МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інженерії
програмного забезпечення

_____ Євген ДАВИДЕНКО

«10» червня 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

ВЕБСАЙТ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЇ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ ПОШУКУ ФІЛЬМІВ

Спеціальність 121 Інженерія програмного забезпечення
Освітня програма «Інженерія програмного забезпечення»

Здобувачка _____ **Олександра МАР'ЯНКО**

«10» червня 2025 р.

Керівник роботи
канд. техн. наук,
доцент _____ **Євген ДАВИДЕНКО**

«10» червня 2025 р.

Миколаїв – 2025

Завдання на виконання кваліфікаційної роботи

Чорноморський національний університет імені Петра Могили

Факультет	Комп'ютерних наук
Кафедра	Інженерії програмного забезпечення
Рівень вищої освіти	Перший (бакалаврський)
Освітній ступінь	Бакалавр
Спеціальність	121 Інженерія програмного забезпечення
Освітня програма	Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри інженерії
програмного забезпечення
_____ Євген ДАВИДЕНКО
«27» січня 2025 р.

ЗАВДАННЯ

на кваліфікаційну бакалаврську роботу здобувачки

Мар'янко Олександри

1. Тема кваліфікаційної роботи «Вебсайт з використанням технології штучного інтелекту для пошуку фільмів» затверджена наказом ректора ЧНУ ім. Петра Могили № 315 від «13» листопада 2024 р.
2. Строк представлення кваліфікаційної роботи «18» червня 2025 р.
3. Очікуваний результат роботи та початкові дані, якщо такі потрібні

Очікуваним результатом є вебсайт для пошуку фільмів

4. Перелік питань, що підлягають розробці:

– дослідження предметної області;
– огляд та аналіз існуючих аналогів;
– формування специфікації вимог;

– проєктування програмного забезпечення;
– розробка програмного забезпечення;
– опис керівництва користувача.

5. Перелік графічних матеріалів:

Презентація

6. Консультанти:

Консультант	**Кафедра (організація)**	**Частина роботи**

Дата видачі завдання «27» січня 2025 р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: **Вебсайт з використанням технології штучного інтелекту для пошуку фільмів**

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КБР	27.01.2025	01.02.2025	Виконано
2.	Огляд літератури за темою роботи	24.02.2025	09.03.2025	Виконано
3.	Складання календарного плану КБР	18.03.2025	19.03.2025	Виконано
4.	Аналіз предметної області	24.03.2025	30.03.2025	Виконано
5.	Розробка проєктних рішень	31.03.2025	06.04.2025	Виконано
6.	Моделювання та конструювання ПЗ	31.03.2025	09.04.2025	Виконано
7.	Кодування, тестування розробленого ПЗ, розробка керівництва користувача	09.04.2025	01.06.2025	Виконано
8.	Оформлення КБР та презентації	28.04.2025	26.05.2025	Виконано
9.	Попередній захист	28.05.2025	28.05.2025	Виконано
10.	Відгук керівника КБР	09.06.2025	13.06.2025	Виконано
11.	Рецензування	09.06.2025	16.06.2025	Виконано
12.	Завершення оформлення КБР та презентації	02.06.2025	13.06.2025	Виконано
13.	Захист кваліфікаційної роботи	23.06.2025	23.06.2025	Виконано

Здобувачка _____ **Олександра МАР'ЯНКО**

«19» березня 2025 р.

Керівник роботи

канд. техн. наук,

доцент _____ **Євген ДАВИДЕНКО**

«19» березня 2025 р.

АНОТАЦІЯ

до кваліфікаційної бакалаврської роботи

Вебсайт з використанням технології штучного інтелекту для пошуку фільмів

Здобувачка 409 гр.: Мар’янко Олександра

Керівник: канд. техн. наук, доцент Давиденко Євген

Кваліфікаційна бакалаврська робота присвячена розробці вебсайту, який використовує технології штучного інтелекту для ефективного пошуку фільмів, а також персоналізації контенту за допомогою алгоритмів, які аналізують інтереси користувача, його історію переглядів та запити.

Актуальність теми обумовлена стрімким зростанням кількості медіаконтенту в Інтернеті та труднощами його ефективного пошуку. Традиційні фільтри не завжди відповідають потребам користувача, тоді як персоналізовані рекомендації, створенні з використанням штучного інтелекту, дають змогу підбирати точний та задовільний контент.

Об’єктом роботи є процес пошуку фільмів на основі аналізу вподобань користувачів.

Предметом кваліфікаційної роботи є технології штучного інтелекту, що використовуються у вебсайтах для взаємодії з користувачем, аналізу вподобань та надання персоналізованих рекомендацій.

Метою роботи є розробка вебсайту для пошуку фільмів з використанням технології штучного інтелекту із зручним вибором контенту і отриманням персоналізованих рекомендацій для користувачів.

Кваліфікаційна робота складається із вступу, чотирьох розділів, висновків та переліку джерел посилання.

У вступі визначено актуальність обраної теми, мета і завдання роботи, а також об’єкт та предмет дослідження.

У першому розділі проведено аналіз предметної області, розкрито об’єкт та предмет дослідження. Описано функціональні особливості процесу пошуку фільмів. Здійснено огляд та аналіз сучасних програмних рішень у цій сфері.

Другий розділ присвячено пошуку та опису методів і технологій, що використовуються. Сформовано специфікацію вимог до програмного забезпечення, що розробляється.

У третьому розділі описано архітектуру програмного забезпечення, проведено огляд технологій та мов програмування, що використовуються. Наведено UML-діаграми, що моделюють роботу вебсайту.

У четвертому розділі описано процес створення програмного забезпечення. Сформовано керівництво користувача. Представлено результати розробки вебзастосунку.

У висновках проведено аналіз роботи та отриманих результатів.

Кваліфікаційна робота викладена на 61 сторінці машинописного тексту, складається із вступу, 4 розділів, загальних висновків та переліку джерел посилання з 20 найменувань. Праця містить 9 таблиць та 28 рисунків.

Ключові слова: вебсайт, віртуальний помічник, персоналізовані рекомендації, пошук фільмів, штучний інтелект, OpenAI API, PostgreSQL, React, Spring Boot, TMDb API.

ABSTRACT

to the qualifying bachelor's thesis

Website using artificial intelligence technology for movie search

Student of 409 group: Maryanko Oleksandra

Supervisor: Candidate of Technical Sciences, Associate Professor Davydenko Yevhen

This work is dedicated to developing a website that uses artificial intelligence technologies for efficient movie search and content personalization through algorithms that analyze user interests, viewing history and requests.

The relevance of the topic is due to the rapid growth of media content on the Internet and the difficulty of effectively searching through it. Traditional filters do not always meet the user`s needs, whereas personalized recommendations created using artificial intelligence make it possible to select accurate and satisfactory content.

The object of the work is the process of searching for movies based on user preferences analysis.

The subject of the qualification work is artificial intelligence technologies used on websites for user interaction, preference analysis, and providing personalized recommendations.

The purpose of this work is to develop a website for movie search using artificial intelligence technology, enabling convenient content selection and personalized recommendations for users.

The qualification work consists of an introduction, four sections, conclusions and a list of references.

The introduction defines the relevance of the chosen topic, the goal and objectives of the work, as well as the object and subject of the research.

The first section analyzes the subject area, revealing the object and subject of the research. It describes the functional features of the movie search process and provides an overview and analysis of current software solutions in this field.

The second section is dedicated to the search and description of the methods and technologies used. The software requirements specification is formulated.

The third section describes the software architecture, reviews the technologies and programming languages used and provides UML diagrams modeling the website's operation.

The fourth section describes the software development process. A user guide has been created. The results of the web application development are presented.

The conclusions provide an analysis of the work and the obtained results.

The qualification work is presented on 61 pages of typewritten text, consists of an introduction, 4 sections, general conclusions and a list of references with 20 titles. The work contains 9 tables and 28 figures.

Keywords: artificial intelligence, movie search, OpenAI API, personalized recommendations, PostgreSQL, React, Spring Boot, TMDb API, virtual assistant, website.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

КБР	–	Кваліфікаційна бакалаврська робота
ПЗ	–	Програмне забезпечення
ШІ	–	Штучний інтелект
AI	–	Artificial intelligence
API	–	Application Programming Interface
HTTPS	–	HyperText Transfer Protocol Secure
SQL	–	Structured query language
TMDb	–	The Movie Database
UML	–	Unified Modeling Language

ВСТУП

Актуальність теми кваліфікаційної бакалаврської роботи зумовлена стрімким зростанням кількості відеоконтенту в Інтернеті та складністю його ефективного пошуку. Традиційні методи пошуку, такі як вибір за жанром, роком випуску чи рейтингом, не завжди дають бажані результати. Сучасні платформи пропонують величезні бібліотеки фільмів і серіалів, серед яких користувачам стає дедалі складніше знаходити щось справді цікаве. Активно зростає популярність персоналізованих рекомендацій, оскільки вони дозволяють враховувати оцінки, вподобання та поведінку користувача для підбору відповідного контенту. Проте такі технології здебільшого використовують великі стримінгові сервіси, тоді як звичайні вебсайти для пошуку фільмів часто обмежуються лише базовими фільтрами.

Використання штучного інтелекту у вебсайтах для пошуку фільмів дозволяє створити розумні алгоритми підбору, які аналізують інтереси користувача, його історію переглядів та текстові запити, щоб формувати точніші рекомендації. Це значно покращує взаємодію з системою та робить пошук фільмів швидшим, зручнішим і приємнішим.

Метою роботи є розробка вебсайту для пошуку фільмів з використанням технології штучного інтелекту із зручним вибором контенту і отриманням персоналізованих рекомендацій для користувачів.

Для досягнення визначеної мети необхідно вирішити такі **завдання**:

– провести аналіз предметної області;
– провести огляд та аналіз існуючих аналогів вебсайтів;
– розробити специфікацію вимог до вебсайту, що розробляється;
– виконати моделювання та проєктування програмного забезпечення;
– реалізувати front-end з урахуванням зручності використання;
– розробити back-end з використанням штучного інтелекту для персоналізованого пошуку;
– описати керівництво користувача.

Об’єктом роботи є процес пошуку фільмів на основі аналізу вподобань користувачів.

Предметом кваліфікаційної роботи є технології штучного інтелекту, що використовуються у вебсайтах для взаємодії з користувачем, аналізу вподобань та надання персоналізованих рекомендацій.

Сфера застосування розробленого вебсайту охоплює групу користувачів, які шукають зручний, швидкий та персоналізований спосіб підбору фільмів для перегляду. Вебсайт може бути корисним для широкого кола глядачів, які хочуть отримувати рекомендації на основі власних вподобань. Система є гнучкою для масштабування, додавання нових API та оновлення функціоналу відповідно до потреб користувача.

Кваліфікаційна бакалаврська робота викладена на 61 сторінок, містить 4 розділи, 28 рисунків, 9 таблиці, 20 джерел в переліку посилань.

Ключові слова: вебсайт, віртуальний помічник, персоналізовані рекомендації, пошук фільмів, штучний інтелект, OpenAI API, PostgreSQL, React, Spring Boot, TMDb API.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд предметної області

У сучасному світі обсяг медіаконтенту щодня стрімко зростає. Швидкий розвиток цифрових технологій та загальна діджиталізація призвели до появи великої кількості стримінгових платформ, онлайн-кінотеатрів, соціальних мереж та інших цифрових сервісів, які пропонують широкий вибір фільмів, телесеріалів та коротких відео на будь-який смак. Це відкриває нові можливості для самовираження, культурного збагачення та дозвілля, а також дає змогу глядачам постійно знаходити для себе щось нове. Завдяки доступу до Інтернету кожен користувач може знайти саме той контент, який відповідає його потребам, інтересам або емоційному стану [1].

Однак такі можливості спричинили появу нової проблеми – перенасичення контентом. Велика кількість варіантів не завжди полегшує прийняття рішення, а навпаки, часто його ускладнює, що може призвести до людської пасивності. Це явище називається парадоксом вибору. Психолог Баррі Шварц стверджує, що на практиці людей робить щасливим не кількість доступних опцій, а здатність зробити вдалий і задовільний вибір [2].

У результаті виникає стан емоційного виснаження – фрустрація. Це емоційний стан, який виникає внаслідок неможливості задовільнити потребу або досягти бажаної мети, і найчастіше супроводжується роздратуванням, внутрішньою напругою, зниженням продуктивності або навіть агресією. Цей стан може зрештою призвести до відмови приймати будь-яке рішення. Наприклад, після тривалого перегляду рекомендацій на різних платформах, не побачивши нічого, що одразу привернуло б увагу, користувач, частіше за все, дуже швидко втрачає інтерес до перегляду [3].

Рекомендаційні системи почали активно впроваджуватись наприкінці минулого століття. Прикладом такої реалізації є платформи Amazon та Netflix, які першими використовували алгоритми колаборативної фільтрації для персоналізації. З того часу моделі рекомендацій значно розвинулися та

поширилися, що свідчить про постійне зростання інтересу до взаємодії з інформаційним простором.

Сучасні сервіси все більше використовують персоналізацію контенту, щоб позбутися інформаційного перевантаження користувачів. Застосування відповідних алгоритмів для аналізу попередньої активності, урахування індивідуальних уподобань користувача та формування особистих рекомендацій спрощує вибір. Користувач швидко отримує контент, який відповідає його інтересам. З появою технологій штучного інтелекту системи дуже стрімко розвиваються та вдосконалюються. Штучний інтелект здатний не лише враховувати поведінку та взаємодію користувача з системою, але й аналізувати контекст запитів та виявляти потребу в певному типі інформації. Все це дозволяє створювати більш динамічний, гнучкий та персоналізований контент [4].

Крім того, значну роль також відіграють простота використання та інтуїтивно зрозумілий інтерфейс. Система, яка виглядає перевантаженою та змушує користувача витрачати час на пошук потрібних функцій, швидко викликає втрату інтересу та спонукає до відмови від її використання. Тому під час розробки необхідно враховувати не лише функціональні можливості системи, а й потреби користувача. При побудові системи персоналізації швидкість прийняття рішення користувачем має велике значення. Завдяки зручному інтерфейсу користування сервісом стає ефективним та комфортним, що, своєю чергою, підвищує рівень задоволеності користувачів.

З огляду на стрімке зростання обсягів відеоконтенту та ускладнення процесу його вибору, дедалі актуальнішою стає потреба у створенні систем, здатних не лише фільтрувати інформацію, але й адаптувати її відповідно до індивідуальних потреб користувачів. Використання технологій штучного інтелекту дає змогу створювати інтелектуальні вебплатформи, які надають персоналізовані рекомендації. Інтеграція таких рішень у вебзастосунки відкриває шлях до нового рівня взаємодії з контентом, орієнтованого на ефективність персоналізації.

1.2 Огляд існуючих аналогів

Огляд вебзастосунків-аналогів є важливою складовою для аналізу. Він необхідний для вивчення сучасних підходів до реалізації подібних рішень, функціонального наповнення, виявлення переваг та недоліків конкурентів. Такий аналіз дозволяє виявити ефективні рішення, які варто адаптувати, а також дозволяє визначити можливості для створення власного вебсайту. Огляд аналогів сприяє виявленню потреб цільової аудиторії та формулюванню специфікації вимог до програмного забезпечення, що розробляється. Для аналізу було обрано три вебзастосунки: TasteDive, Movie Decider та Taste.io. Кожен із них реалізує різний підхід до формування рекомендацій і відрізняється функціональним наповненням та можливостями.

TasteDive

TasteDive (раніше відомий як TasteKid) – це вебзастосунок для пошуку нового розважального контенту на основі особистих вподобань користувача. Платформа охоплює різноманітні категорії, зокрема музику, фільми, телесеріали, книги, ігри та подкасти. Користувачі можуть створювати профіль, додавати свої вподобання та отримувати персоналізовані рекомендації за допомогою системи пропозицій. Чим більше користувач досліджує сайт, тим точніші стають рекомендації. Система враховує попередню активність і дозволяє переглядати схожий контент. Сайт також підтримує функцію соціальної взаємодії. Користувачі можуть ділитися рекомендаціями та переглядати вподобання інших [5].

Таблиця 1.1 – Опис TasteDive

Назва	TasteDive
Виробник	Qloo
Архітектура	Вебзастосунок, клієнт-сервер
Мова реалізації	React.js, Node.js, MongoDB, TMDb API

Кінець таблиці 1.1

Функції	– рекомендації на основі вподобань; – пошук за категоріями; – перегляд деталей; – перегляд списку схожих фільмів, музики, книг та інших категорій; – можливість поділитися контентом.
Переваги	– персоналізовані рекомендації; – безкоштовне використання; – велика база даних; – регулярні оновлення контенту; – простий та інтуїтивно зрозумілий дизайн.
Недоліки	– відсутність посилань на стримінгові сервіси; – відсутність розширених фільтрів для пошуку; – обмежений функціонал для незареєстрованих користувачів; – відсутність вбудованої системи рейтенгу; – обмежена точність рекомендацій.
Джерело інформації	https://tastedive.com/

Проаналізувавши можливості TasteDive, можна зробити висновок, що система орієнтована на надання персоналізованих рекомендацій, точність яких зростає з часом завдяки врахуванню вподобань користувача. Платформа має простий інтерфейс і підтримує базову соціальну взаємодію між користувачами. Однак функціонал пошуку обмежений та не реалізує достатньої гнучкості фільтрації чи рейтингової системи. Відсутність розширених налаштувань рекомендацій може впливати на зручність взаємодії користувачів з системою.

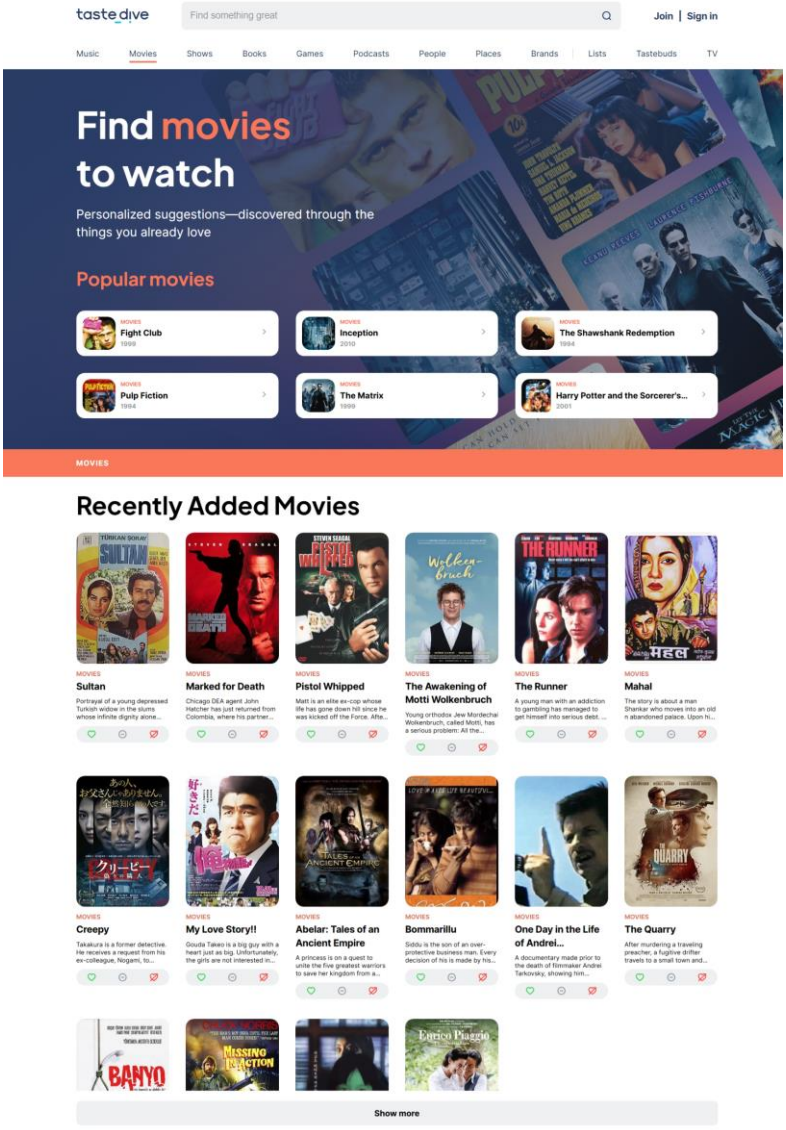

Рисунок 1.1 – Інтерфейс вебсайту TasteDive

Movie Decider

Movie Decider – це вебзастосунок, який підбирає фільми для перегляду відповідно до особистих вподобань користувача. Сервіс використовує інтерактивний підхід до вибору фільмів. Користувач вводить відповіді на запитання або вказує свої вподобання, а система формує персоналізовані рекомендації на основі вказаних даних. Чим більше деталей користувач вводить, тим кращими стають результати. Для аналізу вподобань Movie Decider застосовує алгоритми штучного інтелекту. Платформа працює у браузері та не потребує авторизації, що забезпечує швидкий доступ до основного функціоналу [6].

Таблиця 1.2 – Опис Movie Decider

Назва	Movie Decider
Виробник	Movie Decider, Inc.
Архітектура	Вебзастосунок, клієнт-сервер
Мова реалізації	HTML, CSS, JavaScript, PHP, MySQL, OMDb API
Функції	– інтерактивний підхід; – використання штучного інтелекту для аналізу вподобань; – розширені фільтри пошуку; – отримання рекомендацій без реєстрації; – випадковий вибір фільму.
Переваги	– простий та інтуїтивно зрозумілий дизайн; – відсутність необхідності реєстрації; – велика база даних; – безкоштовне використання; – інтеграція з потоковими платформами; – можливість отримати випадкову рекомендацію.
Недоліки	– відсутність вбудованої системи рейтенгу та коментарів; – відсутність можливості створення власного списку вподобань; – використання лише загальних алгоритмів; – обмежена кількість результатів (10); – відсутність персоналізованого акаунту.
Джерело інформації	https://moviedecider.com/

Проаналізувавши вебсайт Movie Decider, можна зробити висновок, що він реалізує підбір фільмів із використанням штучного інтелекту та розширених фільтрів, однак не передбачає створення персонального аккаунту та має обмежену кількість результатів. Через відсутність облікового запису користувач не може

зберігати свої вподобання або переглядати історію запитів, що зменшує рівень персоналізації. Також відсутність рейтингової системи, індивідуальних рекомендацій та можливості створення власних добірок обмежує гнучкість взаємодії з платформою.

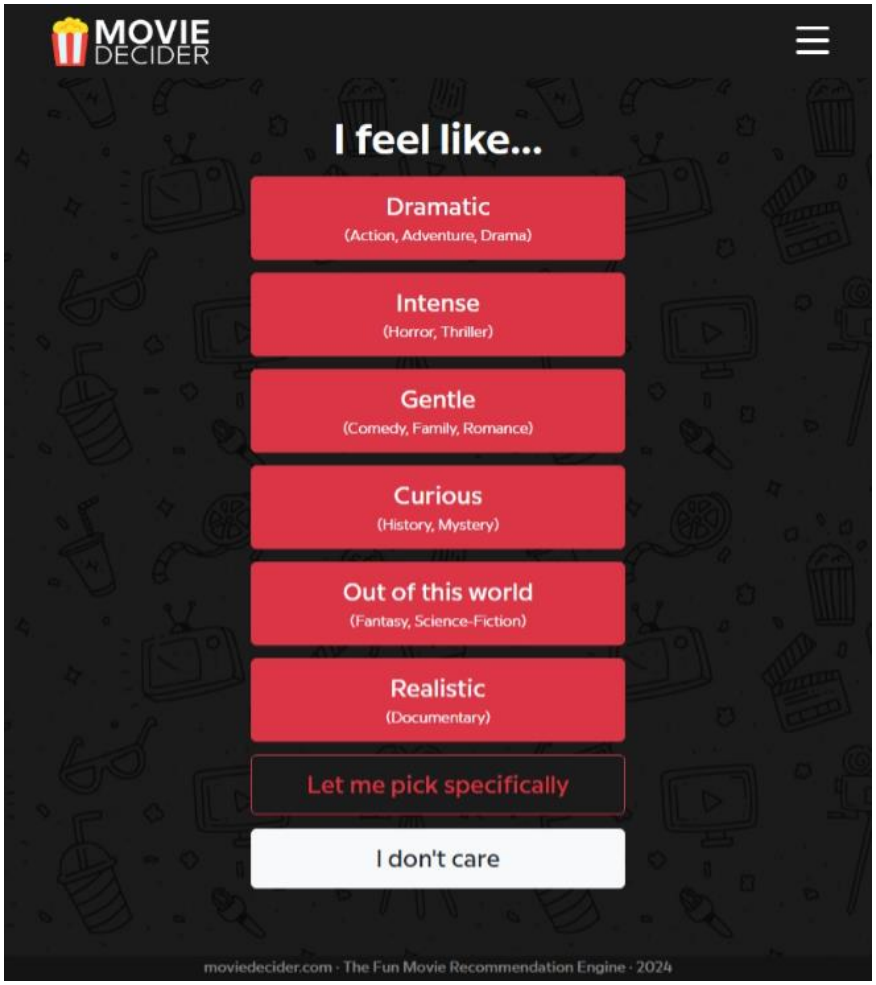

Рисунок 1.2 – Інтерфейс вебсайту Movie Decider

Taste

Taste – це вебзастосунок, який допомагає користувачам знаходити фільми та телешоу, що персоналізовані відповідно до особистих вподобань. Користувачі можуть оцінювати переглянуті фільми, додавати свої підписки на стримінгові сервіси та отримувати персоналізовані рекомендації. Вебзастосунок дозволяє створювати список для майбутнього перегляду. Система аналізує вподобання на основі оцінок і переглядів, та завдяки гнучким алгоритмам Taste забезпечує

високий рівень індивідуальних рекомендацій. Інтерфейс є зручним та адаптованим під різні пристрої [7].

Таблиця 1.3 – Опис Taste

Назва	Taste
Виробник	Taste, Inc.
Архітектура	Вебзастосунок, клієнт-сервер
Мова реалізації	React.js, Node.js, MongoDB, TMDb API
Функції	– оцінка переглянутих фільмів і телешоу; – додавання стримінгових підписок; – отримання персоналізованих вподобань; – створення списку для перегляду; – безкоштовний доступ до контенту.
Переваги	– можливість створювати власний список для перегляду; – інтегрування з популярними стримінговими сервісами; – надання індивідуальних рекомендацій; – простий та інтуїтивно зрозумілий дизайн; – безкоштовне використання.
Недоліки	– обмежений функціонал для незареєстрованих користувачів; – потрібен час, щоб система почала видавати точні рекомендації; – обмежений каталогу фільмів залежно від регіону; – рекомендації не завжди враховують актуальні тренди; – алгоритм може некоректно підбирати фільми з кількома жанрами.
Джерело інформації	https://www.taste.io/

Проаналізувавши вебсайт Taste, можна зробити висновок, що він надає рекомендації та підтримує інтеграцію з підписками на стримінгові сервіси, однак має регіональні обмеження. Сервіс вимагає створення облікового запису, що може створити обмеження отримання доступу до сервісу для нових користувачів. Taste забезпечує високий рівень персоналізації, проте потребує часу для налаштування профілю, оскільки рекомендації формуються виключно на основі активності користувача. Інтерфейс є зручним і зрозумілим, однак частина функціоналу доступна лише після авторизації.

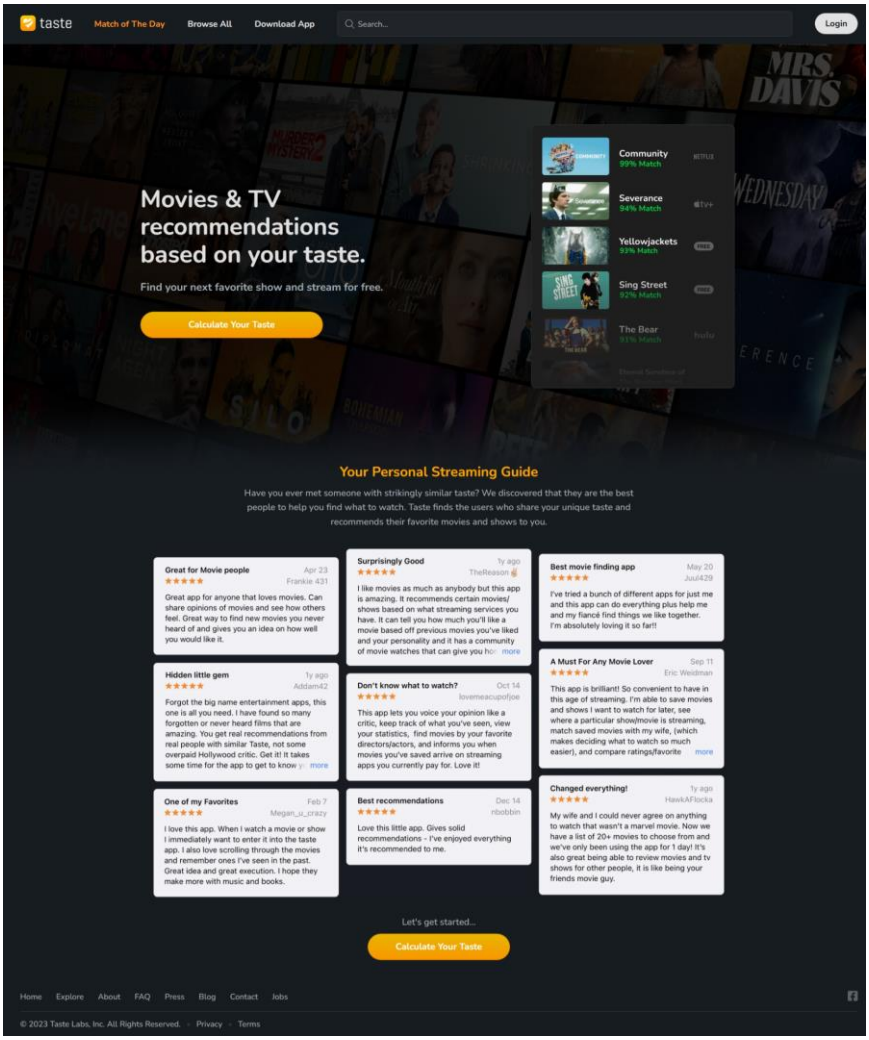

Рисунок 1.3 – Інтерфейс вебсайту Taste

У результаті аналізу вебзастосунків TasteDive, Movie Decider та Taste виявлено, що всі вони орієнтовані на надання персоналізованих рекомендацій користувачам, проте мають як переваги, так і обмеження. Їхній функціонал

варіюється від базового підбору фільмів до складних систем персоналізації. Деякі сервіси потребують реєстрації та авторизації для доступу до повного функціоналу. Системи, що не вимагають створення облікового запису, такі як Movie Decider, обмежені у можливості надання персоналізованих рекомендацій та збереження даних. Аналіз особливостей і недоліків обраних вебзастосунків допоміг у визначенні функціоналу розроблюваного вебсайту, формуванні його інтерфейсу та специфікації вимог.

1.3 Опис функціональних особливостей вебсайту

Після аналізу аналогічних вебзастосунків можна визначити необхідність використання сучасних інструментів, зокрема технологій штучного інтелекту, для спрощення процесу пошуку фільмів і формування рекомендацій у розроблюваній системі. Використання таких технологій дозволяє автоматизувати підбір контенту та підвищити якість взаємодії користувача з платформою. Вебзастосунок поєднуватиме зручний інтерфейс, який буде адаптивним і гнучким, а також передбачатиме різні варіанти пошуку фільмів з урахуванням індивідуальних уподобань користувача. Важливою складовою функціоналу є також віртуальний помічник на основі штучного інтелекту, який допомагатиме швидко знаходити потрібний контент без необхідності довготривалого пошуку.

У процесі розробки системи необхідно приділити увагу оптимізації часу вибору фільмів. Система має формувати пропозиції, враховуючи такі параметри, як вподобання користувача, його настрій та потреби. Це дозволить уникнути ефекту інформаційного перевантаження та зменшити час на прийняття рішення. Інтерфейс повинен бути простим та інтуїтивно зрозумілим, з легкою навігацією, швидким доступом до основних розділів, а також із привабливим й сучасним оформленням.

Функціональна частина сайту повинна включати основні компоненти, зокрема: форми реєстрації та авторизації користувачів, що забезпечують персоналізований доступ і збереження налаштувань; головну сторінку із зручною формою пошуку фільмів і фільтрами; інтерактивний чат із віртуальним

помічником, який здійснює діалог у вільній формі; а також розділ із тематичними добірками та рекомендаціями, що оновлюються на основі аналітики та тенденціях.

Головна сторінка вебсайту має забезпечити простий та зрозумілий інтерфейс для пошуку фільмів. Пошукова форма передбачатиме можливість застосування різноманітних фільтрів, таких як жанр, рік випуску, мова оригіналу тощо. Також необхідно реалізувати функцію випадкового вибору фільму, яка дозволятиме користувачам швидко отримати одну рекомендацію до перегляду. Такий підхід підвищує гнучкість використання платформи і відповідає різним сценаріям поведінки, що робить систему більш конкурентоспроможною порівняно з аналогами.

Однією з ключових функцій є інтеграція віртуального помічника, що працює на основі алгоритмів штучного інтелекту. Взаємодія із системою через запити у вільній формі є зручною для користувачів, які не можуть чітко сформулювати критерії пошуку. Завдяки здатності розуміти контекст і враховувати індивідуальні особливості поведінки, віртуальний помічник зможе надавати більш точні відповіді, тим самим підвищуючи якість взаємодії із системою. У процесі тривалого використання віртуального помічника система формуватиме персоналізовані рекомендації на основі вподобань, оцінок і запитів користувача. Такий підхід допоможе підтримувати актуальність пропонованого контенту і забезпечить залученість, оскільки рекомендації будуть динамічно оновлюватись із урахуванням змін.

Під час розробки вебсайту необхідно реалізувати функціонал перегляду тематичних добірок, класифікованих за жанрами, настроєм або іншими категоріями. Ця можливість значно спростить пошук для користувачів, які мають конкретні уподобання. Списки формуватимуться на основі аналізу актуальних трендів, рівня популярності та сучасних тенденцій на стримінгових платформах, а також із урахуванням індивідуальних вподобань зареєстрованих користувачів.

Висновки до розділу 1

У першому розділі кваліфікаційної бакалаврської роботи проведено аналіз предметної області, пов’язаної з рекомендаційними системами у сфері медіаконтенту, та описано особливості сучасних стримінгових платформ. Розглянуто вплив зростання обсягів інформації та потребу у впровадженні технологій штучного інтелекту, зокрема в системах персоналізованих рекомендацій. Проведено огляд і порівняльний аналіз існуючих вебзастосунків-аналогів, визначено їхні основні функціональні можливості, а також проаналізовано переваги та недоліки.

Зокрема, TasteDive пропонує широкий вибір фільмів і серіалів та має простий у використанні інтерфейс, однак позбавлений гнучких параметрів фільтрації. Movie Decider застосовує технології штучного інтелекту для генерації рекомендацій і містить розширену систему фільтрів, але має низку функціональних обмежень, наприклад, не підтримує створення облікових записів та містить обмежену кількість результатів. Taste дозволяє інтеграцію з популярними стримінговими сервісами та підтримує створення власних списків перегляду, проте має регіональні обмеження та вимагає обов’язкової реєстрації, що може викликати незручності у нових користувачів.

У результаті проведеного аналізу сформульовано функціональні особливості вебсайту, що розробляється. Описано основну структуру сайту, яка складається з головної сторінки, форми пошуку, віртуального помічника, списку добірок фільмів та системи рекомендацій. Визначена структура забезпечує зручність використання та ефективний підбір контенту відповідно до інтересів користувача.

2 ІНСТРУМЕНТАЛЬНІ ЗАСОБИ РОЗРОБКИ ВЕБСАЙТУ

2.1 Огляд технологій

Персоналізація є важливою частиною для створення зручної та ефективної системи підбору фільмів, оскільки кожен користувач хоче отримувати саме той контент, що максимально відповідає його інтересам та потребам. У сучасних умовах інформаційного перенасичення актуальними стають інтелектуальні системи. Популярність набирають платформи, що працюють на основі аналізу вмісту, використовуючи глибоке навчання для визначення характеристики користувачів та передбачення популярності фільмів. Такі системи аналізують текстовий, візуальний та жанровий опис фільмів, зіставляючи їх із інтересами глядача, виявленими на основі попередніх взаємодій. Авторматичне оновлення рекомендацій у реальному часі забезпечує динамічну взаємодію з користувачем. Саме тому, у процесі розробки вебзастосунку доцільним є впровадження сучасних інструментів штучного інтелекту, зокрема OpenAI API, як основного механізму персоналізації. Такий підхід розширить можливості користувача у взаємодії з системою [8].

OpenAI API (рис. 2.6) – це інтерфейс для взаємодії з моделями штучного інтелекту, призначений для обробки природної мови, генерації тексту, класифікації, перекладу та діалогових взаємодій. Його використання необхідне для реалізації віртуального помічника, який аналізує запити користувачів і формує релевантні рекомендації на основі заданих параметрів, описів або уподобань [9].

Рисунок 2.1 – OpenAI

Даний API має велику кількість переваг, наприклад, висока точність відповідей, гнучка контекстна обробка запитів, адаптивність до різних сценаріїв взаємодії та зручна інтеграція з серверною частиною. Завдяки регулярним

оновлення користувачі мають доступ до новітніх можливостей моделей, що забезпечує актуальність і функціональність системи. OpenAI API підтримує масштабування, що дозволяє ефективно обслуговувати будь-яку кількість запитів.

Проте, OpneAI API є зовнішнім сервісом, що вимагає стабільного підключення до мережі та контролю за безперебійною роботою. У разі високого навантаження на сервер можливе зростання затримок у відповіді або обмеження доступу, а повноцінне використання інтерфейсу може бути платним.

Використання актуального джерела даних із великою кількістю інформації про фільми є основою для створення якісної рекомендаційної системи. Це дозволяє не лише наповнити сайт контентом, а й використовувати ці дані для побудови персоналізованих рекомендацій, точної фільтрації, пошуку за ключовими словами, жанрами, сюжетами тощо. Для пошуку схожих фільмів у базі даних необхідно використовувати метод косинусної схожості. Цей підхід дозволяє обчислити ступінь подібності між векторними представленнями фільмів на основі їх опису, жанру, акторського складу тощо, що сприяє точнішому добору відповідного контенту для кожного глядача [8].

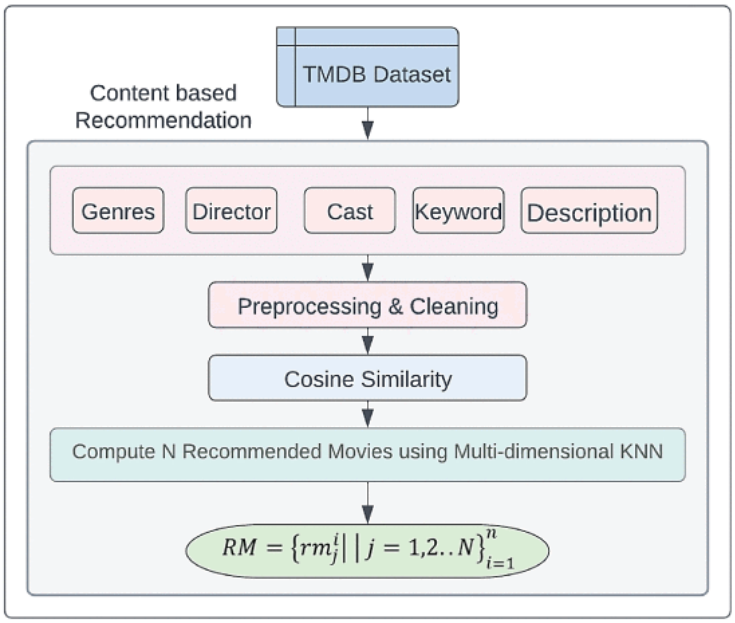

Рисунок 2.2 – Блок-схема модуля пошуку схожих фільмів

TMDb API (рис. 2.7) – це зручний інтерфейс для доступу до великої бази фільмів та іншого медіаконтенту. Він надає структуровану інформацію про назви,

описи, жанри, акторів, рейтинги, постери тощо. Завдяки цьому TMDb є одним із найпопулярніших рішень для розробників, які створюють медіаплатформи. У розроблюваному вебсайті TMDb API необхідний для автоматичного завантаження актуальних даних про фільми, що дає змогу наповнювати інтерфейс сайту динамічним контентом [10].

Рисунок 2.3 – TMDb

Основними перевагами TMDb API є безкоштовне використання для некомерційних цілей, багатомовна підтримка, зручний формат REST-запитів, а також детальна документація зі спрощеною інтеграцією. База даних регулярно оновлюється та гарантує надання актуальної інформації про фільми. Однак є деякі обмеження у кількість запитів на добу при великій кількості користувачів або високому навантаженні системи.

Для реалізації логіки системи необхідно використати технології, які дозволяють працювати з великими обсягами даних, швидко обробляти запити і бути зручними у використані. React та Spring Boot забезпечують зручну побудову інтерфейсу користувача і серверної логіки, дозволяючи створити швидкий, стабільний та зрозумілий у підтримці вебзастосунок. React дозволяє створювати динамічні та інтерактивні сторінки, що швидко реагують на дії користувача без перезавантаження. Spring Boot забезпечує просту роботу з базами даних, обробку запитів та взаємодію між компонентами системи [11].

React (рис. 2.3) є однією з найпопулярніших бібліотек JavaScript для розробки інтерактивних користувацьких інтерфейсів. Порівняно із класичним JavaScript, вона забезпечує створення динамічних інтерфейсів з меншим обсягом коду, забезпечуючи швидку реакцію на дії користувача або зміну даних без необхідності оновлення сторінки. У розроблюваному вебсайті React необхідний при створені клієнтської частини системи для взаємодії з користувачем та відображення результатів пошуку [12].

Рисунок 2.4 – React

React має компонентну архітектуру, що робить код модульним та зручним для повторного використання і створення складних проєктів. Крім того, React підтримує серверний рендеринг зменшуючи час завантаження сторінок та покращуючи продуктивність застосунку. Можна виділити складний синтаксис та обмежену кількість вбудованих рішень, що може потребувати підключення додаткових бібліотек.

Spring Boot (рис. 2.4) – це фреймворк для Java, який дозволяє швидко створювати масштабовані, продуктивні та безпечні вебзастосунки. У розроблюваному вебсайті Spring Boot обрано як основу серверної частина для обробки запитів користувачів, взаємодії з базою даних та API [13].

Рисунок 2.5 – Spring Boot

Цей фреймворк автоматизує багато налаштувань і дозволяє запускати застосунок без потреби у складній конфігурації. Також Spring Boot добре працює з REST-архітектурою, що робить його зручним для створення клієнт-серверних систем із фронтендом на React. Недоліками є складність освоєння для новачків через велику кількість інструментів і налаштувань, які потрібно освоїти для ефективної роботи.

PostgreSQL (рис. 2.5) – це об’єктно-реляційна система управління базами даних, яка поєднує надійність, гнучкість та підтримку сучасних функцій. У розроблюваному вебсайті PostgreSQL необхідний для зберігання даних користувачів, історії запитів до віртуального помічника, а також даних для

формування персоналізованих рекомендацій. Система легко інтегрується з серверною частиною, реалізованою за допомогою Spring Boot [14].

Рисунок 2.6 – PostgreSQL

Перевагами є підтримка складних типів даних, можливість виконання повнотекстового пошуку, використання різних індексів та розширення функціоналу за допомогою модулів. PostgreSQL є безкоштовним (open-source) рішенням із великою спільнотою розробників і документацією. Складнощі можуть виникнути при налаштуванні реплікації, масштабуванні або при використанні просунутих функцій, які потребують глибоких знань SQL та принципів побудови реляційних баз. Однак для більшості задач PostgreSQL забезпечує простоту використання й високу продуктивність.

Google OAuth – це сервіс авторизації, який дозволяє користувачам швидко входити на сайт, використовуючи свій обліковий запис Google. Завдяки цьому інструменту користувачу немає потреби створювати новий акаунт або запам’ятовувати новий пароль. Використання даного сервісу значно спрощує процес реєстрації та входу.

Рисунок 2.7 – Логотип Google OAuth

Сервіс забезпечує високий рівень безпеки, включаючи двофакторну автентифікацію, захист від несанкціонованого доступу та автоматичне оновлення

даних користувача. Google OAuth дозволяє зменшити кількість помилок при введенні даних, оскільки частина інформації (наприклад, ім’я та електронна адреса) підтягується автоматично. Однією з важливих переваг є легка інтеграція з серверною частиною на Spring Boot.

За допомогою спеціалізованих інструментів моделювання можна візуально зобразити архітектуру та структуру системи для кращого розуміння логіки роботи окремих компонентів, їхньої взаємодії та ролі в загальній роботі проєкту. Серед різноманітних інструментів візуального представлення найзручнішими засобами для створення UML-діаграм і опису програмної архітектури є StarUML та Software Ideas Modeler.

StarUML (рис. 2.1) – це сучасний інструмент для моделювання програмного забезпечення. Він дозволяє створювати UML-діаграми для візуалізації структури та логіки роботи системи. Інструмент використовується для побудови діаграм класів, прецедентів, послідовностей, компонентів та інших типів моделей при проєктуванні програм [15].

Рисунок 2.8 – Логотип StarUML

Програма має зрозумілий та зручний інтерфейс. Доступні функції експорту в різні формати та можливість ділитися діаграмами з командою для спільного використання. Основним недоліком є обмеження безкоштовної версії. Програма має мінімальні можливості й обмежений термін використання, тому повноцінна робота з програмою вимагає придбання ліцензії.

Software Ideas Modeler (рис. 2.2) – це зручний інструмент для моделювання програмного забезпечення, який дозволяє створювати велику кількість типів діаграм, таких як діаграми класів, станів, послідовностей тощо. Він є корисним для

візуалізації структури, поведінки та архітектури системи. Даний інструмент значно спрощує процес проєктування програмного забезпечення [16].

Рисунок 2.9 – Логотип Software Ideas Modeler

Серед основних переваг Software Ideas Modeler варто зазначити велику кількість шаблонів, а також зручний і інтуїтивно зрозумілий інтерфейс для редагування та налаштування моделей. Інструмент підтримує зручне редагування, експорт діаграм у різні формати та доступ до безкоштовної версії. Недоліками є перевантажений інтерфейс через велику кількість інструментів та обмежена підтримка спільного використання.

2.2 Специфікація вимог

1. Призначення та межі проєкту

1.1 Призначення системи

Вебсайт з використанням технології штучного інтелекту для пошуку фільмів призначений для допомоги у підборі фільму, який якнайкраще відповідатиме вимогам, вподобанням або настрою користувача. Система забезпечує перегляд опису до фільмів, отримання персоналізованих рекомендацій, а також використання фільтрації, пошуку та спілкування з віртуальним помічником.

1.2 Погодження, ухвалені в програмній документації

– специфікація розроблена на основі аналізу потреб кіноглядачів, сучасних проблем перенасичення контентом та складнощів вибору;

– ухвалено використання сучасних вебтехнологій для забезпечення ефективної розробки та зручної інтеграції з зовнішніми сервісами;

– визначено основні функції, які реалізовуватимуться на першому етапу розробки.

1.3 Межі проєкту

– проєкт охоплює розробку клієнтської та серверної частини вебзастосунку;

– передбачено інтеграцію з зовнішніми API для реалізації функції віртуального помічника та отримання інформації про фільми;

– система не включає функціональність для перегляду фільмів, але передбачає пошук, перегляд опису та рекомендацій;

– не включає створення мобільного застосунку.

2. Загальний опис

2.1 Сфера застосування

Вебсайт використовується індивідуально користувачами для пошуку фільмів за особистими вподобаннями, що забезпечується за допомогою технології штучного інтелекту.

2.2 Характеристики користувачів

– будь-які глядачі, які шукають фільм відповідно до індивідуальних критеріїв;

– користувачі, які цінують простоту, зручність та швидкість системи;

– користувачі повинні володіти медіаграмотністю та базовими навичками роботи з вебзастосунками.

2.3 Загальна структура і склад системи

– клієнтська частина: вебінтерфейс, побудований на основі React;

– серверна частина: Spring Boot із базою даних PostgreSQL;

– API: використання TMDb API для отримання даних про фільми та OpenAI API для використання можливостей ШІ.

2.4 Загальні обмеження

– вебсайт працює лише за наявності стабільного інтернет-з’єднання;

– персоналізовані рекомендації залежать від кількості зібраних даних про користувача;

– підтримка останніх версій веббраузерів (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari).

3. Функції системи

3.1 Форма для пошуку фільмів

3.1.1 Опис функції

Користувач може відповісти на низку питань, на основі яких формується відповідь.

3.1.2 Вхідна і вихідна інформація

– вхідна: відповіді користувача на запитання у формі; дані з бази TMDb;

– вихідна: список рекомендованих фільмів з додатковою інформацією.

3.1.3 Функціональні вимоги

– форма містить питання із можливістю вибору варіантів;

– валідація введених даних;

– виведення результатів у вигляді списку;

– створення запиту до зовнішніх API згідно заповненої форми.

3.2 Спілкування з віртуальним помічником щодо фільмів

3.2.1 Опис функції

Користувач може взаємодіяти з віртуальним помічником, який за допомогою штучного інтелекту надає інформацію про фільми.

3.2.2 Вхідна і вихідна інформація

– вхідна: текстові запитання користувача у вільній формі;

– вихідна: згенерована відповідь відповідно до запитання користувача про фільми.

3.2.3 Функціональні вимоги

– віртуальний помічник відповідає лише на запитання, що стосуються фільмів;

– використовується OpenAI API із заздалегідь сформульованим системним обмеженням на тематику відповідей;

– при некоректному запиті виводиться повідомлення з проханням задавати запитання лише про фільми.

3.3 Обрання випадкового фільму

3.3.1 Опис функції

Користувач отримує рекомендацію фільму без попереднього заповнення форми. Фільм обирається автоматично за допомогою API.

3.3.2 Вхідна і вихідна інформація

– вхідна: натискання кнопки користувачем;

– вихідна: назва та інформація про один випадково обраний фільм.

3.3.3 Функціональні вимоги

– випадкове обрання фільму без попереднього заповнення форми

– звернення до TMDb API;

– запобігання повторного вибору одного й того ж фільму при кількох натисканнях поспіль.

3.4 Перегляд добірок фільмів за категоріями

3.4.1 Опис функції

Система пропонує користувачі добірки фільмів за жанрами, популярністю, настроєм та іншими категоріями для зручного вибору контенту.

3.4.2 Вхідна і вихідна інформація

– вхідна: вибір категорії користувачем;

– вихідна: список фільмів за обраною категорією.

3.4.3 Функціональні вимоги

– можливість перегляду добірок без авторизації;

– автоматичне формування категорій за популярністю.

3.5 Реєстрація та авторизація користувачів

3.5.1 Опис функції

Користувач може створити особистий обліковий запис, входити до системи та виходити з неї. Ця функція надає доступ до повного функціоналу системи.

3.5.2 Вхідна і вихідна інформація

– вхідна: дані користувача (ім’я користувача, електрона пошта, пароль);

– вихідна: повідомлення про успішну реєстрацію/авторизацію або виведення помилки.

3.5.3 Функціональні вимоги

– валідація полів реєстрації та авторизації;

– шифрування паролів та збереження даних у базі даних;

– обмеження доступу до певних функцій без попередньої реєстрація/авторизації.

4. Вимоги до інформаційного забезпечення

4.1 Джерела і зміст вхідної інформації

– дані користувача (ім’я, електрона пошта та пароль) надаються самим користувачем при створенні облікового запису;

– додаткова інформація (вподобання, оцінки та історія користування системою) формуються в процесі взаємодії із вебсайтом;

– інформація про медіаконтент отримується із зовнішнього API (TMDb API);

– пошукові запити надходять у текстовому форматі або за допомогою вибору серед запропонованих категорій.

4.2 Нормативно-довідкова інформація

– довідник жанрів фільмів (драма, комедія, фантастика тощо);

– довідник інформації про фільми (жанр, країна, короткий опис тощо);

– шкала рейтингу (TMDb API та особиста оцінка користувача).

4.3 Вимоги до способів організації, збереження та ведення інформації

– зберігання інформації у базі даних (PostgreSQL);

– регулярне резервне копіювання;

– шифрування персональних даних користувача.

5. Вимоги до технічного забезпечення

– серверна частина: потужний персональний комп’ютер з мінімум 4 GB оперативної пам’яті для запуску вебсайту;

– мережа: стабільне інтернет-з’єднання з підтримкою HTTPS;

– клієнтські пристрої: оновлені версії браузерів (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari).

6. Вимоги до програмного забезпечення

6.1 Архітектура програмної системи

Клієнт-серверна архітектура системи, що складається з клієнтської сторони, серверної частини, бази даних та підключених API.

6.2 Мережне програмне забезпечення

– протокол HTTPS для захищеного з’єднання;

– підключення зовнішніх API.

6.3 Програмне забезпечення ведення інформаційної бази

СКБД PostgreSQL 13 або новіше.

6.4 Мова і технологія розробки

– frontend: React;

– backend: Spring Boot;

– база даних: PostgreSQL;

– API: OpenAI API, TMDb API.

7. Вимоги до зовнішніх інтерфейсів

7.1 Інтерфейс користувача

– адаптивний дизайн;

– зручна навігація, інтуїтивно зрозумілий інтерфейс з підтримкою англійської мови.

7.2 Апаратний інтерфейс

– не потребує спеціалізованого обладнання;

– сумісний із стандартними ПК та мобільними пристроями.

7.3 Програмний інтерфейс

REST API для взаємодії з API.

7.4 Комунікаційний протокол

Використання HTTPS-протоколу для забезпечення безпеки.

8. Властивості програмного забезпечення

8.1 Доступність

Цілодобова доступність вебсайту за умови стабільної роботи серверної частини та підключення до API.

8.2 Супроводжуваність

Реалізація у вигляді окремих модулів для спрощення оновлення та масштабування.

8.3 Переносимість

Підтримка різних сучасних браузерів не залежно від конкретної операційної системи.

8.4 Продуктивність та надійність

– оптимізовані запити до зовнішніх API;

– час відповіді системи до 3 секунд.

8.5 Безпека

– використання шифрування даних користувача;

– авторизація з валідацією введених даних.

Висновки до розділу 2

У другому розділі кваліфікаційної бакалаврської роботи визначено технічну основу для подальшого моделювання, проєктування та реалізації вебсайту. Проведено огляд та детальний аналіз інструментів і технологій, обраних для розробки: StarUML і Software Ideas Modeler для побудови діаграм і візуалізації архітектури, React для створення інтерфейсу користувача, Spring Boot для реалізації серверної частини, PostgreSQL для збереження та обробки даних, а також OpenAI API і TMDb API, які забезпечують функціональність віртуального помічника та динамічний контент. Визначено переваги та недоліки кожного інструменту, а також необхідність їх використання в межах проєкту.

Крім того, у розділі сформовано специфікацію вимог до системи. Описано поведінку програмного забезпечення, що розробляється. Визначено функціональні та нефункціональні вимоги, основні обмеження та загальні принципи реалізації для забезпечення ефективної роботи вебсайту.

3 МОДЕЛЮВАННЯ ВЕБСАЙТУ

3.1 Сценарії використання

Сценарії використання – це метод моделювання, що застосовується для аналізу та формування вимог до програмного забезпечення. Вони описують можливу взаємодію між користувачем і системою під час виконання певної функції, що дозволяє змоделювати поведінку системи з точки зору кінцевого користувача. Сценарії використання спрощують визначення функціональних вимог, допомагають у процесі розробці та дозволяють передбачити можливі помилки в роботі системи.

Кожен сценарій використання містить назву, опис ролей користувача й системи, попередні умови, які мають бути виконані до початку виконання сценарію, та спеціальні вимоги. Основною частиною опису є головний сценарій, який представлений у вигляді послідовних кроків для успішного досягнення мети, та альтернативні ситуації, що враховують збій у роботі системі чи неправильні дії користувача [17].

Сценарій використання для функції реєстрації нового користувача (табл. 3.1) описує основні кроки створення облікового запису, а також можливі помилки під час введення даних. Передбачено стандартну реєстрацію, а також можливість реєстрації через акаунт Google. Визначено вимоги до безпеки та валідації введених даних.

Таблиця 3.1 – Сценарій використання «Реєстрація користувача»

Usecase section	**Comment**
Use Case Name	Реєстрація користувача
Scope	System
Level	User-goal
Primary Actor	Новий користувач

Кінець таблиці 3.1

Stakeholders and interests	Користувач – хоче створити обліковий запис, щоб отримати доступ до повного функціоналу вебзастосунку.
Preconditions	Користувач не має облікового запис
Success guarantee	Обліковий запис створено, користувач може увійти
Main Success Scenario	1. Користувач переходить на сторінку реєстрації. 2. Система відображає форму для введення даних (ім'я користувача, email, пароль). 3. Користувач заповнює форму. 4. Система перевіряє правильність введених даних та створює новий обліковий запис. 5. Система перенаправляє користувача на головну сторінку.
Extensions	1. Користувач уже зареєстрований – система повідомляє про це. 2. Користувач залишає обов'язкові поля порожніми – система показує повідомлення про помилку.
Special Requirements	– валідація полів; – захищене зберігання паролів; – виведення повідомлень при помилках.
Technology and Data Variations List	Реєстрація користувача через Google-акаунт.
Frequency of Occurrence	10%

Сценарій використання для авторизації користувача (табл. 3.2) описує послідовність дій для входу до особистого облікового запису. Передбачено стандартний вхід за допомогою email і пароля, а також авторизацію через акаунт

Google. Враховано можливі помилки користувача при введенні даних та визначено основні вимоги до безпеки.

Таблиця 3.2 – Сценарій використання «Авторизація користувача»

Usecase section	**Comment**
Use Case Name	Авторизація користувача
Scope	System
Level	User-goal
Primary Actor	Зареєстрований користувач
Stakeholders and interests	Користувач – хоче увійти до особистого облікового запису.
Preconditions	Користувач має дійсний обліковий запис
Success guarantee	Користувач успішно входить у систему
Main Success Scenario	1. Користувач переходить на сторінку авторизації. 2. Система відображає форму входу (email та пароль). 3. Користувач вводить свої облікові дані. 4. Система перевіряє коректність введених даних та перенаправляє користувача на головну сторінку.
Extensions	1. Користувач залишає обов’язкові поля порожніми – система показує повідомлення про помилку. 2. Невірні дані – система повідомляє про помилку входу.
Special Requirements	– валідація полів; – шифрування передачі даних.
Technology and Data Variations List	Вхід до системи через Google-акаунт.
Frequency of Occurrence	80%

Сценарій використання для функції отримання випадкових рекомендацій фільмів (табл. 3.3) описує процес швидкого підбору фільму для перегляду. Функція доступна як зареєстрованим, так і незареєстрованим користувачам. Визначено взаємодію з TMDb API та обробку можливих помилок підключення. Зазначено додаткові вимоги до швидкості та різноманітності результатів.

Таблиця 3.3 – Сценарій використання «Отримання випадкових рекомендацій фільмів»

Usecase section	**Comment**
Use Case Name	Отримання випадкових рекомендацій фільмів
Scope	System
Level	User-goal
Primary Actor	Користувач (зареєстрований або незареєстрований)
Stakeholders and interests	Користувач – хоче швидко отримати цікаву рекомендацію для перегляду.
Preconditions	Користувач заходить на головну сторінку сайту.
Success guarantee	1. Користувач отримує випадковий фільм та може переглянути додаткову інформацію до нього.
Main Success Scenario	2. Користувач відкриває головну сторінку та натискає кнопку «Підібрати випадковий фільм». 3. Система обирає один випадковий фільмів з бази даних TMDb. 4. Результат виводиться користувачу.
Extensions	Виникли помилки при підключені до бази даних TMDb – система повідомляє про збій.
Special Requirements	– система швидко оброблює запит; – результати надаються різноманітні.
Technology and Data Variations List	Оновлення списку без перезавантаження сторінки.
Frequency of Occurrence	90%

Сценарій використання для функції перегляду добірок фільмів за тематиками або категоріями (табл. 3.4) описує спосіб знаходження фільмів відповідно до інтересів користувача. Функція доступна для всіх користувачів вебсайту. Для зареєстрованих користувачів функція додатково враховує особисті вподобання. Передбачено обробку помилок при завантаженні даних із TMDb API.

Таблиця 3.4 – Сценарій використання «Перегляд добірок фільмів за тематиками або категоріями»

Usecase section	**Comment**
Use Case Name	Перегляд добірок фільмів за тематиками або категоріями
Scope	System
Level	User-goal
Primary Actor	Користувач (зареєстрований або незареєстрований)
Stakeholders and interests	Користувач – хоче легко та швидко знаходити фільми за певною тематикою чи категорією.
Preconditions	Користувач заходить на сторінку з добірками.
Success guarantee	Користувач бачить список фільмів, що відповідають обраній тематиці чи категорії.
Main Success Scenario	1. Користувач відкриває розділ з добірками та обирає одну з тем або категорій. 2. Система обробляє запит і показує добірку відповідних фільмів. 3. Користувач переглядає список та може обрати фільм для детального перегляду інформації.
Extensions	Сталася помилка під час завантаження добірки або при підключені до бази даних TMDb – система повідомляє про помилку.
Special Requirements	Для зареєстрованих користувачів врахування особистих побажань.

Кінець таблиці 3.4

Technology and Data Variations List	Статичні та динамічно оновлюванні (персоналізовані) категорії.
Frequency of Occurrence	70%

Сценарій використання для функції отримання персоналізованих рекомендацій фільмів (табл. 3.5) описує процес генерації індивідуальних добірок на основі вподобань користувача. Функція доступна лише після авторизації. У разі нестачі інформації система пропонує популярні фільми. Передбачено зібрання даних через взаємодію з віртуальним помічником та заповнення форми пошуку.

Таблиця 3.5 – Сценарій використання «Отримання персоналізованих рекомендацій фільмів»

Usecase section	**Comment**
Use Case Name	Отримання персоналізованих рекомендацій фільмів
Scope	System
Level	User-goal
Primary Actor	Зареєстрований користувач
Stakeholders and interests	Користувач – хоче отримувати індивідуальні рекомендації на основі особистих вподобань.
Preconditions	Користувач увійшов до особистого облікового запису. Система має достатньо даних для генерації даних.
Success guarantee	Користувачу отримує персоналізований списки фільмів у відповідь на запит у чаті або в розділі добірок.
Main Success Scenario	1. Користувач звертається до віртуального помічника або обирає категорію добірки. 2. Система аналізує історію вподобань, поведінки або нещодавні вибори користувача.

Кінець таблиці 3.5

Main Success Scenario	3. Система пропонує персоналізований список фільмів. 4. Користувач обирає фільм для перегляду детальної інформації.
Extensions	1. У системі недостатньо даних для персоналізованих рекомендацій – обираються популярні фільми. 2. Користувач не задоволений списком фільмів – користувач може надіслати запит повторно та отримати новий список.
Special Requirements	Сторінка для взаємодії з віртуальним помічником та сторінка з добірками фільмів.
Technology and Data Variations List	Система збирає дані через дії та запити користувача.
Frequency of Occurrence	90%

Сценарій використання для взаємодії з віртуальним помічником (табл. 3.6) описує процес пошуку фільмів через чат, де користувач формулює запит, а система підбирає відповідні варіанти. Функція передбачає інтеграцію із OpenAI API та TMDb API. Для зареєстрованих користувачів система надає персоналізовані рекомендації на основі історії пошуку.

Таблиця 3.6 – Сценарій використання «Взаємодія з віртуальним помічником для пошуку фільму»

Usecase section	**Comment**
Use Case Name	Взаємодія з віртуальним помічником
Scope	System
Level	User-goal
Primary Actor	Користувач (зареєстрований або незареєстрований)

Продовження таблиці 3.6

Stakeholders and interests	Користувач – необхідність персоналізованих рекомендацій, легкий пошук фільмів.
Preconditions	– користувач заходить у розділ спілкування з віртуальним помічником; – система має доступ до API та коректно працює; – користувач увійшов до особистого облікового запису.
Success guarantee	Користувач отримує відповідь від віртуального помічника зі списком фільмів, що відповідають заданому запиту.
Main Success Scenario	1. Користувач заходить у розділ спілкування з віртуальним помічником. 2. Віртуальний помічник пропонує користувачу варіанти запитів. 3. Користувач вводить свій запит. 4. Система обробляє введені дані і здійснює запит до бази даних або зовнішнього API для отримання відповіді. 5. Віртуальний помічник надає список фільмів, що відповідають запиту.
Extensions	1. Віртуальний помічник не розуміє введену користувачем інформацію – задаються уточнюючі запитання до користувача. 2. Сталася помилка з API або базою даних – віртуальний помічник повідомляє користувача про проблему і пропонує спробувати пізніше.

Кінець таблиці 3.6

Special Requirements	– інтеграція з зовнішніми API; – для зареєстрованих користувачів врахування особистих побажань.
Technology and Data Variations List	– реалізація взаємодії з користувачем через чат; – обробка текстових запитів.
Frequency of Occurrence	90%

Описані сценарії використання демонструють основні функціональні можливості розроблюваної системи та визначають можливі дії користувача. Вони включають типові дії користувача, які допомагають зрозуміти, як система реагує на різні запити та умови. Крім успішних сценаріїв, описані також альтернативні варіанти. Це забезпечує підвищення надійності системи та її зручності у використанні. Враховано частоту виконання дій, вимоги до безпеки, варіанти доступу та можливі помилки.

3.2 Діаграма прецедентів

Діаграма прецедентів – це вид діаграми у мові моделювання UML, що візуально представляє взаємодію між користувачами та системою. Вона дозволяє легко побачити, які функції виконує система і актори при різних взаємодіях. Діаграма допомагає визначити межі системи та що до неї входить.

Основними елементами діаграми прецедентів є актори (користувачі) та прецеденти (дії, які може виконати користувач). Елементи пов’язані між собою лініями, що показують взаємозв’язки. Додатково можуть використовуватись спеціальні типи зв’язків, які дозволяють детальніше описати логіку роботи системи. Зв’язок «include» означає, що одна дія обов’язково включає іншу, а «extend» означає, що дія може бути розширена додатковими можливостями, які не завжди виконуються [18].

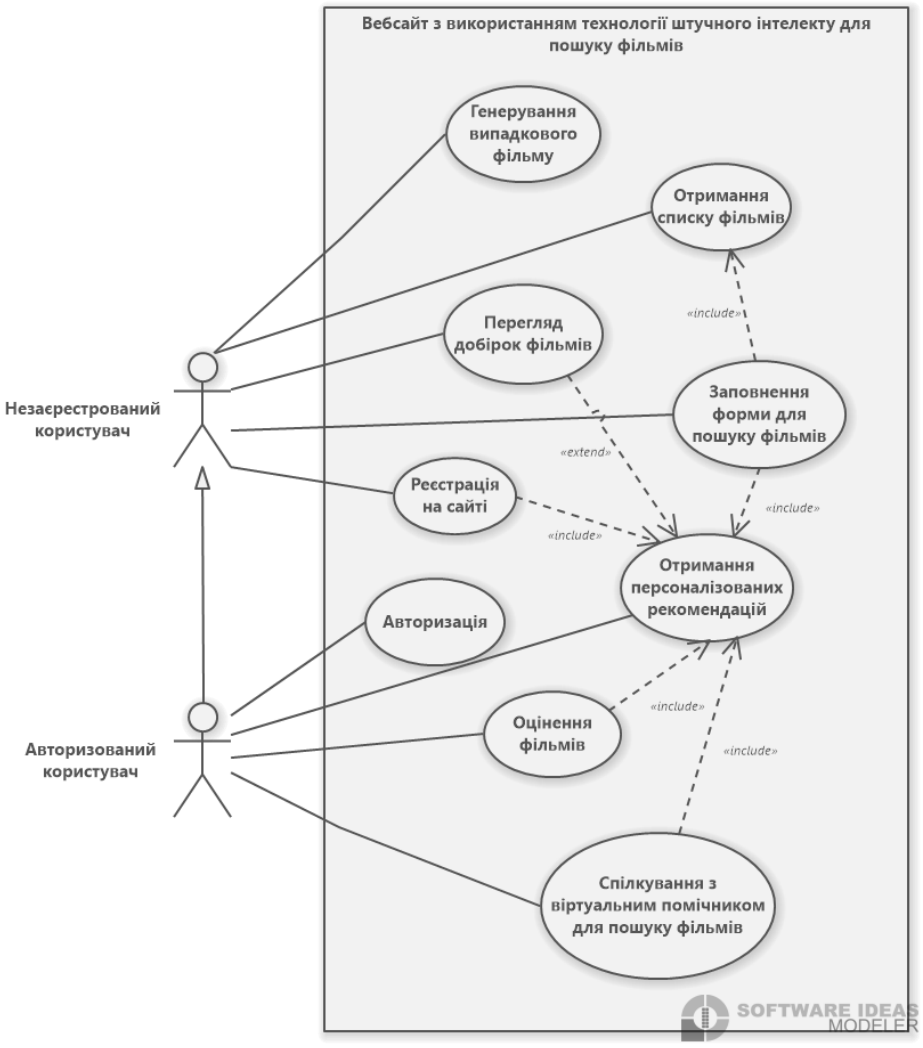

Рисунок 3.1 – Діаграма прецедентів

Діаграма прецедентів демонструє основні сценарії взаємодії з вебсайтом, який використовує штучний інтелект для пошуку фільмів. Ролі акторів поділено на зареєстрованого та незареєстрованого користувача. Можливості системи залежать від типу користувача.

Незареєстрований користувач має доступ до базової функціональності системи, а саме:

– введення пошукових запитів;
– отримання випадкової рекомендації;
– перегляд результатів;
– перегляд добірок фільмів за категоріями;
– можливість зареєструватися для розширення доступу.

Відкриті можливості дозволяють користувачам швидко оцінити функціональність сайту без потреби у створенні облікового запису. Це зручно, адже користувач одразу може спробувати основні функції. Реєстрація відкриває додаткові переваги. Зареєстрований користувач має доступ до всіх можливостей, доступних незареєстрованим користувачам, а також до розширеного функціоналу, зокрема:

– отримання персоналізованих рекомендацій на основі інтересів;
– взаємодія з віртуальним помічником для пошуку фільмів;
– можливість авторизації до особистого облікового запису;
– оцінювання фільмів для побудови індивідуальних рекомендацій.

Діаграма демонструє можливість створення системи з реалізацією індивідуально користувацького досвіду. У діаграмі використано зв’язки типу «include» та «extend», які допомагають чітко структурувати логіку дій і гнучко описати поведінку системи в різних ситуаціях. Діаграма прецедентів відображає основний функціонал та варіанти взаємодії користувача з системою.

3.3 Діаграма класів

Діаграма класів – це вид діаграми у мові моделювання UML, що використовується для візуального подання структури програмної системи. Вона демонструє класи, їхні атрибути, методи, а також зв’язки між цими класами. Діаграма допомагає краще зрозуміти архітектуру системи, логіку взаємодії її компонентів та об’єктів, а також виявленню потенційних проблем на етапі проєктування. Такий підхід дозволяє отримати узагальнений вид системи ще до її реалізації.

Діаграма класів складається з прямокутників, кожен із яких представляє окремий клас. Прямокутник містить назву класу, його атрибути (властивості), а також методи (функції), що визначає його поведінку. Зв’язки між класами показують типи відносин, які можуть бути асоціаціями, агрегаціями, композиціями або спадкуванням. За допомогою цих зв’язків можна відстежити, як дані

передаються між різними компонентами. Така структура дає можливість детально описати та показати структуру взаємозв’язків системи [19].

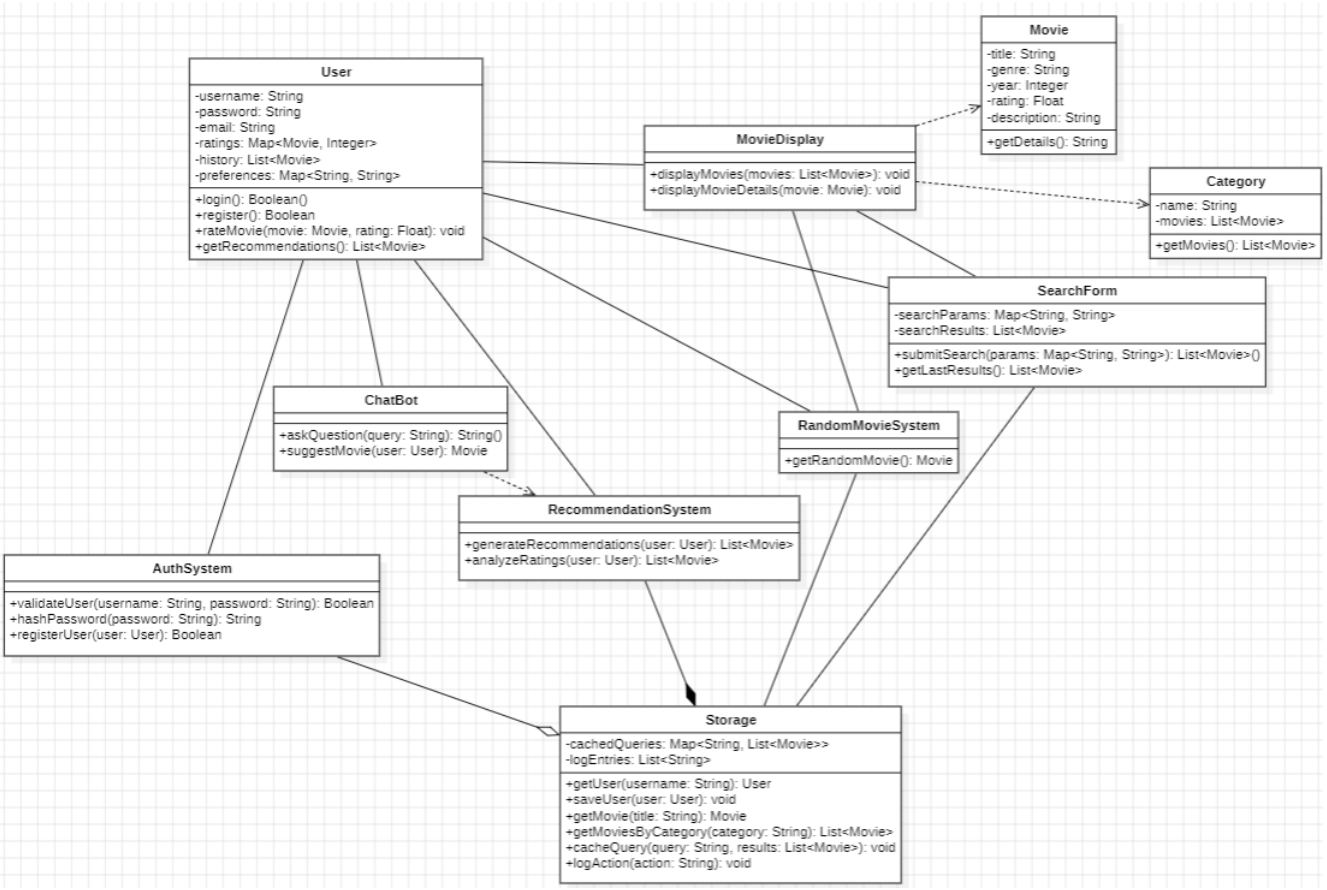

Рисунок 3.2 – Діаграма класів

Діаграма класів відображає основні компоненти системи вебсайту для пошуку фільмів із використанням штучного інтелекту. Клас User відповідає за зберігання особистих даних користувача, його історії пошуку фільмів, вподобань та виставлених оцінок. Саме цей клас ініціює взаємодію з іншими частинами системи, зокрема формуванням рекомендацій і пошуком фільмів. Компонент AuthSystem реалізує функції реєстрації, авторизації та перевірки облікових даних, забезпечуючи безпечний доступ до вебсайту. Система рекомендацій представлена класом RecommendationSystem та обробляє дані, що надходять від користувача, створюючи персоналізовані добірки фільмів. Для інтерактивної взаємодії користувача з системою створено клас ChatBot, який відповідає на запити та може запропонувати фільм на основі вхідної інформації. Клас RandomMovieSystem забезпечує можливість генерації випадкового фільму. Пошук фільмів за критеріями

реалізовано через клас SearchForm, а відображення результатів та деталей окремих фільмів реалізовано через клас MovieDisplay. Інформація про фільми зберігається у класі Movie, який включає назву, жанр, рік випуску, опис та рейтинг. Для об’єднання контенту за тематиками або жанрами використовується клас Category. Клас Storage забезпечує збереження та отримання інформації, реалізує хешування результатів пошуку та логування дій користувача. Класи мають різні типи зв’язків: асоціації, агрегації та залежності.

3.4 Діаграма станів та переходів

Діаграма станів та переходів – це вид діаграми у мові моделювання UML, що використовується для детального опису поведінки системи залежно від зміни її внутрішнього стану. Діаграма допомагає зрозуміти, як система реагує на певні події, які дії виконує в кожному стані та як відбувається перехід між цими станами. Дані діаграми часто застосовуються для моделювання об’єктно-орієнтованих систем, де поведінка змінюється в залежності від взаємодії з користувачем або іншими компонентами системи [20].

Діаграма станів та переходів складається з основних елементів, таких як:

– стани – відображають поточне положення системи або її складових;
– переходи – події або дії, що призводять до зміни стану;
– умови – визначають, за яких обставин відбувається перехід;
– дії – визначають, що відбувається під час перебування в стані або під час переходу.

Загальна діаграма станів і переходів (рис. 3.3) описує процес функціонування вебсайту з використанням технологій штучного інтелекту для пошуку фільмів. Початковим є стан Waiting, який відповідає початковій неактивній фазі та очікуванню дій користувача. Після цього система переходить до станів, що відповідають вибору користувачем певного способу пошуку: введення запиту через форму (FillingForm), вибір категорії (EnteringCategory) або звернення до віртуального помічника (EnteringBotInteraction). Ці дії ініціюють обробку запиту, після чого система відображає результати або деталі вибраного фільму. Відповідні

стани DisplayingResults, DisplayingBotResponse та ViewingMovieDetails демонструють результати взаємодії.

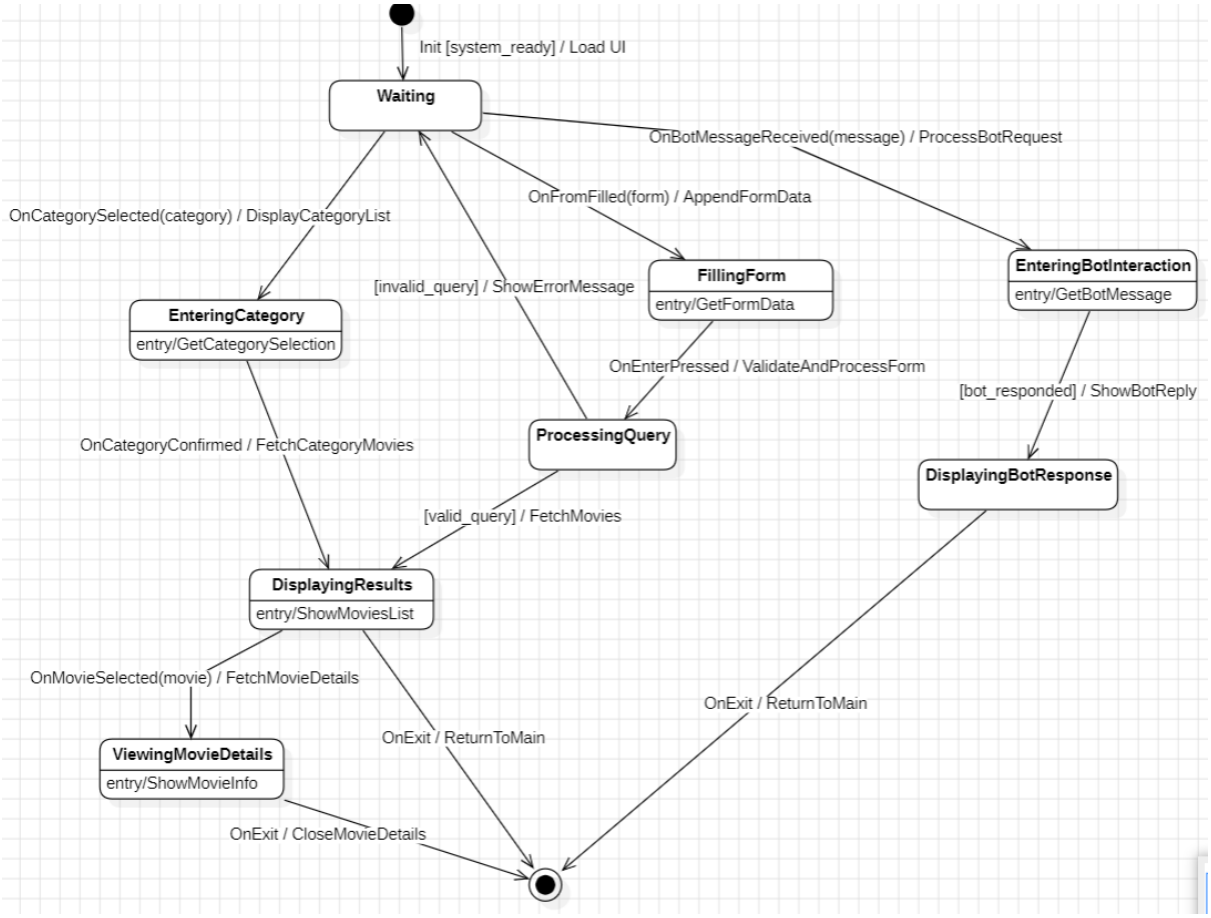

Рисунок 3.3 – Діаграма станів та переходів для опису процесу функціонування системи

Діаграма станів і переходів (рис. 3.4) відображає логіку роботи рекомендаційної системи, яка підбирає фільми відповідно до вподобань користувача. Процес починається зі стану Waiting, після чого система переходить до аналізу наявних даних (AnalyzingUserData). Цей стан визначає, чи має система достатньо інформації для створення рекомендацій. Якщо даних досить, виконується формування списку фільмів (GeneratingRecommendations), який згодом виводиться у вигляді результатів (DisplayingRecommendations). Далі користувач має змогу переглянути детальну інформацію про конкретний фільм у стані ViewingMovieDetails.

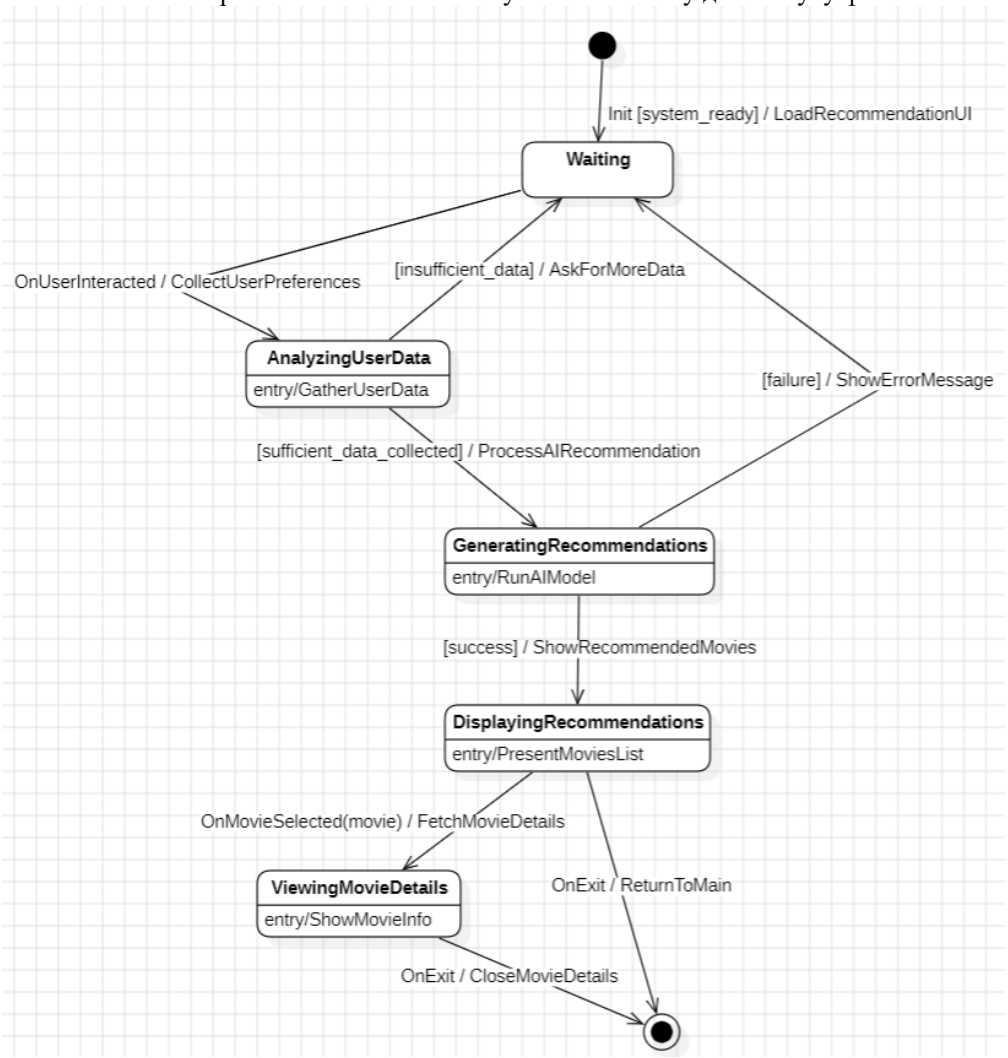

Рисунок 3.4 – Діаграма станів та переходів для опису рекомендаційної системи

Такі діаграми дозволяють краще зрозуміти внутрішню логіку роботи вебсайту, показують послідовність дій, взаємозв’язки між подіями та станами, а також забезпечують візуальне представлення при аналізі функціональності. Діаграми станів допомагають визначити точки прийняття рішень системою та можливі варіанти поведінки при зміні вхідних умов. Вони дають змогу прогнозувати поведінку системи в нестандартних ситуаціях, таких як відсутність даних, некоректні запити або збої при роботі з API.

Висновки до розділу 3

У третьому розділі кваліфікаційної бакалаврської роботи здійснено моделювання вебзастосунку для рекомендації фільмів із використанням сучасних засобів візуального проєктування. Розроблено сценарії використання, які

визначають основні дії користувача та реакції системи під час взаємодії. Такі сценарії описують як типові, так і альтернативні способи виконання функції, що дозволяє врахувати різні варіанти поведінки в межах одного прецеденту.

На основі сценаріїв використання побудовано діаграму прецедентів. Вона відображає основні функціональні можливості системи та демонструє зв’язки компонентами і користувачем. Дана діаграма візуалізує вимоги до програмного забезпечення та формує ролі акторів для взаємодії із системою.

Для структурного представлення системи створено діаграму класів. Вона демонструє основні об’єкти, що складають програмну модель. Діаграма включає класи, їхні атрибути, методи та типи зв’язків, які забезпечують обмін даними між різними частинами застосунку. Діаграма є основою для реалізації програмної логіки і допомагає при масштабуванні системи.

Побудовано діаграми станів і переходів для моделювання динамічної поведінки застосунку. Вони описують зміни станів системи залежно від дій користувача, внутрішніх подій чи зовнішніх умов. Даний тип діаграм дозволяє чітко уявити життєвий цикл окремих об’єктів, послідовність переходів і реакцій на тригери.

Створенні діаграми та описані сценарії використання є основою для подальшої розробки вебзастосунку, оскільки дають змогу краще зрозуміти, як працює система. Завдяки діаграмам легше побачити, як усе пов’язано між собою, як саме користувач взаємодіє із системою, і чи всі функції враховано. Також ці діаграми дозволяють помітити можливі помилки або слабкі місця ще до початку реалізації.

4 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Розробка системи

Основною вебсайту з використанням технології штучного інтелекту для пошуку фільмів є клієнт-серверна архітектура. Така реалізація забезпечує чітке розділення функціональності між клієнтською та серверною частинами системи. Обмін даними у системі відбувається через REST API з використанням HTTP-запитів у форматі JSON.

Клієнтська частина реалізована з використанням бібліотеки React. Вона відповідає за взаємодію користувача із системою, а саме, введення запитів, виведення результатів, взаємодія з віртуальним помічником, можливість реєстрації та авторизації. Серверна частина реалізована на основі Spring Boot та виконує функції обробки запитів, з базою даних PostgreSQL, генерації персоналізованих рекомендацій та інтеграції зі сторонніми API.

На головній сторінці вебзастосунку створено форму для пошуку фільмів, яку користувач повинен заповнити за особистими вподобаннями. Після заповнення та надсилання форми HTTP-запит звертається до контролера, який виконує запит до TMDb API для отримання списку фільмів (рис. 4.1). Також реалізовано функцію отримання випадкової рекомендації фільму при натисканні на кнопку «Випадковий фільм», яка також надсилає запит на отримання фільму з TMDb API. Результати запитів динамічно відображаються на сторінці.

```
const fetchRecommendedMovies = async (formData) => {
  const res = await fetch("/api/movies/recommend", {
    method: "POST",
    headers: {
      "Content-Type": "application/json"
    },
    body: JSON.stringify(formData)
  });
  const data = await res.json();
  setMovies(data);
};
```

Рисунок 4.1 – Функція для надсилання запиту на отримання списку фільмів

Основною функцією вебзастосунку є взаємодія з віртуальним помічником. Ця функція працює за допомогою штучного інтелекту і дозволяє отримувати персоналізовані рекомендації у вільному форматі. У клієнтській частині реалізовано текстове поле для введення особистих побажань, при надсиланні якого виконується HTTP-запит на сервер. На серверній частині запит передається до методу, що взаємодіє з OpenAI API (рис. 4.2). Результат повертається у вигляді відповіді до інтерфейсу користувача.

```java
@Service
public class OpenAiService {
    2 usages
    private final RestTemplate restTemplate;
    1 usage
    private final String OPENAI_API_URL = "https://api.openai.com/v1/chat/completions";
    1 usage
    private final String OPENAI_API_KEY = "Bearer sk-proj-7KDgoXdmr93T4b3V-JtDDK5nHsykccXpJnq1kyUwXp23ELuF
    public OpenAiService(RestTemplateBuilder builder) {
        this.restTemplate = builder.build();
    }
    public String generateReply(String message) {
        HttpHeaders headers = new HttpHeaders();
        headers.setContentType(MediaType.APPLICATION_JSON);
        headers.set("Authorization", OPENAI_API_KEY);
        Map<String, Object> requestBody = new HashMap<>();
        requestBody.put("model", "gpt-3.5-turbo");
        List<Map<String, String>> messages = List.of(
                Map.of( k1: "role",  v1: "user",  k2: "content", message)
        );
        requestBody.put("messages", messages);
        requestBody.put("temperature", 0.7);
        HttpEntity<Map<String, Object>> request = new HttpEntity<>(requestBody, headers);
        ResponseEntity<Map> response = restTemplate.postForEntity(OPENAI_API_URL, request, Map.class);
        List<Map<String, Object>> choices = (List<Map<String, Object>>) response.getBody().get("choices");
        if (choices != null && !choices.isEmpty()) {
            Map<String, Object> messageMap = (Map<String, Object>) choices.get(0).get("message");
            return (String) messageMap.get("content");
        }
        return "Sorry, I couldn't find a recommendation.";
    }
```

Рисунок 4.2 – Сервіс взаємодії з OpenAI API

Персоналізовані рекомендації формуються після тривалого використання вебзастосунком авторизованими користувачами. Система запам’ятовує, які фільми шукав користувач, і на основі цих даних формує список можливого контенту. Коли користувач хоче отримати персональні рекомендації, система звертається до бази даних, де зберігається історія його активності. Ці вподобання використовуються для створення спеціального запиту, який надсилається до штучного інтелекту. OpenAI API аналізує отримані дані та повертає список фільмів, які можуть сподобатися саме цьому користувачу (рис. 4.3).

```
public String generateRecommendation(String userId) {
    List<String> preferences = userHistoryRepository.findRecentPreferencesByUserId(userId);

    if (preferences == null || preferences.isEmpty()) {
        return "There is currently no data to recommend. Please check out something popular.";
    }

    int maxPreferences = 10;
    List<String> limitedPreferences = preferences.stream()
            .limit(maxPreferences)
            .toList();

    String prompt = String.format(
            "The user was interested in movies: %s. Suggest 3 personalized movies they might like.",
            String.join( delimiter: ", ", limitedPreferences)
    );

    try {
        return openAiService.ask(prompt);
    } catch (Exception e) {
        log.error("Error generating recommendations for userId: " + userId, e);
        return "Recommendations could not be generated. Please try again later.";
    }
}
```

Рисунок 4.3 – Метод сервісу який генерує персоналізовані рекомендації фільмів

У системі також реалізована стандартна реєстрація та авторизація (рис. 4.4). Користувач може заповнити необхідні дані та створити новий обліковий запис абсолютно безкоштовно. Вся інформація зберігаються у базі даних із підтримкою шифрування паролю для гарантування безпеки користувачів.

```
@PostMapping("/signin")
public ResponseEntity<?> loginUser(@RequestBody LoginRequest loginRequest) {
    Optional<User> userOptional = userRepository.findByEmail(loginRequest.getEmail());
    if (userOptional.isEmpty()) {
        return ResponseEntity.status(HttpStatus.UNAUTHORIZED)
                .body(Map.of( k1: "error",  v1: "User not found"));
    }
    User user = userOptional.get();
    if (!passwordEncoder.matches(loginRequest.getPassword(), user.getPassword())) {
        return ResponseEntity.status(HttpStatus.UNAUTHORIZED)
                .body(Map.of( k1: "error",  v1: "Invalid credentials"));
    }
    return ResponseEntity.ok(Map.of(
            k1: "message",  v1: "Login successful",
            k2: "username", user.getUsername(),
            k3: "email", user.getEmail()
    ));
}

@PostMapping("/signup")
public ResponseEntity<?> signup(@RequestBody SignupRequest request) {
    try {
        User user = userService.registerUser(request.getUsername(), request.getEmail(), request.getPassword());
        return ResponseEntity.ok( body: "User registered");
    } catch (IllegalArgumentException e) {
        return ResponseEntity.badRequest().body(Map.of( k1: "error", e.getMessage()));
    }
}
```

Рисунок 4.4 – Метод, що оброблює реєстрацію та авторизацію користувачів

Крім того, реалізована підтримка авторизації через акаунт Google. Ця функція працює завдяки інструменту Google OAuth (рис. 4.5). Він дозволяє користувачам швидко і безпечно авторизуватися без необхідності створювати окремий пароль для сайту.

```
@PostMapping("/google")
public ResponseEntity<?> googleAuth(@RequestBody Map<String, String> payload) {
    String email = payload.get("email");
    String username = payload.get("username");

    Optional<User> userOpt = userRepository.findByEmail(email);

    User user;
    if (userOpt.isPresent()) {
        user = userOpt.get();
    } else {
        user = new User();
        user.setEmail(email);
        user.setUsername(username);
        user.setPassword("");
        userRepository.save(user);
    }

    return ResponseEntity.ok(Map.of(
            k1: "message", v1: "Google login successful",
            k2: "username", user.getUsername(),
            k3: "email", user.getEmail()
    ));
}
```

Рисунок 4.5 – Метод, що оброблює запит на автентифікацію користувача через Google

Створено мінімалістичний, сучасний, адаптивний та інтуїтивно зрозумілий дизайн інтерфейсу зі зручною навігацією. Перехід між різними сторінками вебсайту здійснюється швидко та з мінімальною можливістю збою в системі. Усі основні компоненти інтерфейсу структуровані у вигляді окремих React-компонентів, що спрощує розширення системи.

4.2 Керівництво користувача

Керівництво користувача пояснює як саме відбувається взаємодія з вебзастосунком на всіх ключових етапах. У ньому покроково описано, що потрібно зробити, щоб повноцінно користуватися системою, а також всіма можливостями після реєстрації. Керівництво дозволяє новим користувачам швидко розібратися в

роботі сайту, структурі сервісу та використовувати всі функції. Саме інтерфейс створює перше враження про вебзастосунок і впливає на те, наскільки зручно та зрозуміло буде ним користуватися у подальшому.

Реєстрація користувача.

Після переходу на головну сторінку сайту користувач, який ще не має облікового запису, може зареєструватися (рис. 4.6). Для створення акаунту необхідно ввести ім’я користувача, електрону пошту, створити пароль і підтвердити його. Усі поля є обов’язковими для заповнення. Після підтвердження форми акаунт створюється та всі дані зберігаються у базі даних. Користувач також може скористатися швидкою реєстрацією через Google. Цей варіант є зручною альтернативою та потребує значно менше часу для реалізації, лише достатньо натиснути кнопку Sign up with Google та дозволити доступ до свого акаунту.

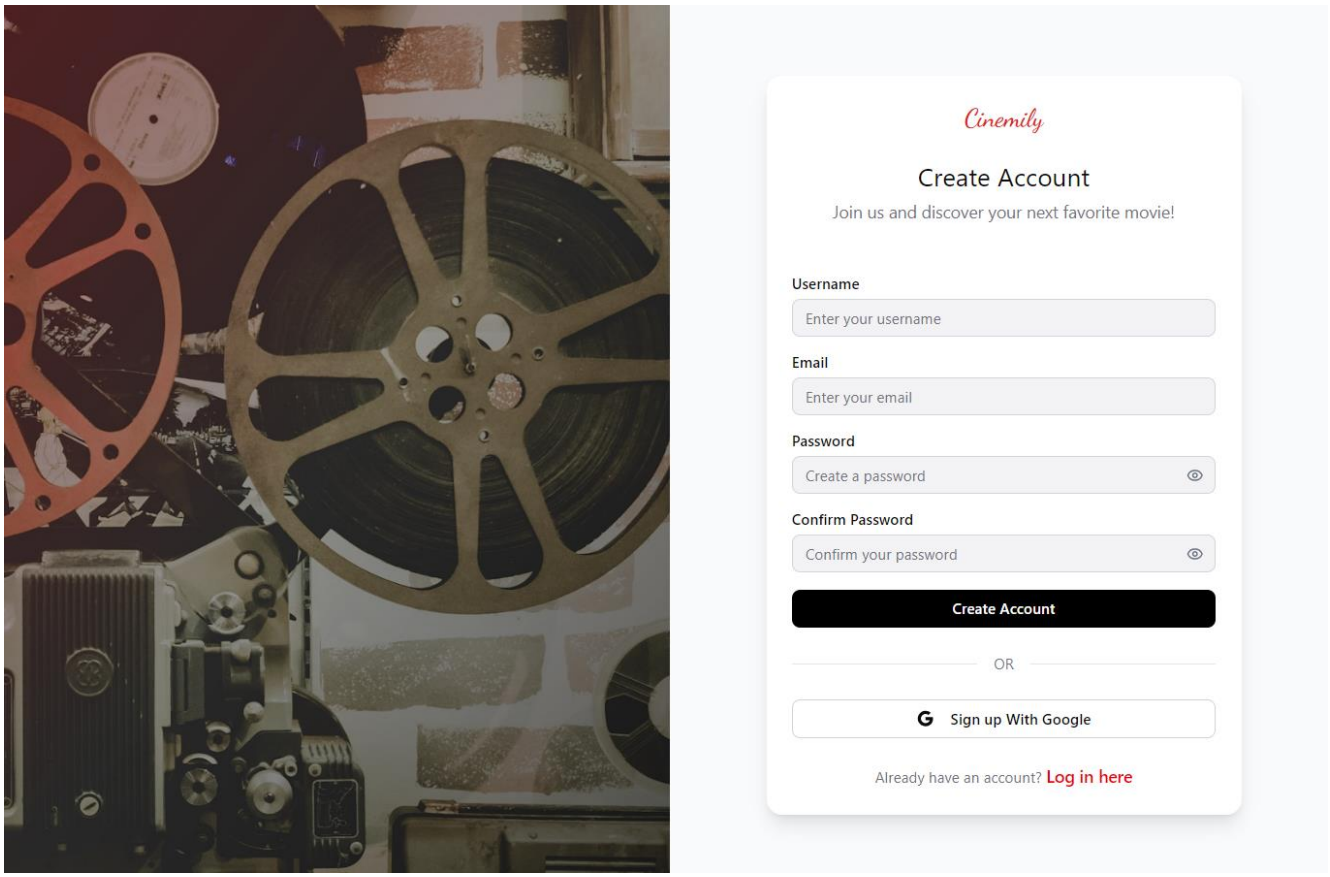

Рисунок 4.6 – Форма реєстрації

Авторизація користувача.

Якщо користувач вже має обліковий запис, то йому необхідно перейти на форму входу (рис. 4.7). Для авторизації потрібно ввести зареєстровану електрону пошту та пароль. При правильному введені даних користувач переходить на головну сторінку вебсайту та має доступ до розширеного функціоналу. Також доступна авторизація через Google. Якщо акаунт було зареєстровано відповідним способом, то система автоматично здійснить вхід до облікового запису.

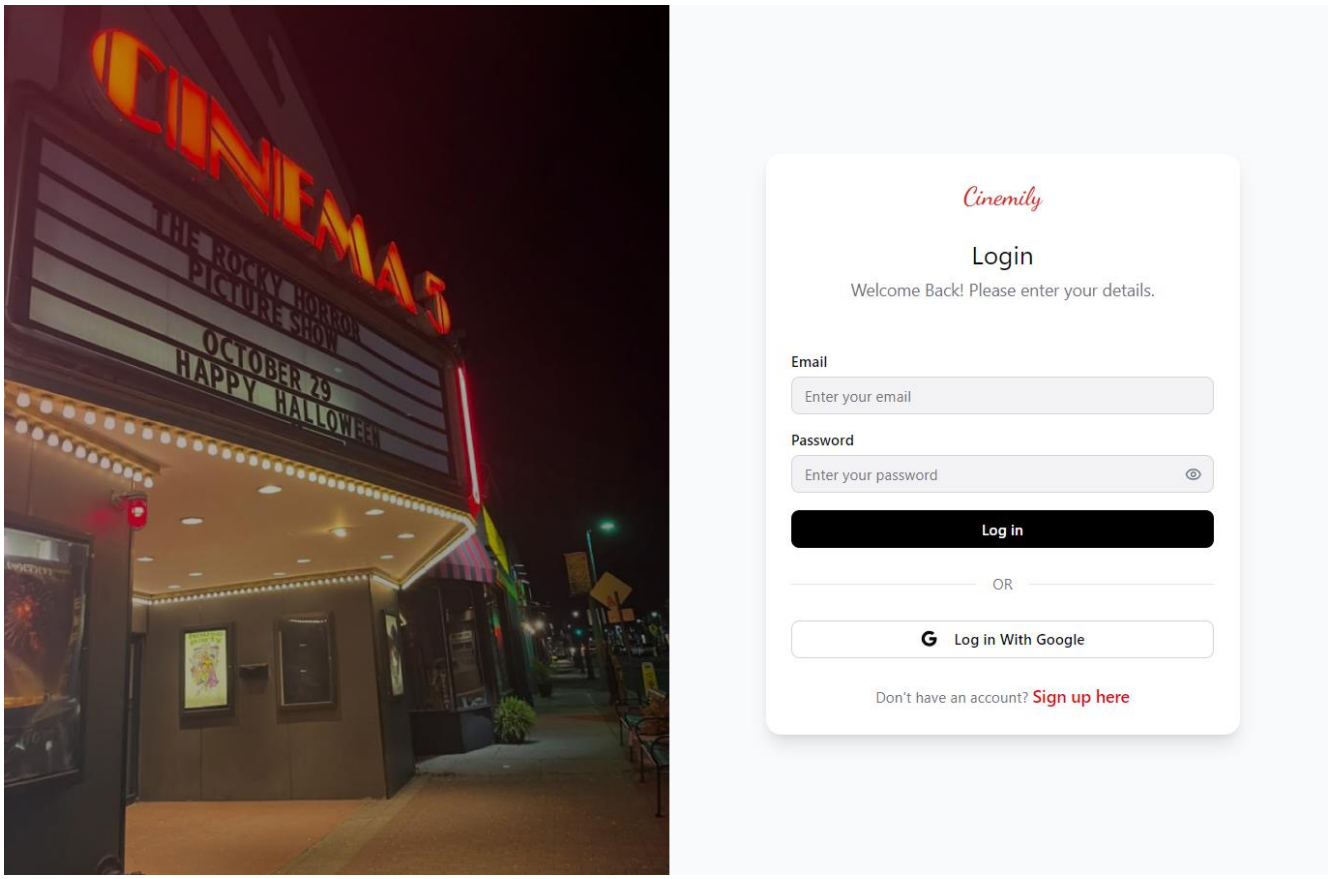

Рисунок 4.7 – Форма авторизації

Головна сторінка та форма пошуку.

Після успішного входу користувач автоматично повертається на головну сторінку, яка містить зручну форму для пошуку фільмів (рис. 4.8). Користувач може обрати жанр, рік випуску та настрій. За зазначеними фільтрами здійснюється пошук в системі. Також на головній сторінці розташована кнопка випадкового вибору, яка дозволяє отримати рекомендацію фільму незалежно від вподобань. Система згенерує один фільм без урахування заданих фільтрів. Після заповнення

форми або використання кнопки випадкової генерації, результати одразу відображаються на цій самій сторінці. Кожен фільм зі згенерованого списку представлений карткою з постером, назвою, роком виходу і коротким описом.

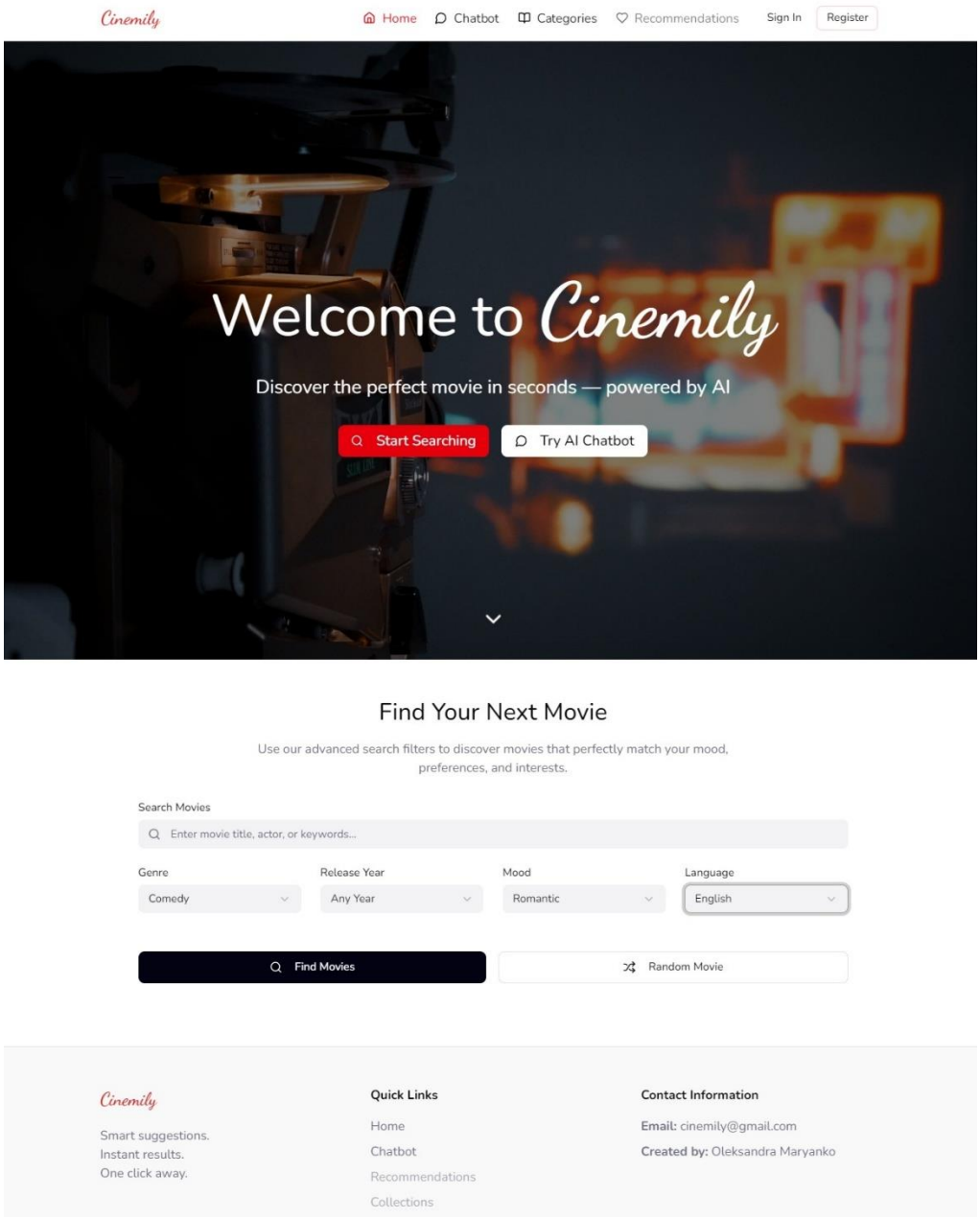

Рисунок 4.8 – Головна сторінка вебсайту

Віртуальний помічник і персоналізовані рекомендації.

Крім класичного пошуку, користувач може скористатися допомогою віртуального помічника (рис. 4.9). У спеціальній формі можна ввести запит у довільній формі або обрати один із готових варіантів пошуку. Система на основі штучного інтелекту обробляє запит і пропонує фільми, які найбільше відповідають

описаним побажанням. Ця функція доступна як зареєстрованим, так і незареєстрованим користувачам. Проте лише авторизовані користувачі можуть отримувати персоналізовані рекомендації. Система враховує історію попереднього пошуку, що дозволяє точніше формувати добірки фільмів відповідно до індивідуальних вподобань.

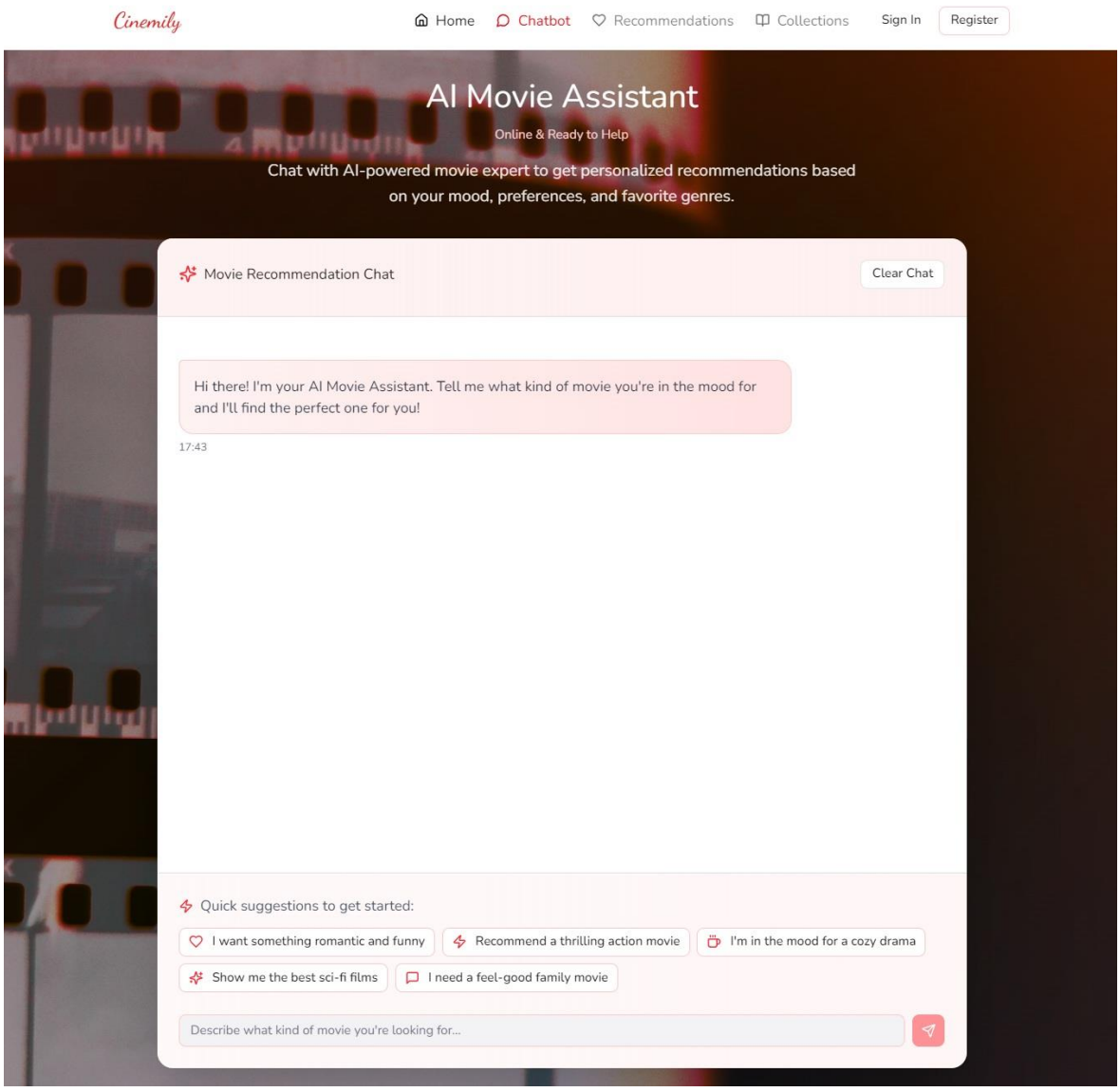

Рисунок 4.9 – Чат з віртуальним помічником

Тематичні добірки фільмів.

На окремій сторінці вебзастосунку визначено тематичні добірки фільмів, які згруповані за жанрами, настроєм та темами (рис. 4.10). Готові списки допомагають швидко знайти щось цікаве для перегляду без необхідності вводити запит чи

обирати фільтри. Користувач може перейти до будь-якої категорії, ознайомитися з її вмістом і вибрати фільм із представленого списку.

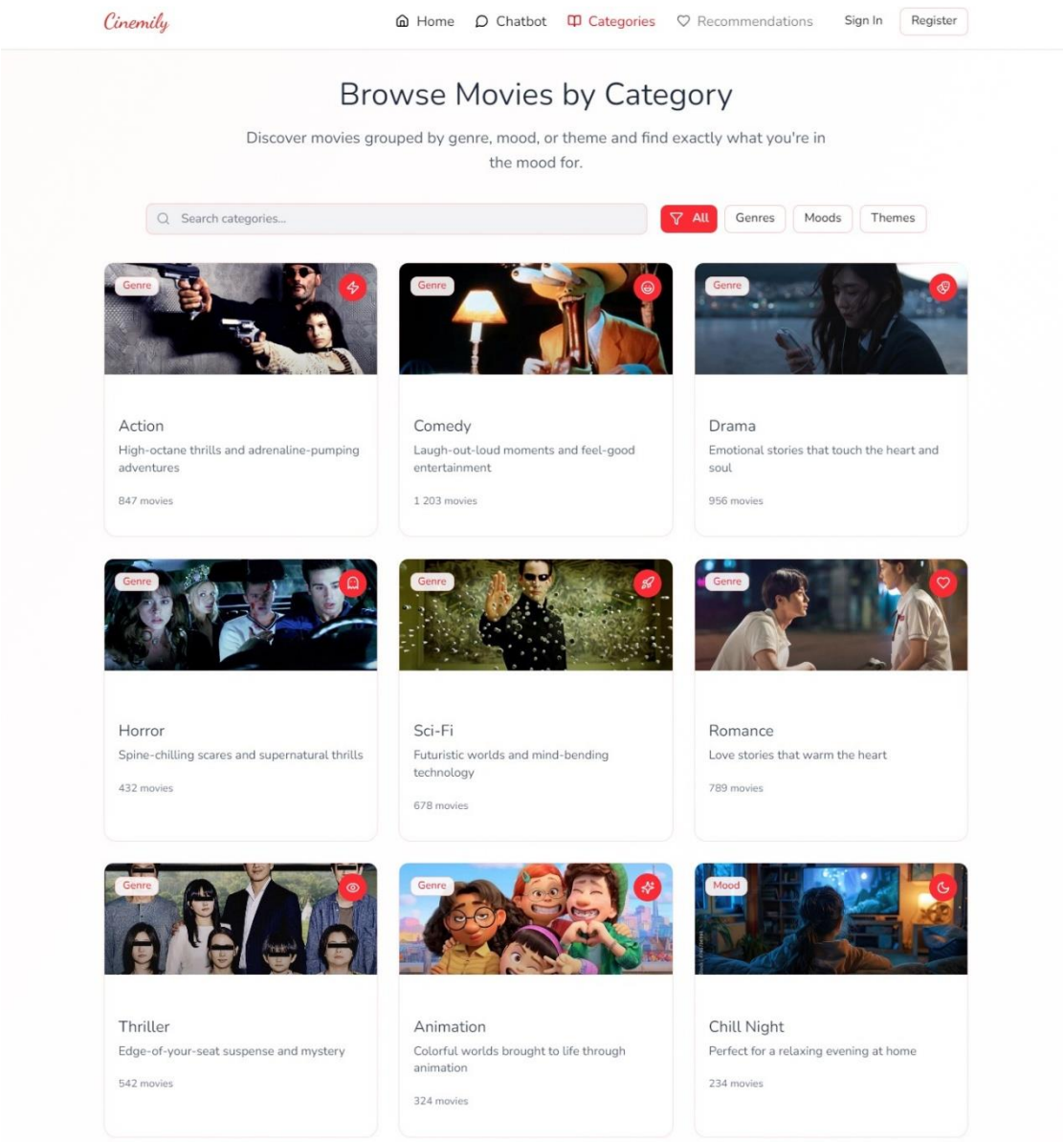

Рисунок 4.10 – Сторінка із добірками фільмів

Після вибору фільму з добірки користувач може переглянути детальну інформацію (рис. 4.11). Для цього необхідно натиснути на картку фільму. На екрані відкриється модальне вікно з усією основною інформацією: назва, рік випуску, жанр, рейтинг, настрій і короткий опис сюжету. Додано посилання на перегляд трейлеру до фільму.

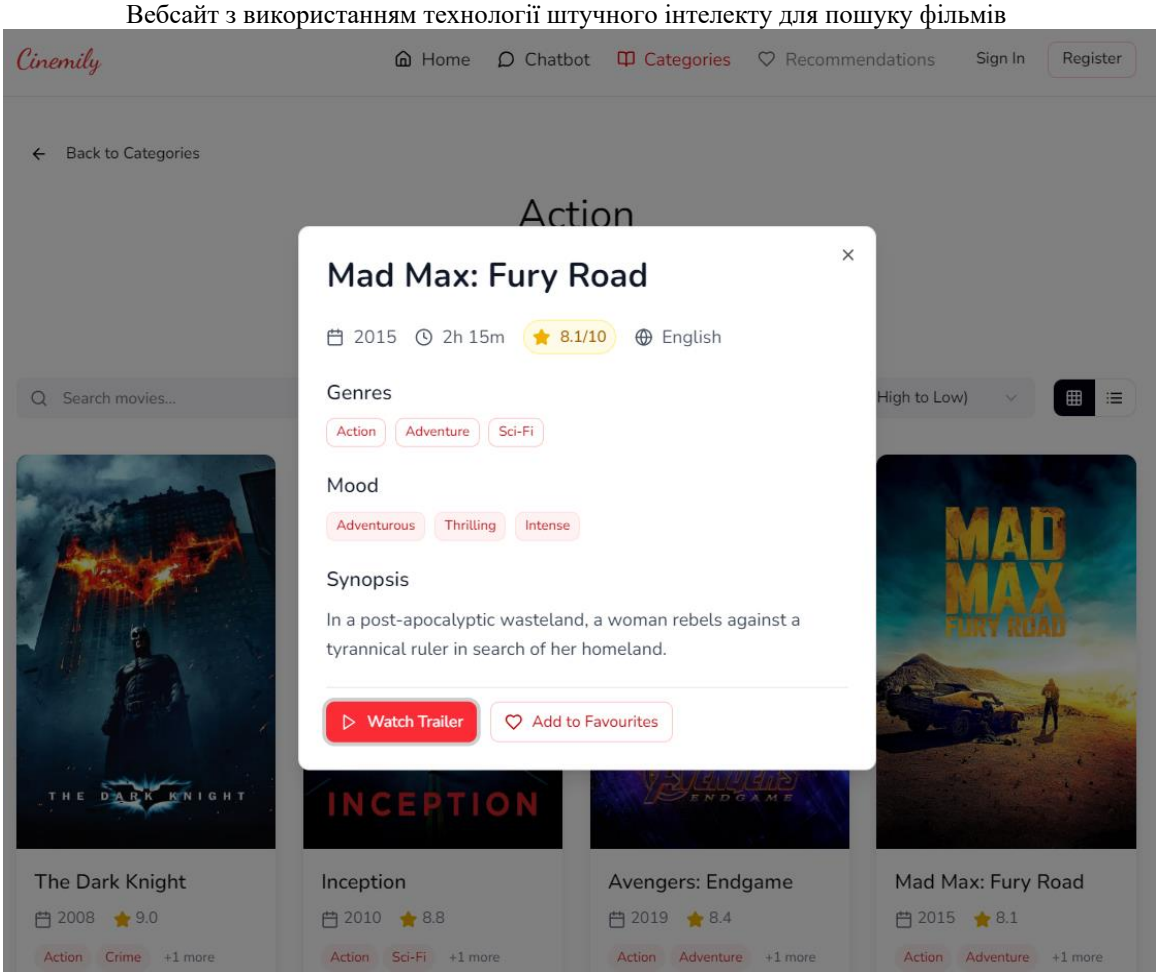

Рисунок 4.11 – Перегляд детальної інформації до фільму

Сторінка персоналізованих рекомендацій.

Сторінка персоналізованих рекомендацій доступна лише для авторизованих користувачів (рис. 4.12). На основі попередніх запитів, обраних жанрів та переглядів система формує унікальний список фільмів для кожного користувача. Інтерфейс сторінки побудований у вигляді зручного списку, де кожна картка містить постер, назву, жанр, настрій, рік випуску та короткий опис до кожного фільму. При натисканні на картку відкривається модальне вікно з детальною інформацією. Алгоритм рекомендацій постійно оновлюються відповідно до збереженої інформації. Користувача також може позначати фільми як вподобані. Це дозволяє системі точніше формувати персональні добірки.

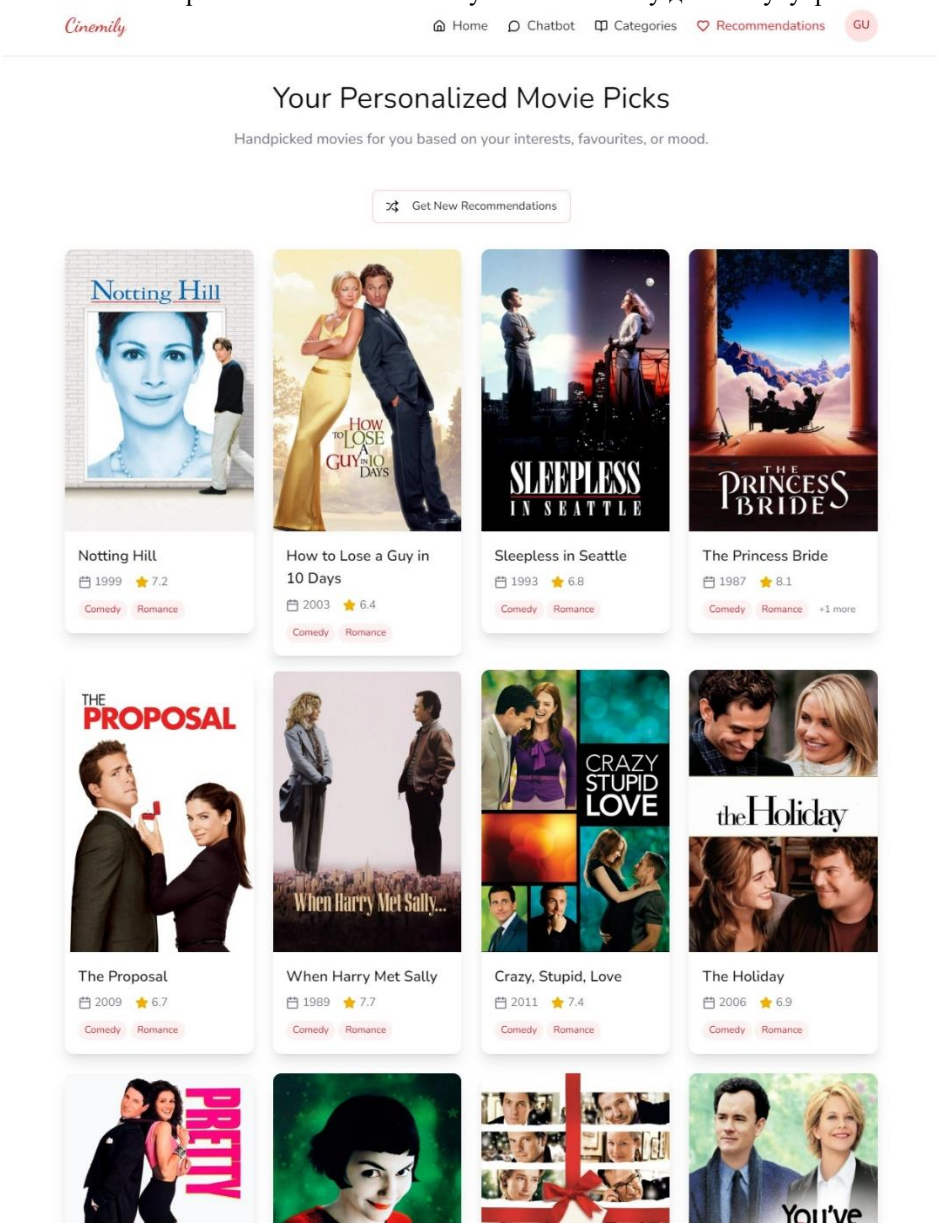

Рисунок 4.12 – Сторінка із добірками фільмів

Вебзастосунок є послідовним, інтуїтивним та логічно побудованим. Зрозумілий інтерфейс, зручне розташування елементів і можливості вибору між різними способами пошуку (заповнення форми, спілкування з віртуальним помічником або перегляд тематичних добірок) допомагає користувачу швидко знайти фільм, що відповідатиме особистим запитам. Система враховує індивідуальні вподобання та формує персоналізовані рекомендації. Інтерфейс оформлено в одному мінімалістичному стилі з червоними акцентами, що виділяють важливі елементи.

Висновки до розділу 4

У четвертому розділі кваліфікаційної бакалаврської роботи представлено реалізацію програмного забезпечення. Розроблено як клієнтську, так і серверну частини системи. Створено функціональний вебсайту, що забезпечує швидкий та зручний пошук фільмів, орієнтований на покращення користувацького досвіду.

Для розробки застосунку обрано сучасні технологій. Клієнтська частина відповідає за відображення інформації та взаємодію з користувачем, а серверна частина – за обробку даних, зберігання інформації та логіку роботи системи. Особливістю розробленої системи є інтеграція з технологіями штучного інтелекту, що дозволяє формувати персоналізовані рекомендації та індивідуальний підхід до користувача.

Користувачі можуть здійснювати реєстрацію та авторизацію в особистому кабінеті, що дає змогу зберігати вподобання, історію пошуку та особисті добірки фільмів. Реалізовано чат з віртуальним помічником, який допомагає швидко знаходити інформацію у довільній формі. Така функція підвищує зручність взаємодії із системою. У розділі також наведено детальне керівництво користувача, яке допомагає швидко ознайомитися з основними функціями застосунку та ефективно ними користуватися.

ВИСНОВКИ

Під час виконання кваліфікаційної бакалаврської роботи на тему «Вебсайт з використанням технології штучного інтелекту для пошуку фільмів» досягнуто поставленої мети шляхом вирішення усіх завдань, визначених у вступі, а саме:

– проведено аналіз предметної області;
– здійснено огляд та аналіз існуючих аналогів вебсайтів;
– розроблено специфікацію вимог до вебсайту;
– виконано моделювання та проєктування програмного забезпечення;
– реалізовано front-end з урахуванням зручності використання;
– розроблено back-end з використанням штучного інтелекту для персоналізованого пошуку;
– описано керівництво користувача.

Проведено аналіз предметної області та визначено сучасні тенденції у сфері пошуку фільмів. Досліджено особливості роботи існуючих аналогів, таких як TasteDive, Movie Decider і Taste.io. Виявлено їхні переваги та недоліки, обґрунтувати вибір функціоналу вебзастосунку. Визначення штучного інтелекту як ключового елементу для формування персоналізованих рекомендацій.

Виконано моделювання структури системи за допомогою діаграм прецедентів, класів, станів та переходів. Реалізовано проєктування логіки застосунку та його компонентів. Для реалізації програмного забезпечення обрано сучасні технології та інструменти. React використано для клієнтської частини, Spring Boot – для серверної логіки, PostgreSQL – для зберігання даних, а також OpenAI API і TMDb API – для реалізації пошуку. Створено зручний інтерфейс з легкою навігацією, функціями пошуку, добірок фільмів і взаємодії з віртуальним помічником. Розроблений вебсайт надає користувачам зручний доступ до пошуку фільмів, персоналізованих рекомендацій, взаємодії з віртуальним помічником та перегляду добірок за категоріями.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Огляд і порівняння популярних стримінгових сервісів. *Полемікс*. URL: https://polemix.com.ua/oglyad-i-porivnyannya-populyarnyh-strimingovyh-servisiv/ (дата звернення: 20.03.2025).

2. Коноплицький С. Парадокс вибору. *PSYSK*. URL: https://psysk.com/paradoks-vyboru/ (дата звернення: 23.04.2025).

3. Український психологічний ХАБ. Фрустрація це. *Український психологічний ХАБ*. URL: https://www.psykholoh.com/post/фрустрація-це (дата звернення: 23.04.2025).

4. Система навігації сайту та особливості її створення. *Gl.ua – розробка та підтримка сайтів*. URL: https://gl.ua/blog/systema-navihatsiyi-saytu-ta-osoblyvosti-yiyi-stvorennya?page=8 (дата звернення: 23.04.2025).

5. TasteDive | Recommends music, movies, TV shows, books, games, people, places, brands and podcasts. *TasteDive*. URL: https://tastedive.com/ (date of access: 23.04.2025).

6. Movie Decider. *Movie Decider - The Fun Movie Recommendation Engine*. URL: https://moviedecider.com/ (date of access: 24.04.2025).

7. Taste. *Taste - Movie and TV show recommendations from likeminded people.* URL: https://www.taste.io/ (date of access: 24.04.2025).

8. Movie Popularity and Target Audience Prediction using the Content-Based Recommender System / S. Sahu et al. *IEEE Access*. 2022. P. 1. URL: https://doi.org/10.1109/access.2022.3168161 (date of access: 27.04.2025).

9. OpenAI developer platform. *OpenAI*. URL: https://platform.openai.com/docs/overview (date of access: 01.05.2025).

10. Getting Started. *The Movie Database (TMDB)*. URL: https://developer.themoviedb.org/docs/getting-started (date of access: 01.05.2025).

11. Full Stack Development with Spring Boot and React: Build Modern and Scalable Web Applications Using the Power of Java and React. de Gruyter GmbH, Walter, 2022.

12. Що таке React JS і для чого він потрібен? *DAN.IT*. URL: https://dan-it.com.ua/uk/blog/chto-takoe-react-js-i-dlja-chego-on-nuzhen/ (дата звернення: 01.05.2025).

13. Фреймворк Spring та його особливості. *Highload.tech*. URL: https://highload.tech/uk/frejmvork-spring-ta-jogo-osoblivosti/ (дата звернення: 01.05.2025).

14. База даних PostgreSQL: інформація та короткий гайд. *itProger*. URL: https://itproger.com/ua/news/baza-dannih-postgresql-informatsiya-i-kratkiy-gayd (дата звернення: 01.05.2025).

15. StarUML. *StarUML*. URL: https://staruml.io/ (date of access: 01.05.2025).

16. Diagram CASE Tool for Software Modeling & Analysis - UML, BPMN, ERD. *Software Ideas Modeler*. URL: https://www.softwareideas.net/ (date of access: 01.05.2025).

17. Brush K. What is a Use Case? *Search Software Quality*. URL: https://www.techtarget.com/searchsoftwarequality/definition/use-case (date of access: 02.05.2025).

18. Use Case Diagram – Unified Modeling Language (UML). *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/use-case-diagram/ (date of access: 02.05.2025).

19. Class Diagram – Unified Modeling Language (UML). *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/ (date of access: 02.05.2025).

20. State Transition Diagram. *Ingenious*. URL: https://ingenious.wiki.utwente.nl/statediagram (date of access: 02.05.2025).