Geekbrains

Создание современного веб-портфолио для студии "Mosh.su", реализованная с применением передовых технологий.

Программа: Разработчик

Специализация: Frontend разработка (React)

ФИО: Зверев Владимир Сергеевич

Москва

2025

Оглавление

[ВВЕДЕНИЕ 3](#_Toc189753495)

[1.1 Актуальность темы 3](#_Toc189753496)

[1.2 Цель и задачи дипломной работы 7](#_Toc189753497)

[1.3 Краткое описание проекта 9](#_Toc189753498)

[2. ОБЗОР ИСПОЛЬЗУЕМЫХ ТЕХНОЛОГИЙ 10](#_Toc189753499)

[2.1 HTML5, CSS3, SaSS 10](#_Toc189753500)

[2.2 JavaScript 12](#_Toc189753501)

[2.3 React: описание и применение 14](#_Toc189753502)

[2.4 Bootstrap: Фреймворк для визуализации данных 17](#_Toc189753503)

[2.5 React Spring — это JavaScript-библиотека для создания интерактивных анимаций 20](#_Toc189753504)

[2.6 Инструменты и среда для разработки 22](#_Toc189753505)

[2.7 Хостинг для размещения веб проекта в сети. 24](#_Toc189753506)

[3. ПРОЕКТИРОВАНИЕ ВЕБ-ПОРТФОЛИО 25](#_Toc189753507)

[3.1 Структура веб-портфолио 25](#_Toc189753508)

[3.2 Дизайн интерфейса и пользовательского опыта 28](#_Toc189753509)

[3.3 Разработка адаптивной верстки 30](#_Toc189753510)

[3.4 Запуск проекта и загрузка его на хостинг. 31](#_Toc189753511)

[4. РЕАЛИЗАЦИЯ ПРОЕКТОВ 33](#_Toc189753512)

[4.1 Реализация проекта вёрстки сайта (HTML+CSS) 33](#_Toc189753513)

[4.2 Реализация проекта вёрстки сайта с применением React 36](#_Toc189753514)

[5. ТЕСТИРОВАНИЕ И АНАЛИЗ ПРОЕКТА 46](#_Toc189753515)

[5.1 Тестирование на различных устройствах и браузерах 46](#_Toc189753516)

[5.2 Анализ полученных результатов 47](#_Toc189753517)

[ЗАКЛЮЧЕНИЕ 49](#_Toc189753518)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 52](#_Toc189753519)

# ВВЕДЕНИЕ

# Актуальность темы

Создание сайта-портфолио для команды разработчиков в настоящее время приобрело особую актуальность. Это не просто демонстрация выполненных проектов, но и эффективный инструмент для развития бизнеса, напрямую способствующий росту корпоративной репутации и конкурентоспособности на рынке информационных технологий.

В эпоху цифровых технологий, когда практически невозможно представить успешное развитие бизнеса без онлайн-присутствия, наличие профессионально оформленного веб-портфолио становится необходимым условием для достижения успеха. Его важность обусловлена рядом ключевых факторов:

1. **Корпоративная идентичность и презентация компании**

В сфере делового взаимодействия корпоративная идентичность имеет решающее значение для формирования узнаваемости и доверия к организации. Презентация служит действенным инструментом для укрепления этой идентичности и воздействия на клиентов и партнеров. Веб-портфолио представляет собой эффективный способ демонстрации профессиональных навыков и опыта.

Статические документы или простые списки завершенных проектов уже не отвечают современным стандартам. Интерактивное портфолио предоставляет возможность наглядной демонстрации функциональности созданных приложений, веб-сайтов и других разработок. Оно может включать не только скриншоты, но и интерактивные демоверсии, позволяющие потенциальным клиентам оценить качество кода, архитектуру проекта и пользовательский интерфейс.

1. **Использование актуальных технологий**

Грамотно составленное онлайн-портфолио является действенным инструментом продвижения, позволяющим представить не только профессиональные навыки команды специалистов, но и их способность к эффективной презентации достигнутых результатов.

Дизайн портфолио, его структура и удобство навигации играют ключевую роль в формировании первого впечатления о потенциальных партнерах, способных оказать поддержку бизнесу.

Современные платформы, такие как Tilda, WordPress, uCoz, а также индивидуальные разработки на React, Angular или Vue.js, предоставляют широкий спектр возможностей для создания уникального и запоминающегося дизайна, отражающего индивидуальный стиль и профессиональные предпочтения.

Онлайн-портфолио способствует формированию уникального имиджа или бренда, позволяя выразить свою индивидуальность и стиль, что в свою очередь помогает выделиться среди конкурентов.

1. **Адаптивность и кроссплатформенность**

В современном цифровом пространстве пользователи всё чаще пользуются интернет-ресурсами с разнообразных устройств, включая смартфоны, планшеты, ноутбуки и персональные компьютеры. В связи с этим, адаптация веб-портфолио под различные размеры экранов и браузеры приобретает первостепенное значение.

Применение современных методов адаптивной верстки, а также использование CSS-фреймворков, таких как Bootstrap, позволяет обеспечить доступность сайта для более широкого круга пользователей и поддерживать кроссплатформенную совместимость, что в конечном итоге повышает удобство использования и качество взаимодействия пользователя с сайтом.

1. **Динамические и интерактивные элементы**

Современные анимационные библиотеки, такие как React Spring, предоставляют эффективные возможности для создания динамичных и интерактивных веб-сайтов.

React Spring, являясь популярной библиотекой на основе фреймворка React, упрощает процесс разработки сложной анимации с помощью лаконичного синтаксиса. Поддержка различных типов анимации, включая переходы, трансформации и интерполяцию, позволяет создавать увлекательный и интуитивно понятный пользовательский интерфейс.

Внедрение анимации в элементы сайта, такие как слайдеры, прокрутка страницы и переходы между разделами, способствует плавной навигации и повышению уровня взаимодействия пользователя. Динамическая загрузка контента по мере необходимости помогает пользователю сосредоточиться на ключевой информации.

Интерактивные элементы, реализованные с помощью анимации, способствуют удержанию внимания посетителя и стимулируют активное изучение представленных материалов. Визуализация сложных концепций и идей посредством графиков и анимации делает информацию более доступной и понятной.

**5) Простота поддержки и масштабируемость**

Веб-портфолио предоставляет непрерывный доступ к информации о компании для потенциальных клиентов, независимо от их местонахождения и времени суток. Интеграция с социальными сетями и наличие контактной информации упрощает взаимодействие с компанией и способствует получению обратной связи. Анализ трафика на портфолио с помощью специальных инструментов позволяет оценить его эффективность и внести необходимые изменения в контент и дизайн.

**6) SEO-оптимизация и доступность в сети**

В условиях стремительного развития цифрового пространства оптимизация сайта для поисковых систем (SEO) приобретает ключевое значение для успешного продвижения компании. Современные технологии веб-разработки предлагают широкий спектр инструментов, направленных на повышение SEO-показателей, таких как использование метатегов, оптимизация скорости загрузки страницы и другие. Реализация этих мер способствует повышению видимости Интернет-ресурса в поисковых системах, что, в свою очередь, приводит к увеличению трафика и расширению клиентской базы.

**7) Подготовка к профессиональной деятельности**

Разработка веб-портфолио с применением современных технологий свидетельствует о наличии навыков решения практических задач и использования востребованных в отрасли инструментов.

**8) Поддержка карьерного роста и профессионального развития**

Чтобы оставаться конкурентоспособным на современном рынке труда, необходимо непрерывно совершенствовать свои профессиональные навыки и следить за появлением новых технологий. Портфолио служит эффективным инструментом для демонстрации карьерного роста и успехов, а также хранения информации о реализованных проектах.

**9) Гибкость и персонализация**

Создание высококачественного веб-портфолио на основе современных технологий является стратегической инвестицией в успех компании. Это не просто модная тенденция, а необходимый инструмент для группы разработчиков, стремящихся к успешной карьере в быстрорастущей сфере информационных технологий. Вложение времени и ресурсов в разработку и регулярное обновление собственного онлайн-портфолио принесет многократную отдачу в виде увеличения числа клиентов и их вовлеченности в сотрудничество.

Предоставление подробной информации о команде разработчиков, включая их опыт, образование и достижения, способствует установлению личного контакта с потенциальными клиентами. Персонализированный подход позволяет потенциальным клиентам лучше понять, как компания может удовлетворить их потребности и решить стоящие перед ними задачи.

# 1.2 Цель и задачи дипломной работы

**Цель дипломной работы**

Дипломная работа посвящена созданию интерактивного и адаптивного веб-портфолио, которое может служить современным и эффективным инструментом презентации для демонстрации корпоративных навыков и реализованных проектов.

Целью проекта является разработка веб-ресурса, отвечающего не только требованиям визуальной привлекательности и удобства использования, но и всем современным стандартам веб-дизайна и разработки. Важным аспектом является обеспечение лёгкости обновления и управления контентом. В рамках проекта планируется разработка веб-сайта для студии профессионалов в области разработки и SEO-оптимизации интернет-ресурсов, адаптированного под различные устройства (персональные компьютеры, планшетные компьютеры, смартфоны) с использованием современных технологий, таких как HTML5, CSS3 и JavaScript.

**Задачи дипломной работы**

Для достижения поставленной цели необходимо решить следующие задачи:

1. Анализ современных веб-технологий и их практическое применение:
   * Проведение комплексного анализа современных инструментов и технологий, используемых в разработке веб-портфолио.
   * Определение оптимального набора технологий для реализации проекта, с учетом интеграции HTML5, CSS3, JavaScript, фреймворка React и дополнительных библиотек, предназначенных для анимации и визуализации данных
2. Разработка архитектуры и пользовательского интерфейса для веб-портфолио:
   * Разработать информационную архитектуру и структуру веб-портфолио.
   * Необходимо разработать дизайн пользовательского интерфейса с учётом принципов адаптивности и обеспечения доступности контента для пользователей различных устройств.
3. Разработка адаптивного дизайна и верстки основе React:
   * Необходимо выполнить верстку веб-страницы, используя HTML5 и CSS3, с обеспечением адаптивности интерфейса для корректного отображения на различных устройствах (персональные компьютеры, планшетные компьютеры, мобильные телефоны)
   * Разработать основные компоненты сайта с использованием библиотеки React, а также реализовать взаимодействие между компонентами и управление состоянием приложения
4. Оптимизация и тестирование веб-портфолио:
   * Необходимо провести оптимизацию кода и ресурсов с целью повышения производительности загрузки веб-страниц.
   * Провести тестирование для выявления и исправления потенциальных ошибок, и обеспечения совместимости с различными браузерами, и устройствами.
5. Документирование процесса разработки:
   * Детально описать все этапы создания проекта, охватывающие проектирование, разработку и тестирование.
   * Создание исчерпывающей документации, которая должна содержать описание применяемых технологий, исходный код проекта и иллюстрации, визуализирующие процесс разработки.
6. Оценка и анализ полученных результатов:
   * Произвести всесторонний анализ результатов функционирования проекта, с акцентом на выявление его достоинств и недостатков.
   * Определить возможности дальнейшего совершенствования и развития веб-портфолио.

Решение поставленных задач позволит разработать высококачественный продукт, демонстрирующий не только теоретическую подготовку исполнителя, но и его способность применять полученные знания и навыки для решения практических задач в сфере веб-разработки.

Исследование основано на гипотезе, согласно которой применение технологий, таких как HTML5, CSS3, React, и Bootstrap, способствует созданию продукта с высоким уровнем функциональности и визуальной привлекательности.

# 1.3 Краткое описание проекта

Этот проект представляет собой онлайн-портфолио студии веб-дизайна [Mosh.su](https://mosh.su/), созданное с использованием передовых технологий. Основной целью портфолио является демонстрация профессиональных навыков студии и реализованных ею проектов в области веб-разработки.

Портфолио не просто предоставляет информацию о выполненных работах, но и служит платформой для самовыражения и креативного подхода к дизайну и функциональности.

Разработка веб-портфолио основана на HTML5, CSS3 и JavaScript. Важное значение уделено структуре сайта, которая обеспечивает удобную навигацию и поиск интересующей информации. Портфолио включает в себя описание нескольких проектов, каждый из которых реализован с использованием различных технологий.

Размещение проекта в интернете гарантирует его доступность из любой точки мира. Адаптивный дизайн сайта обеспечивает корректное отображение на различных устройствах и браузерах, что расширяет аудиторию пользователей.

# 2. ОБЗОР ИСПОЛЬЗУЕМЫХ ТЕХНОЛОГИЙ

# 2.1 HTML5, CSS3, SaSS

**HTML5** (HyperText Markup Language, version 5) — [язык для структурирования и представления](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%82%D0%BA%D0%B8) содержимого всемирной паутины. Это пятая версия [HTML](https://ru.wikipedia.org/wiki/HTML). Хотя стандарт был завершён (рекомендованная версия к использованию) только в 2014 году (предыдущая, четвёртая, версия опубликована в 1999 году), уже с 2013 года браузерами оперативно осуществлялась поддержка, а разработчиками — использование рабочего стандарта ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) HTML Living Standard). Цель разработки HTML5 — улучшение уровня поддержки [мультимедиа](https://ru.wikipedia.org/wiki/%D0%9C%D1%83%D0%BB%D1%8C%D1%82%D0%B8%D0%BC%D0%B5%D0%B4%D0%B8%D0%B0)-технологий с одновременным сохранением обратной совместимости, удобочитаемости кода для человека и простоты анализа для [парсеров](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7). В HTML5 реализовано множество новых синтаксических особенностей. Например, элементы <video>, <audio> и <canvas>, а также возможность использования [SVG](https://ru.wikipedia.org/wiki/SVG) и [математических формул](https://ru.wikipedia.org/wiki/MathML). Эти новшества разработаны для упрощения создания и управления графическими и мультимедийными объектами в сети без необходимости использования сторонних [API](https://ru.wikipedia.org/wiki/API) и [плагинов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D0%B8%D0%BD). Другие новые элементы, такие как <section>, <article>, <header> и <nav>, разработаны для того, чтобы обогащать [семантическое](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D0%BC%D0%B0%D0%BD%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B2%D1%91%D1%80%D1%81%D1%82%D0%BA%D0%B0) содержимое документа (страницы). Новые атрибуты были введены с той же целью, хотя ряд элементов и атрибутов был удалён. Некоторые элементы, например, <a>, <menu> и <cite>, были изменены, переопределены или стандартизированы. API и [DOM](https://ru.wikipedia.org/wiki/Document_Object_Model) стали основными частями спецификации HTML5. HTML5 также определяет некоторые особенности обработки ошибок вёрстки, поэтому синтаксические ошибки должны рассматриваться одинаково всеми совместимыми браузерами.

**CSS3** (Cascading Style Sheets3) — третье поколение стандарта CSS, которое находится в активной разработке. Это не просто стандарт, а «набор технологий».

В отличие от предыдущих версий, спецификация разбита на модули, разработка и развитие которых идут независимо.

Некоторые из ключевых модулей CSS3:

* блочная модель;
* значения изображений и заменённый контент;
* текстовые эффекты;
* фоновые изображения и границы;
* анимации;
* пользовательский интерфейс (UI);
* макет с несколькими колонками;
* 2D/3D-преобразования.

CSS3 используется с HTML для создания и форматирования структуры контента. Он отвечает за цвета, свойства шрифтов, выравнивание текста, фоновые изображения, графику, таблицы и т. д.

**Sass (Syntactically Awesome Style Sheets)**  
Sass – это один из самых популярных препроцессоров CSS, предназначенный для упрощения и улучшения процесса написания стилей. Он добавляет в CSS функциональные возможности, такие как переменные, вложенность, миксины и функции, которые упрощают управление стилями и делают их более структурированными и легко поддерживаемыми:

* **переменные** позволяют хранить значения стилей, такие как цвета, размеры шрифтов и отступы, в одном месте, что упрощает их повторное использование и изменение;
* **вложенность** дает возможность писать стили в контексте родительских элементов, что улучшает читаемость кода и соответствует структуре HTML;
* **миксины** – это повторно используемые блоки кода, которые можно вызывать в разных частях стилей. Это позволяет сократить дублирование и упростить создание адаптивных компонентов.
* **импорт и частичные файлы**: Sass поддерживает разбиение стилей на модули и их последующее объединение, что упрощает структуру проекта и управление кодом.

Sass компилируется в обычный CSS, что позволяет использовать все преимущества препроцессора без потери совместимости с браузерами

# 2.2 JavaScript

JavaScript (JS) — мультипарадигменный [язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F). Поддерживает [объектно-ориентированный](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [императивный](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и [функциональный](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) стили. Является реализацией спецификации ECMAScript (стандарт ECMA-262).

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам [приложений](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0). Наиболее широкое применение находит в [браузерах](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) как [язык сценариев](https://ru.wikipedia.org/wiki/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) для придания [интерактивности](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D1%81%D1%82%D1%8C) [веб-страницам](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0).

Основные архитектурные черты: [динамическая типизация](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [слабая типизация](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%B0%D0%B1%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [автоматическое управление памятью](https://ru.wikipedia.org/wiki/%D0%A1%D0%B1%D0%BE%D1%80%D0%BA%D0%B0_%D0%BC%D1%83%D1%81%D0%BE%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [прототипное программирование](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D1%82%D0%B8%D0%BF%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), функции как [объекты первого класса](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82_%D0%BF%D0%B5%D1%80%D0%B2%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0).

На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на [Java](https://ru.wikipedia.org/wiki/Java).

Язык поддерживается всеми современными веб-браузерами и является неотъемлемой частью стека веб-технологий наряду с HTML и CSS. Основная задача JavaScript – взаимодействие с пользователем и изменение содержимого веб-страниц без перезагрузки, что делает его важным инструментом для разработки интерфейсов, веб-приложений и игр.  
JavaScript поддерживает как императивный, так и функциональный стили программирования и имеет множество встроенных возможностей, таких как:

* манипуляция DOM (Document Object Model): JavaScript позволяет изменять структуру, стиль и содержимое HTML-страниц, что делает его идеальным для создания динамических веб-приложений;
* обработка событий: позволяет реагировать на действия пользователя (нажатия кнопок, ввод текста, движения мыши и т.д.) и выполнять определенные действия;
* асинхронное программирование: с помощью таймеров, коллбеков и Promises JavaScript поддерживает выполнение асинхронных операций, таких как запросы к серверу и обработка ответов;
* поддержка модулей и пакетов: JavaScript может использоваться в связке с модулями и пакетами, например, при работе с Node.js для серверной разработки или при использовании модульных систем в клиентской разработке;

# 2.3 React: описание и применение

React (ReactJS) — JavaScript-[библиотека](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_JavaScript) с [открытым исходным кодом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) для разработки [пользовательских интерфейсов](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F).

React разрабатывается и поддерживается Facebook, Instagram и сообществом отдельных разработчиков и корпораций.

React может использоваться для разработки [одностраничных](https://ru.wikipedia.org/wiki/%D0%9E%D0%B4%D0%BD%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%87%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) и мобильных приложений. Его цель — предоставить высокую скорость разработки, простоту и [масштабируемость](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%88%D1%82%D0%B0%D0%B1%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D0%BE%D1%81%D1%82%D1%8C). В качестве библиотеки для разработки пользовательских интерфейсов React часто используется с другими библиотеками, такими как MobX, Redux и GraphQL.

**Основные характеристики и возможности React:**

1. Компоненты: React основан на использовании компонентов – автономных и многократно используемых частей интерфейса. Каждый компонент представляет собой отдельный блок, который может включать в себя как визуальную часть (HTML-разметка), так и логику (JavaScript-функции и состояния). Компоненты могут быть функциональными (основанными на функциях) или классовыми (основанными на классах).
2. JSX (JavaScript XML): JSX – это расширение синтаксиса JavaScript, которое позволяет писать разметку (похожие на HTML конструкции) прямо в коде JavaScript. Благодаря JSX разработчики могут легко описывать структуру интерфейса, используя привычный HTML-подобный синтаксис, и при этом использовать мощь JavaScript.
3. Однонаправленный поток данных (props): в React данные передаются от родительских компонентов к дочерним через свойства (props). Это обеспечивает четкую и предсказуемую структуру передачи данных, упрощая отладку и управление состоянием приложения.
4. Состояние компонента (state): React позволяет управлять состоянием компонентов с помощью useState в функциональных компонентах и свойства state в классовых компонентах. Состояние – это объект, который хранит данные, влияющие на внешний вид и поведение компонента. При изменении состояния React автоматически перерисовывает компонент, обеспечивая синхронизацию данных и интерфейса.
5. Виртуальный DOM (Virtual DOM): React использует концепцию виртуального DOM для повышения производительности. Виртуальный DOM – это облегченная копия реального DOM, которая хранится в памяти. При изменении состояния компонента React обновляет виртуальный DOM и вычисляет разницу с предыдущим состоянием, а затем минимально обновляет реальный DOM, что значительно ускоряет рендеринг интерфейсов.
6. Роутинг и навигация: Для создания многостраничных приложений в React используется библиотека React Router, которая позволяет реализовать навигацию по различным маршрутам (URL) без перезагрузки страницы. Это особенно полезно для одностраничных приложений, где разные компоненты отображаются в зависимости от текущего URL.
7. Контекст и глобальное состояние: React предоставляет возможность создавать глобальные состояния с помощью Context API или сторонних библиотек, таких как Redux или MobX. Это позволяет управлять состоянием на уровне всего приложения и передавать данные между компонентами, которые не связаны между собой напрямую.
8. React Hooks: Начиная с версии 16.8, React представил хуки (Hooks) – функции, которые позволяют использовать состояние и другие возможности React в функциональных компонентах. Основные хуки включают:
   * useState – для управления локальным состоянием.
   * useEffect – для выполнения побочных эффектов (например, загрузка данных).
   * useContext – для доступа к данным контекста.

Хуки делают код компонентов более лаконичным и гибким, избавляя от необходимости использования классовых компонентов.

**Применение React:**

React используется для создания интерактивных пользовательских интерфейсов с богатым функционалом. Он подходит для разработки различных типов веб-приложений, таких как:

* Одностраничные приложения (Single Page Applications, SPA): React позволяет создавать приложения, в которых пользователь взаимодействует с одним HTML-документом, а изменение контента происходит без перезагрузки страницы. Это обеспечивает более быстрый отклик и плавный пользовательский опыт.
* Административные панели и информационные панели (Dashboards): React часто используется для создания интерактивных и динамических интерфейсов админ-панелей, где важна быстрая реакция на изменения данных.
* Веб-сервисы и социальные сети: React применяют для создания интерфейсов с интенсивной работой с данными, например, в социальных сетях или сервисах для работы с контентом, таких как Facebook или Instagram.
* E-commerce приложения: React активно используется для разработки интернет-магазинов, где важна быстрая работа с каталогами товаров, корзинами и системой оформления заказов.
* Мобильные приложения: с помощью React Native разработчики могут создавать мобильные приложения на основе React, что позволяет использовать единый стек технологий для веб- и мобильной разработки.

React – мощная и гибкая библиотека, которая позволяет создавать современные, производительные и масштабируемые веб-приложения, обеспечивая комфортную разработку и удобное управление состоянием интерфейса.

# 2.4 Bootstrap: Фреймворк для визуализации данных

**Bootstrap –** это популярный фронтенд-фреймворк с открытым исходным кодом, предназначенный для быстрой и удобной разработки адаптивных и кросс-браузерных веб-сайтов и веб-приложений. Он был разработан командой Twitter и впервые выпущен в 2011 году. Bootstrap предлагает широкий набор готовых компонентов, таких как кнопки, формы, таблицы, модальные окна, навигационные панели и многое другое, что позволяет существенно ускорить процесс разработки интерфейсов.

Основное преимущество Bootstrap заключается в том, что он предоставляет разработчикам удобный и понятный инструментарий для создания стилей и компонентов интерфейса с минимальными усилиями и без необходимости написания большого количества CSS-кода. Используя готовые стили и классы Bootstrap, можно легко построить современный и визуально привлекательный интерфейс.

**Основные особенности Bootstrap:**

1. Сетка (Grid System): Bootstrap предоставляет адаптивную систему сеток (grid system), которая позволяет создавать гибкие и адаптивные макеты. Сетка основана на flexbox, что обеспечивает поддержку различных типов макетов, включая:
   * Контейнеры (containers): основа для выравнивания содержимого на странице.
   * Ряды (rows): строки, которые служат для размещения столбцов.
   * Столбцы (columns): отдельные блоки, которые можно объединять или делить на ячейки.

Сетка Bootstrap поддерживает различные размеры экранов (sm, md, lg, xl, xxl), что позволяет создать адаптивные интерфейсы, автоматически подстраивающиеся под размер устройства (мобильные телефоны, планшеты, десктопы и т.д.).

1. Компоненты: Bootstrap включает более 20 готовых компонентов, которые помогают строить интерфейсы с минимальными усилиями:
   * Кнопки (btn): кнопки различных стилей, размеров и с эффектами наведения.
   * Формы (form-group): элементы для создания форм, включая поля ввода, флажки, радиокнопки и селекторы.
   * Навигация (navbar, nav): компоненты для создания навигационных панелей, меню и хлебных крошек.
   * Карточки (card): для отображения содержимого в структурированных блоках.
   * Модальные окна (modal): всплывающие окна для отображения диалогов или дополнительных сведений.

Эти компоненты можно легко модифицировать с помощью предустановленных классов и атрибутов, а также комбинировать друг с другом для создания сложных интерфейсов.

1. Тема и стилизация: Bootstrap включает готовые стили для типографики, кнопок, форм и других элементов интерфейса, что позволяет добиться единого и современного внешнего вида приложения без дополнительных усилий. Все стили можно переопределить с помощью CSS или подключить собственные темы.
2. Интерактивные компоненты и JavaScript плагины: в состав Bootstrap входят JavaScript-плагины, основанные на jQuery, которые добавляют интерактивность компонентам. Например:
   * Модальные окна (modal): всплывающие окна для отображения дополнительной информации.
   * Слайдеры и карусели (carousel): автоматическая прокрутка изображений или контента.
   * Выпадающие меню (dropdown): интерактивные списки и меню.
   * Всплывающие подсказки и popovers: информационные всплывающие элементы.
3. Адаптивность и кросс-браузерность: Bootstrap изначально разработан для создания адаптивных макетов, поддерживающих отображение на экранах различных размеров. Адаптивность достигается за счет медиа-запросов и гибкой сетки, что обеспечивает корректное отображение интерфейса как на мобильных устройствах, так и на больших экранах.
4. Иконки: Начиная с Bootstrap 5, фреймворк поддерживает набор иконок Bootstrap Icons, которые можно использовать в веб-приложениях для улучшения визуального восприятия и повышения читаемости интерфейсов.

**Применение Bootstrap:**

Bootstrap используется в самых разных проектах – от небольших лендингов и блогов до крупных корпоративных порталов и e-commerce сайтов. Благодаря своей простоте и гибкости, Bootstrap стал одним из наиболее распространенных фреймворков для разработки UI, что делает его идеальным выбором для создания прототипов и MVP (Minimum Viable Product).

**Преимущество Bootstrap:**

* Быстрый старт: для использования Bootstrap достаточно подключить CSS и JavaScript файлы фреймворка и сразу приступить к разработке интерфейса.
* Адаптивность: встроенная поддержка адаптивных макетов упрощает создание кроссплатформенных интерфейсов.
* Сообщество и документация: обширная документация и большое сообщество разработчиков предоставляют множество готовых решений и примеров кода.
* Поддержка тем и кастомизация: разработчики могут легко менять стили компонентов или создавать свои темы с помощью переменных Sass.

Bootstrap – это мощный и гибкий фреймворк, который упрощает процесс создания современных веб-приложений, обеспечивая адаптивность, кроссплатформеность и интуитивно понятную стилизацию интерфейсов. Использование Bootstrap позволяет сосредоточиться на функциональности приложения, а не на разработке базовых стилей и компонентов, что существенно ускоряет разработку и повышает качество интерфейсов.

# 2.5 React Spring — это JavaScript-библиотека для создания интерактивных анимаций

React Spring — это JavaScript-библиотека для создания интерактивных анимаций в React-приложениях. В отличие от традиционной анимации на основе CSS или других библиотек анимации React, React Spring использует анимацию на основе физики, которая имитирует реальные движения и создаёт более естественный эффект.

Эти анимации можно применить к любому свойству React-компонентов, включая положение, масштаб, непрозрачность и многое другое. Это делает его мощным инструментом для разработчиков, желающих улучшить пользовательский опыт React-приложений с помощью увлекательных анимаций.

Настройка React Spring в проекте

Чтобы анимировать компоненты в React проекте с помощью React Spring, нужно выполнить следующие шаги:

Скачать и установить библиотеку React Spring с помощью npm или yarn:

npm install react-spring

yarn add react-spring

Эти команды установят библиотеку **react-*spring*** и её зависимости в каталог проекта.

После установки React Spring нужно импортировать необходимые компоненты и хуки в компоненты React, чтобы начать анимировать элементы. Это можно сделать с помощью следующего синтаксиса:

import { animated, (hook) } from 'react-spring'

В приведённом выше фрагменте кода мы импортируем две зависимости (хук и анимацию) из библиотеки React Spring. Вот подробное описание того, как работает каждый из них, и почему их необходимо импортировать.

#### Animated:

В React Spring анимированное (animated) пространство имён предоставляет набор компонентов, которые позволяют анимировать элементы в React-приложении. Эти компоненты представляют анимированные версии стандартных HTML элементов, таких как <div>, <span>и <img>. Мы можем использовать эти анимированные элементы вместо обычных HTML элементов и применять к ним анимации с помощью хуков анимации React Spring.

#### Хуки:

Для создания анимации в React-компонентах React Spring предлагает несколько хуков. Они упрощают процесс управления анимацией и позволяют легко интегрировать их в компоненты. Вот некоторые из них:

* **useSpring**. Используется в большинстве случаев, так как создаёт одну пружинную анимацию, которая изменяет данные из начального состояния в другое.
* **useTransition**. Анимирует добавление, удаление или переупорядочивание элементов списка. Она управляет жизненным циклом анимации элементов, когда они входят или выходят из DOM, обеспечивая плавные переходы между различными состояниями списка.
* **useTrail**. Используется для создания нескольких пружинных анимаций, создающих эффект «следа», когда каждая пружина следует за предыдущей или отстаёт от неё.
* **useChain**. Как и цепочка, используется для определения последовательности анимации с указанием порядка их следования.
* **useSprings**. Хотя это похоже на useSpring, useSprings используется для управления несколькими пружинными анимациями одновременно, в то время как useSpring управляет одной пружинной анимацией.

Чтобы лучше понять, как они работают, давайте рассмотрим различные стили анимации, которых можно добиться с помощью каждого из этих хуков.

**React Spring** — это мощная библиотека анимации, которая позволяет создавать потрясающие интерактивные анимации в React-приложениях. Как мы увидели, эти анимации могут применяться к различным элементам в проектах.

Используя возможности React Spring, мы можем добиться более плавных переходов с более естественными эффектами, а также получить больший контроль над анимацией.

**2.6 Инструменты и среда для разработки**

Современная веб-разработка требует использования инструментов, которые обеспечивают удобную среду разработки. В этой секции мы рассмотрим один из популярных [текстовых редактор](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D0%B0)ов Visual Studio Code. Данный инструмент играет важную роль в процессе создания и редактирования, предоставляя разработчикам мощные возможности для быстрой разработки с огромным количеством дополнений и различных плагинов, облегчающих разработку веб-проектов.

Visual Studio Code (VS Code) — [текстовый редактор](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D0%B0), разработанный Microsoft для [Windows](https://ru.wikipedia.org/wiki/Windows), Linux и macOS. Позиционируется как «лёгкий» редактор кода для [кроссплатформенной](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%BE%D1%81%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D1%8C) разработки [веб](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5)- и [облачных](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BB%D0%B0%D1%87%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F) приложений. Включает в себя [отладчик](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BB%D0%B0%D0%B4%D1%87%D0%B8%D0%BA), инструменты для работы с Git, [подсветку синтаксиса](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D1%81%D0%B2%D0%B5%D1%82%D0%BA%D0%B0_%D1%81%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81%D0%B0), IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, [сочетания клавиш](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%87%D0%B5%D1%82%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BA%D0%BB%D0%B0%D0%B2%D0%B8%D1%88) и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с [открытым исходным кодом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), но готовые сборки распространяются под проприетарной лицензией.

Visual Studio Code основан на Electron и реализуется через веб-редактор Monaco, разработанный для Visual Studio Online.

**Возможности**

Visual Studio Code — это редактор исходного кода. Он имеет многоязычный интерфейс пользователя и поддерживает ряд [языков программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), подсветку синтаксиса, IntelliSense, рефакторинг, [отладку](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BB%D0%B0%D0%B4%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D1%8B), навигацию по коду, поддержку Git и другие возможности. Многие возможности Visual Studio Code недоступны через [графический интерфейс](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F), зачастую они используются через палитру команд или JSON-файлы (например, пользовательские настройки). Палитра команд представляет собой подобие [командной строки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%BE%D0%B9_%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B8), которая вызывается [сочетанием клавиш](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%87%D0%B5%D1%82%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BA%D0%BB%D0%B0%D0%B2%D0%B8%D1%88).

VS Code также позволяет заменять [кодовую страницу](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0) при сохранении документа, [символы перевода строки](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%B2%D0%BE%D0%B4_%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B8) и язык программирования текущего документа.

С 2018 года появилось расширение Python для Visual Studio Code с открытым исходным кодом. Оно предоставляет разработчикам широкие возможности для редактирования, отладки и тестирования кода.

Также VS Code поддерживает редактирование и выполнение файлов типа «Блокнот Jupyter» (Jupyter Notebook) напрямую «из коробки» без установки внешнего модуля в режиме визуального редактирования и в режиме редактирования исходного кода.

На март 2019 года посредством встроенного в продукт пользовательского интерфейса можно загрузить и установить несколько тысяч расширений только в категории «programming languages» (языки программирования).

Также расширения позволяют получить более удобный доступ к программам, таким как Docker, Git и другие. В расширениях можно найти линтеры кода, темы для редактора и поддержку синтаксиса отдельных языков.

# 2.7 Хостинг для размещения веб проекта в сети.

Хостинг для сайта — услуга, позволяющая арендовать необходимые ресурсы на сервере и разместить на них свой сайт, веб⁠-⁠приложение или другой онлайн⁠-⁠контент.

Выбор надёжного хостинга важен, поскольку хостинг — основа для создания сайтов и запуска онлайн⁠-⁠проектов. Когда вы размещаете файлы на удалённом веб⁠-⁠сервере, провайдер отвечает за их сохранность и безопасность. В отличие от free⁠-⁠сервисов, качественный веб⁠-⁠хостинг гарантирует быстрый доступ к страницам для большого числа посетителей без лишней рекламы.

Выбор пал на Рег.ру. Технические характеристики web⁠-⁠хостинга включают высокопроизводительный SSD RAID, Linux CentOS c поддержкой Python, PHP, Perl и фреймворка Django. К ftp серверу доступ осуществляется по протоколам FTP и SSH.

Данный хостинг предоставляет дополнительные услуги в сфере безопасности такие как:

* — защиту от DDoS⁠-⁠атак;
* — безлимитный трафик;
* — двойную антивирусную проверку;
* — SSL⁠-⁠сертификат.

В этом хостинге есть все нужное и даже больше: поддержка 24/ 7, удобный и функциональный личный кабинет, и панель управления с различными дополнительными услугами как для бизнеса, так и для простых пользователей сети интернет.

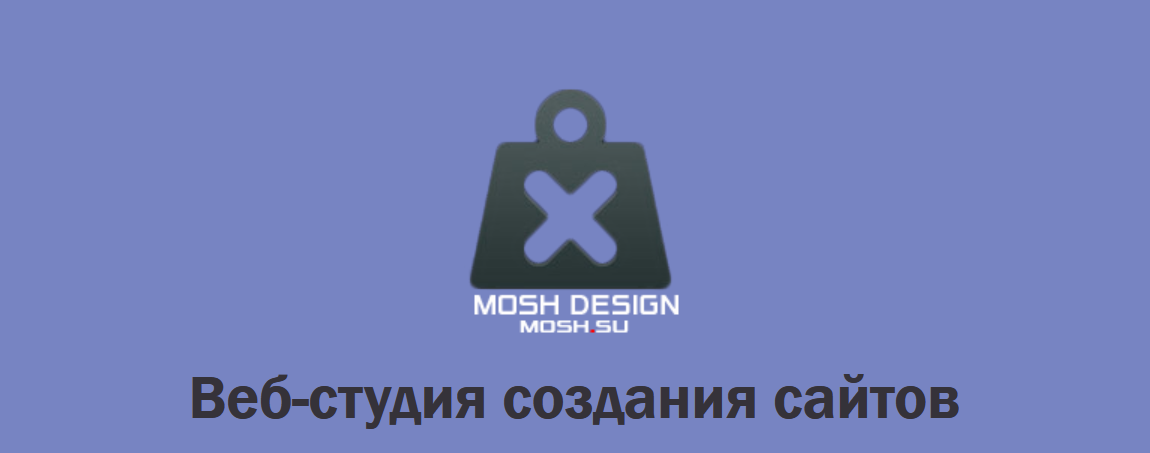
# 3. ПРОЕКТИРОВАНИЕ ВЕБ-ПОРТФОЛИО

# 3.1 Структура веб-портфолио

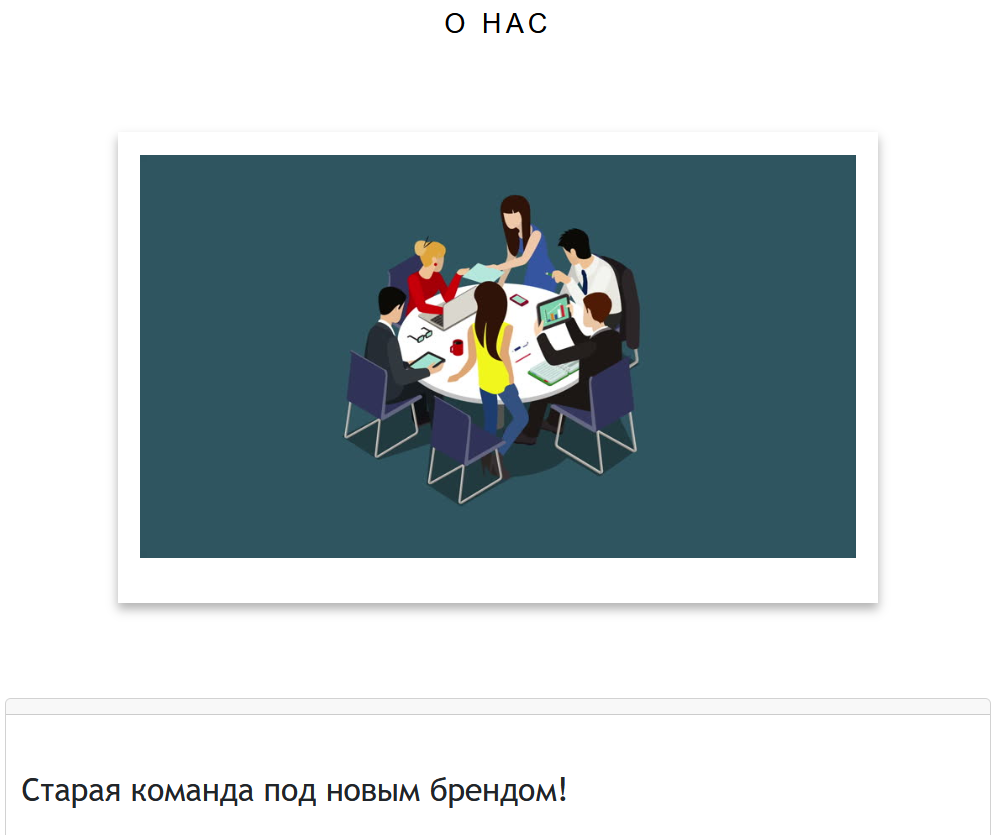
[Портфолио](https://mosh.su/) представляет собой одностраничный веб-сайт, организованный по тематическим разделам, предназначенный для презентации профессиональных компетенций, завершённых проектов и достигнутых показателей. При его разработке были применены современные дизайнерские подходы и технологические решения, гарантирующие интерактивность интерфейса и визуально-эстетическую привлекательность.

**Основная структура веб-портфолио**

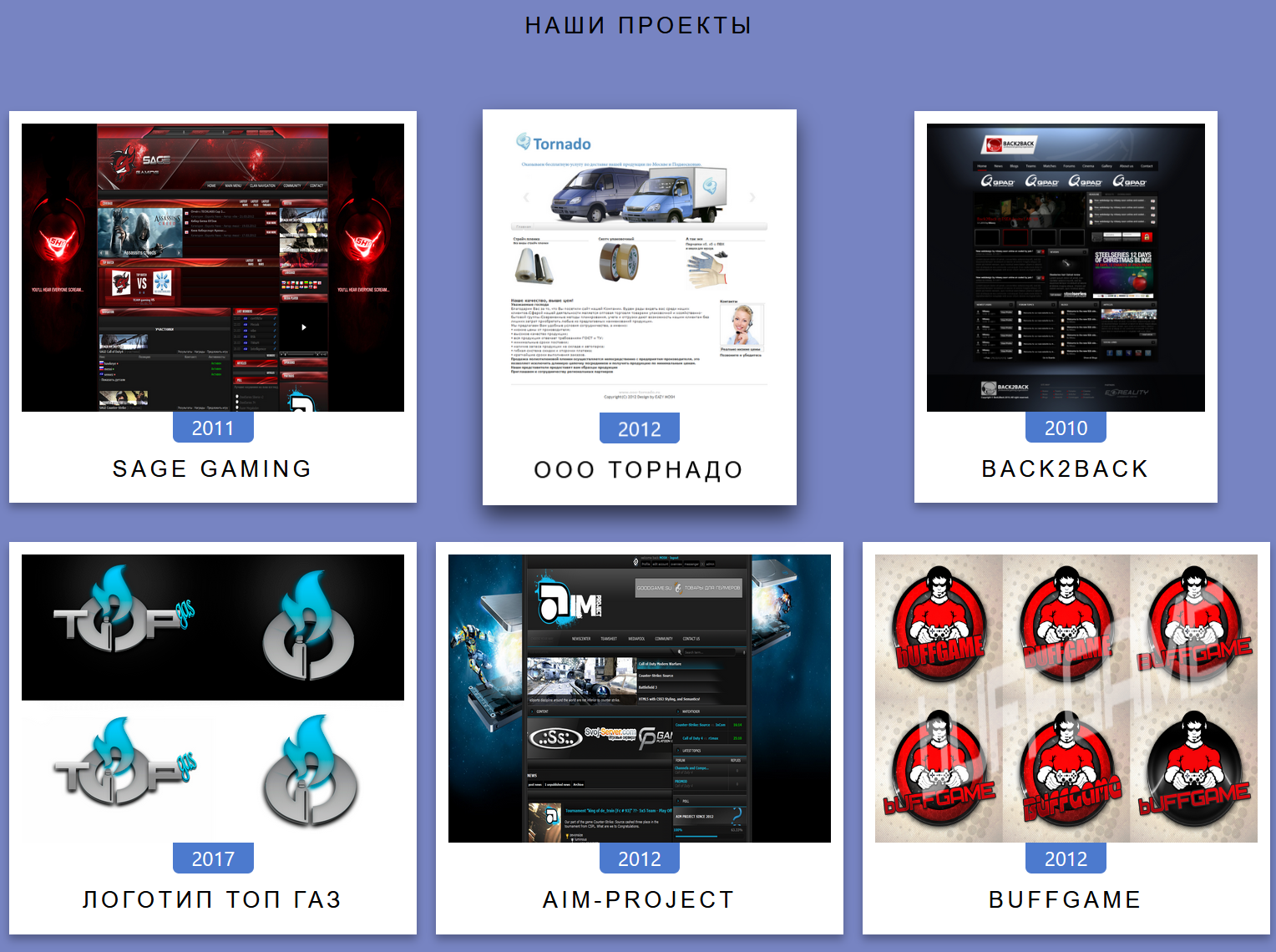
1. Главная страница:

На главной странице ресурса присутствует название компании, что способствует оперативной идентификации пользователем сферы профессиональной деятельности.

1. О нас:

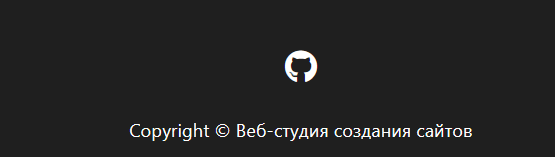
Раздел посвящён профессиональной деятельности компании. В нём содержится сжатое изложение корпоративных качеств, принципов работы и сферы интересов. 

1. Работы:

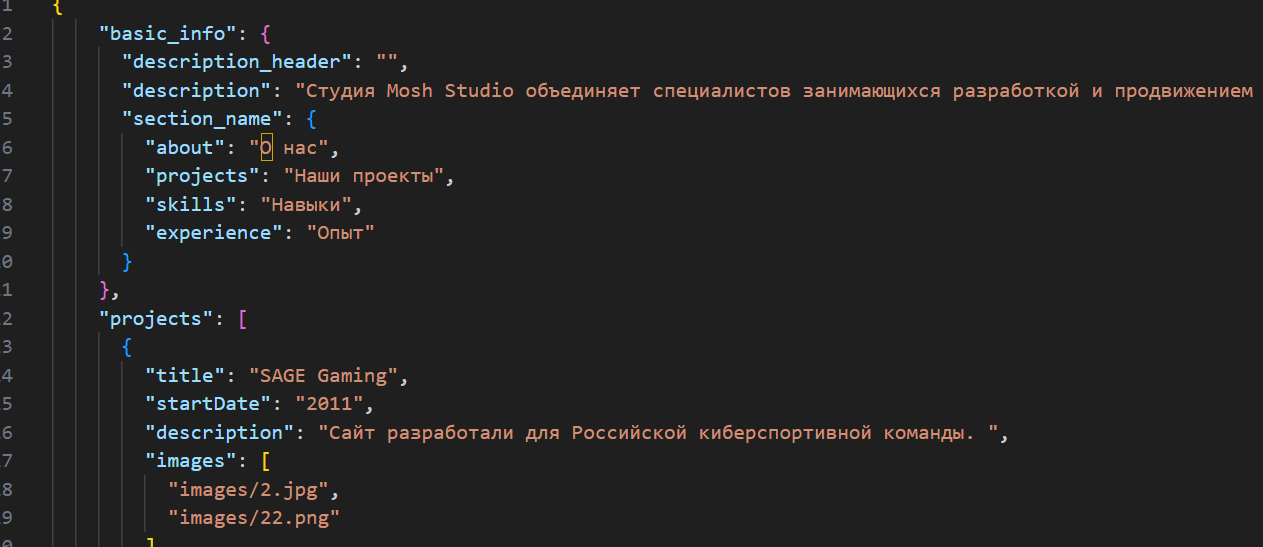
* Данный раздел содержит портфолио, демонстрирующее разнообразие выполненных проектов, охватывающих как самостоятельные проекты, так и совместные работы.

1. Контакты:

* В «подвале» сайта кликабельное лого GitHub’а для связи и Copyright внизу сайта означает, что все материалы, выложенные на сайте, принадлежат владельцу сайта, и их использование без разрешения с его стороны запрещено.



1. Контент сайта.

Наполняемость сайта данными реализована с помощью JSON (JavaScript Object Notation).

Данное веб-портфолио отличается продуманной структурой и ясной презентацией информации. Единый стилистический дизайн каждой секции сайта обеспечивает посетителям полное представление о профессиональных компетенциях компании. Привлекательный визуальный ряд делает ресурс запоминающимся и удобным для использования. Применение современных технологий разработки, таких как React и Bootstrap, подчеркивает высокий уровень мастерства команды в области фронтенд-разработки.

# 3.2 Дизайн интерфейса и пользовательского опыта

Веб-портфолио спроектировано в соответствии с современными принципами дизайна пользовательского интерфейса (UI) и опыта (UX).

Интерфейс и UX-дизайн направлены на обеспечение удобного восприятия информации, простоты навигации и эстетически привлекательного внешнего вида. В нижеследующем тексте представлены ключевые аспекты дизайна и применённые в проекте UX-подходы.

**1. UI-дизайн (User Interface Design)** — это процесс создания интерфейсов в программном обеспечении или устройствах с акцентом на внешний вид и стиль.

**Цель UI-дизайна** — сделать взаимодействие пользователя с продуктом максимально удобным и интуитивно понятным. Хороший UI-дизайн помогает пользователю легко находить нужную информацию и выполнять задачи без лишних усилий.

**UI-дизайн включает в себя**:

* **Работу с композицией**. Элементы должны смотреться эстетично и гармонично: текст должен быть читабельным, по кнопкам удобно кликать, а анимация подсказывать пользователям, что нужно сделать.
* **Подбор цветовых сочетаний и шрифтовых пар**. Цвета могут создавать настроение, привлекать внимание к важным элементам и улучшать читаемость контента.
* **Настройку анимации**. Например, баннер на сайте может увеличиваться при наведении курсора, а кнопка — менять цвет.
* **Визуальное оформление кнопок и текстовых блоков**. Кнопки должны быть легко различимы и интуитивно понятны.
* Кроме того, UI-дизайн делает внешний вид сайта узнаваемым и способен передавать атмосферу и настроение бренда.

**2. Пользовательский опыт (UX)** — это опыт, получаемый конечным пользователем в ходе его взаимодействия с интерфейсом сайта, сервиса, продукта или услуги.

**UX-дизайн** фокусируется на изучении потребностей и поведения пользователей, а затем на создании интуитивно понятных продуктов, ориентированных на конкретные цели. Ключевая цель — создать логичный и простой в использовании продукт.

**Основные характеристики пользовательского опыта (UX)**:

* **Юзабилити**. Степень лёгкости решения стоящих перед пользователем задач с помощью программного продукта.
* **Ценность**. Показатель того, насколько продукт оказывается полезным для пользователя, то есть насколько его функциональность и контент соответствуют ожиданиям и требованиям пользователей.
* **Доступность**. Возможность быстрых приобретения, установки, загрузки и применения продукта пользователем, а также надёжность и популярность того или иного бренда.
* **Привлекательность**. Показатель связан с визуальной составляющей продукта: с его дизайном, информационной архитектурой (структурой размещения контента), с отображением и читабельностью текста (типографикой).
* **Цель UX-дизайна** — улучшить степень удовлетворённости пользователя и его лояльность благодаря пользе, простоте использования и удовольствию, которое он получает в процессе взаимодействия с интернет-ресурсом.

# 3.3 Разработка адаптивной верстки

**Разработка адаптивной вёрстки** предполагает создание макета сайта, который сохраняет изначальную концепцию и основную суть. При этом некоторые элементы, выполняющие функцию вспомогательных данных или украшающих деталей, допустимо скрыть или удалить.

**Суть процесса адаптивной вёрстки** заключается в проработке параметров контента — как текста, так и картинки. Для этого используют технологии CSS3 и язык разметки HTML5.

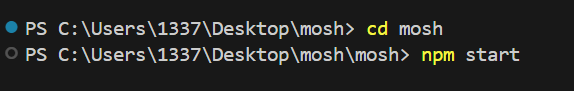
**Некоторые техники адаптивной вёрстки**:

* **Flexible layout (гибкий макет)**. При этой технике фиксированные значения переходят в относительные. Для этого используется формула, определяющая процентное соотношение.
* **Flexible media (гибкое изображение)**. Любой вид медиаконтента адаптируют, применяя свойство {max-width: 100%}. Такой подход способствует изменению размеров картинок, видео и прочих видов медиа в соответствии с изменениями ширины экрана.
* **Media queries (медиазапросы)**. Этот способ адаптирует сайт таким образом, что он при помощи CSS модуля приобретает способность подстраиваться под различные типы устройств. CSS устанавливает взаимосвязь между разрешением экрана и установленными стилями.
* **При разработке адаптивного сайта веб-дизайнеры учитывают** не конечные размеры страницы, а конкретные пределы, превышение которых приводит к изменению формата отображения.
* **Для тестирования адаптивной вёрстки** нужно открыть сайт на разных моделях смартфонов с разными размерами экранов и разрешениями. Для десктопных версий сайт нужно проверить на разных мониторах. Также следует проверить работу анимаций и дополнительных эффектов, контрастность и читаемость текста.

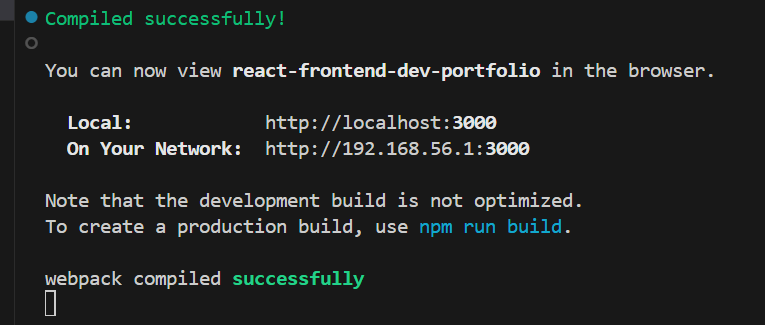
Сайт mosh.su спроектирован с применением адаптивной верстки, которая реализована посредством медийных запросов, гибких сеток и других передовых технологий. Данный подход обеспечивает комфортный доступ к контенту сайта с различных устройств, гарантируя его доступность и читаемость для пользователей независимо от типа используемого устройства.

# Запуск проекта и загрузка его на хостинг.

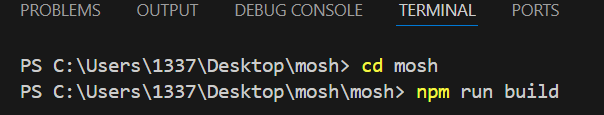
1. Для запуска проекта необходимо открыть консоль в главной директории проекта и выполнить команду `npm start`. После этого следует дождаться завершения процесса инициализации.

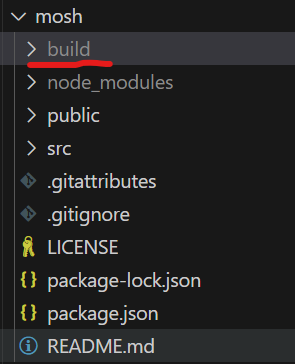


По завершении, автоматически откроется страница в веб-браузере, отображающая запущенный проект.



1. Для подготовки проекта к размещению на хостинге необходимо выполнить следующие действия:

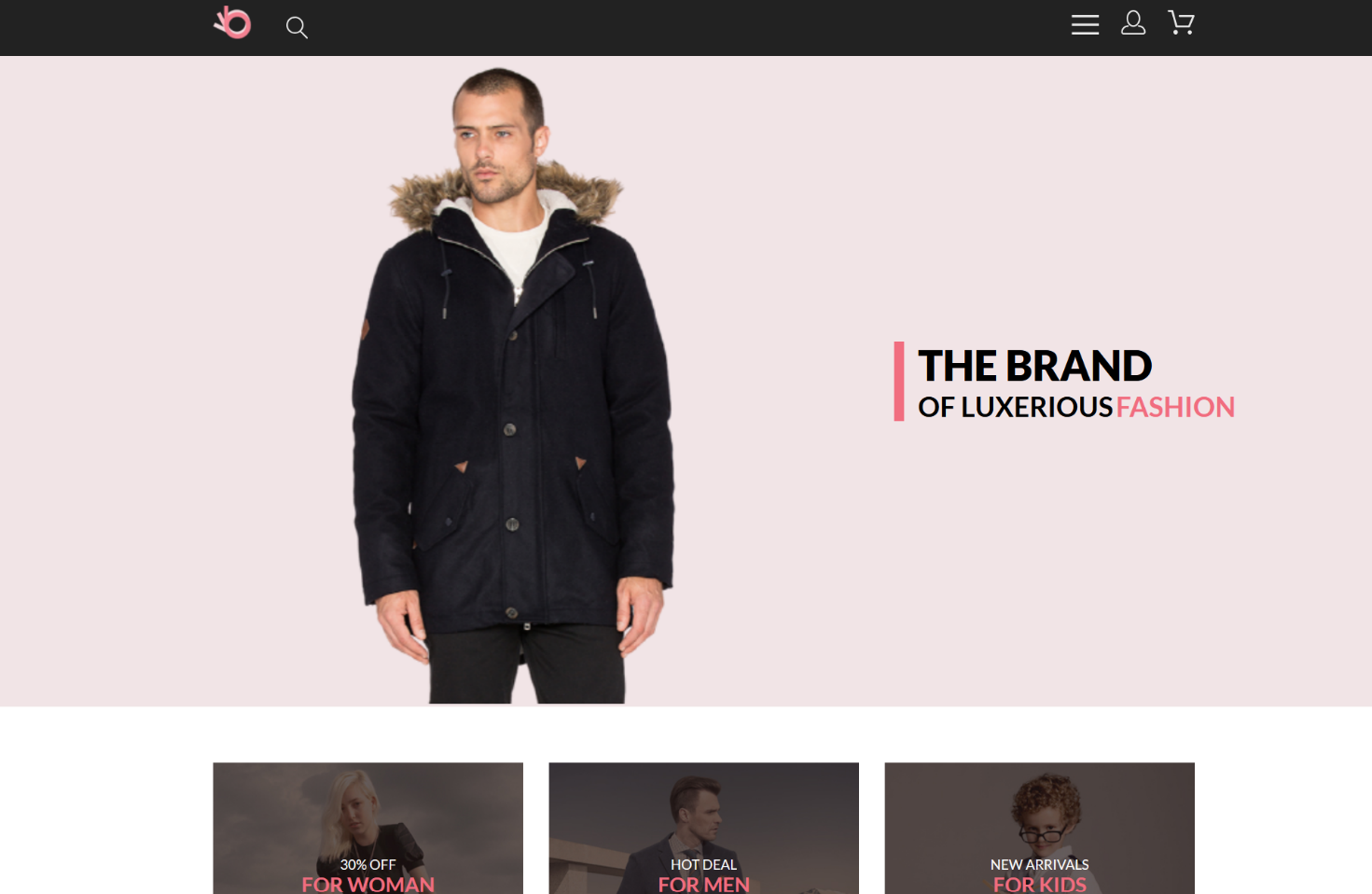
* Откройте консоль в корневой директории проекта. Введите команду `npm build` и дождитесь завершения процесса сборки. 
* По завершении сборки в основной директории проекта появится новая папка с именем "Build".



Содержимое этой папки представляет собой готовый к развертыванию проект, который можно загрузить на хостинг.

# 4. РЕАЛИЗАЦИЯ ПРОЕКТОВ

# 4.1 Реализация проекта вёрстки сайта (HTML+CSS)



Проект [Веб-вёрстка HTML/CSS](https://vladimirmosh.github.io/index.html). Данный проект взят из [Figma](https://www.figma.com/file/mnLY69cYE5cqWM5w6n5hXx/Seo-%26-Digital-Marketing-Landing-Page?node-id=190%3A1194), включает в себя ключевые элементы, такие как заголовки, изображения, кнопки и информационные блоки, которые были точно реализованы в коде. Figma — это онлайн-сервис для дизайнеров, веб-разработчиков и маркетологов. Он предназначен для создания прототипов сайтов или приложений, иллюстраций и векторной графики.

В редакторе можно настроить совместную работу, вносить и обсуждать правки, причём как в браузере, так и через приложение на компьютере.

Figma предоставляет возможность удобной работы с цветами, шрифтами и отступами, что способствует высокой точности в реализации проектов. Каждый из этих элементов был перенесён в код, что обеспечило значительное соответствие оригинальному дизайну. Применение Figma существенно упростило процесс взаимодействия с цветовыми палитрами, типографикой и отступами, что, в свою очередь, способствовало улучшению точности реализации каждого компонента. HTML-код организован с использованием семантических тегов, таких как <header>, <main>, <section>, и <footer>, что улучшает доступность и SEO-оптимизацию страницы.

Включена навигационная панель, позволяющая пользователям быстро перемещаться между различными разделами сайта.

Стили были разработаны с использованием Sass, что дало возможность установить цвета, шрифты и отступы в соответствии с дизайном, представленным в Figma.

**Sass** (Syntactically Awesome Style Sheets) — это метаязык, который упрощает и ускоряет написание CSS-кода. Его часто называют препроцессором CSS — это означает, что Sass имеет свой синтаксис, из которого генерирует CSS-код, понятный любому браузеру.

**Препроцессор Sass помогает**:

* **Сделать CSS-код понятнее и проще**. Его легче масштабировать, обновлять и поддерживать.
* **Расширить функциональность**. С помощью Sass можно использовать CSS-константы, встроенные функции, вложенные правила, примеси (смешанные стили), наследование и так далее.
* **Избежать многократного повторения одинаковых фрагментов кода**. Это экономит время разработчика, уменьшает объём файлов стилей и ускоряет обработку страниц.

Для обеспечения адаптивности были внедрены медиазапросы. Это решение играет ключевую роль в создании комфортных условий для пользователей и улучшении общего пользовательского опыта, так как способствует более быстрому поиску необходимой информации.

Сайт реализует адаптивную верстку, что гарантирует корректное отображение контента на различных устройствах — от мобильных телефонов, планшетов до настольных компьютеров.

Также были использованы основные техники SEO для повышения видимости сайта в поисковых системах, включая применение метатегов.

# 4.2 Реализация проекта вёрстки сайта с применением React

Проект представляет собой реализацию уже готовой верстки Веб-вёрстка HTML/CSS в работу, разработанную с использованием технологий React.

Главные задачи которые должны быть реализованы в проекте следующие:

* Разбить сайт на необходимые компоненты, чтобы исключить дублирование кода, и чтобы можно было легче редактировать проект.
* На странице каталога есть сортировка по размеру (S, XS, L, M) необходимо реализовать данный функционал, к какому именно размеру будет относиться тот или иной товар, не имеет значения.
* Реализовать переключение между страницами с помощью Routing.
* На странице корзины реализовать верный подсчет общей цены товаров (GRAND TOTAL $900) плюс реализовать возможность менять количество товара, это также должно влиять на цену товар (GRAND TOTAL $900).

**Структура проекта и инструменты разработки**

Реализовывать проект будем в программе Visual Studio Code. Создание будет осуществляться среде Node.js.

**Node.js** — это бесплатная кроссплатформенная среда выполнения JavaScript с открытым исходным кодом. Она создана на движке V8 от Google, который используется для преобразования JavaScript в компьютерный код.

**Некоторые задачи, которые решаются в Node.js:**

* **Разработка серверных приложений**. Среда позволяет разрабатывать веб-серверы, способные работать с миллионами одновременных соединений.
* **Асинхронное программирование**. Node.js выполняет код асинхронно, благодаря чему ресурсы сервера расходуются грамотнее, а приложение может обрабатывать множество запросов одновременно.
* **Работа с большими объёмами данных**. За счёт того, что Node.js хорошо справляется с высокой нагрузкой, её часто применяют для создания приложений, где нужна обработка больших объёмов информации.
* **Создание микросервисов**. Архитектура среды позволяет разрабатывать небольшие изолированные сервисы, которые легко масштабировать и изменять.
* **Разработка сетевых приложений**. К примеру, Node.js подходит для разработки чатов, игровых серверов и RESTful API, в которых нужна высокая скорость обработки запросов.
* **Интеграция со сторонними инструментами через API**. С помощью Node.js настраивают взаимодействие сайта с базами данных, фреймворками и библиотеками.

Рассмотрим, как происходит создание React-проекта.

Create React App — удобная среда для изучения React и лучший способ начать создание нового [одностраничного](https://ru.legacy.reactjs.org/docs/glossary.html#single-page-application) приложения на React.

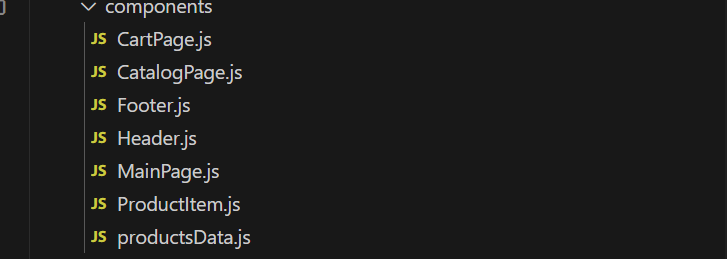
Инструмент настраивает среду для использования новейших возможностей JavaScript, оптимизирует приложение для продакшена и обеспечивает комфорт во время разработки. Для создания проекта введите в консоль следующие команды:

npx create-react-app my-app

cd my-app

npm start

Вместо «my-app» разработчик может задавать любое имя проекта.

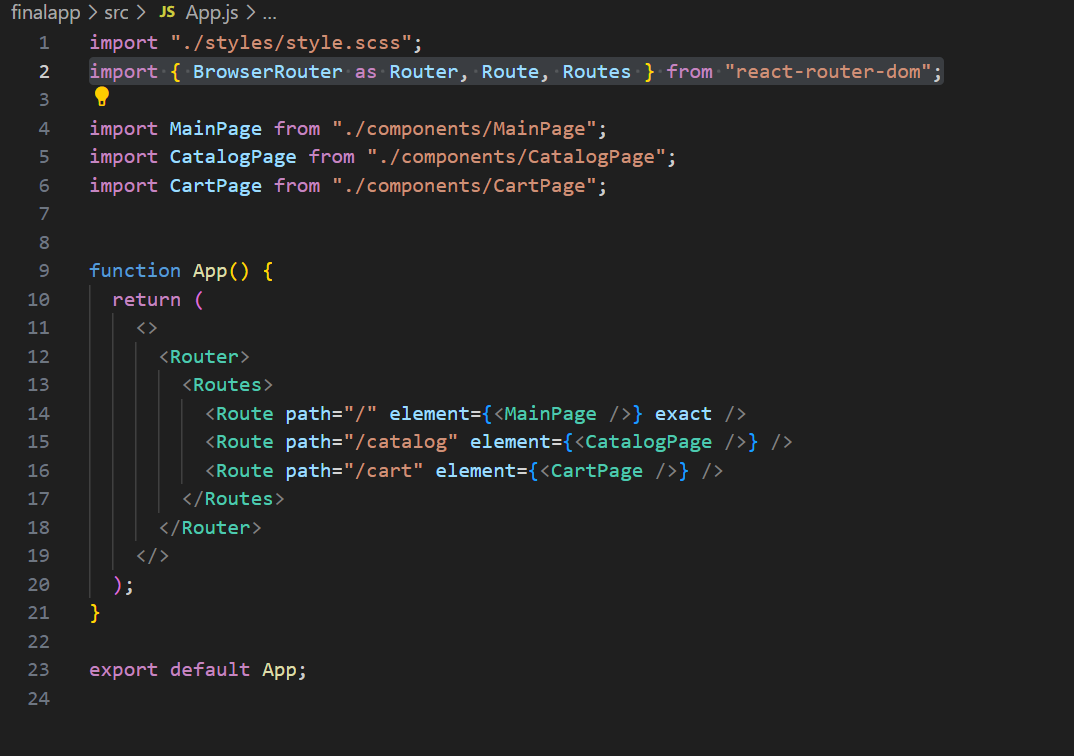


Разбиваем сайт на компоненты путем создания файлов с расширением .js в директории ...\src\components для удобства редактирования. Для реализации переключения между страницами с помощью маршрутизации в React-приложениях можно использовать библиотеку React Router.

**Основные компоненты React Router**:

* **BrowserRouter**. Обёртывает приложение и отвечает за работу маршрутизации. Он отслеживает изменения в URL-адресе и подставляет соответствующий компонент.
* **Routes и Route**. Компонент Routes содержит список всех маршрутов приложения, а Route определяет, какой компонент отображать по определённому пути.
* **Link**. Заменяет стандартные HTML-ссылки (<a>). Вместо того чтобы перезагружать страницу, он обновляет URL и перенаправляет на соответствующий маршрут.

**Чтобы начать работу с React Router**, необходимо установить пакет react-router-dom в проект. Это делается с помощью команды npm install react-router-dom.

Импортируем параметры как показано на скриншоте ниже,

и заключаем в теги наши страницы ранее созданные и импортируемые в проекте.

<Router>

        <Routes>

          <Route path="/" element={<MainPage />} exact />

          <Route path="/catalog" element={<CatalogPage />} />

          <Route path="/cart" element={<CartPage />} />

        </Routes>

      </Router>

Реализовываем сортировку товара:

function pagination(array, pageSize, pageNumber) {

    return array.slice((pageNumber - 1) \* pageSize, pageNumber \* pageSize);

}

function CatalogPage() {

    const [currentPage, setCurrentPage] = useState(1);

    const pageSize = 9;

    const [selectedSizes, setSelectedSizes] = useState([]);

    const handlePageChange = (pageNumber) => {

        setCurrentPage(pageNumber);

    };

    const handleSizeChange = (size) => {

        if (selectedSizes.includes(size)) {

            setSelectedSizes(selectedSizes.filter((s) => s !== size));

        } else {

            setSelectedSizes([...selectedSizes, size]);

        }

    };

    const displayedProducts = pagination(

        products.filter(

            (product) =>

                selectedSizes.length === 0 || selectedSizes.includes(product.size)

        ),

        pageSize,

        currentPage

    );

    const totalPages = Math.ceil(

        products.filter(

            (product) =>

                selectedSizes.length === 0 || selectedSizes.includes(product.size)

        ).length / pageSize

    );

Реализовываем верный подсчет товаров в корзине и их стоимость влияющих на итоговую стоимость следующим образом.

Для начала нам нужен файл с данными о товарах. Создаем файл productData.js.

Импортируем его и заполняем его данными в виде массива (так прописываем каждый товар.)

const products = [

    {

        id: 1,

        title: "ELLERY X M'O CAPSULE",

        description:

            "Known for her sculptural takes on traditional tailoring, Australian arbiter of cool Kym Ellery teams up with Moda Operandi.",

        price: 52.0,

        size: "M",

        color: "Red",

        imageUrl: "images/prod1.jpg",

    },

Код всей реализации сортировки.

import React, { useContext, useState, useEffect } from "react";

import { Link } from "react-router-dom";

import Header from "./Header";

import Footer from "./Footer";

import { CartContext } from "../contexts/CartContext";

import products from "./productsData";

import { ReactComponent as CloseIcon } from "../images/close\_icon.svg";

const CartPage = () => {

    const { cartItems, removeFromCart, updateCartItemQuantity, clearCart } = useContext(CartContext);

    const [totalPrice, setTotalPrice] = useState(0);

    useEffect(() => {

        let total = 0;

        cartItems.forEach((item) => {

            const product = products.find((product) => product.id === item.id);

            total += product.price \* item.quantity;

        });

        setTotalPrice(total);

    }, [cartItems]);

    const handleQuantityChange = (event, productId) => {

        const newQuantity = parseInt(event.target.value);

        updateCartItemQuantity(productId, newQuantity);

    };

    const handleRemoveItem = (productId) => {

        removeFromCart(productId);

    };

    const handleClearCart = () => {

        clearCart();

    };

return (

<div className="box-content">

<Header />

<content className="content">

<div className="head center">

<h1 className="head\_title">SHOPPING CART</h1>

</div>

<div className="cart-box center">

{cartItems.length === 0 ? (

<h2>Your cart is empty</h2>

) : (

<div className="cart-box\_\_left">

<div className="cart-box\_\_products">

{cartItems.map((item) => {

const product = products.find((product) => product.id === item.id);

if (product) {

return (

<div className="cart-box\_\_product\_card" key={item.id}>

<div className="cart-box\_\_card\_img">

<img src={product.imageUrl} alt="Product" />

</div>

<div className="cart-box\_\_card\_text">

<h4 className="cart-box\_\_card\_title">{product.title}</h4>

<p className="cart-box\_\_card\_content">

Price:&nbsp;

<span className="cart-box\_\_card\_price cart-box\_\_card\_value">

${product.price.toFixed(2)}

</span>

</p>

<p className="cart-box\_\_card\_content">

Color:&nbsp;

<span className="cart-box\_\_card\_value">

{product.color}

</span>

</p>

<p className="cart-box\_\_card\_content">

Size:&nbsp;

<span className="cart-box\_\_card\_value">

{product.size}

</span>

</p>

<label htmlFor="quantity" className="cart-box\_\_card\_content">

Quantity:

</label>

<input

id="quantity"

type="number"

min="1"

value={item.quantity}

onChange={(e) => handleQuantityChange(e, item.id)}

/>

</div>

<p className="cart-box\_\_close\_icon" onClick={() => handleRemoveItem(item.id)}>

{/\* поправить \*/}

<CloseIcon />

</p>

</div>

);

}

return null;

})}

</div>

<div className="cart-box\_\_buttons">

<div className="cart-box\_\_shopping\_button" onClick={handleClearCart}>

<p className="cart-box\_\_shopping\_button\_title" >

Clear shopping cart

</p>

</div>

<div className="cart-box\_\_shopping\_button">

<Link className="cart-box\_\_shopping\_button\_title" to="/catalog">

Continue shopping

</Link>

</div>

</div>

</div>

)}

<div className="cart-box\_\_right">

<div className="cart-box\_\_shipping-form">

<h4 className="cart-box\_\_shipping-title">SHIPPING ADRESS</h4>

<input

className="cart-box\_\_shipping-field"

type="text"

id="country"

placeholder="Country"

value="Bangladesh"

required

/>

<input

className="cart-box\_\_shipping-field"

type="text"

id="territory"

placeholder="State"

required

/>

<input

className="cart-box\_\_shipping-field"

type="text"

id="postcode"

placeholder="Postcode / Zip"

required

/>

<div className="cart-box\_\_shipping\_button">

<a className="cart-box\_\_shipping\_button\_title" href="#">

GET A QUOTE

</a>

</div>

</div>

<div className="cart-box\_\_checkout-box">

<div className="cart-box\_\_subtotal">

<div>SUB TOTAL</div>

<div>${totalPrice.toFixed(2)}</div>

</div>

<div className="cart-box\_\_grandtotal">

<div>GRAND TOTAL</div>

<div className="cart-box\_\_totalprice">${totalPrice.toFixed(2)}</div>

</div>

<hr className="cart-box\_\_checkout-line" />

<div className="cart-box\_\_checkout-button">

<a className="cart-box\_\_checkout\_button\_title" href="#">

PROCEED TO CHECKOUT

</a>

</div>

</div>

</div>

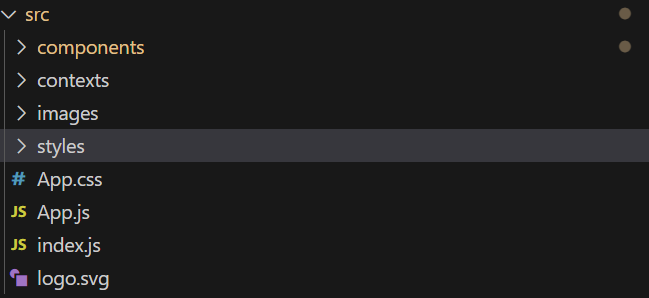
</div>

</content>

После успешного перевода проекта на React можно проверить, как будет выглядеть он с помощью команды: npm run

Эта команда запускает локальный сервер и отображает итоговую собранную версию сайта, что позволяет убедиться, что все работает корректно.

**Основные компоненты проекта**



1. App.js — Главный компонент в приложениях, созданных на основе фреймворка React, выполняет роль центрального управляющего элемента. Он объединяет все подчинённые компоненты и координирует их взаимодействие, управление состояниями и поведение во всём приложении.
2. /components – страницы сайта и данные.
3. /images – все изображения, задействованные на сайте.
4. /styles – стилизация нашего проекта.

В рамках проекта были успешно применены передовые технологии, в частности, React, что позволило реализовать эффективный сайт. Использование данной технологии способствовало созданию кода, отличающегося простотой, а главное предоставило удобство для редактирования проекта.

# ТЕСТИРОВАНИЕ И АНАЛИЗ ПРОЕКТА

Раздел посвящен оценке работоспособности проекта mosh.su на различных типах устройств и веб-браузеров с последующим анализом полученных данных. Проведение тестирования позволяет гарантировать корректное и единообразное отображение сайта на всех поддерживаемых платформах, соответствие современным стандартам качества и обеспечение комфортного взаимодействия пользователя с сайтом.

# 5.1 Тестирование на различных устройствах и браузерах

В ходе тестирования проекта осуществлялась проверка его работоспособности на разнообразных устройствах, таких как настольные компьютеры, планшеты и смартфоны, а также на широко распространённых веб-браузерах. Целью тестирования являлось определение соответствия адаптивной вёрстки и функциональности сайта ожиданиям и требованиям пользователей.

Список устройств и браузеров для тестирования:

1. Настольные компьютеры:
   * Операционная система: Windows 11, macOS.
   * Браузеры: Google Chrome, Mozilla Firefox, Internet Explorer, Safari.
2. Планшеты:
   * Устройства: iPad Pro.
   * Браузеры: Safari (iOS), Google Chrome (Android).
3. Смартфоны:
   * Устройства: iPhone 11 PRO, iPhone 14 PRO, Samsung Note 10.
   * Браузеры: Safari (iOS), Google Chrome (Android).

В ходе тестирования были проведены следующие процедуры:

* Тестирование адаптивной верстки: Проверка корректного отображения контента на устройствах с различными размерами экрана и разрешениями. Для этого использовались инструменты DevTools в браузерах для симуляции размеров экрана, а также тестирование на реальных устройствах.
* Кроссбраузерное тестирование: Оценка совместимости сайта с различными браузерами для выявления потенциальных несоответствий или различий в рендеринге элементов.
* Тестирование интерактивных элементов: Функциональное тестирование всех кнопок, ссылок, анимации, форм обратной связи и навигационных элементов.
* Тестирование производительности: Оценка скорости загрузки сайта на различных устройствах и анализ объема загружаемых ресурсов.

Время загрузки сайта не превышает 2 секунд на большинстве устройств, что является удовлетворительным показателем для сайтов данного типа.

# 5.2 Анализ полученных результатов

Анализ результатов тестирования показал, что сайт [mosh.su](https://mosh.su/) успешно демонстрирует совместимость с различными устройствами и браузерами, обеспечивая удовлетворительную производительность и положительный опыт взаимодействия для пользователя.

Тем не менее, в ходе тестирования были выявлены отдельные аспекты, которые целесообразно усовершенствовать для повышения общей стабильности и удобства использования ресурса:

1. Адаптивная верстка: Сайт в целом демонстрирует хорошую адаптацию к различным размерам экранов. Однако на некоторых устройствах (например, iPad в горизонтальной ориентации) наблюдаются незначительные смещения элементов. Данный вопрос может быть решен путем дополнительной настройки стилей CSS.

2. Кроссбраузерная совместимость: Зафиксированы незначительные различия в отображении шрифтов и размеров элементов в браузере Internet Explorer. Учитывая, что данный браузер считается устаревшим, возможность учета этих отличий для поддержки пользователей, использующих его на корпоративных устройствах, заслуживает рассмотрения.

3. Производительность: Сайт загружается быстро на большинстве устройств. Однако на устройствах с низкой производительностью загрузка анимации может занимать больше времени. Для минимизации времени загрузки рекомендуется оптимизировать размер изображений и использовать метод отложенной загрузки (lazy loading).

4. Пользовательский опыт (UX): Навигация и доступность интерактивных элементов соответствуют современным стандартам. В то же время, можно улучшить плавность переходов между разделами сайта путем добавления анимации перехода и плавных эффектов при смене контента.

# ЗАКЛЮЧЕНИЕ

В ходе реализации проекта [mosh.su](https://mosh.su/) были применены передовые технологии для разработки интерактивного и адаптивного веб- портфолио. Проект был реализован на библиотеке React с использованием HTML5, CSS3, SASS, JavaScript, а также фреймворков. Данный подход позволил создать удобный и эстетически привлекательный интерфейс, а также обеспечить высокую производительность и совместимость с различными браузерами.

В ходе анализа теоретической базы и практического исследования были сделаны следующие выводы:

1. Теоретические и практические выводы:  
    Применение современных веб-технологий, к примеру, фреймворка React, способствует существенному повышению качества структуры кода и облегчению его редактированию в долгосрочной перспективе.
2. Оценка проведённого исследования и его результатов:

Проект прошел тщательное тестирование на разнообразных аппаратных и программных платформах. Это позволило убедиться в его работоспособности и способности функционировать в различных условиях эксплуатации. В ходе проекта были достигнуты все поставленные цели: веб-сайт стал полностью адаптивным к различным устройствам, обрел современный дизайн и соответствует актуальным стандартам web-разработки..

1. Разработанный проект обладает существенной практической ценностью, так как может служить демонстрационной площадкой для презентации профессиональных навыков и компетенций в сфере веб-разработки.

Открытый доступ к проекту в сети Интернет обеспечивает удобство ознакомления с ним потенциальными клиентами, что способствует эффективному представлению возможностей компании. Гибкость используемых технологий позволяет легко модифицировать проект и адаптировать его к различным задачам, таким как расширение функциональности за счет добавления новых страниц, интеграция API или настройка взаимодействия с базами данных.

1. В результате реализации проекта была успешно достигнута поставленная цель – создание интерактивного веб-портфолио на базе современных технологий. Разработанный ресурс отличается адаптивностью к различным устройствам и характеризуется удобством использования. Успешное завершение всех задач проекта подтвердило выдвинутую гипотезу о том, что применение технологий, таких как React, позволяет достичь высокого уровня функциональности и эстетической привлекательности веб-сайта.
2. Возможные направления развития проекта:

В целях дальнейшего совершенствования объекта исследования предлагается рассмотреть следующие усовершенствования:

* Реализация бекэнд части на базе Node.js: Это позволит оптимизировать обработку серверной логики и взаимодействие с базами данных.
* Внедрение механизма обратной связи, способствует повышению уровня взаимодействия пользователей с сайтом и их вовлеченности в его контент.
* Оптимизация мультимедийного контента:

Для повышения производительности веб-сайта и обеспечения его доступности для пользователей с устройствами ограниченных возможностей предусмотрена оптимизация мультимедийных элементов. Данная процедура включает в себя усовершенствование анимации и изображений, что позволит сократить нагрузку на систему и обеспечить плавную работу сайта на различных устройствах.

* Внедрение Application Programming Interface (API) обеспечит динамическое обновление данных, что позволит демонстрировать пользователям актуальную информацию в режиме реального времени. К примеру, API может использоваться для отображения статистики просмотров резюме или уведомления о публикации новых проектов.

Создание данного веб-портфолио стало не только способом демонстрации навыков и знаний в области фронтенд-разработки, но и подтверждением эффективности использования современных фреймворков и инструментов разработки для упрощения создания удобных, и адаптивных интерфейсов. Осуществление проекта послужило ценным опытом для дальнейшего профессионального развития в сфере веб-разработки и внедрения более сложных и функциональных веб-приложений.

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Никсон Р. HTML5 и CSS3. Мастер-класс. СПб.: БХВ-Петербург, 2024.
2. Прохоренок Н. А. Bootstrap и CSS- препроцессор Sass. Самое необходимое. СПб.: БХВ-Петербург, 2021.
3. Хавербеке М. Выразительный JavaScript. Современное веб-программирование. 3-е издание СПб.: Питер, 2024.
4. Дронов В. А. React 17. Разработка веб-приложений на JavaScript. СПб.: БХВ-Петербург, 2022.
5. Официальная документация React: <https://ru.react.js.org/>