МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Информационные системы»

Дисциплина «Интернет-программирование»

**Лабораторная работа № 3**

Верстка сайта с использованием Bootstrap, простая логика на JS

Выполнил:

студент гр. ПИбд-23

Пазушкин И.П.

Проверил:

преподаватель

Истюков Т.А.

Ульяновск,   
2025 г.

**Задание лабораторной работы**

1. Необходимо сформировать окружение проекта на основе примера. В проекте должны быть использованы: npm, vite, eslint, prettier. Проект должен содержать файл package.json, в котором необходимо объявить основные скрипты для сборки и запуска проекта, а также указать зависимости. Для сборки проекта рекомендуется использовать Vite. Для проверки качества кода рекомендуется использовать ESLint. Для форматирования кода рекомендуется использовать Prettier. Для использования указанных инструментов необходима установка Node.js.

2. По результатам лабораторной работы 2 необходимо переработать все страницы с использованием фреймворка Bootstrap. Все стилевые особенности должны быть сформированы средствами CSS с использованием классов фреймворка Bootstrap. Содержимое собственного CSS-файла должно содержать только необходимый минимум настроек внешнего вида.

3. Необходимо использовать не менее 5 разных иконок из пакета bootstrap-icons при верстке страниц.

4. Макет должен поддерживать адаптивную верстку для мобильных устройств, планшетов и ПК. Макеты для разных типов устройств должны отличаться.

5. Необходимо разработать форму для добавления сведений о сущностях проблемной области на некоторую страницу (карточки с товарами, карточки фильмов, карточки друзей и т. д). При отправке формы должен вызываться обработчик на языке JS, который выполняет функцию создания и настройки атрибутов элемента (карточки) с последующим добавлением в контейнер в качестве потомка.

6. Файлы проекта и отчет должны быть загружены в git-репозиторий на сервере http://student.git.athene.tech/

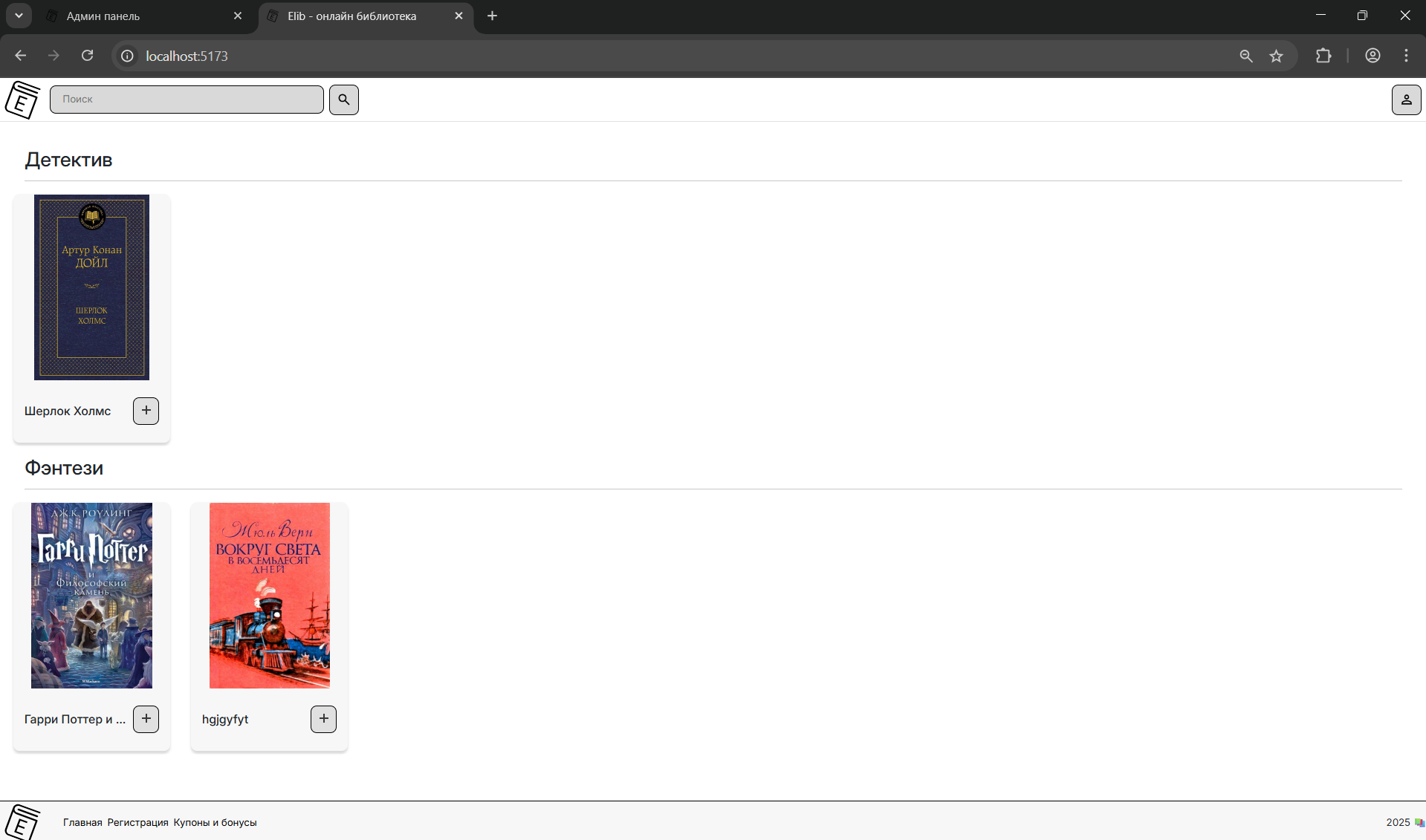
**Порядок выполнения работы**

Началом данной работы было формирование окружения проекта по первому пункту задания. Далее был очищен файл «styles.css» от настроек, которые стали не нужны или реализацию заменил фреймворк «Bootstrap». Все страницы используют иконки из пакета bootstrap-icons. Страницы выполнены с подходом mobile first. По заданию пункта пять я реализовал на странице «admin.html» с предметами библиотеки. Логика работы карточек товаров:

* Добавление товаров: при отправке формы динамически создаётся карточка с изображением, названием.
* Удаление и очистка: каждую карточку можно удалить с помощью кнопки "Удалить", она полностью удаляется и DOM.

**Скриншоты**

*Страница index.html:*

**

*Страница admin.html*

*Изображение выглядит как текст, снимок экрана, программное обеспечение, компьютер

Контент, сгенерированный ИИ, может содержать ошибки.*

*Страница user.html*

*Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.*

*Страница certificates.html*

*Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.*

*Страница login.html*

*Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.*

*Страница register.html*

*Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.*

**Исходный код**

**eslint.config.js**

import perfectionistPlugin from 'eslint-plugin-perfectionist'; import stylisticJsPlugin from '@stylistic/eslint-plugin-js'; import tseslint from '@typescript-eslint/eslint-plugin'; import prettierPlugin from 'eslint-plugin-prettier'; import tsParser from '@typescript-eslint/parser'; import jsPlugin from '@eslint/js'; import globals from 'globals'; export default [ jsPlugin.configs.recommended, perfectionistPlugin.configs['recommended-line-length'], { rules: { 'stylistic/padding-line-between-statements': [ 'error', { blankLine: 'always', prev: 'import', next: '\*' }, { blankLine: 'any', prev: 'import', next: 'import' }, { blankLine: 'always', next: 'return', prev: '\*' }, { blankLine: 'always', prev: 'block-like', next: '\*' }, { blankLine: 'always', next: 'block-like', prev: '\*' }, { blankLine: 'always', prev: 'export', next: '\*' }, { blankLine: 'always', next: 'export', prev: '\*' }, { prev: 'singleline-const', blankLine: 'always', next: '\*' }, { prev: 'singleline-let', blankLine: 'always', next: '\*' }, { next: 'singleline-const', blankLine: 'always', prev: '\*' }, { next: 'singleline-let', blankLine: 'always', prev: '\*' }, { prev: 'singleline-let', next: 'singleline-let', blankLine: 'any' }, { prev: 'singleline-const', next: 'singleline-const', blankLine: 'any', }, ], 'prettier/prettier': [ 'error', { jsxSingleQuote: false, endOfLine: 'auto', singleQuote: true, }, ], 'no-promise-executor-return': 'error', 'array-callback-return': 'error', 'no-unused-expressions': 'error', 'prefer-arrow-callback': 'error', 'prefer-destructuring': 'error', 'consistent-return': 'error', 'arrow-body-style': 'error', 'object-shorthand': 'error', 'no-return-assign': 'error', 'no-await-in-loop': 'error', 'no-throw-literal': 'error', 'no-extend-native': 'error', 'no-return-await': 'error', 'prefer-template': 'error', 'no-else-return': 'error', 'accessor-pairs': 'error', 'no-lone-blocks': 'error', 'require-await': 'error', 'prefer-const': 'error', 'dot-notation': 'error', 'no-multi-str': 'error', 'camelcase': 'error', 'no-proto': 'error', 'curly': 'error', }, languageOptions: { globals: { ...globals.browser, ...globals.es2025, ...globals.jest, }, }, plugins: { stylistic: stylisticJsPlugin, prettier: prettierPlugin, }, files: ['\*\*/\*.{js,ts}'], }, { rules: { ...tseslint.configs.recommended.rules, '@typescript-eslint/explicit-function-return-type': 'off', '@typescript-eslint/no-explicit-any': 'warn', '@typescript-eslint/no-unused-vars': 'warn', }, languageOptions: { parserOptions: { project: './tsconfig.json', }, parser: tsParser, }, plugins: { '@typescript-eslint': tseslint, }, files: ['\*\*/\*.ts'], }, { ignores: ['node\_modules', 'vite.config.js', 'dist'], }, ];

**package.json**

{ "name": "elib", "version": "0.0.0", "devDependencies": { "@eslint/js": "^9.20.0", "@stylistic/eslint-plugin-js": "^3.1.0", "@typescript-eslint/eslint-plugin": "^8.26.0", "@typescript-eslint/parser": "^8.26.0", "eslint": "^9.20.0", "eslint-plugin-perfectionist": "^4.8.0", "eslint-plugin-prettier": "^5.2.3", "globals": "^15.14.0", "prettier": "^3.5.0", "vite": "^6.1.0" }, "private": true, "scripts": { "dev": "vite", "build": "vite build", "preview": "vite preview", "lint": "eslint . --ext .js,.ts", "lint:fix": "eslint . --ext .js,.ts --fix", "format": "prettier --write .", "fix": "bun lint:fix && bun format", "server": "bun run server/index.ts" }, "type": "module", "dependencies": { "@popperjs/core": "^2.11.8", "@types/bootstrap": "^5.2.10", "@types/cors": "^2.8.17", "@types/express": "^5.0.0", "@types/express-fileupload": "^1.5.1", "@types/node": "^22.13.10", "bootstrap": "^5.3.3", "bootstrap-icons": "^1.11.3", "cors": "^2.8.5", "epubjs": "^0.3.93", "express": "^4.21.2", "express-fileupload": "^1.5.1", "sass": "^1.85.1", "typescript": "^5.8.2" } }

**server\bookIO.ts**

import fs from 'node:fs'; import { BOOKS\_PATH, Book } from './constants'; export async function updateJson(book: Book) { try { const data = await readJson(); console.log('data', data); data.push(book); await fs.promises.writeFile( BOOKS\_PATH, JSON.stringify(data, null, 2), 'utf8', ); return true; } catch (error) { console.error('Ошибка при обновлении файла', error); return false; } } export async function readJson() { try { const data = await fs.promises.readFile(BOOKS\_PATH, 'utf8'); const jsonData = JSON.parse(data) as Book[]; console.log(jsonData); return jsonData; } catch (error) { console.error('Ошибка при чтении файла', error); return []; } }

**server\constants.ts**

import { fileURLToPath } from 'node:url'; import path from 'node:path'; const \_\_filename = fileURLToPath(import.meta.url); const \_\_dirname = path.dirname(\_\_filename); export const BOOKS\_PATH = path.join( \_\_dirname, '..', 'src', 'public', 'books.json', ); console.log(BOOKS\_PATH); export type Book = { author: string; title: string; cover: string; genre: string; path: string; };

**server\index.ts**

import fileUpload from 'express-fileupload'; import express from 'express'; import path from 'node:path'; import cors from 'cors'; import type { Book } from './constants'; import { updateJson, readJson } from './bookIO'; const app = express(); app.use(cors()); app.use(fileUpload()); app.post('/add-book', async (request: express.Request, response: any) => { if (!request.files || !request.files.file) { return response.status(400).send('Файл не загружен'); } const file = request.files.file as fileUpload.UploadedFile; const poster = request.files.poster as fileUpload.UploadedFile; if (!file.name.endsWith('.epub')) { return response.status(400).json({ message: 'Пожалуйста, загрузите файл в формате EPUB', }); } if (!(poster.name.endsWith('.jpg') || poster.name.endsWith('.png'))) { return response.status(400).json({ message: 'Пожалуйста, загрузите обложку в формате JPG или PNG', }); } if ( await checkBook({ author: request.body.author, title: request.body.title, genre: request.body.genre, cover: poster.name, path: file.name, }) ) { return response.status(400).json({ message: 'Книга с таким названием уже существует', }); } file.mv(path.join(\_\_dirname, '..', 'src', 'public', file.name), err => { if (err) { console.error(err); return response.status(500).json({ message: 'Ошибка при сохранении файла', }); } }); console.log( '1', path.join( \_\_dirname, '..', 'src', 'public', 'images', 'covers', `/images/covers/${poster.name}`, ), ); poster.mv( path.join( \_\_dirname, '..', 'src', 'public', 'images', 'covers', `${poster.name}`, ), err => { if (err) { console.error(err); return response.status(500).json({ message: 'Ошибка при сохранении обложки', }); } }, ); await updateJson({ cover: `/images/covers/${poster.name}`, author: request.body.author, title: request.body.title, genre: request.body.genre, path: file.name, }); return response.status(200).json({ message: 'Книга успешно загружена', }); }); app.get('/books', async (\_request: express.Request, response: any) => { const books = await readJson(); console.log('get books: books to send', books); return response.status(200).json(books); }); app.listen(3000, () => { console.log('Server is running on port 3000'); }); async function checkBook(book: Book) { const books = await readJson(); return Array.isArray(books) && books.some(b => b.title === book.title); }

**src\admin.html**

<!doctype html> <html lang="en"> <head> <meta charset="UTF-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <title>Админ панель</title> <link rel="icon" href="/icon.svg" /> <script src="../scripts/book/bookService.ts" type="module" defer></script> <script src="../scripts/index.ts" type="module" defer></script> <link rel="stylesheet" href="../styles/style.scss" /> </head> <body> <header class="border-bottom d-flex justify-content-between"> <a href="/" class=""> <img src="/icon.svg" alt="Elib logo" /> </a> <section class="flex-grow-1 d-flex justify-content-around justify-content-md-around justify-content-lg-between align-items-center gap-2 px-2" > <form class="flex-grow-1 d-flex max-width-500 align-items-center gap-2"> <input name="search" class="flex-grow-1 form-input" type="text" placeholder="Поиск" spellcheck /> <button type="submit" class="btn-secondary"> <img src="/images/icons/search.svg" alt="search icon" /> </button> </form> <nav class="dropdown"> <div class="btn-secondary" type="button" id="dropdownMenuButton1" data-bs-toggle="dropdown" data-bs-auto-close="true" aria-expanded="false" > <img src="/images/icons/account.svg" alt="user icon" /> </div> <ul class="dropdown-menu dropdown-menu-end dropdown-menu-lg" aria-labelledby="dropdownMenuButton1" data-bs-popper="static" > <li><a class="dropdown-item" href="/user">user</a></li> <li> <a class="dropdown-item" href="/sertificates">Сертификаты</a> </li> <li><hr class="dropdown-divider" /></li> <li> <a class="dropdown-item" href="/logout">Выйти</a> </li> </ul> </nav> </section> </header> <main class="main d-flex flex-column justify-content-center align-items-center" > <section class="container"> <div class="row align-items-center mt-0 mt-md-4 mt-lg-4"> <h1 class="text-center fs-2 mb-0">Добавление книги:</h1> </div> </section> <form id="bookForm" class="container mt-0 mt-md-4 mt-lg-4 max-width-500"> <div class="mb-0 mb-md-2 mb-lg-4"> <label for="genre" class="form-label">Жанр</label> <select class="form-select" id="genre" name="genre" required> <option value="" selected disabled>Выберите жанр</option> <option value="Фэнтези">Фэнтези</option> <option value="Научная фантастика">Научная фантастика</option> <option value="Детектив">Детектив</option> <option value="Роман">Роман</option> <option value="Ужасы">Ужасы</option> <option value="Триллер">Триллер</option> <option value="Классика">Классика</option> <option value="Другое">Другое</option> </select> </div> <div class="mb-0 mb-md-2 mb-lg-4"> <label for="title" class="form-label">Название книги</label> <input type="text" class="form-control" id="title" name="title" placeholder="Название книги" /> </div> <div class="mb-0 mb-md-2 mb-lg-4"> <label for="author" class="form-label">Автор</label> <input type="text" class="form-control" id="author" name="author" placeholder="Автор" /> </div> <div class="mb-0 mb-md-2 mb-lg-4"> <label for="poster">Обложка (png, jpg)</label> <input type="file" class="form-control" id="poster" name="poster" accept=".jpg, .jpeg, .png" required /> </div> <div class="mb-2 mb-md-4 mb-lg-6"> <label for="file" class="form-label">Файл книги (EPUB)</label> <input type="file" class="form-control" name="file" id="file" accept=".epub" required /> </div> <div class="d-flex justify-content-center"> <button type="submit" class="btn btn-primary">Загрузить книгу</button> </div> </form> </main> <footer class="footer"> <div class="footer\_\_img"> <img class="footer\_\_img--content" src="/icon.svg" alt="logo" /> </div> <div class="flex-grow-1 d-flex gap-4 justify-content-between"> <div class="footer\_\_links d-flex flex-column flex-md-row flex-lg-row gap-2" > <a class="link" href="/">Главная</a> <a class="link" href="/register">Регистрация</a> <a class="link" href="/sertificates">Купоны и бонусы</a> </div> <div>2025 &#128218;</div> </div> </footer> </body> </html>

**src\book.html**

<!doctype html> <html lang="ru"> <head> <meta charset="UTF-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <link rel="icon" type="image/svg" href="/icon.svg" /> <title>Книга</title> <link href="./styles/custom.css" rel="stylesheet" /> <script type="module" src="./scripts/book/book.ts" defer></script> </head> <body> <main class="book-reader d-flex align-items-center position-relative vh-100" > <button id="prev" type="button" class="btn btn-primary position-absolute start-0 top-50 translate-middle-y d-flex align-items-center justify-content-center p-3 z-3" > <img src="/images/icons/arrow-back.svg" alt="Назад" /> </button> <div id="area" class="w-100 h-100"></div> <button id="next" type="button" class="btn btn-primary position-absolute end-0 top-50 translate-middle-y d-flex align-items-center justify-content-center p-3 z-3" > <img src="/images/icons/arrow-forward.svg" alt="Вперед" /> </button> </main> </body> </html>

**src\index.html**

<!doctype html> <html lang="ru"> <head> <meta charset="UTF-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <link rel="icon" type="image/svg" href="/icon.svg" /> <title>Elib - онлайн библиотека</title> <script src="../scripts/index.ts" type="module"></script> <script src="../scripts/book/bookFetcher.ts" type="module"></script> <link rel="stylesheet" href="../styles/style.scss" /> </head> <body> <header class="border-bottom d-flex justify-content-between"> <a href="/" class=""> <img src="/icon.svg" alt="Elib logo" /> </a> <section class="flex-grow-1 d-flex justify-content-around justify-content-md-around justify-content-lg-between align-items-center gap-2 px-2" > <form class="flex-grow-1 d-flex max-width-500 align-items-center gap-2"> <input name="search" class="flex-grow-1 form-input" type="text" placeholder="Поиск" spellcheck /> <button type="submit" class="btn-secondary"> <img src="/images/icons/search.svg" alt="search icon" /> </button> </form> <nav class="dropdown"> <div class="btn-secondary" type="button" id="dropdownMenuButton1" data-bs-toggle="dropdown" data-bs-auto-close="true" aria-expanded="false" > <img src="/images/icons/account.svg" alt="user icon" /> </div> <ul class="dropdown-menu dropdown-menu-end dropdown-menu-lg" aria-labelledby="dropdownMenuButton1" data-bs-popper="static" > <li><a class="dropdown-item" href="/user">user</a></li> <li> <a class="dropdown-item" href="/sertificates">Сертификаты</a> </li> <li><hr class="dropdown-divider" /></li> <li> <a class="dropdown-item" href="/logout">Выйти</a> </li> </ul> </nav> </section> </header> <main class="main-content"></main> <footer class="footer"> <div class="footer\_\_img"> <img class="footer\_\_img--content" src="/icon.svg" alt="logo" /> </div> <div class="flex-grow-1 d-flex gap-4 justify-content-between"> <div class="footer\_\_links d-flex flex-column flex-md-row flex-lg-row gap-2" > <a class="link" href="/">Главная</a> <a class="link" href="/register">Регистрация</a> <a class="link" href="/sertificates">Купоны и бонусы</a> </div> <div>2025 &#128218;</div> </div> </footer> </body> </html>

**src\login.html**

<!doctype html> <html lang="ru"> <head> <meta charset="UTF-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <link rel="icon" type="image/svg" href="/icon.svg" /> <title>Вход</title> <link rel="stylesheet" href="../styles/style.scss" /> </head> <body> <header class="border-bottom d-flex justify-content-between"> <a href="/" class=""> <img src="/icon.svg" alt="Elib logo" /> </a> <section class="flex-grow-1 d-flex justify-content-around justify-content-md-around justify-content-lg-between align-items-center gap-2 px-2" > <form class="flex-grow-1 d-flex max-width-500 align-items-center gap-2"> <input name="search" class="flex-grow-1 form-input" type="text" placeholder="Поиск" spellcheck /> <button type="submit" class="btn-secondary"> <img src="/images/icons/search.svg" alt="search icon" /> </button> </form> <nav class="dropdown"> <div class="btn-secondary" type="button" id="dropdownMenuButton1" data-bs-toggle="dropdown" data-bs-auto-close="true" aria-expanded="false" > <img src="/images/icons/account.svg" alt="user icon" /> </div> <ul class="dropdown-menu dropdown-menu-end dropdown-menu-lg" aria-labelledby="dropdownMenuButton1" data-bs-popper="static" > <li><a class="dropdown-item" href="/user">user</a></li> <li> <a class="dropdown-item" href="/sertificates">Сертификаты</a> </li> <li><hr class="dropdown-divider" /></li> <li> <a class="dropdown-item" href="/logout">Выйти</a> </li> </ul> </nav> </section> </header> <main class="d-grid grid-cols-1 grid-md-cols-2 align-items-center justify-content-center gap-4" > <section class="p-4 d-flex justify-content-center"> <div class="border rounded p-4 d-flex flex-column gap-2 px-4 fit-content" > <h1 class="text-center">С возвращением!</h1> <form class="d-flex flex-column gap-2 px-4"> <!-- <label class="text-center" for="email">введите email</label> --> <input id="email" name="email" type="email" class="border rounded auth-input" autocomplete="email" placeholder="Email" required /> <!-- <label class="text-center" for="password">придумайте пароль</label> --> <input id="password" name="password" type="password" autocomplete="new-password" class="border rounded auth-input" placeholder="Пароль" required /> <div class="d-flex justify-content-center"> <button type="submit" class="btn-primary">Войти</button> </div> </form> <a class="link" href="./register">зарегистрироваться</a> </div> </section> <section class="d-none d-md-block"> <img src="/images/posters/book-shkaf.svg" /> </section> </main> <footer class="footer"> <div class="footer\_\_img"> <img class="footer\_\_img--content" src="/icon.svg" alt="logo" /> </div> <div class="flex-grow-1 d-flex gap-4 justify-content-between"> <div class="footer\_\_links d-flex flex-column flex-md-row flex-lg-row gap-2" > <a class="link" href="/">Главная</a> <a class="link" href="/register">Регистрация</a> <a class="link" href="/sertificates">Купоны и бонусы</a> </div> <div>2025 &#128218;</div> </div> </footer> </body> </html>

**src\public\books.json**

[ { "title": "Шерлок Холмс", "author": "Артур Конан Дойл", "path": "sherlockholmes.epub", "genre": "Детектив", "cover": "/images/covers/sherlockholmes.jpg" }, { "title": "Гарри Поттер и философский камень", "author": "Джоан Роулинг", "path": "harrypotter.epub", "genre": "Фэнтези", "cover": "/images/covers/harrypotter.jpg" }, { "cover": "/images/covers/cover\_370796.jpg", "author": "mnbbj", "title": "hgjgyfyt", "genre": "Фэнтези", "path": "92192227.epub" } ]

**src\register.html**

<!doctype html> <html lang="ru"> <head> <meta charset="UTF-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <link rel="icon" type="image/svg" href="/icon.svg" /> <title>Регистрация</title> <link rel="stylesheet" href="../styles/style.scss" /> </head> <body> <header class="border-bottom d-flex justify-content-between"> <a href="/" class=""> <img src="/icon.svg" alt="Elib logo" /> </a> <section class="flex-grow-1 d-flex justify-content-around justify-content-md-around justify-content-lg-between align-items-center gap-2 px-2" > <form class="flex-grow-1 d-flex max-width-500 align-items-center gap-2"> <input name="search" class="flex-grow-1 form-input" type="text" placeholder="Поиск" spellcheck /> <button type="submit" class="btn-secondary"> <img src="/images/icons/search.svg" alt="search icon" /> </button> </form> <nav class="dropdown"> <div class="btn-secondary" type="button" id="dropdownMenuButton1" data-bs-toggle="dropdown" data-bs-auto-close="true" aria-expanded="false" > <img src="/images/icons/account.svg" alt="user icon" /> </div> <ul class="dropdown-menu dropdown-menu-end dropdown-menu-lg" aria-labelledby="dropdownMenuButton1" data-bs-popper="static" > <li><a class="dropdown-item" href="/user">user</a></li> <li> <a class="dropdown-item" href="/sertificates">Сертификаты</a> </li> <li><hr class="dropdown-divider" /></li> <li> <a class="dropdown-item" href="/logout">Выйти</a> </li> </ul> </nav> </section> </header> <main class="d-grid grid-cols-1 grid-md-cols-2 align-items-center justify-content-center gap-4" > <section class="d-flex justify-content-center p-4"> <div class="border rounded p-4 d-flex flex-column gap-2 px-4 justify-content-center" > <h1 class="text-center">Добро пожаловать!</h1> <form class="d-flex flex-column gap-2 px-4"> <label class="text-center" for="email">введите email</label> <input id="email" name="email" type="email" class="border rounded auth-input" autocomplete="email" placeholder="Email" required /> <label class="text-center" for="password">придумайте пароль</label> <input id="password" name="password" type="password" autocomplete="new-password" class="border rounded auth-input" placeholder="Пароль" required /> <label class="text-center" for="cc-number" >и отдайте все ваши деньги</label > <input id="cc-number" type="number" autocomplete="cc-number" class="border rounded auth-input" placeholder="xxxx xxxx xxxx xxxx" /> <div class="d-flex justify-content-center"> <button type="submit" class="btn-primary"> Зарегистрироваться </button> </div> </form> <a class="link" href="./login">войти</a> </div> </section> <section class="d-none d-md-block"> <img src="/images/posters/book-shkaf.svg" /> </section> </main> <footer class="footer"> <div class="footer\_\_img"> <img class="footer\_\_img--content" src="/icon.svg" alt="logo" /> </div> <div class="flex-grow-1 d-flex gap-4 justify-content-between"> <div class="footer\_\_links d-flex flex-column flex-md-row flex-lg-row gap-2" > <a class="link" href="/">Главная</a> <a class="link" href="/register">Регистрация</a> <a class="link" href="/sertificates">Купоны и бонусы</a> </div> <div>2025 &#128218;</div> </div> </footer> </body> </html>

**src\scripts\book\book.ts**

import { BookReader } from './bookReader'; let reader: BookReader | null = null; document.addEventListener('DOMContentLoaded', async () => { console.log('[DOM] Content loaded'); try { const params = new URLSearchParams(window.location.search); const bookPath = params.get('book'); console.log('[DOM] Book path from URL:', bookPath); if (!bookPath) { throw new Error('Add ?book= parameter to URL'); } reader = new BookReader(bookPath.replace('-', '').concat('.epub')); console.log('[DOM] Reader instance created'); await reader.init(); console.log('[DOM] Reader initialization completed'); } catch (error) { console.error('[DOM Error]', error); } }); const [next, prev] = document.querySelectorAll('button'); next?.addEventListener('click', async () => { console.log('[UI] Prev button clicked'); await reader?.prevPage(); }); prev?.addEventListener('click', async () => { console.log('[UI] Next button clicked'); await reader?.nextPage(); });

**src\scripts\book\bookFetcher.ts**

import { Book } from '../../../server/constants'; document.addEventListener('DOMContentLoaded', async () => { const response = await fetch('http://localhost:3000/books'); console.log(response); if (!response.ok) { console.warn('Ошибка при получении книг', await response.text()); return; } const books = await response.json(); console.log(books, Object.keys(books).length); if (!books || Object.keys(books).length === 0) { console.warn('Книги не найдены'); return; } const container = document.querySelector('.main-content'); const booksByGenre: Record<string, HTMLElement[]> = {}; console.log(container); books.forEach((book: Book) => { const card = document.createElement('product-card'); card.setAttribute('name', book.title); card.setAttribute('link', '/'); card.setAttribute('img', book.cover); if (!booksByGenre[book.genre]) { booksByGenre[book.genre] = [card]; return; } booksByGenre[book.genre].push(card); }); console.log(booksByGenre); Object.entries(booksByGenre).forEach(([genre, books]) => { const genreContainer = document.createElement('section'); genreContainer.innerHTML = ` <div class="main-content\_\_heading-block"> <h2>${genre}</h2> <hr /> </div> <div class="main-content\_\_books"> ${books.map(book => book.outerHTML).join('')} </div> `; console.log(genre, books); container?.appendChild(genreContainer); }); });

**src\scripts\book\bookReader.ts**

import ePub from 'epubjs'; export class BookReader { private rendition: ePub.Rendition | null; private isReady: boolean; private book: ePub.Book; constructor(bookPath: string) { console.log('[Constructor] Initializing with book path:', bookPath); this.book = ePub(bookPath); this.rendition = null; this.isReady = false; this.book.on('error', (error: unknown) => { console.error('[Book Error]', error); }); } async init() { try { console.log('[Init] Starting initialization...'); await this.book.ready; console.log('[Init] Book metadata:', (this.book as any).metadata); console.log('[Init] Creating rendition...'); this.rendition = this.book.renderTo('area', { allowScriptedContent: true, height: '100%', spread: 'none', width: '100%', }); console.log('[Init] Displaying content...'); await this.rendition.display(); this.isReady = true; console.log('[Init] Rendition ready:', this.rendition); console.log('[Init] Current location:', this.rendition.currentLocation()); } catch (error) { console.error('[Init Error]', error); throw error; } } async prevPage() { console.log('[Navigation] Attempting previous page...'); if (!this.isReady) { console.warn('[Navigation] Reader not ready!'); return; } try { await this.rendition?.prev(); console.log( '[Navigation] New location:', this.rendition?.currentLocation(), ); } catch (error) { console.error('[Navigation Error]', error); } } async nextPage() { console.log('[Navigation] Attempting next page...'); if (!this.isReady) { console.warn('[Navigation] Reader not ready!'); return; } try { await this.rendition?.next(); console.log( '[Navigation] New location:', this.rendition?.currentLocation(), ); } catch (error) { console.error('[Navigation Error]', error); } } }

**src\scripts\book\bookService.ts**

const bookForm = document.getElementById('bookForm') as HTMLFormElement | null; const handleAddBook = async (event: SubmitEvent) => { event.preventDefault(); if (!bookForm) { console.error('Book form not found'); return; } const formData = new FormData(bookForm); const file = formData.get('file') as File; const poster = formData.get('poster') as File; console.log(Object.fromEntries(formData)); if (!file.name.endsWith('.epub')) { alert('Пожалуйста, загрузите файл в формате EPUB'); return; } if (!(poster.name.endsWith('.jpg') || poster.name.endsWith('.png'))) { alert('Пожалуйста, загрузите обложку в формате JPG или PNG'); return; } const response = await fetch('http://localhost:3000/add-book', { method: 'POST', body: formData, }); if (!response.ok) { console.warn('Ошибка при загрузке книги', await response.text()); return; } console.log('Книга успешно загружена', await response.json()); alert('Книга успешно загружена'); }; bookForm?.addEventListener('submit', handleAddBook);

**src\scripts\components\card.ts**

import stylesString from './card.scss?inline'; export class Card extends HTMLElement { static get observedAttributes() { return ['img', 'name', 'link', 'icon']; } /\* захардкоженная иконка на кнопке \*/ private \_icon: string = '/images/icons/add.svg'; private \_link: string = ''; private \_name: string = ''; /\* обложка \*/ private \_img: string = ''; constructor() { super(); } render() { if (this.\_link !== '/') { this.\_icon = '/images/icons/resume.svg'; } return ` <style> ${stylesString} </style> <div class="card"> <div class="card\_\_cover"> <img class="card\_\_cover\_\_img" src="${ this.\_img }" alt="${this.\_name} cover" lazy /> </div> <div class="card\_\_description"> <span class="card\_\_description\_\_text"> ${this.\_name} </span> <button class="card\_\_button"> <a href="${this.\_link}"> <img src="${this.\_icon}" class="card\_\_button\_\_img" alt="add icon" /> </a> </button> </div> </div> `; } attributeChangedCallback(name: string, oldValue: string, newValue: string) { if (name === 'img') { this.\_img = newValue; } else if (name === 'href') { this.\_link = newValue; } else if (name === 'name') { this.\_name = newValue; } else if (name === 'icon') { if (newValue) { this.\_icon = newValue; } } } updateData() { this.\_img = this.getAttribute('img') ?? ''; this.\_link = this.getAttribute('link') ?? ''; this.\_name = this.getAttribute('name') ?? ''; this.\_icon = this.getAttribute('icon') ?? '/images/icons/add.svg'; } connectedCallback() { this.attachShadow({ mode: 'open' }); this.updateData(); this.shadowRoot!.innerHTML = this.render(); } }

**src\scripts\index.ts**

// eslint-disable-next-line @typescript-eslint/no-unused-vars import \* as bootstrap from 'bootstrap'; import 'bootstrap-icons/font/bootstrap-icons.css'; console.log(bootstrap); import { Card } from './components/card'; customElements.define('product-card', Card);

**src\sertificates.html**

<!doctype html> <html lang="ru"> <head> <meta charset="UTF-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <link rel="icon" type="image/svg" href="/icon.svg" /> <title>Сертификаты</title> <link rel="stylesheet" href="../styles/style.scss" /> </head> <body> <header class="border-bottom d-flex justify-content-between"> <a href="/" class=""> <img src="/icon.svg" alt="Elib logo" /> </a> <section class="flex-grow-1 d-flex justify-content-around justify-content-md-around justify-content-lg-between align-items-center gap-2 px-2" > <form class="flex-grow-1 d-flex max-width-500 align-items-center gap-2"> <input name="search" class="flex-grow-1 form-input" type="text" placeholder="Поиск" spellcheck /> <button type="submit" class="btn-secondary"> <img src="/images/icons/search.svg" alt="search icon" /> </button> </form> <nav class="dropdown"> <div class="btn-secondary" type="button" id="dropdownMenuButton1" data-bs-toggle="dropdown" data-bs-auto-close="true" aria-expanded="false" > <img src="/images/icons/account.svg" alt="user icon" /> </div> <ul class="dropdown-menu dropdown-menu-end dropdown-menu-lg" aria-labelledby="dropdownMenuButton1" data-bs-popper="static" > <li><a class="dropdown-item" href="/user">user</a></li> <li> <a class="dropdown-item" href="/sertificates">Сертификаты</a> </li> <li><hr class="dropdown-divider" /></li> <li> <a class="dropdown-item" href="/logout">Выйти</a> </li> </ul> </nav> </section> </header> <main class="d-flex flex-column gap-4 justify-content-center"> <div class="d-flex justify-content-center p-4"> <div class="d-flex align-items-center justify-content-between border rounded p-5" > <h2 class="text-center">Дарите знания!</h2> <div class="overflow-hidden"> <img class="w-100" src="/images/posters/book-present.svg" alt="banner logo" /> </div> </div> </div> <div class="d-grid grid-cols-1 grid-md-cols-2 gap-4 p-4"> <section class="col d-flex justify-content-center border rounded p-4"> <div class="d-flex justify-content-center"> <img class="w-100" src="/images/posters/book-3-m.svg" alt="banner logo" /> </div> <div class="d-flex flex-column gap-2 align-items-center justify-content-center" > <h3 class="text-center">Подписка на 3 месяца</h3> <span class="text-center" >Всё, от Гарри Поттера до Незнайки. Открывайте для себя новые удивительные миры и занимательные истории!</span > <button type="button" class="btn-primary">Купить</button> </div> </section> <section class="col d-flex justify-content-center border rounded p-4"> <div class="d-flex justify-content-center"> <img class="w-100" src="/images/posters/book-3-m.svg" alt="banner logo" /> </div> <div class="d-flex flex-column gap-2 align-items-center justify-content-center" > <h3 class="text-center">Подписка на 3 месяца</h3> <span class="text-center" >Всё, от Гарри Поттера до Незнайки. Открывайте для себя новые удивительные миры и занимательные истории!</span > <button type="button" class="btn-primary">Купить</button> </div> </section> </div> </main> <footer class="footer"> <div class="footer\_\_img"> <img class="" src="/icon.svg" alt="logo" /> </div> <div class="flex-grow-1 d-flex gap-4 justify-content-between"> <div class="d-flex flex-column flex-md-row flex-lg-row gap-2"> <a class="link" href="/">Главная</a> <a class="link" href="/register">Регистрация</a> <a class="link" href="/sertificates">Купоны и бонусы</a> </div> <div>2025 &#128218;</div> </div> </footer> </body> </html>

**src\user.html**

<!doctype html> <html lang="ru"> <head> <meta charset="UTF-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <title>Личный кабинет</title> <link rel="icon" type="image/svg" href="/icon.svg" /> <link rel="stylesheet" href="../styles/style.scss" /> <script type="module" src="../scripts/index.ts"></script> </head> <body> <header class="border-bottom d-flex justify-content-between"> <a href="/" class=""> <img src="/icon.svg" alt="Elib logo" /> </a> <section class="flex-grow-1 d-flex justify-content-around justify-content-md-around justify-content-lg-between align-items-center gap-2 px-2" > <form class="flex-grow-1 d-flex max-width-500 align-items-center gap-2"> <input name="search" class="flex-grow-1 form-input" type="text" placeholder="Поиск" spellcheck /> <button type="submit" class="btn-secondary"> <img src="/images/icons/search.svg" alt="search icon" /> </button> </form> <nav class="dropdown"> <div class="btn-secondary" type="button" id="dropdownMenuButton1" data-bs-toggle="dropdown" data-bs-auto-close="true" aria-expanded="false" > <img src="/images/icons/account.svg" alt="user icon" /> </div> <ul class="dropdown-menu dropdown-menu-end dropdown-menu-lg" aria-labelledby="dropdownMenuButton1" data-bs-popper="static" > <li><a class="dropdown-item" href="/user">user</a></li> <li> <a class="dropdown-item" href="/sertificates">Сертификаты</a> </li> <li><hr class="dropdown-divider" /></li> <li> <a class="dropdown-item" href="/logout">Выйти</a> </li> </ul> </nav> </section> </header> <main class="main-content"> <section> <div class="main-content\_\_heading-block"> <h2>Ваши книги, <span style="color: red">user</span></h2> <hr /> </div> <div class="main-content\_\_books"> <product-card img="/images/covers/harrypotter.jpg" link="/book.html?book=harry-potter" name="Гарри Поттер и философский камень" ></product-card> <product-card img="/images/covers/harrypotter.jpg" link="/book.html?book=harry-potter" name="Гарри Поттер и философский камень" ></product-card> </div> </section> </main> <footer class="footer"> <div class="footer\_\_img"> <img class="footer\_\_img--content" src="/icon.svg" alt="logo" /> </div> <div class="flex-grow-1 d-flex gap-4 justify-content-between"> <div class="footer\_\_links d-flex flex-column flex-md-row flex-lg-row gap-2" > <a class="link" href="/">Главная</a> <a class="link" href="/register">Регистрация</a> <a class="link" href="/sertificates">Купоны и бонусы</a> </div> <div>2025 &#128218;</div> </div> </footer> </body> </html>

**tsconfig.json**

{ "compilerOptions": { "target": "ES2020", "useDefineForClassFields": true, "module": "ESNext", "lib": ["ES2020", "DOM", "DOM.Iterable"], "skipLibCheck": true, "moduleResolution": "bundler", "allowImportingTsExtensions": true, "resolveJsonModule": true, "isolatedModules": true, "noEmit": true, "strict": true, "noUnusedLocals": true, "noUnusedParameters": true, "noFallthroughCasesInSwitch": true, "types": ["vite/client"] }, "include": ["src/\*\*/\*.ts", "src/\*\*/\*.js", "src/\*\*/\*.html", "src/scripts/\*\*/\*.ts", "server/\*\*/\*.ts"] }

**vite.config.js**

import { defineConfig } from 'vite'; import { resolve } from 'path'; export default defineConfig({ root: 'src', resolve: { alias: { '~bootstrap': resolve(\_\_dirname, 'node\_modules/bootstrap'), }, }, build: { outDir: '../dist', rollupOptions: { input: { index: 'index.html', book: 'book.html', login: 'login.html', register: 'register.html', sertificates: 'sertificates.html', user: 'user.html', admin: 'admin.html', }, output: { entryFileNames: 'assets/js/[name].js', chunkFileNames: 'assets/js/[name].js', assetFileNames: assetInfo => { if (/\.(css|scss|sass)$/.test(assetInfo.name)) { return 'assets/css/[name][extname]'; } return 'assets/[name][extname]'; }, }, }, }, });