Національний технічний університет України

«Київський політехнічний інститут»

Факультет Інформатики та обчислювальної техніки

Кафедра Обчислювальної техніки

Розрахунково-графічна робота

З дисципліни: «Інтеграція програмних систем»

На тему: «Пошукова система для книг»

Виконали студенти 4-го курсу

ФІОТ групи ІО-41

Ковальчук А.Д.

Іванько Д.С.

Київ – 2017

**Вступ**

Об’єктом розробки даної розрахункової роботи є сайт з функціональністю пошуку книг за різними критеріями: назвою книги, ім’ям автора та жанрами.

Результат - готовий сайт, рушій якого написаний на мові програмування Java, а верстання – за допомогою мови розмітки гіпердектових документів HyperText Markup Language (HTML).

Графічний інтерфейс користувача — інтерфейс між комп'ютером і його користувачем, що використовує вкладки браузера, піктограми, меню, і вказівний засіб для вибору функцій та виконання команд. Відповідно, можна відкривати більше однієї сторінки сайту в браузері.

**1.Огляд MVC**

* 1. *Загальна характеристика*

Model - view - controller (MVC, «модель - представлення - поведінка», «модель-подання-контролер», «модель - вигляд - контролер») - схема використання кількох шаблонів проектування, за допомогою яких модель даних програми, користувальницький інтерфейс і взаємодія з користувачем розділені на три окремих компонента таким чином, щоб модифікація одного з компонентів виливала на інші мінімально. Дана схема проектування часто використовується для побудови архітектурного каркаса, коли переходять від теорії до реалізації в конкретній предметній області.

В оригінальній концепції була описана сама ідея і роль кожного з елементів: моделі, вигляду і контролера. Але зв'язки між ними були описані без конкретизації. Крім того, розрізняли дві основні модифікації:

* Пасивна модель - модель не має жодних способів впливати на вигляд або контролер, і використовується ними як джерело даних для відображення. Всі зміни моделі відслідковуються контролером і він же відповідає за перемальовування уявлення, якщо це необхідно. Така модель частіше використовується в структурному програмуванні, так як в цьому випадку модель являє просто структуру даних, без методів їх обробних.
* Активна модель - модель оповіщає уявлення про те, що в ній відбулися зміни, а вигляди, які зацікавлені в оповіщенні, підписуються на ці повідомлення. Це дозволяє зберегти незалежність моделі як від контролера, так і від вигляду.

Класичною реалізацією концепції MVC прийнято вважати версію саме з

активною моделлю.

* 1. *Структура*

Концепція MVC дозволяє розділити дані, подання та обробку дій користувача на три окремих компонента:

● Модель (англ. Model). Модель зберігає дані і методи роботи з цими даними, реагує на запити, змінюючи свій стан. Не містить інформації, про візуалізацію.

● Вигляд, представлення (англ. View). Представлення служить для відображення інтерфейсу додатку. Часто це форма (вікно) з графічними елементами. Інтерфейс користувача зазвичай створюється на основі даних моделі. Прикладом може служити представлення для редагування таблиці Products, яке містить текстові поля, списки, що розкриваються і прапорці, значення яких засновані на поточному стані об'єкта Product.

● Контролер (англ. Controller). Контролери здійснюють взаємодію з користувачем, роботу з моделлю, а також вибір представлення, що відображає користувальницький інтерфейс. У додатку MVC представлення лише відображають дані , а контролер обробляє дані, що вводяться і відповідає на дії користувача. Наприклад, контролер може обробляти значення запиту і передавати їх в модель, яка може використовувати ці значення для відправлення запиту в базу даних.

Важливо відзначити, що як вигляд, так і контролер залежать від моделі. Але модель не залежить ні від представлення, ні від контролера. Це дозволяє будувати модель незалежно від візуального представлення, а також створювати кілька різних виглядів для однієї моделі.

Шаблон MVC дозволяє створювати додатки , різні аспекти яких (логіка введення, бізнес-логіка і логіка інтерфейсу) розділені, але досить тісно взаємодіють один з одним. Ця схема вказує розташування кожного виду логіки у додатку. Інтерфейс користувача розташовується в представленні. Логіка введення розташовується в контролері. Бізнес-логіка знаходиться в моделі. Це розділення дозволяє працювати зі складними структурами при створенні програми, так як забезпечує одночасну реалізацію тільки одного аспекту. Наприклад, розробник може сконцентруватися на створенні подання окремо від бізнес-логіки.

Зв'язок між основними компонентами додатку MVC також полегшує паралельну розробку. Наприклад, один розробник може створювати уявлення, інший - логіку контролера, а третій - бізнес-логіку моделі.

* 1. *Шаблони програмування, що використовуються в MVC*

Для реалізації схеми Model - View - Controller використовується досить велика кількість шаблонів проектування (залежно від складності архітектурного рішення), основні з яких «спостерігач», «стратегія», «компонувальник».

Найбільш типова реалізація відокремлює представлення від моделі шляхом встановлення між ними протоколу взаємодії, використовуючи апарат подій (підписка/сповіщення). При кожній зміні внутрішніх даних в моделі вона оповіщає всі залежні від неї представлення, і вони оновлюються. Для цього використовується шаблон «спостерігач». При обробці реакції користувача представлення вибирає, залежно від потрібної реакції, потрібний контролер, який забезпечить той чи інший зв'язок з моделлю. Для цього використовується шаблон «стратегія», або замість цього може бути модифікація з використанням шаблону «команда». Крім того, можуть використовуватися й інші шаблони проектування, наприклад, «фабричний метод», який дозволить задати за замовчуванням тип контролера для відповідного представлення.

* 1. *Завдання MVС*

Основна мета застосування цієї концепції полягає в відокремленні бізнес-логіки (моделі) від її візуалізації (представлення, виду). За рахунок такого поділу підвищується можливість повторного використання. Найбільш корисне застосування даної концепції в тих випадках, коли користувач повинен бачити ті ж самі дані одночасно в різних контекстах та/або з різних точок зору. Зокрема, виконуються наступні завдання:

1. До однієї моделі можна приєднати кілька представлень, при цьому не зачіпаючи реалізацію моделі. Наприклад, деякі дані можуть бути одночасно представлені у вигляді електронної таблиці, гістограми і кругової діаграми.
2. Не торкаючись реалізації представлень, можна змінити реакції на дії користувача (натискання на кнопку, введення даних), для цього досить використовувати інший контролер.
3. Ряд розробників спеціалізується тільки в одній з областей: або розробляють графічний інтерфейс, або розробляють бізнес-логіку. Тому можна досягти того, що програмісти, які займаються розробкою бізнес-логіки (моделі), взагалі не будуть знати про те, який вигляд буде використовуватися.
   1. *Java*

В даній розрахунковій роботі було побудовано рушій сайту на мові програмування Java із застосуванням бібліотек org.**springframework** та org.**thymeleaf.** На мою думку, вони найбільш зручні для побудови сайтів на Java, якщо найважливішим є функціонал та оформлення.

* 1. *Використання концепцій MVC в даній РГР*

В цій роботі я застосував саме модель MVC, оскільки виконання програми умовно поділяється на три частини: модель даних (база даних), вигляд даних (графічний інтерфейс користувача) та керування (логіка програми). Це зроблено для того, щоб відокремлення даних (моделі) від графічного інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

**2.Проектування програмного додатку**

*2.1 Прецеденти*

Користувач може шукати книги на свій розсуд: за назвою книги, ім’ям письменника чи поета, який видав свою збірку чи за жанрами, до яких обов’язково має належати будь-яка книга з представленого каталогу; може заходити на сторінку книги чи автора, щоб більше дізнатися про них; може обрати окремий жанр і подивитися всі існуючі на сайті книги, які належать даному жанру; може поповнювати бібліотеку книг своїми екземплярами. Можливості користувача наведені на рисунку 2.1.



Рис. 2.1. Можливості користувача

Користувач може увійти до каталогу з книгами, натиснувши кнопку «Всі книги», як показано на рисунку 2.2.

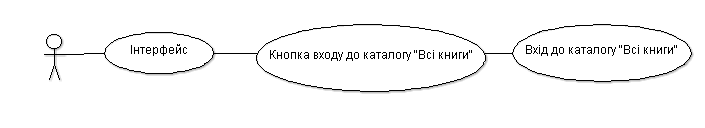


Рис. 2.2. Вхід до каталогу «Всі книги»

Користувач може увійти до каталогу з авторами, натиснувши кнопку «Всі автори», як показано на рисунку 2.3.

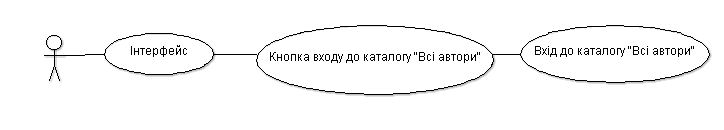


Рис. 2.3. Вхід до каталогу «Всі автори»

Користувач може додати книги, яких ще немає в каталозі і які йому цікаві, натиснувши на кнопку «Додати книги», як показано на рис. 2.4.

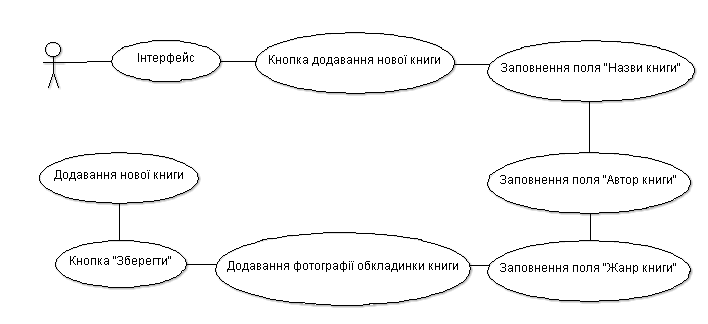


Рис. 2.4. Додавання книги до каталогу

Користувач за допомогою пошукової системи може знайти книги, яка йому цікава, натиснувши на кнопку «Пошук» і обрати вкладку «Книги», заповнивши деякі або всі із доступних полей. Після натискання на кнопку «Знайти», система запропонує йому всі доступні результати за критеріями, які задав користувач. Схема пошуку книг показана на рис. 2.5.

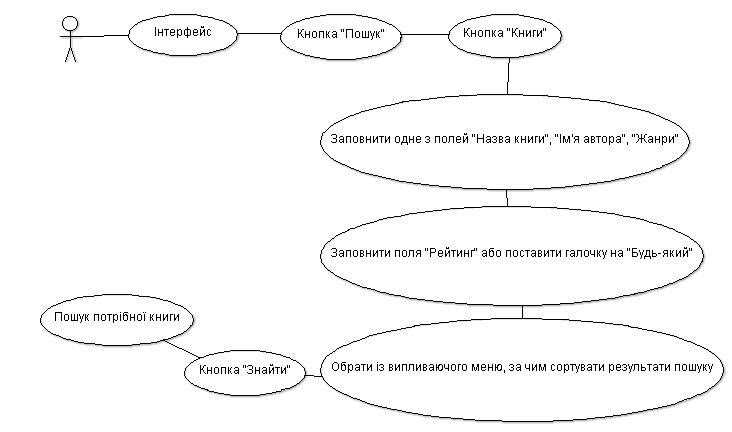


Рис. 2.5. Пошук книг за даними, які вказав користувач

Також користувач може здійснити пошук за авторами, натиснувши на кнопку «Пошук», обравши вкладку «Автори», і заповнивши одне доступне поле – Ім’я автора. Після натискання на кнопку «Знайти», система покаже всі можливі результати за даними, які вказав користувач. Схема такої взаємодії показана на рис. 2.6.

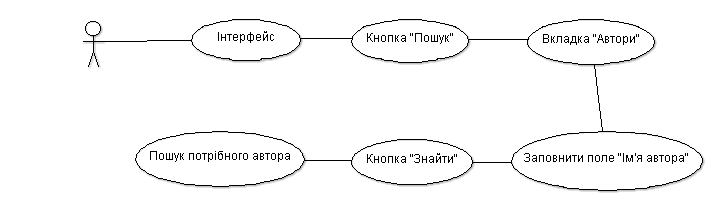


Рис. 2.6. Пошук автора за іменем, вказаного користувачем

За допомогою системи, написаної мною, можна виділити книги, які належать до конкретного з жанрів. Для отримання такого результату, потрібно опуститися вниз на головній сторінці і обрати жанр запропонований системою. Результатом виконання такої дії буде група книг, які належать до жанру, якого вибрав користувач. Приклад можна побачити на рис. 2.7.

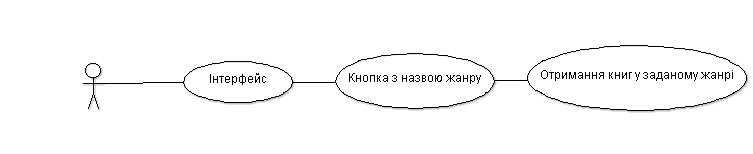


Рис. 2.7. Виділення книг, які належать до конкретного жанру

Користувач може дізнатися більше про обрану книгу, якщо натисне на кнопку «Прочитати більше». Після цього система покаже текст про дану книгу, як показано на рис. 2.8.

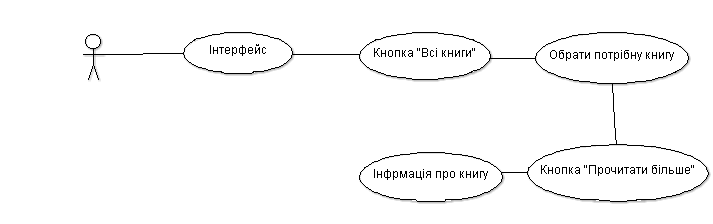


Рис. 2.8. Отримання детальнішої інформації про обрану книгу

Користувач може дізнатися більше про обраного автора, якщо натисне на кнопку «Прочитати більше». Після цього система покаже інформацію про обраного користувачем автора, як показано на рис. 2.9.



Рис. 2.9.Отримання детальнішої інформації про обраного автора

*2.2 Ескізи працездатності програми*



Рис. 2.10. Головна сторінка екрану



Рис. 2.11. Відображення на головній сторінці книг та авторів

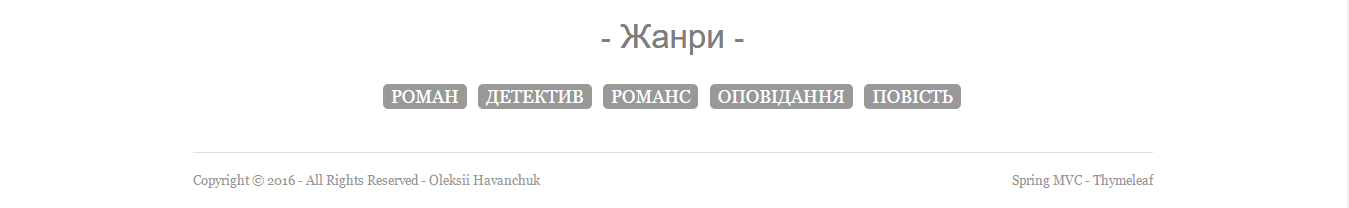


Рис. 2.12. Відображення на головній сторінці жанрів книг

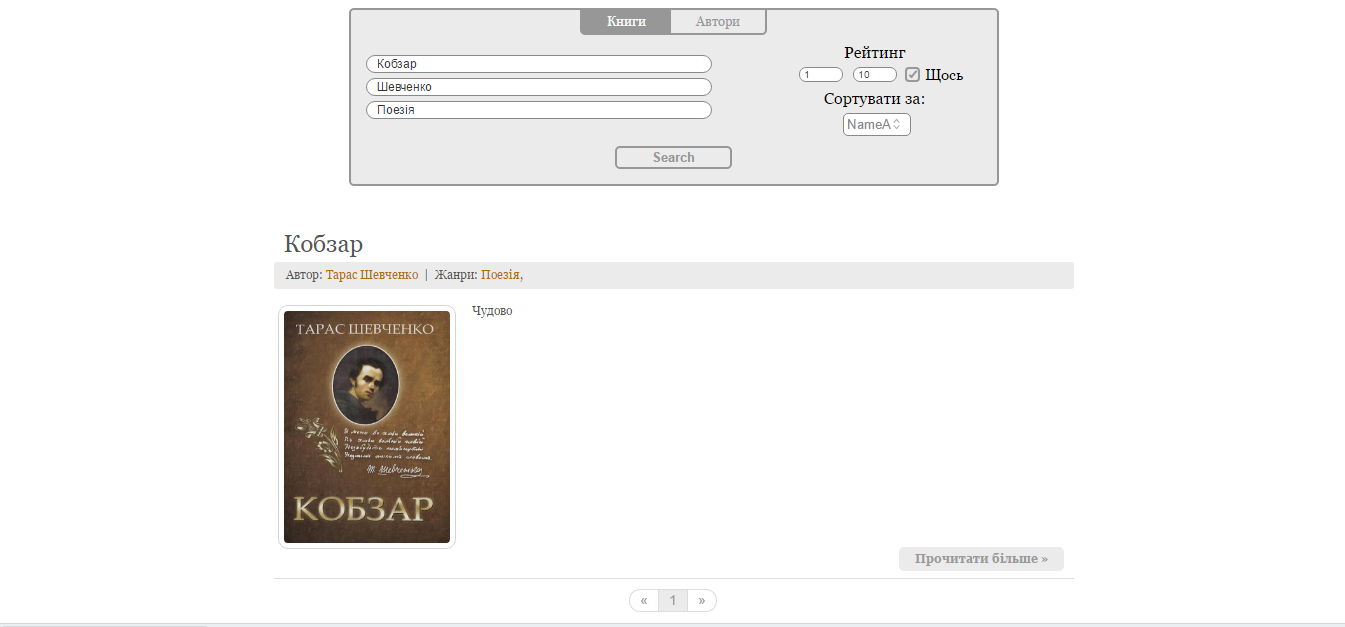


Рис. 2.13. Демонстрація пошукової системи книг

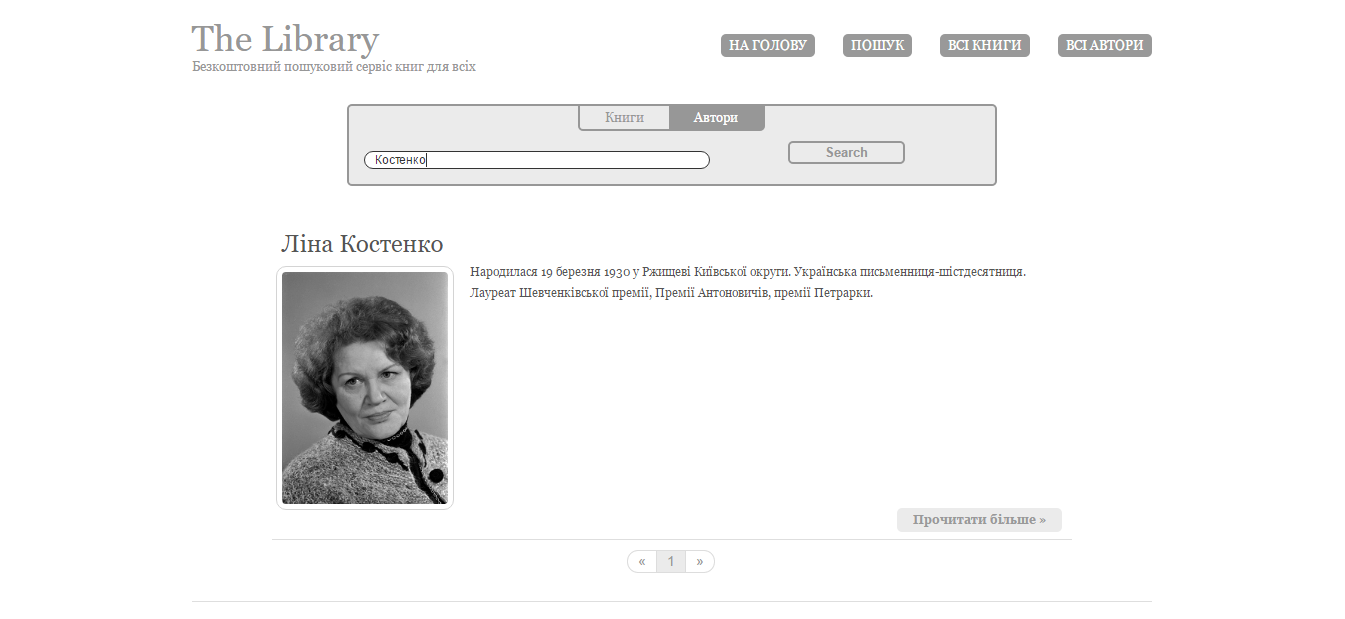


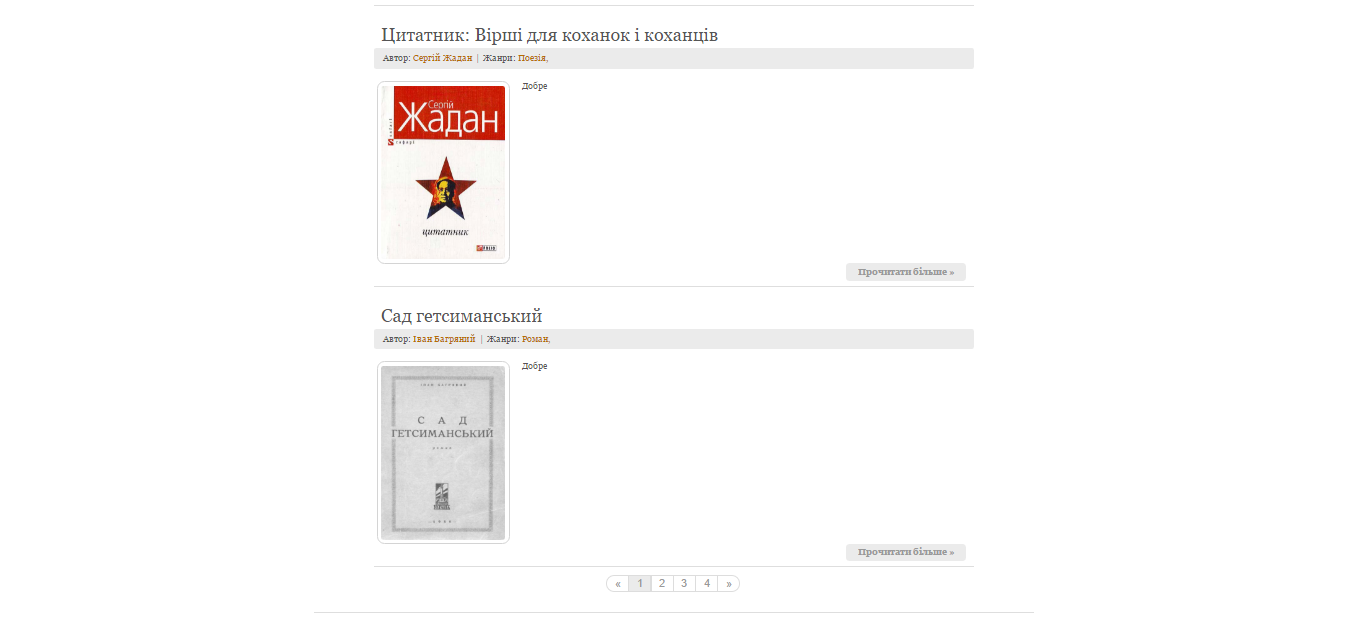
Рис. 2.14. Демонстрація пошукової системи

Рис. 2.15. Ескіз розділу «Всі книги»

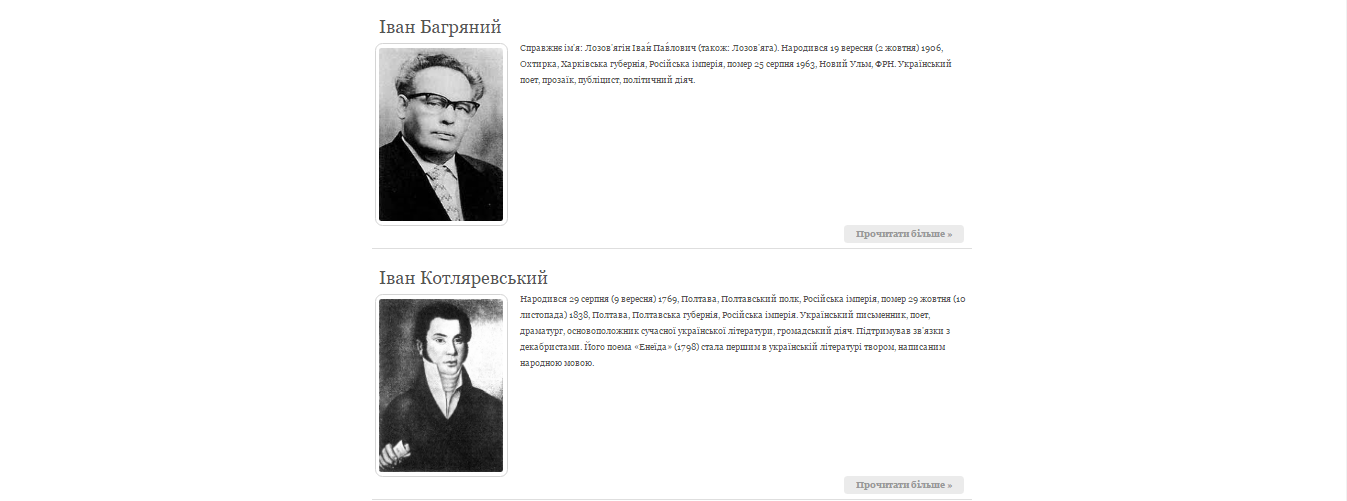


Рис. 2.16. Ескіз розділу «Всі автори»

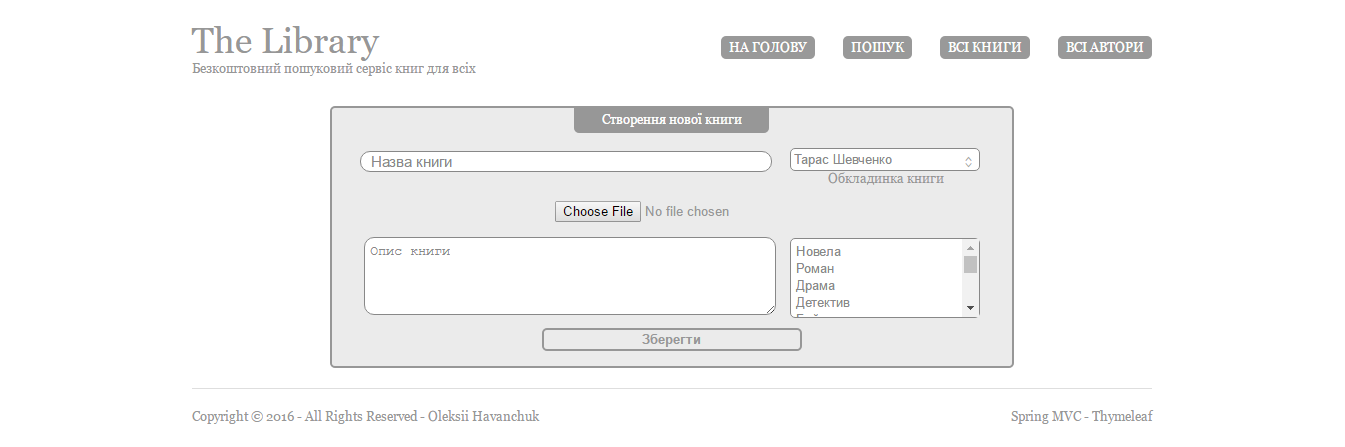


Рис.2.17. Демонстрація екрану додавання нової книги

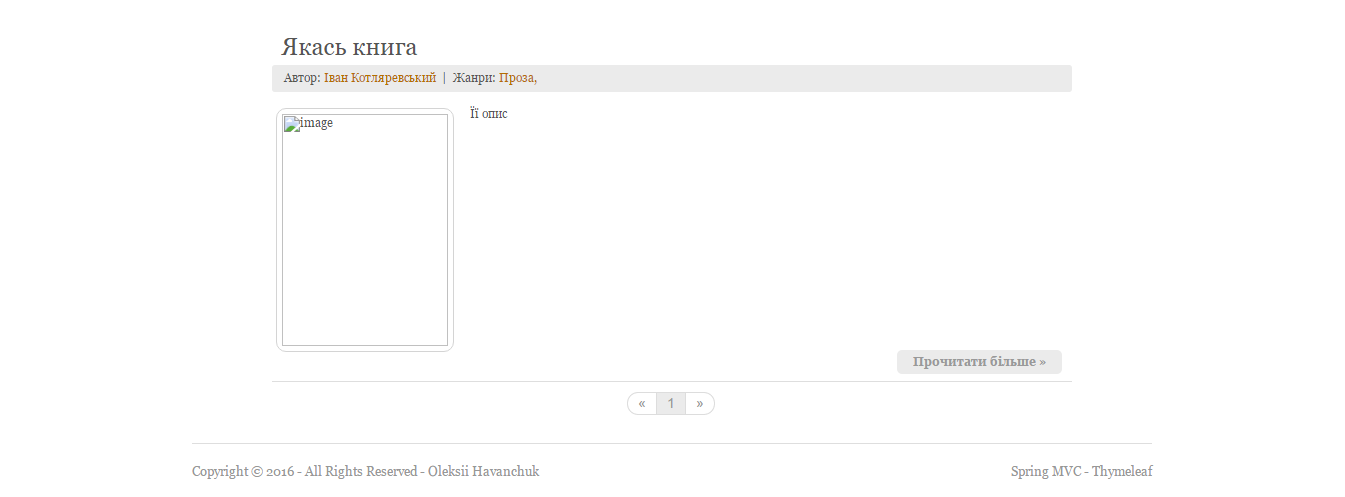


Рис. 2.18. Новостворена користувачем книга

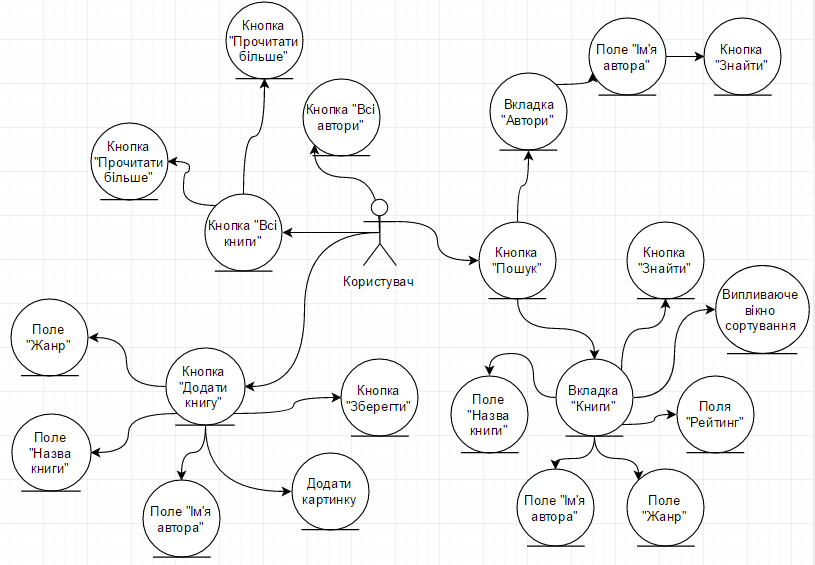


Рис.2.19. Діаграма граничних класів

**3.Розробка програмного продукту**

*3.1 Діаграма класів*

Згідно шаблону MVC, виконання програми ділиться на три частини: модель, інтерфейс, контроль. Створена Java-програма – частина контролю. Це рушій сайту і його логіка. На рис. 3.1 зображена діаграма класів програми на Java.

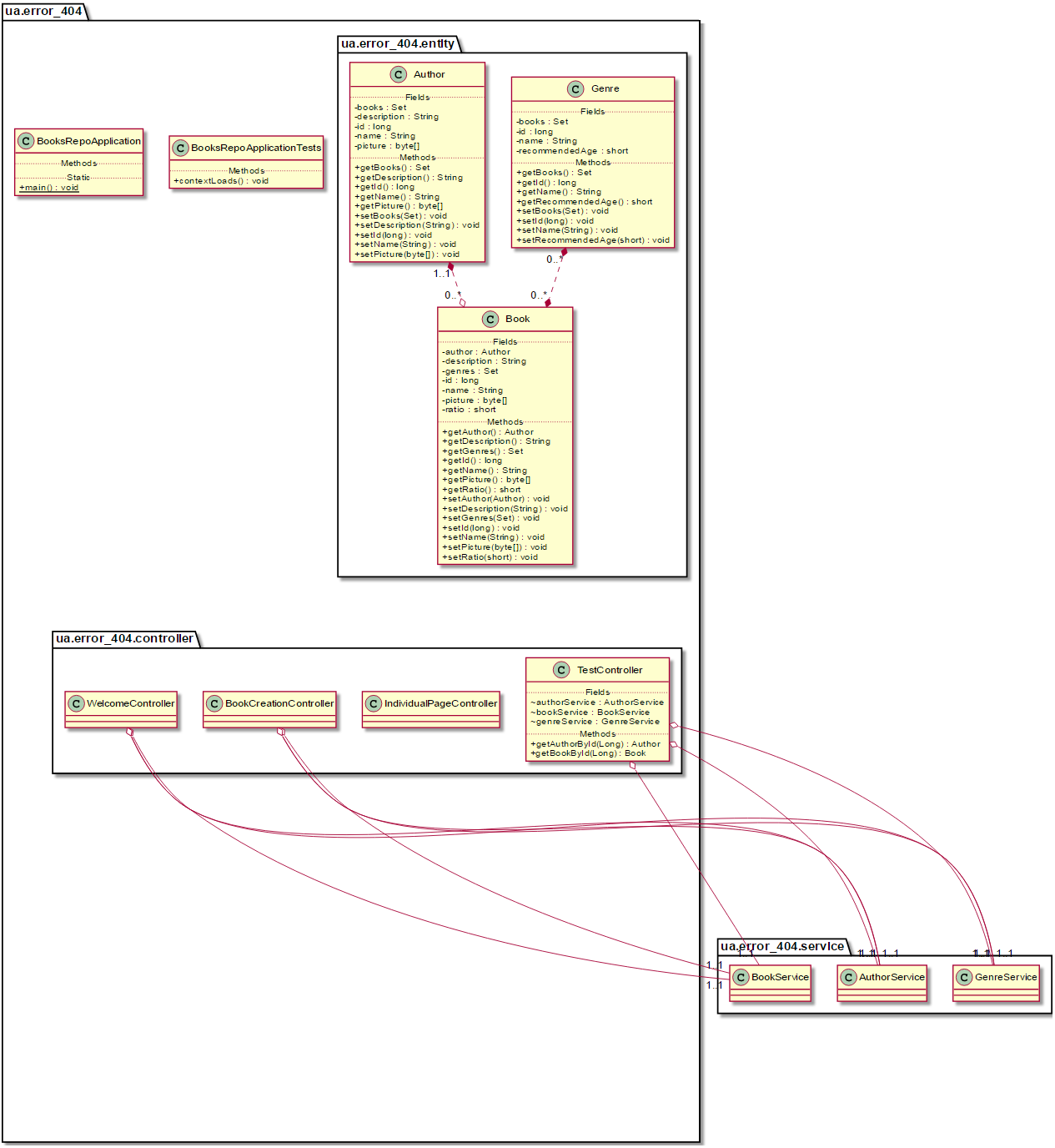


Рис. 3.1. Діаграма класів

*3.2 Опис Model*

Класи entity - Author, Book, Genre, це класи сутностей. Вони описують модель.

Поля класу Author: id (номер в списку БД), name (ім’я автора), description (опис автора, його кортка біографія), picture (фото або портрет автора), books (книги, які належать цьому автору).  
 Поля класу Book: id (номер в списку БД), name (назва книги), author (автор, який написав цю книгу), picture (обкладинка книги), genres (жанр книги), description (опис книги).

Поля класу Genre: id(номер в списку БД), name (назва жанру), recommendedAge (рекомендований вік для читання книги), book (книга, яка відноситься до заданого жанру).

База даних, як відомо, служить засобом зберігання моделей.

*3.3 Опис View*

Для опису вигляду інтерфейсу я використав HTML. Візуальна складова сайту складається із п’яти документів:

● createBookPage розрахований для візуалізації сторінки створення книги на сайті;

● footer, найнижча частина на сторінках сайту, створений для відображення року створення сайту, несе в собі повідомлення про захищені права сайту на назву власника;

● header, найвища частина на сторінках сайту, створений для відображення логотипу; сайту, та який відповідає за розташування кнопок «На головну», «Пошук», «Всі книги» та «Всі автори»;

● index відображає по три екземпляри книг, три екземпляри авторів та п’ять екземпляри жанрів;

● searchPage візуалізовує пошукову систему сайту.

*3.4 Опис Control*

Я умовно розбив засоби контролю на дві частини: це контролери WEB і контролери бази даних. Контролери WEB відповідають за те, як система повинна обробляти http запити і як обробляти їх. Контролери БД відповідають за обробку запитів від користувача і відсилати вже оброблені запити назад користувачу.

Контролери WEB знаходяться в пакеті controller, тобто класи BookCreationController, SearchPageController, WelcomeController.

● BookCreationController відповідає за обробку http-запитів на сторінці додавання нової книги в каталог сайту;

● SearchPageController відповідає за обробку http-запитів на сторінці пошуку книг та авторів;

● WelcomeController відповідає за обробку http-запитів на головній сторінці;

● ImageController відповідає за обробку http-запитів від зображень.

Контролери БД знаходяться в двох пакетах: repository і service, тобто інтерфейси AuthorRepository, BookRepository, GenreRepository та класи AuthorService, BookService, GenreService відповідно.

● Інтерфейси AuthorRepository, BookRepository, GenreRepository створені для того, щоб використовувати їх у набагато більшій кількості класів, ніж є зараз, оскільки я збираюся працювати з цією програмою надалі і розвивати все більше ідей.

● Класи AuthorService, BookService, GenreService реалізовують всю логіку і працездатність програм.

**4. Тестування програмного забезпечення**

Щоб зайти на сайт, потрібно запустити сервер (поставити програму Java на виконання), після чого в адресному рядку браузера написати команду: ***localhost:8080***. Виконання даної процедури показано на рис. 4.1.

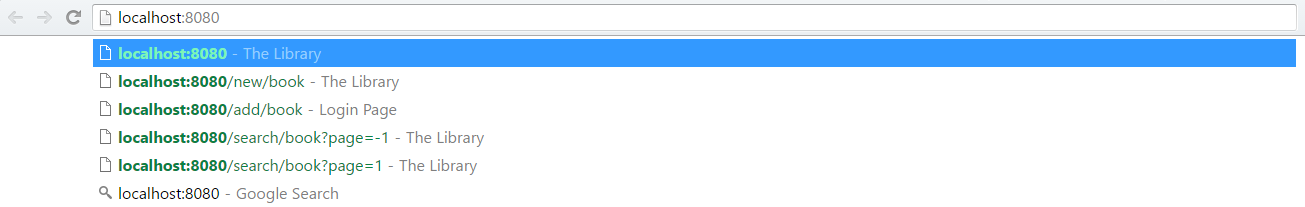


Рис. 4.1. Процедура входу на сайт

Після правильного вводу команди, користувач побачить головну сторінку сайту пошукової системи книг і авторів, яка зображене на рис. 4.2.



Рис. 4.2. Головна сторінка сайту

Користувач може обрати будь яку дію, зображену на рис. 4.3. Натиснувши на кнопку «На головну», користувач просто оновить сторінку. Ця кнопка корисна в тому випадку, якщо користувач знаходиться в іншому розділі сайту, до прикладу, у «Всі автори». Натиснувши кнопку «Пошук», користувач може здійснити пошук книг та авторів, задавши власні критерії пошуку. У розділі «Всі книги», користувач побачить всі книги, доступні в каталозі сайту, а натиснувши на кнопку «Всі автори», сайт запропонує користувачу оглянути всіх авторів, книги яких викладені на сайті.

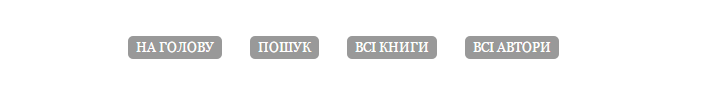


Рис. 4.3. Кнопки, які дозволяють перейти до різних розділів сайту

Користувач може додати книгу на сайт, якщо натисне на кнопку «Додати книгу». Після заповнення полей «Назва книги», «Опис книги», обирання із випливаючого меню автора книги, жанру книги і додавання обкладинки, книга з’явиться на сайті. Даний процес зображений на рис. 4.4 і рис. 4.5.

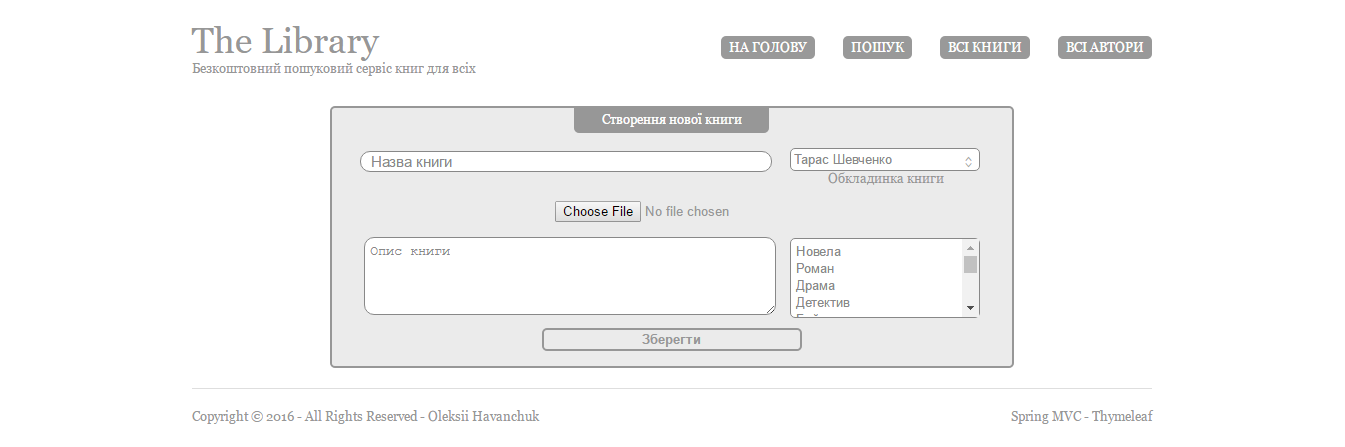


Рис. 4.4. Форма для додавання нової книги

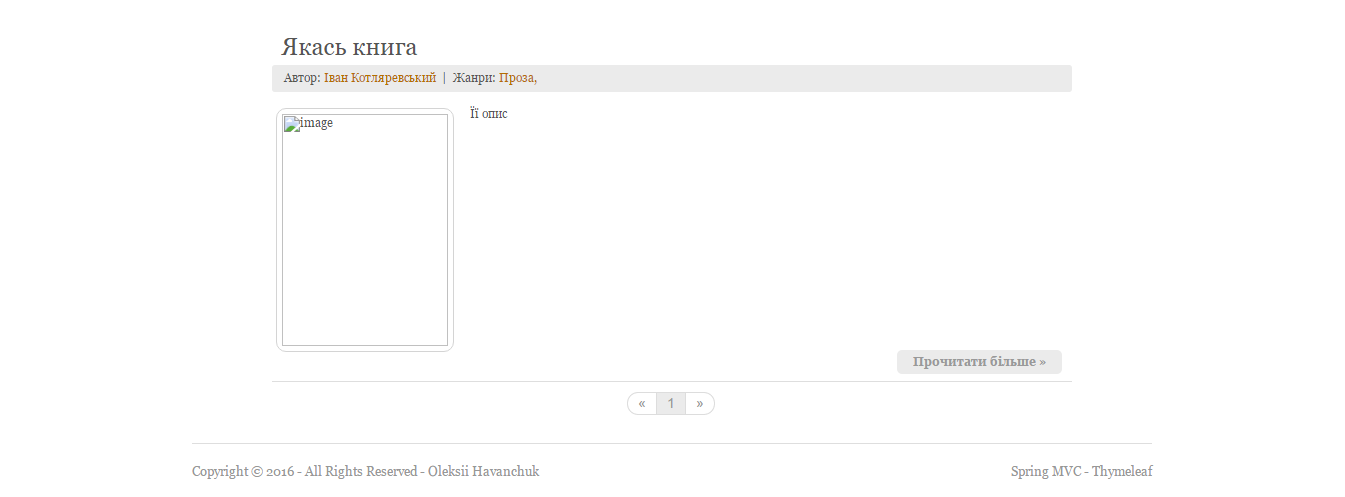


Рис. 4.5. Новостворена книга в пошуковому розділі

Користувач може подивитися на всі книги, викладені на сайті та всіх авторів після натискання на кнопки «Всі книги» та «Всі автори» відповідно. Вигляд розділів «Всі книги» та «Всі автори» на рис. 4.6 та рис. 4.7 відповідно.

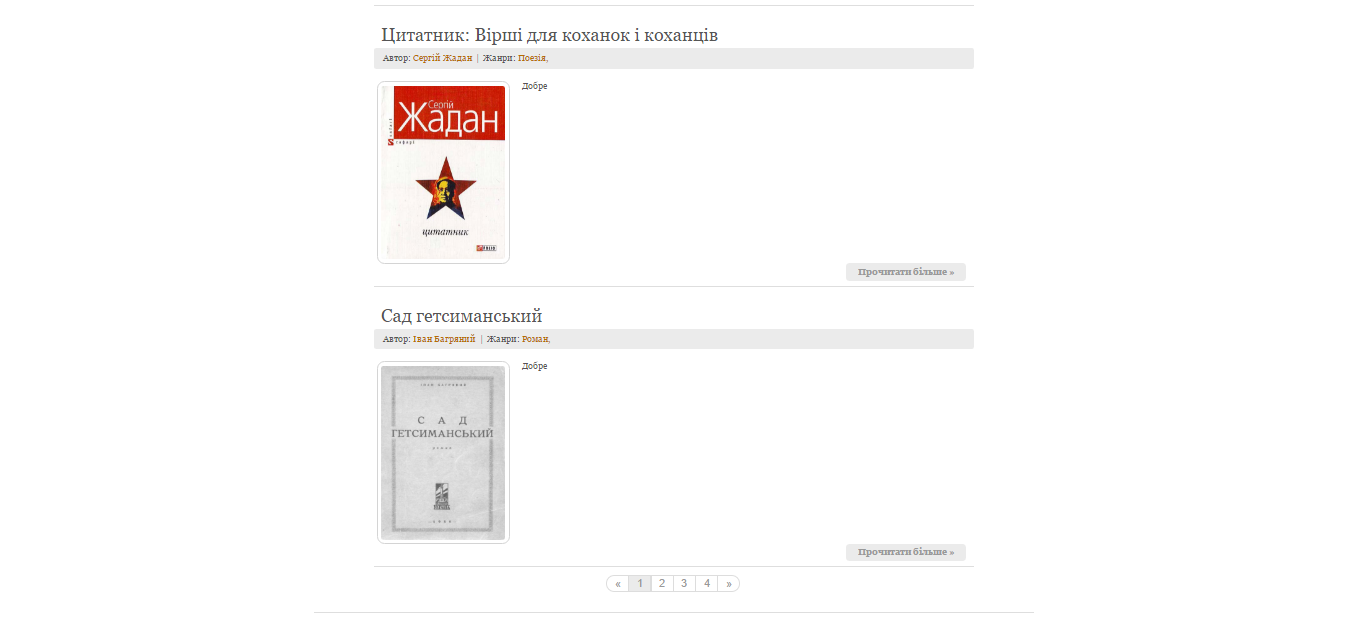


Рис. 4.6. Розділ «Всі книги»

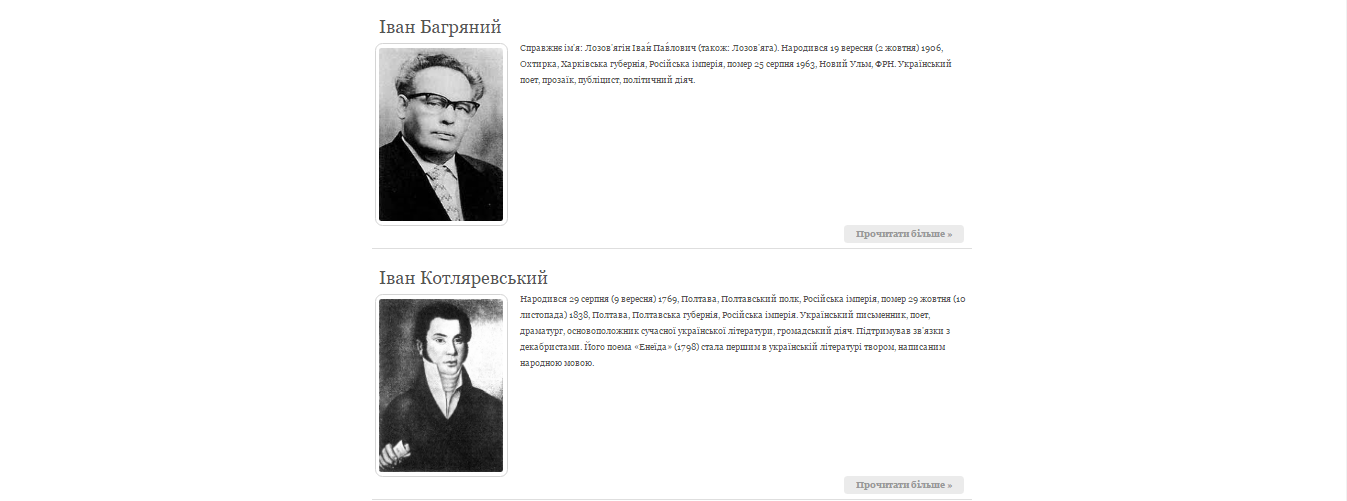


Рис. 4.7. Розділ «Всі автори»

Зайшовши в розділ «Пошук», користувач зможе знайти потрібну йому книгу або автора, заповнивши всі потрібні поля. Для книг це: назва книги, ім’я автора, жанр, рейтинг, за чим сортувати (назвою чи рейтингом). Для авторів це ім’я автора. Після натискання на кнопку «Знайти», система пропонує результати. Виконання даної процедури зображено на рис. 4.8 для книг і на рис. 4.9 для авторів.

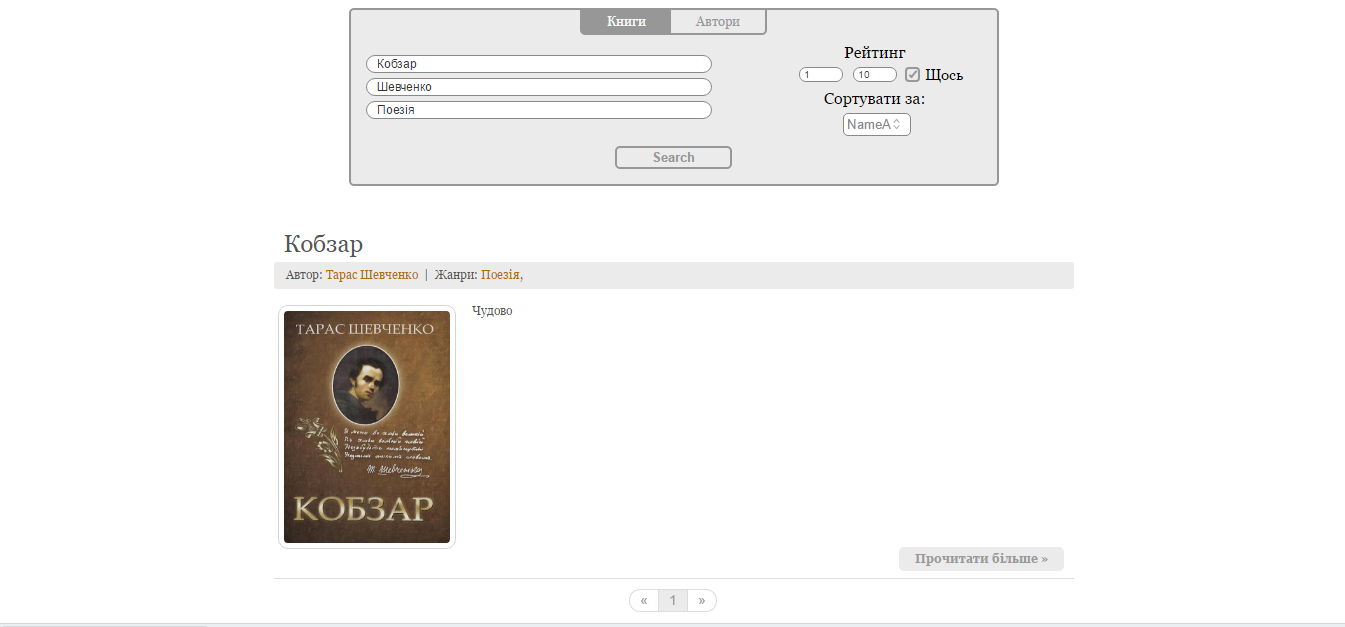


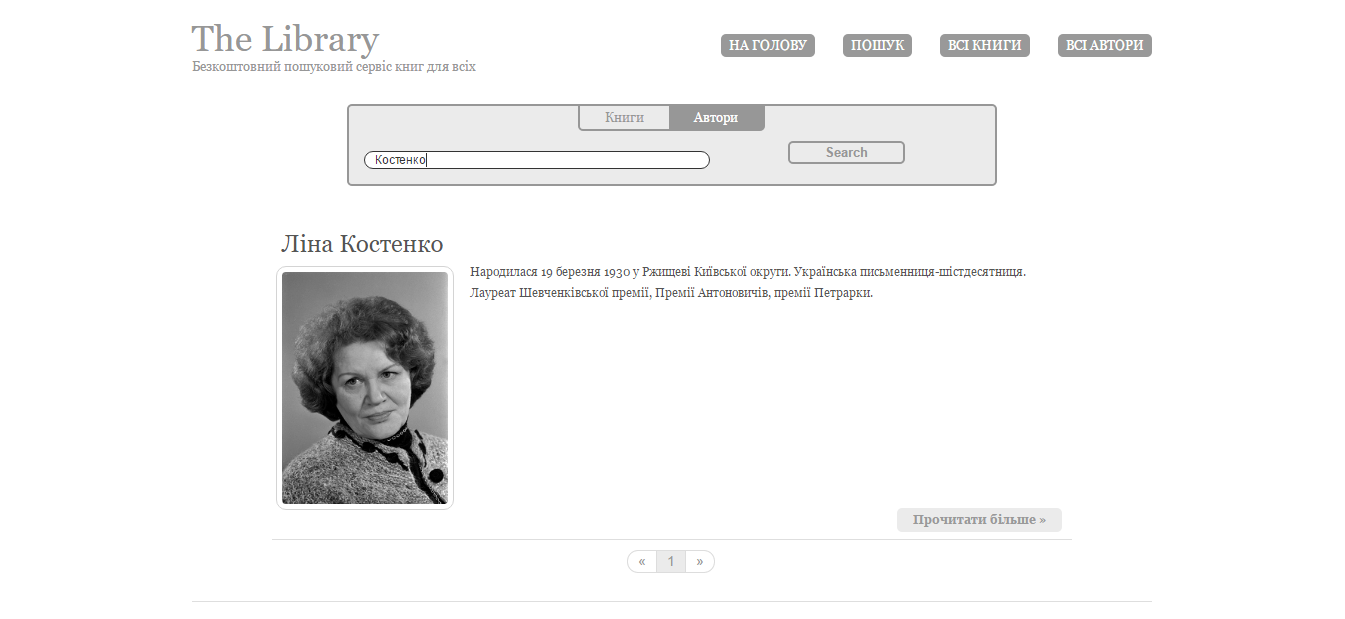
Рис. 4.8. Пошук книг

Рис. 4.9. Пошук авторів