Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання та програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття ступеня вищої освіти бакалавр
зі спеціальності 121 – Інженерія програмного забезпечення

На тему: Розробка функціональної web-платформи підтримки авторів літературних творів

Засвідчую, що в цій кваліфікаційній роботі немає запозичень із праць інших авторів без відповідних посилань.

Студентка гр. ІПЗ-21-1_____ / Є.В.Каршина /

Керівник
кваліфікаційної роботи _____ / І.А.Котов/

Завідувач кафедри _____ / А.М. Стрюк /

Кривий Ріг
2025

Криворізький національний університет
Факультет: Інформаційних технологій
Кафедра Моделювання та програмного забезпечення
Ступінь вищої освіти: бакалавр
Спеціальність: 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Зав. кафедри
_____ А.М.Стрюк
«___» _____20___ р.

ЗАВДАННЯ
на кваліфікаційну роботу
студентці групи ІПЗ-21-1 Каршиної Єлизавети Валентинівни

1. Тема: Розробка функціональної web-платформи підтримки авторів літературних творів затверджено наказом по КНУ № 199с від «10» квітня 2025 р.
2. Термін подання студентом закінченої роботи: «02» червня 2025р
3. Вихідні дані по роботі: розроблювана платформа має допомагати авторам-початківцям знаходити перших читачів при публікації своїх перших творів
4. Зміст пояснювальної записки: виконати аналіз предметної області, спроектувати платфому для публікації творів, розробити програмну реалізацію платформи
5. Перелік ілюстрованого матеріалу: блок-схеми алгоритмів, знімки інтерфейсів, UML-діаграми, знімки прикладів

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	Пошук науково-практичних джерел та інтернет-ресурсів з тематики роботи	29.01.25 – 03.02.25
2	Аналіз існуючих підходів і методів вирішення проблеми	04.02.25 – 20.02.25
3	Формулювання актуальності і постановка завдань роботи	21.02.25 – 01.03.25
4	Оформлення матеріалів першого розділу роботи	02.03.25 – 15.03.25
5	Розробка структурної і функціональної схем та основних алгоритмів програмного проєкту	16.03.25 – 30.03.25
6	Оформлення матеріалів другого розділу роботи	31.03.25 – 10.04.25
7	Розробка візуального інтерфейсу, баз даних і програмних модулів проєкту	11.04.25 – 02.05.25
8	Оформлення матеріалів третього розділу роботи	03.05.25 – 09.05.25
9	Оформлення додатків	10.05.25 – 21.05.25
10	Тестування розробленої програми	22.05.25 – 01.06.25
11	Оформлення пояснювальної записки та супровідних документів	02.06.25 – 08.06.25

Дата видачі завдання: «28» січня 2025 р.

Студентка _____ / Є. В. Каршина /

Керівник роботи _____ / І. А. Котов /

РЕФЕРАТ

ПЛАТФОРМА, ПОЧИНАЮЧІ АВТОРИ, WEB-САЙТ, ЛІТЕРАТУРНІ ТВОРИ, АВТОРСЬКЕ ПРАВО

Пояснювальна записка: 100с., 5 табл., 26 рис., 15 дод., 8 джерел.

Метою кваліфікаційної роботи є розробка функціональної web-платформи підтримки авторів лтературних творів.

Головною задачею при розробці web-платформи є вирішення проблеми пошуку читачів для непопулярних та починаючих авторів. Під час роботи було проведено аналіз предметної області, де було виявлено критичні моменти та їх вплив на пошук читачів. Було проведено аналіз існуючих платформ та web-сайтів задля вдосконалення та виправлення проблем з якими може зіткнутися автор.

Після аналізу результатів можна виділити кілька ключових моментів: збереження усіх версій творів займає багато пам’яті (особливо якщо твір займає більше 100 сторінок), пошук за жанром або персонажами може вивести не усі наявні твори.

Як результат, ми маємо функціональну web-платформу для допомоги починаючим авторам.

ABSTRACT

PLATFORM, BEGINNER AUTHORS, WEBSITE, LITERARY WORKS, COPYRIGHT

Explanatory notes: 100 p., 5 tab., 26 fig., 15 app., 8 sources.

The goal of the qualification project is the development of a functional web platform to support authors of literary works.

The main objective in developing the web platform is to solve the problem of finding readers for little-known and beginner authors. During the work, an analysis of the subject area was carried out, which identified critical issues and their impact on the process of reaching readers. An analysis of existing platforms and websites was conducted to improve and correct problems that authors may face.

After analyzing the results, several key issues were identified: storing all versions of literary works consumes a lot of memory (especially if a work exceeds 100 pages), and searching by genre or characters may not return all available works.

As a result, a functional web platform was developed to assist beginner authors.

ЗМІСТ

ВСТУП

У наш час існує дуже багато авторів, які тільки починають свій шлях та хочуть опублікувати свої твори на широкий загал. На сьогодні більшість авторів обирають викласти свій твір в Інтернеті перед тим, як іти до видавництва. Важливо отримати зворотній відгук від читачів, для того щоб відредагувати текст та вже потім друкувати його та продавати у книгарнях.

Більшість письменників використовують популярні онлайн платформи, які забирають свою частку з кожної публікації та продажу книги. На таких платформах стає складніше отримати зворотній відгук від читачів та обговорити свій твір з ними та важчим стає контролювати авторське право, через те, що кожен користувач може зайти на сайт прочитати твір та скопіювати його.

Для вирішення цих проблем необхідно створити web-платформу, що дозволить авторам викладати твори, отримувати зворотній зв’язок та комунікувати з читачами. Така платформа має забезпечувати користувачів усіма необхідними функціями, такими як: редагування, публікація, коментування, оцінювання творів, надавати доступ до приватного чату з читачами та зберігати усю інформацію про користувачів та твори.

Під час дослідження буде виявлено основні потреби авторів літературних творів та оснащення платформи усіма необхідними функціями. Web-платформа має стати інструментом, що допомагатиме авторам здобувати омріяну популярність, а читачами віднаходити улюблені твори та авторів.

Таким чином, запропонована web-платформа підтримки авторів літературних творів допоможе цифровізувати літературну творчість та допоможе підтримати починаючих письменників.

1. ПОСТАНОВКА ЗАВДАННЯ РОЗРОБКИ ФУНКЦІОНАЛЬНОЇ WEB-ПЛАТФОРМИ ПІДТРИМКИ АВТОРІВ ЛІТЕРАТУРНИХ ТВОРІВ

1.1 Загальний аналіз функціонування багатокористувацьких web-платформ підтримки літературної творчості

У сучасному світі починаючим авторам літературних творів необхідно пройти багато випробувань для того, щоб їх рукопис не лише розглянули, а в подальшому й надрукували та виставили у книгарнях. На сьогодні, нові письменники стикаються з багатьма труднощами для того, аби їх твори були надруковані, а потім їх помітили сучасні читачі.

На сьогодні багато починаючих авторів використовують багатокористувацькі онлайн-платформи для розміщення своїх творів, для того, щоб привернути увагу потенційних читачів та отримати зворотній відгук. Приклади онлайн-платформ наведено на рисунках 1.1-1.3 («Wattpad» [1], «Букнет» [2], «Medium» [3]).

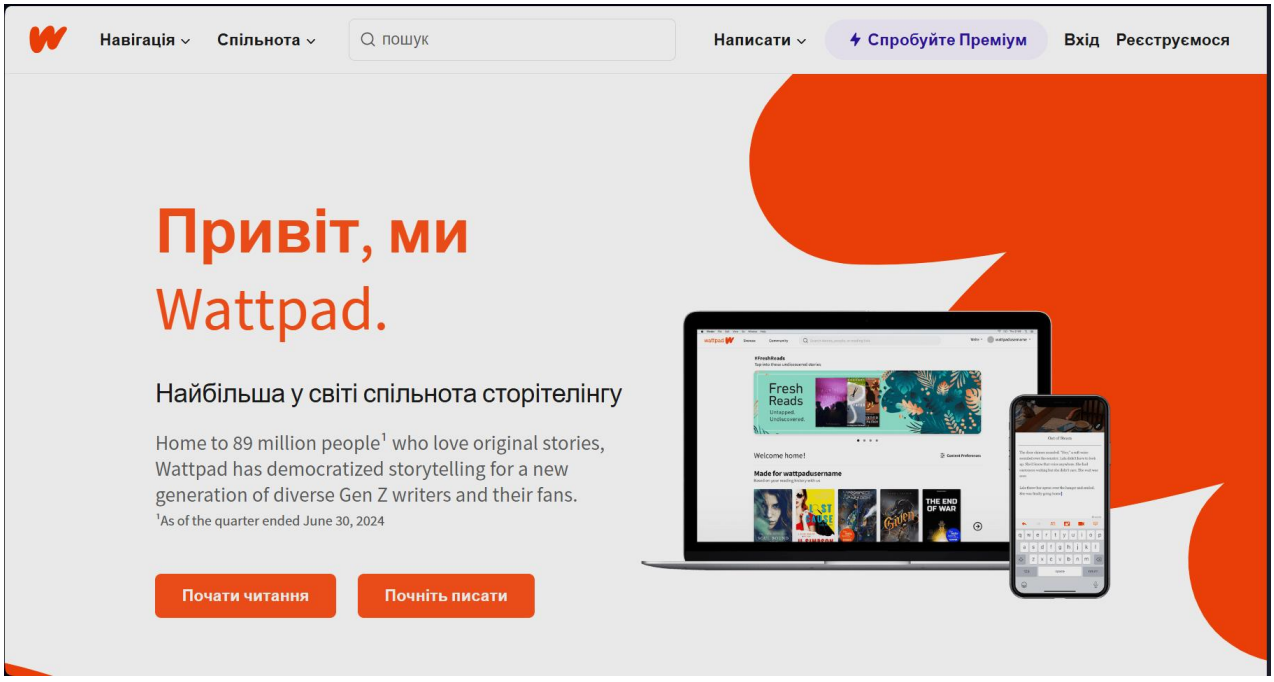

Рисунок 1.1 – Головна сторінка платформи «Wattpad»

Такі платформи можуть запропонувати не лише безкоштовне розміщення твору, а ще й доступ до аналітики перегляду читачами твору, на якій сторінці читач припинив читання, доступ до коментарів та оцінок. Деякі web-сайти для розміщення творів також пропонують монетизацію для творів, за умови, що автор набув достатньої популярності та його читачі зацікавлені у тому, щоб продовжити читання опублікованого твору.

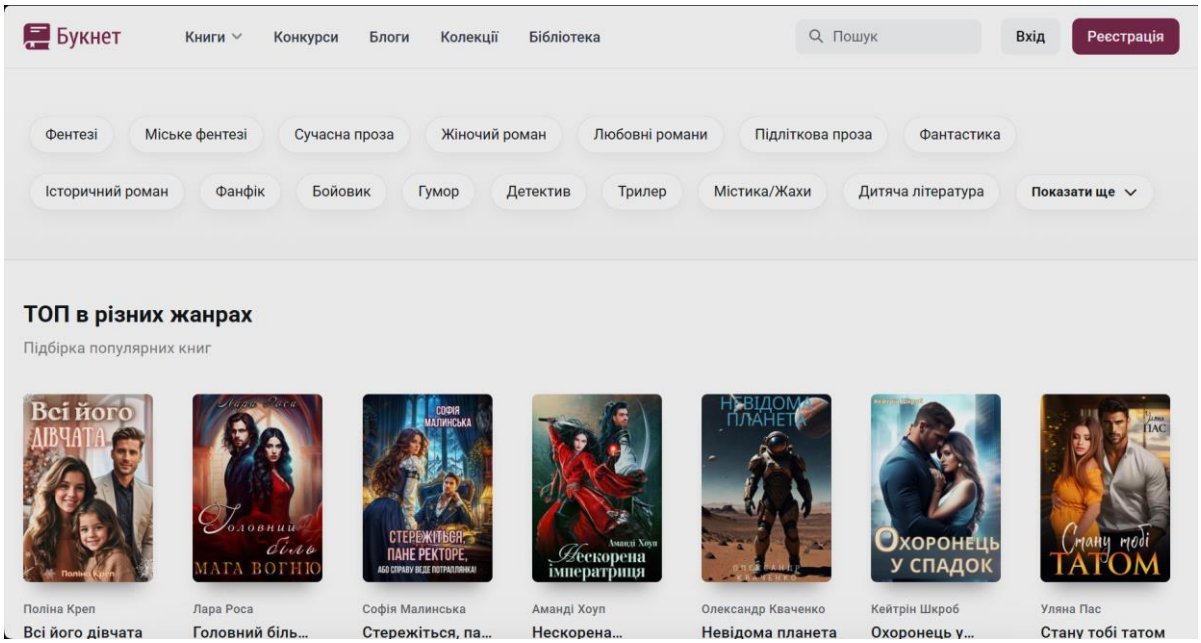

Рисунок 1.2 – Головна сторінка платформи «Букнет»

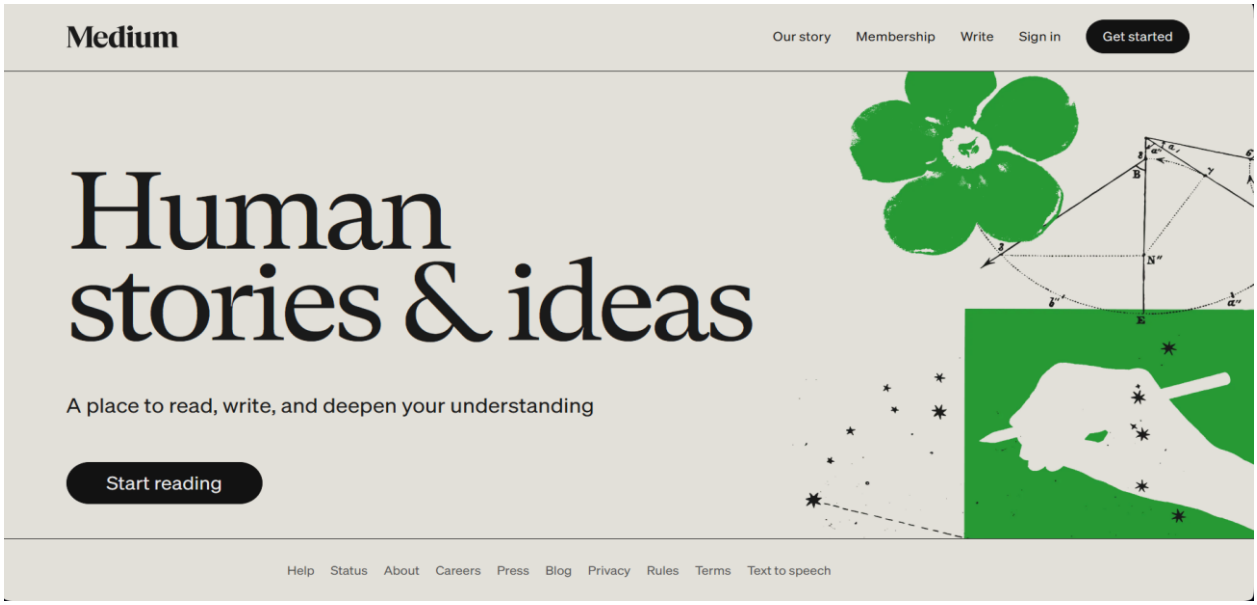

Рисунок 1.3 – Головна сторінка платформи «Medium»

Такі платформи розроблені саме для того, щоб допомогти авторам-початківцям знайти аудиторію та в подальшому спонсорувати друк їх найвдаліших творів задля збільшення їх популярності. Також автори отримують зворотній зв’язок від читачів, що дозволяє у реальному часі змінювати власний твір, якщо автор згоден з думкою своїх читачів. Викладення власного твору на подібній платформі стимулює розвиток твору через відгуки читачів, його редагування, а після цього й друк найкращої версії рукопису.

Тож, багатокористувацькі web-платформи для публікації творів є посередниками між авторами та їх читачами, що беруть на себе більшу частину роботи, дозволяючи авторам створювати нові твори та надалі викладати їх на тій самій платформі та здобувати більшу популярність.

1.2. Аналіз web-технологій розробки онлайн-порталів колективної творчості

Під час аналізу web-технологій розробки онлайн порталів колективної творчості було з’ясовано на які аспекти слід звернути увагу при створені web-платформи, яка допомагатиме майбутнім авторам почати свій шлях та опублікувати перші твори.

Сучасні автори все частіше звертаються до електронного формату, тому що це зручно та ефективно. Пройшли часи, коли письменники писали твори власноруч на папері або друкували на спеціальних машинах. Їм на заміну прийшли комп’ютери на яких не лише зручно друкувати свої твори, а й зберігати їх і не боятися що дані зітруться або зникнуть.

Також в останні роки все більше авторів починають комунікувати одне з одним та висловлювати особисту думку про той чи інший твір, що допомагає навіть досвідченим авторам корегувати свої твори, роблячи їх кращими.

Наразі доречно спілкуватися не лише з іншими авторами, що випускають власні твори, а й з читачами, які ці самі твори читають. Тому все більше і більше

письменників починають використовувати спеціальні онлайн-платформи на яких вони публікують твори, отримують критику та не тільки вдосконалюють свій рукопис, а ще й популяризують його, а іноді й отримують за це кошти.

Спеціальні web-платформи для допомоги починаючим авторам, проходять з письменником увесь шлях, починаючи від завантаження/написання твору, до створення обкладинки, анотації, обирання тегів, публікації та просування твору. Автори тепер можуть переглядати аналітику переглядів та відстежувати на якому моменті читачі втрачають інтерес до твору та виправити це.

Однієї з важливих частин створення будь-якої платформи є інтерфейс користувача. Кожен користувач, відкривши головну сторінку має розуміти як використовувати сайт та які функції він може надати, тож найважливішим і найпершим пунктом при створенні має бути інтуїтивно зрозумілий інтерфейс.

На самому початку необхідно створити макет майбутнього сайту на якому розмістити усі необхідні компоненти, такі як: логотип, меню сайту та коротка інформація про можливості платформи, яка допоможе користувачу зрозуміти що і як працює.

Після створення макету є важливим підібрати вдалі рішення, які допоможуть втілити дизайн сайту у життя.

Задля клієнтської частини web-платформи було б доречно обрати наступні рішення:

1. HTML5 – мова розмітки, яка дозволить задати структуру сайту та його вмісту, розмістити усі важливі елементи;
2. CSS3 – каскадна таблиця стилів, яка дозволить покращити вигляд web-платформи, розмістити усі необхідні компоненти у правильних місцях та виділити їх таким чином, щоб користувач одразу зрозумів, який компонент за що відповідає;
3. JavaScript – мова, що забезпечує різні візуальні ефекти та функції на стороні клієнта та надсилає запити для подальшої обробки їх сервером;

4. Vue.js – фреймворк для відображення різних пунктів меню для окремих ролей користувачів;

На відміну від клієнтської частини, яка має покращити досвід користувача у використанні сайту, серверна частина відповідає за обробку запитів користувача, виведення відповідей та за час, за який його запити обробляються.

Багатокористувацька web-платформа має обробляти велику кількість запитів щосекунди та коректно надавати відповіді користувачам.

Для створення серверної частини платформи було прийнято обрати наступне рішення: PHP – мова, яка повністю відповідає за сторону сервера, обробку даних (збереження, редагування, видалення);

Важливою також є хмарне сховище, яке зберігатиме інформайцію про користувачів, їх оцінки, коментарі, твори та примітки. Усі користувачі мають мати доступи до творів у будь-який час та мати змогу коментувати або редагувати їх (за умови, що користувач автор чи співавтор).

Для web-платформи підтримку майбутніх авторів було б доречно обрати наступне рішення: PostgreSQL – потужна СУБД для підтримання великого об’єму даних та складних запитів.

1.3. Аналіз систем-аналогів web-платформ підтримки літературної творчості

Наразі існує багато систем-аналогів, які допомагають як авторам початківцям публікувати свої перші твори та здобувати популярність серед читачів, так і досвідченим авторам, які хочуть налагодити контакт зі своєю аудиторією, зробити висновки про реакцію на вже опубліковані твори та зробити роботу над помилками.

Такі платформи підтримують велику кількість користувачів та творів, багато функцій є безкоштовними, інші ж потребують оплати задля доступу. Деякі платформи надають можливість для монетизації опублікованих творів та можуть стати спонсорами для друку більш популярних творів на web-сайті.

Однією з таких платформ є «Wattpad» [1] (приклад інтерфейсу користувача зображено на рисунку. 1.4), який є однією з найпопулярніших платформ для тих авторів, які прагнуть у майбутньому надрукувати власні твори. Платформа дозволяє не тільки опублікувати рукопис, а й додати до нього обкладинку, яка приверне увагу читачів. Також перед публікацією автор може встановити теги, за якими зацікавлена аудиторія зможе знайти твір за вподобаннями. Платформа надає більшість послуг безкоштовно, що робить її більш популярною серед користувачів.

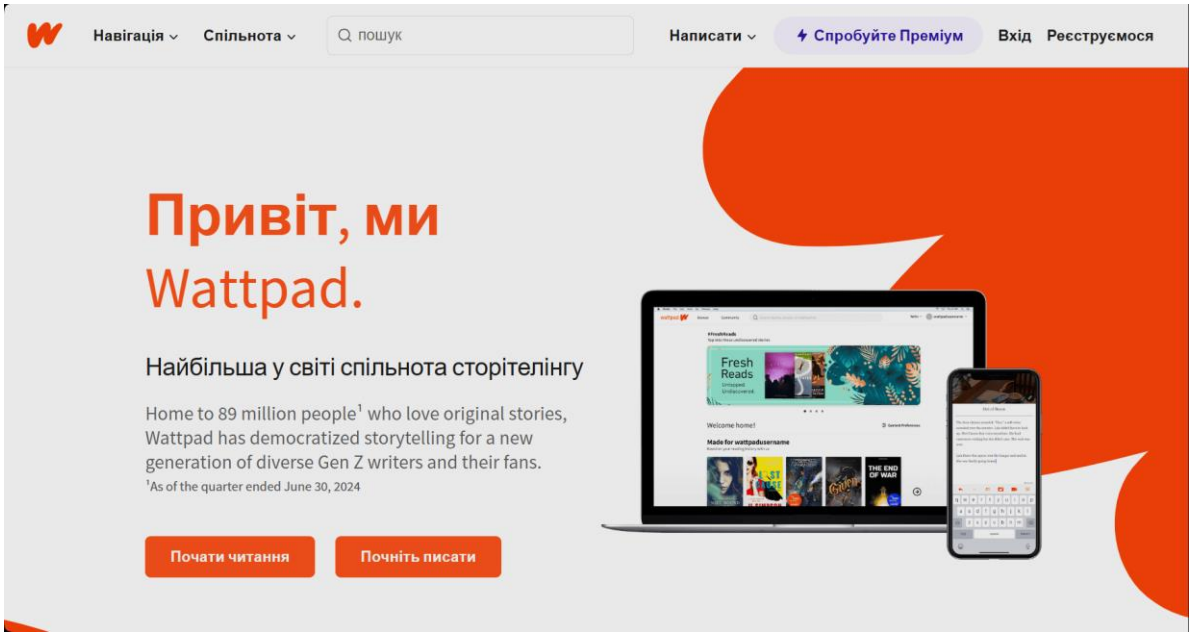

Рисунок 1.4 – Приклад інтерфейсу користувача web-платформи «Wattpad»

Перевагами є простий інтерфейс, велика база користувачів та доступність більшості функцій.

Недоліками є боротьба авторів за увагу, погано реалізована функція пошуку за назвою та за тегами.

Ще однією платформою для розміщення книг є «Букнет» [2] (приклад інтерфейсу користувача зображено на рисунку 1.5) – українська платформа, що підтримує починаючих авторів та допомагає віднайти свою аудиторію. Сайт надає послуги монетизації творів та дозволяє розмістити власний твір з

обкладинкою, задля привернення уваги потенційних читачів. Допомагає авторам продавати свої книги, друкувати їх у подальшому, отримувати зворотній відгук від шанувальників не лише через коментарі до твору а й у приватному чаті.

Перевагами є монетизація творчості та можливість заробити на улюбленій справі, простий та зрозумілий інтерфейс.

Недоліком є необхідність мати велику та активну аудиторію для того, щоб монетизувати власні твори, обмежений перелік жанрів.

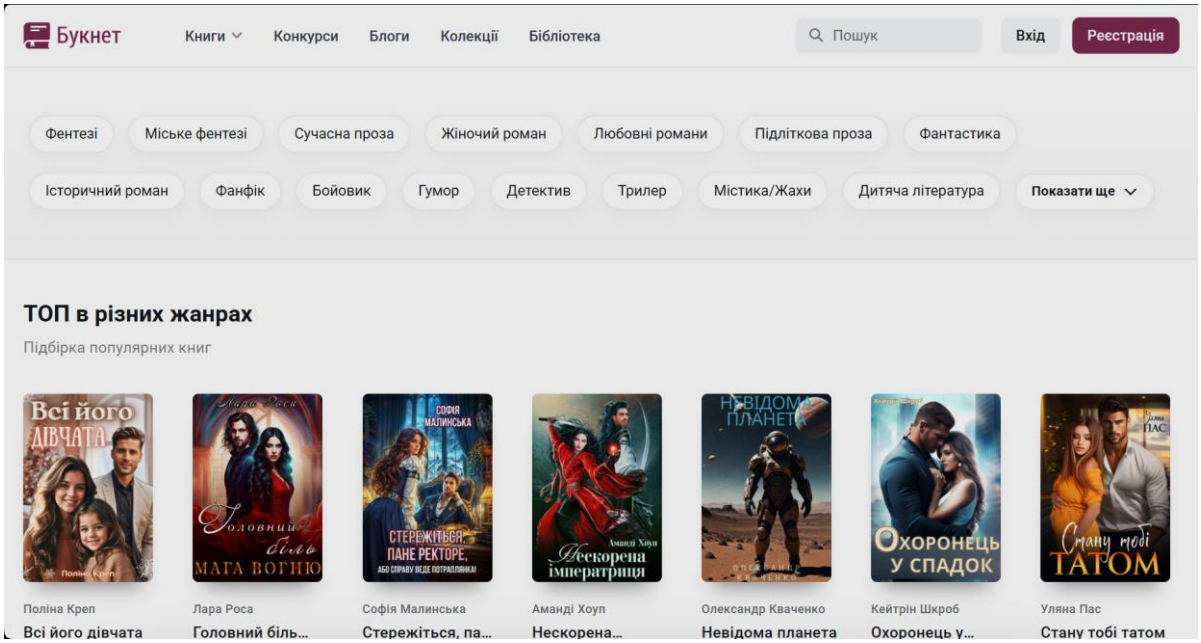

Рисунок 1.5 – Приклад користувацького інтерфейсу платформи «Букнет»

Також популярною платформою є «Medium» [3] (приклад інтерфейсу користувача зображено на рисунку 1.6). Це зарубіжна платформа, яка надає можливість написати та опублікувати власний твір, також має можливість монетизації власної творчості. Платформа має окремий блог для починаючих авторів, де розміщаються статті, які можуть допомогти для опублікування першого твору.

Перевагами є простий інтерфейс та можливість публікувати твори у вільному доступі безкоштовно.

Недоліками є можливість монетизації власних творів доступна лише на преміум акаунті та більшість функцій є платними.

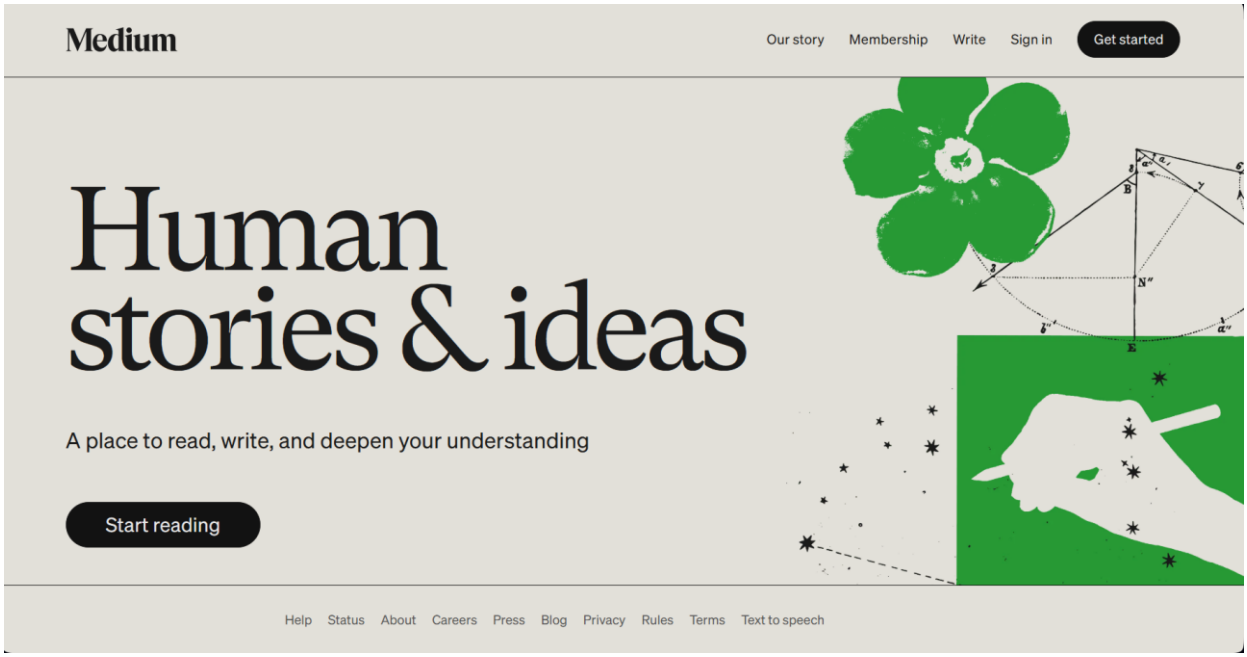

Рисунок 1.6 – Приклад інтерфейсу користувача платформи «Medium»

Отже, мета даної розробки web-платформи для підтримки починаючих авторів є зробити платформою більш зручною та усунути недоліки, які присутні у платформах аналогах, а саме: зручний пошук на сайті, доступність інтерфейсу, велика кількість жанрів та безкоштовний доступ до усіх функцій.

1.4. Порівняльний аналіз сучасних інструментальних засобів розробки колективних web-платформ

Під час розробки сучасних web-платформ слід звернути увагу на усі існуючі технології, які можна застосувати для подальшої розробки та обрати ту, що відповідатиме запиту. Платформа має бути зручною, підтримувати одночасне використання великою кількістю користувачів, зберігати велику кількість даних про користувачів та твори, розміщені на сайті, а також має керувати доступом до акаунтів та безпекою.

Наразі є три зручні та сучасні інструменти для розробки клієнтської частини: React.js, Vue.js та Angular. У таблиці 1.1 наведено переваги та недоліки кожної з технологій.

Таблиця 1.1 – Інструменти для розробки клієнтської частини платформи

Технологія	Переваги	Недоліки
React.js	Компонентна структура, розгалужена екосистема. Підтримується Meta	Занадто складний для новачків
Vue.js	Простота, проста та зрозуміла документація, підходить для новачків	Менш популярний, ніж React.js
Angular	Є повноцінним фреймворком, TypeScript	Занадто складний для новачків, багато функцій

Популярними інструментами для розробки серверної частини вважаються: Node.js, Django, PHP. У таблиці 1.2 наведено основні переваги та недоліки технологій.

Таблиця 1.2 – Інструменти для розробки серверної частини платформи

Технологія	Переваги	Недоліки
Node.js	JavaScript на сервері, асинхронність, легкість масштабування	Не підходить для важких задач
Django	Легко розібратися новачкам	Менш гнучкий у високонавантажених системах
PHP	Гарна структура, зручний та зрозумілий	Вважається менш сучасним

Платформа з великою кількістю користувачів та даних має використовувати сучасну та багатофункціональну базу даних. У таблиці 1.3 наведено популярні бази даних для складних систем: PostgreSQL, MySQL (MariaDB), MongoDB.

Таблиця 1.3 – Бази даних

СУБД	Переваги	Недоліки
PostgreSQL	Підтримує складні запити, стабільна	Потребує налаштування та вміння використовувати систему
MySQL (MariaDB)	Популярна, легко знайти хостинг	Менші можливості для аналітики
MongoDB	Гнучка зі структурою даних	Не завжди є зручною для реляційних задач

Також необхідно враховувати інструментальні засоби для розробки колективних web-платформ під час створення, так як дана розробка орієнтована на багатокористувацьке використання. Тому потрібно враховувати й інструменти, що забезпечать користувачам синхронну співпрацю, безпеку та збереження їх даних.

Насамперед дуже важливим є функціонал, який дозволяє кільком користувачам одночасно користуватися платформою не тільки під час спільного редагування твору, а й перегляду творів інших авторів, публікації власних, редагування власного профілю та використання пошуку. Технології, які можна використати для виконання цього завдання це: Socket.IO, Firebase Realtime Database та WebRTC.

Таблиця 1.4 – Технології для підтримки платформи у реальному часі

Технологія	Переваги	Недоліки
Socket.IO	Підходить для чатів та онлайн-редакторів	Складне масштабування, потребує ретельного опрацювання безпеки
Firebase Realtime Database	Простий у налаштування, миттєва синхронізація даних	Залежність від Google
WebRTC	Підходить для відео- або голосового зв’язку між користувачами	Сумісність з браузерами, складна реалізація

Також важливим є керування користувачами платформи: розділення користувачів на ролі, що визначають права доступу; аутентифікація та авторизація користувачів. Для цього можна використати наступні технології: Firebase Auth, JWT, OAuth 2.0.

Таблиця 1.5 – Технології для керування користувачами

Технології	Переваги	Недоліки
Firebase Auth	Простота інтеграції	Обмежена кастомізація, залежність від стороннього сервісу
JWT (JSON Web Token)	Безстрокове зберігання сесії	Неправильне налаштування або зберігання може зробити платформу вразливою
OAuth 2.0	Гнучкість, підтримує enterprise-рішення	Обмежена безкоштовна версія

Отже, для того, щоб платформа працювала коректно, необхідно враховувати завдання платформи та обрати ті технології та інструменти, які повністю підходять для вирішення поставлених задач. Технології мають забезпечити гнучкість у використанні платформи (робота з текстами творів, синхронна співпраця між користувачами), модерація сайту.

1.5 Постановка завдання розробки функціональної web-платформи підтримки авторів літературних творів

На основі проведеного аналізу існуючих аналогів, технологій та інструментів, які можна використати для розробки платформи, а також врахування недоліків конкурентів та бажань цільової аудиторії, а саме авторів, необхідно сформулювати повне завдання для створення функціональної web-платформи, яка буде спрямована на підтримку авторів літературних творів, та більшу увагу приділити саме авторам початківцям.

Метою платформи є створення багатокористувацькї web-платформи, що дозволить авторам розміщувати власні твори, отримувати зворотній зв’язок від читачів, взаємодіяти з ними та вдосконалювати свої тексти і в подальшому надрукувати їх.

Враховуючи все вищесказане було поставлено наступні завдання:

1. Етап планування:

1.1 Розробити математичні моделі кількісної оцінки літературних творів;

1.2 Розробити структурну модель web-платформи;

1.3 Розробити функціональну модель колективної роботи багатокористувацької web-платформи;

1.4 Розробити схеми бази даних системи багатокористувацької web-платформи;

1.5 Розробка блок-схем алгоритмів програмних модулів web-платформи;

1.6 Розробити моделі візуального інтерфейсу web-платформи.

2. Етап розробки платформи:

2.1 Розробити візуальний інтерфейс онлайн-системи web-платформи для авторів літературних творів;

2.2 Розробити базу даних web-платформи;

2.3 Розробити програмні модулі багатокористувацької web-платформи;

2.4 Розробити систему забезпечення авторських прав авторів літературних творів;

2.5 Створити регламент роботи онлайн-системи web-платформи підтримки авторів літературних творів.

2. РОЗРОБКА МАТЕМАТИЧНИХ МОДЕЛЕЙ І АЛГОРИТМІВ ФУНКЦІОНУВАННЯ WEB-ПЛАТФОРМИ ПІДТРИМКИ АВТОРІВ ЛІТЕРАТУРНИХ ТВОРІВ

2.1. Розробка математичних моделей кількісної оцінки літературних творів

Створення математичної моделі кількісної оцінки літературних творів має допомогти зрозуміти адміністраторам та платформі чи твір цікавий і чи слід його рекомендувати більшій кількості читачів. Зазвичай літературні твори неможливо якісно оцінити, тому що читачі оцінюють прочитаний твір на основі своїх вподобань, переживань та захоплень. Виходить, що якість тексту не грає надважливу роль в оцінці написаного автором, а важливим є саме сприйняття читача.

Проте, в умовах web-платформи все ж існує можливість підсумувати загальне враження читачів від написаного твору, щоб дійти до висновку чи твір справді чудово написаний та цікавий. Для цього необхідно заохотити наступні фактори: кількість прочитань твору, кількість позитивних та негативних оцінок, кількість коментарів, середній та максимальний час прочитання твору, кількість збережень твору у «обране» та у збірки та середній рейтинг твору від 1 до 10 (що ґрунтується на позитивних та негативних оцінках).

Математична модель необхідна для того, щоб розрахувати інтегральний показник кількісної оцінки твору, що відображатиме рівень зацікавленості читачів під час та після взаємодії з твором.

Формула (2.1) демонструє математичну модель кількісної оцінки літературних творів.

$$Q = x_1 \frac{N_P}{N_k} + x_2 \frac{N_g - N_n}{N_\rho + 1} + x_3 \frac{N_c}{N_p + 1} + x_4 \frac{T_{с.ч.}}{T_{max}} + x_5 \frac{S}{N_p + 1} + x_6 \frac{R}{10} \qquad (2.1)$$

де Q – зведена кількісна оцінка твору;

X1, X2, X3, X4, X5, X6 – вагові коефіцієнти, що допомагають визначити вплив кожного чинника;

Np – кількість прочитань твору;

Nk – загальна кількість користувачів;

Ng – кількість позитивних оцінок;

Nn – кількість негативних оцінок;

Nc – кількість коментарів до твору;

Tc.ч. – середній час, проведений на сторінці твору;

Tmax – максимальний час, проведений на сторінці твору;

S – кількість збережень твору (у «обране» або у збірки);

R – середній рейтинг твору.

Дана математична модель демонструє свою універсальність та адаптивність до будь-якого жанру та виду твору. Вагові коефіцієнти можуть змінюватись залежно від цільової аудиторії твору та типу контенту (наприклад, якщо це проза, яку складно порівнювати з повноцінною розповіддю на більш ніж 900 сторінок).

Цю математичну модель можна та необхідно використовувати у наступних напрямках:

1. Для формування рейтингу літературних творів;
2. Для визначення, чи варто рекомендувати твір більшій кількості читачів;
3. Відстеження аналітики твору.

Отже, дана математична модель цілком може допомогти відстежувати популярність творів, взаємодію користувачів з кожним твором (ставити вподобайку, коментувати та зберігати до збірки) та видавати максимально точнй результат про зацікавленість читачів у даному творі на основі дій читача.

2.2. Розробка структурної моделі web-платформи

Структурна модель web-платформи є важливою частиною у побудові структури сайту. На ній зображено усі компоненти, які входять до структури сайту та показано як один компонент взаємодіє з іншим. Також дана модель дозволяє побачити всю картину розробки цілком, що можна безпечно додати чи що необхідно прибрати для полегшення та покращення розробки та майбутнього функціонування платформи.

На рисунку 2.1 зображено структурну модель web-платформи підтримки авторів літературних творів.

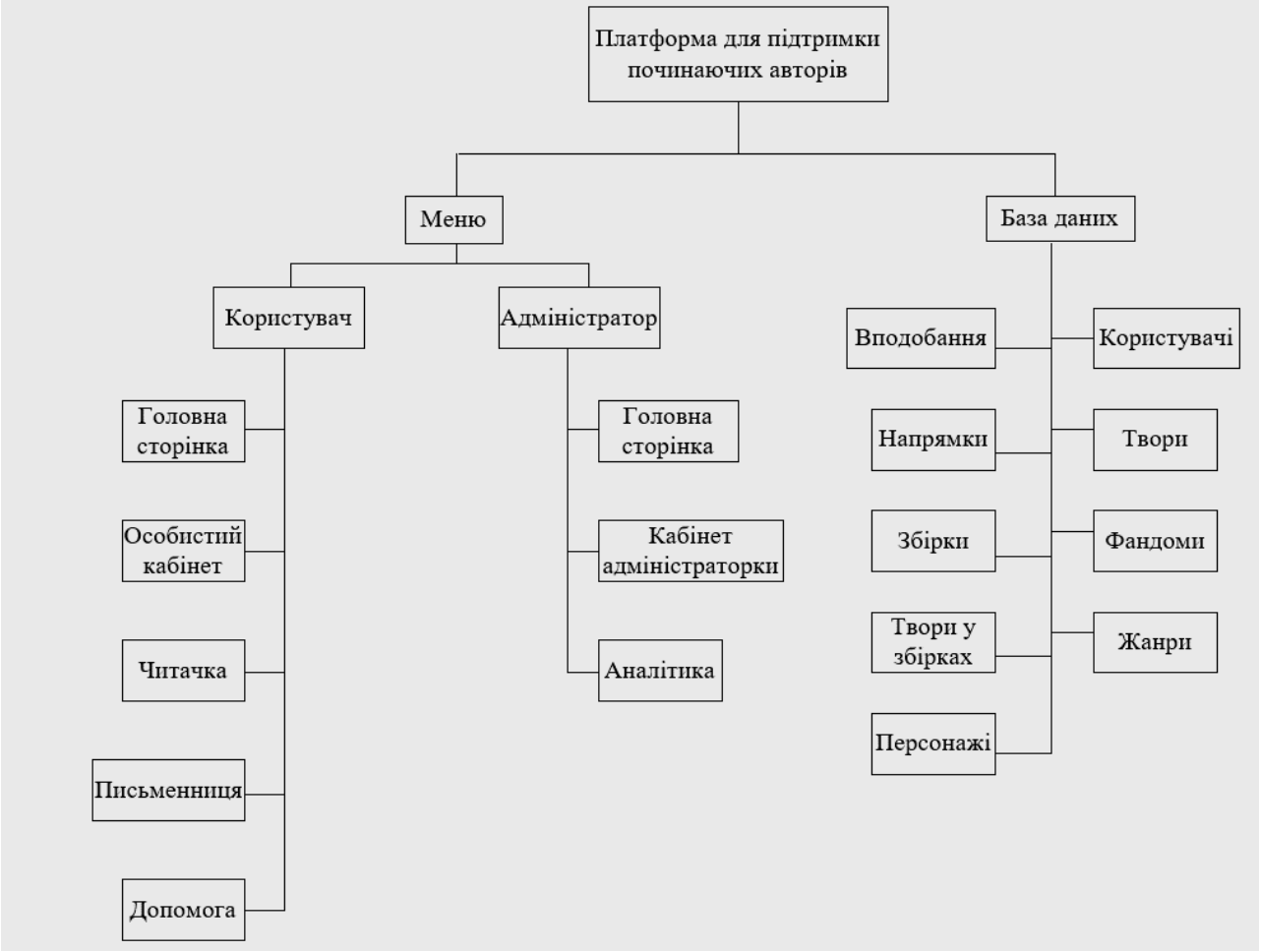

Рисунок 2.1 – Структурна модель web-платформи підтримки авторів літературних творів

Отже, структурна модель є основної для реалізації майбутньої платформи та її подальшого розвитку. Вона забезпечує чітке розмежування компонентів, що дозволяє зручно масштабувати систему.

2.3. Розробка функціональної моделі колективної роботи багатокористувацької web-платформи

Колективна робота кількох авторів одночасно над одним текстом є важливою та пріоритетною задачею для розробки багатокористувацької web-платформи. Багатьом авторам може бути складно самостійно створити свій шедевр і вони об’єднуються з іншими для того, щоб створити твір та в подальшому опублікувати його.

Дана платформа має мати змогу допомогти кільком користувачам, а саме майбутнім авторам, писати текст у реальному часі та бачити усі зміни, які відбулися у тексті одразу, без перезавантаження сторінки. Умі автори мають мати доступ до редагування тексту та змогу редагувати його, коли це робить хтось інший.

Задля цього було розроблено функціональну модель колективної роботи багатокористувацької web-платформи. Перший рівень моделі зображено на Рисунках 2.2 – 2.3.

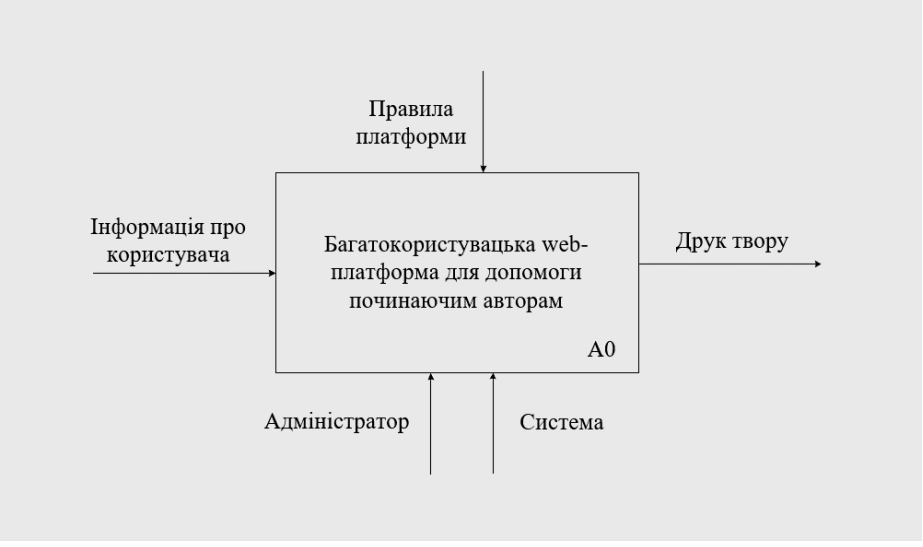

Рисунок 2.2 – Перший рівень функціональної моделі колективної роботи багатокористувацької платформи

Рисунок 2.3 – Перший рівень функціональної моделі колективної роботи багатокористувацької платформи

Другий рівень демонструє процес створення нового твору перед тим, як прийти до спільного написання/редагування твору.

Другий рівень функціональної моделі зображено на рисунку 2.4.

Рисунок 2.4 – Другий рівень функціональної моделі колективної роботи багатокористувацької платформи

Четвертий рівень функціональної моделі зображує процес колективної роботи кількох авторів над твором, а саме: під’єднання до сторінки редагування твору, одночасне редагування твору кількома користувачами та збереження змін після успішного редагування. Після цього будь-який з цих користувачів також може редагувати твір окремо (самотужки).

Четвертий рівень функціональної моделі зображено на рисунку 2.5.

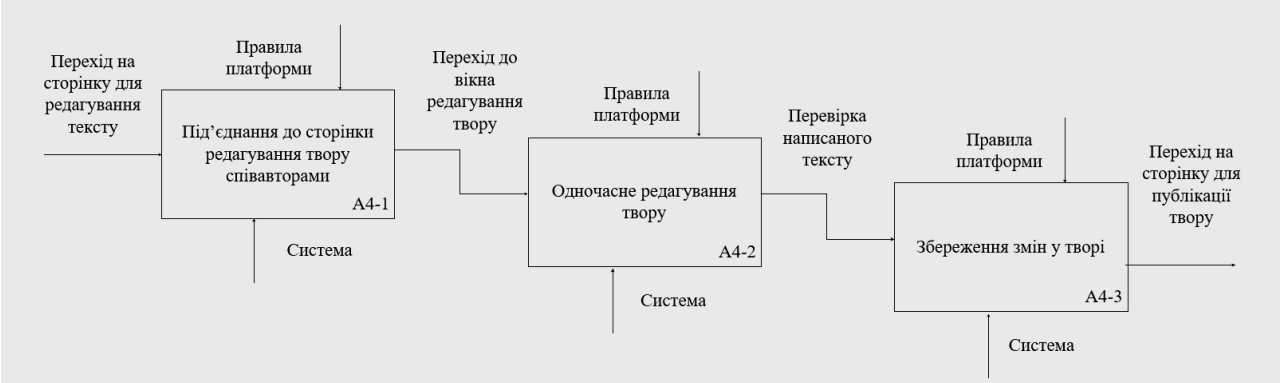

Рисунок 2.5 – Четвертий рівень функціональної моделі колективної роботи багатокористувацької платформи

Також було розроблено Use-Case діаграму задля масштабування функцій доступних для користувачів, які мають різні ролі.

На рисунку 2.6 зображено Use-Case діаграму.

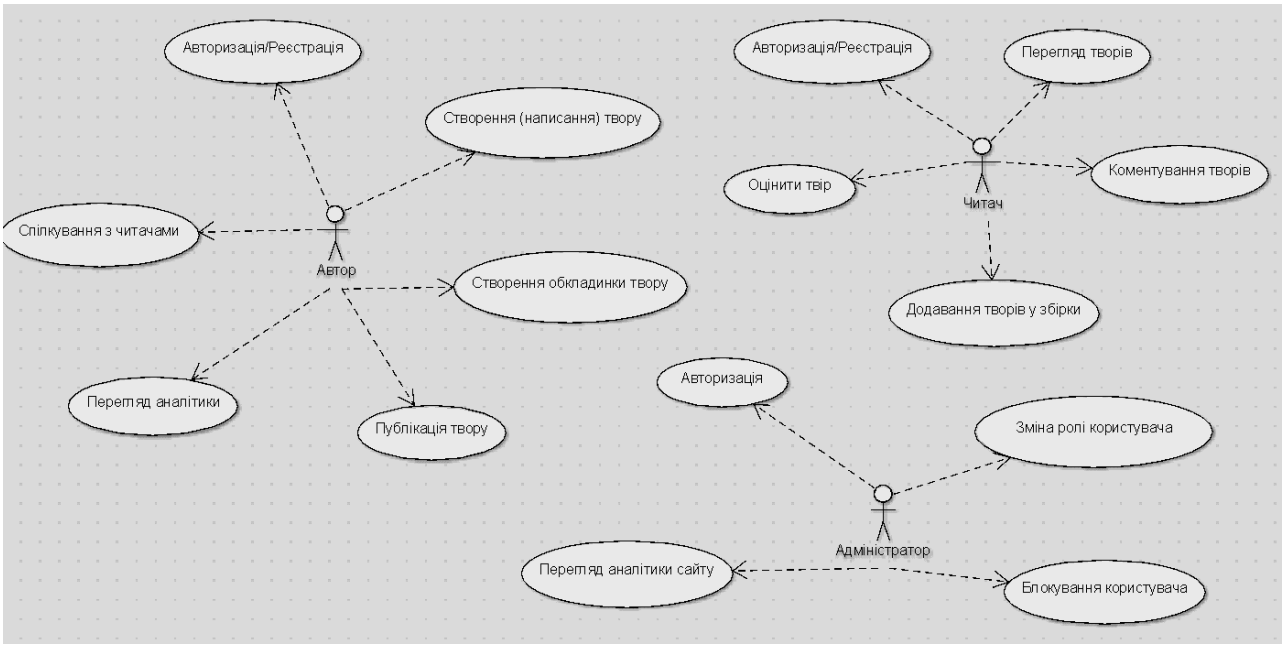

Рисунок 2.6 – Use-Case діаграма багатокористувацької web-платформи

Отже, функціональна модель допомагає масштабувати усі процеси та подивитися алгоритми взаємодії користувачів із платформою, а саме колективну роботу та одночасну взаємодію кількох користувачів не лише з платформою, а й з одним текстом.

2.4. Розробка схеми бази даних системи багатокористувацької web-платформи

Створення схеми бази даних є ключовою для майбутньої розробки функціональної web-платформи для підтримки майбутніх авторів, яка зберігатиме дані не лише про користувачів сайту, а й про їхні твори, вподобання та коментарі.

Майбутня база даних має включати у себе таблиці для збереження даних про твір, кількість вподобань, кількість збірок, у які його було додано, коментарі під кожною частиною. Також вона має зберігати чернетки майбутніх творів та інформацію про них.

База даних для web-платформи має наступний вигляд:

1. Таблиця "Characters":
   - 1.1 "id" primary key, integer;
   - 1.2 "fandom_id" integer;
   - 1.3 "name" character varying 255;
2. Таблиця "Collection_works":
   - 2.1 "id" primary key, integer;
   - 2.2 "collection_id" integer;
   - 2.3 "work_id" integer;
3. Таблиця "Collections":
   - 3.1 "id" primary key, integer;
   - 3.2 "user_id" integer;
   - 3.3 "name" character varying 255;
   - 3.4 "created_at" timestamp without time zone;
4. Таблиця "Direction":
   - 4.1 "id" primary key, integer;
   - 4.2 "name" character varying 50;
5. Таблиця "Fandoms":

5.1 "id" primary key, integer;

5.2 "name" character varying 255;

6. Таблиця "Genres":

6.1 "id" primary key, integer;

6.2 "name" character varying 100;

7. Таблиця "Likes":

7.1 "id" primary key, integer;

7.2 "user_id" integer;

7.3 "work_id" integer;

7.4 "created_at" timestamp without time zone;

8. Таблиця "Users":

8.1 "id" primary key, bigint;

8.2 "login" character varying 25;

8.3 "password" character varying;

8.4 "name" character varying 25;

8.5 "user_type" text;

9. Таблиця "Works":

9.1 "id" primary key, integer;

9.2 "user_id" integer;

9.3 "title" text;

9.4 "fandom" text;

9.5 "pairing" text;

9.6 "characters" text;

9.7 "direction" text;

9.8 "genre" text;

9.9 "description" character varying 255;

9.10 "content" text;

9.11 "created_at" timestamp without time zone;

9.12 "photo" bytea.

На рисунку 2.7 зображено схему бази даних платформи у вигляді ER-діаграми.

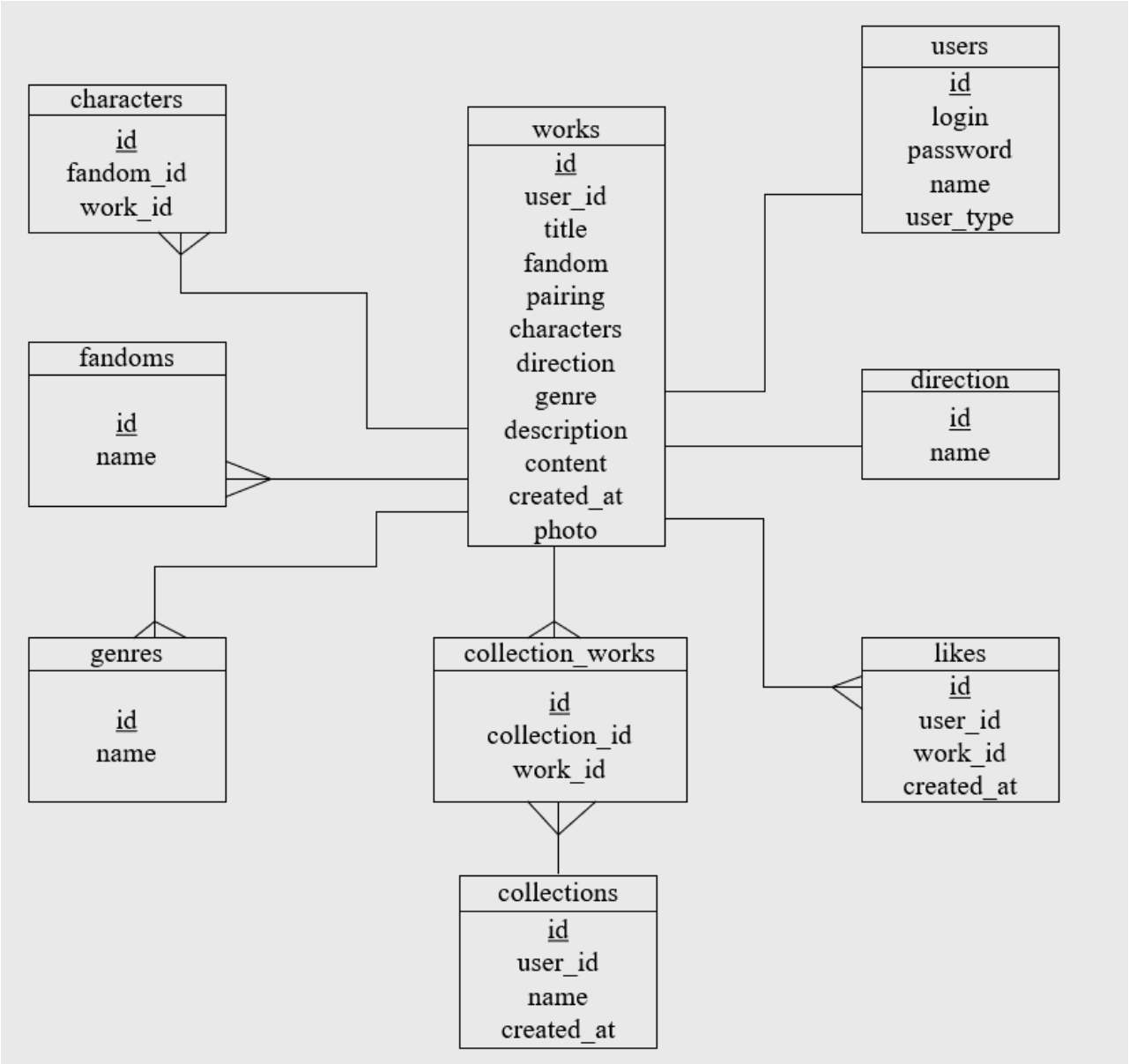

Рисунок 2.7 – Схема бази даних у вигляді ER-діаграми

Отже, база даних багатокористувацької платформи має вміщати у себе велику кількість даних, які пов’язані між собою.

2.5. Розробка блок-схем алгоритмів програмних модулів web-платформи

Блок-схеми алгоритмів програмних модулів дозволяють зрозуміти не лише як працює система, а й як користувач має використовувати функції платформи. Основною функцією будь-якої платформи є реєстрація/авторизація. Алгоритм дій має бути максимально простим задля того, щоб навіть недосвічений користувач міг впоратись із цим завданням.

На рисунку 2.8 зображено блок-схему алгоритму авторизації користувача на сайті.

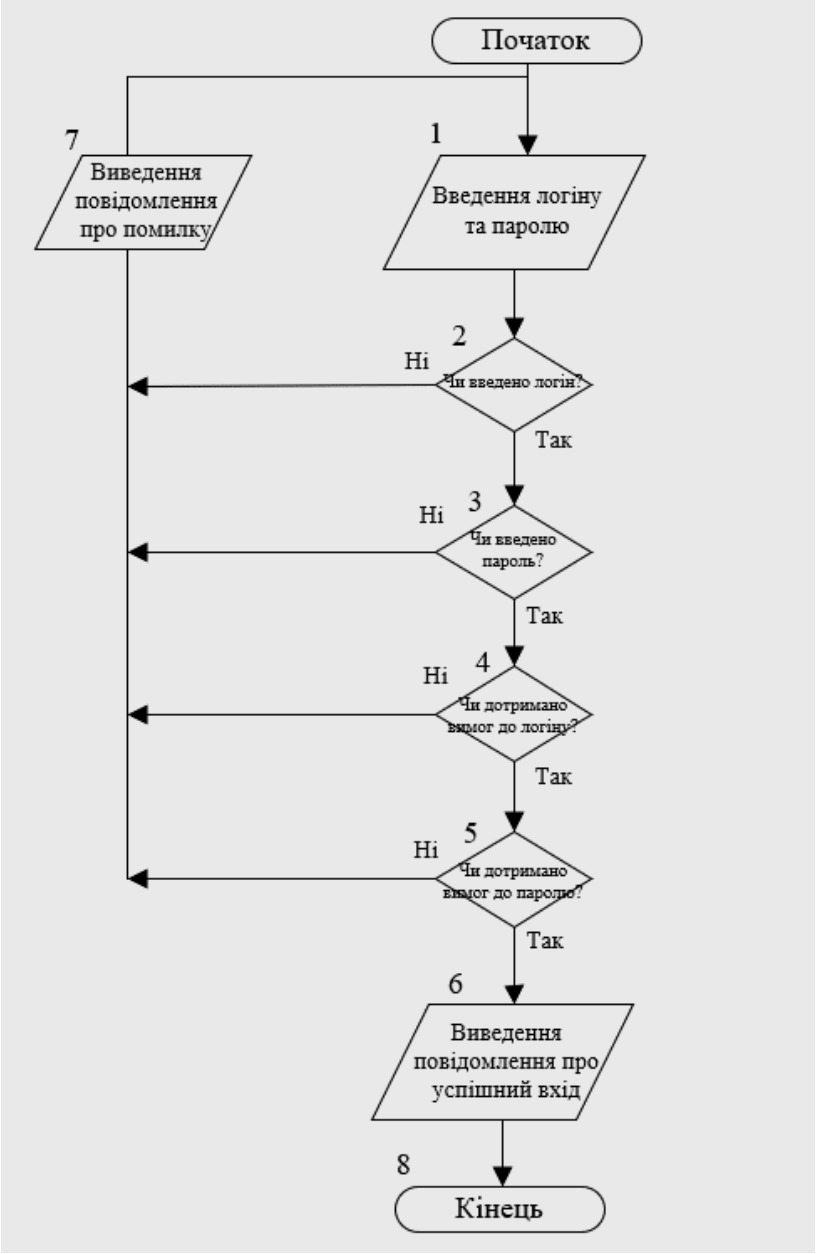

Рисунок 2.8 – Блок-схема алгоритму авторизації користувача на сайті

Наступними важливими алгоритмами для платформи є створення твору автором, перегляд аналітики твору автором та додавання коментаря до твору читачем. Дані алгоритми зображені на Рисунках 2.9 – 2.11.

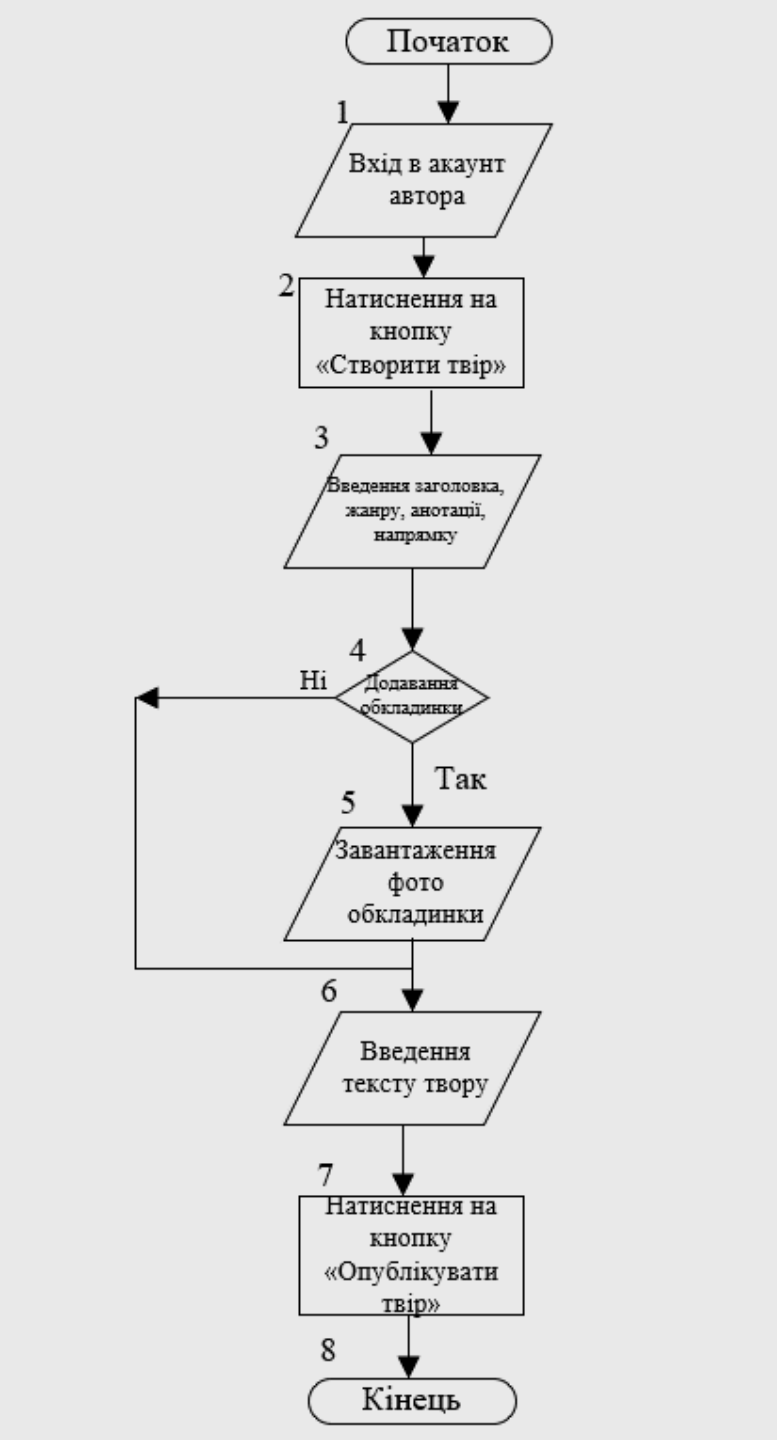

Рисунок 2.9 – Блок-схема алгоритму створення літературного твору на сайті

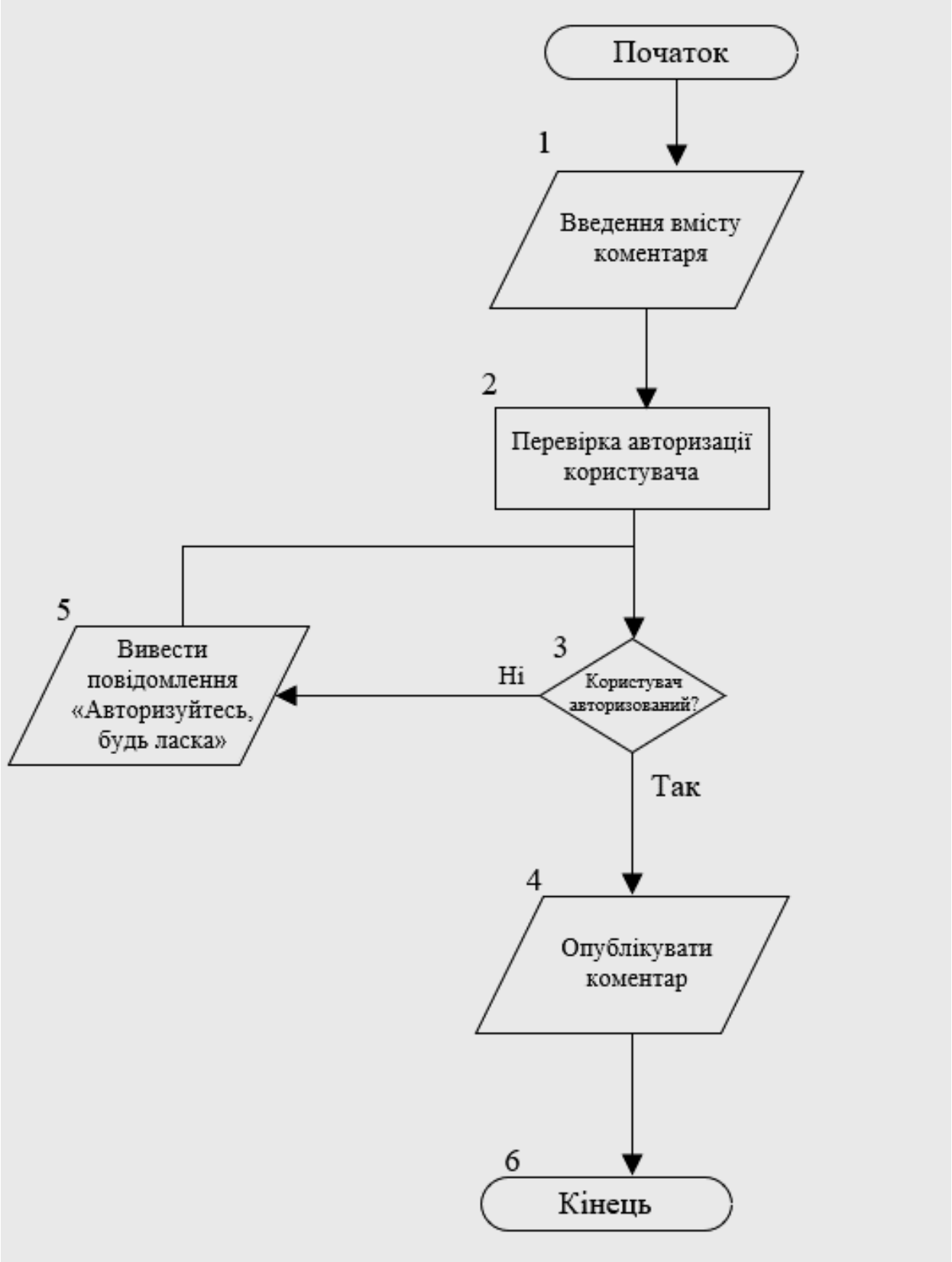

Рисунок 2.10 – Блок-схема алгоритму публікації коментаря до твору читачем

Рисунок 2.11 – Блок-схема алгоритму перегляду аналітики твору автором

Отже, запропоновані блок-схеми зображують основні функції системи та алгоритм за яким користувач може скористатися функцією.

2.6. Розробка моделі візуального інтерфейсу web-платформи

Найважливішим етапом у розробці багатокористувацької web-платформи є розробка моделі візуального інтерфейсу, так як користувач взаємодіє саме з ним.

Інтерфейс має бути інтуїтивно зрозумілим для будь-якої категорії користувачів та мати презентабельний вигляд, о саме візуальна частина сайту може як привернути увагу користувачів, так і спонукати закрити сайт.

На рисунку 2.12 зображено модель головної сторінки сайту.

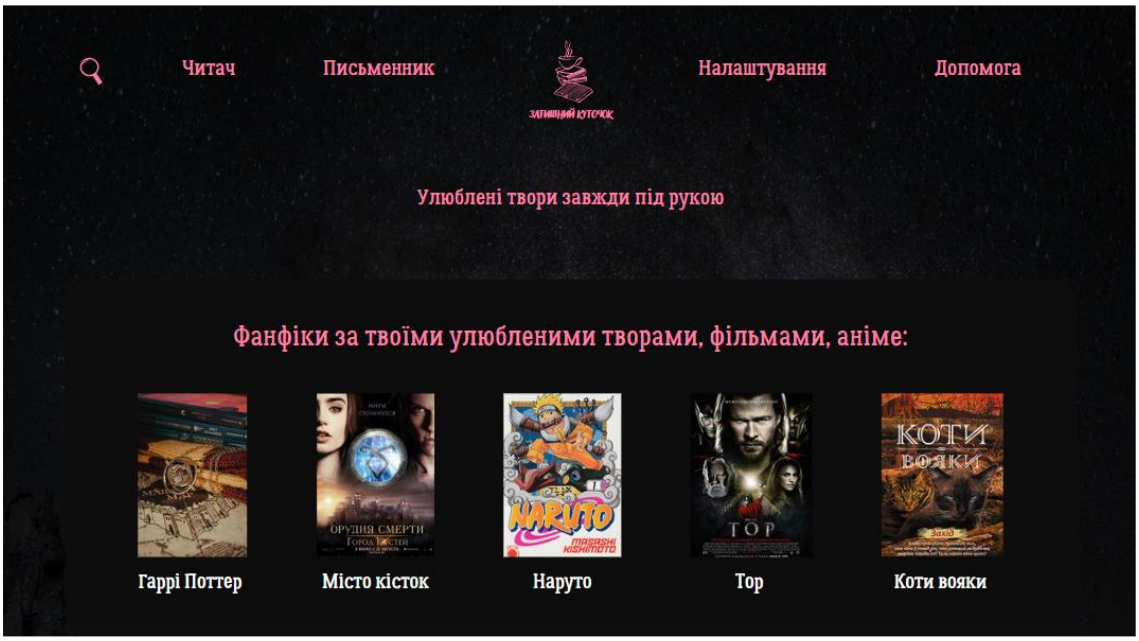

Рисунок 2.12 – Модель головної сторінки багатокористувацької web-платформи

На рисунках 2.13 – 2.14 зображено інші сторінки сайту «Читач» та «Письменник».

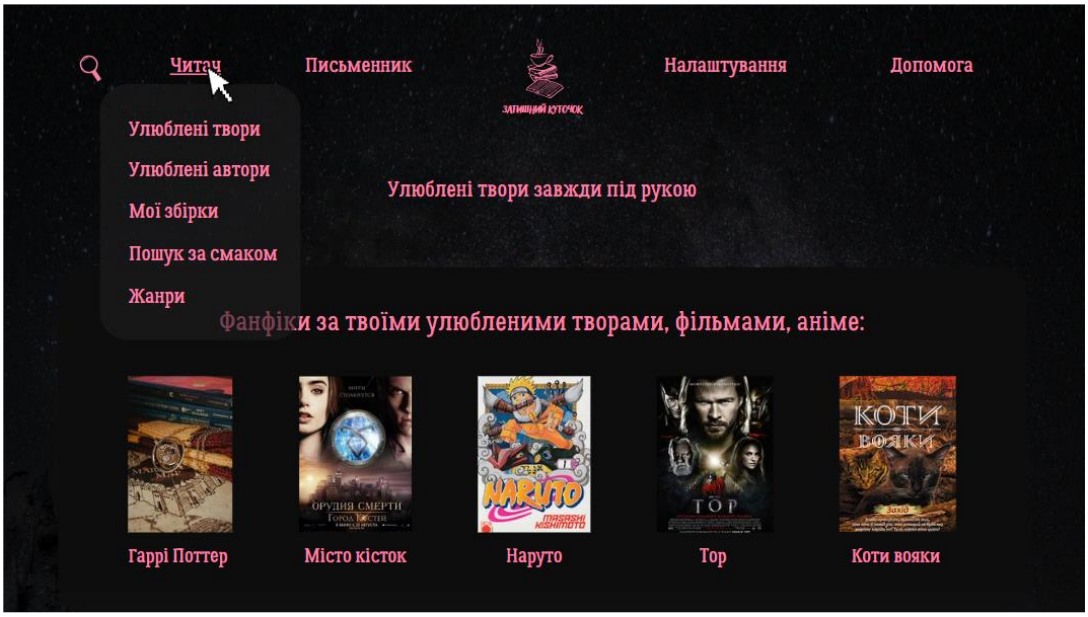

Рисунок 2.13 – Модель головного меню багатокористувацької web-платформи. Пункт «Читач»

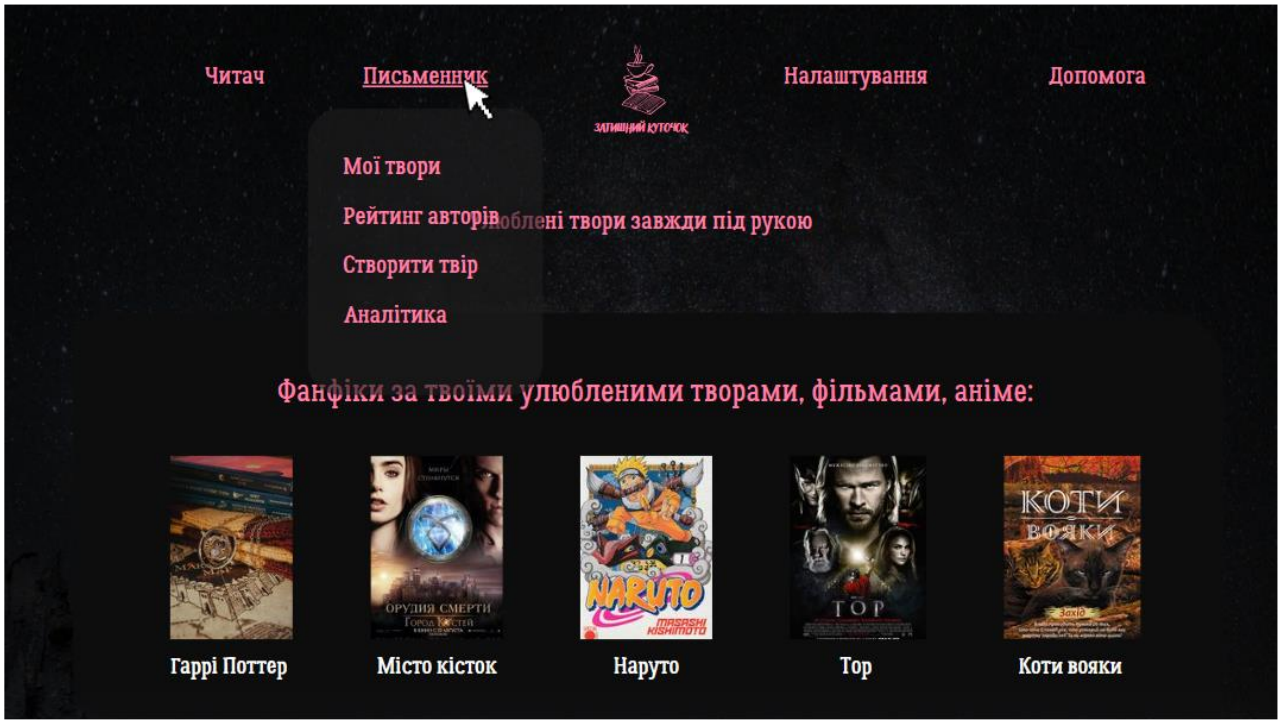

Рисунок 2.14 – Модель головного меню багатокористувацької web-платформи. Пункт «Письменник»

На рисунку 2.15 зображено модель сторінки реєстрації користувача на сайті.

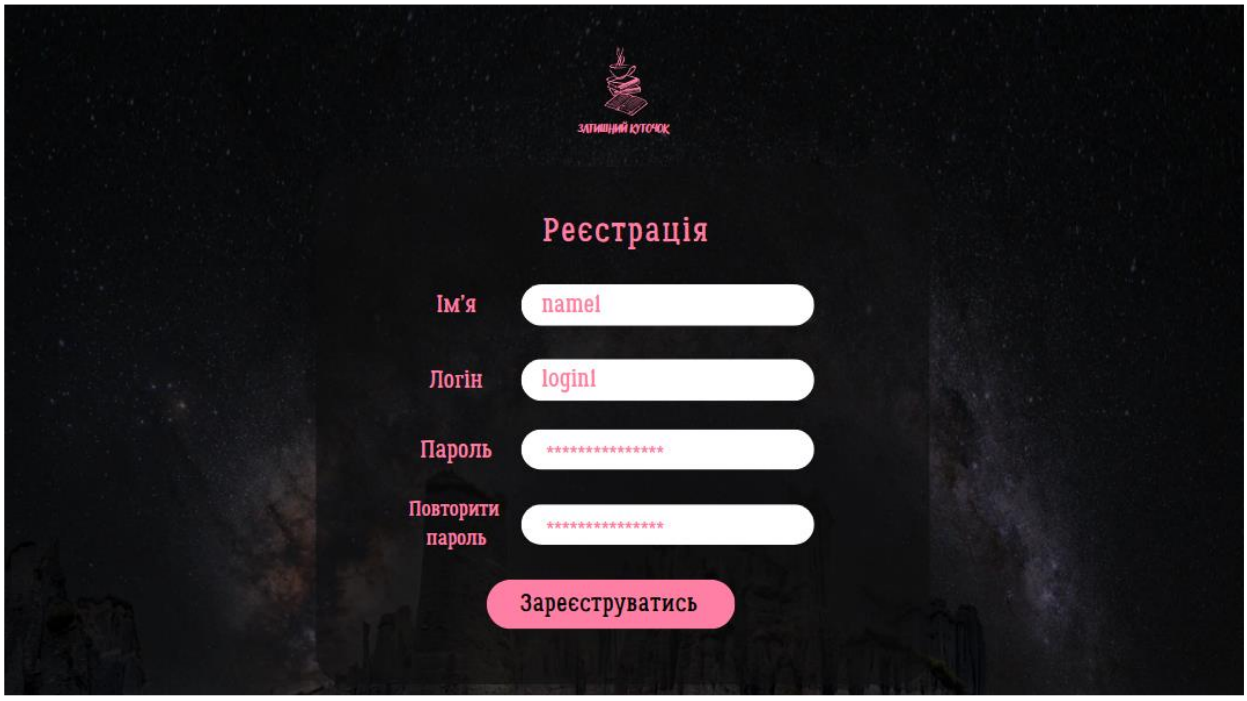

Рисунок 2.15 – Модель сторінки реєстрації користувача багатокористувацької web-платформи.

На рисунку 2.16 зображено модель сторінки авторизації користувача на сайті.

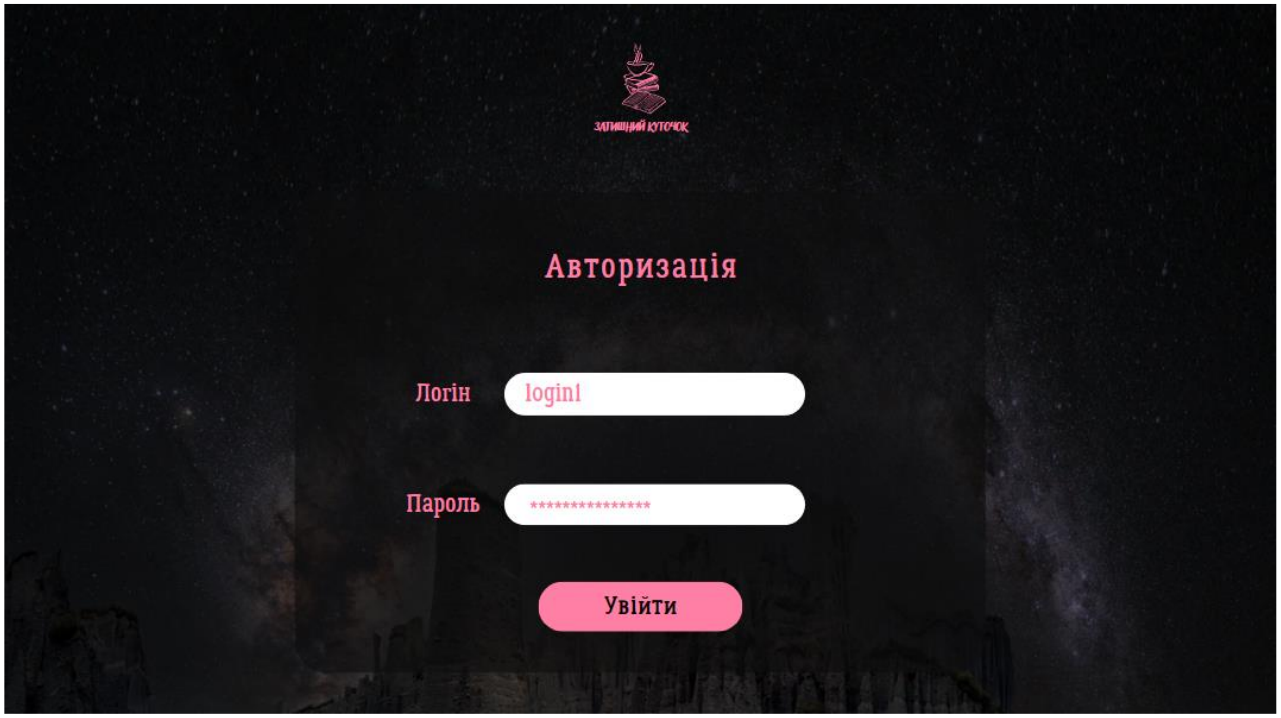

Рисунок 2.16 – Модель сторінки авторизації користувача багатокористувацької web-платформи.

Отже, інтерфейс системи відіграє важливу роль у користуванні платформою та використанню її основних функцій.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ БАГАТОКОРИСТУВАЦЬКОЇ WEB-ПЛАТФОРМИ ПІДТРИМКИ АВТОРІВ ЛІТЕРАТУРНИХ ТВОРІВ

3.1. Розробка візуального інтерфейсу онлайн-системи web-платформи для авторів літературних творів

Під час розробки візуального інтерфейсу платформи допомоги починаючим авторам було обрано наступні технології:

1. HTML5 – для розробки структури сторінки;
2. CSS3 – для поліпшення зовнішнього вигляду сторінок платформи;
3. JavaScript – анімації, спливаючі повідомлення та інші візуальні ефекти, які забезпечують гарний досвід використання сайту;
4. Vue.js – поділ користувачів за типами та відображення того контенту, що призначений для конкретного користувача.

За допомогою прийнятих рішень було розроблено усі сторінки платформи.

Сторінки мають візуальні ефекти, сповіщення після певних дій та під час помилок.

На рисунках 3.1 – 3.4 показано головну сторінку, сторінку реєстрації, авторизації та акаунт користувача.

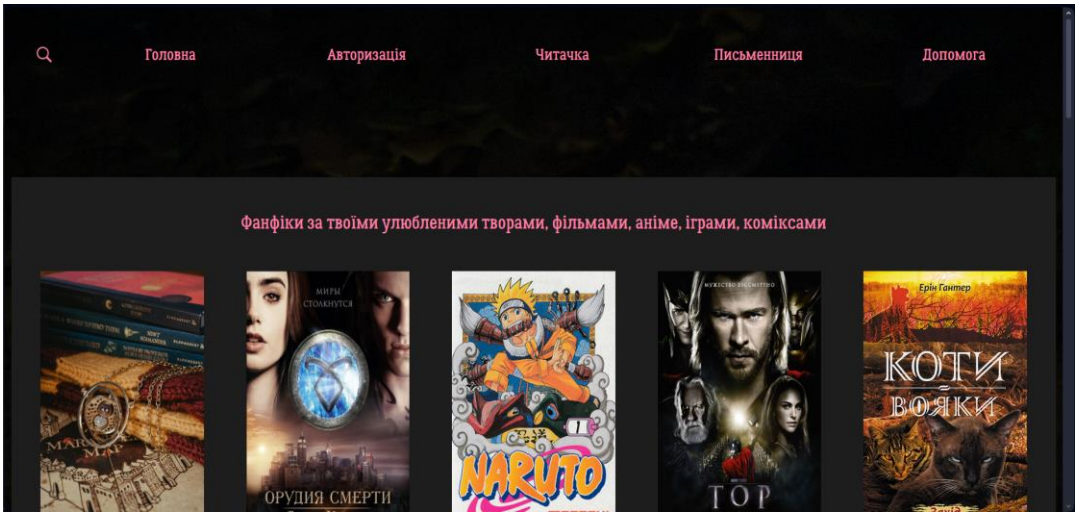

Рисунок 3.1 –Головна сторінка багатокористувацької web-платформи

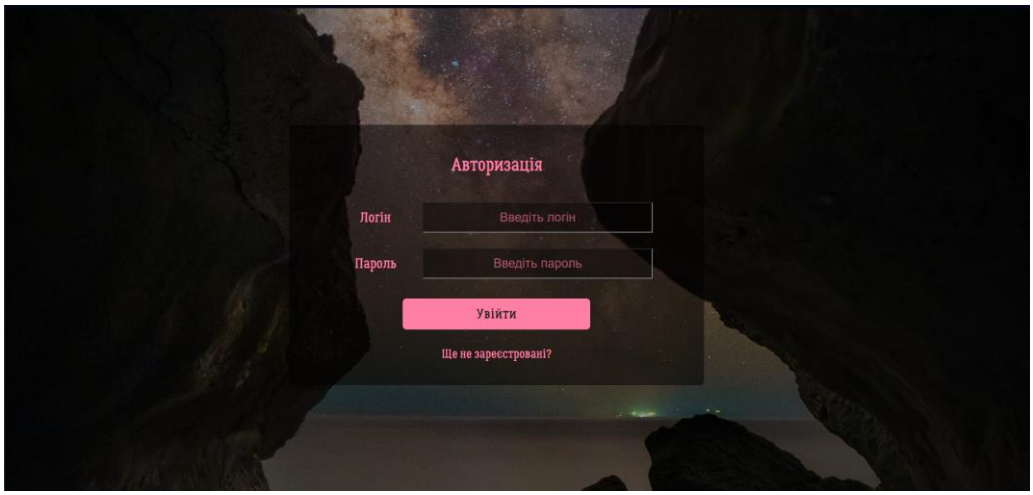

Рисунок 3.2 –Сторінка авторизації багатокористувацької web-платформи

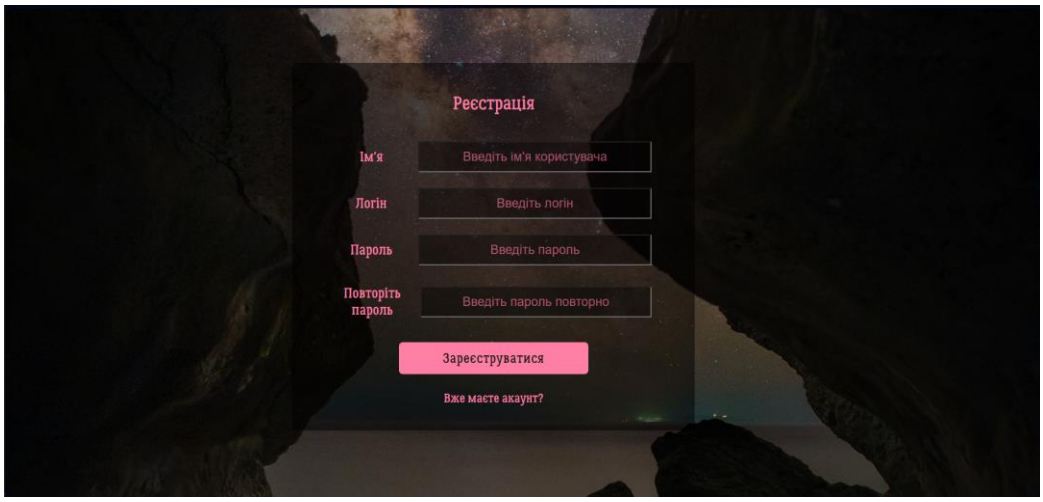

Рисунок 3.3 –Сторінка реєстрації багатокористувацької web-платформи

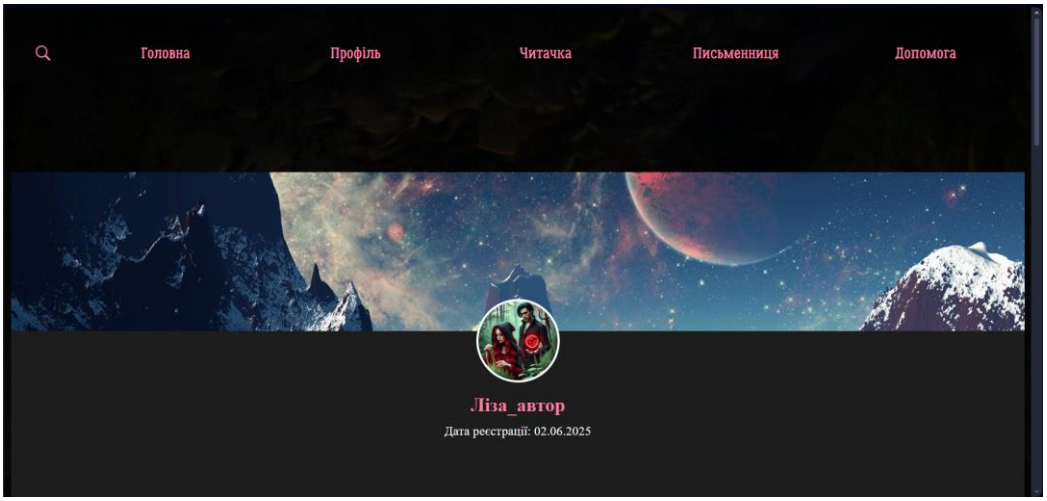

Рисунок 3.4 –Сторінка акаунту користувача багатокористувацької web-платформи

3.2. Розробка бази даних web-платформи

Під час розробки було прийнято рішення використовувати PostgreSQL для керування базою даних.

База даних платформи має наступну структуру:

1. Таблиця "characters":
   1.1 "id" – primary key, integer, not null;
   1.2 "fandom_id" – integer, not null;
   1.3 "name" – character varying 255, not null;
2. Таблиця "collection_works":
   2.1 "id" – primary key, integer, not null;
   2.2 "collection_id" – integer, not null;
   2.3 "work_id" – integer, not null;
3. Таблиця "collections":
   3.1 "id" – primary key, integer, not null;
   3.2 "user_id" – integer, not null;
   3.3 "name" – character varying 255, not null;
   3.4 "created_at" – timestamp without time zone;
4. Таблиця "direction":
   4.1 "id" – primary key, integer, not null;
   4.2 "name" – character varying 50, not null;
5. Таблиця "fandoms":
   5.1 "id" – primary key, integer, not null;
   5.2 "name" – character varying 255, not null;
6. Таблиця "genres":
   6.1 "id" – primary key, integer, not null;
   6.2 "name" – character varying 100, not null;
7. Таблиця "likes":

7.1 "id" – primary key, integer, not null;

7.2 "user_id" – integer, not null;

7.3 "work_id" – integer, not null;

7.4 "created_at" – timestamp without time zone;

8. Таблиця "users":

8.1 "id" – primary key, bigint, not null;

8.2 "login" – character varying 25, not null;

8.3 "password" – character varying, not null;

8.4 "name" – character varying 25, not null;

8.5 "user_type" text, not null;

9. Таблиця "works":

9.1 "id" – primary key, integer, not null;

9.2 "user_id" – integer, not null;

9.3 "title" – text, not null;

9.4 "fandom" – text, not null;

9.5 "pairing" – text, not null;

9.6 "characters" – text, not null;

9.7 "direction" – text, not null;

9.8 "genre" – text, not null;

9.9 "description" – character varying 500, not null;

9.10 "content" – text;

9.11 "created_at" – timestamp without time zone;

9.12 "draft" – integer, not null.

3.3. Розробка програмних модулів багатокористувацької web-платформи

Під час розробки web-платформи було прийнято рішення зменшити кількість повторюваного коду за допомогою вставки готової частини сторінки

(за допомогою функції include) там, де це необхідно та надсилати необхідні для кожної сторінки запити, які можуть бути різними. Такими частинами сторінок є файли: «menu.php» [1], «fanfics.php» [2], «read.php» [3], «details.php» [4], «session.php» [5] та «footer.php» [6].

Було розроблено логіку пошуку інформації по базі даних («search.php» [7]), яка не враховує регістр. Також було розроблено сторінки для створення твору («create_work.php» [8]), редагування твору («update_work.php» [9]), перегляду власних творів («my_works.php» [10]), які поділяються на опубліковані роботи та чернетки. Є можливість додати твір до збірки, переглянути твори у збірці («my_collections.php» [11]), видалити твір зі збірки та видалити саму збірку (окрім збірок «Прочитано» та «Улюблене»).

Було розроблено функціонал для адміністратора, який дозволяє додавати нові фандоми («admin_add.php» [12]) та редагувати їх («admin_change.php» [13]) та керувати користувачами («users.php» [14]) (змінювати тип користувача).

Для розробки програмних модулів було використано мови програмування PHP та JavaScript, які дозволили розробити функціонал, що дозволяє надсилати запити та оновлювати частину сторінки без примусового перезавантаження, що є дуже важливим у випадках, коли у формі є дані, які користувач не хоче втратити.

3.4. Розробка системи забезпечення авторських прав авторів літературних творів

Більшість починаючих авторів перед тим, як викласти свій твір на сайт мають бути впевнені, що ніхто не зможе вкрасти чи використати їхній твір у комерційних цілях без їхньої згоди.

Система забезпечення авторських прав авторів на платформі забезпечена збереженням дати публікації під час створення твору.

3.5. Регламент роботи онлайн-системи web-платформи підтримки авторів літературних творів

Задля допомоги користувачам у використанні платформи було створено сторінку «Допомога» («help.php» [15]),де кожен користувач може знайти відповіді на поширені запитання.

Базовим алгоритмом використання платформи є:

1. Реєстрація;
2. Авторизація;
3. Перегляд головної сторінки;
4. Читання творів;
5. Створення нового твору;
6. Редагування твору;
7. Публікація твору;
8. Завантаження зображень на сторінці акаунту;
9. Редагування розділу «Про себе».

ВИСНОВКИ

Метою роботи була розробка функціональної web-платформи допомоги починаючим авторам.

На початку роботи було проведено аналіз предметної області та прочитано відповідну літературу задля складання цілісної картини. Було проведено аналіз існуючих рішень та виділено недоліки, які необхідно було усунути під час розробки.

Після виявлення недоліків та постановки задач для розробки нової платформи було сформовано план розробки. Після формування плану було розроблено моделі сторінок майбутньої платформи та почато верстку.

Після завершення розробки візуального інтерфейсу платформи, було прийнято рішення почати розробку серверної частини платформи.

Після завершення розробки було отримано готову функціональну web-платформу і розпочато тестування в ході якого було знайдено та успішно вирішено кілька помилок.

Попередня оцінка програмного продукту дуже позитивна.

ПЕРЕЛІК ПОСИЛАНЬ

1. Сайт «Wattpad» [Електронний ресурс] / Режим доступу URL: https://www.wattpad.com Дата: 13.05.25

2. Сайт «Booknet» [Електронний ресурс] / Режим доступу URL: https://booknet.ua Дата: 13.05.25

3. Сайт «Medium» [Електронний ресурс] / Режим доступу URL: https://medium.com Дата: 13.05.25

4. Сайт «W3schoolsUa» [Електронний ресурс] / Режим доступу URL: https://w3schoolsua.github.io/html/index.html#gsc.tab=0 Дата: 03.06.25

5. Сайт «CSS.IN.UA» [Електронний ресурс] / Режим доступу URL: https://css.in.ua Дата: 03.06.25

6. Сайт «JavaScriptInfo» [Електронний ресурс] / Режим доступу URL: https://uk.javascript.info Дата: 03.06.25

7. Сайт «FormindEd» [Електронний ресурс] / Режим доступу URL: https://foxminded.ua/vue-js/ Дата: 03.06.25

8. Сайт «Canva» [Електронний ресурс] / Режим доступу URL: https://www.canva.com Дата: 03.06.25

9. Tom Butler PHP & MySQL: Novice to Ninja / Tom Butler, Kevin Yank – 2017. – 475 с.

10. Robin Nixon. Learning PHP, MySQL, JavaScript, CSS & HTML5 / Robin Nixon – 2014. – 729 с.

11. Richard Blum. PHP, MySQL & JavaScript All-in-One For Dummies / Richard Blum, John Wiley – 2018. – 795

ДОДАТКИ

1. Код сторінки «menu.php»:

```
<?php
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
include "session.php";
global $user_id;
$fav_collection_id = null;
if ($user_id) {
    try {
        $pdo = new PDO($dsn, $db_username, $db_password, [
            PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
        ]);
        $stmt = $pdo->prepare("SELECT id FROM collections WHERE user_id =
:user_id AND name = 'Улюблене' LIMIT 1");
        $stmt->execute(['user_id' => $user_id]);
        $fav_collection_id = $stmt->fetchColumn();
    } catch (PDOException $e) {
    }
}
?>
<link rel="stylesheet" href="../css/menu.css">
<div id="searchModal" class="search-modal">
    <div class="search-modal-content">
        <input type="text" id="searchInput" placeholder="Введіть пошуковий
запит...">
```

```
            <div id="searchResults" class="search-results"></div>
        </div>
    </div>
    <div class="menu">
        <div class=" menu-item">
            <img id="search" class="search" src="../img/search.png">
        </div>
        <div class="menu1 menu-item" onclick="navigateTo('main.php')">
            Головна
        </div>
        <?php if ($_SESSION['user_type'] === 'user'): ?>
            <div class="menu1 menu-item">Профіль
                <div class="submenu">
                    <div class="menu1 menu-item" onclick="navigateTo('account.php')">Мій профіль</div>
                    <div class="menu1 menu-item" onclick="navigateTo('logout.php')">Вийти</div>
                </div>
            </div>
        <?php elseif ($_SESSION['user_type'] === 'admin'): ?>
            <!-- Якщо користувач адміністратор -->
            <div class="menu1 menu-item">Адміністраторка
                <div class="submenu">
                    <div class="submenu-item" onclick="navigateTo('admin_add.php')">Додати фандом</div>
                    <div class="submenu-item" onclick="navigateTo('admin_change.php')">Редагувати фандом</div>
                    <div class="submenu-item" onclick="navigateTo('users.php')">Користувачі</div>
```

```
                <div class="menu1 menu-item" onclick="navigateTo('logout.php')">Вийти</div>
            </div>
        </div>
        <?php else: ?>
            <div class="menu1 menu-item" onclick="navigateTo('login_form.php')">Авторизація</div>
        <?php endif; ?>
        <div class="menu1 menu-item">Читачка
            <div class="submenu">
                <div class="submenu-item"
                    onclick="<?= $fav_collection_id
                        ?
"navigateTo('collection_works.php?collection_id=$fav_collection_id')"
                        : "alert('Збірка не знайдена')" ?>">
                    Улюблені твори
                </div>
                <div class="submenu-item" onclick="navigateTo('favourite_authors.php')">Улюблені авторки</a></div>
                <div class="submenu-item" onclick="navigateTo('my_collections.php')">Мої збірки</a></div>
            </div>
        </div>
        <div class="menu1 menu-item">
            Письменниця
            <div class="submenu">
                <div class="submenu-item" onclick="navigateTo('my_works.php')">Мої твори</div>
```

```
            <div class="submenu-item" onclick="navigateTo('create_work.php')">Створити твір</a></div>
        </div>
    </div>
    <div class="menu1 menu-item" onclick="navigateTo('help.php')">Допомога</div>
</div>
<script src="../js/menu.js"></script>
```

2. Код сторінки «fanfics.php»:

```
<?php if (!isset($works)) {
    echo '<p style="color:white;">Помилка: не передано список творів.</p>';
    return;
}
global $works_in;
global $user_id;
global $current_collection_id;
global $collection;
global $is_my_works;
include "fanf_menu.php";
?>
<link rel="stylesheet" href="../css/fanfics.css">
<div id="collectionModal" class="modal" style="display:none;">
    <div class="modal-content">
        <span class="close" onclick="closeModal()">&times;</span>
        <h3>Оберіть збірку:</h3>
        <div id="collection-list"></div>
        <hr>
```

```
            <h4>Нова збірка:</h4>
            <label>Назва нової збірки:</label>
            <input type="text" id="newCollectionName">
            <button onclick="createAndAddCollection()">Створити і додати</button>
        </div>
    </div>
    <div class="fanfics">
        <div class="title">
            <p><?= $block_title ?? 'Список творів' ?></p>
        </div>
        <?php if (empty($works)): ?>
            <p class="without">Немає творів для відображення.</p>
        <?php else: ?>
            <?php foreach ($works as $work): ?>
                <?php if ((!isset($work['draft']) || $work['draft'] == 0) && (!$is_my_works)) continue; ?>
                <div class="fanfic">

                    <?php
                    global $dsn;
                    global $db_username;
                    global $db_password;
                    global $author;
                    $author_id = $work['user_id'];
                    $likes_map = [];
                    $user_likes = [];
                    try {
                        $pdo = new PDO($dsn, $db_username, $db_password, [
```

```
                PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
            ]);
            $stmt = $pdo->prepare("SELECT * FROM users WHERE id = :id");
            $stmt->execute(['id' => $author_id]);
            $author = $stmt->fetch();
            // Отримати імена персонажів твору
            $character_names = [];
            $stmt = $pdo->prepare("
        SELECT c.name
        FROM characters c
        INNER JOIN work_characters wc ON c.id = wc.character_id
        WHERE wc.work_id = ?
    ");
            $stmt->execute([$work['id']]);
            $character_names = $stmt->fetchAll(PDO::FETCH_COLUMN);
            $characters_str = $character_names ? implode(', ', $character_names) : '-';
            $work_ids = array_column($works, 'id');
            $placeholders = implode(',', array_fill(0, count($work_ids), '?'));
            // Кількість лайків
            $stmt = $pdo->prepare("SELECT work_id, COUNT(*) as count FROM likes WHERE work_id IN ($placeholders) GROUP BY work_id");
            $stmt->execute($work_ids);
            foreach ($stmt->fetchAll(PDO::FETCH_ASSOC) as $row) {
                $likes_map[$row['work_id']] = $row['count'];
            }
            // Чи користувач лайкнув
            if ($user_id) {
```

```
                $stmt = $pdo->prepare("SELECT work_id FROM likes WHERE user_id = ? AND work_id IN ($placeholders)");
                $stmt->execute(array_merge([$user_id], $work_ids));
                foreach ($stmt->fetchAll(PDO::FETCH_ASSOC) as $row) {
                    $user_likes[$row['work_id']] = true;
                }
            }
        } catch (PDOException $e) {
            echo "Помилка бази даних: " . $e->getMessage();
            exit;
        }
        ?>
        <div class="text">
            <p class="title-fanfic">
                <a class="title-fanfic" href="read.php?id=<?= $work['id'] ?>">
                    <?= htmlspecialchars($work['title']) ?>
                </a>
            </p>
            <?php
            $like_count = $likes_map[$work['id']] ?? 0;
            $user_liked = isset($user_likes[$work['id']]);
            ?>
            <div class="like-container">
                <button class="like-button" onclick="toggleLike(this, <?= $work['id'] ?>)"
                        data-liked="<?= $user_liked ? '1' : '0' ?>">
                    <span class="heart"><?= $user_liked ? '❤️' : '♡' ?></span>
                    <span class="like-count"><?= $like_count ?></span>
                </button>
```

```
                </div>
                <p><a style="color: #ff7fa5">Автор:</a> <a
                        href="guest_profile.php?id=<?=  $work['user_id']  ?>"><?=
htmlspecialchars($author['name']) ?> </a>
                </p>
                <p><a          style="color:          #ff7fa5">Фандом:</a>          <?=
htmlspecialchars($work['fandom']) ?></p>
                <p><a          style="color:          #ff7fa5">Пейринг:</a>          <?=
htmlspecialchars($work['pairing']) ?></p>
                <p><a          style="color:          #ff7fa5">Персонажі:</a>          <?=
htmlspecialchars($characters_str) ?></p>
                <p><a          style="color:          #ff7fa5">Напрямок:</a>          <?=
htmlspecialchars($work['direction']) ?></p>
                <p><a          style="color:          #ff7fa5">Жанр:</a>          <?=
htmlspecialchars($work['genre']) ?></p>
                <p><a          style="color:          #ff7fa5">Опис:</a>          <?=
htmlspecialchars($work['description']) ?></p>
                <p><a style="color: #ff7fa5">Дата
                        публікації:</a>                <?=                date("d.m.Y",
strtotime($work['created_at'])) ?></p>
            </div>
            <div class="menu-container">
                <button                                        class="menu-button"
onclick="toggleMenu(this)">☰</button>
                <div class="dropdown-menu">
                    <div       class="menu-item1"       onclick="addToCollection(<?=
$work['id'] ?>)">Додати до збірки</div>
                    <?php
                    $isRead = in_array($work['id'], $works_in['прочитано']);
```

```
                    $isFav = in_array($work['id'], $works_in['улюблене']);
                    ?>
                    <div class="menu-item1"
                        onclick="<?= $isRead
                            ?                "removeFromNamedCollection({$work['id']},
'Прочитано')"
                            :    "addToNamedCollection({$work['id']},    'Прочитано')"
?>">
                        <?= $isRead ? "Видалити з Прочитаного" : "Прочитано" ?>
                    </div>

                    <div class="menu-item1"
                        onclick="<?= $isFav
                            ?                "removeFromNamedCollection({$work['id']},
'Улюблене')"
                            :    "addToNamedCollection({$work['id']},    'Улюблене')"
?>">
                        <?= $isFav ? "Видалити з Улюбленого" : "Улюблене" ?>
                    </div>
                </div>
            </div>
            <?php if (isset($show_remove_button) && $show_remove_button
&& isset($current_collection_name)): ?>
                <div class="remove-from-collection">
                    <button onclick="removeFromNamedCollection(<?= $work['id']
?>, '<?= htmlspecialchars($current_collection_name, ENT_QUOTES) ?>')">
                        Видалити зі збірки
                    </button>
                </div>
```

```
                <?php endif; ?>
                <?php
                // Якщо користувач є власником твору, показати кнопки
                if (isset($_SESSION['user_id']) && $_SESSION['user_id'] == $work['user_id']):
                    if (isset($is_my_works)):
                    ?>
                    <div class="owner-actions" style="margin-top: 10px;">
                        <form method="post" action="delete_work.php" id="delete-form-<?= $work['id'] ?>"
                              style="display:inline;">
                            <input type="hidden" name="work_id" value="<?= $work['id'] ?>">
                            <button type="button" class="delete-button" onclick="confirmDelete(<?= $work['id'] ?>)">
                                Видалити твір
                            </button>
                        </form>
                        <a href="update_work.php?id=<?= $work['id'] ?>">
                            <button class="edit-button">Редагувати твір</button>
                        </a>
                        <a href="change_header_work.php?id=<?= $work['id'] ?>">
                            <button class="edit-button">Редагувати шапку твору</button>
                        </a>
                    </div>
                <?php endif; ?>
            <?php endif; ?>
        </div>
```

```
        <?php endforeach; ?>
    <?php endif; ?>
</div>
<script src="../js/fanfics.js"></script>
```

3. Код сторінки «read.php»:

```
<?php
include "session.php";
global $user_id;
require_once __DIR__ . '/../libs/htmlpurifier-4.15.0/library/HTMLPurifier.auto.php';
if (!isset($_GET['id']) || !is_numeric($_GET['id'])) {
    echo "Невірний ID фанфіка.";
    exit;
}
$id = (int)$_GET['id'];
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    // Отримання фанфіка по id
    $stmt = $pdo->prepare("SELECT * FROM works WHERE id = :id");
    $stmt->execute(['id' => $id]);
    $work = $stmt->fetch();
    $author_id = $work['user_id'];
```

```
        if (!$work) {
            echo "Фанфік не знайдено.";
            exit;
        }
        if ($work['draft'] == 0 && $user_id != $work['user_id']) {
            echo "Ви не маєте доступу";
            exit;
        }
        $stmt = $pdo->prepare("SELECT * FROM users WHERE id = :id");
        $stmt->execute(['id' => $author_id]);
        $author = $stmt->fetch();
    } catch (PDOException $e) {
        echo "Помилка бази даних: " . $e->getMessage();
        exit;
    }
    function purify_html($dirty_html) {
        static $purifier = null;
        if ($purifier === null) {
            $config = HTMLPurifier_Config::createDefault();
            $config->set('HTML.Allowed',
'p,b,i,u,div,span,br,strong,em,s,sub,sup,ul,ol,li,blockquote,hr,pre,code');
            $config->set('CSS.AllowedProperties', ['text-align', 'color', 'font-weight',
'font-style', 'text-decoration']);
            $config->set('HTML.AllowedAttributes', 'style');
            $purifier = new HTMLPurifier($config);
        }
        return $purifier->purify($dirty_html);
    }
    // Отримати імена персонажів твору
```

```
$character_names = [];
$stmt = $pdo->prepare("
    SELECT c.name
    FROM characters c
    INNER JOIN work_characters wc ON c.id = wc.character_id
    WHERE wc.work_id = ?
");
$stmt->execute([$work['id']]);
$character_names = $stmt->fetchAll(PDO::FETCH_COLUMN);
$characters_str = $character_names ? implode(', ', $character_names) : '-';
?>
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title><?= htmlspecialchars($work['title']) ?></title>
    <link rel="stylesheet" href="../css/base.css">
    <link rel="stylesheet" href="../css/read.css">
</head>
<body>
<div class="container">
    <?php include "menu.php" ?>
    <div class="content">
        <h1>"<?= htmlspecialchars($work['title']) ?>"</h1>
        <div class="fanf">

            <p><a style="color: #ff7fa5; font-size: 19px; font-family: Lumberjack;
width: auto"> Автор:</a> <a
```

```
                    href="guest_profile.php?id=<?=  $work['user_id']  ?>"><?=
htmlspecialchars($author['name']) ?> </a>
                </p>
                <p><a style="color: #ff7fa5; font-size: 19px; font-family: Lumberjack;
width: auto"> Фандом:</a> <?= htmlspecialchars($work['fandom']) ?></p>
                <p><a style="color: #ff7fa5; font-size: 19px; font-family: Lumberjack;
width: auto"> Пейринг:</a> <?= htmlspecialchars($work['pairing']) ?></p>
                <p><a  style="color:  #ff7fa5">Персонажі:</a>  <?=
htmlspecialchars($characters_str) ?></p>
                <p><a style="color: #ff7fa5; font-size: 19px; font-family: Lumberjack;
width: auto"> Напрямок:</a> <?= htmlspecialchars($work['direction']) ?></p>
                <p><a style="color: #ff7fa5; font-size: 19px; font-family: Lumberjack;
width: auto"> Жанр:</a> <?= htmlspecialchars($work['genre']) ?></p>
                <p><a style="color: #ff7fa5; font-size: 19px; font-family: Lumberjack;
width: auto"> Опис:</a> <?= htmlspecialchars($work['description']) ?></p>
                <p><a style="color: #ff7fa5; font-size: 19px; font-family: Lumberjack;
width: auto"> Дата публікації:</a> <?= date("d.m.Y", strtotime($work['created_at']))
?></p>
                <?php if ($user_id == $work['user_id']): ?>
                  <div class="change-button-header">
                    <a href="change_header_work.php?id=<?= $work['id'] ?>">
                      <button>Редагувати шапку твору</button>
                    </a>
                  </div>
                <?php endif; ?>
              </div>
              <hr>
              <div class="content-text">
                <?= purify_html($work['content']) ?>
```

```
            </div>
            <?php if ($user_id == $work['user_id']): ?>
               <div class="change-button">
                  <a href="update_work.php?id=<?= $work['id'] ?>">
                     <button>Редагувати</button>
                  </a>
               </div>
            <?php endif; ?>
            <?php include "footer.php" ?>
         </div>
      </div>
      </body>
      </html>
```

4. Код сторінки «details.php»:

```
      <?php include "session.php"?>
      <?php
      $table = $_GET['table'] ?? '';
      $id = $_GET['id'] ?? '';
      if (!in_array($table, ['users', 'works', 'collections', 'fandoms', 'genres', 'characters']) || !is_numeric($id)) {
         echo "Невірні параметри.";
         exit;
      }
      $dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
      $db_username = "postgres";
      $db_password = "1131";
      try {
```

```
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    if ($table === 'users') {
        header("Location: guest_profile.php?id=" . urlencode($id));
        exit;
    }
    if ($table === 'works') {
        $stmt = $pdo->prepare("SELECT * FROM works WHERE id = :id AND
draft = 1");
        $stmt->execute(['id' => $id]);
        $work = $stmt->fetch();
        $works = $work ? [$work] : [];
        $block_title = "Твір: " . htmlspecialchars($work['title'] ?? '');
    }
    elseif ($table === 'collections') {
        $stmt = $pdo->prepare("
        SELECT w.* FROM works w WHERE draft = 1
        JOIN collection_works cw ON w.id = cw.work_id
        WHERE cw.collection_id = :id
        ORDER BY w.created_at DESC
    ");
        $stmt->execute(['id' => $id]);
        $works = $stmt->fetchAll();
        $f = $pdo->prepare("SELECT * FROM collections WHERE id = :id");
        $f->execute(['id' => $id]);
        $collection = $f->fetch();
        $block_title = "Збірка: " . htmlspecialchars($collection['name'] ?? '');
    }
```

```
        elseif ($table === 'fandoms') {
            $stmt = $pdo->prepare("SELECT * FROM works WHERE fandom ILIKE (SELECT name FROM fandoms WHERE id = :id) AND draft = 1 ORDER BY created_at DESC");
            $stmt->execute(['id' => $id]);
            $works = $stmt->fetchAll();
            $f = $pdo->prepare("SELECT name FROM fandoms WHERE id = :id");
            $f->execute(['id' => $id]);
            $fandom = $f->fetch();
            $block_title = "Фандом: " . htmlspecialchars($fandom['name'] ?? '');
        }
        elseif ($table === 'genres') {
            $stmt = $pdo->prepare("SELECT * FROM works WHERE genre ILIKE (SELECT name FROM genres WHERE id = :id) AND draft = 1 ORDER BY created_at DESC");
            $stmt->execute(['id' => $id]);
            $works = $stmt->fetchAll();
            $f = $pdo->prepare("SELECT name FROM genres WHERE id = :id");
            $f->execute(['id' => $id]);
            $genre = $f->fetch();
            $block_title = "Жанр: " . htmlspecialchars($genre['name'] ?? '');
        }
        elseif ($table === 'characters') {
            $stmt = $pdo->prepare("
            SELECT w.* FROM works w
            JOIN work_characters wc ON w.id = wc.work_id
            WHERE wc.character_id = :id AND w.draft = 1
            ORDER BY w.created_at DESC
        ");
```

```
            $stmt->execute(['id' => $id]);
            $works = $stmt->fetchAll();
            $f = $pdo->prepare("SELECT name FROM characters WHERE id = :id");
            $f->execute(['id' => $id]);
            $character = $f->fetch();
            $block_title = "Персонаж: " . htmlspecialchars($character['name'] ?? '');
        }
        if (!isset($works)) {
            $works = [];
        }
        $works_in = ['прочитано' => [], 'улюблене' => []];
        if (isset($_SESSION['user_id'])) {
            $stmt = $pdo->prepare("
                SELECT c.name, cw.work_id FROM collections c
                JOIN collection_works cw ON c.id = cw.collection_id
                WHERE c.user_id = :user_id AND c.name IN ('Прочитано', 'Улюблене')
            ");
            $stmt->execute(['user_id' => $_SESSION['user_id']]);
            while ($row = $stmt->fetch()) {
                $key = mb_strtolower($row['name'], 'UTF-8');
                $works_in[$key][] = $row['work_id'];
            }
        }
    } catch (PDOException $e) {
        echo "Помилка БД: " . $e->getMessage();
        exit;
    }
    ?>
```

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="../css/base.css">
    <link rel="stylesheet" href="../css/details.css">
    <link rel="stylesheet" href="../css/footer.css">
    <title>Результати пошуку</title>
</head>
<body>
<div class="container">
    <?php include "menu.php"?>
    <div class="content">
        <?php
        global $works, $works_in;
        include "fanfics.php";
        ?>
        <?php include "footer.php"?>
    </div>
</div>
</body>
</html>
```

5. Код сторінки «session.php»:

```
<?php
session_start();
$role = $_SESSION['user_type'] ?? 'guest';
$username = $_SESSION['username'] ?? null;
$user_id = $_SESSION['user_id'] ?? null;
?>
```

6. Код сторінки «footer.php»:

```
<?php
?>
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="../css/footer.css">
    <title>Футер</title>
</head>
<body>
<div class="footer">
    <div class="footer-content">
        <div class="footer-section">
            <h4 class="footer-h4">Про платформу</h4>
            <p class="footer-p">Платформа для починаючих авторів, де кожен може викладати свої твори, читати інших і обмінюватися ідеями.</p>
```

```
            </div>
            <div class="footer-section">
                <h4 class="footer-h4">Контакти</h4>
                <p class="footer-p">Email: fanfic.support@google.com</p>
                <p class="footer-p">Телефон: +380 12 345 6789</p>
            </div>
            <div class="footer-section">
                <h4 class="footer-h4">Соціальні мережі</h4>
                <a href="#">Facebook</a> |
                <a href="#">Instagram</a> |
                <a href="#">Twitter</a>
            </div>
        </div>
        <div class="footer-bottom">
            <p class="footer-p">© 2025 Всі права захищено. Авторська платформа</p>
        </div>
    </div>
    </body>
    </html>
```

7. Код сторінки «search.php»:

```
<?php
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
// Отримуємо пошуковий запит
$q = trim($_GET['q'] ?? '');
```

```
if (strlen($q) < 3) {
    exit;
}
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    $query2 = "%$q%";
    $q = mb_convert_case($q, MB_CASE_TITLE, "UTF-8");
    $query = "%$q%";
    $results = [];
    $stmt = $pdo->prepare(
        "SELECT id, title FROM works
                WHERE (title ILIKE :q OR title ILIKE :q2) AND draft = 1
                ORDER BY created_at DESC LIMIT 10");
    $stmt->execute([
        'q' => $query,
        'q2' => $query2]);
    $results['works'] = $stmt->fetchAll();
    //пошук у жанрах
    $stmt = $pdo->prepare(
        "SELECT id, name FROM genres
                WHERE (name ILIKE :q OR name ILIKE :q2)
                LIMIT 10");
    $stmt->execute([
        'q' => $query,
        'q2' => $query2]);
    $results['genres'] = $stmt->fetchAll();
    $stmt = $pdo->prepare(
```

```
        "SELECT id, name FROM characters
                WHERE (name ILIKE :q OR name ILIKE :q2)
                LIMIT 10");
    $stmt->execute([
        'q' => $query,
        'q2' => $query2]);
    $results['characters'] = $stmt->fetchAll();
//пошук у збірках
    $stmt = $pdo->prepare("
    SELECT id, name FROM collections
    WHERE name ILIKE :q1 OR name ILIKE :q2
    ORDER BY created_at DESC
    LIMIT 10
");
    $stmt->execute([
        'q1' => $query,
        'q2' => $query2
    ]);
    $results['collections'] = $stmt->fetchAll();
    // Пошук у fandoms (за name)
    $stmt = $pdo->prepare(
        "SELECT id, name FROM fandoms
                WHERE name ILIKE :q OR name ILIKE :q2
                ORDER BY name LIMIT 10");
    $stmt->execute([
        'q' => $query,
        'q2' => $query2]);
    $results['fandoms'] = $stmt->fetchAll();
    // Пошук у users (за name)
```

```
        $stmt = $pdo->prepare(
            "SELECT id, name FROM users
                WHERE name ILIKE :q OR name ILIKE :q2
                ORDER BY name LIMIT 5");
        $stmt->execute([
            'q' => $query,
            'q2' => $query2]);
        $results['users'] = $stmt->fetchAll();
        // Виведення результатів
        foreach ($results as $table => $rows) {
            if (count($rows)) {
                echo "<div><strong>" . ucfirst($table) . ":</strong></div>";
                foreach ($rows as $row) {
                    $label = htmlspecialchars($row['title'] ?? $row['name'] ?? $row['login']);
                    echo "<div onclick=\"location.href='details.php?table=$table&id={$row['id']}'\">$label</div>";
                }
            }
        }
        if (empty(array_filter($results))) {
            echo "<div>Нічого не знайдено.</div>";
        }
    } catch (PDOException $e) {
        echo "<div>Помилка підключення до бази.</div>";
    }
    ?>
```

8. Код сторінки «create_work.php»:

```
<?php include "session.php"?>
<?php
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
header('Content-Type: text/html; charset=UTF-8');
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    // Завантаження опцій
    $fandoms = $pdo->query("SELECT * FROM fandoms ORDER BY name")->fetchAll();
    $characters = $pdo->query("SELECT * FROM characters ORDER BY name")->fetchAll();
    $directions = $pdo->query("SELECT * FROM directions ORDER BY name")->fetchAll();
    $genres = $pdo->query("SELECT * FROM genres ORDER BY name")->fetchAll();
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $title = trim($_POST['title'] ?? '');
        $description = trim($_POST['description'] ?? '');
        $fandom_id = !empty($_POST['fandom_id']) ? (int)$_POST['fandom_id'] : null;
        $character_ids = !empty($_POST['characters']) ? explode(',', $_POST['characters']) : [];
        $pairing_ids = !empty($_POST['pairing']) ? explode(',', $_POST['pairing']) : [];
```

```
$direction_id = (int)($_POST['direction_id'] ?? 0);
$genre_id = (int)($_POST['genre_id'] ?? 0);
// Назви фандому, напрямку, жанру
$fandom_name = 'Оріджин';
if ($fandom_id) {
   $stmt = $pdo->prepare("SELECT name FROM fandoms WHERE id =
:id");
   $stmt->execute(['id' => $fandom_id]);
   $fandom_name = $stmt->fetchColumn() ?: $fandom_name;
}
$direction_name = '';
if ($direction_id) {
   $stmt = $pdo->prepare("SELECT name FROM directions WHERE id =
:id");
   $stmt->execute(['id' => $direction_id]);
   $direction_name = $stmt->fetchColumn();
}
$genre_name = '';
if ($genre_id) {
   $stmt = $pdo->prepare("SELECT name FROM genres WHERE id =
:id");
   $stmt->execute(['id' => $genre_id]);
   $genre_name = $stmt->fetchColumn();
}
// Формуємо пейринги
$pairing_str = '-';
if (!empty($pairing_ids)) {
   $stmt = $pdo->prepare("SELECT name FROM characters WHERE id =
?");
```

```
            $pairs = [];
            foreach ($pairing_ids as $pair) {
                if (strpos($pair, '/') !== false) {
                    [$id1, $id2] = explode('/', $pair);
                    $stmt->execute([$id1]);
                    $name1 = $stmt->fetchColumn();
                    $stmt->execute([$id2]);
                    $name2 = $stmt->fetchColumn();
                    if ($name1 && $name2) {
                        $pairs[] = "$name1/$name2";
                    }
                }
            }
            $pairing_str = implode(', ', $pairs) ?: '-';
        }
        // Додавання твору
        $stmt = $pdo->prepare("INSERT INTO works (user_id, title, fandom, pairing, characters, direction, genre, description, created_at)
                               VALUES (:user_id, :title, :fandom, :pairing, '', :direction, :genre, :description, now())");
        $stmt->bindParam(':user_id', $user_id);
        $stmt->bindParam(':title', $title);
        $stmt->bindParam(':fandom', $fandom_name);
        $stmt->bindParam(':pairing', $pairing_str);
        $stmt->bindParam(':direction', $direction_name);
        $stmt->bindParam(':genre', $genre_name);
        $stmt->bindParam(':description', $description);
        $stmt->execute();
        // Отримуємо ID нового твору
```

```
            $work_id = $pdo->lastInsertId("works_id_seq");
            // Додаємо персонажів до зв’язкової таблиці
            if (!empty($character_ids)) {
                $insertCharacter = $pdo->prepare("INSERT INTO work_characters (work_id, character_id) VALUES (:work_id, :character_id)");
                foreach ($character_ids as $char_id) {
                    $insertCharacter->execute([
                        'work_id' => $work_id,
                        'character_id' => (int)$char_id
                    ]);
                }
            }
            header("Location: update_work.php?id=" . $work_id);
            exit;
        }
    } catch (PDOException $e) {
        echo "Database error: " . $e->getMessage();
        exit;
    }
    ?>
    <!doctype html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport"
              content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <link rel="stylesheet" href="../css/base.css">
```

```
        <link rel="stylesheet" href="../css/create_work.css">
        <link rel="stylesheet" href="../css/footer.css">
        <title>Створити твір</title>
    </head>
    <body>
    <div class="container">
        <?php include "menu.php"?>
        <div class="content">
            <div class="forms">
                <h2>Додати новий твір</h2>
                <form method="post" enctype="multipart/form-data">
                    <label>Назва:</label>
                    <input type="text" name="title" MAXLENGTH="255" required>
                    <label>Опис:</label>
                    <textarea name="description" MAXLENGTH="500"
required></textarea>
                    <br>
                    <label>Фандом:</label>
                    <select name="fandom_id" id="fandom-select"
onchange="filterCharactersByFandom()">
                        <option value="">Оберіть фандом</option>
                        <?php foreach ($fandoms as $f): ?>
                            <option value="<?= $f['id'] ?>"><?= htmlspecialchars($f['name'])
?></option>
                        <?php endforeach; ?>
                    </select>
                    <label>Персонажі:</label>
                    <select id="character-select"></select>
                    <button type="button" onclick="addCharacter()">Додати</button>
```

```
                <div id="character-list"></div>
                <input type="hidden" name="characters" id="characters-hidden">
                <label>Пейринг:</label>
                <select id="pairing-select"></select>
                <button type="button" onclick="addPairing()">Додати</button>
                <div id="pairing-list"></div>
                <input type="hidden" name="pairing" id="pairing-hidden">
                <label>Напрямок:</label>
                <select name="direction_id" required>
                    <option value="">Оберіть напрямок</option>
                    <?php foreach ($directions as $d): ?>
                        <option value="<?= $d['id'] ?>"><?=
htmlspecialchars($d['name']) ?></option>
                    <?php endforeach; ?>
                </select>
                <label>Жанр:</label>
                <select name="genre_id" required>
                    <option value="">Оберіть жанр</option>
                    <?php foreach ($genres as $g): ?>
                        <option value="<?= $g['id'] ?>"><?=
htmlspecialchars($g['name']) ?></option>
                    <?php endforeach; ?>
                </select>
                <button type="submit">Створити</button>
            </form>
        </div>
        <?php include "footer.php"?>
    </div>
</div>
```

```
<script>
    const allCharacters = <?= json_encode($characters, JSON_UNESCAPED_UNICODE) ?>;
</script>
<script src="../js/create_work.js"></script>
</body>
</html>
```

9. Код сторінки «update_work.php»:

```
<?php
include "session.php";
require_once __DIR__ . '/../libs/htmlpurifier-4.15.0/library/HTMLPurifier.auto.php';
// Отримуємо ID твору з параметру GET
$work_id = $_GET['id'] ?? null;
if (!$work_id) {
    header("Location: account.php");
    exit;
}
// Підключення до бази
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    // Отримуємо поточний контент
```

```
        $stmt = $pdo->prepare("SELECT content, title, draft FROM works WHERE id = :id");
        $stmt->execute(['id' => $work_id]);
        $work = $stmt->fetch(PDO::FETCH_ASSOC);
        if (!$work) {
            echo "Твір не знайдено.";
            exit;
        }
    } catch (PDOException $e) {
        echo "Database error: " . $e->getMessage();
        exit;
    }
    function purify_html($dirty_html) {
        static $purifier = null;
        if ($purifier === null) {
            $config = HTMLPurifier_Config::createDefault();
            $config->set('HTML.Allowed', 'p,b,i,u,div,span,br,strong,em,s,sub,sup,ul,ol,li,blockquote,hr,pre,code,style');
            $purifier = new HTMLPurifier($config);
        }
        return $purifier->purify($dirty_html);
    }
    // Збереження змін
    if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['content'])) {
        $new_content = $_POST['content'];
        $stmt = $pdo->prepare("UPDATE works SET content = :content WHERE id = :id");
        $stmt->execute([
```

```
            'content' => $new_content,
            'id' => $work_id
        ]);
        // Якщо це зміна чернетки, перенаправлення на ту ж сторінку
        if (isset($_POST['toggle_draft'])) {
            $new_draft_value = ($_POST['toggle_draft'] == '1') ? 1 : 0;
            $stmt = $pdo->prepare("UPDATE works SET draft = :draft WHERE id =
:id");
            $stmt->execute([
                'draft' => $new_draft_value,
                'id' => $work_id
            ]);
            header("Location: update_work.php?id=$work_id");
            exit;
        }
        // Якщо це просто збереження
        header("Location: read.php?id=$work_id");
        exit;
    }
    ?>
    <!doctype html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport"
              content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <link rel="stylesheet" href="../css/base.css">
```

```
        <link rel="stylesheet" href="../css/update_work.css">
        <title>Редагувати твір</title>
    </head>
    <body>
    <div class="container">
        <?php include "menu.php" ?>
        <div class="content">
            <h2>Редагування твору: "<?= $work['title'] ?>"</h2>
            <form method="post" id="editForm">
            <div class="update">
                <div class="toolbar">
                    <button type="button" onclick="format('bold')"><b>B</b></button>
                    <button type="button" onclick="format('italic')"><i>K</i></button>
                    <button type="button" onclick="format('underline')"><u>U</u></button>
                    <button type="button" onclick="format('strikeThrough')"><s>S</s></button>
                    <button type="button" onclick="format('justifyLeft')">Ліворуч</button>
                    <button type="button" onclick="format('justifyCenter')">По центру</button>
                    <button type="button" onclick="format('justifyRight')">Праворуч</button>
                </div>
                <div id="editor" contenteditable="true"><?= $work['content'] ?></div>
                <textarea name="content" id="content" hidden></textarea>
```

```
                <br>
                <div class="toolbar">
                    <button type="submit" name="save">Зберегти</button>
                    <?php if ($work['draft'] == 0): ?>
                        <button type="submit" name="toggle_draft" value="1">Опублікувати</button>
                    <?php else: ?>
                        <button type="submit" name="toggle_draft" value="0">Зробити чернеткою</button>
                    <?php endif; ?>
                    <a href="read.php?id=<?= $work_id ?>">
                        <button type="button">Перейти до твору</button>
                    </a>
                </div>
            </div>
        </form>
    </div>
</div>
<script src="../js/update_work.js"></script>
</body>
</html>
```

10. Код сторінки «my_works.php»:

```
<?php
include "session.php";
global $user_id;
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
```

```
$db_password = "1131";
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    $stmt = $pdo->prepare("SELECT * FROM works WHERE user_id = :user_id ORDER BY created_at DESC");
    $stmt->execute(['user_id' => $user_id]);
    $all_works = $stmt->fetchAll();

    include "fanf_menu.php";
} catch (PDOException $e) {
    echo "Database error: " . $e->getMessage();
    exit;
}
?>
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="../css/base.css">
    <link rel="stylesheet" href="../css/my_works.css">
    <link rel="stylesheet" href="../css/fanfics.css">
    <title>Мої твори</title>
</head>
```

```
<body>
<div class="container">
  <?php include "menu.php" ?>
  <div class="content">
    <div class="work-tabs">
      <button class="tab-button" onclick="showTab('published')">Опубліковані</button>
      <button class="tab-button" onclick="showTab('drafts')">Чернетки</button>
    </div>
    <div id="published" class="tab-content">
      <?php
      $is_my_works = true;
      global $works;
      global $works_in;
      $published = array_filter($all_works, function($w) { return $w['draft'] == 1; });
      $works = $published;
      $block_title = "Опубліковані твори";
      include "fanfics.php";
      ?>
    </div>
    <div id="drafts" class="tab-content" style="display: none;">
      <?php
      global $works;
      global $works_in;
      $is_my_works = true;
      $drafts = array_filter($all_works, function($w) { return $w['draft'] == 0; });
```

```
            $works = $drafts;
            $block_title = "Чернетки";
            include "fanfics.php";
            ?>
        </div>
        <?php include "footer.php" ?>
    </div>
</div>
<script src="../js/my_works.js"></script>
</body>
</html>
```

11. Код сторінки «my_collections.php»:

```
<?php include "session.php"; ?>
<?php
global $user_id;
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    $stmt = $pdo->prepare("SELECT id, name FROM collections WHERE user_id = :user_id ORDER BY created_at ASC ");
    $stmt->execute(['user_id' => $user_id]);
    $collections = $stmt->fetchAll();
} catch (PDOException $e) {
```

```
    echo "Database error: " . $e->getMessage();
    exit;
}
?>
<!doctype html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Мої збірки</title>
    <link rel="stylesheet" href="../css/base.css">
    <link rel="stylesheet" href="../css/my_collections.css">
    <link rel="stylesheet" href="../css/footer.css">
</head>
<script src="../js/my_collections.js"></script>
<body>
<div class="container">
    <?php include "menu.php" ?>
    <div class="content">
        <h2>Мої збірки</h2>
        <div class="collections">
            <?php if (empty($collections)): ?>
                <p style="color:white; padding: 15px; text-align: center; font-size: 20px;">У вас ще немає збірок.</p>
            <?php else: ?>
                <ul class="collection-list">
                    <?php foreach ($collections as $collection): ?>
                        <li>
                            <a href="collection_works.php?collection_id=<?= $collection['id'] ?>">
```

```
                        <?= htmlspecialchars($collection['name']) ?>
                    </a>
                    <?php
                    $name_lc = mb_strtolower($collection['name'], 'UTF-8');
                    if (!in_array($name_lc, ['прочитано', 'улюблене'])): ?>
                        <span class="delete-btn" onclick="confirmDelete(<?=
$collection['id'] ?>)">❌</span>
                    <?php endif; ?>
                </li>
            <?php endforeach; ?>
        </ul>
    <?php endif; ?>
</div>
<?php include "footer.php" ?>
</div>
</div>
</body>
</html>
```

12. Код сторінки «admin_add.php»:

```
<?php
session_start();
// Перевіряємо, чи користувач — адміністратор
if (!isset($_SESSION['user_type']) || $_SESSION['user_type'] !== 'admin') {
    die("Доступ заборонено. Ви не адміністратор.");
    header("Location: login_form.php");
    exit;
}
```

```
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC
    ]);
    // Додавання фандому з перевіркою унікальності
    if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['add_fandom'])) {
        $fandom_name = trim($_POST['fandom_name']);
        if (!empty($fandom_name)) {
            $fandom_name_lower = mb_strtolower($fandom_name, 'UTF-8');
            $stmt = $pdo->prepare("SELECT id FROM fandoms WHERE LOWER(name COLLATE \"und-x-icu\") = LOWER(:name COLLATE \"und-x-icu\")");
            $stmt->execute(['name' => $fandom_name_lower]);
            $existing_fandom = $stmt->fetch();
            if ($existing_fandom) {
                $error = "Цей фандом вже існує!";
            } else {
                $stmt = $pdo->prepare("INSERT INTO fandoms (name) VALUES (:name)");
                $stmt->execute(['name' => $fandom_name]);
                $success = "Фандом успішно доданий!";
            }
        }
    }
```

```
// Додавання персонажа
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['add_character'])) {
    $fandom_id = $_POST['fandom_id'];
    $character_name = trim($_POST['character_name']);
    if (!empty($character_name) && !empty($fandom_id)) {
        // Приводимо ім'я персонажа до нижнього регістру з підтримкою всіх мов
        $character_name_lower = mb_strtolower($character_name, 'UTF-8');
        $stmt = $pdo->prepare("SELECT id FROM characters WHERE fandom_id = :fandom_id AND LOWER(name COLLATE \"und-x-icu\") = LOWER(:name COLLATE \"und-x-icu\")");
        $stmt->execute([
            'fandom_id' => $fandom_id,
            'name' => $character_name_lower
        ]);
        $existing_character = $stmt->fetch();
        if ($existing_character) {
            $error = "Цей персонаж вже існує у вибраному фандомі!";
        } else {
            try {
                // Додаємо персонажа (оригінальний регістр зберігається)
                $stmt = $pdo->prepare("INSERT INTO characters (fandom_id, name) VALUES (:fandom_id, :name)");
                $stmt->execute([
                    'fandom_id' => $fandom_id,
                    'name' => $character_name
                ]);
                $success = "Персонаж успішно доданий!";
```

```
                } catch (PDOException $e) {
                    if ($e->getCode() == '23505') {
                        $error = "Цей персонаж вже існує у вибраному фандомі!";
                    } else {
                        $error = "Помилка бази даних: " . $e->getMessage();
                    }
                }
            }
        }
        // Видалення фандому
    if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['delete_fandom'])) {
        $fandom_id = $_POST['delete_fandom'];
        $stmt = $pdo->prepare("DELETE FROM fandoms WHERE id = :id");
        $stmt->execute(['id' => $fandom_id]);
    }
    // Видалення персонажа
    if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['delete_character'])) {
        $character_id = $_POST['delete_character'];
        $stmt = $pdo->prepare("DELETE FROM characters WHERE id = :id");
        $stmt->execute(['id' => $character_id]);
    }
    // Отримання списку фандомів
    $stmt = $pdo->query("SELECT * FROM fandoms ORDER BY name");
    $fandoms = $stmt->fetchAll();
    // Отримання списку персонажів
```

```
        $stmt = $pdo->query("SELECT characters.id, characters.name, fandoms.name AS fandom_name
                             FROM characters
                             JOIN fandoms ON characters.fandom_id = fandoms.id
                             ORDER BY fandoms.name, characters.name");
        $characters = $stmt->fetchAll();
    } catch (PDOException $e) {
        die("Помилка підключення: " . $e->getMessage());
    }
    ?>
    <!doctype html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport"
              content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <link rel="stylesheet" href="../css/base.css">
        <link rel="stylesheet" href="../css/admin_add.css">
        <link rel="stylesheet" href="../css/footer.css">
        <title>Додати фандом</title>
    </head>
    <body>
    <div class="container">
        <?php include "menu.php" ?>
        <div class="content">
            <div class="forms">
                <?php if (isset($error)): ?>
```

```
            <p class="error-message"> <?= $error ?> </p>
        <?php endif; ?>
        <?php if (isset($success)): ?>
            <p class="success-message"> <?= $success ?> </p>
        <?php endif; ?>
        <h2>Додати фандом</h2>
        <form method="POST">
            <label>Назва фандому:</label>
            <input type="text" name="fandom_name" required>
            <button type="submit" name="add_fandom">Додати фандом</button>
        </form>
        <h2>Додати персонажа</h2>
        <form method="POST">
            <label>Фандом:</label>
            <select name="fandom_id" required>
                <option value="">Оберіть фандом</option>
                <?php foreach ($fandoms as $fandom): ?>
                    <option value="<?= $fandom['id'] ?>"><?= htmlspecialchars($fandom['name']) ?></option>
                <?php endforeach; ?>
            </select>
            <label>Ім'я персонажа:</label>
            <input type="text" name="character_name" required>
            <button type="submit" name="add_character">Додати персонажа</button>
        </form>
        <h2>Фандоми</h2>
        <div class="list">
```

```
            <?php foreach ($fandoms as $fandom): ?>
                <div class="item">
                    <span class="label"><?= htmlspecialchars($fandom['name']) ?></span>
                    <form method="POST" style="display:inline;">
                        <button type="submit" name="delete_fandom" value="<?= $fandom['id'] ?>" class="delete-btn"
                                onclick="return confirm('Ви дійсно хочете видалити фандом?')">❌
                        </button>
                    </form>
                </div>
            <?php endforeach; ?>
        </div>
        <h2>Персонажі</h2>
        <div class="list">
            <?php foreach ($characters as $character): ?>
                <div class="item">
                    <?= htmlspecialchars($character['name']) ?> (<?= htmlspecialchars($character['fandom_name']) ?>)
                    <form method="POST" style="display:inline;">
                        <button type="submit" name="delete_character" value="<?= $character['id'] ?>"
                                class="delete-btn">❌
                        </button>
                    </form>
                </div>
            <?php endforeach; ?>
        </div>
```

```
        </div>
        <?php include "footer.php" ?>
    </div>
</div>
</body>
</html>
```

13. Код сторінки «admin_change.php»:

```
<?php include "session.php"?>
<?php
session_start();
// Перевіряємо, чи користувач — адміністратор
if (!isset($_SESSION['user_type']) || $_SESSION['user_type'] !== 'admin') {
    die("Доступ заборонено. Ви не адміністратор.");
}
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC
    ]);
    // Додавання фандому з перевіркою унікальності
    if ($_SERVER["REQUEST_METHOD"] == "POST" &&
isset($_POST['add_fandom'])) {
        $fandom_name = trim($_POST['fandom_name']);
        if (!empty($fandom_name)) {
```

```
            $fandom_name_lower = mb_strtolower($fandom_name, 'UTF-8');
            $stmt = $pdo->prepare("SELECT id FROM fandoms WHERE LOWER(name COLLATE \"und-x-icu\") = LOWER(:name COLLATE \"und-x-icu\")");
            $stmt->execute(['name' => $fandom_name_lower]);
            $existing_fandom = $stmt->fetch();
            if ($existing_fandom) {
                $error = "Цей фандом вже існує!";
            } else {
                $stmt = $pdo->prepare("INSERT INTO fandoms (name) VALUES (:name)");
                $stmt->execute(['name' => $fandom_name]);
                $success = "Фандом успішно доданий!";
            }
        }
    }
    // Редагування фандому
    if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['edit_fandom'])) {
        $fandom_id = $_POST['fandom_id'];
        $new_name = trim($_POST['new_fandom_name']);
        if (!empty($new_name)) {
            $stmt = $pdo->prepare("UPDATE fandoms SET name = :name WHERE id = :id");
            $stmt->execute(['name' => $new_name, 'id' => $fandom_id]);
            $success = "Фандом успішно оновлено!";
        }
    }
    // Редагування персонажа
```

```
        if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['edit_character'])) {
            $character_id = $_POST['character_id'];
            $new_name = trim($_POST['new_character_name']);
            if (!empty($new_name)) {
                $stmt = $pdo->prepare("UPDATE characters SET name = :name WHERE id = :id");
                $stmt->execute(['name' => $new_name, 'id' => $character_id]);
                $success = "Персонаж успішно оновлено!";
            }
        }
        // Отримання списку фандомів
        $stmt = $pdo->query("SELECT * FROM fandoms ORDER BY name");
        $fandoms = $stmt->fetchAll();
        // Отримання списку персонажів
        $stmt = $pdo->query("SELECT characters.id, characters.name, fandoms.name AS fandom_name
                             FROM characters
                             JOIN fandoms ON characters.fandom_id = fandoms.id
                             ORDER BY fandoms.name, characters.name");
        $characters = $stmt->fetchAll();
    } catch (PDOException $e) {
        die("Помилка підключення: " . $e->getMessage());
    }
    ?>
    <!doctype html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
```

```
        <meta name="viewport"
              content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <link rel="stylesheet" href="../css/base.css">
        <link rel="stylesheet" href="../css/admin_change.css">
        <link rel="stylesheet" href="../css/footer.css">
        <title>Редагувати фандом</title>
    </head>
    <body>
    <div class="container">
        <?php include "menu.php"?>
        <div class="content">
            <div class="forms">
                <h1>Редагування фандомів</h1>
                <form method="POST">
                    <select name="fandom_id" required>
                        <option value="">Оберіть фандом</option>
                        <?php foreach ($fandoms as $fandom): ?>
                            <option value="<?= $fandom['id'] ?>"><?=
htmlspecialchars($fandom['name']) ?></option>
                        <?php endforeach; ?>
                    </select>
                    <label>Нова назва:</label>
                    <input type="text" name="new_fandom_name" required>
                    <button type="submit" name="edit_fandom">Оновити</button>
                </form>
                <h1>Редагування персонажів</h1>
                <form method="POST">
```

```
                <select name="character_id" required>
                    <option value="">Оберіть персонажа</option>
                    <?php foreach ($characters as $character): ?>
                        <option value="<?= $character['id'] ?>"><?=
htmlspecialchars($character['name']) ?> (<?=
htmlspecialchars($character['fandom_name']) ?>)</option>
                    <?php endforeach; ?>
                </select>
                <label>Нове ім'я:</label>
                <input type="text" name="new_character_name" required>
                <button type="submit" name="edit_character">Оновити</button>
            </form>
        </div>
        <?php include "footer.php"?>
    </div>
</div>
</body>
</html>
```

14. Код сторінки «users.php»:

```
<?php include "session.php";
if (!isset($_SESSION['user_type']) || $_SESSION['user_type'] !== 'admin') {
    die("Доступ заборонено. Ви не адміністратор.");
}
$dsn = "pgsql:host=localhost;dbname=postgres;port=5432";
$db_username = "postgres";
$db_password = "1131";
global $user_id;
```

```
try {
    $pdo = new PDO($dsn, $db_username, $db_password, [
        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
    ]);
    $stmt = $pdo->prepare("SELECT * FROM users");
    $stmt->execute();
    $users = $stmt->fetchAll();
} catch (PDOException $e) {
}
?>
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="../css/base.css">
    <link rel="stylesheet" href="../css/users.css">
    <title>Користувачі</title>
</head>
<body>
<div class="container">
    <?php include "menu.php" ?>
    <div class="content">
        <div class="users-table">
            <h2>Користувачі</h2>
            <table>
```

```
                <tr>
                    <th>Ім'я користувача</th>
                    <th>Роль</th>
                </tr>
                <?php foreach ($users as $user): ?>
                    <tr>
                        <td><?= htmlspecialchars($user['name']) ?></td>
                        <td>
                            <select onchange="updateUserRole(<?= $user['id'] ?>, this.value)">
                                <option value="user" <?= $user['user_type'] === 'user' ? 'selected' : '' ?>>user
                                </option>
                                <option value="admin" <?= $user['user_type'] === 'admin' ? 'selected' : '' ?>>admin
                                </option>
                            </select>
                        </td>
                    </tr>
                <?php endforeach; ?>
            </table>
        </div>
        <?php include "footer.php" ?>
    </div>
</div>
<script src="../js/users.js"></script>
</body>
</html>
```