МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп’ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

МЕЛЬНИК Анна Василівна

Розподіл завдань членам проєктної команди на основі інтелектуальної рекомендаційної системи / Tasks Allocation for Project Team Members based on Intelligent Recommendation System

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Управління проектами

Кваліфікаційна робота

Виконала студентка групи
КНУПм-21
А.В. Мельник
_____

Науковий керівник:
к.т.н., доцент
Т.В. Лендюк
_____

Кваліфікаційну роботу
допущено до захисту:
«___» _____ 20___ р.
Завідувач кафедри
_____ Н.В. Дзюбановська

ТЕРНОПІЛЬ - 2025

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «магістр»
спеціальність: 122 – Комп'ютерні науки
освітньо-професійна програма – Управління проектами

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Н.М. Васильків
«____» _____20__р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
МЕЛЬНИК Анна Василівна

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Розподіл завдань членам проєктної команди на основі інтелектуальної рекомендаційної системи / Tasks Allocation for Project Team Members based on Intelligent Recommendation System

керівник роботи к.т.н., доцент Т.В. Лендюк
затверджені наказом по університету від 20 грудня 2024 року № 938 та від 14 жовтня 2025 року №724.
2. Строк подання студентом закінченої кваліфікаційної роботи 1 грудня 2025 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити
- описати проблему з точки зору управління проєктами;
- провести аналіз методів рекомендаційних систем;
- провести аналіз наявних рішень у галузі управління проєктами;
- розробити та описати алгоритм рекомендаційної системи;
- описати математичний компонент системи;
- описати інформаційне забезпечення системи;
- розробити та описати програмне забезпечення;
- описати сценарії роботи користувача.

5. Перелік графічного матеріалу у роботі
- схеми роботи методів рекомендаційних систем;
- схема роботи системи;
- графічний інтерфейс для взаємодії із системою.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 20 грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи.	до 01.01. 2025 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.03. 2025 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 20.05.2025 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 28.10. 2025 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 11.11.2025 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів.	до 25.11.2025 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту.	до 1.12.2025 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 04.12.2025 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії.	до 14.12. 2025 р.	

Студент _____ А.В. Мельник
підпис

Керівник роботи _____ к.т.н., доцент Т.В. Лендюк
підпис

РЕЗЮМЕ

Кваліфікаційна робота на тему «Розподіл завдань членам проєктної команди на основі інтелектуальної рекомендаційної системи» на здобуття освітнього ступеня «Магістр» зі спеціальності 122 «Комп'ютерні науки» освітньої програми «Управління проектами» написана обсягом в 66 сторінок і містить 10 ілюстрацій, 3 таблиці, 2 додатки та 35 використаних джерел.

Метою даної кваліфікаційної роботи є підвищення ефективності управління ресурсами проєкту та результативності виконання цілей та завдань проєкту шляхом розробки інтелектуальної рекомендаційної системи, що напівавтоматизує процес призначення завдань членам команди з урахуванням факторів, розрахованих під кожного виконавця.

Методи досліджень: методи аналізу й синтезу, лінійна алгебра, нормалізація даних, методи рекомендаційних систем, методи програмної інженерії, методи сценарного тестування.

Результати дослідження: розроблено повноцінну інтелектуальну рекомендаційну систему для призначення завдань членам проєктної команди. Було сформовано інформаційну модель системи, що включає структуру вхідних даних, зв'язки між сутностями «виконавець», «завдання» та «історія виконання», а також розроблено математичну складову для оцінювання релевантності виконавців.

Результати роботи можуть успішно застосовуватися для використання у галузі управління проєктів для оптимізації роботи керівників команд.

Ключові слова: УПРАВЛІННЯ ПРОЄКТАМИ, ПРИЗНАЧЕННЯ ЗАВДАНЬ, РЕКОМЕНДАЦІЙНА СИСТЕМА, РАНДЖУВАННЯ ВИКОНАВЦІВ .

ABSTRACT

The qualification work on the topic "Tasks Allocation for Project Team Members based on Intelligent Recommendation System" for obtaining the educational degree "Master" in specialty 122 "Computer Science" of the educational program "Project Management" is written in 66 pages and contains 10 illustrations, 3 tables, 2 appendices and 35 sources used.

The purpose of this qualification work is to increase the efficiency of project resource management and the effectiveness of implementing project goals and objectives by developing an intelligent recommendation system that semi-automates the process of assigning tasks to team members, taking into account factors calculated for each performer.

Research methods: methods of analysis and synthesis, linear algebra, data normalization, methods of recommender systems, methods of software engineering, methods of scenario testing.

Research results: a full-fledged intelligent recommendation system for assigning tasks to project team members has been developed. An information model of the system was formed, which includes the structure of input data, relationships between the entities "user", "task" and "execution history", and a mathematical component was developed to assess the relevance of performers.

The results of the work can be successfully applied for use in the field of project management to optimize the work of team leaders.

Keywords: PROJECT MANAGEMENT, TASK ASSIGNMENT, RECOMMENDATION SYSTEM, RANKING OF PERFORMERS.

ЗМІСТ

Перелік умовних позначень

ПРВ – показник релевантності виконавця.

ОР – оцінка релевантності.

ОРВ – оцінка релевантності виконавця.

УП – управління проєктами.

MAE (Mean absolute error) - середня абсолютна помилка

RMSE (Root Mean Squared Error) - середньоквадратична помилка

UI (User Interface) – інтерфейс користувача.

Вступ

У сучасних умовах цифрової трансформації проєктно-орієнтованих підприємств та зростання складності проєктної діяльності ефективний розподіл завдань між членами команди є одним із ключових факторів успіху. Традиційні методи призначення завдань часто базуються на суб'єктивному досвіді керівника проєкту та в умовах багатопроєктного управління не завжди ідеально точно можуть врахувати таку велику кількість факторів, як індивідуальні особливості виконавців, їхні навички, завантаженість та динаміку розвитку компетенцій. Це може призводити до нерівномірного розподілу роботи, зниження продуктивності та рівня задоволеності членів команди.

В цей час стрімкий розвиток штучного інтелекту та рекомендаційних систем відкриває нові можливості для оптимізації цього процесу. Рекомендаційні системи вже успішно застосовуються у галузях електронної комерції, стримінгових сервісів, медицини та освіти. Їх адаптація у галузі управління проєктами дозволяє автоматизувати такий процес, як призначення завдань, роблячи його більш ефективним, обґрунтованим та прозорим. У результаті впровадження системи особа, відповідальна за розподіл та призначення завдань членам команди, отримує можливість зосередити увагу на більш пріоритетних аспектах діяльності та використати зекономлений час для виконання завдань, що здатні забезпечити вищу цінність для проєкту та підприємства.

Актуальність роботи обумовлена необхідністю створення інструментів у галузі управління проєктами, здатних забезпечити ефективне, збалансоване й обґрунтоване призначення завдань у проєктних командах за допомогою методів штучного інтелекту. Особливо це важливо в умовах дистанційної та гібридної роботи, коли командна взаємодія ускладнюється браком фізичної присутності, а контроль за завантаженістю співробітників стає дедалі складнішим. Використання інтелектуальних моделей і пояснювальних рекомендацій сприяє підвищенню ефективності управління командами в цифровому середовищі.

Метою роботи є підвищення ефективності управління ресурсами проєкту та результативності виконання цілей та завдань проєкту шляхом розробки інтелектуальної рекомендаційної системи, що напівавтоматизує процес призначення завдань членам команди з урахуванням факторів, розрахованих під кожного виконавця.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

- описати проблему з точки зору управління проєктами;
- провести аналіз методів рекомендаційних систем;
- провести аналіз наявних рішень у галузі управління IT-проєктами;
- розробити та описати алгоритм рекомендаційної системи
- описати математичний компонент системи;
- описати інформаційне забезпечення системи;
- розробити та описати програмне забезпечення;
- описати сценарії роботи користувача;

Об'єктом дослідження є процес призначення завдань членам команди у проєктно-орієнтованих підприємствах.

Предметом дослідження є методи та алгоритми побудови математичної моделі та рекомендаційної системи із елементами пояснювальної рекомендації для вибору релевантного виконавця із урахуванням персональних характеристик.

Методами дослідження, що були використані для досягнення мети кваліфікаційної роботи: аналізу й синтезу, порівняльний метод, математичні методи, методи управління проєктами, методи рекомендаційних систем, методи програмної інженерії, методи сценарного тестування.

Наукова новизна одержаних результатів полягає у розробці рекомендаційної системи, яка, на відміну від існуючих підходів, інтегрує показник завантаженості виконавця до процесу формування рекомендацій, а також містить компонент пояснювальної рекомендації, що забезпечує прозорість та обґрунтованість прийнятих рішень.

Практичне значення даного дослідження полягає в розробці проєкту рекомендаційної системи, яка може бути використана керівниками та іншими

фахівцями у галузі проєктного менеджменту для економії часу та вирішення такої рутинної, проте важливої, задачі - призначення завдань.

Робота складається зі вступу, трьох розділів, загальних висновків, та списку використаної літератури.

Апробація результатів дослідження. Основні теоретичні положення роботи й практичні результати дослідження представлено у збірнику тез доповідей «Інтелектуальні інформаційні технології в прикладних дослідженнях» [34], а також у збірнику тез міжнародної науково-практичної конференції «Наука, освіта, економіка та суспільство: адаптація до викликів XXI століття» [35].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Опис методик та проблема з точки зору управління проєктами

Сучасні IT-компанії активно використовують системи управління проєктами (УП), наприклад Jira, Asana, ClickUp, Trello, Wrike, Monday [1] тощо для планування, контролю та моніторингу виконання завдань. У цих системах існує можливість створення завдань, пріоритезації, контролю статусів, однак розподіл завдань, як правило, здійснюється вручну керівником команди. Це створює додаткове навантаження та підвищує ймовірність людських помилок. У великих проєктах із сотнями відкритих завдань та десятками учасників це може призвести до значних втрат часу та неефективного використання людських ресурсів. Автоматизація цього процесу могла б знизити навантаження на керівників команд та забезпечити оптимальніше використання людських ресурсів.

З точки зору управління проєктами, основною проблемою є нераціональний розподіл завдань між членами команди, який часто здійснюється без повного урахування їхньої завантаженості чи рівня досвіду у певній області. Це призводить до:

- описати проблему з точки зору управління проєктами;
- нерівномірного розподілу навантаження між учасниками команди;
- збільшення часу виконання завдань через неефективні призначення;
- перекидання завдань, що погіршує планування проєкту;
- зниження прозорості процесу управління та ускладнення контролю виконання.

Зазвичай проєктні або технічні керівники в проектах розробки програмного забезпечення призначають задачі вручну [2], що є трудомістким і суб’єктивним процесом. Крім того, якщо призначити завдання неправильному розробнику, це сповільнює розробку та збільшує витрати. Саме тому впровадження більш автоматизованої системи, яка базується на об’єктивних показниках та характеристиках виконавців — є критично важливим кроком до підвищення

якості управління.

В сьогоднішніх умовах динамічної розробки програмного забезпечення актуальним стає впровадження та використання інтелектуальних технологій, які аналізують попередню діяльність розробників та, наприклад, пропонують найкращого виконавця для нового завдання. Такий підхід дозволяє автоматизувати частину управлінських процесів та підвищити точність прийняття рішень.

Рекомендаційні системи є важливим компонентом сучасних цифрових платформ, допомагаючи покращити користувацький досвід, стимулювати залученість та надавати інструменти для прийняття рішень в різних сферах застосування. На сьогодні відомо про дослідження, що доводять користь та ефективність застосування таких систем у галузі УП. В цілому додатки, що використовують рекомендаційні системи, допомагають у фільтрації інформації та прогнозуванні. Як приклад можна навести: врахування уподобаннь користувача щодо товару та пропонування відповідних продуктів, контенту або послуг на основі його минулої поведінки та уподобань. Такий підхід може допомогти і у розв'язанні задачі призначення завдання виконавцю у команді. Наприклад, рекомендуючи найкращого члена команди для виконання завдання, беручи до уваги певні критерії за якими й буде визначено найбільш компетентну для цього людину.

Загалом існує три основних методи рекомендаційних систем - на основі колаборативної фільтрації, фільтрації на основі вмісту та гібридні [3].

Колаборативна фільтрація – це метод пошуку інформації, який рекомендує елементи користувачам на основі того, як інші користувачі зі схожими вподобаннями та поведінкою взаємодіяли з цим елементом [4]. Іншими ж словами, алгоритми колаборативної фільтрації групують користувачів на основі поведінки та використовують загальні характеристики групи для рекомендації елементів цільовому користувачеві. Системи колаборативної рекомендації працюють за принципом, що схожі користувачі (за поведінкою) мають схожі інтереси та відповідно схожі смаки [4] (рисунок 1.1).

Перевагою кобаборативної фільтрації є те, що вона не потребує залучення предметно-орієнтованих знань, оскільки навчання моделі ґрунтується виключно на історії взаємодій користувачів з об’єктами. Це спрощує її впровадження та дає змогу формувати рекомендації, які можуть бути нетривіальними й неочікуваними для користувача, завдяки виявленню прихованих закономірностей у поведінці подібних користувачів. Водночас даний підхід має обмеження, пов’язані з проблемою «холодного старту», коли нові користувачі або об’єкти не можуть бути коректно рекомендовані через відсутність даних, а також зі складністю інтеграції додаткових ознак без істотного ускладнення моделі [5].

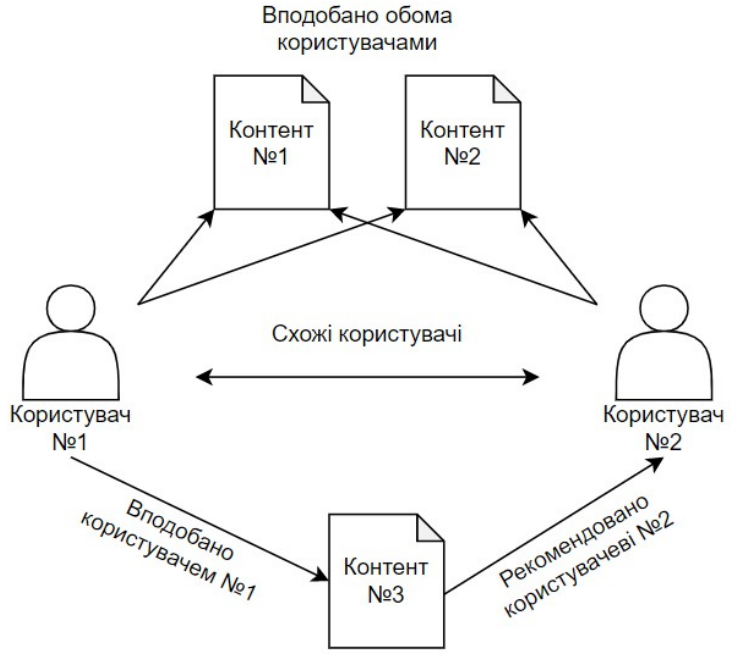


Рисунок 1.1 – Принцип роботи методу колаборативної фільтрації

Фільтрація на основі контенту – це метод пошуку інформації, який використовує характеристики елементів, щоб рекомендувати інші елементи, схожі на ті, що подобаються користувачеві, на основі його попередніх дій або явних відгуків [6]. Цей метод часто враховує характеристики інших елементів, якими користувач зацікавився або уподобав. Варто зазначити, що поняття “контент” в даному випадку є досить широким, оскільки деякі алгоритми рекомендацій на основі контенту зіставляють елементи відповідно до описових

характеристик (наприклад, метаданих) або доданих до елементів, а не до фактичного вмісту елемента. Тим не менш, деякі методи на основі контенту, наприклад, пошук зображень на основі контенту або програми обробки природної мови, зіставляють елементи відповідно до внутрішніх атрибутів елементів (див. рисунок 1.2) [7].

Одна з основних переваг фільтрації на основі контенту полягає в тому, що вона не залежить від даних інших користувачів для створення пропозицій, що робить її особливо ефективною для людей зі специфічними смаками або товарів з низьким рівнем даних про взаємодію з користувачами. Однак вона може бути обмежена якістю характеристик товарів та здатністю алгоритму враховувати тонкощі людських уподобань [3].

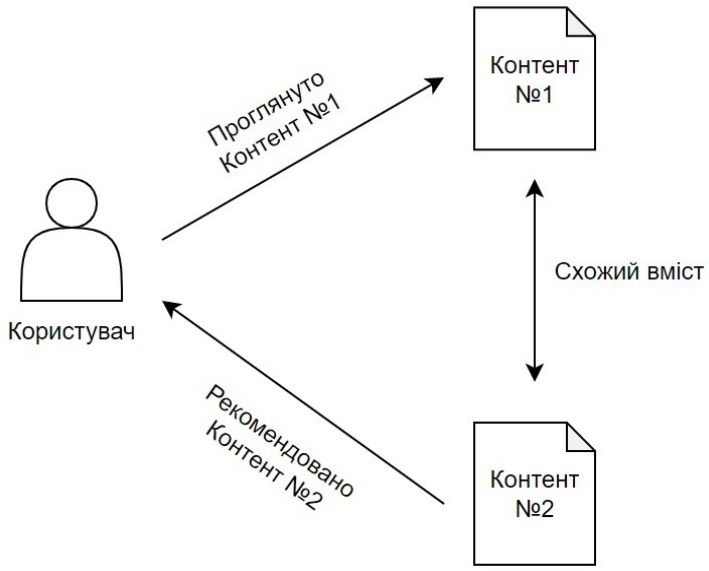


Рисунок 1.2 – Принцип роботи фільтрації на основі контенту

Обидва методи за останні роки знайшли своє застосування в реальному світі, від електронної комерції, як-от Amazon, до соціальних мереж та стрімінгових сервісів. Разом колаборативні та контентні системи утворюють гібридні системи рекомендацій [4].

Гібридна фільтрація в системах рекомендацій поєднує два або більше методів рекомендацій для підвищення точності та усунення обмежень окремих

методів. Найпоширеніший підхід об'єднує колаборативну фільтрацію (яка спирається на взаємодію користувача з елементами) з фільтрацією на основі контенту (яка використовує характеристики елементів або налаштування користувача) [8]. Такі системи часто починають з фільтрації на основі контенту для вивчення нових користувачів і поступово інтегрують спільну фільтрацію, коли стає доступно більше даних про взаємодію [3]. Це призводить до більш точних та різноманітних рекомендацій, оскільки вони балансують загальні тенденції з індивідуальними смаками [9].

Таким чином, основною перевагою гібридних систем є їхня здатність обходити обмеження, властиві однометодним підходам. Колаборативна фільтрація має проблеми з “холодним стартом”, тоді як системи на основі контенту можуть не враховувати нюанси вподобань користувачів, що виходять за рамки функцій елементів [9]. Поєднуючи методи, гібридні системи можуть використовувати метадані елементів для самовизначення рекомендацій для нових користувачів або елементів, водночас використовуючи сигнали спільної роботи для персоналізації.

Гібридні рекомендаційні системи можна розділити на зважені, комбіновані, каскадні, системи доповнення ознак, метарівневі, комутаційні та змішані моделі [3].

Зважені гібридні системи поєднують результати кількох рекомендаційних моделей шляхом агрегації їх оцінок із використанням заданих або адаптивних ваг, що дозволяє збалансувати внесок кожного підходу та підвищити точність рекомендацій [10].

Гібридні системи з перемиканням здійснюють вибір однієї моделі з набору залежно від контексту або доступності даних [10] шляхом використання одиного рекомендатора за певних умов і перемикання на інший, якщо ці умови не виконуються [11]. Це робить їх ефективними для вирішення проблеми холодного старту та адаптації до різних користувачів [10].

Змішані гібридні системи формують рекомендації паралельно за допомогою кількох моделей, забезпечуючи різноманітність і ширше покриття об’єктів без

жорсткого об’єднання результатів [10].

Гібридні системи на основі комбінації ознак об'єднують базові ознаки з різних моделей в єдиний, збагачений набір ознак. Цей об'єднаний набір потім використовується для навчання кінцевої моделі прогнозування [10].

Гібридні системи з розширенням ознак використовують допоміжну модель для генерації нових, більш інформативних представлень даних, які надалі застосовуються основною рекомендаційною моделлю. Цей метод перетворює завдання рекомендації на стандартну задачу машинного навчання з учителем [12].

Каскадні гібридні системи організовують роботу моделей послідовно, де кожна наступна модель уточнює результати попередньої, поєднуючи швидкий відбір і точне ранжування [10].

Гібридні метарівневі системи характеризуються тим, що модель, навчена в межах одного підходу, повністю використовується як вхідна структура для іншої моделі. Даний підхід має певну схожість із системами рекомендацій на основі доповнених ознак, однак принципова відмінність полягає в рівні інтеграції моделей. У підході з доповненими ознаками відбувається лише генерація додаткових характеристик, у випадку ж як метарівневих систем вся модель використовується як вхідне представлення на метарівні. [13]. Це забезпечує глибоку інтеграцію різних методів і дозволяє будувати складні багаторівневі рекомендаційні системи [10].

1.1.1 Огляд методів валідації рекомендаційних систем

В галузі рекомендаційних систем оцінка ефективності рекомендацій має вирішальне значення [14]. Системи рекомендацій можна оцінювати за допомогою кількох метрик та офлайн-експериментів [15]. В межах даної роботи буде описано деякі із них.

Метрики точності допомагають оцінити, наскільки добре система визначає релевантні елементи. Для цього можна використати регресійні метрики, такі як середня абсолютна помилка (MAE) або середньоквадратична помилка (RMSE),

які вимірюють відхилення між прогнозованими та фактичними значеннями [16].

Середня абсолютна помилка (MAE) є однією з найпоширеніших метрик оцінювання якості регресійних моделей та систем прогнозування [15]. Нехай тестова вибірка складається з n спостережень, для кожного з яких відомі прогнозоване системою значення y та фактичне значення x. Для кожного окремого спостереження абсолютна помилка визначається як модуль різниці між фактичним і прогнозованим значеннями. Значення MAE обчислюється шляхом усереднення абсолютних помилок за всією вибіркою та відображає середній рівень відхилення прогнозів системи від реальних значень [15].

З огляду на специфіку метрики, нижчі значення MAE свідчать про вищу точність моделі та кращу відповідність прогнозів фактичним даним [17].

Іншим способом оцінювання якості регресійних моделей і систем прогнозування є середньоквадратична помилка (RMSE) [15]. Порівняно із середньою абсолютною помилкою, ця метрика є більш чутливою до великих відхилень прогнозованих значень від фактичних, оскільки значні помилки роблять непропорційно більший внесок у загальне значення метрики [15]. Це зумовлює доцільність використання даного способу при оцінюванні систем, де виникнення значних помилок є особливо критичним [18].

Принципова відмінність RMSE полягає в тому, що замість усереднення абсолютних значень похибок обчислюється середнє значення квадратів цих похибок. Піднесення помилки до квадрата забезпечує додатність усіх значень та посилює вплив великих відхилень прогнозу від фактичного значення. Після обчислення середнього значення квадратів похибок береться квадратний корінь, що дозволяє повернути значення метрики до тієї ж шкали, в якій вимірюється прогнозована величина, та спростити її інтерпретацію [15].

Як і у випадку з попередньою метрикою, підвищення значення середньоквадратичної помилки свідчить про погіршення якості прогнозування та, відповідно, зростання рівня помилковості системи [15].

Метрики точності і повноти при K – це поширені метрики, які допомагають

оцінити продуктивність алгоритмів ранжування [19].

Метрика точності при k застосовується для оцінювання того, яка частка рекомендацій у сформованому системою списку довжиною k є дійсно релевантною. Вона характеризує якість роботи рекомендаційного або ранжувального алгоритму та відображає частку релевантних елементів серед перших позицій списку, з якими користувач взаємодіє найчастіше. Формально ця метрика визначається як відношення кількості релевантних елементів у топ-k до загальної кількості елементів у топ-k, де топ-k — це перші k елементів у відсортованому (ранжованому) списку результатів. На відміну від метрики повноти при k, яка вимірює, скільки релевантних елементів було знайдено загалом, точність при k надає пріоритет мінімізації нерелевантних результатів [20]. Таким чином, метрика дозволяє оцінити, наскільки ефективно система здатна відбирати найбільш відповідні об’єкти у верхній частині списку рекомендацій [19].

Параметр k у даному випадку виступає як довільно обраний пороговий ранг, що встановлюється дослідником залежно від специфіки сценарію використання. Це виправдано в умовах, коли передбачається, що користувач взаємодіє лише з першими декількома елементами списку, і саме якість цих рекомендацій є критично важливою для оцінювання системи [19].

Для доповнення використовується метрика повноти при k, яка дає змогу оцінити іншу характеристику системи — її здатність знаходити всі релевантні об’єкти у межах перших k позицій. Показник визначається як відношення кількості релевантних елементів, що потрапили до рекомендованого списку довжиною k, до загальної кількості всіх релевантних елементів у тестовому наборі [19].

У поєднанні точність при k і повнота при k забезпечують комплексне та всебічне оцінювання ефективності рекомендаційної системи: перша метрика демонструє, наскільки влучними є рекомендації в топ-k, тоді як друга показує, наскільки повно система охоплює всі релевантні варіанти у цьому діапазоні.

1.2 Огляд наявних підходів в управлінні IT-проєктами

В сучасних дослідженнях розглядається рішення із впровадженням штучного інтелекту у процес призначення проєктних завдань. Наприклад у дослідженні “Issue Auto-Assignment in Software Projects with Machine Learning Techniques” [2] було проведено дослідження на реальних даних з мобільного підрозділу компанії LG Electronics у одному з міст Бразилії. Автори розглядали призначення задач як задачу оптимізації та застосовували кілька алгоритмів машинного навчання серед яких були Naive Bayes, KNN, SVM, Random Forest, навчаючи їх на історичних даних з реальних проєктів вищезгаданої компанії. У дослідженні оцінювались точність призначень, швидкість роботи та зниження кількості помилок.

Автори встановили, що ефективність моделі значною мірою залежить від якості даних та правильної побудови ознак [2]. Попри те, що не всі алгоритми забезпечують однаково високу точність, результати показали, що автоматизація, навіть частково, може значно скоротити навантаження на управлінців і підвищити продуктивність команди. Також підкреслюється важливість практичної інтеграції таких систем у робочі процеси компанії — не лише як технічного рішення, а як частини процесу цифрової трансформації [2]. Методи машинного навчання змогли скоротити час на призначення issue порівняно з ручним методом, хоча і не усі алгоритми давали значно кращу точність. Найкращими моделями виявились SGDText, та Logistic Regression .

Ще одне дослідження [21, 22, 23] від авторів Етема Утку Акташа та Джемаля Їлмаза представили досвід впровадження системи автоматичного призначення завдань у великій комерційній організації — Softtech у Стамбулі, Туреччина. Організація стикнулась із проблемою дуже великого щоденного навантаження на команду, тому ручне призначення виявилось досить трудомістким та могло приводити до затримок у вирішенні критичних проблем. Дослідники розробили та впровадили систему IssueTAG, яка дозволила автоматизувати процес призначення завдань, використовуючи методи аналізу

даних.

Результати досліджень [21] показали підвищення точності призначень завдань, ефективності та зниження помилковості при визначенні. При впровадженні система показала середню точність автоматичних призначень трохи нижчу за ручне призначення: середня щоденна точність до впровадження — 0,864, після — 0,831. Проте незважаючи на незначне зниження точності, система змінила процеси: зменшила ручну працю, скоротила середній час вирішення звітів з 3,26 днів до 2,61 дні за рахунок автоматизації та удосконалення процесу [21].

Дослідження [21] демонструє, що автоматичне призначення звітів про помилки/задачі у промисловому середовищі може бути корисним навіть за умови, що точність алгоритму не перевищує ручну на 100 %. Крім того, перевагою стало те, що на відміну від попередніх досліджень, які здебільшого базувалися на відкритих проєктах або проводилися ретроспективно, IssueTAG була впроваджена у реальному виробничому середовищі і з 12 січня 2018 року здійснює всі призначення завдань у Softtech.

У дослідженні Liang Wei та Luiz Fernando Capretz «Recommender Systems for Software Project Managers» [24, 25] на основі порівняльного аналізу чотирьох відкритих платформ — EasyRec, LensKit, Apache Mahout та Turi — автори оцінили ефективність алгоритмів колаборативного фільтрування на основі користувача та на основі елементів. Результати експерименту показали, що колаборативне фільтрування на основі елементів забезпечує стабільні рекомендації навіть за обмеженості даних, тоді як фільтрування на основі користувача менш ефективне через проблему «холодного старту».

У дослідженні [24] також визначено основні виклики впровадження RSSE у практику: контекстна залежність проєктів, динамічність та гетерогенність даних. Дослідники роблять висновок, що системи рекомендацій мають значний потенціал для оптимізації процесів управління ПЗ, зокрема у виборі технологій, розподілі завдань і формуванні команд, однак потребують адаптації до конкретного середовища розробки.

Наступне дослідження Tao Zhang & Byungjeong Lee під назвою “An

Automated Bug Triage Approach: A Concept Profile and Social Network Based Developer Recommendation” [26] присвячене проблемі «сортування багів» у розробці програмного забезпечення. Коли користувачі або тестувальники знаходять помилку, її потрібно призначити конкретному розробнику для виправлення. Процес сортування помилок є критично важливим, але трудомістким і схильним до помилок. Призначення задачі виправлення невідповідному розробнику збільшує час виконання та витрати на роботу.

Автори роботи [26] досліджують, як можна автоматизувати цей процес за допомогою алгоритмів машинного навчання, щоб підвищити його точність. Основна мета дослідження полягала у порівнянні ефективності різних моделей класифікації (алгоритмів машинного навчання) для автоматичного призначення багів найбільш відповідним розробникам. Дослідники хотіли визначити, яка з поширених моделей (таких як Naive Bayes, Decision Trees, Support Vector Machines тощо) покаже найкращі результати при призначенні розробників.

Дослідження [26] підтвердило, що машинне навчання є життєздатним і ефективним рішенням для автоматизації процесу сортування помилок, що може допомогти компаніям заощадити час та ресурси, швидше виправляючи помилки.

1.3 Постановка задачі

Команда - це група осіб, які виконують роботи проєкту для досягнення його цілей [27]. У сучасних проєктно-орієнтованих організаціях процес розподілу завдань між членами команди є важливим чинником ефективності виконання робіт та досягнення стратегічних цілей проєкту. Традиційні підходи, що ґрунтуються на суб’єктивному оцінюванні управлінця, не завжди забезпечують оптимальний вибір виконавця. В умовах цифрової трансформації та зростання складності проєктного середовища виникає потреба у створенні інтелектуальних інструментів підтримки управлінських рішень.

Перевага запропонованого методу полягає в тому, що він забезпечує більш комплексний, прозорий та адаптивний процес призначення виконавців, порівняно

з традиційними або суто формальними підходами. На відміну від класичних методів, які ґрунтуються переважно на одному критерії — наприклад, досвіді співробітника чи його наявній ролі, — запропонована модель інтегрує одразу кілька ключових показників, що детальніше враховують професійні можливості фахівця. Вона одночасно враховує релевантний досвід, широту навичок (універсальність), фактичну продуктивність, вимірювану середнім часом виконання завдань, а також поточну завантаженість, яка часто ігнорується у простіших підходах. Крім того, система також забезпечує можливість гнучкого налаштування пріоритетів, що дозволяє керівнику самостійно визначати вагомість тих чи інших показників виконавця залежно від специфіки завдання або стратегічних цілей проєкту.

Таким чином актуальність даного дослідження зумовлена потребою у розробці інтелектуальних інструментів у галузі управління проєктами, які можуть забезпечити об’єктивний, раціональний та збалансований розподіл завдань у проєктних командах на основі методів штучного інтелекту.

Метою роботи є підвищення ефективності управління ресурсами проєкту та результативності виконання цілей та завдань проєкту шляхом розробки інтелектуальної рекомендаційної системи, що напівавтоматизує процес призначення завдань членам команди з урахуванням факторів, розрахованих під кожного виконавця.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

1. Проаналізувати проблему з точки зору управління проєктами, розглянути методи рекомандаційних систем, а також готові рішення із застосуванням таких систем у галузі управління проєктами.

2. Розробити алгоритм рекомендаційної системи контентно-орієнтованого типу, який визначає релевантність кожного виконавця щодо обраного завдання.

3. Розробити математичну модель системи, що дозволить обчислювати показник релевантності виконавців на основі показників: досвіду, універсальності, середнього часу виконання аналогічних завдань, рівня поточної завантаженості.

4. Реалізувати програмне забезпечення, що включає: серверну частину на основі Python та бібліотек Pandas, NumPy, Scikit-learn, а також веб-інтерфейс для взаємодії користувача з системою, реалізований за допомогою Flask.

5. Провести тестування та валідацію результатів системи.

Таким чином, розв’язання поставленої задачі дозволить створити адаптивний інтелектуальний інструмент підтримки прийняття рішень, який оптимізує процес призначення завдань, підвищує продуктивність команди та сприяє ефективнішому управлінню ресурсами проєкту.

Висновки до розділу 1

1. У першому розділі було здійснено комплексний аналіз предметної області, спрямований на виявлення проблем та особливостей процесу призначення завдань у командах проєктного типу.

2. Було розглянуто сучасні методики та підходи до автоматизації процесів управління завданнями, включно з використанням різних типів рекомендаційних систем і моделей штучного інтелекту. Аналіз існуючих рішень показав, що хоч на ринку уже й існують дослідження спрямовані на розв'язання задачі призначення завдань, проте, у даній роботі, окрім врахування інших важливих характеристик виконавців, ще й буде враховуватись завантаженість та пріоритетність кожної із характеристик. Це обґрунтовує доцільність розробки спеціалізованої інтелектуальної системи для формування рекомендацій щодо призначення завдань.

3. Також було описано методи валідації рекомендаційної системи, що дозволить підвищити рівень довіри до її роботи.

4. На основі проведеного аналізу було сформульовано постановку задачі дослідження, що передбачає створення контентно-орієнтованої рекомендаційної системи, здатної здійснювати багатокритеріальне оцінювання потенційних виконавців та обґрунтовано визначати найбільш релевантного кандидата для кожного завдання.

5. Таким чином, перший розділ дає теоретичне підґрунтя для подальшої розробки математичної моделі та програмної реалізації системи. Отримані результати є базою для побудови алгоритмів, вибору відповідних технологій та реалізації інтелектуальної системи розподілу завдань у наступних розділах роботи.

2 МОДЕЛЬ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

2.1 Робота системи рекомендацій

Робота системи рекомендацій ґрунтується на багатофакторному оцінюванні релевантності та поточного стану завантаженості розробників проекту. Мета системи рекомендацій — знайти найбільш компетентного і водночас найменш перевантаженого виконавця для нового завдання.

Оцінка виконавця здійснюється за чотирма критеріями:

- C_1 (досвід виконавця для конкретного завдання);
- C_2 (універсальність досвіду);
- C_3 (середній час виконання);
- C_4 (завантаженість виконавця).

Загалом робота системи поділяється на два основні етапи: етап побудови профілів виконавців (етап I) та етап рекомендації (етап II).

На етапі I виконується обробка даних та обчислення всіх необхідних показників. Оскільки інформація про нове завдання може бути непідготовленою, вона спершу проходить попередню обробку та маркування ознак, щоб перетворити інформацію про завдання на змістовне представлення.

Модель виконує вимірювання подібності між новим завданням та наявними історичними даними про завдання з метою надання списку релевантних розробників.

Після цього генерується оцінка релевантності (OP) для кожного розробника, дані про якого належать до відповідного набору даних. Ця оцінка розробника генерується на основі чотирьох критеріїв: досвід, універсальність, середній час виконання та завантаженість працівника.

На етапі II ранжуються всі здібні розробники, що є необхідним для вибору кандидата та зменшення часу на виконання. Список ранжування розробників надає відповідних розробників, які мають можливості та досвід для вирішення нового завдання. Цей процес застосовується лише до тих розробників, які пройшли етап I. Після того, як була застосована стратегія пошуку вимірювання

подібності до здібних працівників, система отримує оцінку релевантності кандидатів за вищезазначеними критеріями. Останній етап складається з ранжування кандидатів у порядку спадання на основі їх відповідної оцінки релевантності та генерування списку потенційних кандидатів, з якого найперший буде рекомендований на виконання, а також пояснення, чому найкраще підходить саме він (рисунок 2.1).

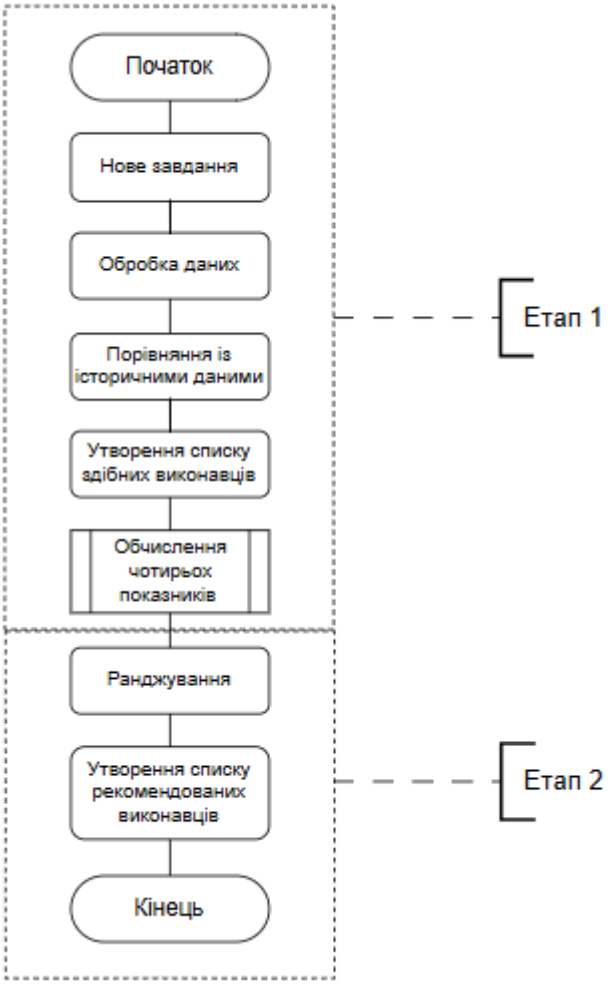


Рисунок 2.1 – Алгоритм роботи системи

Рекомендація здійснюється шляхом вибору найрелевантніших працівників зі списку. Нове завдання рекомендується першому розробнику у списку для початку аналізу та роботи.

Варто зазначити, що підхід не вирішує питання додавання нових розробників, які приєднуються до системи, і не розглядає ранжування нового

розробника у команді, який не має історії виконаних завдань. Ця система працює лише з тими виконавцями, які мають дані у датасеті із історією виконаних задач у минулому.

2.1.1 Математична модель

На етапі I для оцінки ступеня подібності між новим завданням та наявними історичними даними застосовується косинусна подібність [28]. Кожне завдання та профіль виконавця моделюються у вигляді векторів характеристик, елементи яких відображають навички, компетенції, категорії завдань та інші релевантні ознаки. Такий підхід дозволяє сформувати список релевантних розробників, які мають досвід або відповідні компетенції у виконанні подібних завдань, для подальшого обчислення показника релевантності виконавця (ПРВ) для кожного виконавця.

У межах кваліфікаційної роботи розроблено модель оцінки релевантності виконавця, що враховує як історичні показники, так і навантаження на працівника.

В основу математичної моделі покладено обчислення показника релевантності виконавця (РВ), який визначає рівень відповідності виконавця конкретному завданню. Показник релевантності виконавця — це показник, який відображає відповідність виконавця певному завданню з точки зору досвіду, універсальності, середнього часу виконання, а також поточного стану завантаженості виконавця.

Обчислення відбувається на основі чотирьох критеріїв (C_1, C_2, C_3, C_4), а саме: показника досвіду виконання подібних завдань, показника універсальності досвіду розробника, середній час виконання завдань та робоча завантаженість навантаження.

Показник релевантності виконавця розраховується, як зважена сума чотирьох критеріїв:

$$S=\alpha_1 C_1+\alpha_2 C_2+\alpha_3 C_3+\alpha_4 C_4\ , \qquad (2.1)$$

де C_1– показник досвіду у подібних типах завдань;

C_2 – універсальність виконавця;

C_3 – середній час виконання завдань;

C_4 – робоча завантаженість виконавця;

$\alpha_1, \alpha_2, \alpha_3, \alpha_4$ – вагові коефіцієнти.

В даному випадку чотири критерії множаться на ваговий коефіцієнт □□для розрахунку ПРВ, де значення параметрів є показником важливості чотирьох знак. Параметр □□поділяється на чотири ваги таким чином, що їхня сума має бути рівною одиниці. Ваги визначені експертним методом на основі опитування осіб, які володіють практичним досвідом в розробці програмних систем або управлінні командами в межах проєктно-орієнтованих підприємств.

1. Критерій C_1 - Показник досвіду виконання подібних завдань

Даний критерій показує досвіду виконавця [29], та обчислюється, як:

$$C_1=\frac{N_{type}}{N_{total}}, \qquad (2.2)$$

де N_{type} - кількість завдань певного типу,

N_{total} - загальна кількість виконаних завдань певного типу.

2. Критерій C_2- Показник універсальності виконавця [29].

Даний критерій показує спеціалізацію розробника [29] на основі завдань, які він виконував у минулому.

Значення обчислюється за формулою:

$$C_2=-\sum_{j=1}^{k} p_j \log\left(p_j\right), \qquad (2.3)$$

де k — кількість різних типів завдань, з якими працював виконавець,

p_j — частка завдань типу j, що буде розраховуватися за тією самою

формулою (2.2), за якою був знайдений показник C_1.

Для забезпечення порівнянності результатів показник нормалізується:

$$C_2^{norm} = \frac{C_2}{\log(k_{max})}, \quad (2.4)$$

де k_{max}— загальна кількість типів завдань у системі.

3. Критерій C_3 — середній час виконання завдань.

Цей показник визначає тривалість виконання завдань працівником і розраховується як співвідношення загального часу, витраченого на вирішення завдань та загальної кількості вирішених помилок розробником.

Обчислюється за наступною формулою:

$$C_3 = \frac{1}{n}\sum_{i=1}^{n} t_j, \quad (2.5)$$

де t_j — фактичний час виконання j-го завдання,

n — загальна кількість виконаних завдань виконавцем.

Для забезпечення порівнянності результатів показник нормалізується:

$$C_3^{norm} = 1 - \frac{C_3}{H_{max}}, \quad (2.6)$$

де H_{max} — максимальний час серед всіх виконаних завдань.

4. Критерій C_4— робоча завантаженість виконавця

Цей показник дозволяє системі бути динамічною та здатною враховувати поточний стан виконавця при прийнятті рішення про призначення завдання. Критерій C_4 визначає ступінь робочої доступності виконавця та обчислюється за формулою:

$$C_4 = 1 - L_{curr}, \quad (2.7)$$

де L_{curr}- це показник рівня завантаженості, що розраховується як співвідношення сумарної ваги всіх активних завдань, які виконує працівник, до максимально допустимого обсягу навантаження для даного працівника.

Він розраховується за наступною формулою:

$$L_{curr} = \sum_{i=1}^{k} \frac{w_j}{W_{max}}, \tag{2.8}$$

де L_{curr}- вага (складність) і-го активного завдання;

k — кількість активних завдань;

W_{max}— максимально допустиме навантаження виконавця.

Оскільки L_{curr} може перевищувати одиницю у разі перевантаження, застосовується обмеження:

$$L_{curr} := (1, L_{curr}), \tag{2.9}$$

Величина W_{max} є важливим елементом та визначається за формулою:

$$W_{max} = H_{work} * (1 - r_{meet} - r_{other}), \tag{2.10}$$

де H_{work} — середня кількість робочих годин виконавця за тиждень;

r_{meet}— частка часу, що витрачається на зустрічі;

r_{other} — частка часу, що витрачається на допоміжні завдання.

2.2 Елемент пояснювальної рекомендації

Більшість сучасних систем рекомендацій функціонують як «чорні скриньки»: вони надають результат без пояснення логіки прийняття рішень. У контексті управління проєктами це є суттєвим обмеженням, оскільки керівник повинен розуміти, на підставі яких критеріїв система рекомендувала конкретного

виконавця для виконання завдання. Відсутність пояснень може знижувати довіру до системи та ускладнювати прийняття обґрунтованих управлінських рішень.

У запропонованій системі реалізовано підхід пояснювальної рекомендації. Метою цього підходу є не лише призначення виконавця для викоання завдання, а й формування пояснення вибору на основі показника релевантності виконавця (ПРВ). Система відображає, які фактори (досвід, універсальність навичок, середній час виконання завдань, поточна завантаженість) мали найбільший вплив на ранжування виконавців.

Наприклад, для завдання «Розробка модулю обліку клієнтських даних» система може запропонувати виконавця А, як найрелевантнішого, вказавши, що основними перевагами виконавця А над іншими стали, наприклад, високі показники досвіду в схожих завданнях, низький середній час виконання завдань, а також оптимальна поточна завантаженість.

Такий підхід підвищує прозорість прийняття рішень, збільшує довіру користувачів до системи та сприяє більшій прийнятності рекомендацій у командному середовищі. Крім того, він забезпечує можливість корекції: якщо управлінець не згоден із запропонованою рекомендацією, він може відкоригувати вибір, орієнтуючись на фактори, що мали більший вплив на ранжування, і таким чином адаптувати рішення до специфіки завдання проєкту.

2.3 Інформаційне забезпечення

Для коректної роботи варто сформувати набір вхідних даних. Ці дані використовуються для розрахунку показника оцінки релевантності, який дозволить визначити найбільш доцільного виконавця для кожного завдання з урахуванням його досвіду, універсальності, середнього часу виконання завдань та поточного навантаження. Всі дані системи представлені у вигляді трьох наборів даних: “Виконавці”, “Завдання” та “Історія”. Всі таблиці будуть представлені у форматі csv. Така структура забезпечує простоту інтеграції системи з іншими джерелами даних (наприклад, корпоративними системами управління завданнями

— Jira, Trello, Asana) та дозволяє швидко оновлювати значення показників у реальному часі.

Таблиця 2.1 із даними про виконавців містить інформацію про кожного члена команди, яка необхідні для розрахунку індивідуальних показників досвіду, універсальності, тривалості виконання завдання та навантаження.

Кожен рядок відповідає одному виконавцю і містить поля:

Таблиця 2.1 – Поля набору даних “Виконавці”

Поле	Опис
user_id	Унікальний ідентифікатор працівника
nickname	Ім’я або нік виконавця
H_work	Середня кількість робочих годин на тиждень
r_meet	Частка часу, що витрачається на зустрічі
r_other	Частка часу, що витрачається на допоміжні задачі
avg_task_time	Тривалість виконання одного завдання
skill_tags	Перелік основних навичок виконавця

В даному випадку дані про середню кількість робочих годин на тиждень (H_work), Частка часу, що витрачається на зустрічі (r_meet), Частка часу, що витрачається на допоміжні задачі (r_other) використовуються для розрахунку максимально допустимого навантаження (Wmax). Водночас середній час виконання одного завдання (avg_task_time) використовується для критерію C_3 (тривалість виконання), а перелік основних навичок (skill_tags) допоможе для аналізу збігу навичок із завданнями при розрахунку C_1 та C_2.

Таблиця 2.2 із даними про завдання містить інформацію про всі всі активні та завершені завдання, які можуть бути використані системою для формування рекомендацій. Кожне завдання описується набором характеристик, що

дозволяють оцінити його тип, складність і поточний стан. Вона містить наступні поля:

Таблиця 2.2 – Поля набору даних “Завдання”

Поле	**Опис**
task_id	Унікальний ідентифікатор завдання
title	Назва завдання
type	Тип або категорія завдання (наприклад, frontend, backend, ai)
complexity	Складність завдання
assigned_to	Ідентифікатор працівника, якому було або буде призначено завдання
status	Поточний стан завдання (open, in_progress, done)
skills_required	Перелік необхідних навичок для виконання завдання

Значення категорії завдання (type) та переліку необхідних навичок (skills_required) використовуються для обчислення показників C_1 (досвід у подібних завданнях) та C_2 (універсальність). Крім того, показники складності (complexity), ідентифікатор відповідального за завдання працівника (assigned_to) і та статус завдання (status) будуть важливими для обчислення C_4(завантаженість виконавця).

Таблиця 2.3 з історією виконання завдань містить інформацію про виконані завдання, яка використовується для аналізу минулого досвіду, універсальності та середнього часу роботи виконавців. Як тільки завдання у таблиці із даними про завдання отримує статус “Готово” - воно приходить в дану таблицю. В таблиці історією виконання завдань містяться такі поля та опис:

Таблиця 2.3 – Поля набору даних “Історія”

Поле	Опис
user_id	Ідентифікатор працівника,, який виконав завдання
task_id	Ідентифікатор завдання
type	Тип завдання (frontend, backend, data тощо)
complexity	Вага завдання
time_spent	Фактичний час, витрачений на виконання
skills_required	Перелік необхідних навичок для виконання завдання

Значення типу завдання (type) використовується для оцінки досвіду виконавця в певній категорії задач (C_1), уся історія типів завдань для розрахунку універсальності (C_2), а затрачений час (time_spent) для визначення середнього часу виконання завдань (C_4).

Висновки до розділу 2

1. У другому розділі було розроблено та обґрунтовано алгоритмічну й математичну основу функціонування інтелектуальної рекомендаційної системи для призначення завдань членам команди. Визначено ключові параметри, що впливають на релевантність виконавця, та сформовано модель багатокритеріального оцінювання, що базується на показнику релевантності виконавця (ПРВ). Докладно описано методи нормалізації та зважування критеріїв, що дають змогу забезпечити об’єктивність та адаптивність процесу ранжування кандидатів.

2. Також було сформовано інформаційне забезпечення системи, включно зі структурою основних датасетів: даних про виконавців, їхні навички, історію

виконання завдань та характеристики самих завдань. Визначено логічні зв’язки між таблицями, а також механізми інтеграції з алгоритмічною частиною.

3. Проведене у розділі опрацювання створює цілісну методологічну основу для подальшої програмної реалізації системи. Сформовані моделі, алгоритми та інформаційні структури є ключовими компонентами, що забезпечують точність, інтерпретованість і адаптивність рекомендацій. Отримані результати дозволяють перейти до практичної реалізації програмного забезпечення та інтеграції розроблених компонентів у робочий веб-застосунок, що буде розглянуто у третьому розділі.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОЄКТУ РОЗРОБЛЕНОГО МЕТОДУ

3.1 Опис та аналіз програмних засобів для розробки системи

Розробка інтелектуальної рекомендаційної системи вимагає використання сучасних технологій, здатних забезпечити ефективну роботу з великими обсягами даних та можливість подальшого масштабування системи відповідно до зростання кількості виконавців і завдань. Вибір програмних інструментів здійснювався на основі аналізу їх функціональних можливостей, продуктивності, підтримки спільноти та відповідності вимогам поставленої задачі. За результатами цього аналізу було визначено набір технологій, що оптимально підходять для побудови системи рекомендацій у галузі управління завданнями.

Мову програмування Python обрано як основний засіб реалізації. Це рішення обумовлене універсальністю мови, простотою синтаксису, високою читабельністю коду та широкою підтримкою інструментів для роботи зі штучним інтелектом. Python має одну з найбільш розвинених екосистем бібліотек для обробки даних, статистичного аналізу, розроблення моделей машинного навчання та інтеграції різноманітних алгоритмів. Завдяки цьому Python забезпечує як швидкість розробки, так і гнучкість подальшої адаптації системи під нові вимоги.

Для реалізації функціоналу рекомендаційної системи використано низку спеціалізованих бібліотек. Зокрема, Pandas застосовується для обробки, структурування та аналізу даних, включаючи читання наборів даних, їх очищення, перетворення, фільтрацію та агрегацію [30]. Бібліотека надає високопродуктивні засоби маніпуляції табличними структурами, що є важливим для формування масивів даних виконавців, завдань та історії їх виконання.

Комплекс бібліотек Scikit-learn, що також включає інструменти з пакета SciPy [31], використовується для реалізації математичного апарату системи. Зокрема, засоби Scikit-learn використовувались для перетворення наборів текстових або категоріальних даних у числові вектори, обчислення косинусу схожості між наборами навичок виконавців і вимогами завдань, а також побудови функції оцінювання релевантності, яка формує основний рекомендаційний

рейтинг. Ці інструменти забезпечують стабільність розрахунків, а також створюють основу для подальшого розширення системи алгоритмами машинного навчання або гібридними моделями рекомендацій.

Для забезпечення взаємодії користувача з системою було розроблено веб-застосунок на основі фреймворку Flask. Він є гнучким та розширюваним для створення веб-інтерфейсів, що дозволяє швидко розгортати серверні модулі, обробляти запити, керувати маршрутизацією та передавати дані до шаблонів інтерфейсу. Завдяки інтеграції шаблонізатора та підтримці механізмів обробки запитів користувачів, Flask забезпечує зручність представлення результатів обчислень, а також надає можливість створити інтуїтивний та адаптивний інтерфейс для кінцевого користувача.

У комплексі описані технології створюють надійну технологічну основу для функціонування інтелектуальної рекомендаційної системи, забезпечуючи як обчислювальну ефективність, так і зручність взаємодії з користувачем.

3.2 Програмна складова системи

Для забезпечення зручної та інтуїтивно зрозумілої взаємодії користувача із системою було розроблено веб-інтерфейс, реалізований на основі фреймворку Flask. Інтерфейс виступає проміжною ланкою між користувачем та функціональними модулями системи, забезпечуючи доступ до основних можливостей, відображення результатів обчислень та здійснення керування даними.

Під час проєктування інтерфейсу користувача (UI) було враховано принципи зручності використання (Usability Principles), зокрема: мінімізацію кількості дій до виконання ключових операцій, логічну структуризацію елементів, візуальну однорідність та адаптивність [32]. Особлива увага приділялась зрозумілій навігації та можливості швидко отримувати зворотний зв’язок від системи, що є важливим у контексті роботи з рекомендаційним модулем.

Після завантаження веб-сторінки користувачеві насамперед пропонується

обрати конкретне завдання із наявного переліку. Додатково представлено підсекцію, у якій можна задати вагові коефіцієнти для ключових показників, що використовуються у процесі розрахунку рейтингу виконавців (рисунок 3.1). Зокрема, користувач має можливість ввести коефіцієнти для таких метрик, як досвід, універсальність, середній час виконання завдань та індекс завантаженості. Після заповнення всіх необхідних параметрів стає доступною опція переходу до наступного етапу шляхом натискання кнопки «Ранжувати виконавців».

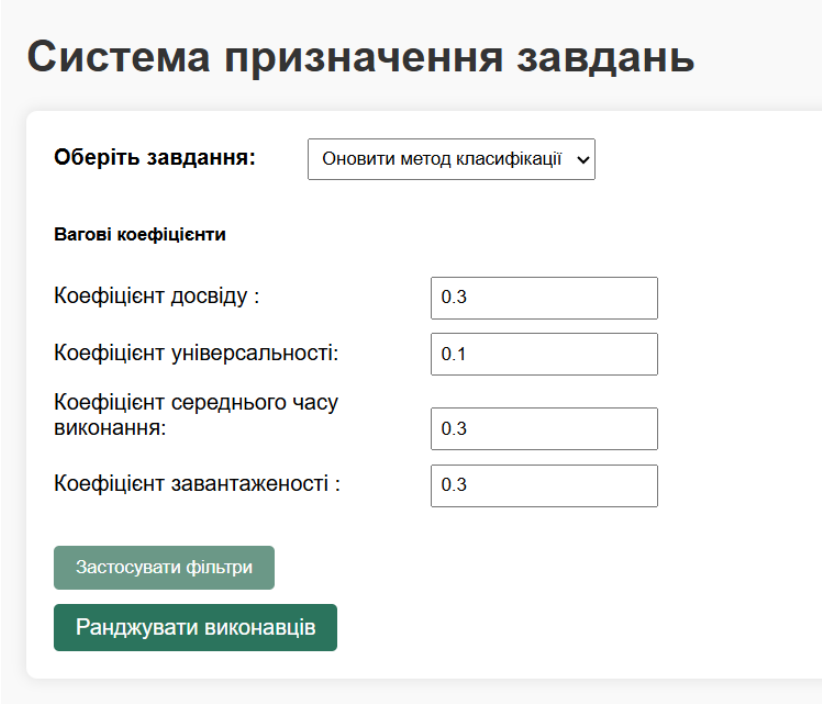


Рисунок 3.1 – Початкова сторінка для взаємодії із системою

Оскільки зазначені коефіцієнти мають істотний вплив на результати обчислення та формування кінцевих рекомендацій, система здійснює автоматичну валідацію їхніх значень. Якщо сума вагових коефіцієнтів не дорівнює одиниці, користувач отримує відповідне повідомлення про помилку, а подальше виконання операції блокується до моменту коректного виправлення введених даних (рисунок 3.2). Такий підхід забезпечує коректність математичної моделі та мінімізує ризик отримання хибних результатів.

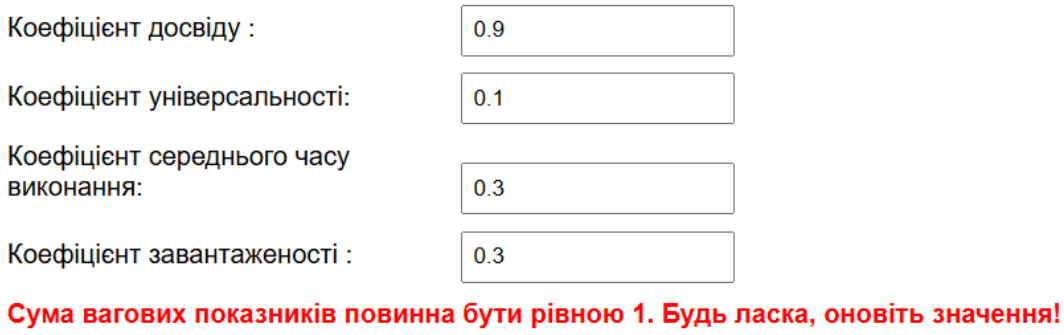


Рисунок 3.2 – Валідація даних

Крім того, користувач має можливість застосувати додаткові фільтри, натиснувши кнопку «Застосувати фільтри». Після активації цього режиму відкривається інтерфейс для налаштування параметрів «Мінімальна завантаженість», «Максимальна завантаженість» та «Фільтрування за навичкою» (рисунок 3.3).

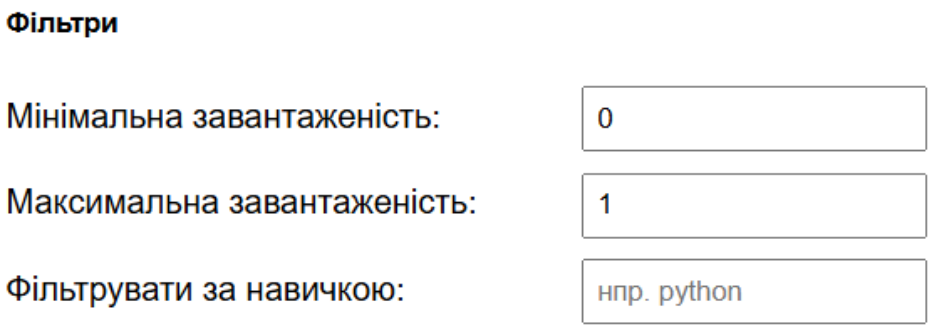


Рисунок 3.3 – Вкладка “Фільтри”

Наприклад, якщо користувачу необхідно сформувати перелік виконавців, чий рівень завантаженості відповідає певному інтервалу, інтерфейс дозволяє здійснити таке обмеження швидко і зручно. Аналогічно, у випадку, коли певна навичка вважається критично важливою для конкретного завдання, користувач може скористатися параметром «Фільтрування за навичкою», що дає змогу виключити кандидатів, які не володіють необхідною компетенцією (рисунок 3.4).

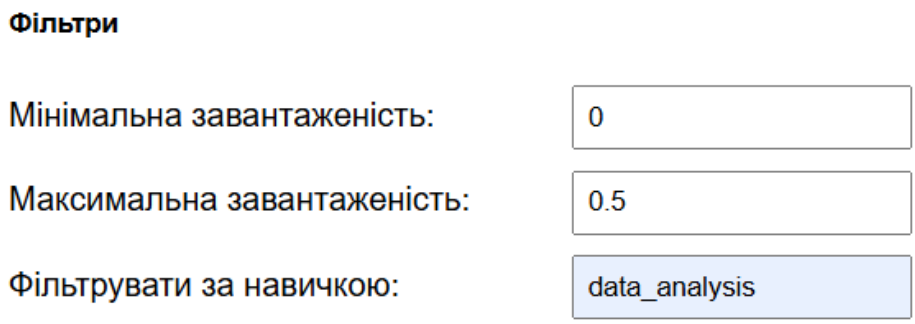


Рисунок 3.4 – Приклад встановлених фільтрів

Після активації відповідної кнопки запускається комплексний процес обчислення показника релевантності виконавця, застосування фільтраційних правил та формування впорядкованого списку кандидатів. Після завершення цих процедур користувачеві відображається ім’я найбільш релевантного виконавця, виділене збільшеним шрифтом з метою акцентування уваги. Додатково представлено пояснювальну інформацію, наприклад, ступінь відповідності навичок кандидата вимогам конкретного завдання, виражений у відсотковому співвідношенні (рисунок 3.5).

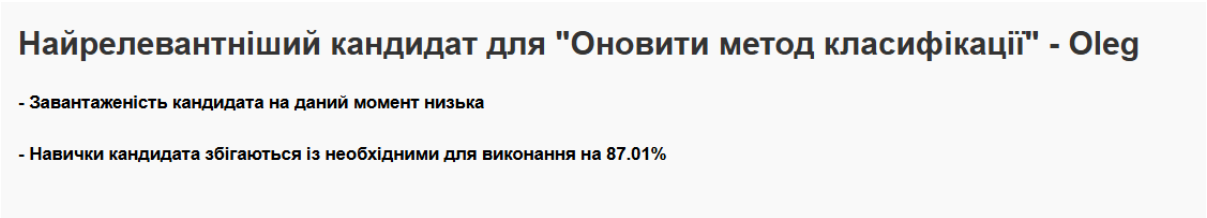
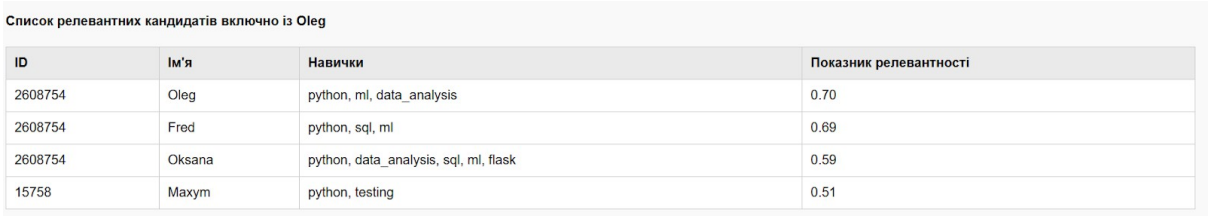


Рисунок 3.5 – Показ та пояснення найрелевантнішого кандидата

Після основного блоку пояснення формується розширений список усіх релевантних виконавців, включаючи кандидата з найвищим показником (рисунок 3.6). Система автоматично впорядковує цей список за спаданням у рейтингу, що забезпечує пріоритетне відображення найбільш доцільних кандидатів. Одночасно алгоритм виключає зі списку тих виконавців, чий показник релевантності є нижчим за встановлений поріг (0.50). Такий підхід дозволяє уникнути

інформаційного перевантаження та зосередити увагу користувача на найбільш перспективних кандидатах.



Список релевантних кандидатів включно із Oleg

ID	Ім'я	Навички	Показник релевантності
2608754	Oleg	python, ml, data_analysis	0.70
2608754	Fred	python, sql, ml	0.69
2608754	Oksana	python, data_analysis, sql, ml, flask	0.59
15758	Maxym	python, testing	0.51

Рисунок 3.6 – Список релевантних кандидатів

У нижній частині UI демонструється оновлений список кандидатів після застосування додаткових фільтрів, наприклад фільтрування за навичкою «data_analysis» (рисунок 3.7). У результаті обсяг списку значно скорочується, однак у ньому залишаються лише ті виконавці, які повністю відповідають обраним критеріям та вимогам конкретного завдання. Це забезпечує підвищену точність рекомендацій і дає змогу користувачу оперативно обрати оптимального виконавця.

Список релевантних кандидатів включно із Oleg

ID	Ім'я	Навички	Показник релевантності
2608754	Oleg	python, ml, data_analysis	0.70
2608754	Oksana	python, data_analysis, sql, ml, flask	0.59

Рисунок 3.7 – Відфільтрований список кандидатів

3.3 Сценарії роботи користувача із системою

Після запуску програмного забезпечення користувач отримує доступ до інтерфейсу, який забезпечує взаємодію з функціональними модулями системи рекомендацій. На початковому етапі користувачу пропонується підготований перелік завдань, що дає змогу швидко розпочати роботу без необхідності попереднього ручного пошуку. Обравши конкретне завдання та використавши базові (попередньо визначені) значення вагових коефіцієнтів і фільтраційних

параметрів, користувач може ініціювати процес ранжування виконавців шляхом натискання кнопки «Ранжувати виконавців». У відповідь система генерує впорядкований список кандидатів відповідно до обчисленої міри релевантності, а також інформацію про найкращого з них.

Для користувачів, які прагнуть здійснювати більш гнучке й тонке налаштування процесу оцінювання, у секції «Вагові коефіцієнти» передбачена можливість модифікації значень відповідних параметрів. Важливою умовою коректності обчислень є те, що сума всіх заданих коефіцієнтів має дорівнювати одиниці. У випадку порушення цього обмеження - система автоматично формує повідомлення про помилку, що унеможливлює запуск механізму ранжування до моменту встановлення користувачем валідних значень. Такий підхід забезпечує захист від помилкових налаштувань та підвищує надійність отриманих рекомендацій.

Окремий модуль «Фільтри» дозволяє здійснювати додаткове звуження списку потенційних виконавців. Після натискання кнопки «Застосувати фільтри» користувачу відкривається панель налаштувань, що забезпечує можливість відбору кандидатів за рівнем їхньої поточної завантаженості, а також за наявністю конкретної навички, релевантної вимогам обраного завдання. Це сприяє формуванню більш таргетованих рекомендацій і підвищує практичну цінність системи.

Після завершення процедури ранжування система окремо виділяє кандидата з найвищим показником релевантності. Користувачу надається детальне пояснення, чому саме цей виконавець є найбільш доцільним, що підвищує прозорість роботи алгоритмів і довіру до системи. Після цього відображається впорядкований список інших можливих виконавців, які мають нижчі значення релевантності, але все ще можуть розглядатися як потенційні кандидати. Такий підхід забезпечує комплексне представлення результатів та підтримує користувача у прийнятті обґрунтованого рішення керівника.

3.4 Результати тестування системи

Тестування розробленої рекомендаційної системи є ключовим етапом оцінювання її функціональних характеристик, точності, надійності та відповідності поставленим вимогам. Метою тестування є перевірка коректності роботи алгоритмів обчислення показника релевантності кандидата, відповідності рекомендацій очікуванням, а також оцінювання зручності використання інтерфейсу веб-застосунку.

У процесі тестування було здійснено перевірку коректності функціонування елементів веб-інтерфейсу та механізмів валідації введених даних. Зокрема, встановлено, що система коректно блокує виконання операції ранжування виконавців у випадках, коли сума вагових коефіцієнтів не дорівнює одиниці, а також у разі, якщо введені користувачем числові значення фільтрів виходять за допустимі межі інтервалу [0;1]. Це засвідчує наявність надійного контролю цілісності та валідності даних, що мінімізують ризик некоректної роботи алгоритмічного модуля системи.

Тестування алгоритмічної частини, включало оцінку правильності обчислення проміжних показників — зокрема, досвіду виконавця, універсальності навичок, середнього часу виконання завдань та рівня завантаженості, а також показника ПРВ. Встановлено, що система коректно розраховує всі згадані значення.

Окремо було здійснено експертне тестування за участю фахівців у галузі управління проєктами та розробки ПЗ. Було оцінено запропоновані системою рекомендації, а також обґрунтованість пояснень, які представлені елементом пояснювальної рекомендації. Це дозволило встановити відповідність результатів реальним управлінським сценаріям.

Таким чином, проведена верифікація охопила як обчислювальні характеристики, так і поведінку системи. Результати тестування підтвердили, що система стабільно функціонує, забезпечує коректне ранжування виконавців, а також пропонує обґрунтовані рекомендації.

Висновки до розділу 3

1. У третьому розділі було здійснено комплексний опис та аналіз програмних засобів, використаних для реалізації інтелектуальної рекомендаційної системи призначення завдань членам команди. На основі проведеного порівняльного аналізу було обґрунтовано вибір мови програмування Python, бібліотек для обробки даних та побудови моделей (Pandas, NumPy, Scikit-learn), а також веб-фреймворку Flask, що забезпечує створення зручного та функціонального інтерфейсу користувача. Вибраний технологічний стек продемонстрував відповідність вимогам щодо масштабованості, продуктивності та гнучкості системи.

2. У межах програмного компонента було реалізовано ключовий функціонал системи: обробку та валідацію даних, застосування вагових коефіцієнтів і фільтрів, а також ранжування виконавців на основі багатокритеріальної оцінки.

3. Окрему увагу приділено реалізації компонента пояснювальної рекомендації, що формує ясне та обґрунтоване пояснення вибору найкращого кандидата для конкретного завдання. Це забезпечує інтерпретованість результатів та підвищує рівень прозорості системи для кінцевого користувача.

4. Також було розроблено та проаналізовано сценарії роботи користувача із системою, які відображають типові кроки взаємодії з веб-застосунком — від вибору завдання та встановлення вагових коефіцієнтів до застосування фільтрів і отримання повного списку ранжованих виконавців із поясненнями. Представлені сценарії підтверджують інтуїтивність інтерфейсу, логічність послідовності дій та зручність роботи з системою в умовах реальних управлінських процесів.

5. Загалом, третій розділ демонструє практичну реалізацію розроблених теоретичних та математичних положень, підтверджуючи доцільність обраних технологій та ефективність створених програмних компонентів. Реалізоване програмне забезпечення забезпечує стабільну роботу, адаптивність до різних видів завдань та надає користувачу інструмент для обґрунтованого прийняття

рішень щодо призначення виконавців. Розроблена система є готовою до подальшого тестування, оптимізації та інтеграції в комплексні системи управління проєктами.

ВИСНОВКИ

В межах даної кваліфікаційної роботи було здійснено комплексне дослідження, спрямоване на розроблення інтелектуальної рекомендаційної системи з елементами пояснювальної рекомендації та підтримки прийняття рішень.

Відповідно до поставлених у вступі завдань у магістерській роботі отримано такі результати:

1. Було сформульовано та обґрунтовано проблему з точки зору управління проєктами, визначивши ключові труднощі, що пов’язані із нераціональним призначенням або розподілом завдань між членами команди, а також обґрунтовано створення інтелектуальних інструментів підтримки прийняття рішень.

2. Проведено детальний аналіз методів рекомендаційних систем, включаючи контентно-орієнтовані, колаборативні, гібридні, а також методи їх валідації, що дозволило визначити підхід для розробки проєкту рекомендаційної системи.

3. Проведено огляд існуючих рішень у галузі управління проєктами, виявлено обмеження, зокрема недостатнє урахування індивідуальних характеристик виконавців та відсутність пояснювальних рекомендацій.

4. Розроблено алгоритм рекомендаційної системи, який враховує досвід, універсальність, середній час виконання завдань та поточну завантаженість виконавців, формуючи інтегральний показник релевантності виконавця.

5. Математичну складову системи описано шляхом формалізації всіх параметрів, що впливають на показник релевантності виконавця (ПРВ). Наведено формули для обчислення кожного з критеріїв, а також представлено використання косинусної подібності для формування списку релевантних виконавців.

6. Представлено інформаційне забезпечення системи та описано структури наборів даних для представлення інформації про виконавців, завдання та

історичні дані. Наведено використання полів для обчислень показників математичної моделі.

7. Розроблено програмне забезпечення, яке реалізує алгоритм рекомендацій у середовищі Python з використанням Pandas, NumPy та Flask, забезпечуючи як серверну логіку, так і веб-інтерфейс для взаємодії користувача.

8. Описано сценарії взаємодії користувача із системою, включаючи процес вибору завдання, зміну вагових коефіцієнтів (за потреби), застосування фільтрів, отримання ранжованого списку виконавців та перегляд пояснення рекомендацій.

Отримані результати підтверджують ефективність обраних підходів та можливість подальшого розвитку системи у напрямі масштабування, інтеграції нових методів штучного інтелекту та розширення функціоналу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bieliaiev, O. (2025). Overview of artificial intelligence-based project management tools for business representatives. Development Management, 24(3), 57-67. https://doi.org/10.63341/devt/3.2025.57

2. P. Oliveira, R. M. C. Andrade, I. Barreto, T. P. Nogueira and L. Morais Bueno, "Issue Auto-Assignment in Software Projects with Machine Learning Techniques," 2021 IEEE/ACM 8th International Workshop on Software Engineering Research and Industrial Practice (SER&IP), Madrid, Spain, 2021, pp. 65-72, doi: 10.1109/SER-IP52554.2021.00018.

3. What are recommender systems [Електронний ресурс]. – 2025. – Режим доступу: https://www.geeksforgeeks.org/machine-learning/what-are-recommender-systems/#types-of-recommendation-systems (дата звернення: 18.12.2025).

4. Murel J., Kavlakoglu E. What is collaborative filtering? [Електронний ресурс]. – Режим доступу: https://www.ibm.com/think/topics/collaborative-filtering (дата звернення: 18.12.2025).

5. Collaborative filtering summary [Електронний ресурс]. – Режим доступу: https://developers.google.com/machinelearning/recommendation/collaborative/summary (дата звернення: 18.12.2025).

6. Content-based recommendation basics [Електронний ресурс]. – Режим доступу: https://developers.google.com/machinelearning/recommendation/content-based/basics (дата звернення: 18.12.2025).

7. Murel J., Kavlakoglu E. What is content-based filtering? [Електронний ресурс]. – Режим доступу: https://www.ibm.com/think/topics/content-based-filtering (дата звернення: 18.12.2025).

8. What is hybrid filtering in recommender systems [Електронний ресурс]. – Режим доступу: https://milvus.io/ai-quick-reference/what-is-hybrid-filtering-in-recommender-systems (дата звернення: 18.12.2025)

9. What defines a hybrid recommender system and what are its benefits [Електронний ресурс]. – Режим доступу: https://milvus.io/ai-quick-reference/what-defines-a-hybrid-recommender-system-and-what-are-its-benefits (дата звернення: 18.12.2025).

10. 7 types of hybrid recommendation system [Електронний ресурс]. – Medium. Режим доступу: https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-system-3e4f78266ad8 (дата звернення: 18.12.2025).

11. Switching and mixed hybridization [Електронний ресурс]. – Building ML Recommendation System. – Режим доступу: https://apxml.com/courses/building-ml-recommendation-system/chapter-6-constructing-a-hybrid-recommendation-system/switching-and-mixed-hybridization (дата звернення: 18.12.2025).

12. Feature combination methods for hybrid recommendation [Електронний ресурс]. – Building ML Recommendation System. – Режим доступу: https://apxml.com/courses/building-ml-recommendation-system/chapter-6-constructing-a-hybrid-recommendation-system/feature-combination-methods (дата звернення: 18.12.2025).

13. Hybrid recommender systems [Електронний ресурс]. – Bluepiit.com. Режим доступу: https://www.bluepiit.com/blog/hybrid-recommender-systems (дата звернення: 18.12.2025).

14. Pullakandam K. Understanding precision, recall, and F-score at K in recommender systems [Електронний ресурс]. – 2024. – Режим доступу: https://krishnapullak.medium.com/understanding-precision-recall-and-f-score-at-k-in-recommender-systems-7146a0dce68e (дата звернення: 18.12.2025).

15. Sen S. Evaluating recommender systems [Електронний ресурс]. – 2020. – Режим доступу: https://medium.com/the-owl/evaluating-recommender-systems-749570354976 (дата звернення: 18.12.2025).

16. 10 metrics to evaluate recommender and ranking systems [Електронний ресурс]. – 2025. – Режим доступу: https://www.evidentlyai.com/ranking-metrics/evaluating-recommender-systems#ranking-quality-with-evidently (дата звернення: 18.12.2025).

17. Evaluating model performance: understanding MAE, MSE, RMSE, and R^2 score [Електронний ресурс]. – Режим доступу: https://medium.com/@srivastavashivansh8922/evaluating-model-performance-understanding-mae-mse-rmse-and-r%C2%B2-score-5516e10fe8d6 (дата звернення: 18.12.2025).

18. Understanding MAE, MSE, and RMSE: key metrics in machine learning [Електронний ресурс]. – Режим доступу: https://dev.to/mondal_sabbha/understanding-mae-mse-and-rmse-key-metrics-in-machine-learning-4la2 (дата звернення: 18.12.2025).

19. Precision and recall at K in ranking and recommendations [Електронний ресурс]. – 2025. – Режим доступу: https://www.evidentlyai.com/ranking-metrics/precision-recall-at-k (дата звернення: 18.12.2025).

20. What does the retrieval metric “precision@K” tell us about the top-K documents returned, and why might a high precision@3 be critical for the subsequent generation step? [Електронний ресурс]. – Режим доступу: https://milvus.io/ai-quick-reference/what-does-the-retrieval-metric-precisionk-tell-us-about-the-topk-documents-returned-and-why-might-a-high-precision3-be-critical-for-the-subsequent-generation-step (дата звернення: 18.12.2025).

21. Aktas, E.U., & Yilmaz, C. (2020). Automated issue assignment: results and insights from an industrial case. Empirical Software Engineering, 25, 3544 - 3589.

22. Chhabra, D., & Chadha, D.R. (2024). Machine Learning Approaches in Contemporary Automatic Bug Triaging and Analysis of Research Gaps. 2024 Sixth International Conference on Computational Intelligence and Communication Technologies (CCICT), 583-588.

23. Dedik, Vaclav and Bruno Rossi. “Automated Bug Triaging in an Industrial Context.” 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (2016): 363-367.

24. Liang Wei and Luiz Fernando Capretz. 2021. Recommender Systems for Software Project Managers. In Proceedings of the 25th International Conference on Evaluation and Assessment in Software Engineering (EASE '21). Association for

Computing Machinery, New York, NY, USA, 412–417. https://doi.org/10.1145/3463274.3463951

25. Bavota, Gabriele, et al. "Recommending refactoring operations in large software systems." Recommendation systems in software engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. 387-419.

26. T. Zhan, B. Lee,An automated bug triage approach: a concept profile and social network based developer recommendation, in: Intelligent Computing Technology. ICIC 2012. Lecture Notes in Computer Science, Springer, Berlin, Heidelber., 2012, URL: pp. 505–512, doi: 10.1007/978-3-642-31588-6_65

27. The standard for project management and a guide to the project management body of knowledge (PMBOK guide). (2021). Project Management Institute, Inc.

28. Krantz T., Jonker A. What is cosine similarity? [Електронний ресурс]. – Режим доступу: https://www.ibm.com/think/topics/cosine-similarity (дата звернення: 18.12.2025).

29. Yadav, Asmita & Kumar Singh, Sandeep & Suri, Jasjit. (2019). Ranking of Software Developers based on Expertise Score for Bug Triaging. Information and Software Technology. 112. 10.1016/j.infsof.2019.03.014.

30. The pandas development team. pandas-dev/pandas: Pandas (v2.3.3) [Електронний ресурс]. – 2025. – Zenodo. – Режим доступу: https://doi.org/10.5281/zenodo.17229934 (дата звернення: 18.12.2025).

31. Virtanen P., Gommers R., Oliphant T. E. et al. SciPy 1.0: fundamental algorithms for scientific computing in Python [Електронний ресурс]. – Nat Methods, 17, 261–272 (2020). – Режим доступу: https://doi.org/10.1038/s41592-019-0686-2 (дата звернення: 18.12.2025).

32. Usability principles [Електронний ресурс]. – Режим доступу: https://improvement.stanford.edu/resources/usability-principles (дата звернення: 18.12.2025).

33. Комар М.П., Саченко А.О., Васильків Н.М., Загородня Д.І. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної

програми «Комп’ютерні науки» спеціальності 122 «Комп’ютерні науки» за другим (магістерським) рівнем вищої освіти. – Тернопіль: ЗУНУ, 2024. – 32 с.

34. Мельник А., Ліп’яніна-Гончаренко Х. В. Рекомендаційна система для призначення завдань членам команди на основі штучного інтелекту // Інтелектуальні інформаційні технології в прикладних дослідженнях : збірник тез доповідей студентської науково-практичної конференції (IITAR-2025). – 2025. – С. 355–358.

35. Мельник А., Лендюк Т. В. Проєкт рекомендаційної системи для призначення завдань членам команди // Наука, освіта, економіка та суспільство: адаптація до викликів XXI століття : збірник тез доповідей міжнародної науково-практичної конференції. – 2025. – С. 213–217.

ДОДАТОК А

Програмний код

```
import math
from flask import Flask, render_template, request
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import pandas as pd

app = Flask(__name__)

# Load data
users = pd.read_csv("users.csv")
tasks = pd.read_csv("tasks.csv")
history = pd.read_csv("history.csv")

@app.route("/", methods=["GET", "POST"])
def index():
    selected_task = None
    ranking = None
    top_user = None
    error_message = None
    # workload_type = None

    if request.method == "POST":
        task_id = request.form["task_id"]
        min_workload = float(request.form.get("min_workload", 0))
        max_workload = float(request.form.get("max_workload", 1))
        skill_filter = request.form.get("skill_tag", "").strip().lower()
        alpha = float(request.form.get("alpha", 0.35)) * 0.5
        beta = float(request.form.get("beta", 0.15)) * 0.5
```

```
        gamma = float(request.form.get("gamma", 0.25)) * 0.5
        delta = float(request.form.get("delta", 0.25)) * 0.5
        theta = 0.5
        weight_sum = theta + alpha + beta + gamma + delta

        if abs(weight_sum - 1.0) > 0.01:  # Allow tiny rounding error
            error_message = "Сума вагових показників повинна бути рівною 1. Будь ласка, оновіть значення!"

        task = tasks[tasks["task_id"] == task_id].iloc[0]

        # Skill similarity
        vectorizer = CountVectorizer(tokenizer=lambda x: x.split(","))
        skill_matrix = vectorizer.fit_transform(list(users["skills"]) + [task["required_skills"]])
        task_vec = skill_matrix[-1]
        user_vecs = skill_matrix[:-1]
        similarities = cosine_similarity(user_vecs, task_vec.reshape(1, -1)).flatten()
        users["similarity"] = similarities

        # Apply filters
        filtered_users = users[
            (users["workload"] >= min_workload) &
            (users["workload"] <= max_workload)
        ]

        if skill_filter:
            filtered_users = filtered_users[filtered_users["skills"].str.contains(skill_filter, case=False)]
```

```
        #  alpha, beta, gamma = 0.6, 0.3, 0.1
        filtered_users = filtered_users.copy()
        filtered_users["TES"] = (
            alpha * filtered_users["exp"] +
            beta * filtered_users["performance"] +
            gamma * (1 - filtered_users["workload"]) +
            delta * (1 - filtered_users["avg_time"]) +
            theta * filtered_users["similarity"]
        )

        ranking = filtered_users.sort_values("TES", ascending=False)[["user_id",
"username", "skills", "TES", "workload", "similarity"]]

        selected_task = task.task_name
        if not ranking.empty:
            top_user = ranking.iloc[0]
            top_user.similarity = round(top_user.similarity, 2)
            if top_user.workload < 0.5:
                top_user["workload_type"] = "низька"
            elif top_user.workload >= 0.5 and top_user.workload <= 0.7:
                top_user["workload_type"] = "помірна"
            else:
                top_user["workload_type"] = "висока. Потенційно краще обрати
наступного кандидата"

    return render_template("index.html", tasks=tasks["task_id"],
tasksNames=tasks["task_name"],
                           ranking=ranking, selected_task=selected_task, top_user=top_user,
zip=zip, error_message=error_message)
    def filtered_users_by_similarity():
```

ДОДАТОК Б

Апробація отриманих результатів

Західноукраїнський національний університет

Факультет комп'ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління

ЗБІРНИК ТЕЗ ДОПОВІДЕЙ

Студентської науково-практичної конференції

ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРИКЛАДНИХ ДОСЛІДЖЕННЯХ

(IITAR-2025)

27-29 травня 2025 року

Тернопіль

2025

Мельник Анна
студентка групи КНУПм-11
melnikanna524@gmail.com
Ліп'яніна-Гончаренко Христина
д.т.н., доцент
kh.lipianina@wunu.edu.ua
Західноукраїнський національний університет
Тернопіль, Україна

РЕКОМЕНДАЦІЙНА СИСТЕМА ДЛЯ ПРИЗНАЧЕННЯ ЗАВДАНЬ ЧЛЕНАМ КОМАНДИ НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ

В умовах цифровізації бізнес-процесів та зростаючої складності проєктної діяльності особливої актуальності набуває питання ефективного управління командною роботою. Однією з ключових задач у цьому процесі є грамотний розподіл завдань між членами команди, оскільки це відіграє важливу роль у досягненні цілей проєкту. Традиційні методи призначення завдань, які базуються на суб'єктивній оцінці працівників, що відповідальні за призначення завдань або ж використанні простих правил, можуть не завжди виявитися ефективними, особливо у великих або динамічних проєктах, де потрібно враховувати значну кількість факторів: навички співробітників, їхній досвід, поточне завантаження, дедлайни, складність задач тощо. Саме тому метою роботи є створення рекомендаційної системи, що на основі методів штучного інтелекту дозволить автоматизувати процес призначення завдань, підвищити його об'єктивність та ефективність, а також враховувати навички, досвід, завантаженість та інші характеристики членів команди. Це в свою чергу допоможе прискорити процес розподілу завдань, уникнути перевантаження одних учасників та недовантаження інших, ефективніше використовувати сильні сторони кожного члена команди, а також покращить ефективність команди в цілому. Задачі роботи можна сформувати як:

1. Проаналізувати вже існуючі підходи до автоматизованого розподілу завдань у командах.
2. Визначити ключові параметри, що впливають на ефективність призначення завдань.
3. Розробити архітектуру рекомендаційної системи з використанням методів машинного навчання.

В сучасних дослідженнях розглядається рішення із впровадженням штучного інтелекту у процес призначення проектних завдань. Наприклад у дослідженні "Issue Auto-Assignment in Software Projects with Machine Learning Techniques" [1] було проведено дослідження на реальних даних з мобільного підрозділу компанії LG Electronics у одному з міст Бразилії. Автори розглядали призначення задач як задачу оптимізації та застосовували кілька алгоритмів машинного навчання серед яких були Naive Bayes, KNN, SVM, Random Forest,

навчаючи їх на історичних даних з реальних проектів вищезгаданої компанії. У дослідженні оцінювались точність призначень, швидкість роботи та зниження кількості помилок. В результаті найкращу ефективність показали ансамблеві моделі, зокрема Random Forest. Автоматичне призначення допомогло значно зменшити кількість неправильних розподілів і скоротити час обробки задач.

Ще одне дослідження [2] від авторів Етема Утку Акташа та Джемаля Їлмаза представили досвід впровадження системи автоматичного призначення завдань у великій комерційній організації — Softtech у Стамбулі, Туреччина. Організація стикнулась із проблемою дуже великого щоденного навантаження на команду, тому ручне призначення виявилось досить трудомістким та могло приводити до затримок у вирішенні критичних проблем. Дослідники розробили та впровадили систему IssueTAG, яка дозволила автоматизувати процес призначення завдань, використовуючи методи аналізу даних. Також результати досліджень показали підвищення точності призначень завдань, ефективності та зниження помилковості при визначенні. Крім того, перевагою дослідження стало те, що на відміну від попередніх досліджень, які здебільшого базувалися на відкритих проектах або проводилися ретроспективно, IssueTAG була впроваджена у реальному виробничому середовищі і з 12 січня 2018 року здійснює всі призначення завдань у Softtech.

В рамках даної роботи для побудови рекомендаційної системи буде використано гібридний підхід, який поєднуватиме в собі методи контентного фільтрування, алгоритмів машинного навчання та аналізу наявних даних. Почнемо із збору та підготовки поточних даних. На цьому етапі варто зібрати дані про учасників команди, їхні навички, досвід, роль у команді та завантаженість. Також потрібні дані про самі завдання, їхню складність, терміни, необхідні навички та історію виконання завдань. В даному випадку мається на увазі те, як приблизно аналогічні завдання виконував той чи інший учасник команди.

Після збору даних, вони пройдуть етап обробки та векторизації. Під цим мається на увазі перетворення у числові вектори якісні характеристики для можливості майбутніх обчислень. Після цього буде етап розробки моделі машинного навчання, де за основу буде взято метод класифікації Random Forest, щоб передбачити чи конкретний учасник команди зможе виконати завдання. Для оцінки буде використано такі метрики, як точність, середнє значення рейтингу відповідності та задоволеність користувачів. Як технічну основу буде використано Python та широкий набір бібліотек, серед яких наприклад pandas, numpy та інші. Рисунок 1 представляє схему послідовності етапів.

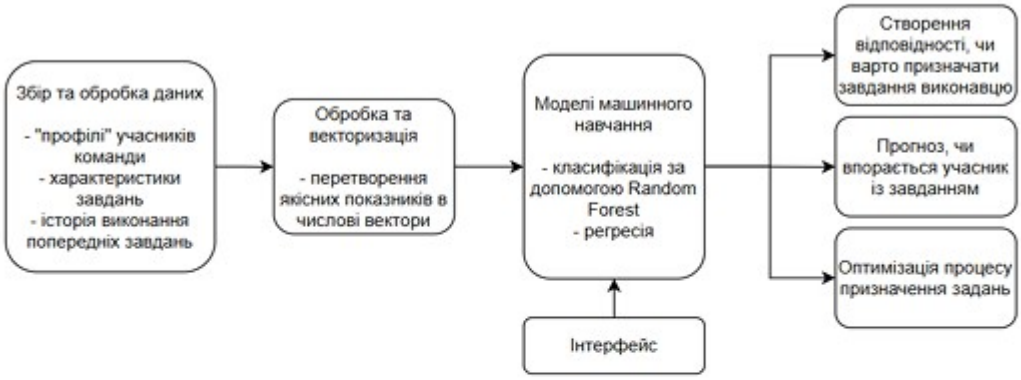


Рисунок 1. Схема послідовності етапів розробки

У межах дослідження було розроблено концепцію рекомендаційної системи для автоматизованого призначення завдань учасникам команди з використанням методів штучного інтелекту. В результаті було проведено аналіз предметної області, тобто визначено ключові фактори, що впливають на ефективність призначення завдань, а саме технічні навички виконавців, рівень досвіду, поточне завантаження, виконані раніше завдання, а також специфіка і складність самих задач. Визначено напрямок реалізації, а саме обрано класифікаційну модель на основі алгоритму Random Forest, яка визначає найдоцільнішого виконавця для кожного завдання.

Таблиця 1 – Очікуванні ефекти від впровадження ШІ в робочий процес команди.

Функція	**Очікуваний ефект**	**Приклад**
Призначення завдань	Модель успішно на основі попередніх даних рекомендує найкращого виконавця під конкретне завдання	Модель дозволила ефективно призначити завдання новому члену команди
Покращення точності	Модель допомагає точніше призначити завдання члену команди, який має для цього відповідну компетенцію і встановити терміни	Учасник отримав завдання, яке повністю сходиться з його досвідом і навичками, а також із правильним часовим обмеженням
Балансування навантаження	Дозволить не перенавантажувати когось одного і недовантажувати іншого	Всі учасники будуть мати збалансовану навантаженість
Зменшення часу на призначення завдань	Дозволить зменшити час керівника на призначення завдань та дозволить сконцентруватися на більш пріоритетних речах	Керівник зміг встигнути більше за робочий день

У ході дослідження було розглянуто актуальну проблему автоматизації процесу розподілу завдань у командній роботі за допомогою рекомендаційної системи, що базується на методах штучного інтелекту. Встановлено, що традиційні методи призначення завдань часто є неефективними в умовах багатофакторного середовища, де потрібно враховувати індивідуальні особливості виконавців, характеристики самих завдань тощо. Крім того, було розглянуто наявні дослідження та їхні результати, що дозволило підкреслити та проаналізувати досвід попередників.

Результати дослідження свідчать про доцільність впровадження інтелектуальних рекомендаційних систем у сферу управління проєктами. Запропоноване рішення може бути адаптоване до потреб різних команд і середовищ. Проте варто зазначити, що інтеграція такої системи може зайняти час, оскільки модель має навчитись на аналізувати дані під окрему команду та її особливості.

Список використаних джерел

1. Issue Auto-Assignment in Software Projects with Machine Learning Techniques [Електронний ресурс]. 2021. Режим доступу: https://ieeexplore.ieee.org/abstract/document/9474835
2. Automated Issue Assignment: Results and Insights from an Industrial Case [Електронний ресурс]. 2021. Режим доступу: https://www.researchgate.net/publication/350311744_Automated_Issue_Assignment_Results_and_Insights_from_an_Industrial_Case

CERTIFICATE OF PARTICIPATION

ICSSH

Serial № GQ/06-12-25/55

Мельник Анна Василівна

FOR PARTICIPATING INTERNATIONAL SCIENTIFIC AND PRACTICAL CONFERENCE **"SCIENCE, EDUCATION, ECONOMICS, AND SOCIETY: ADAPTING TO THE CHALLENGES OF THE 21ST CENTURY"**

Total number of academic hours: 24 (0.8 ECTS credits)

Director of the International Center of Social Sciences and Humanities
Miroslava Kocisova

INTERNATIONAL CENTER OF SOCIAL SCIENCES AND SCIENTIFIC CONFERENCE ICSSH

December 06, 2025
San Francisco, USA

ICSSH

INTERNATIONAL SCIENTIFIC AND PRACTICAL CONFERENCE

SCIENCE, EDUCATION, ECONOMICS, AND SOCIETY: ADAPTING TO THE CHALLENGES OF THE 21ST CENTURY

Book of abstracts

December 6, 2025

San Francisco,
USA

УДК 005.8:004.8

Мельник А. В.
здобувачка другого (магістерського) рівня вищої освіти
Західноукраїнський національний університет
м. Тернопіль
Лендюк Т. В.
к.т.н., доцент
доцент кафедри інформаційно-обчислювальних систем і управління,
Західноукраїнський національний університет
м. Тернопіль

ПРОЄКТ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ ДЛЯ ПРИЗНАЧЕННЯ ЗАВДАНЬ ЧЛЕНАМ КОМАНДИ

В межах команд проєктів конкретні завдання можуть бути делеговані окремим особам або обрані самими членами команди проєкту [1, с. 30]. У контексті цифрової трансформації проєктно-орієнтованих підприємств та зростання складності проєктної діяльності забезпечення раціонального розподілу завдань між членами команди набуває особливого значення. Традиційні підходи, що ґрунтуються переважно на суб'єктивних оцінках керівника проєкту, в умовах багатопроєктного управління виявляються недостатньо ефективними, оскільки не здатні повною мірою врахувати широкий спектр факторів, зокрема індивідуальні характеристики виконавців, їхні компетенції, поточне завантаження та динаміку розвитку навичок. Як наслідок, це може спричиняти дисбаланс у розподілі роботи, зниження загальної продуктивності та погіршення рівня задоволеності учасників команди.

Актуальність роботи обумовлена необхідністю створення інструментів у галузі управління проєктами, здатних забезпечити ефективне, збалансоване й обґрунтоване призначення завдань у проєктних командах за допомогою методів рекомендаційних систем.

Метою роботи є підвищення ефективності управління ресурсами проєкту шляхом розробки інтелектуальної рекомендаційної системи, що напівавтоматизує процес призначення завдань членам команди з урахуванням факторів, розрахованих під кожного виконавця. Зазвичай менеджери або технічні керівники в проектах розробки програмного забезпечення призначають задачі вручну [2, с. 65], тому такий підхід спрямований на оптимізацію розподілу ресурсів, зменшення ризику

нераціонального навантаження на членів команди проєкту, підвищення якості управлінських рішень та забезпечення більшої прозорості й обґрунтованості процесу делегування завдань у рамках проєктної діяльності.

Постановка завдання. Для досягнення поставленої мети було поставлено наступні завдання:

1. Розробити алгоритм рекомендаційної системи контентно-орієнтованого типу, який визначає релевантність кожного користувача щодо обраного завдання.
2. Розробити математичну модель системи, що дозволить обчислювати показник релевантності виконавців на основі показників: досвіду, універсальності, середнього часу виконання аналогічних завдань, поточної завантаженості.
3. Розробити програмне забезпечення, що включає: серверну частину, а також веб-інтерфейс для взаємодії користувача з системою.

Результати дослідження. Було розроблено та обґрунтовано алгоритмічну й математичну основу функціонування інтелектуальної рекомендаційної системи для призначення завдань членам команди. Визначено ключові параметри, що впливають на визначення релевантності виконавця, а саме показників досвіду, універсальності, середнього часу виконання завдання, а також завантаженості виконавця. Це створило цілісну методологічну основу для подальшої програмної реалізації системи.

Загалом робота системи поділяється на два основні етапи: етап побудови профілів виконавців (етап I) та етап рекомендації (етап II).

На етапі I виконується обробка даних та обчислення всіх необхідних показників. Оскільки інформація про нове завдання може бути неструктурованою або містити нерелевантні дані, вона спершу проходить попередню обробку та маркування ознак, щоб перетворити інформацію про завдання на змістовне представлення. Модель виконує вимірювання подібності між новим завданням та наявними історичними даними про завдання з метою надання списку здібних розробників. У даній системі було застосовано порівняння за косинусоїдною подібністю. Після цього генерується оцінка релевантності (OP) для кожного розробника, який належить до відповідного набору даних. Ця оцінка розробника генерується на основі чотирьох критеріїв: досвід, універсальність, середній час виконання та завантаженість працівника.

На етапі II ранжуються всі здібні розробники, що є необхідним для найкращої точності вибору кандидата та зменшення часу на виконання. Список ранжування розробників надає відповідних розробників, які мають можливості та досвід для розв'язування нової задачі. Цей процес застосовується лише до тих розробників (здатних розробників), чиї імена пройшли етап I. Після того, як була застосована стратегія пошуку вимірювання подібності до здібних працівників, система отримує оцінку релевантності кандидатів за вищезазначеними критеріями. Останній етап складається з ранжування кандидатів у порядку спадання на основі їх відповідної оцінки релевантності та генерування списку потенційних кандидатів, з якого найперший буде рекомендований на виконання, а також пояснення, чому найкраще підходить саме він (рисунок 1).

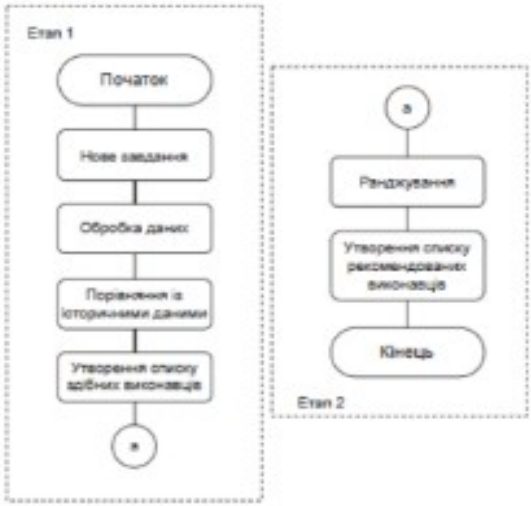


Рис. 1. Алгоритм роботи системи

Програмне забезпечення включало серверну частину на основі Python та бібліотек Pandas, NumPy, Scikit-learn, а також веб-інтерфейс для взаємодії користувача з системою, реалізований за допомогою Flask. Згадані інструменти забезпечили стабільність розрахунків, а також створили основу для подальшого розширення системи алгоритмами машинного навчання або гібридними моделями рекомендацій. Створена система сприяє оптимізації розподілу ресурсів, підвищенню прозорості та обґрунтованості управлінських рішень.

Висновки. Проведене дослідження підтвердило ефективність застосування рекомендаційних систем у процесі призначення завдань членам проєктних команд. Розроблена алгоритмічна, математична та програмна складові забезпечують можливість об'єктивного визначення релевантності виконавців на основі комплексного набору показників, що включають досвід, універсальність, середній час виконання аналогічних завдань та поточне завантаження. Отримані результати формують основу для подальшого розвитку системи, зокрема впровадження гібридних рекомендаційних моделей, що дозволить підвищити точність і адаптивність прийняття рішень у багатопроєктному середовищі.

Список літератури

1. The standard for project management and a guide to the project management body of knowledge (PMBOK guide). (2021). Project Management Institute, Inc.

2. P. Oliveira, R. M. C. Andrade, I. Barreto, T. P. Nogueira and L. Morais Bueno, "Issue Auto-Assignment in Software Projects with Machine Learning Techniques," 2021 IEEE/ACM 8th International Workshop on Software Engineering Research and Industrial Practice (SER&IP), Madrid, Spain, 2021, pp. 65-72, doi: 10.1109/SER-IP52554.2021.00018.