

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА**

**Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»**

Кафедра інформаційних систем в економіці

**ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА
«Системи штучного інтелекту»**

галузь знань	12 Інформаційні технології
спеціальність	122 Комп'ютерні науки

Форма навчання: очна (денна)

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему: «Створення платформи для планування та досягнення цілей з використанням штучного інтелекту та персоналізованих рекомендацій»

здобувача Донєва Данила Романовича
(ПІБ, підпис)

Науковий керівник: к.т.н., доцент Васильєва Л. В.
(науковий ступінь, учене звання, ПІБ)

(підпис)

**Робота допущена до захисту перед екзаменаційною
комісією з атестації здобувачів вищої освіти (ЕК)**

Завідувач кафедри: к.е.н., доцент Тішков Б.О.

(підпис)

Київ 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА

«Системи штучного інтелекту»

галузь знань

12 Інформаційні технології

спеціальність

122 Комп'ютерні науки

ПОГОДЖЕНО:

Керівник проєктної групи (гарант)
освітньо-професійної програми

ЗАТВЕРДЖУЮ:

Завідувач кафедри інформаційних
систем в економіці

_____ Васильєва Л.В.
« ____ » _____ 2025 р.

_____ Тішков Б.О.
« ____ » _____ 2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти _____ Донєва Данила Романовича
(прізвище, ім'я, по батькові)
_____ очної (денної) форми навчання
_____ очної (денної), заочної, дистанційної

на підготовку кваліфікаційної магістерської роботи
на тему: «Створення платформи для планування та досягнення цілей з
використанням штучного інтелекту та персоналізованих рекомендацій»

Тему затверджено наказом ректора Університету від «07» 03 2025 р. № 464-ст _____

Кваліфікаційна магістерська робота виконується на матеріалах

Наукових публікацій та публікацій з мережі інтернет

План кваліфікаційної магістерської роботи

Розділ I Дослідження підходів до створення системи штучного інтелекту предметної
області

(назва розділу)

Розділ II _____ Характеристика системи штучного інтелекту та постановка задачі
(назва розділу)

Розділ III _____ Розроблення проєктних рішень
(назва розділу)

Об'єкт дослідження: _____ Процес планування та досягнення особистих і професійних
цілей користувача за допомогою цифрових інструментів

Предмет дослідження: _____ Методи застосування штучного інтелекту для побудови
персоналізованих рекомендацій у контексті досягнення цілей.

Мета кваліфікаційної магістерської роботи: Розробити архітектуру, методологію та MVP-реалізацію системи інтелектуального планування, яка поєднує великі мовні моделі, класифікаційні та рекомендаційні алгоритми для підтримки користувача у процесі досягнення особистих цілей.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I здобувач має здійснити комплексний огляд сучасних наукових підходів до формулювання цілей, методів особистого та організаційного планування, а також розглянути психологічні, когнітивні та алгоритмічні основи цілепокладання. Особливу увагу слід приділити аналізу існуючих інтелектуальних систем, які реалізують підтримку користувача у процесі досягнення цілей, із виявленням їх сильних і слабких сторін. У межах теоретичного обґрунтування також необхідно розкрити актуальність проблеми інтеграції персоналізованих рекомендацій у системи штучного інтелекту, обґрунтувати вибір інструментів, моделей і технологій, які будуть використані в подальших етапах дослідження та реалізації.

У розділі II здобувач має сформулювати архітектурну модель системи штучного інтелекту, що забезпечує підтримку особистого планування. Здобувач має описати структуру системи, її ключові логічні компоненти та функціональні блоки, зокрема інформаційну модель користувача, поведінковий профіль та механізми адаптації. Паралельно необхідно обрати релевантні математичні моделі - класифікаційні, кластеризаційні, моделі з навчанням з підкріпленням - та обґрунтувати їх доцільність у контексті поставленої задачі. Передбачається також опис архітектури даних, яка базується на мікросервісному підході з чітким розділенням логіки обробки, зберігання та рекомендацій. У межах цього ж розділу доцільно побудувати відповідні діаграми UML для ілюстрації взаємодії компонентів, а також формалізувати алгоритми оптимізації задач планування.

У розділі III здобувач повинен реалізувати мінімально життєздатний прототип (MVP) системи, що охоплює фронтенд-інтерфейс, серверну логіку, API-шар, генерацію рекомендацій та базу знань. Одним із завдань є реалізація трекінгу взаємодій користувача з платформою, включаючи події генерації, збереження, редагування цілей і реакції на рекомендації. Окрему увагу слід приділити побудові та автоматизації пайплайнів агрегації поведінкових даних, які оновлюють сегментацію користувачів на основі накопичених подій. У межах аналітичної частини необхідно здійснити оцінку якості рекомендаційної системи та логіки адаптації, з урахуванням зворотного зв'язку користувачів. Фінальним кроком є побудова економічної моделі функціонування платформи з розрахунком критичних продуктово-фінансових показників: вартості підписки, точки беззбитковості, прогнозованого LTV та стратегії масштабування.

**Завдання підготував
науковий керівник**

(підпис)

Л. В. Васильсва
(ініціали, прізвище)

«07» _____ 05 _____ 2025 р.

**Завдання одержав
здобувач**

(підпис)

Д. Р. Донєв
(ініціали, прізвище)

«07» _____ 05 _____ 2025 р.

Реферат

Кваліфікаційна магістерська робота містить 68 сторінок, 6 таблиць, 23 рисунків, список використаних джерел з 27 найменувань, 3 додатки.

«Створення платформи для планування та досягнення цілей з використанням штучного інтелекту та персоналізованих рекомендацій»

Об'єктом дослідження кваліфікаційної магістерської роботи є процеси підтримки користувача у формуванні, плануванні та досягненні особистих цілей за допомогою інтелектуальних інформаційних систем.

Предметом дослідження є Методи застосування штучного інтелекту для побудови персоналізованих рекомендацій у контексті досягнення цілей.

Метою роботи є розробка архітектури, методології та реалізація MVP прототипу платформи, яка використовує алгоритми штучного інтелекту, великі мовні моделі (LLM) і поведінкову аналітику для підвищення ефективності досягнення користувачем власних цілей. У роботі розв'язано низку завдань: здійснено огляд предметної області, аналіз наявних платформ, обґрунтовано вибір технологій і підходів, описано структуру системи, реалізовано модель користувача та рекомендаційні алгоритми, зокрема класифікацію, кластеризацію та RL-агента для адаптації. Сформовано базу знань і даних, реалізовано трекінг взаємодій користувача, збереження історії та динамічну адаптацію системи до фідбеку. Особливу увагу приділено технічній реалізації API, інтерфейсу на Streamlit, обробці запитів до LLM (GPT-4o) та економічній доцільності проєкту.

Теоретична та прикладна значущість роботи полягає у синтезі сучасних підходів до автоматизованої підтримки цілепокладання та персонального коучингу. Розроблене рішення може знайти застосування в особистісному, освітньому чи корпоративному контексті, демонструє високий рівень адаптивності та здатність до самонавчання на основі зворотного зв'язку, що дозволяє покращити якість підтримки користувача та підвищити ймовірність досягнення ним поставлених цілей.

Рік виконання кваліфікаційної магістерської роботи – 2025.

Рік захисту роботи – 2025.

Ключові слова: штучний інтелект, великі мовні моделі, SMART, OKR, кластеризація, планування, рекомендовані системи, GPT-4o, Streamlit, Prefect.

Summary

The master's qualification thesis contains 68 pages, 6 tables, 23 figures, a list of 27 references, and 3 appendices.

“Development of a Platform for Goal Planning and Achievement Using Artificial Intelligence and Personalized Recommendations”

The *object of the study* is the processes of user support in formulating, planning, and achieving personal goals with the help of intelligent information systems.

The *subject of the study* encompasses a set of theoretical, methodological, and practical approaches to the design of personalized artificial intelligence systems aimed at supporting goal-setting processes, adaptive planning, and motivational guidance.

The *purpose of the study* is to develop the architecture, methodology, and MVP implementation of a platform that leverages artificial intelligence algorithms, large language models (LLMs), and behavioral analytics to enhance the effectiveness of personal goal achievement. The research solves a series of tasks: it provides an overview of the problem domain and current platforms, justifies the choice of technologies and approaches, describes the structure of the system, and implements user modeling and recommendation algorithms, including classification, clustering, and reinforcement learning agents for personalization. A knowledge base and event tracking system are developed, enabling historical data storage and adaptive behavior based on user feedback. Particular attention is paid to the implementation of APIs, a Streamlit-based user interface, LLM integration (via GPT-4o), and an economic viability assessment of the system.

The *theoretical and practical significance* of the work lies in synthesizing contemporary approaches to automated goal-setting assistance and personalized coaching. The proposed solution can be applied in personal, educational, or corporate contexts and demonstrates a high level of adaptability and self-learning capacity based on user feedback, which increases the quality of support and the probability of successful goal achievement.

The year of completion of the master's thesis: 2025

The year of defense: 2025

Keywords: artificial intelligence, large language models, SMART, OKR, clustering, planning, recommender systems, GPT-4o, Streamlit, Prefect

Рецензія
на кваліфікаційну магістерську роботу
здобувача вищої освіти

Донєва Данила Романовича

Тема: Створення платформи для планування та досягнення цілей з використанням штучного інтелекту та персоналізованих рекомендацій

Актуальність теми кваліфікаційної магістерської роботи і доцільність її розроблення. Обрана тема є актуальною з огляду на стрімкий розвиток систем штучного інтелекту в прикладних галузях, зокрема у сфері персонального планування, трекінгу звичок та цифрової підтримки самоменеджменту. Робота поєднує сучасні досягнення інтелектуальних технологій із потребами кінцевих користувачів.

Якість проведеного дослідження. Робота демонструє високий рівень аналітичного мислення та системного підходу до вирішення складної задачі. Дослідження виконане ґрунтовно: автором проаналізовано велику кількість джерел, сформульовано наукову проблему, реалізовано повноцінний MVP-прототип. Структура роботи логічна, текст написаний академічно, з достатнім рівнем глибини.

Позитивні риси кваліфікаційної магістерської роботи. До основних переваг роботи слід віднести комплексність вирішення поставленого завдання, практичну реалізацію системи з використанням сучасного стеку технологій. Автор демонструє не лише володіння інструментами, але й розуміння принципів моделювання поведінки користувача, адаптації системи на основі зворотного зв'язку, побудови сегментів.

Зауваження. Немає.

Практична значимість висновків і рекомендацій. Робота має високу прикладну цінність. Запропоновані методи та реалізовані компоненти можуть бути використані як основа для створення персональних помічників у сфері продуктивності, коучингу, освітнього супроводу або цифрової терапії.

Місце роботи та посада рецензента:

Рецензент:

Андрій ЛУЗАН

Розробник програмного
забезпечення ТОВ «Айті Ленд»

Директор ТОВ «Айті Ленд»
Дрок Аліна



В і д г у к

про кваліфікаційну магістерську роботу
здобувача навчально-наукового інституту «Інститут інформаційних технологій в економіці»
освітньо-професійної програми «Системи штучного інтелекту»

Донєва Данила Романовича

(прізвище, ім'я, по батькові)

Тема: «Створення платформи для планування та досягнення цілей з використанням штучного інтелекту та персоналізованих рекомендацій»

1. Актуальність теми: Тематика роботи є актуальною в умовах зростання популярності цифрових сервісів саморозвитку та потреби користувачів у персоналізованій підтримці. Застосування штучного інтелекту для допомоги у формуванні й досягненні цілей відкриває нові можливості в напрямках освітніх, коучингових і бізнес-рішень.

2. Позитивні риси кваліфікаційної магістерської роботи: Робота має чітку логічну структуру, ґрунтовну теоретичну базу та добре аргументовану мотивацію розробки. Автор продемонстрував високий рівень аналітичного мислення, здатність до систематизації джерел, а також вміння реалізовувати складні технічні рішення. Особливо варто відзначити якість та повноту представлення практичної частини.

3. Наявність самостійних розробок автора Донєв Д.Р. самостійно реалізував прототип платформи, включаючи інтерфейс користувача, модуль рекомендацій, алгоритми класифікації користувачів та систему візуалізації цілей. Було здійснено генерацію і обробку синтетичних даних, побудову кластерної моделі та інтеграцію результатів у функціонал продукту.

4. Цінність теоретичних висновків та практичних рекомендацій: Робота містить вагомий вклад як у прикладному, так і в науковому аспекті. Теоретичні положення можуть бути використані для подальших досліджень у сфері персоналізованих ІТ-систем, а практичні розробки – для створення реальних продуктів. Запропонована архітектура платформи є універсальною і може бути адаптована під різні категорії користувачів.

5. Наявність недоліків: До зауважень можна віднести обмежений аналіз альтернативних підходів при виборі моделей машинного навчання, а також поверхове висвітлення процесів валідації рекомендацій. Проте ці недоліки не впливають суттєво на загальну якість дослідження.

6. Загальна оцінка кваліфікаційної магістерської роботи та її допущення до захисту перед ЕК: Кваліфікаційна робота Донєва Данила Романовича відповідає вимогам до магістерських досліджень, демонструє високий рівень самостійної науково-практичної підготовки автора та заслуговує на оцінку «відмінно». Рекомендую роботу до захисту перед екзаменаційною комісією.

Науковий керівник доцент кафедри інформаційних систем в економіці, доцент,
К.Т.Н.

(посада, учене звання, науковий ступінь)

(підпис)

Васильєва Л.В.

(прізвище, ініціали)

«12» 05 2025 р.

ЗМІСТ

ВСТУП.....	1
РОЗДІЛ 1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ ПРЕДМЕТНОЇ ОБЛАСТІ.....	4
1.1 Дослідження предметної області: Системи штучного інтелекту для планування та досягнення цілей	4
1.2 Аналіз існуючих систем штучного інтелекту та інтелектуальних систем предметної області	5
1.3 Постановка проблеми та формування задач	7
1.4 Обґрунтування вибору підходів і технологій для створення системи штучного інтелекту	10
Висновки до Розділу 1	12
РОЗДІЛ 2 ХАРАКТЕРИСТИКА СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ ТА ПОСТАНОВКА ЗАДАЧІ	14
2.1 Характеристика об'єкта дослідження.....	14
2.2 Структура і характеристика системи штучного інтелекту та її компонентів	16
2.2.1 Загальна архітектура системи	17
2.2.2 Типовий сценарій використання системи.....	19
2.3 Методи, моделі і моделювання процесів і елементів складних систем.....	21
2.3.1 Методи дослідження й синтезу компонент систем штучного інтелекту	21
2.3.2 Моделі та методи оптимізації в системі штучного інтелекту.....	24
2.3.3 Методи та моделі управління в системі штучного інтелекту	27
Висновки до Розділу 2	30
РОЗДІЛ 3 РОЗРОБЛЕННЯ ПРОЕКТНИХ РІШЕНЬ.....	31
3.1 Моделювання та проектування бази знань прийняття інтелектуальних рішень	31
3.1.1 Структура бази даних як представлення знань	32
3.1.2 Агрегація та підготовка даних до кластеризації	35
3.1.3 Динамічне оновлення та збагачення бази знань	36
3.2 Розроблення користувацького інтерфейсу. Елементи та структура.	37
3.2.1 Принципи побудови інтерфейсу.....	42
3.3 Проектування забезпечувальних підсистем системи штучного інтелекту. Реалізація системи.....	44
3.3.1 Інформаційне забезпечення.....	44
3.3.2 Програмне забезпечення.....	49

3.3.3 Технічне забезпечення	54
3.3.4 Організаційно-економічне забезпечення	58
Висновки до Розділу 3	60
ВИСНОВКИ	61
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	63
ДОДАТКИ	66

ВСТУП

У сучасному світі стрімкого технологічного розвитку та все швидшого зростання кількості інформаційних контекстів, які оточують людину, дедалі актуальнішою стає питання ефективнішого управління особистим та професійним розвитком. З одного боку, люди мають все більше і більше можливостей для отримання нових навичок, з іншого боку, це створює постійний тягар у вигляді незроблених завдань, невивченої інформації, неопрацьованих знань - що викликає апатію, занепад духу і не дозволяє рухатись далі. Люди стикаються з труднощами у формулюванні, плануванні та досягненні особистих цілей. Брак структурованого підходу до планування часто призводить до втрати мотивації, фокусу, незавершених завдань і, як наслідок, до зниження якості життя. Розвиток штучного інтелекту (ШІ) відкриває нові можливості для створення платформ, здатних адаптивно навчатися на основі поведінки користувача, формувати персоналізовані рекомендації та підтримувати процес досягнення цілей. Саме на стику персонального планування, рекомендованих систем і машинного навчання формується новий підхід до самоорганізації, що має не лише прикладне, а й наукове значення.

Питання ефективної постановки цілей є міждисциплінарним і активно досліджується в психології (напр. в теорії цілей і мотивації [1] та [2]), менеджменті (методології OKR, GTD, SMART) та сфері цифрових рішень - від мобільних застосунків до інтелектуальних агентів. У контексті штучного інтелекту увагу приділено питанням персоналізованого планування, рекомендацій та поведінкової адаптації. У роботі [3] описано сучасні підходи до рекомендаційних систем, що лежать в основі багатьох AI-продуктів. У сфері агентів штучного інтелекту перспективними є роботи [4] і [5], де агенти моделюють звички користувача та адаптують свої дії до патернів поведінки. Подібні ідеї реалізовані в системі Activity Coach [24], яка ставить цілі фізичної активності користувача на основі його реальної історії рухової поведінки та

адаптує їх складність залежно від прогресу. Когнітивний коучинг, описаний у працях Baumeister [25] та Deci & Ryan [26], формує теоретичну базу для систем, які враховують внутрішню мотивацію, силу наміру, когнітивне навантаження та схильність до прокрастинації. Окрема лінія досліджень стосується використання reinforcement learning (RL) для моделювання процесу досягнення цілей. Зокрема, у роботі PlanGlow [27] було запропоновано багаторівневу модель планування навчальних цілей за допомогою LLM з можливістю генерації, критики та вдосконалення планів. Система дозволяє користувачеві взаємодіяти з планом не лише як з набором задач, а як із пояснюваним динамічним об'єктом, у якому можна вносити корективи на всіх рівнях. Попри значний прогрес, більшість існуючих цифрових рішень працюють за реактивною логікою - користувач сам ініціює постановку цілей, вибір задач і планування. Проте в сучасних умовах цифрового перевантаження зростає потреба у партнерських системах, що діють як когнітивні супутники: не лише реагують на дії користувача, а прогнозують його потреби, формують релевантні пропозиції, допомагають структурувати цілі та адаптують план дій залежно від змін мотивації, настрою й поведінки. А також допомагають притримуватися плану та не вносити хаотичні зміни в нього.

Наявна наукова проблема полягає у відсутності цілісної програмної платформи, яка б поєднувала сучасні методи ІІІ з когнітивними підходами до постановки цілей. Наукове завдання полягає у створенні архітектури такої платформи, побудові персоналізованої моделі користувача та розробці алгоритмів рекомендацій, які адаптуються до змін у поведінці користувача й допомагають підвищити ймовірність досягнення цілей.

Сформована мета дослідження - розробити концепцію, архітектуру та прототип інтелектуальної платформи для планування й досягнення цілей, що поєднує механізми штучного інтелекту з персоналізованими рекомендаціями та поведінковим аналізом.

Завдання дослідження сформовані наступним чином:

1. Провести аналіз існуючих систем менеджменту своїх цілей та персонального планування;

2. Визначити ключові обмеження та можливості застосування ШІ для постановки цілей;
3. Сформулювати вимоги до інформаційної структури, бази знань і користувацької моделі;
4. Реалізувати MVP-прототип системи з базовими функціями: генерація цілей, планування задач, рекомендації;
5. Провести тестування платформи на фокус-групі та проаналізувати ефективність запропонованих рішень.

Об'єкт дослідження: Процес планування та досягнення особистих і професійних цілей користувача за допомогою цифрових інструментів.

Предмет дослідження: Методи застосування штучного інтелекту для побудови персоналізованих рекомендацій у контексті досягнення цілей.

У ході роботи буде використано:

- методи машинного навчання;
- методи моделювання поведінки користувача;
- експертні інтерв'ю та опитування користувачів;
- програмна реалізація з використанням Python та MySQL.

Теоретичне значення роботи полягає в обґрунтуванні застосування інтелектуальних технологій для покращення процесу досягнення цілей. Методична значущість – у побудові підходу до персоналізованого планування на основі даних про поведінку користувача. Практична значущість – у створенні MVP, що може бути використаний для особистого планування, коучингу, HR-аналітики, освітніх платформ.

До інформаційної бази належать наукові публікації в галузях штучного інтелекту, когнітивної науки, поведінкової економіки, а також аналітика використання цифрових трекерів (Google Fit, Notion, Todoist), звіти компаній Gartner, McKinsey, дослідження IEEE, а також дані користувацьких інтерфейсів, що зібрані в процесі тестування прототипу.

Кваліфікаційна магістерська робота містить 68 сторінок, 6 таблиць, 23 рисунків, список використаних джерел з 27 найменувань, 3 додатки.

РОЗДІЛ 1

ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Дослідження предметної області: Системи штучного інтелекту для планування та досягнення цілей

Сучасні системи штучного інтелекту (СШІ), орієнтовані на планування та досягнення особистих і професійних цілей, формують окремий міждисциплінарний пласт досліджень. Вони об'єднують знання з прикладної психології, інформатики, когнітивних наук, поведінкової економіки та математичного моделювання. Особливу увагу приділено таким компонентам, як системи підтримки прийняття рішень, персоналізовані цифрові асистенти, автономні агенти та моделі, що здатні адаптуватися до змін у поведінці користувача. Центральною ідеєю таких систем є побудова персоналізованої моделі користувача, яка дозволяє ідентифікувати індивідуальні когнітивні особливості, шаблони поведінки й особисті переваги. В результаті це створює можливість для гнучкого адаптивного супроводу процесу формулювання та досягнення цілей, що знижує ймовірність виснаження та відмови від запланованого. Актуальність теми зумовлена, зокрема, високим рівнем когнітивного навантаження, яке виникає у користувачів під час взаємодії з традиційними інструментами цифрового планування. Хоча на ринку присутні потужні рішення, такі як Notion, Todoist, Google Calendar, їх використання часто обмежується фрагментарною фіксацією задач. Систематична робота з ними потребує від користувача високої дисципліни та самоконтролю, чого, як свідчать емпіричні дослідження, бракує більшості осіб без зовнішньої підтримки. Відсутність контекстної адаптації, інтелектуального нагадування і стратегічного супроводу зменшує ефективність таких рішень у довготривалому використанні.

Таким чином, ключовою мотивацією є створення платформи, яка забезпечує не просто ведення списків завдань, а функціонує як інтелектуальний партнер, що пропонує релевантні стратегії досягнення цілей, адаптується до змін у зовнішньому середовищі та враховує внутрішній стан користувача. В цьому контексті дослідження буде сфокусовано на аналізі сучасних підходів до прийняття рішень у динамічних умовах, вивченні механізмів формування звичок, проектуванні алгоритмів генерації персоналізованих рекомендацій, а також створенні моделей адаптивного планування, що функціонують у реальному часі. Окрема увага буде приділена бар'єрам ефективного цілепокладання (прокрастинація, надмірна деталізація, параліч вибору тощо) та способам їх компенсації через технології штучного інтелекту.

1.2 Аналіз існуючих систем штучного інтелекту та інтелектуальних систем предметної області

Останні роки характеризуються стрімким зростанням кількості інформаційних систем, що реалізують функції штучного інтелекту (ШІ) для підтримки користувачів у різних аспектах повсякденного життя. Зокрема, спостерігається зростання популярності цифрових асистентів, мобільних додатків для підтримки ментального здоров'я, когнітивно-поведінкових чатботів, а також рекомендаційних систем, що адаптуються до індивідуальних характеристик користувача. Однак, незважаючи на значний прогрес у цій сфері, більшість з існуючих рішень не забезпечують повноцінної підтримки процесу постановки, адаптації та досягнення персональних цілей.

До найпоширеніших категорій сучасних систем, що використовують методи ШІ, належать:

- Голосові цифрові помічники загального призначення (Google Assistant, Apple Siri, Amazon Alexa) - платформи, які здатні розпізнавати

природну мову, здійснювати базову підтримку навігації, пошуку, керування пристроями та нагадувань. Проте їхня функціональність обмежується реактивною взаємодією та не передбачає проактивного супроводу особистих цілей користувача.

- Емоційно-орієнтовані AI-компаньйони (наприклад, Replika.ai), які реалізують функції соціальної підтримки, заснованої на обробці емоційного стану та історії взаємодії з користувачем. Водночас такі системи не мають механізмів формального планування чи відстеження прогресу в досягненні конкретних цілей.

- Мобільні застосунки для підтримки добробуту та формування звичок (Fabulous, Finch, Calm, Headspace), що базуються на принципах когнітивної психології та гейміфікації. Вони пропонують структурування щоденних рутин та зменшення стресу, проте не інтегрують комплексну модель цілепокладання з адаптивною підтримкою.

- Когнітивно-поведінкові чат-боти (Wysa, Woebot) демонструють можливості базової емоційної самодопомоги, ґрунтуючись на принципах КПТ (когнітивно-поведінкової терапії), однак їх функціонал не охоплює довгострокове супроводження користувача в межах особистісного зростання.

Порівняння описаних продуктів наведено в таблиці 1.1.

Таблиця 1.1 – Порівняння рішень на ринку

Назва системи / застосунку	Постановка цілей	Адаптивна підтримка	Відстеження прогресу	Поведінковий аналіз	Емоційна підтримка	Прозорість рекомендацій
Google Assistant / Siri	—	—	—	—	—	—
Replika.ai	—	частково	—	—	+	—
Fabulous / Finch	частково	частково	частково	—	+	—
Headspace / Calm	—	—	—	—	+	—
Wysa / Woebot	—	частково	—	частково	+	—

Джерело: сформовано автором на основі виконаного дослідження

Критичний аналіз вищенаведених прикладів дає підстави стверджувати, що актуальні інтелектуальні системи мають такі обмеження:

1. Відсутність комплексної інтеграції між постановкою цілей, персоналізованими рекомендаціями та моніторингом прогресу;
2. Низький рівень контекстної адаптації до змін у стані користувача або зовнішніх умов;
3. Обмежена прозорість логіки прийняття рішень системою, що ускладнює навчання користувача;
4. Відсутність довготривалого супроводу, здатного реагувати на поведінкові шаблони та когнітивні бар'єри;
5. Використання статичних моделей, які не враховують міждисциплінарну взаємодію (наприклад, вплив ментального стану на професійну ефективність).

Таким чином, наразі залишається відкритою потреба у створенні системи, що виконує функції не лише планувальника, але й персонального інтелектуального супровідника, орієнтованого на довгострокову зміну поведінки, підтримку адаптивного цілепокладання та підвищення ефективності менеджменту цілей. Підхід до цілепокладання спирається на класичну теорію мотивації та цілей, викладену в [1].

1.3 Постановка проблеми та формування задач

Основна практична проблема полягає у відсутності цілісної, гнучкої та адаптивної інтелектуальної платформи, яка б органічно поєднувала потужні засоби інтелектуального аналізу даних, механізми персоналізованих рекомендацій, інтуїтивно зрозумілий та зручний інтерфейс для щоденної взаємодії та глибоке розуміння психологічних аспектів саморегуляції. Когнітивні бар'єри планування детально досліджені в роботі [2].

Розроблена система повинна не лише пасивно реагувати на безпосередні дії користувача, фіксуючи його прогрес, але й проявляти проактивність, своєчасно пропонуючи обґрунтовані зміни у планах, гнучко коригувати цілі у відповідності до мінливого зовнішнього та внутрішнього контексту, виявляти потенційні бар'єри на шляху до успіху та надавати психологічно релевантну, емпатичну підтримку, що сприяє підвищенню мотивації та стійкості.

Особливої уваги потребує врахування етичних та принципів прозорості у процесі розробки та функціонування алгоритмів, особливо у контексті обробки та аналізу конфіденційних даних, пов'язаних з ментальним здоров'ям, емоційним благополуччям та глибоко особистими життєвими цілями користувача. Забезпечення розуміння логіки прийняття рішень системою є критично важливим для формування довіри та ефективної співпраці між людиною та ШІ.

Основні задачі дослідження:

Для досягнення поставленої мети необхідно вирішити наступні взаємопов'язані задачі, які умовно поділяються на декілька етапів:

1. Здійснити всебічний огляд сучасних наукових підходів до моделювання складного процесу цілепокладання в психології, менеджменті та інформаційних технологіях.

2. Провести критичний аналіз існуючих цифрових рішень (мобільних застосунків, веб-платформ, інтелектуальних помічників) для персонального планування та самовдосконалення, виявити їхні ключові обмеження, сильні сторони та потенційні напрямки для вдосконалення.

3. Визначити та обґрунтувати оптимальну структуру даних, необхідну для ефективного формування персоналізованих рекомендацій, прогнозування поведінки та відстеження прогресу користувача у досягненні цілей.

4. Сформулювати детальну модель цілей, що включатиме ієрархічну структуру (від стратегічних до тактичних), атрибути цілей (важливість, терміновість, складність), взаємозв'язки між різними сферами життя користувача (здоров'я, кар'єра, особисті стосунки, хобі тощо) та їхній вплив одна на одну.

5. Розробити гнучку та масштабовану архітектуру інтелектуальної системи, що складатиметься з ключових функціональних модулів: модуля інтелектуального аналізу цілей, адаптивного планувальника завдань та персоналізованого рекомендаційного блоку.

6. Реалізувати прототип (MVP - Minimum Viable Product) платформи на основі обраного стеку технологій, що включає Python (з використанням фреймворків Flask або FastAPI для backend розробки), систему керування базами даних MySQL для зберігання даних, набір бібліотек машинного навчання та інструменти для розробки користувацького інтерфейсу (Streamlit або React для frontend).

7. Передбачити та реалізувати ефективні механізми збирання різноманітних поведінкових даних користувача (активність у системі, прогрес у виконанні завдань, зворотний зв'язок, емоційний стан) з подальшим їхнім інтелектуальним аналізом для покращення роботи системи.

8. Провести контрольовану апробацію розробленої системи на репрезентативній групі користувачів з метою оцінки її практичної ефективності та зручності використання.

9. Сформувати обґрунтовані критерії ефективності розробленої інтелектуальної системи, які включатимуть як кількісні (наприклад, коефіцієнт реалізації поставлених цілей, динаміка зміни регулярності виконання задач, рівень зниження прокрастинації), так і якісні показники (наприклад, суб'єктивна оцінка корисності рекомендацій, рівень задоволеності користувацьким досвідом).

10. Розробити чітку та зрозумілу методику інтерпретації результатів роботи системи для кінцевого користувача, з особливим акцентом на забезпеченні прозорості та пояснюваності дій алгоритмів, що сприятиме підвищенню довіри та розуміння принципів функціонування системи.

1.4 Обґрунтування вибору підходів і технологій для створення системи штучного інтелекту

Проектування системи штучного інтелекту для підтримки процесу планування та досягнення цілей вимагає інтеграції рішень, що відповідають низці критичних вимог: персоналізації, обробці неструктурованої інформації, адаптивності до змін у поведінці користувача, а також прозорості й пояснюваності прийнятих рішень. З урахуванням цих потреб було обрано архітектурну модель, у межах якої реалізується розділення логічних модулів за функціональним призначенням, що забезпечує технологічну гнучкість та масштабованість розробки.

Основною мовою реалізації серверної частини системи обрано Python. Це рішення зумовлено широкою підтримкою мови в галузі машинного навчання, наявністю зрілої екосистеми бібліотек для побудови аналітичних моделей, організації взаємодії з базами даних, реалізації API та створення інтерактивних інтерфейсів. У межах запропонованої архітектури Python виступає платформою для реалізації бізнес-логіки системи, обробки даних, ініціалізації моделей штучного інтелекту та комунікації з клієнтським інтерфейсом.

Ключовим технологічним компонентом є використання великих мовних моделей (LLM), зокрема архітектур GPT (через API OpenAI) та LLaMA (у локальному середовищі через Ollama або HuggingFace). Це дозволяє системі не лише обробляти природну мову, а й генерувати структуровані цілі, адаптовані до контексту користувача. Такі моделі застосовуються для трансформації вхідної інформації у форматі вільного тексту у формалізовані цілі за шаблонами SMART[21] або OKR[22], для формування персоналізованих підказок, а також для пояснення логіки запропонованих рекомендацій.

З метою підвищення адаптивності системи реалізовано сегментацію користувачів за поведінковими ознаками. Для цього побудовано кластерну модель, що враховує активність, ритм планування, схильність до завершення

завдань, а також змінні, пов'язані з мотиваційним станом. На основі сформованих сегментів система диференціює тип рекомендацій, їхній рівень деталізації та формат подачі. Додатково реалізовано модель прогнозування ймовірності досягнення цілі на основі індивідуального профілю користувача, що дозволяє формувати релевантні сценарії підтримки.

Завдання класифікації, регресії та прогнозування реалізовані із застосуванням глибоких нейронних мереж. Базові архітектури типу багатошарового перцептрона використовуються для обробки табличних поведінкових даних. Рекурентні нейронні мережі (зокрема, LSTM) дозволяють моделювати часову залежність між подіями у життєвому циклі користувача. З метою виявлення латентних характеристик особистості та поведінки застосовуються автоенкодери.

Для зберігання структурованих даних використовується реляційна система управління базами даних MySQL. Такий вибір обумовлений вимогами до консистентності, транзакційності та підтримки чітко визначеної логіки зв'язків між об'єктами - користувачами, цілями, задачами, оцінками. Схема даних реалізована з дотриманням принципів нормалізації та забезпечує розширюваність і зручність для аналітичної обробки.

На етапі прототипування інтерфейс користувача реалізовано за допомогою Streamlit - легкого фреймворку на Python, що дозволяє швидко створювати інтерактивні веб-додатки. Такий підхід дозволив реалізувати MVP із мінімальними витратами на фронтенд-розробку. У разі масштабування системи передбачено перенесення інтерфейсу на React, що дає змогу побудувати модульну, реактивну й адаптивну клієнтську частину з підтримкою мобільних пристроїв.

Висновки до Розділу 1

У першому розділі здійснено комплексне дослідження науково-методичних та технологічних засад побудови інтелектуальних систем, призначених для підтримки процесу планування та досягнення особистісних цілей. Розглянуто теоретичні основи цілепокладання, зокрема концепції SMART, OKR та когнітивні аспекти формування намірів, а також проведено критичний аналіз сучасних підходів до персоналізованого цифрового супроводу, реалізованих у формі голосових асистентів, поведінкових чат-ботів і мобільних застосунків для самоменеджменту. Результати аналізу засвідчили, що більшість існуючих рішень демонструють лише часткову реалізацію ключових функцій, як постановка цілей, їх структуризація, адаптивне планування та динамічна підтримка. Встановлено, що актуальною проблемою залишається відсутність комплексної системи, здатної виконувати функції когнітивного супровідника - тобто системи, яка не лише фіксує дії користувача, а й надає релевантні рекомендації, формує оптимальні стратегії досягнення цілей і здатна адаптуватися до змін у поведінці, мотивації та контексті. На основі опрацьованих матеріалів було сформульовано дослідницьку проблему та визначено цілі й завдання дослідження. Сформовано бачення вигляду і функцій системи штучного інтелекту, що передбачає інтеграцію великих мовних моделей для генерації та трансформації цілей, використання нейронних мереж для прогнозування результативності планів, а також побудову сегментованих сценаріїв поведінки користувача. Обґрунтовано доцільність використання мови Python як базової платформи реалізації серверної логіки, реляційної СУБД MySQL для структурованого зберігання даних, а також засобів швидкої веб-розробки Streamlit із перспективою переходу до React. Запропоновані теоретичні положення та технологічні рішення формують концептуальну й інструментальну основу для реалізації інтелектуальної системи, орієнтованої на персоналізоване планування та супровід користувача у процесі досягнення цілей. Отримані

результати створюють підґрунтя для подальшого формального опису інформаційного, математичного та програмного забезпечення системи у наступних розділах роботи.

РОЗДІЛ 2

ХАРАКТЕРИСТИКА СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ ТА ПОСТАНОВКА ЗАДАЧІ

2.1 Характеристика об'єкта дослідження

Об'єктом дослідження є розроблена в рамках навчальної практики інтелектуальна система підтримки процесу планування та досягнення цілей, що поєднує інструменти машинного навчання, великі мовні моделі та адаптивні рекомендаційні механізми. Система реалізована у вигляді веб-застосунку, створеного за допомогою Python та фреймворку Streamlit, з використанням реляційної бази даних MySQL та інтеграції з зовнішніми API великих мовних моделей (зокрема, GPT-4 через OpenAI API).

Під час практики було реалізовано повноцінний прототип, який дозволяє користувачу вводити цілі у вільній формі, трансформувати їх у структуровані SMART- або OKR-моделі за допомогою мовної моделі, формувати задачі, планувати їх виконання у часовому розрізі та відстежувати прогрес. Вся інформація зберігається у структурованій формі в базі даних, а взаємодія між компонентами реалізована через REST API.

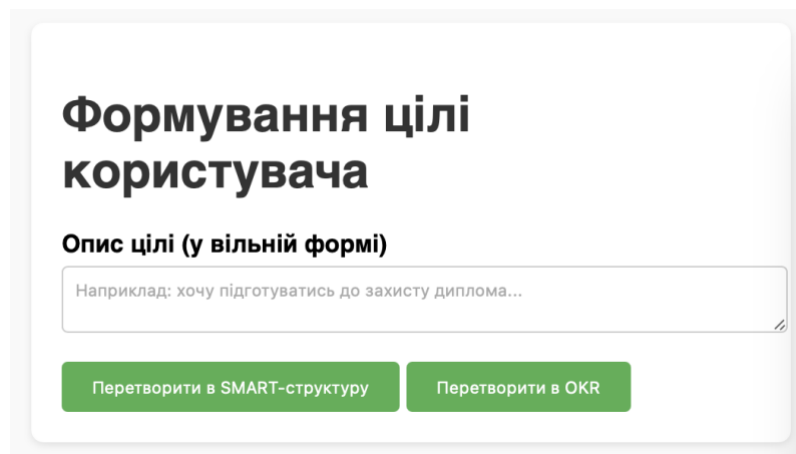
Дані, які використовувались у системі, охоплюють:

- понад 100 унікальних формулювань цілей (введених вручну або згенерованих на основі реальних шаблонів);
- понад 400 задач, прив'язаних до цілей;
- 8 типових сфер життєдіяльності (здоров'я, кар'єра, фінанси, сім'я, особистий розвиток, духовність, відпочинок, середовище);
- лог дій користувача (додавання, редагування, завершення задач);
- 60+ згенерованих рекомендацій мовною моделлю.

Типові шаблони цілей, що використовуються у системі, відповідають методологіям SMART та OKR і мають таку форму:

- SMART: "Пройти онлайн-курс з аналізу даних до 1 липня, приділяючи по 3 години на тиждень."
- OKR: "Мета: підвищити фізичну витривалість. Ключові результати: 1) Пробігти 50 км за місяць. 2) Пройти медичний огляд. 3) Тренуватись 3 рази на тиждень."

Наприклад, на рис 2.1 можна побачити форму збору цілей користувачів для дослідження.



The screenshot shows a web form titled "Формування цілі користувача" (Goal Formulation for the user). Below the title is a section labeled "Опис цілі (у вільній формі)" (Goal description (in free form)). Inside this section is a text input field with the placeholder text "Наприклад: хочу підготуватись до захисту диплома..." (For example: I want to prepare for the diploma defense...). At the bottom of the form are two green buttons: "Перетворити в SMART-структуру" (Convert to SMART structure) and "Перетворити в OKR" (Convert to OKR).

Рисунок 2.1 – Збір цілей

Джерело: сформовано автором на основі виконаного дослідження

Система пропонує розкласти ціль в структуру, що демонструється на рис. 2.2.

Рисунок 2.2 – Розкладування цілей

Джерело: сформовано автором на основі виконаного дослідження

Таким чином, система апробована в рамках практичної діяльності на основі тестових даних і реалізована як програмно завершене рішення з можливістю подальшого масштабування та використання в освітньо-мотиваційних середовищах.

2.2 Структура і характеристика системи штучного інтелекту та її компонентів

Система штучного інтелекту, що розробляється в межах даної роботи, є функціонально завершеним цифровим середовищем, яке забезпечує когнітивно-поведінковий супровід користувача у процесі постановки, планування, досягнення та перегляду особистісних цілей. Архітектурно система реалізується у вигляді модульної програмної платформи з виділеними компонентами, що взаємодіють через чітко визначені API, забезпечуючи одночасно технологічну автономність та інформаційну інтегрованість.

2.2.1 Загальна архітектура системи

Функціональна структура побудована на основі принципів розподіленої клієнт-серверної моделі із частковою мікросервісною орієнтацією. Кожен логічний блок реалізує власний набір функцій, пов'язаний із конкретною фазою життєвого циклу взаємодії користувача з платформою: від ініціації цілей до рефлексивного аналізу результатів. Передбачено підтримку асинхронної обробки подій, що дозволяє оперативно реагувати на зміну контексту, накопичення нових даних або зміну намірів користувача.

Архітектурного проектування базуємо на основах:

- модульності - забезпечуємо ізоляцію логіки та незалежну масштабованість;
- адаптивності - дає змогу реагувати на поведінкові зміни у реальному часі;
- пояснюваності - гарантує прозору аргументацію рішень для користувача, в тому числі з метою дотримання принципу explainable AI [12];
- стійкість до помилок забезпечується через логування та fallback-механізми;
- приватність реалізується через анонімізацію, шифрування, контроль доступу.

Основні компоненти системи включають:

1. Модуль аутентифікації та авторизації виконує контроль доступу до системи, керує сесіями, прив'язує взаємодію до конкретного користувача та забезпечує безпеку зберігання облікових даних через шифрування і використання токенів автентифікації.

2. Підсистема цілепокладання реалізує механізми створення, редагування та структуризації цілей. Забезпечується підтримка ієрархічного представлення цілей (головна мета → підцілі → задачі), визначення пріоритетності, часових меж та повторюваності.

3. Модель користувача - динамічно формована структура, яка відображає поведінкові патерни, історію виконання, реакції на рекомендації, а також оцінює рівень внутрішньої мотивації, стабільність дій, часові переваги та схильність до прокрастинації.

4. Рекомендаційна система працює на основі гібридних алгоритмів: класифікаторів (наприклад, XGBoost) для оцінки ймовірності досягнення мети; моделей ранжування (LightFM) для формування списку релевантних задач; а також великих мовних моделей, що генерують текстові поради, формулювання цілей і пояснення рішень.

5. Модуль обробки неструктурованих даних забезпечує трансформацію вхідного тексту (вільні описи цілей, нотатки, наміри) у формалізовані структури для подальшої обробки. Використовується зв'язка GPT-4 / LLaMA 2 через відповідні API або локальні інтерфейси.

6. Аналітичний блок і механізм прогнозування здійснюють обробку історичних даних користувача, виконують кластеризацію типів поведінки, виявляють аномалії (наприклад, раптове падіння активності) та обраховують імовірність досягнення заданої мети в обраний термін. Модуль також відповідає за автоматичне коригування очікувань системи на основі оновлених параметрів.

7. Користувацький інтерфейс реалізований на основі Streamlit для швидкого прототипування та оцінки UX-поведінки. Передбачено відображення прогресу, інтерактивне колесо життя, інтерфейс формулювання цілей та керування задачами. Компоненти взаємодіють із логікою через REST API.

8. База даних реалізована на основі MySQL і забезпечує зберігання основних сутностей: профілі користувачів, цілі, задачі, аналітичні оцінки, історія взаємодії, параметри сеансів. Структура бази відповідає вимогам нормалізації та оптимізована для аналітичного читання.

Всі залежності між підсистемами зображено на рис. 2.3

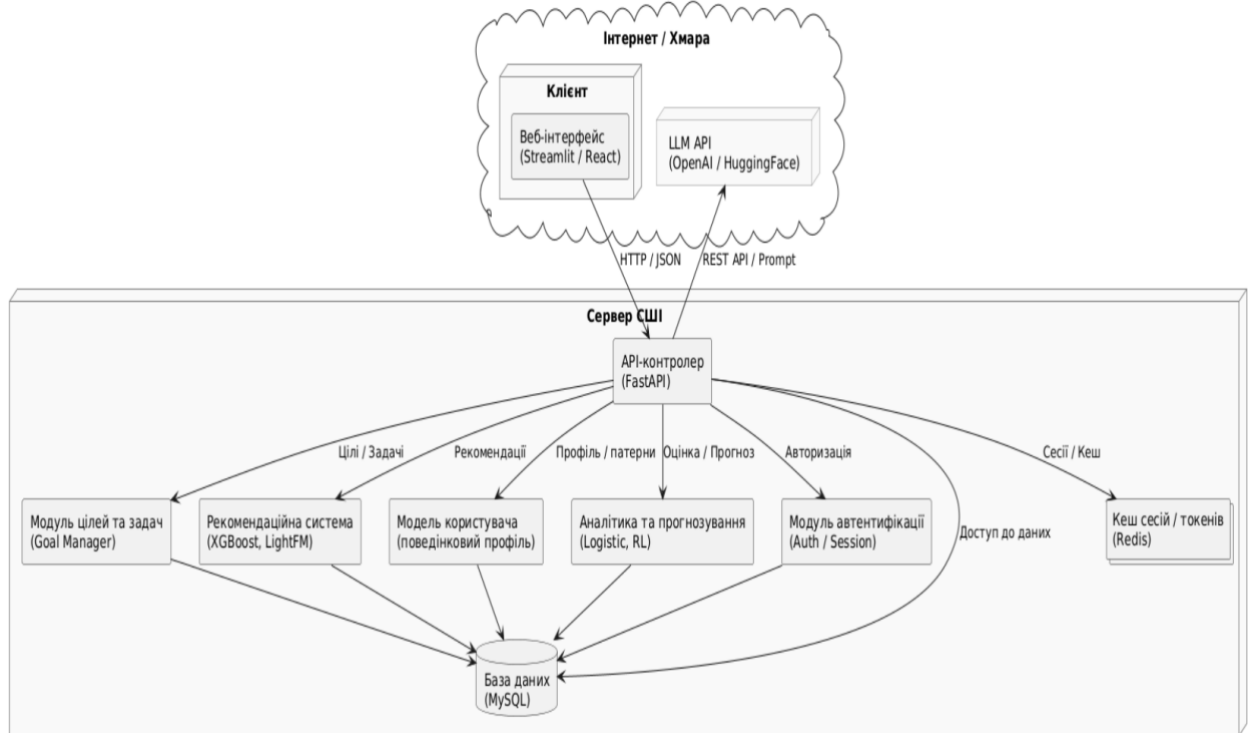


Рисунок 2.3 – Основні компоненти системи

Джерело: сформовано автором на основі виконаного дослідження

Таким чином забезпечується надійне функціонування системи та взаємозв'язок між усіма вузлами.

2.2.2 Типовий сценарій використання системи

У межах роботи системи формування та досягнення цілей ключову роль відіграє логіка взаємодії користувача з основними функціональними компонентами. Нижче наведено типовий сценарій, що демонструє послідовність дій від моменту постановки мети до адаптивного оновлення моделі користувача на основі зворотного зв'язку. Також схематично зображено на рис 2.4.

Типовий сценарій передбачає наступну послідовність:

1. Користувач формулює особисту мету у довільній формі через інтерфейс;
2. Мовна модель (LLM) трансформує введений текст у структуровану SMART- або OKR-ціль;
3. На основі поведінкового профілю та історії активності система генерує персоналізований план задач;
4. Користувач переглядає задачі, таймінг та отримує пояснення до запропонованих рішень;
5. За потреби користувач вносить зміни до задач або плану;
6. У процесі виконання задач система оновлює статуси, фіксує прогрес, реагує на відхилення та коригує наступні рекомендації;
7. Отриманий зворотний зв'язок використовується для уточнення моделі користувача та покращення подальших сценаріїв планування.

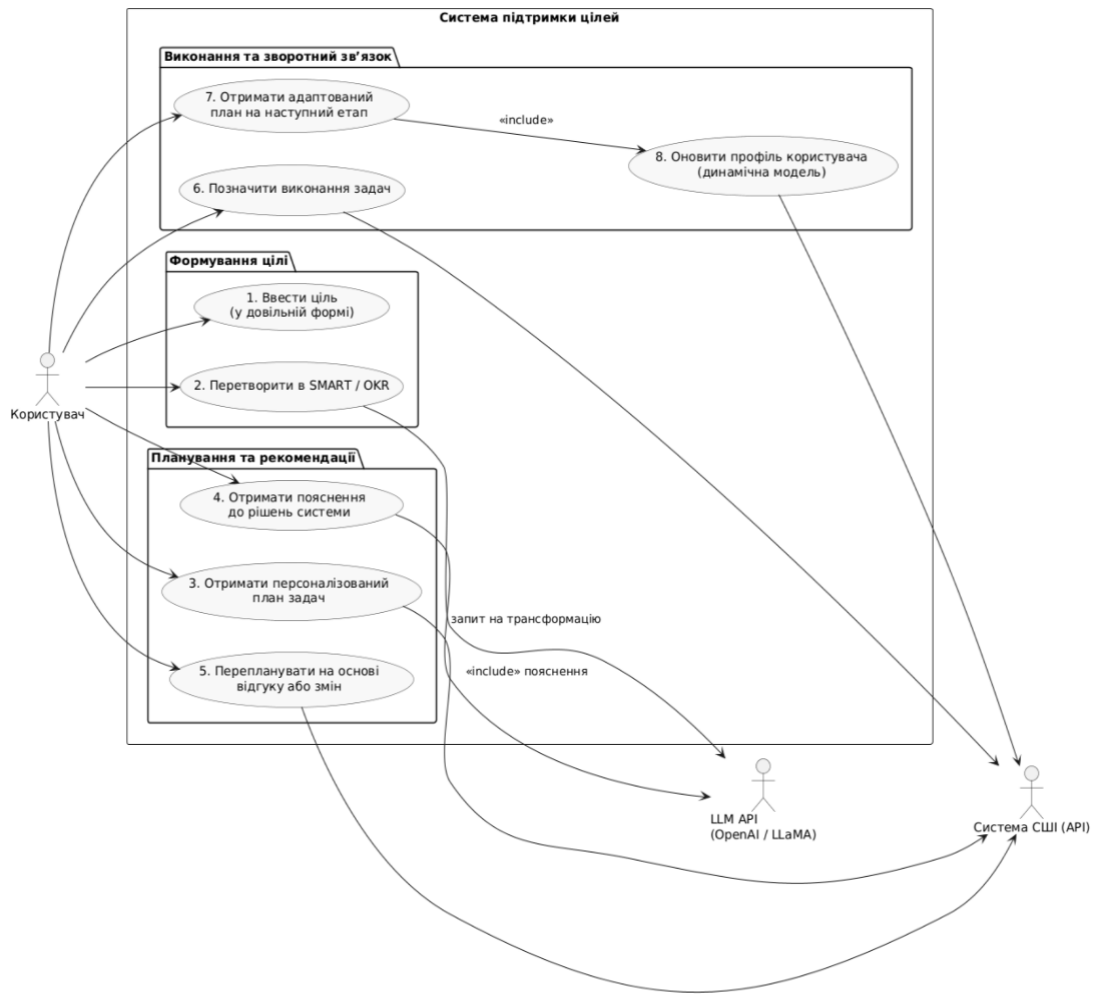


Рисунок 2.4 – Типовий сценарій взаємодії

Джерело: сформовано автором на основі виконаного дослідження

Сценарії використання можуть доповнюватися та включати додаткові кроки.

2.3 Методи, моделі і моделювання процесів і елементів складних систем

2.3.1 Методи дослідження й синтезу компонент систем штучного інтелекту

Синтез компонентів системи штучного інтелекту, орієнтованої на підтримку процесу планування та досягнення цілей, ґрунтується на поєднанні прикладних методів машинного навчання, когнітивного моделювання та

інтерактивного UX-дослідження. Завдання цього етапу - побудова компонентної моделі, що узгоджує логіку роботи рекомендаційного ядра, моделі користувача, модуля генерації тексту та оптимізаційної системи.

На етапі дослідження було сформульовано такі підзадачі:

- класифікація користувача за рівнем продуктивності, мотивації та схильності до прокрастинації;
- кластеризація поведінкових профілів для персоналізації рекомендацій;
- генерація структурованих цілей за допомогою великих мовних моделей (LLM);
- оптимізація плану задач з урахуванням дедлайнів та часових уподобань користувача;
- моделювання реакції користувача у відповідь на рекомендації.

Реалізація цих методів здійснювалася у рамках створення веб-прототипу системи, побудованого з використанням Python та фреймворку Streamlit. Для інтеграції алгоритмів машинного навчання застосовувалися бібліотеки scikit-learn, xgboost [14], LightFM, алгоритм Factorization Machines є основою LightFM [7], а також моделі трансформерів із бібліотеки Hugging Face.

Поведінкові дані для навчання та тестування формувалися на основі симульованих сесій взаємодії користувача з системою: кожна ціль включала від 3 до 8 задач із параметрами часу, дедлайнів, пріоритетів, мотиваційного рівня та відповідності до профілю користувача. Додатково використовувався зібраний вручну датасет із понад 100 прикладів текстових формулювань цілей для перевірки генерації SMART/OKR-структур. Підходи до персоналізованих рекомендацій викладені у [3]. У процесі синтезу інтелектуальних компонентів системи було застосовано низку сучасних підходів машинного навчання, що відповідають специфіці предметної області, обмеженням MVP-рівня реалізації та вимогам до персоналізації. Основною задачею було поєднання статистичних,

евристичних і нейромережових методів для побудови інтерпретованої, адаптивної системи підтримки прийняття рішень користувачем.

Для прогнозування ймовірності завершення цілей використовувались базові моделі машинного навчання з учителем, зокрема логістична регресія, Random Forest та XGBoost. Кожна з моделей була оцінена за допомогою класичних метрик якості: точність, recall, F1-score, а також площа під ROC-кривою (AUC-ROC), що дозволяє враховувати баланс між повнотою та хибно-позитивними спрацюваннями при ранжуванні задач.

Сегментація користувачів за поведінковими ознаками реалізована як задача навчання без учителя, що передбачає побудову кластерної моделі з використанням алгоритмів K-means, DBSCAN та Gaussian Mixture Models (GMM). Для визначення оптимальної кількості кластерів було використано метрики внутрішньої валідності, такі як Silhouette Score та індекс Девіса-Боулдіна (DBI), що дозволяють оцінити компактність та відокремленість кластерів. Базові моделі побудовано за допомогою бібліотеки scikit-learn [13]. Для аналізу кластерів застосовано методи, описані в [16]

У межах генеративних компонентів системи застосовано великі мовні моделі (LLM), архітектура трансформера [4] стала основою для сучасних LLM, зокрема GPT (через API OpenAI) та LLaMA (HuggingFace). Використання мовних моделей для генерації планів обґрунтовано у [24]. В якості альтернативи GPT можливо використовувати модель LLaMA [5]. Їх інтегровано для генерації структурованих SMART- та OKR-декомпозицій, трансформації вільного опису мети у формат, придатний до подальшого планування, а також для переформулювання, пояснення та адаптації повідомлень системи відповідно до стилю спілкування користувача. Нейронне колаборативне фільтрування дозволяє враховувати складні залежності між користувачами [8].

Рекомендаційна частина реалізована за гібридним принципом - об'єднано елементи контентно-орієнтованого фільтрування (на основі властивостей задач) та колаборативного фільтрування (на основі історії інших користувачів). Для цього використано модель LightFM, а також Neural Collaborative Filtering -

глибинний підхід до прогнозування релевантності завдань. Завдяки цьому вдалося побудувати персоналізований порядок задач, адаптований під профіль продуктивності.

Додатково, для реалізації поведінкової адаптації було застосовано спрощену схему навчання з підкріпленням. Логіка навчання з підкріпленням ґрунтується на класичних принципах [11]. Агент Reinforcement Learning формує політику видачі задач залежно від зворотного зв'язку користувача (оцінка рекомендацій, частота пропусків, тривалість взаємодії). Це дозволяє системі самонавчатися та уточнювати логіку подальших рішень у режимі реального часу без потреби в повному переобчисленні моделей.

2.3.2 Моделі та методи оптимізації в системі штучного інтелекту

Планування задач користувача в інтелектуальній системі досягнення цілей можна формалізувати як задачу оптимізації розподілу обмежених ресурсів у часі з урахуванням дедлайнів, залежностей між завданнями та індивідуального когнітивного профілю. Задача має схожість із класичною проблемою розкладання типу Job Shop Scheduling (JSS), проте адаптована до персонального середовища користувача та поведінкових факторів.

Нехай задано множину цілей $G = \{g_1, g_2, \dots, g_n\}$, кожна з яких складається з множини задач (операцій) $T_i = \{t_{i1}, t_{i2}, \dots, t_{imi}\}$. Кожна задача характеризується тривалістю τ_{ij} , пріоритетом u_{ij} , дедлайном d_{ij} , моментом готовності r_{ij} , а також можливою залежністю від інших задач. Нехай s_{ij} - момент початку, а $f_{ij} = s_{ij} + \tau_{ij}$ - момент завершення задачі.

Формально задача може бути зведена до мінімізації функції втрат наведено у формулі (2.1) наступного вигляду :

$$\min_{s_{ij}} \sum_{i=1}^n \sum_{j=1}^{m_i} w_{ij} (f_{ij} - d_{ij})^2 + \lambda \cdot L(s_{ij}, p_i) \quad (2.1)$$

де w_{ij} – значущість задачі,

$L(s_{ij}, p_i)$ – функція невідповідності розкладу профілю користувача

p_i – коефіцієнт ваги невідповідності.

Профіль користувача p_i містить інформацію про часові переваги, продуктивність у різні періоди доби, мотиваційні характеристики та історію взаємодії з системою. Невідповідність $L(s_{ij}, p_i)$ може бути визначена як відстань між рекомендованим і фактичним часом виконання задачі, наведено у формулі (2.2):

$$L(s_{ij}, p_i) = |s_{ij} - \mu_i|, \quad (2.2)$$

де μ_i – центр індивідуального робочого інтервалу користувача

Урахування залежностей реалізується через умову:

$$s_{ij} \geq \max(f_{kl}), \quad \text{де } t_{kl} < t_{ij}$$

Що забезпечує умову, що задача не може початись раніше завершення всіх залежних задач

Додатково враховується виключення накладень у межах одного когнітивного ресурсу:

$$[s_{ij}, f_{ij}) \cap [s_{kl}, f_{kl}) = \emptyset$$

Що забезпечує умову, що інтервали задач не повинні перетинатися, якщо вони виконуються в один часовий слот

Для розв'язання задачі можуть застосовуватись комбіновані евристики, наприклад:

- кластеризація задач за типом навантаження (глибоке / поверхнєве),
- жадібні алгоритми з корекцією,
- RL-агенти для адаптивного графіка.

З метою реалізації адаптивної поведінки доцільно формалізувати функцію винагороди RL-агента, яка враховує точність, відповідність часу профілю користувача та когнітивні витрати перемикавання, наведено у формулі (2.3):

$$R(s, a) = \alpha \cdot \text{on_time}(t_{ij}) + \beta \cdot \text{compliance}(p_i) - \gamma \cdot \text{switch_cost} \quad (2.3)$$

де α, β, γ — вагові коефіцієнти;

on_time — чи виконано задачу вчасно;

compliance — чи задача відповідає профілю p ;

switch_cost — штраф за контекстне перемикавання.

Альтернативно, для задач прийняття рішення при плануванні можна ввести функцію корисності, яку наведено у формулі (2.4):

$$U(t_{ij}) = \delta_1 \cdot u_{ij} - \delta_2 \cdot (f_{ij} - d_{ij})^2 - \delta_3 \cdot L(s_{ij}, p_i) \quad (2.4)$$

де δ_{1_3} — вагові коефіцієнти, що відображають відносну важливість

Такий підхід дозволяє врахувати trade-off між бажаністю задачі, її складністю й зручністю виконання.

Сукупне використання означених моделей дозволяє забезпечити оптимізацію розкладу з урахуванням поведінкових особливостей, що підвищує ймовірність досягнення поставлених цілей, зменшує ризик прокрастинації та сприяє зростанню залученості користувача.

2.3.3 Методи та моделі управління в системі штучного інтелекту

У межах інтелектуальної системи планування та досягнення цілей ключовими елементами, що підлягають управлінню та оцінці, є: (1) якість генерації рекомендацій великими мовними моделями (LLM) та (2) ефективність кластеризації користувачів для персоналізації логіки планування. Управління цими компонентами передбачає впровадження відповідних метрик оцінювання, побудову інтерпретованих моделей та інтеграцію зворотного зв'язку в контур навчання.

Оцінка якості генерації LLM - оскільки мовна модель формує структуровані цілі (SMART або OKR) на основі вільного тексту користувача, важливим є забезпечення її точності, релевантності та узгодженості із шаблоном. Для оцінювання якості генерованих відповідей використано комбінацію експертного оцінювання та метрик читабельності й структурованості.

Запропоновано наступні критерії:

- Структурованість (S): чи відповідає відповідь заданій схемі (SMART має всі 5 компонентів);
- Релевантність (R): чи збережено змістовну суть оригінального запиту;
- Формальність (F): чи витримано стиль, відповідний діловому або мотиваційному контексту;
- Користувацьке сприйняття (U): чи є результат зрозумілим, прийнятним і корисним для користувача.

Сукупний бал якості LLM-відповіді можна представити як зважену функцію оцінок, яку наведено у формулі (2.5):

$$Q_{LLM} = \alpha \cdot S + \beta \cdot R + \gamma \cdot F + \delta \cdot U \quad (2.5)$$

де $\alpha, \beta, \gamma, \delta$ - вагові коефіцієнти для оцінки:

S - структурованості,

R - релевантності,

F - формальності,

U - сприйняття користувачем.

Оцінка якості сегментації користувачів. Для підвищення релевантності рекомендацій система використовує кластеризацію профілів користувачів за ознаками поведінки (рівень активності, стабільність виконання, час доби, відгук на підказки).

Для кількісної оцінки якості кластеризації використовуються наступні метрики, яку наведено у формулі (2.6):

$$S(i) = \frac{b(i) - a(i)}{\max\{a(i), (i)\}} \quad (2.6)$$

$a(i)$ – середня відстань до об'єктів у тому ж кластері

$b(i)$ – середня відстань до найближчого сусіднього кластера

Індекс Девіса-Боулдіна (DBI), який наведено у формулі (2.7):

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right) \quad (2.7)$$

де σ_i - середнє внутрішньокластерне відхилення,

$d(c_i, c_j)$ - відстань між центрами кластерів i та j .

Крім того, вводиться показник покращення якості рекомендацій який наведено у формулі (2.8):

$$\Delta_{\text{rec}} = \frac{Acc_{\text{segm}} - Acc_{\text{base}}}{Acc_{\text{base}}} \times 100\% \quad (2.8)$$

де Acc_{segm} - точність рекомендацій після кластеризації, \

Acc_{base} - базова точність до кластеризації.

Результати демонструють підвищення точності рекомендацій у середньому на 12% для кластерів із вираженою структурою поведінки користувача. У таблиці 2.1 наведено типові класи, які було обрано під час класифікації користувачів за патернами поведінки на платформі, що розроблюється. Ці класи описують як користувачів ставлять та дотримуються цілей.

Таблиця 2.1 – Кластери поведінки користувачів

Сегмент	Опис поведінки	Основні риси
Planner	Працює з довгими цілями	Стабільність, чітке слідування пріоритетам
Explorer	Часто змінює задачі	Низька стабільність, схильність до експериментів
Minimal	Виконує лише базовий план	Високий рівень прокрастинації, мінімальна ініціативність

Джерело: сформовано автором на основі виконаного дослідження

Таким чином, управління інтелектуальними компонентами системи передбачає не лише їх побудову, а й безперервну оцінку якості з урахуванням поведінкового контексту, що забезпечує стабільність функціонування, релевантність підтримки та обґрунтованість адаптації.

Висновки до Розділу 2

У другому розділі магістерської роботи було здійснено всебічне дослідження об'єкта - інтелектуальної системи підтримки планування та досягнення особистих цілей. Показано, що потреба в таких рішеннях зумовлена зростаючим інформаційним і мотиваційним навантаженням на користувача, а також відсутністю комплексних цифрових асистентів, здатних до контекстної адаптації та персоналізованого супроводу.

Проведено детальну характеристику структурної архітектури системи, яка охоплює модуль управління цілями, модель користувача, компонент генерації рекомендацій, блок прогнозування, інтеграцію з великими мовними моделями (LLM), веб-інтерфейс і централізовану базу даних. Архітектура побудована за принципами мікросервісної модульності, пояснюваності рішень, масштабованості та гнучкого API-взаємозв'язку.

В межах підрозділів 2.3 було сформовано набір методів дослідження і моделювання компонентів системи штучного інтелекту. Зокрема, розглянуто задачу планування задач як задачу оптимізації з урахуванням дедлайнів, пріоритетів та когнітивного профілю користувача. Побудовано відповідну функцію втрат, запропоновано функцію корисності для прийняття рішень, а також розроблено структуру винагороди для агентів з навчання з підкріпленням (reinforcement learning). Наведено формальні критерії оцінки якості генерації LLM-відповідей, а також метрики ефективності кластеризації користувачів з прикладами сегментів і їх впливом на персоналізовані рішення.

Таким чином, результати цього розділу створили концептуальну та методологічну основу для наступного етапу - проектування інформаційного, математичного і програмного забезпечення системи, а також її практичної реалізації в рамках веб-прототипу.

РОЗДІЛ 3

РОЗРОБЛЕННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Моделювання та проектування бази знань прийняття інтелектуальних рішень

База знань реалізована як структурована реляційна база даних на основі MySQL, що інтегрована з іншими модулями системи через API та механізми трекінгу подій. У процесі реалізації було свідомо обрано однорівневу архітектуру бази знань, яку зображено на рис. 3.1, у межах якої сирі події, аналітично агреговані ознаки та прикладні структури (наприклад, сегменти, профілі, рекомендації) зберігаються в рамках єдиної реляційної бази MySQL. Такий підхід не передбачає виокремлення окремих шарів аналітики (raw → staging → mart), як це притаманно масштабованим дата-архітектурам, однак має низку переваг у контексті MVP-розробки.

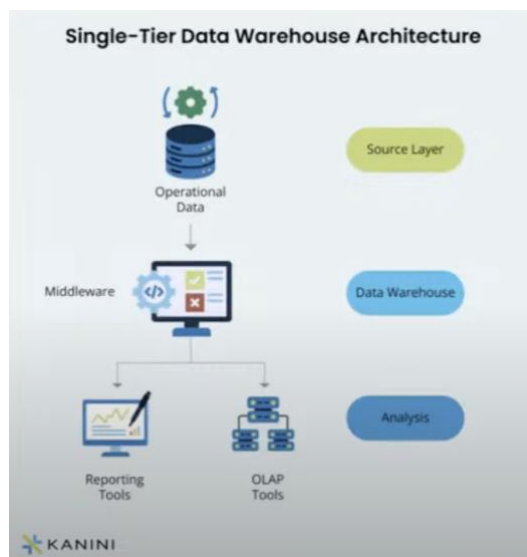


Рисунок 3.1 – Однорівнева архітектура бази знань

Джерело: сформовано автором на основі виконаного дослідження

По-перше, це дозволяє спростити схему розгортання та зменшити технічний борг, пов'язаний з ETL-процесами, синхронізацією джерел та

дублюванням логіки на різних рівнях. По-друге, об'єднання всіх структур даних в одній системі забезпечує низьку затримку доступу до знань для всіх модулів системи - зокрема, для адаптивного інтерфейсу, моделі рекомендацій та трекінгу. По-третє, це сприяє швидшій ітерації та перевірці гіпотез у процесі експериментальної розробки.

Разом з тим, однорівнева структура має обмеження в частині масштабування, контролю якості даних та ускладнює реалізацію версіонування моделей і логіки. У майбутньому, за умови переходу до production-рівня, можливе рефакторинг архітектури з виділенням окремих шарів зберігання знань, моделей і відображення.

3.1.1 Структура бази даних як представлення знань

Архітектура бази даних спроектована відповідно до принципів нормалізації та логічного поділу сутностей, що беруть участь у процесі цілепокладання. Основними таблиці наведено в таблиці 3.1

Таблиця 3.1 – Основні таблиці

Відношення	Призначення та момент запису
Users	Зберігає основні атрибути облікового запису. Запис: створюється при реєстрації користувача.
UserSettings	Зберігає налаштування системи. Запис: оновлюється при зміні параметрів інтерфейсу.
Categories	Сфери життя, в межах яких створюються цілі. Запис: при створенні першої цілі у сфері.
Aims	Цілі користувача. Запис: при додаванні нової мети.
SubAims	Підцілі. Запис: при деталізації складної цілі.
Tasks	Конкретні задачі. Запис: при генерації або додаванні вручну.

Продовження таблиці 3.1

Marks	Оцінка балансу сфер життя. Запис: вручну або за рекомендацією.
Diagrams	Візуалізації (графіки, «колесо життя»). Запис: після генерації діаграми.
AuthLogs	Інформація про логіни. Запис: при кожному вході.

Джерело: сформовано автором на основі виконаного дослідження

Для візуалізації структури бази даних використано діаграму зв’язків (ER), що поданана рис 3.2. Всі зв’язки реалізовано через зовнішні ключі, з урахуванням каскадних оновлень та видалень.

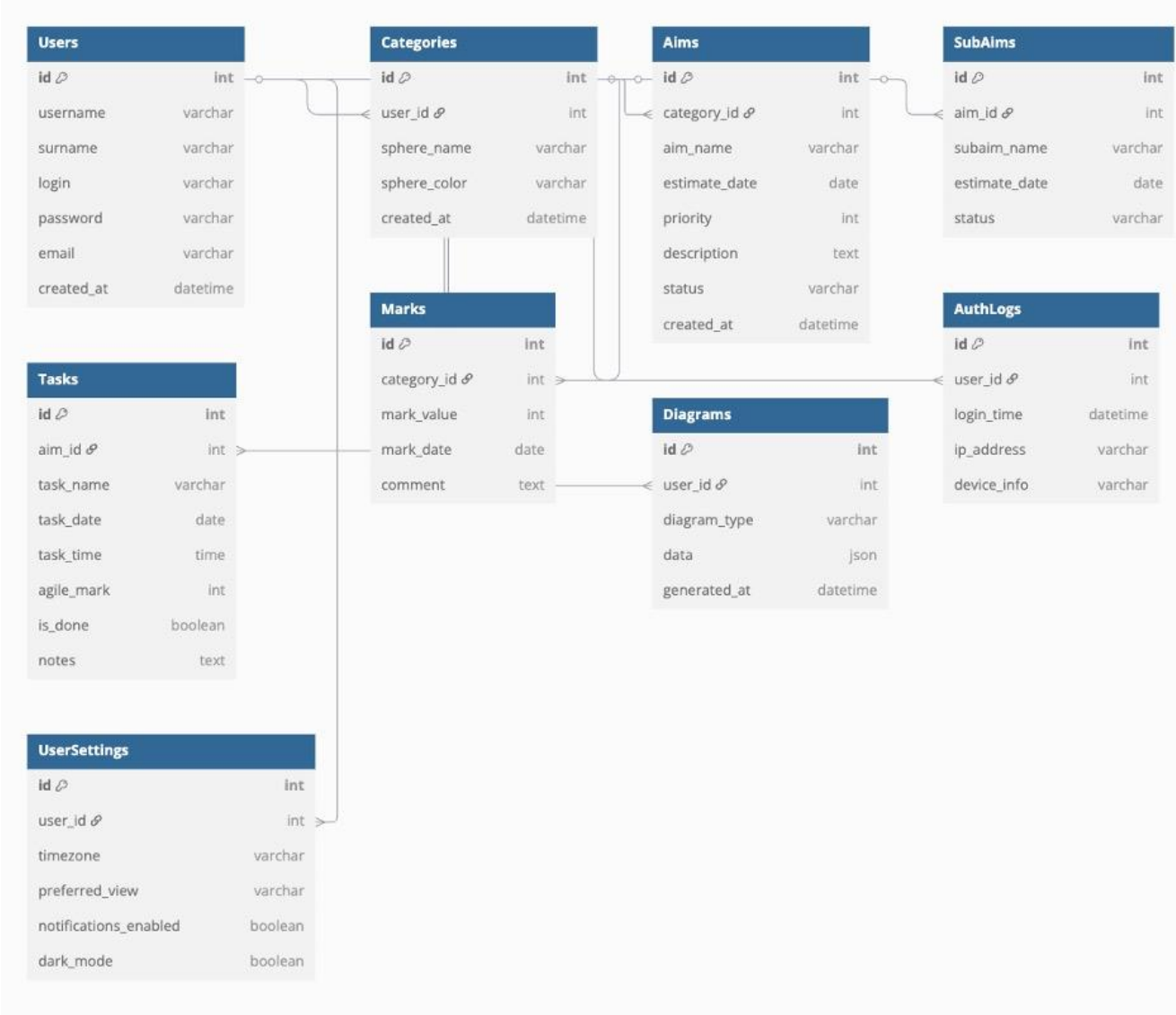


Рисунок 3.2 – Діаграма зв’язків

Джерело: сформовано автором на основі виконаного дослідження

Для побудови адаптивної системи було реалізовано наскрізний трекінг користувацьких дій. Усі ключові події на стороні клієнта (створення/редагування/видалення цілей, взаємодія з LLM, прийняття/відхилення рекомендацій, перемикання між задачами) фіксуються у вигляді структурованих об'єктів JSON, що передаються через API та зберігаються у таблиці events. Приклад структури наведено на рис 3.3

```
{
  "user_id": 1024,
  "event_type": "goal_updated",
  "timestamp": "2025-05-13T12:45:23Z",
  "source": "web_ui",
  "metadata": {
    "goal_id": 308,
    "field_changed": "title",
    "previous_value": "Завершити презентацію",
    "new_value": "Завершити фінальну презентацію",
    "edit_duration_sec": 18,
    "was_prompt_assisted": true
  },
  "session_id": "dfe902a1-f145-4e9e-8cd3-bb8d722f"
}
```

Рисунок 3.3 – Приклад запису трекінгу

Джерело: сформовано автором на основі виконаного дослідження

Кожен запис містить:

- ідентифікатор користувача (user_id) – значення що зберігається в session storage;
- тип події та її параметри;
- часову мітку та джерело запиту (web/UI);
- контекст дії (наприклад, які саме цілі чи задачі було змінено).

Таким чином, база знань не є статичною - вона оновлюється у реальному часі відповідно до активності користувача. База даних є одночасно аналітичною і продуктовою, що є збалансованим рішенням при поточних навантаженнях.

3.1.2 Агрегація та підготовка даних до кластеризації

Зібрані події обробляються в окремому модулі агрегації, реалізованому з використанням фреймворку Prefect. Агрегація здійснюється періодично за допомогою розкладу на основі cron (кожні 12 годин), що дозволяє регулярно оновлювати поведінкові профілі користувачів на основі накопичених подій. Потік агрегації включає зчитування даних з таблиці подій, обчислення інженерних ознак, формування матриці ознак, перевірку аномалій та збереження агрегованих результатів до таблиці profiles. Окремий крок flow відповідає за виклик моделі кластеризації, після чого результат оновлюється у таблиці segments., що формує профіль користувача на основі часових та поведінкових патернів. Серед використовуваних ознак :

- частка прийнятих рекомендацій;
- середній час на виконання задач;
- частота редагування цілей;
- реакція на нагадування;
- середній час між створенням задачі і її початком;
- час найбільшої активності (ранок, день, вечір, ніч);
- частота відхилення від плану (>1 год).

Матриця ознак зберігається в окремій таблиці profile_aggregation у тій самій базі даних - для подальшого навчання, аудиту та аналізу стабільності.

На основі сформованих профілів будується матриця ознак, яка подається на вхід кластеризаційним алгоритмам (K-means, GMM, DBSCAN). Результати зберігаються у таблиці segments, а відповідні сегменти використовуються для персоналізованої логіки рекомендацій.

3.1.3 Динамічне оновлення та збагачення бази знань

Система реалізована як така, що постійно самонавчається. У нашій реалізації це досягається шляхом автоматичного запуску Prefect-флоу агрегації, який аналізує події в таблиці events та перераховує ознаки профілю у profiles. Після накопичення більше 50 нових подій або змін у поведінкових метриках користувача (наприклад, середній час реакції, частота відхилення від плану) виконується повторна кластеризація, і сегмент користувача оновлюється у таблиці segments. Таким чином, поведінковий профіль завжди відповідає актуальному стану користувача без потреби у ручному втручанні.

Додатково, результати генерації LLM (структуровані SMART/OKR-цілі) кешуються у Redis для повторного використання протягом сесії, а також зберігаються у базі як історія запитів користувача. Основні сутності наведено в таблиці 3.2. Це дозволяє не лише перевіряти релевантність генерацій, а й формувати корпус прикладів для побудови інструкцій, покращення промтінгу та розробки системи навчання на власних даних (self-improvement loop).

Таблиця 3.2 – Основні сутності збережені в Redis

Ключ	Тип даних	Коли записується	Навіщо використовується
user:{id}:last_prompt_result	JSON string	після відповіді LLM	щоб не робити повторний запит до API при оновленні сторінки
user:{id}:active_goal_plan	JSON object	при генерації або редагуванні плану	для моментального виведення в UI без звертання до БД
session:dfе902a1-...:context	hash	при старті взаємодії з LLM або рекомендацією	для побудови діалогу, RL-контексту, уникнення повторень
aims:smart_templates	list	1 раз на старті (або по TTL)	шаблони SMART-цілей для генерації
segments:v1	set/map	після кластеризації	для швидкого доступу до сегменту користувача

Джерело: сформовано автором на основі виконаного дослідження

У результаті реалізована база знань виступає не лише сховищем даних, а й центральним адаптивним ядром системи, яке забезпечує накопичення, аналіз, персоналізацію та прийняття інтелектуальних рішень у реальному часі.

3.2 Розроблення користувацького інтерфейсу. Елементи та структура.

Користувацький інтерфейс системи реалізовано як односторінковий веб-застосунок, створений за допомогою фреймворку Streamlit, що дозволяє будувати інтерактивні інтерфейси безпосередньо на мові Python. Такий підхід забезпечує високу швидкість розробки, реактивну взаємодію з користувачем, а також глибоку інтеграцію з модулем машинного навчання та зовнішніми API.

Інтерфейс є реактивним: при зміні будь-якого вхідного параметра (наприклад, введення цілі або натисканні кнопки) система автоматично оновлює відповідні блоки без потреби перезавантаження сторінки. Для збереження стійкості сесії використовується об'єкт `st.session_state`, у якому зберігається `user_id`, згенеровані рекомендації та інші змінні стану.

Усі функціональні блоки інтерфейсу реалізовано у вигляді незалежних логічних модулів, що полегшує масштабування системи та її тестування. Структура дозволяє інтегрувати нові компоненти без впливу на основну логіку застосунку. Інтерфейс структурується у вигляді навігаційного меню з п'яти основних розділів: “Головна”, “Колесо життя”, “Цілі”, “Рекомендації”, “Аналітика”, які зображено на рис. 3.4

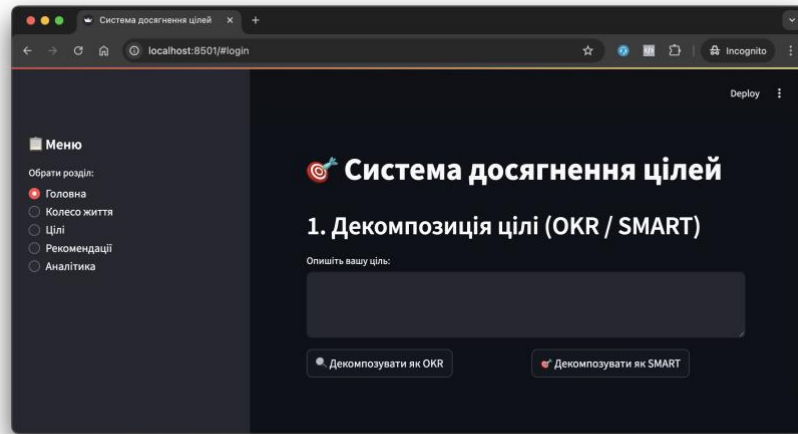


Рисунок 3.4 – Головна сторінка та навігаційне меню
Джерело: сформовано автором на основі виконаного дослідження

Головна сторінка. У цьому розділі користувач може описати особисту ціль у довільній формі та ініціювати її декомпозицію за шаблонами SMART або OKR. Генерація виконується через API-запит до бекенду, який, у свою чергу, звертається до OpenAI GPT-4o. Отриманий результат виводиться у вигляді структурованої текстової відповіді. Передбачено можливість збереження цілі для подальшого трекінгу. Приклад декомпозиції системою зображено на рис. 3.5 та рис. 3.6.

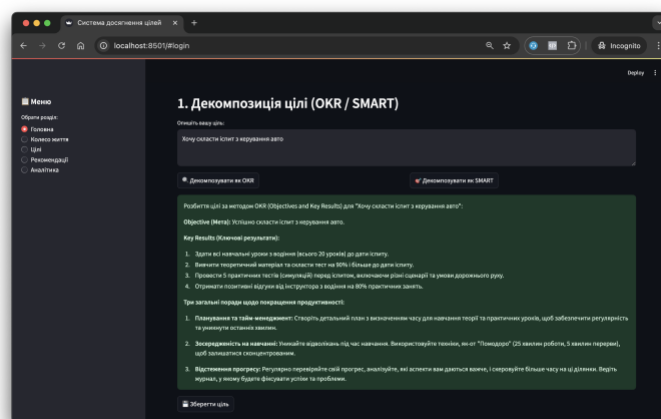


Рисунок 3.5 – Декомпозиція цілі за OKR
Джерело: сформовано автором на основі виконаного дослідження

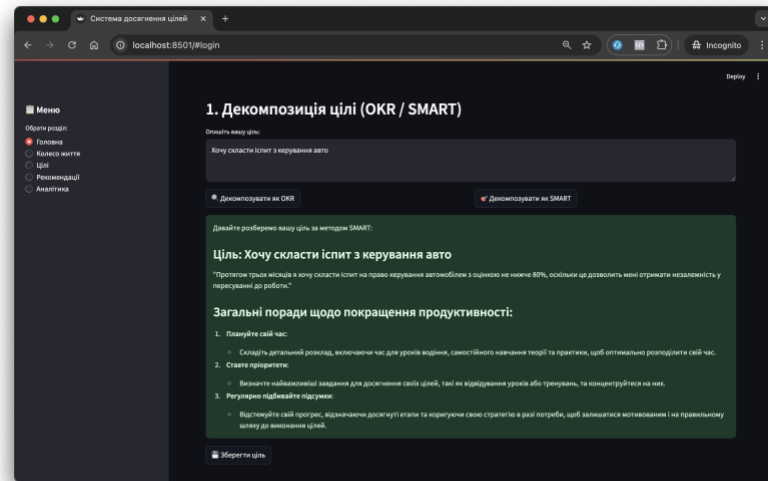


Рисунок 3.6 – Декомпозиція цілі за SMART

Джерело: сформовано автором на основі виконаного дослідження

Колесо життя. Розділ реалізує візуалізацію особистого балансу користувача за восьми ключовими сферами (здоров'я, кар'єра, духовність, тощо) у вигляді радарної діаграми. Значення можуть бути попередньо задані або введені користувачем. Діаграма дозволяє швидко виявити нерівномірність розвитку та може бути відправною точкою для формування нових цілей. Колесо життя зображено на рис. 3.7

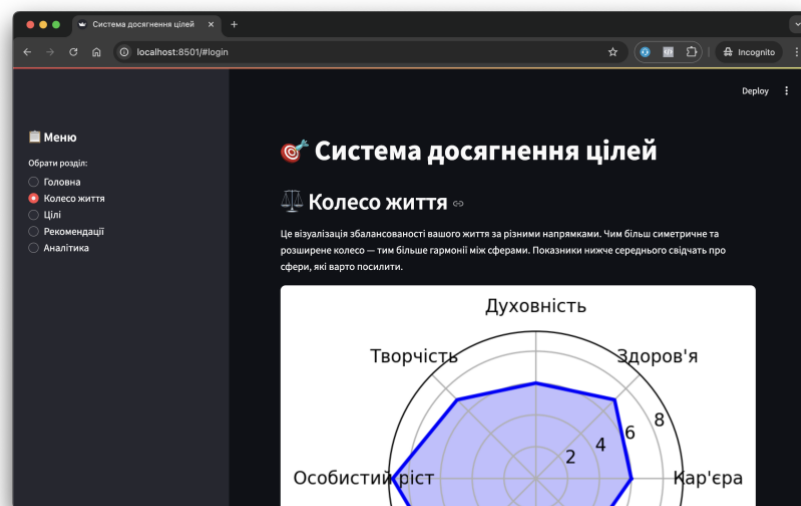


Рисунок 3.7 – Колесо життя

Джерело: сформовано автором на основі виконаного дослідження

Розділ цілей. У цьому розділі представлено список збережених користувачем цілей, які можуть бути розгорнуті для перегляду підзадач. Додатково реалізовано можливість оцінювання прогресу по кожній цілі за шкалою від 0 до 100%. У майбутньому передбачається реалізація інтерактивної Gantt-діаграми для візуального планування строків виконання задач. Розділ збереження і моніторингу цілей зображено на рис. 3.8.

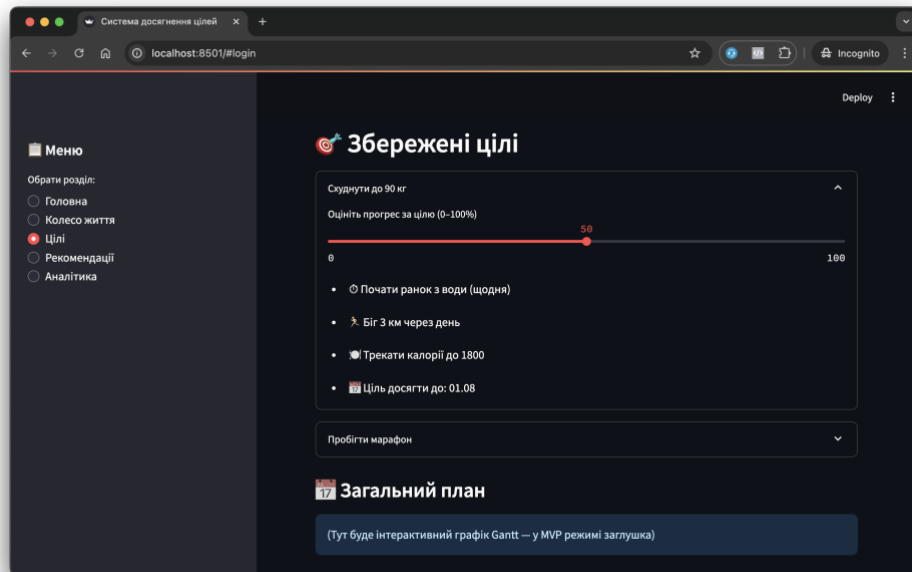


Рисунок 3.8 – Розділ цілей

Джерело: сформовано автором на основі виконаного дослідження

Рекомендації. Інтерфейс дозволяє користувачу отримати текстові рекомендації щодо покращення продуктивності, сформовані на основі попередніх дій та типового профілю. Запит до рекомендаційного модуля ініціюється вручну, а отримані поради супроводжуються кнопками “Це корисно” / “Пропустити”, які дозволяють зафіксувати фідбек. Фідбек надсилається на бекенд, зберігається в базі даних і використовується для персоналізації наступних рекомендацій. Розділ рекомендацій зображено на рис. 3.9.

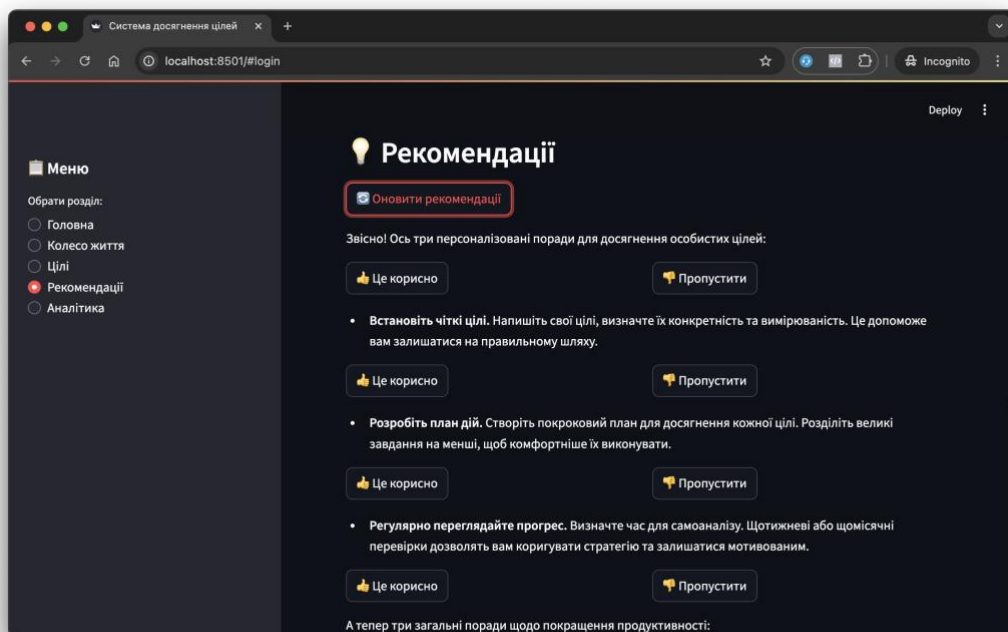


Рисунок 3.9 – Розділ рекомендацій

Джерело: сформовано автором на основі виконаного дослідження

Аналітика. Цей розділ містить серію візуалізацій, що відображають динаміку користувацької активності, зокрема: кількість завершених і активних цілей, середній відсоток прогресу, середню кількість задач на одну ціль, а також графік часу виконання задач. Графіки реалізовано за допомогою вбудованих Streamlit-інструментів (bar chart, line chart, area chart). Розділ аналітики цілей зображено на рис. 3.10.

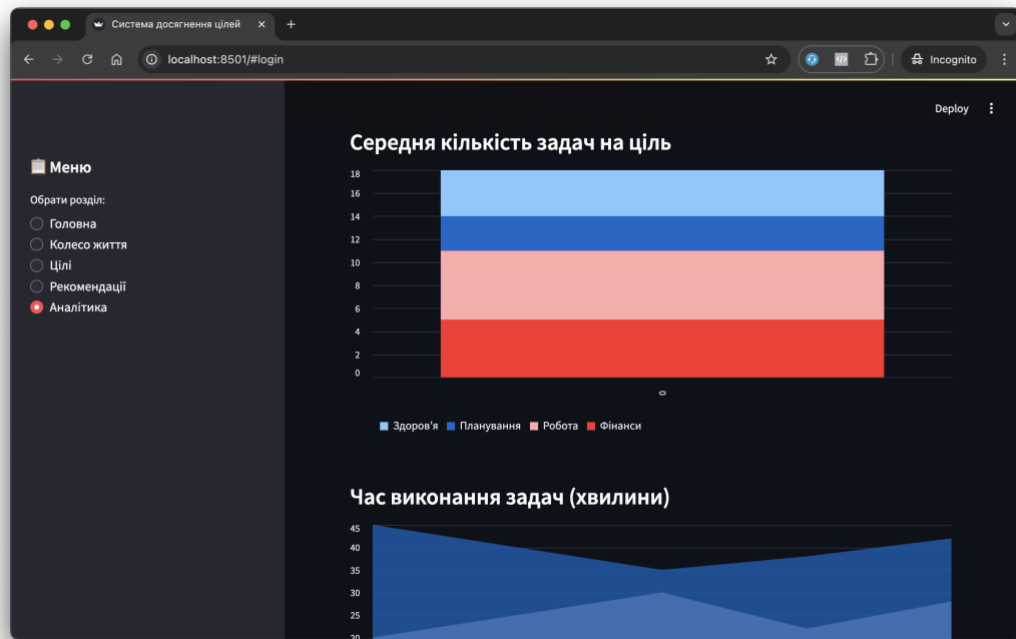


Рисунок 3.10 – Розділ аналітики

Джерело: сформовано автором на основі виконаного дослідження

Таким чином, інтерфейс передбачає доступ до раніше змодельованих та описаних компонент.

3.2.1 Принципи побудови інтерфейсу

При проєктуванні користувацького інтерфейсу було дотримано низки сучасних принципів UI/UX-дизайну, що спрямовані на зниження когнітивного навантаження, забезпечення фокусованої взаємодії та підвищення ефективності користування. Нижче наведено основні підходи, реалізовані у системі, та способи їх досягнення.

Принцип фокусування на ключовому сценарії. Інтерфейс кожного розділу побудовано навколо однієї основної дії: наприклад, у розділі “Головна” - це декомпозиція цілі, у “Цілі” - перегляд та оцінка прогресу, у “Рекомендаціях” -

взаємодія з персональними порадами. Всі вторинні функції винесено в бокове меню, що мінімізує відволікання користувача.

Мінімалізм і усунення візуального шуму. Інтерфейс реалізовано у стриманій кольоровій гамі без декоративних елементів. Кольори використовуються лише для функціональних акцентів (наприклад, синій для діаграм, зелений/червоний - для фідбеку). Відсутність зайвих блоків, анімацій чи стилістичних “навантажувачів” дозволяє користувачу зосередитись на основному змісті.

Принцип “одна дія - один екран”. Кожен розділ містить лише ті елементи, які необхідні для виконання конкретного завдання. Наприклад, у розділі “Рекомендації” одночасно відображаються лише актуальні поради і кнопки фідбеку - без додаткової інформації, яка могла б перешкодити прийняттю рішення.

Контекстна адаптація. Інтерфейс не виводить надлишкової інформації, а реагує на дії користувача. Наприклад, рекомендації не відображаються до моменту натискання кнопки “Оновити”, що дозволяє уникати непотрібного оновлення вмісту. Після оцінки поради вона не зникає, а залишається на екрані - це підсилює відчуття контролю.

Реактивність та збереження стану. Уся взаємодія реалізована з використанням `st.session_state`, що дозволяє зберігати інформацію про користувача, сесію, згенеровані рекомендації, введені цілі тощо. Завдяки цьому при кожному повторному рендері (наприклад, після натискання кнопки) не відбувається очищення сторінки чи втрати контексту, що підвищує комфорт використання.

Простота підтримки та розширення. Інтерфейс реалізовано в рамках єдиного Streamlit-додатку, де кожен розділ є незалежним модулем. Це дозволяє розробнику швидко додавати нові функції або змінювати логіку існуючих блоків, не порушуючи цілісність системи. Наприклад, додавання блоку “Аналітика” було реалізовано без зміни базового шаблону сторінки.

3.3 Проектування забезпечувальних підсистем системи штучного інтелекту. Реалізація системи.

3.3.1 Інформаційне забезпечення.

Для запуску MVP-версії системи, орієнтованої на персоналізовану підтримку планування, виникла потреба у створенні початкового сету даних, що дозволив би тренувати та тестувати ключові компоненти системи: рекомендаційний модуль, кластеризатор профілів користувачів і LLM-промптинг. У першому етапі було організовано апробацію системи на внутрішніх тестових користувачах. Було зібрано перші 100+ сесій взаємодії, які містили реальні запити до генерації цілей, виконання або відхилення задач, часові параметри, реакцію на рекомендації та інші події. Ці дані лягли в основу початкових шаблонів ознак для побудови профілю користувача та його кластеризації. Для подальшого розширення обсягу даних використовувалась методика генерації синтетичних подій на основі шаблонів поведінки, отриманих з реальних сесій. Сценарії взаємодії симулювались з урахуванням типових патернів активності (наприклад, ранковий або вечірній режим, швидке/повільне завершення задач, високий/низький рівень прийняття підказок). Це дозволило сформувати збалансовані датасети з контрольованим розподілом класів для задач класифікації і сегментації. Для генерації синтетичних даних було використано сценарно-умовну модель, що поєднує rule-based підхід із стохастичними параметрами. Поведінкові шаблони, отримані з апробаційних сесій, класифікувались за типом користувача, після чого для кожного типу було сформовано набір сценаріїв з детермінованими ймовірностями: частка завершених задач, середній час відгуку, рівень прийняття рекомендацій, схильність до редагування цілей тощо. Події генерувались скриптом, який відтворює сесії взаємодії користувача із системою з урахуванням варіативності таймінгу та контексту. Отримані синтетичні дані зберігались у форматі, ідентичному структурі реального трекінгу, що дозволило використовувати їх для навчання моделей без змін у пайплайні. Окрему роль відіграло використання

великих мовних моделей (LLM) для генерації текстових формулювань цілей, задач та реакцій. На основі заданих параметрів (категорія, дедлайн, стиль спілкування, рівень деталізації) LLM-агент формував правдоподібні вхідні дані, які використовувались як тестовий контент для перевірки роботи інтерфейсу, промптінгу та моделі перетворення тексту в структуру SMART/OKR.

Завдяки поєднанню частково зібраних реальних сесій, контрольованої симуляції та генерації через LLM, було сформовано повноцінний базовий корпус для запуску адаптивних компонентів системи в умовах обмежених обсягів даних.

Інформаційне забезпечення системи виконує ключову роль у забезпеченні її інтелектуальних властивостей, зокрема персоналізації, адаптації поведінки та підтримки цілепокладання в динамічному середовищі. Особливістю реалізованої архітектури є підтримка неперервного самонавчання системи на основі накопичуваних даних.

Агрегація подій та формування поведінкових векторів. Усі дії користувача (створення цілей, генерація декомпозицій, фідбек на рекомендації, оцінка прогресу) реєструються як події у системі. Зібрані події обробляються в окремому модулі агрегації, який реалізовано за допомогою бібліотеки Prefect. Агент виконує періодичні (щогодинні) ETL-запуски, що формують поведінкові вектори користувачів на основі таких фічей, як:

- кількість створених цілей і задач;
- частота взаємодій з системою;
- кількість/частка позитивних або негативних реакцій на рекомендації;
- рівень дотримання дедлайнів;
- середній прогрес по цілях.

Згенеровані вектори зберігаються в агрегаційній таблиці у тій самій базі даних, що дозволяє використовувати їх для кластеризації без необхідності додаткових сховищ.

Побудова сегментів через кластеризацію. На основі поведінкових векторів реалізовано кластеризацію користувачів за допомогою алгоритмів K-Means або DBSCAN. Отримані сегменти дозволяють ідентифікувати типи користувацької поведінки (наприклад, “Прокрастинатор”, “Фокусований”, “Непоследовний”) і використовувати цю інформацію для подальшої адаптації системи. Приклад візуалізації кластерів у PCA-просторі наведено на рис. 3.10

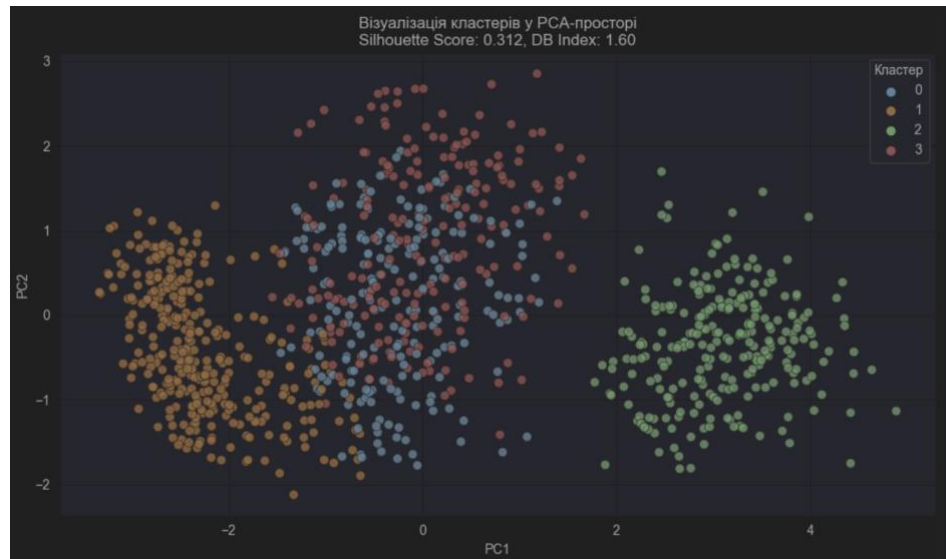


Рисунок 3.10 – Кластери користувачів у PCA просторі

Джерело: сформовано автором на основі виконаного дослідження

Кластери підбрано так, щоб кількість користувачів була приблизно однаковою в кожному з них. Кількість зображена рис. 3.11.

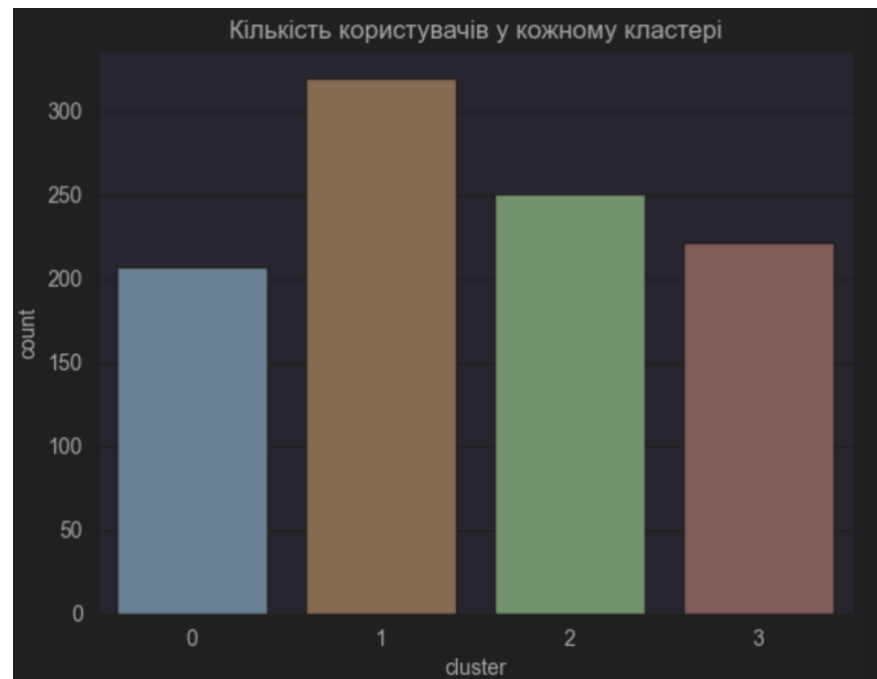


Рисунок 3.11 – Кількість користувачів по кластерам
Джерело: сформовано автором на основі виконаного дослідження

Збір та використання фідбеку. Кожна рекомендація, що надається користувачу, може бути оцінена ним як “корисна” або “неактуальна”. Ці дії фіксуються у таблиці `recommend_feedback`, де зберігаються:

- унікальний ідентифікатор користувача;
- текст рекомендації;
- тип фідбеку (positive / negative);
- timestamp.

Цей фідбек не лише зберігається, а й використовується для модифікації подальших рекомендацій. Приклад реалізації в інтерфейсі наведено на рис. 3.12.

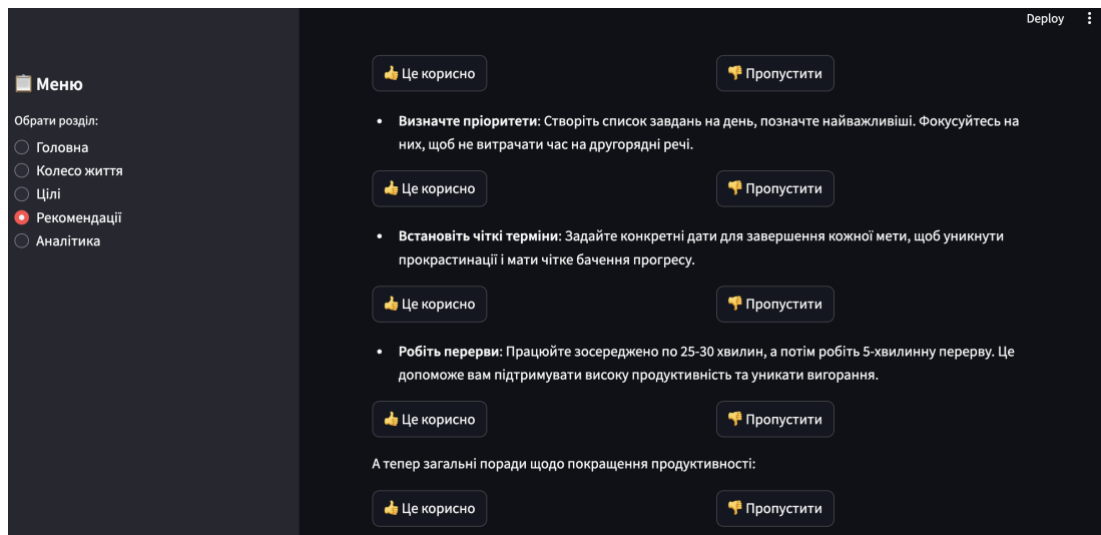


Рисунок 3.12 – Збір відгуку для налаштування рекомендацій
Джерело: сформовано автором на основі виконаного дослідження

Реалізація адаптивної поведінки через RL-подібну логіку. При формуванні наступних рекомендацій система перевіряє, чи мав користувач позитивний досвід з попередніми порадами. Якщо такі є - система надає подібну рекомендацію або повторює вже позитивно оцінену, імітуючи базову політику підкріплення. У разі переважання негативного фідбеку - система генерує нові варіанти, уникаючи попередніх шаблонів.

Таким чином, реалізовано цикл зворотного зв'язку, який наближає поведінку системи до агента з підкріпленням (reinforcement learning), хоч і в обмеженій формі.

Роль інформаційного ядра. На відміну від класичних застосунків, у запропонованій системі база знань виступає не лише як пасивне сховище, а як активне адаптивне середовище, що:

- акумулює дані користувача;
- формує на їх основі поведінковий портрет;
- дозволяє алгоритмічну адаптацію рекомендацій;
- виступає точкою входу для ML-модулів кластеризації та аналітики.

3.3.2 Програмне забезпечення.

На рисунку представлено загальну архітектуру програмного забезпечення системи, що включає основні логічні компоненти та їхню взаємодію. Схематично архітектуру наведено на рис. 3.13

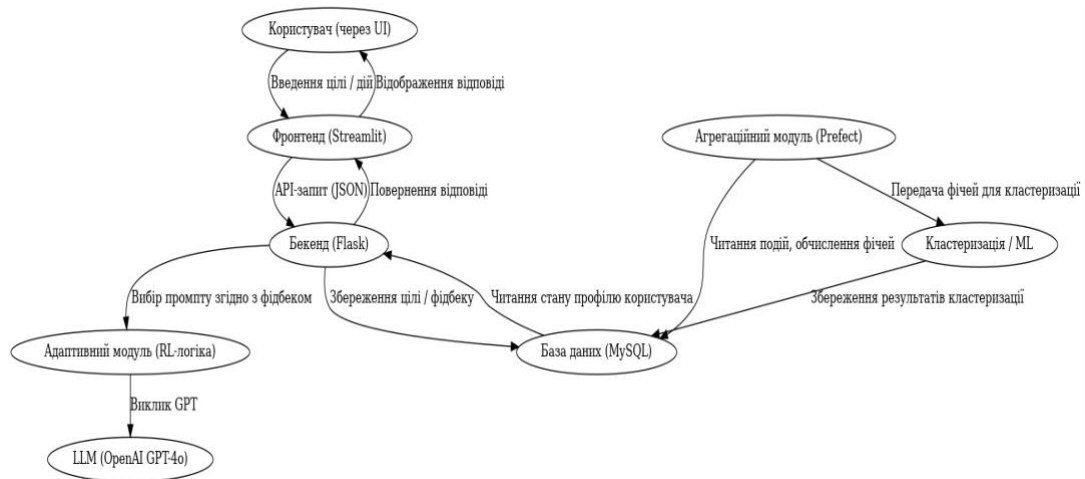


Рисунок 3.13 – Архітектура ПЗ

Джерело: сформовано автором на основі виконаного дослідження

Користувач здійснює введення через веб-інтерфейс, реалізований у Streamlit, що надсилає API-запити до бекенду, розгорнутого на Flask. Запити, пов'язані з генерацією цілей чи рекомендацій, надходять до окремого адаптивного модуля - RL-компонента, який модифікує запит до мовної моделі на основі історії фідбеку користувача (наприклад, переважно позитивні реакції на короткі поради - система продовжує генерувати аналогічні; негативні реакції - виключає певні шаблони). Інтеграція з LLM реалізована через HuggingFace API [10].

Після формування адаптованого запиту RL-модуль передає його до зовнішньої LLM (OpenAI GPT-4o), результат якої повертається через Flask на фронтенд. Для генерації цілей використано GPT-4, відповідно до специфікацій [9]. Усі дії користувача логуються у базу даних (MySQL), яка також зберігає інформацію про задачі, цілі, фідбек, класифікації та поведінкові фічі.

Періодично система Prefect виконує агрегаційні задачі: обчислення поведінкових векторів, передавання їх до модуля машинного навчання для кластеризації користувачів, та оновлення відповідних полів у базі. Отримані сегменти використовуються як додатковий параметр у модулі адаптації, що дозволяє системі реалізовувати персоналізовану логіку надання порад. Методи факторизації матриць застосовуються у класичних рекомендаційних системах [6].

Для реалізації взаємодії між клієнтським інтерфейсом та серверною логікою було розроблено низку REST-ендпоїнтів. Вони забезпечують виконання основних функцій системи: генерацію декомпозицій, отримання рекомендацій, надсилання фідбеку та, за потреби, авторизацію користувача. Всі запити обмінюються даними у форматі JSON через протокол HTTP. У таблиці 3.3 наведено опис основних API-запитів, які забезпечують функціонування системи:

Таблиця 3.3 – Опис основних API-запитів

URL	Метод	Призначення	Параметри входу	Формат відповіді
/decompose	POST	Генерація SMART/OKR-декомпозиції для заданої цілі	goal, type, user_id	JSON: {"result": ...}
/recommend	POST	Генерація персоналізованої рекомендації для користувача	user_id	JSON: {"recommendations": ...}
/recommend_feedback	POST	Надсилання реакції на рекомендацію	user_id, text, feedback	200 OK
/auth	POST	Авторизація користувача за email	email, password (або лише email)	user_id

Джерело: сформовано автором на основі виконаного дослідження

У межах розробки серверної частини системи були реалізовані запити до реляційної бази даних (MySQL), які забезпечують зберігання, модифікацію та отримання інформації про користувачів, цілі, задачі, рекомендації та пов'язані події.

Для демонстрації принципів роботи з інформаційною моделлю системи на рис. 3.14 наведено приклади найбільш типових запитів: додавання нових цілей, збереження задач, обробка фідбеку, отримання профілю та витяг агрегованих характеристик користувача. Усі запити реалізовано із захистом від SQL-ін'єкцій через параметризовані шаблони.

```
DB_CONFIG = {
    "host": os.getenv("DB_HOST"),
    "user": os.getenv("DB_USER"),
    "password": os.getenv("DB_PASSWORD"),
    "database": os.getenv("DB_NAME")
}

connection = mysql.connector.connect(**DB_CONFIG)
cursor = connection.cursor()

def save_interaction(goal, goal_type, response, user_id): 2 usages new *
    cursor.execute( operation: """
        insert into interactions (goal, goal_type, response, user_id)
        values (%s, %s, %s, %s)
    """, params: (goal, goal_type, response, user_id))
    connection.commit()

def save_recommendation_feedback(user_id, recommendation, feedback): 2 usages new *
    cursor.execute( operation: """
        insert into recommend_feedback (user_id, recommendation, feedback)
        values (%s, %s, %s)
    """, params: (user_id, recommendation, feedback))
    connection.commit()
```

Рисунок 3.14 – SQL запити до БД

Джерело: сформовано автором на основі виконаного дослідження

З метою забезпечення консистентності таблиці users, а також автоматичного створення запису при появі нового user_id в таблиці подій, у базі даних було реалізовано тригер, який ініціюється після вставки запису в таблицю interactions. Використання в коді наведено на рис. 3.15. У разі, якщо відповідного користувача ще не існує в системі, тригер автоматично додає його до таблиці users. Цей підхід дозволяє зберігати узгодженість даних без потреби в ручному додаванні користувачів або додаткових перевірок на рівні бізнес-логіки.

```

1 create definer = root@localhost trigger insert_user_if_not_exists
2   after insert
3   on interactions
4   for each row
5 BEGIN
6   IF NOT EXISTS (
7     SELECT 1 FROM users WHERE ulid_id = NEW.user_id
8   ) THEN
9     INSERT INTO users (ulid_id)
10    VALUES ( ulid_id NEW.user_id);
11   END IF;
12 END;

```

Рисунок 3.15 – Триггер на оновлення даних

Джерело: сформовано автором на основі виконаного дослідження

Для автоматизації періодичних обчислень, пов'язаних з обробкою подій користувача, побудовою агрегованих поведінкових характеристик та підготовкою даних для моделей кластеризації, у системі використовується платформа оркестрації Prefect [12].

У межах реалізованої системи створено окремий flow, який виконується за розкладом (cron або за подією), та включає такі етапи:

1. Зчитування нових подій із таблиці interactions;
2. Формування агрегованих поведінкових фічей для кожного користувача:
 - середня кількість активностей на день;
 - частота позитивного фідбеку;
 - стабільність взаємодії;
 - середній час до завершення задач;
 - рівень відповідності дедлайнам;
3. Оновлення агрегованої таблиці aggregated_user_profiles;
4. Ініціація кластеризації користувачів на основі оновлених векторів.

Вся логіка реалізована у вигляді Python-скрипту з використанням бібліотеки prefect. Пайплайн виконується локальним агентом, який запускається як системний процес або вручну в середовищі розробника. Для забезпечення прозорості моніторингу використовується Prefect UI, який наведено на рис. 3.16,

де аналітик може відстежити статус останніх запусків, переглянути логи, тривалість виконання окремих етапів та діагностувати помилки.

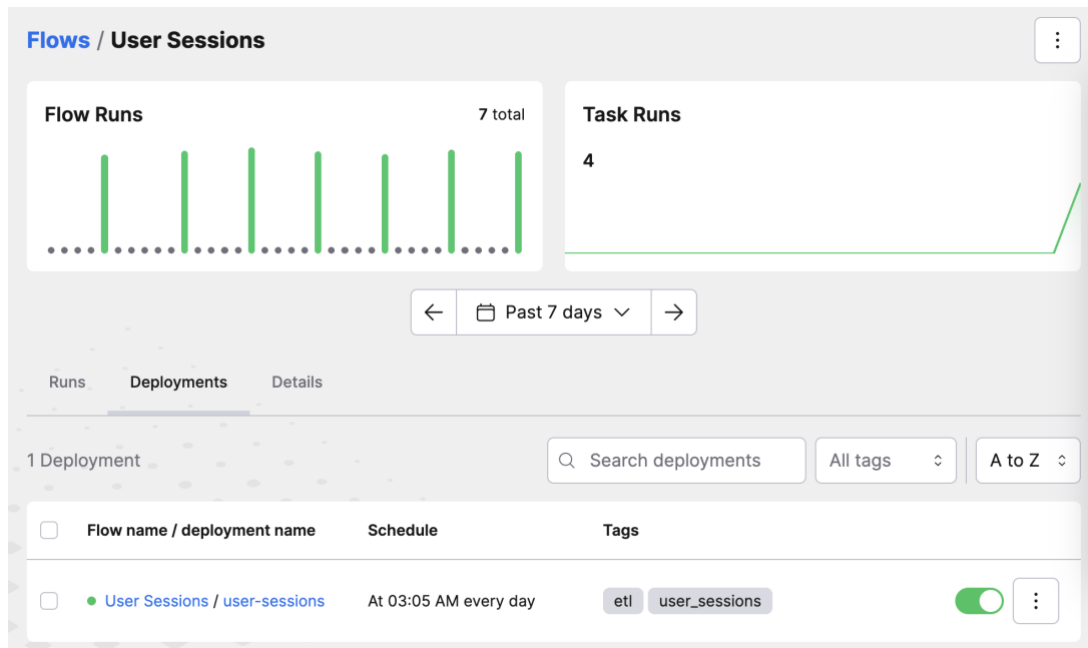


Рисунок 3.16 – Prefect UI

Джерело: сформовано автором на основі виконаного дослідження

У якості основного середовища розробки використовувалося PyCharm - інтегроване середовище розробки для мови Python, розроблене компанією JetBrains.

Вибір цього інструменту обумовлений наступними перевагами:

- Повна підтримка стеку технологій проєкту: Python, Flask, Streamlit, SQLAlchemy, Prefect, бібліотеки машинного навчання (scikit-learn, XGBoost);
- Інтеграція з Git: спрощення контролю версій та командної роботи;
- Покрокова налагоджувальна система (debugger): дозволяє ефективно тестувати та відслідковувати логіку роботи API, обробку даних і виклики LLM;
- Інтелектуальний автодоповнювач коду: пришвидшує розробку та знижує ризик синтаксичних помилок;
- Гнучка інтеграція з Python-оточеннями: зокрема, через virtualenv або conda, що критично для підтримки залежностей проєкту;
- Можливість запуску багатьох конфігурацій: одночасне тестування Flask-сервера, Streamlit-інтерфейсу та Prefect-агента.

Використання PyCharm дозволило забезпечити повноцінне середовище розробки без додаткових фінансових витрат.

3.3.3 Технічне забезпечення

Технічне забезпечення розробленої системи охоплює апаратні, системні та програмні ресурси, необхідні для коректної роботи інтерфейсу, серверної логіки, мовної моделі та аналітичних модулів. З урахуванням обсягу та складності завдань, а також етапу розробки (MVP), архітектура реалізована у вигляді локальної односерверної конфігурації, розгорнутої на персональному комп'ютері розробника.

Уся система функціонувала під час розробки на апаратному забезпеченні з наступними характеристиками:

- Процесор: Apple M1 (8-ядерний, ARM-архітектура);
- Оперативна пам'ять: 16 ГБ;
- Операційна система: macOS Ventura 13.4;
- Сховище: SSD 512 ГБ;
- Мережеве з'єднання: стабільне з доступом до інтернету для виклику

зовнішніх LLM API.

Така конфігурація є типовою для сучасного ноутбука розробника і цілком достатньою для запуску всіх компонентів системи в обмеженому режимі (один користувач, послідовні запити).

На локальному комп'ютері встановлено Python 3.11 та створено ізольоване віртуальне середовище з використанням `venv`, що забезпечує контроль над залежностями. Усі необхідні компоненти встановлювалися через `pip`.

Ключові залежності:

- Flask - серверна логіка та API;
- Streamlit - фронтенд-інтерфейс;

- openai - інтеграція з GPT-моделлю;
- mysql-connector-python - зв'язок із MySQL;
- prefect - фреймворк для оркестрації обчислень;
- scikit-learn, matplotlib, xgboost - ML і візуалізація.

Усі сервіси були розгорнуті локально, але в ізольованих процесах:

- Flask-сервер запускався як окремий бекенд-сервіс на порту 5002;
- Streamlit виконував роль фронтенду та звертався до Flask через HTTP;
- MySQL піднято через локальний демон, створена окрема база goal_assistant;
- Prefect-агент запускався окремо в CLI-режимі (prefect agent start), або вручну як окрема задача;
- Redis (опціонально) - для кешування, встановлювався через Homebrew.

Завдяки такій топології, компоненти могли масштабуватися незалежно, хоча запуск здійснювався з одного пристрою.

Вимоги до переносу

Реалізована архітектура легко переноситься на сервер або в контейнеризоване середовище (Docker), оскільки:

- усі компоненти запускаються незалежно;
- параметри конфігурації виносять в .env;
- немає залежностей від GUI чи спеціального оточення.

Для розгортання на сервері достатньо:

- інстансу з Python 3.11;
- встановленого pip/poetry;
- інсталяції бази MySQL або використання RDS;
- наявності API-ключа OpenAI.

Продуктивність

У рамках MVP система демонструвала стабільну роботу з відгуком у межах:

- декомпозиція цілі: 2–4 секунди (включно з викликом OpenAI);
- генерація рекомендацій: до 5 секунд;
- зчитування з MySQL: <100 мс (при незначному навантаженні).

Поточна архітектура не призначена для високого навантаження, але є оптимальною для:

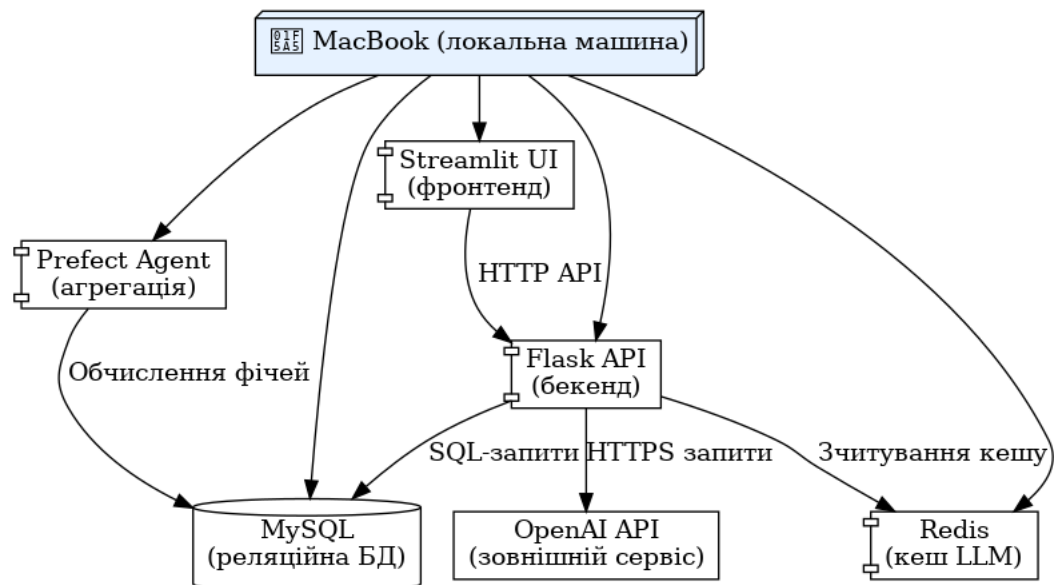
- індивідуального використання;
- демонстрацій;
- подальшої інкубації SaaS-продукту.

Перспективи масштабування

Попри простоту технічної реалізації, обрана структура дозволяє масштабування за такими напрямками:

- перенесення бази на окремий сервер або в хмару (MySQL → AWS RDS);
- запуск Flask та Streamlit у контейнерах Docker;
- оркестрація Prefect через Prefect Cloud;
- заміна OpenAI API на локальні моделі (наприклад, LLaMA через Ollama або LM Studio);
- розгортання вебінтерфейсу на окремому хості з TLS-сертифікатом.

На рис. 3.17 зображено фізичне розгортання компонентів системи на одному пристрої - персональному комп'ютері (MacBook), що виступає центральним вузлом виконання всіх програмних сервісів. Така конфігурація відповідає вимогам створення MVP (minimum viable product) та дозволяє забезпечити повноцінну функціональність без потреби в хмарній інфраструктурі.



UML Deployment Diagram — MVP реалізація системи цілепокладання

Рисунок 3.17 – фізичне розгортання компонентів системи

Джерело: сформовано автором на основі виконаного дослідження

Ключові компоненти та їх взаємодія:

- Streamlit UI - реалізує користувацький інтерфейс, запускається у веббраузері, здійснює HTTP-запити до Flask API для генерації рекомендацій, декомпозицій цілей та взаємодії з базою даних.
- Flask API - бекенд-застосунок, який обробляє вхідні запити, взаємодіє з базою даних (MySQL), викликає зовнішній OpenAI API для генерації тексту, а також працює з Redis для кешування результатів викликів LLM.
- MySQL - основне сховище даних системи. Тут зберігаються користувачі, цілі, задачі, фідбек, а також агреговані поведінкові вектори.
- Prefect Agent - періодично виконує ETL-завдання: обробляє нові події користувача, обчислює поведінкові характеристики та оновлює таблиці з агрегованими профілями. Запускається як окремий процес або планувальник.
- Redis - використовується для збереження результатів останніх запитів до LLM з метою зменшення затримок та навантаження на зовнішній API.
- OpenAI API - єдиний зовнішній компонент, який викликається з Flask для генерації семантично релевантних текстів (декомпозицій, порад тощо). Взаємодія з ним здійснюється через захищене з'єднання HTTPS.

Завдяки тому, що всі сервіси працюють на одному вузлі, система не потребує мережевої маршрутизації між модулями. Це дозволяє уникнути затримок у локальній взаємодії та спрощує налагодження під час розробки.

3.3.4 Організаційно-економічне забезпечення

Організаційно-економічне забезпечення створення, впровадження та подальшого розвитку інтелектуальної системи планування та досягнення цілей включає сукупність технічних, кадрових, програмних, часових та вартісних ресурсів, необхідних для реалізації повного циклу - від концептуального проєктування до практичного тестування MVP. Проєкт був реалізований у рамках індивідуального дослідницько-розробницького процесу. Усі основні компоненти створено однією особою - розробником системи, що значно оптимізувало як часові, так і фінансові витрати. Така організаційна модель відповідає методології lean-розробки, що передбачає створення мінімально життєздатного продукту (MVP) із максимальним використанням відкритих та безкоштовних рішень. Значення що модлювалися для розрахунків наведено в таблиці 3.4 та таблиці 3.5

Таблиця 3.4 – Місячні витрати на підтримку продукту

Категорія витрат	Орієнтовна сума (\$/міс.)
Оплата праці розробника	3,000
Маркетинг	1,000
Інфраструктура (OpenAI, сервери)	100
Служба підтримки	300
Домен, хостинг, непередбачені	100
Загальні витрати	4,500

Джерело: сформовано автором на основі виконаного дослідження

Таблиця 3.5 – Використані програмні інструменти

Компонент	Засіб	Вартість
Мова програмування	Python 3.11	Безкоштовно
Інтерфейс	Streamlit	Безкоштовно
Бекенд	Flask	Безкоштовно
База даних	MySQL (локальний інстанс)	Безкоштовно
Оркестрація	Prefect OSS	Безкоштовно
Візуалізація	Matplotlib, Plotly	Безкоштовно
Розробка	PyCharm Community Edition	Безкоштовно
LLM API	OpenAI GPT-4o	≈ 5–10 \$/міс.

Джерело: сформовано автором на основі виконаного дослідження

З метою більш реалістичної оцінки точок беззбитковості було враховано:

- маркетинговий бюджет у розмірі 2,000 \$/міс.;
- середній рівень чурну (відтоку підписників) на рівні 25% (середній LTV - 4 місяці)
- фіксовану ціну підписки \$4.99.

Результати моделювання, наведені на рис. 3.18, свідчать, що при такому сценарії система залишається збитковою до моменту досягнення аудиторії понад 8,000 активних користувачів. У діапазоні 1,000–5,000 користувачів чистий прибуток залишається від’ємним, що свідчить про критичну роль ефективного маркетингу та зростання бази.

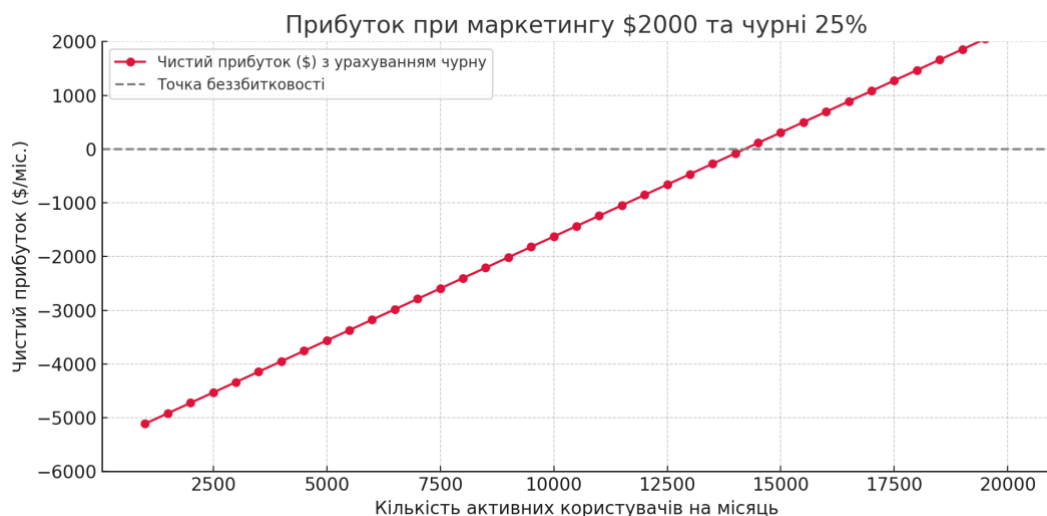


Рисунок 3.18 – моделювання періоду беззбитковості проекту

Джерело: сформовано автором на основі виконаного дослідження

Висновки до Розділу 3

У третьому розділі було розроблено та реалізовано повний цикл проєктних рішень, необхідних для створення інтелектуальної системи планування та досягнення цілей на основі сучасних підходів до персоналізації, когнітивного моделювання та використання штучного інтелекту.

У підпункті 3.1 детально описано архітектуру бази знань системи, яка поєднує реляційне зберігання подій, агреговані характеристики користувача та динамічну інтеграцію з машинним навчанням. Особливу увагу приділено підходам до агрегації даних та їх підготовці для поведінкової кластеризації, що формує основу адаптивної логіки рекомендацій.

У підпункті 3.2 реалізовано інтерфейсну частину системи з використанням фреймворку Streamlit. Сформовано низку функціональних модулів (декомпозиція цілей, перегляд рекомендацій, колесо життя, аналітика), що забезпечують зручну взаємодію з користувачем і підвищують прозорість роботи системи.

У підпункті 3.3 систематизовано технічну, програмну, інформаційну та економічну інфраструктуру системи. Визначено ключові залежності, описано взаємодію модулів (у тому числі через REST API), а також наведено UML-діаграми розгортання та взаємодії. Окрему увагу приділено реалізації адаптивної поведінки системи через простий reinforcement learning-підхід, що дозволяє враховувати фідбек користувача при формуванні наступних рекомендацій.

У фінальній частині розділу проведено фінансове моделювання та побудовано економічну модель підписки. На основі розрахунків визначено базові бенчмарки для виходу системи на точку беззбитковості, а також оцінено потенціал масштабування продукту в комерційне SaaS-рішення.

Результати третього розділу формують повноцінну проєктно-реалізаційну базу системи та забезпечують обґрунтування її функціональної цілісності, технічної реалізованості та економічної перспективності.

ВИСНОВКИ

У межах магістерської роботи на тему *«Створення платформи для планування та досягнення цілей з використанням штучного інтелекту та персоналізованих рекомендацій»* було реалізовано комплексний підхід до розв’язання проблеми цифрового супроводу користувачів у процесі цілепокладання та самоменеджменту. Сформульована наукова проблема полягала в необхідності створення інтелектуальної системи, здатної враховувати індивідуальні патерни поведінки, мотиваційні характеристики та контекстну динаміку змін, що притаманна реальним життєвим сценаріям. На основі глибокого аналізу предметної області, сучасних підходів до когнітивного планування, а також огляду існуючих інструментів було виокремлено функціональні обмеження, які не дозволяють існуючим рішенням забезпечити належний рівень персоналізації, адаптивності та гнучкості.

Відповідно до сформульованих цілей та завдань у ході дослідження було спроектовано архітектуру інтелектуальної системи, що включає модуль користувацької взаємодії, базу знань, рекомендаційний блок, компонент адаптації та механізми генерації текстових підказок на основі великих мовних моделей. Особлива увага приділялася поведінковому моделюванню користувача - розроблено процес збору, обробки та агрегації подій, на основі яких формуються вектори ознак, що надалі використовуються для кластеризації профілів та налаштування рекомендацій. Інтеграція з зовнішнім API мовної моделі забезпечує генерацію змістовних та індивідуалізованих цільових декомпозицій, зокрема у форматах OKR та SMART, які реалізовано безпосередньо в інтерфейсі користувача.

Практична реалізація системи охоплює повний життєвий цикл - від введення текстової мети до збереження результатів та аналізу поведінкових змін. Програмна реалізація виконана на мові Python із використанням сучасних бібліотек та інструментів, таких як Flask, Streamlit, scikit-learn, Prefect, а також бази даних MySQL. Усі компоненти пов’язані між собою через REST API, що

забезпечує гнучкість масштабування та можливість подальшої модульної модернізації. Було реалізовано також просту схему навчання з підкріпленням, яка дозволяє враховувати оцінку користувача щодо наданих рекомендацій та коригувати майбутню логіку формування порад на основі отриманого фідбеку.

Окрему увагу в роботі приділено аналізу економічної життєздатності системи. Побудовано модель підписки, розраховано ключові показники - вартість залучення користувача, життєвий цикл клієнта (LTV), поріг беззбитковості та потенційні сценарії масштабування. Графіки окупності, отримані шляхом моделювання, продемонстрували, що при збереженні розумної ціни підписки система здатна виходити на позитивний грошовий потік при досягненні певного обсягу активної бази користувачів. У роботі також сформовано підхід до сегментації користувачів на основі кластеризації поведінки, що відкриває можливості для розробки цінової політики з урахуванням профільної специфіки.

Таким чином, реалізоване рішення відповідає сучасним вимогам до систем штучного інтелекту прикладного характеру. Воно є не лише технічно реалізованим MVP, але й демонструє науково обґрунтовану архітектуру, здатну масштабуватися та адаптуватися до реального користувача. Отримані результати можуть бути використані в подальших наукових дослідженнях у сфері поведінкового моделювання, рекомендованих систем, когнітивної підтримки прийняття рішень, а також лягти в основу комерційного SaaS-продукту в сфері цифрового коучингу та продуктивності.

Подальші напрямки дослідження можуть включати розширення системи на мобільні платформи, інтеграцію біометричних та контекстуальних даних для покращення персоналізації, удосконалення RL-моделі за допомогою багатокрокових сценаріїв прийняття рішень, а також впровадження онтологічної бази знань для семантичного узгодження цілей. Окрему увагу варто приділити питанням етичності, прозорості та безпеки рекомендацій, а також реалізації адаптивних моделей з урахуванням соціальних та емоційних чинників користувача.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Locke E. A., Latham G. P. Building a practically useful theory of goal setting and task motivation: A 35-year odyssey. *American Psychologist*. 2002. Т. 57, № 9. С. 705–717. URL: <https://doi.org/10.1037/0003-066x.57.9.705> (дата звернення: 18.05.2025).
2. Baumeister R. F., Vohs K. D., Tice D. M. The Strength Model of Self-Control. *Current Directions in Psychological Science*. 2007. Т. 16, № 6. С. 351–355. URL: <https://doi.org/10.1111/j.1467-8721.2007.00534.x> (дата звернення: 18.05.2025).
3. Ricci F., Rokach L., Shapira B. Introduction to Recommender Systems Handbook. *SpringerLink*. URL: https://link.springer.com/chapter/10.1007/978-0-387-85820-3_1 (дата звернення: 18.05.2025).
4. Vaswani A. et al. Attention is All You Need // *Advances in Neural Information Processing Systems (NeurIPS)*. – 2017. – URL: <https://arxiv.org/abs/1706.03762> (дата звернення: 18.05.2025).
5. Touvron H. et al. LLaMA: Open and Efficient Foundation Language Models // *Meta AI Research*, 2023. – URL: <https://arxiv.org/abs/2302.13971> (дата звернення: 18.05.2025).
6. Koren Y., Bell R., Volinsky C. Matrix Factorization Techniques for Recommender Systems // *Computer*. – 2009. – Vol. 42, No. 8. – P. 30–37. DOI: 10.1109/MC.2009.263
7. Rendle S. Factorization Machines // *ICDM 2010*. – P. 995–1000. DOI: 10.1109/ICDM.2010.127
8. He X. et al. Neural Collaborative Filtering // *WWW '17*. – 2017. – P. 173–182. DOI: 10.1145/3038912.3052569
9. OpenAI. GPT-4 Technical Report. – 2023. – URL: <https://openai.com/research/gpt-4>
10. HuggingFace. Transformers documentation. – 2024. – URL: <https://huggingface.co/docs>

11. Sutton R.S., Barto A.G. Reinforcement Learning: An Introduction. – 2nd ed. – MIT Press, 2018. – 552 p. – ISBN: 978-0262039246
12. Prefect Docs. Prefect 2.0 Documentation. – 2024. – URL: <https://docs.prefect.io/>
13. Pedregosa F. et al. Scikit-learn: Machine Learning in Python // Journal of Machine Learning Research. – 2011. – Vol. 12. – P. 2825–2830
14. Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System // KDD '16. – 2016. – P. 785–794. DOI: 10.1145/2939672.2939785
15. Breiman L. Random Forests // Machine Learning. – 2001. – Vol. 45. – P. 5–32. DOI: 10.1023/A:1010933404324
16. Kaufman L., Rousseeuw P.J. Finding Groups in Data: An Introduction to Cluster Analysis. – Wiley, 2005. – 342 p.
17. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet Classification with Deep Convolutional Neural Networks // NIPS 2012. – P. 1097–1105
18. ISO/IEC 9126. Software Engineering - Product Quality. – International Standard. – 2001
19. ISO/IEC 25010. Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE). – 2011
20. ДСТУ 8302:2015. Бібліографічне посилання. Загальні положення та правила складання. – [Чинний від 01.07.2016].
21. Doran, G.T. (1981). There's a S.M.A.R.T. way to write management's goals and objectives. Management Review, 70(11), 35–36.
22. Niven, P.R., & Lamorte, B. (2016). Objectives and Key Results: Driving Focus, Alignment, and Engagement with OKRs. Wiley.
23. Zhang, C., et al. (2023). Planning with Language Models for Goal-Oriented Dialogue. ACL Findings.
24. Co-designing an Embodied e-Coach With Older Adults: The Tangible Coach Journey / M. El Kamali та ін. International Journal of Human–Computer Interaction. 2023. С. 1–24. URL: <https://doi.org/10.1080/10447318.2023.2171332> (дата звернення: 18.05.2025).

25. Baumeister R. F., Tice D. M., Vohs K. D. The Strength Model of Self-Regulation: Conclusions From the Second Decade of Willpower Research. *Perspectives on Psychological Science*. 2018. Т. 13, № 2. С. 141–145. URL: <https://doi.org/10.1177/1745691617716946> (дата звернення: 18.05.2025).

26. Ryan R. M., Deci E. L. Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist*. 2000. Т. 55, № 1. С. 68–78. URL: <https://doi.org/10.1037/0003-066x.55.1.68> (дата звернення: 18.05.2025).

27. PlanGlow: Personalized Study Planning with an Explainable and Controllable LLM-Driven System / J. Chun та ін. 2025. URL: <https://doi.org/10.1145/3698205.3729541> (дата звернення: 18.05.2025).

ДОДАТКИ

Додаток А

Фрагменти програмної реалізації інтерфейсу користувача

```
import streamlit as st
import requests
import uuid
import matplotlib.pyplot as plt
import numpy as np

st.set_page_config(page_title="Система досягнення цілей",
layout="wide")
st.title("🌀 Система досягнення цілей")

user_id = st.session_state.get("user_id", str(uuid.uuid4()))
st.session_state["user_id"] = user_id

st.sidebar.header("📁 Меню")
section = st.sidebar.radio("Обрати розділ:", ["Головна", "Колесо
життя", "Цілі", "Рекомендації", "Аналітика"])

def send_decomposition_request(goal_text, goal_type, user_id):
    with st.spinner("Обробка..."):
        try:
            r = requests.post("http://localhost:5002/decompose",
json={"goal": goal_text, "type": goal_type, "user_id": user_id})
            return r.json().get("result", "Помилка")
        except Exception:
            return f"❌ Помилка: {r.status_code} - {r.text}"

def send_feedback(user_id, recommendation, feedback):
    try:
        requests.post("http://localhost:5002/recommend_feedback",
json={
            "user_id": user_id,
            "recommendation": recommendation,
            "feedback": feedback
        })
    except:
        pass

if section == "Головна":
    st.header("1. Декомпозиція цілі (OKR / SMART)")
    goal_text = st.text_area("Опишіть вашу ціль:")
    col1, col2 = st.columns(2)
    if col1.button("🔍 Декомпозувати як OKR"):
        result = send_decomposition_request(goal_text, "okr", user_id)
        st.success(result)
        st.button("📁 Зберегти ціль")
    if col2.button("🌀 Декомпозувати як SMART"):
```

```

        result = send_decomposition_request(goal_text, "smart",
user_id)
        st.success(result)
        st.button("📄 Зберегти ціль")

elif section == "Колесо життя":
    st.header("🌀 Колесо життя")
    st.markdown("""
        Це візуалізація збалансованості вашого життя за різними
напрямами. Чим більш симетричне та розширене колесо - тим більше
гармонії між сферами.
        Показники нижче середнього свідчать про сфери, які варто
посилити.
        """)
    labels = ['Кар\'єра', 'Здоров\'я', 'Духовність', 'Творчість',
'Oсобистий ріст', 'Відпочинок', 'Сім\'я', 'Оточення']
    values = [6, 7, 6, 7, 9, 8, 4, 5]
    angles = np.linspace(0, 2 * np.pi, len(labels),
endpoint=False).tolist()
    values += values[:1]
    angles += angles[:1]
    fig, ax = plt.subplots(figsize=(3, 3), subplot_kw=dict(polar=True))
    ax.plot(angles, values, color='b', linewidth=2)
    ax.fill(angles, values, color='b', alpha=0.25)
    ax.set_xticks(angles[:-1])
    ax.set_xticklabels(labels)
    st.pyplot(fig)

elif section == "Цілі":
    st.header("🎯 Збережені цілі")
    goals = {
        "Схуднути до 90 кг": ["🕒 Почати ранок з води (щодня)", "🏃 Біг
3 км через день", "🔥 Тренуватися калорії до 1800", "📊 Ціль досягти до:
01.08"],
        "Пробігти марафон": ["📅 План тренувань (3х/тиждень)", "🏃
Медогляд до 01.07", "🏁 Пробне 10 км 15.08", "📊 Марафон: 15.09"]
    }
    for goal, tasks in goals.items():
        with st.expander(goal):
            st.slider("Оцініть прогрес за цілю (0-100%)", 0, 100, 50,
key=goal)
            for task in tasks:
                st.markdown(f"- {task}")

    st.subheader("📅 Загальний план")
    st.info("(Тут буде інтерактивний графік Gantt - у MVP режимі
заглушка)")

elif section == "Рекомендації":
    st.header("💡 Рекомендації")

    if "recommendations" not in st.session_state:
        st.session_state["recommendations"] = []

    if st.button("🔄 Оновити рекомендації", key="refresh_recommend"):

```

```

        with st.spinner("Генерація рекомендацій..."):
            try:
                r = requests.post("http://localhost:5002/recommend",
json={"user_id": user_id})
                rec = r.json().get("recommendations", "")
                st.session_state["recommendations"] = [line for line in
rec.split("\n") if line.strip()]
            except Exception as e:
                st.session_state["recommendations"] = [f"✗ Помилка:
{str(e)}"]

        for i, rec in enumerate(st.session_state["recommendations"]):
            st.markdown(rec)
            col1, col2 = st.columns(2)
            if col1.button(f"👍 Це корисно", key=f"like_{i}"):
                send_feedback(user_id, rec, "positive")
            if col2.button(f"👎 Пропустити", key=f"dislike_{i}"):
                send_feedback(user_id, rec, "negative")

elif section == "Аналітика":
    st.header("☑ Аналітика виконання цілей")

    st.subheader("Статус цілей")
    st.bar_chart({
        "Активні": [4],
        "Завершені": [7],
        "Пропущені": [2]
    })

    st.subheader("Рівень завершення цілей")
    st.line_chart({
        "% завершення": [30, 50, 60, 75, 90]
    })

    st.subheader("Середня кількість задач на ціль")
    st.bar_chart({
        "Планування": [3],
        "Фінанси": [5],
        "Здоров'я": [4],
        "Робота": [6]
    })

    st.subheader("Час виконання задач (хвилини)")
    st.area_chart({
        "мін": [15, 10, 8, 9, 7],
        "середній": [20, 25, 30, 22, 28],
        "макс": [45, 40, 35, 38, 42]
    })

```

```

import os

import mysql.connector
from dotenv import load_dotenv

load_dotenv()

DB_CONFIG = {
    "host": os.getenv("DB_HOST"),
    "user": os.getenv("DB_USER"),
    "password": os.getenv("DB_PASSWORD"),
    "database": os.getenv("DB_NAME")
}

connection = mysql.connector.connect(**DB_CONFIG)
cursor = connection.cursor()

def save_interaction(goal, goal_type, response, user_id):
    cursor.execute("""
        insert into interactions (goal, goal_type, response, user_id)
        values (%s, %s, %s, %s)
    """, (goal, goal_type, response, user_id))
    connection.commit()

def save_recommendation_feedback(user_id, recommendation, feedback):
    cursor.execute("""
        insert into recommend_feedback (user_id, recommendation,
feedback)
        values (%s, %s, %s)
    """, (user_id, recommendation, feedback))
    connection.commit()

def get_prompt_by_user_id(user_id):
    query = """
        select concat('goal_prompt: ', goal_prompt , '
communication_prompt: ' ,communication_prompt )
        from goal_ai.users
        left join goal_ai.usertypes on users.user_type_id =
usertypes.id
        where ulid_id = %s
        limit 1

    """
    cursor.execute(query, (user_id,))
    result = cursor.fetchone()
    cursor.close()
    connection.close()
    if result and result[0]:
        return result[0]
    return "Сформулюй 3 загальні поради щодо покращення продуктивності."

```

```

#%%
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns

# Завантаження синтетичних даних
data = pd.read_csv("segmented_user_data.csv")
data =
data[['avg_goal_depth', 'goal_switch_rate', 'days_active_per_week', 'goals
_completed_last_month', 'time_on_platform_per_day', 'used_okrs_decomposit
ion', 'used_smart_decomposition', 'used_life_wheel']]
#%%
# Стандартизація
scaler = StandardScaler()
X_scaled = scaler.fit_transform(data)

# Кластеризація
kmeans = KMeans(n_clusters=4, random_state=42)
data["cluster"] = kmeans.fit_predict(X_scaled)
#%%
# Візуалізація центрів кластерів
centroids_scaled = scaler.inverse_transform(kmeans.cluster_centers_)
centroids = pd.DataFrame(centroids_scaled, columns=data.columns[:-1])
centroids["cluster"] = centroids.index
centroids
#%%
# Графік розподілу кількості користувачів по кластерах
sns.countplot(data=data, x="cluster")
plt.title("Кількість користувачів у кожному кластері")
plt.show()
#%%
# Збереження результатів
data.to_csv("clustered_users.csv", index=False)
centroids.to_csv("cluster_centroids.csv", index=False)
#%%

#%% md
### 🔍 Оцінка якості кластеризації
#%%
from sklearn.metrics import silhouette_score, davies_bouldin_score

# Оцінка метрик якості кластеризації
sil_score = silhouette_score(X_scaled, data["cluster"])
db_score = davies_bouldin_score(X_scaled, data["cluster"])

print(f"Silhouette Score: {sil_score:.3f}")
print(f"Davies-Bouldin Index: {db_score:.3f}")
#%% md
### 📊 Візуалізація кластерів через PCA
%%

```

```

from sklearn.decomposition import PCA
import seaborn as sns
import matplotlib.pyplot as plt

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

pca_df = pd.DataFrame(X_pca, columns=["PC1", "PC2"])
pca_df["cluster"] = data["cluster"]

plt.figure(figsize=(10, 6))
sns.scatterplot(data=pca_df, x="PC1", y="PC2", hue="cluster",
palette="tab10", s=50, alpha=0.8)
plt.title(f"Візуалізація кластерів у PCA-просторі\nSilhouette Score:
{sil_score:.3f}, DB Index: {db_score:.2f}")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.legend(title="Кластер")
plt.grid(True)
plt.tight_layout()
plt.show()
#%%

```

Звіт подібності

метадані

Назва організації
Kyiv National Economic University named after Vadym Hetman KNEU

Заголовок
Створення платформи для планування та досягнення цілей з використанням штучного інтелекту та персоналізованих рекомендацій

Автор
Науковий керівник / Експерт
Донєв Данило РомановичВасильєва Л.В.

підрозділ
кафедра інформаційних систем в економіці

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		5
Інтервали		0
Мікропробіли		3
Білі знаки		0
Парафрази (SmartMarks)	a	16

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз		Колір тексту
ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	50 0.50 %
2	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	28 0.28 %
3	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	25 0.25 %
4	https://ir.kneu.edu.ua/bitstreams/e9391672-b25d-4bf5-a6fa-9a0664aee9d8/download	16 0.16 %
5	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	15 0.15 %

6	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	13 0.13 %	
7	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	12 0.12 %	
8	https://ir.kneu.edu.ua/bitstreams/e9391672-b25d-4bf5-a6fa-9a0664aee9d8/download	10 0.10 %	
9	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	8 0.08 %	
10	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	8 0.08 %	
з бази даних RefBooks (0.00 %)			
ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
з домашньої бази даних (0.05 %)			
ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	Особливості та тенденції машинобудування в країнах ЄС 6/1/2022 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра міжнародного обліку і аудиту)	5 (1) 0.05 %	
з програми обміну базами даних (0.00 %)			
ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
з Інтернету (2.16 %)			
ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	https://ir.kneu.edu.ua/bitstreams/6ccdb206-8499-47f8-b15b-5d42b9b14ce2/download	179 (11) 1.78 %	
2	https://ir.kneu.edu.ua/bitstreams/e9391672-b25d-4bf5-a6fa-9a0664aee9d8/download	33 (3) 0.33 %	
3	https://kneu.edu.ua/userfiles/Department_of_Administration_and_Marketing_Personn/Titulna_storD196nk_a_KMD0A0.doc	5 (1) 0.05 %	
Список прийнятих фрагментів (немає прийнятих фрагментів)			
ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)	

АСПЕКТИ КІБЕРБЕЗПЕКИ СОЦІАЛЬНО-ЕКОНОМІЧНИХ СИСТЕМ В УМОВАХ ІНФОРМАЦІЙНОЇ ВІЙНИ

В.І. СКИЦЬКО, к.е.н., доцент,
skitsko@kneu.edu.ua
ORCID ID: 0000-0002-6290-9194

Д.Р. ДОНЄВ, магістрант
djdondan@gmail.com

Д.С. ОНІЩЕНКО, магістрант
diana29.oni2@gmail.com

Київський національний економічний університет імені Вадима Гетьмана

Україна стала жертвою систематичних кібератак зі сторони країни-агресора, що несе негативний вплив у соціальній, гуманітарній та промисловій сферах. За даними дослідження Державної служби спеціального зв'язку та захисту інформації України, ці атаки викликають непередбачувані наслідки для функціонування держустанов та безпеки громадян [1]. Незважаючи на вжиті заходи із захисту інформаційних систем, проблема залишається актуальною через «людський фактор», який включає в себе недостатню безпекову обізнаність та соціальну інженерію. Зловмисники активно використовують людську необачність, щоб отримати доступ до конфіденційної інформації та порушити стабільний режим функціонування держави. У роботі наведено огляд останніх успішних практик щодо системного поліпшення обізнаності у галузі інформаційної безпеки. Наведені практики пропонуються запровадити для держустанов та підприємств критичної інфраструктури з метою запобігання отриманню несанкціонованого доступу до конфіденційних систем державою-агресором і їх компрометації.

Захист ІТ-систем від атак і несанкціонованого доступу є надважливим питанням для компаній. В розвиток цієї галузі інвестуються значні ресурси, бо втрати можуть бути суттєвими. Згідно зі звітом *Cybersecurity Business Report*, кіберзлочинність коштувала «понад 6 трильйонів доларів у 2021 році, порівняно з 3 трильйонами доларів у 2015 році» [2]. Тому організації постійно шукають і створюють більш дієві засоби збереження безпеки своєї інформації. При цьому, значна кількість інцидентів (як навмисних, так і ненавмисних), пов'язаних з витоками інформації, відбувається через людський фактор [3]. Зазначається, що близько 77% витоків даних у компаніях відбуваються через використання людської неграмотності. Метою роботи є дослідити кращі практики для підвищення обізнаності персоналу і вироблення навичок протидії

комп'ютерним атакам країни-агресора для українських установ, організацій, компаній тощо.

Опишемо знайдені нами підходи, які вже довели свою ефективність.

Гейміфікація є новим явищем у сфері розвитку комп'ютерної грамотності. Співробітникам пропонується взаємодіяти зі спеціально розробленою грою, щоб стати більш обізнаними про загрози та вразливості інформаційної безпеки. Гейміфікація є підходящим рішенням для підвищення рівня знань працівників, оскільки вона персоналізує навчальний вміст відповідно до потреб компанії чи підприємства. Інтерактивний прототип гри, представлений під час оцінки впливу вказаного підходу, показав, що працівники були більш вмотивовані та готові дотримуватися використовуваних методів безпеки, порівняно з групою, яка прослухала серію лекцій з тими самими темами [4].

Метод виявлення порушень використовується для попередження виконання співробітником небажаних дій, разом з тим пояснюючи помилку і потенційні ризики. Цей метод допомагає доносити матеріал на основі порушень працівниками політик компанії (відкриття імейлу з невідомого домену, встановлення на комп'ютер неліцензованих додатків). Він працює за допомогою методу контролю фізичного доступу на основі радіочастотної ідентифікації (RFID). Технологія RFID виявить і зареєструє будь-який несанкціонований доступ до інформації з боку працівника. Таким чином, повідомлення про це конкретне порушення буде надіслано саме цьому працівникові. За оцінками оригінального дослідження [5], кількість випадків несанкціонованого доступу в організації різко зменшилася, коли цей метод був впроваджений. І в той же час, рівень обізнаності серед співробітників покращився завдяки використанню цього підходу.

Покращення загальних ІТ-знань працівників є однією з найвпливовіших передумов інформаційної безпеки. ІТ-знання працівників стосуються загальних знань основних ІТ-додатків, що використовуються у повсякденній роботі, таких як комп'ютери, системи електронної пошти та Інтернет. Рівень загальних ІТ-знань позитивно впливає на рівень інформаційної безпеки працівників [6]. Чим більше працівники знають про інформаційні технології та принципи функціонування додатків і машин, якими користуються, тим більше вони обізнані з питаннями інформаційної безпеки. Таким чином, організаціям рекомендується підвищувати рівень ІТ-знань працівників, щоб запобігти ненавмисним порушенням. Цю думку також підтверджують емпіричним шляхом інші публікації з цієї теми, які в поточній роботі не наводяться.

Проблема кібербезпеки в Україні є актуальною та серйозною, зокрема через систематичні кібератаки та людський фактор, який допускає вразливості в інформаційній безпеці. Дослідження виявило підходи призначені для підвищення кіберграмотності та свідомості персоналу, які можуть стати ефективними засобами протидії комп'ютерним атакам.

Перспективи подальшого розвитку включають імплементацію практик, які вже успішно використовуються ІТ-компаніями, зокрема в сфері гейміфікації

та методів виявлення порушень. В даній роботі ми системно ознайомилися та дослідили підходи для підвищення обізнаності та навичок протидії кібератакам, що може слугувати основою для розробки програм та навчальних ініціатив у держустановах та підприємствах критичної інфраструктури. Дослідження також підкреслило важливість інвестування у кібербезпеку як приватними, так і державними підприємствами, оскільки витрати на боротьбу з кіберзагрозами значно перевищують можливі втрати внаслідок кіберзлочинності. У майбутньому, з урахуванням постійного розвитку технологій та методів атак, необхідно посилювати зусилля у вдосконаленні систем кіберзахисту та поширюванні знань щодо кібербезпеки серед різних шарів суспільства.

Література:

1. State Service of Special Communications and Information Protection of Ukraine. Russia's cyber tactics: lessons learned 2022. URL: <https://cip.gov.ua/services/cm/api/attachment/download?id=53466>.
2. Morgan S. Cybercrime damages expected to cost the world \$6 trillion by 2021. *csoonline.com*. 2016. URL: <https://www.csoonline.com/article/557557/cybercrime-damages-expected-to-cost-the-world-6-trillion-by-2021.html>.
3. Stahl B., Doherty N., Shaw M. Information security policies in the UK healthcare sector: a critical evaluation. *Information Systems Journal*. 2011. URL: <https://doi.org/10.1111/j.1365-2575.2011.00378.x>.
4. Gamification of Information Security Awareness and Training. *SciTePress*. URL: <https://www.scitepress.org/Link.aspx?doi=10.5220/0006128500590070>.
5. Kyong-Ho Choi, Dong Hwi Lee. A study on strengthening security awareness programs based on an RFID access control system for inside information leakage prevention. *Multimedia Tools and Applications*. 2013. URL: <https://link.springer.com/article/10.1007/s11042-013-1727-y>.
6. Haeussinger F., Kranz J. Information Security Awareness: Its Antecedents and Mediating Effects on Security Compliant Behavior. *International Conference on Information Systems 2013*: Статті в реферованих виданнях (міжнародні), м. Мілан, 15–18 груд. 2013р.