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 «Скриптовые языки программирования»

Тема «Интернет магазин быстрого питания»

**Исполнитель**

студент(ка) 2 курса 7 группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ероховец И. А.

подпись, дата

**Руководитель**

Доцент каф. ИСиТ\_\_\_\_\_\_\_\_\_\_ Жиляк Н. А.

должность, ученая степень, ученое звание подпись, дата

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

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

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

дата, подпись

Курсовой проект защищен с оценкой

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Жиляк Н. А.

подпись дата инициалы и фамилия

Оглавление

[Введение 5](#_Toc121881687)

[1 Постановка задачи 6](#_Toc121881688)

[2 Проектирование веб-сайта 7](#_Toc121881689)

[2.1 Обзор аналогичных решений 7](#_Toc121881690)

[2.2 Выбор средств реализации программного продукта 9](#_Toc121881691)

[2.3 Проектирование страниц 11](#_Toc121881692)

[2.4 Роли пользователей 11](#_Toc121881693)

[2.5 Выбор стилевого оформления 11](#_Toc121881694)

[2.6 Разработка пользовательских элементов 12](#_Toc121881695)

[2.7 Выводы 13](#_Toc121881696)

[3 Реализация структуры веб-сайта 14](#_Toc121881697)

[3.1 Структура JSX-документа 14](#_Toc121881698)

[3.2 Структура Redux-Toolkit 16](#_Toc121881699)

[3.3 Добавление стилей SCSS и CSS 18](#_Toc121881700)

[3.4 Использование SVG 23](#_Toc121881701)

[3.5 Выводы 24](#_Toc121881702)

[4 Тестирование веб-сайта 25](#_Toc121881703)

[4.1 Адаптивный дизайн веб-сайта 25](#_Toc121881704)

[4.2 Кроссбраузерность веб-сайта 26](#_Toc121881705)

[4.3 Вывод 27](#_Toc121881706)

[Заключение 28](#_Toc121881707)

[Список использованных источников 29](#_Toc121881708)

# Введение

Интернет представляет собой одну из наиболее динамично развивающихся средств, открывающую широкие возможности перед творческими людьми. Смена стилей происходит настолько быстро, насколько это возможно себе представить, при этом происходит постоянное улучшение предыдущих стилей, а также появляются новые методики разработки. Всё это ведёт к созданию бесчисленного количества разнообразных веб-сайтов.

Современное общество сегодня справедливо можно назвать информационным. Люди привыкли получать значительную часть информации из интернета: просматривать новостные сайты, переписываться в социальных сетях, получать новые знания. Интернет всё больше интегрируется в нашу жизнь. Также немаловажной особенностью современного человека, является привычка людей использовать большое количество различных услуг, конкуренция среди которых очень велика. Со временем фирмы, предоставляющие услуги, осознали значимость интернета и необходимость создания собственных сайтов, так как в противном случае люди, которые привыкли искать информацию из глобальной сети, могут никогда не узнать о вашем товаре. Создание своего веб-сайта стало важно для любого бизнеса, в том числе и для крупных фирм, о которых и так все всё знают.

Актуальность темы состоит в том, что собственный сайт стал неотъемлемым атрибутом любой организации. Сегодня в создании сайта принимают участие не только программисты, но и дизайнеры, которые принимают тонкости подачи информации и потребления её посетителями сайта. Дизайн сайта играет большую роль в процессе его продвижения и раскрутки. Основная задача дизайнера, это правильное оформление главной странички, т.к в самом начале пользователь видит её и сразу же складывается первое впечатление от сайте. Дизайн сайта и его качественное оформление – это показатель имиджа, т.к не каждый предприниматель готов отдать крупную сумму за найм хорошего программиста и дизайнера. Дизайн сайта должен разрабатывать с учётом тематики и целей функционирования веб-ресурса. Все вышеперечисленные факты отражают актуальность тематики курсовой работы.

Для разработки будут использоваться такие технологии как Redux, TypeScript, React, HTML/CSS/SASS.

# 1 Постановка задачи

Основные задачи для интернет-магазина:

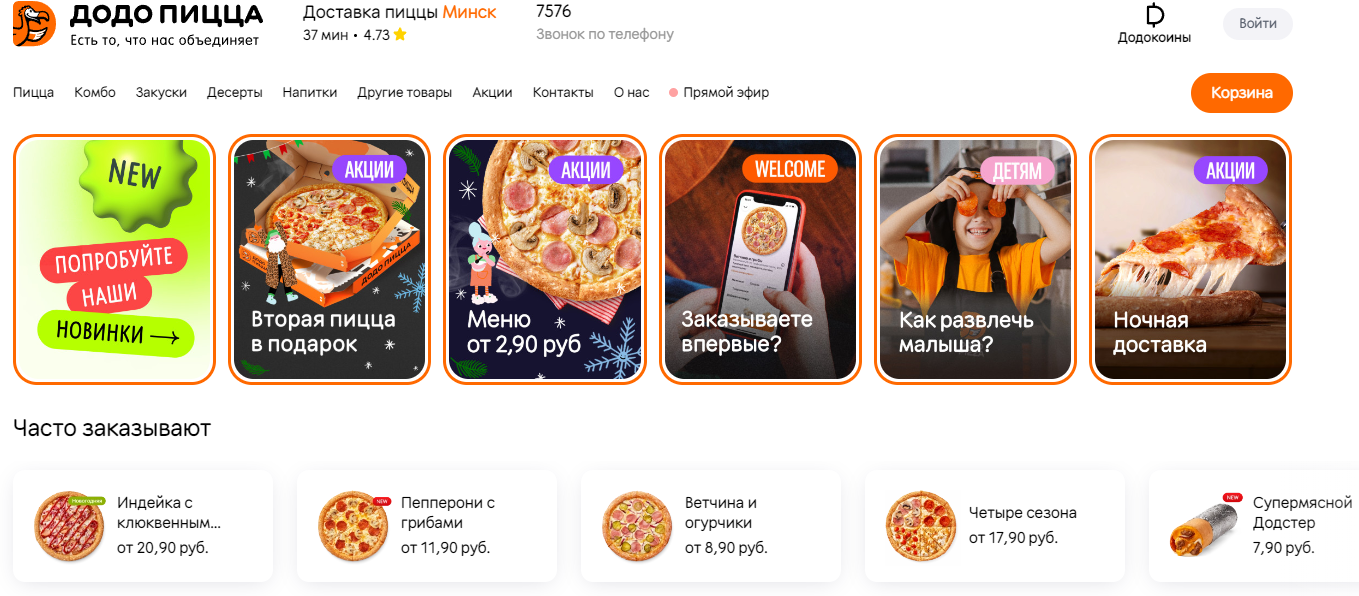
* Разработать возможность добавления и удаления товаров из корзины;
* Разработать структуру веб-сайта;
* Разработать возможность различных сортировок товара;
* Разработать систему глобального поиска товара;

# 2 Проектирование веб-сайта

## 2.1 Обзор аналогичных решений

Курсовой проект представляет из себя веб-сайт для магазина быстрого питания. В такой сфере уже существуют некоторые аналогичные решения. В этом разделе будут приведены веб-сайты существующих онлайн-магазинов по продаже еды быстрого питания.

Для начала введём критерии, по которым будет оценивать сайты, в первую очередь это удобства, на втором месте располагается дизайн и цветовая палитра, ну и не стоит забывать об адаптивном дизайне.

Dodopizza [1]. Dodopizza – один из популярных брендов фастфуда в Беларуси (было бы довольно глупо обойти его стороной). Сам бренд предоставляет свои услуги с 2011 года. На сайте мы можем увидеть достаточно понятный рядовому пользователю интерфейс. На сайте так-же адаптивный дизайн.

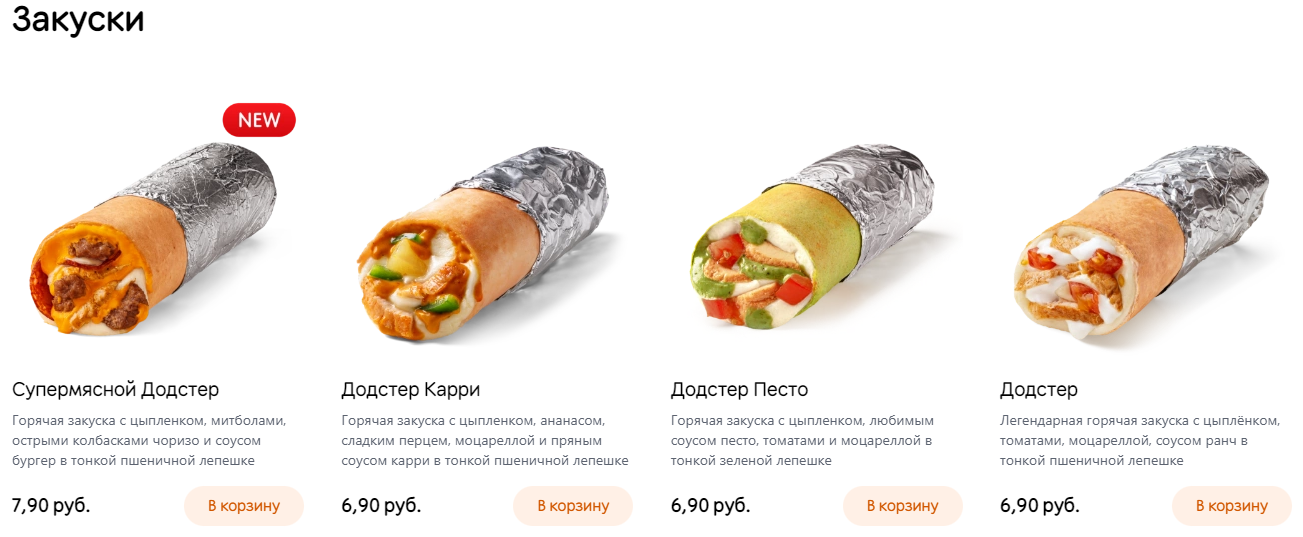
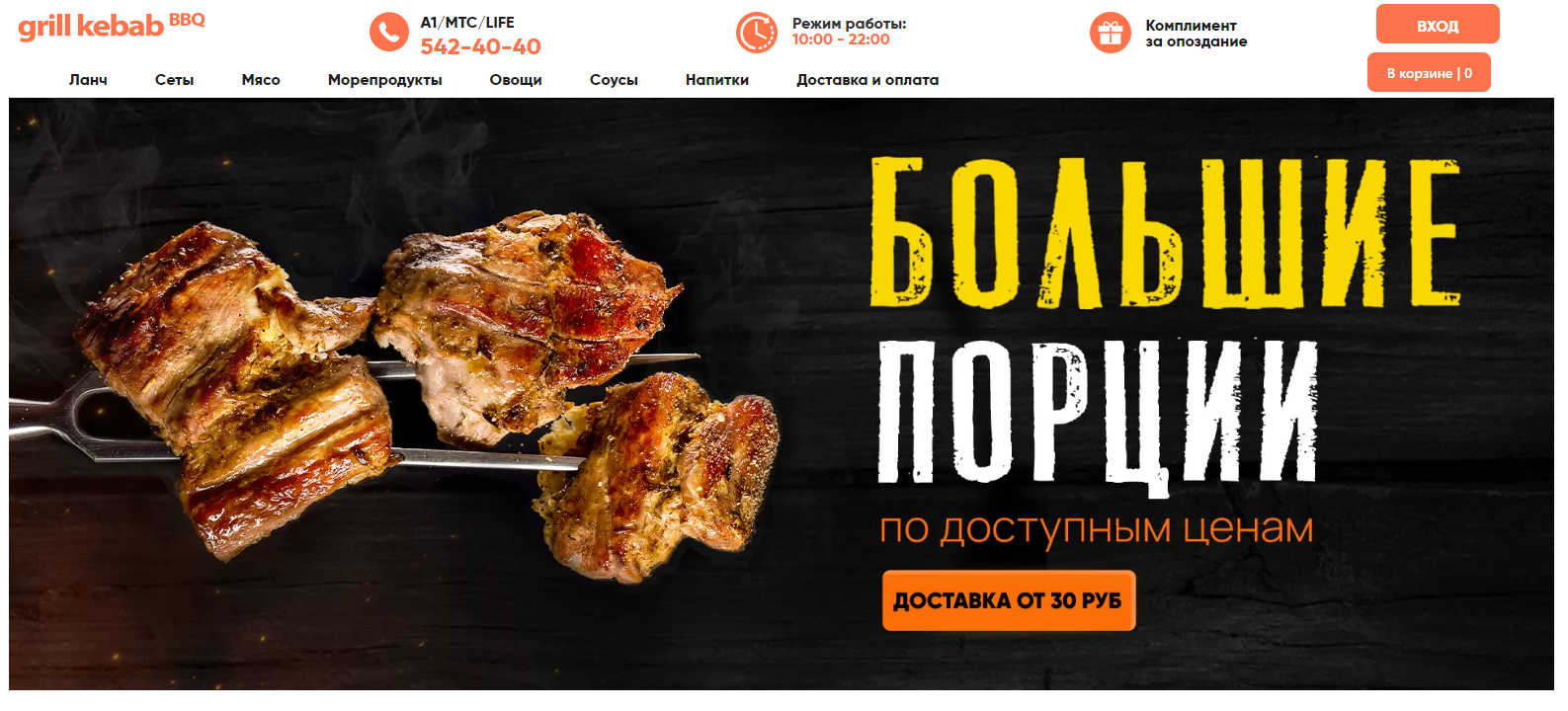
Рисунок 1.1 – Главная страница dodopizza.by

Рисунок 1.2 – Главная страница dodopizza.by

Список элементов, которые стоит перенять:

* Дизайн шапки
* Цветовую гамму сайта
* Дизайн товара

Gkb.by [2]. Grill kebab – так-же является одним из главных конкурентов в области магазинов быстрого питания. Однако меня немного пугает оформление главной страницы интернет магазина – она имеет очень много квадратных кнопок из это может только отпугивать возможных клиентов.

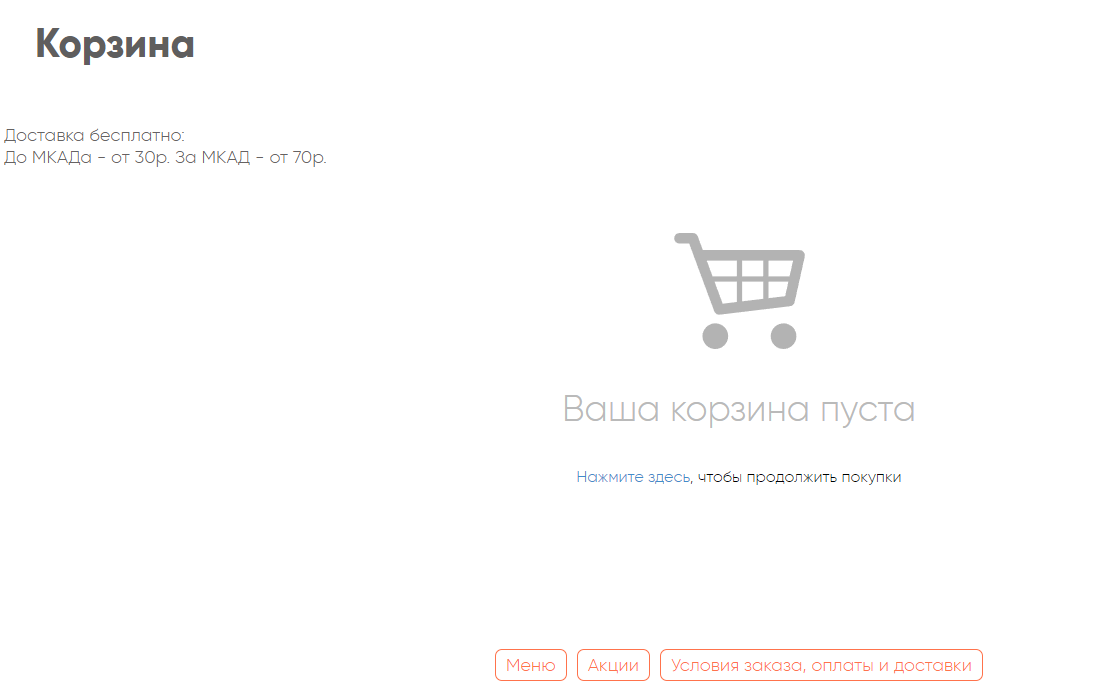
Рисунок 1.3 – Главная страница gkb,by

Рисунок 1.4 – Корзина gkb.by

Список элементов, которые стоит перенять:

* Дизайн корзины
* Разбиение товара на правильные подгруппы

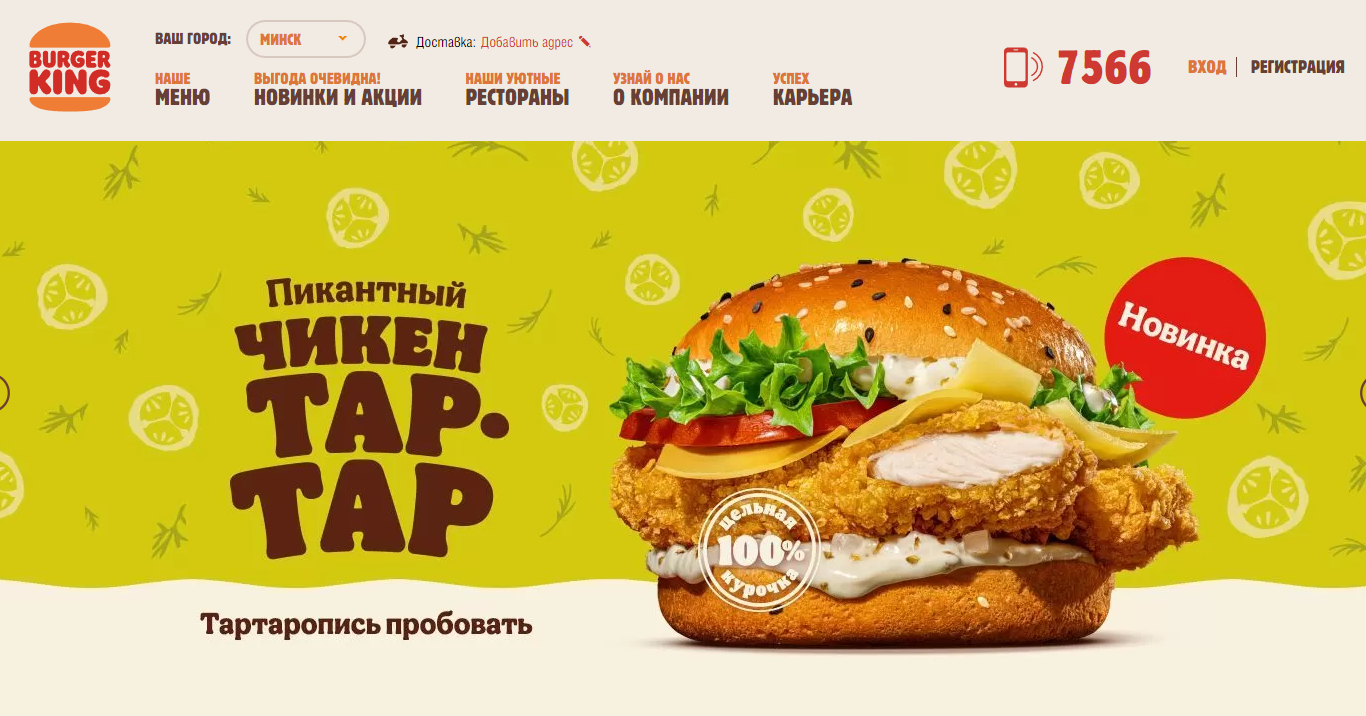
Burger-king.by [3]. Burger-king.by – является лидером продаж в сфере продуктов быстрого питания на территории Беларуси. Сам сайт имеет очень приятную цветовую гамму и простой интерфейс.

Рисунок 1.5 – Главная страница Burger-king.by

Список элементов, которые стоит перенять:

* Цветовую палитру
* Логотип

## 2.2 Выбор средств реализации программного продукта

Главной задачей сайта является предоставление информации о товарах, которые можно приобрести. Данный интернет-магазин будет предназначен для людей, которые хотят купить такие товары, как просто перекусить или полноценно покушать. Сайт будет адаптирован под мобильные устройства, а также будет кроссплатформенным.

Интернет-магазин будет разработан в текстовом редакторе Visual Studio Code. Страницы сайта могут просматриваться в таких браузерах как Microsoft Edge, Safari, Opera, Mozilla Firefox, Google Chrome.

Visual Studio Code — редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений.

Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса и средства для рефакторинга.

Для реализации будущего сайта будет использована следующая архитектура:

1.Front-end

2.Back-end

Front-end — это часть сайта, с которой взаимодействует пользователь. В данном случае это будет внешний вид сайта, его дизайн, а также навигация по сайту. В данном случае будет использованы следующие технологии: React JS, CSS3, Redux-Toolkit, Axios, React Router.

React JS — это библиотека, которая использует компонентный подход, который позволяет разрабатывать сложные интерфейсы из простых компонентов. С помощью React разработчики создают веб-приложения, которые изменяют отображение без перезагрузки страницы. Благодаря этому приложения быстро реагируют на действия пользователя, например, заполнение форм, применение фильтров, добавление товаров в корзину и так далее. React применяют для отрисовки компонентов пользовательского интерфейса. Также библиотека может полностью управлять фронтендом. В этом случае React используют с библиотеками для управления состоянием и роутинга, например, Redux и React Router.

Redux — это библиотека для управления состоянием приложения.

Redux Toolkit — это набор инструментов для упрощения работы с Redux.

Axios — это библиотека для работы с HTTP-запросами. Axios позволяет создавать запросы к API.

Back-end - это часть приложения, которая отвечает за обработку данных, ко-торые приходят от клиента и возвращают данные клиенту. Также Back-end отвечает за работу с базой данных, которая хранит данные приложения. Для создания Back-end будет использованы следующие технологии: Node.js, Express.

Node.js — это JavaScript-окружение, которое позволяет запускать JavaScript-код на сервере.

Express — это фреймворк для Node.js, который позволяет создавать сервер, который будет обрабатывать запросы от клиента и возвращать данные клиенту.

HTML (от английского HyperText Markup Language) — это код, который используется для структурирования и отображения веб-страницы и её контента. Например, контент может быть структурирован внутри множества параграфов, маркированных списков или с использованием изображений и таблиц данных. HTML используется, чтобы сообщать вашему браузеру, как отображать веб-страницы, которые вы посещаете. Браузер интерпретирует HTML-документ, выстраивая его структуру (DOM) и отображая ее в соответствии с инструкциями, включенными в этот файл (таблицы стилей, скрипты).

JavaScript – это интерпретируемый язык, то есть исходный код скриптов не нуждается в компиляции. Интерпретаторы этого языка автоматически управляют памятью и очищают неиспользуемые блоки, следовательно, программисту не нужно следить за выделением и очисткой памяти, как, например, в C++. JavaScript отличается от большинства объектно-ориентированных языков программирования тем, что в нём отсутствует понятие класса. Все объекты являются копиями уже существующего экземпляра объекта. Синтаксис JavaScript был вдохновлён языками C и Java. Все идентификаторы (названия переменных, функций и т.д.) являются регистрозависимыми. Операторы разделяются при помощи точки с запятой «;», однако использование этого символа необязательно: оператор автоматически завершается в конце строки.

## 2.3 Проектирование страниц

В проекте будет использована Grid верстка, так как она позволяет создавать адаптивные веб-страницы, которые будут отображаться корректно на всех устройствах. CSS Grid поддерживается практически всеми современными браузерами. С помощью этой технологии можно очень просто располагать наши товары.

Структура веб-сайта будет состоять из данных страниц:

* Главная страница
* Корзина товара
* Каталог товаров

На главной странице присутствует меню поиска по каталогу и сортировки товара по различным параметрам. Так же на ней расположен и весь каталог товаров, представленный карточками с изображением товара и основной информацией о нём. При нажатии на карточки товар отправляется в корзину в количестве 1шт. При повторном нажатии количество штук данного товара увеличивается на 1.

С главной страницы мы можем попасть на другие страницы с помощью нажатий на различные кнопки.

Следующая страница – корзина товаров, в которую пользователь добавил товары, которые он желает приобрести. Так же на странице в виде списка находятся товары, добавленные в корзину пользователем.

## 2.4 Роли пользователей

Как только пользователь заходит на веб-сайт, его ролью является Пользователь. Это означает, что он может просматривать, выбирать, заносить с корзину и покупать товары.

## 2.5 Выбор стилевого оформления

Для создания стилевого решения веб-сайта было принято решение о создании минималистического дизайна. Который будет подчёркивать наш стиль и давать некую изюминку, которая порадует покупателей.

На веб-сайте можно заменить не так много функционала, в этом и вся прелесть минимализма. Любому пользователю будет легче ориентироваться на нашем сайте, что повысит вероятность продажи товара.

Хочется отметить, что сайт не перегружен лишними элементами, а сами товары имеют лишь краткую информацию о таваре, дабы не загружать и не загромождать страницу.

Из цветовых решений было выбрано тёмно-белые тона с красными оттенками (Orange, White). Пример использования цветового решения предоставлен.

## 2.6 Разработка пользовательских элементов

**Разработка логотипа**

Логотип для веб-сайта был разработан в формате SVG. Для его создания был использован редактор векторных изображений Adobe Illustrator.

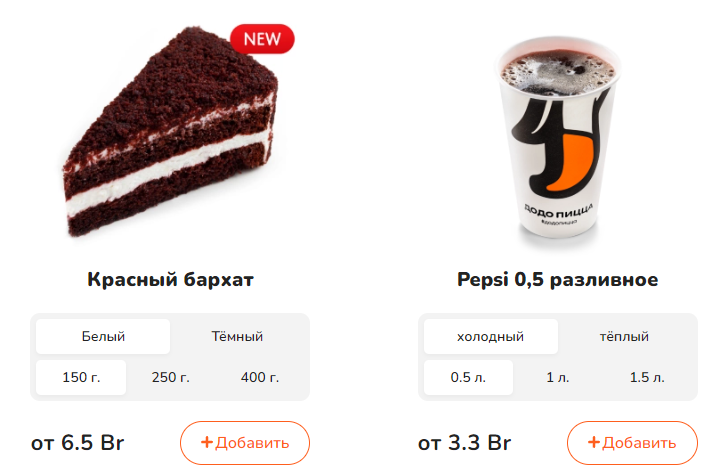
Логотип был разработан из концепта нашего товара. Т.к наш магазин ориентируется на продукты быстрого питания, было принято решение сделать кусочек пиццы, как логатип

Рисунок 1.6 – Логотип сайта Gril Bar

В верхнем колонтитуле сайта будут использоваться кнопки и пользовательские элементы по страницам, также для сортировки каталога и поиска по нему.

Рисунок 1.7 – Пользователь элемент сайта Gril Bar

В компоненте каталога будут реализованы карточки товаров

Рисунок 1.8 – Оформление товара на сайте Gril Bar

## 2.7 Выводы

Подводя итоги этого раздела можно сказать, что были разработаны макеты сайта, которые будут использоваться для дальнейшей разработки сайта. Были выбраны основные цвета и шрифты, которые будут использоваться на сайте. Были разработаны некоторые динамические эффекты и анимации, которые позволят придать сайту интерактивность и сделают его более привлекательным для пользователя. Самое главное, чтобы сайт был удобным для пользователя и отвечал его потребностям.

# 3 Реализация структуры веб-сайта

## 3.1 Структура JSX-документа

React 18 использует в основном функциональные компоненты для отрисовки веб-страницы. С помощью JSX эти компоненты принимают вид подобный структуре HTML-документа.

Например, структура компонента App приведена в листинге 1.

|  |
| --- |
| import Loadable from 'react-loadable';  import React, { Suspense } from 'react';  import { Routes, Route } from 'react-router-dom';  import Home from './pages/Home';  import './scss/app.scss';  import MainLayout from './layouts/MainLayout';  const Cart = Loadable({  loader: () => import(/\* webpackChunkName: "Cart" \*/ './pages/Cart'),  loading: () => <div>Идёт загрузка корзины...</div>,  });  const FullPizza = React.lazy(() => import(/\* webpackChunkName: "FullPizza" \*/ './pages/FullPizza'));  const NotFound = React.lazy(() => import(/\* webpackChunkName: "NotFound" \*/ './pages/NotFound'));  function App() {  return (  <Routes>  <Route path="/" element={<MainLayout />}>  <Route path="" element={<Home />} />  <Route  path="cart"  element={  <Suspense fallback={<div>Идёт загрузка корзины...</div>}>  <Cart />  </Suspense>  }  />  <Route  path="pizza/:id"  element={  <Suspense fallback={<div>Идёт загрузка...</div>}>  <FullPizza />  </Suspense>  }  />  <Route  path="\*"  element={  <Suspense fallback={<div>Идёт загрузка...</div>}>  <NotFound />  </Suspense>  }  />  </Route>  </Routes>  );  }  export default App; |

Рисунок 3.1 – Структура JSX документа

В нём использованы импортированные из react-router-dom компоненты BrowserRouter, Routes, Route, необходимые для навигации на веб-сайте

## 3.2 Структура Redux-Toolkit

Redux-Toolkit был использован для работы с состояниями и их изменением.

|  |
| --- |
| import { createSlice, PayloadAction } from '@reduxjs/toolkit';  import { calcTotalPrice } from '../../utils/calcTotalPrice';  import { getCartFromLS } from '../../utils/getCartFromLS';  import { CartItem, CartSliceState } from './types';  const initialState: CartSliceState = getCartFromLS();  const cartSlice = createSlice({  name: 'cart',  initialState,  reducers: {  addItem(state, action: PayloadAction<CartItem>) {  const findItem = state.items.find((obj) => obj.id === action.payload.id);  if (findItem) {  findItem.count++;  } else {  state.items.push({  ...action.payload,  count: 1,  });  }  state.totalPrice = calcTotalPrice(state.items);  },  minusItem(state, action: PayloadAction<string>) {  const findItem = state.items.find((obj) => obj.id === action.payload);  if (findItem) {  findItem.count--;  }  state.totalPrice = calcTotalPrice(state.items);    },  removeItem(state, action: PayloadAction<string>) {  state.items = state.items.filter((obj) => obj.id !== action.payload);  state.totalPrice = calcTotalPrice(state.items);  },  clearItems(state) {  state.items = [];  state.totalPrice = 0;  },  },  });  export const { addItem, removeItem, minusItem, clearItems } = cartSlice.actions;  export default cartSlice.reducer; |

Рисунок 3.2 – Структура Redux

## 3.3 Добавление стилей SCSS и CSS

Все стили были написаны на SCSS, который в свою очередь преобразовывался в CSS. В этом мне помогла программа Koala – бесплатный продукт, который преобразует файлы SCSS типа в файлы CSS типа. Всё, что она требует, так это указать папку, в которой хранятся стили SCSS, и папку, в которой будут храниться преобразованные CSS стили. Преобразовании стилей происходит автоматически при сохранении обновлённого файла SCSS.

|  |
| --- |
| import './variables';  @import './libs/normalize';  @import './components/all';  body {  background-color: $background;  }  .wrapper {  width: calc(100vw - 100px);  height: 100%;  background-color: #fff;  margin: 50px auto;  border-radius: 10px;  max-width: 1400px;  @media (max-width: 645px) {  width: calc(100vw - 30px);  margin-top: 15px;  }  }  .content {  padding: 40px 0;  @media (max-width: 600px) {  width: 200px;  }  &\_\_error-info {  margin: 100px auto;  width: 500px;  text-align: center;  h2 {  font-size: 32px;  font-weight: 700;  margin-bottom: 10px;  }  p {  font-size: 20px;  }  }  &\_\_title {  font-weight: 800;  font-size: 38px;  margin-top: 60px;  margin-bottom: 50px;  }  &\_\_items {  display: grid;  grid-template-columns: repeat(4, 1fr);  grid-template-rows: repeat(1, 1fr);  grid-column-gap: 11px;  @media (max-width: 1400px) {  grid-template-columns: repeat(3, 1fr);  }  @media (max-width: 1060px) {  grid-template-columns: repeat(2, 1fr);  }  @media (max-width: 730px) {  grid-template-columns: repeat(1, 1fr);  }  }  &\_\_top {  display: flex;  align-items: center;  justify-content: space-between;  @media (max-width: 1260px) {  .categories {  width: 100%;  overflow: auto;  ul {  width: 785px;  }  }  .sort {  padding-left: 20px;  b {  display: none;  }  }  }  @media (max-width: 760px) {  .categories {  ul {  li {  padding: 8px 20px;  }  }  }  } |

Листинг 3.3 – SCSS код App файла

Реализация API для товаров

|  |  |
| --- | --- |
| React.useEffect(() => {  async function fetchPizza() {  try {  const { data } = await axios.get('https://637d290316c1b892ebc8e9a3.mockapi.io/items2' + id);  setPizza(data);  } catch (error) {  alert('Ошибка при получении пиццы!');  navigate('/');  }  } | React.useEffect(() => {  async function fetchPizza() {  try {  const { data } = await axios.get('https://637d290316c1b892ebc8e9a3.mockapi.io/items2' + id);  setPizza(data);  } catch (error) {  alert('Ошибка при получении пиццы!');  navigate('/');  }  } |

Листинг 3.4 – API для товаров

## 3.4 Использование SVG

|  |
| --- |
| <path  d="M5.92001 3.84V5.76V8.64C5.92001 9.17016 5.49017 9.6 4.96001 9.6C4.42985 9.6 4.00001 9.17016 4.00001 8.64L4 5.76L4.00001 3.84V0.96C4.00001 0.42984 4.42985 0 4.96001 0C5.49017 0 5.92001 0.42984 5.92001 0.96V3.84Z"  fill="#EB5A1E"></path>  <path  d="M5.75998 5.92001L3.83998 5.92001L0.959977 5.92001C0.429817 5.92001 -2.29533e-05 5.49017 -2.29301e-05 4.96001C-2.2907e-05 4.42985 0.429817 4.00001 0.959977 4.00001L3.83998 4L5.75998 4.00001L8.63998 4.00001C9.17014 4.00001 9.59998 4.42985 9.59998 4.96001C9.59998 5.49017 9.17014 5.92001 8.63998 5.92001L5.75998 5.92001Z"  fill="#EB5A1E"></path> |

Листинг 3.5 – SVG код

## 3.5 Выводы

Из за использования React и его дополнительных пакетов мы можем реализовывать то, что было невозможно сделать используя только JavaScript, а именно динамические обновления товара и его оптимизация.

Проблема адаптирования в том, что качество картинок портится при изменении размера. Поэтому на сайте размещены логотипы компании и социальных сетей, написанных на SVG. SVG позволяет избежать нам этих проблем потому, что в его основе лежит векторная графика.

# 4 Тестирование веб-сайта

## 4.1 Адаптивный дизайн веб-сайта

В современных реалиях адаптивный дизайн очень важен, ибо почти каждый в мире имеет в своём кармане мобильный телефон с выходом в интернет. Сайты обязаны быть адаптивными, потому что это улучшает процесс использования продукта.

Мой веб-сайт по всем законам адаптирован ничуть не хуже, чем у конкурентов. Сайтом удобно пользоваться на таких девайсах, как планшеты и компьютеры. Для телефонов, как и оговаривалось ранее, весь дизайн выполнен в колоночной системе. Это значит, что каждый элемент следует вниз за другим, что даёт нам много место для самих этих элементов.

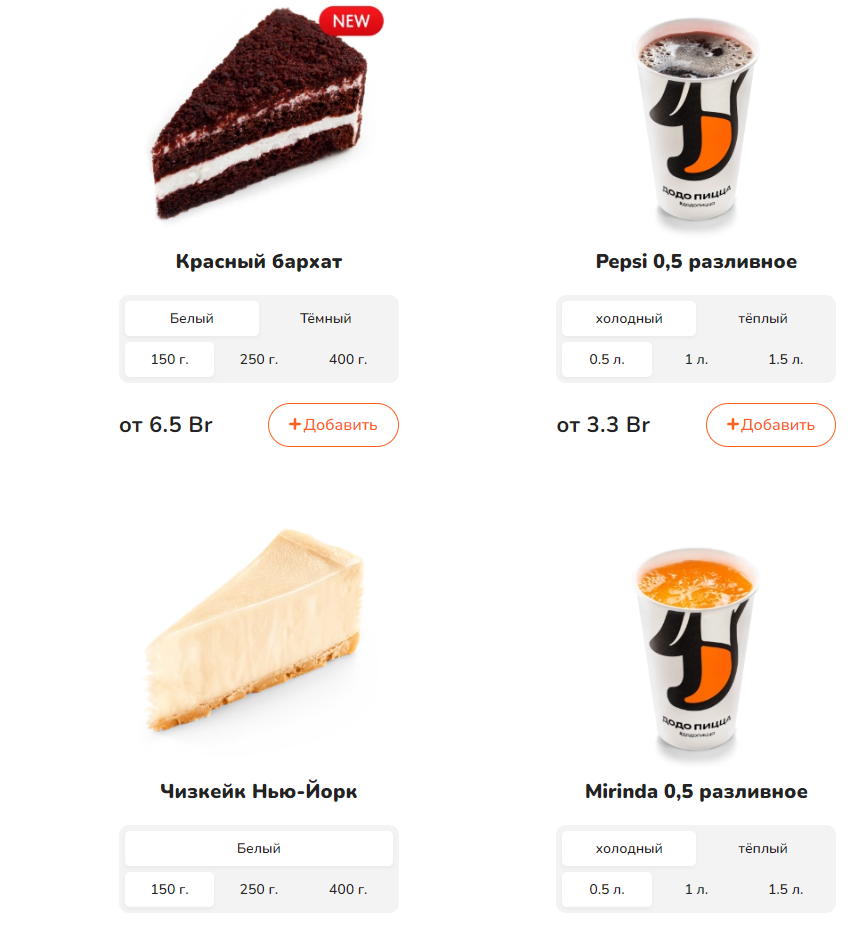
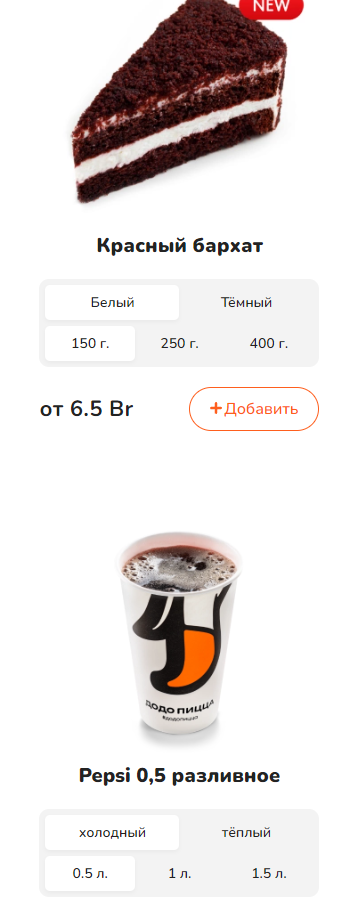
 Чтобы адаптировать мой веб-сайт, я использовал так называемые медиа-запросы в CSS. Понять их довольно несложно. Мы просто указываем минимальную и максимальную ширину расширения экрана и подстраиваем стили элементов под эти значения. Если значения расширения экрана выйдут за условные пределы медиа-запросов, то стили станут прежними, если мы не писали другие запросы.

Рисунок 4.1 – Адаптивность 1060x900

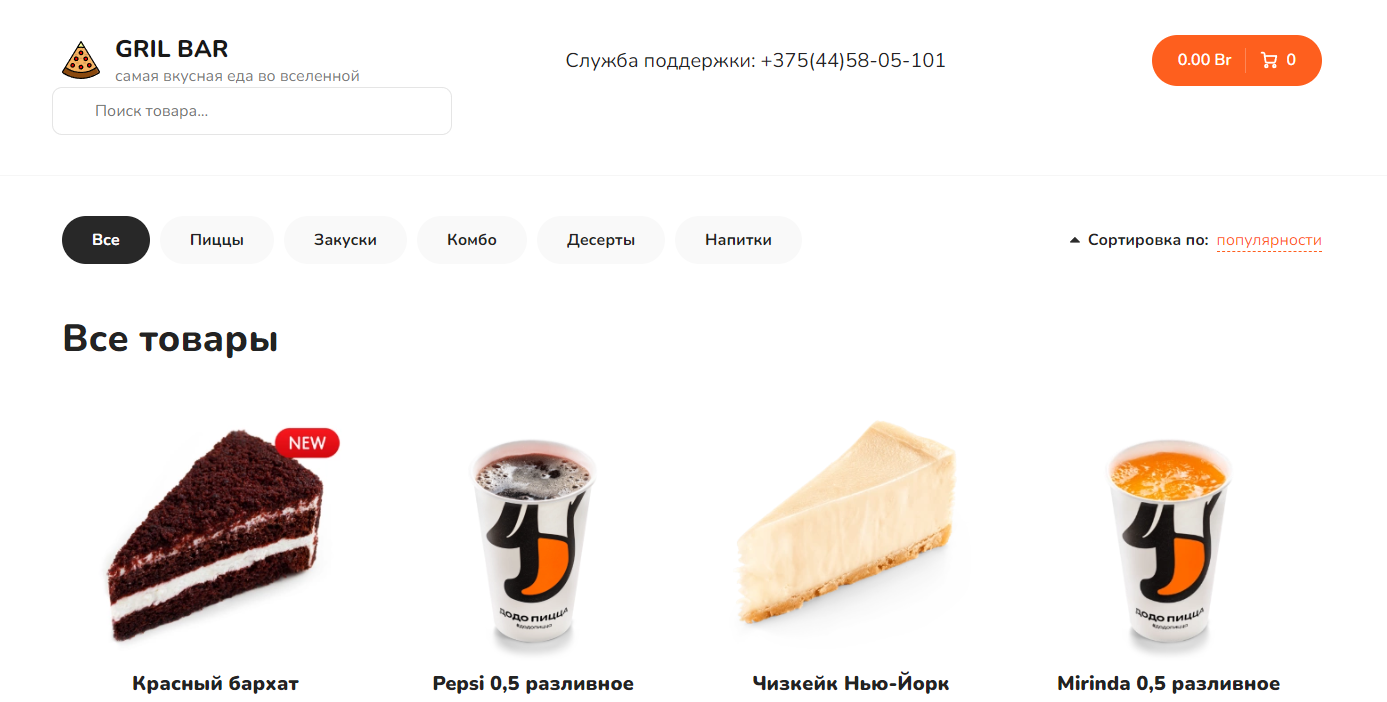
Рисунок 4.2 – Адаптивность 600x900

## 4.2 Кроссбраузерность веб-сайта

Кроссбраузерность, также, как и адаптивность, является одним из важнейших пунктов в веб-программировании. На сегодняшний день существует очень много популярных веб-браузеров, которые устроены по-своему и ведут себя по-другому при выводе контента.

Сайт адаптирован на такие популярные браузеры, как Mozilla Firefox, Google Chrome, Opera, Opera GX, Microsoft Edge. Плюсом для разработчиков будет то, что браузеры поддерживают очень много стилей. А значит проблем с выводом веб-сайта в других браузерах не будет.

Стандартное отображение в Mozilla Firefox, как всё и задумывалось:



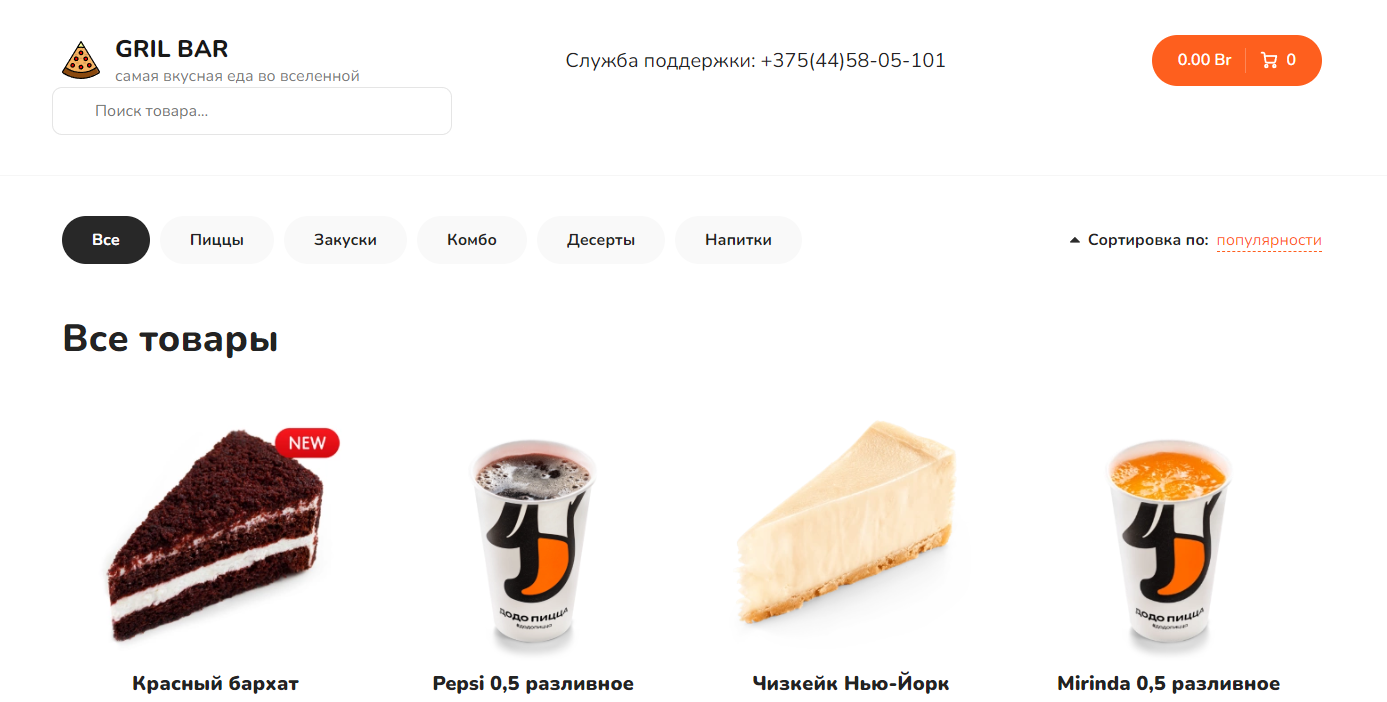
Рисунок 4.3 – Сайт в браузере Opera

Рисунок 4.4 – Сайт в браузере Google

## 4.3 Вывод

В ходе тестирования моего сайта в разных браузерах с разными значениями расширения экрана я пришёл к выводу, что мой сайт отображается схоже в современных браузерах, таких, как: Chrome, Firefox, Microsoft Edge, Opera, Safari, Brave и подобные. Из этого следует, что код для адаптивности и кроссбраузерности был написан правильно.

# Заключение

Целью на курсовое проектирование было создание интернет-магазина для продажи продуктов быстрого питания. Он был реализован с использованием различных пакетов и других технологий frontend программирования.

Структура веб-сайта будет состоять из данных страниц:

* Главная страница
* Корзина товара
* Каталог товаров

Средой для разработки был выбран Visual Studio Code, который послужил отличным решением для разработки веб-сайта. После разработки последовал этап тестирования, в ходе которого был разработан адаптив для веб-сайта. Результатом курсового проекта стал готовый веб-сайт, который не уступает своим аналогам.

Таким образом я считаю, что цель курсового проекта полностью достигнута и реализована.

# Список использованных источников

1. Burger-king.by – интернет-магазин продуктов быстрого питания [Электронный ресурс] / Сайт https://burger-king.by. ­– 2022. – Режим доступа: https://burger-king.by. – Дата доступа: 10.10.2022.
2. Dodopizza.by– интернет-магазин продуктов быстрого питания [Электронный ресурс] / Сайт Dodopizza.by. ­– 2022. – Режим доступа: https://dodopizza.by. – Дата доступа: 16.10.2022.
3. Gkb.by – интернет-магазин продуктов быстрого питания [Электронный ресурс] / Сайт Gkb.by. ­– 2022. – Режим доступа: https://gkb.by. – Дата доступа: 16.10.2022.
4. Документация React [Электронный ресурс] / Сайт ru.reactjs.or.. ­– 2022. – Режим доступа: ru.reactjs.or. – Дата доступа: 20.10.2022.
5. CSS GRID. Полный курс [Электронный ресурс] / Сайт youtube.com. ­– 2022. – Режим доступа: https://www.youtube.com/watch?v=XXlw7TUxRVY. – Дата доступа: 20.10.2022.