**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**Факультет: Информационные технологии и управление**

**Кафедра: Информационные технологии**

**Предмет: «Веб-системы и технологии»**

курсовая работа

**Тема** «Разработка онлайн информационной системы для вокзалов»

**Группа:** 680.22

**Курc:** 3

**Специальность: 050616** Информационные технологии

**Студент**: Амиров Амир

**Руководитель:** **асс. Халилов М. Э.**

**Баку – 202****5**

**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**ЗАДАНИЕ ПО КУРСОВОЙ РАБОТЕ**

**ФАКУЛЬТЕТ**: Информационные технологии и управление

**КАФЕДРА**: Информационные технологии

**Группа** 680.22 **курс** III

**Специальность** **050616** Информационные технологии

**Студент** Амиров Амир

**Зачетная книжка**

**Руководитель курсовой работы**  **асс. Халилов М. Э.**

**Срок выдачи**

**Дата сдачи**

**Тема курсовой работы «**Разработка онлайн информационной

системы для вокзалов»

**Отзыв руководителя курсовой**:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись студента**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись рук.курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись зав.кафедрой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дата защиты курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Оценка**\_\_\_\_\_\_\_\_\_\_

**Пред.комиссии**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**Члены комиссии**:1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc133637923)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 6](#_Toc133637924)

[1.1 Обзор веб-систем и технологий 6](#_Toc133637925)

[1.2 Архитектура и дизайн веб-систем 8](#_Toc133637926)

[1.3 Инструменты и фреймворки для веб-разработки 10](#_Toc133637927)

[1.4 Веб-безопасность 13](#_Toc133637928)

[1.5 Оптимизация веб-производительности 16](#_Toc133637929)

[2. ПРАКТИЧЕСКАЯ ЧАСТЬ 18](#_Toc133637930)

[2.1 Описание предметной области 18](#_Toc133637931)

[2.2 Начало разработки 19](#_Toc133637932)

[2.3 Перенос макета 20](#_Toc133637933)

[2.4 Изменения в макете 21](#_Toc133637934)

[2.5 Создание секции 22](#_Toc133637935)

[2.6 Заполнение контента 26](#_Toc133637936)

[3. РЕЗУЛЬТАТ 34](#_Toc133637937)

[4. ЛИТЕРАТУРА 35](#_Toc133637938)

[5. ПРИЛОЖЕНИЯ 36](#_Toc133637939)

**ВВЕДЕНИЕ**

Использование веб-систем и технологий стало важным аспектом современного общества, облегчая общение, образование, торговлю и развлечения. Интернет превратился из основного средства связи в сложную и динамичную платформу, на которой работают различные веб-службы и приложения. Веб-системы и технологии охватывают широкий спектр инструментов, языков и платформ, которые используются для разработки, развертывания и обслуживания веб-приложений. Эти технологии со временем развивались, чтобы не отставать от растущего спроса на веб-сервисы, что привело к появлению большого разнообразия языков программирования, фреймворков, библиотек и баз данных.

Сегодня использование веб-систем и технологий широко распространено, и организации всех размеров используют Интернет для охвата глобальной аудитории, расширения своего присутствия в Интернете и оптимизации бизнес-операций. Веб-системы и технологии, от платформ социальных сетей до веб-сайтов электронной коммерции, составляют основу цифровой экономики.

Веб-системы и технологии охватывают различные дисциплины, такие как веб-разработка, веб-дизайн, веб-безопасность, веб-аналитика и веб-приложения. Эти технологии используются для создания различных веб-приложений, включая статические веб-сайты, динамические веб-приложения, системы управления контентом, веб-сайты электронной коммерции и платформы социальных сетей.

Разработка веб-приложений является жизненно важным компонентом веб-систем и технологий с использованием таких языков программирования, как HTML, CSS, JavaScript, и серверных языков сценариев, таких как PHP, Python, Ruby и Java. Эти языки программирования обеспечивают основу для разработки интерактивных и динамических веб-приложений, которые могут реагировать на вводимые пользователем данные и предоставлять контент по запросу.

Веб-безопасность — еще один важный аспект веб-систем и технологий. Поскольку конфиденциальная информация все чаще распространяется через Интернет, безопасность в Интернете является серьезной проблемой как для отдельных лиц, так и для организаций. Технологии веб-безопасности, такие как SSL, HTTPS и брандмауэры, используются для защиты веб-приложений от несанкционированного доступа, утечки данных и других угроз безопасности.

Таким образом, веб-системы и технологии являются фундаментальной частью современного общества, определяя то, как мы общаемся, учимся, работаем и играем. Поскольку Интернет продолжает развиваться и все больше интегрируется в нашу повседневную жизнь, понимание этих технологий и их приложений имеет решающее значение для отдельных лиц и организаций, стремящихся добиться успеха в цифровой экономике.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Обзор веб-систем и технологий

Веб-системы и технологии стали неотъемлемой частью нашей повседневной жизни. От социальных сетей до электронной коммерции веб-системы используются для самых разных приложений. В этом разделе мы определим и объясним, что такое веб-системы и технологии, включая их историю, эволюцию и текущее состояние.

**Определение и объяснение:**

Веб-системы и технологии представляют собой набор инструментов, сред и языков, используемых для разработки и эксплуатации веб-приложений. Эти приложения могут варьироваться от простых веб-сайтов до сложных сетевых программных систем. Веб-системы обычно состоят из клиентского интерфейса, с которым взаимодействуют пользователи, и серверного компонента, обрабатывающего запросы и ответы.

Историю веб-систем и технологий можно проследить до первых дней Интернета, когда были разработаны первые веб-страницы. Первая веб-страница была создана Тимом Бернерсом-Ли в 1991 году и состояла из простого текста и ссылок. Со временем возможности веб-систем и технологий росли, и разработчики начали включать в свои веб-страницы изображения, видео и другие мультимедиа.

В начале 2000-х появились динамические веб-приложения, которые могли взаимодействовать с пользователями и реагировать на их действия в режиме реального времени. Это стало возможным благодаря развитию веб-технологий, таких как AJAX, которые позволяли обновлять веб-страницы без перезагрузки всей страницы.

В последние годы веб-системы и технологии продолжали развиваться с появлением новых технологий, таких как HTML5, CSS3 и фреймворков JavaScript, таких как React и Angular. Эти технологии позволили разработать высокоинтерактивные веб-приложения, которые могут конкурировать по возможностям с программным обеспечением для настольных компьютеров.

**Текущее состояние:**

Сегодня веб-системы и технологии используются для широкого спектра приложений, от платформ социальных сетей до веб-сайтов электронной коммерции. Веб-системы также используются в таких отраслях, как здравоохранение, финансы и образование, где они используются для предоставления услуг и приложений пользователям.

Одной из ключевых тенденций в веб-системах и технологиях сегодня является переход к облачным сервисам. Облачные сервисы позволяют развертывать веб-приложения и управлять ими на удаленных серверах, что может обеспечить масштабируемость и экономию средств для организаций.

Еще одна тенденция в веб-системах и технологиях — повышенное внимание к безопасности. С ростом использования веб-приложений безопасность стала критической проблемой, и разработчики включают в свои веб-приложения функции безопасности, такие как шифрование, аутентификация и контроль доступа.

В заключение, веб-системы и технологии являются неотъемлемой частью нашей повседневной жизни. Они превратились из простых текстовых веб-страниц в интерактивные веб-приложения. Сегодня веб-системы используются в самых разных приложениях, и основное внимание уделяется облачным сервисам и безопасности. С продолжающейся эволюцией веб-систем и технологий мы можем ожидать появления новых интересных разработок в будущем.

1.2 Архитектура и дизайн веб-систем

Архитектура и дизайн веб-систем являются важными аспектами их функциональности и производительности. Эти системы состоят из нескольких компонентов, которые работают вместе для предоставления определенной услуги или функции. Понимание архитектуры и дизайна веб-систем имеет важное значение для веб-разработчиков, поскольку позволяет им создавать эффективные и действенные системы, отвечающие требованиям пользователей.

Одним из фундаментальных компонентов архитектуры веб-системы является модель клиент-сервер. В этой модели веб-система разделена на две части: клиентскую и серверную. Клиентская сторона относится к пользовательскому интерфейсу, где пользователи взаимодействуют с веб-системой через веб-браузер или мобильное приложение. С другой стороны, серверная часть относится к серверной части веб-системы, где данные обрабатываются и хранятся.

Серверы веб-приложений являются еще одним важным компонентом архитектуры веб-системы. Эти серверы предоставляют платформу для запуска веб-приложений, обработки запросов от клиентов и управления потоком данных между клиентской и серверной сторонами. Серверы веб-приложений обычно предназначены для обработки больших объемов трафика и обеспечения масштабируемости, чтобы веб-системы могли приспосабливаться к росту и увеличению спроса.

Системы баз данных также являются жизненно важным компонентом архитектуры веб-систем. Эти системы используются для хранения и управления данными, которые используются веб-приложениями. Системы баз данных обеспечивают структурированный и организованный способ хранения данных, упрощая их извлечение и использование. Некоторые из самых популярных систем управления базами данных, используемых в веб-разработке, включают MySQL, PostgreSQL и MongoDB.

Архитектура и дизайн веб-систем также должны учитывать различные факторы, такие как безопасность, производительность и удобство использования. Механизмы безопасности, такие как шифрование, аутентификация и контроль доступа, необходимы для защиты веб-систем от киберугроз и обеспечения безопасности пользовательских данных. Методы оптимизации производительности, такие как кэширование, сжатие и сети доставки контента (CDN), используются для повышения скорости и отклика веб-систем, обеспечивая бесперебойную работу пользователей.

Удобство использования — еще один важный фактор проектирования веб-системы, поскольку он определяет, насколько легко пользователи могут взаимодействовать с системой. Веб-разработчики должны обеспечить, чтобы пользовательский интерфейс был интуитивно понятным и простым в навигации, предоставляя пользователям приятную и эффективную работу.

В заключение, архитектура и дизайн веб-систем сложны и многогранны. Эти системы состоят из различных компонентов, которые работают вместе для обеспечения определенной функции или услуги. Понимание архитектуры и дизайна веб-систем имеет важное значение для веб-разработчиков, поскольку позволяет им создавать эффективные и действенные системы, отвечающие требованиям пользователей. Принимая во внимание такие факторы, как безопасность, производительность и удобство использования, веб-разработчики могут создавать безопасные, быстро реагирующие и удобные для пользователя веб-системы.

1.3 Инструменты и фреймворки для веб-разработки

Фреймворк (с англ. framework – «каркас, структура») — это набор ИТ-инструментов, который упрощает разработку и поддержку сложных и высоконагруженных веб-проектов. Это промежуточный вариант по гибкости и сложности между написанием кода с нуля и использованием CMS. Фреймворк содержит базовые программные модули — специфичные же компоненты создают ИТ-специалисты, но на основе этих модулей.

Фреймворк — это программный каркас. Для удобства, приведем простую аналогию. Представьте: для приготовления любимой лазаньи вы самостоятельно замешиваете тесто, придаете особую форму, сушите и нарезаете пласты пасты перед готовкой. Но можно поступить проще: купить в магазине готовые листы, выложить пасту слоями вместе с фаршем, сыром и соусом бешамель и отправить в духовку запекаться. Второй случай — пример, как работает фреймворк.

У инструмента есть постоянная часть, которая остается неизменной вне зависимости от конфигурации. Есть и переменная — это различные модули и компоненты, которые нужны для кастомизации конечной системы.

Библиотека — это набор подпрограмм, который не влияет на архитектуру ИТ-продукта, не ставит никаких ограничений. Пользователю достаточно вызвать нужную команду, чтобы выполнить какие-то действия, упростив работу. Примеры библиотек: React, jQuery.

Фреймворки же накладывают жёсткие рамки на правила проектирования. А внутри него — множество библиотек.

Паттерн проектирования — это способ построения программного кода специальным образом, метод решения проблемы. Он помогает правильно выстроить ИТ-продукт. Часто паттерн проектирования лежит в основе фреймворков, позволяя реализовать модель MVC (Model-View-Controller), т.е. разделить логику продукта на три независимых части:

* Model — получает информацию от контроллера, делает некоторые операции и передает их дальше во View;
* View — обрабатывает и отправляет данные пользователю в удобном виде;
* Controller — проверяет информацию и передает ее в Model.

Любой фреймворк на различных языках упрощает жизнь программиста при создании архитектуры, разработке и поддержке проекта. В фреймворках на архитектурном уровне заложено множество принципов оптимизации. Рассмотрим основные преимущества ниже.

* **Более высокая производительность.** Фреймворки разработаны с использованием готовых функций для обеспечения высокой эффективности и производительности при создании ИТ-решений. Скорость загрузки веб-сайтов и программ, разработанных с их использованием, значительно возрастает.
* **Сокращение количества ошибок.** Большинство фреймворков включают в себя лучшие практики разработки программного обеспечения. Многие из них имеют встроенный механизм тестирования, который проверяет код сразу, уменьшая количество ошибок в конечном коде.
* **Быстрое развитие проекта или продукта.** У фреймворков есть заранее написанные шаблоны, которые можно использовать для выполнения избыточных задач программирования. Эти инструменты экономят время разработчиков и позволяют им сосредоточиться на более приоритетных для команды действиях, обеспечивая более быстрые результаты.
* **Надежность и безопасность**. Фреймворки включают в себя сотни готовых компонентов, созданных и регулярно обновляемых сообществом разработчиков. Это своего рода гарантия, что ваш проект — актуальное и одно из лучших решений бизнеса.

**Типы фреймворков в веб-разработке**

Каждое веб-приложение включает в себя интерфейсную часть (или фронтенд) и серверную (бэкенд). Это и определило два вида веб-фреймворков, доступных как для внешнего интерфейса, так и для «начинки» продукта.

* **Фронтенд фреймворки**

Эти фреймворки используются для разработки внешнего (пользовательского) интерфейса. Они чаще всего основаны на таких языках программирования, как JavaScript, HTML и CSS.

Среди решаемых задач — разработка дизайна UX/UI, SEO-оптимизация, фрагменты кода, шаблоны, управление взаимодействием с пользователем и многое другое. А среди самых известных фронтенд фреймворков — React, Vue.js, Ember, Bootstrap и Angular.

* **Бэкэнд фреймворки**

Эти фреймворки используются для разработки серверной части, которая отвечает за функционирование ИТ-продукта. Они основаны на таких языках программирования, как Python, .NET, Ruby, Java и PHP.

Среди решаемых задач — функционирование сервера и базы данных, протоколы маршрутизации, логика и архитектура сервиса, параметры авторизации, безопасность и многие другие. Среди наиболее популярных фреймворков — Django, Laravel, Ruby On Rails, Spring, Express, ASP.NET Core.

1.4 Веб-безопасность

Веб-безопасность является критическим аспектом веб-систем и технологий. С ростом использования Интернета и веб-приложений риски безопасности стали более значительными. Нарушение безопасности в веб-приложении может нанести серьезный ущерб организации, включая потерю конфиденциальной информации, финансовые потери и ущерб репутации организации. Следовательно, важно обеспечить защиту веб-систем от различных угроз. В этом разделе мы обсудим важность веб-безопасности, различные риски безопасности, с которыми сталкиваются веб-системы, и различные механизмы безопасности.

**Важность веб-безопасности:**

Веб-безопасность необходима по нескольким причинам. Во-первых, веб-приложения часто содержат конфиденциальную информацию, такую как личная информация, финансовые данные и интеллектуальная собственность. Если эта информация попадет в чужие руки, она может быть использована в злонамеренных целях, таких как кража личных данных, финансовое мошенничество и корпоративный шпионаж. Во-вторых, веб-приложения часто используются для выполнения критически важных функций, таких как онлайн-банкинг, электронная коммерция и медицинские услуги. Нарушение безопасности в этих приложениях может привести к значительным финансовым потерям и причинить вред отдельным лицам. Наконец, веб-безопасность необходима для защиты от различных типов киберугроз, таких как вредоносное ПО, фишинг и атаки типа «отказ в обслуживании».

**Риски безопасности в веб-системах:**

Веб-системы сталкиваются с рядом угроз безопасности, включая следующие:

* Атаки путем внедрения. Атаки путем внедрения включают использование уязвимостей в веб-приложениях для внедрения вредоносного кода или команд в базу данных или другие части системы. Атаки SQL-инъекций являются распространенным типом атак путем внедрения.
* Межсайтовый скриптинг (XSS): атаки XSS включают в себя внедрение вредоносного кода на веб-страницу, который затем выполняется веб-браузером пользователя. Эти атаки могут использоваться для кражи конфиденциальной информации или выполнения других вредоносных действий.
* Подделка межсайтовых запросов (CSRF). Атаки CSRF заключаются в том, чтобы заставить пользователя выполнить непреднамеренное действие в веб-приложении. Например, пользователя могут заставить щелкнуть ссылку, которая выполняет вредоносное действие в веб-приложении, в которое он вошел.
* Вредоносное ПО. Вредоносное ПО представляет собой распространенную угрозу для веб-систем, которая может быть представлена через вложения электронной почты, зараженные веб-сайты или другими способами. Вредоносное ПО может использоваться для кражи конфиденциальной информации или выполнения других вредоносных действий.

**Механизмы безопасности:**

Для защиты веб-систем от этих угроз безопасности можно использовать различные механизмы безопасности. Некоторые из этих механизмов включают:

* Шифрование: Шифрование — это процесс преобразования конфиденциальной информации в нечитаемый формат, который можно расшифровать только с помощью определенного ключа. Это гарантирует, что даже если информация будет перехвачена, она не сможет быть прочитана посторонними лицами.
* Аутентификация: Аутентификация — это процесс проверки личности пользователя или системы. Обычно это делается с помощью имен пользователей и паролей, но также может включать более сложные методы, такие как биометрическая аутентификация.
* Контроль доступа. Контроль доступа включает контроль за тем, кто имеет доступ к конфиденциальной информации и ресурсам в веб-системе. Это можно сделать с помощью различных методов, включая управление доступом на основе ролей и управление доступом на основе атрибутов.
* Брандмауэры: Брандмауэры — это распространенный механизм безопасности, используемый для защиты веб-систем от различных типов атак. Брандмауэры можно использовать для блокировки входящего трафика из известных вредоносных источников и предотвращения несанкционированного доступа к конфиденциальной информации.

В заключение, веб-безопасность необходима для защиты веб-систем и конфиденциальной информации, которую они содержат. Веб-системам угрожают различные риски безопасности, включая атаки путем внедрения, атаки XSS, атаки CSRF и вредоносное ПО. Для защиты от этих рисков могут использоваться различные механизмы безопасности, такие как шифрование, аутентификация, контроль доступа и брандмауэры. Внедрив эти механизмы безопасности, веб-системы можно защитить от различных типов киберугроз, обеспечив конфиденциальность, целостность и доступность конфиденциальной информации.

1.5 Оптимизация веб-производительности

Оптимизация веб-производительности является критическим аспектом веб-систем и технологий. С ростом использования Интернета и веб-приложений производительность веб-систем стала важнее, чем когда-либо. Медленный или неотзывчивый веб-сайт может привести к разочарованию пользователей, повышению показателя отказов и снижению вовлеченности. В этом разделе мы обсудим стратегии, используемые для оптимизации производительности веб-систем, включая такие методы, как кэширование, сжатие и сети доставки контента (CDN). Мы также коснемся важности адаптивного веб-дизайна и того, как он влияет на производительность.

**Стратегии оптимизации веб-производительности:**

Существует несколько стратегий, которые можно использовать для оптимизации производительности веб-систем. Эти стратегии включают следующее:

* Кэширование: Кэширование — это процесс хранения часто используемых данных в памяти или на диске. Это может помочь сократить время, необходимое для загрузки страницы, и повысить общую производительность. Общие методы кэширования включают кэширование браузера, кэширование на стороне сервера и кэширование сети доставки контента.
* Сжатие. Сжатие — это процесс уменьшения размера файлов, таких как изображения и скрипты, для сокращения времени, необходимого для их передачи через Интернет. Общие методы сжатия включают gzip и deflate.
* Сети доставки контента (CDN): CDN — это сети серверов, расположенных по всему миру, которые используются для доставки контента пользователям с ближайшего к ним сервера. Это может помочь сократить время, необходимое для загрузки страницы, и повысить общую производительность.
* Минификация: Минификация — это процесс удаления ненужных символов, таких как пробелы и комментарии, из файлов кода для уменьшения их размера и повышения производительности.
* Оптимизация изображений. Изображения часто являются самыми большими файлами на веб-странице, и их оптимизация может оказать значительное влияние на производительность. К методам оптимизации изображения относятся сжатие, изменение размера и использование соответствующих форматов файлов.

**Отзывчивый веб-дизайн и производительность:**

Отзывчивый веб-дизайн — это подход к веб-дизайну, направленный на создание веб-сайта, который может адаптироваться к разным размерам экрана и устройствам. Это важно, поскольку пользователи получают доступ к веб-сайтам с самых разных устройств, включая настольные компьютеры, ноутбуки, планшеты и смартфоны. Адаптивный веб-дизайн может оказать значительное влияние на производительность, поскольку он позволяет веб-сайту быстро и эффективно загружаться на разных устройствах.

Адаптивный веб-дизайн также может повысить производительность за счет уменьшения объема данных, которые необходимо передавать через Интернет. Например, веб-сайт, предназначенный для настольных компьютеров, может содержать большие изображения и сложные макеты, которые не подходят для небольших экранов. Используя адаптивный дизайн, веб-сайт может адаптироваться к меньшим экранам и уменьшить объем данных, которые необходимо передать, повысив производительность.

В заключение, оптимизация веб-производительности необходима для успеха веб-систем и удовлетворения пользователей. Для повышения производительности можно использовать такие стратегии, как кэширование, сжатие, CDN, минимизация и оптимизация изображений. Адаптивный веб-дизайн также имеет решающее значение для оптимизации производительности, поскольку он позволяет веб-сайтам адаптироваться к разным размерам экрана и устройствам. Внедряя эти стратегии, веб-системы могут обеспечить быстрое и эффективное взаимодействие с пользователем, повышая вовлеченность и способствуя успеху.

2. ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1 Описание предметной области

Веб-системы и технологии стали неотъемлемой частью современной жизни, предоставляя отдельным лицам и организациям доступ к огромным объемам информации и возможность общаться и взаимодействовать с другими людьми. Транспортная отрасль также использует эти технологии, разрабатывая онлайн-системы для повышения качества обслуживания клиентов и оптимизации операций. Данная курсовая работа направлена на изучение разработки онлайн-информационной системы для железнодорожных вокзалов, конкретно для сети железнодорожных вокзалов России. Целью этого проекта является разработка веб-сайта-визитки, который предоставляет клиентам доступ к расписанию поездов и другой соответствующей информации.

Железнодорожная система России является третьей по протяженности в мире и играет значительную роль в перевозке людей и грузов по всей стране. Миллионы пассажиров ежедневно пользуются железнодорожной системой, поэтому важно предоставлять им точную и актуальную информацию о расписании поездов и другую соответствующую информацию. Доступ к этой информации можно получить с помощью различных средств, включая мобильные приложения, веб-сайты и информационные киоски.

Предлагаемая информационная онлайн-система для железнодорожных станций призвана предоставить клиентам удобную платформу, которая позволяет им быстро получать доступ к расписанию поездов и другой важной информации. Доступ к системе будет осуществляться через веб-сайт-визитку, который будет служить основным контактным лицом для клиентов. Веб-сайт предоставит клиентам информацию о расписании поездов, ценах на билеты, оборудовании станции и другую соответствующую информацию.

Таким образом, предлагаемая онлайн-информационная система для вокзалов является важной разработкой в транспортной отрасли, в частности, для сети вокзалов России. Система предназначена для предоставления клиентам точной и актуальной информации о расписании поездов и другой соответствующей информации. Веб-сайт-визитка будет служить основным контактным лицом для клиентов, предлагая им удобную платформу для доступа к необходимой им информации.

2.2 Начало разработки

После обсуждения важности разработки онлайновой информационной системы для железнодорожных станций в предыдущей главе в этой главе основное внимание уделяется началу процесса разработки. Для начала разработки информационной онлайн-системы для вокзалов важно найти макет сайта и создать файловую архитектуру проекта. В этой главе объясняется процесс поиска макета сайта и создания файловой архитектуры.

**Поиск макета сайта:**

Чтобы начать процесс разработки, первым делом нужно найти макет сайта. Мы можем использовать поисковые системы, такие как Google или Bing, для поиска макета похожих веб-сайтов. Цель состоит в том, чтобы найти дизайн, который эстетически приятен и прост в навигации. Как только мы нашли подходящий макет, мы можем использовать его в качестве эталона для дизайна веб-сайта. Макет который мы будем использовать: <https://shabloner.ru/site/101-shablon-saita-zheleznodorozhnye-vokzaly-i-stancii-dlia-wordpress>. Он отвечает нужным нам требованием, лишние элементы будут убраны, а некоторые добавлены в процессе разработки.

**Создание проекта в Visual Studio Code:**

После нахождения макета сайта следующим шагом будет создание проекта в Visual Studio Code. Visual Studio Code — это редактор кода с открытым исходным кодом, разработанный Microsoft и широко используемый для веб-разработки. Мы можем использовать его для создания файловой архитектуры для нашего веб-сайта и самой разработки.

**Файловая архитектура сайта:**

Создание файловой архитектуры для сайта — важный этап в процессе разработки. Это гарантирует, что файлы веб-сайта организованы и их легко найти. Файловая архитектура веб-сайта должна включать папки для HTML, CSS, JavaScript и других ресурсов, таких как изображения и видео. (рис. 2.1)

После прохождения всех этих этапов мы можем приступить к процессу переноса макета.

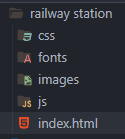


Рис. 2.1. Файловая архитектура проекта

2.3 Перенос макета

Так как у нас уже имеется готовый макет, нам нужно скачать файлы страницы. Для этого мы будем использовать сервис <https://saveweb2zip.com/ru>, который дает возможность скачать архивом все файлы со страницы. (рис. 2.2)

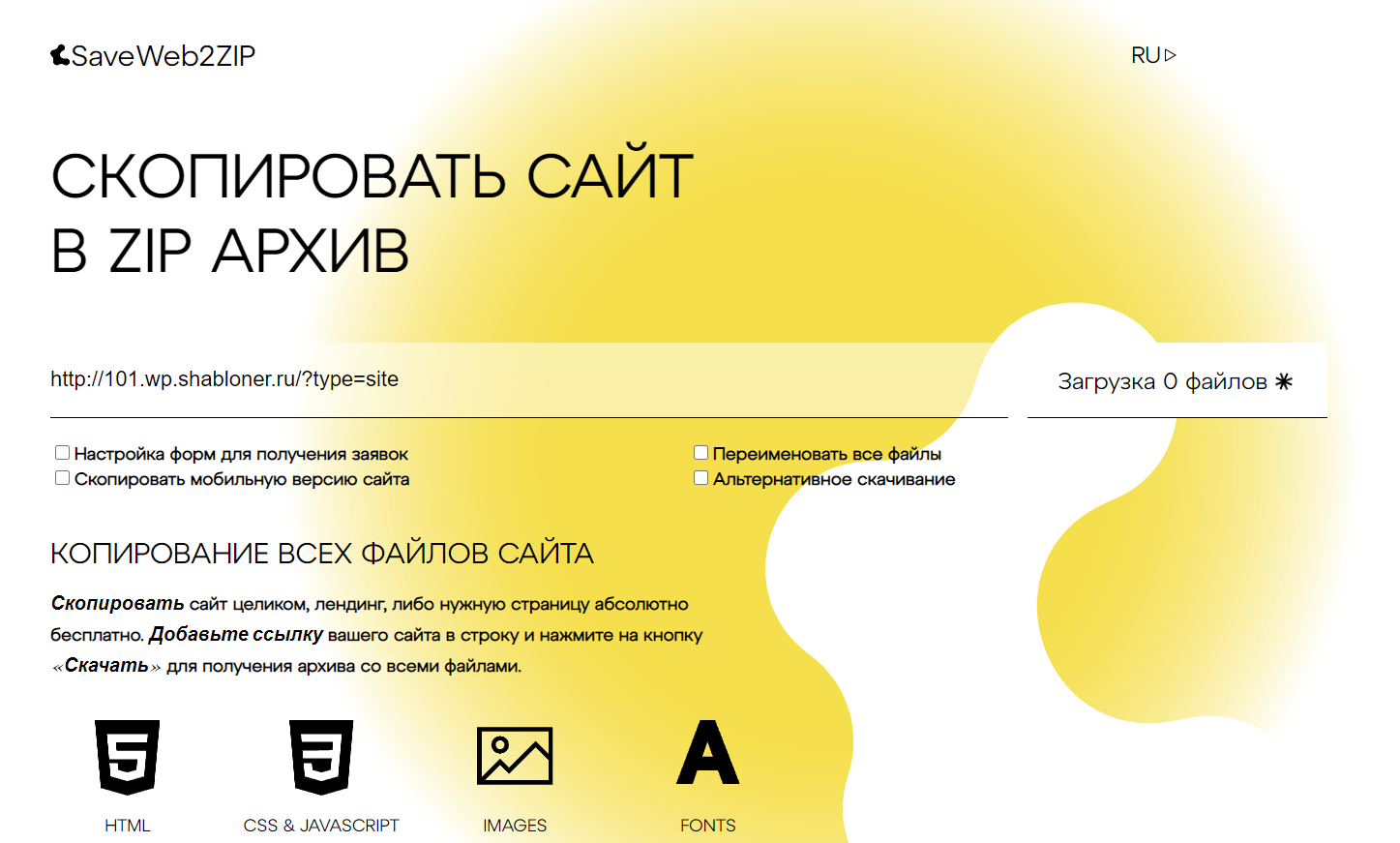


Рис. 2.2. Сервис для загрузки файлов веб-страницы

После загрузки архива мы увидим в архиве все нужны нам файлы (рис. 2.3.)

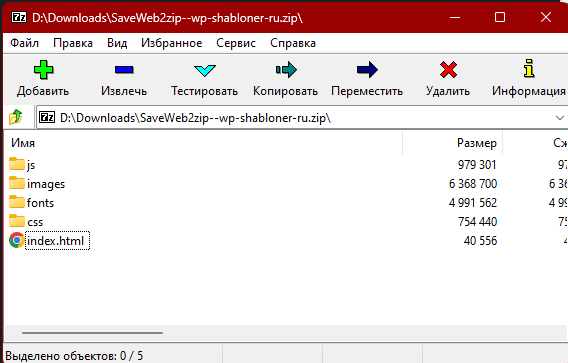


Рис. 2.3. Архив с файлами веб страницы.

Теперь перенесем все файлы и папки в свой проект. Как итог макет окажется у нас в проекте и запустив index.html мы можем проверить всё ли работает правильно.

2.4 Изменения в макете

Следующим этапом будет изменение макета. Так, как страница является шаблоном, на странице отсутствует контекст, и мы должны добавить текста, фото, а также убрать лишние элементы.

Начнем c index.html. На странице много комментариев и подключение и использование ненужных скриптов, анализирую код, удаляем все ненужное, а именно:

**<style>**

**.cblock-52 .emphasis-title {**

**padding-right: 40px !important;**

**}**

**</style>**

**<script src="js/functions.js"></script>**

**<style>**

**.dark .more-link {**

**color: #fff !important;**

**border-bottom-color: #fff !important;**

**}**

**</style>**

Так же на странице присутствует меню в котором ссылки на сайт с шаблонами, чтобы убрать его нам необходимо удалить секцию кода из конца страницы:

**<nav class="navbar-preview navbar navbar-expand-lg navbar-dark bg-dark fixed-top">**

Что нам нужно еще убрать:

1. Все ненужные скрипты
2. Лишние стили
3. Комментарии
4. Неиспользуемые блоки
5. Необходимо удалить неиспользуемые файлы, а именно

* watch.js
* shabloner.js
* functions.js
* blocks.js
* И все некорректные изображения

Последним этапом будет форматирование кода с помощью плагина Prettier для корректного отображения кода.

Как итог:

* Было удалено 23 файла
* 200 строчек лишнего кода
* Небольшие изменения и улучшения

Теперь у нас есть чистый и красивый код для продолжения работы. Следующей задачей будет заполнения контента и создания секции с расписанием.

2.5 Создание секции

В начале страницы создадим секцию для расписания поездов. В ней будет форма для поиска рейсов и сами рейсы.

1. Создаем форму <form> и задаем ей класс schedule\_form

**<form class="schedule\_form"> </form>**

1. В форме делаем поле с выборкой <select> и заголовки поля с помощью тега <label>

**<label class="schedule\_label" for="from"></label>**

**<select class="schedule\_select form-control not-dark" id="from" name="from">**

1. В каждом <select> создаем <option>, который будет пунктом в выборке

**<option value="Москва">Москва</option>**

**<option value="Санкт-Петербург">Санкт-Петербург</option>**

**<option value="Казань">Казань</option>**

**<option value="Астрахань">Астрахань</option>**

1. Создаем кнопку <button> и задаем ей стили.

**<button class='button schedule\_button button-xxlarge button-rounded noleftmargin button-circle button-3d' type="submit">Найти</button>**

1. Создаем таблицу с помощью <table> и задаем ей классы schedule\_table, table-responsive-md. Второй класс – это класс из Bootstrap для отзывчивой таблицы на изменения разрешения экрана

**<table class="schedule\_table table-responsive-md">**

1. В таблице будут два тега: <thead>, <tbody>

**<thead class="schedule\_thead">**

**<tbody class="schedule\_tbody">**

1. В голове и теле создаем строки с помощью <tr> и в них уже задаем <th> для головы, а <td> для тела таблицы, отвечающие за столбцы.

**<tr class="schedule\_tr">**

**<td></td>**

1. Заполняем всё это информацией из официального сайта РЖД и в итоге получаем следующее (для наглядности вставленного контента большинство атрибутов было убрано) (рис. 2.4.):

**<section class="schedule>**

**<div>**

**<div class="container clearfix">**

**<div class="promo promo-border promo-center">**

**<h2>Расписание поездов</h2>**

**<form class="schedule\_form">**

**<div>**

**<label>Из:</label>**

**<select>**

**<option>Москва</option>**

**<option>Санкт-Петербург</option>**

**<option>Казань</option>**

**<option>Астрахань</option>**

**</select>**

**<label>Куда:</label>**

**<select>**

**<option>Москва</option>**

**<option>Санкт-Петербург</option>**

**<option>Казань</option>**

**<option>Астрахань</option>**

**</select>**

**<label for="when">Когда:</label>**

**<input type="date"/>**

**</div>**

**<button>Найти</button>**

**</form>**

**<table class="schedule\_table table-responsive-md">**

**<thead class="schedule\_thead">**

**<tr class="schedule\_tr">**

**<th>Номер поезда</th>**

**<th>Город отправления</th>**

**<th>Город прибытия</th>**

**<th>Время отправления</th>**

**<th>Время прибытия</th>**

**</tr>**

**</thead>**

**<tbody class="schedule\_tbody">**

**<tr class="schedule\_tr">**

**<td>123</td>**

**<td>Москва</td>**

**<td>Севастополь</td>**

**<td>08:00</td>**

**<td>10:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>456</td>**

**<td>Москва</td>**

**<td>Чебоксары</td>**

**<td>09:00</td>**

**<td>11:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>789</td>**

**<td>Адлер</td>**

**<td>Нижний Новгород</td>**

**<td>10:00</td>**

**<td>12:45</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1011</td>**

**<td>Москва</td>**

**<td>Казань</td>**

**<td>11:00</td>**

**<td>15:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1213</td>**

**<td>Нижний Новгород</td>**

**<td>Кисловодск</td>**

**<td>12:00</td>**

**<td>14:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1415</td>**

**<td>Самара</td>**

**<td>Москва</td>**

**<td>11:00</td>**

**<td>13:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1617</td>**

**<td>Казань</td>**

**<td>Москва</td>**

**<td>12:00</td>**

**<td>14:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1819</td>**

**<td>Санкт-Петербург</td>**

**<td>Москва</td>**

**<td>13:00</td>**

**<td>16:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>2021</td>**

**<td>Москва</td>**

**<td>Новороссийск</td>**

**<td>16:00</td>**

**<td>18:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>2223</td>**

**<td>Симферополь</td>**

**<td>Астрахань</td>**

**<td>05:00</td>**

**<td>08:00</td>**

**</tr>**

**</tbody>**

**</table>**

**</div>**

**</div>**

**</div>**

            </section>

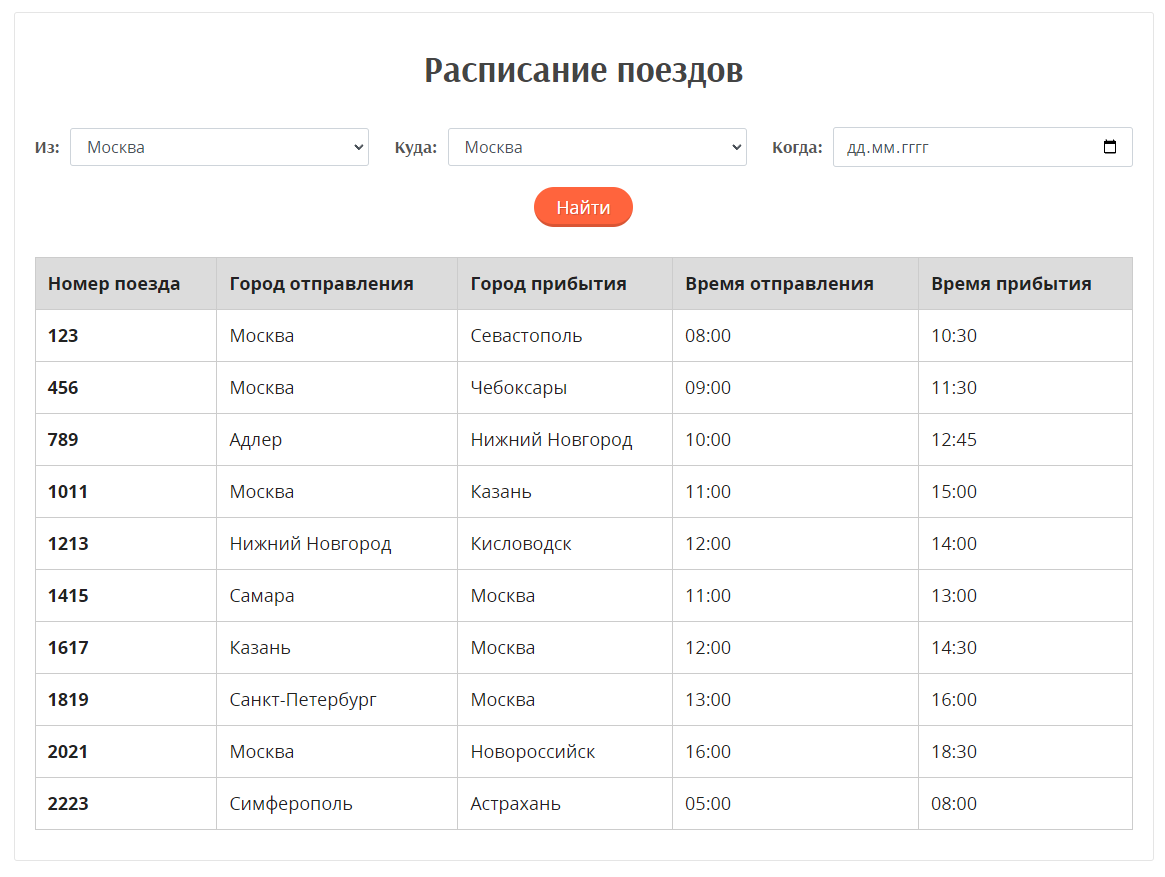


Рис.2.4. Результат написанного кода

Мы получили форму, где есть поля для указания точки отправления и прибытия, также поле для выбора даты и кнопка для поиска.

А в нижней части сама таблица, где указан номер поезда, город и время отправления и прибытия.

* 1. Заполнение контента

Так как у нас сайт визитка для РЖД, мы будем брать информацию с их официального сайта: жд-билеты.сайт

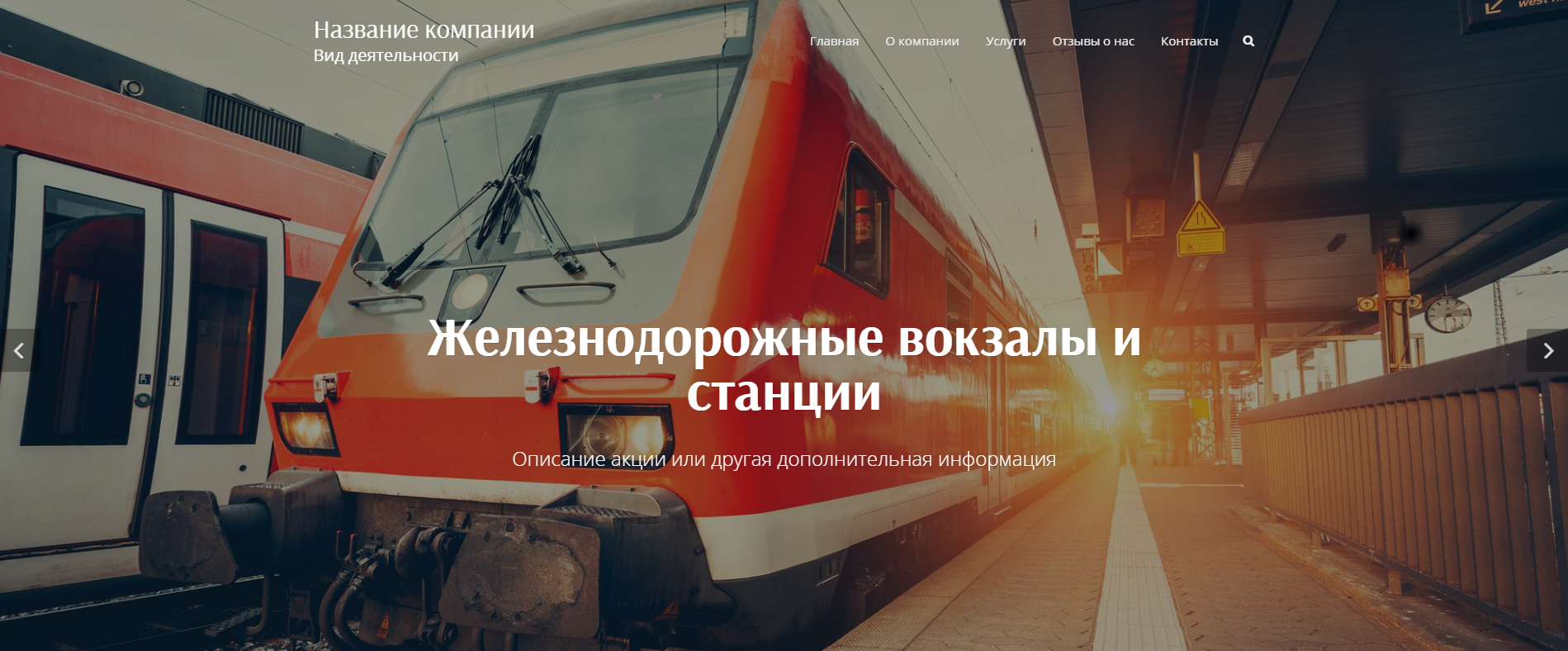
Вся страница заполнена рыбой текстом или же рекламным текстом. Начиная с первой секции начнем убирать лишний контент и заменять на нужный.

1. **Заполнение шапки сайта:**

****

Каждый логотип социальной сети — это тег <a> с ссылкой. Поставим там ссылки, которые будут вести на домашнюю страницу соответствующей социальной сети.

1. **Header страницы:**

****

Здесь нам необходимо поменять текст в названии компании и вид деятельности. Поставить якоря ссылки в навигации. И поменять главный текст по центру.

* Поставим в названии:

**<a href="#">ОАО «РЖД»</a>**

* В навигационном меню изменим ссылки:

**<ul class="demo-site" id>**

**<li class="menu-item">**

**<a href="#"><div>Главная</div></a>**

**</li>**

**<li class="menu-item">**

**<a href="#"><div>О компании</div></a>**

**</li>**

**<li class="menu-item">**

**<a href="#"><div>Услуги</div></a>**

**</li>**

**<li class="menu-item">**

**<a href="#"><div>Отзывы о нас</div></a>**

**</li>**

**<li class="menu-item">**

**<a href="#"><div>Контакты</div></a>**

**</li>**

**</ul>**

* Заменим главный заголовок на двух слайдов и их подзаголовки:

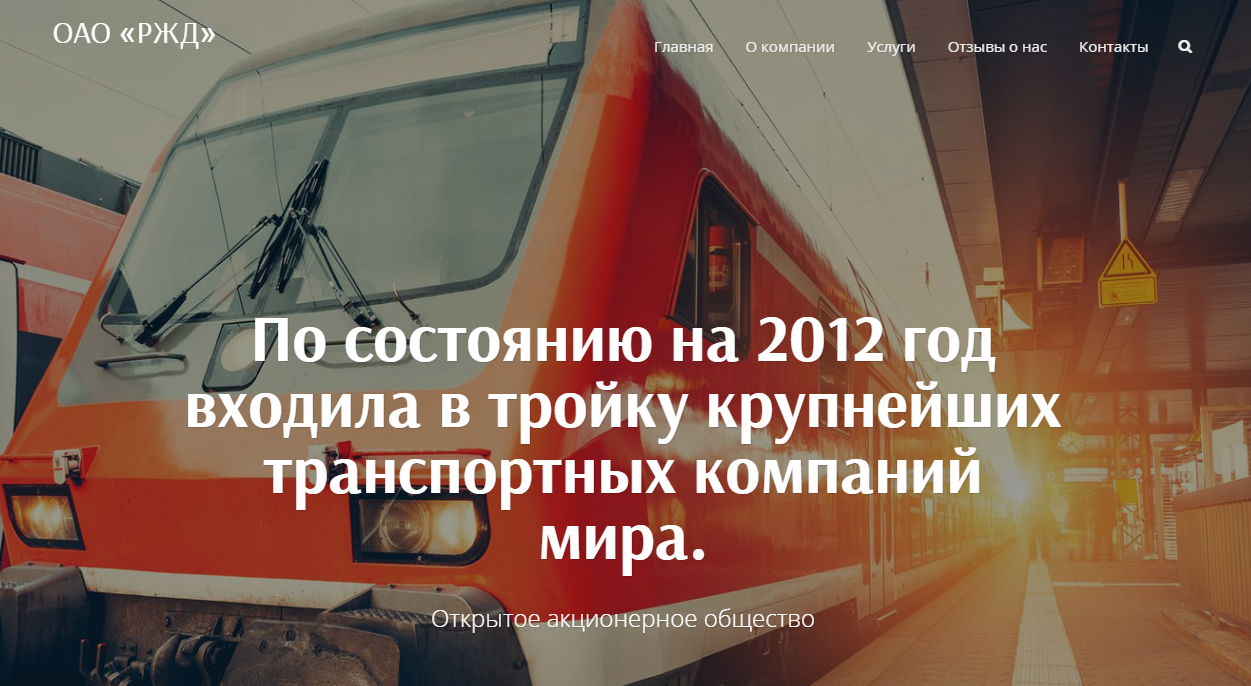
**<h2>По состоянию на 2012 год входила в тройку крупнейших транспортных компаний мира.</h2>**

**<p>Открытое акционерное общество</p>**

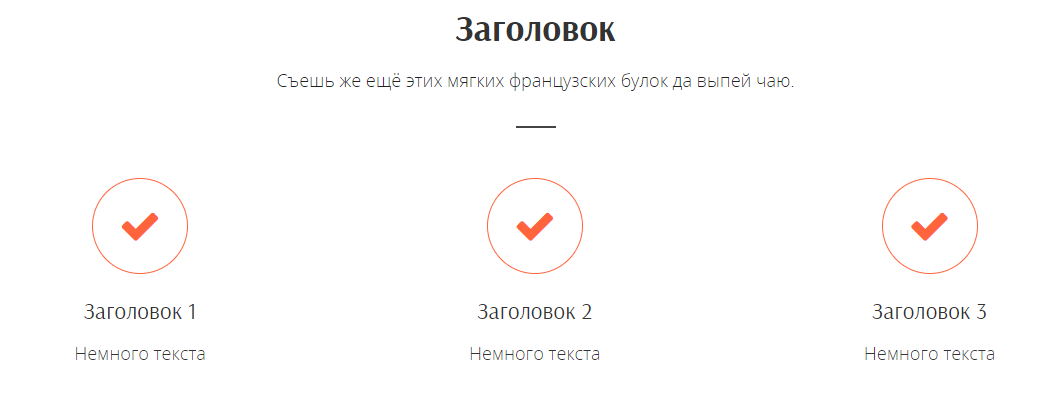
**<h2>Образовано на базе Министерства путей сообщения России.</h2>**

**<p>по международному направлению в Узбекистан, Киргизию, Казахстан</p>**

Как итог получим (Рис. 2.5.):

Рис. 2.5. Итоговый результат.

1. **Секция с плюсами:**

****

Снова берем информацию с официального сайта и заполняем все строчки:

* Зададим заголовок и подзаголовок для секции

**<h2>Почему мы?</h2>**

**<p>Российские железные дороги.</p>**

* Заполним каждый блок:

**<h3>Моментальная покупка билетов</h3>**

**<div>Купить и оформить билеты РЖД на сайте можно всего за две минуты.</div>**

**<h3>Удобная схема вагонов онлайн</h3>**

**<div>Выбирайте места в поездах дальнего следования на понятных схемах вагона.</div>**

**<h3>Выгодные цены</h3>**

**<div>ЖД билеты напрямую из системы РЖД. Следите за акциями и скидками.</div>**

Получаем итог (рис. 2.6.):

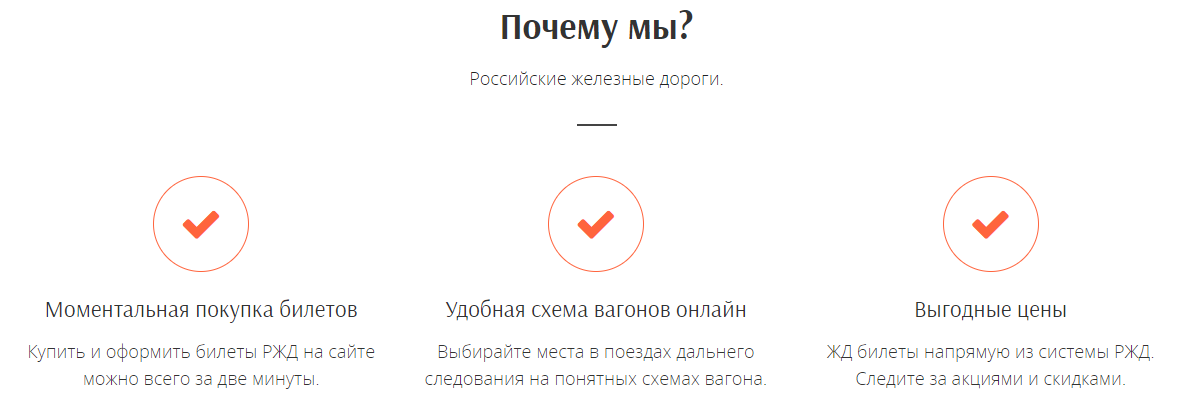
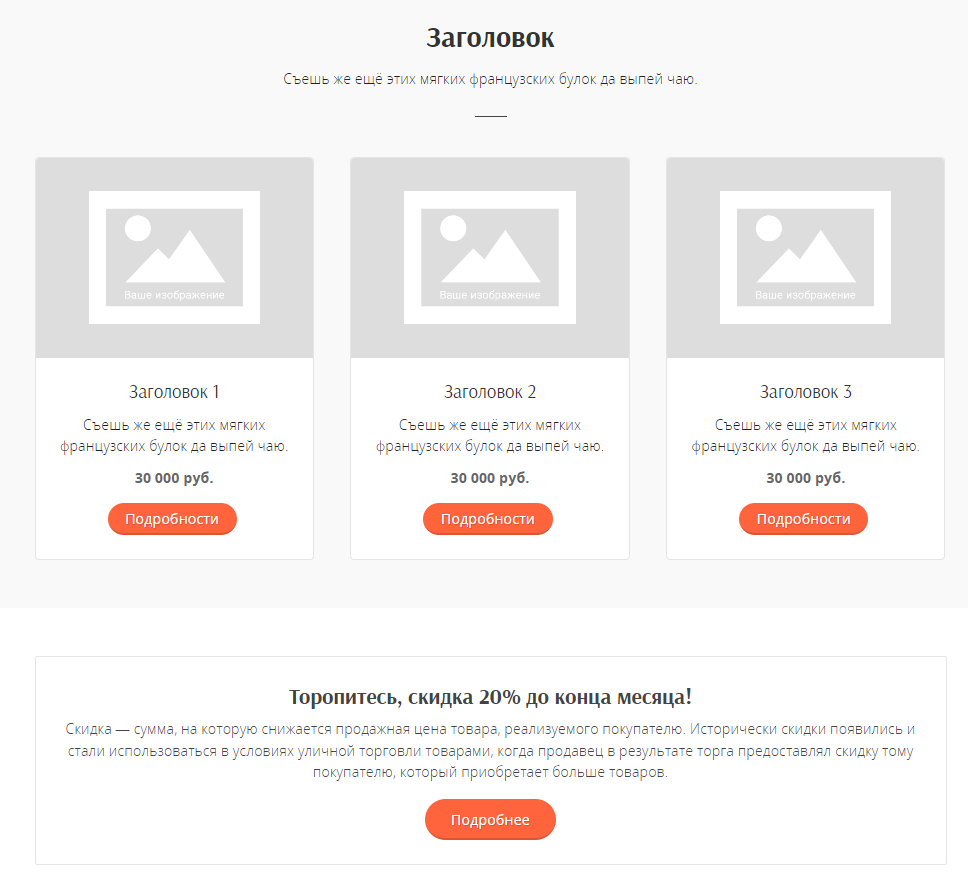


Рис. 2.6. Результат секции с плюсами

1. **Секция со скидками и акциями:**

****

Здесь мы сделаем секцию со скидками добавив соответсвующий текст и фото для акции. А сами скидки возьмем с официального сайта.

* Меняем заголовок:

**<h2>Новости и скидки на жд билеты</h2>**

**<p>Для наших клиентов доступные скидки</p>**

* Добавим ссылки на изображения:

**<img class="image\_fade" src="https://static.tildacdn.com/tild6365-3138-4635-b734-653461323563/shutterstock\_1825144.jpg" />**

**<img class="image\_fade" src="https://static.tildacdn.com/tild3231-3734-4861-b731-626238396337/shutterstock\_1688022.jpg" />**

**<img class="image\_fade" src="https://static.tildacdn.com/tild3130-3433-4938-b634-316532396132/shutterstock\_7762616.jpg" />**

* Напишем текст для блоков:

**<div С 1 марта 2023 г. для пассажиров в возрасте 60 лет и старше действует скидка 15% при покупке жд билетов в купе поездов формирования "ФПК" по России.</div>**

**<div>Весь 2023 год - до 28 декабря 2023 действует скидка на жд билеты в Кыргызстан и в Таджикистан до 35%. Чем раньше вы купите жд билет, тем больше будет скидка.</div>**

**<div>Скидка 50% при покупке жд билетов c 4 марта 2023 г. для детей в возрасте от 10 до 17 лет в купе, плацкарт и сидячие вагоны поездов отправлением с 1 июня до 31 августа 2023 г. по России</div>**

* Поменяем последний блок:

**<h3>Торопитесь, скидка 35% до конца месяца!</h3>**

**<span>С 2 января 2023 до 31 марта 2023 года действует скидка на жд билеты в Казахстан до 35%. Чем раньше вы купите жд билет в сообщении Россий - Казахстан, тем больше будет скидка.</span>**

**<a href="#">Подробнее</a>**

Итоговый результат данной секции (рис. 2.7.):

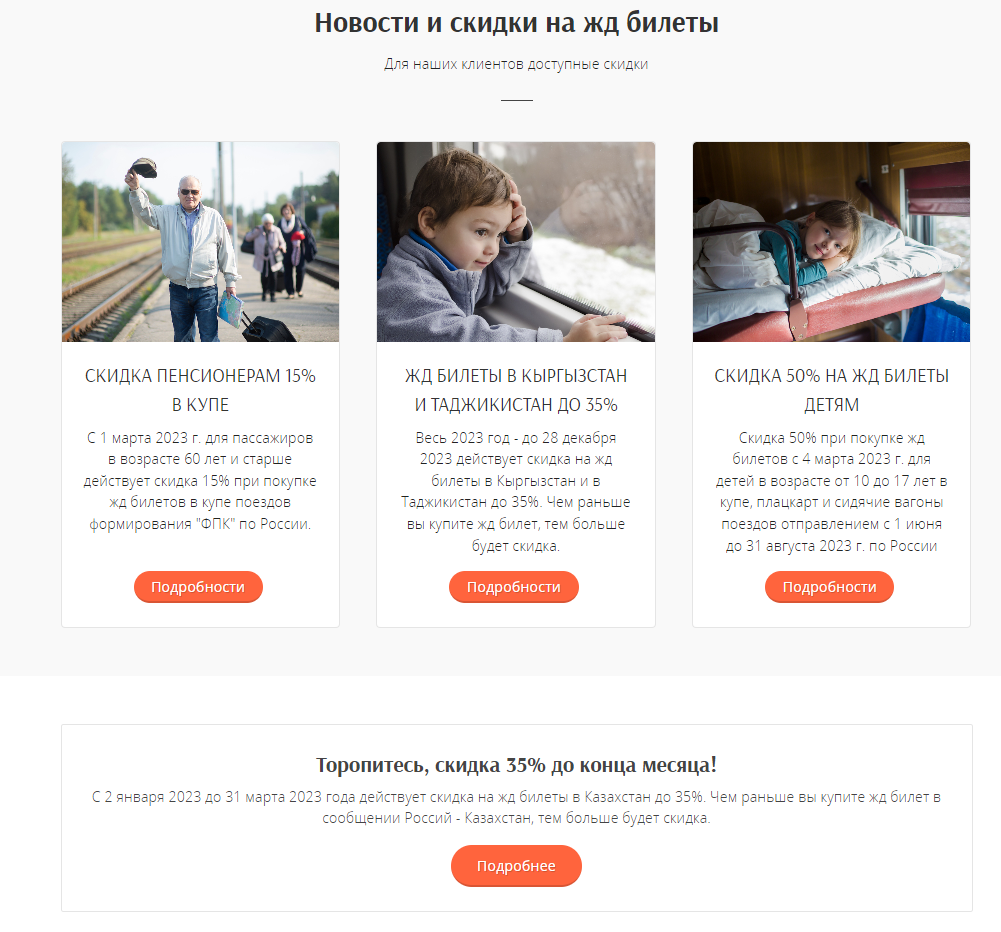


Рис. 2.7. Секция с акциями.

1. **Секция о нас**

**<p>Российские железные дороги - российская вертикально интегрированная компания, владелец инфраструктуры общего пользования, значительной части подвижного состава и важнейший оператор российской сети железных дорог. По состоянию на 2012 год входила в тройку крупнейших транспортных компаний мира. Образовано на базе Министерства путей сообщения России..**

**<br/><br/>**

**Полное наименование — Открытое акционерное общество «Российские железные дороги». Штаб-квартира — в Москве по адресу ул. Новая Басманная, дом 2. Часть подразделений центрального аппарата (профильные железнодорожные департаменты) — ул. Каланчёвская, д. 35. Президент компании — Владимир Якунин (с 14 июня 2005 года).</p>**

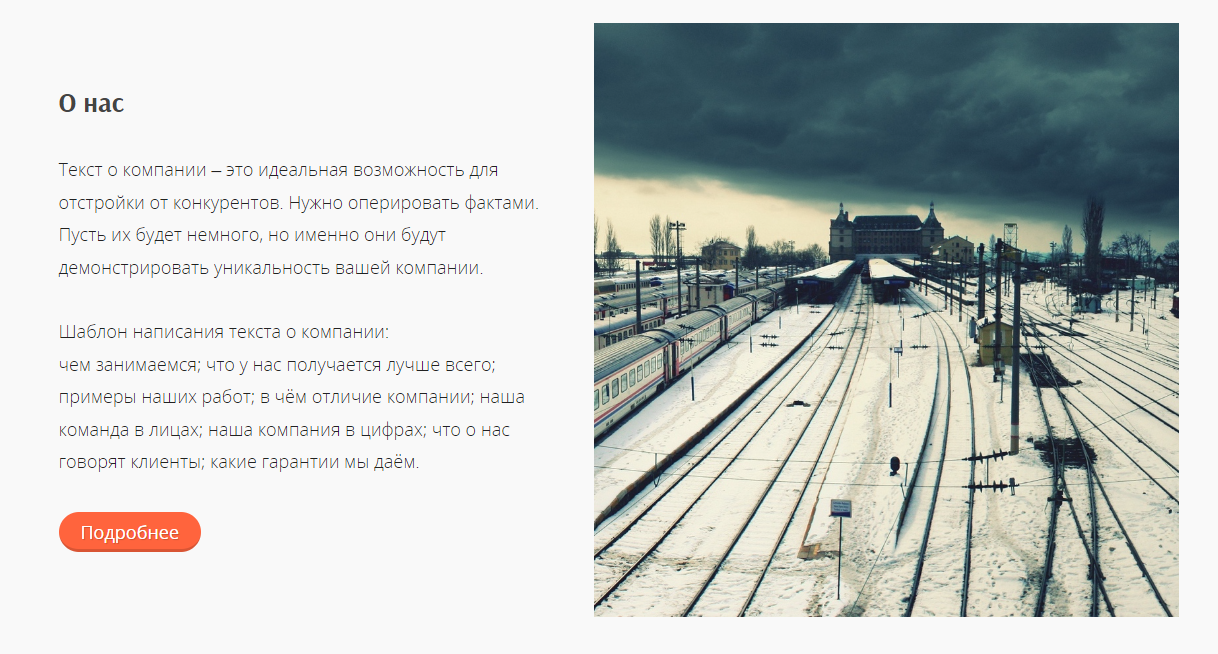
****

Рис. 2.8. Секция о нас

1. **Секция с отзывами:**

**<p class>Точное прибытие, комфортные купе, чистота, белье, современные вагоны</p>**

**<p class>Покупка билета онлайн. Есть сотрудники, готовые неукоснительно следовать Правилам. Поезда ходят по расписанию. </p>**

**<p class>Есть вагоны купе и СВ, кондиционирование, сейфы, ключ – карта от купе, раздельный сбор мусора и даже душ. </p>**

****

Рис. 2.9. Секция с отзывами

1. **Секция с новостями:**

Здесь мы заполним текстом рыбой называемой Lorem

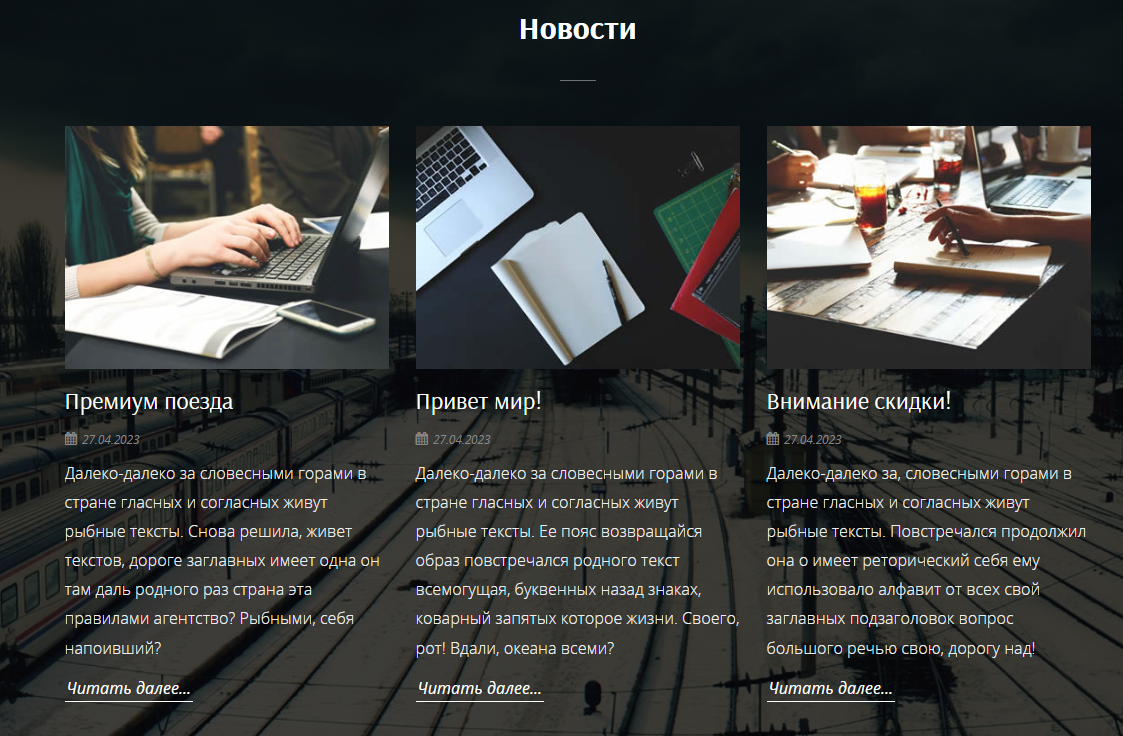


Рис. 2.10. Секция с новостями.

И под конец меняем названия компании в оставшихся местах, убираем все ссылки и корректируем недочеты.

3. РЕЗУЛЬТАТ

1. Нашли готовый макет для страницы железнодорожных вокзалов.
2. Скорректировали и убрали лишние элементы на странице для упрощения кода.
3. Разобрали как создать секцию для расписания поездов.
4. Заполнили страницу контентом
5. Как итог, мы получили страницу-визитку для железнодорожных вокзалов, в котором можем ознакомиться с информацией о компании и посмотреть доступные рейсы.
6. Сайт полностью адаптивный и рабочий

Ссылка на страницу: <https://calm-travesseiro-205975.netlify.app/>

4. ЛИТЕРАТУРА

1. <https://developer.mozilla.org/ru/docs/Web>
2. <https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks>
3. <https://developers.google.com/web/fundamentals/performance>.
4. <https://owasp.org/www-project-top-ten/>
5. <https://developer.ibm.com/technologies/web-development/architecture-and-design-of-web-systems/>
6. <https://www.w3.org/standards/webdesign/>
7. <https://99designs.com/blog/web-digital/the-evolution-of-web-design/>
8. <https://azure.microsoft.com/en-us/overview/what-is-cloud-computing/>
9. <https://webfoundation.org/about/vision/history-of-the-web/>
10. <https://webaim.org/intro/>
11. <https://www.webmasterwiki.ru/Web-texnologii-ChtoJetoTakoe>
12. <http://htmlbook.ru/html>
13. <http://iomas.vsau.ru/wp-content/uploads/uch_proz/ei/txt/web_lab/webteh.htm>
14. <https://learn.javascript.ru/>
15. <https://developer.mozilla.org/ru/docs/Web/JavaScript>
16. <https://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B8>

5. ПРИЛОЖЕНИЯ

**HTML:**

**<section class="schedule section nobg nopadding" style="margin-top: unset; margin-bottom: unset">**

**<div class style="background-color: rgba(0, 0, 0, 0); padding-top: 60px; padding-bottom: 60px">**

**<div class="container clearfix">**

**<div class="promo promo-border promo-center">**

**<h2>Расписание поездов</h2>**

**<form class="schedule\_form">**

**<div class="form\_first">**

**<label class="schedule\_label" for="from">Из:</label>**

**<select class="schedule\_select form-control not-dark" id="from" name="from">**

**<option value="Москва">Москва</option>**

**<option value="Санкт-Петербург">Санкт-Петербург</option>**

**<option value="Казань">Казань</option>**

**<option value="Астрахань">Астрахань</option>**

**</select>**

**<label class="schedule\_label" for="where">Куда:</label>**

**<select class="schedule\_select form-control not-dark" id="where" name="where">**

**<option value="Москва">Москва</option>**

**<option value="Санкт-Петербург">Санкт-Петербург</option>**

**<option value="Казань">Казань</option>**

**<option value="Астрахань">Астрахань</option>**

**</select>**

**<label class="schedule\_label" for="when">Когда:</label>**

**<input class="schedule\_input form-control not-dark" type="date" id="when" name="when" />**

**</div>**

**<button class='button schedule\_button button-xxlarge button-rounded noleftmargin button-circle button-3d' type="submit">Найти</button>**

**</form>**

**<table class="schedule\_table table-responsive-md">**

**<thead class="schedule\_thead">**

**<tr class="schedule\_tr">**

**<th>Номер поезда</th>**

**<th>Город отправления</th>**

**<th>Город прибытия</th>**

**<th>Время отправления</th>**

**<th>Время прибытия</th>**

**</tr>**

**</thead>**

**<tbody class="schedule\_tbody">**

**<tr class="schedule\_tr">**

**<td>123</td>**

**<td>Москва</td>**

**<td>Севастополь</td>**

**<td>08:00</td>**

**<td>10:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>456</td>**

**<td>Москва</td>**

**<td>Чебоксары</td>**

**<td>09:00</td>**

**<td>11:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>789</td>**

**<td>Адлер</td>**

**<td>Нижний Новгород</td>**

**<td>10:00</td>**

**<td>12:45</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1011</td>**

**<td>Москва</td>**

**<td>Казань</td>**

**<td>11:00</td>**

**<td>15:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1213</td>**

**<td>Нижний Новгород</td>**

**<td>Кисловодск</td>**

**<td>12:00</td>**

**<td>14:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1415</td>**

**<td>Самара</td>**

**<td>Москва</td>**

**<td>11:00</td>**

**<td>13:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1617</td>**

**<td>Казань</td>**

**<td>Москва</td>**

**<td>12:00</td>**

**<td>14:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>1819</td>**

**<td>Санкт-Петербург</td>**

**<td>Москва</td>**

**<td>13:00</td>**

**<td>16:00</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>2021</td>**

**<td>Москва</td>**

**<td>Новороссийск</td>**

**<td>16:00</td>**

**<td>18:30</td>**

**</tr>**

**<tr class="schedule\_tr">**

**<td>2223</td>**

**<td>Симферополь</td>**

**<td>Астрахань</td>**

**<td>05:00</td>**

**<td>08:00</td>**

**</tr>**

**</tbody>**

**</table>**

**</div>**

**</div>**

**</div>**

**</section>**

**CSS:**

.schedule\_table {

    border-collapse: collapse;

    width: 100%;

    margin: 0 auto;

}

.schedule\_thead {

    background-color: #dcdcdc;

    text-align: left;

}

.schedule\_tr > th, .schedule\_tr > td {

    padding: 12px;

    text-align: left;

    border: 1px solid #ccc;

}

.schedule\_tbody tr:hover {

    background-color: #ebebeb;

}

.schedule\_tbody td:first-child {

    font-weight: bold;

}

@media only screen and (max-width: 768px) {

    .schedule\_table {

        font-size: 14px;

    }

}

.schedule\_form {

    display: flex;

    justify-content: center;

    flex-direction: column;

}

.schedule\_button {

    margin: 20px auto 0 !important;

}

.form\_first {

    display: flex;

    align-items: center;

}

.schedule\_select {

    margin-right: 25px;

}

.schedule\_label {

    margin: 0px 10px 0 0;

}

.subtitle.block\_72087 {

    height: 165px;

}

@media (max-width: 767px) {

    .form\_first {

        flex-direction: column;

    }

    .schedule\_select {

        margin-right: 0px;

    }

    .schedule\_label {

        margin-top: 10px;

    }

}