**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «Web-технологии»**

Тема:REST-ПРИЛОЖЕНИЕ УПРАВЛЕНИЯ БИБЛИОТЕКОЙ

|  |  |  |
| --- | --- | --- |
| Студентка гр. 1381 |  | Васильева О. М. |
| Преподаватель |  | Беляев С.А. |

Санкт-Петербург

2023

## Цель и задачи работы.

Целью работы является 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. Для достижения поставленной цели требуется решить следующие задачи:

– разработка интерфейса web-приложения;

– задание стилей для отображения web-приложения с учётом размера экрана (использование на компьютере, на мобильном телефоне);

– создание web-сервера на основе express;

– создание шаблонов web-страниц;

– настройка маршрутов;

– создание json-хранилища;

– обработка REST-запросов;

**Основные теоретические сведения.**

CSS (Cascading Style Sheets – каскадные таблицы стилей) – язык описания внешнего вида документа, написанного с использованием языка разметки, используется 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 и EJS – модули, позволяющие использовать шаблоны для HTMLстраниц. REST (Representational State Transfer – передача состояния представления) – 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 задание.

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 с модулем express.

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

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

5. Предусмотрена страница для карточки книги, в которой ее можно отредактировать (минимум: автор, название, дата выпуска) и выдать читателю или вернуть в библиотеку. В карточке книги должно быть очевидно: находится ли книга в библиотеке, кто ее взял (имя) и когда должен вернуть (дата).

6. Информация о читателе вводится с использованием всплывающего модального диалогового окна (<dialog>).

7. Оформление страниц выполнено с использованием CSS (допустимо использование w3.css).

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

9. Фильтрация списка книг осуществляется с использованием AJAXзапросов.

10. Логика приложения реализована на языке JavaScript.

11. Для всех страниц web-приложения разработан макет интерфейса с использованием Figma (<https://www.figma.com/>).

12. При оформлении элементов управления используются иконки (например, Font Awesome, <https://fontawesome.ru/all-icons/>).

Преимуществом будет создание и использование аутентификации на основе passport.js (http://www.passportjs.org/), в качестве примера можно использовать https://medium.com/devschacht/node-hero-chapter-8-27b74c33a5ce. Преимуществом будет реализация загрузки и отображения обложек книг.

## Выполнение работы.

1. Разработка шаблонов с помощью Figma.

Было создано три основных шаблона для сайта (главная страница,

страница библиотеки и страница книги). См. Рис. 1.

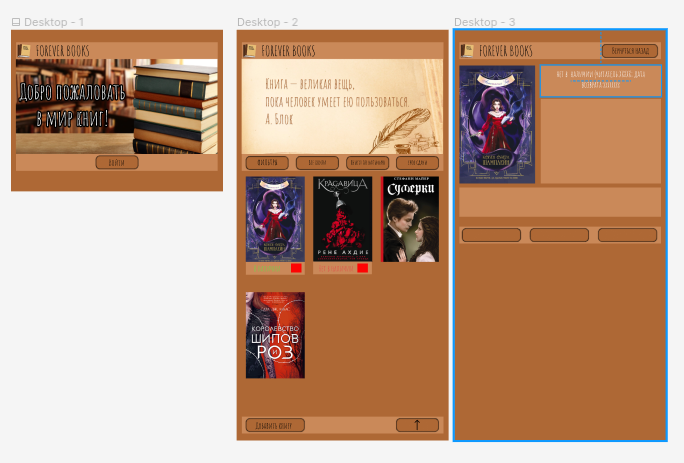


Рис. 1 — шаблоны страницы с использованием Figma

2. Главная страница (input.pug).

Для главной страницы был написан файл input.pug.

На страницы расположена «шапка» сайта с его названием и логотипом в виде книги. Далее идет изображение с приветствием и ниже кнопка входа, которая перенаправляет пользователя в саму библиотеку.

Рис. 2 — Главная страница

3. Страница библиотеки (library.pug).

На странице библиотеки, аналогично главной, сверху расположена «шапка», далее картинка с цитатой. Затем ниже идет лента фильтров (все книги, книги в наличии, срок выдачи). Потом расположена основное содержимое библиотеки — сами книги. В блоке о книги имеется обложка, в наличии/нет в наличии и кнопка удалить книгу (рис. 4).

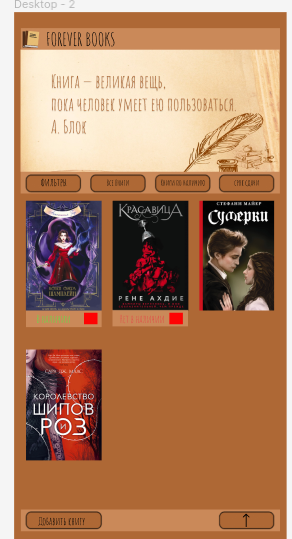


Рис. 3 — страница библиотеки

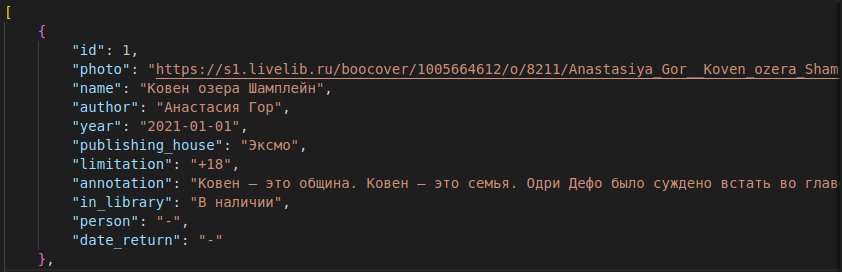


Рис. 4 — вид книги на странице библиотеки

При наведении курсора на «урну» высвечивается «удалить книгу».

На нижней панели имеются две кнопки: добавить книгу, по которой пользователя перенаправляют на страницу добавления книги, и вернуться наверх страницы.

Библиотека включает в себе изначально 4 книги, информация о которых прописана в файле book.json:

Фильтрация книг осуществлена с помощью ajax.

При удалении и добавление книги на странице появляется окно оповещения о сделанном действие:

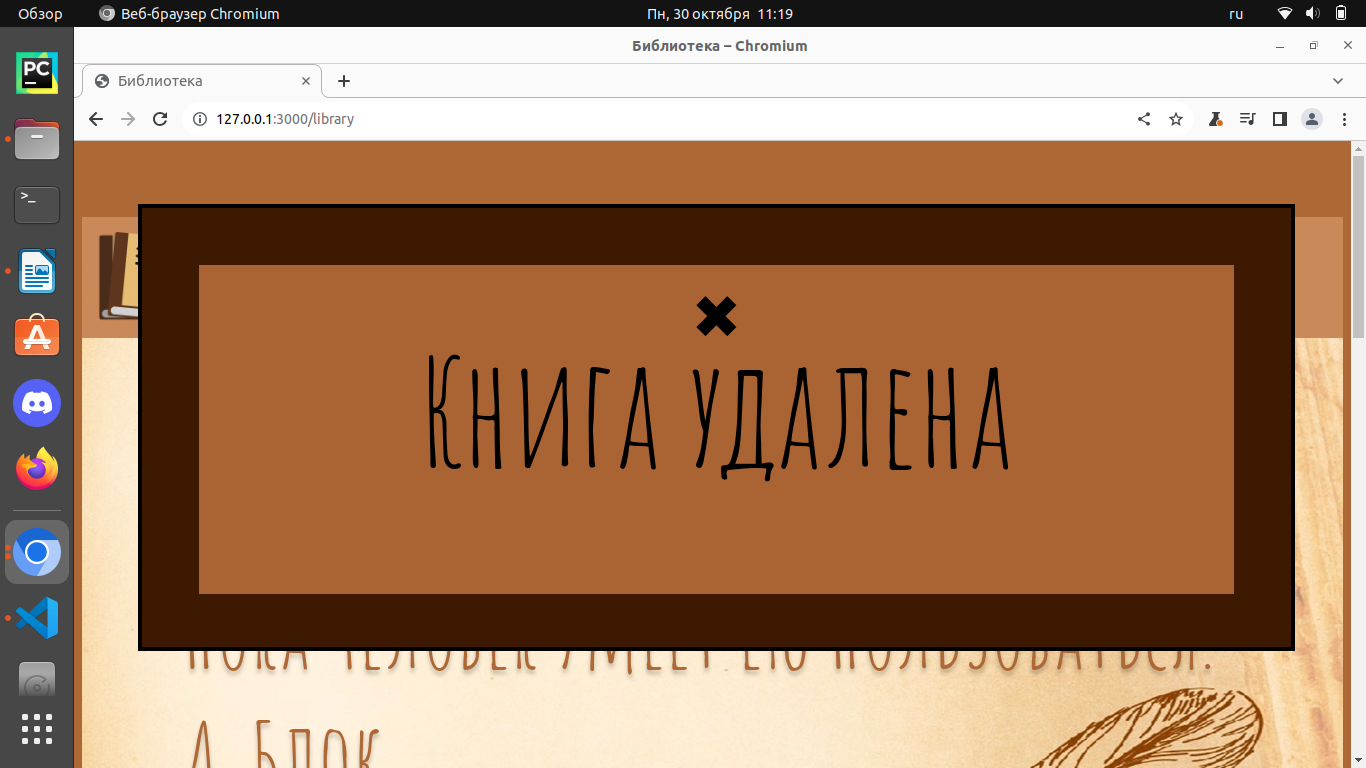


Рис. 5 — книга удалена

4. Карточка книги (card.pug).

На странице книги, на которую пользователь попадает со странице библиотеки при нажатии на обложку любой из книг, расположена «шапка» с кнопкой вернуться назад, обложка и три блока (информация о наличии книги, информация о самой книги и аннотация). В нижнем блоке расположено три кнопки — вернуть книгу, редактировать и взять книгу. Кнопки взять/вернуть активны/неактивны в зависимости от состояния книги (см рис 7).

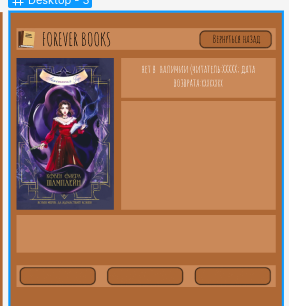
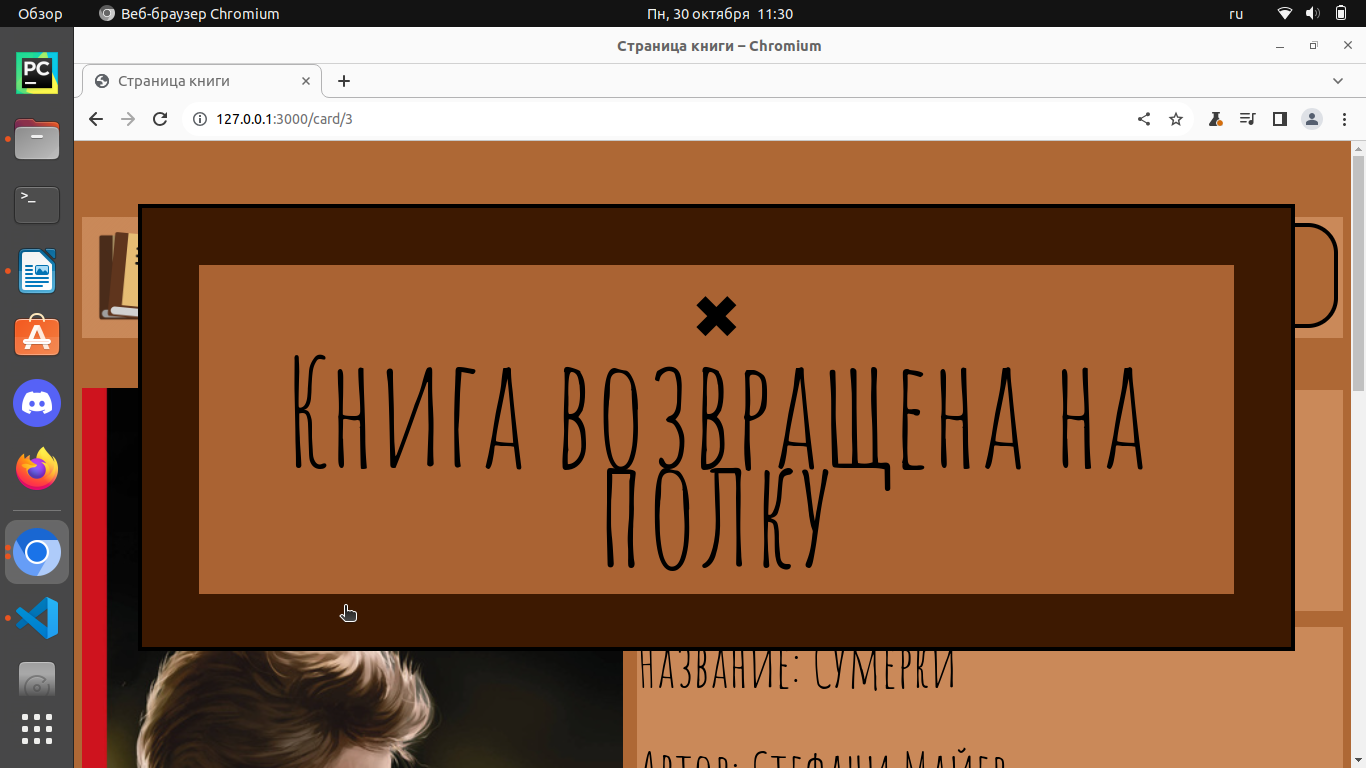


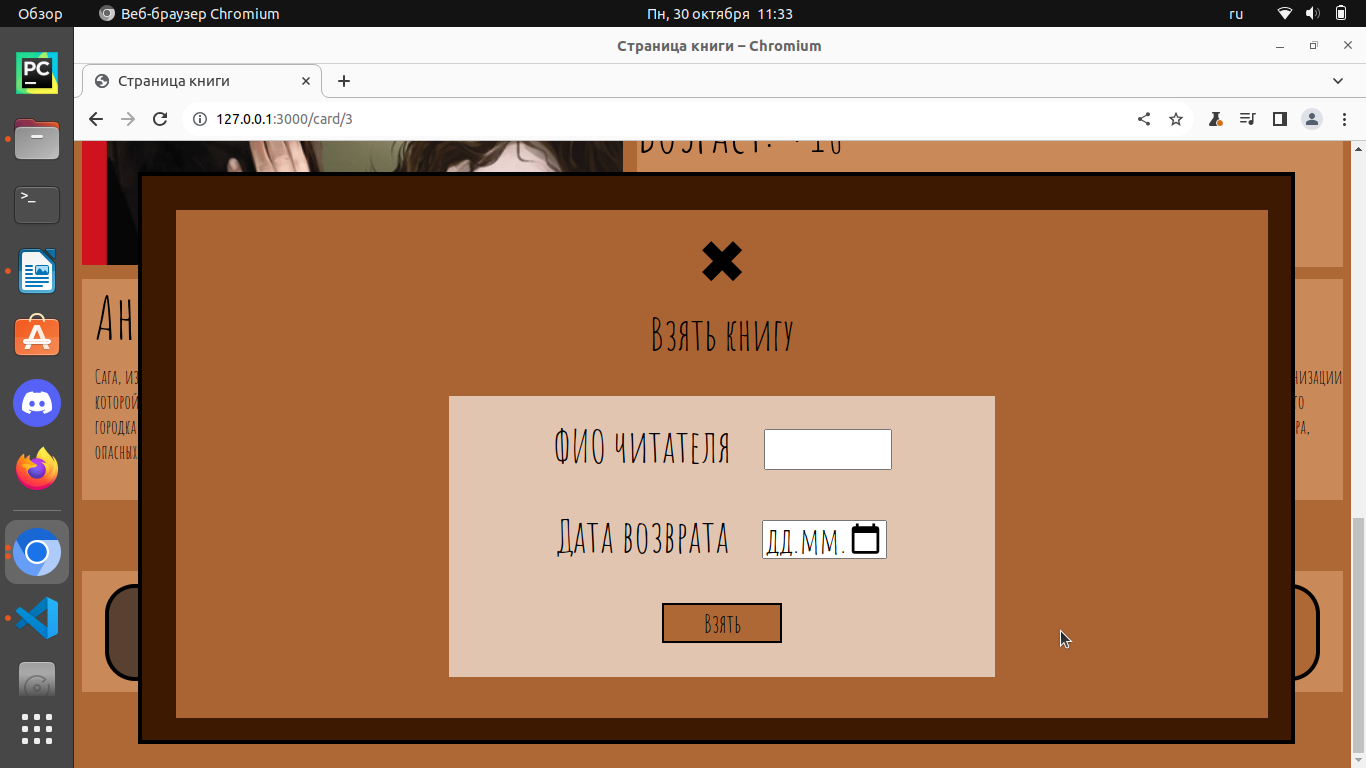
Рис. 6 — карточка книги

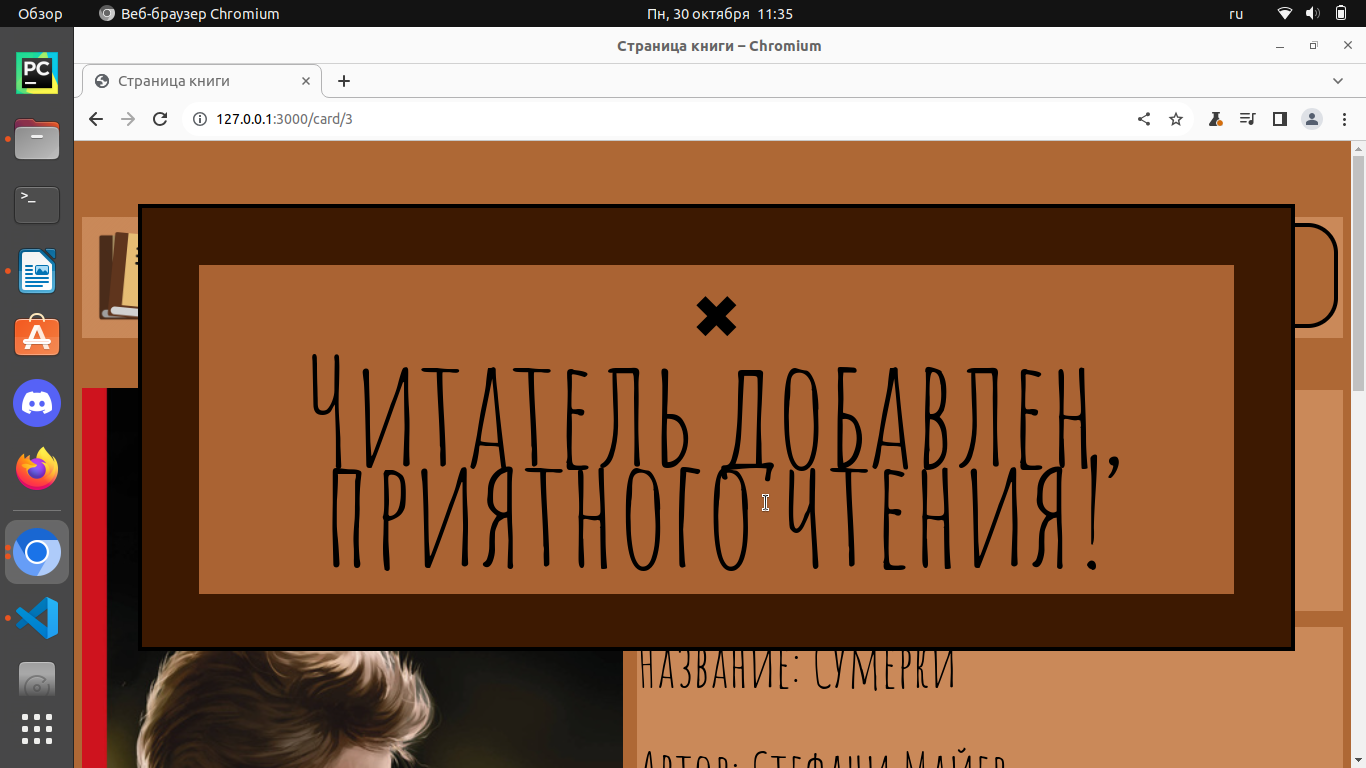
Рис. 7 — активность кнопок Вернуть/взять книгу

## При нажатии на кнопку редактировать, пользователя перенаправляет на страницу редактирования книги. При нажатии на вернуть книгу выскакивает окно оповещения:

Рис. 8 — Книга возвращена на полку

А при нажатии на кнопку взять книгу сначала выскакивает окно добавления пользователя и даты возврата, а потом окно оповещения.

Рис. 9 — Взять книгу

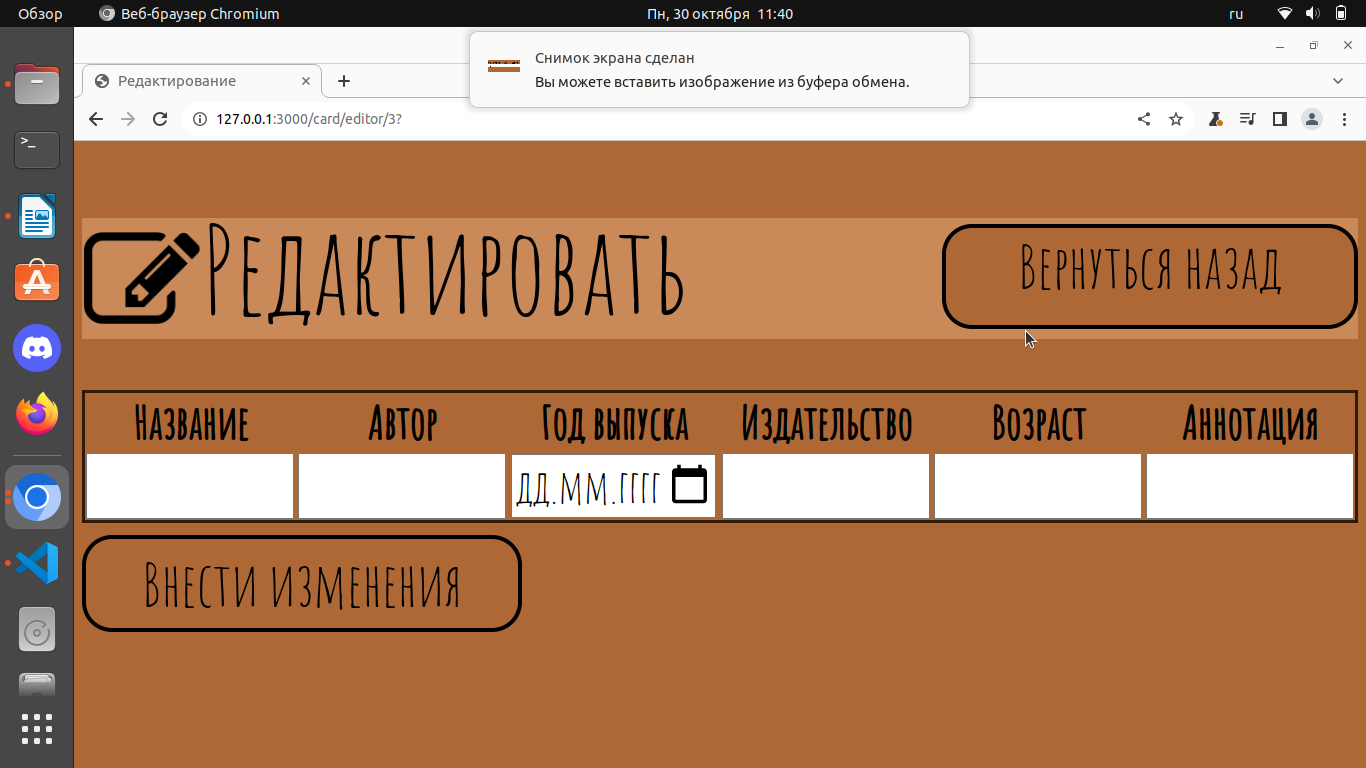


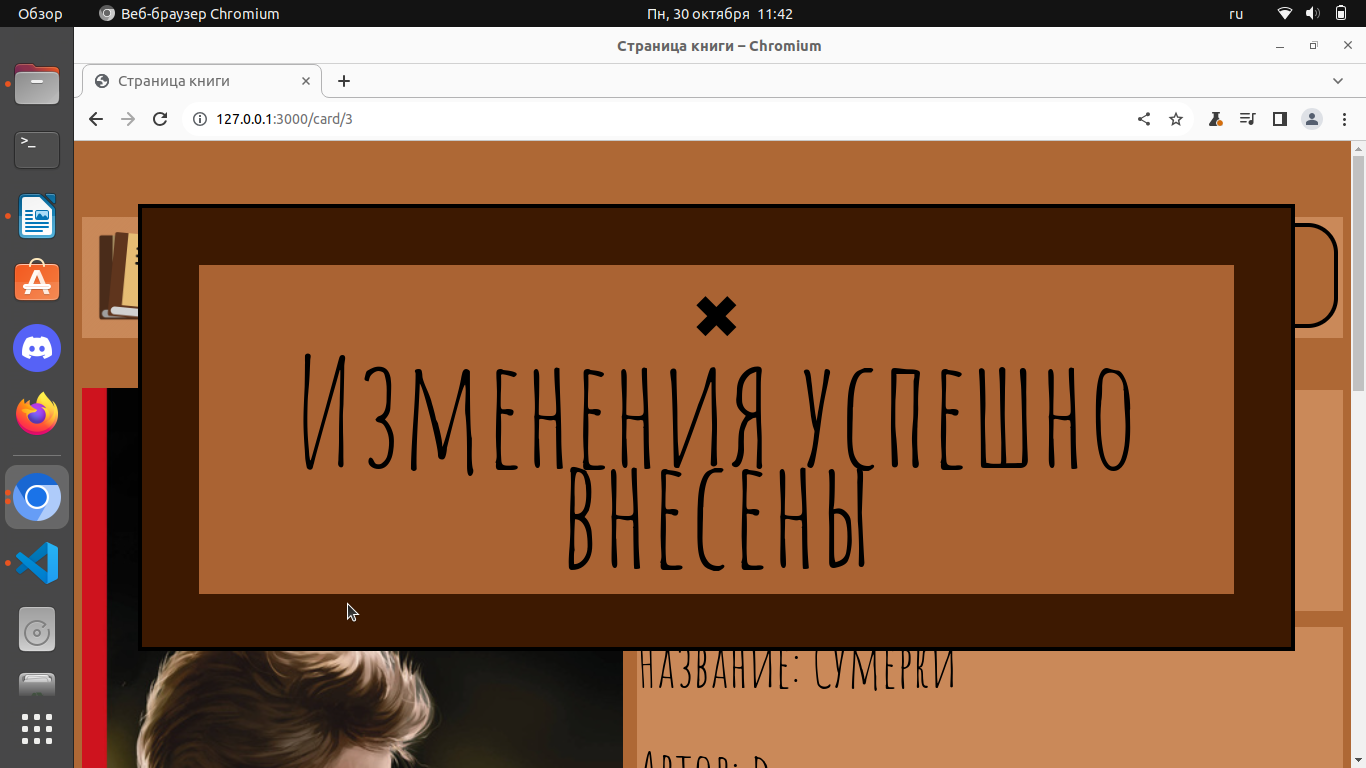
## Рис. 10 — Читатель добавлен, приятного чтения!

Рис. 11 — блок информации о наличии книги

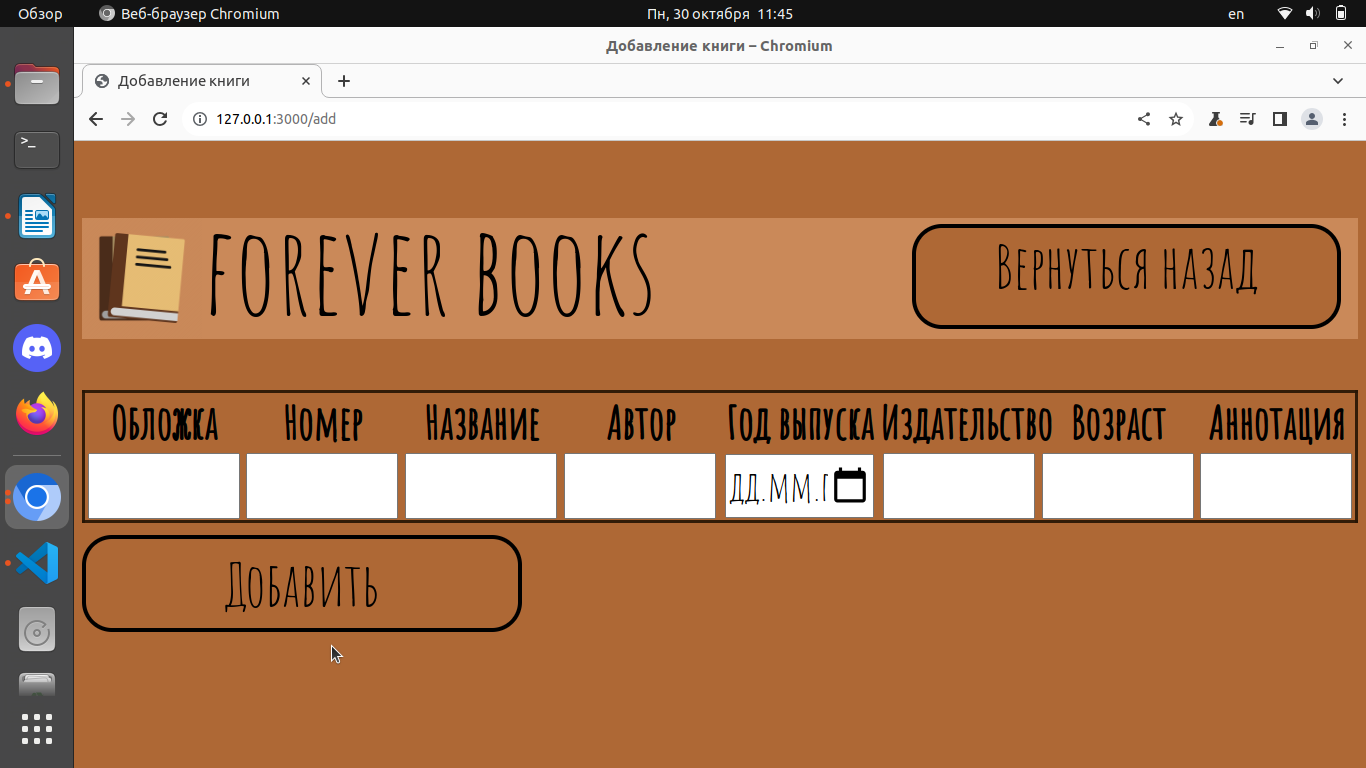
5. Страница редактирования (editor.pug).

Страница редактирования содержит таблицу для редактирования данных, кнопки вернуться назад и внести изменения. После внесения изменений выскакивает окно оповещений на странице редактируемой книги, на которую пользователя затем перенаправляют.

Рис. 12 — страница редактирования книги.

Рис. 13 — изменения успешно внесены

6. Страница добавления книги (add.pug).

Рис. 14 - Страница добавления книги

7. Стилизация сайта.

Всё оформление страниц прописано в файле style.css. Также в проекте имеется файл шрифта AmaticSC-Regular.woff.

8. Файл router.js и app.js.

В дпервом файле прописаны маршруты, а во втором запуск сервера.

## Вывод.

В ходе лабораторной работы были изучены взаимодействия клиентского приложения с серверной частью, освоены шаблоны web-страниц, сформированы навыки разработки динамических HTML-страниц, освоены принципы построения приложений с насыщенным интерфейсом пользователя.