

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ
ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»

**институт информационных технологий и технологического образования
кафедра информационных технологий и электронного обучения**

Основная профессиональная образовательная программа
Направление подготовки 09.03.01 Информатика и вычислительная техника
Направленность (профиль) «Технологии разработки программного обеспечения»
форма обучения – очная

Курсовая работа

по дисциплине «Организация электронной образовательной среды»

Разработка веб-портфолио программиста с использованием Vue и Nuxt

Обучающегося 4 курса
Фирсова Кирилла Александровича

Руководитель:
д.п.н, профессор

_____ Жуков Н.Н.

«_____» _____ 2025 г.

Санкт-Петербург
2025

Оглавление

Введение	3
Теоретическая часть	4
Требования к веб-портфолио программиста	4
Сравнительный обзор инструментов разработки и выбор технологического стека	5
Выводы по главе I	6
Практическая часть	7
Формирование общей архитектуры проекта	7
Реализация макета страниц	7
Разработка навигационной панели	8
Реализация страниц приложения	8
Страница “Обо мне”	8
Страница “Мои работы в вузе”	9
Реализация компонента отображения файлов	10
Организация хранения учебных работ	11
Реализация серверной логики	11
Получение списка файлов	11
Скачивание файлов	11
Интеграция клиентской и серверной частей	12
Выводы по практической части	12
Заключение	13
Список литературы	14
Приложения	17

Введение

Актуальность: в условиях высокой конкуренции на IT-рынке персональное портфолио превратилось из опции в необходимость. Это визитная карточка, платформа для самопрезентации и ключевой инструмент при поиске работы или фриланс-проектов. Статические сайты уступают место динамическим, интерактивным приложениям, которые не только представляют информацию, но и напрямую демонстрируют техническую экспертизу автора. Стек Vue/Nuxt, благодаря своей гибкости, отличной производительности и встроенным возможностям для SEO, является одним из оптимальных выборов для такой задачи.

Предмет: автоматизация процесса самопрезентации программиста посредством разработки веб-портфолио.

Цель: разработать веб-портфолио программиста с использованием Vue и Nuxt.

Для достижения цели необходимо решить следующие задачи:

1. Исследовать современные требования и лучшие практики создания веб-портфолио программиста.
2. Выбрать технологии для реализации проекта.
3. Спроектировать структуру, интерфейс и пользовательские сценарии портфолио.
4. Разработать основные функциональные модули приложения.
5. Оптимизировать производительность, SEO и доступность сайта.
6. Развернуть рабочее приложение на публичном хостинге и протестировать его.

Теоретическая часть

Требования к веб-портфолио программиста

Веб-портфолио является ключевым инструментом самопрезентации IT-специалиста и выполняет сразу несколько функций: информирование потенциальных работодателей или заказчиков, демонстрацию профессиональных компетенций, а также подтверждение практического опыта разработчика.

К основным требованиям, предъявляемым к современному веб-портфолио программиста, относятся:

1. **Информативность и структурированность** — наличие разделов с описанием навыков, проектов, опыта работы, используемых технологий и контактных данных.
2. **Наглядная демонстрация компетенций** — представление реализованных проектов с описанием архитектурных решений, стеков технологий, а также ссылками на репозитории и работающие версии приложений.
3. **Адаптивность интерфейса** — корректное отображение на различных устройствах (настольные компьютеры, планшеты, мобильные устройства).
4. **Высокая производительность** — минимальное время загрузки страниц, оптимизированные ресурсы, отсутствие избыточных запросов.
5. **SEO-оптимизация** — корректная работа мета-тегов, семантическая разметка, индексация страниц поисковыми системами.
6. **Доступность** — соблюдение базовых принципов web accessibility (контрастность, навигация с клавиатуры, альтернативный текст для изображений).

Таким образом, портфолио программиста должно выступать не просто в роли информационной страницы, а в качестве полнофункционального веб-приложения, демонстрирующего уровень технической подготовки автора.

Сравнительный обзор инструментов разработки и выбор технологического стека

Для реализации портфолио, соответствующего перечисленным требованиям, был проведён сравнительный анализ популярных фронтенд-решений.

1. Чистый HTML, CSS и JavaScript

Данный подход подходит для простых статических сайтов, однако при увеличении количества страниц и логики приводит к усложнению поддержки кода, отсутствию модульности и необходимости ручной настройки сборки, оптимизации и маршрутизации.

2. React + Next.js

Современное и мощное решение с поддержкой серверного рендеринга и статической генерации. Вместе с тем данный стек характеризуется более высоким порогом входа, необходимостью использования дополнительных библиотек и более сложной конфигурацией проекта.

3. Vue.js + Nuxt.js

Прогрессивный фреймворк, ориентированный на удобство разработки, читаемость кода и целостность экосистемы. Nuxt.js предоставляет готовую инфраструктуру для SSR и SSG, что особенно важно для портфолио с точки зрения SEO и производительности.

Выбор стека **Vue.js 3** и **Nuxt.js 3** обусловлен следующими преимуществами:

1. высокая читаемость и поддерживаемость кода;
2. использование Composition API и современных возможностей JavaScript;
3. встроенная маршрутизация и автоматическая оптимизация;
4. поддержка статической генерации сайта;
5. активное сообщество и развитая экосистема.

Выводы по главе I

В первой главе были рассмотрены современные требования к веб-портфолио программиста, а также проведён сравнительный анализ инструментов его разработки. В результате были решены следующие задачи, поставленные во введении:

1. исследованы требования и лучшие практики создания веб-портфолио;
2. обоснован выбор технологического стека для реализации проекта.

Практическая часть

Практическая часть курсовой работы посвящена поэтапной реализации веб-портфолио программиста с использованием фреймворков Vue.js 3 и Nuxt.js 3. Разработка велась с опорой на стандартную структуру проекта Nuxt, что позволило обеспечить логичное разделение ответственности между страницами, компонентами, макетами и серверной частью приложения.

Формирование общей архитектуры проекта

На начальном этапе была сформирована базовая структура проекта, соответствующая принципам «конвенция над конфигурацией», используемым в Nuxt.js. Это позволило минимизировать объём ручной настройки и сосредоточиться на реализации функциональности.

Проект был разделён на следующие ключевые каталоги:

1. pages/ — отвечает за маршрутизацию и формирование страниц;
2. components/ — содержит переиспользуемые элементы интерфейса;
3. layouts/ — определяет общий макет страниц;
4. works/ — содержит учебные работы, публикуемые в портфолио;
5. server/api/ — реализует серверную логику для работы с файлами.

Такое разделение упрощает сопровождение проекта и делает архитектуру понятной для других разработчиков.

Реализация макета страниц

Следующим шагом стала разработка основного макета приложения, расположенного в каталоге layouts/.

Файл default.vue определяет:

1. общую структуру страниц;
2. размещение навигационной панели;
3. область вывода основного контента.

Использование единого макета позволяет:

1. обеспечить визуальную целостность сайта;
2. избежать дублирования кода;

3. централизованно управлять изменениями интерфейса.

В макете подключается компонент навигации `NavBar.vue`, а также используется слот `<NuxtPage />` для динамического отображения содержимого страниц.

Разработка навигационной панели

Компонент `NavBar.vue`, размещённый в каталоге `components/`, реализует навигацию между основными разделами портфолио.

Основные функции навигационной панели:

1. переход на страницу “Обо мне”;
2. переход на страницу с работами из вуза;
3. визуальное выделение активного маршрута.

Компонент был реализован как переиспользуемый UI-элемент, что позволяет при необходимости расширять навигацию без изменения структуры страниц. Использование компонентов повышает модульность и удобство сопровождения кода. Визуальное представление навигационной панели представлено на рисунке 1.



Рисунок 1. Навигационная панель

Реализация страниц приложения

Каталог `pages/` является ключевым элементом `Nuxt.js`, так как на его основе автоматически формируется система маршрутизации.

Страница “Обо мне”

Файл `index.vue` реализует главную страницу портфолио. На данной странице размещена информация о разработчике, включая:

1. краткое описание профессионального профиля;
2. перечень ключевых навыков;
3. используемые технологии.

Данная страница выполняет функцию первичного знакомства пользователя с автором портфолио и формирует общее впечатление о его компетенциях. Разработанная страница представлена на рисунке 2.

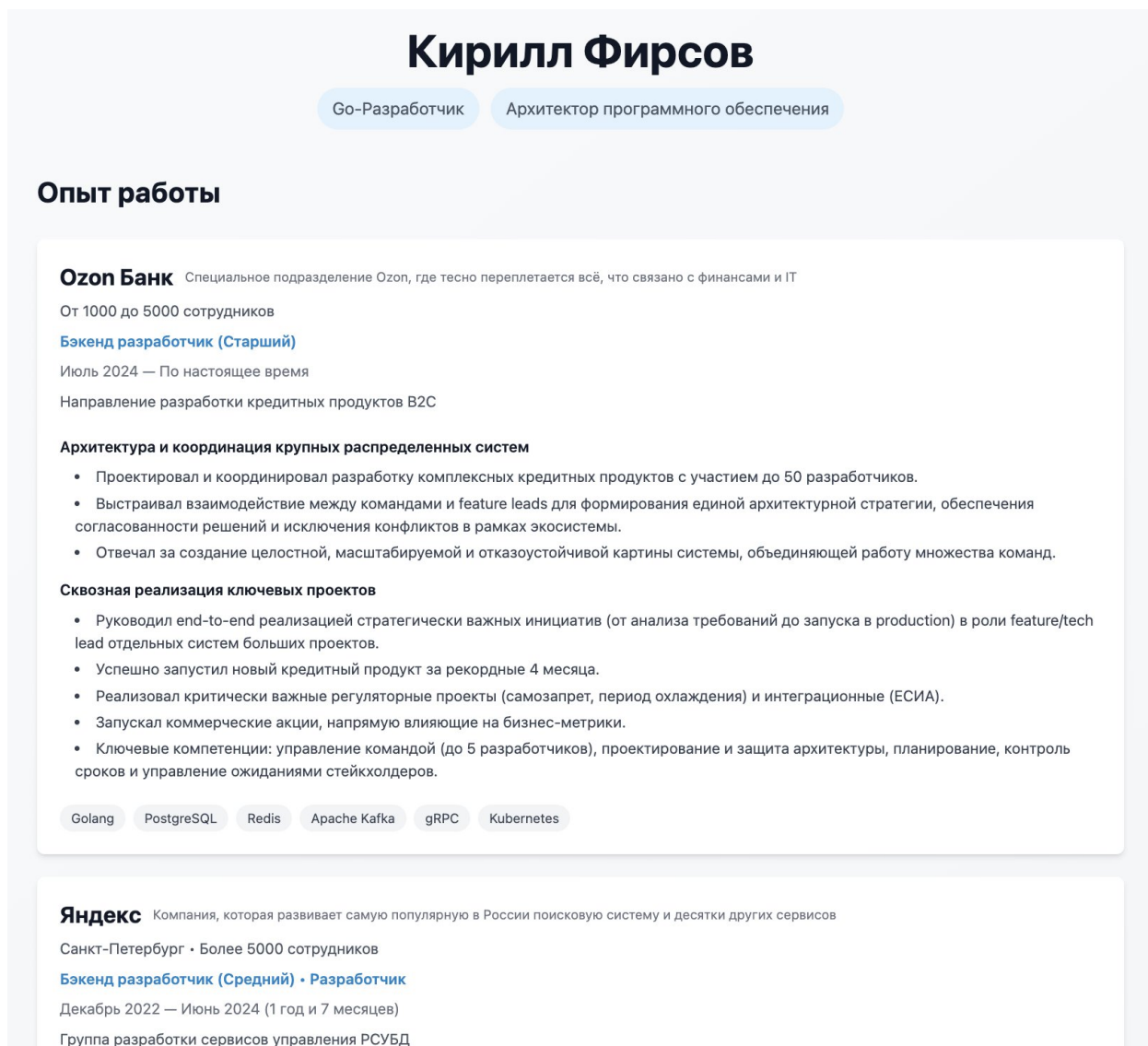


Рисунок 2. Страница "Обо мне"

Страница "Мои работы в вузе"

Страница works.vue предназначена для отображения учебных работ, выполненных в рамках образовательного процесса. В отличие от статического списка, содержимое страницы формируется динамически на основе данных, полученных с серверной части приложения.

Для отображения структуры файлов используется компонент FileTree.vue, который визуализирует иерархию папок и файлов. Разработанная страница представлена на рисунке 3.

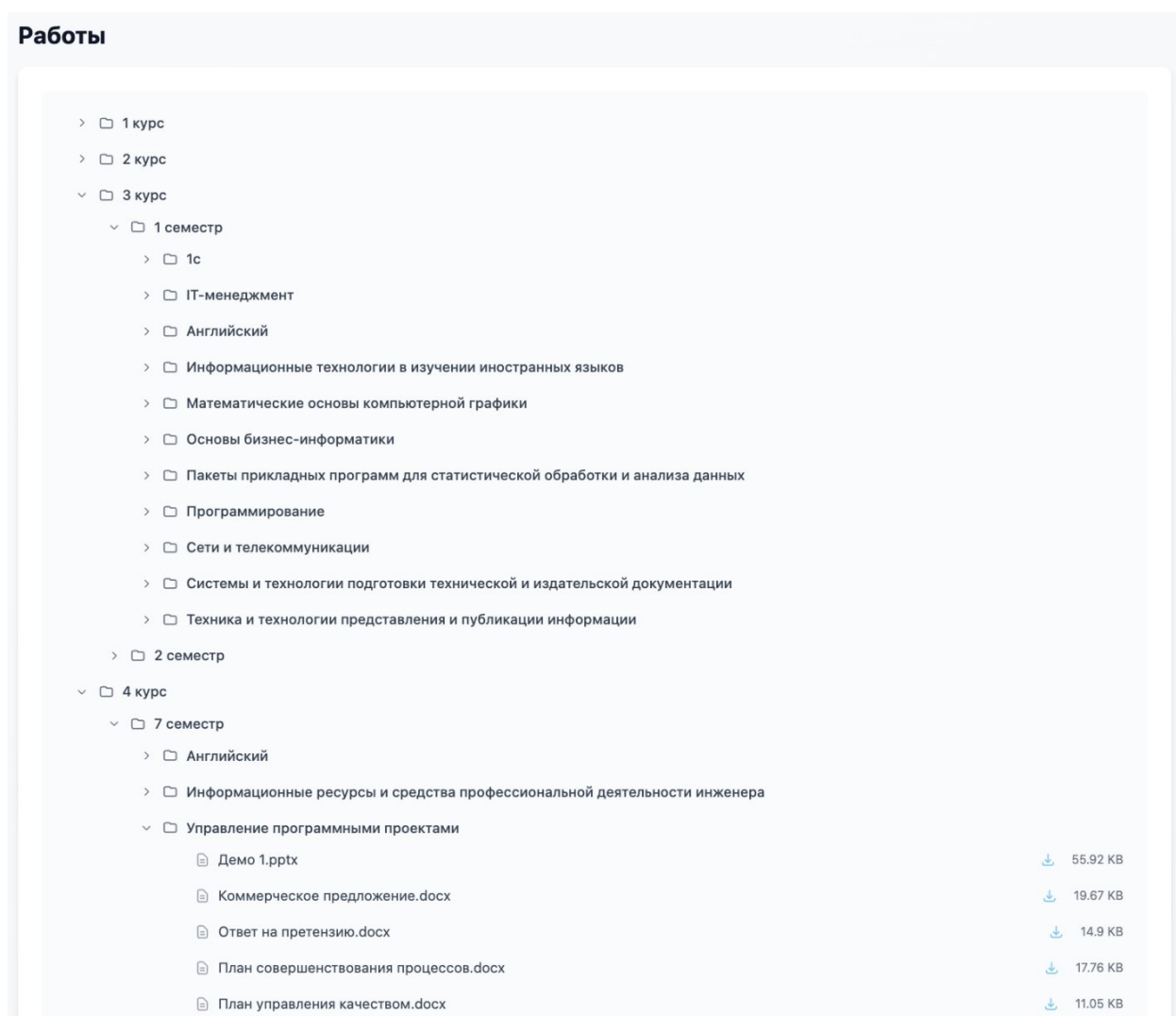


Рисунок 3. Страница "Мои работы в вузе"

Реализация компонента отображения файлов

Компонент FileTree.vue отвечает за отображение дерева файлов и папок, содержащихся в каталоге works/.

Основные задачи компонента:

1. визуализация иерархической структуры файлов;
2. поддержка вложенных директорий;

3. обработка пользовательских действий (выбор и скачивание файлов).
Компонент получает данные от серверного API и отображает их в удобном для пользователя виде. Такой подход позволяет абстрагировать логику отображения от логики получения данных.

Организация хранения учебных работ

Каталог `works/` предназначен для хранения файлов учебных работ, которые публикуются в портфолио. Добавление новых материалов осуществляется через систему контроля версий Git, что обеспечивает:

1. прозрачную историю изменений;
2. удобство обновления содержимого портфолио;
3. контроль целостности данных.

Использование файловой системы в качестве источника данных позволяет отказаться от сложной базы данных и упростить архитектуру проекта.

Реализация серверной логики

Для обеспечения доступа к файлам из каталога `works/` была реализована серверная часть приложения.

Получение списка файлов

Файл `works.get.ts` реализует API-endpoint, который:

1. сканирует содержимое каталога `works/`;
2. формирует структуру файлов и папок;
3. возвращает данные в формате JSON.

Этот endpoint используется на странице `works.vue` для динамического формирования списка доступных работ.

Скачивание файлов

Файл `[...path].get.ts` реализует catch-all маршрут, позволяющий скачивать файлы, находящиеся в любой вложенности каталогов.

Данный подход обеспечивает:

1. универсальный механизм доступа к файлам;

2. поддержку вложенных директорий;
3. безопасность за счёт контроля допустимых путей.

Интеграция клиентской и серверной частей

Заключительным этапом практической реализации стала интеграция клиентской и серверной логики. Клиентская часть запрашивает данные через API, а серверная часть отвечает за обработку файловой системы и передачу данных. Такой подход:

1. разделяет ответственность между слоями приложения;
2. повышает надёжность и масштабируемость решения;
3. соответствует современным принципам веб-разработки.

Выводы по практической части

В практической части курсовой работы была поэтапно реализована система веб-портфолио программиста с использованием Vue.js и Nuxt.js. На основе структуры проекта были реализованы страницы, компоненты, серверные API и механизм публикации учебных работ. Используемая архитектура обеспечивает удобство сопровождения, расширяемость и соответствие современным требованиям к веб-приложениям.

Заключение

В ходе выполнения курсовой работы была достигнута поставленная цель — разработано веб-портфолио программиста с использованием Vue.js и Nuxt.js. В процессе работы были изучены современные требования к портфолио, выбран и обоснован технологический стек, реализовано функциональное веб-приложение и выполнено его развёртывание.

В дальнейшем развитие проекта может включать:

1. добавление системы управления контентом;
2. интеграцию анимаций и интерактивных элементов;
3. расширение функциональности (блог, форма обратной связи);
4. дальнейшую оптимизацию доступности и пользовательского опыта.

Разработанное портфолио может использоваться как практический инструмент самопрезентации и основа для дальнейших учебных и профессиональных проектов.

Список литературы

1. Vue.js — прогрессивный JavaScript-фреймворк. — Текст : электронный // Vue.js на русском : [сайт]. — URL: <https://ru.vuejs.org/> (дата обращения: 15.11.2025).
2. Nuxt.js — фреймворк для создания SSR-приложений. — Текст : электронный // Nuxt : [сайт]. — URL: <https://nuxt.com/docs/getting-started/introduction> (дата обращения: 16.11.2025).
3. Шавин М. Изучаем Vue: основные концепции и практические паттерны для современных и масштабируемых пользовательских интерфейсов. - 1-е изд. - СПб.: БХВ Петербург, 2025. - 336 с.
4. Диков, А. В. Web-программирование на JavaScript : учебное пособие / А. В. Диков. — 4-е изд., стер. — Санкт-Петербург : Лань, 2025. — 168 с. — ISBN 978-5-507-50536-4. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/445289> (дата обращения: 18.11.2025).
5. Frain B. Responsive Web Design with HTML5 and CSS: Build future-proof responsive websites using the latest HTML5 and CSS techniques: IEEE: Packt Publishing, 2025. — 428 с. — ISBN 978-1-83702-822-1.
6. Официальное руководство по фреймворку Tailwind CSS. // Документация Tailwind CSS URL: <https://tailwindcss.com/docs/installation/using-vite> (дата обращения: 22.11.2025).
7. Янцев, В. В. Разработка web-страниц на HTML, CSS и JavaScript : учебное пособие для СПО / В. В. Янцев. — 2-е изд., стер. — Санкт-Петербург : Лань, 2025. — 148 с. — ISBN 978-5-507-52427-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/449585> (дата обращения: 25.11.2025).

8. Руководство по языку TypeScript. // Официальная документация TypeScript URL: <https://www.typescriptlang.org/docs> (дата обращения: 27.11.2025).
9. Калиберда, Е. А. Разработка web-приложений : учебное пособие / Е. А. Калиберда, К. В. Кравченко. — Омск : ОмГТУ, 2023. — 100 с. — ISBN 978-5-8149-3679-0. — Текст : электронный // Лань : 14 электронно-библиотечная система. — URL: <https://e.lanbook.com/book/421766> (дата обращения: 01.12.2025).
10. Веб-портфолио как инструмент саморепрезентации IT-специалиста // Habr. URL: <https://habr.com/ru/articles/781730/> (дата обращения: 03.12.2025).
11. Lina Sierra, Piedad Martin, M del Carmen Sierra (2012). E-portfolio, a Self-assessment Tool for Evaluating Subject Content Acquisition: A Practical Example in a European Project. Journal of Modern Education Review January 2012, Volume 2, No. 1, pp. 56–61 URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2096937 (дата обращения: 02.12.2025).
12. Ахметвалеева Л. В., Хапаева Л. Х., Алиева М., В. Обзор сайта-портфолио для представления достижений и реализации проектов. Журнал прикладных исследований 2024, doi: 10.47576/2949-1878.2024.11.11.004 URL: <https://cyberleninka.ru/article/n/obzor-sayta-portfolio-dlya-predstavleniya-dostizheniy-i-realizatsii-proektov/viewer> (дата обращения: 04.12.2025).
13. Kate Dunne, Pauline Logue (2021). A Higher Education Action Research Study on the Effectiveness of an ePortfolio as a Learning Tool to Promote Reflective Professional Development. Irish Journal of Technology Enhanced Learning (2021) Volume 6, Issue 1 URL: <https://journal.ilta.ie/index.php/telji/article/view/79/106> (дата обращения: 07.12.2025).

14. Ollila R., Mäkitalo N., Mikkonen T. Modern Web Frameworks: A Comparison of Rendering Performance: IEEE: Journal of Web Engineering, 2022. — Том 21, Вып. 3. — С. 789- 813. — DOI: 10.13052/jwe1540-9589.21312.
15. «Документация по GitHub Pages» GitHub Docs: сайт. — 2025. — URL: <https://docs.github.com/ru/pages> (дата обращения: 20.12.2025).

Приложения

1. Ссылка на портфолио: <https://fizic.github.io/herzen-7-sem-portfolio/>
2. Ссылка на код: <https://github.com/Fizic/herzen-7-sem-portfolio>