

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего
образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных
систем

(код и наименование направления подготовки, специальности)

Мобильные и сетевые технологии

(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Применение технологии для разработки Next.js веб - приложения по
размещению результатов студенческих исследований в области внутренней
баллистики.»

Студент

А.А. Щекин

(И.О. Фамилия)

(личная подпись)

Руководитель

А.И. Сафронов

(ученая степень, звание, И.О. Фамилия)

Консультант

И.Ю. Усатова

(ученая степень, звание, И.О. Фамилия)

Тольятти 2022

Аннотация

Выпускная квалификационная работа на тему: «Применение технологии для разработки Next.js веб - приложения по размещению результатов студенческих исследований в области внутренней баллистики».

Ключевые слова: веб-приложение, веб-разработка, программирование, программное обеспечение, внутренняя баллистика.

Объём работы - 52 страницы, на которых представлены 60 рисунков. При написании работы использовалось 20 источников.

Объект исследования: сервисы публикации выпускных работ студентов.

Предмет исследования: веб-приложения публикующие выпускные работы студентов.

Цель работы: разработать веб-приложение по размещению результатов студенческих исследований в области внутренней баллистики.

Практическая значимость работы заключается в том, что разработанное веб-приложение позволяет реализовать более полное ознакомление потенциальных заказчиков с работами студентов по теме внутренней баллистики.

Структурно работа состоит из трех глав. Первая – задачи исследования, вторая – определение технологий и разработка веб-приложения, третья – описание разработанного веб-приложения.

Abstract

Final qualifying work on the topic: «Application of technology for development Next.js web application for posting the results of student research in the field of internal ballistics».

Keywords: web application, web development, programming, software, internal ballistics.

The volume of work is 52 pages, which contain 60 drawings. When writing the work, 20 sources were used.

Object of study: services for publishing students' graduation papers.

Subject of research: web applications publishing students' graduation papers.

The purpose of the work: to develop a web application for posting the results of student research in the field of internal ballistics.

The practical significance of the work lies in the fact that the developed web application allows for a more complete acquaintance of potential customers with the work of students on the topic of internal ballistics.

Structurally, the work consists of three chapters. The first is the research objectives, the second is the definition of technologies and the development of a web application, the third is a description of the developed web application.

Содержание

Введение	6
1 Задачи исследования	7
1.1 Характеристика предметной области	7
1.2 Формулировка задачи	8
2 Определение технологий и разработка веб-приложения	9
2.1 Выбор и настройка среды разработки	9
2.1.1 Определение среды разработки	9
2.1.2 Выбор среды разработки	9
2.1.3 Настройка среды разработки	13
2.2 Выбор языков разработки	15
2.2.1 Выбор фреймворка для разработки	15
2.2.2 Определение jsx	16
2.2.3 Определение Next.js технологии	18
2.3 Разработка веб-приложения с применением Next.js технологии	19
2.3.1 Components	26
2.3.2 SSR(server side rendering)	28
2.3.3 Навигация по приложению с помощью Link	30
2.3.4 Image	30
2.4 Настройка взаимодействия с поисковыми роботами	30
2.5 Публикация сайта в сети-интернет (сайт + гитхаб)	33
2.5.1 Создание репозитория на GitHub	33
2.5.2 Деплой проекта	34
3 Описание разработанного веб-приложения	36
3.1 Описание интерфейса пользователя	36
3.2 Описание основного функционала веб-приложения	38
3.3 Адаптивность и логика приложения	40
3.3.1 Адаптивность	40
3.3.2 Логика приложения	48

Заключение	50
Список используемой литературы	51

Введение

Разработка веб-приложения для демонстрации материалов дипломных работ на тему внутренней баллистики является актуальной. Исследования проведенные в этой сфере профессором Сафроновым А.И. и опубликованные работы под его руководством пользуются спросом. Поэтому было необходимо разработать веб-приложение, в котором будут представлены, как и основные проекты Александра Ивановича Сафронова, так и проекты под его руководством. В будущем это веб-приложение будет постоянно пополняться новыми работами. Для расширения проекта, добавления в него новых материалов, выпускных работ студентов, работ Александра Ивановича Сафронова, необходим дополнительный подход к разработке приложения. В данный подход входит реализация добавления бэкенд составляющей в проект кода. Поэтому важно использовать инструменты, которые позволят наполнять контентом ресурс, без дополнительных изменений кода, сохраняя его функционал и быстроту загрузок. Фреймворк Next.js лучше всего подходит для этих целей.

Дизайн проекта включает в себя элементы минимализма. Минимализм подразумевает выдвигание основного текста и изображений на первый план. Принципиально показать главную информацию о дипломах студентов без лишних визуальных эффектов. Этот подход позволяет комплексно подходить к изучению проектов, представленных в веб-приложении.

В настоящее время выполнение больших задач принимают участие коллективы исследователей. На примере Тольяттинский государственный университета такой работой является, задача баллистической проработки систем эстафетной схемы. Данная работа позволяет потенциальным заказчикам ознакомиться с продуктами разработанными студентами по кафедральной теме внутренней баллистики. И данное веб-приложение поможет им досконально изучить выпускные квалификационные работы студентов.

1 Задачи исследования

1.1 Характеристика предметной области

Все что мы видим в интернете от одностраничного сайта до платформы, которая создает эти одностраничные сайты, называется - web-приложение. Web-разработка делится на backend и frontend. Что же это такое, давайте разбираться по подробней.

Frontend - это всё, что выполняется прямо в браузере. А тех, кто занимается фронтэндом называют frontend-разработчики или просто фронтендеры. У фронтендера есть 3 главных инструмента.

Первое - это Html. Это язык разметки, который помогает браузеру понять, как и где расположить блоки, текст, картинки и т.д. на странице. Например, вверху мы располагаем шапку сайта, по центру мы располагаем каталог вещей, а внизу ссылки на соцсети.

Второе - это Css или просто стили. Это такой код, который помогает делать ваш сайт красивее. Благодаря Css, например, мы можем сделать кнопку красной или зеленой. Или сделать окошки внутри странички с закругленными углами. Или даже поиграться со шрифтами.

Третье - это JavaScript. Либо другие аналогичные скрипты, которые обрабатывают действия пользователя на сайте. Например, при регистрации вы вводите свой пароль. И у вас появляется окно, что у вас не хватает каких-то символов или цифр. Это frontend-разработчик обработал ваши действия и написал вам, что у вас слишком простой пароль (это называется валидация).

Backend - это код, который выполняется на сервере там, где у пользователя нет доступа. Именно здесь проверяется ваш логин и пароль, здесь из базы данных подгружаются сообщения и именно здесь производятся оплаты. По аналогии с фронтэндом, тех, кто занимается бэкендом называются backend-разработчики, или просто бэкендеры. Всё общение между frontend и backend происходит с помощью http-запросов (http - это протокол для передачи

данных). Эта передача данных происходит из браузера на сервер и обратно. И всё это происходит незаметно для пользователя.

1.2 Формулировка задачи

Предмет исследования – веб-приложение по размещению результатов студенческих исследований с помощью технологии Next.js.

Цель работы – разработать вычислительное приложение на Next.js.

Для достижения поставленной цели необходимо решить ряд задач:

- подобрать необходимые инструменты разработки;
- изучить технологии включенные в Next.js;
- разработать веб-приложение;
- опубликовать веб-приложение в сеть интернет;

Вывод по разделу 1

В результате данной главы, были сформулированы характеристика предметной области и задачи необходимые для выполнения работы.

2 Определение технологий и разработка веб-приложения

2.1 Выбор и настройка среды разработки

2.1.1 Определение среды разработки

Среда разработки (IDE), также единая среда разработки, ECP – комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО) [2]. Среда разработки включает в себя:

- текстовый редактор;
- транслятор (компилятор и/или интерпретатор);
- средства автоматизации сборки;
- отладчик.

2.1.2 Выбор среды разработки

Для того чтобы начать программирование в IDE, необходимо сравнить и выбрать наиболее подходящую под выбранную задачу среду. Далее перечислены основные IDE для веб-разработки:

а) Brackets - благодаря специализированным визуальным инструментам и поддержке препроцессора Brackets - это современный текстовый редактор, который упрощает разработку в браузере (рисунок 1). Он создан с нуля для веб-дизайнеров и разработчиков интерфейса [5].

1) плюсы Brackets:

- встроенные редакторы;
- предварительный просмотр веб-страниц в реальном времени;
- комплексная поддержка препроцессора.

2) минусы Brackets:

- плохая поддержка разработчиками;
- мало обновлений;
- мало функционала;
- медленная работа приложения.

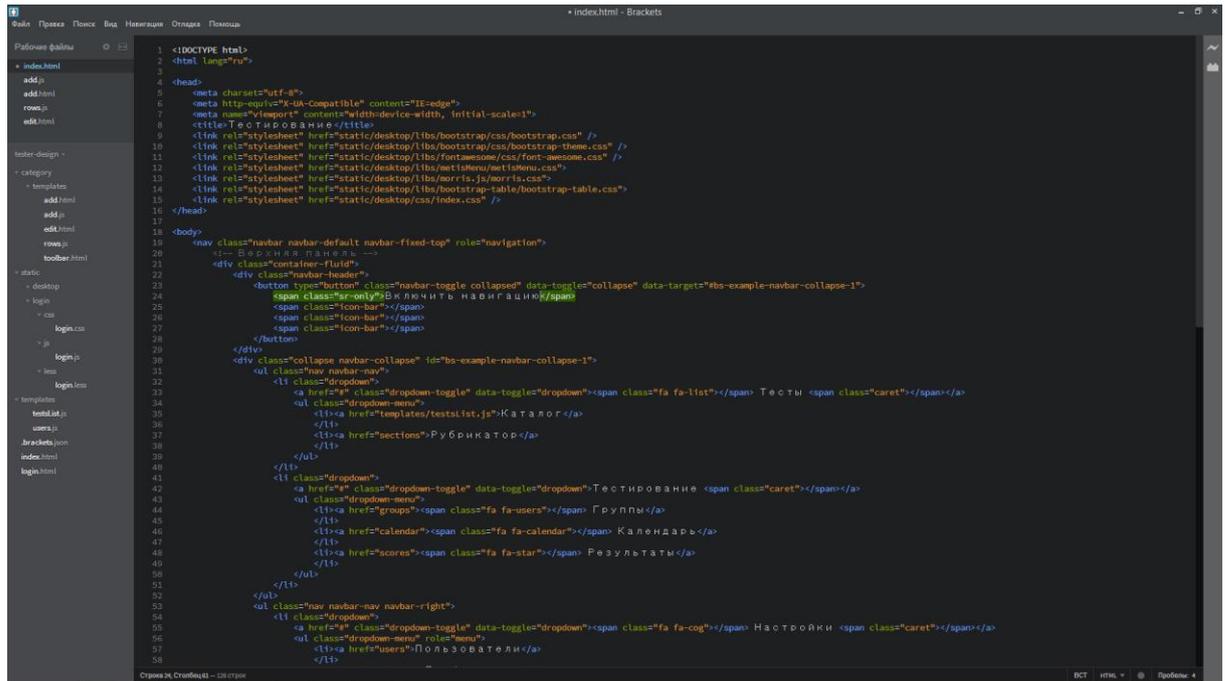


Рисунок 1 – Интерфейс редактора Brackets

б) Atom – бесплатный текстовый редактор с открытым исходным кодом для Linux, macOS, Windows с поддержкой плагинов, написанных на JavaScript (рисунок 2), и встраиваемых под управлением Git [9].

1) плюсы Atom:

- создан и поддерживается разработчиками GitHub;
- встроенный контроль версий git.

2) минусы Atom:

- ограниченный функционал;
- неинтуитивный интерфейс.

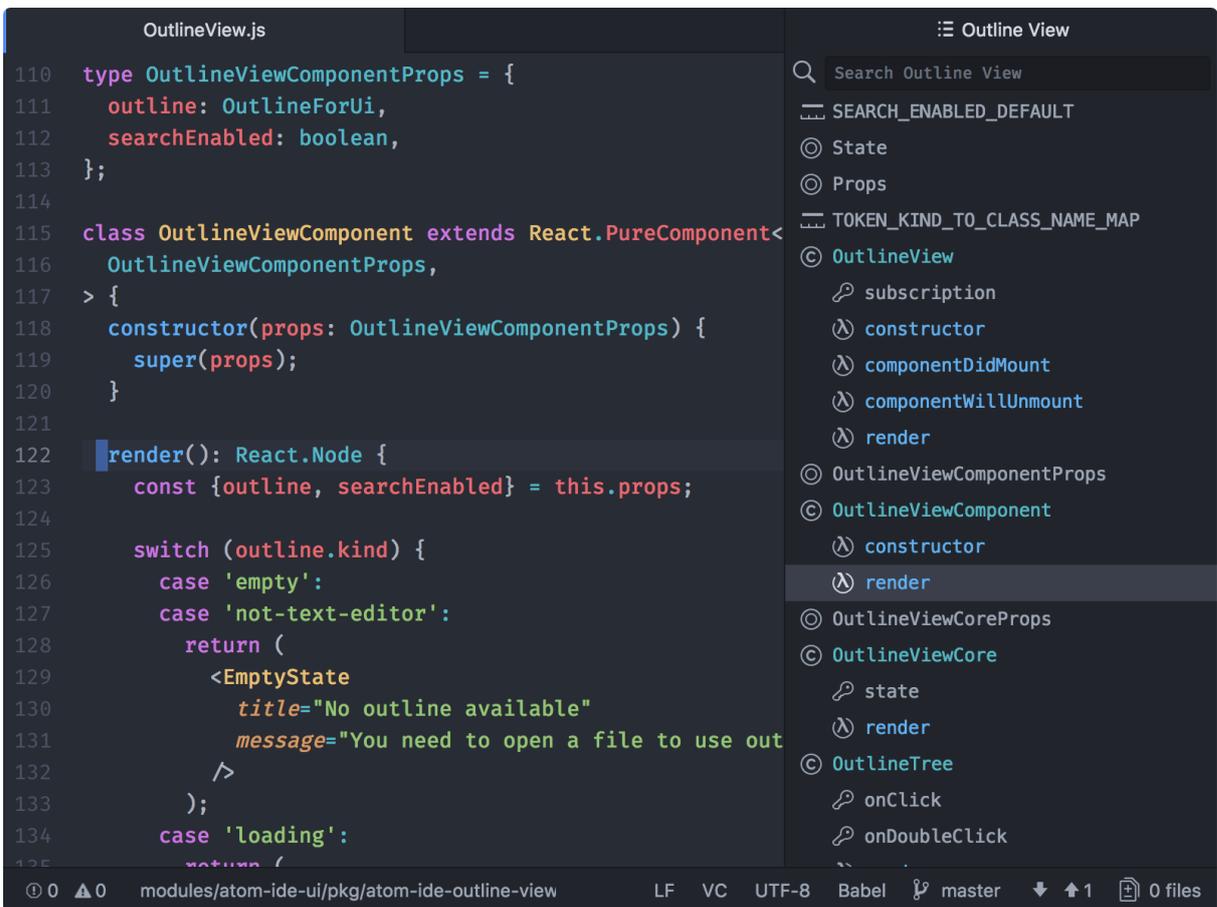


Рисунок 2 – Интерфейс редактора Atom

в) Sublime Text – проприетарный текстовый редактор (рисунок 3). [10]

1) плюсы Sublime Text:

- легкий и быстрый редактор;
- удобные горячие клавиши.

2) минусы Sublime Text:

- подходит в основном только для верстки;
- проприетарность.

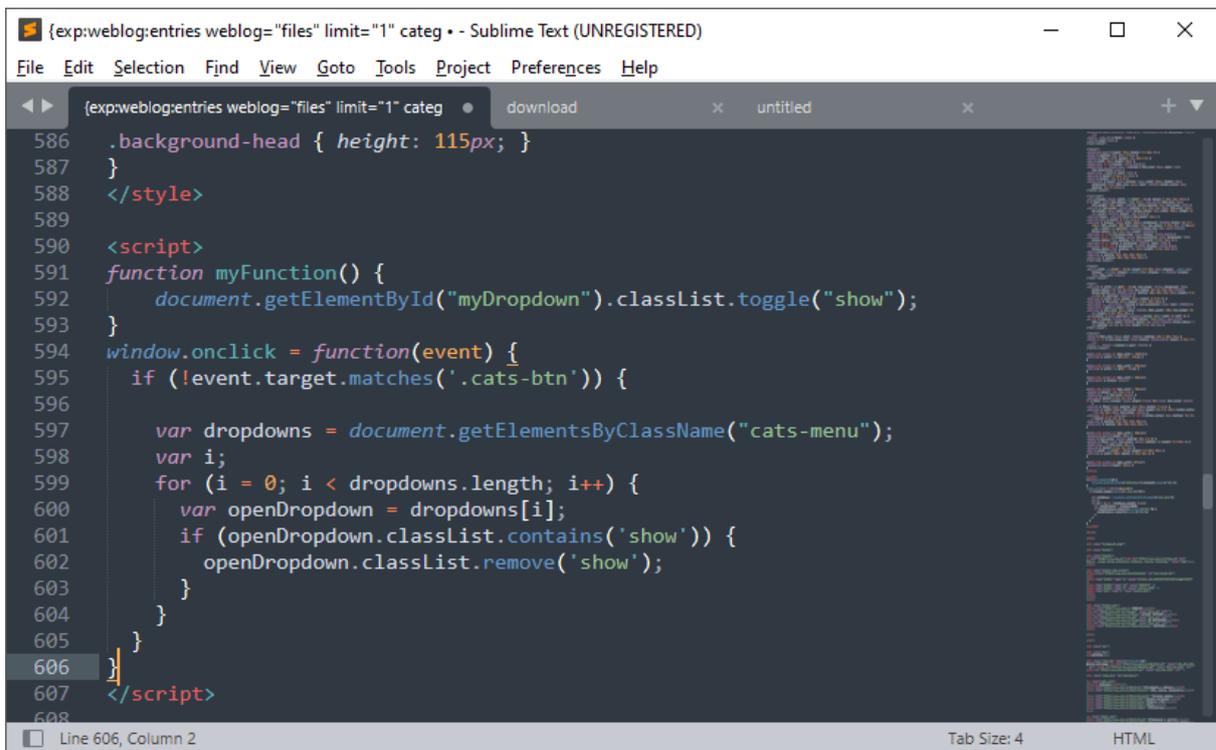


Рисунок 3 – Интерфейс редактора Sublime Text

г) Visual Studio Code – бесплатный. Построен на открытом исходном коде (рисунок 4). Запускается везде [14].

1) плюсы Visual Studio Code:

- кастомизация;
- быстрота;
- поддержка разработчиками Microsoft;
- огромная библиотека расширений.

2) минусы Visual Studio Code:

- большое количество инструментов, которые не нужны большинству разработчиков.

Мой выбор пал на Visual Studio Code, так как по соотношению плюсов и минусов он самый подходящий.

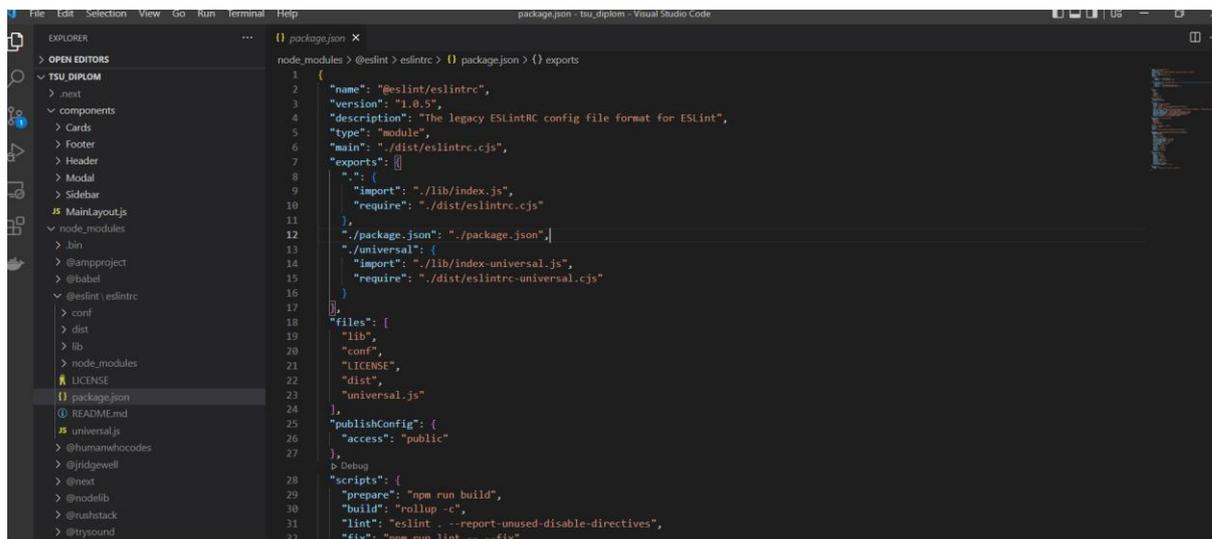


Рисунок 4 – Интерфейс редактора Visual Studio Code

2.1.3 Настройка среды разработки

В первую очередь скачиваю и устанавливаю IDE с официального сайта: <https://code.visualstudio.com/> . Далее открываю вкладку с расширениями и устанавливаю такие расширения:

– Auto Import – автоматически находит, анализирует и предоставляет кодовые действия и завершение кода для всех доступных импортов [7]. Это расширение понадобится для быстрого импорта (рисунок 5);

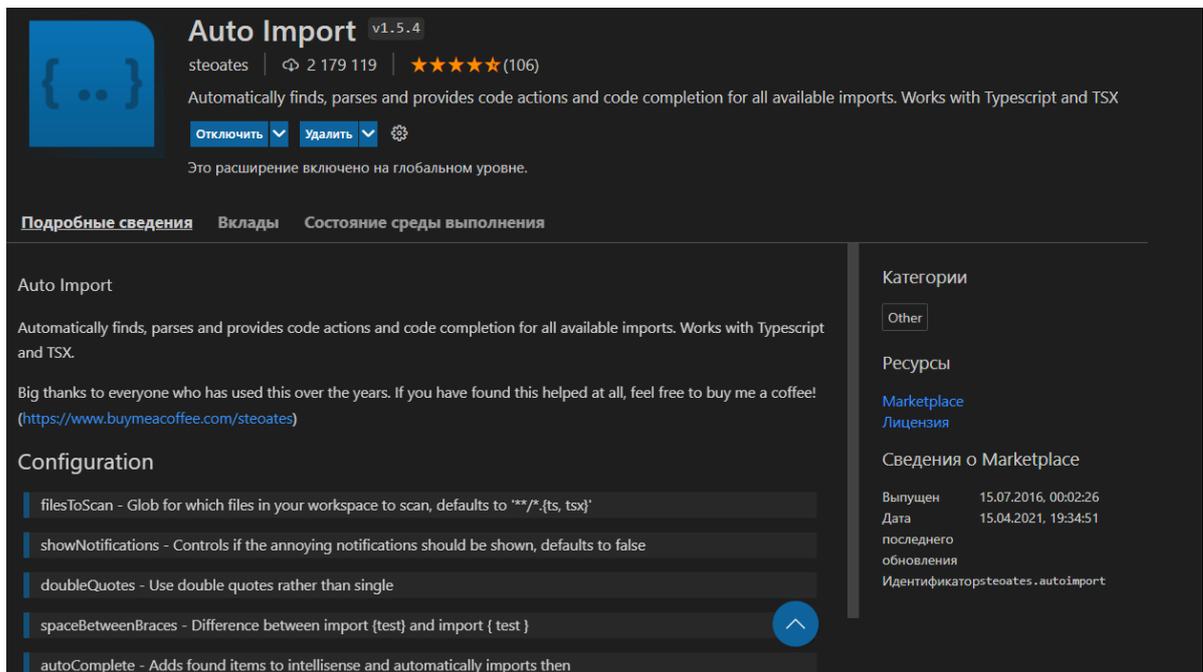


Рисунок 5 – Описание расширения Auto Import в редакторе Visual Studio Code

– ESLint – Интегрирует ESLint JavaScript в VS Code [6]. Это расширение статически анализирует код, чтобы быстро найти проблемы (рисунок 6).

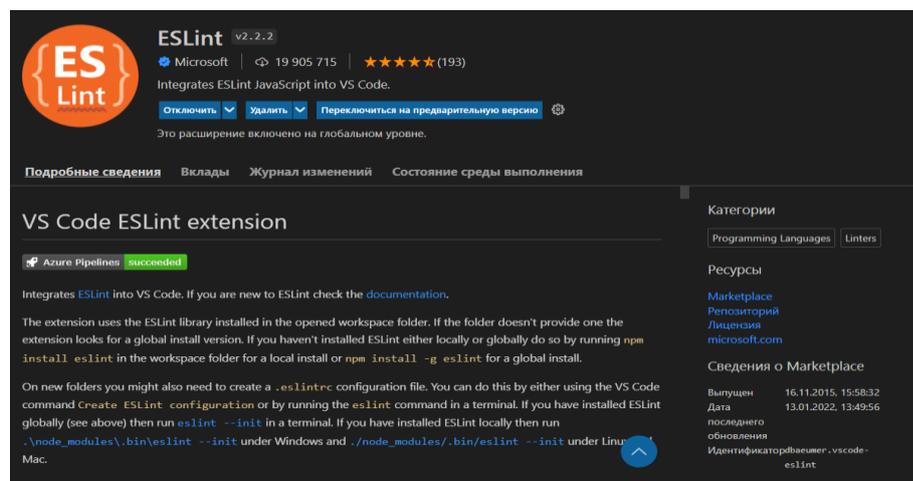


Рисунок 6 – Описание расширения ESLint в редакторе Visual Studio Code

2.2 Выбор языков разработки

Основными языками для разработки веб-приложений зачастую являются:

- HTML – стандартизированный язык разметки документов для просмотра веб-страниц в браузере [8];
- CSS – это код, который вы используете для стилизации вашей веб-страницы [1];
- JavaScript – это легковесный, интерпретируемый [19] или JIT-компилируемый, объектно-ориентированный язык с функциями первого класса [15].

В настоящее время все эти языки заменяются Фреймворками в которых используется синтаксис описанных выше языков.

2.2.1 Выбор фреймворка для разработки

React – это JavaScript библиотека для написания user-интерфейса созданная компанией Facebook в 2013 году (рисунок 7). Данная библиотека используется для обработки слоя представления в веб-приложениях. Является модифицируемым фреймворком, поверх которого можно использовать сторонние библиотеки [17].



Рисунок 7 – Библиотека React

Особенности React:

– компонентный подход – это подход при котором приложение можно разбивать на маленькие блоки, что упрощает разработку и будущую поддержку проекта [18];

– виртуальный DOM – позволяет изменять только необходимые компоненты в страницы без обновления всей страницы, что позволяет приложению работать быстрее (рисунок 8);

– постоянная поддержка разработчиками и обновления фреймворка.

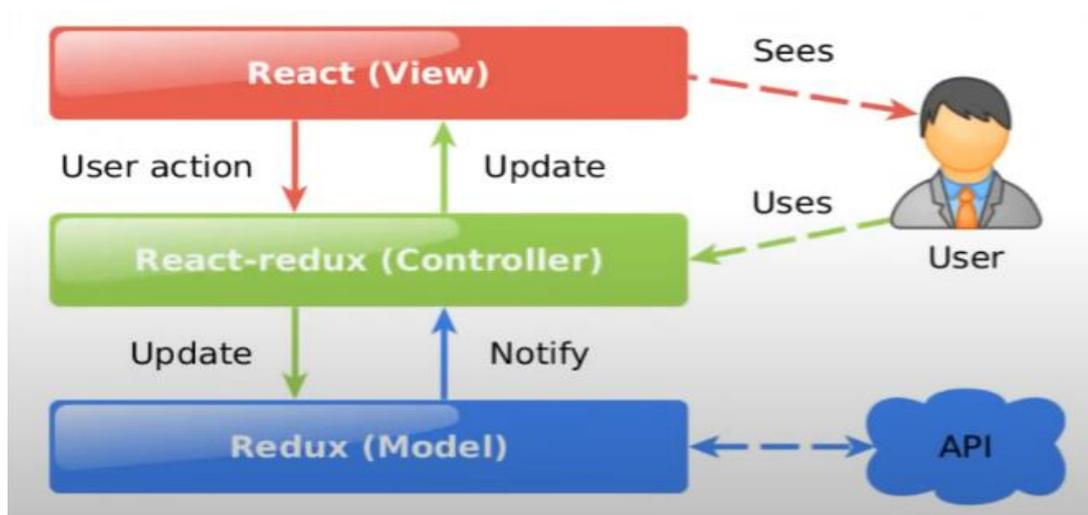


Рисунок 8 – Схема работы библиотеки React в браузере

2.2.2 Определение jsx

JSX (JavaScript Syntax Extension) – расширение синтаксиса JavaScript. JSX представляет из себя синтаксис очень похожий на Html (рисунок 9). При создании элемента с помощью функции [20].

```
React.createElement('h1', null, 'React');
```

Рисунок 9 – Пример React элемента после обработки JSX

Аналогичный элемент в JSX представлен на рисунке 10.

```
<h1>React</h1>
```

Рисунок 10 – Пример JSX кода

Не браузер, не движок JavaScript не знает что такое JSX. Это означает, что код на JSX необходимо транспилировать (перевести) в JavaScript. Пример кода представлен на рисунке 11.

```
<div className="container">  
    
  <h1>React</h1>  
  <p>...</p>  
</div>
```

Рисунок 11 – Пример JSX кода до компиляции в JavaScript

Превратится в такой код на JavaScript представленный на рисунке 12:

```
React.createElement('div', { className: 'container' },  
  React.createElement('img', { src: '...' }),  
  React.createElement('h1', null, 'React'),  
  React.createElement('p', null, '...')  
);
```

Рисунок 12 – Пример JavaScript кода после компиляции из JSX

2.2.3 Определение Next.js технологии

Next.js – Next.js дает вам лучший опыт разработки со всеми функциями (рисунок 13), которые вам нужны для производства: гибридный статический и серверный рендеринг, поддержка TypeScript, интеллектуальная сборка, предварительная выборка маршрута и многое другое [11].



Рисунок 13 – Логотип фреймворка Next.js

Next.js построен поверх React фреймворка и является библиотекой с открытым исходным кодом. Next.js в своих продуктах используют такие компании как: Binance, Twitch, Netflix, Nike, Uber и другие.

Главными функциями определяющие популярность среди разработчиков Next.js являются:

- оптимизация изображений;
- SSR (Server-Side Rendering);
- встроенная поддержка CSS, Sass;
- маршрутизация файловой системы.

2.3 Разработка веб-приложения с применением Next.js технологии

Чтобы пользоваться инструментами разработки Next.js необходимо использовать пакеты npm - менеджер пакетов, входящий в состав Node.js [3]. Для использования пакетов npm нужно установить Node.js - представляет среду выполнения кода на JavaScript, которая построена на основе движка JavaScript Chrome V8, который позволяет транслировать вызовы на языке JavaScript в машинный код [13]. Установка проводится через официальный сайт платформы - <https://nodejs.org/en/> (рисунок 14).

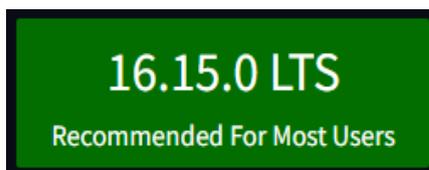


Рисунок 14 – Кнопка скачивания рекомендованной версии Node.js

Дальше устанавливается пустой проект Next.js командой которая представлена на рисунке 15.

```
npx create-next-app nextjs-blog --use-npm --example "https://github.com/vercel/next-learn/tree/master/basics/learn-starter"
```

Рисунок 15 – Команда установки Next.js проекта

Следуя командой установки, таким как: выбор пути проекта, название проекта и т.д. создается проект Next.js (рисунок 16).

```
Need to install the following packages:
  create-next-app
Ok to proceed? (y) y
Creating a new Next.js app in C:\Users\shche\react\test_project\nextjs-blog.

Downloading files from repo https://github.com/vercel/next-learn/tree/master/basics/learn-starter. This might take a moment.

Installing packages. This might take a couple of minutes.

added 15 packages, and audited 16 packages in 48s

2 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

Success! Created nextjs-blog at C:\Users\shche\react\test_project\nextjs-blog
Inside that directory, you can run several commands:

  npm run dev
    Starts the development server.

  npm run build
    Builds the app for production.

  npm start
    Runs the built app in production mode.

We suggest that you begin by typing:

  cd nextjs-blog
  npm run dev

A new version of `create-next-app` is available!
You can update by running: npm i -g create-next-app
```

Рисунок 16 – Установка Next.js

Проект запускается на локальной машине командой (рисунок 17)

```
PS C:\Users\shche\react\test_project\nextjs-blog> npm run dev

> dev
> next dev

ready - started server on 0.0.0.0:3000, url: http://localhost:3000
wait - compiling...
event - compiled client and server successfully in 3.7s (113 modules)
█
```

Рисунок 17 – Запуск Next.js проекта с указанием локального домена

Проект доступен по адресу: <http://localhost:3000> (рисунок 18). На нем присутствует одностраничный сайт, в котором находятся ссылка на:

- документацию Next.js;

- бесплатные интерактивные курсы для изучения Next.js;
- примеры Next.js проектов;
- сервис для деплоя.

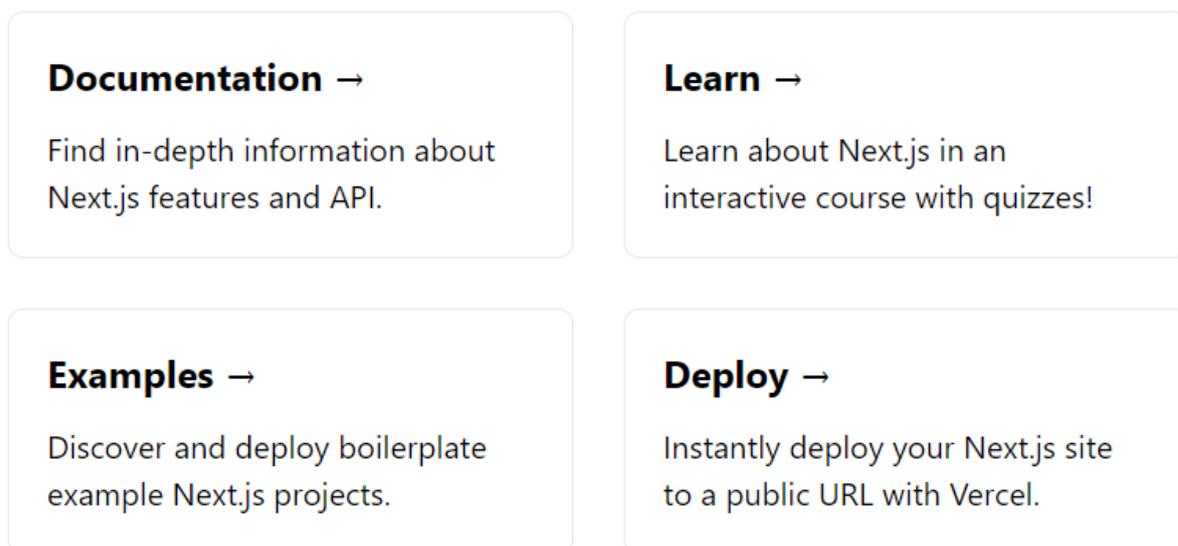


Рисунок 18 – Кнопки в созданном Next.js проекте

Папки и файлы, которые доступны сразу после установки:

- `.next` - папка с системными файлами необходимая для работы Next.js (рисунок 19);

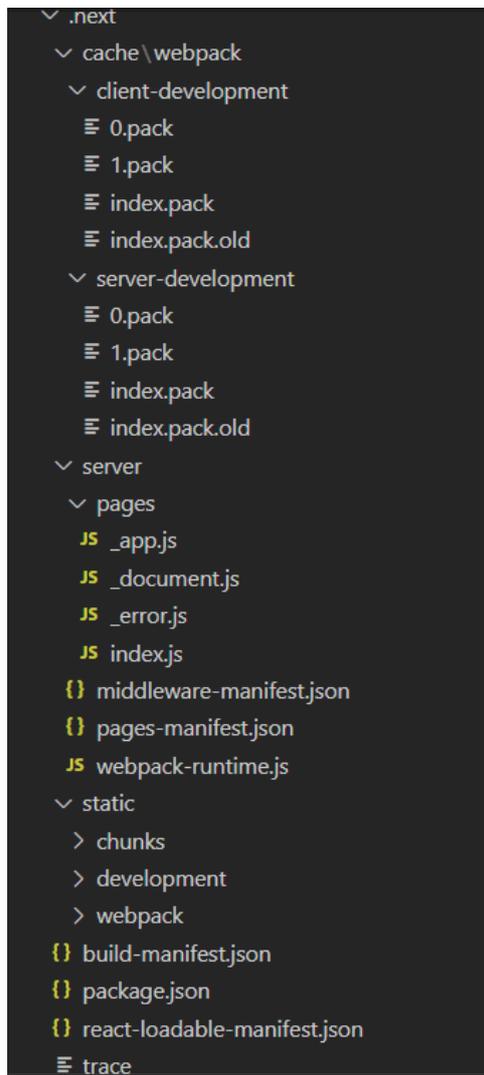


Рисунок 19 – содержимое папки .next

– node_modules - папка пакетов npm (рисунок 20);

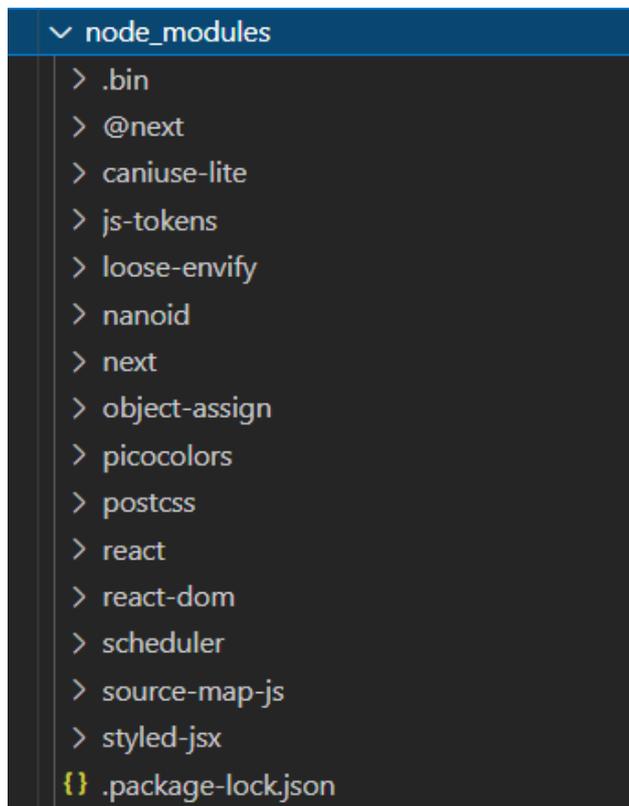


Рисунок 20 – содержимое папки node_modules

– pages - в этой папке хранятся страницы проекта (рисунок 21);



Рисунок 21 – содержимое папки pages

– public - папка с изображениями проекта (рисунок 22);



Рисунок 22 – содержимое папки public

– .gitignore - файл в котором указывается какие папки либо файлы игнорировать git (рисунок 23);

```
1 # See https://help.github.com/articles/ignoring-files/ for more about ignoring files.
2
3 # dependencies
4 /node_modules
5 /.pnp
6 .pnp.js
7
8 # testing
9 /coverage
10
11 # next.js
12 /.next/
13 /out/
14
15 # production
16 /build
17
18 # misc
19 .DS_Store
20
21 # debug
22 npm-debug.log*
23 yarn-debug.log*
24 yarn-error.log*
25
26 # local env files
27 .env.local
28 .env.development.local
29 .env.test.local
30 .env.production.local
31
```

Рисунок 23 – содержимое файла .gitignore

– package-lock.json (рисунок 24);

```

1  {
2    "name": "nextjs-blog",
3    "lockfileVersion": 2,
4    "requires": true,
5    "packages": {
6      "": {
7        "dependencies": {
8          "next": "latest",
9          "react": "17.0.2",
10         "react-dom": "17.0.2"
11       }
12     },
13     "node_modules/@next/env": {
14       "version": "12.1.6",
15       "resolved": "https://registry.npmjs.org/@next/env/-/env-12.1.6.tgz",
16       "integrity": "sha512-Te/OBDXF5odPU6jLXYPAXpmZr/AkG6DCATxttQxq0Waq6eDFX25Db3dK0120GZrSZmv4QCe9KsZmJKDbWs40A==",
17     },
18     "node_modules/@next/swc-android-arm-eabi": {
19       "version": "12.1.6",
20       "resolved": "https://registry.npmjs.org/@next/swc-android-arm-eabi/-/swc-android-arm-eabi-12.1.6.tgz",
21       "integrity": "sha512-BxB3QAAAXWgk/K7EedvzxJr2dE014mghBSA9i0EAv0bMgF+MRq4PoASjuHi15M2zFowpcRG8XQhMFtxftCleQ==",
22       "cpu": [
23         "arm"
24       ],
25       "optional": true,
26       "os": [
27         "android"
28       ],
29       "engines": {
30         "node": ">= 10"

```

Рисунок 24 – содержимое файла package-lock.json

– package.json (рисунок 25);

```

1  {
2    "private": true,
3    "scripts": {
4      "dev": "next dev",
5      "build": "next build",
6      "start": "next start"
7    },
8    "dependencies": {
9      "next": "latest",
10     "react": "17.0.2",
11     "react-dom": "17.0.2"
12   }
13 }

```

Рисунок 25 – содержимое файла package.json

– README.md - файл с комментариями разработчиков Next.js (рисунок 26).

```
nextjs-blog > ⓘ README.md  
1 This is a starter template for [Learn Next.js](https://nextjs.org/learn).
```

Рисунок 26 – содержимое файла README.md

В первую очередь необходимо организовать архитектуру приложения. Создать папку для компонентов и стилей. Создать файлы в которых будет писаться код.

2.3.1 Components

Поскольку Next.js это всего лишь Фреймворк над React то синтаксис и концепции аналогичны. В основе создания интерфейса лежат функциональные переиспользуемые элементы, которые называются компонентами. Именно из этих блоков и строится UI. Компоненты в структуре Next.js приложения - должны храниться отдельно от страниц. Для начала нужно создать папку components в которой будут храниться все создаваемые компоненты.

Основным и самым важным компонентом в проекте является MainLayout.js. На большинстве сайтов используются такие элементы интерфейса как: header, footer и подобные. Без использования компонентного подхода будет необходимость писать код несколько раз, прописывая header и footer для каждой страницы отдельно. Это не только замедлит разработку, но и плохо скажется на оптимизации самого проекта. Components решает такую проблему (рисунок 27).

```

1 import Head from "next/head";
2 import Footer from "../Footer";
3 import Header from "../Header";
4 export function MainLayout({ children,title = 'Next App' }) {
5   return (
6     <>
7       <Head>
8         <title>{title} | Next diplom</title>
9         <meta name="keywords" content="university,safronov,tsu"/>
10        <meta name="description" content="site about diploms in tsu"/>
11        <meta charSet="utf-8"/>
12      </Head>
13      <Header/>
14      <main>
15        {children}
16      </main>
17      <Footer/>
18      <style jsx global>{`
19        main{
20          margin-top:100px;
21        }
22      `}</style>
23    </>
24  )
25 }

```

Рисунок 27 – Пример применения компонентов в проекте

В данном файле прописывается основная логика приложения:

- нахождение на странице header в теге <Header /> контент которого прописан в файле Header (рисунок 28);

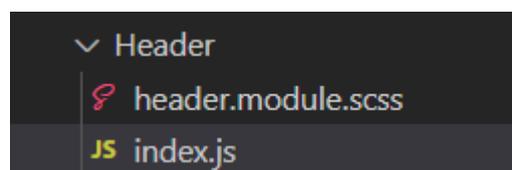


Рисунок 28 – Содержимое компонента Header

- тело приложения находится в {children};

– нахождение на странице footer в теге <Footer /> контент которого прописан в файле <Footer /> (рисунок 29).

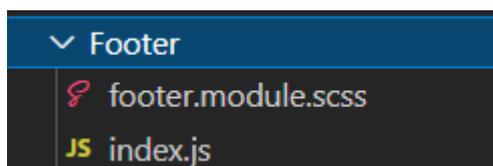


Рисунок 29 – Содержимое компонента Footer

2.3.2 SSR(server side rendering)

SSR - описывает процесс предварительного рендеринга страницы на сервере, который затем генерируется при каждом запросе пользователя [12]. Особенностью SSR является то, что сервер на каждый запрос генерирует Html на своей стороне (стороне сервера). А клиент уже получает заполненную данными страницу.

Пользователь заходит браузер, открывает приложение. Приложение делает первый запрос на сервер, и далее уже сервер посылает запрос на какой-либо API, либо напрямую в базу данных и получает ответ в виде JSON. После чего формирует готовый HTML со всеми данными со всеми HTML элементами и отдает уже готовую страницу клиенту. Таким образом в браузере, клиент получит готовый HTML со всеми необходимыми данными (рисунок 30). И никаких запросов уже клиенту делать будет не обязательно, так как все данные сервер уже подготовил.

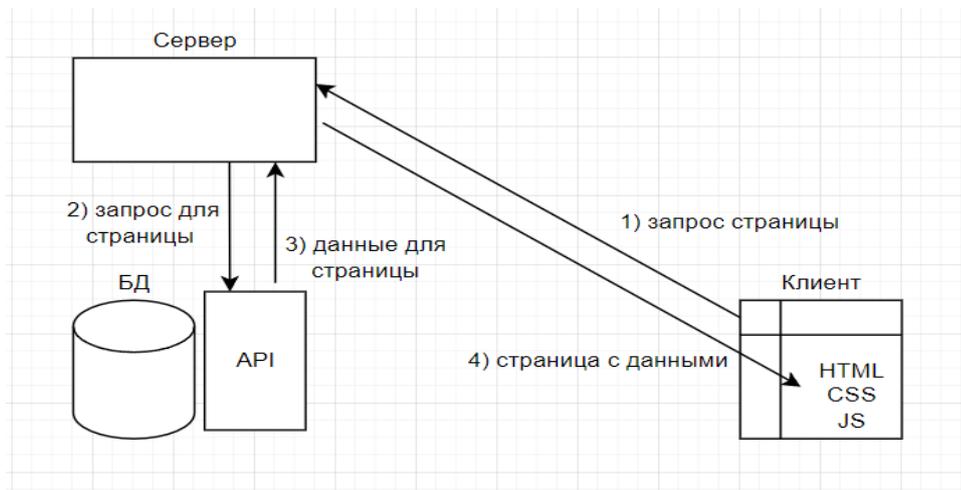


Рисунок 30 – Схема работы server side rendering

Server side rendering реализован при помощи функции `getServerSideProps()`, которая будет выполнять запрос к данным. Далее эти данные мы будем отправлять в виде `props` в сам компонент `Page`. После чего эти данные `props: {data}` соответствующие имени поля которое будет возвращено из `getServerSideProps()` (рисунок 31). Будет отображать данные в разметке. Таким образом готовая страница попадет пользователю в браузер.

```

1  function Page({ data }) {
2    // Render data...
3  }
4
5  // This gets called on every request
6  export async function getServerSideProps() {
7    // Fetch data from external API
8    const res = await fetch(`https://.../data`)
9    const data = await res.json()
10
11   // Pass data to the page via props
12   return { props: { data } }
13 }
14
15 export default Page
  
```

Рисунок 31 – Код для включения Server side rendering

2.3.3 Навигация по приложению с помощью Link

С помощью встроенного компонента Link создается маршрутизация на стороне клиента, то есть в браузере. Это означает, что при переходе на страницу, она загружается с помощью JavaScript без создания запроса на сервер. Именно благодаря этому, перемещение по страницам становится заметно быстрее.

В первую очередь необходимо заимпортировать этот компонент (рисунок 32):

```
import Link from "next/link";
```

Рисунок 32 – Импорт компонента Link

После чего использовать данный компонент при навигации внутри самого приложения, например с главной страницы на страницу с контактными данными (рисунок 33).

```
<Link href="/">Главная</Link>  
<Link href="/projects">Реализованные задачи</Link>  
<Link href="/inprogress">Задачи в разработке</Link>  
<Link href="/contacts">Контакты</Link>
```

Рисунок 33 – Применение компонента Link

2.3.4 Image

Изображения в проекте тратят больше всего времени на загрузку страницы, для этого в Next.js есть решение. Зачастую картинки подключаются с помощью тега ``. Однако в Next.js доступен встроенный тег `<Image />`. Особенности тега `<Image />` перед ``:

- lazy loading;

- показ preloader;
 - подгрузка оптимальных версий картинки под разные экраны.
- Для начала нужно также заимпортировать данный компонент (рисунок 34).

```
import Image from "next/image"
```

Рисунок 34 – Импорт компонента Image

Далее для вывода картинки применять его, задавая параметры изображения (высота, ширина) и путь до этой картинки (рисунок 35).

```
<Image  
  alt="diagram"  
  src="/stepanova_images/model.jpg"  
  width="849"  
  height="537"  
  placeholder="blur"  
>
```

Рисунок 35 – Применение компонента Link

Props placeholder со значением blur запускает механизм изображения размытой картинки до того момента пока не загрузится полная версия изображения.

2.4 Настройка взаимодействия с поисковыми роботами

Мета-теги — часть программного HTML-кода, заключенная между тегом `<head>...</head>` (заголовок страницы). Эти атрибуты не являются обязательными: они не видны пользователю, но сообщают определенную информацию поисковой системе (описание, заголовок, ключевые слова и т.д.) [4].

Одна из проблем которую решает SSR, это добавление мета-тегов на конкретную страницу. Для этого есть предустановленный компонент `<Head>`. Для начала нужно заимпортировать данный компонент (рисунок 36).

```
import Head from "next/head";
```

Рисунок 36 – Импорт компонента Head

После добавить компонент и прописать в нем (рисунок 37):

- ключевые слова;
- название страницы;
- описание страницы;
- charSet.

```
<Head>
  <title>{title}</title>
  <meta name="keywords" content="university,
    Сафронов,
    tsu, ТГУ,
    Тольятти,
    Баллистика,
    Государственный Университет
  "/>
  <meta name="description" content="Разработка выпускных квалификационных работ.
    Тематика баллистических исследований.
    Ведется студентами Тольяттинского Государственного Университета."/>
  <meta charset="utf-8"/>
</Head>
```

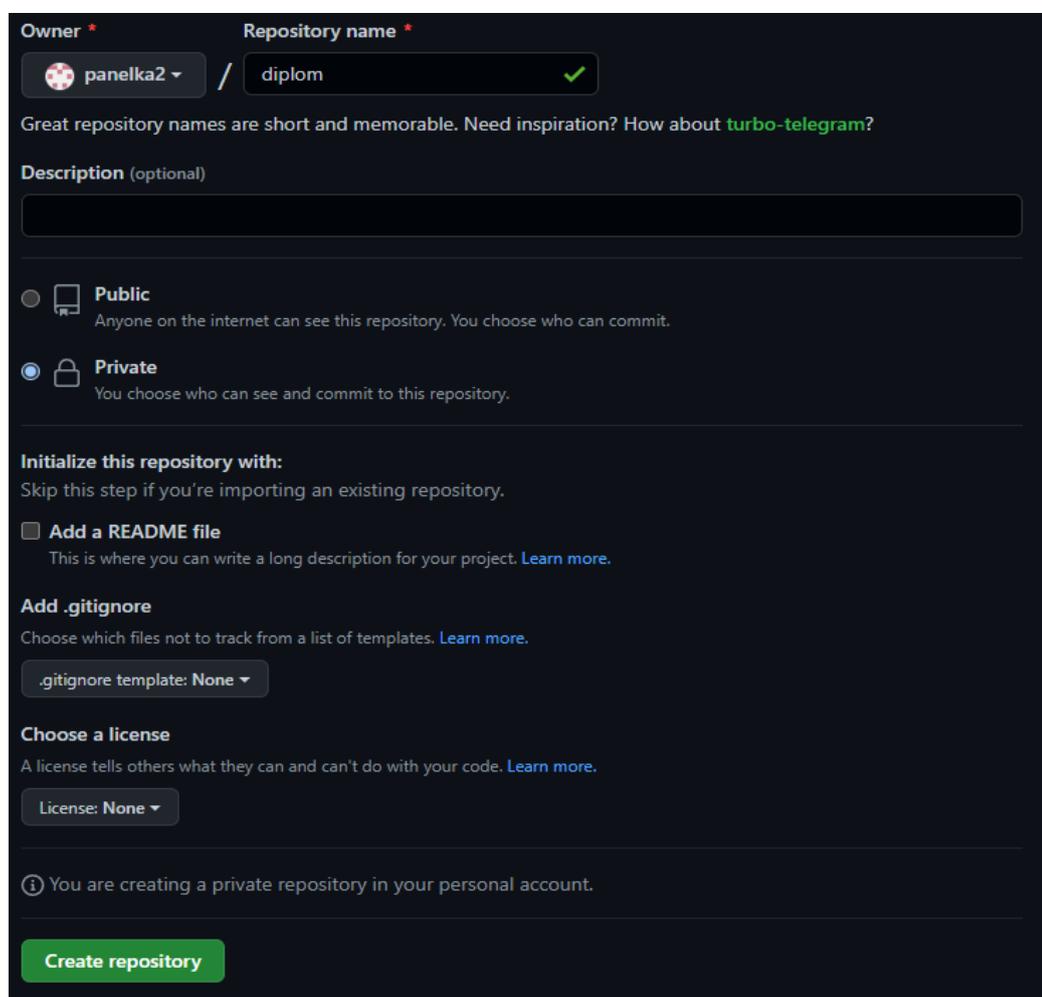
Рисунок 37 – Применение компонента Head

2.5 Публикация сайта в сети-интернет

Для того чтобы сайт можно было найти в поисковике либо просто открыть в браузере его нужно загрузить на сервер. Для того чтобы люди могли удобно изучать ваш код без необходимости скачивание проекта, подойдет GitHub.

2.5.1 Создание репозитория на GitHub

В первую очередь нужно зарегистрироваться на GitHub для появления возможности создания личного репозитория. После создается репозиторий (рисунок 38).



The screenshot shows the GitHub repository creation form. At the top, the 'Owner' is 'panelka2' and the 'Repository name' is 'diplom', which is marked as valid with a green checkmark. Below this, there is a suggestion for repository names: 'Great repository names are short and memorable. Need inspiration? How about turbo-telegram?'. The 'Description' field is empty. The visibility is set to 'Private' (selected with a radio button), with the note 'You choose who can see and commit to this repository.' Below this, the 'Initialize this repository with:' section is active, with the instruction 'Skip this step if you're importing an existing repository.' There are three options: 'Add a README file' (unchecked), 'Add .gitignore' (selected), and 'Choose a license' (selected). The '.gitignore template' is set to 'None' and the 'License' is also set to 'None'. At the bottom, there is a note: 'You are creating a private repository in your personal account.' and a green 'Create repository' button.

Рисунок 38 – Создание репозитория на GitHub

И добавляются файлы проекта (рисунок 39).

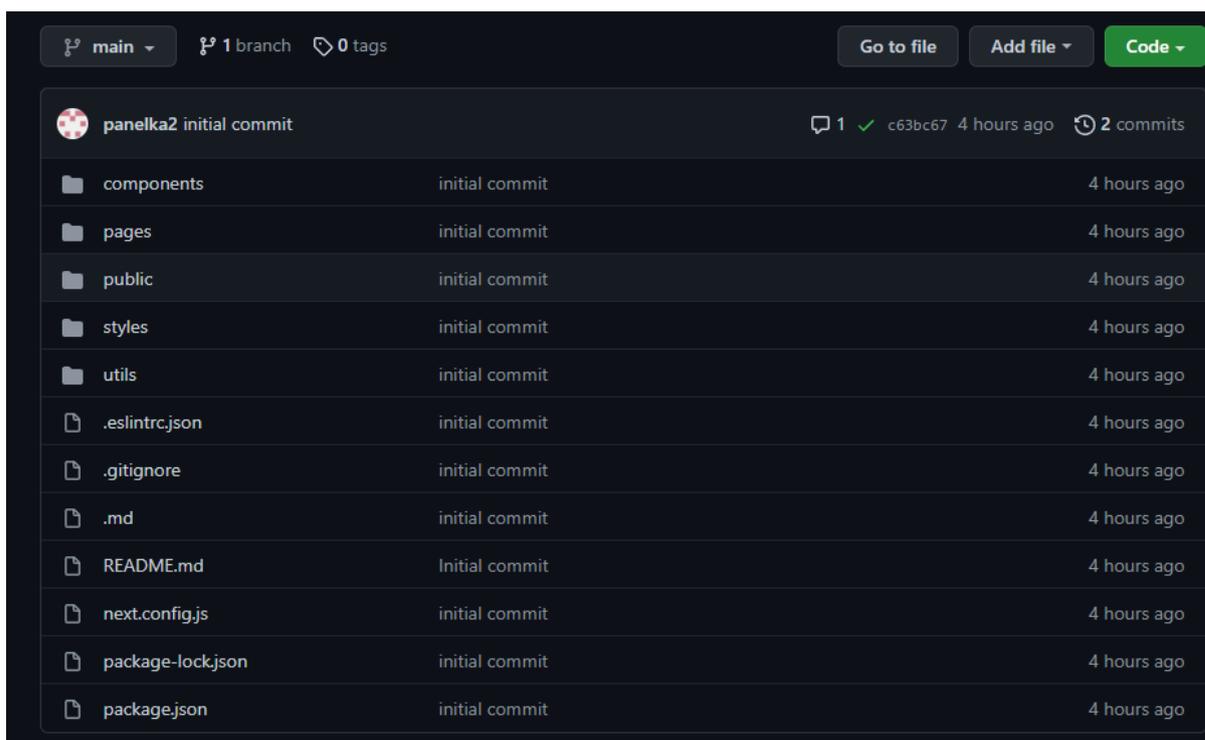


Рисунок 39 – Файлы и папки репозитория проекта

2.5.2 Деплой проекта.

Разработчики Next.js компания Vercel предоставляют свои сервера для публикации в сети интернет. И предоставляют свои SSL сертификаты, правда без возможности задавать свой домен. Но для тестирования продуктов более чем подходит, за счет скорости самого деплоя (рисунок 40).

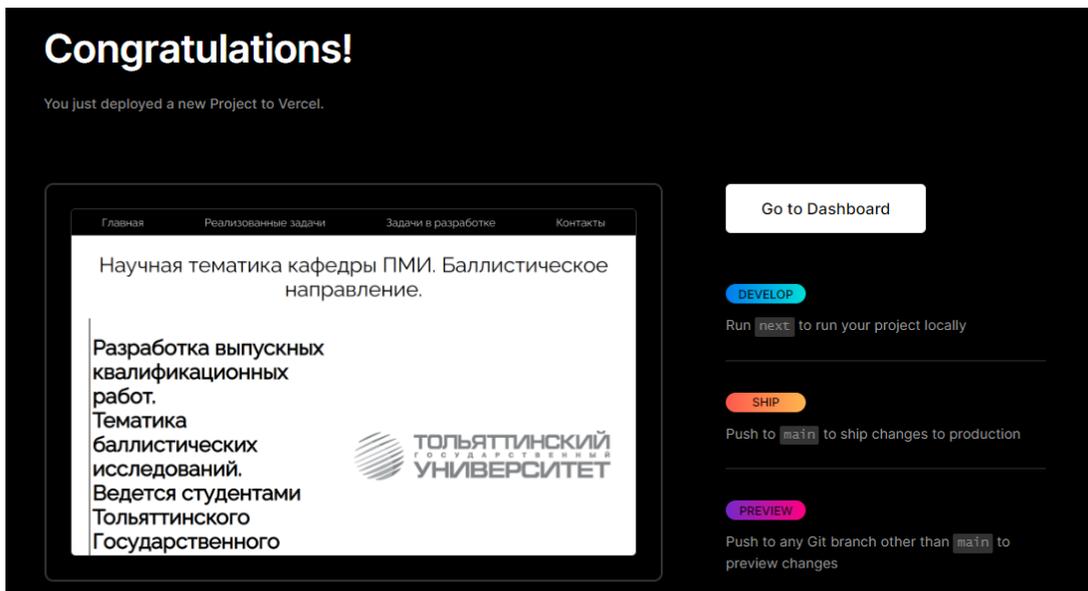


Рисунок 40 – Экран готовности деплоя

Вывод по разделу 2

В результате данной главы, было разработано веб-приложение и опубликовано в сети интернет.

3. Описание разработанного Web-приложения

3.1 Описание интерфейса пользователя

Навигация основных страниц приложения находится в так называемой “шапке” (рисунок 41).



Рисунок 41 – Шапка веб-приложения

По ссылкам вверху приложения можно перейти на страницы:

- главная;
- реализованные задачи;
- задачи в разработке;
- контакты.

На главной странице пользователь знакомится с направлением дипломных работ (рисунок 42), руководителем направления, а также с более подробным описанием остальных страниц в приложении (рисунок 43).

Научная тематика кафедры ПМИ. Баллистическое направление.

Разработка выпускных
квалификационных работ.
Тематика баллистических
исследований.
Ведется студентами
Тольяттинского
Государственного
Университета.



Рисунок 42 – Дизайн главной страницы веб-приложения

Александр Иванович Сафронов	Узнать подробнее о преподавателе можно на сайте Тольяттинского Государственного Университета - перейти
Реализованные задачи	Ознакомиться с выпускными квалификационными работами можно в шапке сайта, либо по ссылке
Задачи в разработке	Для ознакомления с выпускными квалификационными работами, находящимися в настоящее время в разработке, можно в шапке сайта, либо по ссылке
Контакты	Данные для связи с профессором Сафроновым или разработчиком ресурса доступны в шапке сайта, либо по ссылке

Рисунок 43 – Дизайн главной страницы веб-приложения с ссылками на страницы

При переходе на несуществующую страницу приложения пользователь увидит сообщение об ошибке (рисунок 44).

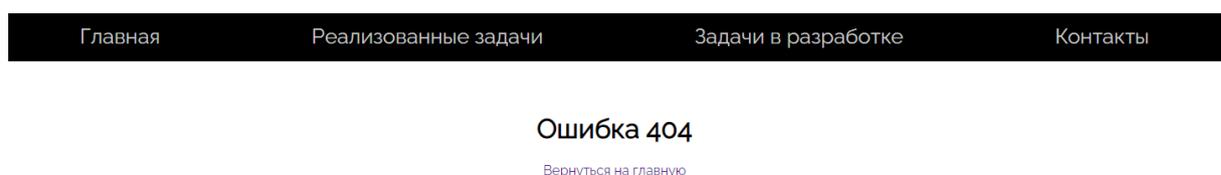


Рисунок 44 – Сообщение об ошибке

Цвет в каждой карточке (рисунок 45) и странице проекта разный и подобран индивидуально, это позволяет выделять работы на фоне друг друга (рисунок 46).

Разработка ПО оценки знаний студентов

Период расчета	Разработка (проект)				Прибыль	
	Частичная стоимость	Сальдо начальное по кредиту	Погашение кредита	Сальдо конечное по кредиту	Расчетная прибыль	Чистая прибыль
1-4. 200х	88906,12	-320062,03	88906,12	-231155,9	13779,38	11023,51
5-8. 200х	88906,12	-231155,9	88906,12	-142249,79	13779,38	11023,51
9-12. 200х	88906,12	-142249,79	88906,12	-53343,67	13779,38	11023,51
1-3. 200х	88906,12	-53343,67	55006,12	0	49341,83	39473,47
4-6. 200х	88906,12	0	0	0	102685,5	82148,4

Студент Степанова М.В.

Разработанная система оценки знаний может стать основой для дальнейшей модернизации, а также для внедрения в различные учебные заведения и предприятия, где есть необходимость в проверке знаний.

[Подробнее](#)

Рисунок 45 – Дизайн карточки работы студента



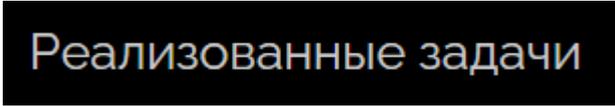
Рисунок 46 – Блоки со схемами работы студента

3.2 Описание основного функционала Web-приложения

Пользователь может перемещаться по приложению посредством нажатия на ссылки (рисунок 48), которые находятся вверху страницы (рисунок 49), либо подчеркнутые синим шрифтом слова (рисунок 47).

Ознакомиться с выпускными квалификационными работами можно в шапке сайта, либо по [ссылке](#)

Рисунок 47 – Пример ссылки на главной странице



Реализованные задачи

Рисунок 48 – Кнопка перехода на страницу реализованных задач

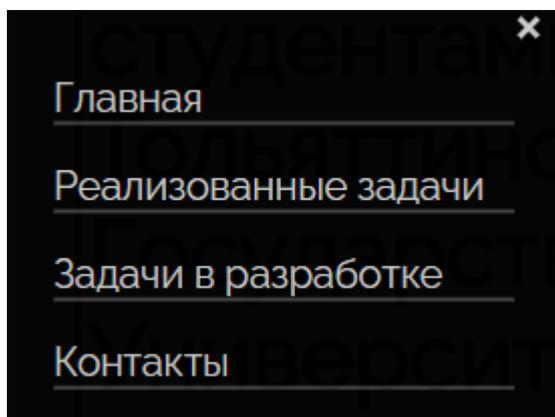


Рисунок 49 – Кнопки перехода по страницам веб-приложения в мобильной версии

Работы профессора Сафронова выложены в виде окон, которые можно открыть отдельным окном, либо скачать (рисунок 50).

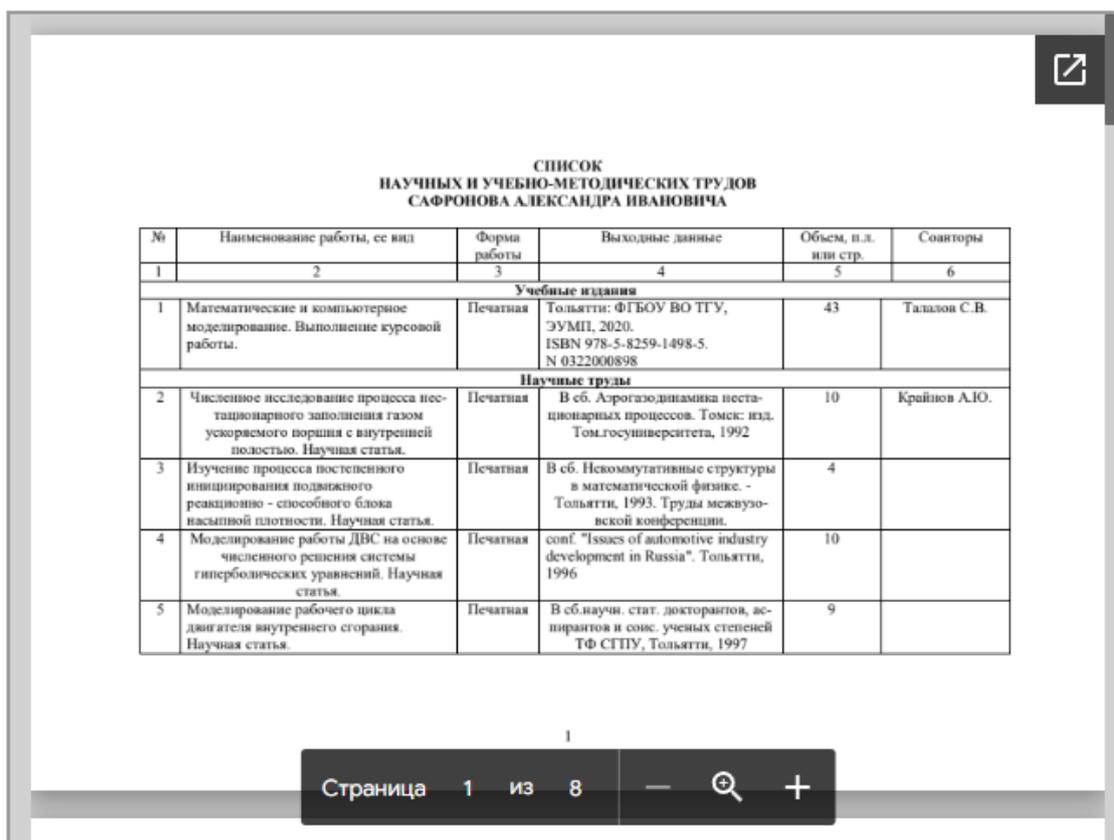


Рисунок 50 – Прикрепленный файл на главной странице

3.3 Адаптивность и логика приложения

3.3.1 Адаптивность

Дизайн веб-страниц, обеспечивающий правильное отображение сайта на различных устройствах, подключённых к интернету, и динамически подстраивающийся под заданные размеры окна браузера [16]. Минимальной шириной приложения для безошибочной работы приложения выбрана ширина монитора в 375 пикселей. Данная ширина относится в основном к устройствам Apple, а именно модели Iphone 5, так как данные устройства продолжают пользоваться люди.

Разработка велась под основные размеры экранов устройств Apple (рисунок 51).

	Разрешение дисплея	Viewport
iPhone		
iPhone XR	828 x 1792	414 x 896
iPhone XS	1125 x 2436	375 x 812
iPhone XS Max	1242 x 2688	414 x 896
iPhone X	1125 x 2436	375 x 812
iPhone 8 Plus	1080 x 1920	414 x 736
iPhone 8	750 x 1334	375 x 667
iPhone 7 Plus	1080 x 1920	414 x 736
iPhone 7	750 x 1334	375 x 667
iPhone 6 Plus/6S Plus	1080 x 1920	414 x 736
iPhone 6/6S	750 x 1334	375 x 667
iPhone 5	640 x 1136	320 x 568

Рисунок 51 – Расширения мобильных устройств Apple

Под основные размеры экранов Android (рисунок 52).

Nexus 6P	1440 x 2560	412 x 732
Nexus 5X	1080 x 1920	412 x 732
Google Pixel 3 XL	1440 x 2960	412 x 847
Google Pixel 3	1080 x 2160	412 x 824
Google Pixel 2 XL	1440 x 2560	412 x 732
Google Pixel XL	1440 x 2560	412 x 732
Google Pixel	1080 x 1920	412 x 732
Samsung Galaxy Note 9	1440 x 2960	360 x 740
Samsung Galaxy Note 5	1440 x 2560	480 x 853
LG G5	1440 x 2560	480 x 853
One Plus 3	1080 x 1920	480 x 853
Samsung Galaxy S9+	1440 x 2960	360 x 740
Samsung Galaxy S9	1440 x 2960	360 x 740
Samsung Galaxy S8+	1440 x 2960	360 x 740
Samsung Galaxy S8	1440 x 2960	360 x 740
Samsung Galaxy S7 Edge	1440 x 2560	360 x 640
Samsung Galaxy S7	1440 x 2560	360 x 640

Рисунок 52 – Расширения мобильных устройств Android

И под основные экраны персональных компьютеров (рисунок 53).

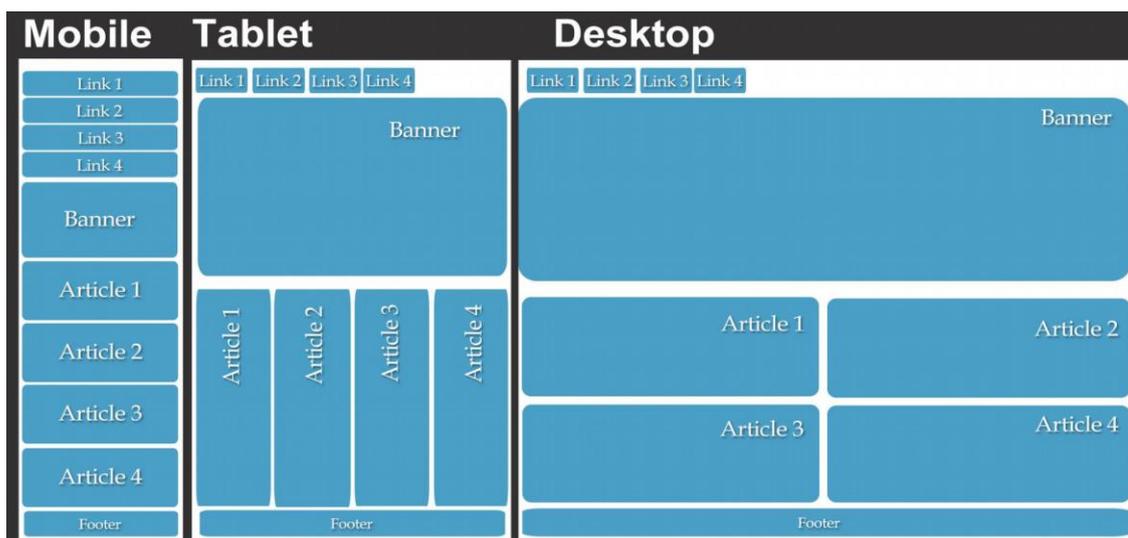


Рисунок 53 – Логика блоков разных версий приложения

Адаптивность реализована с помощью flex подхода при написании визуальной части разработки приложения (рисунок 54).

```
.block_photos_fourth_section {  
  padding-left: $padding-in-sections;  
  display: flex;  
  justify-content: space-around;  
  margin-top: 4rem;  
}  
.solo_block {  
  border: 1px solid #ef800b;  
  border-radius: 30px;  
  width: 31%;  
  display: flex;  
  flex-direction: column;  
}
```

Рисунок 54 – Код реализации flex подхода

А также с использованием @media в коде SCSS (рисунок 55).

```
@media (max-width: 1155px) {  
  .first_section header, .second_section header, .third_section header {  
    position: absolute;  
    top: -44px;  
    padding-right: 10px;  
    display: block;  
    background-color: #white;  
  }  
  .fourth_section header, .fifth_section header {  
    top: -37px;  
    position: absolute;  
    padding-right: 10px;  
    display: block;  
    background-color: #white;  
  }  
  .main_page {  
    margin: 0 20px;  
  }  
  .main_page section {  
    position: relative;  
  }  
  .about_blocks {  
    padding-left: 0;  
    flex-direction: column;  
  }  
}
```

Рисунок 55 – Код реализации @media подхода

Данный код изменяет дизайн определенных элементов при изменении ширины устройства.

Дизайн страницы реализованных задач при ширине экрана 1920 пикселей (рисунок 56).

Реализованные задачи

Ознакомьтесь с завершенными проектами,
которые нашли,
либо найдут себе применение в будущем.

Разработка ПО оценки знаний студентов

Период расчета	Разработка (проект)				Прибыль		Студент Степанова М.В.
	Частичная стоимость	Сальдо начальное по кредиту	Погашение кредита	Сальдо конечное по кредиту	Расчетная прибыль	Чистая прибыль	
1-4. 200x	88906,12	-320062,03	88906,12	-231155,9	13779,38	11023,51	Разработанная система оценки знаний может стать основой для дальнейшей модернизации, а также для внедрения в различные учебные заведения и предприятия, где есть необходимость в проверке <u>знаний</u> .
5-6. 200x	88906,12	-231155,9	88906,12	-142249,79	13779,38	11023,51	
9-12. 200x	88906,12	-142249,79	88906,12	-53343,67	13779,38	11023,51	
1-3. 200x	88906,12	-53343,67	55006,12	0	49341,83	39473,47	

Рисунок 56 – Дизайн страницы реализованных задач при ширине 1920 пикселей

Дизайн страницы реализованных задачи при ширине экрана 375 пикселей (рисунок 57).

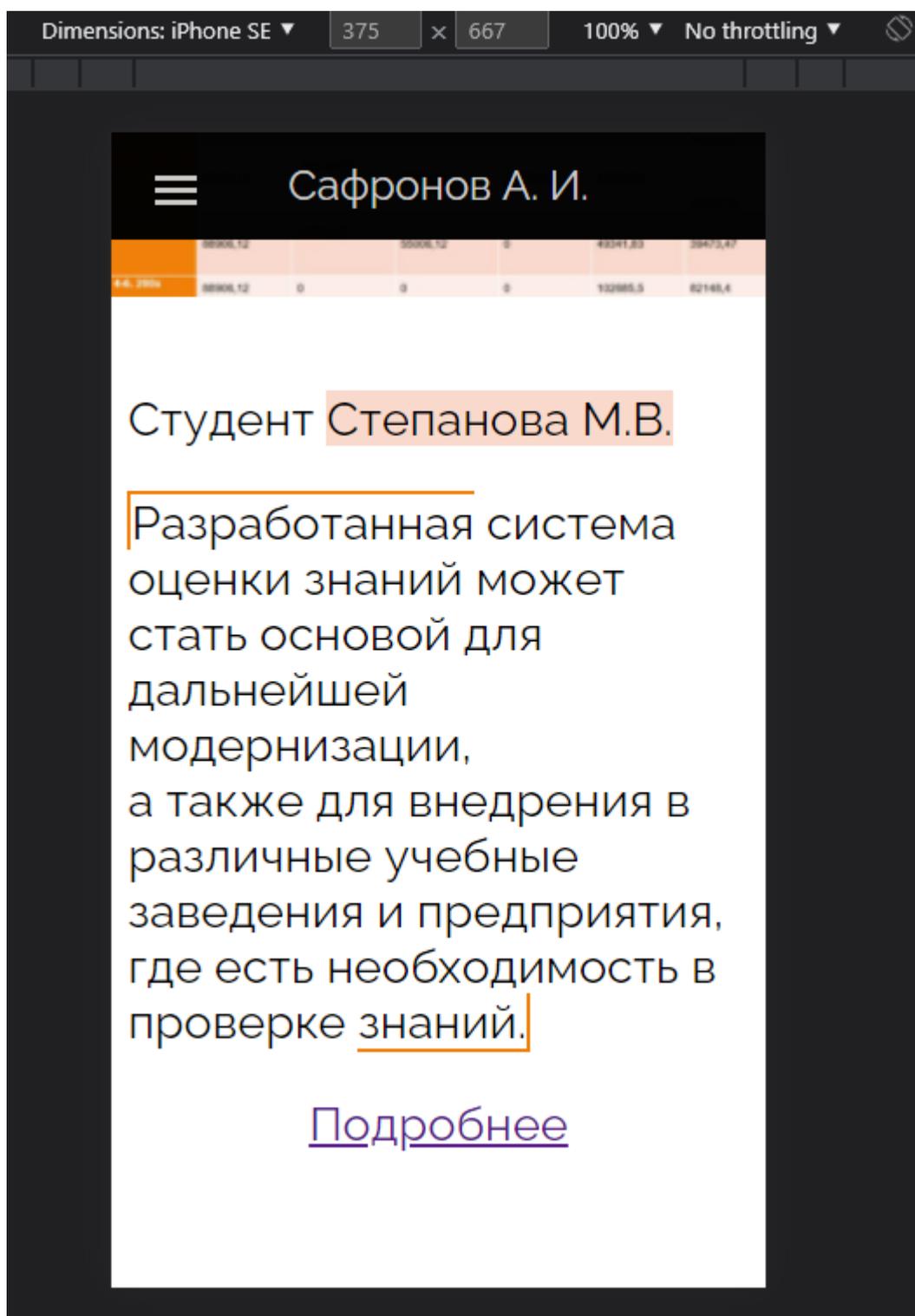


Рисунок 57 – Дизайн страницы реализованных задач при ширине 375 пикселей

Дизайн подробной страницы о проекте при ширине экрана 1920 пикселей (рисунок 58).

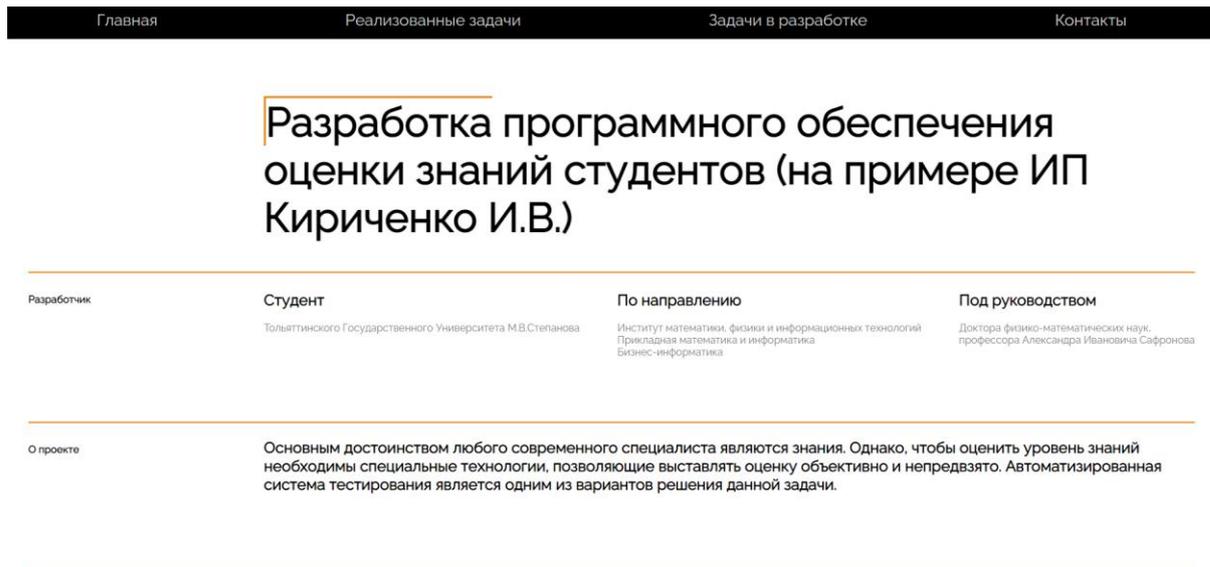


Рисунок 58 – Дизайн подробной страницы о проекте при ширине экрана 1920 пикселей

Дизайн подробной страницы о проекте при ширине экрана 375 пикселей (рисунок 59).

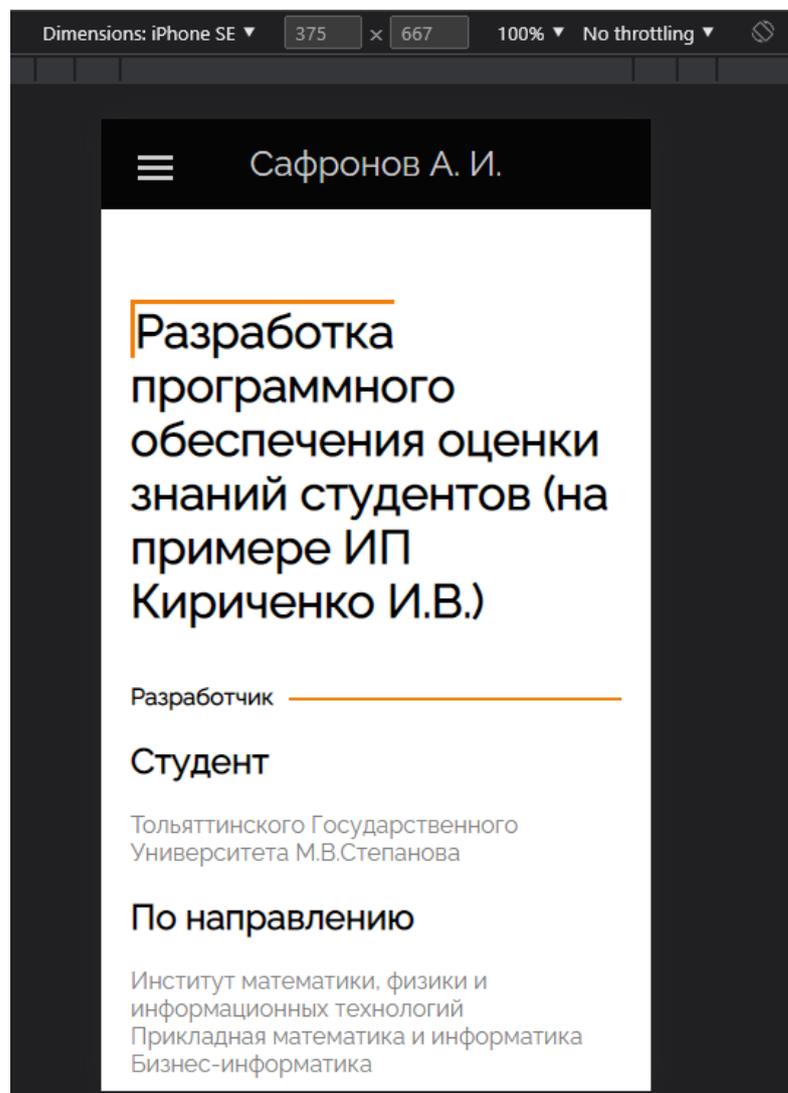


Рисунок 59 – Дизайн подробной страницы о проекте при ширине экрана 375 пикселей

3.3.2 Логика приложения

Доступ к основным страницам приложения доступны на всех страницах в области проекта, как в полной версии, так и в мобильной (рисунок 60).

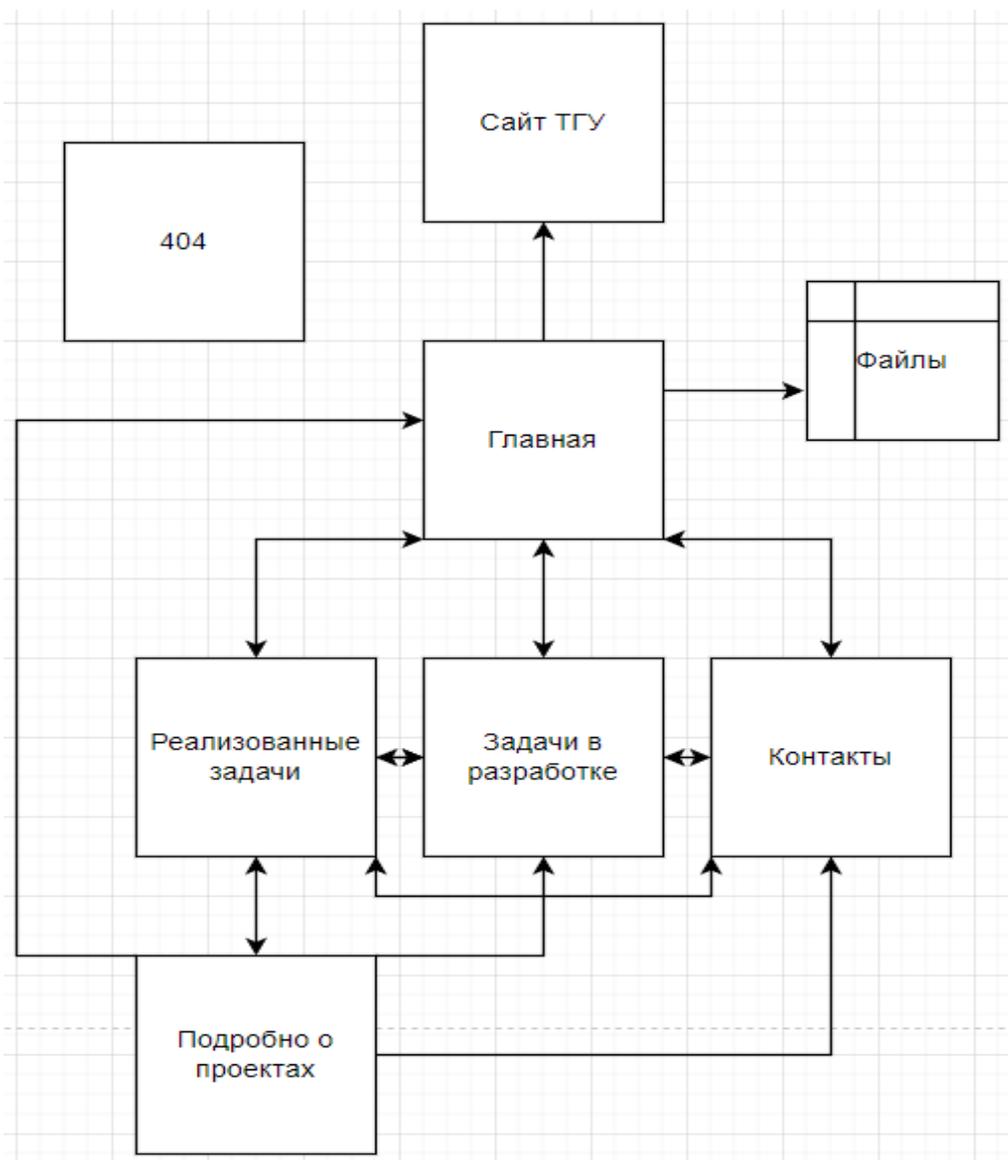


Рисунок 60 – Логика приложения

Дополнительная навигация присутствует на главной странице. Файлы с проектами и разработками профессора Сафронова доступны только на главной странице.

Вывод по разделу 3

В результате данной главы, было описано веб-приложение, его возможности и функционал.

Заключение

В рамках выполнения данной выпускной квалификационной работы было создано веб-приложение по размещению результатов студенческих исследований в области внутренней баллистики по тематике кафедральной научной работы.

В веб-приложение вошли основные труды А.И Сафронова по которым в дальнейшем предполагается проведение широких баллистических исследований. Отдельной темой разработанного веб-приложения является демонстрация выпускных квалификационных работ студентов кафедры по внутренним баллистическим исследованиям, с подробным описанием выполненных работ.

Представление тезисов, графиков, выводов и контактной информации студентов позволяют комплексно изучить представленные работы и связаться с разработчиком либо с руководителем данной работы.

Большая часть исследований посвящена разработке предложенной эстафетной схемы метания. Детальные разработки этой предложенной схемы находят потенциальных заказчиков в виде отраслевых предприятий.

В качестве инструмента разработки был выбран Фреймворк Next.js. Он является инструментом разработки для оптимизации работы веб-приложения. А именно изображений и структуры страницы DOM. Разработка с помощью Next.js позволяет масштабировать приложение без дополнительного переписывания кода.

Подводя итог, можно сделать вывод, что разработанное веб-приложение реализует все поставленные задачи, а именно, выполняет функции сопровождения научных проектов с выполнением документирования проводимых этапов.

Список используемой литературы

1. Alex Banks, Eve Porcello Learning React. 2020 23с.
2. Atom (текстовый редактор) [Электронный ресурс] // URL: [https://ru.wikipedia.org/wiki/Atom_\(%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80\)](https://ru.wikipedia.org/wiki/Atom_(%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80))
3. Azat Mardan React Quickly. 2019 42 с.
4. CSS basics [Электронный ресурс] // URL: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics
5. Douglas Crockford How JavaScript Works. 2018 122 с.
6. HTML [Электронный ресурс] // URL: <https://ru.wikipedia.org/wiki/HTML>
7. IDE Brackets [Электронный ресурс] // URL: <https://brackets.io/>
8. IDE Vs Code [Электронный ресурс] // URL: <https://code.visualstudio.com/>
9. JavaScript [Электронный ресурс] // URL: <https://developer.mozilla.org/ru/docs/Web/JavaScript>
10. Mark Thomas React in Action. 2019 234 с.
11. Next.js SSR [Электронный ресурс] // URL: <https://dev-gang.ru/article/ssg-protiv-ssr-v-nextjs-ph74uqpemd/>
12. Next.js.org [Электронный ресурс] // URL: <https://nextjs.org/>
13. Node.js 1.1 [Электронный ресурс] // URL: <https://metanit.com/web/nodejs/1.1.php>
14. NPM [Электронный ресурс] // URL: <https://ru.wikipedia.org/wiki/Npm>
15. Sublime_Text [Электронный ресурс] // URL: https://ru.wikipedia.org/wiki/Sublime_Text

16. VisualStudio autoimport [Электронный ресурс] // URL:
<https://marketplace.visualstudio.com/items?itemName=steoates.autoimport>

17. VisualStudio eslint [Электронный ресурс] // URL:
<https://marketplace.visualstudio.com/items?itemName=dbaumeier.vscodeslint&ssr=false#overview>

18. Адаптивный веб-сайт [Электронный ресурс] // URL:
https://ru.wikipedia.org/wiki/%D0%90%D0%B4%D0%B0%D0%BF%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D0%B2%D0%B5%D0%B1-%D0%B4%D0%B8%D0%B7%D0%B0%D0%B9%D0%BD

19. Интегрированная среда разработки [Электронный ресурс] // URL:
https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8

20. Мета теги [Электронный ресурс] // URL:
<https://www.ashmanov.com/education/articles/meta-tegi/>