

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Онлайн платформа для письменників та читачів»

Виконав студент групи ПЗ-20-3

ШЕВЧЕНКО Дарина Володимирівна

Керівник роботи:

КРАВЧЕНКО Світлана Миколаївна

Рецензент:

САВІЦЬКИЙ Роман Святославович

Житомир – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»
Зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«26» лютого 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ШЕВЧЕНКО Дарина Володимирівна**
Керівник роботи: **КРАВЧЕНКО Світлана Миколаївна**
Тема роботи: **«Онлайн платформа для письменників та читачів»**
затверджена Наказом закладу вищої освіти від «23» лютого 2024 р. №74/с
Вихідні дані для роботи: предмет дослідження: **Методи та засоби розробки онлайн платформи для письменників та читачів**, об'єкт дослідження:
Використання фреймворку Laravel для розробки веб-додатку у поєднанні з мовою програмування PHP та бібліотекою JQuery.

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Кравченко С. М.	27.02.2024р.	27.02.2024р.
2	Кравченко С. М.	27.02.2024р.	27.02.2024р.
3	Кравченко С. М.	27.02.2024р.	27.02.2024р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з додатку «Онлайн платформа для письменників та читачів» та пояснювальної записки. Пояснювальна записка містить 86 сторінок, 74 ілюстрацій та 11 таблиць.

Метою роботи є розробка онлайн платформи для письменників та читачів, яка має сторінки авторизації, реєстрації, 2-FA; головної, сторінки каталогу та сторінки окремої книги; форми зворотнього зв'язку; системи оплати за допомогою Stripe з зовнішніми сторінками для оплати; адмін панелі з можливістю маніпулювати всіма сутностями на сайті; системи моніторингу помилок Sentry, системи для роботи з ElasticSearch – Kibana; RabbitMQ management.

В роботі визначено основні завдання на розробку онлайн платформи, проаналізовано аналоги розробленої платформи, обґрунтовано вибір популярного фреймворку на PHP – Laravel. Наведено сценарії роботи програмного комплексу, загальна структура програмного комплексу, опис бази даних онлайн платформи (СУБД PostgreSQL), опис алгоритмів взаємодії окремих модулів системи, опис об'єктної структури системи (діаграма класів), алгоритми роботи основних модулів, реалізованих з використанням мови програмування PHP. Також наводиться результат тестування платформи, визначено порядок встановлення та налаштування платформи під керівництвом на веб-сервері nginx.

КЛЮЧОВІ СЛОВА: ОНЛАЙН ПЛАТФОРМА, ПИСЬМЕННИКИ, ЧИТАЧІ, PHP, LARAVEL.

					ІПЗ.КР.Б – 121 – 24 - ПЗ				
Змн.	Арк.	№ докум.	Підпис	Дата	Онлайн платформа для письменників та читачів		Літ.	Арк.	Аркушів
Розроб.	Д. Шевченко								
Перевір.	С. Кравченко							4	86
Керівник							Житомирська політехніка, група ІПЗ-20-3		
Н. контр.	О. Тичина								
Зав. каф.	Т. Вакалюк				Пояснювальна записка				

ABSTRACT

The final qualification work of the bachelor consists of the application "Online platform for writers and readers" and an explanatory note. Explanatory note contains 86 pages, 74 illustrations and 11 tables.

The purpose of the work is to develop an online platform for writers and readers, which has pages of authorization, registration, 2-FA; Home, catalog pages and individual book pages feedback forms; Stripe payment systems with external payment pages; admin panel with the ability to manipulate all entities on the site; Sentry error monitoring systems, systems for working with Elasticsearch – Kibana; RabbitMQ management.

The work defines the main tasks for the development of an online platform, analyzes the analogues of the developed platform, justifies the choice of a popular framework in PHP – Laravel. The scenarios of the software complex operation, the general structure of the software complex, description of the online platform database (PostgreSQL DBMS), description of algorithms of interaction of individual modules of the system, description of the object structure of the system (class diagram), algorithms of operation of the main modules implemented using the PHP programming language are presented. The result of testing the platform is also given, the procedure for installing and configuring the platform under the guidance on the nginx web server.

KEYWORDS: ONLINE PLATFORM, WRITER, READER, PHP, LARAVEL.

		<i>Д. Шевченко</i>			ІІЗ.КР.Б – 121 – 24 – ІІЗ	<i>Арк.</i>
		<i>С. Кравченко</i>				
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		<i>4</i>

ЗМІСТ

РЕФЕРАТ.....	3
ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ДОДАТКУ ОНЛАЙН ПЛАТФОРМА ДЛЯ ПИСЬМЕННИКІВ ТА ЧИТАЧІВ.....	9
1.1 Постановка задачі	9
1.2 Аналіз аналогів програмного продукту	11
1.3 Визначення архітектури онлайн платформи	18
1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення.....	20
Висновки до першого розділу	23
РОЗДІЛ 2. ПРОЄКТУВАННЯ ОНЛАЙН ПЛАТФОРМИ ДЛЯ ПИСЬМЕННИКІВ ТА ЧИТАЧІВ.....	24
2.1 Визначення варіантів використання та структури системи	24
2.2 Розробка бази даних	34
2.3 Проектування та реалізація алгоритмів роботи додатку	37
2.4 Реалізація функціоналу додатку	42
2.4.1. Реалізація основної архітектури додатку.....	42
2.4.2. Реалізація системи реєстрації та авторизації.....	44
2.4.3. Реалізація системи підтвердження номеру телефону.....	46
2.4.4. Реалізація системи купівлі преміум аккаунту	46
2.4.5. Реалізація системи взаємодії Elasticsearch та PostgreSQL	48
Висновки до другого розділу	50
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ ОНЛАЙН ПЛАТФОРМИ ДЛЯ ПИСЬМЕННИКІВ ТА ЧИТАЧІВ	51
3.1 Порядок встановлення та налаштування параметрів додатку.....	51
3.2 Структура інтерфейсу веб-додатку та порядок роботи.....	53
3.3 Тестування роботи додатку	76
Висновки до третього розділу	77
ВИСНОВКИ.....	78
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	79
ДОДАТКИ	82

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

CRUD – create, read, update, delete (з англ. Створити, прочитати, оновити, видалити)

MVC – Model–View–Controller (з англ. Модель–Представлення–Контролер)

БД – база даних

API – прикладний програмний інтерфейс

СУБД – система управління базами даних

ORM – програмний інтерфейс для зв'язку з базою даних

ЧЕРГА – брокер повідомлень налаштований через RabbitMQ

ПОШУКОВИЙ ДВИГУН – частина документно-орієнтованої бази даних Elasticsearch, яка відповідає за пошукову видачу

ЛОГ – програмний продукт Sentry, який відповідає за відслідковування помилок на порталі

РЕАЛ ТАЙМ – алгоритм, зміни якого відслідковуються іншим алгоритмом в реальному часі (в даній роботі реалізовано за допомогою Laravel Echo)

ОРКЕСТРОВКА – управління контейнерами, використовуючи docker compose

БЕК-ОФІС – користувачі, які відповідають за невидну для звичайних користувачів функціональність порталу (відділ підтримки, модератори)

ДАШБОРД – інтерактивна аналітична панель

JIRA – система управління проектами

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				6
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. У сучасному світі досить популярною є тема онлайн платформ, оскільки можна охопити досить велику аудиторію, надати простір людям з різними точками зору. Користувачі інтернету, які хочуть поділитись інформацією звикли до того типу подачі, який нав'язаний їм блогами та соціальними мережами, обидва варіанти подачі інформації є досить зручними, але вони заносять письменника в жорсткі рамки:

- часто соціальні мережі обмежують кількість символів;
- для того, щоб почати вести блог його треба створити, що майже не можливо для некваліфікованого користувача;
- при використанні соціальних мереж та блогів, досить тяжко заманити читачів до написаних вами робіт;
- рідко трапляється зрозуміла система монетизації для письменника.

Наразі платформи для читання є досить новими на ринку електронної комерції, тому розробка, покращення та система інтеграції цих платформ є актуальними на даний час. Основною проблемою нинішніх, уже розроблених платформ для читання є те, що вони досить «сирі» для сучасного споживача. Як приклад, Wattpad є найбільш поширеною платформою для читання, але вона не надає можливості письменникам заробляти кошти на своїй праці, в такому вигляді до якого звик сучасний споживач (місячні/річні пакети, зрозуміле донесення інформації про те, як почати заробляти кошти, гнучка система монетизації).

Хоч і соціальні мережі та блоги дають можливість розкрити талант письменника в собі, але почати монетизувати свою роботу є досить складно.

Метою кваліфікаційної роботи є розробка та реалізація онлайн платформи, яка надає гнучку, безперебійну, просту для розуміння систему для публікації своїх робіт письменниками, зрозумілий інтерфейс для читачів, багатофункціональну адмін панель для ефективної роботи бек-офісу, зручного дашборду для моніторингу роботи девелоперського та продакшн серверів,

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				7
Змн.	Арк.	№ докум.	Підпис	Дата		

системи для відслідковування помилок, СУБД для нереляційних та реляційних БД, які будуть використовуватися в додатку та оркестраторів Docker контейнерів.

Завдання кваліфікаційної роботи включає такі етапи:

- Створення списку завдань у системі управління проектами;
- Розробка дизайну клієнтської частини сайту;
- Реалізацію ядра системи управління контентом;
- Реалізацію модулів для адміністратора;
- Реалізацію структури збереження даних;
- Створення демонстраційного та тестового контенту сайту;
- Розробка плану тестування.

Об’єктом дослідження є розробка, проектування та реалізація онлайн платформи для письменників та читачів.

Предметом дослідження є використання сучасних технологій для розробки та підтримки гнучких та безперебійних веб-орієнтованих додатків.

Методи дослідження: для розробки онлайн платформи було використано мову програмування PHP 8.1, фреймворк оснований на Enterprise патерні MVC Laravel, Docker, Docker Compose для оркестровки контейнерів на девелоперському та локальному середовищах, PostgreSQL як основна – реляційна БД, Redis як нереляційна БД для кешування, JQuery для реалізації frontend, Elastic Search як пошуковий двигун, RabbitMQ для реалізації системи черг в додатку, NodeJS в поєднанні з Laravel Echo для реал тайм функціональність (чат, сповіщення), Stripe – платіжна система.

Випускна робота складається зі вступу, 3-х розділів, висновків, списку використаних джерел із 30 найменувань, 3 додатків, 11 таблиць та 74 ілюстрацій. Повний обсяг роботи становить 86 сторінок, з них 81 сторінок основного тексту.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				8
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ДОДАТКУ ОНЛАЙН ПЛАТФОРМА ДЛЯ ПИСЬМЕННИКІВ ТА ЧИТАЧІВ

1.1 Постановка задачі

Основна мета створення такої платформи, надати багатьом письменникам, які не розкрили себе, або можливо не впевнені, що їхні твори сподобаються іншим, можливість спробувати написати свою роботу та надати можливість оцінити книгу іншим. Також ця платформа буде цікавою і для читачів, адже окрім читання книг, там можна купувати, коментувати, та додавати у власні бібліотеки та ділитись бібліотекою з іншими.

У кваліфікаційній роботі необхідно розробити онлайн платформу, засоби управління інформаційним контентом, модуль для пошуку книг/авторів, модуль для управління додатком та модуль оплати.

Для реалізації поставленого завдання, основними етапами є:

1. Створити список завдань у системі управління проектами.
2. Розробити дизайн клієнтської частини сайту:

2.1 Спроектувати загальну концепцію дизайну сторінки авторизації і реєстрації та сторінки аккаунту зареєстрованого користувача: профіль, сторінка пошуку письменників та книг, сторінка чату, сторінка написання книги, сторінка адміністрування платформи та сторінка зворотнього зв'язку.

2.2 За допомогою мови розмітки HTML та таблиці стилів CSS створити шаблони для сторінки авторизації/реєстрації та сторінок аккаунту.

3. Реалізувати ядро системи управління контентом:

3.1 Написати програмні модулі для:

- авторизації користувачів у платформі. Розподіл прав доступу користувачів повинен зазначатись за наступними ролями: адміністратор, гість, зареєстрований користувач, преміум користувач, відділ підтримки та модератор;

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				9
Змн.	Арк.	№ докум.	Підпис	Дата		

- профіль користувача. Модуль для виведення інформації про користувача, кількість його книг, список бібліотек, повідомлення;
- можливість створювати та редагувати книги. Переглядати сторінки та книги інших користувачів;
- модуль користувачів. Пошук користувачів, перегляд профілю користувача, написати повідомлення в особисті повідомлення;
- модуль чату. Створити діалог з підписником, написати повідомлення, видалити діалог, переглянути інформацію про співрозмовника;
- модуль для пошуку книг та користувачів. Пошук книг за різними категоріями (жанрами), пошук користувачів. Можливість видаляти з бібліотеки, переглянути створені публічні бібліотеки користувачів.
- модуль написання книги. Додавати, редагувати, видаляти книги з бібліотеки, додавати та редагувати інформацію про книгу.

3.2 Реалізувати CRUD операції для всіх таблиць бази даних платформи, для використання їх в адміністративній панелі.

4. Реалізувати модулі для адміністратора:

- 4.1 Видалення користувачів з системи.
- 4.2 Видалення книги.
- 4.3 Зміна ролі користувача.
- 4.4 Перегляд статистик.

5. Реалізувати структуру збереження даних наступного виду:

- 5.1 Дані про користувачів (ім'я, email, телефон, пароль, тип користувача (видавництво/користувач), адреса, аватар, статус);
- 5.2 Дані написання книги (назва, титульне зображення, короткий опис, головні герої, категорія, теги, цільова аудиторія, мова, авторські права, статус, видавництво, тип книги(платна/безкоштовна), вміст книги);
- 5.3 Замовлення (транзакція, автор, тип оплати, статус оплати)
- 5.4 Зворотній зв'язок (автор, текст, заголовок, email)

6. Створити демонстраційний та тестовий контент сайту.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				10
Змн.	Арк.	№ докум.	Підпис	Дата		

7. Розробити план тестування системи управління контентом. Передбачити наступні види тестування:

7.1 Тестування функціональне.

7.2 Кросбраузерне та адаптивне тестування.

Результатом реалізації поставленого завдання є реалізована онлайн платформа для письменників та читачів, що містить в собі серверну структуру збереження даних, багатокористувацький клієнтський додаток для реалізації функціональних можливостей та засоби контролю та керування доступом.

1.2 Аналіз аналогів програмного продукту

В цьому розділі буде представлено деякі з аналогів та визначено їх переваги та недоліки. В цілому є всього декілька відомих аналогів-додатків, де читачі зможуть читати та публікувати власні роботи. Отже, докладно розглянемо кожен із них.

Один із таких додатків, який є аналогом розробленого додатку, є «inkitt.com» [14]. Даний веб-сайт не обмежує доступу до книг, в ньому є розподіл книг за категорією, що дає можливість легко знайти ту чи іншу книгу. Також там є кнопка переходу до безкоштовних книг (Free Books), але платних книг не було знайдено, тому не зрозуміло навіщо розподіляти книги, якщо на сайті немає платних книг. Також дуже незручним є той факт, що переходу на каталог немає, ми маємо вибрати жанр, тег і тільки тоді ми перейдемо в окремий розділ (рис. 1.1). Для кожної книги є відгуки та оцінки (рис. 1.2).

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				11
Змн.	Арк.	№ докум.	Підпис	Дата		

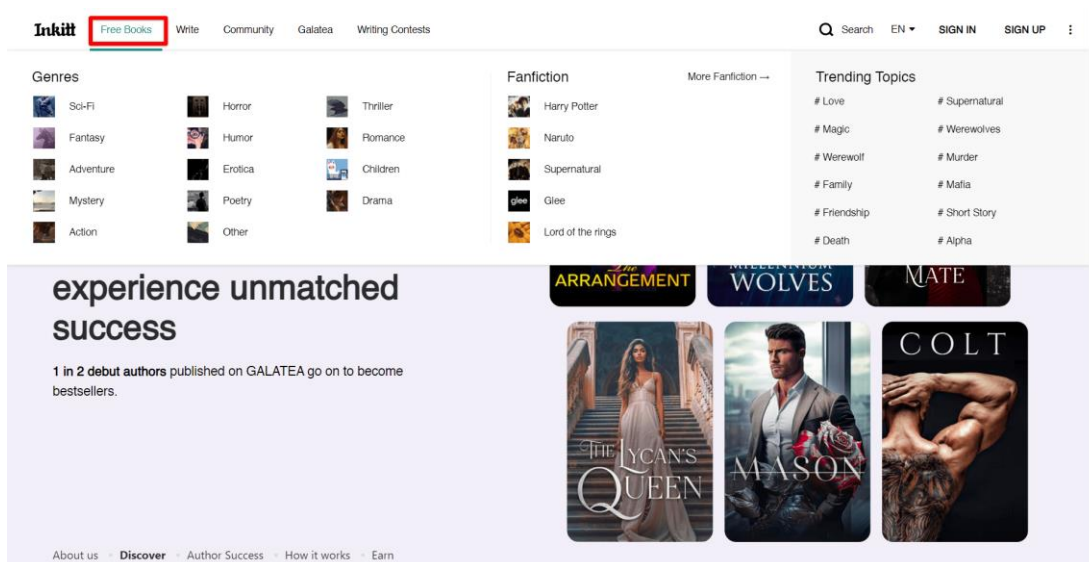


Рис. 1.1 Міні каталог у веб-додатку «inkitt.com»

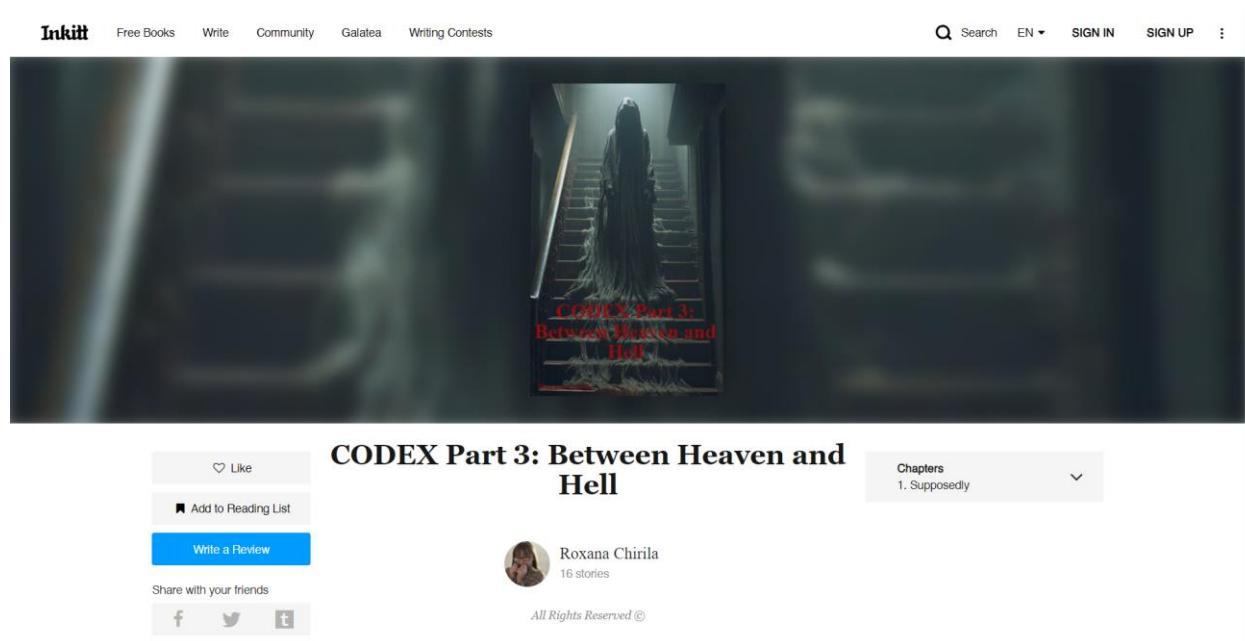


Рис. 1.2 Книга у веб-додатку «inkitt.com»

Також додаток має можливість не тільки написати історію на сайті, а й завантажити її на сайт вже написаний файл (рис. 1.3).

		Д. Шевченко			ПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				12
Змн.	Арк.	№ докум.	Підпис	Дата		

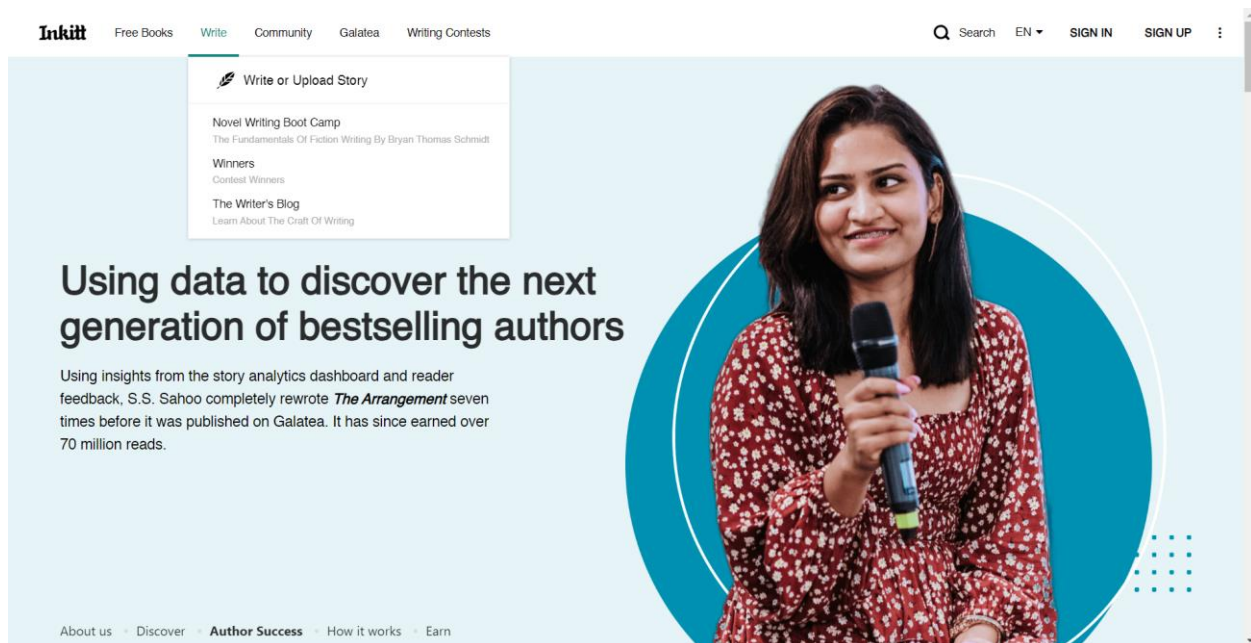


Рис. 1.3 Написання та завантаження книги у веб-додатку «inkitt.com»

Щодо інтерфейсу додатку, він є досить зручним та майже всюди зрозумілим. Можна увійти в особистий кабінет. Інтерфейс додатку представлений на англійській мові, хоч там і є можливість змінити мову, але мов дуже мала кількість, тому це не дуже зручно (рис. 1.4).

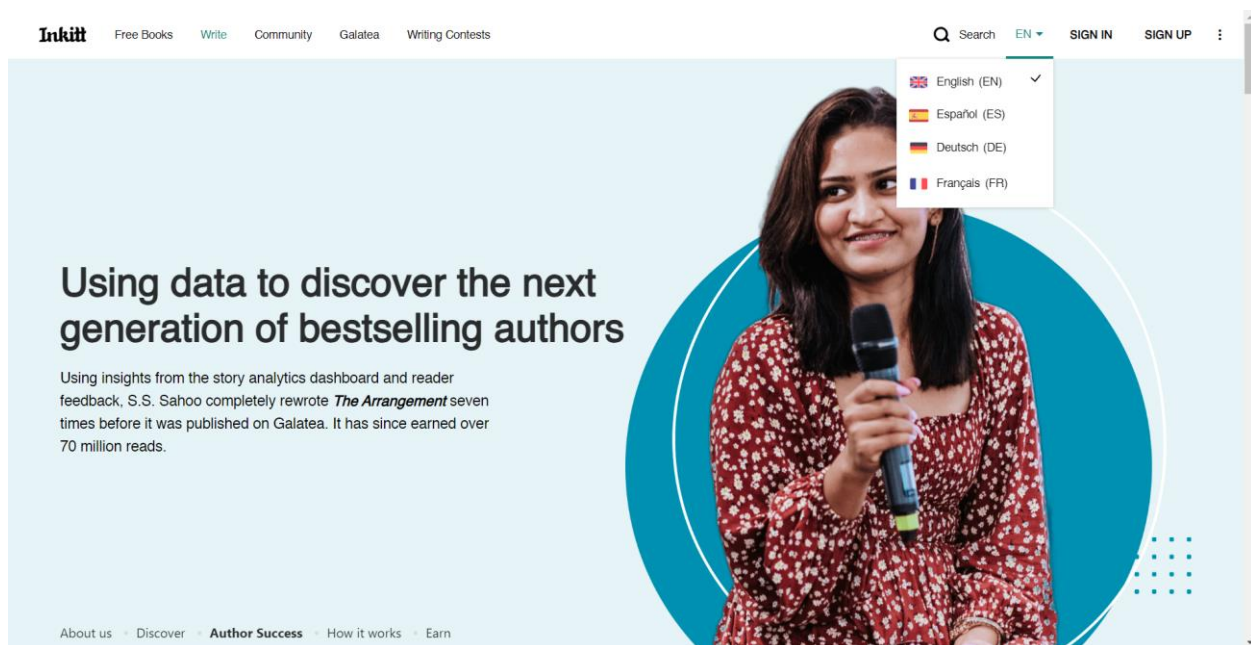


Рис. 1.4. Багатомовність у веб-додатку «inkitt.com»

Другий популярний аналог «wattpad.com» [15] (рис. 1.5). Додаток дає можливість написати та прочитати книги, але ми не можемо побачити хоч частину каталогу оскільки ми не зареєстровані, що є не дуже зручно.

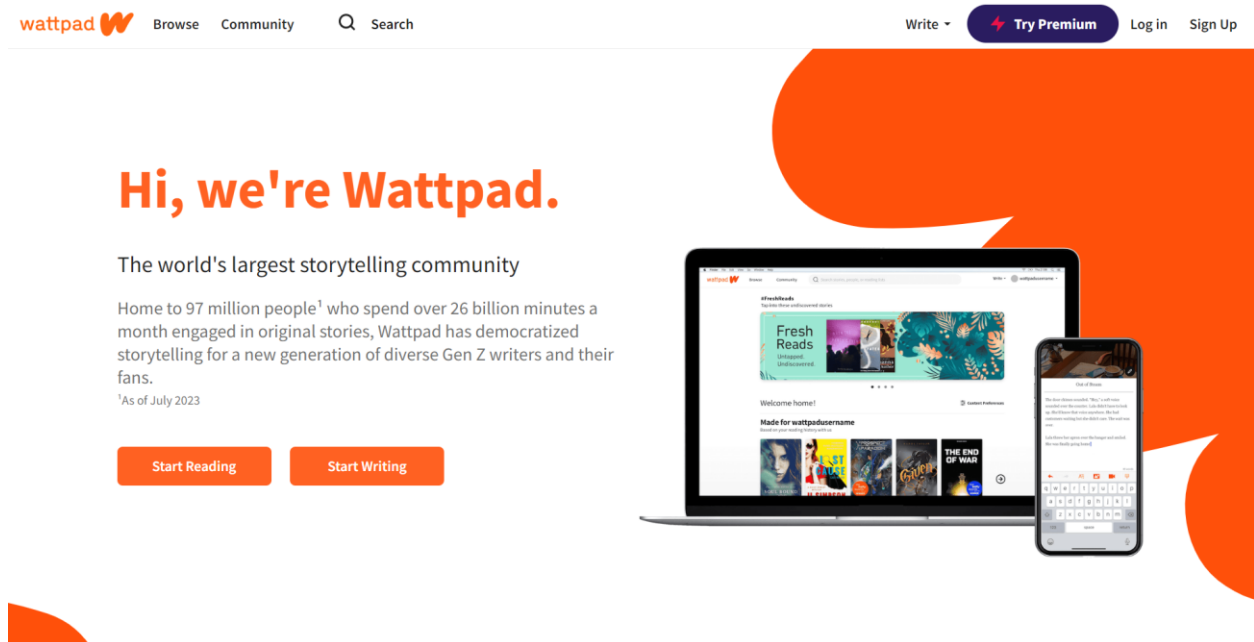


Рис. 1.5. Головна сторінка у веб-додатку «wattpad.com/»

Як можемо побачити, після входу перед нами з'явився невеликий відділ з книгами та історіями (рис. 1.6).

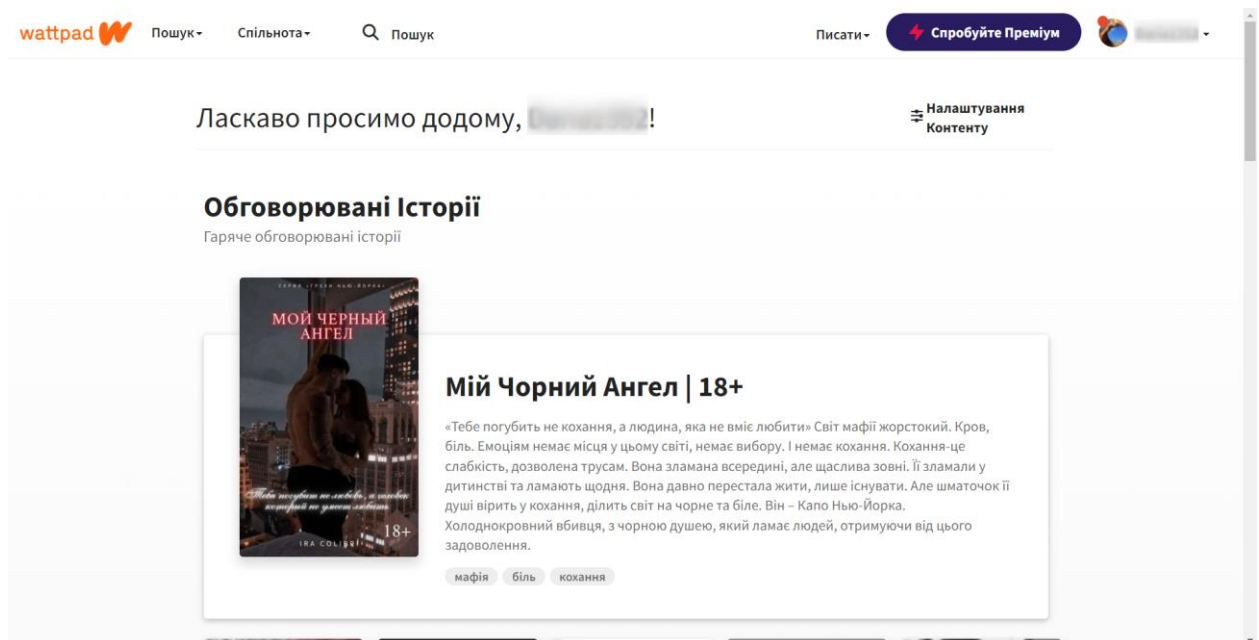


Рис. 1.6 Головна сторінка зареєстрованого користувача

На сайті також є можливість налаштувати контент, тобто заблокувати певні теги та показ дорослого контенту, що є дуже зручно. Є можливість пошуку по категоріям (рис. 1.7), але як на мене зроблено не дуже то і зручно, адже

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІПЗ	Арк.
		С. Кравченко				14
Змн.	Арк.	№ докум.	Підпис	Дата		

перейшовши до певного жанру, ми можемо тільки додати теги, щоб розширити пошук. Недоліком є те, що недостатня кількість критеріїв у фільтрах.

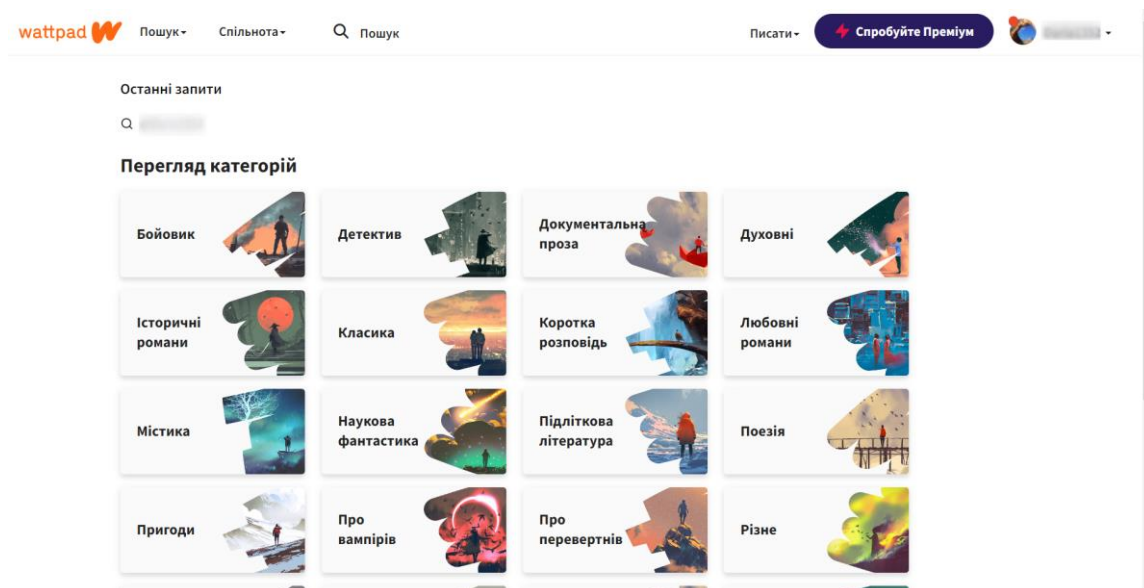


Рис. 1.7 Пошук по категоріям

Також є можливість написати книгу, але в цьому веб-додатку, так як в минулому аналогу, імпортувати книгу неможливо. Приклад форми написання книги додано нижче (рис. 1.8).

Рис. 1.8 Написання книги у веб-додатку «wattpad.com»

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				15
Змн.	Арк.	№ докум.	Підпис	Дата		

Є можливість увійти в особистий кабінет. Інтерфейс досить зрозумілий, привабливий дизайн, зручність навігації для користувачів. Також присутня монетизація, тобто автори можуть отримувати винагороду за свої твори. Також проект має в наявності мобільні додатки для Android та iOS платформ.

Останнім аналогом є веб-додаток «fanfiction.net» [16], через який можна завантажити та прочитати твори (рис. 1.9).

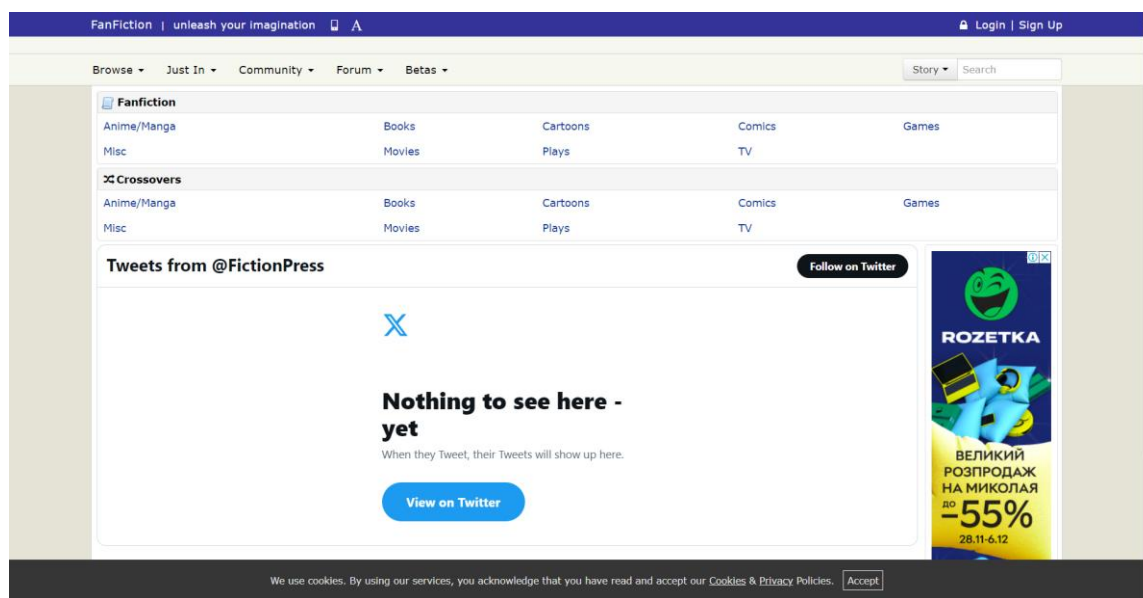


Рис. 1.9 Головна сторінка веб-додатку «fanfiction.net»

Як ми можемо побачити, додаток є досить розширеним, тобто там є не тільки книги, а й мультфільми, комікси, аніме та ін. Враховуючи, що в нас тема стосується книг, то на прикладі будемо розглядати саме їх. Отже, перейдемо до каталогу книг (рис. 1.10).

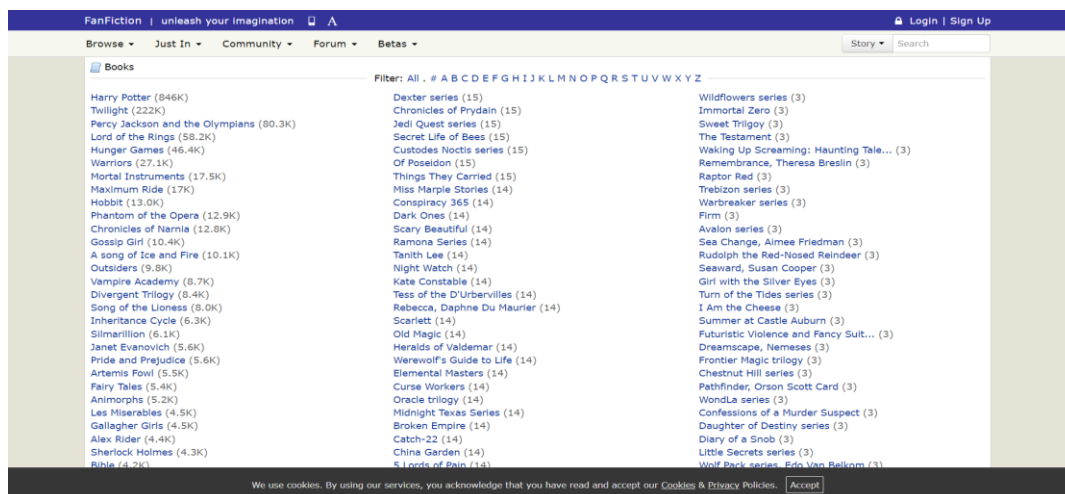


Рис. 1.10 Каталог книг веб-додатку «fanfiction.net»

		Д. Шевченко			Арк.
		С. Кравченко			16
Змн.	Арк.	№ докум.	Підпис	Дата	

Додаток надає можливість пошуку по книзі, письменниках, форумах та спільнотах (рис. 1.11). Інтерфейс досить не зручний.



Рис. 1.11 Пошук за критеріями

Отже, узагальнюючи побачене можна підсумувати – перед нами є список з усіх книг з незадовільним інтерфейсом та безліччю посилань без ніяких стилів, через що важко орієнтуватись на сайті та знайти потрібну нам інформацію. Також тут наявна фільтрація, яка також має малий список критеріїв для підбору. Тобто є можливість відфільтрувати книги лише по літерах, що також є не зручним.

Для порівняння вище описаних веб-додатків було створено порівняльну таблицю (табл. 1.1).

Таблиця 1.1

Порівняння веб-додатків аналогів

Характеристика	Inkitt	Wattpad	FanFiction	LiteraryLoom
Розподіл книг за категорією	Так	Так, пошук по категоріям	Ні	Так, пошук по певним критеріям
Зручний та зрозумілий інтерфейс	Так	Так	Ні	Так

Оригінальний контент	Так	Так	Ні	Так
Доступ до каталогу без реєстрації	Так	Ні	Ні	Так
Адаптивний дизайн	Так	Так	Так	Так
Присутність мобільного додатку	Ні	Так	Так	Наразі лише веб-сайт
Коментарі та оцінка читачів	Так	Так	Ні	Так

1.3 Визначення архітектури онлайн платформи

Для реалізації веб-орієнтованої онлайн платформи для письменників та читачів мною було обрано архітектурний патерн MVC, який вбудовано в обраний фреймворк Laravel [4, 17]. Розглянуті MVVM і MVP-патерни, які не забезпечують такої гнучкості щодо відображення даних. Для MVVM зв'язування View з View-Model здійснюється автоматично, а для MVP – необхідно програмувати даний процес. MVC має більше можливостей по управлінню представленням. В класичному варіанті, MVC складається з трьох частин, які і дали йому назву (рис. 1.12). Розглянемо їх особливості більш детально:

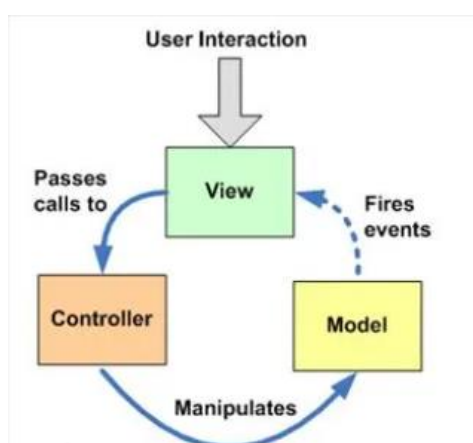


Рис. 1.12 Узагальнена архітектура MVC

2. Модель. Модель - це частина з функціональною бізнес-логікою програми. Модель повинна бути повністю незалежна від інших частин продукту. Даний шар нічого не повинен знати про елементи представлення, і яким чином він буде відображатися. Результат, що дозволяє змінювати представлення даних, отримано без використання моделі.

Модель володіє наступними ознаками:

- реалізує бізнес-логіку програми;
- володіє доступом лише до своїх компонентів;
- в залежності від технології реалізації модель є просто шаром даних (DAO, база даних, XML-файл) або менеджером бази даних, набором об'єктів логіки додатку.

2. Представлення. До компетенцій представлення входить відображення даних отриманих від моделі. Однак, представлення не може безпосередньо впливати на модель. Можна говорити, що представлення отримує доступ «тільки читання» до даних.

Представлення має наступні ознаки:

- у представленні реалізується відображення даних, які отримані від моделі у будь-якій спосіб;
- у деяких випадках, представлення може мати код, який реалізує частину бізнес-логіки.

Приклади представлення: HTML-сторінка, WPF форма, Windows Form.

3. Контролер. Основна ідея патерну MVC в тому, що контролер та представлення залежать від моделі, але модель не залежить від цих двох компонентів.

Ознаки контролера:

- контролер визначає яке представлення має бути відображено в даний момент;
- події представлення можуть вплинути тільки на контролер. Контролер може вплинути на модель і визначити інше представлення;

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				19
Змн.	Арк.	№ докум.	Підпис	Дата		

– можливо реалізувати декілька представлень для одного контролера.

Контролер перехоплює події ззовні і відповідно до закладеної в нього логіки, реагує на цю подію змінюючи модель, за допомогою виклику відповідного методу. Після зміни модель формує повідомлення про те, що відбулись зміни, і все підписані на це події представлення, отримавши його, звертаються до моделі за оновленими даними для відображення.

Таким чином, можна зробити висновок, що даний архітектурний шаблон забезпечує всі технологічні можливості для побудови гнучкого додатку із дотриманням принципів SOLID.

В моєму випадку такий підхід забезпечить не тільки успішну реалізацію проекту, а й подальшу його модифікацію та розвиток у відповідності до зміни вимог користувача та процедур обробки інформації.

1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Для створення онлайн платформи доречно застосувати: мову розмітки гіпертекстових документів HTML, таблиці стилів CSS, мову програмування PHP 8, Laravel, Docker, Docker Compose, PostgreSQL в якості реляційної БД, Redis в якості не реляційної БД, яка буде виконувати функції кешування та зберігання даних в режимі реал тайм, Elasticsearch в якості пошукового двигуна, Rabbit MQ в якості брокера повідомлень.

При виборі версії **PHP 8** [11, 12, 18] є декілька переваг: швидкодія, нові можливості, типізація, зменшення використання ресурсів в порівнянні з попередніми версіями, оновлення бібліотек та інструментів.

Вибір між **PostgreSQL** [19] та **MySQL** [1, 20] залежить від потреб проекту, але є певні аспекти коли вибір PostgreSQL є підходящим. Отже, переваги PostgreSQL полягають в підтримці типів даних, розширених можливостях мови SQL, транзакціях, функціях та хранимих процедурах, спільнота та підтримка.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				20
Змн.	Арк.	№ докум.	Підпис	Дата		

Redis [21], **Aerospike** [22] і **Memcached** [23] – це три системи для кешування та зберігання даних в пам'яті, кожен з них має свої переваги та недоліки. Для розробки буде використовуватись **Redis** тому, що він багатофункціональний, є можливість зберігати дані як на диску так і в оперативній пам'яті, висока продуктивність, підтримка реплікацій.

RabbitMQ [6, 10, 24] та **Apache Kafka** [25] – брокери повідомлень, обое мають переваги та недоліки. Оскільки для реалізації використовуємо **RabbitMQ**, то ось які переваги він має в порівняно з **Apache Kafka**[26]: простота використання та налаштування, гнучкість у роботі з повідомленнями, простота інтеграцій, підтримка власних обмінів повідомленнями.

ElasticSearch [5, 8, 30] – це потужний та розширюваний пошуковий та аналітичний движок, який зазвичай використовується для індексації та швидкого пошуку великих обсягів неструктурованих даних. Переваги **ElasticSearch**: швидкодія та ефективність пошуку, легкість використання та налаштування, розширюваність та горизонтальне масштабування, можливості аналітики та візуалізації, повнотекстова підтримка, можливості розширених запитів.

Таблиця 1.2

Максимально можливий обсяг збережених даних для кожної СУБД

	Розмір БД	Розмір таблиці	Розмір рядка
MySQL	∞	256 ТБ	64 KB
PostgreSQL	∞	32 ТБ	1.6 ТБ

За критерієм тригери й збережені процедури всі альтернативи ідентичні. Усі підтримують тригери, процедури й функції.

Таблиця 1.3

Аналіз підтримуваних альтернативами операційних систем

	Windows	Linux	Unix	Android	Symbian
MySQL	+	+	+	+	+
PostgreSQL	+	+	+	-	+

Інформаційна система організації містить у собі конфіденційну інформацію, для запобігання несанкціонованого доступу застосовуються різні способи захисту. Проведемо порівняння систем забезпечення безпеки даних (табл. 1.4).

Таблиця 1.4

Аналіз систем забезпечення безпеки даних в альтернативах

	Ідентифікація	Захист від brute-force	Шифрування	Сертифікація безпеки
MySQL	+	-	+	-
PostgreSQL	+	+	+	+

Проведений аналіз таких СУБД як MySQL та PostgreSQL показав, що для виконання задач підходящим є як PostgreSQL так і MySQL. PostgreSQL має ряд суттєвих переваг над MySQL, зокрема PostgreSQL зазвичай виявляється більш гнучким та розширюваним рішенням для складних завдань баз даних.

Однією з ключових переваг PostgreSQL є його розширена система типів даних та можливість визначення користувацьких типів. Це дозволяє більш гнучко моделювати дані та реалізовувати складні структури даних. Крім того, PostgreSQL має потужну систему транзакцій та підтримує вищий рівень стандартів SQL, що робить його більш надійним та відповідним для багатьох типів додатків.

Використання **Scaleway інстансів з використанням Docker та Docker Compose** буде використовуватись в якості серверу працюючому на dev середовищі. Такий вибір був зроблений тому, що хмарна платформа Scaleway має легке розгортання та масштабування, ізолювання середовища, швидке розгортання, масштабованість, зручний моніторинг та керування, використання ресурсів ефективніше ніж в конкурентів (AWS, Azure), відокремлення сервісів, резервне копіювання та відновлення.

Таблиця 1.5

Аналіз загальних показників СУБД

	MySQL	PostgreSQL
Логічна модель даних	Реляційна	Постреляційна (об'єктно-реляційна)
Фізична модель даних	Сторінкова	Сторінкова
Типи даних	Всі основні	Всі основні, довільні, множинні значення
Індекси	Всі основні, повнотекстові	Звичайні, повнотекстові

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				22
Змн.	Арк.	№ докум.	Підпис	Дата		

Мови маніпулювання	SQL	Розширений SQL
Вбудовані мови програмування	ANSI C, ANSI C++	C, C++, Java, Perl, PHP, .Net та ін
Генератор форм, звітів	Ні	Ні
Транзакції	Так (в тому числі розподілені)	Так
Тригери процедури, що зберігаються	Так	Так (підтримка декількох мов програмування)
Платформи	MS Windows, Unix, Linux, MacOS X, Novell NetWare та ін.	MS Windows, Linux, FreeBSD, Solaris, MacOS
Область застосування	Інформаційні системи масштабу підприємства	Інформаційні системи масштабу підприємства
Особливості	Можливість логічного об'єднання БД. Широкий вибір платформ. Реплікація	Відкриті вихідні тексти. Механізм наслідування. Масштабування. Реплікація.

Висновки до першого розділу

Провівши аналіз предметної області, визначено завдання та етапи виконання завдань. Також визначено стек технологій, використовуючи які буде розроблено додаток. Було порівняно різні аналоги з метою визначення їх переваг та недоліків. Завдяки порівняльному аналізу було визначено кращі аспекти кожного додатку, які можна використати при проєктуванні власного додатку.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ОНЛАЙН ПЛАТФОРМИ ДЛЯ ПИСЬМЕННИКІВ ТА ЧИТАЧІВ

2.1 Визначення варіантів використання та структури системи

Для забезпечення постійного та безперебійного виконання необхідних нам функцій, наведено вищого рівня вимоги, які має задовільнити онлайн платформа для письменників та читачів. В загальному дана робота орієнтована на можливість об'єднати поціновувачів книг та авторів, надати можливість проявити себе – для авторів та найти собі фаворита з авторів – для читачів.

Бізнес вимоги

1. Основні цілі: дана платформа створюється з наміром забезпечити сприятливе середовище для творчості, створенню літературного співтовариства, підтримки авторів та забезпеченню авторських прав.

2. Представлення проекту: проект буде представлено у вигляді сайту, який містить зрозумілу структуру для навігації та доступ до різного літературного контенту.

Вимоги користувачів

Користувачі мають доступ до наступних функцій:

1. Можливість читати та переглядати книги.
2. Можливість коментувати.
3. Можливість створювати особистий профіль.
4. Можливість взаємодіяти з авторами.
5. Можливість створювати списки читань, тобто створювати публічні бібліотеки.
6. Можливість пошуку та фільтрації контенту.
7. Можливість отримувати сповіщення.
8. Можливість написати та опублікувати книги.

Користувачі бек-офісу – адміністратори сайту, відділ підтримки, модератор – мають доступ до наступних функцій:

1. Можливість керувати користувачами.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				24
Змн.	Арк.	№ докум.	Підпис	Дата		

2. Можливість управляти контентом.
3. Можливість вести аналітику.
4. Можливість керувати ресурсами.
5. Можливість контролювати систему безпеки та авторизацію.

Функціональні вимоги:

1. Реєстрація в платформі.
2. Можливість збереження інформації.
3. Можливість написання книги.

Нефункціональні вимоги:

1. Сприйняття
 - час, необхідний для навчання звичайних користувачів - 2-3 години, а для навчання просунутих користувачів – 1 година.
 - час відгуку для типових задач – не більше 30 секунд, для складних завдань – не більше 50 секунд.
 - інтерфейс має бути зрозумілим, зручним.
2. Надійність
 - доступність – час, що витрачається на обслуговування платформи не має перевищувати 5 % від загального часу роботи.
 - середній час безвідмовної роботи – 20 робочих днів.
 - максимальна норма помилок та дефектів в роботі платформи – 1 помилка на десять тисяч рядків коду.
3. Продуктивність
 - платформа повинна підтримувати мінімум 50 одночасно працюючих користувачів, пов'язаних із загальною базою даних.
4. Можливість експлуатації
 - масштабування – платформа повинна мати можливість збільшувати продуктивність зі збільшенням користувачів таким чином, щоб це не впливало негативно на роботу самої платформи.

Аналіз вимог користувачів дав можливість визначити варіанти використання платформи для письменників та читачів. Розглянемо детальніше кожного з них.

Таблиця 2.1

Варіант використання користувача з роллю «Гість»

 Гість	Перегляд книг
	Перегляд контенту
	Реєстрація та вхід
	Пошук книг

Таблиця 2.2

Варіант використання користувача з роллю «Зареєстрований користувач»

 Зареєстрований користувач	Додавання книг у бібліотеку
	Перегляд книг
	Пошук книг
	Можливість зворотнього зв'язку
	Сповіщення та повідомлення
	Обліковий запис
	Створення бібліотек
	Коментування та оцінювання
	Написання книг
	Додавати книги в бібліотеки
	Підписка на преміум
	Вхід на платформу

Таблиця 2.3

Варіант використання користувача з роллю «Преміум користувач»

 Преміум користувач	Додавання книг у бібліотеку
	Перегляд книг
	Пошук книг
	Можливість зворотнього зв'язку

 Преміум користувач	Сповіщення та повідомлення
	Обліковий запис
	Створення бібліотек
	Коментування та оцінювання
	Написання книг
	Додавати книги в бібліотеки
	Читання преміум книг
	Вхід на платформу

Таблиця 2.4

Варіант використання користувача з роллю «Відділ підтримки»

 Відділ підтримки	Реагування на скарги та повідомлення користувачів
	Дотримування конфіденційності

Таблиця 2.5

Варіант використання користувача з роллю «Модератор»

 Модератор	Моніторинг коментарів
	Дотримування конфіденційності
	Видалення незаконного або некоректного контенту

Таблиця 2.6

Варіант використання користувача з роллю «Адміністратор»

 Адміністратор	Моніторинг роботи сайту
	Оновлення функціональності
	Підтримка персоналу
	Аналітика та статистика
	Розробка та виконання політики сайту
	Забезпечення дотримання законів

Маючи варіанти використання платформи ми тепер можемо побудувати діаграму класів. Як згадувалось раніше, в якості архітектури буде використовуватись модель MVC, що нагадає змогу описувати для кожної сутності майже однакові класи, які по суті відрізняються лише властивостями та операціями.

Кожна сутність буде мати свій контролер, модель та набір представлень. Також для того, щоб тримати свій код в чистоті буде додано сервісний шар. До того ж ми будемо використовувати потужності Laravel для того, щоб оброблювати форму (валідація, попередня обробка даних) використовуючи FormRequests.

Кожна дія контролеру буде приймати на вхід FormRequest в якості параметру. Після чого вже провалідовані та попередньо оброблені дані будуть передаватись в сервіс. Тим часом сервіс підготує дані до запису в БД, закешує чи передасть в чергу, після чого підготує відповідь для контролера.

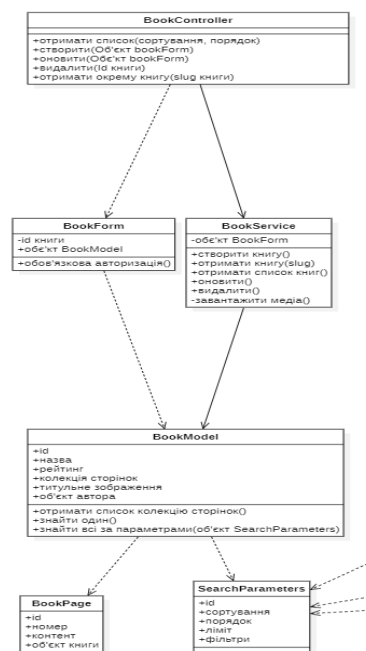


Рис. 2.1 Діаграма класів сутності книги

Обробка запиту користувача до сутності книги (рис. 2.1) починається зі звернення до контролера, який оброблює запит користувача для підготовки даних для передачі в сервіс. Сервіс сутності книг в свою чергу підготовлює дані до запису в БД, використовуючи модель, після чого модель повертає результат запису в БД назад до контролера.

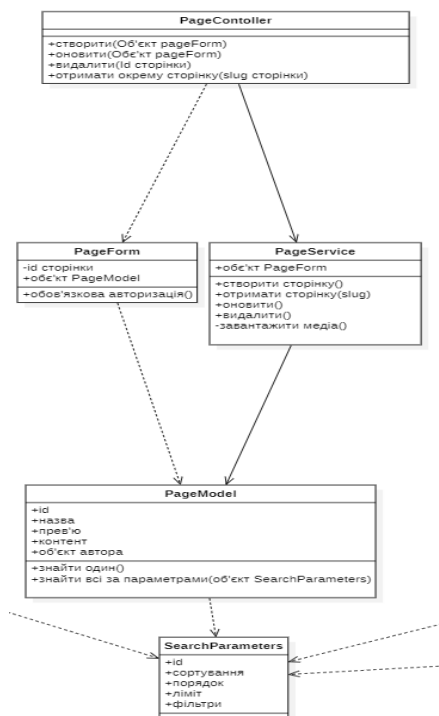


Рис. 2.2 Діаграма класів сутності сторінки

Сутність сторінки (рис. 2.2) майже нічим не відрізняється від сутності книги. Тільки відрізняється тим, що оброблює зовсім інші дані. Також в сутності книги є модель BookPage, яка відповідає за контент книги, в сутності сторінки такі зв'язки відсутні.

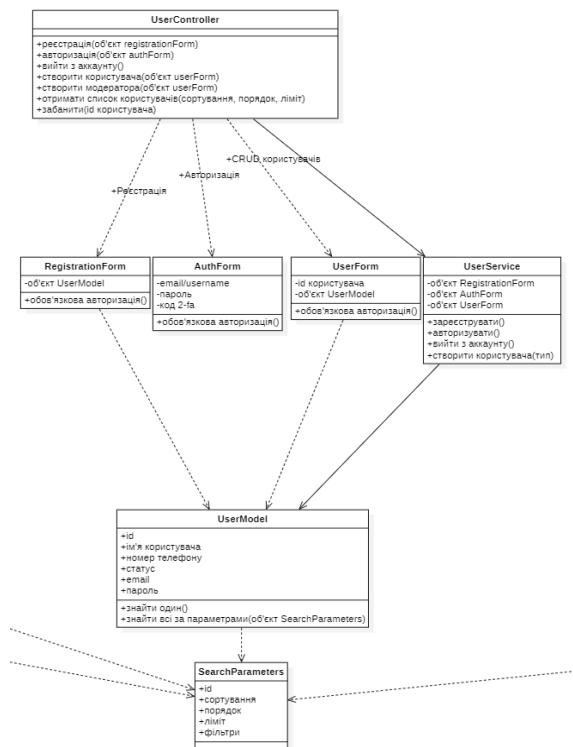


Рис. 2.3 Діаграма класів сутності користувач

Сутність «користувач» (рис. 2.3) є дещо складнішою ніж інші. Ми повинні надати можливість як використати будь-яку CRUD операцію над цією сутністю так і надати можливість реєструватись та авторизовуватись. Тому для одного контролера буде три різні FormRequest: користувач, авторизація та реєстрація. Все інше залишається так як і в інших сутностях.

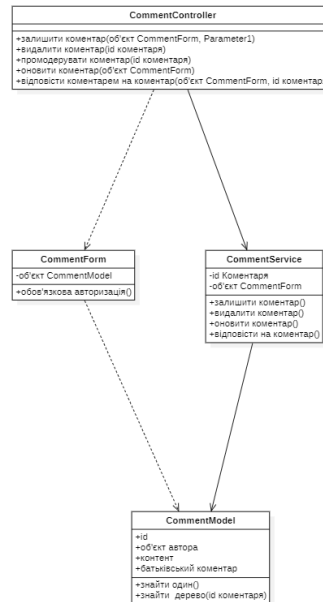


Рис. 2.4 Діаграма класів сутності коментар

Сутність коментар (рис. 2.4) за архітектурою не відрізняється від інших сутностей. Дана сутність відповідає за коментарі, які залишені під книгою та має FormRequest, який відповідає за валідацію даних з форми залишення коментаря.

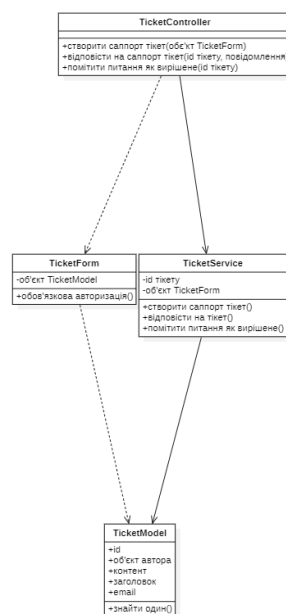


Рис. 2.5 Діаграма класів сутності зворотній зв'язок

Сутність зворотній зв'язок (рис. 2.5) відповідає за обробку форми залишення заявки на зворотній зв'язок. Також дана сутність відповідає за зміну статусу заявки на зворотній зв'язок, модератором, в адмін панелі.

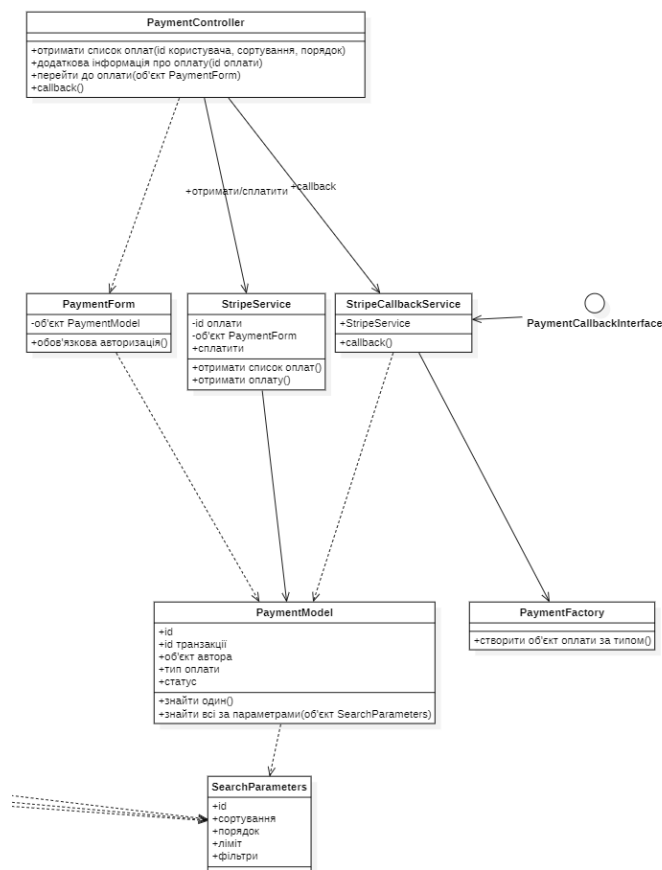


Рис. 2.6 Діаграма класів сутності оплата

Сутність «оплата» (рис. 2.6) також як і сутність «користувач» є дещо складнішою за інші, через те, що ця сутність має зовнішню інтеграцію з платіжними сервісами. Тому ми маємо розробити архітектуру для того, щоб ми могли додавати нові платіжні системи з легкістю.

Для цього було використано патерн стратегія. Як відомо платіжні системи зазвичай повертають відповідь у вигляді зворотного виклику. Тому був створений інтерфейс який будуть імплементувати всі платіжні системи, які будуть використовувати систему зворотного виклику. Щоб сервіс міг створити об'єкт потрібної наразі платіжної системи була додана фабрика.

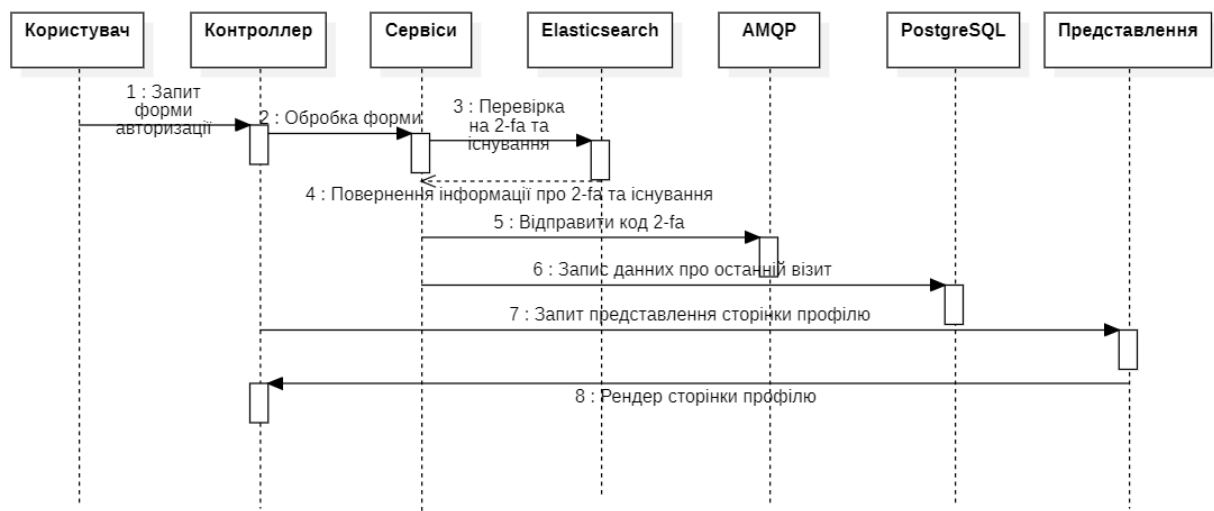


Рис. 2.7 Діаграма послідовності системи авторизації

На рис. 2.7 зображено діаграму послідовності для системи авторизації. Мета цієї діаграми – надати огляд послідовності дій, які будуть відбуватись під час проходження користувачем авторизації. Завдяки діаграмі можна візуально уявити як відбувається обмін між користувачем та бекенд системою.

Розглянемо етапи процесу авторизації: спочатку користувач заходить на саму платформу, переходить на сторінку авторизації, робить запит форми авторизації до контролеру, контролер відправляє форму на обробку сервісу, сервіс відправляє Elasticsearch на перевірку на двухфакторну автентифікацію та на існування даного користувача на порталі, Elasticsearch повертає інформацію сервісу, сервіс дає команду RabbitMQ (AMQP) відправити код з двухфакторної автентифікації користувачу на телефон, паралельно з цим сервіс передає команду PostgreSQL записати данні про останній візит, контролер дає запит на представлення сторінки профілю, представлення повертає відрендерену сторінку профілю.

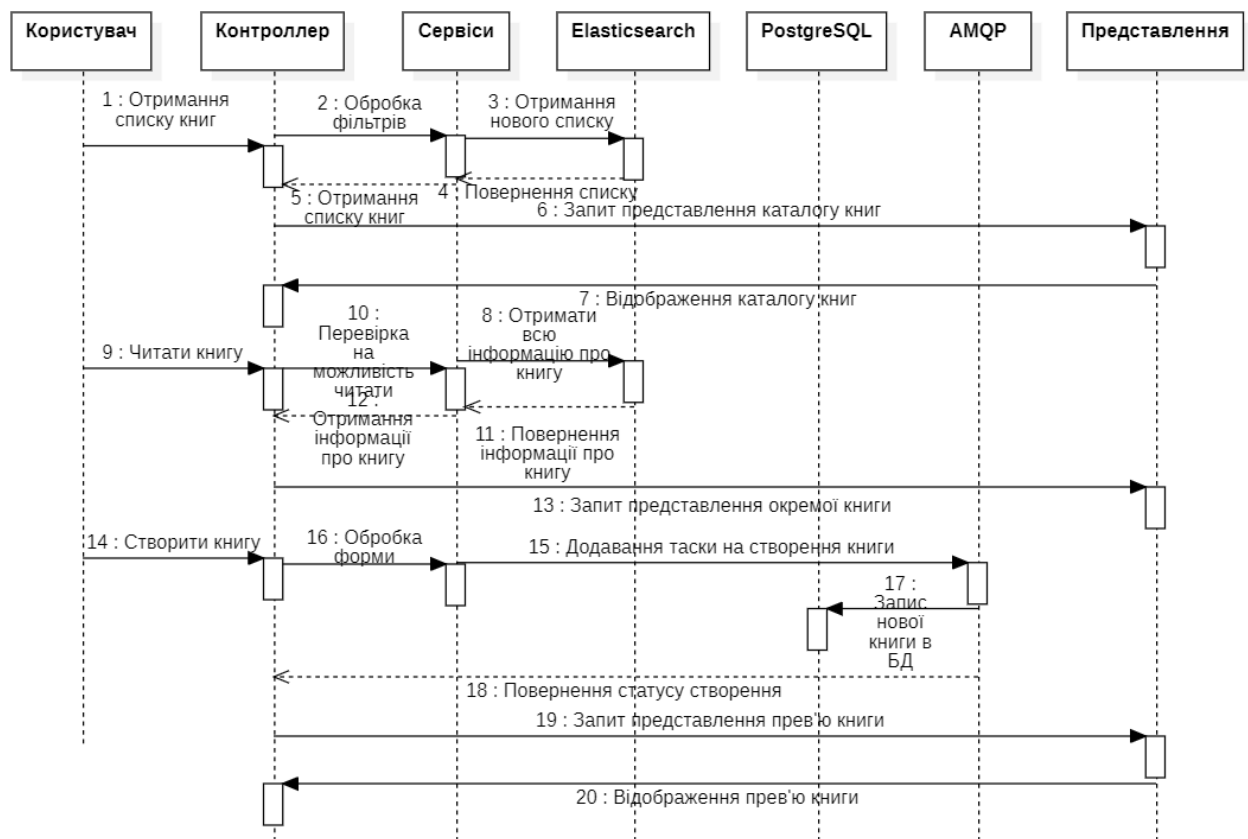


Рис. 2.8 Діаграма послідовності для сутності книги

На рис. 2.8 зображено діаграму послідовності для сутності книги. Мета цієї діаграми – описати послідовність дій, які відбуваються при додаванні, редагуванні, видаленні, перегляді книги та додавання нових книг. Ця діаграма дає нам зрозуміти, як відбувається обмін інформацією між різними компонентами системи та процеси, які відбуваються при взаємодії з книгою.

Розглянемо етапи: користувач заходить на каталог, тобто відправляє на контролер запит на отримання списку книг, і якщо користувач зазначив фільтрацію то контролер відправляє обробку фільтрів на сервіс, в тей момент сервіс відправляє Elasticsearch на отримання нового списку, потім Elasticsearch повертає сервісу новий список, сервіс відправляє отриманий список на контролер, контролер дає запит на представлення каталогу книг, представлення відображає каталог книг і т. д.

2.2 Розробка бази даних

У якості СУБД було обрано PostgreSQL, через її надійність та стійкість. Також PostgreSQL по замовчуванню має підтримку InnoDB, що робить її привабливішою над її головним конкурентом MySQL. Також PostgreSQL має відкритий код, через що постійно оновлюється, дозволяє уникати витрат на ліцензії та мати повний контроль над кодом.

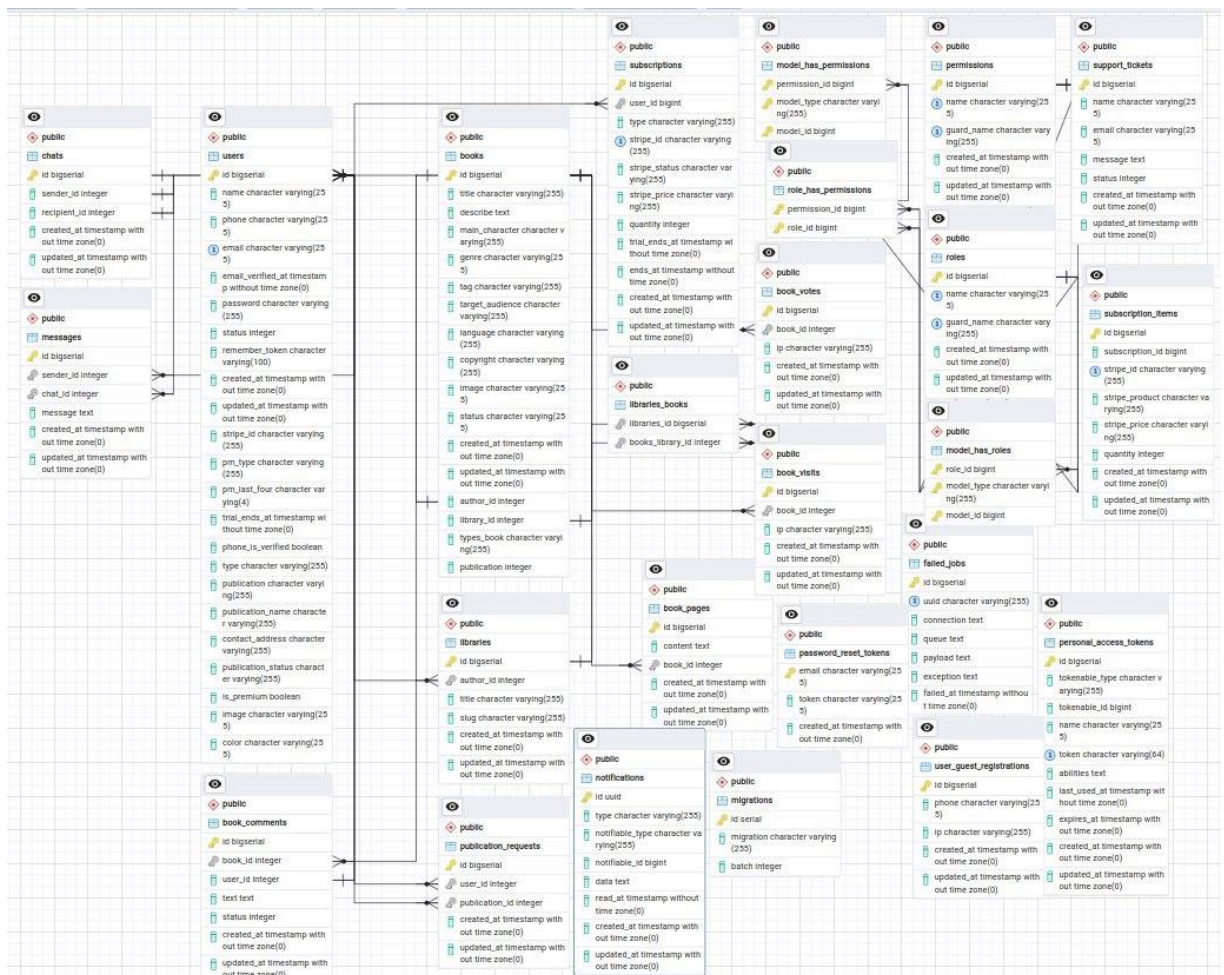


Рис. 2.9 Діаграма БД

База даних додатку складається з 24 таблиць (рис. 2.9). Основними є book_comments, book_pages, book_visits, book_votes, books, chats, libraries, messages, model_has_permissions, model_has_roles, permissions, publication_requests, role_has_permissions, roles, subscription_items, subscriptions, support_tickets, user_guest_registration, users. Решта таблиць була згенерована

самим Laravel – failed_jobs, migrations, password_reset_tokens, personal_access_tokens, notifications.

Підключення до БД відбувається наступним чином:

Лістинг коду:

```
DB_CONNECTION=pgsql
DB_HOST=postgres
DB_PORT=5432
DB_DATABASE=literary-loom
DB_USERNAME=
DB_PASSWORD=
```

Основною сутністю проекту є книги (таблиця books), через що вона має найбільше логіки. Книги мають поля, які описують її: назва, короткий опис, жанр, теги, головні герої, цільова аудиторія, мова на якій написана книга, ліцензія на книгу, титульне зображення та статус книги. Також книга має її автора (author_id).

Видавництво, від якого була написана книга, буде видавництво самого автора. Також автор може змінювати статус написання книги (в процесі, закинуто та завершена).

Також є поле library_id, яке вказує на бібліотеку до якої була додана ця книга. Поля created_at та updated_at вказують на дату коли книга була останній раз створена та оновлена.

Вміст книги зберігається в окремій таблиці (book_pages). Це зроблено для того, щоб не зберігати весь вміст в одному полі так як вміст може бути дуже великим та для можливості динамічно підвантажувати вміст книги.

Користувач пов'язаний майже зі всіма сутностями проекту, так як зареєстрований користувач має багато можливостей на сайті. Він має ім'я та пароль, яке він вводить при реєстрації на сайту. Також Laravel створив поля до користувача для верифікації email (email_verified_at та remember_token). Телефон користувача вводиться також при реєстрації, але його треба ще підтвердити. Статус підтвердження номеру телефону зберігається в полі phone_is_verified. Також користувач має поле status, яке містить інформацію про те чи користувач має доступ до сайту та може змінюватись через адмін панель.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				35
Змн.	Арк.	№ докум.	Підпис	Дата		

Користувач також має поля `created_at` та `updated_at`, які вказують на дату коли користувач був останній раз створений та оновлений.

Поле `stripe_id` містить ідентифікатор останньої транзакції в Stripe, для того щоб в разі помилки оплати, було легко відловити помилку. Поле `type` (тип користувача) може містити рядки `user` або `publication` для ідентифікації, щоб знати який саме це тип користувача (письменник або видавництво). Тип користувача не може бути змінений після реєстрації.

Поле `publication_name` доступне лише для видавництва, так як містить назву поточного видавництва. В такому разі поле `name` буде містити ім'я власника видавництва. Поле `publication` доступне лише для письменників, це ідентифікатор на користувача (видавництва), яке письменник вибере при реєстрації. Поле `publication_status` містить статус верифікації письменника видавництвом, може містити такі значення: `approved`, `pending`, `rejected`.

Поле `is_premium` це прапорець, який вказує чи користувач має доступ до преміального вмісту. Цей прапорець активується після успішної оплати пакету в Stripe.

Поле `image` це назва зображення, яке користувач буде використовувати в якості аватару на порталі.

Поле `color` містить хеш кольору, який автоматично буде згенерований при реєстрації та використовуватись в якості кольору банеру в публічному профілі користувача (цей колір користувач має змогу регенерувати через налаштування профілю).

Таблиця `publication_requests` містить ідентифікатори на письменника та видавництво, якщо в даній таблиці є запис, то це значить, що якесь видавництво не підтвердило співавторство з письменником.

Бібліотека (таблиця `libraries`) має автора бібліотеки, тобто користувача який її створив (`author_id`). В бібліотеки є її назва (поле `title`) та посилання по якому буде доступна бібліотека (поле `slug`).

Чат між користувачами (таблиця `chats`) має лише ідентифікатори на відправника та отримувача повідомлень (поля `sender_id` та `recipient_id`).

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				36
Змн.	Арк.	№ докум.	Підпис	Дата		

Повідомлення в чаті зберігаються в окремій таблиці (messages). В даній таблиці є ідентифікатор на користувача, який відправив повідомлення та ідентифікатор на чат, до якого буде повідомлення відправлено. Також в полі message зберігається вміст повідомлення.

Таблиця user_guest_registration містить телефон та ір-адресу. В цій таблиці будуть зберігатись користувачі, які почали заповнювати форму реєстрації, але щось їм завадило відправити форму протягом 5 хвилин. Інформація з цієї таблиці може використовуватись користувачами адмін панелі для продзвону користувачів та для появи модального вікна «Ви все ще тут?» у користувача, який не закінчив процес реєстрації.

Таблиці book_visits та book_votes містять ідентифікатори книги та ір-адреси користувачів, які проглянули книгу, або проголосували за неї. Також використовується для генерації статистики переглядів та самих популярних книг в адміністративній панелі.

Таблиця support_tickets містить інформацію з форми заявки на зворотній зв'язок та використовується відділом підтримки в адмін панелі.

Також є низка таблиць, які були згенеровані самим фреймворком «failed_jobs», «migrations», «password_reset_tokens», «personal_access_tokens», «notifications» або розширеннями які встановлювались при розробці проекту (Stripe, Filament-shield).

2.3 Проектування та реалізація алгоритмів роботи додатку

Зайшовши на сайт, користувач бачить перед собою головну сторінку, де він також може зареєструватись або авторизуватись, переглянути певні книги або ж перейти до каталогу, подивитись список преміум-книжок та написати в підтримку.

Отже, розглянемо діаграму діяльності для авторизації (рис. 2.9).

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				37
Змн.	Арк.	№ докум.	Підпис	Дата		

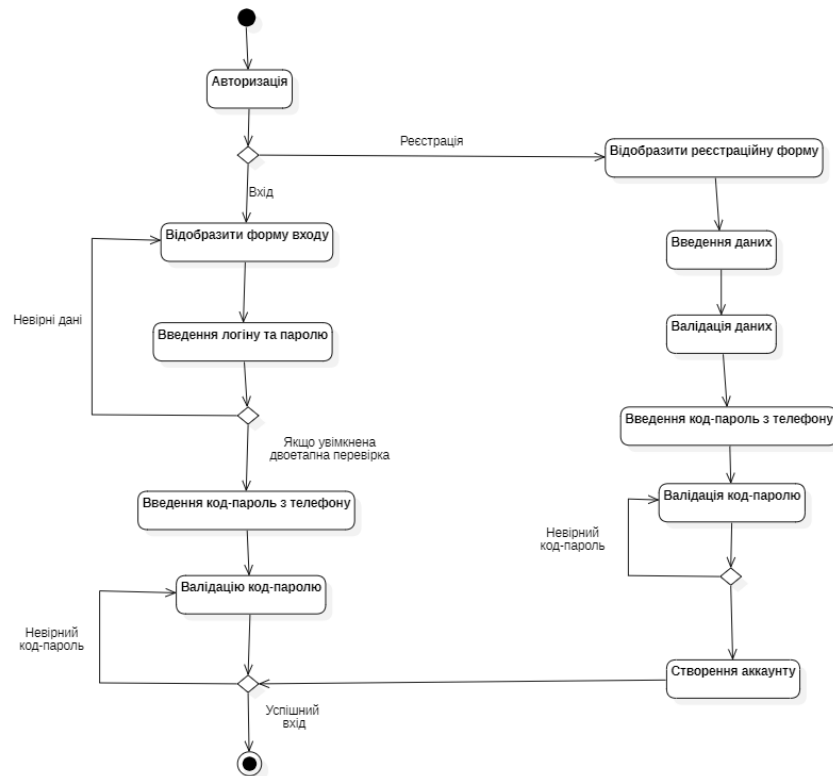


Рис. 2.10 Діаграма діяльності для авторизації

На рис. 2.10 зображено діаграму діяльності для авторизації. Вона включає у себе послідовність етапів, які користувач пройде для успішного входу в систему.

Етапи діаграми:

- початок
- потім користувач має перед собою вибір або вхід на платформу або реєстрацію
- вибравши реєстрацію перед користувачем відображається реєстраційна форма
- після чого користувачу необхідно ввести облікові данні
- після чого відбувається валідація даних
- якщо данні не вірні то ми повертаємось на введення облікових даних
- якщо все правильно то користувачу на телефон приходить код-пароль
- після чого він має його ввести
- потім відбувається валідація код-паролю

- якщо код-пароль невірний то вводимо повторно
- якщо все добре то завершуємо реєстрацію
- для входу на платформу – спочатку користувач бачить перед собою форму входу
- після чого вводить логін та пароль
- потім відбувається перевірка даних
- якщо неправильно введено дані то повторно вводимо їх
- якщо все добре то вводимо код-пароль з телефону
- потім відбувається перевірка код-пароллю
- у випадку якщо код-пароль не правильний то вводимо заново
- якщо все добре то входимо на платформу.

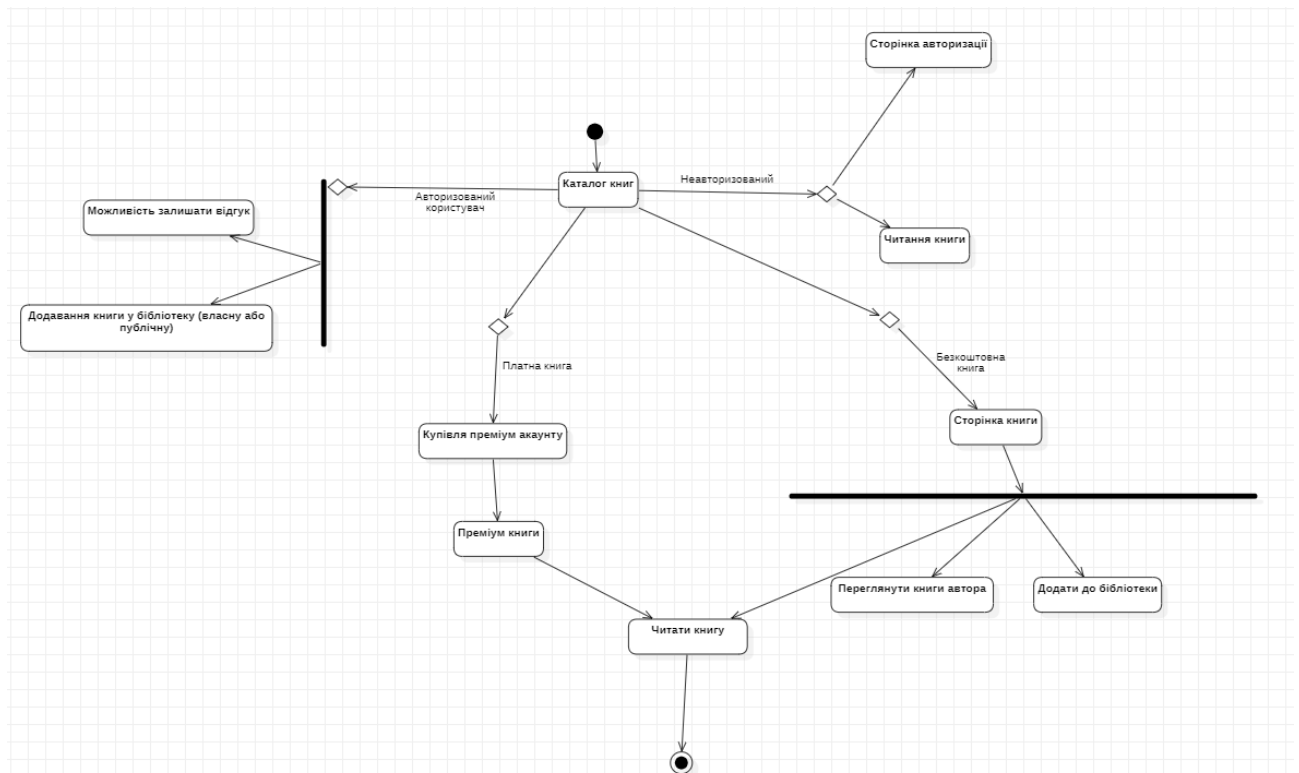


Рис. 2.11 Діаграма діяльності для читання книги

На рисунку вище зображено діаграму діяльності для читання книги. Вона моделює взаємодію користувача з системою під час процесу читання книги на платформі.

Етапи діаграми: користувач обирає книгу, яку бажає прочитати. Це може бути виконано шляхом перегляду каталогу, введення назви чи використання

рекомендаційної системи, користувач активно читає вміст, коли читач завершує читання, він має можливість залишити відгук книзі, або додати її до бібліотеки для подальшого зручного пошуку. Також читаючи книгу користувач має можливість побачити книги автора. Якщо книга платна, користувачу потрібно купити преміум акаунт, для того, щоб мати можливість прочитати книгу.

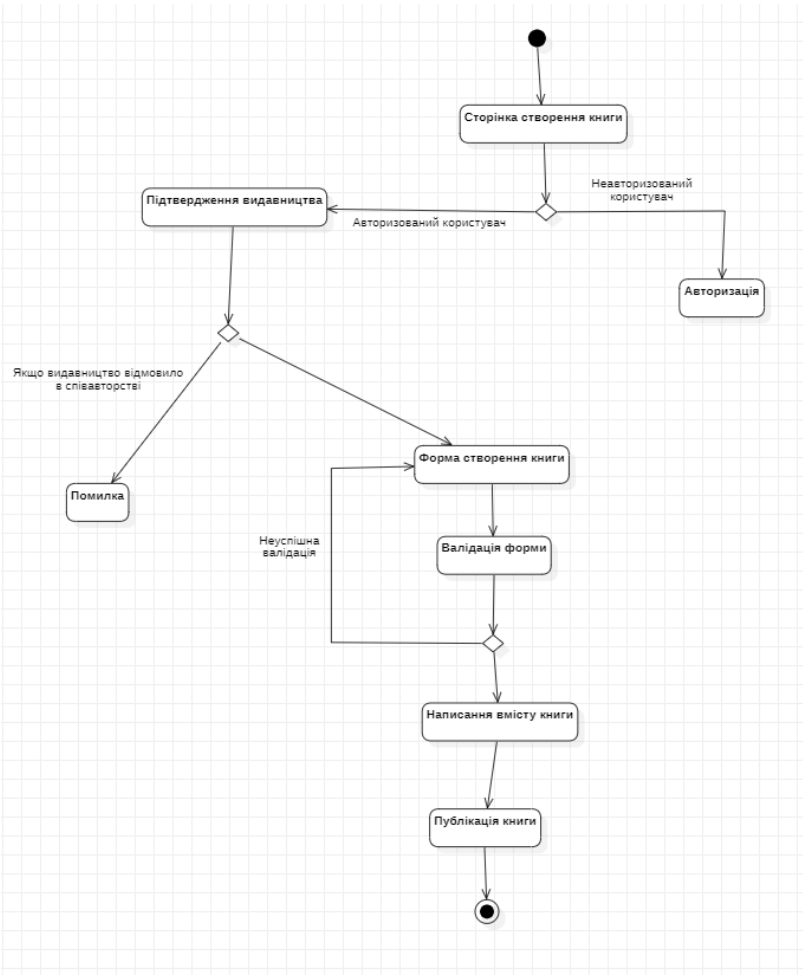


Рис. 2.12 Діаграма діяльності написання книги

Діаграма діяльності для написання книги (рис. 2.12) моделює взаємодію автора з системою під час творчого процесу написання книги на платформі.

Етапи діаграми: спочатку користувач бачить перед собою сторінку створення книги, потім йде перевірка на те, чи зареєстрований користувач, якщо ні користувача відправляє на сторінку авторизації, якщо користувач зареєстрований, він ме отримати доступ на написання книг від імені видавництва. Якщо видавництво відмовило користувачу, то приходить сповіщення про те, що видавництво відмовило, тому ддоступу до написання книг немає, потрібно вибрати інше видавництво. Якщо видавництво

підтвердило користувача то перед ним з'являється форма створення книги, після введення даних йде валідація форми, якщо форма не валідна то показуємо форму знову, для вводу нових даних, якщо валідна то перед собою користувач бачить сторінку для написання вмісту книги, після того користувач може опублікувати її.

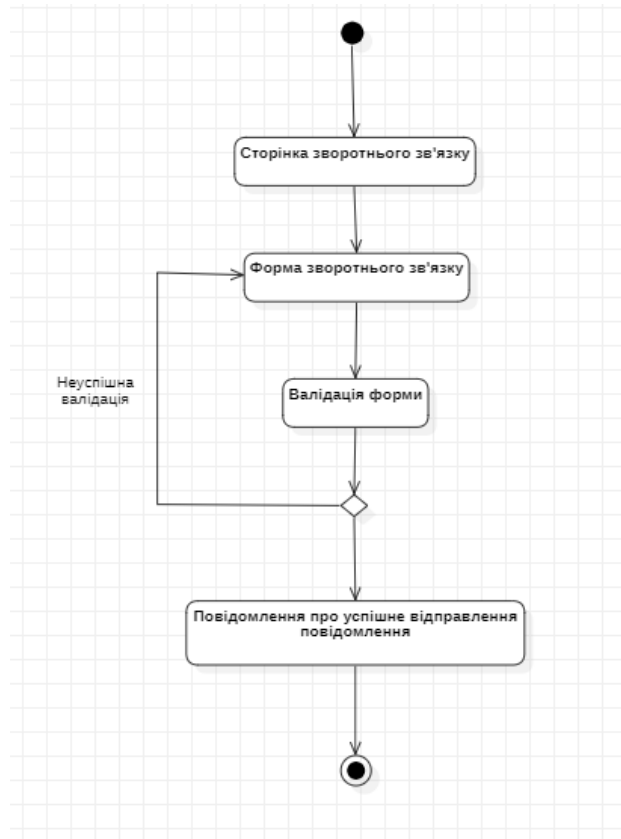


Рис. 2.13 Діаграма діяльності зворотнього зв'язку

На діаграмі діяльності зворотнього зв'язку (рис. 2.13) описано взаємодію користувача з системою під час використання форми зворотнього зв'язку на онлайн платформі для письменників та читачів.

Етапи діаграми: перед собою користувач бачить сторінку зворотнього зв'язку, на тій сторінці користувач бачить перед собою форму зворотнього зв'язку, після відправки форми йде валідація форми, якщо валідація неуспішна, то повертається на форму зворотнього зв'язку для повторного заповнення форми, якщо все добре то повідомлення про успішне відправлення повідомлення.

2.4 Реалізація функціоналу додатку

2.4.1. Реалізація основної архітектури додатку

Проект побудований на базі PHP фреймворку Laravel, використовується така ж сама архітектура як і у фреймворку MVC. Додатково було додано ще один шар до архітектури – сервісний шар, в якому буде зберігатись бізнес логіка проекту. Також додані окремий модуль для реалізації адміністративної панелі, яка має власні контролери, моделі, представлення та ресурси.

Для впровадження системи оплати було використано патерн стратегія, що дозволить в майбутньому без проблем додавати нові платіжні системи, також стратегію використовують для SMS-шлюзу.

Також в проекті використовується ще один патерн - наглядач. Він слугує посередником між двома базами даних, тобто коли Laravel додає, оновлює або видаляє дані з PostgreSQL ці дані автоматично змінюються і в Elasticsearch, у випадку якщо Elasticsearch індексу не існує, цей індекс буде автоматично створений.

Зазвичай для індексації даних в окремій БД, використовується крон-команди, тобто в кожен заданий проміжок часу дві БД будуть синхронізуватись. Використовуючи такий підхід, використовуючи наглядач, ми вирішуємо проблему з тим, що деякий період часу дані не будуть однаковими, але в такому разі ми жертвуємо швидкістю оновлення даних, так як дані потрібно оновлювати в двох БД замість однієї нальоту.

Було додано до архітектури власного посередника між запитом користувача та контролером (Middleware), який контролює верифікацію користувача/видавництва. Тобто додавши конфігурацію до роуту ми можемо сказати контролеру, що цей запит неможливо виконати, через те, що користувач не підтверджений видавництвом.

Також було додано зміни в посередник, який написано розробниками Laravel (Authenticate). Для того, щоб перевіряти, що користувач є не лише

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				42
Змн.	Арк.	№ докум.	Підпис	Дата		

авторизованим до системи, а й підтвердив свій номер телефону. Таким чином ми використовуємо конструкцію вказану нижче (лістинг коду). Ми можемо автоматично додати перевірку на верифікацію номеру телефону.

Лістинг коду:

->middleware("auth")

Для розгортання проекту на локальному та дев середовищі, було використано Docker композицію, яка була написана за допомогою Docker Compose. Композиція має такі сервіси:

- Backend – налаштовує php зі всіма потрібними для роботи проекту модулями та composer.
- Postgres – основна БД проекту (реляційна).
- Nginx – налаштовує сервер на базі nginx та підгружає SSL-сертифікат. Також в конфігурації налаштований редірект з http на https.
- Redis – нереляційна БД, яка використовується для кешування, зберігання сесій користувача та зберігання код-пароллю при підтвердженні номеру телефону.
- Node – використовується для побудови frontend частини додатку.
- Pgadmin – доступ до PostgreSQL БД за допомогою браузера.
- Elasticsearch – документо-орієнтована БД, яка використовується для глобального пошуку по сайту.
- Kibana – доступ до Elasticsearch БД за допомогою браузера.

Лістинг коду host.conf:

```
server {  
    listen 80;  
    listen [::]:80;  
    server_name literary-loom;  
    root /var/www/html/public;  
  
    add_header X-Frame-Options "SAMEORIGIN";  
    add_header X-Content-Type-Options "nosniff";  
  
    index index.php;  
  
    charset utf-8;
```

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				43
Змн.	Арк.	№ докум.	Підпис	Дата		

```

location / {
    try_files $uri $uri /index.php?$query_string;
}

location = /favicon.ico { access_log off; log_not_found off; }
location = /robots.txt { access_log off; log_not_found off; }

error_page 404 /index.php;

location ~ /\.php$ {
    fastcgi_pass backend:9000;
    fastcgi_split_path_info ^(.+\.php)(/.*)$;
    include fastcgi_params;

    fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
    fastcgi_param DOCUMENT_ROOT $realpath_root;

    internal;
}

location ~ /\.(!well-known).* {
    deny all;
}
}

```

2.4.2. Реалізація системи реєстрації та авторизації

При реєстрації користувач може вказати, чи він буде вказаний як видавництво, чи письменник. Обидва типи реєстрації оброблюються за допомогою одного методу контролера. Для ідентифікації типу користувача використовується приховане поле. Також форма має ще два прихованих поля: mode та csrf. Mode відповідає за те, чи можна буде підтвердити номер телефону, використовуючи код «111111», а csrf містить csrf-token для захисту від міжсайтової підробки запиту.

Лістинг коду signup.blade.php:

```

<form action="{{url('register')}}" method="post">
    @csrf
    <input type="hidden" name="type" value="publication">
    <input type="hidden" name="mode" value="{{ $mode }}">

```

Після того як користувач вдало відправить форму його перенаправить на сторінку підтвердження номеру телефону, це зроблено для того, щоб

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				44
Змн.	Арк.	№ докум.	Підпис	Дата		

zareestrovani korystuvachi buili lishe ti, xto mas realnyi nomer telefonu, dlia zahistu vid botiv. Pislja toho yak korystuvach otrimaє SMS z kod-parolem vin mas vvesti yogo u formu.

Yakщо виникли проблеми z відправкою SMS na nomer telefonu, vin mas mozhlivіst' відправити SMS povtorno. U vipadku nevirno vvedenogo nomeru telefonu, korystuvach mas mozhlivіst' zminiti yogo.

Лістинг коду UserController:

```
$user = new User();
$user->phone = $userPhone;
$user->password = Hash::make($signupRequest->get('password'));
$user->status = 1;
$user->type = $signupRequest->get('type');
$user->name = $signupRequest->get('name');
$user->email = $userEmail;

if ($user->type === 'user') {
    $user->publication = $signupRequest->get('publication');
    $user->publication_status = 'pending';
}

if ($user->type === 'publication') {
    $user->publication_name = $signupRequest->get('publication_name');
}
$user->save();

if ($user->type === 'user') {
    $publicationRequest = new PublicationRequest();
    $publicationRequest->user_id = $user->id;
    $publicationRequest->publication_id = $user->publication;
    $publicationRequest->save();
}

$redis = Redis::connection();
$redis->del('registration_monitor_data_' . Request::ip());
$redis->del('registration_monitor_track_' . Request::ip());
Auth::login($user);

$privateLibrary = new Libraries();
$privateLibrary->title = 'Private';
$privateLibrary->author_id = Auth::id();

$privateLibrary->save();

return redirect('/phone-verification?mode=' . $signupRequest->get('mode'));
```

2.4.3. Реалізація системи підтвердження номеру телефону

Для реалізації системи відправки СМС повідомлення з код-паролем на номер телефону користувача використовується СМС-шлюз SMSClub. Також впроваджено можливість додавати інші СМС-шлюзи, та перемикатись між ними, використовуючи патерн – стратегія. Кожна реалізація відправки СМС-повідомлення має імплементувати інтерфейс SMSServiceInterface, чим приводить всі реалізації відправки СМС до одного вигляду.

Лістинг коду SMSServiceInterface:

```
<?php
declare(strict_types=1);
namespace App\Services\SMS;

interface SMSServiceInterface {
    public const SMS_CLUB_API_ENDPOINT = 'https://im.smsclub.mobi/sms/send';
    public const SMS_CLUB_API_KEY = '_mBobh61rYnLPcP';

    public function sendPhoneVerificationCode(int $code, string $phone): array;
}
```

Для того, щоб запобігти спаму через повторну відправку СМС-повідомлень з код-паролем, було реалізовано таймер, який робить відлік часу по закінченню якого користувач може зробити повторну відправку код-паролем. Для реалізації таймеру було використано бібліотеку JQuery countdownTimer.

Лістинг коду app.js:

```
$('.phone-verification-timer').countdownTimer({
    seconds: secondsToNewTry,
    loop: false,
    callback: function () {
        $('.resend-verification-code').show();
        $('.verification-code-resend-timer').hide();
    }
});
```

2.4.4. Реалізація системи купівлі преміум аккаунту

Для того, щоб впровадити можливість купувати преміум пакет на сайті, було додано інтеграцію з платіжним сервісом Stripe. Для того, щоб зробити інтеграцію розширюваною та коректною використано бібліотеку Laravel Cashier.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				46
Змн.	Арк.	№ докум.	Підпис	Дата		

Для перенаправлення користувача на платіжний шлюз додано завантаження сесії Stripe та перенаправлення користувача за посиланням, яке повертає Stripe після створення сесії.

Лістинг коду PaymentController:

```
public function pay(Request $request): Checkout
{
    return $request->user()->newSubscription('default', 'price_1Oi3a4KRRBjPclevDaXQ6UN8')
        ->trialDays(30)
        ->withMetadata(['user_id' => \Auth::id()])
        ->checkout([
            'success_url' => route('pay.pay-success'),
            'cancel_url' => route('pay.failed-success')
        ]);
}
```

Після оплати пакету користувачем Stripe робить запит до /stripe/webhook, в якому робиться обробка відповіді від Stripe при якому у випадку успішної оплати пакету додається інформація про транзакцію та активується прапорець у користувача про те, що він має доступ до преміального контенту.

Лістинг коду StripeEventListener:

```
public function handle($event): void
{
    if ($event->payload['type'] !== 'invoice.payment_succeeded') {
        return;
    }

    $user = User::find($event->payload['data']['object']['lines']['data'][0]['metadata']['user_id']);
    $user->is_premium = true;

    $user->save();

    $subscription = new Subscription();
    $subscription->user_id = $event->payload['data']['object']['lines']['data'][0]['metadata']['user_id'];
    $subscription->type = $event->payload['data']['object']['lines']['data'][0]['metadata']['type'];
    $subscription->stripe_id = $event->payload['id'];
    $subscription->stripe_status = $event->payload['data']['object']['status'];
    $subscription->stripe_price = $event->payload['data']['object']['lines']['data'][0]['plan']['amount'];
    $subscription->quantity = $event->payload['data']['object']['lines']['data'][0]['quantity'];
    $subscription->save();
}
```

Також додано можливість додавати інші платіжні шлюзи, використовуючи патерн – стратегію. Кожен обробник відповіді платіжного шлюзу повинен імплементувати інтерфейс – PaymentEventListenerInterface.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				47
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду PaymentEventListenerInterface:

```
<?php

namespace App\Listeners;

interface PaymentEventListenerInterface
{
    public function handle($event): void;
}
```

2.4.5. Реалізація системи взаємодії Elasticsearch та PostgreSQL

В проєкті використовується дві БД: PostgreSQL та Elasticsearch. На відміну від PostgreSQL, Elasticsearch містить дані лише двох сутностей – користувачів та книг. Elasticsearch використовується лише для глобального пошуку по сайту.

На відміну від PostgreSQL в Elasticsearch потрібно індексувати дані, які ми хочемо додавати в нього, тому була проблема з відмінністю даних в двох БД. Зазвичай для індексації в Elasticsearch використовують консольні команди, які запускаються в заданий проміжок часу, використовуючи cron.

В реалізації проєкту виправлено проблему з індексацією, використовуючи патерн – наглядач. Наглядач прослуховує збереження та видалення даних з PostgreSQL та зберігає або додає їх в Elasticsearch. Також прибрано індексацію користувачів без номеру телефону для того щоб не індексувати користувачів в адмін панелі, користувачів в адмін панелі можна ідентифікувати по відсутності номеру телефону. При оновленні даних в PostgreSQL вони будуть видалятися та створюватися нові в Elasticsearch. Це реалізовано в самій бібліотеці для роботи з Elasticsearch.

Лістинг коду ElasticsearchObserver:

```
public function saved($model)
{
    if ($model instanceof User && !$model->phone) {
        return;
    }

    $this->elasticsearch->index([
        'index' => $model->getSearchIndex(),
        'type' => $model->getSearchType(),
        'id' => $model->getKey(),
        'body' => $model->toSearchArray(),
    ];
```

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				48
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    });
}

public function deleted($model)
{
    if ($model instanceof User && !$model->phone) {
        return;
    }

    $this->elasticsearch->delete([
        'index' => $model->getSearchIndex(),
        'type' => $model->getSearchType(),
        'id' => $model->getKey(),
    ]);
}

```

Для пошуку по ElasticSearch використовуються відповідні індекси – books та users. Пошук здійснюється повнотекстовий, тобто якщо користувач вводить не зовсім коректний пошуковий запит то записи, які мають схожий вміст теж будуть видаватись за результатами пошуку. Але більший пріоритет мають записи з більшою схожістю до запиту.

Лістинг коду пошуку книг:

```

$searchResult = $elasticsearch->search([
    'index' => 'books',
    'body' => [
        'query' => [
            'multi_match' => [
                'query' => $query,
                'fields' => [
                    'title', 'describe', 'main_character', 'tag'
                ],
                'type' => 'bool_prefix',
                'minimum_should_match' => '10%'
            ],
        ],
    ],
]);
$books = Books::whereIn('id', array_column($searchResult['hits']['hits'], '_id'))->get();

```

ElasticSearch повертає лише ідентифікатори (без даних) за якими витягуються повні дані про сутність з PostgreSQL. В ElasticSearch дані про сутність зберігаються лише для того, щоб по ним зберігати лише ідентифікатори.

Висновки до другого розділу

В другому розділі визначено функціональні та нефункціональні вимоги до онлайн платформи, представлено моделювання процесу створення веб-додатку.

Представлено модуль авторизації та реєстрації, профіль користувача, модуль для надання можливості створювати, редагувати та повідомляти про це своїх підписників, модуль підписників, модуль чату, модуль для пошуку книг та користувачів та модуль написання книги.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				50
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ ОНЛАЙН ПЛАТФОРМИ ДЛЯ ПИСЬМЕННИКІВ ТА ЧИТАЧІВ

3.1 Порядок встановлення та налаштування параметрів додатку

Додаток складається з низки компонентів для розгортання яких використовується Docker Compose. Використовуючи такий підхід, користувач має можливість розгорнути будь-яку кількість компонентів системи, використавши одну консольну команду.

Лістинг команди запуску docker композиції:

```
docker compose up -d
```

Після запуску команди для створення docker композиції, система Docker виконає всі потрібні команди для завантаження та базового налаштування всіх компонентів системи.

Компоненти між собою взаємодіють за назвами сервісів, які прописані в docker композиції, тобто якщо нам потрібен хост БД, то замість IP-адреси серверу та порта на якому встановлена БД прописуємо postgres. Це стає можливим через те, що докер працює на локальній IP-адресі 0.0.0.0 та додає потрібний порт, який знаходить по назві сервісу в docker композиції.

Для налаштування окремого компоненту системи, використовується Dockerfile, в якому прописані Linux команди потрібні для налаштування компоненту. Як приклад, компонент php якому для розгортання потрібен встановлений composer та низка php-модулів.

Лістинг коду Dockerfile:

```
FROM composer:2.2 AS composer-image
FROM php:8.2-fpm AS app
```

```
RUN apt-get update --fix-missing
RUN apt-get -y install postgresql
RUN apt-get -y install libpq-dev
RUN docker-php-ext-install pdo pdo_pgsql
RUN docker-php-ext-install sockets
```

```
RUN pecl install -o -f redis \
&& rm -rf /tmp/pear \
&& docker-php-ext-enable redis
COPY --from=composer-image /usr/bin/composer /usr/local/bin/composer
```

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				51
Змн.	Арк.	№ докум.	Підпис	Дата		

```

RUN apt-get install -y libmagickwand-dev --no-install-recommends \
  && pecl install imagick \
  && docker-php-ext-enable imagick \
  && apt-get install -y zlib1g-dev \
  && apt-get install -y libgmp-dev \
  && docker-php-ext-install gmp \
  && docker-php-ext-enable gmp \
  && apt-get install -y libtidy-dev libffi-dev libc-client-dev libkrb5-dev --no-install-recommends \
  && docker-php-ext-configure imap --with-kerberos --with-imap-ssl \
  && docker-php-ext-install -j$(nproc) imap \
  && docker-php-ext-enable imap \
  && apt-get install -y gcc make libssh2-1-dev libssh2-1 \
  && pecl install ssh2 \
  && docker-php-ext-enable ssh2 \
  && apt-get install -y libmsgpack-dev \
  && docker-php-ext-install -j$(nproc) opcache

```

Для коректної роботи проекту потрібно мати nginx, роль якого є перенаправити запит з браузеру в потрібному напрямку. Тобто якщо користувач намагається зайти на сайт, використовуючи протокол http, nginx перенаправляє його на аналогічну адресу з протоколом https. Як тільки з браузеру прийде коректна адреса, nginx каже, що потрібно завантажити файл public/index.php та використовуючи fastcgi вкаже, що цей запит буде виконувати компонент php.

Дана платформа була створена на Laravel, налаштування середовища відбувалось в файлі .env, цей файл створювався на основі .env.example.

В файлі .env відбувається налаштування бази даних, Redis, Mail, Sentry, Elasticsearch, RabbitMQ та Stripe.

Потім в середині контейнеру де знаходиться php, за допомогою Composer завантажуюмо та встановлюємо всі необхідні бібліотеки, які зазначені у файлі composer.json. За допомогою npm встановлюємо бібліотеки JavaScript, також всередині контейнеру node.

Після виконання цих дій потрібно запустити міграції бази даних в середині контейнеру php, для створення необхідних нам таблиць. При потребі, можна запустити сіди для наповнення бази даних.

Також потрібно не забувати про адміністративну панель, потрібно створити користувача та налаштування ролів та дозволів в адмін. панелі.

Після правильного виконання функцій можна відкривати сайт.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				52
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2 Структура інтерфейсу веб-додатку та порядок роботи

У веб-додаток користувачі можуть зайти як з персонального комп'ютера, так і з мобільного телефону. Сайт повністю адаптивний для мобільних пристроїв.

Перейшовши на сайт перед користувачем буде відображено головну сторінку (рис. 3.1).

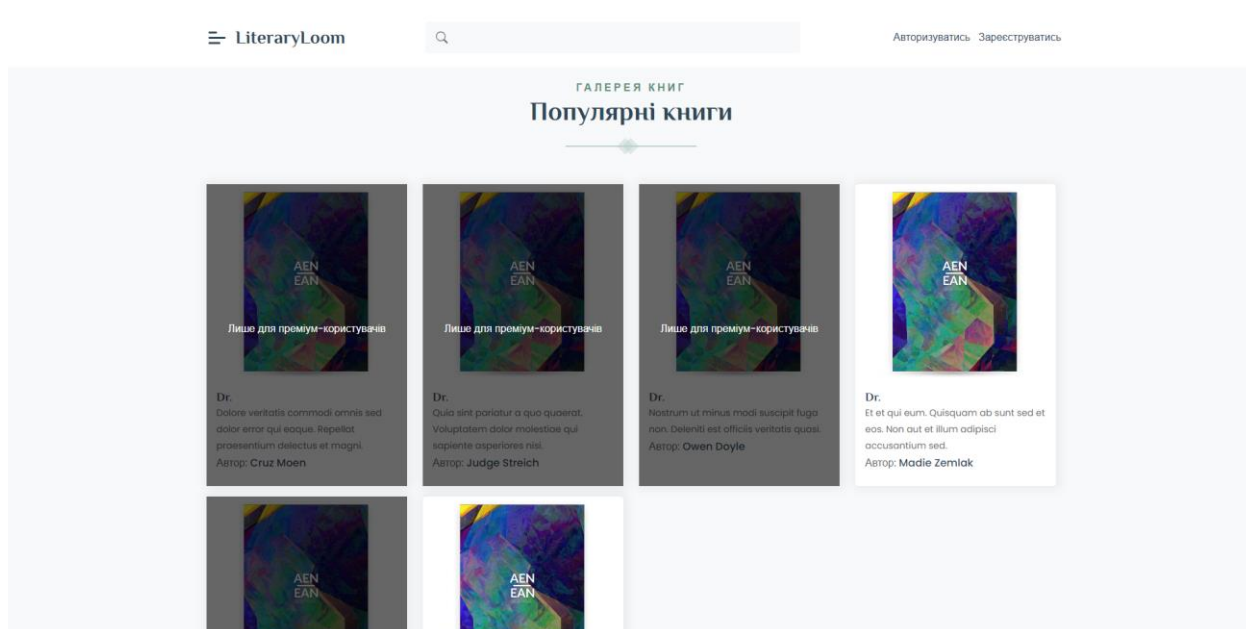


Рис. 3.1 Головна сторінка

В лівому кутку кнопка бокового меню, де при натисканні на неї перед користувачем буде відображено меню з доступними пунктами (рис. 3.2).

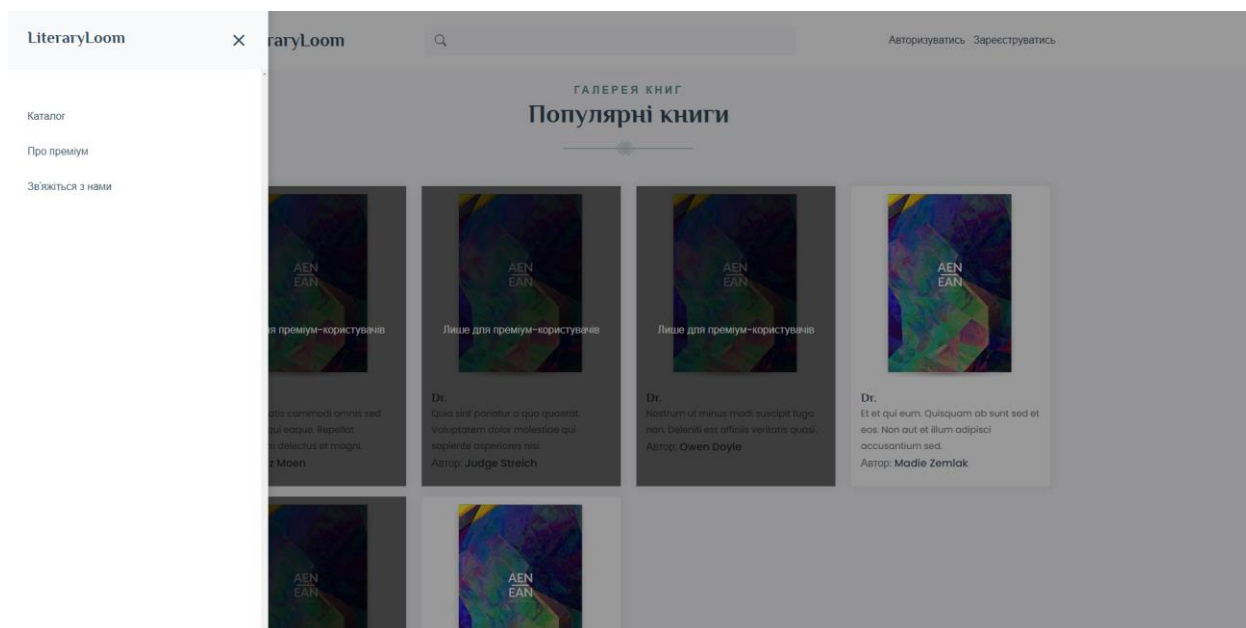


Рис. 3.2 Бокове меню

У верхньому правому кутку є кнопки для авторизації та реєстрації (рис. 3.1). Розглянемо процес реєстрації користувача (рис.3.3).

Рис. 3.3 Форма реєстрації користувача, як письменника

Рис. 3.3 Форма реєстрації користувача, як видавництва

Отже, як ми можемо побачити користувач має можливість зареєструватись як письменник (рис. 3.3) та як видавництво (рис. 3.4). Кожна форма реєстрації має свої поля для реєстрації, в залежності від кого користувач хоче зареєструватись. Вибираючи реєстрацію як письменника, користувачу обов'язково потрібно буде вибрати видавництво з перерахованих. Після натискання кнопки «Зареєструватись» перед користувачем з'явиться форма для

підтвердження номеру телефону (рис. 3.4), що дасть захист від ботів, оскільки після реєстрації потрібно буде підтвердити введений номер телефону.

Рис. 3.4 Підтвердження номеру телефону

Без підтвердження номеру телефону, користувач не матиме доступу до певних функцій, доти доки не підтвердить номер телефону. При спробі перейти, наприклад, на написання книги, перед користувачем буде повідомлення про те, що спочатку потрібно підтвердити номер та при потребі користувач матиме можливість змінити поточний номер телефону.

Рис. 3.5 Повідомлення про необхідність підтвердити телефон та форма змінення номеру телефону

Зареєстрований користувач має можливість увійти на платформу (рис.3.6) ввівши свій номер телефону та пароль. Також користувач має можливість вибрати забули пароль, де перед ним з'явиться форма – Змінити пароль (рис. 3.7).

Там користувач може вибрати через, що змінити пароль, через номер телефону або ж електронну пошту.

LiteraryLoom

АвторизуватисьЗареєструватись

Авторизуватись

Номер телефону

+38 (0__)-____-____

Пароль

Забули пароль

Авторизуватись

Інформація

Якщо ви забули пароль натисніть на кнопку нижче та ви отримаєте SMS-повідомлення з Вашим паролем!

Немає акаунту? Зареєструватись

Інформація про книгу?

Надішліть нам електронний лист на адресу literary_loom_support@gmail.com

Потрібна допомога?

Будь ласка, напишіть нам

LiteraryLoom© 2024 Всі права захищено.

Рис. 3.6 Форма авторизації

LiteraryLoom

АвторизуватисьЗареєструватись

ЗМІНИТИ ПАРОЛЬ

Телефон

+38 (0__)-____-____

Електронна пошта

Змінити пароль

Інформація про книгу?

Надішліть нам електронний лист на адресу literary_loom_support@gmail.com

Потрібна допомога?

Будь ласка, напишіть нам

LiteraryLoom© 2024 Всі права захищено.

Рис. 3.7 Форма зміни паролю

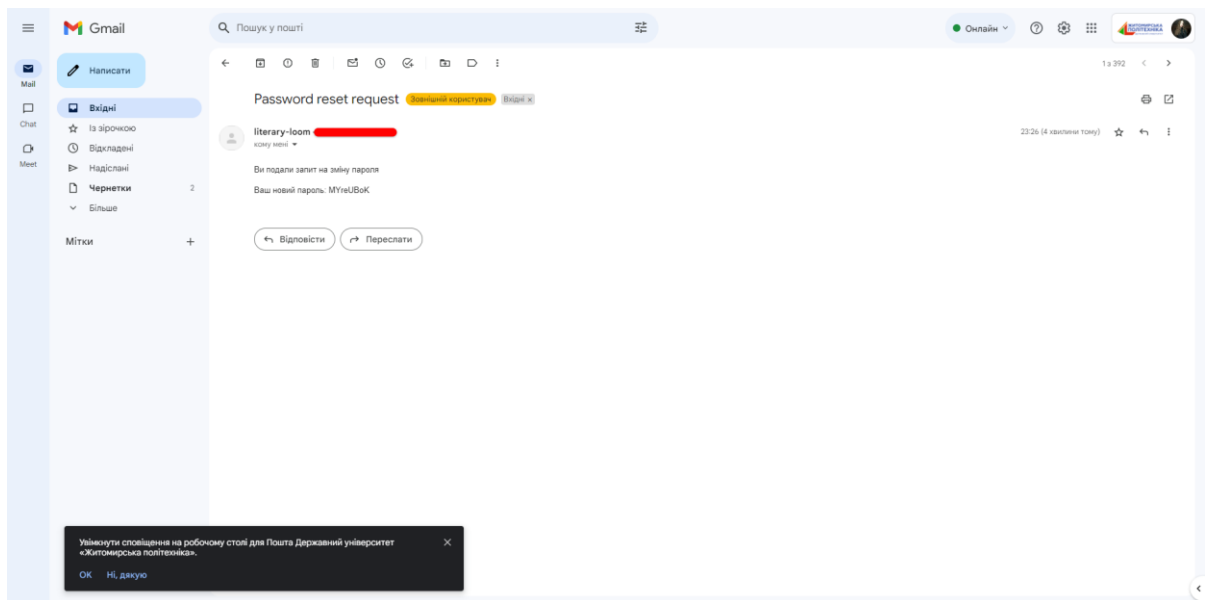


Рис. 3.8 Змінений пароль

Після того як користувач авторизувався він може відредагувати свій профіль (рис. 3.9). В профілі він може змінити ім'я, телефон, електронну пошту, видавництво та аватар. Також користувач може змінити колір профілю та змінити пароль.

Рис. 3.9 Профіль користувача

Також є публічний профіль користувача (рис.3.10). Через який можна буде подивитись інформацію про користувача, тобто книги, списки читання.

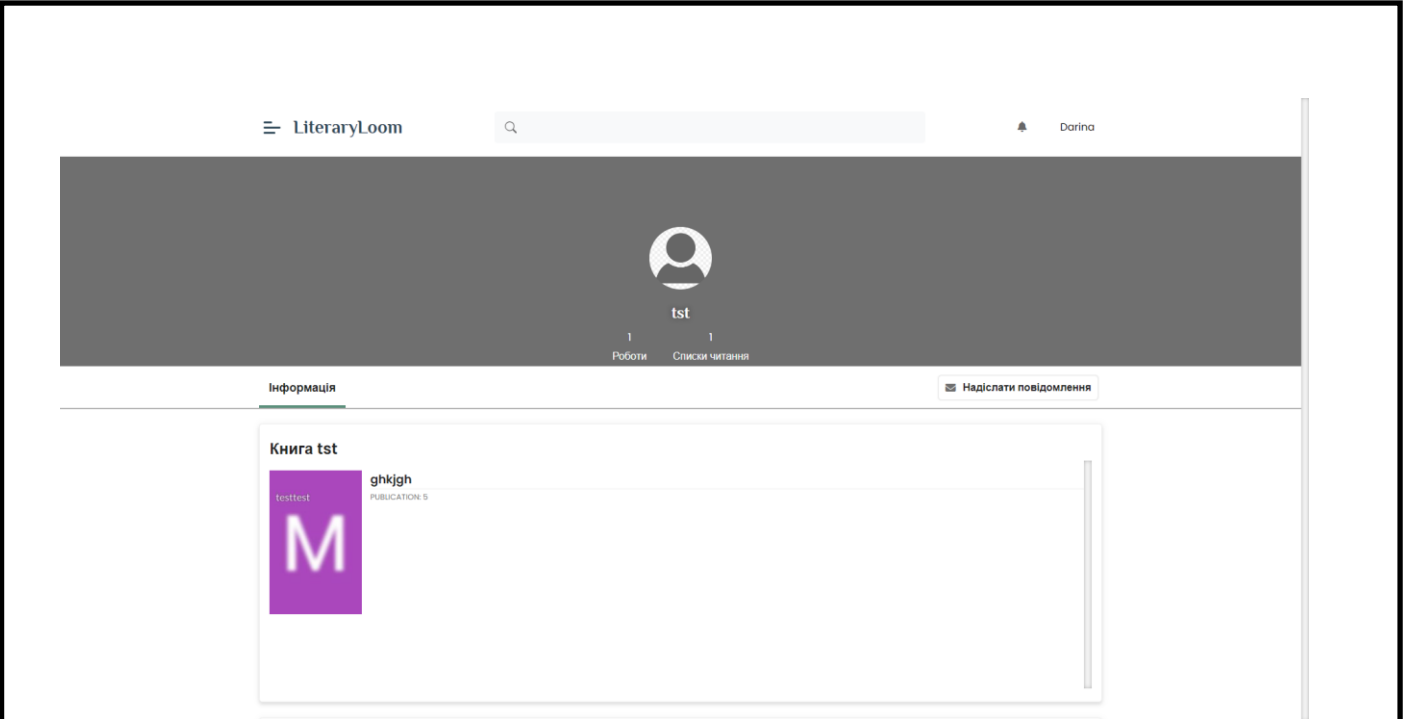


Рис. 3.10 Публічний профіль користувача

Також можна надіслати повідомлення (рис. 3.11). По замовчуванню при відкритті діалогу, буде надіслано повідомлення – «Привіт». Де в режимі реал тайму користувачі будуть отримувати повідомлення.

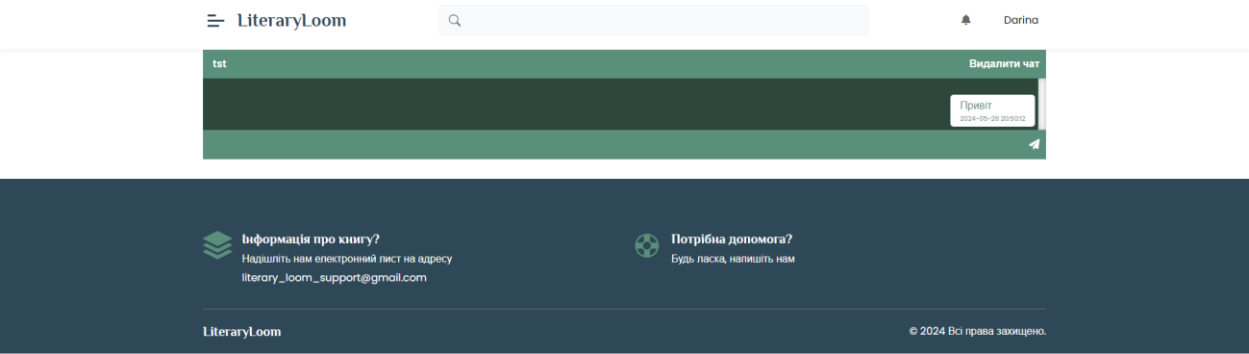


Рис. 3.11 Діалог з користувачем

Перед тим як писати книгу, видавництво повинно підтвердити, що користувач може писати книги від імені видавництва (рис. 3.12).

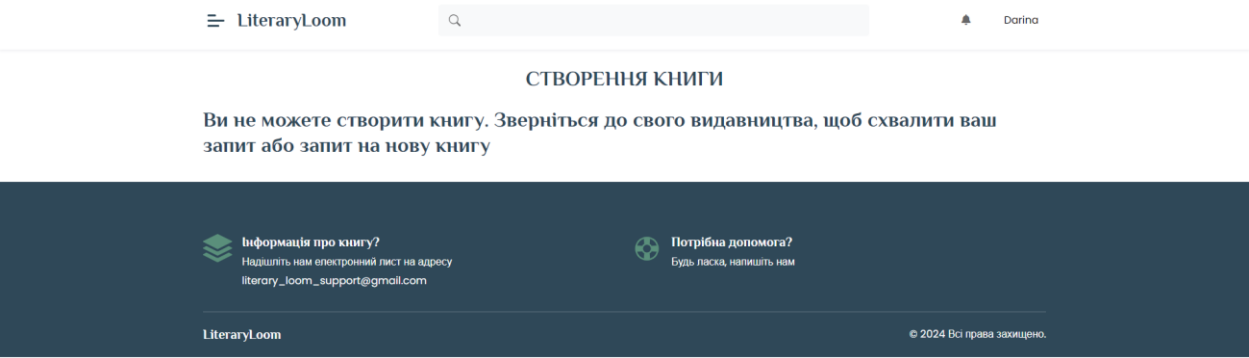


Рис 3.12 Створення книги без підтвердження

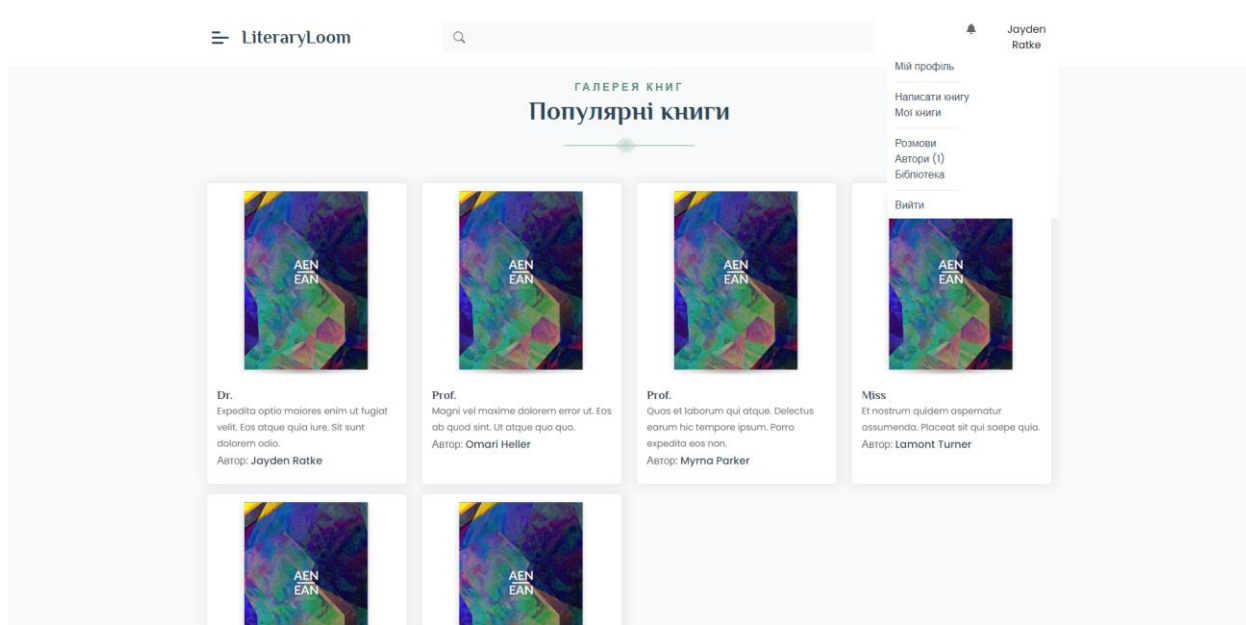


Рис. 3.13 Залогінений користувач, як видавництво

На рисунку 3.13 зображено залогіненого користувача як видавництво, ми можемо побачити пункт – Автори, тоді як перейшовши по ньому перед видавництвом буде запит на те, що хтось хоче доєднатись до видавництва (рис. 3.14). Видавництво може як прийняти так і відхилити користувача. Приймавши його, користувач перейде до прийнятих авторів та зможе писати книги від імені даного видаництва (рис. 3.15). Тоді як, відхиливши користувача він потрапить в список – Відхилено (рис. 3. 16). Також можна прийняти відхиленого користувача.

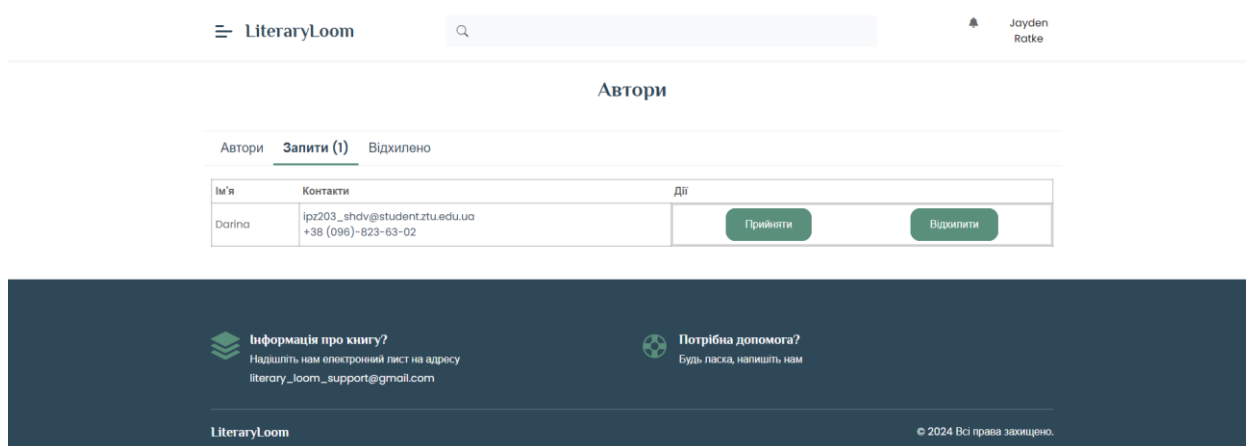


Рис. 3.14 Запити на приєднання до видавництва

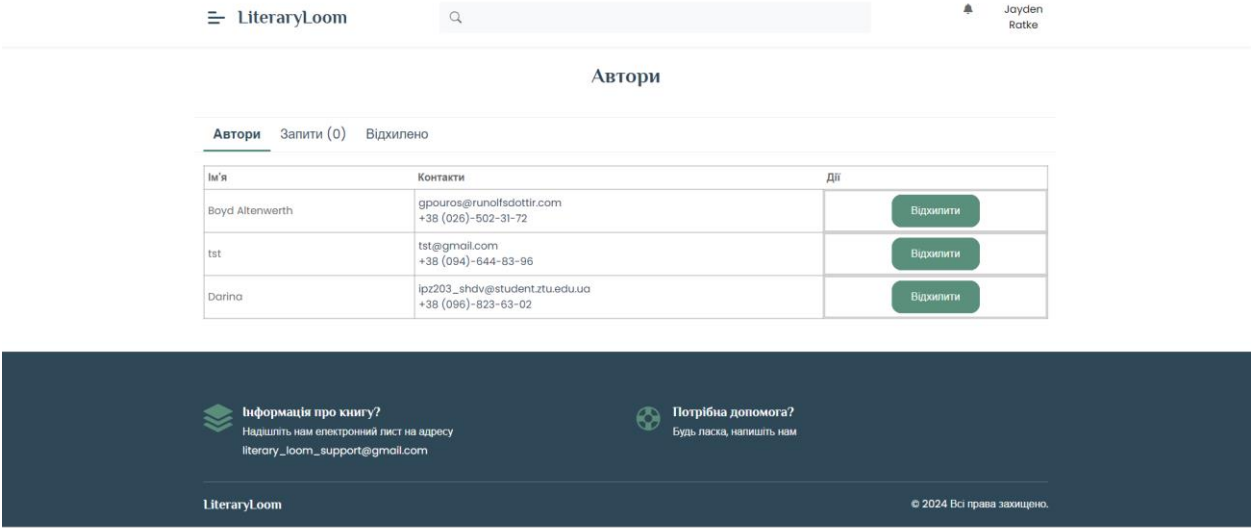


Рис. 3.15 Прийнятий користувач

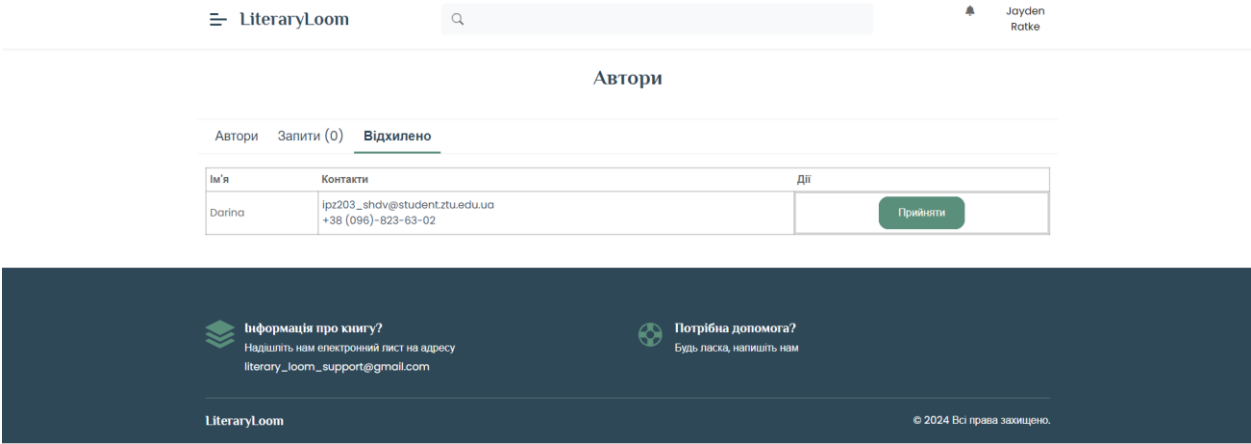


Рис. 3.16 Відхилений користувач

Після того як видавництво прийняло користувачу приходить сповіщення про те, що його прийняли або ж відмовили (3.17).

Рис. 3.17 Сповіщення від видавництва

Після того як видавництво прийняло заявку на приєднання, користувач має змогу написати свою книгу (рис. 3.18).

Рис. 3.18 Створення книги

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІПЗ	Арк.
		С. Кравченко				61
Змн.	Арк.	№ докум.	Підпис	Дата		

< Додати контент для Тестовий варіант

ОпублікуватиПрев'юВидалити копію

FileEditViewInsertFormatToolsTableHelp

↶↷Paragraph

BIL

☰☲☱☴☵☶☷

☰☲☱☴☵☶☷

Після опублікування книги, користувач бачить перед собою вміст книги для читання (рис. 3.20). Тут він може побачити скільки переглядів, оцінок книги, продовжити редагування (якщо він є автором цієї книги), проголосувати (оцінити книгу) (рис. 3.21), додати до бібліотеки – приватної, або ж створити нову бібліотеку (публічну).

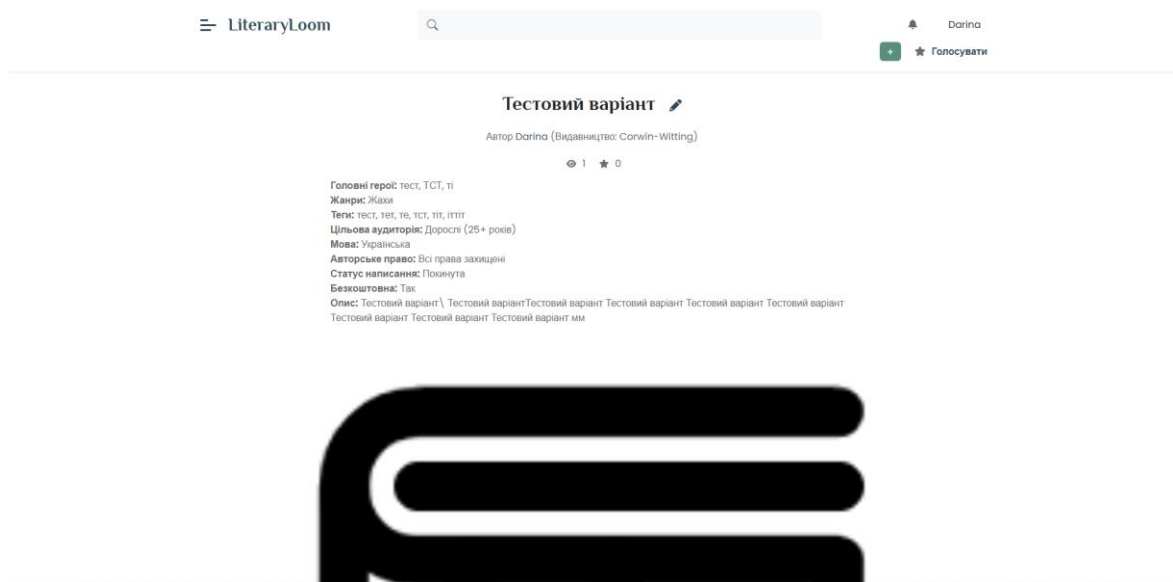


Рис. 3.20 Читання книги

		Д. Шевченко			ІІЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
		С. Кравченко				62
Змн.	Арк.	№ докум.	Підпис	Дата		

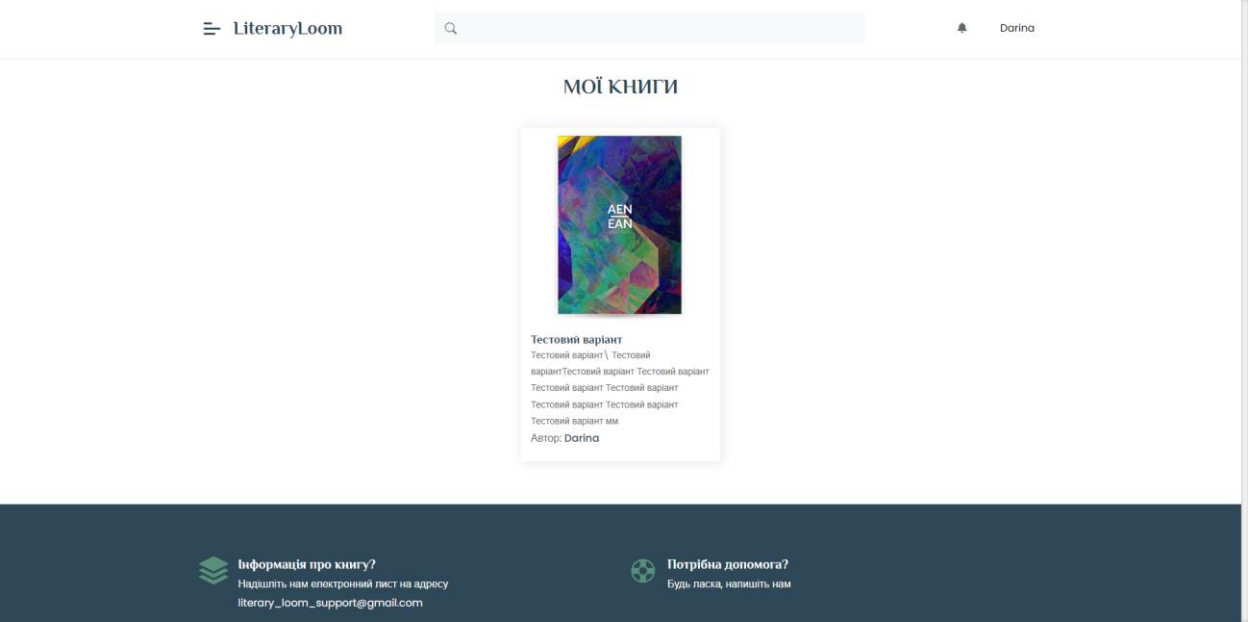


Рис. 3.23 Мої книги

Користувач має можливість переглянути початі чати та продовжити переписку (рис. 3.24).

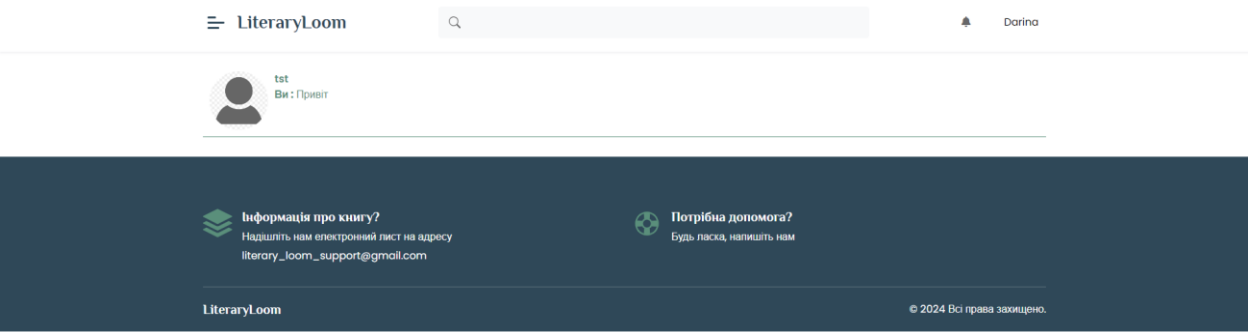


Рис. 3.24 Чати

Отже перейдемо до каталогу книг (рис. 3.25). Перейшовши в каталог користувач має можливість відфільтрувати книги за категоріями. Також ми можемо побачити, що є книги, які є лише для преміум користувачів, тому доступу до них звичайний користувач не матиме.

Відфільтруємо книги за певними категоріями (рис. 3.26).

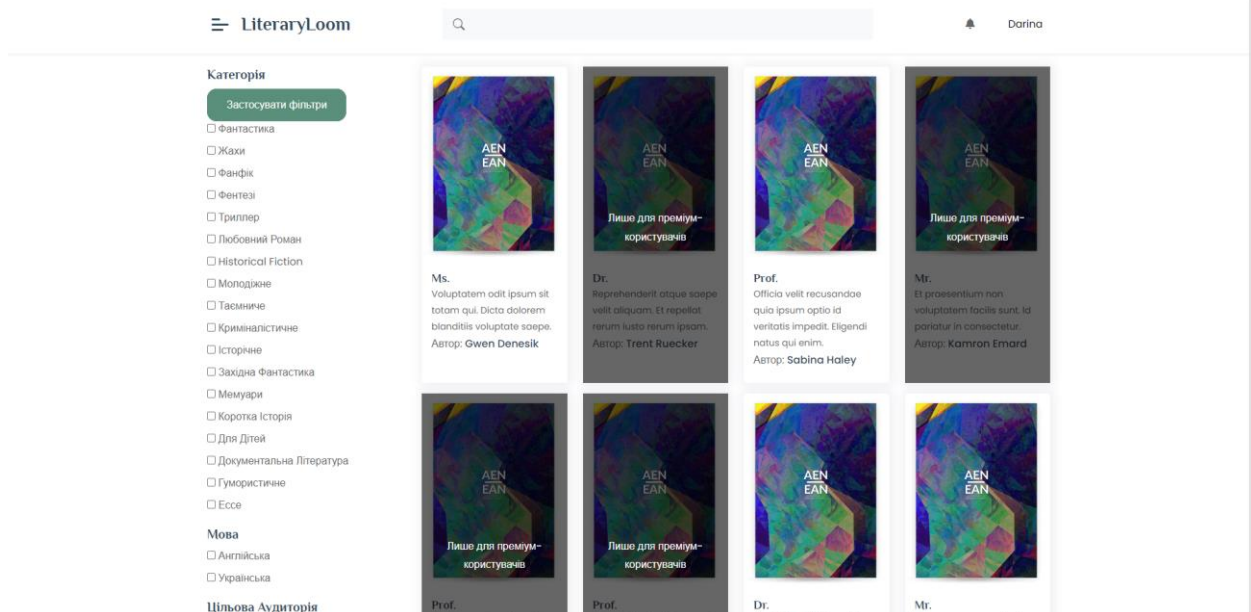


Рис. 3.25 Каталог книг

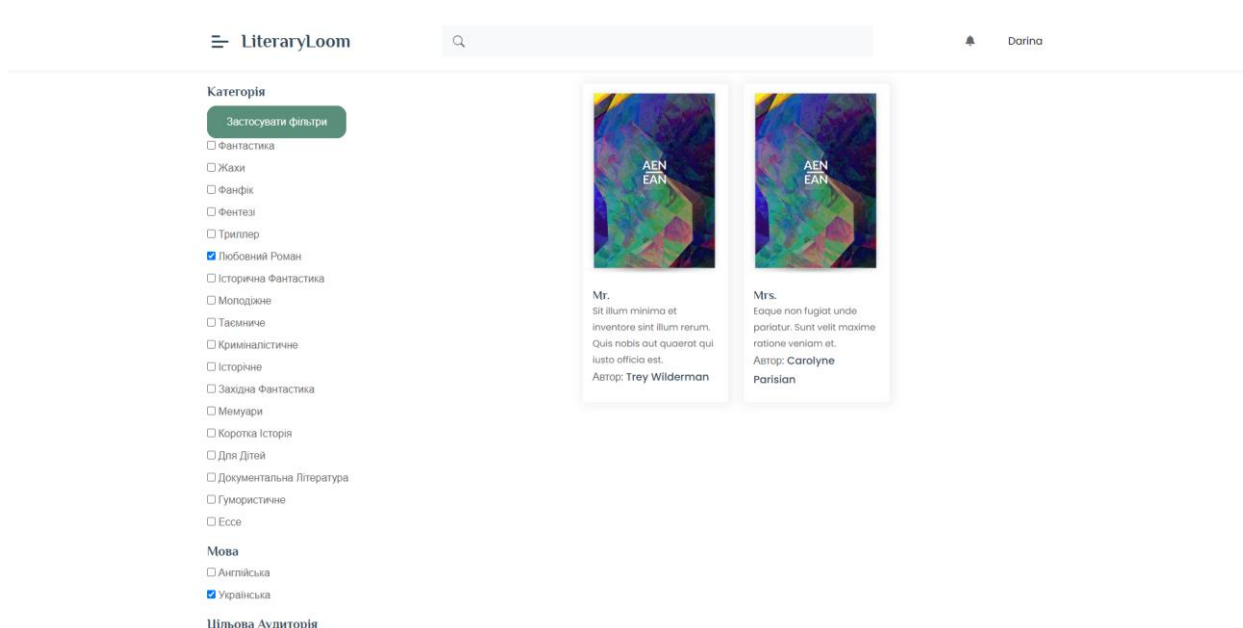


Рис. 3.26 Відфільтровані книги

Зареєстрований користувач має можливість робити пошук книг та авторів (рис. 3.27). Пошук відбувається лише після введення трьох символів.

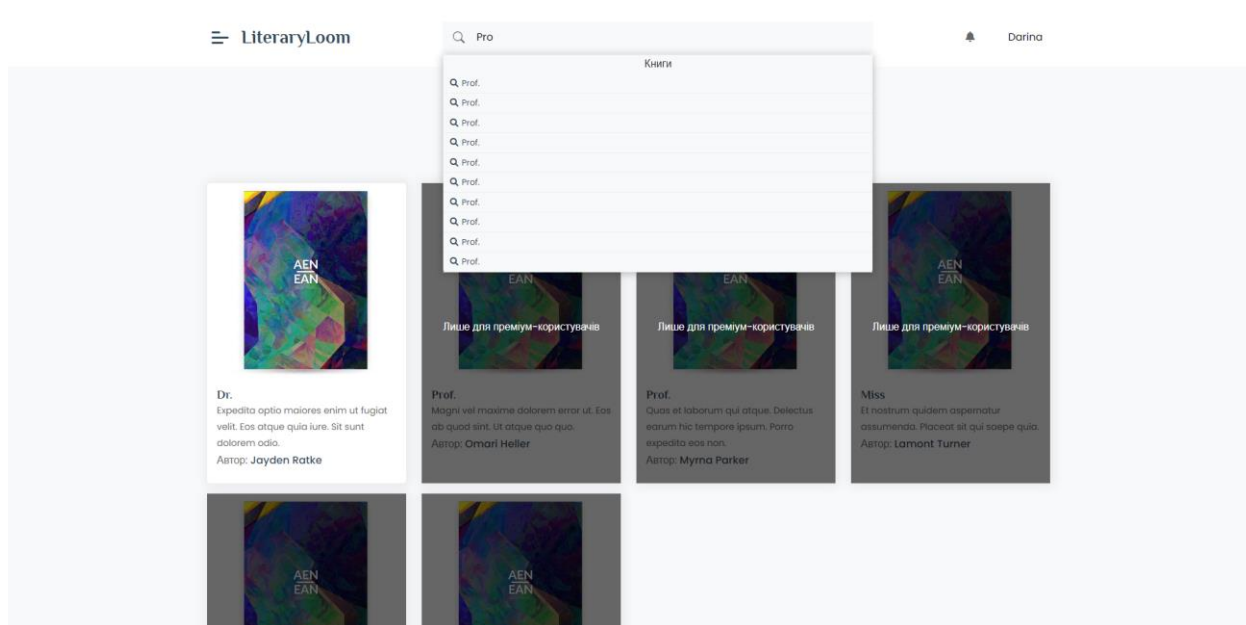


Рис. 3.27 Пошук книг та авторів

Користувач може купити преміум для доступу до оригінального платного контенту (рис. 3.28).

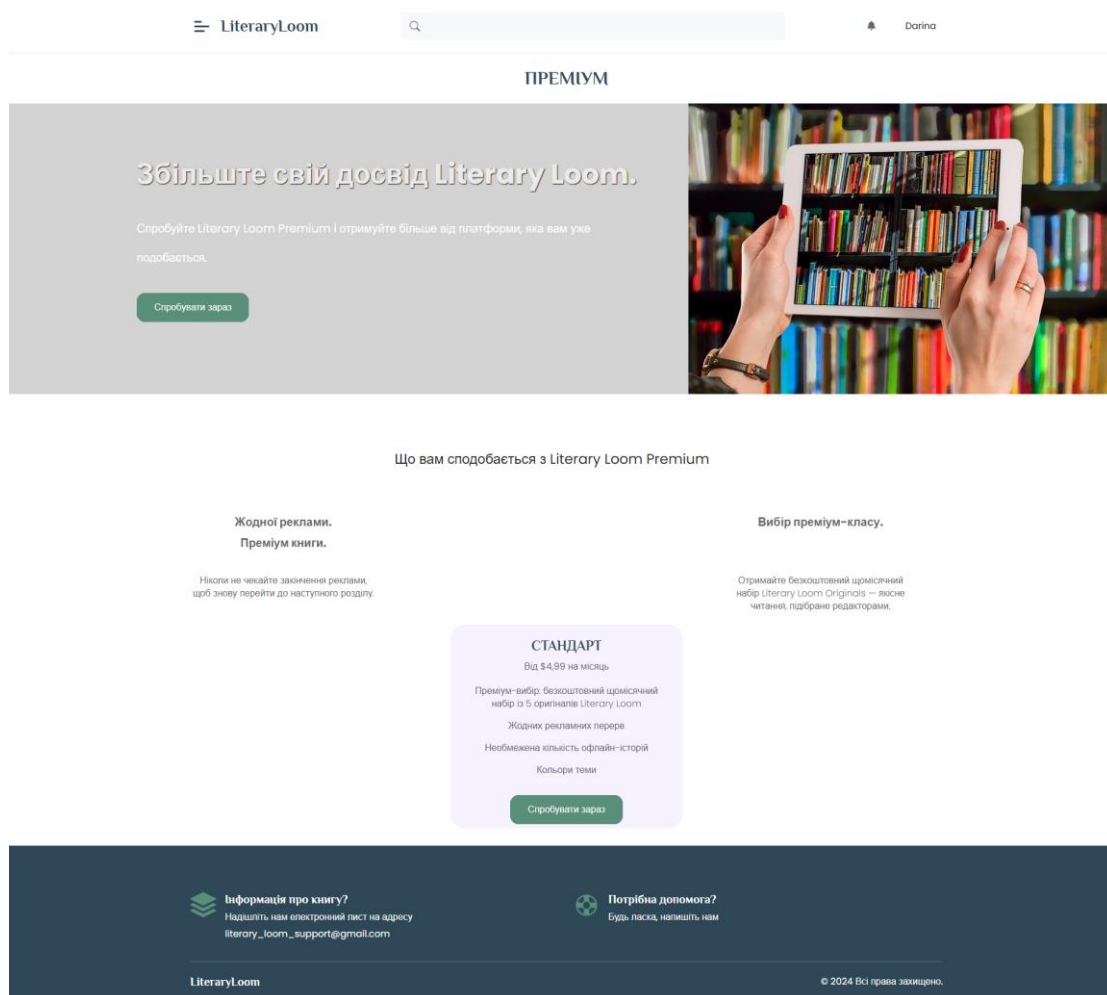


Рис. 3.28 Преміум

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІПЗ	Арк.
		С. Кравченко				66
Змн.	Арк.	№ докум.	Підпис	Дата		

Натиснувши кнопку Спробувати зараз , користувача перенаправляє на оплату плану (рис. 3.29).

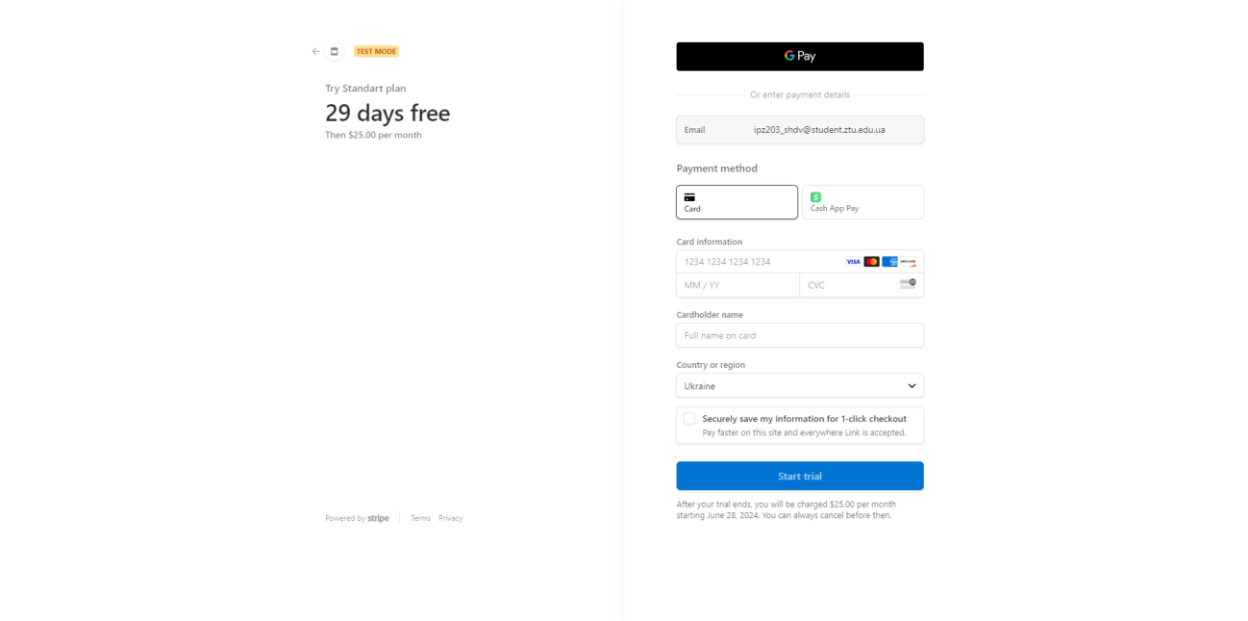


Рис. 3.29 Оплата плану

Після успішної оплати, користувач бачить повідомлення, що оплата пройшла успішно (рис. 3.30).

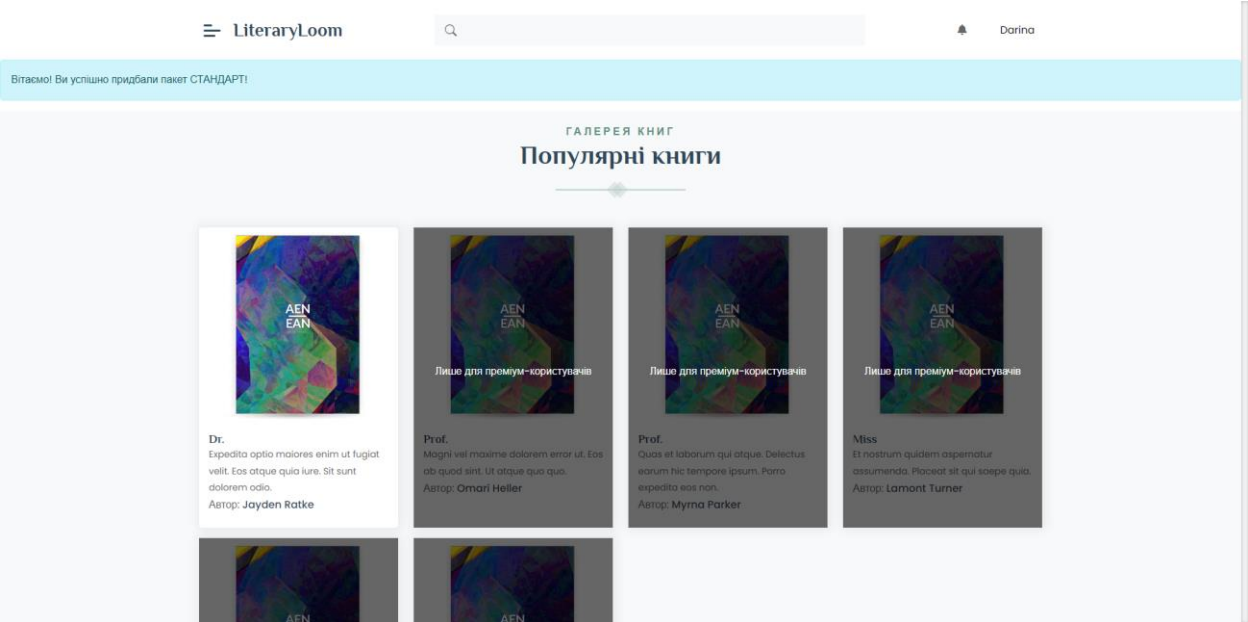


Рис. 3.30 Успішна оплата

Тепер відкривши бокове меню, користувач бачить пункт – Преміум книги (рис. 3.31).

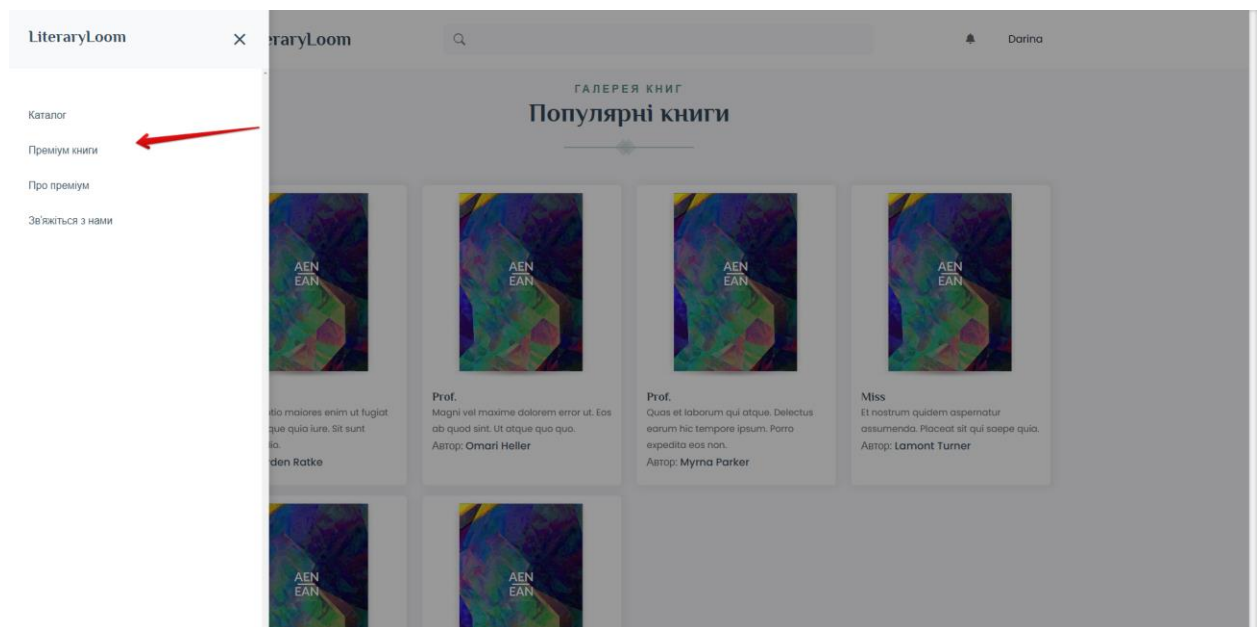


Рис. 3.31 Доданий пункт преміум книги

Перейшовши на цю вкладку перед користувачем будуть відображені всі преміум книги (рис. 3.32).

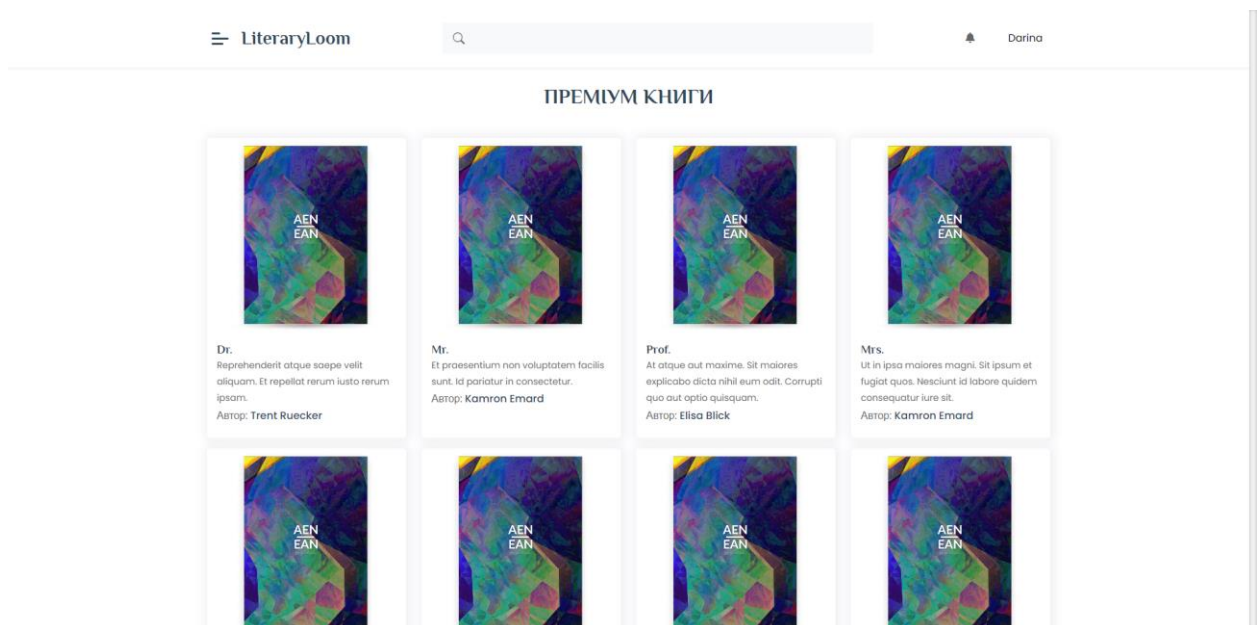


Рис. 3.33 Преміум книги

Користувач може додавати книги до своєї бібліотеки (рис. 3.34). При створенні користувача, в нього відразу є приватна бібліотека (рис. 3.35), куди він також може додати книги, але це не буде відображено в нього на сторінці.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				68
Змн.	Арк.	№ докум.	Підпис	Дата		

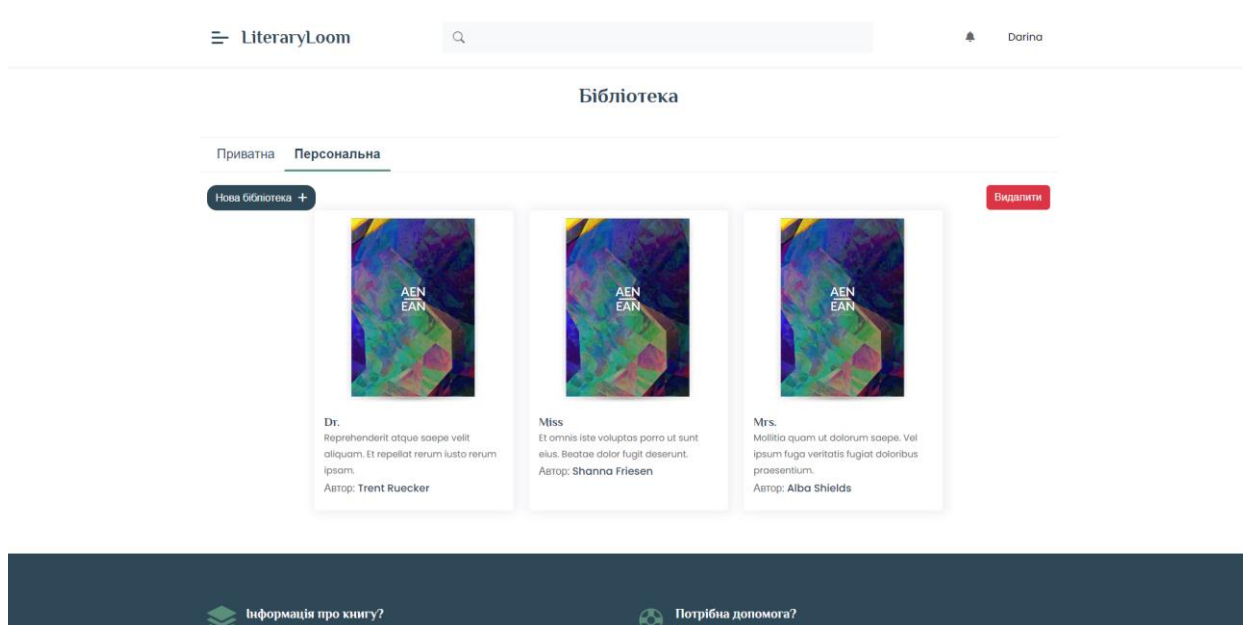


Рис. 3.34 Створена бібліотека та додані книги.

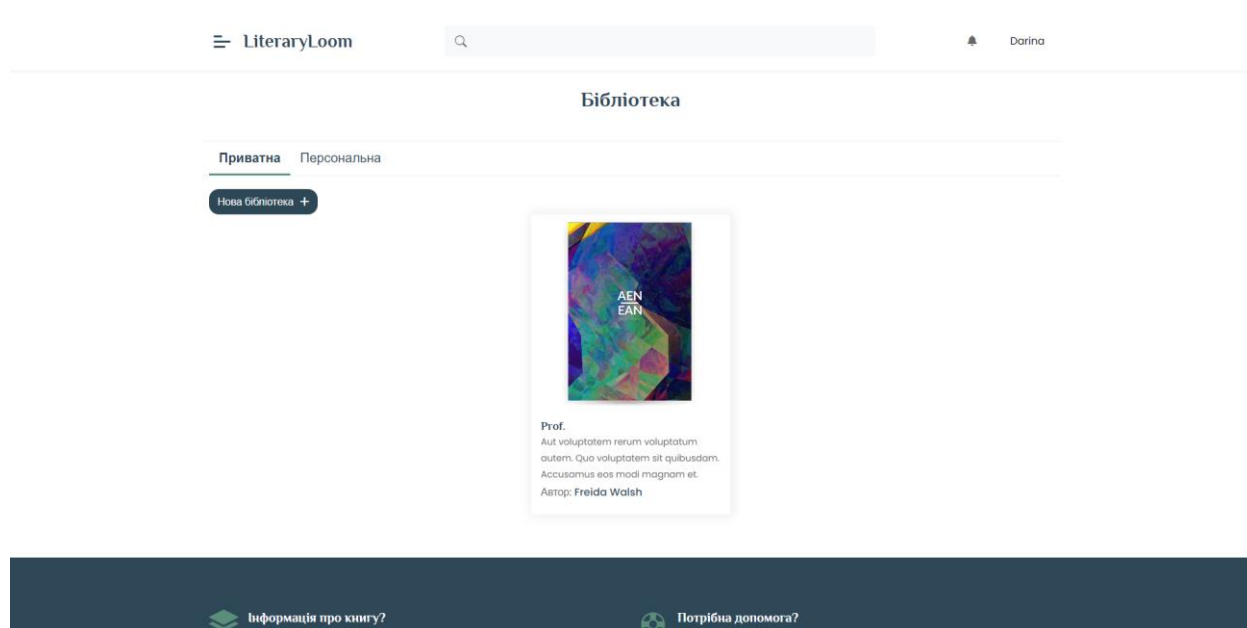


Рис. 3.35 Приватна бібліотека

Користувач має можливість зв'язатись з центром підтримки (рис. 3.36).

Рис. 3. 36 Форма для зв'язку з центром підтримки

Сайт адаптовано під мобільну верстку (рис. 3.37). Всі функції працюють так само.



Рис. 3.37 Мобільна версія головної сторінки

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІПЗ	Арк.
		С. Кравченко				70
Змн.	Арк.	№ докум.	Підпис	Дата		

Рис. 3.38 Мобільна версія зв'яжіться з нами

Перейдемо до адміністративної панелі. Перед адміном є форма логіну до адміністративної панелі (рис. 3.39).

Рис. 3.39 Вхід в адмін панель

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				71
Змн.	Арк.	№ докум.	Підпис	Дата		

Зайшовши в адмін панель користувач бачить перед собою Інфопанель, де відображено статистику (рис. 3.40).

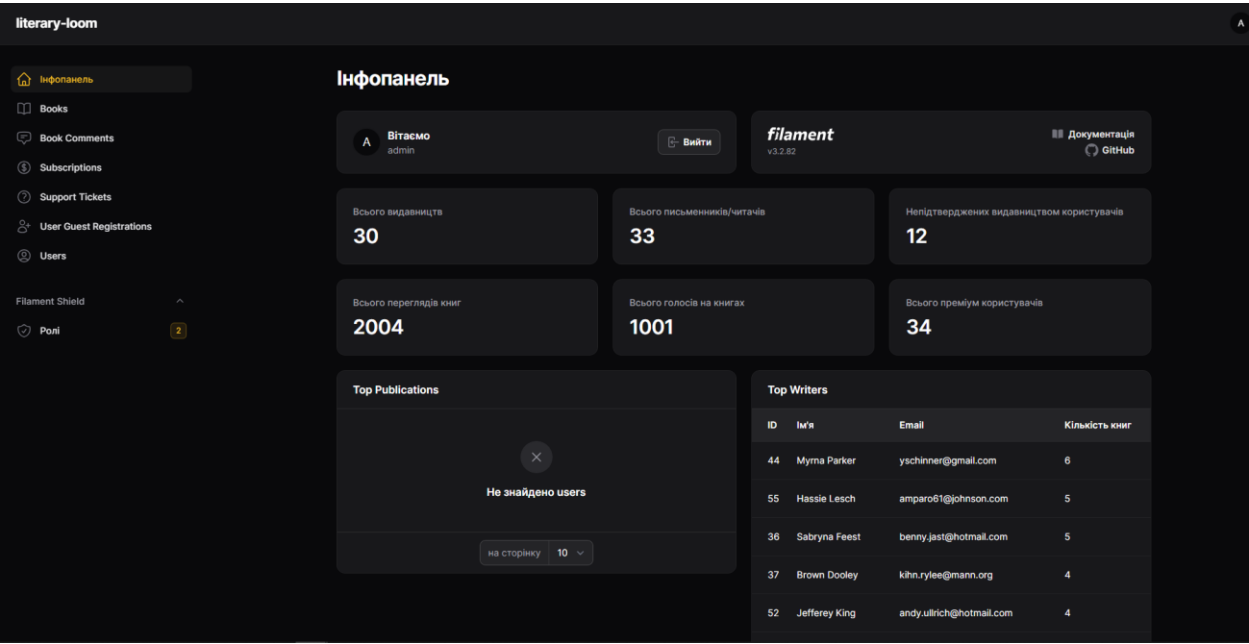


Рис. 3.40 Інфопанель

Також бек офіс може контролювати книги, та змінювати їх при потребі (рис. 3.41).

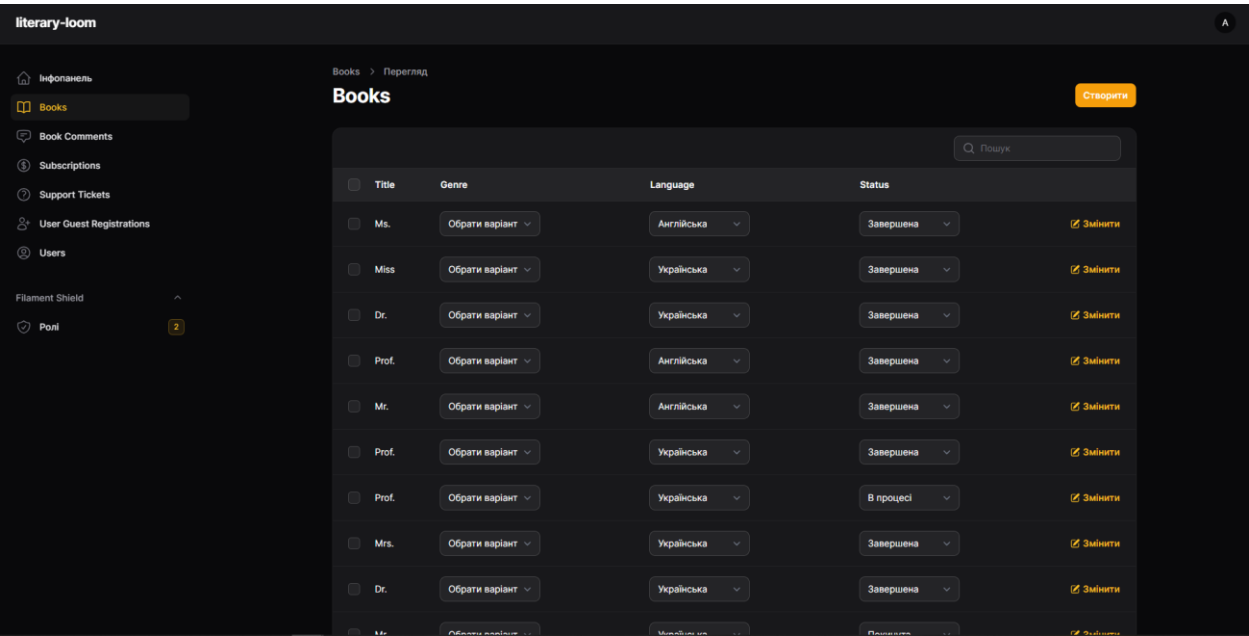


Рис. 3.41 Книги

Підтвердження коментарів відбувається на вкладці Коментарі книги (Books Comments) (рис. 3.42). Тут бек офіс може промодерувати коментар, відхилити, по замовчуванню коментар висить в режимі очікування від тоді як він відправиться.

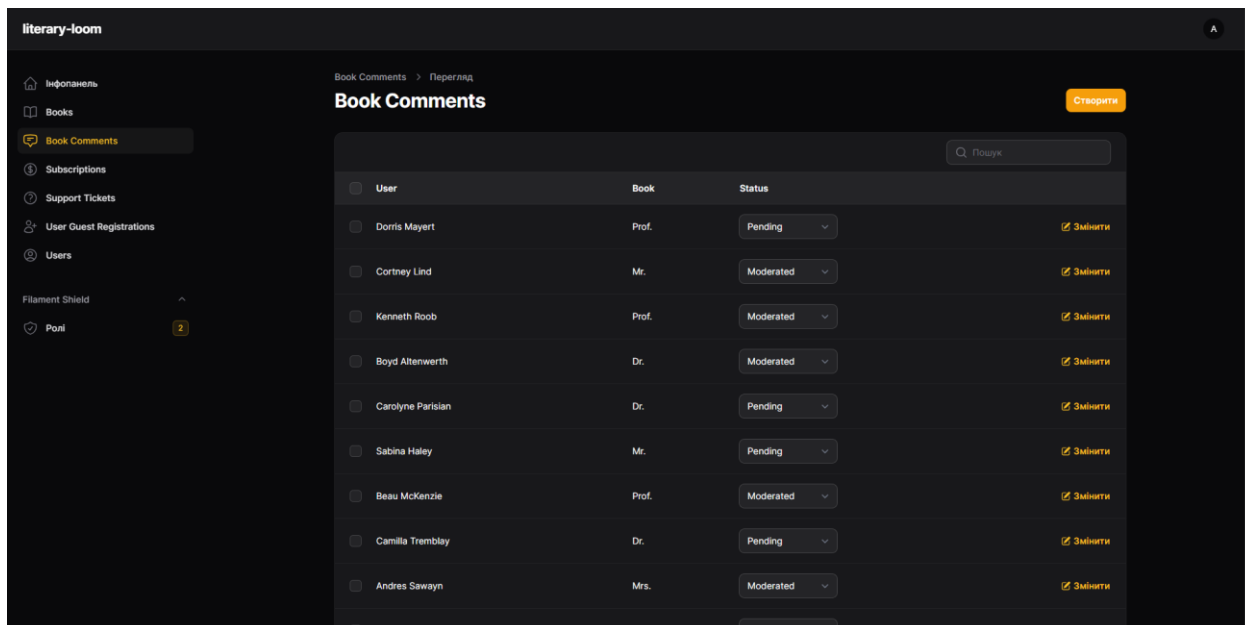


Рис. 3.42 Коментарі книги

В адмін панелі також можна переглянути підписки на преміум аккаунти (рис. 3.43).

Перейдемо до листів які приходять до центру підтримки (рис. 3.44). Відкривши один з них, користувач з центру підтримки може побачити вміст кожного листа, тобто від кого він, його електронну пошту, саме повідомлення та змінити статус листа (рис. 3.45). Також видалити лист, якщо він містить некоректний вміст. Саме листування буде відбуватись через пошту.

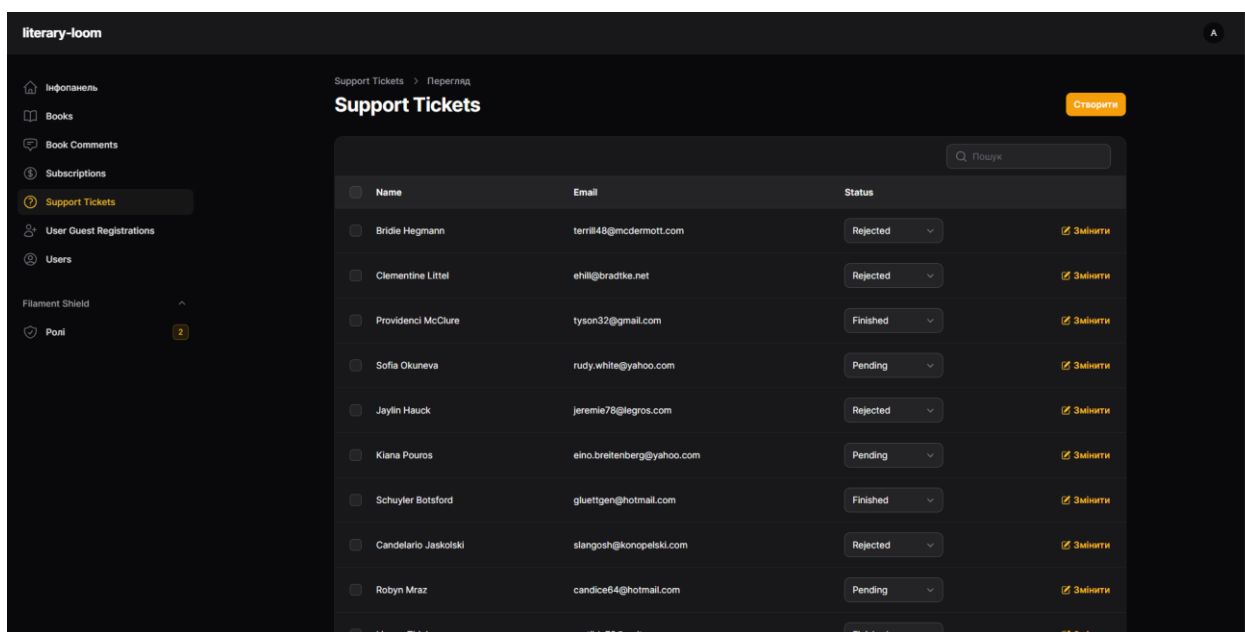


Рис. 3.44 Запити в центр підтримки

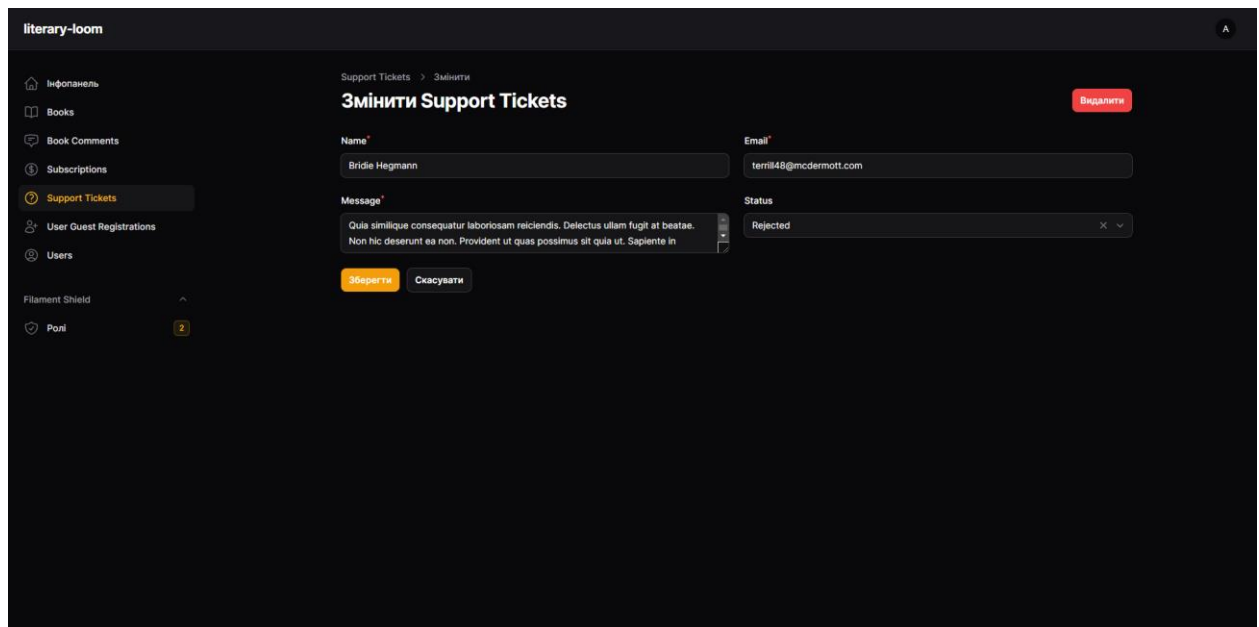


Рис. 3.45 Вміст листа до центру підтримки

На сайті присутня інформація про гостей, тобто коли користувач починав реєстрацію, але так би мовити передумав (рис. 3.46).

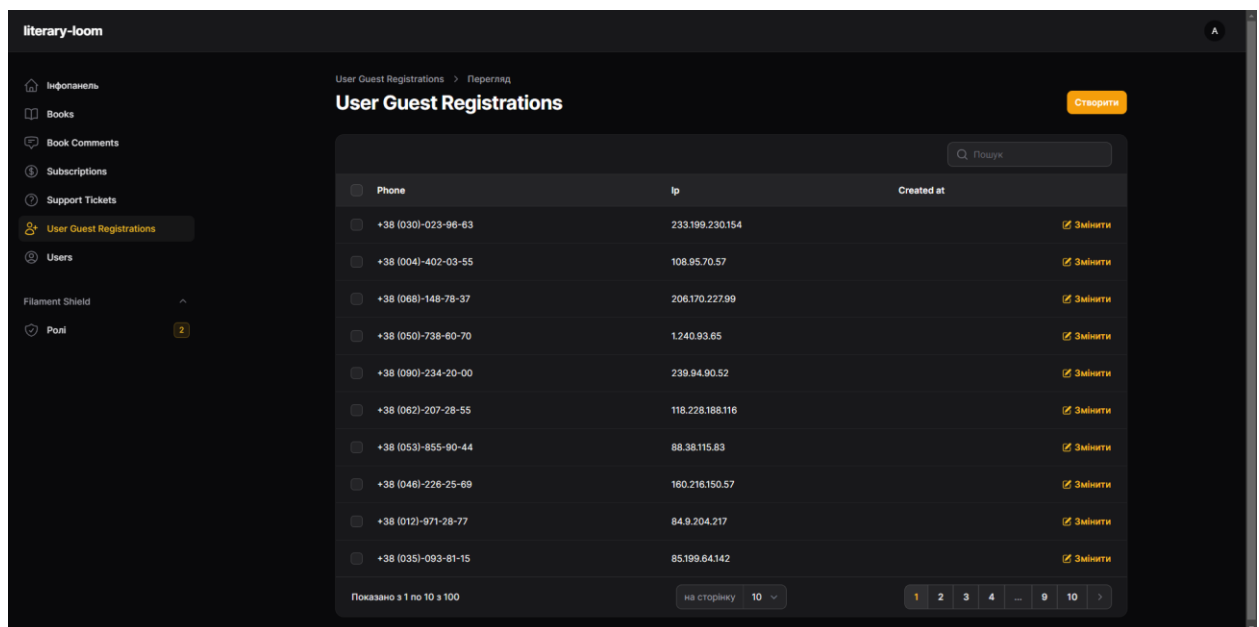


Рис. 3.46 Інформація про гостей

Перейдемо до користувачів (рис. 3.47), бек офіс може надавати ролі для користувачів, переглядати їх інформацію (рис. 3.48). Також на рисунку 3.49 можна переглянути наявні ролі, наразі це: Модератор, Супер адмін та Центр підтримки.

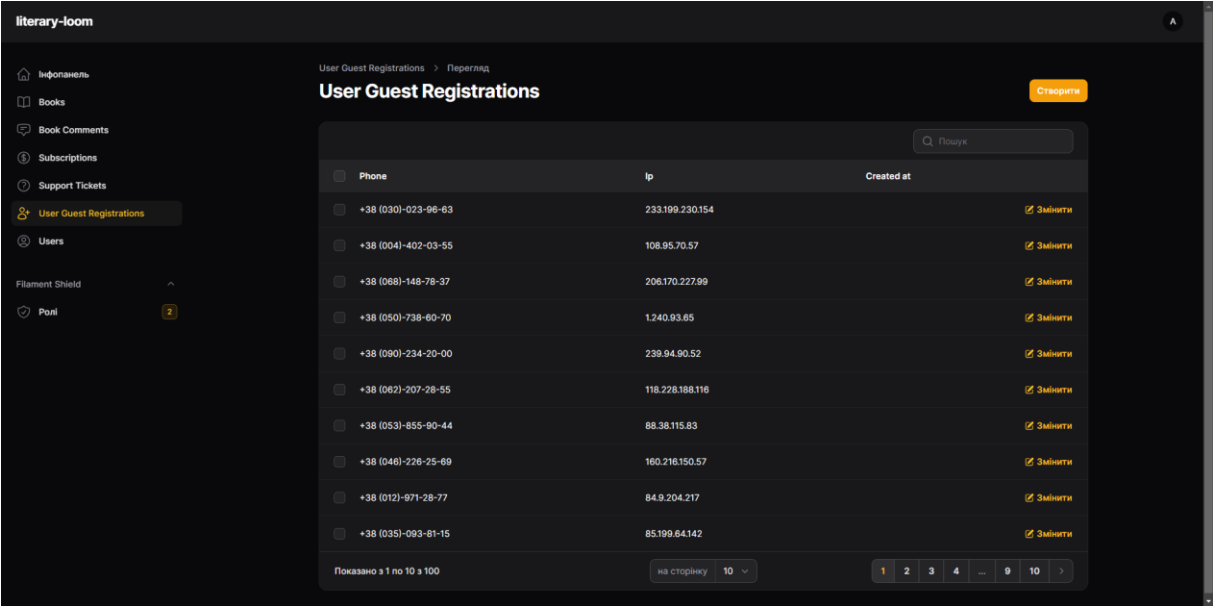


Рис. 3.47 Зареєстровані користувачі

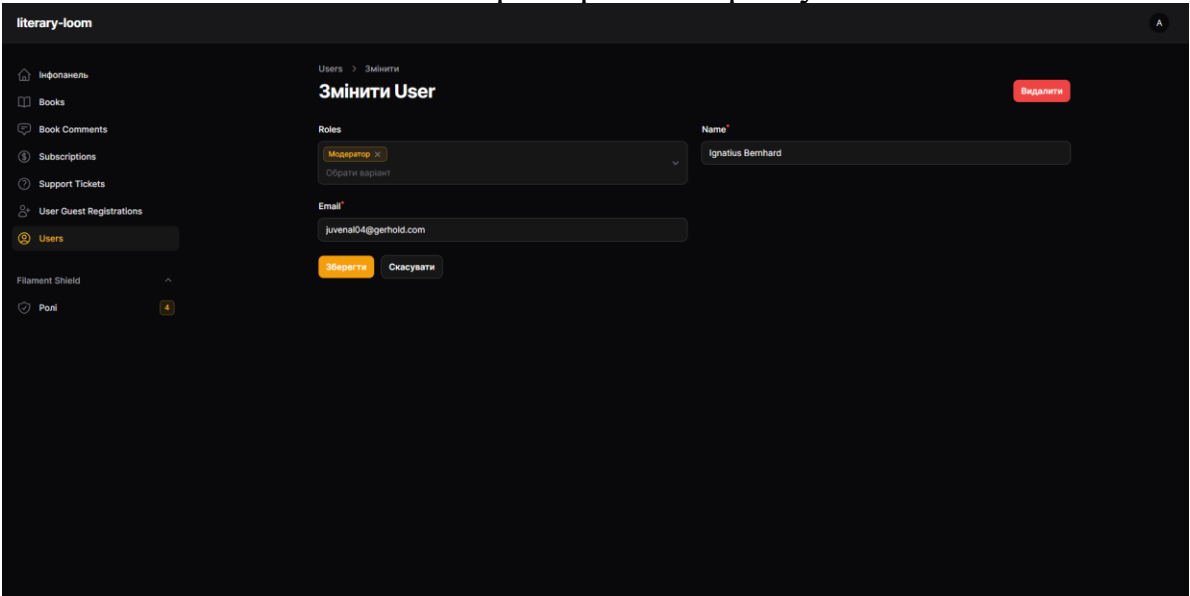


Рис. 3.48 Інформація про користувача, зміна ролі

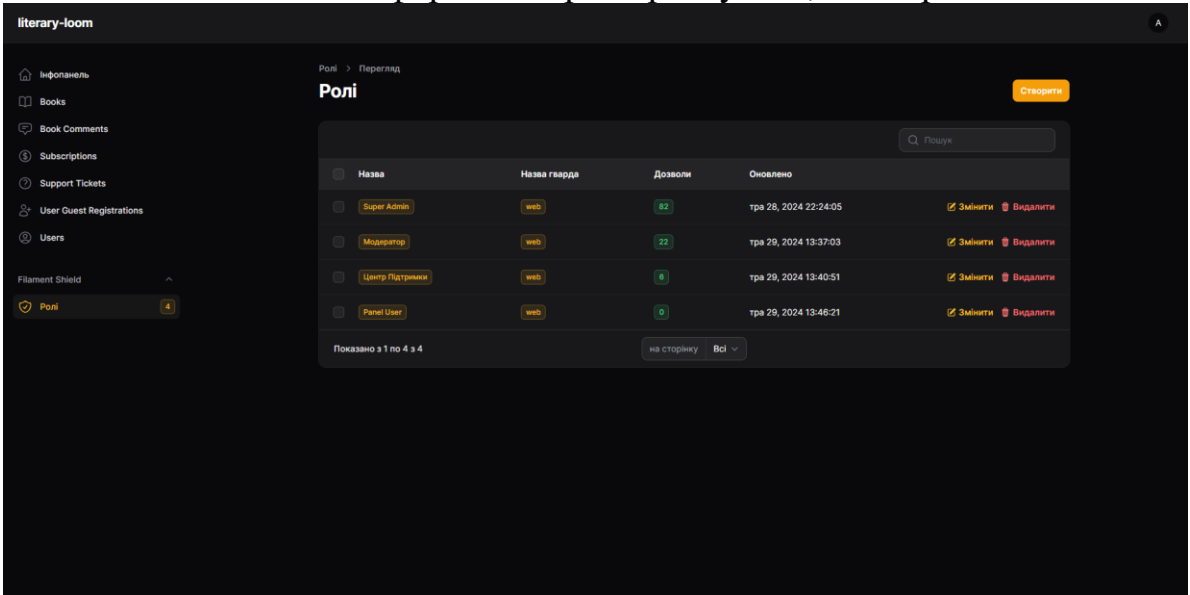


Рис. 3.49 Наявні ролі користувачів

3.3 Тестування роботи додатку

Тестування додатку – це важливий етап у розробці, завдяки ньому ми можемо переконатись, що додаток працює коректно та відповідає встановленим вимогам. Завдяки тестуванню вдається виявити проблеми перед тим як його завантажувати в простори Інтернету.

Також це допоможе зробити веб-додаток якісним, що гарантує те, що користувачі залишаться саме на цьому додатку та продовжать працювати саме з цим, а не виберуть якийсь інший.

Розробляючи платформу для письменників та читачів, потрібно було врахувати, що платформа має забезпечувати зручне середовище для створення та читання книг. Автори повинні мати можливість створювати, редагувати, вибирати видавництво, оцінювати роботи та залишати коментарі.

Тестування платформи включало перевірку таких аспектів:

- Функціональність: перевірка чи всі функції працюють коректно.
- Інтерфейс користувача: перевірка на зручний та інтуїтивно зрозумілий інтерфейс.
- Безпека: перевірка чи шифруються конфіденційні дані користувачів та виявлення можливих вразливостей (CSRF)

Тестування проводилось під час розробки та після завершення розробки. Після виправлення додаток знову було перетестовано.

Отже, які дефекти було знайдено зі сторони користувача:

1. Некоректна поведінка зміни паролю: користувач в будь якому випадку отримує помилку і може змінити пароль тільки за допомогою СМС-повідомлення.
2. Відсутня інформація про оплати в адміністративній панелі.
3. Додавання ролей до користувача: ніяк не впливало на пункти меню адмінки які він бачить, тобто якщо є користувач модератор, то він має такі самі дозволи як і адміністратор.
4. Підложка для преміальних книг наїжджає на фіксований хедер.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				76
Змн.	Арк.	№ докум.	Підпис	Дата		

5. Після перекладу сайту на українську мову посилання на авторизацію наїжджає на посилання на реєстрацію.
6. Відсутній зв'язок між книгою та видавництво від якої вона написана, що призводить до того, що на сторінці читання книги замість назви видавництва виводиться її ідентифікатор.
7. На десктопній версії посилання на авторизацію та реєстрацію, яке має показуватись тільки на мобільному, стає прозорим, замість того щоб ховатись повністю, що спричинило, що посилання в головному меню заїжджають за межі екрану.
8. Відсутнє посилання на профіль співрозмовника в чаті.
9. Автор книги не може видалити власну книгу.
10. Співрозмовник в чаті не може видалити чат.
11. Користувачі не індексуються в ElasticSearch.
12. Список жанрів книги некоректний.
13. На мобільній версії додатку не можливо перейти на головну сторінку натиснувши на логотип.
14. В бібліотеці не можна видалити одну книгу, лише бібліотеку.

Висновки до третього розділу

В даному розділі було наведено порядок налаштування системи. Розглянуто роботу з веб-додатком та з адміністративною панеллю. Приведено роботу функціоналу додатку та протестовано коректність роботи сайту.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				77
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи було реалізовано онлайн платформу для письменників та читачів.

На початку виконання було поставлено задачу, визначено основні етапи для реалізації поставленого завдання. Також було проведено аналіз аналогічних додатків для порівняння їхніх переваг та недоліків, та було порівняно їх з власним проектом. Під час пошуку аналогічних додатків вдалось знайти три, які відповідали вимогам по проекту та були схожі більше ніж на 50%. Були визначені технології та засоби для реалізації даного проекту, тому для реалізації було обрано мову програмування PHP, фреймворк Laravel, Docker, Docker Compose, PostgreSQL, Elasticsearch, Redis, Rabbit MQ, Scaleway інстанс та інтегровано платіжну систему Stripe.

Визначено вимоги користувачів задля правильного створення варіантів використання платформи. Також було сформовано функціональні та нефункціональні вимоги до платформи. Спроектовано базу даних, на основі аспектів, за допомогою яких працює дана платформа. База даних складається з 24 таблиць, їх перелік представлено в підпункті 2.2 розділу 2.

Розроблено алгоритми функціонування платформи та реалізовано веб-додаток. Створено дизайн платформи та її прототип. Описано роботу платформи поєднавши всі компоненти системи та описано їх. Розгорнуто додаток на хмарному середовищі. Створено інфраструктуру при якій можна працювати з багатьма БД різних типів. Запроваджено синхронізацію даних між різними БД. Імплементовано інтеграцію з платіжним сервісом Stripe та СМС-шлюзом SMS-Club. Впроваджено систему черг для відправки повідомлень по email-у.

Проведено тестування платформи. Проведено такі типи тестування: функціональне, кросбраузерне та безпеки. Було знайдено недоліки в роботі деяких функціональностей платформи та виправлено їх. Також було знайдено недоліки в UI/UX та виправлено їх.

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				78
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Robin Nixon. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. O'Reilly. 2021. 832с.
2. Роберт Мартін, Чиста архітектура. Фабула. 2020. 368с.
3. Роберт Мартін, Чистий код. Фабула. 2020. 448с.
4. Мет Стаффер, Laravel 2 видання. O'Reilly. 2019. 512с.
5. Clinton Gormley, Zachary Tong, Elasticsearch: The Definitive Guide. O'Reilly. 2019. 724с.
6. Alvaro Videla, Jason J.W. Williams, RabbitMQ in Action: Distributed Messaging for Everyone. Manning Publications. 2020. 374с.
7. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional. 2020. 395с.
8. Yung-Chun Chang, Practical Elasticsearch SQL: Query Elasticsearch Using SQL and Test Your SQL Skills with 300+ Examples. Independently Published. 2020. 300с.
9. Eric Mann, Adam Culpepper, PHP Cookbook: Solutions & Examples for PHP Programmers. O'Reilly. 2020. 792с.
10. David Dossot, RabbitMQ Essentials. Packt. 2019. 200с.
11. Luis Atencio, Clean Code in PHP: Expert Tips and Techniques. O'Reilly. 2021. 240с.
12. Doug Bierer, PHP 8 Programming Tips, Tricks and Best Practices. Packt 2021. 528с.
13. Josh Lockhart, Modern PHP: New Features and Good Practices. O'Reilly. 2020. 270с.
14. Inkitt [Електронний ресурс] – Режим доступу до ресурсу: <https://www.inkitt.com/>
15. Wattpad [Електронний ресурс] – Режим доступу до ресурсу: <https://www.wattpad.com/>

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				79
Змн.	Арк.	№ докум.	Підпис	Дата		

16. FanFiction [Електронний ресурс] – Режим доступу до ресурсу:
<https://m.fanfiction.net/>
17. Laravel [Електронний ресурс] - Режим доступу до ресурсу:
<https://laravel.com/docs/10.x>
18. PHP 8 [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.php.net/manual/en/>
19. PostgreSQL [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.postgresql.org/docs/16/index.html>
20. MySQL [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.mysql.com/>
21. Redis [Електронний ресурс] - Режим доступу до ресурсу:
<https://redis.io/>
22. Aerospike [Електронний ресурс] - Режим доступу до ресурсу:
<https://aerospike.com/>
23. Memcached [Електронний ресурс] - Режим доступу до ресурсу:
https://aws.amazon.com/memcached/?nc1=h_ls
24. RabbitMQ [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.rabbitmq.com/>
25. Apache Kafka [Електронний ресурс] - Режим доступу до ресурсу:
<https://kafka.apache.org/>
26. RabbitMQ vs Apache Kafka [Електронний ресурс] - Режим доступу до ресурсу:
https://aws.amazon.com/compare/the-difference-between-rabbitmq-and-kafka/?nc1=h_ls
27. PostgreSQL vs MySQL [Електронний ресурс] - Режим доступу до ресурсу:
https://aws.amazon.com/compare/the-difference-between-mysql-vs-postgresql/?nc1=h_ls
28. Redis vs Aerospike vs Memcached [Електронний ресурс] - Режим доступу до ресурсу:
[https://anthonyaje.github.io/file/An empirical evaluation of Memcached Redis and Aerospike kvstore Anthony Eswar.pdf](https://anthonyaje.github.io/file/An%20empirical%20evaluation%20of%20Memcached%20Redis%20and%20Aerospike%20kvstore%20Anthony%20Eswar.pdf)

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
		С. Кравченко				80
Змн.	Арк.	№ докум.	Підпис	Дата		

29. Redis vs Aerospike vs Memcached [Електронний ресурс] - Режим доступу до ресурсу: <https://db-engines.com/en/system/Aerospike%3BMemcached%3BRedis>

30. Elasticsearch [Електронний ресурс] - Режим доступу до ресурсу: <https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html>

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
		С. Кравченко				81
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Д. Шевченко			ІПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
		С. Кравченко				82
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду SMSClubService:

<?php

declare(strict_types=1);

namespace App\Services\SMS;

use GuzzleHttp\Client;

use Illuminate\Http\JsonResponse;

class SMSClubService implements SMSServiceInterface

{

public function sendPhoneVerificationCode(int \$code, string \$phone): array

{

\$client = new Client();

\$response = \$client->post(self::SMS_CLUB_API_ENDPOINT, [

'headers' => [

'Authorization' => __('Bearer ') . self::SMS_CLUB_API_KEY

],

'form_params' => [

'phone' => [\$phone],

'message' => __('Your verification code: ') . \$code,

'src_addr' => __('Shop Zakaz')

]

]);

return json_decode(\$response->getBody()->getContents(), true);

}

public function changePasswordRequest(string \$newPassword, string \$phone): array

{

\$client = new Client();

\$response = \$client->post(self::SMS_CLUB_API_ENDPOINT, [

'headers' => [

'Authorization' => __('Bearer ') . self::SMS_CLUB_API_KEY

],

'form_params' => [

'phone' => [\$phone],

'message' => 'Your new password: ' . \$newPassword,

'src_addr' => __('Shop Zakaz')

]

]);

return json_decode(\$response->getBody()->getContents(), true);

}

}

		Д. Шевченко			ІІЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
		С. Кравченко				83
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду BooksService:

<?php

declare(strict_types=1);

namespace App\Services\Blade;

use App\Models\BookVotes;

use Illuminate\Support\Facades\Request;

class BooksService

{

public function isUserVotedBook(): bool

{

\$url = url()->current();

```
        if (!isset(explode('/', $url)[4]) || !is_numeric(explode('/', $url)[4])) {
            return false;
        }
```

\$bookId = explode('/', \$url)[4];

return BookVotes::where('book_id', \$bookId)->where('ip', Request::ip())->exists();

}

}

Лістинг коду LibrariesService:

<?php

declare(strict_types=1);

namespace App\Services\Blade;

use App\Models\Libraries;

use Illuminate\Support\Facades\Auth;

class LibrariesService

{

public function getUserPrivateLibrary(): ?Libraries

{

return Auth::user()->libraries()->where('slug', '=', 'private')->first();

}

/**

* @return Libraries[]

*/

public function getUserLibraries()

{

return Auth::user()->libraries()->where('slug', '!=', 'private')->get();

}

}

		Д. Шевченко			ІІЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
		С. Кравченко				84
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду docker-compose.yml:

version: '3.4'

services:

backend:

build:

context: .

dockerfile: ./docker/php/Dockerfile

target: app

user: root

volumes:

- ./:/var/www/html

- ./docker/php/70-config.ini:/usr/local/etc/php/php.ini

extra_hosts:

- literary-loom.info:host-gateway

postgres:

image: postgres

ports:

- 5432:5432

volumes:

- ./docker/postgres/data:/var/lib/postgresql/data

environment:

POSTGRES_USER: piyon

POSTGRES_PASSWORD: root

POSTGRES_ROOT_PASSWORD: root

POSTGRES_DB: literary-loom

nginx:

image: nginx

user: root

ports:

- 80:80

volumes:

- ./docker/nginx/host.conf:/etc/nginx/conf.d/default.conf

- ./:/var/www/html

container_name: docker-nginx-diplom

redis:

image: redis

node:

image: node:20-alpine3.17

user: root

volumes:

- ./:/var/www/html

working_dir: /var/www/html

pgadmin:

image: dpage/pgadmin4:latest

container_name: pgAdmin4_container

		Д. Шевченко			ІІЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
		С. Кравченко				85
Змн.	Арк.	№ докум.	Підпис	Дата		

```

restart: always
ports:
  - "5050:80"
environment:
  PGADMIN_DEFAULT_EMAIL: ipz203_shdv@student.ztu.edu.ua
  PGADMIN_DEFAULT_PASSWORD: admin_admin
volumes:
  - pgadmin-data:/var/lib/pgadmin

```

```

elasticsearch:
  image: elasticsearch:7.17.9
  ports:
    - 9300:9300
    - 9200:9200
  environment:
    discovery.type: single-node
    xpack.security.enabled: false
    ES_JAVA_OPTS: -Xms512m -Xmx512m
  volumes:
    - ./docker/elasticsearch/data:/usr/share/elasticsearch/data
  ulimits:
    memlock:
      soft: -1
      hard: -1

```

```

kibana:
  image: kibana:7.17.9
  ports:
    - 5601:5601
  environment:
    - ELASTICSEARCH_HOSTS=http://elasticsearch:9200
    #- NETWORK_HOST=literary-loom.info
  depends_on:
    - elasticsearch

```

```

volumes:
  pgadmin-data:

```

		Д. Шевченко			ІІЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
		С. Кравченко				86
Змн.	Арк.	№ докум.	Підпис	Дата		