**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работe №2**

**по дисциплине «Web - технологии»**

Тема: «REST-приложение управления библиотекой»

|  |  |  |
| --- | --- | --- |
| Студент гр. 0382 |  | Литягин С.М. |
| Преподаватель |  | Беляев С.А. |

Санкт-Петербург

2022

**Цель 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.

**Задачи**

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.

**Описание решения**

Сначала было инициализировано приложение командой *npm init*. Затем была произведена установка командой npm install –save <packet> следующих пакетов: cookie-parser, express, morgan, pug, pug-cli.

Создано json-хранилище для книг библиотеки. Элемент списка книг в хранилище выглядит следующим образом (см. рис. 1). Примечание: если книга находится в библиотеке, то тело по ключу “personCard” будет пустым.

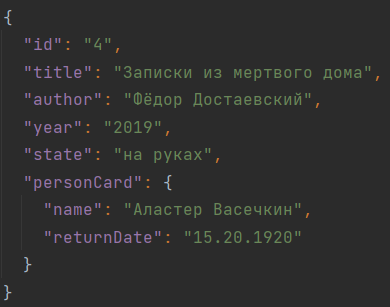


Рисунок 1 – Элемент json-хранилища

Для внесения каких-либо изменений в json-хранилище был написан класс Writer. В нем присутствуют методы:

1. writeToJson(books) - для перезаписи json-хранилища согласно новому списку книг books;
2. filterBooks(filter) – в зависимости от значения filter будет возвращен исходный список книг, или список книг в наличии, или список книг на руках;
3. deleteBook(id) – удаляет книгу из хранилища с указанным id;
4. findBook(id) – находит книгу по указанному id (если id = 0, то вернет заготовку для создания новой книги, если книги нет – вернет ошибку);
5. rewriteBook(book) – перезаписывает переданную книгу book в хранилище;

Затем был написан файл app.js, в котором реализован express-сервер. К данному серверу подключен обработчик шаблонов pug, добавлена статическая папка (корень проекта), был добавлен logger из пакета morgan, добавлена поддержка JSON/URL–закодированных тел запросов, обработка cookies. Также были добавлены два маршрутных модуля: для “/editBook” (из файла editRoutes.js) и “/” (из файла routes.js).

В модуле routes.js обрабатываются маршруты для главной странице библиотеки. Были написаны следующие запросы для “/”: метод GET – возвращает HTML-страницу booksList, метод PUT – выполняет фильтрацию (используется функция класса Writer) согласно указанному в теле запроса параметру state и возвращает HTML-страницу booksList с отфильтрованным набором книг, метод DELETE – удаляет книгу (используется функция класса Writer) по указанному в теле запроса параметру id и возвращает HTML-страницу booksList.

В модуле editRoutes.js обрабатываются маршруты для страницы редактирования информации о книги. Были написаны следующие запросы для “/:id([0-9]{1,})”: метод GET – находит книгу согласно переданному параметру id, и если книга в хранилище есть – возвращает HTML-страницу editBook, метод PUT – перезаписывает или добавляет книгу в хранилище и возвращает HTML-страницу editBook.

В случае если ни один маршрут не удовлетворил запросу, то возвращается HTML-страница error.

Все страницы (booksList, editBook и error) были написаны с применением пакета pug (шаблонные web-страницы). Их вид продемонстрирован на рисунках 2-6.

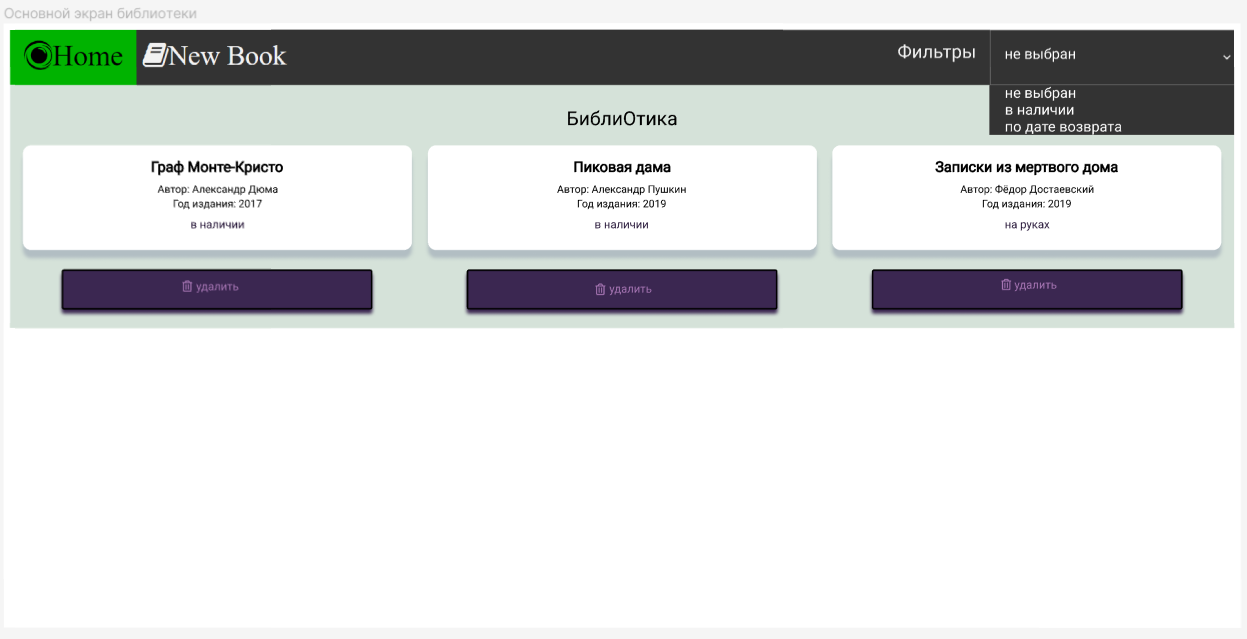


Рисунок 2 – Главная страница booksList

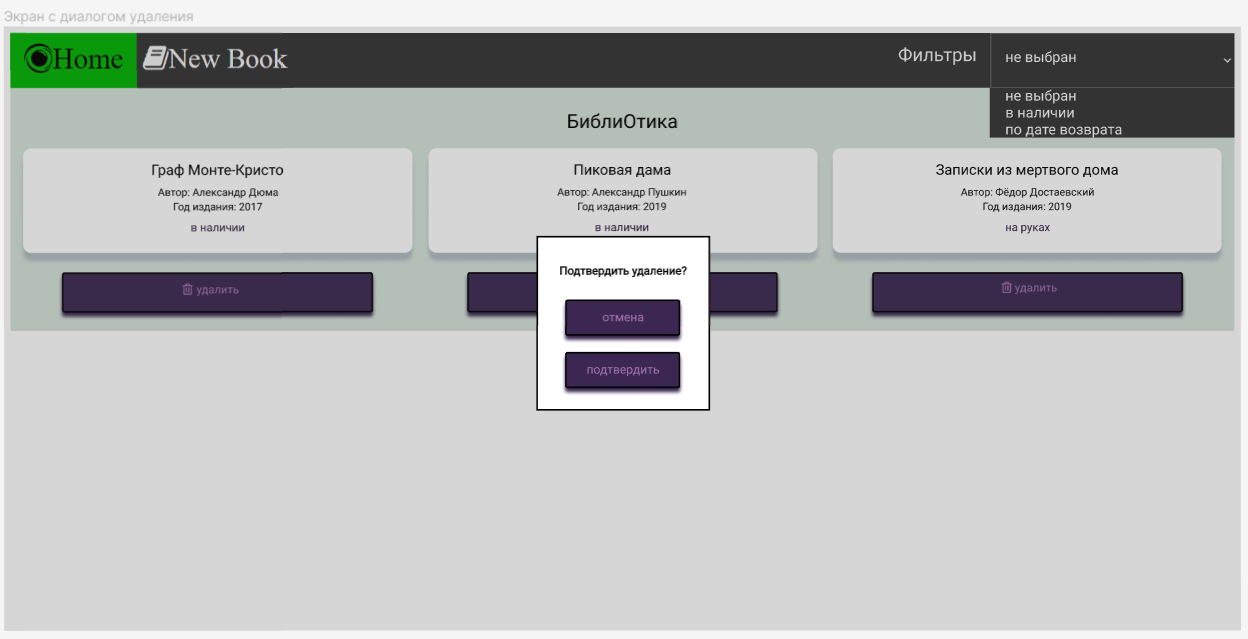


Рисунок 3 – Диалог удаления на странице booksList

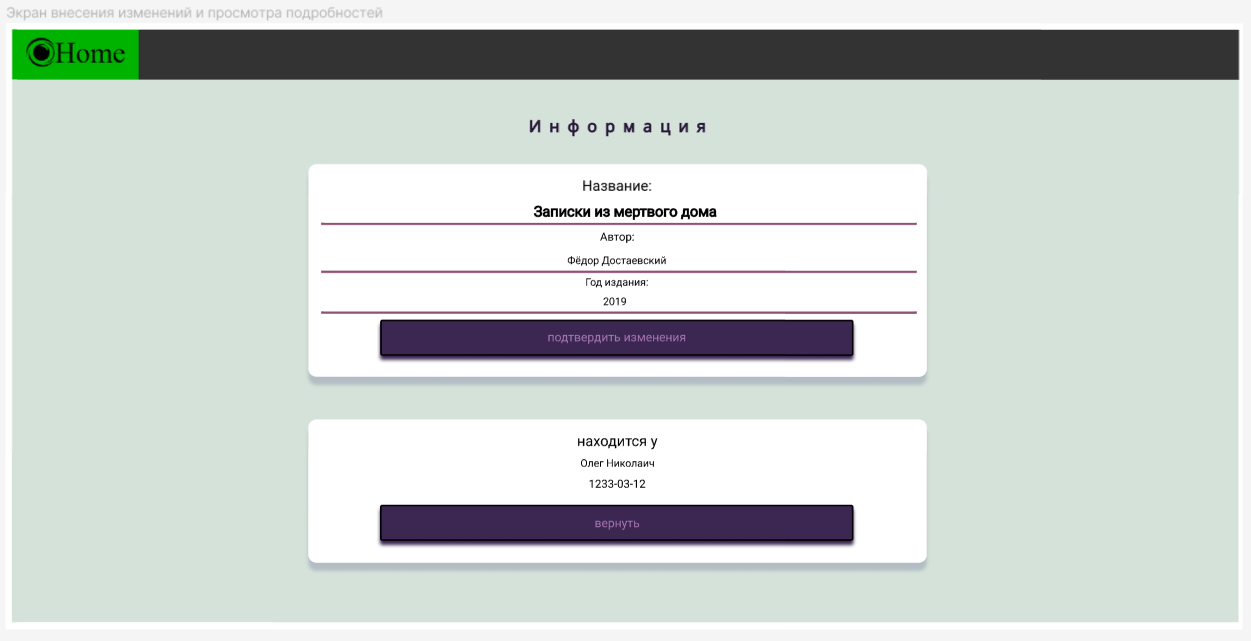


Рисунок 4 – Страница editBook

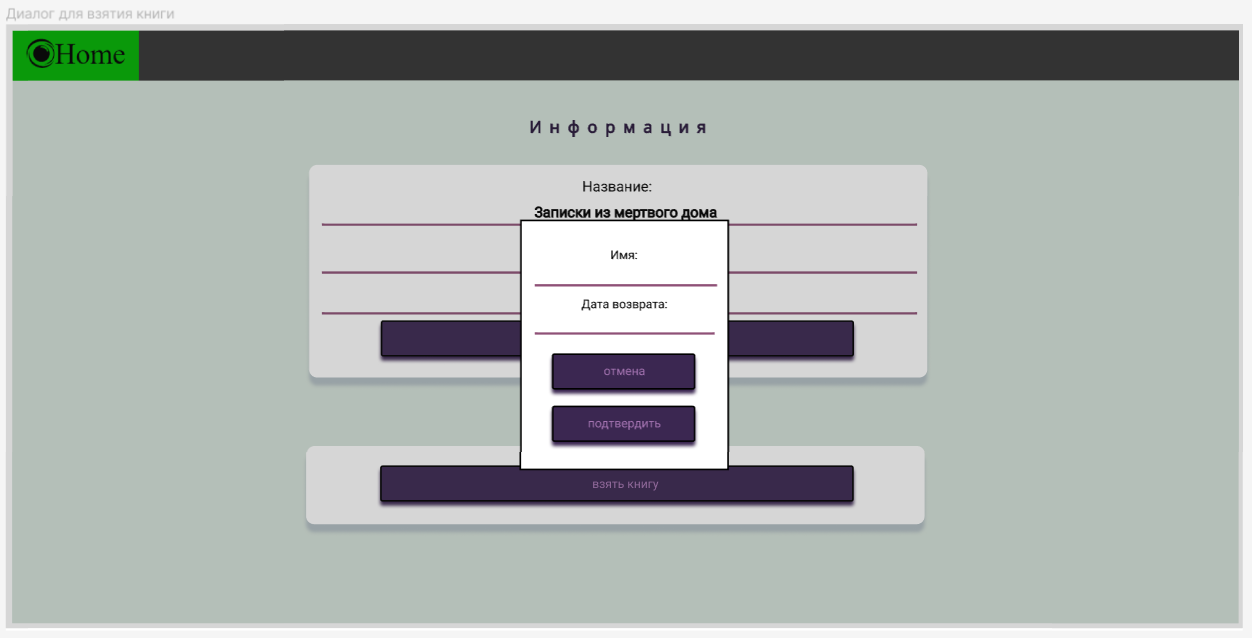


Рисунок 5 – Диалог для взятия книги

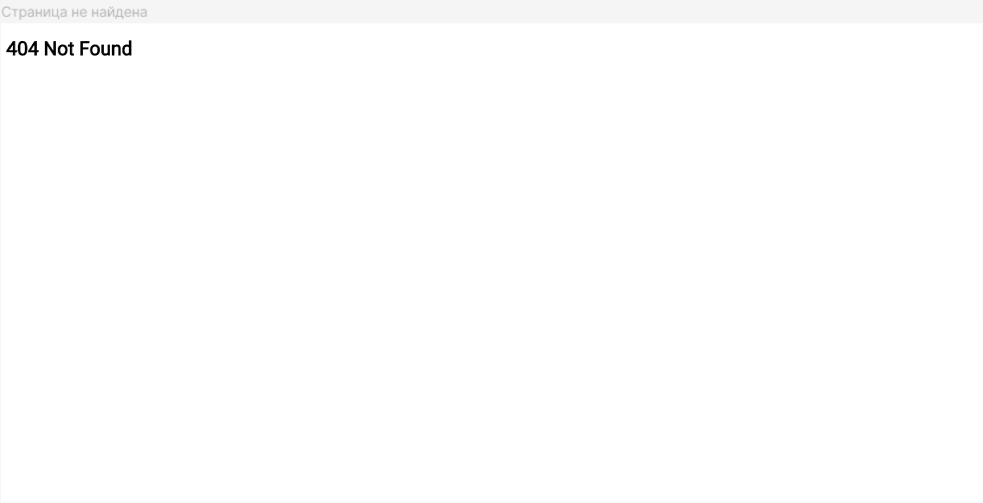


Рисунок 6 – Ошибка при не нахождении страницы

Также дизайн для всего сайта был разработан на figma. Посмотреть можно по [ссылке](https://www.figma.com/file/XMQrCZsstARZtHlNczgywO/Library?node-id=0%3A1). Дизайны представлены на рисунках 2-6.

**Интерфейс**

Основное управление библиотекой:

- Для открытия страницы книги нужно нажать на соответствующую карточку книги;

- Для удаления книги нужно нажать кнопку “удалить” под соответствующей карточкой. Это откроет диалог подтверждения удаления карточки: “отмена” – закрывает диалог без удаления книги, “подтвердить” – удаляет книгу;

- В верхнем тулбаре присутствует список фильтров. При нажатии на каждый из них будет вызвана фильтрация списка книг, что отобразится соответствующим отображением карточек книг на странице;

- Кнопка “Home” в верхнем тулбаре вернет на главную страницу библиотеки (т.е. со списком книг);

- Кнопка “New Book” откроет страницу для создания новой книги;

- При создании новой книги на соответствующей странице все поля должны быть заполнены (книгу не могут взять, пока она не создана);

- На странице книги можно отредактировать информацию о ней (для сохранения изменений нужно нажать кнопку “подтвердить изменения”);

- На странице книги можно взять книгу. Для этого нужно нажать кнопку взять, что откроет диалог для ввода данных читателя. Для сохранения данных нужно нажать кнопку “подтвердить” (все поля диалога должны быть заполнены, а также книга должна быть в библиотеке), так же можно отменить взятие нажатием кнопки “отмена”.

- На странице книги можно вернуть книгу. Для этого нужно нажать кнопку “вернуть”.

**Скриншоты программы**

Скриншоты программы представлены на рисунках 7-13.

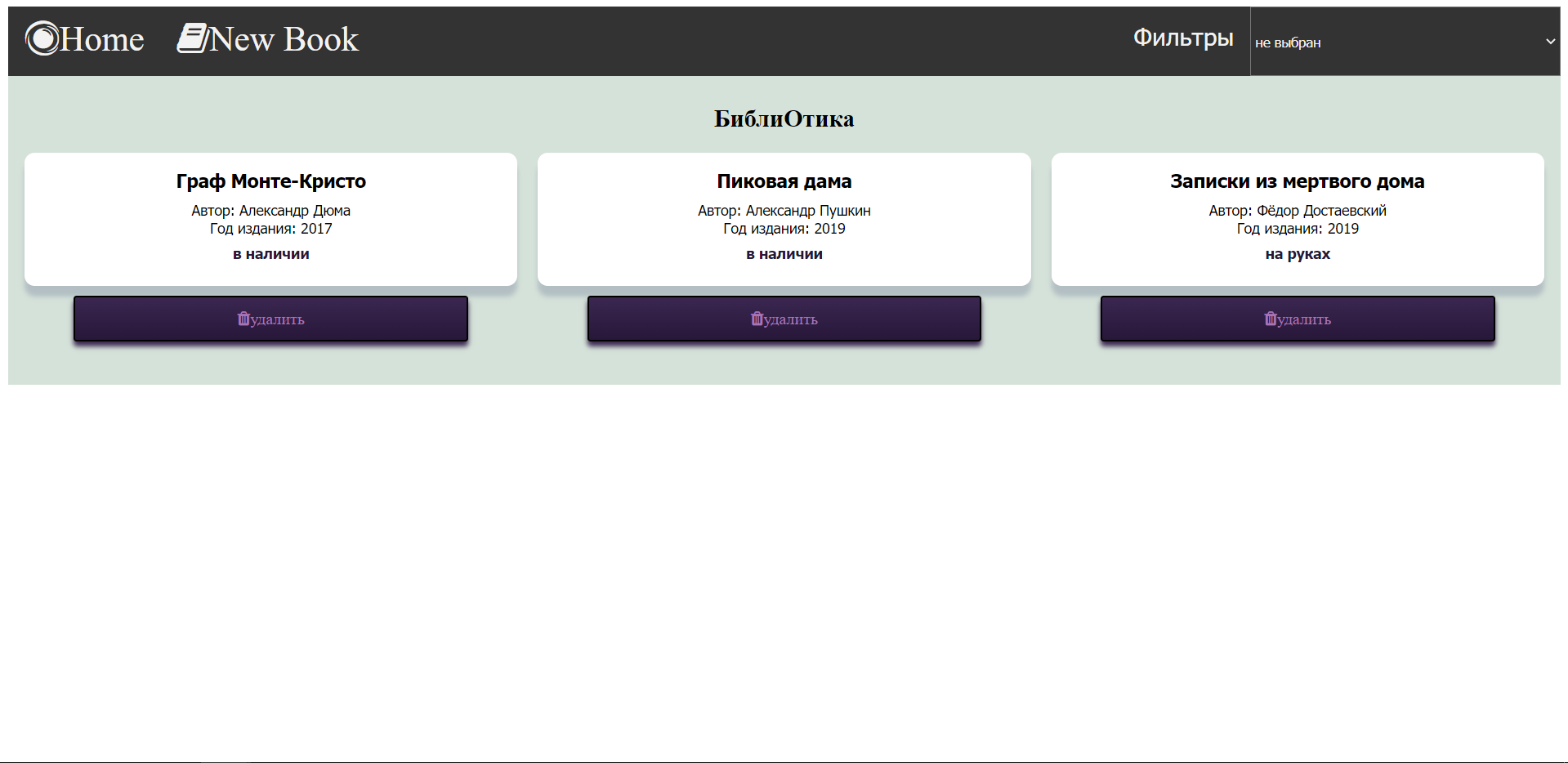


Рисунок 7 – Экран со списком книг

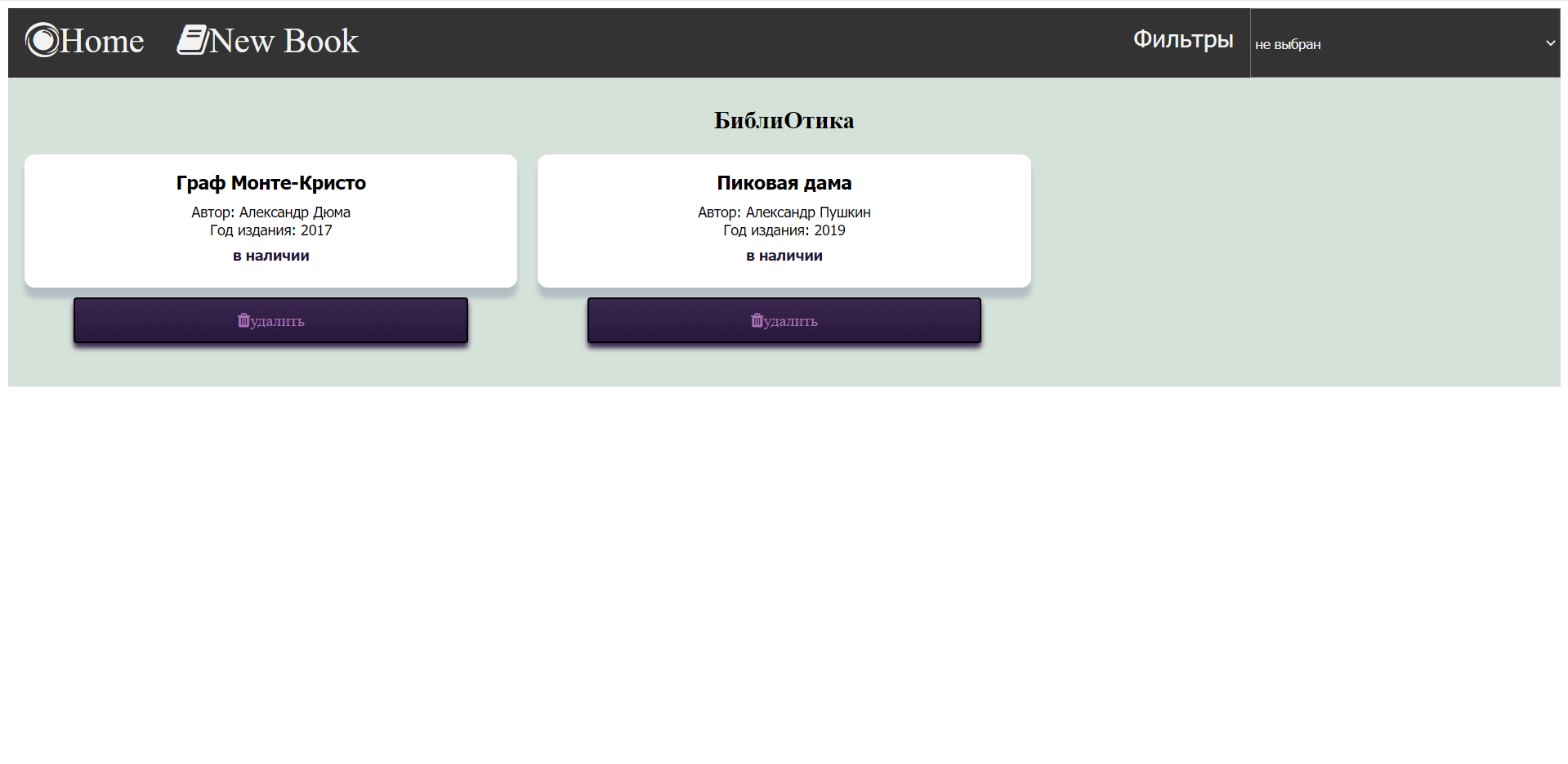


Рисунок 8 – Отфильтрованный по “в наличии”

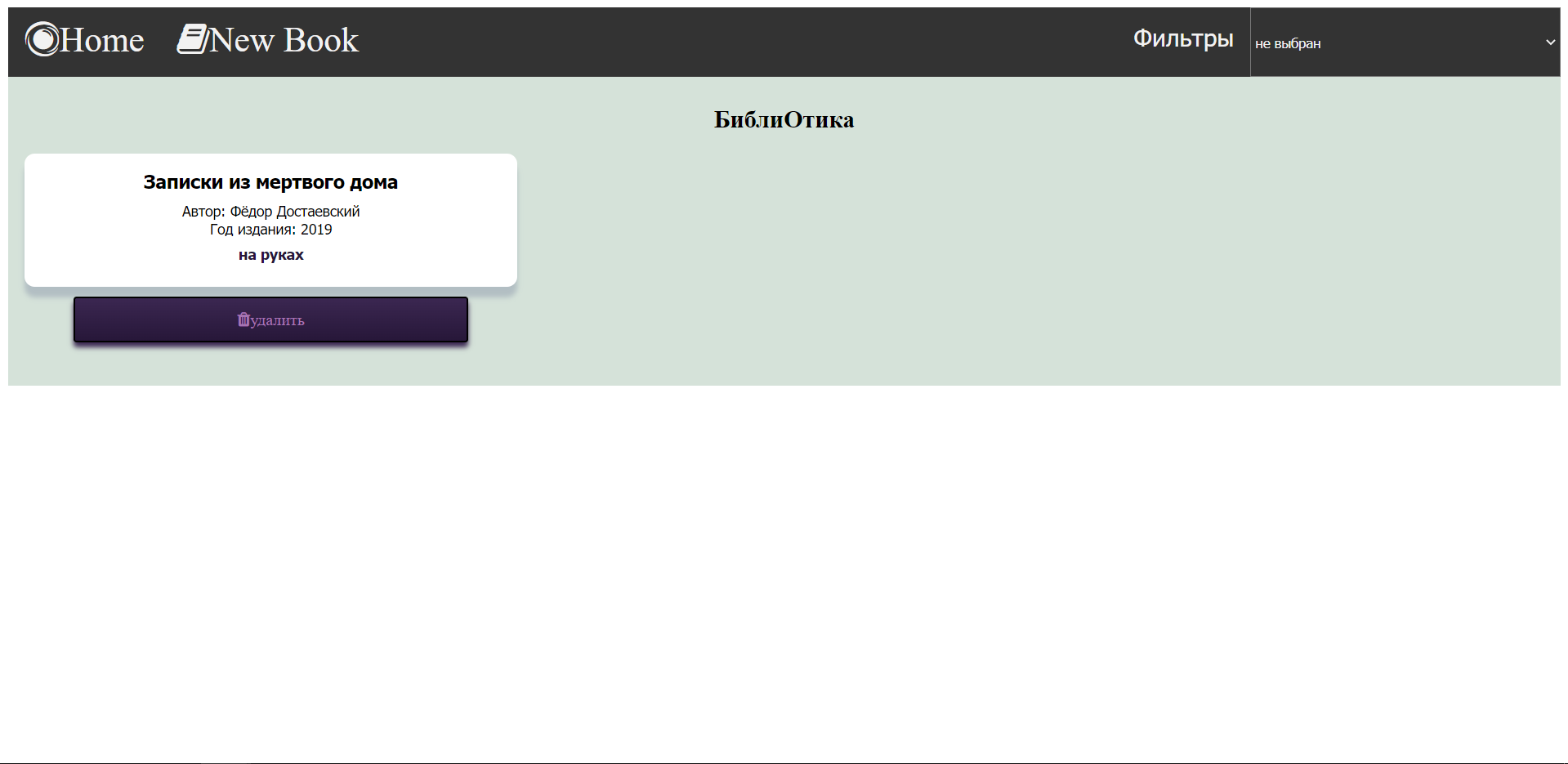


Рисунок 9 – Отфильтрованный по “по дате возврата”

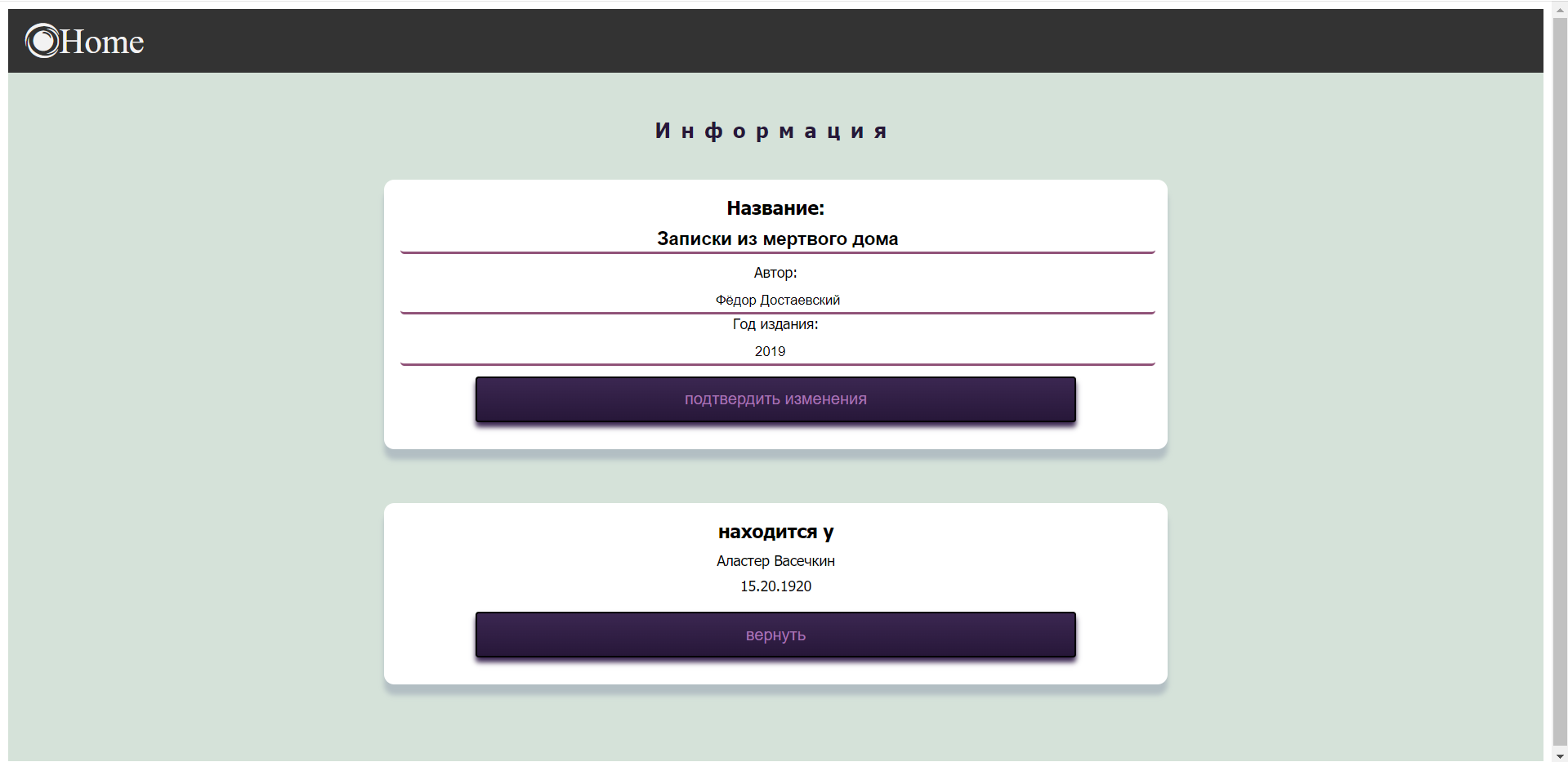


Рисунок 10 – Экран editBook после нажатия соответствующей карточки

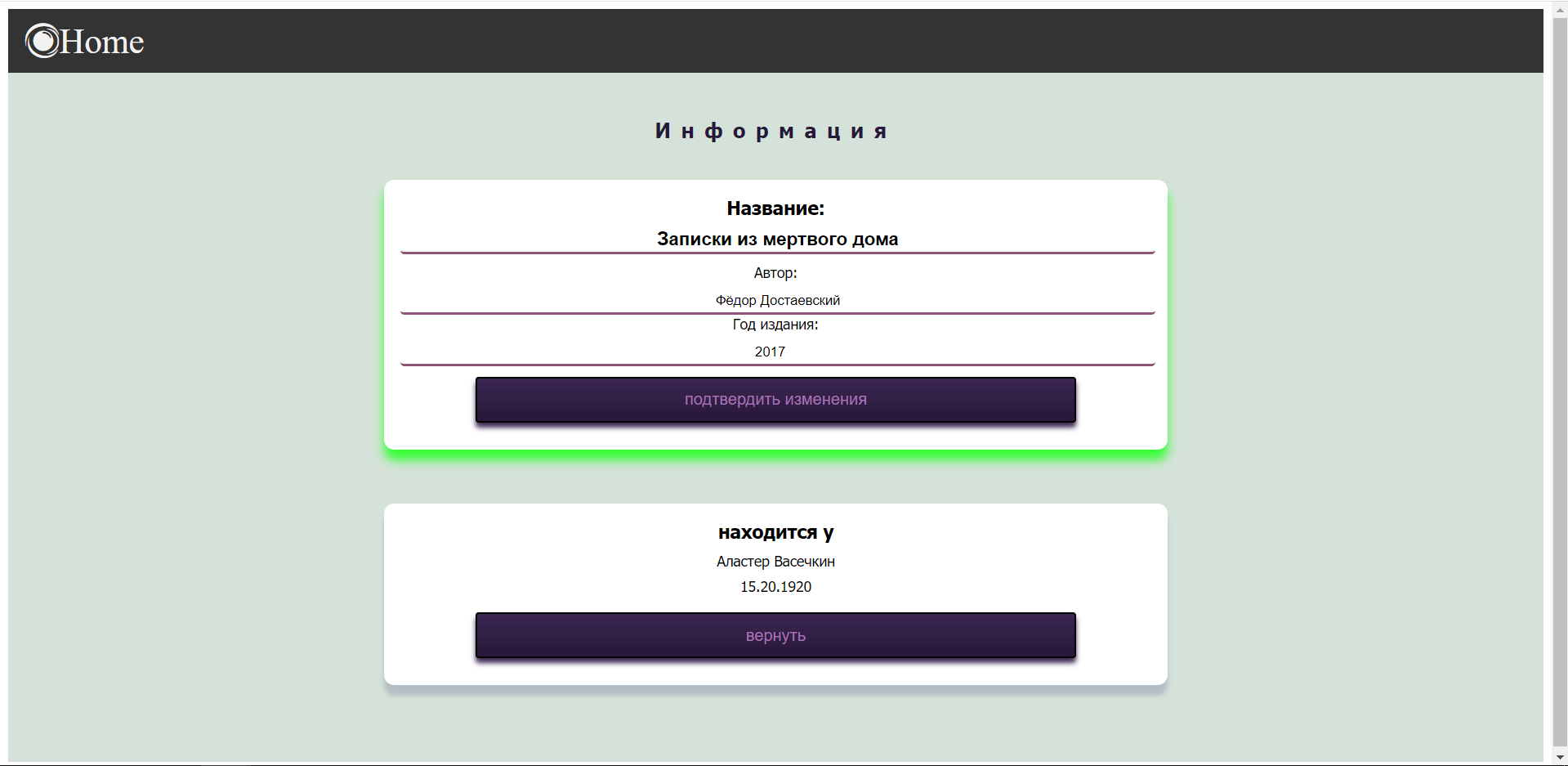


Рисунок 11 – Подсветка после удачного внесения изменения

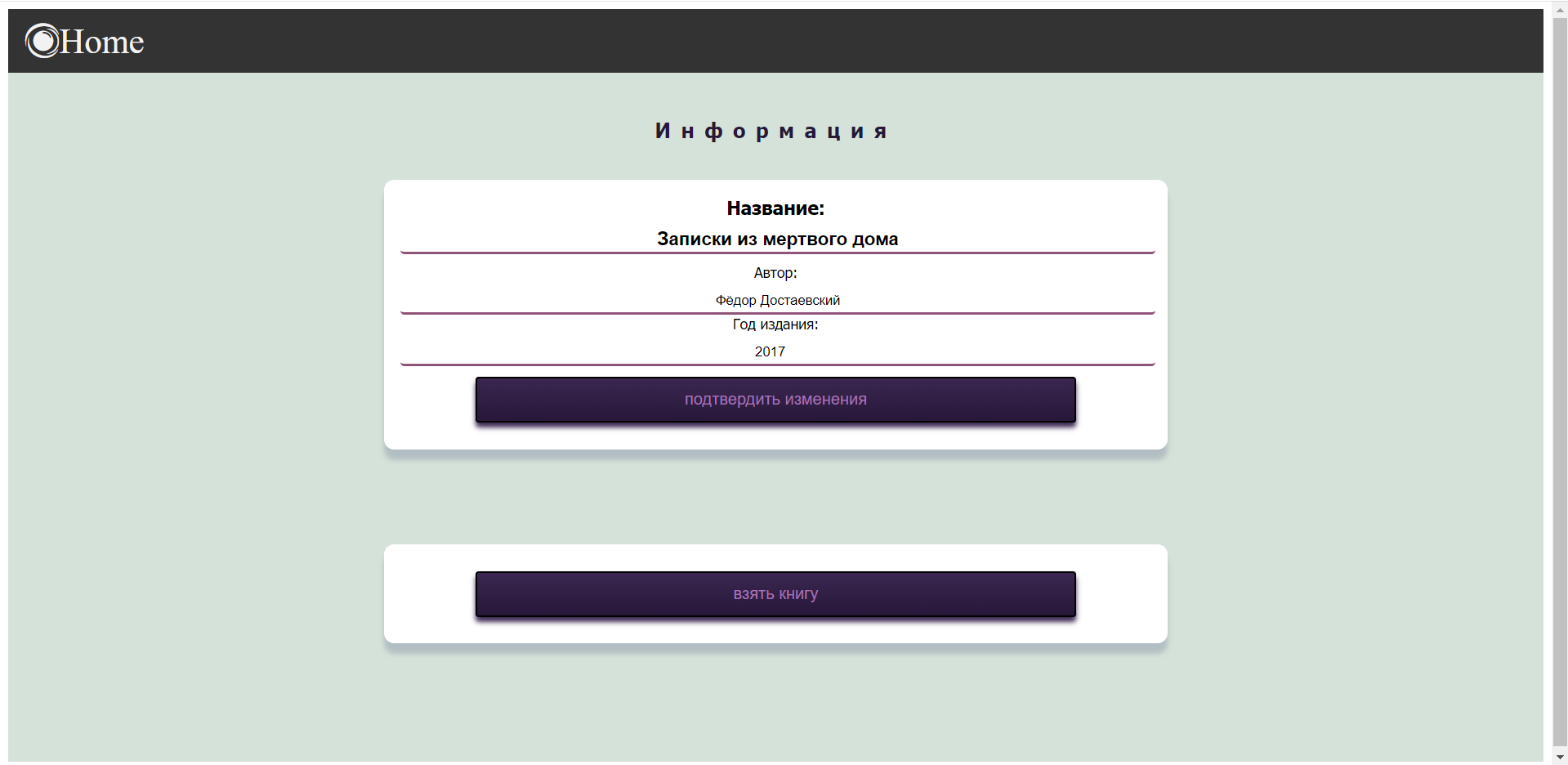


Рисунок 12 – Экран editBook после нажатия “возврат”

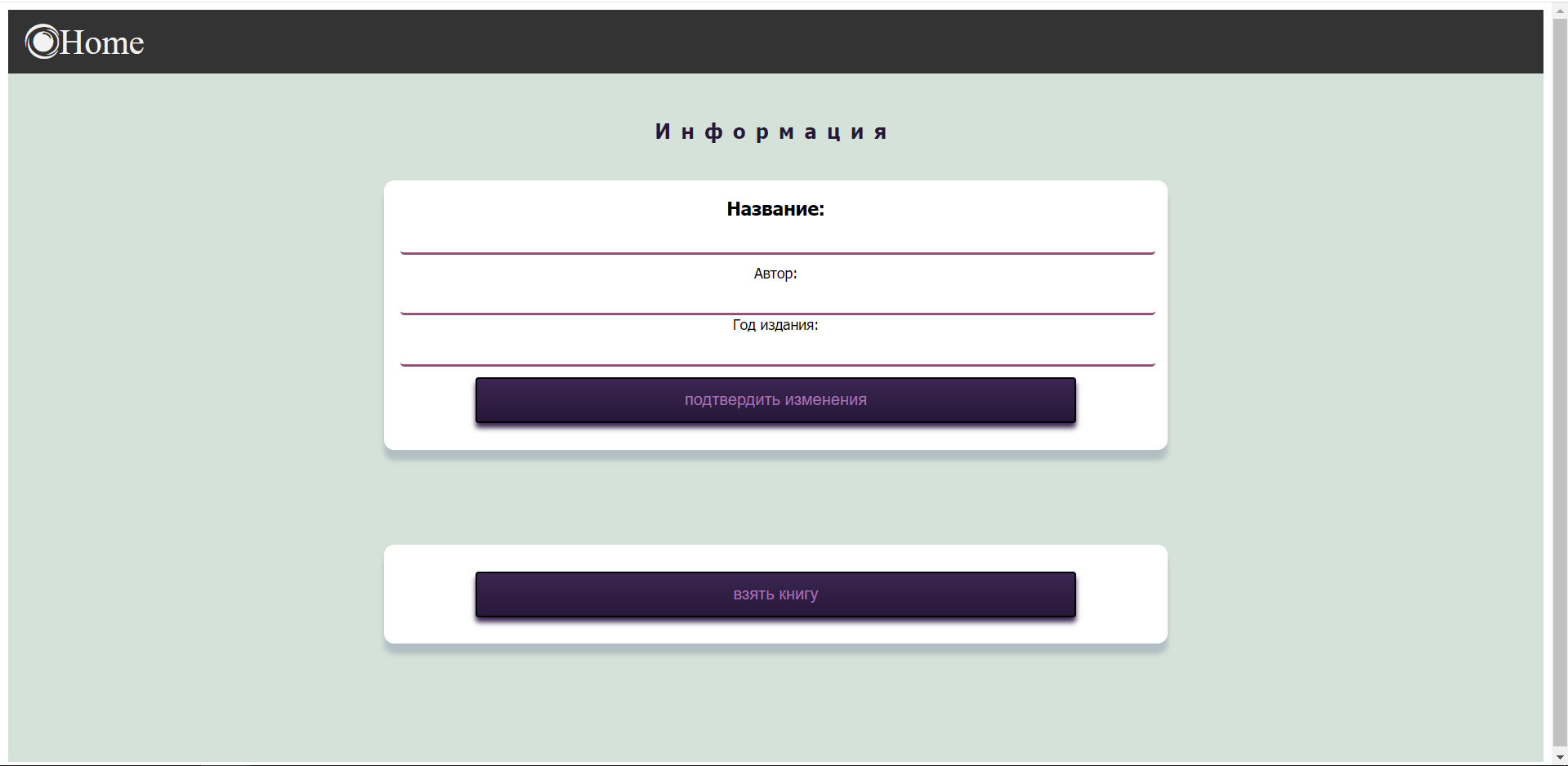


Рисунок 13 – Экран editBook для создания новой книги

**Выводы**

В ходе работы было изучено взаимодействия клиентского приложения с серверной частью, были освоены шаблоны web-страниц, сформирован навык разработки динамических HTML-страниц, освоен принцип построения приложений с насыщенным интерфейсом пользователя.