ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ МІСЬКОГО ГОСПОДАРСТВА ІМЕНІ О. М. БЕКЕТОВА
Навчально-науковий інститут енергетичної, інформаційної та транспортної інфраструктури

Кафедра комп'ютерних наук та інформаційних технологій

Пояснювальна записка
до кваліфікаційної роботи бакалавра

на тему: «Розробка Веб-платформи підтримки студентів з чат-ботом консультантом»

Виконала: студентка 4 курсу, групи ІСтаТ 2022-1
Спеціальності 126 Інформаційні системи та технології
(шифр і назва спеціальності)

Олена ГАРМАТА
(ім'я та прізвище)

Керівник: Володимир БРЕДІХІН
(ім'я та прізвище)

Рецензент Наталія СІЗОВА
(ім'я та прізвище)

м. Харків – 2026 рік

Харківський національний університет міського господарства імені О. М. Бекетова
(повне найменування закладу вищої освіти)
Навчально-науковий Інститут енергетичної, інформаційної
та транспортної інфраструктури

Кафедра комп'ютерних наук та інформаційних технологій

Рівень вищої освіти перший (бакалаврський)

Спеціальність 126 «Інформаційні системи та технології»
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри КНтаІТ


Марина НОВОЖИЛОВА
«__22__» __червня__2026 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Гармата Олени Сергіївни
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка Веб-платформи підтримки студентів з чат-ботом консультантом
керівник роботи к.т.н., доцент Бредіхін В.М.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «22» травня 2026 р. № 440-03
2. Термін подання студентом роботи 20.06.2026р.
3. Вихідні дані до роботи Рекомендації щодо розробки додатку, індивідуальне завдання на розробку
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
проаналізувати предметну область та прототипи системи; обґрунтувати вибір інструментального середовища та технічної платформи; виконати функціональний аналіз додатку; розробити програмну реалізацію та інтерфейс взаємодії з користувачем; описати архітектуру та інтерфейс розробки, перевірити функціонування програмного забезпечення.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація – 15 аркушів

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ I	Володимир БРЕДІХІН	11.05.2026	10.05.2026
Розділ II	Володимир БРЕДІХІН	17.05.2026	15.05.2026
Розділ III	Володимир БРЕДІХІН	21.05.2026	30.05.2026
Розділ IV	Вікторія МАЛИШЕВА	27.05.2026	15.06.2026

7. Дата видачі завдання 10.04.2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми дипломної роботи	03.05.2026	Викон.
2	Затвердження тем, наукових керівників, завдань та календарного плану підготовки дипломної роботи	05.05.2026	Викон.
3	Написання I розділу	10.05.2026	Викон.
4	Написання II розділу	15.05.2026	Викон.
5	Написання III розділу	20.05.2026	Викон.
6	Написання IV розділу	30.05.2026	Викон.
7	Подання дипломної роботи керівнику	05.06.2026	Викон.
8	Робота по усуненню зауважень керівника, уточнення і доповнення практичного матеріалу, оформлення додатків до роботи	10.06.2026	Викон.
9	Подання доопрацьованого варіанту роботи керівнику	15.06.2026	Викон.
10	Захист матеріалів дипломної роботи на засіданні кафедри	18.06.2026	Викон.
11	Офіційний захист матеріалів дипломної роботи на засіданні екзаменаційної комісії	23.06.2026	Викон.

Студент _____ Олена ГАРМАТА
(підпис) (прізвище та ініціали)

Керівник роботи _____ Володимир БРЕДІХІН
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Гармата О. С. Розробка веб-платформи підтримки студентів з чат-ботом консультантом. Кваліфікаційна робота бакалавра за спеціальністю 126 «Інформаційні системи та технології». - Харківський національний університет міського господарства імені О. М. Бекетова, Харків, 2026.

Кваліфікаційна робота присвячена проектуванню та розробці веб-платформи підтримки студентів із вбудованим чат-ботом консультантом, призначеним для автоматизації надання інформаційних і консультативних послуг у середовищі закладу вищої освіти.

Актуальність роботи обумовлена зростаючою потребою закладів вищої освіти в цифрових інструментах для покращення якості взаємодії зі студентами. Традиційні методи підтримки - звернення до деканату, телефонні консультації - є часозатратними та залежними від графіку роботи персоналу. Впровадження автоматизованої платформи усуває ці обмеження, забезпечуючи цілодобовий доступ до актуальної інформації.

Мета роботи - проектування та розробці комплексної веб-платформи підтримки студентів, ключовим елементом якої є чат-бот консультант, здатний оперативно обробляти запити користувачів та забезпечувати зручний доступ до навчально-адміністративної інформації.

У роботі вирішено такі завдання: проведено аналіз предметної області та огляд існуючих аналогів (Ivy.ai, Ada, Tidio, Moodle, Dialogflow); спроектовано архітектуру системи у вигляді триланкової клієнт-серверної моделі; розроблено алгоритм класифікації запитів з обчисленням коефіцієнта релевантності та порогом $\theta = 0{,}3$; реалізовано RESTful API із сімома ендпоінтами; розроблено адміністративну панель для управління базою та перегляду аналітики; складено керівництво користувача.

Практична значущість роботи полягає у створенні готового програмного рішення, яке може бути розгорнуте в будь-якому університеті без додаткових ліцензійних витрат. Система реалізована на відкритому стеку технологій і не

потребує встановлення СУБД у базовій версії.

Результатом роботи є функціональний прототип веб-платформи підтримки студентів, що забезпечує класифікацію запитів за чотирма категоріями, управління базою знань із 15 записами, збереження історії діалогів та збір аналітики звернень.

Ключові слова: веб-платформа, чат-бот, підтримка студентів, Flask, Python, JavaScript, класифікація запитів, RESTful API, база знань, NLP.

ANNOTATION

Harmata O. S. Development of a Student Support Web Platform with a Chatbot Consultant. Bachelor's qualification thesis in specialty 126 "Information Systems and Technologies". - O. M. Beketov National University of Urban Economy in Kharkiv, Kharkiv, 2026.

The qualification thesis is devoted to the design and development of a student support web platform with an integrated chatbot consultant aimed at automating the provision of information and advisory services within a higher education institution (HEI).

The relevance of the work is driven by the growing need of higher education institutions for digital tools to improve the quality of interaction with students. Traditional support methods - visiting the dean's office, telephone consultations - are time-consuming and dependent on staff working hours. Implementing an automated platform eliminates these limitations by providing 24/7 access to up-to-date information.

The objective of the thesis is to develop software for a student support web platform, design and development of a comprehensive web platform for student support, the key element of which is a chatbot consultant capable of promptly processing user requests and providing convenient access to educational and administrative information.

The following tasks have been accomplished: analysis of the subject domain and review of existing analogues (Ivy.ai, Ada, Tidio, Moodle, Dialogflow); system architecture design as a three-tier client-server model; development of a query classification algorithm computing a relevance coefficient with threshold $\theta = 0.3$; implementation of a RESTful API with seven endpoints; development of an administrative panel for knowledge base management and analytics monitoring; compilation of a user manual.

The practical significance of the work lies in creating a ready-made software solution that can be deployed in any university without additional licensing costs.

The system is implemented on an open technology stack and does not require a database management system in its basic version.

The result of the work is a functional prototype of a student support web platform that provides query classification across four categories, knowledge base management with 15 entries, dialogue history storage, and interaction analytics collection.

Keywords: web platform, chatbot, student support, Flask, Python, JavaScript, query classification, RESTful API, knowledge base, NLP.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

У кваліфікаційній роботі використано такі скорочення та умовні позначення:

АПІ (API) – Інтерфейс програмування застосунків (Application Programming Interface)

CORS – Спільне використання ресурсів між різними джерелами (Cross-Origin Resource Sharing)

CSS – Каскадні таблиці стилів (Cascading Style Sheets)

HTML – Мова гіпертекстової розмітки (HyperText Markup Language)

HTTP – Протокол передачі гіпертексту (HyperText Transfer Protocol)

JSON – Формат обміну даними (JavaScript Object Notation)

REST – Архітектурний стиль для розподілених гіпермедіа-систем (Representational State Transfer)

АПІ (API) – Інтерфейс програмування застосунків (Application Programming Interface)

АРМ – Автоматизоване робоче місце

БД – База даних

ЗВО – Заклад вищої освіти

ІС – Інформаційна система

ІТ – Інформаційні технології

КПЗ – Клієнт-серверний програмний застосунок

НЛП (NLP) – Обробка природної мови (Natural Language Processing)

ОС – Операційна система

ПЗ – Програмне забезпечення

ПК – Персональний комп’ютер

СПА (SPA) – Односторінковий застосунок (Single Page Application)

СУБД – Система управління базами даних

УМЛ (UML) – Уніфікована мова моделювання (Unified Modeling Language)

ЧБ – Чат-бот

ВСТУП

Трансформація освітнього процесу в умовах глобальної цифровізації потребує впровадження новітніх інструментів для покращення взаємодії між закладом вищої освіти (ЗВО) та студентом. Сучасний студент живе в умовах динамічного інформаційного потоку, де швидкість отримання відповіді на запитання часто є критичним фактором успішного навчання. Традиційні методи надання підтримки - звернення до деканатів, телефонні дзвінки або листування електронною поштою - поступово втрачають свою ефективність через тривалий час очікування, обмежений графік роботи адміністративних підрозділів та велике навантаження на персонал [1].

Особливої актуальності набуває розробка веб-платформ, інтегрованих із чат-ботами на основі штучного інтелекту або чітко структурованих сценаріїв. Чат-боти дозволяють автоматизувати надання відповідей на типові запити (розклад занять, розташування корпусів, правила оформлення документів, дедлайни сесії) у режимі 24/7. Це не лише підвищує рівень комфорту студентів, а й дозволяє адміністрації закладу зосередитися на вирішенні складних нестандартних завдань, делегуючи рутину програмним рішенням [2].

Мета дослідження полягає у проектуванні та розробці комплексної веб-платформи підтримки студентів, ключовим елементом якої є чат-бот консультант, здатний оперативно обробляти запити користувачів та забезпечувати зручний доступ до навчально-адміністративної інформації.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– проаналізувати особливості предметного середовища та специфіку комунікації в ЗВО;

– провести порівняльний аналіз існуючих аналогів та платформ для створення чат-ботів;

– розробити архітектуру веб-платформи та спроектувати базу даних;

– описати логіку роботи та алгоритми чат-бота (сценарні моделі та інтелектуальні компоненти);

– реалізувати програмний продукт та провести його тестування.

Об’єкт дослідження - процес інформаційної та консультативної підтримки студентів у системі управління вищою освітою.

Предмет дослідження - методи, алгоритми та програмні засоби розробки веб-платформ з інтегрованими чат-ботами для автоматизації надання інформаційних послуг.

Методи дослідження. Для розв’язання поставлених завдань використано методи системного аналізу (для дослідження предметної області), об'єктно-орієнтованого проектування (для створення архітектури системи), а також сучасні веб-технології та мови програмування (наприклад, JavaScript/Python, React/Vue для фронтенду тощо).

РОЗДІЛ 1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Сучасний заклад вищої освіти (ЗВО) є складним інформаційним середовищем, у якому щоденно циркулюють значні обсяги навчально-адміністративних даних. Студенти регулярно потребують актуальної інформації щодо розкладу занять, термінів складання іспитів, правил оформлення документів, розташування корпусів, контактів викладачів та перебігу навчального процесу. Традиційно для отримання таких відомостей студент змушений особисто звертатися до деканату, навчального відділу або викладача, що пов'язано з черговим очікуванням, обмеженим графіком роботи адміністративного персоналу та значним навантаженням на службовців [1].

Цифровізація освітнього процесу відкриває можливість автоматизувати значну частину цих взаємодій шляхом впровадження вебплатформ із вбудованими чат-ботами. Веб-платформа підтримки студентів є інтегрованим програмним рішенням, що поєднує інформаційний портал із модулем автоматизованої консультації, доступним у будь-який час без участі персоналу. Чат-бот консультант у складі такої платформи виконує роль першої лінії підтримки: він обробляє типові запити, надає миттєві відповіді та при необхідності перенаправляє студента до відповідного підрозділу або фахівця.

Предметне середовище даної роботи охоплює інформаційно-комунікаційні процеси між студентами та адміністративно-навчальною структурою ЗВО. Ключовими учасниками цих процесів є: студент як кінцевий користувач системи; адміністратор платформи, що відповідає за актуальність бази знань та налаштування сценаріїв; викладач, якому надходять повідомлення у разі ескалації запиту. Центральним об'єктом автоматизації є запит студента - структурована або довільна текстова фраза, що потребує оперативної та коректної відповіді.

Основними категоріями запитів у середовищі ЗВО є: організаційні

(розклад, сесія, канікули, академічна відпустка), адміністративні (документи, довідки, перезарахування, стипендія), навчальні (завдання, дедлайни, методичні матеріали) та технічні (доступ до електронного кабінету, пошта, платформи дистанційного навчання). Успішна обробка кожної категорії потребує чіткої структури бази знань платформи та відповідного налаштування логіки чат-бота [1].

1.1.1 Опис процесу діяльності

Процес надання інформаційної підтримки студентам у межах розроблюваної платформи розпочинається з моменту звернення користувача до чат-бота через веб-інтерфейс. Загальна послідовність дій охоплює такі етапи: автентифікація або анонімне звернення; введення запиту у текстовій формі; обробка запиту системою; формування відповіді; при необхідності - ескалація до живого оператора або перенаправлення на відповідний розділ платформи.

На етапі автентифікації студент може увійти до системи з використанням університетських облікових даних або скористатись платформою анонімно з обмеженим набором функцій. Авторизовані користувачі отримують персоналізовані відповіді: чат-бот враховує їхній курс, факультет і групу під час формування релевантного контенту.

Після введення запиту система класифікує його за тематичною категорією та зіставляє з відповідними записами бази знань. У разі збігу - надає готову відповідь. Якщо запит є нетиповим або неоднозначним, чат-бот пропонує уточнюючі варіанти або перелік найближчих тем. У критичних випадках - наприклад, при запиті про індивідуальний графік чи специфічну документацію - система ініціює передачу діалогу до відповідального співробітника або надсилає автоматичний запит до потрібного підрозділу [3].

Адміністратор платформи здійснює регулярне оновлення бази знань через адміністративну панель: додає нові записи, редагує застарілі відповіді, керує сценаріями діалогу та переглядає статистику звернень. Журналювання

всіх взаємодій дозволяє виявляти прогалини у наповненні системи та вдосконалювати якість відповідей. Таким чином, процес діяльності платформи є циклічним: кожне нове звернення потенційно збагачує базу знань і підвищує ефективність системи [3].

1.1.2 Опис функціональної моделі

Функціональна модель веб-платформи підтримки студентів відображає сукупність функцій, що виконуються системою для задоволення потреб користувачів. Модель охоплює чотири основні функціональні блоки: управління користувачами, обробка запитів через чат-бот, управління базою знань та адміністрування системи.

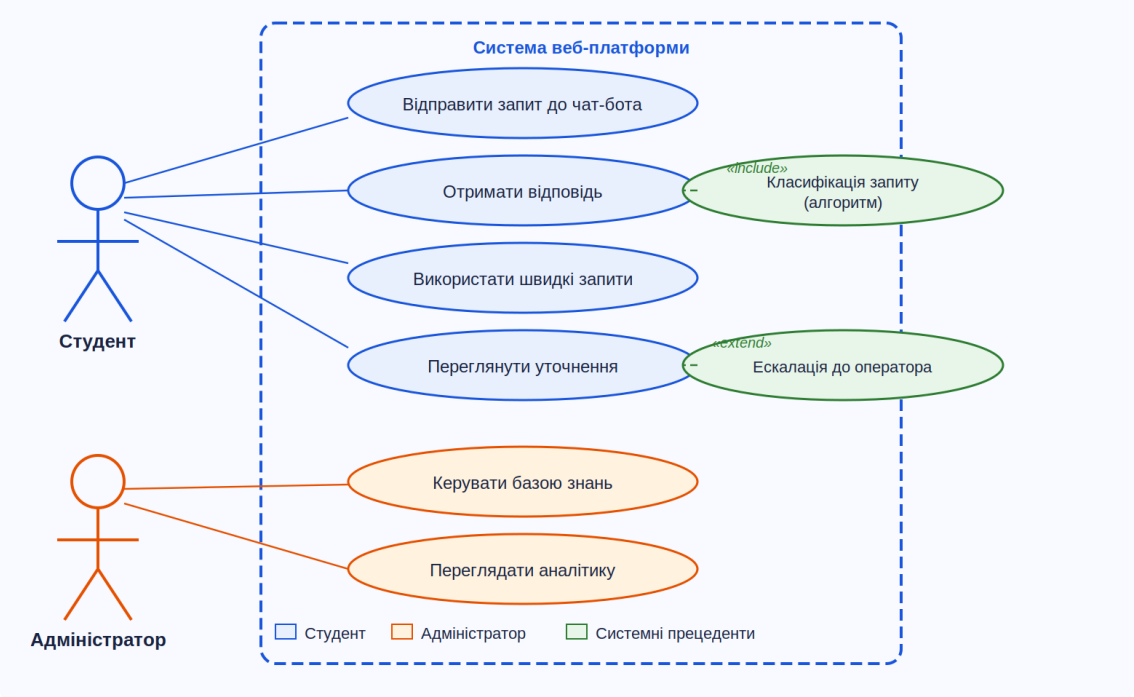

Рисунок 1.1 - Діаграма прецедентів веб-платформи підтримки студентів

Блок управління користувачами забезпечує реєстрацію та автентифікацію студентів, зберігання профілів із параметрами навчання (факультет, курс, група, форма навчання) та розмежування прав доступу між

роллю студента, викладача й адміністратора. Профільні дані використовуються чат-ботом для персоналізації відповідей і відображення контекстно-залежного контенту.

Блок обробки запитів є центральним компонентом платформи та включає підсистему прийому повідомлення, модуль класифікації запиту, рушій вибірки відповіді з бази знань і підсистему формування вихідного повідомлення. Для сценарного чат-бота класифікація базується на зіставленні ключових слів із заздалегідь визначеними патернами. За умови розширення системи до гібридного або інтелектуального рівня цей блок може бути доповнений модулем обробки природної мови (NLP) [2].

Блок управління базою знань реалізує функції створення, редагування та видалення записів, структурованих за тематичними категоріями. Кожен запис бази знань містить набір тригерних фраз, відповідний текст відповіді, посилання на додаткові ресурси та метадані для фільтрації (факультет, курс тощо). Адміністративна панель надає зручний інтерфейс для підтримки актуальності цього контенту.

Блок адміністрування системи охоплює функції перегляду статистики звернень, аналізу необроблених або ескальованих запитів, налаштування сценаріїв діалогу та управління сповіщеннями. Аналітичні звіти дозволяють адміністратору оперативно реагувати на зміни в потребах студентів та коригувати наповнення системи [2].

1.2 Огляд наявних аналогів

Аналіз ринку програмних рішень для підтримки студентів у ЗВО та суміжних галузях дозволив виокремити кілька категорій аналогів: спеціалізовані університетські чат-боти, універсальні платформи для створення чат-ботів та інтегровані системи управління навчанням (LMS) з елементами автоматизованої підтримки. Нижче розглянуто п'ять найбільш показових рішень.

Ivy.ai - американська платформа, розроблена спеціально для закладів вищої освіти, рис. 1.2.

Система забезпечує автоматизовану відповідь на запити щодо вступу, фінансової допомоги, розкладу та студентських послуг. Чат-бот навчається на даних конкретного університету і підтримує інтеграцію з CRM-системами та студентськими порталами. До переваг відносяться висока спеціалізація під освітній сектор, підтримка природної мови та можливість налаштування під бренд закладу. Недоліками є висока вартість впровадження, орієнтація на англомовні ринки та закрита архітектура, що ускладнює кастомізацію.

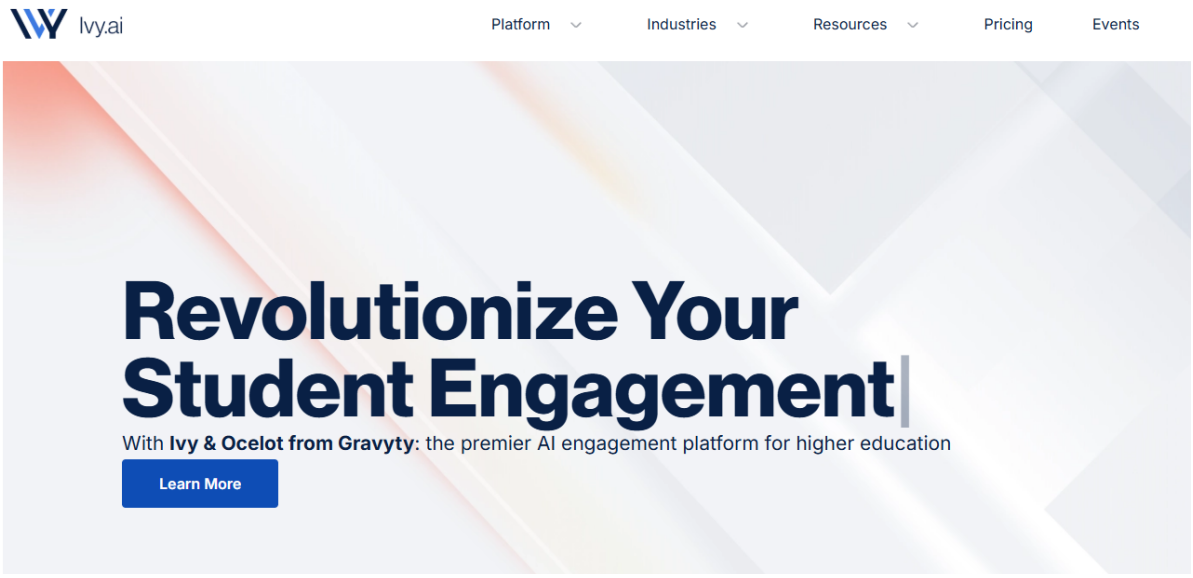

Рисунок 1.2 - Ivy.ai - американська платформа, розроблена спеціально для закладів вищої освіти

Ada - хмарна платформа для побудови чат-ботів без написання коду, рис. 1.3, що використовується низкою університетів для автоматизації служб підтримки. Система пропонує візуальний конструктор сценаріїв, аналітичну панель і підтримку багатомовних відповідей.

Серед переваг - простота налаштування, широкі інтеграційні можливості (Slack, Zendesk, електронна пошта) та масштабованість. Водночас безкоштовний функціонал суттєво обмежений, а глибока персоналізація

потребує технічної експертизи та платних тарифів.

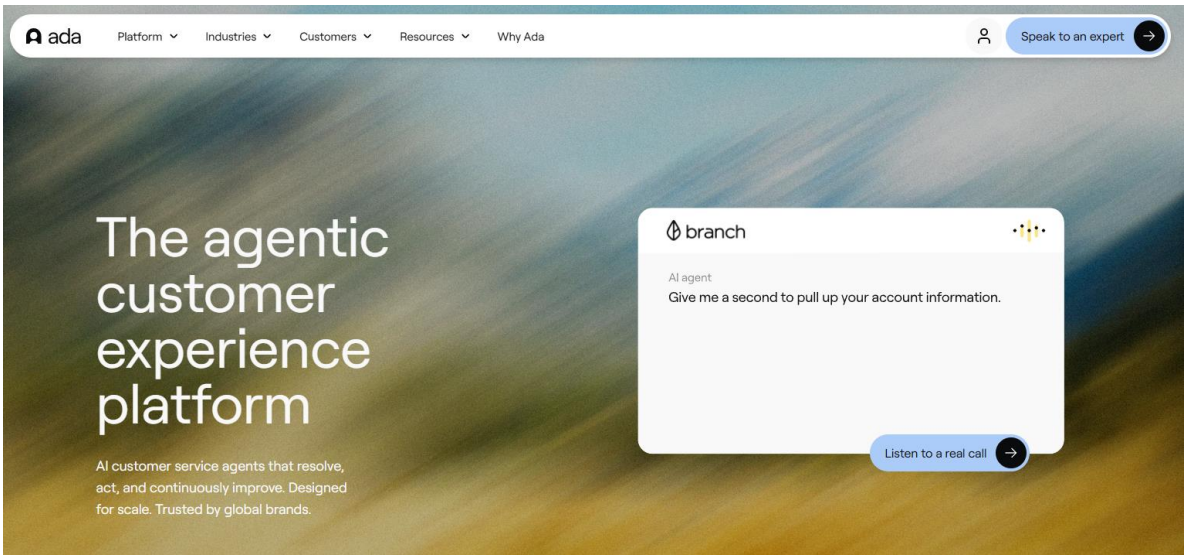

Рисунок 1.3 - Ada - хмарна платформа для побудови чат-ботів без написання коду

Tidio - платформа для побудови чат-ботів із гібридним режимом (автоматичні відповіді + підключення живого оператора). Інструмент широко застосовується на освітніх веб-сайтах для обробки вхідних звернень. Переваги: проста інтеграція на будь-який сайт через скрипт, зручний інтерфейс, наявність безкоштовного тарифного плану. Недолік - відсутність освітньої специфіки: система не підтримує прив'язку відповідей до груп, курсів чи факультетів, а база знань формується вручну без автоматичної синхронізації з університетськими системами.

Moodle + плагіни чат-ботів - система управління навчанням Moodle є одним із найпоширеніших інструментів у вітчизняних ЗВО. Завдяки відкритій архітектурі існують плагіни, що додають елементи чат-бота в навчальне середовище. Перевагою є повна інтегрованість із навчальним контентом і відсутність витрат на ліцензію. Однак реалізація чат-бота в Moodle залишається обмеженою: підтримується переважно сценарна логіка, персоналізація мінімальна, а підключення зовнішніх NLP-сервісів потребує

значних зусиль з боку технічних фахівців [2].

Dialogflow (Google) - платформа для розробки інтелектуальних розмовних агентів на основі машинного навчання та NLP. Дозволяє створювати чат-боти з розпізнаванням намірів (intents), контекстом діалогу та інтеграцією з різноманітними каналами (веб, Telegram, Messenger). Перевагами є потужний NLP-рушій, підтримка багатьох мов і широкі можливості інтеграції. Недоліки - платна модель для комерційного використання, залежність від хмарної інфраструктури Google та необхідність значного часу на навчання моделі під специфіку конкретного ЗВО [2].

Порівняльний аналіз розглянутих аналогів свідчить про те, що жодне з наявних рішень не забезпечує повної відповідності потребам українських ЗВО: спеціалізовані платформи є дорогими та закритими, універсальні - не адаптованими до освітньої специфіки, а відкриті рішення на кшталт Moodle потребують значних доробок. Це підтверджує доцільність розробки власної веб-платформи підтримки студентів із вбудованим чат-ботом, що орієнтована на реальні потреби вітчизняних закладів вищої освіти [1, 2, 3].

1.3 Постановка задачі

На підставі проведеного аналізу предметного середовища та огляду наявних аналогів сформульовано основну задачу дослідження: розробка веб-платформи підтримки студентів із вбудованим чат-ботом консультантом, що забезпечує оперативний доступ до навчально-адміністративної інформації, персоналізацію відповідей відповідно до профілю студента та зручне адміністрування системи.

Актуальність задачі зумовлена сукупністю чинників. По-перше, зростаючим попитом на цілодобову автоматизовану підтримку у вітчизняних ЗВО в умовах дистанційного та змішаного навчання [3]. По-друге, відсутністю доступних україномовних рішень, адаптованих до специфіки вітчизняної системи вищої освіти [1]. По-третє, необхідністю зниження адміністративного

навантаження на персонал закладів при одночасному підвищенні якості сервісу для студентів.

Об'єктом дослідження є процес інформаційної та консультативної підтримки студентів у системі управління вищою освітою.

Предметом дослідження є методи, алгоритми та програмні засоби реалізації веб-платформ із вбудованими чат-ботами для автоматизації надання інформаційних послуг студентам ЗВО.

Для досягнення поставленої мети необхідно вирішити такі задачі:

– проаналізувати предметне середовище та особливості інформаційних потреб студентів ЗВО;

– виконати порівняльний аналіз існуючих аналогів - платформ і чат-ботів для освітнього середовища;

– визначити функціональні та нефункціональні вимоги до розроблюваної платформи;

– розробити архітектуру веб-платформи та спроектувати структуру бази даних;

– реалізувати модуль чат-бота консультанта із логікою обробки запитів та базою знань;

– розробити адміністративну панель для управління контентом і сценаріями діалогу;

– провести тестування системи та оцінити ефективність розробленого рішення.

Для розв'язання поставлених задач використовуються такі методи дослідження: системний аналіз - для дослідження предметної області та формування вимог до системи; об'єктно-орієнтоване проектування - для побудови архітектури платформи; методи порівняльного аналізу - для оцінки наявних аналогів; сучасні веб-технології (Python, Django/Flask, REST API) - для реалізації програмного продукту; методи тестування програмного забезпечення - для перевірки коректності та продуктивності системи.

Практична цінність роботи полягає у створенні готового до

впровадження програмного продукту, який може бути адаптований під потреби конкретного ЗВО з мінімальними зусиллями. Розроблена платформа забезпечить цілодобову підтримку студентів, знизить навантаження на адміністративний персонал та підвищить якість інформаційного обслуговування у закладі.

Висновки до розділу

У першому розділі виконано комплексний аналіз предметного середовища, в якому функціонуватиме розроблювана система. Встановлено, що сучасний заклад вищої освіти потребує ефективних інструментів автоматизації інформаційної підтримки студентів, оскільки традиційні форми взаємодії не відповідають вимогам щодо оперативності та доступності в умовах цифрового освітнього середовища.

Визначено основні категорії інформаційних запитів студентів (організаційні, адміністративні, навчальні, технічні) та описано типовий процес діяльності системи - від звернення користувача до формування персоналізованої відповіді чат-ботом або ескалації до відповідального фахівця. Побудовано функціональну модель платформи, що включає чотири ключові блоки: управління користувачами, обробка запитів, управління базою знань та адміністрування системи.

Проведено огляд і порівняльний аналіз п'яти наявних аналогів: Ivy.ai, Ada, Tidio, Moodle з плагінами та Dialogflow. Аналіз показав, що спеціалізовані зарубіжні платформи є недоступними для більшості вітчизняних ЗВО через високу вартість і закриту архітектуру, тоді як універсальні інструменти не враховують специфіки освітнього процесу та потребують значних доробок. Це підтверджує актуальність і доцільність розробки власного рішення.

Сформульовано задачу дослідження, визначено об'єкт і предмет, окреслено перелік задач, що підлягають вирішенню в рамках роботи.

Обґрунтовано вибір методів дослідження та технологічної бази для реалізації платформи. Результати аналізу, проведеного в цьому розділі, є основою для проектування архітектури системи та розробки програмного продукту в наступних розділах.

РОЗДІЛ 2 ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Аналіз предметної області

Аналіз предметної області є першим і ключовим етапом проектування будь-якої інформаційної системи, оскільки саме на цьому кроці формується чітке розуміння контексту, в якому функціонуватиме програмний продукт. Предметною областю розроблюваної системи є інформаційно-консультаційна підтримка студентів закладів вищої освіти (ЗВО) засобами веб-технологій та автоматизованих чат-ботів [4, 5].

Взаємодія в системі побудована навколо трьох основних компонентів:

– Користувач (студент) - надсилає текстові запити щодо навчального процесу (дедлайни, розклад, документи) та отримує автоматичні відповіді;

– База знань - ядро системи, що містить структуровані шаблони запитань, відповідей та ключових фраз для керування діалогом;

– Адміністратор - забезпечує менеджмент системи, відповідаючи за модерацію, оновлення та наповнення бази знань актуальною інформацією» [7].

Аналіз предметної області дозволив виокремити такі ключові процеси, що підлягають автоматизації: прийом і класифікація студентських запитів; вибірка релевантної відповіді з бази знань; ескалація нестандартних запитів до відповідального персоналу; адміністрування бази знань та сценаріїв діалогу; збір аналітичних даних про частоту та тематику звернень. Кожен із цих процесів потребує відповідного інформаційного та алгоритмічного забезпечення, що визначає структуру системи загалом [8, 9].

Особливістю предметної області є висока варіативність природно-мовних запитів при відносно обмеженому наборі відповідей. Це зумовлює доцільність застосування ключово-орієнтованої класифікації в базовій реалізації з можливістю подальшого розширення до NLP-рушіїв. Крім того, важливою особливістю є необхідність підтримки україномовного контенту,

що обмежує застосування готових зарубіжних рішень та підкреслює потребу у власній розробці [5,11].

2.1.1 Вхідні дані

Вхідними даними системи є вся інформація, яку отримує платформа від користувачів та зовнішніх джерел для формування коректних відповідей. Визначення повного переліку вхідних даних є необхідною умовою правильного проектування структури бази даних і логіки обробки запитів [16].

До основних вхідних даних відносяться:

– текстовий запит студента - довільна або структурована фраза, введена у поле чат-бота; є головним вхідним елементом, на основі якого система формує відповідь;

– профіль користувача - ідентифікатор студента, факультет, курс, група, форма навчання; використовується для персоналізації відповідей та фільтрації релевантного контенту;

– дані бази знань - записи, що містять тригерні фрази, текст відповіді, категорію запиту та метадані (факультет, курс тощо); формуються адміністратором системи;

– налаштування сценаріїв діалогу - логічні правила та дерева рішень, що визначають послідовність взаємодії чат-бота з користувачем;

– дані автентифікації - університетський логін і пароль або токен сесії для ідентифікації авторизованих користувачів.

Додатковими вхідними даними, що використовуються в адміністративному модулі, є: дані оновлення бази знань (нові або відредаговані записи), параметри фільтрації для аналітичних звітів, а також системні події - наприклад, запити на ескалацію або тайм-аут сесії. Коректне опрацювання всіх перерахованих вхідних потоків забезпечує повноцінне функціонування платформи [2, 12].

2.1.2 Вихідні дані

Вихідними даними системи є результати обробки вхідних запитів, що передаються кінцевому користувачеві або суміжним підсистемам. Проектування вихідних даних безпосередньо пов'язане з вимогами до інтерфейсу та структури відповідей чат-бота [7, 18].

Основними вихідними даними платформи є:

– текстова відповідь чат-бота - сформований текст відповіді на запит студента, що відображається в інтерфейсі чату; може містити посилання на ресурси або рекомендації щодо подальших дій;

– уточнюючі варіанти - список найближчих за змістом тем або питань, що пропонуються студенту у разі неоднозначного запиту;

– повідомлення про ескалацію - сповіщення, що надсилається відповідальному співробітнику при передачі запиту до живого оператора;

– аналітичні звіти - зведені дані про кількість звернень, тематичний розподіл запитів та частку необроблених запитів; призначені для адміністратора системи;

– журнал взаємодій - повна послідовність питань і відповідей у рамках кожної сесії, що зберігається для подальшого аналізу та покращення якості системи.

Усі вихідні дані формуються динамічно на основі стану бази знань та параметрів поточного запиту. Коректність та релевантність вихідних даних є ключовим показником ефективності розробленої платформи і безпосередньо впливає на задоволеність студентів якістю обслуговування [5,6].

2.2 Проектування системи

Проектування системи є центральним етапом розробки програмного продукту, що передбачає перехід від концептуального опису предметної області до конкретних архітектурних і структурних рішень. На цьому етапі визначається загальна архітектура веб-платформи, склад та взаємодія

компонентів, структура збереження даних і об'єктно-орієнтована модель системи [14, 15].

Веб-платформа підтримки студентів проектується як триланкова архітектура: клієнтський рівень (браузерний інтерфейс), серверний рівень (бізнес-логіка та REST API) і рівень даних (база даних). Клієнтська частина реалізована як single-page application (SPA), що взаємодіє із сервером через HTTP-запити. Серверна частина побудована на основі фреймворку Django (Python) і реалізує логіку обробки запитів, класифікацію звернень та управління базою знань. Рівень даних представлений реляційною СУБД PostgreSQL [15, 16].

Основними принципами проектування є: модульність - чітке розмежування відповідальності між компонентами системи; масштабованість - можливість розширення функціональності без суттєвого переписування існуючого коду; безпека - розмежування прав доступу між ролями користувачів та захист API-ендпоінтів; підтримуваність - використання загальновживаних патернів і документованих технологій для полегшення супроводу системи [8, 9].

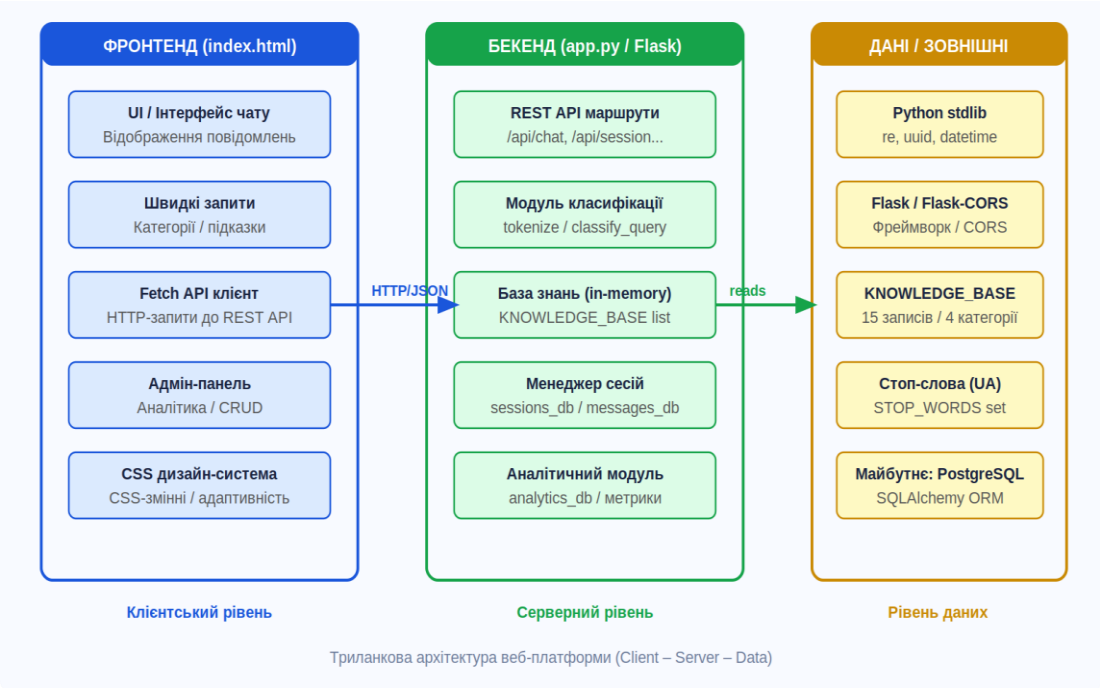

Рис. 2.1 - Діаграма модулів (компонентна архітектура) веб-платформи

2.2.1 Проектування бази даних

База даних є фундаментальним компонентом платформи, що забезпечує зберігання всіх структурованих даних системи: профілів користувачів, записів бази знань, журналів взаємодій та налаштувань сценаріїв діалогу. Для проектування бази даних застосовано метод сутність-зв'язок (ER-моделювання) з подальшою нормалізацією до третьої нормальної форми (3НФ) [12, 13].

Центральними сутностями бази даних є:

– User (Користувач) - зберігає ідентифікатор, логін, хеш пароля, роль (student / admin / teacher), а також зовнішній ключ на таблицю StudentProfile;

– StudentProfile (Профіль студента) - містить факультет, курс, групу та форму навчання; пов'язаний відношенням один-до-одного з сутністю User;

– KnowledgeBase (База знань) - основна таблиця системи; кожен запис містить поля: id, category_id, trigger_phrases, answer_text, related_links, faculty_filter, course_filter, created_at, updated_at;

– Category (Категорія) - класифікатор тематичних груп запитів (організаційні, адміністративні, навчальні, технічні);

– Session (Сесія діалогу) - зберігає user_id, start_time, end_time та статус сесії (active / closed / escalated);

– Message (Повідомлення) - журнал взаємодій; кожен запис містить session_id, role (user / bot), content та timestamp.

Між сутностями визначено такі зв'язки: User - StudentProfile (1:1), User - Session (1:N), Session - Message (1:N), KnowledgeBase - Category (N:1) як показано на рис. 2.1. Застосування зовнішніх ключів та обмежень цілісності забезпечує консистентність даних при будь-яких операціях модифікації [16, 17].

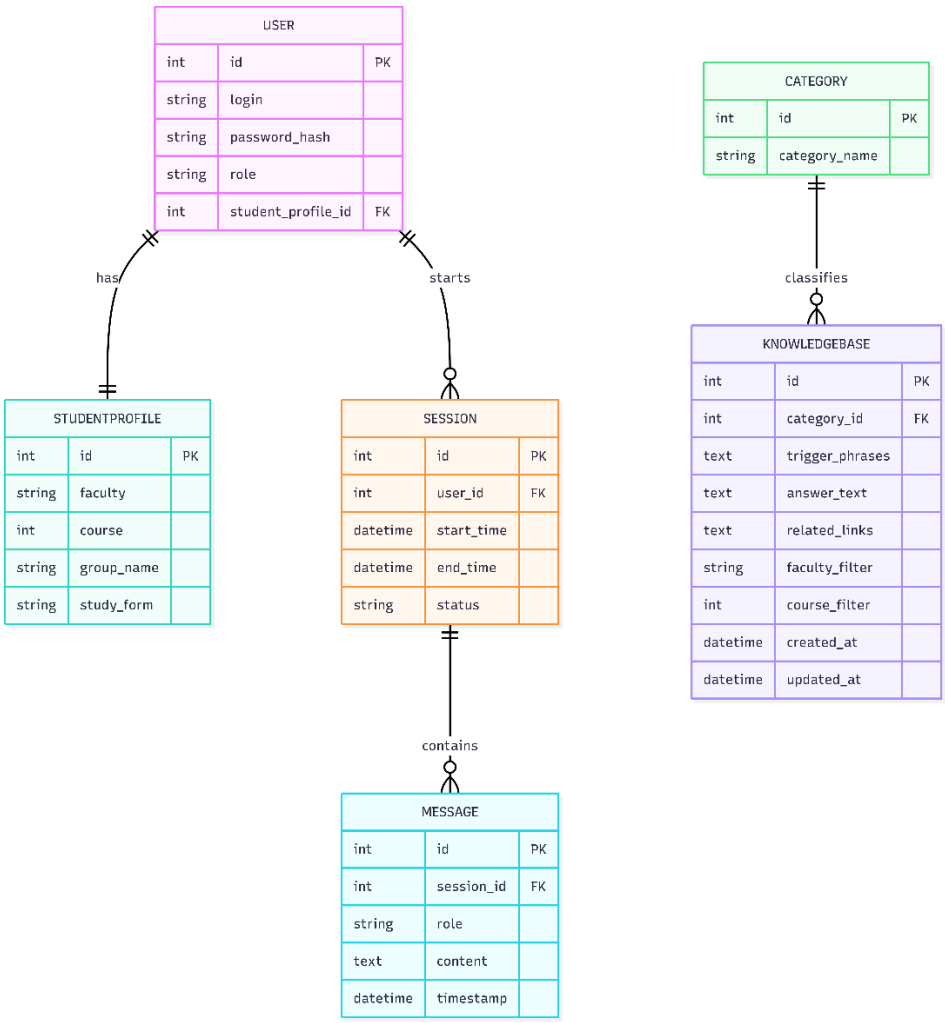

Рис. 2.2 ER-моделювання програми чат-боту

Для підвищення продуктивності запитів передбачено індексування полів: user_id у таблиці Session, session_id у таблиці Message, а також category_id і faculty_filter у таблиці KnowledgeBase. Пошук за тригерними фразами реалізується з використанням повнотекстового індексу GIN (Generalized Inverted Index), що є стандартним інструментом PostgreSQL для ефективної роботи з масивами рядків [10, 17].

2.2.2 Побудова об'єктно-орієнтованої моделі

Об'єктно-орієнтована модель (ООМ) системи описує її структуру у термінах класів, атрибутів, методів та відношень між ними. Побудова ООМ

виконується на основі вимог до функціональності платформи та відображає програмну архітектуру серверної частини [8, 9].

Ключовими класами системи є такі. Клас User інкапсулює дані облікового запису та надає методи authenticate(), getProfile() і hasPermission(). Клас ChatBot є центральним класом обробки запитів і містить методи receiveMessage(), classifyQuery(), fetchAnswer() та escalate(). Клас KnowledgeBaseManager реалізує CRUD-операції над записами бази знань: addEntry(), updateEntry(), deleteEntry() та searchByPhrase(). Клас SessionManager управляє станом сесій діалогу: createSession(), closeSession() та getHistory(). Клас AnalyticsService формує аналітичні звіти та надає методи getStats(), getTopQueries() та getUnansweredRate() [10, 11].

Між класами визначено відношення такого типу: ChatBot залежить від KnowledgeBaseManager (dependency) для вибірки відповідей; SessionManager агрегує об'єкти класу Message; User композиційно пов'язаний зі StudentProfile, оскільки профіль не існує поза обліковим записом. AnalyticsService використовує SessionManager та KnowledgeBaseManager для агрегації даних [9, 18].

Для реалізації сценарного чат-бота застосовано патерн проектування Strategy: клас ChatBot делегує вибір алгоритму класифікації запитів об'єктам-стратегіям KeywordStrategy або (у перспективі) NLPStrategy, що дозволяє розширювати логіку без зміни базового класу. Патерн Observer застосовано для реалізації механізму ескалації: при виникненні події нестандартного запиту SessionManager сповіщає підписані обробники - EmailNotifier та AdminDashboardUpdater. Такий підхід забезпечує гнучкість та низьку зв'язаність компонентів системи [8, 18].

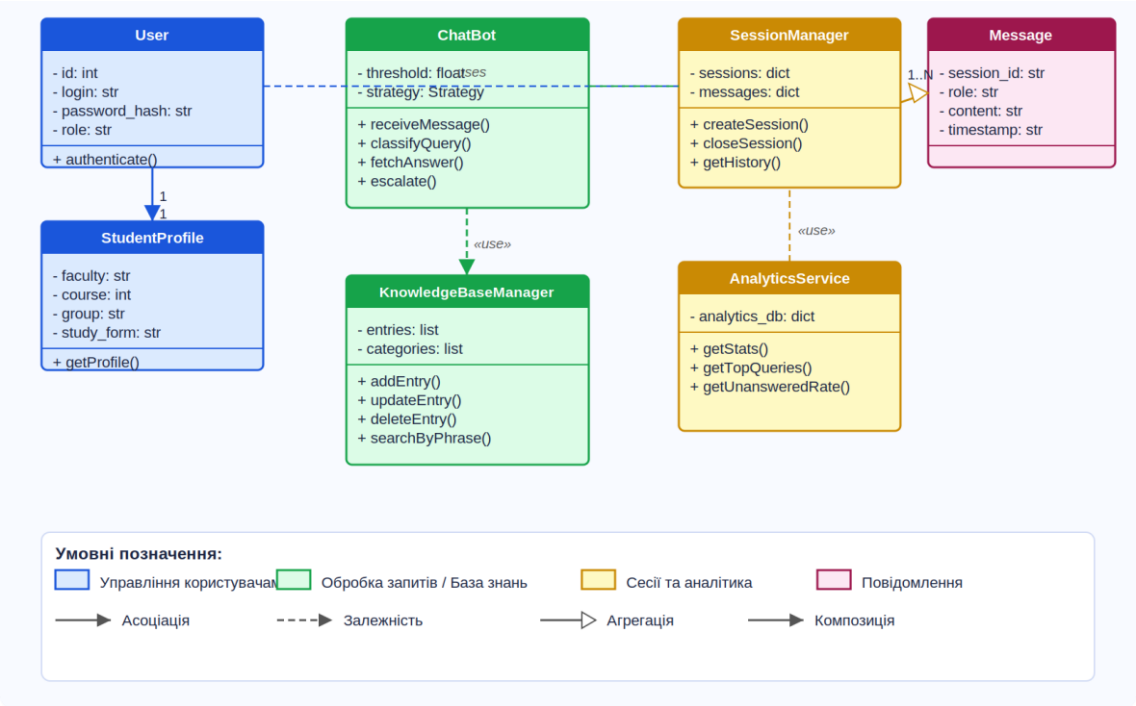

Рисунок 2.3 - UML-діаграма класів веб-платформи підтримки студентів

2.3 Математичне та алгоритмічне забезпечення

Математика та алгоритми - це, по суті, "мізки" нашого чат-бота. Вони відповідають за два ключові процеси: правильно класифікувати запит користувача та швидко витягнути точну відповідь із бази даних. Наскільки грамотно ми налаштуємо ці алгоритми, настільки швидким і розумним буде сам бот [1, 2].

Основним алгоритмом класифікації у базовій реалізації є ключово-орієнтований пошук із зваженим показником збігу. Для кожного запиту q та кожного запису бази знань k обчислюється коефіцієнт релевантності R(q, k) за формулою:

$$R(q, k) = |T(q) \cap T(k)| / |T(k)|, \quad (2.1)$$

де T(q) - множина токенів (слів) запиту після попередньої обробки (приведення до нижнього регістру, видалення стоп-слів, лематизація), T(k) -

множина тригерних токенів запису k. Запис із найвищим значенням R(q, k) при умові R(q, k) ≥ θ обирається як відповідь, де θ - поріг релевантності (рекомендоване значення θ = 0.5). Якщо жоден запис не перевищує поріг, система формує список уточнюючих варіантів або ініціює ескалацію [4].

Алгоритм обробки запиту в загальному вигляді складається з таких кроків:

1. токенізація вхідного рядка та видалення стоп-слів;

2. застосування фільтру персоналізації - обмеження пошуку записами, що відповідають факультету та курсу поточного користувача;

3. обчислення R(q, k) для всіх записів відфільтрованої вибірки;

4. сортування результатів за спаданням R;

5. повернення відповіді або списку уточнень залежно від значення максимального R [6,7].

Часова складність базового алгоритму при пошуку по всій базі знань становить O(n·m), де n - кількість записів у базі знань, m - середня кількість тригерних токенів в одному записі. Застосування повнотекстового індексу GIN дозволяє знизити реальний час виконання пошукових запитів до O(log n) завдяки перевернутому індексуванню токенів, що є прийнятним показником для баз знань ЗВО обсягом до кількох тисяч записів [13, 20].

Для оцінки якості роботи чат-бота застосовуються стандартні метрики інформаційного пошуку:

– точність (Precision) - частка релевантних відповідей серед усіх наданих відповідей;

– повнота (Recall) - частка правильно оброблених запитів серед усіх отриманих запитів;

– F1-міра - гармонічне середнє точності та повноти.

Цільовими значеннями для розроблюваної системи визначено Precision ≥ 0.85 та Recall ≥ 0.80, що є прийнятним рівнем для сценарного чат-бота в обмеженій предметній області [5, 6].

Висновки до розділу

У другому розділі розроблено комплексне інформаційне та математичне забезпечення веб-платформи підтримки студентів. На основі аналізу предметної області визначено повний перелік вхідних і вихідних даних системи, що стало фундаментом для проектування її структури.

Спроектовано реляційну базу даних у третій нормальній формі, що охоплює шість взаємопов'язаних сутностей: User, StudentProfile, KnowledgeBase, Category, Session та Message. Для підвищення продуктивності пошукових запитів передбачено застосування повнотекстового індексу GIN та індексування зовнішніх ключів.

Побудовано об'єктно-орієнтовану модель серверної частини системи, що включає п'ять ключових класів із чітко визначеними відповідальностями та зв'язками. Застосування патернів проектування Strategy та Observer забезпечує гнучкість архітектури та можливість розширення функціональності без зміни базового коду.

Обґрунтовано математичну основу алгоритму класифікації запитів на базі зваженого коефіцієнта релевантності, визначено порогові значення та метрики якості роботи чат-бота. Часова складність алгоритму O(log n) при використанні індексування є достатньою для забезпечення відповіді в реальному часі.

РОЗДІЛ 3 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Засоби розробки

Для реалізації веб-платформи підтримки студентів із вбудованим чат-ботом консультантом обрано сучасний стек технологій, що поєднує потужний серверний фреймворк на мові Python та легку клієнтську частину на базі нативних веб-технологій. Вибір інструментів обумовлений вимогами до швидкості розробки, читабельності коду, широкої документації та можливостей подальшого масштабування системи.

Основним засобом серверної розробки обрано мову Python версії 3.11 у поєднанні з мікрофреймворком Flask. Python є однією з найпоширеніших мов програмування для розробки веб-застосунків та систем, пов'язаних із обробкою природних мов та інтелектуальними алгоритмами [6]. Flask забезпечує мінімалістичну архітектуру без надлишкових абстракцій, що дозволяє точно контролювати всі компоненти системи. Для забезпечення взаємодії між клієнтською та серверною частинами через крос-доменні запити підключено розширення Flask-CORS.

Клієнтська частина реалізована засобами нативного HTML5, CSS3 та JavaScript без залучення зовнішніх фреймворків. Такий підхід зменшує кількість залежностей та забезпечує максимальну швидкість завантаження інтерфейсу. Стилізацію виконано з використанням CSS-змінних (Custom Properties) та сучасних технологій CSS Grid і Flexbox, що дозволяє отримати адаптивний і семантично правильний макет.

У процесі розробки застосовано наступні ключові технології та бібліотеки:

– Python 3.11 - мова програмування серверної частини; забезпечує читабельний синтаксис, потужну стандартну бібліотеку та підтримку регулярних виразів для обробки тексту;

– Flask 3.x - мікрофреймворк для побудови RESTful API; підтримує маршрутизацію, обробку JSON-запитів та сесії;

– Flask-CORS - розширення для обробки CORS-заголовків, що необхідно для взаємодії фронтенду та бекенду в режимі розробки;

– JavaScript (ES2022) - мова клієнтської логіки; використовується для обробки подій, формування запитів до API та динамічного оновлення DOM;

– HTML5 / CSS3 - стандартні технології розмітки та стилізації веб-інтерфейсу;

– Fetch API - вбудований у браузер інтерфейс для здійснення асинхронних HTTP-запитів до серверного API без сторонніх бібліотек;

– Regular Expressions (модуль re) - використовується для токенізації україномовного тексту у модулі класифікації запитів;

– uuid / datetime - стандартні модулі Python для генерації унікальних ідентифікаторів сесій та часових міток.

Середовищем розробки є Visual Studio Code з розширеннями Python, Pylance та Live Server для зручного налагодження обох частин застосунку. Версійний контроль забезпечується системою Git.

3.2 Вимоги до технічного та програмного забезпечення

Визначення вимог до технічного та програмного забезпечення є важливою передумовою успішного розгортання та стабільної роботи розробленої веб-платформи. Вимоги розподілено на дві категорії: вимоги до серверної (хостингової) сторони та вимоги до клієнтського пристрою кінцевого користувача.

Серверне середовище. Для роботи серверної частини на базі Flask необхідне виконання таких вимог:

– операційна система - Linux (Ubuntu 20.04 LTS або новіша), Windows Server 2019+ або macOS 12+;

– процесор - не менше 1 ядра з тактовою частотою від 1.6 ГГц; рекомендовано 2+ ядра для продуктивної роботи;

– оперативна пам'ять - мінімум 512 МБ; рекомендовано 2 ГБ для забезпечення стабільної роботи при навантаженні;

– Python 3.9 або вища версія (рекомендовано 3.11);

– встановлені бібліотеки: Flask 3.x, Flask-CORS (встановлюються через pip);

– наявність вільного порту 5000 (за замовчуванням) або іншого налаштованого порту;

– доступ до мережі Інтернет - для продуктивного розгортання та можливості підключення зовнішніх NLP-сервісів у майбутньому.

Клієнтське середовище. Для роботи з веб-інтерфейсом платформи кінцевому користувачеві необхідний лише сучасний веб-браузер:

– Google Chrome 110+, Mozilla Firefox 110+, Microsoft Edge 110+ або Safari 16+;

– підтримка JavaScript (ES2022) - увімкнена за замовчуванням у всіх сучасних браузерах;

– підтримка CSS Custom Properties (CSS Variables) та Fetch API;

– роздільна здатність екрана - не менше 1024×768 пікселів;

– стабільне підключення до мережі Інтернет або локальної мережі університету.

Вимоги до програмного забезпечення для розробника при локальному розгортанні системи зводяться до встановлення Python 3.11, менеджера пакетів pip та виконання команди встановлення залежностей:

– pip install flask flask-cors

– python app.py

Платформа не потребує встановлення бази даних: у прототипній версії всі дані зберігаються in-memory, що спрощує розгортання та усуває необхідність у налаштуванні СУБД. У продуктивному середовищі рекомендується перехід на PostgreSQL або SQLite з відповідним розширенням SQLAlchemy.

3.3 Опис програмної реалізації

Програмна реалізація веб-платформи підтримки студентів складається з двох взаємопов'язаних компонентів: серверного застосунку (бекенд) на Python/Flask та клієнтського одно-сторінкового інтерфейсу (фронтенд) на нативному HTML/CSS/JavaScript. Між ними організовано взаємодію через RESTful API з обміном даними у форматі JSON.

Серверна частина (app.py). Серверний застосунок побудовано за принципом єдиного модуля, що містить усі компоненти: базу знань, алгоритм класифікації запитів, управління сесіями та API-ендпоінти.

Ядром бекенду є структура бази знань - масив об'єктів KNOWLEDGE_BASE, кожен із яких описує один запис рис.3.1 :

```
KNOWLEDGE_BASE = [
{   "id": 1,
"category": "організаційні",
"triggers": ["розклад", "пари", "заняття", "лекції"],
"answer": "🗓 Розклад занять доступний на сайті університету...",
"links": [{"text": "Сайт університету", "url": "#"}]
},
# ... 14 записів загалом]
```

Рисунок 3.1 - Фрагмент структури бази знань

Класифікація запитів реалізована через функцію tokenize(),як видно на рис.3.2, яка виконує нормалізацію тексту, та compute_relevance(), що обчислює релевантність запиту відносно кожного запису бази знань за формулою з розділу 2.3:

```
    def tokenize(text: str) -> set:
    """Токенізація: нижній регістр, тільки літери, без стоп-
слів."""
    text = text.lower()
    tokens = re.findall(r"[а-яіїєґa-z']+", text)
    return {t for t in tokens if t not in STOP_WORDS and
len(t) > 2}
    def compute_relevance(query_tokens: set, entry: dict) -
> float:
    best_score = 0.0
    for phrase in entry["triggers"]:
    phrase_tokens = tokenize(phrase)
    if not phrase_tokens:
        continue
    intersection = len(query_tokens & phrase_tokens)
    score = intersection / len(phrase_tokens)
    if score > best_score:
        best_score = score
    # Бонус за точне входження рядка
    query_str = " ".join(sorted(query_tokens))
    if  phrase.lower()  in  query_str  or  query_str  in
phrase.lower():
        best_score = max(best_score, 0.95)
    return best_score
```

Рисунок 3.2 - Функції токенізації та обчислення релевантності

Головна функція classify_query() об'єднує токенізацію та ранжування, повертаючи один із трьох типів результату: "answer" (знайдено відповідь), "suggestions" (пропозиції уточнень) або "escalate" (запит не розпізнано, рекомендується ескалація) рис. 3.3:

```
    def classify_query(user_message: str, threshold: float
= 0.3) -> dict:
        query_tokens = tokenize(user_message)
        if not query_tokens:
            return {"type": "empty"}
        scores = []
        for entry in KNOWLEDGE_BASE:
            score = compute_relevance(query_tokens, entry)
            if score > 0:
                scores.append((score, entry))
        scores.sort(key=lambda x: x[0], reverse=True)
        analytics_db["total"] += 1
        if not scores or scores[0][0] < threshold:
            analytics_db["unanswered"] += 1
            suggestions = [e["triggers"][0] for _, e in
scores[:3]]
            return {"type": "suggestions", "suggestions":
suggestions}
        best_entry = scores[0][1]
        cat = best_entry["category"]
        analytics_db["categories"][cat] = \
            analytics_db["categories"].get(cat, 0) + 1
        return {"type": "answer", "entry": best_entry,
"score": scores[0][0]}
```

Рисунок 3.3 - Функція класифікації запитів

API-ендпоінти реалізовані як Flask-маршрути. Основний ендпоінт /api/chat приймає POST-запит із повідомленням користувача та ідентифікатором сесії, викликає класифікацію та повертає структуровану відповідь. Додатково реалізовано ендпоінти для адміністративної панелі: /api/admin/analytics (статистика звернень), /api/admin/knowledge (CRUD-операції над базою знань) та /api/health (перевірка стану сервісу).

Управління сесіями організовано через in-memory словники sessions_db

та messages_db. Кожна сесія отримує унікальний UUID-ідентифікатор та зберігає повну історію діалогу з часовими мітками. Це дозволяє відновлювати контекст розмови та надавати адміністратору журнал звернень.

Клієнтська частина (index.html). Фронтенд реалізований як односторінковий застосунок (SPA) без фреймворків. Інтерфейс містить три функціональні вкладки рис.3.4 : «Чат», «Категорії» та «Адмін-панель».

Рис. 3.4 - Функціональні вкладки веб-платформи для студентів

Навігація між вкладками реалізована через JavaScript-функцію переключення класу active без перезавантаження сторінки:

```
function showView(name) {
  document.querySelectorAll(".view").forEach(v => {
    v.classList.toggle("active", v.id === name + "-
view");
  });
  document.querySelectorAll(".nav-tab").forEach(t => {
    t.classList.toggle("active", t.dataset.view ===
name);
  });
  currentView = name;
  if (name === "admin") loadAnalytics();
}
```

Рисунок 3.5 - Функція навігації між вкладками інтерфейсу

Головна функція відправлення повідомлення sendMessage() формує асинхронний запит до /api/chat через Fetch API, відображає індикатор друку (typing indicator) на час очікування відповіді та динамічно додає бульбашки

повідомлень до DOM:

```
async function sendMessage() {
  const text = msgInput.value.trim();
  if (!text || isLoading) return;
  appendMsg("user", text);
  msgInput.value = "";
  showTyping(true);
  isLoading = true;
  try {
    const res = await fetch(API + "/chat", {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ message: text, session_id:
sessionId })
    });
    const data = await res.json();
    sessionId = data.session_id;
    appendBotMsg(data.reply);
  } catch (e) {
    appendBotMsg({ type:"error",
      text: "Помилка зв'язку із сервером." });
  } finally {
    showTyping(false);
    isLoading = false;
  }
}
```

Рисунок 3.6 - Функція відправлення повідомлення на сервер

Стилізація інтерфейсу виконана з використанням CSS Custom Properties (CSS-змінних), що забезпечує єдину систему дизайн-токенів для кольорів, тіней, заокруглень та відступів. Завдяки такому підходу зміна загальної палітри системи вимагає модифікації лише блоку :root:

```
:root {
  --primary: #1a56db;
  --primary-dark: #1240a8;
  --primary-light: #e8f0fe;
  --bg: #f0f4ff;
  --surface: #ffffff;
  --border: #d0daf5;
  --radius: 14px;
  --shadow: 0 2px 16px rgba(26,86,219,0.10);
}
```

Рисунок 3.7 - Блок CSS-змінних дизайн-системи

Адміністративна панель надає можливість переглядати статистику звернень у реальному часі (загальна кількість, оброблені та необроблені запити, розподіл за категоріями) та управляти записами бази знань. Функція loadAnalytics() викликає ендпоінт /api/admin/analytics та оновлює відповідні DOM-елементи панелі, що дозволяє адміністратору оперативно відстежувати ефективність системи.

3.4 Керівництво користувача

Дане керівництво описує порядок встановлення, запуску та роботи з веб-платформою підтримки студентів для двох категорій користувачів: студента (кінцевого користувача) та адміністратора системи.

Встановлення та запуск. Для розгортання платформи в локальному середовищі необхідно виконати такі кроки.

Крок 1. Переконатися, що на комп'ютері встановлено Python версії 3.9 або вище. Перевірити версію Python можна командою рис. 3.8 :

python –version

```
C:\Users\CyberYou>python --version
Python 3.12.10
```

Рисунок 3.8 - Перевірка встановленої версії Python

Крок 2. Встановити необхідні залежності на рис. 3.9 показано детальніше, що було встановлено. У командному рядку або терміналі виконати:

```
C:\Users\CyberYou>python -m pip install flask flask-cors
Defaulting to user installation because normal site-packages is not writeable
Collecting flask
  Downloading flask-3.1.3-py3-none-any.whl.metadata (3.2 kB)
Collecting flask-cors
  Downloading flask_cors-6.0.5-py3-none-any.whl.metadata (5.4 kB)
Collecting blinker>=1.9.0 (from flask)
  Downloading blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
Collecting click>=8.1.3 (from flask)
  Downloading click-8.4.1-py3-none-any.whl.metadata (2.6 kB)
Collecting itsdangerous>=2.2.0 (from flask)
  Downloading itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting jinja2>=3.1.2 (from flask)
  Downloading jinja2-3.1.6-py3-none-any.whl.metadata (2.9 kB)
Collecting markupsafe>=2.1.1 (from flask)
  Downloading markupsafe-3.0.3-cp312-cp312-win_amd64.whl.metadata (2.8 kB)
Collecting werkzeug>=3.1.0 (from flask)
  Downloading werkzeug-3.1.8-py3-none-any.whl.metadata (4.0 kB)
Collecting colorama (from click>=8.1.3->flask)
  Downloading colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Downloading flask-3.1.3-py3-none-any.whl (103 kB)
Downloading flask_cors-6.0.5-py3-none-any.whl (16 kB)
Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)
Downloading click-8.4.1-py3-none-any.whl (116 kB)
Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Downloading jinja2-3.1.6-py3-none-any.whl (134 kB)
Downloading markupsafe-3.0.3-cp312-cp312-win_amd64.whl (15 kB)
Downloading werkzeug-3.1.8-py3-none-any.whl (226 kB)
Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: markupsafe, itsdangerous, colorama, blinker, werkzeug, jinja2, click, flask, flask-cors
  WARNING: The script flask.exe is installed in 'C:\Users\CyberYou\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2kfra8p0\LocalCache\local-packages\Python312\Scripts' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed blinker-1.9.0 click-8.4.1 colorama-0.4.6 flask-3.1.3 flask-cors-6.0.5 itsdangerous-2.2.0 jinja2-3.1.6 markupsafe-3.0.3 werkzeug-3.1.8

[notice] A new release of pip is available: 25.0.1 -> 26.1.2
[notice] To update, run: C:\Users\CyberYou\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.12_qbz5n2kfra8p0\python.exe -m pip install --upgrade pip
```

Рисунок 3.9 - Встановлення залежностей

Крок 3. Запустити серверний застосунок. Перейти до папки з файлом app.py , як зображено на рис. 3.10 та виконати:

python app.py

```
C:\Users\CyberYou>cd C:\Диплом

C:\Диплом>python app.py
 Веб-платформа підтримки студентів запущена!
 API: http://localhost:5000
 * Serving Flask app 'app'
 * Debug mode: on
```

Рисунок 3.10 - Перехід на папку app.py та запущення сервера

Після успішного запуску в терміналі відобразиться повідомлення рис. 3.11 :

```
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 Веб-платформа підтримки студентів запущена!
 API: http://localhost:5000
 * Debugger is active!
 * Debugger PIN: 401-566-417
```

Рисунок 3.11 - Успішно запущено сервер

Крок 4. Відкрити файл index.html у веб-браузері. Платформа готова до роботи рис. 3.12.

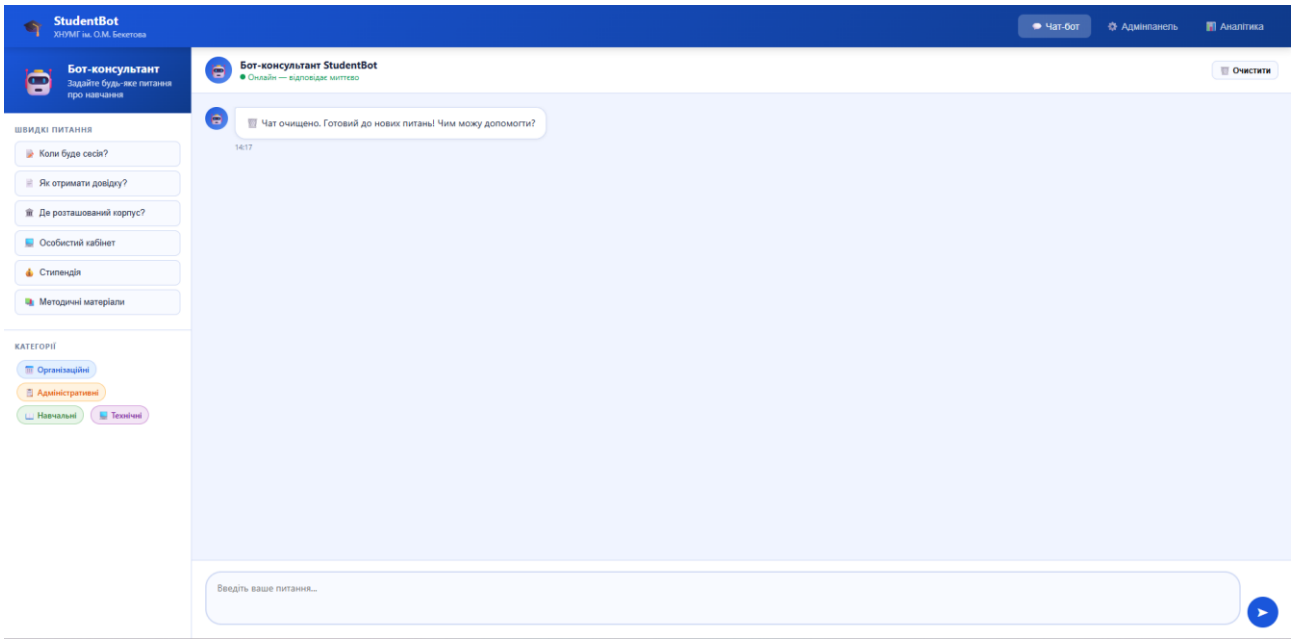

Рисунок 3.12 - Платформа для студента з чат-ботом

Робота з чат-ботом (для студента). Після відкриття веб-платформи активною є вкладка «Чат». Інтерфейс складається з трьох зон рси. 3.13 : бічна панель із категоріями та швидкими запитами, основна область переписки та поле введення повідомлення.

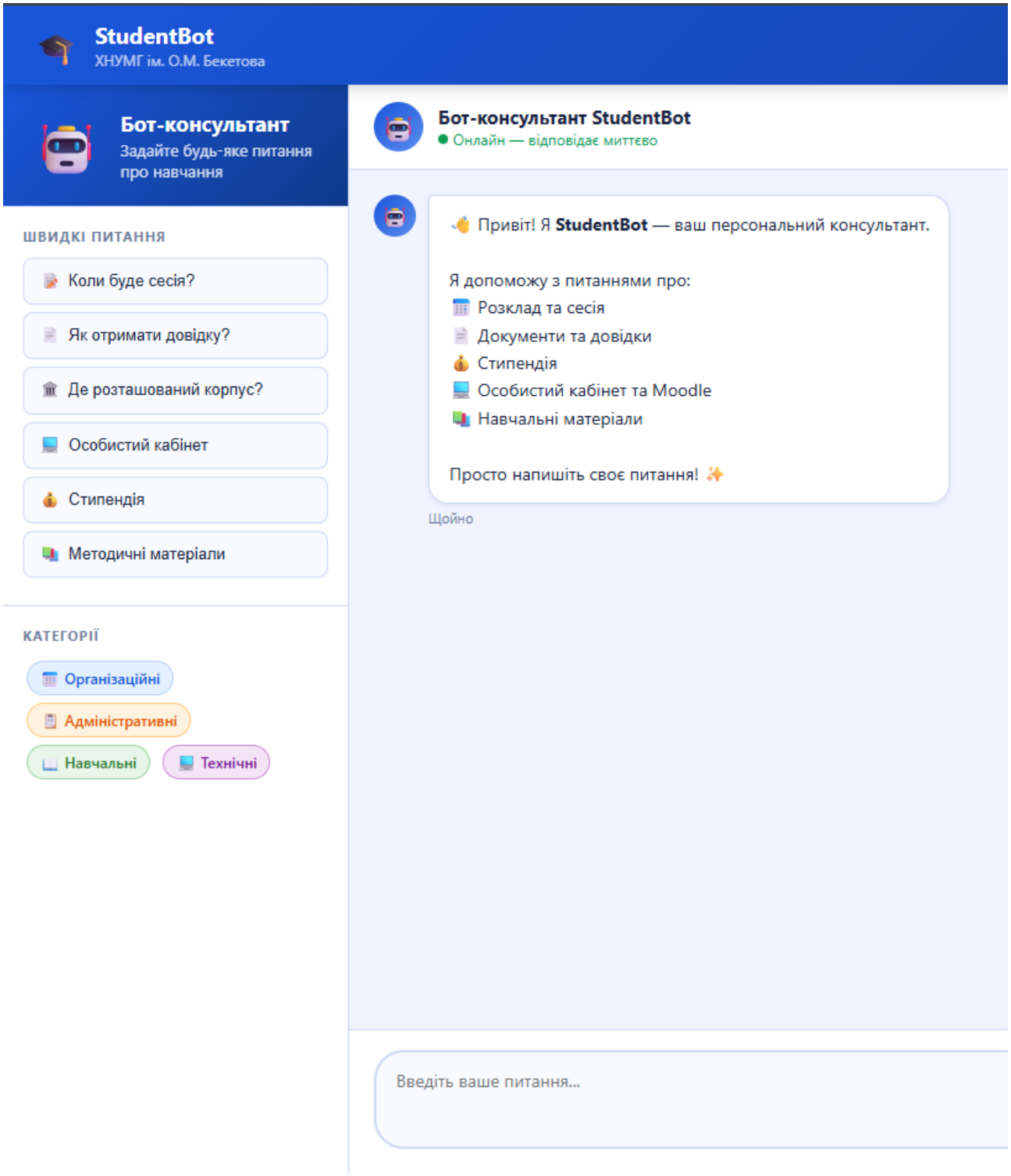

Рисунок 3.13 - Інтерфейс веб-платформи

Для початку роботи необхідно ввести запит у текстове поле внизу сторінки та натиснути кнопку відправлення або клавішу Enter. Повідомлення відображається у полі переписки у вигляді бульбашки праворуч. Протягом 1–2 секунд система опрацьовує запит (відображається анімований індикатор друку) та надає відповідь ліворуч рис.3.14:

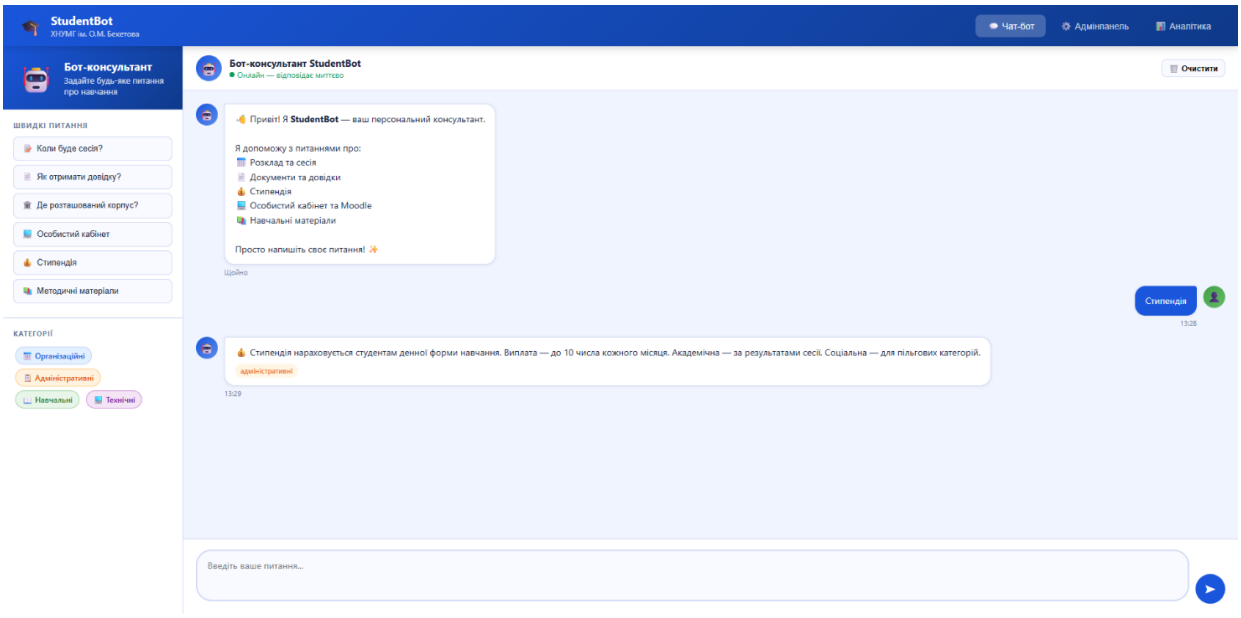

Рисунок 3.14 - Чат студента з чат-ботом

У разі, якщо запит розпізнано,як показує рис. 3.15, система повертає готову відповідь із базою знань. До відповіді може бути додано тематичну мітку-категорію (наприклад, «організаційні» або «технічні») та гіперпосилання на відповідний ресурс університету. Якщо запит не розпізнано однозначно, чат-бот пропонує уточнюючі варіанти у вигляді клікабельних чіпів - студент може обрати найближчу тему одним дотиком.

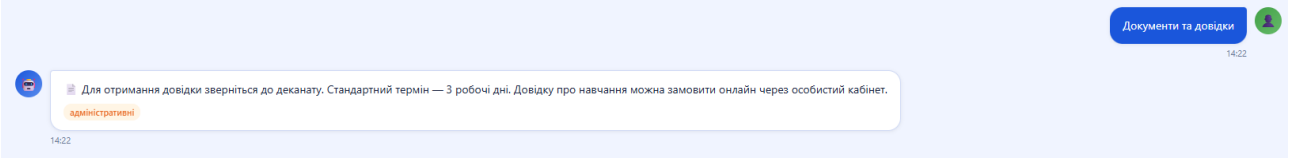

Рисунок 3.15 - Відповідь на запит студента

На бічній панелі розміщено кнопки швидких запитів для найпоширеніших тем (розклад, сесія, стипендія, Moodle) рис. 3.16. Натискання на кнопку автоматично підставляє відповідний текст у поле введення та відправляє його. У нижній частині бічної панелі відображаються кольорові значки категорій - їх можна натискати для фільтрації базових питань.

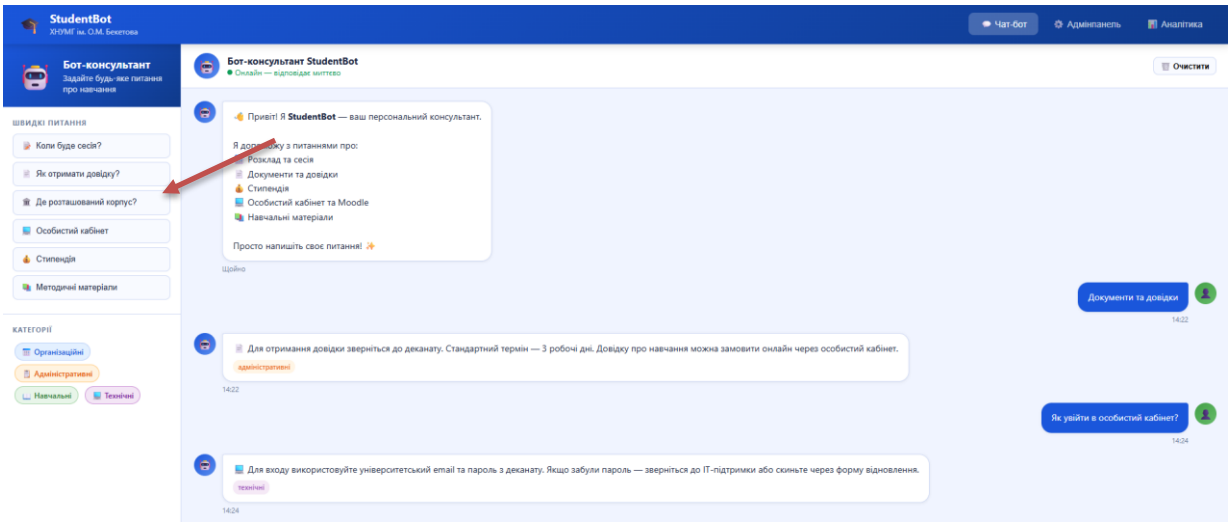

Рисунок 3.16 - Бічна панель з швидкими запитами

Для очищення поточного діалогу передбачено кнопку «Новий чат» у верхній частині інтерфейсу, на рис. 3.17 видно , де саме знаходиться кнопка, яка скидає ідентифікатор сесії та очищає область переписки.

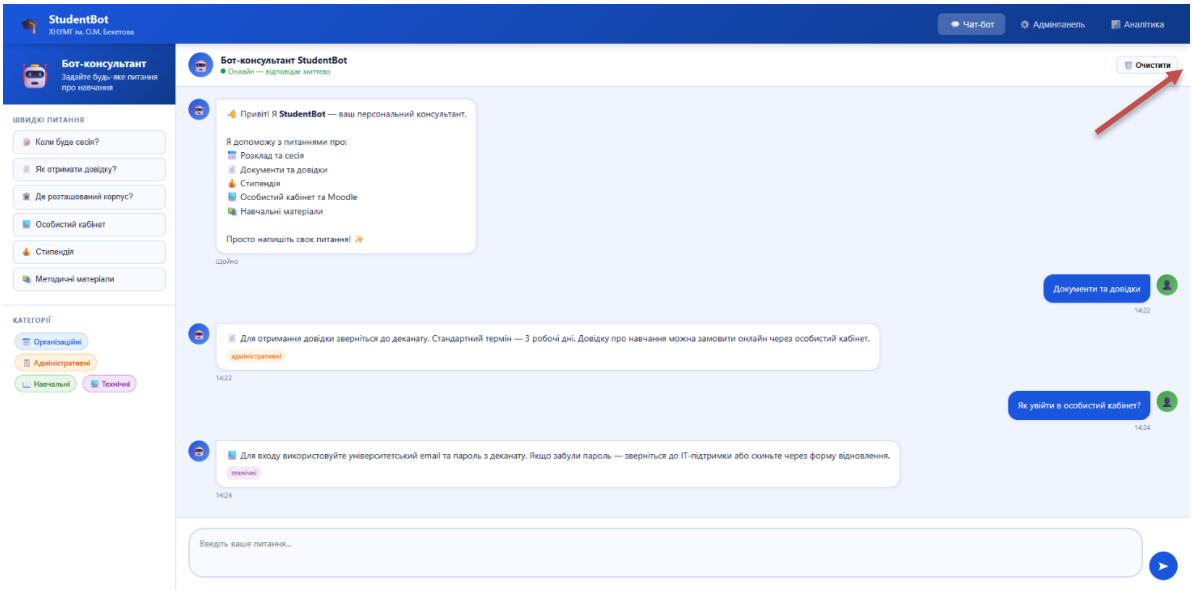

Рисунок 3.17 - Очищення діалогу з Бот-консультантом

Робота з адміністративною панеллю. Для переходу до адміністративних функцій необхідно натиснути вкладку «Адмін», рис. 3.18 показує, де треба натискати у верхній навігаційній панелі. Адміністративний розділ містить дві секції: статистику та управління базою знань.

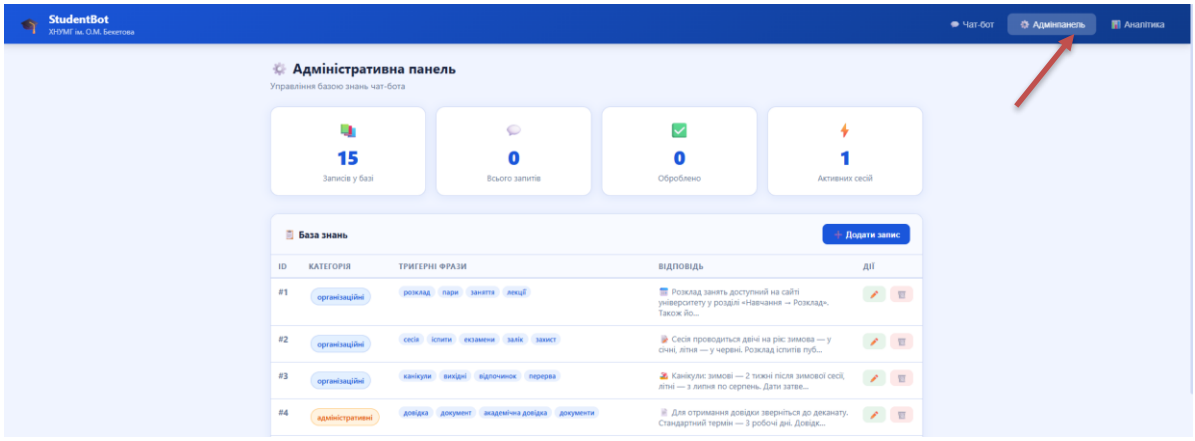

Рисунок 3.18 - Робота з адміністративною панеллю

Секція управління базою знань дозволяє переглядати всі записи у вигляді карток. Кожна картка відображає категорію, перелік тригерних фраз та текст відповіді. Доступні операції: додавання нового запису (кнопка «Додати запис») рис.3.19, редагування та видалення існуючих. При додаванні запису необхідно вказати категорію, тригерні слова (через кому) та текст відповіді.

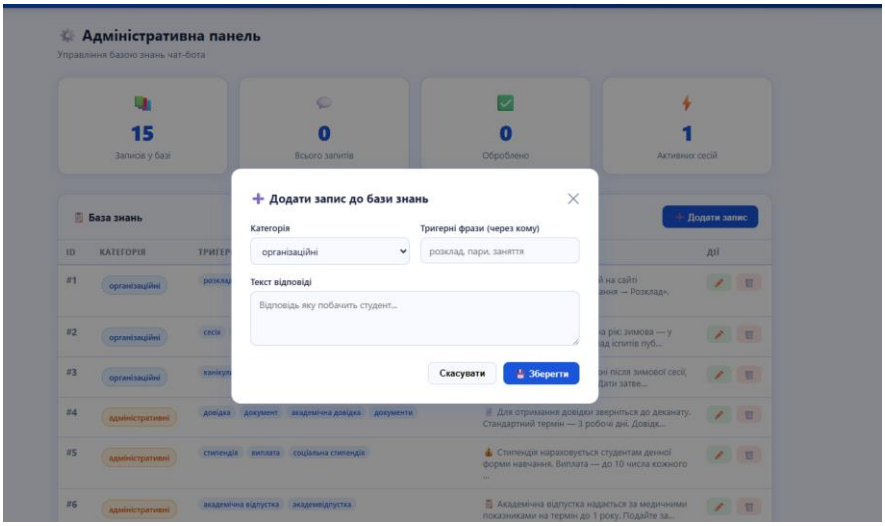

Рис. 3.19 Додавання записів до бази знань

Секція статистики відображає в режимі реального часу загальну кількість звернень до системи, кількість успішно оброблених та необроблених запитів, а також розподіл запитів за тематичними категоріями,як ми бачимо на рис. 3.20.

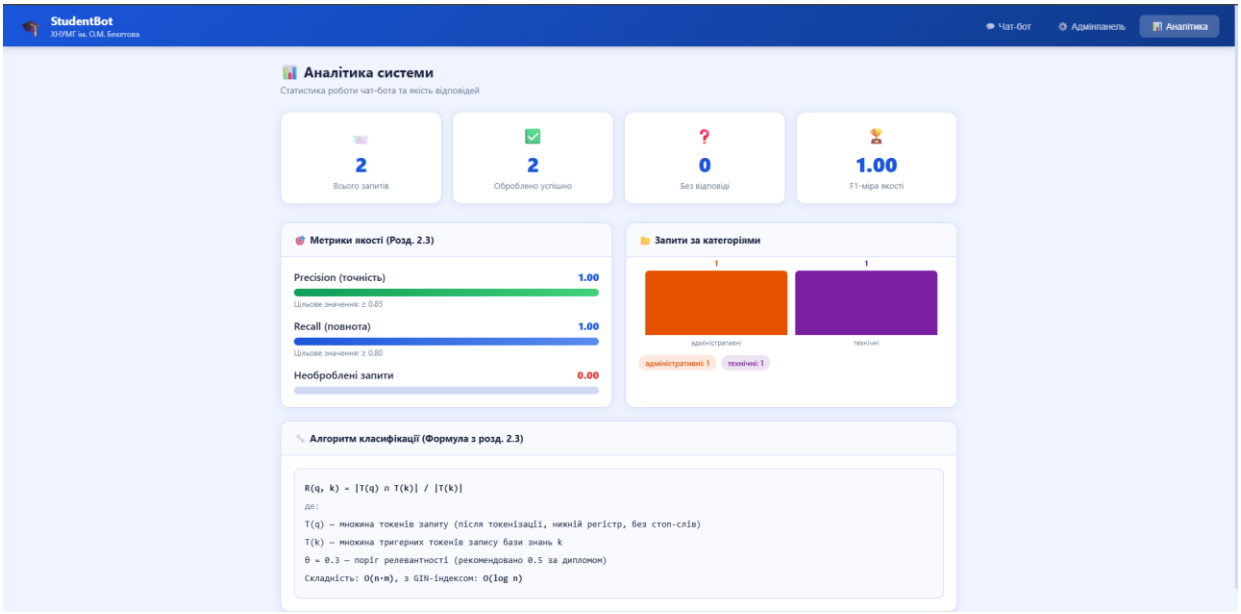

Рисунок 3.20 - Відображення статистики звернесь до бота-консультанта

Висновки до розділу

У третьому розділі описано програмне та технічне забезпечення розробленої веб-платформи підтримки студентів. Визначено стек технологій: Python 3.11 і Flask на серверній стороні та нативні HTML5/CSS3/JavaScript на клієнтській. Обраний набір інструментів забезпечує низький поріг розгортання, відсутність важких зовнішніх залежностей та достатню продуктивність для прототипного і продуктивного середовищ.

Сформульовано мінімальні вимоги до технічного забезпечення: будь-який сервер з Python 3.9+ і 512 МБ оперативної пам'яті для бекенду та сучасний браузер для кінцевого користувача. Відсутність вимоги до СУБД у прототипній версії спрощує розгортання та знижує ризики конфігурації

середовища.

Детально описано програмну реалізацію обох компонентів системи з наведенням ключових фрагментів коду. Серверна частина реалізує алгоритм токенізації україномовного тексту, обчислення релевантності запитів та RESTful API із сімома ендпоінтами. Клієнтська частина забезпечує односторінковий інтерфейс із чатом, аналітичною панеллю та CRUD-інтерфейсом для управління базою знань.

Наведено керівництво користувача, що охоплює кроки встановлення системи, сценарії роботи студента з чат-ботом (включно з механізмом уточнюючих підказок) та адміністративні функції управління базою знань і перегляду аналітики. Розроблена платформа є готовою до дослідного впровадження та може бути розширена за рахунок підключення NLP-сервісу або СУБД без зміни загальної архітектури.

РОЗДІЛ 4 ОХОРОНА ПРАЦІ

3.1 Організаційно-правові основи забезпечення безпеки праці

Збереження здоров'я та забезпечення безпеки персоналу в умовах трудової діяльності належить до числа ключових пріоритетів як державної політики, так і внутрішньої стратегії кожного підприємства. Для фахівців у галузі інформаційних технологій ця проблема набуває особливої специфіки: тривала робота за комп'ютерним обладнанням формує характерний комплекс ризиків — статичне м'язове навантаження, надмірна когнітивна напруга, обмежена рухова активність, вплив електромагнітного випромінювання та несприятливі мікрокліматичні умови на робочому місці [22].

Ключова мета охорони праці полягає у формуванні виробничих умов, що виключають або зводять до прийнятного мінімуму ймовірність виникнення нещасних випадків і профзахворювань. Практична реалізація цієї мети здійснюється через дотримання нормативних вимог, використання засобів захисту, систематичне навчання персоналу та безперервний моніторинг стану виробничого середовища [26].

Правовою основою системи охорони праці в Україні є Закон України «Про охорону праці», який закріплює пріоритет безпеки людини над виробничими показниками, визначає обов'язки роботодавця щодо забезпечення здорового середовища на робочому місці та встановлює механізми управління виробничими ризиками [23]. Законодавча концепція ґрунтується на принципі невідчужуваності права працівника на безпечну працю незалежно від економічних обставин.

Регуляторна база охорони праці не обмежується профільним законом: до неї входять норми Кодексу законів про працю, Кодексу цивільного захисту України, а також система галузевих і міжгалузевих нормативних документів — санітарних норм, будівельних правил та технічних регламентів [24].

Гігієнічні вимоги до параметрів мікроклімату [28], освітленості [29], рівня електромагнітного випромінювання [30] та організації робочих місць з відеодисплейними терміналами зафіксовані в окремих державних санітарних нормах і стандартах.

Сучасна концепція управління безпекою праці орієнтована на проактивне виявлення й оцінювання ризиків, а не на реагування на факти вже вчинених нещасних випадків. Такий підхід реалізовано в міжнародному стандарті ISO 45001, що визначає вимоги до систем менеджменту безпеки й гігієни праці [25] [31]. Його впровадження дозволяє підприємствам систематично виявляти небезпеки, кількісно оцінювати їх і цілеспрямовано знижувати рівень виробничих ризиків.

Для розробників програмного забезпечення безпечна організація праці передусім означає правильне облаштування робочого простору: раціональне розміщення монітора, налаштування освітлення відповідно до нормативів, підтримання комфортного мікроклімату та дотримання регламентованих перерв для відновлення зорової й розумової активності. Дотримання цих вимог не лише захищає здоров'я фахівця, а й безпосередньо підвищує якість та продуктивність його роботи.

Отже, охорона праці є невід'ємним елементом організації трудового процесу у сфері ІТ. Належне функціонування системи безпеки сприяє формуванню здорового виробничого середовища, зниженню ризиків для персоналу та утердженню культури безпеки в організації.

3.2 Характеристика об'єкта та виявлення потенційних небезпек

Об'єктом аналізу в цьому розділі є умови праці розробника веб-платформи підтримки студентів із вбудованим чат-ботом-консультантом. Саме тут здійснюється проектування архітектури системи, написання програмного коду, тестування функціональних компонентів, адміністрування бази даних і перевірка коректності роботи всіх модулів платформи.

Трудова діяльність провадиться в офісному приміщенні, обладнаному стаціонарним комп'ютером або ноутбуком, периферійними пристроями, засобами зв'язку та широкосмуговим інтернет-доступом. Робоче місце включає письмовий стіл, крісло з регульованими параметрами, монітор, клавіатуру, маніпулятор типу «миша» та допоміжне обладнання. Приміщення забезпечене системами електропостачання, освітлення та вентиляції.

До кола функціональних обов'язків фахівця входять: аналіз вимог до програмного продукту, написання й рефакторинг коду, адміністрування СУБД, тестування та налагодження застосунку, ведення технічної документації й участь у дистанційній взаємодії з командою.

Хоча розробка програмного забезпечення формально не належить до категорії підвищено небезпечних видів праці, виробниче середовище, у якому перебуває фахівець, містить сукупність потенційних ризиків — як специфічних для роботи з комп'ютерною технікою, так і загальновиробничих за своєю природою.

Серед фізичних чинників виокремлюються: недостатня або надмірна освітленість [29], відхилення мікрокліматичних параметрів від нормативних значень [28], інтенсивне навантаження на зоровий аналізатор, небезпека ураження електричним струмом за несправності обладнання, а також ризик пожежі внаслідок теплового перевантаження мережі або короткого замикання.

Серед психофізіологічних чинників особливо значущими є: тривале перебування в одній позі, виконання одноманітних операцій, інтенсивне когнітивне навантаження та необхідність дотримуватись жорстких дедлайнів. Накопичений вплив цих факторів може проявлятися у загальній стомлюваності, зниженні продуктивності, захворюваннях опорно-рухового апарату та симптомах емоційного вигорання.

Окремий рівень ризиків зумовлений триваючим воєнним станом в Україні: загроза ракетних ударів і необхідність оперативної евакуації в захисні споруди становлять реальну небезпеку для фізичного стану персоналу.

На підставі проведеного аналізу сформовано зведений реєстр

потенційних небезпек, представлений у табл. 3.1.

Таблиця 3.1 – Реєстр потенційних небезпек стосовно об'єкта проектування

№	Вид небезпеки	Фактор виникнення	Потенційні наслідки
1	Зорове перенавантаження	Безперервна робота з екраном впродовж тривалого часу	Зниження гостроти зору, синдром сухого ока, цефалгія
2	М'язово-скелетне навантаження	Тривале утримання робочої пози без рухової активності	Больовий синдром у хребті та шийному відділі, деформація постави
3	Нервово-психічне напруження	Складні когнітивні задачі, щільний графік, часові обмеження	Хронічний стрес, зниження мотивації, емоційне виснаження
4	Електротравматизм	Дефекти ізоляції кабелів, несправний стан техніки	Електричні ураження різного ступеня тяжкості
5	Пожежонебезпека	Коротке замикання, тепловий перегрів електрообладнання	Матеріальні збитки, загроза здоров'ю та життю персоналу
6	Дефіцит або надлишок освітлення	Неоптимальна розстановка джерел світла	Зорова втома, погіршення концентрації уваги
7	Мікрокліматичний дискомфорт	Неправильний повітрообмін, відмова кліматичного обладнання	Погіршення самопочуття, зниження працездатності
8	Воєнна небезпека	Ракетні атаки, сигнали повітряної тривоги	Фізичне каліцтво або загибель під час обстрілів
9	Ризик падіння	Безладне розміщення кабельної продукції на підлозі	Механічні травми тіла

Результати аналізу засвідчують, що переважна більшість виявлених небезпек піддається нейтралізації або суттєвому зниженню завдяки організаційним, технічним і профілактичним заходам. Для визначення пріоритетності втручань необхідно кількісно оцінити ймовірність реалізації кожного ризику та тяжкість можливих наслідків, що є предметом наступного підрозділу.

3.3 Дослідження ризику реалізації потенційних небезпек та розробка заходів щодо їх попередження

Оцінювання виробничих ризиків є обов'язковою складовою сучасного менеджменту безпеки праці та інструментом обґрунтованого прийняття управлінських рішень [26] [27]. Її мета — виявлення реальних загроз для здоров'я персоналу, визначення ступеня їх критичності та вироблення заходів з мінімізації шкідливих впливів.

Методологія оцінювання охоплює такі послідовні кроки: встановлення переліку небезпек, з'ясування умов їх виникнення, визначення ймовірності настання небезпечної події, оцінку тяжкості потенційних наслідків, розрахунок загального рівня ризику та розроблення профілактичних заходів. Отримані результати слугують основою для вдосконалення умов праці та підвищення безпеки виробничого процесу.

У цьому дослідженні застосовано матричний метод оцінювання ризиків, що ґрунтується на поєднанні двох параметрів:

$$R=P\times S \qquad (4.1)$$

ймовірність виникнення небезпечної події (P);
тяжкість можливих наслідків (S).

Для оцінювання використано п'ятибальну шкалу.
Градація ймовірності (P):

1 – подія практично неможлива;

2 – виникнення малоймовірне;

3 – помірна ймовірність;

4 – висока ймовірність;

5 – подія майже невідворотна.

Градація тяжкості наслідків (S):

1 – мінімальний вплив на здоров'я;

2 – незначні травми або розлади;

3 – порушення середнього ступеня;

4 – серйозне ураження;

5 – летальний результат.

Інтегральний показник ризику визначається за формулою: $R = P \times S$, де R – рівень ризику; P – ймовірність настання небезпеки; S – тяжкість наслідків.

Детальний аналіз виконано для п'яти найбільш характерних небезпечних факторів умов праці розробника веб-платформи.

1. Зорове перенавантаження

Діяльність програміста нерозривно пов'язана з безперервним зоровим контактом з екраном. Тривале фокусування без достатніх перерв спричиняє прогресуючу астенопію, синдром сухого ока та цефалгію [22].

Ймовірність виникнення: $P = 5$

Тяжкість наслідків: $S = 2$

Рівень ризику: $R = 5 \times 2 = 10$

Ризик кваліфікується як середній, що вимагає впровадження превентивних заходів.

2. М'язово-скелетне навантаження

Тривале перебування в статичній позі за відсутності рухової активності негативно позначається на стані хребта та суглобів. У перспективі це здатне спричинити хронічні захворювання опорно-рухової системи.

Ймовірність виникнення: $P = 4$

Тяжкість наслідків: $S = 3$

Рівень ризику: R = 4 × 3 = 12

Ризик оцінюється як підвищений.

3. Нервово-психічне напруження

Виконання технічно складних завдань в умовах стислих термінів та необхідності підтримувати тривалу концентрацію є передумовою для накопичення стресу й розвитку синдрому хронічної втоми.

Ймовірність виникнення: P = 4

Тяжкість наслідків: S = 3

Рівень ризику: R = 12

Ризик належить до категорії підвищених.

4. Ураження електричним струмом

До основних причин електротравматизму відносяться пошкодження ізоляції кабелів, застосування технічно несправного обладнання та порушення вимог електробезпеки [9].

Ймовірність виникнення: P = 2

Тяжкість наслідків: S = 5

Рівень ризику: R = 10

Ризик є середнім, проте через потенційну летальність потребує підвищеного контролю.

5. Воєнна загроза та необхідність евакуації

Поточна воєнна ситуація в Україні зумовлює реальну небезпеку позаштатних ситуацій, пов'язаних із повітряними атаками й необхідністю укриття персоналу.

Ймовірність виникнення: P = 3

Тяжкість наслідків: S = 5

Рівень ризику: R = 15

Цей показник відповідає категорії високого ризику, що вимагає негайного опрацювання та впровадження заходів реагування.

Проведений аналіз виявив, що найвищий ступінь небезпеки становлять воєнні ризики, а також хронічні психофізіологічні навантаження, притаманні

роботі у сфері програмної інженерії.

Таблиця 3.2 – Комплекс заходів щодо зниження виявлених ризиків

№	Небезпечний фактор	Рекомендовані заходи	Прогнозований ефект
1	Зорове перенавантаження	Техніка «20–20–20», якісне підсвічування монітора, антиблікове покриття	Профілактика розладів зору, зниження втоми очей
2	М'язово-скелетне навантаження	Ергономічні меблі з можливістю регулювання, перерви з фізичними вправами	Зменшення ризику захворювань опорно-рухової системи
3	Нервово-психічне напруження	Грамотне планування навантаження, підтримка психологічного мікроклімату	Зниження рівня стресу, запобігання вигоранню
4	Електротравматизм	Технічне обслуговування обладнання, дотримання вимог електробезпеки	Зведення до мінімуму ймовірності травмування
5	Пожежонебезпека	Вогнегасники, справна проводка, інструктажі з пожежної безпеки	Підвищення рівня протипожежного захисту
6	Воєнна небезпека	Ознайомлення з планом евакуації, безперешкодний доступ до укриттів	Захист персоналу під час надзвичайних ситуацій

Запропоновані заходи охоплюють організаційний, технічний і

профілактичний рівні управління ризиками. Їх систематичне впровадження дозволяє суттєво знизити рівень шкідливих впливів на персонал і підвищити загальну безпечність робочого середовища.

Висновки до розділу

У даному розділі розглянуто проблематику охорони праці стосовно умов роботи розробника веб-платформи підтримки студентів. Проаналізовано нормативно-правову базу та визначено роль системи безпеки праці у збереженні здоров'я персоналу й попередженні виробничого травматизму.

ВИСНОВКИ

У межах кваліфікаційної роботи бакалавра досягнуто поставленої мети дослідження - спроектовано та розроблено комплексну веб-платформу підтримки студентів, ключовим елементом якої є чат-бот консультант, здатний оперативно обробляти запити користувачів та забезпечувати зручний доступ до навчально-адміністративної інформації. Усі завдання, сформульовані у вступі, виконано в повному обсязі.

У першому розділі проведено комплексний аналіз предметного середовища та визначено специфіку інформаційної взаємодії в закладі вищої освіти, виявлено основні категорії запитів студентів та побудовано функціональну модель платформи. Порівняльний аналіз п'яти наявних аналогів - Ivy.ai, Ada, Tidio, Moodle та Dialogflow - підтвердив високу вартість і обмежену адаптивність зарубіжних рішень та довів доцільність розробки власної платформи, орієнтованої на специфіку вітчизняного ЗВО.

У другому розділі спроектовано реляційну базу даних у третій нормальній формі, що охоплює шість взаємопов'язаних сутностей, та побудовано об'єктно-орієнтовану модель серверної частини системи із застосуванням патернів проектування Strategy та Observer. Розроблено й математично обґрунтовано алгоритм класифікації запитів на основі зваженого коефіцієнта релевантності; застосування повнотекстового індексу GIN знижує часову складність пошуку до O(log n), що забезпечує формування відповіді в режимі реального часу.

У третьому розділі реалізовано програмний продукт на основі стека технологій Python 3.11 та Flask на серверній стороні й HTML5/CSS3/JavaScript на клієнтській. Розроблено RESTful API із сімома ендпоінтами, односторінковий веб-інтерфейс із чатом, аналітичною панеллю та CRUD-модулем управління базою знань, а також складено керівництво користувача, що підтверджує готовність платформи до дослідного впровадження.

У четвертому розділі досліджено питання охорони праці розробника

веб-платформи: проаналізовано нормативно-правову базу безпеки праці, виявлено потенційні шкідливі й небезпечні фактори робочого місця та здійснено кількісну оцінку ризиків методом матриці для п'яти пріоритетних факторів. На основі результатів оцінювання розроблено комплекс організаційних, технічних і профілактичних заходів, спрямованих на зниження рівня ризиків і підвищення безпеки трудового середовища.

Таким чином, розроблена веб-платформа з чат-ботом консультантом повністю відповідає поставленим вимогам, є функціонально завершеною та придатною для впровадження в освітній процес закладу вищої освіти. Практичне значення роботи полягає у скороченні часу очікування відповіді на типові звернення студентів, зниженні навантаження на адміністративний персонал та підвищенні доступності навчально-адміністративної інформації у режимі 24/7. Перспективним напрямом подальшого розвитку системи є інтеграція сервісу обробки природної мови для розширення можливостей розпізнавання запитів і перехід від прототипної реалізації до повноцінної системи управління базами даних, придатної до промислової експлуатації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Роман, С., & Лариса, К. (2024). Використання чат-ботів для взаємодії студентів із викладачами: виклики та можливості у контексті питань доступності. Zenodo. https://zenodo.org/records/10812853

2. Чат-боти як засіб автоматизації освітніх та адміністративних процесі - Львівська політехніка. Порівняння скриптових та ШІ-орієнтованих ботів для автоматизації адміністративних процесів. https://ena.lpnu.ua/bitstreams/5d17adb3-837e-4b21-a59f-b1c2c0007129/download

3. Методика використання чат-ботів у керуванні дослідницькою діяльністю студентів https://journals.academ.vinnica.ua/index.php/ped-psyh/article/view/274/261

4. Гармата О.С.. , Костенко О.Б. Чат-боти як інструмент цифрової трансформації освіти: досвід розробки та впровадження / матеріали II (VIII) Міжнародної науково-практичної конференції здобувачів вищої освіти і молодих учених «Інформаційні технології: теорія і практика», С. 130-132

5. Жихорська О. Використання чат-ботів у навчанні студентів. Вісник Київського національного університету імені Тараса Шевченка. Серія «Педагогіка». 2023. № 2(18). DOI: 10.17721/2415-3699.2023.18.06. https://pedvisnyk.knu.ua/index.php/pedagogy/article/view/385/520

6. Громик А. П., Бондар С. А. Використання чат-ботів зі штучним інтелектом для покращення комунікації та підтримки здобувачів освіти в мобільних застосунках для дистанційного навчання. Педагогічна Академія: наукові записки. 2025. DOI: 10.33407/pedacad.2025.824. https://eprints.zu.edu.ua/43381/1/1.pdf

7. Шахіна І. Ю., Подзигун О. А. Цифрова трансформація освіти: інтеграція штучного інтелекту в освітню екосистему ЗВО. Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців. 2026. № 79. С. 166–179.

8. Цифровізація освітнього процесу в закладах вищої освіти в умовах воєнного стану. Освітні обрії. 2023. URL:

https://journals.pnu.edu.ua/index.php/obrii/article/view/7090 (дата звернення: 20.04.2026).

9. Цифровізація освітнього процесу в умовах дистанційного навчання в закладах вищої освіти. Наукові записки. Серія: Педагогічні науки. 2021. URL: https://pednauk.cusu.edu.ua/index.php/pednauk/article/view/1174 (дата звернення: 20.04.2026).

10. Цифровізація освіти в Україні: технології та методики навчання. Трансформаційна економіка. 2024. URL: https://transformations.in.ua/index.php/journal/article/view/107 (дата звернення: 22.04.2026).

11. Освітній чат-бот @EducationUaBot. Міністерство освіти і науки України. 2022. URL: https://mon.gov.ua/osvita-2/tsifrova-transformatsiya-osviti-i-nauki/osvitniy-chat-bot (дата звернення: 15.04.2026).

12. Мнушка О. В., Савченко В. М., Маций О. Б. Об'єктно-орієнтоване програмування мовою Python : навч. посіб. Харків : ХНАДУ, 2021. https://repository.kpi.kharkov.ua/server/api/core/bitstreams/20ad9ea5-8d59-46cf-9c97-dd146b480d32/content

13. Гришанович Т. О., Глинчук Л. Я. Основи об'єктно-орієнтованого програмування : навч. посібник. Луцьк : ВНУ імені Лесі Українки, 2022. 120 с. https://evnuir.vnu.edu.ua/server/api/core/bitstreams/48573340-f0c0-4216-97d9-5bad9c544f24/content

14. Васильєв О. М. Програмування в Python. Теорія і практика : навч. посібник. Київ : Видавництво Ліра-К, 2023. 462 с.

15. Гайна Г. А. Основи проектування баз даних : навч. посібник. Київ : КНУБА, 2023. 220 с. https://library.nusta.edu.ua/depository/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BDi%20%D0%BA%D0%BD%D0%B8%D0%B6%D0%BA%D0%B8/%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D0%B8%20%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D

1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%20%D0%B1%D0%B0%D0%B7%20%D0%B4%D0%B0%D0%BD%D0%B8%D1%85/01.pdf

16. Босько В. В., Константинова Л. В., Марченко К. М., Улічев О. С. Web-програмування : навч. посібник. Кропивницький : ЦНТУ, 2022. 208 с. https://dspace.kntu.kr.ua/server/api/core/bitstreams/2bfb5a3c-54d9-4283-89c1-5e190e8aa151/content

17. Томка Ю. Я. Python та Django Full Stack веб-розробка : навч. посібник. 2022. 310 с.

18. Каплун В. А. Основи web-програмування. Теорія і практика. 2023. https://pdf.lib.vntu.edu.ua/books/2023/Kaplun_2023_128.pdf

19. Баран С. В. Основи web-програмування : навч. посібник. 2023. 195 с. http://www.kgemt.org.ua/pdf/about_colege/library_fund/osnovy_webprohramuvannya_navch_posib__s_v_baran_lk3xezdvds.pdf

20. Системи баз даних та знань. Книга 1. Організація баз даних та знань : підручник. Бібліотека УГІ, УАТІ, 2022.

21. Шахіна І. Ю. Інтеграція штучного інтелекту в сферу освіти: проблеми, виклики, загрози, перспективи. Modern Information Technologies and Innovation Methodologies of Education in Professional Training. 2024. № 68. DOI: 10.31652/2412-1142-2024-68.

22. Бедрій Я.І. Основи охорони праці користувачів персональних комп'ютерів : навч. посіб. – Тернопіль : Навч. книга – Богдан, 2014. – 144 с. http://pdf.lib.vntu.edu.ua/books/2021/Ivah_2010_464.pdf

23. 2. Закон України «Про охорону праці» від 14.10.1992 № 2694-XII. URL: https://zakon.rada.gov.ua/laws/show/2694-12 (дата звернення: 14.06.2026).

24. 3. Кодекс цивільного захисту України від 02.10.2012 № 5403-VI. URL: https://zakon.rada.gov.ua/laws/show/5403-17 (дата звернення: 14.06.2026).

25. 4. ДСТУ ISO 45001:2019. Системи управління охороною здоров'я та безпекою праці. Вимоги та настанови щодо застосування. – К. : ДП

«УкрНДНЦ», 2019. dstu_iso_45001_2019.pdf

26. Державна служба України з питань праці. Законодавство з охорони праці. URL: https://dsp.gov.ua/zakony-ukrainy/ (дата звернення: 14.06.2026).

27. Державна служба України з питань праці. Нормативно-правові акти та роз'яснення у сфері охорони праці. URL: https://dsp.gov.ua (дата звернення: 14.06.2026).

28. ДСН 3.3.6.042-99. Санітарні норми мікроклімату виробничих приміщень. – К. : МОЗ України, 1999. Санітарні норми мікроклімату виро... | від 01.12.1999 № 42

29. ДБН В.2.5-28:2018. Природне і штучне освітлення. – К. : Мінрегіон України, 2018. ДБН В.2.5-28-2018 "Природне і штучне освітлення" №ДБН В.2.5-28-2018

30. ДСанПіН 3.3.6.096-2002. Державні санітарні норми і правила при роботі з джерелами електромагнітних полів. – К. : МОЗ України, 2002. Про затвердження Державних саніт... | від 18.12.2002 № 476

31. ISO 45001:2018. Occupational health and safety management systems – Requirements with guidance for use. URL: https://www.iso.org/standard/63787.html (дата звернення: 14.06.2026).