Geekbrains

**Создание веб-приложения,** **позволяющего составить программу тренировок в спортивном зале**

IT-специалист:

Frontend-программист

Дегтярев А.Я

Группа: №5992

Тюмень

2024

Оглавление

[Введение 4](#_Toc169569359)

[Веб-приложение и способы получения данных из внешних источников Интернета. 6](#_Toc169569360)

[Веб-приложение: основы. 6](#_Toc169569361)

[Frontend веб-приложения, его особенности. 8](#_Toc169569362)

[Программные продукты и инструменты web-разработки 10](#_Toc169569363)

[Инструменты frontend части. 13](#_Toc169569364)

[HTML 13](#_Toc169569365)

[CSS 17](#_Toc169569366)

[CSS-препроцессоры 18](#_Toc169569367)

[Less 19](#_Toc169569368)

[Sass (SCSS) 20](#_Toc169569369)

[Stylus 20](#_Toc169569370)

[JavaScript и Vue.js 21](#_Toc169569371)

[Vuex 23](#_Toc169569372)

[Vue Router 23](#_Toc169569373)

[SPA (Single Page Application) 24](#_Toc169569374)

[Практическая разработка и тестирование результатов работы веб-приложения 26](#_Toc169569375)

[Разработка frontend части 26](#_Toc169569376)

[Установка Vue CLI 27](#_Toc169569377)

[Разработка backend части 27](#_Toc169569378)

[Установка Vuex 27](#_Toc169569379)

[Корневой компонент агрегатора 27](#_Toc169569380)

[JavaScript файлы хранилища 28](#_Toc169569381)

[store/index.js – база данных 29](#_Toc169569382)

[HomePage.vue – главная страница 36](#_Toc169569383)

[GuidePage.vue – страница с информацией о видах тренинга 38](#_Toc169569384)

[ConstructorPage.vue – страница со списком упражнений и фильтром. 39](#_Toc169569385)

[Error404Page.vue – страница неправильной ссылки 40](#_Toc169569386)

[TrainerBanner.vue – баннер для рекламы или внутренней рекламы онлайн ведения. 42](#_Toc169569387)

[HeaderComp.vue – компонент шапки сайта со списком, лого и навигацией. 44](#_Toc169569388)

[FooterComp.vue – компонент подвала сайта с навигацией. 53](#_Toc169569389)

[ExercisesComp.vue – компонент, который хранит базы упражнений с фильтром. 57](#_Toc169569390)

[Тестирование работоспособности - UI/UX 61](#_Toc169569391)

[Подведение итогов, оценка результатов и заключение 62](#_Toc169569392)

[Список использованной литературы 64](#_Toc169569393)

# Введение

Суть дипломного проекта по созданию веб-приложения хранящего в себе рекомендации, упражнения и их технику выполнения, которые помогут пользователю самостоятельно составить программу тренировок.

Основная цель проекта — предоставить платформу, облегчающую пользователям вход в сферу фитнеса, посредством предоставления образовательной информации, списка наилучших упражнений и их техники выполнения, а также удобным переносом программы тренировок, посредством ее копирования, тем самым давая возможность хранения на других носителях.

Проект будет включать в себя несколько ключевых компонентов, в том числе:

1. Пользовательский интерфейс: пользовательский интерфейс веб-сайта должен быть разработан таким образом, чтобы обеспечить удобство для посетителей. Он будет включать функции поиска, сортировки и другие функции, облегчающие пользователям поиск необходимой им информации;
2. База данных: собранные данные будут храниться в базе данных. База данных планируется к проектированию таким образом, чтобы обеспечивать эффективный поиск и извлечение информации на основе пользовательского выбора, посредством фильтрации;
3. Тестирование и обслуживание. Проект планируется тщательно протестировать, чтобы убедиться, что он работает должным образом и не содержит ошибок. Для поддержания агрегатора в актуальном состоянии и бесперебойной работы потребуется техническое обслуживание и обновления.

План разработки веб-приложения с использованием HTML, CSS, JS и Vue.js будет включать в себя:

1. Разработку пользовательского интерфейса (UI) и взаимодействие с пользователем (UX) приложения с помощью HTML, CSS. Пользовательский интерфейс должен быть визуально привлекательным и простым в навигации.
2. Использование Vue.js для добавления интерактивности и динамического контента в пользовательский интерфейс. Vue.js также можно использовать для создания компонентов и обработки пользовательского ввода, что сделает приложение более отзывчивым и удобным для пользователя;

Проект будет разрабатываться собственными силами. Однако, с точки зрения создания и запуска веб-приложения, в подобных проектах могут участвовать следующие разработчики:

1. Веб-разработчики, обладающие знаниями в области HTML, CSS и JavaScript, могут помочь создать внешний пользовательский интерфейс и добавить интерактивность в приложение с помощью таких фреймворков, как Vue.js;
2. Backend разработчики с опытом работы с Node.js, Express и управлением базами данных могут помочь создать инфраструктуру на стороне сервера, конечные точки API и подключения к базе данных;
3. Инженеры данных могут помочь со сбором, очисткой и хранением данных. Они могут помочь разработать и внедрить конвейеры данных для сбора информации из различных источников, таких как книжные интернет-магазины;
4. QA Инженеры могут помочь с тестированием приложения на наличие ошибок и проблем. Они могут создавать тестовые случаи и сценарии, чтобы убедиться, что приложение работает должным образом;
5. UX-дизайнеры могут помочь с разработкой удобного и визуально привлекательного интерфейса для приложения. Они также могут предоставить рекомендации по передовым методам использования и доступности;
6. Инженеры DevOps могут помочь с развертыванием приложения в производственной среде.

Для создания успешного проекта важно, чтобы команда разработчиков обладала разнообразным набором навыков и опыта. Сотрудничество и общение между членами команды имеют решающее значение для обеспечения того, чтобы проект соответствовал требованиям и был выполнен вовремя.

Дипломный проект по созданию веб-приложения поиска различных блюд в своем городе потребует сочетания навыков программирования и знаний в области веб-разработки. Конечный продукт станет полезным инструментом для людей, которые хотят сделать первые шаги в улучшении собственного здоровья и тела, посредством физических нагрузок в тренажерном зале или дома.

# Веб-приложение и способы получения данных из внешних источников Интернета.

## Веб-приложение: основы.

Веб-приложение — это тип программного приложения, доступ к которому и использование которого осуществляется через веб-браузер. В отличие от традиционных настольных приложений, веб-приложения не нужно устанавливать на компьютер или устройство пользователя, и они предназначены для работы на любом устройстве, подключенным к Интернету.

Содержание веб-приложения может варьироваться в зависимости от его назначения и функциональности. Веб-приложение может представлять собой как простой калькулятор или список дел, так и сложное корпоративное приложение, управляющее рабочими и бизнес-процессами. Как правило, веб-приложение состоит из внешнего пользовательского интерфейса, внутреннего сервера и базы данных, в которой хранятся данные и информация.

Принципы разработки веб-приложений включают использование веб-технологий, таких как HTML, CSS и JavaScript, для создания динамического и интерактивного пользовательского интерфейса, а также использование серверных технологий, таких как Node.js, PHP или Ruby on Rails, для обработки запросов от пользователей. Веб-приложения также используют веб-протоколы, такие как HTTP и HTTPS, для передачи данных между клиентом и сервером.

Цели веб-приложений могут различаться в зависимости от потребностей пользователя. Некоторые общие цели веб-приложений включают в себя:

1. Электронная коммерция: веб-приложения можно использовать для продажи товаров и услуг в Интернете, позволяя клиентам просматривать, сравнивать и покупать товары из любого места, где есть подключение к Интернету;
2. Социальные сети: веб-приложения могут предоставлять службы социальных сетей, которые позволяют пользователям создавать профили, обмениваться контентом и взаимодействовать с другими пользователями;
3. Продуктивность. Веб-приложения могут предоставлять инструменты для совместной работы, управления задачами и планирования проектов, позволяя командам эффективно и продуктивно работать вместе;
4. Управление информацией. Веб-приложения можно использовать для сбора, хранения и анализа данных, что позволяет организациям принимать обоснованные решения и оптимизировать свою деятельность.

Применительно к дипломному проекту, Веб-приложение позволяющего составить программу тренировок в спортивном зале. Суть веб-приложения будет включать в себя список упражнений, направленных на развитие определённых мышечных групп. Список можно будет фильтровать по тегам, приуроченным к группе мышц, которая задействуется в упражнении. Также будет реализована возможность добавления упражнений в список с возможностью копирования.

С точки зрения принципов разработки, веб-приложение должно быть разработано с учетом масштабируемости и удобства обслуживания.

Цель веб-приложения позволяющего составить программу тренировок в спортивном зале состоит в том, чтобы предоставить пользователям базовую информацию о тренировках в зале, видах тренинга, упражнениях и техники их выполнения, чтобы пользователь мог подобрать более подходящий план развития собственного тела как можно раньше, без углубленного изучения нюансов фитнеса в обширном потоке информации из интернета.

## Frontend веб-приложения, его особенности.

Внешний (frontend) интерфейс веб-приложения будет состоять из пользовательского интерфейса, отображающего результаты фильтрации списка упражнений, информацию об упражнениях и технику их выполнения.

Внешний интерфейс веб-приложения относится к той части приложения, которая видна пользователям и взаимодействует с ними напрямую. Обычно он состоит из пользовательского интерфейса, который представлен в веб-браузере и позволяет пользователям взаимодействовать с приложением.

В случае веб-приложения, позволяющего составить программу тренировок в спортивном зале внешний интерфейс может состоять из следующих компонентов:

1. Пользовательский интерфейс: пользовательский интерфейс будет визуальным представлением веб-приложения и будет включать такие элементы, как панели поиска, страницы результатов, информационные страницы, страницы коммерческих услуг и страницы сведений о упражнениях;
2. HTML — это язык разметки, используемый для структурирования содержимого веб-страниц. HTML используется для определения структуры пользовательского интерфейса, включая заголовки, абзацы, списки и таблицы;
3. CSS — это язык стилей, который используется для управления внешним видом веб-страниц. CSS используется для определения визуальных свойств пользовательского интерфейса, таких как цвет и шрифт текста, расположение элементов и общий стиль веб-приложения;
4. Vue.js — это среда JavaScript для создания пользовательских интерфейсов и одностраничных (SPA) приложений. Vue используется для создания динамических и отзывчивых пользовательских интерфейсов, которые обновляются в режиме реального времени на основе пользовательского ввода;

Frontend должен иметь понятный и простой в использовании интерфейс, который позволяет пользователям искать блюда и места, где их можно попробовать. Пользовательский интерфейс должен быть разработан таким образом, чтобы он реагировал и адаптировался к различным размерам экрана и устройствам.

Интерфейс должен отображать обновления в реальном времени для пользователей, когда данные из бэкэнда изменяются, например, когда добавляется или удаляется упражнения в список пользователем.

Пользовательский интерфейс должен быть разработан с учетом высокой производительности и быстрой загрузкой страниц, а также минимальной задержкой.

Внешний интерфейс должен быть совместим с различными веб-браузерами и операционными системами, плюс предназначен для работы на мобильных устройствах, а также на настольных компьютерах.

Пользовательский интерфейс должен быть разработан с учетом безопасности, чтобы защитить личную информацию пользователей за счет использования безопасных протоколов, таких как HTTPS.

# Программные продукты и инструменты web-разработки

Любая разработка начинается с выбора среды разработки. Интегрированные среды разработки (IDE) — это программные приложения, которые обеспечивают комплексную среду для разработки, тестирования и развертывания программного обеспечения. IDE обычно включают в себя редактор кода, компилятор или интерпретатор и инструменты отладки, а также другие функции, такие как интеграция контроля версий и инструменты управления проектами.

Разработчики часто предпочитают IDE, потому что они обеспечивают универсальное решение для всех аспектов процесса разработки. Разработчики могут писать и редактировать код, компилировать и запускать код, а также устранять ошибки в одной и той же среде. IDE также часто предоставляют дополнительные функции, такие как завершение кода, подсветка синтаксиса и инструменты отладки, которые могут ускорить и повысить эффективность разработки.

Однако некоторые разработчики предпочитают использовать текстовые редакторы и инструменты командной строки, поскольку они обеспечивают большую гибкость и могут быть настроены в соответствии с конкретными потребностями разработки. В конечном счете, выбор между использованием IDE или текстового редактора зависит от личных предпочтений и требований проекта разработки.

Наиболее популярные примеры IDE приведены далее.

Visual Studio — это интегрированная среда разработки, разработанная Microsoft, поддерживает широкий спектр языков программирования.

Eclipse — IDE с открытым исходным кодом, поддерживающая широкий спектр языков программирования, включая Java, C++ и Python.

Xcode — разработанная Apple, эта IDE используется для разработки приложений для macOS, iOS, watchOS и tvOS.

IntelliJ IDEA — разработанная JetBrains, эта IDE обеспечивает поддержку Java, Kotlin и других языков.

PyCharm — также разработанная JetBrains, эта IDE ориентирована на разработку Python.

Дипломный проект будет разрабатываться в VS Code.

Visual Studio Code (VS Code) — это бесплатный редактор кода с открытым исходным кодом, разработанный Microsoft. Он широко используется разработчиками для различных задач программирования, включая веб-разработку и машинное обучение. Он доступен для операционных систем Windows, macOS и Linux.

Одной из главных особенностей VS Code является обширный набор расширений и плагинов, которые позволяют разработчикам настраивать редактор под свои конкретные нужды. VS Code поддерживает широкий спектр языков программирования, и доступны расширения для всего: от подсветки синтаксиса и форматирования кода до отладки и интеграции с Git.

Некоторые из ключевых особенностей VS Code включают в себя:

1. IntelliSense: эта функция предоставляет интеллектуальные предложения по завершению кода по мере ввода в зависимости от языка, с которым вы работаете.
2. Встроенная поддержка Git: VS Code включает встроенную поддержку Git, что позволяет легко управлять репозиториями кода, не выходя из редактора.
3. Поддержка отладки: VS Code предоставляет встроенный отладчик для ряда языков программирования, позволяющий легко выполнять код, выявлять и исправлять ошибки.
4. Запуск задач: VS Code поддерживает создание пользовательских задач, которые могут автоматизировать общие рабочие процессы, такие как создание и тестирование кода.
5. Встроенный терминал: VS Code включает встроенное окно терминала, позволяющее выполнять команды оболочки непосредственно в редакторе.
6. Настройка темы: VS Code включает в себя ряд встроенных цветовых тем, и вы также можете загружать и устанавливать пользовательские темы, чтобы настроить внешний вид редактора.
7. Поддержка нескольких языков: VS Code поддерживает широкий спектр языков программирования, включая JavaScript, Python, Ruby и многие другие.

VS Code — очень гибкий и настраиваемый редактор кода, предоставляющий ряд мощных функций для разработчиков. Он имеет большое и активное сообщество пользователей и разработчиков, а в Интернете доступно множество ресурсов, которые помогут начать работу и научиться эффективно его использовать.

## Инструменты frontend части.

### HTML

HTML (HyperText Markup Language) является основным языком разметки веб-страниц. Он используется для создания содержимого веб-страниц, которое отображается в браузере.

HTML является стандартизированным языком, который определяет структуру веб-страницы с помощью различных элементов и атрибутов. Каждый элемент представляет определенный тип содержимого, такой как заголовки, параграфы, списки, изображения и т.д.

HTML также используется для создания ссылок на другие веб-страницы, внутренние и внешние. Он позволяет создавать формы, которые пользователи могут заполнять и отправлять, а также добавлять мультимедийный контент, такой как видео и аудио.

HTML является важным инструментом для создания веб-страниц и веб-приложений. Он обеспечивает структуру и содержимое веб-страниц, которые пользователи видят в браузере. Без HTML веб-страницы были бы просто набором несвязанных текстов и изображений.

HTML использует теги (tags) для определения элементов страницы. Теги создаются с использованием угловых скобок <>, и обычно имеют начальный и конечный тег, где конечный тег отличается от начального символом "/".

Внутри тегов могут находиться атрибуты (attributes), которые определяют дополнительные характеристики элементов, такие как их цвет, размер, ссылки на другие страницы и т.д.

Некоторые наиболее часто используемые теги в HTML:

1. <html>: Начальный тег для HTML-документа.
2. <head>: Содержит метаданные документа, такие как заголовки страницы, ссылки на стили CSS, и другие данные, которые не отображаются на странице.
3. <title>: Определяет заголовок документа.
4. <body>: Определяет содержимое страницы, которое отображается в браузере.
5. <div>: Определяет блок элементов на странице.
6. <p>: Определяет абзац текста.
7. <img>: Определяет изображение на странице.
8. <a>: Определяет ссылку на другую страницу или ресурс.
9. <form>: Определяет форму на странице, которую пользователь может заполнить и отправить.

В современной верстке для распределения элементов на странице чаще всего используются теги <div> и <section>.

<div> - это тег в HTML, который используется для определения контейнеров на веб-страницах. Он создает блоки, в которые можно помещать другие элементы, такие как текст, изображения, видео, таблицы и т.д. <div> не предназначен для определения семантики содержимого, а скорее служит для оформления и стилизации. Он не влияет на структуру документа, но может использоваться для группировки и организации элементов страницы.

Пример использования тега <div>:

<div>  
 <h1>Заголовок</h1>  
 <p>Это простой текст на странице.</p>  
</div>

Здесь создается блок, внутри которого находится заголовок первого уровня и абзац текста. Контейнер <div> позволяет объединить эти элементы, чтобы упростить их оформление и стилизацию. <div> можно также использовать для разделения элементов страницы на группы, например для создания секций или блоков с футером и хедером. В сочетании с CSS стилями, <div> может быть мощным инструментом для создания разнообразных макетов и оформления веб-страниц.

<section> - это тег в HTML, который используется для определения разделов на веб-страницах. Он обычно содержит группы связанных элементов, таких как заголовки, параграфы, изображения и другие элементы. <section> обычно используется для структурирования документа и улучшения доступности и семантической структуры.

Пример использования тега <section>:

<section>

<h2>Название раздела</h2>

<p>Это текст в разделе.</p>

<img src="image.jpg" alt="Изображение">

</section>

Здесь создается раздел страницы, содержащий заголовок второго уровня, абзац текста и изображение. Контейнер <section> позволяет группировать эти элементы, чтобы указать, что они относятся к одному разделу страницы.

<section> обычно используется в сочетании с другими семантическими тегами, такими как <article>, <header>, <footer> и <nav>, чтобы создать структурированную и понятную семантическую разметку веб-страницы. <section> также может использоваться для стилизации страницы с помощью CSS. Он позволяет определять общие стили для групп связанных элементов и упрощает оформление веб-страницы.

Понимание тегов является необходимым для создания и правильного размещения элементов на странице. Например, правильное использование тегов позволяет браузеру корректно интерпретировать содержимое страницы и отображать его правильно. Также это может повлиять на поисковую оптимизацию (SEO), что важно для любого веб-сайта.

### CSS

CSS (Cascading Style Sheets) - это язык стилей, который используется для оформления веб-страниц. Он определяет, как должны выглядеть элементы HTML на странице, включая их цвет, размер, расположение и другие характеристики.

CSS позволяет разработчикам создавать стильные и привлекательные веб-страницы, которые могут быть адаптированы под различные устройства и размеры экранов.

Основные принципы CSS:

1. Каскадность - стили могут быть определены на разных уровнях, и значения, определенные на более высоком уровне, могут быть переопределены на более низком уровне.
2. Наследование - стили, определенные для родительского элемента, могут быть унаследованы его дочерними элементами.
3. Приоритетность - стили могут иметь разную приоритетность, и значение, определенное для элемента с более высокой приоритетностью, будет использоваться вместо значения элемента с более низкой приоритетностью.

Пример CSS:

body {

background-color: #f0f0f0;

font-family: Arial, sans-serif;

}

h1 {

color: #333;

font-size: 28px;

text-align: center;

}

p {

color: #666;

font-size: 16px;

line-height: 1.5;

}

В примере выше определены стили для элементов <body>, <h1> и <p>. <body> задает фоновый цвет и шрифт для всей страницы, <h1> задает цвет, размер и выравнивание для заголовков первого уровня, а <p> задает цвет, размер и высоту строки для параграфов.

CSS может быть определен внутри HTML-документа с использованием тега <style>, или может быть вынесен в отдельный файл .css для повторного использования на других страницах. С помощью CSS можно создавать сложные макеты страниц, анимации и интерактивные элементы, чтобы улучшить пользовательский опыт на веб-сайте.

### CSS-препроцессоры

CSS препроцессор (от англ. CSS preprocessor) — это надстройка над CSS, которая добавляет ранее недоступные возможности для CSS, с помощью новых синтаксических конструкций.

Основная задача препроцессора — это предоставление удобных синтаксических конструкций для разработчика, чтобы упростить, и тем самым, ускорить разработку и поддержу стилей в проектах.

CSS препроцессоры преобразуют код, написанный с использованием препроцессорного языка, в чистый и валидный CSS-код.

При помощи препроцессоров вы можете писать код, который нацелен на:

* Читабельность для человека
* Структурированность и логичность
* Производительность

Виды популярных CSS-препроцессоров:

* Less
* Sas (SCSS)
* Stylus

### Less

Собственно, герой этой книги. Самый популярный на момент написания книги препроцессор. Основан в 2009 году Алексис Сельер (Alexis Sellier) и написан на JavaScript (изначально был написан на Ruby, но Алексис вовремя сделал правильный шаг). Имеет все базовые возможности препроцессоров и даже больше, но не имеет условных конструкций и циклов в привычном для нас понимании. Основным плюсом является его простота, практически стандартный для CSS синтаксис и возможность расширения функционала за счёт системы плагинов.

### Sass (SCSS)

Самый мощный из CSS-препроцессоров. Имеет довольно большое сообщество разработчиков. Основан в 2007 году как модуль для HAML и написан на Ruby (есть порт на C++). Имеет куда больший ассортимент возможностей в сравнении с Less. Возможности самого препроцессора расширяются за счёт многофункциональной библиотеки Compass, которая позволяет выйти за рамки CSS и работать, например, со спрайтами в автоматическом режиме.

Имеет два синтаксиса:

Sass (Syntactically Awesome Style Sheets) — упрощённый синтаксис CSS, который основан на идентации. Считается устаревшим. SCSS (Sassy CSS) — основан на стандартном для CSS синтаксисе.

### Stylus

Самый молодой, но в тоже время самый перспективный CSS-препроцессор. Основан в 2010 году небезызвестной в наших кругах личностью TJ Holowaychuk. Говорят, это самый удобный и расширяемый препроцессор, а ещё он гибче Sass. Написан на JavaScript. Поддерживает уйму вариантов синтаксиса от подобного CSS до упрощённого (отсутствуют :, ;, {} и некоторые скобки).

В данной дипломной работе будет использован css-препроцессор SASS для удобной и практичной работы со стилями.

### JavaScript и Vue.js

JavaScript — это язык программирования, обычно используемый для создания динамических веб-страниц и веб-приложений. Это язык высокого уровня, поддерживающий объектно-ориентированный, императивный и функциональный стили программирования. JavaScript можно использовать для различных задач, таких как добавление интерактивных функций на веб-страницы, создание браузерных игр, создание веб-приложений и мобильных приложений, а также разработка серверных приложений.

Некоторые из ключевых особенностей JavaScript включают в себя:

1. Программирование, управляемое событиями: JavaScript позволяет разработчикам создавать интерактивные веб-страницы, которые реагируют на действия пользователя, такие как нажатие кнопки, прокрутка страницы или ввод данных в форму.
2. Асинхронное программирование: JavaScript поддерживает асинхронное программирование, что позволяет разработчикам писать код, который не блокирует выполнение другого кода.
3. Кроссплатформенная совместимость: JavaScript поддерживается всеми основными веб-браузерами, что делает его популярным выбором для веб-разработки.
4. Расширяемость: JavaScript можно расширить с помощью библиотек и фреймворков, таких как jQuery, React и Vue.js, которые упрощают разработку сложных веб-приложений.

Vue.js - это прогрессивный фреймворк JavaScript, который используется для создания пользовательских интерфейсов и одностраничных приложений (SPA). Он представляет собой модульный фреймворк, который может использоваться вместе с другими библиотеками и инструментами, такими как Vuex, Vue Router, Vuetify и другие.

Основные преимущества Vue.js:

1. Легковесность - Vue.js имеет небольшой размер и быстро загружается на страницу, что позволяет создавать быстрые и отзывчивые пользовательские интерфейсы.
2. Гибкость - Vue.js можно использовать как для маленьких, простых приложений, так и для крупных и сложных проектов. Он также предоставляет широкий выбор опций и настроек, которые позволяют настроить его под различные потребности.
3. Простота в использовании - Vue.js легко устанавливается и используется даже начинающими разработчиками. Он также предоставляет простые и интуитивно понятные API и документацию.
4. Однофайловые компоненты - Vue.js поддерживает однофайловые компоненты, которые содержат в себе все необходимые HTML, CSS и JavaScript коды, что облегчает разработку и обслуживание приложений.

Vue.js предоставляет набор компонентов и директив, которые можно использовать для создания пользовательского интерфейса, а также API для управления состоянием приложения. Он также поддерживает использование шаблонов, фильтров и многих других функций, которые упрощают разработку.

Vue.js можно использовать как самостоятельный фреймворк, так и в сочетании с другими технологиями, такими как HTML, CSS, JavaScript, Bootstrap и другие. Он также может быть использован с различными инструментами и плагинами, такими как Vuex, Vue Router, Vuetify и другие, что позволяет создавать более сложные приложения и улучшать производительность.

### Vuex

Vuex — паттерн управления состоянием + библиотека для приложений на Vue.js. Он служит централизованным хранилищем данных для всех компонентов приложения с правилами, гарантирующими, что состояние может быть изменено только предсказуемым образом. Vuex интегрируется с официальным расширением vue-devtools, предоставляя «из коробки» такие продвинутые возможности, как «машину времени» для отладки и экспорт/импорт слепков состояния данных.

Vuex помогает управлять совместно используемым состоянием ценой привнесения новых концепций и вспомогательного кода. Компромисс, когда кратковременная продуктивность страдает на благо долгосрочной.

В данной дипломной работе Vuex будет использован для симуляции работы с базой данных избегая тем самым задержек получения информации из базы или исключая возможность не получить информацию с сервера.

### Vue Router

Vue Router — официальная библиотека маршрутизации для Vue.js (opens new window). Она глубоко интегрируется с Vue.js и позволяет легко создавать SPA-приложения. Включает следующие возможности:

* Вложенные маршруты/представления
* Модульная конфигурация маршрутизатора
* Доступ к параметрам маршрута, query, wildcards
* Анимация переходов представлений на основе Vue.js
* Удобный контроль навигации
* Автоматическое проставление активного CSS класса для ссылок
* Режимы работы HTML5 history или хэш, с авто-переключением в IE9
* Настраиваемое поведение прокрутки страницы

### SPA (Single Page Application)

Single Page Application — это веб-приложение или веб-сайт, использующий единственный HTML-документ как оболочку для всех веб-страниц и организующий взаимодействие с пользователем через динамически подгружаемые HTML, CSS, JavaScript,

Когда пользователь открывает страницу, браузер загружает сразу весь код приложения. Но показывает только конкретный модуль — часть сайта, которая нужна пользователю. Когда пользователь переходит в другую часть приложения, браузер берёт уже загруженные данные и показывает ему. И, если нужно, динамически подгружает с сервера нужный контент без обновления страницы.

Одностраничные приложения чаще всего используют в сервисах, где пользователь проводит на одной странице много времени или совершает с ней какие-то действия, например:

* просматривает почту и отмечает письма как спам;
* пишет посты и комментирует чужие;
* смотрит сериалы;
* выбирает квартиру;
* разглядывает картинки и собирает их в тематические доски.

Веб-версии Gmail, Facebook Netflix, AirBnB и Pinterest — одностраничные приложения. Технология настолько распространена, что её используют даже для сайтов компаний.

SPA может обмениваться данными с сервером без перезагрузки страницы. Благодаря этому наполнение страницы может меняться динамически. Например, раньше в социальной сети нужно было перезагрузить страницу, чтобы проверить, нет ли новых сообщений, теперь они появляются автоматически.

Обратите внимание, например, как работает переключение между разделами в интерфейсе Фейсбука: если открыть главную, а потом перейти в «Группы», то содержание страницы изменится, но шапка останется на своём месте — перезагрузки страницы не происходит.

Преимущества SPA

SPA быстрые. Переход между модулями в приложении происходит быстрее: нужные ресурсы уже загружены, нужно просто подставить данные, которые запросил пользователь. Часто при этом сервер возвращает не тяжеловесный HTML, а лёгкий JSON или XML.

Ещё использование JSON упрощает разработку приложения для разных платформ. Если для веб-версии разработать обычный сайт, который принимает от сервера HTML, то для мобильного приложения придётся писать доработку, так как там HTML не подойдёт. JSON делает ответ сервера универсальным.

SPA гибкие. Раз пользователь всё время работает с одной страницей, проще делать интересные переходы и анимацию элементов. Можно работать с состоянием кнопок, вкладок и переключателей. Таким образом, интерфейс SPA может быть похож скорее на полноценное приложение, а не на простой сайт.

SPA работают везде. Всё, что нужно для SPA — поддержка JavaScript. Такие сайты хорошо работают и на десктопе, и в вебе, могут отчасти заменить полноценные мобильные приложения.

# Практическая разработка и тестирование результатов работы веб-приложения

## Разработка frontend части

Vue.js был выбран за его компонентный подход к созданию пользовательского интерфейса, использование директив и реактивных свойств для управления динамическим контентом, а также удобной интеграции с другими библиотеками и фреймворками.

Веб-приложение с использованием Vue.js состоит из компонентов, которые можно сравнить с независимыми блоками кода, отвечающими за отображение, логику и данные. Компоненты могут быть переиспользованы в разных частях приложения, что делает код более модульным и поддерживаемым.

Для создания интерфейса веб-приложения на Vue.js были использованы HTML-шаблоны, содержащие директивы Vue.js и JavaScript-код, отвечающий за логику и управление состоянием приложения.

### Установка Vue CLI

npm install -g @vue/cli

Данный тип установки позволяет использовать компонентный подход в разработке веб приложения.

После установки создается папка в которой преднастроены все необходимые модули.

Запуск разработки производится командой:

npm run serve

## Разработка backend части

### Установка Vuex

После установки Vue CLI подключим библиотеку Vuex специальной командой и выбрав интересующую версию

npm install vuex

### Корневой компонент агрегатора

App.vue — корневой компонент приложения Vue.js. Это начальная точка приложения, где визуализируются все остальные компоненты. Это однофайловый компонент, который служит контейнером для других компонентов и определяет базовую структуру приложения.

App.vue обычно состоит из трех основных частей: шаблона, скрипта и стиля. Раздел шаблона содержит HTML-структуру приложения, раздел сценария содержит код JavaScript, управляющий поведением приложения, а раздел стиля содержит код CSS, определяющий стиль приложения.

App.vue может также содержать другие компоненты и импортировать библиотеки по мере необходимости. Это центральная точка приложения, позволяющая легко управлять компонентами и данными приложения.

<template>

<div class="top" id="app">

<HeaderComp/>

<router-view/>

<FooterComp/>

</div>

</template>

<script>

import HeaderComp from './components/HeaderComp.vue'

import FooterComp from './components/FooterComp.vue'

export default {

name: 'App',

components: {

HeaderComp,

FooterComp,

}

}

</script>

<style lang="sass">

\*

padding: 0

margin: 0

box-sizing: border-box

a

text-decoration: none

.top

position: relative

background-color: rgb(15, 15, 15)

width: 100%

.center

margin-left: calc(50% - 1200px / 2)

margin-right: calc(50% - 1200px / 2)

</style>

### JavaScript файлы хранилища

* 1. **store/index.js** – База данных сайта;
  2. **router/index.js** – Хранилище путей для компонентов Vue Router для SPA

Дочерние компоненты агрегатора

1. **HomePage.vue** **–** главная страница содержащая в себе информацию о веб-приложении;
2. **GuidePage.vue –** страница блюда содержащая в себе информацию о видах тренировок в зале;
3. **Error404Page.vue –** страница если был использован некорректный URL;
4. **ConstructorPage.vue –** страница, содержащая весь список упражнений и фильтр по тегу .
5. **HeaderComp.vue –** компонент “шапки” страницы.
6. **FooterComp.vue** - компонент “подвала” страницы.
7. **TrainerBanner.vue** – компонент с предложением онлайн ведения тренерами веб-приложения.
8. **TopContent.vue** -компонент с информацией на главной странице.
9. **GuideComp.vue –** компонент **с информацией о видах тренировочного процесса.**
10. **ExercicesComp.vue –** компонент**, содержащий отрисовку упражнений с фильтром.**

# store/index.js – база данных

import { createStore } from "vuex";

export default createStore({

  state: {

    topBanner: {

      title: "FIT BRO",

      img: "gym.jpg",

      name: "topBannerImg",

      text: "Это проект, призванный помочь вам составить свою идеальную программу тренировок для зала.",

    },

    topButtons: [

      { title: "Что нужно знать", url: "/GuidePage" },

      { title: "Упражнения", url: "/ConstructorPage" },

      { title: "Помощь профи", url: "#" },

    ],

    topList: [

      { text: "Виды программ" },

      { text: "Лучшие упражнения" },

      { text: "Техника выполнения" },

      { text: "Процент вовлечения мыщечных групп" },

      { text: "Онлайн ведение" },

      { text: "Полезная информация" },

    ],

    links: [

      { id: 1, title: "Home", url: "./HomePage" },

      { id: 2, title: "Гайд", url: "./GuidePage" },

      { id: 3, title: "Упражнения", url: "./ConstructorPage" },

    ],

    waves: [{ id: "wave1" }, { id: "wave2" }, { id: "wave3" }, { id: "wave4" }],

    linksFooter: [

      { url: "#", title: "Home" },

      { url: "#", title: "About" },

      { url: "#", title: "Services" },

    ],

    icons: [

      { name: "vk", img: "vk.svg", url: "#" },

      { name: "youtube", img: "youtube.svg", url: "#" },

      { name: "telegram", img: "telegram.svg", url: "#" },

    ],

    banners: [

      { img: "Trainer1.jpg" },

      { img: "Trainer2.jpg" },

      { img: "Trainer3.jpg" },

    ],

    TrainerBannerText: {

      subtitle: "Достигай цели!",

      text1:

        "Наши опытные тренера обладают глубокими знаниями и высоким профессионализмом для того, чтобы помочь каждому клиенту достичь своих целей в зале",

      text2:

        "Специально разработанные программы тренировок, персональный подход и мотивационная поддержка позволят нашим клиентам раскрыть свой потенциал и добиться желаемых результатов. Наши тренеры не просто следят за выполнением упражнений, они также помогают с питанием, коррекцией техники и поддерживают клиента на пути к здоровью и физической форме",

      text3:

        "Вместе с нашей командой вы сможете преодолеть любые трудности и достичь новых вершин в своем фитнес-путешествии",

    },

    Exercises: [

      {

        id: 1,

        name: "Жим штанги лежа",

        desc: "Это упражнение отлично развивает грудные мышцы, дельты и трицепсы, помогая укрепить верхнюю часть тела и улучшить общую силу.",

        tegs: ["Грудь", "Трицепс", "Дельты"],

        img: "Ex1.jpg",

        url: "https://www.youtube.com/watch?v=haAmJyTFMKE&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 2,

        name: "Жим штанги на наклонной скамье",

        desc: "Позволяет сосредоточиться на верхней части груди, развивая и придавая форму этой области, что способствует созданию более сбалансированного развития грудных мышц.",

        tegs: ["Грудь", "Трицепс", "Дельты"],

        img: "Ex2.jpg",

        url: "https://www.youtube.com/watch?v=5OlJ6WFLWQM&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 3,

        name: "Присед со штангой",

        desc: "Отличное упражнение для развития нижней части тела, включая квадрицепсы, ягодицы и стабилизирующие мышцы кора. Помогает улучшить силу и выносливость ног.",

        tegs: ["Квадрицепс", "Ягодицы"],

        img: "Ex3.jpg",

        url: "https://www.youtube.com/watch?v=EgGx7tp7mTs&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 4,

        name: "Становая тяга",

        desc: "Развивает почти все основные группы мышц, включая спину, ягодицы, бедра и предплечья. Укрепляет спину, улучшает осанку и помогает в повседневных движениях.",

        tegs: ["Спина", "Ягодицы", "Бедра"],

        img: "Ex4.jpg",

        url: "https://www.youtube.com/watch?v=80BlWHAxPag&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 5,

        name: "Отжимания на брусьях",

        desc: "Прекрасное упражнение для развития грудных мышц, дельт и трицепсов, а также для укрепления стабилизаторов плечевого пояса.",

        tegs: ["Грудь", "Трицепс", "Дельты"],

        img: "Ex5.jpg",

        url: "https://www.youtube.com/watch?v=ORE34\_2m62A&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 6,

        name: "Подъем штанги на бицепс",

        desc: "Это упражнение направлено на развитие бицепсов и предплечий, помогая улучшить силу и форму рук.",

        tegs: ["Бицепс", "Предплечья"],

        img: "Ex6.jpg",

        url: "https://www.youtube.com/watch?v=WpbyWZjmkpU&ab\_channel=FreshLife28",

      },

    ],

    Cart: [],

    Filters: [

      { id: 1, text: "Все" },

      { id: 2, text: "Грудь" },

      { id: 3, text: "Дельты" },

      { id: 4, text: "Трицепс" },

      { id: 5, text: "Спина" },

      { id: 6, text: "Квадрицепс" },

      { id: 7, text: "Ягодицы" },

      { id: 8, text: "Бедра" },

      { id: 9, text: "Бицепс" },

      { id: 10, text: "Предплечья" },

    ],

    FilterExercises: [

      {

        id: 1,

        name: "Жим штанги лежа",

        desc: "Это упражнение отлично развивает грудные мышцы, дельты и трицепсы, помогая укрепить верхнюю часть тела и улучшить общую силу.",

        tegs: ["Грудь", "Трицепс", "Дельты"],

        img: "Ex1.jpg",

        url: "https://www.youtube.com/watch?v=haAmJyTFMKE&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 2,

        name: "Жим штанги на наклонной скамье",

        desc: "Позволяет сосредоточиться на верхней части груди, развивая и придавая форму этой области, что способствует созданию более сбалансированного развития грудных мышц.",

        tegs: ["Грудь", "Трицепс", "Дельты"],

        img: "Ex2.jpg",

        url: "https://www.youtube.com/watch?v=5OlJ6WFLWQM&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 3,

        name: "Присед со штангой",

        desc: "Отличное упражнение для развития нижней части тела, включая квадрицепсы, ягодицы и стабилизирующие мышцы кора. Помогает улучшить силу и выносливость ног.",

        tegs: ["Квадрицепс", "Ягодицы"],

        img: "Ex3.jpg",

        url: "https://www.youtube.com/watch?v=EgGx7tp7mTs&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 4,

        name: "Становая тяга",

        desc: "Развивает почти все основные группы мышц, включая спину, ягодицы, бедра и предплечья. Укрепляет спину, улучшает осанку и помогает в повседневных движениях.",

        tegs: ["Спина", "Ягодицы", "Бедра"],

        img: "Ex4.jpg",

        url: "https://www.youtube.com/watch?v=80BlWHAxPag&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 5,

        name: "Отжимания на брусьях",

        desc: "Прекрасное упражнение для развития грудных мышц, дельт и трицепсов, а также для укрепления стабилизаторов плечевого пояса.",

        tegs: ["Грудь", "Трицепс", "Дельты"],

        img: "Ex5.jpg",

        url: "https://www.youtube.com/watch?v=ORE34\_2m62A&ab\_channel=SMARTTRAININGChannel",

      },

      {

        id: 6,

        name: "Подъем штанги на бицепс",

        desc: "Это упражнение направлено на развитие бицепсов и предплечий, помогая улучшить силу и форму рук.",

        tegs: ["Бицепс", "Предплечья"],

        img: "Ex6.jpg",

        url: "https://www.youtube.com/watch?v=WpbyWZjmkpU&ab\_channel=FreshLife28",

      },

    ],

    TypesList: [

      { id: 1, name: "SPLIT" },

      { id: 2, name: "FULL BODY" },

      { id: 3, name: "PUSH/PULL" },

    ],

    GuideImgList: [

      {

        id: 1,

        url: "Guide1.jpg",

        text: "Система Split-тренировки предполагает разделение тренировочной программы на отдельные дни, посвященные работе с определенными группами мышц. Обычно используются следующие разделения: разделение по группам мышц (грудь/трицепс, спина/бицепс, ноги/плечи) или разделение по конкретным мышцам (грудные мышцы, спина, ноги, плечи, бицепсы, трицепсы).",

        title: "Split-тренировка",

      },

      {

        id: 2,

        url: "Guide2.jpg",

        text: "Система Full Body предполагает выполнение упражнений для всех основных групп мышц в одной тренировке. Этот подход обеспечивает стимуляцию всего организма за одну тренировку и может быть особенно полезен для начинающих спортсменов или тех, кто хочет тренироваться несколько раз в неделю.",

        title: "Full Body",

      },

      {

        id: 3,

        url: "Guide3.jpg",

        text: "Система Push-Pull разделяет упражнения на две категории: толкающие (push) и тянущие (pull). В одной тренировке вы выполняете упражнения, которые 'толкают' (например, жим штанги лежа, отжимания) и в другой — упражнения, которые 'тянут' (например, подтягивания, тяга штанги к подбородку).",

        title: "Push-Pull",

      },

    ],

    activeButton: 1,

    activeType: 1,

  },

  getters: {},

  mutations: {

    ADD\_\_EXERCISE(state, { exercise }) {

      state.Cart.push(exercise);

    },

    DEL\_\_EXERCISE(state, id) {

      const index = state.Cart.findIndex((item) => item.id == id);

      if (index !== -1) {

        state.Cart.splice(index, 1);

      }

    },

    CLEAR\_\_CART(state) {

      state.Cart.splice(0, state.Cart.length);

    },

    FILTER\_\_ARR(state, tag) {

      if (tag.text == "Все") {

        state.FilterExercises = state.Exercises;

      } else {

        state.FilterExercises = state.Exercises.filter((exercise) =>

          exercise.tegs.includes(tag.text)

        );

      }

      this.state.activeButton = tag.id;

    },

    COPY\_\_CART(state) {

      const cartData = state.Cart.map((item) => {

        return item.name;

      });

      const dataToCopy = cartData.join("\n");

      navigator.clipboard

        .writeText(dataToCopy)

        .then(() => {

          alert("Данные успешно скопированы в буфер обмена");

        })

        .catch((err) => {

          alert("Ошибка копирования данных: ", err);

        });

    },

    ACTIVE\_\_TYPE(state, type) {

      state.activeType = type.id;

    },

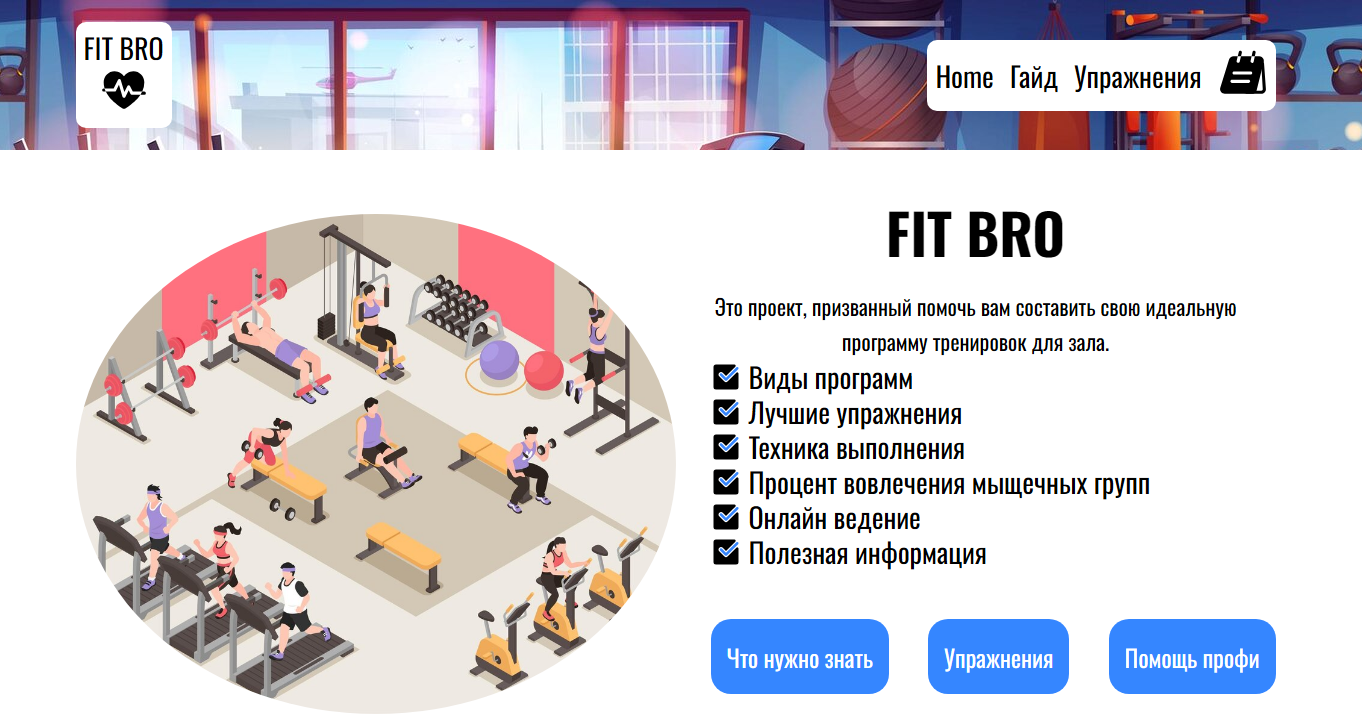
  },

  actions: {},

  modules: {},

});

### HomePage.vue – главная страница



<template>

  <div>

    <TopContent />

    <TrainerBanner />

  </div>

</template>

<script>

import TopContent from "@/components/TopContent.vue";

import TrainerBanner from "@/components/TrainerBanner.vue";

export default {

  name: "HomePage",

  components: {

    TopContent,

    TrainerBanner,

  },

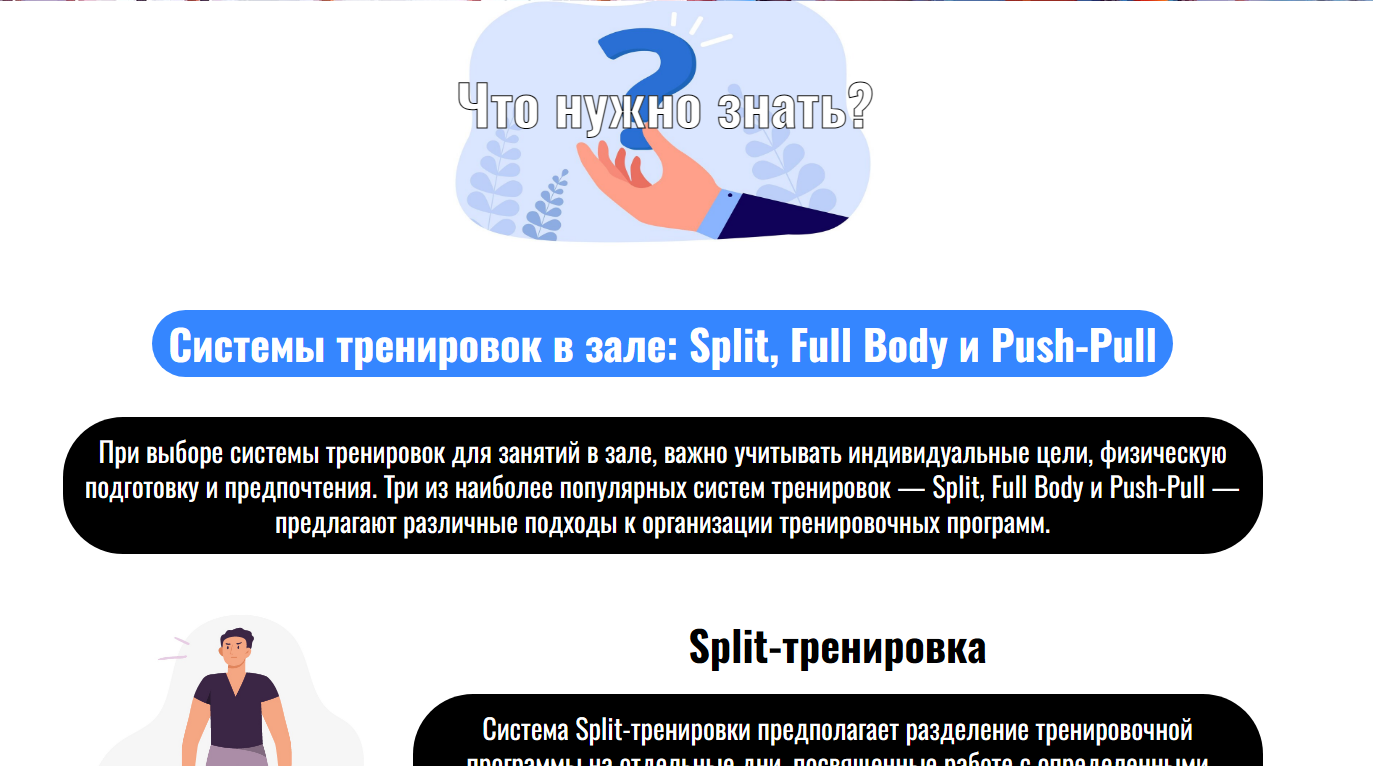
};

</script>

<style lang="sass"></style>

Содержит в информацию о том, чем является веб-приложение.

### GuidePage.vue – страница с информацией о видах тренинга



<template>

  <div>

    <GuideComp />

  </div>

</template>

<script>

import GuideComp from "@/components/GuideComp.vue";

export default {

  name: "GuidePage",

  components: {

    GuideComp,

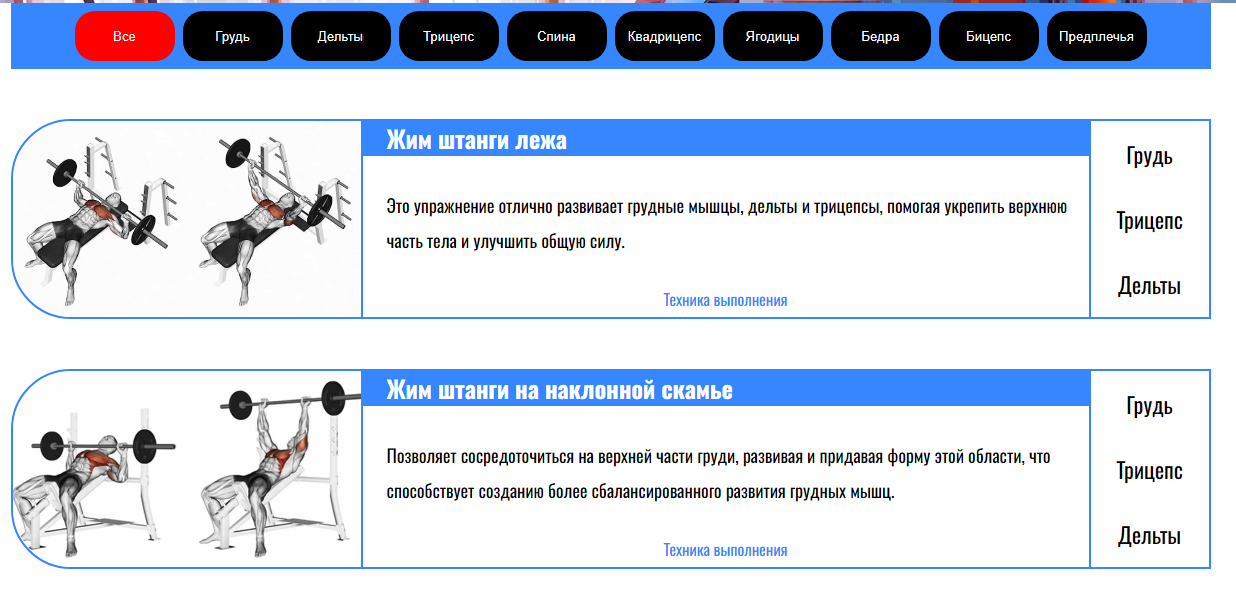
  },

};

</script>

<style lang="sass"></style>

### ConstructorPage.vue – страница со списком упражнений и фильтром.



<template>

  <div>

    <ExercisesComp />

  </div>

</template>

<script>

import ExercisesComp from "@/components/ExercisesComp.vue";

export default {

  name: "ConstructorPage",

  components: {

    ExercisesComp,

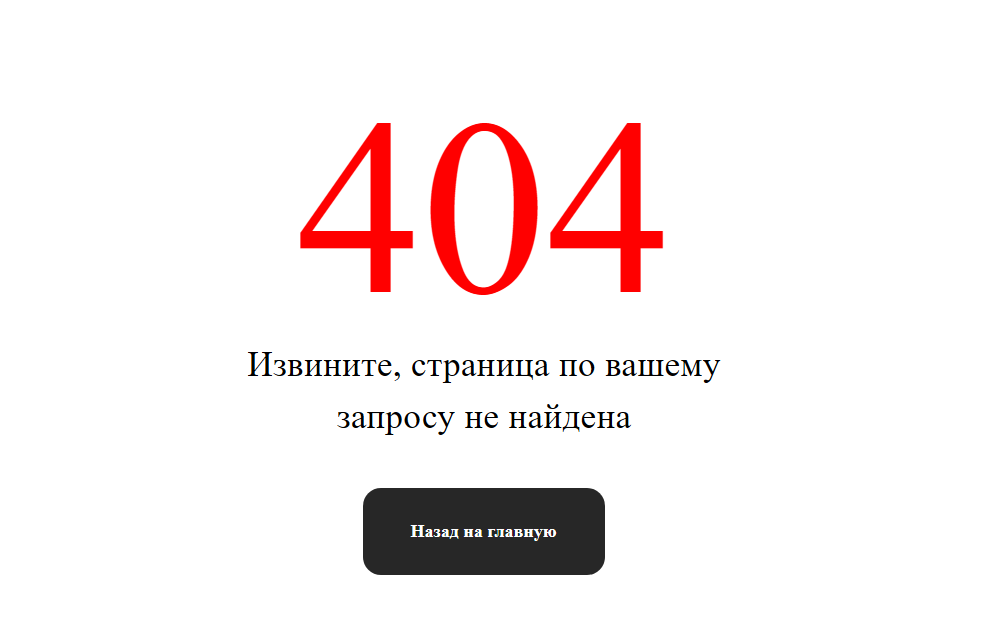
  },

};

</script>

<style lang="sass"></style>

### Error404Page.vue – страница неправильной ссылки



<template lang>

  <section class="page404 center">

    <div class="page404\_text">

      <h1 class="page404\_text\_\_title">404</h1>

      <p class="page404\_text\_\_description">

        Извините, страница по вашему запросу не найдена

      </p>

      <router-link to="/HomePage" class="page404\_text\_\_link">

        Назад на главную

      </router-link>

    </div>

  </section>

</template>

<script>

export default {

  name: "ErrorComp",

};

</script>

<style lang="sass" scoped>

.page404

    display: flex

    height: 800px

    margin-bottom: 50px

    flex-direction: column

    align-items: center

    justify-content: center

    &\_text

        display: flex

        flex-direction: column

        align-items: center

        width: 569px

        gap: 44px

        &\_\_title

            color: red

            font-family: Jost

            font-size: 250px

            font-style: normal

            font-weight: 400

            line-height: 70%

        &\_\_description

            color: black

            font-family: Jost

            font-size: 35px

            font-style: normal

            font-weight: 400

            line-height: 150%

            letter-spacing: 0.35px

            text-align: center

        &\_\_link

            display: flex

            gap: 10px

            align-items: center

            color: white

            font-family: Jost

            font-size: 18px

            font-weight: 600

            padding: 26px 49px 26px 48px

            border-radius: 18px

            background: #272727

            &:hover

                cursor: pointer

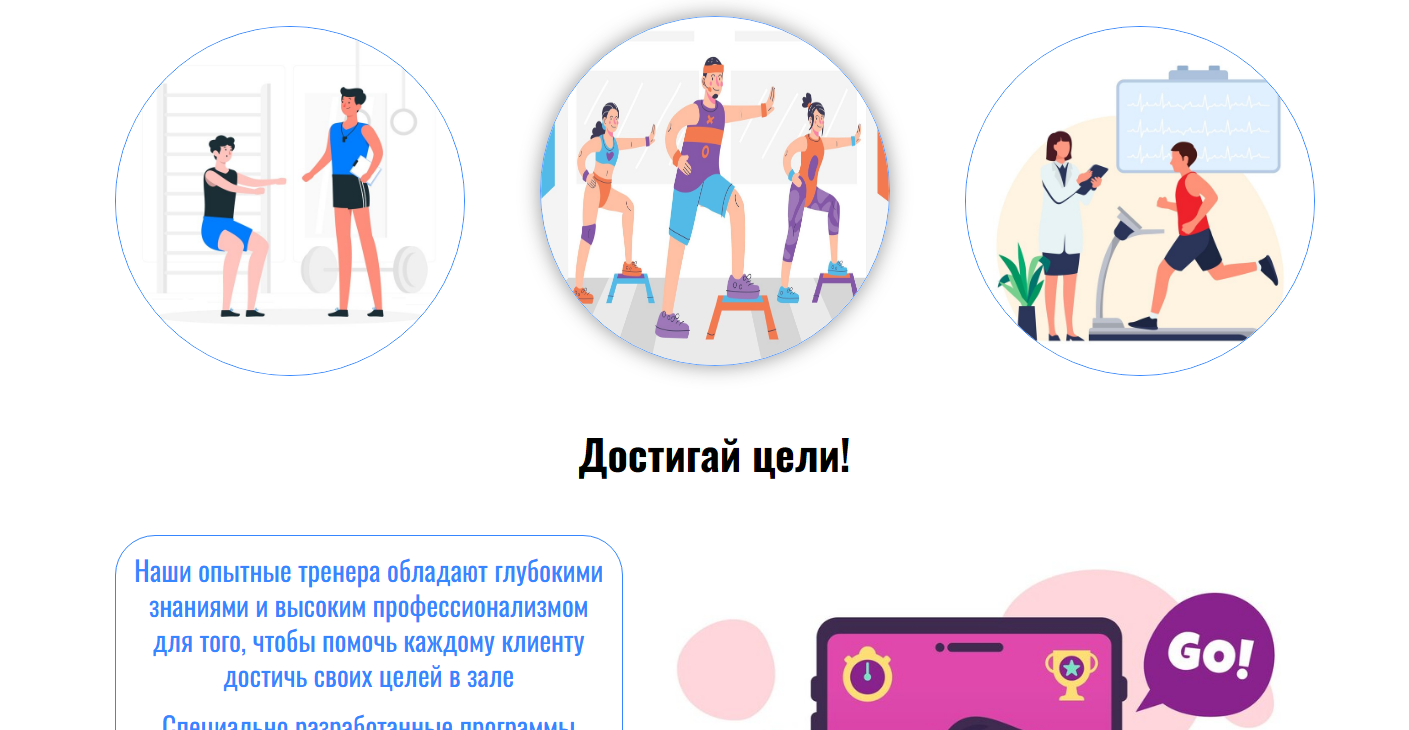
                filter: brightness(1.9)

            &:active

                filter: brightness(1.0)

</style>

### TrainerBanner.vue – баннер для рекламы или внутренней рекламы онлайн ведения.



<template>

  <div class="banner center">

    <div class="banner\_\_container">

      <img

        class="banner\_\_container\_img"

        :src="require(`@/assets/${banner.img}`)"

        alt="banner\_img"

        v-for="banner in banners"

        :key="banner"

      />

    </div>

    <h2 class="banner\_\_subtitle">{{ TrainerBannerText.subtitle }}</h2>

    <div class="banner\_\_bottom">

      <div class="banner\_\_text-container">

        <p class="banner\_\_text">{{ TrainerBannerText.text1 }}</p>

        <p class="banner\_\_text">{{ TrainerBannerText.text2 }}</p>

        <p class="banner\_\_text">{{ TrainerBannerText.text3 }}</p>

      </div>

      <div class="banner\_\_img">

        <img src="@/assets/TrainerBanner.jpg" alt="" />

      </div>

    </div>

  </div>

</template>

<script>

import { mapState } from "vuex";

export default {

  name: "TrainerBanner",

  computed: {

    ...mapState(["banners", "TrainerBannerText"]),

  },

};

</script>

<style lang="sass" scoped>

.banner

    &\_\_container

        display: flex

        flex-direction: row

        justify-content: space-between

        padding-bottom: 60px

        &\_img

            width: 350px

            border-radius: 50%

            border: 1px solid #3586ff

            transition: 0.5s

        &\_img:hover

            transform: translateY(-10px)

            opacity: 0.8

            -webkit-box-shadow: 0px 0px 20px 0px rgba(0, 0, 0, 0.5)

            -moz-box-shadow: 0px 0px 20px 0px rgba(0, 0, 0, 0.5)

            box-shadow: 0px 0px 20px 0px rgba(0, 0, 0, 0.5)

    &\_\_bottom

        display: flex

        flex-direction: row

        justify-content: space-between

    &\_\_subtitle

        text-align: center

        padding-bottom: 64px

    &\_\_text-container

        display: flex

        flex-direction: column

        gap: 16px

        text-align: center

        width: 50%

        padding: 16px

        border-radius: 40px

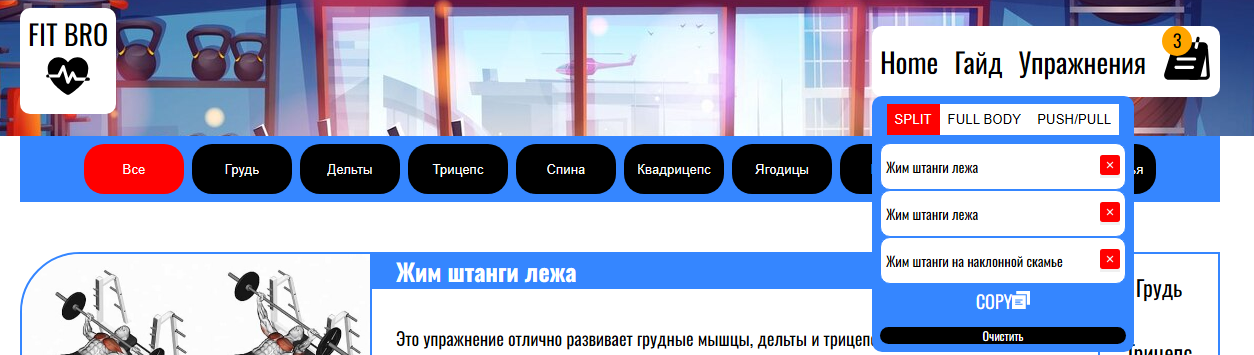
        border: 1px solid #3586ff

    &\_\_text

        color: #3586ff

</style>

### HeaderComp.vue – компонент шапки сайта со списком, лого и навигацией.



Самый мощный компонент, в котором сокрыт основной функционал сайта. Мы можем наблюдать навигацию и список, выполняющий роль корзины, куда пользователь может добавлять/удалять упражнения, а также копировать список.

<template>

  <header class="header center">

    <div class="header\_\_left">

      <p class="header\_\_logo\_text">FIT BRO</p>

      <router-link class="header\_\_logo" to="./">

        <svg

          class="logo"

          fill="#000000"

          width="50px"

          height="50px"

          viewBox="0 0 512 512"

          xmlns="http://www.w3.org/2000/svg"

        >

          <path

            d="M193.69,152.84a16,16,0,0,1,29.64,2.56l36.4,121.36,30-59.92a16,16,0,0,1,28.62,0L345.89,272h96.76A213.08,213.08,0,0,0,464,176.65C463.37,114.54,413.54,64,352.92,64c-48.09,0-80,29.54-96.92,51-16.88-21.49-48.83-51-96.92-51C98.46,64,48.63,114.54,48,176.65A211.13,211.13,0,0,0,56.93,240h93.18Z"

          />

          <path

            d="M321.69,295.16,304,259.78l-33.69,67.38A16,16,0,0,1,256,336q-.67,0-1.38-.06a16,16,0,0,1-14-11.34l-36.4-121.36-30,59.92A16,16,0,0,1,160,272H69.35q14,29.29,37.27,57.66c18.77,22.88,52.8,59.46,131.39,112.81a31.84,31.84,0,0,0,36,0c78.59-53.35,112.62-89.93,131.39-112.81a316.79,316.79,0,0,0,19-25.66H336A16,16,0,0,1,321.69,295.16Z"

          />

          <path

            d="M464,272H442.65a260.11,260.11,0,0,1-18.25,32H464a16,16,0,0,0,0-32Z"

          />

          <path

            d="M48,240a16,16,0,0,0,0,32H69.35a225.22,225.22,0,0,1-12.42-32Z"

          /></svg

      ></router-link>

    </div>

    <div class="header\_\_right">

      <ul class="header\_\_links">

        <router-link

          :to="link.url"

          class="header\_\_link"

          v-for="link in links"

          :key="link.id"

        >

          {{ link.title }}

        </router-link>

      </ul>

      <div class="cart\_\_img" @click="open">

        <svg

          width="50px"

          height="50px"

          viewBox="0 0 24 24"

          fill="none"

          xmlns="http://www.w3.org/2000/svg"

        >

          <path

            d="M20.9009 9.85L21.4909 19.74C21.5109 20.01 21.3809 20.19 21.3109 20.27C21.2309 20.36 21.0609 20.5 20.7809 20.5H18.0509L20.2109 9.85H20.9009ZM22.0009 6L21.9909 6.02C22.0109 6.26 21.9909 6.51 21.9309 6.76L14.5609 20.29C14.3209 21.3 13.4209 22 12.3809 22H20.7809C22.0709 22 23.0909 20.91 22.9909 19.62L22.0009 6Z"

            fill="black"

          />

          <path

            d="M11.4502 2.24136C11.5502 1.84136 11.3002 1.43136 10.9002 1.33136C10.5002 1.24136 10.0902 1.48136 9.99023 1.88136L9.49023 3.95136H11.0302L11.4502 2.24136Z"

            fill="black"

          />

          <path

            d="M18.0509 2.20859C18.1409 1.79859 17.8809 1.40859 17.4709 1.31859C17.0709 1.22859 16.6709 1.48859 16.5809 1.89859L16.1309 3.96859H17.6709L18.0509 2.20859Z"

            fill="black"

          />

          <path

            d="M21.8198 5.33141C21.4898 4.53141 20.7098 3.96141 19.7498 3.96141H17.6698L17.1098 6.55141C17.0298 6.90141 16.7198 7.14141 16.3798 7.14141C16.3298 7.14141 16.2698 7.14141 16.2198 7.12141C15.8198 7.03141 15.5598 6.63141 15.6398 6.23141L16.1298 3.95141H11.0298L10.3998 6.55141C10.3198 6.89141 10.0098 7.12141 9.66975 7.12141C9.60975 7.12141 9.54976 7.11141 9.48975 7.10141C9.08976 7.00141 8.83975 6.60141 8.93975 6.19141L9.47975 3.94141H7.44975C6.46975 3.94141 5.59975 4.58141 5.30975 5.52141L1.09975 19.0714C0.659754 20.5214 1.72975 22.0014 3.23975 22.0014H16.3798C17.4198 22.0014 18.3198 21.3014 18.5598 20.2914L21.9298 6.76141C21.9898 6.51141 22.0098 6.26141 21.9898 6.02141C21.9698 5.78141 21.9198 5.54141 21.8198 5.33141ZM14.6998 16.7514H6.69975C6.28975 16.7514 5.94975 16.4114 5.94975 16.0014C5.94975 15.5914 6.28975 15.2514 6.69975 15.2514H14.6998C15.1098 15.2514 15.4498 15.5914 15.4498 16.0014C15.4498 16.4114 15.1098 16.7514 14.6998 16.7514ZM15.6998 12.7514H7.69975C7.28975 12.7514 6.94975 12.4114 6.94975 12.0014C6.94975 11.5914 7.28975 11.2514 7.69975 11.2514H15.6998C16.1098 11.2514 16.4498 11.5914 16.4498 12.0014C16.4498 12.4114 16.1098 12.7514 15.6998 12.7514Z"

            fill="black"

          />

        </svg>

        <div class="cart\_\_counter" v-if="Cart.length !== 0">

          {{ Cart.length }}

        </div>

      </div>

      <div class="cart" v-show="isShow">

        <div class="cart\_\_adviсes" @click="ANALIZE\_\_ARR">1</div>

        <div class="cart\_\_types" v-if="Cart.length > 0">

          <button

            class="cart\_\_types\_btn"

            v-for="type in TypesList"

            :key="type.id"

            @click="ACTIVE\_\_TYPE(type)"

            :class="{ active: type.id === activeType }"

          >

            {{ type.name }}

          </button>

        </div>

        <div class="cart\_\_item" v-for="item in Cart" :key="item">

          <p cart\_\_item\_title>{{ item.name }}</p>

          <button @click="DEL\_\_EXERCISE(item.id)" class="cart\_\_item\_btn-del">

            <svg

              class="cart\_\_item\_btn-svg"

              width="20px"

              height="20px"

              viewBox="0 0 32 32"

              version="1.1"

              xmlns="http://www.w3.org/2000/svg"

              xmlns:xlink="http://www.w3.org/1999/xlink"

              xmlns:sketch="http://www.bohemiancoding.com/sketch/ns"

            >

              <title>cross-square</title>

              <desc>Created with Sketch Beta.</desc>

              <defs></defs>

              <g

                id="Page-1"

                stroke="none"

                stroke-width="1"

                fill="none"

                fill-rule="evenodd"

                sketch:type="MSPage"

              >

                <g

                  id="Icon-Set-Filled"

                  sketch:type="MSLayerGroup"

                  transform="translate(-206.000000, -1037.000000)"

                  fill="red"

                >

                  <path

                    d="M226.95,1056.54 C227.34,1056.93 227.34,1057.56 226.95,1057.95 C226.559,1058.34 225.926,1058.34 225.536,1057.95 L222,1054.41 L218.464,1057.95 C218.074,1058.34 217.441,1058.34 217.05,1057.95 C216.66,1057.56 216.66,1056.93 217.05,1056.54 L220.586,1053 L217.05,1049.46 C216.66,1049.07 216.66,1048.44 217.05,1048.05 C217.441,1047.66 218.074,1047.66 218.464,1048.05 L222,1051.59 L225.536,1048.05 C225.926,1047.66 226.559,1047.66 226.95,1048.05 C227.34,1048.44 227.34,1049.07 226.95,1049.46 L223.414,1053 L226.95,1056.54 L226.95,1056.54 Z M234,1037 L210,1037 C207.791,1037 206,1038.79 206,1041 L206,1065 C206,1067.21 207.791,1069 210,1069 L234,1069 C236.209,1069 238,1067.21 238,1065 L238,1041 C238,1038.79 236.209,1037 234,1037 L234,1037 Z"

                    id="cross-square"

                    sketch:type="MSShapeGroup"

                  ></path>

                </g>

              </g>

            </svg>

          </button>

        </div>

        <p class="cart\_\_copy" v-if="Cart.length > 0" @click="COPY\_\_CART">

          COPY<svg

            fill="#fff"

            width="18px"

            height="18px"

            viewBox="0 0 1920 1920"

            xmlns="http://www.w3.org/2000/svg"

          >

            <path

              d="M1468.183 451.76v1468.184H0V451.76h1468.183ZM1920 0v1468.296h-338.812V338.812H451.704V0H1920ZM338.812 1016.446h790.56V903.509h-790.56v112.937Zm0 225.874h564.686v-112.937H338.812v112.937Zm0 225.988h790.56v-113.05h-790.56v113.05Z"

              fill-rule="evenodd"

            />

          </svg>

        </p>

        <button

          class="cart\_\_item\_btn-clr"

          v-if="Cart.length !== 0"

          @click="CLEAR\_\_CART"

        >

          Очистить

        </button>

        <p class="cart\_\_text" v-if="Cart.length == 0">

          Добавьте упражнения в список

        </p>

      </div>

    </div>

  </header>

</template>

<script>

import { mapMutations, mapState } from "vuex";

export default {

  name: "HeaderComp",

  data() {

    return {

      isShow: false,

    };

  },

  computed: {

    ...mapState(["links", "Cart", "TypesList", "activeType"]),

  },

  methods: {

    ...mapMutations([

      "DEL\_\_EXERCISE",

      "CLEAR\_\_CART",

      "COPY\_\_CART",

      "ACTIVE\_\_TYPE",

      "ANALIZE\_\_ARR",

    ]),

    open() {

      this.isShow = !this.isShow;

    },

  },

};

</script>

<style lang="sass">

.header

    position: sticky

    top: 0

    z-index: 3

    height: 150px

    background-color: gray

    display: flex

    justify-content: space-between

    align-items: center

    box-sizing: border-box

    background-image: url("@/assets/headerImg.jpg")

    &\_\_left

      display: flex

      flex-direction: column

      align-items: center

      background-color: #fff

      border-radius: 10px

      padding: 8px

    &\_\_logo\_text

      font-family: "Oswald", sans-serif

      font-size: 28px

      line-height: 125%

      color: black

    &\_\_links

      display: flex

      flex-direction: row

      gap: 16px

    &\_\_link

      font-family: "Oswald", sans-serif

      font-size: 28px

      line-height: 125%

      color: black

      transition: 0.4s

    &\_\_right

      position: relative

      z-index: 2

      background-color: #fff

      border-radius: 10px

      padding: 8px

      align-items: center

      gap: 16px

      display: flex

      flex-direction: row

.logo

    transition: 0.4s

.logo:hover

    fill: #3586ff

.logo:active

    fill: red

.header\_\_link:hover

    color: #3586ff

.header\_\_link:active

    color: red

.cart

    position: absolute

    display: flex

    flex-direction: column

    z-index: 1

    width: 262px

    background-color: #3586ff

    top: 70px

    border-radius: 10px

    left: 0

    padding: 8px

    &\_\_item

      display: flex

      align-items: center

      justify-content: space-between

      border: 1px solid #3586ff

      border-radius: 10px

      padding: 5px

      background-color: white

      font-size: 14px

      &\_btn-del

        border: none

      &\_btn-svg path

        transition: 0.4s

      &\_btn-svg path:hover

        fill: #3586ff

      &\_btn-svg path:active

        fill: black

      &\_btn-clr

        margin-top: 8px

        border: none

        border-radius: 10px

        background-color: black

        color: white

        font-family: "Oswald", sans-serif

        font-size: 12px

        transition: 0.4s

      &\_btn-clr:hover

        background-color: red

      &\_btn-clr:active

        background-color: #3586ff

      &\_\_text

        display: flex

        text-align: center

        font-size: 22px

        color: white

.cart\_\_img

    position: relative

.cart\_\_counter

    display: flex

    justify-content: center

    align-items: center

    position: absolute

    top: -8px

    width: 30px

    height: 30px

    border-radius: 50%

    background-color: orange

    font-size: 18px

.cart\_\_copy

    text-align: center

    font-size: 18px

    color: white

.cart\_\_types

    display: flex

    justify-content: center

    padding-bottom: 8px

    &\_btn

      transition: 0,5s

      border: none

      background-color: #fff

      padding: 8px

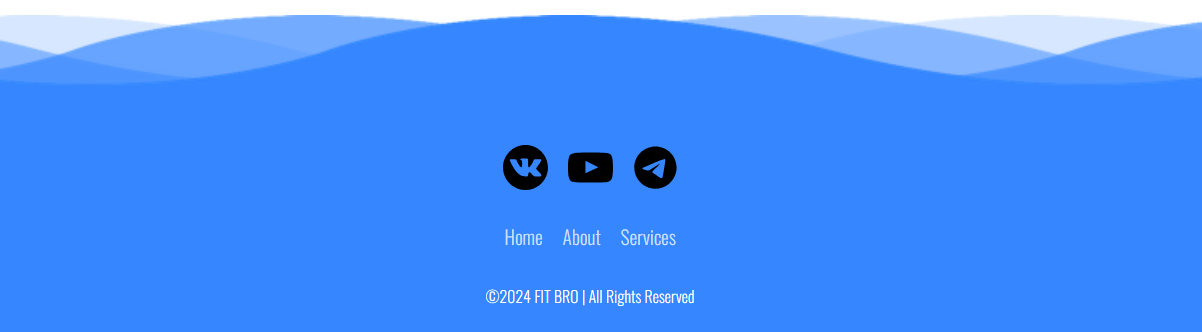
.active

  background-color: red

  color: white

</style>

### FooterComp.vue – компонент подвала сайта с навигацией.



<template>

  <footer class="footer center">

    <div class="waves">

      <div class="wave" :id="wave.id" v-for="wave in waves" :key="wave"></div>

    </div>

    <ul class="social-icon">

      <li class="social-icon\_\_item">

        <a

          class="social-icon\_\_link"

          :href="icon.url"

          v-for="icon in icons"

          :key="icon"

        >

          <img

            class="social-icon\_\_img"

            :src="require(`@/assets/svg/${icon.img}`)"

            alt="icon.name"

          />

        </a>

      </li>

    </ul>

    <ul class="menu">

      <li class="menu\_\_item" v-for="link in linksFooter" :key="link">

        <a class="menu\_\_link" :href="link.url">{{ link.title }}</a>

      </li>

    </ul>

    <p>&copy;2024 FIT BRO | All Rights Reserved</p>

  </footer>

</template>

<script>

import { mapState } from "vuex";

export default {

  name: "FooterComp",

  computed: {

    ...mapState(["waves", "linksFooter", "icons"]),

  },

};

</script>

<style lang="sass" scoped>

@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700;800;900&display=swap")

.footer

  position: relative

  width: 100%

  background: #3586ff

  min-height: 100px

  padding: 20px 50px

  display: flex

  justify-content: center

  align-items: center

  flex-direction: column

  margin-top: 150px

.social-icon,

.menu

  position: relative

  display: flex

  justify-content: center

  align-items: center

  margin: 10px 0

  flex-wrap: wrap

.social-icon\_\_img

    width: 45px

    height: 45px

.social-icon\_\_item,

.menu\_\_item

  list-style: none

.social-icon\_\_link

  font-size: 2rem

  color: #fff

  margin: 0 10px

  display: inline-block

  transition: 0.5s

.social-icon\_\_link:hover

  transform: translateY(-10px)

.social-icon\_\_link:active

  transform: scale(1, 0.5)

.menu\_\_link

  font-size: 1.2rem

  color: #fff

  margin: 0 10px

  display: inline-block

  transition: 0.5s

  text-decoration: none

  opacity: 0.75

  font-weight: 300

.menu\_\_link:hover

  opacity: 1

.footer p

  color: #fff

  margin: 15px 0 10px 0

  font-size: 1rem

  font-weight: 300

.wave

  position: absolute

  top: -100px

  left: 0

  width: 100%

  height: 100px

  background: url("https://i.ibb.co/wQZVxxk/wave.png")

  background-size: 1000px 100px

.wave#wave1

  z-index: 1000

  opacity: 1

  bottom: 0

  animation: animateWaves 4s linear infinite

.wave#wave2

  z-index: 999

  opacity: 0.5

  bottom: 10px

  animation: animate 4s linear infinite !important

.wave#wave3

  z-index: 1000

  opacity: 0.2

  bottom: 15px

  animation: animateWaves 3s linear infinite

.wave#wave4

  z-index: 999

  opacity: 0.7

  bottom: 20px

  animation: animate 3s linear infinite

@keyframes animateWaves

  0%

    background-position-x: 1000px

  100%

    background-positon-x: 0px

@keyframes animate

  0%

    background-position-x: -1000px

  100%

    background-positon-x: 0px

</style>

### ExercisesComp.vue – компонент, который хранит базы упражнений с фильтром.

<template>

  <div class="Exercises center">

    <div class="Exercises\_\_filter">

      <button

        class="Exercises\_\_filter\_btn"

        v-for="item in Filters"

        :key="item.id"

        @click="FILTER\_\_ARR(item)"

        :class="{ active: item.id === activeButton }"

      >

        {{ item.text }}

      </button>

    </div>

    <div class="Exercises\_\_container">

      <div

        class="Exercises\_\_item"

        v-for="exercise in FilterExercises"

        :key="exercise.id"

      >

        <img

          class="Exercises\_\_item\_img"

          :src="require(`@/assets/exercises/${exercise.img}`)"

          alt="Ex\_img"

        />

        <div class="Exercises\_\_item\_\_block">

          <h2 class="Exercises\_\_item\_\_block\_title">{{ exercise.name }}</h2>

          <p class="Exercises\_\_item\_\_block\_text">{{ exercise.desc }}</p>

          <a

            class="Exercises\_\_item\_\_block\_link"

            :href="exercise.url"

            target="\_blank"

            >Техника выполнения</a

          >

        </div>

        <div class="Exercises\_\_item\_\_block\_teg">

          <ul class="Exercises\_\_item\_\_block\_ul">

            <li

              class="Exercises\_\_item\_\_block\_li"

              v-for="teg in exercise.tegs"

              :key="teg"

            >

              {{ teg }}

            </li>

          </ul>

        </div>

        <button @click="ADD\_\_EXERCISE({ exercise })" class="Exercises\_\_btn">

          ДОБАВИТЬ

        </button>

      </div>

    </div>

  </div>

</template>

<script>

import { mapState, mapMutations } from "vuex";

export default {

  name: "ExercisesComp",

  data() {

    return {};

  },

  computed: {

    ...mapState(["Exercises", "Filters", "FilterExercises", "activeButton"]),

  },

  methods: {

    ...mapMutations(["ADD\_\_EXERCISE", "FILTER\_\_ARR"]),

  },

};

</script>

<style lang="sass" scoped>

a:link

    color: #3586ff

a:visited

    color: #3586ff

a:hover

    color: black

a:active

    color: red

.Exercises

    padding-bottom: 96px

    &\_\_filter

      background-color: #3586ff

      z-index: 2

      width: 100%

      position: sticky

      top: 150px

      padding: 8px

      display: flex

      flex-wrap: wrap

      gap: 8px

      justify-content: center

      &\_btn

        transition: 0.4s

        border-radius: 20px

        background-color: black

        border: none

        width: 100px

        height: 50px

        color: white

    &\_\_container

        display: flex

        flex-direction: column

        gap: 50px

        margin-top: 50px

    &\_\_item

        position: relative

        display: flex

        height: 200px

        border: 2px solid #3586ff

        border-radius: 60px 0px 0px 60px

        box-sizing: border-box

        &\_img

          border-right: 2px solid #3586ff

          max-width: 350px

          border-radius: 60px 0px 0px 60px

        &\_\_block

            display: flex

            flex-direction: column

            justify-content: space-between

            &\_title

              display: flex

              padding-left: 24px

              display: flex

              align-items: center

              text-align: left

              background-color: #3586ff

              font-size: 24px

              color: white

            &\_text

                display: flex

                padding-top: 8px

                padding-left: 24px

                font-size: 18px

            &\_teg

                display: flex

            &\_ul

                display: flex

                flex-direction: column

                align-items: center

                justify-content: center

                gap: 30px

                width: 120px

                border-left: 2px solid #3586ff

            &\_li

                font-size: 22px

            &\_link

                display: flex

                justify-content: center

                font-size: 16px

    &\_\_btn

        font-family: "Oswald", sans-serif

        font-size: 20px

        line-height: 125%

        color: black

        height: 80px

        width: 160px

        border: none

        color: white

        border-radius: 60px

        background-color: #3586ff

        position: absolute

        top: 28%

        left: 7%

        opacity: 0

        transition: 0.4s

    &\_\_item:hover .Exercises\_\_btn

        opacity: 1

    &\_\_btn:hover

        background-color: black

    &\_\_btn:hover .Exercises\_\_item\_img

        opacity: 0.5

    &\_\_btn:active

        background-color: red

.active

  background-color: red

</style>

Компонент взаимодействует с vuex, где хранится база упражнений и мутации для функционала с корзиной/списком.

## Тестирование работоспособности - UI/UX

UI (пользовательский интерфейс) и UX (пользовательский опыт) — это два термина, которые часто используются вместе для описания дизайна и разработки цифровых продуктов, таких как веб-сайты, мобильные приложения и программное обеспечение.

Пользовательский интерфейс относится к визуальным и интерактивным элементам продукта, с которыми взаимодействует пользователь. Сюда входят такие вещи, как кнопки, меню, значки, текст, цвета и анимация. Цель дизайна пользовательского интерфейса — создать интерфейс, который будет визуально привлекательным, простым в навигации и интуитивно понятным в использовании.

UX, с другой стороны, относится к общему опыту пользователя при взаимодействии с продуктом. Это включает в себя все, от простоты использования до эмоционального отклика пользователя при использовании продукта. Цель UX-дизайна — создать продукт, удобный для пользователя, привлекательный и отвечающий потребностям и ожиданиям пользователя.

В процессе разработки сайта были выявлены следующие UI/UX проблемы:

1.Невозможность поиска конкретного упражнения по названию. Решается добавлением функционала по поиску внутри массива с упражнениями.

2.Не сохраняется список корзины при обновлении сайта. Решается добавлением функционала LocalStorage. Что было полностью реализовано.

3.Отсуствие регистрации. “Не баг, а фича”, в целом проект не подразумевает регистрации пользователя, функционал веб-приложения хоть и полезный, но буквально не нуждается в том, чтобы пользователь заводил аккаунт. Даже в случае, если пользователь хочет оформить онлайн-ведение, то есть приобрести коммерческую услугу, он может заполнить форму, где получит обратную связь по почте, а не внутри экосистемы сайта.

# Подведение итогов, оценка результатов и заключение

В результате успешно выполненного дипломного проекта было создано полноценное веб-приложение, позволяющее пользователю получить базовые знания о тренировочном процессе и его видах, реализован функционал поиска упражнений и добавления их в список, с возможностью последующего копирования.

В данном дипломном проекте был разработан веб-агрегатор информации с использованием технологий HTML, CSS, JavaScript, Vue.js.

Одним из главных достижений проекта является умение правильно использовать различные инструменты и технологии для достижения поставленных целей. Кроме того, были применены современные подходы и методологии в разработке, что позволило получить высокое качество и удобство использования приложения.

Frontend был разработан с использованием Vue.js, а backend был реализован с помощью библиотеки Vuex для симуляции взаимодействия с серверами. Также, за счет модульности самого сайта в дальнейшем можно расширять функционал сайта во многих направлениях.

В процессе разработки проекта были рассмотрены основные принципы и подходы к созданию веб-приложений, включая UI/UX дизайн, frontend и backend разработку, а также использование различных технологий для достижения поставленных целей.

Для успешной разработки веб-приложений необходимы хорошие знания и практические навыки работы с технологиями HTML, CSS, JavaScript, Vue.js. Для начинающих разработчиков рекомендуется изучение соответствующих руководств, видеоуроков и книг по каждой из этих технологий.

Мы наблюдаем приложение с большим потенциалом к масштабированию, поскольку тема фитнеса и составления тренировочного плана весьма обширна. Имеется коммерческий потенциал в виде онлайн-ведения клиентов тренерами, реклама БАД и иной фитнес продукции. Функционал, который может быть добавлен при дальнейшей работе над проектом:

1.Поиск по упражнениям – не был добавлен ввиду отсутствия надобности в данный момент, список упражнений, среди которых пользователь может ввести поиск хоть и хорош по своей выборке, но не так велик, особенно при наличии фильтрующий тегов.

2.Нативные подсказки – которые бы появлялись возле списка/корзины в определенных случаях. Например, происходил бы анализ корзины на наличие упражнений, которые имеют одинаковые теги – они в то же время отвечают за группы мышц, участвующие в выполнении упражнения. Если упражнений на одну группу мышц больше допустимого количества, мы бы могли сказать об этом пользователю, что потенциально уберегло бы его от перетренированности и травматизации, а также помогло улучшить темпы мышечного роста. Данные подсказки позволили бы автоматически давать возможность пользователю корректировать список упражнений в лучшую сторону.

В целом, проект был успешно завершен и достиг своей цели – создания веб-приложения помогающего составить план тренировок в спортивном зале.

# Список использованной литературы

**1.HTML, CSS, JS**

1. W3Schools - https://www.w3schools.com/

2. Codecademy - https://www.codecademy.com/learn/learn-html и по JS - https://www.codecademy.com/learn/introduction-to-javascript

3. FreeCodeCamp - https://www.freecodecamp.org/

4. MDN Web Docs - https://developer.mozilla.org/en-US/docs/Learn

5. CSS-Tricks - https://css-tricks.com/

6. HTML Dog - https://htmldog.com/

7. Learn to Code HTML & CSS - https://learn.shayhowe.com/html-css/

8. TutorialsPoint - https://www.tutorialspoint.com/index.htm

9. SoloLearn - https://www.sololearn.com/

10. Документация на русском языке JS - https://learn.javascript.ru/

12. https://gb.ru/posts/chto-takoe-css-obyasnyaem-prostymi-slovami

13. https://developer.mozilla.org/ru/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript

14. <https://www.youtube.com/@VladilenMinin> - html, css, js

**2.Vue.js**

1. <https://www.youtube.com/@dmitrylavrik> – Vue.js

2. ITVDN - <https://itvdn.com/ru> - frontend разработка

3. <https://v3.router.vuejs.org/ru/> - руководство Vue Router

4. <https://vuex.vuejs.org/ru/> - руководство Vuex

5. [https:/vuejs.org/ru/](https://v3.router.vuejs.org/ru/) - руководство Vue.js

6. <https://habr.com/ru/articles/421551/> - Vue.js

**3. GitHub**

1. <https://github.com/Empinado/GW/tree/main/gw_v2/project> - GitHub ссылка