Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Стогова Володимира Володимировича*
(ПІБ)

академічної групи *122-17-2*
(шифр)

спеціальності *122 Комп'ютерні науки та інформаційні технології*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки та інформаційні технології*
(назва освітньої програми)

на тему: *Розробка веб-додатку для бібліотеки електронних книжок на основі фреймворка Spring*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи				
розділів:				
спеціальний				
економічний				
ЫІ				
Рецензент				
Нормоконтролер				

Дніпро
2021

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2021 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *122-17-2* *Стогова Володимира Володимировича*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка веб-додатку для бібліотеки електронних книжок на основі фреймворка Spring*

затверджена наказом ректора НТУ «ДП» від *26.06.2021 р.* № *275-с*

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів виробничої практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі. Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***14.06.2021 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***28.06.2021 р.***

Завдання видав _____ _____
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Стогов В.В.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *10.06.2021 р.*

Термін подання кваліфікаційної роботи до ЕК: *03.06.2021 р.*

РЕФЕРАТ

Пояснювальна записка: 52 с., 19 рис., 0 табл., 3 дод., 18 джерел.

Об'єкт розробки: сайт електронної бібліотеки. Мета кваліфікаційної роботи: розробка веб-додатка для бібліотеки Електрон книжок на основе фреймворка Spring.

У вступі розглядається сучасний стан проблеми, конкретизується мета кваліфікаційної роботи, актуальність та галузь її застосування, уточнюється постановка завдання.

У першому розділі розповідається про предметну галузь, визначено актуальність розробки, сформульовано постановку завдання, зазначено вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі порівнювалися наявні рішення, буда обрана платформа для розробки, виконано проєктування і розробка програми, описана робота програми, алгоритм і структура її функціонування, визначено вхідні і вихідні дані, охарактеризовані параметри технічних засобів.

В економічному розділі визначено трудомісткість розробленої інформаційної системи, проведений підрахунок вартості роботи по створенню програми та розраховано час на його створення.

Практичне значення полягає у створенні сайту, що надає можливість переглядати та заказати книги.

Актуальність інформаційної системи визначається великим попитом на електронні книги та можливість діджиталізації інформації.

Список ключових слів: КОМП’ЮТЕР, КНИГА, БІБЛІОТЕКА, ДОДАТОК, ЕЛЕКТРОННА БІБЛІОТЕКА.

ABSTRACT

Explanatory note: 52 pages, 19 figures, 0 tables, 3 appendices, 18 sources.

Object of development: electronic library site. The purpose of the qualification work: development of a web application for the Electron library of books based on the Spring framework.

The introduction considers the current state of the problem, specifies the purpose of the qualification work, relevance and scope of its application, clarifies the task.

The first section tells about the subject area, determines the relevance of development, formulates the problem, specifies the requirements for software implementation, technology and software.

The second section compares the available solutions, selects a platform for development, designed and developed the program, describes the program, algorithm and structure of its operation, identifies input and output data, describes the parameters of hardware.

The economic section determines the complexity of the developed information system, calculates the cost of work to create a program and calculates the time for its creation.

The practical value is to create a site that allows you to view and order books.

The relevance of the information system is determined by the high demand for e-books and the possibility of digitalization of information.

Keyword list: COMPUTER, BOOK, LIBRARY, APPENDIX, ELECTRONIC LIBRARY.

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ

БД – База даних

ІС – Інформаційна система

JSP - Java Server Pages

JRE - Java Runtime Environment

JDK - Java Development Kitба

ЗМІСТ

ВСТУП

"ІС", як термін належить до класу програмних продуктів, що можуть автоматизувати роботу будь-яких закладів, наприклад бібліотек. Така система оброблює, збирає та шукає інформацію для більш зручного керування звичайним сайтом або навіть бізнесом. Саме цю тему я вибрав для своєї дипломної роботи, тому що ІС електронної бібліотеки дуже близько пов'язана з моєю спеціальністю 122 «Комп’ютерні науки»

Автоматизована ІС це сукупність інформації, технологій та програмного забезпечення. Отже, За допомогою такої системи з'являється можливість встановлення зв'язку між усіма частинами сайту електронної бібліотеки, що покращує можливості контролю й регулювання процесів. ІС, при формуванні якої використано принцип зворотного зв'язку на всіх рівнях управління і сучасні інформаційно-комунікаційні технології, забезпечує зв'язок між елементами системи управління й елементами користувачів, а також надає можливість накопичення, обробку та аналізу даних.

На мою думку, для створення власної ІС по-перше потрібно зробити БД, бо БД буде необхідною для використовування запитів, а по-друге, підготувати сайт для зручного використання користувачами та адміністрацією.

Метою дипломного проекту є створення веб-сайту, який давав би змогу читачам переглядати та читати книги у зручній формі. Зробивши висновок, була сформована тема дипломної роботи: «Проектування ІС електронної бібліотеки». Електронна бібліотека з одного боку виконує функції традиційної бібліотеки: надання інформації читачеві, з іншого - виконує роль, характерну для автоматизованої бібліотечної ІС, - організація і зберігання локальних і віддалених електронних ресурсів і доступу до них.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ І ПОСТАНОВКА ЗАДАЧІ

1.1. Загальні відомості з предметної галузі

В наш час інформація існує не тільки в друкованому, а ще й електронному форматі. Інформаційні технології щороку проникають майже у всі сфери діяльності. Інформація в електронному вигляді забезпечує загальнодоступність та швидке поширення інформації по всьому світу. У наш час, доступ користувачів до електронних ресурсів є одна з першочергових задач науки, освіти і культури. Розвиток інформаційних технологій стимулює швидке зростання обсягу інформаційних ресурсів, які знаходяться в ІС, наприклад, в електронних бібліотеках. Створюючи електронну бібліотеку потрібно розуміти, що перш за все, вона створюється для читачів, а тільки потім – для бібліотекарів. Загальновизнано, що найбільш зручно та ефективно зберігати інформацію не в друкованому, а в електронному вигляді.

На заміну старим бібліотекам, для обслуговування освіти, культури та науки приходять мережі інтернет-бібліотек, що зберігають інформацію вже в електронному вигляді. Така ІС буде зручніше ніж звичайна бібліотека, бо зможе самостійно обробляти запити користувачів, без втручання персоналу. Доступ до такого ресурсу буде доступний через мережу інтернет для будь-якого користувача.

Фундаментом, основою, яка об'єднує традиційні та електронні бібліотеки, є принцип обслуговування користувачів. Функції електронної бібліотеки відрізняються від класичних бібліотечних. З формальної точки зору істотна частина електронних ресурсів - є копії друкованих версій і в цьому сенсі електронна колекція складається з копій, а не з оригіналів, першоджерел. Електронна бібліотека (ЕБ) на сьогоднішній день вторинна по відношенню до традиційної класичної бібліотеки.

Внаслідок фінансових обмежень рівень вітчизняних бібліотек на один – два порядки нижче, ніж в Європі. В найближчі роки швидко змінити ситуацію не вийде, але за допомогою підвищення рівня інформаційного забезпечення вітчизняних фахівців це представляється можливим. Так можна пояснити те, що держава проявляє такий інтерес у діджиталізції тих сфер життя, у яких це можливо.

Основою будь-якої електронної бібліотеки буде доступ до БД книг, список нових надходжень, інформація про події в бібліотеці: конференціях, книжкових виставках. Тобто потрібно відзначити важливу роль технологій БД в електронних бібліотеках. При створенні електронних бібліотек використовуються системи управління БД, засновані на різних моделях даних - реляційні, об'єктні, об'єктно-реляційні. Такі БД підтримують в електронних бібліотеках різноманітні колекції структурованих даних і забезпечують ефективний доступ до них. На Рис. 1.1 можна побачити БД моєї електронної бібліотеки.

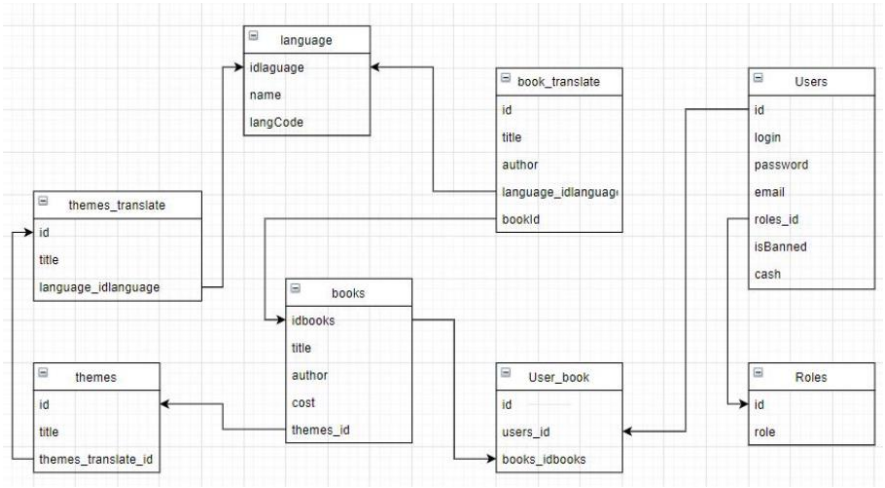

Рис. 1.1. БД електронної бібліотеки

Найпопулярнішою електронною бібліотекою України є Національна бібліотека України імені В. І. Вернадського, яка є державним сайтом. Не дивлячись на це, ця електронна бібліотека має безліч недоробок. Зараз ми їх розглянемо.

Перше, що можна побачити, зайшовши на сайт це застарілий дизайн, який вже давно потрібно було оновити. Але більш важливою проблемою цього сайту є відсутність адаптивної верстки для телефону, так як книги велику кількість користувачів читає з телефону або планшета. Наступне, що захоче побачити користувач, після відвідування сайту, це вхід в свій аккаунт. На даному інтернет ресурсі немає реєстрації нових користувачів, де, наприклад, могла зберігатися історія прочитання книг. На сайт Національний Бібліотеки України можуть зайти тільки ті користувачі, яким адміністрація видала логін і пароль.

Також можна побачити, що в електронній бібліотеці "Відкрита КНИГА" теж немає входу в аккаунт, однак сам сайт зроблений дуже якісно. Розташування розділів в шапці сайту дуже зручно для користувача. На такому сайті складно заплутатися читачеві, в той час як на сайті Національної бібліотеки України занадто багато накладення тексту.

Найзручнішим сайтом, на мою думку, є закордонний сайт Internet archive, де є фільтри для сортування книг та особистий кабінет. Фільтри можна побачити на Рис. 1.2.

Рис. 1.2. Фільтри сайту Internet archive

В моїй електронній бібліотеці я можу надати рішення всіх перерахованих вище проблем, які я хотів би освятити в даній дипломній роботі, так як вважаю, що найважливішою частиною будь-якого інформаційного сайту є простота і розуміння користувачем.

1.2. Призначення розробки та галузь застосування

Призначення розробки та галузь застосування – це автоматизація ІС електронної бібліотеки:

- читання таблиць БД;

- зручний інтерфейс для користувача та адміністратора;

- перегляд, редагування та видалення книг;

1.3. Підстава для розробки

Відповідно до ОКХ та ОПП, згідно начального плану та графіків навчального процесу, в кінці навчання студент виконує кваліфікаційну роботу (дипломний проект). Тема роботи узгоджується з керівником проекту, випускаючою кафедрою, та затверджується наказом ректора.

Отже, підставами для розробки (виконання кваліфікаційної роботи) є:

- ОКХ та ОПП за напрямом підготовки 6.050101 «Комп’ютерні науки»;
- Графік навчального процесу та навчальний план;
- наказ ректора Національного технічного університету «Дніпровська політехніка» № _____ від __.__.2021 р;
- завдання на дипломний проект на тему «Проектування ІС електронної бібліотеки».

1.4. Постановка завдання

Завданням дипломного проекту є проектування ІС електронної бібліотеки. Програмне забезпечення призначене для надання користувачам можливість перегляду книг.

Програма повинна реалізувати наступні функції:

- можливість користувачів доступу до бібліотеки книг;

- можливість додавання, редагування та видалення книг;

- можливість перегляду даних користувачів, з можливістю блокування.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

Вимоги до програми – це проектування автоматизованої ІС електронної бібліотеки:

- читання таблиць БД з даними про книги;

- читання таблиць БД з даними про користувачів;

- перегляд користувачів

- швидка навігація по сайту

- коректна візуалізація елементів веб-додатку

1.5.2 Вимоги до інформаційної безпеки.

Головне вікно програми повинно давати змогу для входу користувача як адміністратор, або звичайний користувач.

Рівнем доступу будуть виступати:

- користувач;

- адміністратор.

Для входу в свій аккаунт потрібно ввести логін та пароль, який був раніше зареєстрований в БД користувачем або директором.

1.5.3 Вимоги до складу та параметрів технічних засобів.

Для забезпечення надійного функціонування програмного забезпечення необхідно, щоб обчислювальна машина, на якій буде експлуатуватися веб-додаток, мала такі характеристики:

- Маніпулятор “миша”.
- Клавіатура.
- Доступ до онлайн мережі.
- 1 Гб вільного місця на жорсткому диску.
- Процесор Intel Core i3-2348 з тактовою частотою 2.3 ГГц.
- Не менше ніж 4 Гб оперативної пам’яті.
- Рідкокристалічний монітор з діагоналлю 17".

Вище наведені характеристики являють собою рекомендовані. Це означає, що при наявності характеристик не нижче зазначених, розроблений додаток буде функціонувати відповідно до вимог щодо надійності, безпеки та швидкості обробки даних.

1.5.4 Вимоги інформаційної та програмної сумісності.

Для коректного функціонування програми необхідно, щоб програмне забезпечення обчислювальної машини, на якій буде експлуатуватися веб-додаток, відповідало наступним вимогам:

- Веб браузер Firefox або Google Chrome.
- Операційна система Windows 7/10.

Веб-орієнтована підсистема має бути реалізована на мові програмування Java. Для візуалізації було використано CSS який описує зовнішній вигляд сторінки.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1. Функціональне призначення системи

Призначення програми складається з автоматизації ІС електронної бібліотеки. Головна сторінка сайту дозволяє користувачеві побачити свіжу інформацію про бібліотеку. З цієї сторінки можна побачити меню сайту, де можна перейти в розділ з книгами і на сторінку логіна / реєстрації. При переході в розділ з книгами, можна тільки побачити перелік книг, для того, щоб їх читати або додати в свою бібліотеку, потрібно зайти в свій акаунт, вводячи особисті дані.

Даними виступають:

- логін;

- пароль.

Для входу на сайт як адміністратор потрібно ввести логін і пароль адміністратора в відповідні поля на формі.

Переглянути свої придбані книги можна в розділі Особистий кабінет, також в цьому розділі можна побачити свій логін і пошту. Для редагування книг потрібно зайти на сайт з логіном і паролем адміністратора, який може додавати / змінювати / видаляти інформацію в БД через сайт.

2.2. Опис застосованих математичних методів

Оскільки особливості предметної області ІС електронної бібліотеки не передбачають застосування математичних методів при розробці, математичні методи не використовувалися.

2.3. Опис використаних технологій та мов програмування

При розробці даного сайту ми використовувалося середовище Microsoft Visual Studio 2021 та технологію JSP.. Для обробки даних сайту я використовував інтерфейс MySQL.

SQL - це мова, що використовується для взаємодії з БД. За допомогою неї отримують доступ до інформації, що зберігається в таблицях MySQL. Мова ділиться на три частини:

- Ідентифікація окремих компонентів БД.
- Управління даними в базі, який допомагає оновлювати і шукати інформацію.
- Можливість видавати користувачам права на окремі одиниці БД.

JSP (JavaServer Pages) - технологія, що дозволяє веб-розробникам створювати вміст, який має як статичні, так і динамічні компоненти. Сторінка JSP містить текст двох типів: статичні вихідні дані, які можуть бути оформлені в одному з текстових форматів HTML, SVG, WML, або XML, і JSP-елементи, які конструюють динамічний вміст. Також часто використовуються бібліотеки JSP-тегів, а також Expression Language (EL), для впровадження Java-коду в статичне вміст JSP-сторінок.

В Java реалізований принцип WORA: write once, run anywhere або «пиши один раз, запускай скрізь». Це означає, що написану на Java програму, в нашому випадку сайт, можна запустити на будь-якій платформі, якщо на ній встановлена середовище розробки Java (JRE).

Це стає можливим завдяки компіляції написаного на Java коду в байт-код. Цей формат виконує віртуальна машина Java або JVM. Віртуальна машина не залежить від платформи.

Середовище виконання JRE надає собою «контейнер» для коду щоб запустити додаток. JDK - це «компілятор», який інтерпретує сам код і виконує його. В JDK також є інструменти розробника, необхідні для написання коду

Java. Дуже зручно, що розробникам потрібно тільки завантажити JDK, для написання коду.

Клас, в об'єктно-орієнтованому програмуванні, являє собою шаблон для створення об'єктів, що забезпечує початкові значення станів: ініціалізація полів-змінних і реалізація поведінки функцій або методів.

Зараз я докладніше розповім про найважливіші частинах програми:

Метод postLogin відповідає за вхід користувача в акаунт, а саме отримує логін і пароль, далі отримані дані порівнюються з даними з БД, після чого читач або заходить на сайт для користування, або йому відмовляють. Також в цій частині коду є перевірка на неправильні логін і пароль, наприклад якщо користувач не ввів логін і намагається зайти в акаунт, то сайт продовжить роботу і виведе відповідне повідомлення.

```
public String postLogin(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
    String login = req.getParameter("login");
    String password = req.getParameter("password");
    User user = userDAO.findByLogin(login).orElse(null);
    if (user != null && user.getLogin().equals(login) && user.getPassword().equals(password) && !user.getIsBanned()) {
        HttpSession session = req.getSession();
        session.setAttribute("user", user);
        session.setAttribute("userCash", user.getCash());
        session.setAttribute("userId", user.getId());
        session.setAttribute("login", login);
        session.setAttribute("password", password);
        session.setAttribute("isLogged", user);
        session.setAttribute("userRole", user.getRole());
        session.setAttribute("lang", "en");
        if (user.getRole().getRole().equals("admin")) {
            session.setAttribute("isAdmin", true);
        }
        return "redirect:/about";
    } else if (user != null && user.getIsBanned()) {
        req.setAttribute("ban", true);
        return  "login";
    } else {
        req.setAttribute("pass", true);
```

```
        return  "login";
    }
  }
```

Метод getLogout дозволяє користувачеві вийти з свого облікового запису і перенаправляє на сторінку входу в аккаунт.

```
public String getLogout(HttpServletRequest req) throws IOException {
    HttpSession session = req.getSession();

    if(session.getAttribute("login")!=null){
        session.invalidate();
    }
    return "redirect:/login";
  }
```

Метод postOrder відповідає за купівлю книг і в разі недостатнього балансу на рахунку у користувача, не дозволяє йому купувати нові книги.

```
  public String postOrder(HttpServletRequest req) {
    HttpSession session = req.getSession();
    Long userId = Long.valueOf(String.valueOf(session.getAttribute("userId")));
    Long bookId = Long.valueOf(req.getParameter("book_id"));
    BigDecimal bookCost = new BigDecimal(String.valueOf(req.getParameter("book_cost")));
    BigDecimal userCash = new BigDecimal(
            String.valueOf(session.getAttribute("userCash"))
    ).subtract(bookCost);
    if (userCash.compareTo(new BigDecimal("0.00")) < 0) {
        return "redirect:/main";
    }
    orderDAOImpl.orderBook(bookId, userId);
    userDAOImpl.updateCash(userId, bookCost);
    session.setAttribute("userCash", new BigDecimal(
            String.valueOf(session.getAttribute("userCash"))
    ).subtract(bookCost));
    return "redirect:/main";
  }
```

За допомогою методу postRegistration користувач може зареєструватися на сайті і внести свої дані в БД сайту. При введенні некоректно даних, сайт повідомить користувача.

```
public String postRegistration(HttpServletRequest req, HttpServletResponse resp) {
      String login = req.getParameter("login");
      String password = req.getParameter("password");
      String email = req.getParameter("email");
      if (login == null || password == null || email == null || login.isEmpty() || password.isEmpty() || email.isEmpty())
{
         req.getRequestDispatcher("error.jsp");
      }
      if (userDAO.findByLogin(login).isPresent()) {
         throw new UserExistException();
      }
      if (userDAO.findByEmail(email).isPresent()) {
         throw new UserExistException();
      }
      User user = User.builder()
            .login(login)
            .password(password)
            .email(email)
            .role(null)
            .isBanned(false)
            .build();
      userDAO.create(user);
      return "main";
   }
```

GetUser відповідає за виведення на сторінку особистого кабінету користувача його куплених книг і загальну вартість покупок.

```
public String getUser(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
      HttpSession session = req.getSession();
      User userId =
userDAOImpl.findById(Long.valueOf(String.valueOf(session.getAttribute("userId")))).orElseThrow(RuntimeExce
ption::new);
      req.setAttribute("user", userId);
      String str = String.valueOf(session.getAttribute("lang"));
      List<Book> bookLists =
bookService.findAllUser(Long.valueOf(String.valueOf(session.getAttribute("userId"))), str);
      req.setAttribute("bookList", bookLists);
      Map<Book, Long> booksToCount = bookLists.stream().collect(Collectors.groupingBy(Function.identity(),
Collectors.counting()));
      req.setAttribute("booksToCount", booksToCount);
      req.setAttribute("totalPrice", bookLists.stream().map(Book::getCost).reduce(BigDecimal.ZERO,
BigDecimal::add));
      return "user";
   }
```

За допомогою класу UserListController стало можливим виводити всіх користувачів з бази даних з можливістю блокування / розблокування аккаунта читача.

```
public class UserListController extends HttpServlet {
   private final UserDAO userDAO;

   @GetMapping
   public String getList(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
      List<User> userList = userDAO.findALL();
      req.setAttribute("userList", userList);
      return "userList";
   }
   @PostMapping
   public String postList(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException
{
      Long userId = Long.valueOf(req.getParameter("userId"));
      Boolean isBanned = Boolean.valueOf(req.getParameter("isBanned"));
      userDAO.updateStatus(userId, !isBanned);
      return "redirect:/userList";
   }

}
```

Дуже важливою частиною програми є фільтри, які не дозволяють незареєстрованому користувачу потрапляти на сторінки / user, / userList, це можна побачити в класі LoginFilter.

```
@WebFilter({"/user", "/userList"})
public class LoginFilter implements Filter {
   @Override
   public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain)
throws IOException, ServletException {
      HttpServletRequest request = (HttpServletRequest) servletRequest;
      HttpServletResponse response = (HttpServletResponse) servletResponse;
      HttpSession session = request.getSession(false);
      String loginURI = request.getContextPath() + "/login";
      boolean loggedIn = session != null && session.getAttribute("login") != null &&
session.getAttribute("userRole") != null;
      boolean loginRequest = request.getRequestURI().equals(loginURI);
      if (loggedIn || loginRequest){
         filterChain.doFilter(request,response);
      }else{
         response.sendRedirect(loginURI);
      }
   }
```

Наступний файл, про який я б хотів розповісти, це application.properties. Цей файл зберігає в собі дані для підключення до БД MySQL, а саме логін і пароль.

```
database.url = jdbc:mysql://localhost:3306/library_store?createDatabaseIfNotExist=true&serverTimezone=UTC
database.username = password
database.password = password
```

2.4. Опис структури системи та алгоритмів її функціонування.

Структура мого проекту складається з великої кількості файлів, зараз ми їх розглянемо детальніше. На рис. 2.1., 2.2., 2.3., 2.4. наведена структура файлів проекту.

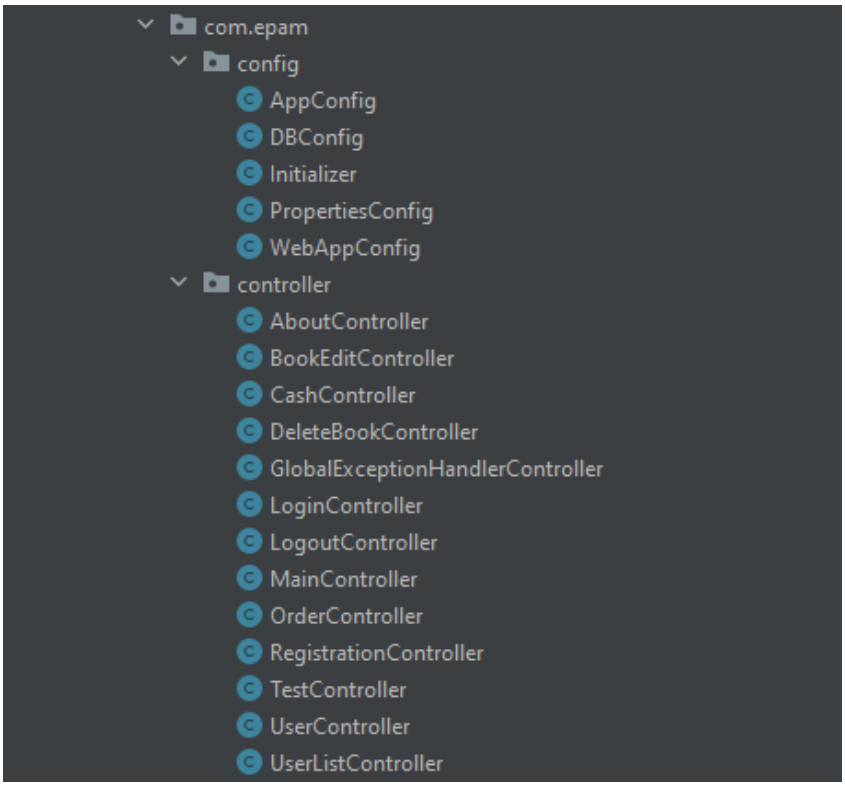

Рис. 2.1. Структура файлів проекту

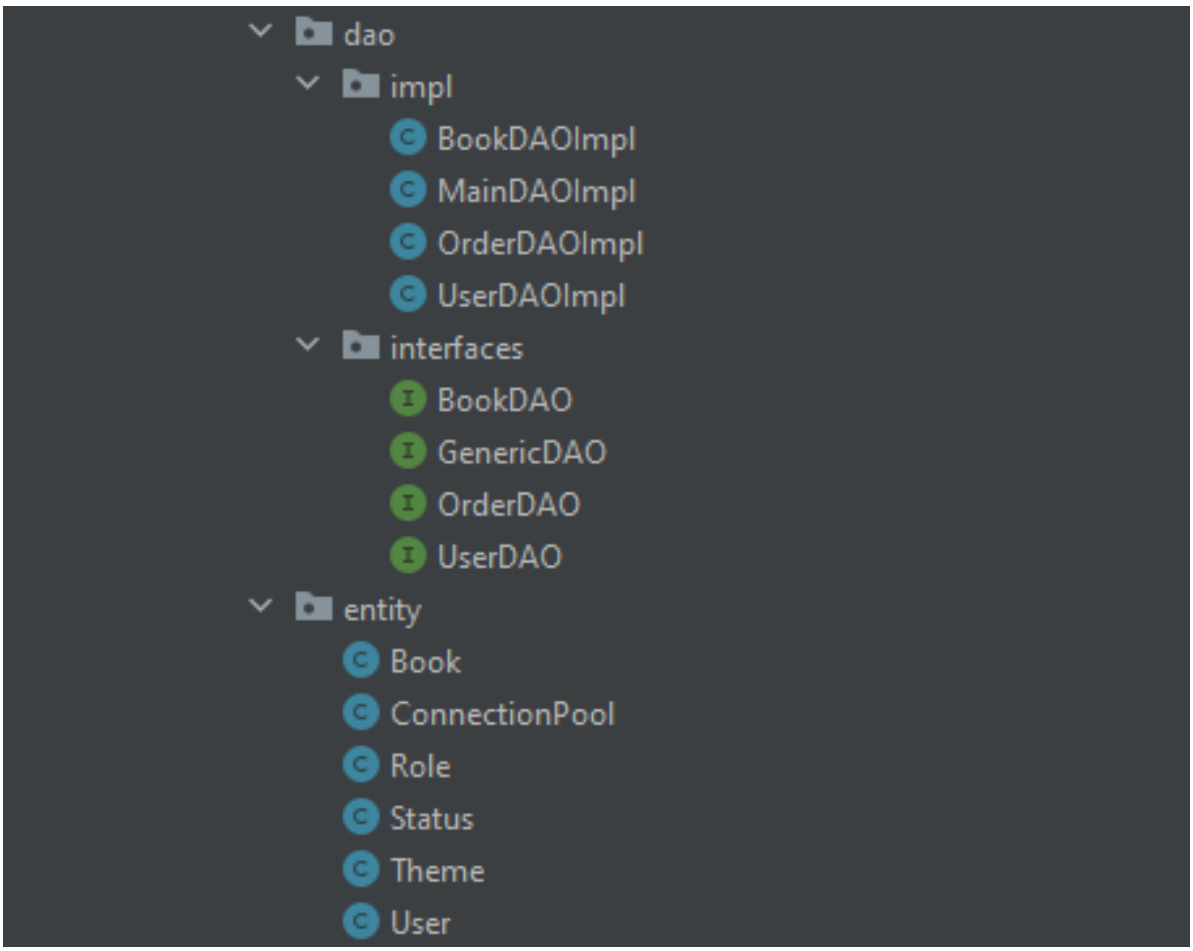

Рис. 2.2. Структура файлів проекту

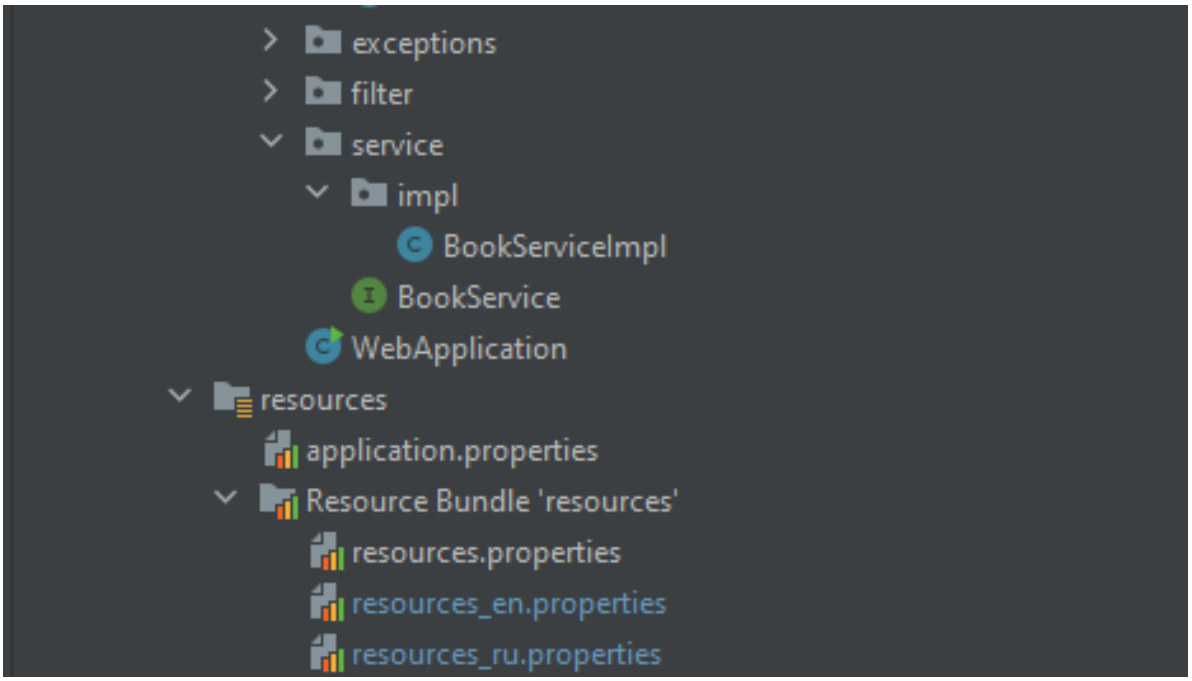

Рис. 2.3. Структура файлів проекту

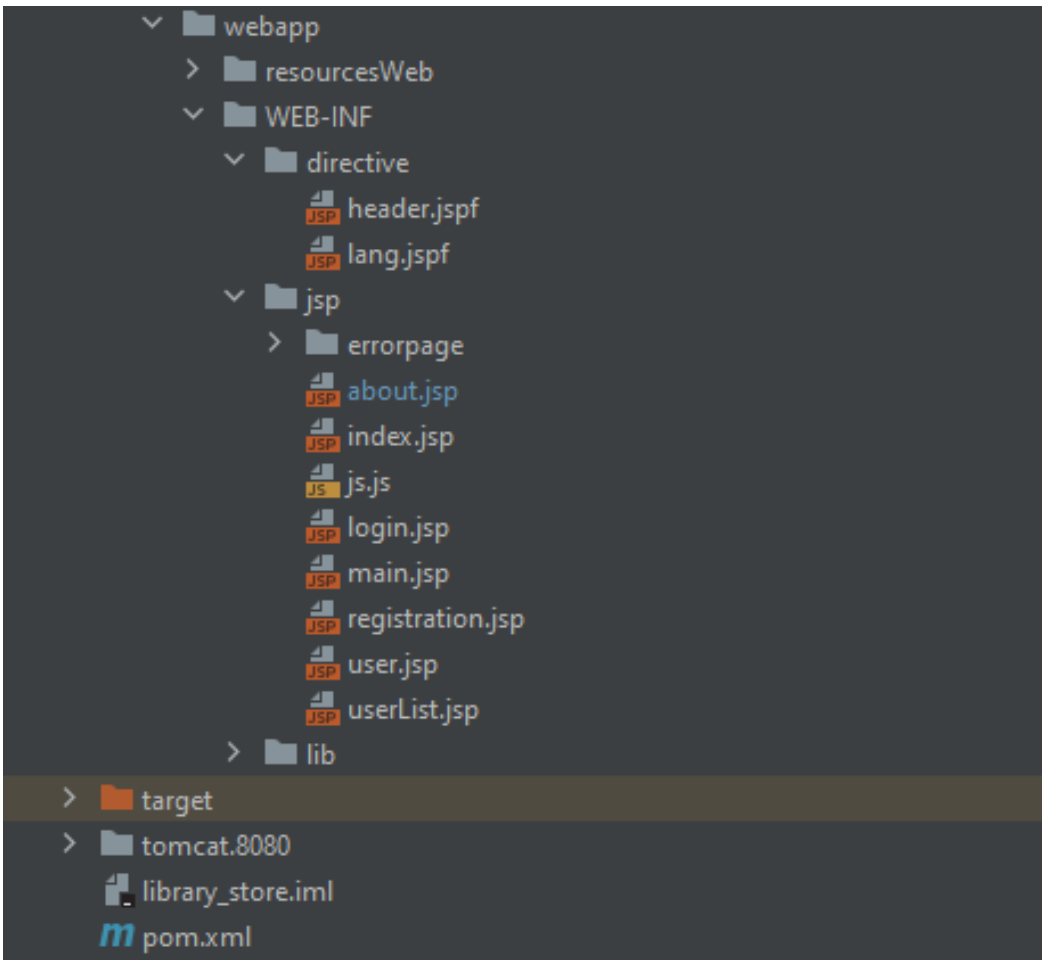

Рис. 2.4. Структура файлів проекту

Опис файлів структури:

1. Config – файли, які дозволяють користуватися фреймворком Spring.
2. Controller – папка, де зберігаються java класи для обробки сторінок сайту.
3. Impl – реалізація інтерфейсу, основний функціонал.
4. Interfaces – шаблон, поведінку класу.
5. Entity – опис об'єктів ІС.
6. Exceptions – винятки та їх обробка.
7. Filter - попередня обробка запиту, перш ніж той потрапляє в Controller.
8. Service – набір класів, що відповідають за бізнес логіку.
9. WebApplication – файл для запуску програми.
10. Resources – папка з ресурсами, необхідними для виконання програми.

11. ResourcesWeb – папка з ресурсами для змалювання сторінок сайту.

12. Web-Inf – цей каталог містить всі речі, пов'язані з проектом, які не перебувають в кореневому каталозі документа проекту.

13. Pom.xml – конфігураційний файл, містить різну інформацію про проект та його конфігурації.

Дуже важливою частиною моєї ІС є БД, яка зроблена в програмі My SQL, це видно на Рис.2.5.

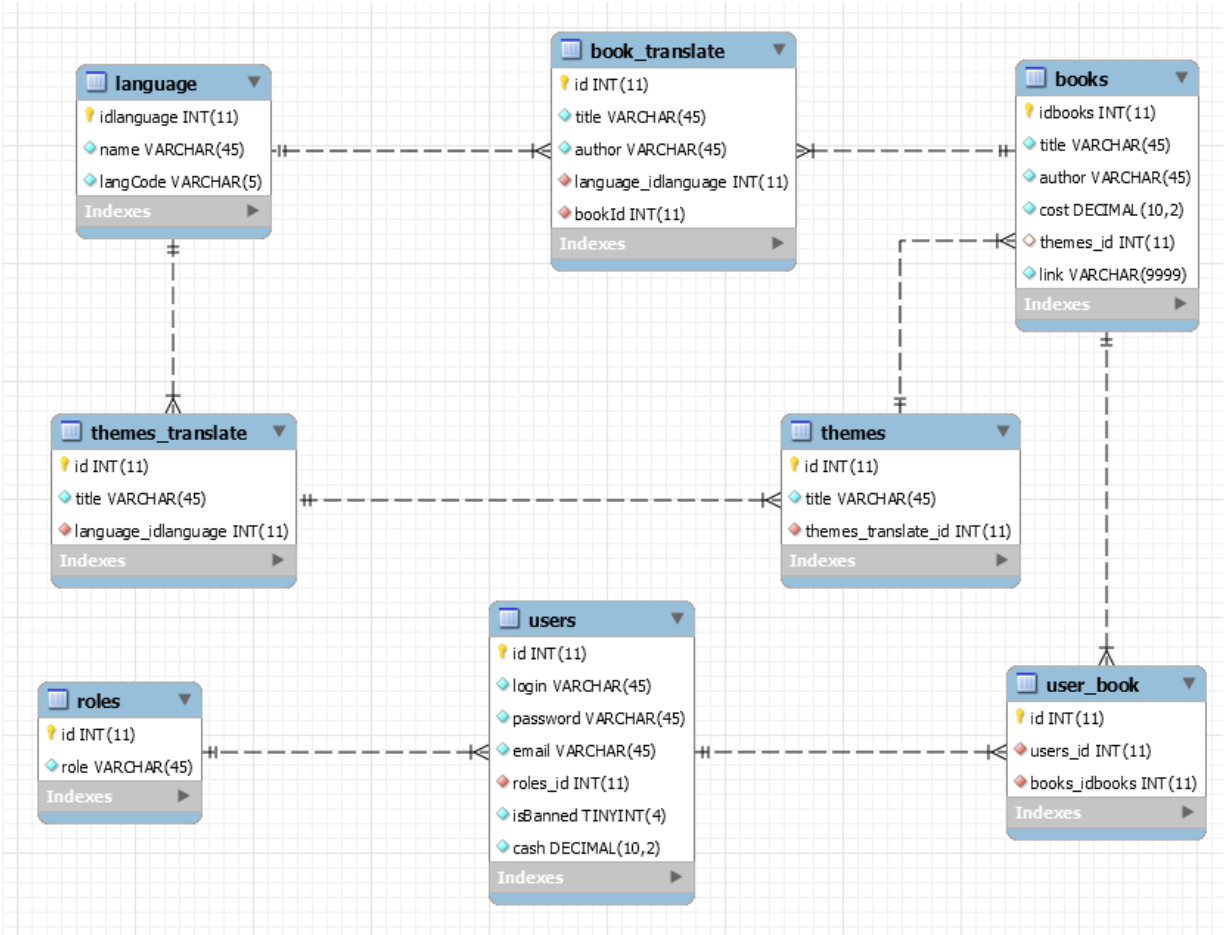

Рис.2.5. Структура БД

Моя БД є реляційною, тому що дані організовані у вигляді набору таблиць, що складаються із стовпців і рядків, де зберігається інформація про об'єкти, представлених в БД. У кожному стовпчику таблиці зберігається певний тип даних, в кожному осередку - значення атрибута.

2.5 «Обґрунтування та організація вхідних та вихідних даних програми

Згідно задачам, які вирішує даний програмний засіб, організація вхідних та вихідних даних програми має такий вигляд.

Вхідними даними для даного програмного комплексу є:

- електронні таблиці формату My SQL;
- логін користувача;
- пароль користувача.

Вихідними даними комплексу є:

- таблиця користувачів
- таблиця книг;

2.6. Опис розробленої системи.

2.6.1. Використані технічні засоби.

Для функціонування системи необхідна клієнтська персональна ЕОМ з наступними мінімальними характеристиками:

- процесор класу Intel Core i3 2 ядра 3,9Ггц;
- монітор;
- не менше 4Гб ОЗУ;
- 500Мб вільного місця на диску;
- клавіатура;
- маніпулятор «миша».

2.6.2. Використані програмні засоби.

При створенні мого сайту була використана середовище розробки Intellij Idea, оскільки саме ця програма має велику кількість плюсів, а саме:

- Зручність читання коду, тому що синтаксис підсвічується і код зручно форматується табами
- Інтеграція з сайтом Git і іншими системами контролю версій
- Вбудований зручний відладчик програми

Наступною програмою, якої я користувався для створення сайту електронної бібліотеки, була MySQL. Ця програма існує для створення і редагування баз даних. Я скористався саме MySQL, тому що вона повністю задовольняє потреби, а саме:

- Нить, підтримка декількох запитів одночасно
- Гнучка підтримка форматів змінної і їх довжини
- Доступність програми, так як вона є безкоштовною

2.6.3. Виклик та завантаження програми

Сайт не потребує інсталювання, може бути використаний з будь-якого носія стандартними засобами браузера.

Сайт розроблений для будь-яких комп'ютерів або телефонів з інтернетом. Для використання сайт не потребує додаткових програмних засобів.

2.6.4. Опис інтерфейсу користувача

На головній сторінці, тобто розділі About us, можна побачити новини бібліотеки і України, також на всіх сторінках сайту можна поміняти мову, це видно на Рис. 2.6.

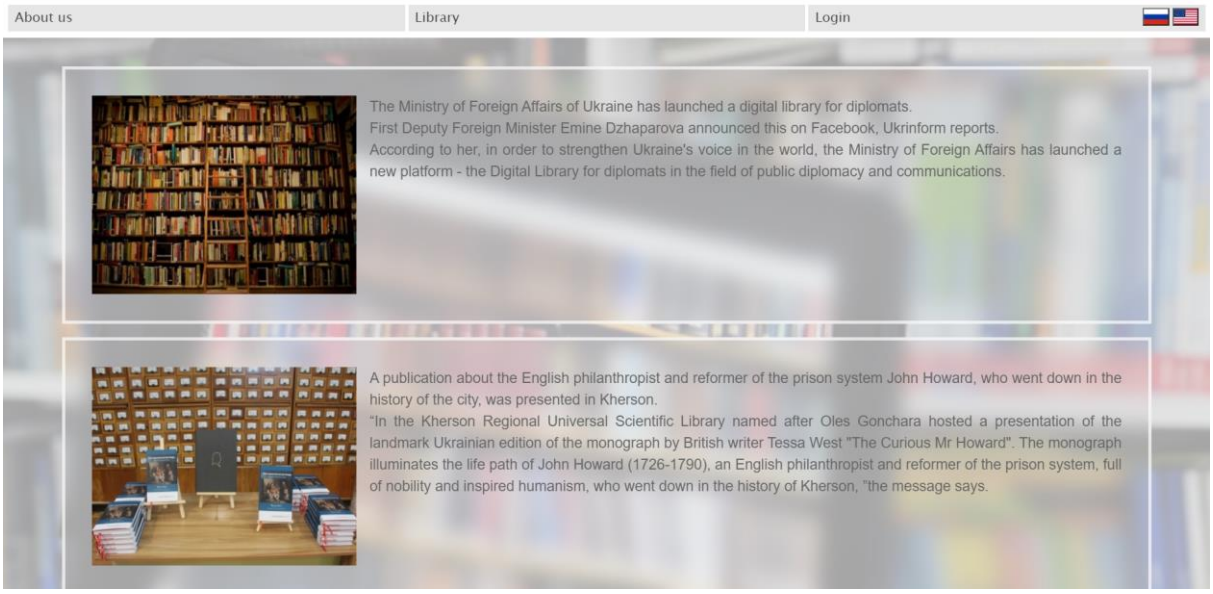

Рис. 2.6. Головна сторінка сайту

В розділі Library можна побачити саму бібліотеку, яка представляє собою таблицю з книгами. Користувач, що не зайшов під своїми особистими даними в акаунт, не може взаємодіяти з таблицею (Рис. 2.7.)

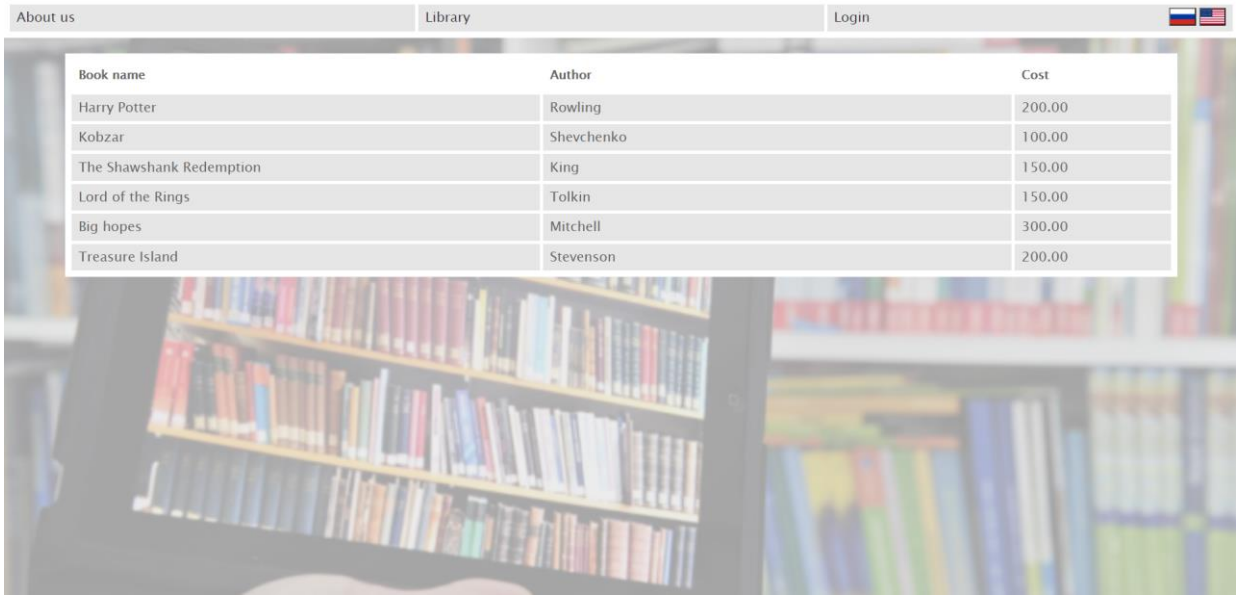

Рис. 2.7. Сторінка списку книг

Наступний розділ, який побачить користувач, буде від мене вимагається залогуватись в обліковий запис під своїм логіном і паролем (Рис. 2.8.).

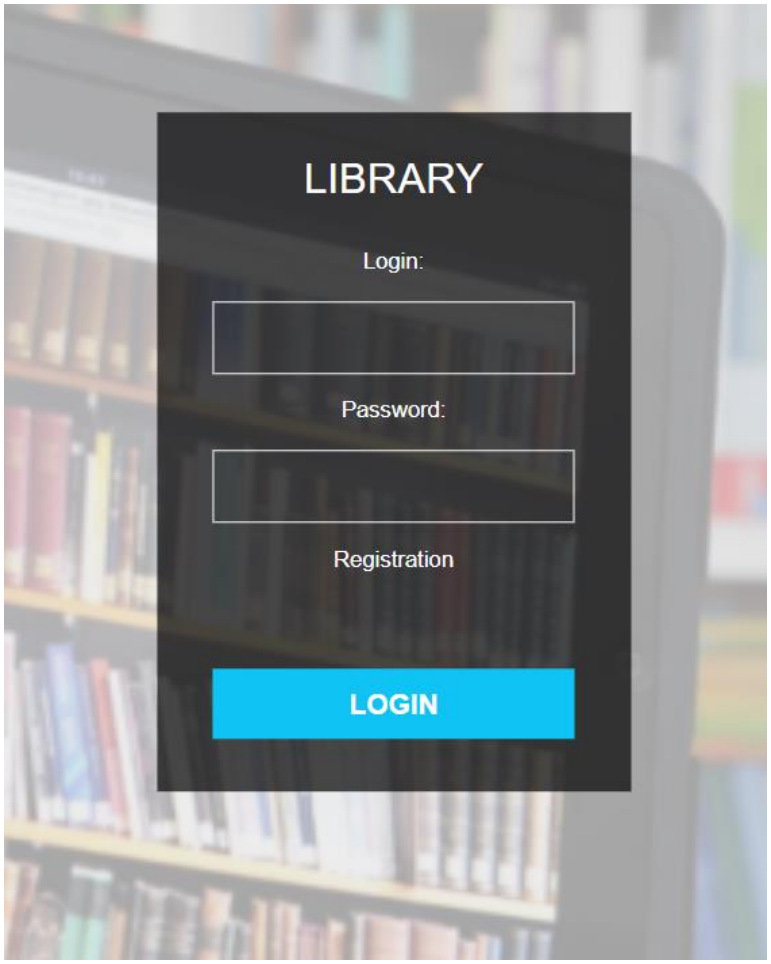

Рис. 2.8. Вікно логіна користувача

Якщо користувач не був зареєстрований в нашій електронній бібліотеці, то при натисканні на кнопку Registration він зможе створити свій новий акаунт в БД. На Рис.2.9. видно, що для реєстрації користувача потрібно вказувати крім логіна і пароля свою пошту, так як в майбутньому можна буде реалізувати розсилку повідомлень зареєстрованим читачам на пошту.

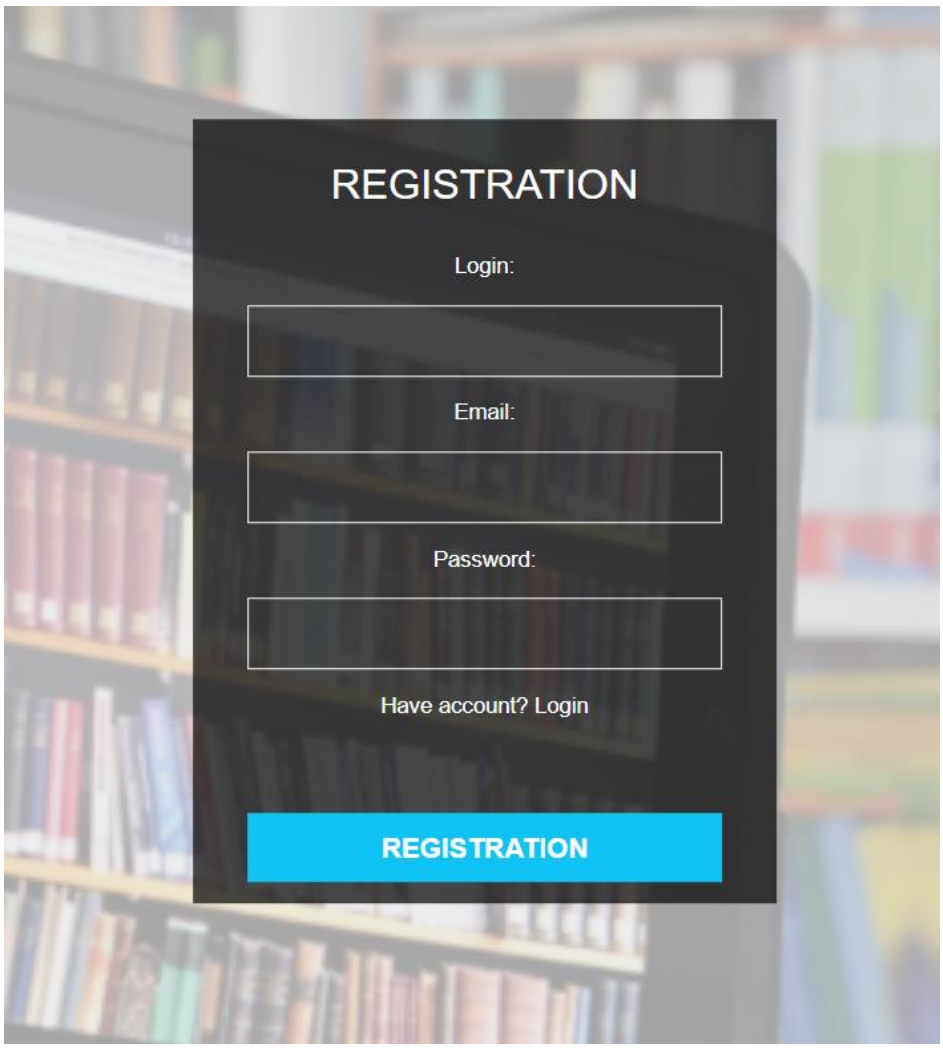

Рис.2.9. Вікно реєстрації користувача

Коли користувач входить під своїм обліковим записом, у нього з'являються пункти меню, які може побачити:

- користувач, який не ввійшов в обліковий запис (Рис.2.10.)
- звичайний читач бібліотеки (Рис.2.11.)
- адміністратор бібліотеки (Рис.2.12.)

Просто перейти на розділи, які вам не доступні не вийде, і користувач буде перенаправлений на сторінку логіна.

Рис.2.10. Пункти меню користувача без акаунта

Рис.2.11. Пункти меню читача з обліковим записом

Рис.2.12. Пункти меню адміністратора

Коли користувач зайшов під своїм логіном і паролем, на сторінці бібліотеки можна побачити нові поля, а саме Order і Online reading (Рис.2.13.)

Book name	Author	Cost	Order	Online reading
Harry Potter	Rowling	200.00	Order	Read
Kobzar	Shevchenko	100.00	Order	Read
The Shawshank Redemption	King	150.00	Order	Read
Lord of the Rings	Tolkin	150.00	Order	Read
Big hopes	Mitchell	300.00	Order	Read
Treasure Island	Stevenson	200.00	Order	Read

Рис.2.13. Пункти меню адміністратора

Натиснувши на кнопку Order навпроти будь-якої книги, користувач додає її до себе в кошик, для того, щоб потім замовити собі друковану версію. Оскільки це саме електронна бібліотека, а не книжковий магазин, натиснувши на кнопку Read, користувачеві відкривається електронний варіант книги для читання(Рис.2.14.).

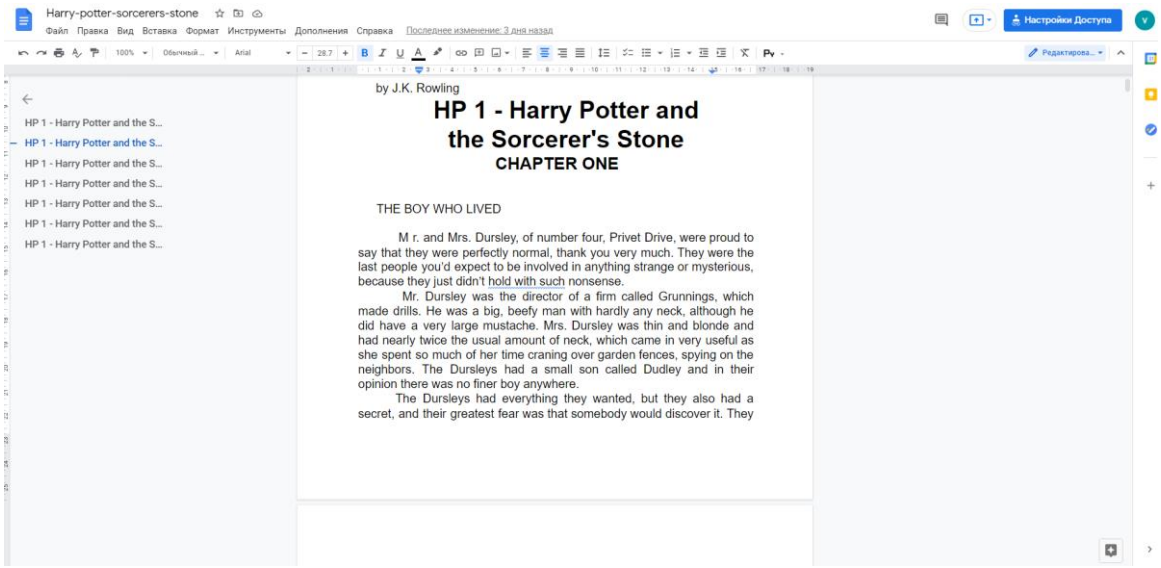

Рис.2.14. Електронний варіант книги

При натисканні на Userinfo, залогіненний читач потрапляє на сторінку користувача. На Рис.2.15. можна побачити особистий грошовий рахунок людини, вартість всіх куплених книг і список всіх куплених книг. Якщо користувачеві потрібно поповнити рахунок, то потрібно натиснути Deposit funds і рахунок поповнюється на 100 Грн.

Рис.2.15. Особистий кабінет користувача

Коли в акаунт заходить адміністратор, у нього з'являється пункт User list в меню, де можна побачити список всіх читачів і їх e-mail. Також адміністратор може блокувати або розблокувати користувачів Рис.2.16.

11	maxkapusta	kapusta@gmail.com	user	true	Block/Unblock user
12	newmeme	newmeme@gmail.com	user	false	Block/Unblock user
13	12345u	12345u@gmail.com	user	false	Block/Unblock user
14	bogdan	bogdan@gmail.com	user	false	Block/Unblock user
15	nexor	Nexor@gmail.com	user	true	Block/Unblock user
16	test	test@gmail.com	user	true	Block/Unblock user
17	Danya	Vova_stogov@mail.ru	user	false	Block/Unblock user
18	1234	1234@gmail.com	user	false	Block/Unblock user
19	Jeka	Joker@gmail.com	user	true	Block/Unblock user
20	Tyoma	Tyoma@gmail.com	user	false	Block/Unblock user
21	uuu	uuu@gmail.com	user	false	Block/Unblock user
22	Artemida	Artemida@gmail.com	user	false	Block/Unblock user
24	null	null@gmail.com	user	false	Block/Unblock user
9	admin	admin@gmail	admin	false	Block/Unblock user

Рис.2.16. Список користувачів

Заблокований користувач при логіні бачить напис, що він заблокований, це видно на Рис.2.17.

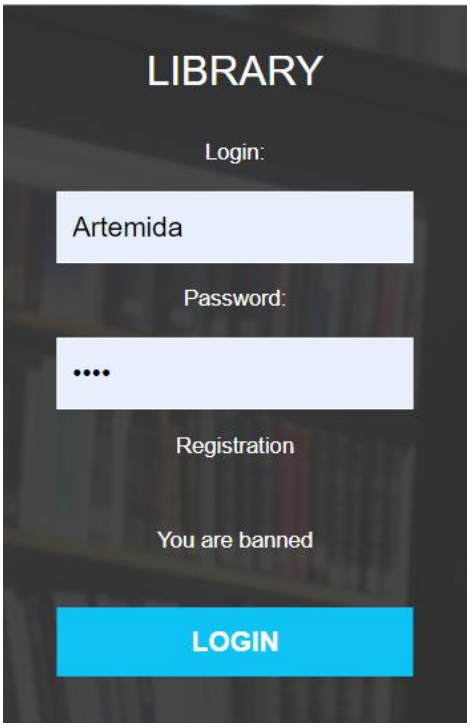

Рис.2.17. Список користувачів

При вході під акаунтом адміністратора, на сторінці з книгами з'являється додатковий функціонал, а саме Edit і Delete, що дозволяє змінювати (Рис.2.18.) і видаляти (Рис.2.19.) книги зі списку у всіх користувачів.

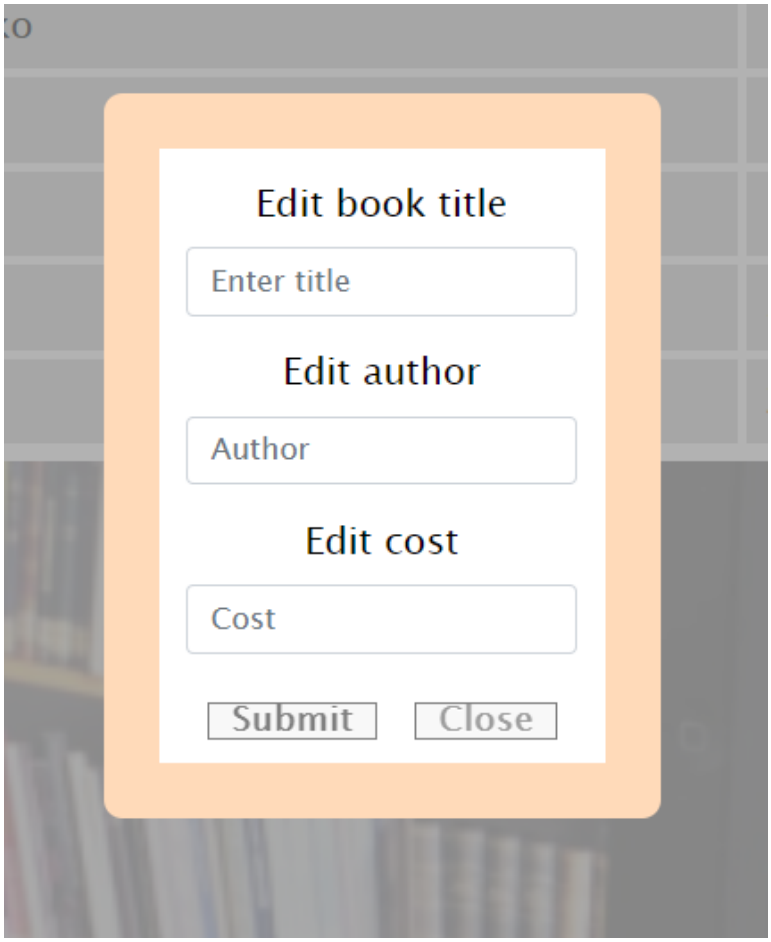

Рис.2.18. Редагування книги

Рис.2.19. Кнопка видалення книги

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

Вихідні дані розробки програмного забезпечення:

а) передбачуване число операторів – 1283;

б) коефіцієнт складності програми – 1,31;

в) коефіцієнт кореляції програми в ході її розробки – 0,2;

г) середня годинна заробітна плата програміста, грн/год – 100;

д) коефіцієнт кваліфікації програміста, обумовлений від стажу – 0,8;

е) вартість машино-години ЕОМ, грн/год – 8.

3.1. Визначення трудомісткості розробки програмного забезпечення

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д \text{, людино-годин,} \qquad (3.1)$$

де t_o – витрати праці на підготовку й опис поставленої задачі (приймається 60 людино-годин);

t_u – витрати праці на дослідження алгоритму рішення задачі,

t_a – витрати праці на розробку блок-схеми алгоритму,

t_n – витрати праці на програмування по готовій блок-схемі,

$t_{отл}$ – витрати праці на налагодження програми на ЕОМ,

$t_д$ – витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \cdot C(1 + p), \text{ людино-годин,} \qquad (3.2)$$

де q – передбачуване число операторів,

C – коефіцієнт складності програми,

p – коефіцієнт кореляції програми в ході її розробки.

$$Q = 1283 \cdot 1{,}31 \cdot (1 + 0{,}2) = 2\ 016{,}8$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q \cdot B}{(75..85) \cdot k}, \text{ людино-годин,} \qquad (3.3)$$

де B, яке дорівнює 1,4, – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі,

k, яке дорівнює 0,8, – коефіцієнт кваліфікації програміста, обумовлений стажем роботи з даної спеціальності.

$$t_u = \frac{2\ 016{,}8 \cdot 1{,}4}{82 \cdot 0{,}8} = 43{,}04, \text{ людино-годин.}$$

Витрати праці на розробку алгоритму рішення задачі:

$$t_a = \frac{Q}{(20..25)\cdot k}; \quad (3.4)$$

$$t_a = \frac{2016{,}8}{20\cdot 0{,}8} = 126{,}05, \text{ людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20..25)\cdot k}, \quad (3.5)$$

$$t_n = \frac{2016{,}8}{21\cdot 0{,}8} = 120, \text{ людино-годин.}$$

Витрати праці на налагодження програми на ЕОМ:

– за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4..5)\cdot k}; \quad (3.6)$$

$$t_{отл} = \frac{2016{,}8}{5\cdot 0{,}8} = 504{,}2, \text{ людино-годин,}$$

– за умови комплексного налагодження завдання:

$$t_{отл}^{к} = 1{,}4 \cdot t_{отл}; \tag{3.7}$$

$$t_{отл}^{к} = 1{,}4 \cdot 504{,}2 = 705.88$$, людино-годин.

Витрати праці на підготовку документації:

$$t_{д} = t_{др} + t_{до}, \tag{3.8}$$

де $t_{др}$ – трудомісткість підготовки матеріалів і рукопису

$$t_{др} = \frac{Q}{(15..20) \cdot k}; \tag{3.9}$$

$$t_{др} = \frac{2016{,}8}{15 \cdot 0{,}8} = 168{,}06$$, людино-годин.

де $t_{до}$ – трудомісткість редагування, печатки й оформлення документації

$$t_{до} = 0{,}75 \cdot t_{д}; \tag{3.10}$$

$$t_{до} = 0{,}75 \cdot 168{,}06 = 126{,}04$$, людино-годин.

$$t_{д} = 168{,}06 + 126{,}04 = 294{,}1$$, людино-годин.

Отримаємо трудомісткість розробки програмного забезпечення:

$$t_д = 60 + 43{,}04 + 126{,}05 + 120 + 504{,}2 + 294{,}1 = 1147{,}39, \text{ людино-годин.}$$

У результаті ми розрахували, що в загальній складності необхідно 1147,39 людино-годин для розробки даного програмного забезпечення.

3.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ $К_{ПО}$ включають витрати на заробітну плату виконавця програми $З_{ЗП}$ і витрат машинного часу, необхідного для налагодження програми на ЕОМ.

$$К_{ПО} = З_{ЗП} + З_{МВ}, \text{ грн,} \qquad (3.11)$$

де $З_{ЗП}$ – заробітна плата виконавців, яка визначається за формулою:

$$З_{ЗП} = t \cdot С_{ПР}, \text{ грн,} \qquad (3.12)$$

де t – загальна трудомісткість, людино-годин,

$С_{ПР}$ – середня годинна заробітна плата програміста, грн/година.

$$З_{ЗП} = 1147{,}39 \cdot 100 = 114739, \text{ грн.}$$

$З_{МВ}$ – вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = t_{отл} \cdot С_{МЧ}, \text{ грн,} \qquad (3.13)$$

де $t_{отл}$ – трудомісткість налагодження програми на ЕОМ, год.

$C_{МЧ}$ – вартість машино-години ЕОМ, грн/год.

$$З_{МВ} = 504{,}2 \cdot 8 = 4033{,}6 \text{, грн,}$$

$$К_{ПО} = 114739 + 4033{,}6 = 118772{,}6 \text{, грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p} \text{, мес,} \qquad (3.14)$$

де B_k – число виконавців,

F_p – місячний фонд робочого часу (при 40 годинному робочому тижні Fp = 176 годин).

$$T = \frac{1147{,}39}{1 \cdot 176} \approx 6{,}51 \text{ міс.}$$

Висновки: час розробки даного програмного забезпечення складає 1147,39 людино-годин. Таким чином, очікувана тривалість розробки складе 6,51 місяця при 40 годинному робочому тижні (місячний фонд робочого часу 176 годин), а витрати на створення програмного забезпечення складатимуть 118772,6 грн.

ВИСНОВКИ

Метою мого проекту було створення інтернет бібліотеки, яка дозволяє зручно і зрозуміло для користувача читати і купувати книги.

Програма буде дуже корисна для тих, хто не тільки читає книги в інтернеті, але і відразу хоче собі замовити друковану версію для домашньої колекції. Зручність моєї програми в тому, що даний сайт можна запустити майже на будь-якому пристрої з підключенням до інтернету.

Відповідно до мети даної роботи були вирішені всі завдання і вимоги до електронної бібліотеки, а саме:

- Зручний і зрозумілий графічний інтерфейс
- Можливість реєстрації користувача і особистий кабінет
- Можливість читання та покупки книг
- Можливість редагування книг
- Можливість блокування користувачів адміністратором

Визначено трудомісткість розробленої інформаційної системи (1147,39 людино-годин), проведений підрахунок вартості роботи по створенню програми (118772,6 грн) та розраховано час на його створення (6,51 міс).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Horstmann C. Core Java Volume I – Fundamentals / Cay Horstmann., 2018.

2. Bloch J. Effective Java / Joshua Bloch., 2017. – 416 с.

3. Walls C. Spring in Action / Craig Walls., 2018.

4. Duckett J. HTML and CSS : Design and Build Websites / Jon Duckett.

5. Robson E. Head First HTML and CSS: A Learner's Guide to Creating Standards-Based Web Pages / E. Robson, E. Freeman., 2012. – 768 с.

6. Database Concepts / D.Kroenke, D. Auer, S. Vandenberg, R. Yoder., 2019. – 552 с.

7. Introduction to Java Spring Boot: Learning By Coding / [A. Henley, D. Wolf, A. Ankomah та ін.]., 2019. – 132 с.

8. Ansari H. Learn Spring Boot: Designed for Java developers to understand and develop production-ready spring applications with minimum configurations. / H. Ansari., 2021. – 626 с.

9. Coronel C. Database Systems: Design, Implementation, & Management 13th Edition / Carlos Coronel., 2018. – 816 с.

10. McKay E. UI is Communication: How to Design Intuitive, User Centered Interfaces by Focusing on Effective Communication 1st Edition / Everett McKay., 2013. – 378 с.

11. Farrington T. UX Design 2020: The Ultimate Beginner's Guide to User Experience / T. Farrington, T. Brooks, L. Ferrante., 2020.

12. Fields D. IntelliJ IDEA in Action: Covers IDEA v.5 / D. Fields, S. Saunders, E. Belyaev., 2006. – 450 с.

13. PETERSON R. PHP AND MY-SQL A FULL BASICS & ADVANCED / ROBERT PETERSON., 2019. – 1338 с.

14. Shields W. SQL QuickStart Guide: The Simplified Beginner's Guide to Managing, Analyzing, and Manipulating Data With SQL / Walter Shields., 2019. – 251 с.

15. Loy M. Learning Java: An Introduction to Real-World Programming with Java 5th Edition / M. Loy, P. Niemeyer, D. Leuck., 2020. – (518).

16. Leonard A. Java Coding Problems: Improve your Java Programming skills by solving real-world coding challenges / Anghel Leonard., 2019. – 816 с.

17. Schildt H. Java: A Beginner's Guide, Eighth Edition 8th Edition / Herbert Schildt., 2018. – 720 с.

18. Mellor A. Java OOP Done Right: Create object oriented code you can be proud of with modern Java Kindle Edition / Alan Mellor., 2021. – 220 с.

ДОДАТОК А

КОД ПРОГРАМИ

```
public class BookEditController {
   private final BookDAO bookDAO;
   @PostMapping
   public void postBookEdit(HttpServletRequest req,HttpServletResponse resp) throws IOException {
      String cost = req.getParameter("cost");
      BigDecimal bigDecimal = new BigDecimal(cost);
      String idbooks = req.getParameter("editBookId");
      Long longIdBooks = Long.valueOf(idbooks);
      Book book = Book.builder()
            .author(req.getParameter("author"))
            .cost(bigDecimal)
            .idbooks(longIdBooks)
            .title(req.getParameter("title")).build();
      bookDAO.update(book);
      resp.sendRedirect("/main");
   }
}

public class CashController extends HttpServlet {
   private final UserDAO userDAOImpl;
```

```
    @PostMapping
    public void postMagic(HttpServletRequest req, HttpServletResponse resp) throws IOException {
        HttpSession session = req.getSession();
        Long userId = Long.valueOf(String.valueOf(session.getAttribute("userId")));
        BigDecimal cash_add = new BigDecimal(String.valueOf(req.getParameter("cash_add")));
        userDAOImpl.updateCash(userId, cash_add);
        session.setAttribute("userCash", new BigDecimal(
                String.valueOf(session.getAttribute("userCash"))
        ).subtract(cash_add));
        resp.sendRedirect("/user");
    }
}

public class DeleteBookController extends HttpServlet {
    private final BookDAO bookDAO;
    @PostMapping
    public String postDelete(HttpServletRequest req) {
        String idBook = req.getParameter("book_id");
        Long longIdBooks = Long.valueOf(idBook);
        bookDAO.deleteById(longIdBooks);
        return "redirect:/main";
    }
}

@ControllerAdvice
public class GlobalExceptionHandlerController {
    @ExceptionHandler(UserExistException.class)
    public final String handleRuntimeException(UserExistException ex){
        return "errorpage/UserExist";
    }
    @ExceptionHandler(LoginException.class)
    public final String handleLoginException(LoginException ex){
        return "errorpage/loginException";
```

```
    }
}


@RequestMapping("/login")
@Controller
@AllArgsConstructor
public class LoginController extends HttpServlet {
    private final UserDAO userDAO;
    @GetMapping
    public String getLogin() {
        return "login";
    }
    @PostMapping
    public String postLogin(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
        String login = req.getParameter("login");
        String password = req.getParameter("password");
        User user = userDAO.findByLogin(login).orElse(null);
        if (user != null && user.getLogin().equals(login) && user.getPassword().equals(password) &&
!user.getIsBanned()) {
            HttpSession session = req.getSession();
            session.setAttribute("user", user);
            session.setAttribute("userCash", user.getCash());
            session.setAttribute("userId", user.getId());
            session.setAttribute("login", login);
            session.setAttribute("password", password);
            session.setAttribute("isLogged", user);
            session.setAttribute("userRole", user.getRole());
            session.setAttribute("lang", "en");
            if (user.getRole().getRole().equals("admin")) {
                session.setAttribute("isAdmin", true);
            }
            return "redirect:/about";
        } else if (user != null && user.getIsBanned()) {
```

```
            req.setAttribute("ban", true);
            return  "login";
        } else {
            req.setAttribute("pass", true);
            return  "login";
        }
    }
}

@RequestMapping("/logout")
@Controller
@AllArgsConstructor
public class LogoutController {
    @GetMapping
    public String getLogout(HttpServletRequest req) throws IOException {
        HttpSession session = req.getSession();
        if(session.getAttribute("login")!=null){
            session.invalidate();
        }
        return "redirect:/login";
    }
}

@RequestMapping("/main")
@Controller
@AllArgsConstructor
public class MainController extends HttpServlet {
    private final BookService bookService;
    @GetMapping
    public String getMain(HttpServletRequest req) {
        HttpSession session = req.getSession();
        String str = String.valueOf(session.getAttribute("lang"));
        List<Book> bookLists = bookService.findAll(str);
```

```
        req.setAttribute("bookList", bookLists);
        return "main";
    }
    @PostMapping
    public String postMain(HttpServletRequest req) {
        HttpSession session = req.getSession();
        System.out.println(req.getParameter("lang"));
        session.setAttribute("lang", req.getParameter("lang"));
        return "about";
    }
}

@RequestMapping("/order")
@Controller
@AllArgsConstructor
public class OrderController {
    private final OrderDAOImpl orderDAOImpl;
    private final UserDAOImpl userDAOImpl;
    @PostMapping
    public String postOrder(HttpServletRequest req) {
        HttpSession session = req.getSession();
        Long userId = Long.valueOf(String.valueOf(session.getAttribute("userId")));
        Long bookId = Long.valueOf(req.getParameter("book_id"));
        BigDecimal bookCost = new BigDecimal(String.valueOf(req.getParameter("book_cost")));

        BigDecimal userCash = new BigDecimal(
                String.valueOf(session.getAttribute("userCash"))
        ).subtract(bookCost);
        if (userCash.compareTo(new BigDecimal("0.00")) < 0) {
            return "redirect:/main";
        }
        orderDAOImpl.orderBook(bookId, userId);
        userDAOImpl.updateCash(userId, bookCost);
```

```
        session.setAttribute("userCash", new BigDecimal(
                String.valueOf(session.getAttribute("userCash"))
        ).subtract(bookCost));
        return "redirect:/main";
    }
}


@RequestMapping("/registration")
@Controller
@AllArgsConstructor
public class RegistrationController extends HttpServlet {
    private final UserDAO userDAO;
    @GetMapping
    public String getRegistration() {
        return "registration";
    }
    @PostMapping
    public String postRegistration(HttpServletRequest req, HttpServletResponse resp) {
        String login = req.getParameter("login");
        String password = req.getParameter("password");
        String email = req.getParameter("email");
        if (login == null || password == null || email == null || login.isEmpty() || password.isEmpty() ||
email.isEmpty()) {
            req.getRequestDispatcher("error.jsp");
        }
        if (userDAO.findByLogin(login).isPresent()) {
            throw new UserExistException();
        }
        if (userDAO.findByEmail(email).isPresent()) {
            throw new UserExistException();
        }
        User user = User.builder()
                .login(login)
                .password(password)
```

```
                .email(email)
                .role(null)
                .isBanned(false)
                .build();
        userDAO.create(user);
        return "main";
    }
}

@RequestMapping("/test")
@Controller
@AllArgsConstructor
public class TestController {
    private final UserDAO userDAOImpl;
    @GetMapping
    public String getTest(HttpServletRequest req) throws ServletException, IOException {
        User userLists = userDAOImpl.findById(7L).orElseThrow(RuntimeException::new);
        req.setAttribute("user",userLists);
        return "test.jsp";
    }
}

@RequestMapping("/user")
@Controller
@AllArgsConstructor
public class UserController extends HttpServlet {
     private final BookService bookService;
        private final UserDAO userDAOImpl;
    @GetMapping
    public String getUser(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
        HttpSession session = req.getSession();
```

```
        User userId =
userDAOImpl.findById(Long.valueOf(String.valueOf(session.getAttribute("userId")))).orElseThrow(Run
timeException::new);

        req.setAttribute("user", userId);

        String str = String.valueOf(session.getAttribute("lang"));

        List<Book> bookLists =
bookService.findAllUser(Long.valueOf(String.valueOf(session.getAttribute("userId"))), str);

        req.setAttribute("bookList", bookLists);

        Map<Book, Long> booksToCount =
bookLists.stream().collect(Collectors.groupingBy(Function.identity(), Collectors.counting()));

        req.setAttribute("booksToCount", booksToCount);

        req.setAttribute("totalPrice", bookLists.stream().map(Book::getCost).reduce(BigDecimal.ZERO,
BigDecimal::add));

        return "user";

    }

}


@RequestMapping("/userList")

@Controller

@AllArgsConstructor

public class UserListController extends HttpServlet {

    private final UserDAO userDAO;

    @GetMapping

    public String getList(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {

        List<User> userList = userDAO.findALL();

        req.setAttribute("userList", userList);

        return "userList";

    }

    @PostMapping

    public String postList(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {

        Long userId = Long.valueOf(req.getParameter("userId"));

        Boolean isBanned = Boolean.valueOf(req.getParameter("isBanned"));

        userDAO.updateStatus(userId, !isBanned);

        return "redirect:/userList";
```

```
    }
}
public class WebApplication {
    private static File getRootFolder() {
        try {
            File root;
            String runningJarPath =
WebApplication.class.getProtectionDomain().getCodeSource().getLocation().toURI().getPath().replaceAl
l("\\\\", "/");
            int lastIndexOf = runningJarPath.lastIndexOf("/target/");
            if (lastIndexOf < 0) {
                root = new File("");
            } else {
                root = new File(runningJarPath.substring(0, lastIndexOf));
            }
            System.out.println("application resolved root folder: " + root.getAbsolutePath());
            return root;
        } catch (URISyntaxException ex) {
            throw new RuntimeException(ex);
        }
    }
    public static void main(String[] args) throws Exception {
        File root = getRootFolder();
        System.setProperty("org.apache.catalina.startup.EXIT_ON_INIT_FAILURE", "true");
        Tomcat tomcat = new Tomcat();
        Path tempPath = Files.createTempDirectory("tomcat-base-dir");
        tomcat.setBaseDir(tempPath.toString());
        String webPort = System.getenv("PORT");
        if (webPort == null || webPort.isEmpty()) {
            webPort = "8080";
        }
        tomcat.setPort(Integer.valueOf(webPort));
        File webContentFolder = new File(root.getAbsolutePath(), "src/main/webapp/");
        if (!webContentFolder.exists()) {
```

```
            webContentFolder = Files.createTempDirectory("default-doc-base").toFile();
        }
        StandardContext ctx = (StandardContext) tomcat.addWebapp("",
webContentFolder.getAbsolutePath());
        ctx.setParentClassLoader(WebApplication.class.getClassLoader());
        System.out.println("configuring app with basedir: " + webContentFolder.getAbsolutePath());
        File additionWebInfClassesFolder = new File(root.getAbsolutePath(), "target/classes");
        WebResourceRoot resources = new StandardRoot(ctx);
        WebResourceSet resourceSet;
        if (additionWebInfClassesFolder.exists()) {
            resourceSet = new DirResourceSet(resources, "/WEB-INF/classes",
additionWebInfClassesFolder.getAbsolutePath(), "/");
            System.out.println("loading WEB-INF resources from as '" +
additionWebInfClassesFolder.getAbsolutePath() + "'");
        } else {
            resourceSet = new EmptyResourceSet(resources);
        }
        resources.addPreResources(resourceSet);
        ctx.setResources(resources);
        tomcat.start();
        tomcat.getServer().await();
    }
}
```

ДОДАТОК Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	Опис
Появнювальні документи	
	Пояснювальна записка кваліфікаційної роботи. Документ Word.
	Пояснювальна записка кваліфікаційної роботи в форматі PDF.
Програма	
	Архів. Містить коди програми.
Презентація	
	Презентація кваліфікаційної роботи.