Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

Розробка MVP платформи з продажу вживаних книг

Текстова частина до курсової роботи
за спеціальністю «Інженерія програмного забезпечення» - 121

Керівник курсової роботи
с.в. _Борозенний С.О.__
(прізвище та ініціали)
_____
(підпис)
“____” _____ 2021 р.

Виконав студент _____
___Слободяник М.А._____
(прізвище та ініціали)
“____” _____ 2021 р.

Київ 2021
Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри мультимедійних систем,
доцент, к.ф-м.н.
_____О. П. Жежерун (підпис)
„____”_____2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студента Слободяника Максима Анатолійовича факультету інформатики 3-го курсу

ТЕМА: Розробка MVP платформи з продажу вживаних книг

Зміст ТЧ до курсової роботи:

- Індивідуальне завдання
- Вступ
- Розділ 1: Аналіз предметної області. Постановка завдання курсової роботи
- Розділ 2: Теоретичні відомості
- Розділ 3: Опис реалізації програмного продукту
- Висновки
- Список літератури
- Додатки

Дата видачі „___” _____ 2020 р. Борозенний С.О. _____
(підпис)

Завдання отримав _____
(підпис)

Календарний план виконання курсової роботи

Тема: Розробка MVP платформи з продажу вживаних книг

Календарний план виконання курсової роботи:

№ п/п	Назва етапу дипломного проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу	28.09.2020	
2.	Аналіз матеріалів за темою	01.01.2021	
3.	Розробка практичної частини до курсової роботи	17.02.2021	
4.	Написання записки до курсової роботи	03.04.2021	
5.	Коригування виконаної роботи	01.05.2021	
6.	Створення слайдів для доповіді та написання доповіді.	07.05.2021	
7.	Кінцеве оформлення слайдів та роботи	15.05.2021	
8.	Захист курсової роботи	24.05.2021	

Слободяник М. А. _____

Борозенний С. О. _____

“_____” _____

Зміст

Вступ

Все більше сфер нашого життя з кожним роком переходять в мережу інтернет. Раніше було важко уявити купівлю книг де-небудь ще окрім як книжкової крамниці. Проте вже зараз ми все рідше заходимо в книгарні для того, щоб купити нове видання. Сайти з продажу нових книг заполонили мережу інтернет, часто пропонуючи значно нижчі ціни ніж у фізичних магазинах. Проте кожну куплену книгу можна перечитати лише обмежену кількість разів, тож логічним є бажання продати прочитані книги, щоб звільнити місце для нових видань. На даний момент не існує спеціалізованої платформи для продажу книг, що були у вжитку, тож використовуються майданчики, що можуть надати лише частково зручні послуги для продавців вживаних книг.

З огляду на це у якості теми для своєї курсової роботи я обрав створення MVP платформи для продажу вживаних книг, що б могла надати зручний функціонал її користувачам. Свою роботу я поділив на три розділи.

В першому розділі я проаналізував власне опитування проведене серед студентів, щодо їхніх книжкових уподобань та купівельних звичок. Також цей розділ містить огляд платформ, що пропонують схожі послуги. Завершується розділ описом груп користувачів та постановкою функціонального завдання.

В другому розділі я навів теоретичні відомості щодо технологій та термінів, що були використані мною під час створення застосунку.

Третій розділ описує створення MVP платформи для продажу книг. Описує основні складові застосунку. А також окремо описує алгоритм використаний для пошуку схожих зображень в оголошеннях. Розділ завершується оглядом користувацького інтерфейсу створеної платформи.

Розділ 1: Аналіз предметної області. Постановка завдання курсової роботи

1.1 Аналіз сучасного стану питання та обґрунтування теми

1.1.1 Загальні відомості

Книга супроводжує людину вже досить довгий час. Перша друковане видання було виготовлене ще в 1574 році у Львові Іваном Федоровим і з того часу книги стали невід'ємною частиною життя багатьох людей по всьому світу. Говорячи про сьогодення, в умовах пандемії та обмежень, що накладаються урядами в більшості країн світу, книга залишилася одним з небагатьох занять доступних в межах помешкання. Тож не дивно, що згідно з повідомленням The New York Times [1] обсяги проданих книг в США лише зросли з початком пандемії попри занепад у багатьох інших галузях.

Схожу динаміку демонструє і Український книжковий ринок, згідно з дослідженням видавничої галузі України [2] наданого Українським інститутом книги. Особливе зростання спостерігається на ринку онлайн продажів звичайних та електронних книг. Окрім того, це ж дослідження також повідомляє про збільшення інтересу до книг виданих саме Українською мовою та зниження попиту на виключно російськомовні книги. Так в порівнянні з 2018 роком попит на книги українською мовою зріс на 8%, а кількість людей, що надавала перевагу виключно російськомовним книгам знизилася на 26%.

Згідно іншого дослідження [3] проекту Ukrainian Reading and Publishing Data середня ціна купленої книги складає 135 грн, що є досить невеликою цифрою, якщо поглянути на полиці книгарень. В цьому ж дослідженні також йдеться про те, що 21.8% опитаних, які зазвичай не читають книги, не роблять цього через відсутність коштів на придбання нових видань. Також 35% опитаних завантажували хоча б одну книгу безкоштовно, тобто в піратський спосіб

Згідно цих даних можна стверджувати, що тема книг є актуальною в Україні та світі, проте питання доступності книг займає важливе місце у їхньому розповсюдженні.

В Українському сегменті інтернету не існує популярної інтернет платформи, яка спеціалізується на продажу актуальних книг, що були у вжитку. Водночас можна знайти безліч груп з продажу подібних товарів, в різноманітних соцмережах. Пошук по таким оголошенням неможливий або незручний, тож результативність подібних продажів не є високою. Наявність великої кількості оголошень свідчить про значний попит на подібні послуги.

1.1.2 Аналіз опитування, щодо книжкових уподобань серед студентів

Для визначення актуальності створення платформи для продажу вживаних книг, мною було проведено опитування [4] серед студентів Києво-Могилянської академії де були поставлені питання, що дозволяють визначити книжкові уподобання сучасної молоді. Загалом в опитуванні взяв участь 91 респондент. Було отримано такі результати:
Більше половини опитаних, а саме 67% надають перевагу друкованим виданням.

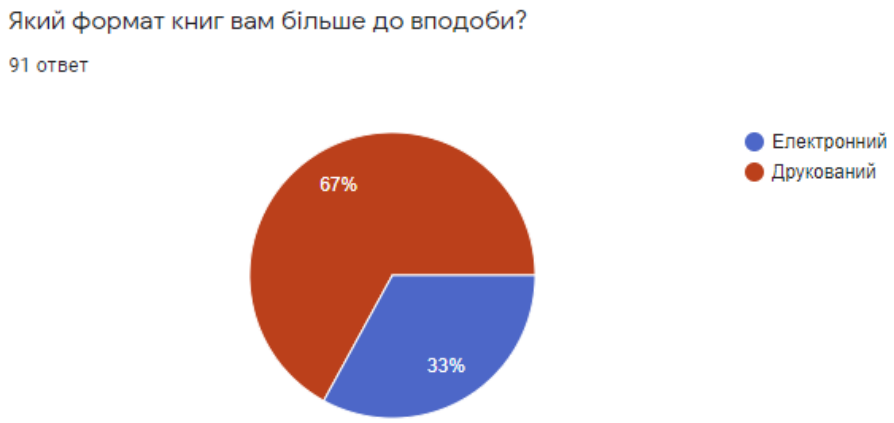

Рис. 1.1 Кругова діаграма розподілу відповідей студентів на питання, щодо формату книг, що їм до вподоби

Найпопулярнішими способами отримання нових книг є: завантаження електронних версій книг з піратських сайтів (58% опитаних), купівля паперових книг (49% опитаних), купівля електронних книг (37% опитаних) та обмін книгами з друзями (31% опитаних).

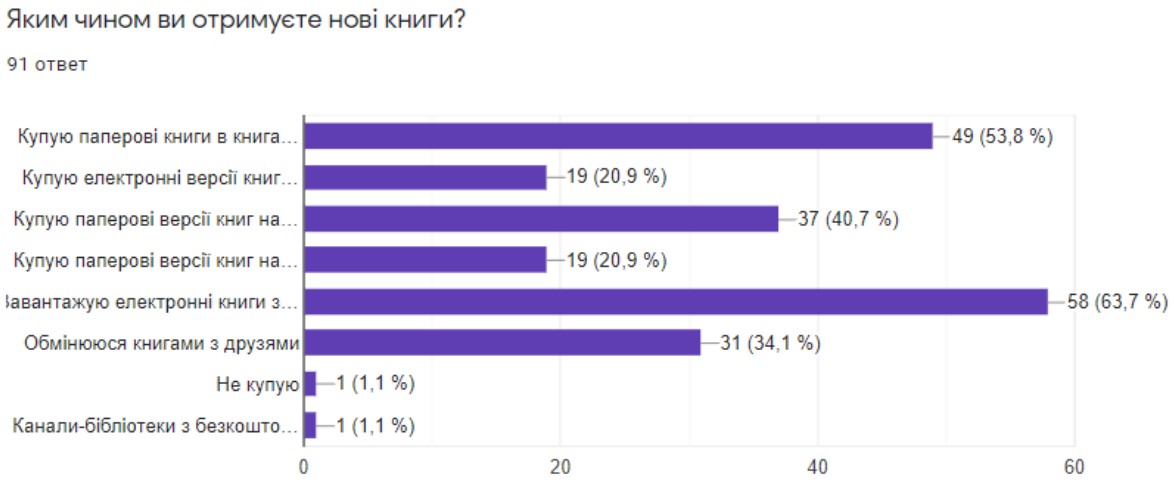

Рис. 1.2 Стовпчикова діаграма розподілу відповідей студентів на питання, щодо способів отримання нових книг

Більшість опитаних витрачають на книги не більше 400 гривень на місяць.

Рис. 1.3 Кругова діаграма розподілу відповідей студентів на питання, щодо кількостей грошей які вони зазвичай витрачають на книги протягом місяця

79.1% опитаних позитивно сприймають ймовірність створення спеціалізованого ресурсу для продажу вживаних книг.

Рис. 1.4 Кругова діаграма розподілу відповідей студентів на питання, щодо ставлення до створення спеціалізованої платформи продажу вживаних книг

78% респондентів готові купувати актуальні книги, що були у вжитку.
Також лише 15% відповіли, що вже користуються сайтом покупки вживаних книг. Вони використовують соціальну мережу Instagram або сайт розміщення оголошень OLX.
53.8% опитаних надають перевагу виданням написаним українською мовою, 29.7% російською, 12.1% англійською, для решти мова книги не є важливою.
22% респондентів читають більше 15 книжок щороку, 15,4% читають від 10 до 15 книжок, 17.6% 5 -10 книжок, 16.5% 3-5 книжок, 24.4% 1-3 книги, 4.4% опитаних не читають книг.

Аналізуючи результати опитування, можна дійти висновку, що найбажанішою книгою – є друковане видання українською мовою. Для половини опитаних студентів ціна книг є занадто високою, що стає поштовхом до безкоштовного завантаження піратських книг для задоволення своїх читацьких потреб. Варто також

зазначити, що сучасні україномовні книги не поширені на піратських сайтах, тож єдиним можливим способом прочитати твір – є його придбання.

Результати проведеного мною опитування підтверджують результати опитувань, що наводилися в Розділі 1.1.1 Загальні відомості, та наштовхують на висновок, що в Україні існує попит на книжкову продукцію, проте не існує авторитетного майданчику для зручного продажу або покупки вживаної книги. Зважаючи на отримані результати розробка MVP платформи для продажу вживаних книг є актуальною для українського сегменту мережі інтернет.

1.2 Огляд існуючих аналогів

1.2.1 OLX

Однозначним фаворитом серед сайтів подачі оголошень можна назвати сайт olx.ua. Платформа пропонує подачу та пошук оголошень в безлічі категорій. Серед переваг цього майданчику можна назвати його репутацію, а отже він містить величезну кількість оголошень пов’язаних з книгами. Також існує можливість шукати оголошення лише в певній області або місті. Присутня фільтрація цін та стану товару. При відкритті певного оголошення, майданчик запропонує вам схожі оголошення, проте ця функція у випадку книг не дуже релевантна. Відкривши оголошення про продаж детективного роману, в схожих оголошеннях можна знайти пропагандистську радянську літературу. Платформа містить перевірку на продубльовані оголошення та не допускає їх до публікації. Підтримку безпечної доставки товарів та банери, що попереджають про можливе шахрайство, що зменшує шанси бути ошуканим. Проте також присутні і мінуси. Не релевантний фільтр в категорії книги/журнали адже неможливо відфільтрувати книгу за жанром, автором, видавництвом, кількістю сторінок, тощо. Створення оголошення потребує великої

кількості часу, адже потрібно заповнити багато інформації про книгу, придумати опис, та описати всі характеристики, що можуть бути потрібні при пошуку книги. Користувач, що шукає підходящу книгу повинен відкривати кожне оголошення для отримання подробиць про книгу. надокучлива реклама, що часто маскується під оголошення, що зменшує результативність пошуку.

На даний момент на сервісі знаходиться близько 1250 активних оголошень в розділі книги/журнали.

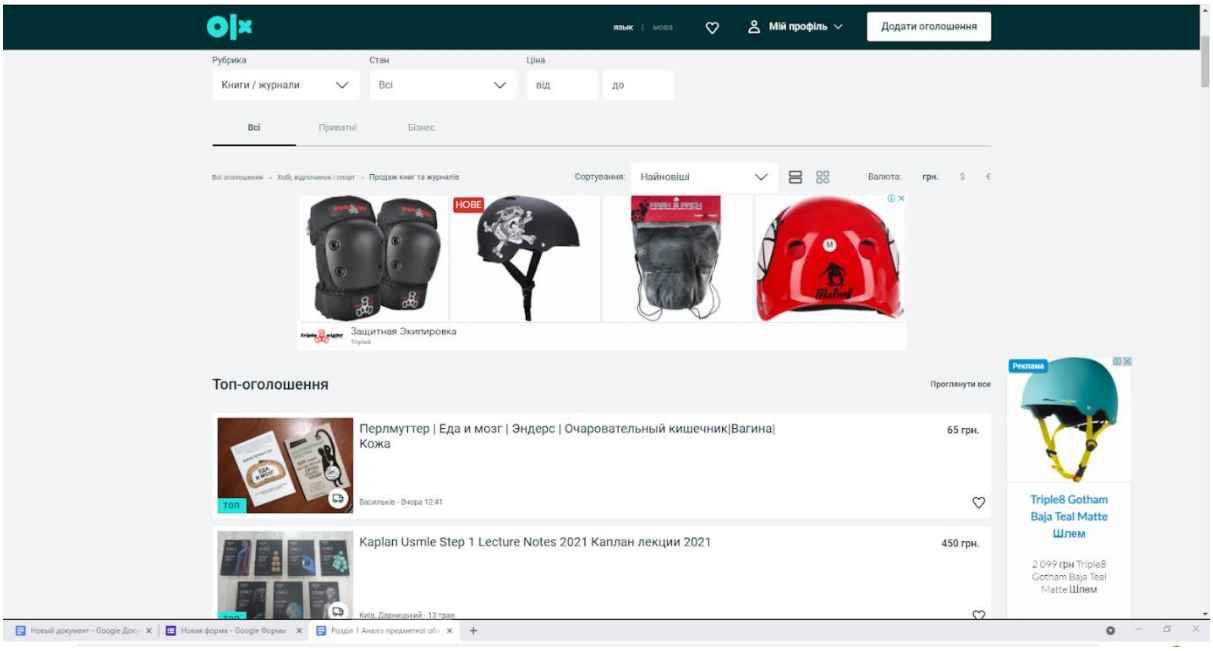

Рис. 1.5 Користувацький інтерфейс сайту з розміщення оголошень OLX[15]

1.2.2 prom.ua

Інший відомий майданчик створення оголошень нових та вживаних товарів. Цей маркетплейс був заснований 2008 року та містить в собі більше 100 мільйонів товарів. Майданчик має велику кількість оголошень пов’язаних з книжковою тематикою. Можна окреслити такі переваги. Наявність поділу книг за напрямками, що значно звужує коло пошуку необхідної книги. В кожному напрямі присутні

релевантні фільтри. Платформа слідкує за доброчесністю продавців, та вживає заходів задля дотримання правил публікацій оголошень. До недоліків можна віднести переважну кількість бізнес оголошень, оскільки платформа розрахована на продавців, що створюють свій онлайн магазин на платформі, а не звичайних користувачів, що продають 1-2 книги, щоб звільнити свої книжкові полиці. Фільтрування ціни не відповідає оголошенням, що відображаються. Відсутність найважливішої інформації у списку знайдених оголошень, необхідно заходити в оголошення для отримання більшої кількості деталей.

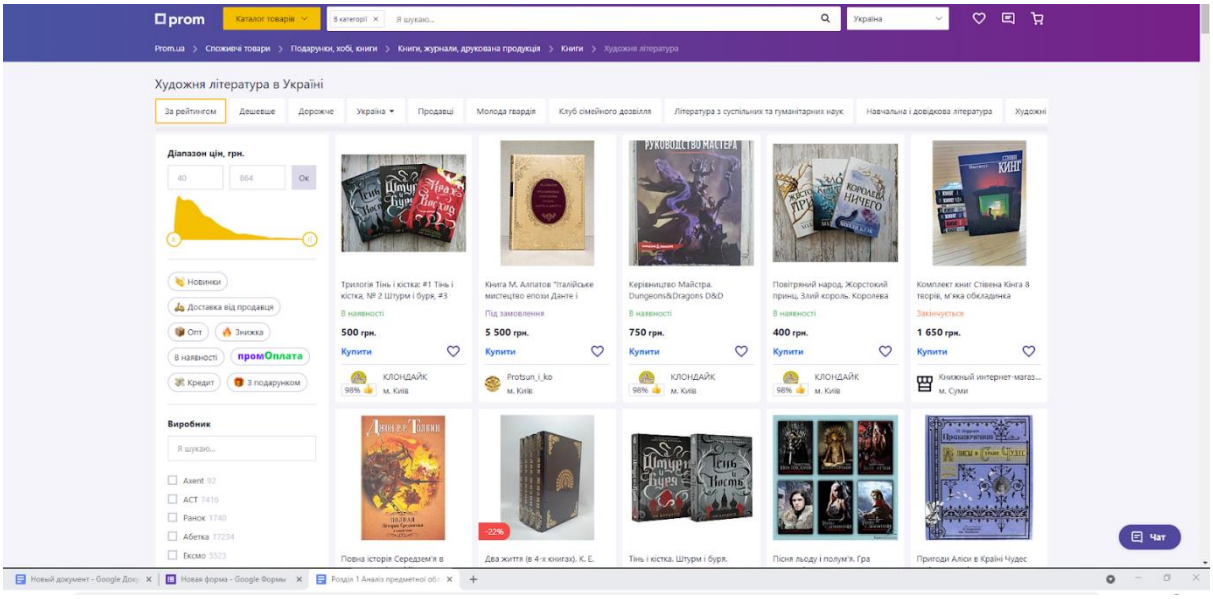

Рис. 1.6 Користувацький інтерфейс сайту з розміщення оголошень prom.ua[16]

1.2.3 Instagram

Instagram - соціальна мережа, що охоплює величезну аудиторію різноманітного віку та соціального статусу. Кількість користувачів перевищує 1 мільярд. Соціальна мережа дозволяє створювати таргетовані рекламні кампанії, що дозволяють тримати ціну залучення користувача на досить низькому рівні. Також розміщення реклами інтегрується в стрічку користувача, тож він гарантовано дізнається про ваш магазин. До переваг можна віднести доступність вашого магазину пересічному користувачеві,

адже йому не потрібно завантажувати окремий додаток або заходити на окремий сайт.

Відносно легка рекламна взаємодія проте українські Instagram магазини, що продають вживані книги рідко користуються рекламою, адже не мають досить грошей для цього.

За умови підписки користувача можна час від часу повідомляти йому різноманітну інформацію безкоштовно не залучаючи рекламу. Недоліки використання соціальної мережі Instagram такі. Неможливість пошуку книг за параметрами. Єдина можливість щось знайти - це гортання стрічки. Зазвичай подібні магазини не мають великої

кількості підписників та актуальних книг. Також існує велика ймовірність натрапити на шахраїв. Невеликий асортимент оголошень.

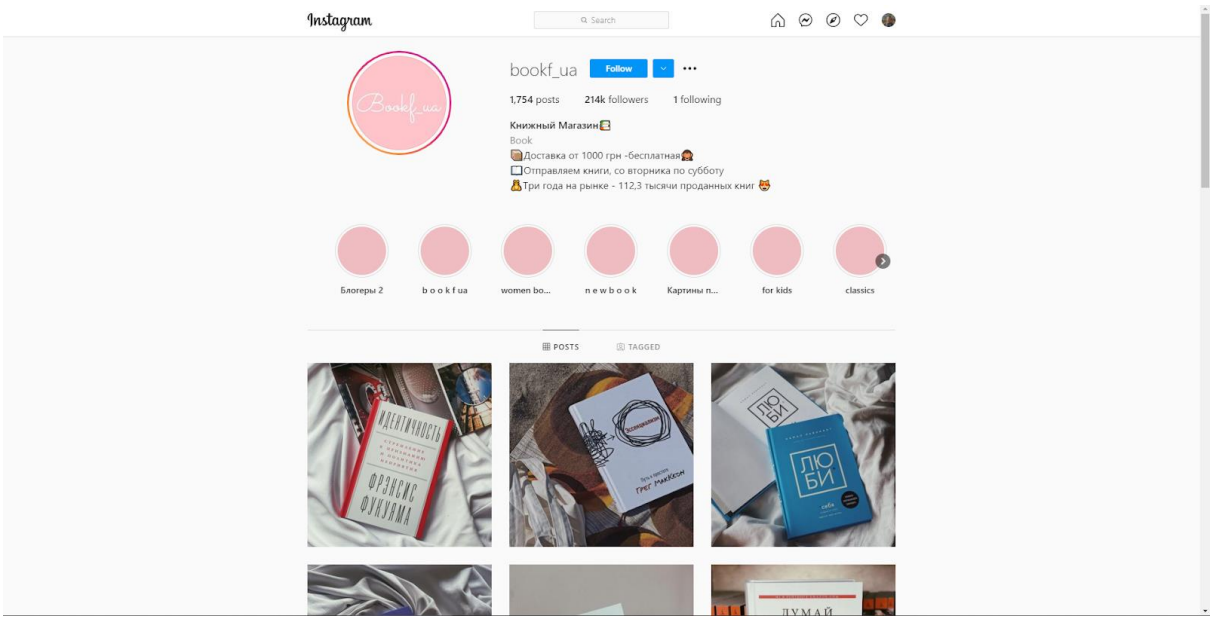

Рис. 1.7 Користувацький інтерфейс магазину з продажу книг в соціальній мережі Instagram[17]

1.3 Опис груп користувачів

Платформа матиме дві групи користувачів наділених різними правами та рівнем доступу до інформації, а саме User та Moderator

До групи User належить будь-який користувач, який самостійно зареєструвався в системі. Користувач з групи User може виконувати такі дії в застосунку:

1) Робота з акаунтом:
   - реєстрація акаунту за допомогою електронної пошти користувача, імені користувача та паролю;
   - Вхід до акаунту за допомогою електронної пошти користувача та паролю
   - Видалення акаунту;
2) *Робота з профілем:*
   - Редагування інформації акаунту (ім’я користувача, пароль, місце де живе користувач, аватар користувача)
3) Взаємодія з власними оголошеннями:
   - Створення оголошення
   - Створення оголошення у швидкому форматі використовуючи isbn книги
   - Редагування оголошення
   - Перегляд власних оголошень
   - Перегляд статусу власних оголошень
4) Взаємодія з оголошеннями інших користувачів
   - Перегляд стрічки оголошень
   - Пошук оголошень застосовуючи фільтри
   - Помітка оголошень такими, що зацікавили користувача
   - Перегляд певного оголошення

- Перегляд оголошень відмічених як такі, що сподобалися

До групи Moderator належить користувач, який був доданий до системи з роллю Moderator. Користувач з групи Moderator може виконувати такі дії в застосунку:

1. Модерація
    - Перегляд оголошень користувачів, що були запідозрені у порушенні правил платформи.
    - Відмова у публікації оголошень користувачів, що були запідозрені у порушенні правил платформи.
    - Підтвердження унікальності підозрілого оголошення та подальшу його публікацію

1.4 Постановка задачі, функціональні вимоги до платформи

Задачею моєї курсової є розробка веб-сервісу(платформи), що надає можливості для зручного продажу та покупки книг, що були у вжитку. Платформа має пришвидшити процес створення оголошень, та надати зручні інструменти пошуку серед наявних оголошень. Проаналізувавши аналогічні майданчики оголошень, я дійшов висновку, що платформа повинна відповідати таким функціональним вимогам:

- Реєстрація нових та вхід в систему для вже існуючих користувачів .
- Перегляд та взаємодія з оголошеннями інших користувачів.
- Пошук серед наявних оголошень.
- Створення редагування та видалення власного оголошення.
- Модерація поданих оголошень
- Перегляду та редагування користувачем свого профілю

Розділ 2: Теоретичні відомості

2.1 REST

Для організації зв’язку клієнтської та серверної частин я використовую архітектуру REST (REpresentational State Transfer).
Однією з переваг REST підходу, можна назвати передачу даних в тому ж форматі, що вони зберігаються на сервері. Це не лише пришвидшує роботу системи в цілому, але знижує її складність, адже неіснує проміжних рівнів які б здійснювали упаковку даних в XML формат як у випадку підходу SOAP.
Сервіси, що реалізують REST архітектуру, називаються RESTful сервісами. Для доступу
до даних необхідно реалізувати REST API в якому кожному покликанню відповідатимуть дії над даними та їхнє повернення у відповідь. Архітектурний стиль REST побудований на базі HTTP протоколу. Та повністю підтримує типи запитів, що надаються HTTP протоколом, а саме GET, PUT, POST, DELETE, OPTIONS, PATCH, HEAD.
Запит типу GET застосовується у випадку необхідності отримання даних з сервера відповідно до заданих параметрів. Наприклад при необхідності отримання всіх клієнтів, що відвідують магазин в певному місті.
Запит типу PUT застосовується у випадку необхідності оновлення вже існуючих даних
на сервері цілком. Наприклад оновлення всіх полів клієнта магазина.
Запит типу POST застосовується для додавання даних на сервіс. Наприклад для реєстрації нового клієнта магазину
Запит типу DELETE застосовується для видалення даних з сервісу.
Наприклад для видалення акаунту клієнта з магазину
Запит типу PATCH застосовується у випадку необхідності оновлення вже існуючих

даних на сервері частково. Запит типу PATCH схожий за застосуванням з методом PUT, проте дозволяє змінювати не весь об’єкт цілком, а лише деякі його поля. Наприклад оновлення лише віку клієнта магазина.
Запит типу HEAD схожий за своєю функціональність з методом GET, проте не повертає ніяких даних. Зазвичай використовується для отримання заголовків (Headers) які будуть повернуті з методом GET. Часто використовується перезавантаженням великої кількості даних.
Запит типу OPTIONS застосовується для отримання доступних типів запитів, які можуть звернутися за певним покликанням. Використовується деякими браузерами перед надсиланням запитів з типами описаними вище.

Найпопулярнішими типами HTTP запитів, що використовуються в REST є POST, GET, PUT, DELETE, що є відповідниками функціям управління даними, що позначаються акронімом CRUD (Create, Read, Update, Delete).

2.2 MVC

MVC - архітектурний шаблон, що є досить поширеним в сучасних веб-застосунках. Акронім MVC можна розшифрувати як Model(модель даних), View (інтерфейс користувача) та Controller (модуль керування). Отже він передбачає розділення застосунку на 3 основні рівні.

В цьому архітектурному шаблоні центральним компонентом є модель. Від моделі даних залежить формат та групування даних при їх збереженні застосунком. Іноді в моделі знаходиться також бізнес логіка, необхідна для роботи з моделлю. Проте часто для її реалізації використовують проміжний рівень сервісу, що знаходиться в дата флоу між модулем керування та моделлю. Це дозволяє не загромаджувати модель даних

додатковою не притаманною їй логікою, а бізнес логіку тримати в окремому гарно структурованому сервісі.

Модуль керування відповідальний за отримання запитів з клієнтського застосунку, та сповіщення сервісу про необхідність виконати додавання, оновлення або отримання

даних для користувача. Іноді бізнес логіка розміщується всередині модуля керування, проте це не є гарним стилем проектування контролерів.

Представлення відповідає за відображення користувацького інтерфейсу та отриманні дій користувача, які повинні бути переданими контролеру.

Така архітектура застосунку дозволяє безболісно змінювати її частини не створюючи необхідності в змінах в інших частинах застосунку.

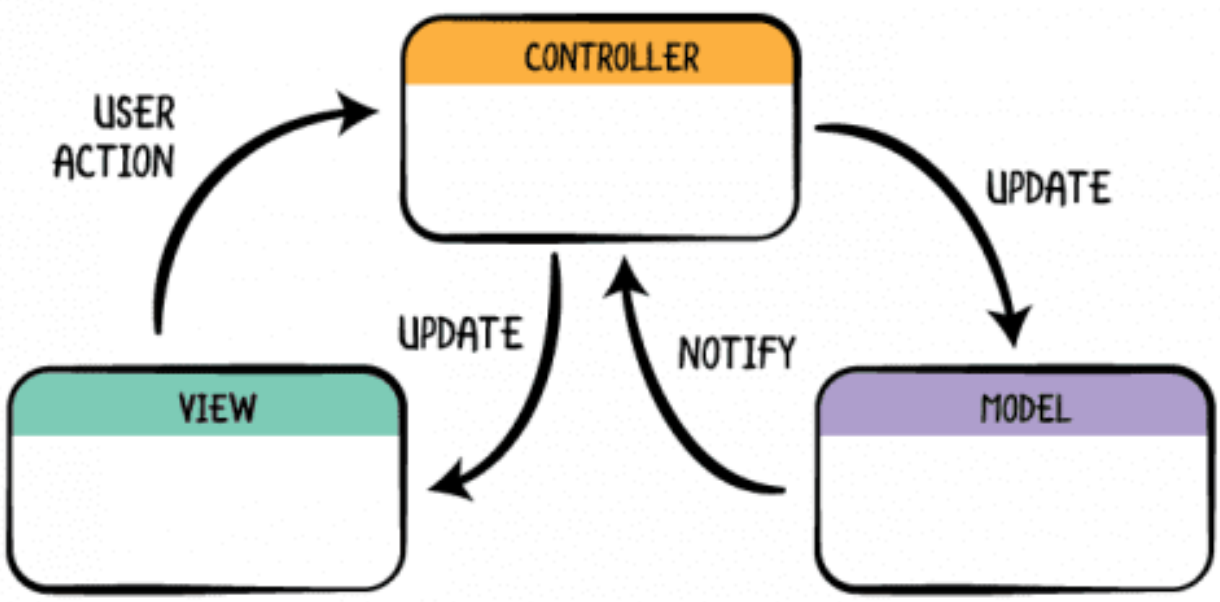

Рис. 2.1 Ілюстрація взаємодії частин партерну MVC

2.3 JWT

JWT - це об’єкт у вигляді JSON який визначений в стандарті RFC 7519. JWT

складається з трьох частин: заголовку (Header), корисного навантаження (Payload) та підпису (VERIFY SIGNATURE).

Заголовок містить загальну інформацію про токен, таку як алгоритм шифрування, тип. Дані заголовку шифруються.

Приклад закодованого заголовку:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9

В розкодованому вигляді це звичайний JSON об’єкт:

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Корисне навантаження також є JSON об’єктом і може містити в собі будь-яку інформацію, яка може знадобитися для аутентифікації користувача.

Приклад закодованого корисного навантаження:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9

В розкодованому вигляді це звичайний JSON об’єкт:

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022
}
```

Підпис обраховується на основі перших двох частин. Після цього всі три частини поєднуються через крапку, таким чином формуючи кінцевий токен.

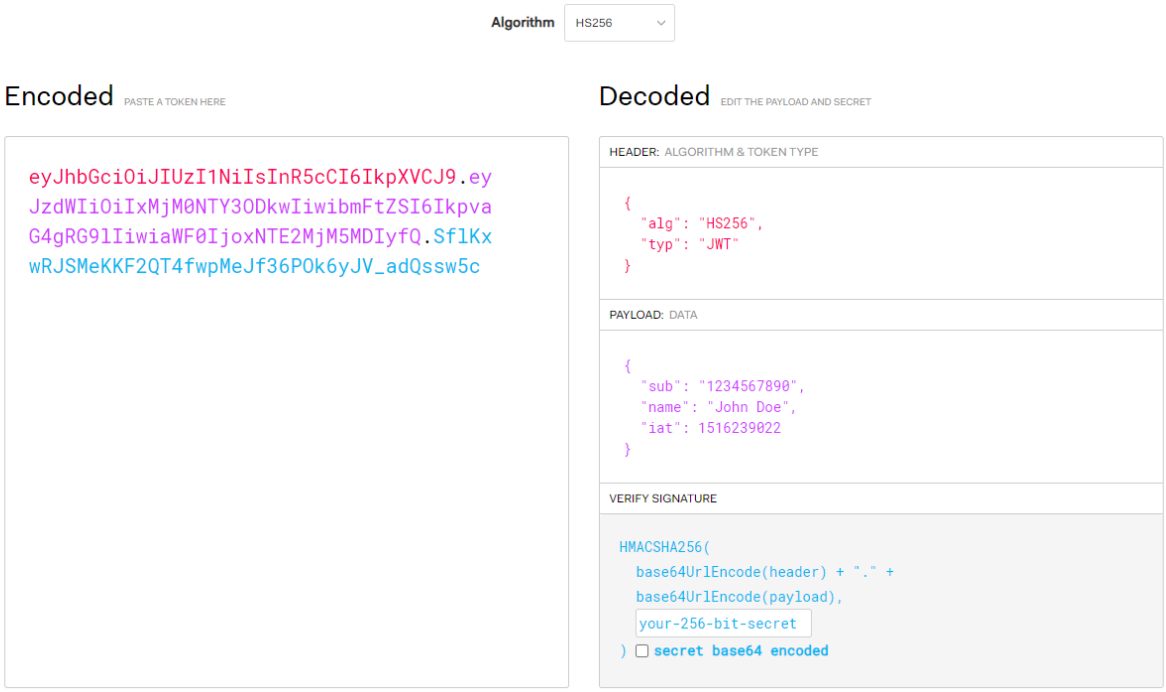

Рис. 2.2 Ілюстрація зашифрованого та розшифрованого JWT токену[7]

В своєму застосунку я використовую JWT токен для аутентифікації користувача. При намаганні увійти в систему користувачеві повертатиметься згенерований токен, який він зможе використовувати для подальшої взаємодії з серверною частиною. Фронтенд
частина застосунку додає до кожного запиту окрім запиту на вхід та реєстрацію токен в той же час на бекенд частині застосунку існує фільтр, що дозволяє перевіряти валідність токена, а також чи існує користувач, чиї дані знаходяться в токені. У разі некоректного токена у доступі буде відмовлено.

```java
package ua.slobodianyk.bouquiniste.config;

import ...

@Component
public class JWTRequestFilter extends OncePerRequestFilter {
    @Autowired
    private TokenService tokenService;

    @Override
    protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain chain)
            throws ServletException, IOException {

        final String header = request.getHeader(HEADER_STRING);

        if (header == null || !header.startsWith(TOKEN_PREFIX)) {
            chain.doFilter(request, response);
            return;
        }

        UsernamePasswordAuthenticationToken authentication = getAuthentication(header);

        SecurityContextHolder.getContext().setAuthentication(authentication);
        chain.doFilter(request, response);
    }

    private UsernamePasswordAuthenticationToken getAuthentication(String token) {
        if (token != null) {
            var tokenString = token.replace(TOKEN_PREFIX, "");
            String user = tokenService.extractUserid(tokenString);
            if (user != null && !tokenService.isTokenExpired(tokenString)) {
                return new UsernamePasswordAuthenticationToken(user, null, new ArrayList<>());
            }
            return null;
        }
        return null;
    }
}
```

Рис. 2.3 Ілюстрація обробки автентифікації користувача на бекенді з використанням JWT токену

2.4 ORM

Рівень ORM (Object-Relational Mapping)[8] знаходиться між рівнями застосунку та базою даних. Його основна мета полягає у співставленні об’єктів мов об’єктно-орієнтованої парадигми з таблицями в базі даних і навпаки. На основі моделей даних можуть генеруватися таблиці в базі даних. Зазвичай до моделі додається додаткова інформація

яка дозволяє коректно побудувати зв’язки між сутностями, або накласти певні

обмеження на додавання або отримання даних з таблиці. В своєму застосунку я використовую Spring Data JPA який в своїй реалізації використовує Hibernate у якості

ORM рівня. Таким чином таблиці в базі даних генеруються на основі моделей. Існує можливість вказувати додаткові параметри для колонок та таблиці, а також типи зав’язків між таблицями за допомогою анотацій.

Розділ 3: Опис реалізації програмного продукту

3.1 Опис технічного завдання на розробку

Метою виконання технічного завдання – є розробка MVP майданчику продажу вживаних книг. Суттєвими відмінностями, що покращать користувацький досвід цього майданчика в порівнянні з наведеними Розділі 1 аналогами, повинні стати: спеціалізація виключно на подачі оголошень з продажу вживаних книг, релевантна фільтрація оголошень, можливість швидкого заповнення полів оголошення за допомогою ISBN, модерація оголошень на предмет розміщення оголошень з однаковими фотографіями.
Застосунок повинен мати клієнтську частину та серверну.

Оголошення про продаж книги повинно містити таку інформацію:

- Від 1 до 3 фотографій книги
- ISBN книги
- Назва книги
- Автор книги
- Видавець книги
- Рік видання книги
- Жанр книги
- Мова книги
- Ціна книги
- Кількість сторінок в книзі
- Опис книги

Система повинна мати такий функціонал:

1. Надавати можливість реєстрації користувача в системі.

2. Надавати можливість входу користувача до системи.

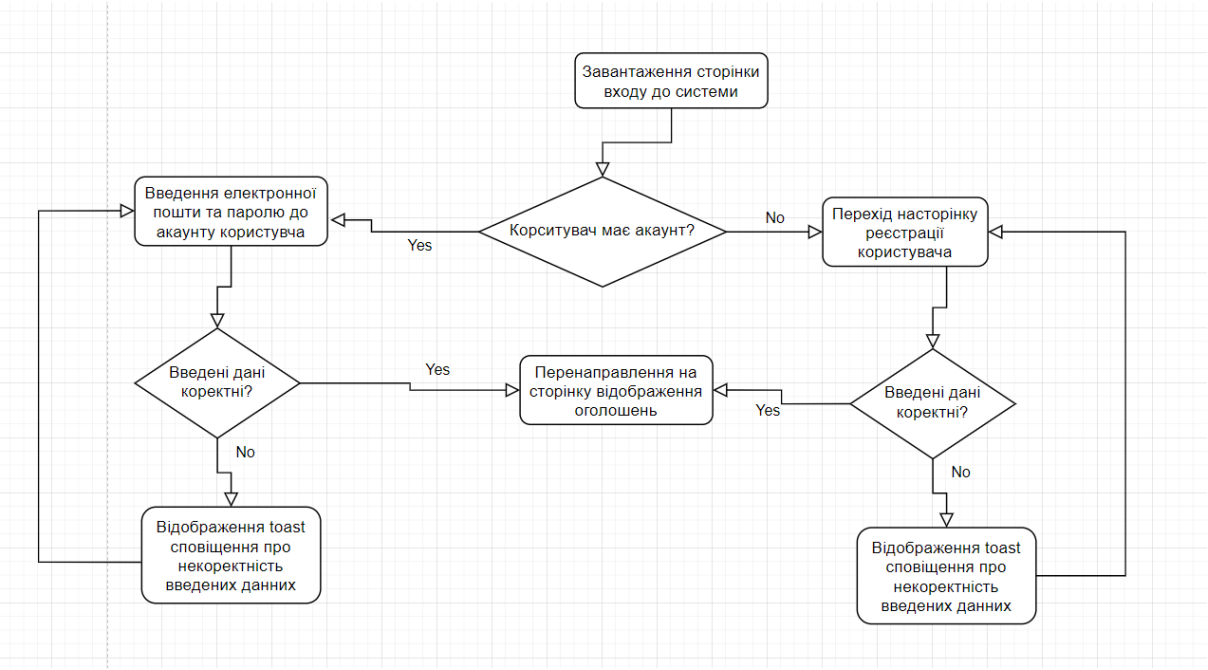

Рис. 3.1 Блок-схема шляху користувача під час процесу реєстрації та входу в систему

3. Надавати можливість перегляду опублікованих оголошень
4. Надавати можливість пошуку оголошень з застосуванням фільтрації.

   Параметри фільтрації:

   - ISBN книги
   - Назва книги
   - Автор книги
   - Видавець книги
   - Рік видання книги
   - Жанр книги
   - Мова книги
   - Ціна книги більше за введене значення
   - Ціна книги менше за введене значення
   - Місто де знаходиться книга

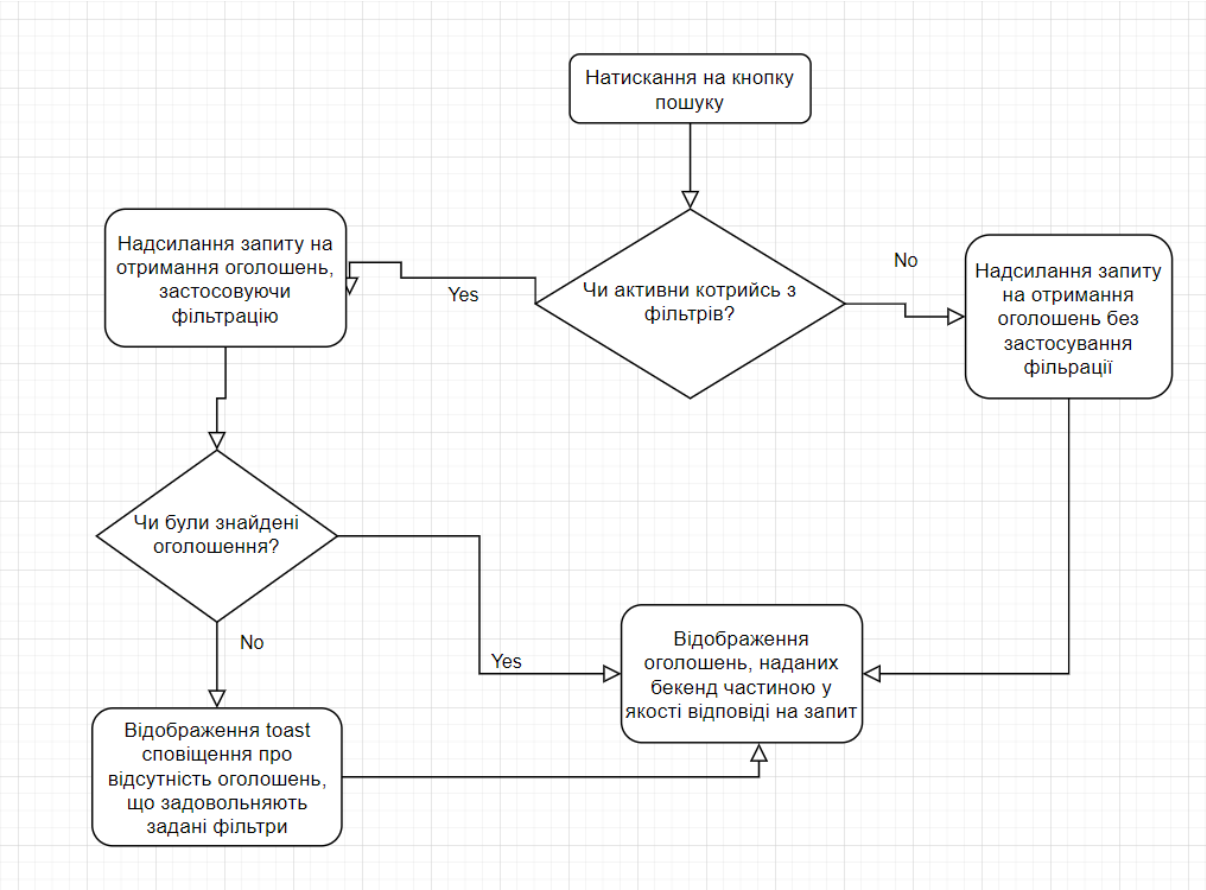

Рис. 3.2 Блок-схема шляху користувача під час процесу пошуку оголошень

5. Картка оголошення, що відображається в списку оголошень повинна містити фото книги, що продається та інформацію щодо: ціни книги, автора книги, жанру книги та мови якою написана книга.
6. Надавати можливість перегляду повної інформації про розміщене оголошення на його сторінці.
7. Надавати можливість створення оголошення.
8. Надавати можливість швидкого заповнення полів оголошення за допомогою ISBN книги, що міститься в оголошенні.

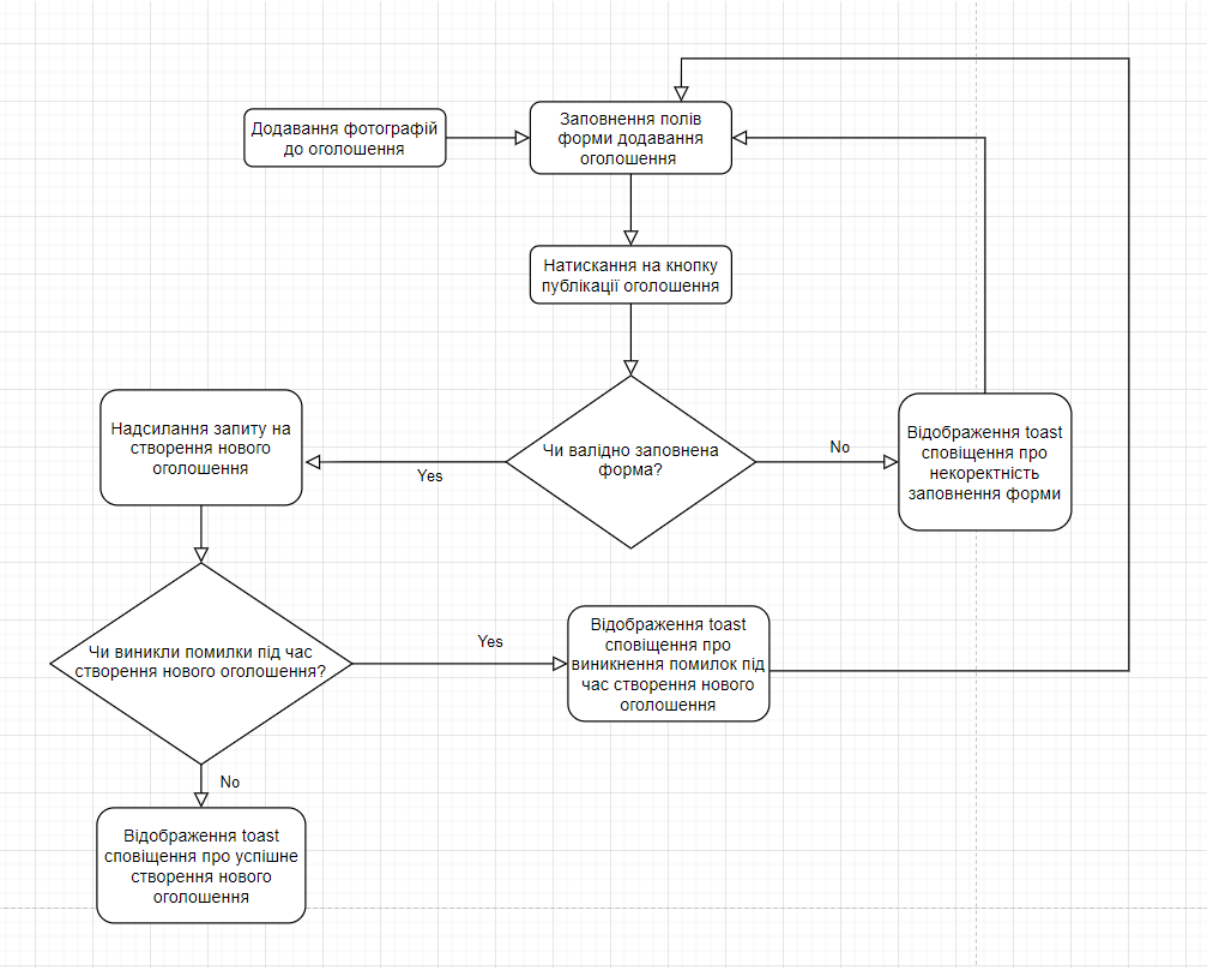

Рис. 3.3 Блок-схема шляху користувача під час процесу додавання оголошення

9. Надавати можливість редагування оголошення
10. Надавати можливість видалення оголошення
11. Перевірка оголошення перед публікацією на предмет дублювання прикріплених фотографій
12. Занесення підозрілих оголошень до списку модерації

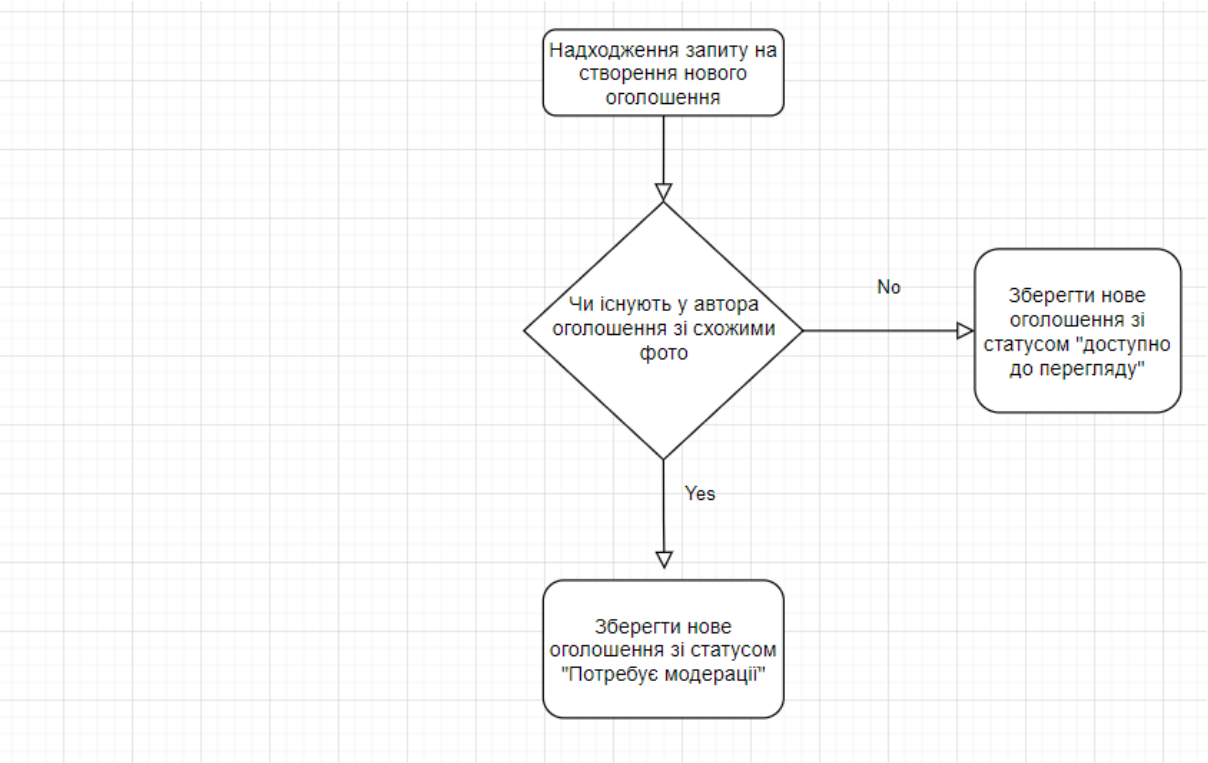

Рис. 3.4 Блок-схема поведінки програми під час додавання нового оголошення

13. Надавати можливість модерувати оголошення.
14. Надавати можливість переглядати оголошення автора перед прийняттям рішення про модерацію.

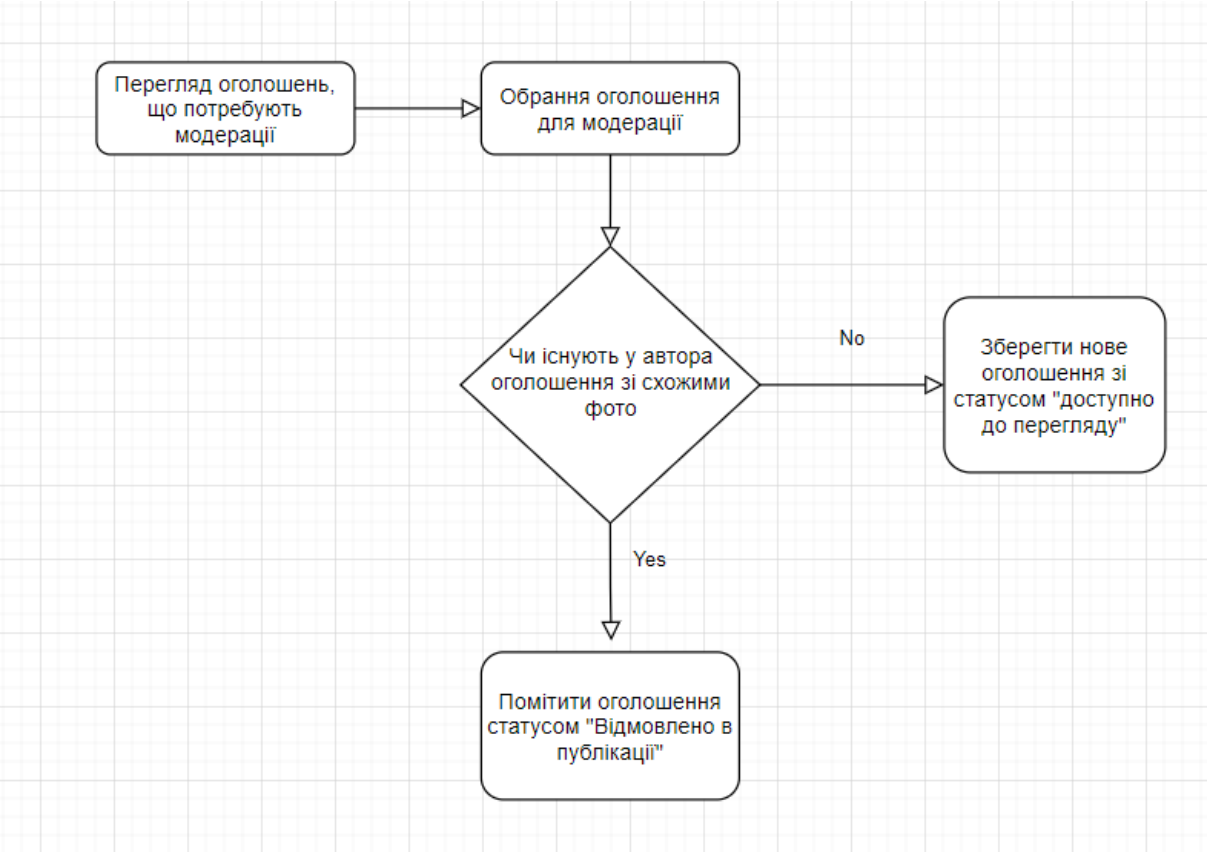

Рис. 3.5 Блок-схема шляху користувача під час процесу модерації оголошення

15. Надавати можливість відмічати оголошення як таке, що сподобалося.
16. Надавати можливість перегляду оголошень, що сподобалися.
17. Надавати можливість перегляду користувачем свого профілю.
18. Надавати можливість редагування користувачем свого профілю.
19. Надавати можливість видалення користувачем свого акаунту.

3.2 Обґрунтування вибору засобів розробки

Застосунок складається з клієнтської частини, серверної частини та бази даних. Для клієнтської частини я обрав реалізацію з використанням бібліотеки React.js[10] у зв’язці з Redux[13] для організації store.

React.js був створений компанією Facebook в 2013 році. На зараз вже вийшла 17 версія цієї бібліотеки. На сьогоднішній день React є одним з найпопулярніших

інструментів для створення SPA (single page application). Його особливістю є віртуальний DOM який дозволяє значно підвищити швидкодію застосунку, адже віртуальний DOM дозволяє не перемальовувати DOM для кожної зміни, він намагається групувати зміни і застосовувати їх пакетами. Не менш важливим фактором є спільнота розробників, що використовують цю бібліотеку для своїх проектів, тож при розробці завжди можна буде знайти відповідь на труднощі, що можуть виникнути під час розробки. Існує безліч бібліотек для React.js, що дозволяють додати різноманітний функціонал який React.js не надає з коробки. Наприклад бібліотека для організації маршрутизації не входить в комплект поставки React.js, проте існує багато бібліотек, що надають подібний функціонал, наприклад react-router-dom, яку я використав в своєму застосунку. React.js надає можливості для гнучкості розробки адже функціонал, який не постачається з коробки, можна додавати з будь-яких наявних бібліотек. Тож розробник завжди має вибір з чим працювати. Також я використав бібліотеку Redux для керування станом застосунку. Redux дозволяє в зручний спосіб керувати станом застосунку. За його використання всі дані, що використовуються в застосунку зберігаються в одному місці - стані. Така організація даних дозволяє застосунку бути передбачуваним та легким в підтримці. Використовуючи розширення reduxDevTools[14], розробник може переглядати зміни в стані та його актуальний стан, що пришвидшує та спрощує розробку.
Підсумовуючи мною була обрана така зв’язка бібліотек у зв’язку з актуальністю технологій, швидкодією бібліотеки React.js, швидкістю розробки та легкістю подальшої підтримки.
Для серверної частини я обрав реалізацію з використанням Spring Boot та моаою Java.
Spring Boot[11] - проект, що націлений на спрощення створення проектів на базі Spring. Його перевагами є керування залежностям, автоматична конфігурація а також внутрішні контейнери сервлетів. Розширення функціоналу застосунків на базі Spring Boot можна здійснювати додаючи стартер-пакети. Вони містять в собі всі

необхідні залежності для додавання певного функціоналу. Для реалізації цього проекту я використав такі стартер -пакети: spring-boot-starter-jpa, spring-boot-starter-security, spring-boot-starter-web. Завдяки такій гнучкості і простоті в налаштуваннях проекти можливо швидко запускати та додавати новий функціонал.

У якості бази даних я використав СКБД postgreSQL[12]
PostgreSQL - об’єктно-реляційна система керування базами даних з великою кількістю переваг, що вона може запропонувати своїм користувачам.
Серед них можна виділити:

- підтримка бази даних необмеженого розміру, що є корисним для застосунків, що планують активно маштабуватися.
- Потужні механізми транзакцій та реплікацій.
- Наявність індексів, що пришвидшує пошук даних.
- Підтримка нестандартних типів для таких типів даних як ip адреса або геодані.
- Надійність та стабільність.
- Можливість створення тригерів та функцій.
- Можливість створення власних типів.
- Можливість збереження JSON.

Ця СУБД також має досить велику спільноту, тож зіткнувшись з труднощами буде простіше знайти рішення. PostgreSQL має JDBC драйвер для Java, тож є зручною для використання з цією мовою.

3.3 Розробка програмного застосунку

3.3.1 Проектування бази даних

База даних застосунку складається з 8 таблиць

Основною є таблиця з назвою books. Вона містить первинний ключ id типу uuid. Зовнішні ключі genre типу varchar(255) та user_id типу uuid, а також інші поля: created_at типу timestamp,
deleted_on типу timestamp, updated_on типу timestamp, author типу varchar(255), description типу text, isbn типу varchar(255), language типу varchar(255), number_of_pages типу integer, price типу double precision,
publisher типу varchar(255), title типу varchar(255), year_of_publishing типу integer.
Назви доступних жанрів містяться в таблиці genre. Вона містить первинний ключ id типу uuid.
Таблиця ad_images містить в собі зображення, що були додані до оголошення під час створення або редагування оголошення. Вона має первинний ключ id типу uuid. Зовнішній ключ book_id типу uuid, а також інші поля: created_at типу timestamp, deleted_on типу timestamp, updated_on типу timestamp,
delete_hash типу varchar(255), image_hash типу bigint, is_primary типу boolean, link типу varchar(255).
Таблиця users містить в собі користувачів системи. Вона має первинний ключ id типу uuid. Та зовнішній ключ avatar_id типу uuid, а також інші поля:
created_at типу timestamp, deleted on типу timestamp, updated_on типу timestamp, email
типу varchar(255), location типу varchar(255), password типу varchar(255), tel типу varchar(255), username типу varchar(255),
Таблиця avatar_images містить в собі зображення, що були додані користувачем у якості фото свого профілю. Вона має первинний ключ id типу uuid, а також інші поля: created_at типу timestamp, deleted_on типу timestamp, updated_on типу timestamp, delete_hash типу varchar(255), link типу varchar(255).
Таблиця bookmarks містить в собі закладки користувачів. Вона має первинний ключ id типу uuid. Зовнішні ключі book_id типу uuid та user_id типу uuid, а також інші

поля: created_at типу timestamp, deleted_on типу timestamp, updated_on типу timestamp.

Таблиця roles містить в собі ролі користувачів. Вона має первинний ключ id типу uuid, а також поле name типу varchar(255).

Схема таблиць також містить допоміжну таблицю user_roles, що слугує для поєднання ролей та користувачів. Вона має user_id типу uuid та role_id типу uuid.

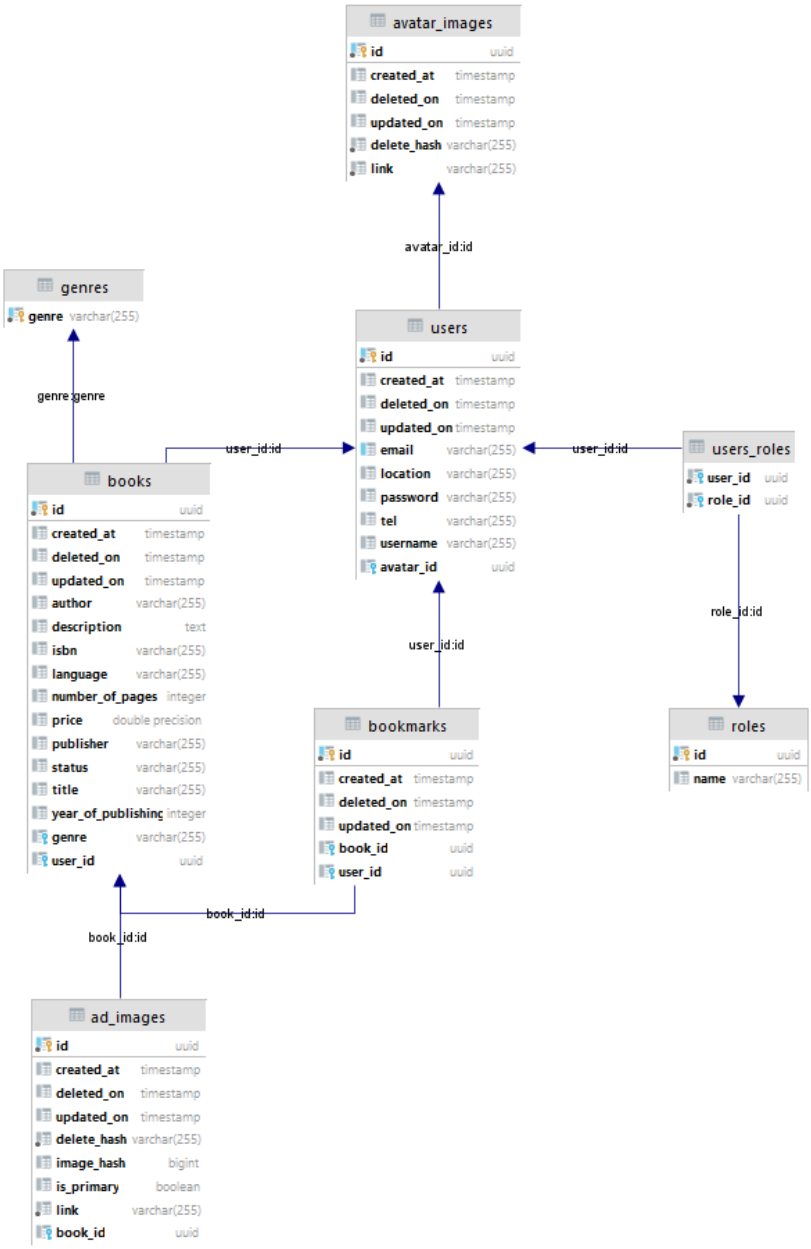

Рис. 3.6 Схема таблиць бази даних

3.3.2 Створення компонентів React.js

Реакт компоненти бувають класові та функціональні. Сучасний React.js надає перевагу саме функціональним компонентам, тож саме такі я використовував в своєму застосунку. Традиційно компоненти поділяються на контейнерні компоненти та презентаційні компоненти. Контейнерна компонента може отримувати доступ до стану застосунку
та до actions. Контейнерна компонента отримує дані необхідні для презентаційних компонент та передає ці дані всередину презентаційних компонент через пропси. В свою чергу презентаційні компоненти отримують свої дані та колбеки лише через пропси, таким чином вони отримують лише необхідні для відображення дані. До контейнерних компонентів я вдніс всі компоненти, що відповідають сторінкам та обгорткам над сторінками:

- AddBookPage - сторінка додавання нового оголошення з продажу книг. Містить в собі логіку додавання фотографій до оголошень
- BookPage - сторінка відображення інформації про книгу. Містить в собі презентаційні компоненти BookView та BookEditor
- BookmarksPage - сторінка відображення оголошень які були помічені
- користувачем закладками. Містить в собі презентаційні компоненти BookCard.
- BooksFeedPage - сторінка перегляду та пошуку оголошень. Містить в собі презентаційні компоненти Filter та BookCard
- ContentContainer - компонент обгортка над сторінками містить. Використовується для єдиного стилю сторінок з іконкою користувача та назвою сторінки яка визначається за шляхом відкритої сторінки.
- LandingPage - сторінка що містить форми реєстрації та входу в систему.
- ModerationPage - сторінка перегляду оголошень, що потребують модерації.
- Містить в собі презентаційні компоненти ModerationBookCard.
- PrivateRoute - компонент обгортка над компонентом ContentContainer для

- доступу до сторінки лише авторизованих користувачів.
- ProfilePage - сторінка профілю. Містить в собі презентаційні компоненти ProfileEditor та ProfileView
- PublicRoute - компонент обгортка над компонентом LandingPage для доступу до сторінки лише не авторизованих користувачів.
- Routing - компонент конейнер в якому відбувається маршрутизація застосунку.
- UserBooksPage - сторінка перегляду оголошень певного користувача

До презентаційних компонентів належать такі компоненти:

- BookCard - компонент для відображення оголошень у списку
- AvatarProceeder - компонент для обробки фотографій профілю(вибору та обтинання)
- Filter - компонент з пошуковим полем та фільтрами за параметрами оголошень.
- ModerationBookCard - компонент для відображення оголошень, що потребують модерування у списку.
- ProfileEditor - компонент редагування профілю
- ProfileView - компонент перегляду профіля користувача
- BookEditor - компонент редагування оголошення.
- BookView - компонент перегляду оголошення.

Основним компонентом є App він вміщує в собі під’єднання стану застосунку, маршрутизації та базового компоненту спливаючих сповіщень.

3.3.3 Створення серверної частини

Кореневий пакет застосунку має назву ua.slobodianyk.bouquiniste.
До базового проекту Spring Boot були додані starter-пакети: spring-boot-starter-jpa, spring-boot-starter-security, spring-boot-starter-web.
Першочергово була реалізована безпекова сторона застосунку.

Був реалізований пакет ua.slobodianyk.bouquiniste.auth, що містить в собі: DTO необхідні для реєстрації та входу користувача до системи, контроллер, що відповідає за обробку входу в систему та реєстрації користувачів, а також сервіс для обробки токена. Також був реалізований пакет ua.slobodianyk.bouquiniste.config, в якому розміщується фільтр JWT токенів, а також так налаштування безпеки як: white-list шляхів для яких не потрібна автентифікація та авторизація, час придатності токена та інші Для імплементації бізнес логіки реєстрації та входу в систему, також був реалізований пакет ua.slobodianyk.bouquiniste.user. Цей пакет містить в собі модель User, що відповідає таблиці users в базі даних. Пакет також містить контроллер та сервіс для роботи з сутністю користувача. Також був доданий ua.slobodianyk.bouquiniste.image. Він містить моделі AvatarImage,що використовуєтсья для зберігання фотографій профіля користувача, а також AdImage, яка містить інформацію про зображення додані до оголошень. В пакеті знаходиться клас Хешер, що здійснює обрахунок хеша для кожного зображення AdImage, перед його збереженням в базу даних. А також сервіс, що містить методи для пошуку схожих зображень в користувача, про які буде детальніше розказано в розділі 3.3.4. Пакет ua.slobodianyk.bouquiniste.book містить модель Book яка відповідає таблиці books в базі даних, а також бізнес логіку для роботи з сутністю книги. Для надання списку існуючих в системі жанрів був реалізований пакет ua.slobodianyk.bouquiniste.genre, що містить модель, що відповідає таблиці genres в базі даних. Щоб реалізувати зв’язок між книгою та користувачем, який би вказував на те, що користувач розмістив закладку на певній книзі, був реалізований пакет ua.slobodianyk.bouquiniste.bookmarks, який містить в собі модель Bookmark, яка відповідає таблиці bookmarks, а також містить бізнес логіку для роботи з закладками. Наприкінці роботи я додав реалізацію ролей для користувачів, щоб обмежувати

дозволені дії для різних груп користувачів.

3.3.3 Реалізація перевірки схожості зображень

Всередині концепції пошуку схожих зображень знаходиться алгоритм dHash (Difference hash)[5][6]. В основу алгоритму закладено обчислення напрямку градієнта зображення.

Для отримання результатів за цим алгоритмом, необхідно виконати три дії:

- Стиснути зображення до розмірів 9 пікселів в ширину та 8 пікселів у висоту.
- Перевести зображення у кольоровий тон grayscale.
- Ініціалізувати хеш значенням 0.
- Обчислити хеш обробленого зображення.

Для обрахунку хешу може бути декілька стратегій: діагональна, вертикальна та горизонтальна. Для реалізації в своєму додатку я обрав горизонтальний варіант. Починати обрахунок хешу таким способом потрібно з елемента (0,1) матриці значень пікселів зображення, що ми попередньо обробили. з цієї позиції ми повинні обійти матрицю порівнюючи значення яскравості поточного пікселя з тим що передує йому в рядку. У випадку якщо яскравість поточного пікселя більша за яскравість попереднього пікселя, поточному біту надається значення 1. При цьому на кожній ітерації ми робимо побітове зміщення значення хешу вліво на одиницю. Після підрахунку хешу зображення його значення зберігаються разом з посиланням на картинку.

При запиті на збереження нового оголошення від автора. Застосунок порівнює хеші попередньо збережених зображень за допомогою відстані Хеммінга та повертає

зображення які вважаються схожими на вже існуючі
в оголошеннях автора. У випадку якщо було знайдено схожі зображення оголошенню надається статус “Очікування модерації”, після чого воно стає доступним модератору
для перевірки.

3.4 Опис інтерфейсу застосунку

Після завантаження застосунку користувач має можливість увійти до системи.

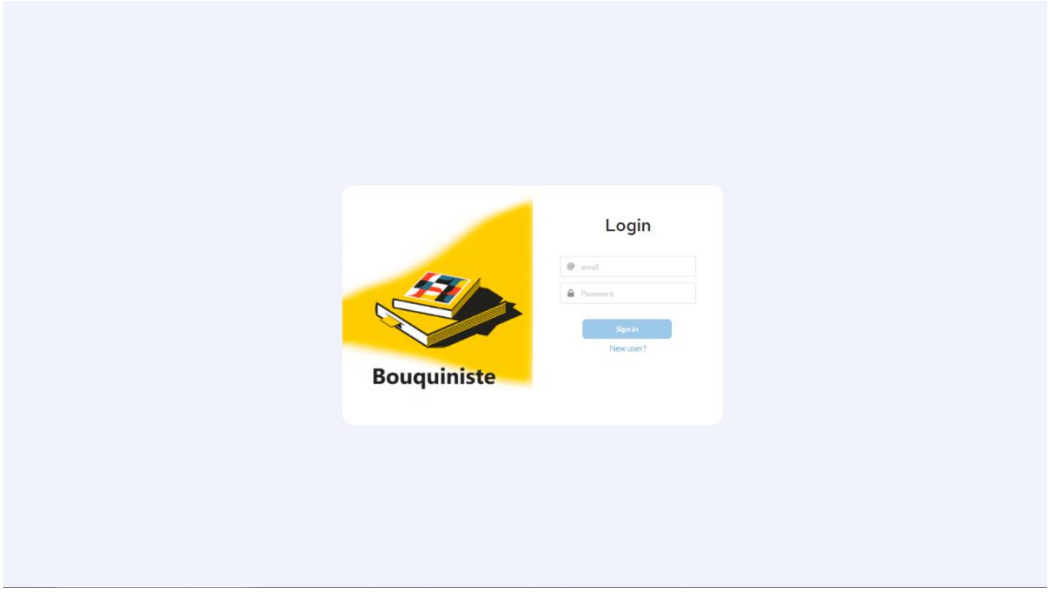

Рис. 3.7 Користувацький інтерфейс: входу до застосунку

Якщо ж у користувача відсутній акаунт він може натиснути на напис “New user?”, таким чином він перейде на сторінку реєстрації нового користувача.

Рис. 3.8 Користувацький інтерфейс реєстрації нового користувача

Після реєстрації або входу користувач потрапляє на сторінку зі списком створених оголошень.

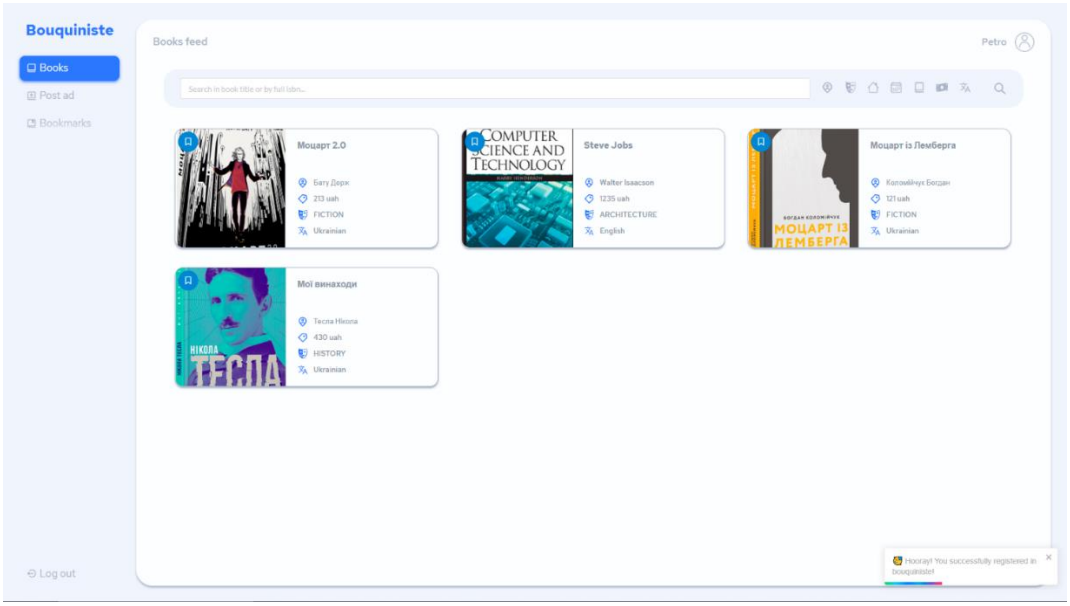

Рис. 3.9 Користувацький інтерфейс: список оголошень

На сторінці списку оголошень користувач має можливість відфільтрувати оголошення за параметрами.

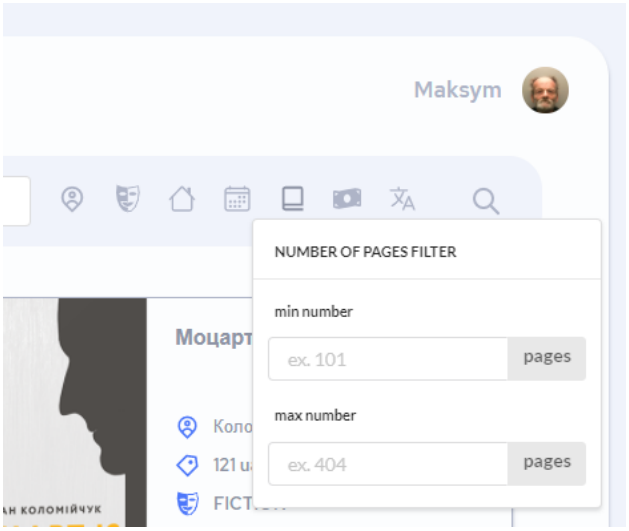

Рис. 3.10 Користувацький інтерфейс: фільтр типу значення від X до Y

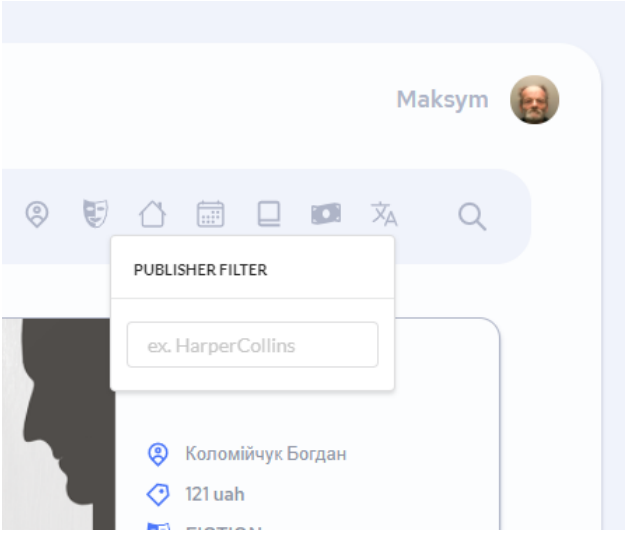

Рис. 3.11 Користувацький інтерфейс: фільтр типу значення

Натиснувши на назву книги користувач перейде на сторінку книги, де може отримати повну інформацію про книгу.

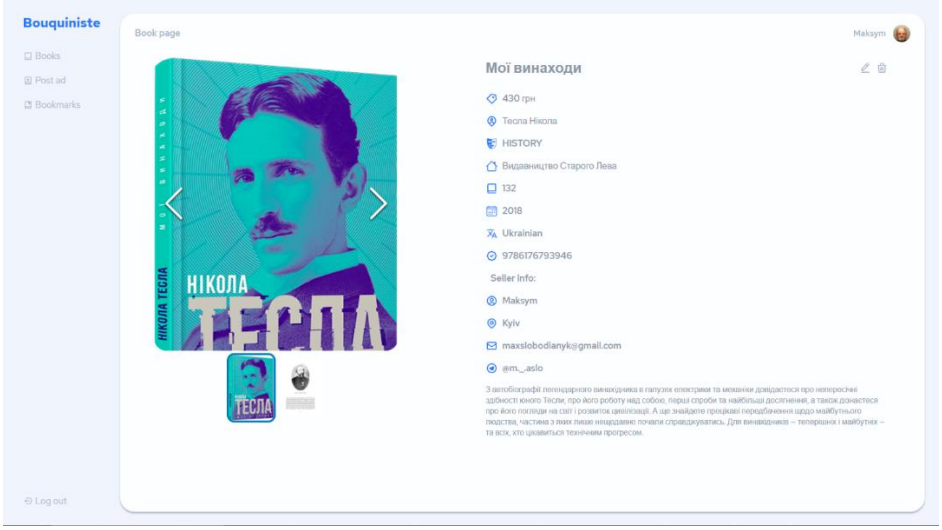

Рис. 3.11 Користувацький інтерфейс: сторінка оголошення

Натиснувши на фотографію її можна відкрити в більшому розмірі.

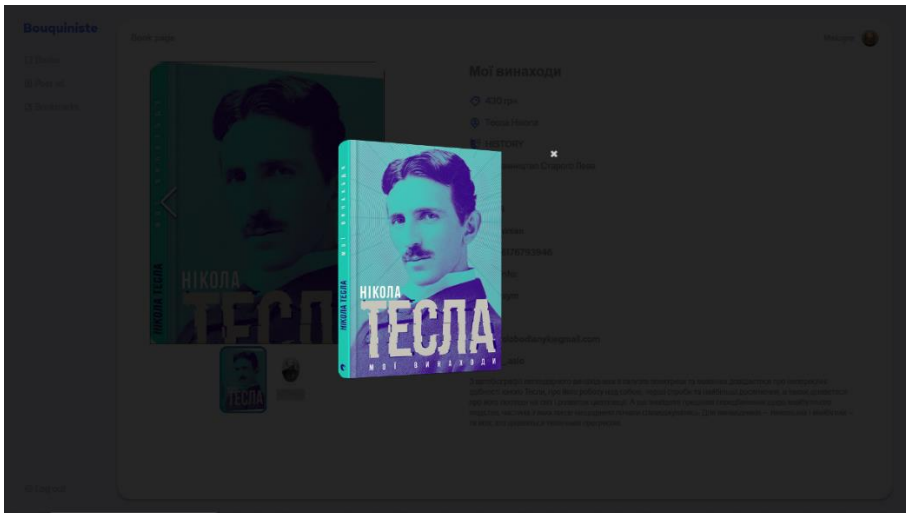

Рис. 3.11 Користувацький інтерфейс: перегляд фотографії

Натиснувши на кнопку редагування можна перейти на сторінку редагування оголошення

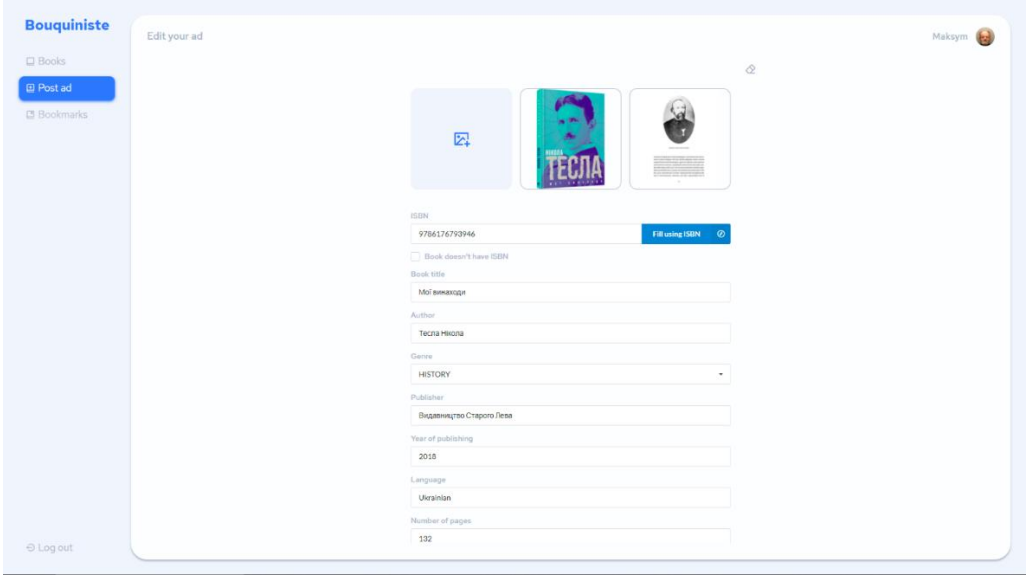

Рис. 3.12 Користувацький інтерфейс: редагування оголошення

Натиснувши на кнопку видалення, можна видалити оголошення.

Натиснувши на фотографію профіля користувача, можна перейти на сторінку профіля користувача.

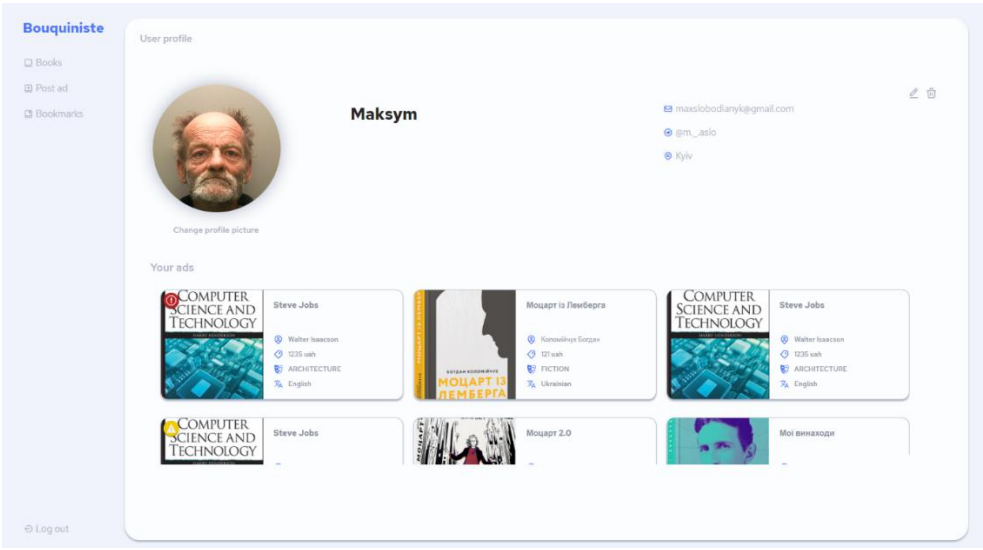

Рис. 3.13 Користувацький інтерфейс: перегляд профіля користувача

Натиснувши на кнопку видалення можна видалити власний профіль.

Натиснувши на кнопку редагування можна перейти на сторінку редагування профіля

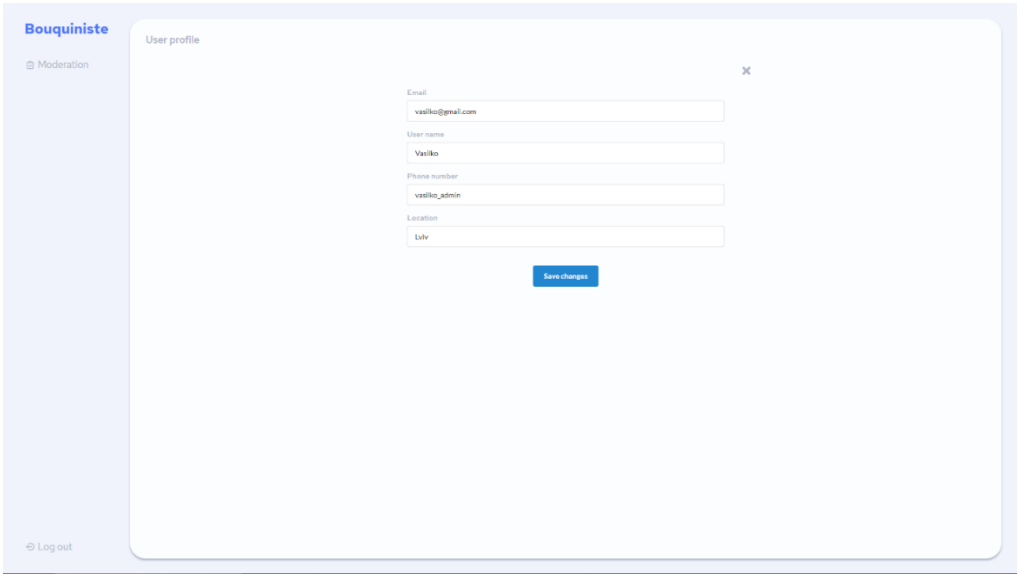

Рис. 3.13 Користувацький інтерфейс: редагування профіля користувача

Натиснувши на кнопку “Post ad” можна перейти на сторінку додавання оголошення.

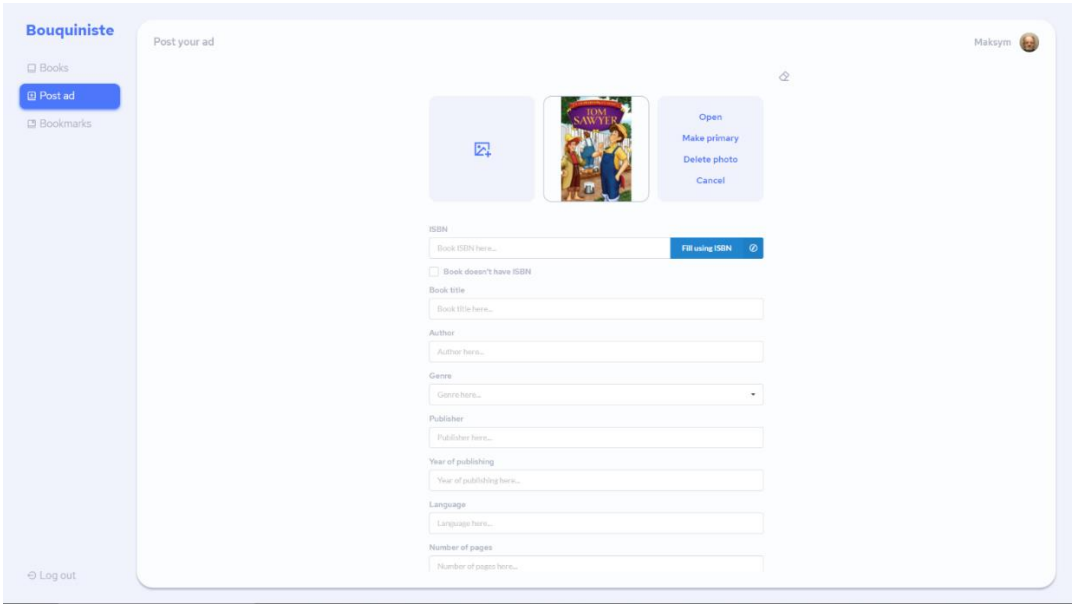

Рис. 3.13 Користувацький інтерфейс: додавання нового оголошення

Натиснувши на кнопку “Bookmarks” можна перейти на сторінку оголошень, що були помічені користувачем закладкою.

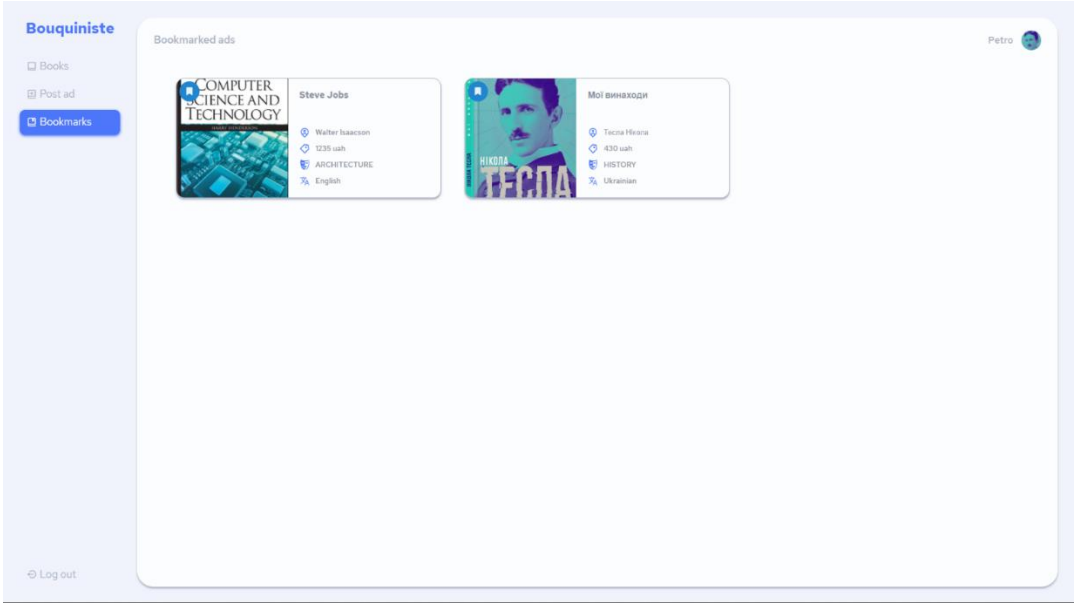

Рис. 3.13 Користувацький інтерфейс: сторінка оголошень помічених закладками

Натиснувши на кнопку “Moderation” можна перейти на сторінку оголошень, що потребують модерації.

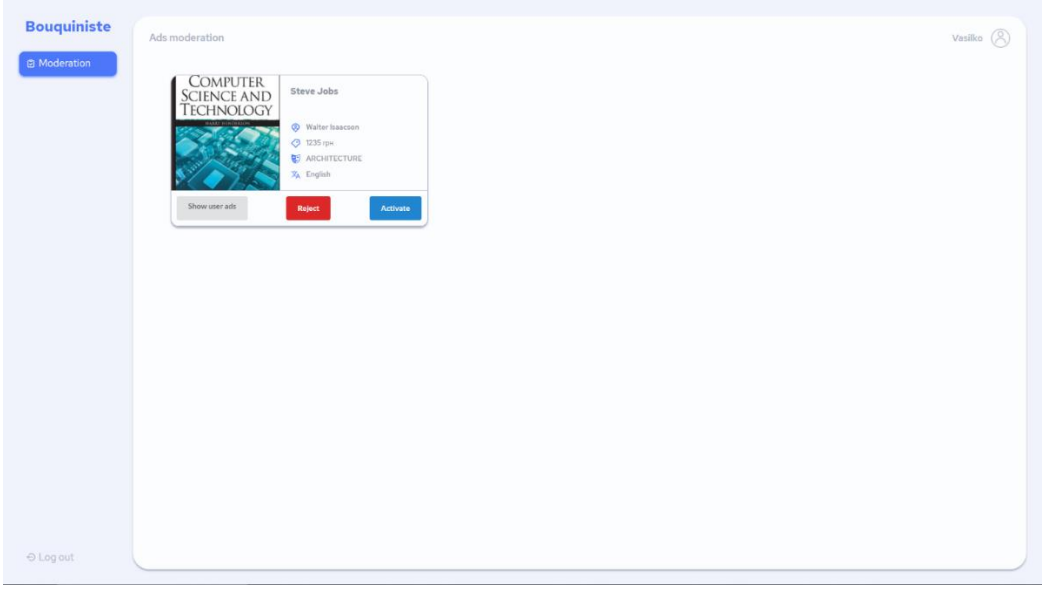

Рис. 3.13 Користувацький інтерфейс: сторінка оголошень, що потребують модерації

Натиснувши на кнопку “Reject” можна скасувати додавання оголошення.

Натиснувши на кнопку “Activate” можна підтвердити створення оголошення.

Натиснувши на кнопку “Show user ads” можна перейти на сторінку перегляду оголошень користувача, чиє оголошення виявилося підозрілим.

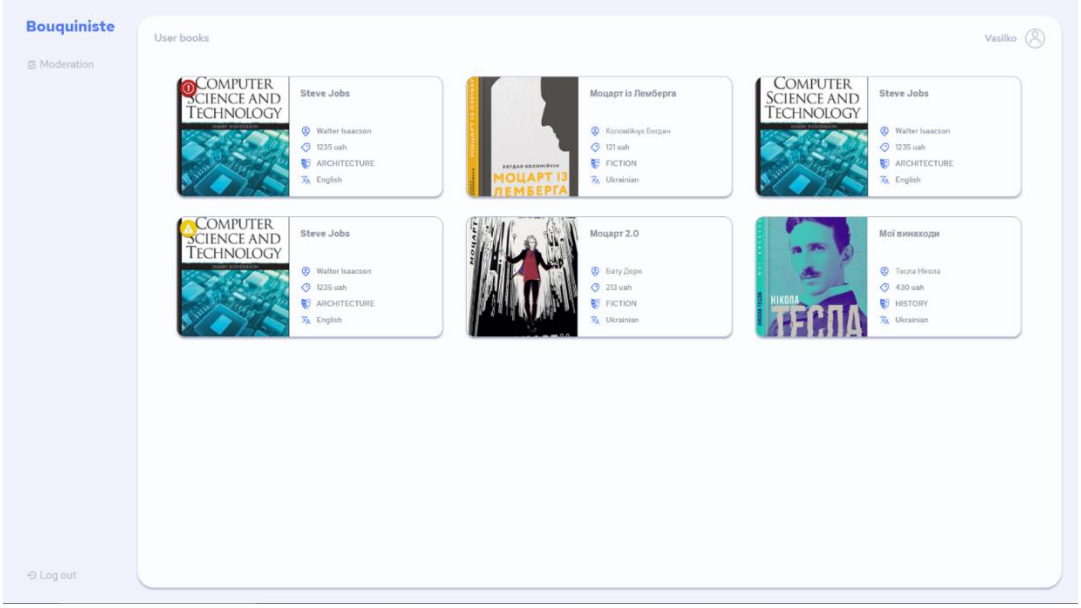

Рис. 3.13 Користувацький інтерфейс: сторінка оголошень користувача

Натиснувши на кнопку “Log out” можна вийти з системи.

Висновки

По закінченню роботи, було виконано всі поставлені задачі. Провівши опитування серед студентів вдалося визначити книжкові вподобання, а також проблеми з якими стикаються потенційні користувачі системи при отриманні нових книг. Також згідно опитування було визначено готовність опитаних купувати вживані книги задля зниження їх вартості. Проаналізувавши аналогічні системи було знайдено найнеобхідніший функціонал, який повинно містити MVP на базі цього аналізу було сформовано функціональні вимоги до додатку, а також прописані групи користувачів.

Також були досліджені та описані теоретичні відомості, що є пов’язаними з технологіями використаними під час розробки застосунку

В фінальній стадії роботи було створено MVP платформи для продажу вживаних книг. Розроблений застосунок наділений зручним функціоналом, що дозволяє не лише швидко знаходити книгу за заданими параметрами, проте також дозволяє швидко створювати оголошення, ввівши лише ISBN книги.

Список використаної літератури

1. Стаття газети The New York Times з приводу змін на книжковому ринку в період пандемії [Електронний ресурс] https://www.nytimes.com/2020/12/29/books/book-publishing-2020.html?fbclid=IwAR1mU6bQLz-qd28kXQryWWWjuyJop8SrEnrdBNHRN1KyPyACOXEu1KA6_80
2. Результати дослідження книжкового ринку українського інституту книги [Електронний ресурс] https://drive.google.com/file/d/1hARQYry6tR5eSJpb3jzHTXDJPyTY75d3/view
3. Дані проекту Ukrainian Reading and Publishing Data [Електронний ресурс] http://data.chytomo.com/chytannya-v-ukrayini/
4. Проведене опитування серед студентів, щодо їхніх книжкових уподобань [Електронний ресурс] https://docs.google.com/forms/d/1dDQyuypgax82jSnaYV6rXm8Fh2GAwKFjzFsZBB7jwns/edit#responses
5. Стаття про принцип роботи алгоритму dHash [Електронний ресурс] http://hackerfactor.com/blog/?/archives/529-Kind-of-Like-That.html
6. Стаття про методи пошуку зображень, що дублюються [Електронний ресурс] https://habr.com/ru/post/205398/
7. Інтернет сторінка з прикладом шифрування та дешифрування JWT токена[Електронний ресурс] https://jwt.io/
8. Стаття Скота Вамблера з детальним описом роботи ORM[Електронний ресурс] http://www.agiledata.org/essays/mappingObjects.html
9. Стаття на сайті IBM з поясненням роботи REST та RESTful веб-сервісів[Електронний ресурс] http://www.agiledata.org/essays/mappingObjects.html
10. Офіційна документація React.js[Електронний ресурс] https://reactjs.org/
11. Офіційна документація Spring Boot[Електронний ресурс] https://spring.io/projects/spring-boot
12. Офіційна документація PostgreSQL[Електронний ресурс] https://www.postgresql.org/docs/
13. Офіційна документація Redux[Електронний ресурс] https://redux.js.org/tutorials/essentials/part-1-overview-concepts

14. Офіційна документація Redux DevTools[Електронний ресурс] https://github.com/reduxjs/redux-devtools#documentation
15. Платформа розміщення онлайн оголошень OLX[Електронний ресурс] https://www.olx.ua/uk/
16. Платформа розміщення онлайн оголошень prom.ua[Електронний ресурс] https://prom.ua/
17. Instagram магазин з оголошеннями про продаж книг, що були у вжитку [Електронний ресурс] https://www.instagram.com/bookf_ua/