Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования «Омский государственный технический университет»

Факультет «Информационных технологий и компьютерных систем»

Кафедра «Информатика и вычислительная техника»

**Расчётно-графическая работа**

по дисциплине «Проектирование и тестирование программного обеспечения»

Тема: «Электронный каталог для библиотеки»

Выполнил: студент группы ПИ-182

Шкабров Д.А.

Проверил: ассистент кафедры ИВТ

Блохин А. В.

Омск 2020

**Содержание**

[Описание задания 3](#_Toc40479720)

[Выбранные средства выполнения 4](#_Toc40479721)

[Описание основных модулей и функций 5](#_Toc40479722)

[Пример работы программы 7](#_Toc40479723)

[Результаты тестирования с описанием 11](#_Toc40479724)

[Заключение 13](#_Toc40479725)

[Приложение 14](#_Toc40479726)

# Описание задания

Разработать электронный каталог для библиотеки CRUD-информационной системы с REST API сервером под Web и базой данных.

# Выбранные средства выполнения

Для выполнения расчетно-графической работы были использованы языки программирования Java и JavaScript. Для BACK-end был использован Maven и MySQL Server. Для FRONT-end был использован Maven и Tomcat Server. Набор исходного кода осуществлялся в интегрированной среде разработки IntelliJ IDEA 2020.1.

# Описание основных модулей и функций

1. register:

Пользователь заполняет свои данные на сайте. Данные отправляются на сервер в виде Json строки, Json строка преобразуется в String и отправляется в базу данных с помощью запроса в таблицу “clients”. При наличии в базе данных данного пользователя возвращается ошибка.

1. login:

Пользователь заполняет свои данные на сайте. Данные отправляются на сервер в виде Json строки, Json строка преобразуется в String и отправляется в базу данных с помощью запроса в таблицу “login”. Если в данной таблице “login” уже есть данные, то возвращается ошибка.

1. outputLogin:

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса очищается таблица “login” от данных.

1. checkLogin:

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса проверяется наличие данных в таблице “login”. Результат отправляется на сервер в виде Json строки.

1. search:

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса производится поиск книги, соответствующей запросу пользователя в таблице “books”. Далее информация о книге отправляется на FRONT-end.

1. checkSearch:

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса проверяется, нахождение данных в таблице “searchbook”.

1. searchOutput

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса пользователю отправляется вся информация о книге находящейся в таблице “searchbook ”.

1. booking:

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса в таблице “books” значение статуса бронирования меняется на 1, далее в таблице данные из таблицы “searchbook” удаляются и заносится данные о книге в таблицу “rbooks”.

1. reservations:

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса на FRONT-end отправляется ответ о забронированных книгах данного пользователя из таблицы “rbooks”.

1. getBooks:

На сервер приходит сообщение в виде Json строки, далее Json строка преобразуется в String, и с помощью запроса на FRONT-end отправляются все книги, находящиеся в таблице “books”.

# Пример работы программы

Работа программы представлена на рисунках 1–8.

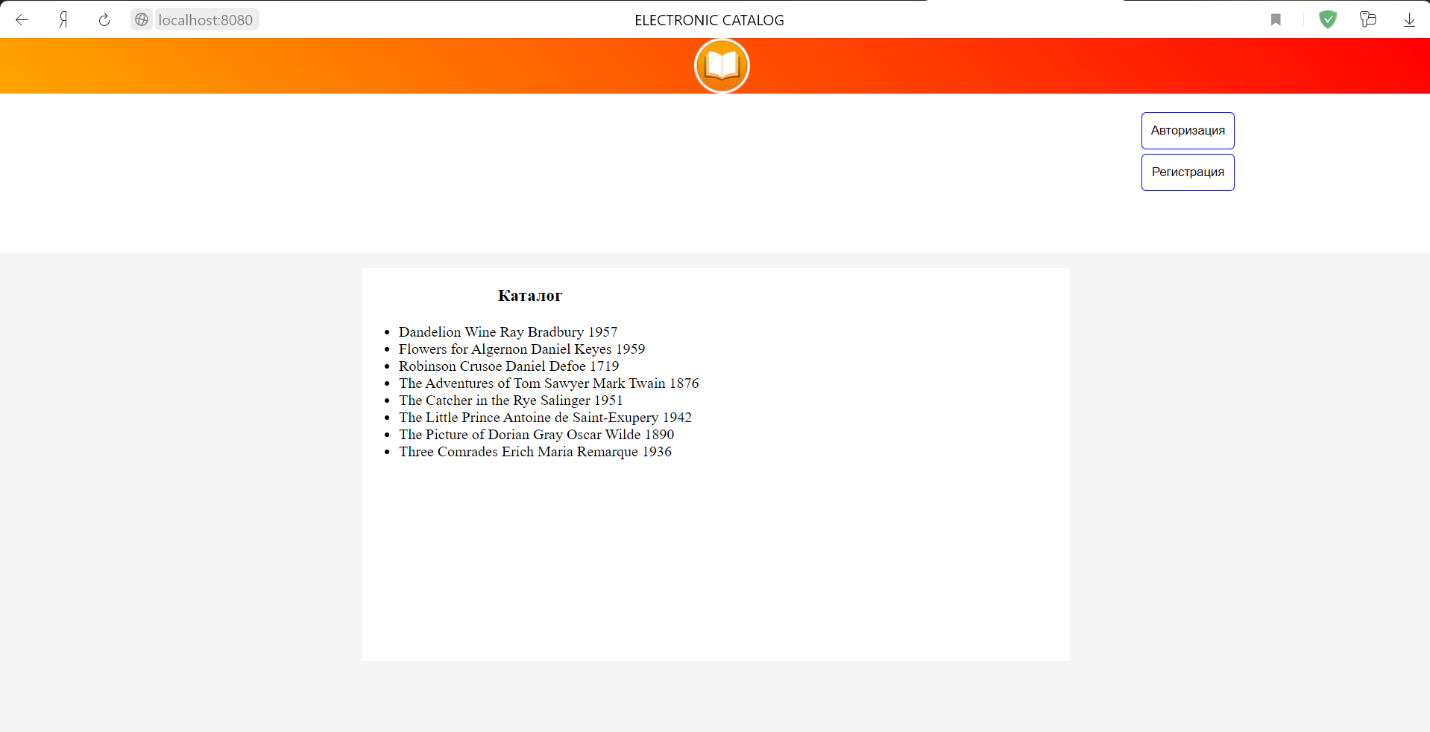


Рисунок 1 – Главная страница незарегистрированного пользователя

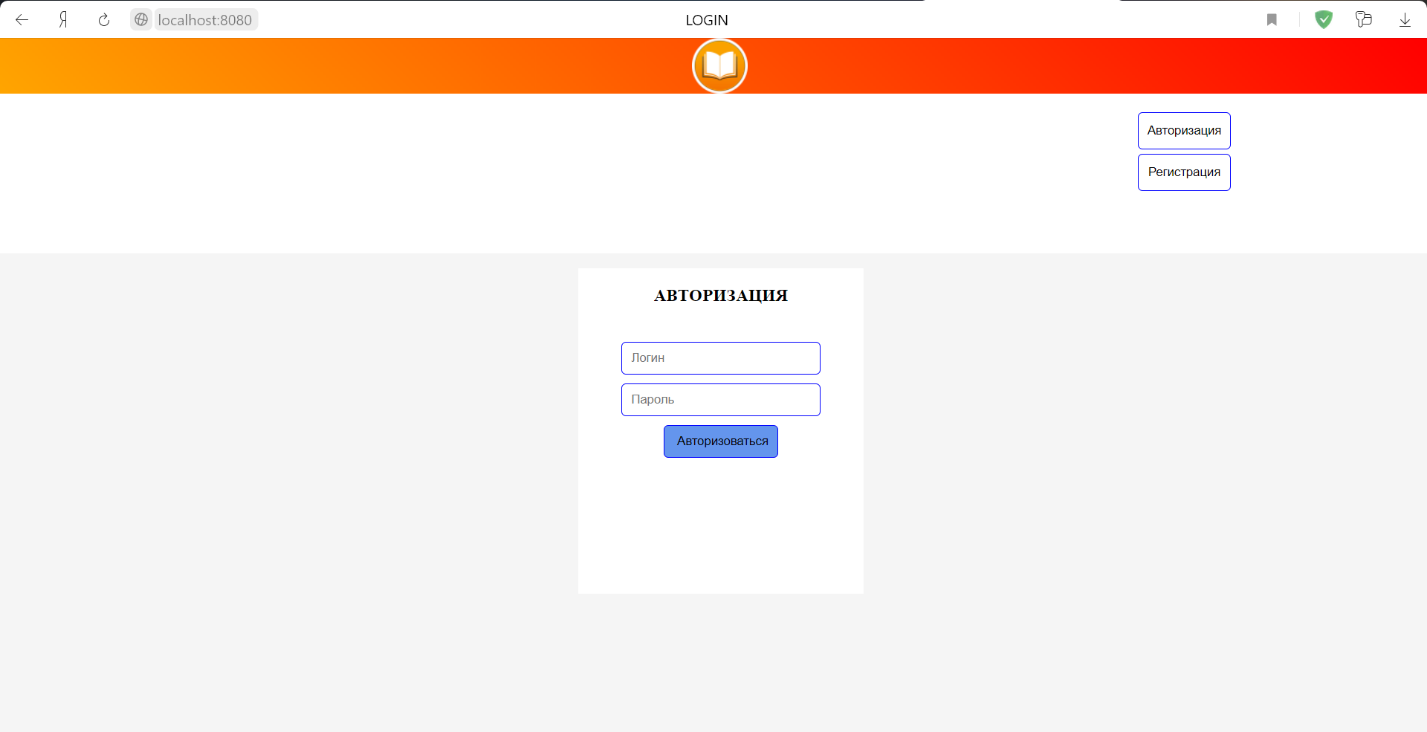


Рисунок 2 – Страница авторизации

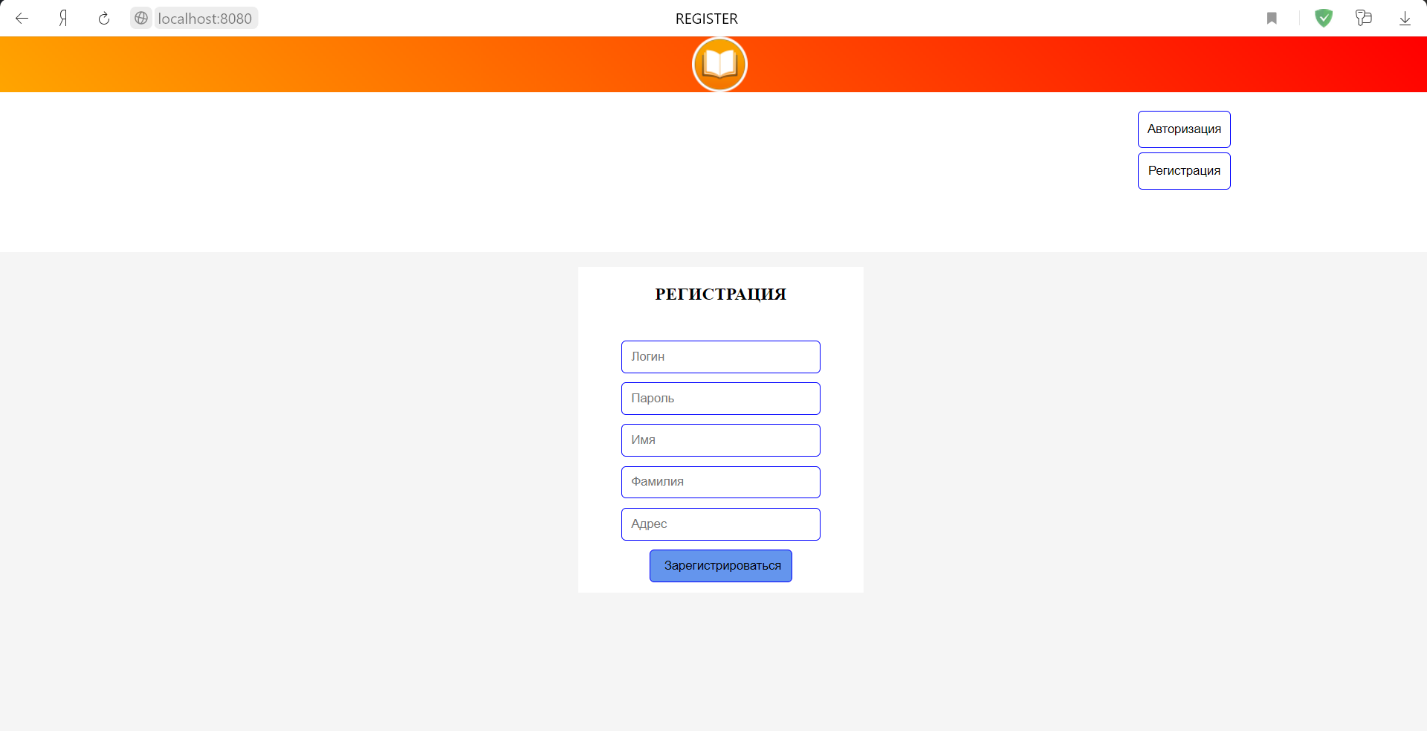


Рисунок 3 – Страница регистрации

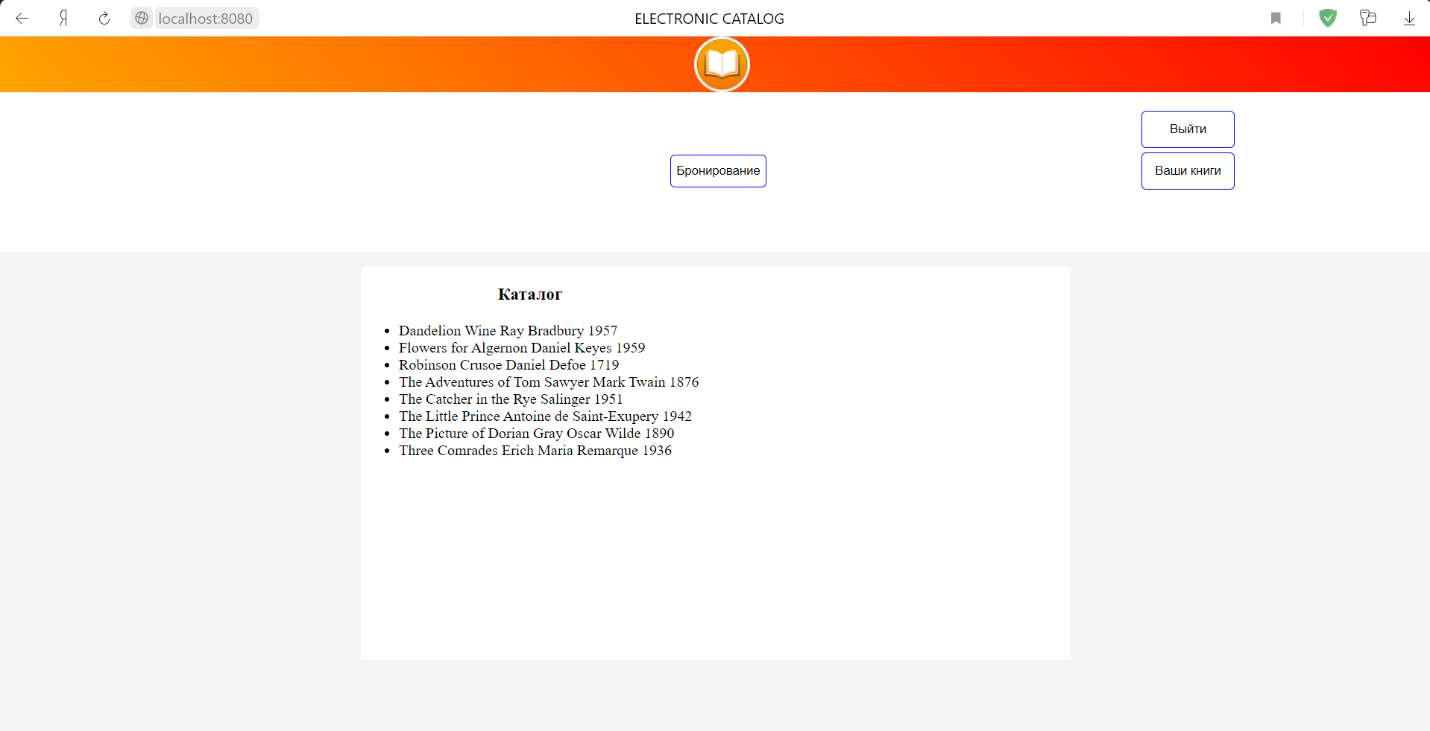


Рисунок 4 – Главная страница зарегистрированного пользователя

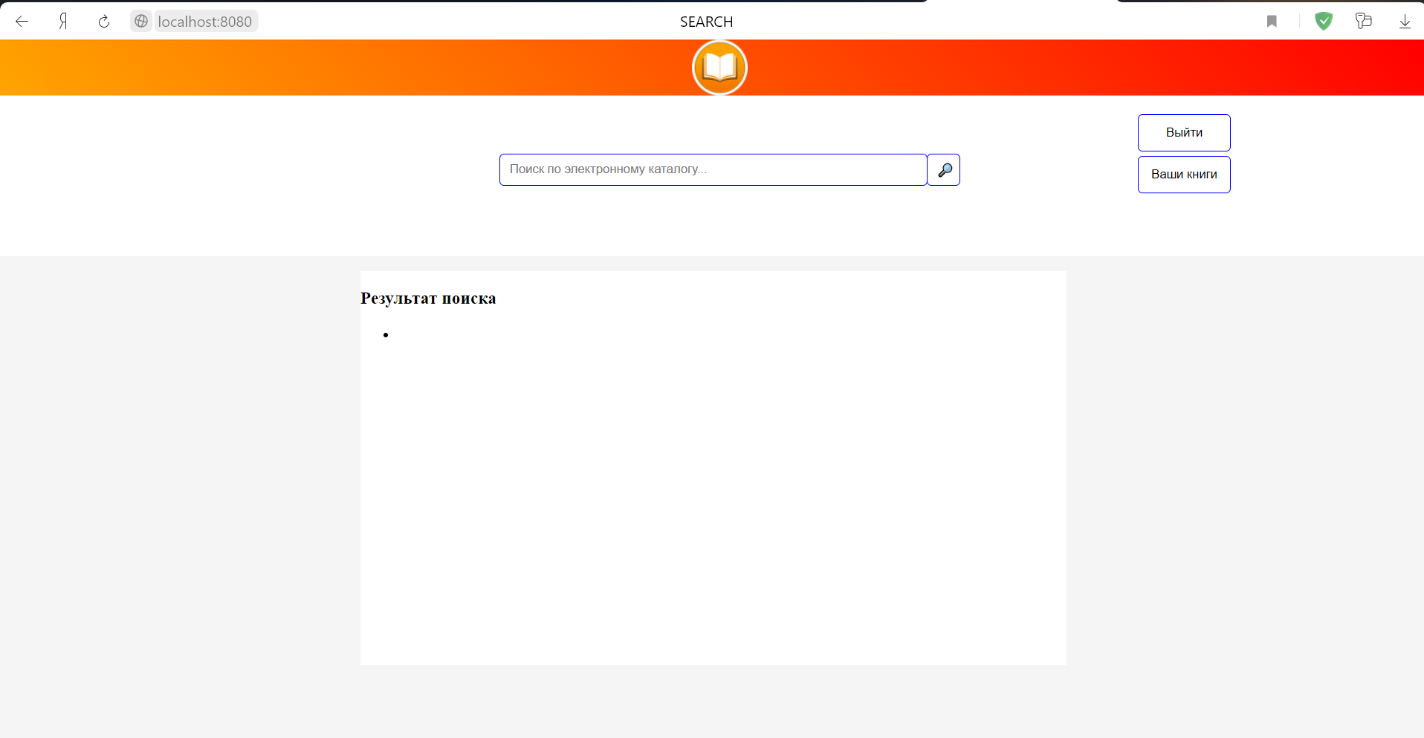


Рисунок 5 – Страница поиска без введенных данных

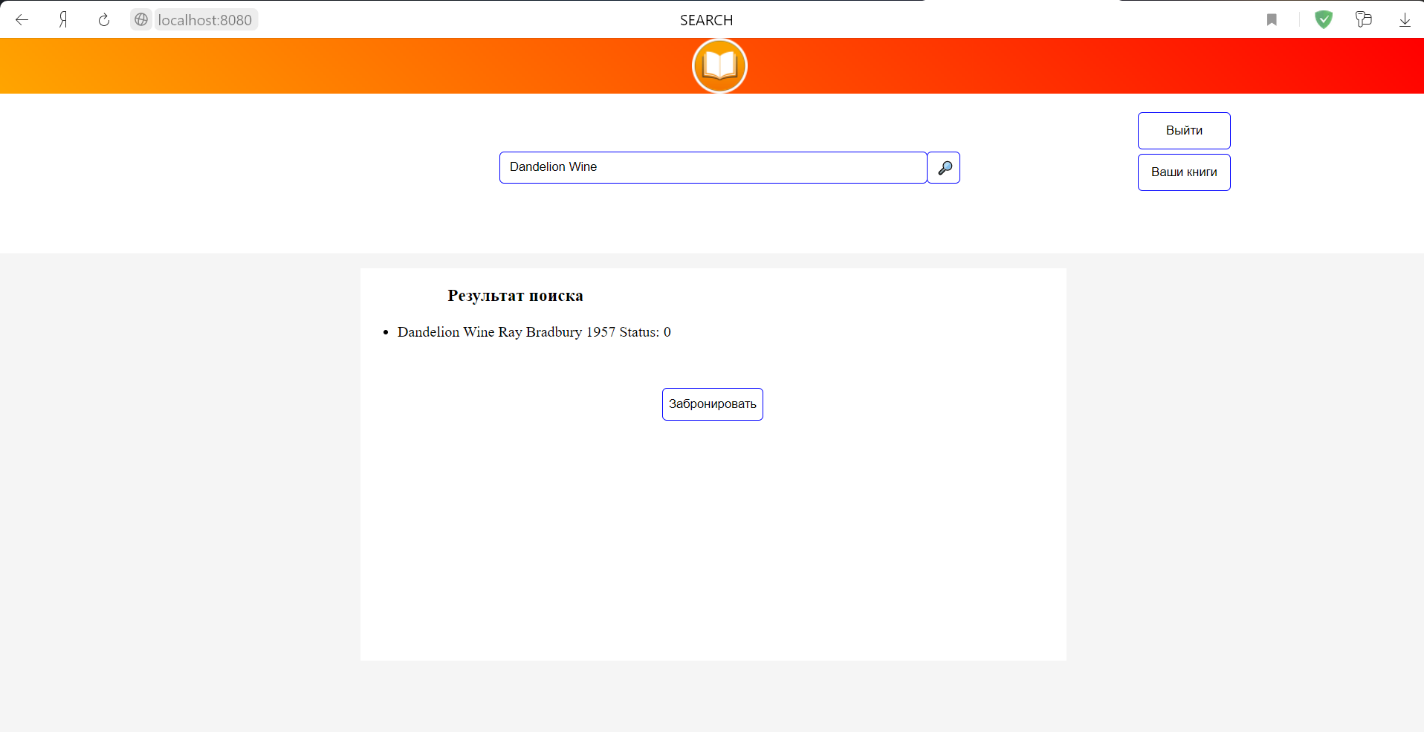
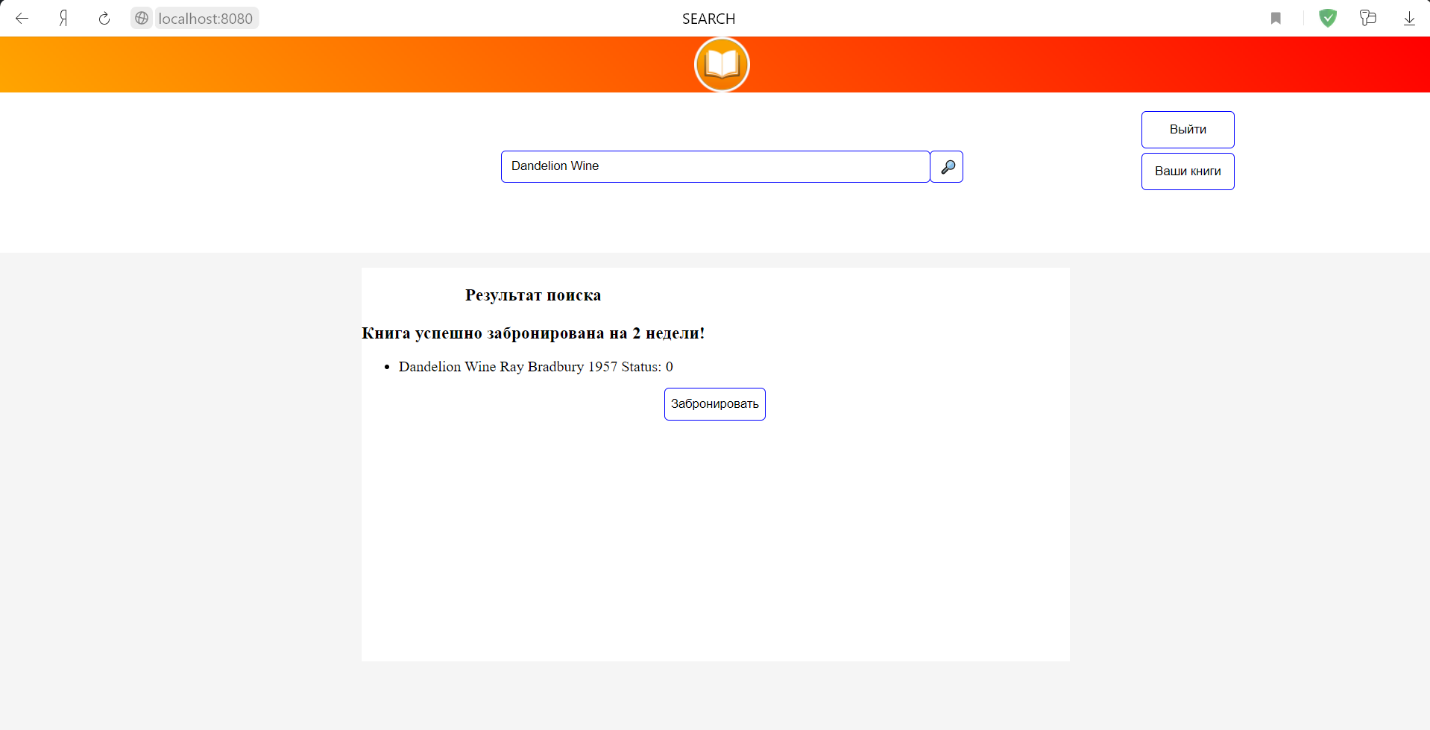
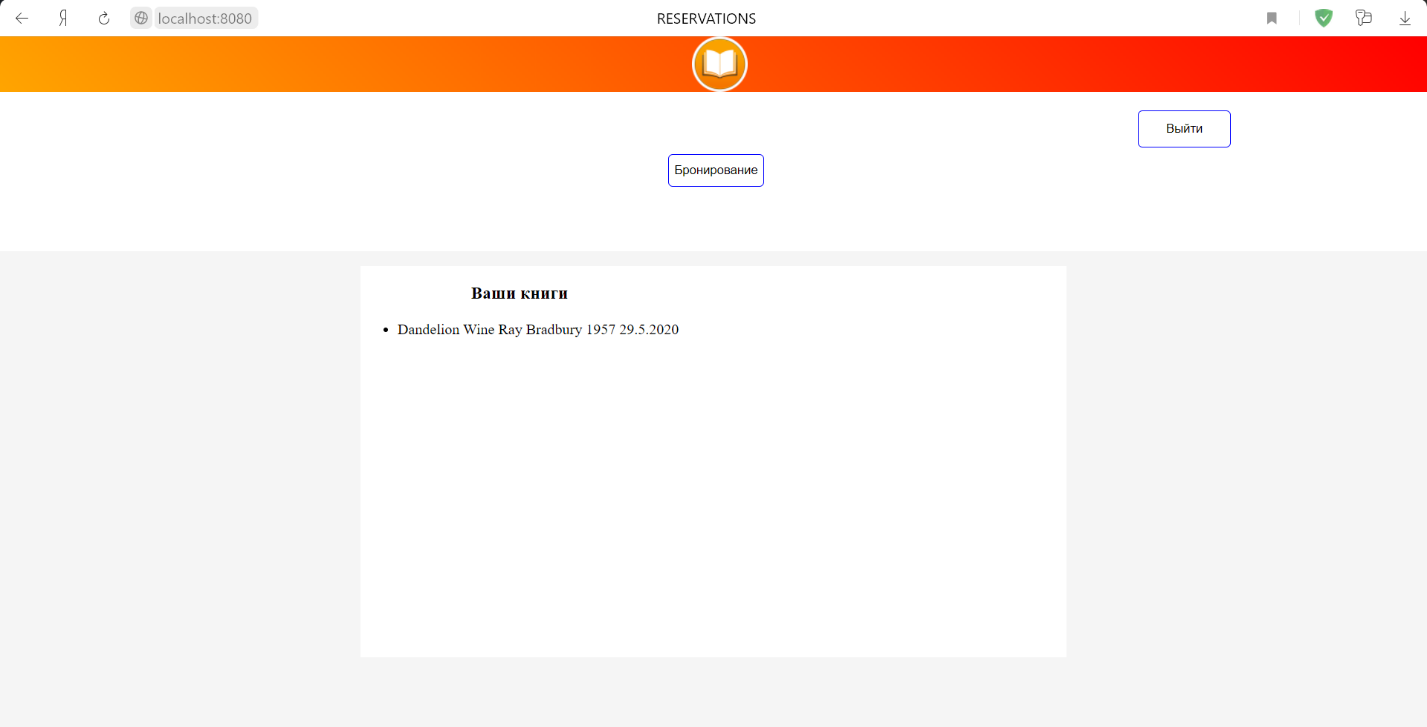


Рисунок 6 – Страница поиска с введенными данными

Рисунок 7 – Страница поиска при бронировании

Рисунок 8 – Страница ваших бронированных книг

# Результаты тестирования с описанием

Тестирование проводилось с помощью статического анализатора PVS-studio и Unit тестов. С помощью Unit тестов было проведено тестирование только классов: Handler, DataBase и Parser. Классы Main и Client не были протестированы, потому что для работы данных классов нужно подключение через Socket. Результат тестирования представлен на рисунке 9.

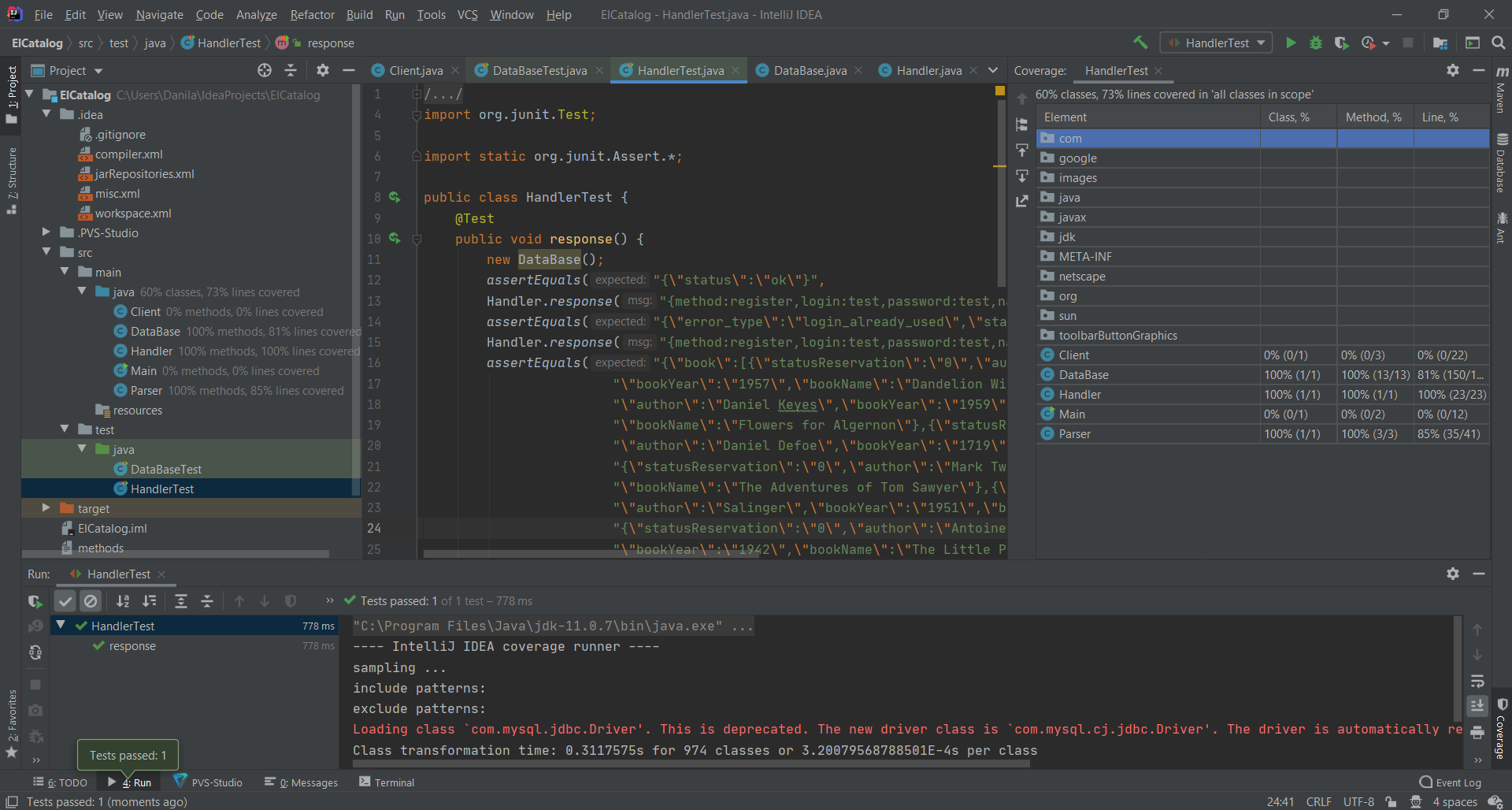


Рисунок 9 – Результат тестирования

Результат проверки статического анализатора кода PVS-studio представлен на рисунке 10.

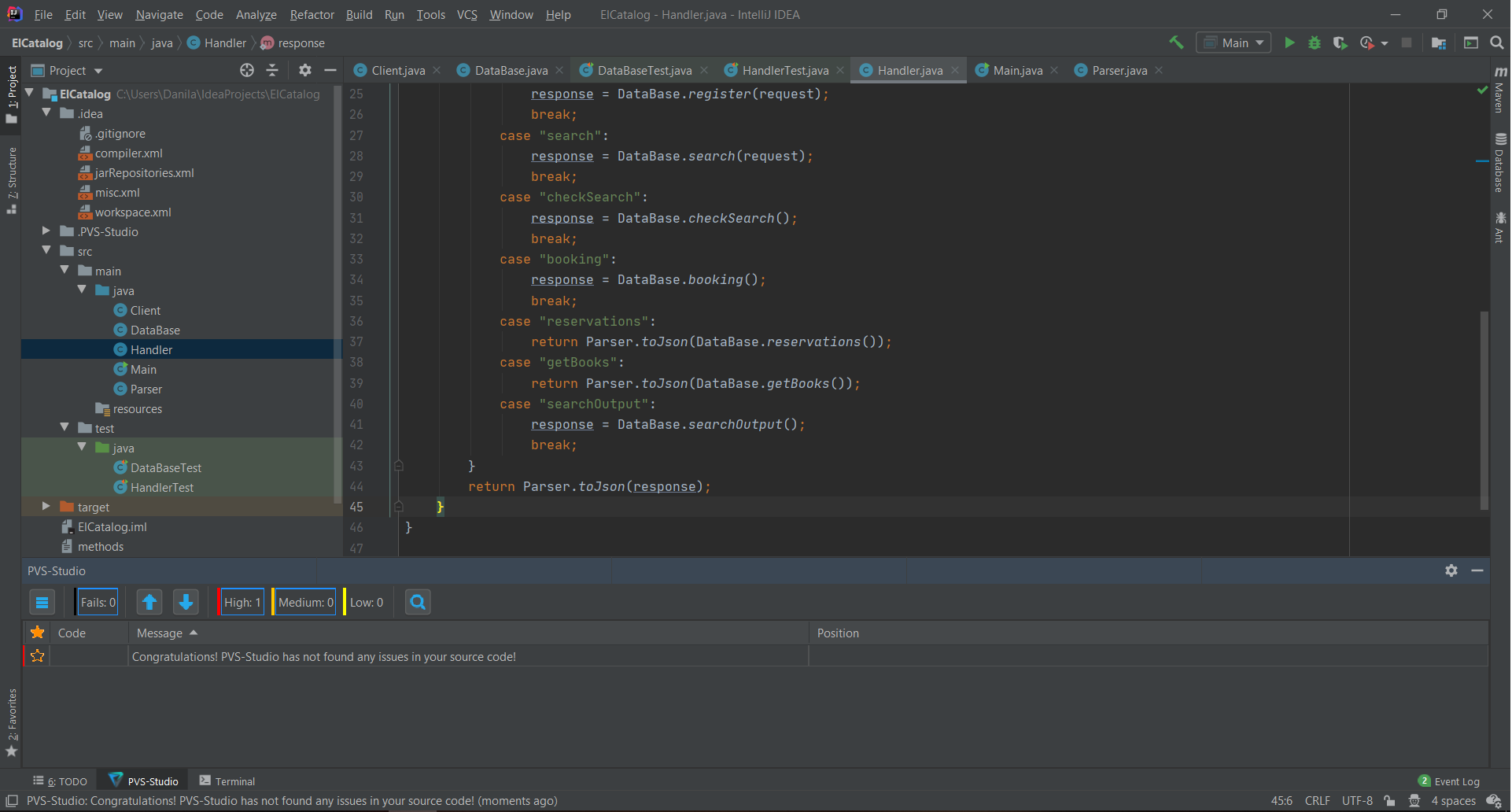


Рисунок 10 – Результат тестирования

Код Unit тестов находится в приложении.

# Заключение

В результате выполнения расчетно-графической работы был создан сайт – электронный каталог для библиотеки, связанный с базой данной. Программные модули были написаны на языках программирования Java и JavaScript, и разработаны на платформе «IntelliJ IDEA». А также код был протестирован анализатором кода PVS-studio и Unit тестами.

# Приложение

// This is a personal academic project. Dear PVS-Studio, please check it.  
  
// PVS-Studio Static Code Analyzer for C, C++, C#, and Java: http://www.viva64.com  
  
import org.junit.Test;  
  
import static org.junit.Assert.\*;  
  
public class HandlerTest {  
 @Test  
 public void response() {  
 new DataBase();  
 *assertEquals*("{\"status\":\"ok\"}",  
 Handler.*response*("{method:register,login:test,password:test,name:test," +  
 "lastName:test,address:test}"));  
 *assertEquals*("{\"error\_type\":\"login\_already\_used\",\"status\":\"error\"}",  
 Handler.*response*("{method:register,login:test,password:test,name:test," +  
 "lastName:test,address:test}"));  
 *assertEquals*("{\"book\":[{\"statusReservation\":\"0\",\"author\":\"Ray Bradbury\"," +  
 "\"bookYear\":\"1957\",\"bookName\":\"Dandelion Wine\"},{\"statusReservation\":\"0\"," +  
 "\"author\":\"Daniel Keyes\",\"bookYear\":\"1959\"," +  
 "\"bookName\":\"Flowers for Algernon\"},{\"statusReservation\":\"0\"," +  
 "\"author\":\"Daniel Defoe\",\"bookYear\":\"1719\",\"bookName\":\"Robinson Crusoe\"}," +  
 "{\"statusReservation\":\"0\",\"author\":\"Mark Twain\",\"bookYear\":\"1876\"," +  
 "\"bookName\":\"The Adventures of Tom Sawyer\"},{\"statusReservation\":\"0\"," +  
 "\"author\":\"Salinger\",\"bookYear\":\"1951\",\"bookName\":\"The Catcher in the Rye\"}," +  
 "{\"statusReservation\":\"0\",\"author\":\"Antoine de Saint-Exupery\"," +  
 "\"bookYear\":\"1942\",\"bookName\":\"The Little Prince\"},{\"statusReservation\":\"0\"," +  
 "\"author\":\"Oscar Wilde\",\"bookYear\":\"1890\"," +  
 "\"bookName\":\"The Picture of Dorian Gray\"},{\"statusReservation\":\"0\"," +  
 "\"author\":\"Erich Maria Remarque\",\"bookYear\":\"1936\",\"bookName\":\"Three Comrades\"}]}",  
 Handler.*response*("{method:getBooks}"));  
 *assertEquals*("{\"status\":\"ok\"}",  
 Handler.*response*("{method:login,login:test,password:test}"));  
 *assertEquals*("{\"error\_type\":\"wrong\_password\",\"status\":\"error\"}",  
 Handler.*response*("{method:login,login:test,password:1}"));  
 *assertEquals*("{\"idClients\":\"3\"}",  
 Handler.*response*("{method:checkLogin}"));  
 *assertEquals*("{\"statusReservation\":\"0\",\"author\":\"Daniel Defoe\",\"bookYear\":\"1719\"," +  
 "\"bookName\":\"Robinson Crusoe\"}",  
 Handler.*response*("{method:search,bookName:Robinson Crusoe}"));  
 *assertEquals*("{\"idBooks\":\"1\"}",  
 Handler.*response*("{method:checkSearch}"));  
 *assertEquals*("{\"statusReservation\":\"0\",\"author\":\"Daniel Defoe\",\"bookYear\":\"1719\"," +  
 "\"bookName\":\"Robinson Crusoe\"}",  
 Handler.*response*("{method:searchOutput}"));  
 *assertEquals*("{}",  
 Handler.*response*("{method:booking}"));  
 *assertEquals*("{\"book\":[{\"date\":\"29.5.2020\",\"author\":\"Daniel Defoe\",\"bookYear\":\"1719\"," +  
 "\"bookName\":\"Robinson Crusoe\"}]}",  
 Handler.*response*("{method:reservations}"));  
 *assertEquals*("{}",  
 Handler.*response*("{method:outputLogin}"));  
 }  
}