**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Владимирский государственный университет**

**имени Александра Григорьевича и Николая Григорьевича Столетовых»**

**(ВлГУ)**

**КИТП**

**ДИПЛОМНЫЙ ПРОЕКТ**

Студент \_Филатов Никита Александрович\_

Институт \_Информационных технологий и радиоэлектроники\_

Направление \_09.02.07 – Информационные системы и программирование

**Тема дипломного проекта**

Разработка веб-приложения интернет-магазина обуви

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_В. В. Данилов\_

подпись инициалы, фамилия

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Н. А. Филатов\_

подпись инициалы, фамилия

**Допустить дипломный проект к защите**

**в государственной экзаменационной комиссии**

Заведующий кафедрой ИСПИ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_И.Е. Жигалов\_\_\_

подпись инициалы, фамилия

«\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_\_\_ г.

**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Владимирский государственный университет**

**имени Александра Григорьевича и Николая Григорьевича Столетовых»**

**(ВлГУ)**

**КИТП**

УТВЕРЖДАЮ:

Заведующий кафедрой ИСПИ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.Е. Жигалов

(подпись) (ФИО)

«\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

**НА ДИПЛОМНЫЙ ПРОЕКТ**

Студенту\_\_\_\_ гр. ИСПспк-218\_\_Филатову Никите Александровичу\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Тема дипломного проекта \_\_\_Разработка веб-приложения интернет-магазина обуви \_\_\_\_\_\_

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

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

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

утверждена приказом по университету № \_\_74/4\_\_ от \_\_16.02.2022\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Срок сдачи студентом законченного Дипломного проекта на кафедру \_\_\_\_10.06.2022\_\_\_\_\_

3. Исходные данные к Дипломному проекту \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

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

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

4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов)

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ВВЕДЕНИЕ

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Описание предметной области

1.2 Обзор и анализ аналогов

1.3 Цели и задачи дипломного проектирования

2 ПРОЕКТИРОВАНИЕ ИС

2.1 Техническое задание на создание ИС

2.2 Выбор средств разработки ИС

2.3 План выполнения проектных работ

2.4 Проектирование структуры ИС

3 РЕАЛИЗАЦИЯ ПРОТОТИПА ИС

3.1 Интерфейс разрабатываемой ИС

3.2 Разработка серверной части ИС

3.3 Разработка клиентской части ИС

4 РАЗРАБОТКА ТЕХНИЧЕСКОЙ ДОКУМЕНТАЦИИ

4.1 Разработка инструкции пользователя ИС

4.2 Разработка инструкции администратора ИС

4.3 Разработка графической части проекта ИС

5 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ПРОЕКТА

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

РИЛОЖЕНИЕ А Пользовательский интерфейс

ПРИЛОЖЕНИЕ Б Листинг кода реализации серверной части

ПРИЛОЖЕНИЕ В Листинг кода реализации клиентской части

5. Перечень графического материала (с указанием обязательных чертежей)

1. Диаграмма вариантов использования (прецедентов) (UML).

2. Диаграмма видов деятельности (UML).

3. Диаграмма компонентов системы (UML).

4. ER-модель базы данных (если имеет место).

5. Схема работы системы (ГОСТ).

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

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

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

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

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

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

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

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

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

6. Консультанты по работе (с указанием относящихся к ним разделов работы)

6.1. От выпускающей кафедры (общее методическое руководство) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. преп. каф. ИСПИ Шамышева О.Н.

6.2. Нормоконтроль \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ преподаватель каф. ИСПИ Маркова В.Б.

Дата выдачи задания «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_ г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

подпись ФИО

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись студента

АННОТАЦИЯ

В данной работе представлена реализация веб-приложения для интернет-магазина обуви. На сайте есть возможность заказать интересующий пользователя товар.

Ключевые слова: интернет-магазин, автоматизированная информационная система, база данных.

Дипломный проект содержит 85 страниц, 8 рисунков, 13 таблиц, 3 приложения, 6 источников литературы.

ABSTRACT

This paper presents the implementation of a web application for an online shoe store. On the website there is an opportunity to order the product of interest to the user.

Keywords: online store, automated information system, database.

In this diploma thesis is contained 85 pages, 8 illustrations, 13 tables, 3 appendices, 6 literature sources.

СОДЕРЖАНИЕ

[ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 7](#_Toc105522924)

[Введение 8](#_Toc105522925)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 11](#_Toc105522926)

[1.1 Описание предметной области 11](#_Toc105522927)

[1.2 Обзор и анализ аналогов 14](#_Toc105522928)

[1.3 Цели и задачи дипломного проектирования 18](#_Toc105522932)

[2 ПРОЕКТИРОВАНИЕ Информационной системы 19](#_Toc105522933)

[2.1 Техническое задание на создание информационной системы 19](#_Toc105522934)

[2.2 Выбор средств разработки информационной системы 25](#_Toc105522939)

[2.3 План выполнения проектных работ 30](#_Toc105522943)

[2.4 Проектирование структуры информационной системы 32](#_Toc105522944)

[3 РЕАЛИЗАЦИЯ ПРОТОТИПА Информационной системы 38](#_Toc105522949)

[3.1 Интерфейс разрабатываемой информационной системы 38](#_Toc105522950)

[3.2 Разработка серверной части информационной системы 39](#_Toc105522951)

[3.3 Разработка клиентской части информационной системы 40](#_Toc105522952)

[4 РАЗРАБОТКА ТЕХНИЧЕСКОЙ ДОКУМЕНТАЦИИ 41](#_Toc105522953)

[4.1 Разработка инструкции пользователя информационной системы 41](#_Toc105522954)

[4.2 Разработка инструкции администратора информационной системы 41](#_Toc105522955)

[5 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ПРОЕКТА 43](#_Toc105522956)

[5.1 Расчет стоимости разработки системы 43](#_Toc105522957)

[5.2 Расчет стоимости выполнения процесса до автоматизации 48](#_Toc105522958)

[5.2 Расчет стоимости выполнения системы после автоматизации 52](#_Toc105522959)

[5.3 Расчет показателей эффективности системы 55](#_Toc105522960)

[Заключение 57](#_Toc105522961)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 58](#_Toc105522962)

[ПРИЛОЖЕНИЕ А Пользовательский интерфейс 59](#_Toc105522963)

[ПРИЛОЖЕНИЕ Б Листинг кода реализации серверной части 66](#_Toc105522965)

[ПРИЛОЖЕНИЕ В Листинг кода реализации клиентской части 76](#_Toc105522967)

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

Интернет-магазин – это интерактивный сайт, рекламирующий товары или услуги в интернете. Позволяет пользователям сформировать заказ на покупку, выбрать способ оплаты и доставки заказа в сети Интернет.

Автоматизированная информационная система (АИС) – это комплекс программных, технических, информационных, лингвистических, организационно - технологических средств и персонала, предназначенный для решения задач справочно - информационного обслуживания и (или) информационного обеспечения пользователей.

База данных – совокупность данных, организованных в соответствии с концептуальной структурой, описывающей характеристики этих данных и взаимоотношения между ними, которая поддерживает одну или более областей применения.

Введение

В настоящее время большое количество людей очень часто обращаются к пространству Интернет с различными интересующими их вопросами. Часто для общения с друзьями и родственниками, с коллегами по работе. Интернет обеспечивает практически бесперебойную возможность находиться всегда на связи и быть в курсе всех новостей. И это очень удобно особенно в бизнесе, ведь осведомленность играет большую роль в дальнейшем существовании и развитии бизнеса. Интернет развивается с каждым годом все сильнее, оказывает колоссальное влияние на многих людей. Вследствие того, что глобальная сеть становится все более развитой средой, обычным людям становится необходимым знание навыков работы с Интернетом, умение пользоваться электронной почтой, поисковыми системами, телеконференциями, и многими другими возможностями сети Интернет.

В настоящее время Интернет становиться все более развитой средой для осуществления коммуникаций с потребителями. В тоже время, существенным является и тот факт, что Интернет становиться удобной и достаточно дешевой «торговой площадкой». Все большее количество фирм старается представить свою продукцию в on-line среде. С развитием Интернета развивается и само предложение. Теперь люди могут не только получать интересующую их информацию, но и совершать покупки. При этом с помощью Интернет-магазинов можно приобретать товары совершенно разных категорий, как элементарные потребительские, так и высокотехнологичные.

Такое положение вещей обусловлено, во-первых, постоянным, стабильным ростом аудитории пользователей глобальной сети; во-вторых, нехваткой времени у людей, их сильную занятость и загруженность на работе. Ведь основную часть аудитории сети Интернет составляют люди, работающие в офисах и проводящие большую часть времени за компьютером. Как правило, их образ жизни не позволяет им тратить большое количество времени на походы по обычным магазинам в поисках именно того, что им нужно. Возможность заказать интересующий продукт в Интернете является для них действительно выходом из ситуации.

Ниже приведены некоторые преимущества систем электронной коммерции:

1. информация о товарах и услугах оборачивается быстрее. Фактически, вы получаете дополнительный коммуникационный канал, открытый 24/7 (оборот, вошедший в английский язык в связи с развитием электронной коммерции и обозначающий работу 24 часа в сутки, 7 дней в неделю). Вы более доступны для клиента географически и во времени, к тому же он получает новое средство поиска и работы с информацией;
2. быстрее оборачивается внутренняя информация (документы, служебная переписка, принятие и утверждение решений и пр.). Расширенная возможность для контроля исполнения. Иначе говоря, ускоряются все бизнес-процессы за счет доступности и скорости передачи информации

Интернет предлагает новые сервисы для клиентов – например, отслеживание продвижения заказа. Наличие дополнительных сервисов создает конкурентное преимущество и приводит к вам новых посетителей.

Поскольку Интернет является технологической оболочкой, он позволяет собирать важную информацию о ваших клиентах. Кроме того, он позволяет использовать маркетинговый инструментарий – опросы, рассылки и т.п. оперативно и без дополнительных затрат.

Системы электронной коммерции помогут сэкономить на персонале и на аренде площадей для торговых залов.

Таким образом, развитие спроса породило предложение приобретать интересующие товары в Интернет-магазинах, а также разрабатывать сами Интернет-магазины. Более того, с каждым годом развитие торговли через всемирную сеть становится более обширным. Товары и услуги, предлагаемые в обычных магазинах, пользуются успехом в Интернет-магазинах.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Описание предметной области

Сейчас каждый желающий может создать веб-приложение, интернет-магазин или простой информационный сайт. Для этого не нужно прикладывать много усилий, так как существует множество программ, которые делают большую часть работы самостоятельно.

Способов разработки веб-приложений много и одни из них – создание веб-приложения с нуля. Почти для каждого предприятия нужен сайт, который облегчит взаимодействия с клиентом.

Веб-приложение для магазина позволит клиенту приобрести нужный товар онлайн.

Несмотря на то, что основная целевая группа будет пользоваться веб-приложением со смартфонов, стоит предусмотреть возможность удобной работы на сайте с компьютеров. Поэтому верстка должна быть адаптивна и корректно открываться со следующих устройств:

1. смартфоны;
2. компьютеры

Следует обратить внимание, что дизайн сайта должен быть удобен и понятен для использования, соответствовать современным тенденциям.

На сайте должны быть реализованы следующие страницы:

1. главная страница, предоставляющая возможности авторизации и регистрации пользователей и сортировки товара;
2. страница с информацией о каждом товаре;
3. личные кабинеты администратора и авторизованных пользователей;
4. панель администратора, для добавления/удаления товара;
5. корзина

Логотип должен быть реализован в соответствии с требованиями:

1. в логотипе должны быть использованы основные цвета сайта;
2. логотип представляет собой изображение;
3. разработанный логотип должен быть сохранен в корневом каталоге в директории img со следующим именем – logo.png

Портал должен поддерживать возможности 3 типов пользователей:

1. гость;
2. авторизованный пользователь;
3. администратор

Возможности гостя:

1. вход в личный кабинет по логину и паролю;
2. регистрация;
3. просмотр главной страницы;
4. просмотр информации о товаре

Возможности авторизованного пользователя:

1. регистрация, авторизация, выход;
2. доступ к личному кабинету;
3. просмотр информации о товаре;
4. добавление товара в корзину;
5. добавление товара в избранное

Разделы администратора:

1. добавление/удаление товара;
2. добавление/удаление бренда;
3. добавление/удаление типа товара

Авторизация пользователя и администратора:

1. при вводе неправильной пары логин-пароль пользователю отображается сообщение об ошибке;
2. при успешной авторизации пользователь должен перенаправляться в личный кабинет с возможностью просмотра избранных товаров;
3. личный кабинет доступен только авторизованному пользователю;
4. панель управления сайтом доступна только администратору

Регистрация пользователя:

1. вся валидация должна работать без перезагрузки страницы, все поля
2. обязательные для заполнения;
3. email – валидный формат email-адрес – проверка на стороне клиента;
4. пароль
5. в случае несоответствия любым требованиям выводится анимированное сообщение об ошибке;
6. все авторизованные пользователи могут выйти из профиля по нажатию на ссылку/кнопку «выход»

1.2 Обзор и анализ аналогов

1.2.1 Аналоги

Официальный сайт интернет-магазина «Нэфэртити».

Официальный сайт интернет-магазина «ИнтерМода».

Официальный сайт интернет-магазина «Рефлекс».

1.2.2 Описание аналогов

Официальный сайт интернет-магазина «Нэфэртити».

Сайт привлекает простым дизайном, понятным интерфейсом и удобным поиском товар. Оформление сайта соответствует тематике интернет-магазина. Сайт организован так, что случайный посетитель легко может найти товар, который его интересует.

Гостю доступны:

1. просмотр информации на сайте, таких разделов как:
   1. категории товара;
   2. страница о товаре;
   3. отзывы;
   4. акции;
   5. страница о нас
2. авторизация;
3. регистрация;
4. оформление заказа

Зарегистрированному пользователю доступны:

1. личный кабинет;
2. персональные данные;
3. возможность оставить отзыв;
4. возможность добавить товар в избранное

Официальный сайт интернет-магазина «ИнтерМода».

Сайт привлекает понятным интерфейсом. Оформление сайта соответствует тематике интернет-магазина.

Дизайн не соответствует современным тенденциям, функционал сайта сильно ограничен.

Пользователю доступны:

1. просмотр информации на сайте, таких разделов как:
   1. о компании;
   2. каталог товаров;
   3. страница о товаре;
   4. контакты
2. добавления товара в корзину;
3. возможность оставить отзыв

Официальный сайт интернет-магазина «Рефлекс».

Сайт имеет единый стиль. Общие элементы – логотип, меню, одна цветовая гамма и набор шрифтов на каждой странице. Трудностей восприятия текстов на выбранном фоне нет. На сайте удобная навигации.

Много лишней информации, устаревший дизайн, который не соответствует современным тенденциям.

Пользователю доступны:

1. просмотр информации на сайте, таких разделов как:
   1. каталог;
   2. акции и скидки;
   3. страница о товаре;
   4. отзывы;
   5. контакты
2. регистрация/авторизация;
3. возможность оставить отзыв

1.2.3 Сравнение аналогов

Сравнение аналогов проводится по критериям:

1. актуальность и частота обновления информации;
2. удобство восприятия информации;
3. удобство навигации;
4. наличие личного кабинета у пользователя;
5. скорость действий на сайте;
6. возможность добавление товара в корзину

Сравнение аналогов представлено в виде таблице 1 с разбаловкой за каждый критерий и баллов у каждого аналога за каждый критерий.

Таблица 1 – Сравнительный анализ приложений-аналогов по критериям

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерий | «Нэфэртити» | «ИнтерМода» | «Рефлекс» | Разрабатываемая система (ожидаемая оценка) |
| Актуальность и частота обновления информации | 7 | 5 | 6 | 10 |
| Удобство восприятия информации | 7 | 3 | 4 | 9 |

*Продолжение таблицы …*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерий | «Нэфэртити» | «ИнтерМода» | «Рефлекс» | Разрабатываемая система (ожидаемая оценка) |
| Удобство навигации | 8 | 4 | 5 | 9 |
| Наличие личного кабинета у пользователя | 10 | 10 | 10 | 10 |
| Скорость загрузки страниц | 9 | 9 | 9 | 9 |
| Возможность добавления товара в корзину | 9 | 9 | 9 | 10 |

К положительным относятся:

1. формирование бренда;
2. продвижение;
3. дополнительный маркетинговый канал;
4. быстрое информирование аудитории;
5. коммуникации с клиентами;
6. бонусы в конкурентной борьбе;
7. удобство быстрой подачи заявки в автошколу, не выходя из дома

К отрицательным:

1. неудобно в использовании;
2. устаревший дизайн сайтов

В итоге из анализа информационных сайтов рассматриваемых интернет-магазинов можно выделить как положительные, так и отрицательные стороны. Было принято решение о разработке собственного интернет-магазина с учётом положительных и исправлением отрицательных особенностей

1.3 Цели и задачи дипломного проектирования

Целями данной работы являются:

1. построение моделей ис;
2. проектирование ис;
3. выбор технологий;
4. реализация ис;
5. тестирование ис;
6. анализ экономической эффективности;
7. программная реализация информационной системы

Для достижения поставленных целей необходимо решить следующие задачи:

1. ознакомиться с организацией учебного процесса на кафедре учебного заведения; поставить задачу по автоматизации работы;
2. разработать модели предметной области и информационного обеспечения;
3. разработать базу данных системы, интерфейс всех модулей и функционал

2 ПРОЕКТИРОВАНИЕ Информационной системы

2.1 Техническое задание на создание информационной системы

2.1.1 Общие положения

Цели создания интернет-магазина:

1. минимизация издержек на работу с клиентами за счет функционала интернет-магазина;
2. создание площадки для успешного продвижения и продажи товаров заказчика

Задачи, которые должен выполнять интернет-магазин:

1. возможность покупки товаров покупателем;
2. полная автоматизация процесса покупки товаров покупателем;
3. минимизация телефонных/почтовых контактов с покупателем за счет удобной подачи информации в интернет-магазине;
4. постоянная возможность получения актуальных значений товарных остатков в интернет-магазине;
5. автоматическое формирование платежных документов и прайс-листов;
6. автоматизация процесса оплаты товаров покупателем;
7. повышение лояльности клиентов за счет высокого уровня обслуживания как при первом контакте, так и впоследствии

Посетитель – физическое лицо, выполняющее с помощью браузера запрос или передачу информации к Сайту и получающее информацию с Сайта в рамках открытой области Сайта.

Администратор – пользователь, сотрудник Компании, имеющий полный доступ ко всем функциям Сайта, в том числе к редактированию сайта.

Описание сайта: Сайт должен быть удобен для пользователя, простой в управлении, информативный. Должен ассоциироваться с высоким уровнем сервиса и заботы о клиенте. Сайт предназначен для функционирования в компьютерной сети Интернет, с целью предоставления возможности пользователям Сети просмотра информации о перечне поставляемых товаров. Сайт должен позволять производить заказ товаров из каталога.

Сайт состоит из следующих функциональных областей (разделов):

1. открытая область;
2. область управления сайтом

Администратору Сайта доступны все функции Сайта.

Все функциональные области, в т. ч. их части, существуют в единой информационной среде: единый сервер, единая БД, единые программные модули, сегментированные по функциональным областям.

Должна быть реализована возможность масштабируемости Сайта – добавления/удаления новых страниц Сайта без ущерба действующей функциональности и дизайну Сайта.

2.1.2 Структура интернет-магазина

Структура Интернет-магазина предусматривает ведение следующих страниц:

1. главная;
2. каталог;
3. о товаре;
4. где нас найти;
5. регистрация;
6. авторизация;
7. личный кабинет;
8. корзина;
9. панель администратора

2.1.3 Детализация структуры

На главной странице (переход через логотип) – будут находиться следующие элементы:

1. хэдер интернет-магазина:
   1. логотип;
   2. название сайта;
   3. ссылка на страницу «Каталог»;
   4. ссылка на страницу «Где нас найти»;
   5. кнопки Авторизации и Регистрации;
   6. блок корзины (Для авторизированных пользователей);
   7. ссылка на страницу «Личный кабинет» (Для авторизированных пользователей);
   8. ссылка на страницу «Панель администратора» (Для администратора);
   9. кнопка «Выйти» (Для авторизированных пользователей);
2. контентная зона:
   1. ротатор Баннеров: Кнопка «В Каталог»;
   2. баннер с товаром;
   3. баннер с товаром;
   4. панель с преимуществами интернет-магазина;
3. футер сайта:
   1. контактная информация;
   2. ссылки на социальные сети;
   3. ссылки на мобильные приложения;
   4. копирайты и разработчик

Страница «Каталог»:

1. вывод всех товаров;
2. список брендов, представленных на сайте (дополнительное горизонтальное меню);
3. список типов товаров, представленных на сайте (дополнительное вертикальное меню)

Товары, добавленные на сайте, будут представлены в виде плитки. Каждая единица товара в данном разделе будет отображаться в виде блока: название и фото со ссылкой для перехода к детальному описанию товара, ценой, кнопка «в корзину» для возможности оформления заказа, кнопка «в избранное».

Страница «Где нас найти»:

1. контактная информация;
2. ссылки на социальные сети;
3. карта с расположением организации

Страница «О товаре»:

1. изображение товара;
2. название товара;
3. описание товара;
4. цена товара;
5. кнопка «добавить в корзину»;
6. характеристики товара

Страница «Регистрация»:

1. форма регистрации нового пользователя состоит из:
   1. поле ввода имени пользователя;
   2. поле ввода фамилии пользователя;
   3. поле ввода адреса электронной почты;
   4. поле ввода пароля;
   5. кнопка «Зарегистрироваться»;
2. ссылка на форму авторизации

Страница «Авторизация»:

1. форма авторизации пользователя состоит из:
   1. поле ввода адреса электронной почты;
   2. поле ввода пароля;
   3. кнопка «Войти»;
2. ссылка на форму регистрации нового пользователя

Страница «Личный кабинет»:

1. вывод данных о пользователе;
2. вывод избранных товаров

Страница «Корзина»:

1. вывод товаров, добавленных в корзину;
2. вывод общей стоимости товаров;
3. кнопка оформить заказ

Страница «Панель администратора»:

1. модальное окно «добавить тип»:
   1. поле ввода названия типа;
   2. кнопка закрыть;
   3. кнопка добавить;
2. модальное окно «добавить бренд»:
   1. поле ввода названия бренда;
   2. кнопка закрыть;
   3. кнопка добавить;
3. модальное окно «добавить товар»:
   1. выпадающий список «выберите тип»;
   2. выпадающий список «выберите бренд»;
   3. поле ввода названия товара;
   4. поле ввода цены товара;
   5. выбор изображения;
   6. поле ввода описания товара;
   7. поле ввода характеристик товара;
   8. кнопка закрыть;
   9. кнопка добавить

2.1.4 Подробное описание функционала страницы категории товаров

При щелчке на кнопку карточку товара должен происходить переход на страницу с товара.

При нажатии на кнопку «Добавить в корзину» выбранный товар должен появляться в «Корзине сайта», которая отображена на всех страницах сайта. В «Корзине сайта» должны показываться все выбранные товары.

При нажатии на кнопку «Добавить в избранное» выбранный товар должен появляться в «Личном кабинете». В «Личном кабинете» должны показываться все выбранные товары.

После нажатия кнопки «Купить» товар появляется в «Корзине товаров». При переходе на страницу корзины товаров пользователь видит все выбранные товары, цены. После этого покупатель может нажать кнопку «Заказать» и указав свои контактные данные (Телефон, Адрес доставки, комментарии к заказу).

Требования к безопасности и защите информации Сайта:

1. информация сайта должна быть расположена в закрытой БД, доступ к которой разрешен только из программных модулей Сайта и только с использованием пароля доступа;
2. доступ к закрытым областям сайта разрешен только пользователям, прошедшим процедуру регистрации

2.2 Выбор средств разработки информационной системы

Интернет-магазин реализовывается как на стороне сервера, так и на стороне клиента. В первом случае используются серверные сценарии, построенные на технологии Node.js и систем управления базами данных MySQL. Во втором случае – React.

MySQL – небольшой, компактный многопоточный сервер баз данных. MySQL характеризуется большой скоростью, ошибкоустойчивостью и легкостью в использовании. MySQL был разработан компанией TcX для внутренних нужд, которые заключались в быстрой обработке очень больших баз данных. Ядро, на котором сформирован MySQL – набор подпрограмм, которые использовались в высокотребовательном окружении много лет. MySQL является идеальным решением для малых и средних приложений. Исходные тексты сервера компилируются на множестве платформ. Наиболее полно возможности сервера проявляются на Unix-серверах, где есть поддержка многопоточности, что дает значительный прирост производительности. В варианте под Windows, MySQL может запускаться как сервис Windows NT или как обычный процесс на Windows 95/98. В то время, как MySQL все еще находится в разработке, это уже предоставляет богатый и полезный функциональный набор. MySQL-сервер является бесплатным для некоммерческого использования, является относительно небольшой и быстрой реляционной СУБД основанной на традициях языка запросов SQL в стандарте ANSI 92, и кроме этого имеет множество расширений к этому стандарту, которых нет ни в одной другой СУБД.

JavaScript является относительно новым языком программирования и имеет преимущества от улучшения дизайна языка по сравнению с другими традиционными языками для веб-серверов (например, Python, PHP, и т.д.). Многие другие новые и популярные языки компилируются/конвертируются в JavaScript, поэтому вы можете также использовать CoffeeScript, ClosureScript, Scala, LiveScript, etc.

Node (или более формально Node.js) – кросплатформенная среда исполнения с открытым исходным кодом, которая позволяет разработчикам создавать всевозможные серверные инструменты и приложения используя язык JavaScript. Среда исполнения предназначена для использования вне контекста браузера (т.е. выполняется непосредственно на компьютере или на серверной ОС). Таким образом, среда исключает API-интерфейсы JavaScript для браузера и добавляет поддержку более традиционных OS API-интерфейсов, включая библиотеки HTTP и файловых систем.

Node был разработан для оптимизации пропускной способности и масштабируемости в веб-приложениях и очень хорошо справляется со многими распространёнными проблемами веб-разработки (например, веб-приложения реального времени).

Менеджер пакетов Node (NPM) обеспечивает доступ к сотням тысяч многоразовых пакетов. Он также имеет лучшее в своём классе разрешение зависимостей и может также использоваться для автоматизации большинства инструментов построения.

Он портативен, имеет версии для Microsoft Windows, OS X, Linux, Solaris, FreeBSD, OpenBSD, WebOS, и NonStop OS. Кроме того, он имеет хорошую поддержку среди многих хостинг-провайдеров, которые часто предоставляют конкретную инфраструктуру и документацию для размещения сайтов, работающих на Node.

React – это библиотека JavaScript с открытым кодом для создания внешних пользовательских интерфейсов. В отличие от других библиотек JavaScript, предоставляющих полноценную платформу приложений, React ориентируется исключительно на создание представлений приложений через инкапсулированные единицы (называются компонентами), которые сохраняют состояние и генерируют элементы пользовательского интерфейса. Вы можете разместить отдельный компонент на веб-странице или вложить иерархии компонентов для создания сложного пользовательского интерфейса.

Компоненты React часто пишутся на JavaScript и JSX (JavaScript XML), который является расширением JavaScript, очень похожим на HTML и включающим некоторые функции синтаксиса для оптимизированного выполнения распространенных задач, например регистрации обработчиков событий для элементов пользовательского интерфейса. Компонент React реализует метод отрисовки, который возвращает код JSX, представляющий пользовательский интерфейс компонента. В веб-приложении код JSX, возвращаемый компонентом, преобразуется в поддерживаемый браузером код HTML, который затем обрабатывается для отображения браузером.

2.2.1 Создание веб-приложения

Для создания сервера были использованы библиотеки Axios, React Bootstrap, React Router DOM, MobX, **Jwt decode**.

Axios – это широко известная JavaScript-библиотека. Она представляет собой HTTP-клиент, основанный на промисах и предназначенный для браузеров и для Node.js.

React Bootstrap – это React библиотека UI компонентов, которая основана на Bootstrap 4 и использует его стили, темы и т.д.

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

MobX – это автономная библиотека, для управления фронтенд-состоянием приложения. MobX обеспечивает консистентность и согласованность внутреннего состояния фронтенд-приложения, предоставляя удобные инструменты для его изменения.

**Jwt decode** – это небольшая библиотека браузера, которая помогает декодировать токен JWT, который кодируется Base64Url.

2.2.2 Создание веб-сервера

Для создания сервера были использованы библиотеки Express, mysql2, Sequelize, CORS, Dotenv, uuid, JSON Web Token, express-fileupload, bcrypt.

Библиотека Express – это простейшая, гибкая и оптимизированная платформа Node.js, упрощающая разработку веб-приложения, которое может обрабатывать различные типы запросов, например GET, PUT, POST и delete. Express поставляется с генератором приложений, который автоматически создает файловую архитектуру для приложения. Библиотека mysql2 использовалась для управления базой данных.

Библиотека Sequelize – это ORM-библиотека для приложений на Node.js, которая осуществляет сопоставление таблиц в БД и отношений между ними с классами.

Библиотека CORS – это механизм, который дает контролировать доступ к тегам на веб странице по сети. Механизм классифицируется на три разные категории доступа тегов:

1. запись из разных источников;
2. вставка из разных источников;
3. считывание из разных источников

Библиотека Dotenv – это модуль нулевой зависимости, который загружает переменные среды из файла .env

Библиотека JSON Web Token – это открытый стандарт (RFC 7519) для создания токенов доступа, основанный на формате JSON. Использовалась 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 своей личности.

Библиотека Bcrypt – это адаптивная криптографическая хеш-функция формирования ключа, используемая для защищенного хранения паролей.

На сервере осуществляется авторизация, регистрация, вывод списка всех товаров и информация о них (через бд).

2.2.3 Подключение веб-сервера к веб-приложению

При подключении сервера к клиентской части выполняется запрос с клиентской части на сервер, а сервер уже запрашивает данные у БД.

В данном проекте сервер совершает get и post запросы. В get запросах выводится информация о товарах.

В post запросах:

* + - 1. регистрируемся, передавая данные о почте и пароле;
      2. авторизуемся, передавая данные о почте и пароле

2.3 План выполнения проектных работ

В данной диаграмме используется график работы по проекту (Диаграмма Ганта). Срок исполнения использован в неделях:

1. 16 февраля – Получение задания дипломного проекта;
2. с 17 февраля по 21 февраля – Подготовка и утверждение предварительного плана работы (содержания);
3. с 22 февраля по 24 февраля – Подбор, изучение и анализ основных источников информации;
4. с 25 февраля по 11 марта – Разработка первого раздела работы;
5. с 14 марта по 16 марта – Корректировка первого раздела по замечаниям руководителя;
6. с 17 марта по 30 марта – Разработка второго раздела работы;
7. с 31 марта по 4 апреля – Корректировка второго раздела по замечаниям руководителя;
8. с 5 апреля по 18 апреля – Разработка третьего раздела работы;
9. с 19 апреля по 20 апреля – Корректировка третьего раздела по замечаниям руководителя;
10. с 21 апреля по 4 мая – Разработка четвертого раздела работы;
11. с 5 мая по 10 мая – Корректировка четвертого раздела по замечаниям руководителя;
12. с 11 мая по 24 мая – Разработка пятого раздела работы;
13. с 25 мая по 31 мая – Корректировка пятого раздела. Разработка введения и заключения, уточнение плана работы, согласование состава приложений;
14. 2 июня – Представление дипломного проекта научному руководителю;
15. 3 июня – Сдача готового дипломного проекта на нормоконтроль;
16. с 3 июня по 8 июня – Проверка работ на антиплагиат;
17. 8 июня – Предзащита дипломного проекта;
18. с 8 июня по 9 июня – Корректировка по результатам предзащиты;
19. с 5 июня по 10 июня – Получение отзыва руководителя;
20. с 5 июня по 10 июня – Получение рецензии на дипломный проект;
21. 10 июня – Сдача готового дипломного проекта на кафедру;
22. с 14 июня по 17 июня – Защита дипломного проекта в ГЭК

План выполнения проектных работ представлен на рисунке 1.

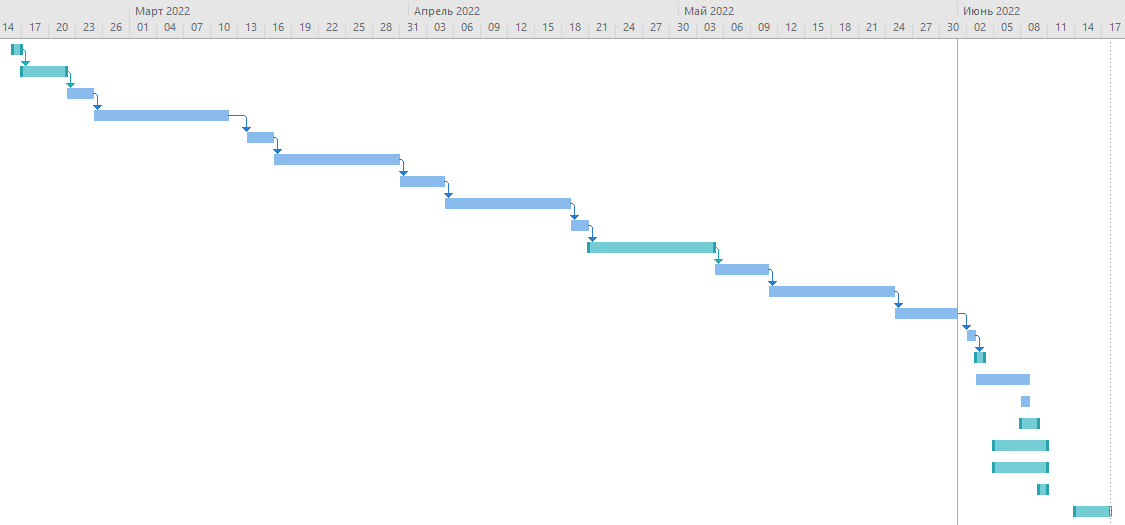


Рисунок 1 – План выполнения проектных работ

2.4 Проектирование структуры информационной системы

2.4.1 Диаграмма вариантов использования (прецедентов) (UML)

Для определения требований к системе была разработана диаграмма прецедентов. Работа прецедентов заключается в описании взаимодействий между пользователями системы и самой системой и предоставлении описания процесса ее функционирования.

В разработанном сайте будут предусмотрены следующие функции:

1. авторизация;
2. регистрация;
3. просмотр товаров;
4. добавление товаров в корзину;
5. просмотр информации на сайте;
6. оформление заказа;
7. добавление\удаление товаров (для администратора);
8. добавление\удаление брендов (для администратора);
9. добавление\удаление типов (для администратора)

Диаграмма прецедентов представлена на рисунке 2.

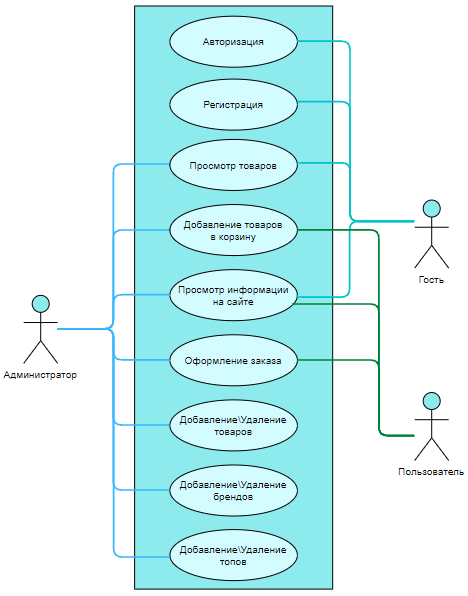


Рисунок 2 – Диаграмма прецедентов

2.4.2 Диаграмма видов деятельности

Основная цель данной информационной системы – это предоставление клиенту ассортимента товаров, и оформление заказа. Для этого требуется регистрация на сайте.

Деятельность авторизации пользователя представлена на рисунке 3.

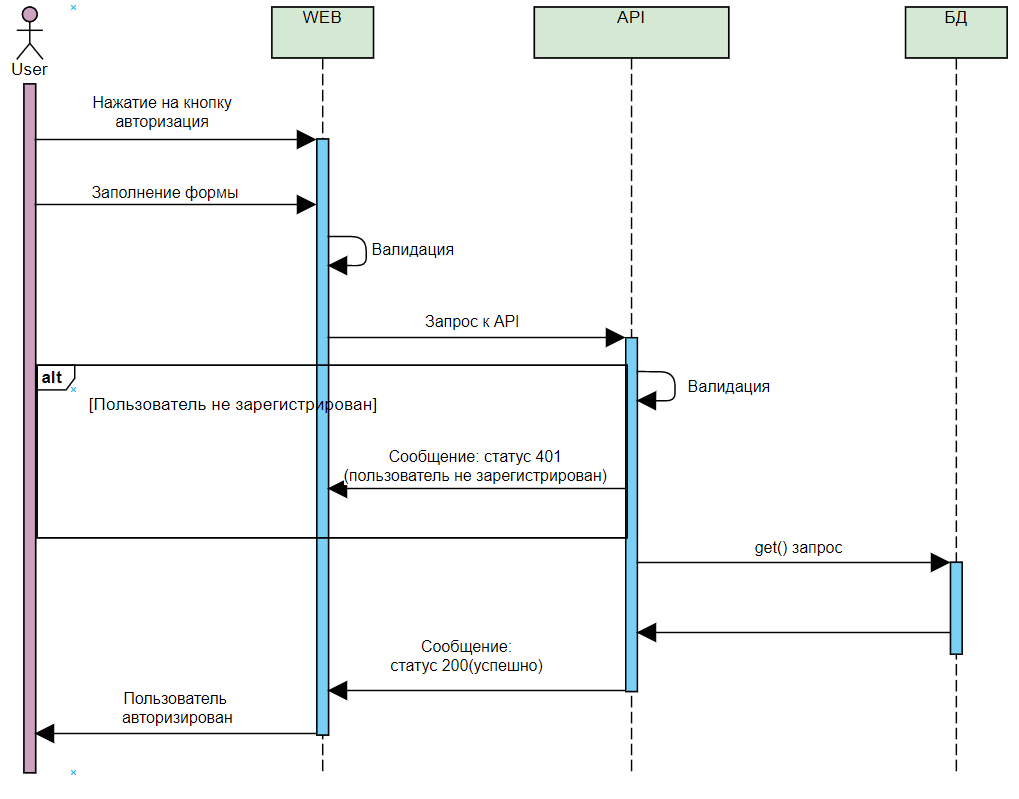


Рисунок 3 – Деятельность авторизации пользователя

2.4.3 Диаграмма компонентов системы

Данная система состоит из трёх основных компонентов – серверная часть, база данных и клиентская часть которые взаимодействуют между собой, в процессе выполнения какого-либо запроса пользователя.

Диаграмма компонентов системы представлена на рисунке 4.

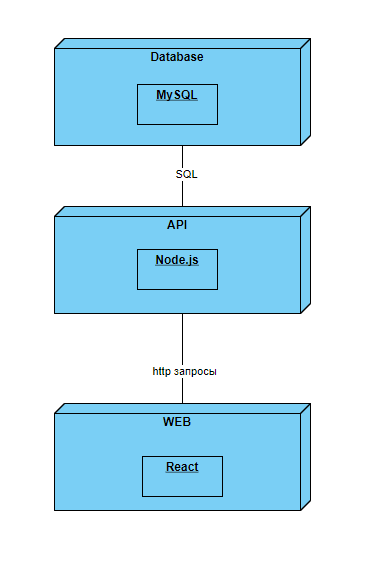


Рисунок 4 – Диаграмма компонентов системы

2.4.4 Схема работы системы

Схема - графическое представление определения, анализа или метода решения задачи, в котором используются символы для отображения операций, данных, потока, оборудования и т.д.

Схема работы системы предназначена для отображения управления операциями и потоком данных в системе.

Схема работы системы представлена на рисунке 5.

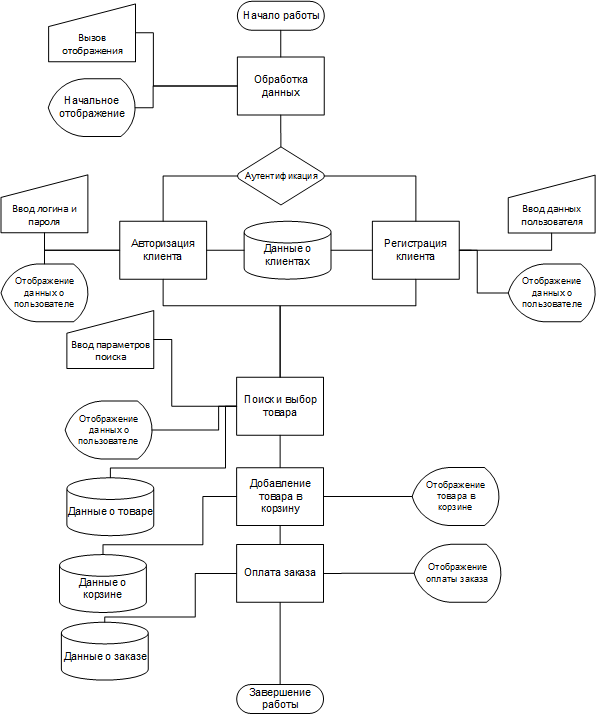


Рисунок 5 – Схема работы системы

3 РЕАЛИЗАЦИЯ ПРОТОТИПА Информационной системы

3.1 Интерфейс разрабатываемой информационной системы

Основными целями проектирования базы данных являются:

1. представление данных и связей между ними, необходимых для всех основных областей применения данного приложения и любых существующих групп его пользователей;
2. создание модели данных, способной поддерживать выполнение любых требуемых транзакций обработки данных;
3. разработка предварительного варианта проекта, структура которого позволяет удовлетворить все основные требования, предъявляемые к производительности системы – например, ко времени реакции системы

В основу проектирования БД должны быть положены представления конечных пользователей конкретной организации - концептуальные требования к системе. Именно конечный пользователь в своей работе принимает решения с учетом получаемой в результате доступа к базе данных информации. От оперативности и качества этой информации будет зависеть эффективность работы организации. Данные, помещаемые в базу данных, также предоставляет конечный пользователь.

При рассмотрении требований конечных пользователей необходимо принимать во внимание, что база данных должна удовлетворять актуальным информационным потребностям организации. Получаемая информация должна по структуре и содержанию соответствовать решаемым задачам.

ER-модель базы данных представлена на рисунке 6.

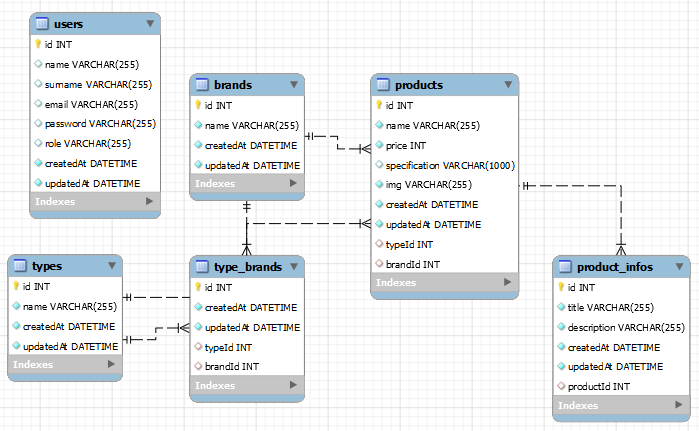


Рисунок 6 – ER-модель базы данных

3.2 Разработка серверной части информационной системы

Серверная часть – это основная программа, обеспечивающая работу сайта, разработка такой программы и подразумевает создание сайта. Сервер является связывающим звеном между клиентской частью и базой данных, в которой хранится вся пользовательская информация.

В качестве реализации API был использован Node.js с фреймворком Express.

Для начала была подключена БД, после этого были написаны сами запросы. Код представлен в Приложении Б.

3.3 Разработка клиентской части информационной системы

Клиентская составляющая – это то, что видит пользователь на экране браузера. Веб - проект строится из текста, изображений, ссылок, списков, форм ввода данных, таблиц и тому подобному.

Для реализации клиента был выбран React с его библиотеками.

Интерфейс выполнен из цветов, которые не отвлекают и не раздражают пользователя. Скриншоты с интерфейсом представлены в Приложении А. Код представлен в Приложении В.

4 РАЗРАБОТКА ТЕХНИЧЕСКОЙ ДОКУМЕНТАЦИИ

4.1 Разработка инструкции пользователя информационной системы

При открытии интернет-магазина загружается главная форма. Интерфейс пользователя прост, понятен и доступен. В основном все выполняемые операции соответствуют надписям на кнопках в формах. В верхней части экрана вы можете увидеть навигационную панель, которая поможет ориентироваться в программе.

При нажатии на кнопку выполняется определенное действие – открывается/закрывается форма, открывается страница, выполняется запрос, вносятся данные.

4.2 Разработка инструкции администратора информационной системы

База данных интернет-магазина написана в СУБД MySQL. Для запуска базы нужно открыть файл при помощи программы MySQL Workbanch.

При необходимости добавить товар необходимо авторизироваться на сайте под логином: admin паролем: admin, зайти в панель администратора, выбрать нужное действие из предложенных, заполнить все данные о товаре в модальном окне и нажать кнопку добавить товар.

При необходимости удалить товар необходимо авторизироваться на сайте под логином: admin паролем: admin, зайти в панель администратора, выбрать нужное действие из предложенных, выбрать нужный товар и нажать кнопку удалить товар.

При необходимости добавить бренд необходимо авторизироваться на сайте под логином: admin паролем: admin, зайти в панель администратора, выбрать нужное действие из предложенных, заполнить все данные о бренде в модальном окне и нажать кнопку добавить бренд.

При необходимости удалить бренд необходимо авторизироваться на сайте под логином: admin паролем: admin, зайти в панель администратора, выбрать нужное действие из предложенных, выбрать нужный бренд и нажать кнопку удалить бренд.

При необходимости добавить тип необходимо авторизироваться на сайте под логином: admin паролем: admin, зайти в панель администратора, выбрать нужное действие из предложенных, заполнить все данные о типе в модальном окне и нажать кнопку добавить тип.

При необходимости удалить тип необходимо авторизироваться на сайте под логином: admin паролем: admin, зайти в панель администратора, выбрать нужное действие из предложенных, выбрать нужный тип и нажать кнопку удалить тип.

При необходимости защиты информации от посторонних лиц, СУБД MySQL предоставляет возможность установить\изменить пароль базы данных. Для этого необходимо выбрать меню Настройки \ Пароль \ Задать пароль базы данных…

5 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ПРОЕКТА

Для расчета экономического эффекта от использования информационной системы. Метод функционально-стоимостного анализа (ФСА или ABC – **A**ctivity **B**ased **C**osting), который позволяет оценить реальную стоимость продукта или услуги вне зависимости от организационной структуры организации.

Для проведения функционально - стоимостного анализа необходимо:

1. определить процессы (действия), выполняемые на предприятии, и построить функциональные модели выделенных процессов;
2. посчитать количество выделенных действий за определённый интервал времени;
3. определить базовую стоимостную величину каждого процесса;
4. разложить косвенные затраты согласно базовой стоимости на процессы

5.1 Расчет стоимости разработки системы

Стоимость системы складывается из следующих составляющих:

1. затрат на заработную плату участникам процесса разработки системы;
2. затрат на расходные материалы;
3. расходов на амортизацию оборудования и нематериальных активов

Стоимость разработки системы автоматизации рассчитывается по следующей формуле:

, (5.1)

где Сис – стоимость разработки системы автоматизации, руб.;

З – затраты по заработной плате специалистам, задействованным в разработке систем, руб.;

М – затраты на расходные материалы, необходимые при разработке системы, руб.;

А – амортизация оборудования и нематериальных активов, используемых в процессе разработки системы, руб.

Для расчета затрат на выплату заработной платы специалистам, задействованным в разработке системы, целесообразно составить квалификационный план проекта разработки системы (таблица 2). Оклад сотрудников основан на средних данных о зарплате IT-специалистов в России. При расчете затрат работы специалиста в течение одного дня предполагается, что в среднем количество рабочих дней в месяце равно 21.

Таблица 2 – Квалификационный план проекта разработки системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование  специалиста | Оклад, руб/мес. | Заработная плата, руб/день | Ответственность в процессе разработки | Выполняемые функции |
| Системный администратор | 70000,00 | 3333,33  (15 рабочих дней) | Принимает систему  внутри фирмы, управляет  ходом работ | Разрабатывает требования  к системе, концепцию, план проекта, журнал хода проекта и глоссарий. |

*Продолжение таблицы …*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование  специалиста | Оклад, руб/мес. | Заработная плата, руб/день | Ответственность в процессе разработки | Выполняемые функции |
| Веб - разработчик | 40000,00 | 1905 (20 рабочих дней) | Разрабатывает  рабочую  документацию  к системе | Программирование компонентов системы.  Рецензирует требования к системе, концепцию и типовые  настройки. |
| Разработчик архитектуры | 30000,00 | 1428,5 (10 рабочих дней) | Разрабатывает  архитектуру  системы | Разрабатывает диаграммы  на этапах разработки технического и рабочего проектов.  Рецензирует глоссарий, требования к ИС, концепцию, типовые настройки системы. |

Разработка ИС в целом продлится 9 недель. Процесс создания ИС происходит под руководством менеджера проекта. Также, в качестве учета различного рода рисков можно дополнительно добавить одну неделю в общий срок разработки системы. Таким образом, общая длительность выполнения работ:

5 (кол-во рабочих дней в неделе) \* 9 = 45 дней

Тис = 45 рабочих дней.

Точные сроки разработки определить невозможно, так как сложность и риски, возникающие при создании ИС, зависят от множества вариативных факторов и задач, поэтому данные и все остальные расчеты имеют примерный характер и основаны на средних значениях.

Затраты по заработной плате рассчитываются следующим образом:

, (5.2)

где Ззп – заработная плата задействованных специалистов, руб.;

СВ – страховые взносы во внебюджетные фонды, руб.

, (5.3)

где n – количество задействованных специалистов, количество людей;

Оi – оклад i-го специалиста, руб.;

Д – количество рабочих дней в месяце, дни;

ti – время участия специалиста в проекте, определяется в соответствии с разработанным сетевым планом проектных работ, дни.

Страховые взносы во внебюджетные фонды составляют 30%:

СВ = Ззп\*0,3, (5.4)

где Ззп – заработная плата задействованных специалистов, руб.;

СВ – страховые взносы во внебюджетные фонды, руб.

Учитывая сроки разработки системы и квалификационный план выполнения проектных работ, риски, а также затраты на заработную плату задействованных специалистов составят:

Ззп = 3333,33\* 15+ 1905\* 20 + 1428,5\*10 = 102 380 (руб.)

С полученной суммы в страховой взнос во внебюджетные фонды будут произведены отчисления в размере:

СВ = 102 380 \* 0,3 = 30 714 (руб.)

В итоге затраты по заработной плате составят:

З = 102 380 + 30 714 = 133 094 (руб.)

Основными расходными материалами, задействованными при разработке ИС, являются электроэнергия, необходимая для работы компьютера, и бумага. В процессе разработки системы будет задействовано два компьютера (с учетом рабочих станций, использованных в качестве автономных серверов). Номинальная мощность компьютера составляет 200 Вт/ч.

Расчеты затрат на расходные материалы будем проводить по следующим формулам:

М = Б +Э, (5.5)

где М – стоимость затраченных расходных материалов;

Б – стоимость бумаги;

Э – стоимость электроэнергии, руб.

Стоимость бумаги и электроэнергии рассчитаем по следующим формулам:

Б = К\*Ц, (5.6)

Э = Р\*Ц\*Т, (5.7)

где К – количество бумаги, упаковки;

Р – мощность компьютера, КВт;

Ц – цена потребляемого ресурса, руб.

Результаты расчета затрат на расходные материалы сведены в таблицу 3.

Таблица 3 – Затраты на расходные материалы

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Цена, руб. | Количество, ед. | Стоимость, руб.  (с учетом НДС = 20%) |
| за Тис |
| Электроэнергия | 4,48 | 400 КВт | 1792 |
| Бумага | 216,00 | 2 упаковки | 432 |
| Всего |  |  | 2224 |

Амортизация, входящая в формулу стоимости ИС – это амортизации оборудования, используемого при разработке системы.

А = А1, (5.8)

где А – общая амортизация, руб.;

А1 – амортизация оборудования руб.

В таблице 4 приведены расчеты норм амортизации оборудования, а в таблицу 5 сведены затраты на амортизацию оборудования и нематериальных активов, используемых в процессе разработки системы.

Таблица 4 – Расчеты норм амортизации

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование | Стоимость, руб. | Срок эксплуатации, лет | Норма амортизации, руб/мес. | Норма амортизации, руб/день |
| Компьютер | 80 000,00 | 10 | 80 000/  10/12 = 666,66 | 666,66/21 = 31,74 |
| Всего |  |  | 666,66 | 31,74 |

Таблица 5 – Расчет амортизации оборудования и нематериальных активов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Амортизационные активы | Кол-во, шт. | Норма амортизации, руб/день | Длительность, дней | Стоимость, руб. |
| Компьютер | 2 | 19,84 | 55 | 2182,4 |
| Всего (***А***) |  |  |  | 2182,4 |

Исходя из полученных расчетных данных стоимость разработки системы автоматизации:

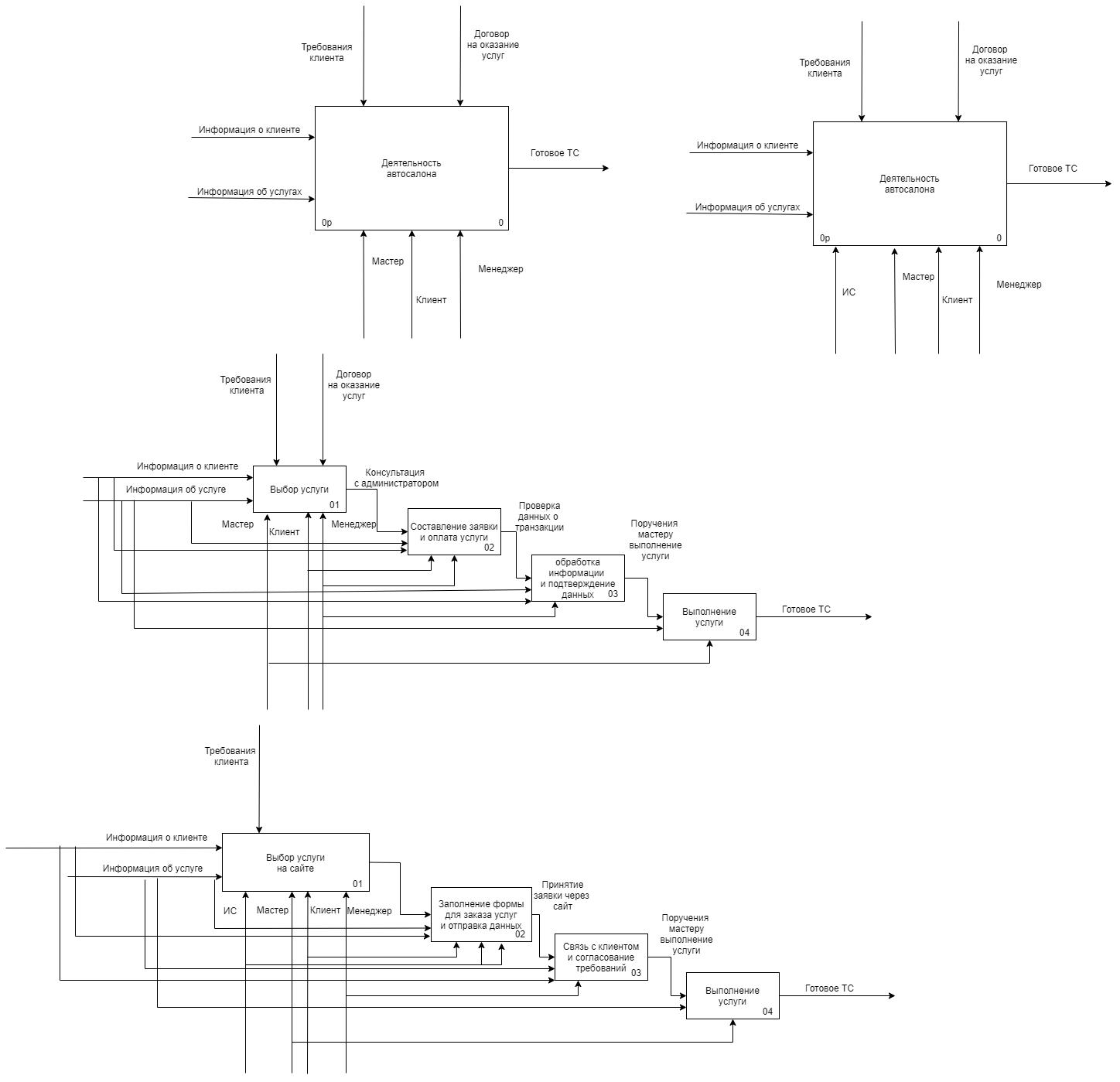
С = 196237,9 + 2224 + 2182,4 = 200 644,3 (руб.)

5.2 Расчет стоимости выполнения процесса до автоматизации

Применение метода ФСА для оценки экономической эффективности начинается с построения функциональной модели процесса, для которого будут проводиться расчеты.

Рассмотрим процесс деятельности.

На рисунке 7 изображена последовательность операций рассматриваемого процесса.

Рисунок 7 – Декомпозиция деятельности до автоматизации

До внедрения автоматизированной подсистемы процесс деятельности отдела услуг состоит из следующих операций:

1. выбор услуг;
2. составление заявки и оплата услуги;
3. обработка информации и подтверждение данных;
4. выполнение услуги

В данном случае выбор услуг и согласование с компанией у клиента осуществляется двумя сотрудниками отдела (менеджер и администратор); принятые заказы обрабатывают также два сотрудника (менеджер и администратор; подтверждением оплаты заказа занимается менеджер по продажам.

Система автоматизации будет затрагивать менеджера по продажам и бухгалтера в нашем отделе, так что внимательно проанализируем их рабочий процесс.

Предположим, что на проведение бухгалтерского учета за один расчетный период (1 месяц) наш единственный бухгалтер тратит по 8 часов в день, 5 дней в неделю.

Затраты на выполнение процесса до автоматизации рассчитываются по следующей формуле:

, (5.9)

где n – количество операций в процессе;

ЗOi – заработная плата сотрудника при выполнении i-ой операции, руб.;

МOi– затраты на расходные материалы, необходимые при выполнении i-ой операции, руб.;

АOi – амортизация оборудования и нематериальных активов для i-ой операции, руб.

Расчет затрат на заработную плату сотрудников представлен в таблице 6.

Таблица 6 – Расчет затрат на заработную плату сотрудникам

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Наименование сотрудников | Заработная плата | Зарплата, руб./час | Примерное время работы за один расчетный период, час | Количество сотрудников, чел. | Затраты за один расчетный период, руб.  (с учетом СВ (30%)) |
| Бухгалтер | 20 000,00 | 20 000/21/8 = 119 | 168 | 1 | 19992 + 5997,6 =  25989,6 |
| Менеджер | 35 000,00 | 35 000/21/8 = 208 | 168 | 1 | 34944 + 10 483,2 = 45427,2 |
| Всего |  |  |  |  | 68 416,8 |

При расчетах затрат на расходные материалы (таблица 7) необходимо учесть следующие данные:

1. стоимость пачки бумаги объемом 500 листов составляет 216 руб.;
2. потребляемая мощность компьютера Р = 0,2 кВт/ч; на протяжении всех операций будет задействовано 2 компьютера.

Таблица 7 – Расчет затрат на расходные материалы до автоматизации

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование операции | Затрачиваемые ресурсы | Количество | Цена, руб. | Затраты, руб. |
| О1 – Выбор услуг | Бумага | 25 листов | 2,5 | 25 \* 2,5 + 9,6 \* 4,48 = 105,508 |
| Электроэнергия  48 часов | 9,6 кВт | 4,48 |
| Наименование операции | Затрачиваемые ресурсы | Количество | Цена, руб. | Затраты, руб. |
| О2 – Составление заявки и оплата услуги; | Бумага | 30 листов | 2,5 | 30\*2,5 + 14 \* 4,48= 137,72 |
| Электроэнергия  70 часов | 14 кВт | 4,48 |
| О3 – Обработка информации и подтверждение данных | Бумага | 30 листов | 2,5 | 30 \* 2,5 +  10 \* 4,48 =  119,8 |
| Электроэнергия  50 часов | 10 кВт | 4,48 |
| О4 – Выполнение услуги | Бумага  Электроэнергия  50 часов | 30 листов | 2,5 | 30 \* 2,5 +  10 \* 4,48 =  119,8 |
| Всего |  |  |  | 482 |
| НДС(20%) |  |  |  | 96,4 |
| ИТОГО |  |  |  | 578,7 |

В таблицах 8, 9 приведены расчеты норм амортизации и затрат на амортизацию оборудования и нематериальных активов.

Таблица 8 *–* Оборудование и программное обеспечение

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование | Стоимость  руб. | Срок эксплуатации, лет | Норма амортизации, руб./мес. | Норма амортизации, руб./час |
| Компьютер | 50 000,00 | 10 | 50 000/  10/12 = 416,66 | 416,66/21 = 19,84 |
| Всего |  |  |  | 19,84 |

Таблица 9 – Расчет амортизации оборудования и нематериальных активов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Амортизационные активы | Кол-во, шт. | Норма амортизации, руб./день | Длительность, дней | Стоимость  руб. |
| Компьютер | 2 | 19,84 | 21 | 833,28 |
| Всего (***А***) |  |  |  | 833,28 |

Исходя из данных, приведенных в таблицах 12, 13, стоимость одной итерации процесса деятельности отдела клининговых услуг при реализации товара, который длиться 1 месяц:

*Cдо =* 71 416,8*+*435,628*+*833,28=72 685,708 (руб.).

5.2 Расчет стоимости выполнения системы после автоматизации

Теперь рассмотрим тот же процесс после внедрения системы автоматизации. На рисунке 8 изображена последовательность действий процесса обслуживания клиентов деятельности после автоматизации.

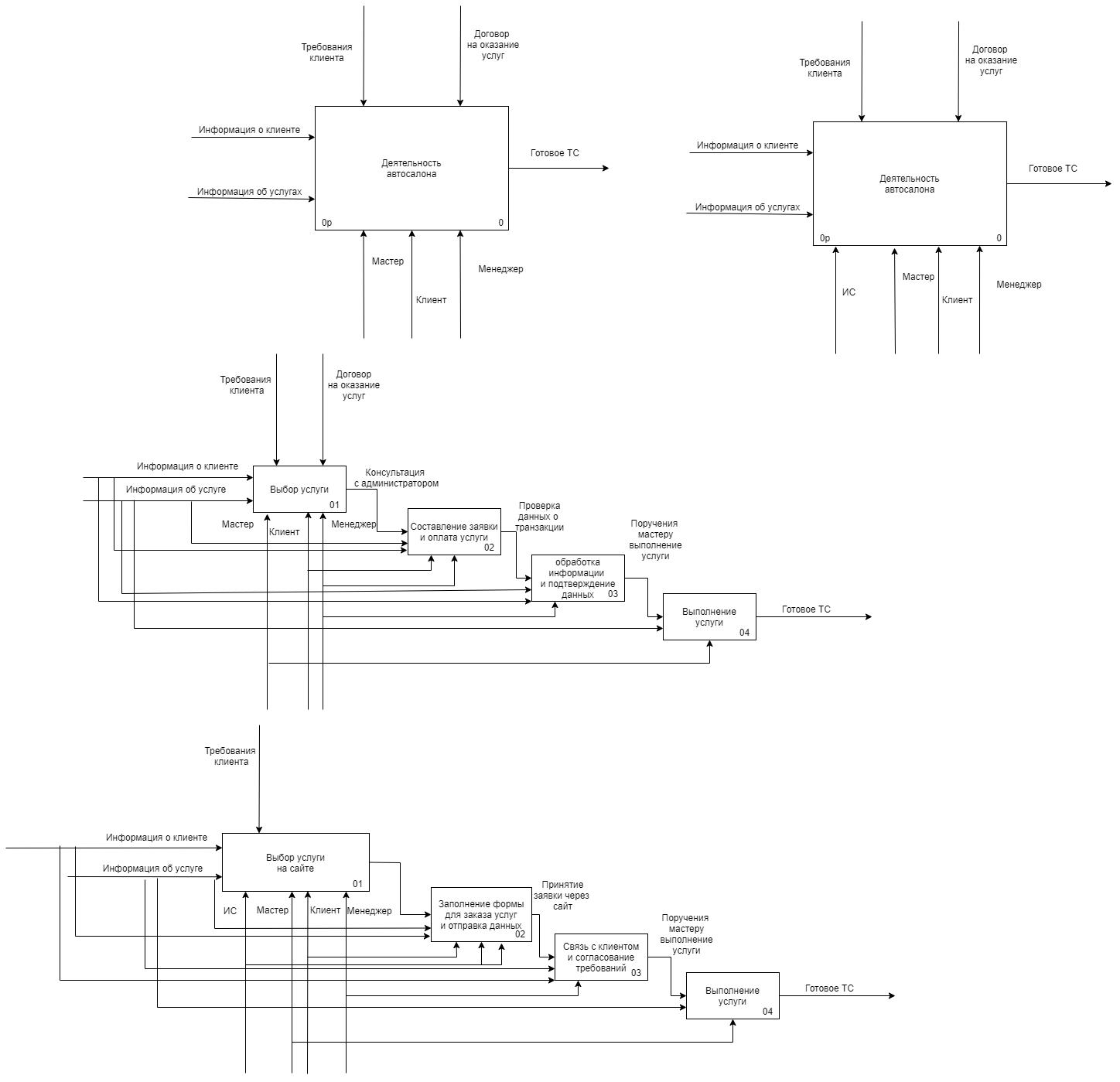


Рисунок 8 – Декомпозиция процесса после внедрения ИС

После внедрения системы автоматизации некоторые процессы сократится по времени, также произойдет уменьшения расхода подручных средств, однако основные операции останутся те же.

Затраты на выполнение процесса после автоматизации рассчитываются по следующей формуле:

, (5.10)

n – количество операций в процессе;

ЗOi – заработная плата сотрудников при выполнении i-ой операции, руб.;

МOi– затраты на расходные материалы, необходимые при выполнении i-ой операции, руб.;

АOi – амортизация оборудования и нематериальных активов для i-ой операции, руб.

Для обеспечения нормального функционирования проектируемой системы необходимо ПК стандартной офисной комплектации для работы бухгалтера и менеджера.

Для указанного оборудования необходимо рассчитать нормы амортизации. Расчет норм амортизации производится по следующим формулам:

А = А1, (5.11)

где *А* – общая амортизация, руб.;

*А1* – амортизация оборудования, руб.

Дополнительных программных продуктов для обеспечения работы ИС не требуется.

В таблице 10 - 13 приведены расчеты норм амортизации оборудования и программного обеспечения.

Таблица 10 – Расчеты норм амортизации

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование | Стоимость, руб. | Срок эксплуатации, лет | Норма амортизации, руб./мес. | Норма амортизации, руб./час |
| Компьютер | 50 000,00 | 10 | 50 000/  10/12 = 416,66 | 416,66/21 = 19,84 |
| Всего |  |  |  | 19,84 |

Таблица 11 – Расчет амортизации оборудования и нематериальных активов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Амортизационные активы | Кол-во, шт. | Норма амортизации, руб./день | Длительность, дней | Стоимость, руб. |
| Компьютер | 2 | 19,84 | 5 | 198,4 |
| Всего (А) |  |  |  | 198,4 |

При расчете затрат на заработную плату необходимо учесть, что после автоматизации добавляется такой специалист как бухгалтер сможет тратить вчетверо меньше своего рабочего времени на бухгалтерский учет, и менеджер по продажам при принятия заказа у клиента (таблица 12).

Таблица 12 – Расчет расходов заработной платы на выполнение процесса

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Наименование сотрудников | Заработная плата | Зарплата, руб./час | Примерное время работы за один расчетный период, час | Количество сотрудников, чел. | Затраты за один расчетный период, руб.  (с учётом СВ (30%)) |
| Бухгалтер | 25 000,00 | 25 000  /21/8 = 148,8 | 120 | 1 | 17856 + 5356,8 =  23212,8 |
| Менеджер | 37 000,00 | 37 000/21/8 = 220,23 | 120 | 1 | 26427,6 + 7928,28 = 34355,88 |
| Всего |  |  |  |  | 57568,68 |

В процессе автоматизации конфигурация рабочего оборудования не изменилась, однако количество расходных материалов значительно сократилось (таблица 13).

Таблица 13 – Расчет затрат на расходные материалы после автоматизации

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование операции | Затрачиваемые ресурсы | Количество | Цена, руб. | Затраты, руб. |
| О1 – Выбор услуги на сайте | Электроэнергия  20 часов | 4 кВт | 4,48 | 4 \* 4,48 = 17,92 |
| О2 – Заполнение формы для заказа услуги и отправка данных | Электроэнергия  25 часов | 5 кВт | 4,48 | 5 \* 4,48 = 22,4 |

*Продолжение таблицы …*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование операции | Затрачиваемые ресурсы | Количество | Цена, руб. | Затраты, руб. |
| О3 – Связь с клиентом и согласование требований | Электроэнергия  25 часов | 4 кВт | 4,48 | 4 \* 4,48 =  17,92 |
| О4 – Выполнение услуги | Электроэнергия  28 часов | 3 кВт | 4,48 | 3 \* 4,48 = 13,44 |
| Всего |  |  |  | 71,68 |
| НДС (20%) |  |  |  | 14, 336 |
| ИТОГО |  |  |  | 86,016 |

Таким образом, согласно данным таблиц 10 – 13 стоимость процесса после автоматизации составит:

Cпосле = 198,4 + 57568,68 + 86,016= 57 853,096 (руб.).

5.3 Расчет показателей эффективности системы

На основании вышеизложенных расчетов, определим экономический эффект от автоматизации основных процессов.

Экономический эффект для предприятия, рассчитывается по формуле:

Э = (Cдо – Cпосле) \* Ч, (5.12)

где Ч – итерация (расчётный период 1 месяц), руб.

В течении года происходит 12 итерации данного процесса

Э = (68 685,708 – 57 853,096) \* 12 = 129 991,344 (руб.)

В результате получаем, что экономический эффект от автоматизации процесса составляет 129 991,344 рублей за год.

Исходя из полученных результатов расчета годового экономического эффекта, можно рассчитать коэффициент экономической эффективности:

Е = Э/Сис, (5.13)

где Э – годовой экономический эффект, руб.;

Сис – стоимость разработки системы, руб.

Е = 129 991,344 / 200 644,3 ≈ 0,64

Срок окупаемости данного проекта можно рассчитать по формуле:

Т=1/Е, (5.14)

Таким образом, срок окупаемости данного проекта составляет

Т = 1 / 0,64 = 1,6 ≈ 548 дней.

Рассчитанные показатели свидетельствуют об умеренной экономической эффективности проектируемой системы.

Заключение

Постоянно увеличивается число пользователей Интернета, и лиц, осуществляющих покупки посредством него. Данный проект является удобной WEB - системой, которая дает возможность пользователям заказать продукцию, не выходя из дома или офиса. Интернет-магазин удобен для людей, которым не хватает времени для походов по магазинам.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. Браун Итан Веб-разработка с применением Node и Express. Полноценное использование стека JavaScript. – СПб.: Питер, 2017.
  2. Использование localStorage с React Hooks [Электронный ресурс] – Режим доступа: <https://blog.logrocket.com/using-localstorage-react-hooks/>
  3. Используем localStorage в React [Электронный ресурс] – Режим доступа: <https://bookflow.ru/ispolzuem-localstorage-v-react/>
  4. Стоян Стефанов React.js. Быстрый старт. Питер, 2017.
  5. React и Redux: функциональная веб-разработка/Алекс Бэнкс, Ева Порселло. Питер, 2018.
  6. Node.js в действии / Кантелон М., Хартер М., Головайчук T., Райлих Н., - 2-е изд. - СПб.: Питер, 2018

ПРИЛОЖЕНИЕ А

Пользовательский интерфейс

1. Главная страница

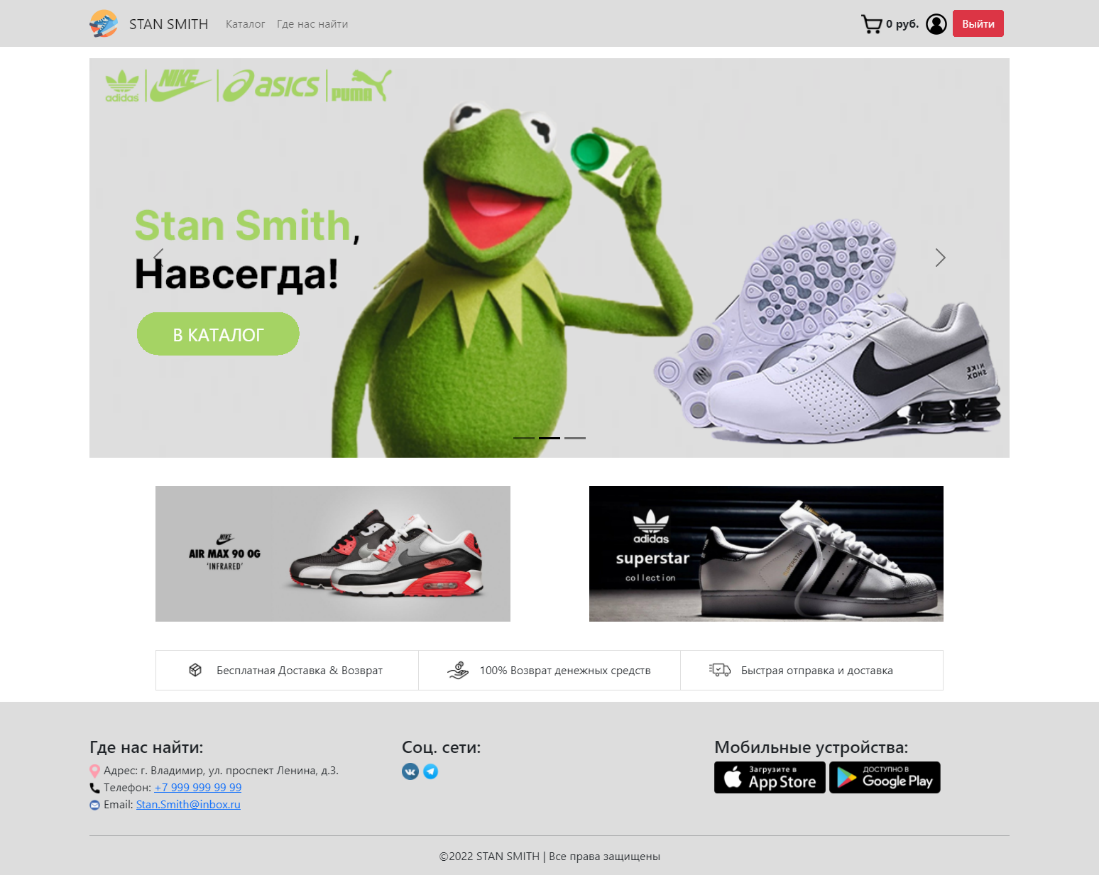


Рисунок А.1 – Главная страница

1. Каталог

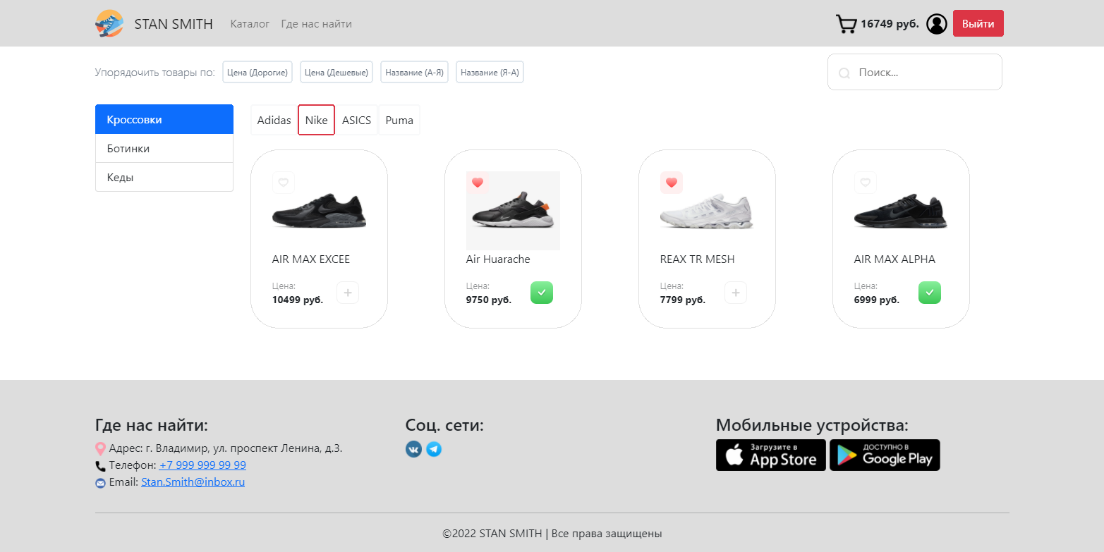


Рисунок А.2 – Каталог

1. Страница «Где нас найти»

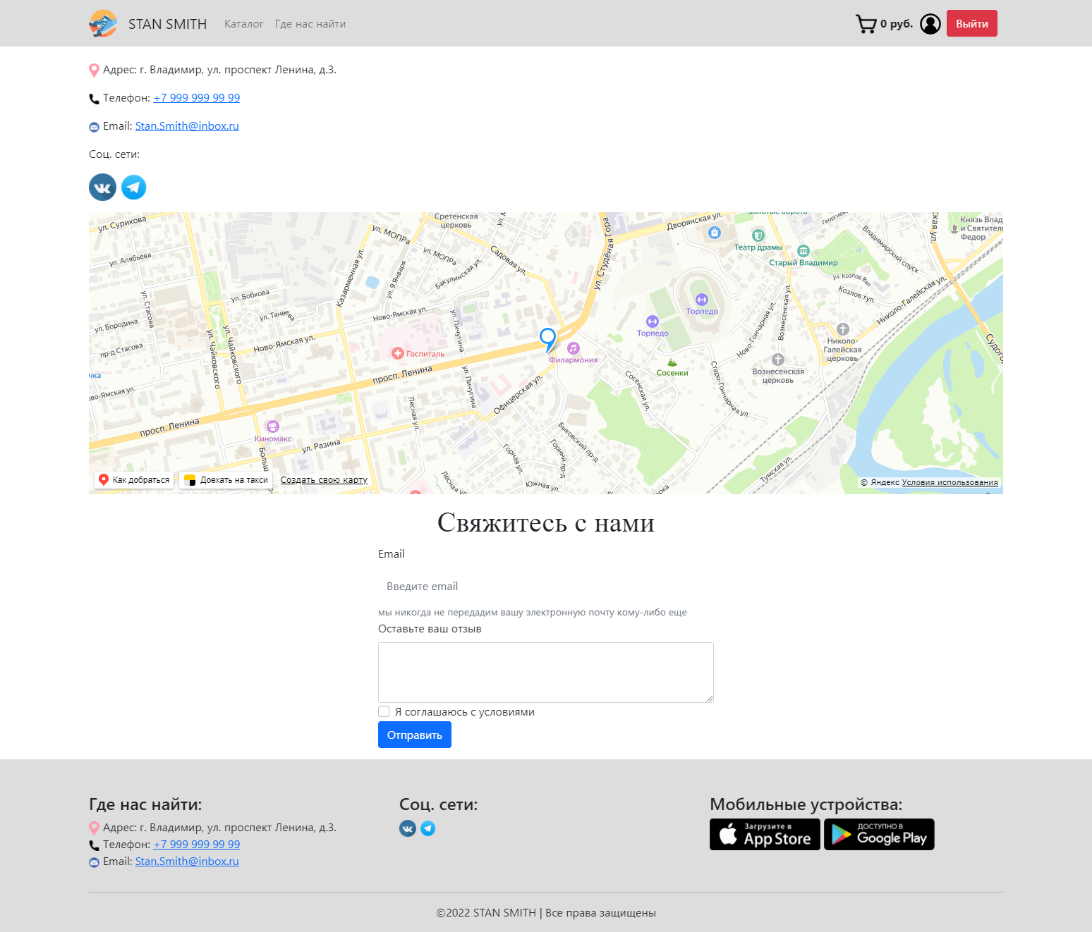


Рисунок А.3 – Страница «Где нас найти»

1. Корзина

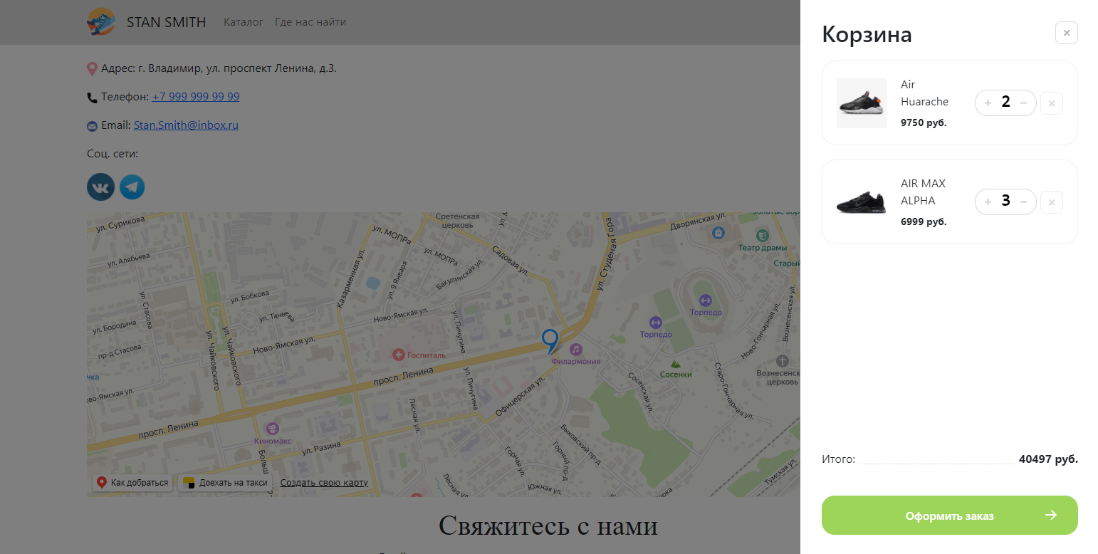


Рисунок А.4 – Корзина

1. Личный кабинет

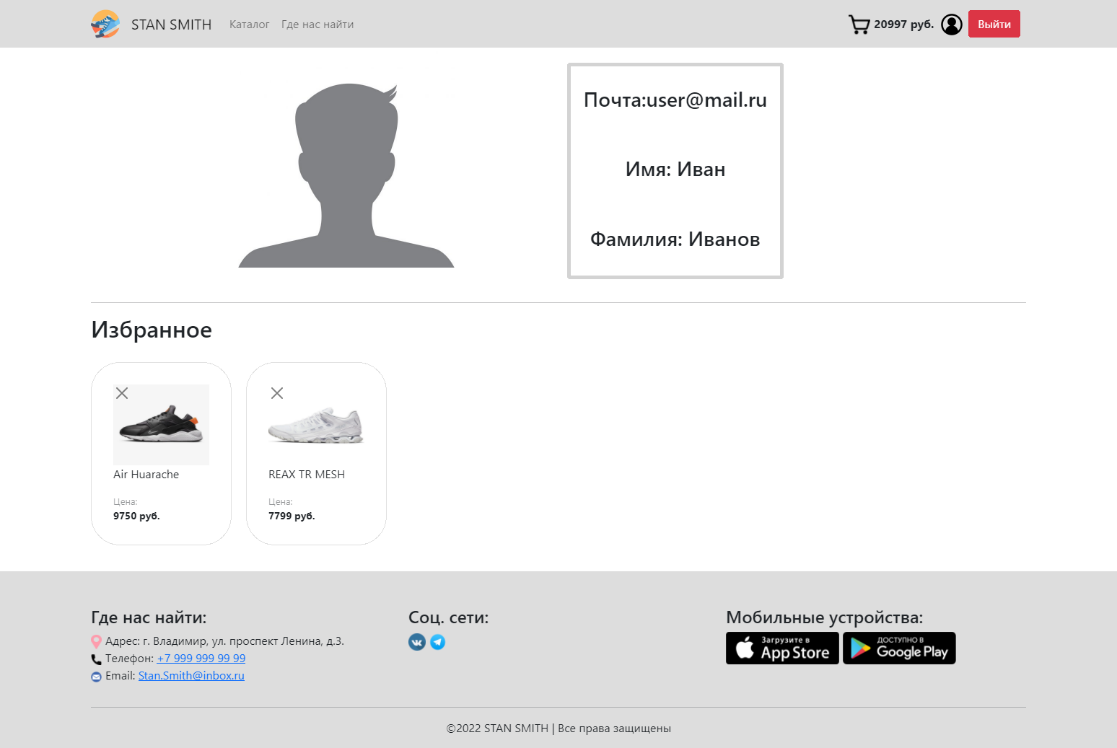


Рисунок А.5 – Личный кабинет

1. Страница «О товаре»

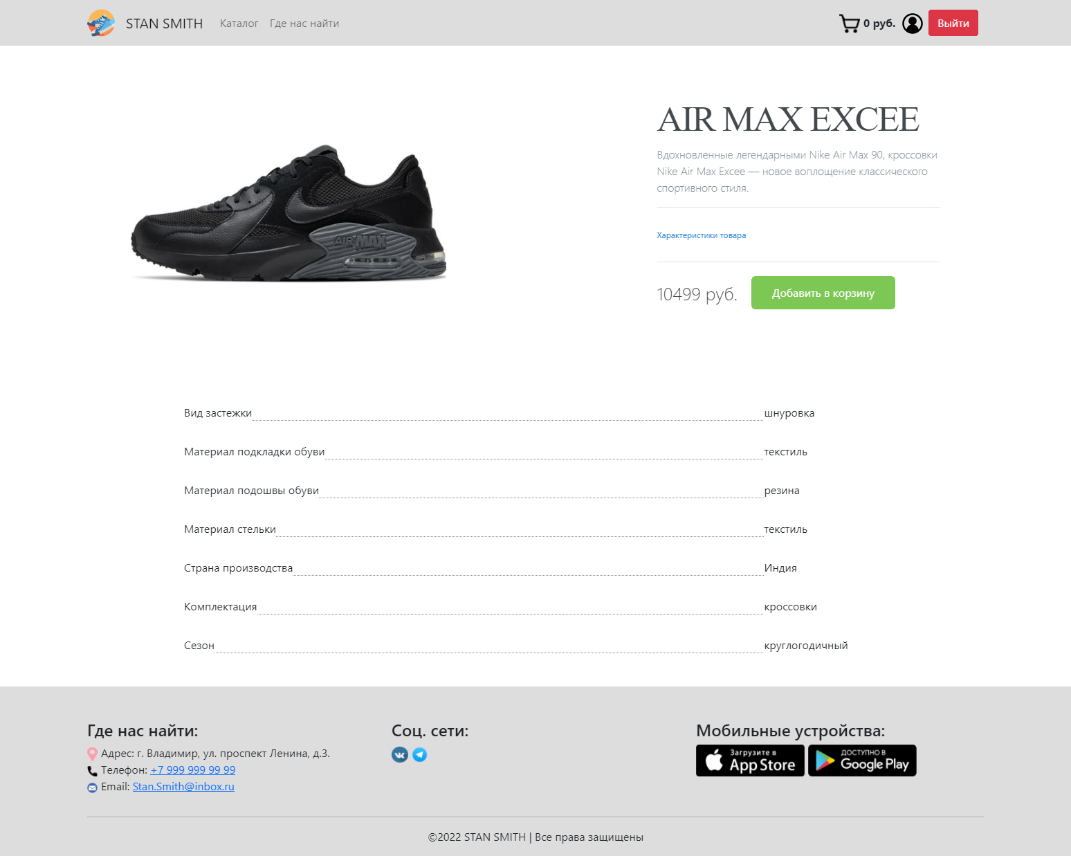


Рисунок А.6 – Страница «О товаре»

1. Страница «Оплата заказа»

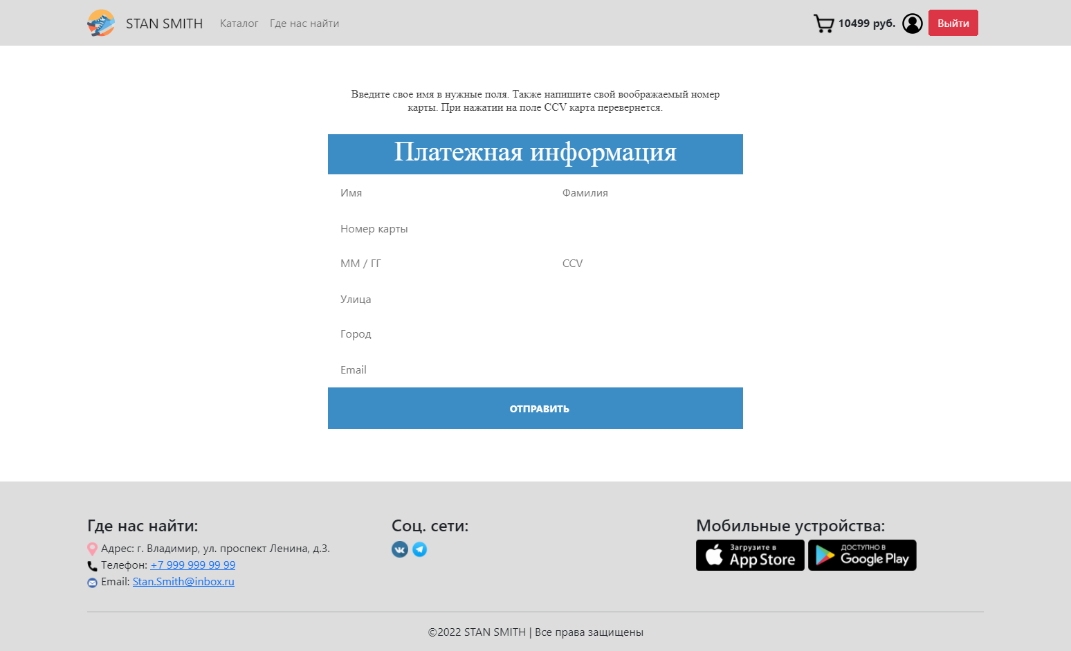


Рисунок А.7 – Страница «Оплата заказа»

1. Страница 404

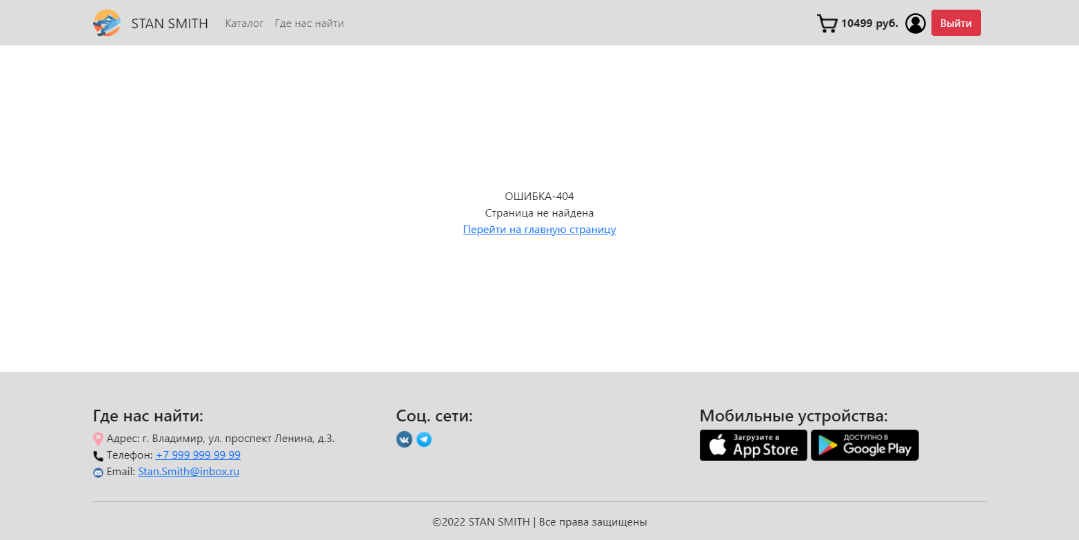


Рисунок А.8 – Страница 404

1. Страница Авторизации

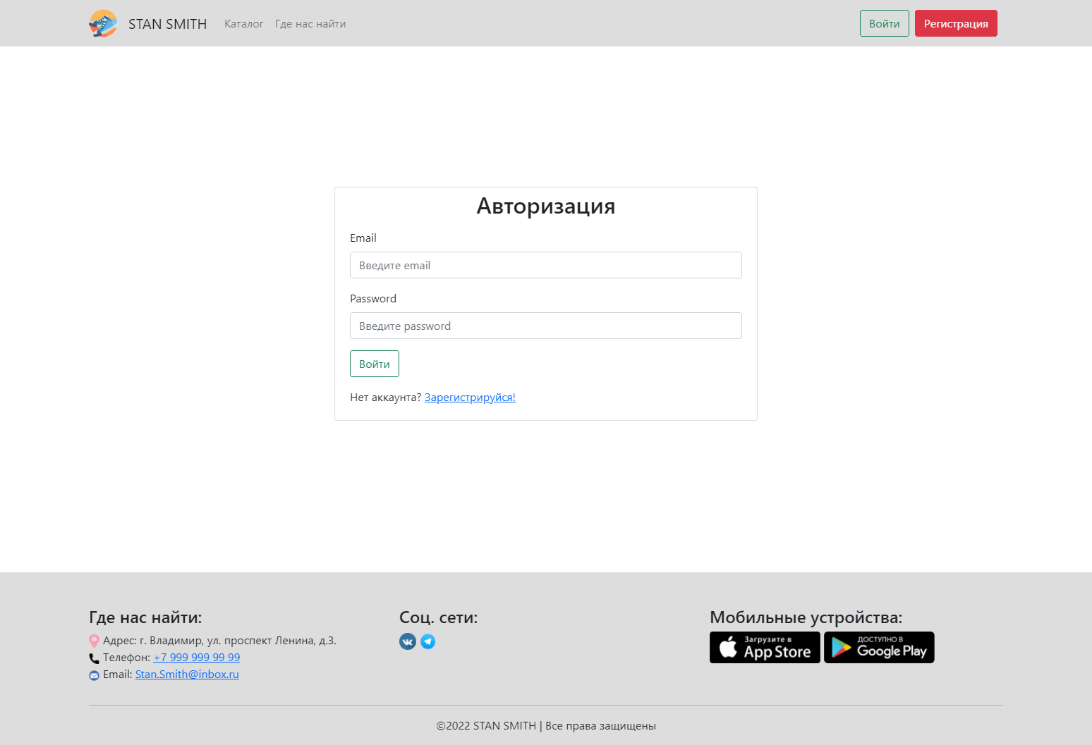


Рисунок А.9 – Страница Авторизации

1. Страница Регистрации

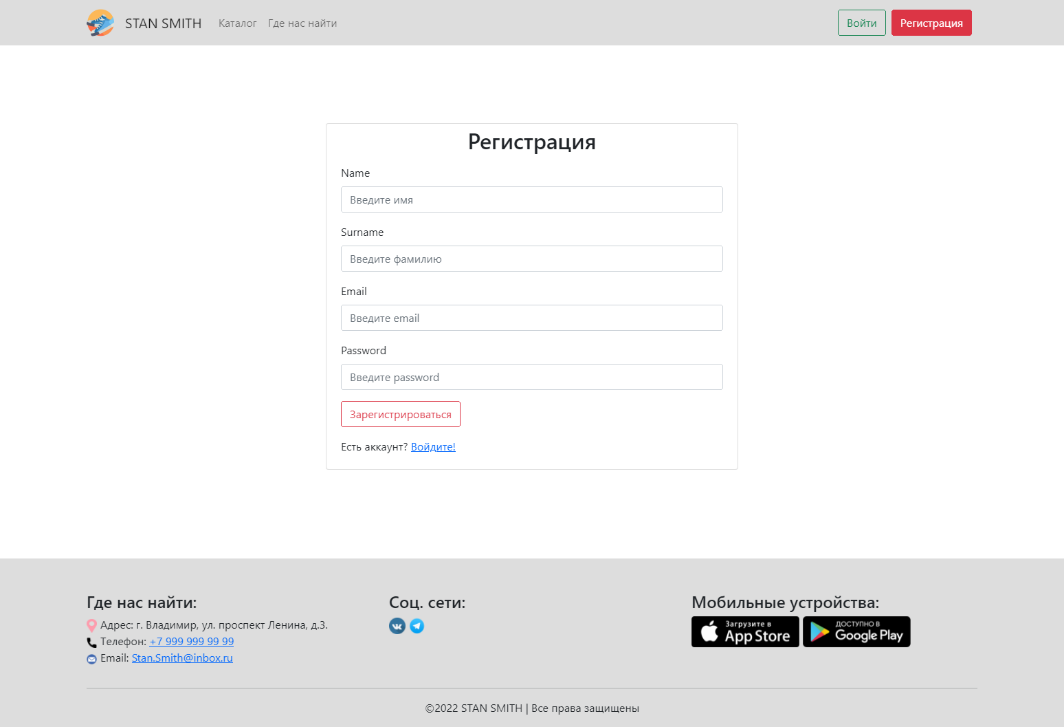


Рисунок А.10 – Страница Регистрации

1. Модальное окно «Добавить тип»

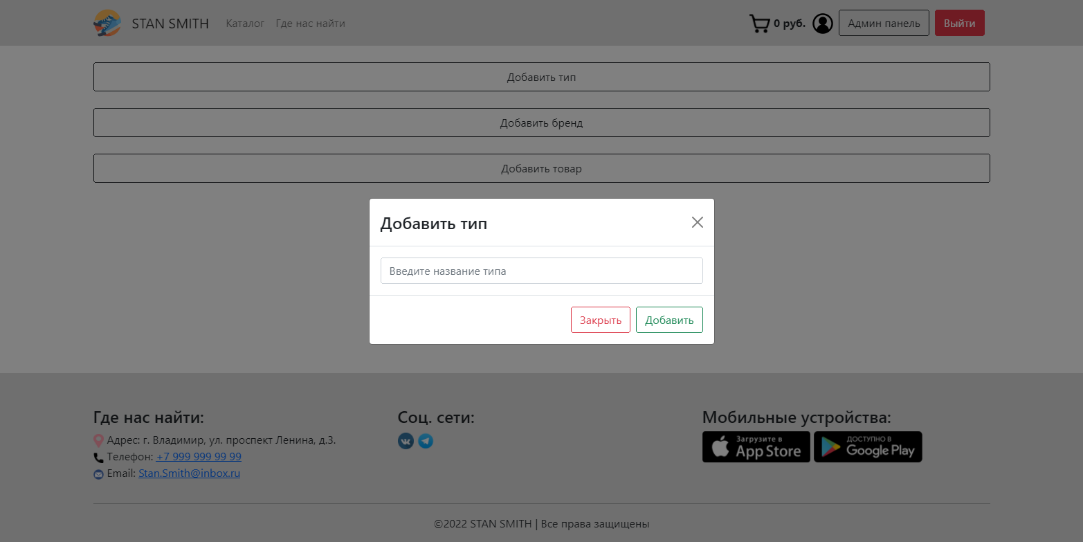


Рисунок А.11 – Модальное окно «Добавить тип»

1. Модальное окно «Добавить бренд»

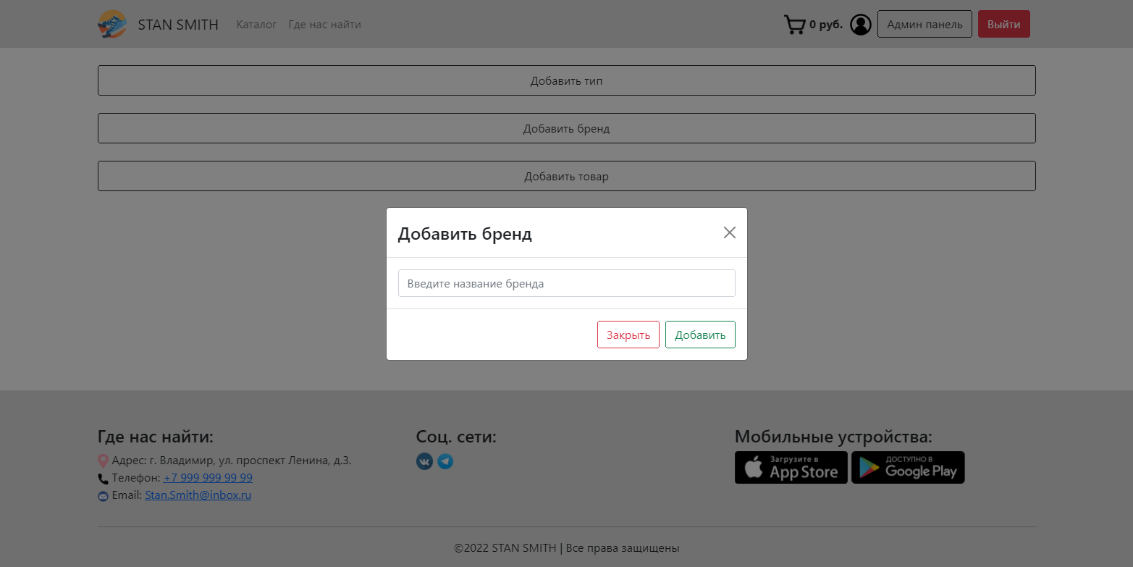


Рисунок А.12 – Модальное окно «Добавить бренд»

1. Модальное окно «Добавить товар»

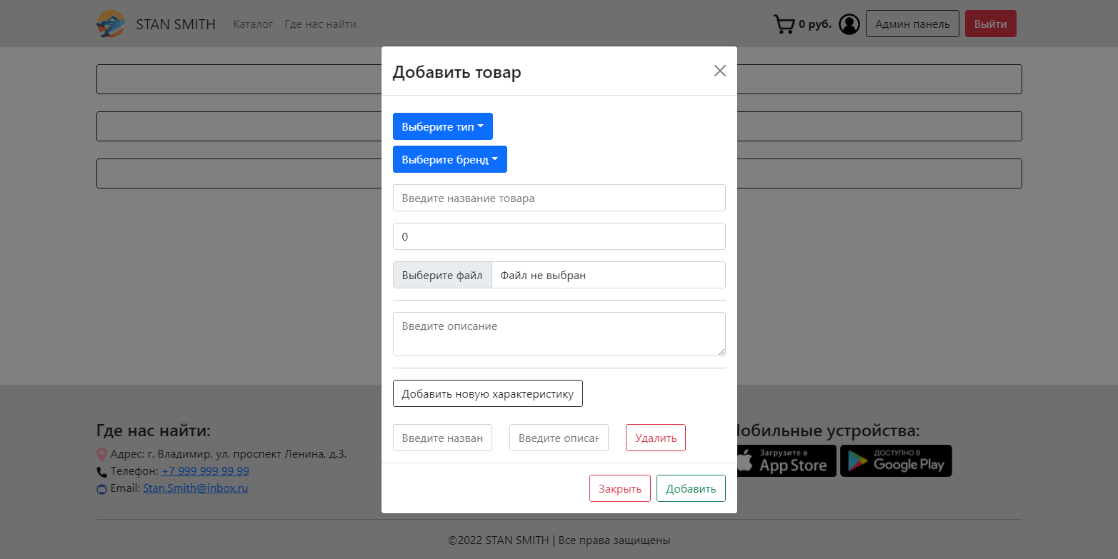


Рисунок А.13 – Модальное окно «Добавить товар»

ПРИЛОЖЕНИЕ Б

Листинг кода реализации серверной части

1. Файл index.js

require('dotenv').config()

const express = require('express')

const sequelize = require('./db')

const models = require('./models/models')

const cors = require('cors')

const fileUpload = require('express-fileupload')

const router = require('./routes/index')

const errorHandler = require('./middleware/ErrorHandlingMiddleware')

const path = require('path')

const PORT = process.env.PORT || 5000

const app = express()

app.use(cors())

app.use(express.json())

app.use(express.static(path.resolve(\_\_dirname, 'static')))

app.use(fileUpload({}))

app.use('/api', router)

app.use(errorHandler)

const start = async () => {

try {

await sequelize.authenticate()

await sequelize.sync()

app.listen(PORT, () => console.log(`Server started on port ${PORT}`))

} catch (e) {

console.log(e)

}

}

start()

1. Файл models.js

const sequelize = require('../db')

const { DataTypes } = require('sequelize')

const User = sequelize.define('user', {

id: { type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true },

name: { type: DataTypes.STRING },

surname: { type: DataTypes.STRING },

email: { type: DataTypes.STRING, unique: true },

password: { type: DataTypes.STRING },

role: { type: DataTypes.STRING, defaultValue: 'USER' },

})

const Product = sequelize.define('product', {

id: { type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true },

name: { type: DataTypes.STRING, allowNull: false },

price: { type: DataTypes.INTEGER, allowNull: false },

specification: { type: DataTypes.STRING(1000), allowNull: true },

img: { type: DataTypes.STRING, allowNull: false },

})

const Type = sequelize.define('type', {

id: { type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true },

name: { type: DataTypes.STRING, unique: true, allowNull: false },

})

const Brand = sequelize.define('brand', {

id: { type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true },

name: { type: DataTypes.STRING, unique: true, allowNull: false },

})

const ProductInfo = sequelize.define('product\_info', {

id: { type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true },

title: { type: DataTypes.STRING, allowNull: false },

description: { type: DataTypes.STRING, allowNull: false },

})

const TypeBrand = sequelize.define('type\_brand', {

id: { type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true },

})

Type.hasMany(Product)

Product.belongsTo(Type)

Brand.hasMany(Product)

Product.belongsTo(Brand)

Product.hasMany(ProductInfo, { as: 'info' })

ProductInfo.belongsTo(Product)

Type.belongsToMany(Brand, { through: TypeBrand })

Brand.belongsToMany(Type, { through: TypeBrand })

module.exports = {

User,

Product,

Type,

Brand,

TypeBrand,

ProductInfo,

}

1. Файл userController.js

const ApiError = require('../error/ApiError')

const bcrypt = require('bcrypt')

const jwt = require('jsonwebtoken')

const { User } = require('../models/models')

const { default: jwtDecode } = require('jwt-decode')

const generateJwt = (id, email, role) => {

return jwt.sign({ id, email, role }, process.env.SECRET\_KEY, {

expiresIn: '24h',

})

}

class UserController {

async registration(req, res, next) {

const { name, surname, email, password, role } = req.body

if (!email || !password || !name || !surname) {

return next(

ApiError.badRequest('Некорректный Имя, Фамилия, email или password')

)

}

const candidate = await User.findOne({ where: { email } })

if (candidate) {

return next(

ApiError.badRequest('Пользователь с таким email уже существует')

)

}

const hashPassword = await bcrypt.hash(password, 5)

const user = await User.create({

name,

surname,

email,

role,

password: hashPassword,

})

const token = generateJwt(user.id, user.email, user.role)

return res.json({ token })

}

async login(req, res, next) {

const { email, password } = req.body

const user = await User.findOne({ where: { email } })

if (!user) {

return next(ApiError.internal('Пользователь не найден'))

}

let comparePassword = bcrypt.compareSync(password, user.password)

if (!comparePassword) {

return next(ApiError.internal('Указан неверный пароль'))

}

const token = generateJwt(user.id, user.email, user.role)

return res.json({ token })

}

async check(req, res, next) {

const token = generateJwt(req.user.id, req.user.email, req.user.role)

return res.json({ token })

}

async getAll(req, res) {

const users = await User.findAll()

return res.json(users)

}

async getOne(req, res) {

const { id } = req.params

const user = await User.findOne({

where: { id },

})

return res.json(user)

}

}

module.exports = new UserController()

1. Файл productController.js

const uuid = require('uuid')

const path = require('path')

const { Product, ProductInfo } = require('../models/models')

const ApiError = require('../error/ApiError')

class ProductController {

async create(req, res, next) {

try {

let { name, price, specification, brandId, typeId, info } = req.body

const { img } = req.files

let fileName = uuid.v4() + '.jpg'

img.mv(path.resolve(\_\_dirname, '..', 'static', fileName))

const product = await Product.create({

name,

price,

specification,

brandId,

typeId,

img: fileName,

})

if (info) {

info = JSON.parse(info)

info.forEach(i =>

ProductInfo.create({

title: i.title,

description: i.description,

productId: product.id,

})

)

}

return res.json(product)

} catch (e) {

next(ApiError.badRequest(e.message))

}

}

async delete(req, res, next) {

try {

let delprod = await Product.destroy({

where: {

id: req.body.id,

}

});

return res.json(delprod)

} catch (e) {

next(ApiError.badRequest(e.message))

}

}

async getAll(req, res) {

let { brandId, typeId } = req.query

let products

if (!brandId && !typeId) {

products = await Product.findAndCountAll()

}

if (brandId && !typeId) {

products = await Product.findAndCountAll({

where: { brandId },

})

}

if (!brandId && typeId) {

products = await Product.findAndCountAll({

where: { typeId },

})

}

if (brandId && typeId) {

products = await Product.findAndCountAll({

where: { typeId, brandId },

})

}

return res.json(products)

}

async getOne(req, res) {

const { id } = req.params

const product = await Product.findOne({

where: { id },

include: [{ model: ProductInfo, as: 'info' }],

})

return res.json(product)

}

}

module.exports = new ProductController()

1. Файл authMiddleware.js

const jwt = require('jsonwebtoken')

module.exports = function (req, res, next) {

if (req.method === "OPTIONS") {

next()

}

try {

const token = req.headers.authorization.split(' ')[1]

if (!token) {

return res.status(401).json({message: "Не авторизован"})

}

const decoded = jwt.verify(token, process.env.SECRET\_KEY)

req.user = decoded

next()

} catch (e) {

res.status(401).json({message: "Не авторизован"})

}

};

1. Файл checkRoleMiddleware.js

const jwt = require('jsonwebtoken')

module.exports = function(role) {

return function (req, res, next) {

if (req.method === "OPTIONS") {

next()

}

try {

const token = req.headers.authorization.split(' ')[1]

if (!token) {

return res.status(401).json({message: "Не авторизован"})

}

const decoded = jwt.verify(token, process.env.SECRET\_KEY)

if (decoded.role !== role) {

return res.status(403).json({message: "Нет доступа"})

}

req.user = decoded;

next()

} catch (e) {

res.status(401).json({message: "Не авторизован"})

}

};

}

ПРИЛОЖЕНИЕ В

Листинг кода реализации клиентской части

1. Файл index.js

import React, { createContext } from 'react'

import ReactDOM from 'react-dom'

import './index.scss'

import App from './App'

import 'bootstrap/dist/css/bootstrap.min.css'

import 'macro-css'

import UserStore from './store/UserStore'

import ProductStore from './store/ProductStore'

export const Context = createContext(null)

ReactDOM.render(

<Context.Provider

value={{

user: new UserStore(),

product: new ProductStore(),

}}

>

<App />

</Context.Provider>,

document.getElementById('root')

)

1. Файл App.js

import React, { useContext, useEffect, useState } from 'react'

import { BrowserRouter } from 'react-router-dom'

import Footer from './components/Footer'

import Header from './components/Header'

import Cart from './components/Cart'

import AppRouter from './components/AppRouter'

import { Context } from './index'

import { observer } from 'mobx-react-lite'

import { check } from './http/userAPI'

const App = observer(() => {

const { user } = useContext(Context)

const [loading, setLoading] = useState(true)

const [cartOpened, setCartOpened] = React.useState(false)

const [cartItems, setCartItems] = React.useState([])

useEffect(() => {

check()

.then(data => {

user.setUser(true)

user.setIsAuth(true)

})

.finally(() => setLoading(false))

}, [])

return (

<div className='page-container'>

<div className='content-wrap'>

{cartOpened && (

<Cart items={cartItems} onClose={() => setCartOpened(false)} />

)}

<BrowserRouter>

<Header onClickCart={() => setCartOpened(true)} />

<AppRouter />

</BrowserRouter>

</div>

<Footer />

</div>

)

})

export default App

1. Файл AppRouter.js

import React, { useContext } from 'react'

import { Routes, Route } from 'react-router-dom'

import { authRoutes, adminRoutes, publicRoutes } from '../routes'

import { Context } from '../index'

function AppRouter() {

const { user } = useContext(Context)

return (

<Routes>

{user.isAuth &&

authRoutes.map(({ path, Component }) => (

<Route key={path} path={path} element={<Component />} exact />

))}

{user.isAdmin &&

adminRoutes.map(({ path, Component }) => (

<Route key={path} path={path} element={<Component />} exact />

))}

{publicRoutes.map(({ path, Component }) => (

<Route key={path} path={path} element={<Component />} exact />

))}

</Routes>

)

}

export default AppRouter

1. Файл Main.js

import { Link } from 'react-router-dom'

import { Container, Col, Row } from 'react-bootstrap'

import CarouselBox from '../components/CarouselBox'

import { SHOP\_ROUTE } from '../utils/const'

function Main() {

return (

<div>

<CarouselBox />

<Container>

<div className='main-container'>

<div className='banners'>

<a href={SHOP\_ROUTE} className='banners\_\_link'>

<img

src='/img/nike.jpg'

alt='Banner 1'

className='banners\_\_img'

/>

</a>

<a href={SHOP\_ROUTE} className='banners\_\_link'>

<img

src='/img/adidas.png'

alt='Banner 2'

className='banners\_\_img'

/>

</a>

</div>

</div>

<div className='main-container main-container-advantages'>

<div className='advantages'>

<div className='advantages\_\_item'>

<div className='advantages\_\_icon advantages\_\_icon\_fast'>

<img src='/img/dostavka.png' alt='dostavka' />

</div>

<span className='advantages\_\_item\_text'>

Бесплатная Доставка &amp; Возврат

</span>

</div>

<div className='advantages\_\_item'>

<div className='advantages\_\_icon advantages\_\_icon\_money'>

<img src='/img/vosvrat.png' alt='vozvrat' />

</div>

<span className='advantages\_\_item\_text'>

100% Возврат денежных средств

</span>

</div>

<div className='advantages\_\_item'>

<div className='advantages\_\_icon advantages\_\_icon\_car'>

<img src='/img/otpravka.png' alt='otpravka' />

</div>

<span className='advantages\_\_item\_text'>

Быстрая отправка и доставка

</span>

</div>

</div>

</div>

</Container>

</div>

)

}

export default Main

1. Файл ProductItem.js

import React, { useContext } from 'react'

import { Col, CloseButton } from 'react-bootstrap'

import { useNavigate } from 'react-router-dom'

import { PRODUCT\_ROUTE } from '../utils/const'

import CartStore from '../store/CartStore'

import FavoriteStore from '../store/FavoriteStore'

import { delProduct } from '../http/productAPI'

import { Context } from '../index'

const ProductItem = ({ product }) => {

const [isAdded, setIsAdded] = React.useState(false)

const [isFavorite, setIsFavorite] = React.useState(false)

const [value, setValue] = React.useState('')

const { user } = useContext(Context)

const onClickFavorite = () => {

setIsFavorite(!isFavorite)

}

const onClickPlus = () => {

setIsAdded(!isAdded)

}

const DelProduct = () => {

delProduct()

}

const navigate = useNavigate()

console.log(navigate)

return (

<Col md={3}>

<div className='products'>

<div className='product-wrapper'>

<div className='product'>

{!user.isAdmin ? (

null

):(

<div className='close'>

<CloseButton onClick={DelProduct} />

</div>)}

<div className='favorite' onClick={onClickFavorite}>

<img

onClick={() => FavoriteStore.addFavorite(product)}

src={

isFavorite ? '/img/heart-liked.svg' : '/img/heart-unliked.svg'

}

alt='unliked'

/>

</div>

<img

className='cu-p'

width={133}

height={112}

src={'http://localhost:5000/' + product.img}

alt='a'

onClick={() => navigate(PRODUCT\_ROUTE + '/' + product.id)}

/>

<p

className='cu-p'

onClick={() => navigate(PRODUCT\_ROUTE + '/' + product.id)}

>

{product.name}

</p>

<div

className='d-flex justify-between align-center'

style={{

justifyContent: 'space-between',

marginRight: '10px',

}}

>

<div className='d-flex flex-column'>

<span>Цена:</span>

<b>{product.price} руб.</b>

</div>

<div onClick={onClickPlus}>

<img

onClick={() => CartStore.addProduct(product)}

src={isAdded ? '/img/btn-checked.svg' : '/img/btn-plus.svg'}

className=' cu-p'

alt='plus'

/>

</div>

</div>

</div>

</div>

</div>

</Col>

)

}

export default ProductItem

1. Файл ProductList.js

import React, { useContext } from 'react'

import { observer } from 'mobx-react-lite'

import { Context } from '../index'

import { Row } from 'react-bootstrap'

import ProductItem from './ProductItem'

const ProductList = observer(() => {

const { product } = useContext(Context)

return (

<Row className='d-flex'>

{product.products.map(product => (

<ProductItem key={product.id} product={product} />

))}

</Row>

)

})

export default ProductList

import axios from "axios";

const $host = axios.create({

baseURL: 'http://localhost:5000/'

})

const $authHost = axios.create({

baseURL: 'http://localhost:5000/'

})

const $adminHost = axios.create({

baseURL: 'http://localhost:5000/'

})

const authInterceptor = config => {

config.headers.authorization = `Bearer ${localStorage.getItem('token')}`

return config

}

$authHost.interceptors.request.use(authInterceptor)

export {

$host,

$authHost,

$adminHost

}