МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-изданий)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА:**

по дисциплине «Программирование серверных кроссплатформенных приложений»

Тема «WEB-приложение “Интернет-магазин спорттоваров”»

Исполнитель

студентка 2 курса группы 5 Дубень Полина Васильевна

(Ф.И.О.)

Руководитель работы Сенюк В.К.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель

(подпись)

Минск 2020

Содержание

[ВВЕДЕНИЕ 3](#_Toc41291345)

[1 Аналитический обзор литературы 4](#_Toc41291346)

[2 Анализ требований к программному средству и разработка функциональных требований 8](#_Toc41291347)

[3 Проектирование программного средства 10](#_Toc41291348)

[3.1 Реализация каталога 12](#_Toc41291349)

[3.2 Работа с базой данных 13](#_Toc41291350)

[4 Реализация программного средства 14](#_Toc41291351)

[4.1 Структура проекта 14](#_Toc41291352)

[4.2 Реализация добавления, изменения и удаления записей 16](#_Toc41291353)

[4.3 Реализация Web-socket 17](#_Toc41291354)

[5 Тестирование, проверка работоспособности и анализ полученных результатов 18](#_Toc41291355)

[5.1 Соответствие техническому заданию 18](#_Toc41291356)

[5.2 Тестирование программного средства 22](#_Toc41291357)

[6 Руководство по установке и использованию 25](#_Toc41291358)

[6.1 Руководство для неаутентифицированного пользователя 26](#_Toc41291359)

[6.2 Руководство для аутентифицированного пользователя 27](#_Toc41291360)

[6.3 Руководство для работника магазина 29](#_Toc41291361)

[ЗАКЛЮЧЕНИЕ 31](#_Toc41291362)

[СПИСОК ЛИТЕРАТУРЫ 32](#_Toc41291363)

[Приложение А 33](#_Toc41291364)

[Приложение Б 34](#_Toc41291365)

[Приложение В 35](#_Toc41291366)

[Приложение Г 36](#_Toc41291367)

# ВВЕДЕНИЕ

В современном мире информационные технологии позволяют автоматизировать и упростить работу во многих сферах деятельности человека. Базы данных, интернет, быстрый поиск, автоматическое вычисление данных позволяют упростить работу магазина. Это и является целью моего курсового проекта ­– создание простого и удобного программного средства, позволяющего клиентам ознакомиться с ассортиментом магазина, выбрать товар, положить его в корзину и заказать. Также работники магазина получают доступ к заказам, а значит и возможность организовать доставку или самовывоз товара.

Данное Web-приложение реализовано на платформе Node JS, взаимодействует с базой данных, размещено на платформе «Heroku». Отображение, бизнес логика и база данных отделены друг от друга в целях простого расширения, архитектура приложения – «Model-View-Controller». Управление сайтом интуитивно понятно и удобно.

Данный тип приложений в данный момент актуален, многие магазины открывают свои сайты в целях увеличения прибыли и расширения потребительской аудитории.

Для выполнения данного программного средства были поставлены следующие задачи:

1. Обзор литературы по теме, примеры похожих программных средств, анализ их с критической стороны;
2. Установление требований к программному средству, описание функциональности;
3. Проектирование, создание логических схем работы программного средства;
4. Программирование, отладка модулей проекта, сборка и комплексная отладка. Описание классов, атрибутов и методов;
5. Проведение тестирования программного средства;
6. Разработка руководства по установке и использованию приложения.

Решение каждой из задач отображено соответственно в главах:

1. Аналитический обзор литературы;
2. Анализ требований к программному средству и разработка функциональных требований;
3. Проектирование программного средства;
4. Создание (реализация) программного средства;
5. Тестирование, проверка работоспособности и анализ полученных результатов;
6. Руководство по установке и использованию.

**1 Аналитический обзор литературы**

Для составления технических требований к проекту был проведен анализ похожих сайтов. В процессе анализа были рассмотрены достоинства и недостатки найденных проектов, на основе которых разрабатывались технические требования для разрабатываемого Web-приложения. Ссылки на материалы, используемые в данной главе, предоставлены всписке литературы.

Приведенный на рисунке 1.1 сайт содержит сразу несколько недостатков.

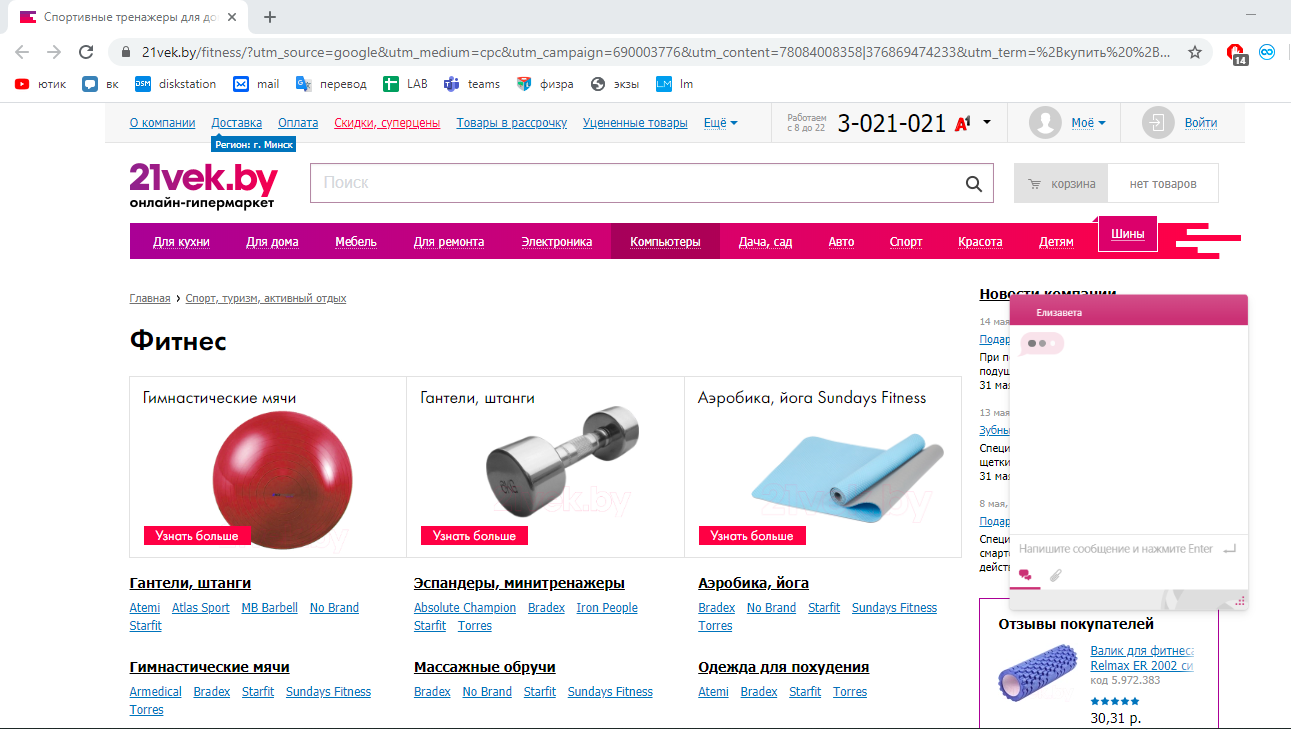


Рисунок 1.1 – Сайт магазина 21vek

Контент перекрывает друг друга, на странице множество разных цветов, не всегда сочетающихся, также здесь много мелких надписей, которые уменьшают читабельность и привлекательность страницы. Также сайт довольно долго откликается на запросы, временами загрузка превышает 5 секунд.

Отображенный на рисунке 1.2 сайт компании «Зона спорта» имеет больше плюсов. Так как компания специализируется на продаже конкретно спорттоваров, сайт более специализирован и стилизирован под спортивный контент. Также здесь очень большой каталог, что можно отнести как к плюсам, так и минусам. Минус в том, что навигация по сайту становится затруднительной, множество пунктов меню, которые имеют ещё большое количество подпунктов.

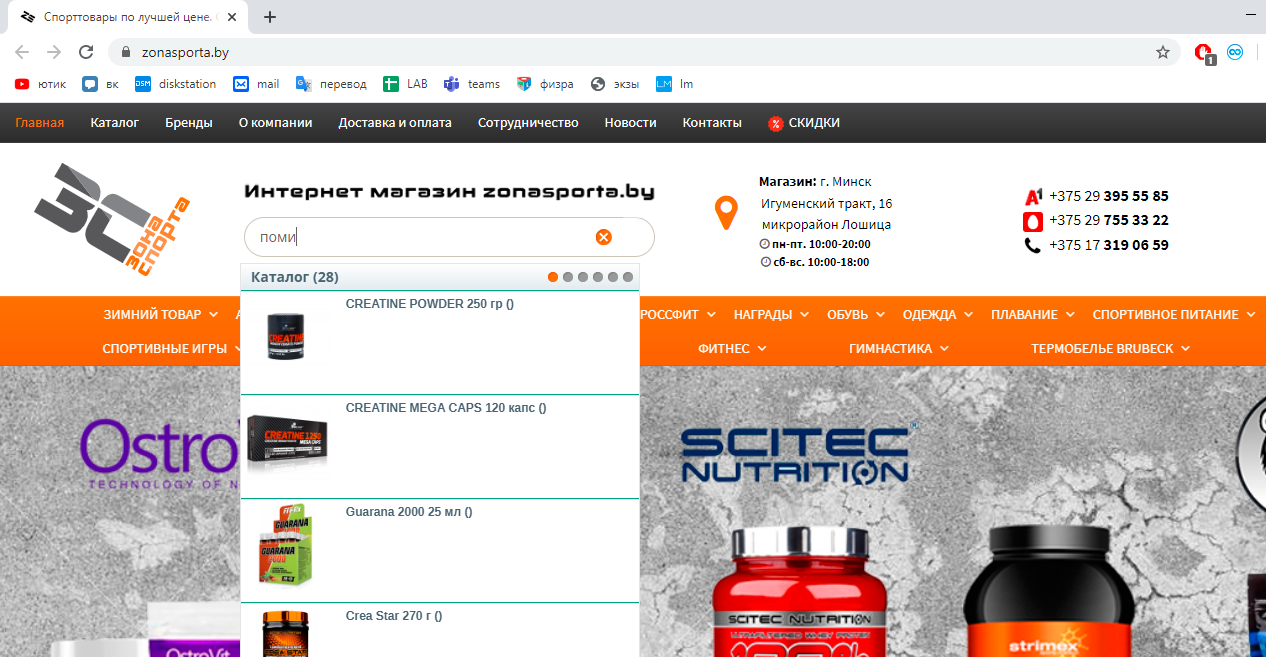


Рисунок 1.2 – Сайт компании «Зона спорта»

На сайте присутствует поиск, но алгоритм поиска выводит много товаров, которые не соответствуют критерию поиска. Также сайт не отображает отзывы. В процессе ознакомления с каталогом, был обнаружен товар, у которого был рейтинг, но не было отзывов.

На рисунке 1.3 отображено веб-приложение компании bigsport.

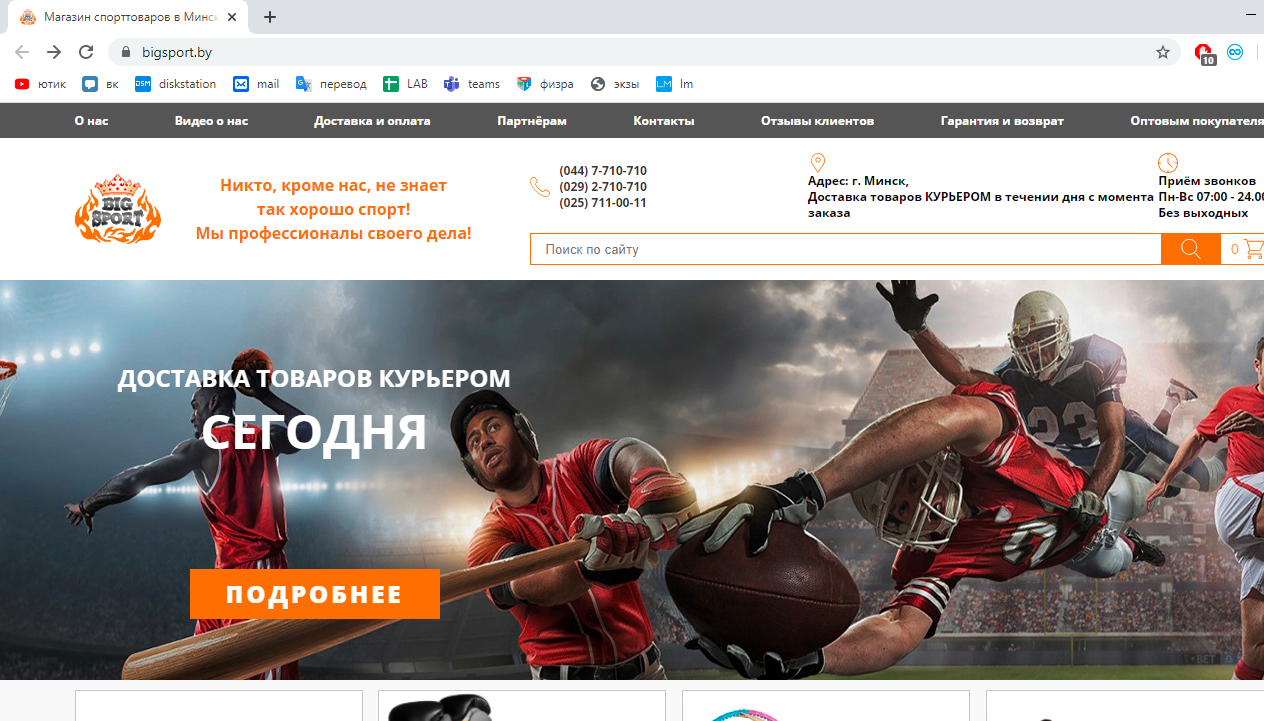


Рисунок 1.3 – Сайт магазин bigsport

Ниже представлен список недостатков этого приложения:

* На странице большое количество ненужного контента и рекламы;
* В каталоге не отображаются цены товаров;
* Приложение не позволяет оставлять отзывы к товарам;
* Сложная навигация по сайту.

Программное средство, отображенное на рисунке 1.4, является лучшей программой, которая была найдена в сети на заданную тему. В ней реализованы отзывы и обновление данных без перезагрузки страницы, интуитивно понятный дизайн. Присутствует разделение функционала в зависимости от учётной записи, справочная информация.

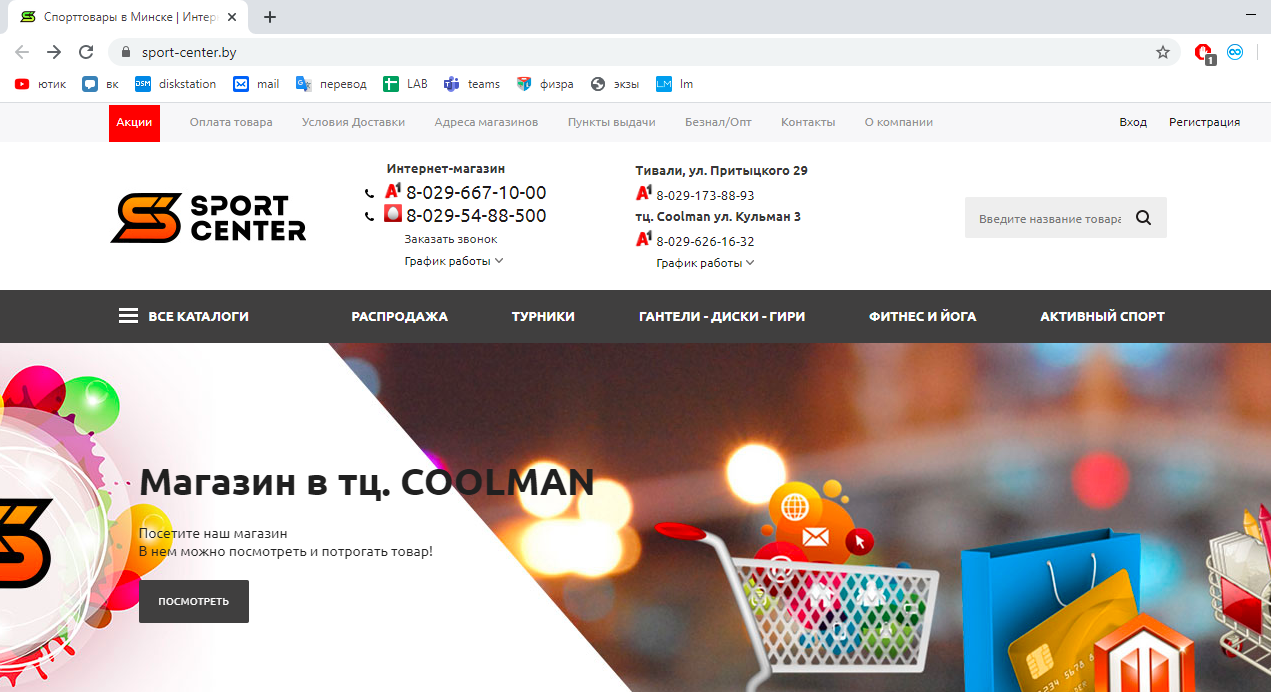


Рисунок 1.4 – Веб-приложение sport-center

На рисунке 1.5 продемонстрировано веб-приложение со схожим функционалом. В отличие от прошлого, данный сайт оснащён меньшим функционалом, однако это делает его проще в управлении. Навигация по сайту интуитивно понятна, цветовые решения не вызывают отторжения. Также как и первый рассмотренный сайт, приложение довольно долго отвечает на запросы.

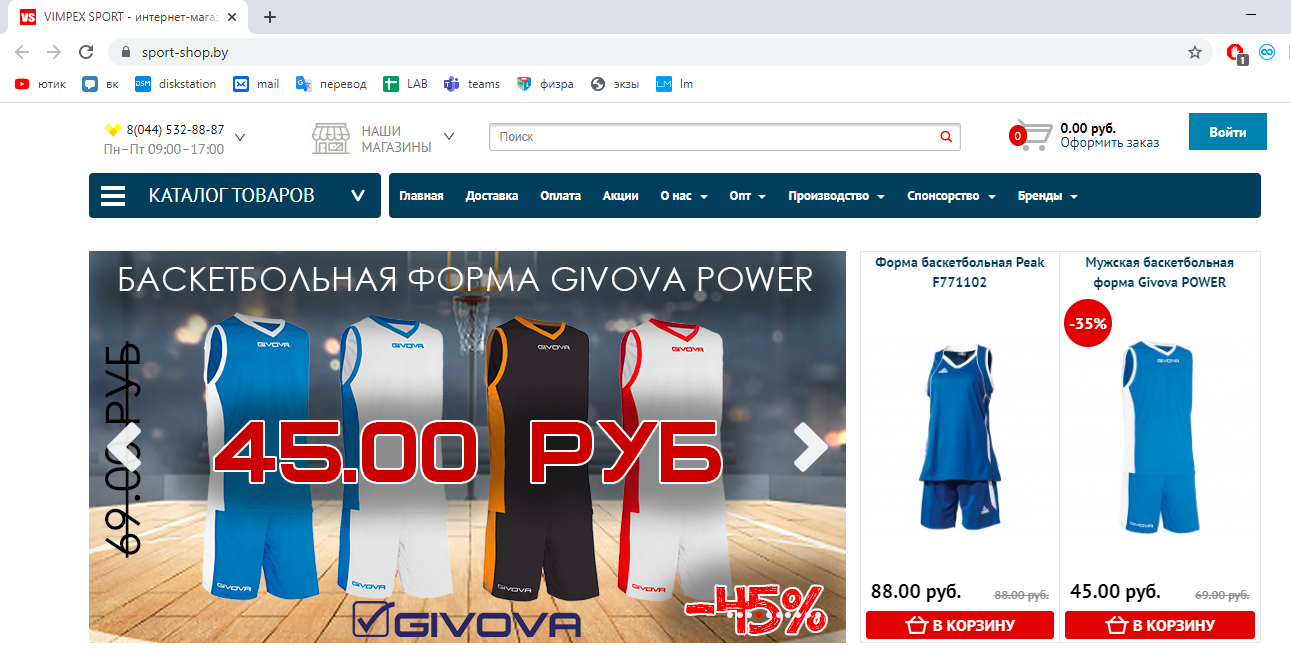


Рисунок 1.5 – Сайт магазина Vimpex Sport

Проанализировав программы на схожую тематику, были разработаны технические требования к разработке веб-приложения. Приложение должно:

* реализовывать валидацию вводимых данных;
* иметь интуитивно понятное и простое управление;
* обращаться к базе данных, успешно извлекать, изменять и дополнять данные, осуществлять поиск по базе и автоматическую запись некоторых полей;
* иметь различный функционал в зависимости от учётной записи. Работник магазина может добавлять и изменять товары, клиент может просматривать их и добавлять в заказ, оставлять рецензии;
* выглядеть аккуратно, не иметь лишней информации;
* быстро отвечать на запросы (не более 5 секунд).

Эти пункты и составляют техническое задание разрабатываемого проекта.

# 2 Анализ требований к программному средству и разработка функциональных требований

Приложение должно иметь разделение ролей. Для каждой роли были разработаны UML-диаграммы вариантов использования. При первом открытии сайта пользователь не аутентифицирован. Диаграмма вариантов использование неаутентифицированного пользователя отображена на рисунке 2.1.



Рисунок 2.1 – Диаграмма использования неаутентифицированного пользователя

Как видно из диаграммы, неаутентифицированный пользователь может зарегистрироваться, аутентифицироваться и просматривать товары. При попытке добавления товара в корзину, пользователя будет перенаправлять на страницу регистрации и аутентификации. При попытке зайти на страницу, требующую аутентификации (например, путём явного ввода url), будет отображаться сообщение об ошибке с предложением войти в аккаунт.

Роль каждой учётной записи определена в базе данных. Создать администратора на сайте невозможно. Итак, зарегистрировав и войдя в аккаунт, пользователь получает роль аутентифицированного пользователя. Диаграмма вариантов использования для этой роли отображена на рисунке 2.2.

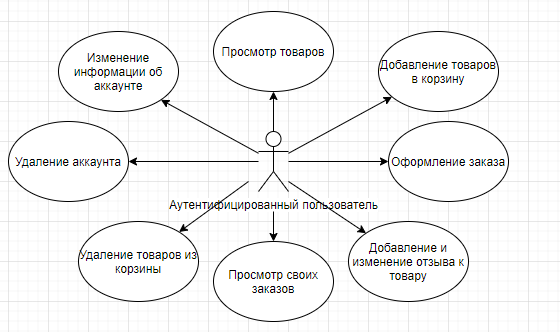


Рисунок 2.2 – Диаграмма использования аутентифицированного пользователя

В приложении есть роль администратора или же работника магазина. Для этой роли диаграмма вариантов использования отображена на рисунке 2.3.



Рисунок 2.3 – Диаграмма использования для работника магазина

На основе данных UML‑диаграмм и словесного описания функциональных требований строится вся бизнес логика программы.

# 3 Проектирование программного средства

В предыдущей главе были разработаны технические требования к программному средству. Для реализации разрабатываемого проекта была выбрана технология Node JS и база данных MySQL.

В ходе проектирования были разработаны на языке Unified Modeling Language следующие диаграммы: диаграмма деятельности для участка программного средства, отвечающего за оставление пользователем комментария, и предоставленные в предыдущей главе диаграммы вариантов использования.

Схема базы данных для разрабатываемого программного средства отображена в приложении А. Рассмотрим её подробнее.

Схема отображает таблицы базы данных и связи между ними.

Как видно из схемы, в приложении используются 5 таблиц: comments, users, orders, orderedproducts и products.

Таблица users содержит информацию о пользователях. Она включает в себя следующие поля:

* id – уникальный идентификатор пользователя;
* surname – фамилия, длиной до 255 символов;
* name – имя пользователя;
* phone – телефон, представлен строкой, не валидируется;
* mail – электронная почта пользователя;
* password – поле, содержащее хеш пароля;
* passwordSalt – поле, содержащее соль для хеширования пароля этого пользователя;
* status – значение 0 или 1, где 0 – пользователь удалён, 1 – активный пользователь;
* role – значение 0 или 1, где 0 – обычный пользователь, 1 – аккаунт работника магазина.

По задумке приложения, при регистрации пользователю необходимо ввести почту и пароль, остальные поля являются необязательными. После регистрации, создаётся запись пользователя в базе данных, автоматически пользователь получает значения role равное 0 и status равное 1. Это означает, что пользователь не является работником магазина и его аккаунт не удалён.

Каталог будет состоять из выборки из таблицы Products. Данная таблица состоит из следующих полей:

* id – уникальный идентификатор товара;
* productName – уникальное имя товара;
* productDescription – описание товара;
* brand – обозначает бренд, которому принадлежит товар;
* productPrice – цена товара, принимает значения от 0.1 до 99999.99;
* imageURL – обозначает URL картинки, на которой изображен товар.

Обязательными для заполнения являются поля названия товара и его цены. Обозначать URL картинки не обязательно, при незаполнении этого поля в каталоге будет отображаться картинка по умолчанию. При добавлении нового товара с картинкой, она будет сохраняться в облаке («Cloudinary»), а полученный URL добавится к записи созданного товара.

При добавлении товара в корзину, создаётся новая запись в таблице orders. Она состоит из следующих полей:

* id – уникальный идентификатор товара;
* orderStatus – текущий статус товара;
* orderDate – дата последнего изменения статуса заказа;
* userId – ссылка на запись в таблице users, указывает на пользователя, которому принадлежит товар.

Ссылка на товар, при этом, помещается в новую запись в таблице orderedproducts, которая имеет следующую структуру:

* id – уникальный идентификатор записи;
* countOfProducts – количество заказанных товаров данного вида в текущем заказе;
* productId – ссылка на товар;
* orderId – ссылка на текущий заказ.

В приложении пользователи имеют возможность оставлять отзывы о товарах. Отзывы помещаются в таблицу comments и имеют следующую структуру:

* id – уникальный идентификатор отзыва;
* commentDate – дата создания или последнего изменения комментария;
* rating – число от 1 до 5, указывающее оценку пользователем товара;
* message – текст отзыва;
* productId – ссылка на товар, под которым пользователь разместил отзыв;
* userId – ссылка на пользователя, который оставил отзыв.

Данная структура базы данных позволит получать всю необходимую информацию в приложении, не имеет лишних таблиц и полей, соответствует третьей нормальной форме.

Приложению для обновления комментариев и рейтинга понадобится второй сервер (web-socket), запуск которого будет осуществляться внутри первого сервера.

## 3.1 Реализация каталога

Теперь подробно рассмотрим логику программы. На диаграмме деятельности, представленной в приложении Б, отображен процесс оставления пользователем комментария. Как видно из диаграммы, для того, чтобы оставить отзыв, пользователь должен быть аутентифицированным.

На основе оценок в каждом отзыве вычисляется рейтинг. Оценки могут быть от 1 до 5, рейтингом товара считается среднее арифметическое всех оценок.

Для отображения рейтинга товара на всех страницах, необходимо при чтении данных из базы данных добавлять объединение таблицы товаров с таблицей комментариев. Затем комментарии передаются в функцию для обработки. Функция считывает рейтинг в каждом комментарии и выдаёт результирующую оценку. Рейтинг обновляется, использую веб-сокет подключение. Подробнее это будет рассмотрено в главе 4 в подразделе о реализации Web-socket.

Если пользователь аутентифицирован, то, возможно, он уже добавил некоторые товары в корзину. В таком случае не нужно отображать кнопку «В корзину», а заменять её на кнопки «+» и «–». Для того, чтобы это реализовать, необходимо добавлять объединение с таблицей заказов.

Таким образом, при входе на страницу каталога приложение должно вычислить, аутентифицировался ли пользователь. Если нет, то достаточно будет выборки таблиц комментариев и продуктов из базы данных. Если же пользователь вошёл в аккаунт, необходимо проверять, есть ли у него заказ, который ещё на стадии добавления товаров в корзину, и если есть, то необходимо отображать кнопки «В корзину» и «+», «–» в зависимости от результата.

## 3.2 Работа с базой данных

В разрабатываемом программном средстве существует возможность добавления, изменения и удаления записей в базе данных.

Допускается редактирование существующих товаров и создание новых, если пользователь вошел под учётной записью работника магазина. При регистрации пользователя, создаётся новая запись в базе данных. Также в приложении есть реализация изменения записи пользователя, что включает добавления новых полей, не заполненных ранее, и изменение старых. Каждый аутентифицированный пользователь имеет возможность добавления комментария и его изменения.

Добавление товаров в корзину автоматически создаёт новую строку в таблице «Orders», а также добавляет запись со ссылкой на выбранный товар в таблицу «OrderedProducts». Изменение количества товаров изменяет значение столбца «countOfProducts» в соответствующей записи. При установлении «countOfProducts» меньше единицы, запись автоматически удаляется из таблицы.

На страницах каталога, корзины, товара, из базы данных достаётся информация о самом товаре, заказах, комментариях к самому товару. Это позволяет отобразить рейтинг товара на основе всех отзывов, выбрать, отобразить кнопку «добавить в корзину» или кнопки «+» и «–», если товар уже находится в корзине.

Для работы с базой данных используется ORM sequelize. Это позволяет абстрагироваться от самой базы данных, даёт возможность легко заменить её. Во время синхронизации с базой данных, sequelize выполняет скрипт создания таблиц, если таблицы не созданы, что позволяет избежать некоторых ошибок при запуске сервера на новой машине. Так как приложение размещено на платформе heroku, строка подключения выбирается исходя из того, задан ли параметр «process.env.CLEARDB\_DATABASE\_URL». Если данный параметр задан, то берётся строка подключения из этого параметра (подключение к базе данных на heroku, иначе берётся строка подключения к базе данных на локальной машине).

# 4 Реализация программного средства

## 4.1 Структура проекта

На рисунке 4.1.1 отображена структура разрабатываемого программного средства.

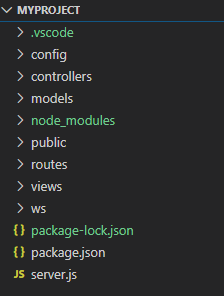


Рисунок 4.1.1 – Структура проекта

В директории «controllers» содержатся контроллеры. Контроллеры обеспечивают взаимодействие пользователя и системы, обрабатывая запросы. Они используют модель и представления для реализации необходимого действия.

Директорий «models» содержит файлы с классами-сущностями базы данных. Приложение использует ORM-библиотеку «sequelize» для взаимодействия с базой данных. Конфигурацию подключения модель импортирует из файла config.js в директории «config». В качестве базы данных используется MySQL. Благодаря правильно прописанной модели, изменить базу данных не сложно, достаточно указать другую конфигурацию. Структура директория «models» представлена на рисунке 4.1.2.

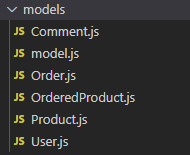


Рисунок 4.1.2 – Структура директория «models»

Директорий содержит файлы моделей (Order, Product, Comment и др.), а также файл model.js, который импортирует все модели, собирает их в один объект, синхронизирует с базой данных и экспортирует.

В директории «views» содержатся представления. Представления используют «слой» main.hbs, который включает все необходимые файлы-библиотеки, вставляет на каждую страницу header и footer, которые являются частичными представлениями. Для реализации этого необходимо зарегистрировать в главном файле partials с указанием директория, установить hbs, указать layout по умолчанию (main.hbs). Страницы реализованы с использованием handlebars. Структура данного директория представлена на рисунке 4.1.3.

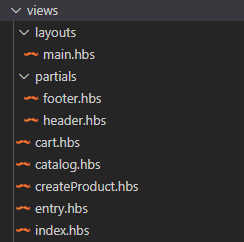


Рисунок 4.1.3 – Структура директория «views»

Директорий «ws» содержит файлы для управление Web-сокетами.

Директорый «routes» содержит роутеры, которые реализуют маршрутизацию. Каждый роутер зарегистрирован в главном файле приложения.

Наконец, директорий «public» содержит статические файлы, такие как .css, .js файлы, шрифты, иконки.

Приложение состоит из трёх основных частей: представлений, контроллеров и модели данных. Таким образом, в разрабатываемом приложении реализована схема «Model-View-Controller».

## 4.2 Реализация добавления, изменения и удаления записей

В разрабатываемом программном средстве есть возможность добавлять, изменять и удалять записи в базе данных.

В общем виде добавление записей в базу данных осуществляется следующим образом:

* пользователь заполняет поля для ввода информации;
* пользователь нажимает на кнопку подтверждения;
* проверяется заполнение обязательных полей. Если обязательные поля не заполнены, то выводится соответствующее сообщение, информация не заносится в базу данных;
* проводится валидация данных. Валидация осуществляется для чисел, в частности цены товара, почты.
* если все данные валидны, создаётся новая запись в базе данных.

Создание новой записи осуществляется путём вызова метода create у класса модели. В этот метод передаётся объект, в котором в формате ключ-значение хранятся поля и значения, которыми они инициализируются.

Этот алгоритм применяется при каждом добавлении записи в базу данных.

Изменение записи также использует этот алгоритм. Отличие лишь в том, что вместо метода create вызывается метод update, при этом в него не обязательно передавать значения обязательных полей, а лишь те, которые необходимо изменить.

Удаление записи реализуется вызовом метода destroy у класса модели. В параметре можно передавать условие удаление, в частности id или имя, которое присвоено записи для удаления.

## 4.3 Реализация Web-socket

Веб-сокеты в данном приложении нужны для обновления рейтинга и появления и изменения комментариев без перезагрузки страницы.

Реализация серверной части веб-сокетов представлена в приложении В. В данном коде присутсвует класс RatingUpdateSubscribers, который содержит список веб-сокет клиентов, а также функции по добавлению и удалению из него. Веб-сокет клиентами являются страницы, на которых отображается рейтинг товара, а это каталог, корзина, страница заказа, страница товара. При загрузке каждой из этих страниц создается подключение к веб-сокет серверу, при этом массивом отсылаются все названия товаров, которые необходимо отслеживать.

При появлении нового комментария вызывается функция WSNotify. Она вычисляет новый рейтинг для товара, у которого появился новый комментарий, а затем передаёт всю необходимую информацию (имя товара, новый рейтинг, комментарий и пользователя, который его оставил) в функцию notifyRatingUpdate. Эта функция в свою очередь сверяет список клиентов и отсылает всем клиентам, у которых на странице расположен этот товар, информацию об изменении.

Код веб-сокет клиента представлен в приложении Г. Первым делом, приложение вычисляет адрес веб-сокет сервера и создаёт подключение. Потом при открытии веб-сокета осуществляется поиск всех товаров на странице и результат помещается в массив, а после этого отправляется сообщение о подписке на обновления. При закрытии происходит отправка сообщения об отписке соответственно. При получении сообщения от веб-сокет сервера происходит его обработка, а в частности ищется элемент radiobutton, который отвечает за соответствующую цифру рейтинга, а затем отмечается.

Таким образом происходит обновление рейтинга без обновления страницы.

# 5 Тестирование, проверка работоспособности и анализ полученных результатов

## 5.1 Соответствие техническому заданию

Разработанное программное средство полностью соответствует составленному для него техническому заданию. Был реализован весь описанный функционал, также добавлены некоторые дополнительные функции.

Программа функционирует без ошибок.

В качестве проверки корректного функционирования программы рассмотрим классический сценарий использования приложение. На рисунке 5.1.1 представлена главная страница приложения.

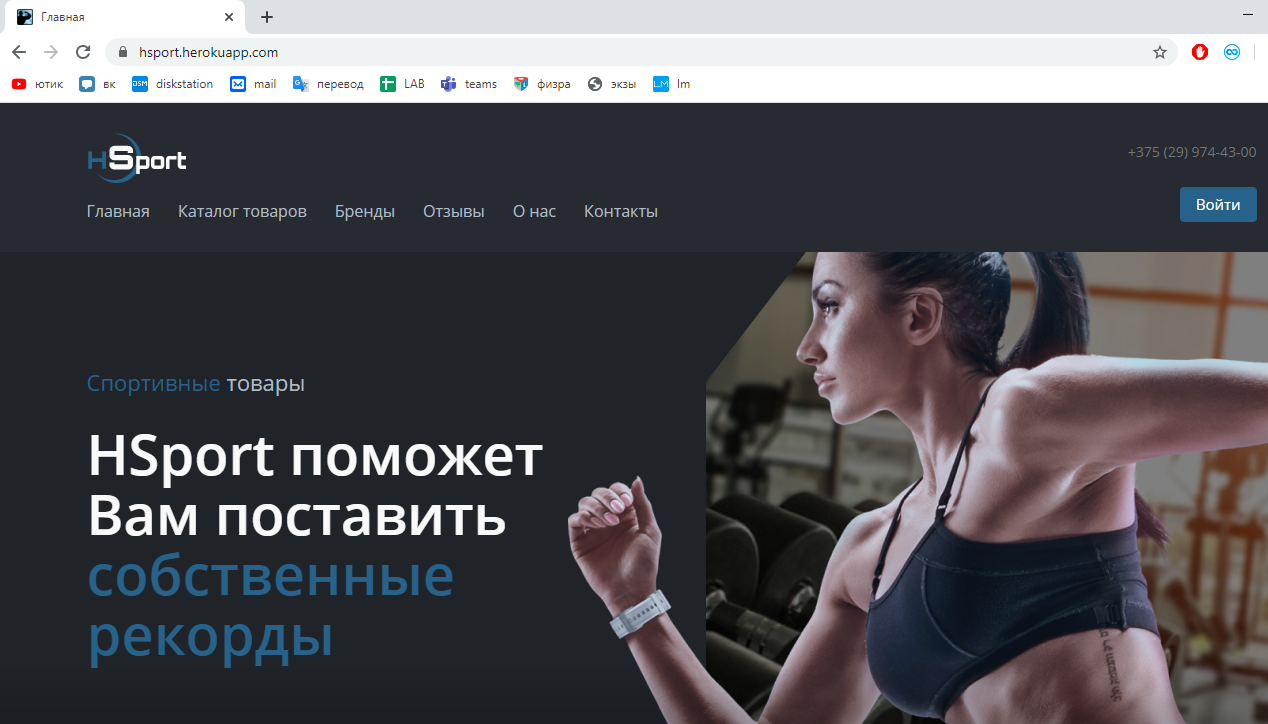


Рисунок 5.1.1 – Главная страница приложения

После нажатия на кнопку «Войти» пользователь оказывается на странице регистрации и аутентификации(рисунок 5.1.2).

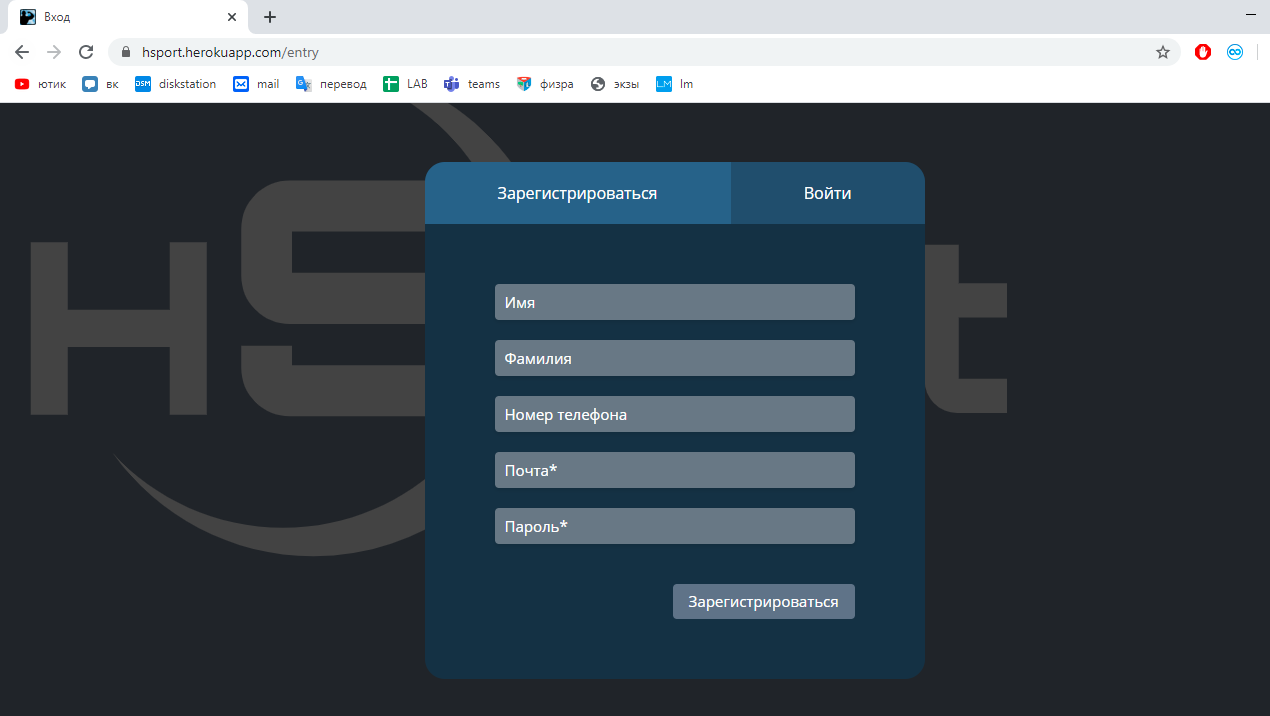


Рисунок 5.1.2 – Страница регистрации и аутентификации

После заполнения полей корректной информации и нажатия на кнопку «Зарегистрироваться», запись пользователя появляется в базе данных (рисунок 5.1.3).

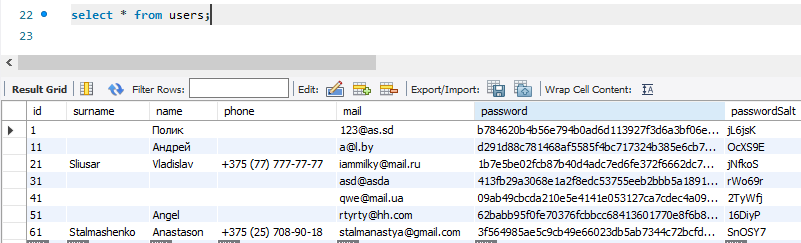


Рисунок 5.1.3 – Записи пользователей в базе данных

После входа в аккаунт (обычного пользователя), открываем каталог, который представлен на рисунке 5.1.4.

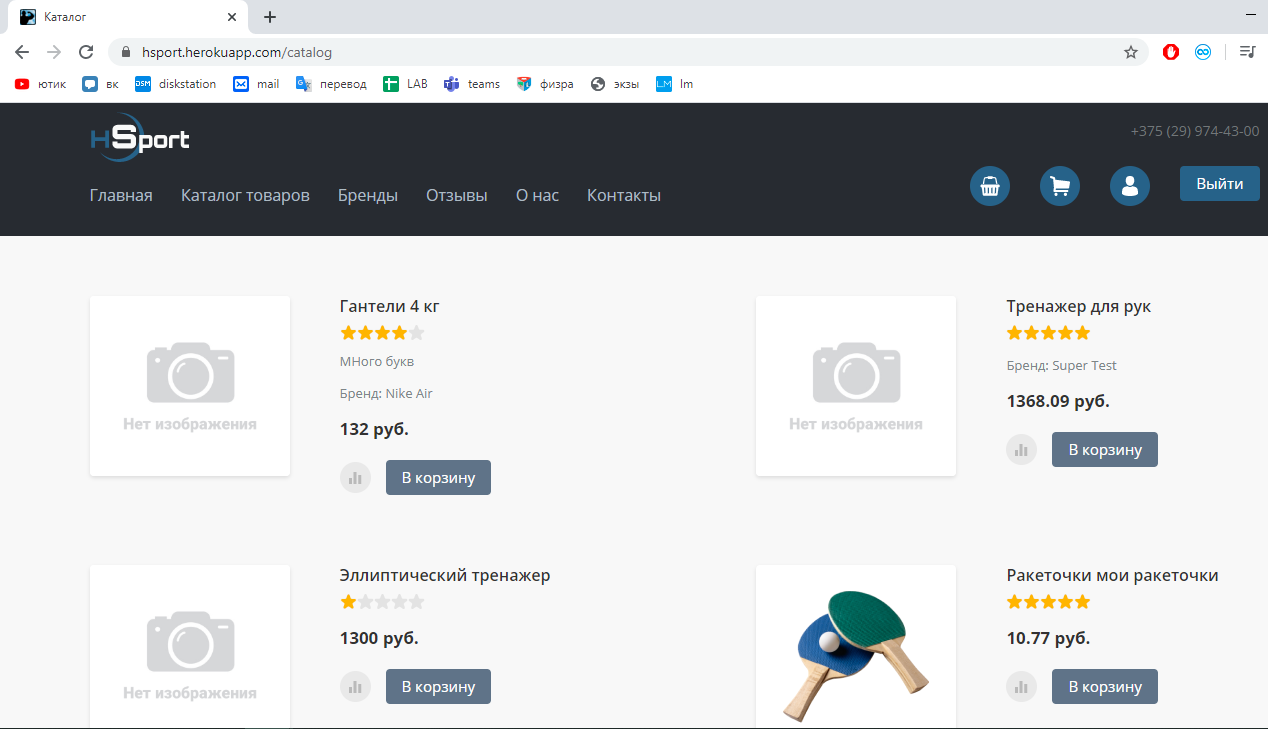


Рисунок 5.1.4 – Каталог

На странице отображаются товары, которые содержатся в базе данных. Картинки отображаются по url. При нажатии на «В корзину», на месте этой кнопки появляются другие кнопки, отображенные на рисунке 5.1.5.

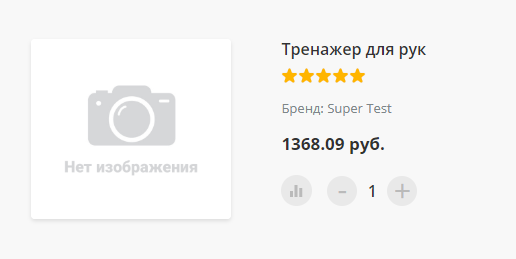


Рисунок 5.1.5 – Товар после добавления в корзину

Отображается количество товаров в корзине. При нажатии на кнопки «+» и «–», количество данного товара в корзине соответственно увеличивается и уменьшается. Если количество становится равным нулю, вновь отображается кнопка «В корзину».

После перехода в корзину, пользователь видит страницу, представленную на рисунке 5.1.6.

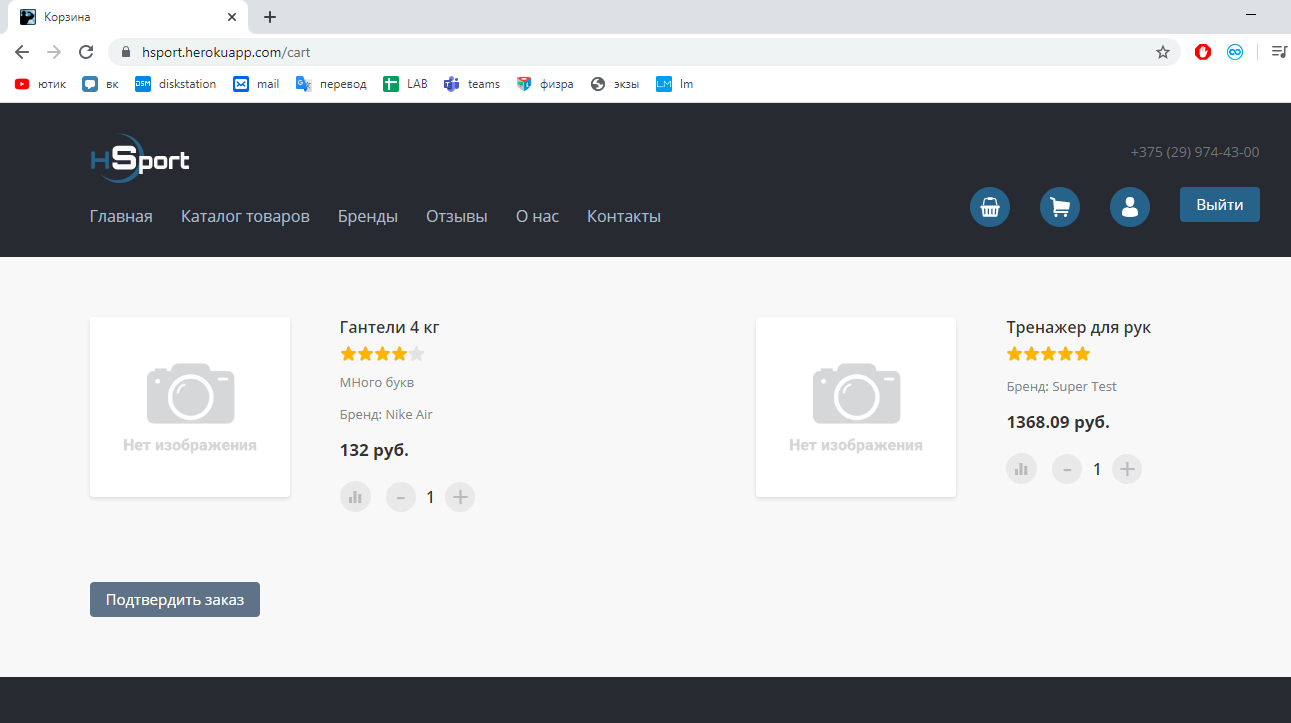


Рисунок 5.1.6 – Корзина

После нажатия на кнопку «Подтвердить заказ» заказ получает статус «Обрабатывается», а пользователя перенаправляет на страницу с его заказами, отображенную на рисунке 5.1.7.

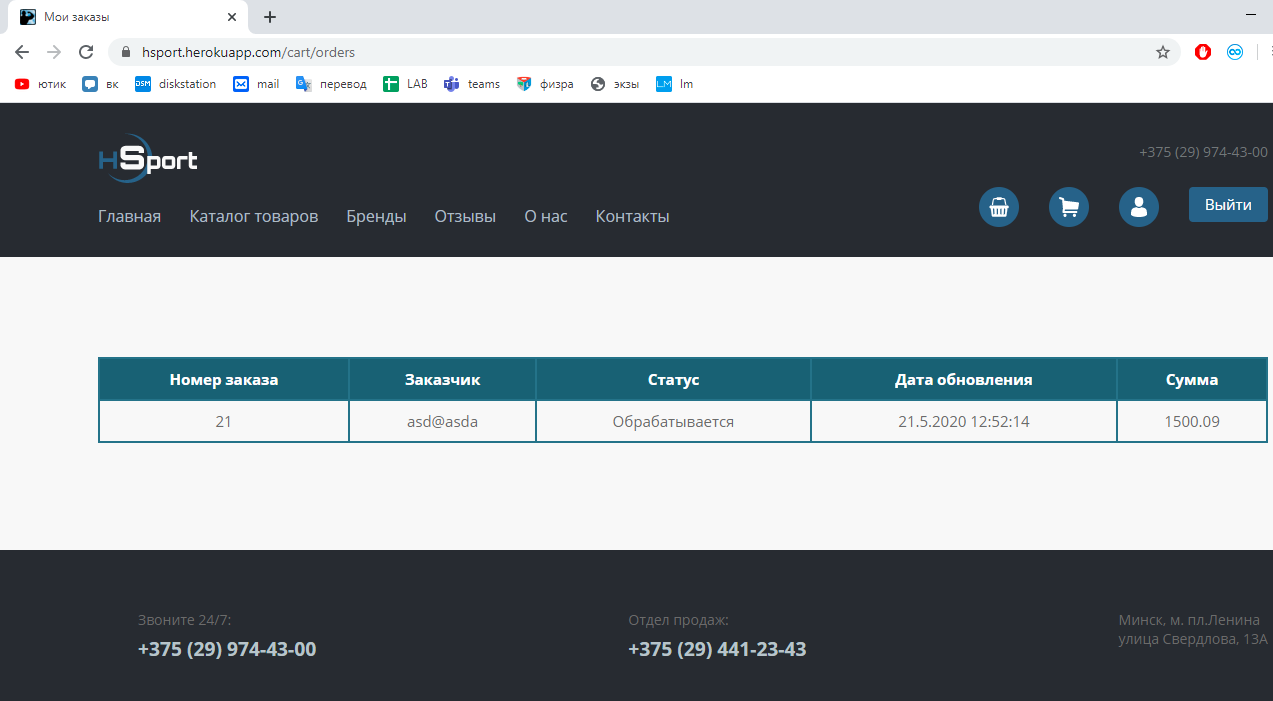


Рисунок 5.1.7 – Все заказы

Строки таблицы являются ссылками, перенаправляют на страницу, схожую по структуре со страницей на рисунке 5.1.7.

При нажатии на иконку «Мой профиль», открывается страница редактирования профиля. При нажатии на кнопку «Выйти», пользователь выходит из аккаунта.

## 5.2 Тестирование программного средства

В разработанном программном средстве предусматриваются многие негативные сценарии.

Создавая новую запись, предназначенную для помещения в базу данных, при незаполненных обязательных полях на форме появляются сообщения, продемонстрированные на рисунке 5.2.1.

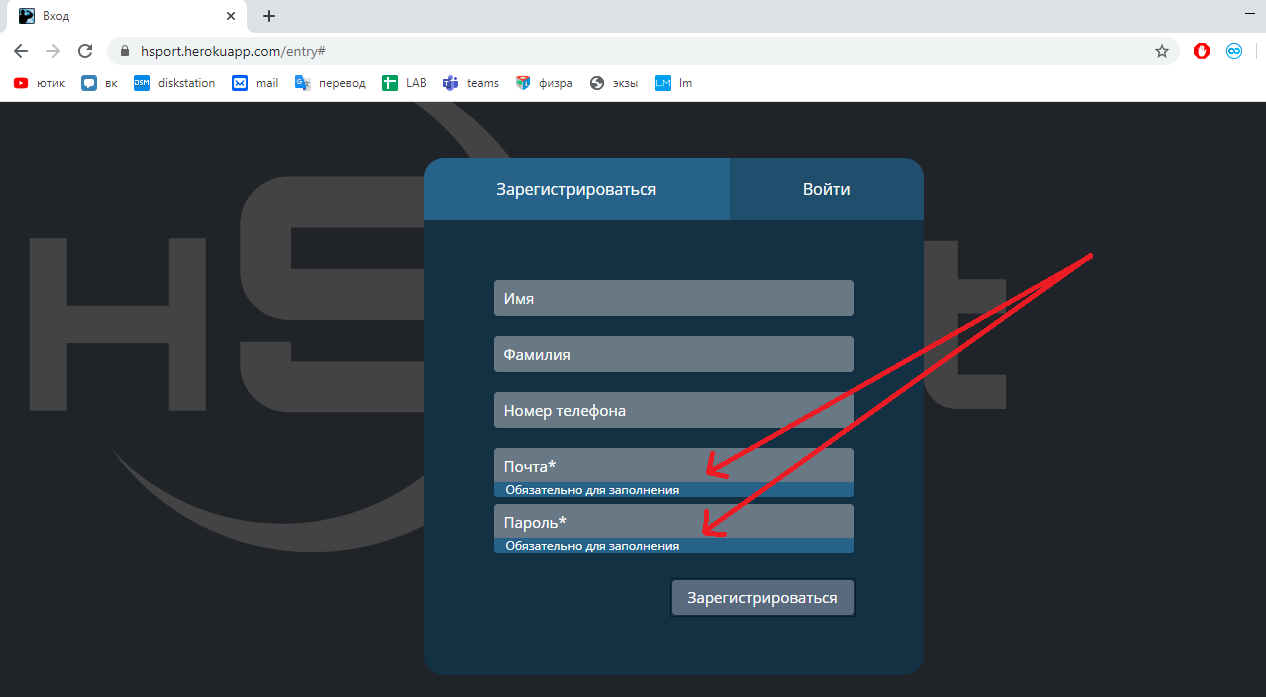


Рисунок 5.2.1 – Сообщение, возникающее при пустых обязательных полях

При попытке зарегистрировать пользователя с почтой, которая уже имеется в базе данных, выводится сообщение, отображенное на рисунке 5.2.2.

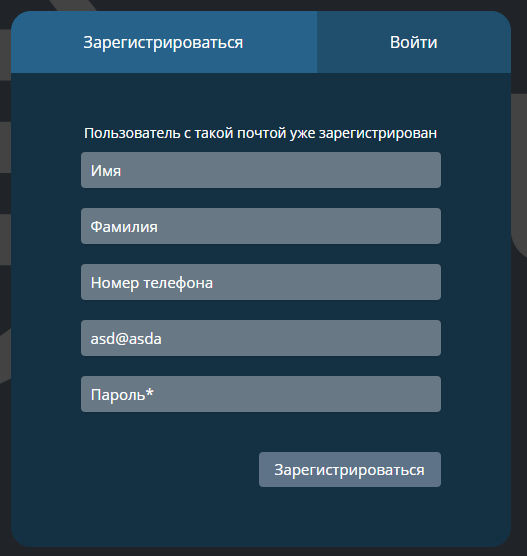


Рисунок 5.2.2 – Сообщение о необходимости выбора записи

В приложении присутствует аутентификация пользователя. В базе данных хранятся допустимые комбинации значений логина и пароля. Если при входе в систему указываются неверные данные, выводится сообщение об ошибке. Одно из таких сообщений продемонстрировано на рисунке 5.2.3.

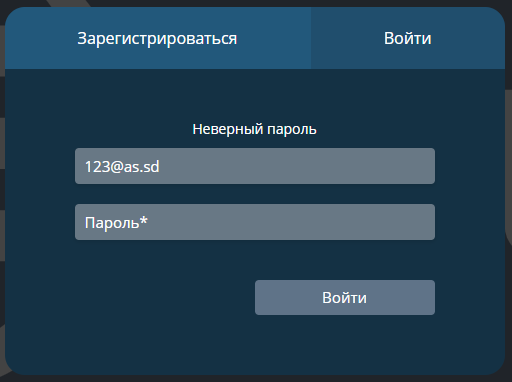


Рисунок 5.2.3 – Сообщение о неправильно введенных данных пользователя

При попытке отправления отзыва, в котором не будет выставлен рейтинг товара, будет выведено сообщение, представленное на рисунке 5.2.4.

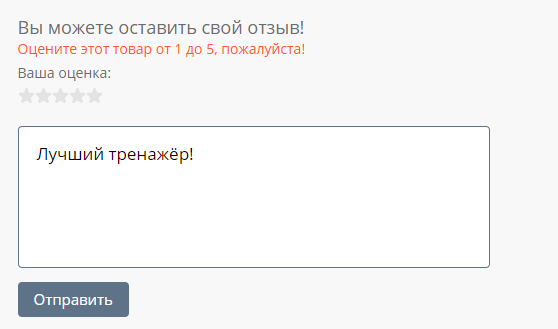


Рисунок 5.2.4 – Сообщение о недостаточности введенных данных

При попытке входа неаутентифицированного пользователя на страницу, к которой доступ имеют только аутентифицированные пользователи, отображается страница с сообщением об ошибке, отображенная на рисунке 5.2.5.

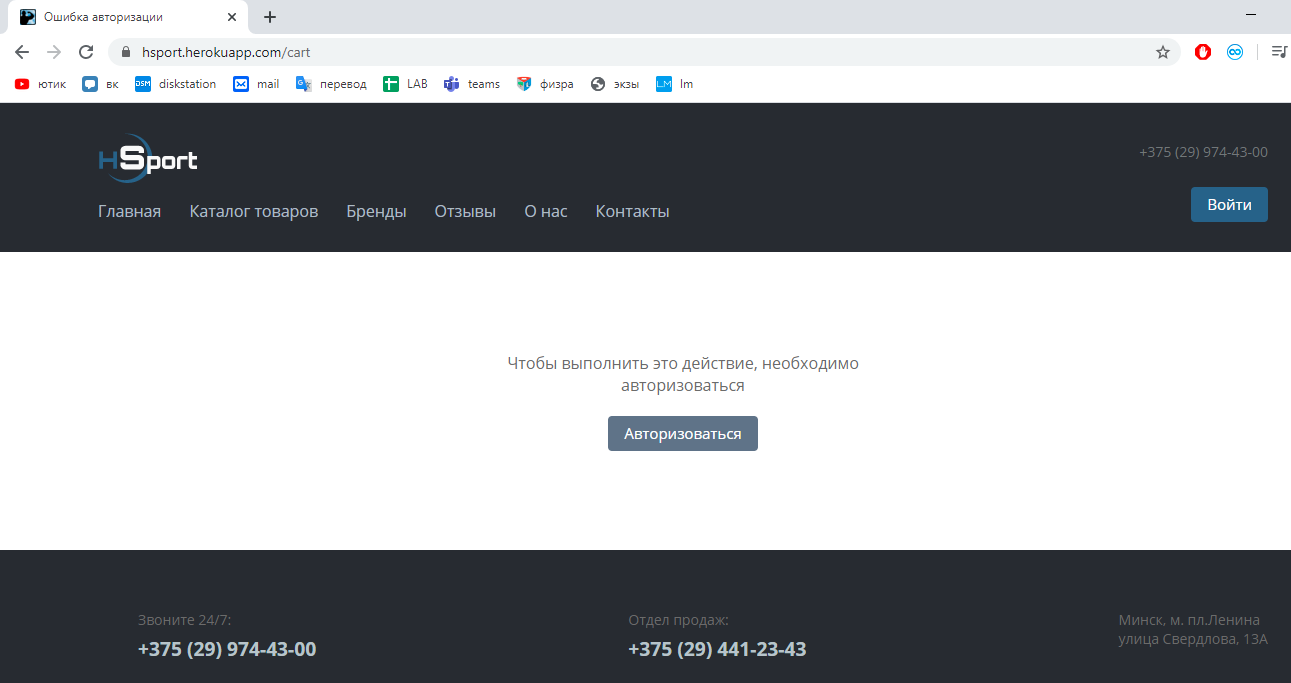


Рисунок 5.2.5 – Сообщение о некорректности введенных данных

Такими страницами являются страницы «Корзина» и «Мои заказы». Страница «Корзина» также недоступна для администратора, при этом отображается сообщение «Чтобы выполнить это действие, необходимо зайти под записью обычного пользоваться».

Таким образом, программное средство прошло разработанный в данной главе ограниченный набор тестов. Результаты тестирования оказались успешны, ни в одной ситуации приложение не создало неконтролируемого исключения. Можно сделать вывод, что программное средство работает корректно и готово к использованию.

# 6 Руководство по установке и использованию

Для корректной работы приложения необходимо наличие компьютера со всеми необходимыми для его работы компонентами (процессором, видеокартой, оперативной памятью, монитором и т.д.), браузер-клиент, а также подключение к сети Internet.

Фактически, установка для разработанного в данном курсовом проекте приложения не требуется. Если существует запустить приложение в локальной сети, нужно установить конфигурацию приложения, в частности подключение к базе данных, а также создать базу данных «hsport». При запуске приложения, таблицы автоматически создадутся.

Для того чтобы начать использовать приложение, нужно ввести в url ссылку «https://hsport.herokuapp.com/». Откроется главная страница приложения.

Чтобы иметь возможность заказать товары и оставлять отзывы, необходимо войти в аккаунт.

Для создания аккаунта необходимо ввести почту и пароль, при этом почта должна быть уникальной.

При входе в аккаунт, необходимо ввести данные, указанные при регистрации.

Если необходимо получить роль администратора, необходимо обратиться к инженеру-программисту, чтобы отредактировать базу данных, установить роль у желаемого пользователя в значение администратора.

## 6.1 Руководство для неаутентифицированного пользователя

В разработанном веб-приложении пользователь является неаутентифицированным, если токен, хранящийся в куках браузера, не является действующим, или же если этого токена вообще нет.

Функционал неаутентифицированного пользователя представляет собой следующие возможности. Пользователь может просматривать главную страницу и каталог магазина, открывать страницы товаров, видеть отзывы для каждого отдельного товара. Также пользователю будет отображаться кнопка «В корзину» на странице каталога, однако при нажатии на неё пользователь будет перенаправлен на страницу регистрации и авторизации. Формы регистрации и авторизации отображены на рисунках 5.2.2 и 5.2.3.

Для успешной регистрации пользователю необходимо ввести пароль и почту, обязательно соответствующую шаблону «a@a», где а – произвольная строка не короче одного символа. Остальные поля на форме регистрации являются необязательными и не валидируются, только поле телефона имеет шаблон для заполнения, но при неполном заполнении не выдаёт ошибку при регистрации. Регистрация осуществляется после нажатия на кнопку «Зарегистрироваться».

Пользователь также может войти в созданный аккаунт, заполнив поля на форме аутентификации. Чтобы аутентификация прошла успешно, пользователю необходимо указать почту и пароль, которые были указаны при регистрации.

Если неаутентифицированный пользователь совершит попытку выполнить действия или открыть страницы, которые доступны только аутентифицированным пользователям, запрос будет перенаправлен на страницу с соответствующим сообщением.

## 6.2 Руководство для аутентифицированного пользователя

После аутентификации, у пользователя появляются в шапке сайта новые кнопки: «Мои заказы», «Корзина», «Мой профиль» и «Выйти». Все они расположены на рисунке 6.2.1 в соответствующем порядке.



Рисунок 6.2.1 – Кнопки-действия для аутентифицированного пользователя

При нажатии на кнопку «Выйти», пользователь выйдет из аккаунта и окажется на странице регистрации и авторизации.

Кнопка «Мои заказы» перенаправит пользователя на страницу, где будут отображена таблица со всеми его заказами. Эта страница отображена на рисунке 5.1.7. При нажатии на строку таблицы, пользователь окажется на странице заказа, на которой будут отображены все товары из этого заказа, а также их количество в заказе.

Кнопка «Корзина» перенаправит пользователя на страницу с товарами, которые он поместил в корзину. Если данных товаров ещё нет, то на странице отобразится соответствующее сообщение. Когда пользователь собрал в корзину всё, что он хочет, он может нажать на кнопку «Подтвердить заказ», и заказ начнёт обрабатываться работниками магазина. Статус заказа пользователь может отслеживать на странице заказов.

При нажатии кнопки «Мой профиль», пользователю открывается страница для редактирования профиля (рисунок 6.2.2).

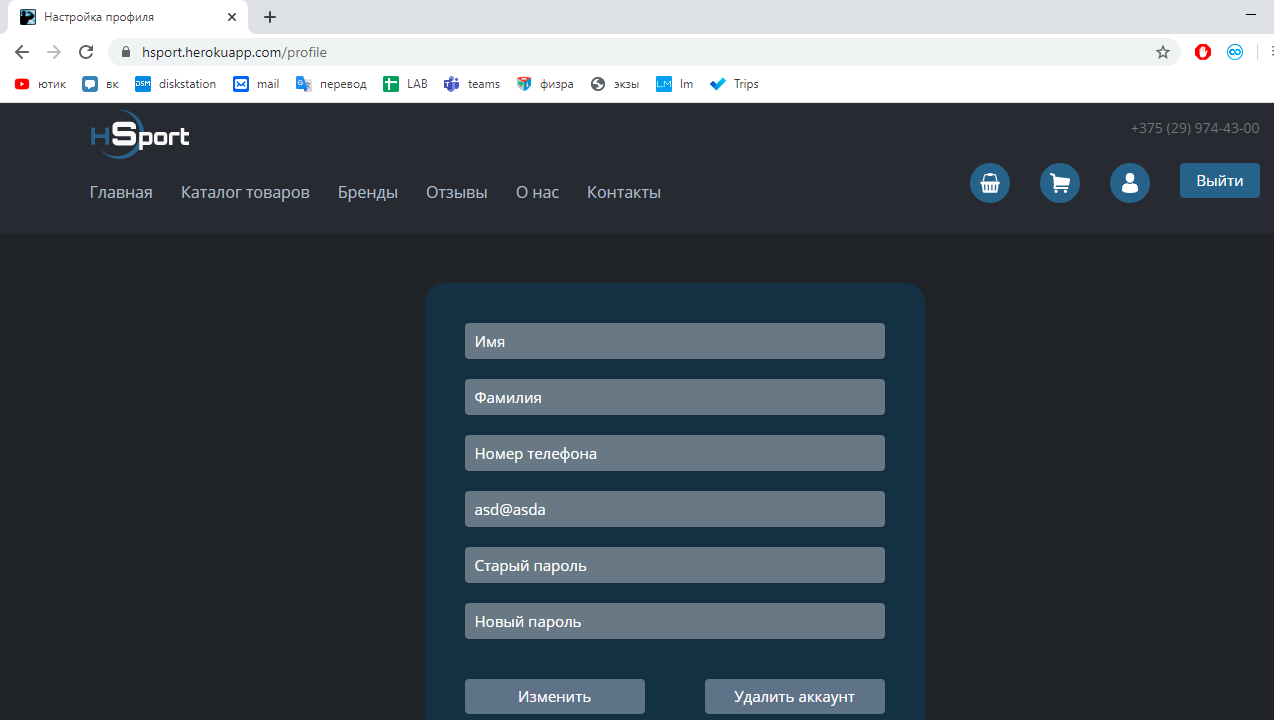


Рисунок 6.2.2 – Страница редактирования профиля

На этой странице пользователь может ввести новые данные о себе, изменить существующие, в том числе почту и пароль. Данная страница валидируется так же, как и страница регистрации. Дополнительно проверяется соответствие старого пароля с тем, что хранится в базе данных.

При нажатии на кнопку «Изменить» введённые данные проверяются и обновляется база данных. При нажатии на кнопку «Удалить аккаунт» статус пользователя устанавливается в значение «0», теперь комментарии этого пользователя будут помечены как [Удалённый пользователь], а доступ к некоторым страницам, например, корзине, будет ограничен. При попытке доступа будет отображаться сообщение, показанное на рисунке 6.2.3.

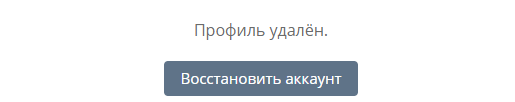


Рисунок 6.2.3 – Сообщение об удалении аккаунта

При нажатии на кнопку «Восстановить аккаунт» статус пользователя вновь установится в единицу, имя пользователя вновь будет отображаться на его комментариях, а доступ к некоторым страницам появится вновь.

Аутентифицированные пользователь может открыть список товаров, нажав на ссылку «Каталог» в шапке страницы. Около каждого товара отображается кнопка «В корзину», которая сохранит данный товар в корзине. Сразу же после нажатия на кнопку, отобразятся новые кнопки «+» и «–», а между ними будет размещено количество товаров данного типа в корзине. При нажатии на плюс и минус количество будет соответственно увеличиваться и уменьшаться.

Также рядом с кнопкой «В корзину» расположена кнопка «Просмотреть рейтинг», которая перенаправляет пользователя при нажатии на страницу товара. Там отображены комментарии пользователей, рейтинг и информация о товаре, а также расположена форма для оставления собственного отзыва. Пользователь может отметить оценку, которую он ставит товару, а также оставить сообщение. При нажатии на кнопку «Отправить» комментарий сохранится в базе данных и отобразится на странице.

## 6.3 Руководство для работника магазина

После ввода верных данных на странице аутентификации, откроется главная страница приложения. В шапке у работника магазина расположены иконки, отображенные на рисунке 6.3.1.



Рисунок 6.3.1 – Кнопки в шапке у работника магазина

Кнопки расположены в следующем порядке: «Все заказы», «Мой профиль», «Добавить товар» и «Выйти». Кнопки «Мой профиль» и «Выйти» выполняют те же функции, как и аналогичные кнопки, описанные в разделе 6.2.

Кнопка «Все заказы» перенаправляет на страницу, где отображены, в отличие от страницы «Мои заказы», заказы всех пользователей. При нажатии на заказ, откроется страница со списком товаров в выбранном заказе. Если заказ в стадии обработки или готов к самовывозу, работник магазина может перевести заказ в следующую стадию, нажав на кнопку внизу страницы.

Кнопка «Добавить товар» перенаправит пользователя на страницу, отображенную на рисунке 6.3.2.

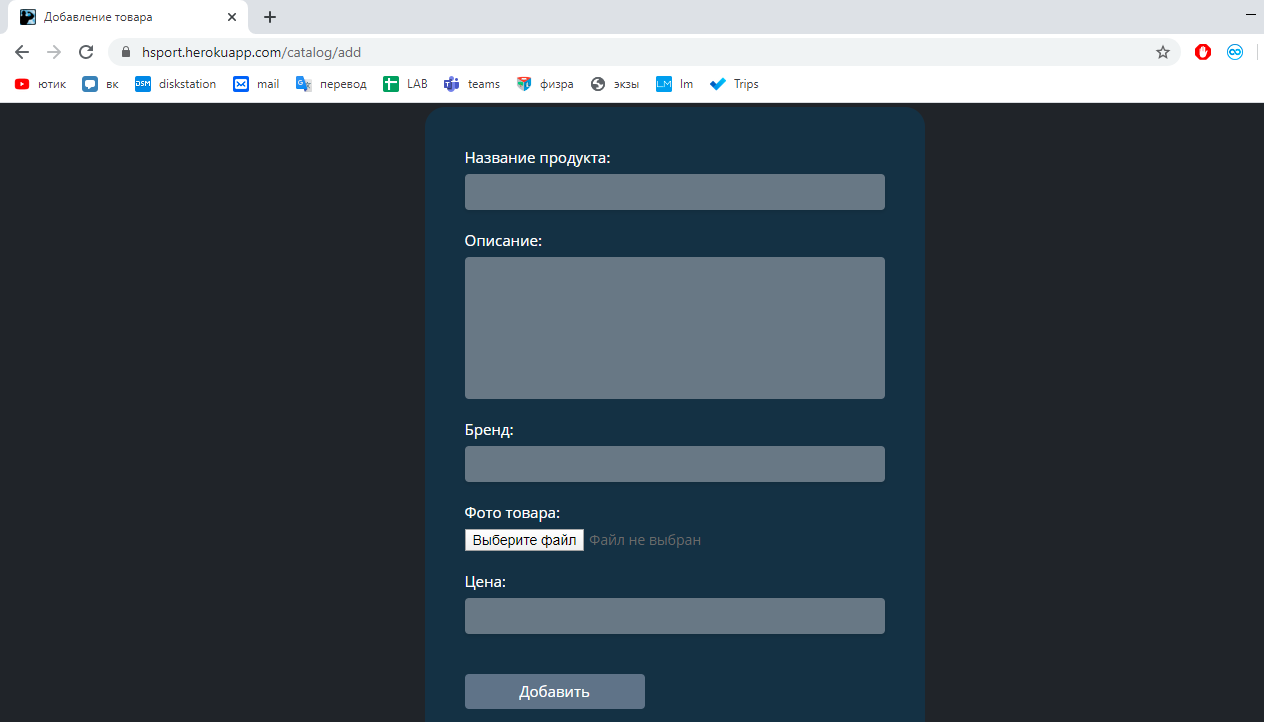


Рисунок 6.3.2 – Форма добавления товара

Чтобы успешно добавить товар, пользователю необходимо заполнить поля названия товара и цены. При этом значение цены может быть от 0.01 до 99999.99, а название товара должно быть уникальным. Остальные поля необязательны для заполнения. Пользователь может нажать на кнопку «Выберите файл» и затем выбрать картинку в формате .png, .jpg или .gif. При нажатии на кнопку «Добавить» откроется каталог. Товар уже будет добавлен в базу данных.

На странице товара у администратора видна кнопка «Редактировать», её расположение представлено на рисунке 6.3.3.

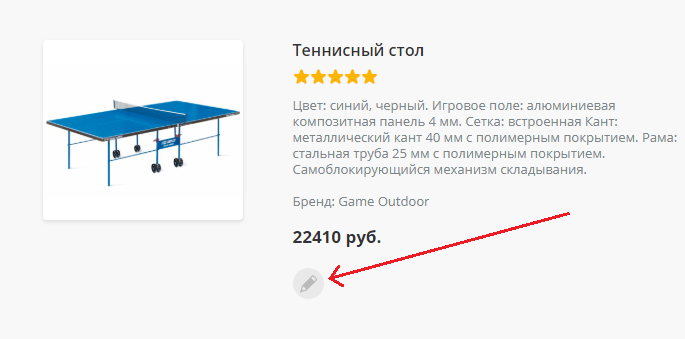


Рисунок 6.3.3 – Кнопка «Редактировать»

При нажатии на кнопку «Редактировать», откроется та же страница, что и при создании товара. Поля, которые были заполнены в базе данных, поместятся в соответствующие поля ввода. Валидационные правила такие же, как и при создании. При нажатии на кнопку «Изменить» изменения сохранятся в базе данных и отобразятся в каталоге.

# ЗАКЛЮЧЕНИЕ

В результате курсовой работы было разработано WEB-приложение «Интернет-магазин спорттоваров», которое выполняет следующие функции:

* имеет интуитивно понятное и простое управление;
* обращается к базе данных, успешно извлекает и вносит новые данные, осуществляет поиск по базе и автоматическую запись некоторых полей;
* имеет различный функционал в зависимости от учётной записи. Работник магазина может добавлять и изменять товары, клиент может просматривать их и добавлять в заказ;
* быстро обрабатывает запросы (не более 5 секунд);
* позволяет пользователю оставлять отзывы на товары, а также изменять их;
* функционирует по протоколам https и wss;
* выглядит аккуратно, не содержит лишнюю информацию.

Таким образом, разработанное программное средство удовлетворяет всем требованиям технического задания, а также реализует несколько дополнительных функций.

Данный программный продукт является завершённым и готов к использованию. Приложение размещено в сети Интернет.

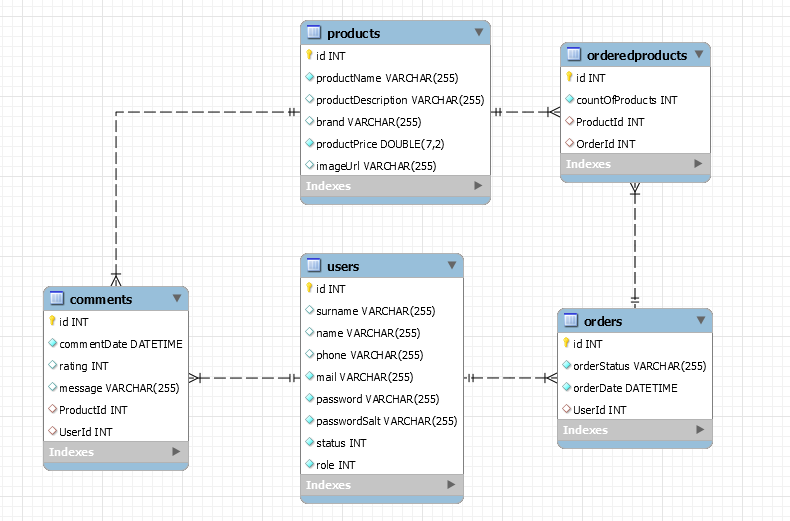
С учётом активной интеграции цифровых технологий во все сферы жизни, разработанное приложение является актуальным.

# СПИСОК ЛИТЕРАТУРЫ

1. 21vek.by [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.21vek.by/fitness/.
2. Зона спорта [Электронный ресурс]. – Электронные данные. – Режим доступа: https://zonasporta.by/.
3. bigsport [Электронный ресурс]. – Электронные данные. – Режим доступа: https://bigsport.by/.
4. Спорт-центр [Электронный ресурс]. – Электронные данные. – Режим доступа: https://sport-center.by/.
5. Vimpex Sport [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.sport-shop.by/.

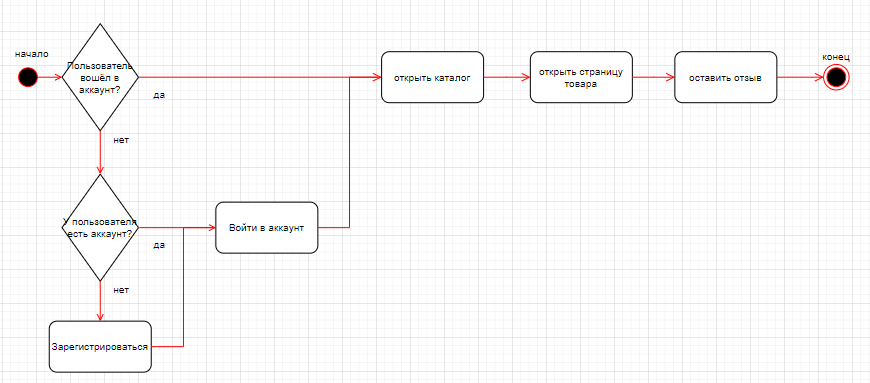
# Приложение А

**Схема базы данных**



# Приложение Б

**Диаграмма деятельности**



# Приложение В

**Реализация WebSocket сервера**

|  |
| --- |
| const ratingUpdateSubscribers = new RatingUpdateSubscribers();  module.exports.wsServer = (httpServer) => {      const ws = new WebSocket.Server({          server: httpServer      });      ws.on('connection', (ws) => {          ws.on('message', (message) => {              const parsedMessage = JSON.parse(message);              switch (parsedMessage.event) {                  case 'add-rating-sub':                      ratingUpdateSubscribers.addClient(parsedMessage.products, ws);                      break;                  case 'remove-rating-sub':                      ratingUpdateSubscribers.removeClient(parsedMessage.products, ws);                      break;              }          })      })  }  module.exports.notifyRatingUpdate = (productName, rating, comment, user) =>{      const clients = ratingUpdateSubscribers.clientList[productName];      if (!clients) {          return;      }      clients.forEach(c => {          c.send(JSON.stringify({              event: 'rating-update',              data: {                  productName: productName,                  rating: rating,                  comment: comment,                  user: user              }          }))      })  } |

# Приложение Г

**Реализация WebSocket клиента**

|  |
| --- |
| <script type="text/javascript">      const HOST = location.origin.replace(/^http/, 'ws')      const socket = new WebSocket(HOST);      let products = [];      socket.onopen = function () {          const productsElements = document.getElementsByClassName('product-name');          for (let element of productsElements) {              products.push(element.innerText)          }          const outgoingMessage = {              event: 'add-rating-sub',              products: products          };          socket.send(JSON.stringify(outgoingMessage));          return false;      };      socket.onclose = function () {          const outgoingMessage = {              event: 'remove-rating-sub',              products: products          };          socket.send(JSON.stringify(outgoingMessage));          return false;      }      socket.onmessage = function (event) {          const message = JSON.parse(event.data);          if (message.event == "rating-update") {              var data = message.data;              showMessage(data);          }      };      function showMessage(message) {          document.getElementById(`${message.productName}-${message.rating}`).checked = true;      }  </script> |