**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе № 2**

**по дисциплине «WEB-технологии»**

**Тема: REST-приложение управления библиотекой**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Григорьев И.С. |
| Преподаватель |  | Беляев С.А. |

Санкт-Петербург

2018

# **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.

# **Постановка задачи**

Для достижения поставленной цели требуется решить следующие задачи:

1) Разработка интерфейса web-приложения.

2) Задание стилей для отображения web-приложения.

3) Создание web-сервера на основе express.

4) Создание шаблонов web-страниц.

5) Настройка маршрутов.

6) Создание json-хранилища.

7) Обработка REST-запросов.

Необходимо создать web-приложение – управления домашней библиотекой, которая предоставляет список книг, их можно отфильтровать по признакам «в наличии», «возврат просрочен», есть возможность выдать книгу для чтения и вернуть книгу. Основные требования:

# **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 – модуль, позволяющий использовать шаблоны для HTML-страниц. REST (Representational State Transfer – передача состояния представления) – стиль взаимодействия компонентов распределённого 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.

# **Ход работы**

1. Начальное состояние библиотеки хранится в JSON-файле на сервере. Текущее состояние – в переменной в памяти сервера. Переменная в памяти сервера представляет собой массив объектов (книг), в каждом объекте хранится необходимая информация о книге, представленная в формате ключ-значение. Все данные загружаются в массив на сервере из json файла, формат которого представлен на рис. 1.

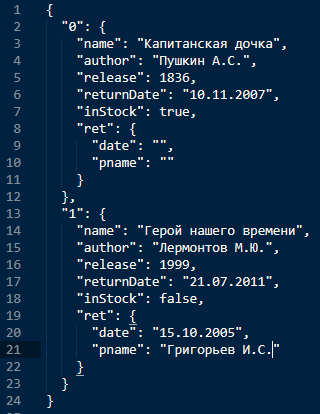


Рисунок 1 – Пример хранения книг в json

1. В качестве сервера используется Node.JS с модулем express.
2. В качестве модуля управления шаблонами HTML-страниц используется pug.
3. Предусмотрена страница для списка книг, страница продемонстрирована на рис. 2. В списке предусмотрена фильтрация по дате возврата и признаку «в наличии», предусмотрена возможность добавления и удаления книг. Предусмотрена страница для карточки книги, в которой её можно отредактировать (минимум: автор, название, дата выпуска) и дать читателю или вернуть в библиотеку. В карточке книги должно быть очевидно: находится ли книга в библиотеке или кто её взял (имя) и когда должен вернуть (дата). Тестирование вышеперечисленных возможностей показано на рис. 2-20.



Рисунок 2 – Страница со списком книг

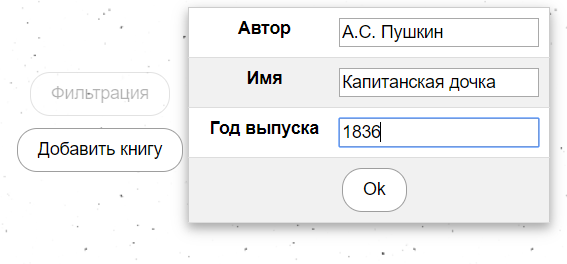


Рисунок 3 – Форма ввода данных для добавления книги



Рисунок 4 – Успешное добавление книги

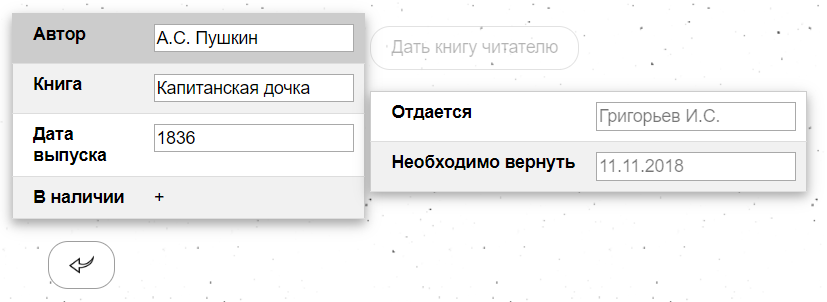


Рисунок 5 – Страница с книгой

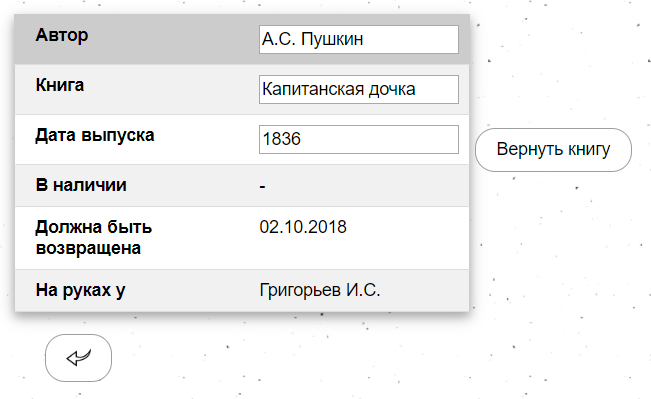


Рисунок 6 – Выдача книги на руки

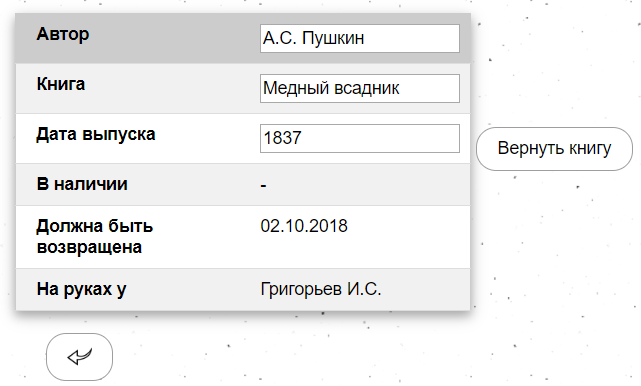


Рисунок 7 – Изменение полей названия и даты



Рисунок 8 – Корректное отображение на главной странице



Рисунок 9 – Книга №11 возвращена



Рисунок 10 – Выдача некоторых книг для проверки фильтра

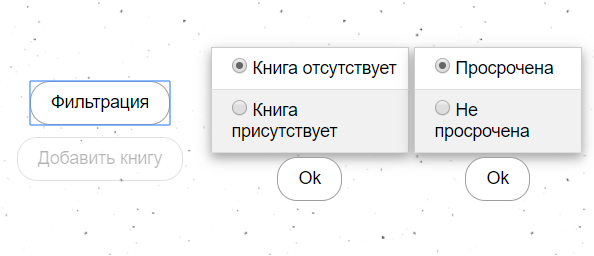


Рисунок 11 – Поля фильтрации

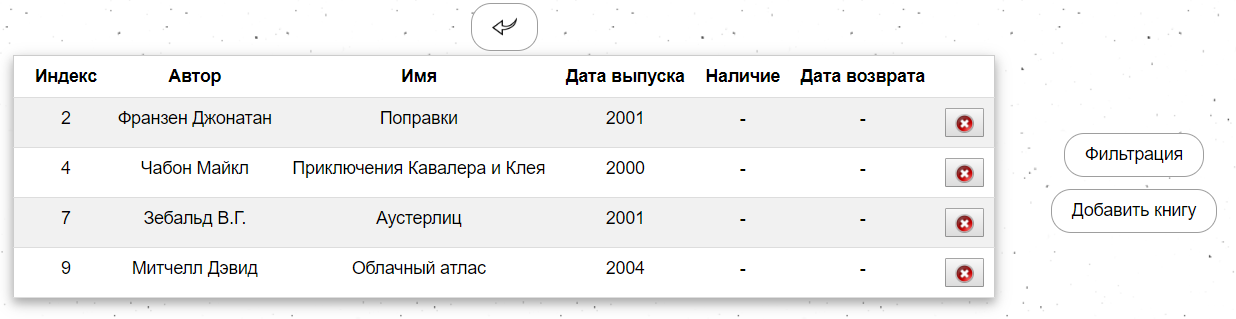


Рисунок 12 – Фильтрация по отсутствию

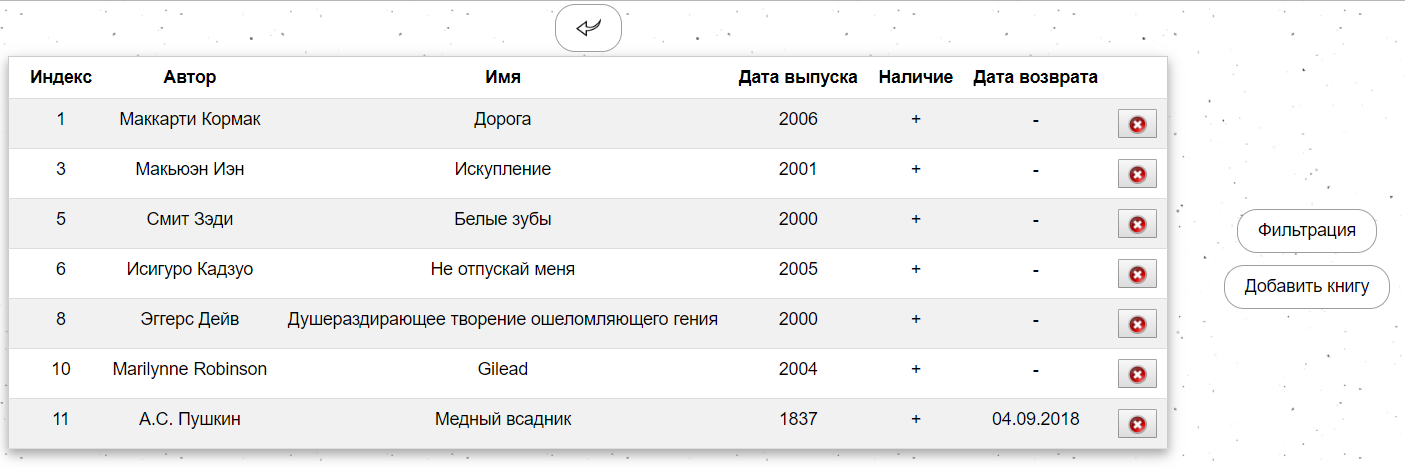


Рисунок 13 – Фильтрация по присутствию

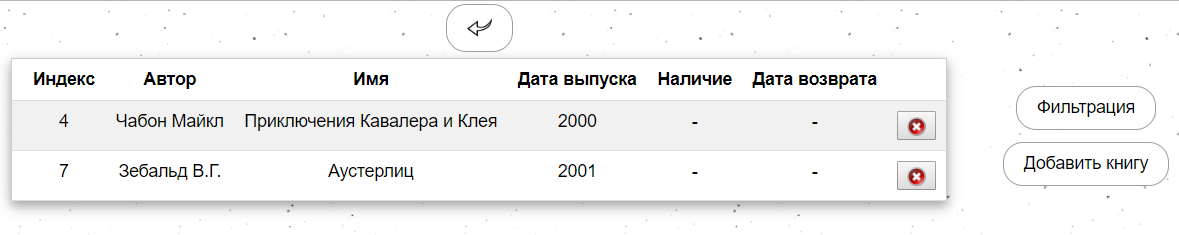


Рисунок 14 – Фильтрация по истечению даты возврата

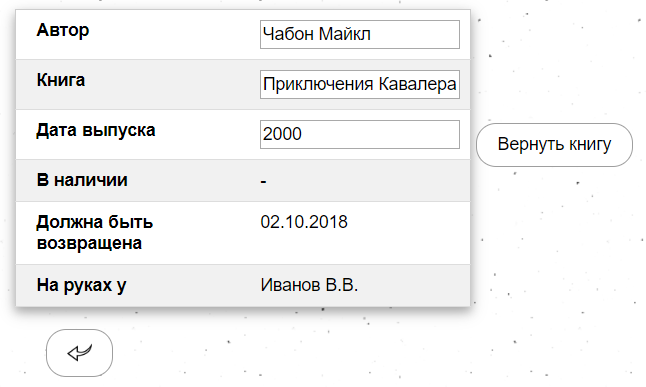


Рисунок 15 – Книга №1 из списка на рис. 14

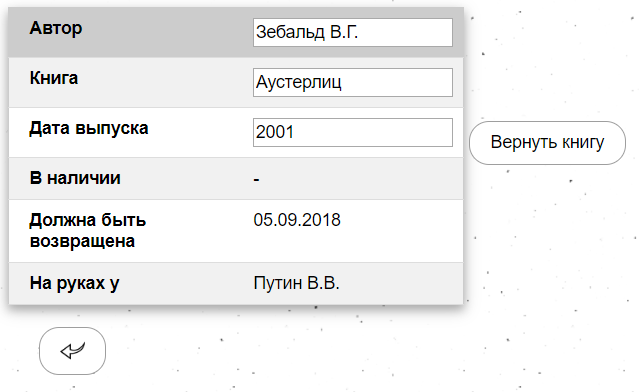


Рисунок 16 – Книга №2 из списка на рис. 14

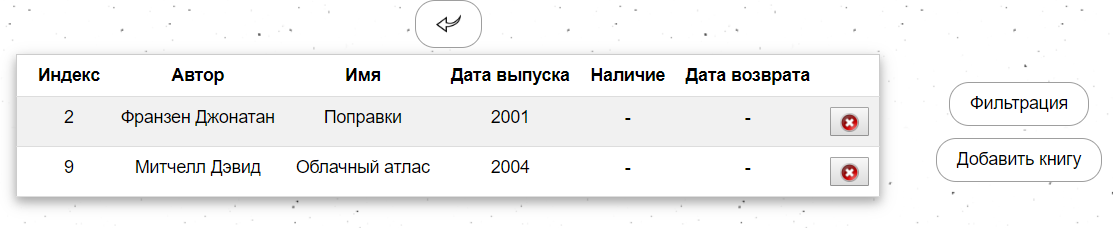


Рисунок 17 – Фильтрация по не истечению даты возврата

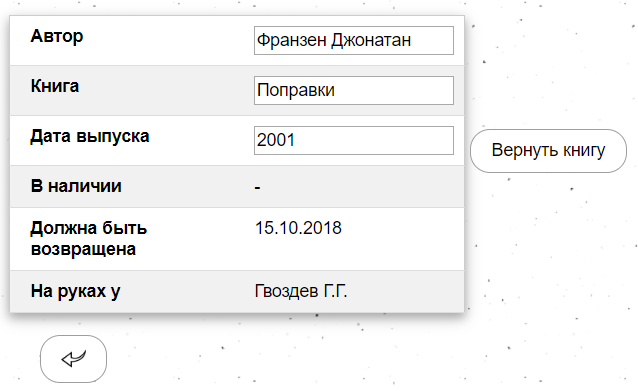


Рисунок 18 – Книга №1 из списка на рис. 17

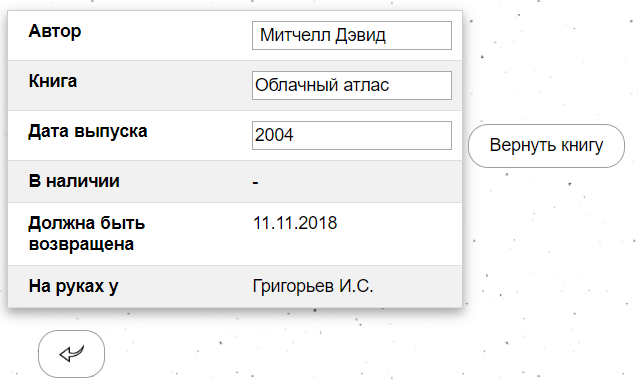


Рисунок 19 – Книга №2 из списка на рис. 17



Рисунок 20 – Текущая дата

1. Оформление страниц выполнено с использованием чистого CSS и w3.css.
2. Взаимодействие между браузером и web-сервером осуществляется с использованием REST.
3. Весь функционал приложения реализован с использованием AJAX-запросов.
4. Логика приложения реализована на языке JavaScript.

# **Вывод**

В ходе лабораторной работы изучены работа сервера Node.JS с модулем express, динамическое создание html страниц посредством использования шаблонизатора pug, также для хорошего внешнего вида элементов страницы был применен как чистый css, так и w3.css. Весь функционал был реализован с использованием асинхронных AJAX-запросов. Более того реализован сервер и клиент, взаимоотношение между которыми осуществляется с помощью REST. Таким образом полученные теоретические знания закреплены на практике.