МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КОМПЛЕКСНИЙ КУРСОВИЙ ПРОЄКТ
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Програмна система для рецензій фільмів. Back-end розробка
(тема)

Виконав:
здобувач 3 курсу, групи ПЗПІ-22-8
Володимир ЛОПАТЕНКО
(Власне ім'я, ПРІЗВИЩЕ)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник доц. кафедри ПІ Ірина ГРУЗДО
(посада, Власне ім'я, ПРІЗВИЩЕ)

Члени комісії (Власне ім'я, ПРІЗВИЩЕ, підпис)
Доц. Олексій ТУРУТА

Доц. Анастасія ЧУПРИНА

Ст. викл. Дмитро МАТВЄЄВ

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

Курс 3 Група ПЗПІ-22-8 Семестр 6

ЗАВДАННЯ
на курсовий проект(роботу) студента

здобувачеві Лопатенко Володимиру Володимировичу
(прізвище, ім’я, по батькові)

1. Тема роботи Програмна система для рецензій фільмів. Back-end розробка
2. Термін здачі студентом закінченої роботи „20” червня 2025 р.
3. Вихідні дані до проєкту *Розробити back-end частину веб-застосунку “Absolute Cinema” для підтримки функціоналу рецензій на фільми, обговорень у реальному часі, управління користувачами та модерації контенту. Використати JavaScript (Node.js + Express.js), Supabase для бази даних та аутентифікації, WebSocket для чату, JWT для захисту сесій, REST API для взаємодії з клієнтською частиною.*
4. Перелік питань, що потрібно опрацювати в роботі

Вступ, аналіз предметної галузі, виявлення та формулювання проблем, постановка задачі, UML-проєктування та архітектура програмного забезпечення, опис алгоритмів і методів реалізації, технічні рішення та особливості реалізації серверної частини (аутентифікація, чат, управління профілем, захист даних), аналіз результатів, висновки, список використаних джерел, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	15.03.2025	виконано
2	Розробка постановки задачі	26.03.2025	виконано
3	Проектування ПЗ	06.04.2025	виконано
4	Програмна реалізація	20.05.2025	виконано
5	Аналіз результатів	27.05.2025	виконано
6	Підготовка пояснювальної записки.	01.06.2025	виконано
7	Перевірка на наявність ознак академічного плагіату	11.06.2025	виконано
8	Захист роботи	20.06.2025	виконано

Дата видачі завдання “10” лютого 2025р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. кафедри ПІ Ірина ГРУЗДО
(підпис) (посада, Власне ім’я, ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до курсового проєкту містить: 78 стор., 11 рис., 23 джерел.

ВЕБОРІЄНТОВАНА СОЦІАЛЬНА ПЛАТФОРМА, ВЕБ-ЗАСТОСУНОК ДЛЯ РЕЦЕНЗІЙ ФІЛЬМІВ, CSS3, EXPRESS.JS, HTML5, JAVASCRIPT, JWT, NODE.JS, SUPABASE, WEBSOCKET

Об'єкт розробки – є проблема відсутності універсальної платформи для кіноманів, яка б поєднувала рецензування, соціальну взаємодію, персоналізовані рекомендації та реальний діалог між користувачами в одному застосунку. Предметом дослідження є розробка програмної системи (ПС) для управління кінематографічним контентом, користувацькими профілями, рецензіями, обговореннями та чатами у межах єдиної інтерактивної веб-платформи.

Актуальність теми дослідження обумовлена зростаючою потребою у цифрових платформах, які дозволяють кіноманам ділитися рецензіями, вести обговорення та отримувати персоналізовані рекомендації. Попри популярність кіноконтенту, в Україні відсутні комплексні рішення, що об'єднують функціональність рецензій, соціальних мереж та інтерактивного обговорення в реальному часі. Створення такої платформи дозволяє заповнити цю нішу, забезпечивши сучасний користувацький досвід за допомогою передових веб-технологій.

Мета розробки – створення сучасної веб-платформи “Absolute Cinema”, яка забезпечує можливості для написання та перегляду рецензій на фільми, обговорення кінематографічних творів, спілкування між користувачами в реальному часі та управління особистими профілями.

Методи рішення – використання технологій Node.js та Express.js для серверної частини, Supabase для управління базою даних та аутентифікації, WebSocket для реалізації чату в реальному часі, HTML5/CSS3/JavaScript для клієнтської частини, JWT для безпечної аутентифікації користувачів.

У результаті розробки створено повнофункціональний веб-застосунок, який включає систему аутентифікації та управління профілями користувачів, каталог фільмів з можливістю пошуку та рекомендацій, систему рецензій та рейтингів, інтерактивні обговорення з модерацією контенту, чат для спілкування в реальному часі, адміністративну панель та систему безпеки з захистом від CSRF-атак.

Практична цінність роботи полягає у створенні зручної, функціонально насиченої та адаптивної веб-платформи, яка є дешевою альтернативою існуючим системам. Вона дозволяє користувачам легко обмінюватися думками про фільми, залишати рецензії, оцінки, вести дискусії, а також знаходити однодумців. Застосунок має зрозумілий інтерфейс, забезпечує безпеку даних та підтримує роботу на різних пристроях.

ABSTRACT

WEB-ORIENTED SOCIAL PLATFORM, WEB APPLICATION FOR MOVIE REVIEWS, CSS3, EXPRESS.JS, HTML5, JAVASCRIPT, JWT, NODE.JS, SUPABASE, WEBSOCKET

Object of development – the lack of a universal platform for movie enthusiasts that combines reviewing, social interaction, personalized recommendations, and real-time dialogue in a single application. The subject of the research is the development of a software system (SS) for managing cinematographic content, user profiles, reviews, discussions, and chats within a unified interactive web platform.

Relevance of the topic is due to the growing demand for digital platforms that allow movie fans to share reviews, participate in discussions, and receive personalized recommendations. Despite the popularity of movie content, there are no comprehensive solutions in Ukraine that integrate reviewing functionality, social networking, and real-time interaction. The development of such a platform fills this gap by providing a modern user experience using advanced web technologies.

Purpose of the development – to create a modern web platform “Absolute Cinema” that provides functionality for writing and viewing movie reviews, discussing films, communicating with other users in real-time, and managing personal profiles.

Solution methods – the backend is developed using Node.js and Express.js, Supabase is used for database management and authentication, WebSocket is implemented for real-time chat, the client-side is developed with HTML5/CSS3/JavaScript, and JWT is used for secure user authentication.

As a result, a fully functional web application was created, including an authentication and user profile management system, a movie catalog with search and recommendation features, a review and rating system, moderated discussion threads, real-time chat, an admin panel, and a security system with CSRF protection.

Practical value of the work lies in providing a convenient, feature-rich, and adaptive web platform that is a low-cost alternative to existing systems. It enables users to easily share opinions about films, write reviews, rate content, participate in

discussions, and connect with like-minded individuals. The application has an intuitive interface, ensures data security, and supports operation across various devices.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

API – Application Programming Interface

CORS – Cross-Origin Resource Sharing

CSRF – Cross-Site Request Forgery

DB – Database – база даних

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

JWT – JSON Web Token

ORM – Object-Relational Mapping

REST – Representational State Transfer

SQL – Structured Query Language

UML – Unified Modeling Language

UX/UI – User Experience / User Interface

WS – WebSocket

XSS – Cross-Site Scripting

ВСТУП

Кінематограф займає особливе місце в сучасній культурі та мистецтві, виступаючи не лише засобом розваги, але й потужним інструментом впливу на свідомість, формування світогляду та суспільних цінностей. Фільми здатні викликати глибокі емоції, змушувати замислюватися над важливими питаннями життя, надихати на дії та змінювати погляди на світ. Кожен фільм несе в собі унікальне послання, художню цінність та культурне значення, що робить його предметом для аналізу, обговорення та критичної оцінки.

Сучасна кіноіндустрія характеризується надзвичайним різноманіттям жанрів, стилів та підходів до створення фільмів. Від блокбастерів до авторського кіно, від документальних стрічок до анімації – кожен напрямок має свою аудиторію та своїх прихильників. Така різноманітність створює потребу в платформах, де кіномани можуть ділитися своїми враженнями, аналізувати кінематографічні твори та знаходити однодумців для обговорення улюблених фільмів.

Розвиток цифрових технологій та інтернету кардинально змінив спосіб, яким люди споживають та обговорюють кіноконтент. Традиційні форми кінокритики, представлені в друкованих виданнях та телевізійних програмах, в останній час доповнюються онлайн-платформами, де кожен користувач може стати рецензентом та поділитися своєю думкою з широкою аудиторією. Це демократизує процес кінокритики та дозволяє формувати більш різноманітні та репрезентативні оцінки кінематографічних творів.

Важливість обміну думками про фільми полягає не лише в можливості поділитися враженнями, але й у розвитку критичного мислення, аналітичних здібностей та культурної грамотності. Обговорення фільмів сприяє глибшому розумінню кінематографічного мистецтва, допомагає виявляти приховані смисли та символіку, а також формує естетичні смаки та preferences глядачів.

Сучасні веб-технології надають унікальні можливості для створення інтерактивних платформ, що об'єднують функціонал рецензування, соціального

спілкування та персоналізації контенту. Використання таких технологій як Node.js, Express.js, Supabase та WebSocket дозволяє створювати динамічні, масштабовані та безпечні застосунки, які можуть задовольнити потреби сучасних користувачів у швидкому, зручному та безпечному обміні інформацією.

Метою роботи є створення комплексного веб-застосунку “Absolute Cinema”, який об'єднає функціонал рецензування фільмів, соціального спілкування кіноманів та персоналізованих рекомендацій. Розроблений застосунок має забезпечити зручну платформу для обміну думками про кінематографічні твори, формування спільнот за інтересами та підтримки активного діалогу між користувачами, сприяючи розвитку кінокультури та критичного мислення в цифровому середовищі.

Для реалізації поставленої мети використовується сучасний стек веб-технологій. Серверна частина розробляється з використанням Node.js як середовища виконання JavaScript та Express.js як веб-фреймворку для створення API. Управління базою даних та аутентифікацією користувачів здійснюється через Supabase, що забезпечує надійне зберігання даних та безпечний доступ до них. Для реалізації функціоналу чату в реальному часі застосовується технологія WebSocket, яка дозволяє миттєвий обмін повідомленнями між користувачами. Безпека додатку забезпечується за допомогою JSON Web Tokens (JWT) для управління сесіями, CORS для контролю кросс-доменних запитів, а також Cookie Parser та Body Parser для правильної обробки даних. Клієнтська частина реалізується з використанням стандартних веб-технологій HTML5, CSS3 та JavaScript, що гарантує сумісність з усіма сучасними браузерами та пристроями. Розробка проекту здійснювалася у середовищі Visual Studio Code, яке забезпечує потужні інструменти для веб-розробки, підтримку всіх використовуваних технологій та зручний інтерфейс для управління кодом.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

У сучасному цифровому світі кіноіндустрія переживає справжній ренесанс завдяки стрімкому розвитку стримінгових платформ та інтернет-технологій. Щорічно у світі випускається понад 10 тисяч фільмів різних жанрів та форматів, що створює величезний обсяг контенту для глядачів [7]**.** Однак разом із зростанням кількості кінопродукції зростає і потреба в якісних платформах для обговорення, оцінювання та рекомендацій фільмів.

Предметна галузь веб-застосунків для рецензій фільмів охоплює декілька ключових аспектів сучасної цифрової екосистеми кіно. По-перше, це соціальний аспект – потреба людей ділитися враженнями про переглянуті фільми, обговорювати сюжетні лінії, акторську гру та режисерську роботу. По-друге, це інформаційний аспект – необхідність систематизації великих обсягів даних про фільми, їх характеристики, рейтинги та відгуки. По-третє, це технологічний аспект – розробка надійних серверних рішень, які можуть обробляти великі навантаження та забезпечувати стабільну роботу веб-платформи.

Сучасні веб-застосунки для рецензій фільмів повинні вирішувати комплексні завдання: від забезпечення швидкого доступу до інформації про фільми до створення інтерактивного середовища для спілкування користувачів. Бекенд-частина таких застосунків відіграє критично важливу роль, оскільки вона забезпечує обробку користувацьких запитів, управління базами даних, аутентифікацію користувачів та інтеграцію з зовнішніми сервісами для отримання актуальної інформації про фільми.

Особливо важливим аспектом є реалізація систем реального часу для чатів та обговорень. Сучасні користувачі очікують миттєвої взаємодії, можливості обговорювати фільми “наживо” під час або відразу після перегляду. Це вимагає використання передових технологій, таких як WebSocket, для забезпечення стабільного двостороннього зв'язку між клієнтом та сервером.

Аналіз ринку показує, що найуспішніші платформи поєднують в собі функціональність соціальних мереж із спеціалізованими інструментами для

роботи з кінематографічним контентом. Користувачі цінують можливість не лише оцінити фільм, але й детально обговорити його з іншими кіноманами, створювати списки улюблених фільмів, отримувати персоналізовані рекомендації та відстежувати активність друзів.

1.2 Виявлення та вирішення проблем

Аналіз існуючих рішень у сфері веб-застосунків для рецензій фільмів виявляє ряд критичних проблем, які потребують вирішення в рамках розробки сучасної бекенд-системи.

Першою значною проблемою є недостатня швидкість завантаження та обробки великих обсягів даних. Багато існуючих платформ страждають від повільної роботи при обробці запитів користувачів, особливо в періоди підвищеного навантаження. Це призводить до погіршення користувацького досвіду та відтоку аудиторії. Для вирішення цієї проблеми необхідно реалізувати ефективні алгоритми кешування, оптимізувати запити до бази даних та використовувати сучасні підходи до управління навантаженням на сервер.

Другою критичною проблемою є відсутність якісної системи реального часу для спілкування користувачів. Більшість існуючих платформ обмежуються статичними коментарями без можливості живого обговорення фільмів. Це особливо актуально під час прем'єр великих фільмів, коли користувачі прагнуть миттєво поділитися враженнями. Вирішення цієї проблеми вимагає імплементації WebSocket-з'єднань та розробки надійної архітектури для обробки повідомлень у реальному часі.

Третьою проблемою є недосконалість систем рекомендацій. Багато платформ використовують примітивні алгоритми, які не враховують персональні уподобання користувачів або базуються лише на популярності фільмів. Це призводить до того, що користувачі отримують нерелевантні рекомендації та втрачають інтерес до платформи. Для розв'язання цієї проблеми необхідно розробити інтелектуальну систему аналізу користувацької поведінки та

впровадити алгоритми машинного навчання для генерації персоналізованих рекомендацій.

Четвертою важливою проблемою є питання безпеки та захисту користувацьких даних. З огляду на зростаючу кількість кібератак та підвищені вимоги до конфіденційності, сучасні веб-застосунки повинні забезпечувати найвищий рівень захисту [6]. Це включає безпечне зберігання паролів, захист від SQL-ін'єкцій, CSRF-атак та інших видів зловмисних дій.

П'ятою проблемою є складність інтеграції з зовнішніми сервісами для отримання актуальної інформації про фільми. Багато платформ покладаються на застарілі або ненадійні джерела даних, що призводить до неточної або неповної інформації про фільми. Вирішення цієї проблеми потребує розробки гнучкої системи інтеграції з множинними API кіносервісів та реалізації механізмів валідації та синхронізації даних.

1.2.1 Цільова аудиторія

Основною цільовою аудиторією веб-застосунку для рецензій фільмів є люди:

- які активно переглядають фільми різних жанрів та епох;
- які прагнуть ділитися власними думками та враженнями про кінематограф;
- які цікавляться думкою інших глядачів перед переглядом фільму;
- віком від 16 до 45 років із середнім та вище середнього рівнем доходу;
- які володіють базовими навичками користування інтернетом та соціальними мережами;
- які цінують якісний контент та готові витрачати час на детальні обговорення;
- професійні кінокритики та журналісти кіноіндустрії;
- студенти кіновузів та викладачі кінематографічних дисциплін; організатори кінофестивалів та кіноклубів;
- власники кінотеатрів та представники дистриб'юторських компаній.

Додатковою цільовою аудиторією є користувачі, які: шукають рекомендації фільмів на основі своїх уподобань; хочуть відстежувати новинки кіноіндустрії; прагнуть знайти однодумців серед кіноманів; цікавляться статистикою та трендами у світі кіно; потребують платформи для професійного спілкування у кіносфері.

1.2.2 Аналіз специфіки ринку України та інших регіонів

Український ринок кінематографу та веб-платформ для обговорення фільмів має свої унікальні особливості, зумовлені як економічними, так і культурними факторами. Лише за 2024 рік кількість годин перегляду українських стрічок зросла на 300 млн, що на 30% більше у порівнянні з попереднім роком [8], що свідчить про зростаючий інтерес до вітчизняного кінематографу.

Специфікою українського ринку є домінування безкоштовних онлайн-платформ з українською локалізацією, таких як UAkino, Mixfilm та інші, які задовольняють потребу користувачів у перегляді контенту українською мовою. Водночас спостерігається активне зростання інтересу до стрімінгових сервісів, особливо Netflix, що адаптують свій контент під українську аудиторію.

Важливою особливістю є високий рівень піратства, який змушує розробників веб-застосунків для рецензій фільмів конкурувати не лише з офіційними платформами, але й з неліцензійними ресурсами. Це створює потребу в розробці максимально зручних та функціональних рішень, які б могли залучити користувачів якістю сервісу та унікальними можливостями для спілкування про кіно.

Регіональна специфіка також проявляється у мовних преференціях користувачів. Українська аудиторія активно шукає контент українською мовою, що створює нішу для локалізованих платформ. Крім того, існує значний попит на обговорення як голлівудських блокбастерів, так і європейського арт-хаусного кіно.

У порівнянні з західними ринками, український сегмент характеризується меншою готовністю користувачів платити за преміум-функції веб-сервісів, що

вимагає розробки гнучких моделей монетизації та акцентування на рекламних доходах. Водночас молода аудіторія демонструє високу активність у соціальних мережах та готовність до активного обговорення кінематографічного контенту, що створює сприятливі умови для розвитку платформ з соціальними функціями.

1.3 Аналіз аналогів програмного забезпечення

Для комплексного розуміння вимог до розробки веб-застосунку для рецензій фільмів було проведено детальний аналіз основних конкурентів на ринку. Розглянуто як міжнародні платформи, так і локальні українські проекти.

Для аналізу було обрано п’ять популярних систем створення списків фільмів, які мають широку аудиторію та значний вплив на індустрію.

Оцінювання проводилося за такими критеріями:

- функціональність – повнота набору функцій для роботи з фільмами та рецензіями;
- соціальні можливості – наявність чатів, коментарів, обговорень між користувачами;
- база даних контенту – обсяг та якість інформації про фільми;
- швидкодія – час завантаження сторінок та відгук системи;
- персоналізація – можливості налаштування під потреби користувача;
- модерація контенту – якість контролю за публікаціями користувачів.

Аналіз показав наступні характеристики основних аналогів:

IMDb [1] є найавторитетнішою базою даних фільмів у світі, заснованою в 1990 році. Платформа надає вичерпну інформацію про кінематографічні твори та є стандартом індустрії для професіоналів кіно (див. рис. 1.3.1).

а) переваги платформи IMDb:

1. найбільша база даних фільмів у світі з понад 8 мільйонами назв;
2. детальна інформація про акторів, режисерів та технічний персонал;
3. інтеграція з професійною кіноіндустрією та офіційними джерелами;
4. високий авторитет серед професіоналів кіно;

5. розвинена система рейтингів на основі мільйонів голосів користувачів;

б) недоліки платформи IMDb:

1. застарілий інтерфейс користувача, який не відповідає сучасним стандартам UX/UI;
2. обмежені можливості для інтерактивного спілкування між користувачами;
3. відсутність якісної системи чату та обговорень у реальному часі;
4. складність навігації для пересічних користувачів.

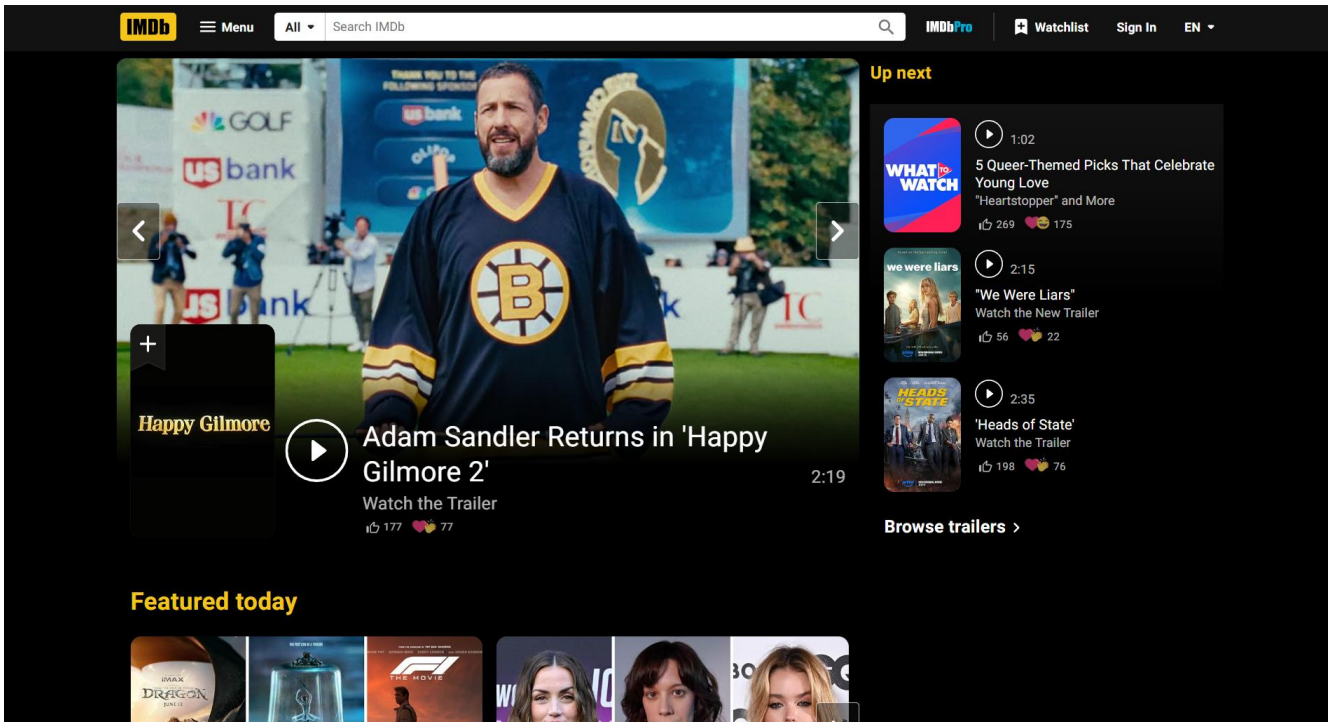


Рисунок 1.3.1 – Головна сторінка IMDb [1]

Letterboxd [2] – це соціальна мережа для кіноманів, заснована в 2011 році, яка поєднує ведення щоденника переглядів з соціальними функціями та естетично привабливим дизайном (див. рис. 1.3.2).

в) переваги платформи Letterboxd:

1. сучасний та естетично привабливий дизайн;
2. активна спільнота кіноманів з якісними рецензіями;
3. зручна система створення списків фільмів;

4. соціальні функції для відстеження активності друзів;

г) недоліки платформи Letterboxd:

1. обмежена база даних порівняно з IMDb;
2. повільна швидкість завантаження сторінок;
3. відсутність повноцінної системи чату;

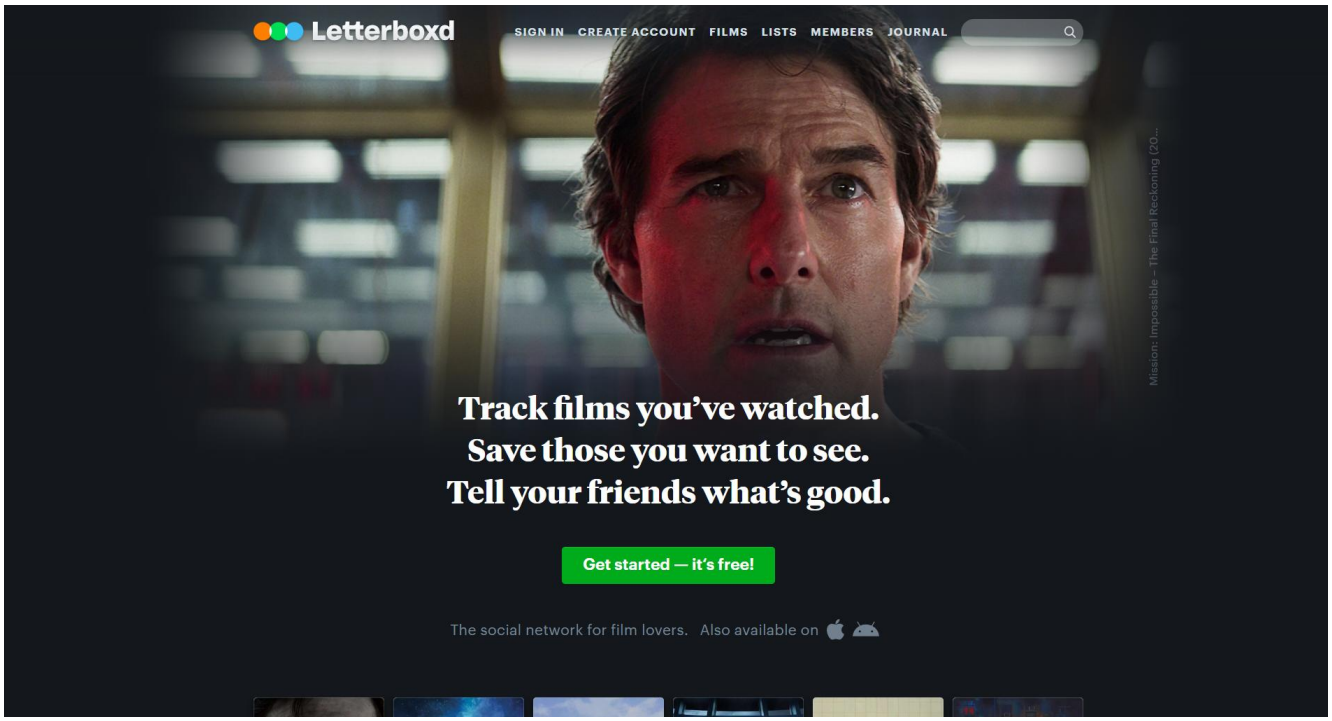


Рисунок 1.3.2 – Головна сторінка Letterboxd [2]

Rotten Tomatoes [3] – американський агрегатор рецензій, заснований в 1998 році, відомий своєю унікальною системою оцінювання "свіжий/гнилий" та розділенням оцінок критиків і глядачів (див. рис. 1.3.3).

д) переваги платформи Rotten Tomatoes:

1. унікальна система оцінювання з розділенням на критиків та глядачів;
2. зручний агрегатор рецензій з професійних видань;
3. широка популярність серед масової аудиторії;
4. інтеграція з великими кіностудіями та дистриб'юторами;
5. система сертифікації "Fresh" та "Rotten" для швидкої оцінки якості;

е) недоліки платформи Rotten Tomatoes:

1. обмежені соціальні функції для взаємодії користувачів;

2. відсутність системи особистих рекомендацій;
3. слабкі можливості для створення власних списків фільмів;
4. недостатня глибина аналізу фільмів поза рейтингами.

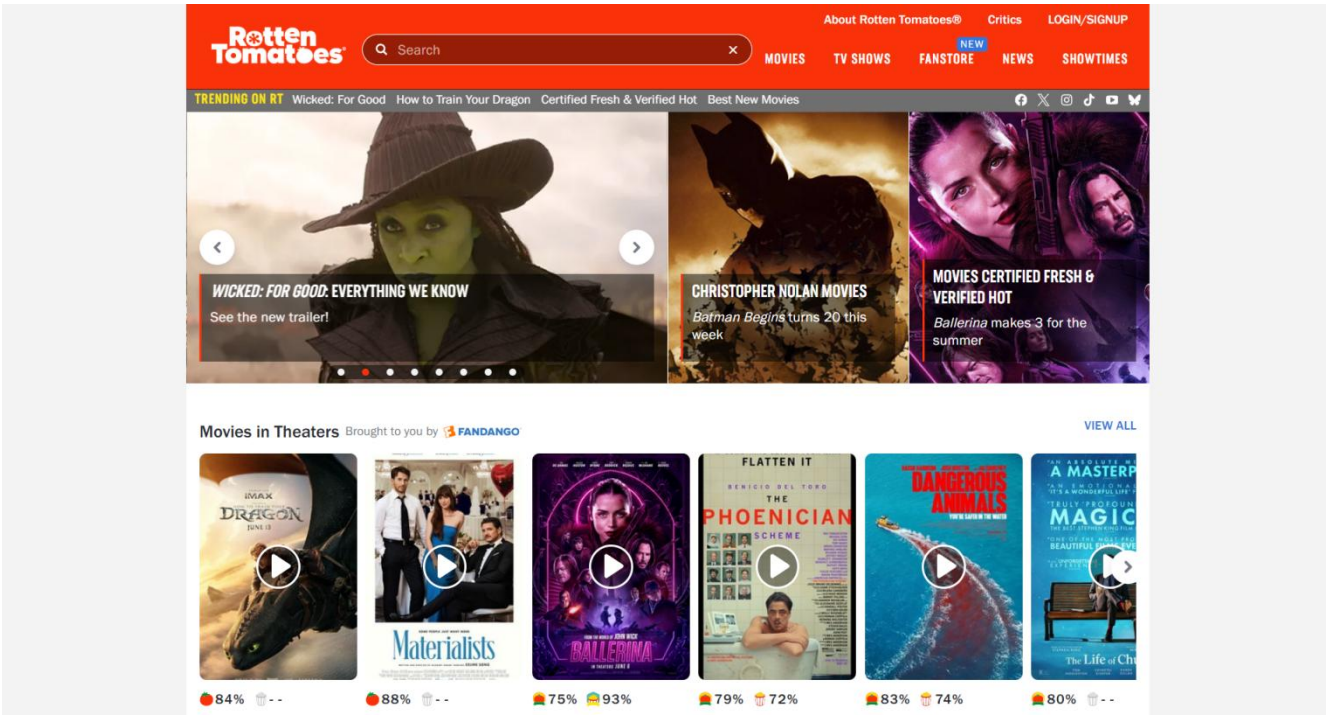


Рисунок 1.3.3 – Головна сторінка Rotten Tomatoes [3]

Metacritic [4] – платформа агрегації рецензій, заснована в 2001 році, яка спеціалізується на зваженому оцінюванні від професійних критиків та охоплює фільми, ігри та музику (див. рис. 1.3.4).

є) переваги платформи Metacritic:

1. професійна система зваженого оцінювання від критиків;
2. охоплення не лише фільмів, але й ігор та музики;
3. детальні метадані про кожну рецензію;
4. високий рівень довіри серед професійних критиків;
5. прозора методологія розрахунку рейтингів;

ж) недоліки платформи Metacritic:

1. мінімальні можливості для користувацької взаємодії;
2. відсутність системи коментарів та обговорень;
3. обмежена база даних менш популярних фільмів;

4. застарілий дизайн інтерфейсу.

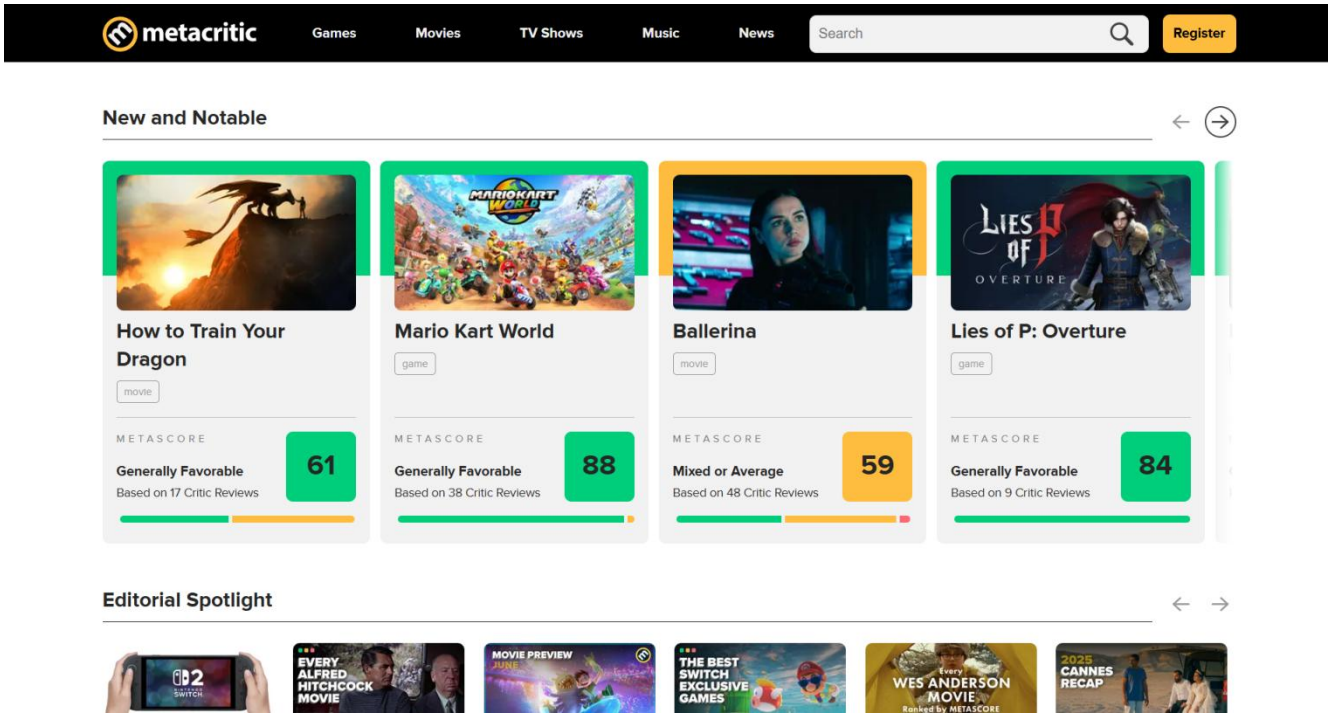


Рисунок 1.3.4 – Головна сторінка Metacritic [4]

Trakt [5] – сервіс для відстеження переглянутого контенту, заснований в 2010 році, який автоматично синхронізується з медіаплеєрами та надає детальну статистику перегляду (див. рис. 1.3.5).

з) переваги платформи Trakt:

1. потужна система трекінгу переглянутого контенту;
2. автоматична синхронізація з різними медіаплеєрами;
3. детальна статистика перегляду для користувачів;
4. активна спільнота з обміном рекомендаціями;
5. підтримка як фільмів, так і серіалів;

и) недоліки платформи Trakt:

1. складний для новачків інтерфейс з надмірною кількістю функцій;
2. відсутність повноцінної системи рецензій;
3. обмежені можливості для групових обговорень;
4. слабка інтеграція з соціальними мережами.

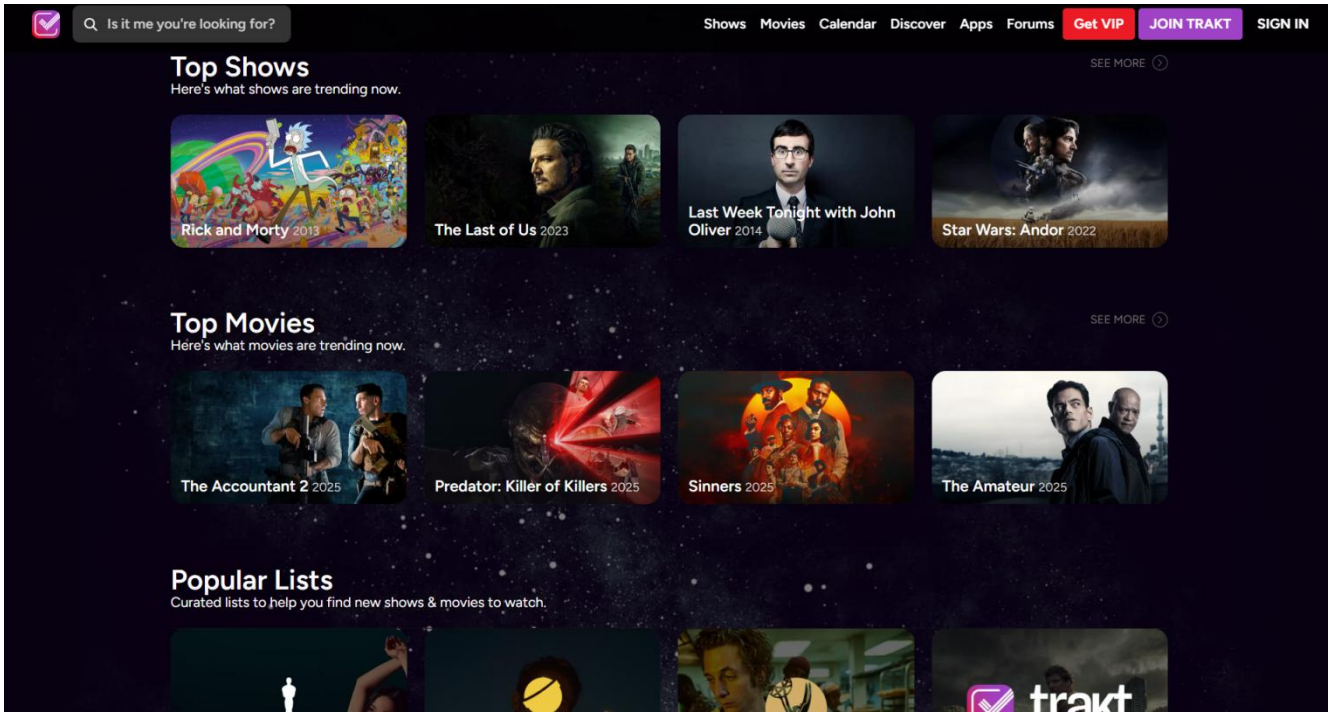


Рисунок 1.3.5 – Головна сторінка Trakt [5]

Аналіз функціональних можливостей показав, що існуючі рішення мають спільні недоліки: відсутність повноцінних систем реального часу для спілкування; обмежені можливості персоналізації рекомендацій; недостатню швидкість роботи при великому навантаженні; слабку інтеграцію соціальних функцій з основним функціоналом платформи; відсутність гнучких інструментів модерації контенту; обмежені можливості для створення тематичних спільнот та груп.

На основі проведеного аналізу було визначено ключові вимоги до розробки власного веб-застосунку. Система повинна поєднувати найкращі риси існуючих рішень, усуваючи виявлені недоліки. Особлива увага має бути приділена розробці надійної бекенд-архітектури, яка забезпечить швидку обробку запитів, ефективне управління даними та стабільну роботу систем реального часу.

Таким чином, розробка веб-застосунку для рецензій є актуальним завданням, яке дозволить заповнити прогалини існуючих рішень та запропонувати користувачам інноваційний продукт для спілкування про кіно.

2 ПОСТАНОВКА ЗАДАЧІ

У сучасному світі кіноіндустрії зростає потреба у спеціалізованих платформах, де кіномани можуть ділитися враженнями про переглянуті фільми, отримувати рекомендації та спілкуватися з однодумцями. Тому актуальним завданням є розробка серверної частини веб-застосунку “Absolute Cinema”, який забезпечить надійну роботу системи рецензування фільмів та соціальної взаємодії користувачів.

Метою даної роботи є створення бекенд системи для веб-платформи кіноманів, яка забезпечить ефективну обробку запитів клієнтської частини, управління даними користувачів та фільмів, а також реалізацію комунікації в реальному часі. Для досягнення поставленої мети визначено наступні завдання:

- дослідити сучасні підходи до розробки серверних застосунків та архітектурних рішень для веб-платформ;
- проаналізувати методи інтеграції з хмарними сервісами та системами управління даними;
- обрати оптимальну архітектуру для веб-застосунку відповідно до вимог;
- спроектувати архітектуру серверної частини для забезпечення ефективної взаємодії компонентів системи;
- розробити бізнес-логіку застосунку з урахуванням специфіки предметної області;
- створити систему управління даними з підтримкою операцій створення, читання, оновлення та видалення;
- впровадити механізми забезпечення безпеки та контролю доступу до ресурсів системи;
- розробити систему обробки запитів в реальному часі для підтримки інтерактивної взаємодії;
- забезпечити масштабованість та продуктивність серверної частини під різними навантаженнями.

Серверна частина системи має забезпечувати обробку різноманітних типів запитів від клієнтських додатків, управління сесіями користувачів, зберігання та

отримання інформації з бази даних, а також підтримку комунікації в реальному часі. Архітектура бекенду повинна передбачати модульну структуру з чітким розділенням відповідальності між компонентами системи.

Основною задачею розробки бекенд системи є створення надійної, масштабованої та безпечної серверної інфраструктури, яка забезпечить якісну роботу веб-платформи для кіноманів. Система повинна ефективно обробляти запити користувачів, забезпечувати швидкий доступ до даних та підтримувати одночасну роботу великої кількості користувачів без втрати продуктивності.

Детальна специфікація програмного забезпечення (SRS) з повним описом функціональних та нефункціональних вимог до серверної системи наведена у Додатку А цієї пояснювальної записки.

3 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 UML проєктування ПЗ

Перед початком розробки веб-застосунку для рецензій фільмів “Absolute Cinema” було проведено детальне проєктування архітектури системи з використанням UML діаграм. Це дозволило визначити основні компоненти системи, їх взаємодію та послідовність виконання операцій, особливо з точки зору серверної частини застосунку. UML діаграми стали фундаментом для створення масштабованої та безпечної архітектури, яка забезпечує всі необхідні функції веб-платформи для кіноманів.

Основою для проєктування серверної архітектури стала діаграма активностей, яка демонструє основні бізнес-процеси системи з акцентом на серверну обробку запитів. Діаграма активностей (див. рис. 3.1.1) показує послідовність дій від початкового запиту користувача до формування відповіді сервером, включаючи всі критичні точки прийняття рішень та можливі шляхи виконання.

Діаграма активностей відображає складну логіку серверної обробки запитів, починаючи з прийняття запиту від клієнта. Перший етап включає валідацію прав доступу через спеціалізовані middleware функції Express.js, які перевіряють наявність та дійсність JWT токенів. Цей процес є критично важливим для забезпечення безпеки всієї системи, оскільки дозволяє розмежувати доступ до різних ресурсів залежно від ролі користувача.

Процес аутентифікації включає взаємодію з Supabase Auth сервісом, який виконує верифікацію користувачів та управління сесіями. Система підтримує автоматичне оновлення токенів для забезпечення безперервної роботи користувачів без необхідності повторного входу в систему. При цьому кожен токен має обмежений час життя, що підвищує рівень безпеки застосунку.

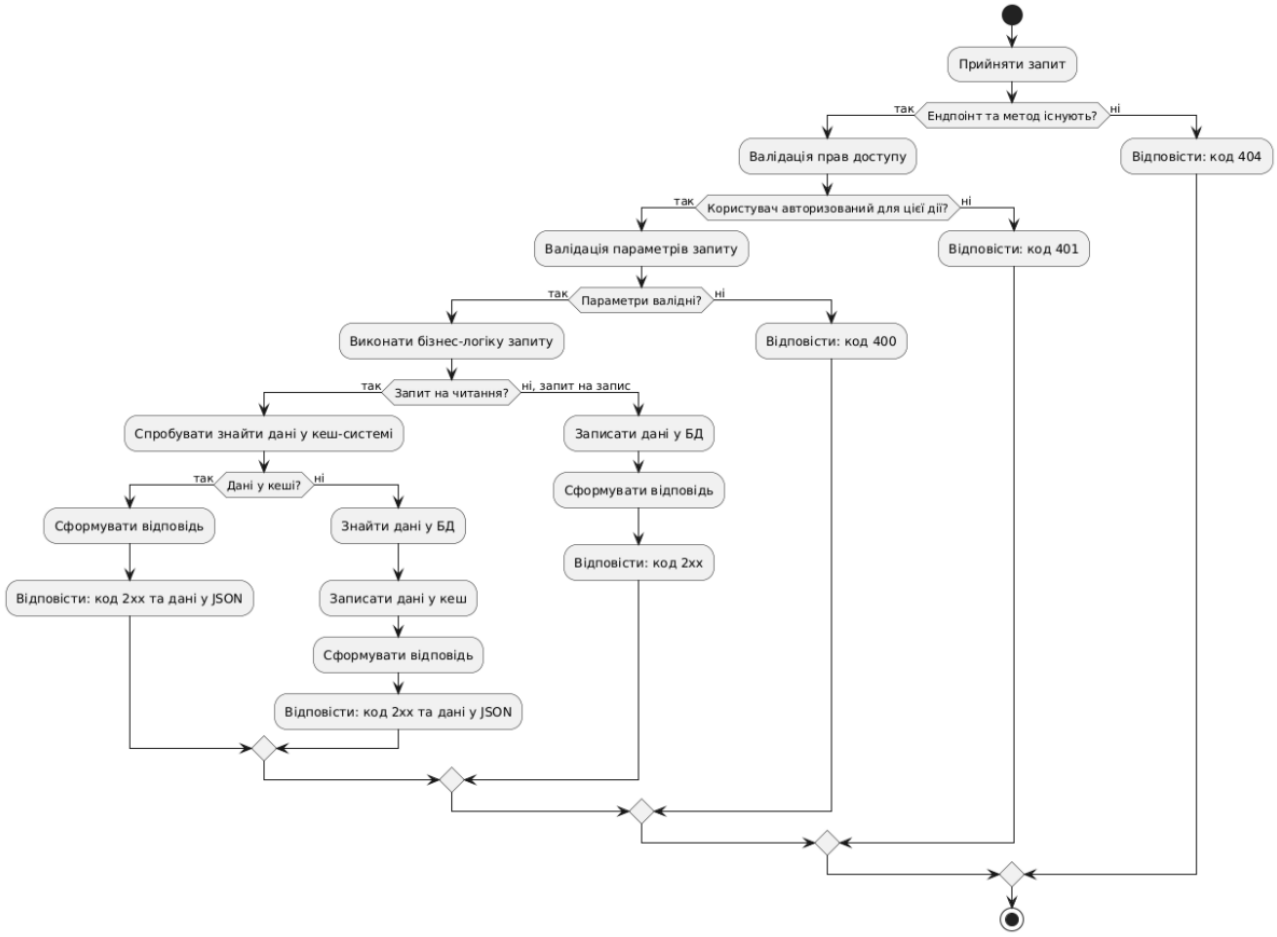


Рисунок 3.1.1 – Діаграма активностей веб-застосунку (рисунок виконаний самостійно)

Серверна валідація параметрів запиту є наступним критичним етапом, який включає перевірку типів даних, обов'язкових полів, обмежень довжини та формату. Всі вхідні дані проходять санітизацію для запобігання SQL-ін'єкціям, XSS-атакам та інших видів зловмисних дій. Система також перевіряє відповідність параметрів бізнес-правилам застосунку, таким як унікальність email адрес, складність паролів та валідність рейтингів фільмів.

Обробка бізнес-логіки включає взаємодію з базою даних через Supabase клієнт, виконання складних запитів з JOIN операціями для отримання зв'язаних даних, кешування часто використовуваних даних у пам'яті сервера для підвищення продуктивності. Система також реалізує механізми пагінації для великих наборів даних та сортування результатів за різними критеріями.

Діаграма детально показує обробку різних типів помилок, які можуть виникнути на кожному етапі виконання запиту.

Наприклад: код помилки 400 (Bad Request) повертається при некоректних параметрах запиту, в ситуації, відсутності обов'язкових полів, невалідному форматі email адреси, або при спробі створити рейтинг з неприпустимим значенням. Цей код також використовується при порушенні бізнес-правил системи, таких як спроба створити дублікат запису або використання заборонених символів у назвах.

Код помилки 401 (Unauthorized) генерується при проблемах з аутентифікацією, включаючи відсутність токену авторизації, використання недійсного або простроченого токену, або при спробі доступу до ресурсів без необхідних прав. Це включає ситуації, коли користувач намагається редагувати чужий профіль, видаляти коментарі інших користувачів, або отримати доступ до адміністративних функцій без відповідних повноважень.

Код помилки 404 (Not Found) повертається, коли запитуваний ресурс не існує в системі. Це може бути спроба отримати інформацію про неіснуючий фільм, переглянути профіль видаленого користувача, або звернутися до обговорення, яке було модераторами приховано або видалено. Система завжди перевіряє існування ресурсу перед виконанням будь-яких операцій з ним.

Код помилки 2xx (Success) використовується для позначення успішного виконання операцій, при цьому система повертає структуровані JSON відповіді з усією необхідною інформацією та метаданими для клієнтської частини.

Для детального розуміння взаємодії між компонентами системи була розроблена діаграма послідовностей (див. рис. 3.1.2), яка демонструє обмін повідомленнями між різними рівнями архітектури застосунку в часовій послідовності.

Діаграма послідовностей детально показує взаємодію між користувацьким інтерфейсом, Express.js сервером, системою аутентифікації Supabase Auth та базою даних Supabase DB. Кожна взаємодія включає передачу специфічних даних та обробку різних сценаріїв виконання.

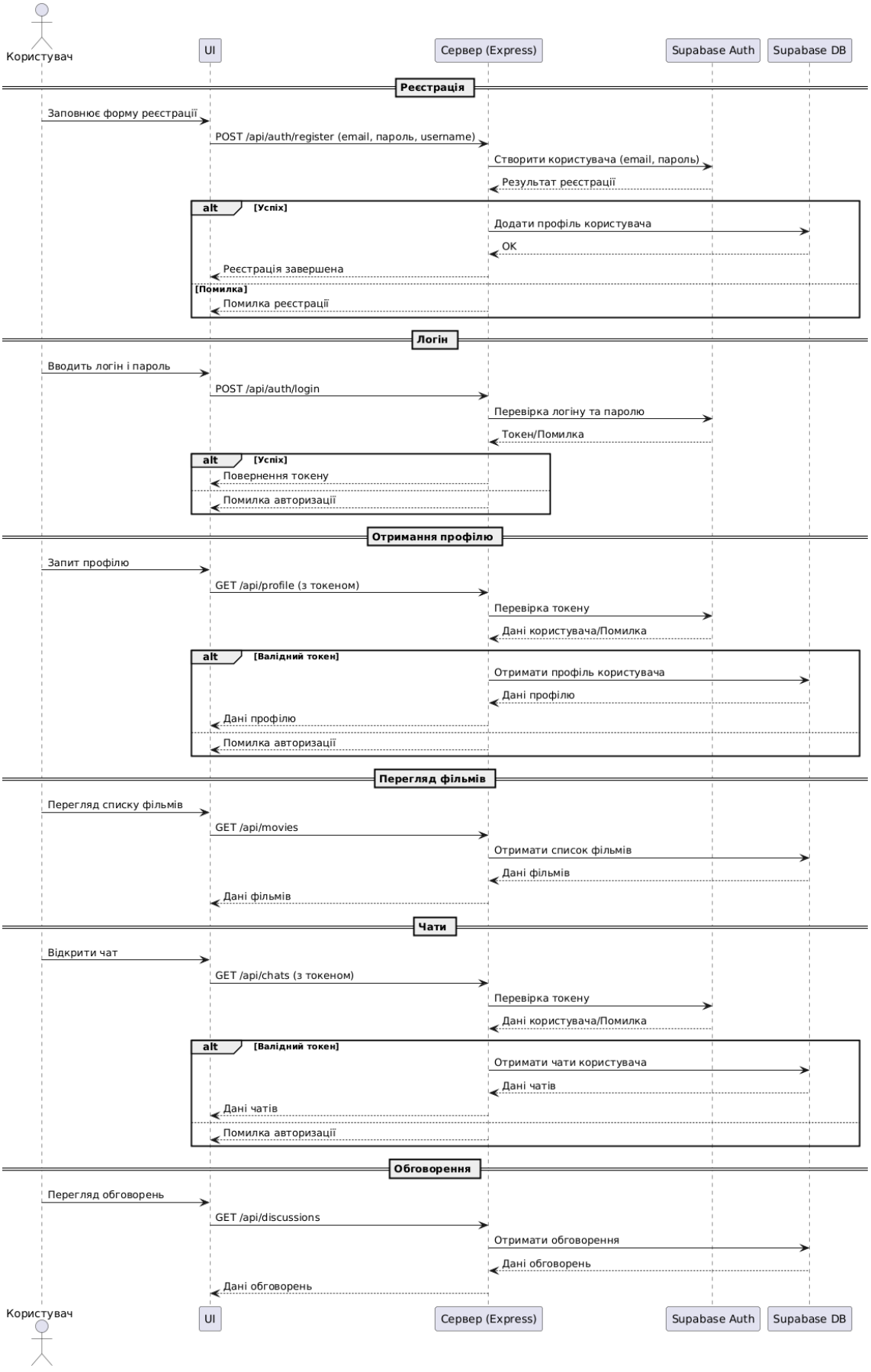


Рисунок 3.1.2 – Діаграма послідовностей веб-застосунку (рисунок виконаний самостійно)

Процес реєстрації користувача розпочинається з відправки POST запиту на ендпоінт /api/auth/register з даними користувача, включаючи email, пароль та додаткову інформацію профілю. Сервер виконує комплексну валідацію отриманих даних, перевіряючи унікальність email адреси, відповідність пароля політиці безпеки, та валідність інших полів. Після успішної валідації сервер звертається до Supabase Auth для створення нового користувача, а потім створює відповідний запис у таблиці профілів з додатковою інформацією.

При успішному завершенні процесу клієнт отримує JWT токен та базову інформацію про створений профіль.

Процес аутентифікації включає POST запит на /api/auth/login з обліковими даними користувача. Сервер передає ці дані до Supabase Auth для верифікації, після чого генерує JWT токен з відповідними правами доступу та інформацією про користувача. Токен має обмежений термін дії та може бути автоматично оновлений через refresh токен без втрати сесії користувача.

Управління профілем користувача реалізовано через серію ендпоінтів, які дозволяють отримувати, оновлювати та видаляти інформацію профілю. GET запит на /api/profile включає перевірку валідності токену через middleware, витягування ідентифікатора користувача з токену, запит до бази даних для отримання повної інформації профілю, включаючи статистику активності та налаштування приватності.

Робота з каталогом фільмів включає GET запити на /api/movies з підтримкою широкого спектру параметрів фільтрації та сортування. Сервер обробляє параметри пошуку, жанру, року випуску, рейтингу та інших критеріїв, формує оптимізований SQL запит до бази даних, та повертає paginated результати з метаданими для навігації. Система також підтримує отримання детальної інформації про конкретний фільм, включаючи усі пов'язані рецензії та рейтинги.
Функціональність обговорень реалізована через ендпоінти /api/discussions, які дозволяють створювати нові теми, додавати коментарі, та взаємодіяти з існуючим контентом. Система підтримує ієрархічну структуру коментарів з можливістю

відповідей на конкретні повідомлення, систему лайків та дизлайків, та механізми модерації контенту.

WebSocket з'єднання для чату в реальному часі встановлюється через окремий протокол з аутентифікацією через JWT токени. Сервер підтримує множинні одночасні з'єднання, ефективну систему broadcasting повідомлень між користувачами, автоматичне збереження історії повідомлень у базі даних, та систему сповіщень про нові повідомлення для офлайн користувачів.

На основі розроблених UML діаграм було реалізовано модульну архітектуру Express.js з чітким розділенням відповідальностей між різними компонентами системи. Маршрути організовані за функціональними областями з використанням Express Router для групування пов'язаних ендпоінтів. Middleware функції забезпечують кросс-функціональні можливості, такі як аутентифікація, логування запитів, обробка CORS політик, та централізована обробка помилок.

Інтеграція з Supabase забезпечує єдину точку доступу до всіх даних з автоматичним застосуванням Row Level Security політик, які захищають дані користувачів на рівні бази даних. Система використовує Supabase PostgREST API для оптимізованих запитів з підтримкою складних фільтрів та joins між таблицями.

Безпека API реалізована через багаторівневий підхід, включаючи JWT аутентифікацію з ротацією токенів, CORS конфігурацію для контролю доступу з різних доменів, комплексну валідацію та санітизацію всіх вхідних даних, rate limiting для запобігання зловживанням API, та систему логування всіх критичних операцій для аудиту безпеки.

3.2 Проєктування архітектури ПЗ

Архітектура веб-застосунку “Absolute Cinema” побудована на основі багаторівневої клієнт-серверної моделі з чітким розподілом функціональних обов'язків між компонентами системи. Вибір такого підходу обумовлений необхідністю забезпечення високої продуктивності, можливості масштабування та ефективної організації процесів розробки і супроводу.

Фронтенд-рівень застосунку реалізований за допомогою нативних веб-технологій: HTML5 для структурування контенту, CSS3 для стилізації інтерфейсу та JavaScript для реалізації інтерактивної функціональності. Особливістю клієнтської частини є інтеграція WebSocket-з'єднань для забезпечення двостороннього обміну даними в режимі реального часу, що критично важливо для функціонування системи чатів.

Серверна архітектура базується на платформі Node.js, що дозволяє використовувати JavaScript як на клієнті, так і на сервері, забезпечуючи уніфікацію технологічного стеку. Основним фреймворком для побудови серверної логіки обрано Express.js, який надає потужний набір інструментів для створення RESTful API та обробки HTTP-запитів. Для управління даними використовується Supabase - хмарна платформа, що поєднує функціональність PostgreSQL бази даних з вбудованими сервісами автентифікації та авторизації.

Центральним елементом бекенд-архітектури є система маршрутизації API, організована за принципом розподілу endpoints за функціональними доменами. Структура API включає п'ять основних груп маршрутів: /api/auth для управління автентифікацією користувачів, /api/profile для операцій з користувацькими профілями, /api/movies для роботи з каталогом фільмів, /api/discussions для управління тематичними обговореннями та /api/chat для обслуговування системи повідомлень.

Система автентифікації реалізована з використанням JSON Web Tokens (JWT), що забезпечує stateless підхід до управління сесіями користувачів. Токени зберігаються в HTTP-only cookies для підвищення безпеки та автоматично оновлюються при наближенні терміну дії. Додатково реалізовано механізм refresh tokens для забезпечення безперервної роботи користувачів без необхідності повторної автентифікації.

UML-діаграма компонентів (рис. 3.2.1) ілюструє архітектурну структуру системи та взаємозв'язки між основними модулями. Користувацький інтерфейс виступає координуючим центром, який взаємодіє з чотирма спеціалізованими модулями через відповідні API endpoints. Модуль автентифікації забезпечує

безпечний доступ до системи через Auth API та WebSocket для підтримки активних сесій. Модуль управління профілем взаємодіє з Profiles API для виконання операцій створення, читання, оновлення та видалення користувацьких даних.

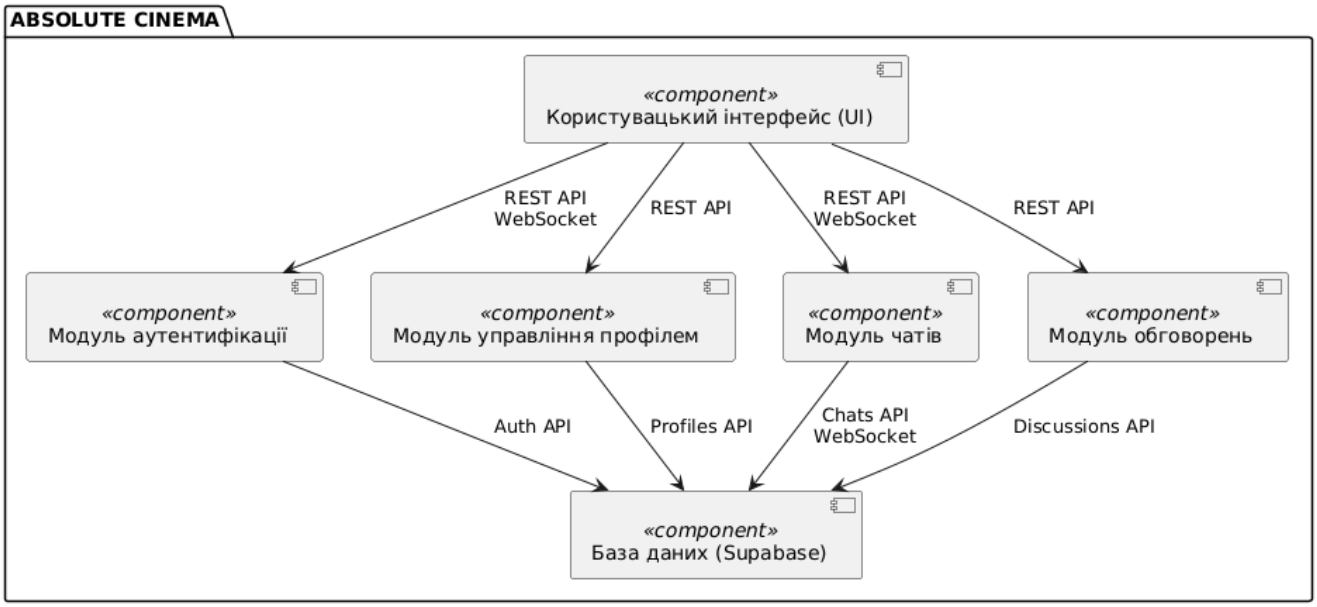


Рисунок 3.2.1 – Діаграма компонентів (рисунок виконаний самостійно)

Модуль чатів використовує комбінацію Chats API для збереження історії повідомлень та WebSocket з'єднання для миттєвої доставки нових повідомлень усім учасникам розмови. Модуль обговорень працює через Discussions API та забезпечує створення тематичних дискусій, додавання коментарів та управління модерацією контенту. Всі модулі централізовано взаємодіють з базою даних Supabase через відповідні API інтерфейси.

UML-діаграма пакетів (рис. 3.2.2) відображає логічну організацію системи та ієрархічну структуру програмних компонентів. Діаграма демонструє, як різні модулі згруповані в пакети та які залежності існують між ними.

На верхньому рівні розташований пакет “Бекенд”, який містить всю серверну логіку застосунку. Точка входу представлена сервером, який оброблює всі вхідні запити та розподіляє їх між відповідними підсистемами.

Конфігураційний пакет містить файл config.env з налаштуваннями середовища та системними параметрами.

Пакет “API” включає два основні компоненти: Express.js сервер для обробки HTTP-запитів та WebSocket сервер для забезпечення real-time комунікації. Цей пакет служить інтерфейсом між клієнтською частиною та внутрішньою бізнес-логікою системи.

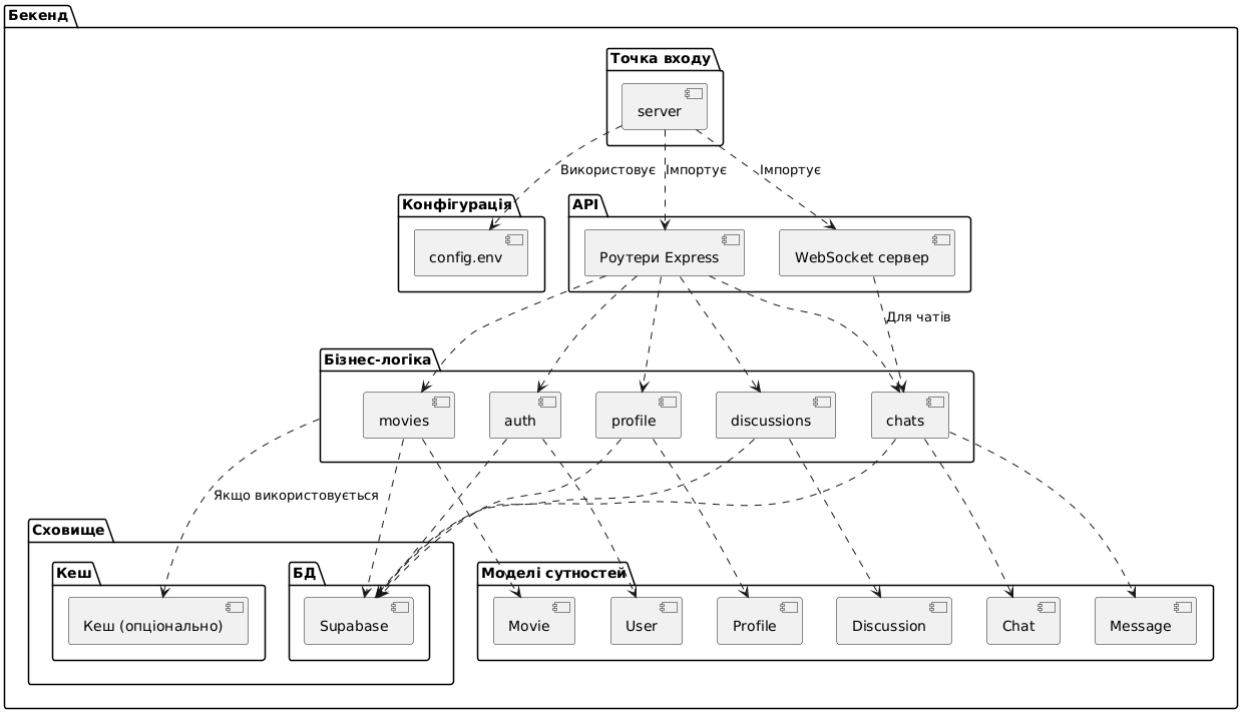


Рисунок 3.2.2 – Діаграма послідовностей веб-застосунку (рисунок виконаний самостійно)

Центральним елементом архітектури є пакет “Бізнес-логіка”, який містить п'ять спеціалізованих модулів. Модуль movies інкапсулює логіку роботи з каталогом фільмів, модуль auth відповідає за процеси автентифікації та авторизації, модуль profile управляє користувацькими профілями, модуль discussions реалізує систему тематичних обговорень, а модуль chats забезпечує функціональність обміну повідомленнями.

Пакет “Сховище” представлений двома компонентами: опціональною системою кешування та основною базою даних Supabase. Система кешування

оптимізує продуктивність за рахунок зберігання часто запитуваних даних у пам'яті, а Supabase забезпечує надійне збереження всієї інформації системи.

Нижній рівень займає пакет “Моделі сутностей”, який містить шість основних сутностей предметної області: Movie, User, Profile, Discussion, Chat та Message. Ці моделі визначають структуру даних та бізнес-правила системи, забезпечуючи консистентність інформації на всіх рівнях архітектури.

Архітектурні рішення включають використання декількох важливих патернів проектування. Repository Pattern застосовано для абстракції операцій з базою даних, що дозволяє легко змінювати способи збереження даних без впливу на бізнес-логіку. Middleware Pattern використовується в Express.js для обробки cross-cutting concerns, таких як логування запитів, валідація даних та перевірка автентифікації. Event-Driven Pattern реалізовано через WebSocket для асинхронної обробки подій в системі чатів та сповіщень.

Безпека системи забезпечується через багаторівневий підхід, що включає валідацію та санітизацію всіх вхідних даних, використання CORS middleware для контролю cross-origin запитів, реалізацію rate limiting для запобігання DoS атакам та шифрування чутливих даних перед збереженням у базі даних.

Масштабованість архітектури забезпечується через stateless дизайн API, можливість горизонтального масштабування серверних інстансів, використання зовнішньої бази даних Supabase з вбудованими механізмами реплікації та розподіл навантаження між HTTP та WebSocket серверами.

Обрана архітектура створює надійну основу для розвитку платформи “Absolute Cinema”, забезпечуючи високу продуктивність, безпеку та можливості для майбутнього розширення функціональності без кардинальних змін у структурі системи.

3.3 Приклади використаних алгоритмів та методів

3.3.1 Алгоритм аутентифікації користувача з автоматичним оновленням токену

Одним з ключових алгоритмів бекенду веб-застосунку Absolute Cinema є система аутентифікації з автоматичним оновленням токенів доступу. Цей алгоритм забезпечує безперервний та безпечний доступ користувачів до ресурсів системи без необхідності повторної авторизації при закінченні терміну дії токену доступу.

Система аутентифікації працює на основі двох типів токенів: access_token (короткотерміновий токен доступу) та refresh_token (довготерміновий токен оновлення). Коли користувач надсилає запит до API, сервер автоматично перевіряє валідність access_token з підписаних HTTP cookies. У випадку, якщо основний токен невалідний, але присутній refresh_token, система намагається автоматично оновити сесію через Supabase та оновити куки без втручання користувача.

Діаграма активностей алгоритму представлена на рисунку 3.3.1.1, де показано повний цикл роботи системи аутентифікації, включаючи всі етапи перевірки токенів, прийняття рішень та обробки різних сценаріїв.

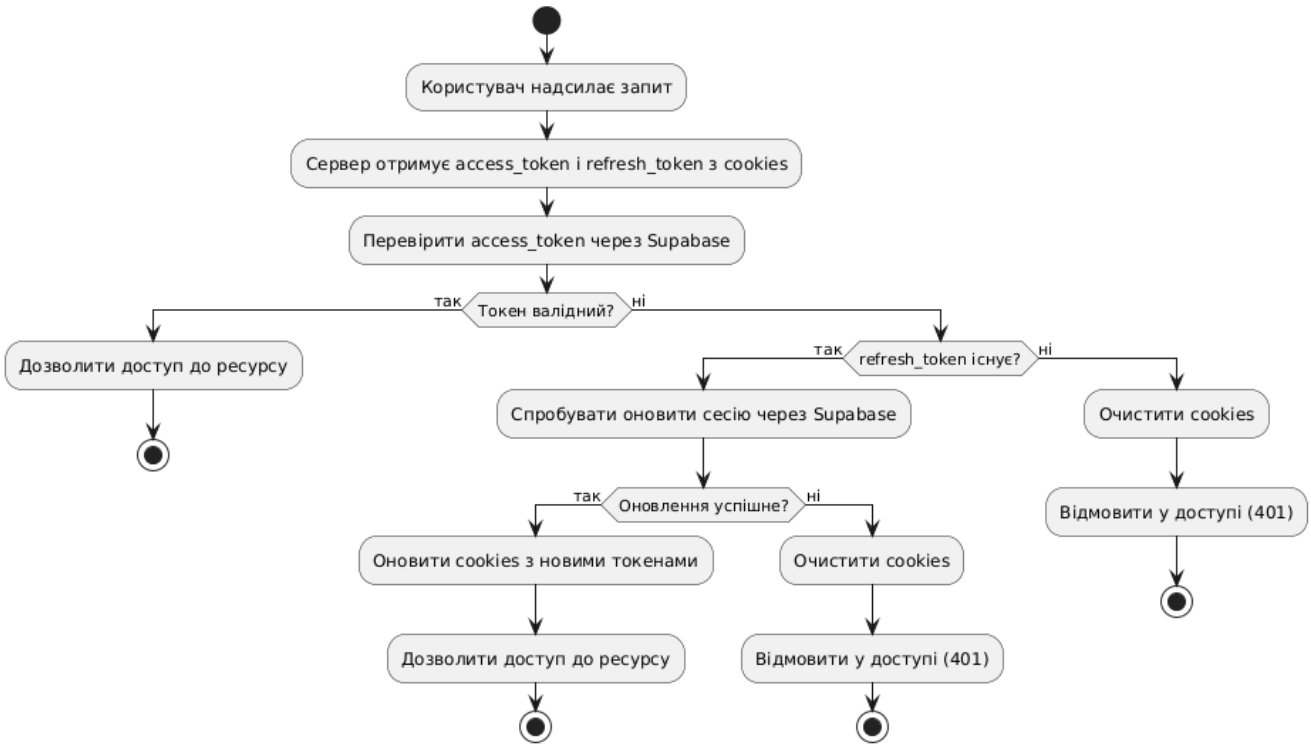


Рисунок 3.3.1.1 – Діаграма активностей алгоритму (рисунок виконаний самостійно)

У Node.js/Express.js застосунку алгоритм реалізовано у вигляді middleware функції, яка автоматично застосовується до всіх захищених маршрутів API. При отриманні запиту middleware витягує токени з cookies та передає їх до Supabase клієнта для валідації. Якщо access_token прострочений, система викликає метод refreshSession() з наявним refresh_token. У разі успішного оновлення, нові токени записуються в HTTP cookies з встановленими secure та httpOnly флагами для забезпечення безпеки.

Цей алгоритм критично важливий для веб-застосунку Absolute Cinema, оскільки користувачі можуть проводити тривалий час на платформі, переглядаючи фільми, залишаючи відгуки та спілкуючись в чатах. Автоматичне оновлення токенів забезпечує безшовний користувацький досвід, підвищує безпеку системи через використання коротких термінів життя access_token та зменшує навантаження на сервер завдяки відсутності необхідності повторної аутентифікації. Система також автоматично очищає невалідні токени, підтримуючи чистоту cookies та запобігаючи накопиченню застарілих даних сесії.

3.3.2 Алгоритм створення чату між двома користувачами з урахуванням унікальності

Важливою частиною бекенду веб-застосунку Absolute Cinema є система управління приватними чатами між користувачами. Алгоритм створення чату забезпечує унікальність кожної розмови між парою користувачів та оптимізує процес пошуку існуючих діалогів.

Основною особливістю алгоритму є нормалізація ідентифікаторів користувачів для забезпечення консистентності даних у базі. Коли користувач ініціює створення чату з іншим учасником, система автоматично впорядковує їхні ID за зростанням, створюючи стандартизований ключ для пошуку. Це дозволяє уникнути дублювання чатів та забезпечує швидкий доступ до існуючих розмов незалежно від того, хто був ініціатором діалогу.

Діаграма активностей алгоритму представлена на рисунку 3.3.2.1, де показано повний процес створення чату від ініціації користувача до повернення результату.

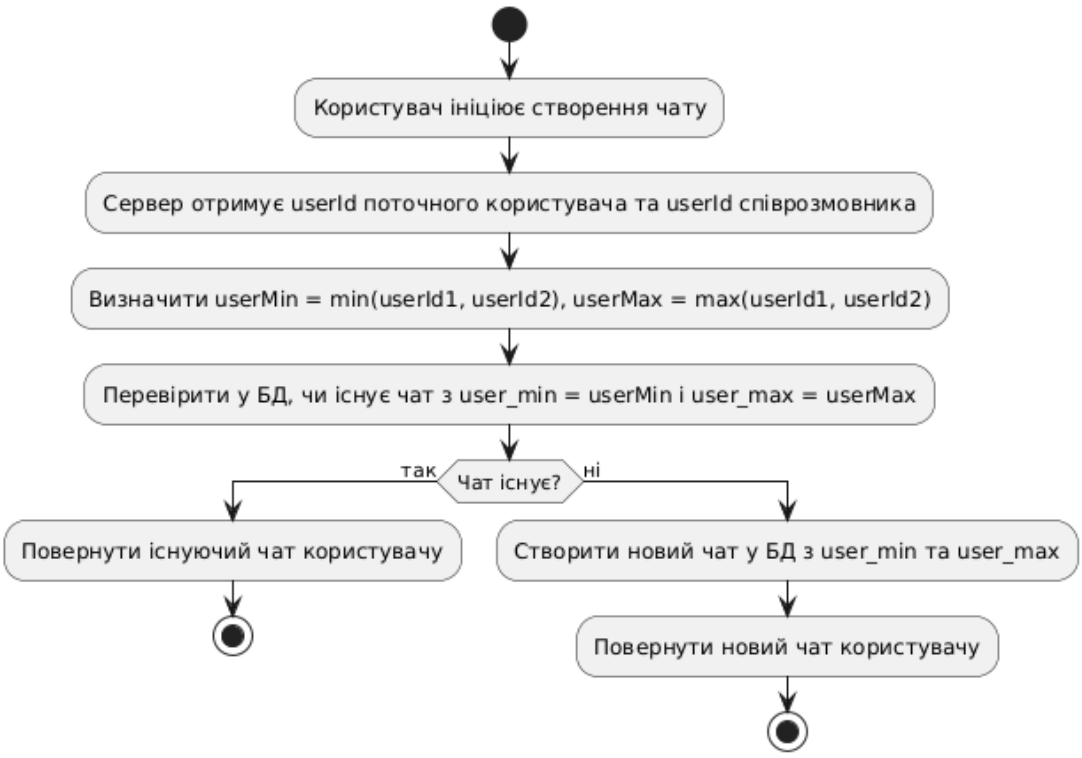


Рисунок 3.3.2.1 - Діаграма активностей алгоритму (рисунок виконаний самостійно)

Технічна реалізація алгоритму в Node.js/Express.js використовує SQL-запити до бази даних Supabase для перевірки існування чату та його створення. При отриманні запиту на створення чату, сервер витягує userId поточного користувача з JWT токену та userId співрозмовника з параметрів запиту. Далі система обчислює userMin та userMax за допомогою функцій Math.min() та Math.max(), що забезпечує детермінований порядок незалежно від послідовності передачі параметрів.

Запит до бази даних виконується з умовою WHERE user_min = userMin AND user_max = userMax, що дозволяє швидко знайти існуючий чат. У разі відсутності запису, система створює новий чат з нормалізованими значеннями user_min та user_max, після чого повертає його клієнту. Такий підхід забезпечує O(log n) складність пошуку при наявності індексу на полях user_min та user_max.

Алгоритм особливо важливий для платформи Absolute Cinema, де користувачі активно обговорюють фільми та обмінюються думками. Унікальність чатів запобігає плутанині та дублюванню повідомлень, а стандартизований підхід до зберігання спрощує майбутнє масштабування системи. Крім того, нормалізація ідентифікаторів дозволяє ефективно реалізувати функції пошуку чатів користувача та статистики активності без необхідності складних JOIN операцій у базі даних.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Архітектурні принципи та комунікаційні протоколи

Бекенд веб-застосунку Absolute Cinema побудований на основі REST архітектури з використанням Node.js та Express.js фреймворку. Система забезпечує взаємодію між клієнтською та серверною частинами через HTTP протокол із дотриманням принципів stateless комунікації, де кожен запит є незалежним і містить всю необхідну інформацію для обробки. REST архітектура гарантує уніфікованість інтерфейсу між компонентами, що дозволяє передавати інформацію у стандартній формі та забезпечує можливість ідентифікації ресурсів незалежно від їх представлення.

Серверна частина реалізує систему ендпоінтів, організованих за функціональними групами. Група аутентифікації (/api/auth/) включає ендпоінти для перевірки стану авторизації (GET /api/auth/status), автентифікації користувача (POST /api/auth/login), реєстрації нового користувача (POST /api/auth/register), завершення сесії (POST /api/auth/logout) та відновлення пароля (POST /api/auth/reset-password). Управління профілем реалізовано через ендпоінти /api/profile для отримання та оновлення даних профілю, а також завантаження аватара. Робота з рецензіями забезпечується через групу /api/reviews з можливостями створення, отримання, редагування та видалення рецензій.

Реалізація REST принципів демонструється на прикладі ендпоінту реєстрації користувача, який ілюструє повний життєвий цикл обробки HTTP запиту від отримання до відповіді. Ендпоінт використовує POST метод для створення нового ресурсу (користувача) та дотримується принципу idempotency через перевірку унікальності email:

```
// Приклад реєстрації користувача
app.post('/api/auth/register', async (req, res) => {
    try {
        const { username, email, password } = req.body;

        // Валідація вхідних даних
        if (!username || !email || !password) {
            return res.status(400).json({ error: 'Всі поля обов\'язкові для заповнення' });
```

```
        }

        // Реєстрація через Supabase Auth
        const { data, error } = await supabase.auth.signUp({
            email,
            password,
            options: {
                data: { username }
            }
        });

        if (error) {
            return res.status(400).json({ error: error.message });
        }

        return res.json({ success: true, user: data.user });
    } catch (err) {
        return res.status(500).json({ error: 'Помилка сервера' });
    }
});
```

Цей код демонструє багатошарову архітектуру обробки запитів, що включає шар валідації даних з перевіркою наявності обов'язкових полів username, email та password через деструктуризацію req.body. Бізнес-логіка реєстрації делегована Supabase Auth сервісу, що забезпечує централізоване управління аутентифікацією та шифрування паролів за стандартами безпеки. Обробка помилок реалізована через каскадну систему try-catch блоків з поверненням специфічних HTTP статус-кодів: 400 для помилок валідації клієнтських даних, 400 для помилок Supabase Auth та 500 для непередбачених серверних помилок. Успішна реєстрація повертає об'єкт з булевим полем success та даними створеного користувача, що відповідає принципам RESTful дизайну.

Система авторизації реалізує складну багаторівневу архітектуру з використанням JWT токенів, що зберігаються у підписаних HTTP-only cookies для максимального захисту від XSS та CSRF атак. Механізм автоматичного оновлення токенів забезпечує безперервність користувацької сесії без втрати стану додатку. Middleware authMiddleware функціонує як перехоплювач запитів, що виконується перед основною логікою обробки:

```
// Middleware для перевірки авторизації
const authMiddleware = async (req, res, next) => {
    const accessToken = req.signedCookies.access_token;
```

```
        const refreshToken = req.signedCookies.refresh_token;

        if (!accessToken) {
            return next();
        }
        try {
            // Перевірка валідності сесії
            const { data: { user }, error } = await
supabase.auth.getUser(accessToken);

            if (error && refreshToken) {
                // Спроба оновити токен
                const { data, error: refreshError } = await
supabase.auth.refreshSession({
                    refresh_token: refreshToken
                });

                if (!refreshError && data.session) {
                    setAuthCookies(res, data.session);
                    req.user = data.user;
                }
            } else {
                req.user = user;
            }
        } catch (err) {
            clearAuthCookies(res);
        }

        next();
    };
```

Middleware реалізує інтелектуальну систему управління токенами з багатоступеневою логікою валідації та відновлення сесій. Спочатку відбувається extraction токенів з підписаних cookies через req.signedCookies, що гарантує їх цілісність та автентичність. Основна логіка включає первинну перевірку наявності access_token, при його відсутності middleware пропускає запит далі без встановлення req.user. Валідація токену здійснюється через Supabase Auth API методом getUser(), який повертає користувача або помилку. Критичною особливістю є система graceful fallback - при невалідності access_token middleware автоматично намагається відновити сесію через refresh_token за допомогою методу refreshSession(). У разі успішного оновлення нові токени встановлюються через функцію setAuthCookies(), а об'єкт req.user заповнюється даними користувача для подальшого використання в ендпоінтах. Обробка винятків включає очищення некоректних cookies через clearAuthCookies() для запобігання

накопиченню невалідних даних. Виклик next() забезпечує передачу управління наступному middleware або основному обробнику запиту.

Архітектура взаємодії з базою даних побудована на принципах безпечного доступу через Supabase клієнт, що ініціалізується з використанням змінних середовища для ізоляції конфігурації від коду. Ініціалізація через process.env забезпечує гнучкість розгортання в різних середовищах без зміни коду:

```
// Ініціалізація Supabase клієнта
const supabase = createClient(
    process.env.SUPABASE_URL,
    process.env.SUPABASE_ANON_KEY
);

// Приклад роботи з профілем користувача
app.get('/api/profile', async (req, res) => {
    try {
        if (!req.user) {
            return res.status(401).json({ error: 'Користувач не авторизований' });
        }

        const { data, error } = await supabase
            .from('profiles')
            .select('*')
            .eq('user_id', req.user.id)
            .single();

        if (error) {
            return res.status(500).json({ error: 'Помилка при отриманні профілю' });
        }

        return res.json({ profile: data });
    } catch (err) {
        return res.status(500).json({ error: 'Помилка сервера' });
    }
});
```

Ендпоінт отримання профілю демонструє комплексний підхід до безпечного доступу до персональних даних користувача з багаторівневою системою захисту. Архітектура включає авторизаційний шар з перевіркою наявності req.user (встановлюється authMiddleware), шар доступу до даних через Supabase ORM з використанням методу from() для вибору таблиці та select() для специфікації полів. Критичною особливістю є застосування Row Level Security через фільтрацію eq('user_id', req.user.id), що гарантує доступ користувача тільки

до власних даних навіть при компрометації SQL запитів. Метод single() оптимізує запит для отримання одного запису та автоматично валідує унікальність результату. Система обробки помилок розрізняє помилки авторизації (401), помилки бази даних (500) та загальні серверні помилки з відповідними HTTP статус-кодами та дескриптивними повідомленнями українською мовою для покращення користувацького досвіду.

Cookies налаштовані з параметрами безпеки httpOnly, secure (у продакшені), signed та sameSite: 'lax' для захисту від CSRF атак. CORS налаштування обмежують доступ до API тільки з дозволених доменів. Для реалізації чату в реальному часі використовується WebSocket протокол з автентифікацією з'єднань через JWT токени, групуванням користувачів у кімнати та обробкою подій підключення, відключення та надсилання повідомлень. Така архітектурна організація забезпечує масштабованість, безпеку та високу продуктивність веб-застосунку.

4.2 Система обміну повідомленнями та безпека

Система обміну повідомленнями в реальному часі реалізована на основі WebSocket протоколу, що забезпечує двостороннє з'єднання між клієнтом та сервером з мінімальною затримкою. WebSocket архітектура дозволяє підтримувати постійне з'єднання без необхідності періодичного polling, що значно знижує навантаження на сервер та покращує користувацький досвід при обміні миттєвими повідомленнями.

Ініціалізація WebSocket сервера здійснюється через модульний підхід з винесенням логіки в окремий файл для забезпечення читабельності та масштабованості коду:

```
// Ініціалізація WebSocket сервера
const { initWebSocket } = require('./server/websocket');
const server = http.createServer(app);
initWebSocket(server);
```

Модульна архітектура дозволяє ізолювати WebSocket логіку від основного Express серверу та забезпечує можливість незалежного тестування та розширення функціональності чату. HTTP сервер створюється на основі Express додатку та передається до WebSocket ініціалізатора для створення гібридного сервера, що обслуговує як HTTP запити, так і WebSocket з'єднання на одному порту.

Обробка повідомлень реалізована через event-driven архітектуру з комплексною системою валідації, збереження та розповсюдження повідомлень між учасниками чату:

```
// Приклад обробки повідомлень
wss.on('connection', (ws) => {
    ws.on('message', async (message) => {
        try {
            const data = JSON.parse(message);

            // Збереження повідомлення в базі даних
            const { error } = await supabase
                .from('messages')
                .insert([{
                    chat_id: data.chatId,
                    user_id: data.userId,
                    content: data.content,
                    created_at: new Date()
                }]);

            if (error) throw error;

            // Відправка повідомлення всім учасникам чату
            broadcastToChat(data.chatId, {
                type: 'message',
                data: {
                    userId: data.userId,
                    content: data.content,
                    timestamp: new Date()
                }
            });
        } catch (err) {
            console.error('Помилка обробки повідомлення:', err);
        }
    });
});
```

Система обробки повідомлень включає декілька критичних етапів: парсинг JSON повідомлення з клієнта з обробкою можливих помилок синтаксису, персистентне збереження повідомлення в базі даних Supabase з заповненням

метаданих (chat_id, user_id, timestamp), та broadcast механізм для синхронного розповсюдження повідомлення всім активним учасникам чату через функцію broadcastToChat(). Асинхронна природа операцій забезпечує не блокуючу обробку повідомлень навіть при високому навантаженні.

Архітектура безпеки побудована на принципах defense-in-depth з багаторівневим захистом від основних векторів атак. CORS (Cross-Origin Resource Sharing) налаштування забезпечують контрольований доступ до API з різних доменів:

```
// Налаштування CORS
app.use(cors({
    origin: true,
    credentials: true,
    methods: ['GET', 'POST', 'PUT', 'DELETE', 'OPTIONS'],
    allowedHeaders: ['Content-Type', 'Authorization']
}));
```

CORS конфігурація дозволяє запити з будь-якого домену (origin: true) з підтримкою credentials для передачі cookies та авторизаційних заголовків. Список дозволених HTTP методів обмежений до необхідних для функціонування додатку, а allowedHeaders контролює, які заголовки можуть бути відправлені в preflight запитах. Такий підхід забезпечує гнучкість розробки при збереженні контролю над кросс-доменними запитами.

Система управління токенами реалізована через безпечне зберігання в HTTP cookies з розширеними параметрами захисту:

```
// Безпеке зберігання токенів
const setAuthCookies = (res, session) => {
    const maxAge = parseInt(process.env.COOKIE_MAX_AGE) || 604800000;

    res.cookie('access_token', session.access_token, {
        httpOnly: true,
        secure: process.env.NODE_ENV === 'production',
        signed: true,
        maxAge: maxAge,
        sameSite: 'lax'
    });

    if (session.refresh_token) {
        res.cookie('refresh_token', session.refresh_token, {
```

```
            httpOnly: true,
            secure: process.env.NODE_ENV === 'production',
            signed: true,
            maxAge: maxAge,
            sameSite: 'lax'
        });
    }
};
```

Функція setAuthCookies реалізує найкращі практики безпечного зберігання токенів через комплекс захисних механізмів. Параметр httpOnly забороняє доступ до cookies через JavaScript, що ефективно захищає від XSS атак. Опція secure активується в продакшені для забезпечення передачі cookies тільки через HTTPS з'єднання. Підпис cookies (signed: true) гарантує їх цілісність та захист від підробки через cryptographic signing. Параметр maxAge контролює час життя cookies через змінні середовища для гнучкого управління сесіями. Налаштування sameSite: 'lax' забезпечує компроміс між безпекою та функціональністю, дозволяючи передачу cookies при top-level navigation, але блокуючи при кросс-сайтових запитах, що захищає від CSRF атак.

Загальна архітектура безпеки включає валідацію всіх вхідних даних на серверному рівні, sanitization користувацького контенту для запобігання injection атакам, rate limiting для захисту від DDoS атак, та comprehensive error handling з логуванням подій безпеки. WebSocket з'єднання додатково захищені через авторизацію при підключенні та верифікацію прав доступу до чатів на основі user_id з JWT токенів.

5 АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

В результаті виконання проекту було успішно створено повнофункціональний веб-застосунок “Absolute Cinema” для рецензування фільмів та соціального спілкування кіноманів. Аналіз отриманих результатів показує високий рівень відповідності системи поставленим вимогам та цілям проекту.

Серверна частина застосунку, розроблена на базі Node.js та Express.js, повністю відповідає поставленим технічним вимогам. Всі заплановані API-ендпоінти було успішно реалізовано та протестовано. Система аутентифікації забезпечує надійну реєстрацію, вхід та управління сесіями користувачів з використанням JWT токенів, що гарантує безпеку даних та контроль доступу. Функціонал управління профілями дозволяє користувачам створювати, редагувати та налаштовувати персональну інформацію, включаючи завантаження аватарів та збереження історії активності.

Робота з каталогом фільмів реалізована через комплексний API, який забезпечує перегляд детальної інформації про фільми, систему пошуку з фільтрацією, а також алгоритм рекомендацій на основі переглянутих користувачем фільмів. Система обговорень та рецензій дозволяє створювати теми для дискусій, залишати коментарі з можливістю модерації контенту, що сприяє підтримці якісного спілкування між користувачами.

Особливо успішною виявилася реалізація чату в реальному часі з використанням WebSocket технології. Серверна частина забезпечує стабільний обмін повідомленнями, підтримку групових та приватних чатів, систему сповіщень про нові повідомлення та індикатори непрочитаних повідомлень. Інтеграція з Supabase для управління базою даних показала високу ефективність у забезпеченні швидкого доступу до даних та надійного зберігання інформації.

Система безпеки серверної частини успішно реалізує захист від CSRF-атак, безпечне зберігання паролів з хешуванням, валідацію всіх вхідних даних та захист API-ендпоінтів. CORS налаштування забезпечують контрольований доступ до

ресурсів серверу, а Cookie Parser та Body Parser коректно обробляють всі типи запитів від клієнтської частини.

Архітектура серверної частини побудована з урахуванням принципів масштабованості та модульності. Код розділений на логічні модулі, що полегшує подальше розширення функціоналу та підтримку системи. Розвинена система обробки помилок та логування забезпечує стабільність роботи застосунку та швидке виявлення потенційних проблем.

Тестування серверної частини показало стабільну роботу всіх компонентів системи під навантаженням. API демонструє швидкий час відгуку, ефективне використання ресурсів сервера та коректну обробку великої кількості одночасних запитів. Система аутентифікації витримує спроби несанкціонованого доступу, а база даних показує оптимальну продуктивність при роботі з великими обсягами даних.

Отримані результати підтверджують успішність реалізації всіх поставлених завдань щодо серверної частини проекту. Застосунок готовий до практичного використання та може забезпечити стабільну роботу для значної кількості користувачів. Для подальшого вдосконалення рекомендується впровадження додаткових алгоритмів машинного навчання для покращення системи рекомендацій, розширення можливостей модерації контенту та інтеграція з додатковими зовнішніми API для отримання більш детальної інформації про фільми.

ВИСНОВКИ

Результатом роботи є створення повнофункціонального веб-застосунку “Absolute Cinema” для рецензування фільмів та соціального спілкування кіноманів, що включає комплексну серверну частину та зручний користувацький інтерфейс. В процесі розробки було проаналізовано сучасні потреби користувачів у сфері обговорення кінематографічних творів та визначено оптимальний стек технологій для реалізації поставлених завдань.

Основними досягненнями проекту є успішна інтеграція Node.js та Express.js для створення масштабованої серверної архітектури, ефективне використання Supabase для управління базою даних та аутентифікацією, а також реалізація системи спілкування в реальному часі за допомогою WebSocket технології. Розроблений застосунок забезпечує надійну систему безпеки з використанням JWT токенів, захист від CSRF-атак та валідацію всіх вхідних даних.

Створена система API включає повний функціонал для роботи з профілями користувачів, каталогом фільмів, системою рецензій та рейтингів, інтерактивними обговореннями та чатом в реальному часі. Архітектура проекту побудована з урахуванням принципів модульності та масштабованості, що дозволяє легко розширювати функціонал та підтримувати систему в майбутньому.

Застосунок готовий до практичного використання та може забезпечити якісний сервіс для спільноти кіноманів, сприяючи розвитку культури обговорення кінематографічних творів у цифровому середовищі.

Перспективами подальшого розвитку проекту є впровадження алгоритмів машинного навчання для покращення системи рекомендацій, розширення можливостей модерації контенту та інтеграція з додатковими зовнішніми API для отримання більш повної інформації про фільми та кіноіндустрію.

Посилання на GitHub репозиторій:
https://github.com/NureLopatenkoVolodymyr/2025_B_PI_PZPI-22-8_Lopatenko_V_V/tree/main

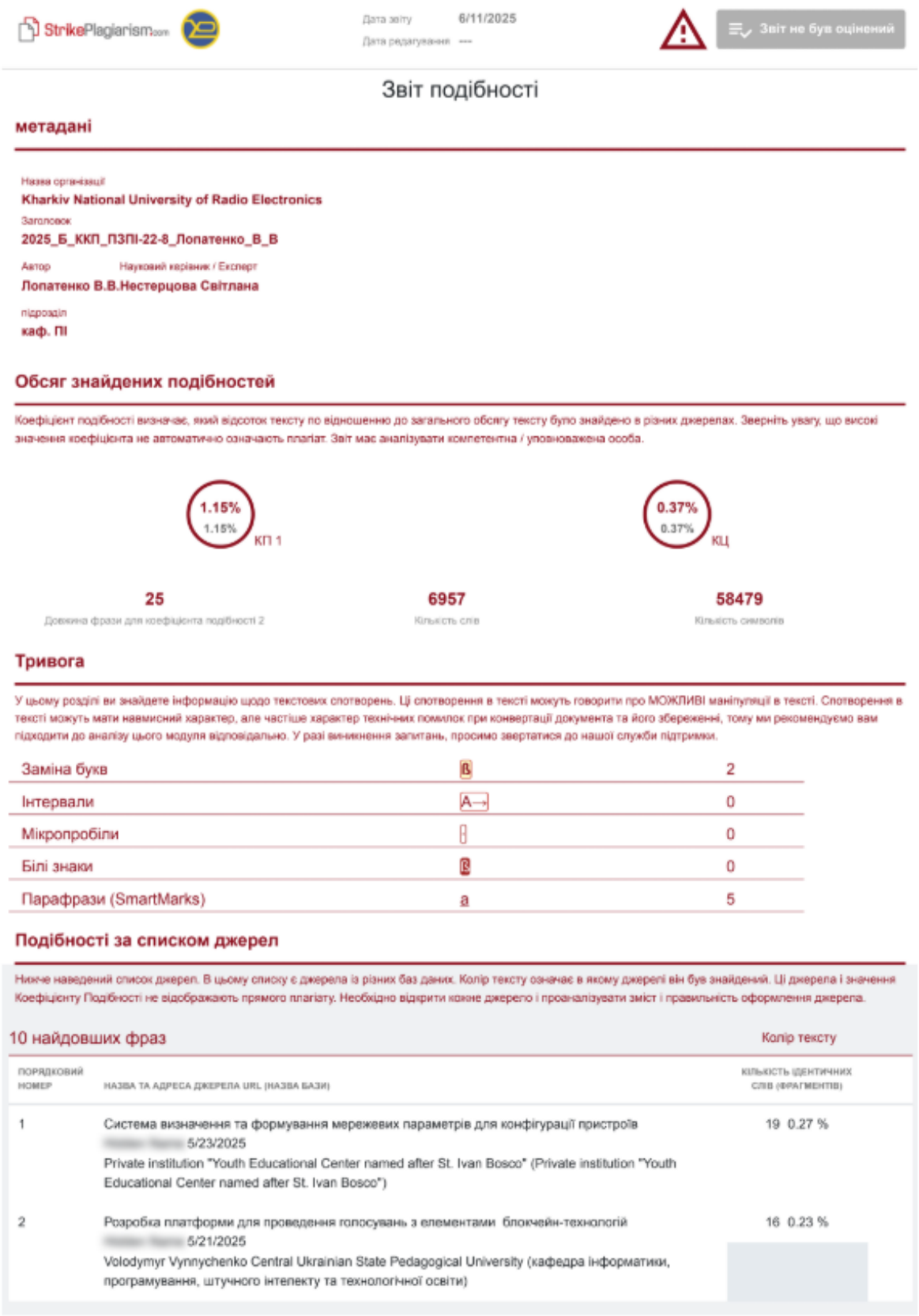
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. IMDb: Internet Movie Database [Електронний ресурс]. – URL: https://www.imdb.com/
2. Letterboxd [Електронний ресурс]. – URL: https://letterboxd.com/
3. Rotten Tomatoes [Електронний ресурс]. – URL: https://www.rottentomatoes.com/
4. Metacritic [Електронний ресурс]. – URL: https://www.metacritic.com/
5. Trakt [Електронний ресурс]. – URL: https://trakt.tv/
6. Cloudflare. Application Security report: 2024 update [Електронний ресурс]. – URL: https://blog.cloudflare.com/application-security-report-2024-update/
7. World Intellectual Property Organization (WIPO). Global Film Production Hits Historic High, Surpassing Pre-Pandemic Levels [Електронний ресурс]. – URL: https://www.wipo.int/en/web/global-innovation-index/w/blogs/2025/global-film-production
8. Дьяконов В. П. Основи JavaScript: навчальний посібник / В. П. Дьяконов. — Харків : ХНУРЕ, 2018. — 250 с.
9. Бойко В. В. HTML5 та CSS3: навч. посіб. / В. В. Бойко. — Харків : ХНУРЕ, 2020. — 195 с.
10. Ларман К. Застосування UML та шаблонів проектування / К. Ларман. — Київ: Діалектика, 2005. — 550 с.
11. Fowler M. UML. Основи. Короткий довідник / М. Фаулер ; пер. з англ. — Київ: Діалектика, 2007. — 208 с.
12. Мельникова Д. В. Веб-технології: методичні вказівки для лабораторних робіт / Д. В. Мельникова. — Харків : ХНУРЕ, 2022. — 48 с.
13. Парфьонов О. М. Основи веб-програмування: навч. посіб. / О. М. Парфьонов, І. Ю. Барабаш. — Харків : ХНУРЕ, 2021. — 256 с.
14. Чуприна А. В. Методичні рекомендації до виконання курсових проєктів з дисципліни «Програмна інженерія» / А. В. Чуприна. — Харків : ХНУРЕ, 2023. — 40 с.

15. Express.js Official Documentation [Електронний ресурс] – URL: https://expressjs.com/en/guide/ (дата звернення: 12.04.2025)
16. HTML5 Specification [Електронний ресурс] – URL: https://html.spec.whatwg.org/ (дата звернення: 08.04.2025)
17. JSON Web Tokens Introduction [Електронний ресурс] – URL: https://jwt.io/introduction/ (дата звернення: 18.04.2025)
18. Mozilla Developer Network. Express.js Framework [Електронний ресурс] – URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs (дата звернення: 15.04.2025)
19. Official Node.js Documentation [Електронний ресурс] – URL: https://nodejs.org/en/docs/ (дата звернення: 10.04.2025)
20. Supabase Documentation and Guides [Електронний ресурс] – URL: https://supabase.com/docs (дата звернення: 20.04.2025)
21. Visual Studio Code Documentation [Електронний ресурс] – URL: https://code.visualstudio.com/docs (дата звернення: 05.04.2025)
22. WebSocket API Reference [Електронний ресурс] – URL: https://developer.mozilla.org/en-US/docs/Web/API/WebSocket (дата звернення: 22.04.2025)
23. GitHub репозиторій з проєктом. — URL: https://github.com/NureLopatenkoVolodymyr/2025_B_PI_PZPI-22-8_Lopatenko_V_V/tree/main

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



StrikePlagiarism.com

Дата звіту 6/11/2025
Дата редагування ---

Звіт не був оцінений

Звіт подібності

метадані

Назва організації
Kharkiv National University of Radio Electronics
Заголовок
2025_Б_ККП_ПЗПІ-22-8_Лопатенко_В_В
Автор Науковий керівник / Експерт
Лопатенко В.В.Нестерцова Світлана
підрозділ
каф. ПІ

Обсяг знайдених подібностей

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

1.15% 1.15% КП 1

0.37% 0.37% КЦ

25	6957	58479
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

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

Заміна букв	ß	2
Інтервали	A→	0
Мікропробіли	-	0
Білі знаки	ß	0
Парафрази (SmartMarks)	a	5

Подібності за списком джерел

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

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Система визначення та формування мережевих параметрів для конфігурації пристроїв [redacted] 5/23/2025 Private institution "Youth Educational Center named after St. Ivan Bosco" (Private institution "Youth Educational Center named after St. Ivan Bosco")	19 0.27 %
2	Розробка платформи для проведення голосувань з елементами блокчейн-технологій [redacted] 5/21/2025 Volodymyr Vynnychenko Central Ukrainian State Pedagogical University (кафедра інформатики, програмування, штучного інтелекту та технологічної освіти)	16 0.23 %

3	https://medium.com/@adarsh-d/create-a-custom-token-in-node-js-f7a2035e2618	14 0.20 %
4	тези конф образи сучасності [redacted] 12/16/2024 State University of Trade and Economics (Кафедра журналістики та реклами)	11 0.16 %
5	https://ir.nmu.org.ua/bitstream/handle/123456789/162019/%D0%A2%D0%BA%D0%B0%D1%87%D0%B5%D0%BD%D0%BA%D0%BE.pdf?sequence=1	11 0.16 %
6	https://supabase.com/docs/reference/javascript/auth-admin-updateuserbyid	9 0.13 %

з бази даних RefBooks (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з домашньої бази даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з програми обміну базами даних (0.66 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Система визначення та формування мережевих параметрів для конфігурації пристроїв [redacted] 5/23/2025 Private institution "Youth Educational Center named after St. Ivan Bosco" (Private institution "Youth Educational Center named after St. Ivan Bosco")	19 (1) 0.27 %
2	Розробка платформи для проведення голосувань з елементами блокчейн-технологій [redacted] 5/21/2025 Volodymyr Vynnychenko Central Ukrainian State Pedagogical University (кафедра інформатики, програмування, штучного інтелекту та технологічної освіти)	16 (1) 0.23 %
3	тези конф образи сучасності [redacted] 12/16/2024 State University of Trade and Economics (Кафедра журналістики та реклами)	11 (1) 0.16 %

з Інтернету (0.49 %)

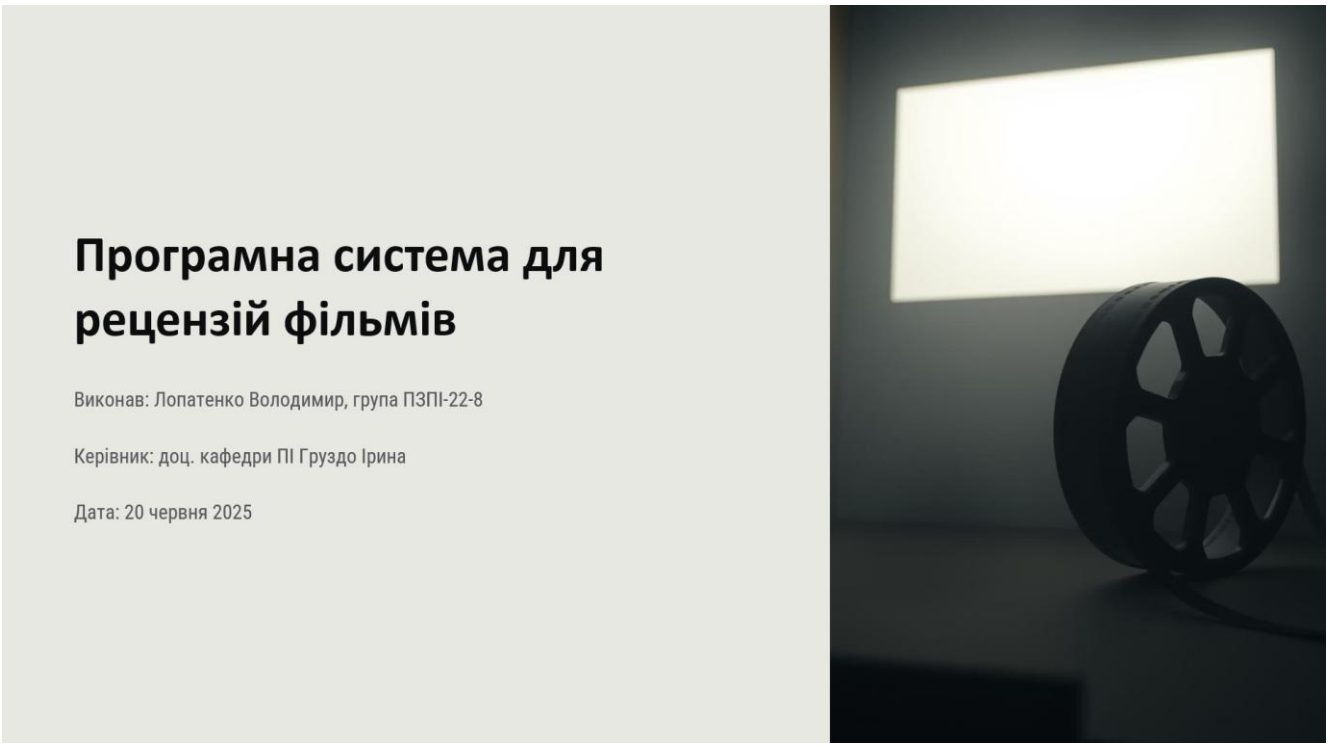
ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://medium.com/@adarsh-d/create-a-custom-token-in-node-js-f7a2035e2618	14 (1) 0.20 %
2	https://ir.nmu.org.ua/bitstream/handle/123456789/162019/%D0%A2%D0%BA%D0%B0%D1%87%D0%B5%D0%BD%D0%BA%D0%BE.pdf?sequence=1	11 (1) 0.16 %
3	https://supabase.com/docs/reference/javascript/auth-admin-updateuserbyid	9 (1) 0.13 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

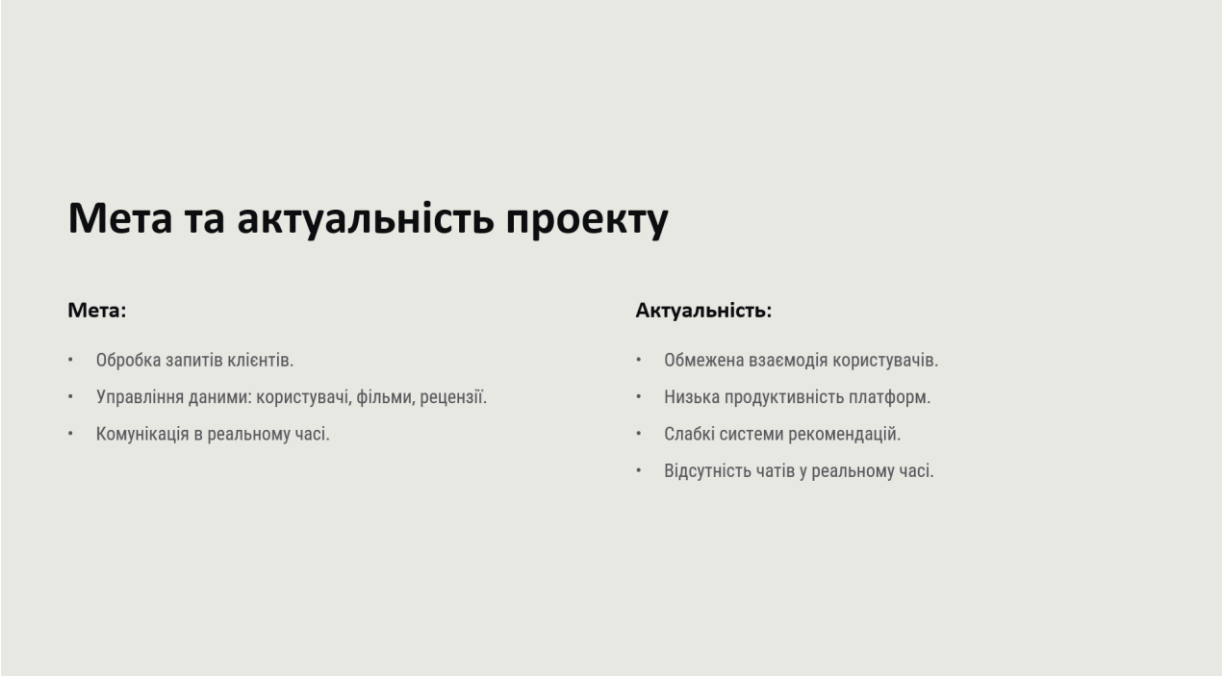
ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

ДОДАТОК Б

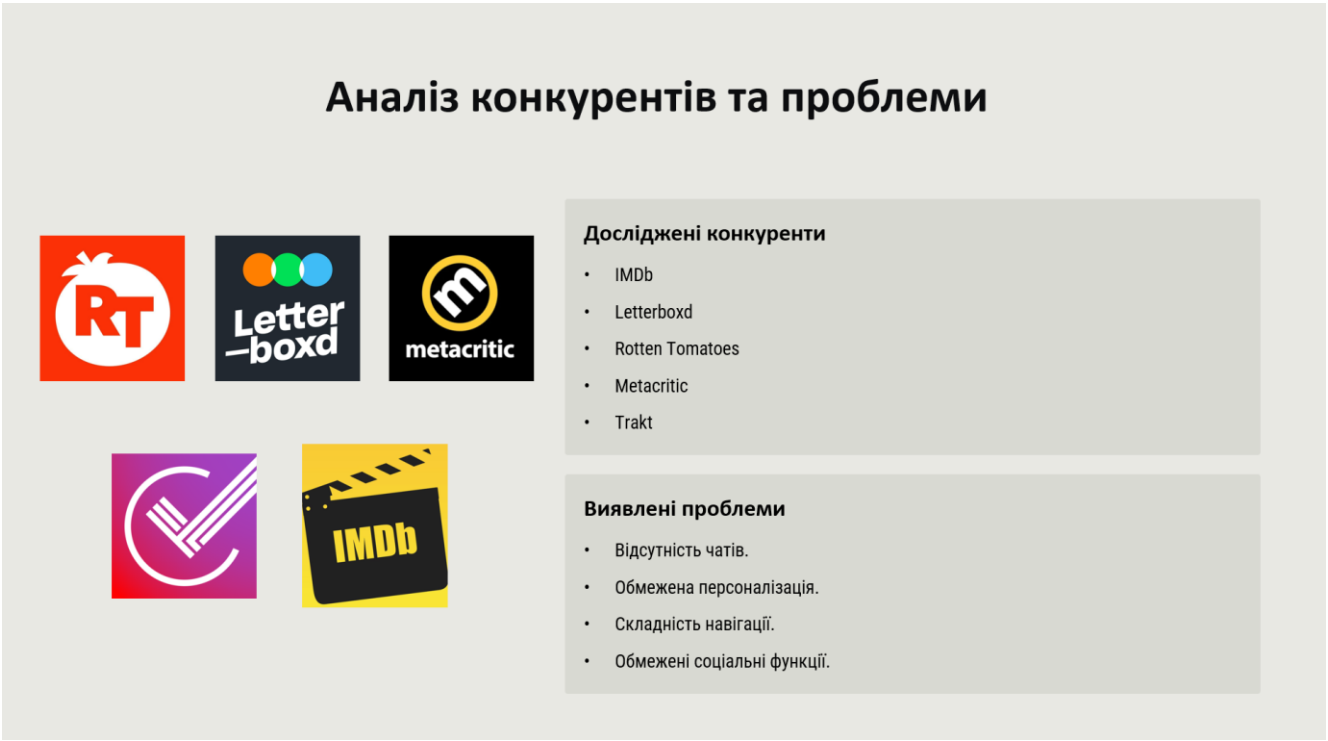
Слайди презентації



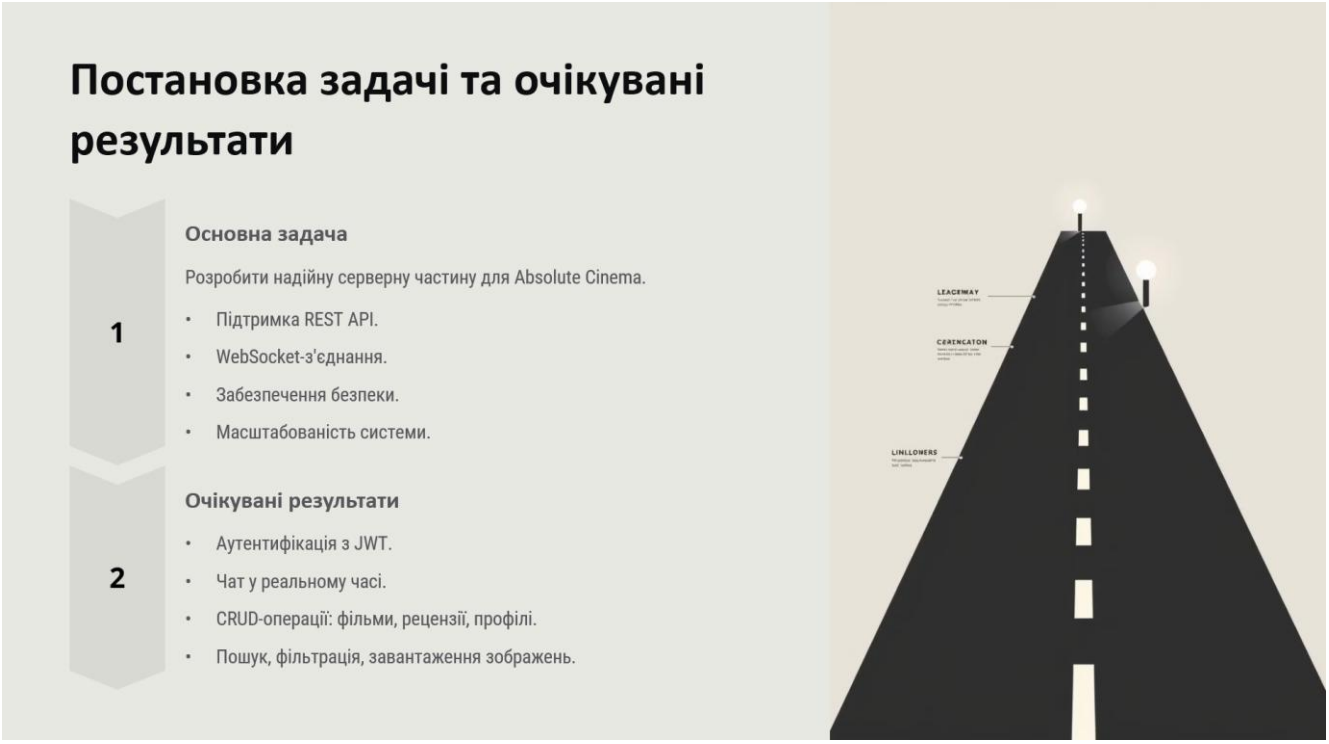
Слайд Б1 – Програмна система для рецензій фільмів



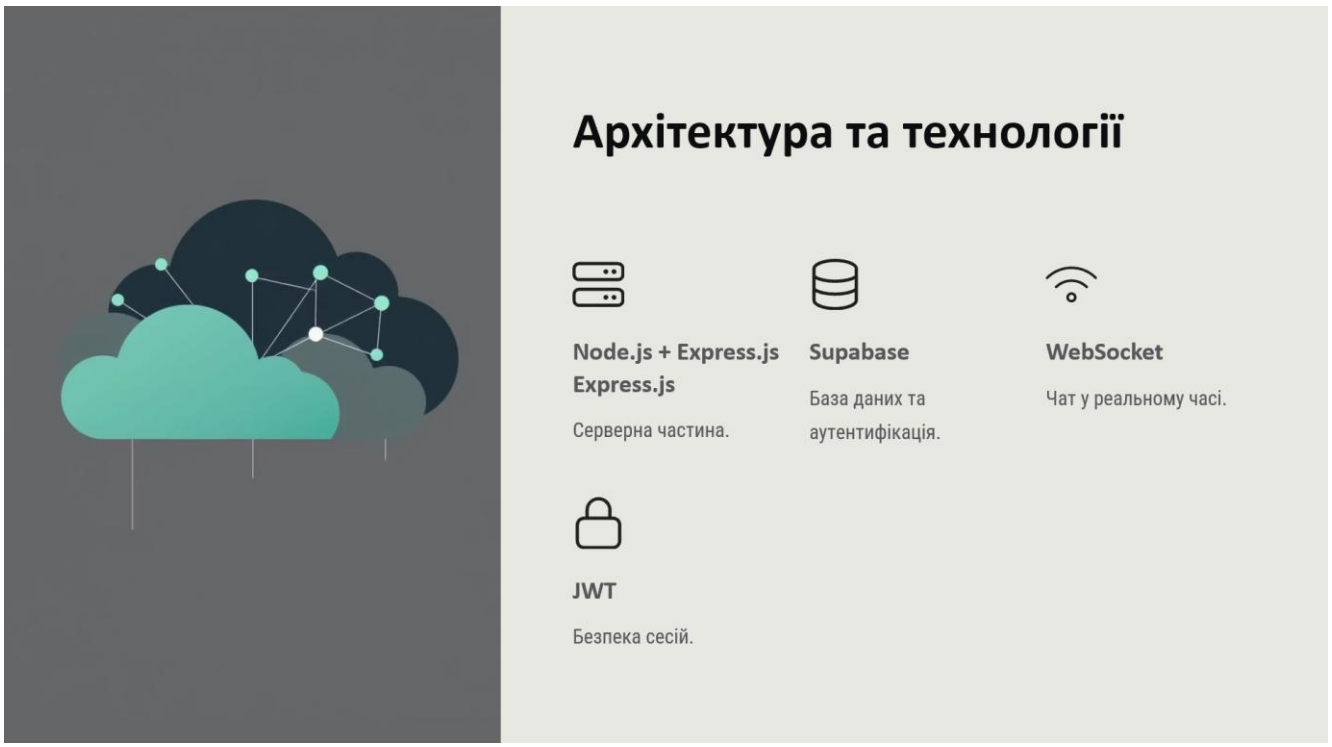
Слайд Б2 – Мета та актуальність проекту



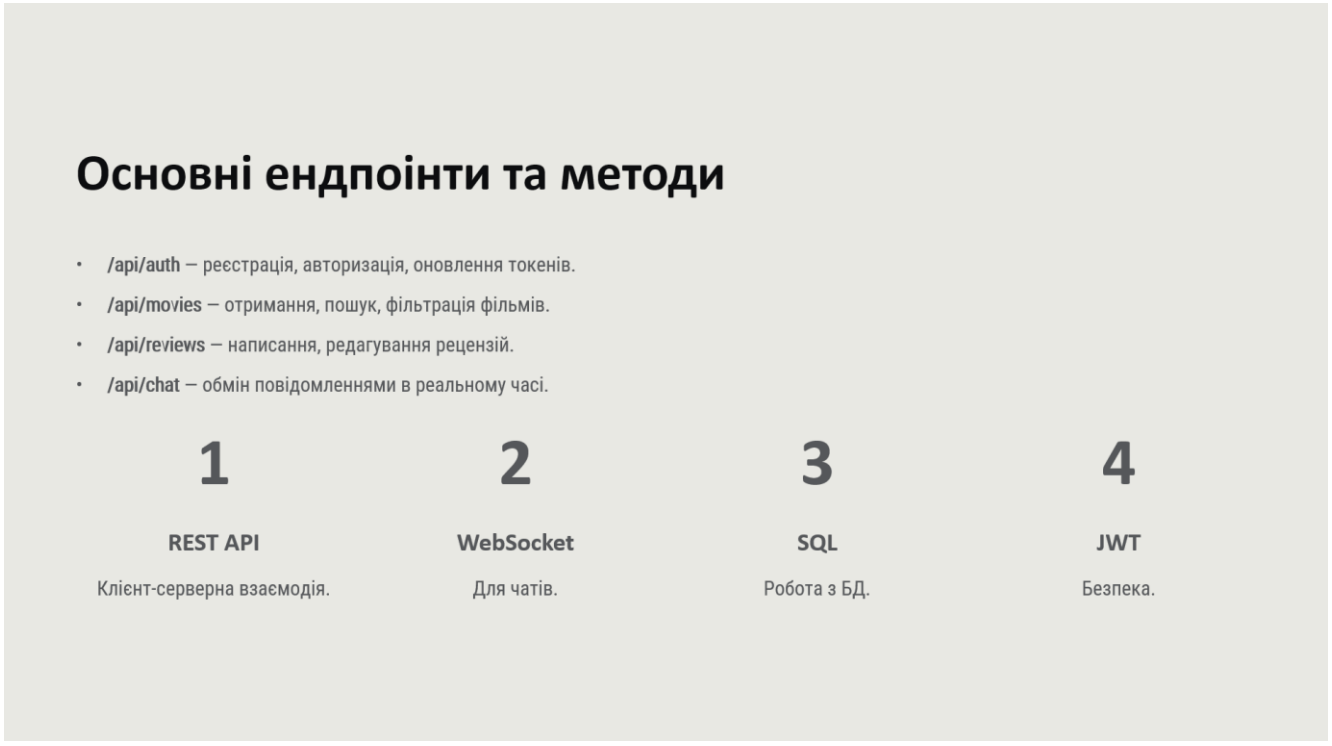
Слайд Б3 – Аналіз конкурентів та проблеми



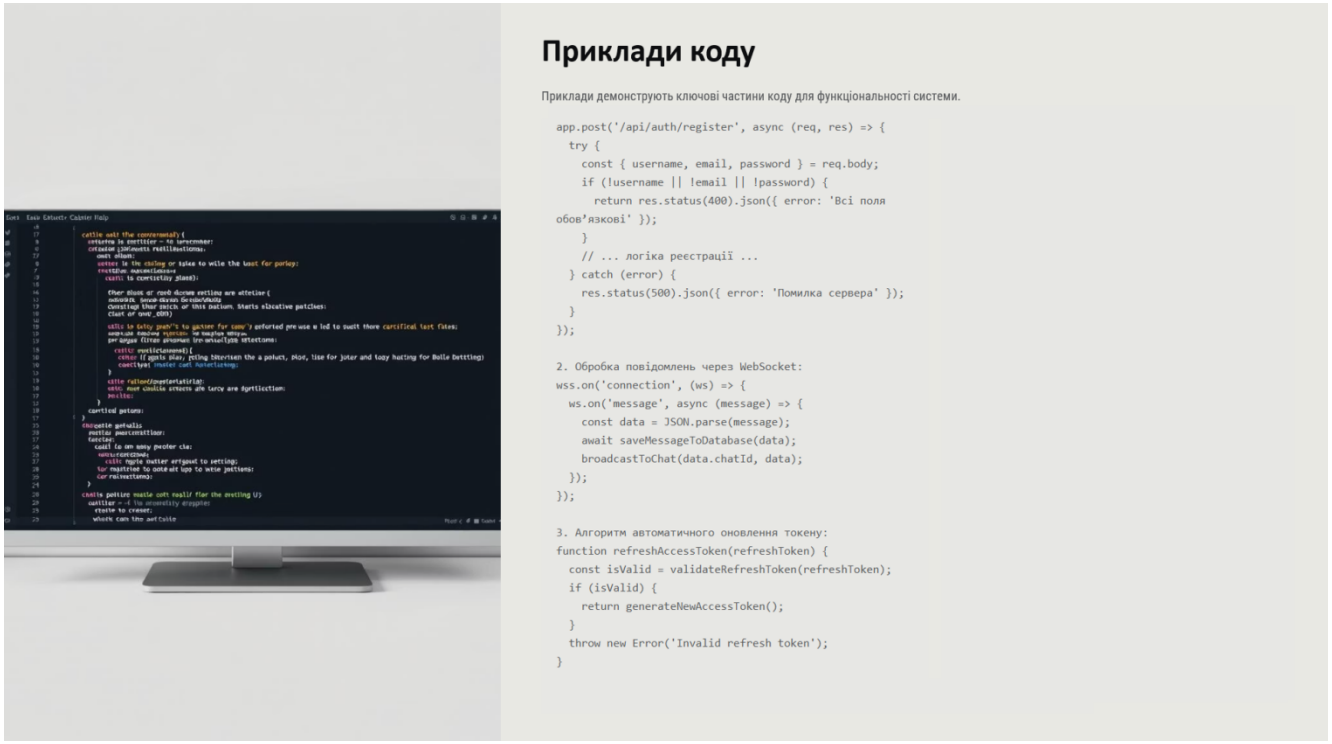
Слайд Б4 – Постановка задачі та очікувані результати



Слайд Б5 – Архітектура та технології



Слайд Б6 – Основні ендпоінти та методи



Слайд Б7 – Приклади коду

Слайд Б8 – Особливості алгоритмів та безпека

Структура бази даних

База даних спроектована для ефективного зберігання інформації.

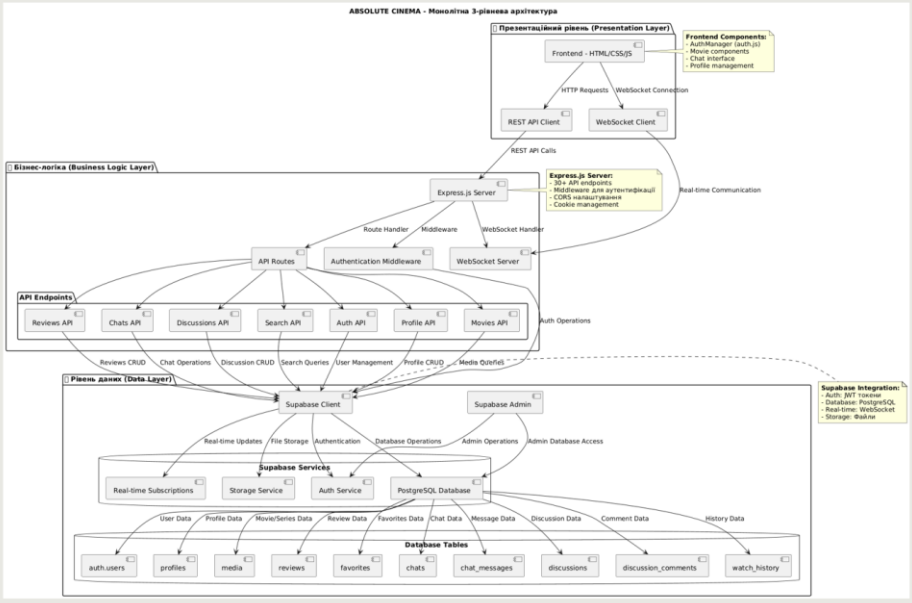
profiles	Профілі користувачів.
movies	Дані про фільми.
genres, movie_genres	Жанри фільмів.
reviews, review_likes	Рецензії та лайки до них.
chats, chat_messages	Чати та повідомлення.

Принципи: цілісність (зовнішні ключі), індексування (оптимізація запитів), резервне копіювання (Supabase).

Слайд Б9 – Структура бази даних

Архітектура Back-end

Проект ABSOLUTE CINEMA використовує монолітну архітектуру з чітким розділенням на 3 рівні



Слайд Б10 – Архітектура Back-end

Детальніше про архітектуру

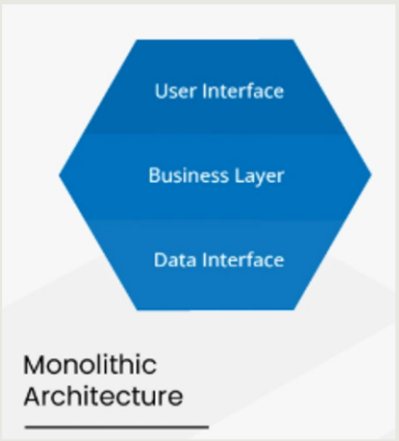
Чому саме така архітектура?

1. **Простота розробки та розгортання**

•Легке тестування та відладка
•Всі компоненти знаходяться в одному додатку
•Швидкий старт проекту без складних конфігурацій
•2. Оптимальність для середнього масштабу
•30+ API endpoints обслуговуються ефективно
•Немає накладних витрат на мікросервну комунікацію
•Централізоване управління станом та конфігурацією
•3. Інтеграція з Supabase Supabase надає готові сервіси (Auth, Database, Real-time)
•Моноліт дозволяє максимально використати ці сервіси
•Мінімізація складності інтеграції

3-рівнева структура:

Рівень даних (Data Layer) Supabase PostgreSQL як основна БД
•Supabase Auth для аутентифікації
•Real-time підписки через WebSocket
Бізнес-логіка (Business Logic Layer) Express.js сервер з 30+ API endpoints
•Middleware для аутентифікації та авторизації
•WebSocket сервер для чатів
Презентаційний рівень (Presentation Layer)REST API для frontend
•Статичні файли (HTML, CSS, JS)
•WebSocket з'єднання для real-time функцій



Слайд Б11 – Детальніше про архітектуру

Функціональність системи

Система Absolute Cinema надає широкий спектр можливостей користувачам.

Перегляд
Фільми, рецензії, дискусії.

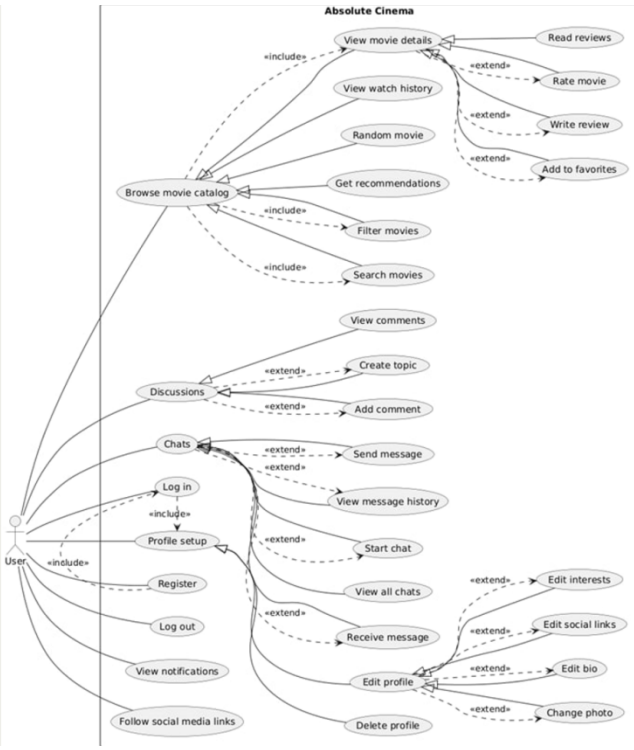
Пошук
За назвою, жанром, акторами.

Аутентифікація
Авторизація та реєстрація.

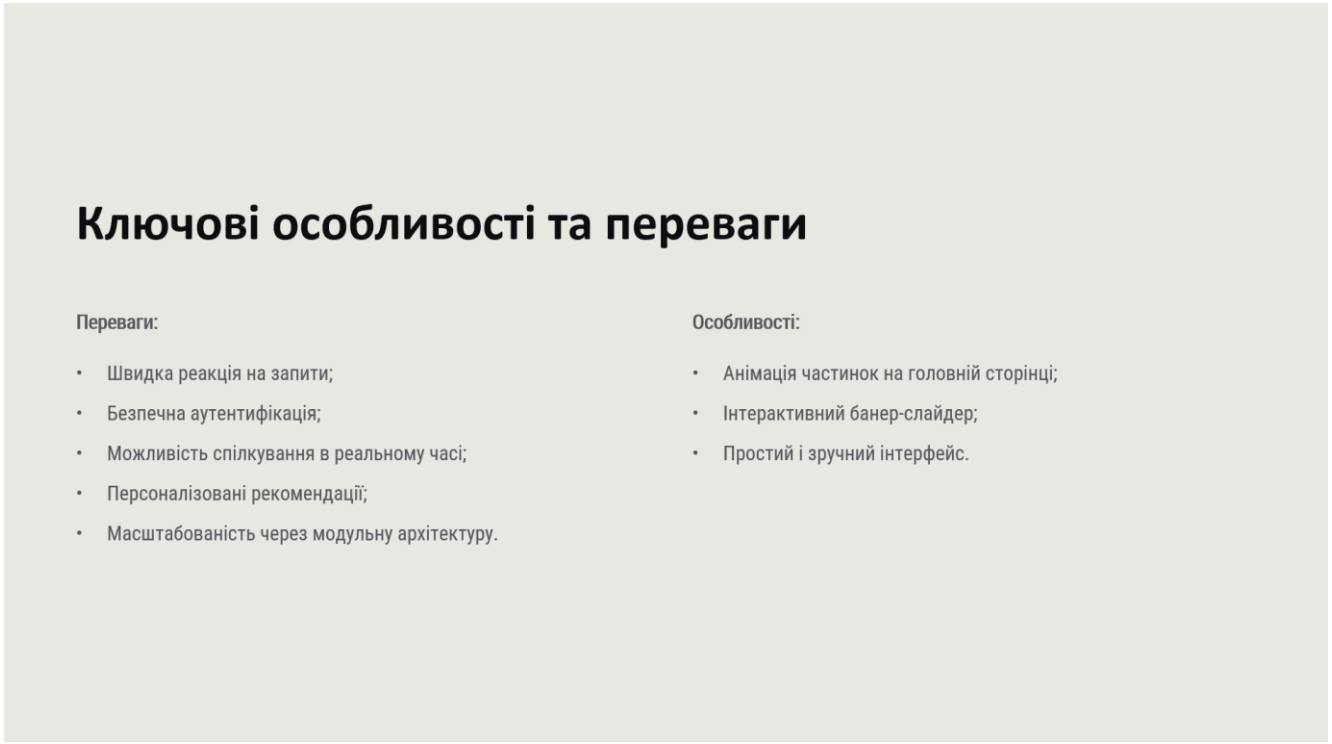
Взаємодія
Написання рецензій, додавання у вибране.

Комунікація
Чат у реальному часі.

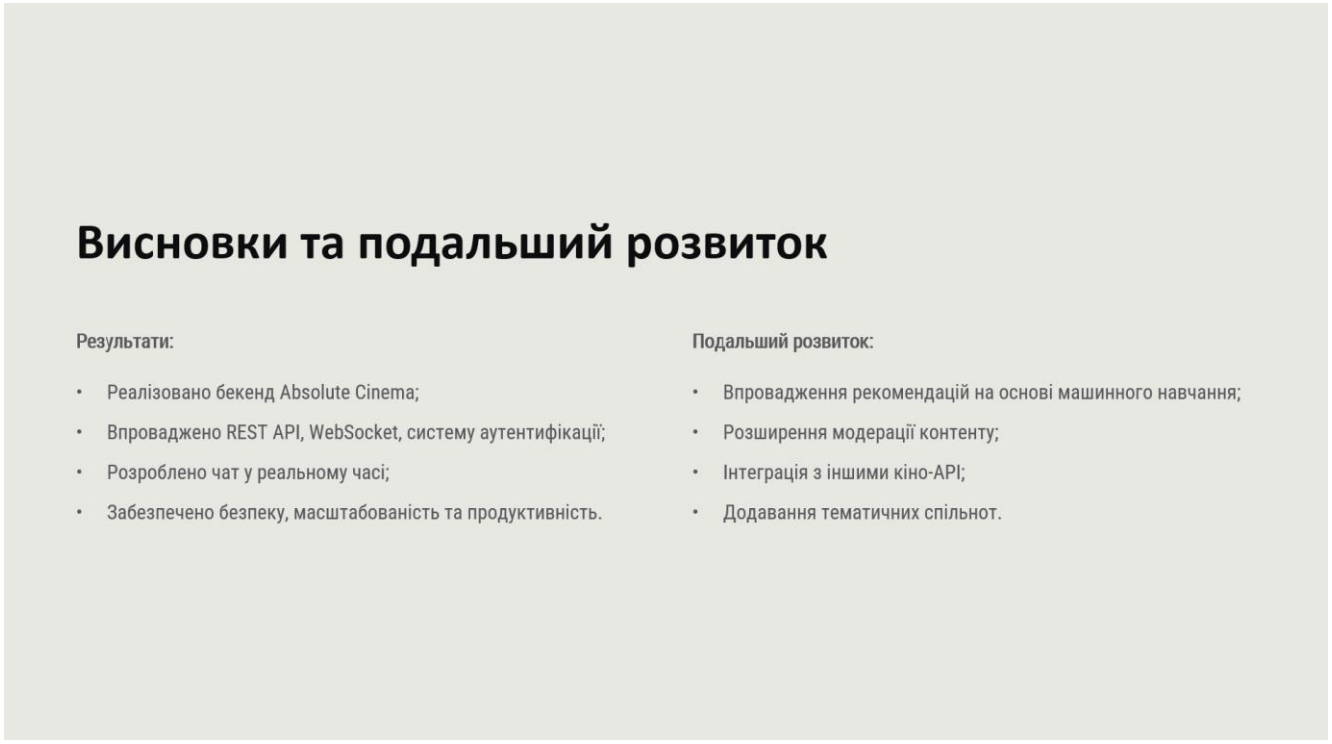
Профіль
Редагування, завантаження аватарів.



Слайд Б12 – Функціональність системи



Слайд Б13 – Ключові особливості та переваги



Слайд Б14 – Висновки та подальший розвиток

Слайд Б15 – «Дякую за увагу!»

ДОДАТОК В

Специфікація програмного забезпечення

Вебсистема для рецензій фільмів

Software Requirements Specification

1.0

21.03.2025

Гмирак Михайло Дмитрович

Лопатенко Володомир Володимирович

Сліпко Денис Віталійович

ІСТОРІЯ ЗМІН

Дата	Опис	Автор	Коментарі
19.03.2025	Створено пункти 1.1 - 1.3	Гмирак Михайло Дмитрович	
20.03.2025	Створено пункти 1.4 - 1.5, 2.1 - 2.3	Лопатенко Володомир Володимирович	
21.03.2025	Створено пункти 2.4, 2.5, 3.1 - 3.2	Сліпко Денис Віталійович	
22.03.2025	Створено пункти 3.3 - 3.5	Гмирак Михайло Дмитрович	

ЗАТВЕРДЖЕННЯ ДОКУМЕНТУ

Наступну специфікацію вимог до програмного забезпечення було прийнято та схвалено:

Підпис	Друковане ім’я	Назва	Дата

ЗМІСТ

В.1 ВСТУП

В.1.1 Огляд продукту

ABSOLUTE CINEMA — це веб-застосунок для кіноманів, який надає користувачам можливість перегляду фільмів, написання рецензій, обговорення кіно та спілкування в реальному часі. Система складається з веб-клієнту та серверної частини на базі Node.js та Supabase, які дозволяють користувачам шукати й переглядати фільми, писати рецензії, вести приватні чати через WebSocket, брати участь у дискусіях та управляти власними профілями.

В.1.2 Мета

Цей документ описує розроблюваний проект та вимоги до нього, а саме специфікацію системних, користувацьких та програмних вимог. Метою проекту є розробка веб-застосунку, який об'єднує кіноманів у спільноті, надає зручний доступ до інформації про фільми та забезпечує інтерактивну взаємодію між користувачами через рецензії, дискусії та чати в реальному часі.

В.1.3 Межі

Веб-застосунок дозволяє усім користувачам переглядати інформацію про фільми, читати рецензії та переглядати дискусії. Зареєстровані користувачі отримують доступ до розширеного функціоналу: додавання фільмів у обране, ведення історії переглядів, написання власних рецензій та оцінок, участь у дискусіях з коментуванням, створення та використання приватних чатів. Система забезпечує пошук фільмів та користувачів, персоналізовані рекомендації та інтерактивний інтерфейс з анімацією частинок та слайдером банера.

В.1.4 Посилання

1. Express.js Web Framework. URL: https://expressjs.com/ - (дата звернення 01.06.2025).

2. Supabase - The Open Source Firebase Alternative. URL: https://supabase.com/ - (дата звернення 01.06.2025).

3. WebSocket API. URL: https://developer.mozilla.org/en-US/docs/Web/API/WebSocket - (дата звернення 01.06.2025).

В.1.5 Означення та абревіатури

API — програмний інтерфейс додатку. Інтерфейс крізь який інші програми або користувачі взаємодіють з певною програмою.

JWT — токени, що використовуються для роботи модулю авторизації.

WebSocket — протокол для двостороннього зв'язку між клієнтом та сервером у реальному часі.

В.2 ЗАГАЛЬНИЙ ОПИС

В.2.1 Перспективи продукту

Веб-застосунок надає користувачам комплексний досвід для взаємодії з кінематографічним контентом. Метою проекту є створення спільноти кіноманів з можливостями перегляду деталізованої інформації про фільми, написання та читання рецензій, участі в обговореннях та спілкування в реальному часі. Розроблений продукт демонструє ефективну взаємодію між статичним фронтендом та потужним бекендом на базі Supabase. Система має значний потенціал для розвитку, включаючи розширення бази фільмів, покращення алгоритмів рекомендацій та додавання нових соціальних функцій.

В.2.2 Функції продукту

Веб-застосунок забезпечує перегляд актуальних, популярних та високорейтингових фільмів з детальною інформацією про жанри, акторський склад та рецензії. Користувачі можуть здійснювати пошук фільмів та інших користувачів, додавати фільми до вибраного, переглядати власну історію. Авторизовані користувачі мають можливість писати рецензії з оцінками, ставити лайки рецензіям інших користувачів, створювати дискусії та коментувати їх, вести приватні чати в реальному часі через WebSocket, редагувати власні профілі з аватарами та соціальними посиланнями.

В.2.3 Характеристики користувачів

Користувачами є люди, зацікавлені у кінематографі, які прагнуть ділитися думками про фільми та спілкуватися з однодумцями. Необхідні базові навички роботи з веб-браузером та розуміння принципів соціальних мереж.

У системі є 2 типи користувачів:

- гість: не зареєстрований користувач, який переглядає інформацію про фільми, читає рецензії та дискусії;
- користувач: зареєстрований учасник, який має доступ до повного функціоналу включно з написанням рецензій, участю в дискусіях та приватними чатами.

В.2.4 Загальні обмеження

Доступ до системи здійснюється через веб-інтерфейс та REST API. Для використання необхідне стабільне підключення до Інтернету та сучасний веб-браузер: Google Chrome версії 52+, Mozilla Firefox версії 48+, Microsoft Edge, Internet Explorer версії 11+, або Safari. Для роботи чатів у реальному часі потрібна підтримка WebSocket.

В.2.5 Припущення і залежності

Припускається, що API системи використовується переважно через веб-додаток. Веб-застосунок залежить від доступності сервісів Supabase для аутентифікації та зберігання даних. Користувачі мають завантажувати аватари у форматах .jpeg або .png розміром до 5МБ. WebSocket-з'єднання може бути нестабільним залежно від мережевих умов користувача.

В.3 КОНКРЕТНІ ВИМОГИ

В.3.1 Вимоги до зовнішніх інтерфейсів

В.3.1.1 Інтерфейс користувача

Інтерфейс користувача являє собою адаптивний веб-застосунок з підтримкою мобільних пристроїв.

Вимоги до веб-додатку:

- наявність інтерактивного банера-слайдера з рекомендованими фільмами;
- функціонал пошуку фільмів та користувачів з миттєвими результатами;
- підтримка форматів зображень — jpeg, png для аватарів користувачів;
- можливість відображення детальної інформації про фільми з жанрами та акторським складом;
- інтерфейс для управління вибраними фільмами та історією переглядів;
- система написання та перегляду рецензій з оцінками та лайками;
- функціонал створення дискусій та коментування;
- інтерфейс приватних чатів з підтримкою реального часу;
- можливість редагування профілю з аватаром та соціальними посиланнями;
- анімація частинок на фоні для покращення візуального досвіду.

В.3.1.2 Апаратний інтерфейс

Система не взаємодіє безпосередньо з апаратним забезпеченням користувача, окрім стандартних засобів введення (клавіатура, миша, сенсорний екран) через веб-браузер.

В.3.1.3 Програмний інтерфейс

Програмний інтерфейс реалізований як REST API на базі Express.js з додатковим WebSocket-сервером для чатів. API включає маршрути для аутентифікації, управління профілями, роботи з фільмами, рецензіями, дискусіями та чатами. WebSocket-інтерфейс забезпечує обмін повідомленнями в реальному часі.

В.3.1.4 Комунікаційний протокол

Веб-застосунок використовує протокол HTTPS для REST API запитів та WSS (WebSocket Secure) для з'єднань реального часу. Аутентифікація здійснюється через JWT токени, що передаються в HTTP заголовках або WebSocket повідомленнях.

В.3.1.5 Обмеження пам'яті

Рекомендовані характеристики сервера:

- процесор з тактовою частотою вищою за 2.0 ГГц;
- щонайменше 4 гігабайти оперативної пам'яті;
- 20 гігабайт вільного місця на жорсткому диску або більше;
- стабільне підключення до Інтернету для роботи з Supabase.

В.3.1.6 Операції

Система підтримує наступні основні операції:

- CRUD операції з фільмами, рецензіями, профілями користувачів;
- операції пошуку та фільтрації контенту;
- операції аутентифікації та авторизації;
- операції обміну повідомленнями в реальному часі;
- операції завантаження та обробки зображень.

В.3.1.7 Функції продукту

Список функцій користувача-гостя:

- перегляд списку фільмів (новинки, популярні, високорейтингові);
- перегляд детальної інформації про фільми;
- читання рецензій та їх оцінок;
- перегляд дискусій та коментарів;
- пошук фільмів за назвою;
- реєстрація та авторизація у системі.

Список функцій авторизованого користувача:

- усі функції гостя плюс:
- додавання фільмів до вибраного та перегляд власної колекції;
- ведення та перегляд історії переглядів;
- написання, редагування та видалення власних рецензій;
- оцінювання фільмів та лайки рецензій інших користувачів;
- створення дискусій та участь у коментуванні;
- пошук та перегляд профілів інших користувачів;
- створення та участь у приватних чатах;
- редагування власного профілю, завантаження аватара;
- отримання сповіщень про непрочитані повідомлення.

В.3.1.8 Припущення і залежності

Припускається, що сервер має доступ до Supabase та стабільне інтернет-з'єднання. Користувачі повинні мати базові навички роботи з веб-інтерфейсами. Припускається наявність на сервері Node.js версії 16+ та всіх необхідних залежностей згідно з package.json.

В.3.2 Властивості програмного продукту

ABSOLUTE CINEMA представляє собою повнофункціональний веб-застосунок для кіноманів з широким спектром можливостей. Веб-застосунок

забезпечує повний цикл управління користувачами через реєстрацію, аутентифікацію та персоналізовані профілі з можливістю редагування особистої інформації. Каталог фільмів включає детальну інформацію про кожен фільм з жанрами, акторським складом, рейтингами та описами, а також забезпечує різноманітні способи пошуку та фільтрації контенту.

Персоналізація реалізована через систему улюблених фільмів, історію переглядів та індивідуальні рекомендації. Соціальна складова включає можливість написання рецензій з оцінками, створення та участь в дискусіях, а також систему лайків для оцінювання контенту інших користувачів. Комунікація в реальному часі забезпечується через приватні чати з підтримкою WebSocket технології для миттєвого обміну повідомленнями.

Продуктивність системи забезпечується через ефективну архітектуру з розділенням клієнтської та серверної логіки, використання RESTful API та WebSocket для реального часу. Масштабованість досягається завдяки модульній структурі та використанню Supabase як backend-as-a-service рішення з PostgreSQL базою даних.

Безпека реалізована через JWT токени для аутентифікації, middleware для авторизації та валідацію даних на сервері. Надійність забезпечується обробкою помилок, автоматичним оновленням токенів та резервним копіюванням через Supabase. Зручність використання досягається через адаптивний дизайн, інтуїтивний інтерфейс та оптимізацію для мобільних пристроїв.

Архітектура веб-застосунку побудована за принципом Client-Server з чітким розділенням ролей. Backend реалізовано на Node.js з Express.js фреймворком та Supabase для управління даними та аутентифікацією. Frontend використовує чистий JavaScript без додаткових фреймворків, HTML5 та CSS3 з підтримкою сучасних веб-стандартів.

Візуальна складова включає анімації через Canvas API, інтерактивні елементи інтерфейсу та сучасну типографіку з Google Fonts. Система

розгортання підтримує різні середовища через змінні оточення та автоматизовані скрипти запуску.

В.3.3 Атрибути програмного продукту

В.3.3.1 Надійність

Усі помилки у веб-застосунку відображаються користувачеві через зрозумілі повідомлення. Час недоступності системи не повинен перевищувати 15 хвилин на добу. Дані користувачів захищені відповідно до політики безпеки Supabase, втрата персональних даних неприпустима. Веб-застосунок має механізми відновлення після збоїв та резервного копіювання критичних даних.

В.3.3.2 Доступність

Веб-застосунок забезпечує 99.5% часу безперебійної роботи. Інтерфейс адаптований для користувачів з обмеженими можливостями, включаючи підтримку програм зчитування екрана та навігацію за допомогою клавіатури. Час відгуку на користувацькі дії не перевищує 2 секунд для стандартних операцій.

В.3.3.3 Безпека

Аутентифікація здійснюється через Supabase Auth з використанням JWT токенів. Паролі зберігаються у хешованому вигляді. Використовуються підписані HTTP-only куки для зберігання токенів. WebSocket-з'єднання авторизуються через передачу валідних токенів. Усі API-запити проходять перевірку авторизації для захищених ресурсів. Веб-застосунок захищений від основних веб-загроз: XSS, CSRF, SQL-ін'єкцій.

В.3.3.4 Супроводжуваність

Код структурований згідно з принципами модульності, всі модулі мають чітко визначені інтерфейси. Система логування забезпечує

відстеження помилок та моніторинг продуктивності. Документація API автоматично генерується та підтримується в актуальному стані. Використовується система контролю версій для відстеження змін.

В.3.3.5 Переносимість

Використання веб-технологій забезпечує доступність на всіх платформах з сучасними браузерами. Серверна частина може бути розгорнута на різних хостинг-платформах, що підтримують Node.js. Docker-контейнеризація спрощує розгортання та масштабування системи.

В.3.3.6 Продуктивність

Система підтримує одночасну роботу до 1000 користувачів. Час завантаження сторінок не перевищує 3 секунд при стандартному інтернет-з'єднанні. WebSocket-з'єднання забезпечують затримку повідомлень менше 100мс. Веб-застосунок використовує кешування для оптимізації продуктивності бази даних.

В.3.4 Вимоги бази даних

База даних реалізована на PostgreSQL через платформу Supabase та включає наступні основні таблиці:

- profiles — профілі користувачів з персональною інформацією;
- media — інформація про фільми та серіали;
- genres, media_genres — жанри та їх зв'язки з медіаконтентом;
- people, media_people — інформація про акторів та їх участь у фільмах;
- reviews, review_likes — рецензії користувачів та їх оцінки;
- favorites, watch_history — вибрані фільми та історія переглядів;
- discussions, discussion_comments — дискусії та коментарі;
- chats, chat_messages — приватні чати та повідомлення.

Веб-застосунок забезпечує цілісність даних через зовнішні ключі, індексування для оптимізації запитів та автоматичне резервне копіювання через Supabase.

В.3.5 Інші вимоги

Веб-застосунок повиннен підтримувати міжнародну локалізацію для розширення географії користувачів. Необхідна інтеграція з зовнішніми API для отримання актуальної інформації про фільми. Веб-застосунок має бути готовим до масштабування при зростанні кількості користувачів та контенту.

В.4 ДОДАТКОВІ МАТЕРІАЛИ

Додаткові матеріали включають діаграму прецедентів, діаграму архітектури системи, схеми бази даних (див. рис. В.4.1 – В.4.3).

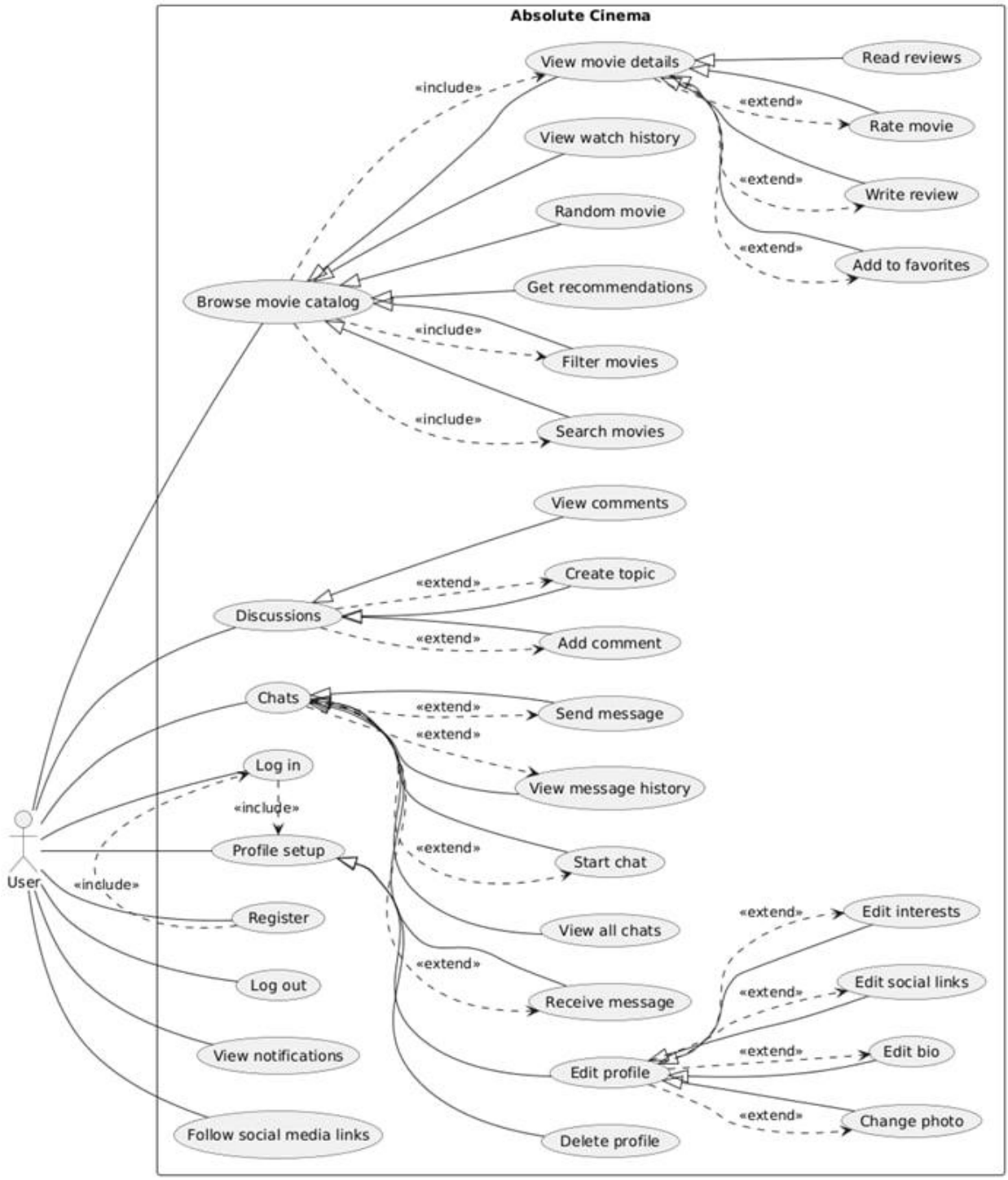


Рисунок В.4.1 – UML Діаграма прецедентів

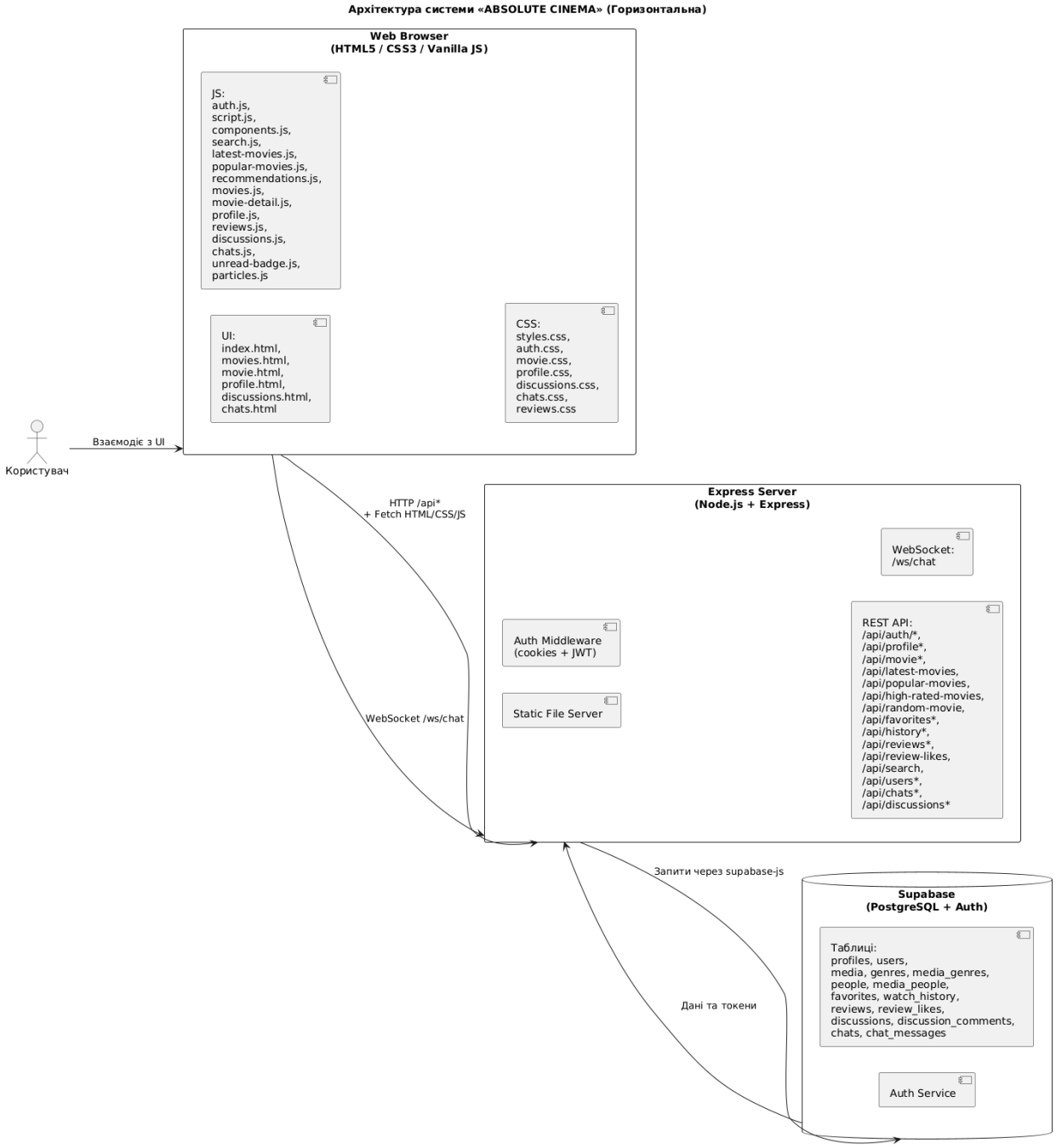


Рисунок В.4.2 – Схема архітектури системи

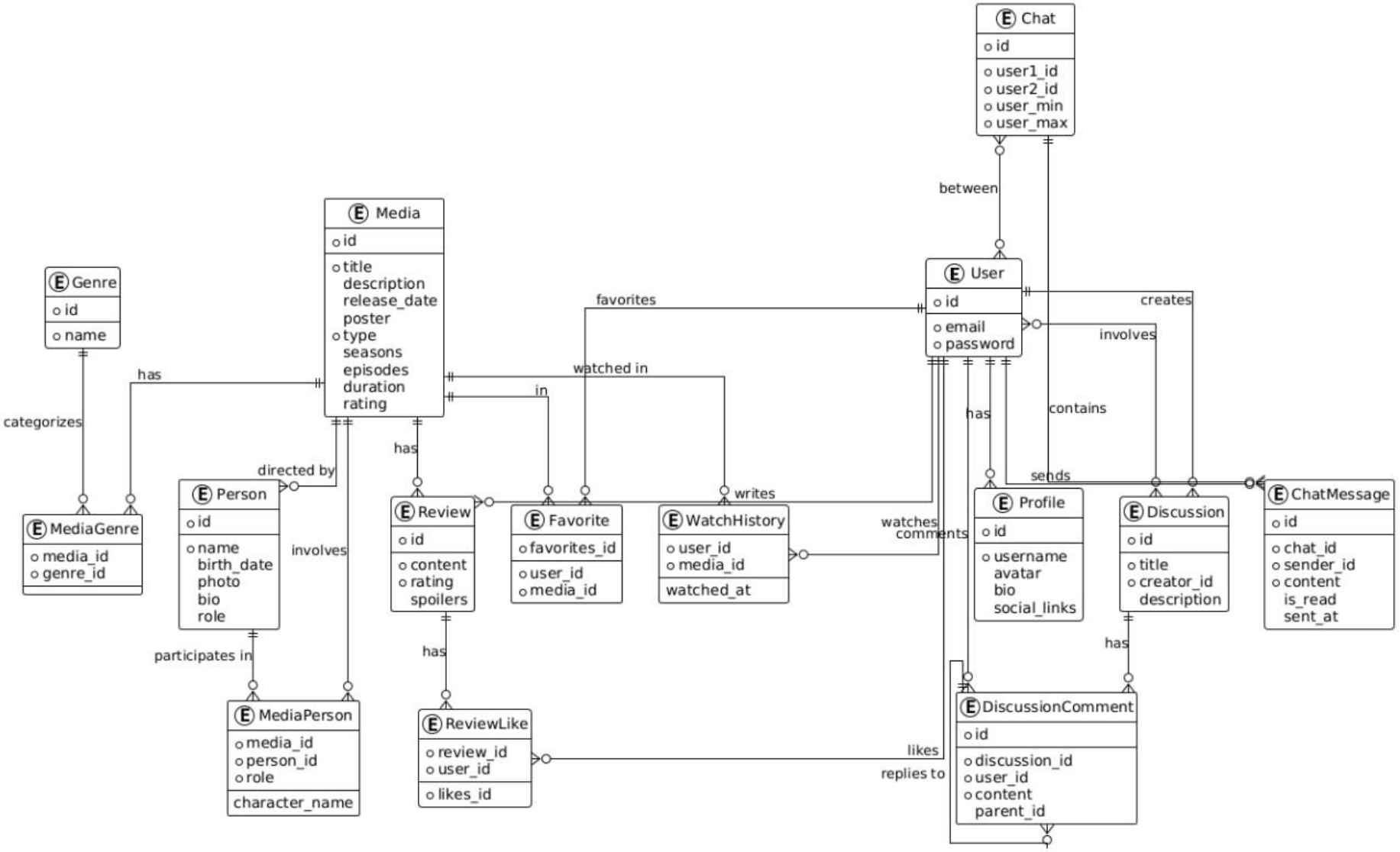


Рисунок В.4.3 – Схема бази даних