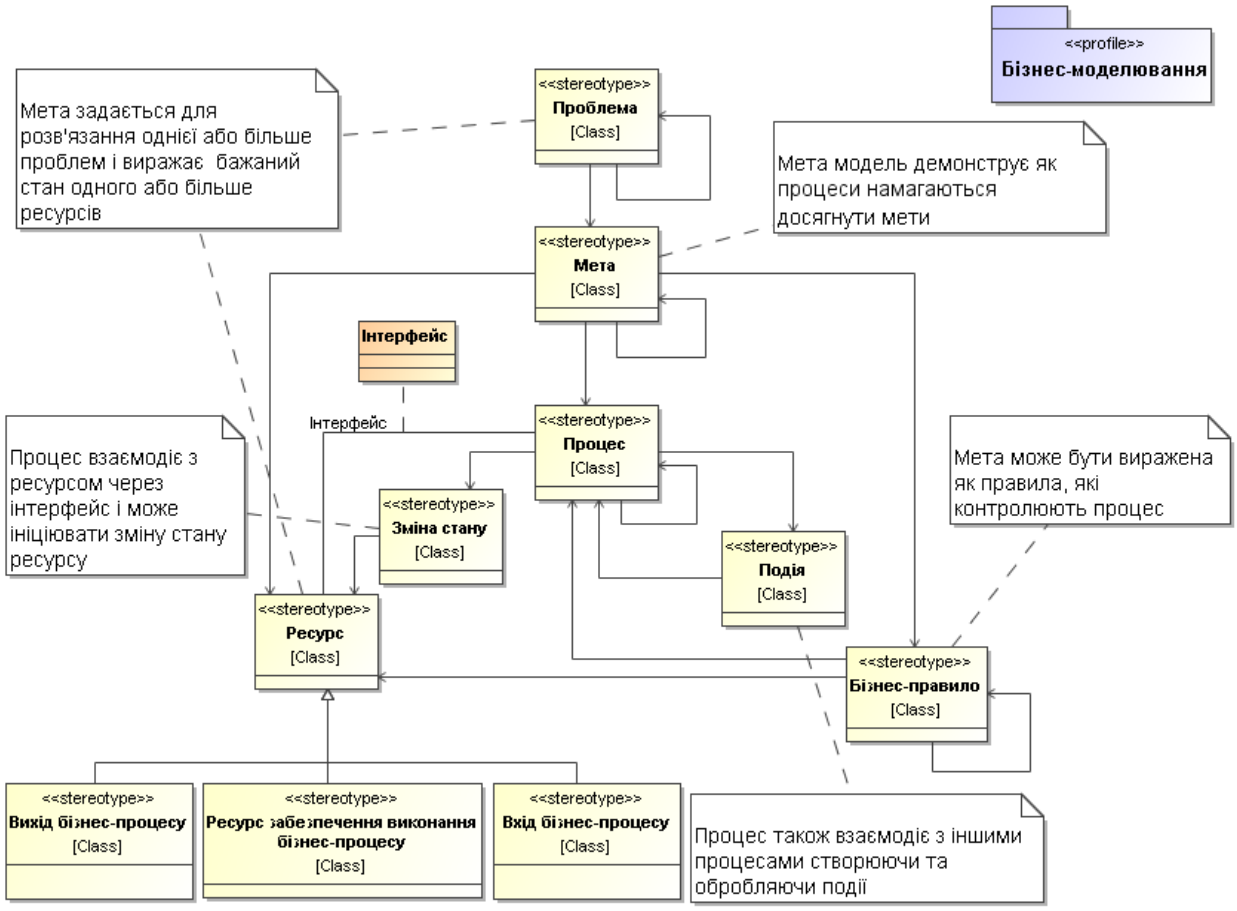


Рис. 5.1 - Вдосконалений бізнес-профіль Еріксона-Пенкера.
Діаграма класів в нотації UML [5, 8]

.

Формалізуємо зміст кожного з класів сутностей діаграми у термінах постановки задачі представлення уніфікованої моделі системи проходження практики, а саме таких класів діаграми рис. 5.1.

1. *Клас Проблема* (актуальне питання, що потребує вирішення, основна мотивація необхідності мети вирішення поставленого питання).

Проблема цієї роботи: необхідність формалізованого представлення структури, змісту і процесів проходження практики на основі ринкових категорій та індивідуальної програми практики для кожного студента, а саме:

- різноманітність університетської та поза університетської освіти для отримання знань і умінь;

- потребу високотехнологічних галузей у міждисциплінарних креативних рішеннях;

- суттєве посилення ролі «умінь» поряд із «знаннями» та кращі університетські та корпоративні практики продукування «умінь»;

- дистанційну, колективну форму виконання проектів та неперервну звітність результатів (релізів) роботи;

- власний багаторічний досвід і досвід колег IT індустрії з системної інженерії проектів інформатизації організаційних систем.

– *Клас Мета* (відображає власне всі категорії, що відповідають на питання: «Що конкретно потрібно розробити і що дає користувачам і споживачам результат застосування БКР, покликану розв’язати поставлену проблему?»).

Мета цієї роботи: розробити індивідуальну модель структури, змісту і процесів проходження практики на основі індивідуальної програми практики та надати теоретичні і прикладні інструменти для реалізації завдань і формування бакалаврської кваліфікаційної роботи, показати приклади застосування теоретичних і прикладних інструментів проходження практики.

2. *Клас Процес* (множина процесів діяльності системи, внаслідок якої досягають мети, чітко визначена послідовність дій/підпроцесів, що призводить до виконання певного завдання).

Процесом проходження практики є Дорожня карта проходження практики: навчально-методичні заходи практики, науково-дослідні заходи практики, конкретні індивідуальні завдання практики.

3. *Клас Зміна стану* (можливі зміни певних ресурсів унаслідок роботи процесів.

Модель проходження практики передбачає три стадії зміни станів:

- *Навчально-методичні заходи практики;*
- *Науково-дослідні заходи практики;*
- *Виконання конкретних завдань практики.*

4. *Клас Ресурс* (будь-які сутності (матеріальні чи нематеріальні), що їх споживає, використовує та продукує *Модель проходження практики*).

Для *Моделі проходження практики* це теоретичні та інструментальні ресурси, потрібні для реалізації конкретної БКР.

5. Ресурси найнижчого рівня ієрархії, що беруть безпосередню участь у процесах, також поділяють за характером впливу на перебіг процесів на такі три класи:

- *Клас Вихід бізнес-процесу* (ресурси, що їх продукує *Модель проходження практики*, кінцевий результат її функціонування).

До них належать звіт з практики, щоденник, захищений звіт з практики, тобто повний перелік документів і результатів практики;

– *Клас Ресурс забезпечення виконання бізнес-процесу* (ресурси, що забезпечують виконання процесів, але не є кінцевим результатом роботи).

Для Моделі проходження практики це люди (студенти, керівники практики, консультанти, наукові керівники, інші посадові особи), організаційні одиниці (кафедра, підприємство, установа, організація, підрозділ, тощо, дотичні до процесу проходження практики), систематизовані Знання, Уміння та множина виробничих і обчислювальних

ресурсів необхідна та достатня для продукування класу Вихід бізнес-процесу;

– *Клас Вхід бізнес-процесу* (первинні ресурси входу початкових процесів, які ініцалізують цикл роботи *Моделі проходження практики*).

Для *Моделі проходження практики* це Формалізація постановки задачі БКР, дані, інформація і знання щодо предмету дослідження/розробки для кожної стадії Дорожньої карти проходження практики;

6. *Клас Подія* (виникає через певні зовнішні фактори чи як результат взаємодії між процесами).

Для Моделі проходження практики це будь які події що виникають в результаті виконання завдань Дорожньої карти практики, початок, кінець виконання, результат виконання завдань.

7. Клас Бізнес-правило (формальні інструкції, що регулюють, обмежують, встановлюють контекст і рамки функціонування процесів проходження практики.)

Для Моделі проходження практики це необхідна множина нормативних правил, умов і вимог до проходження практики, до класів сутностей уніфікованої моделі Моделі проходження практики та відношень між ними.

Таким чином, на основі бізнес-профіля Еріксона-Пенкера, ми формалізували структурне представлення уніфікованої *Моделі проходження практики*, описали і сутності системи, і відношення між ними.

Ця модель має як теоретичне так і прикладне застосування.

Теоретичне застосування полягає у розумінні всіх сутностей уніфікованої *Моделі проходження практики,* формалізації і документування їх змісту та відношень між ними.

Прикладне застосування полягає в усвідомленому застосуванні уніфікованої *Моделі проходження практики* їх змісту для реалізації індивідуальної програми практики, проведення наукових досліджень/розробок за індивідуальними темами БКР.

А починається прикладне застосування з систематизації накопичених студентами знань. За означенням [20] систематизація є форма дослідження об’єктів, процесів і явищ на основі певної об’єднуючої ідеї або гіпотези, результатом якої є побудова упорядкованої системи сутностей і знань про них.

Власне такою об’єднуючою ідеєю і виступає бізнес-профіль Еріксона-Пенкера, що дозволяє інтегрувати розрізнену множину сутностей практики в уніфіковану *Модель проходження практики*.

Проведемо таку систематизацію у форматі Матриці Сутностей уніфікованої *Моделі проходження практики* (МС УМ ПП) для формування поточного стану розробки індивідуального бізнес-профіля наукових досліджень/розробок за темою БКР, де показані приклади можливої реалізації сутностей уніфікованої моделі (табл. 5.1).

МС УМ ПП систематизує всі ключові сутності і атрибути *Моделі проходження практики* та може бути розширена і доповнена у процесі реалізації Дорожньої карти проходження практики.

Звертаємо увагу і на необхідність зазначати активні посилання на всі необхідні для роботи інформаційні ресурси, що суттєво полегшує поточну операційну роботу проходження практики і підготовки БКР.

Така МС УМ ПП є досить зручним інструментом наглядного відображення ключових, поточних результатів проходження практики, інструментом самоконтролю і звітності.

Таблиця 5.1 Матриця Сутностей уніфікованої моделі проходження практики для формування поточного стану індивідуального бізнес-профіля проходження практики за темою БКР студент гр. КМ-хх,уу,zz 4-й курс 23/24, v 0.x

Сутності та атрибути УМС П	Опис сутності уніфікованої моделі проходження практики	Примітки (E-mail:, Tel.: , бібліографія [ ], посилання https://..., інше)
1	2	3
№ за списком / ПІБ студента	*ХХ / ПІБ студента (повністю)*	№ за списком / ПІБ студента
Н/керівник ПІБ	*Вказати повний ПІБ н/керівника*	Н/керівник ПІБ
Назва н/досліджень		Назва н/досліджень
(назва н/досл. за темою БКР)	Вказати Назву н/досліджень (назва н/досл. за темою БКР)	(назва н/досл. за темою БКР)
Клас Формалізація постановки задачі наук. досліджень/розробок: 1. Об'єкт досл., 2. Предмет досл. 3. Мета проведення конкретного н/досл. 4. Кінцевий результат н/досл./розробки.	*Вказати Об'єкт, Предмет, Мету проведення конкретного н/досл.* *Запланований кінцевий результат н/досл.*	Клас Формалізація постановки задачі наук. досліджень: 1. Об'єкт досл., 2. Предмет досл. 3. Мета проведення конкретного н/досл. 4. Кінцевий результат н/досл.

Продовження Таблиця 5.1

1	2	3
Клас Ресурси/теоретичні, потрібні для реалізації конкретного н/досл./роботи (Методи, моделі, алгоритми. Назви математичних дисциплін прослуханих в університеті та/ або за його межами і назви конкретних математичних інструментів цих дисциплін, тощо, потрібні для реалізації конкретного н/досл)	*Вказати Ресурси/теоретичні, потрібні для реалізації конкретного н/досл./роботи:* - *метод;* - *методологія;* - *Модель;* - *Процес;* - *Алгоритм;* - *Спосіб;* - *Система;* - *підсистема;* - *програмний модуль;* - *програмний продукт;* - *математичне та програмне забезпечення;* - *логістична модель;* - *функціональна модель;* - *динамічна модель;*	

Продовження Таблиця 5.1

1	2	3
	- класифікатор; *- тощо, з подальшим уточненням до чого/для чого, власне, цей ключовий результат відноситься.* ***Назви математичних дисциплін прослуханих в університеті та/або за його межами і назви конкретних математичних інструментів цих дисциплін, тощо, необхідні для реалізації БКР**:* *- Дискретна математика;* *- Алгоритми і структури даних;* *- Математична логіка та теорія алгоритмів;* *- Теорія ймовірності;* *- Криптографічні методи захисту інформації;* *- Алгоритмічні основи обчислювальної геометрії та комп`ютерної графіки;*	

Продовження Таблиця 5.1

1	2	3
	- Програмне забезпечення та інструментальні засоби мультимедіа; *- Математичні основи комп'ютерної графіки та мультимедіа;* *- Математична статистика;* *- Методи обчислень;* *- Автоматизоване тестування програмного забезпечення;* *- Методи оптимізації;* *- Рівняння математичної фізики;* *- Бази даних та інформаційні системи;* *- Системи Data Science;* *- Розподілені і хмарні обчислення;* *- Аналіз даних;* *- Математичне моделювання;*	

Продовження Таблиця 5.1

1	2	3
	- Методи штучного інтелекту; *- Алгоритми і системи комп'ютерної математики;* *- Системний аналіз;* *- Теорія керування;* *- Методи теорії надійності та ризику;* *- Моделювання складних систем;* *- Чисельні методи математичної фізики;* *- Машинне навчання;* *- Інтелектуальний аналіз даних;* *- Технологія Blockchain;* *- Математичне моделювання біомедичних систем і процесів;* *- Психологічні моделі прийняття управлінських рішень;*	

Продовження Таблиця 5.1

1	2	3
	- Нечітка математика; *- Архітектура та технології систем з великими обсягами даних;* *- Системна інженерія* *- Архітектура обчислювальних систем;* *- Аналіз даних;* *- Машинне навчання;* *- Математична статистика;* *- Моделювання складних систем;* *- тощо.*	
Клас Ресурси/засоби, потрібні для реалізації конкретного н/досл.	***Вказати Ресурси/засоби, потрібні для реалізації конкретного н/досл.:*** *- обчислювальний ресурс;* *- математичне, програмне, методичне, та інші необхідні види забезпечення;*	

Продовження Таблиця 5.1

1	2	3
	- *Matchad;* - *Matlab;* - *RStudio;* - *Бібліотеки ML;* - *Бібліотеки DS;* - *Бібліотеки BD;* - *Dataset[name];* *Тощо.*	
Клас Процеси реалізації конкретного н/досл. *(Водоспадний, спіральний, ітеративно-інкрементний, тощо)*	*Вказати Процеси реалізації конкретного н/досл. Із застосуванням до конкретного н/дослідження.* Процесом уніфікованої моделі системи «практика» є Дорожня карта проходження практики: Навчально-методичні заходи практики, Науково-дослідні заходи практики, Конкретні завдання практики.*(Водоспадний, спіральний, ітеративно- інкрементний, тощо)*	

Кінець Таблиця 5.1

1	2	3
Клас Бізнес-правила реалізації конкретного н/досл. (Технічні умови, технічні вимоги, календарний план виконання, тощо.)	*Вказати Бізнес-правила реалізації конкретного н/досл. Технічні умови, технічні вимоги до кінцевого результату н/досл та процесів реалізації (Призначення, область застосування, функціональність, вимоги до верифікації і валідації, повна вартість володіння, тощо), календарний план виконання, тощо.*	*Клас Бізнес-правила реалізації конкретного н/досл. (Технічні умови, технічні вимоги, календарний план виконання, тощо.)*
Клас Подія (виникає через певні зовнішні фактори чи як результат взаємодії міжпроцесами).	*Для уніфікованої моделі системи «Практика» це будь які події що виникають в результаті виконання завдань Дорожньої карти практики, початок, кінець виконання, результат виконання завдань.*	*Клас Подія (виникає через певні зовнішні фактори чи як результат взаємодії міжпроцесами).*
Інші суттєві сутності і атрибути		

Зупинимось дещо детальніше на сутності «Процес» *Моделі проходження практики*. Як було вже встановлено, процесом *Моделі проходження практики* є Дорожня карта проходження практики, зокрема навчально-методичні заходи і практики, науково-дослідні заходи, конкретні завдання практики і можливі моделі реалізації процесу, як то водоспадний, спіральний, ітеративно- інкрементний, тощо.

Далі покажемо деякі суттєві особливості найбільш популярних моделей процесів серед яких студенти можуть самостійно обирати ту, чи іншу модель процесу для максимально ефективної організації проходження переддипломної практики.

5.1. Процеси ітеративні і водоспадні

У загальному вигляді процес розробки (development process) визначається як упорядкований, достатній і функціонально повний ряд робіт з розробки проектів інформатизації організаційних систем із заданими показниками ефективності. Тут і надалі, під визначенням “організаційна система (Орг.С)” будемо розуміти множину сутностей (мету діяльності, бізнес-процеси, бізнес-правила, персонал і ресурси), необхідну і достатню для проведення певної діяльності [21].

Як критерій поділу процесів розробки проектів інформатизації на водопадні (послідовні) та ітеративні береться критерій поділу проекту на частини [19, 21].

Так при організації робіт з розробки проектів інформатизації за типом водопаду проект поділяється на основі виду робіт. Це фактично мережевий графік окремих робіт з виконання проекту інформатизації, який передбачає послідовне їх виконання. Однак водопадний процес має певні недоліки: що в результаті виконання чергового етапу або фази проекту виконавець повинен передати замовнику повністю закінчену частину проекту інформатизації, що не завжди вдається зробити, особливо при виконанні середніх і великих

проектів. Як правило, при проведенні подальших етапів з’являється необхідність в доопрацюванні попередніх етапів, а для цього потрібні додаткові кошти, не передбачені бюджетом проекту інформатизації.

При застосуванні ітеративного стилю проект поділяється на частини на основі функціональності проекту інформатизації. На першому етапі виконання проекту шляхом кількох ітерацій розробки забезпечується зафіксована частина функціональності проекту. Одна ітерація передбачає виконання аналізу, проектування, реалізацію і розгортання. Одна або кілька послідовних ітерацій на першому етапі дозволяє отримати першу версію проекту інформатизації із зафіксованою частиною функціональності. На наступних етапах проводиться повний аналіз результатів, вносяться поправки і зміни, додаються нові функції, і процес аналізу, проектування, реалізації і розгортання повторюється до повного завершення проекту інформатизації.

Одна з істотних переваг ітеративного процесу перед водопадним полягає в можливості постійного контролю і вдосконалення процесу розробки на всіх етапах. При цьому на кожному етапі і замовник, і розробник отримують версію проекту інформатизації, мають змогу вносити в нього зміни і своєчасно приймати необхідні рішення.

На рис. 5.2 показано структурну схему ітеративного процесу і співвідношення фаз розробки, стадій моделювання та ітерацій розробки проектів інформатизації [18]. Тут стадії моделювання співвідносяться з ітеративним процесом розробки і його фазами: початком, розвитком, конструюванням і переходом. На кожному етапі виконання проекту всі ці фази ідуть одна за одною, при цьому кожна з них вміщує в собі від однієї до кількох ітерацій. У кожній ітерації окремі частини проекту проходять шлях від аналізу до розгортання з різною швидкістю. Ступінь завершеності етапу визначається встановленою функціональністю версії проекту.

Ітеративні процеси і їх сучасні модифікації на сьогодні домінують на світовому ринку процесів розробок інформаційно-комунікаційних систем.

Тим не менше, практика свідчить і про певні обмеження в застосуванні ітеративних процесів.

Рис. 5.2 Структурна схема ітеративного процесу і співвідношення фаз розробки, стадій моделювання та ітерацій розробки проектів [18].

Якщо умовно розділити проекти інформатизації Орг.С на три класи за критерієм поділу Орг.С на малі або локальні, середні або регіональні і великі або глобальні, то можна однозначно стверджувати, що ітеративні процеси практично в повному обсязі працюють при проектуванні малих або локальних проектів, працюють при певних, іноді істотних, обмеженнях і налаштуванні при проектуванні середніх або регіональних проектів та практично не працюють у класичному вигляді при проектуванні глобальних проектів інформатизації.

Аналіз особливостей застосування ітеративних процесів при виконанні цих класів проектів інформатизації показує, що:

- водоспадні та ітеративні процеси розробки були з самого початку зорієнтовані на розробку виключно програмних систем і, природньо, не враховували особливостей розробки і складових проектів інформатизації Орг.С в цілому;

- для середніх і великих проектів поділ проекту інформатизації за функціональною ознакою не завжди виправданий з економічної точки зору. Якщо чергова версія програмної системи може бути змінена і доопрацьована на наступному етапі без додаткових фінансових витрат, то закуплене апаратне забезпечення і розроблені загальносистемні рішення для реалізації визначеної групи функцій змінити і доопрацювати неможливо, його можна тільки замінити, що тягне за собою великі додаткові фінансові витрати;

- для середніх і великих проектів поділ проекту інформатизації за функціональною ознакою не передбачає виділення загальносистемних рішень, які потрібні для реалізації функціональних характеристик.

Таким чином, застосування ітеративних процесів для проектування середніх і великих проектів інформатизації потребує інших підходів та істотного вдосконалення. Одним із таких підходів може бути застосування системного аналізу і теорії систем для визначення іншого критерію поділу проектів інформатизації на частини не за функціональною ознакою, а за загальносистемними ознаками. В такому випадку прикладна функціональність базується на реалізації необхідних загальносистемних рішень.

Ітеративний процес із загальносистемним критерієм поділу проекту інформатизації на частини за функціональними ознаками забезпечує:

- поділ проекту на загальносистемні частини, які реалізують визначені групи функцій і, таким чином, забезпечують максимально обґрунтований поділ проекту на етапи;

- врахування всіх сутностей проекту інформатизації;

- масштабованість і розширюваність проекту від ітерації до ітерації в межах одного етапу та при переході від етапу до етапу;

- зберігання загальносистемних рішень попередніх ітерацій і етапів для подальших ітерацій і етапів;

- незалежність процесів розробки і паралельну розробку окремих частин проекту інформатизації;

- інтеграцію окремих частин проекту інформатизації при переході від однієї версії до іншої;

- паралельну розробку необхідної документації з усіх видів забезпечення.

Ідея застосування системного аналізу і теорії систем для визначення системного критерію поділу проектів інформатизації на частини не за функціональною ознакою, а за загальносистемними ознаками передбачає формування відповідного документа і моделі проекту інформатизації. Таким документом може бути загальносистемна специфікація проекту інформатизації.

Тут, під визначенням "специфікація" ми розуміємо формалізований опис властивостей, характеристик і функцій об'єктів, компонентів, пакетів та способів і правил їх взаємодії в системі. Загальносистемна специфікація на весь проект інформатизації є власне моделлю системи інформатизації Орг.С. Така системна модель проекту інформатизації ще називається системним проектом.

Метасутності представлення системного проекту інформатизації в нотації UML наведено в таблиці 5.2 [18].

Представлення системної моделі проекту – це підмножина конструкцій графічної мови моделювання UML, яка відображає один із вибраних аспектів проекту інформатизації.

Метасутності відображають такі групи представлення системного проекту інформатизації:

- структурне представлення, яке описує системні сутності та їх співвідношення;

- динамічну поведінку, яка описує поведінку проекту інформатизації в часі;

- фізичне розміщення сутностей інформатизації в просторі;
- представлення керування системним проектом через поділ проекту на ієрархічні блоки – пакети.

Структурне представлення описує системні сутності та їх співвідношення. До структурного представлення входять статичне представлення, представлення проектування і представлення Use Case. Концепцію сутності в системі моделює класифікатор, який є окремою концепцією, що описує сутності проектів інформатизації, їх властивості, стан, поведінку, співвідношення, а також внутрішню структуру. До числа класифікаторів графічної мови моделювання UML належать класи, пакети і компоненти, інтерфейси, елементи Use Case, актори, вузли та кооперація.

Таблиця 5.2 Метасутності представлення і діаграми системного проекту в нотації UML [18].

Метасутність представлення систем	Представлення	Діаграма	Основна концепція
1	2	3	4
Структурна	Статичне представлення	Діаграма класів	Клас, асоціація, узагальнення, залежність, реалізація, інтерфейс
	Представлення проектування	Внутрішня структура	З'єднувач, інтерфейс, частина, порт, забезпечений інтерфейс, роль, інтерфейс на вимогу
		Діаграма кооперації	З'єднувач, кооперація, використання кооперації, роль
		Діаграма компонентів	Компонент, залежність, порт, забезпечений інтерфейс, інтерфейс на вимогу, підсистема
	Представлення Use Case	Діаграма Use Case	Актор, асоціація, розширення, включення, елемент Use Case, узагальнення елемента Use Case
Динамічна	Представлен	Діаграма	Завершення переходу,

	ня скінченних автоматів	автомата	здійснення діяльності, ефект, подія, область, стан, перехід, тригер
	Представлення діяльності	Діаграма діяльності	Дія, діяльність, потік керування, вузол керування, потік даних, виключення, область розширення, поділ, злиття, об’єктний вузол, контакт

Продовження Таблиця. 5.2

1	2	3	4
	Представлення взаємодії	Діаграма послідовності	Специфікація входження, специфікація виконання, взаємодія, фрагмент взаємодії, операнд взаємодії, лінія життя, повідомлення, сигнал
Фізична	Представлення розгортання	Діаграма розгортання	Артефакт, залежність, маніфестація, вузол
Керування моделлю	Представлення керування моделлю	Діаграма пакетів	Імпорт, модель, пакет
	Профіль	Діаграма пакетів	Обмеження, профіль, стереотип, тегова величина

Динамічне представлення описує динамічну поведінку системи або окремого класифікатора в часі, тобто зміну стану системних сутностей у часі. Представлення динамічних моделей системи вміщує в собі скінченні автомати, представлення діяльності і представлення взаємодії.

Фізичне розміщення сутностей описує розміщення всіх ресурсів системи і розгортання на них артефактів. Фізичне розміщення описується представленням розгортання.

Представлення керування системним проектом відображає внутрішню організацію моделей системи з різних точок зору. Воно об’єднує всі

необхідні представлення в єдину структуру, яка забезпечує керованість і спостереження за ходом виконання проекту інформатизації.

Для розробки специфікації проекту інформатизації найважливішими класифікаторами, які описують взаємодію всіх інших сутностей, є пакет, клас, компонент та інтерфейс.

Метамодель представлення ітеративного процесу виконання проекту інформатизації Орг.С на всіх етапах життєвого циклу у вигляді діаграми пакетів зображено на рис. 5.3 та рис.5.4.

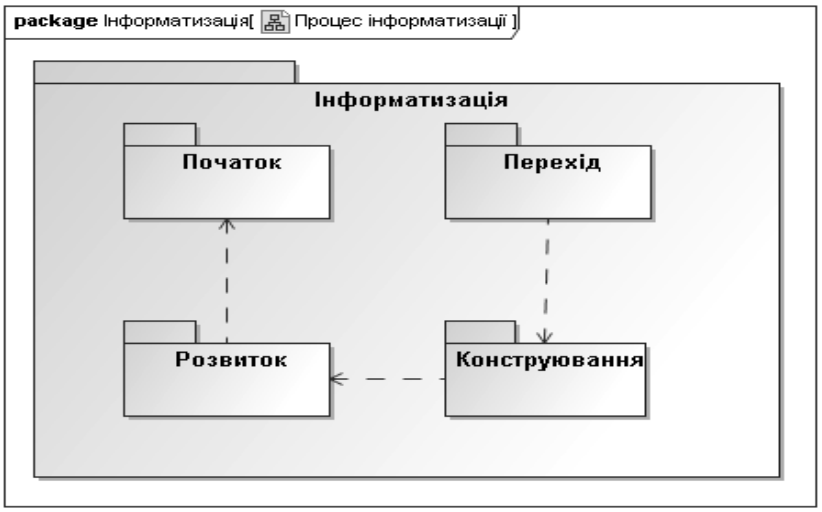


Рис. 5.3 Метамодель представлення ітеративного процесу виконання проекту інформатизації Орг.С. Діаграма пакетів у нотації UML [21].

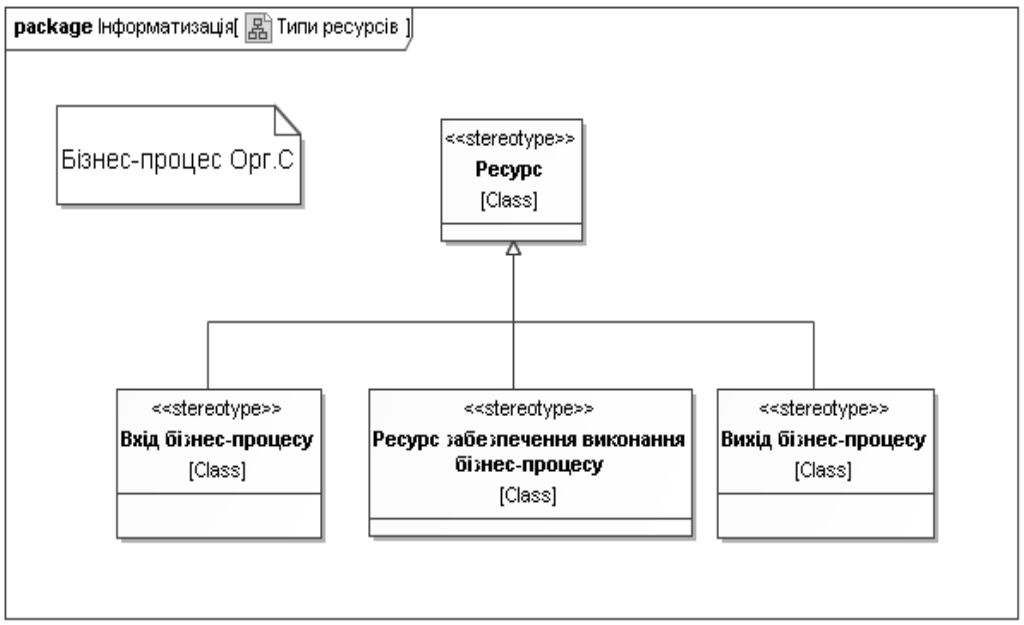


Рис. 5.4 Метамодель ітеративного процесу інформатизації Орг.С залежно від типу ресурсів. Діаграма класів у нотації UML [21].

Для наочності, на рис. 5.5 продемонструємо практичне застосування такого підходу для представлення процесу «Дорожня карта проходження практики» системи «практика».

На цьому рисунку схематично показано ітеративно – інкрементний процес реалізації класів завдань усіх стадій життєвого циклу системи практика»

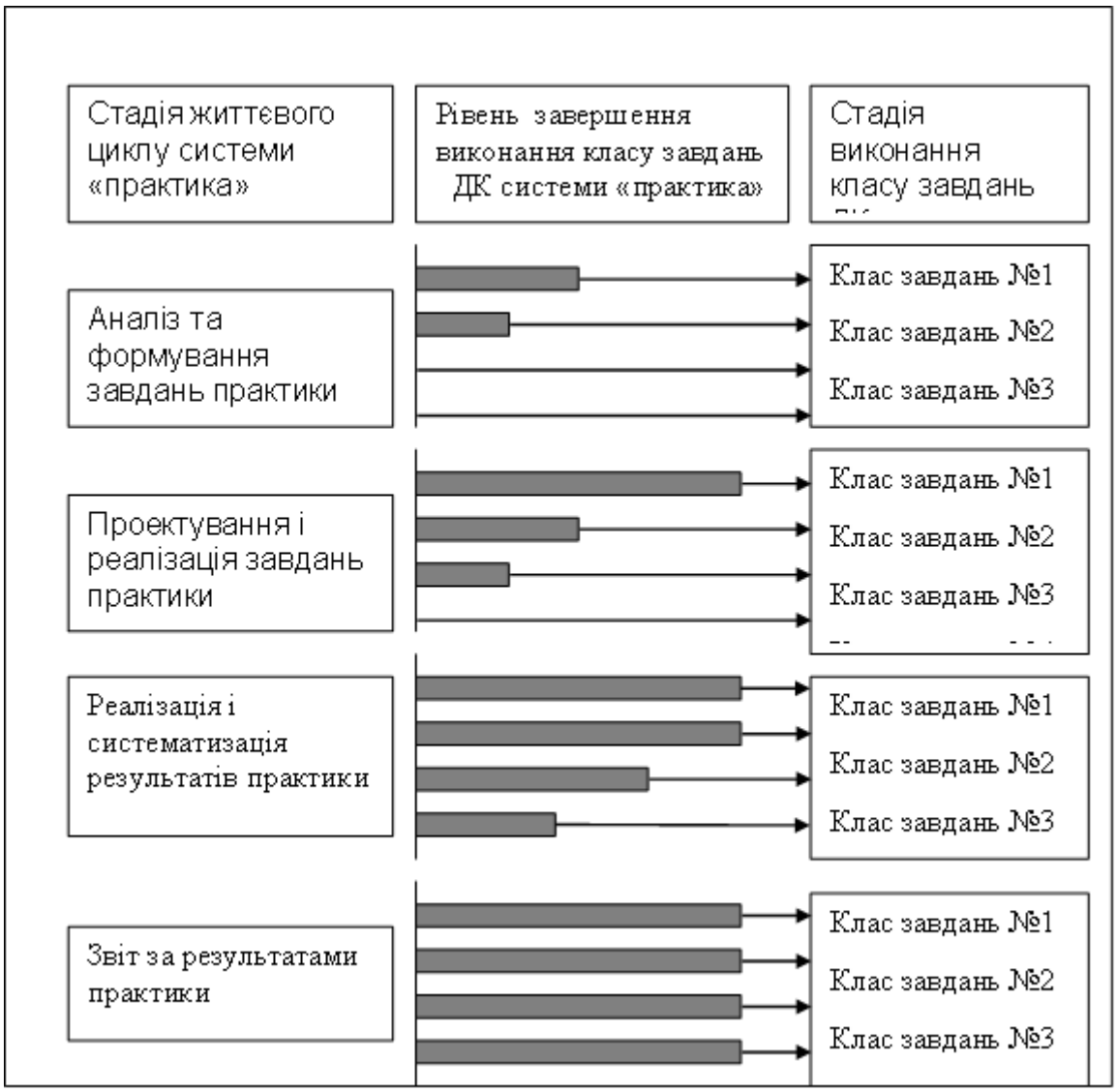


Рис. 5.5. - Структурна схема ітеративного процесу і співвідношення стадій життєвого циклу системи «практика», стадій виконання класу завдань та ітерацій виконання класів завдань ДК Моделі проходження практики.

Наведемо основні висновки щодо можливості застосування проаналізованих класів моделей процесів розробки для реалізації процесу Дорожня карта Моделі проходження практики [21].

1. Ідея застосування системного аналізу і теорії систем для визначення системного критерію поділу проектів інформатизації на частини не за функціональною ознакою, а за загальносистемними ознаками дозволяє істотно вдосконалити ітеративний процес інформатизації Орг.С і забезпечити

представлення проекту інформатизації як системи всіх сутностей і відношень між ними. Таке представлення формується у вигляді загальносистемної специфікації проекту інформатизації.

Тут під визначенням “специфікація” розуміється формалізований опис властивостей, характеристик і функцій об’єктів, компонентів, пакетів та способів і правил їх взаємодії в системі.

Загальносистемна специфікація на весь проект інформатизації є власне моделлю системи інформатизації Орг.С.

2. Використання системного аналізу і теорії систем для визначення системного критерію поділу проектів інформатизації на частини не за функціональною ознакою, а за загальносистемними ознаками дозволяє застосувати ітеративний процес для реалізації середніх і великих проектів інформатизації Орг.С.

3. Ітеративний процес із загальносистемним критерієм поділу проекту інформатизації на частини забезпечує:

- поділ проекту на загальносистемні частини, які реалізують визначені групи функцій і, таким чином, забезпечують максимально обґрунтований поділ проекту на етапи;
- врахування всіх сутностей проекту інформатизації;
- масштабованість і розширюваність проекту від ітерації до ітерації в межах одного етапу та при переході від етапу до етапу;
- зберігання загальносистемних рішень попередніх ітерацій і етапів для наступних ітерацій і етапів; незалежність процесів розробки і паралельну розробку окремих частин проекту інформатизації;
- інтеграцію окремих частин проекту інформатизації при переході від однієї версії до наступної;
- паралельну розробку необхідної документації з усіх видів забезпечення.

4. Ітеративний процес із загальносистемним критерієм поділу проекту інформатизації на частини не накладає практично ніяких системних

обмежень на застосування систем автоматизації проектування інформаційно-комунікаційних систем, які підтримують UML.

Таким чином вже на попередніх стадіях роботи над БКР можна максимально ефективно організувати процес «Дорожня карта Моделі проходження практики» за рахунок обґрунтованого вибору моделі процесу розробки.

Список посилань до розділу

1. Згуровський М.З., Панкратова Н.Д. Системний аналіз: проблеми, методологія, застосування. – К.: Наук. думка, 2005. – 743 с.

2. Агошкова Е.Б., Ахлибининский Б.В. Эволюция понятия системы // Вопр. философии. – 1998. – № 7. – С. 170–179

3. Booch G., Jacobson I., Rumbaugh J. The Unified Modeling Language User Guide. – 2nd ed. // Addison-Wesley Professional. – 2005. – P. 496.

4. Липаев В.В. Программная инженерия. Методологические основы. – М: “ТЕИС”, 2006. – С. 608.

5. Маслянко П.П., Майстренко О.С. Бізнес інжинірінг організаційних систем // Наукові вісті НТУУ “КПІ”. – 2011. – № 1. – С. 69-78.

6. Маслянко П.П., Майстренко А.С. Система сущностей бизнес моделей организационных систем // Кибернетика и системный анализ. – 2012. – № 1. – С. 118-128.

7. Penker M., Ericsson H.-E. Business Modeling with UML: Business Patterns at Work. – Wiley, 2000. – 480 p.

8. Маслянко П.П., Майстренко О.С. Системна інженерія проектів інформатизації організаційних систем // Наукові вісті НТУУ “КПІ”. – 2008. – № 6. – С.:34-42.

9. VVernadat F. UEML: towards a unified enterprise modelling language // Actes 3ème Conférence Francophone de Modélisation et Simulation - MOSIM'01 (2001). – 2001. – P. 3-10.

10. Майстренко, О. С. Моделювання організаційних систем на основі бізнес-профіля / О. С. Майстренко, П. П. Маслянко // Компьютерное моделирование в наукоемких технологиях, КМНТ-2012. — 2012. — С. 281-284.

11. Uschold M., King M., Moralee S., Zorgios Y. The enterprise ontology // The Knowledge Engineering Review. – 1998. – № 13. – P. 31-89.

12. Fox M.S., Chionglo J.F., Fadel F.G. A common-sense model of the enterprise // In Proceedings of the Second Industrial Engineering Research Conference. Norcross Ga.: Institute for Industrial Engineers. – 1993. – P. 425-429.
13. Jonkers H., Lankhorst M., Van Buuren R., Bonsangue M., Van Der Torre L. Concepts for modeling enterprise architectures // International Journal of Cooperative Information Systems. – 2004. – № 13. – P. 257-287.
14. Sousa P., Caetano A., Vasconcelos A., Pereira C., Tribolet J. Enterprise architecture modeling with the unified modeling language // Business Information Systems: Concepts, Methodologies, Tools and Applications. – 2005. – № 4. – P. 719-742.
15. Winter R., Fischer R. Essential layers, artifacts, and dependencies of enterprise architecture // Journal of Enterprise Architecture. – 2007. – 3, № 2. – P. 7-18.
16. Zachman J.A. A framework for information systems architecture // IBM Systems Journal. – 1987. – 26, № 3. – P. 276-292.
17. OMG BPMN Specification. Version 2.0. OMG Document Number: formal/2011-01-03. Standard document URL: hhttp://www.omg.org/spec/BPMN/2.0/
18. Rumbaugh J., Jacoson I., Booch G. The Unified Modeling Language Reference Manual, Second Edition, by Pearson Education, Inc.
19. Fowler M. UML Distilled: A Brief Guide to the Standart Objekt Modeling Language, Third Edition. – Copirigth, 2004. – 200 с.
20. П. Йолон // Філософський енциклопедичний словник / В. І. Шинкарук (гол. редкол.) та ін. — Київ : Інститут філософії імені Григорія Сковороди НАН України : Абрис, 2002. — 742 с. — 1000 екз. — ББК 87я2. — ISBN 966-531-128-X.
21. Маслянко П.П. Системне проектування процесів інформатизації. // Наукові вісті НТУУ “КПІ”. - 2008.- № 1. С.28-36.

6. ІНДИВІДУАЛЬНЕ ЗАВДАННЯ ПРАКТИКИ

Індивідуальне завдання переддипломної практики, - це упорядкований перелік конкретних класів організаційних завдань та завдань продукування змістовної частини документів звіту з практики, що отримує кожен студент перед проходженням практики у відповідності до робочої програми практики.

Індивідуальне завдання практики формує керівник БКР і керівник практики від університету на основі знань і умінь накопичених студентом на момент проходження практики та на основі формалізації постановки задачі бакалаврської атестаційної роботи (ФПЗ БКР). Індивідуальне завдання записується у щоденник практики і формалізується у вигляді конкретної Моделі проходження практики та МС УМ ПП за темою БКР.

У розділі 3 ми вже визначили мету і чотири класи завдань практики у загальному вигляді. Виконання кожного класу завдань закінчується проведенням поточного контролю, а виконанням завершального четвертого класу завдань – календарним контролем (заліком). Звертаємо увагу студентів на перелік документів для проходження поточного та календарного контролю виконання класів завдань практики.

6.1. Клас завдань 1. Завдання формування і систематизації ресурсів для проходження практики

Для організації процесу проходження переддипломної практики необхідно визначитись з темою бакалаврської атестаційної роботи, сформулювати назву БКР та обрати наукового керівника БКР згідно з педагогічним навантаженням поточного семестру. Студент може продовжити роботу с призначеним йому науковим керівником на молодших курсах або змінити його, попередньо домовившись з відповідним викладачем. Для закріплення

теми та наукового керівника студент подає заяву встановленого зразка на ім’я завідувача кафедри з попередньою назвою теми БКР і проханням призначити наукового керівника (Додаток 3).

6.1.1. Формалізація постановки задачі бакалаврської атестаційної роботи

Ключовим, початковим завданням і ключовим результатом виконання **Завдань класу 1** є формалізація постановки задачі БКР – визначення об’єкта, предмета, мети дослідження/розробки та кінцевого результату виконання БКР.

Формалізація постановки задачі БКР (ФПЗ БКР) розробляється на основі обраної теми БКР, або назви обраного напряму чи області наукових інтересів студента, що відповідають спеціалізації кафедри прикладної математики за спеціалізацією «Наука про дані та математичне моделювання».

Розробка ФПЗ БКР є надзвичайно важливим етапом підготовки БКР і від того наскільки буде усвідомлена суть, способи реалізації та кінцевий результат роботи залежить і якість виконання БКР.

На підтвердження цієї тези процитуємо лише декілька думок з цього приводу від авторитетних мислителів і вчених, (курсив наш).

Так, за виразом Конфуція, древнього китайського мислителя і філософа, «Якщо почитати з неправильного, то мало надії на правильне завершення».

А на думку К. Норберг-Шульца «Тільки при повному розумінні задач можна знайти відповідні способи їх розв’язку. Для результатів важливіше поставити правильні питання, чим правильно відповісти на помилкові питання».

Тому, тема БКР, або назва обраного напряму чи області наукових інтересів, повинна, як правило, починатись з назви кінцевого, основного результату виконання завдань БКР *(метод, методологія, модель, процес,*

алгоритм, спосіб, системний аналіз, система, підсистема, програмний модуль, програмний продукт, застосунок, математичне та програмне забезпечення, логістична модель, функціональна модель, динамічна модель, класифікатор, тощо), з подальшим уточненням методів реалізації, та до чого/для чого, власне, цей ключовий результат відноситься.

Зрозуміло, що Тема БКР, або назва обраного напряму чи області наукових інтересів формується не одразу, носить ітеративний характер і може бути уточнено у процесі виконання БКР та отриманих результатів.

Таким чином формалізація постановки задачі БКР передбачає підготовку і формальне представлення системи ключових сутностей, що достатньо повно відображають суть БКР при заданих умовах і вимогах. Це, зокрема, такі сутності як об'єкт дослідження, предмет дослідження, мета дослідження і кінцевий результат дослідження.

Дамо авторське визначення цих сутностей і особливості їх застосування.

Визначення 1. Об'єкт дослідження, - це множина вже існуючих сутностей визначеної області досліджень, що описують предметну область з заданою точністю і повністю, або частково вирішує задекларовані у БКР задачі.

Об'єкт дослідження, як правило, складається з двох частин: переліку існуючих теоретичних інструментів і переліку практичних засобів, що повністю, або частково вирішують поставлені завдання БКР.

Застосування визначення. Повне розкриття переліку сутностей об'єкту досліджень на вербальному рівні дозволяє зробити фаховий огляд і аналіз існуючих теоретичних рішень і практичних засобів для розділу БКР з умовною назвою «Огляд існуючих рішень за темою БКР».

Визначення 2. Предмет дослідження, - це виділена з множини сутностей об'єкту досліджень підмножина сутностей визначеної області досліджень та розробок, що є конкретним предметом досліджень та розробок, задекларованих у БКР.

Застосування визначення. Повне розкриття переліку сутностей предмету досліджень і розробок на вербальному рівні та у інших необхідних нотаціях дозволяє конкретизувати множину сутностей дослідження/розробки, встановити обмеження і правила та провести дослідження і отримати теоретичні та практичні результати у процесі вирішення завдань задекларованої теми БКР.

Визначення 3. Мета досліджень, - це досягнення вирішення всіх поставлених у БКР завдань і визначення конкретних наслідків та значення від їх застосування (наукових, економічних, фінансових, соціальних, технічних, тощо).

Застосування визначення. Правильно формалізована мета досліджень з встановленими границями і обмеженнями дозволяє отримати конкретні результати проведених досліджень.

Визначення 4. Кінцевий результат виконання завдань БКР, - це систематизовані теоретичні результати та/або практичні засоби отримані в результаті виконання завдань БКР і формалізовані у формах наукової новизни, практичного значення отриманих результатів та висновків до БКР.

Застосування визначення. Кінцевий результат виконання завдань БКР дозволяє у повному обсязі оцінити рівень, повноту і адекватність наукових/ науково-технічних (прикладних) результатів виконання завдань БКР та ступінь досягнення мети досліджень/розробки.

Для демонстрації конкретної реалізації формалізації постановки задачі конкретної теми БКР покажемо два приклади. Назви розділів і підрозділів прикладів виділено курсивом.

Приклад 1. [Для прикладу використані фрагменти формалізації постановки задачі БКР студента Євгенія Буззінського на тему «Математичне та програмне забезпечення системи генерації зображень за допомогою методів глибокого навчання» започатковані в процесі виконання лабораторних робіт та вивчення дисципліни «Системи Data Science». Викладач дисципліни та науковий керівник проєкту Маслянко Павло Павлович, к.т.н., доцент»].

1. Об’єкт дослідження

Методи застосування операційних перетворень для побудови систем паралельного редагування контенту, моделі цілісності системи операційних перетворень: CC, CCI, CSM, CA моделі, алгоритми treeOPT, GOT, GOTO, AnyUndo, COT, властивості збіжності та оберненості функції трансформації, методи застосування синхронізації змінами для побудови систем паралельного редагування контенту, алгоритм тристороннього об’єднання (Three-Way Merge), Dual Shadow Method, Guaranteed Delivery Method, функції знаходження різниці між двома файлами, алгоритм знаходження найбільшої спільної підпослідовності, методи застосування безконфліктних реплікованих типів даних для побудови системи редагування контенту, технології передачі даних в комп’ютерних мережах, OSI модель, датаграми, потокова передача даних, протоколи прикладного рівня HTTP, WebSocket, методи забезпечення захищеності даних, що передаються в комп’ютерних мережах, шифрування протоколів транспортного рівня TLS.

Існуючі системи паралельного редагування контенту: Google Docs, Google RealTime API, Google Wave, CoVim, CoWord, DistEdit.

2. Предмет дослідження

Алгоритм реалізації операційних перетворень для систем паралельного редагування контенту з використанням функцій трансформації тексту та алгоритму рекурсивної трансформації операцій в мережевих ресурсах з використанням протоколів прикладного рівня WebSocket та шифрування протоколів транспортного рівня TLS. Моделі верифікації(перевірки адекватності) алгоритму. Порівняльний аналіз алгоритмів реалізації операційних перетворень для систем паралельного редагування контенту.

3. Мета роботи

Розробка математичного та програмного забезпечення системи паралельного редагування контенту в мережевих ресурсах для автоматизації і підвищення ефективності колективної роботи при реалізації проектів інформатизації організаційних систем.

4. Кінцевий результат та призначення роботи

Математичне та програмне забезпечення системи паралельного редагування контенту в мережевих ресурсах інформаційно-комунікаційних систем для підвищення якості та продуктивності роботи персоналу з реалізації паралельного редагування документів за рахунок зменшення витрат часу на опрацювання та формалізацію документів.

Наразі, для розкриття суті бакалаврської атестаційної роботи або бакалаврської атестаційної роботи, існує практика додавати до формалізації постановки задачі коротке і змістовне обґрунтування актуальності роботи та зміст і способи її виконання.

Дамо приклад такої розгорнутої формалізації постановки задачі.

Приклад 2, розгорнутий. [Для прикладу використані фрагменти формалізації постановки задачі МД і фрагменти МД студентки Юлії Білошапки започатковані в процесі вивчення дисциплін «Основи наукових досліджень» та «НДР за темою МД» і виконання МД на тему ««Математичне та програмне забезпечення системи автоматизованої підтримки користувачів компанії», викладач дисциплін та науковий керівник МД Маслянко Павло Павлович, к.т.н., доцент»].

У сучасному світі можна помітити суттєву перевагу застосування інтернету у сфері комерції, покупок, ведення бізнесу, та взагалі використання інформаційних технологій будь-якою організацією. Також можна відслідкувати, що коли клієнтам або кінцевим споживачам потрібна допомога або прохання допомоги у разі виникнення проблем, пов'язаних з користуванням продукції компаній, вони рідко коли фізично взаємодіють із обслуговуючим персоналом. Вони надають перевагу звернутись до центру

прийому дзвінків, веб-системи управління випусками продукції та подібні віртуальні системи, тому, більшість компаній намагаються обирати рентабельні та ефективні віртуальні системи підтримки клієнтів. З однієї сторони, втрата або мінімізація взаємодії людей у службі обслуговування клієнтів, негативно впливає на кінцеву задоволеність клієнтів. Проте, щоб подолати даний недолік, компанії намагаються покращити задоволеність клієнтів за рахунок кращої якості обслуговування послуг, швидкого реагування на клієнтські запити та вирішення питань з мінімальними процедурними кроками. Все це свідчить про те, що віртуальні системи підтримки клієнтів відіграють вирішальну роль в операціях із підтримки організації. До того ж, автоматизація таких систем з впровадженими алгоритмами передбачення, класифікації клієнтських звернень, а згодом з рекомендаційними алгоритмами пошуку найкращого і доцільного вирішення все більше цікавить свідомих власників компаній.

Тому проведення досліджень і виконання робіт з інформатизації бізнес-процесів взаємодії клієнтів з компаніями є досить актуальними.

***Об'єктом дослідження** є методи, моделі, способи, алгоритми, процеси та системи підтримки користувачів компаній, методологія опрацювання запитів користувачів про проблеми, види та канали звернень користувачів про проблеми, задоволення потреб користувачів.*

***Предметом дослідження** є методи та моделі бізнес аналізу для системного моделювання бізнес стратегій категоризації проблем користувачів на основі їх звернень до систем підтримки компанії для автоматичного їх вирішення.*

***Метою роботи** є розробка та оптимізація математичної моделі ідентифікації суті проблеми користувацьких звернень до систем підтримки компанії для автоматичної категоризації та призначення їх на кваліфіковані відділи для їх вирішення.*

Методи дослідження. *В роботі використовуються методи системного аналізу, бізнес-моделювання, аналізу даних, теорії систем, статистичні методи аналізу та прогнозування, методи машинного навчання*.

Кінцевий результат досліджень. *Математичне та програмне забезпечення системи автоматизованої підтримки користувачів компанії, методи та моделі бізнес аналізу для системного моделювання бізнес стратегій категоризації проблем користувачів на основі їх звернень до систем підтримки компанії для автоматичного їх вирішення*.

Ключові слова: *бізнес-моделювання, методи прийняття рішень, архітектура системи підтримки користувачів, класифікація тексту, категоризація звернень, рівні системи підтримки, автоматизація системи підтримки, методи класифікації для набору документів, точність класифікації, матриця помилок, однозначна багато класова класифікація, метод опорних векторів.*

Існує гіпотеза, яка передбачає, що автоматизація бізнесу витіснить людські ресурси чи знання, які працюють на бізнес. Дана гіпотеза є хибною, адже автоматизація лише полегшує працівникам зосередитися на більш інтуїтивних, інтелектуальних завданнях. Вона звільняє їх від відповідальності за адміністративні чи часто повторювані завдання та передбачає втручання людини лише в тих місцях, де вони не можуть бути замінені. Автоматизація можлива майже скрізь, але вона відіграє більшу роль, особливо в організаціях, де ресурси та знання є дуже цінними. Можна виокремити чотири переваги впровадження автоматизації для служб обслуговування клієнтів[2, 51с.]:

1. Відсутність часто повторюваних операцій. Оператори підтримки, як правило, переповнюються повторюваними запитами. Часто вони змушені відповідати на однакові питання, що згодом стає для них неприємним та немотивуючим. Автоматизована система, наприклад, виходячи з критичності чи пріоритетності запитів, спрямовуватиме

розподілення запитів до відповідного керівника служби підтримки, який має належний набір навичок та досвід у вирішенні певного виду проблем;

2. Впорядкованість. Автоматизація дозволяє групувати або централізувати знання та забезпечує його належне зберігання та запис. Після автоматизації процесу збору інформації, його легко оптимізувати та легко посилатись в межах компанії. Це важливо для централізації знань, оскільки команди підтримки та розробки працюють окремо, стаються ситуації, коли недоліки в комунікації завдають шкоди;

3. Надійна масштабованість. Зазвичай команди служби підтримки починаються з малих розмірів. Та допоки клієнтська база невелика та кількість запитів не критична, для вирішення вони можуть використовувати документацію або внутрішні комунікації. Як тільки клієнтів стає більше, та відповідно, потік їх запитів стає інтенсивнішим, трудомісткі ручні операції почнуть негативно впливати на ефективність підтримки та прибутковість бізнесу. В такому випадку автоматизація стає єдиним вирішенням даної проблеми;

4. Задоволеність агентів підтримки. Обов'язки представників агентів обслуговування клієнтів одноманітні та вимогливі. Вони повинні працювати з потоком запитів однакової інтенсивності весь робочий час. Надання агентам потужності баз автоматичних знань та персоналізованих вказівок щодо вирішення проблем підвищує задоволеність користувачів системи - агентів. Автоматична маршрутизація запитів до агентів з конкретною кваліфікацією, рис особистості та показників продуктивності показує, що цінується їх індивідуальність, а також, наявністю належними джерелами даних для забезпечення їх успіху. Інакшими словами, автоматизація підтримки клієнтів визнає індивідуальність та дає змогу працівникам бути максимально активними.

Як варіант ефективної автоматизації, сучасні компанії використовують машинне навчання для надання «мудрої та розумної» підтримки клієнтів. Наприклад, воно допомагають своїм клієнтам вирішувати більші обсяги

запитів на обслуговування клієнтів та забезпечує швидке надання зворотного зв'язку своїм клієнтам. Таким чином, агенти з обслуговування клієнтів можуть зосереджуватись на вирішенні більш важливих користувацьких або продуктових проблем, а не на рутинній роботі.

За допомогою методів машинного навчання можна визначати групи агентів обслуговування, які мають отримати запит клієнта про проблему, базуючись на навичках та компетентності тієї чи іншої групи. Після цього, система автоматично додаватиме запити клієнтів у чергу на обслуговування даним агентами. Також, машинне навчання може дозволити інтерфейсу підтримки рекомендувати агентам найкращий шаблон відповідей для зворотного зв'язку з клієнтами, який формується на основі вмісту та історії звернень клієнта. Більш того, якщо у систему впроваджені алгоритми машинного навчання, система може знаходити рішення для піднятого клієнтом проблеми чи запиту в базі знань, автоматично підбирати вирішення, виправляти проблему та виконувати зворотній зв'язок за допомогою повідомлень [2, 50с.].

В цій роботі будемо розглядати таку складову віртуальних інструментів підтримки як система відслідковування запитів про проблеми - ITS (англ. Issue Tracing System). Також використовуються альтернативні назви: Help Desk System, Incident Ticket System, Support Ticket System, Request Management or Trouble Ticket System. Відправка проблемних запитів до відповідної особи чи підрозділу підтримки має вирішальне значення для забезпечення обслуговування кінцевих користувачів, підтримуючи кращий розподіл ресурсів підтримки. Основними проблемами роботи з проблемними запитами клієнтів є:

1. Ручне призначення запиту на відповідні групи підтримки. Дана проблема найчастіше зустрічається у великих компаніях, коли потік клієнтських запитів інтенсивний. Також, це забирає багато часу і зусиль людини, що також може призвести до помилок. Ручне призначення збільшує

час відгуку, що призводить до погіршення задоволеності кінцевих користувачів;

2. Освоєння нових користувачів з розгорнутою структурою категоризації. Робота з системою, які дають користувачеві можливість вибору пов'язаних категорій або підрозділу в межах визначених категорій, може бути продуктивнішою в рази, проте вона не буде ефективною для нових користувачів, які ніколи використовували систему раніше та не мають поняття про зв'язки між категоріями та відповідними групи для призначення;

3. Також користувачі не хочуть заповнювати довгі форми запитів, необхідні для виявлення проблеми.

Все це свідчить про те, що віртуальні системи підтримки клієнтів відіграють вирішальну роль в операціях підтримки організації. До того ж, автоматизація таких систем з впровадженими алгоритмами передбачення, класифікації клієнтських звернень, а згодом з рекомендаційними алгоритмами пошуку найкращого і доцільного вирішення буде все більше цікавити топ менеджерів і власників компаній.

Таким чином, у цьому підрозділі ми розглянули приклади формалізації постановки задачі БКР на основі встановлених означень.

6.1.2. Матриця сутностей уніфікованої моделі проходження практики.

Поточний/календарний контроль та систематизація сутностей проходження практики за конкретною темою БКР та завданнями, доцільно проводити за допомогою матриці сутностей моделі проходження переддипломної практики – МС УМ ПП, що була детально прописана у розділі 2.2.

Зрозуміло, що для конкретних тем БКР і індивідуальних завдань практики контент матриці набуває ексклюзивного характеру і відображає

індивідуальну МС УМ ПП БКР за темою конкретного наукового дослідження.

Індивідуальна МС УМ ПП БКР формується як робочий індивідуальний початковий/поточний інформаційний ресурс для кожного студента, включає умови і вимоги як до процесу проведення практики, так і до БКР у цілому, а в процесі проходження практики доповнюється і розширюється конкретними результатами.

Приклад індивідуальної МС УМ ПП БКР показаний у табл. 1.

Приклад. [Для прикладу використані фрагменти МС УМ ПП МД студента Івана Савчука виконані у співавторстві в процесі вивчення дисциплін «Основи наукових досліджень» та «НДР за темою МД» і виконання МД на тему «Метод системної інженерії уніфікованої моделі DevOps системи для (продуктів) IT індустрії», викладач дисциплін та науковий керівник МД Маслянко Павло Павлович, к.т.н., доцент,»]

Таблиця 1. Матриця Сутностей уніфікованої моделі проходження практики (МС УМ ПП) для формування поточного стану індивідуального бізнес-профіля наукових досліджень за темою МД магістрант гр..КМ 11 2-й курс 21/22, v 0.9

Сутності та атрибути моделі системи «Практика»	Опис сутності уніфікованої моделі проходження практики	Примітки (E-mail:, Tel.: , бібліографія [ ], посилання https://..., інше)
1	2	3
№ за списком. ПІБ студента	*Савчук Іван Васильович*	№ за списком. ПІБ студента
Н/керівник ПІБ	*Маслянко Павло Павлович*	Н/керівник ПІБ
Назва н/досліджень *(назва н/досл. за темою МД)*	Метод системної інженерії уніфікованої моделі DevOps системи для (продуктів) IT індустрії.	Назва н/досліджень *(назва н/досл. за темою МД)*
ФПЗ: 1. Об'єкт досл., 2. Предмет досл. 3. Мета проведення конкретного н/досл. 4. Кінцевий результат наукового дослідження	**Об'єкт дослідження:** Поняття «система», стадії життєвого циклу системи, класи сутностей – складових IT систем, поняття DevOps IT систем, класифікація (систематизація) DevOps IT систем, сутності DevOps IT систем, існуючі системи DevOps IT систем, функціональність призначення і область застосування DevOps IT систем, повна вартість володіння DevOps систем, інше.	

Продовження Таблиці 1.

1	2	3
	Підходи до подання текстової інформації за встановленими наборами тематик, яка потрібна для створення систем питання-відповідь: SQuAD, SelQA, QuAC; моделі векторного представлення слів: Word2Vec, FastText, BPEmb; архітектура нейронних мереж трансформерів: Encoder-Decoder Transformer, Reformer, Longformer, ELECTRA. Існуючі програмні та апаратні засоби для обробки та використання текстів написаних природньою мовою: Tensorflow, Keras, PyTorch, Trax, Hugging Face (transformers). Методики застосування DevOps систем. Методи продукування DevOps та моделі DevOps систем, математичні інструменти та алгоритми реалізації DevOps систем (класи математичних інструментів та алгоритмів), обмеження та області застосування.	

Продовження Таблиці 1.

1	2	3
	Предмет дослідження: Метод системної інженерії уніфікованої моделі DevOps системи для IT індустрії, структурне та динамічне представлення системної інженерії уніфікованої моделі DevOps, моделі інтерфейсів компонентного представлення, класи математичного забезпечення компонентів моделі DevOps. Математичне та програмне забезпечення системи типу питання-відповідь на основі мереж трансформерів базуючись на контексті питань які задають користувачі за визначеним переліком тематик, а саме: культура, тварини, люди, історія, події пов'язані з людьми, географія, здоров'я. Моделі структурного та динамічного представлення системи типу питання відповідь. Дослідження та порівняльний аналіз математичних моделей трансформерів та представлення	

Продовження Таблиці 1.

1	2	3
	тексту, вибір та обгрунтування моделей трансформерів. Системна інженерія системи типу питання відповідь, математичне та програмне забезпечення системи на основі мереж трансформерів. Верифікація, валідація системи типу питання відповідь. **Мета дослідження:** Розробка методу системної інженерії уніфікованої моделі DevOps системи для IT індустрії призначеної для застосування на всіх стадіях життєвого циклу IT систем. Аналіз текстової інформації і встановлення предметного- змістовного наповнення текстів відповідної тематики для автоматизації знаходження максимально точних відповідей на запити користувачів за різними темами.	

Продовження Таблиці 1.

1	2	3
	Кінцевий результат дослідження: Метод системної інженерії уніфікованої моделі DevOps системи для продуктів IT індустрії, уніфікована модель DevOps системи, методика застосування, необхідне математичне та програмне забезпечення. Система типу питання-відповідь, математичне, програмне, методичне та інші види забезпечення задля цифровізації роботи з користувачами.	
Вказати Клас Ресурси/теоретичні, потрібні для реалізації конкретного н/досл. (Методи, моделі, алгоритми. Назви математичних дисциплін прослуханих	*Вказати Ресурси/теоретичні, потрібні для реалізації конкретного н/досл.:* - ***Методи:*** - Метод системної інженерії, теорія стстем і системного аналізу, методи Data Science, Методи продукування та моделі DevOps систем. - ***Моделі:*** Моделі DevOps систем, моделі інтерфейсів	

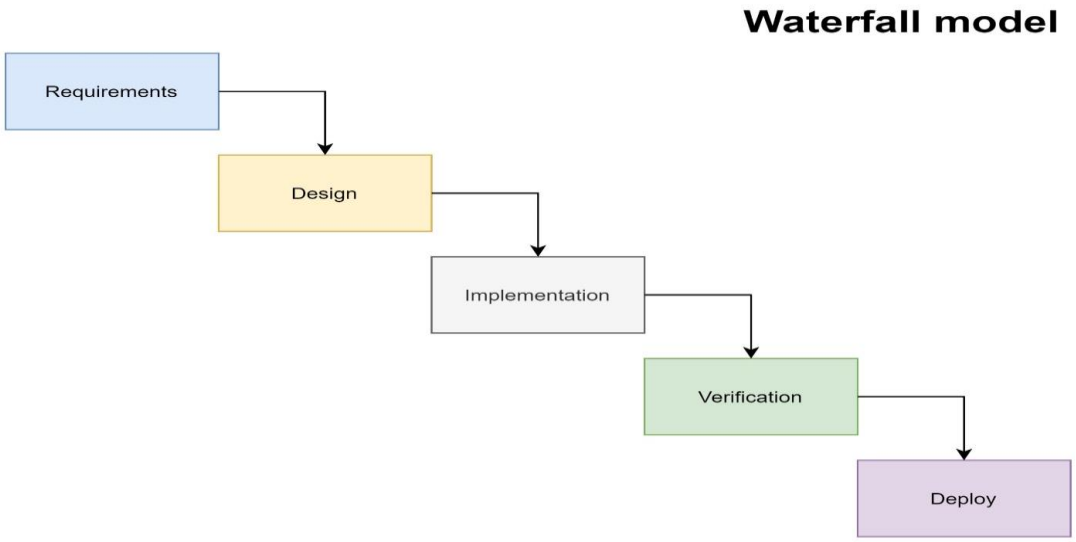
Продовження Таблиці 1.

1	2	3
Назви математичних дисциплін прослуханих в університеті та/ або за його межами і назви конкретних математичних інструментів цих дисциплін, тощо, потрібні для реалізації конкретного н/досл)	представлення, Моделі структурного та динамічного представлення системи типу питання відповідь, моделі векторного представлення слів (Word2Vec, FastText, BPEmb), моделі нейронних мереж трансформерів (ELECTRA, Longformer, Reformer) Назви математичних дисциплін прослуханих в університеті та/або за його межами і назви конкретних математичних інструментів цих дисциплін, тощо, необхідні для реалізації МД: - Методи оптимізації; - Бази даних та інформаційні системи; - Аналіз даних; - Методи штучного інтелекту; - Алгоритми і системи комп'ютерної математики; - Машинне навчання; - Системна інженерія	

Продовження Таблиці 1.

1	2	3
	- Алгоритми і системи комп'ютерної математики; - Машинне навчання; - Системна інженерія	
Клас Ресурси/засоби, потрібні для реалізації конкретного н/досл. *(Matchad, Matlab, RStudio,…)*	*Вказати Ресурси/засоби, потрібні для реалізації конкретного н/досл.:* - *Jupyter Notebook;* - *Python;* - *Tensorflow;* - *Keras;* - *Trax;* - *PyTorch;* - *Hugging Face (transformers);* - *Gradient;*	

Продовження Таблиці 1.

1	2	3
Клас Процеси реалізації конкретного н/досл. *(Водоспадний, спіральний, ітеративно-інкрементний, тощо)*	*Вказати Процеси реалізації конкретного н/досл. Із застосуванням до конкретного н/дослідження.* *(Водоспадний, спіральний, ітеративно- інкрементний, тощо)*  Для реалізації буде використовуватись модель Waterfall: - *Requirements:* На цьому етапі буде збиратися вся інформація про DevOps системи та системи питання відповідь, це різного роду	

Продовження Таблиці 1.

1	2	3
	методологічне та математичне забезпечення (архітектура нейронних мереж, принципи роботи з текстом тощо.) - Design: Буде розроблено модель системи, її представлення у вигляді діаграм, розписані всі основні процеси системи. - Implementation: Конкретна реалізація системи за встановленими вимогами та дизайном, це включає в себе кодування, математичну та апаратну обробку даних, налаштування математичних алгоритмів роботи з текстом. - Verification: Якість отриманих моделей має бути виміряна та підтверджена відповідними тестами, окрім того програмне забезпечення також має бути покрито тестами принаймні на 50%. -Deploy: Включає в себе розгортання застосунку на хмарному сервері, та розміщення проекту на GitHub.	

Продовження Таблиці 1.

1	2	3
Клас Бізнес-правила реалізації конкретного н/досл. (Технічні умови, технічні вимоги, календарний план виконання, тощо.)	Вказати Бізнес-правила реалізації конкретного н/досл. Технічні умови, технічні вимоги до кінцевого результату н/досл. (Призначення, область застосування, функціональність, вимоги до верифікації і валідації, повна вартість володіння, тощо), календарний план виконання, тощо. Бізнес правила: BR1.Довжина введеного користувачем тексту в систему не може перевищувати 500 слів. BR2.Система має надавати відповідь користувачу не більш як за 5 секунд. BR3.Точність системи не має бути нижчою за 60% за метриками EM (exact match) та F1. BR4. Система має надавати якісні відповіді користувачу тільки за визначеними в системі темами.	

Продовження Таблиці 1.

1	2	3
	BR5.Інтерфейс системи має налічувати поле для введення даних користувачем, та поле де буде виводитись відповідь системи. BR6.Інтерфейс системи має бути адаптовано для формату моніторів ПК та мобільних пристроїв. Функціональність і призначення компонентів системи питання-відповідь: 1. Сховище даних для розробки – сховище в якому знаходяться текстові дані для навчання нейронних мереж. 2. Модуль обробки тексту – призначений для обробки текстової інформації і приведення її до вигляду в якому нейронна мережа буде здатна з ними працювати. Обробка в себе включає: первинне очищення тексту від спеціальних символів, пропусків, приведення до нижнього регістру, нормалізація;	

Кінець Таблиці 1.

1	2	3
	розбиття речень на слова; представлення слів у формі векторів чисел. 3. Навчання мережі трансформера – модуль в яку проводиться навчання, налаштування, адаптація та тестування нейронної мережі. Його основним результатом є навчена модель мережі трансформера. 4. Генерація відповіді – компонент який на основі отриманих даних з модуля обробки тексту та навченої нейронної мережі генерує відповідь на поставлені питання користувачів. 5. Інтерфейс користувача – модуль який надає графічний інтерфейс користувача для роботи з системою питання відповідь.	
Примітки		

Таким чином Матриця сутностей уніфікованої моделі проходження практики призначена для формування поточного стану індивідуального бізнес-профіля наукових досліджень/розробок за темою БКР систематизує результати роботи за кожним проектом БКР і дозволяє контролювати стан його виконання у відповідності до умов та вимог як до процесу проведення практики, так і до самої БКР.

6.1.3. Шаблон бакалаврської атестаційної роботи

Ключовими результатами виконання **Завдань класу 1** і **Завдань класу 2** є:

- підготовка шаблону БКР – першої версії БКР за форматом і вимогами Положення про державну атестацію студентів КПІ ім. Ігоря Сікорського з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами (поточний контроль проведення практики);

- формування і захист першої версії шаблону БКР з виконаними всіма розділами (поточний контроль проведення практики);

Для формування індивідуального шаблону БКР доцільно скористатись методичними вказівками щодо формування структури і змісту БКР, розміщеними за посиланням на сайті каф. ПМА

https://pma.fpm.kpi.ua/uk/studentam/vipuskna-atestatsiya

https://pma.fpm.kpi.ua/uk/studentam/vipuskna-atestatsiya/vimogi-do-strukturi-ta-zmistu-diplomnoyi-roboti-bakalavra

За цими посиланнями детально описані всі особливості форматування структури та змісту БКР, зокрема БКР складається з таких елементів:

- Титульний аркуш встановленого зразка ;
- Завдання на БКР встановленого зразка;
- Реферат (українською мовою);
- Abstract (реферат англійською мовою);
- Зміст;

- Перелік умовних позначень, скорочень і термінів;

- Вступ;

- Постановка задачі бакалаврської атестаційної роботи;

- Основна частина з переліком розділів;

- Висновки;

- Список використаної літератури;

- Додатки.

А готові шаблони оформлення БКР розміщені за посиланнями:

https://pma.fpm.kpi.ua/_next/static/media/f008b68c-c5bd-4cc2-83cb-501b2b8dc3ba.dotx

З методичної точки зору, це дуже хороша підказка і дуже хороша метамодель БКР яку потрібно застосувати для конкретної теми БКР. А як же це відбувається на практиці?

А на практиці все дуже просто, не дивлячись на ці добре формалізовані шаблони, у студентів, тим не менше, виникає основне питання: - «А з чого потрібно починати?», «А що потрібно конкретно писати у тих, чи інших розділах БКР, за конкретними темами БКР, за конкретними постановками задачі БКР і за конкретними знаннями і ресурсами, що напрацьовані студентом за конкретною темою БКР ?».

В рамках цього навчального посібника ми спробуємо на прикладах дати відповідь на ці, та інші питання, на основі порівняльного методу наукового пізнання. Нагадаємо, що суть цього методу полягає у навчанні на добре описаних і прокоментованих прикладах і порівнянні своєї реалізації результатів з цими прикладами. Такий підхід суттєво полегшує розуміння студентом змісту того, чи іншого розділу і його відповідності темі і постановці задачі БКР.

Покажемо на прикладах змістовну частину розділів БКР. Назви розділів і підрозділів прикладів виділено курсивом.

Приклад 1. [Для прикладу використані фрагменти розділів БКР студента Олега САМОДРИГИ започатковані у співавторстві з науковим керівником в процесі вивчення

дисциплін «Системи Data Science» і виконання БКР на тему «Уніфікована модель корпоративної CRM системи для підприємств оптової та роздрібної торгівлі», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»]

6.1.3.1. Вступ

Почнемо з розділу Вступ. Вступ - вже перший розділ БКР, до формування змісту якого виникають питання.

На наш погляд у вступі дійсно потрібно висвітлити стан інженерної, науково-прикладної задачі та її актуальність, показати задачі, що залишились невирішеними і будуть об’єктом подальшої наукової та інженерної розробки.

І тут у Вступі важливо показати систему наукової і прикладної складових існуючих напрацювань за темою задекларованою у БКР з відповідним системним узагальненням стану, перспектив та актуальності подальших інженерних розробок в рамках БКР. Тобто, зміст розділу Вступ започатковує опис об'єкту дослідження/розробки, наводить перелік основних літературних джерел з останніми результатами досягнутими у цій області, демонструє актуальність і практичну значимість проведення досліджень/розробки у цій області.

Рекомендований розмір Вступу три – п’ять сторінок.

Наведемо приклади Вступу до БКР з системним узагальненням стану, перспектив та актуальності предмету дослідження/розробки за конкретною темою БКР.

Звертаємо увагу на добре прописаний зв'язок і застосування теоретичних та прикладних інструментів з предметною областю наукових досліджень. Вступна частина закінчується короткими узагальненими висновками щодо аналізу застосування розглянутих теоретичних та прикладних інструментів для вирішення встановлених завдань. Назви розділів і підрозділів прикладів виділено курсивом.

ВСТУП

CRM системи базуються на ідеї, що центром всієї теорії бізнесу є клієнт, а головними напрямками діяльності компанії є заходи щодо забезпечення ефективного маркетингу, продажів і обслуговування клієнтів.

Одні з перших концепцій CRM систем зародилися ще в 1970 року— задоволеність клієнтів оцінювалась за допомогою опитувань. Пізніше було запропоновано інструментарій управління контактами та концепт маркетингу баз даних, а саме застосування методів статистики для збору даних та аналізу клієнтів.

Такі тенденції були підтримані великою кількістю компаній та незалежних розробників, метою яких стала максимізація потенціалу лідів.
В сучасному світі дуже велика кількість різноманітних CRM систем. Ринок цієї галузі з кожним роком розростається все швидше і швидше, а в особливості CRM для оптової та роздрібної торгівлі. Майже кожне підприємство замислюється над питанням: як прискорити та покращити взаємодію з клієнтом? З цього виникає необхідність в створенні уніфікованої моделі корпоративної CRM системи для підприємств оптової та роздрібної торгівлі.

У даній роботі проаналізовано математичні моделі, що використовуються в CRM системах та вже існуючі CRM системи з метою розробки уніфікованої моделі корпоративної моделі CRM системи для підприємств для оптової та роздрібної торгівлі.

***Актуальність розробки уніфікованої моделі** CRM системи для сегменту підприємств оптової та роздрібної торгівлі визначається тим, що на основі такої уніфікованої моделі може бути розроблена модель конкретної CRM системи для реалізації потрібної замовнику бізнес-моделі діяльності підприємства оптової, або роздрібної торгівлі.*

Приклад 2. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»]

Вступ

Сучасна світова криза змушує людей частіше змінювати місце роботи у пошуках кращої пропозиції. Саме тому, під час пошуку нового місця роботи дуже важливо мати ефективний метод пошуку вакансій.

Існуючі рішення, зазвичай, рекомендують вакансії, що розміщено на їх власних платформах. Такий підхід є доцільним з боку самої платформи з вакансіями, але це суттєво зменшує кількість потенційних пропозиції для кандидатів та змушує їх слідкувати за великою кількістю різних платформ-альтернатив.

Також, одним з основних критерії при виборі нового місця роботи є заробітна плата, але більшість роботодавців не вказують даної інформації, в силу різних причин: не має чіткого плану оплати, оплата є пропорційною до фактичного набору навичок, погодинна оплата, етикет, тощо.

Тому, дана робота націлена на вирішення обох проблем, а саме створення автоматизованої системи у вигляді чат боту, що збирає інформацію про існуючі вакансії з різних популярних платформ, та рекомендує їх користувачу. А для вакансій без поля «заробітна плата» буде використано алгоритм для передбачення можливої заробітної плати на основі текстового опису поточної вакансії.

У наведених прикладах видно, що вступна частина закінчується короткими узагальненими висновками (виділено жирним курсивом) щодо аналізу стану застосування розглянутих теоретичних та прикладних інструментів для вирішення встановлених завдань у заданій конкретній області.

6.1.3.2 Формалізація постановки задачі бакалаврської атестаційної роботи

Приклад 1. [Для прикладу використані фрагменти розділів БКР студента Олега САМОДРИГИ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему **«**Уніфікована модель корпоративної CRM системи для підприємств оптової та роздрібної торгівлі», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»]

Об'єкт дослідження *— CRM системи, класифікація CRM систем, моделі, методи реалізації CRM систем, алгоритми функціонування.*

Предмет дослідження *— уніфікована модель корпоративної CRM системи для підприємств оптової та роздрібної торгівлі з використанням моделей Data Mining, інструменти та бізнес-процеси, що використовуються в CRM системах.*

Метою написання даної роботи *є дослідження існуючих CRM рішень, проектування уніфікованої корпоративної CRM системи для підприємств оптової та роздрібної торгівлі, формалізація бізнес процесів управління відносинами з клієнтами.*

Кінцевий результат*: уніфікована модель корпоративної CRM системи для підприємств оптової та роздрібної торгівлі, програмне забезпечення реалізації спроектованої моделі.*

Успішність CRM системи базується на наступних ознаках:

а) робота з угодою— зручність роботи з угодою;

б) виконання задач — маркетингові цілі, чи заходи що проводяться компанією;

в) розмежування прав доступу— обмеження прав доступу до розділів чи записів розділу, прав на редагування, читання;

г) звітність по проведеним заходам— інформація про продажі, договори,

клієнтів, компанії, поставлених задач, історії взаємодії з клієнтом.

ґ) зручність використання API —посилання, що викликає сервісну логіку, критерій характеризує легкість підтримки, доопрацювання CRM системи;

д) можливість доопрацювання даної в залежності від потреб компанії— характеризує доопрацювання системи під нужди компанії, створення нової бізнес-логіки чи модифікацію існуючої..

е) розділення на люди і контакти— критерій необхідний для коректного складання воронки продажу та розмежування потенційних клієнтів та існуючих.;

є) Складність освоєння системи новим користувачем— характеризує складність освоєння та застосування системи користувачем.

Для розв'язання поставленої задачі необхідно:

а) дослідити та проаналізувати існуючі рішення;

б) спроектувати модель уніфікованої CRM системи для підприємств оптової та роздрібної торгівлі;

в) формалізувати бізнес процеси взаємодії з клієнтами;

г) розробити математичне та програмне забезпечення на основі спроектованої моделі та бізнес процесів.

ґ) провести тестування розробленої CRM системи.

е) провести верифікацію та валідацію результатів роботи.

Приклад 2. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»]

Актуальність теми. *Реалії сучасного світу змушують людей все частіше змінювати місце роботи, через бажання обрати найбільш оптимальну професію, що відповідає набору навиків та очікуванням з*

заробітній платі людини. Ринок праці надає широкий список можливих варіантів, але в більшості випадків, в описі вакансії є відсутнім значення майбутньої заробітної плати. Дана робота націлена на вирішення даної проблеми шляхом передбачення заробітної плати для вакансії на основі її опису.

Об'єктом дослідження *є моделі машинного навчання для прогнозування на основі текстових даних, такі як: наївний Баєсів класифікатор, одновимірні згорткові нейронні мережі, ансамблеві моделі типу random forest.*

Предметом дослідження *є математичне та програмне забезпечення системи у вигляді чат боту для збору інформації про вакансії з різних платформ та прогнозування заробітної плати на основі текстового опису відповідної вакансій з використанням одновимірних згорткових нейронних мереж.*

Мета і задачі дослідження. *Метою даної роботи є спрощення процесу пошуку пропозиції для роботи, що розміщено на різних платформах та підвищення ефективності вакансій, у яких відсутнє поле «заробітна плата» на основі чат боту.*

Методи дослідження*. Для розв'язання даної задачі було використано наступні методи: теорія системного аналізу, системна інженерія, проектування систем Data Science, проектування Інформаційних Систем, обробка та аналізу текстових даних на основі методів машинного навчання, теорія алгоритмів, аналіз даних та математична статистика, моделювання на основі бізнес профілю Еріксона-Пенкера.*

Кінцевим результатом роботи *є система у вигляді чат боту, що проводить збір інформації про вакансії з різних платформ для пошуку роботи, рекомендує їх користувачу на основі заданих характеристик та прогнозує заробітну плату на основі опису для вакансій де відповідне поле відсутнє, що дозволить пришвидшити та підвищити ефективність процесу пошуку роботи в цілому.*

Для досягнення вказаної мети та отримання кінцевого результату роботи потрібно вирішити такі завдання:

– *Проведення аналізу існуючих рішень для пошуку роботи;*

– *Обґрунтування та вибір функціоналу чат боту для пошуку роботи;*

– *Проведення аналізу основних методів обробки текстових даних;*

– *Обґрунтування та вибір методів обробки текстових даних;*

– *Здійснення порівняльного аналізу потенційних архітектур моделей прогнозування заробітної плати.*

– *Розробка алгоритму для прогнозування заробітної плати з використанням моделі одновимірних згорткових мереж;*

Проектування системи для пошуку роботи на основі чат боту.

Ключові слова: *чат бот, вакансія, пошук роботи, машинне навчання, згорткові нейроні мережі, одновимірні згорткові нейронні мережі, Data Science, система Data Science, обробка текстових даних, передбачення заробітної плати, оптимізація процесу пошуку роботи, інформаційні системи.*

6.1.3.3. Основна частина з переліком розділів

Основна частина з переліком розділів викликає чи не найбільше питань щодо логічного порядку назв розділів та змісту окремих розділів і багато в чому залежить від кількості та якості напрацьованих студентом матеріалів за темою БКР.

Автори навчального посібника рекомендують авторський алгоритм роботи над цим розділом з умовною назвою *Основна частина*.

1. Першим кроком ми можемо рекомендувати розділ з обґрунтуванням актуальності і, як логічний результат, формалізоване представлення постановки задачі БКР, щоб потім виконати огляд існуючих рішень.

2. Другим кроком - Огляд існуючих рішень за темою БКР що прямо, або опосередковано вже сьогодні вирішують завдання задекларованої теми БКР. Як правило, існуючі рішення складають два класи, клас теоретичних інструментів та клас практичних засобів. Це, власне вже було формалізовано у Об'єкті дослідження, Формалізації постановки задачі. Детальний розгляд і аналіз цих класів дає можливість встановити і систематизувати область застосування і призначення, функціональність і повну вартість володіння, встановити переваги і недоліки інструментів і засобів.

Виконання цього розділу дає розробнику і перший науковий результат, що формалізується за вербальною формулою: «Досліджено і встановлено».

3. Третій, наступний логічний крок - перехід до вибору теоретичних методів і практичних засобів для вирішення завдань БКР, особливостей застосування та необхідності внесення змін і вдосконалення.

4. Четвертий крок. Важливою частиною є розробка структурного представлення, архітектури, компонентної моделі сутності розробки, яка є першим системним формалізованим результатом предмету БКР. Структурне представлення, архітектура, компонентна модель сутності розробки з чітко прописаними елементами і інтерфейсами між ними надають розробникам можливість упорядкувати порядок розробки окремих елементів, проводити верифікацію і валідацію результатів їх взаємодії, вчасно вносити корективи і проводити документування проміжних результатів.

5. Результати попереднього кроку дають обґрунтовані підстави для розробки /застосування математичного забезпечення для реалізації окремих компонентів і інтерфейсів між ними. Математичне забезпечення часто плутають з математичними моделями, або з набором математичних формул. Насправді це не так.

Математичне забезпечення, це окремий вид забезпечення інформаційно-комунікаційних систем і в залежності від області застосування надаються і відповідні означення. Тим не менше, практично всі означення зводяться до того, що математичне забезпечення, - це система математичних методів,

моделей і алгоритмів призначених для обробки передачі і збереження інформації в інформаційно-комунікаційних системах. Тут ключовим терміном є слово «система», що чітко вказує на обов'язкову інтеграцію математичних методів, моделей і алгоритмів у, власне, систему. Тому математичне забезпечення має бути і відповідним чином описане і задокументоване.

Справа в тому що більшість математичних інструментів є запозиченими і тут важливо продемонструвати особливості їх застосування, взаємодії окремих математичних моделей і результатів їх роботи.

6. Добре задокументоване математичне забезпечення слугує основою для розробки програмного забезпечення, верифікації і валідації результатів розробки і нарешті формування висновків за результатами виконання БКР.

Програмне забезпечення - це система окремих програм і програмних модулів загальносистемного чи прикладного призначення та формалізованої документації, що призначена для безпосереднього виконання заданих функцій і надання задекларованих сервісів для користувачів.

7. І, нарешті, формування висновків за результатами виконаної роботи. Підкреслимо, саме висновків, а не анотації. Різниця висновків від анотації полягає у тому, що висновки формалізують значимість отриманих результатів у кількісних і якісних метриках, а анотація є коротким змістом результатів виконаної роботи.

Тут ми рекомендуємо декілька важливих пунктів авторської версії висновків.

Перший пункт висновків продукується на основі результатів огляду існуючих рішень за формулою «Досліджено і встановлено».

Другий пункт висновків відображає елементи наукової новизни (за наявності) результатів роботи у порівнянні з існуючими рішеннями.

Третій пункт висновків демонструє практичну цінність кінцевого результату роботи: наукову, науково-технічну, економічну, соціальну або іншу, яку можна отримати в результаті застосування результатів роботи.

Четвертий пункт висновків показує перспективи подальших досліджень/розробок у задекларованому напрямі наукових досліджень.

Зазначимо надзвичайно важливу особливість запропонованого алгоритму, що полягає у прозорій і чіткій інтеграції його окремих частин.

Далі покажемо приклади застосування цього алгоритму для реалізації окремих частин БКР у процесі проходження практики. У цих прикладах, для зручності і визначення однозначної належності змісту до того чи іншого розділу, ми запроваджуємо умовну нумерацію розділів прикладів БКР з відповідними назвами цих розділів.

6.1.3.4. Обґрунтуванням актуальності тематики досліджень/розробки

Звертаємо увагу на наглядний і добре проілюстрований зміст цього прикладу обґрунтування актуальності, що демонструє, з одного боку, добру обізнаність розробників у предметній області, а з іншого, наочність і зрозумілість для потенційних користувачів розробки.

У цьому прикладі ми звертаємо увагу читача на чіткий зв’язок сутностей предметної області розглянутих попередньому розділі.

Приклад 1. [Для прикладу використані фрагменти розділів БКР студента Олега САМОДРИГИ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему **«**Уніфікована модель корпоративної CRM системи для підприємств оптової та роздрібної торгівлі», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»]

Актуальність розробки уніфікованої моделі *CRM системи для сегменту підприємств оптової та роздрібної торгівлі визначається тим, що на основі такої уніфікованої моделі може бути розроблена модель конкретної CRM системи для реалізації потрібної замовнику бізнес-моделі діяльності підприємства оптової, або роздрібної торгівлі.*

Успішність CRM системи базується на наступних ознаках:

а) робота з угодою— зручність роботи з угодою;

б) виконання задач — маркетингові цілі, чи заходи що проводяться компанією;

в) розмежування прав доступу— обмеження прав доступу до розділів чи записів розділу, прав на редагування, читання;

г) звітність по проведеним заходам— інформація про продажі, договори, клієнтів, компанії, поставлених задач, історії взаємодії з клієнтом.

ґ) зручність використання API —посилання, що викликає сервісну логіку, критерій характеризує легкість підтримки, доопрацювання CRM системи;

д) можливість доопрацювання в залежності від потреб компанії— характеризує доопрацювання системи під нужди компанії, створення нової бізнес-логіки чи модифікацію існуючої..

е) розділення на ліди і контакти— критерій необхідний для коректного складання воронки продажу та розмежування потенційних клієнтів та існуючих.;

є) складність освоєння системи новим користувачем— характеризує складність освоєння та застосування системи користувачем.

Приклад 2. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»].

Проблематика пошуку вакансій

На сьогоднішній день, більшість працевлаштувань відбувається на основі вакансій, розміщених в мережі інтернет. Для цього існує величезна кількість платформ, що надають можливість шукати вакансії різного типу та напрямку.

Але, однією з основних проблем процесу пошуку вакансій, є саме велика кількість незалежних платформ. Кожна платформа розміщує, в основному, різні вакансії, що сприяє її конкурентно-спроможності, але

впливає на ефективність пошуку роботи. Оскільки, людина змушена паралельно здійснювати процес пошуку роботи на усіх доступних платформах для підвищення своїх шансів на успіх.

Також, кожна платформа, в загальному, надає свій специфічний функціонал, спосіб рекомендації вакансій та систему оповіщення.

Різнорідність даного типу також негативно впливає на ефективність пошуку.

Іншою проблемою є те, що зазвичай роботодавці включають достатню кількість інформації з описом конкретної вакансії, для зацікавлення майбутнього працівника. Але, також часто не включають заробітну плату до опису вакансії. Це може бути зумовлено різними причинами, наприклад дана спеціальність не має чіткого плану оплати роботи, і все вирішується під кожен випадок індивідуально, чи оплата є пропорційною до фактичного набору навичок, що наявні у кандидата, чи як звичайне правило етикету.

Незалежно від причини, переважна більшість вакансій фактично залишається без поля «заробітна плата». З боку кандидата, одним з найголовніших критеріїв, після напрямку спеціальності, є саме заробітна плата. Оскільки, кандидат завжди зацікавлений в найкращому способі конвертації своїх навиків у заробітну плату. Тому, в результаті цього, текстовий опис вакансії втрачає значну частину інформативності для потенційного кандидата.

У цих прикладах звертаємо увагу і на формування висновків до цього розділу, де у термінах і категоріях предметної області чітко систематизована актуальність предмету дослідження і практична цінність кінцевого результату розробки.

Також нагадуємо, що нумерація розділу і підрозділів прикладу відноситься лише до цього прикладу і до нумерації розділів змісту навчального посібника ніякого відношення не має. Назви розділів і підрозділів прикладів виділено жирним курсивом.

6.1.3.5. Огляд існуючих рішень за темою БКР

Як ми уже зазначали у алгоритмі формування розділу БКР з умовною назвою Основна частина, Огляд існуючих рішень за темою БКР що прямо, або опосередковано вже сьогодні вирішують завдання задекларованої теми БКР, складаються з двох класів сутностей: класу теоретичних інструментів та класу практичних засобів. У цьому розділі має бути проведений детальна систематизація і аналіз цих класів за встановленими ознаками, властивостями, функціональністю, повною вартістю володіння, тощо, і в результаті встановити рівень придатності сутностей цих класів для досягнення мети дисертаційного дослідження.

Як правило, Висновки до цього розділу формуються на основі вербальної моделі «Досліджено і встановлено» у вигляді таблиць порівняльного аналізу сутностей класів існуючих рішень.

Звертаємо увагу на чітку кореляцію сутностей класів існуючих рішень з завданнями прогнозування предметної області БКР, обґрунтування вибору сутностей існуючих рішень для подальшого застосування, і висновків до цього розділу.

Також нагадуємо, що нумерація розділу і підрозділів прикладу відноситься лише до цього прикладу і до нумерації розділів змісту навчального посібника ніякого відношення не має.

1. АНАЛІЗ ІСНУЮЧИХ CRM СИСТЕМ

Приклад 1. [Для прикладу використані фрагменти розділів БКР студента Олега САМОДРИГИ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Уніфікована модель корпоративної CRM системи для підприємств оптової та роздрібної торгівлі», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»]

1.1 Математичні моделі CRM аналітики

1.1.1Моделі регресії

Моделі регресії застосується в оцінці ймовірності події або її числове значення. Основне застосування в задачах прогнозування попиту, оцінки еластичності цін, оцінки ймовірності повторної продажі, розрахунку використання складу, магазину, замовлення, аналіз впливу факторів на попит[5, 6, 7]. Також застосується за для вирішення задачі класифікації, проте в порівнянні з іншими моделями визнано не точним.

До регресійних моделей можна віднести: факторну, лінійну, поліноміальну, поверхневого оклику, поверхневої суміші регресії

1.1.2 Моделі кластеризації

Метою проведення кластерного аналізу клієнтської бази є її поділ на групи клієнтів зі схожими ознаками, засноване на оцінці значень безлічі характеристик клієнтів (їх поведінкової історії) [7, 8].

При проведенні кластеризації заздалегідь не відомо кількість формованих кластерів-груп клієнтів, також як і правила приналежності клієнта до певного кластеру. Кластеризація дозволяє виявляти в наборі необроблених даних поведінкової історії клієнтів приховані закономірності з метою проведення адресному персоніфікованої маркетингової політики у відношенні клієнтів.

Модель виконує задачі кластеризація товарів, виявлення товарів з подібною структурою попиту, розподіл споживачів на групи, подібні за структурою та поведінкою, аналіз попиту залежно від поєднання вхідних показників, виявлення аномальних відхилень [6].

До моделей кластерного аналізу можна віднести: алгоритм k-середніх, c-середніх, ієрархічний алгоритм, мінімально покриваюче дерево, алгоритм пошарової кластеризації, нейрону мережу Кохонена.

1.1.3 Моделі пошуку асоціативних правил

Аналіз подій, що відбуваються разом. Виявлення залежності, що від однієї події з певною ймовірністю слідує інша подія. Використовується для передбачення поведінки клієнта та пропозиції товару, який, швидше за все, його зацікавить, розміщення товарів на полицях, в каталогах, стимулювання збуту одних товарів продаючи інші, оптимізація складських запасів [6, 7].

До основних моделей пошуку асоціативних правил можна віднести алгоритм Apriori, Eclat, FP-росту

1.1.4 Моделі класифікації

Класифікація - знаходження функціональної залежності між вхідними параметрами і дискретним вихідним параметром. Класифікація дозволяє віднести об'єкт до одного з відомих класів, що дозволяє оцінити перспективність клієнтів; аналізувати ризики: чи варто давати кредит чи ні; оцінка знижок: якої категорії клієнтів надавати знижки; прогнозування успіху угоди, оцінка ефективності рекламної компанії [6, 7].

До моделей класифікації можна віднести: дерева рішень, багатошарові персепртони, ймовірнісні нейроні мережі

1.1.5 Моделі послідовностей.

Прогнозують зміну неперервних числових параметрів [6, 7]. Вони будуються на підставі даних про зміну деякого параметра за минулий період часу.

До моделей послідовностей модна віднести: моделі екстраполяції, моделі страхування.

1.1.6 Порівняння математичних моделей

Проведемо порівняльний аналіз моделей, що застосуються в аналітиці CRM систем.

Порівняльний аналіз наведено в таблиці 1.1.

Таблиця 1.1.– Порівняння моделей для розв'язку задач

Модель /Характеристика	*Моделі класифікація*	*Моделі асоціативних правил*	*Моделі кластеризація*	*Моделі послідовностей*	*Моделі регресії*
Контрольованість навчання	*Контрольоване навчання*	*Неконтрольоване навчання*	*Неконтрольоване навчання*	*Контрольоване навчання*	*Контрольоване навчання*
Стратегія	*Навчання з учителем*	*Навчання без учителя*	*Навчання без учителя*	*Навчання з учителем*	*Навчання з учителем*
Модель /Характеристика	*Моделі класифікація*	*Моделі асоціативних правил*	*Моделі кластеризація*	*Моделі послідовностей*	*Моделі регресії*
Наявність мітки класу	*Мітка присутня.*	*Мітки невідомі*	*Мітки невідомі*	*Мітка присутня.*	*Мітка присутня.*
Основа для застосування	*Нові данні класифікуються згідно навчальній вибірки*	*Встановлення спільних особливостей множин об'єктів*	*Встановлення класів та кластерів на множинні даних*	*Прогнозування зміни числових характеристик*	*Нові данні класифікуються згідно навчальній вибірки*
Тип задачі	*Описова*	*Описові*	*Описова*	*Прогнозована*	*Прогнозована/Описова*

Всі моделі в CRM аналітиці можна розділити за призначенням розв'язку задач:

а) описові — належність об'єкта до групи;

б) прогнозовані — імовірнісне судження про майбутній стан об'єкта;

1.2 Порівняльний аналіз CRM систем

За своєю функціональною направленістю CRM системиможна поділити на наступні типи [9]:

– *Операційні*

– *Аналітичні*

– *Колаборaційні*

Операційні– направленні на облегшення виконання повсякденних задач компанії. Такі системи спрощують взаємодію з клієнтами, систематизують дані про заявки і угодах, самі виставляють рахунки, нагадують передзвонити клієнтові і можуть самі відправити йому sms-повідомлення, записують телефонні дзвінки, тобто автоматизують роботу підприємства[9].

Аналітичні – містять детальну інформацію про клієнтів і взаємодію з ними. Завдяки чому дозволяє визначити закономірності в продажах та сформулювати з яких джерел клієнти купують найчастіше та на якому етапі зриваються угоди [9].

Колаборaційні – направленні на налагодження комунікації з клієнтом для збору зворотного зв'язку. Кожна система такого типу розробляється індивідуально, або інформація з існуючих каналів зв'язку фіксуються в CRM системі [9].

Порівняння функціональної направленості CRM систем наведено у таблиці 1.2. [10]

Таблиця 1.2 – Функціональна направленість типів CRMсистем[10]

Направленість	*Завдання*	*Функція*	*Реалізація*
Операційна	*Збереження інформації для подальшого застосування про клієнтів, продажі, маркетингові акції, взаємодію з клієнтом, обслуговування клієнтів, потенційних клієнтів, етапів формування продажу*	*Забезпечення взаємодії, через усі канали зв'язку*	*Автоматизація роботи усіх каналів зв'язку, відділів продаж, маркетингу, служб підтримки, центра телефонного обслуговування*

Продовження таблиці 1.2

Направленість	*Завдання*	*Функція*	*Реалізація*
Аналітична	*Аналіз результатів продажу, клієнтів та його компанії та контактів з ним за для подальшого прогнозування взаємодії та складання рекомендації керівництву компанії*	*Обробка інформації, що поступає про всю взаємодію з клієнтом. Аналіз потреб та складання попиту, рекомендацій та пропозицій кожному конкретному повторному клієнтові.*	*Система визначення цінностей клієнта, побудови та аналізу моделі поведінки клієнтів та аналізу усієї роботи з ними.*
Колаборaційна	*Вплив клієнта на безпосередню роботу виготовлення продукту, роботи контакт-центру, обслуговування*	*Забезпечення зв'язку з клієнтом зручним йому способом, інтеграція з іншими системами*	*Контакт-центр, імейл розсилки, веб-сайти, соціальні мережі, веб портали*

Сучасні CRM – рішення в своїй більшості здатні збирати, систематизувати всю необхідну інформацію, робити на її основі аналізпрогнози, а також сприяти спрощенню контактів зі споживачами.
У таблиці 1.3 представленні CRM – продуктів залежно від цільового призначення[11].

Таблиця1.3 – Приклади цільового використання CRM-продуктів[11]

Цільове використання	*CRM системи*
Оперативне	*Oracle Siebel CRM, Onyx, Creatio, Maximaizer, GoldMine, Sales Logix, SAP, Sales Expert, Парус, Microsoft Dynamics CRM, Mango CRM, amoCRM, Megaplan CRM, Bitrix24*
Аналітичне	*1C: CRM, Hyperion, SAS Marketing Automation*
Колаборaційне	*Broadvision,Microstrategy, Salesforce*

1.2.1 Oracle Siebel CRM

Готові стандартні рішення. Програмне забезпечення пропонує близько 20 готових рішень з урахуванням конкретних галузей. Це зменшує час, необхідний для впровадження Oracle Siebel CRM, та витрати на адаптацію програми до конкретних завдань - основні модулі вже попередньо встановлені та налаштовані.

Модульна архітектура – особливість CRM-системи, що дозволяє вибрати набір продуктів, необхідних на кожному етапі роботи, та розширити або звузити цей набір. Немає переплат за невикористані функції, програма оптимізована для потреб споживача.

Широкі можливості інтеграції. Siebel CRM дозволяє поєднувати не тільки користувальницькі інтерфейси, програми, але і бази даних, що суттєво відрізняє її від конкурентів.

Можна використовувати CRM-систему на різних платформах, наприклад, від комп'ютера до телефону або планшета, що робить бізнес мобільним.

CRM-система Siebel пропонує рішення для будь-якого каналу зв'язку: інтернету, контактного центру, працівника в торговій точці або цілого відділу. Всі ці канали об'єднані в загальну систему і відповідають загальним правилам роботи з клієнтами.

Програмне забезпечення характеризується підвищеним рівнем шифрування. Основним недоліком в зрівнянні з конкурентами є схожість з базою даних, це не як не впливає на функціонал, проте щоб навчитися користуватися CRM системою в більшості випадків необхідно буде пройти навчання.

1.2.2 Onyx

Присутня спеціальна інформаційна панель, яка допомагає користувачам сортувати кілька наборів даних та відстежувати різні ділові дії. Також присутня навігація, яка допомагає користувачам ідентифікувати та керувати інформацією про клієнтів.

Підтримує мобільну версію додатка, що дозволяє віддалено отримати доступ до CRM через мобільні пристрої.

Крім того, система пропонує автоматизацію та побудову сценаріїв процесів, управління продажами та маркетингом, управління сервісом та підтримкою, механізм звітування для генерації періодичних спеціальних звітів та підтримує інтеграцію з Microsoft Outlook та різними веб-сайтами соціальних мереж.

1.2.3 Maximaizer

Однією з найважливіших переваг є інтуїтивно зрозумілий інтерфейс за для роботи з системою немає необхідності проходити навчання використання системи.

Гнучке налаштування та легка модифікація під нужди користувача.

Легке налаштування прав доступу, що дозволяє засобами CRM приховувати чи показувати інформацію користувачам.

Проте ця CRM система має вагомі недоліки–занадто довга обробка великих об'ємів даних, повільне видалення даних та оновлення інформації.

Основна проблема: узагальнені назви стовбців в базі даних, через розроблену гучність системи–через що пошук інформації при роботи з базою даних дуже ускладняється.

1.2.4 Gold Mine

Однією з найважливіших особливостей даної CRM є зберігання усієї інформації в картці клієнта, в порівнянні з іншими CRM, інформація зберігається у різних розділах. Картка клієнта містить всю історію попередніх контактів та плани подальших взаємодій, документи, що стосуються клієнта, інформацію про поточні операції та проекти. При необхідності можна описати організаційну структуру компанії-клієнта, схему підпорядкування працівників та їх роль у процесах взаємодії. Система забезпечує зручні інструменти пошуку за будь-якими інформаційними полями та універсальні можливості для класифікації та групування ваших клієнтів за будь-якими критеріями будь-якої складності..

Гнучкість та розширюваність – архітектура та засоби налаштування дозволяють налаштувати відповідно до вимог клієнта. Дозволяє змінювати склад інформації, бізнес-правила та права доступу до інформації.

Шаблони готових рішень для різних видів діяльності безкоштовно розповсюджуються, що дозволяє скористатися досвітом інших компанії, що може значно збільшити швидкість та ціну впровадження.

1.2.5 Sales Logix

Рішення, що дозволяє повністю використовувати CRM систему: об'єднувати працівників, партнерів, процеси та технології в рамках повного замкненого циклу взаємодії з клієнтами. Дозволяє ефективно автоматизувати бізнес-процеси компанії вести бізнес.

Підтримує координування діяльності усіх структурних підрозділів, що взаємодіють з клієнтом, а також керувати роботою з використанням різних каналів зв'язку: особиста зустріч, телефон, Інтернет.

SAP надає доступ кожному організаційному підрозділу компанії заповнити актуальну інформацію про клієнта, необхідну для збільшення якість відносин з клієнтом та підвищення його лояльності.

Основні функції SAP CRM:

а) єдине інформаційне середовище – вся інформація про реальних потенційних клієнти, партнерів, конкурентів та історій змін. Всі дані зберігаються централізовано в єдиній системі, що зберігає інформацію про товари, послуги та ціни компанії;

б) визначення найбільш вигідних клієнтів – на основі даних картки клієнтів за звітом або пошуком за параметром, можна відсортувати клієнтів і визначити найвигідніших;

в) управління контактами – планування, моніторинг та аналіз усіх контактів з клієнтами. Зберігання історії взаємодії; продажів, етапів продажу, угод, створення воронки продаж;

г) аналітика – будь-яка необхідна аналітика. Порівняння успіху робота менеджерів, підрозділів, звіти та аналіз. SAPCRM дозволяє зберігати та аналізувати всю ділову інформацію

1.2.6 Sales Expert

Дуже гнучка CRM система для налаштуванням під нужди користувача, не потрібно наймати програміста, усе можна виконати внутрішніми засобами CRM, з цього випливає деяка обмеженість функцій в порівнянні з конкурентами. Має модульну архітектуру і може бути допрацьована засобами розробників чи сторонніми інтеграторами під потреби замовника. Оптимальний варіант для малих підприємств де може бути відсутня команда чи наймана компанія інтеграторів. В порівнянні з конкурентами має більш низьку ціну, проте базовий функціонал занадто малий для застосування на великих підприємствах.

1.2.7 Парус

Просте багатофункціональне рішення, є незалежним модулем більш великої комплексної системи автоматизації для малого та середнього бізнесу. У поєднанні з додатковими модулями дозволяє вирішувати управлінські завдання: бухгалтерського податкового обліку, продажу товарів і послуг, облік складських запасів, нарахування зарплатні, кадровий облік та набір персоналу.

1.2.8 Creatio

Об'єднує можливості системи управління взаємовідносинами з клієнтами (CRM) і системи управління бізнес-процесами (BPM).

Основні функції:

а) управління інформацією про клієнта: підтримка контактів та компаній, повна історія стосунків, легкий доступ до інформації про клієнта, можливість створювати власні поля та закладки, розподіл прав доступу;

б) бізнес-процеси: автоматизація рутинних операцій, можливість створення умов для розгалуження та дій щодо бізнес-процесу, організація колективної роботи, автоматичний контроль виконання функціональних ролей у проекті;

в) управління продажами: управління потенційними угодами, управління проектами, контроль умов оплати, доставки та реалізації, воронка збуту;

г) управління маркетингом: функціональність для планування та проведення маркетингової кампанії будь-якої складності, маркетингові дослідження, анкети, розсилки, звіти;

ґ) побудова ефективного контактного центру (інтеграція з модулем контакт центру);

д) автоматизація документообігу: ведення договорів та платежів, створення будь-яких шаблонів документів, можливість інтеграції з фінансовими системами;

е) дозволяє створювати статистичні діаграми, аналітичні звіти;

є) управління робочим часом: органайзер, календар груп;

ж) електронна пошта: інтеграція з MS Outlook, основна автоматизація персоналізовані розсилки електронною поштою за допомогою шаблонів.

1.2.9 Microsoft Dynamics CRM

Інструмент управління відносини з клієнтами. Це підвищує продуктивність праці працівників всередині та поза організацією та полегшує взаємодію відділів продажів, маркетинг та обслуговування споживачів із використанням сучасних технологій, інтегровані в єдине робоче середовище.

Основні результати використання Microsoft Dynamics CRM:

а) зменшення витрат на залучення нових клієнтів, висока якість маркетингових заходів та можливість аналізу проведених заходів;

б) зменшення циклу продажів і собівартості, управління воронками продажів, збільшення кількості закритих угод;

в) збільшення продажів існуючим клієнтам, зменшення собівартості обслуговування клієнтів, підвищити їх задоволеність і лояльність.

Переваги Microsoft Dynamics CRM для керівників:

а) комплексне CRM-рішення, що включає управління маркетингу, продажів та обслуговування споживачів, охоплює ціле коло завдань взаємодії з клієнтами;

б) аналітичні можливості ключових показників діяльності, що забезпечує контроль керівництва;

в) гарантії виробника на розробку та підтримку системи захисту інвестиції в технології.

1.2.10 Mango CRM

Підтримує телефонію, для обліку клієнтів, їх звернень та оптимізації робочих процесів.

Функціональні можливості для управління продажами:

а) можливість закріплювати менеджерів за клієнтами. чітко розподіляти клієнтів між співробітниками, перерозподіляти в разі хвороби або звільнення;

б) за допомогою модуля статистики CRM керівник завжди буде в курсі всіх угод і звернень клієнтів. Керівник буде знати, хто з менеджерів зробив найкращу продаж в місяць, а хто упускає клієнтів;

в) є шаблони рахунків і комерційних пропозицій.

1.2.11 amo CRM

Система обліку клієнтів. Головна відмінність полягає в тому, що системо утворюючою є не клієнтський файл, а файл угоди, що дозволяє керівнику слідкувати за роботою менеджерів: дивіться кількість здійснених дзвінків, запланованих та проведені зустрічі, результати переговорів. Всі зміни події, що відбулися протягом дня, підсумовуються у звіті "Події".
У базі даних з інформацією про клієнта є функція нагадування про наступний контакт.
Важливою перевагою є функція прямого маркетингу. Система дозволяє здійснювати розсилку пошти та SMS. В системі присутня побудова воронки продажу.

1.2.12 Megaplan CRM

Дозволяє зберігати всю інформацію про клієнтів у структурованій базі даних та відстежувати події та завдання, пов'язані з ними.

Megaplan також дозволяє скласти схему продажів будь-якої складності або скористайтеся однією з готових схем. Використовуючи функції управління транзакцій можливо відстежувати роботу конкретних менеджерів та відділу продажів загалом, фіксує доходи, витрати та грошові операції з кожним клієнтом.

На жаль, систему не можна назвати простою. Однак це проблема більшості CRM.

1.2.13 Bitrix24

Дозволяє підключати телефонію або орендувати номер. Система інтегрується з сайтами та іншими сервісами. Підтримує відправку листів прямо з CRM. В системі фіксується отримання і прочитання кожного email, а також зберігається історія листувань. Має засоби аналітики. Можливе інтегрування з інтернет магазином. Система має вбудований каталог

товарів та послуг. Також можлива інтеграція з поштою: дані з кореспонденції компанії будуть автоматично введені до CRM. Є можливість встановлювати завдання співробітникам, які обробляють транзакції.

Система дозволяє простежити шлях від ліда до укладення угоди. Лід це будь-який проявлений інтерес, початковий контакт з потенційним клієнтом. Дані про нього вводяться в систему, і закропляється відповідальний згодом Лід перетворюється на "Контакт", "Компанію" чи "Угоду".

CRM реєструє всі події та дії, виконані на цьому шляху. Крім того, система дозволяє створювати звіти, які аналізують ефективність цих дій. Для аналізу продажів CRM містить вісім стандартних звітів.

Також виконує функцію обліку робочого часу.

Дозволяє складати звіти про продажі, графіки операцій, які були завершені, та їх ймовірність укладення.

1.2.14 1C: CRM

1C: CRM пропонує «шаблонні» бізнес-процеси для роботи з клієнтською базою, але ви також можете редагувати та створювати їх самостійно без програмування. Можна прописати певні правила для бізнес-процесів: продажу, сервіс обслуговування; реагування на скарги; маркетингу. Платформа відкрита для редагування. Також швидкість обробки запитів є однією з найшвидших з конкурентів.

Система має стандартний набір звітності, проте в ній можна створювати й власні звіти, хоча задача ускладняється через недружелюбність системи і надлишковий функціонал.

Одна із особливостей аналітики даної системи "Наскрізна аналітика", що дозволяє відслідковувати не лише історію взаємодії клієнта з менеджером, а й відвідування сайтів, кількість натискань на рекламні банери, кількість

показів в пошукових системах та унікальних відвідувачів. Окрім цього система підтримує Яндекс Метрику, Google Analytics та інші сервіси.

1.2.15 Hyperion

Система надає інструментарій для ефективного управління бюджетом: порівняння стратегічних та середньострокових планів, пошук резервів для досягнення найбільш амбіційних цілей, виявлення та усунення відхилень у планах.

Основні функції:

а) визначення цільових показників управління бюджетом;

б) побудова моделей розвитку та планування ініціативи;

в) оцінка ресурсних потреб;

г) моніторинг та аналіз реалізації планів.

Hyperion дозволяє спеціалістам з економічних послуг, економістам залучати працівників усіх рівнів та підрозділів компанії до процесів бюджетування, забезпечуючи прозорість процесів бюджетного планування "зверху вниз" і "знизу вгору", одночасно підвищуючи рівень відповідальності менеджерів за досягнення поставлених фінансових результатів.

Hyperion надає керівникам миттєвий доступ до даних, необхідних для прийняття рішень з одного джерела інформації. Завдяки можливостям системи планування діяльності компанії відбувається в єдиному середовищі, консолідуються потреби відділів продажів у продуктах та послугах порівняно з можливостями операційних підрозділів.

1.2.16 Marketing analytic

На відміну від інших CRM-рішень, розробка яких починалася з операційних блоків, розробка була почата з аналітичної частини, призначеній для управління маркетингом.

Проблема більшості програм планування полягає в тому, що для того, щоб якісно виконувати свої функції, вони повинні поєднувати великі обсяги даних, розкиданих по всьому підприємству: дані про виробництво, дані про продажі, моніторинг зовнішнього ринку тощо. Як результат, маркетингові програми, призначені для розробки плану реалізує лише загальну методологію маркетингового планування і вимагає введення результатів готових досліджень, залишаючи процес аналізу "часткою" користувача— вимагають ручного введення великої кількості даних, пошук яких залежить від користувача.

Існують також маркетингові блоки, які дозволяють отримувати ряд звітів у межах даних, зібраних цими системами. Як правило, корпоративні системи ведуть облік лише внутрішньої роботи підприємства — продажів, відправлень, фінансових потоків тощо. Маркетингові інформаційні системи також повинні збирати інформацію про потенційних клієнтів, конкурентів та макросередовище. Крім того, системи управління маркетингом повинні містити аналітичні модулі, призначені для обробки цієї інформації та переведення її у форми, зручні для прийняття управлінських рішень.

1.2.17 SAS Marketing Automation

Рішення що дозволяє планувати, тестувати і проводити маркетингові кампанії. Користувачі можуть визначати цільові сегменти, визначати пріоритети правил вибору, вибирати канали зв'язку, аналізувати результати і вносити коригування для підвищення ефективності майбутніх кампаній.

1.2.18 BroadVision

Набір інструментів призначений для побудови інтернет-порталу, допомагає компанії організувати всі сфери діяльності, які вона може проводити через інтернет — побудова інформаційної частини порталу,

налаштування відносин з постачальниками та клієнтами, побудова платежів, аналіз маркетингових стратегій. Будь-яка частина порталу дозволяє персоналізацію відповідно до потреб користувача.

1.2.19 Microstrategy

Система включає компоненти: MicroStrategy Architect, MicroStrategy Agent, MicroStrategy Administrator.Продукт можна використовувати для створення та ефективного управління інтернет-порталами, магазинами, іншими системами. Система складається з трьох модулів.

MicroStrategy Web Business Analyzer — призначений для створення веб-сайту, що розробляється, розробки інтерфейсу, простого наповнення його вмістом, перегляду статистичних даних на веб-сайті тощо.

MicroStrategy Customer Analyzer — включає всі функції попереднього модуля і дозволяє працювати з даними для кожного клієнта окремо та в групах.

MicroStrategy Marketing Automation — дозволяє створювати та оцінювати маркетингові кампанії, організовувати цільовий Інтернет-маркетинг за групами клієнтів тощо. Інтегрується з попередніми модулями.

1.2.20 Sales force

Можливість інтеграції з різними додатками та інформаційними системами. Окрім того система основана на модульній архітектурі та легко розширюється та допрацьовується.

Пошук та індексація документів з різних джерел інформації. Портал включає сервер пошуку, який індексує дані, включаючи документи, що зберігаються на сервері співпраці, веб-сторінки, керовані сервером вмісту, та інші проіндексовані дані з файлових систем, веб-сайтів та баз даних

документів, розташованих у каталозі Документація. Сервер пошуку можна запустити з будь-якого комп'ютера.

Дозволяє налаштувати параметрів для відображення в документі.

Реалізує рольовий механізм адміністрування та контролю доступу. Доступ до кожного інформаційного ресурсу порталу може бути налаштований індивідуально для кожного користувача або групи користувачів. Існують інструменти для реєстрації дій користувачів. Адміністратор порталу може обмежити права користувача на зміну особистих налаштувань.

1.3 Порівняння CRM систем

За для порівняння CRM системи були розроблені наведені критерії.

Робота з угодою— зручність роботи з угодою, систематизування усіх договорів компанії, автоматичне узгодження, маршрутизації між співробітниками компанії, контролю термінів дії.

Задачі— маркетингові цілі, чи заходи що проводяться компанією за ради залучення потенційних клієнтів, покращення обслуговування чи продукту.

Права доступу— обмеження прав доступу до розділів чи записів розділу, прав на редагування, читання. Один з найважливіших критеріїв, оскільки менеджер може бачити лише продажі та угоди в яких бере участь і не має інформацію про продажі інших менеджерів. Розмежування прав доступу зберігає інформацію від крадіжки та зміни даних випадково чи з метою нашкодити компанії чи людині.

Звітність— інформація про продажі, договори, клієнтів, компанії, поставлених задач, історії взаємодії з клієнтом з метою подальшої аналітики та покращення роботи компанії.

API—посилання, що викликає сервісну логіку, критерій характеризує легкість підтримки, доопрацювання CRM системи, є найважливішим

критерієм у випадку інтегрування CRM системи в інші системи, веб-додатки, веб-портали та інтернет магазини

Можливість доопрацювання— характеризує доопрацювання системи під нужди компанії, створення нової бізнес-логіки чи модифікацію існуючої.

Розділення на Ліди і контакти— критерій необхідний для коректного складання воронки продажу та розмежування потенційних клієнтів та існуючих.

Варіанти поставки— варіант розгортання системи. Сучасні CRM системи пропонують два варіанти:

а) SAAS — розгортання в хмарі, на серверах компанії, що пропонує CRM систему, в цьому випадку оплата здійснюється щомісяця;

б) Stand alone — розгортання на сервері компанії, що покупає CRM. В цьому випадку оплата здійснюється одноразово, але у випадку обновлення системи може знадобитися додатковий платіж.

Складність системи— характеризує складність освоєння та застосування системи користувачем, чи потрібно проводити навчальні заходи та їх кількість для пояснення. як використовується дана CRM.

Ціна— критерій, що характеризує конкурентоспроможність та розмір бізнес для буде використовуватися дана CRM.

Критеріям "Робота з угодою" , "Задачі", "Права доступу", "Звітність", "API", "Можливість доопрацювання" буде виставлена оцінка від 0 до 3, де

0 — відсутній функціонал в системі

1 — функціонал присутній, проте зміна чи застосування його не можливо, в силу його складності чи навпаки примітивності.

2 — функціонал повністю доступний, але за для його застосування необхідно доручити спеціально навчений персонал.

3 — функціонал повністю доступний, його може застосувати користувач інтуїтивно, прослухавши курс навчання чи прочитавши навчальну документацію до даної CRM

Порівняльний аналіз CRM систем наведено у таблиці 1.4.

Таблиця 1.4 Порівняння CRM систем

	Creatio	*Bitrix 24*	*AMO CRM*	*1C:CRM*	*Mango CRM*
Робота з угодою	3	2	1	1	2
Задачі	2	3	1	3	1
Права доступу	2	2	1	2	1
Звітність	2	1	1	3	1
API	1	2	3	3	1
Можливість допрацювання	3	2	1	3	0
Ліди і	*Так*	*Так*	*Так*	*Ні*	*Ні*
Варіанти поставки	*SAAS і Stand alone*	*SAAS*	*SAAS*	*Stand alone*	*SAAS*
Складність системи	*Висока*	*Висока*	*Низька*	*Висока*	*Висока*
Ціна	*користувач/місяц*	*користувач/міс*	*корирустувач/мі*	*доларів за*	*користувач/міс*

Продовження таблиці 1.4

Onyx	*Oracle Siebel CRM*	*Microsoft Dynamics CRM*	*Megaplan CRM*	
2	*2*	*3*	*3*	*Робота з угодою*
1	*3*	*3*	*3*	*Задачі*
1	*2*	*2*	*2*	*Права доступу*
2	*2*	*3*	*1*	*Звітність*
2	*3*	*3*	*2*	*API*
2	*3*	*2*	*0*	*Можливість допрацювання*
Ні	*Ні*	*Так*	*Ні*	*Розділення на Ліди і контакти*
SAAS	*SAAS*	*SAAS*	*SAAS*	*Варіанти поставки*
Низька	*Висока*	*Висока*	*Висока*	*Складність системи*
15 — 30 доларів користувач/місяць	*80 — 120 доларів/користувач*	*16 — 110 доларів користувач/місяць*	*4 — 19 доларів користувач/місяць*	*Ціна*

Продовження таблиці 1.4

	Maximaizer	*GoldMine*	*Sales Logix*	*SAP*
Робота з угодою	2	*1*	2	*1*
Задачі	2	*1*	2	*2*
Права доступу	2	2	3	*1*
Звітність	2	2	2	*2*
API	2	3	3	*3*
Можливість допрацювання	2	2	3	*3*
Розділення на Ліди і контакти	*Так*	*Так*	*Так*	*Ні*
Варіанти поставки	*SAAS*	*SAAS*	*Stand alone*	*SAAS*
Складність системи	*Середня*	*Висока*	*Середня*	*Низька*
Ціна	*30 — 90 доларів користувач/місяць*	*26 — 55 доларів користувач/місяць*	*Потрібно придбати разово*	*20 — 30 долларів*

Продовження таблиці 1.4

	Sales Expert	*Парус*	*Hyperion*	*SAS Marketing Automation*
Робота з угодою	*1*	3	2	*1*
Задачі	*1*	3	3	3
Права доступу	2	2	2	*1*
Звітність	2	2	3	3
API	2	*0*	3	*1*
Можливість допрацювання	*1*	*0*	3	*0*
Розділення на Ліди і контакти	*Hi*	*Hi*	*Hi*	*Hi*
Варіанти поставки	Stand alone	*Stand alone*	*SAAS*	*SAAS*
Складність системи	*Середня*	*Середня*	*Висока*	*Висока*
Ціна	*15 — 20 доларів за користувача*	*18 — 40 доларів за користувача*	*130-140 доларів користувач/місяць*	*40 — 80 доларів користувач/місяць*

Продовження таблиці 1.4

	Broadvision	*Microstrategy*	*Salesforce*
Робота з угодою	*1*	*1*	3
Задачі	3	3	3
Права доступу	*1*	3	2
Звітність	*1*	3	2
API	*2*	3	3
Можливість допрацювання	*1*	2	3
Розділення на Ліди і контакти	*Hi*	*Hi*	*Так*
Варіанти поставки	SAAS	SAAS	SAAS
Складність системи	*Низька*	*Висока*	*Висока*
Ціна	*10 — 20 доларів користувач/місяць*	*80 — 150 доларів користувач/місяць*	*25 — 300 доларів користувач/місяць*

1.4 Висновки до розділу

1. У розділі описано математичні моделі, що використовуються в CRM системах. В ході порівняння можна зробити висновок, що кожна з моделей використовуються в залежності від цілей та задач використання CRM систем на підприємстві.

2. Розглянуті та проаналізовані основні типи CRM систем та їх основні особливості. Також були розглянуті існуючі CRM системи та розроблені критерії оцінки CRM систем. Відповідно до розроблених критеріїв можна зробити висновок, що є необхідність створення уніфікованої моделі корпоративної моделі CRM системи для підприємств оптової та роздрібної торгівлі.

Приклад 2. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему **«**Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»].

Огляд існуючих засобів для пошуку вакансій

Більшість існуючих засобів для пошуку роботи, мають чудовий функціонал, схожий набір переваг, та виконують своє основне завдання доволі непогано. Але вони також мають спільний перелік явних та важливих недоліків. Тому, було проведено аналіз основних характеристик відповідних сайтів для пошуку роботи.

В якості платформ альтернатив для пошуку роботи, було розглянуто наступний перелік варіантів веб-сайтів: Djinni.co, Jooble.ua, Robota.ua та Work.ua.

Таблиця 2.1 – Порівняння існуючих засобів для пошуку вакансій

	Djinni.co	*Jooble.ua*	*Robota.ua*	*Work.ua*
Наявність чат боту	*наявний*	*наявний*	*наявний*	*наявний*
Джерело вакансій	*внутрішнє*	*внутрішнє*	*внутрішнє*	*внутрішнє*
Аналітика вакансій	*детальна*	*частково*	*відсутня*	*відсутня*
Прогнозування ЗП	*відсутнє*	*відсутнє*	*відсутнє*	*відсутнє*

На основі результатів порівняння існуючих рішень з таблиці 2.1, можна зробити висновок, що для всіх платформ існує певний чат бот. Але відповідний бот рекомендує вакансії використовуючи власний сайт. Для підвищення ефективності пошуку роботи, було б доцільно розширити цей перелік.

Додатково, важливим недоліком всіх наведених платформ є відсутність алгоритму, який би дозволяв прогнозувати заробітну плату для вакансій в яких це поле відсутнє.

1.1 Огляд теоретичних рішень проблематики пошуку вакансій

1.1.1 Уніфікація джерел пошуку вакансій

Для проблеми уніфікації джерел пошуку вакансій, в загальному випадку, можна використати систему, що буде збирати, обробляти та рекомендувати вакансій. Залишається лише питання, в якому саме вигляді користувач буде взаємодіяти з системою такого класу. Найбільш популярними інтерфейсами для взаємодії з користувачем є програмні реалізаціє наступного вигляду:

- *Асистент;*
- *Веб-сайт;*
- *Чат бот.*

В загальному випадку, асистент є комплексною системою, що зазвичай створюють під певну лінійку платформ. Дана система, зазвичай, утилізує повний функціонал конкретної платформи, що перешкоджає легкому розгортанню асистента на більшій кількості доступних платформ.

Асистент є чудовим, потужним видом взаємодії з користувачем, що надає можливість для високого рівня персоналізації, але, в свою чергу, складним в реалізації та не універсальним інтерфейсом взаємодії з користувачем.

Натомість, веб-сайт є більш гнучким способом взаємодії з користувачем. Потужність системи можна варіювати за допомогою хостингу, на якому сайт буде розміщено. Універсальність такої системи забезпечується взаємодією через веб-браузер, який наявний на більшості платформ. Також, веб-сайти, є відносно легкими в реалізації, але, в свою чергу, способи оповіщення повинні бути реалізовані з залучення сторонніх програм, таких як електрона пошта, месенджери, тощо.

Чат-бот, як система, що базується на основі існуючого месенджера, може забезпечити легкий процес реалізації, варіацію в потужності, персоналізованість та легкий та зручний процес оповіщення користувача. Також, дана система не потребує окремої розробки дизайну, оскільки успадковує його від базового месенджера.

Отже, проведемо порівняння наведених альтернатив для інтерфейсів взаємодії з користувачем для пошуку роботи, за відповідними основними критеріями:

Таблиця 2.2 – Порівняння альтернатив для інтерфейсу взаємодії з користувачем

	Асистент	*Веб-сайт*	*Чат бот*
Складність реалізації	*складно*	*легко*	*легко*
Мультиплатформність	*вузька*	*широка*	*широка*
Персоналізованість	*висока*	*достатня*	*достатня*
Зручність оповіщення	*висока*	*низька*	*висока*
Необхідність дизайну	*повний дизайн*	*повний дизайн*	*мінімум*

На основі порівняння основних критеріїв, що наведено в таблиці 2.2, можна зробити висновок, що фаворитом є інтерфейс на основі чат боту. Оскільки дана альтернатива має легку реалізацію, широку мультиплатформність та низьку потребу у дизайні у порівняні з «асистентом». Також, у порівняні з «веб-сайтом», «чат бот» має високу зручність оповіщення та низьку потребу у дизайні інтерфейсу.

Єдиним недоліком чат боту, у порівняні з іншими альтернативами, а саме «асистентом», є нижчий рівень персоналізованості, але легка реалізація та відсутність потреби у дизайні є більш вагомими критеріями для системи з пошуку роботи даного класу. Тому, найкращою альтернативою є інтерфейс на основі чат боту.

1.1.2 Передбачення заробітної плати

Для вирішення проблеми відсутності поля «заробітна плата» у більшості наявних вакансій, необхідно припустити, що існує зв'язок між текстовим описом вакансії та заробітною платою. Дане припущення має сенс, оскільки опис вакансій містить інформацію про обов'язки та необхідні навики для даного вакантного місця. В свою чергу, навики та обов'язки повинні корелювати з значенням заробітної плати, оскільки ці дані є один з критеріїв визначення заробітної плати.

Тому, необхідно спробувати знайти взаємозв'язок між наявним текстовим описом вакансії та значенням заробітної плати. Для такого класу прогнозування широкий спектр методів та алгоритмів. Оскільки, аналіз текстових даних це досить комплексна задача, більшість потенційних методів та алгоритмів не можу бути обрано [1]. Основним класом алгоритмів, що можуть вирішити цю задачу є алгоритми для обробки природної мови (Natural Language Processing, NLP), що є підкласом методів штучного інтелекту (ШІ) [1, 2]. Тому було обрано саме методи NLP для вирішення задачі передбачення ЗП на основі текстового опису вакансій.

1.2 Огляд алгоритмів для передбачення на основі текстових даних

Оскільки, найкраще з обробкою текстових даних справляються методи на основі ШІ, а саме алгоритми NLP [1]. Серед найбільш популярних рішень, можна виділити три наступних методи прогнозування, що іноді виділяють як основні:

- *Наївний Баєсів класифікатор;*
- *Ансамблеві моделі типу random forest;*
- *Одновимірні згорткові нейронні мережі.*

1.2.1 Наївний Баєсів класифікатор

Наївний Баєсів класифікатор (Naïve Bayes Classifier, NBC) – це ймовірнісний алгоритм МН, який широко використовується для завдань класифікації в NLP [3]. NBC базується на теоремі Байєса, яка розраховує ймовірність гіпотези (або класу) за наявними доказами (або ознаками) як [4]:

$$P(E) = \frac{P(E|H) \cdot P(H)}{P(E)}, \qquad (2.4.1)$$

де $P(H)$ *– це попередня ймовірність гіпотези,*

$P(E|H)$ *– це ймовірність доказів, враховуючи гіпотезу,*

$P(E)$*– гранична ймовірність доказів,*

$P(H|E)$ *– апостеріорна ймовірність гіпотези за наявності доказів.*

У контексті NLP, NBC зазвичай використовується для завдань класифікації тексту, таких як аналіз настроїв, виявлення спаму та категоризація тем. Модель NBC припускає, що ознаки (слова) у вхідному тексті є умовно незалежними з урахуванням мітки класу, що означає, що поява одного слова не впливає на появу інших слів. Це припущення дозволяє спрощено обчислити член правдоподібності, що визначається за теоремою Байєса як:

$$P(H) = P(H) \cdot P(H) \cdot \ldots \cdot P(H), \qquad (2.4.2)$$

де $word_1$, $word_2$, ..., $word_n$ *– це слова у вхідному тексті.*

NBC можна навчити за допомогою промаркованого набору даних, де кожен одиниця текстової інформації пов'язана з певною міткою класу. Попередню ймовірність кожного класу можна оцінити за частотою міток класів у навчальному наборі. Імовірності ймовірності кожного слова в кожному класі можна оцінити за допомогою частоти слова в навчальному наборі для цього класу, нормалізованого загальною кількістю слів у цьому класі.

Даний метод має кілька переваг для завдань NLP, включаючи його простоту, масштабованість і здатність обробляти багатовимірні простори ознак [5]. Проте NBC має деякі обмеження, наприклад припущення про незалежність між функціями, яке в деяких випадках може

не відповідати дійсності, схильність до перенавчання, якщо навчальний набір даних занадто малий [5]. Також, після репрезентації даних у вигляді невпорядкованого набору слів, модель втрачає інформацію що існує в взаємо розміщенні слів у реченні [6].

1.2.2 Ансамблеві моделі типу random forest

Ансамблеві моделі типу random forest (RF) – це популярний алгоритм МН, який використовується для завдань класифікації та регресії для вирішення завдань NLP, таких як класифікація тексту, аналіз настроїв і розпізнавання іменованих об'єктів [7]. Модель RF – це методика ансамблевого навчання, яка поєднує кілька дерев рішень для створення більш надійної та точної моделі.

У випадку NLP, RF часто використовується як методика «навчання з вчителем», де вхідні текстові дані проходять попереднє кодування у вигляд векторів, використовуючи такі методи як Bag-Of-Words, n-grams, або Word Embeddings [8]. Потім алгоритм навчається класифікувати текстові дані на основі цих відповідних кодування у вигляді векторів, в залежності від репрезентації даних.

Алгоритм RF працює шляхом створення кількох дерев рішень, кожне з яких навчено на випадковій підмножині вхідних даних і випадковій підмножині вхідних функцій. Потім алгоритм об'єднує прогнози кожного з цих дерев, щоб отримати остаточний результат. Цей підхід зменшує перенавчання та підвищує продуктивність узагальнення та точність моделі.

Однією з переваг використання RF для завдань NLP є те, що ця модель може обробляти дані великої розмірності з розрідженими характеристиками, які часто зустрічаються в текстових даних. Крім того,

модель RF є стійкою до шуму та викидів у вхідних даних і може обробляти відсутні значення.

В загальному випадку, серед моделей на основі дерев прийняття рішень, ансамблеві моделі типу RF є одними з найкращих методів для широкого спектру класифікаційних задач [9]. Але, нажаль, їх результативність, має сильну залежність від якості текстових даних, що знаходяться в навчальному наборі.

1.2.3 Одновимірні згорткові нейронні мережі

Згорткові нейронні мережі (Convolutional Neural Network, CNN) показали великий успіх у задачах обробки зображень. Однак із розвитком глибокого навчання CNN також успішно застосовуються в задачах NLP. Одновимірні згорткові нейронні мережі (One Dimensional Convolutional Neural Network, 1D-CNN) є однією з популярних моделей, які використовуються в завданнях NLP, зокрема в завданнях класифікації тексту [10, 13].

Моделі 1D-CNN працюють з послідовними даними, такими як текст, де вхідними даними є послідовність токенів (слів). Ці моделі використовують одновимірні згорткові шари для вилучення функцій із вхідної послідовності. У контексті завдань NLP ці послідовності можуть представляти лінгвістичні властивості, такі як послідовності слів, n-grams та частини мови.

Базова структура моделі 1D-CNN для NLP зазвичай складається з шару кодування, за яким слідує один, або більше одновимірних згорткових шарів, а потім шар агрегування (pooling). Потім вихідні дані рівня об'єднання передаються на повнозв'язні шари для подальшої класифікації.

Шар кодування відображає вхідні токени в певний щільний векторний простір, де кожен токен представлений вектором фіксованої довжини. Цей рівень кодування зазвичай попередньо навчається на великому масиві текстових даних за допомогою таких методів, як Word2Vec або GloVe.

Згорткові шари застосовують набір фільтрів до закодованих у вектори токенів вхідної послідовності. Кожен фільтр ковзає по вхідній послідовності та обчислює скалярний добуток між вагами фільтра та вхідною послідовністю в кожній позиції. Це створює карту ознак для кожного фільтра.

Шар агрегування зменшує розмірність карт ознак, створених згортковими шарами. Найпоширенішою операцією для цього, в моделі 1D-CNN є «max pooling», яке бере максимальне значення з кожної карти ознак. Після чого, повнозв'язаний шар застосовує набір ваг до об'єднаних ознак і створює розподіл ймовірностей за можливими класами.

Основними переваги моделі 1D-CNN для завдань NLP є: висока здатні виділяти локальні патерни у вхідних даних, такі як «n-grams» та послідовності слів [10]; вимагають менше параметрів, ніж традиційні моделі NLP; швидше навчаються [11]; менш чутливі до вхідного шуму [13]; висока узагальнююча здатність обмеженого набору даних [13].

Основними недоліками моделі 1D-CNN є: низька здатність виділяти глобальні патерни у вхідних даних [12]; не так добре виконувати завдання, які вимагають глибшого розуміння послідовності текстових даних, наприклад, створення мови [12].

Моделі 1D-CNN є потужним інструментом для завдань NLP, зокрема для завдань класифікації тексту. Вони здатні проводити ефективну екстракцію корисної інформації та менш дорогі за обчисленнями, ніж традиційні моделі NLP. Однак вони можуть працювати не так добре, як інші моделі, у завданнях, які потребують глибшого розуміння текстових даних.

1.2.4 Аналіз описаних алгоритмів для передбачення

Узагальнивши основні переваги та недоліки розглянутих алгоритмів для передбачення на основі текстових даних, а саме: наївний Баєсів

класифікатор; ансамблеві моделі типу random forest; одновимірні згорткові нейронні мережі. Було створено порівняльну таблицю, що має наступний вигляд:

Таблиця 2.3 – Порівняння обраних методів прогнозування

	NBC	*RF*	*1D-CNN*
Складність реалізації	*низька*	*низька*	*висока*
Швидкість навчання	*висока*	*низька*	*низька*
Схильність до перенавчання	*висока*	*низька*	*низька*
Узагальнююча здатність	*низька*	*середня*	*висока*
Стійкість до шуму	*низька*	*висока*	*висока*
Вплив розміру датасету	*високий*	*середній*	*низький*
Вплив якості датасету	*високий*	*високий*	*середній*
Результативність	*середня*	*середня*	*висока*

На основі порівняння характеристик обраних методів прогнозування на основі текстових даних, що наведено в таблиці 2.3, можна зробити висновки, що незважаючи на складність в реалізації, для відповідного завдання, найкраще підходить модель на основі 1D-CNN. Оскільки дана модель є потужним інструментом для вирішення завдань NLP, зокрема для завдань класифікації тексту.

Даний вибір зумовлений, високими показниками узагальнюючої здатності, стійкості до шуму та результативності прогнозування, що буде корисним зважаючи на обмежений набір навчальних даних.

Додатково, дана модель, має низькі показники для схильності до перенавчання, вплив розміру та якості датасету, що є важливими характеристиками для вирішення завдання прогнозування з малим набором навчальних даних.

Тому, незважаючи на низькі показники швидкості навчання, модель 1D-CNN є найкращим рішення для вирішення завдання передбачення ЗП на основі текстових даних як частини системи чат боту для пошуку роботи.

1.3 Висновки до розділу

1. Виявлено основні проблеми при пошуку роботи, такі як велика кількість платформ та відсутність поля «заробітна плата» для більшості описів вакансій. Запропоновано створити систему, яка збиратиме вакансії з різних джерел та рекомендуватиме їх користувачам, для чого було вирішено використовувати чат бот як інтерфейс для взаємодії з користувачем.

2. Проаналізовано існуючі платформи для пошуку роботи, такі як Djinni.co, Jooble.ua, Robota.ua та Work.ua. Виявлено, що вони не мають функціоналу передбачення заробітної плати.

3. Для передбачення заробітної плати використано методи обробки природньої мови (NLP). Виділено підгрупу методів для аналізу та прогнозування на основі текстових даних, таких як наївний Баєсів класифікатор (NBC), ансамблеві моделі типу random forest (RF) та одновимірні згорткові нейронні мережі (1D-CNN).

4. Проведено аналіз та порівняння методів прогнозування та обрано, в якостосі основної, модель 1D-CNN, як найкращу варіацію. Ця модель має високу узагальнюючу здатність, стійкість до шуму та ефективність прогнозування, а також низькі показники швидкості навчання та схильності до перенавчання, що є важливими характеристиками.

6.1.3.6. Структурне представлення системи, підсистеми, програмного продукту, програмного застосунку.

Важливою частиною алгоритму формування Основної частини є розробка структурного представлення системи, підсистеми, програмного продукту, програмного застосунку, далі «системи». Принцип розробки структурного представлення системи – «від загального до часткового», тобто від Метамоделі системи до моделей конкретизації необхідного рівня деталізації. Структурне представлення - архітектура, діаграма класів, компонентна модель, тощо з чітко прописаними елементами і інтерфейсами між ними надають розробникам можливість упорядкувати порядок розробки окремих елементів, проводити верифікацію і валідацію результатів їх взаємодії, вчасно вносити корективи і проводити документування проміжних результатів.

Результати цього кроку дають обґрунтовані підстави для розробки /застосування математичного забезпечення з метою реалізації окремих компонентів системи і інтерфейсів між ними.

Також нагадуємо, що нумерація розділу і підрозділів цього прикладу відноситься лише до цього прикладу і до нумерації змісту навчального посібника ніякого відношення не має. Назви розділів і підрозділів прикладів виділено курсивом.

Приклад 1. [Для прикладу використані фрагменти розділу Структурне представлення системи до МД студента Івана САВЧУКА започатковані у співавторстві в процесі вивчення дисциплін «Основи наукових досліджень» та «НДР за темою МД» і виконання МД на тему «Метод системної інженерії уніфікованої моделі DevOps-системи для продуктів IT індустрії», викладач дисциплін та науковий керівник МД Маслянко Павло Павлович, к.т.н., доцент»].

3 МЕТОД СИСТЕМНОЇ ІНЖЕНЕРІЇ УНІФІКОВАНОЇ МОДЕЛІ DEVOPS-СИСТЕМИ ДЛЯ ПРОДУКТІВ ІТ-ІНДУСТРІЇ

3.1 Опис методу

Основна ідея методу системної інженерії DevOps-систем полягає у застосуванні методології системної інженерії та бізнес-профіля Еріксона–Пенкера для формалізації упорядкованого способу розроблення та експлуатації DevOps-систем для всіх стадій ЖЦ продуктів, товарів та послуг [13, 14, 15].

Метод системної інженерії базується на чотирьох основних категоріях [15, 16]:

1. «Система», як множини сутностей і відношень між ними необхідних і достатніх для забезпечення колаборації сутностей інженерії та підтримки життєдіяльності продуктів, товарів та послуг на всіх стадіях їх життєвого циклу при заданих умовах і обмеженнях, що в межах прийнятих припущень та обмежень моделює, власне, DevOps-систему.

2. «Життєвий цикл», що передбачає представлення генезису системи DevOps, від народження та до утилізації самої системи DevOps [14].

3. «Зацікавлені сторони», що передбачає формування вичерпної множини умов і вимог, які зацікавлені сторони висувають до системи DevOps.

4. «Бізнес-профіль», що формалізує представлення DevOps-системи на всіх стадіях життєвого циклу продуктів, товарів та послуг.

Однією з найпоширеніших моделей представлення діяльності є бізнес-профіль Еріксона–Пенкера [13] в контексті якого автори сформулювали чотири основні сутності формального представлення діяльності будь-якої бізнес-системи:

– Цілі (уособлюють мету діяльності системи та сформульовані як правила. Цілі можуть бути розбиті на підцілі, що досягаються завдяки реалізації процесів);

– Процеси (основні дії, що складають діяльність системи та призначені для досягнення мети відповідно до встановлених бізнес-правил. Процеси зазвичай підпорядковуються правилам, можуть змінювати стан вхідних ресурсів, а також продукувати нові ресурси – ресурси виходу системи згідно з умовами та вимогами, встановленими зацікавленими особами);

– Ресурси (фізичні, абстрактні чи інформаційні об'єкти, які система споживає, використовує, обробляє та продукує впродовж всієї своєї діяльності для досягнення мети);

– Правила (певні формалізовані обмеження, рамки, умови та вимоги тощо, що накладаються на процеси, а також описують характер зв'язків між ресурсами).

Така впорядкована множина формалізованих (зокрема, в нотації UML) сутностей і представлень системи DevOps на основі бізнес-профіля Еріксона–Пенкера є повною моделлю діяльності бізнес-системи DevOps, що враховує і мультидисциплінарність концепту DevOps, і вимоги всіх зацікавлених осіб.

Метод інженерії DevOps-систем (надалі – метод) на основі застосування технік системної інженерії складається з п'яти основних етапів.

1. Формалізація класу/об'єкта класу розробки і підтримки продуктів, товарів та послуг. Розробка технічного завдання на розробку класу/об'єкта класу розробки і підтримки продуктів, товарів та послуг

2. Формалізація моделі DevOps-системи на основі бізнес-профіля Еріксона–Пенкера для визначеного класу/об'єкта класу розробки і підтримки продуктів, товарів та послуг;

3. Формалізація структурного представлення моделі DevOps-системи для визначеного класу/об'єкта класу розробки і підтримки продуктів, товарів та послуг – діаграма компонентів, та інші діаграми структурного представлення за необхідності;

4. Формалізація динамічного представлення моделі DevOps-системи для визначеного класу/об'єкта класу розробки і підтримки продуктів, товарів та послуг – діаграма діяльності, та інші діаграми динамічного представлення за необхідності;

5. Імплементація, верифікація та валідація методу системної інженерії DevOps-систем для всіх стадій ЖЦ продуктів, товарів та послуг.

Далі детальніше про суть кожного з етапів методу.

3.2 Формалізація класу/об'єкта класу та модель життєвого циклу розробки і підтримки продуктів, товарів та послуг

Зміст першого етапу методу містить повний пакет підготовки проектної і проектно-конструкторської документації на обраний клас/об'єкт класу та модель життєвого циклу розробки і підтримки продуктів, товарів та послуг. Зокрема це технічні умови і технічні вимоги до обраного класу/об'єкту класу та модель/стадія життєвого циклу розробки і підтримки продуктів, товарів та послуг у відповідності до вимог стейкхолдерів. Основний, заключний документ, це технічне завдання на розробку та підтримку обраного класу/об'єкту класу на визначених стадіях життєвого циклу обраного класу/об'єкту класу продуктів, товарів та послуг.

3.3 Формалізація моделі DevOps-системи на основі бізнес-профіля Еріксона–Пенкера

На цьому етапі, модель DevOps-систем моделюють на основі бізнес-профіля Еріксона–Пенкера [13, 17] на рисунку 3.1 і на основі результатів формалізації сутностей класу/об'єкта класу та моделі життєвого циклу розробки і підтримки продуктів, товарів та послуг, встановлюють проблему, мету, ресурси, процеси, цілі, правила, проектують структурне

представлення в контексті, означеннях і моделях представлень області DevOps-систем з урахуванням інтересів усіх зацікавлених сторін.

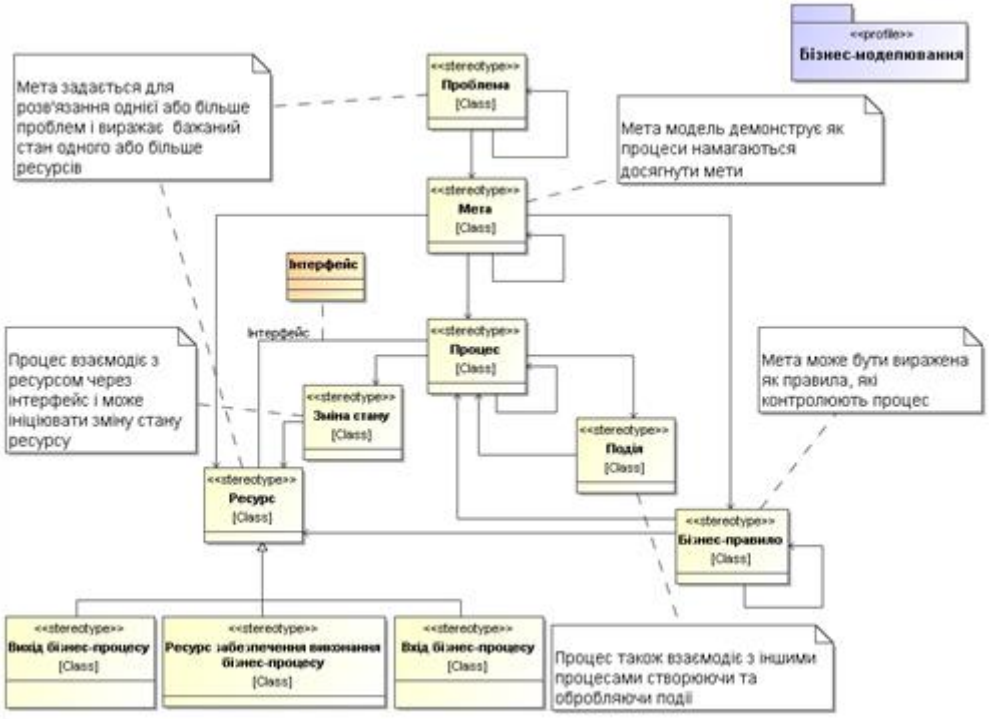


Рисунок 3.1 – Удосконалений бізнес-профіль Еріксона–Пенкера. Діаграма класів у нотації UML [13, 17]

Визначимо зміст кожного з класів діаграми на рисунку 3.1 у термінах постановки задачі системної інженерії DevOps-систем.

1. Проблема (актуальна множина питань, що потребують відповідних рішень, основна мотивація розроблення DevOps-систем, яка спонукає до формулювання конкретної мети. Проблема цієї роботи: необхідність підвищення продуктивності інженерії та підтримки життєдіяльності продуктів, товарів та послуг на всіх стадіях їх життєвого циклу при заданих умовах і обмеженнях.

2. Мета (виражає глобальну ціль роботи, покликану розв'язати поставлену проблему. Мета цієї роботи: Метою цієї роботи є розроблення, формалізація та імплементація методу системної інженерії DevOps-систем

для всіх стадій життєвого циклу продуктів, товарів або послуг на основі методології системної інженерії та бізнес-профіля Еріксона–Пенкера

3. Процес (множина процесів діяльності системи, внаслідок якої досягають мети, чітко визначена послідовність дій/під процесів, що призводить до виконання певного завдання. Процесами цієї DevOps-системи є [2]: процеси реалізації міждисциплінарної методології управління колаборацією сутностей інженерії і підтримки життєдіяльності продуктів, товарів та послуг на всіх стадіях їх життєвого циклу при заданих умовах та обмеженнях. Модель внутрішньої структури класу «Процес» бізнес профіля Еріксона-Пенкера показано на рисунку 3.2.

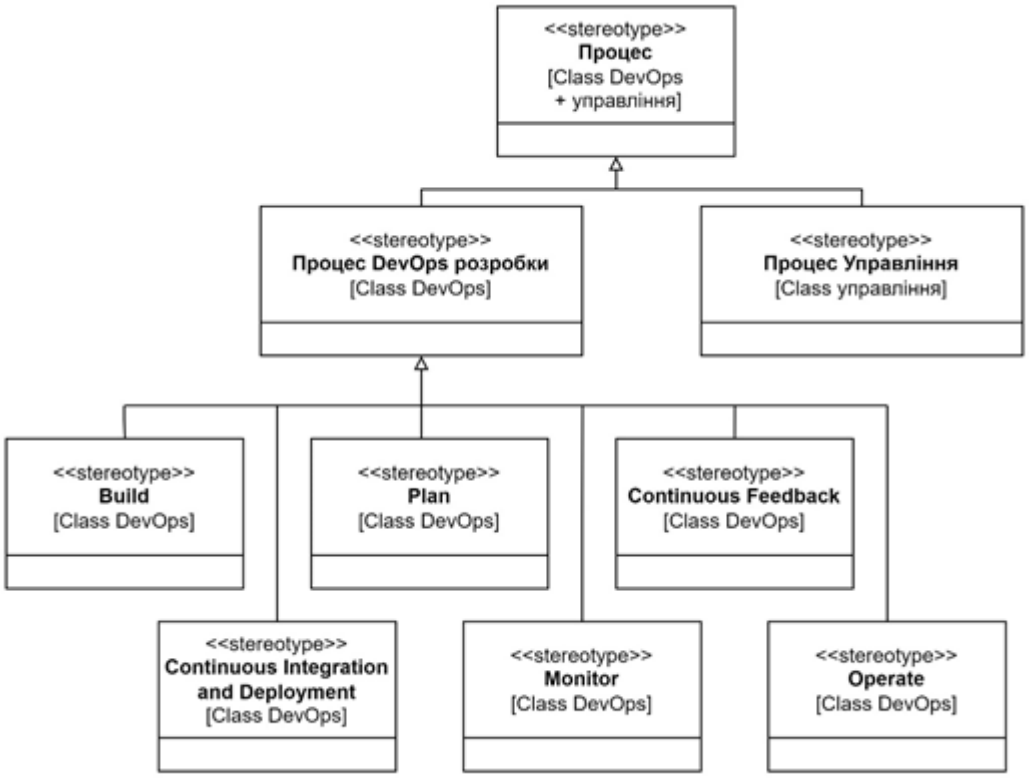


Рисунок 3.2 – Модель внутрішньої структури класу «Процес» бізнес профіля Еріксона-Пенкера

Процес DevOps розробки – ця абстракція описує процес розробки за методологією DevOps. Активна співпраця команд, їх колаборація та

комунікація мають призводити до підвищення якості та ефективності розробки.

Процес управління – це множина видів діяльності які потрібні для забезпечення координації та систематизації функціонування цілої системи DevOps.

Окрім цього, детальніше опишемо складові процесу DevOps розробки [2].

Планування – на етапі планування команди DevOps розробляють, визначають та описують особливості та можливості додатків та систем, які вони створюють. Вони відстежують прогрес на низькому та високому рівнях деталізації-від завдань окремого продукту до завдань, які охоплюють портфоліо різних продуктів. Створення резервних копій, відстеження помилок, керування гнучкою розробкою програмного забезпечення за допомогою Scrum, використання дошок Kanban та візуалізація прогресу за допомогою інформаційних панелей. Підходи які використовують: Agile, Kanban, реактивне програмування [7].

Збирання – ця фаза включає всі аспекти кодування - написання, тестування, перегляд та інтеграцію коду учасниками команди - а також побудову цього коду в артефакти побудови, які можна розгортати в різних середовищах. Команди DevOps прагнуть швидко впроваджувати інновації без шкоди для якості, стабільності та продуктивності. Для цього вони використовують високопродуктивні інструменти, автоматизують повсякденні та ручні кроки, а також повторюють невеликі кроки за допомогою автоматизованого тестування та безперервної інтеграції. Інструменти які використовують: Docker, Kubernetes [7].

Неперервна інтеграція та розгортання – дозволяє частіше випускати якісні продукти, прямо з репозиторію в production. Команди розробки можуть часто змінювати код, випускати нові версії та впроваджувати найкращі практики тестування. Інструменти які використовують: Jenkins, GitLab [18].

Моніторинг – означає постійний нагляд та негайне вирішення проблем, що виникають і які впливають на швидкість роботи продукту та його функціональність. Працівники мають негайно отримувати повідомлення про оновлення, ризики, роботи, та проблеми. Інструменти які використовують: Nagios, Prometheus [18].

Діяльність – означає постійну доставку послуг клієнтам. Це включає в себе практику проектування, налаштування, розгортання та обслуговування всієї IT інфраструктури, що підтримує послуги організації [19].

Неперервний зворотній зв'язок – командам DevOps слід оцінювати кожен випуск та формувати звіти для покращення майбутніх версій. Постійно збираючи відгуки, команди можуть покращити свої процеси та включити побажання клієнтів для покращення наступної версії [7].

4. Зміна стану (можливі зміни певних ресурсів внаслідок роботи процесів. DevOps-система може мати декілька змін станів задіяних ресурсів в залежності від реалізації процесів управління колаборацією сутностей інженерії і підтримки життєдіяльності продуктів, товарів та послуг на всіх стадіях їх життєвого циклу при заданих умовах та обмеженнях;

5. Ресурс (будь-які сутності (матеріальні чи нематеріальні), що їх споживає та продукує розроблювана система. Модель внутрішньої структури класу «Ресурс» бізнес профіля Еріксона-Пенкера показано на рисунку 3.3.

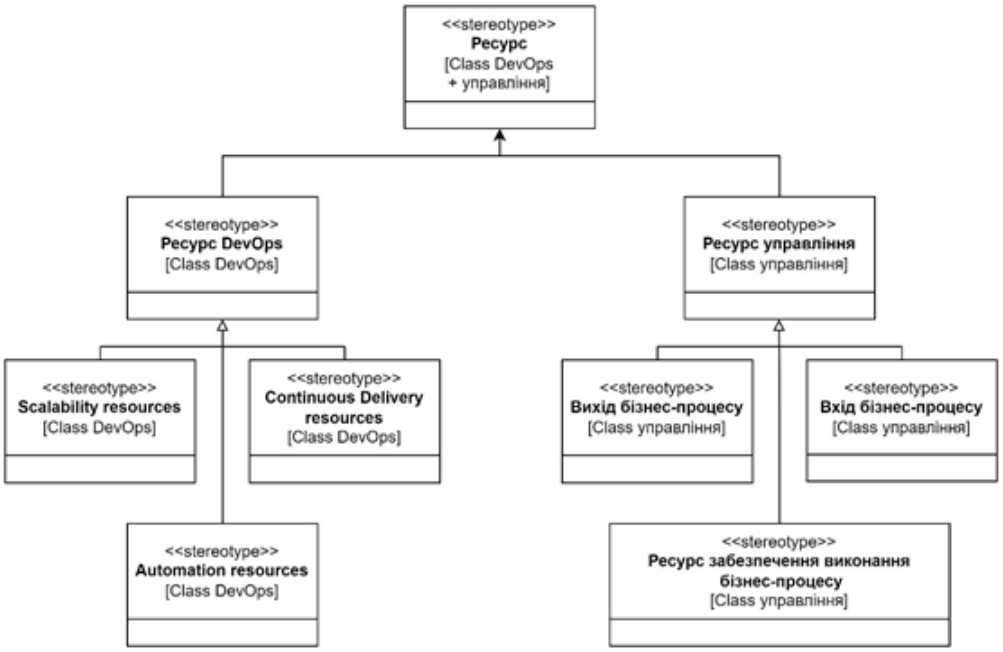


Рисунок 3.3 – Модель внутрішньої структури класу Ресурс бізнес профілю Еріксона-Пенкера

Ресурси масштабування – це ресурси, які дозволяють виконувати масштабування роботи певної команди / системи. Масштабування зазвичай відбувається вертикально (коли збільшується потужності кожного окремого компонента) та горизонтально (збільшення потужності команди / системи досягається шляхом додавання нових компонентів).

Ресурси автоматизації – ресурси, які дозволяють підвищити ефективність роботи шляхом зменшення участі людей в процесах. Зазвичай такими ресурсами є різноманітне програмне забезпечення функціонування команди та засобів розробки.

Ресурси неперервної доставки – ресурси, які необхідні для того аби забезпечити якомога швидшу доставку продукту до користувача, інтеграцію коду та занесення в існуючі репозиторії, автоматичне виконання збирання та тестування проекту і тд. В якості таких ресурсів наразі виступають різноманітні хмарні платформи та їх програмне забезпечення.

6. Ресурси найнижчого рівня ієрархії, що беруть безпосередню участь у процесах, також поділяють за характером впливу на перебіг процесів на такі три класи:

Вихід бізнес-процесу (ресурси, що їх продукує DevOps-система, кінцевий результат її функціонування. До них належать метрики результатів процесів управління колаборацією сутностей інженерії і підтримки життєдіяльності продуктів, товарів та послуг на всіх стадіях їх життєвого циклу при заданих умовах та обмеженнях;

Ресурс забезпечення виконання бізнес-процесу (ресурси, що забезпечують виконання процесів, але не є кінцевим результатом роботи, як правило це ресурси Орг.С необхідні для реалізації бізнес-процесів діяльності DevOps-системи;

Вхід бізнес-процесу (первинні ресурси входу початкових процесів, які ініціалізують цикл роботи системи: ресурси продукування, підтримки, управління та візуалізації результатів управління колаборацією.

7. Подія (виникає через певні зовнішні фактори чи як результат взаємодії між процесами. Потенційними подіями цієї системи вважають зміну завдань управління колаборацією, зміну множини інтегрованих міждисциплінарних ресурсів, зміну налаштувань візуалізації результатів роботи DevOps-системи.

8. Бізнес-правило - формальні інструкції, що регулюють, обмежують, встановлюють контекст і рамки функціонування процесів. Приклад бізнес-правила DevOps-систем: правила, алгоритми, умови та обмеження, що накладаються на процеси управління колаборацією, зміну множини інтегрованих міждисциплінарних ресурсів, зміну налаштувань візуалізації результатів роботи DevOps-системи, тощо. Модель внутрішньої структури класу «Бізнес-правило» бізнес профіля Еріксона-Пенкера показано на рисунку 3.4.

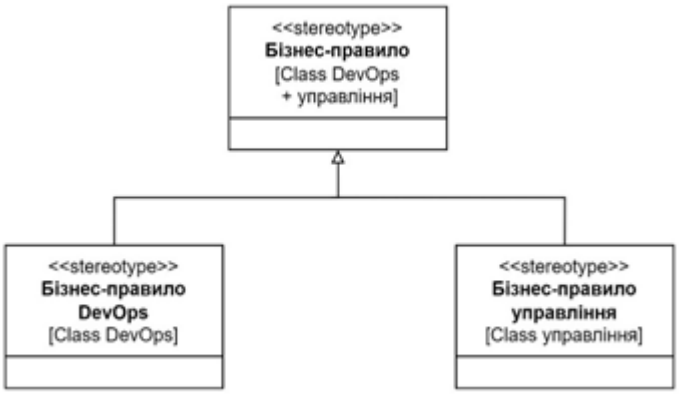


Рисунок 3.4 – Модель внутрішньої структури класу Бізнес-правило бізнес профілю Еріксона-Пенкера

В організації має бути чітке розділення бізнес правил на розробку та бізнес правил управління, бо ці дві окремі ланки є однаково ключовими, але досить різними. Тож наведемо визначення того, як трактувати ці поняття.

Бізнес-правило DevOps – стосуються безпосередньо процесу розробки DevOps, вони впливають на те, яка буде колаборація на проекті. Ці бізнес правила можуть стосуватись: функціональних можливостей об'єкта розробки; час відведений на розробку (час на один Agile спринт); обмежень на розмір команди; обмеження на затребувані ресурси та використовувані технології і тд.

Бізнес-правило управління DevOps – мають перед собою ціль чітко регулювати процес управління, як має бути вибудована взаємодія в компанії, як буде виконуватись процес прийняття рішень. Окрім цього, такі бізнес правила можуть визначати вектор надання послуг компанії: клієнтів яким будемо надавати / не надавати послуги; з якими сферами взаємодіяти / не взаємодіяти і т. ін.

3.4 Формалізація структурного представлення моделі DevOps-системи

На основі Уніфікованої моделі DevOps-системи [11] формалізуємо структурне представлення Моделі DevOps-системи у відповідності до стадій життєвого циклу DevOps-системи [2], рисунок 3.5.

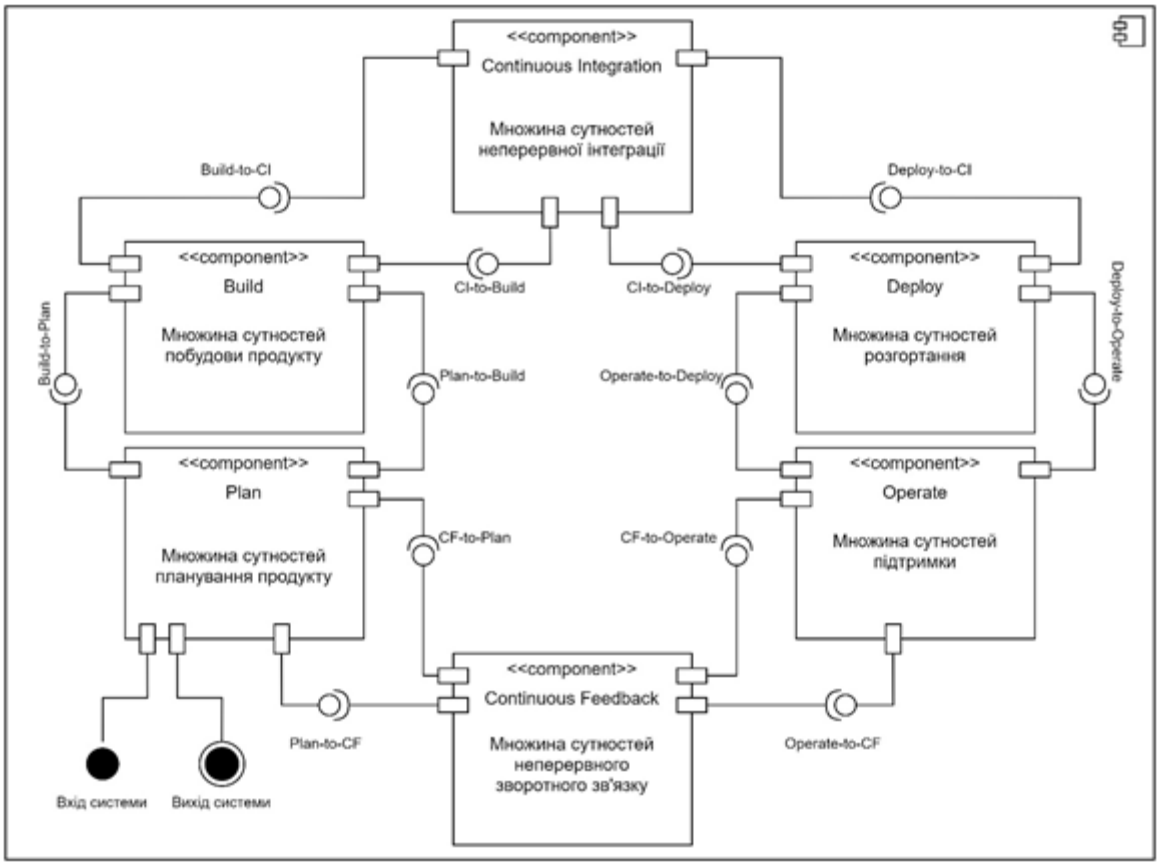


Рисунок 3.5 – Уніфікована модель DevOps-системи. Деталізована діаграма компонентів у відповідності до стадій життєвого циклу DevOps-системи [2].

Спираючись на попередні викладки, покажемо опис кожного компоненту та інструменти їх реалізації.

Множина сутностей планування продукту. На етапі планування команди, що приймають участь в DevOps розробці, визначають та описують бажані можливості продуктів які вони створюють, визначають технічні умови та технічні вимоги, на низькому рівні деталізації визначають

перелік завдань які необхідно виконати для інженерії продукту. Все це супроводжується детальним документуванням отриманих результатів, активно використовуються такі інструменти як Jira, Trello, Slack, DropBox та інші подібні інструменти які забезпечують комунікацію та обмін інформацією не тільки між сутностями планування, а також між усіма іншими компонентами. Одним з найважливіших артефактів результату планування є технічне завдання.

Множина сутностей побудови продукту. Ця фаза включає в себе всі аспекти розробки програмного продукту, такі як: програмний дизайн, написання коду, тестування та інші. Вся розробка відбувається строго з урахуванням технічного завдання визначеного на етапі планування, однак в результаті колаборації між всіма іншими сутностями DevOps, команди можуть змінювати умови, вимоги та завдання динамічно. Інструментами найчастіше виступають різноманітні середовища розробки, фреймворки та пакети прикладних програм: PyCharm, VSCode, Selenium, Docker та інші. Результатами побудови є версії продукту, а також різна інформація яка з'являється в процесах розробки яка допомагає іншим компонентам розуміти різні аспекти імплементації.

Множина сутностей неперервної інтеграції. Після розробки, обов'язково необхідно проводити інтеграцію та злиття різних частин програмного продукту, це можуть бути частини програмного коду, різні множини даних та інші артефакти розробки. Суть даного компоненту полягає в тому аби забезпечити уніфікованість та відповідність. На даному етапі ретельно перевіряється код та відслідковуються помилки, в разі виявлення яких слідує їх негайне вирішення. Серед інструментів часто зустрічаються GitHub, GitLab та інші.

Множина сутностей розгортання. Важливим етапом є розгортання продукту, він включає в себе налагодження автоматизації розгортання продукту, саме цей аспект розкриває одну з найсильніших сторін методології DevOps, а саме забезпечення частих випусків програмного

продукту, що надає можливості для якісного вдосконалення продукту. Для процесів розгортання використовують такі інструменти як Jenkins, Kubernetes, Docker та інші.

Множина сутностей підтримки. Ця частина DevOps означає постійну доставку послуг клієнтам. Це включає в себе різноманітні налаштування, тестування, обслуговування всієї інфраструктури продукту, та навіть всієї організації. В якості інструментів використовують Jenkins, Nagios, Splank, DataDog, Kubernetes, Ansible та інші.

Множина сутностей неперервного зворотного зв'язку. Для того аби створювати більш якісний продукт, командам варто оцінювати кожен випуск свого продукту. Для цього необхідно постійно збирати відгуки, проводити аналіз метрик та оцінювати ступінь відповідності поставленим технічним вимогам. Для даних цілей можна використовувати Google Forms, Google Spreadsheets, Jira, Slack та інші.

Перелік усіх інтерфейсів DevOps системи, можна формалізувати так, як описано нижче:

– *CF-to-Plan – інтерфейс передавання інформації про функціонування продукту, сценарії використання, відповідність вимогам та правилам бізнесу, задоволення потреб зацікавлених осіб;*

– *Plan-to-CF – інтерфейс для уточнення інформації про можливості продукту, вимог, сценаріїв використання;*

– *Plan-to-Build – інтерфейс передавання інформації про технічні вимоги, умови, задачі яких необхідно дотримуватись та виконати для створення продукту. Окрім цього слугує для спорядження відповідними ресурсами;*

– *Build-to-Plan – інтерфейс служить для того, щоб надавати інформацію про прогрес у створенні продукту, важливі результати, труднощі та дотримання вимог. Окрім цього може надаватись технічна інформація для кращого розуміння процесів створення продукту;*

Представлення даних інтерфейсів та їх взаємодія з компонентами представлена на рисунку 3.6.

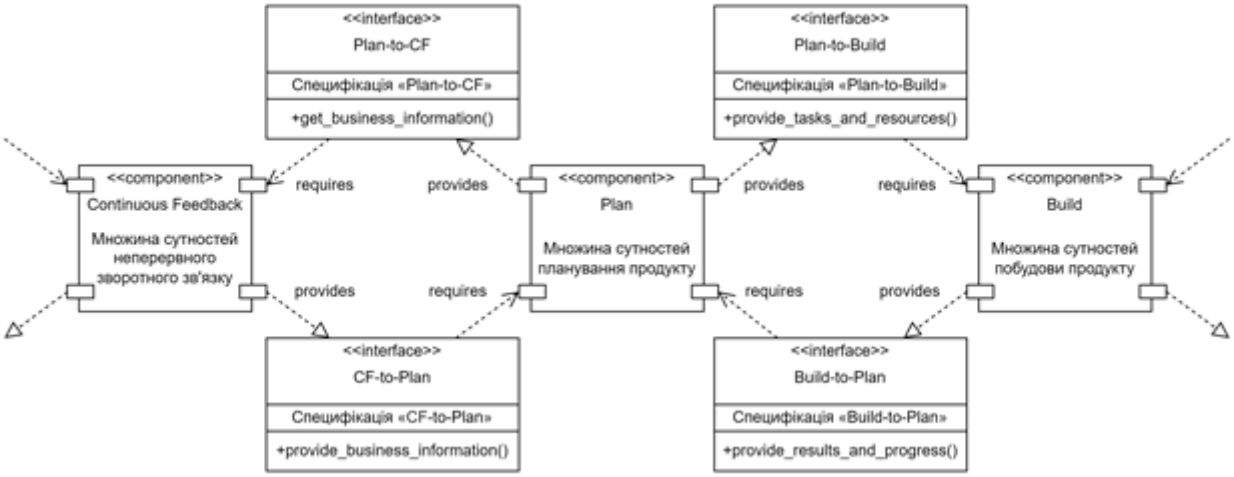


Рисунок 3.6 – представлення інтерфейсів Plan-to-CF, CF-to-Plan, Plan-to-Build, Build-to-Plan та взаємодія компонентів з ними

– Build-to-CI – інтерфейс передавання версій програмного продукту, програмного коду та інших артефактів які виникли під час розробки програмного продукту;

– CI-to-Build – інтерфейс передавання інформації про результати проведеної інтеграції. Надаються розгорнуті коментарі, зауваження, уточнення, побажання до отриманих артефактів;

– CI-to-Deploy – інтерфейс передавання версій продукту на випуск. Надається необхідна інформація про програмний продукт, його особливості, архітектуру та вимоги до розгортання;

– Deploy-to-CI – інтерфейс передавання інформації про розгортання. Вказується інформація про те як було розгорнуто застосунок, особливості проведеного розгортання, аналітика по лог-файлам;

Представлення даних інтерфейсів та їх взаємодія з компонентами представлена на рисунку 3.7.

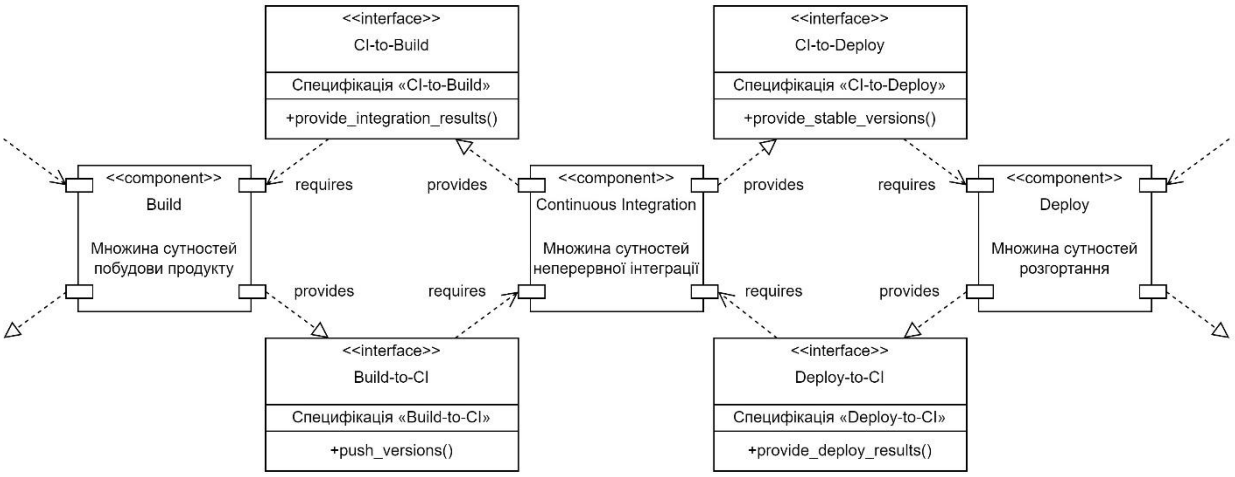


Рисунок 3.7 – представлення інтерфейсів CI-to-Build, Build-to-CI, CI-to-Deploy, Deploy-to-CI та взаємодія компонентів з ними

– *Deploy-to-Operate – інтерфейс передавання розгорнутого продукту на обслуговування та підтримку. Надається технічна інформація про середовище розгортання, використані засоби, та можливості моніторингу стану продукту;*

– *Operate-to-Deploy – інтерфейс передавання інформації про обслуговування продукту, інформація про аномалії та несправності;*

– *Operate-to-CF – інтерфейс для передавання інформації про вимоги, метрики та функції які необхідно дослідити та проаналізувати;*

– *CF-to-Operate – інтерфейс для сповіщення про технічний аналіз вимог, метрик та функції. Вказується інформація про несправності, аномалії, побажання та досвід користувачів;*

Представлення даних інтерфейсів та їх взаємодія з компонентами представлена на рисунку 3.8.

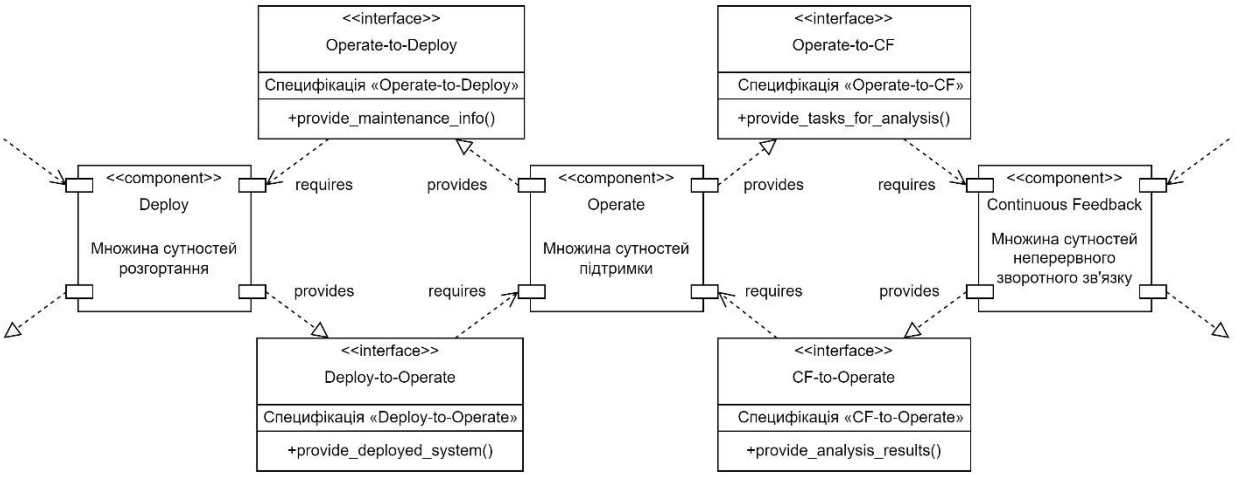


Рисунок 3.8 – Представлення інтерфейсів Operate-to-Deploy, Deploy-to-Operate, Operate-to-CF, CF-to-Operate та взаємодія компонентів з ними

3.5 Формалізація динамічного представлення моделі DevOps

Формалізуємо модель DevOps-системи у вигляді діаграми діяльності, зображену на рисунку 3.9.

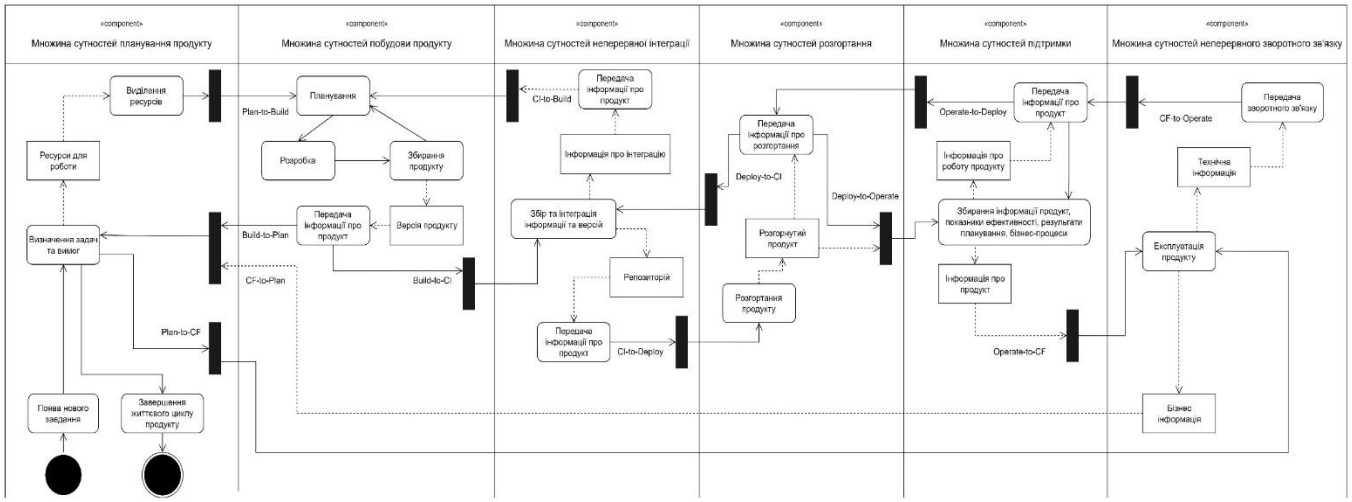


Рисунок 3.9 – Уніфікована модель DevOps-системи. Деталізована діаграма діяльності у відповідності до стадій життєвого циклу DevOps-системи [14]

Таким чином формалізація уніфікованої моделі DevOps-системи у вигляді діаграми компонентів і специфікацій інтерфейсів між цими

компонентами рисунку 3.5 та формалізація уніфікованої моделі DevOps-системи у вигляді діаграми діяльності цих компонентів рисунку 3.9 є результатом продукування Уніфікованої моделі DevOps-системи для обраної моделі життєвого циклу DevOps-системи [11, 2].

З урахуванням класу/об'єкту класу продуктів, товарів та послуг на всіх стадіях їх життєвого циклу при заданих умовах і обмеженнях, її складності та необхідності в деталізації, можна проводити розширення представлень компонентів моделі з (рис. 9) на діаграми нижчого рівня. На нашу думку, особливої деталізації потребують саме компоненти «Множина сутностей планування продукту» та «Множина сутностей побудови продукту», їх представлення можна надати, наприклад, за допомогою діаграм зображених на рисунку 3.10 та рисунку 3.11 відповідно.

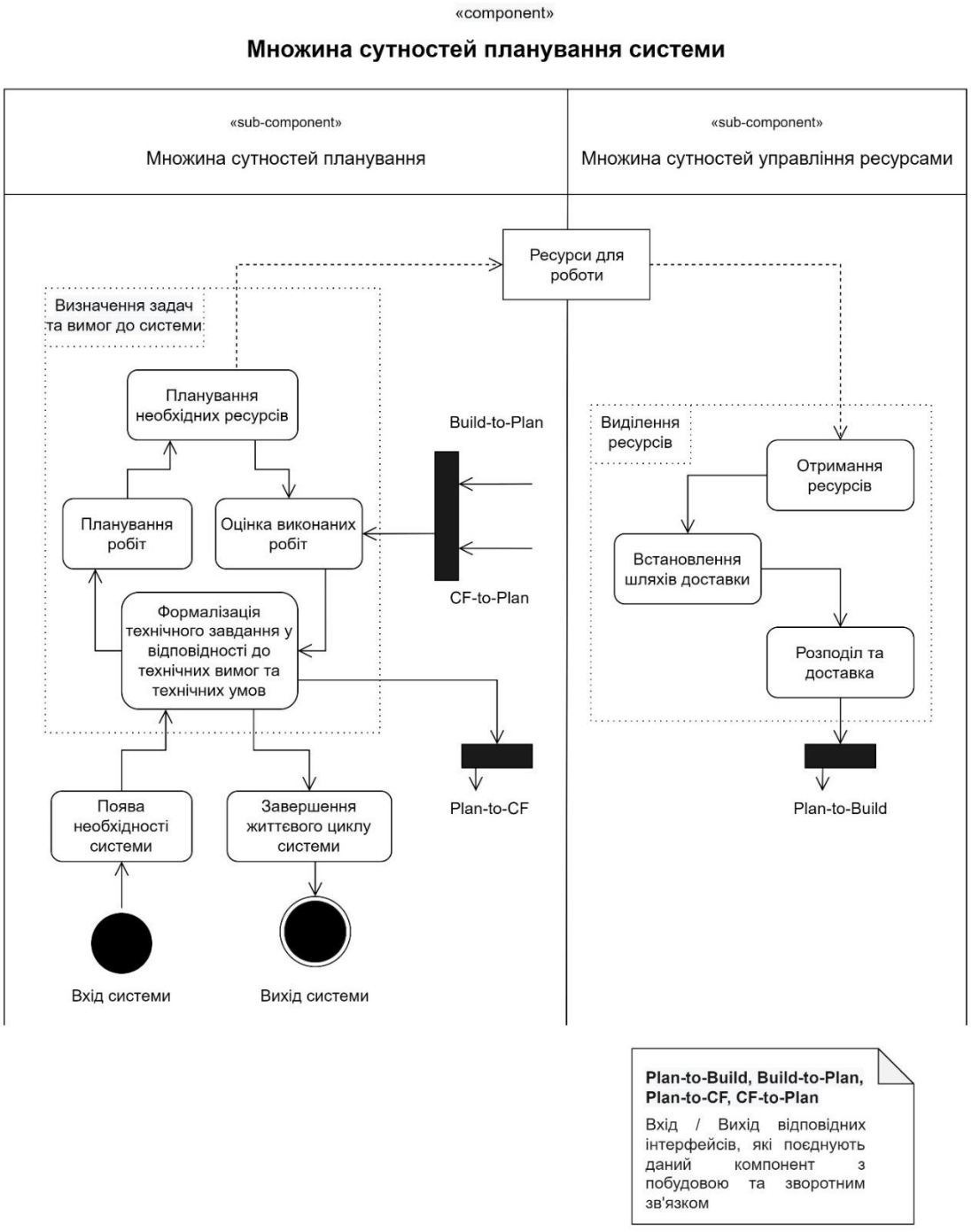


Рисунок 3.10 – Модель компонента «Множина сутностей планування продукту», розширене представлення.

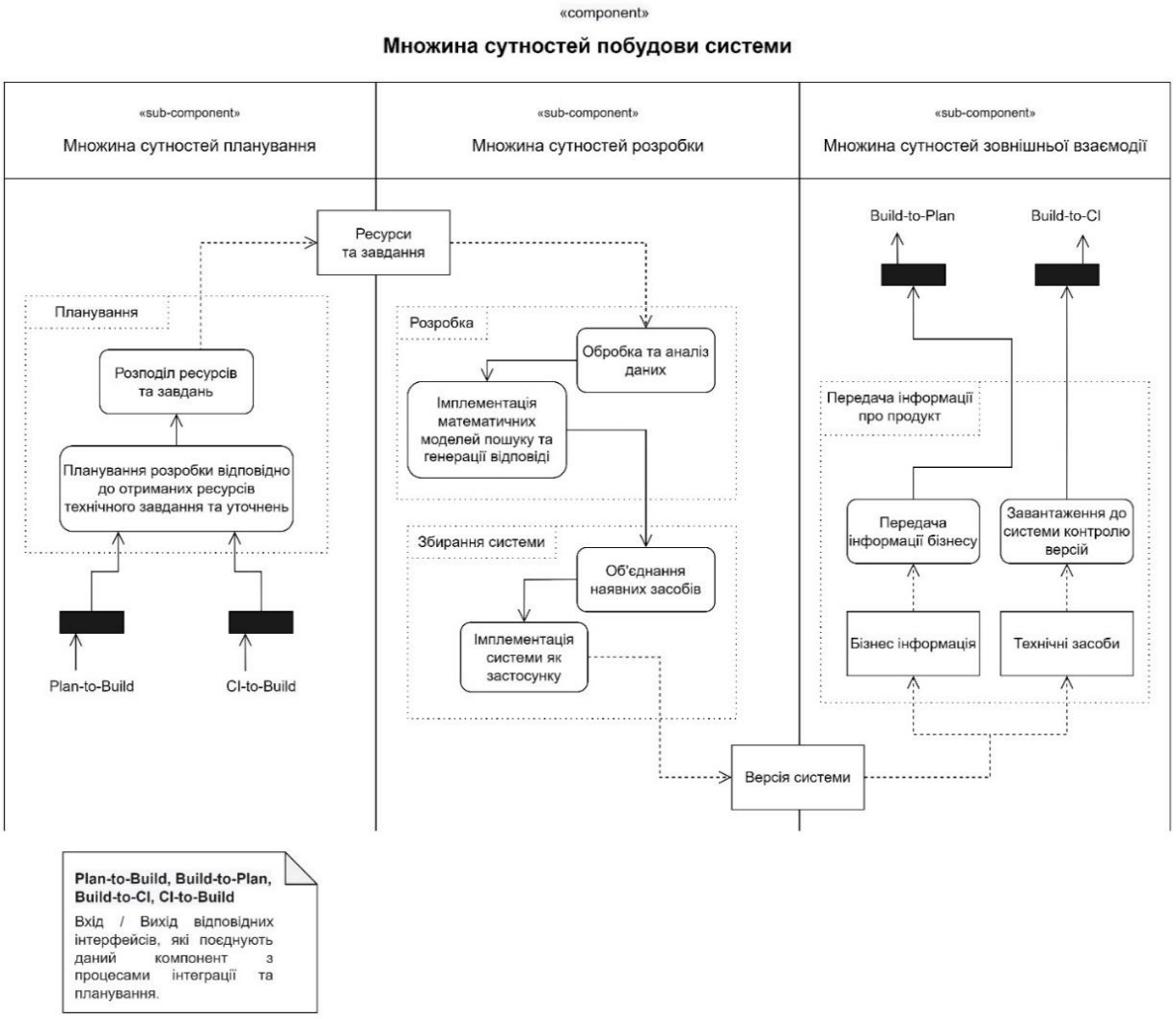


Рисунок 3.11 – Модель компонента «Множина сутностей побудови продукту», розширене представлення

3.6 Висновки до розділу

В розділі 3:

1. Формалізовано метод метод системної інженерії уніфікованої моделі DevOps-системи для продуктів IT індустрії.

2. Спроектовано представлення для окремих частин запропонованого методу у вигляді діаграм, з різним рівнем деталізації класів, підкласів, інтерфейсів, процесів та компонентів.

3. Запропоновані інструменти реалізації окремих частин даного методу в контексті його застосування для інженерії продуктів IT-індустрії.

6.1.3.7. Математичне забезпечення системи, підсистеми, програмного продукту, методи, моделі та алгоритми результатів досліджень

Математичне забезпечення для реалізації окремих компонентів і інтерфейсів між ними формується на основі обраних і обґрунтованих математичних моделей і алгоритмів попереднього розділу.

Нагадаємо, що математичне забезпечення це система математичних методів, моделей і алгоритмів призначених для обробки передачі і збереження інформації в інформаційно-комунікаційних системах. Тут ключовим терміном є слово «система», що чітко вказує на обов'язкову інтеграцію математичних методів, моделей і алгоритмів у, власне, систему. Тому математичне забезпечення має бути і відповідним чином описане і задокументоване.

Оскільки більшість математичних інструментів є запозиченими то важливо продемонструвати особливості їх застосування і взаємодії окремих математичних моделей і результатів їх роботи.

Для забезпечення функціональності системи, та її властивостей розширюваності та масштабованості, вкрай необхідно розробити такі інтерфейси між компонентами системи, що задовольняли б будь яку внутрішню структуру того, чи іншого компонента. Іншими словами система має функціонувати при будь якій реалізації внутрішньої структури компонентів.

Також нагадуємо, що нумерація розділу і підрозділів прикладу відноситься лише до цього прикладу і до нумерації розділів змісту навчального посібника ніякого відношення не має. Назви розділів і підрозділів прикладів виділено курсивом.

Приклад 1 [Для прикладу використані фрагменти МД студента Івана САВЧУКА започатковані у співавторстві в процесі вивчення дисциплін «Основи наукових досліджень» та «НДР за темою МД» і виконання МД на тему «Метод системної інженерії

уніфікованої моделі DevOps-системи для продуктів IT індустрії», викладач дисциплін та науковий керівник МД Маслянко Павло Павлович, к.т.н., доцент»].

4 ІМПЛЕМЕНТАЦІЯ МОДЕЛІ СИСТЕМНОЇ ІНЖЕНЕРІЇ СИСТЕМ DEVOPS ДЛЯ ПРОДУКТІВ IT-ІНДУСТРІЇ

Покажемо результати застосування методу системної інженерії DevOps-системи для продукування, підтримки та управління процесами продукування і підтримки систем відповіді на питання медичного напряму.
Система відповіді на питання (Question Answering System, QA system) [20] – це автоматизація процесів отримання правильних відповідей на запитання, які задає людина природною мовою. Фундаментальна ідея системи QA систем – полягає в тому, щоб допомогти людино-машинній взаємодії.
Отже, для продукування QA DevOps-системи, застосуємо метод системної інженерії уніфікованої моделі DevOps-системи для продуктів IT індустрії, і відповідно до кроків (1-4) методу опишемо ті підрозділи які стосуються планування та інженерії системи.

4.1 Формалізація постановки задачі інженерії QA DevOps-системи

На першому етапі імплементації, формалізація класу/об'єкта класу розробки і підтримки продуктів, товарів та послуг, у відповідності до описаного в попередніх розділах методу, ми звертаємось насамперед до формалізації об'єкта класу розробки і підтримки. Для цього формалізуємо технічне завдання.

Підстави для проведення роботи. Необхідність розробки DevOps системи для продукування, підтримки та управління процесами продукування і підтримки QA-систем «медико-біологічного напряму.

Об'єкт дослідження. DevOps системи для продукування, підтримки та управління процесами продукування і підтримки QA-систем «медико-біологічного напряму.

Предметно орієнтований клас QA систем, процеси її продукування і підтримки за допомогою DevOps-системи.

Предмет дослідження. Системна інженерія DevOps-системи для продукування, підтримки та управління процесами продукування і підтримки QA-систем медичного напряму, QA-системи медичного напряму, застосування методу системної інженерії DevOps-систем для продукування предметно орієнтованих QA-систем, математичне, програмне, інформаційне та інші види забезпечення DevOps-систем та QA систем медичного напряму.

Мета. Аналіз текстової інформації «медико-біологічного напряму» і встановлення предметного-змістовного наповнення текстів відповідної тематики для автоматизації знаходження максимально точних відповідей на запити користувачів за різними темами для підвищення ефективності служби підтримки.

Призначення. Система призначена для покращення якості пошуку та підвищення ефективності служби підтримки систем автоматизації медико-біологічної діяльності.

Актуальність. На сьогоднішній день організаціям доводиться витрачати багато коштів на штат служби підтримки та організацію якісного пошуку, дана робота призначена розробити предметно орієнтовану систему, яка змогла би максимально якісно знаходити відповіді на питання користувачів за «медико-біологічним напрямом».

Вихідні дані. При створенні системи будуть виконані результати попередніх досліджень методу системної інженерії DevOps та предметно орієнтованих QA-систем. Окрім цього будуть використані результати прикладних застосувань алгоритмів в задачах інформаційного пошуку.

Вимоги до роботи. В якості вимог буде використано набір бізнес-правил описаних під час бізнес-профілювання.

Очікувані результати. В результаті виконання роботи буде розроблена предметно орієнтована QA-система «медико-біологічного напряму» з застосуванням методу системної інженерії DevOps-систем.

Будуть отримані наступні результати:

1. Розроблено предметно орієнтовану QA-систему «медико-біологічного напряму» із застосуванням методу системної інженерії DevOps-систем;

2. Визначено найбільш ефективні алгоритми пошуку відповідей на запитання;

3. Імплементовано математичне та програмне забезпечення у вигляді застосунку;

4. Проведено порівняльні аналізи результатів з існуючими QA-системами;

5. Проведено порівняльні дослідження різних алгоритмів пошуку.

Можливі користувачі. Отримана система може задовольняти запити пацієнтів яких цікавить інформація «медико-біологічного напряму», система не є орієнтованою на лікарів.

Матеріали, що надаються після закінчення роботи. Технічне завдання, математичне, програмне, інформаційне та інші види забезпечення на розроблену QA-систему.

На другому етапі, проводимо формалізацію моделі DevOps-системи на основі бізнес-профіля Еріксона–Пенкера. Як зазначають автори роботи [13, 17], потрібно визначити зміст кожного з класів моделі на рисунку 3.1, в нашому випадку, у термінах постановки задачі інженерії QA-системи медико-біологічного напряму.

Проблема. Необхідність створення QA-системи медико-біологічного напряму для підвищення якості роботи пошуку відповідей на питання які часто задають.

Мета. Створення QA-системи задля знаходження максимально точних відповідей на питання користувачів за встановленими темами медико-біологічного напряму.

Процеси в системі. Для опису процесів у системі, керуємось діаграмою процесів на рисунку 3.2:

- *Процеси планування. Визначення технічних вимог та технічних умови; Постановка технічного завдання; Визначення необхідних ресурсів; Виділення ресурсів.*
- *Процеси побудови. Завантаження текстових даних; Первинна обробка текстових даних; Навчання алгоритмів пошуку; Відповідь на питання за допомогою навчених алгоритмів пошуку; Функціонування застосунку.*
- *Процеси інтеграції. Збір та інтеграція версій; Передача інформації про систему іншим сутностям.*
- *Процеси розгортання. Розгортання системи; Передача інформації про результати розгортання іншим сутностям.*
- *Процеси підтримки. Збирання інформації про продукт, різні метрики та показники; Передача інформації про систему іншим сутностям.*
- *Процеси зворотного зв'язку. Експлуатація продукту; Передача інформації іншим сутностям.*
- *Процеси управління. Аналіз наявних ресурсів; Встановлення шляхів доставки; Розподіл ресурсів; Аналіз результатів та показників; визначення технічних вимог та технічних умов.*

Зміна стану. Необроблений текст запитання; Оброблений текст запитання; Текстова відповідь на задане питання; Ініціалізовані алгоритми пошуку; Навчені алгоритми пошуку.

Ресурс. Для роботи з цим пунктом керуємось діаграмою ресурсів на рисунку 3.3:

- *Вихід бізнес-процесу: Відповіді на запитання користувачів; навчені алгоритми пошуку здатні знаходити релевантні відповіді на запитання, на визначеному наборі даних; QA-система.*

- *Ресурс забезпечення виконання бізнес процесу: Оброблені текстові дані, Модель текстової обробки; Застосунок.*

- *Вхід бізнес-процесу: Необроблені текстові дані, Ініціалізовані алгоритми пошуку.*

- *Ресурси масштабування: ресурси веб-хостингів які дозволяють обирати ресурси системи та керувати ними.*

- *Ресурси автоматизації: програми та пакети які дозволяють підвищити ефективність інтеграції, розробки окремих частин системи та її тестування.*

- *Ресурси неперервної доставки: програми та пакети які дозволяють встановити шляхи доставки та налагодити її максимальну ефективність.*

- *Події, що можуть виникати в даній системі: Зміна навчальних даних; Зміна архітектури алгоритмів пошуку; Поява ефективнішої версії алгоритмів пошуку; Введення текстової інформації в систему (речення запитання, порожні речення, речення що не є питаннями) в процесі функціонування веб-застосунку та реакція системи на неї.*

Бізнес-правила. Для роботи з цим пунктом керуємось діаграмою бізнес-правил на рисунку 3.4.

Визначимо необхідні бізнес-правила. Бізнес-правила DevOps:

- *BR1. Довжина введеного користувачем тексту в систему не може перевищувати 200 символів.*
- *BR2. Система має надавати відповідь користувачу не більш як за 5 секунд.*
- *BR3. Система має надавати якісні відповіді користувачу тільки за визначеними в системі темами.*

- *BR4. Точність системи не має бути нижчою за 50% за метрикою Precision, та не нижчою за 80% за метрикою MRR.*

Бізнес-правила управління:

- *BR5. QA система має відповідати на питання медичної тематики;*
- *BR6. QA система має бути націлена не на лікарів, а на звичайних користувачів яких турбує питання на медичну тематику;*
- *BR7. Команди розробки мають чітко документувати функціонал системи та зберігати всі версії QA системи.*

4.2 Формалізація структурного та динамічного представлення QA DevOps-системи

На третьому та четвертому етапі проводять формалізацію структурного та динамічного представлень DevOps-системи для продукування, підтримки та управління процесами продукування і підтримки QA-систем медичного напряму. Враховуючи детальні попередні викладки в нашій роботі, ми в нашій системі, за замовчуванням, будемо використовувати наведені діаграми високого рівня на рисунках 3.1, 3.5, 3.9-3.11. Але, враховуючи особливості QA-системи, ми маємо розкрити її особливості проектування. Для цього, відповідно до діаграми на рисунку 3.11, опишемо множину сутностей компоненту розробки на рисунках 3.9, 4.1 у вигляді набору суб-компонентів на діаграмі діяльності QA-системи на рисунку 4.2.

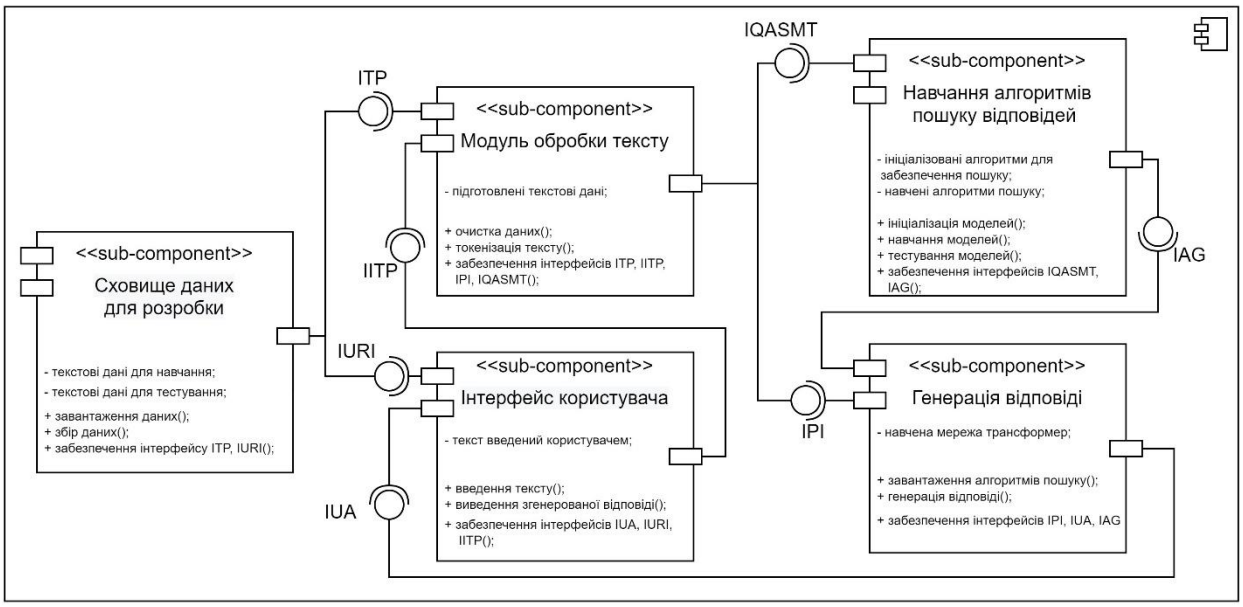


Рисунок 4.1 – Уніфікована модель QA-системи. Деталізована діаграма компонентів у нотації UML

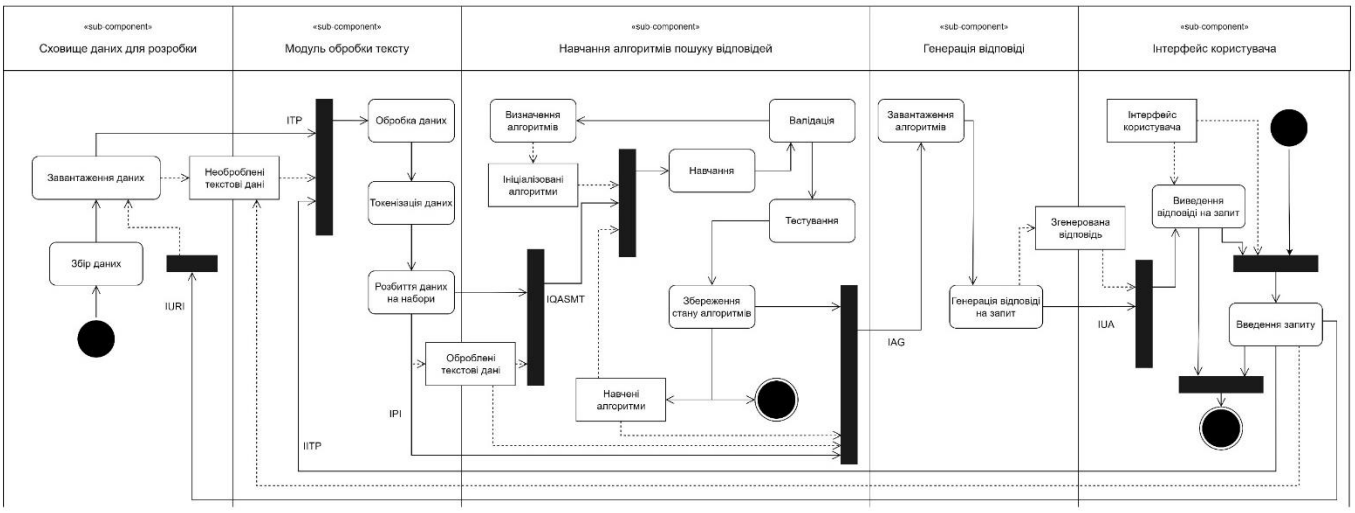


Рисунок 4.2 – Уніфікована модель QA-системи. Деталізована діаграма діяльності QA-системи в нотації UML

Перелік усіх інтерфейсів QA-системи, можна формалізувати як:

- *ITP (Interface Text Processing) – інтерфейс передавання завантажених даних на вхід процесу текстової обробки;*

- *IQASMT (Interface Question Answering System Model Training) – інтерфейс передавання текстових даних на вхід навчання алгоритмам пошуку відповідей;*

- *IPI (Information Processing Interface) – інтерфейс передавання оброблених текстових даних, введених користувачем, на вхід модуля генерації відповіді;*

- *IAG (Interface Answer Generation) – інтерфейс завантаження найефективніших алгоритмів пошуку для генерації відповіді на основі обробленого тексту введеного користувачем;*

- *IUA (Interface User-Application) – інтерфейс передавання отриманих відповідей до модуля інтерфейсу користувача для їх виведення користувачу;*

- *IITP (Interface Input Text Processing) – інтерфейс передавання тексту, введеного користувачем, до модуля первинного текстового оброблення;*

- *IURI (Interface User Request Initialization) – інтерфейс ініціалізації запитів для обробки текстових запитів з модуля інтерфейсу користувача.*

Як показано на рисунку 4.1, множину сутностей розробки QA-системи ми формалізуємо за допомогою шести суб-компонентів. Аби зрозуміти детальне призначення кожного суб-компоненту наведемо їх опис.

Сховище даних для розробки – сховище в якому знаходяться текстові дані для навчання алгоритмів пошуку.

Модуль обробки тексту – призначений для обробки текстової інформації і приведення її до вигляду в якому алгоритми пошуку будуть здатні з ними працювати. Обробка може включати в себе наступні етапи стандартизації текстової інформації:

– Первинне очищення тексту від спеціальних символів, пропусків, приведення до нижнього регістру, нормалізація документа;

– *Токенізація тексту – розбиття тексту на певні атомарні текстові фрагменти – токени, які можуть бути як окремими словами так і певними частинами тексту.*

– *Розбиття наявних наборів даних на частини для навчання, валідації та тестування.*

Навчання алгоритмів пошуку – модуль в якому проводиться навчання, налаштування, адаптація та тестування алгоритмів пошуку. Його основним результатом є навчені алгоритми пошуку. Визначення параметрів алгоритмів та усіх гіперпараметрів визначається безпосередньо під час навчання з використанням набору даних для валідації.

В даному компоненті системи передбачається такий функціонал:

– *Створення/ініціалізація алгоритмів пошуку вищеописаної архітектури;*

– *Навчання алгоритмів з адаптацією гіперпараметрів моделі;*

– *Кількісне та якісне тестування якості наданих відповідей на запитання за медичним напрямом.*

Генерація відповіді – компонент який на основі отриманих даних з модуля обробки тексту (за використання інтерфейсу IPI) та навчених алгоритмів пошуку (завантаженої через інтерфейс IAG) генерує відповідь на поставлені питання користувачів (в режимі який не передбачає оновлення параметрів алгоритмів). Завантаження останньої найефективнішої версії алгоритмів пошуку входить до складу цього компоненту, як фактична імплементація інтерфейсу IAG. Даний інтерфейс передає завантажені алгоритми на вхід процесу системи за для проведення операції генерації відповіді оброблених текстових даних, введених користувачем.

Інтерфейс користувача – модуль який надає елементарний графічний інтерфейс користувача для роботи з QA-системи, для цих потреб може бути використаний будь-яка мова програмування та фреймворки, які будуть мати можливість завантаження натренованих моделей.

В даному додатку має бути доступно дві основні функції:

– *Введення запитання, за «медико-біологічним напрямом» для отримання відповіді: введена текстова інформація передається інтерфейсом IITP на вхід функції очищення тексту модулю обробки текстових даних;*
– *Переглянути результат (отриманий через інтерфейс IUA) – функція виведення отриманої відповіді модулю інтерфейсу користувача.*

4.3 Математичне забезпечення QA DevOps-системи

Перед тим як перейти до використовуваних да нашій роботі математичних інструментів, ми зробимо огляд інших популярних на сьогодні рішень які могли би бути альтернативами до тих моделей які будуть використані в подальшому. Всі розглянуті підходи в підрозділі огляду існуючих рішень є не менш важливими від застосовуваних, так як вони слугують фундаментом для них і є важливою частиною розвитку даних технологій.

4.3.1 Існуючі методи та підходи вирішення задач пов'язаних з пошуком текстових документів

До появи можливості активного застосування нейронних мереж, необхідні були методи які ґрунтуються на нескладних обчисленнях та швидко працюють. Для пошуку документів зазвичай використовувались, та досі активно використовуються, моделі, що базуються на статистиках тексту, тому які слова там часто зустрічаються та інші. В якості огляду існуючих рішень ми поглянемо на два методи які крізь роки зарекомендували себе найкращим чином при оцінці відповідності документів до запиту користувача, а саме про TF-IDF та Okapi BM25 [20]. А також різні моделі токенізаторів, векторного представлення слів та трансформерів, і вже потім перейдемо до інструментів які ми будемо використовувати в даній роботі.

TF-IDF (Term Frequency – Inverse Document Frequency) – це модель [20] порівняння що базується на оцінці частоти появи токенів в запиті та документах. TF – це [20] підрахунок токенів в документах (4.1).

$$\mathrm{TF} = \mathrm{count}(w, d) \tag{4.1}$$

IDF – це [20] підрахунок документів d *що містить відповідний токен* w *(4.2).*

$$\mathrm{IDF} = \log\left(\frac{M+1}{df(w)}\right) \tag{4.2}$$

M *– це загальна кількість документів по яким здійснюється пошук,* $df(w)$ *– частота документів які містять відповідний токен* w*.*

Відповідно до цієї концепції [20] записують функцію ранжування TF-IDF (4.3).

$$f(q, d) = \sum_{w \in q \cap d} c(w, q)\, c(w, d) \log\left(\frac{M+1}{df(w)}\right) \tag{4.3}$$

Okapi BM25 є модифікацією TF-IDF і зазвичай саме ця модель використовується в сучасних системах. Основна ідея цього підходу [20], полягає в штрафі TF при обчисленнях, це робиться для того аби документи з дуже частою появою певних токенів не могли значно легко отримувати перевагу і записується у вигляді (4.4).

$$f(q, d) = \sum_{w \in q \cap d} c(w, q) \frac{(k+1)c(w, d)}{c(w, d) + k\left(1 - b + b\frac{|d|}{avdl}\right)} \log\left(\frac{M+1}{df(w)}\right) \tag{4.4}$$

Де $b \in [0,1], k \in [0, +\infty)$ *і можуть налаштовуватись,* $avdl$ *– середня довжина документів в усьому наборі даних.*

Зазвичай перед ранжуванням документів, передує пошук [21] документів які за частотою слів є найбільш підходящими, такий підхід називають зворотним індексом.

Тепер розглянемо існуючі моделі представлення слів у вигляді токенів. Одним з популярних алгоритмів токенізації є BPE (Byte Pair Encoding) [21]. При застосуванні даного алгоритму, всі слова в тексті розбиваються на літери, з них формується словник, потім з них формують послідовні пари і знаходять ту пару яка частіше всього зустрічається додають в словник і таким чином, повторюють цю операцію доти, поки не буде досягнуто бажаної кількості слів в словнику.

Іншим підходом є просто розбиття слів на окремі букви або символи, це називають – «Character Level» [22], даний підхід може здатись збитковим, але як показують останні дослідження цей підхід також має право на життя, тим паче коли з'являються моделі що можуть оброблювати наддовгі послідовності на вході.

Перейдемо до векторного представлення токенів розглянемо такі моделі як Continuous Bag-Of-Words (CBOW), skip-gram, sisg. Для отримання векторних представлення документів, беруть середні значення для векторних представлень слів з документу.

Перші дві моделі це прості мережі прямого розповсюдження і використовують дуже схожі підходи та архітектуру на рисунку 4.3. Де модель CBOW [23], використовує інформацію про декілька попередніх да декілька наступних токенів для отримання векторного представлення токену, а skip-gram [23] прогнозує те якими мають бути сусідні токени, загалом ці підходи використовують коли токенами виступають саме слова, а не їх частини, тому даний підхід при порівнянні з іншими моделями вважається застарілим та неефективним [24].

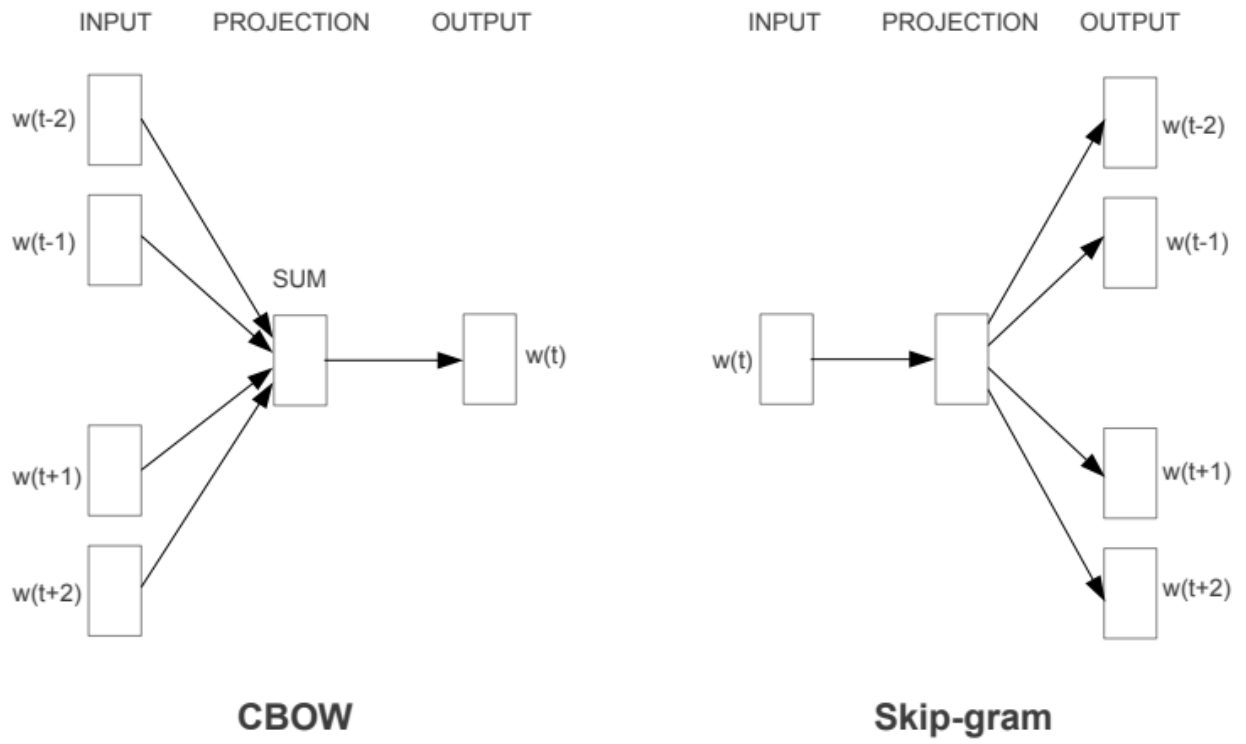


Рисунок 4.3 – схематичне представлення моделей CBOW та skip-gram [25]

Якщо говорити про sisg [24], то дана модель представляє слова у вигляді послідовних n-gram, і таким чином розбиває кожне слово на деяку кількість токенів, а векторне представлення рахується як сума векторних представлень для кожного з цих токенів. Дана модель на сьогодні вважається однією з найкращих для використання, але її проблема полягає в тому, що вона не підлягає адаптації до нових даних (принаймні поточні імплементації), що є основною мотивацією для використання трансформерів в нашій роботі.

4.3.2 Токенізація текстових даних

Токенізацією в даній роботі будемо називати певне розбиття тексту на окремі фрагменти – токени, які можуть бути як окремими словами, частинами слів, або частинами тексту.

Враховуючи природу трансформерів, ми маємо розуміти, що вони найкраще себе проявляють в умовах коли даних дуже велика кількість, це мільйони документів. Саме з цієї причини часто в сфері роботи з природньою мовою використовується трансфер знань. В контексті трансформерів, такий підхід передбачає, використання не тільки внутрішніх ваг мережі, але й векторних представлень токенів які в трансформерах також окремо навчаються. І одним з найпопулярніших підходів до токенізації в даних мережах, є алгоритм WordPiece [26]. Даний алгоритм може розбивати слова на частини, з яких і утворюється кінцевий словник, слова розбиваються таким чином, аби послідовні пари розбиття частини слова мали найбільшу частоту серед усіх інших можливих пар, тобто найбільшу ймовірність бути разом в слові. Автори статті [26] описують даний алгоритм як:

1. *Для початку визначте початковий словник у вигляді всіх можливих літер які зустрічаються в корпусі тексту, а також додайте спеціальні символи.*
2. *Побудуйте мовну модель на даних базуючись на словнику з кроку 1.*
3. *Створіть новий токен, об'єднавши два токени з вже існуючого словника. Додайте цей токен до існуючого словника. Новий токен обирається з усіх можливих комбінацій токенів таким чином, аби збільшувати ймовірність його наявності в вже існуючих даних.*
4. *Виконати перехід до кроку 2 в тому випадку, якщо не досягнуто бажаної кількості токенів, або поки приріст ймовірності не досягне певного, заданого ліміту.*

Враховуючи попередній загальний опис алгоритму, можна записати дещо детальнішу його реалізацію, наприклад [26]:

1. *Розбити весь корпус текстів на слова, та визначити частоту появи для конкретного слова в тексті.*
2. *Визначити перелік унікальних токенів – словник. В даному алгоритмі – це літери, знаки пунктуації та позначення частини речення, важливо*

зазначити, що всі літери в слові, окрім першої, отримують позначу «##» попереду, для того аби знати що перед ними йде якийсь токен, наприклад: «##a», «b», «##ce».

3. *Розбити всі унікальні слова в тексті на токени з існуючого словника.*
4. *Для кожної послідовної пари токенів обчислити оцінку яка має визначати наскільки пара токенів часто зустрічається разом в тексті, для цього, наприклад, можна скористатись формулою (4.5).*
5. *Визначити пару з найбільшою оцінкою, виконати злиття цих токенів та отриманий новий токен додати до словника. Для прикладу пара «a» та «##b» перейде в токен «ab».*
6. *Якщо не досягнуто бажаного розміру словника, то перейти до кроку 3.*

Формулу для оцінки того як часто пара токенів зустрічається разом в тексті можна записати як [26] (4.5).

$$Score = \frac{frequency\ of\ pair}{frequency\ of\ first\ token \times frequency\ of\ second\ token} \qquad (4.5)$$

4.3.3 Архітектура мережі трансформера

З появою трансформерів, світ глибокого навчання змінився. Дана архітектура зарекомендувала себе добре не тільки у відповідності до аналогів, а й як зовсім нова фундаментальна модель, яка завдяки інноваційним застосуванням попередніх напрацювань, дозволяє даний тип мереж на багатьох завданнях природньої мови таким чином, що вони перевершують показники людей при виконанні аналогічних завдань.

Автори статті [27] в якій було вперше презентовано дану модель, наводять структурне представлення мережі на рисунку 4.4.

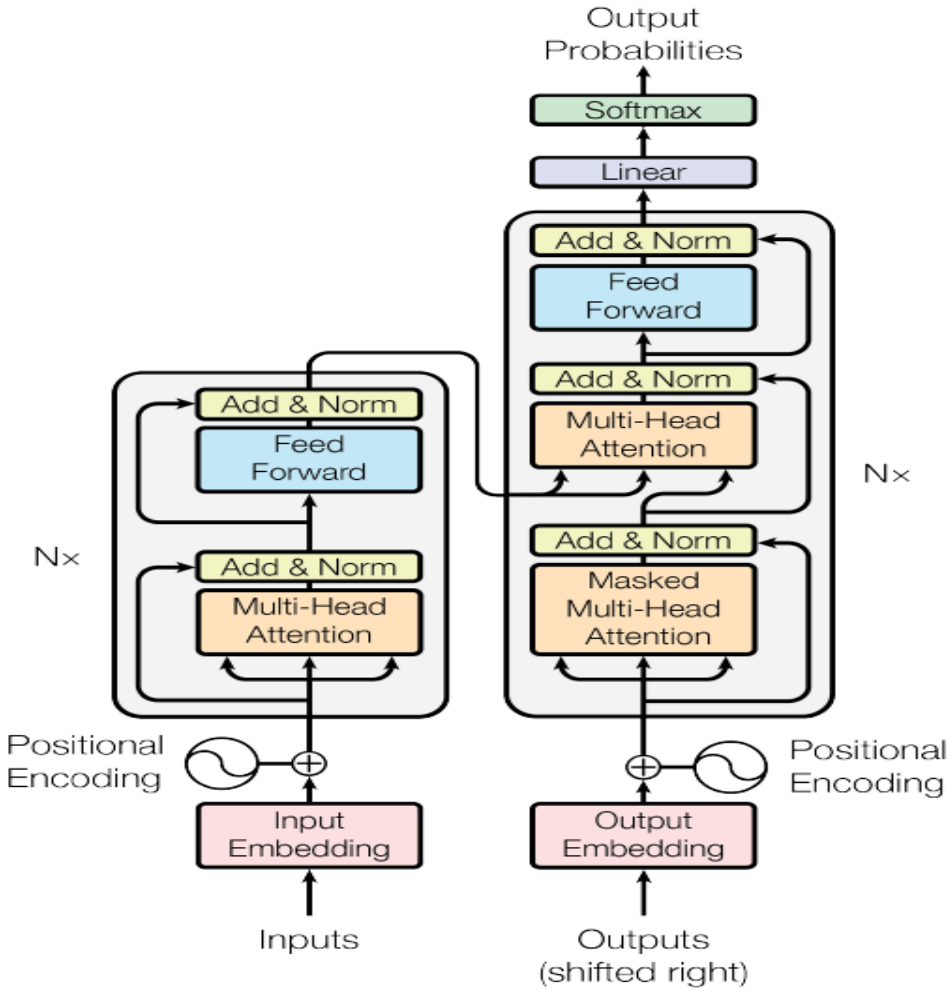


Рисунок 4.4 – структурне представлення мережі трансформера [27]

Як видно на рисунку 4.4, трансформер складається з двох великих частин, вони називаються кодувальником (encoder) та декодером (decoder), справа та зліва відповідно. В даній архітектурі використовується багато концепцій які раніше не використовували разом, серед цих нововведень, такі як: «Attention Mechanism», «Positional Encoding», «Residual Connections», «Layer Normalization», «Multi-Headed Attention», «Masking». Тепер, по черзі, розкажемо про суть кожного з цих нововведень.

Attention Mechanism. Як вказують автори в роботі [27] функцію уваги можна описати як відображення запиту (query) та набору пар ключ-значення (key-value pairs) до виводу, де запит (query), ключі (key), значення (value) та вихід - це матриці ваг. Результат обчислюється як зважена сума значень, де вага, призначений кожному значенню, обчислюється функцією

сумісності запит з відповідним ключем. Такий механізм також називають «(Q, K, V) Attention» або «Scaled Dot-Product Attention». Саме ваги Q, K з кодувальнику передаються в декодер і тим самим використовують ідею з архітектури sequence to sequence [28]. Алгоритм за яким обчислюють «(Q, K, V) Attention» представлено [27] на рисунку 4.5 і формулою (4.6).

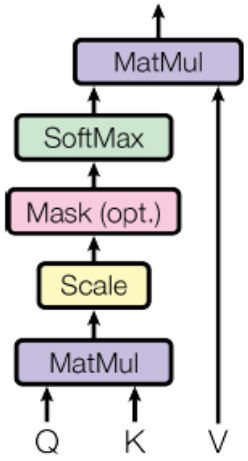


Рисунок 4.5 – алгоритм «(Q, K, V) Attention» [27]

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \qquad (4.6)$$

Де $\frac{1}{\sqrt{d_k}}$, *– [27] це нормуючий множник, а* d_k, *– це розмірність матриці* K, *даний множник застосовують для того аби уникнути проблеми затухаючих градієнтів. Матриці* Q, K, V *– відносяться до параметрів мережі оптимальні значення яких знаходяться за допомогою алгоритму зворотного розповсюдження помилки. Механізм уваги дозволяє краще зрозуміти структуру речень та те наскільки релевантними є слова один відносно одного, тобто, мережа буде розуміти які слова часто зустрічаються разом та з якими словами в реченні.*

Positional Encoding. Як було вказано в роботі [27] трансформери на вхід приймають векторні представлення слів, для того аби ці вектори несли

ще більше інформації, до них додають вектори позиційного кодування (positional encodings), їх знаходять за наступними формулами (4.7)

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$
$$PE(pos, 2i + 1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (4.7)$$

Де pos *– це порядковий номер слова в реченні,* d_{model} *– довжина вектору представлення слова,* $i = 0, \dots d_{model}$.

Residual Connections. Це зв'язки в нейронних мережах, які дозволяють додавати інформацію на наступний шар не тільки шляхом послідовної передачі з попереднього, але й з шарів що були перед попереднім, такий підхід вперше був описаний в статті [29], і як стверджують автори, такий підхід дозволяє не втрачати інформацію на більш глибоких шарах мережі, тобто такий підхід забезпечує можливість для того аби робити скільки завгодно глибокими, і при цьому не тільки не втрачаючи в якості навчання, але й значно покращуючи її. На схемі трансформеру це показано у вигляді стрілок що обходять шари, як, наприклад, на рисунку 4.6.

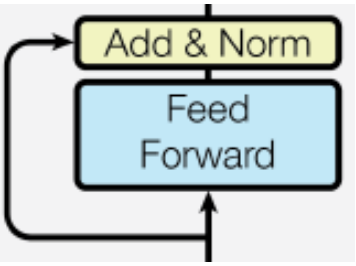


Рисунок 4.6 – зображення residual connections на схемі трансформеру [27]

Layer Normalization. Як вказують автори в роботі [30], тренування трансформерів хоча і менш затратне по ресурсам в порівнянні з sequence-to-sequence [28], але для того аби тренувати дуже великі моделі (мільярди параметрів), потрібно пришвидшувати процес тренування, для цього в

трансформерах є шари нормалізації, їх суть полягає в тому аби нормалізувати активність нейронів і базується цей підхід на статистичному підході, який можна описати [30] за допомогою формул (4.8)

$$\mu^l = \frac{1}{H}\sum_{i=1}^{H} a_i^l$$
$$\sigma^l = \sqrt{\frac{1}{H}\sum_{i=1}^{H}\left(a_i^l - \mu^l\right)^2} \qquad (4.8)$$
$$\hat{a}^l = g\frac{a^l - \mu^l}{\sigma^l}$$

Де H *– кількість параметрів шару,* l *– номер шару,* a^l*– параметр шару,* g *– параметр накопичування (знаходиться за допомогою алгоритму зворотного поширення помилки).*

Multi-Head Attention. Як зазначають в роботі [27], для того аби покращити механізм роботи Attention, було розроблено алгоритм Multi-Head Attention, схема обчислень зображена на рисунку 4.7.

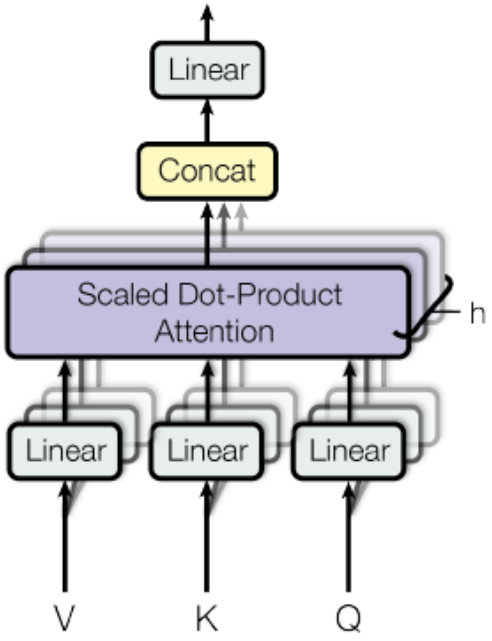


Рисунок 4.7 – графічне представлення Multi-Head Attention [27]

Обчислення на рисунку 4.7, можна надати у вигляді [27] наступних формул (4.9).

$$\begin{gathered} \mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head}_1,\dots,\mathrm{head}_h)W^O \\ \mathrm{head}_i = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{gathered} \qquad (4.9)$$

Де $i = 1,\dots h$*, значення* h *зазвичай обирають з множини* $\{8, 16, 32\}$*. Такий підхід до обчислення Attention дає певні переваги, а саме [27] це дає ще краще можливість вибудовувати зв'язки між словами та їх позиціями в реченні.*

Masking. Даний концепт часто використовується в трансформерах, особливо при генеруванні послідовностей. У вигляді маски виступає матриця спеціального виду, яка допомагає трансформеру навчатися так, аби при генерації наступного слова в послідовності брати до уваги які слова вже з'явились в реченні, а не на ті які знаходяться в реченні далі. Матриця маски має на діагоналі під нею нулі, а всі значення над нею дорівнюють нескінченності. Використання маски [27], можна записати за допомогою формули (4.10).

$$\mathrm{MaskedAttention}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^T + \mathrm{Mask}}{\sqrt{d_k}}\right)V \qquad (4.10)$$

Навчання векторних представлень токенів не показано на рисунку 4.4, але перед тим як представлення токенів перейдуть до шару позиційного кодування, зазвичай дають звичайний повно зв'язний шар нейронної мережі, який представляє собою матрицю векторних представлень токенів, і в процесі навчання мережі, представлення слів також проходять навчання.

Цей підхід, при навчанні моделі на нових даних, навчати разом з нею і нові представлення. Перед тим як подати токени на даний шар, кожному з них присвоюють векторне one-hot представлення, таким чином, що при

подачі їх до мережі і виконавши матричне перемноження з вагами першого шару ми і отримаємо векторне представлення того чи іншого токену.

4.3.4 Варіації мережі трансформера

Перед тим як знайомитись з тією моделлю яка використовується в даній роботі, вважаємо необхідним навести тут короткий опис архітектури яка поклала новий вектор в розробці нових моделей трансформерів. BERT [31] (Bidirectional Encoder Representations from Transformer) – це одна з варіацій оригінального трансформера [27] але яка базується на використанні лише модуля кодувальника, зараз дану модель називають однією з моделей засновників. Модель засновник (foundational model) [32] – це будь-яка модель, яка тренується на великій кількості різноманітних даних, та може бути адаптована до широкого кола різних завдань. В статті [32] наводять візуалізацію на рисунку 4.8, що пояснює моделі засновники.

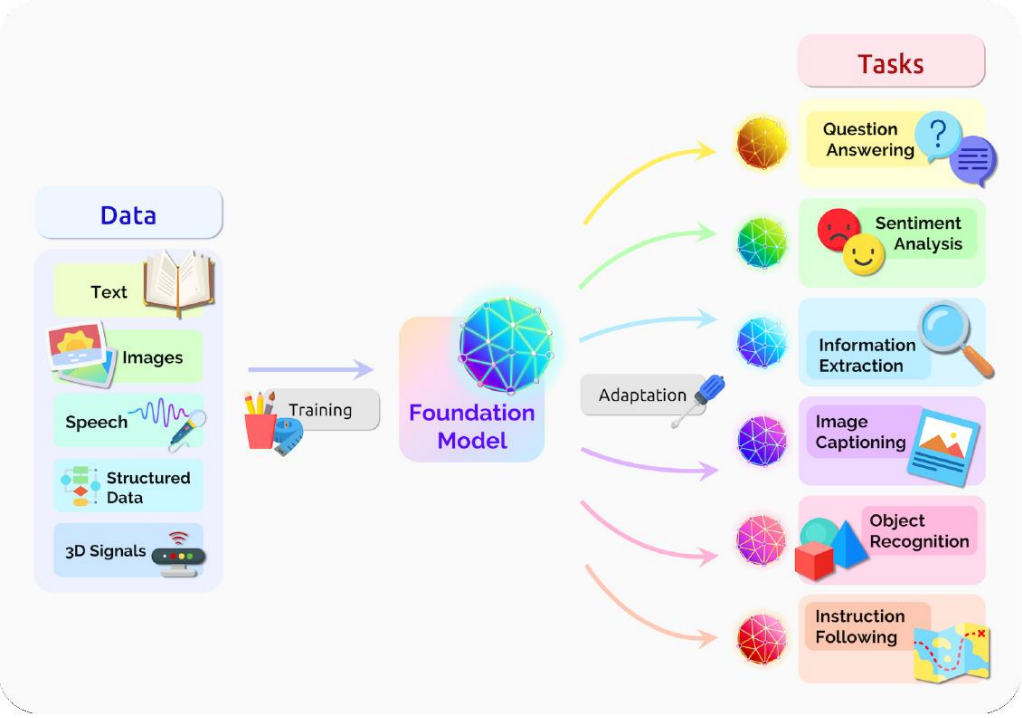


Рисунок 4.8 – принцип роботи та адаптації моделей засновників [32]

Нам важливо зауважити що BERT [31] є саме такою моделлю, оскільки зазвичай сценарії використання цієї архітектури саме такі, це адаптація наперед натренованих ваг до нових задач з заміною шару виходу. І як було вже зазначено в цій роботі, оскільки трансформери можна робити майже як завгодно великими (мільярди параметрів), то модель може запам'ятовувати дуже багато інформації. На останок наведемо зображення архітектури BERT на рисунку 4.9.

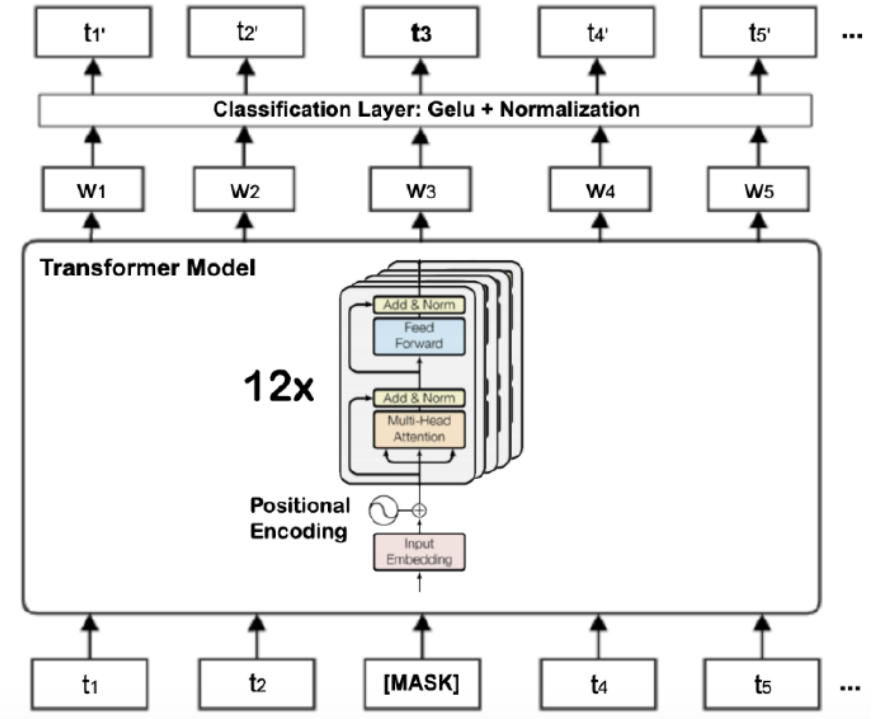


Рисунок 4.9 – приклад можливої архітектури BERT [33]

Тепер ознайомимося з моделлю MiniLM [34] яка використовується в цій роботі, її основна особливість полягає в тому, що вона імітує ваги з BERT, це робиться для того, аби при менших розмірах моделі демонструвати аналогічну ефективність в розпізнаванні. Головна відмінність даної архітектури від подібних інших, полягає в тому, що вона імітує ваги лише з останнього шару мережі, а якщо бути точним, то з останнього кодувальника імітує механізм уваги. Даний підхід дозволяє

додати гнучкості в тому, що кількість шарів меншої моделі може бути налаштована, а не така сама як в оригінальній моделі. Дана концепція зображена на наступній діаграмі з рисунку 4.10.

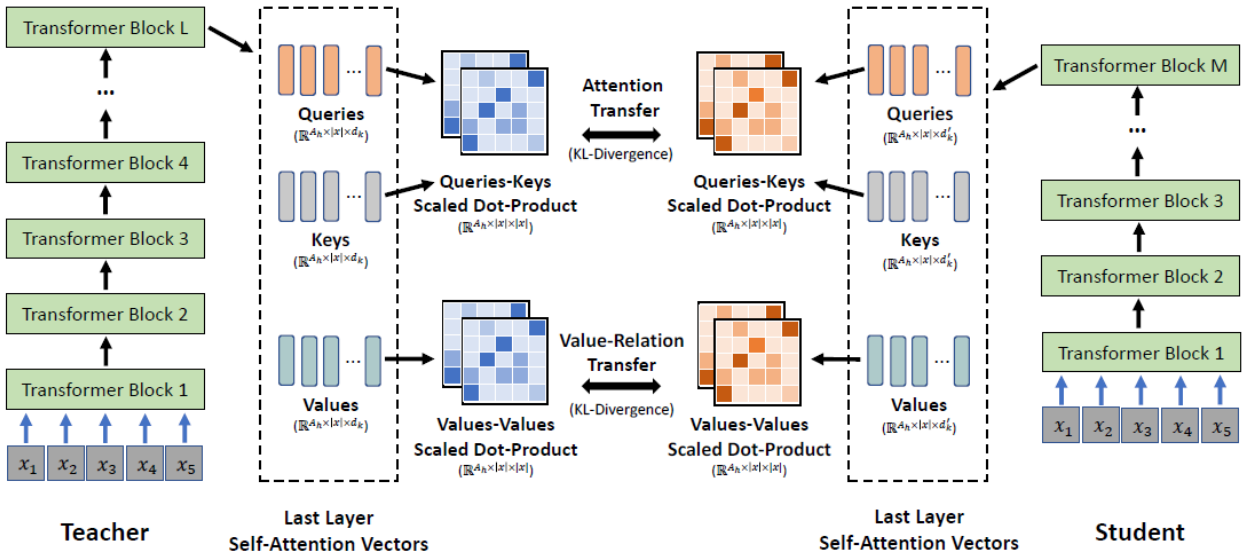


Рисунок 4.10 – процес імітації ваг мережі вчителя в архітектурі MiniLM [34]

Тепер поглибимось в деякі деталі даної моделі. Як видно з рисунку 4.10, процес навчання моделі учня пов'язаний не тільки з навчанням на даних, але й на тому аби імітувати ваги моделі вчителя. Для того аби зрозуміти наскільки якісно це робить модель – використовується дивергенція Кульбака-Лейблера [35], яку можна записати за формулою (4.11).

$$D_{KL} = \sum_i p_i \log \frac{p_i}{q_i} \tag{4.11}$$

Для якісної імітації, в процесі навчання модель імітує не тільки результуючий розподіл ваг з механізму уваги, але й так зване «Value» з «Q, K, V Attention». І в результаті [34], загальна функція втрат навчання обчислюється як сума втрат по механізму уваги $\mathcal{L}_{AT}$ *(Attention Loss), та по «Value»* $\mathcal{L}_{VR}$ *(Value-Relation Loss), і записується у вигляді формули (4.12).*

$$\mathcal{L} = \mathcal{L}_{AT} + \mathcal{L}_{VR} \tag{4.12}$$

Тепер опишемо кожну складову даної функції. Почнемо звісно з втрат по механізму уваги. Проводиться мінімізація різниці між розподілами вчителя та учня [34], і це записується у вигляді формули (4.13).

$$\mathcal{L}_{AT} = \frac{1}{A_h|x|}\sum_{a=1}^{A_h}\sum_{t=1}^{|x|} D_{KL}(A^T_{L,a,t}, A^S_{M,a,t}) \tag{4.13}$$

Де, A_h *– кількість голів уваги,* $|x|$ *– довжина вхідної послідовності,* A^T_L, A^S_M *– це ваги з механізму уваги вчителя та учня відповідно.*

І на останок, формули (4.14-4.16) [34] за якими проводиться імітація «Value» складової механізму уваги.

$$VR^T_{L,a} = \text{softmax}\left(\frac{V^T_{L,a}\left(V^T_{L,a}\right)^T}{\sqrt{d_k}}\right) \tag{4.14}$$

$$VR^S_{M,a} = \text{softmax}\left(\frac{V^S_{M,a}\left(V^S_{M,a}\right)^T}{\sqrt{d'_k}}\right) \tag{4.15}$$

$$\mathcal{L}_{VR} = \frac{1}{A_h|x|}\sum_{a=1}^{A_h}\sum_{t=1}^{|x|} D_{KL}(VR^T_{L,a,t}, VR^S_{M,a,t}) \tag{4.16}$$

Ще однією з потужних варіацій трансформера, яку було вирішено використовувати в даній роботі, є мережа DistillBERT [36]. Дана архітектура, в традиційному виконанні, є на сорок відсотків меншою від BERT [31] але при цьому втрачає лише три відсотки потужності, як зазначають автори статті [36] DistillBert навчається за такою ж концепцією як і MiniLM, тобто вчитель-учень. Але механізм навчання учня менш складний, для дистиляції знань з оригінальної мережі використовується функція втрат у вигляді лінійної комбінації [31, 36, 37] (4.17-4.20).

$$\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{MLM} + \mathcal{L}_{cosine} \quad (4.17)$$

$$\mathcal{L}_{ce} = -\sum_i t_i \log(s_i) \quad (4.18)$$

$$\mathcal{L}_{MLM} = -\sum_i p_i \log(p_i) \quad (4.19)$$

$$\mathcal{L}_{cosine} = \begin{cases} 1 - \text{similarity}(x_1, x_2), y = 1 \\ \max(0, \text{similarity}(x_1, x_2) - \text{margin}, y = 0 \end{cases} \quad (4.20)$$

Де $\mathcal{L}_{ce}$, – це [36] функція крос-ентропії для порівняння значень виходів мережі учня (s_i) та мережі вчителя (t_i), $\mathcal{L}_{MLM}$ – це функція втрат для задачі на яку тренується модель в даному випадку функція крос-ентропії для маскованого моделювання мови, $\mathcal{L}_{cosine}$ – це функція витрат для порівняння векторних представлень моделі вчителя та моделі учня, де в якості метрики використовується косинус подібності.

4.3.5 Отримання векторного представлення документу

Врахувавши все вище описане, перейдемо до векторного представлення документу, яке необхідне для того аби здійснювати семантичний пошук. Як було видно з рисунку 4.9, моделі такі як BERT [31] це буквально сукупність послідовних кодувальників, отже вони на вхід та вихід видають інформацію однакової розмірності. Такий вихід називають контекстним представленням початкових токенів. Отже, для того аби отримати векторне представлення якоїсь послідовності токенів, нам потрібно зменшити розмірність виходу, для цього в даній роботі використовується стратегія усередненого об'єднання (mean pooling) [38], це означає, що по рядкам матриці виходу береться середнє, і результуючий вектор і вважається векторним представленням послідовності токенів, рисунок 4.11.

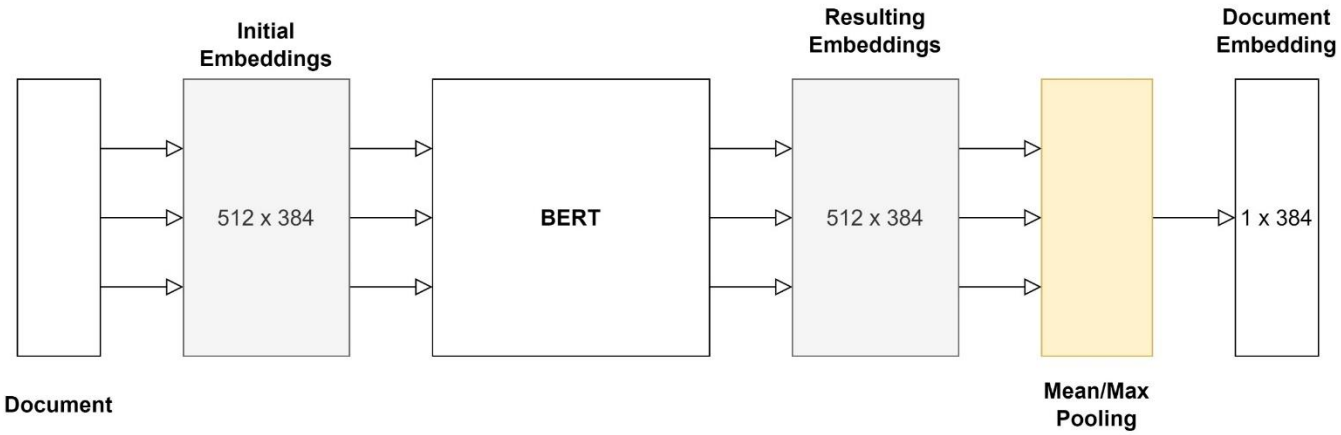


Рисунок 4.11 – отримання вектонного представлення документу за допомогою BERT та Mean/Max pooling

4.3.6 Семантичний пошук

Для того аби виконати пошук по заранння підготовленим векторним представленням документів існує багато способів, в даній роботі, зважаючи на наявні об'єми даних, ми скористаємось алгоритмом пошуку найближчих сусідів. Суть його полягає в тому [39], аби просто рахувати відстані до кожного з наявних екземплярів, і обирати ті до яких відстань за визначеною метрикою найближча. Однією з найпопулярніших метрик є Евклідова (4.21), хоча можуть використовуватись і інші.

$$d(x_1, x_2) = ||x_1 - x_2|| \qquad (4.21)$$

Звісно даний алгоритм має багато недоліків, серед яких, наприклад, неможливість зберігати всі вектори в пам'яті для використання алгоритму, або ж повільні програмні реалізації. Але перевагами є те, що будуть знайдені всі найближчі сусіди, та те що сам алгоритм є доволі простим та зрозумілим.

Вирішення проблем алгоритму можна подолати декількома шляхами, перший з яких це застосувати стиснення векторів до менших розмірів, наприклад, за допомогою алгоритмів Principal Component Analysis (PCA) [39]

або Product Quatization (PQ) [40]. Застосування цих алгоритмів дозволить частково нівелювати проблему з розміщенням великої кількості векторів в пам'яті обчислювальної машини.

Якщо говорити про пришвидшення роботи пошуку схожих векторів, то, в разі необхідності, можна скористатись алгоритмом K-Means [39]. Використовуючи даний алгоритм, ми будемо проводити кластеризацію, і замість того щоб порівнювати новий екземпляр з кожним наявним вектором, ми спочатку будемо шукати найбільш близький центр кластеру, і проводити пошук в кластері, це значно зекономить час, для підвищення якості роботи можна проводити пошук в декількох кластерах, і налаштовувати цей параметр за бажанням.

Всі вищеописані кроки пошуку були описані у роботі [41], де вказані також окремі варіації алгоритму, та рекомендації до налаштування параметрів. Та більш того, автори показують, що даний підхід можна використовувати для пошуку в колекціях що налічують навіть мільярд векторів.

4.3.7 Удосконалення результатів кінцевого ранжування

Для удосконалення результатів пошуку ми скористаємося моделлю Cross-Encoder [37, 38]. Дана модель також заснована на використанні мереж трансформерів. Незважаючи на те, що вже від зазначеного в попередньому розділі алгоритму пошуку і отримаємо найбільш схожі документи, даний підхід не є оптимальним, це описали автори роботи [42] і показали, що використання додаткової мережі трансформера для продукування кінцевого ранжування є найбільш оптимальним і покращує результати пошуку на десятки відсотків за різними метриками. Концепція Cross-Encoder [38] нічим не відрізняється від трансформера з архітектурою подібною до BERT [31], однак, в цьому випадку застосовується концепт нейронних мереж близнюків [38], і на виході застосовується функція

«sigmoid», однак не для того аби отримати бінарний клас, а для того аби отримати значення на проміжку $[0,1]$ *яке буде вказувати наскільки схожими є два документи, чим ближче до одиниці тим більш вірогідним є їх семантична схожість, рисунок 4.12.*

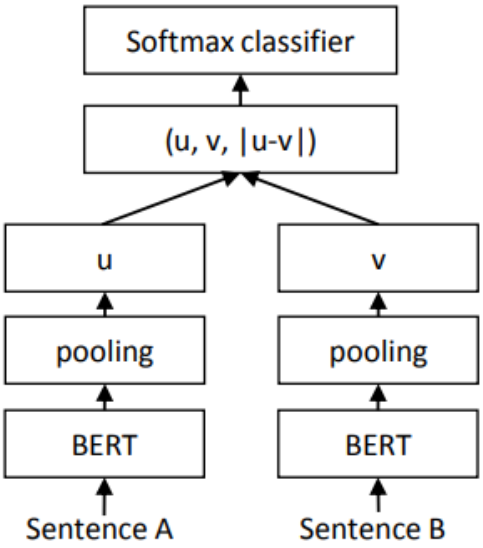


Рисунок 4.12 – одна з можливих архітектур Cross-Encoder [38]

4.3.8 Метрики оцінки якості пошуку

Для оцінки якості пошуку потрібні особливі метрики, які враховують як кількість, так і порядок релевантних документів в отриманому ранжуванні. В даній роботі для оцінки результатів будуть використані метрики «Precision at N» (Precision@N), «Mean Reciprocal Rank» (MRR), «Normalized Discounted Cumulative Gain» (NDCG).

Метрика Precision@N [20] говорить про те, скільки було отримано релевантних документів з результуючого ранжування у вигляді множини з N *документів. А визначається вона так само як а класична метрика Precision, формула (4.22) [20], за матрицею помилок, рисунок 4.13.*

$$\text{Precision@N} = \frac{TP}{TP + FP} \tag{4.22}$$

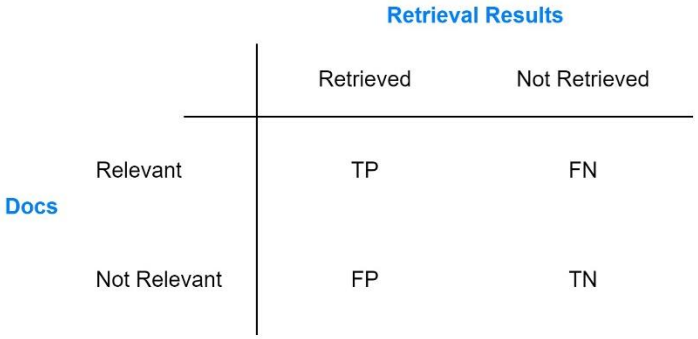


Рисунок 4.13 – матриця помилок [20]

Mean Reciprocal Rank (MRR) – це метрика яка каже [20], наскільки високо в кінцевому ранжуванні знаходиться перший релевантний документ, і записується за формулою (4.23) [20, 43]. Як і Precision може бути розширена для N документів і тоді запишеться як MRR@N.

$$\text{MRR} = \frac{1}{d}\sum_{i=1}^{d}\frac{1}{r_i} \tag{4.23}$$

Де r_i *– ранг документу,* d *– кількість документів.*

Normalized Discounted Cumulative Gain (NDCG) – це метрика [20, 43] яка на відміну від двох попередніх не просто намагається оцінити кінцеве ранжування під кутом зору «релевантний/не релевантний», а врахувати порядок в кінцевому ранжуванні, для цього дані для тренування мають бути спеціально розмічені. Як і попередні може бути розширена для N документів і тоді запишеться як NDCG@N. Дана метрика записується за формулою (4.24)

$$\text{NDCG} = \frac{DCG}{IDCG} \tag{4.24}$$

Як бачимо дана метрика складається з двох складових [20], «Discounted Cumulative Gain» та «Ideal Discounted Cumulative Gain». Розберемо ці частини окремо.

DCG – оцінка [20] релевантності ранжування зі штрафом на порядок в ньому, записується за формулою (4.25).

$$DCG = r_1 + \sum_{i=2}^{n} \frac{r_i}{\log_2 i} \qquad (4.25)$$

IDCG – оцінка аналогічна DCG, однак вона застосовується не для отриманого ранжування, а для ідеального, тобто такого в якому найбільш релевантні документи йдуть першими, а найменш релевантні останніми.

4.3.9 Алгоритм взаємодії математичного забезпечення

В даному підрозділі, було наведено досить багато різномантіних алгоритмів та методів роботи з текстом, але може виявитись незрозумілим, як саме будується взаємодія цих алгоритмів між собою для досягнення максимальної ефективності при використанні. Наведемо кроки алгоритму, які необхідно виконати для отримання кінцевого ранжування документів, під запит введений користувачем:

1. За допомогою мережі трансформеру отримати векторні представлення всіх наявних документів в колекції.

2. При введенні запиту, за допомогою мережі трансформеру отримати векторне представлення запиту.

3. За допомогою алгоритму KNN знайти п'ятдесят найбільш схожих на запит документів.

4. Застосувати Cross-Encoder для всіх пар (документ з результуючого ранжування, запит), та отримати нове, удосконалене, ранжування.

Опишемо приведені кроки за допомогою алгоритму зображеного на блок-схемі на рисунку 4.13.

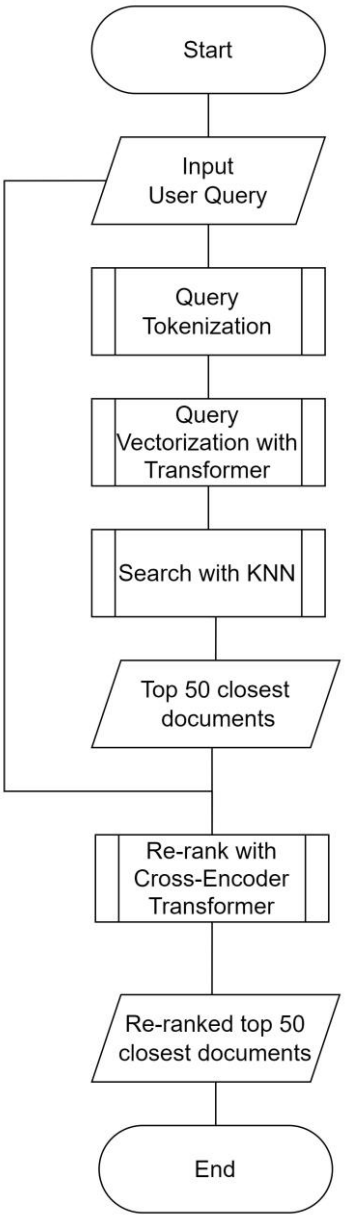


Рисунок 4.13 – алгоритм взаємодії математичного забезпечення

4.4 Висновки до розділу

В розділі 4:

1. Виконано проектування QA DevOps системи, відповідно до методу системної інженерії DevOps-систем для продукування IT продуктів, а саме:

– Формалізація класу/об’єкта класу розробки і підтримки продуктів, товарів та послуг. Розробка технічного завдання на розробку класу/об’єкта класу розробки і підтримки продуктів, товарів та послуг.

– Формалізація моделі DevOps-системи на основі бізнес-профіля Еріксона–Пенкера для визначеного класу/об’єкта класу розробки і підтримки продуктів, товарів та послуг.

– Формалізація структурного та динамічного представлення моделі DevOps-системи для визначеного класу/об’єкта класу розробки і підтримки продуктів, товарів та послуг – діаграма компонентів, діаграма діяльності.

2. Проведено огляд алгоритмів пошуку тексту в сучасних QA системах.

3. Продемонстровані сучасні підходи семантичного пошуку в QA системах на основі мереж трансформерів та алгоритму пошуку найближчих сусідів.

4. Наочно показано алгоритм функціонування математичного забезпечення в QA DevOps-системі.

Приклад 2. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»].

Структурнене представлення чат боту з пошуку роботи має наступний вигляд:

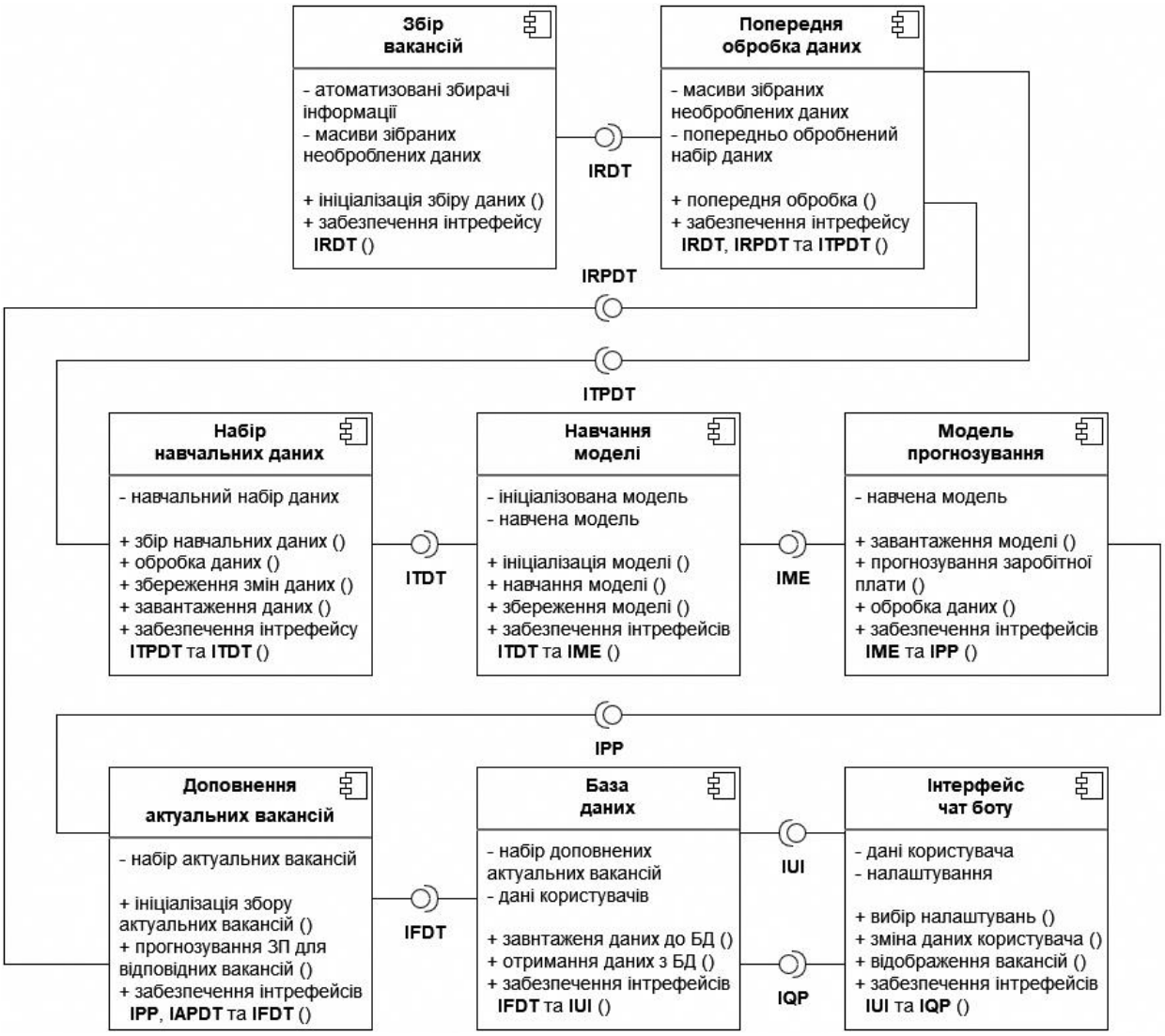


Рисунок 3.2 – Модель чат боту з пошуку роботи. Діаграма компонентів у нотації UML

Функціональність і призначення компонентів системи:

- *Збір вакансій: набір скриптів на основі алгоритмів вебскрапінгу, що відповідає за збір даних про вакансії, що розміщено на веб-сайтах відповідних платформ для пошуку роботи;*

– *Попередня обробка даних: набір інструментів, що проводить попередню обробку зібраних вакансій. В загальному випадку, приводить дані з різних джерел до більш однорідного вигляду;*

– *Набір навчальних даних: локальне сховище даних, на якому розміщено зібраний та попередньо оброблений набір текстових описів вакансій, що буде використано для навчання відповідної моделі прогнозування ЗП. Перед навчання, відповідний набір даних проходить додаткову обробку даних для підвищення ефективності процесу навчання;*

– *Навчання моделі: набір інструментів, що дозволяє створити модель прогнозування необхідної структури та провести її навчання на відповідному навчальному наборі текстових даних;*

– *Модель прогнозування: набір інструментів, що використовують попередньо навчену модель прогнозування, для реалізації процесу визначення ЗП на основі текстового опису вакансій, для їх доповнення;*

– *Доповнення актуальних вакансій: набір інструментів, що дозволяє ініціалізувати збір актуальних вакансій та провести прогнозування ЗП для вакансій, в яких відповідне поле відсутнє;*

– *База даних: сховище, яке містить структуровані, актуальні та доповненні дані про вакансії, що можуть бути застосовано для рекомендації на основі налаштувань користувача;*

– *Інтерфейс чат боту: компонент, який уособлює інтерфейс взаємодії з користувачем. Відповідний інтерфейс відповідає за збір даних необхідних для рекомендації вакансій, збір налаштувань поведінки відповідного чат боту та відображення процесу функціонування системи в цілому.*

Список інтерфейсів:

– *IRDT (Interface of Raw Data Transferring) – інтерфейс передавання необроблених даних (із модуля Збір вакансій) на вхід відповідного модуля Попередньої обробки даних;*

– *IRPDT (Interface of Relevant Preprocessed Data Transferring) – інтерфейс передавання попередньо оброблених даних (із модуля Попередньої обробки даних) на вхід відповідного модуля Доповнення актуальних вакансій;*

– *ITPDT (Interface of Training Preprocessed Data Transferring) – інтерфейс передавання попередньо оброблених даних (із модуля Попередньої обробки даних) на вхід відповідного модуля Набору навчальних даних;*

– *ITDT (Interface of Training Data Transferring) – інтерфейс передавання набору навчальних текстових даних (із модуля Набору навчальних даних) на вхід відповідного модуля Обробки текстових даних;*

– *IME (Interface of Model Extraction) – інтерфейс отримання та передачі результуючої навченої моделі прогнозування ЗП (із модуля Навчання моделі) на вхід відповідного модуля Моделі прогнозування для подальшої реалізації;*

– *IPP (Interface of Prediction Process) – інтерфейс реалізації процесу передбачення ЗП (на основі модуля Моделі Прогнозування) для відповідних тестових даних з описом актуальних вакансій у яких відсутнє поле ЗП, що находять з відповідного модуля Доповнення актуальних вакансій;*

– *IFDT (Interface of Full Data Transferring) – інтерфейс передавання набору навчальних текстових даних (із модуля Набору навчальних даних) на вхід відповідного модуля Обробки текстових даних;*

– *IUI (Interface of User Interactions) – інтерфейс введення необхідних параметрів та передачі налаштувань, що було надано користувачем (із модуля Інтерфейсу чат боту) на вхід відповідного модуля Бази даних вакансій;*

– *IQP (Interface of Query Processing) – інтерфейс ініціалізації процесу пошуку роботи та рекомендації вакансій (із модуля Бази даних вакансій) на вхід відповідного модуля Інтерфейсу чат боту.*

Далі, було спроектовано деталізовану діаграму діяльності другого рівня:

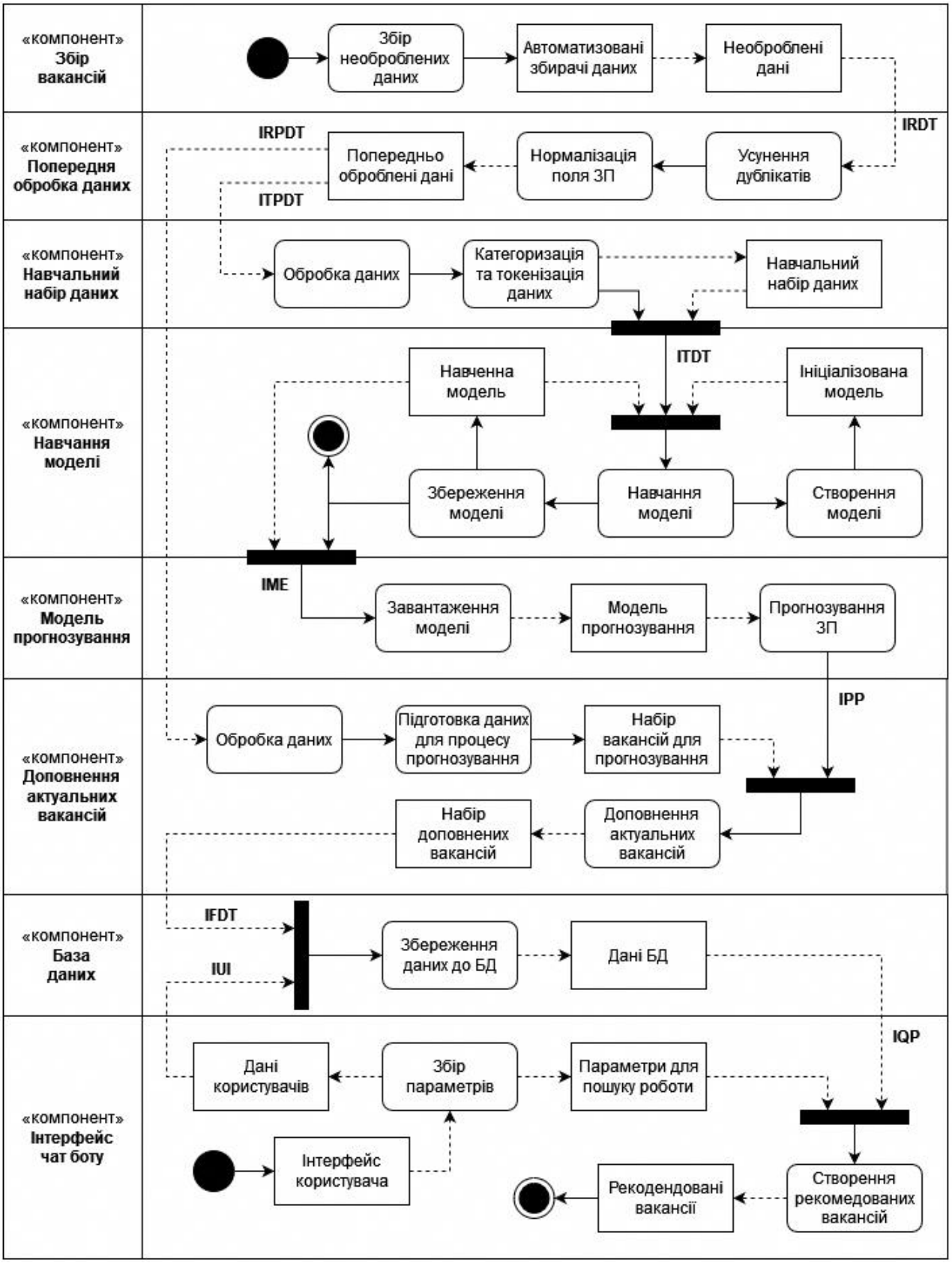


Рисунок 3.3 – Модель чат боту з пошуку роботи. Деталізована діаграма процесів на основі діаграми діяльності з водними доріжками другого рівня в нотації UML

1 ІМПЛЕМЕНТАЦІЯ МОДЕЛІ ЧАТ БОТУ З ПОШУКУ РОБОТИ

1.1 Математичне забезпечення чат боту

1.1.1 Підхід до вирішення задачі рекомендації вакансій

На основі аналізу існуючих рішень, було прийнято рішення, перед початком запуску процесу пошуку вакансій, вимагати від користувача наступний перелік відповідних даних:

- *Регіон пошуку;*
- *Професія;*
- *Очікувана ЗП;*
- *Інтервал отримання рекомендацій.*

Дані регіону можна задавати, обиравши всю територію України, окрему область або місто. Більшість вакансій, мають поле «region», тому обмеження на рекомендації у даному випадку досить тривіальне.

Професія, в свою чергу, більш складна сутність. Вакансії не мають поля, що окремо визначає професію. Але, в загальному випадку, інформація про професію міститься в полі «title». Звичайне порівняння професії та назви вакансії не містить сенсу. Для вирішення цієї проблеми було вирішено застосувати методи NLP.

У даному випадку, було вирішено використати метод NLP, що базується на основі обчислення косинусу подібності (cosine similarity) [15] між професією та назвою вакансією, що було репрезентовано у вигляді токенів. Для виконання даного процесу, було використано зовнішню бібліотеку.

Значення очікуваної ЗП, дозволяє визначити мінімальний поріг рекомендованих вакансій. Для даного процесу фільтрування, справжні та передбаченні ЗП, розцінюються як рівні. Дане фільтрування може бути легко реалізоване, оскільки вакансії, у яких відсутнє поле «salary»,

автоматично доповнюються прогнозованою ЗП, перед завантаження до бази даних.

В свою чергу, інтервал отримання рекомендацій, можна визначати як: щодня, щотижня, щомісяця або відсутність автоматичних рекомендації.

Дане налаштування, впливає лише на періодичність запуску внутрішнього алгоритму рекомендування та розсилання вакансій користувачу.

1.1.2 Механізм рекомендації вакансій

Після збору необхідних даних налаштувань процесу рекомендації для конкретного користувача та збереження їх до бази даних, можна розпочинати процес рекомендації. У даному випадку, механізм рекомендації вакансій, використовуючи базу даних попередньо завантажених актуальних вакансій, намагається сформувати список вакансій, які можуть бути потенційно рекомендовано користувачу.

Для формування даного списку, спочатку, з відвідної бази даних обираються вакансії, що відповідають обмеженням, які надав користувач. Далі, використовуючи метод «cosine similarity» [15], вакансій сортуються за схожістю поля назви та професії, що вказав користувач. Для створення актуального списку вакансій, було вирішено використати не лише дане значення подібності, а й дату створення відповідної вакансії. Тобто, новіші вакансії будуть мати пріоритет у процесі формування рекомендацій.

На основі вимог до функціональності, в процес рекомендації вакансій також було інтегровано обмеження унікальності, тобто кожна вакансія може бути рекомендована одному й тому ж користувачу лише один раз.

Також, оскільки механізм рекомендації вакансій, залежить від бази даних вакансій, відповідна база даних потребує постійних оновлень інформації. Для зменшення навантаження на платформи-джерела вакансій, актуалізацію вакансій було вирішено проводити лише один раз на день.

1.1.3 Підхід до вирішення задачі передбачення на основі текстових даних

Попередньо оброблені дані містять велику кількість полів, що описують відповідні вакансії. До списку даних полів належать: «title», «description», «company», «region», «remote_type», «employment_type», «salary», «additional_info», «seniority».

Оскільки, в результаті порівняння методів передбачення ЗП на основі текстових даних, було обрано модель 1D-CNN, тому можна передавати дані про вакансію у вигляді одного речення, у якому об'єднані всі відповідні поля.

Даний підхід має місце, але структура нейронної мережі надає можливість об'єднати дві моделі, першу, на основі повно зв'язної неорної мережі (ANN), для аналізу категоріальних даних та другу, на основі 1D-CNN, для аналізу виключно текстової частини даних.

Отже, для реалізації відповідної композитної моделі, необхідно визначати які поля належать до категоріальних, а які до виключно тестових. На основі аналізу відповідного навчального датасету, було вирішено створити певні розбиття.

Список категоріальних полів:

- *«region»;*
- *«remote_type»;*
- *«employment_type»;*
- *«seniority».*

Список текстових полів:

- *«title»;*
- *«company»;*
- *«additional_info»;*
- *«description».*

1.1.4 Категоризація та токенізація текстових даних

Для використання відповідних виділених категоріальних та текстових полів, необхідно провести їх перетворення до чисельної репрезентації, яку можна буде передати на вхід відповідної композитної моделі.

Для категоріальних даних, краще використати «one-hot» кодування [16], як найкращий варіант. Дане кодування замінює дані на вектор, у якого лише один не нульовий елемент, що репрезентує значення. На основі аналізу відповідних полів, було вирішено створити наступні кодування.

Кодування поля «region»:

- *«відсутнє»* $\longrightarrow [1,0,0]$;
- *Київ* $\longrightarrow [0,1,0]$;
- *Інші* $\longrightarrow [0,0,1]$.

Кодування поля «remote_type»:

- *«відсутнє»* $\longrightarrow [1,0,0,0]$;
- *remote* $\longrightarrow [0,1,0,0]$;
- *hybrid* $\longrightarrow [0,0,1,0]$;
- *in office* $\longrightarrow [0,0,0,1]$.

Кодування поля «employment_type»:

- *«відсутнє»* $\longrightarrow [1,0,0]$;
- *full time* $\longrightarrow [0,1,0]$;
- *part time* $\longrightarrow [0,0,1]$.

Кодування поля «seniority»:

- *«відсутнє»* $\longrightarrow [1,0,0,0,0,0]$;
- *mid level* $\longrightarrow [0,1,0,0,0,0]$;
- *senior level* $\longrightarrow [0,0,1,0,0,0]$;
- *team lead* $\longrightarrow [0,0,0,1,0,0]$;
- *tech lead* $\longrightarrow [0,0,0,0,1,0]$;

– *Інші* $\longrightarrow [0,0,0,0,0,1]$.

В результаті об'єднання усіх чотирьох «one-hot» репрезентації, було отримано 16-вимірний вектор, що репрезентує категоріальні дані та може буде використано для аналізу відповідної повно зв'язною мережею з відповідної композитної моделі.

Для процесу токенізації даних, спочатку необхідно відповідні текстові поля об'єднати в одну стрічку тексту. Для спрощення завдання виокремлення цих частин моделлю, при об'єднані, кожному поля передуватиме слово-ключ, що характеризуватиме відповідну частину тексту.

Ідейно, відповідне об'єднання матиме наступний вигляд:
"назва «title» компанія «company» інформація «additional_info» опис «description»"

Після створення відповідної стрічки тексту, потрібно перетворити дані спочатку у токени, а потім перекодувати їх у вектори. В загальному випадку, токенізацію та векторизацію текстових даних можна виконати готовими попередньо навченими моделями, такими як Word2Vec, Fasttext, тощо.

У даному випадку, було вирішено залишити завдання токенізації та векторизації текстових даних на модель 1D-CNN. Тобто, перші два шари моделі відповідають за перетворення вхідних даних спочатку у токени, а потім відображають їх у певний простір з фіксованою кількістю вимірів. Ці шари також приймають участь в загальному процесі навчання моделі.

Отже, такий підхід, теоретично, повинен справлятись краще ніж загальні попередньо навчені моделі векторизації тексту. Також, інтегровані шари мають менший розмір, тому теоретично повинні працювати швидше ніж готові рішення.

1.1.5 Опис обраного методу для передбачення

Запропонована комбінація 1D-CNN та моделі ANN мотивована тим, що різні типи даних потребують різних типів обробки. Текстові дані є послідовними і можуть отримати користь від використання згорткових шарів, які можуть захоплювати локальні функції та шаблони. З іншого боку, категоріальні дані не є послідовними і можуть бути найкраще оброблені моделлю типу ANN.

Поєднуючи моделі відповідні моделі та об'єднавши їх результати, з'являється можливість репрезентувати як локальні функції, так і не послідовні характеристики даних [17]. Це призводить до більш всебічного представлення вхідних даних і може покращити продуктивність моделі.

Відповідна композитна модель – це специфічне поєднання ANN та 1D-CNN, з результатами кожної моделі, об'єднані в додаткові шари ANN. Вхідні дані складаються з чистого тексту та категоріальних даних. Відповідні текстові дані обробляються моделлю 1D-CNN, тоді як категоріальні дані обробляються ANN.

Модель 1D-CNN складається з декількох згортокових шарів та шарів агрегування. Згорткові шари витягують певні ознаки з даних вхідного тексту, а шари агрегування зменшують розмірність витягнутих ознак. Вихід моделі 1D-CNN – це набір ознак високого рівня, які фіксують важливі характеристики відповідних текстових даних [18].

Модель ANN використовується для обробки категоріальних даних, оскільки є більш чутливою та простішою в навчанні для даного типу класифікації [19]. Він складається з декількох шарів нейронів, кожен з яких застосовує нелінійну функцію активації до вхідних даних. Вихід моделі ANN – це набір ознак високого рівня, які фіксують важливі характеристики категоріальних даних [19].

Виходи відповідних моделей 1D-CNN та ANN потім об'єднуються в додаткові шари ANN, що дозволяє підвищити ефективність даної моделі

[17]. Об'єднані ознаки передаються через кілька шарів нейронів, кожен з яких також застосовує нелінійну функцію активації.

Під час навчання модель проходить процес оптимізації за допомогою відповідної функції втрат та відповідного алгоритму оптимізації для зменшення результуючої похибки прогнозування. На кожній ітерації процесу навчання, модель проходить валідацію відповідних метрик для оцінки її продуктивності.

Підводячи підсумок, запропонована модель – це поєднання ANN та 1D-CNN, яка обробляє чистий текст та категоричні дані відповідно. Виходи кожної моделі об'єднуються в додаткові шари ANN, які виробляють остаточне прогнозування.

1.1.6 Архітектура моделі прогнозування

Модель прогнозування має композитний вигляд. Тобто структура моделі об’єднує дві інші моделі, а саме текстової моделі 1D-CNN та категоріальної моделі ANN. В результаті підбору та аналізу різних моделей, було отримано:

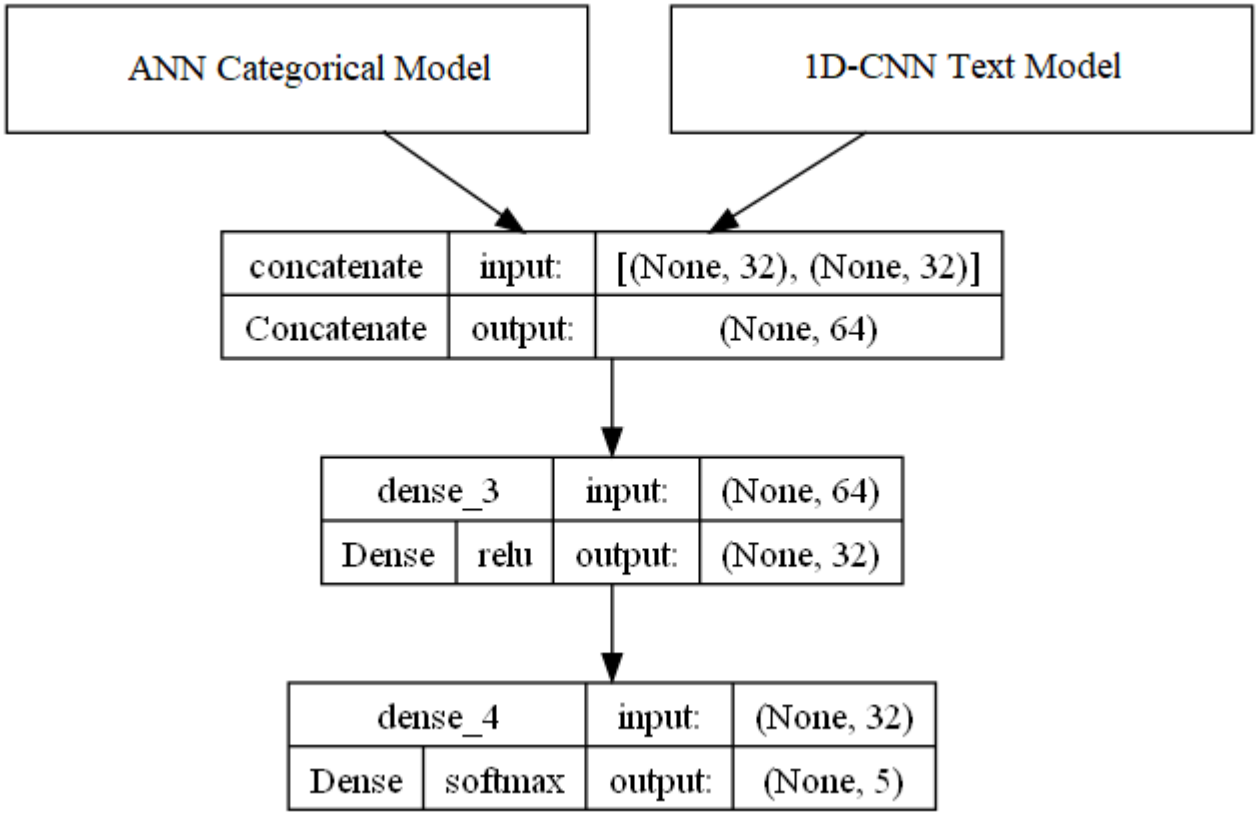


Рисунок 4.1.1 – Структура композитної моделі прогнозування ЗП

Дана структура моделі прогнозування ЗП, містить дві додаткових моделі, що утворюють композицію та структуру яких було не відображено, а саме:

- *«ANN Categorical Model»;*
- *«1D-CNN Text Model».*

Модель «ANN Categorical Model» представляє собою модель звичайної нейронної мережі, що відповідальна за обробку та аналіз категоріальної частини репрезентації навчальних даних. Відповідна структура обраної моделі має вигляд:

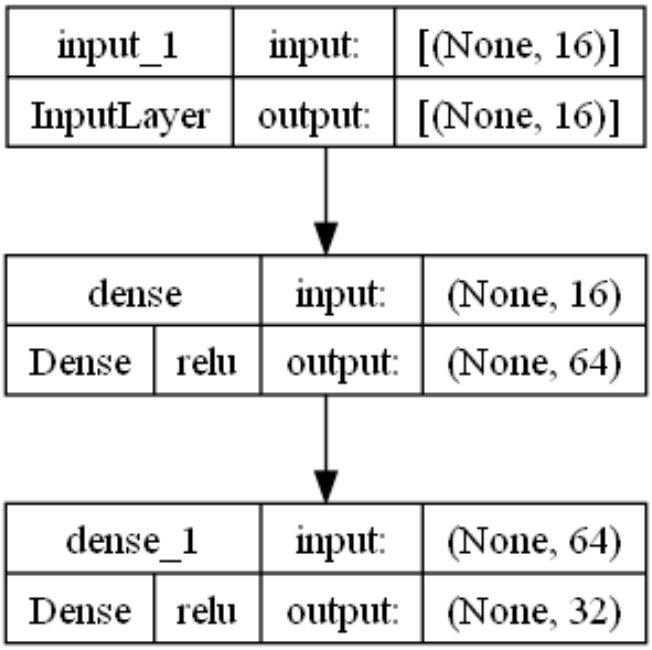


Рисунок 4.1.2 – Структура моделі «ANN Categorical Model»

Модель «1D-CNN Text Model» представляє собою модель одновимірної згорткової нейронної мережі, що відповідальна за обробку та аналіз текстової частини репрезентації навчальних даних. Відповідна структура даної моделі має вигляд:

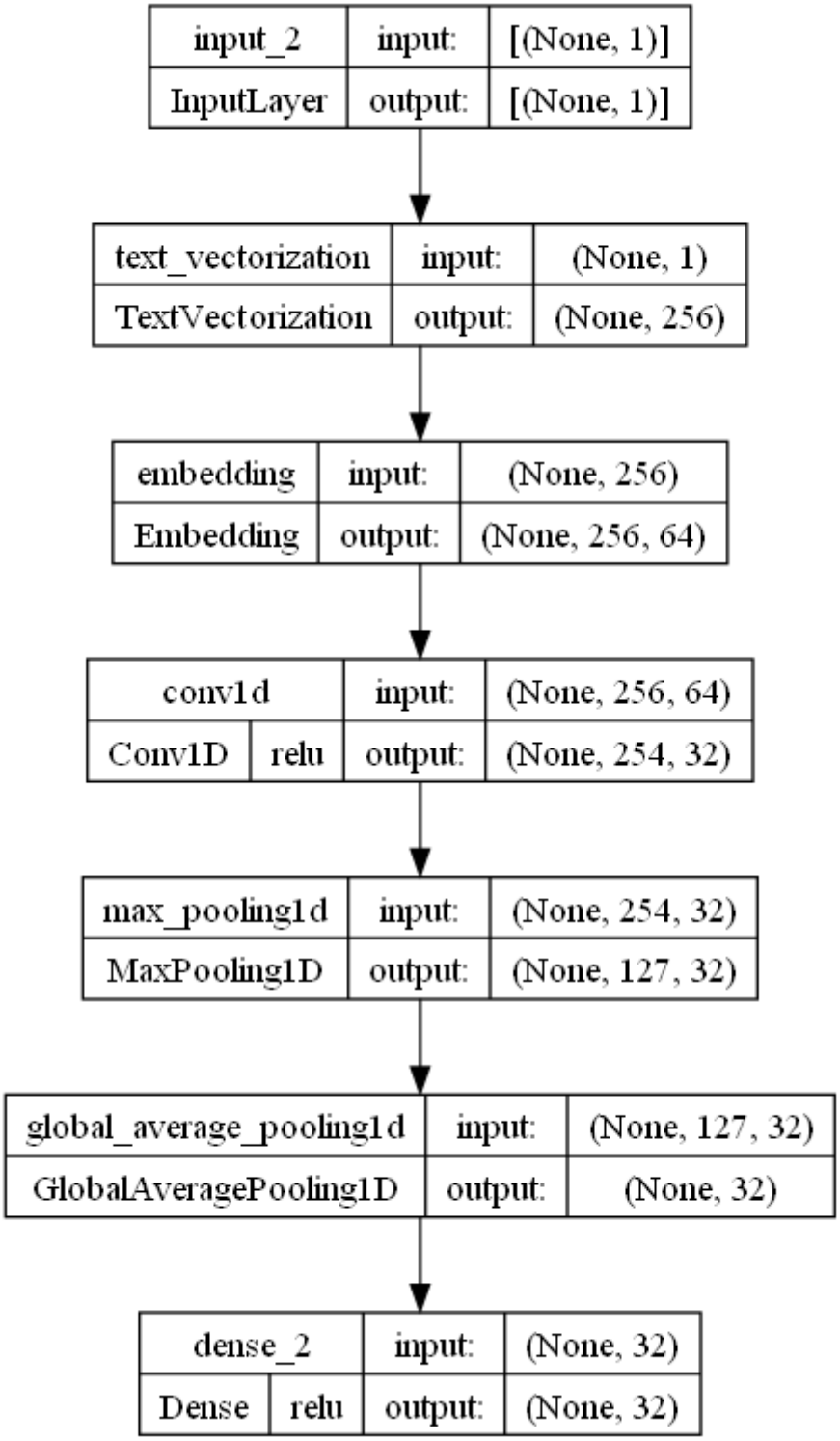


Рисунок 4.1.3 – Структура моделі «1D-CNN Text Model»

Під час проведення навчання відповідної моделі прогнозування (рисунок 4.1.1), було використано певний перелік гіперпараметрів, що визначають характер процесу навчання. Відповідний перелік гіперпараметрів має наступний вигляд:

Таблиця 4.1 – Гіперпараметри використані для моделі прогнозування ЗП

Гіперпараметр	*Значення*
Розмір словнику для текстового кодувальника	10′000
Розмірність вектору для кодування слова	64
Максимальна кількість слів	256
Розмірність вектору категорій	16
Кількість епох навчання	35
Розмір партії даних для навчання	64
Коефіцієнт швидкості навчання	$1 \cdot 10^{-4}$
Оптимізатор	*Adam*
Використання зміщень	*True*
Функція втрат	*MSE*
Функція метрики	*Accuracy*
Розмір навчального набору даних	5′325 (90% від 5`917)
Розмір валідаційного набору даних	592 (10% від 5`917)

6.1.3.8. Програмне забезпечення системи, підсистеми, програмного продукту, методів, моделей та алгоритмів результатів досліджень

Як уже зазначалось, добре задокументоване математичне забезпечення слугує основою для розробки програмного забезпечення (приклад листингу Додаток 1), верифікації і валідації результатів розробки і нарешті формування висновків за результатами виконання БКР.

Нагадаємо, що програмне забезпечення, це система окремих програм і програмних модулів загальносистемного чи прикладного призначення та формалізованої документації, що призначена для безпосереднього виконання заданих функцій і надання задекларованих сервісів для користувачів.

Розробка програмного забезпечення, як правило, завершується верифікацією і валідаціією результатів роботи.

Стандарт ICO 9000:2000 визначає ці терміни таким чином:

«**Верифікація (**verification — перевірка,**)** — підтвердження на основі надання об'єктивних свідчень того, що *встановлені вимоги були виконані*».

«**Валідація** (validation — надання законної сили)— підтвердження на основі надання об'єктивних свідчень того, що встановлені вимоги, *призначені для конкретного використання і застосування, виконані*».

Таким чином можна зробити висновок:

верифікація — проводиться методом порівняння значень конкретних характеристик продукції, товарів і послуг з заданими в документації значеннями цих характеристик для формування висновку про відповідність продукції, товарів і послуг заданим в документації вимогам;

валідація — проводиться при необхідності і виконується методом аналізу заданих умов застосування/експлуатації і оцінки відповідності значень характеристик продукції, товарів і послуг цим умовам, з наступним висновком про можливість застосування продукції, товарів і послуг для встановлених умов застосування/експлуатації.

Також нагадуємо, що нумерація розділу і підрозділів прикладу відноситься лише до цього прикладу і до нумерації розділів змісту навчального посібника ніякого відношення не має. Назви розділів і підрозділів прикладів виділено курсивом.

Приклад 1. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»].

1.2 Програмне забезпечення чат боту

1.2.1 Компонент «Збір вакансій»

Даний компонент відповідає за автоматизований процес збору даних на основі методів вебскрапінгу. Відповідний компонент містить персоналізовані алгортими для збору даних з трьох веб-сайтів для пошуку роботи: Djinni.co, Robota.ua та Work.ua.

Збір проводить за наступними полями: «url», «title», «description», «company», «updated», «region», «country», «remote_type», «employment_type», «salary», «additional_info», «seniority», «date_gathered».

Кожен запуск процесу вебскрапінгу генерує датасети, назви яких складаються з сайту-джерела вакансій та дати коли було поведено збір у форматі «НазваПлатформи_День-Місяць». Відповідні датасети зберігаються до папки «temp_data». Відповідні файли зберігаються у форматі «json».

Додатково, з міркувань забезпечення продуктивності та простоти системи, кожен запуск процесу збору вакансій призводить до видалення усіх файлів з відповідної папки «temp_data».

Отже, на виході відповідного компоненту, генерується відповідний набір з трьох датасетів, що містять актуальні вакансії з відповідних платформ для пошуку роботи. Дані містять однаковий набір полів, але потребують подальшої обробки.

1.2.2 Компонент «Попередня обробка даних»

Даний компонент відповідає за обробку та перетворення текстових даних, до більш однорідного вигляду. Відповідний компонент використовує результати функціонування компоненту «Збір вакансій», що збережено у вигляді json-файлів у відповідній папці «temp_data».

Для початку, даний компонент намагається об'єднати файли з папки «temp_data» у один набір даних. Оскільки, існують дублікати вакансій що розміщено на різних платформах, потрібно залишити лише унікальні входження.

Ключ унікальності створюється з об'єднання полів «title» та «company». Додатково, також було введено обмеження на унікальність поля «url», що боротися з дублікатами, для яких було змінено в поля «title» або «company». Тобто процес відбору унікальних записів має наступний вигляд:

- *Створення трьох масивів для додавання ключів унікальності, унікальних посилань та відповідних унікальних вакансій;*
- *Ініціалізується основний цикл, у якому буде здійснено перегляд усіх вакансій з усіх доступних файлів. Після чого відбувається наступний алгоритм:*

1) Для поточної вакансій генерується ключ унікальності;

2) Перевіряється існування даного ключа в масиві унікальних ключів:

a. Ключ вже існує: вакансія пропускається, перехід до кроку №5;

b. Ключ не існує: перехід до кроку №3.

3) Перевіряється існування даного посилання в масиві унікальних посилань:

a. Посилання вже існує: вакансія пропускається, перехід до кроку №5;

b. Посилання не існує: перехід до кроку №4.

4) Поточна вакансія є унікальною, тому ключ унікальності, посилання та сама вакансія додається у відповідні масиви унікальності;

5) Перевіряється чи досягнуто кінця вакансій що розглядаються:

a. Так: завершення циклу;

b. Ні: перехід до наступної вакансії, повернення до кроку №1.

- *Екстракція та повернення масиву унікальних вакансій.*

Після чого, розпочинається процес обробки тестових даних. На даному етапі, до процесу обробки належать наступні дії:

- *Заміна символів усіх, що є різновидами пробілу на знак пробілу;*
- *Проведення екстракції числових даних з поля «salary», видалення знаків валюти та заміна на пусте значення, у випадку відсутності числових даних;*
- *Видалення вакансій, у яких поле «description» є пустим;*
- *Визначення мови поля «description» та на основі результатів видалення вакансій які описані не українською мовою.*

В свою чергу, в якості цільового поля використовується поле «salary». Враховуючи алгоритм попередньої обробки даних, відповідне поле вже зведено до необхідного формату.

Отже, на виході відповідного компоненту, генерується відповідний попередньо оброблений набір даних, що може бути в подальшому використаний для навчання чи прогнозування за допомогою відповідної моделі.

1.2.3 Компонент «Набір навчальних даних»

Даний компонент реалізує механізм збору та обробки текстових даних, що можуть бути використані для навчання моделі прогнозування ЗП.

Попередньо зібрані дані відбувається містять наступні поля: «url», «title», «description», «company», «updated», «region», «country», «remote_type», «employment_type», «salary», «additional_info», «seniority», «date_gathered».

Для процесу навчання, кількість відповідних полів було зменшено до наступного набору: «title», «description», «company», «region», «remote_type», «employment_type», «salary», «additional_info», «seniority». Після чого, для спрощення процесу навчання, текстові дані розділяються на дві групи: категоріальні та текстові.

До категоріальних даних віднесено наступні поля: «region», «remote_type», «employment_type», «seniority». Через невелику кількість значень, що використовують для опису цих полів, їх може бути замінено на векторну «one-hot» репрезентацію. Після чого, відповідний набір векторів можна об'єднати і отримати категоріальну репрезентацію частини текстових даних. У даному випадку результуючий категоріальний вектор має розмірність 16.

До текстових даних віднесено: «title», «company», «additional_info», «description». Відповідна репрезентація утворена за допомогою об'єднання текстових даних усіх 4 полів, з відповідним ключовим словом попереду (назва, компанія, інформація, опис). Іншими словами, текстова частина репрезентована у вигляді однієї об'єднаної стрічки тексту.

У свою чергу, в якості цільового використовується поле «salary». Враховуючи формат, у якому відображаються значення ЗП для вакансій, використовується нормалізація значення ЗП за допомогою знаходження середнього арифметичного.

Отже, на виході відповідного компоненту, генерується відповідний оброблений набір навчальних даних, різної репрезентації, що може бути в подальшому використаний для навчання відповідної моделі прогнозування ЗП.

1.2.4 Компонент «Навчання моделі»

Даний компонент реалізує процес навчання композитної моделі для прогнозування ЗП визначеної архітектури (див. 4.1.6). Репрезентація даних, для функціонування моделі, складається з двох частин: категоріальної та текстової. Тобто структура моделі прогнозування, має специфічний вигляд, що об'єднує в собі аналіз цих двох представлень вхідних даних.

Відповідну модель, було попередньо навчено, використовуючи скрипт «salary_predictor/train.py». В якості навчальних даних, було використано результати функціонування компонента «Набір навчальних даних». Метрики, «accuracy» та «loss», отримані в процесі навчання репрезентовано у вигляді графіків та збережено у папку «salary_predictor/plots».

Навчену модель прогнозування збережено у папці «salary_predictor/models». В подальшому, дана попередньо навчена модель може бути використана компонентом «Модель прогнозування» для передбачення проміжку ЗП на основі опису вакансії.

Отже, на виході відповідного компоненту, генерується навчена модель визначеної композитної структури, що в подальшому може бути використано для функціонування компоненту «Моделі прогнозування».

1.2.5 Компонент «Модель прогнозування»

Даний компонент реалізує механізм використання, отриманої в результаті навчання, моделі для прогнозування ЗП на основі текстових даних відповідних вакансій. Модель отримується шляхом завантаження файлу з попередньо навченою, запакованою моделлю прогнозування, що розміщено у відповідній папці «salary_predictor/models». Дані, для прогнозування, надходять із компоненту «Доповнення актуальних вакансій».

Текстові дані, проходять обробку, перед тим як почати процес прогнозування. Відповідні дані перетворюються до вигляду, який може

обробити відповідна модель. Заміни та репрезентація класів ЗП розміщено у папці «salary_predictor/dataset», де окрім файлу навчального набору даних «dataset.json», існують файли «replacement.json» та «salary_decode.json» відповідно.

Функціонал прогнозування реалізовано, у файлі «salary_predictor/__init__.py», який використовує попередньо навчену модель прогнозування та відповідні заміни та декодування, для того, щоб повернути прогнозоване значення ЗП, що теоретично відповідає вхідному опису вакансії.

Отже, результатом роботи відповідного компонента, є набір діапазонів ЗП, кількість яких збігається з кількістю переданих вакансій, що було отримано у процесі прогнозування ЗП попередньо навченою моделлю.

1.2.6 Компонент «Доповнення актуальних вакансій»

Даний компонент реалізує механізм збору та обробки актуальних вакансій, схожим чином до компоненту «Набір навчальних даних». Відмінність полягає в тому, що у даному випадку, жодне з полів не відкидається. Дані розділяються на дві групи, ті які мають поле ЗП, та ті для яких воно є відсутнім.

Після чого, для вакансій без поля «salary» здійснюється процес прогнозування ЗП, використовуючи компонент «Модель прогнозування». Далі, відповідні дані об'єднуються у один загальний датасет, де існують вакансії зі справжнім полем «salary» та передбаченим. Тобто, створюється доповнений набір актуальних вакансій.

Отже, результатом роботи відповідного компонента, є набір актуальних доповнених вакансій, які готові до передачі на вхід компонента «База даних» для збереження та подальшого процесу рекомендації вакансій.

1.2.7 Компонент «База даних»

Даний компонент відповідає за зберігання актуальних доповнених вакансій та даних користувачів. Відповідний компонент використовується компонентами «Доповнення актуальних вакансій» для збереження набору зібраних та доповнених вакансій та компонентом «Інтерфейс чат боту» для збереження даних користувача, що буде використано для формування рекомендації

Отже, результатом роботи відповідного компонента, є сховище даних, які готові до використання у процесі формування рекомендації для користувача.

1.2.8 Компонент «Інтерфейс чат боту»

Даний компонент відповідає за взаємодію з користувачем. Однією з функцій даного компонента є збір даних необхідних для процесу рекомендації. Іншою функцією є відображення результатів функціонування системи, а саме створених рекомендації на основі відповідних даних та актуальних вакансій.

Отже, результатом роботи відповідного компонента, є процес збору даних користувача та відображення рекомендації.

1.3 Інтерпретація результатів навчання

Отже, в результаті проведення попереднього навчання моделі прогнозування ЗП, визначеної композитної структури (див 4.1.6), було отримано наступні метрики:

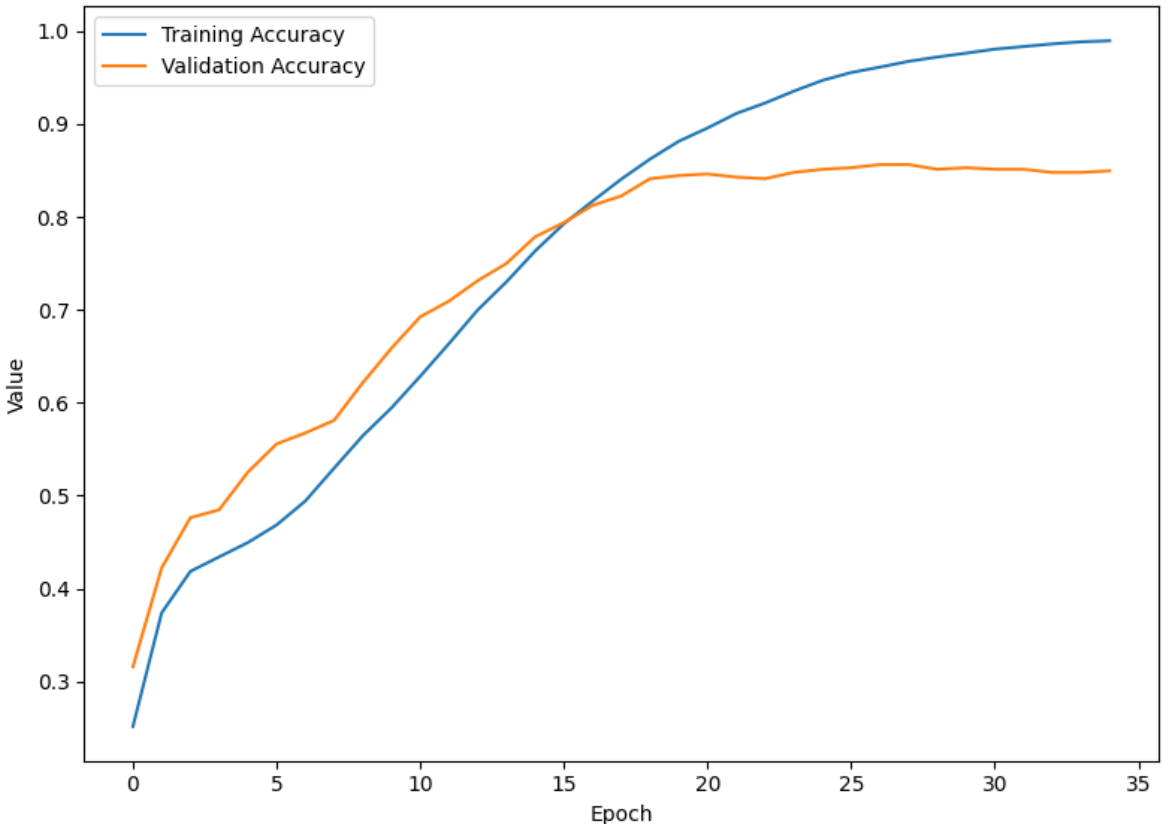


Рисунок 4.3.1 –Модель прогнозування ЗП. Значення «accuracy» під час навчання

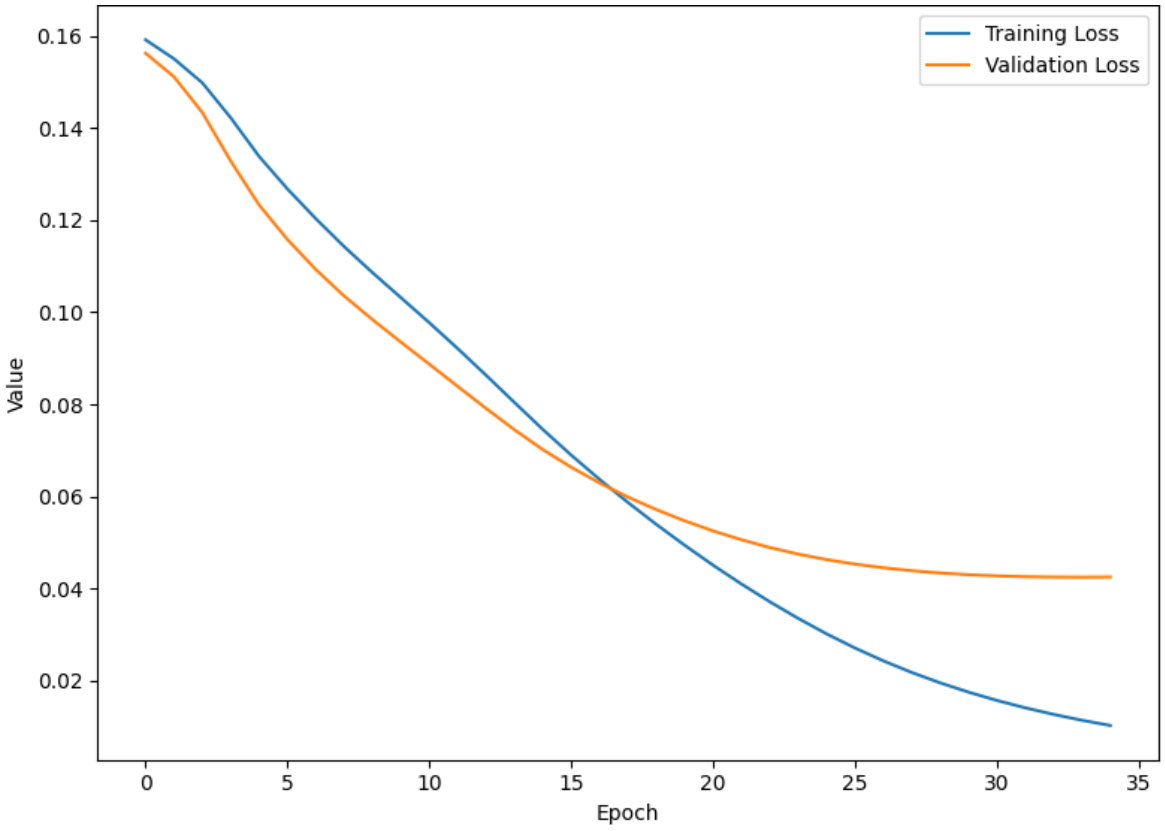


Рисунок 4.3.2 –Модель прогнозування ЗП. Значення «loss» під час навчання

В результаті процесу навчання, валідаційне значення метрики точності («accuracy») досягло значення 84.957%. У даному випадку, ця метрика репрезентує як у кількість прикладів описів вакансій модель віднесла до провального проміжку (класу) значень ЗП. Відповідні класи мають наступний вигляд:

- *Клас 1: до 15’000;*
- *Клас 2: від 15’000 до 24’000;*
- *Клас 3: від 24’000 до 40’000;*
- *Клас 4 від 40’000 до 80’000;*
- *Клас 5: від 80’000.*

Підвищення кількості епох не приносило бажаного ефекту по збільшенню точності моделі. Додатково, під час навчання, після 20-тої епохи, валідаційне значення точності зупинилося на значенні ~85%. В свою чергу, валідаціне значення функції втрат («loss»), після 25-тої епохи навчання, також досягло свого плато.

Тобто, кількість епох навчання можна зменшити, для уникнення перенавчання. Також, для покращення відповідного результату, швидше за все, необхідно збільшувати розмір начального набору даних.

1.4 Висновки до розділу

1. *Описано математичне забезпечення чат-боту з пошуку роботи, включаючи підхід до вирішення задачі рекомендації вакансій. Визначено необхідні дані для процесу рекомендацій, а також способи їх обробки. До даного переліку було віднесено: регіон пошуку, професія, очікувана ЗП та інтервал отримання рекомендацій.*
2. *Розглянуто підхід до прогнозування заробітної плати на основі тестових даних. Було прийнято рішення використовувати композитну модель, що поєднує модель штучних нейронних мереж (ANN) і модель*

одновимірних згорткових нейронних мереж (1D-CNN) для обробки даних. На основі визначеного перелік полів вакансій, на основі яких буде проводитись прогнозування, було здійснено розбиття на категоріальні та текстові дані. До категоріальних даних було віднесено: «region», «remote_type», «employment_type», «seniority». В свою чергу, до текстових: «title», «company», «additional_info», «description»;

3. *Розроблено механізм категоризації та токенізації для відповідних даних. Для категоріальних даних, було виділену множину можливих значень та створено «one-hot» репрезентації загальним вектором розмірністю 16. Для текстових даних, було прийнято рішення об'єднати відповідні виділенні текстові поля, а механізм токенізації інтегрувати в структуру відповідної текстової моделі 1D-CNN;*

4. *Проведено детальний опис обраної композитної моделі та її архітектури. Обрано модель «ANN Categorical Mode» для обробки категоріальних даних і модель «1D-CNN Text Model» для обробки текстових даних. Виходи цих моделей об'єднано в ANN структуру. Визначено перелік використаних гіперпараметрів.*

Проведено процес навчання обраної композитної моделі. Зібрано та наведено метрики накопичені в процесі навчання, а саме: точність (accuracy) та функцію втрат (loss). В результаті навчання, було отримано точність моделі прогнозування ЗП на рівні 85%. Проведено інтерпретацію результатів та формування висновків проведення навчання моделі. На основі яких, було припущено, що для зменшення перенавчання моделі можна зменшити кількість. А для підвищення точності моделі, краще збільшити розмір навчального набору даних.

6.1.3.9. Формування наукової новизни, практичної цінності та висновків за результатами виконання БКР.

Формування наукової новизни, практичної цінності та висновків за результатами виконання БКР детально описано у розділі 3. Мета і завдання переддипломної практики бакалаврів за освітньо-професійною програмою.

Зазначимо, що формалізація наукової новизни та практичної цінності результатів БКР потрібна лише для креативних робіт у яких отримані наукові і прикладні результати. Таким чином автори навчального посібника заохочують і всіляко підтримують творчі роботи студентів зорієнтовані не тільки на виконання інженерної розробки в рамках БКР, а і на проведення глибших наукових і прикладних досліджень за обраною темою БКР.

Студенти, автори творчих робіт, як правило, продовжують розвивати і поглиблювати тематику БКР вже у своїх магістерських дисертаціях.

Далі детальніше зупинимось на особливостях формалізації висновків за результатами виконання БКР які є обов'язковим розділом кожної БКР.

Висновки, за результатами виконання БКР, формалізовані за авторським алгоритмом, наведеному у цьому навчальному посібнику.

Перший пункт висновків продукується на основі результатів огляду існуючих рішень за формулою «Досліджено і встановлено» і систематизується у формі таблиці порівняльного аналізу.

Другий пункт відображає елементи наукової, або науково-технічної новизни результатів роботи/інженерного рішення у порівнянні з існуючими рішеннями що розглядались у попередньому розділі.

Третій пункт демонструє практичну цінність кінцевого результату роботи/інженерного рішення: наукову, науково-технічну, економічну, соціальну або іншу, яку можна отримати в результаті застосування результатів роботи/інженерного рішення.

Четвертий пункт показує перспективи подальших досліджень у задекларованому напрямі наукових досліджень/ інженерного рішення.

Дуже бажано у пунктах висновків вказувати конкретні значення кількісних і якісних показників у абсолютних чи відносних одиницях.

Продемонструємо на прикладі структуру і зміст висновків на прикладі. Назви розділів і підрозділів прикладів виділено курсивом.

Приклад 1. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»].

ВИСНОВКИ

1. В роботі досліджено основну проблематику пошуку вакансій, на основі якої було вирішено, що для підвищення ефективності пошуку роботи необхідно створити систему, що дозволяє отримувати вакансії з різних джерел для пошуку роботи та реалізувати функціонал передбачення заробітної плати для вакансій без відповідного поля. Було розглянуто основні існуючі засоби для пошуку роботи, до яких належать наступні веб-сайти: Djinni.co, Jooble.ua, Robota.ua та Work.ua. На основі аналізу було вирішено, що основними недоліками кожного розглянутого існуючого рішення дійсно є використання лише внутрішньої бази даних як джерела вакансій та відсутність функціоналу передбачення ЗП для відповідних вакансій. Було проведено аналіз потенційних видів інтерфейсу відповідної системи, а саме проаналізовано переваги та недоліки асистенту, веб-сайту та чат боту. На основі аналізу, було вирішено обрати саме інтерфейс у вигляді чат боту як рішення що є досить легким в реалізації, мультиплатформне, персоналізоване, зі зручним способом сповіщення користувача та мінімальною необхідністю в дизайні.

2. Було висунуто припущення про існування зв'язку між описом вакансій та значення заробітної плати. На основі відповідного припущення було вирішено проаналізувати основні алгоритми обробки природної мови (NLP)для виділення відповідного взаємозв'язку. Для цього було виділено

основний перелік рішень для обробки текстових даних, а саме: наївний Баєсів класифікатор (NBC), Ансамблеві моделі типу random forest (RF), одновимірні згорткові нейронні мережі (1D-CNN). На основі порівняльного аналізу було вирішено обрати 1D-CNN, як найкращий варіант. Оскільки 1D-CNN мають високі показниками узагальнюючої здатності, стійкості до шуму та результативності прогнозування. Додатково, дана модель має низькі показники для схильності до перенавчання та впливу розміру та якості датасету на результати навчання.

3. *Рзробблено загальну структуру чат боту на основі бізнес-профілю Еріксона-Пенкера та створено діаграми компонентів і процесів взаємодії між компонентами. Розроблено персоналізовані алгоритми збору даних з відповідних платформ з вакансіями (Djinni.co, Robota.ua та Work.ua). Було проведено аналіз отриманих датасетів. На основі якого, було вирішено, що для створення навчального набору даних потрібно провести спробу покращення даних. Для цього було проаналізовано значення кожного поля. На основі відповідного аналізу було прийнято рішення замінити значення поля «region» на три можливих значення, а саме: «відсутній», Київ, Інші. Заміна такого типу дозволила приблизити розподіл значень відповідного поля до рівномірного. Також, оскільки поле ЗП є цільовим, було вирішено також наблизити розподіл значення ЗП до рівномірного, застосовуючи заміни на певні класи ЗП. В результаті підбору меж відповідних класів, було отримано наступних 5 класів: К1 (до 15’000), К2 (від 15’000 до 24’000), К3 (від 24’000 до 40’000), К4 (від 40’000 до 80’000), К5 (від 80’000).*

4. *Проведено аналіз підходів до вирішення задачі рекомендації вакансій, на основі якого було прийнято рішення, що процес рекомендування буде базуватись на регіоні пошуку, назві професії, очікуваній ЗП та інтервалі отримання рекомендації. Імплементація функціоналу звуження можливих рекомендації на основі регіону та мінімальної очікуваної ЗП та інтегрування інтервалу отримання рекомендації, не викликала проблем. Але, визначення професії на основі опису вакансії виявилось більш складним завданням. Було*

запропоновано визначати професію на основі назви вакансії використовуючи метод NLP, що базується на основі обчислення косинусу подібності (cosine similarity) [15]. Основною ідеєю такого підходу є репрезентація професії та назви вакансії, у вигляді векторів. Потім, між кожним словом професії та назви вакансії обчислити відповідний косинус подібності, та обрати максимальне значення. Використовуючи певний поріг (у даному випадку – 0.9), можна виділити вакансії, що підходять до відповідної професії, що було наданого користувачем. Для формування списку релевантних вакансій, вони одночасно сортуються за значення косинусу подібності та датою створення вакансії. Додатково, ігноруються вакансії, які вже було рекомендовано поточному користувачу, для запобігання повторювань.

5. Розглянуто підходи до вирішення проблеми передбачення ЗП на основі опису вакансій. В результаті було вирішено розділити дані, що описують вакансії, на два можливих класи, а саме категоріальні та текстові дані. Для категоріальних даних було вирішено використовувати «one-hot» кодування після якого буде створено вектор розмірністю 16. Для текстових даних було вирішено проводити об'єднання обраних полів в одну стрічку тексту. Також, було вирішено застосовувати процес токенізації даних, але його було інтегровано в структуру відповідної моделі. Для самого процесу передбачення ЗП, було запропоновано скомбінувати дві моделі, а саме ANN та 1D-CNN. Відповідна композитна модель дозволяє окремо обробляти категоріальну та текстову частину даних та підвищує загальну точність прогнозування та ефективність моделі прогнозування [17]. В результаті проведення навчання моделі , було отримано значення точності на рівні 84.957%, що можна назвати задовільним результатом. Додатково, на основі інтерпретації результатів навчання, було припущено про необхідність зменшення кількості епох навчання для зменшення впливу перенавчання та необхідність збільшення навчального набору даних для підвищення точності моделі.

6. Проведено верифікацію та валідацію системи чат боту з пошуку роботи. Проведено аналіз та перевірку виконання вимог, розроблення структурних представлень та перевірку їх відповідності бізнес правилам. На основі результатів порівняння, можна зробити висновок, що загальна мета відповідної роботи була досягнута. Система дозволяє отримувати рекомендації з кількох джерел вакансій та реалізовано функціонал передбачення ЗП для вакансій без відповідного поля.

7. Розроблений чат бот дозволяє підвищити ефективність пошуку роботи, оскільки надає можливість отримувати вакансії з кількох джерел та дозволяє підвищити інформативність вакансій без поля ЗП за допомогою передбачення відповідного значення на онові опису вакансії. Але існує потенціал для подальшого вдосконалення даної системи в цілому. Одним з основних напрямків можливих покращень, є вдосконалення моделі передбачення ЗП, а саме накопичення більшої кількості навчальних даних та більш детальний аналіз впливу гіперпатаметрів на розроблену композитну модель.

Ще одним потенційним напрямком вдосконалення чат боту, як ефективного рішення для пошуку роботи, є розширення даної системи до повної екосистеми. А саме, виділення певного основного ядра системи та подальше його перетворення на окрему та самостійну частину, що дозволить легко інтегрувати в систему інші платформи та інтерфейси взаємодії з користувачем, окрім чат боту. Відповідне розширення дозволить додати додатковий функціонал до системи, а саме робити аналітику профілів користувачів та доступних вакансій, вдосконалення алгоритму рекомендації вакансій, функціонал обробки резюме, вдосконалення наданого резюме, аналітика для наданого резюме тощо. Тобто метою подальшої розробки системи може стати підвищення ефективності та інтеграція у процес пошуку роботи на усіх етапах працевлаштування.

Приклад.2 [Для прикладу використані фрагменти розділу МД студента Івана САВЧУКА започатковані у співавторстві в процесі вивчення дисциплін «Основи наукових досліджень» та «НДР за темою МД» і виконання МД на тему «Метод системної інженерії уніфікованої моделі DevOps-системи для продуктів IT індустрії», викладач дисциплін та науковий керівник МД Маслянко Павло Павлович, к.т.н., доцент»].

ВИСНОВКИ

В магістерській дисертаційній роботі одержано такі нові теоретичні та практичні результати:

1) Досліджено існуючі способи формалізації концепту DevOps, впровадження DevOps в організаціях, підходи до продукування систем керуючись принципами DevOps. В ході дослідження, встановлено, що наразі відсутній консенсус для дефініції концепту DevOps, роль концепту недооцінена і зазвичай використовується лише розробниками в контексті налаштування програмного забезпечення. Встановлено актуальність та необхідність формалізації концепту DevOps та розробки науково-обгрунтованого методу системної інженерії DevOps-систем для продукування продуктів, товарів та послуг на всіх стадіях їх життєвого циклу.

2) Запропоновано формалізацію концепту DevOps за допомогою апарату теорії множин. Виконано синтез означень концепту DevOps та DevOps системи, відповідно до яких були в подальшому отримані структурне та динамічне представлення концепту DevOps. Описано призначення та область застосування концепту DevOps для продукування та / або підтримки IT продуктів на всіх стадіях їх життєвого циклу.

Вперше запропоновано метод системної інженерії уніфікованої моделі DevOps-системи для продуктів IT індустрії на основі бізнес-профілю Еріксона-Пенкера, що охоплює всі стадії життєвого циклу продуктів IT індустрії.

4) Проведено імплементацію методу системної інженерії для продукування QA системи медичного спрямування, на основі: методу найближчих сусідів та мереж трансформерів.

5) Валідація розробленої системи QA DevOps-система «Medical Search», показує, що розроблена система не поступається реальній системі ChiQA, яка в результаті верифікації та валідації на датасеті MEDIQA, продемонструвала, за визначеним класом метрик, результати $Precision@10 = 0.517$, $MRR@10 = 0.895$, *а у розробленої QA DevOps-системи «Medical Search», ці результати складають* $Precision@10 = 0.574$, $MRR@10 = 0.913$.

6) В контексті подальших досліджень DevOps як концепту, як складової методу системної інженерії DevOps-систем для продукування IT продуктів, та розвитку QA DevOps-системи, передбачені наступні дослідження:

– *Верифікація та валідація методу системної інженерії DevOps-систем для продукування IT продуктів в умовах реальних організацій.*

– *Розробки наступної версії методу з урахування отриманого, від реальних організацій досвіду використання та зворотніх зв'язків, з урахуванням кросплатформного обчислювального середовища.*

– *У випадку відсутності можливості застосування методу для організацій, провести імітаційне моделювання їх діяльності у процесі продукування IT продуктів.*

Для розширення можливостей QA DevOps-системи необхідно провести дослідження способів застосування фільтрів в процесі семантичного пошуку, можливості комбінації статистичного та семантичного підходів для пошуку текстових документів, способи реалізації пошуку за допомогою графічних зображень в додаток до текстового опису.

6.1.3.10. Список використаної літератури

Список використаної літератури потрібно оформлювати за вимогами, викладеними на сторінці «Вимоги до оформлення документації» https://pma.fpm.kpi.ua/uk/studentam/ofitsiyno/vimogi-do-oformlennya-dokumentatsiyi сайту кафедри прикладної математики.

У Списку використаної літератури обов'язково повинні бути посилання на власні роботи студента якщо вони є, на які у свою чергу, повинні бути посилання в тексті БКР.

У «Списку використаної літератури» не допускаються:

1) посилання на джерела, що не мають відношення до БКР;
2) посилання на сторінки Вікіпедії та схожі ресурси, у яких відсутній процес наукового рецензування;
3) сайти рефератів та інші накопичувачі документів, що не мають наукової цінності;
4) курсові й лабораторні роботи;
5) файлообмінники (та інші схожі ресурси).

Замість посилання на файл із відсканованим джерелом потрібно наводити бібліографічний опис друкованого видання.

Приклад 1. [Для прикладу використані фрагменти розділів БКР студентки Катерини ПАВЛОВСЬКОЇ започатковані у співавторстві з науковим керівником в процесі вивчення дисциплін «Системи Data Science» і виконання БКР на тему «Математичне та програмне забезпечення чат боту з пошуку роботи», викладач дисциплін та науковий керівник БКР Маслянко Павло Павлович, к.т.н., доцент»].

ПЕРЕЛІК ПОСИЛАНЬ

1. Collobert, R. Natural Language Processing (Almost) from Scratch / Ronan Collobert [та ін.] // The Journal of Machine Learning Research. – 2011. – Т. 12. – С. 2493–2537.

2. Gruetzemacher R. Deep Transfer Learning & Beyond: Transformer Language Models in Information Systems Research / Ross Gruetzemacher, David Paradice // ACM Computing Surveys. – Т. 54, № 10. – С. 1–35.

3. Manning C. D. Introduction to Information Retrieval / Christopher D. Manning, Prabhakar Raghavan, Hinrich Schutze. – [Б. м.] : Cambridge University Press, 2008. – 506 с.

4. Jurafsky D. Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition / Daniel Jurafsky, James H. Martin. – [Б. м.] : Youcanprint, 2023. – 638 с.

5. McCallum A. A comparison of event models for naive bayes text classification / Andrew McCallum, Kamal Nigam // AAAI-98 workshop on learning for text categorization. – 1998. – Т. 752. – С. 41–48.

6. Frank E. Naive bayes for text classification with unbalanced classes / Eibe Frank, Remco R. Bouckaert // Knowledge Discovery in Databases: PKDD 2006: 10th European Conference on Principles and Practice of Knowledge Discovery in Databases. – 2006. – С. 503–510.

7. Breiman L. Random forests / Leo Breiman // Machine learning. – 2001. – Т. 45. – С. 5–32.

8. Liaw A. Classification and regression by randomForest / Andy Liaw, Matthew Wiener // R news. – 2002. – Т. 2. – С. 18–22.

9. Ho T. K. Random decision forests / Tin Kam Ho // Proceedings of 3rd international conference on document analysis and recognition. – [Б. м.], 1995. – С. 278–282.

10. Kim Y. Convolutional Neural Networks for Sentence Classification / Yoon Kim // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). – [Б. м.], 2014. – С. 1746–1751.

11. Zhang Y. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification / Ye Zhang, Byron Wallace // Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers). – [Б. м.], 2017. – С. 253–263.

12. Shen, Y. A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval / Yelong Shen [та ін.] // Proceedings of the 23rd ACM international conference on conference on information and knowledge management. – [Б. м.], 2014. – С. 101–110.

13. Kiranyaz, S. 1D convolutional neural networks and applications: A survey / Serkan Kiranyaz [та ін.] // Mechanical Systems and Signal Processing. – 2021. – Т. 151. – 107398.

14. Maslianko P. P. Метод системної інженерії систем нейронного машинного перекладу / Pavlo P. Maslianko, Yevhenii P. Sielskyi // KPI Science News. – 2021. – № 2. – С. 48.

15. Rahutomo F. Semantic cosine similarity / Faisal Rahutomo, Teruaki Kitasuka, Masayoshi Aritsugi // The 7th international student conference on advanced science and technology ICAST. – [Б. м.], 2012. – С. 1.

16. Rodríguez, P. Beyond one-hot encoding: Lower dimensional target embedding / Pau Rodríguez [та ін.] // Image and Vision Computing. – 2018. – Т. 75. – С. 21–31.

17. Wang X. Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts / Xingyou Wang, Weijie Jiang, Zhiyong Luo // Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. – [Б. м.], 2016. – С. 2428–2437.

18. Kim Y. Convolutional Neural Networks for Sentence Classification / Yoon Kim // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). – [Б. м.], 2014. – С. 1746–1751.

Goodfellow I. Deep learning / Ian Goodfellow, Yoshua Bengio, Aaron Courville. – [Б. м.] : MIT press, 2016. – 798 с.

Приклад 2. [Для прикладу використані фрагменти розділу МД студента Івана САВЧУКА започатковані у співавторстві в процесі вивчення дисциплін «Основи наукових досліджень» та «НДР за темою МД» і виконання МД на тему «Метод системної інженерії

уніфікованої моделі DevOps-системи для продуктів IT індустрії», викладач дисциплін та науковий керівник МД Маслянко Павло Павлович, к.т.н., доцент»].

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. *Senapathi M., Buchan J., Osman H. DevOps capabilities, practices, and challenges: Insights from a case study. EASE 2018: Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering, Christchurch, New Zealand, 28 June – 29 June, 2018. New York, NY: Association for Computing Machinery. P. 57–67.*
2. *Jha P., Khan R. A review paper on DevOps: Beginning and more to know. International Journal of Computer Applications. 2018. Vol. 180, No.48. P.16–20.*
3. *DevOps 101 with Atlassian. URL: https://www.atlassian.com/blog/wp-content/uploads/devops-101-atlassian.pdf (Last accessed: 01.09.2022).*
4. *Competitive Advantage through DevOps: Improving speed, quality and efficiency in the digital world. Harvard Business Review Analytic Service. 2019. URL: https://inthecloud.withgoogle.com/hbr-report-19/hbr-research-report-google-cloud.pdf (Last accessed: 01.09.2022).*
5. *2021 State of DevOps. 2021. URL: https://blog.claydesk.com/wp-content/uploads/2021/07/Puppet-State-of-DevOps-Report-2021.pdf (Last accessed: 01.09.2022).*
6. *Pietrantuono R. et. al. Towards continuous software reliability testing in DevOps. IEEE/ACM 2019: Proceedings of 14th International Workshop on Automation of Software Test, Montreal, QC, Canada, 27 May, 2019. P. 21–27.*
7. *What is DevOps? URL: https://www.atlassian.com/devops (Last accessed: 01.09.2022).*
8. *Altunel H., Say B. Software product system model: a customer-value oriented, adaptable, devops-based product model. SN Computer Science. 2022. Vol. 3, No. 1. P.1–11.*

9. *Bucena I., Kirikova M. Simplifying the DevOps Adoption Process. BIR Workshops. 2017. Vol. 1898, No. 14. P. 1–15.*

10. *Майстренко О. С., Маслянко П. П. Моделювання організаційних систем на основі бізнес-профіля. Компьютерное моделирование в наукоемких технологиях (КМНТ-2012): матеріали наук.-тех. конференції, м. Харків, 24–27 квітня 2012 р. Харків: ХНУ імені В.Н.Каразіна, 2012. С.380–381.*

11. *Маслянко П. П., Савчук І. В. DevOps – Концепт і структурне представлення. Наукові вісті КПІ. 2021. 14 лют. (№ 4). С. 39–51.*

12. *Systems and software engineering – Vocabulary. URL: https://www.iso.org/obp/ui/#iso:std:iso-iec-ieee:24765:ed-1:v1:en (Last accessed: 01.09.2022).*

13. *Eriksson H.-E.,Penker M. Business modeling with UML: Business Patterns at Work. New York, NY: John Wiley & Sons, 2000. 459 p.*

14. *14. A. Kossiakoff et al. Systems Engineering Principles and Practice. / ed. V. K. Batovrin. Moscow, Russia: DMK Press, 2014. 624 p.*

15. *Hitchins D. K. Systems Engineering: A 21st Century Systems Methodology. New York, NY: John Wiley & Sons, 2007. 528 p.*

16. *Maslianko P. P., Sielskyi Y.P. Method of system engineering of neural machine translation systems. KPI Science News. 2021. No. 2. P. 34–42.*

17. *Maslianko P. P., Maystrenko O. S. The system engineering of organizational system informatization projects. KPI Science News. 2008. No. 6. P. 34–42.*

18. *Pedra M. L., de Silva M. F., Azevedo L. G. DevOps adoption: Eight emergent perspectives. (Preprint / arXiv:2109.09601). 2021.*

19. *Sallin M., Kropp M., Anslow C., Quilty J. W., Meier A. Measuring software delivery performance using the four key metrics of DevOps. XP 2021. Proceedings of Agile Processes in Software Engineering and Extreme Programming., Montreal, QC, Canada, 21 May – 25 May, 2019. New York, NY: Association for Computing Machinery. P. 103–119.*

20. Zhai C. X., Massung S. Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining. New York, NY: ACM Books, 2016. 531 p.

21. Sennrich R., Haddow B., Birch A. Neural Machine Translation of Rare Words with Subword Units. (Preprint / arXiv:1508.07909). 2016.

22. Lee J., Cho K., Hofmann T. Fully Character-Level Neural Machine Translation without Explicit Segmentation. Transactions of the Association for Computational Linguistics. 2017. Vol. 5. P. 365–378.

23. Rong X. word2vec Parameter Learning Explained. (Preprint / arXiv:1411.2738). 2014.

24. Bojanowski P., Grave E., Joulin A., Mikolov T. (2017). Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics. 2017. Vol. 5. P. 135–146.

25. Mikolov T., Chen K., Corrado G. S., Dean J. Efficient Estimation of Word Representations in Vector Space. (Preprint / arXiv:1301.3781). 2013.

26. Schuster M., Nakajima K. Japanese and Korean voice search. IEEE 2012: Proceedings of 37th International Conference on Acoustics, Speech and Signal Processing, Kyoto, Japan, 25 May – 30 May, 2012. P. 5149–5152.

27. Vaswani, A. et. al. Attention is All you Need. NIPS 2017: Proceedings of 30th Advances in neural information processing systems, Long Beach, CA, USA, 4 Dec – 9 Dec, 2017. P. 6000–6010.

28. Sutskever I., Vinyals O., Le Q.V. Sequence to Sequence Learning with Neural Networks. NIPS 2014: Proceedings of 27th Advances in neural information processing systems, Montreal, QC, Canada, 8 Dec – 11 Dec, 2014. P. 3104–3112.

29. Szegedy C., Ioffe S., Vanhoucke V., Alemi A. A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. AAAI 2017: Proceedings of 31st AAAI conference on artificial intelligence, San-Francisco, CA, USA, 4 Feb – 9 Feb, 2017, P. 4278–4284.

30. Ba J., Kiros J. R., Hinton G. E. Layer Normalization. (Preprint / arXiv:1607.06450). 2016.

31. Devlin J., Chang M., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (Preprint / arXiv:1810.04805). 2019.

32. Bommasani R., et. al. On the opportunities and risks of foundation models. (Preprint / arXiv:2108.07258). 2021.

33. Khalid U., Beg M., Arshad M. RUBERT: A Bilingual Roman Urdu BERT Using Cross Lingual Transfer Learning. (Preprint / arXiv:2102.11278). 2021.

34. Wang W., Wei F., Dong L., Bao H., Yang N., Zhou M. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. NIPS 2020: Proceedings of 33rd Advances in neural information processing systems, virtual, 6 Dec – 12 Dec, 2020. P. 5776–5788.

35. Yokozuka M., Koide K., Oishi S., Banno A. LiTAMIN2: Ultra Light LiDAR-based SLAM using Geometric Approximation applied with KL-Divergence. IEEE/ICRA 2021: Proceedings of 2021 IEEE International Conference on Robotics and Automation, Xi'an, China, 30 May – 5 June, 2021. P. 11619–11625.

36. Sanh V., Debut L., Chaumond J., Wolf T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. (Preprint / arXiv: 1910.01108). 2019.

37. Barz B., Denzler J. Deep Learning on Small Datasets without Pre-Training using Cosine Loss. WACV 2020: Proceedings of 2020 IEEE Winter Conference on Applications of Computer Vision, Snowmass Village, CO, USA, 1 Mar – 5 Mar, 2020. P. 1360–1369.

38. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. (Preprint / arXiv: 1908.10084). 2019.

39. Hastie T., Tibshirani R., Friedman J. H. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed. New York, NY: Springer, 2017. 745 p.

40. Ge T., He K., Ke Q., Sun J. Optimized Product Quantization. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2013. Vol. 36, No. 4. P. 744–755.

41. Johnson J., Douze M., Jégou H. Billion-scale similarity search with GPUs. IEEE Transactions on Big Data. 2019. Vol. 7, No. 3. P. 535–547.

42. Sakata W., Shibata T., Tanaka R., Kurohashi S. FAQ retrieval using query-question similarity and BERT-based query-answer relevance. SIGIR 2019: In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21 Jul – 25 Jul, 2019. New York, NY: Association for Computing Machinery. P. 1113–1116.

43. Schwartz I. Ensemble of MRR and NDCG models for Visual Dialog. (Preprint / arXiv: 2104.07511). 2021.

44. Abacha A. B., Shivade C., Demner-Fushman D. Overview of the mediqa 2019 shared task on textual inference, question entailment and question answering. BioNLP@ACL 2019: Proceedings of the 18th BioNLP Workshop and Shared Task, Florence, Italy, August 1, 2019. P. 370–379.

45. Pennington J., Socher R., Manning C. D. Glove: Global vectors for word representation. EMNLP 2014: Proceedings of the 2014 conference on empirical methods in natural language processing, Doha, Qatar, 26 Oct – 28 Oct, 2014. P. 1532–1543.

46. Joulin A., Grave E., Bojanowski P., Douze M., Jégou H., Mikolov T. Fasttext. zip: Compressing text classification models. (Preprint / arXiv:1612.03651). 2016.

47. Zarour M., Alhammad N., Alenezi M., Alsarayrah K. Devops Process Model Adoption in Saudi Arabia: An Empirical Study. Jordanian Journal of Computers and Information Technology. 2020. Vol. 6, No. 3. P. 234–246.

48. Demner-Fushman D., Mrabet Y., Abacha A. B. Consumer health information and question answering: helping consumers find answers to their health-related information needs. Journal of the American Medical Informatics Association. 2020. Vol. 27, No. 2. P. 194–201.

49. Маслянко П. П., Савчук І. В. Математичне і програмне забезпечення DevOps-системи продукування систем типу Question-Answering // Прикладна математика та комп'ютинг. ПМК, 2022: п’ятнадцята наук. конф.

магістрантів та аспірантів, Київ, 16—18 лист. 2022 р. : зб. тез доп. / [редкол.: Дичка І. А. та ін.]. — К. : Просвіта, 2022. — 368 с. с.78-85 ISBN 978-617-7010-23-3

Поточний/календарний контроль № 1

Поточний/календарний контроль № 1 проведення практики – за результатами виконання індивідуального завдання практики і змісту документів класу завдань 1, а саме : подання заяви з темою БКР за підписом наукового керівника (Додаток 4), формалізації постановки задачі (ФПЗ), шаблону БКР – першої версії БКР за форматом і вимогами Положення про державну атестацію бакалаврів КПІ ім. Ігоря Сікорського з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами.

6.2. Клас завдань 2. Завдання продукування результатів практики

Для формування наказу про допуск до випускної атестації та затвердження теми і наукових керівників бакалаврської атестаційної роботи студент, разом зі своїм науковим керівником, подає завідувачу кафедри остаточну тему, формалізацію постановки задачі і завдання БКР.

На основі цих матеріалів завідуючий кафедри приймає рішення про допуск до захисту бакалаврської атестаційної роботи, затверджує тему та керівника бакалаврської атестаційної роботи. Кафедра оформлює подання в деканат факультету для формування відповідного наказу.

Студент повинен представити керівникам практики від кафедри першу версію шаблону БКР з виконаними всіма розділами, яка погоджена з науковим керівником БКР та керівником практики від бази практики. За результатами виконаної роботи приймається рішення про допуск до заліку з практики. Якщо було виявлено недоліки, то студенту повертають версію шаблону БКР для доопрацювання.

Поточний/календарний контроль проведення практики № 2 - за результатами виконання завдань і змісту документів, а саме : перша версія шаблону БКР з затвердженими остаточними темою, формалізацією постановки задачі і завданнями БКР та виконаними всіма розділами.

6.3. Клас завдань 3. Завдання апробації результатів практики

1) Провести верифікацію і валідацію результатів практики.
2) Формалізувати наукову новизну, практичну цінність та висновки за результатами виконання БКР.
3) Підготувати, та затвердити у н/керівника, презентацію результатів Звіту з переддипломної практики для заліку з практики;

Поточний/календарний контроль проведення практики №3 – за результатами виконання завдань і змісту документів класу завдань 3.

6.4. Клас завдань 4. Завдання звітування (семестровий контроль)

1) Підготувати Звіт з практики;
2) Подати до комісії з прийому результатів переддипломної практики остаточну заяву з темою БКР, заповнений і підписаний щоденник з практики, звіт з практики, презентацію звіту з практики;
3) Захистити результати практики;
4) Ознайомитись з порядком попереднього захисту БКР.

Семестровий контроль (залік) проведення практики - за результатами звіту з практики, виконання завдань і змісту документів для класу завдань 4.

Після закінчення практики студенти повинні представити керівнику практики від кафедри письмовий звіт разом із щоденником у встановлений термін (не пізніше трьох днів після дати закінчення практики) для перевірки, рецензування і допуску до захисту. Письмова

рецензія керівника практики від кафедри та наукового керівника БКР заносяться до щоденника студента.

Звіт із практики має містити відомості про виконання студентом програми практики та індивідуального завдання. Систематизація зібраних матеріалів здійснюється студентом під час практики і завершується протягом спеціально виділеного для цієї мети часу, відповідно до програми практики.

Звіт захищається студентом на комісії [3], призначеній завідувачем кафедри. До складу комісії входять керівники практики від кафедри, керівники БКР. Комісія приймає залік протягом перших десяти днів після закінчення практики.

Семестровий контроль проведення практики (Залік) – за результатами звіту з практики, виконання завдань і змісту документів, а саме: остаточну заяву, щоденник з практики, звіт з практики, презентація звіту з практики, результати захисту звіту з практики.

7. ВИМОГИ ДО ЗВІТУ З ПРАКТИКИ

Після закінчення практики студенти повинні представити керівнику практики від кафедри письмовий звіт з практики, презентацію результатів практики.

Звіт з практики повинен інтегрувати результати виконання всіх завдань дорожньої карти проходження практики у єдиний системний документ.

Структура і зміст розділів Звіту з практики складається з таких розділів або документів:

- Титульна сторінка Звіту (за зразком Додаток 5);
- Зміст Звіту з практики;
- Перелік умовних позначень, скорочень і термінів (за необхідністю);
- Вступ, з коротким оглядом проблемної області за якою виконується БКР, короткою аргументацією актуальності задекларованих досліджень, коротка анотація розділів Звіту;
- Формалізація постановки задачі БКР (об’єкт, предмет і мета дослідження/розробки, кінцевий результат виконання БКР);
- Перша повна версія БКР з виконаними всіма розділами і додатками;
- Додаток А до БКР - Демонстраційна версія програмного забезпечення (архітектура, короткий опис функціональності, мова програмування, результати роботи програмного забезпечення, тощо);
- Додаток В до БКР - Презентація першої повної версії БКР з виконаними всіма розділами (окремі слайди можуть бути ще не розроблені);
- Публікації за темою БКР (за наявності) опубліковані (подані і прийняті до публікації) у фаховому виданні;
- Висновки до звіту/підсумки практики;
- Додатки до звіту (за необхідності)
- Презентація звіту з практики;

Перелік посилань формується після кожного розділу/документу звіту.

Правила оформлення звіту за ДСТУ 3008-15 можна знайти за посиланням https://pma.fpm.kpi.ua/uk/studentam/ofitsiyno/vimogi-do-oformlennya-dokumentatsiyi

Захист звіту проводиться у вигляді доповіді та співбесіди на комісії, яка призначається завідувачем кафедри, за участю наукового керівника МД та керівника практики від кафедри.

8. ПІДСУМКИ ПРАКТИКИ

Підсумки практики формуються у відповідності до завдань розділів 3 МЕТА І ЗАВДАННЯ ПЕРЕДДИПЛОМНОЇ ПРАКТИКИ БАКАЛАВРІВ ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ та розділу 6 ІНДИВІДУАЛЬНЕ ЗАВДАННЯ ПРАКТИКИ і мають коротко охарактеризувати результати виконання кожного з цих завдань.

Клас завдань 1. Завдання формування і систематизації ресурсів для проходження практики:

1. Написати заяву встановленого зразка на ім’я завідувача кафедри з попередньою назвою теми БКР і проханням призначити наукового керівника;
2. Формалізувати постановку задачі БКР: об’єкт, предмет і мета дослідження/роботи, кінцевий результат виконання БКР;
3. Сформувати матрицю інформаційних ресурсів (МІР) необхідних для виконання БКР;
4. Розробити шаблон БКР – першої версії БКР за форматом і вимогами Положення про державну атестацію бакалаврів КПІ ім. Ігоря Сікорського [2] з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами.

4.1 Формалізація постановки задачі бакалаврської атестаційної роботи.

4.2 Матриця сутностей уніфікованої моделі проходження практики.

4.3. Шаблон бакалаврської атестаційної роботи

4.3.1. Вступ.

4.3.2. Основна частина з переліком розділів.

4.3.3. Обґрунтуванням актуальності тематики досліджень/розробок.

4.3.4. Огляд існуючих рішень за темою БКР.

4.3.5. Структурне представлення системи, підсистеми, програмного продукту.

4.3.6. Математичне забезпечення системи, підсистеми, програмного продукту, методи, моделі та алгоритми результатів досліджень

4.3.7. Програмне забезпечення системи, підсистеми, програмного продукту, методів, моделей та алгоритмів результатів досліджень

4.3.8. Формування наукової новизни, практичної цінності (за наявності) та висновків за результатами виконання БКР.

4.3.9. Список використаної літератури.

Структура та зміст звіту з практики відповідає цим завданням, таким чином результатом виконання індивідуального завдання є практично повністю підготовлена до захисту бакалаврська кваліфікаційна робота.

Клас завдань 2. Завдання продукування результатів практики:

1. Затвердити остаточну тему, формалізацію постановки задачі і завдання БКР. У разі необхідності уточнити зміст заяви;
2. Сформувати і захистити першу версію шаблону БКР з виконаними всіма розділами та додатками. Додаток А. Демонстраційний матеріал – презентація БКР, Додаток Б. Текст програмного коду;

Клас завдань 3. Завдання апробації результатів практики:

1. Провести верифікацію і валідацію результатів практики.
2. Формалізувати наукову новизну, практичну цінність та висновки за результатами виконання БКР.
3. Підготувати, та затвердити у н/керівника, презентацію результатів Звіту з переддипломної практики для заліку з практики;

Клас завдань 4. Завдання звітування (семестровий контроль):

1. Підготувати Звіт з практики;
2. Подати до комісії з прийому результатів переддипломної практики остаточну заяву з темою БКР, заповнений і підписаний щоденник з практики, звіт з практики, презентацію звіту з практики;

3. Захистити результати практики;

4. Ознайомитись з порядком попереднього захисту БКР.

Структура та зміст звіту з практики відповідає цим завданням, таким чином результатом виконання індивідуального завдання практики є практично повністю підготовлена до захисту бакалаврська кваліфікаційна робота.

9. ФОРМИ ТА МЕТОДИ КОНТРОЛЮ

Основним документом, за яким здійснюється контроль проходження практики, є щоденник практики, та результати виконання завдань Дорожньої карти практики.

Поточний контроль проходження практики здійснюють керівник практики від кафедри, керівник БКР і керівник від підприємства. Поточний контроль здійснюється перевіркою щоденника практики щотижня. Щоденник (Додаток 4) містить індивідуальне завдання, календарний план проходження практики, щотижневі записи. Керівники практики від підприємства та від кафедри щотижня перевіряє щоденник і записує свої зауваження. Після закінчення терміну практики керівник від підприємства надає в щоденнику відгук про проходження практики студентом і оцінює її результати оцінкою.

Керівник БКР вносить запис у щоденник і виставляє рекомендовану оцінку за результатами виконання індивідуального завдання практики.

Календарний контроль здійснюється перевіркою виконання чотирьох класів завдань згідно календарного плану.

Семестровий контроль проводиться згідно навчального плану, за результатами якого студент отримує залік/незараховано. Умови допуску до семестрового контролю: мінімальна позитивна оцінка з виконання індивідуального завдання з практики, мінімальний рейтинг 60 балів.

10. КРИТЕРІЇ ОЦІНЮВАННЯ

Семестровий контроль (залік) за підсумками практики проводиться на підставі захисту письмового звіту та презентації результатів виконання завдань Звіту, наданого щоденника проходження практики, оформленого відповідно до встановлених вимог, та відгуку наукового керівника. За підсумками атестації виставляється оцінка. Умови допуску до семестрового контролю: мінімальна позитивна оцінка за виконання індивідуального завдання, мінімальний рейтинг 60 балів.

Критерії оцінювання:

1) Наявність документів: звіту з практики з виконаними у повному обсязі індивідуальними завданнями практики; презентації звіту з практики; заповнений, оформлений і підписаний належним чином щоденник з практики - 60 балів.
2) Захист звіту практики та виконання індивідуального завдання практики, максимальна оцінка - 40 балів

Критерії оцінювання:

a) повне виконання індивідуального завдання практики – 40 балів;
b) неповне виконання індивідуального завдання –10 – 20 балів;
c) достатня відповідність змісту індивідуального завдання –0 – 5 балів.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою:

Кількість балів	*Оцінка*
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно
64-60	Достатньо
Менше 60	Незадовільно
Не виконані умови допуску	Не допущено

11. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Положення про організацію освітнього процесу в КПІ ім.Ігоря Сікорського: затверджено наказом Ректора, НАКАЗ № 7-124 від 20.07.2020. [електроний ресурс] . – Режим доступу – https://document.kpi.ua/regulations . – Назва з екрану.- Мова укр.

2. Положення про порядок проведення практики здобувачів вищої освіти Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» затверджено наказом Ректора, НАКАЗ № 7/172 від 24.09.2020. [електроний ресурс] . – Режим доступу – https://document.kpi.ua/files/2020_7-172.pdf. – Назва з екрану.- Мова укр.

3. Методичні рекомендації з питань організації практики студентів та складання робочих програм практики Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» [Текст] / Уклад.: Н. М. Лапенко, І.Л. Співак, І.В. Федоренко, О.М. Шаповалова; за заг. Ред. П.М. Яблонського. – К.: КПІ ім. Ігоря Сікорського, 2018. – 29 с.

4. Закон України Про наукову і науково технічну діяльність, [електроний ресурс]. – Режим доступу – https://zakon.rada.gov.ua/laws/show/848-19#Text–Назва з екрану.- Мова укр.

5. Положення про систему запобігання академічному плагіату в КПІ ім.Ігоря Сікорського: затверджено наказом Ректора, НАКАЗ № 1-76 від 25.02.2020. [електроний ресурс] . – Режим доступу – https://document.kpi.ua/files/2020_1-76.pdf. – Назва з екрану.- Мова укр.

ДОДАТОК 1 (ЗРАЗОК)

ЛІСТИНГ ПРОГРАМИ

Лістинг файлу data_extraction/__init__.py

```
import pandas as pd
import preprocessor
import salary_predictor


def get_unique_data(path: str) -> pd.DataFrame:
    """
        Отримує унікальні дані з вказаного шляху та повертає їх у вигляді DataFrame.

        Функція використовує обробник `preprocessor` для отримання унікальних записів з вказаного шляху,
        а потім виконує процес очищення даних, використовуючи метод `cleanup_process` з того ж обробника.

        Args:
            path (str): Шлях до даних.

        Returns:
            pd.DataFrame: Унікальні дані, очищені від зайвої інформації.
    """
    data = preprocessor.get_unique_entries(path=path)
    return preprocessor.cleanup_process(data=data)


def get_full_vacancies_data(df: pd.DataFrame, salary_predict_col_name: str = "salary_predict") -> pd.DataFrame:
    """
        Отримує дані про вакансії та повертає DataFrame зі заповненими прогнозованими зарплатами.

        Функція приймає DataFrame `df`, який містить дані про вакансії, та додатковий параметр
        `salary_predict_col_name`, який визначає назву стовпця, в якому будуть збережені прогнозовані зарплати.
        Спочатку функція додає порожній стовпець з вказаною назвою до вихідного DataFrame. Потім вона відбирає
        рядки зі значенням `salary` рівним порожньому рядку і створює новий DataFrame `to_predict_df` з цими
        рядками. На основі `to_predict_df` функція виконує прогнозування зарплат за допомогою
        `salary_predictor.predict` та присвоює отримані прогнози відповідним рядкам у стовпці
        `salary_predict_col_name` в початковому DataFrame. Нарешті, функція замінює порожні значення у
        вихідному DataFrame на `None`.

        Args:
            df (pd.DataFrame): DataFrame з даними про вакансії.
            salary_predict_col_name (str, optional): Назва стовпця для збереження прогнозованих зарплат.
                За замовчуванням - "salary_predict".

        Returns:
            pd.DataFrame: DataFrame з заповненими прогнозованими зарплатами.
    """
    df[salary_predict_col_name] = ""
    to_predict_df = df.where(df["salary"] == "").dropna()
    df.loc[df["salary"] == "", salary_predict_col_name] = salary_predictor.predict(df=to_predict_df)
    return df.replace("", None)
```

Лістинг файлу preprocessor/__init__.py

```
import os
import re
import json

import langdetect
import textdistance
import pandas as pd

# Обмеження на довжину тексту, яку буде використано для визначення мови
TEXT_LENGTH_FOR_LANGUAGE_DETECTION = 128

# Регулярний вираз для виокремлення текстових даних
TEXT_RE = r"[a-za-яєі0-9]+"
```

```
def compare_title_and_query(title: str, query: str) -> float:
    """
        Порівнює заголовок та запит і повертає значення подібності.

        Функція порівнює текст `title` (заголовок) та текст `query` (запит) і повертає значення
        подібності між ними. Спочатку обидва тексти перетворюються на нижній регістр та вилучаються
        зайві символи за допомогою регулярного виразу `TEXT_RE`. Далі кожне слово з `query` порівнюється
        з кожним словом з `title` за допомогою косинусної відстані. За кожну пару слів обчислюється
        косинусна відстань, і зберігається максимальне значення для кожного слова з `query`.
        Загальний бал розраховується як сума максимальних значень для кожного слова з `query`.
        На кінець бал ділиться на кількість слів у `query` для нормалізації значення подібності.

        Args:
            title (str): Заголовок тексту.
            query (str): Запит для порівняння з заголовком.

        Returns:
            float: Значення подібності між заголовком і запитом.
    """
    title_clean = re.findall(TEXT_RE, title.lower())
    query_clean = re.findall(TEXT_RE, query.lower())
    score = 0
    for query_word in query_clean:
        curr_score = 0
        for title_word in title_clean:
            cosine_dist = textdistance.cosine(query_word, title_word)
            curr_score = cosine_dist if cosine_dist > curr_score else curr_score
        score += curr_score
    score /= len(query_clean)
    return score


def check_if_have_any_files_inside(path: str) -> bool:
    """
        Перевіряє, чи є вказана директорія містить будь-які файли.

        Функція рекурсивно перебирає всі файли у вказаній директорії `path` та її піддиректоріях
        за допомогою функції `os.walk()`. Якщо знайдено хоча б один файл, функція повертає `True`.
        Якщо в директорії немає жодного файлу, функція повертає `False`.

        Args:
            path (str): Шлях до директорії, яку потрібно перевірити.

        Returns:
            bool: Прапорець, що показує, чи є в директорії будь-які файли.
    """
    for _, _, files in os.walk(path):
        if len(files):
            return True
        return False


def check_if_file_exist(path: str, create_it: bool = False) -> bool:
    """
        Перевіряє наявність файлу за вказаним шляхом і, за необхідності, створює його.

        Функція перевіряє, чи існує файл з вказаним шляхом `path`. Якщо файл існує,
        функція повертає `True`. Якщо файл не існує, функція може створити його,
        якщо аргумент `create_it` встановлено на `True`. Якщо `create_it` встановлено
        на `False` (за замовчуванням), функція не створює файл.

        Args:
            path (str): Шлях до файлу, який потрібно перевірити або створити.
            create_it (bool, optional): Прапорець, що показує, чи створювати файл, якщо він не існує.
                За замовчуванням встановлено `False`.

        Returns:
            bool: Прапорець, що показує, чи існує файл.
    """
    try:
        open(path, 'x').close()
    except FileExistsError:
        return True
    if not create_it:
        os.remove(path)
    return False
```

```
def delete_all_files_from_folder(path: str):
    """
        Видаляє всі файли з вказаної директорії.

        Функція перебирає всі файли в директорії, вказаній шляхом `path`, і видаляє їх за допомогою
        функції `os.remove()`. Всі файли в директорії будуть безповоротно видалені.

        Args:
            path (str): Шлях до директорії, з якої потрібно видалити файли.
    """
    for file in os.listdir(path):
        os.remove(os.path.join(path, file))


def get_uniqueness_key(data: dict) -> str:
    """
        Генерує ключ унікальності на основі даних про заголовок і компанію.

        Функція приймає словник `data`, який містить дані про заголовок (`title`) та компанію (`company`).
        Використовуючи ці дані, функція генерує ключ унікальності у форматі "заголовок | компанія",
        де `|` є роздільником між заголовком і компанією.

        Args:
            data (dict): Словник з даними, що містять заголовок і компанію.

        Returns:
            str: Ключ унікальності, згенерований на основі даних про заголовок і компанію.
        """
    return f"{data['title']} | {data['company']}"


def get_unique_entries(path: str) -> list:
    """
        Отримує унікальні записи з даних, збережених у файлах всередині вказаної директорії.

        Функція перебирає файли всередині директорії, вказаної шляхом `path`, та отримує дані з кожного файлу.
        За допомогою функції `get_uniqueness_key()` генерується ключ унікальності для кожного запису.
        Записи з унікальними ключами додаються до результату `unique_data`.

        Args:
            path (str): Шлях до директорії, з якої потрібно отримати унікальні записи.

        Returns:
            list: Список унікальних записів з даних.
    """
    unique_register = []
    unique_data = []
    unique_url = []

    # Проходимось за всіма доступними файлами всередині `path`
    for root, _, files in os.walk(path):
        with open(os.path.join(root, files[0]), mode="r", encoding="utf-8") as f:
            data = json.load(fp=f)["jobs"]

        for el in data:
            unique_data.append(el)
            u_key = get_uniqueness_key(data=el)
            unique_register.append(u_key)
            unique_url.append(el['url'])

        for file in files[1:]:
            with open(os.path.join(root, file), mode="r", encoding="utf-8") as f:
                data = json.load(fp=f)["jobs"]

            for el in data:
                u_key = get_uniqueness_key(data=el)
                u_url = el['url']
                if u_key not in unique_register and u_url not in unique_url:
                    unique_data.append(el)
                    unique_register.append(u_key)
                    unique_url.append(u_url)
    return unique_data


def salary_cleanup(data: str) -> str:
    """
        Очищує рядок з даними про заробітну плату.
```

```
        Функція приймає рядок `data`, який містить інформацію про заробітну плату.
        Перевіряє, чи рядок починається з цифри.
        Якщо так, то виконує очищення рядка шляхом заміни символу знаку заробітної плати на "грн" (гривні),
        розділяє рядок за символом "грн" і повертає першу частину розділеного рядка після видалення пробілів,
        якщо розділено на дві частини.
        Якщо рядок не починається з цифри, повертає порожній рядок.

        Args:
            data (str): Рядок з даними про заробітну плату.

        Returns:
            str: Очищений рядок з даними про заробітну плату, або порожній рядок.
    """
    salary_sign = "₴"
    if re.match("[0-9]", data):
        data = data.replace(salary_sign, "грн")
        data = data.split("грн")
        return data[0].replace(" ", "") if len(data) >= 2 else ""
    else:
        return ""


def cleanup_process(data: list, target_column: str = "salary") -> pd.DataFrame:
    """
        Очищує дані та виконує фільтрацію у DataFrame.

        Функція приймає список `data` і назву цільового стовпця `target_column` (за замовчуванням - "salary").
        Завантажує список даних у DataFrame.
        Виконує заміну відповідних символів на пробіли.
        Обирає значення в стовпці `target_column`, які містять числові дані, та застосовує до них функцію
        `salary_cleanup`.
        Видаляє записи з пустим полем `description`.
        Обирає записи, які мають опис українською мовою шляхом використання модуля `langdetect`.
        Повертає очищений DataFrame.

        Args:
            data (list): Список даних.
            target_column (str, optional): Назва цільового стовпця. За замовчуванням - "salary".

        Returns:
            pd.DataFrame: Очищений DataFrame.
    """
    # Завантажуємо датасет в DataFrame
    df = pd.DataFrame(data)

    # Замінюємо відповідні символи на пробіл
    for space_el in [u"\u202f", u"\u2009", u"\xa0", "\n"]:
        df = df.applymap(lambda x: x.replace(space_el, " "))

    # Обираємо значення де `target_column` містить якісь числові дані
    df[target_column] = df[target_column].apply(lambda x: salary_cleanup(x))

    # Видаляємо записи з пустим полем `description`
    df = df.where(df['description'] != "").dropna().reset_index(drop=True)

    # Обираємо записи, що мають опис українською
    lang_df = df['description'].apply(lambda x: langdetect.detect(x[:TEXT_LENGTH_FOR_LANGUAGE_DETECTION]))
    df = df.where(lang_df == 'uk').dropna().reset_index(drop=True)
    return df
```

Лістинг файлу salary_predictor/__init__.py

```
import os
import pandas as pd
import numpy as np
import tensorflow as tf

from salary_predictor import utils

# Шлях для збереження попередньо навченої моделі
TRAINED_MODEL_PATH = 'models/salary_predictor'

# Шлях до файлу із попередньо визначеними замінами значень
REPLACEMENT_PATH = 'dataset/replacement.json'
```

```
# Шлях до файлу з декодуваннями значень класів ЗП
SALARY_DECODE_PATH = 'dataset/salary_decode.json'


def get_tf_dataset(cat: np.ndarray, text: np.ndarray, batch_size: int = 64):
    """
        Створює TensorFlow-датасет з категорій та текстових даних.

        Функція приймає масиви `cat` і `text` та створює TensorFlow-датасет з цих даних.
        Датасет створюється шляхом розбиття масивів `cat` і `text` на пари, де кожна пара складається з
        елементів з відповідних індексів масивів, та нулевого масиву, що використовується для відповідей.
        Результатний датасет розбивається на пакети розміром `batch_size`.

        Args:
            cat (np.ndarray): Масив категорій.
            text (np.ndarray): Масив текстових даних.
            batch_size (int, optional): Розмір пакету. За замовчуванням - 64.

        Returns:
            tf.data.Dataset: TensorFlow-датасет.
    """
    ds = tf.data.Dataset.from_tensor_slices(((cat, text), np.zeros(text.shape)))
    return ds.batch(batch_size)


def predict(df: pd.DataFrame):
    """
        Здійснює передбачення на основі моделі для наданого DataFrame.

        Функція приймає DataFrame `df` з вхідними даними і здійснює передбачення на основі попередньо навченої моделі.
        Дані з DataFrame перетворюються до потрібного формату та подаються на вхід моделі.
        Результат передбачення повертається у вигляді списку.

        Args:
            df (pd.DataFrame): Вхідний DataFrame з даними.

        Returns:
            list: Результат передбачення.
    """
    # Перетворюємо дані до потрібної репрезентації
    cat = utils.get_categorical_representation(data=df, replacement_path=REPLACEMENT_PATH)
    text = utils.get_text_representation(data=df)

    # Створюємо датасет, що буде використано для передбачення
    x = get_tf_dataset(cat=cat, text=text)

    # Завантажуємо попередньо навчену модель
    dir_name = os.path.dirname(__file__)
    model = tf.keras.models.load_model(os.path.join(dir_name, TRAINED_MODEL_PATH))

    # Розпочинаємо процес прогнозування
    classes = utils.get_target_classes(salary_decode_path=SALARY_DECODE_PATH)
    guess = np.argmax(model.predict(x, verbose=0), axis=-1)
    return [classes[x] for x in guess]
```

Лістинг файлу salary_predictor/plots.py

```
import json
import numpy as np
import pandas as pd

from matplotlib import pyplot as plt
import utils

# Шлях до відповідного обробленого датасету
DATASET_PATH = 'dataset/dataset.json'

# Шлях до файлу із попередньо визначеними замінами значень
REPLACEMENT_PATH = 'dataset/replacement.json'

# Шлях до файлу з декодуваннями значень класів ЗП
SALARY_DECODE_PATH = 'dataset/salary_decode.json'

# Шлях до папки де буде збережено результати створення графіків
PLOTS_FOLDER_PATH = "plots"
```

```
# Розмір кожного графіку
FIG_SIZE = (8, 4)


def draw_salary_non_empty_pie():
    """
        Створює діаграму для порівняння вакансій з та без зазначеної зарплати.

        Функція завантажує дані з набору даних і створює діаграму для порівняння кількості
        вакансій з зазначеною зарплатою та без неї. Результат діаграми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])

    all_len = len(df)
    non_empty_len = np.sum((df['salary'] != "").values)

    plt.figure(figsize=FIG_SIZE)
    plt.title("Comparison of vacancies with and without salary")
    plt.pie(
        x=[all_len - non_empty_len, non_empty_len],
        labels=["without salary", "with salary"],
        autopct="%1.1f%%",
        textprops={'size': 14}
    )
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/w_and_wo_salary_comparison.png")


def draw_region_distribution():
    """
        Створює діаграму розподілу вакансій за регіонами з зазначеною зарплатою.

        Функція завантажує дані з набору даних та обирає вакансії з зазначеною зарплатою.
        Далі обчислює розподіл вакансій за регіонами та створює діаграму стовпчиків для відображення цього розподілу.
        Результат діаграми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])
        df = df.where(df["salary"] != "").dropna().reset_index(drop=True)

    data = df["region"].value_counts()
    data = data.where(data > 0.1 * len(data)).dropna()

    plt.figure(figsize=FIG_SIZE)
    plt.title("Region distribution of vacancies with salary (<10% truncate)")
    plt.barh(data.index, data.values)
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/region_distribution_w_salary.png")


def draw_region_distribution_normalized():
    """
        Створює нормалізовану діаграму розподілу вакансій за регіонами з зазначеною зарплатою.

        Функція завантажує дані з набору даних та обирає вакансії з зазначеною зарплатою.
        Далі обчислює розподіл вакансій за регіонами, групуючи деякі регіони разом.
        Створюється діаграма стовпчиків для відображення нормалізованого розподілу.
        Результат діаграми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])
        df = df.where(df["salary"] != "").dropna().reset_index(drop=True)

    chosen_city_value = ["", "київ"]
    other_label_value = "Інші"
    data = df["region"].apply(lambda x: x if x.lower() in chosen_city_value else other_label_value)
    data = data.value_counts()

    plt.figure(figsize=FIG_SIZE)
    plt.title("Normalized region distribution of vacancies with salary")
    plt.barh(data.index, data.values)
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/region_distribution_normalized_w_salary.png")


def draw_remote_type_distribution():
```

```
    """
        Створює діаграму розподілу типів віддаленої роботи для вакансій з зазначеною зарплатою.

        Функція завантажує дані з набору даних та обирає вакансії з зазначеною зарплатою.
        Далі обчислює розподіл типів віддаленої роботи для цих вакансій.
        Створюється діаграма стовпчиків для відображення розподілу типів віддаленої роботи.
        Результат діаграми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])
        df = df.where(df["salary"] != "").dropna().reset_index(drop=True)

    data = df["remote_type"].value_counts()

    plt.figure(figsize=FIG_SIZE)
    plt.title("Remote type distribution of vacancies with salary")
    plt.barh(data.index, data.values)
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/remote_type_distribution_w_salary.png")


def draw_employment_type_distribution():
    """
        Створює діаграму розподілу типів зайнятості для вакансій з зазначеною зарплатою.

        Функція завантажує дані з набору даних та обирає вакансії з зазначеною зарплатою.
        Далі обчислює розподіл типів зайнятості для цих вакансій.
        Створюється діаграма стовпчиків для відображення розподілу типів зайнятості.
        Результат діаграми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])
        df = df.where(df["salary"] != "").dropna().reset_index(drop=True)

    data = df["employment_type"].value_counts()

    plt.figure(figsize=FIG_SIZE)
    plt.title("Employment type distribution of vacancies with salary")
    plt.barh(data.index, data.values)
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/employment_type_distribution_w_salary.png")


def draw_seniority_distribution():
    """
        Створює діаграму розподілу рівня кваліфікації для вакансій з зазначеною зарплатою.

        Функція завантажує дані з набору даних та обирає вакансії з зазначеною зарплатою.
        Далі обчислює розподіл рівня кваліфікації для цих вакансій.
        Створюється діаграма стовпчиків для відображення розподілу рівня кваліфікації.
        Результат діаграми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])
        df = df.where(df["salary"] != "").dropna().reset_index(drop=True)

    data = df["seniority"].value_counts()

    plt.figure(figsize=FIG_SIZE)
    plt.title("Seniority distribution of vacancies with salary")
    plt.barh(data.index, data.values)
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/seniority_distribution_w_salary.png")


def draw_salary_distribution():
    """
        Створює гістограму розподілу зарплат для вакансій з зазначеною зарплатою.

        Функція завантажує дані з набору даних та обирає вакансії з зазначеною зарплатою.
        Далі обчислює розподіл зарплат для цих вакансій.
        Створюється гістограма для відображення розподілу зарплат.
        Результат гістограми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])
        df = df.where(df["salary"] != "").dropna().reset_index(drop=True)
```

```
    data = utils.get_target_data(data=df)

    plt.figure(figsize=FIG_SIZE)
    plt.title("Salary distribution for vacancies")
    plt.hist(data, bins=50)
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/salary_distribution.png")


def draw_salary_distribution_normalized():
    """
        Створює нормалізовану гістограму розподілу зарплат для вакансій з зазначеною зарплатою.

        Функція завантажує дані з набору даних та обирає вакансії з зазначеною зарплатою.
        Далі обчислює нормалізований розподіл зарплат для цих вакансій.
        Створюється гістограма для відображення нормалізованого розподілу зарплат.
        Результат гістограми зберігається у файлі.
    """
    with open(DATASET_PATH, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)["jobs"])
        df = df.where(df["salary"] != "").dropna().reset_index(drop=True)

    data = utils.get_target_representation(data=df, salary_decode_path=SALARY_DECODE_PATH)
    data = pd.Series(np.argmax(data, axis=1)).value_counts().sort_index()
    classes = list(utils.get_target_classes(salary_decode_path=SALARY_DECODE_PATH).values())

    plt.figure(figsize=FIG_SIZE)
    plt.title("Normalized salary distribution for vacancies")
    plt.bar(classes, data.values)
    plt.tight_layout()
    plt.savefig(f"{PLOTS_FOLDER_PATH}/salary_distribution_normalized.png")


def main():
    draw_salary_non_empty_pie()
    draw_region_distribution()
    draw_region_distribution_normalized()
    draw_remote_type_distribution()
    draw_employment_type_distribution()
    draw_seniority_distribution()
    draw_salary_distribution()
    draw_salary_distribution_normalized()


if __name__ == '__main__':
    main()
```

Лістинг файлу salary_predictor/preprocessing.py

```
import json

import preprocessor

# Шлях до папки з необробленими даними
RAW_DATASETS_FOLDER = 'dataset_raw'

# Шлях до результуючого файлу з обробленими даними
PREPROCESSED_DATASET_PATH = 'dataset/dataset.json'


def create_preprocessed_dataset():
    """
        Створює попередньо оброблений набір даних.

        Функція отримує унікальні записи з вихідної теки даних.
        Далі проводить обробку даних з використанням попередньо заданої процедури очищення.
        Попередньо оброблений набір даних зберігається у форматі JSON.
    """
    # Отримання унікальних записів
    data = preprocessor.get_unique_entries(path=RAW_DATASETS_FOLDER)

    # Обробка даних
    df = preprocessor.cleanup_process(data=data)

    # Перемішування набору даних
```

```
    df = df.sample(frac=1).reset_index(drop=True)

    # Збереження попередньо обробленого датасету
    output_data = df.to_dict(orient="records")
    with open(PREPROCESSED_DATASET_PATH, mode="w", encoding="utf-8") as f:
        json.dump(obj={"jobs": output_data}, fp=f, ensure_ascii=False, indent=4)


def main():
    print('Starting...')
    print(f'Checking if file `{PREPROCESSED_DATASET_PATH}` exist...')

    if preprocessor.check_if_file_exist(path=PREPROCESSED_DATASET_PATH):
        print(f'File `{PREPROCESSED_DATASET_PATH}` already exist!\n'
              f'Successful finish. No need to create anything')

    else:
        print(f'File `{PREPROCESSED_DATASET_PATH}` doesnt exist yet!\n'
              f'Initializing creation of dataset file...\n'
              f'Checking for any files inside `{RAW_DATASETS_FOLDER}` folder')

        if preprocessor.check_if_have_any_files_inside(path=RAW_DATASETS_FOLDER):
            print('Creating preprocessed dataset...')
            create_preprocessed_dataset()
            print(f'Done! Result saved in `{PREPROCESSED_DATASET_PATH}`\n'
                  f'Successful finish. Everything needed was created')

        else:
            print(f'No files found in `{RAW_DATASETS_FOLDER}` folder!\n'
                  f'Unsuccessful finish. Add files to `{RAW_DATASETS_FOLDER}` folder!')


if __name__ == '__main__':
    main()
```

Лістинг файлу salary_predictor/train.py

```
import json
import numpy as np
import pandas as pd
import tensorflow as tf
from matplotlib import pyplot as plt

import utils

# Шлях до попередньо обробленого датасету
DATASET_PATH = 'dataset/dataset.json'

# Шлях до файлу із попередньо визначеними замінами значень
REPLACEMENT_PATH = 'dataset/replacement.json'

# Шлях до файлу з декодуваннями значень класів ЗП
SALARY_DECODE_PATH = 'dataset/salary_decode.json'

# Шлях для збереження навченої моделі
TRAINED_MODEL_PATH = 'models/salary_predictor'

# Шлях для збереження графіків
TRAINING_PLOT_PATH = 'plots'

CAT_INPUT_SIZE = 16
TEXT_INPUT_SIZE = 1
SALARY_OUTPUT_SIZE = 5

MAX_TOKENS = 10000
WORD_VECTOR_SIZE = 64
OUTPUT_SEQUENCE_LENGTH = 256

EPOCHS = 35
BATCH_SIZE = 64

LEARNING_RATE = 1e-4


def load_dataset(path: str) -> pd.DataFrame:
```

```
    """
        Завантажує набір даних з вказаного шляху.

        Функція завантажує набір даних з вказаного шляху до формату DataFrame.
        Виконується фільтрація записів, залишаються тільки ті, що мають непусте поле 'salary'.

        Parameters:
            path (str): Шлях до файлу з набором даних.

        Returns:
            pd.DataFrame: Завантажений набір даних у форматі DataFrame.
    """
    with open(path, mode="r", encoding="utf-8") as f:
        df = pd.DataFrame(json.load(fp=f)['jobs'])
        df = df.where(df['salary'] != "").dropna().reset_index(drop=True)
    return df


def make_tf_dataset(cat: np.ndarray, text: np.ndarray,
                    target: np.ndarray = None, batch_size: int = 64) -> tf.data.Dataset:
    """
        Створює TensorFlow-датасет з вхідних даних.

        Функція створює TensorFlow-датасет з категоріальних і текстових вхідних даних
        та, опціонально, цільових даних. Датасет розбивається на пакети розміром `batch_size`.

        Parameters:
            cat (np.ndarray): Масив категоріальних вхідних даних.
            text (np.ndarray): Масив текстових вхідних даних.
            target (np.ndarray, optional): Масив цільових даних. За замовчуванням - None.
            batch_size (int, optional): Розмір пакету. За замовчуванням - 64.

        Returns:
            tf.data.Dataset: TensorFlow-датасет з вхідними і цільовими даними.
    """
    if target is None:
        target = np.zeros(text.shape)
    ds = tf.data.Dataset.from_tensor_slices(((cat, text), target))
    return ds.batch(batch_size)


def create_dataset(path: str) -> tuple[tf.data.Dataset, tf.data.Dataset]:
    """
        Створює тренувальний та тестовий TensorFlow-датасети з набору даних.

        Функція створює тренувальний та тестовий TensorFlow-датасети з набору даних,
        розділеного за вказаною шляхом. Дані розділяються у співвідношенні `train_data_split`,
        а потім перетворюються у відповідні репрезентації для використання у моделі.

        Parameters:
            path (str): Шлях до набору даних.

        Returns:
            Tuple[tf.data.Dataset, tf.data.Dataset]: Кортеж з тренувальним та тестовим TensorFlow-датасетами.
    """
    train_data_split = 0.9
    df = load_dataset(path=path)

    # Створення необхідних замін значень на основі датасету, та зберігання їх у відповідний файл
    utils.create_categorical_representation(data=df, replacement_path=REPLACEMENT_PATH)

    # Отримання необхідних репрезентації з датасету
    cat = utils.get_categorical_representation(data=df, replacement_path=REPLACEMENT_PATH)
    text = utils.get_text_representation(data=df)
    target = utils.get_target_representation(data=df, salary_decode_path=SALARY_DECODE_PATH)

    # Процес створення TensorFlow датасету
    idx = int(train_data_split * target.shape[0])
    train_ds = make_tf_dataset(cat=cat[:idx], text=text[:idx], target=target[:idx], batch_size=BATCH_SIZE)
    test_ds = make_tf_dataset(cat=cat[idx:], text=text[idx:], target=target[idx:], batch_size=BATCH_SIZE)
    return train_ds, test_ds


def create_model(vectorization_layer: tf.keras.layers.Layer) -> tf.keras.Sequential:
    """
        Створює модель для передбачення заробітної плати на основі категоріальних та текстових ознак.

        Функція створює модель для передбачення заробітної плати на основі категоріальних та текстових ознак.
```

```
    Категоріальна частина моделі складається зі шарів Dense, а текстова частина використовує векторизацію,
    згодом застосовується шар Embedding та звичайні 1D-згортки та пулінг для обробки текстових даних.
    Обидві частини моделі об'єднуються, і на виході отримуємо прогнози заробітної плати.

    Parameters:
        vectorization_layer (tf.keras.layers.Layer): Шар векторизації тексту.

    Returns:
        tf.keras.Sequential: Створена модель для передбачення заробітної плати.
    """
    # Структура категоріальної частини моделі
    cat_model = tf.keras.Sequential()
    cat_model.add(tf.keras.layers.Input(shape=(CAT_INPUT_SIZE,)))
    cat_model.add(tf.keras.layers.Dense(64, activation='relu'))
    cat_model.add(tf.keras.layers.Dense(32, activation='relu'))

    # Структура текстової частини моделі
    text_model = tf.keras.Sequential()
    text_model.add(tf.keras.layers.Input(shape=(TEXT_INPUT_SIZE,), dtype='string'))
    text_model.add(vectorization_layer)
    text_model.add(tf.keras.layers.Embedding(input_dim=MAX_TOKENS, output_dim=WORD_VECTOR_SIZE,
                                             input_length=OUTPUT_SEQUENCE_LENGTH, mask_zero=True))
    text_model.add(tf.keras.layers.Conv1D(filters=32, kernel_size=3, activation="relu"))
    text_model.add(tf.keras.layers.MaxPooling1D())
    text_model.add(tf.keras.layers.GlobalAveragePooling1D())
    text_model.add(tf.keras.layers.Dense(32, activation='relu'))

    # Точка об'єднання двох частин моделі
    x = tf.keras.layers.Concatenate()([text_model.output, cat_model.output])
    x = tf.keras.layers.Dense(32, activation="relu")(x)
    output = tf.keras.layers.Dense(SALARY_OUTPUT_SIZE, activation="softmax")(x)
    model = tf.keras.Model(inputs=[cat_model.input, text_model.input], outputs=[output])
    return model


def draw_and_save_metric(model: tf.keras.Sequential, path: str, key: str, name: str = None):
    """
    Малює та зберігає графік метрики моделі.

    Функція малює графік значень вказаної метрики для тренування та валідації моделі,
    а також зберігає графік у файлі з вказаним шляхом.

    Parameters:
        model (tf.keras.Sequential): Модель, для якої треба намалювати графік метрики.
        path (str): Шлях для збереження графіку.
        key (str): Ключ метрики для візуалізації.
        name (str, optional): Назва метрики, яка буде відображена у заголовку графіку.
            Якщо не вказано, використовується значення ключа.
    """
    if name is None:
        name = key
    plt.figure(figsize=(8, 6))
    plt.title(f"Training {name} values")
    plt.plot(model.history.history[key], label=f"Training {name}")
    plt.plot(model.history.history[f"val_{key}"], label=f"Validation {name}")
    plt.ylabel("Value")
    plt.xlabel("Epoch")
    plt.legend()
    plt.tight_layout()
    plt.savefig(f'{path}/{key}.png')


def train(train_ds: tf.data.Dataset, test_ds: tf.data.Dataset, batch_size: int, epochs: int):
    """
    Навчання моделі на тренувальному датасеті та оцінка на тестовому датасеті.

    Функція навчає модель на тренувальному датасеті і оцінює її на тестовому датасеті
    протягом заданої кількості епох. Також функція зберігає графіки метрик навчання та
    зберігає навчену модель.

    Parameters:
        train_ds (tf.data.Dataset): Тренувальний датасет.
        test_ds (tf.data.Dataset): Тестовий датасет.
        batch_size (int): Розмір пакета для навчання.
        epochs (int): Кількість епох навчання.
    """
    # Ініціалізація шару векторизації для текстової частини даних
    vectorization_layer = tf.keras.layers.TextVectorization(
```

```
        max_tokens=MAX_TOKENS, output_sequence_length=OUTPUT_SEQUENCE_LENGTH)
    vocab_text = np.hstack([x.numpy() for (_, x), _ in train_ds])
    vectorization_layer.adapt(vocab_text, batch_size=batch_size)

    # Створення композитної моделі
    model = create_model(vectorization_layer=vectorization_layer)

    # Вибір оптимізатора та функції втрат
    optimizer = tf.keras.optimizers.Adam(learning_rate=LEARNING_RATE)
    loss = tf.keras.losses.MeanSquaredError()

    # Компіляція композитної моделі з обраними параметрами
    model.compile(loss=loss, optimizer=optimizer, metrics=["accuracy"])

    # Запуск процесу навчання
    model.fit(x=train_ds, epochs=epochs, validation_data=test_ds)

    # Збереження графіків відповідних метрик
    draw_and_save_metric(model=model, path=TRAINING_PLOT_PATH, key='loss', name='Loss')
    draw_and_save_metric(model=model, path=TRAINING_PLOT_PATH, key='accuracy', name='Accuracy')

    # Збереження навченої моделі до `TRAINED_MODEL_PATH`
    model.save(TRAINED_MODEL_PATH)


def main():
    # Завантаження відповідного датасету
    train_ds, test_ds = create_dataset(path=DATASET_PATH)

    # Запуск процесу навчання
    train(train_ds=train_ds, test_ds=test_ds, batch_size=BATCH_SIZE, epochs=EPOCHS)


if __name__ == '__main__':
    main()
```

Лістинг файлу salary_predictor/translate.py

```
import json
import time

from deep_translator import GoogleTranslator
import langdetect

# Словник зі шляхами до необроблених датасетів, що необхідно перекласти
DATASET_PATHS = (
    "dataset_raw/djinni.co_15-04.json",
    "dataset_raw/rabota.ua_14-04.json",
    "dataset_raw/rabota.ua_21-11.json",
    "dataset_raw/work.ua_13-04.json",
    "dataset_raw/work.ua_24-11.json"
)


def translation_process(path: str):
    """
        Процес перекладу текстових полів у датасеті.

        Функція здійснює процес перекладу текстових полів у датасеті
        з однієї мови на українську мову. Використовується мовний визначник
        для виявлення мови тексту та Google Translator для перекладу.

        Parameters:
            path (str): Шлях до файлу датасету JSON.
    """
    translator = GoogleTranslator(source='auto', target='uk')

    with open(path, mode='r', encoding='utf-8') as f:
        data = json.load(fp=f)['jobs']

    length = len(data)
    print(f"\nWorking on: {path}\n"
          f"0 / {length}")

    for i in range(length):
```

```
            original_description = data[i]["description"]
            if original_description == "":
                continue

            if langdetect.detect(original_description) != "uk":
                try:
                    data[i]["description"] = translator.translate(original_description)
                except Exception as e:
                    print(f"!!! SKIP !!! \n{e}")
                time.sleep(0.1)

            original_additional_info = data[i]["additional_info"]
            if original_additional_info == "":
                continue

            if langdetect.detect(original_additional_info) != "uk":
                try:
                    data[i]["additional_info"] = translator.translate(original_additional_info)
                except Exception as e:
                    print(f"!!! SKIP !!! \n{e}")
                time.sleep(0.1)

            if (i + 1) % 100 == 0:
                print(f"{i + 1} / {length}")

        print(f"{length} / {length}")

        save_path = ".".join(path.split(".")[:-1])
        with open(f"{save_path}_translated.json", mode='w', encoding='utf-8') as f:
            json.dump(obj={"jobs": data}, fp=f, ensure_ascii=False, indent=4)


    def main():
        # Запуск процесу переведення кожно файлу окремо
        for path in DATASET_PATHS:
            translation_process(path=path)

        print("\nFINISHED!")


    if __name__ == '__main__':
        main()
```

Лістинг файлу salary_predictor/utils.py

```
import os
import re
import json
import pandas as pd
import numpy as np

# Регулярний вираз для отримання текстових даних
TEXT_RE = r"[a-zA-Zа-яА-Яєїґ-Ї0-9]+"


def text_cleanup(text: str) -> str:
    """
        Очищення тексту від зайвих символів та форматування.

        Функція здійснює очищення текстового рядка від зайвих символів,
        зберігає лише букви, цифри та пробіли. Також перетворює текст у
        нижній регістр.

        Parameters:
            text (str): Вхідний текст для очищення.

        Returns:
            str: Очищений текст.
    """
    return " ".join(re.findall(TEXT_RE, text)).lower()


def create_categorical_representation(data: pd.DataFrame, replacement_path: str):
    """
        Створення категоріальної репрезентації для даних.
```

```
    Функція здійснює обробку категоріальних даних в DataFrame, проводить
    заміну значень на числові еквіваленти і зберігає цю заміну до json файлу.

    Parameters:
        data (pd.DataFrame): Вхідний DataFrame з даними.
        replacement_path (str): Шлях для збереження файлу з заміною значень.
    """
    cat_columns = ['region', 'remote_type', 'employment_type', 'seniority']
    df = data[cat_columns].applymap(lambda x: text_cleanup(text=x)).replace('', None)

    # Маніпуляція з регіонами
    chosen_city = ["київ"]
    df['region'] = df['region'].apply(lambda x: x if x in chosen_city else None)

    # Проведення заміни для категоріальних даних
    replacement = {}
    for col in cat_columns:
        values = df[col].value_counts()
        replacement[col] = dict(zip(values.index, range(1, len(values) + 1)))

    # Збереження визначеної заміни значень до json файлу
    dir_name = os.path.dirname(__file__)
    with open(os.path.join(dir_name, replacement_path), mode='w+', encoding='utf-8') as f:
        json.dump(replacement, fp=f, ensure_ascii=False, indent=4)


def get_categorical_representation(data: pd.DataFrame, replacement_path: str) -> np.ndarray:
    """
    Отримання категоріальної репрезентації для даних.

    Функція застосовує заміну значень на числові еквіваленти на основі файлу з
    заміною і повертає матрицю категоріальної репрезентації даних.

    Parameters:
        data (pd.DataFrame): Вхідний DataFrame з даними.
        replacement_path (str): Шлях до файлу з заміною значень.

    Returns:
        np.ndarray: Матриця категоріальної репрезентації даних.
    """
    # Завантаження файлу із замінами значень
    try:
        dir_name = os.path.dirname(__file__)
        with open(os.path.join(dir_name, replacement_path), mode='r', encoding='utf-8') as f:
            replacement = json.load(fp=f)
    except FileNotFoundError:
        print(f'File `{replacement_path}` is not found!\n'
              f'Add it to be able to start prediction process! Exiting...')
        exit(-1)

    # Отримання необхідних колонок з датасету
    cat_columns = list(replacement.keys())
    df = data[cat_columns].applymap(lambda x: text_cleanup(text=x)).replace('', None)

    # Застосування відповідної заміни значень
    df = df[cat_columns].replace(replacement)

    # Об'єднання невизначених регіонів
    chosen_city = list(replacement['region'].values())
    new_idx = len(chosen_city) + 1
    df['region'] = df['region'].apply(lambda x: x if x in chosen_city or x is None else new_idx)
    replacement['region']['інші'] = new_idx

    df = df.applymap(lambda x: x if re.match(r"[0-9]", str(x)) else None)
    df = df.fillna(0).astype('int')

    # Визначення додаткових розширених змінних
    df_dim = len(df)
    rep_lengths = [len(x) for x in replacement.values()]
    vector_dim = len(replacement) + sum(rep_lengths)
    res = np.zeros((df_dim, vector_dim))

    # Створення категоріальної репрезентації
    for i in range(df_dim):
        vector_pieces = list()
        for j in range(len(cat_columns)):
            piece = np.zeros(rep_lengths[j] + 1)
```

```
            piece[df.iloc[i][cat_columns[j]]] = 1
            vector_pieces.append(piece)
        res[i] = np.hstack(vector_pieces)
    return res


def get_text_representation(data: pd.DataFrame) -> np.ndarray:
    """
        Отримання текстової репрезентації для даних.

        Функція об'єднує різні текстові стовпці даних у єдиний рядок і повертає
        масив з текстовою репрезентацією для кожного запису.

        Parameters:
            data (pd.DataFrame): Вхідний DataFrame з даними.

        Returns:
            np.ndarray: Масив з текстовою репрезентацією для кожного запису.
    """
    text_columns = ['title', 'company', 'additional_info', 'description']
    df = data[text_columns].applymap(lambda x: text_cleanup(x))
    full_text = 'назва ' + df['title'].map(str) + ' ' + \
                'компанія ' + df['company'].map(str) + ' ' + \
                'інформація ' + df['additional_info'].map(str) + ' ' + \
                'опис ' + df['description'].map(str)
    return full_text.values


def get_target_data(data: pd.DataFrame) -> np.ndarray:
    """
        Отримання цільових даних.

        Функція обробляє стовпець зарплати у вихідному DataFrame і повертає масив з цільовими даними.
        Значення зарплати замінюються на середнє значення чисел, що входять у рядок зарплати.

        Parameters:
            data (pd.DataFrame): Вхідний DataFrame з даними.

        Returns:
            np.ndarray: Масив з цільовими даними.
    """
    result = data['salary'].apply(func=lambda x: x.replace(" ", ""))
    for i in range(len(result)):
        match = re.finditer(pattern="[0-9]+", string=result[i])
        curr_list = [int(x.group()) for x in match]
        result[i] = sum(curr_list) / len(curr_list) if curr_list else -1
    return np.asarray(result.values).astype("float32")


def get_target_classes(salary_decode_path: str) -> dict[int, str]:
    """
        Отримання словника цільових класів.

        Функція завантажує файл з декодуванням зарплатних класів та створює словник з відповідностями
        між числовими класами та текстовими описами зарплатних інтервалів.

        Parameters:
            salary_decode_path (str): Шлях до файлу з декодуванням зарплатних класів.

        Returns:
            dict[int, str]: Словник з відповідностями між числовими класами та текстовими описами зарплатних інтервалів.
    """
    dir_name = os.path.dirname(__file__)
    with open(file=os.path.join(dir_name, salary_decode_path), mode="r", encoding="utf-8") as f:
        salary_decode = json.load(fp=f)

    result = dict()
    for key, (lower_bound, upper_bound) in salary_decode.items():
        current_class = int(key)
        if lower_bound != 0 and upper_bound != -1:
            result[current_class] = f"{lower_bound} - {upper_bound}"
        else:
            if lower_bound == 0:
                result[current_class] = f"< {upper_bound}"
            if upper_bound == -1:
                result[current_class] = f"> {lower_bound}"

    return result
```

```
def get_target_representation(data: pd.DataFrame, salary_decode_path: str) -> np.ndarray:
    """
        Отримання представлення цільових значень у векторному вигляді.

        Функція обчислює векторне представлення цільових значень на основі заданих меж зарплатних класів.

        Parameters:
            data (pd.DataFrame): Дані, що містять цільові значення.
            salary_decode_path (str): Шлях до файлу з декодуванням зарплатних класів.

        Returns:
            np.ndarray: Векторне представлення цільових значень у вигляді numpy-масиву.
    """
    absolute_target = get_target_data(data=data)

    dir_name = os.path.dirname(__file__)
    with open(file=os.path.join(dir_name, salary_decode_path), mode="r", encoding="utf-8") as f:
        salary_decode = json.load(fp=f)

    result = np.zeros((len(absolute_target), len(salary_decode)))
    for i in range(result.shape[0]):
        for j in range(result.shape[1]):
            lower_bound, upper_bound = salary_decode[str(j)]
            if upper_bound == -1:
                if lower_bound < absolute_target[i]:
                    result[i, j] = 1
            else:
                if lower_bound < absolute_target[i] <= upper_bound:
                    result[i, j] = 1
    return result
```

Лістинг файлу scrapers/__init__.py

```
from scrapers import djinni_co
from scrapers import rabota_ua
from scrapers import work_ua


def run_all(save_path: str, max_pages_to_collect: int):
    """
        Запуск усіх скраперів для збору вакансій.

        Функція запускає усі доступні скрапери для збору вакансій з різних джерел і зберігає результат у вказану теку.

        Parameters:
            save_path (str): Шлях до теки, в яку будуть збережені результати.
            max_pages_to_collect (int): Максимальна кількість сторінок, які необхідно зібрати з кожного джерела.
    """
    djinni_co.run(save_path=save_path, max_pages_to_collect=max_pages_to_collect)
    rabota_ua.run(save_path=save_path, max_pages_to_collect=max_pages_to_collect)
    work_ua.run(save_path=save_path, max_pages_to_collect=max_pages_to_collect)
```

Лістинг файлу scrapers/djinni_co.py

```
import os
import requests
import re
import json
import time
import dateparser

from lxml import etree
from bs4 import BeautifulSoup
from datetime import date
from config import config
from typing import Optional

VERBOSE_FLAG = config.SCRAPERS_VERBOSE_FLAG
CONVERSION_MULTIPLIER = 35

TODAY = date.today()
```

```
PAGE_NEXT = 'https://djinni.co/jobs/?region=UKR&page={}'
MAIN_PAGE = 'https://djinni.co/jobs/?region=UKR'
FILE_NAME = f"djinni.co_{TODAY.strftime('%d-%m')}.json"

HEADERS = {
    'User-Agent':
        'Mozilla/5.0 (Windows NT 10.0; Win64; x64) '
        'AppleWebKit/537.36 (KHTML, like Gecko) '
        'Chrome/107.0.0.0 '
        'Safari/537.36'
}


def get_response_with_sleep_time(link, sleep_time=1.0):
    """
        Отримання відповіді на запит з врахуванням часу очікування.

        Функція відправляє запит до вказаного посилання і отримує відповідь з врахуванням часу очікування між запитами.
        Якщо отримано помилку, функція спробує виконати запит з більшим часом очікування.

        Parameters:
            link (str): Посилання, до якого потрібно відправити запит.
            sleep_time (float): Час очікування між запитами (в секундах). За замовчуванням 1.0.

        Returns:
            BeautifulSoup: Об'єкт BeautifulSoup з вмістом відповіді на запит, або порожній рядок,
                якщо не вдалося отримати дані.
    """
    if sleep_time > 300:
        if VERBOSE_FLAG:
            print('Can`t get a data from page')
        return ''
    else:
        time.sleep(1 / 5)  # Get maximum 5 requests per second
        response = requests.get(link, headers=HEADERS)
        if response.status_code != requests.codes.ok:
            time.sleep(sleep_time)
            return get_response_with_sleep_time(link, sleep_time * 2)
        else:
            return BeautifulSoup(response.text, 'html.parser')


def get_pages_links():
    """
        Отримання посилань на всі сторінки.

        Функція відправляє запит до головної сторінки і отримує список посилань на всі сторінки.
        Кількість сторінок визначається шляхом отримання значення останньої сторінки з головної сторінки.

        Returns:
            list: Список посилань на всі сторінки.
    """
    response = requests.get(MAIN_PAGE, headers=HEADERS)
    soup = BeautifulSoup(response.text, 'html.parser')

    total_pages_li_value = "//li[@class='page-item'][last()]/preceding-sibling::li[@class='page-item'][1]/a"

    try:
        total_pages = etree.HTML(str(soup)).xpath(total_pages_li_value)[0].text
        total_pages = int(total_pages)
    except Exception as e:
        if VERBOSE_FLAG:
            print(f'Can`t get count of total pages.\n{e}\nPages set 400')
        total_pages = 400

    pages_links = [PAGE_NEXT.format(page) for page in range(1, total_pages + 1)]
    return pages_links


def get_job_links(page_link):
    """
        Отримання посилань на вакансії на сторінці.

        Функція отримує посилку на сторінку та повертає список посилань на вакансії, знайдені на цій сторінці.

        Args:
            page_link (str): Посилання на сторінку.
```

```
        Returns:
            list: Список посилань на вакансії на сторінці.
    """
    if VERBOSE_FLAG:
        print(f'Page: {page_link}')
    soup = get_response_with_sleep_time(page_link)

    job_link_tags = soup.find_all('a', {'class': 'profile'})
    job_links = []
    for jobLink_tag in job_link_tags:
        job_link = 'https://djinni.co' + jobLink_tag.get('href')
        job_links.append(job_link)

    return job_links


def parse_date(date_raw: str) -> Optional[dateparser.date]:
    """
        Розбір дати з рядка.

        Функція отримує рядок, що містить дату, та повертає об'єкт дати, який отриманий з розбору рядка.

        Args:
            date_raw (str): Рядок, що містить дату.

        Returns:
            dateparser.date: Об'єкт дати, отриманий з розбору рядка.
    """
    return dateparser.parse(date_raw, settings={'TIMEZONE': 'UTC'}, date_formats=["%d %m %Y"])


def parse_remote_type(remote_raw: str) -> str:
    """
        Розбір типу роботи (віддалено, офіс, гібридний) з рядка.

        Функція отримує рядок, що містить інформацію про тип роботи (віддалено, офіс, гібридний) і повертає
        відповідний тип роботи.

        Args:
            remote_raw (str): Рядок, що містить інформацію про тип роботи.

        Returns:
            str: Тип роботи: 'remote' (віддалено), 'in-office' (офіс), 'hybrid' (гібридний). Повертається
                порожній рядок, якщо тип роботи не знайдений.
    """
    if re.findall(r"тільки віддалено", remote_raw, re.IGNORECASE):
        return 'remote'
    elif re.findall(r"тільки офіс", remote_raw, re.IGNORECASE):
        return 'in-office'
    elif re.findall(r"(гібридна робота|office/remote|на ваш вибір)", remote_raw, re.IGNORECASE):
        return 'hybrid'
    else:
        return ""


def parse_region(region_raw: str) -> [str, str]:
    """
        Розбір регіону з рядка.

        Функція отримує рядок, що містить інформацію про регіон, і повертає відокремлені назви міст та країн,
        які входять до цього регіону.

        Args:
            region_raw (str): Рядок, що містить інформацію про регіон.

        Returns:
            Tuple[str, str]: Кортеж, що містить два елементи: назви міст (розділені комами) та назви країн.
                Якщо інформація про міста або країни відсутня, відповідний елемент кортежу буде порожнім рядком.
    """
    raw = re.sub('\\n', ' ', region_raw)

    cities = ''
    cities_groups = re.findall('\\([^)]+\\)', raw)
    if cities_groups:
        for group in cities_groups:
            cities += group + ', '
        cities = re.sub(',\\s*$', '', cities).replace('(', '').replace(')', '').strip()
```

```
    countries = re.sub('\\([^)]+\\)', '', raw).strip()
    return cities, countries


def parse_seniority(title_raw: str) -> str:
    """
        Розбір рівня кваліфікації з рядка.

        Функція отримує рядок, що містить інформацію про рівень кваліфікації, і повертає розібраний рівень.

        Args:
            title_raw (str): Рядок, що містить інформацію про рівень кваліфікації.

        Returns:
            str: Розібраний рівень кваліфікації. Якщо рівень не знайдений, повертається порожній рядок.
    """
    if re.findall(r"(junior|assistant|intern|trainee|entry)", title_raw, re.IGNORECASE):
        return 'entry-level'
    elif re.findall(r"(middle|mid-level|\smid\s)", title_raw, re.IGNORECASE):
        return 'mid-level'
    elif re.findall(r"(senior|sen-level|\ssen\s|president|director|head of)", title_raw, re.IGNORECASE):
        return 'senior-level'
    elif re.findall(r"tech lead", title_raw, re.IGNORECASE):
        return 'tech-lead'
    elif re.findall(r"team lead|lead", title_raw, re.IGNORECASE):
        return 'team-lead'
    else:
        return ""


def parse_salary(salary_raw: str) -> str:
    """Розбір заробітної плати з рядка.

        Функція отримує рядок, що містить інформацію про заробітну плату, і повертає розібрану
        заробітну плату у валюті UAH.

        Args:
            salary_raw (str): Рядок, що містить інформацію про заробітну плату.

        Returns:
            str: Розібрана заробітна плата у валюті UAH. Якщо заробітна плата не знайдена, повертається порожній рядок.
    """
    if salary_raw == "":
        return salary_raw

    salary_raw = salary_raw.replace("₴", "грн")
    uah_match = re.findall("грн", salary_raw)
    if uah_match:
        return salary_raw

    usd_match = re.findall("\\$", salary_raw)
    if usd_match:
        number_matches = re.finditer("[0-9]+", salary_raw)
        pointer = 0
        uah_salary = ""
        for counter, usd_number in enumerate(number_matches):
            if counter >= 2:
                break
            cut_start, cut_end = usd_number.span()
            usd_value = float(usd_number.group())
            uah_value = int(round(CONVERSION_MULTIPLIER * usd_value))
            uah_salary += f"{salary_raw[pointer:cut_start]}{uah_value}"
            pointer = cut_end

        uah_salary = uah_salary.replace("$", "")
        uah_salary += " грн"
        return uah_salary

    return salary_raw


def get_jobs_data(job_link):
    """
        Отримання даних про роботу з посилання.

        Функція отримує посилання на роботу і повертає словник з розібраними даними про роботу.
```

```
        Args:
            job_link (str): Посилання на роботу.

        Returns:
            dict: Словник з даними про роботу. Ключі словника відповідають різним атрибутам роботи, таким як URL,
                назва, опис, компанія, дата оновлення, регіон, країна, тип віддаленості, тип зайнятості,
                заробітна плата, додаткова інформація, рівень кваліфікації та дата збору даних.
    """
    soup_job = get_response_with_sleep_time(job_link)

    description = soup_job.find('div', class_='col-sm-8 row-mobile-order-2')
    description = description.getText(separator=" ", strip=True) if description else ""

    title = soup_job.find('div', class_='detail--title-wrapper')
    title = title.getText(separator=" ", strip=True) if title else ""
    title = re.sub('\\s*\\n.*', '', title)

    seniority = parse_seniority(title)

    company = soup_job.find('a', class_='job-details--title')
    company = company.getText(separator=" ", strip=True) if company else ""

    date_raw = soup_job.find('span', class_='bi bi-pencil-square')
    date_raw = date_raw.nextSibling.getText(separator=" ", strip=True) if date_raw else ""
    date_raw = re.search('\\d.*', date_raw).group(0)
    date_ = parse_date(date_raw).strftime('%d/%m/%Y')

    regions = soup_job.find('span', class_='location-text')

    additional_regions = regions.find('span', {'data-toggle': 'tooltip'}) if regions else ""
    additional_regions = additional_regions.get('title') if additional_regions else ""

    regions_text = regions.getText(separator=" ", strip=True) if regions else ""
    region_raw = re.sub('\\s*\\+ ще \\d+ міст(а)?', ', ' + additional_regions, regions_text)
    region, _ = parse_region(region_raw)
    region = region.split(",")[0].strip() if region else ""

    remote_type_raw = soup_job.find('span', {'class': re.compile('^bi bi-building mr-2.*')})
    remote_type_raw = remote_type_raw.parent()[1].getText(separator=" ", strip=True) if remote_type_raw else ""
    remote_type = parse_remote_type(remote_type_raw) if remote_type_raw else ""

    employment_type = soup_job.find('span', {'class': re.compile('^bi bi-clock-history mr-2.*')})
    employment_type = employment_type.parent()[1].getText(separator=" ", strip=True) if employment_type else ""
    employment_type = employment_type.lower()

    salary = soup_job.find('span', class_='public-salary-item')
    salary = salary.getText(separator=" ", strip=True) if salary else ""
    salary = parse_salary(salary_raw=salary)

    additional_info = soup_job.find('div', class_='card job-additional-info')
    additional_info = additional_info.getText(separator=" ", strip=True) if additional_info else ""

    vacancy = {
        'url': job_link,
        'title': title,
        'description': description,
        'company': company,
        'updated': date_,
        'region': region,
        'country': 'Україна',
        'remote_type': remote_type,
        'employment_type': employment_type,
        'salary': salary,
        'additional_info': additional_info,
        'seniority': seniority,
        'date_gathered': TODAY.strftime('%d/%m/%Y')
    }

    return vacancy


def write_to_json(path: str, vacancies: list[dict]):
    """
        Записування даних про роботи у JSON-файл.

        Функція записує надані дані про роботи у JSON-файл з вказаною шляхом.

        Args:
```

```
        path (str): Шлях до файлу, у який будуть записані дані.
        vacancies (List[Dict[str, Any]]): Список словників з даними про роботи.
    """
    with open(os.path.join(path, FILE_NAME), 'w', encoding='utf-8') as f:
        json.dump({"jobs": vacancies}, fp=f, indent=4, ensure_ascii=False)


def run(save_path: str, max_pages_to_collect: int):
    """
        Запуск збору даних про роботи з вказаною кількістю сторінок і збереження у JSON-файл.

        Функція запускає процес збору даних про роботи з обмеженою кількістю сторінок
        та зберігає отримані дані у JSON-файл за вказаним шляхом.

        Args:
            save_path (str): Шлях до каталогу, у якому буде збережений JSON-файл з даними.
            max_pages_to_collect (int): Максимальна кількість сторінок для збору даних.
    """
    pages = get_pages_links()[:max_pages_to_collect]

    data = []
    for current_page in pages:
        jobs_link = get_job_links(current_page)
        for current_job_link in jobs_link:
            job_data = get_jobs_data(current_job_link)
            data.append(job_data)
    write_to_json(path=save_path, vacancies=data)
```

Лістинг файлу scrapers/rabota_ua.py

```
import os
import re
import json
import dateparser
import time

from selenium.common.exceptions import WebDriverException
from selenium import webdriver
from selenium.webdriver.common.by import By
from bs4 import BeautifulSoup
from datetime import date
from config import config
from typing import Optional

VERBOSE_FLAG = config.SCRAPERS_VERBOSE_FLAG

TODAY = date.today()

MAIN_PAGE = 'https://rabota.ua/ua/zapros/it/%D1%83%D0%BA%D1%80%D0%B0%D0%B8%D0%BD%D0%B0'
PAGE_NEXT = 'https://rabota.ua/ua/zapros/it/%D1%83%D0%BA%D1%80%D0%B0%D0%B8%D0%BD%D0%B0?page={}'
FILE_NAME = f"rabota.ua_{TODAY.strftime('%d-%m')}.json"

HEADERS = {
    'User-Agent':
        'Mozilla/5.0 (Windows NT 10.0; Win64; x64) '
        'AppleWebKit/537.36 (KHTML, like Gecko) '
        'Chrome/107.0.0.0 '
        'Safari/537.36'
}


def get_response_with_sleep_time(web_driver, sleep_time=1.0):
    """
        Отримання відповіді зі затримкою.

        Функція виконує запит до веб-сторінки за допомогою веб-драйвера із заданою затримкою між запитами.
        Якщо затримка перевищує 300 секунд, функція повертає порожню рядок.

        Args:
            web_driver: Веб-драйвер для виконання запиту до веб-сторінки.
            sleep_time (float): Затримка між запитами у секундах.

        Returns:
            BeautifulSoup: Об'єкт BeautifulSoup з вмістом веб-сторінки, або порожній рядок,
                якщо не вдалося отримати дані.
```

```
    """
    if sleep_time > 300:
        if VERBOSE_FLAG:
            print('Can`t get a data from page')
        return ''
    else:
        time.sleep(1 / 5)  # Get maximum 5 requests per second
        try:
            soup_job = BeautifulSoup(web_driver.page_source, 'html.parser')
            return soup_job
        except Exception as e:
            web_driver.implicitly_wait(sleep_time)
            if VERBOSE_FLAG:
                print(f"Error occur\n{e}\nTrying again ...")
            return get_response_with_sleep_time(web_driver=web_driver, sleep_time=sleep_time * 2)


def get_web_driver():
    """
        Отримання об'єкта веб-драйвера.

        Функція створює та повертає об'єкт веб-драйвера для використання у веб-скрапінгу.

        Returns:
            WebDriver: Об'єкт веб-драйвера.

        Raises:
            Exception: Якщо не вдалося знайти жоден підтримуваний браузер (Chrome, Firefox, Edge).
    """

    option_list = ['--headless', '--disable-extensions', '--disable-gpu', '--disable-dev-shm-usage',
                   '--ignore-certificate-errors', '--hide-scrollbars', '--no-sandbox', '--single-process']

    web_driver = None
    if web_driver is None:
        try:
            options = webdriver.ChromeOptions()
            for opt in option_list:
                options.add_argument(opt)
            web_driver = webdriver.Chrome(options=options)
            if VERBOSE_FLAG:
                print("Using `Chrome` browser as a base for selenium")
        except WebDriverException:
            if VERBOSE_FLAG:
                print("`Chrome` browser not found. Trying different one")
            web_driver = None

    if web_driver is None:
        try:
            options = webdriver.FirefoxOptions()
            for opt in option_list:
                options.add_argument(opt)
            web_driver = webdriver.Firefox(options=options)
            if VERBOSE_FLAG:
                print("Using `Firefox` browser as a base for selenium")
        except WebDriverException:
            if VERBOSE_FLAG:
                print("`Firefox` browser not found. Trying different one")
            web_driver = None

    if web_driver is None:
        try:
            web_driver = webdriver.Edge()
            if VERBOSE_FLAG:
                print("Using `Edge` browser as a base for selenium")
        except WebDriverException:
            if VERBOSE_FLAG:
                print("`Edge` browser not found. Trying different one")
            web_driver = None

    if web_driver is None:
        raise Exception("Browser not found! Search list: [Chrome, Firefox, Edge]")

    return web_driver


def get_pages_links(web_driver):
    """
```

```
        Отримання посилань на сторінки вакансій.

        Args:
            web_driver (WebDriver): Об'єкт веб-драйвера.

        Returns:
            list: Список посилань на сторінки вакансій.
    """
    web_driver.get(MAIN_PAGE)
    total_pages_div_value = "//div[@class='disable ng-star-inserted']/following-sibling::div"

    try:
        total_pages = web_driver.find_element(by=By.XPATH, value=total_pages_div_value).text
        total_pages = int(total_pages)
    except Exception as e:
        if VERBOSE_FLAG:
            print(f'Can`t get count of total pages.\n{e}\nTrying again!')
        web_driver.implicitly_wait(10)
        total_pages = None

    if total_pages is None:
        try:
            total_pages = web_driver.find_element(by=By.XPATH, value=total_pages_div_value).text
            total_pages = int(total_pages)
        except Exception as e:
            if VERBOSE_FLAG:
                print(f'Still can`t get count of total pages.\n{e}\nPages set 100')
            total_pages = 100
    else:
        if VERBOSE_FLAG:
            print("Successfully get count of total pages")

    pages_links = [PAGE_NEXT.format(page) for page in range(1, total_pages + 1)]
    return pages_links


def get_job_links(page_link, web_driver):
    """
        Отримання посилань на вакансії на сторінці.

        Args:
            page_link (str): Посилання на сторінку з вакансіями.
            web_driver (WebDriver): Об'єкт веб-драйвера.

        Returns:
            list: Список посилань на вакансії.
    """
    if VERBOSE_FLAG:
        print(f'Page: {page_link}')

    web_driver.get(page_link)
    for scroll in range(0, 15000, 100):
        web_driver.execute_script(f"window.scrollTo(0, window.scrollY + {scroll})")

    soup = BeautifulSoup(web_driver.page_source, 'html.parser')
    job_link_tags = soup.find_all('a', {'class': re.compile('^card ng-tns')})
    job_links = []
    for jobLink_tag in job_link_tags:
        job_link = 'https://rabota.ua/ua' + jobLink_tag.get('href')
        job_links.append(job_link)

    return job_links


def parse_date(date_raw: str) -> Optional[dateparser.date]:
    """
        Розбір дати з рядка.

        Функція отримує рядок, що містить дату, та повертає об'єкт дати, який отриманий з розбору рядка.

        Args:
            date_raw (str): Рядок, що містить дату.

        Returns:
            dateparser.date: Об'єкт дати, отриманий з розбору рядка.
    """
    return dateparser.parse(date_raw, settings={'TIMEZONE': 'UTC'}, date_formats=["%d %m %Y"])
```

```
def parse_remote_type(remote_raw: str) -> str:
    """
        Розбір типу роботи (віддалено, офіс, гібридний) з рядка.

        Функція отримує рядок, що містить інформацію про тип роботи (віддалено, офіс, гібридний) і повертає
        відповідний тип роботи.

        Args:
          remote_raw (str): Рядок, що містить інформацію про тип роботи.

        Returns:
            str: Тип роботи: 'remote' (віддалено), 'in-office' (офіс), 'hybrid' (гібридний). Повертається
                порожній рядок, якщо тип роботи не знайдений.
    """
    if re.findall(r"(гібридна|office/remote|на ваш вибір)", remote_raw, re.IGNORECASE):
        return 'hybrid'
    elif re.findall(r"тільки віддалено|віддалена робота|remote", remote_raw, re.IGNORECASE):
        return 'remote'
    elif re.findall(r"В офісі/на місці", remote_raw, re.IGNORECASE):
        return 'in-office'
    else:
        return ""


def parse_seniority(title_raw: str) -> str:
    """
        Розбір рівня кваліфікації з рядка.

        Функція отримує рядок, що містить інформацію про рівень кваліфікації, і повертає розібраний рівень.

        Args:
            title_raw (str): Рядок, що містить інформацію про рівень кваліфікації.

        Returns:
            str: Розібраний рівень кваліфікації. Якщо рівень не знайдений, повертається порожній рядок.
    """
    if re.findall(r"(junior|assistant|intern|trainee|entry|молодший|без досвіду)", title_raw, re.IGNORECASE):
        return 'entry-level'
    elif re.findall(r"(middle|mid-level|\smid\s)", title_raw, re.IGNORECASE):
        return 'mid-level'
    elif re.findall(r"(senior|sen-level|\ssen\s|president|director|head of)", title_raw, re.IGNORECASE):
        return 'senior-level'
    elif re.findall(r"tech lead", title_raw, re.IGNORECASE):
        return 'tech-lead'
    elif re.findall(r"team lead|lead", title_raw, re.IGNORECASE):
        return 'team-lead'
    else:
        return ""


def parse_employment_type(employment_type: str) -> str:
    """
        Розбір типу зайнятості з рядка.

        Функція отримує рядок, що містить інформацію про тип зайнятості, і повертає розібраний тип.

        Args:
            employment_type (str): Рядок, що містить інформацію про тип зайнятості.

        Returns:
            str: Розібраний тип зайнятості. Якщо тип не знайдений, повертається порожній рядок.
    """
    if re.findall(r"часткова зайнятість|неповна зайнятість", employment_type, re.IGNORECASE):
        return 'part-time'
    elif re.findall(r"повна зайнятість", employment_type, re.IGNORECASE):
        return 'full-time'
    else:
        return ""


def parse_salary(salary_raw: str) -> str:
    """
        Обробка рядка заробітної плати.

        Args:
            salary_raw (str): Рядок заробітної плати.
```

```
        Returns:
            str: Оброблений рядок заробітної плати.
    """
    return salary_raw.replace("₴", "грн")


def get_jobs_data(job_link, web_driver):
    """
        Отримання даних про вакансію.

        Args:
            job_link (str): Посилання на вакансію.
            web_driver (WebDriver): Об'єкт веб-драйвера.

        Returns:
            dict: Інформація про вакансію.
    """
    web_driver.get(job_link)
    soup_job = get_response_with_sleep_time(web_driver=web_driver)

    description = soup_job.find('div', class_='full-desc ng-star-inserted')
    description = description.getText(separator=" ", strip=True) if description else ""

    title = soup_job.find('h1', {'data-id': 'vacancy-title'})
    title = title.getText(separator=" ", strip=True) if title else ""
    title = re.sub('\\s*\\n.*', '', title)

    seniority = parse_seniority(title)

    company = soup_job.find('div', class_='santa-mr-10 ng-star-inserted')
    company = company.find('a') if company else None
    company = company.getText(separator=" ", strip=True) if company else ""

    date_raw = soup_job.find(
        'span', class_='santa-text-white santa-flex santa-justify-center santa-typo-additional ng-star-inserted')
    date_raw = date_raw.getText(separator=" ", strip=True) if date_raw else None
    date_ = parse_date(date_raw).strftime('%d/%m/%Y') if date_raw else ""

    region = soup_job.find('span', {'data-id': 'vacancy-city'})
    region = region.getText(separator=" ", strip=True) if region else ""

    labels = soup_job.find('div', class_='santa-flex santa-flex-wrap')
    labels = labels.getText(separator=" ", strip=True) if labels else ""

    remote_type = parse_remote_type(labels + " " + title)

    employment_type = parse_employment_type(labels)

    salary = soup_job.find('span', {'data-id': 'vacancy-salary-from-to'})
    salary = salary.getText(separator=" ", strip=True) if salary else ""
    salary = parse_salary(salary_raw=salary)

    additional_info = labels

    vacancy = {
        'url': job_link,
        'title': title,
        'description': description,
        'company': company,
        'updated': date_,
        'region': region,
        'country': 'Україна',
        'remote_type': remote_type,
        'employment_type': employment_type,
        'salary': salary,
        'additional_info': additional_info,
        'seniority': seniority,
        'date_gathered': TODAY.strftime('%d/%m/%Y')
    }
    return vacancy


def write_to_json(path: str, vacancies: list[dict]):
    """
        Записування даних про роботи у JSON-файл.

        Функція записує надані дані про роботи у JSON-файл з вказаною шляхом.
```

```
        Args:
            path (str): Шлях до файлу, у який будуть записані дані.
            vacancies (List[Dict[str, Any]]): Список словників з даними про роботи.
    """
    with open(os.path.join(path, FILE_NAME), 'w', encoding='utf-8') as f:
        json.dump({"jobs": vacancies}, fp=f, indent=4, ensure_ascii=False)


def run(save_path: str, max_pages_to_collect: int):
    """
        Запуск збору даних про роботи з вказаною кількістю сторінок і збереження у JSON-файл.

        Функція запускає процес збору даних про роботи з обмеженою кількістю сторінок
        та зберігає отримані дані у JSON-файл за вказаним шляхом.

        Args:
            save_path (str): Шлях до каталогу, у якому буде збережений JSON-файл з даними.
            max_pages_to_collect (int): Максимальна кількість сторінок для збору даних.
    """
    web_driver = get_web_driver()
    pages = get_pages_links(web_driver=web_driver)[:max_pages_to_collect]

    data = []
    for current_page in pages:
        jobs_link = get_job_links(current_page, web_driver=web_driver)
        for current_job_link in jobs_link:
            job_data = get_jobs_data(current_job_link, web_driver=web_driver)
            data.append(job_data)

    web_driver.quit()
    write_to_json(path=save_path, vacancies=data)
```

Лістинг файлу scrapers/work_ua.py

```
import os
import requests
import re
import json
import time
import dateparser

from bs4 import BeautifulSoup
from datetime import date
from config import config
from typing import Optional

VERBOSE_FLAG = config.SCRAPERS_VERBOSE_FLAG

TODAY = date.today()

MAIN_PAGE = 'https://www.work.ua/jobs-it/?advs=1'
PAGE_NEXT = 'https://www.work.ua/jobs-it/?advs=1&page={}'
FILE_NAME = f"work-ua_{TODAY.strftime('%d-%m')}.json"

HEADERS = {
    'User-Agent':
        'Mozilla/5.0 (Windows NT 10.0; Win64; x64) '
        'AppleWebKit/537.36 (KHTML, like Gecko) '
        'Chrome/107.0.0.0 '
        'Safari/537.36'
}


def get_response_with_sleep_time(link, sleep_time=1.0):
    """
        Отримання відповіді на запит з врахуванням часу очікування.

        Функція відправляє запит до вказаного посилання і отримує відповідь з врахуванням часу очікування між запитами.
        Якщо отримано помилку, функція спробує виконати запит з більшим часом очікування.

        Parameters:
            link (str): Посилання, до якого потрібно відправити запит.
            sleep_time (float): Час очікування між запитами (в секундах). За замовчуванням 1.0.

        Returns:
```

```
            BeautifulSoup: Об'єкт BeautifulSoup з вмістом відповіді на запит, або порожній рядок,
                якщо не вдалося отримати дані.
    """
    if sleep_time > 300:
        if VERBOSE_FLAG:
            print('Can`t get a data from page')
        return ''
    else:
        time.sleep(1 / 5)  # Get maximum 5 requests per second
        response = requests.get(link, headers=HEADERS)
        if response.status_code != requests.codes.ok:
            time.sleep(sleep_time)
            return get_response_with_sleep_time(link, sleep_time * 2)
        else:
            return BeautifulSoup(response.text, 'html.parser')


def get_pages_links():
    """
        Отримання посилань на всі сторінки.

        Функція відправляє запит до головної сторінки і отримує список посилань на всі сторінки.
        Кількість сторінок визначається шляхом отримання значення останньої сторінки з головної сторінки.

        Returns:
            list: Список посилань на всі сторінки.
    """
    response = requests.get(MAIN_PAGE, headers=HEADERS)
    soup = BeautifulSoup(response.text, 'html.parser')

    try:
        total_pages = soup.find('ul', class_='pagination pagination-small visible-xs-block').get_text().strip()
        total_pages = int(re.search('\\d+$', total_pages).group(0))
    except Exception as e:
        if VERBOSE_FLAG:
            print(f'Can`t get count of total pages.\n{e}\nPages set 300')
        total_pages = 300

    pages_links = [PAGE_NEXT.format(page) for page in range(1, total_pages + 1)]
    return pages_links


def get_job_links(page_link):
    """
        Отримання посилань на вакансії на сторінці.

        Функція отримує посилку на сторінку та повертає список посилань на вакансії, знайдені на цій сторінці.

        Args:
            page_link (str): Посилання на сторінку.

        Returns:
            list: Список посилань на вакансії на сторінці.
    """
    if VERBOSE_FLAG:
        print(f'Page: {page_link}')
    soup = get_response_with_sleep_time(page_link)

    job_link_tags = soup.find('div', id='pjax-job-list').find_all('h2')
    job_links = []
    for jobLink_tag in job_link_tags:
        a_tag = jobLink_tag.find('a', {'href': re.compile('/jobs/\\d+')})
        if not a_tag:
            continue

        job_date = a_tag.get('title')
        job_date = re.sub('.*вакансія від\\s*', '', job_date) if job_date else ''

        href_link = a_tag.get('href')
        if not href_link:
            continue
        job_link = 'https://www.work.ua' + href_link

        job_links.append([job_link, job_date])
    return job_links


def parse_date(date_raw: str) -> Optional[dateparser.date]:
```

```
    """
    Розбір дати з рядка.

    Функція отримує рядок, що містить дату, та повертає об'єкт дати, який отриманий з розбору рядка.

    Args:
        date_raw (str): Рядок, що містить дату.

    Returns:
        dateparser.date: Об'єкт дати, отриманий з розбору рядка.
    """
    return dateparser.parse(date_raw, settings={'TIMEZONE': 'UTC'}, date_formats=["%d %m %Y"])


def parse_remote_type(remote_raw: str) -> str:
    """
    Розбір типу роботи (віддалено, офіс, гібридний) з рядка.

    Функція отримує рядок, що містить інформацію про тип роботи (віддалено, офіс, гібридний) і повертає
    відповідний тип роботи.

    Args:
        remote_raw (str): Рядок, що містить інформацію про тип роботи.

    Returns:
        str: Тип роботи: 'remote' (віддалено), 'in-office' (офіс), 'hybrid' (гібридний). Повертається
            порожній рядок, якщо тип роботи не знайдений.
    """
    if re.findall(r"(гібридна|office/remote|на ваш вибір)", remote_raw, re.IGNORECASE):
        return 'hybrid'
    elif re.findall(r"тільки віддалено|віддалена робота|remote|дистанційна робота", remote_raw, re.IGNORECASE):
        return 'remote'
    elif re.findall(r"В офісі/на місці", remote_raw, re.IGNORECASE):
        return 'in-office'
    else:
        return ""


def parse_region(region_raw: str) -> str:
    """
    Розбір регіону з рядка.

    Функція отримує рядок, що містить інформацію про регіон, і повертає розібраний регіон.

    Args:
        region_raw (str): Рядок, що містить інформацію про регіон.

    Returns:
        str: Розібраний регіон. Якщо регіон не знайдений, повертається порожній рядок.
    """
    return re.sub('\\s*шукаємо', ', шукаємо', re.sub(',.*', '', re.sub(',\\s*шукаємо', ' шукаємо', region_raw)))


def parse_seniority(title_raw: str) -> str:
    """
    Розбір рівня кваліфікації з рядка.

    Функція отримує рядок, що містить інформацію про рівень кваліфікації, і повертає розібраний рівень.

    Args:
        title_raw (str): Рядок, що містить інформацію про рівень кваліфікації.

    Returns:
        str: Розібраний рівень кваліфікації. Якщо рівень не знайдений, повертається порожній рядок.
    """
    if re.findall(r"(junior|assistant|intern|trainee|entry|молодший|без досвіду)", title_raw, re.IGNORECASE):
        return 'entry-level'
    elif re.findall(r"(middle|mid-level|\smid\s)", title_raw, re.IGNORECASE):
        return 'mid-level'
    elif re.findall(r"(senior|sen-level|\ssen\s|president|director|head of)", title_raw, re.IGNORECASE):
        return 'senior-level'
    elif re.findall(r"tech lead", title_raw, re.IGNORECASE):
        return 'tech-lead'
    elif re.findall(r"team lead|lead", title_raw, re.IGNORECASE):
        return 'team-lead'
    else:
        return ""
```

```
def parse_employment_type(employment_type: str) -> str:
    """
        Розбір типу зайнятості з рядка.

        Функція отримує рядок, що містить інформацію про тип зайнятості, і повертає розібраний тип.

        Args:
            employment_type (str): Рядок, що містить інформацію про тип зайнятості.

        Returns:
            str: Розібраний тип зайнятості. Якщо тип не знайдений, повертається порожній рядок.
    """
    if re.findall(r"часткова зайнятість|неповна зайнятість", employment_type, re.IGNORECASE):
        return 'part-time'
    elif re.findall(r"повна зайнятість", employment_type, re.IGNORECASE):
        return 'full-time'
    else:
        return ""


def get_jobs_data(job_link_and_date):
    """
        Отримання даних про роботу з посилання.

        Функція отримує посилання на роботу і повертає словник з розібраними даними про роботу.

        Args:
            job_link_and_date (str): Посилання на роботу та значення дати.

        Returns:
            dict: Словник з даними про роботу. Ключі словника відповідають різним атрибутам роботи, таким як URL,
                назва, опис, компанія, дата оновлення, регіон, країна, тип віддаленості, тип зайнятості,
                заробітна плата, додаткова інформація, рівень кваліфікації та дата збору даних.
    """
    job_link, date_raw = job_link_and_date
    soup_job = get_response_with_sleep_time(job_link)

    description = soup_job.find('div', id='job-description')
    description = description.getText(separator=" ", strip=True) if description else ""

    title = soup_job.find('h1', id='h1-name')
    title = title.getText(separator=" ", strip=True) if title else ""

    seniority = parse_seniority(title)

    company = soup_job.find('span', class_='glyphicon glyphicon-company text-black glyphicon-large')
    company = company.nextSibling.getText().strip() if company else ""

    date_ = parse_date(date_raw).strftime('%d/%m/%Y')

    region_raw = soup_job.find('span', class_='glyphicon glyphicon-map-marker text-black glyphicon-large')
    if region_raw:
        region_raw = region_raw.nextSibling.getText(separator=" ", strip=True)
        region = parse_region(region_raw)
    else:
        region = ""

    remote_type_raw = soup_job.find('span', class_='glyphicon glyphicon-remote text-black glyphicon-large')
    if remote_type_raw:
        remote_type_raw = remote_type_raw.parent.getText(separator=" ", strip=True)
        remote_type = parse_remote_type(remote_type_raw)
    else:
        remote_type = ""

    employment_type_raw = soup_job.find('span', class_='glyphicon glyphicon-tick text-black glyphicon-large')
    if employment_type_raw:
        employment_type_raw = employment_type_raw.parent.getText(separator=" ", strip=True)
        employment_type = parse_employment_type(employment_type_raw)
    else:
        employment_type = ""

    salary = soup_job.find('span', class_='glyphicon glyphicon-hryvnia text-black glyphicon-large')
    salary = salary.nextSibling.nextSibling.getText(separator=" ", strip=True) if salary else ""

    additional_info = soup_job.find('span', class_='glyphicon glyphicon-tick text-black glyphicon-large')
    additional_info = additional_info.parent.getText(separator=" ", strip=True) if additional_info else ""
```

```
    vacancy = {
        'url': job_link,
        'title': title,
        'description': description,
        'company': company,
        'updated': date_,
        'region': region,
        'country': 'Україна',
        'remote_type': remote_type,
        'employment_type': employment_type,
        'salary': salary,
        'additional_info': additional_info,
        'seniority': seniority,
        'date_gathered': TODAY.strftime('%d/%m/%Y')
    }
    return vacancy


def write_to_json(path: str, vacancies: list[dict]):
    """
        Записування даних про роботи у JSON-файл.

        Функція записує надані дані про роботи у JSON-файл з вказаною шляхом.

        Args:
            path (str): Шлях до файлу, у який будуть записані дані.
            vacancies (List[Dict[str, Any]]): Список словників з даними про роботи.
    """
    with open(os.path.join(path, FILE_NAME), mode='w', encoding='utf-8') as f:
        json.dump({"jobs": vacancies}, fp=f, indent=4, ensure_ascii=False)


def run(save_path: str, max_pages_to_collect: int):
    """
        Запуск збору даних про роботи з вказаною кількістю сторінок і збереження у JSON-файл.

        Функція запускає процес збору даних про роботи з обмеженою кількістю сторінок
        та зберігає отримані дані у JSON-файл за вказаним шляхом.

        Args:
            save_path (str): Шлях до каталогу, у якому буде збережений JSON-файл з даними.
            max_pages_to_collect (int): Максимальна кількість сторінок для збору даних.
    """
    pages = get_pages_links()[:max_pages_to_collect]

    data = []
    for current_page in pages:
        jobs_link = get_job_links(current_page)
        for current_job_link in jobs_link:
            job_data = get_jobs_data(current_job_link)
            data.append(job_data)
    write_to_json(path=save_path, vacancies=data)
```

ДОДАТОК 2 (ЗРАЗОК)

Додаток 1

Ілюстративний матеріал

Рис Б.1 – Слайд 1

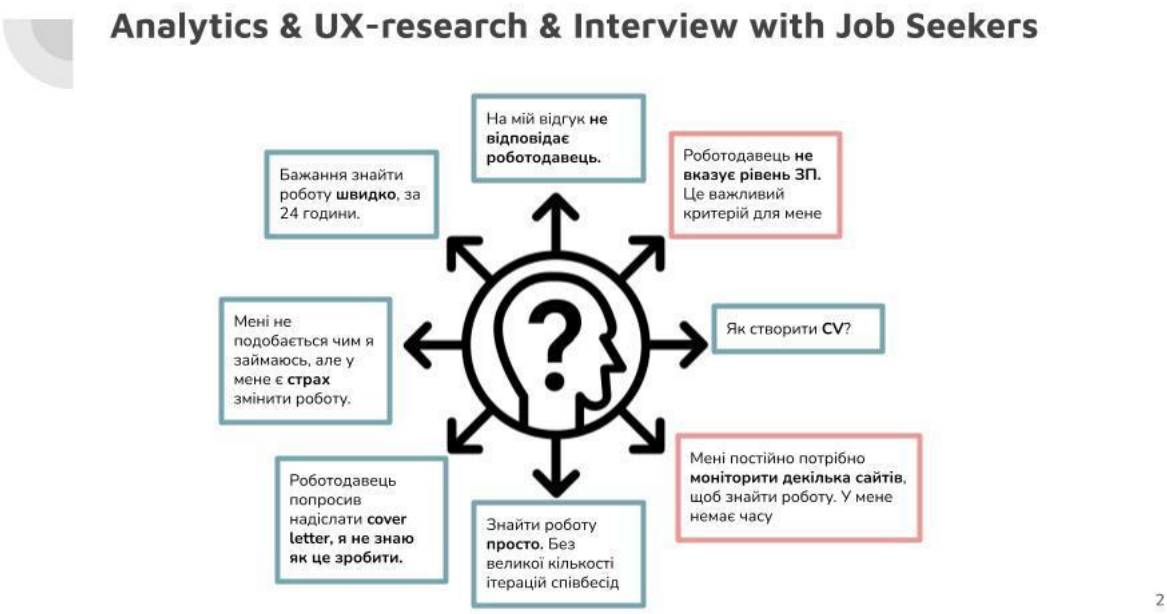


Рис Б.2 – Слайд 2

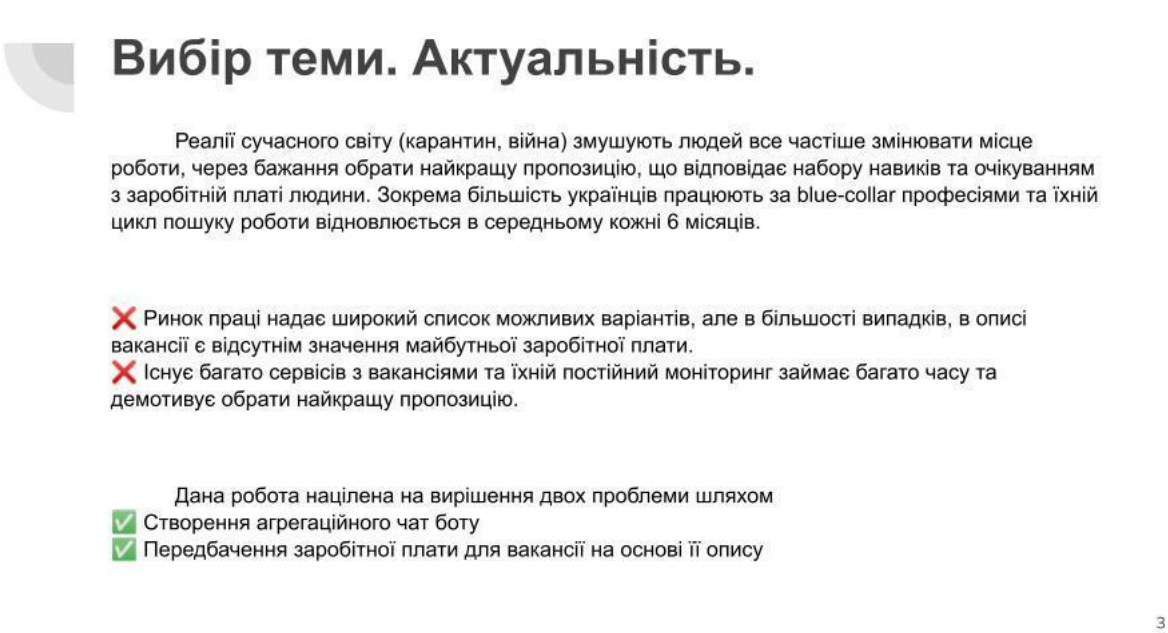


Рис Б.2 – Слайд 3

Рис Б.2 – Слайд 4

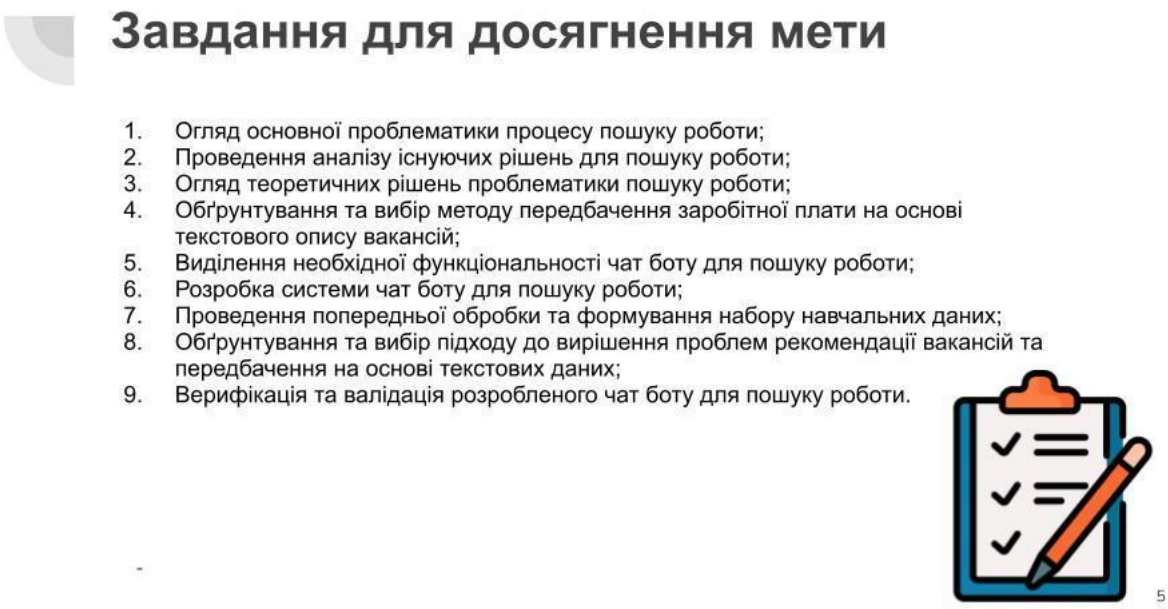


Рис Б.2 – Слайд 5

Рис Б.2 – Слайд 6

Огляд існуючих засобів для пошуку роботи.

На сьогоднішній день, більшість працевлаштувань відбувається **на основі вакансій, розміщених в мережі інтернет.**

Але, **однією з основних проблем** процесу пошуку вакансій, є саме **велика кількість незалежних платформ**. Людина змушена паралельно здійснювати процес пошуку роботи на усіх доступних платформах для підвищення своїх шансів на успіх.

Більшість платформ при створенні вакансії не зазначають поле "заробітна плата" - обов'язковим до заповнення. В той самий час ЗП є одним з найголовніших критеріїв при виборі вакансії.

В Україні найпопулярнішими за кількістю трафіку є сайти - **Djinni.co, Jooble.co, Rabota.ua, Work.ua.**

7

Рис Б.2 – Слайд 7

Порівняння найпопулярніших засобів для пошуку роботи в Україні

	Djinni.co	Jooble.ua	Robota.ua	Work.ua
Наявність чат боту	*наявний*	*наявний*	*наявний*	*наявний*
Джерело вакансій	*внутрішнє*	*внутрішнє*	*внутрішнє*	*внутрішнє*
Аналітика вакансій	*детальна*	*частково*	*відсутня*	*відсутня*
Прогнозування ЗП	*відсутнє*	*відсутнє*	*відсутнє*	*відсутнє*

Таблиця 1 - Порівняння існуючих засобів для пошуку роботи в Україні

❌ **Для всіх платформ існує чат бот**, але рекомендаційна система чат боту **використовує лише власний сайт**

❌ **Недоліком усіх платформ є відсутність алгоритму**, який би дозволяв **прогнозувати заробітну плату** для вакансій в яких це поле відсутнє.

8

Рис Б.2 – Слайд 8

Інтерфейси взаємодії з користувачем

Найбільш популярними інтерфейсами для взаємодії з користувачем є програмні реалізаціє наступного вигляду:

- Асистент;
- Веб-сайт;
- Чат бот.

9

Рис Б.2 – Слайд 9

Порівняння альтернатив інтерфейсів взаємодії з користувач

	Асистент	Веб-сайт	Чат бот
Складність реалізації	*складно*	*легко*	*легко*
Мультиплатформність	*вузька*	*широка*	*широка*
Персоналізованість	*висока*	*достатня*	*достатня*
Зручність оповіщення	*висока*	*низька*	*висока*
Необхідність дизайну	*повний дизайн*	*повний дизайн*	*мінімум*

Таблиця 2 - Порівняння альтернатив для інтерфейсу взаємодії з користувачем

На основі порівняння основних критеріїв, що наведено в таблиці 2, можна зробити висновок, що фаворитом є **інтерфейс на основі чат боту.** Оскільки дана альтернатива має легку реалізацію, широку мультиплатформність та низьку потребу у дизайні у порівняні з «асистентом». Також, у порівняні з «веб-сайтом», «чат бот» має високу зручність оповіщення та низьку потребу у дизайні інтерфейсу.

Єдиним недоліком чат боту, у порівняні з іншими альтернативами, а саме «асистентом», є нижчий рівень персоналізованості, але легка реалізація та відсутність потреби у дизайні є більш вагомими критеріями для системи з пошуку роботи даного класу. **Тому, найоптимальнішою альтернативою є інтерфейс на основі чат боту.**

10

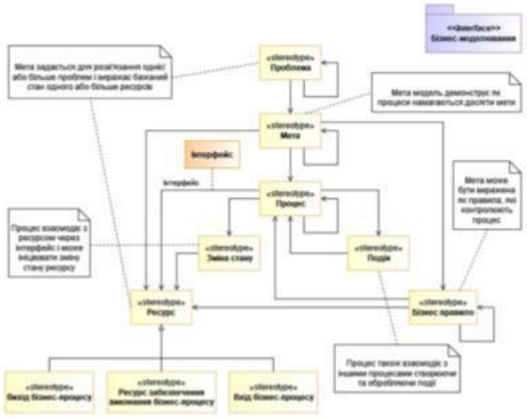
Рис Б.2 – Слайд 10

Концептуальна модель чат боту з пошуку роботи

11

Рис Б.2 – Слайд 11

Бізнес профіль чат боту



Процеси:

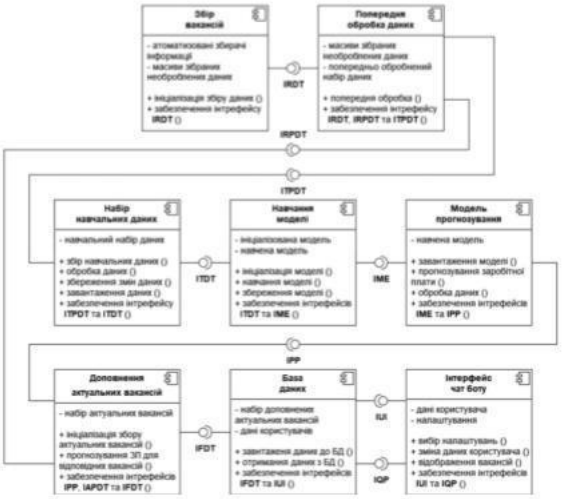
1. Автоматизований збір даних з текстовим описом вакансій;
2. Попередня обробка даних;
3. Формування навчального набору даних;
4. Навчання моделі для прогнозування ЗП;
5. Прогнозування ЗП на основі опису вакансій;
6. Формування набору актуальних вакансій;
7. Доповнення актуальних вакансій за допомогою прогнозування ЗП;
8. Рекомендація вакансій користувачу;
9. Функціонування інтерфейсу користувача.

Рисунок 1 - Удосконалений бізнес-профіль Еріксона-Пенкера [14]

12

Рис Б.2 – Слайд 12

Модель чат боту. Діаграма компонентів в нотації UML



Компоненти

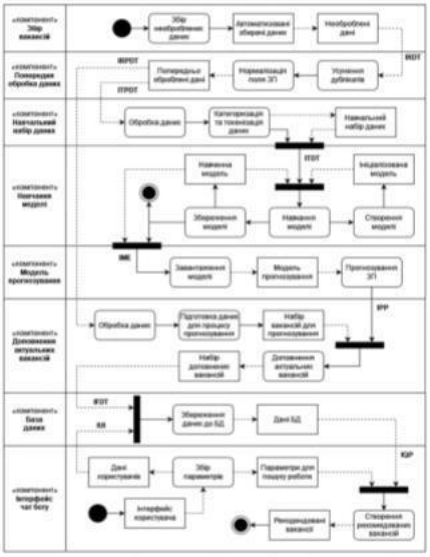
1. Збір вакансій.
2. Попередня обробка даних
3. Набір навчальних даних
4. Навчання моделі
5. Модель прогнозування
6. Доповнення актуальних вакансій
7. База даних
8. Інтерфейс чат боту

13

Рисунок 2 - Модель чат боту. Діаграма компонентів UML

Рис Б.2 – Слайд 13

Модель діяльності чат боту. Діаграма діяльності з водними доріжками другого рівня в нотації UML



Список інтерфейсів:

IRDT (Interface of Raw Data Transferring) – інтерфейс передавання необроблених даних (із модуля Збір вакансій) на вхід відповідного модуля Попередньої обробки даних;

IRPDT (Interface of Relevant Preprocessed Data Transferring) – інтерфейс передавання попередньо оброблених даних (із модуля Попередньої обробки даних) на вхід відповідного модуля Доповнення актуальних вакансій;

ITPDT (Interface of Training Preprocessed Data Transferring) – інтерфейс передавання попередньо оброблених даних (із модуля Попередньої обробки даних) на вхід відповідного модуля Набору навчальних даних;

ITDT (Interface of Training Data Transferring) – інтерфейс передавання набору навчальних текстових даних (із модуля Набору навчальних даних) на вхід відповідного модуля Обробки текстових даних;

IME (Interface of Model Extraction) – інтерфейс отримання та передачі результуючої навченої моделі прогнозування ЗП (із модуля Навчання моделі) на вхід відповідного модуля Моделі прогнозування для подальшої реалізації;

IPP (Interface of Prediction Process) – інтерфейс реалізації процесу передбачення ЗП (на основі модуля Моделі Прогнозування) для відповідних тестових даних з описом актуальних вакансій у яких відсутнє поле ЗП, що знаходять з відповідного модуля Доповнення актуальних вакансій;

IFDT (Interface of Full Data Transferring) – інтерфейс передавання набору навчальних текстових даних (із модуля Набору навчальних даних) на вхід відповідного модуля Обробки текстових даних;

IUI (Interface of User Interactions) – інтерфейс введення необхідних параметрів та передачі налаштувань, що було надано користувачем (із модуля Інтерфейсу чат боту) на вхід відповідного модуля Бази даних вакансій;

IQP (Interface of Query Processing) – інтерфейс ініціалізації процесу пошуку роботи та рекомендації вакансій (із модуля Бази даних вакансій) на вхід відповідного модуля Інтерфейсу чат боту.

14

Рисунок 3 - Деталізована діаграма процесів на основі діаграми діяльності з водними доріжками другого рівня в нотації UML

Рис Б.2 – Слайд 14

Математичне забезпечення чат боту

15

Рис Б.2 – Слайд 15

Огляд алгоритмів для передбачення на основі даних про вакансію

Оскільки, найкраще з обробкою текстових даних справляються методи на основі **ШІ**, а саме алгоритми **NLP** [1]. Серед **найбільш популярних рішень**, можна виділити три наступних методи прогнозування, що іноді виділяють як основні:

1. Наївний Баєсів класифікатор;
2. Ансамблеві моделі типу random forest;
3. Одновимірні згорткові нейронні мережі.

16

Рис Б.2 – Слайд 16

Порівняння алгоритмів

	NBC	RF	1D-CNN
Складність реалізації	*низька*	*низька*	*висока*
Швидкість навчання	*висока*	*низька*	*низька*
Схильність до перенавчання	*висока*	*низька*	*низька*
Узагальнююча здатність	*низька*	*середня*	*висока*
Стійкість до шуму	*низька*	*висока*	*висока*
Вплив розміру датасету	*високий*	*середній*	*низький*
Вплив якості датасету	*високий*	*високий*	*середній*
Результативність	*середня*	*середня*	*висока*

Таблиця 3 - Порівняння методів прогнозування

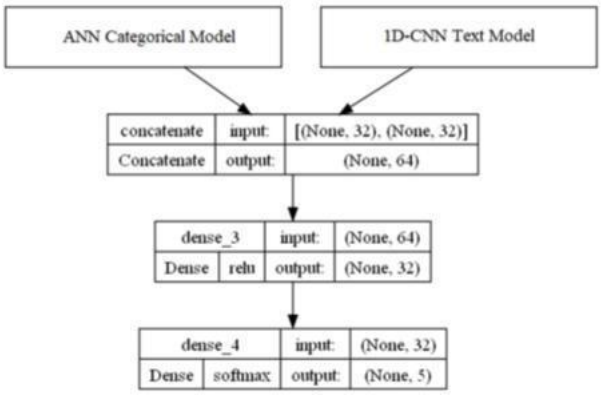
1D - CNN

- ✅ Високі показники узагальнюючої здатності
- ✅ Стійкість до шуму
- ✅ Низька схильність до перенавчання
- ✅ Вплив розміру датасету
- ✅ вплив якості датасету

17

Рис Б.2 – Слайд 17

Архітектура моделі прогнозування



Модель прогнозування має **композитний вигляд** моделей **1D-CNN та ANN** який мотивована тим, що різні типи даних потребують різних типів обробки.

Опис вакансії - текстові дані - послідовні дані
Тип вакансії, сенйорність вакансії - категоріальні дані.

Тобто структура моделі об'єднує дві інші моделі, а саме текстової моделі 1D-CNN та категоріальної моделі ANN.

Поєднуючи моделі відповідні моделі та об'єднавши їх результати, з'являється можливість репрезентувати як локальні функції, так і не послідовні характеристики даних. Це призводить до більш всебічного представлення вхідних даних і може покращити продуктивність моделі.

Рисунок 4 - Структура композитної моделіпрогнозування ЗП

18

Рис Б.2 – Слайд 18

Інтерпретація результатів навчання

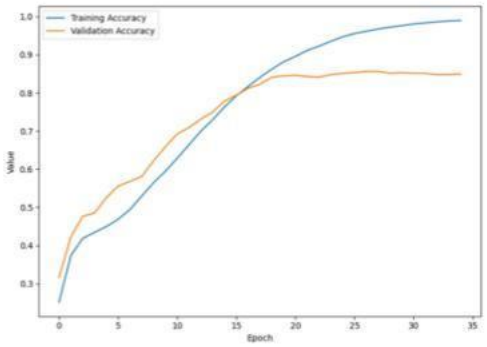


Рисунок 5 - Модель прогнозування ЗП. Значення «accuracy» під час навчання

У результаті процесу навчання, валідаційне значення метрики точності («accuracy») досягло значення **84.957%.**

Гіперпараметр	Значення
Розмір словнику для текстового кодувальника	10'000
Розмірність вектору для кодування слова	64
Максимальна кількість слів	256
Розмірність вектору категорій	16
Кількість епох навчання	35
Розмір партії даних для навчання	64
Коефіцієнт швидкості навчання	$1 \cdot 10^{-4}$
Оптимізатор	*Adam*
Використання зміщень	*True*
Функція втрат	*MSE*
Функція метрики	*Accuracy*
Розмір навчального набору даних	5'325 (90% від 5'917)
Розмір валідаційного набору даних	592 (10% від 5'917)

Таблиця 4 Гіперпараметри для навчання моделі

19

Рис Б.2 – Слайд 19

Програмне забезпечення чат боту

20

Рис Б.2 – Слайд 20

Підхід до User flow - Гейміфікація

Гейміфікація (gamification) - це процес використання елементів гри в негрівованих контекстах для стимулювання участі, мотивації та залучення користувачів до виконання певних завдань чи досягнення певних цілей.

- Пояснити цінність розробленого чат-боту, його функції
- Створити зрозуміле user flow з елементами гейміфікації, щоб покращити такі продуктові метрики як engagement&stikeness

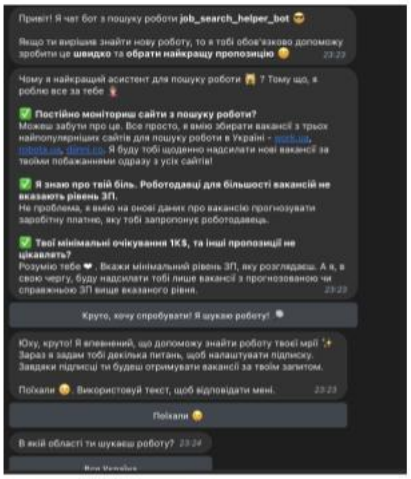
21

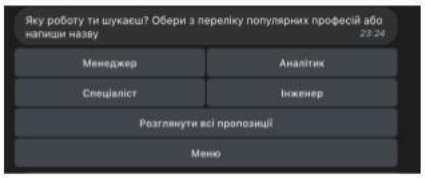
Рис Б.2 – Слайд 21

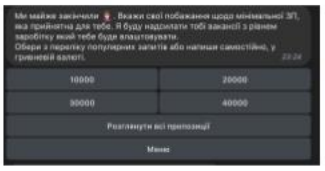
Діалог з користувачем

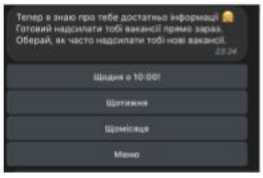
На основі аналізу існуючих рішень, було прийнято рішення, перед початком запуску процесу пошуку вакансій, вимагати від користувача наступний перелік відповідних даних:

1. Регіон пошуку;
2. Професія;
3. Очікувана ЗП;
4. Інтервал отримання рекомендацій.









22

Рис Б.2 – Слайд 22

Рекомендація вакансій

Реалізована на основі даних отриманих в діалозі з пошукачем

- ✅ Регіон
- ✅ Професія
- ✅ Рівень ЗП
- ✅ Частота нотифікацій

Користувач отримує вакансії, які

- ✅ З усіх сайтів - агрегація
- ✅ З прогнозованим рівнем ЗП
- ✅ Найрелевантніші
- ✅ Найновіші

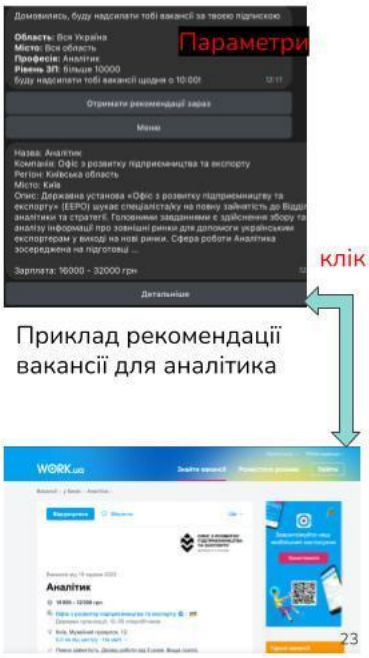


Рис Б.2 – Слайд 23

Рекомендація вакансій

Професія - складна сутність, яка не завжди співпадає з назвою вакансії.

✅ Щоб забезпечувати користувача лише релевантними вакансіями використано метод NLP, що базується на основі обчислення **косинусу подібності (cosine similarity)** [15] між професією та назвою вакансією

Назви вакансій водій, водій автобусу, водій машини, водій категорії А, водій категорії В відповідають професії **водій**.

Водій автобусу ❌ водій
Водій машини ❌ водій
Водій категорії А ❌ водій

cosine similarity

Водій автобусу ✅ водій
Водій машини ✅ водій
Водій категорії А ✅ водій

24

Рис Б.2 – Слайд 24

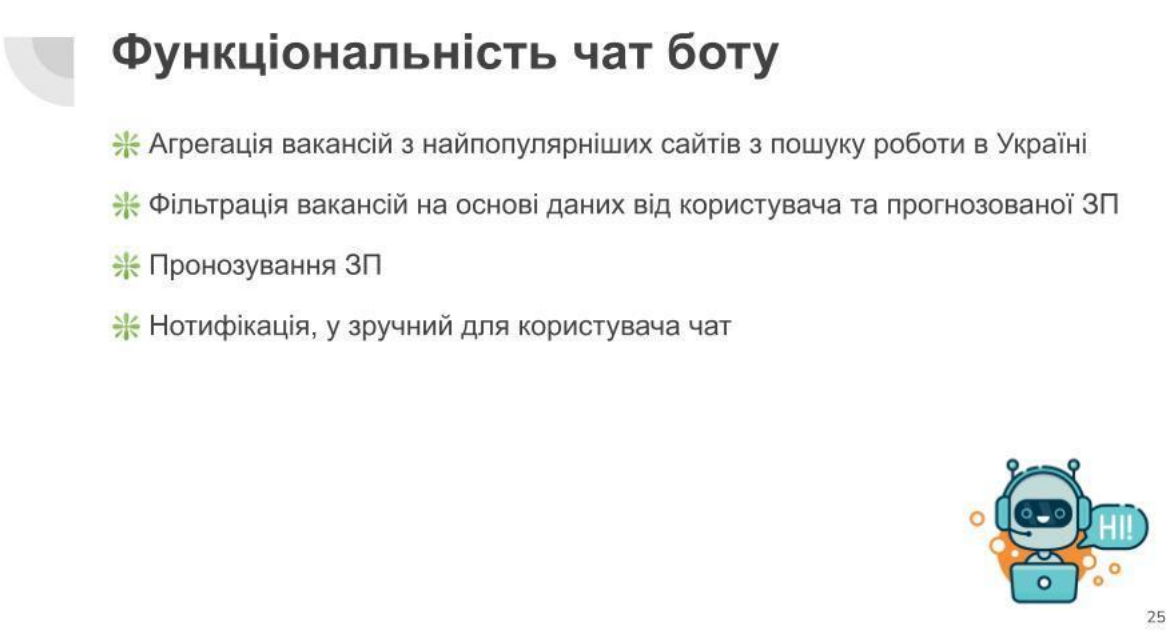


Рис Б.2 – Слайд 25

Рис Б.2 – Слайд 26

Верифікація та валідація порівняльний аналіз розробленої системи

Основні бізнес правила :

BR1 – модель прогнозує заробітну плату в гривневій валюті;
BR2 – для процесу прогнозування заробітної плати, необхідно використовувати такі поля як: назва, компанія, опис, додаткова інформація, регіон, тип працевлаштування, тип базування (remote type), рівень знань (seniority);
BR3 – забезпечити можливість використовування не всіх полів опису вакансії для процесу прогнозування;
BR4 – швидкість прогнозування не повинна перевищувати 1 секунди;
BR5 – навчальний набір даних повинен містити, щонайменше 5000 текстових описів вакансій, що розподіленні за різними спеціальностями;
BR6 – збір вакансій повинно бути реалізовано на основі вебскрапінгу;
BR7 – база даних вакансій повинна оновлюватись щодня;
BR8 – забезпечити механізм звужування пошуку вакансій за наступними критеріями: професія, регіон та заробітна плата;
BR9 – забезпечити автоматичний механізм рекомендації вакансій за можливими інтервалами, а саме: щодня, щотижня, щомісяця та ніколи.

27

Рис Б.2 – Слайд 27

Дотримання бізнес правил

✅ В процесі збору даних, за допомогою вебскраперів, значення поля ЗП було форматовано та конвертовано до гривневого еквіваленту. Тому, розроблена модель прогнозування, навчалась на наборі даних, що представляли значення ЗП у гривневій валюті. Тому, результати, що генерує відповідна модель також у гривневій валюті, що задовольняє вимогу **BR1.**

✅ Після проведення попередньої обробки даних, було виділено відповідний перелік полів: «title», «description», «company», «region», «remote_type», «employment_type», «salary», «additional_info», «seniority». Де поле «salary» є цільовим полем. Тому, в процесі навчання моделі прогнозування ЗП усі поля, на які було накладено вимоги, що задовольняє вимогу **BR2.**

✅ Під час обробки даних, було помічено, що для певних полів значення просто відсутні. Тому у процесі обробки, до кожного поля з пустим значення, було застосовано певне кодування. Тобто, відповідний підхід дозволяє не надавати мережі усі значення для полів, що задовольняє вимогу **BR3.**

✅ У процесі тестування, для вибірки розміром 100 вакансій, модель змогла спрогнозувати відповідні значення ЗП за час менший ніж 1 секунда (0.16 секунди). В загальному випадку, моделі типу 1D-CNN мають низьку обчислювальну складність. Тому, відповідна модель прогнозування задовольняє вимогу **BR4.**

✅ Після процесу попередньої обробки, відповідний набір навчальних даних із тестових описом вакансій містив 5917 записів, що розподілені за різним типами спеціальностей, що задовольняє вимогу **BR5.**

✅ Для процесу збору актуальних вакансій, було обрано три платформи, а саме: Djinni.co, Jooble.ua та Robota.ua. Оскільки, у даному випадку, платформи реалізовано на основі веб-сайтів, було прийнято рішення розробити персоналізовані алгоритми збору даних для кожного джерела вакансій. Відповідні алгоритми використовують ідею вебскрапінгу, що задовольняє вимогу **BR6.**

✅ Розроблена система чат боту, містить в своїй структурі певний таймер, що спрацьовує кожні 24 години. На основі відповідного реалізовано щоденне оновлення бази даних вакансій. Попередні дані про вакансії відкидаються, та проводиться процес їх заміни на нові, актуальні дані. Тому, даний підхід задовольняє вимогу **BR7.** Перед початком процесу рекомендування, чат бот вимагає від користувача перелік даних, до яких входить: регіон (область, місто) пошуку, бажану професію та мінімальний рівень ЗП. Відповідні дані, в подальшому, буде використано для фільтрування потенційних рекомендації для кожного окремого користувача.

✅ Відповідний механізм фільтрування рекомендацій, задовольняє вимогу **BR8.** Додатково, при початку роботи з чат ботом, система запитує бажаний інтервал проведення інформування про актуальні вакансії. В загальному випадку, існує 4 варіанти, які можна змінити в будь-який час, а саме: щодня, щотижня, щомісяця та відсутність автоматичного рекомендування. На основі обраного варіанту, та застосовуючи внутрішній таймер системи, користувач може отримувати повідомлення з актуальними вакансіями згідно вказаного інтервалу. Тому, відповідний підхід задовольняє вимогу BR9.

28

Рис Б.2 – Слайд 28

Рис Б.2 – Слайд 29

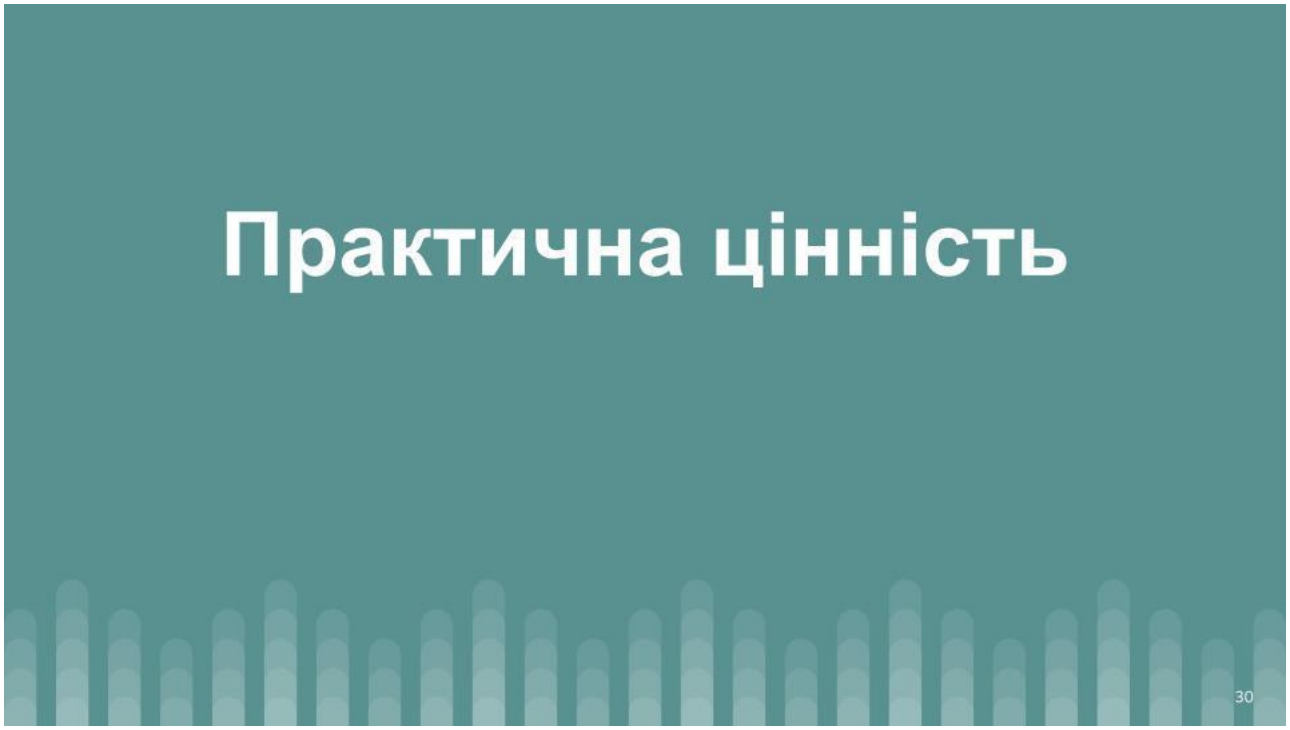


Рис Б.2 – Слайд 30

Розроблено продукт, який:

- 👩‍🍳 👩‍🎤 🧑‍🎤 вирішує реальні потреби пошукачів роботи.

  а) Економія часу на моніторинг сайтів шляхом створення агрегаційного чат боту
  б) Відповідність очікувань щодо ЗП шляхом створення моделі прогнозування.

- 💸 може бути монетизований, за рахунок підписок

- 🌏 легкомаштабований, в напрямку розширення функціональності на інші країни, додавання нової функціональності

31

Рис Б.2 – Слайд 31

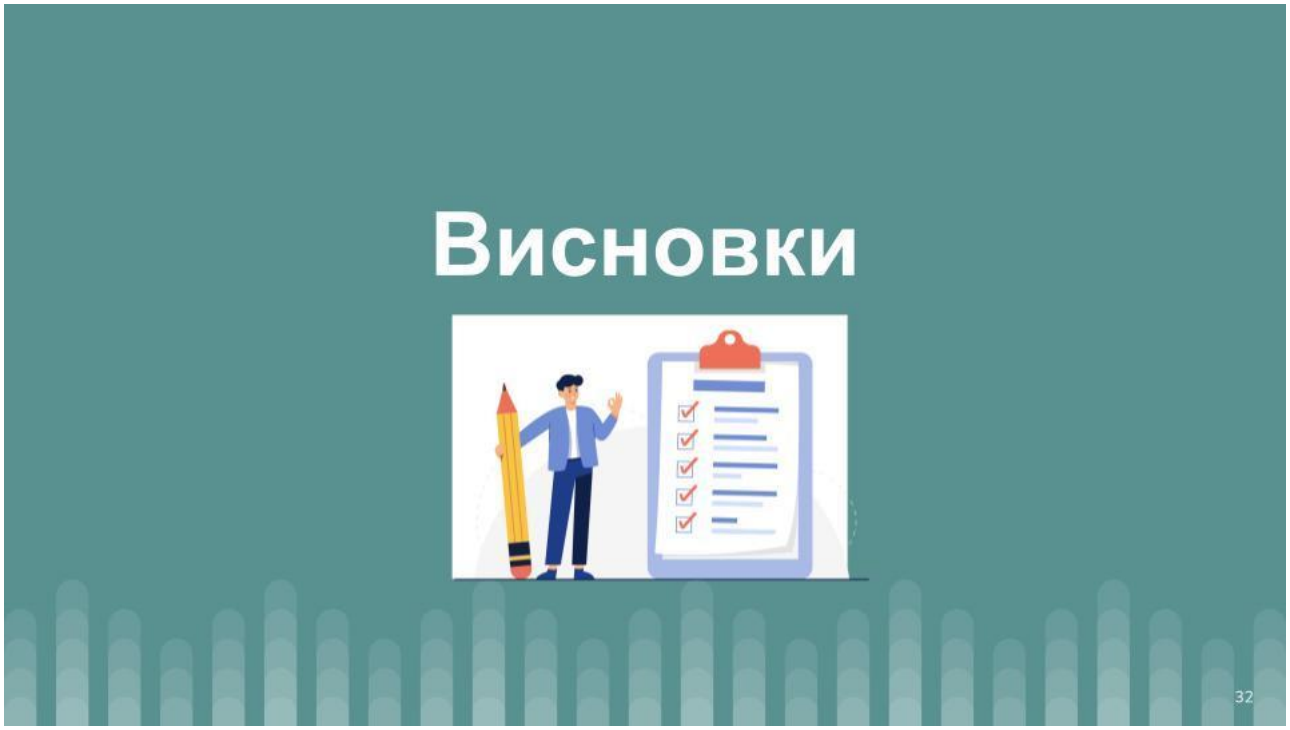


Рис Б.2 – Слайд 32

1. **Досліджено** існуючі системи та мат забезпечення для пошуку роботи і **встановлено**, що в існуючих системах відсутня гнучка функціональність прогнозування ЗП і функції агрегації вакансій з відкритих сайтів.

2. **Розроблено** моделі чат боту яка **реалізує** прогнозування ЗП, агрегацію вакансій класи функцій і відрізняється від існуючи повнотою функціональності та низькою ціною.

3. **Модель** системи **доволяє** маштабувати і розширювати перелік функцій чат боту.

4. **Перспективи** подальших досліджень **передбачають** створення екосистеми чат боту.

33

Рис Б.2 – Слайд 33

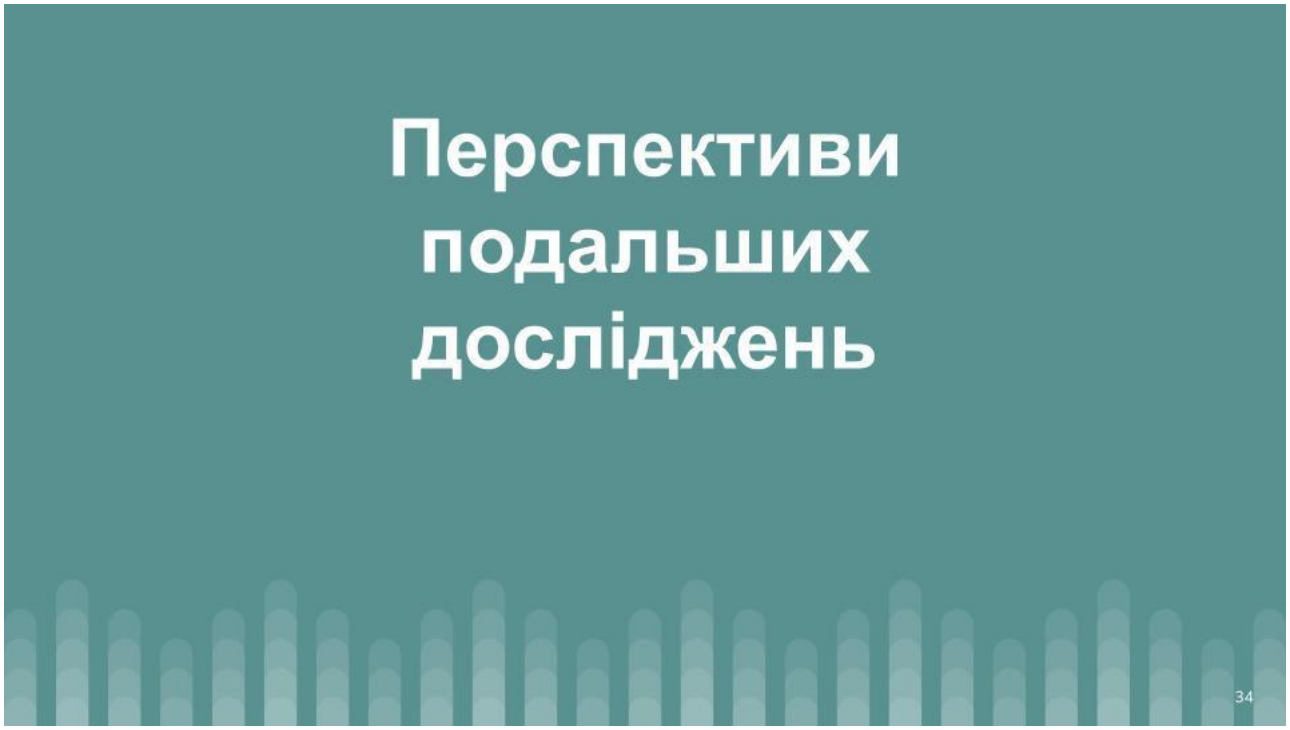


Рис Б.2 – Слайд 34

1. Рекомендація вакансій на основі даних про користувача (CV, профіль)
2. Створення екосистеми :
   a. Аналітика вакансій
   b. Аналітика профілю
   c. Аналітика конкурентів
   d. Salary calculator
   e. Tax calculator
   f. CV review
   g. Cover letter
   h. CV match
3. Дотримання всіх правил JDPR
4. Монетизація продукту (підписка)
5. Маштабування на інші ринки

35

Рис Б.2 – Слайд 35

Перелік посилань

1. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011, March 2). Natural language processing (almost) from scratch. arXiv.org. Retrieved April 20, 2023, from https://arxiv.org/abs/1103.0398
2. Gruetzemacher, R. (2022, April 19). The power of Natural Language Processing. Harvard Business Review. Retrieved April 20, 2023, from https://hbr.org/2022/04/the-power-of-natural-language-processing
3. Manning, C. D., Raghavan, P., & Schütze, H. (2008, July 1). Introduction to information retrieval: Semantic scholar. Semantic Scholar. Retrieved April 20, 2023, from https://www.semanticscholar.org/paper/Introduction-to-information-retrieval-Manning-Raghavan/806c2c4327a31fded64a5d673ab82b133194c234
4. Jurafsky, D., Martin, J. H., & Kehler, A. (2000). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition. Prentice Hall.
5. McCallum, A., & Nigam, K. (1998, January 1). A comparison of event models for naive Bayes text classification. Semantic Scholar. Retrieved April 20, 2023, from https://www.semanticscholar.org/paper/A-comparison-of-event-models-for-naive-bayes-text-McCallum-Nigam/04ce064505b1635583fa0d9cc07cac7e9ea993cc
6. Frank, E., & Bouckaert, R. R. (1970, January 1). Naive Bayes for text classification with unbalanced classes. SpringerLink. Retrieved April 20, 2023, from https://link.springer.com/chapter/10.1007/11871637_49
7. Breiman, L. (2001, October 1). Random forests. Semantic scholar. Retrieved April 20, 2023, from https://www.semanticscholar.org/paper/Random-Forests-Breiman/13d4c2f76a7c1a4d0a71204e1d5d263a3f5a7986
8. Liaw, A., & Wiener, M. (2001, November). Classification and regression by RandomForest. Research Gate. Retrieved April 20, 2023, from https://www.researchgate.net/publication/228451484_Classification_and_Regression_by_ RandomForest
9. Ho, T. K. (1995, August 1). Random decision forests: Proceedings of the third international conference on document analysis and recognition (volume 1) - volume 1. Guide Proceedings. Retrieved April 20, 2023, from https://dl.acm.org/doi/10.5555/844379.844681
10. Kim, Y. (2014, September 3). Convolutional neural networks for sentence classification. arXiv.org. Retrieved April 20, 2023, from https://arxiv.org/abs/1408.5882
11. Zhang, Y., & Wallace, B. (2016, April 6). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv.org. Retrieved April 20, 2023, from https://arxiv.org/abs/1510.03820
12. Shen, Y., He, X., Gao, J., Deng, L., & Mesnil, G. (2014). A latent semantic model with convolutional-pooling structure for Information Retrieval. Semantic scholar. Retrieved April 20, 2023, from https://www.semanticscholar.org/paper/A-Latent-Semantic-Model- with-Convolutional-Pooling-Shen-He/7e8d5a108c28cdfb92f419ce919fbf7993dfebfc
13. Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2019, May 9). 1d convolutional neural networks and applications: A survey. arXiv.org. Retrieved April 20, 2023, from https://arxiv.org/abs/1905.03554
14. Maslianko, P. P., & Sielskyi, Y. P. (2021, August 31). МЕТОД СИСТЕМНОЇ ІНЖЕНЕРІЇ СИСТЕМ НЕЙРОННОГО МАШИННОГО ПЕРЕКЛАДУ. KPI Science News. Retrieved April 20, 2023, from https://doi.org/10.20535/kpisn.2021.2.236938
15. Chamblee, B. (2022, February 7). What is cosine similarity? how to compare text and images in Python. Medium. Retrieved May 3, 2023, from https://towardsdatascience.com/what-is-cosine-similarity-how-to-compare-text-and- images-in-python-d2bb6e411ef0
16. Brownlee, J. (2020, June 30). Why one-hot encode data in machine learning? MachineLearningMastery.com. Retrieved May 3, 2023, from https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/
17. Wang, X., Jiang, W., & Zhiyong Luo. (2016). Combination of convolutional and recurrent neural network for sentiment analysis of short texts. ACL Anthology. Retrieved May 3, 2023, from https://aclanthology.org/C16-1229/
18. Kim, Y. (2014, September 3). Convolutional neural networks for sentence classification. arXiv.org. Retrieved May 3, 2023, from https://arxiv.org/abs/1408.5882
19. Goodfellow, I., Bengio, Y., & Courville, A. (2016, November 10). Deep learning (Adaptive Computation and Machine Learning series). Deep Learning. Retrieved May 3, 2023, from https://www.deeplearningbook.org/

36

Рис Б.2 – Слайд 36

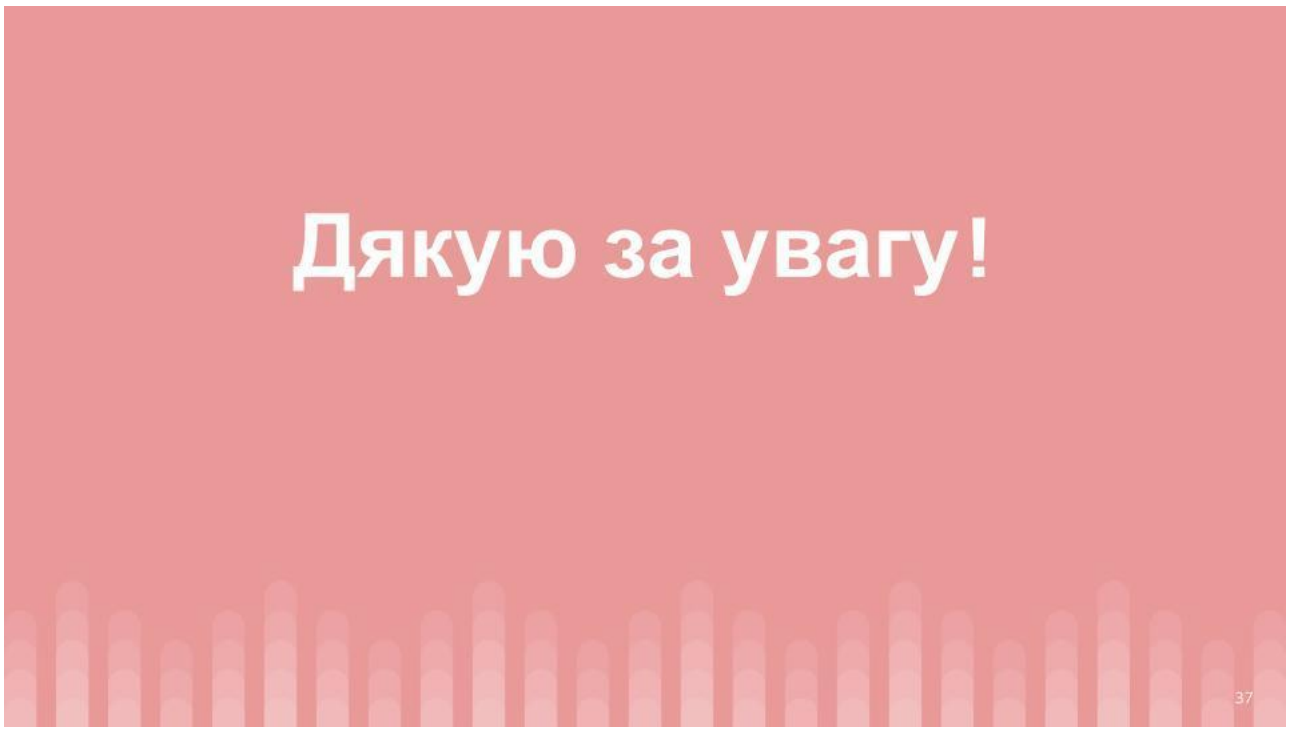


Рис Б.2 – Слайд 37

ДОДАТОК З

Заява студента про затвердження теми та наукового керівника

Завідувачу кафедри прикладної математики
Олегу ЧЕРТОВУ
студентки
Миколенко Оксани Юріївни
Курс 4, група КМ-11

ЗАЯВА

Для виконання бакалаврської кваліфікаційної роботи прошу призначити мені наукового керівника, доцента кафедри прикладної математики ФПМ Маслянко П.П.

Орієнтована тема моєї бакалаврської кваліфікаційної роботи «Система прогнозування розвитку IT сектора в Україні».

Формалізація постановки задачі бакалаврської кваліфікаційної роботи додається.

11.09.2024

Оксана МИКОЛЕНКО _____
(підпис)

Павло МАСЛЯНКО _____
(підпис)

ДОДАТОК 4

Щоденник практики

ВІДГУК І ОЦІНКА РОБОТИ СТУДЕНТА НА ПРАКТИЦІ

Керівник практики від підприємства, організації, установи _____.

_____
(найменування підприємства, організації, установи)

_____ _____
(підпис) (прізвище та ініціали)

"___" _____ 20__ року

ВІДГУК ОСІБ, ЯКІ ПЕРЕВІРЯЛИ ПРОХОДЖЕННЯ ПРАКТИКИ

ВИСНОВОК КЕРІВНИКА ПРАКТИКИ ВІД ВИЩОГО НАВЧАЛЬНОГО ЗАКЛАДУ ПРО ПРОХОДЖЕННЯ ПРАКТИКИ

Дата складання заліку "___" _____ 20__ року
Оцінка:
за національною шкалою _____
(словами)
кількість балів _____
(цифрами і словами)
за шкалою ECTS _____

Керівник практики від
вищого навчального закладу_____
(підпис, прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
"КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО"

ЩОДЕННИК ПРАКТИКИ

Студента _____
Факультет, інститут _____
Кафедра_____
освітня програма _____
код и назва спеціальності_____
назва спеціалізації _____

_____ курс, група _____

РОЗПОРЯДЖЕННЯ

Студент _____ направляється
(прізвище, ім'я, по батькові)
на _____ в м. _____ для
(назва підприємства, установи)
проходження _____ практики
(назва практики)

з "____" _____20__ р. по "____" _____20__ р.

Декан (директор) _____
М.П. (прізвище, ім'я, по батькові)

Студент _____
(прізвище, ім'я, по батькові)

на практику
п р и б у в
"___" _____ 20__ р.

в и б у в
"___" _____ 20__ р.

Керівник підприємства
М.П. _____
Підпис

Керівник практики від підприємства _____
Підпис

Порядок проходження практики

1. Напередодні практики керівник практики від кафедри проводить інструктаж студентів і видає:
— заповнений щоденник (або посвідчення про відрядження);
— робочу програму на групу і для керівника практики від підприємства;
— індивідуальні завдання з практики;
— направлення на практику

2. Після прибуття на підприємство, студент повинен подати керівнику практики від підприємства щоденник і робочу програму практики, ознайомити його із змістом індивідуального завдання, пройти інструктаж з охорони праці, ознайомитися з робочим місцем, правилами експлуатації устаткування та узгодити план проходження практики.

3. Під час практики студент має дотримуватися правил внутрішнього розпорядку підприємства. Відлучатися з місця практики студент може лише з дозволу керівника практики від підприємства.

4. Звіт з практики складається студентом відповідно до програми практики та індивідуального завдання. Залік з практики проводиться в останній день практики комісією кафедри, на яку студент подає повністю оформлені щоденник та звіт.

Порядок ведення і оформлення щоденника

1. Щоденник є основним документом студента під час проходження практики, в якому студент веде короткі записи про виконання програми практики та індивідуального завдання.

2. Раз на тиждень студент зобов'язаний подати щоденник на перегляд керівникам практики від університету та підприємства.

3. Після закінченню практики щоденник і звіт мають бути переглянуті керівниками практики і підписані; складені відгуки про практику і все видано студенту в остаточно оформленому вигляді.

4. Оформлений щоденник разом із звітом студент повинен здати на кафедру. Без заповненого і затвердженого щоденника практика не зараховується.

Примітка. Щоденник заповнюється студентом особисто, крім розділів відгуку про роботу студента на практиці.

_____ тиждень практики

_____ _____
(Дати) (Записи про виконання завдання)

_____

_____

_____

_____

_____

_____

_____

_____ тиждень практики

_____ _____
(Дати) (Записи про виконання завдання)

_____

_____

_____

_____

_____

_____

_____

№ з/п	Назви робіт	Тижні проходження практики					Відмітки про виконання
		1	2	3	4	5	
8.	Сформувати і захистити першу версію шаблону БКР з виконаними всіма розділами та додатками.						
9.	Демонстраційний матеріал – презентація БКР, текст програмного коду.						
10.	Формалізувати наукову новизну (за наявності), практичну цінність та висновки за результатами виконання БКР.						
11.	Підготувати, та затвердити у керівника БКР, презентацію результатів Звіту з переддипломної практики для заліку з практики.						
12.	Подати до комісії з прийому результатів практики остаточну заяву з темою БКР, щоденник з практики, звіт з практики, презентацію звіту з практики.						
13.	Захистити результати практики.						

Керівники практики:

від університету _____

(підпис) **(прізвище та ініціали)**

від підприємства, організації, установи _____

(підпис) **(прізвище та ініціали)**

6

КАЛЕНДАРНИЙ ГРАФІК ПРОХОДЖЕННЯ ПРАКТИКИ

№ з/п	Назви робіт	Тижні проходження практики								Відмітки про виконання
		1	2	3	4	5	6	7	8	
1.	Подати заяву встановленого зразка на ім’я завідувача кафедри з попередньою назвою теми БКР і проханням призначити наукового керівника.									
2.	Виконати формалізацію постановки задачі БКР: об’єкт, предмет і мета роботи, кінцевий результат виконання БКР.									
3.	Сформувати матрицю інформаційних ресурсів (МІР) необхідних для виконання БКР.									
4.	Розробити шаблон БКР – першої версії БКР за форматом і вимогами Положення про державну атестацію бакалаврів КПІ ім. Ігоря Сікорського [2] з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами.									
5.	Затвердити остаточну тему, формалізацію постановки задачі і завдання БКР. У разі необхідності уточнити зміст заяви.									
6.	Подати заяву встановленого зразка на ім’я завідувача кафедри з попередньою назвою теми БКР і проханням призначити наукового керівника.									
7.	Виконати формалізацію постановки задачі БКР: об’єкт, предмет і мета роботи, кінцевий результат виконання БКР.									

3

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ З ПРАКТИКИ

Тема _____

_____

Зміст _____

_____

_____

_____

_____

_____

_____

_____ тиждень практики

_____ _____

(Дати) (Записи про виконання завдання)

_____

_____

_____

_____

_____

_____

_____

_____ тиждень практики

_____ _____

(Дати) (Записи про виконання завдання)

_____

_____

_____

_____

_____

_____

_____

_____ тиждень практики

_____ _____

(Дати) (Записи про виконання завдання)

_____

_____

_____

_____

_____

_____

_____

ДОДАТОК 5

Зразок титульної сторінки звіту з практики

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет прикладної математики
Кафедра прикладної математики

ЗВІТ З ПРАКТИКИ
на тему: «----------»

Виконав: студент 4 курсу, групи КМ-ХХ
ПІБ _____

Керівник практики від університету:
звання, н.ступінь, посада
ПІБ _____

Прийнято з оцінкою _____ балів___

Голова комісії

Члени комісії:
ПІБ _____
ПІБ _____
ПІБ _____

Засвідчую, що в цьому звіті немає запозичень із праць інших авторів без відповідних посилань.
Студент _____

Київ — 2024

ДОДАТОК 6

Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Кафедра прикладної математики

НАЗВА КУРСУ
ПЕРЕДДИПЛОМНА ПРАКТИКА БАКАЛАВРІВ (ПО 10)
Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	*Перший бакалаврський*
Галузь знань	*Математика і статистика*
Спеціальність	*113 прикладна математика*
Освітня програма	*ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА першого (бакалаврського) рівня вищої освіти Наука про дані та математичне моделювання*
Статус дисципліни	*Нормативна*
Форма навчання	*очна*
Рік підготовки, семестр	*4 курс, весінній*
Обсяг дисципліни	*6 кредитів*
Семестровий контроль/ контрольні заходи	*Залік*
Розклад занять	
Мова викладання	*Українська*
Інформація про керівника курсу / викладачів	Керівники практики, призначені згідно Наказу по університету
Розміщення курсу	*https://drive.google.com/drive/folders/1QODKhJpxoXp77OgxkIBJ2M9rEiiD6-d4*

Програма навчальної дисципліни

1 Опис навчальної дисципліни, її мета, предмет вивчення та результати навчання

Практика є однією із основних практичних дисциплін підготовки фахівців з вищої освіти, а переддипломна практика є одним із завершальних етапів підготовки бакалавра освітньо-професійного напряму підготовки у закладах вищої освіти (ЗВО) України.

Мета переддипломної практики – систематизація накопичених студентами знань і умінь для продукування наукового результату та вирішення практичних задач виробництва.

Зокрема метою практики є:

- систематизація, отриманих у процесі навчання, теоретичних знань і практичних навичок та їх застосування для вирішення наукових і прикладних досліджень та розробок;

- набуття практичних навичок реалізації реальних наукових та прикладних розробок для підприємств, установ і організацій;

- уміння формалізувати постановку задачі та проводити науково обґрунтований вибір методів і засобів для вирішення реальних наукових і прикладних досліджень та інженерних розробок;

- уміння формувати структуру і зміст бакалаврських атестаційних робіт за індивідуальними темами, оприлюднення результатів наукових і прикладних досліджень у наукових працях, доповідях, презентаціях, тощо;

- набуття навичок продукування наукових та науково-прикладних результатів;

- набуття професійних компетентностей за спеціальністю і здатності позиціонувати себе на ринку праці.

Предметом практики є поглиблення навичок самостійної практичної роботи, розширення наукового світогляду студентів, остаточного формулювання теми бакалаврської атестаційної роботи, визначення її структури та змісту основних розділів.

Під час практики у повному обсязі перевіряється рівень теоретичної та практичної підготовки студентів, їхніх знань, умінь, навичок, засвоєних у процесі навчання, та вміння застосовувати їх для вирішення практичних інженерних задач.

Згідно з вимогами освітньо-професійної програми, студенти після засвоєння кредитного модуля «Переддипломна практика» повинні набути таких компетентностей:

Загальні компетентності

ЗК2	Здатність застосовувати знання у практичних ситуаціях
ЗК 7	Здатність до пошуку, оброблення та аналізу інформації з різних джерел.
ЗК 8	Знання та розуміння предметної області та розуміння професійної діяльності
ЗК 10	Навички у використанні інформаційних і комунікаційних технологій

Фахові компетентності

ФК 3	Здатність обирати та застосовувати математичні методи для розв’язання прикладних задач, моделювання, аналізу, проектування, керування, прогнозування, прийняття рішень методів та технологій опису моделі об’єкта дослідження та взаємозв’язків між елементами моделі.
ФК 6	Здатність розв’язувати професійні задачі за допомогою комп’ютерної техніки, комп’ютерних мереж та Інтернету, в середовищі сучасних операційних систем, з використанням стандартних офісних додатків.
ФК 8	Здатність використовувати сучасні технології програмування та тестування програмного забезпечення.
ФК 14	Здатність сформулювати математичну постановку задачі, спираючись на постановку мовою предметної галузі, та обирати метод її розв’язання, що забезпечує потрібні точність і надійність результату.
ФК 15	Здатність брати участь у складанні наукових звітів із виконаних науково-дослідних робіт та у впровадженні результатів проведених досліджень і розробок.

Мають продемонструвати такі **результати навчання**:

РН 1	Демонструвати знання й розуміння основних концепцій, принципів, теорій прикладної математики і використовувати їх на практиці.
РН 3	Формалізувати задачі, сформульовані мовою певної предметної галузі; формулювати їх математичну постановку та обирати раціональний метод вирішення; розв'язувати отримані задачі аналітичними та чисельними методами, оцінювати точність та достовірність отриманих результатів.
РН 8	Поєднувати методи математичного та комп'ютерного моделювання з неформальними процедурами експертного аналізу для пошуку оптимальних рішень.
РН 12	Розв'язувати окремі інженерні задачі та/або задачі, що виникають принаймні в одній предметній галузі: в соціології, економіці, екології та медицині
РН 14	Виявляти здатність до самонавчання та продовження професійного розвитку
РН 16	Демонструвати навички взаємодії з іншими людьми, уміння працювати в команді.

Програма практики повинна:

- базуватись на освітніх програмах підготовки бакалаврів;
- запропонувати студентам дорожню карту і регламентувати стадії засвоєння знань і набуття умінь у процесі проходження практики;
- надавати чіткі критерії і процедури оцінювання рівня набутих знань, умінь і компетенцій, набутих у процесі проходження практики.

2 Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Дисципліна «Переддипломна практика» запланована в весняному семестрі на 4 курсі для бакалаврів за освітньо-професійною програмою за спеціальністю 113 Прикладна математика та забезпечують єдиний комплексний підхід до організації практичної підготовки, системності, безперервності і наступності навчання студентів.

Пререквізитами є основні дисципліни, які студенти вивчали упродовж 4 років навчання в бакалавраті:

1. Математична статистика.
2. Математичне моделювання.
3. Бази даних.
4. Основи машинного навчання.
5. Машинне навчання.
6. Алгоритми і системи комп'ютерної математики.
7. Методи штучного інтелекту.
8. Системи Data Science
9. Бази даних та інформаційні системи.
10. Системний аналіз.
11. Теорія керування.
12. Методи оптимізації.

Постреквізити дисципліни «Переддипломна практика» є «Виконання бакалаврської атестаційної роботи»:

- Повна версія бакалаврської атестаційної роботи(БКР).
- Апробація отриманих результатів у вигляді тез для участі у науково-практичних конференціях або публікація у фаховому виданні.

- Захист бакалаврської атестаційної роботи.

3. Зміст навчальної дисципліни

Ринкові вимоги до кваліфікаційного рівня бакалаврів-випускників технічних ЗВО України вимагають наявності теоретичних, прикладних, технологічних та інших видів компетенцій, необхідних для виробництва товарів та послуг в умовах сучасного високотехнологічного виробництва, жорсткої конкуренції, обмеження ресурсів виробництва, ризиків та невизначеності ринкових умов.

Особливості освітньо-професійних програм підготовки бакалаврів у політехнічних ЗВО України, та провідних технічних університетів світу, у більшості випадків, поєднують і науково-дослідну, і технологічну, і переддипломну складові для підготовки бакалаврських атестаційних робіт і набуття професійних компетенцій. Не є винятком поєднання таких вимог і для спеціальностей ІТ індустрії, зокрема для спеціальності Прикладна математика за спеціалізацією Наука про дані та математичне моделювання.

Наявність інженерної дослідницької складової у бакалаврських атестаційних роботах носить обов'язковий характер для освітньо-професійної програми підготовки.

Наявність елементів наукової новизни у бакалаврських атестаційних роботах не є обов'язковою. Тим не менше, для студентів, які подають до захисту інноваційні бакалаврські атестаційні роботи, має бути надана вся повнота можливостей формалізувати і наукову новизну результатів своєї роботи, і практичну цінність.

Означення 1. Наукова новизна – це належним чином формалізована категорія про нові дані, інформацію або знання, отримані про предмет досліджень, що змінюють, доповнюють або вдосконалюють наявні дані, інформацію або знання про цей предмет досліджень.

Наукова новизна може стосуватись як теоретичних так і прикладних результатів наукових досліджень.

Поряд з науковою новизною не менш важливою характеристикою наукового результату – бакалаврської атестаційної роботи, є практична цінність результатів наукових досліджень.

Означення 2. Практична цінність - це міра оцінювання науково-технічних, економічних, соціально-політичних, та інших потрібних характеристик корисності, ефективності, продуктивності, необхідності, тощо, застосування результатів роботи.

Таким чином при формуванні робочої програми та індивідуальних завдань практики потрібно обов'язково включати до них наукову, науково-технічну, конструкторську, технологічну та інші дослідницькі складові необхідні для отримання наукового результату, що матиме і наукову новизну, і практичну цінність.

Для досягнення мети дисципліни «Переддипломна практика» студентам потрібно виконати певний ряд завдань. Всі завдання практики об'єднані у чотири класи за критеріями змісту та порядку їх виконання. Виконання кожного класу завдань закінчується проведенням поточного/календарного контролю, а виконання завершального четвертого класу завдань – семестровим контролем (заліком).

Клас завдань 1. Завдання формування і систематизації ресурсів для проходження практики:

Написати заяву встановленого зразка на ім'я завідувача кафедри з попередньою назвою теми БКР і проханням призначити керівника кваліфікаційної бакалаврської роботи.

Формалізувати постановку задачі БКР: об'єкт, предмет і мета дослідження/роботи, кінцевий результат виконання БКР;

Сформувати матрицю інформаційних ресурсів (МІР) необхідних для виконання БКР;

Розробити шаблон БКР – першої версії БКР за форматом і вимогами Положення про державну атестацію бакалаврів КПІ ім. Ігоря Сікорського [2] з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами.

За результатами виконання завдань передбачити проведення поточного/календарного контролю №1.

Клас завдань 2.Завдання продукування результатів практики:

Затвердити остаточну тему, формалізацію постановки задачі і завдання БКР. У разі необхідності уточнити зміст заяви;

Сформувати і захистити першу версію шаблону БКР з виконаними всіма розділами та додатками. Додаток А. Демонстраційний матеріал – презентація БКР, Додаток Б. Текст програмного коду;

Поточний контроль/ календарний контроль №2 проведення практики - за результатами виконання завдань і змісту документів для класу завдань 2.

Клас завдань 3.Завдання апробації результатів практики:

Формалізувати наукову новизну, практичну цінність та висновки за результатами виконання БКР.

Підготувати, та затвердити у керівника БКР, презентацію результатів звіту з переддипломної практики для заліку з практики;

Поточний контроль/календарний контроль № 3 проведення практики - за результатами виконання завдань і змісту документів для класу завдань 3.

Клас завдань 4.Завдання звітування (семестровий контроль):

Подати до комісії з прийому результатів переддипломної практики остаточну заяву з темою БКР, заповнений і підписаний щоденник з практики, звіт з практики, презентацію звіту з практики;

Захистити результати практики;

Ознайомитись з порядком попереднього захисту БКР.

Семестровий контроль (залік) проведення практики - за результатами звіту з практики, виконання завдань і змісту документів для класу завдань 4.

Індивідуальне завдання розробляється студентом спільно з керівником БКР та керівником базового підприємства, потім узгоджується з керівником практики. Зміст індивідуального завдання узгоджується з тематикою бакалаврської атестаційної роботи.

Разом з індивідуальним завдання студент формує індивідуальний календарний план проходження практики

Виконання індивідуального завдання контролюється керівником практики від кафедри, науковим керівником БКР та представником бази практики згідно календарного плану.

4 Навчальні матеріали та ресурси

1. Положення про організацію освітнього процесу в КПІ ім.Ігоря Сікорського: затверджено наказом Ректора, НАКАЗ № 7-124 від 20.07.2020. [електронний ресурс]. - Назва з екрану.- Мова укр. - Режим доступу - https://document.kpi.ua/regulations

2. Положення про порядок проведення практики здобувачів вищої освіти Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» затверджено наказом Ректора, НАКАЗ № 7/172 від 24.09.2020. [електронний

ресурс] . - Назва з екрану. - Мова укр.- Режим доступу - https://document.kpi.ua/files/2020_7-172.pdf
3. Маслянко, П. П. Практика здобувачів ступеня магістр [Електронний ресурс] : навч. посіб. для здобувачів ступеня магістр за освітньою програмою «Наука про дані та математичне моделювання» спеціальності 113 Прикладна математика / П. П. Маслянко, Л. О. Ковальчук-Химюк ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 3,62 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 229 с. – Назва з екрана https://ela.kpi.ua/handle/123456789/48625

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Переддипломна практика може проходити на підприємстві, в організації або в навчальному закладі. Керівництво практикою з боку університету здійснює викладач кафедри, відповідальний за проходження практики, керівник випускної кваліфікаційної роботи, з боку підприємства — керівник із числа фахівців за профілем спеціальності.

Керівник практики від кафедри забезпечує здійснення всіх організаційних заходів перед початком практики: інструктаж про порядок проходження практики; надання студентам-практикантам потрібних документів: направлення на практику, щоденник практики.

Керівник випускної кваліфікаційної роботи надає консультації з питань, пов’язаних із написанням бакалаврської атестаційної роботи.

Практика розпочинається з проведення обов’язкового для всіх студентів інструктажу з техніки безпеки на підприємстві та робочих місцях, ознайомлення з правилами внутрішнього розпорядку.

Положення про проведення практики здобувачів вищої освіти КПІ ім. Ігоря Сікорського (Наказ №7-172 від 24.09.2020р. регламентує обов’язки керівника практики та студента.

Керівник практики від кафедри повинен:

- розробити робочі програми практики та узгодити їх з базами практики не пізніше, ніж за два тижні до початку практики;
- попередити студентів про оформлення медичної довідки про стан здоров’я (у разі потреби) за 7 днів до початку практики;
- не пізніше, ніж за 7 днів до початку практики, надати базам практики списки студентів-практикантів для оформлення тимчасових перепусток;
- підготувати на надати студенту або групі студентів направлення на практику;
- при направлені для проходження практики двох і більше студентів призначити старшого групи, який є помічником керівника практики;
- провести збори зі студентами та ознайомити їх з робочими програмами практики;
- видати студентам щоденники з індивідуальним завданням та календарним планом проведення практики;
- забезпечити вчасне прибуття студентів на бази практики та контролювати проходження практики;
- систематично, не рідше одного разу в тиждень, консультувати студентів та контролювати етапи виконання індивідуального завдання згідно календарного плану;
- допомагати керівнику практики від підприємства при складанні характеристики кожного студента;
- брати участь у прийняті заліків з практики;

- перевірити повернення всіма студентами перепусток, літератури та майна підприємству;

- оформити журнал виходу на роботу, а також провести інструктаж з техніки безпеки, якщо студенти проходять практику в структурних підрозділах

університету;

- подати до деканату звіт про результати проведення практики з пропозиціями щодо її удосконалення.

Студенти університету при проходженні практики зобов'язані:

- до початку практики отримати від керівника практики від кафедри направлення на практику, робочу програму практики та щоденник практики;
- своєчасно прибути на базу практики;
- у повному обсязі виконувати всі завдання, передбачені робочою програмою практики, та вказівки її керівників;
- знати і суворо дотримуватись правил охорони праці, техніки безпеки і виробничої санітарії та внутрішнього розпорядку підприємства;
- нести відповідальність за виконану роботу;
- своєчасно оформити звіт та скласти залік з практики.

Щоденник практики

Основним документом, за яким здійснюється контроль проходження практики, є щоденник практики, який видається кафедрою. Щоденник містить індивідуальне завдання, календарний план проходження практики, щотижневі записи. Керівники практики від кафедри та бази практики щотижня перевіряє щоденник і записує свої зауваження. Керівник бакалаврської атестаційної роботи контролює виконання індивідуального завдання. Після закінчення терміну практики керівник від бази практики надає в щоденнику відгук про проходження практики студентом і оцінює її результати оцінкою, керівник бакалаврської атестаційної роботи пише відгук про стан бакалаврської атестаційної роботи в щоденник.

6. Самостійна робота студента

За час проходження практики студент повинен виконати наступний обсяг робіт:

1. Визначення об'єкта та предмета дослідження з урахуванням бази практики (підприємства/установи).
2. Разом з керівником БКР сформувати зміст індивідуального завдання.
3. Визначення термінів, обсягів проведення потрібних теоретичних досліджень та розрахунків.
4. Збір та систематизація інформації щодо об'єкта дослідження.
5. Уточнення теми БКР. Формулювання структури та змісту бакалаврської атестаційної роботи.
6. Аналіз та вибір методів і технологій для реалізації поставленої задачі.
7. Верифікація та валідація результатів роботи;
8. Формування звіту з практики.

Календарний план проведення практики

	Зміст завдань СРС	Термін виконання	Помітка про виконання
1.	Подати заяву встановленого зразка на ім'я завідувача кафедри з попередньою назвою теми БКР і проханням призначити наукового керівника.	Перший, другий тижні практики	
2.	Виконати формалізацію постановки задачі БКР: об'єкт, предмет і мета роботи, кінцевий результат виконання БКР.	Перший, другий тижні практики	
3.	Сформувати матрицю інформаційних ресурсів (МІР) необхідних для виконання БКР.	другий тижні практики	
4.	Розробити шаблон БКР – першої версії БКР за форматом і вимогами Положення про державну атестацію бакалаврів КПІ ім. Ігоря Сікорського [2] з прописаними структурою, завданнями на БКР і виконаними першим розділом БКР з попередньою назвою «Огляд існуючих рішень за темою БКР», та іншими розділами.	третій, четвертий тижні практики	
5.	Затвердити остаточну тему, формалізацію постановки задачі і завдання БКР. У разі необхідності уточнити зміст заяви.	Третій, четвертий тижні практики	
6.	Сформувати і захистити першу версію шаблону БКР з виконаними всіма розділами та додатками. Демонстраційний матеріал – презентація БКР, текст програмного коду.	четвертий , п'ятий тижні практики	
7	Формалізувати наукову новизну (за наявності), практичну цінність та висновки за результатами виконання БКР.		
8.	Підготувати, та затвердити у керівника БКР, презентацію результатів Звіту з переддипломної практики для заліку з практики.	п'ятий тиждень практики	
9.	Подати до комісії з прийому результатів практики остаточну заяву з темою БКР, щоденник з практики, звіт з практики, презентацію звіту з практики.	п'ятий тиждень практики	
10.	Захистити результати практики.	п'ятий тиждень практики	
11.	Ознайомитись з порядком попереднього захисту БКР.	п'ятий тиждень практики	

7. Політика навчальної дисципліни (освітнього компонента)

Вимоги до звіту про практику

Після закінчення практики студенти повинні представити керівнику практики від кафедри письмовий звіт разом із щоденником у встановлений термін (не пізніше трьох днів після закінчення практики) для перевірки, рецензування і допуску до захисту. Письмова рецензія керівника практики від кафедри та наукового керівника БКР заносяться до щоденника студента.

Звіт із практики має містити відомості про виконання студентом програми практики та індивідуального завдання. Систематизація зібраних матеріалів здійснюється студентом під час практики і завершується протягом спеціально виділеного для цієї мети часу, відповідно до програми практики.

Структура і зміст розділів звіту з практики

Звіт з практики складається з таких розділів або документів:

1. Титульна сторінка звіту (за зразком).
2. Зміст звіту з практики.
3. Вступ, з коротким оглядом проблемної області за якою виконується БКР, короткою аргументацією актуальності задекларованих досліджень, коротка анотація розділів звіту.
4. Формалізація постановки задачі БКР (об’єкт, предмет і мета дослідження/роботи, кінцевий результат виконання БКР).
5. Перша повна версія БКР з виконаними всіма розділами і додатками. БКР складається з таких елементів:
   - Титульний аркуш встановленого зразка ;
   - Формалізація постановки задачі БКР та Завдання для виконання БКР;
   - Реферат (українською мовою);
   - Abstract (реферат англійською мовою);
   - Зміст;
   - Перелік умовних позначень, скорочень і термінів;
   - Вступ;
   - Основна частина з переліком розділів, підрозділів та висновків до розділів;
   - Висновки;
   - Список використаної літератури;
   - Додаток А до БКР - Демонстраційна версія програмного забезпечення (архітектура, короткий опис функціональності, мова програмування, результати роботи програмного забезпечення, тощо).
   - Додаток Б до - Презентація першої повної версії БКР з виконаними всіма розділами (окремі слайди ще не розроблені).
6. Висновки до звіту.

Підведення підсумків практики

Звіт захищається студентом на комісії [3], призначеній завідувачем кафедри. До складу комісії входять керівники практики від кафедри, керівники бакалаврської атестаційної роботи. Комісія приймає залік протягом перших десяти днів після закінчення практики.

Для допуску до заліку з практики студент повинен представити комісії (не пізніше ніж за три дні до закінчення практики) для перевірки та рецензування:

1) звіт з практики(в електронній формі при дистанційній формі проведення занять);

2) презентацію звіту з практики(в електронній формі при дистанційній формі проведення занять);

3) щоденник практики (в електронній формі при дистанційній формі проведення занять)

8. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Поточний контроль: перевірка щоденника практики щотижня

Календарний контроль: перевірка виконання чотирьох класів завдань згідно календарного плану.

Семестровий контроль: захист звіту з практики, залік.

Умови допуску до семестрового контролю: мінімально позитивна оцінка за виконання індивідуальне завдання мінімальний рейтинг 60 балів.

Результати практики.

1) Наявність документів: звіт з практики; презентації звіту з практики; щоденник практики.

2) Захист звіту практики та виконання індивідуального завдання практики

Критерії оцінювання:

Наявність належним чином оформлених документів: звіт з практики; презентації звіту з практики; щоденник практики - 60 балів.

Захист звіту практики та виконання індивідуального завдання практики:

- повне виконання індивідуального завдання – 40 балів;
- неповне виконання індивідуального завдання –10-20 балів;
- достатня відповідність змісту індивідуального завдання –0-5 балів.

Таким чином за надання повного переліку документації: звіт з практики; презентації звіту з практики; щоденник практики студент отримує - 60 балів.

За захист результатів виконання індивідуального завдання практики студент отримує до 40 балів.

Максимальна оцінка 100 балів.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою:

Кількість балів	*Оцінка*
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно
64-60	Достатньо
Менше 60	Незадовільно
Не виконані умови допуску	Не допущено

Робочу програму навчальної дисципліни (силабус):

Складено ас. Ковальчук-Химюк Л.О., доцентом кафедри ПМА, канд.техн.наук, Маслянко П.П.

Ухвалено кафедрою ПМА (протокол № _10_ від _17.05.23).

Погоджено Методичною комісією факультету (протокол № _11_ від _26.05.23).

Електронне мережне навчальне видання

Маслянко Павло Павович
Ковальчук-Химюк Людмила Олександрівна

ПРАКТИКА ЗДОБУВАЧІВ СТУПЕНЯ БАКАЛАВР

Навчальний посібник

Дизайн, комп’ютерна верстка, коректура авторські

Відповідальний за випуск Сирота С.В.

"ПРОСВІТА"
КИЇВ

Промислово-торговельна фірма «Просвіта»
у формі товариства з обмеженою відповідальністю.
01032, Київ, бульвар Т. Шевченка, 46,
тел. (067) 440 29 96, E-mail prosvita.kyiv@gmail.com
Свідоцтво ДК № 221 від 16.10.2000 р.

Підп. до публікації 03.07.2025.
Обл. вид арк. 10.1

Електронні текстові дані (1 файл: 24,7 Мбайт).