**ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ**

**Государственное бюджетное профессиональное образовательное учреждение города Москвы**

**«Политехнический колледж им. Н.Н. Годовикова»**

Допустить к защите  
Заместитель директора по профессиональному образованию

\_\_\_\_\_\_\_\_\_\_\_\_\_ И.В. Бойцова  
«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (ДИПЛОМНЫЙ ПРОЕКТ)**

**Тема:** Тема проекта

|  |  |  |
| --- | --- | --- |
| **Выполнил:** |  | **Руководитель работы:** |
| Студент 4 курса, группа 4ИС-14/К |  | Преподаватель |
|  |  | Ученая степень, должность |
| ФИО |  | ФИО |
| (Ф.И.О.) |  | (Ф.И.О.) |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| (подпись) |  | (подпись)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  дата защиты, оценка |

Москва, 2023

Оглавление

[Введение 3](#_Toc4762)

[Глава 1 Проектирование интернет-магазина 4](#_Toc32698)

[1.1 Использующиеся технологии и понятия 4](#_Toc5027)

[1.2 Техническое задание 4](#_Toc18108)

[1.3 Схемы работы магазина 4](#_Toc7901)

[Глава 2 Разработка интернет магазина 5](#_Toc18535)

[2.1 Разработка frontend составляющей сайта 5](#_Toc25460)

[2.2 Разработка backend составляющей сайта 5](#_Toc766)

[2.3 Работа с Базой данных 5](#_Toc27452)

[Заключение 6](#_Toc21281)

[Список использованной литературы 7](#_Toc6666)

Введение

Сайт - это необходимый инструмент, который используется в жизни человека повсеместно. Сайты предлагают доступный и быстрый вариант получения любой информации. Сайты могут быть как на компьютере, так и на телефоне, что делает их гибким вариантом получения информации, так как ими можно пользоваться в любом месте. Сайты создаются программистами, которые создают лицевую часть сайта, с которой взаимодействует пользователь, а также и внутреннюю часть сайта, которая отвечает за алгоритмы, подгрузку данных с базы данных, обмен информацией и т.д.

Веб-программирование - это процесс создания сайтов, веб-серверов и иных инструментов помогающих сайтам функционировать и развиваться. Веб-программисты делают неописуемый вклад в развитие сайтов, что позволяет пользователям получать информацию быстрее, например за счет быстрой загрузки сайта и быстрой серверной части.

Понятие веб является актуальной уже на протяжении нескольких лет и будет оставаться неизменной. Сайты не возможно заменить на приложения из-за их гибкости и удобности. Например: приложение на компьютере не возможно открыть на телефоне, если не написать такое же приложение на телефон используя другой язык. Даже если открыть приложение получится, то оно будет отображаться не удобно для пользователя или не будет выполнять свои основные функции. Из-за фактов неудобного отображения приложений придумали браузер, который и воспроизводит сайты, на одном языке. То есть, если программист создаёт сайт, например на JavaScript, то для мобильных устройств будет использоваться тот же язык и с тем же проектом.

Цель дипломного проекта разработать удобный сайт продажи мебели для пользователей используя новые технологии разработки сайтов.

Чтобы грамотно и правильно создать веб-сайт нужно поставить и решить некоторые задачи

1. Изучить теорию - для создания сайтов нужно вникнуть в новые технологии и понять как работают те или иные инструменты.
2. Ознакомиться с техническим заданием - для разработки дипломного проекта к нам обратиться компания со своими требованиями по созданию сайта.
3. Составить схемы работы сайта - Одного изучения теории может не хватить для полного понимая работы. Нужно создать понятные схемы работы сайта и придерживаться им.
4. Разработать клиентскую часть сайта - Клиентская часть сайта, лицо всего сайта, пользователь выбирает сайт по удобному и красивому отображению страницы. Сделая красивый сайт пользователь останется на нем и продолжит выбирать товары.
5. Разработка серверной части сайта - Серверная часть сайта, мозг сайта. Она отвечает за подрузку данных сайта, за предачу и сохранение информации. За составление алгаритмов и иных дополнительных решений сайта. Полноценному сайту невозможно функционировать без серверной части.
6. Настроить базу данных для хранения информации. База данных является созависимым от серверной части сайта. База данных дает возможность хранить информацию о товарах, пользователях или иных данных для сайта. Она является важным элементом сайта на уровне с сервером.
7. Составить заключение о выполненной работе, проверить работу сайта и убедиться в хорошем функционировании. Подвести итоге проделанной работы.

Глава 1 Проектирование интернет-магазина

* 1. Использующиеся технологии и понятия

Бра́узер — это программное обеспечение для просмотра страниц состоящих из веб-документов и файлов к ним.

Веб-страница — это документ, доступ к которому осуществяется в сети через браузер.

Компиляция — сборка программы, во время которой написанный язык программирования переводится в машинный код.

Идентификатор или же ID — уникальный признак объекта, позволяющий отличать его от других объектов, то есть идентифицировать.

Git — Система контроля версий. Создана была для контроля разработки ПО на каждом этапе проектирования. Представляет собой набор коммитов, которые содержат в себе файлы и информацию проекте. При добавлении нового коммита, он сверяется со старой версией коммита и если они рознятся, записывает новые данные, если нет, то создает ссылку на старый коммит.

HTML — Язык гипертекстовой разметки документов для веб-страниц в браузере. Браузер получают документ с расширением .html от сервера по определенным протоколам http или https, браузер интерпритирует код внутри html файла в интерфейс для отображения. Также такие документы можно открывать локально. Внутри html есть блоки, так называемые теги, с помощью которых строиться интерфейс и веб-страница. Приведем пример тегов: <p> - выводит текст на страницу, <img> - выводит фото на страницу. Язык был придуман для обмена информацией, как научной так и технической, для использования обычными людьми. HTML позволяет создать простой красивый документ, который в том числе будет поддерживать гипертекст, то есть создание ссылки на другие документы html. В последствии были придуманы дополнения к языку, которые добавляют стилизацию, а также управление поведением страницы.

DOM — Программный интерфейс, позволяющий программам получить доступ к содержимом, а также изменять содержимое в HTML. Таким образом с помощью DOM документ можно представить как дерево узлов состоящее из атрибутов, классов и тегов, которое связанно между собой отношениями (Родитель-дочерний).

SVG (Scalable Vector Graphics, на русском Масштабируемая Векторная Графика) — язык разметки, созданный и предназначенный для описания двумерной графики. Так же SVG поддерживает как статичное отображение, так и предоставляет анимированное изображение.

CSS (Cascading Style Sheets, на русском Каскадные Таблицы Стилей) — язык стилизации документов, чаще всего html, а также иным документам, например SVG. CSS используется для создания на странице цветов, стилей и иных возможных стилей для страницы. CSS создан для отделения от написания основной логики страницы на html, таким образом логика создания происходит в html, а в css происходит его стилизация. Такой способ помогает улучшить понимание страницы, читабельность для верстальщика и удобство в написании.

ORM или Объектно-реляционное отображение — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». Существуют как проприетарные, так и свободные реализации этой технологии.

NPM (Node Package Manager) — менеджер пакетов, входящий в состав Node.js. Установка пакета производится при помощи команды: «npm install packagename».

JavaScript, а также JS — язык программирования, который поддерживает объектн-ориентированные и функциональные стили. Язык часто используется для браузеров как язык сценариев, чтобы добавлять интерактив веб страницы. Язык также можно использовать как язык программирования благодаря движку , от компании GOOGLE, V8, который позволяет компилировать исходный код JavaScript в собственный машинный код. Благодаря подобной логике была созданна платформа NODEjs.

NODEjs — это программная платформа выпущенная в 2009 году и созданная на движке V8. Платформа превращает JavaScript из узкоспециализированного языка для браузеров в язык общего назначения. Такая методика позволяет удобно создавать как сайты, добавляя в них JavaScript, так и веб-сервера на NODEjs с которыми в последствии будут общаться сайты. Также NODEjs дает возможность создавать десктопные приложения благодаря фрэймворкам ELECTRON.

REACTjs — Популярная бибилиотека JavaScript позволяющая удобно создавать веб страницы. REACT разрабатывается и поддерживается компанией META, а также сообществом разработчиков и компаний. Библотека представляет быструю и высокую скорость работы страницы, а также простоту и маштабируемость путем добавления новых компонентов. В REACT также используются менеджеры состояний, который часто необходимы для создания сайтов, например: MOBx или REDUX. Впервые REACT появился в 2013 году и использовался в новостной ленте месенджера. А уже в мае он был представлен открытым исходным кодом. REACT использует методику виртуального DOM дерева. Он создает структуру страницы и запоминает ее, в последствии сравнивая текущее состоянии с тем, которое лежит в памяти, для оптимального отображения DOM дерева.

Redux — библиотека для JavaScript с открытым исходным кодом, предназначенная для управления состоянием приложения. Чаще всего используется в связке с React или Angular для разработки клиентской части. Содержит ряд инструментов, позволяющих значительно упростить передачу данных хранилища через контекст.

JSX — это расширение синтаксиса языка JavaScript, который позволяет описать структуру интерфейса благодаря HTML.

React Hooks или иначе Реакт Хук— хуки помогают облегчить написание сайтов, помещая логику в отдельный компонент, который в последствии можно будет переиспользовать.

.env или dotenv — инструмент позволяющий создавать переменные окружения, которые помещаются в отдельный файл с форматов .env. В последствии к ним можно будет обращаться в программе с помощью библиотек. Такие переменные записываются большими буквами с разделением в виде нижнего подчёркивания. В реакте следует использовать в начале приставку “REACT\_APP\_”, Изображение 1.



Изображение 1 Название переменных окружения

Dokcer — программное обеспечение, позволяющее удобно развёртывать и управлять приложением в любом месте. Программа позволяет упаковывать приложения со всеми его зависимостями и элементами окружения в изолированный контейнер, который в последствии можно будет развернуть на на любой системе Linux .

Clsx — маленький инуструмент для className структуры в react, который позволяет добавить 2 и более различных стилей. Так как react ограничивает количество стилей до 1.

Gsap — библиотека, позволяющая создавать анимации через javascript удобно и быстро. Обладает огромным списком возможностей и ииспользуется большим количеством людей.

React-Gsap — утилита позволяющая gsap взаимодействовать с контейнерами react.

React-yandex-maps — библиотека помогающая взаимодействовать с яндекс картами встраивая их при помощи компонентов react.

React Router DOM — это модуль узла, который предназначен для маршрутизации в веб-приложениях. Он позволяет программистам создавать маршруты для одностраничного приложения React.

JS-Cookie — Простой и лёгкий JavaScript API для обработки файлов cookie. Позволяет легко и быстро взаимодействовать с cookie файлами.

React Input Mask - библиотека позволяющая быстро и удобно составлять определённые маски на элементах ввода. Пример: +7 999 999 99 99.

Express — фрэймворк для NodeJS позволяющий удобно создавать веб-приложения и API. С помощью express можно удобно создавать веб-сервера.

Nodemon — это инструмент, который помогает разрабатывать приложения на основе Node.js, автоматически перезапуская приложение узла при обнаружении изменений файлов в каталоге. nodemon не требует никаких дополнительных изменений в коде или методе разработки. Nodemon замена оболочки для node. Чтобы использовать nodemon, нужно заменить слово node в командной строке при выполнении скрипта.

JSON Web Token (JWT) — это открытый стандарт для создания токенов доступа, основанный на формате JSON. Как правило, используется для передачи данных для аутентификации в клиент-серверных приложениях. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения подлинности аккаунта.

Cross-origin resource sharing (CORS — «совместное использование ресурсов между разными источниками») — технология современных браузеров, которая позволяет предоставить веб-страницам доступ к ресурсам другого домена.

bcrypt — адаптивная криптографическая хеш-функция формирования ключа, используемая для защищенного хранения паролей. Разработчики: Нильс Провос и David Mazières. Функция основана на шифре Blowfish, впервые представлена на USENIX в 1999 году. Для защиты от атак с помощью радужных таблиц bcrypt использует соль (salt); кроме того, функция является адаптивной, время её работы легко настраивается и её можно замедлить, чтобы усложнить атаку перебором. Шифр Blowfish отличается от многих алгоритмов вычислительно сложной фазой подготовки ключей шифрования (англ.). Провос и Mazières воспользовались этой особенностью, но изменили алгоритм подготовки ключей, получив шифр «Eksblowfish». Количество раундов в подготовке ключей должно быть степенью двойки; конкретная степень может задаваться при использовании bcrypt.

PostgreSQL — свободная объектно-реляционная система управления базами данных (СУБД). Существует в реализациях для множества UNIX-подобных платформ. Функции являются блоками кода, исполняемыми на сервере, а не на клиенте БД. Хотя они могут писаться на чистом SQL, реализация дополнительной логики, например, условных переходов и циклов, выходит за рамки SQL и требует использования некоторых языковых расширений. Функции могут писаться с использованием одного из следующих языков: Встроенный процедурный язык PL/pgSQL, во многом аналогичный языку PL/SQL.

PostgreSQL допускает использование функций, возвращающих набор записей, который далее можно использовать так же, как и результат выполнения обычного запроса. Функции могут выполняться как с правами их создателя, так и с правами текущего пользователя. Иногда функции отождествляются с хранимыми процедурами, однако между этими понятиями есть различие. С девятой версии возможно написание автономных блоков, которые позволяют выполнять код на процедурных языках без написания функций, непосредственно в клиенте.

PG — это набор модулей node.js для взаимодействия с вашей базой данных PostgreSQL. Он поддерживает обратные вызовы, промисы, async/await, пул соединений, подготовленные операторы, курсоры, результаты потоковой передачи, привязки C/C++, расширенный анализ типов. Как и у самого PostgreSQL, у него много функций:

PG-HSTORE — Пакет узлов для сериализации и десериализации данных JSON в формат hstore.

ORM ( Object-Relational Mapping, Объектно-реляционное отображение, или преобразование) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». Существуют как проприетарные, так и свободные реализации этой технологии.

Sequelize — это современная ORM TypeScript и Node.js для Oracle, Postgres, MySQL, MariaDB, SQLite и SQL Server и других. Благодаря надежной поддержке транзакций, отношениям, нетерпеливой и ленивой загрузке, репликации чтения и многому другому.

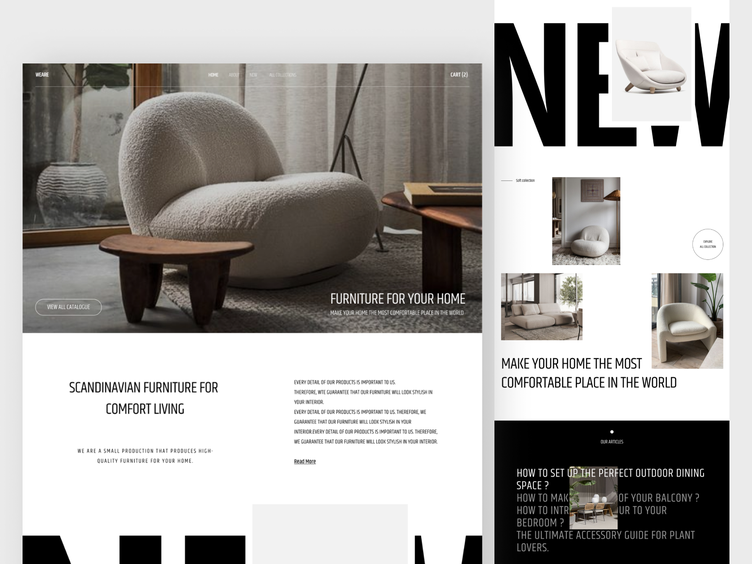
Uuid — Библиотека позволяющая создавать уникальные не повторяющиеся айди. Такие айди удобно использовать на пользователях или фотографиях для предотвращения повторения.

Express Fileupload — Библиотека позволяющая загружать, а также получать файлы через сервер. Библиотека принимает, файлы в теле запроса, в последствии может из загружать или отправлять. Библиотека непосредственно работает через Express.

* 1. Техническое задание

Для дипломного проекта берётся техническое задание или же просто ТЗ.

Компания Stylia Online обратилась к нам для создания сайта. Компании нахватает продаж и хочется повысить спрос путем продажи товаров через интернет магазин. Компания также предоставила макет для проектирования сайта и пару рекомендация.



Изображение 2 Макет для сайта Stylia Online

Компания хочет похожий дизай, но с проработкой дополнительных страниц.

Шапка сайта - Шапка должна быть одинаковой на всех страницах, кроме главной, а на странице с ошибкой её не должно быть. Шапка должна быть статична с цветом представленным ниже тёмным оттенком текстом и с такими же цветами элементами на белом фоне. Шапка главной страницы должна быть как в макете имея анимацию появления везде.

Подвал сайта - подвал должен быть на всех страницах кроме страницы с ошибкой. В подвале должно быть название магазина, сверху по середине, по центру подвала должны быть возможность подписки на рассылку. Слева по центру должно быть меню навигации, а справа по центру социальные сети. По центру снизу должно быть оповещение о защите прав с 2020 года по 23 год. Сслыки на соц сети: vk - [https://vk.com](https://vk.com,) , youtube - https://www.youtube.com/ , telegram - https://www.telegram.org/ , instagram - https://www.instagram.com/ .

Страница «О нас» - Страница должна быть в черно белом стиле. Должна быть сетка блоков, где слева начинается текст, справа картинка, далее наоборот, картинка - текст.

Страницы «Товары» - Справа страницы выбор товара по категориям, справа товары и кнопки фильтрации по популярности, дешевизне, дороговизне.

Страница «Конткаты» - Сообщение с текстом «Нас легко найти», блок слева которого адреса, а справа карта яндекс с отметками адресов.

Блок «Войти» - представляет собой появляющийся элемент с возможностью войти и зарегистрироваться, после входа перебрасывает в аккаунт.

Блок «Корзина» - Появляется справа от страницы в себе хранит товары и возможность перейти к оформлению.

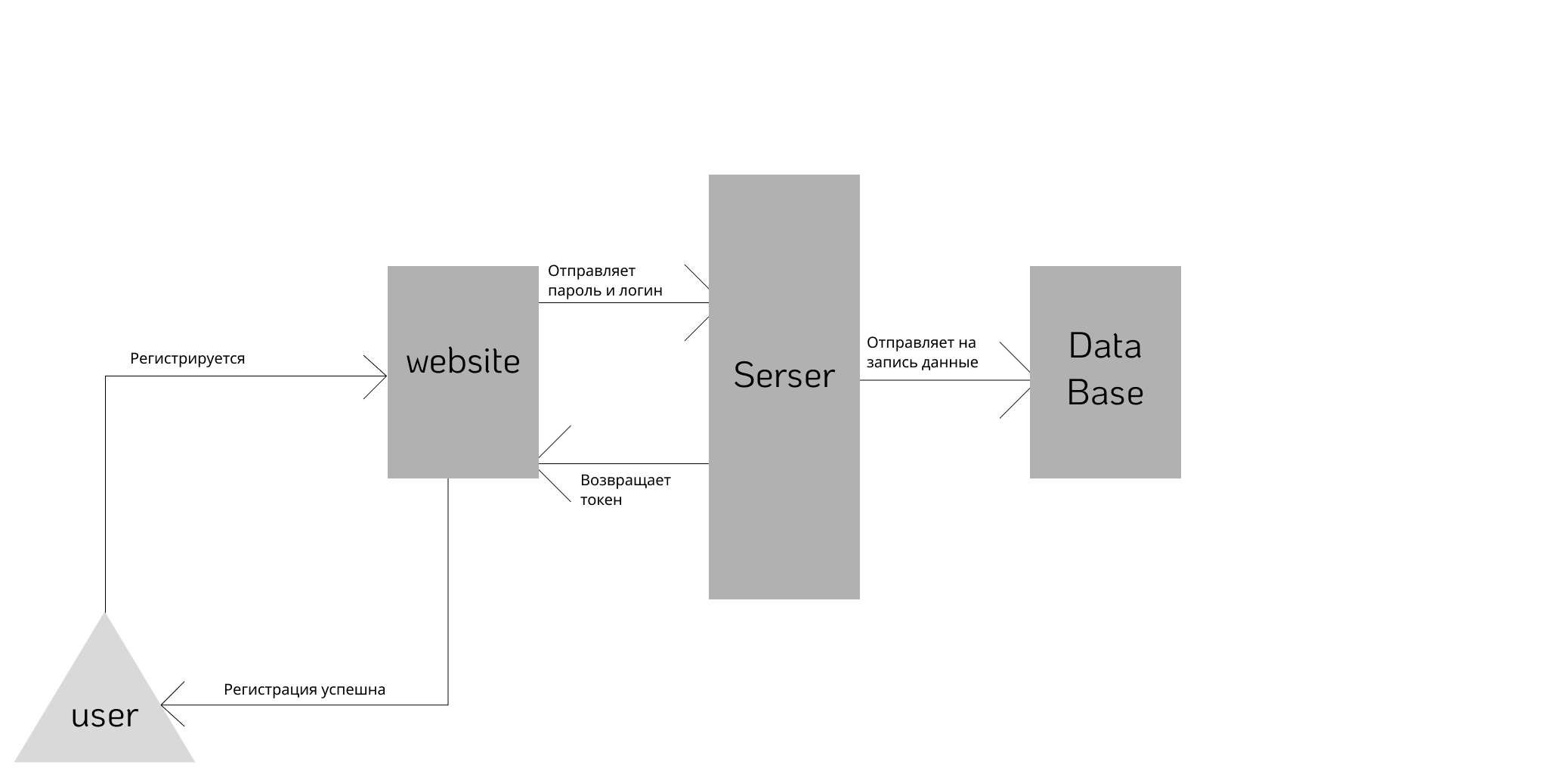
Страница «Профиль» - Позволяет войти в свой аккаунт, если пользователь не зарегистрирован показывает карточку с надписью «ВОЙДИТЕ В АККАУНТ, ЧТОБЫ СОВЕРШАТЬ ПОКУПКИ». При регистрации показывает карточку пользователя, сверху случайный градиент, а снизу информация о пользователе и возможность просматривать заказы.

Страницы «Админ панель» - позволяет создавать, редактировать, удалять товары. Позволяет создавать, удалять, редактировать, категории. Позволяет редактировать роль и удалять аккаунты.

* 1. Схемы работы магазина

Чтобы понимать принцип работы на прямом примере, нужно создать схемы работы.

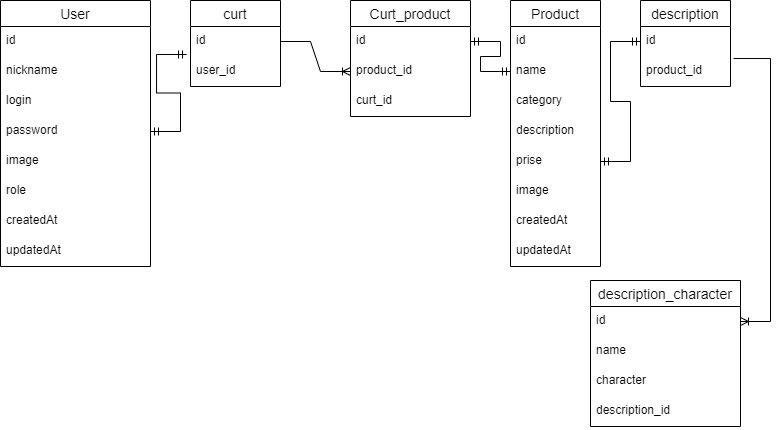
Схема общения сайта, сервера и базы данных. Когда пользователь совершает вход в аккаунт, он отправляет запрос, содержащий его данные, такие как: Логин и пароль. Сервер принимает эти данные, шифрует пароль и создаёт стандартное изображение и никнейм пользователя, в конечном итоге отправляет запрос в базу на создание строчки в базе. База записывает логин, пароль и другие передаваемые сервером данные. В конченом итоге сервер создаёт специальный токен и отправляет его сайту в месте с дополнительными данными, схемы показана на изображении.



Изображение 3 Схема работы сайта, сервера и базы данных

База данных состоит из взаимосвязанных таблиц. Связь эти таблиц нужно описать заранее, чтобы понять, какая таблица от чего зависит. Создаётся диаграмма пользователя. Пользователь будет связан с диаграммой корзины, по методу один к одному, то есть одну корзину может иметь один пользователь. Диаграмма корзины хранит в себе товары, для этого нужно создать диаграмму корзина продуктов, она будет хранить ссылку на корзину и на продукт. Корзина будет связана с корзиной продуктов методом один ко многим, так как корзина может иметь в себе несколько корзин продуктов. Корзина продуктов будет непосредственно иметь связь один к одному с полным списком продуктов сайта. Так как один продукт может находиться в одной корзине продуктов.

Так как у Продукта есть подробное описание, о ширине, цвете, то нужно создать диаграмму описания, которая будет иметь связь один ко многим с диаграмма описание характеристик. Так же один продукт имеет связь один к одному с продуктом, на изображении 4.

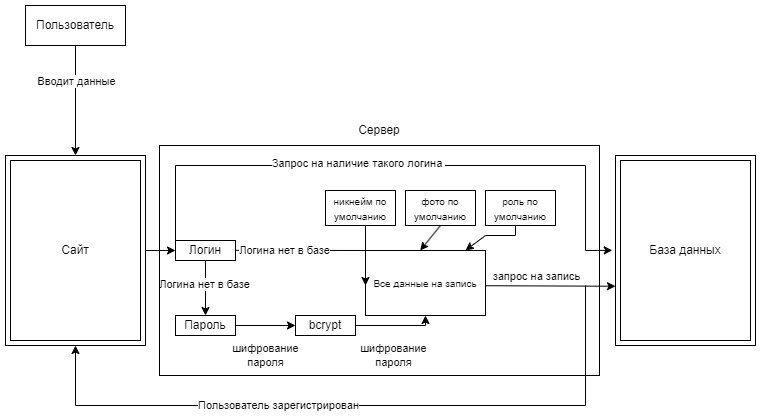


Изображение 4 Диаграмма связей моделей в базе данных

Подробный принцип работы входа пользователя, с шифрованием, регистрацией и проверкой токена. Чтобы понять принцип работы шифрования и создания уникального токена пользователя, нужно составить диаграмму, которая будет показывать, как происходит работа с данными пользователя.

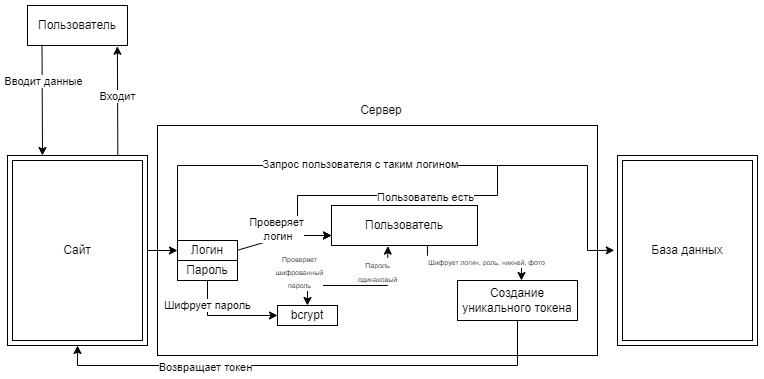
Когда пользователь вводит в форму свои данные, он отправляет «fetch» запрос методом POST, конвертируя их в JSON формат. Эти данные ловит сервер. В зависимости от запроса, который отправляется конкретно на сайт по ссылке будет определенный результат. В нашем варианте регистрация, вход, проверка происходит на разных сайтах. Рассматривается каждый вариант:

1. Регистрация - во время прихода данных сервер их ловит и получает. Смотрит пустой ли запрос. И отправляет запрос в базу на проверку, нет ли такого логина в базе. Далее сервер шифрует пароль и отправляет в базу запрос о записи логина, зашифрованного пароль, а также фотографии, никнейма и роли по стандартным установленным значениям. Если все окей на сайт возвращается сообщение о том, что пользователь зарегистрирован с отметкой об успешной регистрации.



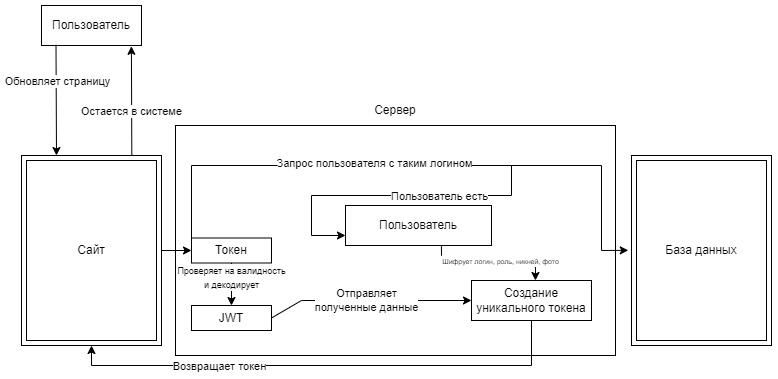
Изображение 5 Схема регистрации

1. Вход в аккаунт, при входе в аккаунт пользователь отправляет теже данные, что и при регистрации. Во время получения данных сервер отправляет запрос на вытягивании пользователя при котором проверяется логин на существование. Если все верно, шифруется пароль и проверяется на схожесть с тем, что есть в базе. Далее генерируется уникальный токен входа на основе никнейма, логина, роли, айди и фотографии. Данный токен возвращается сервером ответом на вход, в дальнейшем данный токен будет использоваться для понимания того, что пользователь зарегистрирован.



Изображение 6 Схема Входа

1. Проверка токена, происходит при каждом обновлении страницы. Данная механика позволяет узнать зарегистрирован ли пользователь и валиден ли его токен, чтобы не было взломов аккаунтов, даже если токен изменили. При обновлении страницы на сервер поступает токен, токен хранит в себе помимо данных и номер аунтефикации, если он совпадает и верен, то пользователю возвращается новый токен, который хранит в себе теже даннные с другим серийным номером. Если токен невалиден, то ничего не отправляется, тем самым выбрасывая пользователя с аккаунта.

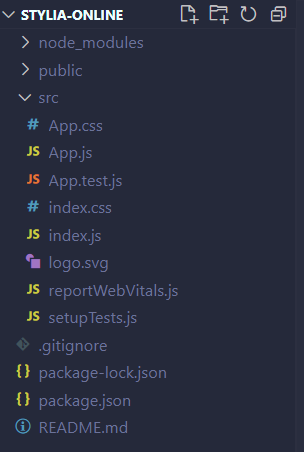


Изображение 7 Схема аунтефикации

Глава 2 Разработка интернет магазина

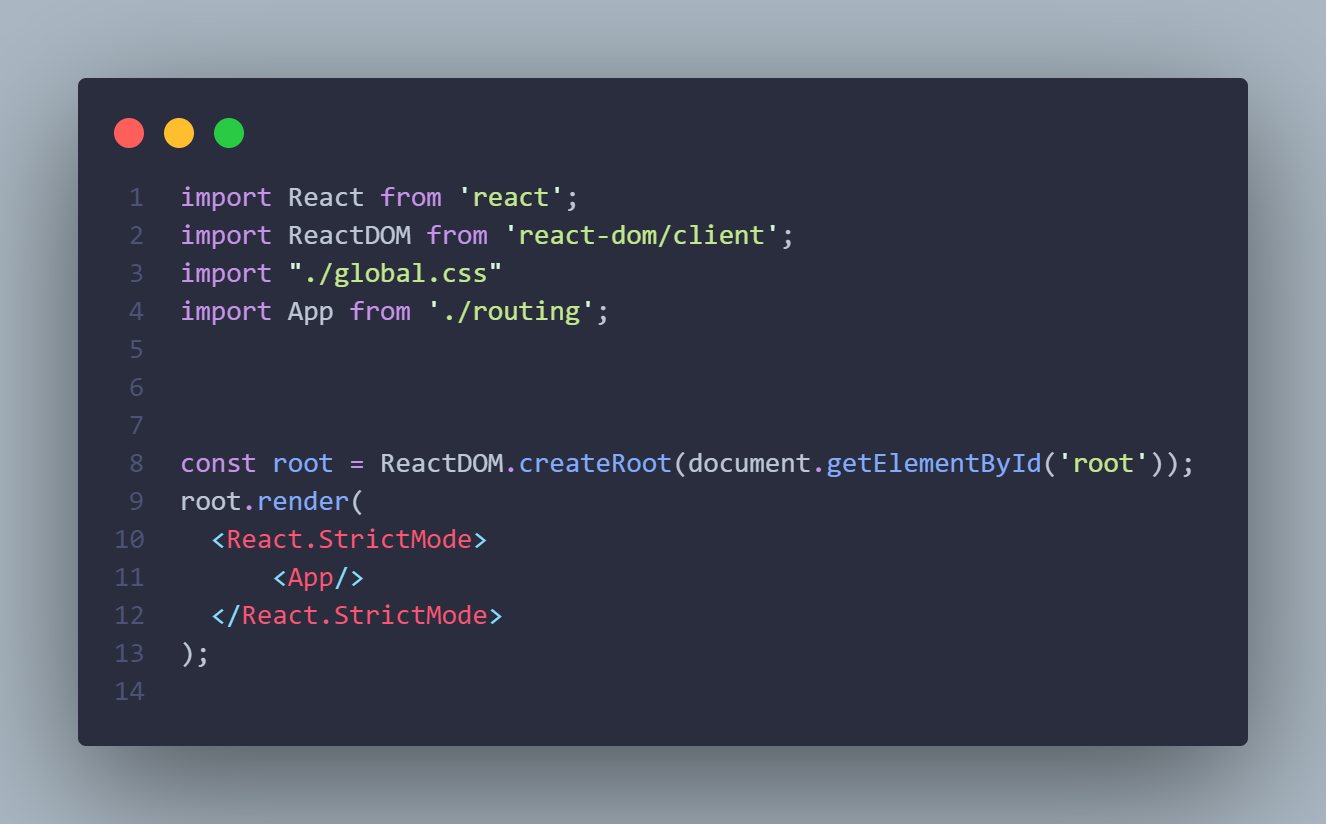
2.1 Разработка frontend составляющей сайта

Для разработки лицевой части сайта возьмём за основу библиотеку React. Данная библиотека меняет подход создания сайтов, делает его быстрее и упрощает разработку. Начав нужно создать папку и открыть её через Visual Studio Code ипользоваться будет стандартная структура, в Visual Studio Code открывается консоль PowerShell через Cntrl+Ё и прописывается команда «npx create-react-app .» .Точка показывает то, что все файлы поместятся в папку в которой сейчас находятся, если же вместо точки написать любой текст на английском, то создаётся папка с этим же названием. После установки всех зависимостей и компонентов выглядеть это будет как на изображении 2.



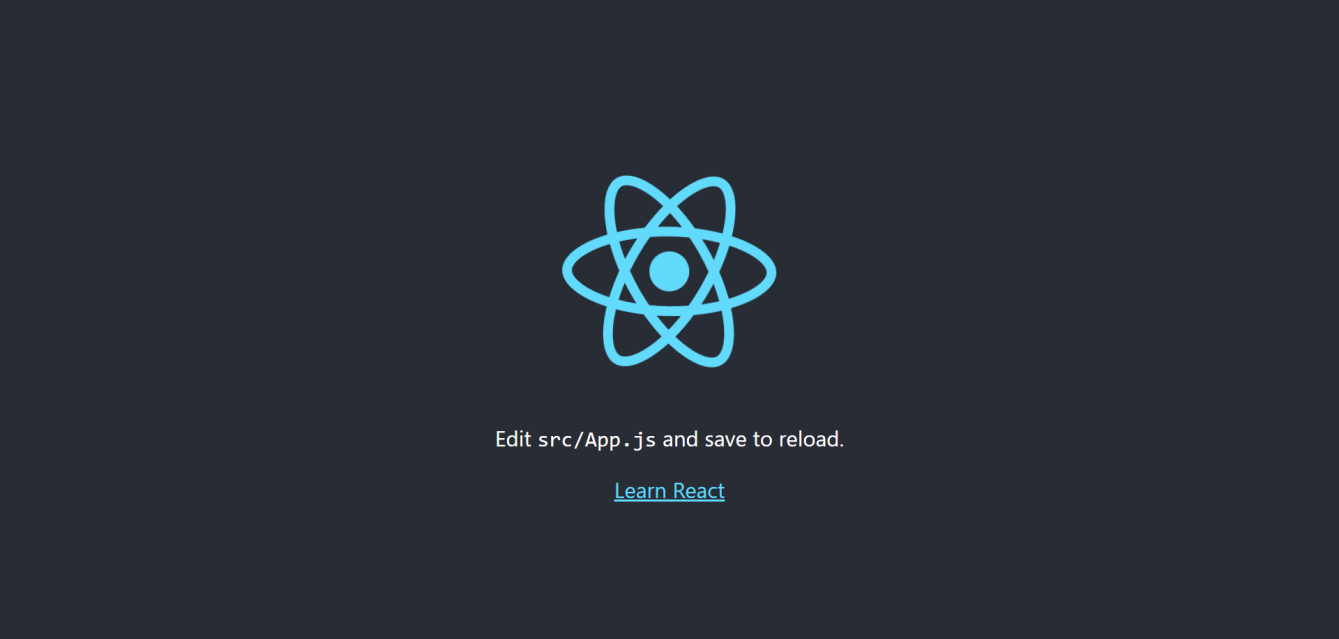
Изображение 8 Созданная структура react

Создались папки, такие как «node\_modules», которая хранит в себе дополнения и зависимости. «public» хранит главный файл index.html, иконки и robots.txt. «src» - это основная папка, которая хранит в себе все компоненты и стили. Внутри нее основым файлом является index.js, который отвечает за рендер компонентов на странице. Импортируется метод ReactDom, данный метод будет внедряется и отслеживает дерево DOM на странице. Создаётся константа, которая обращается к ReactDom и отслеживается тег div с id root в файле index.html. В последствии root рендерит все элементы внутри себя на страницу. «React.StrictMode» отвечает за повторный рендер, все компоненты рендерятся по 2 раза, данная механика позволяет выявлять ошибки на этапе разработки. Также импортируется компонент App в котором теги и иные компоненты. Файл index.js, где рендерится сайт, оставляют всегда таким какой он есть, то есть рендерится в нем только компонент App, чтобы не нарушать понимание проекта изображение 3.



Изображение 9 Структура index.js

«.gitignore» - запрешает передавать файлы на «GitHub». В «pachage.json» находятся новый библиотеки и описание проекта. Также создался файл «Readme.md», который отвечает за описание на «GitHub». Прописав команду «npm start» запуститься сайт по ссылке «localhost:3000» появится там то, что на показано на изображении 4.



Изображение 10 Предустановленное отображение на странице

Нужно удалить лишние файлы, строки кода и иконки чтобы сделать проект пустым. На сайте окажется просто белый экран. Для обязательного обнуления всех стилей нужно создать файл «global.css» в нем добавляется основные стики, которые должны использоваться абсолютно во все проекте. В файле global.css прописывается импорт шрифтов, а также добавляются основные константы цветов, через «root». «root» прикрепляется ко всем элементам на странице и благодаря этому в css можно использовать константы. Внутри «root» создаётся константы цветов, прописывается «--» к названию константы, а через двоеточие сам цвет. Прописывается «\*», которая привязывается также как и «root» внутри которой обнуляется внешние отступы, внутренние отступы и т.д. Также у тега body меняется элемент скролла через webkit-scrollbar и изменяется движущийся элемент через -webkit-scrollbat-thumb. Убираются лишние обводки, текст делается заглавным через «text-transform», изображение 5.



Изображение 11 Общие стили файла

Создаётся файл layout.jsx, файл отвечает за одинаковое отображение сайта на всех страницах. Объявляется стрелочная функция «Layout» и стандартным образом экспортируется. Внутрь функции передаются объекты - пропсы, в данном случае будут передаваться дети, в которые обёрнуты в «Layout». В конченом итоге будет div в котором находятся другие компоненты. Div принимает class, в реакте вместо класса используется className, это тоже самое что class, но может примать объекты изображение 6. После знака равно открываются фигурные скобки, в данном варианте это модуль css. Импортируется модуль css в константу styles. Дальше через styles можно будет получить стили css. В react для удобной работы стилей используется модуль css, так как он позволяет деструктурировать файл и обращаться к стилю напрямую. Это позволяет называть компоненты также, как эти стили назывались в другом компоненте и конфликта стилей из-за этого не будет. Объявляется файл «Layout.css», но перед расширением файла добавляется модуль «Layout.module.css». В нем находится отступ, а также выставляются все буквы заглавными, изображение 7.



Изображение 12 Layout компонент



Изображение 13 файл Layout.module.css

Чтобы в react можно было сделать возможность перемещения между страницами используется библиотека «react router dom». Она позволяет сделать маршрут страниц. С помощью команды «npm i react-router-dom» устанавливается библиотека. После установки создаётся файл routing.js в него импортируются необходимые методы и компоненты, которые будут отвечать за отображение страницы, изображение 8. Метод «createBrowserRouter» создаёт из всех передаваемых в него компонентов маршрут сайтов, а «routerProvider» выводит это на страницу. После импортов объявляется функция App, которая в последствии экспортируется, чтобы к ней могли получить доступ другие компоненты. Внутри функции объявляется константа router, которая принимает в себя «createBrowserRouter». Он хранит в себе массив объектов и делает между ними маршруты. Объект представляет собой: «path» - ссылку на компонет, перейдя по ней пользователь откроет компонент, «Element» - сам компонент который будет отображён, также объект может принять «errorElement», если react router не найдет нужной ссылки по которой перешел пользователь, то он отобразит этот компонент. В последствии функция возвращает «ReactProvider», в который помещается константа router изображение 9.



Изображение 14 Импорт необъодимых елементов в router.js



Изображение 15 Структура маршрутов в routing.js

Каждый импортируемый компонент в routing.js отвечает за страницу, первая страница представляет собой компонент page. Внутри src создается папка page, в ней будут находиться все страницы проекта.

Теперь будут создаваться страницы, которые были импортированы в routing.js

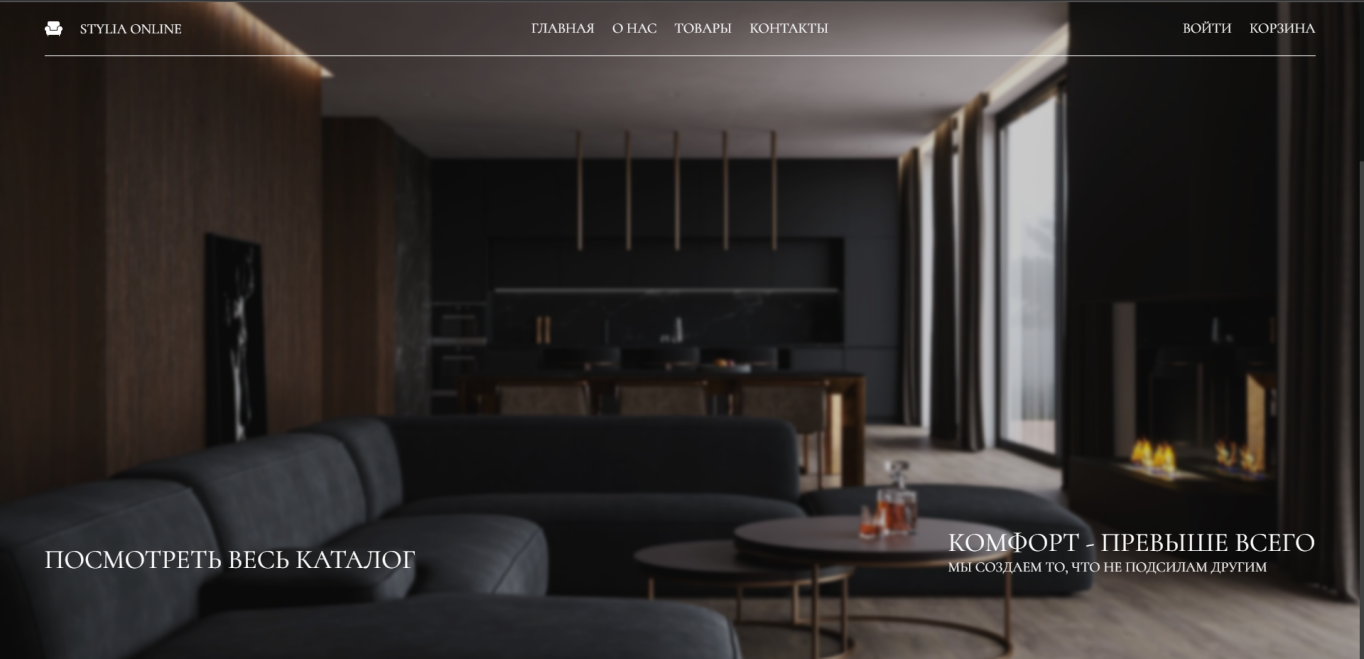
Главная страница: Перейдя в page создается файл page.jsx, а также page.module.css. JSX позволяет использовать внутри функций react теги из html.Чтобы создать компонент нужно сделать стрелочную функцию и внутри нее сделать return передавая html теги. Функция экспортируется, а в самом низу прописывается стандратный экспорт функции. Внутри функции экспортируем и вставляем шаблон как на изображении 10.



Изображение 16 Компонент шаблон

В компонент передаётся также, в будущем созданная шапка сайта через <Header/>. Так как компонент не принимает не какие html элементы, его можно сразу закрыть при объявлении. Также рядом объявляется подвал сайта <Footer/>. Начиная верстать первую страницу нужно обратиться к макету, который предоставила компания. При открытии страницы появляется фотография во весь экран, которая хранит в себе шапку, а также дополнительные элементы. Создаётся контейнер с классов preview, в него будет подгружаться фотография, а чтобы сделать эту фотографию затемнённой и чуть расплытым создаётся внутри добавляется контейнер с классом blur. Blur - растягивается на всю ширину и высоту, также он принимает background с rgb значением (0,0,0, .2) и backdrop-filter со значением blur(2px). Благодаря затемнённому фону и расплывчатому фону, элементы, шапка и текст не будут сливаться с фоном и пользователь будет акцентировать внимание на них. Добавляя к шапке контейнер main и разделяем элементы «Посмотреть товар» и «Слоган компании».Итоговый вариант на изображении 17.

Внутри css зададим через keyframes анимацию для появления текста из пустоты, а также анимацию появления слева у логотипа компании, изображение 18.



Изображение 17 Шапка сайта на главной странице



Изображение 18 Анимация появления текста и логотипа через css

Следующая секция является блоком «О нас». Секция рассказывает кратко о компании и зазывает пользователя на страницу «О нас», где пользователь сможет ознакомиться с компанией подробнее.

Секция делится на 2 элемента, слоганы и опознавательные знаки, а также дополнительное описание с ссылкой на основную страницу. Создаётся тег section в котором будут располагаться элементы. Внутри него создаётся ещё один тег, который будет служить за разделение этих блоков, изображение 19. Импортируется библиотека react-gsap с элементами «ScrollTrigger, Tween» и внутрь секции «О нас» вставляется «ScrollTrigger». «ScrollTrigger» - будет отслеживать движение пользователя, поэтому в него передаются значения «Start и end». Значения отслеживают точку начала и конца анимации начиная от начала самого элемента, подпись center объясняет gsap то, что если пользователь доведёт середину экрана на «Start», то анимация начнется. Также параметром передаётся «Scrub» - отвечающий за плавность анимации. «Tween» - сам элемент у которого будет анимация. Параметрами принимает «From и To» - то есть откуда анимация будет идти и до куда. Для двух элементов есть свой Tween. Важно понимать, что gsap конфликтует с css стилем transition, так как непосредственно его использует и стоит не использовать его на те блоки, которые анимированны, пример кода изображение 20.



Изображение 19 Конечный вариант секции о нас



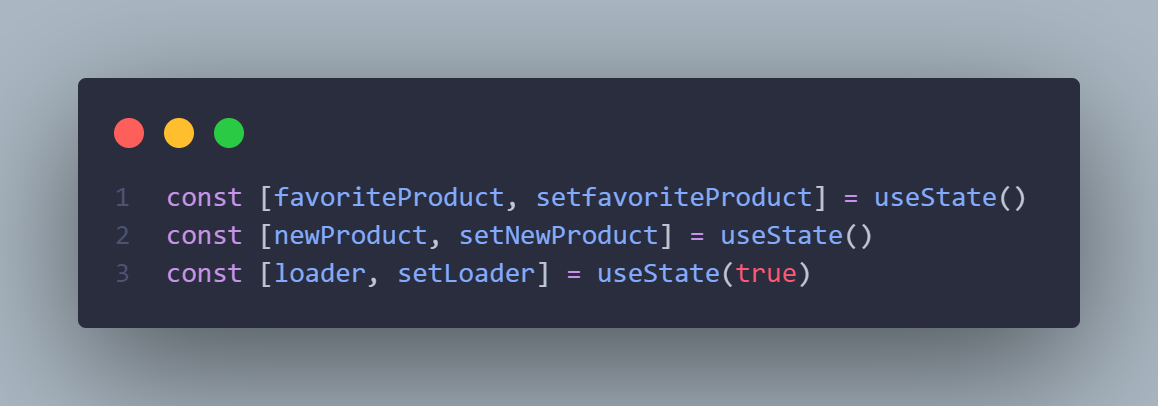
Изображение 20 Код секции о нас

Секция с новыми товарами. Содержит в себе задний фон с текстом «NEW», а также двумя картинками с продуктами. Картинки будут анимированы также как и секция «О нас», только они будут еще крутиться.

Хук useEffect отрабатывает каждый раз, когда изменяется или обновляется компонент. Создаём файл .env в который прописываем переменную «REACT\_APP\_SERVER» в нее перемещается ссылка на сервер, который в последствии будет создан, в данном варианте конченый результат будет показан на изображении 21. Создаём и ниже вызываем асинхронную функцию, внутри useEffect. Асинхронная функция будет делать fetch запрос. Для хранения массивов создаётся два хука useState, а также useState для элемента загрузки. UseState создает константу и функцию, которая может изменять эту константу. Таким образом useState может изменять задавать состояние константе. В зависимости от того, что будет использоваться в useState, то будет вставляться в скобки во время объявления. Если это массив, то ничего не ставится, если строка, то ставятся две кавычки, а если булевое значение (true, false), то ставится одно из них, изображение 22.



Изображение 21 Переменная в REACT\_APP\_SERVER



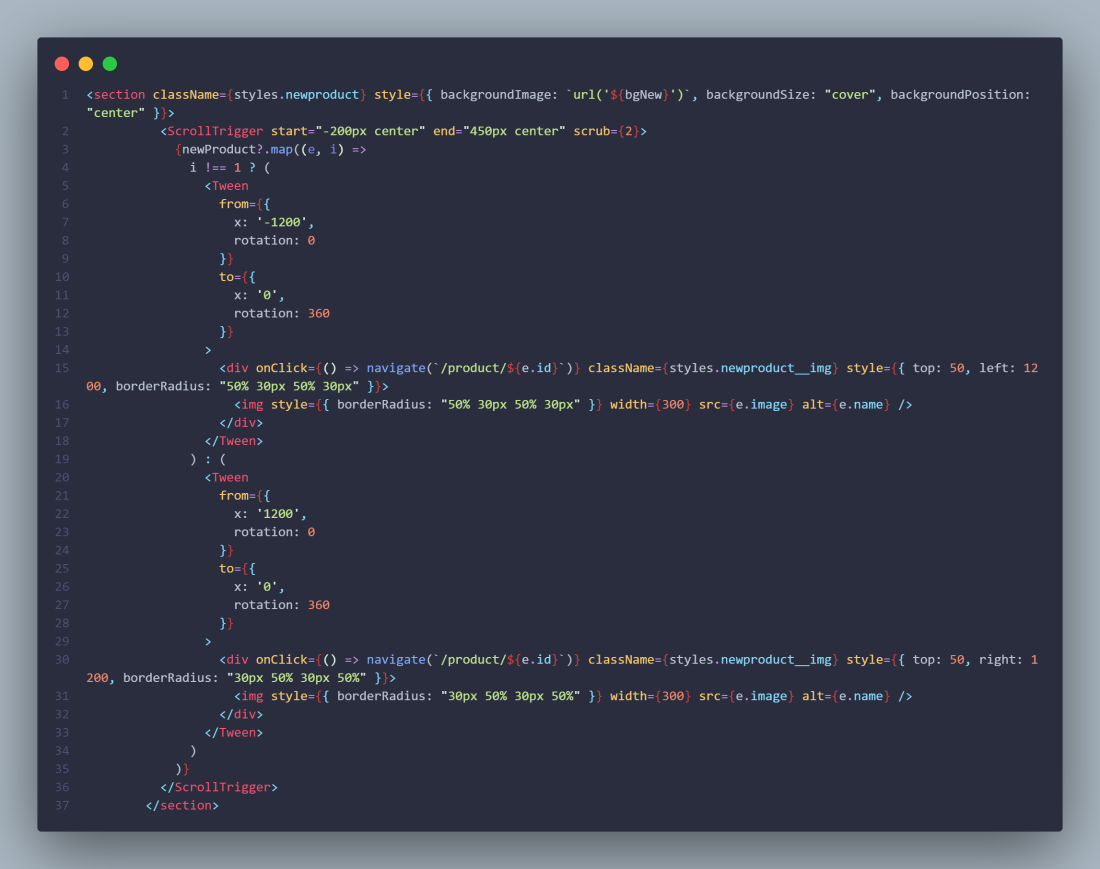
Изображение 22 Хук useState

Создаётся асинхронный запрос на сервер, прописывается внутри fetch ссылка на сайт куда будет запрос,в ссылку вставляется переменная окружения с сервером, далее получается ответ от сервера, который будет конвертирован в json. Запрос получен и если он не пустой, то вырубается лоадер создаётся новый массив productFilter на основе переданных продуктов, так как запрос идёт на получение продуктов. Массив сортируется через метод sort по количеству покупок товара. Далее массив передаётся в useState и внутри него обрезается через метод splice от 0 до 4. Для новых продуктов идёт просто обрезка без сортировки. Для обработки ошибок используется catch в самом конце запроса. Конченый вариант на изображении 23.



Изображение 23 Запрос на получение продуктов

В нутри секции «Новые товары» делается функция перебора map с значением, которое передает товар, она перебирает товары, которые принял сайт при запросе и если его индекс равняется 1 ставит его вправо, а если нет то в лево с разными настройками перемещения, так как 2 блока перемещаются из разных сторон. Внутри анимации «Tween» передаётся фотография товара. Также на блоках есть прослушивание клика, он хранит в себе функцию navigate, это встроенный хук внутри react-router-dom, он позволяет передвигаться между страницами. Если на товар нажмут, то перейдёт на страницу с этим же товаром через его айди. Конечный варинат 24.



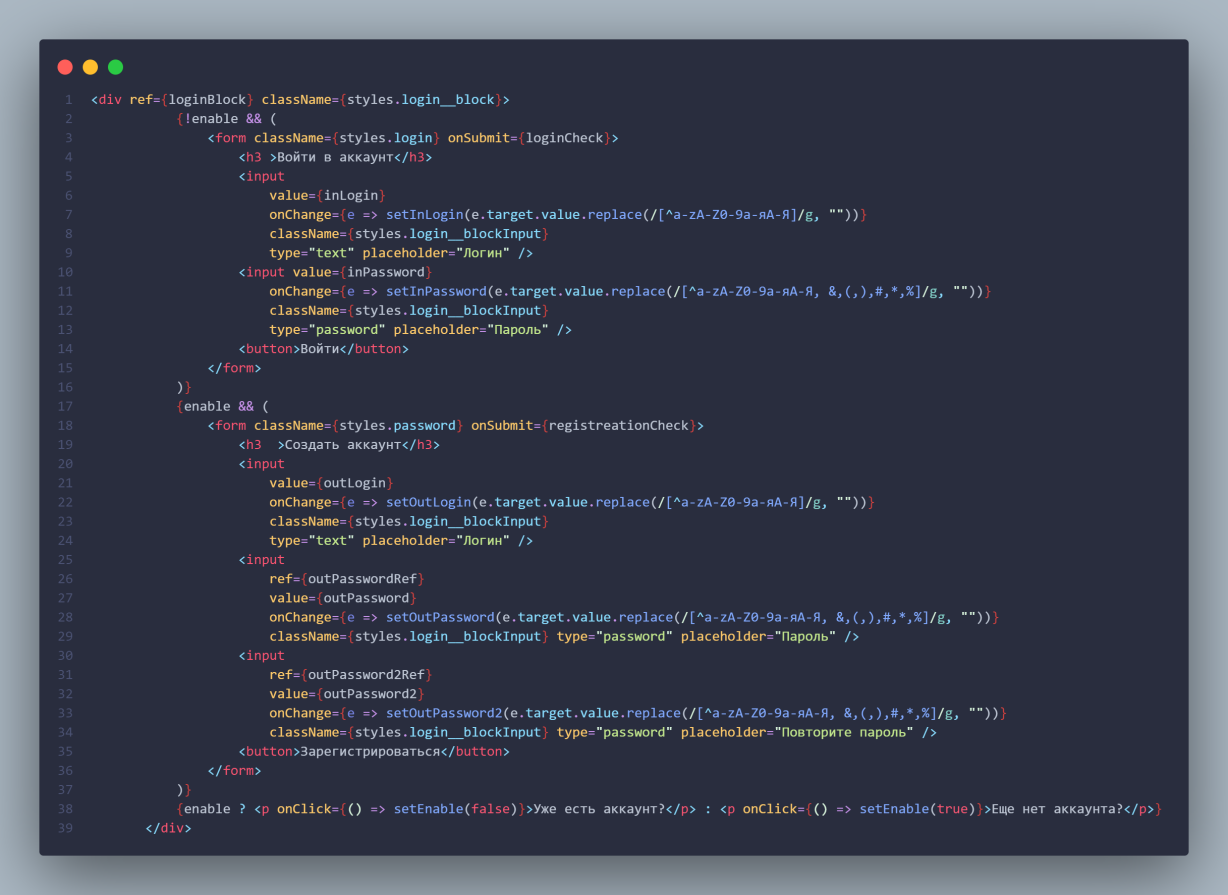
Изображение 24 Конченый вариант с новым товаром

Секция рекомендованных товаров ставит иконку загрузки с самого начала и когда данные получены иконка убирается. Через map вставляются все продукты и добавляется текст «Мы рекомендуем» результат изображение 25.



Изображение 25 Код секции рекомендованных товаров

Блок входа. Блок входа появляется внутри шапки, через специально созданный хук Portal он переносит компонент в html тег div с id «modal». У блока внутри две формы, которые рендерятся от состояния «enbale», если оно false то, появляется вход, иначе регистрация. Также у блока входа есть отслеживающий хук useEffect, он отслеживает нажатия на экране и если пользователь нажимает не на блок и не на элемент в дата атрибутом data-login, код страницы на изображении 26, а код отслеживающего хука на 27.



Изображение 26 Код блока вход



Изображение 27 Код закрытия блока



Изображение 28 Хук портала

Для входа в аккаунт использутся функция, функция вешается как событие отправки формы. То есть, когда пользователь введет данные и нажмет кнопку, на сервер отправится пост запрос, с body, то есть телом. Тело запроса содержит в себе логин и пароль пользователя. С помощью «content-type» запрос показывает о том, что его тело будет содержать json контент. Body отправляет запрос и конвертирует его в json, с помощью «JSON.stringify». Ответом на запрос прилетает токен пользователя, который мы помещаем внутрь стейт менеджера redux, далее помещается токен в куки файл на год выключается компонент входа и перемещается пользователь на страницу его аккаунта Изображение 28.

Чтобы сделать отслеживание переменных в redux создается папка «Store» в ней папка slices. Папка «Slices» хранит в себе структуру массива и функции присваивания с обнулением Изображение 29. В последствии Все это экспротируется через index.js в папке «Store», который принимает эту структуру и экспортирует ее. С помощью «dispatch» в последствии можно будет изменять или добавлять в стейт менеджер переменные для отслеживания.



Изображение 29 Fetch запрос на сервер по регистрации



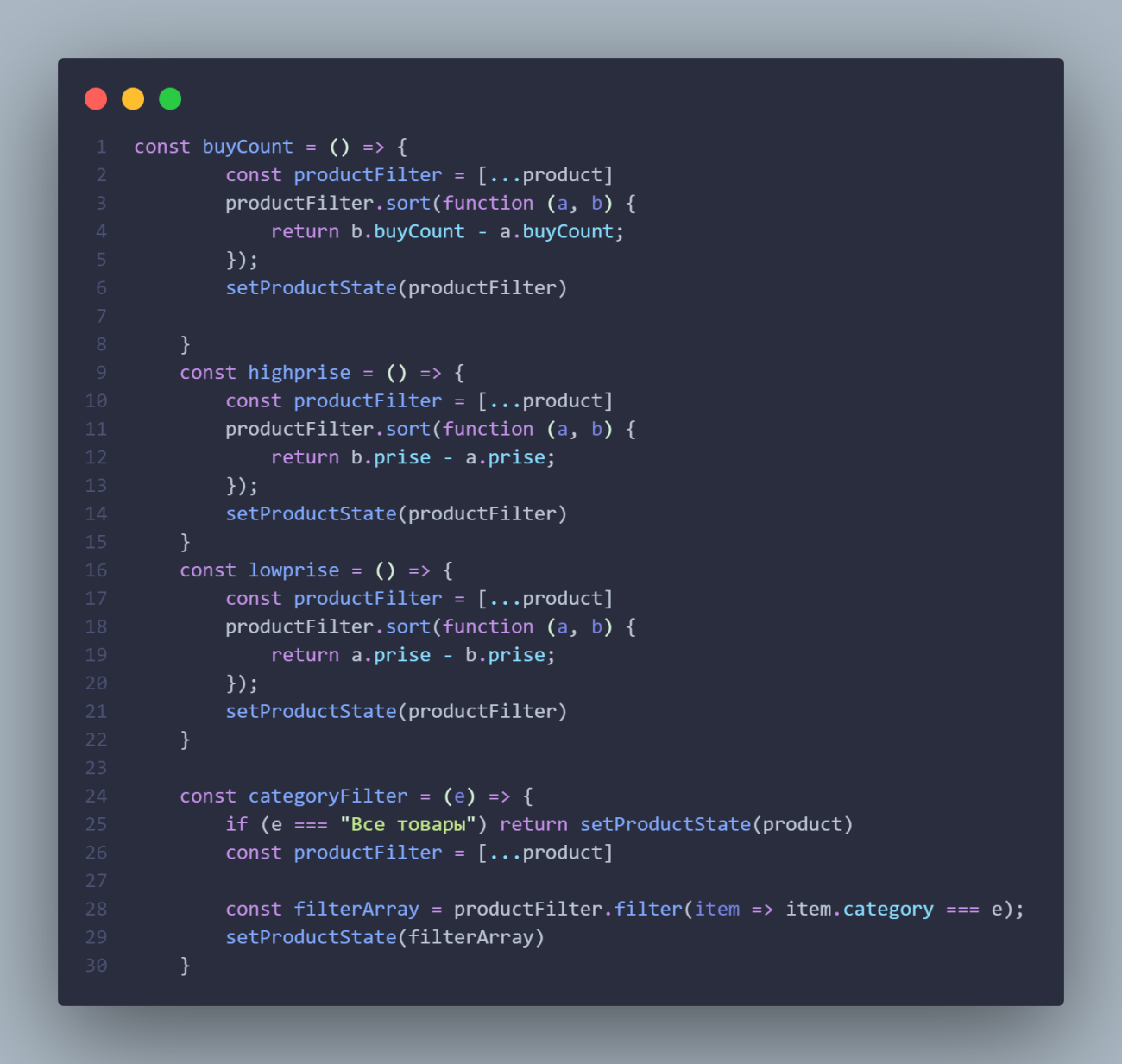
Изображение 30 Структура пользователя в Redux

Страница продукта, которая содержит в себе продукты магазина разделяется на 2 элемента. 1 элемент это категории, которые запрашиваются с базы, а 2 элементы это продукты с фильтрацией, код на изображении 31.



Изображение 31 Код страницы с продуктом

Страница с продуктом содержит в себе всякого рода фильтрации при запросе данных. Ответ, то есть массив продуктов, перемещается в redux и при нажатии какой-то фильтрации запускается функция. Функция создаёт новый массив на основе того, что лежит в redux и фильтрует его. Создавая новую переменную состояния туда прокидывается отсортированный массив, пример изображение 32.



Изображение 32 Код фильтрации продукта

При покупки товара делается запрос на сервер и в определённую корзину пользователя добавляется товар. Корзина пользователя привязана к нему и не может быть две корзины у пользователя. Эту функцию реализовывает запрос, который изображён на изображении 33.



Изображение 33 Перемещение товара в корзину

Корзина содержит в себе некоторые функции. Во 1 корзина проверяет зарегистрирован ли пользователь и если да, то проверяет пустой ли массив с корзиной, который присылается с сервер изображение 34. Если массив не пустой циклом выводит все на страницу.



Изображение 34 код корзины

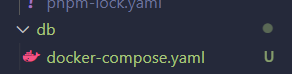
2.2 Разработка backend составляющей сайта

Серверная часть сайта представляет из себя написанное на JavaScript приложение на платформе NodeJS. Создается папка server в которой также создается файл server.js. Открыв консоль прописывается команда для установки необходимых библиотек для веб-сервера «npm install nodemon bcrypt cors dotenv express jsonwebtoken pg pg-hstore sequelize». С помощью фрэймворка express можно создать удобную и мощную базу данных на nodejs.

2.3 Работа с Базой данных

Чтобы создать контейнер с базой данных, который в последствии можно будет развернуть и запустить. На сайте <https://www.docker.com/> устанавливается программное обеспечение.

Для базы данных создается папка db, в ней создается файл docker-compose.yaml данный файл отвечает за запуск контейнера через docker в формате yaml похожи формат на json. Изображение 11.



Изображение 35 Создание контейнера с базой

Перейдя в файл прописывается версия сборки проекта в формате: название, двоеточие, текс в ковычках или вне их. После версии создается сервис который будет собирать один или несколько контейнеров через «servise:». Внутри сервиса объявляется контейнер «container\_name» отвечает за название контейнера. «ports» показывает на каком порту будет запускаться контейнер, он делится на внутренний и внешний. Внутренний контейнер служит внутри контейнера, внешний из вне контейнера. «restart: always» запускает контейнер даже после перезапуска компьютера. «environment» - переменные окружения контейнера. «POSTGRES\_DB» объявляет название базы, так как данная база будет сделана по образу базы PostgreSQL, то в начале добавляется её название. «POSTGRES\_PASSWORD» - пароль от базы данных, а «POSTGRES\_USER» - имя пользователя, по стандарту ставится root. Последним параметром следует «image» - образ базы, он загружается с сайта докер и по образу базы создаётся наша база. Полная структура на изображении 12.



Изображение 36 Структура docker-compose.yaml

В консоли powershell через команду «docker-compose up --build» создается контейнер по найстройкам из docker-compose. После поднятия контейнера можно будет выключить и включить его в программном обеспечении, изображение 13. В конченом итоге мы можем получить доступ к базе данных через ORM.



Изображение 37 Контейнер в приложении

Заключение

Опираясь на выше перечисленное можно сказать, что сайт полностью готов к приходу клиентов. Сайт функционирует в штатном режиме. Теперь пользователи имеют удобный интерфейс для покупки товаров в интернет магазине. Данный сайт даст возможность получать больше клиентов и интересовать большой круг покупателей. Любой бизнес должен иметь свой сайт или интернет магазин. Любой бизнес должен давать возможность пользователям ознакомиться с собой на дистанционном расстоянии, то есть не выходя из дома. Завлечение клиентов в интернете принесёт выгоду, так как почти все люди в мире пользуются интернетом. Поставленные задачи также были выполнены.

Была изучена теория и инструменты, которые позволили создать сайт быстро и удобно как для программиста, так и для пользователя. Благодаря заранее выученным инструментам, можно было легко применить их на практике, например тот же «gsap».

Техническое задание помогло справиться с задачей более легко. Когда программист понимает, что от него хотят на прямом примере, то сайт делается более легко и непринуждённо. Техническое задание ограничило воображение оставив важные элементы понятными и простыми. Готовый дизайн позволил программисту не задумываться о создании его, что увеличило производительность.

Созданные схемы объяснили программисту как должен работать сайт с сервером, каким образом идёт передача и запись данных в базу данных. Схемы помогают наглядно понять принцип работы, это уменьшает количество времени на раздумье и проектирование.

Была написана frontend составляющая сайта. Одна из самых главных и необходимых частей. Пользователь теперь может удобно перемещаться по сайту, удобно получать доступ к товарам, просматривать их. Легко получить адрес магазинов при помощи встроенной карте, а также легко и удобно совершить покупку.

Backend составляющая сайта была сделана и продумана. Теперь сайт может отправлять и получать товары только через наш сервер при помощи «cors». С помощью «Sequelize» было настроено общение с базой данных, что позволило сохранять и записывать товары и пользователей. Также серверная часть позволило и сделало регистрацию для пользователей.

Создали переносную базу данных. Поместив базу данных в контейнер была данна возможность переноса базы данных в любое место, будь то облачные сервера или другой компьютер. Разворачивание базы будет быстрым при помощи «Docker».

Список использованной литературы

1. ГОСТ 34.601-90 - Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.
2. ГОСТ 24.103-84. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Автоматизированные системы управления. Общие положения.
3. ГОСТ 24.104-85 Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Автоматизированные системы управления. Общие требования.
4. ГОСТ 34.602-89 - Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы.
5. ГОСТ 7.32-2001 - Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления.
6. ГОСТ 34.003-90 – Информационная технология. Комплекс стандартов на автоматизированные системы. Термины и определения.
7. Лысенко, В.А. Системное проектирование информационных систем с вебинтерфейсом: моногр. / В.А. Лысенко, М.И. Корзина, И.В. Бачурин; Сев. (Арктич.) федер. ун-т им. М.В. Ломоносова. – Архангельск: САФУ, 2016. – 128 с.
8. Прайс, Джейсон Oracle Database 11g: SQL. Операторы SQL и программы PL/SQL / Джейсон Прайс. - М.: ЛОРИ, 2018. - 660 c.
9. Карвин Билл. Программирование баз данных SQL. Типичные ошибки и их устранение / Билл Карвин. - М.: Рид Групп, 2018. - 336 c.
10. Коннор МакДональд, Хаим Кац. Oracle PL/SQL для профессионалов. Практические решения, 2019. - 560 с. 51 40
11. Администрирование баз данных Oracle в операционной системе UNIX. - М.: СПб: ЦКТиП Газпром, 2018. - 300 c.
12. Робин Никсон. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript, 2019. – 816 с.
13. Глушаков С.В. Базы данных / С.В. Глушаков, Д.В. Ломотько. - М.: Харьков: Фолио, 2018. - 504 c.
14. Кузнецов С.Д. Базы данных, 2020. – 256 с. Mike Shema. Hacking Web Apps: Detecting and Preventing Web Application Security Problems. — Newnes, 2012.
15. Monsur Hossain. CORS in Action: Creating and consuming cross-origin APIs. — Manning Publications Company, 2014.
16. Бэн Фрэйн. Глава 7. Использование SVG для достижения независимости от разрешения. Разработка сайтов для любых браузеров и устройств — 2 издание. — Санкт-Петербург: П, 2017.
17. Дакетт, Джон. Javascript и jQuery. Интерактивная веб-разработка. — М., 2017.
18. Дональд Кнут. Искусство программирования, 2001.
19. Итан Браун. Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript — Питер, 2017
20. Каскиаро М., Маммино Л. Шаблоны проектирования Node.js. — 2017. 24. Коншин, К. (2018). Следующий. js Краткое руководство: рендеринг на стороне сервера выполнен правильно.
21. Мардан Азат. React быстро. Веб-приложения на React, JSX, Redux и GraphQL. — СПб.: «Питер»,2019