Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Медіасистем та технологій
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

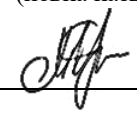
Розробка веб-додатку «Сучасна онлайн-бібліотека»
(тема)

Виконала:
студентка 4 курсу, групи ВПВПС-20-2


Фоменко Ю.В.
(прізвище, ініціали)

Спеціальність 186 Видавництво та поліграфія
(код і повна назва спеціальності)
Тип програми освітньо-професійна

Освітня програма
Видавничо-поліграфічна справа
(повна назва освітньої програми)

Керівник  доц. Табакова І.С.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри МСТ _____ Дейнеко Ж.В.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
Кафедра Медіасистем та технологій
Рівень вищої освіти перший (бакалаврський)
Спеціальність 186 Видавництво та поліграфія
Тип програми Освітньо-професійна
Освітня програма Видавничо-поліграфічна справа
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри МСТ_____
(підпис)
« 20 » травня 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентці Фоменко Юлії Валентинівни
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка веб-додатку «Сучасна онлайн-бібліотека»

Затверджена наказом по університету від 20 травня 2024 р. № 458 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 10 червня 2024 р.

3. Вихідні дані до роботи
Національні та міжнародні стандарти оцінки якості веб-сайтів; ДСТУ Інженерія систем і програмних засобів. Розроблення та керування веб-додатками для систем, програмних засобів та інформаційних послуг.

4. Перелік питань, що потрібно опрацювати в роботі
Вступ; Аналіз технічного завдання; Аналітичний огляд літератури за темою; Аналіз аналогів; Процес проектування веб-додатку; Процес розробки веб-додатку; Економічна частина; Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п. 5 включається до завдання за рішенням випускової кафедри)
Перелік слайдів презентації: титульна сторінка (1 слайд), актуальність проєкту (1 слайд), мета роботи (1 слайд),цільова аудиторія, задачі для досягнення мети (1 слайд), аналіз аналогів (1 слайд), вибір технологій для розробки системи (1 слайд), функціональні та нефункціональні вимоги (1 слайд),фірмові кольори (1 слайд), вибір шрифтів (1 слайд), модульна сітка (1 слайд), структура проєкту (1 слайд), написання коду (1 слайд), тестування (1 слайд), виправлення помилок (1 слайд), результати розробки (1 слайд), економічна частина (1 слайд), висновки (1 слайд).

6. Консультанти розділів роботи (п. 6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п. 1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	доц. Табакова І.С.		02.06.2024
Економічна частина	ас. Помогалова Н.В.		30.05.2024

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз технічного завдання	17.05.2024	виконано
2	Аналітичний огляд літератури за темою	18.05.2024	виконано
3	Аналіз аналогів	19.05.2024	виконано
4	Процес проектування веб-додатку	22.05.2024	виконано
5	Процес розробки веб-додатку	26.05.2024	виконано
6	Економічна частина	28.05.2024	виконано
7	Оформлення пояснювальної записки	01.06.2024	виконано
8	Оформлення графічної частини	01.06.2024	виконано

Дата видачі завдання 20 травня 2024 р.

Студентка _____ Фоменко Ю.В.
(підпис)

Керівник роботи _____ доц. Табакова І.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи містить 76 стор., 31 рис., 8 табл., 1 дод., 11 джерел.

ВЕБ-ДОДАТОК, REACT, EXPRESSJS, MONGODB, CLOUDINARY, OPENAI, AWS, NGINX, GIT, ОПТИМІЗАЦІЯ, ДИЗАЙН, HTML, CSS, JAVASCRIPT, АДАПТИВНІСТЬ, ТЕСТУВАННЯ, ВЗАЄМОДІЯ, ДОКУМЕНТАЦІЯ, ІНТЕРФЕЙС, ПРОТОТИПУВАННЯ, РОЗРОБКА, СЕРВЕР, КЛІЄНТ.

Метою проєкту є створення зручної та функціональної платформи для доступу до різноманітних літературних ресурсів в режимі онлайн. Це сприятиме полегшенню процесу пошуку та використання інформації для користувачів, підтримуючи їх освітні та дослідницькі потреби.

Було розглянуто аналітичний огляд літератури за темою кваліфікаційної роботи та проведенний аналіз аналогів. Наступним кроком було детально проаналізовано ключові аспекти інформаційної системи (ІС) для сучасної онлайн-бібліотеки. Детальний опис стеку технологій для бекенду та фронтенду допоміг зрозуміти, які інструменти та технології будуть використовуватися для розробки системи. Далі було розглянуто структуру проекту для створення сучасної онлайн-бібліотеки та уявлення про сутності інформаційної системи. За допомогою зазначених технологій та компонентів було реалізовано веб-інтерфейс, що дозволяє користувачам реєструватися, публікувати книги, читати їх, а також отримувати короткі змістові огляди кожної глави. У висновку додаток був протестований, а потім наповнений книгами. Проведено економічне обгрунтування проекту, після якого була розрахована ціна проекту.

ABSTRACT

The explanatory note of the qualification work contains 76 p., 31 fig., 8 tabl., 1 app., 11 sources.

WEB APPLICATION, REACT, EXPRESSJS, MONGODB, CLOUDINARY, OPENAI, AWS, NGINX, GIT, OPTIMIZATION, DESIGN, HTML, CSS, JAVASCRIPT, ADAPTABILITY, TESTING, INTERACTION, DOCUMENTATION, INTERFACE, PROTOTYPING, DEVELOPMENT, SERVER, CLIENT.

The goal of the project is to create a convenient and functional platform for accessing various literary resources online. This will facilitate the process of finding and using information for users, supporting their educational and research needs.

An analytical review of the literature on the topic of qualification work was considered and an analysis of analogues was carried out. The next step was to analyze in detail the key aspects of an information system (IS) for a modern online library. A detailed description of the technology stack for the backend and frontend helped to understand which tools and technologies would be used to develop the system. Next, the structure of the project to create a modern online library and an idea of the essence of the information system were considered. With the help of the specified technologies and components, a web interface was implemented that allows users to register, publish books, read them, and also receive brief content overviews of each chapter. In conclusion, the application was tested and then filled with books. An economic justification of the project was carried out, after which the price of the project was calculated.

ЗМІСТ

С.

ВСТУП 8
1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ 9
2 АНАЛІТИЧНИЙ ОГЛЯД ЛІТЕРАТУРИ ЗА ТЕМОЮ 11
2.1 Введення у Web-технологіях 11
2.2 Сучасні тенденції у Web-технологіях 12
2.3 Сучасні тенденції у Web-дизайні 14
3 АНАЛІЗ АНАЛОГІВ 18
3.1 Порівняння аналогів інформаційних систем 18
3.1.1 E-bookua 19
3.1.2 Chtyvo 20
3.1.3 Ukrlib 21
3.1.4 Результат порівнянь 23
3.2 Вибір технологій для розробки системи 23
3.2.1 Вибір архітектурного шаблону 23
3.2.2 Вибір мови програмування 24
3.2.3 Вибір середовища розробки 25
3.2.4 Вибір фреймворків і бібліотек 26
3.2.5 Вибір бази даних 28
3.2.6 Вибір патернів проектування 29
4 ПРОЦЕС ПРОЕКТУВАННЯ ВЕБ-ДОДАТКУ 31
4.1 Мета та задачі ІС 31
4.2 Типи користувачів 32
4.3 Функціональні вимоги 33
4.4 Нефункціональні вимоги 34
4.5 Ідентифікація архетипу ІС 35
4.6 Користувацький інтерфейс (UI View) 35
4.7 Логічне уявлення про ІС (Logical View) 41

4.8 Уявлення розгортання ІС (Deployment View) 41
4.9 Уявлення слоїв ІС (Design View) 42
4.10 Уявлення процесів ІС (Process View) 43
4.11 Уявлення даних ІС (Data View) 44
4.12 Уявлення інтерфейсів ІС (Interface View) 47
4.13 Уявлення безпеки ІС (Security View) 47
4.14 Інфраструктурне уявлення ІС (Infrastructure View) 48
4.15 Опис стеку технологій 48
5 ПРОЦЕС РОЗРОБКИ ВЕБ-ДОДАТКУ 50
5.1 Уявлення про структуру проекту 50
5.2 Уявлення про сутності інформаційної системи 50
5.3 Управління програмним кодом системи 56
5.4 Розрахунок метрики програмного коду системи 56
5.5 Контрольний список по якості реалізації системи 57
5.6 Тестування інформаційної системи 58
5.6.1 Розробка тест–кейсів для функціонального тестування системи ... 58
5.6.2 Протокол проведення функціонального тестування 60
5.7 Інструкція користувача інформаційної системи 62
6 ЕКОНОМІЧНА ЧАСТИНА 68
6.1 Характеристика продукції та ринок збуту 68
6.2 Розрахунок витрат 69
ВИСНОВКИ 74
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 76
ДОДАТОК А Програмний код веб-додатку 77

ВСТУП

У сучасному цифровому світі, де інформація стає все більш доступною та розповсюджується зі швидкістю світла, виникає необхідність в нових підходах до доступу до знань і літератури. Одним із таких інноваційних рішень є створення сучасної онлайн-бібліотеки, що надає користувачам можливість зручного та ефективного доступу до різноманітної літератури.

Метою даної роботи є розробка онлайн-платформи, що забезпечить зручне читання літературних творів та полегшить процес розповсюдження книг. Зокрема, система матиме можливість реєстрації як авторів, так і читачів, відкриваючи шлях для спільного обміну та взаємодії. Автори матимуть можливість публікувати свої твори, а читачі – насолоджуватися їхнім змістом.

Одним із ключових аспектів веб-додатку онлайн-бібліотеки є автоматичне розбиття книг на категорії та створення коротких змістів для кожної з них. Цей підхід максимально полегшить сприйняття контенту та дозволить користувачам швидко ознайомитися з суттю кожного розділу книги.

Основною ідеєю роботи буде створення інтуїтивно зрозумілого інтерфейсу, що матиме можливість користувачам насолоджуватися літературою у найбільш зручний спосіб. Таким чином, онлайн-бібліотека сприятиме розвитку культури читання та розширенню кругозору через доступність та простоту використання.

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

Згідно з технічним завданням на дану кваліфікаційну роботу [1] необхідно розробити веб-додаток «сучасної онлайн-бібліотеки». Вона призначена для надання змоги публікувати свої книги авторам, та читачам їх читати, так само введення нової інформації, коригування та сортування вже існуючих даних.

Надання змоги публікувати свої книги авторам дозволяє розширювати інформаційний простір мережі «інтернет», пов'язаних з пошуком цікавої публікації.

Однією з таких програм є «сучасна онлайн-бібліотека». Ця програма дозволяє публікувати, редагувати, читати, видаляти публікації.

Основні можливості програми:

- додавання нових публікацій з фото;
- редагування публікацій;
- видалення публікацій;
- реєстрація та авторизація користувачів;
- пошук авторів на сайті;
- простий і зручний інтерфейс для роботи користувача.

В даний час існує велика кількість способів створення програмного забезпечення аналогічного «сучасної онлайн-бібліотеки». Для вирішення поставленого завдання можна виділити такі програмні засоби як:

- табличні процесори;
- мова програмування;
- база даних.

За допомогою будь-якого з цих програмних засобів можна забезпечити повне виконання технічного завдання на проектування.

Для створення програмного продукту «сучасної онлайн-бібліотеки» найбільш зручним засобом є база даних MongoDB, так як це найбільш наочне і простий засіб.

База даних, що розробляється в кваліфікаційній роботі, спрощує і прискорює процес обробки інформації довідкової служби шляхом організації запитів до бази даних і відображення необхідної інформації на екрані.

Програма «сучасна онлайн-бібліотека» може використовуватися не тільки як самостійний продукт, але і як складова частина автоматизованих систем обробки інформації та інших аналогічних організаціях. Ця обставина дозволяє розглядати цю програму як універсальну.

Цільова аудиторія.

Цільовою аудиторією даного веб-додатку визначено дітей шкільного віку – особи у віці від 6 до 18 років, які повинні здобувати загальну середню освіту та молодих жінок і чоловіків від 18 до 44 років.

2 АНАЛІТИЧНИЙ ОГЛЯД ЛІТЕРАТУРИ ЗА ТЕМОЮ

2.1 Введення у Web-технологіях

Інтернет це глобальна мережа комп'ютерних мереж, а WWW це один із сервісів Інтернет. Але ця плутанина скоріше говорить про глобальну Популярність всесвітньої павутини (WWW) і підкреслює її значимість для користувачів.

Веб-технології це логічна складова інтернет–технологій, які включають в себе:

а) інтернет-сервіси: WWW – всесвітня павутина;

б) робота в інтернет:

1) браузери;

2) пошукові системи;

3) перегляд сторінок в браузері;

в) інформаційні ресурси Інтернет:

1) веб-сторінки, інтернет-магазини, інтернет-портали;

2) URL і протоколи передачі даних, адресація;

3) створення сайтів;

4) мови веб-програмування.

Основні поняття веб-технологій: веб-сторінка і веб-сайт. Їх не варто плутати.

Веб-сайт – це набір веб-сторінок пов'язаних загальною тематикою. Веб-сайт знаходиться на одному сервері (хостингу) і належить одному власнику. Як варіант, веб-сайт може складатися з однієї веб-сторінки (сайт – візитка).

Сукупністю всіх веб-сайтів і утворюють всесвітню павутину, часто звану веб або WWW-сервісу Інтернет, створеного для пошуку та обміну потрібної інформації.

Веб-сторінка будується на основі мови розмітки гіпертексту. Офіційна назва цієї мови HTML (Hyper Text Markup Language).

Для відображення вмісту на веб-сторінки створені і служать каскадні таблиці стилів, інакше CSS.

У створенні динамічних сторінок, на допомогу розробникам «приходять» мови скриптів. Найпопулярніша Мова JavaScript.

В принципі, за допомогою цих трьох мов HTML, CSS, JavaScript можна створити будь – яку, навіть найскладнішу, веб-сторінку, а створені веб-сторінки зібрати в веб-сайт.

Щоб браузери відображали створені веб-сторінки, а браузери бачать веб-сторінку, як набір об'єктів, був створений стандарт DOM (Document Object Model). Згідно з ним, веб-сторінка повинна представлятися у вигляді набору об'єктів, а сам стандарт DOM називається об'єктна модель документа.

DOM пов'язаний з JavaScript, а по суті, ця модель пов'язує веб-сторінки зі скриптами або мовами програмування.

В'язку HTML, CSS, JavaScript і DOM називають динамічним HTML або Dynamic HTML, а іноді DHTML.

Підводячи підсумки, зазначу, що основна мета у вивченні веб-технологій це створення або зміна веб-сторінок, які будуть правильно відображатися в браузерах. Також введення в web-технології допоможе вам читати коди ваших веб-сторінок, що, безсумнівно, потрібно для будь-якого власника веб ресурсів.

2.2 Сучасні тенденції у Web-технологіях

Кожен день з'являються різні інновації і нові тенденції в розвитку Інтернету. У веб-розробці має бути багато речей, таких як візуальний зовнішній вигляд веб-сайту, конфігурація, веб-API, кешування, послуги веб-сайту тощо. Щоб впоратися з усіма цими труднощами, необхідний висококваліфікований веб розробник. Крім того, існують різні рамки веб-

розробки, які дозволяють значно спростити процес веб-розробки. Крім того, платформи можуть сприяти автоматизації дій веб-розробки

Progressive Web Apps (PWA) – це веб-додаток, який використовує сучасні веб-технології для залучення поль-кличателей до додатків на вкладці браузера. Як і мобільні додатки, PWA має можливість завантажуватися з поганим з'єднанням або за його відсутності. Програми PWA також можуть бути ідентифіковані як програми, що дозволяють більшій кількості користувачів знаходити їх для подальшого використання. Більше користувачів можуть додавати ярлики PWA на початковий екран, щоб повернутися одним клацанням миші. Крім того, дані веб-додатки можуть посилати Push-повідомлення користувачам. Нарешті, PWA є привабливим, надійним та швидким завдяки користувальницькому інтерфейсу та дизайну UX, придбаному з мобільних додатків. Поряд з цим PWA приносять високі вигоди своїм власникам, такі як підвищена взаємодія, більш високі коефіцієнти конверсії і підвищена надійність.

Secure Socket Layer (SSL) – розробка для установки захищенного підключення браузера з сайтом. Дана розробка гарантує цілісність, кодування та ідентифікацію інформації. Якщо веб-ресурс застосовує технологію SSL, в адресний рядок додається приписка «s», а також з'являється зелений замок. Окремі сертифікати SSL також відображають найменування підприємства спеціальним забарвленням для розпізнавання. Тому логічно застосування SSL-сертифікат сьогодні, навіть якщо веб-ресурс не працює з особистими даними клієнтів. З HTTPS сайт більш захищений, ніж сайти HTTP для цільових ключових слів. Не тільки в сфері програмного забезпечення та програмування розвиваються тренди веб-розробки. Також дуже активно розвиваються і тенденції веб-дизайну, як області програмних розробок. Веб-сторінки сьогодні розглядається як свого роду усталена базова концепція, яка допомагає вміло оформити сайт. Особливо творчих особистостей, цей тренд забивав в обумовлені межі. Проте, завжди були хоробрі душі, які створили оригінальні рішення, які не контролюються ніякими жорсткими кордонами. Це заняття

помічається в сфері зображення і формування, але в нинішньому році цей тренд неординарного дизайну буде розподілятися на офіційні групові проекти.

Неординарна компоновка об'єктів пропонує більш цікаві потенціали: використання всієї сторінки, додавання шарів до об'єктів і доступність глибини відчуття. Можливість створення проекту, який здивує клієнтів, навіть без повномасштабної кольорової анімації. З тисячами звичайних моделей в Інтернеті незвичайні і рідкісні оформлення майже завжди відрізняються, а також притягують інтерес клієнтів веб-сторінки.

Підводячи підсумок вищесказаного, слід зазначити, що веб-розробки користується великим попитом в наш час. У нинішньому році ми бачимо веб-сайти, які задовольняють більшості аспектів вищевказаних елементів. Одним з найбільш важливих вимог сучасного сайту є швидка і проста робота, яка забезпечує хорошу продуктивність і економить час користувача. Хороший Розробник сьогодні повинен знати про новітні тенденції та відомі методи розробки веб-ресурсів.

2.3 Сучасні тенденції у Web-дизайні

Від якості дизайну залежить успіх сайту. Тому дуже важливо стежити за трендом, розуміти які інструменти увійшли в моду в сфері сайтобудування, а які вже вважаються застарілими і залишаються позаду. До слова, тренди у веб-дизайні постійно змінюються, удосконалюються, модернізуються. Те, що було актуально в 2021-2022 роках зменшує свої оберти, поступаючись абсолютно новим фішкам. І вам, як дизайнеру або сучасному підприємцю, необхідно знати про них, щоб результат розробки сайтів був не просто ефективним, а створював правильне враження про компанію, якій належить веб-ресурс.

Слідкуючи за тренду веб-дизайну, цільова аудиторія обов'язково по заслугах це оцінить, конвертуючись з відвідувача в постійного клієнта.

В першу чергу, тенденції в дизайні торкнулися мобільних гаджетів. Тільки в минулому році частота переглядів сайтів за допомогою мобільних

пристроїв перевершила використання ноутбуків і комп'ютерів. Логічно, що дизайн адаптивної версії сайту повинен відповідати цьому. Ще зовсім недавно зовнішній вигляд адаптивної версії ресурсу був обмежений, були відсутні яскраві елементи, які додавали вагу сторінок, різні повноформатні банери і тригери через обмежений розмір дисплеїв. Сьогодні ж, коли смартфони та планшети стали високоінтелектуальними, коли система розпізнає запит голосом і з'явилися голосові боти, а продуктивність гаджета і розмір діагоналі його екрану дозволяє включати в дизайн сайту абсолютно будь-які інструменти, для розробників з'явилося більше можливостей. Це і вплинуло на тренди web дизайну 2023 році.

Поняття сторітеллінга прийшло до нас із Заходу і надійно закріпилося в одному з напрямків контенту. Сьогодні дуже часто можна зустріти сторітеллінги в соціальних мережах, які переконують читача придбати який-небудь товар або послугу, посилаючись на історію про те, як він допоміг автору. Але кілька років тому принцип сторітеллінга закріпився у веб-дизайні, показавши високі результати.

За допомогою візуально оформленої історії, яка розповідається відвідувачеві, можна довго утримувати його увагу, мотивуючи до подальших дій: подивитися сайт, дізнатися більше про компанію, її товари або послуги.

Сторітеллінг може виступати у вигляді відео, гіф-анімації або ж розділений поблочно і розміщений по всій довжині сторінці. Це особливо зручно, якщо це довга Головна сторінка або посадкова.

Тренд на великий білий простір з'явився відносно нещодавно в 2016-2017 роках. Однак він надійно осів у веб-дизайні завдяки своїй здатності концентрувати увагу на головній пропозиції.

Великий білий простір візуально збільшує екран, не дозволяє користувачеві втратити концентрацію. До того ж, цей колір поєднується з усіма іншими, тому в якості акцентів або дизайнерських ідей можна вибирати абсолютно будь-які відтінки. І вони будуть виглядати просто чудово.

Свіжий тренд, який відтепер буде завжди актуальний. Адаптивні логотипи, які в залежності від розміру екрану мобільного девайса можуть автоматично підлаштовуватися під нього, одним махом відразу зможуть виконати дві функції: компанія збереже свій бренд перед цільовою аудиторією незалежно від його мобільного пристрою і пошукова система при аналізі мобільної версії сайту оцінить поведінковий фактор, поліпшивши позиції ресурсу в пошуковій видачі.

Спробуйте яскраві і багатоелементні фотографії на своєму сайті замінити на високоякісні знімки, де зображений всього один елемент, що відображає ідею і концепцію компанії. Це сфокусує увагу відвідувача і не дозволить йому відволіктися.

Ніхто і ніколи не зможе вам заборонити використання яскравих кольорів в дизайні свого сайту, якщо вони підкреслюють загальну ідею. Кольори передають емоції, а емоції – найголовніше для користувача. Побачивши соковитих і кричущих квітів, правильно підібраних один до одного, ваша цільова аудиторія оцінить по достоїнству вашу сміливість, залишившись на сайті і продовживши його користування.

Тренди в веб-дизайні – динамічна структура. Те, що ще вчора було модно і актуально, вже завтра може стати пройденим етапом, «минулим століттям». Але наведено приклади елементів дизайну, за допомогою яких завжди вдасться триматися на плаву, позиціонуючи свою компанію сучасною та успішною.

Класичний спосіб розташування блоків вже вважається застарілим. Ні, він не втрачає своєї ефективності, але завдяки новим методам піднесення інформації можна значно її підвищити. Спробуйте структурувати інформацію за методом ламаної сітки, хаотично розмістивши блоки по всій сторінці. Таке рішення створює новий виток з точки зору естетики – він підігріває інтерес користувача і дозволяє не виходити за рамки фірмового стилю навіть вузьконаправленої компанії.

Однак для використання даного способу розміщення блоків необхідно дуже ретельно продумати структуру сайту. Не забувайте, що читається зліва-направо, отже, вся найважливіша інформація повинна бути з лівого боку.

Хоча інфоконтент дозволяє максимально розкрити компанію, її конкурентні переваги та послуги, повноекранні відео можуть впоратися з цим не гірше. По-перше, фонові відео роблять дизайн сайту незвичайним, сучасним, що збільшує залученість відвідувача і призводить проект до більшої ефективності. По-друге, відео можуть стати не найгіршою історією про компанію. При цьому користувачеві не потрібно скроллити сторінку вниз, щоб знайти потрібну йому інформацію.

Насправді фішка з повноформатними відео в якості основного фону з'явилася вже давно. Вона встигла побувати на вершині популярності і піти, як непотрібний елемент, через неможливість підтримки мобільними пристроями і великої ваги, що призводило до зниження швидкості завантаження сайту. Але продуктивність гаджетів і технічні можливості оптимізації знову повернули відео в якості основного фону в лад. А це означає, що вони можуть знову зайняти лідируючі позиції в розробці дизайну в 2023 році.

3 АНАЛІЗ АНАЛОГІВ

3.1 Порівняння аналогів інформаційних систем

Задля створення конкурентоспроможного продукту варто провести аналіз аналогів, виявити їх переваги та недоліки. Спираючись на результати аналізу буде проведене проектування системи що не матиме недоліків конкурентів, це дозволить створити конкурентоспроможний продукт.

В результаті пошуку аналогів було знайдено три системи:

- E-bookua;
- chtyvo;
- ukrlib.

При аналізі аналогів основними критеріями будуть:

- оцінювання, функцій, які надаються кожною системою – це включає доступ до електронних книг, можливість пошуку, підтримку різних форматів файлів, можливість зберігання та організації бібліотеки тощо;
- аналізування, наскільки зручним та інтуїтивно зрозумілим є інтерфейс кожної системи для користувачів, це стосується дизайну, навігації та можливості персоналізації;
- перевірка обсягу та різноманітності електронних книг, доступних у кожній системі, якість контенту та його актуальність також важливі;
- визначення, як швидко та ефективно вирішуються проблеми та питання користувачів у кожній системі – це включає в себе наявність онлайн підтримки, документації та спільнот користувачів;
- оцінювання новаторських функцій та можливостей запропоновані кожною системою, які можуть забезпечити конкурентну перевагу.

3.1.1 E-bookua

E-bookua (рис. 3.1) – це онлайн бібліотека, яка надає доступ до широкого вибору електронних книг для своїх користувачів.

Рисунок 3.1 – Перегляд вступної частини книги на сайті E-bookua

Система пропонує різноманітні можливості для комфортного та зручного читання:

– E-bookua пропонує доступ до різноманітної колекції електронних книг різних жанрів, включаючи художню літературу, наукову літературу, документальні книги тощо. Користувачі можуть знайти книги для будь-якого смаку та інтересу;

– система підтримує різні формати файлів електронних книг, такі як EPUB, PDF, MOBI та інші;

– користувачі можуть налаштувати параметри читання, такі як розмір шрифту, тип шрифту, колір тексту та підсвічування;

– E-bookua дозволяє користувачам робити закладки у книгах та зберігати прогрес читання;

– деякі книги можуть бути завантажені для офлайн-читання;

– система забезпечує можливість синхронізації читання між різними пристроями, що дозволяє переходити від одного пристрою до іншого без втрати прогресу читання.

Загалом, E-bookua надає широкі можливості для комфортного та насиченого читання електронних книг, роблячи процес читання доступним та зручним для своїх користувачів.

3.1.2 Chtyvo

Електронна бібліотека «Чтиво» (рис. 3.2) була завжди працювала як некомерційний проєкт, спрямований на популяризацію українських текстів, які тривалий час було важко дістати, бо вони були переважно російською мовою, мали невеликий наклад і були майже виключно в паперовому вигляді. За цей час було нагромаджено велику кількість текстів, у тому числі рідкісних та унікальних, україномовних, іншомовних текстів української класики, праць з українознавства та історії України. Багато назв було додано авторами, перекладачами та редакторами.

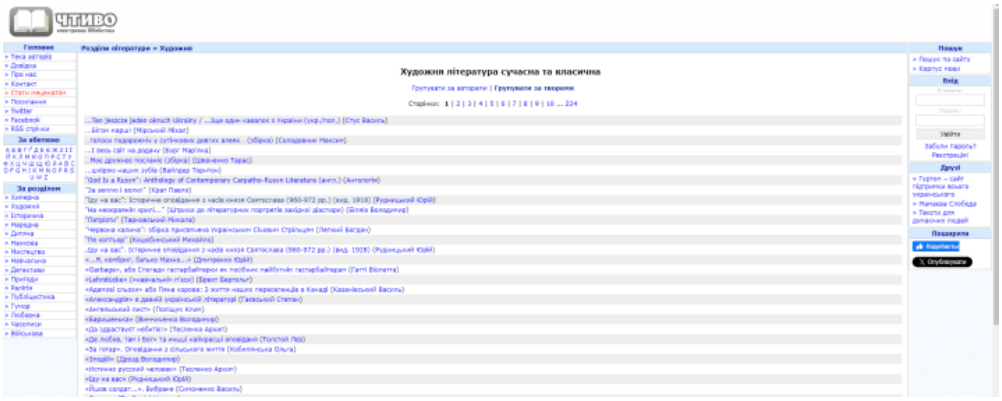


Рисунок 3.2 – Результат пошуку книг за категорією «Художня література» на сайті «Чтиво»

Система надає наступні можливості:

– Chtyvo має велику колекцію книг різних жанрів та тематик, включаючи художню літературу, наукові праці, публіцистику, дитячу

літературу та інше. Користувачі можуть знайти книги на будь-який смак та інтерес;

− система підтримує різні формати електронних книг, такі як EPUB, PDF, MOBI, TXT та інші. Це дозволяє читачам використовувати різноманітні пристрої та програми для читання;

− система дозволяє користувачам створювати закладки та зберігати прогрес читання, що дозволяє зручно повертатися до певного місця в книзі та відслідковувати свій прогрес.

3.1.3 Ukrlib

Укрліб (рис. 3.3) – найбільша електронна бібліотека української літератури, що пропонує книжки українською мовою, а також літературні енциклопедії, біографії, шкільні твори, студентські реферати та короткі анотації до змісту творів української та зарубіжної літератури. Бібліотека була заснована 2000 року. У 2010 році Бібліотеці української літератури було присвоєно Знак якості!

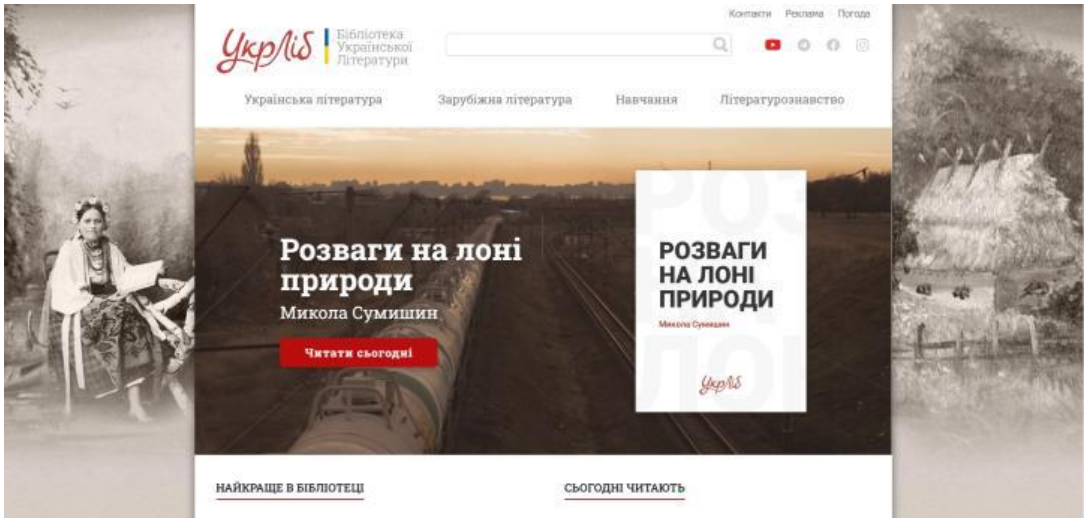


Рисунок 3.3 – Головна сторінка ukrlib

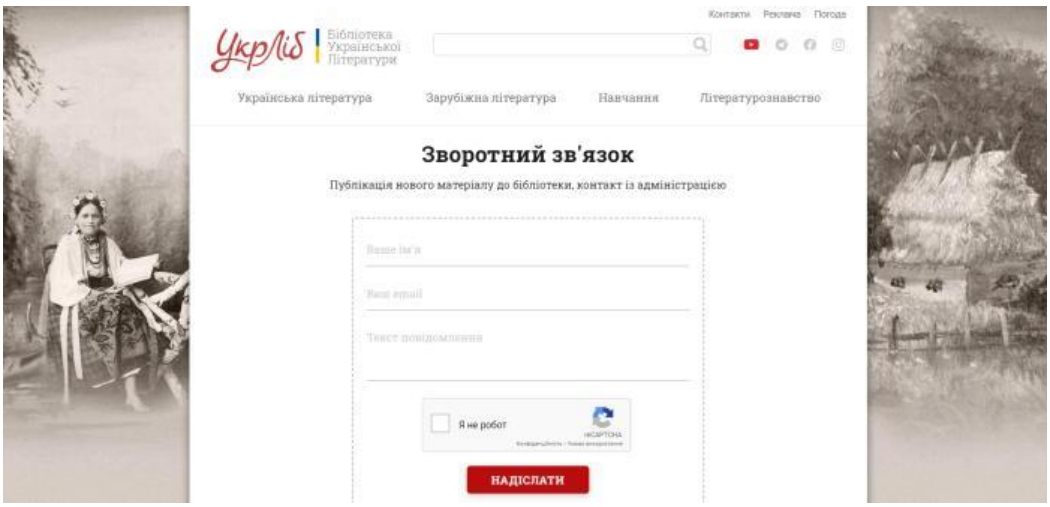


Рис. 3.4 – Сторінка публікування

Рисунок 3.5 – Сторінка читання книги на ukrlib

Система надає наступні можливості:

- різноманітність літератури;
- електронні версії книг;
- пошук і фільтрація;
- можливості читати онлайн;
- збереження закладок і нотаток;
- адаптивний дизайн.

3.1.4 Результат порівнянь

В результаті дослідження декількох аналогів системи можна зробити висновок, що всі представлені варіанти відрізняються один від одного додатковим функціоналом.

Для більш наглядного порівняння створимо таблицю в якій відзначимо наявність/відсутність певних функціональних можливостей у кожній с систем.

Таблиця 3.1 – Результати порівнянь аналогів

Функціональна можливість	**Назва системи**			
	E-bookua	chtyvo	**ukrlib**	**Кваліфікаційна робота**
Різноманітність літератури	+	+	+	+
Зручність читання книг	-	-	+	+
Пошук і фільтрація	+	+	+	+
Дизайн	+	-	+	+
Збереження закладок і нотаток	+	-	+	+

Виходячи з таблиці порівняння можна сказати, що система, котра буде розроблена в рамках кваліфікаційної роботи, матиме такий же базовий функціонал, як і її аналоги. При цьому розроблена система надаватиме можливості переглядати всіх авторів та їх книги, читання книг буде супроводжуватися ілюстраціями.

3.2 Вибір технологій для розробки системи

3.2.1 Вибір архітектурного шаблону

Монолітна архітектура – це традиційна уніфікована модель для проектування програмного забезпечення. Монолітна, в цьому контексті, означає складене все в одному шматку. Монолітне програмне забезпечення

розроблено для автономності; компоненти програми взаємопов’язані та взаємозалежні, а не вільно пов’язані, як це відбувається з модульними програмами. У тісно пов'язаній архітектурі кожен компонент та пов'язані з ним компоненти повинні бути присутніми для виконання або компіляції коду.

Мікросервісний архітектурний стиль (MSA) [3] – це підхід, при якому єдине додаток будується як набір невеликих сервісів, кожен з яких працює у власному процесі і комунікує з іншими використовуючи легковагі механізми, як правило HTTP. Ці сервіси побудовані навколо бізнес-потреб і розгортаються незалежно з використанням повністю автоматизованої середовища. Існує абсолютний мінімум централізованого управління цими сервісами. Самі по собі ці сервіси можуть бути написані на різних мовах і використовувати різні технології зберігання даних.

Оскільки кільки система, що розробляється, має піддаватися масштабуванню для забезпечення оптимального використання ресурсів для кожного замовника, був обраний мікросервісний архітектурний стиль, оскільки саме він, при правильному використанні, надає можливість швидко та з мінімальними витратами масштабувати систему в залежності від її навантаження.

3.2.2 Вибір мови програмування

Наразі існує безліч мов програмування призначених для розробки веб-сервісних додатків. Розглянемо JavaScript.

JavaScript – це мова програмування, яку розробники використовують для створення інтерактивних веб-сторінок. Функції JavaScript можуть поліпшити зручність взаємодії користувача з веб-сайтом: від оновлення стрічки новин в соціальних мережах і до відображення анімації та інтерактивних карт. JavaScript є мовою програмування при розробки скриптів для виконання на стороні клієнта, що робить його однією з базових технологій у всесвітній мережі Інтернет. Наприклад, карусель зображення, що випадає по кліку меню

і динамічно мінливі кольори елементів на веб-сторінці, які ви бачите під час перегляду сторінок в Інтернеті, виконані за допомогою JavaScript.

Історично веб-сторінки були статичними, схожими на сторінки в книзі. Статична сторінка в основному відображала інформацію у фіксованому вигляді і не виконувала всього того, що зараз очікуємо від сучасного сайту. Мова JavaScript виникла як технологія на стороні браузера, щоб зробити веб-додатки більш динамічними. Використовуючи його, браузери могли реагувати на взаємодію з користувачем і змінювати розташування контенту на веб-сторінці.

У міру розвитку мови, розробники JavaScript створили бібліотеки, фреймворки і практики програмування і почали використовувати його за межами веб-браузерів. Сьогодні JavaScript можна використовувати для розробки як на стороні клієнта, так і на стороні сервера. У наступних підрозділах наводимо деякі загальні приклади використання.

Усі мови програмування працюють шляхом перекладу англійського синтаксису в машинний код, який потім виконує операційна система. JavaScript в широкому сенсі можна віднести до категорії скриптових або інтерпретованих мов. Код JavaScript інтерпретується, тобто безпосередньо перекладається в код машинної мови движком JavaScript. В інших мовах програмування компілятор обробляє весь код в машинний на окремому етапі. Таким чином, усі мови сценаріїв є мовами програмування, але не всі мови програмування є скриптовими.

3.2.3 Вибір середовища розробки

Vite.js [10] – це один із інструментів для налаштування середовищ розробки. Найбільше використовується у зв'язці з фреймворками React та Vue.

Visual Studio Code (VS Code) – це редактор коду для різних мов програмування. Він відносно трохи важить, гнучкий і зручний. У ньому можна писати, форматувати і редагувати код на різних мовах.

VS Code не варто плутати з Visual Studio – це IDE, дуже потужна і масштабна, але одночасно з цим великовагова. Назви схожі, оскільки обидва засоби розробки мають одного і того ж творця, але продукти за своєю суттю різні. VS Code менше важить, простіше в освоєнні і підходить в тому числі для початківців розробників.

Редактор коду існує для всіх популярних операційних систем: Windows, Linux та macOS. Він Безкоштовний, на відміну від більшості версій Visual Studio «старший брат».

З VS Code працюють програмісти на різних мовах. Наприклад, їм активно користуються веб-розробники, які пишуть на HTML / CSS, JavaScript, PHP. Але редактор підтримує набагато більшу кількість мов: Python, Go, Ruby, C#, TypeScript тощо. Він працює і з розширеннями і фреймворками для популярних мов – наприклад, з React.js і Vue.js, з мовами стилів SCSS та LESS, які доповнюють CSS.

У теорії користуватися VS Code може Розробник практично на будь-якому з сучасних мов. Але на практиці його застосовують там, де не потрібні потужності повноцінної IDE.

VS Code дозволяє легко писати, форматувати та редагувати код різними мовами. З його допомогою можна швидко створити проект і структуру файлів в ньому, він підсвічує синтаксис коду і допомагає автоматично правити помилки. Він має можливості для налагодження та запуску коду на деяких мовах.

Редактор легко розширюється, тому до перерахованих функцій можна додати нові – досить просто завантажити потрібне доповнення з офіційного каталогу. Доповнення теж поширюються безкоштовно.

3.2.4 Вибір фреймворків і бібліотек

Фреймворк React.js [5] – це фреймворк та бібліотека JavaScript з відкритим кодом, розроблена та підтримувана Facebook та Instagram. Він використовується для швидкого та ефективного створення інтерактивних

інтерфейсів користувача та веб-додатків із застосуванням значно меншої кількості коду, ніж при використанні звичайного JavaScript. У React ви розробляєте свої програми та проекти, створюючи повторно використовувані компоненти, які можна розглядати як незалежні блоки Lego. Ці компоненти являють собою окремі частини остаточного інтерфейсу, які в зібраному вигляді утворюють весь користувальницький інтерфейс програми.

Основною роллю React у програмі є управління відображенням цієї програми так само, як буква V у шаблоні «модель-перегляд-контролер» (MVC), забезпечуючи оптимальне та найефективніше виконання візуалізації. Замість того, щоб розглядати весь інтерфейс користувача як єдине ціле, React.js пропонує розробникам розділити ці складні інтерфейси користувача на окремі повторно використовувані компоненти, які утворюють будівельні блоки всього інтерфейсу користувача. При цьому платформа React.js поєднує в собі швидкість і ефективність JavaScript з більш ефективним методом маніпулювання DOM для швидкого рендеринга веб-сторінок і створення високо динамічних і чуйних веб-додатків.

Node.js [4] – це однопотоковий крос-платформний час виконання з відкритим кодом та бібліотека, яка використовується для запуску веб-додатків, написаних на JavaScript, поза браузером клієнта. Простіше кажучи, Node.js – це програмне середовище, яке дозволяє запускати програми, написані мовою Javascript, поза браузером.

Історично програми, написані на Javascript, на відміну від інших мов програмування, можна було запустити тільки в браузерах, які мали спеціальний вбудований движок виконання коду даної мови. Поза браузером Javascript, можна сказати, не працював.

При розробці Node.js за основу був узятий движок виконання JavaScript під назвою V8, який був створений компанією Google і використовувався в браузері Google Chrome. Так як після створення Node.js Javascript код можна запустити фактично в будь-якому середовищі, за допомогою цієї бібліотеки можна написати не тільки фронтенд, але і серверну частину веб-додатки.

Простіше кажучи, це означає, що цілі сайти тепер можуть працювати з використанням єдиного «стека», що робить розробку і обслуговування набагато більш швидкою і легкою, дозволяючи зосередитися на досягненні бізнес-цілей проекту. Node.js має відкритий вихідний код, тому працювати з ним можна абсолютно безкоштовно. Його і сьогодні продовжує розвивати і покращувати глобальне співтовариство розробників.

Важливо розуміти, що Node.js насправді не є фреймворком і не бібліотекою, як у випадку з традиційним програмним забезпеченням, а часом виконання. Він є легким, гнучким і простим у розгортанні, а всі його функції допоможуть оптимізувати і прискорити ваш додаток.

3.2.5 Вибір бази даних

MongoDB [8] – система керування базами данних, що базується на базі даних NoSQL, яка використовується для зберігання даних у парі ключ-значення. Його робота заснована на концепції документа та колекції.

Система керування базами данних MongoDB можна визначити як документ-орієнтовану систему баз даних, яка використовує концепцію NoSQL. Вона також забезпечує високу доступність, високу продуктивність, разом з автоматичним масштабуванням. Цей продукт з відкритим кодом був розроблений компанією – 10gen у жовтні 2007 року, і досі підтримується. MongoDB має загальну публічну ліцензію (GPL) як безкоштовний інструмент управління базами даних, а також доступний за комерційною ліцензією від виробника. MongoDB також був призначений для роботи з товарними серверами. Компанії різних розмірів у всьому світі у всіх галузях використовують MongoDB як свою базу даних.

Cloudinary [9] – це комплексне рішення для управління зображеннями для веб-сайтів та мобільних додатків.

Додаток підтримує завантаження зображень, хмарне сховище, маніпуляції із зображеннями, оптимізацію зображень для Інтернету та

доставку. Cloudinary пропонує API та гнучкі можливості адміністратора для інтеграції з новими та існуючими веб-додатками та мобільними додатками.

Cloudinary дозволяє завантажувати та зберігати необмежену кількість зображень приватно та безпечно, а також включає автоматичне резервне копіювання та історичні зміни. Ви можете керувати зображеннями, використовуючи широкий спектр опцій, таких як застосування ефектів, зміна розміру, обрізання, виявлення облич, водяні знаки та багато іншого.

Зображення передаються через всесвітню мережу CDN Akamai для швидкої доставки, оптимізованої для будь-якого пристрою. Cloudinary виступає в якості рішення для управління цифровими активами з медіа-бібліотекою, яка використовує RESTful API і SDK для автоматизації управління зображеннями. Ви також можете отримати доступ до розширеної аналітики та звітів, щоб оптимізувати та зрозуміти продуктивність зображень.

3.2.6 Вибір патернів проектування

Dependency injection (DI) або впровадження залежностей представляє механізм, який дозволяє зробити взаємодіючі в додатку об'єкти слабо зв'язаної. Такі об'єкти пов'язані між собою через абстракції, наприклад, через інтерфейси, що робить всю систему більш гнучкою, більш адаптованою і розширюється.

Нерідко для установки залежностей в подібних системах використовуються спеціальні контейнери – IoC-контейнери (Inversion of Control). Такі контейнери служать свого роду фабриками, які встановлюють залежності між абстракціями і конкретними об'єктами і, як правило, управляють створенням цих об'єктів.

Інверсія управління (IoC) – це принцип програмної інженерії, який передає управління об’єктами або частинами програми в контейнер або фреймворк. Найчастіше використовуємо його в контексті об’єктно-орієнтованого програмування.

На відміну від традиційного програмування, в якому наш користувальницький код здійснює виклики до бібліотеки, при використанні IoC розробник передає управління потоком програми та здійсненням викликів до нашого користувацького коду зовнішньому контейнеру. Це надає можливість сконцентруватися на розробці бізнес-логіки, в той час як контейнер буде самостійно піклуватися про моменти створення та видалення об’єктів, перенаправлення викликів з API до відповідних сервісів, передачу повідомлень про помилки і т.д.. Щоб надати таку можливість, фреймворки використовують абстракції із вбудованою додатковою поведінкою. Для додання власної поведінки, потрібно розширити класи фреймворку або вставити власні класи.

Model View Controller – це шаблон дизайну програмного забезпечення, який може бути використаний у багатьох фреймворках із багатьма мовами програмування, як правило, Python, Ruby, PHP, JavaScript тощо. Він широко використовується для проектування веб-додатків та мобільних додатків

MVC популярний у розробці додатків та веб-сайтів, і це один із найбільш широко використовуваних шаблонів дизайну програмного забезпечення для розробки програм та веб-сайтів. Шаблон дизайну контролера моделі перегляду розділяє проблеми на одне з 3 сегментів: модель, вид, контролер.

4 ПРОЦЕС ПРОЕКТУВАННЯ ВЕБ-ДОДАТКУ

4.1 Мета та задачі ІС

Розроблювана система являє собою веб-сервіс, завдяки якому читачам буде зручно читати розміщенні авторами книги, а авторам зручно їх додавати.

Мета ІС – надання можливості створювати та читати книги через єдину систему – онлайн-бібліотеки, а також спрощення процесу пошуку книг.

Створення інформаційної системи (ІС) для сучасної онлайн-бібліотеки з метою забезпечення користувачам зручного та ефективного доступу до літературних творів шляхом автоматизованого розбиття книг на глави та надання короткого змісту кожної глави.

Задачі:

а) реєстрація користувачів:

1) розробити систему реєстрації для читачів та авторів;

2) забезпечити можливість створення та редагування профілю користувача;

б) публікація книг:

1) реалізувати функціонал для авторів щодо публікації їхніх книг на сайті;

2) забезпечити можливість завантаження текстових файлів у форматі, який буде оброблятися системою;

в) автоматичне розбиття книг на глави:

1) розробити алгоритм автоматичного розбиття тексту книги на глави на основі змісту, розділів або інших показників;

2) забезпечити можливість коригування автоматичного розбиття або ручного визначення глав;

г) створення короткого змісту для кожної глави:

1) розробити інструмент для автоматичного або ручного створення короткого змісту для кожної глави книги;

2) забезпечити можливість перегляду та редагування короткого змісту кожної глави;

ґ) інтерфейс для читання книг:

1) розробити зручний інтерфейс для читання книг, що дозволить користувачам мати легку навігацію між главами та переглядати короткі змісті;

2) забезпечити можливість збереження прогресу читання та закладок для кожного користувача;

д) оптимізація відображення: забезпечити оптимізацію відображення контенту для різних типів пристроїв (комп'ютери, планшети, мобільні телефони) з метою забезпечення зручного читання;

е) тестування та вдосконалення:

1) провести тестування системи з метою виявлення помилок та недоліків;

2) здійснювати постійне вдосконалення функціоналу з урахуванням отриманих відгуків користувачів.

4.2 Типи користувачів

Неавтентифікований користувач – користувач, що зайшов на веб-сторінку онлайн-бібліотеки і не пройшов процедуру авторизації / реєстрації, не може читати книги авторів.

Автентифікований клієнт – клієнт пройшов процедуру авторизації, має всі права користувача, і може читати книги.

Автор – автор онлайн-бібліотеки, має всі права неавдентифікованого клієнта, також має доступ до панелі редагування, де може створювати власні книги, додавати фото та категорію, видаляти або оновлювати інформацію книги. Може переглядати сторінки з усіма авторами, своїм профілем, бачити свою кількість написаних книг.

Читач – має всі права автора, але він може лише читати книги (не створювати).

4.3 Функціональні вимоги

а) реєстрація користувачів:

1) система повинна забезпечити можливість реєстрації на сайті як авторів, так і читачів;

2) для реєстрації користувачам потрібно ввести обов'язкові дані, такі як ім'я, електронну пошту та пароль;

б) авторська публікація книг:

1) автори повинні мати можливість завантажувати свої книги на сайт;

2) після завантаження книги система автоматично розбиває її на глави;

в) створення короткого змісту глави:

1) для кожної глави книги повинен створюватися короткий зміст;

2) користувачі можуть створювати короткий зміст вручну або скористатися автоматичною генерацією;

г) інтерфейс для читання книг:

1) система повинна надавати зручний інтерфейс для читання книг, де кожна глава буде доступна окремо;

2) на сторінці кожної глави повинен бути відображений короткий зміст, який може бути прихований або розгорнутий за бажанням користувача;

ґ) навігація та пошук:

1) система повинна надавати можливість швидко переходити між главами книги;

2) користувачі можуть використовувати функцію пошуку для знаходження конкретної книги або глави;

д) збереження прогресу читання: система повинна автоматично зберігати прогрес читання кожного користувача, щоб вони могли продовжити з того місця, де зупинилися;

е) адміністративні можливості: адміністраторам системи потрібно мати можливість керувати контентом, видаляти або редагувати книги та глави;

є) користування на різних пристроях: система повинна бути адаптована до різних типів пристроїв (комп'ютери, планшети, мобільні телефони) з метою забезпечення зручного читання;

ж) комунікація з користувачами: система повинна надавати можливість зворотного зв'язку для користувачів, наприклад, через електронну пошту або онлайн-чат.

4.4 Нефункціональні вимоги

До нефункціональних вимог можна віднести:

- пошук і фільтрація;
- збереження стану читання;
- безпека і захист даних;
- швидкість та продуктивність;
- підтримка різних форматів.

Для детального розгляду побудуємо діаграми прецедентів (рис. 4.1)

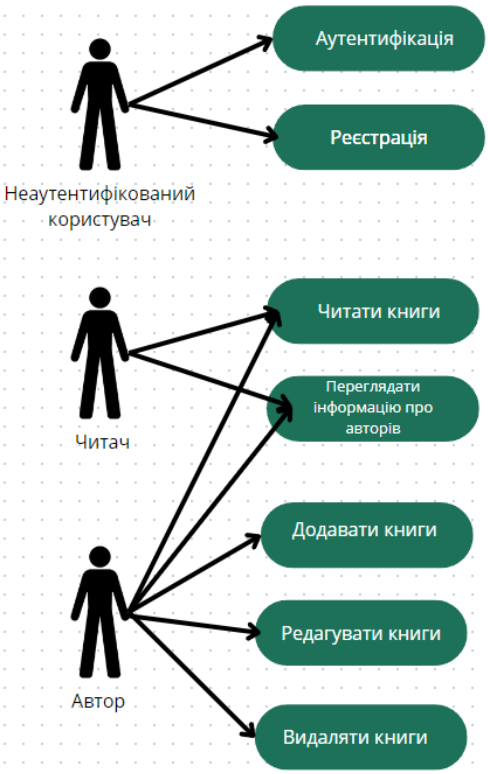


Рисунок 4.1 – Діаграма прецедентів верхнього рівня

4.5 Ідентифікація архетипу ІС

Інформаційна система буде мати мікросервісну архітектуру, де будуть такі мікросервіси, як головна де зображені книги від популярних до останніх доданих, сторінка «Книги», що відповідає за відображення всіх доданих книг авторами, «Всі автори», де можна переглянути всіх авторів, «Про нас» – на цій сторінці написана інформація про сайт, «Панель керування» що відповідає за створення, оновлення, та видалення книг, та надає можливість переглянути свій профіль.

4.6 Користувацький інтерфейс (UI View)

В таблиці 4.1 розібрано пояснення позначок сторінок на розроблених макетах та на діаграмі переходів між сторінками.

Таблиця 4.1 – Пояснення позначок сторінок на розроблених макетах та на діаграмі переходів між сторінками.

Числове позначення сторінки	Пояснення
1	Головна сторінка
2	Книги
3	Всі автори
4	Про нас
5	Панель керування
6	Вийти
7	Створити книгу
8	Мої книги
9	Мій профіль
10	Оновити книгу

Головна сторінка (рис 4.2) – перше, що бачить користувач, який зайшов на веб-сервіс. На цій сторінці він побачить популярні та останні додані книги.

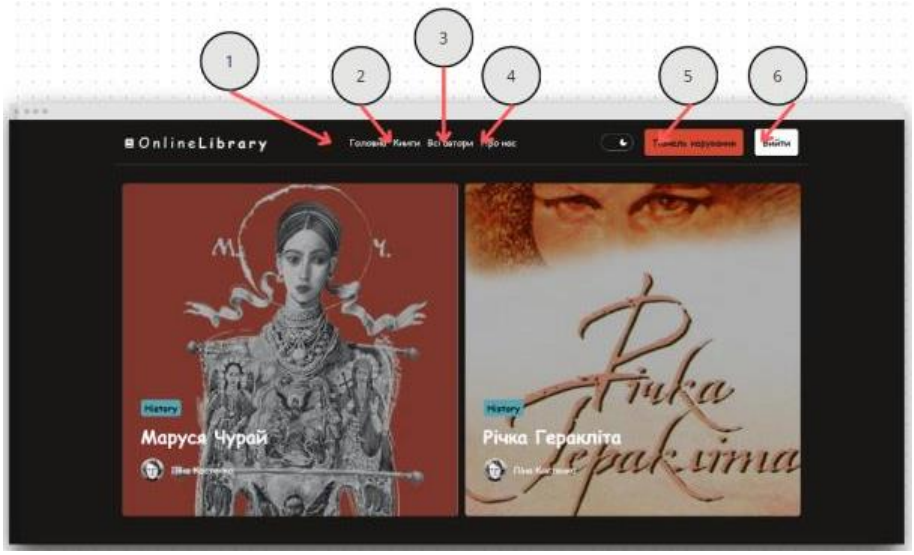


Рисунок 4.2 – Головна сторінка сервісу

Сторінка «Книги» (рис 4.3) – на цій сторінці користувач може продивитися всі книги які були публіковані авторами цього сайту.

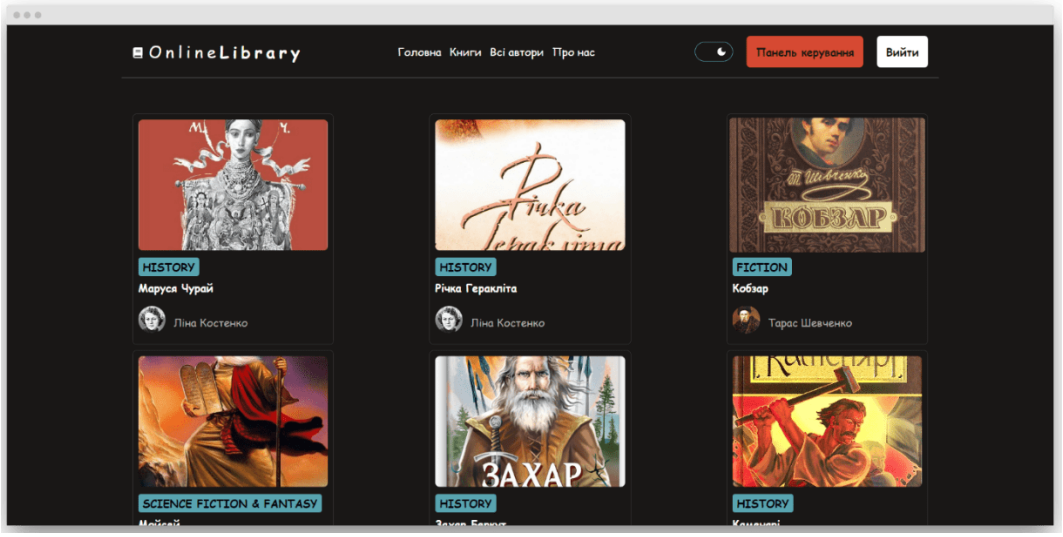


Рисунок 4.3 – Сторінка «Книги»

Сторінка «Всі автори» (рис 4.4) – на цій сторінці відображаються всі зареєстровані автори на цьому сайті.

Особистий кабінет автора (рис 4.5) – на цій сторінці зліва відображаються різні функції можливості переглянути свій профіль, створити книгу, вийти і т.д.

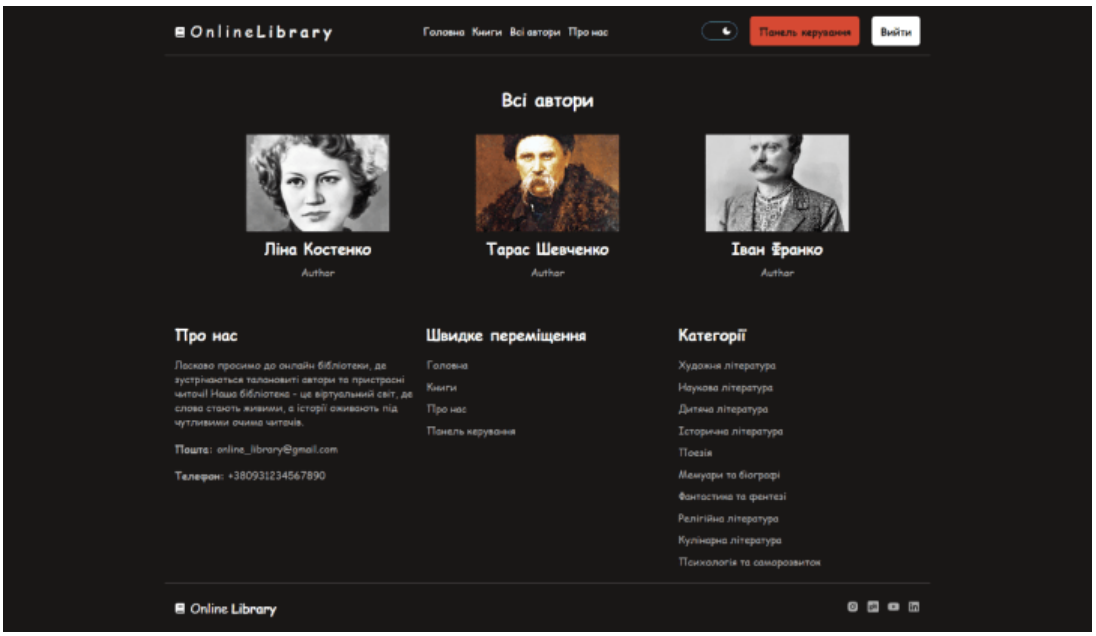


Рисунок 4.4 – сторінка «Всі автори»

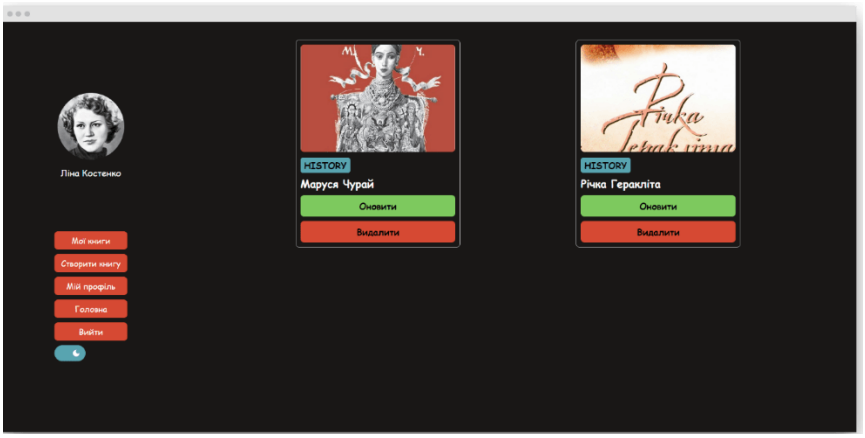


Рисунок 4.5 – сторінка «Панель керування» автора

Сторінка Створення книги (рис 4.6) – на цій сторінці автор може додавати нову книгу, обравши її категорію, фото, написати короткий вступ, назву, частину, і додати фотографії, щоб оживити свою публікацію.

Сторінка оновлення книги (рис 4.7) – автор може виправити свою публікацію, оновивши інформацію про неї та її вміст.

Сторінка перегляду профілю автора (рис 4.8) – ця сторінка містить ту інформацію про автора, котру він вводив при реєстрації.

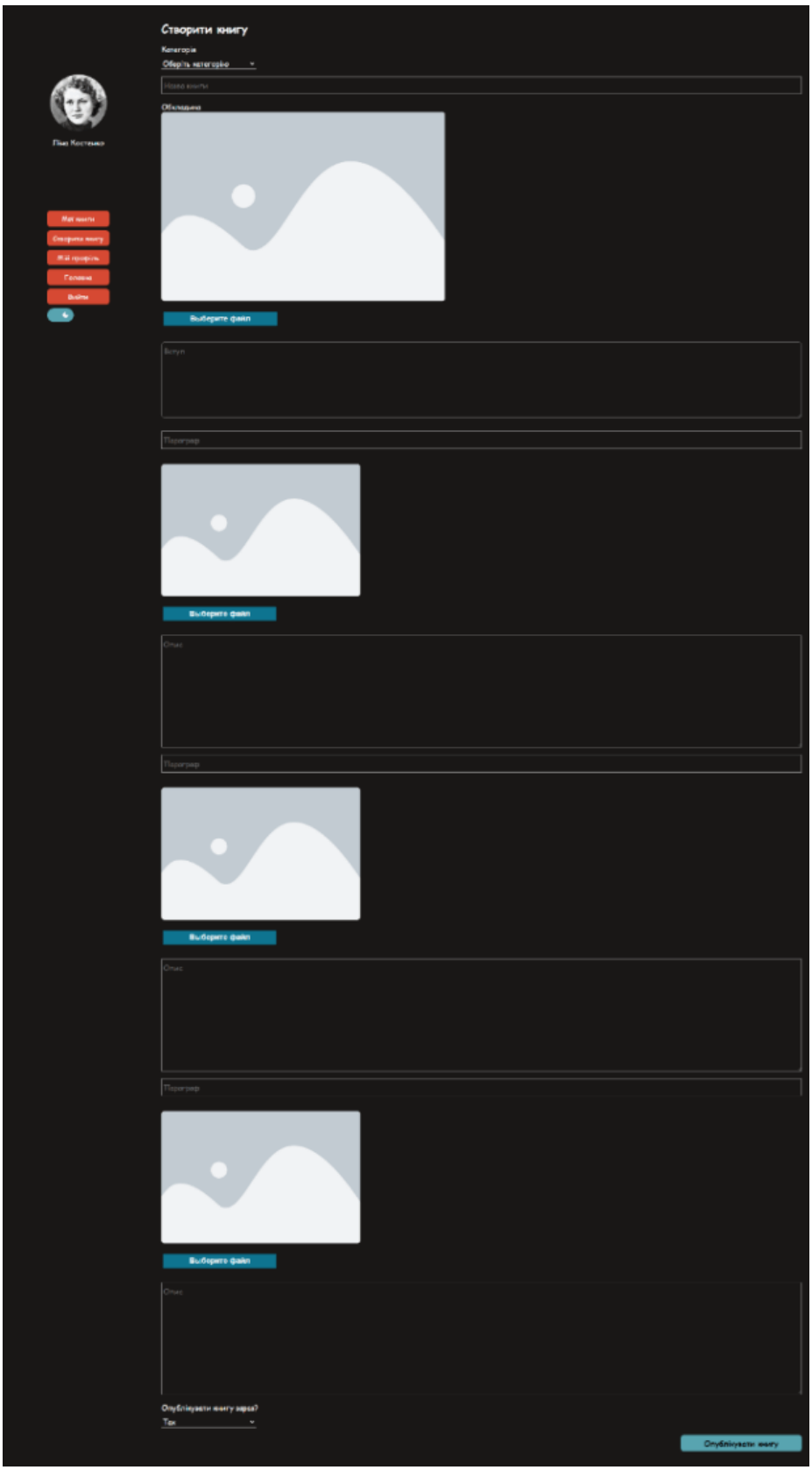


Рисунок 4.6 – Сторінка додавання книги

Рисунок 4.7 – сторінка оновлення публікації

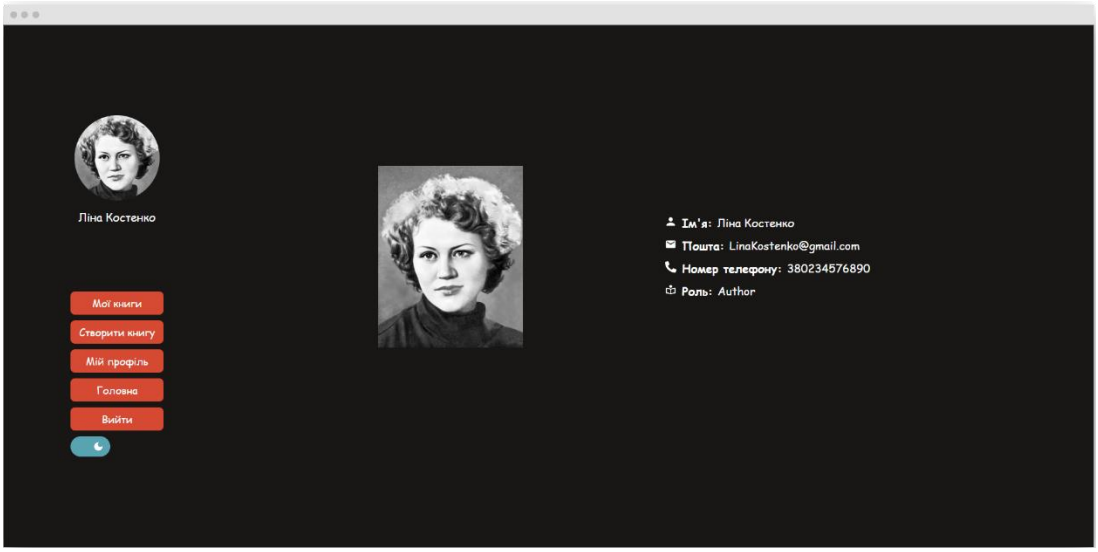


Рисунок 4.8 – Сторінка профілю автора

Приведемо діаграму переходів між основними сторінками додатку на рисунку 4.9. Визначення числовим позначенням на діаграмі представлено в таблиці 4.1. Зеленим кольором позначені публічні сторінки, тобто ті сторінки що може обачити читач онлайн-бібліотеки. Жовтим – приватні сторінки, до яких мають доступ лише автори онлайн-бібліотеки, після проходження процедури авторизації.

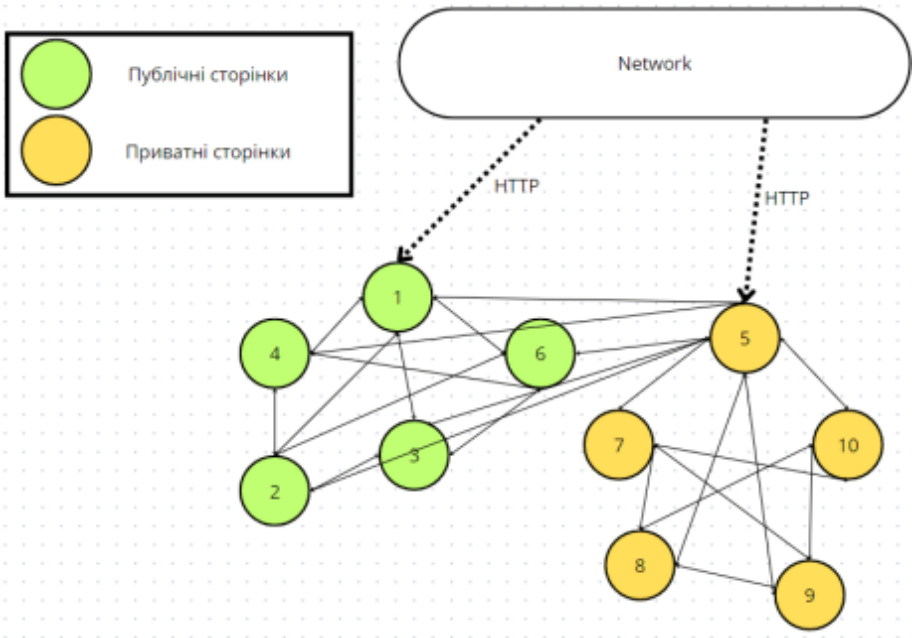


Рисунок 4.9 – Діаграма переходів між сторінками сервісу

4.7 Логічне уявлення про ІС (Logical View)

Для уявлення майбутньої системи, побудуємо логічну діаграму зв'язку компонентів системи (рис. 4.10). Модель складається з оператора системи, який взаємодіє з UI частиною системи, яка відсилає запити на серверну частину. Далі, все залежить від запиту – у відповідності до того котрий із типів даних запитується запит буде направлено до відповідного сервісу

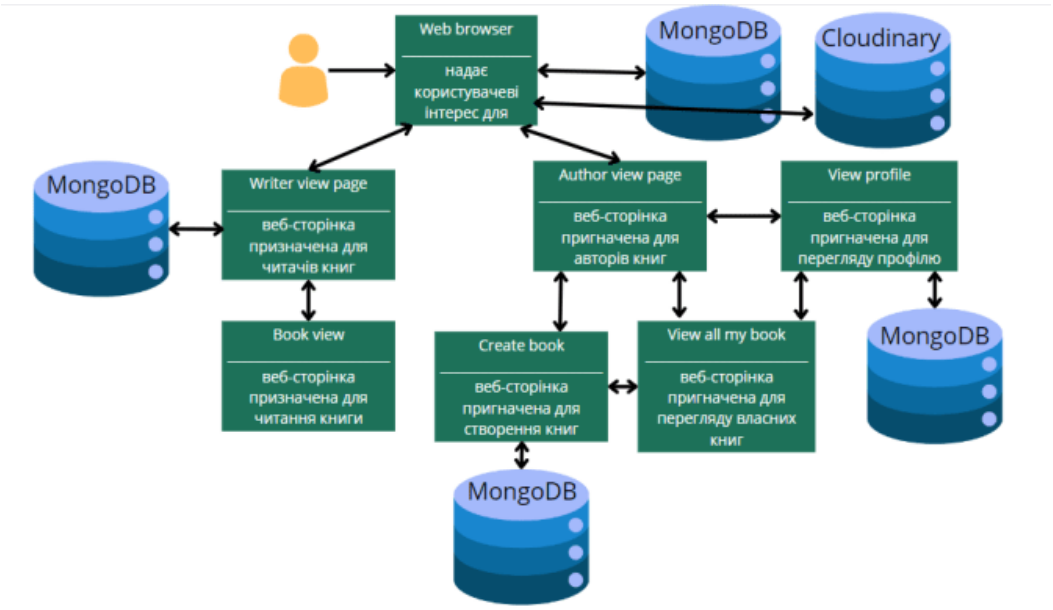


Рисунок 4.10 – Схема логіки роботи системи та зв’язку компонентів

4.8 Уявлення розгортання ІС (Deployment View)

Для своєї роботи додаток використовує веб-браузер і мережу інтернет, тобто без обох цих умов робота додатка неможлива. Додаток працює за рахунок того, що користувач в браузері вибирає необхідний йому функціонал, запит відправляється на відповідний сервіс, далі за допомогою фреймворку Node.js відбувається обмін даними між мікросервісами.

Уявлення розгортання інформаційної системи (Deployment View) відображає структуру та розміщення компонентів системи на обладнанні.

Додаток, описаний вище, розглядається з точки зору розгортання:

− додаток базується на веб-браузері та мережі Інтернет – це означає, що його робота передбачає наявність активного з'єднання з Інтернетом і можливість взаємодії з користувачем через веб-інтерфейс;

− компоненти додатка розташовані на серверах, що забезпечують його функціонування – переважна більшість логіки додатка виконується на боковому сервері (backend), який працює на основі фреймворку Node.js. Цей бекенд управляє мікросервісами, що відповідають за різні аспекти функціоналу додатка;

− клієнтська частина додатка (frontend) відображається у веб-браузері користувача – вона виконується на боковому сервері, але виводиться та інтерактивно взаємодіє з користувачем через веб-інтерфейс;

− розгортання додатка передбачає наявність необхідних ресурсів, таких як сервери для бекенду та сервісів мікросервісів, а також веб-сервер для відображення фронтенду – для забезпечення високої доступності та масштабованості можуть використовуватися різноманітні технології та архітектурні підходи, такі як кластеризація серверів, контейнеризація тощо.

Таке розгортання дозволяє додатку працювати ефективно та надійно, забезпечуючи користувачам швидкий доступ до його функціоналу через веб-браузер.

4.9 Уявлення слоїв ІС (Design View)

В основі кожного з мікросервісів системи лежить патерн MVC, згідно з цим патерном система має кілька шарів: шар представлень, шар контролерів та шар моделей. Оскільки самі мікросервіси працюють у REST-режимі, то у якості шару представлень виступають зовнішні фронтенд сторінки, в самих мікросервісах знаходяться лише шари контроллерів і моделі.

Уявлення слоїв інформаційної системи (Layered View) розкриває структуру кожного мікросервісу згідно з патерном Model-View-Controller (MVC), де кожен шар відповідає за певні аспекти функціонування системи.

Розглянемо розподіл за шарами для кожного мікросервісу:

− шар моделей (Model Layer) – в цьому шарі розташовані моделі даних, які відповідають за представлення бізнес-логіки та структуру даних. Кожен мікросервіс має свій власний набір моделей, які використовуються для взаємодії з даними та забезпечення їх консистентності;

− шар контролерів (Controller Layer) – в цьому шарі знаходяться контролери, які приймають запити від клієнтів та взаємодіють з моделями для обробки цих запитів. Контролери відповідають за приймання та обробку вхідних даних, виконання необхідних операцій та передачу результатів відповідно до логіки додатку;

− шар представлень (View Layer) – оскільки мікросервіси працюють у режимі REST, то зовнішні фронтенд-сторінки виступають у ролі шару представлень. Ці фронтенд-сторінки взаємодіють з мікросервісами через їх API, надсилаючи запити та отримуючи відповіді для відображення вмісту для користувачів.

Кожен мікросервіс має власний набір шарів, що дозволяє забезпечити розділення відповідальностей та покращити модульність системи. Цей підхід дозволяє легко масштабувати та підтримувати кожен мікросервіс окремо, а також реагувати на зміни вимог до функціональності.

4.10 Уявлення процесів ІС (Process View)

Уявлення процесів інформаційної системи (ІС) для сучасної онлайн-бібліотеки є ключовим елементом у забезпеченні зручного та ефективного взаємодії користувачів з ресурсом. Основна мета полягає в тому, щоб забезпечити користувачам можливість реєстрації, публікації, пошуку та зручного читання книг:

− перший етап взаємодії з системою – реєстрація. Користувачі мають можливість створювати свої облікові записи, вказуючи особисту інформацію та обираючи параметри доступу до сервісу;

− автори книг мають змогу завантажувати свої твори на платформу. Після завантаження книги автоматично розбиваються на глави. Для кожної глави генерується короткий зміст, який формується за допомогою інтелектуальних алгоритмів обробки тексту;

− користувачі мають можливість здійснювати пошук книг за різними критеріями, такими як автор, назва, жанр тощо. Додатково можуть бути встановлені фільтри для зручного вибору книг за популярністю, рейтингом чи іншими параметрами;

− користувачам надається зручний інтерфейс для читання книг. Кожна сторінка книги містить короткий зміст глави, що дозволяє швидко ознайомитися з її змістом. Для забезпечення комфортного читання можуть бути доступні різні налаштування шрифту, розміру тексту та інші параметри;

− крім того, користувачі можуть залишати відгуки та оцінювати книги, обмінюватися рекомендаціями та думками з іншими читачами. Це сприяє формуванню активної спільноти та покращенню якості обслуговування.

В цілому, процеси ІС для сучасної онлайн-бібліотеки спрямовані на забезпечення користувачам доступу до якісного контенту та зручної взаємодії з платформою. Реалізація автоматичного розбиття книг на глави та створення короткого змісту додає додаткову цінність для користувачів, забезпечуючи їм можливість швидко зорієнтуватися в змісті книги та ефективно використовувати час при читанні.

4.11 Уявлення даних ІС (Data View)

Система буде мати дві бази даних. Спочатку розглянемо базу користувачів сайту. На рис. 4.11 можна побачити об’єкти зареєстрованих авторів.

У базі даних зареєстрованих авторів сайту в MongoDB можна побачити наступну інформацію про кожного автора:

− ідентифікатор (id) – унікальний ідентифікатор, який однозначно ідентифікує кожного автора в базі даних;

```
_id: ObjectId('66197ed74c7b1e13f7cf0373')
name : "Ліна Костенко"
email : "LinaKostenko@gmail.com"
phone : 380234576890
avatar : Object
education : "PhD"
role : "Author"
password : "$2b$10$BAvnbiRqcCqffhZC5//YaeSc6l5fhK43MKVWlWZxLtFxOvWFjtn1i"
createdOn : 2024-04-12T18:35:03.231+00:00
__v : 0
```


Рисунок 4.11 – Бази даних зареєстрованих авторів

– ім'я (name) – повне ім'я автора, яке використовується для ідентифікації та представлення на сайті;

– електронна пошта (email) – контактна інформація автора для спілкування та отримання повідомлень з сайту;

– телефон (phone) – номер телефону автора, якщо він був наданий при реєстрації;

– аватар (avatar) – зображення або посилання на аватар автора, яке використовується для візуальної ідентифікації;

– освіта (education) – інформація про освітній рівень автора, що може бути корисною для користувачів, які шукають книги за авторством певного фахівця;

– роль (role) – вказує на те, що автор має доступ до публікації своїх книг на сайті;

– пароль (password) – хешований пароль автора для захисту доступу до облікового запису;

– дата створення (createdon) – дата та час, коли обліковий запис автора був створений у системі.

У базі даних написаних авторами книг на сайті в MongoDB (рис. 4.12) можна побачити наступну інформацію про кожну книгу:

– ідентифікатор (id) – унікальний ідентифікатор, який однозначно ідентифікує кожну книгу в базі даних;

– назва (title) – назва книги, яка використовується для її ідентифікації та представлення на сайті;

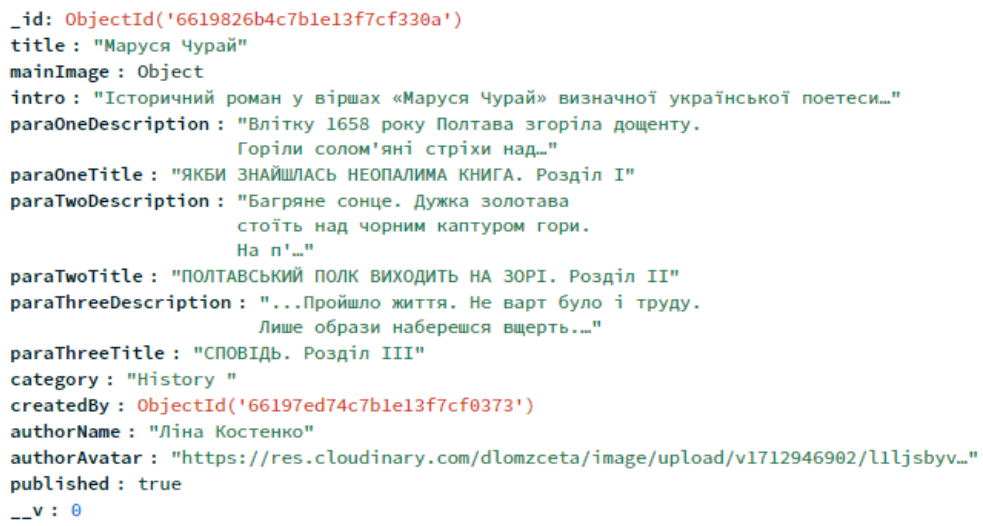


Рисунок 4.12 – База даних доданих книг авторами

– головне зображення (mainImage) – зображення обкладинки або ілюстрація, що ілюструє книгу, і використовується для візуального представлення книги на сайті;

– вступ (intro) – короткий опис або вступ до книги, який надає загальне уявлення про її зміст;

– описи глав (paraoneDescription, paraTwoDescription, paraThreeDescription) – короткі описи кожної глави книги, які можуть використовуватися для створення коротких змістів глав для зручного читання;

– назви глав (paraoneTitle, paraTwoTitle, paraThreeTitle) – назви кожної глави книги, які вказують на їх зміст та послідовність;

– категорія (category) – категорія, до якої належить книга, що допомагає користувачам знаходити книги за певними тематиками;

– автор (createdBy) – інформація про автора книги, включаючи ім'я та ідентифікатор;

– опублікована (published) – показник, що вказує на те, чи доступна книга для читання на сайті.

4.12 Уявлення інтерфейсів ІС (Interface View)

Для взаємодії з системою користувач буде використовувати веб-інтерфейс. Користувач буде відправляти http-запит, а система буде приймати його в контролерах і обробляти в сервісах і повертати належний результат. Для спілкування сервісів між собою буде використовуватись протокол RSocket, що підтримує асинхронну неблокуючу push-pull модель взаємодії, це дозволить значно зменшити навантаження на сервіси тим самим зменшуючи витрати на підтримання серверів. Обмін даними між сервісами та БД буде відбуватися за рахунок протоколу JDBC випадку MongoDB та Cloudinary.

4.13 Уявлення безпеки ІС (Security View)

Уявлення безпеки інформаційної системи (ІС) для сучасної онлайн-бібліотеки включає в себе комплекс заходів, спрямованих на захист конфіденційності, цілісності та доступності даних, а також забезпечення безпеки користувачів та запобігання можливим загрозам інформаційній системі. Для досягнення цих цілей існують наступні аспекти безпеки ІС:

− забезпечення механізмів автентифікації, щоб переконатися, що лише авторизовані користувачі мають доступ до функціональності сайту;

− збереження особистої інформації користувачів у безпечному середовищі та забезпечення дотримання законодавства про захист персональних даних;

− впровадження заходів з обмеження доступу, шифрування даних, виявлення та відновлення збоїв для запобігання кібератак та злому системи;

− проведення регулярного резервного копіювання даних для запобігання втрати інформації та можливості швидкого відновлення в разі аварійних ситуацій;

− постійний моніторинг системи на предмет виявлення потенційних загроз та вразливостей і вжиття заходів для їх усунення;

− використання антивірусного програмного забезпечення та заходів забезпечення безпеки мережі для захисту від шкідливих програм та вірусів;

− проведення навчання та підвищення кваліфікації персоналу щодо впровадження та дотримання політик безпеки інформаційної системи.

4.14 Інфраструктурне уявлення ІС (Infrastructure View)

Для розгортання системи буде використаний хостинг, який має наступні характеристики.

Процесор – AMD Ryzen 9 3900X частотою 3800-4600 МГц і 12 ядрами.

ОЗУ DDR4 2600 MHz 16Gb.

Операційна система – Red Hat Enterprise Linux 64–bit.

З доступом до мережі Інтерент і статичний IP.

З боку користувача очікується сучасний web–browser, з підтримкою HTML5.

4.15 Опис стеку технологій

Для частини з Backend використовувалися такі технології:

− Node.js та Express.js – для створення серверної частини додатку та обробки запитів від клієнтів;

− MongoDB з Mongoose – для збереження даних про користувачів, авторів, книги та глави у базі даних;

− Bcrypt – для шифрування паролів користувачів перед збереженням у базі даних;

− Jsonwebtoken – для генерації та перевірки JWT токенів для автентифікації користувачів;

− Dotenv – для завантаження конфігураційних змінних середовища з файлу .env;

− Express-fileupload – для обробки завантаження файлів, таких як книги у форматі PDF;

− Cloudinary – для збереження та управління зображеннями обкладинок книг у хмарному сервісі.

Frontend:

− React.js – для створення інтерактивного та динамічного інтерфейсу користувача;

− React Router – для маршрутизації та навігації між різними сторінками додатку;

− Axios – для здійснення HTTP-запитів до серверної частини додатку;

− Draft.js та React Quill – для створення візуального редактора для авторів книг;

− React Spinners – для відображення анімації завантаження під час обробки запитів;

− Chart.js та React Chartjs-2 – для створення графіків та діаграм для відображення статистики про читання книг;

− React Icons – для використання набору піктограм та іконок у користувальному інтерфейсі;

− React Multi Carousel – для створення каруселів для відображення списку книг та авторів у привабливому форматі.

5 ПРОЦЕС РОЗРОБКИ ВЕБ-ДОДАТКУ

5.1 Уявлення про структуру проекту

Структура проекту для сучасної онлайн-бібліотеки буде організована з урахуванням функціональних вимог та взаємозв'язків між різними компонентами.

Основні складові цієї структури включають бекенд (backend) та фронтенд (frontend) частини.

У бекенді використовуються такі технології, як Node.js, Express.js [6] та MongoDB з Mongoose для створення серверної частини додатку та збереження даних.

Фронтенд реалізований за допомогою React.js та інших бібліотек, таких як React Router для навігації, Axios [7] для взаємодії з сервером, та React Icons [11] для використання іконок у користувацькому інтерфейсі. Файл package.json фронтенду містить залежності та скрипти для розробки та збірки фронтенду.

Кожна з цих складових має свою власну структуру. Наприклад, у бекенді є теки для маршрутів (routes), контролерів (controllers), моделей (models) та служб (services), а у фронтенді – компоненти (components), роутери (routers), стилі (styles) та інші. Крім того, в проекті є окремі теки для конфігураційних файлів, утиліт та інших ресурсів.

Ця структура дозволяє ефективно організувати роботу розробників, забезпечуючи чітку розділеність між фронтендом та бекендом, а також між різними функціональними частинами додатку.

5.2 Уявлення про сутності інформаційної системи

Уявлення про сутності інформаційної системи для сучасної онлайн-бібліотеки включає наступні компоненти:

− користувачі (Users): автори (Authors) – користувачі, які публікують свої книги у бібліотеці; читачі (Readers) – користувачі, які переглядають та читають книги у бібліотеці;

− книги (Books): книги, які можуть бути публіковані авторами та читані читачами. Після публікації книга автоматично розбивається на глави;

− глави (Chapters): окремі частини книги, на які автоматично розбивається публікована книга. Для кожної глави складається короткий зміст;

− інтерфейс читання (Reading Interface): веб-інтерфейс, який надає зручний доступ до книг та їх глав для читачів. Можливість перегляду повного змісту книги або короткого змісту глави;

− автоматичне розбиття книги (Automatic Book Splitting): механізм, який автоматично розбиває книгу на глави після її публікації;

− генерація коротких змістів (Summary Generation): процес генерації короткого змісту для кожної глави книги. Може виконуватися автоматично або ручно користувачем;

− система реєстрації та автентифікації (Registration and Authentication System): система, яка дозволяє користувачам реєструватися та входити в систему для доступу до функціональності бібліотеки;

− база даних (Database): зберігання інформації про користувачів, книги, глави та інші дані, необхідні для роботи системи.

Ці сутності утворюють основу функціональності та можливостей сучасної онлайн-бібліотеки, забезпечуючи зручний та ефективний доступ до літературних ресурсів для користувачів.

Файл CreateBlog.jsx (рис. 5.1) який відповідає за створення нових публікацій у веб-блозі, який є однією з основних функціональних можливостей онлайн-бібліотеки. Нижче розглянемо його детально:

− використовуються хуки стану для збереження даних, введених користувачем під час створення нової публікації, таких як категорія, назва, вступ, дані глав тощо;

Рисунок 5.1 – Код методу CreateBlog

– компонент містить функції обробки подій, такі як вибір обкладинки, введення даних для глав, вибір категорії, публікація блогу тощо;

– виводиться форма, де користувач може ввести дані для нової публікації, такі як категорія, назва, вступ та інші дані для кожної глави;

– користувач може вибрати обкладинку для публікації та додати дані для кожної глави, включаючи зображення та опис;

– після заповнення форми та натискання кнопки «Опублікувати книгу», дані відправляються на сервер для збереження в базі даних. Після успішної публікації форма очищується, і користувач отримує повідомлення про успішну публікацію;

– при виборі зображень для обкладинки та глав, користувач отримує попередній перегляд цих зображень перед їх завантаженням;

– відбувається перевірка коректності введених даних перед відправленням на сервер. У випадку помилки користувач отримує відповідне повідомлення;

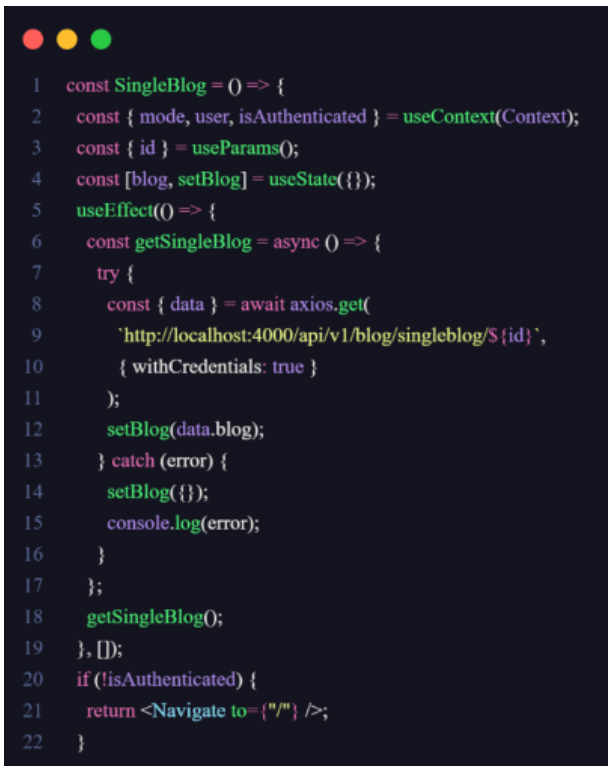
– після відправлення даних на сервер і отримання відповіді, відбувається обробка цієї відповіді. У випадку успішної публікації користувач отримує повідомлення про це, в іншому випадку – повідомлення про помилку.

Розглянемо детальніше файл Blogs (рис. 5.2). Тут на сторінці відбувається відображення усіх останніх доданих авторами книг.

```
import React, { useContext } from "react";
import LatestBlogs from "../miniComponents/LatestBlogs";
import { Context } from "../../main";

const Blogs = () => {
  const { mode, blogs } = useContext(Context);

  return (
    <article className={mode === "dark" ? "dark-bg" : "light-bg"}>
      <LatestBlogs blogs={blogs} title={"Blogs"} />
    </article>
  );
};

export default Blogs;
```

Рисунок 5.2 – Код методу getAvailableRooms

Розглянемо детальніше файл SingleBlog (рис. 5.3). Цей файл відповідає за перегляд сторінки з книгою для її читання.



```
const SingleBlog = () => {
  const { mode, user, isAuthenticated } = useContext(Context);
  const { id } = useParams();
  const [blog, setBlog] = useState({});
  useEffect(() => {
    const getSingleBlog = async () => {
      try {
        const { data } = await axios.get(
          `http://localhost:4000/api/v1/blog/singleblog/${id}`,
          { withCredentials: true }
        );
        setBlog(data.blog);
      } catch (error) {
        setBlog({});
        console.log(error);
      }
    };
    getSingleBlog();
  }, []);
  if (!isAuthenticated) {
    return <Navigate to={"/"} />;
  }
```

Рисунок 5.3 – Код SingleBlog

Основні функціональні можливості коду включають:

– автентифікація користувача – компонент використовує useContext, щоб отримати дані про автентифікованого користувача з контексту. Якщо користувач не автентифікований, відбувається перенаправлення на головну сторінку за допомогою компонента Navigate;

– отримання інформації про книгу – компонент використовує хук useParams, щоб отримати ідентифікатор книги з URL. Після цього виконується HTTP-запит до сервера за допомогою бібліотеки axios, щоб отримати дані про окремий блог за його ідентифікатором. Отримані дані про блог зберігаються у стані компонента за допомогою useState;

– відображення даних про книгу – після отримання даних про блог відбувається їх відображення на сторінці. Відображається категорія книги, заголовок, автор, обкладинка та основний зміст книги. Книга автоматично

розбивається на глави, а для кожної глави відображається її назва, зображення (якщо воно є) та короткий опис. Даний підхід дозволяє користувачам легко переглядати вміст книги та отримувати загальне уявлення про її зміст, переглядаючи короткі змістовні відомості кожної глави;

− підтримка різних тем – класи CSS для відображення компонента змінюються залежно від теми, що дозволяє створити темну та світлу теми для зручності користувачів.

Загально, цей компонент відображає інформацію про окрему книгу, розбиваючи її на глави та надаючи користувачам зручний інтерфейс для її читання. Це відповідає вимогам сучасної онлайн-бібліотеки, яка надає функціональність публікації та читання книг, а також можливість швидкого перегляду їх змісту.

Оскільки у мікросервісна архітектура передбачає спілкування між різними мікросервісами сервісами через веб-мережу виникає необхідність у постійній роботі з веб-запитами. Для того щоб спростити процедуру запиту даних з іншого сервісу та виключити необхідність у кожному щоразу визначати механізми обробки запиту у кожному місці виклику при розробці був використаний патерн Фасад. Завдяки цьому стало можливо об’єднати механізми усіх запитів до конкретного зовнішнього сервісу у одному класі з простим інтерфейсом, ховаючи від клієнтського класу усі механізми обробки запитів.

Наведемо приклад реалізації патерну Фасад на рисунку 5.4.

Як можна побачити на рисунку 5.4 клас RoomClassServiceRSocketAdapterImpl має простий інтерфейс, що повертає звичайні об’єкти, натомість у самому класі відбуваються процеси, отримання адреси зовнішнього сервісу з роутингового сервісу, відправки запиту з первини параметрами до зовнішнього сервісу номерного фонду, отримання відповіді та її серіалізація до внутрішніх об’єктів сервісу, а у разі виникнення помилки – її обробка.

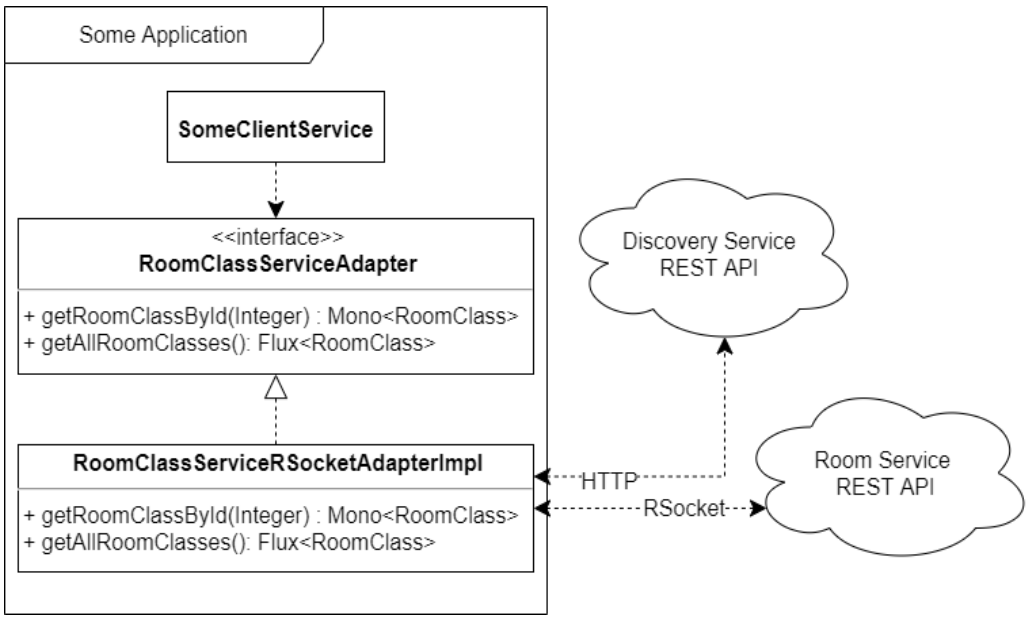


Рисунок 5.4 – Реалізація патерну Фасад

5.3 Управління програмним кодом системи

На роль системи контролю версій був вибраний Git, а для розміщення віддаленого репозиторію сервіс GitHub.

Таблиця 5.1 – Метрика роботи в системі GitHub

Метрика	Значення
Кількість комітів у master branch	16
Загальна кількість бранчів	8
Кількість закритих PR	7

5.4 Розрахунок метрики програмного коду системи

Для визначення метрик коду був використаний плагін git code statistic для visual studio code. Метрики коду зображені в таблиці 5.2. Значення усіх метрик, за винятком відсотка коментарів у коді знаходяться в нормі. Такий мала кількість коментарів зумовлена стилем написання коду, так назви класів, методів та змінних надають достатньо інформації для розуміння процесів. Зайві коментарі будуть лише перевантажувати код.

Таблиця 5.2 – Метрика програмного коду

Метрика	Значення
Загальні кількість строчок коду в проекті	4,669
Середня кількість строчок коду в класі	60
Максимальна кількість строчок коду в одному класі	250
Середня кількість строчок коду в одному методі	14
Максимальна кількість строчок коду в одному методі	31
Максимальна глибина дерева наслідування	2
Середня цикломатична складність методів	1,35
Максимальна цикломатична складність методів	2
Відсоток коментарів у коді	7%
Покриття коду модульними тестами	67%

5.5 Контрольний список по якості реалізації системи

При розробці проекту ми намагалися дотримуватися більшості критеріїв якості системи для відображення виконаних критеріїв зіставимо таблицю 5.3.

Таблиця 5.3 – Контрольний список по якості реалізації

Твердження	Відповідь	Пояснення, якщо відповідь ні
Використання Dependency Injection	так	–
Використання логірування	так	–
Використання модульного тестування	так	–
Захист від Javascript/XSS ін’єкцій	так	–
Використання валідації даних в полях вводу	так	–
Використання інсталяторів чи магазинів	ні	система є веб–додатком
Використання синхронізації даних при багато поточному застосунку	так	–
Підтримка глобалізації, локалізації ІС	ні	–
Відсутність вбудованих в програмний код конфігурацій програми	так	–
Присутність UI патернів при розробці	так	–
Coding style guide lines для вибраної мови	так	–
Використання guide lines при розробці під UI для вибраної ОС	так	–

5.6 Тестування інформаційної системи

Тестування додатку є одним із важливих етапів розробки, саме завдяки тестуванню можливо гарантувати передбачувану поведінку додатку, значно зменшуючи ризик виникнення проблем на етапі введення його в користування. Для перевірки додатку на готовність до випуску використовуються різні види тестування, в тому числі функціональне та модульне тестування, котрі будуть проведені для додатку, що описується у рамках цієї роботи.

Функціональне тестування – вид тестування, що акцентує увагу на успішному виконанні системою усіх визначених функціональних вимог. Для кожної функціональної вимоги складається один, або декілька, тест-кейсів з визначеними вхідними значеннями та критерієм успішності тест-кейсу. Цей вид тестування у даній роботі пуде проводитися у ручному режимі.

Модульне тестування – тестування, призначене для перевірки окремих сегментів коду на коректність їх роботи. Оскільки більшість систем збіки додатків, у тому числі Maven, що використовується у даній роботі, підтримують можливість автоматичного запуску модульних тестів, модельне тестування значно допомагає у попередженні помилок у коді ще до етапу перевірки коду іншими, більш витратними, видами тестування. Для модульного тестування буде використано фреймворк Junit, що має високий рівень інтеграції з Spring Boot.

Параметри комп’ютера, на якому буду проводитися тестування такі: операційна система – Windows 10 64x, процесор AMD Ryzen 3600 з частотою 4.2 ГГц і ОЗУ стандарту DDR4 об’ємом в 8+8 Гб у двоканальному режимі.

5.6.1 Розробка тест–кейсів для функціонального тестування системи

Для функціонально тестування розробимо тест-кейси, згідно з функціональними вимогами, які були описані в другому розділі.

Тест–кейс №1 F1.2.1 – реєстрація користувача – відкриття панелі реєстрації – введення коректного паролю, та некоректного номеру телефона – натискання кнопки підтвердження – користувач отримує повідомлення про невірний формат номеру.

Тест–кейс №2 F1.2.2 – реєстрація користувача – відкриття панелі реєстрації – введення коректного паролю, та некоректного номеру телефона та некоректного e-mail– натискання кнопки підтвердження – користувач отримує повідомлення про невірний формат e-mail.

Тест–кейс №3 F1.3 – реєстрація користувача – відкриття панелі реєстрації – введення паролю коротшого за 8 символів – натискання кнопки підтвердження – користувач отримує повідомлення про недостатню довжину пароля.

Тест–кейс №4 F1.4 – реєстрація користувача – відкриття панелі реєстрації – введення вже існуючого у системі e-mail– натискання кнопки підтвердження – користувач отримує повідомлення що такий e-mail вже зареєстровано в системі

Тест–кейс №5 F1.5 – реєстрація користувача – відкриття панелі реєстрації – вводимо усі дані окрім, одного з полів – натискання кнопки підтвердження – користувач отримує повідомлення що одне з полів не заповнене

Тест–кейс №6 F1.6 – реєстрація користувача – відкриття панелі реєстрації – введення коректного паролю, номеру телефона, та ін. даних – натискання кнопки підтвердження – користувач отримує повідомлення про успішну реєстрацію.

Тест–кейс №7 F2.1.1 – авторизація у аккаунт з роллю автор – відкриття панелі реєстрації працівників – введення коректних даних – натискання кнопки підтвердження – автор отримує повідомлення про успішну реєстрацію.

Тест–кейс №8 F2.1.2 – авторизація у аккаунт з роллю автор – відкриття панелі реєстрації авторів – введення коректних даних – натискання кнопки підтвердження – автор отримує повідомлення, що акаунт успішно зареєстровано.

Тест–кейс №9 F2.2.1 – авторизація у аккаунт з роллю читач – відкриття панелі реєстрації читачів – введення коректного паролю, та некоректного номеру телефона – натискання кнопки підтвердження – читач отримує повідомлення про невірний формат номеру.

Тест–кейс №10 F2.6 – авторизація у аккаунт з роллю читача – відкриття панелі реєстрації читачів – введення коректного паролю, номеру телефона, та ін. даних – натискання кнопки підтвердження – читач отримує повідомлення про успішну реєстрацію.

Тест–кейс №11 F3.1 – відкриття панелі автентифікації – введення даних неіснуючого користувача – натискання кнопки підтвердження – користувач отримує повідомлення про невірне введені дані

Тест–кейс №12 F3.2 – відкриття панелі автентифікації – введення імені існуючого користувача, але невірного паролю – натискання кнопки підтвердження – користувач отримує повідомлення про невірне введені дані

Тест–кейс №13 F3.3 – відкриття панелі автентифікації – введення імені існуючого користувача та вірного паролю – натискання кнопки підтвердження – користувач отримує повідомлення про успішну автентифікацію.

5.6.2 Протокол проведення функціонального тестування

На основі сформованого тест-плану було проведено функціональне тестування інформаційної системи. Його результати були занесені до протоколу в таблиці 5.4.

Таблиця 5.4 – Протокол проведення функціонального тестування системи

Номер тест-кейсу	Очікуваний результат	Фактичний результат	Статус
Тест–кейс №1	користувач отримує повідомлення про невірний формат номеру	відповідне повідомлення було отримане	пройдено
Тест–кейс №2	користувач отримує повідомлення про невірний формат e-mail	відповідне повідомлення було отримане	пройдено

Продовження таблиці 5.4

Номер тест-кейсу	Очікуваний результат	Фактичний результат	Статус
Тест–кейс №3	користувач отримує повідомлення про недостатню довжину пароля	відповідне повідомлення було отримане	пройдено
Тест–кейс №4	користувач отримує повідомлення що такий e-mail вже зареєстровано в системі	відповідне повідомлення було отримане	пройдено
Тест–кейс №5	користувач отримує повідомлення що одне з полів не заповнене	відповідне повідомлення було отримане	пройдено
Тест–кейс №6	користувач отримує повідомлення про успішну реєстрацію	відповідне повідомлення було отримане	пройдено
Тест–кейс №7	автор отримує повідомлення про успішну реєстрацію	відповідне повідомлення було отримане	пройдено
Тест–кейс №8	автор отримує повідомлення про невірний формат номеру	відповідне повідомлення було отримане	пройдено
Тест–кейс №9	автор отримує повідомлення про невірний формат e-mail	відповідне повідомлення було отримане	пройдено
Тест–кейс №10	автор отримує що такий e-mail вже зареєстровано в системі	відповідне повідомлення було отримане	пройдено
Тест–кейс №11	автор отримує що одне з полів не заповнене	відповідне повідомлення було отримане	пройдено
Тест–кейс №12	автор отримує повідомлення про успішну автентифікацію	відповідне повідомлення було отримане	пройдено
Тест–кейс №14	читач отримує повідомлення про невірне введені дані	відповідне повідомлення було отримане	пройдено
Тест–кейс №15	читач отримує повідомлення про невірне введення даних	відповідне повідомлення було отримане	пройдено
Тест–кейс №16	читач отримує повідомлення про успішну автентифікацію	відповідне повідомлення було отримане	пройдено
Всього успішних	16	16	
Всього провалених	0	0	

За результатами тестування можна відзначити, що всі тести були виконані і система пройшла функціональне тестування.

5.7 Інструкція користувача інформаційної системи

Розробимо документацію до системи і покажемо основні принципи використання.

Головна сторінка.

На рис. 5.5 головна сторінка, на яку потраплять всі користувачі

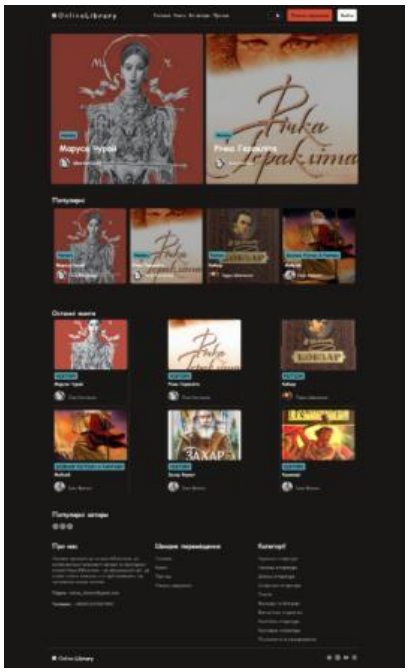


Рисунок 5.5 – Головна сторінка сайту

На цій сторінці можна побачити меню – згори, це меню доступне на усіх сторінках, та дозволяє користувачу переходити між ними. У середині сторінки знаходиться обкладинки книг та їх назви і категорії.

Реєстрація та автентифікація.

Для всіх користувачів сайтом існує єдина сторінка входу та реєстрації, де вони можуть обрати в якості кого вони збираються користуватися сайтом обравши роль «Читач» або «Автор» (рис. 5.6).

На цій сторінці (рис. 5.7) незареєстрований користувач може зареєструватися – обравши роль, рівень освіти, фото, та особисті дані.

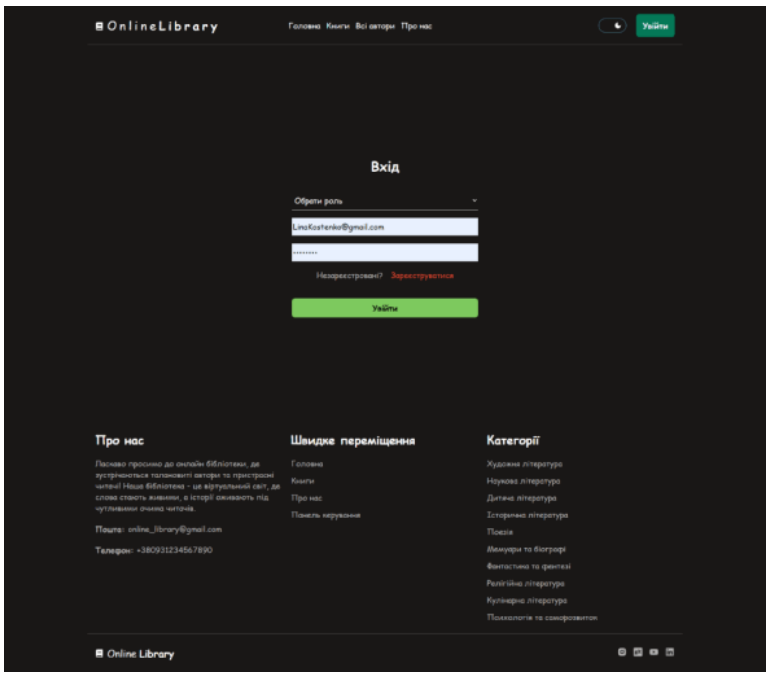


Рисунок 5.6 – Авторизація користувачів

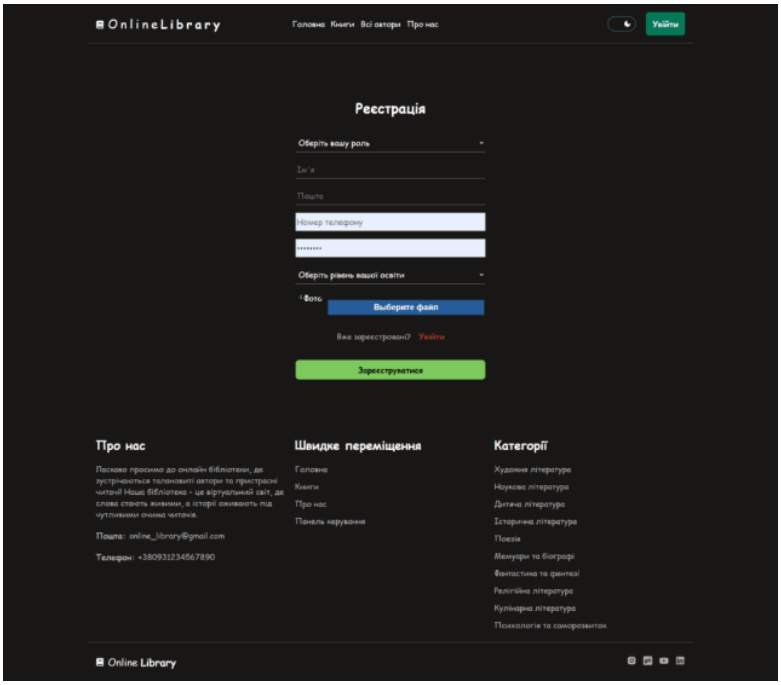


Рисунок 5.7 – Реєстрація користувачів

Читання книги.

Для того щоб обрати книгу для читання перейдемо на сторінку «Книги». Для цього натиснемо на пункт меню «Книги» (рис. 5.8). На цій сторінці з книгами бачимо самі книги їх фото, автора, назву та категорію.

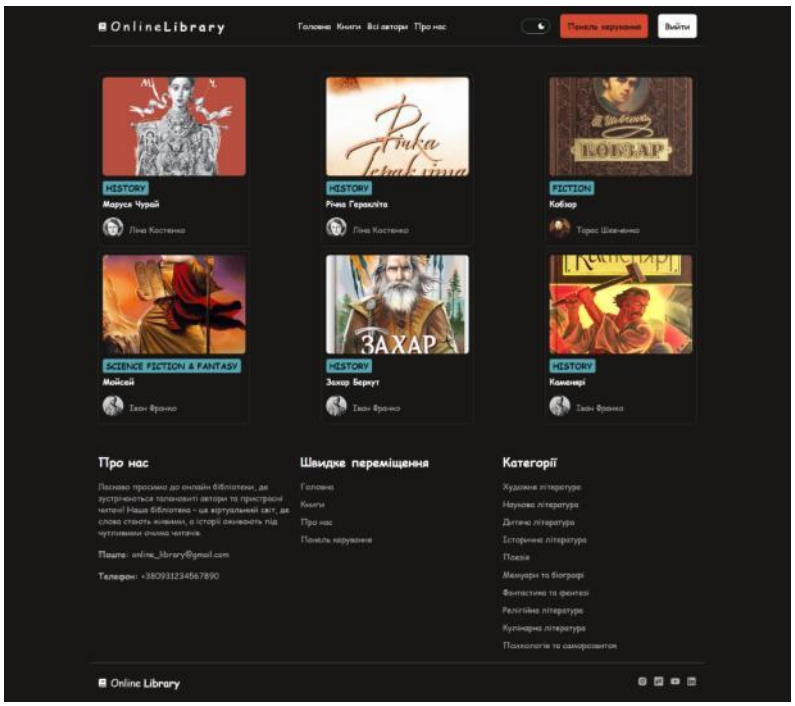


Рисунок 5.8 – Сторінка «Книги»

Обираємо книгу для читання «Кобзар» Тараса Шевченка (рис. 5.9).

Отже можемо побачити, гарно оформлений текст, та картинки які є доповненням цього тексту, це забезпечує зручність читання книги, так як вони виділяють певні сюжети.

Перегляд та редагування книг.

Розглянемо книгу для редагування «Річка Геракліта» – Ліна Костенко. Для цього зайдемо в панель керування, та знайдемо потрібну книгу (рис. 5.10).

Далі натиснемо на кнопку «Оновити» та побачимо сторінку для редагування публікації (рис. 5.11).

Додамо напис «Редагована» до назви книги (рис. 5.12) та натиснимо на кнопку «Оновити» в правому нижньому куту сторінки.

Перейдемо на головну сторінку, і можемо побачити оновлену публікацію серед інших публікацій (рис. 5.13).

Тепер спробуємо видалити дану публікацію з панелі керування автора (рис. 5.14) натиснувши кнопку «Видалити».

На рис. 5.15 головної сторінки сайт, можемо побачити що даної книги також не має.

Рисунок 5.9 – Книга «Кобзар» Тараса Шевченка

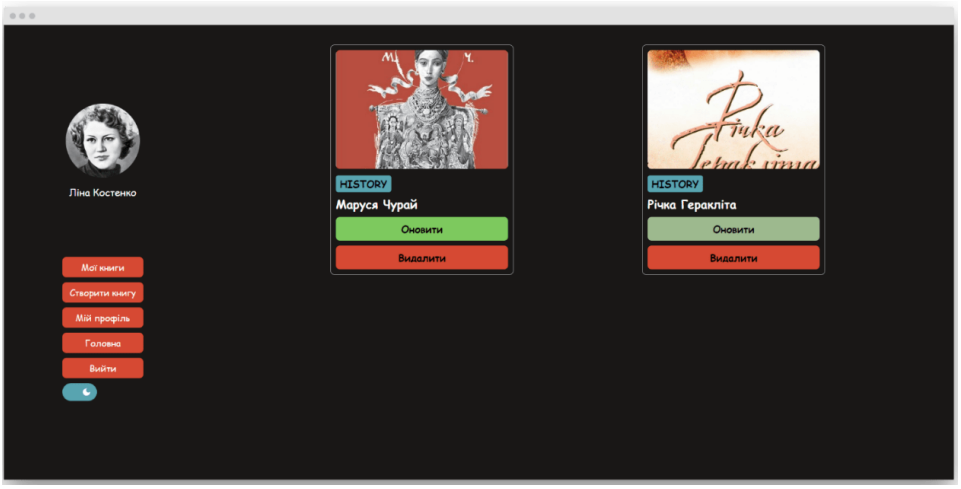


Рисунок 5.10 – Панель керування

Рисунок 5.11 – Сторінка оновлення публікації

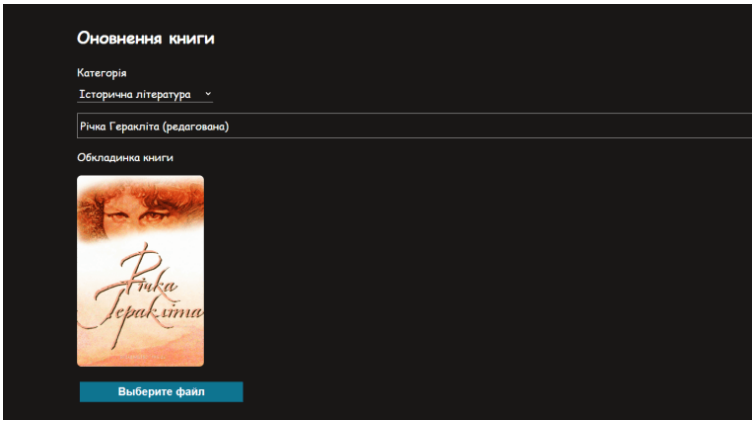


Рисунок 5.12 – Додавання напису до назви публікації

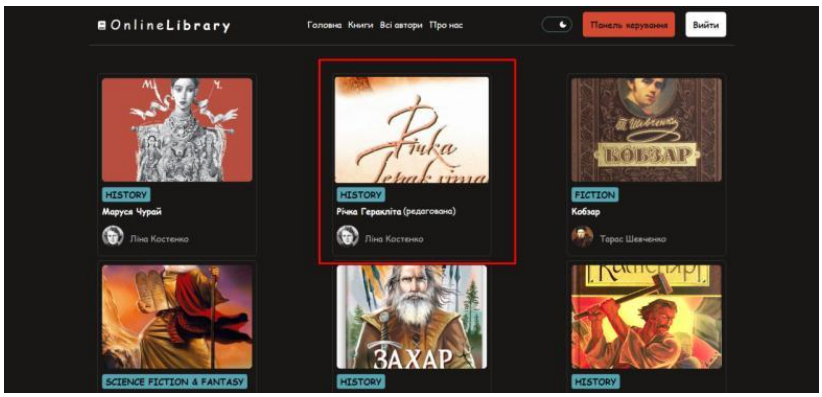


Рисунок 5.13 – Результат оновленої публікації

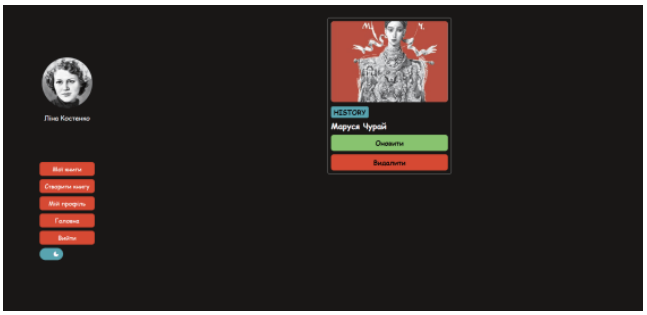


Рисунок 5.14 – Результат видалення публікації
на сторінці «Панель керування»

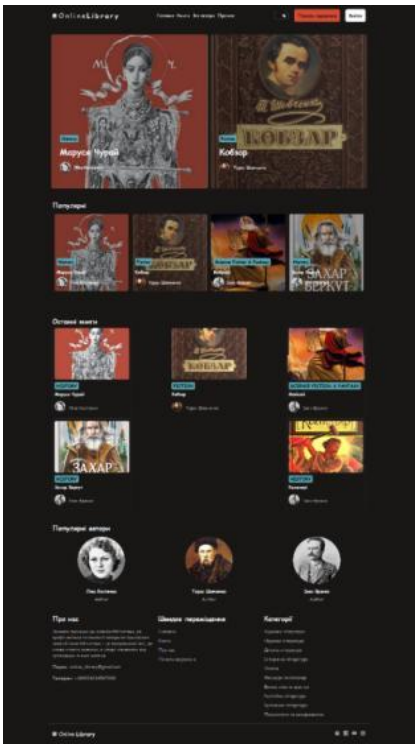


Рисунок 5.15 – Результат видалення публікації

6 ЕКОНОМІЧНА ЧАСТИНА

6.1 Характеристика продукції та ринок збуту

Актуальність онлайн-бібліотек полягає в необхідності забезпечення широкого та зручного доступу до літературних ресурсів через інтернет. Це сприяє розвитку освіти, науки та саморозвитку, особливо в умовах зростаючої цифровізації та дистанційного навчання. Онлайн-бібліотеки допомагають користувачам знаходити потрібну інформацію швидко та ефективно.

Метою проєкту є створення зручної та функціональної платформи для доступу до різноманітних літературних ресурсів в режимі онлайн. Це сприятиме полегшенню процесу пошуку та використання інформації для користувачів, підтримуючи їх освітні та дослідницькі потреби.

Даний додаток підходить для використання у повсякденному житті.

Економічна ефективність проєкту розраховується перед його розробкою, що дозволяє спрогнозувати потенційні вигоди та доцільність впровадження веб-додатку. На початку розраховується собівартість розробки, після чого визначається ціна.

Розглянемо переваги проєктованого додатка.

Додаток має максимально зрозумілий інтерфейс, який може бути легко освоєний користувачами без спеціальної підготовки, що дозволяє швидко залучити нову аудиторію.

Даний додаток має мінімальні системні вимоги, що забезпечує його стабільну роботу на більшості сучасних пристроїв і дозволяє охопити широку користувацьку базу без необхідності в оновленні обладнання.

Впроваджуваний додаток передбачає можливість інтеграції з популярними соціальними мережами, що підвищує його функціональність і привабливість для користувачів, а також сприяє безкоштовному просуванню через рекомендації та обмін інформацією.

6.2 Розрахунок витрат

Розглянемо порядок розробки веб-додатку.

На початковому етапі визначаються основні вимоги, що пред’являються до додатку, описуються основні цілі і розробляються специфікації, тобто виявляються основні властивості і показники, що їх характеризують.

Зовнішнє проєктування включає розробку архітектури і структури додатку, визначення алгоритму побудови, ідентифікацію підсистем і окремих складових їх модулів, а також створення зовнішнього інтерфейсу користувача.

Далі йде етап проєктування і кодування компонентів, який передбачає проєктування і написання коду обраними мовами програмування для окремих модулів додатку.

Після основний етап, який вважається найбільш трудомістким, оскільки включає тестування (перевірка функціональності, зручності користування та ефективності ресурсу) та оптимізацію (покращення швидкості завантаження, доступності контенту та адаптивності дизайну) окремих програмних модулів, а потім комплексне налагодження всього -додатку в цілому.

На заключному етапі проводиться остаточна корекція веб-додатку і тестування окремих блоків розробки.

Здійснимо розрахунок собівартості й ціни розробки веб-додатку.

У собівартість розробки додатку входять наступні статті витрат:

- основна заробітна плата;
- додаткова заробітна плата;
- єдиний соціальний внесок;
- інші витрати [2].

Відштовхуючись від організаційного плану компанії, можна побачити, що у створенні продукту бере участь 3 фахівці: UX/UI дизайнер, веб-програміст та фронтенд-інженер. Заробітна плата фахівців складає:

- дизайнер – 180 грн/год,

- веб-програміст – 220 грн/год,
- інженер – 95 грн/год.

При цьому тривалість робочого дня кожного з них становить 8 годин. Сайт розробляється 9 днів. Розрахунок основної заробітної плати наведено в таблиці 6.1.

Таблиця 6.1 – Розрахунок витрат на заробітну плату

Етап	Вид робіт	Виконавець		Годинна ставка, грн	Тривалість виконання, дні	Заробітна плата, грн
		кількість, ос.	посада			
1. Початковий	Формулювання вимог до веб-сайту	1	інженер	95,00	1	760,00
2. Графічна частина	Розробка графічного матеріалу	1	дизайнер	180,00	2	2880,00
3. Розробка й кодування компонентів	Розробка кожного компонента й верстка	1	програміст	220,00	2	3520,00
4. Основний етап	Тестування компонентів	1	програміст	220,00	1	1760,00
	Комплексне тестування сайту	1	програміст	220,00	1	1760,00
	Оформлення програмної документації	1	програміст	220,00	1	1760,00
5. Заключний етап	Корекція програмної документації	1	інженер	95,00	1	760,00
Разом					9	13200,00
Додаткова заробітна плата (20 %)						2640,00
Усього						15840,00

Додаткова заробітна плата – це винагорода за працю понад установлені норми, за трудові успіхи та винахідливість і за особливі умови праці. Вона включає доплати, надбавки, гарантійні і компенсаційні виплати, передбачені чинним законодавством; премії, пов'язані з виконанням виробничих завдань і

функцій. Отже, додаткова заробітна плата для фахівців цієї компанії становитиме 20 % від основної, та розраховуватиметься наступним чином:

$$13200{,}00 \times 0{,}2 = 2640{,}00 \text{ грн.}$$

Ставка єдиного соціального внеску становить 22 % від величини основної і додаткової заробітної плати:

$$(13200{,}00 + 2640{,}00) \times 0{,}22 = 3484{,}80 \text{ грн.}$$

До інших витрат віднесено оплату за електроенергію та обслуговування комп'ютерної техніки. Вони розраховуються виходячи зі споживаної потужності пристроїв і тарифу на електроенергію та інтернет. У даному випадку передбачається використання 3-х комп'ютерів потужністю 1,2 кВт/год. Вартість однієї кВт/год електроенергії прийнято у розмірі 2,64 грн. Час використання електроенергії в процесі розробки продукту:

$$9 \times 8 = 72 \text{ год.}$$

Отже, плата за електроенергію складе:

$$1{,}2 \times 2{,}64 \times 72 \times 3 = 684{,}29 \text{ грн.}$$

Витрати на обслуговування техніки визначаються виходячи з її вартості та часу експлуатації, після закінчення якого, вона підлягає заміні (зазвичай цей час не перевищує 3-х років). Отже, враховуючи, що вартість кожного комп'ютера дорівнює 27000,00 грн, а протягом року техніка використовується 254 робочих дні, отримаємо наступну суму витрат на обслуговування за час виконання проєкту:

$$(81000{,}00 / (3 \times 8 \times 254)) \times 72 = 956{,}69 \text{ грн.}$$

Проєкт впроваджується для однієї компанії, тому собівартість розробки становить:

$$(15840{,}00 + 3484{,}8 + 684{,}29 + 956{,}69) / 1 = 20965{,}78 \text{ грн.}$$

Розрахуємо суму прибутку від реалізації розробки (виходячи з рівня рентабельності 30 %):

$$20965{,}78 \times 0{,}3 = 6289{,}73 \text{ грн.}$$

Розрахуємо ціну розробки сайту без податку на додану вартість (ПДВ):

$$20965{,}78 + 6289{,}73 = 27255{,}52 \text{ грн.}$$

Розрахуємо суму ПДВ, що дорівнює 20 % від ціни без ПДВ:

$$27255{,}52 \times 0{,}2 = 5451{,}10 \text{ грн.}$$

З урахуванням проведених розрахунків ціна розробки сайту з ПДВ:

$$27255{,}52 + 5451{,}10 = 32706{,}62 \text{ грн.}$$

Результати розрахунків наведено у таблиці 6.2.

Таким чином, повна вартість розробки сайту складе 32706,62 грн. Термін виконання усіх етапів розробки становить 9 днів для команди, до якої входять веб-програміст, дизайнер та фронтенд-інженер. Очікувана сума прибутку складе 6289,73 грн, що свідчить про доцільність впровадження запропонованого сайту на підприємстві.

Таблиця 6.2 – Розрахунок витрат на розробку та ціни веб-додатку

№	Стаття витрат	Сума, грн
1	Основна заробітна плата	13200,00
2	Додаткова заробітна плата	2640,00
3	Єдиний соціальний внесок	3484,8
4	Витрати на обслуговування техніки	956,69
5	Витрати на електроенергію	684,29
6	Собівартість розробки сайту	20965,78
7	Прибуток	6289,73
8	Ціна без ПДВ	27255,52
9	Податок на додану вартість (ПДВ)	5451,10
10	Ціна з урахуванням ПДВ	32706,62

ВИСНОВКИ

У даній кваліфікаційній роботі був описаний процес створення веб-додатку для сучасної онлайн-бібліотеки, спрямований на діджиталізацію та удосконалення доступу до літературних творів. Система передбачає можливість реєстрації для як авторів, так і для читачів. Автори отримують можливість публікувати свої твори, які автоматично розбиваються на категорії після завантаження. Кожна категорія супроводжується коротким змістом, складеним інтелектуальною системою, що спрощує огляд та навігацію для читачів. Це дозволяє забезпечити зручне читання як повного тексту, так і окремих частин книги, зекономивши час користувача та забезпечивши йому зручний доступ до необхідної інформації. Застосунок розроблено з використанням JavaScript, CSS та HTML. Були використані фреймворки React.js, Node.js та бази даних MongoDB та Cloudinary. Таким чином, мета, поставлена в даній кваліфікаційній роботі досягнута в повному обсязі.

Для досягнення мети було виконано ряд завдань, а саме аналіз предметної області та пошук функціоналу, що доцільно запозичити, вибір інструментів для розробки програмного забезпечення вибір патернів проектування, проектування системи, її програмна реалізації та тестування.

В першому розділі при аналізі предметної області було розглянуто три аналоги, що є потенційними конкурентами розроблюваної системи. Було запозичені основні переваги цих систем та прийнято рішення розширити стандартний для подібних систем. Завдяки чому розроблювана система надає більше можливостей, ніж аналоги.

В другому розділі були визначені та описані функціональні та нефункціональні вимоги до системи та були представлені скриншоти користувацького інтерфейсу. Також була спроектована архітектура системи, описана структура її компонентів, шарів, інтерфейсів та структури баз даних

з впровадженням частини шаблонів проектування, які були обрані для використання при розробці.

В третьому розділі були описані основні алгоритми бізнес-логіки системи. Для контролю вихідного коду була використана система Git. Були розраховані метрики програмного коду розробленої системи. У рамка розділу була розроблена схема тестування системи за допомогою функціональних та модульних тестів, котрі система пройшла успішно. Наприкінці розділу була розроблена інструкція користувача системи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Методичні вказівки з виконання кваліфікаційної роботи для студентів денної та заочної форм навчання першого (бакалаврського) рівня вищої освіти спеціальності 186 Видавництво та поліграфія / В.П. Ткаченко, А.В. Бізюк, О.В. Вовк та ін. Харків: ХНУРЕ, 2021. 68 с.

2. Полозова Т.В. Економіка і бізнес: комплекс навчально-методичного забезпечення підготовки бакалаврів спеціальності 186 Видавництво та поліграфія. Харків, ХНУРЕ. 2017. 382 с.

3. Мікросервісна архітектура // Medium. URL: https://medium.com/@IvanZmerzlyi/microservices-architecture-461687045b3d (дата звернення: 21.05.2024)

4. Що таке Node JS простими словами. URL: https://dan-it.com.ua/uk/blog/chto-jeto-takoe-node-js-prostymi-slovami/ (дата звернення: 21.05.2024).

5. Що таке React JS і для чого він потрібен? URL: https://dan-it.com.ua/uk/blog/chto-takoe-react-js-i-dlja-chego-on-nuzhen/ (дата звернення: 19.05.2024).

6. ExpressJS. URL: https://expressjs.com/uk/ (дата звернення: 22.05.2024).

7. Axios. URL: https://www.npmjs.com/package/axios (дата звернення: 14.05.2024).

8. Що таке MongoDB? вступ, Architектура, функції та приклад. URL: https://www.guru99.com/uk/what-is-mongodb.html (дата звернення: 15.05.2024).

9. Cloudinary. URL: https://hellip.com/ua/product/cloudinary.html (дата звернення: 15.05.2024).

10. Vite. URL: https://vitejs.dev/ (дата звернення: 18.05.2024).

11. React Icons. URL: https://react-icons.github.io/react-icons/ (дата звернення: 22.05.2024).