МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ   
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 6-05-0612-01 Программная инженерия

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА**

по дисциплине «Компьютерные языки разметки»

Тема: веб-сайт «Автовокзал»

**Исполнитель**

студент 1 курса 10 группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И. А. Рубцов

подпись, дата

**Руководитель**

Старший-преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.В.Барковский

должность, учен. степень, ученое звание подпись, дата

Допущен к защите \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

дата, подпись

Курсовой проект защищен с оценкой

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ Е.В. Барковский

подпись дата инициалы и фамилия

**Содержание**

|  |  |
| --- | --- |
| Введение | 5 |
| 1. Постановка задачи | 6 |
| 1.1. Обзор аналогичных решений | 6 |
| 1.2. Техническое задание | 8 |
| 1.3. Выбор средств реализации программного продукта | 9 |
| 1.4. Постановка задачи | 9 |
| 1.5. Вывод  2. Проектирование страниц веб-сайта | 9  10 |
| 2.1. Выбор стилевого оформления | 10 |
| 2.2. Выбор шрифтового оформления | 10 |
| 2.3. Разработка логотипа | 10 |
| 2.4. Разработка пользовательских элементов | 11 |
| 2.5. Разработка спецэффектов | 12 |
| 2.6. Выводы | 13 |
| 3. Реализация структуры веб-сайта | 14 |
| 3.1. Структура HTML-документа | 14 |
| 3.2. Добавление таблиц стилей SCSS и CSS | 16 |
| 3.3. Использование стандартов XML (SVG) | 17 |
| 3.4. Использование JS | 18 |
| 3.5. Выводы | 20 |
| 4. Тестирование веб-сайта | 21 |
| 4.1. Адаптивный дизайн веб-сайта | 21 |
| 4.2. Кроссбраузерность веб-сайта | 23 |
| 4.3. Руководство пользователя | 23 |
| 4.4. Выводы | 25 |
| Заключение | 26 |
| Список использованных литературных источников | 27 |
| Приложение А Прототипы веб-страниц | 28 |
| Приложение Б Макет структуры веб-сайта | 31 |
| Приложение В Листинг НТML-документа | 33 |
| Приложение Г Листинг SCSS и CSS | 41 |
| Приложение Д Листинг Адаптивность | 48 |
| Приложение Е Листинг XML-файлов | 58 |
| Приложение Ж Листинг JavaScript | 60 |
| Приложение З Листинг SVG | 62 |
|  |  |
|  |  |

# Введение

В современном мире цифровых технологий наличие веб-сайта становится

неотъемлемой частью успешного функционирования любой организации, включая автовокзалы. Сайт автовокзала играет ключевую роль в предоставлении актуальной информации пассажирам, упрощении процессов бронирования билетов и повышении уровня сервиса. В условиях высокой конкуренции на рынке транспортных услуг веб-сайт становится важным инструментом для привлечения клиентов, улучшения их опыта взаимодействия и укрепления доверия к услугам автовокзала.

Пассажиры все чаще предпочитают использовать онлайн-ресурсы для планирования поездок, поиска расписания, покупки билетов и получения информации о маршрутах. Веб-сайт автовокзала позволяет удовлетворить эти потребности, предоставляя удобный и доступный способ взаимодействия с клиентами. Он становится не только информационной платформой, но и инструментом для повышения лояльности пассажиров, предлагая такие функции, как онлайн-бронирование, уведомления об изменениях в расписании и возможность выбора удобных мест.

Создание веб-сайта автовокзала требует тщательной проработки структуры, дизайна и функциональности, чтобы обеспечить пользователям максимальный комфорт и простоту использования. Внедрение современных технологий, таких как мобильная адаптация, интеграция с платежными системами и системами управления расписанием, позволяет создать эффективный и конкурентоспособный ресурс, который будет отвечать потребностям как пассажиров, так и самого автовокзала.

Целью данного проекта является разработка веб-сайта «Автовокзал» с использованием современных технологий, таких как HTML5 и Scss/CSS3. В ходе работы над проектом необходимо выполнить ряд задач:

1) создание макета веб-сайта

2) создание прототипов веб-страниц

3) создание адаптивного дизайна

4) проверка работы веб-сайта на различных устройствах

5) тестирование веб-сайта

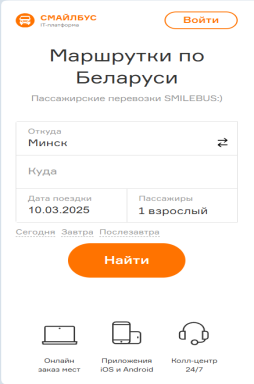
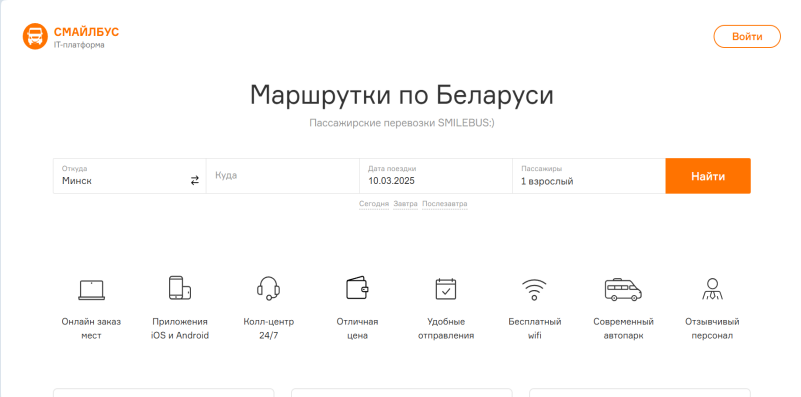
6) реализация структуры веб-сайта

**1. Постановка задачи**

**1.1 Обзор аналогичных решений.**

В современном мире веб-сайты стали неотъемлемой частью любой сферы, включая транспортные услуги. Для автовокзала наличие удобного и функционального сайта — это возможность предоставить пассажирам доступ к информации и услугам в любое время и из любой точки мира. Путешественникам важно иметь простой и понятный интерфейс для покупки билетов, просмотра расписания и получения актуальной информации о рейсах. Примером таких платформ могут служить сайты, подобные СМАЙЛБУС, INFOBUS, но адаптированные под нужды автовокзала.

Первым рассматривается проект «СМАЙЛБУС». Веб-сайт СМАЙЛБУС уникален в своём роде из-за ярких и запоминающихся цветов. Он выполнен в бело-оранжевых тонах, которые дают данному веб-сайту узнаваемость. Макет веб-сайта представлен на рисунке 1.1.



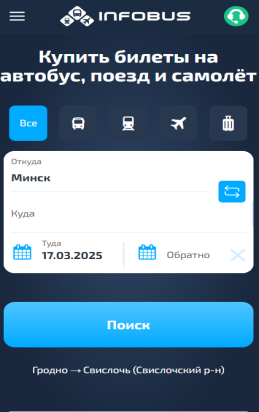
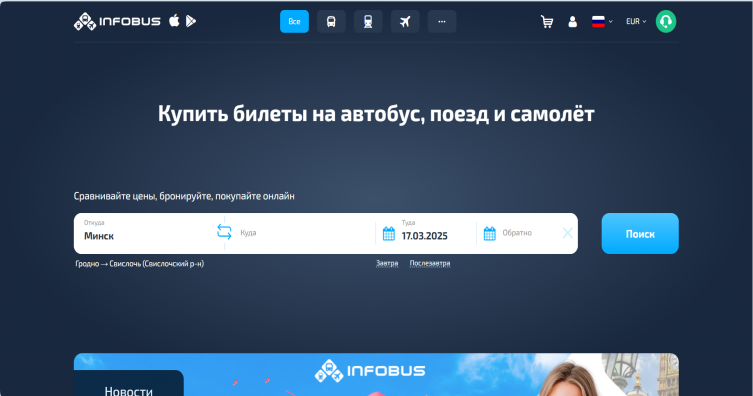
а б

а – desktop версия, б – mobile версия

Рисунок 1.1 – Окно главной страницы веб-сайта «СМАЙЛБУС»

Достоинствами веб-сайта автовокзала могут стать стильное оформление и продуманный дизайн, которые создают приятное впечатление у пользователей. Широкая представленность рейсов и направлений позволяет пассажирам легко выбрать подходящий маршрут. Удобная система поиска и фильтрации, включающая параметры даты, времени отправления, пункта назначения и типа автобуса, помогает пользователям быстро находить нужные рейсы.

Недостатками веб-сайта являются неудобность и неразличимость мобильной версии веб-сайта.



а б

а – desktop версия, б – mobile версия

Рисунок 1.2 – Окно главного каталога веб-сайта «INFOBUS»

Вторым рассматривается проект «INFOBUS». Веб-сайт INFOBUS отличается простотой и удобством в использовании. Дизайн сайта выполнен в сине-белых тонах, что создает приятную и спокойную атмосферу, способствуя легкости восприятия информации. Такое цветовое решение позволяет пользователям сосредоточиться на выборе рейсов и актуальной информации о поездках, не отвлекаясь на излишне яркие или сложные элементы дизайна.

Макет веб-сайта, представленный на рисунке 1.2, демонстрирует продуманную структуру, где основные элементы, такие как поиск рейсов, расписание и раздел с акциями, расположены интуитивно понятно.

На странице рейса веб-сайта "INFOBUS" вы найдете удобное и информативное представление о поездке. Главным элементом страницы является блок с основными данными о рейсе, включая время отправления, пункт назначения и стоимость билета. Эта информация расположена в центре страницы для удобства восприятия.

Справа от основных данных находится подробное описание рейса, которое включает в себя продолжительность поездки, тип автобуса, доступные услуги и дополнительные условия. Это описание помогает вам быстро оценить, насколько подходит данный рейс под ваши требования и предпочтения. Оно содержит достаточно информации, чтобы вы могли принять обоснованное решение, но при этом не перегружено лишними деталями, что делает процесс выбора максимально комфортным.

В целом, современные веб-сайты автовокзалов предлагают удобные и информативные решения для покупки билетов онлайн. Например, "СМАЙЛБУС" выделяется ярким и запоминающимся дизайном, что делает его узнаваемым среди пользователей. "INFABUS" привлекает своей интуитивно понятной навигацией и простотой использования.

**1.2. Технические задание**

Необходимо разработать веб-сайт автовокзала, который будет состоять из шести страниц, предоставляющих информацию о компании и её услугах, с упором на актуальную информацию о рейсах и продаже билетов.  
 Цель веб-сайта — предоставить пользователям удобный доступ к информации о рейсах, расписании, ценах на билеты и дополнительных услугах автовокзала. Веб-сайт будет представлять каждый рейс с его особенностями, временем отправления, пунктом назначения и стоимостью билета, чтобы пользователи могли быстро выбрать подходящий вариант и оформить заказ онлайн. Дизайн веб-сайта будет современным и привлекательным, чтобы обеспечить приятный пользовательский опыт и привлечь целевую аудиторию.

Ключевыми элементами всех страниц будет **header** с навигационными ссылками, а также **footer** с контактными данными, чтобы пользователи могли связаться с администрацией в случае возникновения вопросов или проблем.

**Первая страница «Главная страница»**  
 Это основная страница сайта, которая содержит общую информацию о автовокзале, актуальные новости, ссылки на популярные направления и кнопку для покупки билетов. Она служит отправной точкой для пользователей.

**На второй странице показан раздел «Купить билеты»**  
 Эта страница предназначена для поиска и покупки билетов. Здесь пользователи могут выбрать направление, дату поездки, количество пассажиров и оформить заказ.

**На третьей странице будет раздел «О нас»**  
 Страница содержит информацию о компании, её истории, миссии и ценностях. Здесь пользователи могут узнать больше о автовокзале, его услугах и преимуществах.

**Страница 4 — Партнеры**  
 На этой странице представлена информация о партнёрах автовокзала. Это транспортные компании, туристические агентства или другие организации, сотрудничающие с автовокзалом.

**Пятая страница «Сотрудники»**  
 Эта страница содержит информацию о команде автовокзала. Здесь представлены фотографии и краткие описания сотрудников, их ролей и обязанностей.

Адаптивность веб-сайта будет обеспечена для персональных компьютеров (ширина более 1000px), а также будет предусмотрена версия для мобильных устройств (ширина до 640px). Макет страниц веб-сайта будет разработан в онлайн-редакторе Figma с учётом всех требований и современных трендов в дизайне.

**1.3. Выбор средств реализации программного продукта**

Для реализации программного продукта были использованы следующие приложения: Visual Studio Code для написания кода веб-страницы, Figma для создания макета и прототипа веб-ресурса. Для внешнего описания веб-сайта использовались Scss/CSS3 для описания, XML для хранения информации об забранированных билетах. Для реализации функций использовался JavaScript.

**1.4. Вывод**

**Главная задача данного веб-сайта – это ознакомить потенциального клиента с услугами автовокзала, предоставив ему всю необходимую информацию в удобной и доступной форме. Для создания успешного веб-сайта для автовокзала важно разработать современный, эстетичный и интуитивно понятный дизайн, который не только привлечёт внимание широкой аудитории, но и вызовет доверие у пользователей. Качественный сайт должен сочетать в себе удобство навигации, актуальную информацию о расписании, услугах и контактах, а также визуальную привлекательность, чтобы выделяться среди конкурентов и обеспечивать комфортное взаимодействие с клиентами.**

1. **Проектирование страниц веб-сайта**
   1. **Выбор верстки**

Была выбрана flex-верстка, потому что она обеспечивает гибкость и простоту в управлении расположением элементов на странице. Flexbox позволяет легко выравнивать и распределять пространство между компонентами, адаптируя их под разные размеры экранов без лишних сложностей. Это особенно полезно для создания современных, отзывчивых интерфейсов, где важно сохранять порядок и читаемость контента на любых устройствах. Flex-верстка минимизирует использование "костылей" в CSS, таких как float или таблицы, делая код чище и поддерживаемым.

**2.2. Выбор стилевого оформления**

Необходимо иметь запоминающийся и приятный для глаза цветовой фон, чтобы пользователю было комфортно находиться на нашем сайте.

Стиль дизайна сайта – это самое первое на что пользователь обращает внимание. В зависимости от дизайна сайта, пользователь делает выбор в сторону вашей продукции или конкурентов.

Первый аспект – это композиционное решение. Меню навигации будет размещаться на верхней и нижней частях сайта для удобства пользователей. Прокрутка страницы будет вертикальной.

Второй аспект – это цветовая схема. Правильный выбор цветовой палитры для сайта автовокзала может значительно повлиять на восприятие и настроение пользователей. Цвета должны быть не только приятными для глаз, но и ассоциироваться с надежностью, комфортом и удобством, что особенно важно для транспортных услуг. Основные цвета сайта:

**#FF0033:**

Главная страница: Кнопка «Купить билеты», Заголовки разделов.

Купить билеты: Кнопки выбора билетов, кнопка «Оформить заказ»

О нас: Заголовки разделов, Ссылки на социальные сети, Кнопки «Связаться с нами».

Сотрудники: Заголовки разделов.

**#F5F5F5:**

Главная страница: Фон странцы, фон отдельных блоков.

Купить билеты: Фон формы заказа, Фон блоков с информацией о билетах.

О нас: Фон страницы, Фон блоков с текстовой информацией.

Партнеры: Фон страницы, Фон блоков с логотипами партнеров.

Сотрудники: Фон блоков с информацией о сотрудниках, Фон страницы.

# 2.3 Выбор шрифтового оформления

Выбор гарнитуры и размера влияет на восприятие дизайна также, как и «красивая картинка». На сайте будут использован шрифт GhoticA1 начертания Medium, пример текста, написанного данным шрифтом представлен на рисунке 2.3.



Рисунок 2.1 – Пример текста

GothicA1 Medium – современный шрифт без засечек с чистыми, лаконичными линиями, обеспечивающий отличную читабельность и сбалансированное визуальное восприятие текста.

Шрифт скачен с Google Fonts [4]. Шрифт будет подключен через @font-face.

**2.4. Разработка логотипа**

Логотип - это символ, выражающий сущность и уникальность магазина. Он является визуальным ключом, открывающим дверь в мир компании и создающим первое впечатление о ней.

Хорошо разработанный логотип имеет способность вызывать эмоции, ассоциации и связываться с целевой аудиторией. Он может быть простым и минималистичным или более сложным и детализированным, но всегда должен быть легко узнаваемым и отражать сущность. Дизайн логотипа представлен на рисунке 2.2.

****

Рисунок 2.2 – Логотип

Этот логотип представляет собой стилизованное изображение автобуса с двумя круглыми колесами и окнами, выполненное в минималистичном черно-белом стиле.

**2.5. Разработка пользовательских элементов**

В проекте будут представлены такие элементы пользовательского интерфейса как кнопки для перехода на веб форму, каталог и сама веб форма.

Кнопка для перехода на страницу с формой является стандартной. Кнопка представлена на рисунке 2.3.

{FC0426FE-C313-41B6-BA8D-D1D2AAFDC42E}

Рисунок 2.3 – Кнопка для перехода на страницу с покупками билетов

Для стилизации кнопки были использована инвертированная цветовая гамма, для создания визуального акцента на ней её фоновым цвет был выбран белый цвет.

Настольная версия каталога представлена на рисунке 2.4.

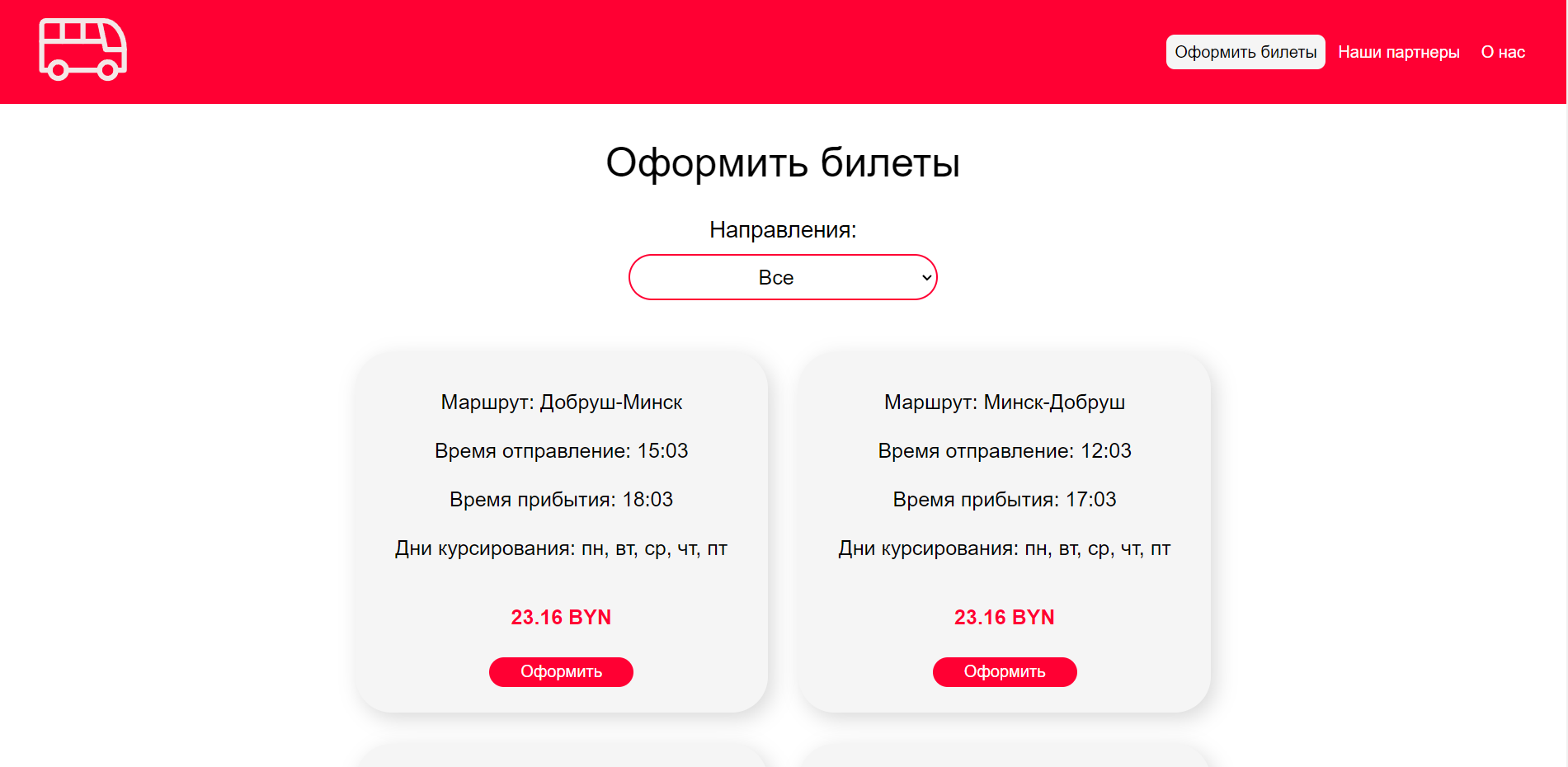


Рисунок 2.4 – Настольная версия каталога

Мобильная версия каталога представлена на рисунке 2.7.

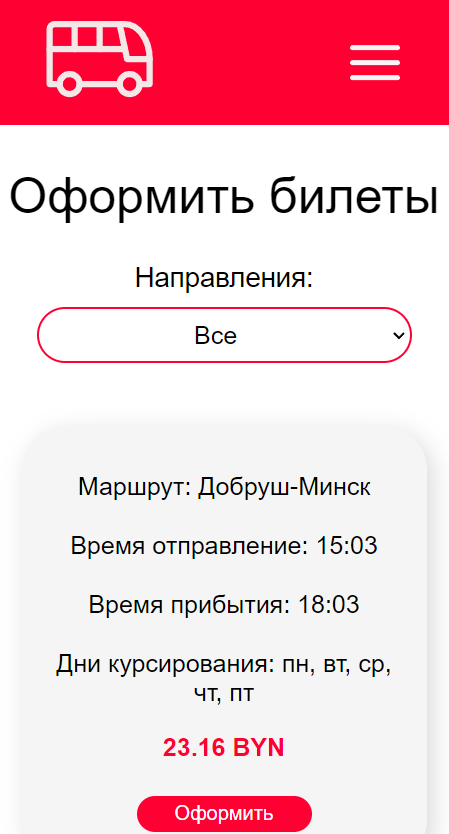


Рисунок 2.7 – Мобильная версия каталога билетов

Для удобства выбора билета на страницу каталога будет добавлен выпадающий список для выбора необходимого направления. Информация о билетах будет представлена в небольших блоках.

**2.6. Разработка спецэффектов**

На страницах веб-сайты будут разработаны динамические эффекты, которые сделают сайт более привлекательным для пользователя. Динамическое «бургер-меню» тоже реализовано в проекте.

Динамическое «бургер-меню» представлено на рисунке 2.8 и 2.9.



Рисунок 2.8 – бургер-меню в свёрнутом состоянии

Кнопка «бургер-меню» использована стандартная [5], а при нажатии кнопка будет поворачиваться на 90 градусов, для обозначения своего состояния. «бургер-меню» в развёрнутом состоянии представлено на рисунке 2.9.

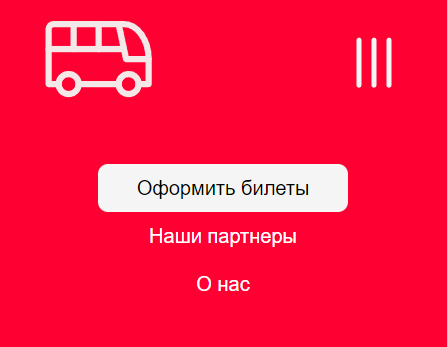


Рисунок 2.9 – бургер-меню в развёрнутом состоянии

Также будет осуществлён preloader-эффект при загрузке страниц. Пример предоставлен на рисунке 2.11.

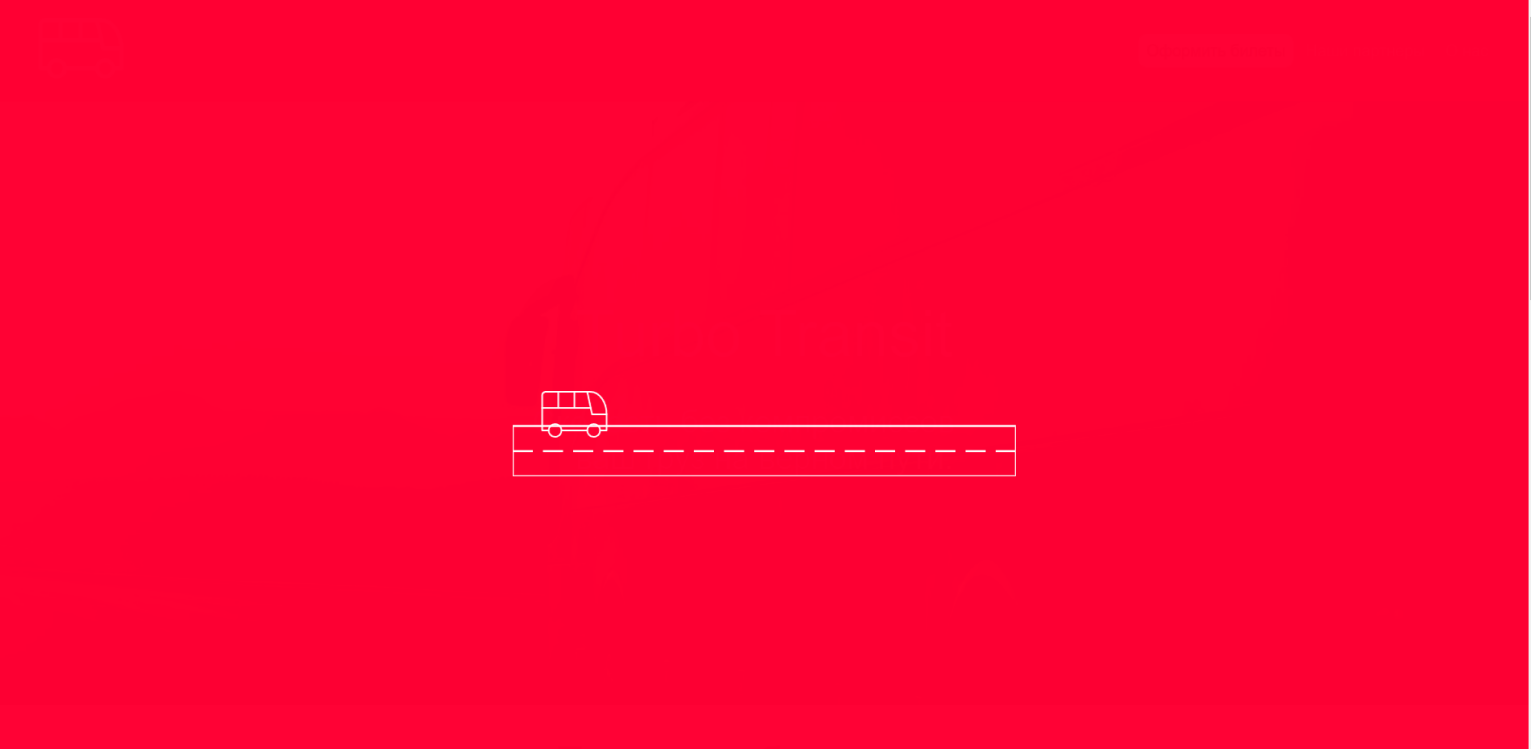


Рисунок 2.10 - preloader-эффект

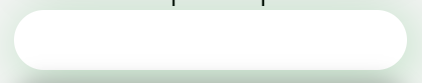
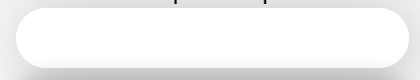
Ещё будут hover-эффекты: при наведении на элемент страницы он будет изменятся в размерах (пример на рисунке 2.11) или появлятся фоновая тень (пример на рисунке 2.12).



а б

а - не наведён, б - наведен

Рисунок 2.11 - при наведении курсора на элемент,он изменится в размерах



а б

а - не наведён, б - наведен

Рисунок 2.11 - при наведении курсора на элемент,

у него увеличится тень

Это сделано для того, чтобы пользователь на подсознательном уровне быстрее и точнее воспринимал направление курсора. Такой визуальный сигнал улучшает удобство взаимодействия, снижает когнитивную нагрузку и делает интерфейс более интуитивно понятным.

**2.7. Выводы**

На данном этапе создан прототип сайта, а затем и макет веб-страниц. Было определено стилевое оформление веб-сайта: основная цветовая палитра, шрифтовое оформление, элементы пользовательского интерфейса, а также эффекты и анимации.

**3. Реализация структуры веб-сайта**

**3.1. Структура HTML-документа**

Структура HTML-документа определяет базовую модель для будущего веб-сайта. Код выполнен по стандарту HTML5.

При создании структуры HTML-документа важно учитывать несколько ключевых моментов:

1. Семантика: Благодаря семантическим тегам браузер и поисковые системы лучше понимают содержимое страницы и правильно интерпретировать её структуру. Листинг представлен в приложении В.
2. Производительность: С помощью правильной и эффективной структуры, браузер понимает что нужно отобразить, что в свою очередь ускоряет процесс загрузки
3. Читаемость и ясность: Благодаря хорошо написанному коду, другие разработчики смогут быстрее разобраться в коде.
4. Масштабируемость и обслуживаемость: Правильная структура помогает без лишних усилий добавить или удалить, а также отредактировать элементы.

В проекте были использованы семантические теги header и footer.

Структура главной страницы данного курсового проекта представлена в листинге 3.1.

<!DOCTYPE html>

<html lang=«ru»>

<head> ... </head>

<body>

<header> ... </header>

<main>

<section id=«main»> ... </section>

<section id=«statistics»> ... </section>

<section id=«reviews»> ... </section>

<section id=«contact»> ... </section>

</main>

<footer> ... </footer>

</body>

</html>

Листинг 3.1 – Базовая структура HTML-документа

В теге <header> находится горизонтальное меню. Оно содержит ссылки на все остальные страницы сайта и логотип. Это позволяет пользователям быстро переходить между различными разделами сайта и вернуться на главную страницу. Навигационное меню было реализовано при помощи «бургер-меню». По умолчанию, бургер-меню скрывает ссылки на страницы, чтобы сохранить пространство на экране, и при нажатии на иконку бургер-меню отображает скрытые навигационные пункты. К скрытым пунктам относятся ссылки на страницы, а именно ссылка на странницу с оформлением билетов, «Наши партнеры»Б, «О нас».

Структура шапки представлена в листинге 3.2.

<header>

...

            <nav id=«headerMenu»>

                <ul>

                    <a href=«/tickets.html»><li id=«menuBttn1»>Оформить билеты</li></a>

                    <a href=«/routes.html»><li id=«menuBttn2»>Маршруты</li></a>

                    <a href=«/partners.html»><li id=«menuBttn3»>Наши партнеры</li></a>

                    <a href=«/aboutus.html»><li id=«menuBttn4»>О нас</li></a>

                </ul>

            </nav>

</header>

Листинг 3.2 – Структура шапки веб-страницы

В теге <footer> находятся дополнительные ссылки на различную информацию.

Структура тега <footer> представлен в листинге 3.3.

<footer>

<div id=«footerLogo»>

...

</div>

<ul id=«footerMenu»>

<a href=«tickets.html»><li id=«footerMenuLink1»>Оформить билеты</li></a>

<a href=«aboutus.html»><li id=«footerMenuLink3»>О компании</li></a>

<a href=«team.html»><li id=«footerMenuLink4»>Сотрудники</li></a>

<a href=«partners.html»><li id=«footerMenuLink5»>Партнеры</li></a>

</ul>

</footer>

Листинг 3.3 – Структура подвала веб-страницы

В проекте также предусмотрена структура страницы с маршрутами, на которую попадает пользователь при выборе одного из направлений. Пользователь может нажать на любой из предложенных рейсов и перейдет на страницу с подробной информацией о выбранном маршруте.

Структура выбора маршрут представлена в листинге 3.3.

<main>

<h2>Оформить билеты</h2>

<div id=«blox»></div>

<div id=«modal» class=«modal»>

<div class=«modal-content»>

<form action=«payment.html» id=«ticketForm»>

<div>

<p id=«pDate»>Выберите дату поездки</p>

<select id=«dateSelect» name=«date» required></select>

</div>

<p>Оставьте свои данные</p>

<div>

<p id=«pFio»>ФИО</p>

<input type=«text» name=«name» id=«fio» placeholder=«Иванов Иван Иванович» required>

</div>

<div>

<p id=«pPhone»>Номер телефона</p>

<input type=«text» name=«phone» id=«phone» value=«+375» required>

</div>

<p id=«formPrice»></p>

<button type=«button» class=«close-button»>Х</button>

<button type=«submit»>Оформить</button>

</form>

</div>

</main>

Листинг 3.3 – Структура продукта веб-сайта

Информация о продукте подгружается из XML, благодаря JS.

3.2. Добавление таблиц стилей SCSS и CSS

Вложенные стили SCSS — данная функция позволяет писать CSS правила и вкладывать их друг в друга. Вложение позволяет писать CSS в той же структуре, что и HTML, создавая визуальную иерархию в селекторах.

Использование вложенных стилей в SCSS упрощает чтение и организацию кода, особенно при работе с большими стилизованными компонентами. Кроме того, SCSS предоставляет дополнительные возможности, такие как использование переменных, миксинов и операций, что делает разработку CSS более эффективной и гибкой.

В рамках курсового проекта использовалась таблица стилей Scss. Одним из преимуществ Scss является возможность обращаться к элементам удобным и понятным способом с помощью вложенности. Вместо повторного указания родительских селекторов для каждого правила, можем просто вложить правила в соответствующие элементы. Это упрощает чтение и поддержку кода, а также позволяет легко представлять визуальную иерархию стилей.

Scss также поддерживает импортирование фрагментов кода из других файлов с помощью директивы @import. Это позволяет разделить таблицу стилей на несколько файлов и импортировать их в один основной файл.

Структура вложенности Scss представлена в листинге 3.4.

main {

    #main {

        display: flex;

        background-image: url(«/img/bg.png»);

        background-position: center;

        background-size: cover;

        color: #FFF;

        h1 {

            font-size: 64px;

            font-weight: normal;

            margin: 0;

        }

    }

}

Листинг 3.4 – Структура продукта веб-сайта

В scss была использована вложенность, а также использование переменных.

**3.3. Использование стандартов XML (SVG)**

XML- это язык разметки, который используется для хранения и передачи структурированных данных. Он представляет собой текстовый формат, который позволяет организовывать данные в иерархическую структуру с помощью тегов.

Пример хранения данных в XML-формате проиллюстрирован в листинге 3.5.

<?xml version=«1.0» encoding=«UTF-8»?>

<tickets>

<ticket>

<id>1</id>

<route>Мозырь-Минск</route>

<departure>15:03</departure>

<comming>18:03</comming>

<days>пн, вт, ср, чт, пт</days>

<price>23.16 BYN</price>

</ticket>

<ticket>

...

</ticket>

</tickets>

Листинг 3.5 – Пример хранения данных в XML-формате

SVG – это векторная графика, которая при увеличении не теряет качества.

SVG изображения имею ряд преимуществ. SVG бесконечно масштабируем, отзывчив, имеет очень маленький размер файла и может быть стилизован, анимирован при помощи веб-технологий таких как CSS и JavaScript. Пример SVG изображения представлен в листинге 3.7.

<svg width=«97» height=«97» viewBox=«0 0 97 97» fill=«none» xmlns=«http://www.w3.org/2000/svg»>

<g clip-path=«url(#clip0\_49\_1090)»>

<path d=«...» stroke=«#F3E7E7» stroke-width=«5» stroke-linecap=«round» stroke-linejoin=«round»/>

</g>

<defs>

<clipPath id=«clip0\_49\_1090»>

<rect width=«96» height=«96» fill=«white» transform=«translate(0.5 0.0605469)»/>

</clipPath>

</defs>

</svg>

Листинг 3.7 – Пример SVG изображения

В качестве программы для редактирования векторного изображения может быть использована Figma [6]. В ней можно изменить уже существующий SVG-файл и создать с нуля. Для этого есть такие инструменты, как перо, квадрат, линия, стрелка, эллипс, треугольник и звезда.

# **3.4 Управление элементами DOM**

JavaScript (JS) является высокоуровневым языком программирования, который интерпретируется браузером и используется в веб-разработке для создания интерактивных и динамических веб-страниц. Он предоставляет широкий набор функций и возможностей для добавления различного поведения на веб-сайтах.

JavaScript позволяет разработчикам обрабатывать события, взаимодействовать с пользователями, изменять содержимое страницы динамически, отправлять и получать данные с сервера без перезагрузки страницы. Создавать анимацию, работать с графикой и многое другое.

JavaScript является одним из ключевых инструментов веб-разработки и широко применяется для создания современных и интерактивных веб-приложений, а также для динамического изменения содержимого веб-страниц и создания сложных анимаций и различных интерактивных сценариев. Пример использования JavaScript показан в листинге 3.8.

function validatePhone(phone) {

const regex = /^\+375\d{9}$/; // Формат +375XXXXXXXXX

return regex.test(phone);

}

phoneInput.addEventListener('input', function() {

let phone = phoneInput.value.trim();

// Форматируем номер телефона

if (phone.startsWith('375') && phone.length === 12) {

phone = '+375' + phone.slice(3);

}

phoneInput.value = phone;

});

Листинг 3.8 – Пример использования JavaScript для валидации номера телефона

Если номер был введён правильно, то вверху экрана появится надпись: «Скоро с вами свяжется оператор».

В рамках курсового проекта JS использовался для подключения XML документов к странице продукта. Пример использования JavaScript в листинге 3.9.

fetch('tickets.xml')

.then(response => response.text())

.then(xmlString => {

const parser = new DOMParser();

const xmlDoc = parser.parseFromString(xmlString, «text/xml»);

const tickets = xmlDoc.getElementsByTagName('ticket');

const container = document.getElementById('blox');

const cityFilter = document.getElementById('cityFilter');

// Создаем список городов

const cities = new Set();

Array.from(tickets).forEach(ticket => {

const route = ticket.getElementsByTagName('route')[0].textContent;

const [from, to] = route.split('-');

cities.add(from.trim());

cities.add(to.trim());

});

// Добавляем города в список

cities.forEach(city => {

const option = document.createElement('option');

option.value = city;

option.textContent = city;

cityFilter.appendChild(option);

});

Листинг 3.9 – Пример использования JS для подключения XML

Из XML-документа бралась информация о существующих маршрутах, после чего названия направлений подключались к определённым тегам HTML-документа. Далее скрипт возвращался в XML и искал рейсы по выбранному направлению в HTML, создавался контейнер с расписанием рейсов, и соответствующий рейс добавлялся в этот контейнер.

Пример JS, который оставляет только рейсы, где начальный или конечный пункт совпадает с выбранным городом отображен в листинге 3.10.

cityFilter.addEventListener('change', function() {

displayTickets(this.value); // Фильтрация по городу

});

const displayTickets = (filterCity = 'all') => {

container.innerHTML = '';

Array.from(tickets).forEach(ticket => {

const route = ticket.getElementsByTagName('route')[0].textContent;

const [from, to] = route.split('-');

if (filterCity === 'all' || from.trim() === filterCity || to.trim() === filterCity) {

container.innerHTML += createTicketElement(ticket);

}

});

};

Листинг 3.10 – Пример использования JS для поиска совпадения начального и конечного пункта с выбранным городом

Благодаря JavaScript стало возможным сделать сайт в разы удобнее и красивее.

**3.5 Выводы**

В данном этапе разработки был выполнен полный цикл создания современной и функциональной структуры веб-сайта, где использованы новейшие технологии и лучшие практики веб-разработки.

Основой проекта стала разметка HTML с активным использованием семантических тегов, что значительно улучшило читаемость кода и повысило его доступность для поисковых систем и вспомогательных технологий. Вёрстка выполнена по современным стандартам, что гарантирует корректное отображение на различных устройствах и в разных браузерах.

Для создания визуального стиля и организации дизайна использованы таблицы стилей, написанные на языке SCSS. Этот подход позволил структурировать CSS-код, вложенность и наследование. Благодаря использованию SCSS, структура сайта стала гибкой и легко масштабируемой.

Был реализован функционал поиска маршрутов по названию пункта назначения, выбранного пользователем.

# 4 Тестирование веб-сайта

# 4.1 Адаптивный дизайн веб-сайта

Для достижения адаптивности в проекте применялись методы flex верстки, а также использовались медиа-запросы. С помощью flex-верстки была разработана основная структура контента на странице. Использование свойства minmax позволило динамически изменять размер блоков в зависимости от ширины экрана.

Ресурс должен одинаково хорошо демонстрироваться на экране компьютера и телефона. Внешний вид настольной версии веб-сайта представлен на рисунке 4.1.



Рисунок 4.1 – Внешний вид настольной версии веб-сайта

Все элементы удобно расположены и имеют лаконичный и минималистичный вид. Внешний вид мобильной версии сайта представлен на рисунке 4.2



Рисунок 4.2 – Внешний вид мобильной версии сайта

Адаптивность является очень важным аспектом веб-разработки, так как позволяет сайту корректно отображаться на устройствах с разными размерами экранов. С увеличением числа устройств с различными размерами экранов, становится критически важным, чтобы сайт был доступен и на всех устройствах.

# 4.2 Кроссбраузерность веб-сайта

Кроссбраузерность – важный критерий корректной работы веб-ресурса. Для проверки соответствия этому критерию сайт был открыт в нескольких браузерах. Внешний вид веб-сайта в браузере Microsoft Edge представлен на рисунке 4.3.

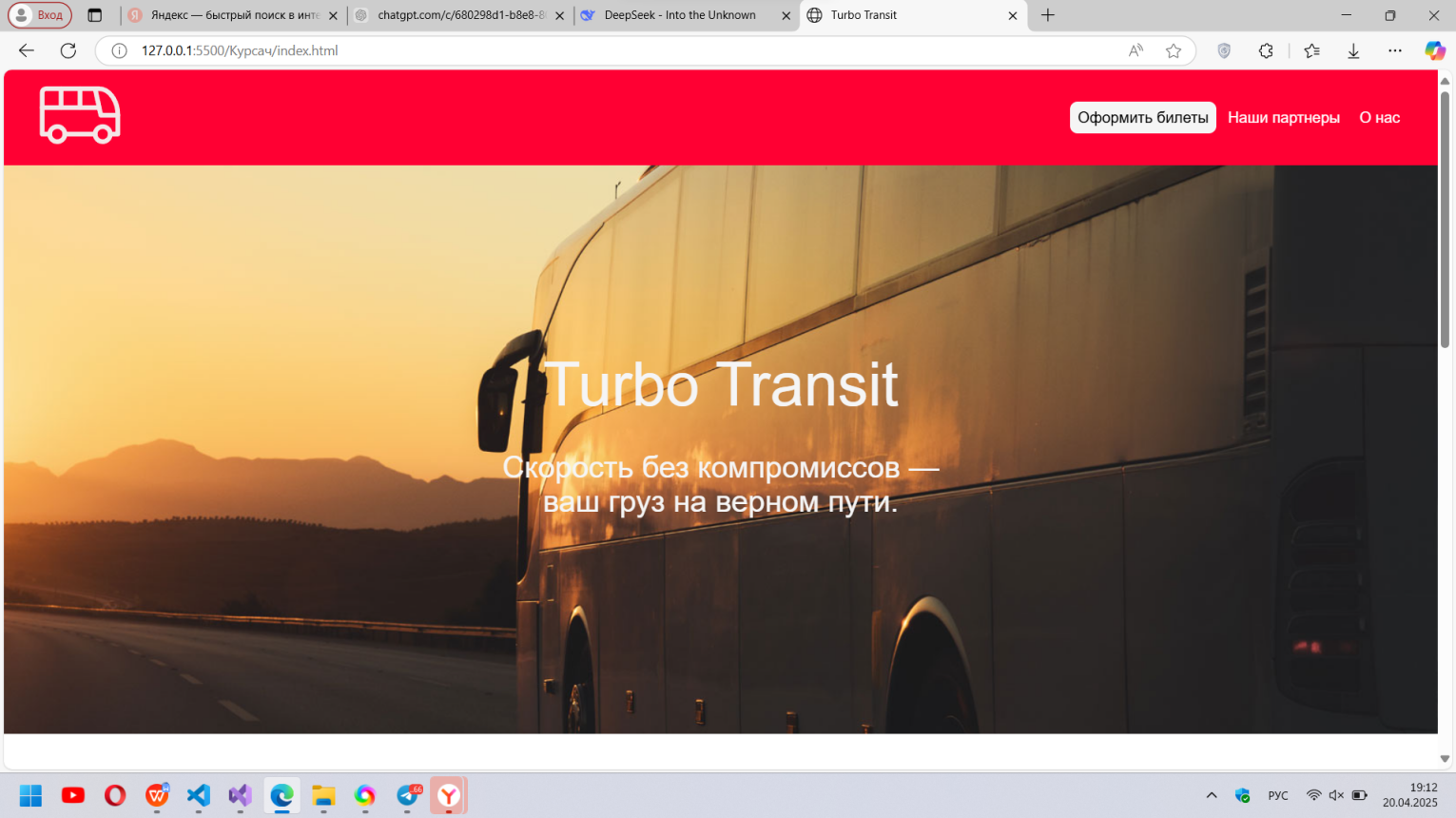


Рисунок 4.3 – Внешний вид веб-сайта в браузере Microsoft Edge

Так же сайт был протестирован в браузере Opera. Его внешний вид в этом браузере представлен на рисунке 4.4.

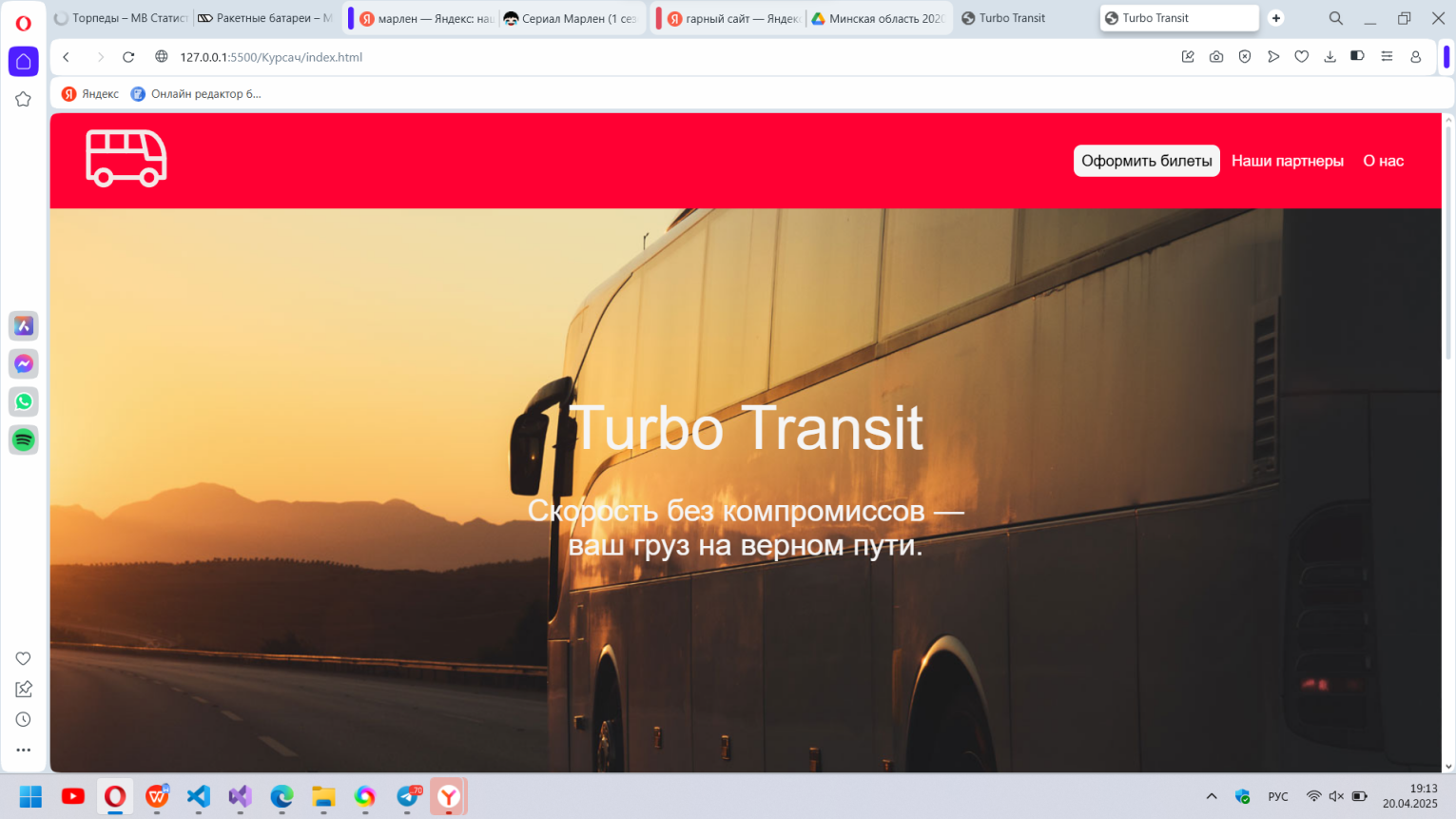


Рисунок 4.4 – Внешний вид веб-сайт в браузере Opera

Проведя тестирование веб-сайта в разных браузерах подтвердилось его корректное отображение во всех популярных браузерах. Что означает отличную кроссбраузерность веб-сайта.

# 4.3 Руководство пользователя

Интерфейс сайта интуитивно понятен пользователю. Основное взаимодействие с сайтом в поиске товара удобнее всего осуществлять с помощью ссылок на различные страницы. Ссылки продемонстрированы на рисунке 4.6.



Рисунок 4.6 – Ссылки для навигации по сайту

С их помощью пользователь может перейти на интересующую его страницу, для последующего изучения на странице. Страница с информацией о доступных билетах продемонстрирована на рисунке 4.7.

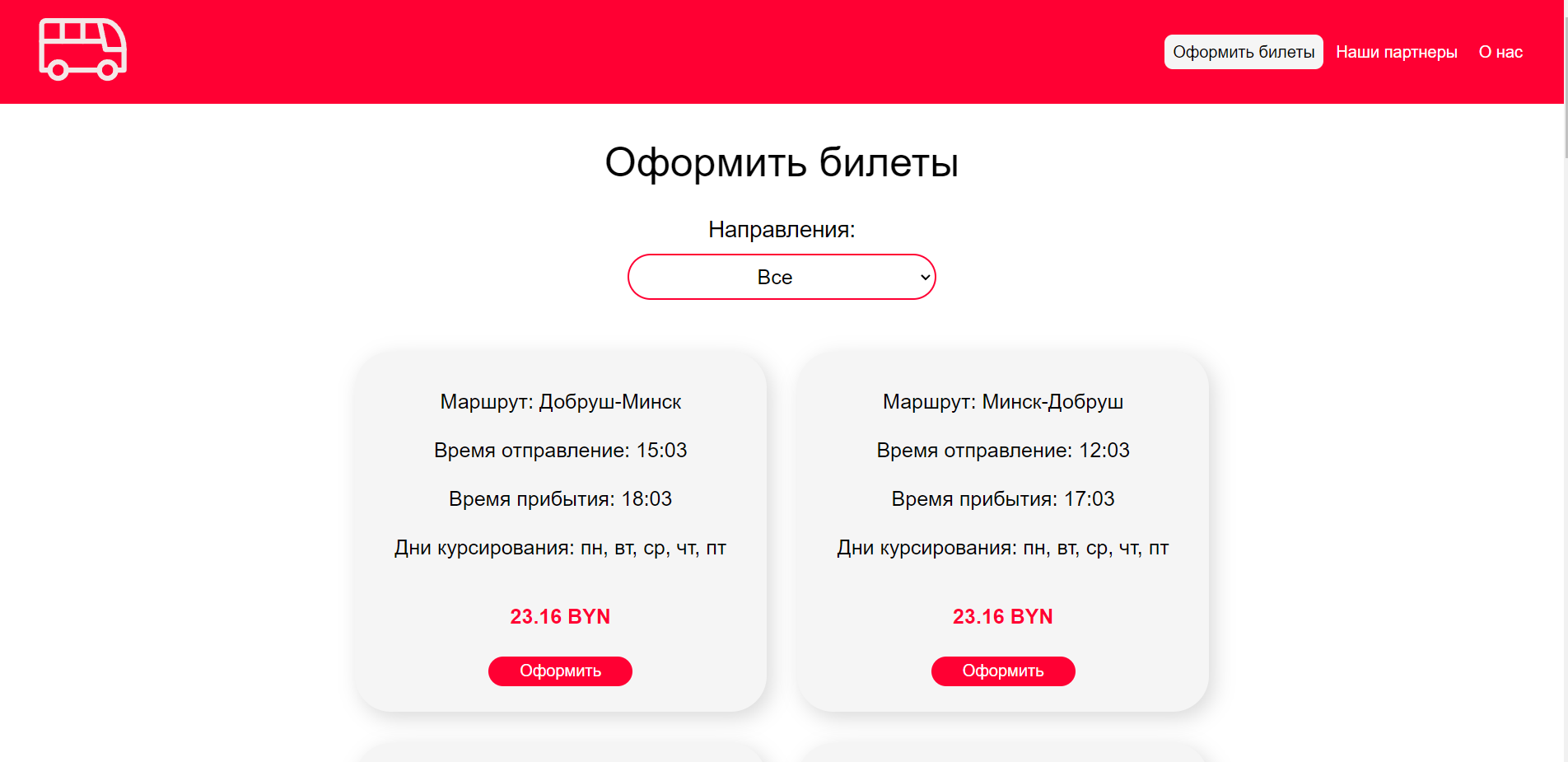


Рисунок 4.7 – Страница с информацией

На этой странице пользователь может выбрать интересующий его билет и по нажатию на кнопку «Оформить», пользователя перенаправит на страницу с вводом данных о банковской карте. Её внешний вид представлен на рисунке 4.9.

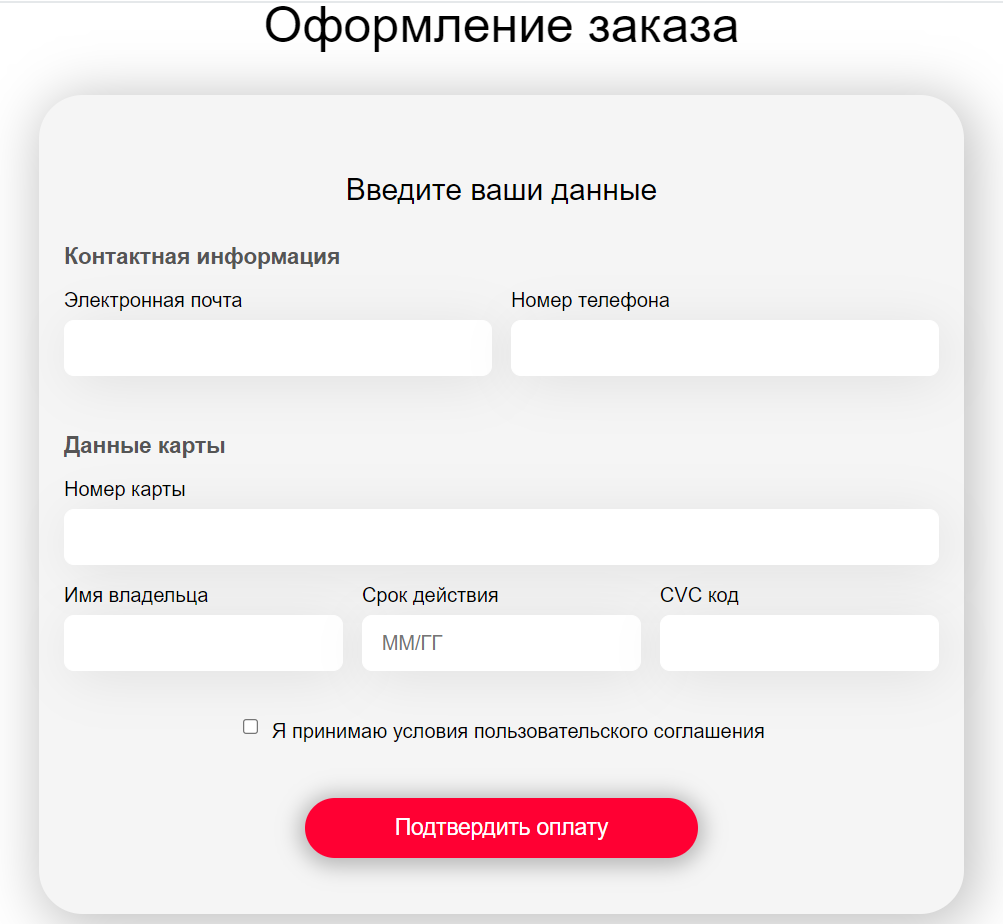


Рисунок 4.9 – Страница ввода данных банковской карты

После корректного ввода данных о банковской карте окно ввода сменится на текст о успешном оформлении и появится кнопка, возвращающая пользователя на главную страницу. Внешний вид этого окна представлен на рисунке 4.10.

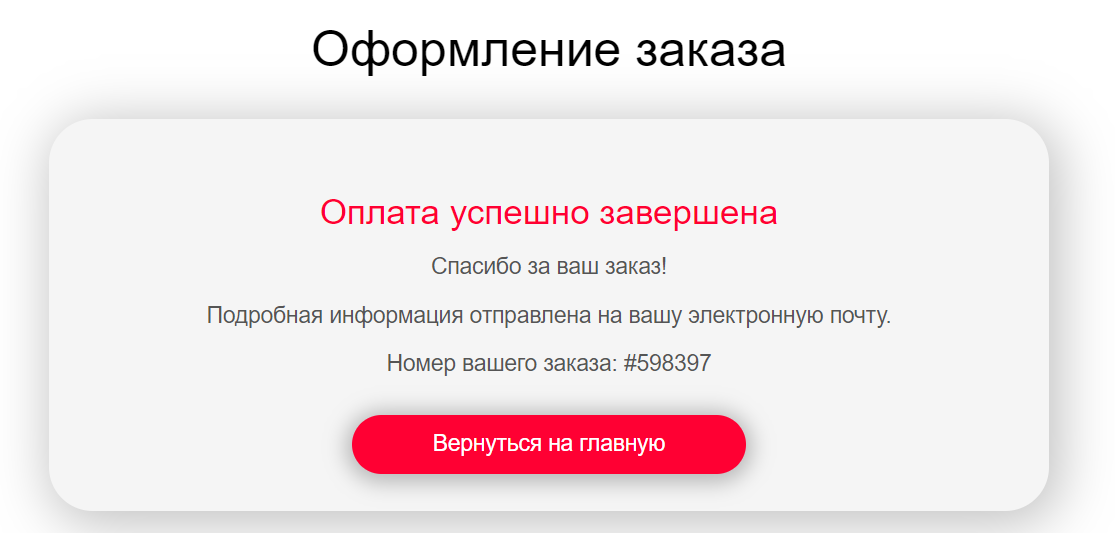


Рисунок 4.10 – Внешний вид информирующего окна

Таким образом, был разработан сайт с целью обеспечить максимально простое и интуитивно понятное использование.

# 4.4 Тестирование кода

Тестирование кода веб-страниц является критически важной частью процесса веб-разработки. Это процесс проверки кода, чтобы убедиться, что он работает так, как должен работать, и что он соответствует определенным стандартам.

Тестирование позволяет убедиться, что код работает правильно и соответствует заданным требованиям. Это помогает обеспечить высокое качество и надежность сайта. Тестирование помогает обнаружить уязвимости и потенциальные угрозы безопасности, что позволяет разработчикам принимать меры для их устранения до того, как они могут быть использованы злоумышленниками. Ошибка в коде, которая обнаруживается на ранней стадии, может быть исправлена быстрее.

Весь код был протестирован с помощью онлайн-сервиса linter. Результат представлен на рисунке 4.11

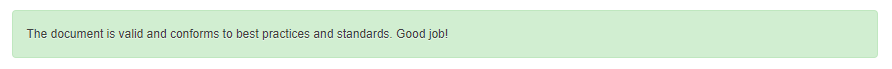


Рисунок 4.11 – Результат тестирования кода с помощью linter.

В целом, тестирование кода веб-страниц является неотъемлемой частью процесса веб-разработки и помогает гарантировать высокое качество, безопасность и удобство использования сайта. Полный код проекта можно получить в Github-репозитории [7].

# 4.5 Выводы

В данном разделе была проведена проверка адаптивности, кроссбраузерности и стабильности работы сайта на различных устройствах и в популярных браузерах. Тестирование охватывало ПК, ноутбуки и смартфоны с разными разрешениями экранов. Интерфейс корректно масштабируется благодаря использованию flexbox и media-запросов, обеспечивая удобство на сенсорных экранах.

Также сайт был протестирован в браузерах Microsoft Edge и Opera. Не были выявлены значительные отличия в отображении и анимациях. Сайт работает одинаково во всех браузерах.

Дополнительно создано краткое руководство пользователя с пошаговыми инструкциями и скриншотами, облегчающее освоение функционала сайта и повышающее удобство его использования.

**Заключение**

В ходе проделанной работы был создан полноценный сайт автовокзала.

При разработке данного веб-сайта был использован широкий ряд языков разметки и таблиц стиля, который позволил увеличить функциональность и многократно повысил качество конечного продукта. Была использована адаптивная верстка для корректного отображения на всех устройствах.

В ходе выполнения данной задачи были рассмотрены плюсы и минусы аналогичных решений. В процессе осмотра выяснилось, каких тенденции в разработке дизайна веб-сайта стоит придерживаться.

При формировании технического задания были продумано содержание основных страниц веб-сайта. Были выбраны инструменты для разработки веб-сайта.

При проектировании были разработаны прототипы, а далее созданы макеты дизайна страниц веб-сайта, при опоре на который был реализован сам веб-сайт. Для разработки прототипов и макетов станиц было использована программа Figma, ранее не изученная в ходе учебного процесса.

Были разработаны пользовательские элементы, спецэффекты и логотип, для идентификации веб-сайта, среди миллионов других интернет ресурсов.

Для качественного отображения картинок, на любом устройстве, было решено применять SVG-изображения, также был создан XML-документ для последующей вставки на веб-страницу HTML.

В ходе тестирования была продемонстрирована валидность, кроссбраузерность и адаптивность веб-сайта. Было создано подробное руководство пользователя для использования веб-сайта.

В пояснительной записке описана реализация поставленных в рамках курсового проекта ряда задач:

1. Проанализировать существующие языки разметки, инструменты и библиотеки для создания веб-сайта.
2. Разработать макет и прототип сайта.
3. Разработать структуру веб-сайта.
4. Наполнить сайт информацией по теме.
5. Протестировать веб-сайт.
6. Разработать руководство пользователя.

Таким образом была полностью достигнута поставленная цель по разработке веб-сайта, были учтены все требования, все задачи курсового проекта выполнены.