**Содержание**

[Введение 5](#_Toc166888746)

[1. Аналитический обзор литературы 6](#_Toc166888747)

[1.1 Аналог «Wildberries» 6](#_Toc166888748)

[1.2 Аналог «FH.BY» 7](#_Toc166888749)

[1.3 Аналог «Mark Formelle» 8](#_Toc166888750)

[2. Проектирование приложения 9](#_Toc166888751)

[2.1 Проектирование базы данных 9](#_Toc166888752)

[2.2 Описание информационных объектов и ограничений целостности 9](#_Toc166888753)

[2.2.1 Коллекция users 10](#_Toc166888754)

[2.2.2 Коллекция item 10](#_Toc166888755)

[2.2.3 Коллекция orders 10](#_Toc166888756)

[2.2.4 Коллекция comments 10](#_Toc166888757)

[2.2.5 Коллекция order\_detail 10](#_Toc166888758)

[2.2.6 Коллекция point\_of\_issue 11](#_Toc166888759)

[2.3 Проектирование мобильного приложения 11](#_Toc166888760)

[3. Реализация приложения 14](#_Toc166888761)

[3.1 Технические средства разработки 14](#_Toc166888762)

[3.2 Разработка мобильного приложения 14](#_Toc166888763)

[4. Тестирование приложения 16](#_Toc166888764)

[5. Руководство по использованию 19](#_Toc166888765)

[5.1 Руководство пользователя 19](#_Toc166888766)

[5.2 Руководство продавца 22](#_Toc166888767)

[5.3 Руководство модератора 24](#_Toc166888768)

[Заключение 27](#_Toc166888769)

[Список использованных источников 28](#_Toc166888770)

[ПРИЛОЖЕНИЕ А. Диаграмма вариантов использований 29](#_Toc166888771)

# Введение

Неотъемлемой частью жизни большинства людей в современном информационном обществе стали онлайн-магазины одежды. Эти платформы позволяют публиковать и продавать различные предметы гардероба, обувь и аксессуары. Такие приложения дают возможность обмениваться новыми коллекциями, трендами и моделями в удобном формате. Стремительный рост числа интернет-пользователей, развитие социальных сетей и всех сфер жизни человека способствовали популярности онлайн-магазинов одежды.

Интернет-магазины одежды направлены на миллионную аудиторию покупателей. Они позволяют потребителям со всего мира легко и удобно ознакомиться с ассортиментом, сделать заказ и приобрести желаемые товары.

В процессе работы над проектом были поставлены следующие задачи:

* Реализовать функции пользователя, модератора и продавца.
* Регистрация и авторизация пользователей.
* Обеспечение добавления, удаления продавцов.
* Обеспечение добавления, удаления, изменения товара.
* Реализовать сортировку товара.
* Реализовать личный кабинет и его управление.

Основная цель курсового проекта: создание приложения для магазина одежды. Язык разработки проекта – flutter. При выполнении курсового проекта будет использована база данных PostgreSQL.

# Аналитический обзор литературы

В современном мире фотохостинги являются достаточно популярными веб–сервисами. Их основная задача – это хранение фотоизображений, которые загружают пользователи. В дальнейшем эти изображения могут быть использованы другими. При разработке приложение для своего приложения я обратила внимания на самые успешные примеры популярнейших фотохостингов. Аналоги рассмотрены ниже.

## **Аналог «Wildberries»**

В качестве одного из аналогов был выбрано приложение «Wildberries» [1], представленный на рисунке 1.1. Это приложение предназначено для просмотра и приобретения товаров, с полным описание товаров и услуг. Также у всех зарегистрировавших себя как продавец пользователей, есть возможность публикации собственных товаров в приложении. У покупателя есть возможность комментирования и приобретения товара. Данное приложение всё больше и больше увеличивает свою популярность, в особенности на просторах СНГ.

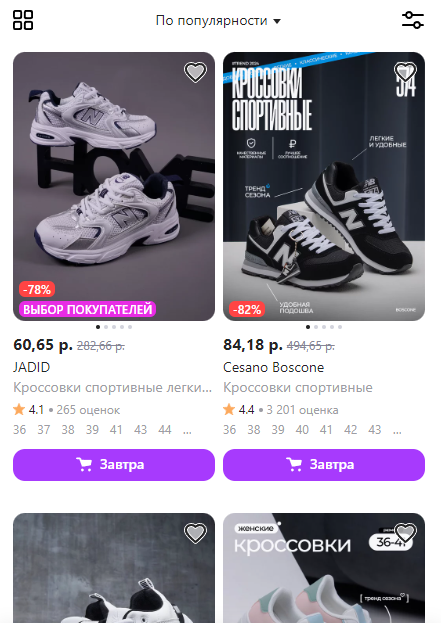


Рисунок 1.1 – Приложение «Pinterest»

Данное приложение получило такую большую популярность в первую очередь из-за простоты использования как для покупателя, так и для предпринимателя, который хочет успешно продавать свой товар или же открыть точку, по выдаче товара.

Из минусов хотелось бы отметить довольно большое количество функционала, что немного перегружает страницы и может сделать немного сложными для начинающих пользователей.

## **Аналог** «FH.BY»

Сервис «FH.BY»[2], представленный на рисунке 1.2, представляет более упрощенный подход к выбору понравившихся и поиску новых товаров. Здесь можно отмечать товары сердечком и можно установить плагин «Сердце», с помощью которого есть возможность добавлять в избранные товары, которые будут находится на отдельной странице.

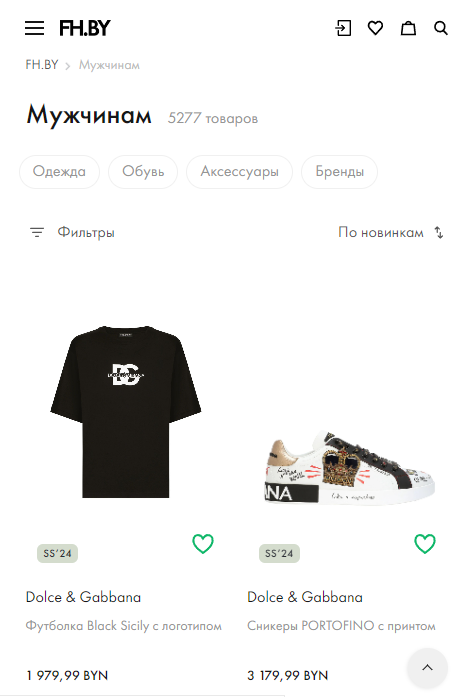


Рисунок 1.2 – Приложение «We Heart It»

В данном приложении мы можем заметить минималистичный и в одно время очень удобный и простой в использовании дизайн, который будет не только радовать глаз пользователям, но и быстрее разобраться с функционалом.

## **Аналог** «Mark Formelle»

Еще одним рассматриваемым аналогом является приложение «Mark Formelle»[3], который можно увидеть на рисунке 1.3. Одной из ключевых особенностей приложения является его огромный выбор товаров. Пользователи могут осуществлять поиск по самым разным фильтрам и сортировкам. Также можно сохранять понравившиеся товары в избранные. Есть полноценная система с личным кабинетом.



Рисунок 1.3 – Приложение «Mark Formelle»

У данного приложения нашлись свои минусы: не самая простая система покупки товаров и услуг. Т.к нужно полностью пройти процедуру регистрации и только после этого можно приобрести товар.

# Проектирование приложения

Проектирование программного обеспечения представляет собой процесс разработки детального плана или концепции программного продукта перед его реализацией. Этот этап включает в себя определение структуры и организации программы, выбор подходящих архитектурных решений, проектирование интерфейсов и определение функциональных возможностей системы.

## Проектирование базы данных

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь.

Для данного курсового проекта была спроектирована база данных «Curs». Логическая модель базы данных представлена на рисунке 2.1.

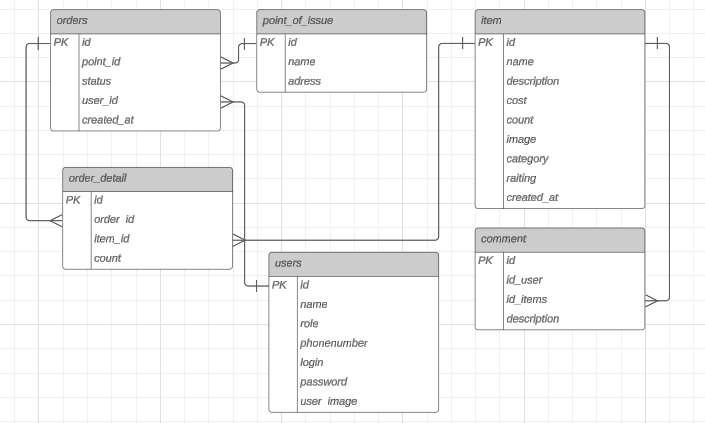


Рисунок 2.1 – Модель БД

Созданная база данных содержит в себе 6 коллекций: Users, Item, Orders, Comments, Order\_detail, Point\_of\_issue.

* 1. Описание информационных объектов и ограничений целостности

Далее будет описана структура каждой коллекции. В коллекции users хранятся данные для идентификации пользователей, зарегистрированных в приложении, item – данные о товарах, orders – для хранения данных о заказе, order-detail – дополнительные данные о заказе, comments – информацию о комментариях, point-of-issue – данные о пункте доставки/выдачи.

### **Коллекция users**

Коллекция users состоит из следующих полей:

* id. Хранит уникальный идентификатор пользователя;
* name. Хранит имя пользователя;
* role. Хранит роль пользователя;
* phonenumber. Хранит информацию о номере телефона пользователя;
* login. Хранит логин пользователя;
* password. Хранит пароль пользователя;
* user\_image. Хранит url на изображение;

### **Коллекция item**

Коллекция item состоит из следующих полей:

* id. Хранит уникальный идентификатор товара;
* name. Хранит название товара;
* description. Хранит описание товара;
* cost. Хранит цену товара;
* count. Хранит информацию о количестве товаров;
* image. Хранит ссылку на изображение товара;
* category. Хранит информацию о категориях;
* raiting. Хранит информацию о рейтинге товара;
* created\_at. Хранит дату создания товара;

### **Коллекция orders**

Коллекция orders состоит из следующих полей:

* id. Хранит уникальный идентификатор заказа;
* point\_id. Хранит идентификатор места доставки;
* status. Хранит информацию о статусе заказа;
* user\_id. Хранит идентификатор пользователя;
* createdAt. Хранит дату создания заказа;

### **Коллекция comments**

Коллекция comments состоит из следующих полей:

* id. Хранит уникальный идентификатор комментария;
* id\_user. Хранит идентификатор клиента добавившего комментарий;
* id\_items. Хранит идентификатор товара, под которым комментарий;
* description. Хранит информацию о содержании комментария;

### **Коллекция order\_detail**

Коллекция order\_detail состоит из следующих полей:

* id. Хранит уникальный идентификатор описания заказа;
* order\_id. Хранит идентификатор заказа из таблицы orders;
* itemId. Хранит идентификатор товара из таблицы item;
* count. Хранит информацию о количестве товаров;

### **Коллекция point\_of\_issue**

Коллекция point\_of\_issue состоит из следующих полей:

* id. Хранит уникальный идентификатор места получения;
* name. Хранит название доставки;
* address. Хранит информацию о адресе доставки;

## Проектирование мобильного приложения

Пользовательский интерфейс в приложении служит связующим звеном между пользователем и программой, формируя первоначальное впечатление о продукте, которое может быть как положительным, так и отрицательным.

В рамках данного проекта ключевое внимание уделяется созданию мобильного приложения, учитывающего особенности проекта, и здесь пользователи ожидают наиболее высокого качества работы. Оптимальным подходом к организации клиентской части является ее разбиение на модули, позволяющие работать над каждым компонентом независимо друг от друга. С учетом всех требований к функциональности, структура клиентской части приложения может быть организована по двум основным направлениям: взаимодействие с пользовательским интерфейсом и внутренние процессы приложения.

Навигация в мобильных приложениях играет ключевую роль, обеспечивая логическую связь между различными экранами и страницами. При проектировании навигации важно учитывать, как пользователи будут перемещаться внутри приложения, чтобы сделать процесс максимально интуитивным и удобным.

При первом запуске приложения пользователь попадает на страницу входа. Оттуда он может попасть на страницу регистрации. После успешной авторизации/регистрации пользователь попадет на экран с четырьмя страницами: все товары, корзина, заказы, профиль.

На странице со всеми товарами пользователь может их просмотреть и нажать кнопку подробнее, чтобы лучше ознакомиться с товаром и купить его.

На странице корзины можно выбрать адрес доставки и ещё раз просмотреть все выбранные товары.

На странице с заказами мы можем увидеть заказанные заказы, узнавать подробную информацию, а также следить за статусом заказа и степенью его готовности.

На странице личного кабинета можно увидеть всю важную информацию о пользователе, а также изменить всю информацию при необходимости.

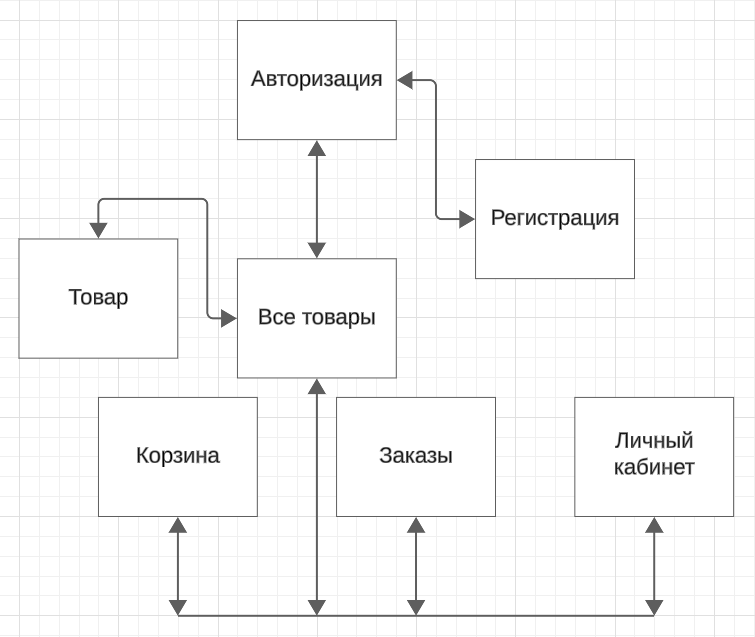
****

Рисунок 2.2 – Схема навигации для пользователя

Для продавца после авторизации доступно 4 страницы. А именно страница для добавления нового товара, для его изменения, а также удаления. На четвёртой странице находится список всех товаров, на котором можно изменять статус заказа.



Рисунок 2.3 – Схема навигации для продавца

Для модератора после авторизации доступно 2 страницы: Модерирование комментариев и список всех пользователей, в котором повышать или понижать пользователей до продавцов или покупателей соответственно.

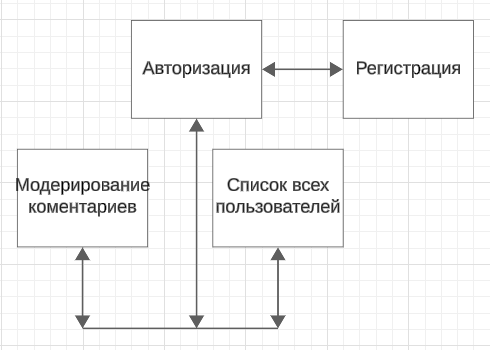


Рисунок 2.4 – Схема навигации администратора

На панели модерирования комментариев модератор может менять роли пользователям.

# Реализация приложения

Реализация программного средства — это процесс преобразования детального плана или концепции программного продукта, разработанного в процессе проектирования, в функциональный продукт.

## Технические средства разработки

Для реализации приложений проекта были использованы следующие технические средства: DataGrip, SQLite, Dart и его фреймворк Flutter, Node.js и модуль Express.

PostgreSQL – это объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире. Имеет открытый исходный код и является альтернативой коммерческим базам данных.

Dart – это язык программирования, предназначенный для разработки клиентов для веб- и мобильных приложений.

Flutter – это фреймворк для разработки мобильных приложений, использующий язык программирования Dart. Flutter использует свой метод построения пользовательского интерфейса, отличающийся от традиционного метода, используемого в Android и iOS разработке. Это позволяет упростить и ускорить процесс разработки и обеспечивает высокую скорость отрисовки интерфейсов.

Node.js – серверная среда выполнения JavaScript, основанная на движке JavaScript V8. Node.js позволяет запускать код JavaScript на стороне сервера и обеспечивает возможность создания масштабируемых сетевых приложений и веб-серверов. Node.js имеет архитектуру "однопоточная событийная цикловая модель", которая позволяет эффективно обрабатывать большое количество соединений с клиентами за счет использования неблокирующих операций ввода-вывода.

Express.js – это фреймворк для Node.js, который позволяет создавать веб-приложения и API более эффективно и удобно. Он предлагает набор методов и инструментов для упрощения обработки запросов и ответов HTTP, маршрутизации запросов, управления сессиями. Express.js является одним из самых популярных Node.js-фреймворков.

Также использовался Postman для тестирования API. Он позволяет пользователям отправлять HTTP-запросы к веб-серверам, проверять их ответы и взаимодействовать с API через интерфейс пользователя.

Для разработки мобильного и веб-приложения использовалась интегрированная среда разработки Android Studio. А для разработки сервера – Visual Studio Code.

## Разработка мобильного приложения

Мобильное приложение содержит в себе несколько папок. Каждая из перечисленных папок в мобильном приложении выполняет определенную функцию:

* widjets представляет классы–помощники, которые используются во всем приложении и содержат некоторую логику.
* utils: содержит методы для взаимодействия с базой данных.
* models содержит классы, которые описывают данные и бизнес–логику приложения. Здесь находятся классы моделей данных, которые отображают структуру данных приложения.
* service: содержит запросы, которые нужны для взаимодействия с базой данных.

Для разработки сервера была выбрана программная платформа Node.js с фреймворком Express. Разработанный сервер использовался для обработки http-запросов. Для обработки запросов используются встроенные функции: get, post. В листинге 3.1 представлена функция для создания пользователя, использующая функцию post.

app.post('/users/create', async (req, res) => {

const { name, password, phonenumber,login,role="customer" } = req.body;

const passwordHash = crypto.createHash('sha256').update(password).digest('hex');

try {

const client = await pool.connect();

const exists = await client.query(

'SELECT 1 FROM users WHERE login = $1 OR phonenumber = $2',

[login, phonenumber]);

if (exists.rows.length > 0) {

res.status(400).send('User with the same login or phone number already exists');return;}

const result = await client.query(

'INSERT INTO users (name, password, phonenumber,login,role) VALUES ($1, $2, $3, $4,$5) RETURNING \*',

[name, passwordHash, phonenumber,login,role]);

client.release();

res.status(201).json(result.rows[0]);

} catch (err) {

console.error(err);

res.status(500).send('Internal server error');

}

Листинг 3.1 – Создание пользователя

Выше представлен пример реализации системы регистрации.

При нажатии на эту кнопку «зарегистрироваться» выполняется функция, которая добавляет нового пользователя, параллельно проверяя все условия. Если пользователь уже существует, то он получит предупреждение о том, что данный пользователь уже существует, а также имеется валидация на всех формах.

# Тестирование приложения

Тестирование мобильных приложений важно и необходимо для обеспечения их качества и производительности. Оно помогает уменьшить риск сбоев приложения, гарантирует отсутствие ошибок и улучшает восприятие пользователей, что в свою очередь способствует увеличению количества загрузок и успешному запуску приложения на рынке.

Для начала протестируем страницу авторизации. Если пользователь не зарегистрирован, ввел неверные данные или не ввел вообще ничего, то он получит предупреждение, показанное на рисунке 4.1.

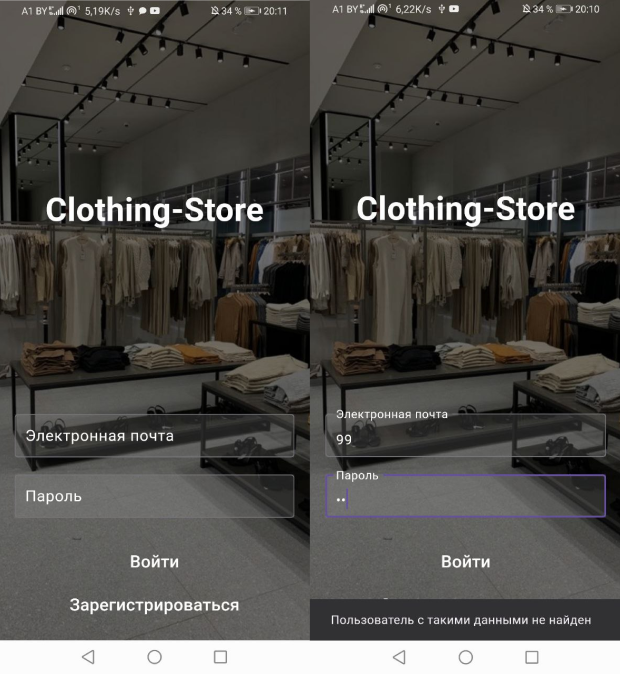


Рисунок 4.1 – Тестирование авторизации

Далее протестируем страницу регистрации. Пользователь может зарегистрироваться с уникальным email и телефоном. Если клиент не хочет выбирать фотографию для профиля, то будет установлено изображение по умолчанию. Кроме того, есть проверка на корректный email и номер телефона. Тестирование продемонстрировано на рисунке 4.2.

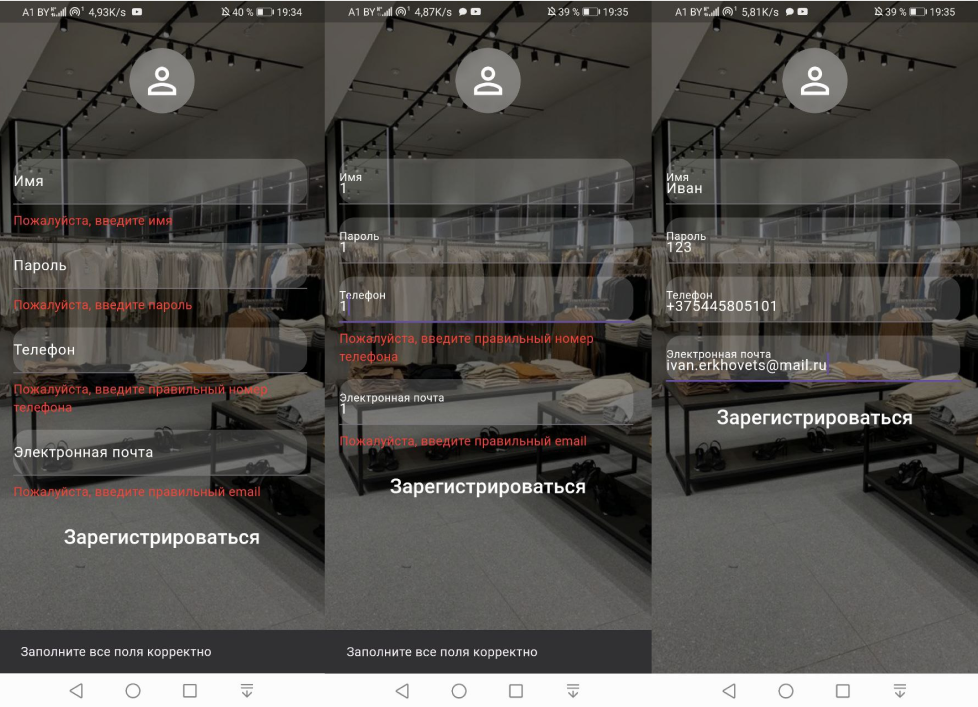


Рисунок 4.2 – Тестирование регистрации

После авторизации или регистрации пользователь попадает на страницу со всеми товарами. Протестируем добавление товара в корзину. При добавлении учитывается товар и количество выбранного товара. После подтверждения заказа и выбора места получения, товар отправляется в заказы. Там можно просмотреть всю информацию о заказе. Последняя страница отвечает за личный кабинет, где можно посмотреть и изменить основную информацию о текущем пользователе. Тестирование продемонстрировано на рисунке 4.3.

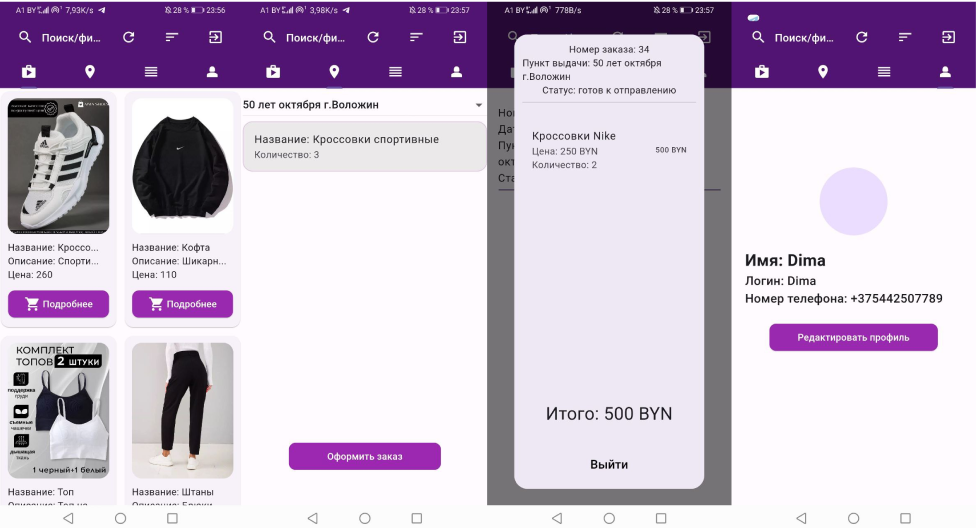


Рисунок 4.3 – Тестирование добавления поста

Кроме рассмотренных возможностей приложения, обработаны возможные ошибки при редактировании профиля. Например, нельзя добавить данные, которые не подходят по нашей валидации в регистрации пользователя.

Когда модератор меняет роли у пользователей, он не может поменять собственную роль. Продемонстрировано на рисунке 4.4.

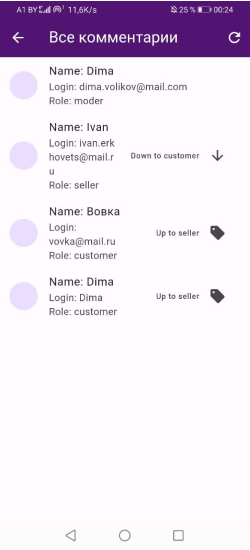


Рисунок 4.4 – Обработка ошибки в панели администратора

Тестирование приложения составляет неотъемлемый этап процесса разработки, обеспечивающий высокое качество продукта и удовлетворенность пользователей. В рамках данного проекта, благодаря тщательной проверке и обработке ошибок, было создано надежное и удобное в использовании мобильное приложение для магазина одежды, которое успешно выполняет свои задачи и отвечает требованиям целевой аудитории.

# Руководство по использованию

В данном разделе будет описано руководство по использованию для пользователя, продавца и модератора.

## Руководство пользователя

Для того чтобы пользователь вошёл, при авторизации нужно нажать на кнопку «Войти». После того пользователь попадает на страницу с товарами, показанную на рисунке 5.1.

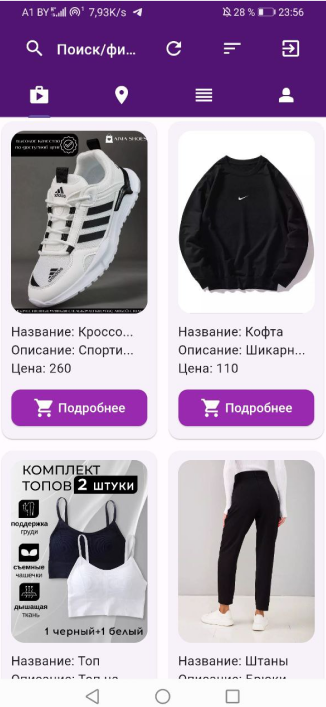


Рисунок 5.1 – Страница «Товары» для пользователя

При клике на любой товар, откроется детальная страница поста, где пользователь может увидеть детальную информацию о товаре и комментарии. При желании пользователь может оставить комментарий о заказе. Детальная страница товара показана на рисунке 5.2.

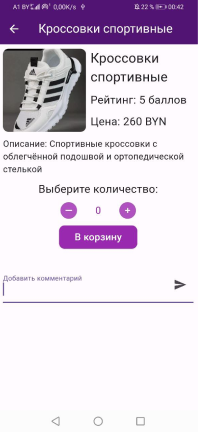


Рисунок 5.2 – Детальная страница поста для гостя

После добавления товара в корзину, мы можем ещё раз просмотреть количество товара, а также выбрать нужный нам адрес доставки/самовывоза. Страница с корзиной показана на рисунке 5.3.

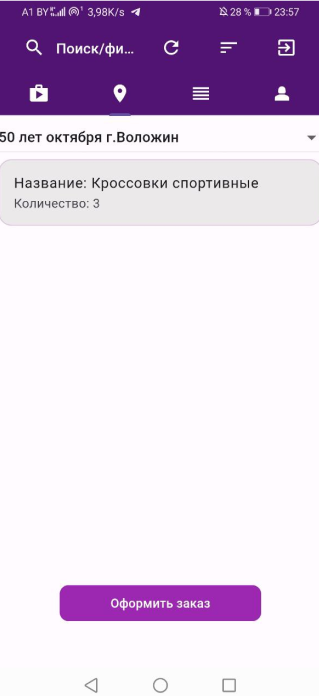


Рисунок 5.3 – Страница с корзиной

Когда заказ оформлен, он попадает в список заказ, мы можем ещё раз посмотреть всю информацию о заказе, а также следить за изменяемым статусом заказа. Страница с заказами показана на рисунке 5.4.

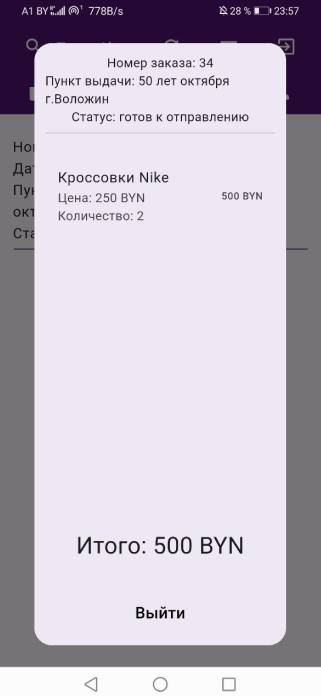


Рисунок 5.4 – Страница с заказами

Пользовать ещё может просмотреть информацию о своём профиле, а также внести изменения. Страница личным кабинетом показана на рисунке 5.5.

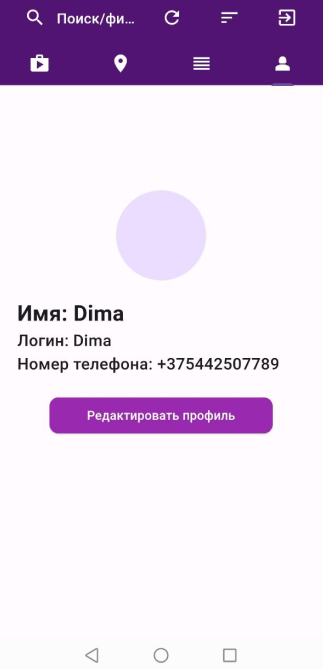


Рисунок 5.5 – Страница с заказами

При изменении информации в личном кабинете, используется точно такая же валидация, как и при регистрации, что не даст неверно заполнить информацию.

## Руководство продавца

Чтобы продолжить работу в приложении как продавец, пользователь должен иметь специальную роль и войти в приложение под соответствующей учетной записью. При входе в приложение как продавец будет открыта страница, показанная на рисунке 5.6. В отличие пользователя, у продавца появляется возможность удалять, изменять, добавлять товары и изменять их статус готовности.

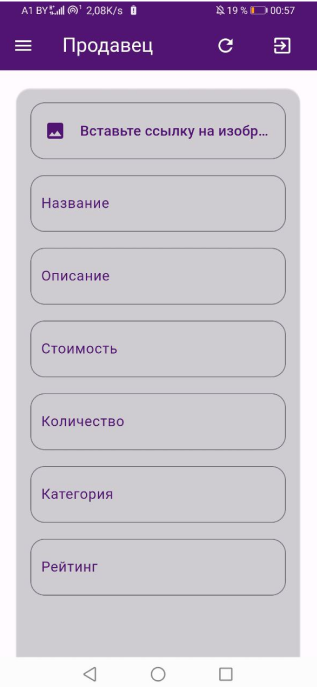


Рисунок 5.6 – Страница «добавления товара» для продавца

На рисунке 5.6 мы видим форму, которая используется для создания товара. Все поля являются обязательными к заполнению. Продавец может вносить изменения в уже созданные товары. Показано на рисунке 5.7.

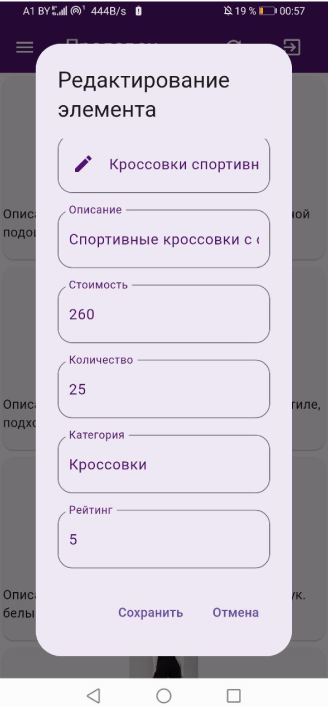


Рисунок 5.7 – Изменения информации о товаре

Продавец может удалять все существующие товары. Показано на рисунке 5.8.



Рисунок 5.8 – Удаление товара по идентификатору

У продавца имеется возможность просматривать все заказы и вносить изменения в степень готовности товара. Показано на рисунке 5.9.



Рисунок 5.9 – Изменение статуса товара

Продавец может быть назначен или же снят только модератором и больше никем.

## Руководство модератора

При заходе модератора, его встречает страница со всеми комментариями. На этой странице можно увидеть все комментарии, которые были написаны к товарам. В комментарии отображает идентификатор того, кто написал а также содержания комментария. Модератор может удалять некорректные комментарии. Страница с комментариями показана на рисунке 5.10.



Рисунок 5.10 – Страница с комментариями

У модератора есть страница со всеми пользователями. Страница с пользователями показана на рисунке 5.11.

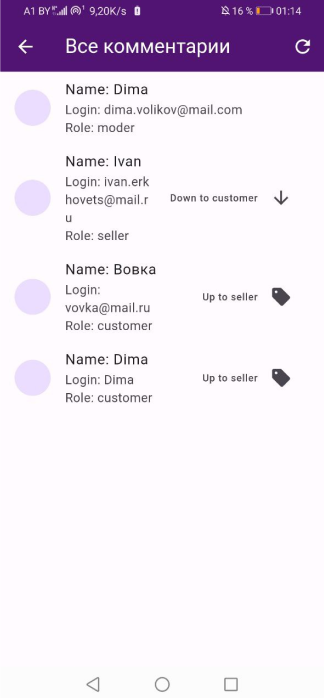


Рисунок 5.11 – Страница с пользователями

На этой странице модератор может повысить или же понизить других пользователей до продавца или обычного пользователя соответственно. Имеется защита от того, что бы модератор не мог понизить самого себя.

# Заключение

В результате выполнения данной курсовой работы было разработано мобильное приложение «Clothing-Store», которое предоставляет собой магазин одежды.

Разработанное программное средство предоставляет пользователю следующие функциональные возможности:

* регистрация пользователей;
* авторизация пользователей;
* добавление в корзину;
* оформление заказа
* добавление и удаление комментариев;
* поиск и фильтрация;
* редактирование профиля пользователя.

Продавцу предоставляет следующие функциональные возможности:

* авторизация;
* просмотр товаров;
* создание, изменение, удаление товара;

Модератор имеет возможность:

* авторизация;
* просмотр и удаление всех комментариев;
* добавление и удаление продавцов.

Разработанное программное средство реагирует на ошибочный ввод данных выводя при этом соответствующее сообщение об ошибке.

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает верно, а требования технического задания выполнены в полном объеме.

# Список использованных источников

1. Wildberries [Электронный ресурс]. – Режим доступа: https://www.wildberries.by/– Дата доступа: 17.04.2024
2. FH.BY [Электронный ресурс]. – Режим доступа: https://fh.by/– Дата доступа: 17.04.2024
3. MF [Электронный ресурс]. – Режим доступа: https://markformelle.by/– Дата доступа: 18.04.2024
4. Flutter [Электронный ресурс]. – Режим доступа: <https://docs.flutter.dev/tools/devtools/overview> – Дата доступа: 18.04.2024
5. Dart [Электронный ресурс]. – Электронные данные. – Режим доступа:  [https://metanit.com/dart/.](https://stackoverflow.com./) – Дата доступа: 20.04.2024;
6. Чернышев, С.А. Основы Dart. 2–е издание / С.А. Чернышев. – Москва: Издательство "Наука", 2023. – 350 с.
7. ItProger [Электронный ресурс]. – Режим доступа: https://itproger.com/course/flutter-dart/– Дата доступа: 25.05.2024
8. Hubr [Электронный ресурс]. – Режим доступа: https://habr.com/ru/companies/surfstudio/articles/515226/ – Дата доступа: 26.04.2024

# ПРИЛОЖЕНИЕ А. Диаграмма вариантов использований

