



МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ 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**

«Веб-портфолио по дисциплине "Анализ данных и основы Data science"»

Обучающегося 4 курса  
Чирцова Тимофея Александровича

Научный руководитель:  
кандидат физико-математических наук,  
доцент кафедры ИТиЭО  
Жуков Николай Николаевич

Санкт-Петербург  
2024

## ОГЛАВЛЕНИЕ

ОГЛАВЛЕНИЕ	2
ВВЕДЕНИЕ	3
Теоретическая часть	5
1.1 Понятие электронного портфолио	5
1.2 Подходы к созданию электронного портфолио	6
1.3 Использование React и связанных технологий для создания портфолио	7
Практическая часть	10
2.1 Создание Next js приложения	10
2.2 Конфигурация Next js приложения	12
2.3 Конфигурация Prettier	13
2.4 Конфигурация Tailwind	14
2.5 Глобальная настройка CSS	16
2.6 Главная страница	18
2.7 Пример страницы для задания	19
2.8 Компонент для отображения скриншотов задания	20
2.9 Компонент для скачивания файлов задания	21
2.10 Получившаяся страница портфолио	21
2.11 Развертка приложения на Github Pages	22
ЗАКЛЮЧЕНИЕ	26

## ВВЕДЕНИЕ

В современном мире информационных технологий наличие электронного портфолио становится всё более актуальным. Это связано с ростом цифровизации и необходимостью эффективно представлять свои профессиональные навыки и достижения. Особенно это важно для разработчиков, которые должны не только владеть современными инструментами и технологиями, но и уметь демонстрировать свои проекты и компетенции в удобной и наглядной форме. Электронное портфолио позволяет собрать все значимые достижения и проекты разработчика в одном месте, что делает его ценным инструментом как для профессионального роста, так и для успешной самопрезентации.

Веб-портфолио является современным решением для представления профессиональных навыков разработчика. Оно помогает не только структурировать проекты, но и сделать их доступными для потенциальных работодателей и коллег. Такие платформы, как GitHub Pages, предоставляют удобные возможности для размещения статических сайтов, а использование современных фреймворков, таких как React и Next.js, упрощает процесс создания и поддержания портфолио. Интеграция таких инструментов, как Next.js и библиотеки Tailwind CSS, позволяет не только быстро разрабатывать веб-приложения, но и создавать уникальный стиль. В руках высококвалифицированного программиста эти технологии трансформируются в мощный инструмент, который демонстрирует уровень его профессионализма через качество реализации и дизайна. Само по себе портфолио, его архитектура и стиль кода становятся индикатором мастерства разработчика.

Целью данной курсовой работы является проектирование и создание веб-портфолио на основе React и Next.js с использованием Tailwind CSS, а также последующее размещение на платформе GitHub Pages. В рамках работы будут решены следующие задачи:

1. Изучение современных подходов к созданию электронного портфолио.
2. Выбор и использование инструментов для разработки веб-приложений.
3. Реализация структуры и функционала портфолио с учётом современных требований.
4. Размещение готового проекта на платформе GitHub Pages.

Структура работы включает введение, в котором обоснована актуальность темы и определены задачи исследования, теоретическую часть с описанием выбранных технологий и инструментов, практическую часть, посвящённую разработке портфолио, и заключение, где будут подведены итоги проделанной работы.

Курсовая работа направлена на разработку функционального и удобного веб-портфолио, которое не только станет инструментом для профессиональной презентации, но и послужит примером использования современных технологий веб-разработки. Визуальный стиль портфолио и качество написанного кода будут демонстрировать уровень профессионализма автора и его подход к деталям.

### 1.1 Понятие электронного портфолио

Электронное портфолио представляет собой современный инструмент, позволяющий систематизировать информацию о профессиональных достижениях, навыках и опыте человека. Это многофункциональный цифровой ресурс, который используется для демонстрации компетенций, хранения проектов и достижения образовательных или профессиональных целей. Его уникальность заключается в возможности совмещения текстовой, визуальной и аналитической информации для наиболее полного и убедительного представления о пользователе.

Значимость электронного портфолио особенно актуальна для студентов, так как оно становится важным элементом их образовательного процесса. Электронное портфолио позволяет структурировать достижения, визуализировать их в удобной и эстетичной форме, а также демонстрировать полученные навыки. Например, в рамках дисциплины "Анализ данных и основы Data Science" портфолио позволяет наглядно представить освоение инструментов анализа данных, включая Python, библиотеки NumPy и Pandas, а также Excel. Это подчёркивает умение студентов применять полученные знания на практике и эффективно работать с данными.

Качественное электронное портфолио должно соответствовать ряду требований. Оно должно быть адаптивным, корректно отображаться на различных устройствах, включая настольные компьютеры, планшеты и смартфоны. Простота обновления контента является важным аспектом, позволяя студентам легко добавлять новые проекты, обновлять достижения и редактировать записи. Эстетическая привлекательность и удобство навигации также играют ключевую роль. Портфолио должно быть оформлено в современном стиле, соответствующем профессиональным стандартам, и обладать интуитивно понятной структурой. Навигация должна быть удобной и логичной, чтобы пользователь мог быстро находить нужную информацию.

Современные технологии, такие как адаптивный дизайн и интерактивные элементы, делают электронное портфолио удобным инструментом не только для обучения, но и для дальнейшего использования. Для дисциплины "Анализ данных и основы Data Science" портфолио может включать примеры визуализаций, результаты анализа данных и описание применённых методов обработки. Это даёт студентам возможность представить себя как компетентных специалистов в своей области.

В рамках данной курсовой работы основное внимание уделено разработке электронного портфолио, использующего современные технологии, включая Next.js и Tailwind CSS. Такое портфолио позволяет создать уникальный стиль, показать качество работы с инструментами анализа данных и заложить основу для дальнейшего развития.

## 1.2 Подходы к созданию электронного портфолио

Создание электронного портфолио требует выбора подхода и инструментов, которые соответствуют целям проекта, профессиональным задачам и уровню подготовки разработчика. Существуют несколько популярных подходов к реализации, каждый из которых обладает своими преимуществами и недостатками.

Одним из наиболее простых и доступных подходов является использование облачных сервисов, таких как Wix, Squarespace или Tilda. Эти платформы позволяют создавать портфолио без навыков программирования, предлагая визуальные редакторы, множество шаблонов и интеграцию с внешними сервисами. Однако такие решения ограничены в возможностях кастомизации и не позволяют реализовать сложные интерактивные элементы или оптимизировать код под индивидуальные задачи.

Другим подходом является использование динамических систем управления контентом (CMS), таких как WordPress или Joomla. CMS-платформы предоставляют гибкий функционал, включая возможность добавления плагинов, настройки дизайна и интеграции с внешними сервисами. Несмотря на удобство, такие решения требуют поддержки серверной инфраструктуры, регулярного обновления ПО и могут быть подвержены угрозам безопасности.

Более современным подходом является создание портфолио с использованием статических генераторов сайтов, таких как Hugo, Jekyll или Gatsby. Эти инструменты позволяют генерировать статические HTML-страницы на основе шаблонов и исходных данных. Подход со статическими генераторами сочетает безопасность, высокую производительность и простоту размещения на платформах, таких как GitHub Pages. Однако он может требовать начальных знаний в программировании и настройке окружения.

Наиболее гибким и профессиональным решением для разработчиков является использование современных фреймворков, таких как Next.js, в сочетании с библиотеками

для стилизации, такими как Tailwind CSS. Этот подход позволяет создавать адаптивные, интерактивные и визуально привлекательные портфолио с уникальным дизайном. Использование Next.js даёт возможность эффективно управлять маршрутизацией, а интеграция Tailwind CSS упрощает процесс создания стилей за счёт использования утилитарных классов. Такой подход подходит для опытных разработчиков, так как требует навыков в работе с JavaScript и фронтенд-разработке.

В рамках данной курсовой работы используется стек технологий, включающий React, Next.js, Tailwind CSS, а также подходы, такие как SPA и SSG. Все эти инструменты дополняют друг друга, создавая основу для создания современных и интерактивных веб-приложений.

### 1.3 Использование React и связанных технологий для создания портфолио

Одностраничные приложения (SPA, Single Page Application) представляют собой сайты, которые не перезагружают страницу при переходе между разделами. Такой подход делает взаимодействие с приложением более плавным и похожим на работу мобильных приложений. Все данные загружаются единожды при открытии приложения, что повышает скорость и удобство использования. SPA — это идеальный выбор для интерфейсов, где важна интерактивность и минимальные задержки.

React.js служит основой для создания SPA благодаря своему компонентному подходу и высокопроизводительной работе с виртуальным DOM. Эта JavaScript-библиотека позволяет легко разрабатывать масштабируемые и интерактивные приложения, поддерживая однонаправленный поток данных для предсказуемого управления состоянием. React также обеспечивает гибкость интеграции с другими инструментами и библиотеками, что делает его универсальным выбором для веб-разработки.

JSX (JavaScript XML) расширяет возможности React, позволяя разработчикам писать разметку, похожую на HTML, прямо в JavaScript-коде. Такой синтаксис делает код более читабельным и упрощает процесс создания пользовательских интерфейсов. Поддержка динамических выражений и интеграция с JavaScript предоставляют гибкость при проектировании сложных интерфейсов. Благодаря этому JSX становится важным элементом в разработке интерфейсов на React.

Фреймворк Next.js расширяет возможности React, предоставляя встроенные инструменты для серверного рендеринга (SSR) и статической генерации страниц (SSG). Это позволяет создавать быстрые, SEO-оптимизированные приложения, подходящие как для SPA, так и для многостраничных решений. Next.js автоматизирует маршрутизацию страниц и предлагает гибридный рендеринг, что повышает производительность и удобство разработки. Он также поддерживает API-роуты, упрощая создание серверных эндпоинтов в рамках одного проекта.

SSG (Static Site Generation) используется в Next.js для предварительной генерации HTML-страниц на этапе сборки. Этот подход обеспечивает высокую производительность и безопасность, так как страницы сохраняются в виде статических файлов и доставляются пользователю напрямую. SSG идеально подходит для контента, который редко меняется, и позволяет эффективно комбинировать его с другими методами рендеринга для более сложных приложений.

Tailwind CSS дополняет стек технологий, предоставляя утилитарные классы для стилизации интерфейсов. Этот CSS-фреймворк предлагает минималистичный и гибкий подход к разработке, позволяя использовать готовые классы прямо в разметке. Tailwind CSS упрощает создание адаптивного дизайна и обеспечивает возможность тонкой настройки через конфигурационный файл. Это делает его идеальным инструментом для разработки современного и стильного портфолио.

Tailwind Typography — это плагин для Tailwind CSS, который позволяет легко создавать стилизованный текстовый контент. Благодаря этому инструменту разработчики могут быстро оформлять заголовки, параграфы, списки, цитаты и другие текстовые элементы, не прибегая к написанию дополнительных CSS-правил. Главной задачей плагина является обеспечение визуальной привлекательности текстового контента при сохранении минимализма и утилитарного подхода, характерного для Tailwind CSS.

Плагин предоставляет утилитарные классы, которые позволяют управлять различными аспектами текстового оформления. Например, с их помощью можно задать размеры шрифта и высоту строки, настроить отступы и поля, определить цвета текста и фона, а также стили для ссылок, списков, таблиц и цитат. Одним из ключевых инструментов плагина является класс `prose`, который применяется для комплексного оформления текстового содержимого. Этот класс контролирует высоту строк, межстрочное расстояние,



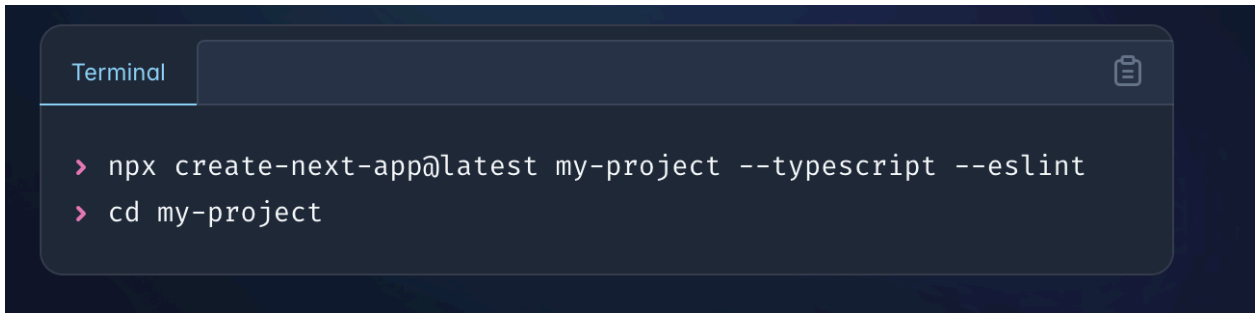
оформление списков, таблиц и изображений, делая текстовый контент более структурированным и визуально привлекательным.

Prettier — это инструмент для автоматического форматирования кода, который используется для обеспечения его читаемости и соответствия стилю проекта. Он автоматически форматирует код на основе предопределённых или настроенных правил, устраняя необходимость ручного редактирования. Prettier поддерживает множество языков программирования, включая JavaScript, TypeScript, CSS и HTML.

Одной из ключевых особенностей Prettier является его интеграция с редакторами кода, такими как VS Code. Это позволяет разработчику автоматически форматировать код при сохранении файла, что экономит время и минимизирует вероятность ошибок, связанных с некорректным форматированием.

### 2.1 Создание Next.js приложения

Для начала работы создадим новый проект с использованием Next.js. Согласно документации на странице Tailwind [1], это можно сделать с помощью команды:

A terminal window with a dark background. The title bar says "Terminal". There are two lines of code entered, each preceded by a red prompt character. The first line is `> npx create-next-app@latest my-project --typescript --eslint` and the second line is `> cd my-project`.

```
Terminal  
  
> npx create-next-app@latest my-project --typescript --eslint  
> cd my-project
```

Рисунок 1. npx команды для инициализации next.js проекта

Эта команда создаёт структуру проекта и автоматически устанавливает необходимые зависимости, такие как TypeScript и ESLint, если они указаны. После выполнения команды нужно перейти в папку с созданным проектом.

Далее необходимо подключить Tailwind CSS и его зависимости. Для этого используется следующая команда:

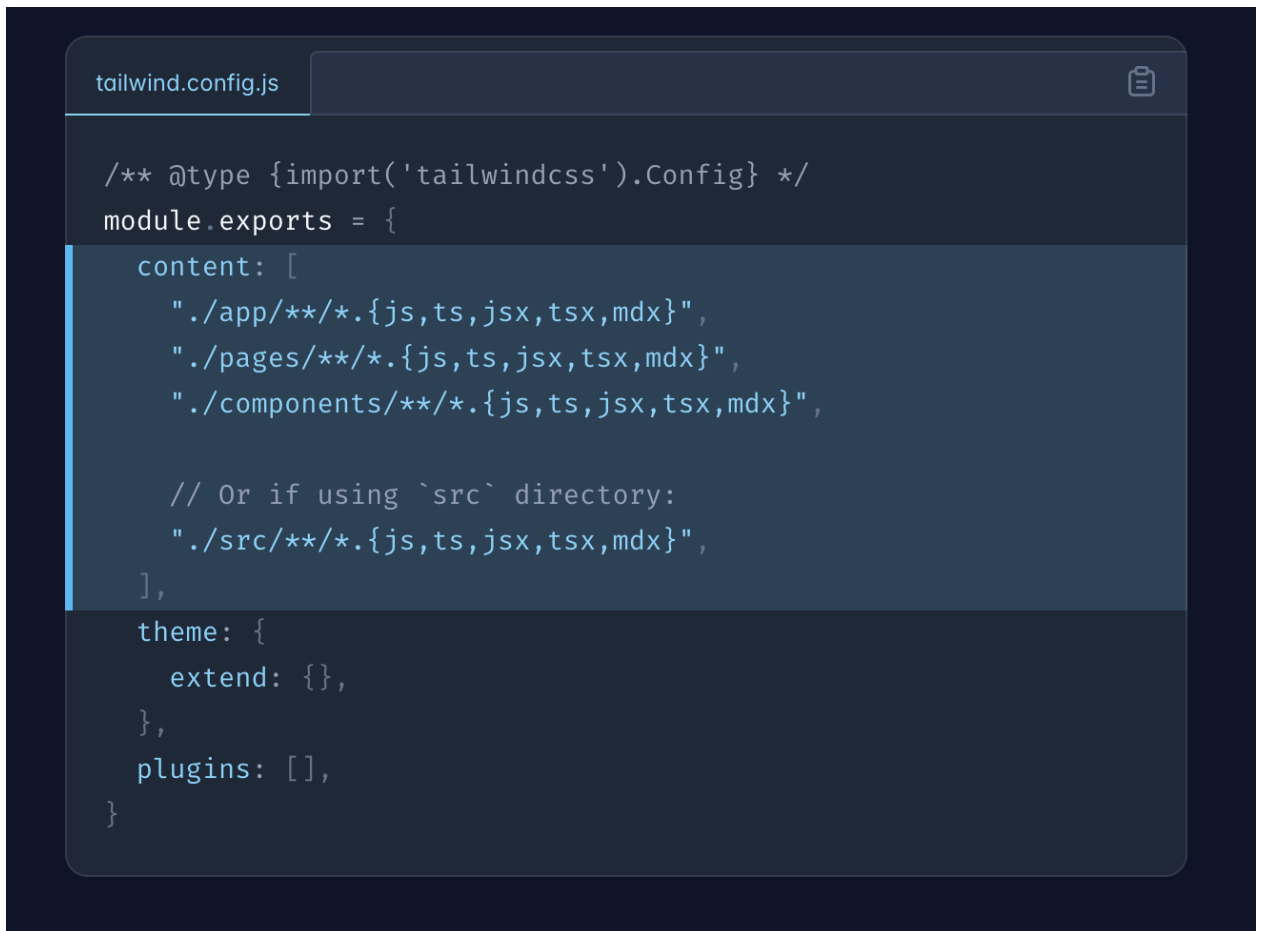
A terminal window with a dark background. The title bar says "Terminal". There are two lines of code entered, each preceded by a red prompt character. The first line is `> npm install -D tailwindcss postcss autoprefixer` and the second line is `> npx tailwindcss init -p`.

```
Terminal  
  
> npm install -D tailwindcss postcss autoprefixer  
> npx tailwindcss init -p
```

Рисунок 2. “npx” команды для загрузки и инициализации Tailwind

Первая команда устанавливает Tailwind CSS и связанные с ним инструменты, такие как PostCSS и Autoprefixer. Вторая команда создаёт конфигурационные файлы `tailwind.config.js` и `postcss.config.js`, которые необходимы для настройки и работы Tailwind CSS в проекте.

После установки Tailwind CSS следующим шагом является настройка конфигурационного файла `tailwind.config.js`. В нём необходимо указать пути ко всем файлам проекта, где предполагается использовать классы Tailwind.



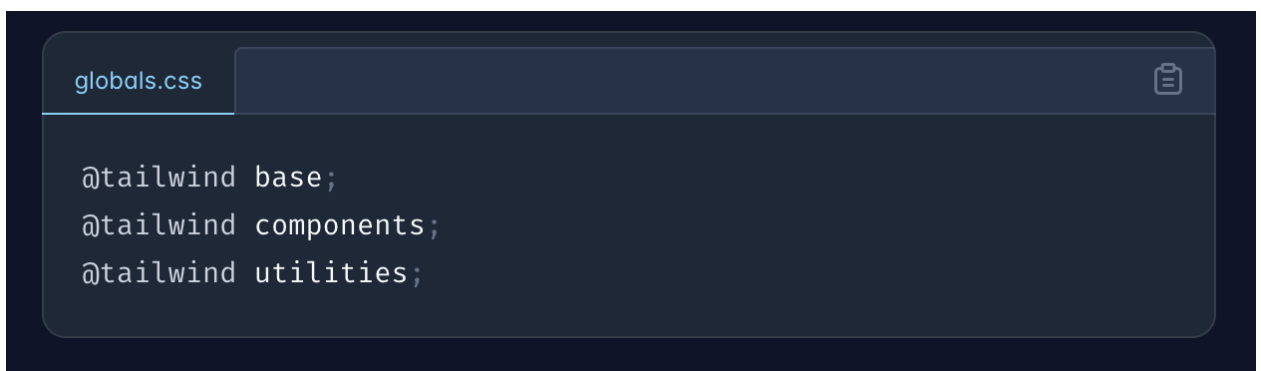
```
tailwind.config.js

/** @type {import('tailwindcss').Config} */
module.exports = {
  content: [
    './app/**/*.{js,ts,jsx,tsx,mdx}',
    './pages/**/*.{js,ts,jsx,tsx,mdx}',
    './components/**/*.{js,ts,jsx,tsx,mdx}',

    // Or if using `src` directory:
    './src/**/*.{js,ts,jsx,tsx,mdx}',
  ],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

Рисунок 3. Конфигурация Tailwind

Далее необходимо подключить основные директивы Tailwind CSS в глобальном CSS-файле `globals.css`. Для этого добавляем следующие строки:



```
globals.css

@tailwind base;
@tailwind components;
@tailwind utilities;
```

Рисунок 4. Настройки по умолчанию для глобальных CSS стилей

Эти директивы обеспечивают подключение базовых стилей, компонентов и утилитарных классов, необходимых для работы Tailwind CSS.

После завершения настройки проекта запускаем его с помощью команды:

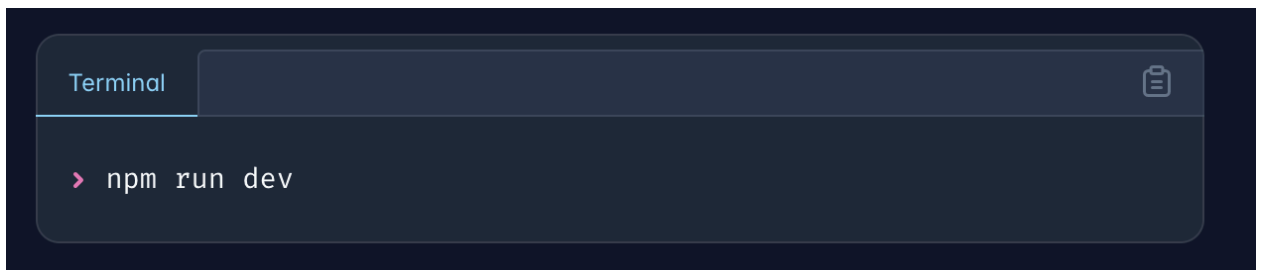


Рисунок 5. npm команда для запуска проекта в режиме разработки

## 2.2 Конфигурация Next js приложения

В конфигурационном файле `next.config.ts` я, согласно официальной документации Next js [2], задал параметры, необходимые для корректной работы статической генерации страниц (SSG) и последующего размещения сайта на GitHub Pages.

Параметр `output: 'export'`:

Этот параметр указывает, что приложение должно быть экспортировано как статический сайт. Все страницы приложения будут заранее сгенерированы в виде статических HTML-файлов. Это важно для SSG, так как GitHub Pages поддерживает только статические сайты, и данный параметр позволяет подготовить проект к такому формату.

Раздел `images`:

В этом разделе настроены следующие параметры:

`path: ''`: Указан пустой базовый путь для изображений. Это позволяет использовать стандартное поведение и упрощает настройку ссылок на изображения при размещении на GitHub Pages.

`unoptimized: true`: Отключена встроенная оптимизация изображений Next.js. Это необходимо, так как GitHub Pages не предоставляет серверной среды для выполнения обработки изображений. В результате изображения используются в исходном виде, что обеспечивает совместимость с платформой.

## 2.3 Конфигурация Tailwind

В конфигурационном файле `tailwind.config.ts` я согласно документации[3], прописал кастомные цвета, которые можно использовать в проекте для стилизации элементов. Например, для основного фона добавлен цвет `background`, для градиентов заголовков первого уровня заданы цвета `h1-color-from` и `h1-color-to`, а для заголовков второго уровня – цвет `h2-color`. Также определены цвета для ссылок (`link-color`), выделенного текста (`highlighted-text`), блоков важной информации (`aside-color`) и кнопок (`button-color`, `extra-button-color`).

Эти цвета позволяют использовать их в коде через утилитарные классы Tailwind.

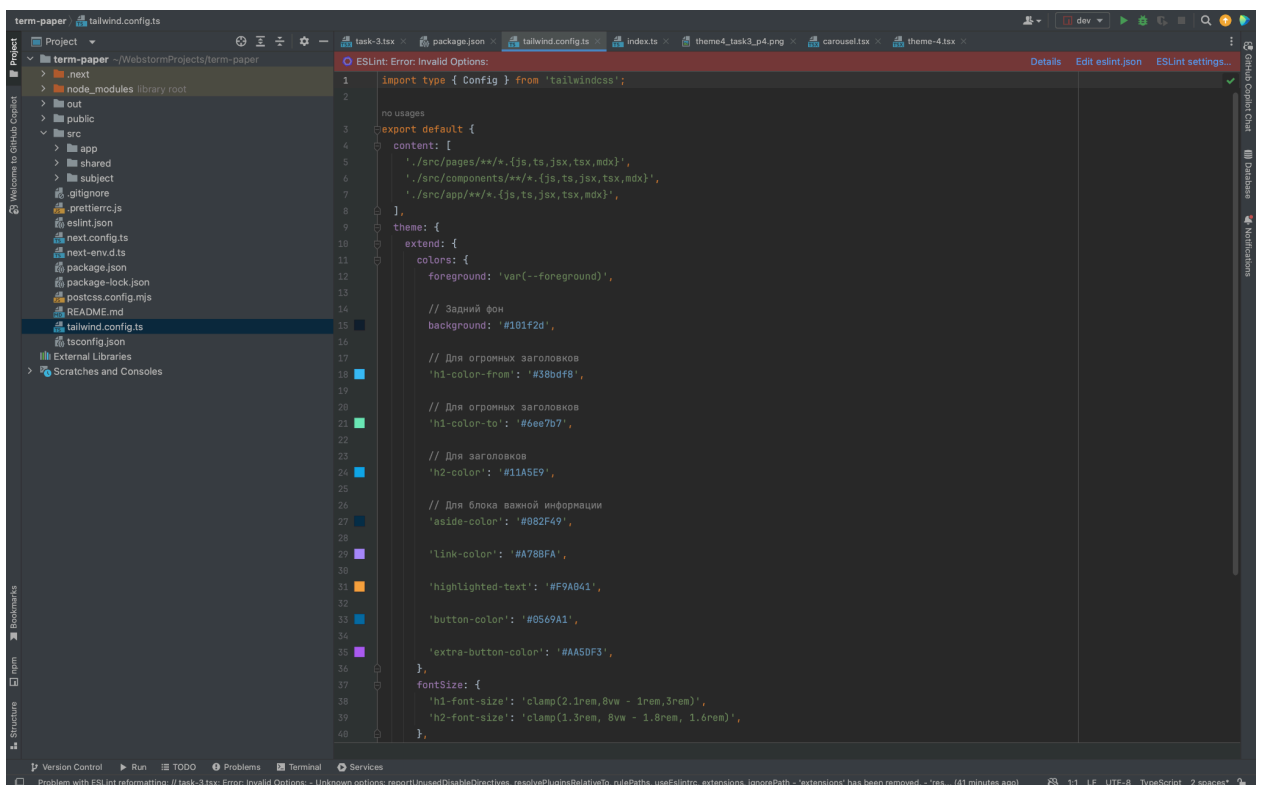


Рисунок 6. Конфигурационный файл Tailwind после добавления стилей для дизайн системы портфолио

## 2.4 Глобальная настройка CSS

После настройки цветов в `tailwind.config.ts` я перешёл к файлу `globals.css`, чтобы согласно документации, [4] задать базовые стили для всего приложения.

Для подключения шрифтов я использовал импорт с Google Fonts, добавив шрифт Roboto с разными вариациями начертания.

В разделе @layer base я переопределил стили для базовых элементов. Например, для элемента <body> установил общий фон с использованием класса bg-background, стили типографики prose, а также кастомный шрифт font-apple, который был добавлен ранее в конфигурацию Tailwind.

Для заголовков <h1> и <h2> я задал индивидуальные стили. <h1> оформлен с использованием градиента, который создаётся с помощью bg-gradient-to-r, from-h1-color-from и to-h1-color-to. <h2> получил стили с кастомным цветом text-h2-color и размером шрифта text-h2-font-size.

Для блоков <aside> я применил цвет bg-aside-color, добавил отступы и границу с цветом h1-color-from.

Также добавил стили для скролла в общей цветовой гамме.

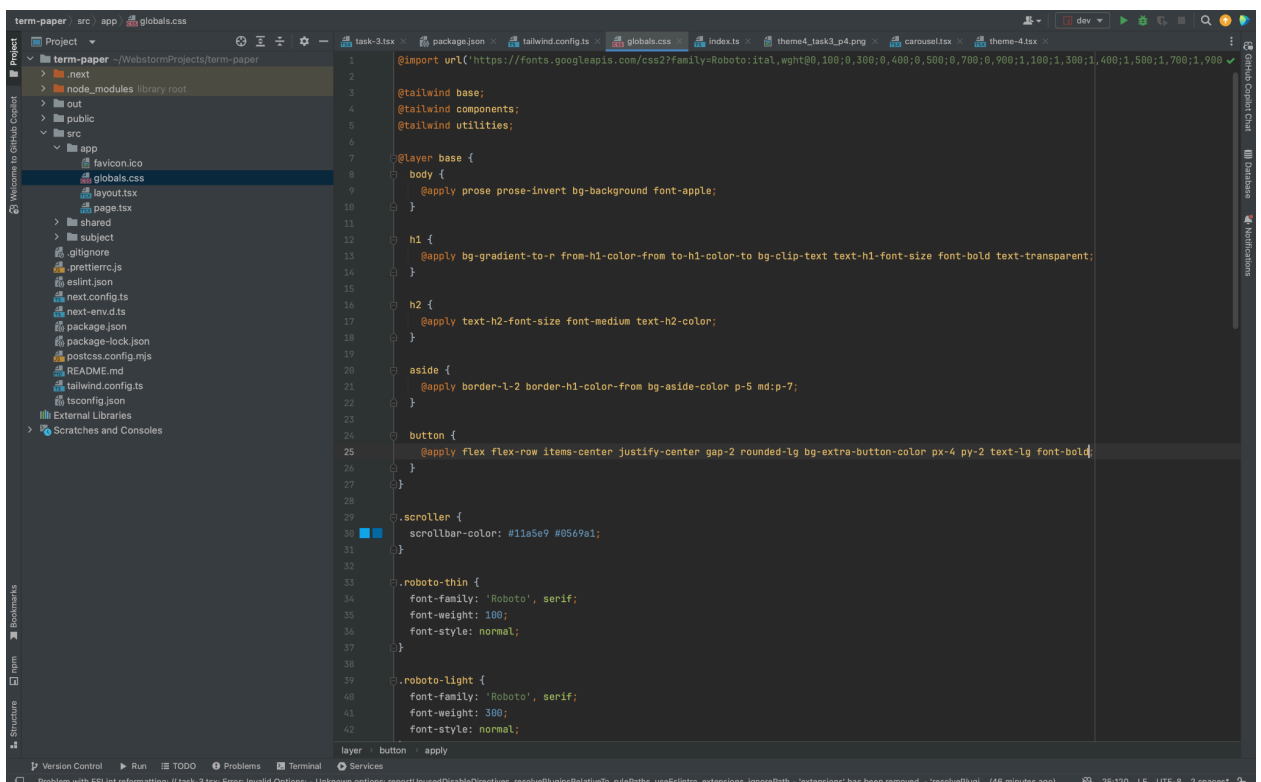


Рисунок 7. Глобальное переопределение стилей для базовых HTML элементов

## 2.5 Главная страница

На данном этапе я создал основной компонент страницы, который представляет собой файл `page.tsx`. Этот компонент отвечает за отображение главной страницы веб-портфолио и включает базовую структуру интерфейса.

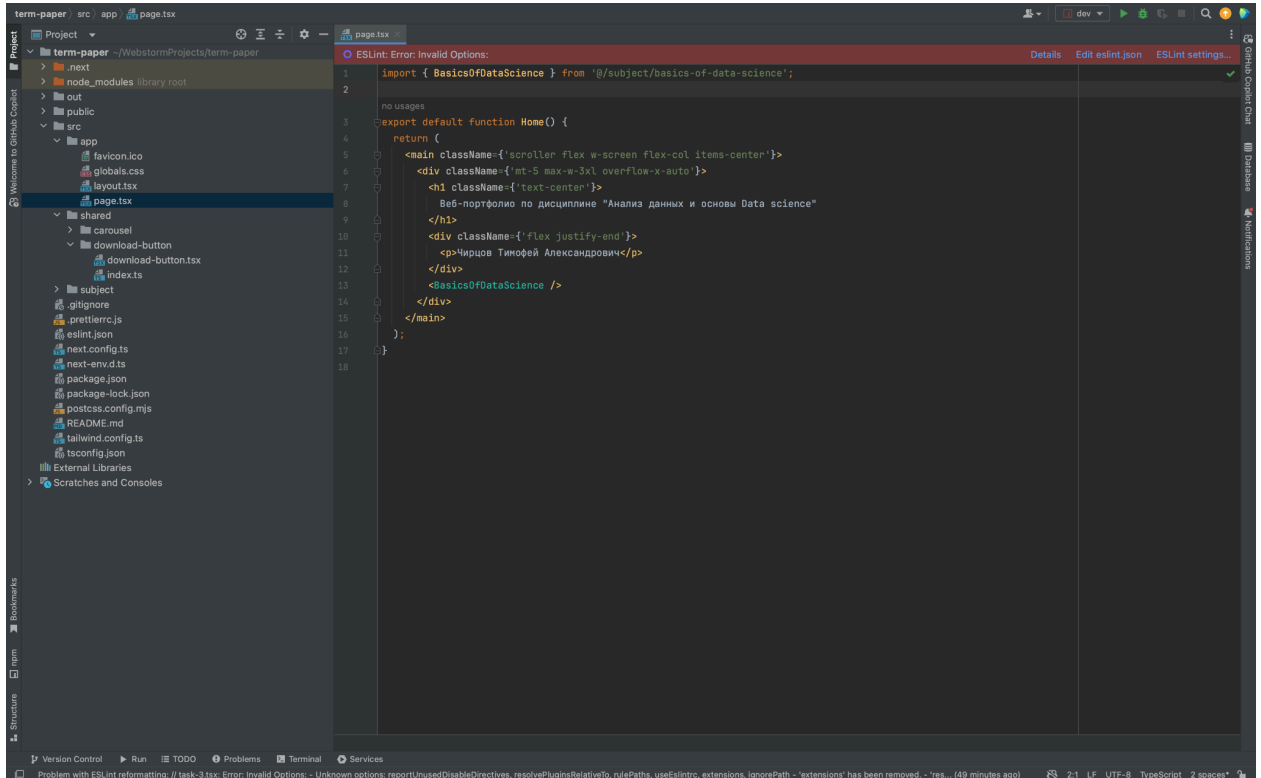


Рисунок 8. React компонент с заголовком страницы портфолио

## 2.6 Пример страницы для задания

Далее я создал страницу для отображения конкретного задания, которое представлено в виде компонента `Task1`.

На странице я разместил описание задания, используя теги `<strong>` для выделения заголовка задания и `<aside>` для отображения пояснительного текста. Добавил компонент `Carousel`, который позволяет пользователю просматривать изображения, связанные с заданием.

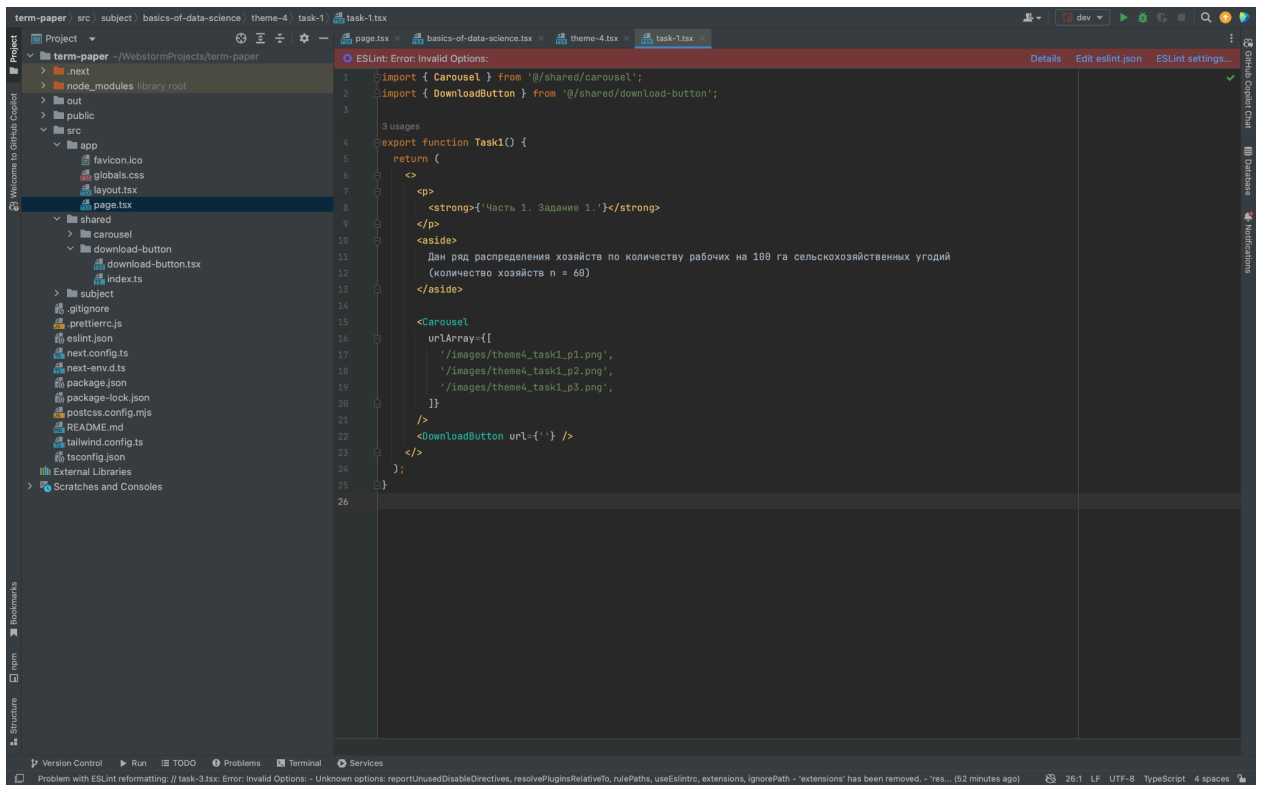


Рисунок 9. React компонент с примером заполнения контента для одного из заданий

## 2.7 Получившаяся страница портфолио

После выполнения всех предыдущих шагов получившееся портфолио выглядит так:



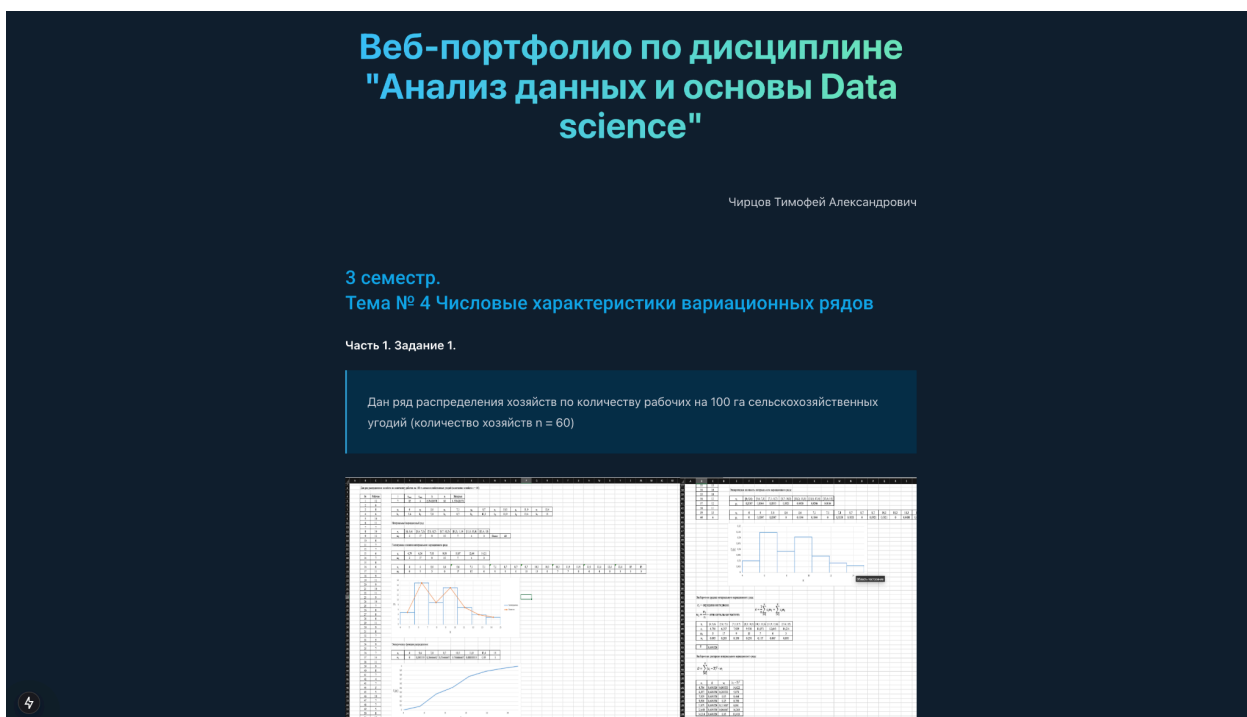


Рисунок 10. Скриншот страницы портфолио, развернутого локально

## 2.8 Развертка приложения на Github Pages

Следующим шагом стала развертка приложения на хостинге Github Pages. Для этого я установил утилиту gh-pages, которая используется для развертывания статических файлов на GitHub Pages

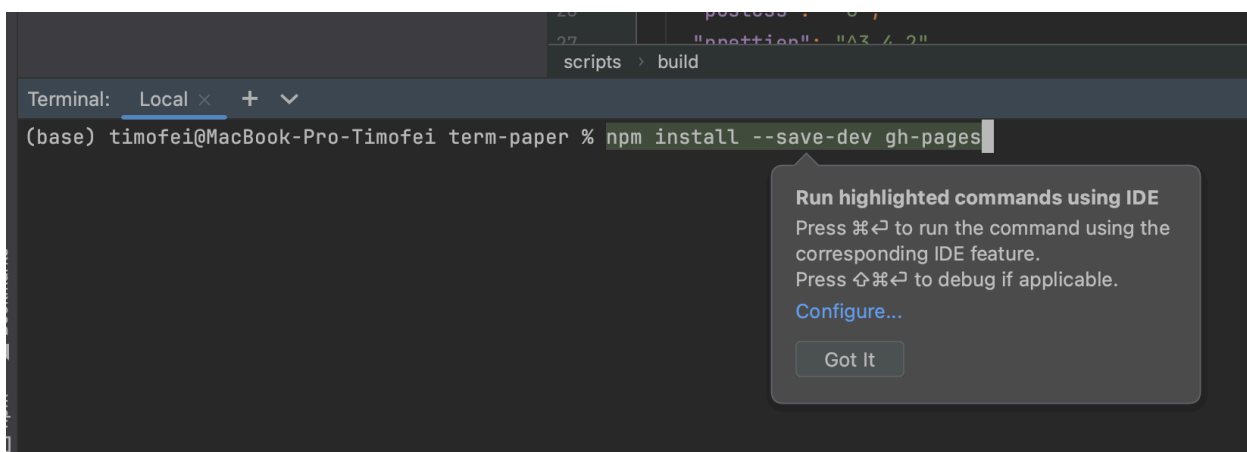
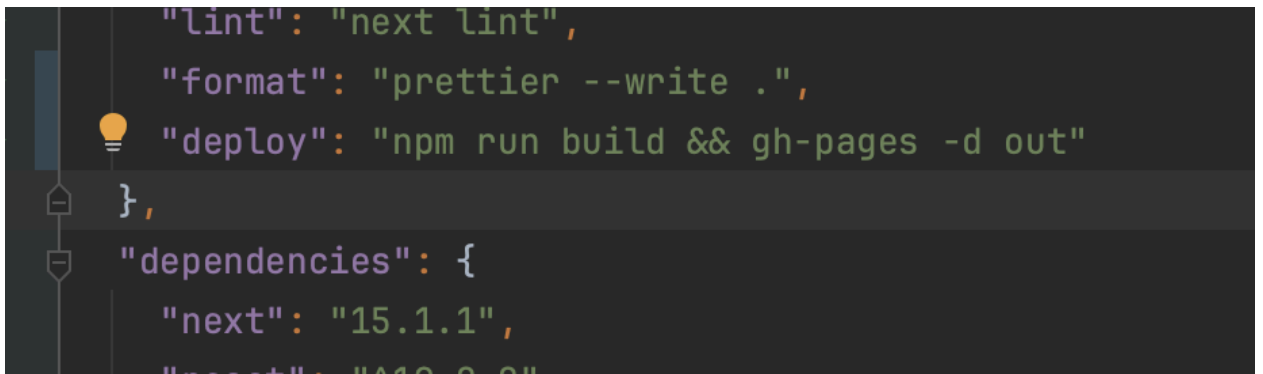


Рисунок 11. Процесс установки библиотеки gh-pages

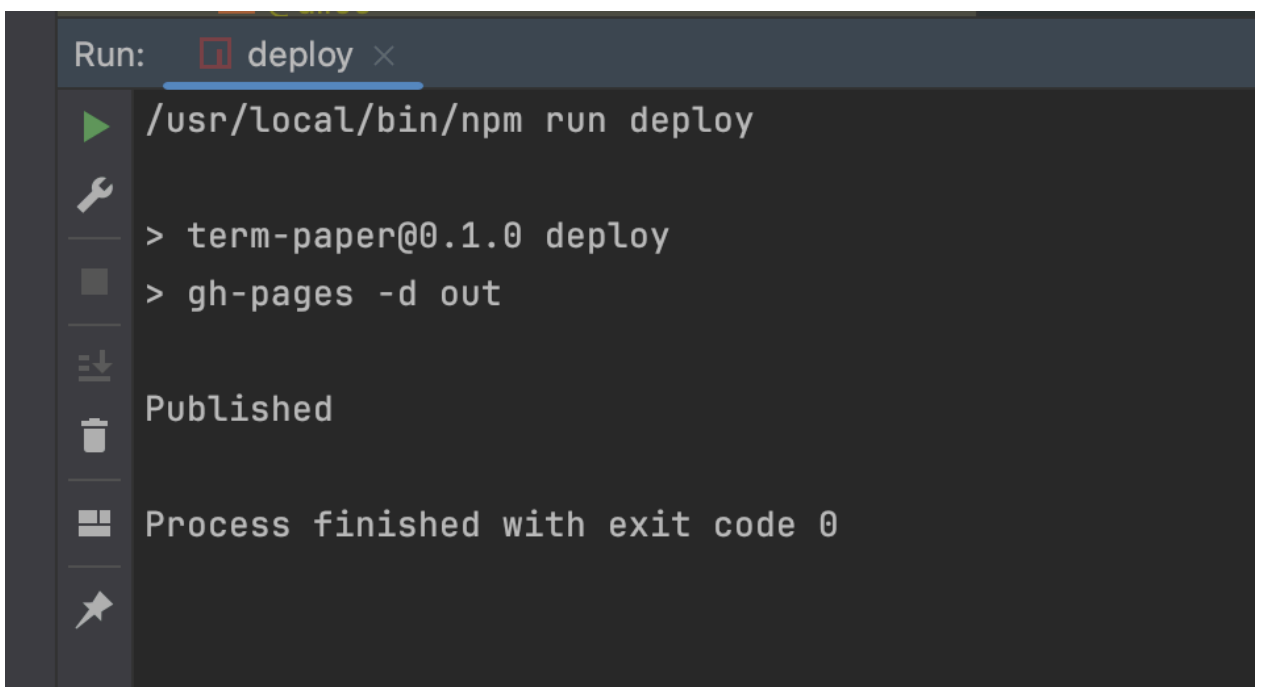
Затем я добавил команду для деплоя проекта в секцию scripts файла package.json



```
"lint": "next lint",  
"format": "prettier --write .",  
"deploy": "npm run build && gh-pages -d out",  
},  
"dependencies": {  
  "next": "15.1.1",  
  "prettier": "3.1.0",  
}
```

Рисунок 12. Команда для сборки и деплоя проект в GitHub Pages

После этого я выполнил команду `npm run deploy`, которая успешно собрала приложение и опубликовала его на GitHub Pages.



```
Run: ❌ deploy ×  
▶ /usr/local/bin/npm run deploy  
> term-paper@0.1.0 deploy  
> gh-pages -d out  
  
Published  
  
Process finished with exit code 0
```

Рисунок 13. Успешно произведенный деплой в GitHub Pages

После этого я перешёл в настройки репозитория на GitHub и выбрал правильную ветку и папку для развертывания сайта через GitHub Pages. В настройках в разделе Build and deployment я указал ветку `gh-pages` и корневую директорию (`/root`) для публикации статических файлов.

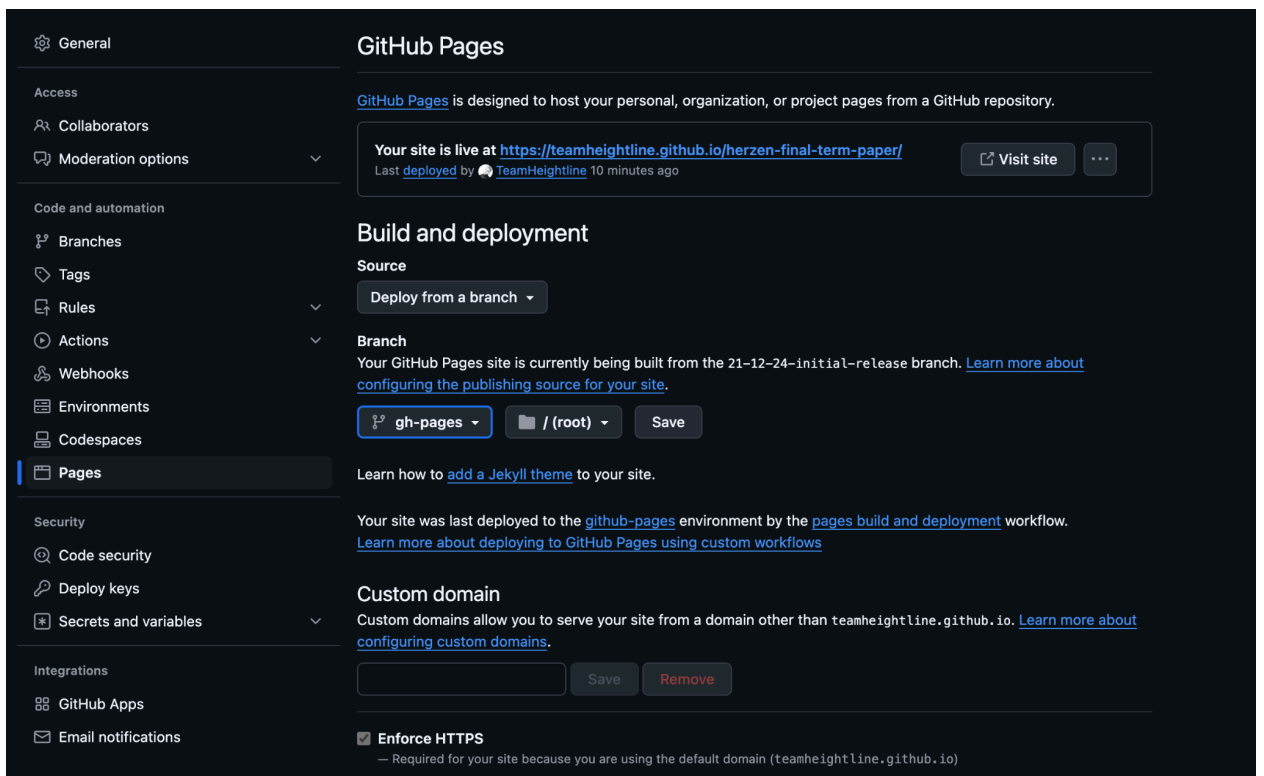


Рисунок 14. Настройка развертки статического сайта на GitHub Pages из git ветки gh-pages

После выполнения всех описанных шагов и добавления навбара и бокового меню навигации, проект был успешно развернут на платформе GitHub Pages. Веб-портфолио теперь доступно по адресу: <https://teamheightline.github.io/herzen-final-term-paper/>.

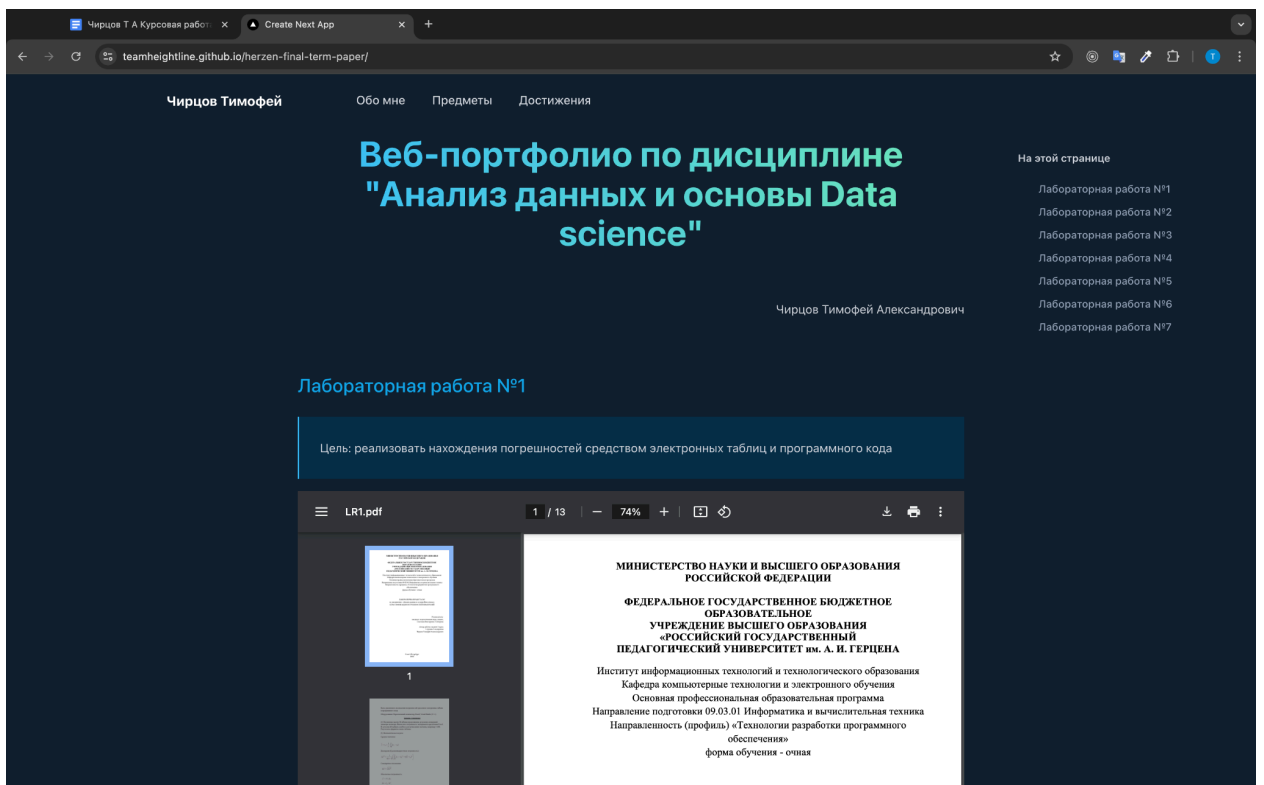


Рисунок 15. Успешно развернутый проект портфолио на хостинге Github Pages

Таким образом в ходе выполнения практической части было с нуля разработано веб-приложение для портфолио, были произведены все этапы от создания проекта при помощи create next app, стилизации при помощи конфига tailwind до окончательной сборки главной страницы с заданиями по предмету анализ данных и основы Data science

## ЗАКЛЮЧЕНИЕ

В рамках выполнения данной курсовой работы была достигнута основная цель — разработка электронного веб-портфолио по дисциплине "Анализ данных и основы Data Science"

В теоретической части был проведён анализ понятия электронного портфолио, его значимости для образовательной и профессиональной среды. Особое внимание было уделено актуальности использования современных технологий и подходов, таких как SPA, React, Next.js и Tailwind CSS, которые позволяют создавать адаптивные и интерактивные интерфейсы. Были также изучены платформы и подходы к размещению портфолио, что помогло обосновать выбор GitHub Pages для публикации проекта.

Практическая часть работы включала проектирование структуры портфолио, разработку разделов для размещения материалов, выполнение настройки конфигурации и публикации проекта на платформе GitHub Pages с использованием Next.js и gh-pages.

Подводя итоги, можно заключить, что все поставленные задачи были успешно выполнены, а цели курсовой работы достигнуты.

## ЛИТЕРАТУРА

1. Get started with Tailwind CSS // Официальная документация Tailwind CSS URL:  
<https://tailwindcss.com/docs/installation> (дата обращения: 26.12.24).
2. Static Site Generation (SSG) // Официальная документация Next js URL:  
<https://nextjs.org/docs/pages/building-your-application/rendering/static-site-generation>  
(дата обращения: 26.12.24).
3. Configuration // Официальная документация Tailwind URL:  
<https://tailwindcss.com/docs/configuration> (дата обращения: 26.12.24).
4. Functions & Directives // Официальная документация Tailwind URL:  
<https://tailwindcss.com/docs/functions-and-directives> (дата обращения: 26.12.24).

## ПРИЛОЖЕНИЕ

Ссылка на электронное портфолио - <https://teamheightline.github.io/herzen-final-term-paper/>

Ссылка на git репозиторий - <https://github.com/TeamHeightline/herzen-final-term-paper>