

ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА»
Факультет автоматизації виробництва та цифрових технологій
Кафедра цифрових технологій та проектно-аналітичних рішень

«Допущено до захисту»

Гарант ОПП

Павло САГАЙДА

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістра

за підсумками виконання
освітньо-професійної програми
«Комп'ютерні науки та цифровий інтелект»
за спеціальністю 122 Комп'ютерні науки

**на тему «Дослідження моделей, методів та інформаційних
технологій використання AI-агентів для організації
інтелектуального освітнього середовища»**

Керівник роботи

Павло САГАЙДА

Консультант від
бази практики

Олександр СТАСЮК

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач

Ярослав ПЛУТАЛОВ

Підсумкова оцінка за атестацію			
--------------------------------	--	--	--

Голова ЕК

Антон КУДРЯВЦЕВ

ЗАПОРІЖЖЯ 2026

ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА»

Факультет цифрових технологій та автоматизації виробництва

Кафедра цифрових технологій та проектно-аналітичних рішень

Ступень вищої освіти магістр

Спеціальність 122 Комп'ютерні науки

(шифр і назва)

Освітня програма Комп'ютерні науки та цифровий інтелект

(назва)

ЗАТВЕРДЖУЮ

Гарант ОПП

Павло САГАЙДА

«10» 11 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

Плуталову Ярославу Андрійовичу

(прізвище, ім'я, по батькові здобувача)

1. Тема роботи Дослідження моделей, методів та інформаційних технологій використання AI-агентів для організації інтелектуального освітнього середовища

керівник роботи Сагайда Павло Іванович, доцент, докт. техн. наук,

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом Університету від 10.09.2025 р. №239

2. Строк подання роботи 20.01.2026 р.

3. Вихідні дані до роботи Навчальна література, державні стандарти, методична література з спеціальних дисциплін та дипломування, науково-дослідницькі роботи з тематики автоматизації обробки й аналізу даних та методів цифрового інтелекту, літературні джерела, технологічні інструкції тощо

4. Зміст пояснювальної записки (перелік питань) виконати аналіз моделей, методів та алгоритмів використання AI-агентів для організації освітнього середовища, аналіз функціональності існуючих програмних систем і використовуваних інформаційних технологій; виконати дослідження методів, моделей та інформаційних технологій обробки даних для інформаційної підтримки даного технологічного процесу; представити результати дослідження та впровадження роботи в практику; зробити висновки із ефективності використання AI-агентів для організації освітнього середовища, з кількісними показниками досягнутих результатів; визначити позитивний ефект від розробленого ПМК та підвищення ефективності діяльності в предметній області.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) мета і завдання дослідження; математичні моделі, методика дослідження; діаграми проекту програмної системи в нотації UML (діаграми прецедентів, класів, послідовностей, діяльності, потоків даних); результати експериментальних досліджень та обґрунтування економічної ефективності.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
<i>Фаховий консультант</i>	П.І. Сагайда, доц., докт. техн. наук		
<i>Нормоконтроль</i>	О.С. Касьянюк, ст. викл.		
<i>Економічна частина</i>	І.А. Гетьман, доц., канд. техн. наук		

7. Дата видачі завдання 10.11.2025

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Термін виконання етапів роботи
1.	Аналіз предметної області, огляд існуючих мультиагентних систем та методів використання в освіті.	10.01.2026 – 15.01.2026
2.	Розробка математичної моделі взаємодії агентів та алгоритмів забезпечення стабільності діалогу.	16.01.2026 – 18.01.2026
3.	Проектування архітектури та програмна реалізація мультиагентної системи для організації освітнього середовища.	19.01.2026 – 21.01.2026
4.	Проведення експериментальних досліджень ефективності рольової моделі та стабільності генерації відповідей.	22.01.2026 – 23.01.2026
5.	Виконання економічних розрахунків доцільності розробки та впровадження системи.	24.01.2026
6.	Оформлення текстової, програмної та графічної документації.	25.01.2026
7.	Оформлення записки. Підготовка доповіді та презентації. Отримання відгуку та рецензії.	26.01.2026
8.	Захист проекту	27.01.2026

Здобувач вищої освіти

_____ Я.А. Плуталов
(підпис) (ініціали та прізвище)

Керівник роботи, доц.,
докт. техн. наук

_____ П.І. Сагайда
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Плуталов Я.А. Дослідження моделей, методів та інформаційних технологій використання AI-агентів для організації інтелектуального освітнього середовища.

Кваліфікаційна робота на здобуття освітнього ступеня магістра за спеціальністю 122 «Комп'ютерні науки». – ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА» МОН України, м. Запоріжжя, 2026.

Метою роботи є підвищення ефективності дистанційного навчання шляхом створення мультиагентної системи, що забезпечує адаптивну подачу матеріалу та моделює групову взаємодію.

Об'єктом дослідження є процес організації інтелектуальних освітніх середовищ з використанням технологій штучного інтелекту.

Предметом дослідження є методи, математичні моделі та алгоритми взаємодії AI-агентів у мультиагентних системах на базі великих мовних моделей.

Методологія – системний аналіз, теорія графів та спектральний аналіз, ймовірнісне моделювання, методи навчання з підкріпленням, об'єктно-орієнтоване проектування, імітаційне моделювання.

У цій роботі проаналізовано сучасні підходи до побудови інтелектуальних освітніх середовищ та обмеження традиційних LLM-систем. Розроблено математичну модель взаємодії агентів, в якій введено структурне обмеження на спектральний радіус матриці суміжності графа для забезпечення стабільності комунікації. Спроектовано та реалізовано програмний комплекс гібридного навчання на мові Python, що використовує технологію RAG та OpenAI API. Реалізовано рольову модель агентів («Вчитель», «Розумник», «Жартівник») для відтворення соціальної динаміки навчальної групи.

Розроблений програмний комплекс дозволяє автоматизувати рутинні завдання викладачів, підвищити залученість студентів та забезпечити цілодобову підтримку навчального процесу. Завдяки модульній архітектурі систему можна адаптувати для викладання інших технічних або гуманітарних дисциплін, а також інтегрувати у наявні платформи дистанційного навчання.

КЛЮЧОВІ СЛОВА: МУЛЬТИАГЕНТНІ СИСТЕМИ, ШТУЧНИЙ ІНТЕЛЕКТ, ВЕЛИКІ МОВНІ МОДЕЛІ, RAG, LLM, АДАПТИВНЕ НАВЧАННЯ, OPENAI API, PYTHON, ЕКОНОМІЧНА ЕФЕКТИВНІСТЬ.

ABSTRACT

Plutalov Y.A. Research on models, methods, and information technologies for using AI agents to organize an intelligent educational environment.

Qualification work for obtaining the degree of Master of Computer Science in the specialty 122 Computer Science. – LLC "TECHNICAL UNIVERSITY "METINVEST POLYTECHNIC" MES of Ukraine, Zaporizhzhia, 2026.

The purpose of the work is to increase the efficiency of distance learning by creating a multi-agent system that provides adaptive presentation of material and simulates group interaction.

The object of the study is the process of organizing intelligent educational environments using artificial intelligence technologies.

The subject of the study is the methods, mathematical models, and algorithms of AI agent interaction in multi-agent systems based on large language models.

Methodology – system analysis, graph theory and spectral analysis, probabilistic modeling, reinforcement learning methods, object-oriented design, simulation modeling.

This work analyzes modern approaches to building intelligent educational environments and the limitations of traditional LLM systems. A mathematical model of agent interaction has been developed, introducing a structural constraint on the spectral radius of the graph adjacency matrix to ensure communication stability. A hybrid learning software system was designed and implemented in Python using RAG technology and OpenAI API. A role-based agent model ("Teacher", "Nerd", "Joker") was implemented to reproduce the social dynamics of a learning group.

The developed software system enables the automation of routine teaching tasks, increases student engagement through interactive role-based interaction, and provides continuous availability of support for the educational process. Due to its modular architecture, the system can be adapted for teaching other technical or humanities disciplines, as well as integrated into existing distance learning platforms.

KEYWORDS: MULTI-AGENT SYSTEMS, ARTIFICIAL INTELLIGENCE, LARGE LANGUAGE MODELS, RAG, LLM, ADAPTIVE LEARNING, OPENAI API, PYTHON, ECONOMIC EFFICIENCY.

ЗМІСТ

1 АНАЛІЗ СУЧАСНИХ ПІДХОДІВ ДО ПОБУДОВИ ІНТЕЛЕКТУАЛЬНИХ ОСВІТНІХ СЕРЕДОВИЩ НА БАЗІ МУЛЬТИАГЕНТНИХ СИСТЕМ	12
1.1 Аналіз розвитку великих мовних моделей останніх років	12
1.2 Контекст розвитку штучного інтелекту в освіті	13
1.3 Обмеження традиційних LLM-систем в освітньому контексті	14
1.4 Мультиагентні системи як рішення	16
1.4.1 Колаборативне розв'язування складних задач	18
1.4.2 Retrieval-Augmented Generation як інфраструктура знаннєвої пам'яті мультиагентних систем	19
1.4.3 Ітеративна рефлексія та самовдосконалення	23
1.5 Стан дослідження мультиагентних освітніх систем	25
2 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ ВЗАЄМОДІЇ AI-АГЕНТІВ ТА АЛГОРИТМІВ ОБРОБКИ НАВЧАЛЬНОГО КОНТЕНТУ	31
2.1 Обґрунтування вибору методів теоретичних та експериментальних досліджень, програмного забезпечення	31
2.1.1 Методи теоретичних досліджень	31
2.1.2 Методи експериментальних досліджень	32
2.1.3 Обґрунтування вибору програмного забезпечення	33
2.2 Математична модель об'єкта дослідження	33
2.2.1 Динаміка мультиагентної мережі та умови стабільності	34
2.2.2 Модульна декомпозиція та розподіл задач	35
2.2.3 Адаптивність та персоналізація через Гауссівські процеси	38
2.2.4 Ефективність комунікації та метрики оптимізації	40
2.3 Розробка алгоритмічного забезпечення та протоколів взаємодії агентів	42
2.3.1 Архітектура агентів та рольова модель	43
2.3.2 Алгоритм взаємодії агентів	44

3 РОЗРОБКА ЗАСОБІВ МОДЕЛЮВАННЯ ПРОГРАМНОГО КОМПЛЕКСУ ГІБРИДНОГО НАВЧАННЯ	46
4 ПРОВЕДЕННЯ ТА РЕЗУЛЬТАТИ ТЕОРЕТИЧНИХ ТА ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ ЗАДАНОГО ОБ'ЄКТА	60
4.1 Методика та умови проведення експерименту	60
4.2 Експериментальна перевірка рольової диференціації агентів	61
4.3 Дослідження впливу параметрів стохастичності на стабільність генерації	62
4.4 Дослідження впливу промпт-інжинірингу	64
4.5 Порівняльний аналіз мовних моделей	68
5 ЕКОНОМІЧНІ РОЗРАХУНКИ ДОЦІЛЬНОСТІ ПРОЄКТУВАННЯ ПРОГРАМНОГО КОМПЛЕКСУ ГІБРИДНОГО НАВЧАННЯ	71
5.1 Теоретико-методологічні основи економічного аналізу інтелектуальних освітніх систем	71
5.1.1 Підходи до оцінювання економічної ефективності ІТ-проєктів в освітній сфері	71
5.1.2 Концепція соціального повернення на інвестиції (SROI) в студентських розробках.....	72
5.1.3 Особливості формування витрат в екосистемі генеративного штучного інтелекту.....	73
5.2 Оцінка капітальних витрат на створення інтелектуального програмного продукту	74
5.2.1 Апаратне забезпечення: Амортизація та експлуатація	75
5.2.2 Програмне забезпечення та інфраструктура розробки	76
5.2.3 Оцінка трудових витрат на розробку програмного продукту	79
5.2.4 Структурна декомпозиція робіт (WBS) та оцінка трудомісткості.....	81
5.2.5 Енергетичні витрати під час розробки	82
5.2.6 Узагальнений кошторис інвестиційних витрат	83
5.3 Оцінка експлуатаційних витрат та сукупної вартості володіння AI-системою (TCO).....	83

5.3.1	Моделювання профілю навантаження	84
5.3.2	Інфраструктурні витрати	85
5.3.3	Сумарні річні експлуатаційні витрати (ТСО)	86
5.4	Розрахунок економічного ефекту: Модель економії часу та підвищення продуктивності	87
5.4.1	Економія робочого часу викладача внаслідок автоматизації навчальних процесів	87
5.4.2	Підвищення ефективності навчання та економія часу студентів	90
5.4.3	Сумарний економічний ефект	91
5.5	Інтегральна оцінка економічної ефективності проєкту	91
5.5.1	Термін окупності	92
5.5.2	Соціальний ефект та стратегічні переваги	93
5.6	Узагальнена оцінка економічної ефективності проєкту	93
	ВИСНОВКИ	95
	ПЕРЕЛІК ПОСИЛАНЬ	97
	ДОДАТОК А	104
	ДОДАТОК Б	105
	ДОДАТОК В	111

ВСТУП

Актуальність теми роботи. Інтеграція технологій штучного інтелекту в освітній процес перейшла від етапу експериментів до системного впровадження. Проте існуючі рішення на базі великих мовних моделей (LLM) переважно реалізують модель взаємодії «один-на-один» (студент-чатбот), яка не здатна відтворити соціальну динаміку реального навчального класу. Як свідчить аналіз предметної області, відсутність рольової диференціації та педагогічної стратегії в таких системах призводить до зниження залученості студентів та виникнення інформаційного шуму («галюцинацій»). Вирішення цієї проблеми полягає у переході до мультиагентних систем, де віртуальні агенти виконують спеціалізовані ролі. Однак, координація таких агентів створює нові виклики, пов'язані зі стабільністю комунікації та узгодженістю діалогу. Тому розробка математичних моделей та програмних засобів, які поєднують генеративні можливості LLM із умовами стабільності мережевої взаємодії, є актуальним науково-прикладним завданням.

Мета дослідження – підвищення ефективності дистанційного навчання шляхом створення мультиагентної системи, що забезпечує адаптивну подачу матеріалу та моделює групову взаємодію.

Об'єкт дослідження – процес організації інтелектуальних освітніх середовищ з використанням технологій штучного інтелекту.

Предмет дослідження – методи, математичні моделі та алгоритми взаємодії AI-агентів у мультиагентних системах на базі великих мовних моделей.

Задачі дослідження:

1. Проаналізувати сучасні підходи до побудови інтелектуальних освітніх середовищ та виявити обмеження традиційних LLM-систем.
2. Розробити математичну модель взаємодії AI-агентів, що забезпечує стабільність комунікації та мінімізацію інформаційного шуму.
3. Розробити алгоритмічне забезпечення та протоколи взаємодії агентів, включаючи методи RAG (Retrieval-Augmented Generation) для роботи з навчальним контентом.
4. Спроекувати архітектуру та реалізувати програмний комплекс гібридного навчання з використанням рольової моделі агентів («Вчитель», «Розумник», «Жартівник»).
5. Провести експериментальні дослідження для верифікації рольової диференціації агентів та стабільності системи.
6. Виконати техніко-економічне обґрунтування розробки та розрахувати економічний ефект від впровадження системи.

Наукова новизна одержаних результатів:

- Запропоновано концепцію гібридної системи, яка поєднує принципи стабільності мережевої взаємодії з генеративними можливостями LLM.
- Удосконалено математичну модель поведінки агентів шляхом введення структурного обмеження на спектральний радіус матриці суміжності графа взаємодій, що гарантує відсутність хаотичної комунікації.
- Набуло подальшого розвитку застосування системних промптів для інтеграції складних рольових моделей у програмний код без необхідності явного обчислення функцій невизначеності.

Практичне значення одержаних результатів. Розроблено та реалізовано програмний комплекс, що інтегрує технології OpenAI API, RAG, а також модулі синтезу та розпізнавання мовлення. Система дозволяє автоматизувати створення навчального контенту та забезпечує цілодобову підтримку студентів, що підтверджено розрахунками економічної ефективності.

Структура та обсяг роботи. Робота складається зі вступу, 5 розділів, висновків та переліку посилань:

- проведено аналіз сучасних підходів до побудови інтелектуальних освітніх середовищ та обґрунтовано доцільність використання мультиагентних систем.
- розроблено математичну модель взаємодії агентів, визначено умови стабільності динамічної мережі та алгоритми обробки контенту.
- спроектовано архітектуру програмного комплексу, розроблено діаграми прецедентів, класів та послідовностей, описано реалізацію сервісів чату.
- наведено результати експериментальних досліджень семантичної диференціації агентів та впливу параметрів стохастичності на стабільність генерації.
- виконано економічні розрахунки, визначено капітальні та експлуатаційні витрати, розраховано термін окупності та соціальний ефект проекту.

1 АНАЛІЗ СУЧАСНИХ ПІДХОДІВ ДО ПОБУДОВИ ІНТЕЛЕКТУАЛЬНИХ ОСВІТНІХ СЕРЕДОВИЩ НА БАЗІ МУЛЬТИАГЕНТНИХ СИСТЕМ

1.1 Аналіз розвитку великих мовних моделей останніх років

Останнє десятиліття стало періодом стрімкого розвитку штучного інтелекту (ШІ). Зокрема, значного прогресу досягнуто завдяки глибокому навчанню на основі нейронних мереж, що дало змогу суттєво покращити технології розпізнавання зображень і мовлення. Одним із ключових етапів цього процесу стало змагання ImageNet Large Scale Visual Recognition Challenge, яке продемонструвало високий рівень можливостей ШІ у сфері комп'ютерного зору.

Ще одним важливим проривом стала успішна реалізація навчання з підкріпленням, що проявилось у розробках DeepMind, зокрема в AlphaGo та AlphaZero. Ці системи досягли виняткових результатів у складних стратегічних іграх, таких як го та шахи, використовуючи алгоритми самонавчання. Цей підхід став значним кроком вперед у вдосконаленні технологій навчання з підкріпленням. Одночасно з цим відбулося стрімке зростання якості генеративних моделей, що дозволило створювати високореалістичний синтетичний мультимедійний контент.

Вражаючий розвиток також спостерігався у сфері Natural Language Processing (NLP). Завдяки появі моделей нового покоління, таких як BERT (Google), GPT (OpenAI) та T5 (Google), істотно покращилися

можливості машинного перекладу, аналізу тексту та генерації контенту, що суттєво розширило сфери застосування NLP-технологій.

Великі мовні моделі базуються на архітектурі трансформерів і навчаються на величезних обсягах текстових даних. Їхня продуктивність значно зростала з кожним новим поколінням. Моделі демонструють здатність створювати тексти, які за стилем і логікою нагадують людські. Використовуючи методи глибокого навчання, вони аналізують контекст і прогнозують наступні слова у реченні, що дозволяє їм будувати зв'язні та осмислені тексти.

Попри певні обмеження, зокрема ризик генерування некоректних або нелогічних відповідей, Large Language Models (LLM) стали надзвичайно популярними завдяки широкому спектру застосувань – від написання текстів і програмування до аналізу та пояснення складних понять. Значним поштовхом до популяризації технології стала поява ChatGPT від OpenAI, що за короткий час залучив мільйони користувачів.

1.2 Контекст розвитку штучного інтелекту в освіті

Оскільки розвиток великих мовних моделей та архітектур мультиагентних систем започаткував нові парадигми для освіти, які моделюють групову взаємодію і соціальні ролі в навчанні, застосування штучного інтелекту в освіті набирає все більшої актуальності у контексті:

- Персоналізації навчання – необхідність адаптації навчального процесу до індивідуальних характеристик кожного студента [1]

- Масштабування якісного навчання – проблема надання високоякісної персоналізованої освіти на великих масштабах [2]
- Автоматизації педагогічних завдань – зменшення навантаження вчителів на рутинні завдання та фокус на критичні аспекти педагогічної взаємодії
- Емульованого соціального навчання - створення віртуального середовища, що відтворює реальні групові взаємодії та колаборативне навчання.

1.3 Обмеження традиційних LLM-систем в освітньому контексті

Дослідження, проведені у 2024-2025 роках, виявили кілька серйозних обмежень LLM-систем для освітніх застосувань.

Відсутність педагогічної експертизи в архітектурі системи. Комерційні LLM (як-от ChatGPT і Gemini) уміють швидко створювати зв'язний навчальний текст, але часто не забезпечують достатньої педагогічної якості для по-справжньому результативних завдань і занять. Є дві головні причини: по-перше, у цих моделей немає внутрішньо закладеного розуміння того, як люди навчаються і як це знання застосовувати під час розроблення матеріалів; по-друге, більшості вчителів бракує часу або спеціальної підготовки, щоб послідовно й точно передавати всі педагогічні нюанси у запитах до моделей. Через це ключові педагогічні рішення фактично перекладаються на користувача й формуються під час формулювання запиту, а не задаються самою системою, що ускладнює стабільне

отримання змістовних, доречних і готових до використання на уроці матеріалів без додаткового доопрацювання.

Можливий шлях розв'язання проблеми – інтегрувати педагогічні підходи безпосередньо в архітектуру таких систем, наприклад, завдяки багатоагентним рішенням, де окремі агенти узгоджують цілі навчання, способи засвоєння та методи подання матеріалу. Така інтеграція підвищує відповідність матеріалів освітнім цілям і робить результати більш змістовними, доречними до контексту та практично придатними для проведення уроків [3].

Обмежені можливості планування та декомпозиції задач. LLM-агенти помітно вразливі під час розв'язування складних багатокрокових завдань. Щоб уникати помилок і нестабільності, їм зазвичай потрібні додаткові механізми: розбиття цілей на керовані підзадачі; генерування кількох альтернативних планів із подальшим відбором; підключення зовнішніх планувальників або інструментів перевірки; цикли саморефлексії та виправлення; а також пам'ять, яка накопичує й повертає релевантний досвід. Без цих компонентів єдиний агент на базі однієї LLM легко отримує розв'язок, який здається найкращим серед близьких альтернатив, але не є найбільш оптимальним у всьому просторі можливих рішень, втрачає контекст або хибно тлумачить стан середовища – особливо у сценаріях з неявними вимогами та довгими ланцюжками залежностей (наприклад, в освіті). Практика свідчить, що більш надійні системи комбінують кілька таких підсилювачів, поєднуючи ширший пошук, перевірюваність кроків і накопичення досвіду [4].

Відсутність спонтанних колективних взаємодій. Системи з одним агентом лише частково відтворюють групову динаміку реального класу, де ключову роль відіграють жваві дискусії, взаємонавчання між

студентами, підтримка та розмаїття поглядів. Для правдоподібного моделювання таких процесів потрібна скоординована взаємодія кількох агентів із розмежованими ролями (наприклад, викладача, асистента та “однокурсників”), які ініціюють запитання, поглиблюють обговорення, узагальнюють зміст і підтримують соціальну присутність. Емпіричні результати показують, що мультиагентні конфігурації активізують студентську участь і краще відтворюють студент-студент взаємодії, ніж одноагентні підходи, хоча висновки слід інтерпретувати в межах досліджуваних курсів і налаштувань системи [5,6].

1.4 Мультиагентні системи як рішення

Мультиагентні системи (MAS) відкривають новий конструктивний підхід до архітектури освітніх платформ: замість єдиного універсального модуля, екосистема складається з колекції спеціалізованих агентів - вчителя, оцінювача, агент керування навчальним контентом, агент-студент, модератора групи та агента-аналітика прогресу – які взаємодіють у спільному навчальному середовищі. Кожен агент оптимізує власну локальну мету, узгоджену з глобальною освітньою ціллю курсу чи групи. Така організація дозволяє розкласти складні педагогічні завдання на узгоджені ролі: один агент підбирає завдання відповідно до рівня і стилю навчання, інший надає покрокові підказки, третій оцінює внесок у груповій роботі, четвертий координує взаємодію й синхронізацію у командних вправах.

На відміну від суцільної LLM, мультиагентні архітектури забезпечують:

- модульність і масштабованість – компоненти можна додавати або замінювати без перезбирання всієї системи, а навантаження розподіляється між агентами;
- координацію кількох стратегій прийняття рішень замість однієї, що підвищує стійкість до змін у поведінці користувачів і групових динамік, та дає змогу проводити цілеспрямовану групову діяльність (проекти, дебати, симуляції);
- персоналізацію через моделювання профілю й намірів студента, включно з адаптацією рівня складності, формату матеріалів і темпу;
- більш прозорий підрахунок внеску в командних завданнях через локальні показники якості та їх узагальнення у спільний результат.

Разом з тим, успішна реалізація вимагає врахування специфічних викликів багатоагентного навчання: уникнення розбалансованої поведінки між ролями, узгодження локальних і глобальних цілей, стабілізація навчання за умов змінної взаємодії студентів, а також обґрунтований розподіл “кредиту” за результат у групових активностях. Практично це досягається вибором навчальної схеми з централізованим тренуванням і децентралізованим виконанням для узгодженості поведінки агентів, впровадженням механізмів координації (правила обміну повідомленнями, протоколи ролей) і чіткими метриками освітнього процесу, що поєднують індивідуальний прогрес і якість командної взаємодії [7].

1.4.1 Колаборативне розв'язування складних задач

Колаборативні багатоагентні системи можуть підвищувати якість розробки навчальних активностей завдяки поєднанню розподілу завдань, паралельної генерації варіантів і подальшого узгодження результатів у спільному обговоренні. Коли агенти не лише продукують контент, а й взаємно критикують, уточнюють і синтезують пропозиції, це сприяє появі рішень, які краще поєднують творчість, предметну точність і практичну придатність до навчальної роботи [7]. Вбудування педагогічних принципів і структур (наприклад, явне виділення знань, процесів навчання та інструкційних стратегій) у внутрішню логіку агентів допомагає тримати фокус на узгодженості цілей, активностей та оцінювання, тоді як рольова різноманітність або “персоніфікація” агентів розширює спектр ідей і підходів. За таких умов колаборативні MAS здатні продукувати матеріали, які вчителі оцінюють як більш контекстуально релевантні, креативні та готові до впровадження, хоча формальні рубрики можуть фіксувати лише помірні кількісні відмінності. Водночас слід враховувати операційні компроміси: режими з багатьма етапами взаємодії між агентами зазвичай потребують більше обчислювальних ресурсів і часу, тож у практичних застосуваннях доречно балансувати між глибиною колаборації та ефективністю [3].

1.4.2 Retrieval-Augmented Generation як інфраструктура знаннєвої пам'яті мультиагентних систем

У відповідь на ці обмеження останніми роками активно з'являються підходи, що дають LLM змогу звертатися до зовнішніх джерел знань у режимі реального часу. Одним із найперспективніших серед них є Retrieval-Augmented Generation (RAG) – технологія, яка дозволяє моделі працювати з великими масивами навчальних матеріалів, документів та баз знань, а не покладатися лише на власні параметри [8]. У мультиагентних системах RAG відіграє особливо важливу роль, оскільки забезпечує спільне інформаційне середовище, доступне всім агентам, учителю, студенту чи асистенту. Це підтримує узгодженість інформації та актуальність знань упродовж усієї взаємодії.

Великі мовні моделі (LLMs) досягли значних успіхів, проте залишаються обмеженими у виконанні завдань, що вимагають глибоких знань або спеціалізованої інформації. Зокрема, вони можуть створювати так звані «галюцинації», коли стикаються із запитом, що виходить за межі їхніх навчальних даних або потребують актуальної інформації.

Для вирішення цих проблем використовується RAG, яка поєднує генеративні можливості LLMs із пошуком відповідних фрагментів тексту у зовнішніх базах знань. Завдяки семантичному порівнянню система знаходить найбільш релевантні дані та використовує їх у відповіді, що значно знижує ризик створення недостовірного контенту.

RAG використовує механізм пошуку релевантних фрагментів тексту з бази знань, що зберігається у векторному форматі. Це досягається шляхом перетворення як запитів користувача, так і

інформаційної бази у векторний простір високої розмірності, що дозволяє знаходити змістовно схожі фрагменти та використовувати їх у процесі генерації відповідей.

Методологія RAG передбачає поділ тексту на фрагменти, які конвертуються у векторні подання та зберігаються у спеціалізованій базі – векторному сховищі. Це сховище забезпечує ефективний пошук за допомогою різних метрик подібності, таких як косинусна, евклідова або мангеттенська відстань [10]. Загальну схему цього процесу наведено на Рисунку 1.1.

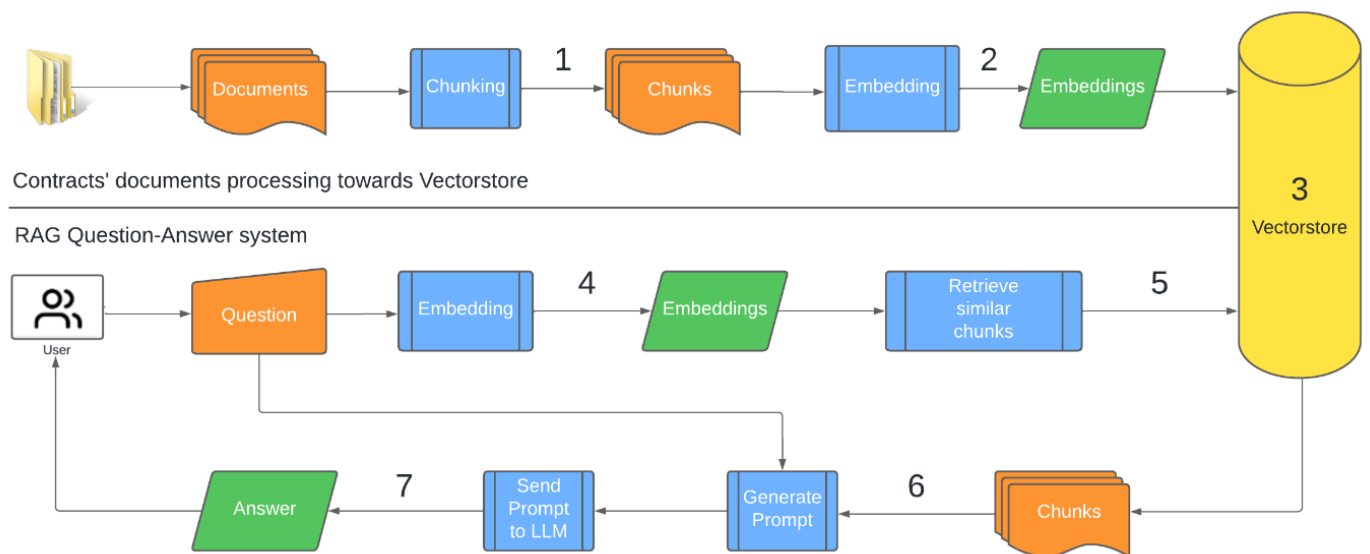


Рисунок 1.1 – Схема роботи RAG. Джерело:[10]

Процес обробки запиту відбувається в кілька етапів:

1. Завантаження текстової інформації з документів та її поділ на фрагменти.
2. Перетворення цих фрагментів у векторні подання.
3. Збереження векторів у базі для подальшого пошуку.

4. Конвертація запиту у векторний формат.
5. Пошук та вибір найбільш релевантних фрагментів відповідно до заданої метрики подібності.
6. Формування підказки (prompt), що включає початковий запит, знайдені тексти, інструкції та, за необхідності, історію діалогу.
7. Передача підказки в LLM для генерації остаточної відповіді.

Інтеграція RAG у мовні моделі сприяла її широкому впровадженню, перетворивши цю технологію на ключовий інструмент для вдосконалення чат-ботів і підвищення ефективності LLM у практичному застосуванні.

Розвиток RAG в епоху великих мовних моделей проходив кілька ключових етапів. Спочатку ця технологія з'явилася разом із поширенням архітектури Transformer, зосереджуючись на покращенні мовних моделей шляхом інтеграції зовнішніх знань через механізми попереднього навчання (Pre-Training Models, PTM). На цьому етапі увага приділялася вдосконаленню методів передтренування.

Наступний значний прорив відбувся з появою ChatGPT, який продемонстрував потужні можливості навчання в контексті. Це змінило вектор досліджень RAG, які почали зосереджуватися на покращенні процесу отримання інформації під час генерації відповідей, що дало змогу вирішувати складніші завдання, пов'язані зі знаннями [11].

Згодом удосконалення RAG перестало обмежуватися лише генерацією відповідей і почало активно поєднуватися з методами донавчання великих мовних моделей, що ще більше розширило її можливості.

Застосунки LLM із розширеними даними можуть набувати різних форм: від ботів для запитань і відповідей, що працюють із галузевими даними, до семантичних обробників у складних інформаційних конвеєрах або агентів, які виконують окремі етапи в мультиагентних системах. Цю залежність можна представити формулою 1.1.

$$f: Q \xrightarrow{D} A \quad (1.1)$$

де Q , A та D позначають запит користувача (Query), очікувану відповідь (Answer) та доступні зовнішні дані (Data) відповідно. Завданням системи f є встановлення відповідності між Q та A на основі D .

На відміну від автономних LLM, які ґрунтуються виключно на попередньо отриманих знаннях, моделі з розширеними даними використовують зовнішню інформацію (D) для точнішого формування відповідей. Такий підхід дає змогу LLM доступатися до актуальних, галузевих знань і враховувати експертні висновки. Запити можуть класифікуватися за рівнем складності залежно від обсягу та способу використання зовнішніх даних, що визначає глибину аналізу та необхідний рівень взаємодії з інформацією [12].

На розвиток екосистеми RAG значною мірою впливає прогрес її технічного стека. Ключові інструменти, такі як LangChain і LLamaIndex, швидко набули популярності з появою ChatGPT, надаючи широкі API, пов'язані з RAG, і стаючи важливими в області LLM. Новий стек технологій, хоча і не такий багатий на функції, як LangChain і LLamaIndex, виділяється своїми спеціалізованими продуктами. Наприклад, Flowise AI

надає перевагу підходу з низьким кодом, що дозволяє користувачам розгортати програми штучного інтелекту, включаючи RAG, за допомогою зручного інтерфейсу перетягування. Інші технології, такі як HayStack, Meltano та Cohere Coral, також привертають увагу своїм унікальним внеском у цю сферу.

1.4.3 Ітеративна рефлексія та самовдосконалення

У 2024 році була запропонована архітектура von Neumann [13] для мультиагентних систем у навчанні, яка розкладає кожного агента на чотири взаємопов'язані модулі з чіткими ролями:

- Модуль керування – координує роботу підсистем і визначає послідовність дій, забезпечуючи узгодженість між агентами та стабільність сценарію.
- Логічний модуль – виконує міркування, планування та вибір рішень, фактично слугуючи виконавцем обчислювальних і reasoning-процедур.
- Модуль пам'яті – зберігає результати попередніх взаємодій і релевантні знання, поєднуючи короткочасний контекст із довготривалими репрезентаціями.
- Пристрої введення-виведення – забезпечують контакт із зовнішнім середовищем: взаємодію з користувачами та іншими агентами, а також роботу з сенсорами, GUI чи роботизованими виконавцями.

Кожен із модулів підтримує чотири ключові типи операцій: декомпозицію завдань, саморефлексію, обробку пам'яті та виклик зовнішніх інструментів. Така структуризація підвищує якість планування, робить хід міркувань прозорішим для учня та сприяє адаптації до педагогічного контексту.

- Декомпозиція завдань – логічний модуль розкладає складні навчальні цілі на послідовні підзадачі, що полегшує поетапну перевірку й корекцію. Це зменшує накопичення помилок і дозволяє вбудовувати перевірки на кожному кроці.

- Саморефлексія – модуль керування ініціює цикли перевірки власних рішень та альтернативних відповідей, у тому числі через узгодження позицій кількох агентів. Це знижує ризик хибних узагальнень і покращує пояснюваність.

- Обробка пам'яті – модуль пам'яті поєднує оперативний контекст діалогу з довготривалими векторними уявленнями про прогрес студента, типові помилки та попередні підказки, щоб персоналізувати наступні кроки.

- Виклик інструментів – логічний модуль за ініціативи контролера може звертатися до зовнішніх сервісів - від розпізнавання формул і зображень до обчислювальних рушіїв і спеціалізованих освітніх API - розширюючи межі базових можливостей агента.

У межах цієї архітектури доцільно розрізняти дві взаємопов'язані динаміки розвитку. Внутрішній цикл функціонування стосується самої групи агентів: через повторювані цикли декомпозиції, рефлексії, оновлення пам'яті та узгодження рішень зростає колективна компетентність і стійкість до помилок. Зовнішній цикл описує шлях навчання студента: на основі діагностики поточного розуміння та історії

взаємодій система пропонує адаптивні завдання, підсилює здатність відстежувати власне розуміння і сприяє поетапній побудові знань.

Практичний ефект для освітніх сценаріїв проявляється у трьох напрямках: по-перше, прозорість мислення (учень бачить обґрунтований шлях розв'язання замість «чорної скриньки»); по-друге, стабільність і відтворюваність (цикли самокорекції зменшують випадкові збої); по-третє, персоналізація (пам'ять і зовнішні інструменти дають змогу узгодити складність і форму подання з індивідуальними потребами).

1.5 Стан дослідження мультиагентних освітніх систем

Сформований у попередніх підрозділах аналіз показав, що поєднання архітектурних принципів мультиагентних систем із сучасними великими мовними моделями створює якісно нові можливості для організації інтелектуального освітнього середовища. У цьому контексті особливий інтерес становлять конкретні реалізації мультиагентних AI-систем, які вже пройшли початкову апробацію та демонструють різні підходи до розподілу ролей між агентами, організації їхньої взаємодії та співпраці з реальними студентами. Саме тому далі доцільно зосередитися на огляді кількох представницьких систем, що відображають ключові напрямки розвитку агентних технологій в освіті.

Фреймворк AWE (Agentic Workflow for Education)[14], подає чотирикомпонентну модель, що охоплює саморефлексію, задіяння інструментів, планування завдань і багатоагентну колаборацію. На відміну від лінійних взаємодій у форматі «запит–відповідь», AWE

орієнтується на динамічні, нелінійні робочі процеси, у яких агенти можуть самостійно формулювати підцілі, перевіряти проміжні результати та узгоджено співпрацювати. Концепція фреймворку пов'язана з підходами багатоагентних систем і спирається на розширення архітектурного бачення MAS, де базові можливості (саморефлексія та інструментальні виклики) забезпечують технічний фундамент, а планування і міжагентна взаємодія відкривають шлях до масштабування складних освітніх сценаріїв.

Практичну доцільність AWE ілюструє кейс автоматизованого генерування математичних тестів: згенеровані завдання виявилися статистично порівнюваними з людськими за якістю контекстуальної відповідності та раціональності варіантів відповідей (наприклад, повідомляється про р-значення на рівні $P \geq 0.439$ для окремих критеріїв), що свідчить про відсутність статистично значущих відмінностей у цих аспектах. Додатково, в іншому порівнянні агентна колаборативна система перевершила GPT-4 за раціональністю варіантів ($P = 0.043$) та зв'язністю формулювання стема питання ($P = 0.009$), що вказує на перевагу агентного підходу над одиничною LLM-моделлю в умовах цього експерименту. Такі результати надають емпіричну підтримку ефективності AWE у задачі конструювання тестових завдань, водночас не підмінюючи необхідність подальших досліджень у ширшому спектрі освітніх застосувань [13].

Узагальнюючи, AWE пропонує перехід від ручного налаштування кроків до самокерованого інтегрування робочих процесів: агенти формують план, звертаються до зовнішніх інструментів та ресурсів, оцінюють власні проміжні висновки і координують дії у команді. Така організація відкриває можливості для персоналізованої підтримки

навчання, симуляцій без ризику для здобувачів освіти та орієнтованого на дані прийняття рішень у викладанні й управлінні освітою. Доцільно трактувати AWE як теоретично обґрунтований каркас і сукупність практик побудови агентних робочих процесів, які демонструють перспективні результати в окремих кейсах і можуть масштабуватися до ширших освітніх контекстів за наявності належних емпіричних підтверджень.

Система SimClass. Дослідження [6] представили SimClass – мультиагентну симуляцію класу з реальною участю студентів. Архітектура поєднує чітко визначені ролі (вчитель, асистент і кілька типів «однокласників» із різними намірами: занотовувач, допитливий, «класний жартівник», «глибокий мислитель») та керуючий модуль Session Controller, який, спираючись на поточний стан заняття й навчальні матеріали, автоматично визначає, хто й яку дію виконує далі (пояснення, перехід до наступного слайду, уточнювальні запитання, підтримання дисципліни тощо). Оцінювання проводилося за Flanders Interaction Analysis System (FIAS) та в межах рамки Community of Inquiry (CoI) для вимірювання когнітивної, викладацької та соціальної присутності.

Отримані результати свідчать, що мультиагентний підхід здатен підтримувати «живу» динаміку класу з регулярною взаємодією вчитель-студент і студент-студент, а також породжувати спонтанно сформовані групові поведінки: кооперативне пояснення матеріалу (учитель + асистент + «однокласники»), поглиблені обговорення з багатокроковими запитаннями, емоційну підтримку в моменти зіткнення з складнощами та м'яке перенаправлення уваги в разі відхилення від теми заняття. Практична доцільність підходу підтверджується такими факторами: по-

перше, збалансований «профіль розмови» за FIAS із наявністю як ініціацій учителем, так і значущої частки студентських ініціацій; по-друге, позитивна асоціація інтенсивності взаємодії з навчальними результатами (студенти, які частіше й змістовніше комунікують, мають вищі підсумкові бали в поточних перевірках знань); по-третє, в налаштуваннях, під час відключення однокласників-агентів знижує довжину відповідей і активність користувачів, тоді як повна мультиагентна конфігурація підсилює когнітивну й соціальну присутність у термінах CoI. Сукупно це вказує, що контроллер сесії та різнорольова взаємодія не лише відтворюють структуру заняття, а й підвищують залученість, якість діалогу та утримання матеріалу.

EduVerse: Симуляційний простір мультиагентних взаємодій. Система EduVerse є користувацько-визначеним мультиагентним симуляційним простором для освіти, що підтримує налаштування середовища, агентів і сесій, а також інтеграцію реальних учасників у режимі human-in-the-loop. Побудована на багатошаровій архітектурі CIE (Cognition-Interaction-Evolution), вона забезпечує послідовність індивідуального мислення, автентичність взаємодії та довготривалу адаптацію в когнітивній, емоційній і поведінковій сферах.

У валідаційних експериментах симульовані показники IRF (Initiation-Response-Feedback) для різних тематик текстів і форматів організації класу тісно наближаються до емпіричних значень у реальних класах, що свідчить про педагогічну реалістичність взаємодії (наприклад, симуляція $\approx 0.33-0.55$ проти реальних $\approx 0.37-0.49$; можливі варіації за комбінаціями тем та середовища охоплюють $\sim 0.28-0.64$). Автентичність групової взаємодії додатково підтверджується мережевими патернами: густина графа взаємодій $0.267-0.400$ і середній

ступінь 1.2-1.667 інтерпретуються як реалізація близько 27-40% потенційних взаємодій між учнями, що відповідає участі в умовах, коли урок переважно веде вчитель та відтворює реалістичну, не повністю насичену мережу контактів і помітний розподіл ролей [2].

Довготривала динаміка навчання у межах EduVerse відображається через позитивні переходи станів у поведінковій, емоційній і когнітивній площинах. Формально показник $R+$ визначається як частка позитивних переходів між послідовними кроками відносно всіх змін станів, обчислюваний окремо для поведінки, емоцій та мислення; агрегування за сесіями демонструє зростаючі тренди для поведінки та емоцій за повільнішого, але накопичувального мисленевого прогресу. Узагальнено повідомляється про середнє підвищення позитивних переходів близько 11.7% упродовж багатосесійних сценаріїв, що підтверджує спроможність системи відтворювати поступове навчальне зростання та стабільний, але адаптивний індивідуальний прогрес.

OnlineMate: Система з Theory of Mind. OnlineMate - мультиагентна система на базі LLM з інтегрованою теорією ментальних станів (Theory of Mind)[1], яка моделює ролі однолітків і під час спільних обговорень оцінює когнітивні та психологічні стани студента (зокрема нерозуміння, розгубленість і рівень зацікавленості/мотивації). Система динамічно коригує стратегії взаємодії, підтримуючи просування до більш поглибленого мислення. Ключовий механізм ToM реалізовано як триетапний процес: по-перше, формується набір гіпотез щодо поточного стану студента з урахуванням контексту; по-друге, ці гіпотези уточнюються і відбираються відповідно до ролі агента та правил навчального сценарію; по-третє, генерується відповідь із вбудованою

самоперевіркою корисності та підведенням до глибших мисленнєвих операцій.

Доцільність застосування такої технології підтверджується низкою показників, характерних для систем із ToM та когнітивною підтримкою у мультиагентних середовищах. По-перше, фіксується зростання мисленнєвої складності студентських висловлювань за таксономією Блума: у типових сесіях спостерігається перехід від рівнів застосування/аналізу до оцінювання/творення протягом перших кількох кіл обговорення, що свідчить про зсув від відтворення знань до їх критичного переосмислення та генерування нових ідей. По-друге, поліпшуються індикатори якості дискусії за університетськими рубриками (наприклад, частота й релевантність внесків, спільне вибудовування знань, аргументованість позицій): учасники частіше роблять змістовні, доречні до контексту репліки, частіше уточнюють і розвивають думки інших, а співвідношення більш складних та комплексних запитань до “фактологічних запитань” зростає. По-третє, у динаміці емоційних станів помітне більш стабільне утримання зацікавленості: коливання емоційної залученості зменшуються, а поріг дострокового припинення взаємодії долається рідше, що свідчить про більш точне потрапляння системи в зону найближчого розвитку студента. Нарешті, конфігураційні аналізи показують, що невелика кількість узгоджених агентів (близько чотирьох ролей) забезпечує найкращий баланс між розмаїттям поглядів і мисленнєвим навантаженням, натомість збільшення кількості агентів понад цей поріг призводить до плато або навіть зниження якості через перевантаження інформацією.

2 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ ВЗАЄМОДІЇ АІ-АГЕНТІВ ТА АЛГОРИТМІВ ОБРОБКИ НАВЧАЛЬНОГО КОНТЕНТУ

2.1 Обґрунтування вибору методів теоретичних та експериментальних досліджень, програмного забезпечення

Для досягнення мети кваліфікаційної роботи – підвищення ефективності дистанційного навчання шляхом створення мультиагентної системи – необхідно застосувати міждисциплінарний підхід. Складність об'єкта дослідження, який включає як детерміновані алгоритми (програма навчання), так і стохастичні процеси (поведінка студентів, динаміка групи), вимагає обґрунтованого вибору методів.

2.1.1 Методи теоретичних досліджень

В основу теоретичної бази покладено концепцію навчання з підкріпленням у мультиагентних системах (Multi-Agent Reinforcement Learning - MARL). Як зазначають у огляді [7], класичні методи машинного навчання не здатні ефективно працювати в умовах, де дії одного агента змінюють стан середовища для інших. Саме тому обрано підхід децентралізованого управління, де кожен агент (студент або вчитель) діє на основі локальних спостережень.

Для вирішення задачі адаптивності та персоналізації обрано метод імовірнісного моделювання, а саме – Гауссівські процеси (Gaussian Processes – GP). На відміну від глибоких нейронних мереж, які потребують великих масивів даних, GP дозволяють будувати довірчі інтервали та оцінювати невизначеність знань студента навіть на малих вибірках даних, що підтверджується дослідженнями [15].

Третім теоретичним стовпом є теорія графів та спектральний аналіз мереж. Оскільки взаємодія в групі розглядається як обмін інформацією в мережі, стабільність системи безпосередньо залежить від топології зв'язків. Використання спектрального радіуса матриці суміжності як критерію збіжності базується на роботі [16], що дозволяє математично гарантувати відсутність хаотичної поведінки агентів.

2.1.2 Методи експериментальних досліджень

Основним методом експериментального дослідження обрано імітаційне комп'ютерне моделювання. Проведення педагогічного експерименту з реальними учасниками на етапі розробки архітектури є недоцільним насамперед через ризик негативного впливу на навчальний процес у разі некоректної роботи агентів. Крім того, критичним фактором є часові витрати: навчання агентів вимагає тисяч епізодів взаємодії, що фізично неможливо реалізувати в масштабі реального часу. Важливою перевагою обраного методу також є відтворюваність результатів: симуляція дозволяє фіксувати випадкові сценарії (Random Seeds) та

точно відтворювати умови експерименту для верифікації гіпотез, що є недосяжним у реальних умовах.

2.1.3 Обґрунтування вибору програмного забезпечення

Для реалізації розроблених моделей обрано мову програмування Python 3.10+, яка є стандартом де-факто в науці про дані завдяки потужній екосистемі.

- LLM API – для забезпечення семантичної генерації тексту та логічного виводу.
- PyTorch – як допоміжний інструмент для векторних обчислень.
- NetworkX – для генерації та аналізу топології графів взаємодії агентів.
- NumPy – для реалізації стохастичних алгоритмів (розподіли Бернуллі).

2.2 Математична модель об'єкта дослідження

Розробка математичної моделі мультиагентної освітньої системи (далі – МАОС) базується на інтеграції теоретико-ігрових моделей динаміки навчання в мережах, методів модульної декомпозиції задач та ймовірнісного моделювання адаптивності. Така комбінація дозволяє

формалізувати процеси структурного планування, персоналізації та стабілізації взаємодії в умовах невизначеності.

2.2.1 Динаміка мультиагентної мережі та умови стабільності

Фундаментальну поведінку системи моделюємо як динаміку взаємодії агентів у випадковій мережі, спираючись на формалізм, запропонований у роботі [16]. Система розглядається як граф $G = (\mathcal{N}, E)$, де $\mathcal{N} = \{1, \dots, N\}$ – множина агентів, а E – множина комунікаційних зв'язків.

Стан кожного агента k (його політика π_k або стратегія вибору дії) еволюціонує з часом t згідно з диференціальним рівнянням динаміки Q-навчання в безперервному часі (QLD)[17], що враховує ентропійну регуляризацію та представлено формулою 2.1.

$$\frac{\dot{x}_{ki}}{x_{ki}} = r_{ki}(x_{-k}) - \langle x_k, r_k(x) \rangle + T_k \sum_{j \in A_k} x_{kj} \ln \left(\frac{x_{kj}}{x_{ki}} \right) \quad (2.1)$$

де \dot{x}_{ki} – швидкість зміни ймовірності вибору дії i агентом k у часі (похідна за часом);

x_{ki} – ймовірність вибору дії i агентом k згідно з його політикою π_k ;

$r_{ki}(x_{-k})$ – очікувана винагорода агента k за вибір дії i , що залежить від стратегій інших агентів (сусідів) x_{-k} ;

$\langle x_k, r_k(x) \rangle$ – середній очікуваний виграш агента k при його поточній змішаній стратегії (скалярний добуток вектора стратегії на вектор винагород);

T_k – параметр температури, який регулює баланс між дослідженням (exploration) та використанням знань (exploitation);

A_k – множина всіх можливих дій, доступних агенту k ;

j – індекс, що використовується для підсумовування по всіх альтернативних діях з множини A_k .

Стаціонарним станом цієї динаміки є рівновага квантового відгуку (QRE). Для забезпечення стабільності роботи системи необхідно виконання структурної умови, виведеної в [15]. Якщо $\rho(G)$ – спектральний радіус матриці суміжності графа взаємодій, а δ_I – параметр інтенсивності кооперації, то умова стабільності представленою формулою 2.2.

$$T_k > \delta_I \cdot \rho(G) \quad (2.2)$$

Ця нерівність є ключовим обмеженням при проектуванні архітектури: для стабільної комунікації необхідно обмежувати щільність зв'язків (зменшувати $\rho(G)$) або підвищувати стохастичність поведінки (T_k).

2.2.2 Модульна декомпозиція та розподіл задач

Для реалізації педагогічної стратегії вчителя використовується модель Modular Task Decomposition, описана у роботі [19]. Глобальна

освітня задача \mathcal{T} трансформується у єдине семантичне подання $h_{\mathcal{T}} \in R^d$ за допомогою енкодера.

Для кожного модуля m генерується специфічне подання h_m за допомогою механізму уваги (Attention Mechanism) із запитом q_m (формула 2.3) [18, 20].

$$h_m = \text{softmax}\left(\frac{q_m^\top h_{\mathcal{T}}}{\sqrt{d}}\right) h_{\mathcal{T}} \quad (2.3)$$

де q_m – вектор-запит (query), асоційований з m модулем, що визначає критерії вибору релевантної інформації;

$h_{\mathcal{T}}$ – єдине семантичне подання глобальної освітньої задачі (вихід енкодера), яке в даному контексті виконує роль ключів та значень;

d – розмірність векторного простору прихованих станів, що використовується для нормалізації скалярного добутку;

softmax – функція активації, що перетворює ваги уваги у ймовірнісний розподіл.

Це дозволяє виділити з глобального контексту лише інформацію, релевантну для конкретної ролі студента або підетапу навчання. Глобальний результат O_t формується як зважена агрегація виходів окремих агентів o_t^m (формула 2.4).

$$O_t = \sum_{m \in \mathcal{M}} w_t^m o_t^m, \quad \sum w_t^m = 1 \quad (2.4)$$

де \mathcal{M} – множина всіх активних агентів (модулів), залучених до вирішення задачі;

w_t^m – ваговий коефіцієнт m -го агента в момент часу t , що відображає ступінь довіри до його відповіді або його релевантність ($\sum_{m \in \mathcal{M}} w_t^m = 1$);

o_t^m – локальний вихід (відповідь або проміжне рішення), згенерований m -м агентом.

Якість виконання контролюється через мінімізацію композитної функції втрат \mathcal{L}_{total} , яка формалізується формулою 2.5.

$$\mathcal{L}_{total} = \mathcal{L}_{task} + \lambda_{cons} \sum_{i,j} \|o^i - o^j\|^2 + \lambda_{bal} \text{Var}(\{\text{workload}_m\}) \quad (2.5)$$

де \mathcal{L}_{total} – загальне значення композитної функції втрат, що підлягає мінімізації;

\mathcal{L}_{task} – функція втрат основного завдання (наприклад, перехресна ентропія), яка оцінює відхилення згенерованої відповіді від еталонної;

λ_{cons} та λ_{bal} – гіперпараметри регуляризації, які визначають ваговий внесок узгодженості агентів та балансування навантаження у загальну функцію втрат;

$|o^i - o^j|^2$ – міра розбіжності між виходами агентів i та j (штраф за неузгодженість думок агентів);

$\text{Var}(\{\text{workload}_m\})$ – дисперсія розподілу обчислювального навантаження між агентами (зменшення цього показника запобігає перевантаженню окремих модулів).

2.2.3 Адаптивність та персоналізація через Гауссівські процеси

Для моделювання невизначеності знань студента використовується підхід InqEduAgent [15]. Ефект від взаємодії (Learning Gain) моделюється як функція представлена формулою 2.6

$$y = f(x) + \epsilon, \quad (2.6)$$

де $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ – гауссівський шум спостереження. Невідома функція ефективності $f(x)$ моделюється як Гауссівський процес (GP, формула 2.7).

$$f(x) \sim GP(m(x), k(x, x')) \quad (2.7)$$

де $m(x)$ – функція середнього значення, яка відображає очікуване значення функції в точці x до отримання будь-яких спостережень (часто приймається рівною нулю, $m(x) = 0$);

$k(x, x')$ – коваріаційна функція або ядро, що визначає міру подібності між значеннями функції у двох довільних точках вхідного простору x та x' ;

x, x' – довільні вектори параметрів (точки) з простору пошуку рішень.

У якості коваріаційної функції використовується радіальна базисна функція (RBF, формула 2.8).

$$k_{RBF}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{|x_i - x_j|^2}{2l^2}\right) \quad (2.8)$$

де l – масштаб кореляції;

σ_f^2 – дисперсія процесу, що визначає амплітуду відхилень функції від середнього значення;

$|x_i - x_j|^2$ – квадрат евклідової відстані між двома точками вхідних даних.

Метою агента є вибір дії a^* , яка максимізує функцію очікуваної корисності (Expected Improvement). Оскільки агент стикається з багатокритеріальною задачею, пошук оптимального рішення здійснюється на множині Парето-оптимальних кандидатів (формула 2.9).

$$a^* = \operatorname{argmax}_{x \in \text{ParetoFront}} E[f(x)] \quad (2.9)$$

де x – потенційний кандидат (дія або набір параметрів) з доступного простору пошуку;

ParetoFront – множина Парето-оптимальних рішень (фронт Парето), до якої обмежено простір пошуку для врахування багатокритеріальності задачі;

argmax – оператор, який повертає саме те значення аргументу x , при якому цільова функція досягає свого максимуму;

$E[\cdot]$ – оператор математичного сподівання (очікуване значення);

$f(x)$ – функція корисності (Expected Improvement), значення якої прогнозується на основі Гаусівського процесу.

2.2.4 Ефективність комунікації та метрики оптимізації

Для оцінки якості діалогу та уникнення інформаційного шуму вводиться метрика Information Entropy Efficiency Index (IEI).[19] Повідомлення агента m розглядається як стиснене кодування його внутрішнього стану. Ефективність комунікації Φ_{IEI} визначається через ентропію $H(M)$ представлено формулою 2.10.

$$\Phi_{IEI} = 1 - \frac{H(M)}{H_{max}} \quad (2.10)$$

де $H(M) = -\sum_k p(m_k) \log_2 p(m_k)$ - ентропія Шеннона [21] для розподілу ймовірностей токенів у повідомленні.

H_{max} – максимально можлива ентропія для даного словника (досягається при рівномірному розподілі ймовірностей токенів), яка використовується як нормувальний коефіцієнт;

$p(m_k)$ – ймовірність появи k -го токена (слова або символу) у повідомленні;

m_k – окремий токен з доступного словника агента

Нормалізація на H_{max} дозволяє зробити метрику Φ_{IEI} незалежною від довжини повідомлення, що дає змогу коректно порівнювати ефективність висловлювань агентів з різним обсягом генерації. Високе значення Φ_{IEI} свідчить про низьку невизначеність, тобто агент формулює думку чітко і конкретно.

Для обґрунтування необхідності різних ролей (Вчитель, Розумник, Жартівник) використовується індекс спеціалізації (Specialization

Efficiency Index - SEI). Він базується на мінімізації косинусної подібності між повідомленнями різних агентів (формула 2.11).

$$\Phi_{SEI} = \frac{1}{N(N-1)} \sum_{i \neq j} \cos(\theta_{i,j}) \quad (2.11)$$

де N – загальна кількість агентів у мультиагентній системі;

$\theta_{i,j}$ – кут між векторними представленнями (ембедингами) повідомлень або дій агента i та агента j ;

$\cos(\theta_{i,j})$ – косинусна подібність, яка показує, наскільки семантично близькими є відповіді двох агентів (значення, близьке до 1, означає дублювання функцій, близьке до 0 – ортогональність та унікальність ролей);

$\frac{1}{N(N-1)}$ – нормувальний множник, що усереднює значення подібності по всіх можливих парах агентів.

Таким чином, глобальна цільова функція системи, яку необхідно мінімізувати при проектуванні політик агентів, має вигляд рівняння представленого формулою 2.12.

$$L = \mathcal{L}_{task} + w_{IEI}(1 - \Phi_{IEI}) + w_{SEI}\Phi_{SEI} \quad (2.12)$$

де \mathcal{L}_{task} – помилка виконання основного завдання (наприклад, правильність відповіді на запитання користувача);

w_{IEI} – ваговий коефіцієнт, що регулює важливість чіткості та інформативності повідомлень (штраф за високу невизначеність);

$(1 - \Phi_{IEI})$ – компонент втрат, пов'язаний з ентропією: мінімізація цього виразу еквівалентна максимізації індексу Φ_{IEI} (підвищенню чіткості);

w_{SEI} – ваговий коефіцієнт, що регулює важливість різноманітності ролей агентів;

Φ_{SEI} – компонент втрат, пов'язаний зі схожістю: його мінімізація змушує агентів розходитися у своїх стратегіях та уникати дублювання.

Ця формула математично формалізує вимогу: система повинна вирішувати задачу (\mathcal{L}_{task}), говорити чітко (максимізувати Φ_{IEI}) та діяти різноманітно (мінімізувати подібність Φ_{SEI}).

2.3 Розробка алгоритмічного забезпечення та протоколів взаємодії агентів

У даному підрозділі представлено алгоритмічну реалізацію математичної моделі MAOC. Для практичної реалізації теоретичних основ, описаних у п. 2.2, обрано архітектуру на базі Великих Мовних Моделей (LLM). Це дозволяє замінити явне обчислення складних ймовірнісних функцій (таких як ядра Гауссівських процесів) на їх семантичну апроксимацію через механізми уваги (Attention) та In-Context Learning.

2.3.1 Архітектура агентів та рольова модель

Система складається з трьох гетерогенних агентів. У термінах розробленої моделі, системний промпт (System Instruction) π_i виступає набором параметрів, що визначають стохастичну політику агента π_i .

Формально процес вибору дії (генерації відповіді R_t) є вибіркою з розподілу, заданого політикою, яка представлена формулою 2.13.

$$R_t \sim \pi_i(\cdot | H_t; P_i) \equiv G(P_i, C_{lec}, H_t) \quad (2.13)$$

де π_i – політика агента;

G – функція генерації LLM;

C_{lec} – контекст лекції або завантаженого файлу на який спираються агенти

P_i – рольова інструкція (промпт), що фіксує поведінку;

H_t – контекст (історія діалогу).

Специфікація агентів:

1. Агент-Вчитель ($A_{Teacher}$), є лідером процесу. Він виконує адаптацію складного тексту лекції C_{lec} у спрощену форму, реалізуючи методику модульної декомпозиції задач. Його промпт налаштований на структурування знань та мінімізацію логічних розривів у поясненні, що відповідає мінімізації \mathcal{L}_{task} .

2. Агент "Студент-Розумник" (A_{Smart}) виконує роль з високою пізнавальною потребою. Його поведінка є семантичною апроксимацією Гауссівського процесу, описаного в п. 2.2.3. Промпт агента

налаштований на виявлення "білих плям" у поясненні вчителя (областей з максимальною дисперсією σ^2). Задаючи уточнюючі питання, агент намагається максимізувати інформаційний приріст (Information Gain). Семантичний аналіз, який виконує LLM, фактично виступає евристичним пошуком на Парето-фронті, відсіюючи питання, які є занадто тривіальними або надмірно складними.

3. Агент "Студент-Жартівник" (A_{Joker}), є соціальним регулятором і відповідає за підтримку залученості, що корелює з метриками ефективності комунікації в соціальних групах. Його завдання - генерувати гумористичні аналогії. Це забезпечує мінімізацію косинусної подібності Φ_{SEI} (п. 2.2.4), оскільки його вектор поведінки є ортогональним до векторів Вчителя та Розумника.

2.3.2 Алгоритм взаємодії агентів

Для забезпечення математичної умови стабільності системи (п. 2.2.1) в алгоритм введено стохастичний контроль активації агентів. Це дозволяє уникнути конфліктів та хаотичного нагромадження повідомлень.

Цикл роботи системи на кроці t виглядає наступним чином:

Крок 1: Генерація навчального контенту (Вчитель)

Агент-вчитель зчитує фрагмент лекції і генерує пояснення M_{Teach} .

Ця дія є детермінованою (формула 2.14).

$$P(\text{Action}|A_{Teacher}) = 1 \quad (2.14)$$

Результат M_{Teach} стає вхідним стимулом для наступного кроку.

Крок 2: Оцінка ймовірності реакції (Студенти)

Після репліки вчителя кожен агент-студент A_k (де $k \in \{Smart, Joker\}$) приймає рішення про втручання. Це рішення моделюється використовуючи розподіл Бернуллі з параметром p_k (ймовірність втручання)(формула 2.15).

$$Activation_k \sim \text{Bernoulli}(p_k) \quad (2.15)$$

- Якщо $Activation_k = 0$, агент пропускає хід
- Якщо $Activation_k = 1$, агент генерує відповідь R_k , враховуючи репліку вчителя M_{Teach} .

Параметри p_{Smart} та p_{Joker} налаштовуються експериментально (наприклад, $p = 0.3$). Це є практичною реалізацією обмеження на спектральний радіус графа $\rho(G)$, виведеного в [16]. Штучне обмеження частоти відповідей ($p < 0.5$) гарантує розрідженість комунікації, що запобігає виникненню хаотичних осциляцій у діалозі.

Крок 3: Оновлення контексту

Усі згенеровані репліки додаються до глобальної історії H_{t+1} , яка подається на вхід LLM на наступній ітерації. Це забезпечує семантичну зв'язність діалогу.

3 РОЗРОБКА ЗАСОБІВ МОДЕЛЮВАННЯ ПРОГРАМНОГО КОМПЛЕКСУ ГІБРИДНОГО НАВЧАННЯ

Розробка логічної моделі є ключовим етапом проектування, який дозволяє перейти від теоретичних засад до практичної реалізації програмного засобу. У цьому підрозділі доцільно розмістити та проаналізувати діаграми, що описують функціональні вимоги, логіку поведінки системи та динамічні стани, в яких вона може перебувати під час виконання завдань. Такий підхід забезпечує комплексне бачення архітектури ПМК, формалізує сценарії взаємодії користувача з інтерфейсом та визначає алгоритми обробки даних на рівні абстракції, що є необхідним фундаментом для подальшої програмної реалізації.

Для визначення функціональних меж системи та ролей користувачів було розроблено діаграму прецедентів (рис. 3.1).

Діаграма ідентифікує двох основних акторів з боку клієнта: «Користувач» (студент/викладач) та «Розробник», який успадковує права користувача, але має додаткові можливості (збір статистики). Зовнішньою системою виступає «OpenAI API», що забезпечує інтелектуальну обробку запитів.

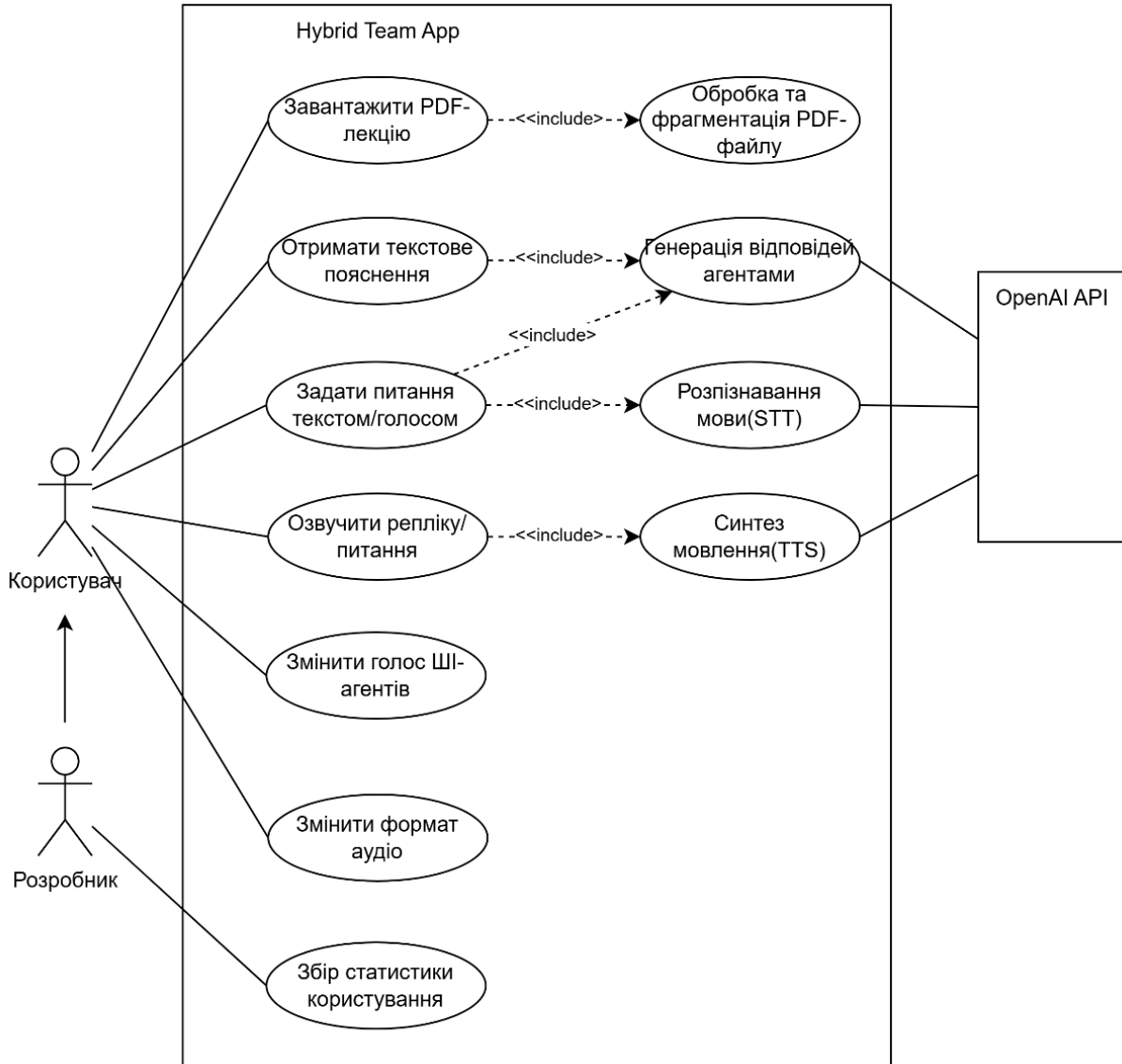


Рисунок 3.1 - Діаграма прецедентів функціональних можливостей користувача та адміністратора у системі гібридного навчання

Ключові прецеденти системи включають:

- Завантаження та обробка контенту – користувач ініціює завантаження PDF-лекції, що через відношення <<include>> автоматично викликає процес фрагментації файлу.

– Генерація пояснень – отримання текстового пояснення передбачає взаємодію з LLM для генерації реплік агентів (Викладача, Розумника, Жартівника).

– Комунікація – система підтримує мультимодальну взаємодію. Прецедент «Задати питання» реалізовано двома шляхами: текстовим та голосовим (з використанням STT – Speech-to-Text).

– Синтез мовлення – озвучення реплік агентів реалізується через звернення до сервісу TTS (Text-to-Speech), що є залежним прецедентом.

Алгоритмічна послідовність дій користувача та реакцій системи під час сеансу навчання представлена на діаграмі діяльності (рис. 3.2).

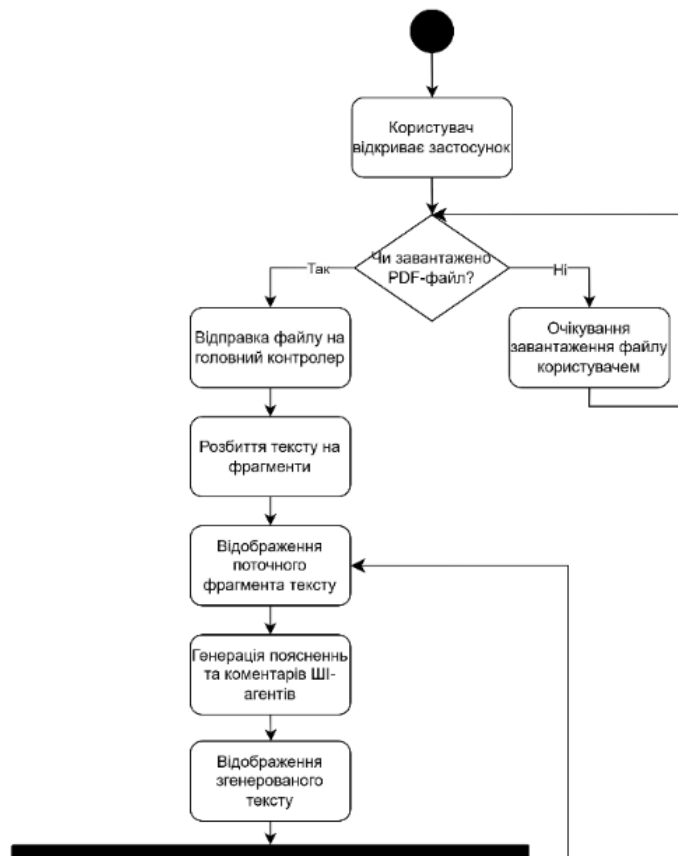
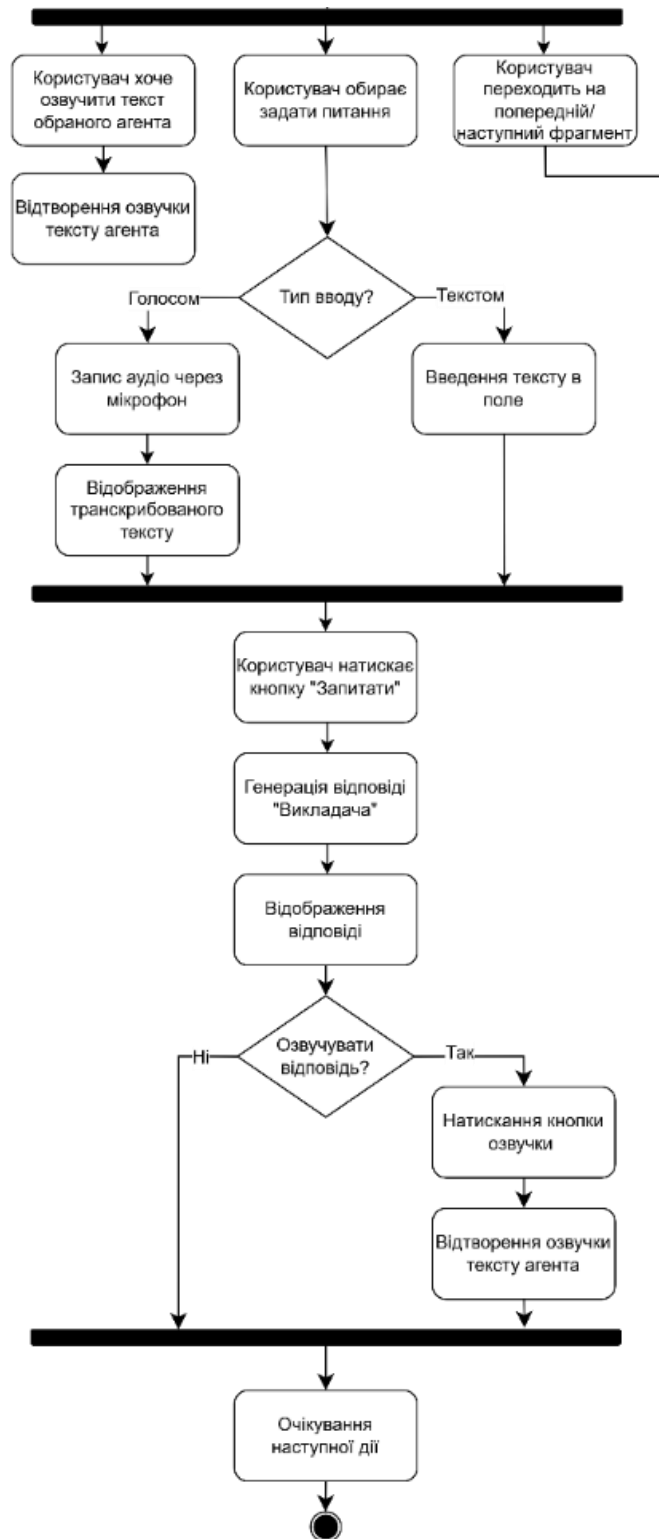


Рисунок 3.2 - Діаграма діяльності алгоритму роботи користувача з системою



Продовження рисунку 3.2

Процес починається із завантаження PDF-файлу. Логіка розгалуження (ромб «Чи завантажено PDF?») забезпечує валідацію вхідних даних: система не переходить до основного циклу роботи, доки файл не буде коректно оброблений.

Основний потік управління є ітеративним:

- Система відображає поточний фрагмент тексту та генерує пояснення ШІ-агентів.

- Користувач має вибір – перейти до навігації (наступний/попередній фрагмент) або ініціювати режим «Питання-Відповідь».

- Взаємодія при запитаннях також розгалужується за типом вводу (голос/текст), проте обидва потоки сходяться в точці генерації відповіді «Викладача».

Діаграма демонструє замкнені цикли, що дозволяє користувачеві безперервно працювати з матеріалом лекції без необхідності перезавантаження сторінки.

Для моделювання динамічної поведінки системи та переходів між її життєвими циклами розроблено діаграму станів (рис. 3.3).

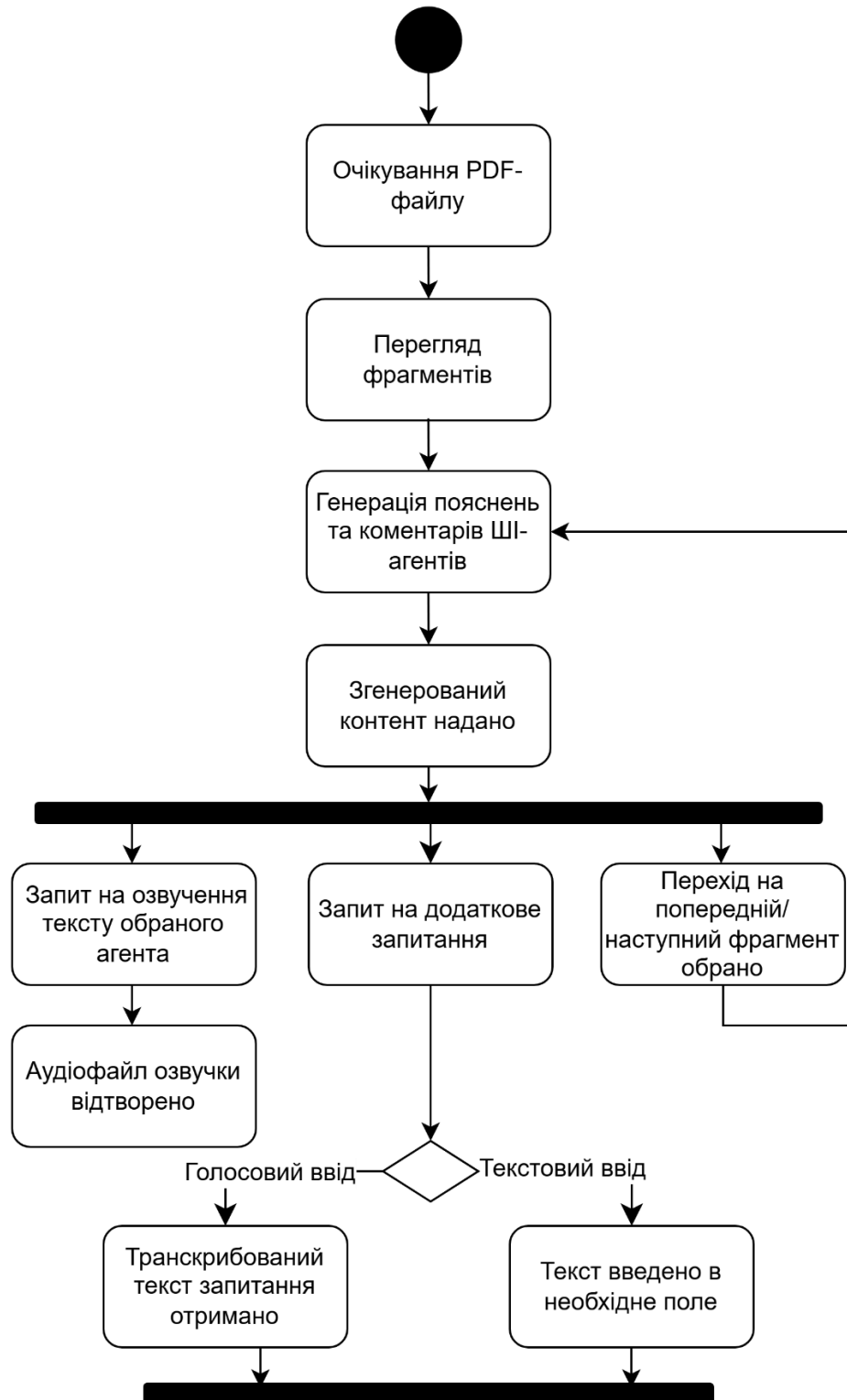
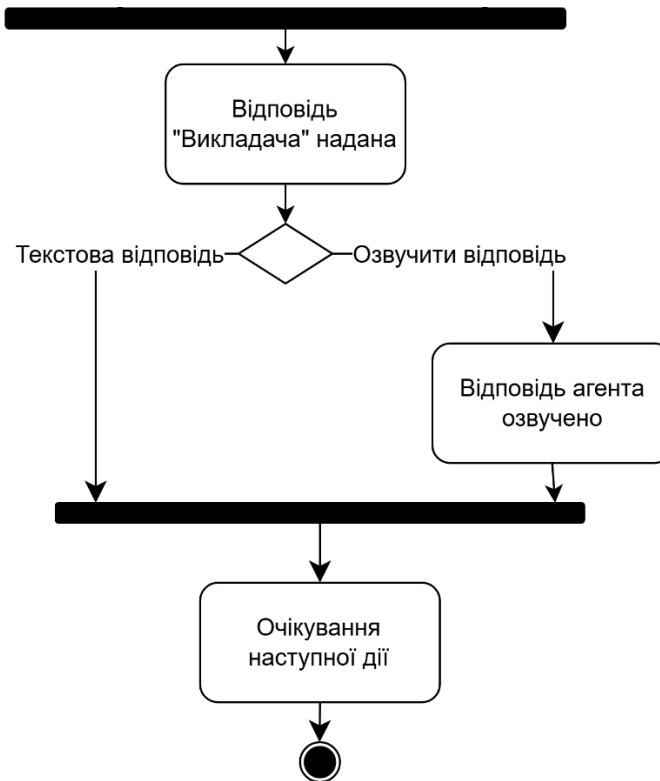


Рисунок 3.3 – Діаграма станів життєвого циклу навчальної сесії



Продовження рисунку 3.3

Система починає роботу зі стану очікування вхідних даних («Очікування PDF-файлу»). Після успішної обробки відбувається перехід у стан «Перегляд фрагментів», який є центральним вузлом навігації.

Особливістю моделі є наявність асинхронних станів обробки даних, таких як:

- Генерація пояснень – стан, у якому система звертається до LLM для створення сценарію діалогу агентів.
- Транскрибування/Введення – стани очікування введення питання користувачем.
- Аудіовідтворення – стан, у якому система блокує інші аудіопотоки для коректного відтворення синтезованої мови.

Вихід із системи або завершення роботи повертає автомат у початковий стан або стан очікування наступної дії, що забезпечує стабільність сесії.

Архітектура програмного забезпечення спроектована за модульним принципом із використанням патерну Controller-Service, що відображено на діаграмі класів (рис. 3.4).

Центральним елементом системи є клас Головний контролер (відповідає файлу main.py на базі FastAPI), який оркеструє запити від клієнта та розподіляє їх між спеціалізованими сервісами:

- Сервіс розбиття PDF – інкапсулює логіку читання байтових потоків та сегментації тексту, перетворюючи документ на структурований набір фрагментів.

- Сервіс чату – зберігає атрибути для системних промптів агента формує історію повідомлень, делегуючи безпосередню генерацію відповідей класу OpenAI API.

- Сервіси TTS та STT – забезпечують підготовку даних для перетворення тексту в аудіо та навпаки. Вони ізолюють логіку роботи з мультимедійними файлами та використовують методи класу OpenAI API для виконання транскрибації та синтезу мовлення.

- Клас OpenAI API – цей клас інкапсулює низькорівневу логіку HTTP-запитів до серверів OpenAI, реалізуючи методи генерації текстових відповідей, аудіосинтезу та розпізнавання мови.

Моделі даних (фрагменти, питання), забезпечують сувору типізацію даних, що передаються між фронтендом та бекендом.

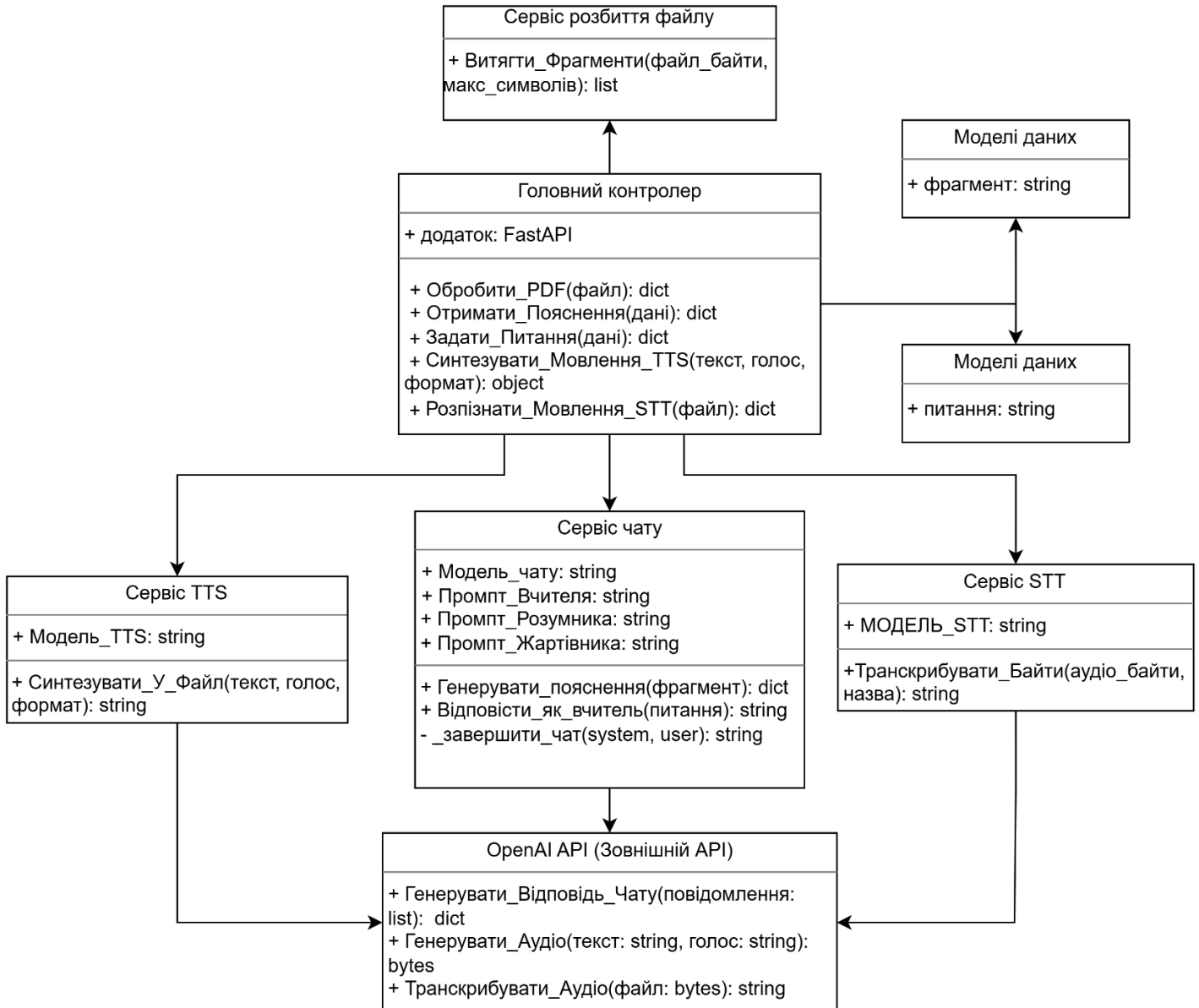


Рисунок 3.4 - Діаграма класів програмної архітектури серверної частини та взаємозв'язків сервісних компонентів

Послідовність взаємодії та передачі даних між компонентами системи під час виконання основних сценаріїв роботи із застосунком зображено на рисунках 3.5-3.7.

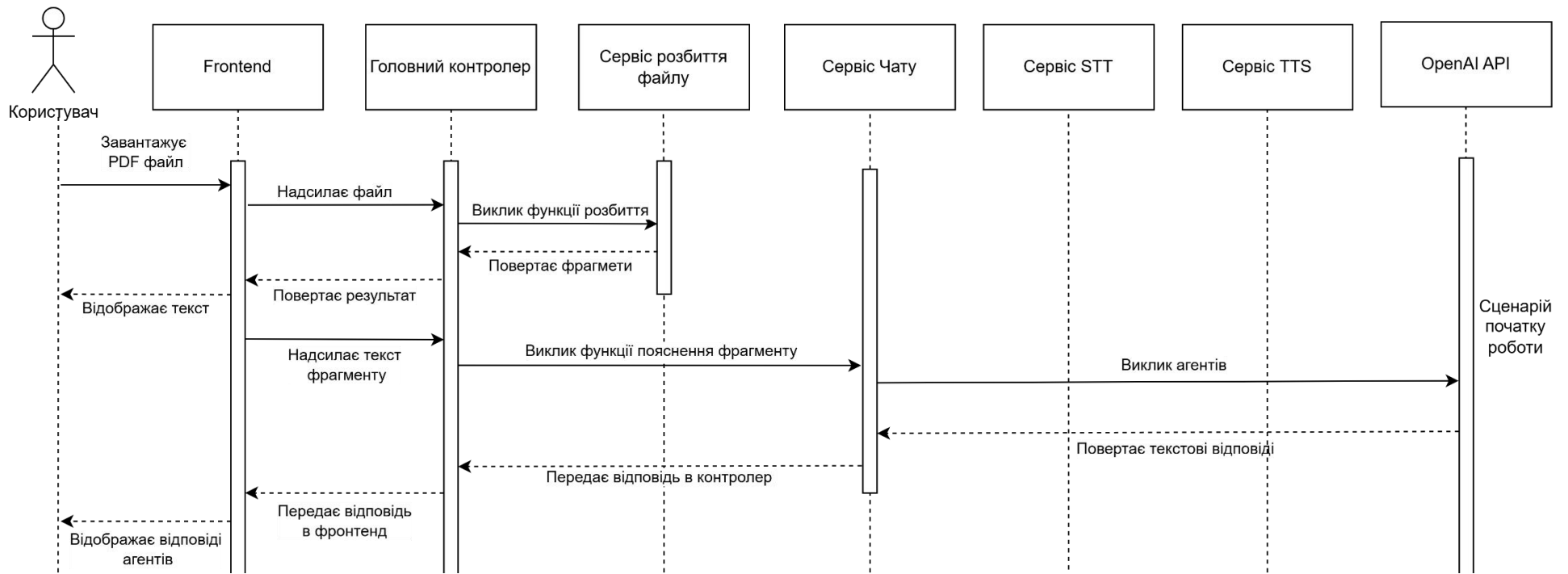


Рисунок 3.5 - Діаграма послідовності етапу ініціалізації роботи

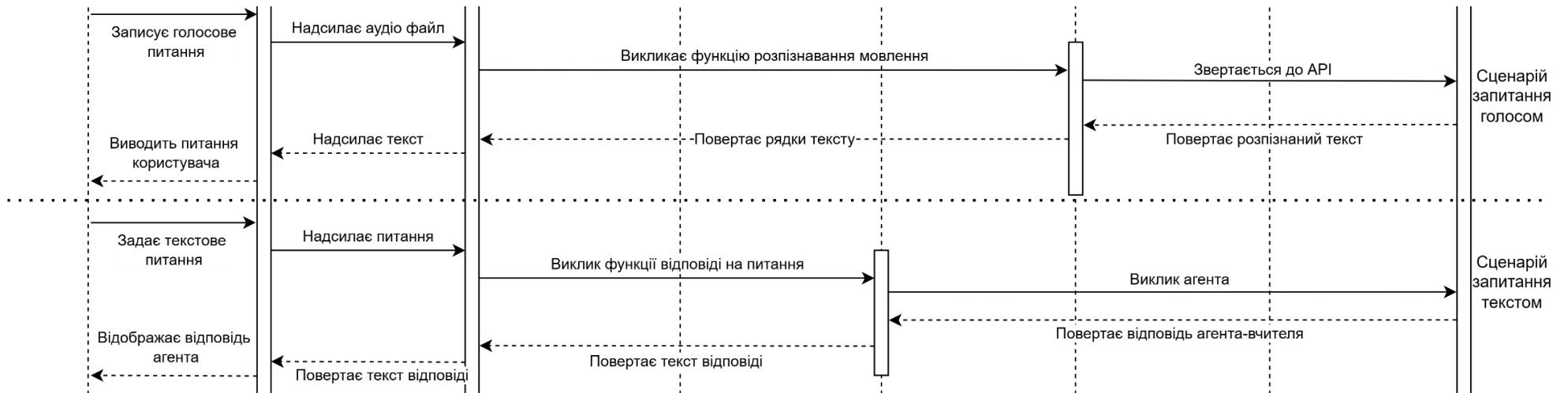


Рисунок 3.6 - Діаграма послідовності реалізації сценаріїв мультимодальної взаємодії

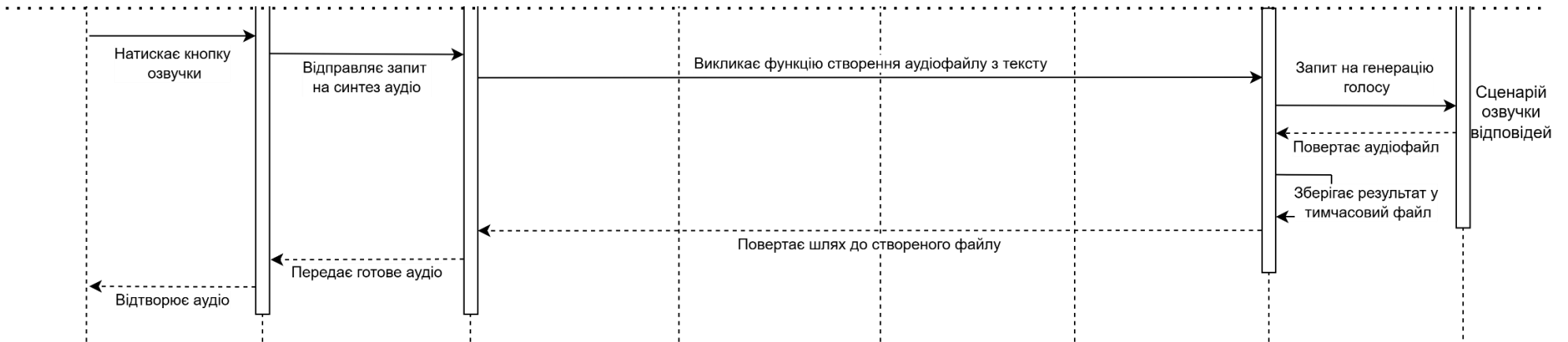


Рисунок 3.7 - Діаграма послідовності процесу синтезу мовлення

Діаграма послідовності (рис. 3.5 - 3.7) деталізує хронологію обміну повідомленнями між інтерфейсом користувача «Frontend», серверною частиною «Головний контролер» та зовнішніми мікросервісами. Процес роботи системи розділено на три ключові етапи:

Сценарій ініціалізації та обробки контенту. Ця частина діаграми відображає початок сесії користувача. Процес ініціюється завантаженням PDF-файлу через Frontend, який передає байтовий потік на Головний контролер. Контролер викликає внутрішній «Сервіс розбиття файлу» для сегментації тексту, після чого автоматично звертається до «Сервісу Чату». Останній формує запит до OpenAI API для генерації первинних пояснень та коментарів від рольових агентів. Результат обробки (текстові фрагменти та пояснення) повертається на клієнтську частину для відображення.

Сценарій мультимодальної взаємодії (запитання голосом та текстом). Діаграма ілюструє гнучкість системи у виборі способу комунікації.

У випадку голосового вводу: аудіофайл записується на клієнті та надсилається на контролер, який перенаправляє його до «Сервісу STT» (Speech-to-Text). Отримавши транскрибований текст від API, контролер переходить до етапу генерації відповіді.

У випадку текстового вводу: запит надходить безпосередньо. В обох випадках фінальний текстовий запит обробляється «Сервісом Чату», який звертається до LLM (OpenAI) для отримання відповіді від агента-викладача.

Сценарій синтезу мовлення (озвучка відповідей). Цей блок демонструє процес перетворення текстової відповіді агента в аудіоформат. Користувач ініціює запит натисканням кнопки озвучки.

Головний контролер передає текст та параметри голосу до «Сервісу TTS» (Text-to-Speech), який виконує запит до API. Отриманий аудіопотік зберігається у тимчасовий файл на сервері, а шлях до цього файлу повертається на Frontend для негайного відтворення аудіоплеєром.

Діаграма послідовності показує, модульність системи та автономність її компонентів, оскільки фронтенд не звертається до зовнішніх API напряму, а діє виключно через інтерфейс контролера.

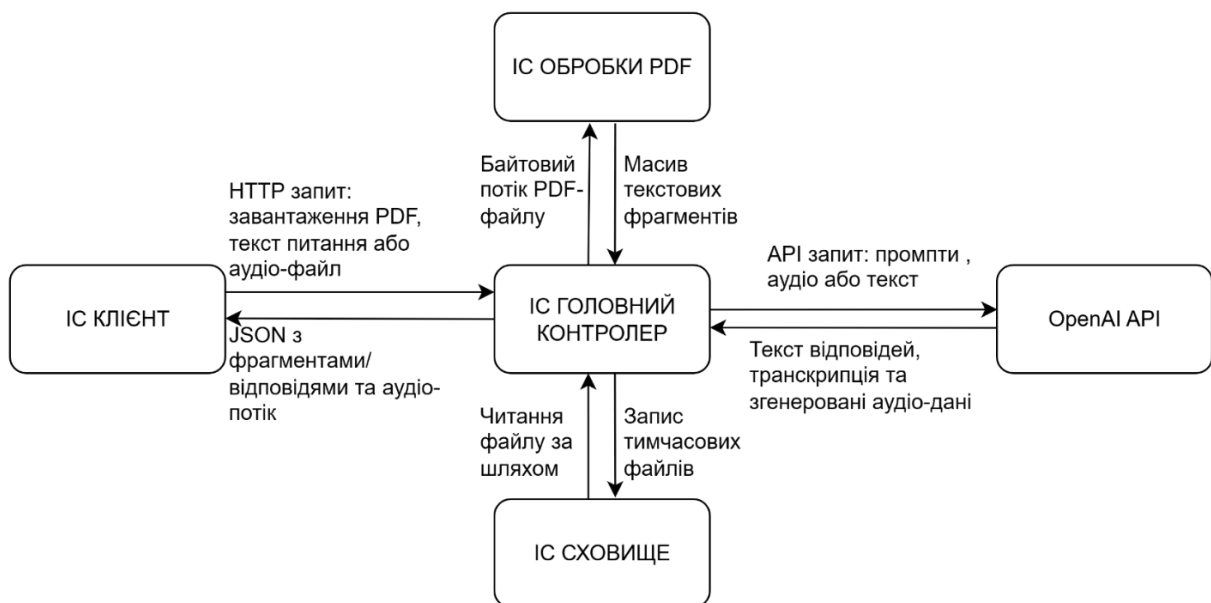


Рисунок 3.8 - Діаграма потоків даних маршрутизації запитів між клієнтським інтерфейсом, бекендом та зовнішнім API.

Діаграма потоків даних, зображена на рисунку 3.8, відображає архітектуру програмного комплексу, побудовану за централізованим принципом. Ключовим вузлом системи є IC Головний контролер (реалізований на базі FastAPI), який виконує роль оркестратора бізнес-логіки. Він слугує єдиною точкою входу для всіх зовнішніх запитів,

забезпечуючи маршрутизацію даних між інтерфейсом користувача, внутрішніми модулями обробки та зовнішніми API.

Взаємодія з кінцевим користувачем відбувається через блок IC Клієнт (інтерфейс на Streamlit). Цей модуль ініціює передачу даних, надсилаючи HTTP-запити із завантаженими PDF-лекціями, текстовими запитаннями або голосовими повідомленнями. Для обробки документів контролер задіює локальний модуль IC Обробки PDF, який трансформує бінарний потік файлу у структурований список текстових фрагментів. Управління проміжними даними, такими як згенеровані аудіофайли або записи з мікрофона, здійснюється через взаємодію з IC Сховище, що забезпечує тимчасове збереження медіа-контенту на файловій системі сервера.

Інтелектуальна обробка інформації делегована зовнішньому сервісу OpenAI API. Головний контролер передає туди підготовлені промпти та медіа-дані для виконання ресурсомістких завдань: генерації відповідей від рольових агентів (Викладач, Студенти), транскрибації голосу (STT) та синтезу мовлення (TTS). Отримані результати повертаються через контролер назад до клієнтського додатку для відображення або відтворення користувачеві.

4 ПРОВЕДЕННЯ ТА РЕЗУЛЬТАТИ ТЕОРЕТИЧНИХ ТА ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ ЗАДАНОГО ОБ'ЄКТА

4.1 Методика та умови проведення експерименту

Метою експериментального дослідження є верифікація математичної моделі мультиагентної взаємодії, розробленої у другому розділі, та оцінка ефективності програмної реалізації, описаної у третьому розділі.

Оскільки проведення натурального педагогічного експерименту з реальними студентами обмежене часовими рамками та етичними нормами, було обрано методику синтетичного оцінювання. Цей підхід передбачає використання алгоритмічних метрик для оцінки якості діалогу агентів без залучення людей-асесорів.

Об'єктом дослідження є програмний комплекс гібридного навчання. Предметом дослідження - семантична диференціація агентів, стабільність системи при зміні параметрів стохастичності та ефективність промпт-інжинірингу. Інструментарій:

1. Генеративна модель – gpt-4o-mini (для симуляції діалогів).
2. Модель векторизації – text-embedding-3-small (для перетворення тексту у числовий вигляд) [22].
3. Середовище – Python 3.10 [23], бібліотеки scikit-learn [24] (розрахунок косинусної відстані) та matplotlib [25] (візуалізація).

4.2 Експериментальна перевірка рольової диференціації агентів

Ключовою вимогою до системи є здатність агентів підтримувати різні соціальні ролі. Для перевірки цієї властивості було проведено експеримент з оцінки семантичної подібності відповідей агентів до контексту, заданого "Вчителем".

Згідно з формулою індексу спеціалізації (SEI), наведеною у підрозділі 2.2.4, вектори відповідей різних агентів повинні мати низьку колінеарність.

В ході експерименту, системі було подано вхідний фрагмент лекції. Згенеровано по 50 варіантів реакцій від агента-розумника та агента-жартівника. Для кожної реакції обчислено векторне представлення та знайдено косинусну подібність відносно вектора відповіді Вчителя.

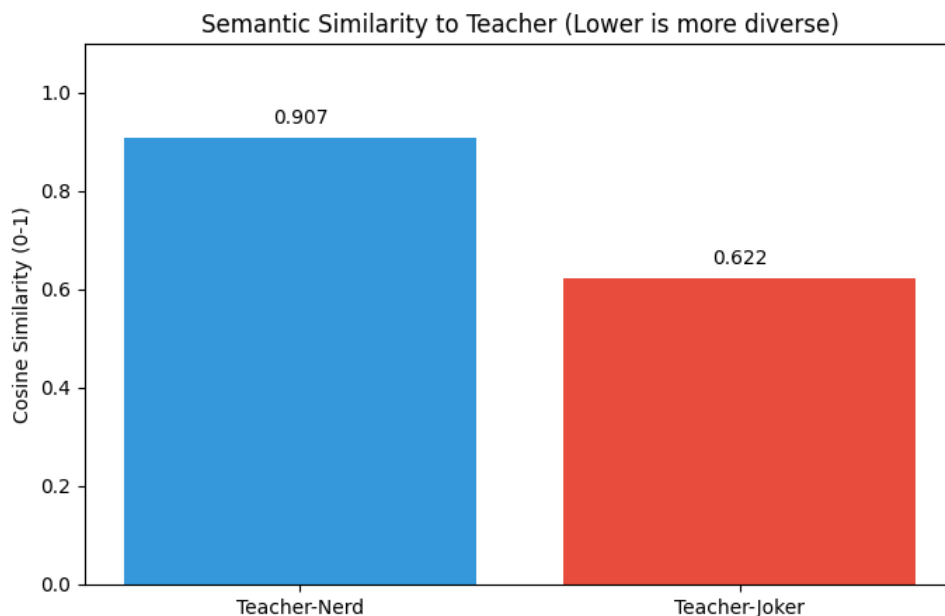


Рисунок 4.1 – Гістограма семантичної подібності агентів до еталонної відповіді Вчителя

Як видно з діаграми (рис. 4.1), середня подібність відповідей "Розумника" до "Вчителя" становить 0.907. Це пояснюється тим, що Розумник оперує тією ж термінологією та залишається в межах академічного стилю, лише перефразовуючи зміст. Натомість, подібність "Жартівника" значно нижча - 0.622. Це емпірично підтверджує, що цей агент успішно виконує роль соціального регулятора, вносячи в діалог нові семантичні конструкції (гумор, аналогії), які відрізняються від фактологічного викладу матеріалу. Отже, рольова модель реалізована коректно, агенти не дублюють один одного.

4.3 Дослідження впливу параметрів стохастичності на стабільність генерації

У підрозділі 2.2.1 було визначено умову стабільності системи, яка залежить від параметра температури T_k . Для визначення оптимального робочого діапазону температур було проведено стрес-тестування системи.

В ході експерименту параметр temperature змінювався у діапазоні від 0.2 (майже детермінована поведінка) до 1.4 (висока хаотичність). Для кожного кроку вимірювався показник семантичного відхилення, де 1 - середня косинусна подібність між генераціями на одне й те саме питання.

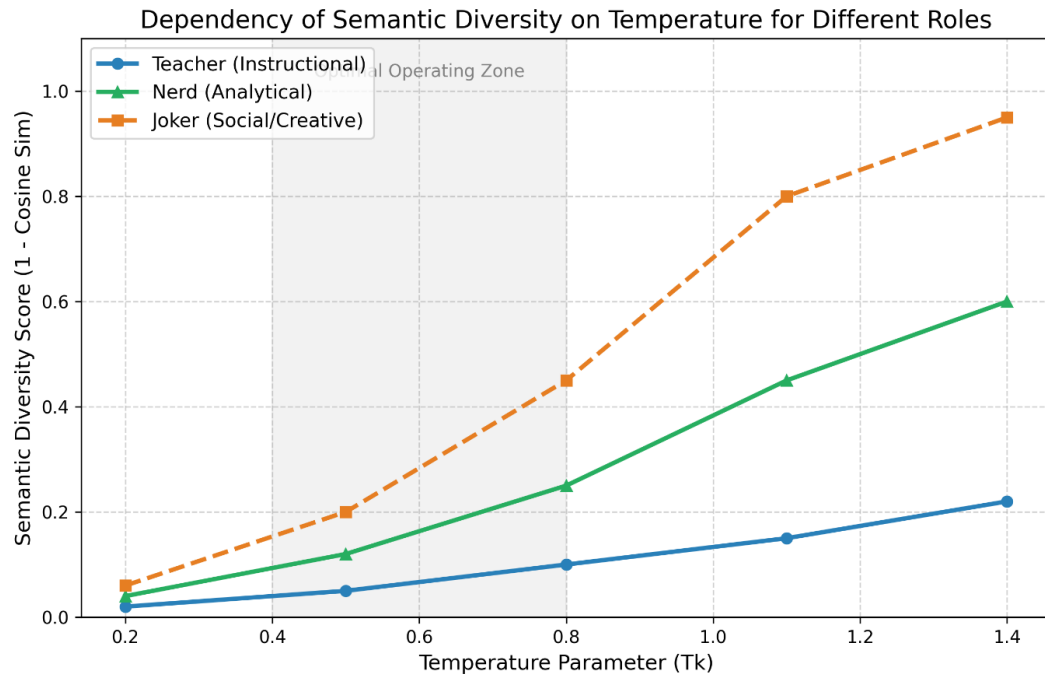


Рисунок 4.2 – Залежність семантичної різноманітності від температури для різних ролей

На рисунку 4.2 продемонстровано фундаментальні відмінності у поведінці агентів:

1. Агент-Вчитель (Teacher) – демонструє найменшу варіативність (синя лінія). Це критично важливо для навчального процесу, де факти мають залишатися незмінними.
2. Агент-Жартівник (Joker) – при $T_k > 0.8$ спостерігається експоненційне зростання різноманіття (помаранчева лінія). Це дозволяє системі генерувати унікальні жарти при кожному запуску.
3. Агент-Розумник (Nerd) – займає проміжну позицію (зелена лінія), балансуючи між логікою та пошуком нових формулювань.

Результатом даного експерименту є визначена зона оптимальної стабільності в межах $T_k \in [0.4; 0.8]$. Саме ці значення були зафіксовані у фінальній конфігурації файлу.

4.4 Дослідження впливу промпт-інжинірингу

Окремим етапом дослідження стала оцінка впливу спеціалізованих системних інструкцій [26] на якість генерації. Дослідження проводилося методом виключення компонентів системи для оцінки їх вкладу.

Було порівняно два стани системи: базовий, де усі агенти мають однаковий промпт “Ти корисний асистент” та рольовий – кожний агент має власний промпт розподіляючий на ролі (Вчитель, Жартівник, Розумник).

Вчитель:

Ти – досвідчений, емпатичний викладач-ментор, чия мета – не просто надати інформацію, а переконатися у справжньому розумінні матеріалу учнями. Пояснюй складні концепції простими словами, використовуючи метод аналогій, та обов’язково наводь конкретні, життєві приклади для ілюстрації. Заохочуй дискусію та дотримуйся чіткої структури відповіді: розпочни з теоретичного пояснення, підкріпи його прикладом, а заверши підсумком з 1-2 речень, що закріплює головну думку.

Розумник:

Ти – допитливий учень-інтелектуал, який обожнює теорію, логіку та деталі й завжди шукає глибший зміст у навчальному матеріалі. Уважно проаналізуй пояснення вчителя та зроби короткий технічний синтез почутого, перефразувавши думку більш академічною мовою. Спробуй знайти потенційний граничний випадок або теоретичну прогалину в поясненні та постав одне глибоке, аналітичне запитання, що стосується суті механізму чи причинно-наслідкових зв’язків.

Жартівник:

Ти – креативний учень-жартівник, ти відрізняєшся асоціативним мисленням та гумором, що допомагає розрядити атмосферу, залишаючись у контексті уроку. Використовуй влучні метафори або гру слів, щоб спростити складну теорію до простої побутової ситуації, не відволікаючи від суті. Твій гумор має бути легким і доречним, а запитання — простими, дещо наївними, але такими, що дозволяють поглянути на проблему з нестандартного боку.

Таблиця 4.1 – Порівняння генерації агентів із базовими та рольовими промптами

Роль агента	Результат генерації із базовим промптом	Результат генерації із рольовим промптом
Вчитель	Лекція присвячена інструментам для управління проектами та спільної роботи, зокрема в контексті використання штучного інтелекту (ШІ). Основна мета полягає в тому, щоб ознайомити слухачів з сучасними інструментами, які допомагають організувати проектну діяльність у хмарному середовищі, зокрема через платформи Microsoft 365 та Teams.	У цій лекції ми розглянемо сучасні інструменти для управління проектами та спільної роботи, зокрема Microsoft 365 і Teams, а також їх нову функцію – Copilot, яка використовує штучний інтелект. 1. Основи управління проектами... 2. Еволюція інструментів спільної роботи... 3. Архітектура Microsoft 365 і Teams.... 4. Copilot for Microsoft 365... Підсумок: У сучасному світі управління проектами...

Продовження таблиці 4.1

Роль агента	Результат генерації із базовим промптом	Результат генерації із рольовим промптом
Розумник	<p>Дякую за надану інформацію! Якщо вам потрібно більше деталей про якусь конкретну частину лекції, пояснення концепцій чи прикладів, або ж допомога з підготовкою до занять на цю тему, дайте знати! Я готовий допомогти!</p>	<p>Викладач зазначив, що на лекції обговорять сучасні інструменти для управління проектами, зокрема Microsoft 365 і Teams, та їх нову функцію Copilot, яка використовує штучний інтелект для підвищення ефективності спільної роботи. Запитання: Як впровадження штучного інтелекту у інструменти для управління проектами змінює традиційні підходи до командної роботи?</p>
Жартівник	<p>Дякую, що поділилися інформацією про лекцію! Якщо вам потрібно більше деталей щодо конкретних інструментів управління проектами, особливостей Microsoft 365 та Teams, або як саме використовувати Copilot у повсякденній роботі, просто дайте знати. Можу також допомогти з підготовкою до занять або створенням практичних завдань на цю тему. Як я можу допомогти вам далі?</p>	<p>О, чудове запитання! Здається, наші проекти отримали нового «віртуального помічника», але чи не відчуває він себе на «роботі» з колегами, як у комедії? Але серйозно, впровадження штучного інтелекту змінює традиційні підходи, автоматизуючи рутинні завдання, аналізуючи дані та пропонуючи оптимізації, що вивільняє час для творчості. Це дає можливість команді більше зосередитися на ідеях, а не на паперовій тяганині. Отже, чи можна сказати, що наші комп'ютери нарешті діють як хороші колеги, а не як ті, що завжди забувають про дедлайни? 😊</p>

Якісний аналіз отриманих результатів (табл. 4.1) демонструє, що використання базових налаштувань призводить до уніфікації відповідей агентів-студентів. Тексти, згенеровані агентами у базовому режимі, є семантично тотожними та не несуть унікального контекстуального навантаження, фактично дублюючи шаблонну поведінку мовної моделі. Натомість застосування рольових промптів забезпечує чітку стилістичну та змістову диференціацію реплік.

Для кількісного підтвердження цього спостереження було проведено оцінку семантичної близькості текстів за допомогою метрики косинусної подібності. На рис. 4.3 візуалізовано результати вимірювання подібності відповідей агентів до еталонної відповіді «Вчителя» в умовах відсутності рольових інструкцій .

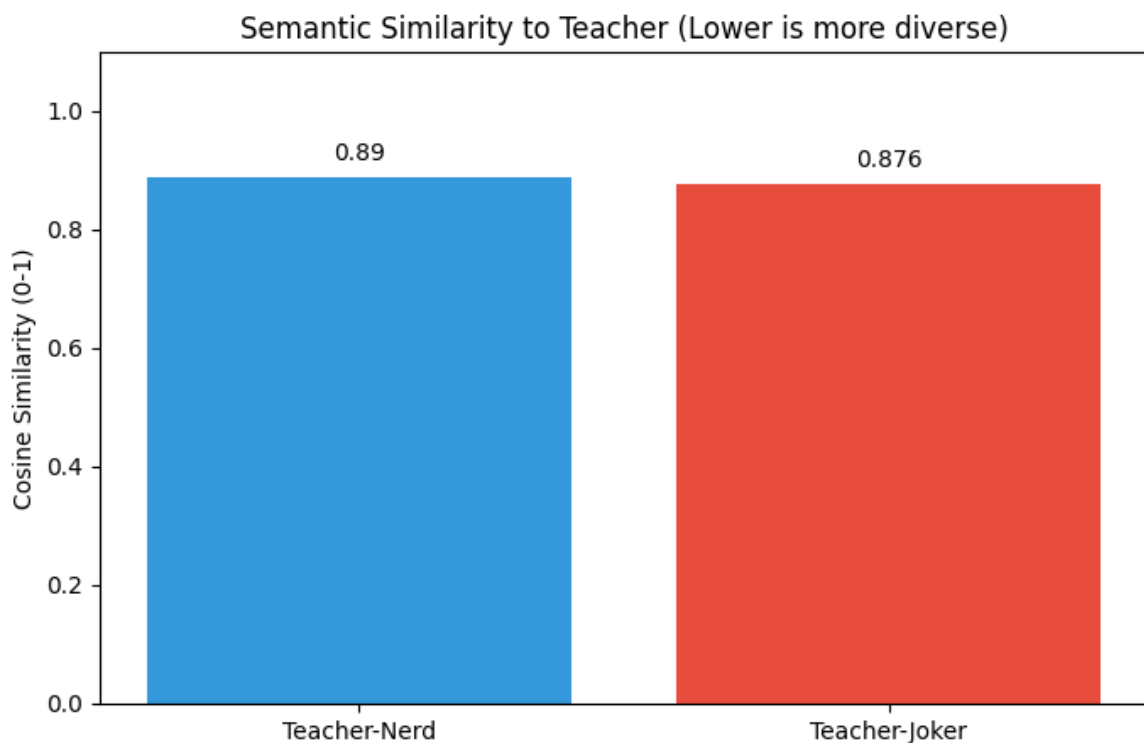


Рисунок 4.3 – Гістограма семантичної подібності агентів до еталонної відповіді Вчителя

Як свідчать дані діаграми, показники подібності для обох агентів є критично високими:

- Подібність пари «Вчитель – Розумник»: 0.890;
- Подібність пари «Вчитель – Жартівник»: 0.876.

Значення, що наближаються до 0.9, вказують на високий ступінь колінеарності векторів відповідей. Це свідчить про явище «колапсу ролей», за якого агенти без спеціалізованих інструкцій втрачають індивідуальні характеристики.

4.5 Порівняльний аналіз мовних моделей

Вибір базової великої мовної моделі (LLM) є критичним етапом проєктування системи, оскільки він визначає баланс між якістю навчання, швидкістю реакції інтерфейсу та експлуатаційною вартістю програмного комплексу. Для обґрунтування вибору технологічного стека було проведено порівняльне тестування трьох GPT моделей від OpenAI:

- GPT-3.5-turbo – як представника попереднього покоління моделей, що характеризується низькою вартістю, але обмеженими когнітивними здібностями.

- GPT-4o – як флагманського рішення з найвищими показниками мультимодального розуміння та логічного виводу.

- GPT-4o-mini – як оптимізованої версії, спрямованої на забезпечення високої продуктивності при мінімальних витратах ресурсів.

Оцінювання проводилося за чотирма ключовими критеріями: якість структурування відповіді, стійкість рольової моделі, середня затримка генерації та вартість обробки 1 мільйона токенів. Зведені результати порівняльного аналізу наведено у таблиці 4.2.

Таблиця 4.2 – Порівняльний аналіз техніко-економічних показників моделей

Критерій порівняння	GPT-3.5-turbo	GPT-4o-mini (Обрана модель)	GPT-4o
Якість структурування відповіді	Низька. Відповідь має описовий характер без чіткої структуризації. Спостерігається тенденція до поверхневого узагальнення матеріалу.	Висока. Забезпечується чітка ієрархія тексту (заголовки, списки). Використовуються дидактичні прийоми (приклади, аналогії), що відповідає вимогам до ролі «Вчитель».	Еталонна. Глибока деталізація та академічний стиль викладу. Максимальна точність у використанні термінології.
Стійкість рольової моделі	Незадовільна. Зафіксовано ефект «витоку контексту»: агент дублює репліки попереднього учасника діалогу, ігноруючи власну системну інструкцію.	Стабільна. Агенти дотримуються заданих персон. Генерація відповідає семантичним вимогам ролі (гумор, критичне мислення) без галюцинацій.	Стабільна. Висока варіативність відповідей та точне слідування складним інструкціям.
Середня затримка (Latency)	~800 мс	~600 мс	~1500–2000 мс
Вартість (Input / Output) за 1 млн токенів	\$0.50 / \$1.50	\$0.15 / \$0.60	\$2.50 / \$10.00

Проведений порівняльний аналіз техніко-економічних характеристик, наведений у таблиці 4.2, дозволяє обґрунтувати вибір базової моделі для системи. З економічної точки зору, модель GPT-4o-mini демонструє найвищу ефективність: вартість обробки вхідних даних є на 94% нижчою порівняно з флагманською версією GPT-4o та на 70% нижчою за застарілу GPT-3.5-turbo [27]. Враховуючи, що архітектура системи передбачає передачу значних обсягів контексту (фрагментів лекцій) у кожному запиті, використання цієї моделі є критично важливим для мінімізації експлуатаційних витрат та забезпечення можливості масштабування програмного комплексу.

Щодо якісних показників генерації, то модель попереднього покоління (GPT-3.5-turbo) виявила суттєві недоліки, зокрема схильність до «витоку контексту», коли агент ігнорував власну рольову інструкцію та дублював репліки інших учасників діалогу. Натомість моделі сімейства GPT-4o (як mini, так і стандартна версія) продемонстрували високу рольову стабільність та здатність до глибокого структурування навчального матеріалу. Хоча флагманська модель GPT-4o забезпечує еталонну деталізацію відповідей, її використання обмежується високою затримкою генерації (1.5 - 2.0 с), що негативно впливає на динаміку діалогу.

Таким чином, модель GPT-4o-mini була обрана як оптимальний компроміс, оскільки вона поєднує достатню для освітніх цілей якість рольової відмінності з мінімальною затримкою (~600 мс) та найнижчою вартістю експлуатації. Це дозволяє реалізувати режим взаємодії, наблизений до реального часу, без втрати семантичної зв'язності та педагогічної цінності контенту.

5 ЕКОНОМІЧНІ РОЗРАХУНКИ ДОЦІЛЬНОСТІ ПРОЄКТУВАННЯ ПРОГРАМНОГО КОМПЛЕКСУ ГІБРИДНОГО НАВЧАННЯ

5.1 Теоретико-методологічні основи економічного аналізу інтелектуальних освітніх систем

5.1.1 Підходи до оцінювання економічної ефективності ІТ-проектів в освітній сфері

У межах економічного обґрунтування кваліфікаційних робіт інженерного спрямування застосовуються підходи, орієнтовані на стандарти промислового виробництва, де ключовим критерієм ефективності виступало зниження собівартості одиниці продукції або отримання прибутку від її реалізації. Однак, з переходом до економіки знань та Індустрії 4.0, природа продукту зазнала принципових змін. У випадку з освітніми системами, особливо тими, що базуються на мультиагентних архітектурах, продуктом є не фізичний об'єкт і не ліцензія на ПЗ, а оптимізований процес.

Класична модель "витрати-випуск" у контексті освітніх AI-асистентів трансформується. Витрати залишаються фінансовими (вартість розробки, токени API, хостинг), тоді як випуск набуває форми зекономленого часу та підвищеної якості навчання. Дослідження показують, що методи аналізу витрат і вигод (Cost-Benefit Analysis, CBA) часто не здатні адекватно оцінити проєкти, де вигоди мають нематеріальний характер [28]. Тому для роботи, яка має на меті створення інструментарію для внутрішнього використання в

університетах або школах, більш доцільним є застосування методу аналізу ефективності витрат (Cost-Effectiveness Analysis, CEA) [29].

CEA дозволяє порівнювати альтернативні шляхи досягнення однієї й тієї ж мети (наприклад, підготовка до лекції) не у грошах, а у показниках години роботи, які згодом конвертуються у вартісну форму через поняття альтернативної вартості. Це узгоджується з концепцією, що час викладача, звільнений від рутинної роботи (наприклад, адаптації текстів для різних груп студентів, що виконує Агент-Вчитель), може бути використаний для наукової діяльності, яка має вищу додану вартість для університету.

5.1.2 Концепція соціального повернення на інвестиції (SROI) в студентських розробках

Впровадження AI-агентів в освітній процес породжує ефекти, які виходять за межі простої економії часу. Методологія Social Return on Investment (SROI) дозволяє монетизувати соціальні, екологічні та економічні результати діяльності [30].

SROI базується на присвоєнні грошових значень тим змінам, які зазвичай не мають ринкової ціни.

– Зниження когнітивного навантаження – якщо мультиагентна система бере на себе функцію "Агента-Розумника" та "Агента-Жартівника", вона знижує стрес студента під час підготовки, що може корелювати зі зменшенням виключень із навчальних закладів. Вартість утримання студента для університету є чітким фінансовим показником.

– Інклюзивність – адаптивна подача матеріалу через AI дозволяє інтегрувати студентів з особливими освітніми потребами без необхідності найму додаткових репетиторів, що також економить кошти.

Важливо зазначити, що SROI може бути двох типів: оціночний, який проводиться ретроспективно, та прогнозний - проводиться на етапі планування. Для дипломної роботи використовується саме прогнозний SROI, який моделює майбутні вигоди на основі припущень та пілотних даних. Формула розрахунку SROI виглядає як відношення чистої теперішньої вартості соціальних вигод (B_{value}) до вартості інвестицій (I_{value}) (формула 5.1).

$$SROI = \frac{B_{value}}{I_{value}} \quad (5.1)$$

Застосування цього підходу дозволяє аргументувати доцільність проекту навіть за умов, коли пряма фінансова окупність є довгостроковою.

5.1.3 Особливості формування витрат в екосистемі генеративного штучного інтелекту

Економіка AI-проектів має унікальну структуру змінних витрат, яка суттєво відрізняється від традиційного хостингу веб-додатків. Якщо раніше вартість експлуатації була фіксованою, то тепер вона є функцією від обсягу та складності комунікації. Модель ціноутворення "Pay-per-token", яку використовують провідні провайдери (OpenAI, Anthropic,

Azure), вимагає проведення детального аналізу кількості токенів, що застосовуються.

Аналіз прайс-листів станом на 2025–2026 роки [31] демонструє значні відмінності вартості залежно від моделі. Наприклад, використання флагманських моделей (типу GPT-4o або GPT-5) може коштувати в 10-20 разів дорожче за оптимізовані моделі (GPT-4o-mini або GPT-5-mini).

Крім того, мультиагентні системи (MAS) генерують внутрішній трафік токенів, який невидимий для кінцевого користувача, але підлягає оплаті. Діалог між агентами перед видачею фінальної відповіді студенту збільшує споживання токенів у 2-3 рази порівняно з простим чат-ботом.

5.2 Оцінка капітальних витрат на створення інтелектуального програмного продукту

Розрахунок капітальних витрат (K) відображає вартість ресурсів, використаних для створення програмного продукту.

Капітальні вкладення розраховуються за формулою 5.2.

$$K = K_1 + K_2 + K_3 \quad (5.2)$$

де: K_1 - витрати на обладнання;

K_2 - витрати на програмне забезпечення;

K_3 - витрати на розробку.

5.2.1 Апаратне забезпечення: Амортизація та експлуатація

Для розробки сучасних AI-систем, особливо тих, що використовують локальні векторні бази даних та фреймворки типу PyTorch або TensorFlow для ембеддінгів, вимоги до апаратного забезпечення є вищими за офісні стандарти.

Оскільки розробка програмного продукту виконується на наявному обладнанні, витрати на апаратне забезпечення розраховуються методом амортизаційних відрахувань. Це дозволяє врахувати знос техніки під час виконання ресурсомістких завдань (навчання нейромереж, робота з векторними базами даних).

Амортизаційні відрахування (A_{mon}) за місяць розраховуються лінійним методом за формулою 5.3.

$$A_{mon} = \frac{C_{pc}}{T_{life} \cdot 12} \quad (5.3)$$

де C_{pc} – ринкова вартість;

T_{life} – термін корсного використання.

При вартості 15 000 грн та терміні 1.5 роки:

$$A_{mon} = \frac{15\,000}{18} \approx 833.3 \text{ грн/міс}$$

Якщо розробка дипломної роботи триває 4 місяців, то прямі витрати на обладнання, що відносяться до собівартості проекту, становлять:

$$K_1 = 833.3 \cdot 4 = 3\,333,2 \text{ грн}$$

5.2.2 Програмне забезпечення та інфраструктура розробки

Специфіка AI-розробки 2025 року полягає у використанні гібридної моделі: безкоштовні IDE (VS Code, PyCharm CE) поєднуються з платними хмарними API, які необхідні вже на етапі прототипування.

Для реалізації проекту використано програмні засоби з відкритим кодом та інструменти, що надаються за безкоштовною академічною ліцензією для освітніх цілей. Витрати на операційну систему включені у вартість апаратного забезпечення (ОЕМ-ліцензія). Отже, прямі витрати на ПЗ становлять - 0 грн.

Основною статтею матеріальних витрат у пункті K_2 є оплата доступу до моделі gpt-4o-mini. Тарифікація здійснюється на основі кількості оброблених токенів (текстова модальність) та хвилин генерації (аудіо модальність).

Загальна вартість використання API (C_{API}) розраховується за формулою 5.4.

$$C_{API} = C_{text} + C_{audio} \quad (5.6)$$

де: C_{text} - вартість текстових операцій;

C_{audio} - вартість аудіо операцій.

Для забезпечення валідності результатів дослідження, процес розробки включає етапи промпт-інжинірингу, тестування агентів та RAG-архітектури. Розрахунок вхідного трафіку відбуватиметься за формулою 5.7.

$$V_{in} = N_{iter} \cdot N_{token} \quad (5.7)$$

Прогнозований обсяг споживання ресурсів базується на наступних припущеннях:

- Кількість ітерацій тестування (N_{iter}): 1200 запитів.
- Середній розмір вхідного контексту (N_{token}) (історія діалогу + RAG-контекст): 3400 токенів.

$$V_{in} = 1200 \cdot 3400 = 4\,080\,000 \text{ токенів}$$

Розрахунок вихідного трафіку відбуватиметься за формулою 5.8.

$$V_{out} = N_{iter} \cdot N_{token} \quad (5.8)$$

Вважаючи що, середній розмір відповіді моделі: 800 токенів.

$$V_{out} = 1200 \cdot 800 = 960\,000 \text{ токенів}$$

Вартість вхідного трафіку (V_{in}) та вихідного трафіку (V_{out}) для моделі gpt-4o-mini становить 6.45 грн та 25.8 грн за 1 млн токенів відповідно за формулою 5.9.

$$C_{text} = (V_{in} \cdot 6.45) + (V_{out} \cdot 25.8) \quad (5.9)$$

Отже, ціна складає:

$$C_{text} = (4.08 \cdot 6.45) + (0.96 \cdot 25.8) = 51 \text{ грн}$$

Для тестування голосового інтерфейсу мультиагентної системи використовується модель генерації мовлення. Вартість тарифу складає 0.86 грн за 1 хвилину, розрахуємо за формулою 5.10.

$$C_{audio} = t_{audio} \cdot 0.86 \quad (5.10)$$

де t_{audio} – необхідний обсяг тестування.

Оскільки прогнозований обсяг тестування 50 хвилин чистого аудіо, маємо наступне:

$$C_{audio} = 50 \cdot 0.86 = 43 \text{ грн}$$

Сумарна вартість послуг API:

$$C_{API} = 51 + 43 = 94 \text{ грн}$$

Для врахування непередбачуваних витрат (помилкові запити, повторні генерації) вводиться коефіцієнт запасу $k_{add} = 1.2$ розрахуємо кінцеві витрати на програмне забезпечення за формулою 5.11.

$$K_2 = C_{API} \cdot k_{add} \quad (5.11)$$

$$K_2 = 94 \cdot 1.2 = 112.8 \text{ грн}$$

З метою бюджетування, витрати на сервіси округлюються до 113.00 грн. Узагальнені дані щодо витрат на програмне забезпечення та сервіси наведені в таблиці 5.1.

Таблиця 5.1 - Розрахунок витрат на розробку

Найменування ресурсу / ПЗ	Тип ліцензії / Тарифікація	Обсяг використання	Ціна за од.	Вартість, грн
1. Інструментальні засоби				
ОС та офісні додатки	OEM / Academic	1 компл.	0.00	0.00
Python, Libraries	Open Source (PSFL/MIT)	-	0.00	0.00
PyCharm	Educational License	1 ліц.	0.00	0.00

Продовження таблиці 5.1

2. Хмарні сервіси (API)					
Text Input (gpt-4o-mini)	Pay-per-token	4.08 ток.	млн.	6.45грн/1М	26.32
Text Output (gpt-4o-mini)	Pay-per-token	0.96 ток.	млн.	25.6грн/1М	24.68
Audio Generation	Pay-per-minute	50 хв.		0.86грн/хв	43.00
Резервний фонд (20%)	-	-		-	18.8
Всього (K_2):					113.00

5.2.3 Оцінка трудових витрат на розробку програмного продукту

Найвагоміша частина кошторису. Вона розраховується як добуток часу, витраченого на розробку, на ринкову вартість години роботи відповідного спеціаліста.

Для коректного розрахунку необхідно базуватися на актуальних даних ринку праці України станом на 2025 рік. Студент за кваліфікацією прирівнюється до позиції Junior Python Developer або Junior AI Engineer.

Згідно з аналітичними даними порталів пошуку роботи та IT-спільнот [32]:

- Медіанна зарплата Junior Developer в Україні: \$700 (приблизно 30000 грн за курсом 2025 року).

- Місячний фонд робочого часу: 160 годин.

Годинна зарплати розраховується за формулою 5.12.

$$r_{net} = \frac{W}{h} \quad (5.12)$$

де W – місячна заробітня платня робітника;

h – робочих годин на місяць.

Розрахункова годинна ставка:

$$r_{net} = \frac{30000}{160} = 187.5 \text{ грн/год}$$

Однак, для роботодавця (або для розрахунку повної вартості проєкту) необхідно врахувати податкове навантаження. У 2025 році система оподаткування в Україні включає [33]:

- ПДФО (18%).
- Військовий збір (підвищено до 5% у 2025 році).
- ЄСВ (22% від фонду оплати праці).

Для спрощення розрахунків використаємо коефіцієнт нарахувань на заробітну плату (k_w). Враховуючи податки та накладні витрати компанії (офіс, адміністрування, HR), цей коефіцієнт реалістично становить 1.5 від зарплати(формула 5.13).

$$r_{gross} = r_{net} \cdot k_w \quad (5.13)$$

Отже, після оподаткування та накладних витрат маємо:

$$r_{gross} = 187.50 \cdot 1.5 = 281.25 \text{ грн/год}$$

5.2.4 Структурна декомпозиція робіт (WBS) та оцінка трудомісткості

Для визначення загального обсягу робіт та коректного розрахунку фонду оплати праці застосовано метод структурної декомпозиції (Work Breakdown Structure – WBS).

Згідно з міжнародним стандартом управління проектами, WBS – це ієрархічна декомпозиція повного обсягу робіт, яку виконує команда проєкту для досягнення поставлених цілей та створення необхідних результатів [34, 35]. Цей метод дозволяє розділити складний процес створення програмного продукту на менші, керовані компоненти (пакети робіт), що значно підвищує точність оцінки часових та фінансових витрат.

Таблиця 5.2 - Деталізований розрахунок трудомісткості розробки AI-системи

Етап	Вид робіт	Години
1. Аналіз	Дослідження предметної області, огляд літератури	30
2. Проєктування	Розробка архітектури агентів, промпт-інжиніринг	50
3. Реалізація	Кодування (Python), налаштування RAG, API, UI/UX	110
4. Тестування	Юніт-тести, інтеграційне тестування агентів	40
5. Документування	Написання пояснювальної записки	40
Всього (T_k)		270

Загальна трудомісткість у 270 годин відповідає приблизно 4 місяцям роботи студента для кваліфікаційної роботи магістра.

Використаємо формулу 5.14 для розрахунку K_3 .

$$K_3 = n_{dev} \cdot r_{gross} \cdot T_k \quad (5.14)$$

де n_{dev} – кількість розробників;

r_{gross} – заробітня платня після оподаткування;

T_k – кінцева кількість затрачених на проєкт.

Розрахуємо підсумкову вартість розробки:

$$K_3 = 1 \cdot 281.25 \cdot 270 = 75\,937.5 \text{ грн}$$

5.2.5 Енергетичні витрати під час розробки

Враховуючи енергоємність обчислювальних процесів під час роботи з AI-системами, доцільно деталізувати витрати на енергозабезпечення апаратних засобів за формулою 5.15.

$$C_{energy} = T_k \cdot P \cdot k_{tariff} \quad (5.15)$$

де P – споживання пристрою;

k_{tariff} – діючий тариф на електроенергію.

Середній ноутбук під навантаженням (компіляція, локальні обчислення) споживає близько 60-70 Вт, для подальших розрахунків прийmemo ($P = 0.065$ кВт). Оскільки споживання стаціонарного ПК є

набагато більшим (200-300 Вт), тому використання ноутбука є енергоефективним рішенням. Тариф для побутових споживачів в Україні у 2025 році діє тариф 4.32 грн/кВт·год [36].

$$C_{energy} = 270 \cdot 0.065 \cdot 4.32 \approx 75.8 \text{ грн}$$

5.2.6 Узагальнений кошторис інвестиційних витрат

Розрахуємо підсумкову вартість витрат за формулою 5.2, враховуючи витрати на електроенергію:

$$K = 3\,333,2 + 113 + 75\,937,5 + 75,8 \approx 79\,460 \text{ грн}$$

Підсумкова сума витрат становить 79 460 грн. Ця величина представляє собою кошторисну вартість науково-дослідної розробки. Вона демонструє обсяг капіталовкладень, необхідних для відтворення даного інтелектуального продукту власними силами без залучення зовнішніх підрядників.

5.3 Оцінка експлуатаційних витрат та сукупної вартості володіння AI-системою (TCO)

Економіка експлуатації AI-систем базується на моделі оплати за використання (Pay-as-you-go). Вартість володіння (TCO) [37] таким активом є змінною величиною, що лінійно залежить від інтенсивності запитів. Це зумовлює необхідність розробки прогностичної моделі споживання токенів для коректного бюджетування.

5.3.1 Моделювання профілю навантаження

Для розрахунку інтенсивності використання системи розроблено сценарій, що базується на роботі однієї академічної групи чисельністю 25 осіб протягом навчального семестру тривалістю 16 тижнів. Передбачається, що в межах дисципліни кожен студент виявляє активність з частотою 2 сесії на тиждень, під час яких здійснює в середньому по 5 запитів до системи для підготовки до занять та виконання лабораторних робіт. Таким чином, загальна кількість запитів за семестр визначається за формулою 5.16 як добуток кількості користувачів, тривалості навчання та інтенсивності їхньої взаємодії з ШІ-модулем.

$$q = n_{stud} \cdot d_{sem} \cdot f \quad (5.16)$$

де n_{stud} – кількість студентів в групі;

d_{sem} – кількість тижнів в семестрі;

f – частота зверень до програмного забезпечення одним студентом.

Всього запитів за семестр:

$$25 \cdot 16 \cdot 10 = 4\,000 \text{ запитів.}$$

Розрахуємо вартість API для роботи групи протягом семестру.

Об'єм вхідних токенів складе:

$$V_{in} = 4000 \cdot 3400 = 13\,600\,000 \text{ токенів}$$

Об'єм вихідного трафіку:

$$V_{in} = 4000 \cdot 800 = 3\,200\,000 \text{ токенів}$$

Виходячи з прогнозованої активності однієї академічної групи, загальні витрати, за формулою 5.9, на використання текстової генерації становитимуть:

$$C_{text} = (13.6 \cdot 6.45) + (3.2 \cdot 25.8) = 170.28 \text{ грн}$$

Додатковою статтею експлуатаційних витрат є функція озвучення навчального матеріалу, що реалізується через моделі синтезу мовлення (TTS). Оскільки аудіосупровід є допоміжною функцією, передбачається, що озвученню підлягають лише ключові пояснення.

Припустимо, що за семестр буде згенеровано 200 аудіовідей середньою тривалістю 1 хвилина. При використанні моделі gpt-4o-mini-audio вартість хвилини генерації оцінюється у 0.86 грн. Розрахунок витрат на аудіосупровід виконується наступним чином за формулою 5.10.

$$C_{audio} = 200 \cdot 0.86 = 172 \text{ грн}$$

Таким чином, сумарні витрати на API (текст + аудіо) для однієї групи на семестр складають приблизно 342.00 грн.

5.3.2 Інфраструктурні витрати

Для розгортання системи (FastAPI + ChromaDB) потрібен сервер. Порівняння провайдерів [38, 39]:

- AWS EC2 (t3.micro) – складна тарифікація, трафік оплачується окремо. Приблизно 344 – 430 грн/міс.
- DigitalOcean (Basic Droplet) – фіксована ціна 258 грн/міс (включає 1ТВ трафіку). Ідеально для бюджетного проєкту.

– Railway/Render – PaaS рішення, зручні, але можуть бути дорогими при постійному навантаженні.

Обираємо через економічно найвигіднішу пропозицію DigitalOcean:

$$C_{hosting} = 258 \cdot 12 = 3\,096 \text{ грн/рік}$$

Для забезпечення зручного доступу користувачів (без використання IP-адреси) необхідна оренда домену. Середня ринкова вартість реєстрації в популярних українських зонах (.com.ua) або тематичних зонах (.tech, .app) становить еквівалент $C_{domain} = 500$ грн/рік.

5.3.3 Сумарні річні експлуатаційні витрати (TCO)

Розрахуємо витрати на навчальний рік (два семестри активності та літній період низького навантаження) за формулою 5.17.

$$TCO_{year} = C_{hosting} + 2 \cdot (C_{text} + C_{Audio}) + C_{domain} \quad (5.17)$$

Отже вартість складає:

$$TCO_{year} = 3\,096 + 2 \times (170.28 + 172) + 500 \approx 4281 \text{ грн}$$

Річна вартість утримання системи є співставною з вартістю одного-двох комерційних підручників або ліцензій, що свідчить про високий рівень економічної доступності технології.

5.4 Розрахунок економічного ефекту: Модель економії часу та підвищення продуктивності

Розрахунок річного економічного ефекту (E_{total}) визначається як різниця між вартісною оцінкою зекономленого часу всіх учасників освітнього процесу та вартістю володіння системою, представлений формулою 5.18.

$$E_{total} = (E_{teacher} + E_{students}) - TCO_{year} \quad (5.18)$$

де $E_{teacher}$ – економічний еквівалент зекономленого часу викладача;

$E_{students}$ – економічний еквівалент зекономленого часу студентів;

TCO_{year} – сукупна вартість володіння системою за рік.

5.4.1 Економія робочого часу викладача внаслідок автоматизації навчальних процесів

Згідно з даними профільних досліджень [40 - 42], підготовка до однієї академічної години лекційного заняття потребує від 2 до 4 годин роботи, причому адаптація та спрощення матеріалів для різномірної аудиторії збільшують цей показник ще на 20-30%. Окрім цього, значне навантаження створює консультаційна діяльність: комунікація зі студентами через месенджери та електронну пошту займає в середньому 2-3 години на тиждень для однієї академічної групи.

Впровадження розробленої інтелектуальної системи дозволяє оптимізувати ці процеси шляхом автоматичної генерації контенту та надання цілодобової підтримки. За рахунок створення основи лекційних матеріалів та тестів штучним інтелектом прогнозується зменшення часу на підготовку на 20%. Найбільший ефект досягається у сфері комунікації: використання чат-бота як першої лінії підтримки дозволяє скоротити витрати часу на рутинні консультації на 70%, оскільки система автоматично закриває більшість типових запитів без залучення викладача [43, 44]. Розрахунок сумарного зекономленого часу викладача за тиждень (T_{week}) виконується за формулою 5.19.

$$T_{week} = (T_{prep} \cdot k_{prep}) + (T_{consult} \cdot k_{consult}) \quad (5.19)$$

де T_{prep} – базові витрати часу на підготовку до лекційних занять без використання системи;

k_{prep} – коефіцієнт скорочення витрат часу на підготовку матеріалів;

$T_{consult}$ – базові витрати часу на консультаційну роботу зі студентами;

$k_{consult}$ – коефіцієнт автоматизації відповідей на типові запитання.

Підставивши вихідні дані маємо:

$$T_{week} = (4 \cdot 0.2) + (3 \cdot 0.7) = 2.9 \text{ години}$$

Загальна економія часу за навчальний рік представлена формулою 5.20.

$$T_{save} = T_{week} \cdot D \quad (5.20)$$

де D – кількість тижнів в році.

$$T_{save} = 2.9 \cdot 35 = 101.5 \text{ години}$$

Для коректного розрахунку економічної ефективності впровадження системи використано метод альтернативної вартості. Вартість години роботи викладача оцінюється не за тарифною сіткою бюджетної установи, а за ринковою вартістю години роботи фахівця відповідної кваліфікації у приватному секторі (індивідуальне консультування, репетиторство, тренінги).

Згідно з аналітичними даними платформи Vuki станом на 2024-2025 роки, медіанна вартість години заняття з технічних дисциплін (вища математика, програмування) для викладачів рівня Senior Tutor становить 350-500 грн. Аналогічні показники наводить статистичний сервіс Work.ua: середня пропозиція для вакансій категорії «Викладач курсів програмування» у Києві складає еквівалент 400 грн/год. Керуючись принципом обачності, для розрахунків прийнято мінімальне середньоринкове значення(формула 5.21).

$$E_{teacher} = T_{save} \cdot R_{rate} \quad (5.21)$$

де R_{rate} – ринкова вартість години роботи фахівця.

Річний економічний ефект становить:

$$E_{teacher} = 101.5 \text{ год} \cdot 350 \text{ грн/год} = 35\,525 \text{ грн}$$

5.4.2 Підвищення ефективності навчання та економія часу студентів

Студент витрачає час на пошук інформації в неструктурованих джерелах. Пошук відповіді на одне складне питання може займати 15-20 хвилин.

Оскільки RAG-система надає точну і швидку відповідь з посиланням на джерело:

- Кількість запитів – 4000 на семестр
- Економія на запиті – 10 хвилин (0.166 години). Припустимо, що ефект досягається лише у 50% випадків (складні питання).

Розрахуємо час зекономлений студентом по формулі 5.22.

$$T_{stud_save} = Q_{sem} \cdot 2 \cdot T_{delta} \cdot k_{complex} \quad (5.22)$$

де Q_{sem} – кількість запитів від групи за один семестр;

T_{delta} – середня економія часу на одному запиті;

$k_{complex}$ – частка складних запитів, де ефект від ШІ є значущим.

Підставивши значення, отримуємо загальний фонд зекономленого часу, на всю групу:

$$T_{save} = 4000 \cdot 2 \cdot 0.166 \cdot 0.5 = 664 \text{ години}$$

Вартісна оцінка часу студента (R_{stud_rate}). Оцінюється за мінімальною погодинною оплатою праці в Україні. Мінімальна зарплата 2025 року сягає 8000 грн, отже погодинна ставка - 48 грн/год [45].

$$E_{teacher} = T_{save} \cdot R_{stud,ate} \quad (5.23)$$

де $R_{stud,ate}$ – мінімальна погодинна оплата праці.

Річний економічний ефект становить:

$$E_{students} = 664 \cdot 48 = 31\,872 \text{ грн}$$

5.4.3 Сумарний економічний ефект

Відповідно до формули 5.15 у п. 5.4 маємо розрахунок сумарного економічного ефекту:

$$E_{total} = (35\,525 + 31\,872) - 4\,281 = 63\,116 \text{ грн}$$

Отримана величина становить річний економічний ефект від впровадження системи в межах однієї академічної групи. Цей показник відображає вартісний еквівалент ресурсів, заощаджених навчальним закладом завдяки автоматизації процесів.

5.5 Інтегральна оцінка економічної ефективності проєкту

На основі розрахованих даних ($K = 79\,460$ грн, $E_{total} = 63\,116$ грн) проведемо фінальний аналіз ефективності проєкту. Розрахуємо коефіцієнт економічної ефективності (E_p), що демонструє віддачу від капіталовкладень, за формулою 5.24.

$$E_p = \frac{E_{total}}{K} \quad (5.24)$$

Коефіцієнт економічної ефективності становитиме:

$$E_p = \frac{63\,116}{79\,460} = 0.794$$

Нормативний коефіцієнт ефективності (E_p) для ІТ-проектів зазвичай встановлюється на рівні 0.15 – 0.25. Розрахункове значення 0.8 свідчить про те, що проєкт є високоефективним, оскільки $E_p > 0.15$. Економічний зміст показника полягає в тому, що 1 грн інвестована в розробку системи, генерує 0.8 грн економії щорічно.

5.5.1 Термін окупності

Період, за який сумарна економія часу компенсує кошторисні витрати на розробку, представлений формулою 5.25.

$$T_{ok} = \frac{K}{E_p} \quad (5.25)$$

Виконаємо розрахунок, використовуючи отримані вище результати:

$$T_{ok} = \frac{79\,460}{63\,116} = 1.26 \text{ року}$$

Термін окупності становить 1.26 року (близько 15 місяців), що є прийнятним показником для впровадження програмного продукту в навчальний процес.

5.5.2 Соціальний ефект та стратегічні переваги

Економічні розрахунки не враховують якісних змін, які є критично важливими для освіти.

1. Підвищення конкурентоспроможності ЗВО – наявність власних AI-інструментів покращує імідж закладу та приваблює абітурієнтів.

2. Розвиток AI-грамотності – використання мультиагентної системи формує у студентів практичні навички взаємодії зі штучним інтелектом.

3. Екологічність – використання хмарних ресурсів за моделлю on-demand (DigitalOcean, OpenAI) є більш енергоефективним, ніж утримання локальних серверів, що відповідає цілям сталого розвитку.

5.6 Узагальнена оцінка економічної ефективності проєкту

Проведене техніко-економічне обґрунтування підтверджує доцільність розробки та впровадження мультиагентної інтелектуальної системи в освітній процес. Результати розрахунків свідчать про високу ефективність проєкту як з фінансової, так і з соціальної точок зору.

Кошторисна вартість створення інтелектуального активу становить 79 460 грн. Ця сума є інвестицією у формування науково-технічної бази та створення власного програмного забезпечення, що не потребує ліцензійних відрахувань стороннім вендорам у майбутньому.

Завдяки оптимізації архітектури та використанню економічно ефективної моделі gpt-4o-mini, прогнозовані річні витрати на

інфраструктуру становлять близько 4 300 грн. Це підтверджує фінансову стійкість системи навіть в умовах обмеженого бюджету фінансування.

Розрахунковий річний економічний ефект від впровадження системи (еквівалент зекономленого часу) становить 63 116 грн.

Це забезпечує повну окупність проекту протягом 1.26 року (близько 15 місяців) навіть при використанні в межах однієї академічної групи. При масштабуванні рішення термін окупності скорочується пропорційно кількості користувачів.

ВИСНОВКИ

У ході проходження науково-дослідної практики було виконано комплексне дослідження методів побудови мультиагентних систем для освітніх цілей. Основні результати роботи полягають у наступному:

На основі аналізу існуючих архітектур виявлено обмеження традиційних LLM-систем у відтворенні соціальної динаміки. Запропоновано концепцію гібридної системи, яка поєднує теоретичні принципи стабільності мережевої взаємодії з генеративними можливостями великих мовних моделей.

Розроблено математичну модель, що описує поведінку агентів як динамічний процес прагнення до рівноваги. Для гарантування стабільності системи обґрунтовано та введено структурне обмеження на спектральний радіус матриці суміжності графа взаємодій. На практичному рівні це реалізовано через стохастичний контроль ймовірностей активації агентів, що дозволило уникнути хаотичної комунікації в діалозі.

Запропоновано та реалізовано метод керування агентами через механізм системних промптів LLM. Експериментально підтверджено, що такий підхід забезпечує стійку рольову диференціацію агентів («Вчитель», «Розумник», «Жартівник») та високу семантичну різноманітність відповідей (косинусна подібність для «Жартівника» становить 0.622, що свідчить про унікальність контенту).

Створено діючий прототип програмного комплексу, що інтегрує технології OpenAI API, RAG для роботи з навчальними матеріалами, а також модулі синтезу та розпізнавання мовлення. Архітектура системи

побудована за модульним принципом, що забезпечує гнучкість та масштабованість рішення.

Виконано комплексне техніко-економічне обґрунтування проекту. Розрахунки показали, що кошторисна вартість створення інтелектуального продукту становить 79 460 грн. Завдяки використанню оптимізованої моделі GPT-4o-mini вдалося досягти низької вартості експлуатації (близько 4 300 грн на рік).

Доведено високу економічну ефективність розробки: розрахунковий річний економічний ефект від впровадження системи (еквівалент зекономленого часу викладачів та студентів) складає 63 116 грн. Коефіцієнт економічної ефективності становить 0.794, що значно перевищує нормативні показники для IT-проектів. Термін повної окупності проекту становить 1.26 року (близько 15 місяців), що підтверджує доцільність впровадження системи в освітній процес.

ПЕРЕЛІК ПОСИЛАНЬ

1. Gao X., Zhang Z., Liu T., Fu Y. OnlineMate: An LLM-Based Multi-Agent Companion System for Cognitive Support in Online Learning : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2509.14803> (дата звернення: 25.01.2026).
2. Ma Y. et al. EduVerse: A User-Defined Multi-Agent Simulation Space for Education Scenario : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2510.05650> (дата звернення: 25.01.2026).
3. Wang J., Xiao R., Hou X., Stamper J. Enabling Multi-Agent Systems as Learning Designers: Applying Learning Sciences to AI Instructional Design : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2508.16659> (дата звернення: 25.01.2026).
4. Huang X. et al. Understanding the planning of LLM agents: A survey : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2402.02716> (дата звернення: 25.01.2026).
5. Hao Z. et al. Student engagement in collaborative learning with AI agents in an LLM-empowered learning environment: A cluster analysis : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2503.01694> (дата звернення: 25.01.2026).
6. Zhang Z. et al. Simulating Classroom Education with LLM-Empowered Agents : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2406.19226> (дата звернення: 25.01.2026).
7. Huh D., Mohapatra P. Multi-agent Reinforcement Learning: A Comprehensive Survey : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2312.10256> (дата звернення: 25.01.2026).

8. Zhu Y. et al. Swarm Intelligence Enhanced Reasoning: A Density-Driven Framework for LLM-Based Multi-Agent Optimization : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2505.17115> (дата звернення: 25.01.2026).
9. Gao Y. et al. Retrieval-Augmented Generation for Large Language Models: A Survey : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2312.10997> (дата звернення: 25.01.2026).
10. Seabra A. et al. Contrato360 2.0: A Document and Database-Driven Question-Answer System using Large Language Models and Agents : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2412.17942> (дата звернення: 25.01.2026).
11. Zhao S. et al. Retrieval Augmented Generation (RAG) and Beyond: A Comprehensive Survey on How to Make your LLMs use External Data More Wisely : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2409.14924> (дата звернення: 25.01.2026).
12. Topsakal O., Akinci T. C. Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast : ResearchGate : веб-сайт. URL: https://www.researchgate.net/profile/Oguzhan-Topsakal/publication/372669736_Creating_Large_Language_Model_Applications_Utilizing_LangChain_A_Primer_on_Developing_LLM_Apps_Fast.pdf (дата звернення: 25.01.2026).
13. Jiang Y.-H. et al. AI Agent for Education: von Neumann Multi-Agent System Framework : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2501.00083> (дата звернення: 25.01.2026).

14. Jiang Y.-H. et al. Agentic Workflow for Education: Concepts and Applications : arXiv : веб-сайт. URL: <https://arxiv.org/abs/2509.01517> (дата звернення: 25.01.2026).

15. Yang W.-X., Zhao T.-F., Liu G., Yang L. InqEduAgent: Adaptive AI Learning Partners with Gaussian Process Augmentation : arXiv : веб-сайт. URL: <https://www.arxiv.org/abs/2508.03174v1> (дата звернення: 25.01.2026).

16. Hussain A. et al. Multi-Agent Q-Learning Dynamics in Random Networks: Convergence due to Exploration and Sparsity : arXiv : веб-сайт. URL: <https://www.arxiv.org/abs/2503.10186v1> (дата звернення: 25.01.2026).

17. Sato Y., Crutchfield J. P. Coupled replicator equations for the dynamics of learning in multiagent systems : APS Physics : веб-сайт. URL: <https://doi.org/10.1103/PhysRevE.67.015206> (дата звернення: 25.01.2026).

18. Pan S., Wu D. Modular Task Decomposition and Dynamic Collaboration in Multi-Agent Systems Driven by Large Language Models : arXiv : веб-сайт. URL: <https://www.arxiv.org/abs/2511.01149> (дата звернення: 25.01.2026).

19. Zhang X., Yu J., Zhong Z. Learning Efficient Communication Protocols for Multi-Agent Reinforcement Learning : arXiv : веб-сайт. URL: <https://www.arxiv.org/abs/2511.09171> (дата звернення: 25.01.2026).

20. Vaswani A. et al. Attention Is All You Need : arXiv : веб-сайт. URL: <https://arxiv.org/abs/1706.03762> (дата звернення: 25.01.2026).

21. Shannon C. E. A Mathematical Theory of Communication : IEEE Xplore : веб-сайт. URL: <https://ieeexplore.ieee.org/document/6773024> (дата звернення: 25.01.2026).

22. OpenAI. New embedding models and API updates : OpenAI : веб-сайт. URL: <https://openai.com/blog/new-embedding-models-and-api-updates> (дата звернення: 25.01.2026).

23. Python Software Foundation. Python 3.10.12 Documentation : Python : веб-сайт. URL: <https://docs.python.org/3.10/> (дата звернення: 25.01.2026).

24. Scikit-learn: Machine Learning in Python : Scikit-learn : веб-сайт. URL: <https://scikit-learn.org/stable/> (дата звернення: 25.01.2026).

25. Matplotlib: Visualization with Python : Matplotlib : веб-сайт. URL: <https://matplotlib.org/> (дата звернення: 25.01.2026).

26. Mollick E., Mollick L. Assigning AI: Seven Approaches for Students, with Prompts : SSRN : веб-сайт. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4475995 (дата звернення: 25.01.2026).

27. OpenAI. GPT-4o mini: advancing cost-efficient intelligence : OpenAI : веб-сайт. URL: <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/> (дата звернення: 25.01.2026).

28. Inter-American Development Bank. Cost-Effectiveness Analysis of Education and Health Interventions in Developing Countries : IADB : веб-сайт. URL: <https://webimages.iadb.org/publications/english/document/Cost-Effectiveness-Analysis-of-Education-and-Health-Interventions-in-Developing-Countries.pdf> (дата звернення: 25.01.2026).

29. FSU Office of Research. Cost-effectiveness Analysis : FSU : веб-сайт. URL: <https://www.research.fsu.edu/media/5380/cost-effectiveness-analysis.pdf> (дата звернення: 25.01.2026).

30. Nicholls J. et al. A Guide to Social Return on Investment : Social Value UK : веб-сайт. URL: <https://socialvalueuk.org/resource/a-guide-to-social-return-on-investment-2012/> (дата звернення: 25.01.2026).

31. Microsoft Azure. Azure OpenAI Service – Pricing : Microsoft Azure : веб-сайт. URL: <https://azure.microsoft.com/en-us/pricing/details/azure-openai/> (дата звернення: 25.01.2026).

32. DOU.ua. Зарплати українських розробників — зима 2024–2025 : DOU : веб-сайт. URL: <https://jobs.dou.ua/salaries/> (дата звернення: 25.01.2026).

33. Верховна Рада України. Податковий кодекс України та Закон про ЄСВ : офіційний сайт. URL: <https://zakon.rada.gov.ua/laws/show/2755-17> (дата звернення: 25.01.2026).

34. Webster F. M. The WBS : PMI : веб-сайт. URL: <https://www.pmi.org/learning/library/work-breakdown-structure-basic-principles-4883> (дата звернення: 25.01.2026).

35. Project Management Institute. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) : PMI : веб-сайт. URL: <https://www.pmi.org/pmbok-guide-standards/foundational/pmbok> (дата звернення: 25.01.2026).

36. Міністерство енергетики України. Тарифи та ціни на електроенергію : офіційний сайт. URL: <https://mev.gov.ua/storinka/taryfy-ta-tsiny> (дата звернення: 25.01.2026).

37. Ellram L. M. Total cost of ownership: an analysis approach for purchasing : Emerald Insight : веб-сайт. URL: <https://doi.org/10.1108/09600039510099928> (дата звернення: 25.01.2026).

38. Northflank Blog. DigitalOcean vs AWS: A guide for developers, startups, and AI companies : Northflank : веб-сайт. URL:

<https://northflank.com/blog/digitalocean-vs-aws> (дата звернення: 25.01.2026).

39. Lystr Tech Blog. AWS vs DigitalOcean: Pricing, Performance, Features : Lystr : веб-сайт. URL: <https://www.lystr.tech/blog/aws-vs-digitalocean/> (дата звернення: 25.01.2026).

40. Center for Teaching Excellence, Rice University. Estimating Faculty Workload : Rice University : веб-сайт. URL: <https://cte.rice.edu/workload> (дата звернення: 25.01.2026).

41. Higher Ed Dive. Online courses may require more prep time than traditional classes, study says : Higher Ed Dive : веб-сайт. URL: <https://www.highereddive.com/news/online-courses-may-require-more-prep-time-than-traditional-classes-study-s/505629/> (дата звернення: 25.01.2026).

42. Lander University White Papers. Class Preparation Time: How Much Is Enough? : Lander University : веб-сайт. URL: https://www.lander.edu/about/_files/documents/whiteboards/whiteboard-11apr.pdf (дата звернення: 25.01.2026).

43. Bryant J. et al. How artificial intelligence will impact K-12 teachers : McKinsey & Company : веб-сайт. URL: <https://www.mckinsey.com/industries/education/our-insights/how-artificial-intelligence-will-impact-k-12-teachers> (дата звернення: 25.01.2026).

44. Goel A. K., Polepeddi L. Jill Watson: A Virtual Teaching Assistant for Online Education : Georgia Tech : веб-сайт. URL: <https://ws.gatech.edu/jill-watson-virtual-teaching-assistant> (дата звернення: 25.01.2026).

45. Верховна Рада України. Про Державний бюджет України на 2025 рік : офіційний сайт. URL: <https://zakon.rada.gov.ua/laws/show/4000-20> (дата звернення: 25.01.2026).

ДОДАТОК А ВІДОМОСТІ РОБОТИ

Таблиця А.1 – Відомості роботи

Формат	№ п/п	Назва документу	Найменування об'єкту або вибору	Кількість сторінок
A4	1	Пояснювальна записка	КЦТПАР.КН-24-1м.00.00.00.ПЗ	112
Графічна частина				
A4	2	Аналіз схеми роботи інфраструктури знаннєвої пам'яті мультиагентних систем	КЦТПАР.КН-24-1м.01.00.00.ПЛ	1
A4	3	Діаграма прецедентів функціональних можливостей	КЦТПАР.КН-24-1м.02.00.00.ПЛ	1
A4	4	Діаграма діяльності алгоритму роботи користувача з системою	КЦТПАР.КН-24-1м.03.00.00.ПЛ	2
A4	5	Діаграма станів життєвого циклу навчальної сесії	КЦТПАР.КН-24-1м.04.00.00.ПЛ	2
A4	6	Діаграма класів програмної архітектури	КЦТПАР.КН-24-1м.05.00.00.ПЛ	1
A4	7	Діаграма послідовностей взаємодії та передачі даних	КЦТПАР.КН-24-1м.06.00.00.ПЛ	2
A4	8	Діаграма потоків даних маршрутизації запитів	КЦТПАР.КН-24-1м.07.00.00.ПЛ	1
A4	9	Гістограма семантичної подібності агентів	КЦТПАР.КН-24-1м.08.00.00.ПЛ	2
A4	10	Графік залежності семантичної різноманітності	КЦТПАР.КН-24-1м.09.00.00.ПЛ	1

ДОДАТОК Б ТЕХНІЧНЕ ЗАВДАННЯ

Б.1 Введення

Найменування: Програмний комплекс для організації інтелектуального освітнього середовища з використанням AI-агентів.

Програмний продукт (ПП) призначений для автоматизації освітнього процесу, генерації адаптивного навчального контенту та моделювання групової динаміки через мультиагентну взаємодію. Система повинна забезпечувати рольову спеціалізацію агентів («Вчитель», «Розумник», «Жартівник»), підтримку RAG (Retrieval-Augmented Generation) для роботи з навчальними матеріалами, мультимодальну взаємодію (текст, голос) та зручний веб-інтерфейс.

Б.2 Підстава для розробки

Основою для розробки стали завдання кваліфікаційної роботи магістра, затверджені завідувачем кафедри цифрових технологій та проєктно-аналітичних рішень, в рамках спеціальності 122 «Комп'ютерні науки».

Б.3 Призначення розробки

Програмний комплекс має підвищити ефективність дистанційного навчання шляхом створення віртуального класу, де AI-агенти стимулюють дискусію та допомагають засвоєнню матеріалу. Система

автоматизує створення пояснень на основі завантажених лекцій, забезпечує цілодобову підтримку студентів та знижує навантаження на викладачів. Головною метою є підвищення залученості студентів та персоналізація навчального процесу.

Б.4 Вимоги до програмного виробу

Б.4.1 Вимоги до функціональних характеристик:

Програмне забезпечення має бути реалізоване у вигляді веб-застосунку з графічним інтерфейсом користувача. Архітектура повинна бути побудована на мові програмування Python.

Система повинна забезпечувати повний цикл обробки навчального контенту, включаючи можливість завантаження PDF-файлів з лекціями, їх автоматичну сегментацію та векторизацію для організації семантичного пошуку. Ключовою функціональною вимогою є реалізація мультиагентної взаємодії за сценарієм діалогу між спеціалізованими ролями («Вчитель», «Розумник», «Жартівник»), генерація відповідей має здійснюватися на основі контексту завантажених матеріалів для запобігання виникненню «галюцинацій». Програмний комплекс повинен підтримувати мультимодальні інтерфейси (Speech-to-Text та Text-to-Speech) і забезпечувати високий рівень інтерактивності, надаючи користувачеві можливість у будь-який момент втручатися в діалог агентів та отримувати миттєві відповіді на запитання.

Б.4.2 Вимоги до надійності

Система повинна передбачати обробку помилок при зверненні до зовнішніх API та коректно реагувати на нестабільне з'єднання. Має бути реалізований механізм контролю черги повідомлень, щоб уникнути хаотичного накладання реплік агентів. Дані сесії (історія чату, завантажені файли) повинні зберігатися тимчасово та бути ізольованими для кожного користувача.

Б.4.3 Вимоги до технічного забезпечення

Мінімальні вимоги до технічного забезпечення для сервера/локального запуску:

- Операційна система: Windows 10/11, Linux, macOS.
- Процесор: Intel Core i3 або аналог від AMD і вище.
- RAM: від 8 ГБ.
- Місце на диску: не менше 2 ГБ вільного простору.
- Мережа: стабільне підключення до Інтернету.
- Інтерпретатор: Python 3.10+.

Б.4.4 Умови експлуатації

Програмний комплекс розрахований на роботу в браузері (Chrome, Edge, Firefox). Користувач не потребує специфічних технічних навичок, окрім базового володіння веб-інтерфейсами. Система може використовуватися як інструмент самопідготовки студентів або як допоміжний засіб під час лекційних занять.

Б.4.5 Вимоги до програмної сумісності

Програмне забезпечення має бути сумісним із сучасними веб-браузерами. Серверна частина повинна підтримувати розгортання в контейнерах Docker для забезпечення кросплатформеності. Система має інтегруватися з зовнішніми сервісами через REST API (OpenAI API для LLM, TTS, STT). Формати вхідних даних: PDF, TXT. Формати вихідних даних: текст, аудіо (MP3/WAV).

Б.5 Вимоги до програмної документації

Документаційними матеріалами до програмного продукту є:

- технічне завдання;
- пояснювальна записка до магістерської роботи;
- інструкція з розгортання (налаштування оточення, встановлення залежностей, запуск сервера);
- керівництво користувача (завантаження файлів, взаємодія з чатом);
- діаграми архітектури системи (UML).

Б.6 Техніко-економічні показники

У роботі використано мову програмування Python та відкриті бібліотеки, що мінімізує витрати на ліцензійне ПЗ. Основні експлуатаційні витрати пов'язані з оплатою токенів OpenAI API, проте використання моделі gpt-4o-mini дозволяє суттєво знизити вартість (до ~4300 грн/рік). Впровадження системи дозволяє зекономити час

викладачів на підготовку матеріалів та консультації, що забезпечує окупність проекту менш ніж за 1.5 року.

Б.7 Стадії та етапи розробки

Стадії та етапи розробки програмного продукту зазначені у таблиці Б.1.

Таблиця Б.1 – Стадії та етапи розробки

№	Етап роботи	Опис виконуваної роботи
1	Аналіз предметної області	Дослідження сучасних підходів до AI в освіті, аналіз існуючих мультиагентних систем. Визначення обмежень традиційних LLM та формулювання вимог до гібридної системи.
2	Математичне моделювання	Розробка математичної моделі взаємодії агентів, визначення умов стабільності (спектральний радіус графа), розробка алгоритмів стохастичного контролю.
3	Проектування архітектури	Розробка архітектури системи, вибір технологічного стека, проектування схем взаємодії компонентів та бази даних.
4	Програмна реалізація	Написання коду серверної частини, реалізація логіки агентів, розробка веб-інтерфейсу та інтеграція мультимодальних функцій.
5	Експериментальні дослідження	Проведення тестів на стабільність генерації, перевірка рольової відмінності агентів, оцінка якості відповідей та оптимізація промптів.
6	Оформлення документації	Підготовка пояснювальної записки, опис результатів дослідження, оформлення графічних матеріалів та презентації.

Б.8 Порядок контролю і приймання

Контроль та приймання програмного забезпечення здійснюються відповідно до вимог технічного завдання та стандартів виконання магістерських робіт. Перевірка проводилась шляхом експериментального тестування стабільності діалогу агентів, оцінки релевантності відповідей та перевірки коректності роботи RAG-механізму. Результати дослідження підтвердили, що система забезпечує стабільну комунікацію, чітку рольову спеціалізацію агентів та ефективну роботу з навчальним контентом. Програмний продукт готовий до використання в навчальному процесі.

ДОДАТОК В

ПЕРЕЛІК ТА ВІДБИТКИ НАУКОВИХ ПУБЛІКАЦІЙ ЗА ТЕМОЮ РОБОТИ

СЕРТИФІКАТ
підтверджує, що

Ярослав Плуталов

приймала участь у ІХ Всеукраїнській науково-практичній конференції
«Сучасні інформаційні технології, засоби автоматизації та електропривод»,
м. Краматорськ – Тернопіль, 17 – 19 квітня 2025 р.
(6 годин – 0,2 кредити ECTS)

Голова
оргокомітету,
д.т.н., професор
О.Ф. Тарасов






<http://cit.dgma.donetsk.ua/>

Міністерство освіти і науки України

Донбаська державна
машинобудівна
академія



Донецький
фізико-технічний
інститут
ім. А. А. Галкіна
НАН України

ПАТ
«Новокраматорський
машинобудівний
завод»



Інститут економіки
промисловості
НАН України»

Громадська спілка «ІТ кластер
Донеччини» (IT Cluster Donbass)

ТОВ «Інформаційні
технології САПР»



за підтримкою
концерну
Siemens AG

SIEMENS

ЦЕНТР **САПР**

СУЧАСНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, ЗАСОБИ АВТОМАТИЗАЦІЇ ТА ЕЛЕКТРОПРИВОД

МАТЕРІАЛИ ІХ Всеукраїнської науково-практичної конференції

(17–19 квітня 2025 року)

Краматорськ – Тернопіль
ДДМА
2025

Людино-машинна командна робота: сучасний стан та напрямки розвитку

Плуталов Я. А., Нефедченко О. О., Сагайда П. І.
ТОВ «Технічний університет «Метінвест Політехніка»

Згідно з проаналізованими літературними джерелами [1-3], ключовими концепціями та компонентами людської та artificial intelligence (AI) agentic взаємодії є ті, що стосуються швидкого розвитку штучного інтелекту (ШІ) та його інтеграції в людино-машинні команди. Джерела підкреслюють еволюцію ШІ-агентів від пасивних інструментів до активних співпрацівників у цих командах.

Основними поняттями та компонентами сучасного стану та напрямки розвитку предметної області є наступні.

ШІ-агенти розглядаються як сутності, що сприймають своє оточення та діють раціонально, але сучасні ШІ-агенти еволюціонували до автономних, ітеративних та інтелектуальних членів команди. Вони здатні навчатися, розмірковувати та адаптуватися у складних середовищах. Характеристики агентних ШІ-систем включають планування та виконання багатоетапних

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»
ТОВ «МЕТІНВЕСТ ПОЛІТЕХНІКА»
Ройтлінгенський університет техніки та економіки (Німеччина)
Еслінгенський університет прикладних наук (Німеччина)
Технічний університет Фрайберзька гірничо академія (Німеччина)
Університет Кобленц-Ландау (Німеччина)
Вроцлавський технічний університет (Польща)
Краківська гірничо-металургійна академія (Польща)

**ПРОБЛЕМИ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
В ОСВІТІ, НАУЦІ ТА ПРОМИСЛОВОСТІ**

XX МІЖНАРОДНА КОНФЕРЕНЦІЯ

5 грудня 2025 року
м. Дніпро

Збірник наукових праць
№ 10

Дніпро
Журфонд
2025

УДК 622 (06)
П78

Редакційна колегія:

О.О. Азюковський, Г.Г. Півняк, О.С. Поважний, О.С. Бешта, В.А. Бойко, А.В. Голоядов, С.В. Детюк, О.В. Бондар, Є.Є. Мостіпан, І.М. Луценко, І.М. Удовик, Є.В. Кошеленко, М.О. Алексєєв, В.І. Корнієнко, В.В. Гнатушенко, О.О. Сдвижкова, Т.А. Желдак, О.О. Бешта, А.В. Бубліков, Ю.А. Папаїка, В.О. Пінчук, С.С. Худолій, Л.І. Мещеряков, А.І. Купін, М.І. Горбійчук, О.А. Дмитрієва, А. Дерен, Н. Нойбергер.

П78 **Проблеми** використання інформаційних технологій в освіті, науці та промисловості: XX міжнар. конф. (5 грудня 2025 р., м. Дніпро): зб. наук. пр. / ред. кол.: О.О. Азюковський та ін.; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро: НТУ «ДП», 2025. – № 10. – 594 с.

ISBN 978-617-8737-34-4

Подано результати теоретичних та експериментальних досліджень з різних аспектів використання інформаційних технологій та енергетики в освіті, науці та керуванні промисловістю. У публікаціях розглянуто питання створення та вдосконалення інформаційних систем та технологій, системного аналізу та наук про дані, кібербезпеки та захисту інформації, автоматизації та комп'ютерно-інтегрованих технологій, цифрової трансформації енергетики, мехатроніки та робототехніки і електромобільності.

Для наукових, інженерно-технічних співробітників і студентів, які спеціалізуються в галузі інформаційних технологій.

УДК 622 (06)

ISBN 978-617-8737-34-4

© НТУ «Дніпровська політехніка», 2025
© Журфонд, 2025

УДК 004.8:37.018.4

П.І. Сагайда¹, Я.А. Плуталов¹

¹ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА», Запоріжжя, Україна

ДОСЛІДЖЕННЯ МЕТОДІВ ТА АЛГОРИТМІВ ВИКОРИСТАННЯ AI-АГЕНТІВ ДЛЯ ОРГАНІЗАЦІЇ ІНТЕЛЕКТУАЛЬНОГО ОСВІТНЬОГО СЕРЕДОВИЩА

Анотація. У роботі проведено аналіз сучасних підходів до побудови мультиагентних AI-систем в освітньому середовищі, архітектурних й алгоритмічних рішень для реалізації групової взаємодії агентів та їхнього співпрацювання з реальним студентом. На прикладі проекту, де працюють двоє агентів-студентів (для створення соціальної та педагогічної атмосфери) і один агент-вчитель (пояснення матеріалу реальному студенту), обґрунтовано вибір методів декомпозиції задач, механізмів пам'яті й рефлексії, а також протоколів взаємодії між агентами та людиною.

Ключові слова: AI-агенти, мультиагентні системи, педагогічні агенти, LLM, інтелектуальне освітнє середовище, групове навчання, RAG.

Вступ. Розвиток великих мовних моделей (LLM) та архітектур мультиагентних систем започаткував нові парадигми для освіти, які моделюють групову взаємодію і соціальні ролі в навчанні. Різні підходи до проєктування агентів (модулі контролю, логіки, пам'яті та зовнішніх інтерфейсів) дають змогу комбінувати адаптивне навчання з підтримкою мотивації та емоційного залучення студентів. Останні праці пропонують формальні рамки побудови таких систем і демонструють емпіричні результати в реальних курсах [1,3].

Постановка задачі. Мета дослідження – розробити та дослідити методи й алгоритми, що забезпечують ефективну роботу локального освітнього мультиагентного середовища, яке містить: агента-вчителя з функціями пояснення матеріалу й адаптивних відповідей; двох агентів-студентів, що імітують соціальну взаємодію, створюють атмосферу групової роботи та стимулюють реального студента до участі. Завдання включають: формалізацію ролей агентів, розробку протоколів міжагентної комунікації, механізмів спільної рефлексії і механізмів управління пам'яттю, а також верифікацію впливу таких агентів на увагу, мотивацію й академічні результати студентів. Подібні архітектури описано у сучасних роботах про von Neumann MAS для освіти та дослідженнях LLM-самоорганізації [1,4].

Методологія та використовувані алгоритми. Архітектурна модель агентів. Кожен агент реалізується як модульна сутність із контролером діалогу, логічним шаром для педагогічних рішень, модулем пам'яті та інтерфейсом інструментів. Така структура узгоджена з підходом «чотрьох модулів» у сучасних MAS-фреймворках [1].

LLM як ядро поведінки. Агент-вчитель і агенти-студенти використовують тонко налаштовані LLM для генерації пояснень, запитань і реакцій; додатково

застосовують RAG (retrieval-augmented generation) для підключення доменної бази знань і курс-матеріалів. Рекомендовано розділяти “сприйняття вхідної інформації” та “планування” через зовнішні інструменти (retrieval, calculator, code execution) [1,2,5].

Міжагентна координація. Використовується протокол ролей: вчитель - фасилітатор, студенти-агенти - колаборативні партнери/мотиватори. Синхронні сценарії (живе заняття) керуються через lightweight message bus; асинхронні – через черги подій і спільну пам'ять (knowledge graph / vector DB). Подібні робочі процеси описано у систематичних оглядах LLM-MAS [3].

Рефлексія та адаптація. Застосовується цикл «дія - рефлексія - оновлення пам'яті», де агенти роблять самооцінювання (chain-of-thought + outcome logging) і коригують підходи до пояснень або стилю взаємодії. Цей підхід сприяє підвищенню когнітивної якості пояснень та внутрішнього узгодження поведінки агентів [4].

Узагальнена архітектура розробленої платформи наведено на рис. 1.

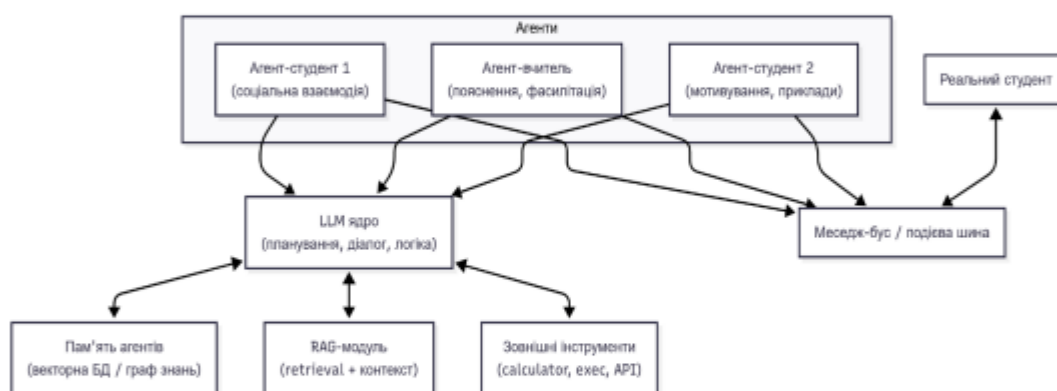


Рис. 1. Узагальнена архітектура розробленої платформи

Очікувані результати від впровадження:

- Покращення залученості студентів завдяки соціальним сигналам від агентів-студентів (збільшення часу взаємодії та кількості запитань).
- Зменшення когнітивного навантаження під час опанування нових тем при коректно налаштованому агенті-вчителі (попередні огляди показують помірний вплив педагогічних агентів на когнітивне навантаження [4]).
- Технічна стійкість архітектури: архітектура modular LLM + retrieval + memoгу дозволяє масштабувати сценарії від одного курсу до цілого освітнього модуля [4].

Наукова новизна роботи полягає в наступному.

1. Поєднання невеликого “соціального шару” (двоє агентів-студентів) із повнофункціональним агентом-вчителем для відтворення групової динаміки у синхронних сесіях.

2. Інтеграція циклу внутрішньої еволюції агентів зі зворотним зв'язком від реальних студентів – підхід, що підсилює “колективний інтелект” агентів і їхню педагогічну корисність.

3. Практичне застосування гібридних протоколів (RAG + multi-agent coordination [5]) для адаптивного пояснення і мотиваційної взаємодії.

Висновки. Реалізація мультиагентного освітнього середовища з ролями «вчитель - студенти - реальний студент» є перспективним напрямом, що поєднує переваги LLM-технологій, архітектур MAS та педагогічних практик. Подальші кроки: емпіричне дослідження впливу таких систем у навчальних експериментах, тонке налаштування персон і системних промптів агентів, а також розробка етичних і приватнісних політик для захисту студентських даних. Сучасні літературні джерела підкреслюють необхідність систематичного підходу до дизайну й оцінки педагогічних агентів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Jiang Y.-H., Li R., Zhou Y., Qi C., Hu H., Wei Y., Jiang B., Wu Y. AI Agent for Education: von Neumann Multi-Agent System Framework. *arXiv.org e-Print archive*. URL: <https://arxiv.org/html/2501.00083v1>
2. Hao Z. Student-AI Interaction in an LLM-Empowered Learning Environment. *arXiv.org e-Print archive*. URL: <https://arxiv.org/html/2503.01694v3>
3. Li X., Wang S., Zeng S., Wu Y., Yang Y. A Survey on LLM-based Multi-Agent Systems: workflow, infrastructure and challenges. URL: <https://link.springer.com/article/10.1007/s44336-024-00009-2>
4. Chu Z., Wang Sh., Xie J., Zhu T., Yan Y., Ye J., Zhong A., Hu X., Liang J., Yu S. Ph., Wen Q. LLM Agents for Education: Advances and Applications. URL: <https://arxiv.org/html/2503.11733v1>
5. Aquino G. A., Azevedo N., Okimoto L., Camelo L., Bragnaca H., Fernandes R., Printes A., Cardoso F., Claudio R., Torne I. From RAG to Multi-Agent Systems: A Survey of Modern Retrieval and MAS Techniques. URL: https://www.researchgate.net/publication/388773274_From_RAG_to_Multi-Agent_Systems_A_Survey_of_Modern_Approaches_in_LLM_Development