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

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 1-40 05 01 «Информационные системы и технологии»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА**

по дисциплине «Компьютерные языки разметки»

Тема «Веб-сайт «сервис по доставке еды»

**Исполнитель**

студент 1 курса 1 группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю. К. Добрего

подпись, дата

**Руководитель**

старший преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. В. Харланович

должность, учен. степень, ученое звание подпись, дата

Допущен(а) к защите \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

дата, подпись

Курсовой проект защищен с оценкой

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ А. В. Харланович

подпись дата инициалы и фамилия

Содержание

|  |  |
| --- | --- |
| Введение | 3 |
| 1. Постановка задачи | 4 |
| 1.1. Обзор аналогичных решений | 4 |
| 1.2. Техническое задание | 9 |
| 1.3. Выбор средств реализации программного продукта | 9 |
| 1.4. Выводы | 10 |
| 2. Проектирование страниц веб-сайта | 12 |
| 2.1. Выбор способа верстки | 12 |
| 2.2. Выбор стилевого оформления | 12 |
| 2.3. Выбор шрифтового оформления | 13 |
| 2.4. Разработка логотипа | 13 |
| 2.5. Разработка пользовательских элементов | 13 |
| 2.6. Разработка спецэффектов | 17 |
| 2.7. Выводы | 20 |
| 3. Реализация структуры веб-сайта | 21 |
| 3.1. Структура HTML-документа | 21 |
| 3.2. Добавление таблиц стилей SCSS и CSS | 23 |
| 3.3. Использование стандартов XML (SVG) | 25 |
| 3.4. Управление элементами DOM | 26 |
| 3.5. Выводы | 30 |
| 4. Тестирование веб-сайта | 31 |
| 4.1. Адаптивный дизайн веб-сайта | 31 |
| 4.2. Кроссбраузерность веб-сайта | 32 |
| 4.3. Руководство пользователя | 33 |
| 4.4. Выводы | 34 |
| Заключение | 35 |
| Список использованных литературных источников | 28 |
| Приложение А Прототипы веб-страниц | 29 |
| Приложение Б Макет структуры веб-сайта | 30 |
| Приложение В Листинг НТML-документа | 33 |
| Приложение Г Листинг SCSS и CSS | 38 |
| Приложение Д Листинг XML-файлов | 44 |
| Приложение Е Листинг SVG | 46 |
| Приложение Ж Листинг JavaScript | 46 |
|  |  |

Введение

Актуальность курсового проекта обусловлена тем, что в связи с развитием современных технологий популярность онлайн-сервисов возрастает. На сегодняшний день существует множество интернет-магазинов и сервисов, которые базируются исключительно в интернете.

Размещение магазина/сервиса исключительно в интернете сегодня является популярной практикой, так как все больше пользователей находит такой способ получения услуг более комфортным.

Целью курсового проекта является создание современного и качественного сайта для веб-сервиса по доставке еды, который будет соответствовать современным требованиям к дизайну и функциональности веб-сайтов, а также удовлетворять потребностям пользователей.

В процессе работы над проектом были поставлены следующие задачи:

1. Разработка визуальной концепции и дизайн-макетов сайта.
2. Верстка и адаптивная вёрстка сайта.
3. Создание кроссбраузерного веб-сайта.
4. Разработка функциональности сайта.
5. Добавление анимаций для веб-сайта.
6. Тестирование сайта.

Целевая аудитория для данного курсового проекта — люди, которые предпочитают быстрое и удобное решение для оформления заказа еды на дом. Данная услуга может быть интересна людям в возрасте от 18 до 30 лет, которые активно пользуются интернетом и мобильными приложениями.

# 1 Постановка задачи

# 1.1 Обзор аналогичных решений

Для сервисов, базирующихся онлайн, крайне важным является конкурентоспособность в интернете. Поэтому необходимо проанализировать сайты конкурентов.

В качестве аналогов были выбраны веб-сайты сервисов с такой же тематикой: «Пицца-лисицца»[1], «KFC» [2], «SmashBurger» [3].

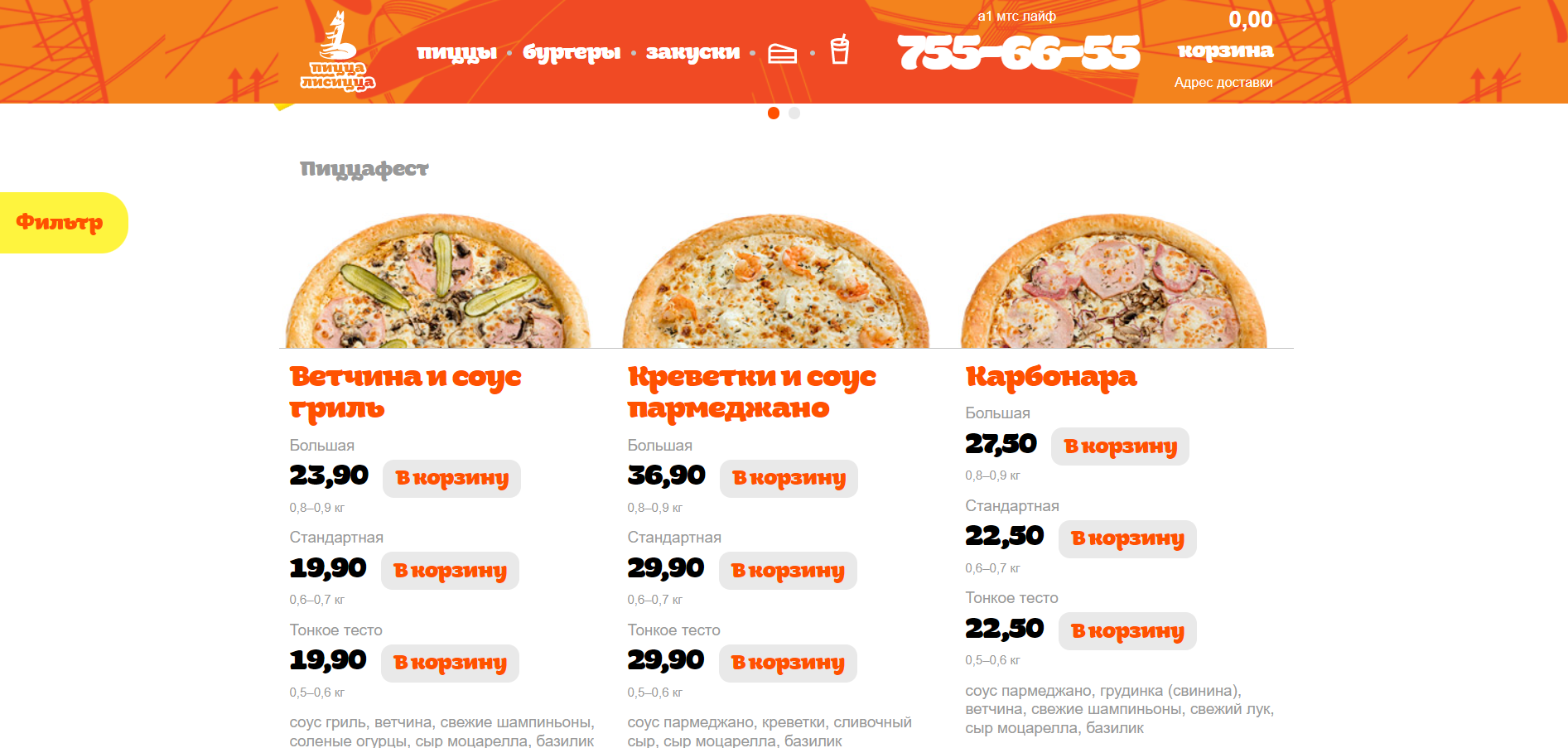
В качестве первого аналога был выбран сайт «Пицца-лисицца». Он является самым простым из перечисленных. Меню располагается на главной странице, представленной на рисунке 1.1.

Рисунок 1.1 – Пример главной страницы

Это не лучшее решение, так как ухудшает читаемость сайта и создает ощущение дешевизны.

Навигация по меню осуществляется при помощи ссылок, расположенных в хедере. Благодаря фиксированному меню, пользователь может легко перемещаться по странице.

Футер (реализованный при помощи тега <div>) занимает весь экран и заполнен информацией, которую трудно анализировать. При этом основной массив данных, который абсолютно не нужен пользователю, сливается с важной информацией, например, с графиком работы. (рисунок 1.2).

Рисунок 1.2 – Пример реализации футера

Хоть ресурс и выполняет основные функции, а именно: знакомит пользователя с товаром и услугами, он не предлагает комфортной среды и приятного дизайна: сайт кажется перегруженным, важная информация ускользает.

Оформление заказа осуществляется при помощи заполнения формы (реализовано при помощи тега <form>), куда заносится основная информация потребителя (адрес, мобильный телефон), что является простым и функциональным способом оформить заказ. (рисунок 1.3)

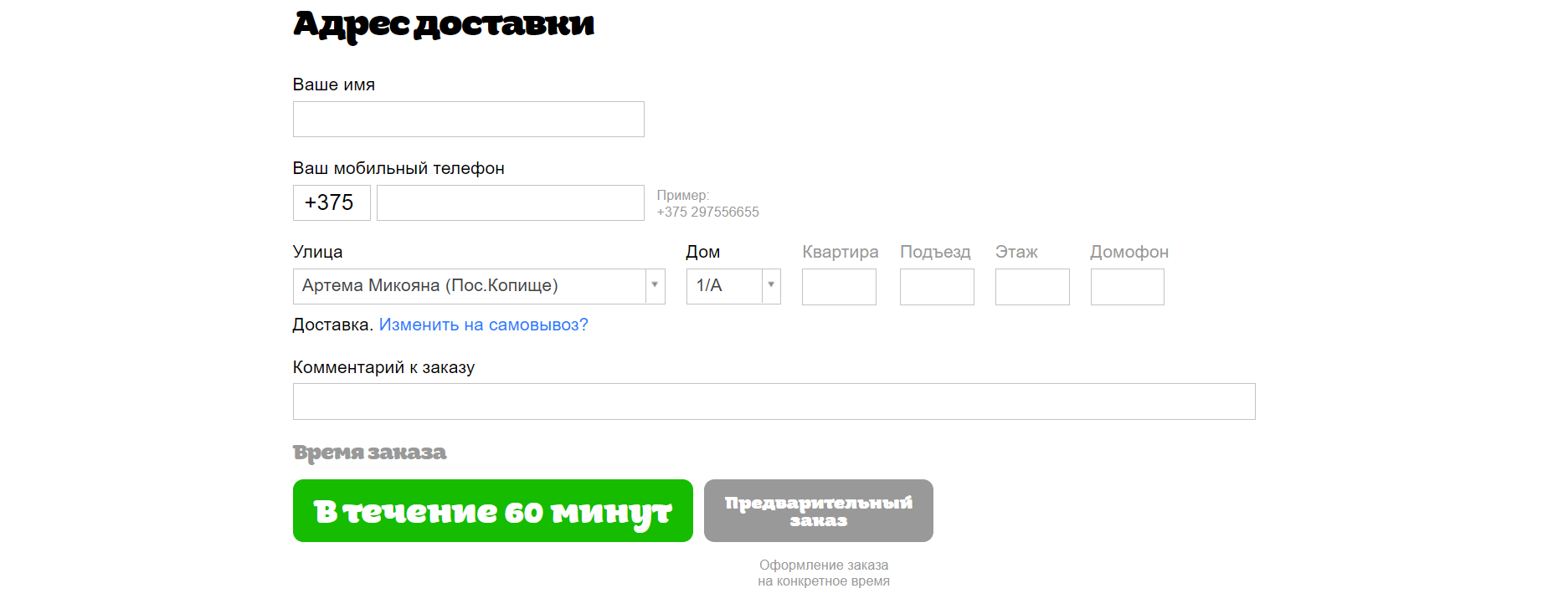


Рисунок 1.3 – форма оформления заказа

В цветовой палитре сайта преобладают яркие насыщенные цвета: оранжевый, белый, синий и зеленый для акцентов. Из-за яркости, цвета немного конфликтуют между собой.

Цветовая палитра представлена на рисунке 1.4.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

Рисунок 1.4 – Цветовая палитра сайта

Преобладают яркие цвета. Такой выбор палитры можно обосновать следующим образом:

1. Оранжевый также привлекает внимание и вызывает аппетит, ассоциируется с энергией, теплотой и сочностью, что отлично сочетается с пиццей как ярким и вкусным блюдом.
2. Белый используется, чтобы сделать сайт более открытым и дружелюбным для пользователей. Белый также часто используется в дизайне интернет-магазинов, так как позволяет подчеркнуть товар без отвлекающих элементов.
3. Голубой также ассоциируется с чистотой и свежестью, что может быть важно на сайте доставки еды.
4. Зеленый подходит для сайта доставки еды, так как ассоциируется с здоровым образом жизни, а также может быть использован для подчеркивания чистоты и свежести ингредиентов.

В ходе анализа были сделаны следующие выводы: необходимо разделить информацию на сайте на информационные блоки, что визуально разгрузит веб-сайт, сделает нахождение на нем приятным и удобным и поощрит потребителя к покупке.

При анализе функционала были выявлены следующие преимущества:

1. Удобное оформление заказа.

К недостатку в функционале веб-сайта можно отнести:

1. Загруженный дизайн сайта.
2. Конфликтующая цветовая палитра.
3. Неактуальный дизайн.

В качестве второго аналога был выбран сайт «KFC» — сайт ресторана быстрого питания с возможностью доставки.

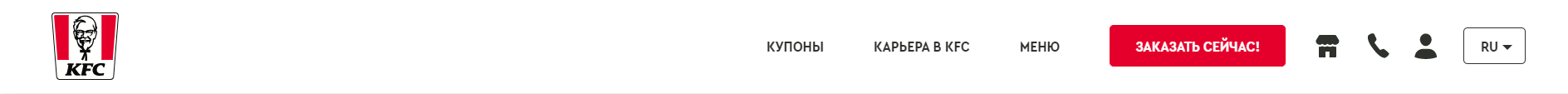
Информация рассредоточена на нескольких основных страницах, ссылки на которые расположены в фиксированном хедере (рисунок 1.5).

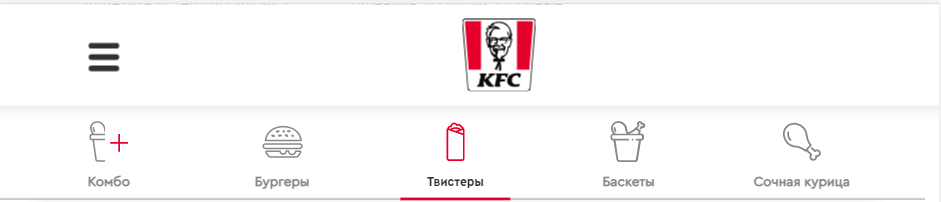
Рисунок 1.5 – Пример хедера с ссылками

Такой подход гарантирует удобство в навигации по сайту.

Меню реализовано при помощи поля навигационной панели, фиксированной в крайнем левом положении, а также поля с продуктами (рисунок 1.6).

Рисунок 1.6 – Пример меню с боковой навигационной панелью

Такое положение меню позволяет легко ориентироваться на странице. Данное меню также является адаптивным и меняет свою структуру на мобильных разрешениях (рисунок 1.7).

Рисунок 1.7 – Пример адаптивности навигации меню

На рисунке 1.7 также видно, что хедер преобразовывается в меню-бургер, что является хорошей практикой, так как позволяет сохранить место и облегчить дизайн визуально.

На сайте преобладают следующие цвета:

1. Белый цвет делает сайт более дружественным и подчеркивает товар.
2. Красный цвет ассоциируется с энергией, страстью, аппетитом и вызывает чувство голода.

При анализе функционала были выявлены следующие преимущества:

1. Удобное адаптивное меню товара.
2. Адаптивное навигационное меню-бургер в шапке.

Недостатки, выявленные при анализе:

1. Преимущественно белое пространство сайта создает ощущение пустоты.

Последним разобранным аналогом стал сайт «SmashBurger». Интересными элементом сайта является бургер-меню, которое появляется сверху (рисунок 1.8).

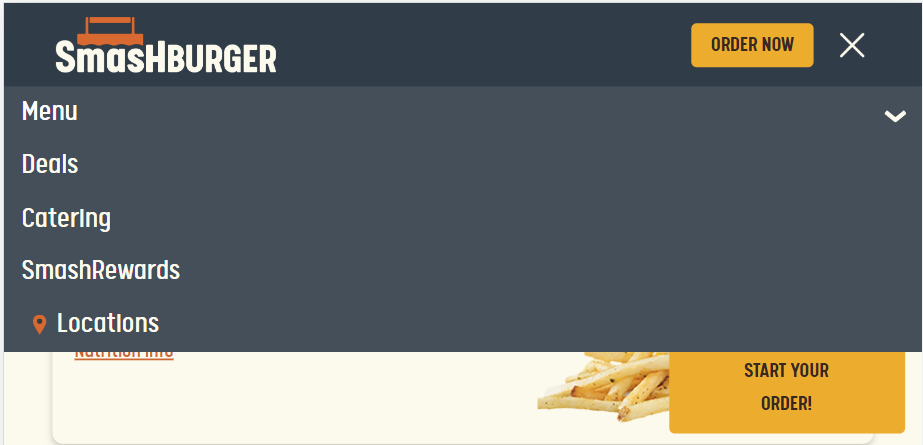


Рисунок 1.8 – Пример бургер-меню

Также сайт отличается яркой и приятной цветовой гаммой, очень приятной для глаз (рисунок 1.9).

|  |  |  |
| --- | --- | --- |
|  |  |  |

Рисунок 1.9 – Цветовая палитра сайта

Выбор цветов соответствует трендам дизайна, сайт приятен для глаз. Цвета из палитры имеют следующие значения:

1. Синий цвет ассоциируется с чистотой, порядком и доверием, что может быть важно для сайта.
2. Горчично-желтый цвет является ярким и привлекательным, и может привлечь внимание посетителей сайта. Кроме того, этот цвет может вызывать аппетит.
3. Бежевый цвет уравновешивает синий и горчично-желтый цвета и добавляет нейтральности сайту, что может быть важно для восприятия сайта как профессионального и надежного.

Преимущества, выявленные при анализе:

1. Интересные цветовые решения привлекают внимание.
2. Удобная навигация по сайту и меню.

# 1.2. Техническое задание

Цель проекта создать многостраничный сайт сервиса по доставке еды. Сайт должен содержать общую информацию о сервисе: время работы, контактные данные, история – а также каталог продуктов и корзину для продуктов. Основным контентом будут фотографии продуктов и описания к ним.

Главными задачами проекта является создание сайт по доставке еды, который обеспечит удобную и быструю возможность выбора, заказа еды. Задачей сайта является привлечение клиентов и побуждение их к заказу еды на сайте. На сайте также будет размещена информация об текущих акциях в сервисе по доставке еды. Сайт должен быть интерактивным, простым и интуитивно понятным. Данный сайт предназначен для желающих заказать еду через интернет.

На всех страницах сайта должна быть шапка(header) с меню для навигации по сайту и подвал(footer) с контактами, временем работы сервиса и другой контактной информацией.

На главной странице должны быть расположены некоторые элементы меню. На странице «О нас» должна быть расположена информация о сервисе. На странице «Меню» должен быть расположен полный список доступных товаров. На странице «Акции» должна быть расположена актуальная информация об акциях. На странице «Корзина» должны быть расположены продукты, выбранные пользователем в меню, а также должна быть представлена возможность оформить заказ.

Основные требования к исполнителю: создание дизайна, проведение анализа существующих решений, создание прототипа, адаптивность и кроссбраузерность сайта. В проекте используется гипертекстовый язык разметки HTML, CSS/Sass, графические элементы в формате SVG; данные хранятся в XML-формате.

­­­Задачи курсового проекта:

1. Разработать удобный и простой интерфейс сайта.
2. Разработать меню.
3. Разместить всю необходимую информацию на сайте.
4. Разработать корзину для заказов.

# 1.3 Выбор средств реализации программного продукта.

Для реализации проекта были выбраны языки: HTML, CSS/Sass, XML, JavaScript.

HTML (HyperText Markup Language) - это язык разметки, используемый для создания веб-страниц. Он определяет структуру и содержание веб-страницы, включая заголовки, текст, ссылки, изображения, таблицы, формы и другие элементы.

HTML5 - это последняя версия HTML, которая включает новые элементы и атрибуты, более семантическую разметку, аудио и видео со встроенными функциями, а также поддержку мобильных устройств.

JavaScript - это язык программирования, который используется для создания интерактивных элементов на веб-страницах. Он может быть использован для создания динамических эффектов, валидации форм, обработки событий и много другого.

CSS (Cascading Style Sheets) - это язык таблиц стилей, который используется для оформления веб-страниц. Он позволяет определить цвет, шрифт, размер, расположение и другие аспекты внешнего вида элементов на веб-странице.

Sass (Syntactically Awesome Style Sheets) - это препроцессор CSS, который расширяет функциональность CSS, добавляя переменные, условия, циклы и другие функции. Это упрощает и ускоряет процесс создания таблиц стилей.

XML (Extensible Markup Language) - это язык разметки, который используется для хранения и передачи данных. Он позволяет создавать пользовательские теги и атрибуты, что делает его более гибким и расширяемым, чем HTML.

Все эти технологии используются в веб-разработке для создания интерактивных и эстетически привлекательных веб-страниц. HTML определяет структуру страницы, CSS определяет ее внешний вид, JavaScript позволяет создавать интерактивные элементы, а Sass расширяет функциональность CSS. XML используется для хранения и передачи данных. Все эти технологии взаимодействуют между собой, обеспечивая создание качественных веб-сайтов.

Курсовая работа выполняется в редакторе кода VS Code.

Visual Studio Code (VS Code) - это бесплатный и открытый исходный код редактор кода, разработанный Microsoft. Он поддерживает множество языков программирования и платформ, в том числе HTML, CSS, JavaScript и другие.

VS Code имеет множество функций, которые облегчают написание сайтов, например:

1. Поддержка Live Server: VS Code поддерживает расширение Live Server, которое позволяет запустить локальный сервер для разработки и мгновенного обновления изменений на странице в браузере.
2. Автодополнение кода.
3. Подсветка синтаксиса и тд.

# Вывод

В данном разделе были рассмотрены аналогичные решения, были проанализированы их недостатки и преимущества. В ходе анализа предметной области и аналогов был составлен список необходимых требований для продукта данной тематики:

1. Информация на сайте должна быть легкодоступна. Для достижения этой цели было принято решения создать фиксированную шапку(header) для навигации по сайту и подвал(footer) с важной информацией. Для упрощения доступа к ссылкам на мобильных версия сайта шапка будет преобразована в бургер-меню. Навигация по странице меню будет осуществляться при помощи фиксированной боковой навигационной панели, а также фиксированной панели под шапкой для мобильных версий сайта.
2. Информация должна легко восприниматься. Чтобы избежать визуальной перегруженности сайта, было принято решение о разделении массива информации на логические блоки-страницы.
3. Веб-сайт должен быть эстетически приятным. Хороший вид сайта побуждает пользователя остаться, что крайне важно для веб-сайта интернет-сервиса. Данная цель будет достигнута подбором интересной цветовой палитры сайта и добавлением анимационных эффектов.

Также были поставлены конкретные задачи для создания интерактивного Интернет-ресурса, были рассмотрены средства реализации программного продукта, такие как языки разметки, редактор кода.

# Проектирование страниц веб-сайта

# Выбор способа верстки

В соответствие с требованиями к проекту, в частности адаптивность и кроссбраузерность сайта, было принято решение об использовании media запросов. Media запросы поддерживаются всеми современными браузерами. Media запросы позволяют создавать стили, которые будут применяться только когда определенные условия будут выполнены. Это позволяет нам создавать адаптивный дизайн, который будет выглядеть хорошо на различных устройствах и экранах.

Для создания отдельных блоков страницы, такие как горизонтальное меню, меню продуктов, будет использована Flexbox-верстка. Flexbox - это технология CSS, которая позволяет создавать гибкие и адаптивные макеты. Flexbox позволяет легко управлять расположением элементов на странице, что делает его идеальным выбором для создания адаптивного дизайна. Он также обеспечивает кроссбраузерность, поскольку он хорошо поддерживается всеми современными браузерами.

# Выбор стилевого оформления

Для реализации стилевого направления веб-сайта было выбрано современное и яркое стилевое направление, для привлечения внимания посетителей и создания условий для приятного использования сайта.

Сайт будет дополнен анимациями и интерактивными объектами, для привлечения внимания.

Основные элементы дизайна будут включать в себя большие и яркие изображения блюд, которые будут привлекать внимание посетителей сайта.

Цветовые решения можно отнести к ярким, но хорошо сочетающимся цветам. Основные цвета: #313D47, #ECAD2F, #FFFFFF. Цветовая палитра представлена на рисунке 2.1.

Рисунок 2.1 – Цветовая палитра сайта

Сочетание ярких и насыщенных цветов создаст контрастный и запоминающийся дизайн.

# 2.3. Выбор шрифтового оформления

В качестве основного шрифта в данном проекте был выбран шрифт Rubik.

Шрифт Rubik создан в 2015 году и разработан Джоном Рэйсом. Он имеет геометрический дизайн, который включает в себя округлые формы и резкие углы.

Шрифт Rubik является современным и универсальным шрифтом, который может использоваться для различных целей в веб-дизайне.

Данный шрифт обладает хорошей читабельностью, что делает его идеальным выбором для использования в веб-дизайне.

Rubik имеет современный и эстетичный дизайн, который подходит для различных проектов. Он отлично сочетается со многими современными стилями дизайна и может использоваться для создания привлекательных и современных веб-сайтов.

Rubik хорошо поддерживается всеми современными браузерами.

# 2.4. Разработка логотипа

Разработка логотипа осуществлялась в графическом редакторе Figma. Логотип представляет из себя svg изображение, состоящее из двух основных частей: название сервиса и небольшое изображение. Логотип представлен на рисунке 2.2.

Рисунок 2.2 – Логотип сервиса по доставке еды

Интересной деталью логотипа является сердце на бургере, будто бы выполненное кунжутными семечками.

Логотип отображает суть сервиса, выглядит лаконично, но достаточно ярко. Логотип подходит под цветовое оформление сайта и приятно дополняет шапку сайта.

# 2.5. Разработка пользовательских элементов

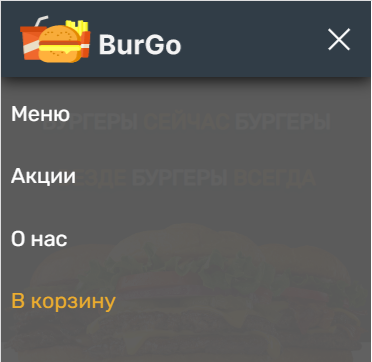
Разработка пользовательских элементов в дизайне веб-сайта — это процесс создания уникальных элементов веб-страницы, которые могут улучшить пользовательский опыт и сделать сайт более привлекательным и функциональным.

Пользовательские элементы веб-дизайна могут быть различными виджетами, формами, кнопками, иконками и другими элементами, которые помогают пользователям взаимодействовать со страницей и получать необходимую информацию. Они могут быть созданы для улучшения навигации, повышения вовлеченности пользователей и улучшения общего пользовательского опыта.

Одним из самых важных элементов страницы является навигационное меню, которое позволяет ориентироваться по сайту. Навигационное меню расположено в шапке сайта и представлено на рисунке 2.3.

Рисунок 2.3 – Шапка с панелью навигации по веб-сайту

Для улучшения пользовательского опыта навигационное меню адаптируется под размер экрана и сворачивается в бургер-меню (рисунок 2.4).

Рисунок 2.4 – Адаптация панели навигации по веб-сайту

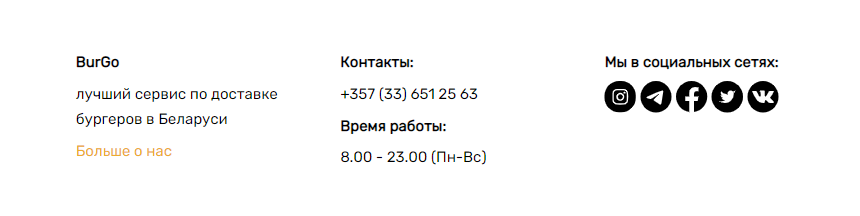
Не менее важным элементом является подвал, хранящий основную информацию (рисунок 2.5).

Рисунок 2.5 – Подвал веб-сайта с основной информацией

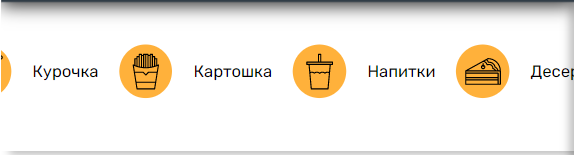
На главной странице сайта расположены блоки, содержащие некоторые элементы меню, призванные заинтересовать пользователя в продукте и вызвать желание ознакомиться с другими продуктами (рисунок 2.4).

Рисунок 2.6 – Блоки с некоторыми элементами меню

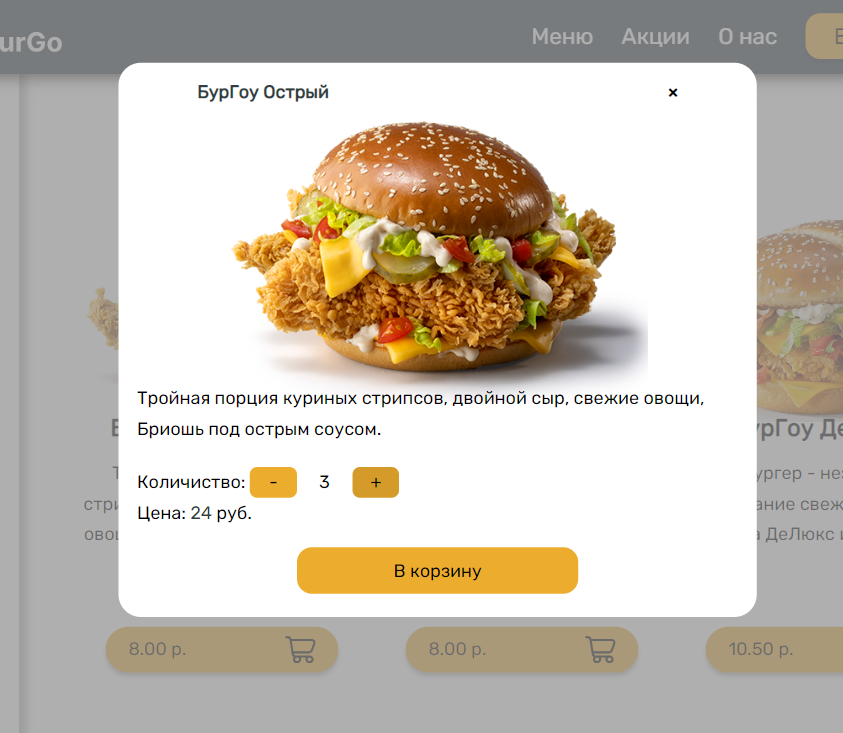
На странице «Меню» расположены элементы меню, выполненные блоками, содержащими краткую информацию о продукте, а также боковая навигационная панель (рисунок 2.7).

Рисунок 2.7 – Меню с боковой понелью

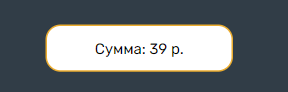
Такое расположение панели навигации позволяет с легкостью перемещаться по меню, без необходимости пролистывать страницу каждый раз. Панель навигации также является адаптивным элементом и меняет свое строение на устройствах с маленькими экранами (рисунок 2.8).

Рисунок 2.8 – Панель навигации по меню на мобильных устройствах.

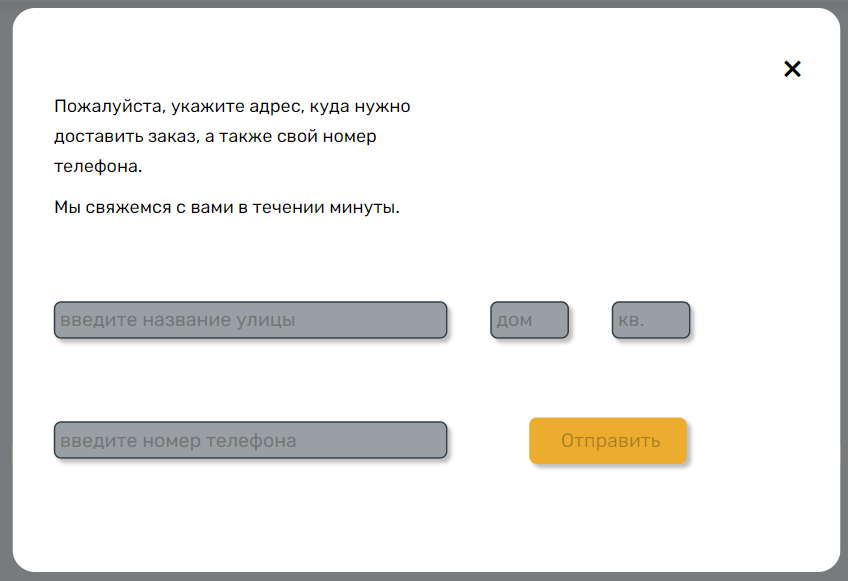
Возможность заказа продуктов реализована следующим образом: при нажатии на кнопку внизу определенного элемента меню, всплывает окно, на котором можно выбрать количество продуктов и отправить их в корзину (рисунок 2.9).

Рисунок 2.9 – Окно добавления продукта в меню

В корзине пользователь может редактировать количество желаемых продуктов и видеть общую сумму заказа (рисунок 2.10)

Рисунок 2.10 – Сумма заказа

После добавление продуктов в корзину и перехода туда, у пользователя появляется возможность сделать заказ. Это можно осуществить при помощи соответствующей кнопки, после нажатия на которую появляется форма, куда необходимо занести контактные данные и адрес пользователя (рисунок 2.11).

Рисунок 2.11 – Форма отправления заказа

Фавикон (Favicon) - это маленькая иконка, которая отображается во вкладке браузера рядом с названием сайта (рисунок 2.12). Фавикон может помочь узнать сайт среди других вкладок и закладок, а также улучшить общий пользовательский опыт.

Рисунок 2.12 – Фавикон веб-сайта

# 2.6 Разработка спецэффектов

Анимации в дизайне играют крайне важную роль. Анимации могут помочь сделать пользовательский интерфейс более привлекательным и понятным. Например, при наведении курсора на кнопку, анимация может показать, что кнопка активна, и что будет происходить при ее нажатии.

Основные виды анимации на веб-сайте: изменение цвета текста, изменения фона, положения и видимости элемента при наведении на него курсора мыши.

Выполнена анимация изменения цвета текста при наведении курсора в шапке веб-сайта (рисунок 2.13).

Рисунок 2.13 – Анимация изменения цвета текста

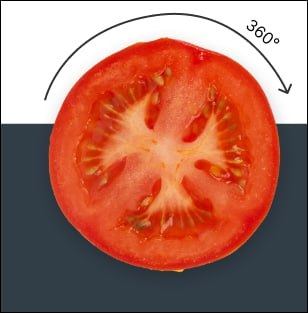
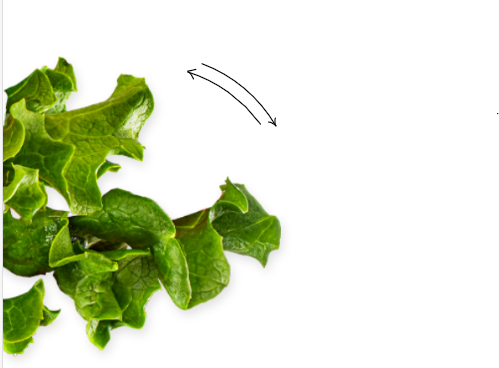
На главной странице размещено множество анимированных элементов, чтобы привлечь внимание пользователя. Например, изображение помидора выполняет полный оборот на 360 градусов при наведении курсора мыши (рисунок 2.14).

Рисунок 2.14 – Анимация вращения изображения

Реализована анимация изменения положения элемента при наведении курсора мыши (рисунок 2.15).

Рисунок 2.15 – Анимация движения изображения

Подобные анимация привлекают внимание пользователя, вызывают положительный отклик и запоминаются пользователю, что благоприятно влияет на общее впечатление от веб-сайта.

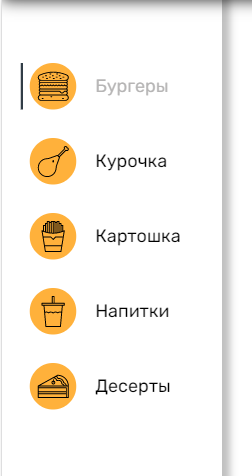
При наведении в навигационной панели меню появляется левая граница, а цвет текста меняется, что позволяет пользователю легче ориентироваться в панели (рисунок 2.16).

Рисунок 2.16 – Анимация навигационной панели меню

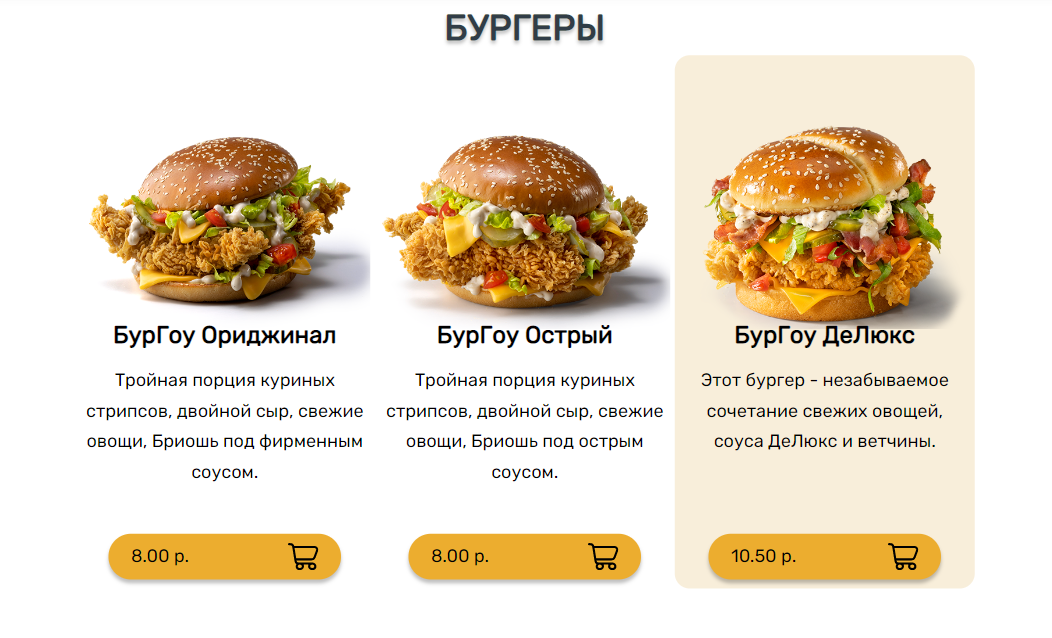
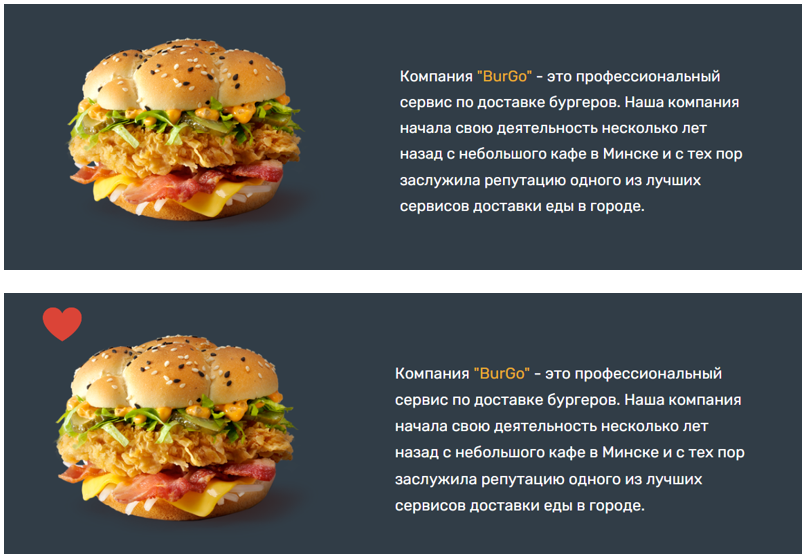
Чтобы отслеживать выбранный продукт была добавлена анимация изменения фона блока при наведении мыши (рисунок 2.17). Была также добавлена анимация изменения цвета фона кнопки при клике мыши.

Рисунок 2.17 – Анимация изменения цвета фона

Помимо практической функции выделения информации, данная анимация также выполняет эстетическую функцию: заполняет белое однообразное пространство страницы цветом в палитре сайта.

На странице «О нас» были добавлены анимации появления изображений при наведении курсора мыши на абзац текста или другое изображение (рисунок 2.18).

Рисунок 2.18 – Появление элемента при наведении мыши

# 2.7. Выводы

Проектирование страниц веб-сайта является важным и необходимым этапом разработки. Использование инструментов для создания макетов, таких как Figma, значительно облегчает процесс проектирования и позволяет получить более качественный результат.

Для разработки страниц веб-сайта необходимо учитывать такие факторы, как пользовательские интерфейсы, навигация, контент, пользовательский опыт и пользовательские элементы. Они влияют на удобство использования сайта для посетителей и повышают его интерактивность и привлекательность.

Особое внимание следует уделять использованию анимаций на сайте. Анимации могут улучшить взаимодействие пользователя с сайтом и дать ему дополнительную информацию.

# 3. Реализация структуры веб-сайта

# 3.1. Структура HTML-документа

Структура HTML-документа - это организация содержания документа в специальном формате, который определяет его содержание и взаимодействие с другими элементами страницы. Он состоит из различных элементов, которые определяют структуру контента HTML-документа и его отображение в браузере.

HTML-документ должен быть организован по определенной структуре, которая состоит из нескольких основных элементов:

1. Элемент doctype: указывает на использование конкретной версии HTML.
2. Элемент HTML: корневой элемент, который всегда присутствует в HTML-документе.
3. Секция head: содержит информацию метаданных, такую как заголовки страницы, ключевые слова, метаописания и т.д.
4. Секция body: содержит непосредственно контент страницы, который отображается в браузере.

В HTML5 появились новые элементы, которые могут использоваться для организации контента на странице. Например, наряду с классическими элементами, такими как заголовки, параграфы и списки, HTML5 также включает новые элементы, такие как header, main, footer, nav и article, которые помогают организовать страницу в соответствии с современными требованиями.

В теге <header> располагается шапка веб-сайта (листинг 3.1).

|  |
| --- |
| <header class="header">      <div class="container">          <div class="header\_body">              <a href="./index.html" class="header\_logo">…</a>              <div class="header\_burger">…</div>              <nav class="header\_menu">                  <ul class="header\_list">                      <li class="header\_link"><a class="header\_link" href="./menu.html">Меню</a></li>                      <li class="header\_link"><a class="header\_link" href="./deals.html">Акции</a></li>                      <li class="header\_link"><a class="header\_link" href="./about.html">О нас</a></li>                      <li><a class="header\_button" href="./cart.html">В корзину</a></li>                  </ul>              </nav>          </div>      </div>  </header> |

Листинг 3.1 – Структура шапки

В шапке находится логотип в формате svg и навигационное меню, размещенное в теге навигации <nav>. Само навигационное меню представлено списком <ul>, в котором элементы списка <li> содержат ссылки на другие страницы.

Контент на всех страница обернут в тег <div> с классом container, такая структура необходима для контроля максимальной ширины страницы и отступов. В листинге 3.2 отображена структура главной страницы с использованием блока-контейнера.

|  |
| --- |
| <body>      <header id="myheader">…</header>      <div class="container">          <section class="banner">…          </section>          <section class="short-menu">…          </section>          <section class="short-menu2">…          </section>      </div>      <footer id="myfooter">…</footer>  </body> |

Листинг 3.2 – Тело главной страницы.

На главной странице для разграничения контента использовался тег <section>. Тег <section> в HTML используется для группировки логически связанного контента и создания разделов страницы (блок новостей, контактная информация и т.д.).

Навигационное меню на странице «Меню» было выполнено при помощи тега <nav> (листинг 3.3).

|  |
| --- |
| <nav class="sidebar" id="sidebar">          <div class="sidebar-menu">              <div class="category"><img src="./img/menu/sidebar/burgers.svg"><a href="#burgers">Бургеры</a></div>              <div class="category"><img src="./img/menu/sidebar/chicken.svg"><a href="#chicken">Курочка</a></div>              <div class="category"><img src="./img/menu/sidebar/potato.svg"><a href="#potato">Картошка</a></div>              <div class="category"><img src="./img/menu/sidebar/drinks.svg"><a href="#drinks">Напитки</a></div>              <div class="category desert"><img src="./img/menu/sidebar/deserts.svg"><a href="#deserts">Десерты</a></div>          </div>      </nav> |

Листинг 3.3 – Навигационная панель на странице «Меню»

Размещение самих элементов меню осуществлялось при помощи тега <section> (листинг 3.4.). В этом теге находятся заголовки <h1> обозначающие категорию меню, а также блоки <div> с классом menu-items содержащие непосредственно продукты конкретной категории.

|  |
| --- |
| <section class="menu">      <h1 id="burgers">бургеры</h1>      <div class="menu-items">…      </div>      …  </section> |

Листинг 3.4 – Структура меню

Теги <div> с классом menu-items являются flex элементами, что позволяет контролировать и размещать элементы внутри них. В тегах с классом menu-items располагаются контейнеры <div> с классом item, служащие объединяющей оболочкой для одного элемента-продукта. Структура элемента <div> с классом item представлена в листинге 3.5.

|  |
| --- |
| <div class="item">                      <img src="./img/menu/burgers/4571.png" alt="">                      <h3>…</h3>                      <p class="discr">…</p>                      <div class="cart">                          <p>8.00 р.</p>                          <img src="./img/menu/cart.svg">                      </div>                  </div> |

Листинг 3.5 – Структура элемента меню

# 3.2 Добавление таблиц стилей SCSS и CSS

Таблицы стилей (CSS) используются для настройки внешнего вида элементов на странице, путем применения к ним стилей и размещения их в блоках CSS-кода. CSS-код определяет такие свойства, как цвет, шрифт, размер и расположение элементов. CSS-код может применяться к элементам на основе классов, идентификаторов, атрибутов или тегов.

В данной курсовой работе подключение CSS производилось использованием тега <link> внутри секции <head> HTML-страницы.

SCSS (Sass) код работает, как и CSS, но предоставляет дополнительные возможности для написания CSS-кода в более удобном и читаемом виде. SCSS позволяет использовать переменные, миксины, функции и вложенные правила.

В данной работе использовались такие преимущества SCSS, как вложенность тегов и задание переменных, которые существенно упрощают написание кода. Пример вложенности тегов и использования переменных в SCSS представлен в листинге 3.6.

|  |
| --- |
| p {      color: white;      font-size: 1.5rem;      span {          color: $yellow\_active;      }  } |

Листинг 3.6 – Пример использования SCSS

CSS-код определяет такие свойства, как цвет, шрифт, размер и расположение элементов. Но помимо этих свойств можно определять поведение элемента в определенных условиях. Например, при помощи псевдокласса :hover можно регулировать поведение и внешний вид элемента при наведении на него мыши (листинг 3.7).

|  |
| --- |
| .header\_button:hover {      background: $yellow\_inactive;  } |

Листинг 3.7 – Использование псевдокласса :hover

CSS также позволяет применять разные стили для разных устройств по типу отображения (например: для принтера, монитора или смартфона), а также для разных размеров экранов. Это можно осуществить при помощи медиа-запросов. Пример медиа-запроса продемонстрирован в листинге 3.8.

|  |
| --- |
| @media (max-width: 767px) {      section.banner {          height: 602px;      }      .banner-bg {          height: 602px+100px;          img {              height: 400px;          }      }  } |

Листинг 3.8 – Использование медиа-запросов

Помимо медиа- запросов в листинге 3.8 продемонстрированно использование такой возможности SCSS как использование арифметических выражений.

В данном курсовом проекте для выравнивания и управления элементаси активно использовалась flex-верстка — технология верстки веб-страниц с использованием CSS3-модуля flexbox. Она позволяет создавать гибкие и адаптивные макеты, которые легко адаптируются к разным размерам экранов и устройствам. Пример использования flexbox представлен в листинге 3.9.

|  |
| --- |
| .menu-items {      display: flex;      flex-wrap: wrap;      margin-bottom: 100px;      flex-direction: row;      justify-content: space-between;  }  .item {      flex: 0 0 33.33333%;      max-width: 33.33333%;  } |

Листинг 3.9 – Использование flexbox

В листинге 3.9 используются основные свойства флексбокса, а именно:

1. Свойство display: flex - этот свойство указывает, что элемент является гибким контейнером.
2. Свойство flex-direction - это свойство определяет направление основной оси флексбокса.
3. Свойство justify-content - это свойство управляет выравниванием элементов вдоль главной оси флексбокса.
4. Свойство flex-wrap - это свойство определяет, как должен переноситься контент.
5. Свойство flex (flex-grow, flex-shrink, flex-basis) - это свойство, которое управляет распределением свободного пространства внутри гибкого контейнера.

# 3.3. Использование стандартов XML (SVG)

SVG (Scalable Vector Graphics) - это формат графики, который используется для создания векторных изображений, которые могут масштабироваться без потери качества. Он основан на стандарте XML (eXtensible Markup Language) и может быть создан и отредактирован в любом текстовом редакторе.

Существует несколько способов подключения SVG к документу. Это можно сделать с помощью открывающего и закрывающего тега svg <svg> (приложение Д.1) или при помощи тега <img> (листинг 3.10).

|  |
| --- |
| <a href="./index.html" class="header\_logo">                  <img src="./img/LOGO.svg" alt="logo">              </a> |

Листинг 3.10 - Использование SVG-изображений

XML (eXtensible Markup Language) - это язык разметки документов, который используется для хранения и передачи данных в структурированном виде. XML позволяет создавать собственные теги и атрибуты, что делает его очень мощным и гибким расширением. Пример использования XML представлен в листинге 3.11.

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" ?>  <footer>      <div class="container">          <fbody class="footer\_body">              <about class="about\_us\_info">                  <h4>BurGo</h4>                  <p>                      лучший сервис по доставке                      бургеров в Беларуси                  </p>                  <a href="./about.html">                      Больше о нас                  </a>              </about>              <contacts class="contact\_info">                  <h4>Контакты:</h4>                  <p>+357 (33) 651 25 63</p>                  <h4>Время работы:</h4>                  <p>8.00 - 23.00 (Пн-Вс)</p>              </contacts>              <socials class="social\_media">                  <h4>Мы в социальных сетях:</h4>                  <links class="links">…                  </links>              </socials>          </fbody>      </div>  </footer> |

Листинг 3.11 – пример использования XML

# 3.4 Управление элементами DOM

DOM (Document Object Model) - это структура документа, которая представляет все элементы и содержимое веб-страницы в виде объектов, которые можно манипулировать с помощью JavaScript. Управление элементами DOM позволяет изменять содержимое и стиль элементов на странице, добавлять или удалять элементы, создавать анимации и многое другое.

Управление элементами DOM очень важно при разработке веб-сайтов и приложений, так как это позволяет создавать динамические и интерактивные пользовательские интерфейсы. Например, можно использовать управление элементами DOM для создания анимаций, которые показывают, как элементы на странице изменяются в ответ на действия пользователя. Пример анимации изменения масштаба изображения при наведении курсора представлен в листинге 3.12.

|  |
| --- |
| function scaleBurger() {      let burger = document.querySelector(".banner-bg").children[1];      burger.style.transform = "scale(1.01,1.01)";      setTimeout(function () {          burger.style.transform = "scale(0.99,0.99)";          setTimeout(function () {              burger.style.transform = "scale(1)";          }, 300);      }, 300);  }  let burger = document.querySelector(".banner-bg").children[1];  burger.addEventListener("mouseover", scaleBurger); |

Листинг 3.12 – Пример анимации масштабирования при помощи JavaScript

В данном примере поиск элемента происходит при помощи .querySelector(), который возвращает первый элемент с классом .banner-bg, а свойство children обеспечивает быстрый доступ к его дочерним элементам.

В данном примере используется метод .**addEventListener()**,который настраивает функцию, которая будет вызываться всякий раз, когда указанное событие доставляется целевому объекту.

При помощи JavaScript были также реализованы анимация вращения изображения при наведении (приложение Ж.1), добавление класса элементу при наведении курсора (приложение Ж.2).

Шапка и подвал были вставлены к HTML-документам извне при помощи JavaScript (листинг 3.13).

|  |
| --- |
| xhttp = new XMLHttpRequest();  xhttp.onreadystatechange = function () {      if (this.readyState == 4 && this.status == 200) {          document.getElementById("myfooter").innerHTML = this.responseText;      }  };  xhttp.open("GET", "../footer.xml", true);  xhttp.send(); |

Листинг 3.13 – Пример вставки элемента в HTML-документ.

Этот код отвечает за отправку HTTP-запроса на сервер для получения содержимого файла "footer.xml" и вставки его в элемент с ID "myfooter" на веб-странице.

Блок кода начинается с создания нового объекта XMLHttpRequest.

Далее устанавливается функция обратного вызова onreadystatechange, которая будет вызываться каждый раз, когда изменится состояние объекта XMLHttpRequest. В частности, код проверяет, что состояние равно 4 (данные получены и обработаны), а статус ответа равен 200 (успешный код ответа сервера).

Затем выполняется метод open для создания нового соединения с сервером для получения файла „footer.xml” методом GET. Последний аргумент метода open - значение truthy - означает использование асинхронного режима.

Наконец, вызывается метод send, который отправляет запрос на сервер и получает ответ. Если ответ содержит данные, то они устанавливаются содержимым элемента с ID myfooter веб-страницы с помощью свойства innerHTML.

На странице «Меню» навигация реализована при помощи бокового меню, чтобы не достижении конца страницы, навигационная панель оставалась на месте, а не заезжала под футер, была написана функция по расчету текущего положения на странице (листинг 3.14).

|  |
| --- |
| document.addEventListener("scroll", function () {      let sidebar = document.querySelector("#sidebar");      let footer = document.querySelector("#myfooter");      let footerHeight = Number(footer.offsetHeight);      let scrollTop = window.pageYOffset;      let fullHeight = Math.max(document.body.scrollHeight,          document.documentElement.scrollHeight,          document.body.offsetHeight, document.documentElement.offsetHeight, document.body.clientHeight,          document.documentElement.clientHeight);      let windowHeight = document.body.clientHeight;      let x = fullHeight - footerHeight,          y = scrollTop + windowHeight;        if (window.innerWidth > 887) {          if (x < y) {              sidebar.style.paddingBottom = `${footerHeight + 80}px`;          }          else {              sidebar.style.paddingBottom = `0px`;          }      }      else {          sidebar.style.paddingBottom = `0px`;      }  }); |

Листинг 3.14 – Изменение состояния боковой навигационной панели

При скроллинге экрана данная функция проверяет, дошел ли пользователь до подвала, и при ширине экрана больше 887 пикселей – брейк-поинт для перестроения боковой панели на верхнюю навигационную панель – данная функция добавляет боковой панели отступ, предотвращая ее исчезновение за подвалом.

Функционал передачи выбранных на странице «Меню» продуктов был реализован при помощи обращения к localStorage. Пример добавления товара в корзину представлен в листинге 3.15.

|  |
| --- |
| let makeOrder = document.querySelector('.into-cart');  makeOrder.addEventListener('click', function (event) {      let overlay = document.getElementById('overlay');      overlay.style.display = "none";      document.body.style.overflow = "auto";      let item = {};      item.src = document.getElementById("img").src;      item.name = document.getElementById("name").innerHTML;      item.amount = document.getElementById("amount").value;      item.cost = parseFloat(document.getElementById("cost\_span").children[0].children[0].innerHTML);      cart.push(item);      localStorage.setItem("cart", JSON.stringify(cart));      console.log(cart);      document.querySelector('#amount').value = 1;  }); |

Листинг 3.15 – Пример добавления продукта в корзину

В данном примере сначала описывается закрытие окна заказа overlay, потом происходит заполнение объекта item, который отвечает за конкретный товар. На следующем этапе текущий элемент item помещается в массив ранее добавленных в корзину элементов – cart. Функция завершается передачей массива со всеми товарами в локальное хранилище браузера, для последующего изъятия на странице корзины (листинг 3.16).

|  |
| --- |
| document.addEventListener('DOMContentLoaded', function (event) {      let data = [];      if (localStorage.getItem("cart") != null) {          data = JSON.parse(localStorage.getItem("cart"));      }  …  } |

Листинг 3.16 – Загрузка данных из локального хранилища

Все функции выполняются после загрузки страницы, так как после получения данных из хранилища они должны быть встроены в DOM. Это происходит при помощи добавления в существующий элемент-оболочку содержимого через свойство .innerHTML. Добавление и прорисовка элементов на странице «Корзина» представлена в листинге 3.17.

|  |
| --- |
| let div = document.getElementById("menu-items");      div.innerHTML = "";      data.forEach(item => {          div.innerHTML += `      <div class="item">          <img src="${item.src}" alt="">          <h3>${item.name}</h3>          <div class="cart">              <button id="minus">-</button>              <p><span>${item.amount}</span> шт. - <span>${item.cost}</span> р.</p>              <button id="plus">+</button>          </div>      </div>    `      }); |

Листинг 3.17 – Прорисовка элементов в корзине

Функции редактирования количества элементов в корзине представлены в приложении Ж.

# 3.5 Выводы

Для того, чтобы веб-сайт выглядел и функционировал так, как задумано, необходимо правильно организовать структуру HTML-документа, добавить таблицы стилей SCSS и CSS, использовать стандарты XML (SVG) и управлять элементами DOM.

На данном этапе была разработана правильная структура документа, написаны таблицы стилей SCSS и CSS. Были реализованы необходимые анимации, спецэффекты и функционал.

Веб-сайт был подготовлен к тестированию.

# 4. Тестирование веб-сайта

# 4.1. Адаптивный дизайн веб-сайта

Адаптивный дизайн сайта - это подход к разработке веб-сайтов, который позволяет создавать сайты, которые могут корректно отображаться на различных устройствах и экранах. Это важно, потому что сегодня большинство пользователей используют для просмотра веб-сайтов мобильные устройства, такие как смартфоны и планшеты.

Адаптивный дизайн сайта позволяет убедиться, что веб-сайт будет выглядеть хорошо на любом устройстве, не зависимо от размера экрана или ориентации устройства. Благодаря этому, пользователи могут легко найти нужную информацию и взаимодействовать с сайтом, независимо от устройства, на котором они находятся.

Для достижения адаптивности в данном курсовом проекте использовалась особая верстка: применялись брейк-поинты, использовались флексбоксы, а также относительные единицы измерения.

Для достижения качественной адаптивности веб-сайта была изменена структура навигационного меню в шапке (рисунок 4.1) и на странице «Меню» (рисунок 4.2).

Рисунок 4.1 – Адаптивная шапка

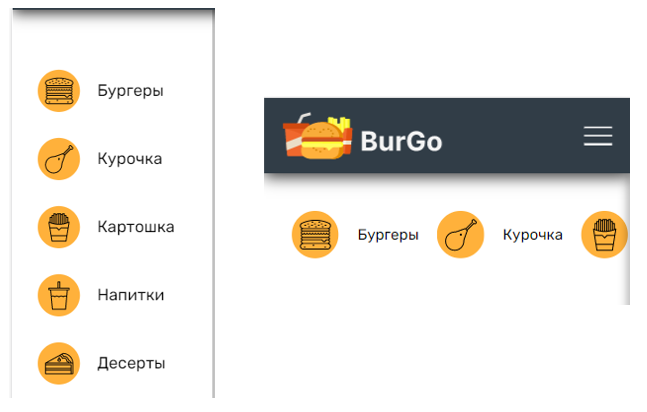


Рисунок 4.2 – Адаптивная навигационная панель страницы «Меню»

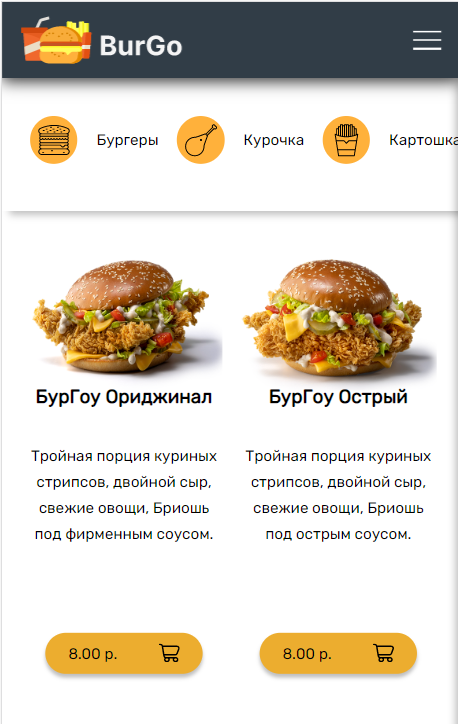
Помимо перестроения навигационных панелей сама страница также меняет свою структуру в зависимости от размера устройства. Например меняется отображение продуктов в меню (рисунок 4.3), чтобы пользователи на любых устройствах имели доступ к информации.

Рисунок 4.3 – Адаптивное меню

Пример не этого меню в версии для больших экранов на рисунке 2.7.

# 4.2. Кроссбраузерность веб-сайта

Каждый браузер имеет свои особенности и специфические функции, которые могут влиять на отображение и функциональность веб-страницы. Например, некоторые браузеры могут не поддерживать некоторые технологии, используемые на сайте, или могут отображать элементы страницы по-разному, что может привести к проблемам с отображением и функциональностью.

Чтобы обеспечить кроссбраузерность, разработчики используют современные технологии и методы, которые позволяют создавать сайты и приложения, которые работают одинаково хорошо на разных платформах и браузерах. Это включает в себя использование стандартов веб-разработки, тестирование на разных браузерах и платформах, а также использование современных методов и технологий, таких как CSS-фреймворки и JavaScript-библиотеки, которые позволяют создавать кроссбраузерные интерфейсы и улучшать пользовательский опыт.

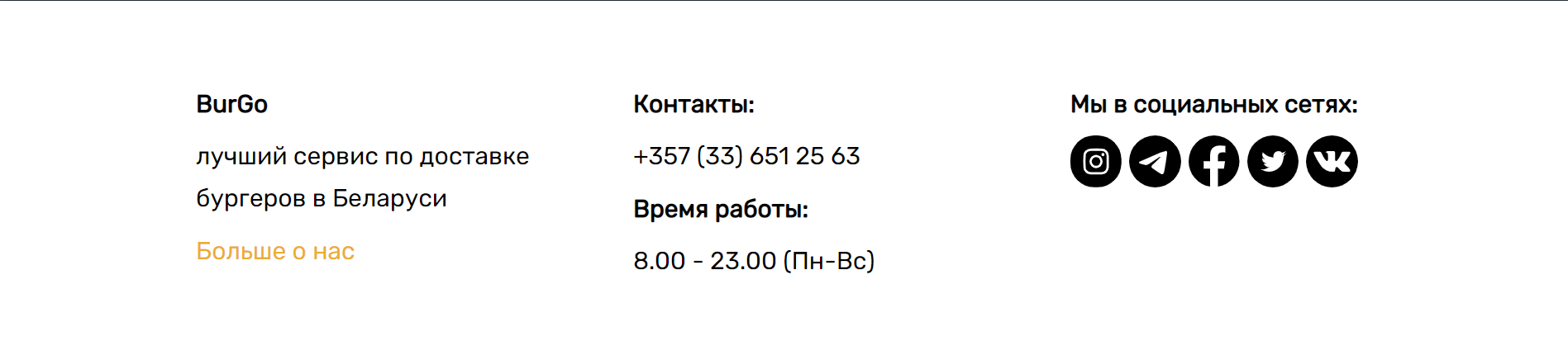
Проверка на кроссбраузерность показала, что веб-сайт одинаково хорошо функционирует и выглядит на любых платформах.

# Руководство пользователя

При входе на сайт пользователя встречает шапка с навигационным меню и логотипом, см. рисунок 1.13. Пользователь может перейти на любую страницу, представленную ссылкой в шапке.

При скроллинге главной страницу пользователь ознакомится с кратким перечнем продуктов из меню, см. рисунок 2.5.

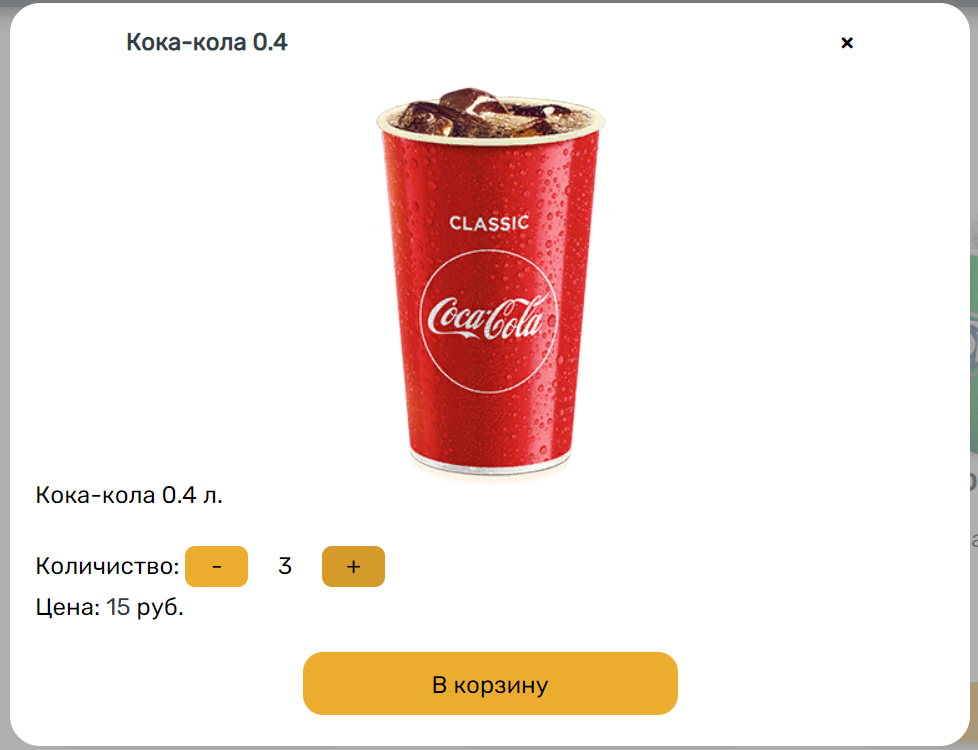
Страница заканчивается подвалом, с основной информацией (рисунок 4.4).

Рисунок 4.4 – Подвал веб-сайта

Через подвал пользователь может перейти на страницу «Больше о нас» и ознакомится с информацией о компании. Об активности кнопки говорит ее выделение цветом, а также наличие анимации смены цвета при наведении мыши.

На страницу «Меню» пользователь имеет возможность передвигаться при помощи навигации, см рисунок 2.16.

При нажатии на товар открывается окно, где пользователь может изменить количество желаемых товаров, еще раз ознакомиться с описанием и добавить продукт в корзину (рисунок 4.5).

Рисунок 4.5 – Форма добавления в корзину

Количество товаров регулируется кнопками «+» и «-», цена меняется автоматически.

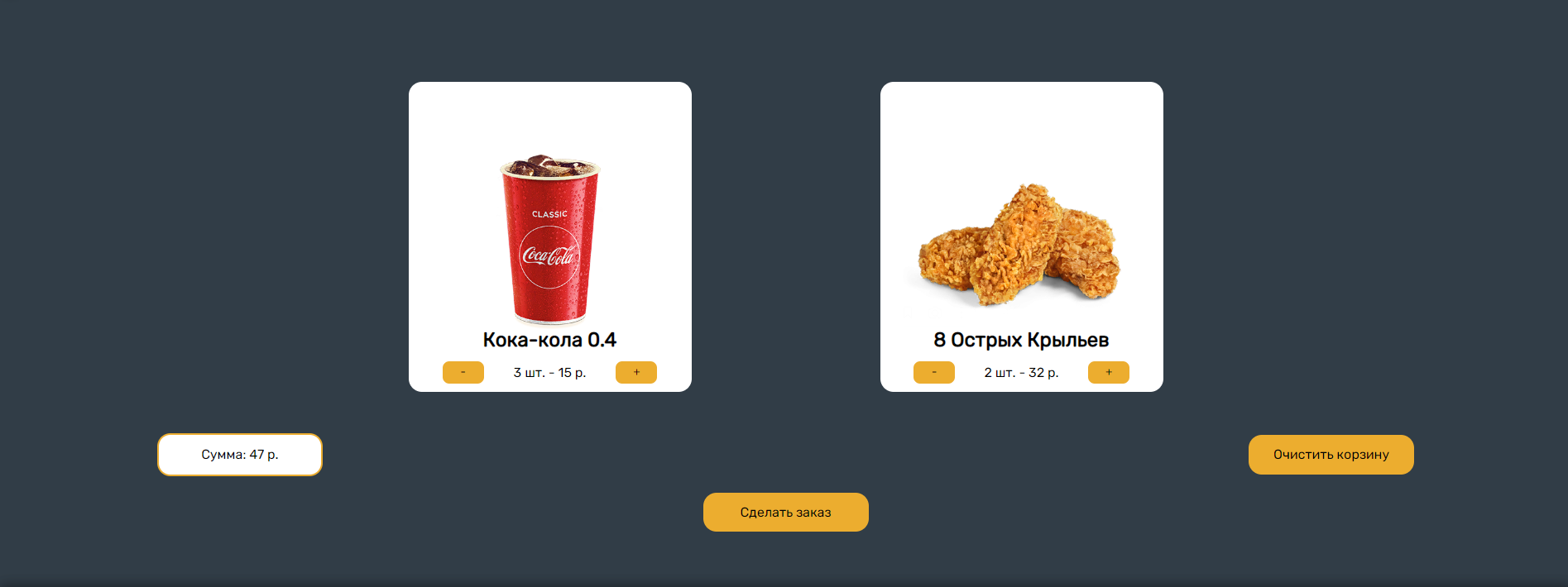
После нажатия кнопки «В корзину», продукт перемещается на соответствующую страницу. На странице «Корзина» пользователь может еще раз поменять количество продуктов при помощи соответствующих кнопок, очистить корзину полностью, ознакомиться с суммарной стоимостью или сделать заказ (рисунок 4.6).

Рисунок 4.6 – Страница «Корзина».

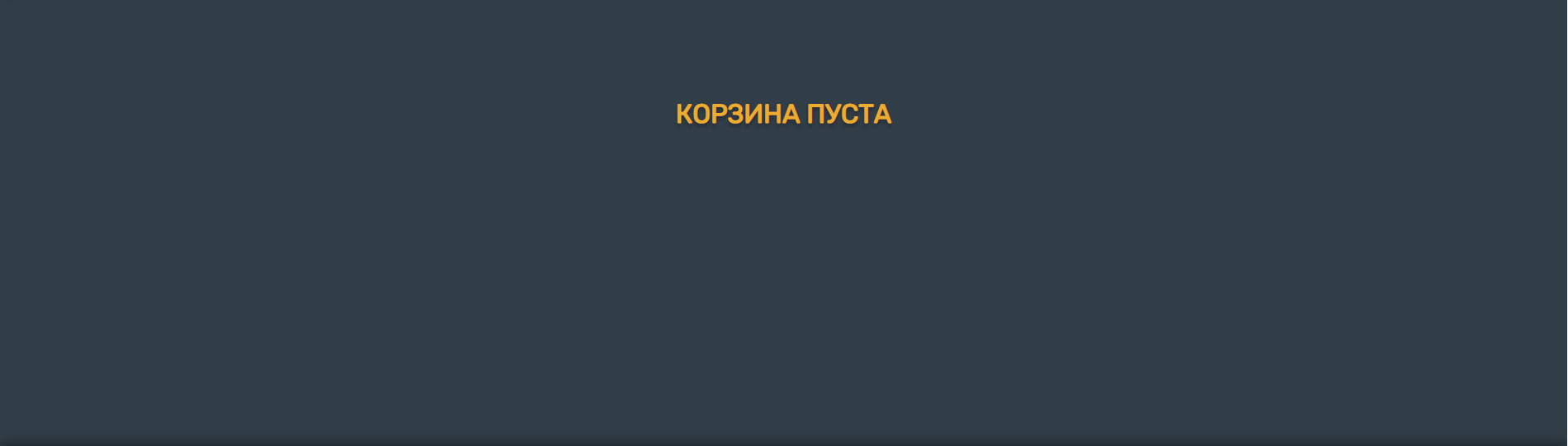
После нажатия кнопки «Сделать заказ» появится форма, рисунок 2.11. После ее заполнения и отправления корзина опять станет пустой (рисунок 4.7).

Рисунок 4.7 – Пустая корзина

# 4.4. Выводы

Тестирование веб-сайта на разных устройствах и браузерах необходимо, чтобы обеспечить его корректное отображение и функционирование.

Адаптивный дизайн и кроссбраузерность являются важными аспектами веб-разработки, которые должны учитываться при создании веб-сайта.

Руководство пользователя является не менее важным элементом, которое помогает пользователям быстрее и эффективнее взаимодействовать с сайтом.

Заключение

В ходе курсового проекта были достигнуты все поставленные цели.

В первом этапе была разработана визуальная концепция и дизайн-макеты сайта. Работа осуществлялась в среде Figma, которая предлагает множественные инструменты для создания дизайнов. Были созданы прототипы и макеты главной страницы, страницы меню, страницы корзины, страницы акций и страницы с информацией, а также формы заказа продукта. Макеты и прототипы были созданы для ПК-версии, планшетной версий, а также для мобильной версии сайта.

Затем была выполнена верстка и адаптивная верстка сайта.

Использовались HTML - язык разметки для создания веб-страниц, CSS - язык таблиц стилей, который используется для оформления веб-страниц, а также SASS - препроцессор CSS, XML - язык разметки, который используется для хранения и передачи данных. Были использованы современные методы и стандарты верстки, например, использование flexbox. Использование flexbox помогло разместить и организовать многочисленные элементы меню, а также сделало сайт более простым к адаптации под устройства меньших размеров.

Адаптация сайта заключалась в использовании «резиновых» блоков, флексбоксов, относительных единиц измерения, а также медиа-запросов. Медиа- запросы помогают не только менять какие-то незначительные характеристики, но также полностью структуру документа. Например, медиа запросы использовались для преобразования боковой навигационной панели на странице «Меню» в верхнюю навигационную панель, а также для реализации меню-бургер.

Функциональность сайта была разработана при помощи JavaScript. Были добавлены функции добавления и удаления товаров в корзину, оформления заказа, редактирование количества продуктов.

Кроме того, были добавлены анимации для веб-сайта. Анимации были добавлены для улучшения пользовательского опыта, привлечения внимания к важным элементам сайта и улучшения его визуального восприятия. Анимации добавлялись как при помощи CSS псевдоклассов, так и при помощи JavaScript.

Далее было проведено тестирование сайта. Были проведены тесты сайта на разных браузерах и платформах, что помогло обнаружить проблемы в отображении и функциональности сайта на разных устройствах и браузерах.

В ходе работы были использованы новые методы и функции, не встречавшиеся ранее в курсе. Например, была освоена работа с локальным хранилищем в браузере localStorage, был использован XMLHttpRequest запрос, была добавлена иконка favicon на страницы сайта. Были освоены новые свойства и методы для построения адаптивности веб-сайта, а также новые методы работы с сайтом через JavaScript.

В результате проделанной работы был создан качественный веб-сайт с адаптивным дизайном, кроссбраузерностью и функциональностью, который удовлетворяет потребности пользователей и соответствует современным требованиям веб-разработки.