МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Веб-технологии»

Тема: REST-приложение управления библиотекой

Студент гр. 8304 _____ Сергеев А. Д.

Преподаватель _____ Беляев С. А.

Санкт-Петербург

2020

Цель работы

Изучение взаимодействия клиентского приложения с серверной частью, освоение шаблонов web-страниц, формирование навыков разработки динамических HTML-страниц, освоение принципов построения приложений с насыщенным интерфейсом пользователя.

Основные теоретические сведения

CSS – язык описания внешнего вида документа, написанного с использованием языка разметки, используетсся как средство оформления внешнего вида HTML-страниц.

Express – это минималистичный и гибкий web-фреймворк для приложений Node.js, предоставляющий обширный набор функций для мобильных и web-приложений.

Pug – модуль, позволяющий использовать шаблоны для HTML-страниц.

REST – стиль взаимодействия компонентов распределенного приложения. В рамках лабораторной работы – браузера и сервера web-приложения. Для взаимодействия используются стандартные методы:

GET – получение записи (записей)

POST – добавление записи

PUT – обновление или добавление записи

DELETE – удаление записи

Общая формулировка задачи

Необходимо создать web-приложение управления домашней библиотекой, которое предоставляет список книг, их можно отфильтровать

по признакам “в наличии”, “возврат просрочен”, есть возможность выдать книгу для чтения и вернуть книгу.

Основные требования следующие:

1. Начальное состояние библиотеки хранится в JSON-файле на сервере. Текущее состояние — в переменной в памяти сервера.

2. В качестве сервера используется Node.JS модулем express.

3. В качестве модуля управления шаблонами HTML-страниц используется pug, все web-страницы должны быть сделаны с использованием pug;

4. Предусмотрена страница для списка книг, в списке предусмотрена, фильтрация по дате возврата и признаку «в наличии», предусмотрена возможность добавления и удаления книг.

5. Предусмотрена страница для карточки книги, в которой ее можно от. редактировать (минимум: автор, звание, дата выпуску) и дать читателю вернуть в библиотеку. В карточке книги должно быть очевидно: находится ли книга в библиотеке, кто с взял (имя) и когда должен вернуть (дата).

6. Информация о читателе вводится с использованием всплывающего модульного окна

7. Оформление страниц выполнено с использованием CSS

8. Взаимодействие между браузером и web-сервером осуществляется с использованием REST.

9. Фильтрация списка книг осуществляется с использованием AJAX-запросов

10. Логика приложения реализована на языке JavaScript

Ход работы

1. Используя среду разработки JetBrains WebStorm, были установлены все необходимые расширения.
2. Используя модуль express, был создан и настроен сервер.
3. Были созданы и настроены pug и css файлы.
4. Для упрощения создания пользовательского интерфейса была использована библиотека bootstrap.
5. Разработка интерфейса пользователя.

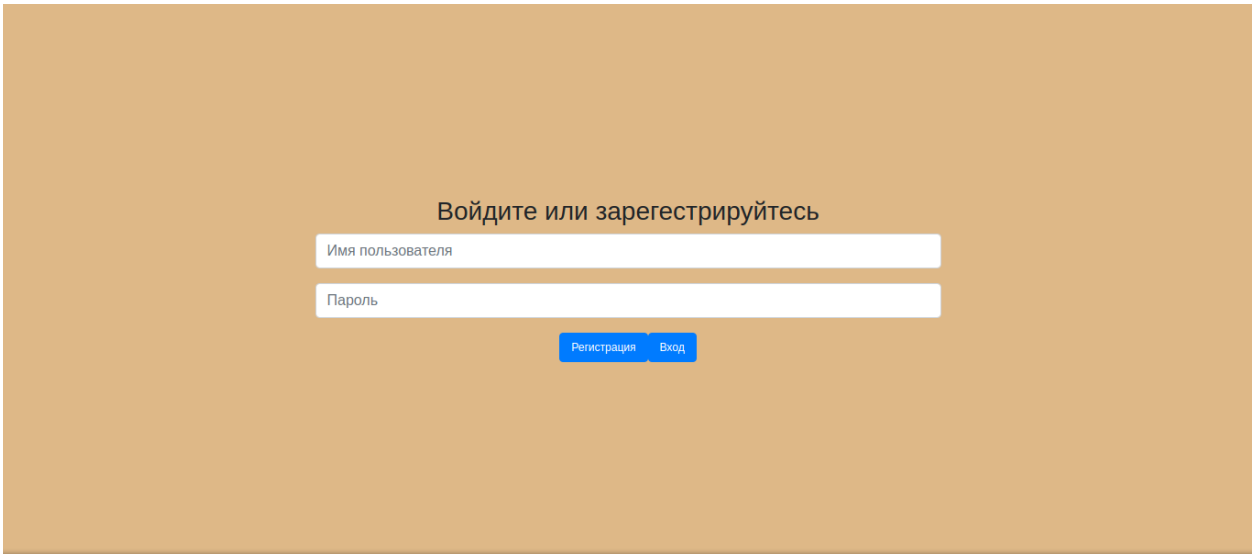

Рисунок 1 – Страница входа на сайт.

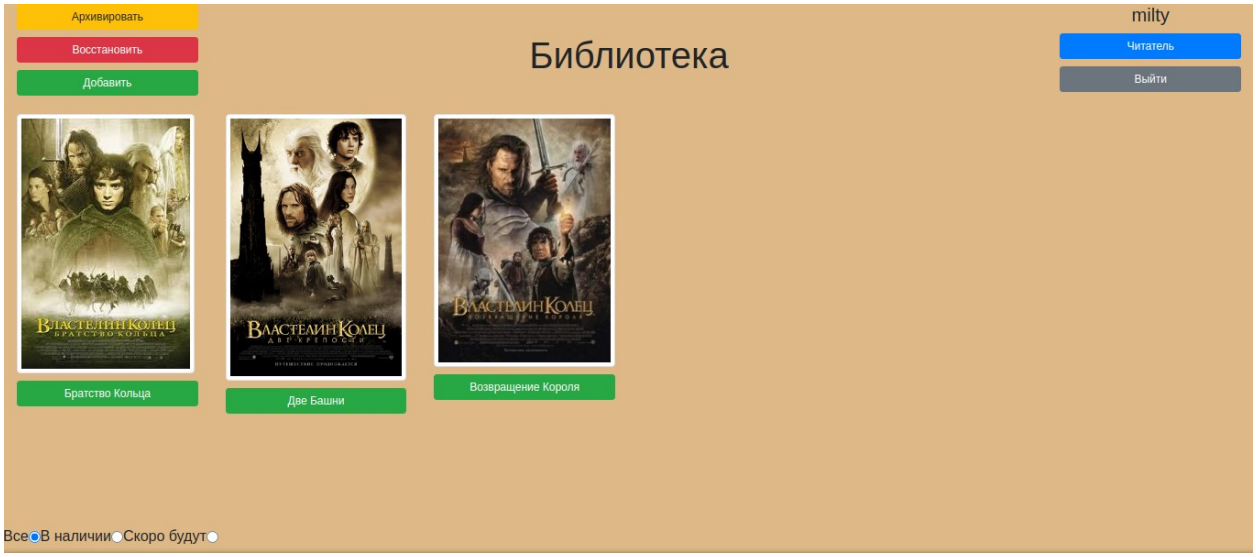

Рисунок 2 – Основная страница сайта.

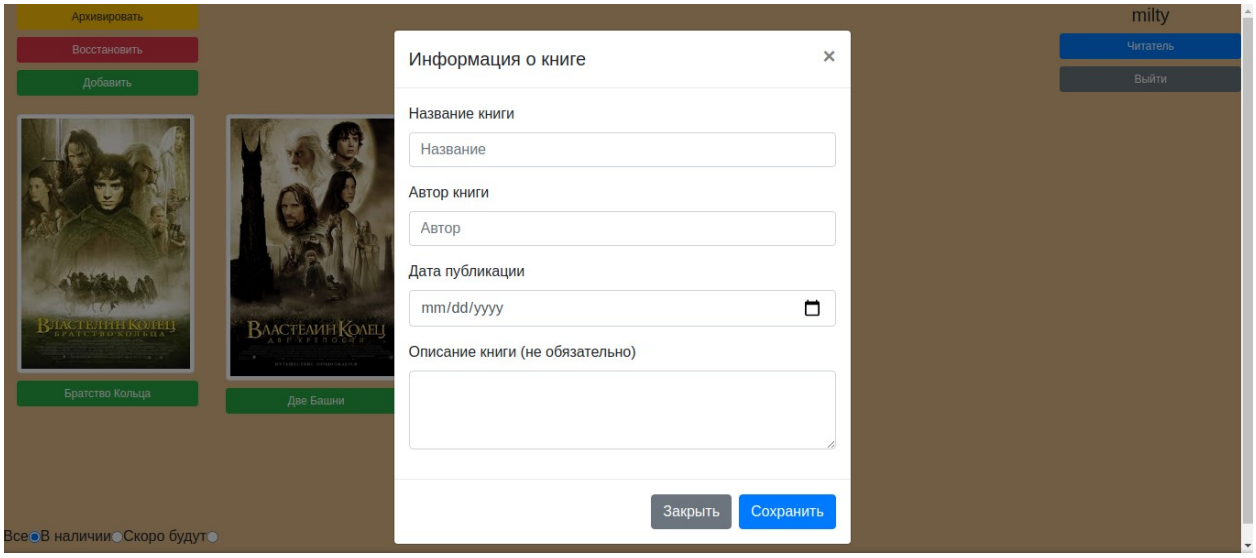

Рисунок 3 – Модальное окно добавления новой книги.

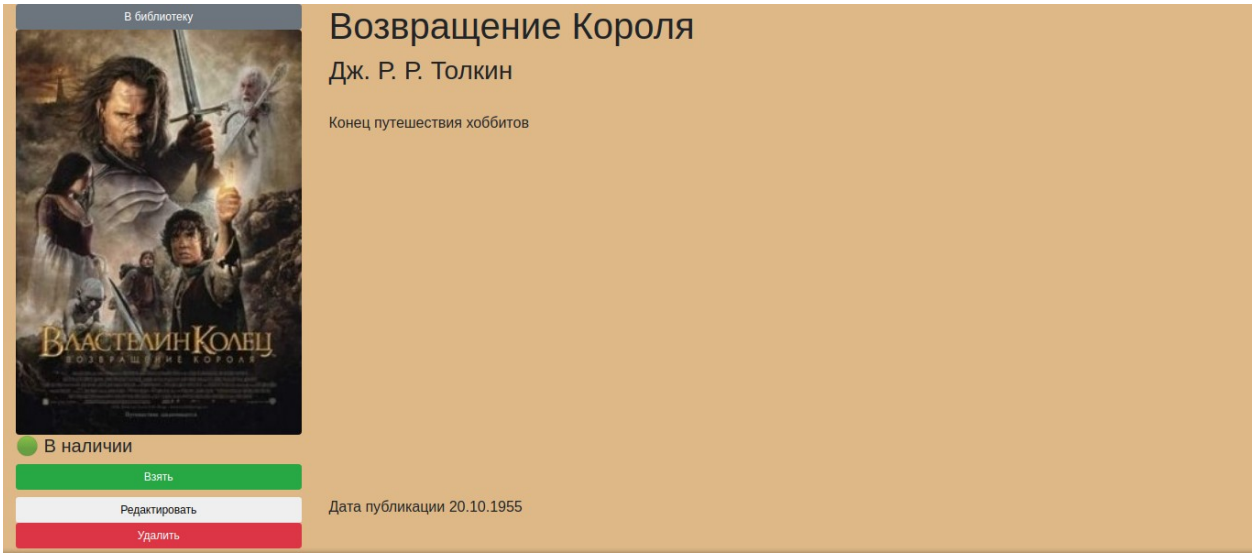

Рисунок 4 – Страница конкретной книги.

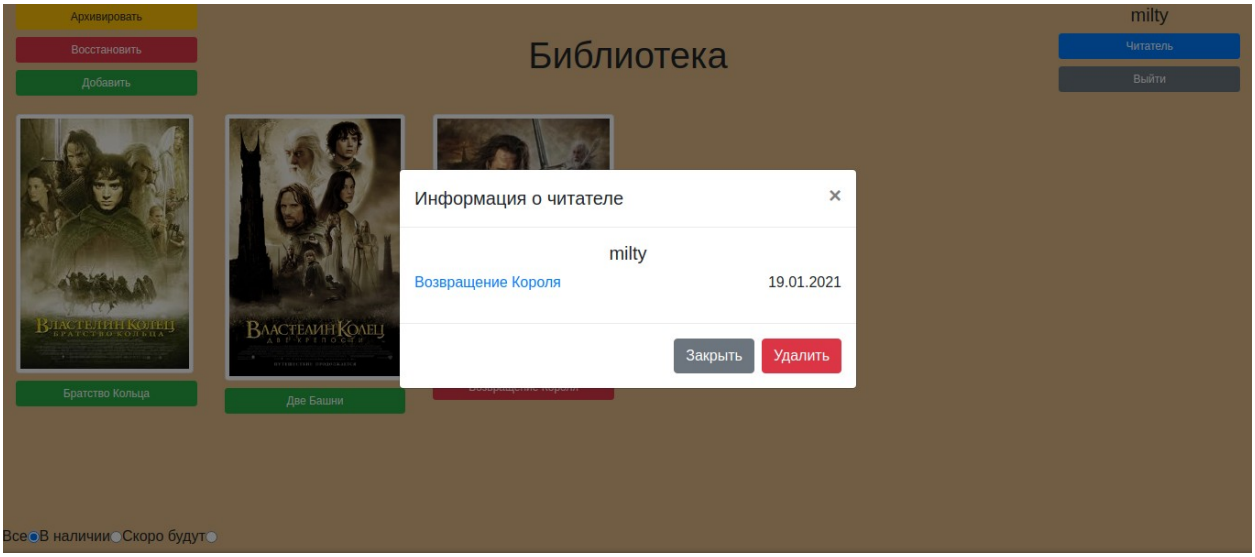

Рисунок 5 – Модальное окно информации о пользователе.

6. Для проекта были созданы следующие файлы.
    1) custom.css – набор стилей для оформления страниц.
    2) base.pug, 404.pug, book.pug, book_modal.pug, lib.pug, login.pug, user_modal.pug – базовая страница и web-страницы для ненайденной страницы, конкретной книги, модального окна редактирования книги, библиотеки, формы входа и регистрации, а также модального окна информации о пользователе соответственно.

3) lib.json, users.json, passwords.json – файлы, хранящие в себе начальные настройки книг, информацию о пользователях и их паролях соответственно.
4) api.js, auth.js, library.js, main.js, population.js, routs.js – содержат функции, обеспечивающие работу сервера, отвечающие за rest api, аутентификацию, библиотеку, основные функции, работу с пользователями и роуты сайта соответственно.
5) book.js, book_modal.js, lib.js, login.js, user_modal.js, accounter.js, recovery.js, book_manager.js – первые 5 содержат скрипты, отвечающие за работу конкретных страниц, а последние 3 — скрипты, генерирующие AJAX-запросы для разных частей сайта.

Вывод

В ходе лабораторной работы был получен опыт работы с pug, css файлами, создании сервера на основе модуля express и генерации AJAX-запросов, на основе создания REST-приложения управления библиотекой