МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю.М. Потебні
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

КАФЕДРА ЕЛЕКТРОНІКИ, ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Кваліфікаційна робота

Перший (бакалаврський)
(рівень вищої освіти)

на тему **Розробка онлайн-бібліотеки японських коміксів Мáнґа за допомогою React та Node.JS**

Виконав: студент 4 курсу, групи 6.1210-пзс-с
спеціальності 121 Інженерія програмного забезпечення
(код і назва спеціальності)
освітньої програми Програмне забезпечення систем
(код і назва освітньої програми)
Я. О. Котенко
(ініціали та прізвище)
Керівник доцент, к.т.н., доцент Ю.О. Лимаренко
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)
Рецензент директор ТОВ «Дісітел»
П.О Лютий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)

Запоріжжя
2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю.М. Потебні
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

Кафедра електроніки, інформаційних систем та програмного забезпечення
Рівень вищої освіти _____перший (бакалаврський)_____
Спеціальність ___121_Інженерія програмного забезпечення_____
(код та назва)
Освітня програма ___Програмне забезпечення систем
(код та назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри_____Т.В. Критська_
“_01_” ___березня__ 2023_року

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

Котенко Ярослав Олександрович
(прізвище, ім’я, по батькові)

1.Тема роботи _Розробка онлайн-бібліотеки японських коміксів Мáнґа за допомогою React таNode.JS

керівник роботи ___Лимаренко Юлія Олексіївна, доцент, к.т.н._____
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом ЗНУ від 29.12.2022 р. № 1893-с

2. Строк подання студентом кваліфікаційної роботи_____14.06.2023_____
3. Вихідні дані бакалаврської роботи:
- комплект нормативних документів ;
- технічне завдання до роботи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
- огляд та аналіз існуючих рішень та аналогів,
- аналіз сучасних технологій для створення клієнтьскої й серверної частини вебзастосунку,
- вимоги замовника,
- розробка макету клієнтської частини,
- побудова бази даних з описом усіх сутностей,
- опис алгоритму додатку,
- розробка вебзастосунку онлайн бібліотеки читання коміксів

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
17 слайдів презентації

6. Консультанти розділів бакалаврської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата Завдання прийняв

7. Дата видачі завдання 01.03.2023

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Аналіз предметної області	20.04.23	виконано
2	Формулювання основної задачі дипломної роботи та узгодження її з науковим керівником	26.04.23	виконано
3	Аналіз існуючих методів рішення	26.04-29.04.23	виконано
4	Аналіз сучасних фреймворків для розробки серверної частини	30.04-03.05.23	виконано
5	Аналіз сучасних технологій для розробки користувацької частини	04.05-07.05.23	виконано
6	Узгодження подальших дій з науковим керівником	10.05.23	виконано
7	Проєктування бази даних	11.05.23	виконано
8	Програмна реалізація серверної частини застосунку	14.05-22.05.23	виконано
9	Представлення отриманих результатів науковому керівнику та узгодження плану подальшого дослідження	27.05.23	виконано
10	Реалізація користувацького інтерфейсу	28.05-08.06.23	виконано
11	Тестування застосунку	08.06.23	виконано
12	Оформлення звіту	09.06-11.06.23	виконано
13	Оформлення презентації.	12.06-15.06.23	виконано

Студент _____ Я.О. Котенко
(підпис) (прізвище та ініціали)

Керівник роботи _____ Ю.О. Лимаренко
(підпис) (прізвище та ініціали)

Нормоконтроль пройдено
Нормоконтролер _____ І.А. Скрипник
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Сторінок – 75

Рисунків – 20

Таблиць – 24

Джерел – 13

Кваліфікаційна робота бакалавра «Розробка онлайн–бібліотеки японських коміксів Ма́нґа за допомогою React та Node.JS», спеціальності 121 «Інженерія програмного забезпечення» / наук. керівник доцент, к.т.н., Ю.О. Лимаренко. Запоріжжя: ЗНУ. 2023. 75 с.

У роботі описано процес розробки системи, включаючи аналіз вимог, проектування архітектури, реалізацію та тестування. Розроблена система надає можливість користувачам зручно переглядати та читати японські комікси Ма́нґа в онлайн–режимі. Вона включає інтуїтивний інтерфейс, де користувачі можуть шукати комікси за назвою, автором, жанром або іншими критеріями.

Основна функціональність системи включає можливість переглядати сторінки коміксів, масштабування, збереження обраної сторінки та позначення вподобаних коміксів. Крім того, користувачі можуть залишати коментарі та оцінки до коміксів, що сприяє активному спілкуванню між читачами.Під час розробки системи було використано React для реалізації фронтенду, забезпечуючи швидкий і реактивний інтерфейс. Node.JS був використаний для побудови бекенду, забезпечуючи обробку запитів користувачів, збереження даних та забезпечення безпеки.

Результати дослідження показують, що розроблена онлайн–бібліотека японських коміксів Ма́нґа забезпечує зручний та ефективний спосіб читання та взаємодії з коміксами. Система може бути використана як основа для подальшого розвитку та розширення функціональності.

Ключові слова: *Манга, онлайн-бібліотека, база даних, клієнт сервер*

ABSTRACT

Pages - 75

Figures - 20

Tables - 24

Sources - 13

Bachelor's thesis "Development of the online library of Japanese comics Manga using React and Node.JS", specialty 121 "Software Engineering".

The paper describes the system development process, including requirements analysis, architecture design, implementation, and testing. The developed system allows users to conveniently view and read Japanese Manga comics online. It includes an intuitive interface where users can search for comics by title, author, genre, or other criteria.

The main functionality of the system includes the ability to browse pages of comics, zoom in, save the selected page, and mark favorite comics. In addition, users can leave comments and ratings on comics, which promotes active communication between readers.During the development of the system, React was used to implement the frontend, providing a fast and responsive interface. Node.JS was used to build the backend, providing user request processing, data storage, and security.

The results of the study show that the developed online library of Japanese comics by Manga provides a convenient and efficient way to read and interact with comics. The system can be used as a basis for further development and expansion of functionality.

Keywords: Manga, online library, database, client server

ЗМІСТ

ВСТУП

Актуальність теми

Читання книжок завжди було однієї з основ вільного проведення часу людьми, але в молодому поколінні зараз більше переважає іноземна культура. І саме через це зараз почали набувати популярність комікси.

У культурі проявляється цікавість до китайських та японських мультиплікаційних фільмів та серіалів. І у зв'язку з цим популярності стала набирати також комікси цих країн. І через те що таких коміксів дуже багато не всі вони активно друкуються, а створюються в електронній версії. Це більш вигідно тому, що не потрібно друкувати та платити видавництву. Проте не дуже зручно було б шукати комікси по певному автору та жанру в усьому Інтернеті тому створюються сайти, де всі вони можуть викладатися для зручного доступу користувачів.

На сьогодні багато подібних ПЗ вже запущені в мережі Інтернет і мати попит. В них міститься інформація про комікс, автора та дані про статус коміксу. Однак, далеко не на всіх є можливість додавання нових коміксів або можливість перегляду відразу різних жанрів, тому що частина сайтів вузько кваліфікована на певний тип або зовсім на певний комікс. Через велику кількість коміксів на сайтах часто може втрачатися інформація про перекладача, дані про комікс та відсутня можливість додання коміксі до свого списку. Тому, програмне забезпечення (ПЗ), що розробляється дозволить заощадити час при пошуку нових тайтлів, а також у будь–який час отримати необхідну інформацію про комікси, створення та реєстрації власної команди перекладачів та можливість додання коміксу до свого списку для отримання сповіщень. Наприклад, таку інформацію як: список і назву коміксів що нещодавно вийшли або список популярних, також дата останнього оновлення коміксу й частоту оновлень.

Це знижує витрати часу читачів та робить Мангу ще більш доступною для широкої аудиторії. Крім того, онлайн–бібліотеки Манга дозволяють читачам знайти та ознайомитися з менш відомими виданнями Манга, які

можуть бути складніше знайти у друкованому вигляді. Також, на онлайн–бібліотеках Манга можна знайти переклади Манга на різні мови, що дозволяє більшій кількості людей насолоджуватися цим видом мистецтва.

Таким чином, актуальним є створення програмного застосунку для читання японських коміксів перекладених українською мовою, що буде розміщувати тайтли й даватиме змогу в переводі тайтлів перекладачами.

Глосарій

Манга (англ. Manga) – японська графічна література, що включає комікси, які створюються в Японії, та відзначаються специфічним стилем малювання та розповідання історій.

Мангаки (англ. Mangaka) – автори манги, які створюють малюнки та розповідають історії за допомогою образів.

Тайтл (англ. Title) – назва манги, яка відображає тему та жанр манги.

Сенен (англ. Seinen) – жанр манги, який адресований чоловікам старшого віку, звичайно з високим рівнем бойових сцен та екшну.

Сьодзе (англ. Shoujo) – жанр манги, який адресований жінкам та дівчатам, зазвичай з високим рівнем романтики та емоцій.

Сейнен (англ. Seinen) – жанр манги, який адресований дорослим чоловікам, з частим зображенням життя в реальному світі.

Дзьосей (англ. Josei) – жанр манги, який адресований дорослим жінкам, з фокусом на відносинах та романтиці.

Шонен–ай (англ. Shounen–ai) – жанр манги, що зображує романтичні стосунки між хлопцями.

Шоджо–ай (англ. Shoujo–ai) – жанр манги, що зображує романтичні стосунки між дівчатами.

Мегакрус (англ. Magical Girl) – популярний жанр манги, що зображує героїнь з надздібностями та суперсили, що зазвичай використовують для захисту світу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Мета проєкту

Метою дипломного проєкту є підтримати розвиток культури, залучення як найбільше більше людей до розвитку перекладів коміксів та надання платформи для вільного власного розвитку та місця збору однодумців.

1.2 Цілі програмного забезпечення

ПЗ, що розроблюється, дозволить:

– полегшити взаємодію між читачем і творцем, який створює комікс;

– просування нових творців;

– залучити нових користувачів і підвищити популярність різних жанрів коміксів внаслідок вебдодатка, розташованого в Інтернеті;

– скоротити користувачам час на пошук однодумців.

1.3 Призначення розробки

Вебдодаток для перегляду та додавання коміксів призначений для перегляду та додавання коміксу будь–якого крім заборонених законодавством жанрів коміксів.

Додаток дозволяє створювати переглядати будь–які комікси і надавати користувачам, зареєстрованим на сайті, можливість додавання власних коміксів. Користувачами даного ПЗ є фірми редакторів, перекладачі, автори, а також звичайні люди які бажають провести час за читанням улюбленого твору. Цей додаток дозволить підвищити популярність коміксів і залучити більше авторів, перекладачів, читачів що прискорить і полегшить взаємодію між ними всіма.

1.4 Можливі розв'язання задачі

Стек PERN – це популярний стек для веброзробки, який складається з чотирьох технологій: PostgreSQL, Express, React та Node.js. PERN стек є варіантом стека MERN, де використовується PostgreSQL як реляційна база даних замість MongoDB.

Основні компоненти стека PERN включають:

1. PostgreSQL: Це реляційна система управління базами даних (RDBMS), яка забезпечує надійне зберігання та управління структурованими даними. PostgreSQL є потужним і розширюваним рішенням з відкритим вихідним кодом, яке підтримує SQL запити, транзакції, індекси та багато інших функцій, що дозволяють ефективно працювати з даними.

2. Express: Це легкий і гнучкий фреймворк для вебдодатків, який використовується для побудови бекенду програми. Express надає простий і зрозумілий спосіб створення маршрутів, обробки запитів та відповідей сервера. Він також підтримує різноманітні проміжне програмне забезпечення (middleware), яке дозволяє розширювати функціональність серверної частини додатка.

3. React: Це бібліотека JavaScript для побудови користувацьких інтерфейсів (UI), яка використовується для створення динамічних та інтерактивних фронтенд–додатків. React дозволяє розробникам створювати компоненти з логікою інтерфейсу, які можуть автоматично оновлюватись при зміні даних. Це полегшує розробку реактивних та швидких користувацьких інтерфейсів.

4. Node.js: Це середовище виконання JavaScript, яке дозволяє розробникам запускати JavaScript–код поза веббраузером. Node.js забезпечує серверну частину додатка, дозволяючи обробляти запити, здійснювати доступ до бази даних та виконувати різні серверні операції. Використовуючи Node.js, розробники можуть створювати повноцінні вебдодатки з однією мовою програмування – JavaScript.

За допомогою стеку PERN розробники можуть створювати масштабовані та потужні вебдодатки з реляційною базою даних. PostgreSQL забезпечує надійне зберігання та управління даними, Express надає можливості для побудови серверної частини, React дозволяє створювати динамічні фронтенд–інтерфейси, а Node.js забезпечує середовище виконання та підтримку серверних операцій.

Цей стек особливо підходить для проєктів, які вимагають роботи зі структурованими даними, складних запитів до бази даних та високою швидкодією. Використання реляційної бази даних може бути корисним для додатків зі складною моделлю даних та потребою відносин між різними сутностями.

Обидва стеки, MERN та PERN, є потужними та ефективними для створення сучасних вебдодатків, і вибір між ними залежить від конкретних потреб проєкту та вподобань розробника.

1.5 Вимоги замовника

Для більш детального визначення призначення розробляємогоПЗ далі у пунктах 1.5.1 та 1.5.2 описані, відповідно, функціональні та нефункціональні вимоги до ПЗ.

1.5.1 Функціональні вимоги

1.5.1.1 Можливість визначення рівня доступу користувача (Адміністратор, зареєстрований користувач, незареєстрований користувач).

1.5.1.2 Управління багатовіконним інтерфейсом користувач–комп'ютер.

1.5.1.3 Видача діагностичних повідомлень різного призначення.

1.5.1.4 Функції режиму «Адміністратор»:

– Управління базою даних:
– Робота з коміксами:
– редагування інформації про комікс;

– видалення коміксу з підтвердженням
– додавання та редагування жанрів;

1.5.1.5 Робота з користувачами сайту:

– блокування користувачів;
– редагування доступу;
– надання прав редагування коміксів;
– зміна прав користувача.

1.5.1.6 Функції режиму «Незареєстрований користувач»:

– реєстрація на сайті;
– реєстрація на виставці;
– вибір коміксу;
– перегляд інформації про комікс;
– перегляд коміксу;
– перегляд каталогу;
– використання пошуку.

1.5.1.7. Функції режиму «Зареєстрований користувач»:

– Авторизація на сайті.
– Вхід в профіль користувача.
– Введення і аналіз реєстраційних даних.
– Редагування особистих даних.
– Створення команди.
– Редагування інформації про команду.
– Вилучення інформації про команду.
– Додавання інших користувачів до команди.
– Додавання нового коміксу до сайту.
– Редагування інформації про комікс.
– Додавання нових глав до коміксу.
– Видалення коміксу.
– Передання прав на редагування коміксу.

– Додавання коміксу до списку свого профілю.
– Додавання коментарю до коміксу.
– Перегляд коміксів на сайті.
– Перегляд каталогу.

1.5.2 Нефункціональні вимоги

1.5.2.7. ПЗ повинно працювати на операційних системах Windows, починаючи з версії Windows 7.

1.5.2.8. ПЗ повинно мати мультимовний інтерфейс «Користувач–ПК».

1.5.2.9. ПЗ має видавати проміжні етапи виконання завдань у вигляді діагностичних повідомлень.

1.5.2.10. ПЗ має швидко відповідати на запити користувача (не більше 5 секунд).

1.5.2.11. ПЗ повинно коректно відображатися в браузерах IE, Opera, Chrome.

1.5.2.12. ПЗ повинно використовувати СУБД MySQL версії 5.1 або вище.

1.6 Аналіз готових рішень

Webtoons – популярна платформа цифрових коміксів, яка надає користувачам широкий вибір коміксів, включаючи романтику, комедію, екшн, драму тощо (див. рис 1.1).

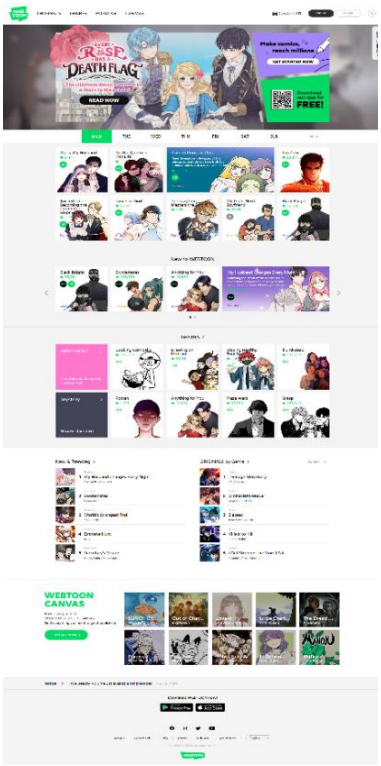

Рис *1.1 - головна сторінка Webtoon*

Тут є щось для кожного, і користувачі можуть легко переглядати сайт, щоб знайти комікси, які відповідають їхнім смакам. Він має направлення на англійську мовну аудиторію.

Сайт має інтуїтивно зрозумілий інтерфейс, що дозволяє користувачам легко орієнтуватися і знаходити те, що вони шукають. Комікси добре організовані, і користувачі можуть швидко переходити від одного розділу до іншого.

На відміну від багатьох інших платформ цифрових коміксів, Webtoons дозволяє користувачам отримати доступ до більшості своїх коміксів безплатно. Це робить його привабливим варіантом для тих, хто любить комікси, але не може дозволити собі їх купити. Він має велику та активну спільноту читачів і творців. Користувачі можуть взаємодіяти один з одним, ділитися улюбленими коміксами і навіть створювати власні комікси.

Найголовнішим мінусом цієї платформи є дозвіл будь–кому створювати і публікувати свої комікси, що може призвести до відсутності контролю якості.

Деякі комікси можуть мати погану ілюстрацію, сюжет або і те, і інше, що може відштовхнути деяких користувачів.

Загалом, Webtoons.com є чудовою платформою для цифрових коміксів. Велика колекція коміксів, зручний інтерфейс та активна спільнота роблять його популярним серед читачів. Однак відсутність контролю якості, обмежений офлайн–доступ і непослідовний графік оновлень можуть бути недоліком для деяких користувачів.

Manga in UA – це український вебсайт, який пропонує великий вибір манґи для онлайн–читання (див. рис 1.2).

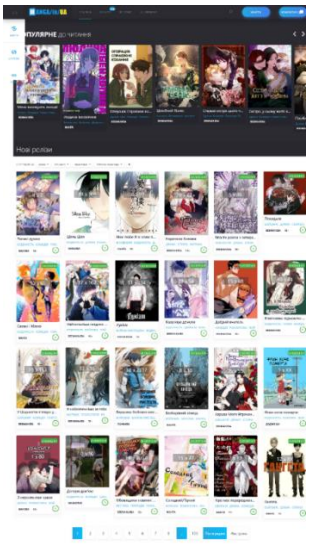

Рис *1.2 - головна сторінка Manga in UA*

Сайт має широкий вибір манґи з різних жанрів, таких як романтика, фентезі, екшн, драма, комедія та інші. Користувачі можуть знайти твори, які відповідають їхнім смакам і інтересам. Для тих, хто шукає манґу українською мовою, цей сайт є зручним варіантом. Він надає можливість насолоджуватися популярними японськими коміксами, перекладеними на українську. Однією з переваг сайту є те, що користувачам не потрібно платити за доступ до манґи. Вони можуть безплатно читати свої улюблені серії, що робить його доступним для більшої кількості людей. При цьому він має зручний інтерфейс, що дозволяє зручно переглядати манґу. Користувачі можуть легко шукати,

переходити між розділами та насолоджуватися читанням без непотрібних перешкод.

Оскільки сайт спеціалізується на українській манґі, деякі користувачі можуть бути розчаровані відсутністю перекладів для деяких популярних японських манґа. Це обмежує вибір доступних творів.

Сайт може мати обмежений графік оновлень нових глав. Це може призвести до відчуття нестабільності для тих, хто хоче бути в курсі найсвіжіших випусків. У порівнянні з деякими іншими манґа–сайтами, Manga in UA може бути обмежений в додаткових функціях, таких як можливість збереження манґи для офлайн–читання або взаємодія з іншими читачами.

Загалом, сайт Manga in UA є привабливим варіантом для українських шанувальників манґи, пропонуючи широкий вибір творів безплатно. Однак, його обмеженості у перекладах, оновленнях та додаткових функціях можуть становити недоліки для деяких користувачів.

Honey–manga – це сайт, який надає користувачам колекцію манґи та аніме–серіалів (див рис 1.3).

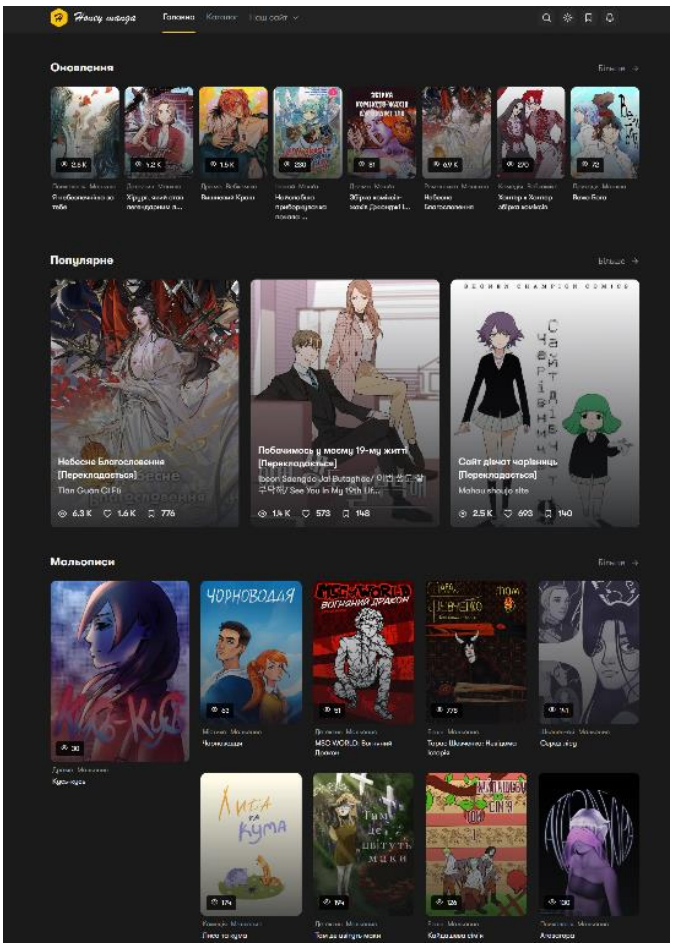

Рис *1.3* - *головна сторінка Honey Manga*

Це українська платформа, яка дасть можливість ознайомитися з величезну колекцію манга–серіалів, включаючи популярні та маловідомі назви. Сайт регулярно поповнюється новими випусками, що робить контент свіжим і цікавим.

Сайт надає безплатний доступ до більшості серій манги, він має мінімалістичний та зрозумілий інтерфейс, що дозволяє користувачам легко орієнтуватися і знаходити те, що вони шукають. Серії манги добре організовані, і користувачі можуть швидко переходити від однієї глави до іншої. Honey–manga має активну спільноту читачів та ентузіастів манги. Користувачі можуть взаємодіяти один з одним, ділитися своїми улюбленими манга–серіями й навіть рекомендувати нові для прочитання.

Хоча вебсайт пропонує деякі аніме–серіали, його колекція є відносно обмеженою порівняно з іншими платформами для потокового перегляду

аніме. Деякі манга–серії на сайті мають нерегулярний графік оновлення, що може розчаровувати читачів, які з нетерпінням чекають на наступну главу.

Загалом, Honey–manga.com.ua є гідною платформою для любителів манґи. Велика колекція манґи, зручний інтерфейс та активна спільнота роблять його привабливим для користувачів. Однак обмежений вибір, непослідовний графік оновлення та відсутність офіційних перекладів можуть сподобатися не всім користувачам.

Shinden – це польський вебсайт, який пропонує користувачам різноманітні аніме та манґа–серіали (див рис 1.4).

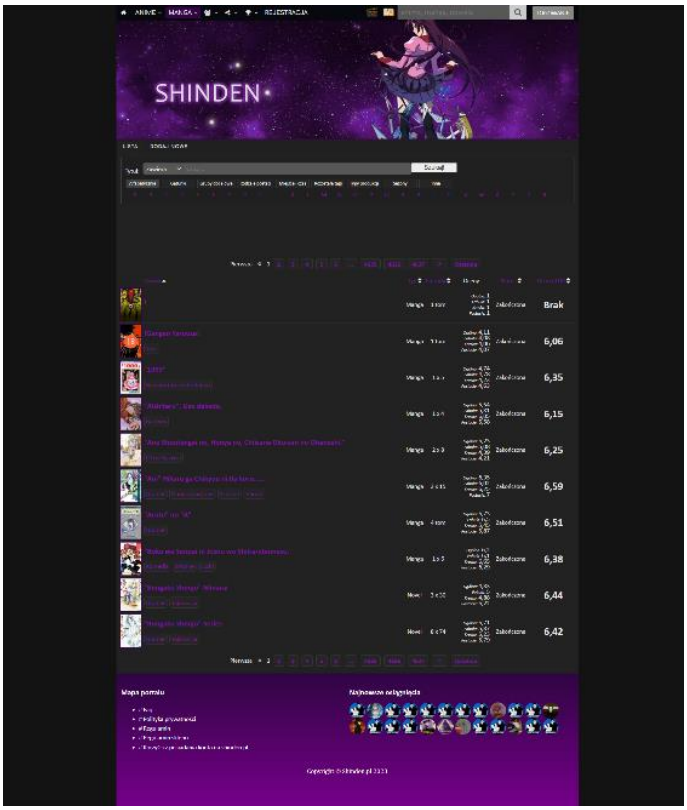

Рис *1.4 - головна сторінка Shinden*

Велика колекція аніме та манги: Shinden.pl пропонує користувачам величезну колекцію аніме та манґа–серіалів, включаючи популярні та маловідомі назви.

Сайт пропонує польські переклади багатьох аніме та манґа–серіалів, що є корисним для користувачів, які не говорять або не розуміють англійську чи японську мови.

Користувачі між собою діляться улюбленими серіалами і коментувати тайтли коміксів. Хоча вебсайт пропонує польські переклади багатьох аніме та манґа–серіалів, він може бути не дуже привабливим для користувачів, які віддають перевагу перегляду або читанню англійською мовою. Англомовний контент на сайті відносно обмежений. Вебсайт містить багато спливаючих рекламних оголошень, які можуть дратувати користувачів. Реклама також може сповільнювати роботу вебсайту й ускладнювати його використання.

Загалом, Shinden є гідною платформою для ентузіастів аніме та манги, які говорять або розуміють польську мову. Велика колекція аніме і манги, зручний інтерфейс і активна спільнота роблять її привабливою для користувачів. Однак обмежений англомовний контент, спливаюча реклама, відсутність офіційного контенту та непостійна якість зображення можуть сподобатися не всім користувачам.

1.7 Висновок з розділу 1

Вивчено вимоги користувача та вимоги до вебзастосунку для читання та додавання коміксів. Для аналізу цих функцій був виконаний повний опис програмного забезпечення, включаючи його функції, характеристики та обмеження. Під час аналізу були розглянуті аналоги програмного забезпечення, такі як "Honey Manga", "Manga in UA", "WebToon" та "Shinden". Виявлено, що ці програмні прототипи мають певні недоліки, включаючи незручну навігацію, обмежений вибір аніме–серіалів, спливаючу рекламу та нерегулярний графік оновлень. На основі цього аналізу було встановлено, що розроблене програмне забезпечення має актуальність на ринку. У результаті розроблено програмне забезпечення з такими функціями, як реєстрація, авторизація, перегляд каталогу коміксів, перегляд останніх оновлень коміксів,

перегляд коміксів, перегляд своїх закладок, додавання та видалення коміксів до закладок, пошук коміксів по назві, фільтр коміксів по жанрах та типу, перегляд сторінок профілю, перекладачів, сценаристів та художників, сортування коміксів по даті, редагування списків жанрів, сценаристів, типів, художників та перекладачів, редагування ролей користувачів. Таким чином, тема дипломного проєкту є актуальною, а розроблюване ПЗ потрібне на ринку й задовольняє вимоги користувачів.

2 ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ ОНЛАЙН БІБЛІОТЕКИ КОМІКСІВ

2.1 Обґрунтування вибору моделі архітектури вебзастосунку

Обґрунтування вибору моделі архітектури "клієнт–сервер" полягає в тому, що ця модель має багато переваг, які роблять її привабливою для розробки вебзастосунків.

По–перше, модель "клієнт–сервер" дозволяє розподілити функції та відповідальності між клієнтами і серверами. Це дозволяє зменшити навантаження на окремі компоненти системи та поліпшити продуктивність.

По–друге, ця модель є масштабованою, що означає, що систему можна легко масштабувати, додаючи нові сервери або розподіляючи навантаження між чинним сервером. Це дозволяє збільшити потужність системи та здатність обслуговувати більше клієнтів.

По–третє, модель "клієнт–сервер" дозволяє чітко розділити відповідальності між клієнтами й серверами. Сервери відповідають за зберігання та обробку даних, бізнес–логіку та забезпечення безпеки, тоді як клієнти відповідають за взаємодію з користувачем та представлення даних.

По–четверте, модель "клієнт–сервер" забезпечує високий рівень безпеки. Сервери можуть контролювати доступ до ресурсів та даних, проводити аутентифікацію користувачів та застосовувати заходи безпеки, такі як шифрування даних та захист від несанкціонованого доступу.

По–п'яте, ця модель спрощує розробку, тестування та сумісність програмного забезпечення. Розділення на клієнтську та серверну частини дозволяє використовувати різні технології та платформи для кожної з них, що допомагає вибрати найкращі рішення для кожної сторони.

Нарешті, модель "клієнт–сервер" підтримує розподілений доступ до даних та ресурсів. Це означає, що клієнти можуть отримувати доступ до даних

та послуг, що знаходяться на віддалених серверах, незалежно від їх фізичного розташування.

Враховуючи ці переваги, вибір моделі "клієнт–сервер" є обґрунтованим для розробки вебзастосунків.

2.2 Обґрунтування вибору мови програмування і середовища розробки

JavaScript є однією з найпопулярніших мов програмування, використовуваних для розробки вебдодатків. Вона є мовою програмування на зворотному боці, яка дозволяє створювати динамічний інтерактивний вміст на вебсторінку. React, з іншого боку, є відкритою бібліотекою JavaScript, яка дозволяє розробникам будувати ефективні і швидкі користувацькі інтерфейси.

Завдяки написанню кросплатформленного коду вебзастосунки можуть працювати на різних платформах. Що забезпечую широку аудиторію для додатка. При цьому

JavaScript є однією з найбільш поширених мов програмування веброзробки. Вона підтримується всіма сучасними браузерами та має велику спільноту розробників, що надає доступ до багатьох інструментів, бібліотек і фреймворків. При цьому JavaScript має багатий вибір бібліотек і фреймворків, які допомагають розробникам прискорити процес розробки та полегшують роботу з вебдодатками. Зокрема, React надає потужний набір інструментів для створення користувацьких інтерфейсів, таких як компоненти й віртуальний DOM.

Маючи досить велику спільноту розробників у світі JavaScript, як мова програмування стає все більш доступною для вивчення та застосування. Завдячуючи цьому JavaScript має велику підтримку від індустрії. Багато великих компаній надають інструменти, бібліотеки та документацію, які спрощують розробку на JavaScript і React. Крім того, є велика кількість

вакансій для розробників JavaScript, що забезпечує хороші можливості для кар'єрного зростання.

Враховуючи популярність JavaScript, потужність React і їх екосистему, кросплатформеність, підтримку спільноти розробників, а також оптимізацію продуктивності, вибір JavaScript з React для розробки вебдодатку є обґрунтованим. Це дозволить створити швидкий, ефективний та масштабований додаток з великою кількістю доступних ресурсів і підтримки індустрії.

2.3 Обґрунтування вибору СКБД для реалізації бази даних вебзастосунку з читання коміксів

Для вебзастосунку, що розробляється, необхідно встановити стабільний зв'язок до БД, де безпосередньо зберігаються дані, а також надати можливість змінювати ці дані, тому виникає потреба у якісній СКБД, яка б задовольняла ці потреби.

Розглянемо такі СКБД:

- MySQL;
- MS SQL;
- PostgreSQL;
- SQLite 3;
- MongoDB.

Для аналізування використано такі умови, де кожний критерій має свою вагу (де 5 це сама висока вага за рейтингом, а 1 це сама низька):

- надійність зв’язку з БД;
- безпека доступу до даних;
- простота інсталяції;
- швидкість опрацювання вхідних запитів;
- швидкість формування вихідного результату;
- кросплатформеність;

– сумісність з обраною мовою програмування (JavaScript).

Результат аналізування перелічених СКБД відображені у таблиці 2.1.

Таблиця *2.1*

Порівняльна таблиця СКБД

Критерії	СКБД				
	MySQL	MS SQL	Postgres SQL	SQLite 3	MongoDB
Надійність зв'язку з БД	5	5	5	5	5
Безпека доступу до даних	4	5	4	3	2
Простота інсталяції	2	5	3	5	5
Швидкість опрацювання вхідних запитів	5	5	5	4	4
Швидкість формування вихідного результату	5	4	5	4	5
Кросплатформеність	5	3	5	5	5
Сумісність з обраною мовою програмування (JavaScript)	4	3	4	2	5
Результат	**30**	**30**	**31**	**29**	**31**

Можна зробити висновок, що PostgresSQL та MongoDB на незначну кількість балів краще серед представлених кандидатів. Тому для розробки вебзастосунку було обрано PostgresSQL спираючись на кращу робочу навантаженість аналізу даних.

2.4 Схема навігації по сторінках клієнтської частини вебзастосунку з перегляду та додавання коміксів

Схема навігації – це структура клієнтської частини вебдодатку. В ній в вигляді дерева продемонстровані всі розділи підрозділи та сторінки системи

згруповані по різному меню. Ця карта дозволяє збудувати зручну інформаційну архітектуру продукту.

На рисунку 2.1 представлена схема навігації по сторінках клієнтської частини сайту для перегляду та додавання коміксів.

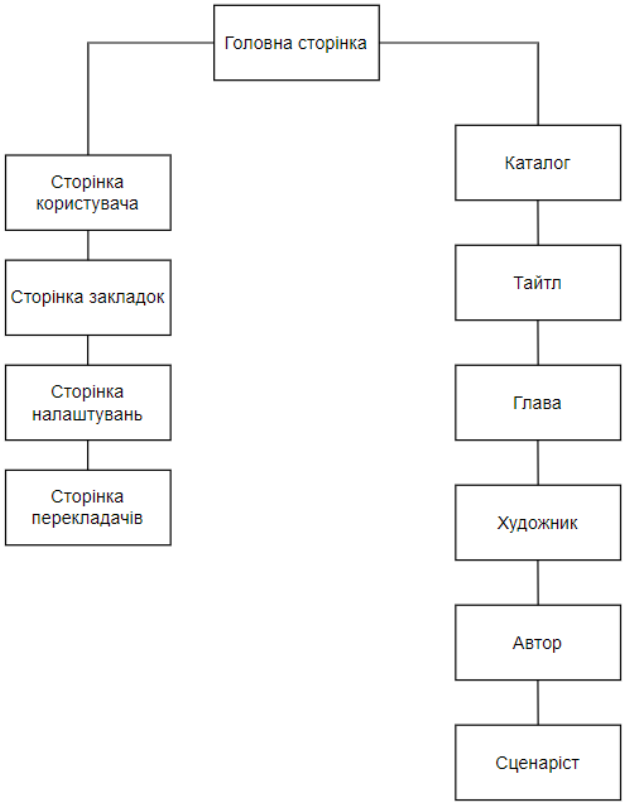

Рис *2.1 - Схема навігації по сторінках клієнтської частини сайту*

Клієнтська частина вебдодатка складається з наступних сторінок компонентів та вікон.

Верхня панель вебдодатка. Панель є доступною з будь–якої сторінки вебдодатка та має наступні посилання:

- реєстрація;
- авторизація;
- головна сторінка;
- каталог.

Сторінка профілю. Перехід на дану сторінку може виконуватись після реєстрації або авторизації користувача. Сторінка має наступні посилання:

- редагування профілю;
- закладки користувача;
- команда.

Головна сторінка вебдодатка. Перехід на дану сторінку можливий після натиску на емблему сайту на головній панелі. Дана сторінка має наступні елементи:

- слайдер новинок;
- найпопулярніші тайтли;
- типи коміксів;
- новини;
- гарячі новинки;
- контакти.

Сторінка коміксу. Перехід на дану сторінку можливий з вікна “Комікс” на головній сторінці або з каталогу вебдодатка.

Дана сторінка має наступні елементи:

- список глав;
- список перекладачів;
- список сценаристів;
- список художників;
- список жанрів;
- загальну інформацію.

В режимі адміністратора біля глав с списку присутні кнопки для додавання та редагування глав. А в режимі перекладача тільки редагування.

Сторінка глави коміксу. Перехід на дану сторінку можливий зі сторінки коміксу при натисканні на посилання “Глава №” в списку глав на сторінці коміксу.

Сторінка перекладачів. Перехід на дану сторінку можливий при натисненні на посилання “Список перекладачів” головної сторінки адміністратора. Сторінка списку перекладачів має посилання “Додати нового перекладача”, а також список перекладачів коміксів на сайті. Під кожним перекладачем присутні такі елементи:

– посилання “Редагувати перекладачів”;

– кнопка “Видалити перекладачів”.

Сторінка сценаристів. Перехід на дану сторінку можливий при натисненні на посилання “Сценарист:” сторінки тайтлу. Сторінка сценаристів має повну інформацію про сценариста.

Сторінка художників. Перехід на дану сторінку можливий при натисненні на посилання “Художник:” сторінки тайтлу. Сторінка художників має повну інформацію про художників.

Сторінка перекладача. Перехід на дану сторінку можливий при натисненні на посилання “Перекладач” зі сторінки коміксу. Сторінка перекладача містить каталог коміксів з посиланням на кожен з них.

2.5 Бібліотеки проєкту

Завдяки прогресу в галузі веброзробки, розробники отримали доступ до потужних інструментів, які дозволяють створювати дивовижні інтернет–проєкт з більшою легкістю й ефективністю. Вони дедалі стають незамінними помічниками для створення власної інтернет–бібліотеки. Ці бібліотеки – Tailwind CSS, Swiper Slider і Axios.

Tailwind CSS – це потужна CSS–бібліотека, яка надає широкий набір готових компонентів та стилів. Один з головних принципів Tailwind CSS – це "utility–first" підхід, що означає, що можливо швидко та легко використовувати короткі класи для застосування стилів безпосередньо до елементів HTML. Це дозволяє значно прискорити розробку і зробити код

більш зрозумілим. Tailwind CSS також має потужну систему налаштувань, яка дозволяє налаштовувати стилі під свої потреби.

Swiper Slider, яка дозволяє створювати виняткові та інтерактивні слайдери для вашої інтернет–бібліотеки. Swiper Slider надає гнучкість і простоту використання. Можливо легко налаштувати вигляд слайдера, включити анімацію, додати кнопки навігації та багато іншого. Завдяки своїм можливостям та легкості використання, Swiper Slider допомагає створити привабливий та зручний інтерфейс для інтернет–бібліотеки.

Axios – це бібліотека для здійснення HTTP–запитів з JavaScript. Вона надає зручний і простий інтерфейс для виконання запитів до сервера та обробки отриманих даних. Axios підтримує всі основні методи HTTP, такі як GET, POST, PUT, DELETE, та дозволяє передавати параметри запиту, заголовки та багато іншого. Ви можете використовувати Axios, щоб отримувати дані з вашого сервера та взаємодіяти з API для отримання потрібної інформації для вашої інтернет–бібліотеки.

Загалом, бібліотеки Tailwind CSS, Swiper Slider і Axios є потужними інструментами, які допоможуть зробити інтернет–бібліотеку винятковою і високофункціональною. З їх допомогою можливо швидко створити привабливий дизайн, зручний інтерфейс та взаємодіяти з сервером для отримання необхідної інформації. Використання цих бібліотек дозволить вам ефективно реалізувати вашу ідею і надати користувачам незабутній досвід використання вашої інтернет–бібліотеки.

2.6 Побудова діаграми варіантів використання

Діаграма варіантів використання (use case diagram) у мові моделювання UML є важливим інструментом для аналізу, проєктування та спілкування щодо функціональної поведінки системи.

Основна мета діаграми варіантів використання полягає у визначенні та описі того, як система буде використовуватися її акторами або зовнішніми

сутностями. Актори можуть бути користувачами системи, зовнішніми системами, апаратними пристроями або будь–якими іншими сутностями, що мають взаємодію з системою. Варіанти використання визначають конкретні дії та функціональні можливості системи, які вона надає своїм акторам.

Основні переваги використання діаграм варіантів використання:

1. *Зрозумілість та комунікація*: Діаграми варіантів використання створюють чіткий та зрозумілий зв'язок між розробниками системи, замовниками та користувачами. Вони служать засобом візуалізації функціональних вимог та сприяють ефективній комунікації між всіма зацікавленими сторонами.

2. *Аналіз вимог*: Діаграми варіантів використання дозволяють ідентифікувати потенційні варіанти використання системи та вимоги до неї. Вони допомагають уточнити та узгодити функціональні можливості системи перед переходом до подальших етапів розробки.

3. *Орієнтація на користувача*: Діаграми варіантів використання дозволяють зосередитися на потребах користувачів та спрямовують розробку системи на задоволення їх вимог. Вони допомагають розробникам зрозуміти, як користувачі будуть взаємодіяти з системою та які функції їм потрібні.

4. *Підтримка розподіленої розробки*: Діаграми варіантів використання можуть бути використані як основа для документації та спілкування між розробниками, які працюють на різних етапах або в різних командних групах. Вони допомагають забезпечити єдність розуміння функціональності системи серед всіх учасників проєкт.

5. *Підтримка аналізу та проєктування*: Діаграми варіантів використання можуть бути використані як основа для подальшого аналізу, проєктування та моделювання системи на більш детальних рівнях. Вони служать вихідною точкою для створення логічних та фізичних моделей системи, які враховують деталі реалізації та технічні аспекти.

Враховуючи всі ці переваги, діаграми варіантів використання є важливим інструментом для аналізу, проєктування та документування систем.

Вони допомагають розробникам отримати ясне розуміння функціональних вимог та потреб користувачів, що сприяє успішній реалізації програмного продукту.

Складені діаграми варіантів використання для незареєстрованого користувача (рис. 2.2), зареєстрованого користувача (рис. 2.3) та адміністратора (рис. 2.4) вебдодатку для читання коміксів.

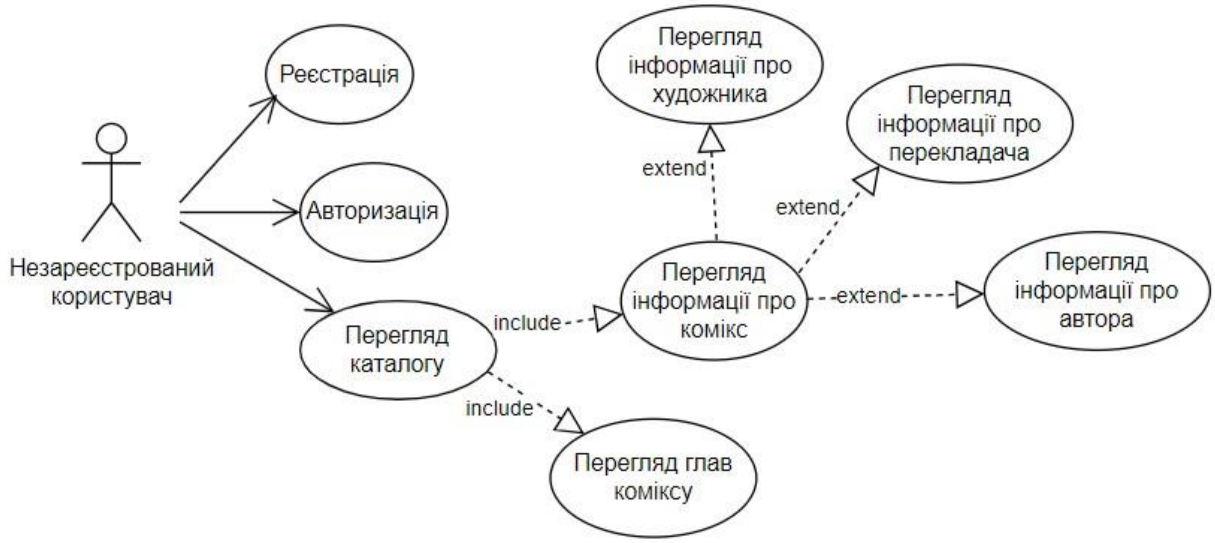

Рис *2.2 – Діаграма варіантів використання для незареєстрованого користувача*

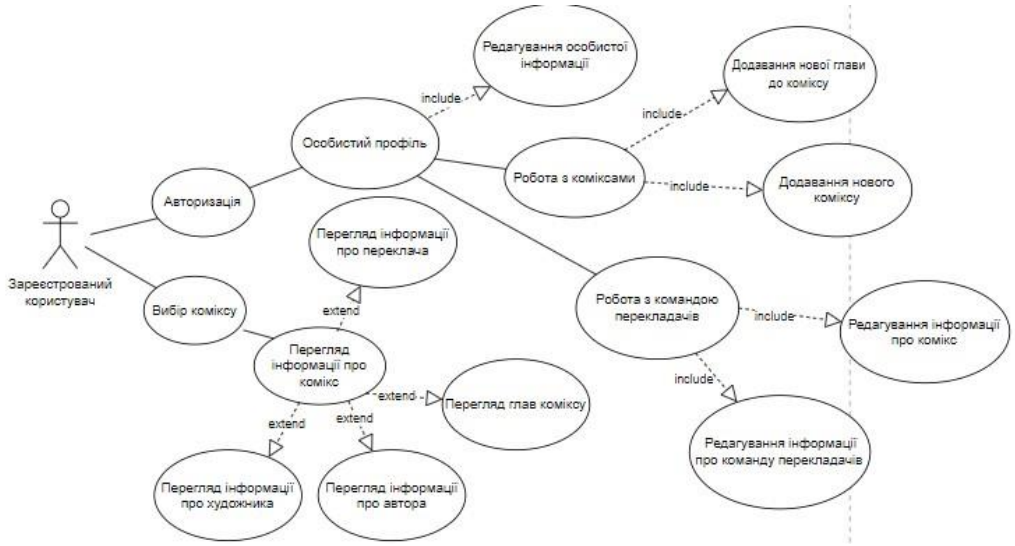

Рис *2.3 – Діаграма варіантів використання для зареєстрованого користувача*

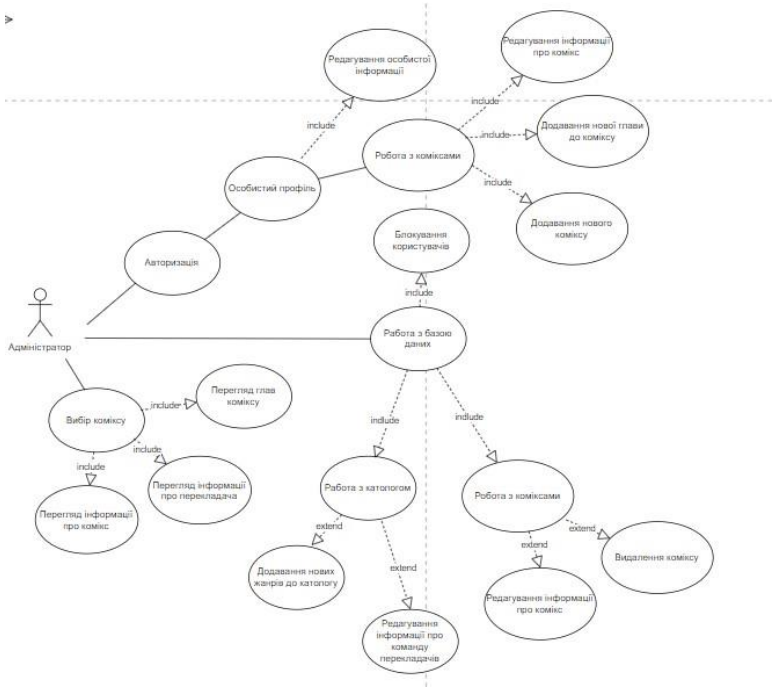

Рис *2.4 – Діаграма варіантів використання для адміністратора*

2.7 Специфікація варіантів використання

Специфікація варіантів використання представлена у вигляді таблиць з описом прецедентів та сценаріїв використання.

Опис сценарію для функції «Перегляд інформації про комікси» наведено в таблиці 2.1

Таблиця *2.1*

Опис прецеденту «Перегляд інформації про комікси»

Ідентифікатор	UC–2
Назва	Перегляд інформації про комікси.
Учасники	Незареєстрований користувач, Зареєстрований користувач, Адміністратор.
Опис	Перегляд інформації про комікси.
Передумова	Вибір коміксу.
Після умова	Отримання інформації про комікс.
Основний потік подій	Користувач вибирає комікс зі списку.
Пріоритет	Низький

Опис сценарію для функції «Реєстрація» наведено в таблиці 2.2

Таблиця *2.2*

Опис прецеденту «Реєстрація»

Ідентифікатор	UC–1
Назва	Реєстрація.
Учасники	Незареєстрований користувач.
Опис	Реєстрація користувача.
Передумова	Відображення форми реєстрації при натисканнях кнопки «Реєстрація»
Після умова	Занесення даних про користувача в базу даних, створення акаунта користувача, перехід в особистий кабінет користувача.
Основний потік подій	1. Користувач вводить інформацію (Логін, Пароль, Підтвердження пароля, Адреса e–mail). 2. Користувач натискає кнопку «Зареєструватися».
Альтернативний потік А	Якщо логін введений невірно або дане поле пусте: Висновок відповідного повідомлення.
Альтернативний потік В	Якщо пароль введений невірно або дане поле пусте: Висновок відповідного повідомлення.
Альтернативний потік З	Якщо підтвердження пароля збігається з паролем або дане поле пусте: Висновок відповідного повідомлення.
Пріоритет	Середній

Опис сценарію для функції «Перегляд глави коміксу» наведено в таблиці 2.3

Таблиця *2.3*

Опис прецеденту «Перегляд глави коміксу»

Ідентифікатор	UC–3
Назва	Перегляд глави коміксу.
Учасники	Незареєстрований користувач, зареєстрований користувач, адміністратор.
Опис	Перегляд глави коміксу.
Передумова	Відображення списку глав у вкладці обраного коміксу або натискання кнопки для початку перегляду.
Після умова	Отримання інформації про главу коміксу.
Основний потік подій	Користувач вибирає главу зі списку
Альтернативний потік А	Якщо глави відсутні: Неактивна кнопка перегляду глав та відсутність списку глав.
Пріоритет	Високий

Опис сценарію для функції «Перегляд глави перекладачів» наведено в таблиці 2.4

Таблиця *2.4*

Опис прецеденту «Перегляд команди перекладачів»

Ідентифікатор	UC–4
Назва	Перегляд команди перекладачів.
Учасники	Незареєстрований користувач, зареєстрований користувач, адміністратор.
Опис	Перегляд команди перекладачів.
Передумова	Вибір команди зі списку.
Після умова	Перехід до профілю команди перекладачів.
Основний потік подій	Отримання інформації про команду.
Пріоритет	Середній

Опис сценарію для функції «Авторизація» наведено в таблиці 2.5

Таблиця *2.5*

Опис прецеденту «Авторизація»

Ідентифікатор	UC–5
Назва	Авторизація.
Учасники	Зареєстрований користувач, адміністратор.
Опис	Авторизація користувача.
Передумова	Відображення форми авторизації при натисканні кнопки «Авторизація»
Після умова	Ініціалізація даних про користувача, перехід в особистий кабінет користувача.
Основний потік подій	1. Користувач вводить інформацію (Логін, Пароль). Користувач натискає кнопку «Увійти».
Альтернативний потік А	Якщо логін введений невірно або дане поле пусте: 2. Висновок відповідного повідомлення.
Альтернативний потік В	Якщо пароль введений невірно або дане поле пусте: Висновок відповідного повідомлення.
Пріоритет	Високий

Опис сценарію для функції «Команда перекладачів» наведено в таблиці 2.6

Таблиця *2.6*

Ключовий прецедент для таблиці «Команда перекладачів»

Ідентифікатор	UC–14
Назва	Команда перекладачів.
Учасники	Адміністратор.
Опис	Робота з таблицею команди.
Передумова	Перехід на таблицю «Команди».
Після умова	Відображення таблиці «Команди».
Основний потік подій	1. Перехід в адміністративну панель. 2. Вибір таблиці «Команд» і перехід на неї.
Пріоритет	Середній

Опис сценарію для функції «Редагування особистої інформації» наведено в таблиці 2.7

Таблиця *2.7*

Опис прецеденту «Редагування особистої інформації»

Ідентифікатор	UC–6
Назва	Редагування особистої інформації.
Учасники	Зареєстрований користувач, адміністратор.
Опис	Редагування особистої інформації користувача в особистому кабінеті.
Передумова	Відображення форми особистих даних при натисканні вкладки «Редагувати особистих даних»
Після умова	Зміна особистих даних користувача в БД.
Основний потік подій	1. Користувач редагує інформацію (E–mail, Логін, Стать, Пароль, Підтвердження пароля, Фотографія користувача). 2. Користувач натискає кнопку «Зберегти».
Альтернативний потік А	Якщо ім'я введено невірно або дане поле пусте: Висновок відповідного повідомлення.
Альтернативний потік В	Якщо пароль введений невірно або дане поле пусте: Висновок відповідного повідомлення.
Пріоритет	Високий

Опис сценарію для функції «Глави» наведено в таблиці 2.8

Таблиця *2.8*

Ключовий прецедент для таблиці «Глави»

Ідентифікатор	UC–15
Назва	Глави.
Учасники	Адміністратор.
Опис	Робота з таблицею глави.
Передумова	Перехід на таблицю «Глави».
Після умова	Відображення таблиці «Глави».
Основний потік подій	1. Перехід в адміністративну панель. 2. Вибір таблиці «Глави» і перехід на неї.
Пріоритет	Середній

2.8 Діаграма класів (class diagram)

Діаграма класів – це тип діаграм, які частіше за все використовуються при моделюванні об’єктно–орієнтованих систем і служить для подання статичної структури моделі системи в термінології класів об'єктно–орієнтованого програмування. Діаграма класів може відображати, зокрема, різні взаємозв'язки окремих сутностей предметної області, такими, як об'єкти і підсистеми, а також описує їх внутрішню структуру і типи відношень.

На рисунках 2.5–2.6 відображена діаграма класів для всієї системи в цілому.

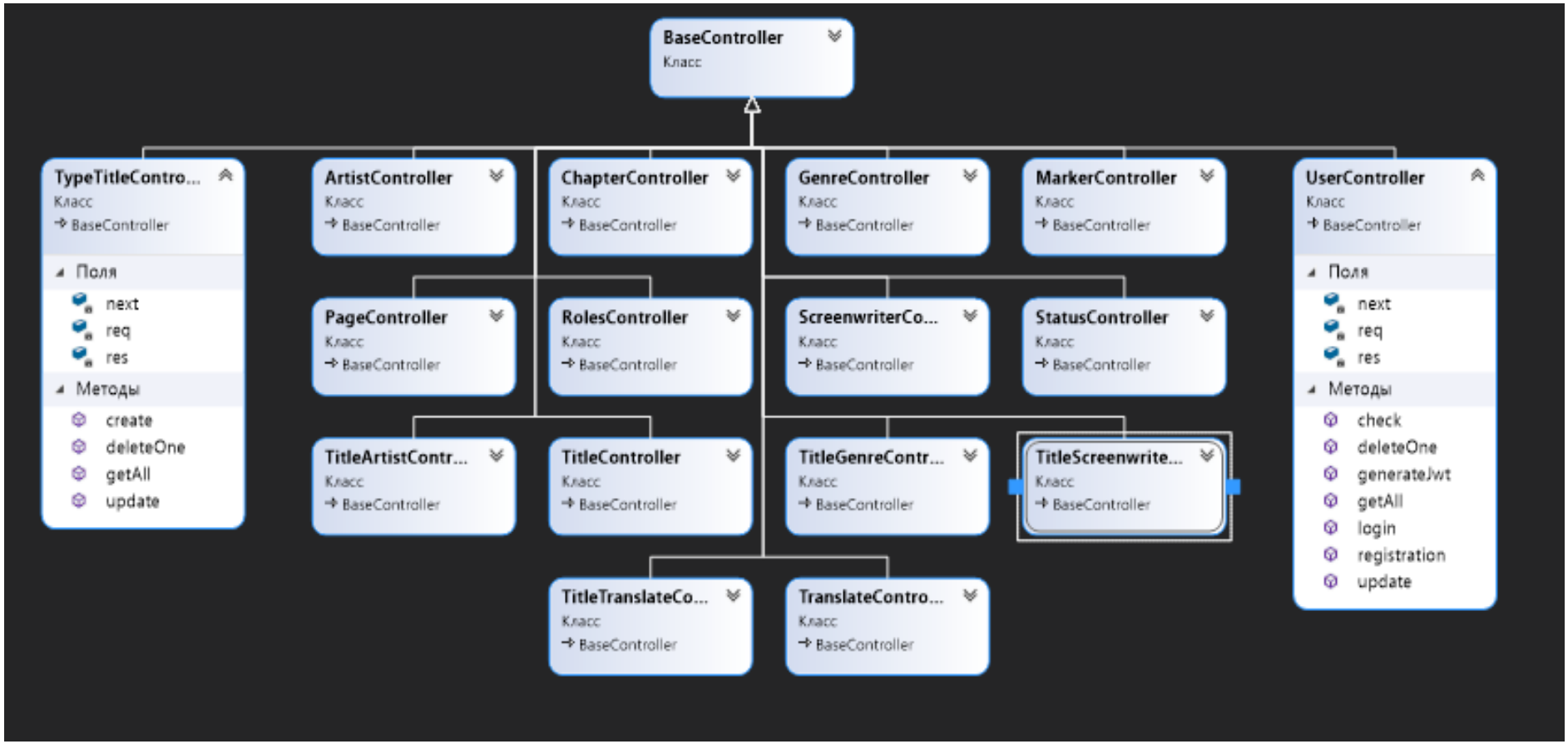

Рис 2.5 - *Діаграма класів контролерів вебсервісу*

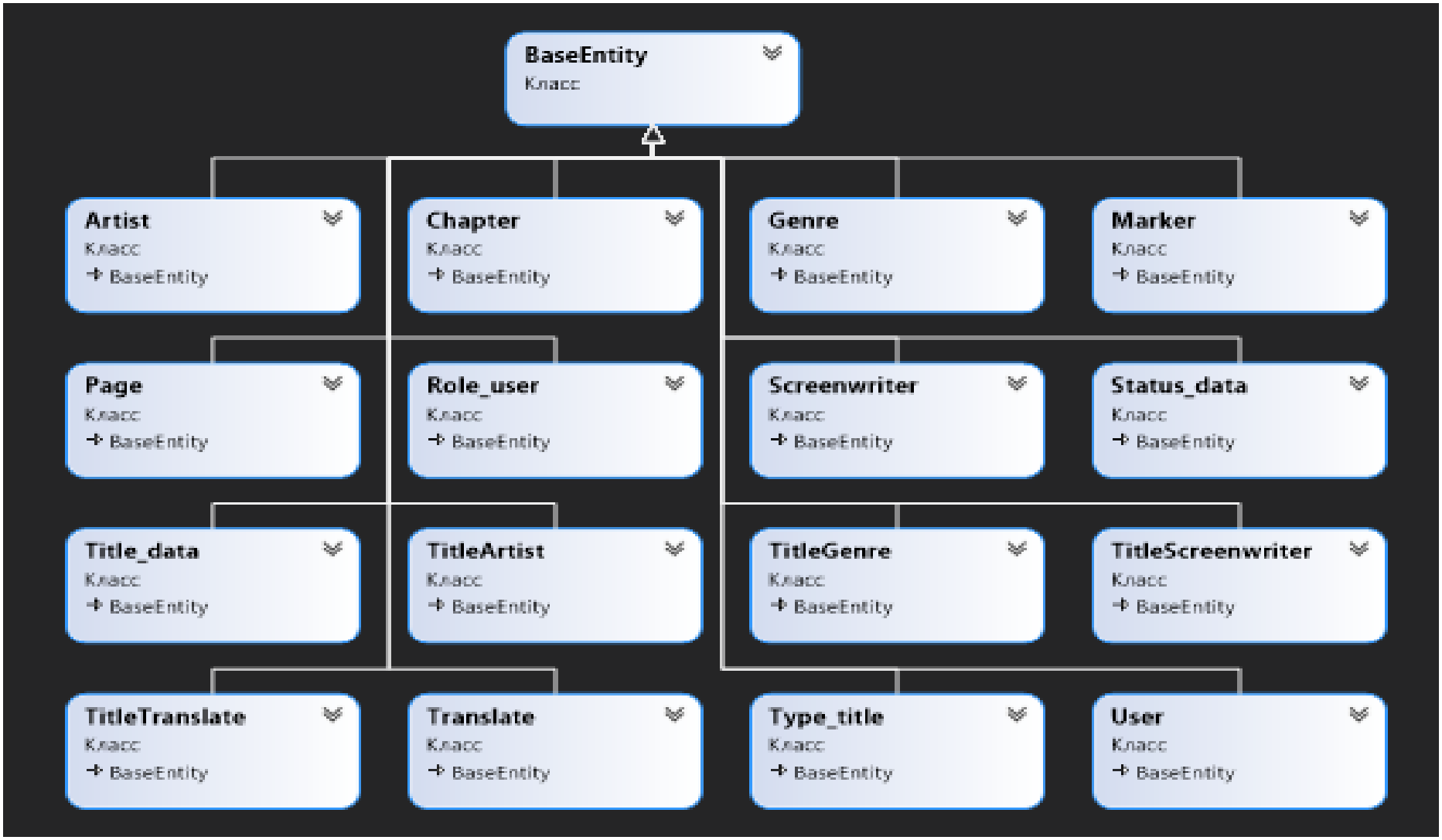

Рис 2.6 - *Діаграма класів основних сутностей вебсервісу*

2.9 Розробка макета вебдодатка

Згідно з вимогами користувача були розробленні макети сторінок вебдодатка для читання коміксів. Для розробки макетів сайту було вирішено використати за стосунок Figma.

Figma – це популярний вебінструмент для дизайну та створення прототипів, який використовується дизайнерами та командами для створення користувацьких інтерфейсів, вебсайтів та мобільних додатків. Деякі з його основних функцій включають співпрацю в режимі реального часу, інструменти векторного редагування та універсальну систему плагінів.

Макет головної сторінки в режимі “Гість” представлено на рисунку 2.7.

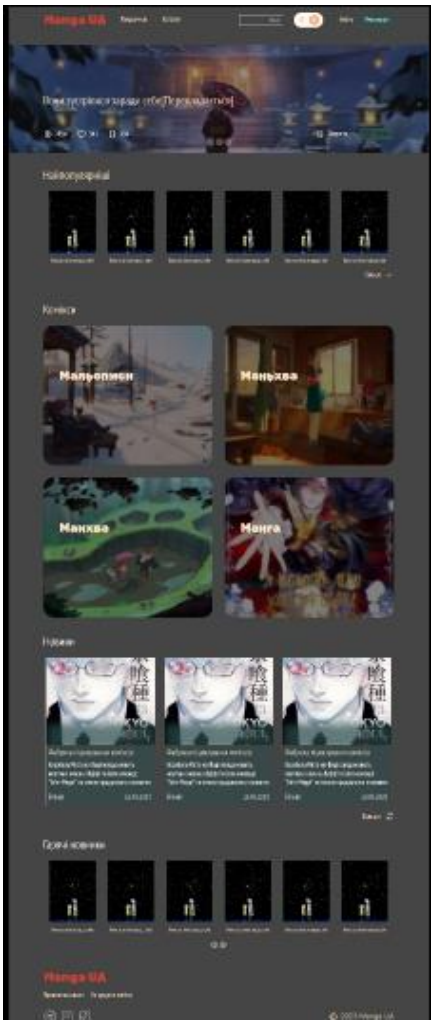

Рис *2.8 – Макет головної сторінки сайту*

Макети інших сторінок Вебдодатку представлені в Додатку Б.

2.10 Висновок з розділу 2

Під час виконання цього розділу була проведена аргументація та розробка архітектури веб-сайту з використанням підходу "Клієнт-Сервер". Також було побудовано діаграми класів.

Для подальшої реалізації веб-сайту були вирішені питання, пов'язані з вибором мови програмування, інструментальних засобів та системи керування базами даних (СКБД). За результатами вибору були обрані наступні технології:

- мова програмування: JavaScript;
- середовище розробки: Visual Studio Code;
- СКБД: PostgresSQL;

При проектуванні бази даних були розроблені ER-модель, логічна та фізична модель. Це допомогло виявити основні сутності предметної області, зв'язки між ними та їх фізичні властивості з точки зору типів даних.

Для наглядності були побудовані діаграми послідовностей для функцій:

- перегляд списку коміксів;
- авторизація користувача;
- створення нового коміксу;
- додавання жанрів;
- видалення коміксу;
- блокування користувача і т.д.

Як результат виконання даного розділу, була реалізована макетна частина вебдодатку.

3 РОЗРОБКА ВЕБЗАСТОСУНКУ ОНЛАЙН БІБЛІОТЕКИ КОМІКСІВ

3.1 Проєктування моделі даних бази даних вебзастосунку з читання коміксів

PostgresSQL – це широко розповсюджена система керування базами даних з відкритим початковим кодом. Вона є об’єктно–реляційною базою даних, що означає, що дані зберігаються у вигляді таблиць зі зв'язками між ними, але з об’єктно–орієнтованою моделлю, що означає підтримку об’єктів, класів та наслідування в схемі даних та мові запитів. PostgresSQL забезпечує надійне, швидке та ефективне збереження та доступ до даних.

PostgresSQL працює з клієнт–серверною архітектурою у поєднанні з JavaScript за допомогою Sequelize.

Sequelize — це ORM (Object–Relational Mapping — об’єктно–реляційне відображення або перетворення) для роботи з такими СУБД, як Postgres, MySQL, MariaDB, SQLite і MSSQL. Це далеко не єдина ORM для роботи з названими базами даних, але, вона є однією із самих просунений і перевірених часом ORM.

Коли вебдодаток виконує клієнтський запит відбуваються наступні дії:

– Встановлення з'єднання: Початковим кроком є створення з'єднання між вебдодатком та сервером бази даних;

– Клієнтська сторона формулює запит до бази даних.

– Після підготовки запиту додаток за допомогою Node.js виконує посилання запиту на сервер бази даних PostgresSQL для обробки;

– Після виконання запиту сервер бази даних PostgresSQL повертає результати до вебдодатка. Запит повертає копію об’єкта з якого може отримуватись вся необхідна інформація;

Реляційна модель PostgresSQL для побудови вебсайту для перегляду та додавання коміксів є обґрунтованим з кількох причин.

Модель гарантує надійність та цілісність даних. Всі операції збереження, оновлення та видалення виконуються успішно та безпечно, забезпечуючи надійність та точність інформації про комікси та групи працюючої над ним. Використання такої моделі дозволяє забезпечити ізольоване середовище для виконання транзакцій. Це означає, що операції, які вносять зміни до даних, не конфліктують з іншими процесами або користувачами, забезпечуючи безпеку та стабільність даних.

Крім того, вона забезпечує стійкість даних, оскільки дані зберігаються на постійному сховищі, такому як жорсткий диск або флеш–пам'ять. Це дозволяє відновлювати дані в разі відмови системи або перезавантаження.

Загалом, вибір реляційної моделі PostgresSQL для вебсайту для перегляду та додавання коміксів забезпечує надійність, безпеку та стабільність даних, а також забезпечує ефективну роботу з даними в поєднанні з JavaScript технологіями.

3.2 Розроблення ER–діаграми бази даних вебзастосунку

Модель сутність–зв'язок (ER–модель) – модель даних, яка дозволяє описати концептуальні схеми предметної області.

ER–модель використовується при високорівневому (концептуальному) проєктуванні БД. З її допомогою можна виділити ключові сутності та визначити зв’язки, які можуть встановлюватись між цими сутностями.

Опис сутностей ER–моделі вебдодатку по перегляду та додаванню коміксів представлено в таблиці 3.1.

Таблиця *3.1*

Опис сутностей ER–моделі даних БД вебдодатка

№	Название	Описание
1	user_data	В даній сутності представлена інформація про авторизованого користувача.
2	title_data	Дана сутність містить інформацію про комікс.
3	chapter_data	Дана сутність містить інформацію про главу коміксу.
4	page_data	Дана сутність містить інформацію про сторінку глави коміксу.
5	role_user	Дана сутність містить інформацію про ролі користувачів.
6	artist_data	Дана сутність містить інформацію про художників.
7	genre_title	Дана сутність містить інформацію про жанри.
8	screenwriter_data	Дана сутність містить інформацію про сценаристів.
9	status_data	Дана сутність містить інформацію про статус коміксу.
10	type_title	Дана сутність містить інформацію про тип коміксу.
11	translate_data	Дана сутність містить інформацію про перекладачів.
12	marker_user_data	Дана сутність містить інформацію про закладки користувача.

Зв’язок є логічним відношенням між сутностями.

Можливо встановити ідентифікаційний зв’язок «один–до–багатьох», зв’язок «багато–до–багатьох» та не ідентифікаційний зв’язок «один–до–одного».

Розрізняють залежні та незалежні сутності. Тип сутності визначається її зв’язком з іншими сутностями.

Що ідентифікує зв’язок встановлюється між незалежною (батьківська сторона зв’язку) та залежною (дочірній кінець зв’язку) сутностями.

При встановленні ідентифікаційного зв'язку атрибути первинного ключа батьківської сутності автоматично переносяться до первинного ключа дочірньої сутності. Ця операція доповнення атрибутів дочірньої сутності під час створення зв'язку називається міграцією атрибутів. У дочірній сутності нові атрибути позначаються як ключ (FK).

При встановленні не ідентифікаційного зв'язку дочірня сутність залишається незалежною, а атрибути первинного ключа батьківської сутності мігрують до складу неключових компонентів дочірньої сутності. Не ідентифікаційний зв'язок служить зв'язуванням незалежних сутностей.

Ідентифікаційний зв'язок показується на діаграмі суцільною лінією з жирною точкою на дочірньому кінці зв'язку, що не ідентифікує – пунктирний.

Кожна сутність може взаємодіяти з іншою унікальною сутністю, утворюючи тим самим зв'язок. Існує 3 типи зв'язків між сутностями:

– «один–до–одного» (у такому зв'язку сутності з однією роллю завжди відповідає не більше однієї сутності з іншою роллю);

– «один–до–багатьох» (коли сутності з однією роллю може відповідати будь–яке число сутностей з іншою роллю);

– «багато–до–багатьох» (у цьому випадку кожна з асоційованих сутностей може бути представлена будь–якою кількістю екземплярів).

В ER–моделі на рисунку 2.5 представлені наступні зв’язки:

– ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “role_user” та дочірньою “user_data” (оскільки користувачі не можуть існувати без ролі);

– ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “user_data” та дочірньою “marker_user_data” (оскільки закладки не можуть існувати без користувача);

– ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “ title_data” та дочірньою “ chapter_data” (оскільки глави не можуть існувати без коміксу);

– ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “ chapter_data” та дочірньою “ page_data” (оскільки сторінки не можуть існувати без глави);

– ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “ type_title” та дочірньою “ title_data ” (оскільки комікси не можуть існувати без типу);

– ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “ status_data” та дочірньою “ title_data ” (оскільки комікси не можуть існувати без статусу);

– ідентифікаційний зв'язок «багато–до–багатьох» між батьківською сутністю “genre_title” та дочірньою “ title_data ” (оскільки комікси не можуть існувати без жанрів);

– ідентифікаційний зв'язок «багато–до–багатьох» між батьківською сутністю “ translate_data” та дочірньою “ title_data ” (оскільки комікси не можуть існувати без перекладачів);

– ідентифікаційний зв'язок «багато–до–багатьох» між батьківською сутністю “ screenwriter_data” та дочірньою “ title_data ” (оскільки комікси не можуть існувати без сценаристів);

– ідентифікаційний зв'язок «багато–до–багатьох» між батьківською сутністю “ artist_data” та дочірньою “ title_data ” (оскільки комікси не можуть існувати без художників);

– не ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “ title_data ” та дочірньою “marker_user_data” (оскільки закладки не можуть існувати без коміксів);

– не ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “translate_data” та дочірньою “user_data” (оскільки перекладачі не можуть існувати без користувачів);

– не ідентифікаційний зв'язок «один–до–багатьох» між батьківською сутністю “ title_data ” та дочірньою “ translate_data ” (оскільки комікси не можу існувати без перекладачів).

На рисунку 3.1 представлена ER–модель БД вебдодатку для перегляду та додавання коміксів.

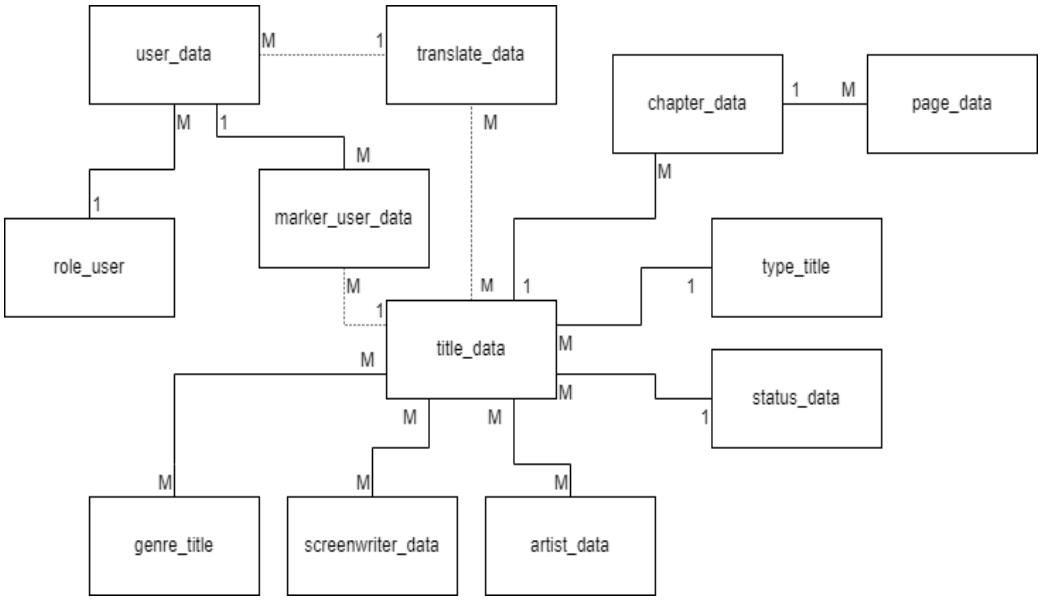

Рис *3.1 - ER–модель БД вебдодатку для перегляду та додавання коміксів*

3.3 Алгоритми роботи додатка

З метою створення діаграми послідовності було виокремлено наступні функції:

- перегляд списку коміксів;
- авторизація;
- додавання коміксу;

Відповідні діаграми послідовностей зображені на рисунках 3.2–3.4.

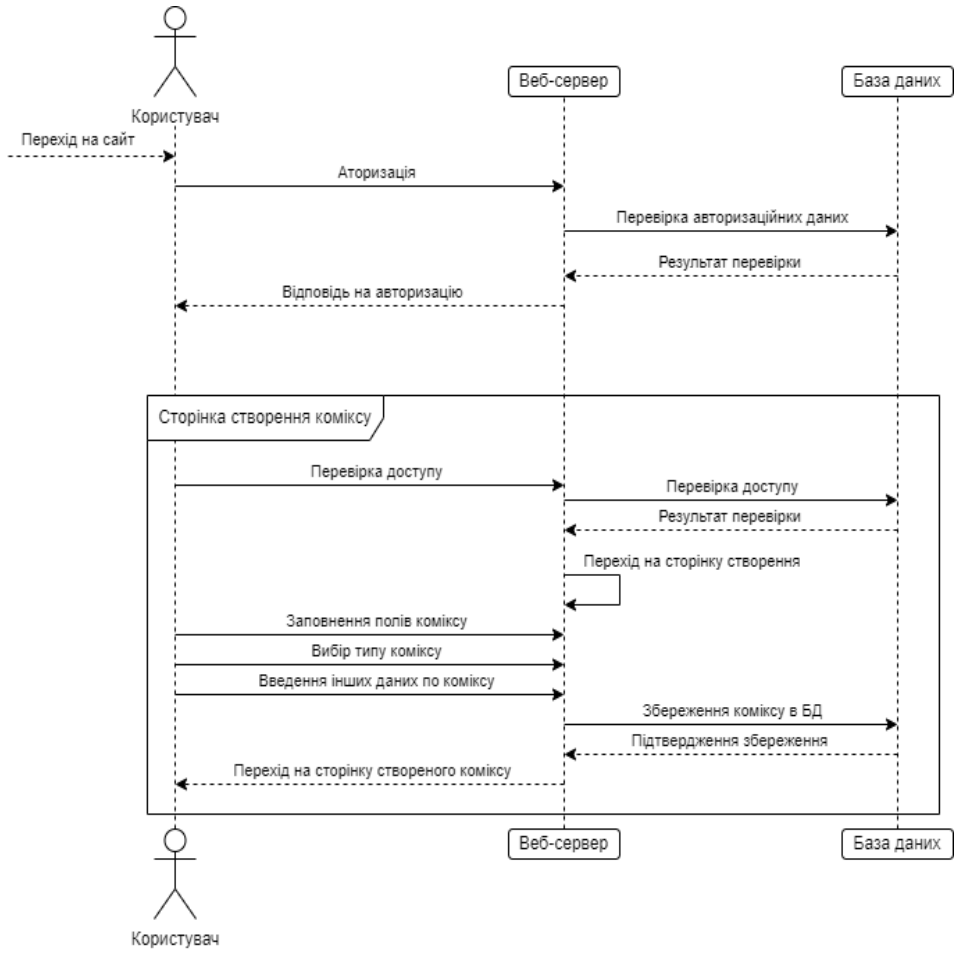

Рис *3.2 – Діаграма послідовності процесу додавання коміксу*

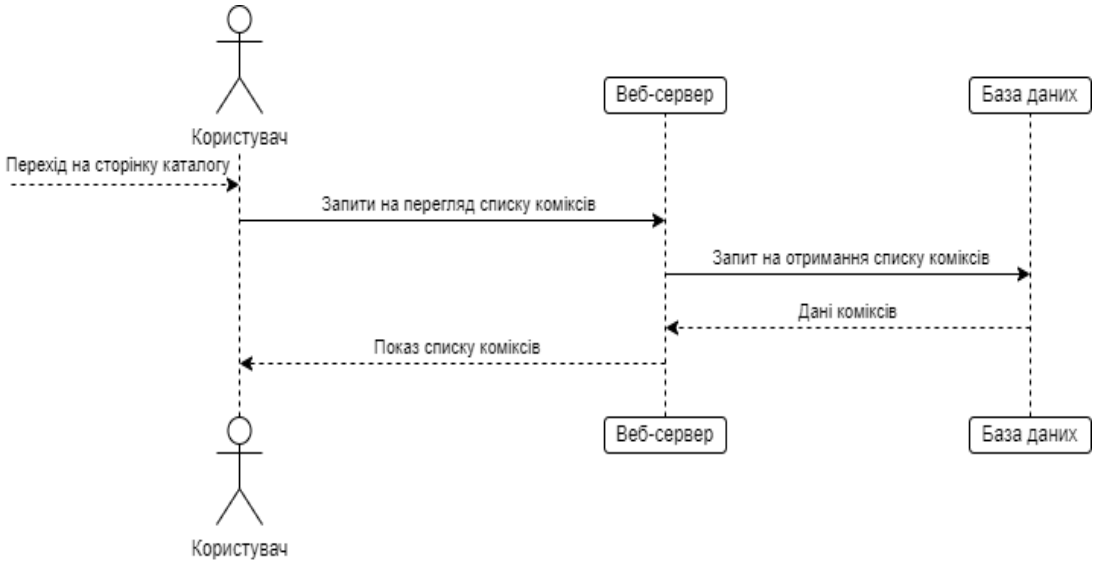

Рис *3.3 – Діаграма послідовності процесу перегляду списку коміксів*

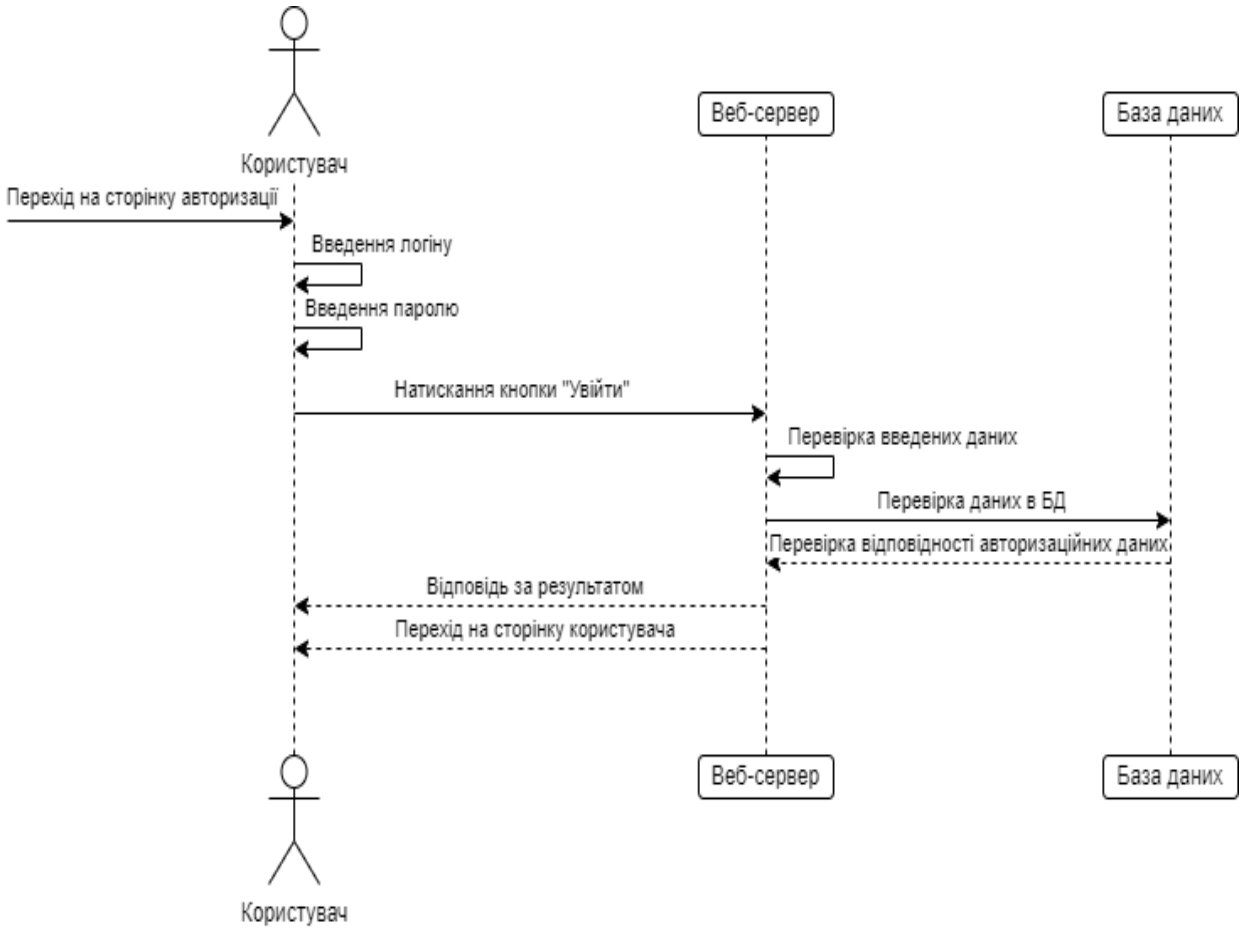

Рис *3.4 – Діаграма послідовності процесу авторизації*

З метою зобразити послідовність дій вебсервера у вигляді блок–схеми було виокремлено функції авторизації користувача на сайті, створення нового коміксу та відображення списку коміксів;

Вони відображені на рисунках 3.5 – 3.7.

У таблицях 3.2–3.4 відображені вхідні та вихідні дані алгоритмів.

Таблиця *3.2*

Вхідні та вихідні дані алгоритму послідовності дій користувача для реєстрації у системі

Вхід	**Опис**
A1	Дані користувача: логін, email, пароль
Вихід	**Опис**
A2	Реєстрація нового користувача та перехід на його сторінку

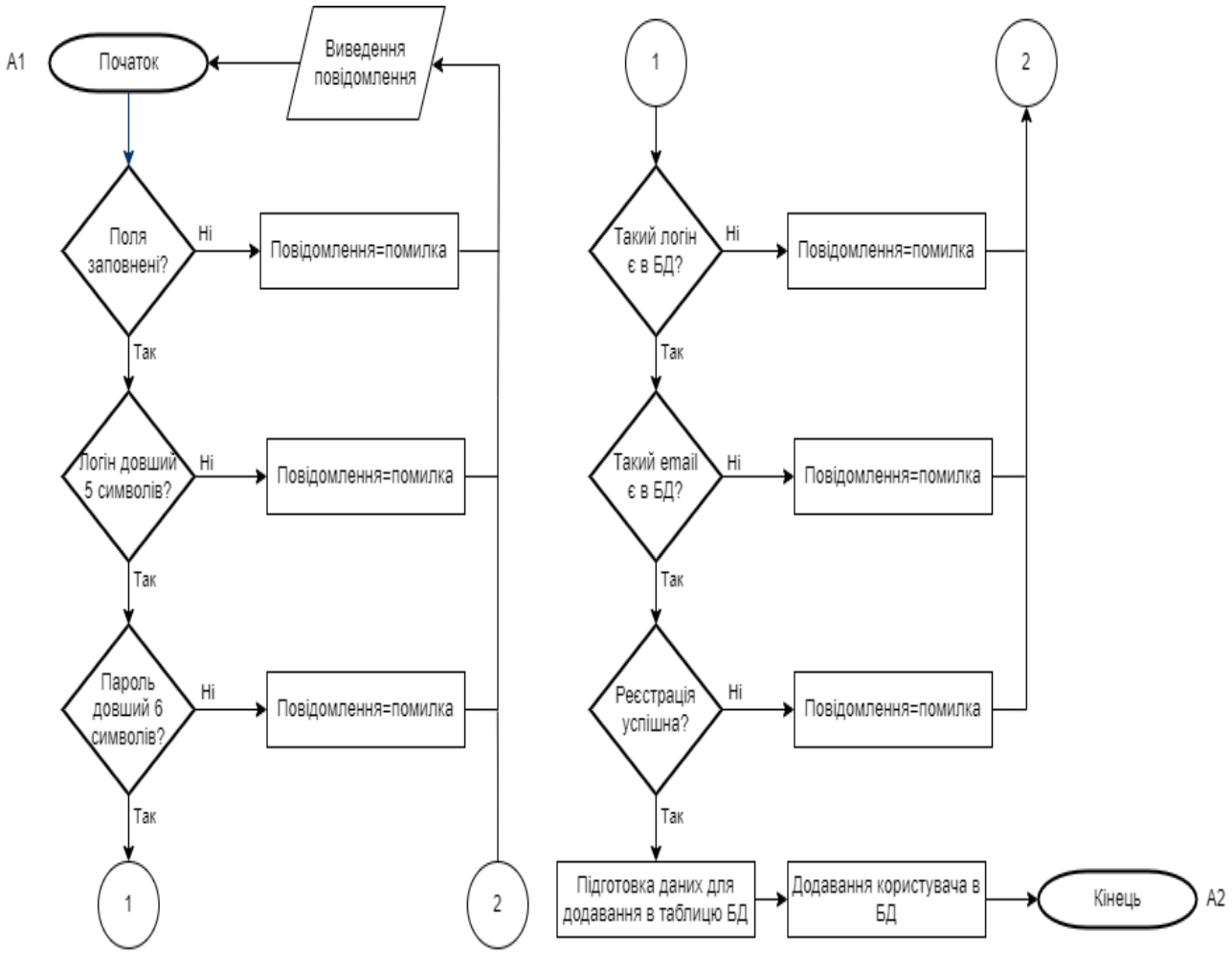

Рис *3.5* – *Блок–схема алгоритму послідовності дій користувача для реєстрації у системі*

Таблиця *3.3*

Вхідні дані та вихідні дані алгоритму послідовності дій створення нового коміксу

Вхід	**Опис**
В1	Авторизований користувач переходить на сторінку додавання коміксу.
Вихід	**Опис**
В2	Перехід на сторінку коміксу та повідомлення про перевірку коміксу адміністратором

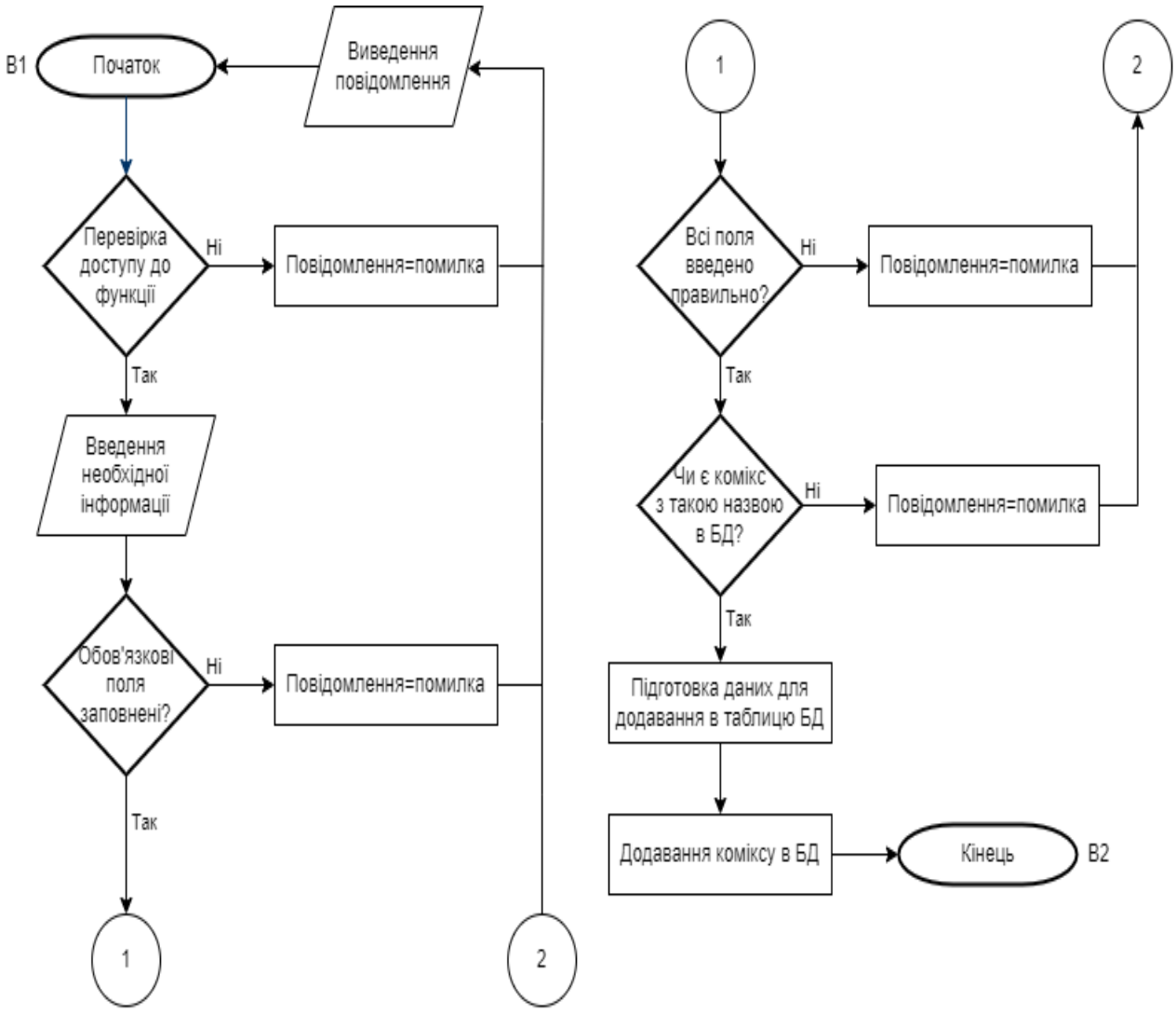

Рис 3.6 – *Блок–схема алгоритму послідовності дій додавання нового коміксу*

Таблиця *3.4*

Вхідні дані та вихідні дані алгоритму послідовності дій користувача для перегляду списку коміксів без фільтрації та з нею

Вхід	**Опис**
С1	Користувач переходить на сторінку каталогу
Вихід	**Опис**
С2	Формування списку коміксів

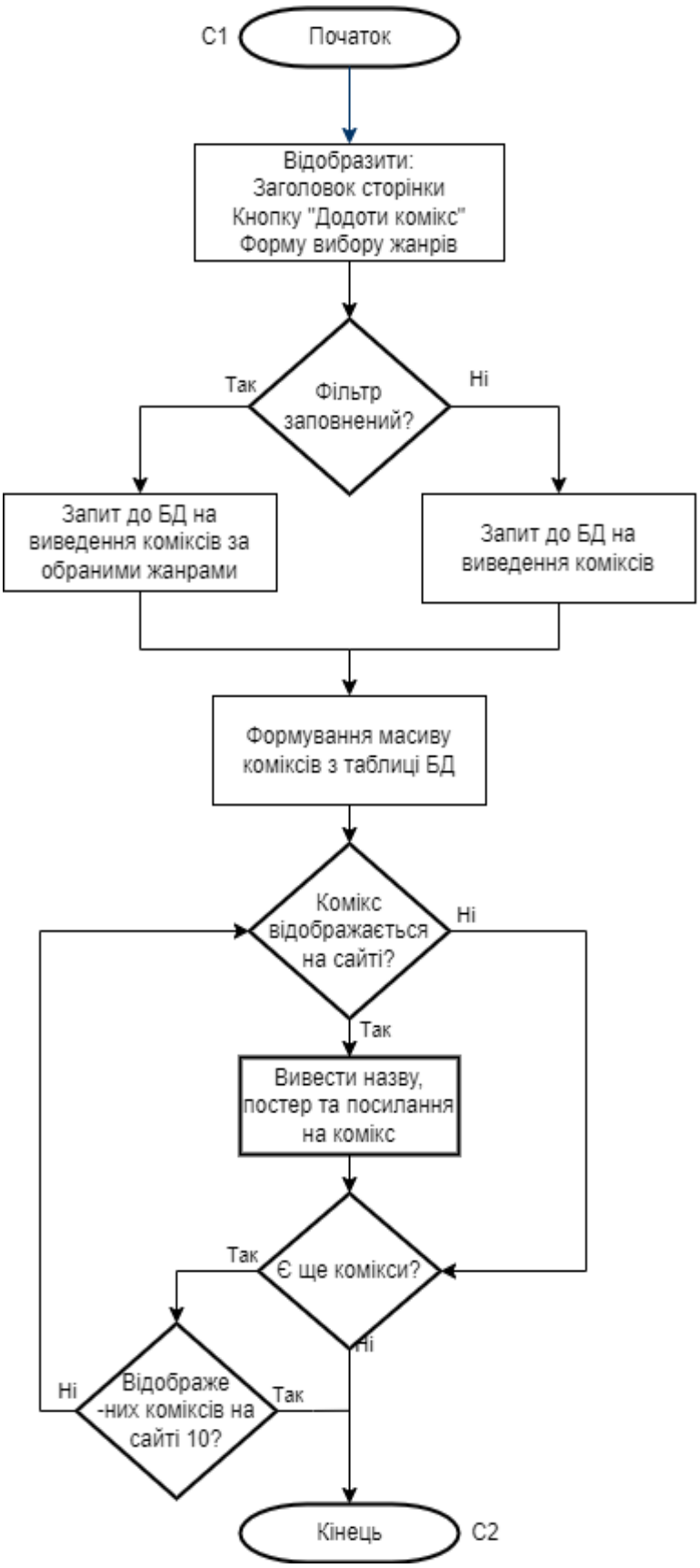

Рисунок 3.7 – Блок–схема алгоритму послідовності дій користувача для перегляду списку коміксів без фільтрації та з нею

3.4 Розроблення фізичної моделі даних бази даних вебзастосунку

На фізичній моделі даних БД відображаються такі елементи:

– фізичний тип даних атрибутів сутностей;

– обмеження таблиць та атрибутів сутностей;

На рисунку 3.8 відображено фізичну модель БД вебсервісу.

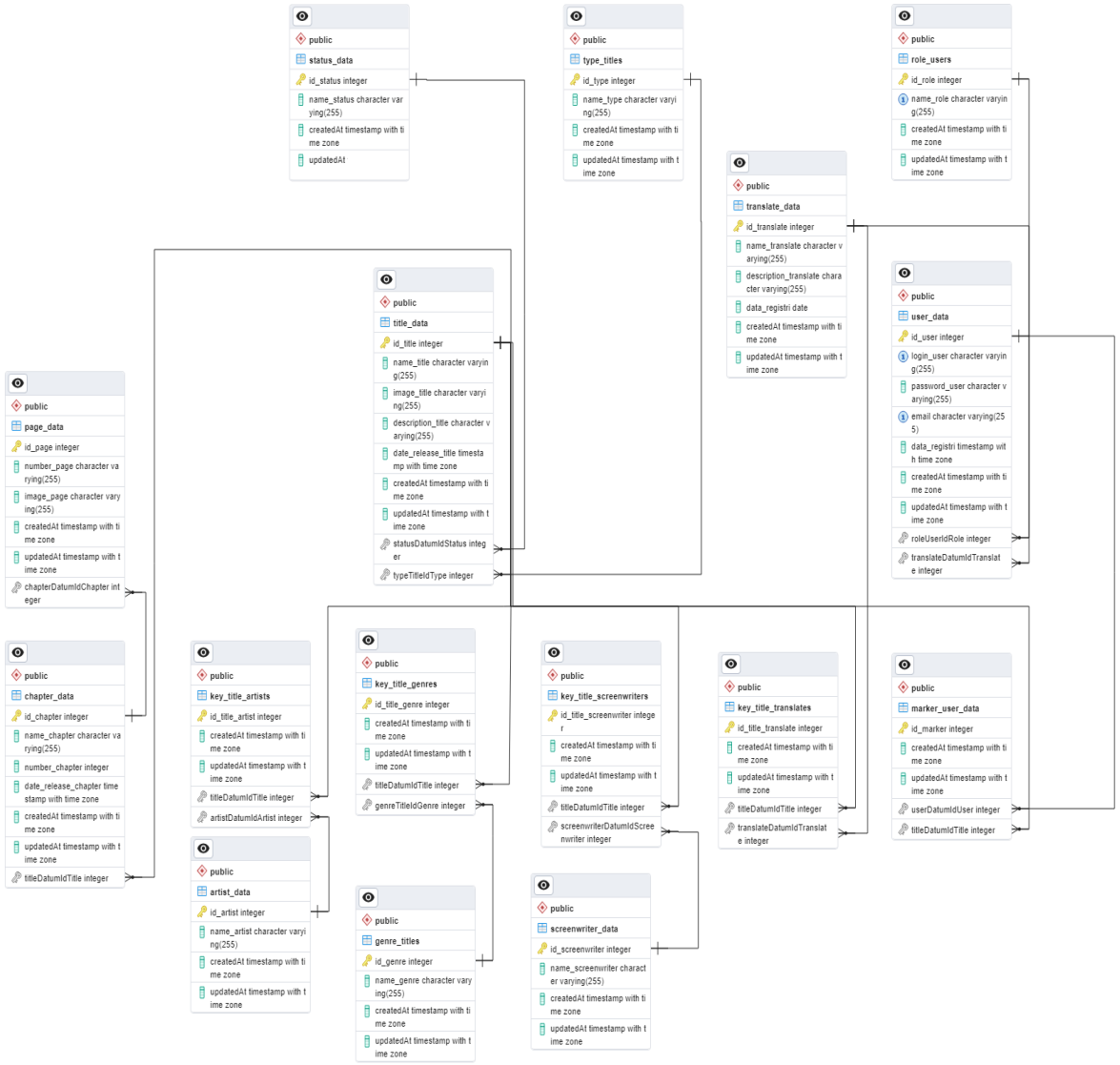

Рисунок 3.8 – Фізична модель БД

У таблицях 3.5 – 3.16 вказаний опис головних сутностей вебсайту.

Таблиця *3.5*

Опис сутності user_data

Сутність user_data		
Назва поля	Опис	Тип
id_user (PK) (NOT NULL)	Ідентифікатор	integer (AUTO_INCREMENT)
login_user (UK) (NOT NULL)	Логін	chapter varying(128)
password_user (NOT NULL)	Хешований пароль	chapter varying (64)
email (UK) (NOT NULL)	Електронна адреса	chapter varying (128)
data_registri (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата реєстрації	timestamp
updatedAt (Default CURRENT_TIMESTAMP)	Дата оновлення інформації	timestamp
roleUserIdRole (FK) (Default 1)	Ідентифікатор ролі користувача	integer
translateDatumIdTranslate (FK)	Ідентифікатор команди перекладачів	integer

Таблиця *3.6*

Опис сутності title_data

Сутність title_data		
Назва поля	Опис	Тип
id_title (PK) (NOT NULL)	Ідентифікатор коміксу	integer (AUTO_INCREMENT)
name_title (NOT NULL)	Назва	chapter varying(128)
image_title (NULL)	Посилання на постер	chapter varying(255)
description_title (UK) (NOT NULL)	Опис	text
date_release_title (UK) (NOT NULL)	Дата випуску	timestamp
createdAt (NOT NULL) (Default CURRENT_TIMESTAMP)	Дата створення	timestamp
updatedAt (NOT NULL) (Default CURRENT_TIMESTAMP)	Дата оновлення	timestamp
statusDatumIdStatus(FK) (NOT NULL)	Ідентифікатор статусу	integer
typeTitleIdType(FK)(NOT NULL)	Ідентифікатор типу	integer

Таблиця *3.7*

Опис сутності chapter_data

Сутність chapter_data		
Назва поля	Опис	Тип
id_chapter (PK) (NOT NULL)	Ідентифікатор глави	integer (AUTO_INCREMENT)
name_chapter	Назва глави	chapter varying (64)
number_chapter (NOT NULL)	Номер глави	integer
date_release_chapter (NOT NULL) (Default CURRENT_TIMESTAMP)	Дата створення	timestamp
updatedAt	Дата оновлення	timestamp
titleDatumIdTitle(FK)(NOT NULL)	Ідентифікатор коміксу	integer

Таблиця *3.11*

Опис сутності status_data

Сутність status_data		
Назва поля	Опис	Тип
id_status (PK) (NOT NULL)	Ідентифікатор статусу	integer (AUTO_INCREMENT)
name_status (NOT NULL)	Назва статусу	chapter varying(255)
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

Таблиця *3.8*

Опис сутності page_data

Сутність news		
Назва поля	Опис	Тип
id_page (PK) (NOT NULL)	Ідентифікатор сторінки	integer (AUTO_INCREMENT)
number_page (NOT NULL)	Номер сторінки	integer
image_page (NOT NULL)	Посилання на зображення	chapter varying(255)
createdAt (NOT NULL)	Дата створення	timestamp
updatedAt (NOT NULL)	Дата оновлення	timestamp
chapterDatumIdChapter (FK) (NOT NULL)	Ідентифікатор глави	integer

Таблиця *3.15*

Опис сутності genre_title

Сутність genre_title		
Назва поля	Опис	Тип
id_genre (PK) (NOT NULL)	Ідентифікатор жанру	integer (AUTO_INCREMENT)
name_genre (NOT NULL)	Назва жанру	chapter varying(255)
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

Таблиця *3.9*

Опис сутності translate_data

Сутність translate_data		
Назва поля	Опис	Тип
id_translate (PK) (NOT NULL)	Ідентифікатор перекладача	integer (AUTO_INCREMENT)
name_translate (NOT NULL)	Назва перекладача	chapter varying(255)
description_translate	Опис перекладача	chapter varying(255)
data_registri (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата реєстрації	timestamp
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

Таблиця *3.12*

Опис сутності screenwriter_data

Сутність screenwriter_data		
Назва поля	Опис	Тип
id_screenwriter (PK) (NOT NULL)	Ідентифікатор сценариста	integer (AUTO_INCREMENT)
name_screenwriter (NOT NULL)	Назва сценариста	chapter varying(255)
description_screenwriter	Опис сценариста	chapter varying(255)
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

Таблиця *3.10*

Опис сутності role_users

Сутність role_data		
Назва поля	Опис	Тип
id_role (PK) (NOT NULL)	Ідентифікатор ролі	integer (AUTO_INCREMENT)
name_role (NOT NULL)	Назва ролі	chapter varying(255)
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

Таблиця *3.13*

Опис сутності artist_data

Сутність artist_data		
Назва поля	Опис	Тип
id_artist (PK) (NOT NULL)	Ідентифікатор художника	integer (AUTO_INCREMENT)
name_artist (NOT NULL)	Назва художника	chapter varying(255)
description_artist	Опис художника	chapter varying(255)
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

Таблиця *3.14*

Опис сутності type_titles

Сутність type_titles		
Назва поля	Опис	Тип
id_type (PK) (NOT NULL)	Ідентифікатор типу	integer (AUTO_INCREMENT)
name_type (NOT NULL)	Назва типу	chapter varying(255)
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

Таблиця *3.16*

Опис сутності marker_user_data

Сутність marker_user_data		
Назва поля	Опис	Тип
id_translate (PK) (NOT NULL)	Ідентифікатор закладки	integer (AUTO_INCREMENT)
userDatumIdUser (FK) (NOT NULL)	Ідентифікатор користувача	integer
titleDatumIdTitle(FK) (NOT NULL)	Ідентифікатор коміксу	integer
createdAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата створення	timestamp
updatedAt (Default CURRENT_TIMESTAMP) (NOT NULL)	Дата оновлення	timestamp

3.5 Програмна реалізація серверної і клієнтської частини

Головний файл клієнтської частини веб додатку App.js (див. лістинг 1). Саме в ньому додається основний імпорт бібліотеки swiper, а також формується повна структура головної сторінки. Результат головної сторінки можна побачити на рисунку 3.8

Лістинг 1 – файл App.js

```
import React from 'react'

// import Swiper styles
import 'swiper/css';
import 'swiper/swiper.min.css';
import { BrowserRouter } from 'react-router-dom';
import AppRouter from './components/AppRouter';
import "./index.css"
import Header from './components/Header';
import Wrapper from './components/wrapper';
import Footer from './components/Footer';
const App = () => {
  return (
      <BrowserRouter>
            <Wrapper>
                  <Header/>
                  <AppRouter />
                  <Footer/>
            </Wrapper>
      </BrowserRouter>
  )
}

export default App
```

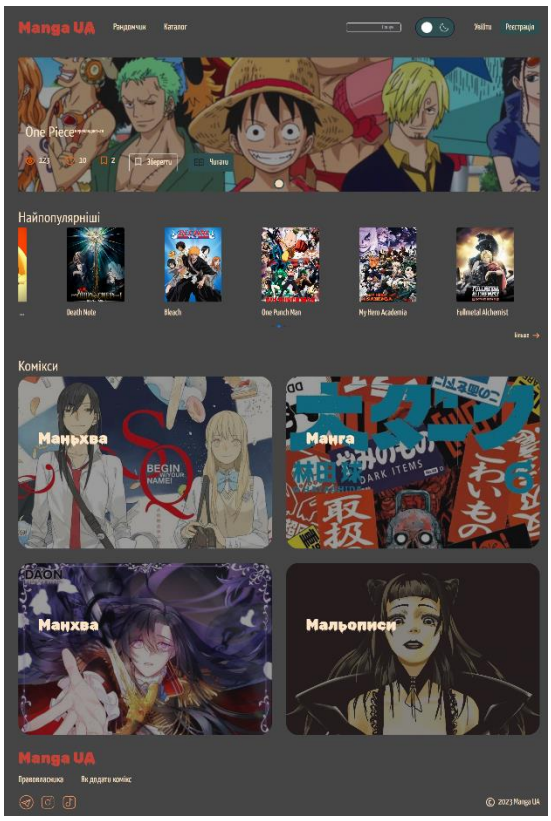

Рис *3.8 – Головна сторінка*

Файл routes.js описує основну логіку навігації по сторінках вебдодатку (див.лістинг 2).

Лістинг 2 – файл routes.js

```
import ArtistPages from "./pages/ArtistPages"
import Auth from "./pages/Auth"
import AuthorPages from "./pages/AuthorPages"
import CatalogPages from "./pages/CatalogPages"
import Manga from "./pages/Manga"
import Profile from "./pages/Profile"
import TeamsPages from "./pages/TeamsPages"
import TitleChapterPages from "./pages/TitleChapterPages"
import TitlePages from "./pages/TitlePages"
import TranlatorPages from "./pages/TranlatorPages"
import UserBookmarkPages from "./pages/UserBookmarkPages"
import UserSettingsPages from "./pages/UserSettingsPages"
import {ARTISTS_ROUTE, AUTHOR_ROUTE, BOOKMARK_ROUTE, CATALOG_ROUTE, CHAPTER_ROUTE, LOGIN_ROUTE, MANGA_ROUTE, PROFILE_ROUTE, REGISTRATION_ROUTE, SETTINGS_ROUTE, TEAMS_ROUTE, TITLE_ROUTE, TRANSLATOR_ROUTE } from "./utils/consts"
```

```
// Описує  усі маршрути(роути) нашого додатка
export const authRoutes = [
      {path: PROFILE_ROUTE, Component: Profile },
      {path: BOOKMARK_ROUTE, Component: UserBookmarkPages },
      {path: SETTINGS_ROUTE,Component: UserSettingsPages},
      {path: TEAMS_ROUTE, Component: TeamsPages},
]
export const publicRoutes = [
      {path: MANGA_ROUTE, Component: Manga},
      {path: LOGIN_ROUTE, Component: Auth},
      {path: REGISTRATION_ROUTE, Component: Auth},
      {path: CATALOG_ROUTE, Component: CatalogPages},
      {path: TITLE_ROUTE + '/:id', Component: TitlePages},
      {path: CHAPTER_ROUTE + '/:id', Component: TitleChapterPages},
      {path: TRANSLATOR_ROUTE + '/:id', Component: TranlatorPages},
      {path: ARTISTS_ROUTE + '/:id',Component: ArtistPages},
      {path: AUTHOR_ROUTE + '/:id', Component: AuthorPages},
]
```

Для створення випадаючого списку були використані хуки з бібліотеки React, а саме useEffect, useRef, useState. Що надали змогу зберігати і обробляти додатковий функціонал випадаючого списку. Компонент був розроблений, як універсальний, що надає можливість отримувати список данних для нього на будь якій сторінці. Код даного компонента описано в лістингу 3. Результат зображено на рисунку 3.9

Лістинг 3 – файл DropDown.jsx

```
import React, { useEffect, useRef, useState } from 'react'
const DropDown = ({ options, onSelect, placeholderText, ImageIcon,
selectedOption}) => {
      // state with open
      const [isOpen, setIsOpen] = useState(false);
      //link for parent div in dropdown
      const dropdownRef = useRef(null);
      // toggle is open drop down
      const handleToggle = () => {setIsOpen(!isOpen);}
      // option select in drop down
```

```
            const handleOptionSelect = (option) => {setIsOpen(false);
onSelect(option); }
            // use effect on click with anyway only not dropdown
            useEffect(()=> {const handleClickOutside = (event) => {
                        if (dropdownRef.current &&
!dropdownRef.current.contains(event.target)) {
                              setIsOpen(false); }};
                        document.addEventListener('click',handleClickOutside);
                  return () => {
                        document.removeEventListener('click',
handleClickOutside);
                  }
            },[])
            return (<div className='relative' ref={dropdownRef}>
                        <button type='button'className='inline-flex items-
center justify-between w-full px-1 py-1 lg:px-4 lg:py-2 text-text-sm lg:text-
text-md border border-[0.8px] border-stroke-dark rounded-md'
onClick={handleToggle}>
                              {selectedOption ? selectedOption :
placeholderText}
                              <span className={`transition delay-200 duration-
300 ease-in-out ${isOpen ? 'rotate-180' : ''}`}>{ImageIcon}</span>
                        </button>
                                    {isOpen && (
                                     <div className={`absolute w-full left-0
backdrop-filter backdrop-blur-[5px] z-10 transition-opacity duration-300
ease-in-out ${isOpen ? 'opacity-100 visible' : 'opacity-0'}`}>
                                     {options.map((option) => (
                                       <button key={option} type='button'
                                           className='block text-left w-full
px-4 py-2 text-text-md hover:bg-slate-400 focus:bg-slate-400 rounded-sm'
                                           onClick={() =>
handleOptionSelect(option)}>
                                           {option}
                                       </button>))}
                                  </div> ) }
                  </div>)}
      export default DropDown
```

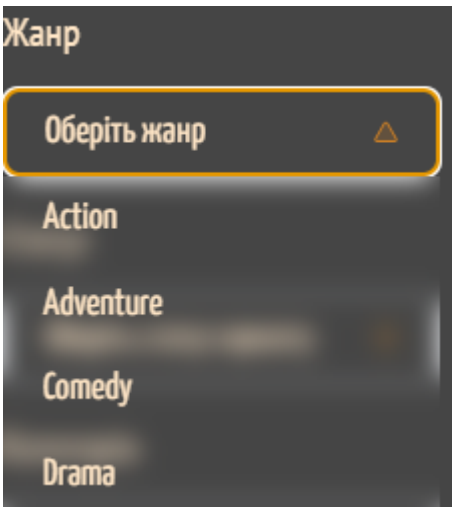

Рисунок 3.9 – випадаючий спосок жанрів

Щоб створити перемикач світлої теми на темну тему сайту, використовувся хук стану, що зберігав першочергове значення в батьківському компоненті, а сам же компонент перемикача , який описано в лістинг 4, приймав значення цього стану. Задля отримання теперешніх значень, компонент був обернений в функцію observer з бібліотеки mob-x. Результат створення перемикача зображено на рисунку 3.10.

Лістинг 4 – RadioBtnTheme.jsx

```
import React, { useContext } from 'react'
import { ReactComponent as SunIcon } from '../images/sun-icon.svg'
import { ReactComponent as MoonIcon } from '../images/moon-icon.svg'
import { Context } from '../index';
import { observer } from 'mobx-react-lite';
import { DARK_THEME, LIGHT_THEME } from '../utils/consts';
const RadioBtnTheme = observer(() => {
      // контекст теми
      const {theme} = useContext(Context);
      // Обробник події теми
      const handleThemeChange = (event) => {
            theme.setTheme(event.target.value);
      }
  return (
      <React.Fragment>
            {
                  theme._theme === LIGHT_THEME
```

```
                        ?
                        <div className='py-[10px] px-[15px] rounded-[35px]
flex items-center space-x-2.5 bg-white border border-solid border-2 border-
stroke-light-theme transition-all delay-100 duration-500 ease-in-out'>
                            <SunIcon/>
                            <input className='h-8 w-8 bg-xl-elipse-light'
type="radio" id="radio" value={LIGHT_THEME} onChange={handleThemeChange}
name="color-theme" />         <label htmlFor="radio">
                            </label>
                        </div> :
                        <div className='py-[10px] px-[15px] rounded-[35px]
flex items-center space-x-2.5 bg-dark-theme-btn border border-solid border-2
border-stroke-dark-theme transition-all delay-100 duration-500 ease-in-out'>
                            <input className='h-8 w-8'
type="radio"id="radio2"  value={DARK_THEME} onChange={handleThemeChange}
name="color-theme" />
                            <label htmlFor="radio2">
                            </label>
                            <MoonIcon/>
                        </div>
                    }
                </React.Fragment>
            )
        })
        export default RadioBtnTheme
```

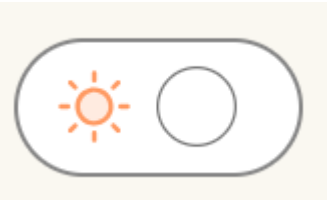

Рисунок 3.10 – перемикач світлої та темної теми

Реалізовані екранні форм й інша логіка серверної частини веб–додатку для читання коміксів відображені у файлах додатку А .

ВИСНОВКИ

У результаті виконання кваліфікаціної роботи було створено програмну реалізацію вебдодатку читання коміксів Манга.

В розробці вебдодатку використовувався наступний стек технологій.

– платформа Node.js, мова JavaScript, фреймфорк Express.js;

– СУБД PostgreSQL, ORM Prisma, бібліотека Prisma Client;

– Мова Javascript, фреймворк React, бібліотеки Mobx, Swiper і Tailwind CSS.

У ході роботи над проєктом було виконано:

– детальний аналіз предметної області та огляд аналогів онлайн-бібліотеки Манги, в результаті було сформульовано функціональні та нефункціональні вимоги до застосунку;

– проєктування схеми даних бази даних;

– визначення основних сутностей взаємодії з вебдодатком;

– створення макету інтерфейсу й розроблення схеми API;

– розробку RESTful API, використовуючи платформи Node.js та фреймворку Express.js;

– розроблення клієнтського додатоку, за допомогою фреймворку React.

В результаті виконання кваліфікаційної роботи було успішно створено онлайн–бібліотеку японських коміксів Манга, яка дозволяє переглядати та додавати комікси та містить функції реєстрації та закладок.

СПИСОК ДЖЕРЕЛ

1. Ковбасенко Ю. І. MANUSCRIPT: КЛАСИЧНА СПАДЩИНА І СУЧАСНИЙ ЛІТЕРАТУРНИЙ ПРОЦЕС, 821.161.2-9.09:7.02, Серія ДК № 1001 від 31 липня 2012 року

2. Honey Manga [Електронний ресурс] – Режим доступу до ресурсу: https://honey-manga.com.ua/

3. Webtoon [Електронний ресурс] – Режим доступу до ресурсу: https://www.webtoons.com/en/

4. Manga In UA [Електронний ресурс] – Режим доступу до ресурсу: https://www.manga.in.ua/

5. Shinden [Електронний ресурс] – Режим доступу до ресурсу: https://www.manga.in.ua/

6. Tailwind.css [Електронний ресурс] документація налаштування і використання, v.3.3.2 https://tailwindcss.com/docs/installation

7. Swiper.js [Електронний ресурс] документація налаштування і використання з Swiper React Component, v.9.4.0, https://swiperjs.com/react

8. Express.js [Електронний ресурс] документація з використання, v.4, https://expressjs.com/en/guide/routing.html

9. Express.js [Електронний ресурс] документація з налаштування, v.4, https://expressjs.com/en/guide/routing.html

10. Axios [Електронний ресурс] документація налаштування і використання: https://axios-http.com/uk/docs/intro

11. React [Електронний ресурс] документація з початку роботи й налаштування середовища https://uk.legacy.reactjs.org/

12. Robin Nixon, Learning PHP, MySQL & JavaScript: A Step-by-Step Guide to Creating Dynamic Websites, 528 с.

13. Wieruch R. The Road to React: Your journey to master plain yet pragmatic React.js. Chicago : Independently published, 2018. 292 с.

ДОДАТОК А

Макети сторінок вебдодатку

На рисунках Б.1 – Б.6 представленні основні макети сторінок розроблюємого вебдодатку для перегляду та додавання коміксів.

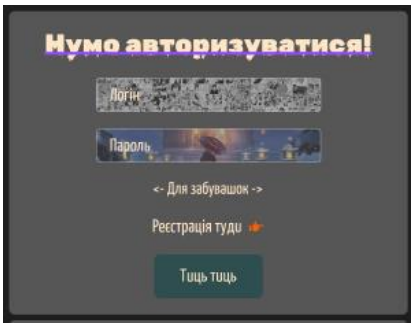

Рисунок А.1 – Макет сторінки авторизації

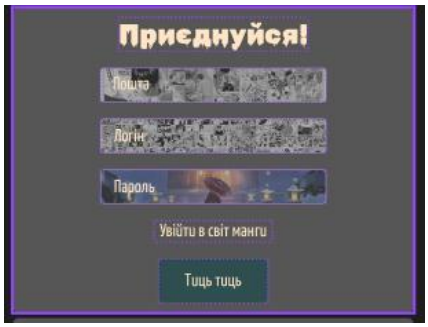

Рисунок А.2 – Макет сторінки реєстрації

Рисунок А.3 – Макет сторінки користувача

Рисунок А.4 – Макет сторінки каталогу

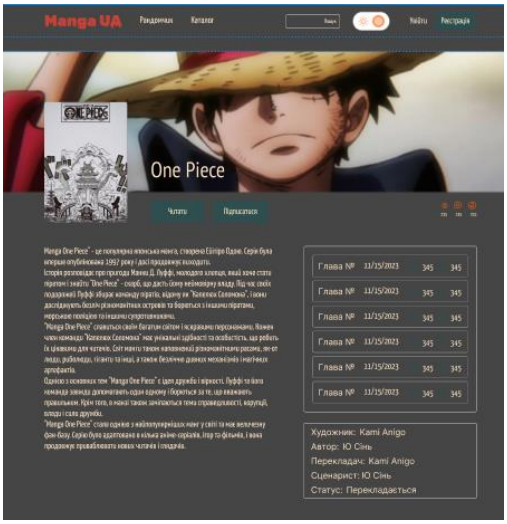

Рисунок А.5 – Макет сторінки коміксу

Рисунок А.6 – Макет сторінки глави