Министерство образования Республики Беларусь

Учреждения образования «БЕЛОРУССКИЙ

ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 1-98 01 03 Программное обеспечение информационной безопасности мобильных систем

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**ДИПЛОМНОГО ПРОЕКТА НА ТЕМУ:**

Мобильное приложение для магазина спортивной обуви

Обучающийся 4 курс, 7 группа И.А. Ероховец

курс, группа дата подпись И. О. Ф.

Руководитель

дипломного проекта ст. преп. Н.И. Уласевич

должность, ученая степень, ученое звание дата подпись И. О. Ф.

И.о. заведующего

кафедрой к.т.н. Е.А. Блинова

ученая степень, ученое звание дата подпись И. О. Ф.

Консультант ст. преп. А.С. Соболевский

должность, ученая степень, ученое звание дата подпись И. О. Ф.

Нормоконтролер асс. А.Н. Николайчук

должность, ученая степень, ученое звание дата подпись И. О. Ф.

Дипломный проект защищен с оценкой

Председатель ГЭК к.т.н., доцент В.К. Дюбков

должность, ученая степень, ученое звание дата подпись И. О. Ф.

**Реферат**

ФИО

Подпись

Дата

Лист

1

ДП 00.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

Реферат

Лит.

Листов

1

БГТУ 1-98 01 03, 2025

**Abstract**

У

ФИО

Подпись

Дата

Лист

1

ДП 00.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

Abstract

Лит.

Листов

1

БГТУ 1-98 01 03, 2025

Оглавление

ФИО

Подпись

Дата

Лист

1

ДП 00.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

Содержание

Лит.

Листов

1

БГТУ 1-98 01 03, 2025

[Введение 7](#_Toc199272996)

[1 Обзор аналогов и постановка задачи 8](#_Toc199272997)

[1.1 Обзор аналогов проектируемого мобильного приложения 8](#_Toc199272998)

[1.1.1 Приложение «Lamoda» 8](#_Toc199272999)

[1.1.2 Мобильное приложение «Zenden» 11](#_Toc199273000)

[1.1.3 Мобильное приложение «Kari» 13](#_Toc199273001)

[1.2 Постановка задачи 16](#_Toc199273002)

[1.3 Выводы по разделу 16](#_Toc199273003)

[2 Проектирование мобильного приложения 18](#_Toc199273004)

[2.1 Выбор технологий и средств 18](#_Toc199273005)

[2.1.1 Среда разработки Android Studio 18](#_Toc199273006)

[2.1.2 Язык программирования Dart 19](#_Toc199273007)

[2.1.3 Облачная база данных Firebase 20](#_Toc199273008)

[2.1.4 Фреймворк Flutter 20](#_Toc199273009)

[2.2 Проектирование базы данных 21](#_Toc199273010)

[2.3 Проектирование мобильного приложения 26](#_Toc199273011)

[2.4 Диаграммы и блок-схемы приложения 29](#_Toc199273012)

[2.4.1 Диаграмма вариантов использования 29](#_Toc199273013)

[2.5.1 Диаграмма развертывания 31](#_Toc199273014)

[2.5.2 Блок-схема оформления заказа из заведения 32](#_Toc199273015)

[2.5.3 Блок-схема смены пароля 33](#_Toc199273016)

[2.7 Выводы по разделу 34](#_Toc199273017)

[3 Реализция приложения 35](#_Toc199273018)

[3.1 Взаимодействие с базой данных 35](#_Toc199273019)

[3.2 Разработка клиентской части 39](#_Toc199273020)

[3.4 Выводы по разделу 42](#_Toc199273021)

[6 Технико-экономическое обоснование проекта 43](#_Toc199273022)

[6.1 Общая характеристика разрабатываемого программного средства 43](#_Toc199273023)

[6.2 Исходные данные для проведения расчётов 43](#_Toc199273024)

[6.3 Методика обоснования цены 44](#_Toc199273025)

[6.3.1 Определение объёма программного средства 45](#_Toc199273026)

[6.3.2 Основная заработная плата 45](#_Toc199273027)

[6.3.3 Дополнительная заработная плата 46](#_Toc199273028)

[6.3.4 Расчёт отчислений на социальные цели 47](#_Toc199273029)

[6.3.5 Расходы на материалы 47](#_Toc199273030)

[6.3.6 Расходы на специальное оборудование и платные услуги 47](#_Toc199273031)

[6.3.7 Расчёт прочих прямых затрат 47](#_Toc199273032)

[6.3.8 Расчёт общепроизводственных и общехозяйственных расходов 48](#_Toc199273033)

[6.3.9 Расчёт суммы расходов на разработку программного средства 48](#_Toc199273034)

[6.3.10 Расходы на сопровождение и адаптацию 49](#_Toc199273035)

[6.3.11 Расчет общей суммы расходов 49](#_Toc199273036)

[6.3.12 Определение цены, оценка эффективности 49](#_Toc199273037)

[6.4 Вывод по разделу 52](#_Toc199273038)

[7 Анализ информационной безопасности приложения 54](#_Toc199273039)

[7.1 Защита пользовательских данных 54](#_Toc199273040)

[7.2 Ограничение прав доступа пользователей 55](#_Toc199273041)

[7.3 Выводы по разделу 55](#_Toc199273042)

[Список использованных источников 56](#_Toc199273043)

# Введение

ФИО

Подпись

Дата

Лист

1

ДП 00.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

Введение

Лит.

Листов

1

БГТУ 1-98 01 03, 2025

В современном мире, где время является одним из самых ценных ресурсов, люди ежедневно сталкиваются с необходимостью решать повседневные задачи, требующие значительных временных затрат. Одной из таких задач является выбор и покупка одежды, в том числе и обуви. С каждым годом все больше людей стремятся ускорить этот процесс, минимизировать возникающие проблемы и максимально упростить удовлетворение своих потребностей.

В ответ на эти вызовы возникла необходимость в создании мобильных приложений, которые не только упрощают процесс выбора и покупки, но и предоставляют доступ к товарам из разных уголков мира. Такие приложения способны значительно экономить время пользователей.

Целью данного дипломного проектирования является разработка мобильного приложения для онлайн-покупки спортивной обуви. Приложение будет не только выполнять базовые функции интернет-магазина, но и предлагать дополнительные возможности: помощь в выборе одежды на основе предпочтений пользователя, доступ к отзывам, а также рекомендации, учитывающие стиль и вкусы клиента.

Для достижения цели дипломного проектирования, необходимо решить ряд ключевых и важных задач:

* изучить аналоги и выявить их преимущества и недостатки;
* подобрать инструменты и платформы для реализации приложения;
* создать базу, содержащую всю необходимую информацию о предоставляемых товарах, пользователях и заказах;
* разработать архитектуру, которая будет обеспечивать удобство использования и стабильность работы;
* спроектировать интуитивно понятный и визуально привлекательный интерфейс с возможностью фильтрации товаров;
* разделить пользователей системы на четыре роли: администратор, продавец, поставщик и пользователь.

Приложение будет интегрировано с базой данных, что позволит хранить и обрабатывать все необходимые данные о товарах и пользователях. Особое внимание будет уделено созданию удобного интерфейса, который обеспечит комфортное взаимодействие с приложением.

Реализация данного проекта призвана упростить процесс выбора и покупки обуви, сделав его быстрым и удобным для пользователей. Актуальность проекта обусловлена ростом популярности онлайн-покупок, а также постоянным спросом на одежду и делает разработку приложения особенно важной.

У

# 1 Обзор аналогов и постановка задачи

ФИО

Подпись

Дата

Лист

1

ДП 01.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

Обзор аналогов и постановка задачи

Лит.

Листов

10

БГТУ 1-98 01 03, 2025

Для составления технических требований к проекту был проведен обзор популярных мобильных приложений с похожим функционалом. Были изучены популярные решения как отечественных, так и зарубежных разработчиков, что позволило составить обширную картину текущих тенденций.

## 1.1 Обзор аналогов проектируемого мобильного приложения

В текущее время представлено достаточно количество мобильных приложений для приобретения обуви онлайн. Основным назначением такого рода приложения – предоставление возможности приобретения какого-либо товара любого предоставленного размера и других критериев. Однако не все приложение следят за актуальностью некоторых товаров, а также их наличия у поставщиков. Важными аспектами при выполнении оценки таких приложений является их функциональность и удобство использования. Также важно, чтобы приложение работало стабильно при высокой нагрузке. Это ещё больше обеспечит комфортное использование для большого количества пользователей одновременно и может кратно увеличить доходы.

Наиболее популярными на данных момент мобильные приложения с похожим функционалом являются:

* Lamoda [1];
* Zenden [2];
* Kari [3].

### 1.1.1 Приложение «Lamoda»

«Lamoda» – это современный сервис для онлайн покупок, которые предлагает большой выбор одежды и обуви.

Пользователям предоставляется выбор одежды различных брендов и категорий, таких как женская, мужская и детская, а также спортивная обувь. «Lamoda» предлагает покупателям разные способы доставки, можно как забрать самостоятельно, так и заказать курьера на дом. Присутствуют и различные способы оплаты, включая как оплату наличными, так и онлайн-платежи.

«Lamoda» активно развивает своё приложение, регулярно следит за новыми трендами и пополняется новыми брендами. За счёт удобного интерфейса и функционала, приложение стало популярным среди пользователей.

На главной странице приложения пользователю предлагается выбор своей страны и дальнейший вход в систему.

У

После входа в систему пользователь может ознакомиться с популярными предложениями, которые разбиты на определённые категории, а именно «Одежда», «Обувь», «Аксессуары», «Головные уборы» и другие, что делает поиск нужных товаров более простым и быстрым.

Приложение стало популярным, благодаря своему удобству использования. На рисунке 1.1 представлена страница со списком товаров.

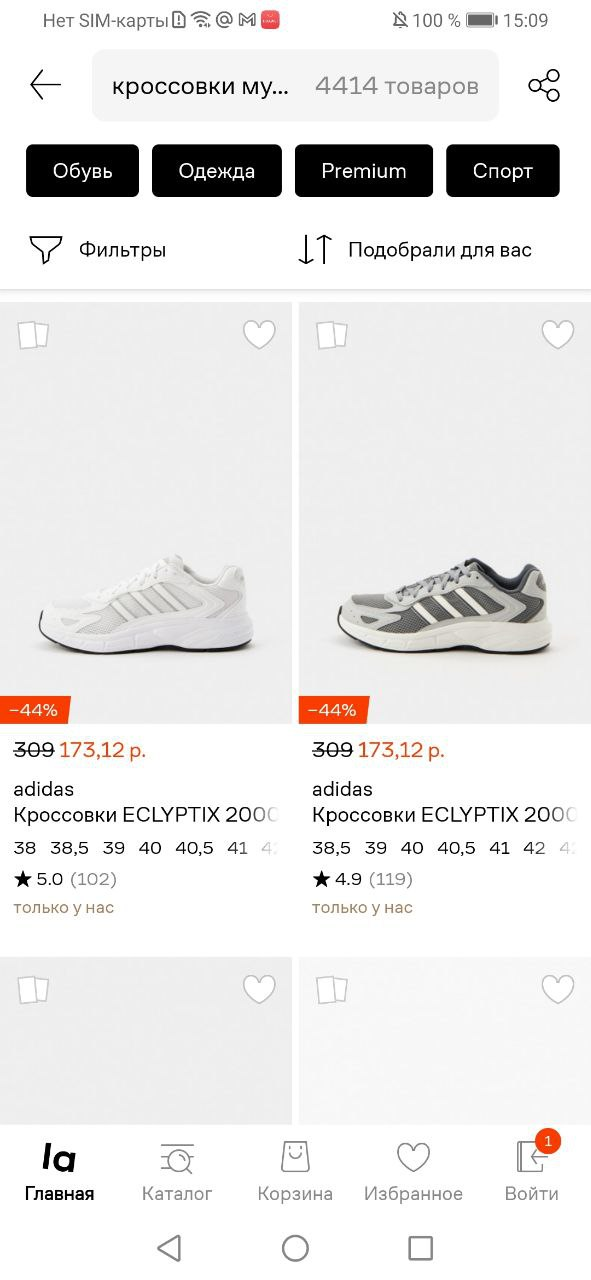


Рисунок 1.1 – Каталог товаров в приложении «Lamoda»

Каждый товар в приложении имеет свою уникальную страницу, на которой представлена вся необходимая информация для потенциального покупателя. Страница товара включает подробное описание товара, доступные размеры, актуальные цены, а также отзывы от других пользователей. Отзывы играют важную роль, так как они позволяют покупателю ознакомиться с мнением людей, которые уже приобрели данный товар. Это помогает сформировать более полное представление о качестве продукции и сделать осознанный и взвешенный выбор при покупке.

Процесс оформления заказа также хорошо продуман и максимально удобен для пользователя. После добавления товаров в корзину покупателю предлагается выбрать наиболее подходящий способ доставки, а также вариант оплаты. Для удобства пользователей предусмотрено несколько способов оплаты: это может быть оплата онлайн, с использованием банковской карты или электронного кошелька, либо оплата наличными при получении товара.

Такой подход позволяет каждому покупателю самостоятельно выбрать наиболее комфортный и удобный для него вариант.

На рисунке 1.2 представлен интерфейс страницы с товаром.

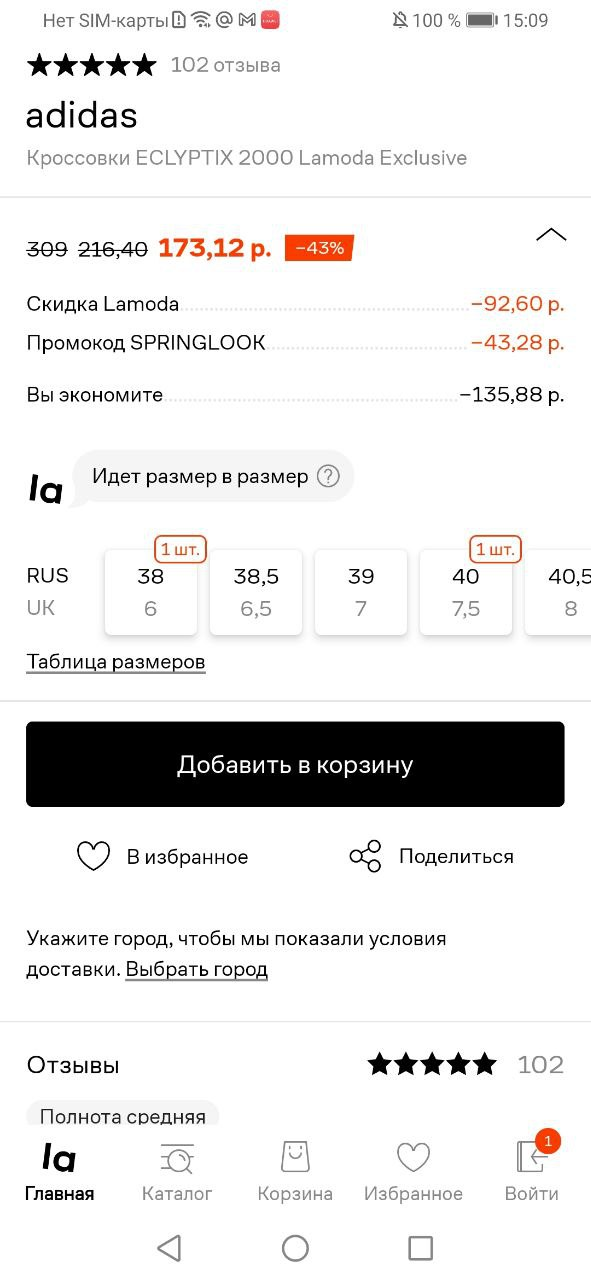


Рисунок 1.2 – Страница товара в приложении «Lamoda»

Также у приложения реализована удобная и функциональная корзина, с которой сможет разобраться любой пользователь. Интерфейс интуитивно понятен, можно быстро добавлять и изменять товары и их количество. Пример корзины представлен на рисунке 1.3.

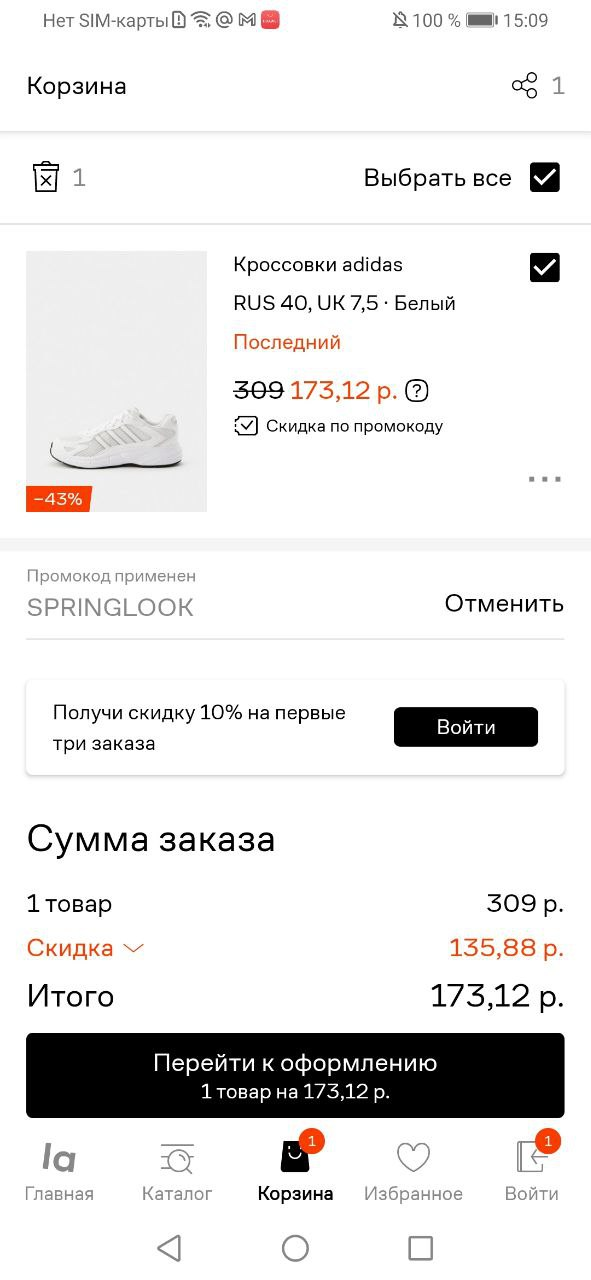


Рисунок 1.3 – Страница корзины в приложении «Lamoda»

Ключевые особенности приложения «Lamoda»:

* наличие широкого ассортимента обуви;
* удобный поиск и навигация по категориям;
* наличие рейтинга и отзывов;
* возможность оформления заказа и оплаты онлайн;
* интеграция с картами для отслеживания доставки;
* продуманная цветовая гамма.

Среди основных недостатков можно выделить обилие представленного товара, что порой заметно затрудняет удобную навигацию и поиск нужных позиций.

В целом, мобильное приложение «Lamoda» является удобным и функциональным, особенно для тех, кто предпочитает шопинг, не выходя из дома.

### 1.1.2 Мобильное приложение «Zenden»

Мобильное приложение «Zenden» предлагает пользователям удобные возможности для покупки обуви, в том числе стильных и спортивных моделей. Основная задача приложения – сделать выбор подходящей пары максимально приятный и быстрым, а также привлечения новых клиентов.

Приложение отличается рядом полезных возможностей, которые делают шопинг комфортным. Среди ключевых функций можно выделить:

* широкий ассортимент обуви для любого случая, включая спортивную;
* возможность добавлять товары в избранное;
* получение пользовательских рекомендаций, на основе предпочтений;
* доступ к эксклюзивным скидкам и акциям.

Однако, несмотря на множество преимуществ, приложение обладает и некоторыми недостатками, которые могут повлиять на впечатления от использования. Например, для полного доступа ко всем функциям, таким как оформление заказов и сохранение товаров в избранное, требуется обязательная регистрация. Это может стать препятствием для пользователей, которые предпочитают делать покупки без создания личного аккаунта.

Еще одним недостатком является отсутствие возможности просматривать историю покупок. Такая функция значительно упростила бы повторное оформление заказов на уже приобретённую обувь.

Помимо вышеперечисленных особенностей, стоит отметить, что приложение «Zenden» активно внедряет современные решения, которые позволяют сделать процесс выбора и покупки обуви максимально индивидуальным. Например, пользователям предоставляется возможность фильтровать ассортимент по различным параметрам – от размера и цвета до бренда и материала. Такой подход помогает не только сэкономить время, но и избежать лишних трат, ведь система быстро подбирает наиболее релевантные предложения. Кроме того, интерфейс приложения выполнен интуитивно понятно: навигация по разделам не вызывает сложностей даже у тех, кто не имеет большого опыта использования мобильных сервисов.

На рисунке 1.4 представлена страница со списком товаров.

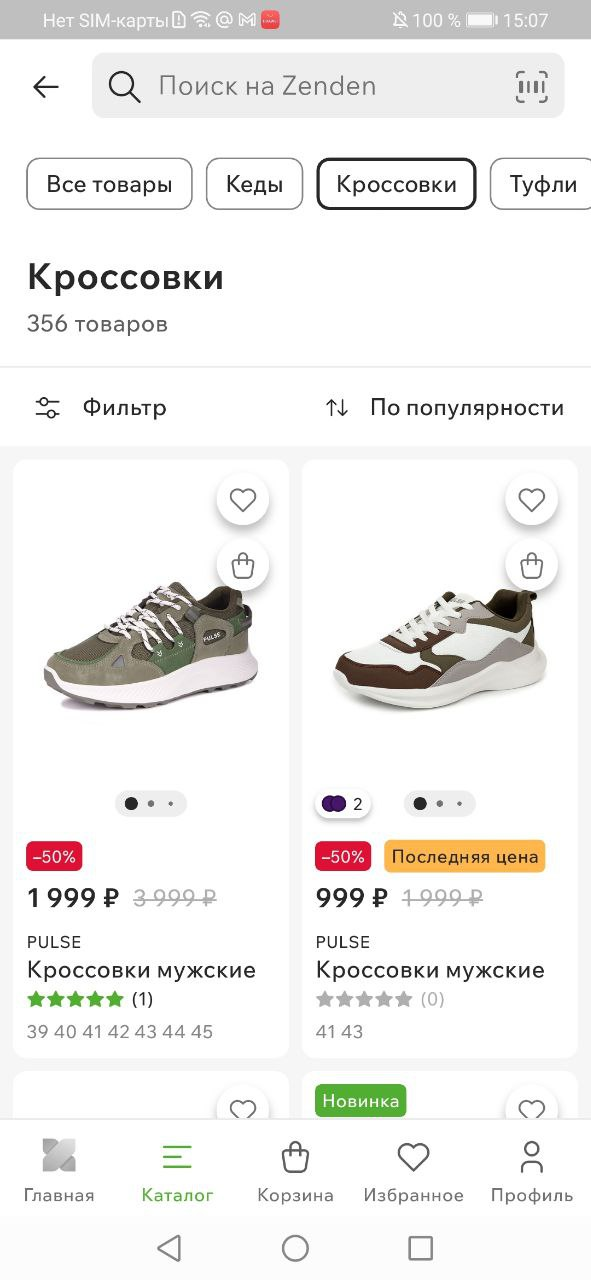


Рисунок 1.4 – Страница с каталогом в приложении «Zenden»

Кроме того, ассортимент приложения ограничивается исключительно продукцией бренда «Zenden», что может не подойти пользователям, ищущим максимально широкий выбор товаров разных производителей. На рисунке 1.5 представлена страница товара.

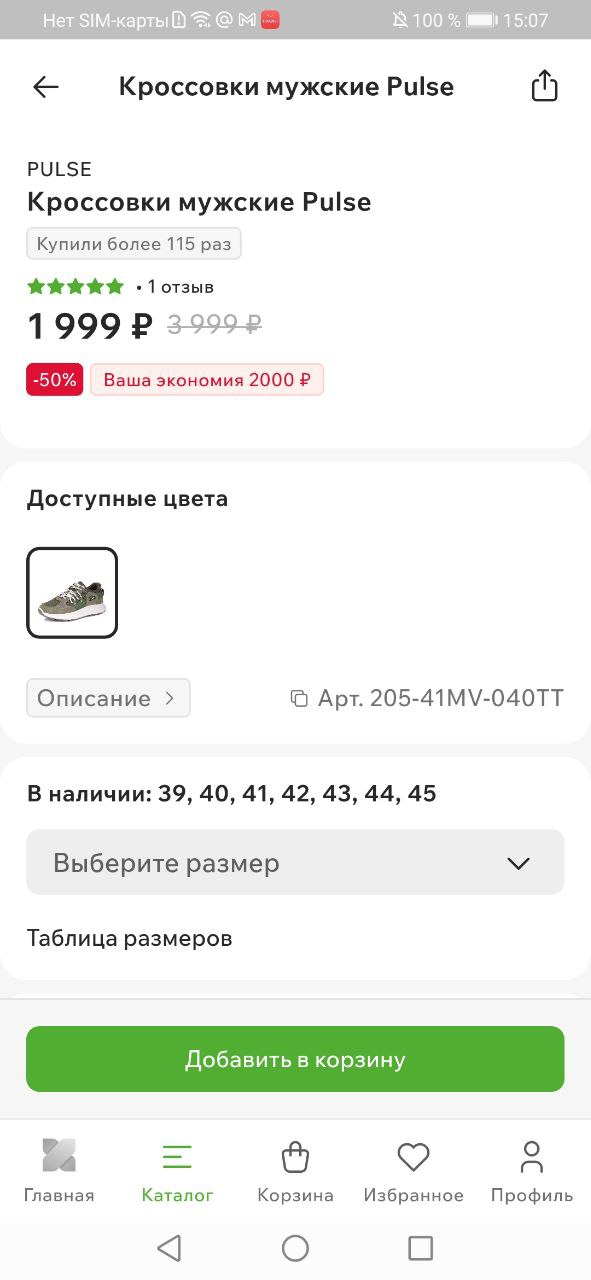


Рисунок 1.5 – Страница товара в приложении «Zenden»

Карточка товара в мобильном приложении «Zenden» разработана так, чтобы пользователь мог получить всю необходимую информацию о выбранной обуви быстро и удобно. На отдельной странице для каждой модели представлено несколько качественных фотографий, позволяющих рассмотреть обувь со всех сторон и детально изучить материалы, отделку и особенности дизайна.

Для удобства покупателей в карточке отображается актуальная информация о наличии выбранного размера, стоимости, а также о действующих скидках или акциях. При необходимости можно сразу добавить товар в корзину и продолжить оформление заказа или продолжить поиск.

На рисунке 1.6 представлена страница корзины.

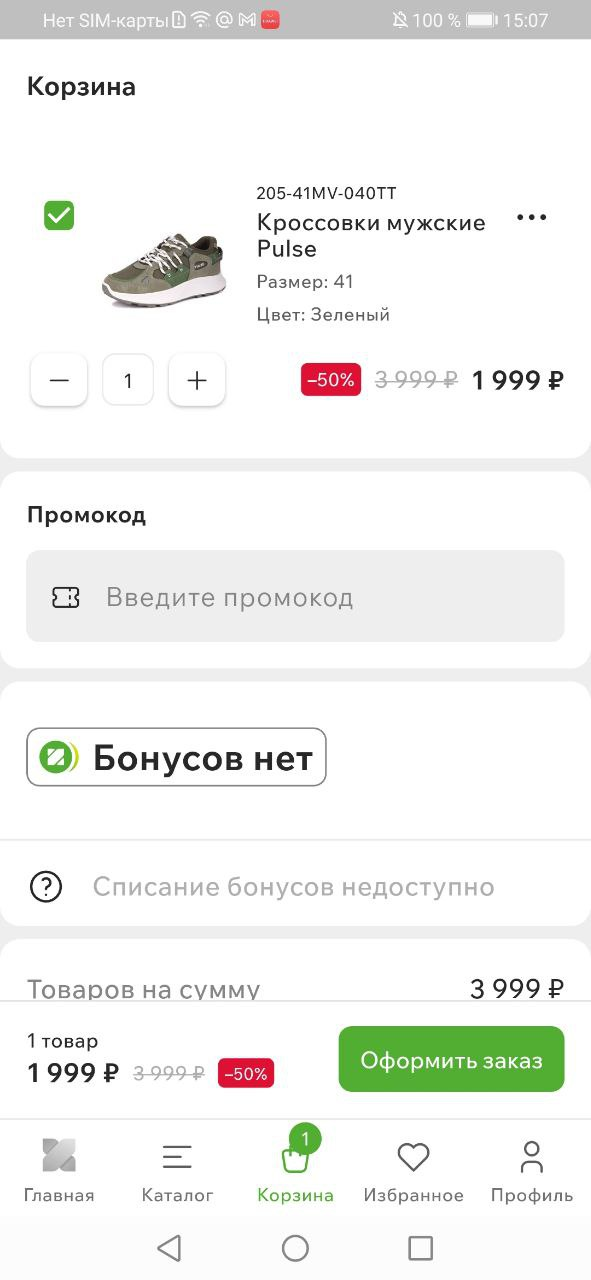


Рисунок 1.6 – Страница корзины в приложении «Zenden»

В целом мобильное приложение «Zenden» стало отличным примером того, как современные технологии делают процесс покупки обуви проще и комфортнее. Сочетает в себе богатый функционал и привлекательный дизайн. Приложение быстро реагирует на действия пользователя, что положительно сказывается на общем впечатлении от использования. Благодаря продуманной структуре меню, основной функционал всегда находится под рукой. Тем не менее, такие недостатки, как обязательная регистрация и ограничение ассортимента только одной маркой, могут быть устранены в будущем, что сделает приложение ещё более удобным и привлекательным для покупателей.

### 1.1.3 Мобильное приложение «Kari»

Мобильное приложение для заказа обуви и аксессуаров бренда «Kari» сделано так, чтобы онлайн-шопинг был максимально удобный и современным. Пользователям доступны все нужные функции: от быстрого поиска това­ров до оформления заказа за пару минут. Задача приложения – дать воз­можность легко познакомиться с полным ассортиментом обуви, особенно спортив­ных моделей, и упростить каждый этап покупки.

Интерфейс приложения соответствует актуальным стандартам мобильной коммерции, что делает навигацию по каталогу простой и интуитивно понятной. Пользователь легко может перейти к интересующим его категориям: женская, мужская, детская или спортивная обувь, а также найти аксессуары для завершения образа. Поиск нужной пары осуществляется по названию, параметрам или с помощью фильтров, что особенно удобно при большом ассортименте.

На рисунке 1.7 представлена страница с каталогом.

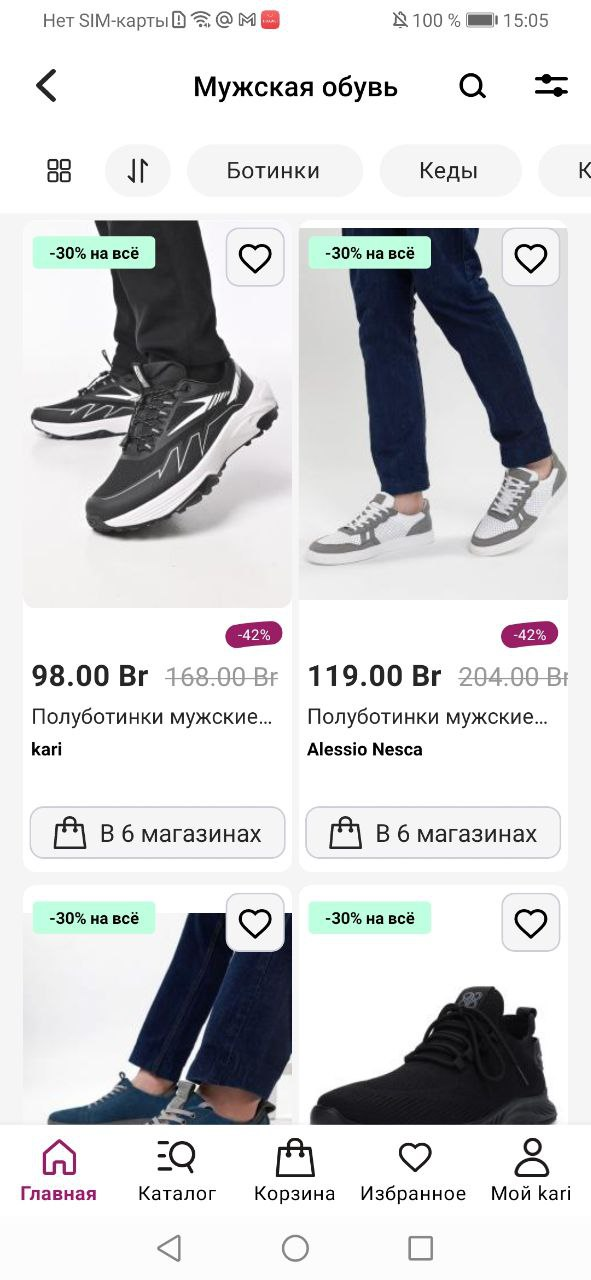


Рисунок 1.7 – Страница с каталогом в приложении «Kari»

Карточка товара в приложении содержит детальное описание выбранной обуви, фотографии с разных ракурсов, информацию о материалах, размерах и наличии, а также отзывы других покупателей. Это помогает сделать осознанный выбор и подобрать наиболее подходящую модель.

К главным особенностям данного приложения можно отнести:

* широкий выбор обуви;
* возможность отслеживания истории покупок;
* простой процесс оформления и изменения заказов.



Рисунок 1.8 –Страница товара в приложении «Kari»

В приложении реализована корзина, где можно управлять выбранными товарами: изменять их количество, удалять ненужные позиции и видеть итоговую сумму заказа. После этого достаточно нажать кнопку «Перейти к оформлению», чтобы перейти к процедуре покупки.

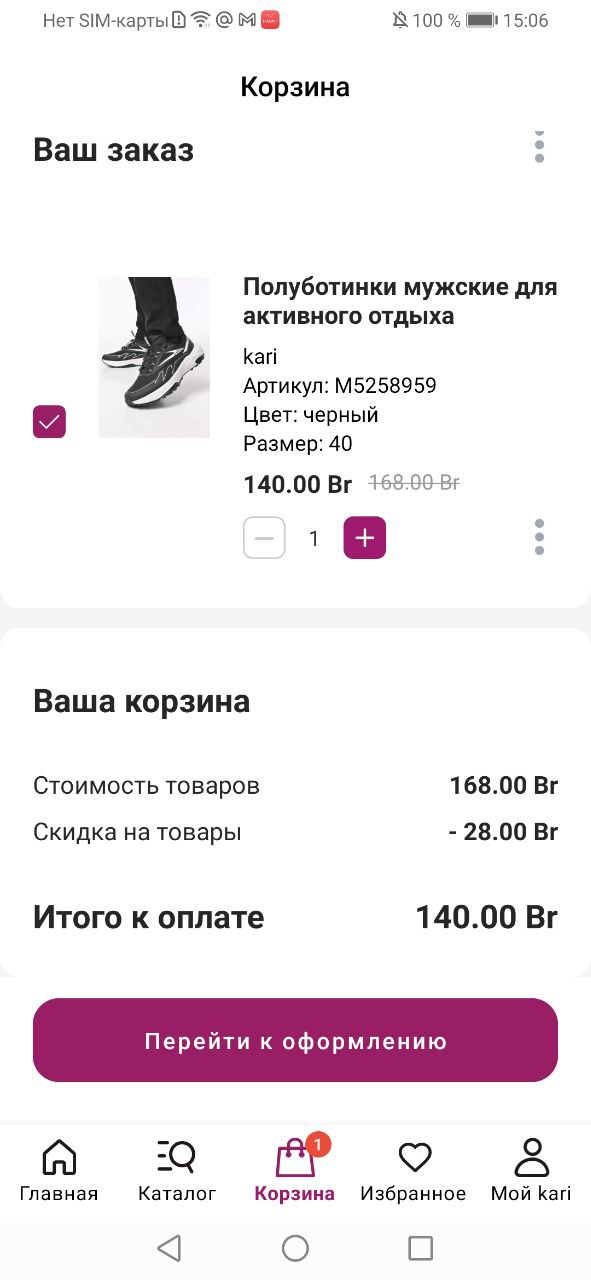
`

Рисунок 1.9 – Страница корзины в приложении «Kari»

На всех этапах оформления заказа пользователь может изменить детали, выбрать удобный способ доставки и оплаты, а также проверить актуальность введённых данных и стоимости заказа.

Несмотря на множество преимуществ, у приложения есть и некоторые недостатки. Так, для его корректной работы требуется стабильное подключение к интернету. К тому же информация о наличии некоторых моделей и актуальных акциях может обновляться с небольшой задержкой, что иногда вызывает определённое неудобство у покупателей. Тем не менее, возможность просматривать и редактировать детали заказа перед подтверждением покупки обеспечивает дополнительный контроль и удобство.

В целом, приложение «Kari» обладает действительно широким функционалом, который полностью удовлетворяет основные потребности современных покупателей обуви. Лаконичный дизайн, отсутствие лишних элементов и продуманные функции делают его эффективным инструментом для быстрой и приятной покупки обуви, в том числе спортивной. Улучшение скорости обновления данных об ассортименте и действующих скидках может сделать этот сервис ещё более привлекательным для пользователей в будущем.

## 1.2 Постановка задачи

Разрабатываемый проект, является мобильным приложением для выбора и приобретения спортивной обуви, оформление, заказ, а также отслеживания товара и у приложения должен быть следующий функционал:

* регистрация и авторизация пользователей;
* возможность просмотра товара;
* возможность фильтрации товара;
* возможность выбора товара по размеру и количеству.
* возможность поделиться товаром;
* возможность добавления товара в корзину;
* возможность оформления заказа с выбором места доставки;
* получать уведомления;
* управление личным кабинетом;
* возможность пользователю просматривать историю заказов.

## 1.3 Выводы по разделу

В данной главе были разобраны три самых популярных приложения, которые являются аналогами приложения мобильное приложение «Интернет-магазин спортивной обуви»: «Lamoda», «Kari» и «Zenden». Мы остановились на трёх приложениях, так как остальные практически не отличаются по функционалу и пользовательскому интерфейсу.

Важно отметить, что рынок мобильных приложений для онлайн-покупки обуви сегодня очень конкурентный. Каждый разработчик стремится внедрять новые решения для повышения удобства пользователей, делая упор не только на базовый функционал, но и на детали интерфейса, индивидуальные предложения и дополнительные возможности. Благодаря этому пользователи могут выбирать те сервисы, которые максимально соответствуют их ожиданиям по скорости работы, простоте оформления заказов и качеству обслуживания.

Кроме того, современные тренды в разработке мобильных приложений диктуют особое внимание к визуальному оформлению и пользовательскому опыту. Красивый и минималистичный дизайн, удобная навигация, своевременные уведомления и прозрачная система отслеживания заказов становятся неотъемлемой частью успешного продукта. Именно поэтому анализ аналогичных решений позволяет не только выявить сильные и слабые стороны конкурентов, но и сформировать собственное видение идеального приложения для интернет-магазина спортивной обуви.

Из приложений «Lamoda» была выделена необходимость работать во всех направлениях, а именно сотрудничать с разными брендами и сде­лать удобный личный кабинет со всей информацией о заказе и пользователе, а из «Kari» и «Zenden» – взять за идею карточку товара, которая сделана просто, понятно и красиво, без лишних деталей.

Проанализировав все вышесказанное, можно выделить основной и необходимый функционал разрабатываемого в дипломном проектирования мобильного приложения для спортивной обуви:

* регистрация и авторизация пользователей;
* наличие нижней навигационной панели;
* возможность просмотра информации в личном кабинете;
* поиск и сортировка товара по разным параметрам;
* просмотр заказов и их статусов.

Стоит отметить, что анализ аналогичных приложений позволяет выявить лучшие практики и технологии, применяемые другими разработчиками. Это может помочь оптимизировать функциональность и удобство использования нашего приложения, что подчеркивает важность такого обзора.

# 2 Проектирование мобильного приложения

ФИО

Подпись

Дата

Лист

1

ДП 02.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

2 Проектирование мобильного приложения

Лит.

Листов

17

БГТУ 1-98 01 03, 2025

## 2.1 Выбор технологий и средств

При реализации дипломного проекта использовались язык программирования Dart и фреймворк Flutter.

В качестве средств разработки, использовались программные средства Android Studio, Visual Studio Code.

### **2.1.1 Среда разработки Android Studio**

Android Studio – это мощная и бесплатная среда разработки, разработанная компанией Google специально для создания приложений на платформе Android. Она предоставляет обширный набор инструментов, которые значительно упрощают процесс разработки и развертывания мобильных приложений.

Android Studio предлагает широкий набор функций, направленных на повышение производительности разработчика. Среди них поддержка подсветки синтаксиса, автодополнения кода, мощные средства отладки и профилирования, а также интеграция с системами контроля версий, такими как Git. Эти возможности помогают разработчикам сосредоточиться на написании кода, снижая затраты времени на рутинные задачи.

Одним из главных преимуществ Android Studio является встроенная поддержка Gradle-системы автоматизации сборки. Gradle позволяет эффективно управлять зависимостями, конфигурациями сборки и процессом развертывания, что особенно полезно при работе с крупными проектами.

Для упрощения тестирования приложений, Android Studio предоставляет мощные инструменты, такие как эмуляторы устройств, позволяющие запускать и тестировать приложения на различных конфигурациях Android без необходимости использования физического устройства. Это значительно ускоряет процесс отладки и улучшения качества приложения.

Одной из ключевых особенностей Android Studio является его тесная интеграция с Android SDK, что обеспечивает разработчикам доступ ко всем новейшим API и библиотекам. Это позволяет быстро адаптироваться к изменениям в платформе Android и использовать новейшие возможности операционной системы.

Android Studio разработана с учетом производительности и ресурсоэффективности, что позволяет комфортно работать с ней даже на компьютерах со скромными характеристиками. Кроме того, среда разработки поддерживает настройку пользовательского интерфейса, что позволяет разработчикам адаптировать ее под свои индивидуальные предпочтения. Дополнительно Android Studio регулярно получает обновления.

На протяжении многих лет среда разработки Android Studio зарекомендовала себя как стандарт для разработки приложений под Android, благодаря своей стабильности, функциональности и поддержке сообщества. Она предоставляет все необходимые инструменты для создания высококачественных мобильных приложений, что делает ее незаменимым инструментом для Android-разработчиков. Помимо этого, регулярные обновления и улучшения Android Studio помогают разработчикам оставаться в курсе новейших технологий и методик, что способствует созданию инновационных и современных приложений.

### 2.1.2 Язык программирования Dart

Dart – это язык программирования общего назначения, разработанный компанией Google, предназначенный прежде всего для создания прикладных приложений. Его основной сферой применения на сегодняшний день является разработка графических приложений с использованием кроссплатформенного фреймворка Flutter. Flutter позволяет создавать клиентские веб-приложения, мобильные приложения под Android и iOS, а также десктопные приложения для Windows, macOS и Linux. Это означает, что мы можем написать программу на Dart с общим кодом и скомпилировать её сразу под все поддерживаемые платформы, включая Android, iOS, Windows, Linux, macOS и веб-браузеры. Однако применение языка Dart не ограничивается только фреймворком Flutter.

В своем развитии Dart испытал влияние более ранних языков, таких как Smalltak, Java, JavaScript. Его синтаксис похож на синтаксис других си-подобных языков. При этом Dart - объектно-ориентированный язык. Все значения, которые используются в программе на Dart, представляют объекты.

Dart предоставляет мощные инструменты и функции, которые существенно упрощают создание мобильных и веб-приложений. Он гарантирует отличную производительность и удобство использования благодаря статической и динамической типизации, что дает возможность выявлять ошибки на этапе компиляции и одновременно обеспечивает гибкость в написании кода. Эти особенности способствуют созданию более надежного и безопасного кода программистами, позволяя сократить время на его отладку.

Одним из ключевых преимуществ Dart является его асинхронная модель программирования, которая позволяет эффективно обрабатывать параллельные задачи, что особенно полезно в приложениях с высокой нагрузкой. Это делает Dart идеальным выбором для разработки отзывчивых пользовательских интерфейсов и работы с сетевыми запросами.

Dart также предлагает обширную стандартную библиотеку, содержащую множество полезных функций и классов для работы с коллекциями, файлами, сетевыми запросами и другими задачами. Данная библиотека существенно уменьшает время разработки, позволяя сконцентрироваться на логике приложения.

Кроме того, язык Dart активно развивается и поддерживается сообществом, что обеспечивает регулярное обновление и появление новых возможностей.

### **2.1.3 Облачная база данных Firebase**

Firebase – платформа для разработки мобильных и веб-приложений, разработанная компанией Google. Она представляет собой обширный многофункциональный инструментарий и сервисы, позволяющие разработчикам оперативно создавать, тестировать и внедрять приложения, а также эффективно управлять их функциональностью и производительностью.

Основным элементом Firebase является облачное хранилище данных Firestore, предоставляющее возможности хранения и синхронизации данных в режиме реального времени. Firestore упрощает создание и управление базами данных, обеспечивая высокую доступность и масштабируемость приложений. Поддержка оффлайн-режима гарантирует возможность работы с приложением даже при временном отсутствии интернет-соединения.

Firebase также предлагает мощные средства аутентификации пользователей, поддерживающие различные методы, включая электронную почту, социальные сети и анонимный вход. Это упрощает управление пользователями и повышает безопасность приложений.

В дополнение, Firebase содержит инструменты аналитики, которые помогают разработчикам отслеживать поведение пользователей и оценивать производительность приложений. Инструменты аналитики предоставляют ценную информацию о взаимодействии пользователей с приложением, что позволяет принимать обоснованные решения по его улучшению.

Firebase предоставляет возможность отправки уведомлений в реальном времени посредством Firebase Cloud Messaging (FCM). Этот сервис позволяет разработчикам отправлять push-уведомления пользователям на мобильных устройствах и в веб-приложениях, что способствует повышению вовлеченности и удержанию пользователей.

Другой значимой функцией Firebase является хостинг, позволяющий разработчикам быстро разворачивать статические и динамические веб-приложения с поддержкой HTTPS. Хостинг Firebase обеспечивает высокую скорость загрузки и легкость управления, что делает его отличным выбором для веб-разработки.

Firebase также интегрируется с другими сервисами Google, такими как Google Analytics и Google Cloud, что позволяет разработчикам использовать мощные инструменты для анализа и масштабирования приложений.

За последние годы Firebase подтвердил свою репутацию надежной платформы для разработки приложений, благодаря своей функциональности, удобству использования и поддержке сообщества. Регулярные обновления и новые функции делают Firebase современным инструментом, который помогает разработчикам создавать качественные и высокопроизводительные приложения.

### **2.1.4 Фреймворк Flutter**

Flutter – это бесплатный и невероятно удобный инструмент разработки приложений, созданный компанией Google. С его помощью можно создавать приложения не только для мобильных платформ Android и iOS, но и для веба и настольных систем, что делает его идеальным вариантом для разработчиков, стремящихся экономии времени и сил, разрабатывая кроссплатформенные решения.

Главное преимущество Flutter – его обширный функционал. Он включает в себя богатую библиотеку виджетов, а также функцию "горячей перезагрузки", позволяющую мгновенно видеть изменения в коде. Это значительно ускоряет процесс разработки. К тому же, интеграция с различными инструментами тестирования и отладки облегчает поиск и исправление ошибок.

Flutter также обеспечивает отличную производительность благодаря своей системе рендеринга. Он использует собственный движок для графики и анимации, поэтому приложения выглядят одинаково хорошо на всех платформах и работают плавно и быстро.

Для тестирования Flutter предоставляет мощные инструменты, включая встроенные фреймворки для модульного и интеграционного тестирования. Это позволяет тщательно проверять каждую часть приложения, обеспечивая его качество на всех этапах разработки.

Ещё одно значительное преимущество Flutter – возможность использования единой кодовой базы для всех поддерживаемых платформ. Разработчики пишут код один раз, и он работает на Android, iOS, вебе и десктопах. Это значительно экономит время и усилия, полностью избавляя от необходимости адаптировать приложение под каждую платформу отдельно.

Не стоит забывать и о поддержке сообщества. Flutter постоянно обновляется, а разработчики могут использовать множество пакетов и библиотек, упрощающих работу. Все эти обновления и улучшения способствуют созданию современных, удобных и функциональных приложений.

В целом, Flutter стал важным инструментом для многих разработчиков благодаря своей гибкости, производительности и активному сообществу. Этот фреймворк позволяет создавать отличные кроссплатформенные приложения, и он подходит для разработчиков, желающих работать быстро и эффективно.

Дополнительно стоит отметить, что Flutter предоставляет широкие возможности для интеграции с внешними сервисами и платформами. Разработчики могут использовать многочисленные плагины для подключения к облачным базам данных, системам аутентификации, платёжным сервисам и инструментам аналитики. Поддержка стандартных протоколов взаимодействия, таких как REST API, позволяет расширять функциональность приложения в соответствии с современными требованиями рынка, сохраняя при этом целостность и удобство архитектуры.

## 2.2 Проектирование базы данных

Первым шагом в разработке приложения является тщательное определение предметной области, которой оно будет охватывать, а также проектирование общей архитектуры базы данных. Необходимо четко понимать, какие именно данные будут обрабатываться, где они будут размещены и в каком формате будут представлены. На этом этапе важно также определить основные связи между сущностями и заранее предусмотреть требования к масштабируемости и безопасности хранения информации.

Логическая схема базы данных представлена на рисунке 2.2.

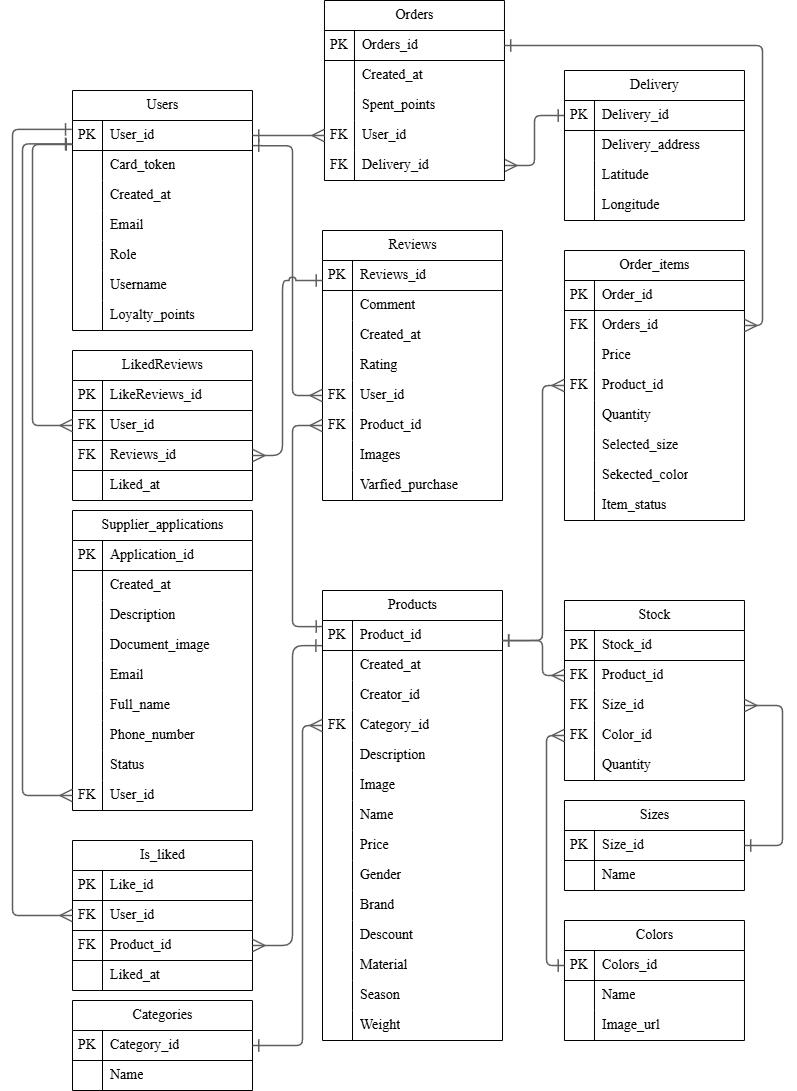


Рисунок 2.2 – Схема базы данных

Всего в базе данных 13 таблиц. Название и назначение каждой спроектированной коллекции приведено в таблице 2.1.

Таблица 2.1 – Таблицы базы данных

|  |  |
| --- | --- |
| Название таблицы | Назначение таблицы |
| Users | Хранит данные о пользователях |
| Orders | Данные о списке всех заказов |
| Products | Хранит информацию о товаре |
| Reviews | Хранит отзывы о товаре |
| Delivery | Хранит возможные места доставки товара |
| OderItems | Данные о каждом отдельном заказе |
| Categories | Хранит доступные категории товаров |
| SupplierAppliccat | Заявка на поставщика |
| IsLiked | Хранит лайки на товарах |
| LikedReview | Хранит лайки на отзывах |
| Sizes | Хранит размеры товара |
| Stock | Хранит общую информацию о размерах, цветах и количестве |
| Colors | Хранит информацию о цвете |

Подробное описание каждой коллекции приведено в таблицах 2.2 – 2.14. А именно описаны типы и названия полей, а также приведено их описание и назначение, благодаря которым можно узнать для чего используется поле.

Таблица 2.2 – Структура таблицы Delivery

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| DeliveryId | string | Уникальный идентификатор |
| DeliveryAddress | string | Точный адрес доставки |
| Latitude | number | Хранит широту |
| Longitude | number | Хранит долготу |

В таблице Delivery (таблица 2.2) содержится информация о возможных местах доставки товара. Благодаря данной таблице можно ознакомиться с пунктами доставки, узнать их местоположение, а также она содержит в себе внешний ключ для связи с таблицей Orders (таблица 2.7).

При создании товара создаются вспомогательные таблицы, которые по итогу вносятся в таблицу Stock (таблица 2.3) и помогают хранить количество товаров, доступные цвета, размеры и количество.

Таблица 2.3 – Структура таблицы Stock

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| StockId | string | Уникальный идентификатор хранилища |
| ProductId | number | Уникальный идентификатор продукта |
| SizeId | string | Уникальный идентификатор размера |
| ColorId | string | Уникальный идентификатор цвета |
| Quantity | number | Количество |

Вся информация о товаре, хранится в таблице Products (таблица 2.4). Благодаря данной таблице, продавцы могу добавлять, изменять или удалять информацию о товарах, тем самым всегда поддерживать актуальность в наличии или точной стоимости каждой позиции.

При создании товара, когда продавец вводит данные товара, они сохраняются в базе данных в таблице Products (таблица 2.4). Каждая строка в таблице Products представляет собой отдельный товар и содержит следующую информацию: идентификатор товара, дату создания товара, идентификатор категории товара, идентификатор создавшего товар, описание товара, изображение с товаром, название товара, цена, гендер товара, бренд, скидка, материал, сезон носки и вес.

Таблица 2.4 – Структура таблицы Products

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| ProductId | number | Уникальный идентификатор товара |
| CreatedAt | timestamp | Дата регистрации товара |
| CategoryId | number | Идентификатор категории |
| CreatorId | string | Идентификатор пользователя |
| Description | string | Дополнительная информация о товаре |
| Image | string | Изображение товара |
| Name | string | Полное название товара |
| Price | number | Цена товара |
| Gender | string | Для кого предназначен товар |
| Brand | string | Фирма товара |
| Descount | number | Скидка товара |
| Material | string | Материал товара |
| Season | string | Сезон носки |
| Weight | number | Вес товара |

Хранение данных в таблице Users (таблица 2.5) обеспечивает возможность сохранения и извлечения информации о пользователях, что позволяет администратору просматривать общее количество зарегистрированных клиентов.

Таблица 2.5 – Структура таблицы Users

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| UserId | string | Уникальный идентификатор пользователя |
| CardToken | string | Токен карты клиента |
| CreatedAt | timestamp | Дата создания аккаунта |
| Email | string | Электронная почта пользователя |
| Role | string | Роль пользователя |
| Username | string | Никнейм пользователя |
| LoyaltyPoints | string | Баллы лояльности |

После оформления заказа, информация об идентификаторе пользователя, датой добавления товара, статусом заказа, общей суммой, идентификатор покупателя, адресом доставки и типом оплаты попадает в таблицу Orders.

Таблица 2.6 – Структура таблицы Sizes

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| SizeId | string | Уникальный идентификатор размера |
| Name | string | Размер товара |

В таблице Orders (таблица 2.7) хранится информация о общем заказе.

Таблица 2.7 – Структура таблицы Orders

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| OrdersId | string | Уникальный идентификатор заказ(а/ов) |
| CreatedAt | timestamp | Дата и время заказа |
| TotalPrice | number | Общая сумма заказа |
| UserId | string | Статус заказа |
| DeliveryId | number | Адрес доставки заказа |

Также в таблице Reviews (таблица 2.8). хранится о отзывах, которые клиент загружает о приобретенном товаре.

Таблица 2.8 – Структура таблицы Reviews

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| ReviewsId | string | Уникальный идентификатор отзыва |
| Comment | string | Текст отзыва |
| CreatedAt | timestamp | Дата создания |
| Rating | number | Оценка товара от 0 до 5 |
| ProductId | number | Идентификатор товара |
| UserId | string | Идентификатор оставившего отзыв |
| Images | string | Изображение |
| LikedBy | string | Идентификатор пользователя |
| VarfiedPurch | boolean | Пользователь с покупкой |
| LikedId | string | Идентификатор лайка |

В таблице Categories (таблица 2.9) представлена информация о категории товара, а именно его Id и название.

Таблица 2.9 – Структура таблицы Categories

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| CategoryId | number | Уникальный идентификатор категории товара |
| Name | string | Название категории |

В таблице isLiked (таблица 2.10) хранится информация о лайкнувших определённый товар, а именно идентификатор лайка, время лайка, идентификатор пользователя и идентификатор продукта.

Таблица 2.10 – Структура таблицы IsLiked

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| LikedId | string | Уникальный идентификатор заказа |
| LikedAt | timestamp | Время лайка |
| UserId | string | Уникальный идентификатор |
| ProductId | number | Уникальный идентификатор товара |

В таблице OrderItems (таблица 2.11) хранится информация о конкретном заказе, а именно идентификатор заказа, идентификатор заказов, название товара, цена, идентификатор товара, количество и его размер, статус товара, а также выбранный цвет товара.

Таблица 2.11 – Структура таблицы OrderItems

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| OrderId | string | Уникальный идентификатор заказа |
| OrdersId | string | Идентификатор заказов |
| Name | string | Название товара |
| Price | number | Цена товара |
| ProductId | number | Идентификатор товара |
| Quantity | number | Количество товара |
| SelectedSize | string | Размер товара |
| ItemStatus | string | Статус товара |
| SelectedColor | string | Цвет товара |

В таблице Colors (таблица 2.12) хранится информация о цветах, а также изображение товара в этом цвете.

Таблица 2.12 – Структура таблицы Colors

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| ColorsId | string | Уникальный идентификатор цвета |
| Name | string | Название цвета |
| ImageUrl | string | Ссылка на изображение |

В таблице SupplierApplications (таблица 2.13) содержит информацию о заявках пользователей на становление поставщиком, а именно идентификатор заявки, дата создания, дополнительная информация, фото для документов, почта, ФИО, номер телефона, статус заказа и идентификатор отправителя.

Таблица 2.13 – Структура таблицы SupplierApplications

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| ApplicationId | string | Уникальный идентификатор заявки |
| CreatedAt | timestamp | Время создания |
| Description | string | Дополнительная информация |
| DocImage | string | Ссылка на фото с документами |
| Email | string | Почта пользователя |
| FullName | string | Полное ФИО пользователя |
| PhoneNumber | string | Номер телефона пользователя |
| Status | string | Статус заявки |
| UserId | string | Уникальный идентификатор пользователя |

В таблице LikedReview (таблица 2.14) хранится информация о лайкнувших определённый отзыв, а именно идентификатор лайка, время лайка, идентификатор пользователя и идентификатор отзыва.

Таблица 2.14 – Структура таблицы LikedReview

|  |  |  |
| --- | --- | --- |
| Поле | Тип | Назначение |
| LikeReviewsId | string | Уникальный идентификатор лайка отзыва |
| LikedAt | timestamp | Время лайка |
| UserId | string | Уникальный идентификатор |
| ReviewsId | number | Уникальный идентификатор отзыва |

В результате проведённого проектирования и структурирования данных была создана база данных, отвечающая требованиям первых трёх нормальных форм. Это обеспечивает отсутствие избыточности, целостность данных и логичное разбиение информации по взаимосвязанным сущностям. Соответствие основным принципам нормализации позволяет повысить эффективность хранения данных, упростить их обработку и минимизировать вероятность возникновения аномалий при обновлении, удалении или добавлении записей.

## 2.3 Проектирование мобильного приложения

Для создания мобильного приложения был выбран язык программирования Dart и фреймворк Flutter [7]. Использование данного языка и фреймворка предоставляет ряд значимых преимуществ:

* высокая производительность благодаря компиляции в нативный код, что обеспечивает высокую скорость работы и быструю загрузку;
* встроенная поддержка асинхронного программирования, что позволяет управлять асинхронными задачами и повышает отзывчивость приложения;
* поддержка JIT компиляции для быстрой разработки и тестирования.

Фреймворк Flutter дополняет преимущества Dart следующими функциями:

* hot Reload, который дает мгновенное применение изменений в коде и немедленный просмотр результата в режиме реального времени, что ускоряет процесс разработки и тестирования;
* использование готовых настраиваемых виджетов, которые следуют рекомендациям по дизайну Material Design (для Android) и Cupertino (для iOS), что позволяет быстро создавать качественные и стильные интерфейсы.

Кроме того, использование Dart и Flutter обеспечивает единообразие в разработке, поскольку Dart является основным языком программирования для Flutter, что упрощает командную работу и поддержку проекта. Подход Flutter к созданию интерфейса также позволяет создавать адаптивные приложения, которые могут автоматически адаптироваться под разные устройства и экраны. Благодаря активной и развивающейся сообществу разработчиков, Dart и Flutter постоянно обновляются и улучшаются, что обеспечивает доступ к новейшим технологиям и инструментам для создания высококачественных мобильных приложений.

Для создания мобильного приложения необходимо в терминале ввести команду «flutter create shop\_project», которая автоматически создает проект с разделением на папки и структурированным кодом.

Папка lib содержит файл main, который является главным файлом приложения. Именно там начинается выполнение приложения. Остальные файлы в этой папке содержат коды страниц приложения. В каждом файле отдельная страница представлена в виде класса, в котором описывается разметка страницы, а также обработка всех функций и создание маршрутов для связи с сервером.

Папка test содержит тесты для приложения, где можно создавать файлы тестов для различных частей приложения, таких как виджеты или модели.

Папка build создается автоматически при сборке приложения и содержит артефакты сборки. Папка android содержит конфигурационные файлы и код для сборки приложения под Android. Папка ios содержит конфигурационные файлы и код для сборки приложения под iOS. Папка web содержит файлы для сборки приложения под веб. Здесь находятся файлы index.html, main.dart.js и другие ресурсы для веб-сборки. Папка macos содержит конфигурационные файлы и код для сборки приложения под macOS. Папка windows содержит конфигурационные файлы и код для сборки приложения под Windows. Папка linux содержит конфигурационные файлы и код для сборки приложения под Linux.

Файл pubspec.yaml используется для управления зависимостями проекта. Здесь указываются пакеты, которые используются в приложении, а также такие настройки, как название приложения, версия и ресурсы.

Файл README.md содержит описание проекта, инструкции по установке и использованию. В папке assets хранятся статические ресурсы приложения, такие как изображения, шрифты. Эти ресурсы указываются в файле pubspec.yaml.

Пример описанной архитектуры приведен на рисунке 2.3.



Рисунок 2.3 – Пример архитектуры файлов мобильного приложения

Страницы являются основополагающими компонентами приложения. На них ведет маршрутизация со стороны клиента. Каждая страница отвечает за отдельные функции, предназначенные именно ей.

В Flutter приложении страницы представляют собой виджеты, которые описывают пользовательский интерфейс для конкретного экрана. Эти виджеты могут включать в себя различные элементы: текст, изображения, кнопки и формы.

Маршрутизация между страницами осуществляется с помощью навигационных методов, таких как Navigator.push и Navigator.pop, которые позволяют перемещаться вперед и назад между страницами.

Каждая страница в приложении может иметь свою собственную логику и состояние, управляемое с помощью различных методов управления состоянием, таких как StatefulWidget, Provider или Bloc.

Страницы также могут взаимодействовать друг с другом, передавая данные через параметры маршрутизации или используя глобальные состояния.

Страницы были разделены на следующие логические сущности:

* addCategoryScreen – страница добавления категории;
* addProductScreen – страницы добавления товара;
* authScreen – страница авторизации;
* cartScreen – страница корзины;
* productDetailScreen –страница с товаром;
* productListScreen – страница со списком товаров;
* profileScreen – страница профиля;

Такое большое количество страниц было необходимо для обеспечения максимального комфорта и интуитивного взаимодействия пользователя с мобильным приложением. А также для разбиения классов, что удобно при написании кода, когда нужно обратиться к определенному классу. Всегда удобно, когда на определенную функцию приложения есть своя страница.

## **2.4 Диаграммы и блок-схемы приложения**

В этом подразделе будут представлены такие блок-схемы и диаграммы как:

* диаграмма вариантов использования;
* диаграмма компонентов;
* блок-схема оформления заказа.

### **2.4.1 Диаграмма вариантов использования**

В диаграмме вариантов использования, отображено разделение возможностей администратора, пользователя, поставщика и продавца.

В диаграмме чётко выделены роли каждого участника системы, что позволяет наглядно представить, какие действия доступны для каждой категории пользователей. Такой подход способствует лучшему пониманию структуры приложения и помогает выявить потенциальные точки взаимодействия между различными ролями. Кроме того, визуализация вариантов использования упрощает процесс проектирования интерфейса и разработки функционала, обеспечивая более эффективное распределение задач между членами команды разработки. Диаграмма вариантов использования администратора обозначена на рисунке 2.5.



Рисунок 2.4 – Диаграмма вариантов использования администратора

Администратор может управлять заказами и их статусами, а также отзывами. Диаграмма вариантов использования пользователя представлена на рисунке 2.6.

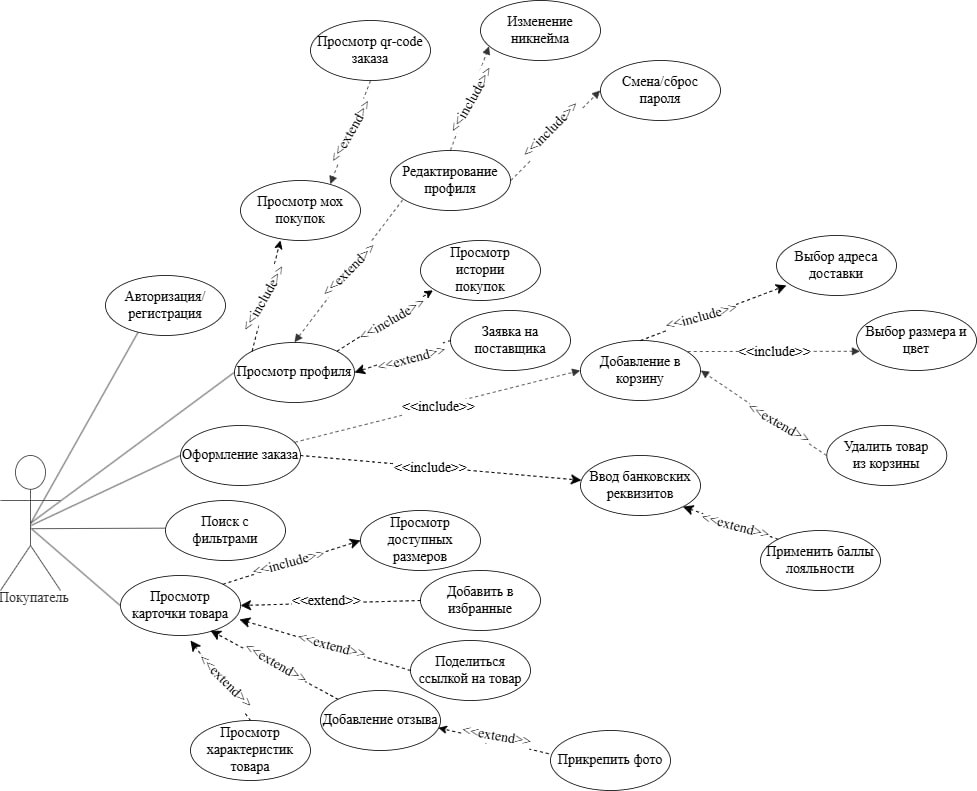


Рисунок 2.5 – Диаграмма вариантов использования пользователя

Пользователь может осуществлять поиск и добавлять необходимые ему товары в корзину, после чего оформлять заказ добавленных в корзину товаров, просматривать заказы.

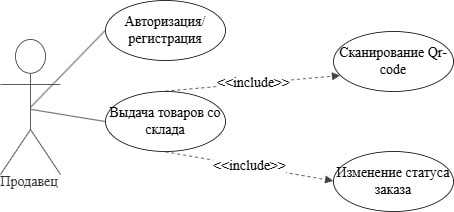


Рисунок 2.5 – Диаграмма вариантов использования продавца

Продавец может создавать и редактировать собственный товар.



Рисунок 2.6 – Диаграмма вариантов использования поставщика

Поставщик может осуществлять создание и редактирование собственных товаров, а также их удаление.

### **2.5.1 Диаграмма развертывания**

Диаграмма развертывания помогает визуализировать архитектуру системы и понять, как различные компоненты взаимодействуют друг с другом.

Для разработки мобильного приложения на основе операционной системы Android был задействован современный язык программирования Dart. В качестве облачной БД использовался сервис Firebase Database.

Диаграмма развертывания представлена на рисунке 2.7.

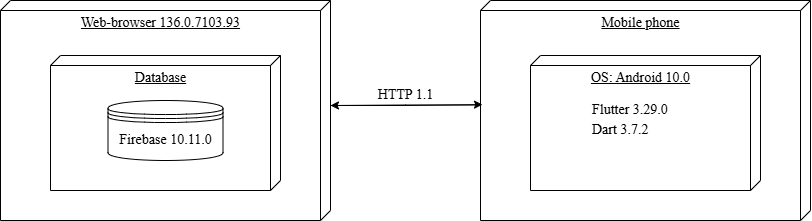


Рисунок 2.7 – Диаграмма развертывания

Цель диаграммы развертывания – предоставить обзорную картину архитектуры системы, выделить основные компоненты и их взаимодействие, а также помочь разработчикам и архитекторам понять и изменять систему.

### **2.5.2 Блок-схема оформления заказа из заведения**

Блок-схема является мощным инструментом, который позволяет визуализировать и упорядочить последовательность действий, потоки данных, принятие решений, ветвления и циклы в компактной и понятной форме. Она представляет собой графическое изображение, которое помогает лучше понять сложные процессы. В блок-схеме каждое действие или компонент представлен прямоугольным блоком, в котором четко указано его название или описание. Эти блоки соединены линиями, которые обозначают потоки данных, управления или связи между компонентами системы. Линии могут иметь различные формы и типы стрелок, что позволяет указать на разные виды связей и направлений потока информации.

Блок-схема является удобным инструментом для анализа и проектирования алгоритмов программ, планирования задач и разработки архитектуры систем. Блок схема оформления заказа представлена в приложении А.

Начинается всё с того, что пользователь попадает на страницу авторизации/регистрации, где ему предоставляется возможность ввести данные от аккаунта или же создать новый. В случае успешного прохождения регистрации/авторизации пользователь попадает на экран с каталогом товаров. Далее выбирает необходимый ему товар, а также размеры и цвет. После выполнения всех действий выше становится доступна кнопка «Добавить в корзину». После нажатия пользователь автоматически попадет на страницу с корзиной, где может выбрать адрес из выпадающего списка или используя метки на карте. Если все обязательные поля заполнены у пользователя выпадает форма с заполнением банковских реквизитов. После успешного оформления заказа пользователя перенаправляет на страницу профиля, где можно просматриваться статус заказанного товара.

### **2.5.3 Блок-схема смены пароля**

Также на рисунке 2.8 представлена блок-схема смены пароля пользователям. Которая отражает все возможные способы смены пароля с различными статусами пользователя.

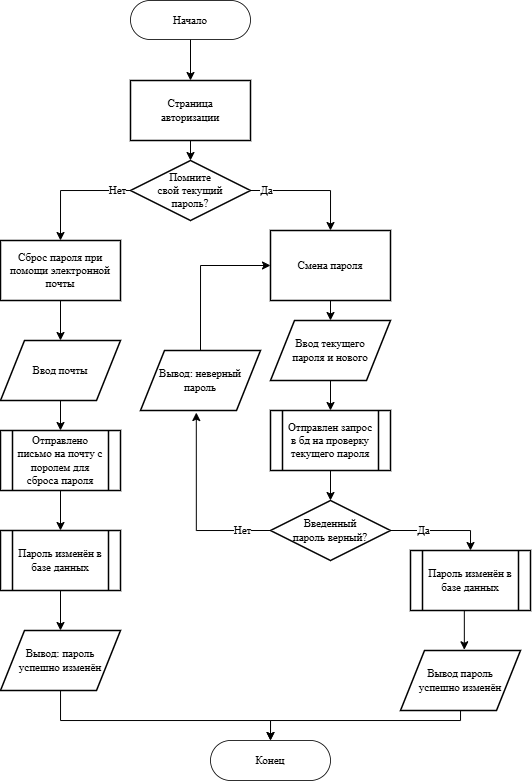


Рисунок 2.8 – Блок-схема смены пароля

Изначально пользователь попадает на страницу авторизаций и пользователю предоставляется два варианта смены пароля. Первый – если пользователь не помнит свой текущий пароль. Пользователь выбирает вариант сброса при помощи электронной почты и заполняет форму. Затем через короткий промежуток времени на почту пользователя приходит письмо с ссылкой на страницу смены пароля. Если же пользователь не сам с инициировал отправку письма, то там же будет короткая инструкция для последующих действий. Второй – если пароль известен пользователю. Тогда пользователь вводит свой предыдущий пароль и новый, затем данные сверяются и если все введённые данные верны, то пароль успешно изменён.

Использование блок-схемы позволяет не только визуализировать процесс, но и выявить потенциальные ошибки и исключения, что, в свою очередь, способствует повышению надежности системы. Благодаря этому разработчики могут заранее обнаружить слабые места в алгоритме и предусмотреть возможные проблемы. Кроме того, блок-схема значительно облегчает коммуникацию между членами команды. Все участники проекта могут быстро и четко понять сложные процессы, что способствует более эффективному взаимодействию и совместной работе. Это особенно важно в командах с разными уровнями опыта и понимания.

Смена пароля одна из важных составляющих для сохранения конфиденциальности и безопасности пользователя приложения.

## 2.7 Выводы по разделу

В данном разделе был выполнен комплекс работ по проектированию мобильного приложения. Проведен обоснованный выбор технологий: язык Dart и фреймворк Flutter обеспечивают кроссплатформенность, высокую производительность и удобство разработки, а использование Firebase позволяет реализовать надежное хранение и обработку данных, а также ряд дополнительных сервисов для повышения функциональности приложения.

Спроектирована база данных, состоящая из 13 взаимосвязанных таблиц, что соответствует требованиям нормализации и позволяет эффективно управлять пользовательскими и товарными данными. Такое построение структуры данных минимизирует избыточность и обеспечивает целостность информации.

В архитектуре приложения определены основные папки, файлы и логика их взаимодействия. Разделение интерфейса на отдельные страницы по функциональным областям обеспечивает удобство пользования и сопровождения приложения. Визуальная документация в виде диаграмм вариантов использования, развертывания и блок-схем ключевых процессов позволила чётко описать взаимодействие между ролями пользователей, а также основные бизнес-процессы приложения.

В целом, все принятые проектные решения закладывают прочную основу для последующей реализации, обеспечивают масштабируемость, безопасность и удобство дальнейшей работы над мобильным приложением.

# 3 Реализция приложения

ФИО

Подпись

Дата

Лист

1

ДП 03.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

3 Реализация приложения

Лит.

Листов

8

БГТУ 1-98 01 03, 2025

В этом разделе будет рассмотрена реализация приложения, охватывающая серверную часть, клиентскую, а также взаимодействие между ними.

Реализация приложения проводилась в среде Android Studio 2024.

Клиентская часть приложения разработана в среде Android Studio с использованием языка программирования Dart и фреймворка Flutter. Такой выбор технологий позволил создать современный, удобный и отзывчивый пользовательский интерфейс для мобильных устройств на базе Android.

## 3.1 Взаимодействие с базой данных

Современные мобильные приложения требуют надёжного, масштабируемого и удобного механизма хранения и обмена данными между пользователями и сервером. Для решения этих задач в рамках моего дипломного проекта был выбран облачный сервис Firebase от Google, который предоставляет целый набор инструментов для аутентификации, хранения данных, обмена сообщениями и других типовых задач мобильной разработки. Firebase обладает рядом преимуществ: он не требует собственной инфраструктуры на старте, легко масштабируется, предоставляет мощные SDK для Flutter, а также гарантирует безопасность и скорость работы даже при большом количестве пользователей.

dependencies:

flutter:

sdk: flutter

cloud\_firestore: ^5.6.5

firebase\_core: ^3.12.1

firebase\_auth: ^5.5.1

firebase\_storage: ^12.4.4

firebase\_messaging: ^15.2.5

google\_sign\_in: ^6.3.0

shared\_preferences: ^2.5.2;

Листинг 3.3 – Пакеты для работы с Firebase

На листинге 3.3 можно увидеть основные пакеты, которые использовались для работы с сервисом Firebase.

Для структурирования кода и удобства дальнейшей поддержки вся работа с Firebase вынесена в отдельный сервисный класс FirebaseService. Это не просто набор функций, а полноценный singleton, то есть единый экземпляр сервиса, который используется во всём приложении. Такой подход гарантирует централизованный и последовательный доступ к базе данных и средствам авторизации.

class FirebaseService

{

static final FirebaseService \_singleton = FirebaseService.\_internal();

factory FirebaseService() => \_singleton;

FirebaseService.\_internal();

final auth = FirebaseAuth.instance;

final firestore = FirebaseFirestore.instance;

}

Листинг 3.4 – Объявление класса FirebaseService

В современном мобильном приложении, особенно если речь идёт о магазине, защита пользовательских данных и удобство входа – необходимые условия успеха. В моём проекте реализована многоуровневая система аутентификации на базе сервисов Firebase, что обеспечивает как безопасность, так и гибкость для пользователя.

Реализация вынесена в отдельный сервисный класс FirebaseService. Такой подход позволяет не дублировать логику и централизовать все процессы, связанные с аккаунтом пользователя – от регистрации до выхода.

Регистрация – первый шаг к персонализации работы в приложении. Пользователь вводит email, пароль и имя. Вся логика работы с аккаунтами реализована через сервис Firebase Auth, а профиль пользователя сохраняется в Firestore.

Код функции регистрации представлен на листинге 3.5.

Future<void> onRegister({

required String email,

required String password,

required String username,

String role = "user"

}) async {

final credential = await auth.createUserWithEmailAndPassword(

email: email,

password: password,

);

String userId = credential.user!.uid;

await firestore.collection('users').doc(userId).set({

'username': username,

'email': email,

'role': role,

'created\_at': FieldValue.serverTimestamp(),

'loyalty\_points': ''

});

await savePushToken(userId);

}

Листинг 3.5 – Функция регистрации

После регистрации пользователь может сразу войти в систему и пользоваться всеми возможностями магазина.

Для максимального удобства пользователей реализована авторизация через Google. Этот способ особенно удобен тем, что избавляет человека от необходимости придумывать и запоминать новый пароль для ещё одного сервиса. Всё, что требуется – выбрать свой Google-аккаунт и подтвердить действие. Процесс занимает буквально несколько секунд и не требует никаких дополнительных усилий: ни подтверждения почты, ни заполнения длинных форм. Пользователь нажимает на кнопку «Войти через Google» в приложении. Следом открывает стандартный диалог Google, где пользователь может выбрать любой из своих аккаунтов. Если пользователь уже входил ранее и выбрал действие запомнить меня, то вход выполнится автоматически без ожиданий действий от пользователя. Код с функцией входа через Google-аккаунт представлен на листинге 3.6.

Future<void> signInWithGoogle(BuildContext context) async {

final GoogleSignInAccount? googleUser = await googleSignIn.signIn();

if (googleUser == null) return;

final GoogleSignInAuthentication googleAuth = await googleUser.authentication;

final credential = GoogleAuthProvider.credential(

accessToken: googleAuth.accessToken,

idToken: googleAuth.idToken,

);

final UserCredential userCredential = await auth.signInWithCredential(credential);

final User? user = userCredential.user;

if (user != null) {

final DocumentSnapshot userDoc = await firestore.collection('users').doc(user.uid).get();

if (!userDoc.exists) {

await firestore.collection('users').doc(user.uid).set({

'username': user.displayName ?? 'Google User',

'email': user.email,

'role': 'user',

'created\_at': FieldValue.serverTimestamp(),

'loyalty\_points': ''

});

}

await savePushToken(user.uid);

await rememberUser(true);

Navigator.pushReplacementNamed(context, '/products');

}

}

Листинг 3.6 – Функция входа через Google

Работает это по такому принципу, что если у пользователя уже есть аккаунт, то происходит автоматическая авторизация, а если заходит впервые, то будет автоматически предлагать создавать новый профиль.

void onListenUser (void Function(User?)? doListen) {

auth.authStateChanges().listen(doListen);

}

Листинг 3.7 – Функция отслеживания статуса пользователя

При запуске приложения слушатель отслеживает, авторизован ли пользователь, и если нет, то перенаправляет на экран входа. Благодаря методу onListenUser приложение всегда в курсе текущего состояния сессии.

Одним из ключевых сценариев взаимодействия с базой данных в мобильном приложении магазина является оформление заказа пользователем. Этот процесс включает не только добавление информации о заказе в базу данных, но и создание связанных записей о товарах внутри заказа, а также автоматический пересчет баллов лояльности, которые могут быть использованы для будущих покупок. Такой подход обеспечивает комплексную обработку всех сопутствующих данных в рамках единой функции, что повышает надёжность и прозрачность бизнес-логики приложения.

Future<void> placeOrder(

List<Map<String, dynamic>> cartItems, double totalPrice, String deliveryId) async {

final userId = FirebaseAuth.instance.currentUser?.uid;

if (userId == null) return;

DocumentReference orderRef = await firestore.collection('orders').add({

'user\_id': userId,

'total\_price': totalPrice,

'created\_at': FieldValue.serverTimestamp(),

'delivery\_id': deliveryId,

});

for (var item in cartItems) {

await firestore.collection('order\_items').add({

'order\_id': orderRef.id,

...item,

'item\_status': 'В пути',

});

}

double loyaltyPoints = totalPrice \* 0.02;

DocumentSnapshot userDoc = await firestore.collection('users').doc(userId).get();

double currentPoints = double.tryParse(userDoc['loyalty\_points'] ?? '0') ?? 0.0;

double newPoints = currentPoints + loyaltyPoints;

await firestore.collection('users').doc(userId).update({

'loyalty\_points': newPoints.toStringAsFixed(2),

});

}

Листинг 3.8 – Функция отслеживания статуса пользователя

При оформлении заказа в приложении реализуется последовательная работа с базой данных Firestore: сначала создаётся документ заказа с основными параметрами (идентификатор пользователя, сумма, дата, адрес или способ доставки), после чего для каждого товара из корзины формируется отдельная запись в коллекции order\_items со ссылкой на заказ и статусом позиции. Завершающим этапом становится автоматическое начисление баллов лояльности 2% от суммы заказа добавляются к текущему количеству баллов пользователя. Такой механизм позволяет не только централизованно хранить все связанные с заказом данные, но и поощрять пользователей за покупки, повышая их вовлечённость и лояльность к магазину. Всё это реализовано в одном методе, что упрощает сопровождение и масштабирование бизнес-логики приложения.

## 3.2 Разработка клиентской части

При создании клиентской части приложения особое внимание уделялось не только технической реализации, но и удобству пользовательского опыта. Основная цель – сделать так, чтобы взаимодействие с магазином было интуитивным, современным и приятным независимо от возраста и опыта пользователя.

Интерфейс построен по принципу mobile-first. Для оформления выбран тёмный стиль с акцентными цветами, что не только придаёт приложению современный внешний вид, но и способствует меньшей утомляемости глаз при длительной работе. В ходе проектирования дизайна были изучены популярные мобильные приложения в сфере e-commerce, чтобы учесть лучшие решения и создать свой узнаваемый стиль.

final filteredProducts = products.where((product) {

final matchesCategory = \_selectedCategory == null || product['category'] == \_selectedCategory;

final matchesColor = \_selectedColor == null || product['color'] == \_selectedColor;

final matchesPrice = product['price'] >= \_priceRange.start && product['price'] <= \_priceRange.end;

return matchesCategory && matchesColor && matchesPrice;

}).toList();

Листинг 3.9 – Фильтрация по нескольким условиям

Основной экран приложения – это каталог товаров. Пользователь сразу видит витрину с удобной поисковой строкой, фильтрами по категориям, цветам, брендам, размеру, сезону и другим параметрам. Такой подход позволяет сразу сосредоточиться на поиске нужного товара, не отвлекаясь на лишние элементы интерфейса. Реализован быстрый поиск как по названию, так и по описанию товара, что делает работу с приложением интуитивно понятной и удобной. Гибкая фильтрация позволяет сузить выбор до конкретных позиций, используя несколько условий одновременно, например, выбрать товары определённого бренда, соответствующие заданному сезону и размеру.

Фильтрация и поиск происходят локально после загрузки данных, что обеспечивает моментальный отклик интерфейса. Это значит, что пользователю не нужно ждать загрузки данных с сервера при каждом изменении фильтров или вводе запроса. Такой подход не только ускоряет работу приложения, но и делает взаимодействие с ним более комфортным, особенно при плохом соединении с интернетом. На рисунке 3.1 продемонстрировано как выглядит фильтрация в самом приложении.

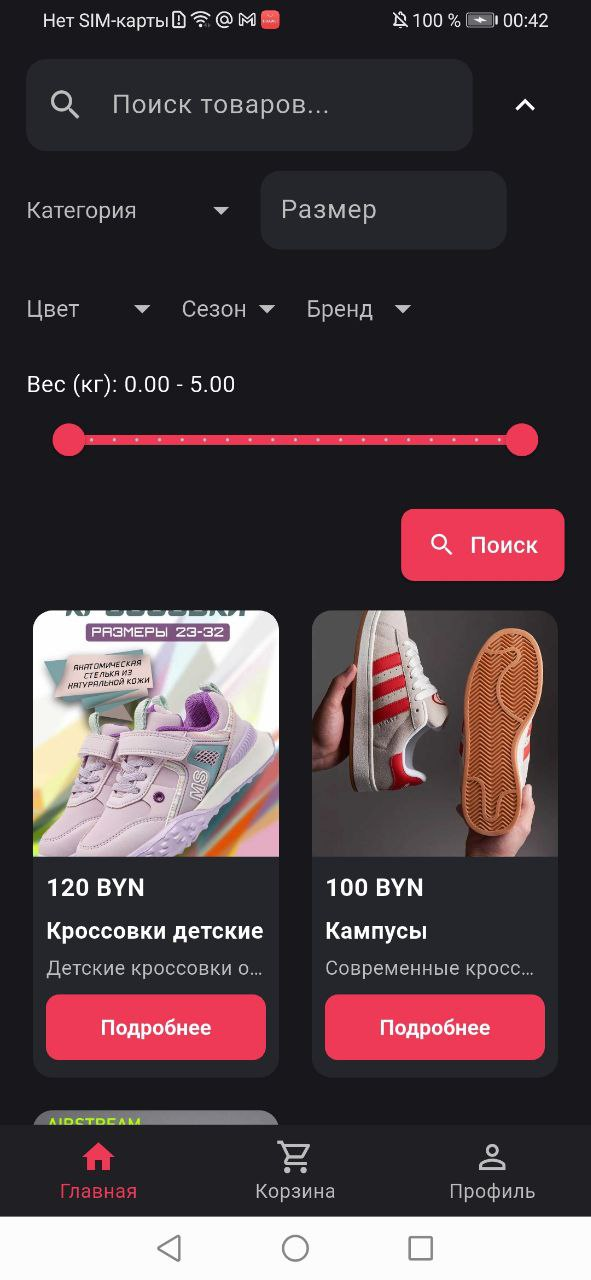


Рисунок 3.1 – Фильтрация товаров

После оформления заказа необходимо актуализировать остатки товаров, чтобы другие пользователи видели только реально доступные позиции. Для этого реализована функция обновления количества товара по выбранному размеру и цвету. Она вызывается при каждом успешном заказе или возврате, и работает напрямую с базой данных Firestore.

Future<void> updateProductStock(

int productId, String size, String color, int quantityChange) async {

final productQuery = await firestore.collection('products')

.where('product\_id', isEqualTo: productId).limit(1).get();

if (productQuery.docs.isNotEmpty) {

final productDoc = productQuery.docs.first;

final productRef = productDoc.reference;

final productData = productDoc.data() as Map<String, dynamic>;

final sizes = productData['sizes'] as Map<String, dynamic>?;

if (sizes != null && sizes.containsKey(size)) {

final sizeData = sizes[size] as Map<String, dynamic>;

final colorQuantities = sizeData['color\_quantities'] as Map<String, dynamic>?;

if (colorQuantities != null && colorQuantities.containsKey(color)) {

final currentStock = colorQuantities[color] as int;

final newStock = currentStock + quantityChange;

if (newStock >= 0) {

await productRef.update({

'sizes.$size.color\_quantities.$color': newStock,});}}}}}

Листинг 3.9 – Обновление остатков товара

Данная функция находит нужный товар по его ID, далее нужный размер и цвет внутри структуры в базе, и изменяет значение количества (например, уменьшает на 1 при покупке или увеличивает при возврате). Благодаря этому остатки всегда актуальны, и пользователь не сможет заказать больше, чем есть на складе. Также имеется реализация функции в случае отказа от товара.

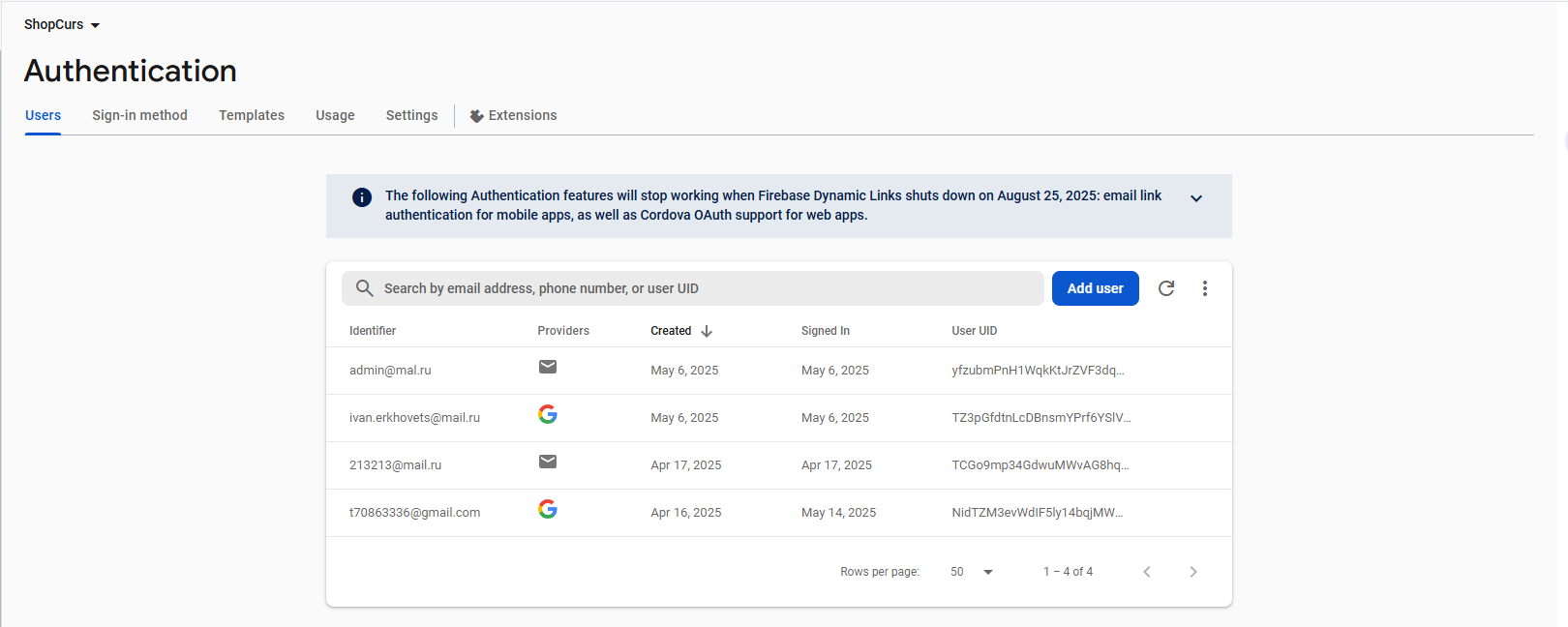


Рисунок 3.1 – Аутентификация на стороне Firebase

На рисунке 3.1 представлено изображение как выглядит хранение данных пользователей на стороне сервера. Все пароли хранятся в хэш виде.

Для повышения вовлечённости пользователей реализована возможность ставить лайки к отзывам на товары. Это позволяет отмечать полезные комментарии и видеть, какие отзывы популярны среди других покупателей.

Future<void> \_toggleLikeReview(String reviewId) async {

final user = FirebaseAuth.instance.currentUser;

if (user == null) return;

final userId = user.uid;

final snapshot = await transaction.get(reviewDoc);

if (!snapshot.exists) throw Exception("Отзыв не найден");

final data = snapshot.data() as Map<String, dynamic>;

final List<dynamic> likedBy = List.from(data['liked\_by'] ?? []);

final isLiked = likedBy.contains(userId);

if (isLiked) {

likedBy.remove(userId);

} else {

likedBy.add(userId);

}

transaction.update(reviewDoc, {

'liked\_by': likedBy,

'likes': likedBy.length,

});

});

}

Листинг 3.10 – Лайки для комментариев

В приведённом листинге показана функция, которая по нажатию на кнопку лайка добавляет или убирает текущего пользователя из списка проголосовавших за конкретный отзыв. Операция выполняется в транзакции Firestore, чтобы избежать конфликтов при одновременных действиях нескольких пользователей. Количество лайков пересчитывается автоматически.

Wrap(

children: availableColors.map((color) {

final selected = \_selectedColor == color;

final inStock = (\_availableQuantities[\_selectedSize!]?[color] ?? 0) >0;

return GestureDetector(

onTap: inStock ? () => \_onColorSelected(color) : null,

child: Container( width: 38,height: 38,

decoration: BoxDecoration(

shape: BoxShape.circle, color: \_getColorFromString(color),

border: Border.all(

color: selected ? Colors.pink : Colors.grey,

width: selected ? 3 : 1, ),),

child: !inStock

? Icon(Icons.close, color: Colors.white70, size: 18)

: selected

? Icon(Icons.check, color: Colors.white, size: 16)

: null,

),);}).toList(),)

Листинг 3.11 – Выбор цвета товара

Для того что бы сразу показывать пользователю, какие варианты товара доступны для покупки, а какие закончились. Реализована специальная зона выбора цвета: каждый цвет представлен в виде круглой иконки, причём недоступные цвета визуально "заблокированы" и не реагируют на нажатие.

## 3.3 Выводы по разделу

В процессе создания клиентской части приложения были реализованы ключевые функции, направленные на повышение удобства, безопасности и интерактивности для пользователя. Использование Flutter позволило создать адаптивный и быстрый интерфейс, одинаково корректно отображающийся на различных устройствах, а интеграция с облачным сервисом Firebase обеспечила надёжную аутентификацию, хранение данных и централизованное управление процессами через отдельный сервисный класс. К числу основных функций относятся быстрый поиск и фильтрация товаров, автоматическое обновление остатков на складе после каждой покупки или возврата, многоуровневая система аутентификации с поддержкой Google, возможность оставлять отзывы с фотографиями и лайками, push-уведомления и поддержка переходов по прямым ссылкам. В результате клиентская часть приложения стала не только современной и функциональной, но и готовой к дальнейшему развитию. Такой подход обеспечивает конкурентоспособность.

# 4 Тестирование приложения

ФИО

Подпись

Дата

Лист

1

ДП 06.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

6 Технико-Экономическое обоснование проекта

Лит.

Листов

5

БГТУ 1-98 01 03, 2025

Тестирование программного средства представляет собой процесс проверки совпадения полученных и ожидаемых результатов работы программы на заранее определённом наборе тестов. Такой подход позволяет выявлять несоответствия между фактическим поведением приложения и установленными требованиями, а также находить ошибки до того, как ими столкнутся реальные пользователи. Основная задача тестирования заключается в том, чтобы убедиться в соответствии программного обеспечения предъявляемым стандартам, повысить уверенность в его качестве и своевременно обнаружить очевидные дефекты.

Современные методы тестирования программного обеспечения, несмотря на своё многообразие, не способны полностью гарантировать отсутствие ошибок и абсолютную корректность работы программы, особенно если речь идёт о закрытых или частных решениях. Вследствие этого все применяемые подходы к тестированию функционируют в рамках формального процесса контроля качества разрабатываемого или анализируемого программного продукта.

Проведение функционального тестирования даёт возможность обнаружить и исправить ошибки на ранних этапах, что значительно повышает качество и стабильность приложения до его выхода на рынок. Всё это в конечном итоге способствует созданию продукта, удобного для пользователей и соответствующего их ожиданиям.

Тестирование занимает ключевое место в процессе разработки, так как обеспечивает соответствие всем требованиям заказчика, предотвращает появление критических недочётов и способствует укреплению репутации разработчиков. Кроме того, благодаря тестированию удаётся выявить потенциальные уязвимости в системе безопасности, что снижает риски утечек данных и других негативных последствий. Подобный подход позволяет не только обеспечить высокое качество программного средства, но и повысить удовлетворённость конечных пользователей за счёт надёжной и безопасной работы приложения.

## 4.1 Тестирование основной роли Android-клиента

Валидация полей для регистрации представлена на рисунке 4.1. В валидации можно увидеть, что все поля обязательны для заполнения. Имеется валидация, которая проверяет введённый адрес электронной почты и если такой уже имеется – сообщает пользователю. Каждое поле формы имеет свои ограничения, такие как минимальная и максимальная длина.

Проверки помогают предотвратить создание дублирующихся аккаунтов и гарантирует, что каждый пользователь имеет уникальные учётные данные.

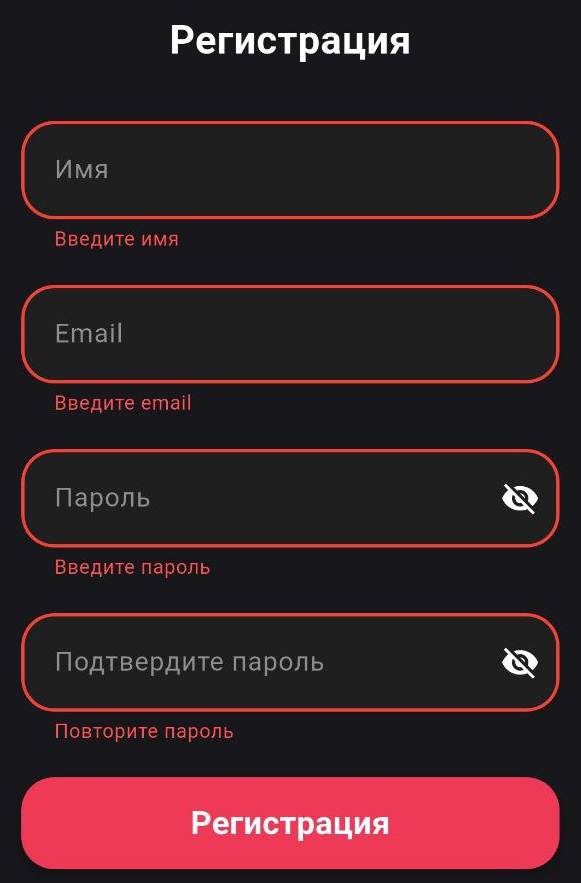


Рисунок 4.1 – Валидация полей при регистрации

Исходя из рисунка 4.1 можно заметить, что валидация распространяется на все поля, нельзя оставить один из полей пустым, тогда пользователь не сможет зарегистрироваться. Если email занят, то пользователю показывается соответствующее сообщение об ошибке.

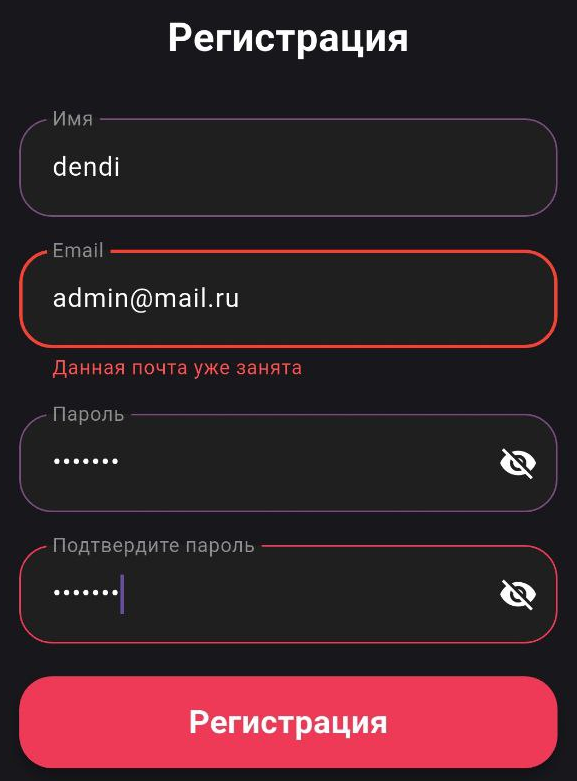


Рисунок 4.2 – Введенный email занят

Стоит отметить и то, что в окне авторизации содержится такая же валидация полей и подсказки, когда, например, пользователь пытается войти с незарегистрированной электронной почты, продемонстрировано на рисунке 4.3.

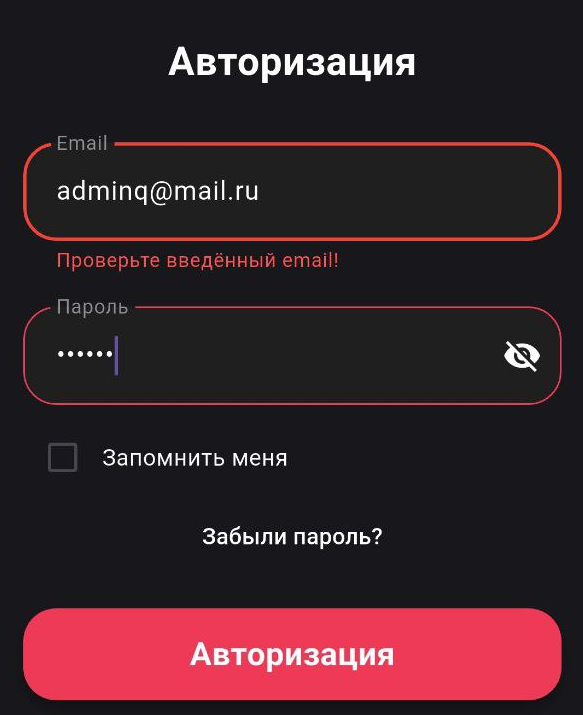


Рисунок 4.3 – Введенный почта не существует

Валидация и всевозможные проверки существуют в приложении повсеместно, но более значимые это те – из-за которых может быть нарушена безопасность пользователя. На рисунке 4.4 продемонстрирована форма смены пароля.



Рисунок 4.4 – Форма для смены пароля с валидацией

Мобильное приложение имеет только портретную ориентация, что является частым подходом в такого рода приложениях, позволяет избавиться от ненужной адаптации и существенно сокращает время написания кода.

## 4.1 Тестирование вспомогательных ролей Android-клиента

Каждый пользователь после того, как получает роль «поставщик» может выставлять свои товары в общий каталог. Для того, чтобы пользователь не забыл указать различные параметры у своего товара, существует валидация, которая не даст создать «пустой» товар без каких-либо значений.

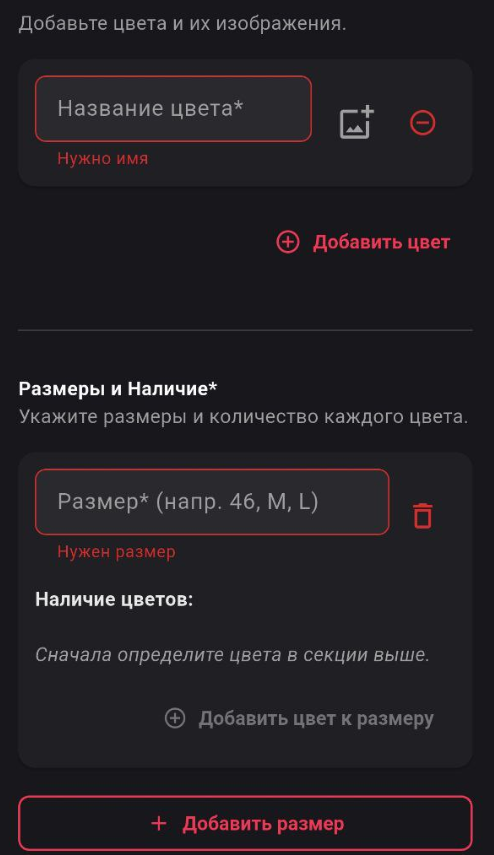


Рисунок 4.5 – Валидация создания товара

На рисунке 4.5 продемонстрировано валидация создания товара. При создании цвета товара поставщик обязан включить его хотя-бы один раз в размер. Также поставщик не может создать размер и выставить значение 0 на складе.



Рисунок 4.5 – Валидация создания адреса

На рисунке 4.5 представлена изображение с созданием адреса доставки. Для корректного создания нужно либо полностью указать в письменной форме или же выбрать меткой на карте.

В данной главе были рассмотрены самые основные моменты, с которыми пользователь можно столкнуться в повседневном использовании. Не исключено, что в приложении присутствуют некоторые баги, о которых еще неизвестно.

## 4.3 Вывод по разделу

В данном разделе отражён процесс проверки разработанного мобильного приложения магазина спортивной обуви вручную. Тестирование такого приложения занимает значимое место в процессе разработки, поскольку именно на этом этапе удаётся обнаружить ошибки, недоработки и проблемы, способные повлиять на стабильность и качество пользовательского опыта.

В рамках проверки были проведены различные виды тестирования: оценивалась функциональность, корректность работы пользовательского интерфейса, производительность, безопасность и совместимость приложения. Каждый из перечисленных аспектов способствует формированию целостной картины работоспособности и эффективности мобильного продукта.

Для имитации реального использования приложения моделировалось типичное поведение пользователя, что позволило охватить все ключевые функции: навигация между разделами, оформление и управление заказами. Отдельное внимание уделялось юзабилити интерфейса, чтобы убедиться в его понятности и комфорте для конечного пользователя.

После завершения основной части работы над приложением была проведена комплексная проверка его функционирования. При этом ещё на этапе создания отдельных компонентов осуществлялись регулярные проверки их работы и взаимодействия между собой. Все обнаруженные технические и логические ошибки были своевременно устранены. Проведённое тестирование показало, что приложение демонстрирует стабильную и корректную работу в различных сценариях и условиях.

В результате тестирования подтверждено, что мобильное приложение магазина спортивной обуви соответствует предъявляемым требованиям и обеспечивает удобство использования для конечных пользователей. Постоянная проверка и внесение нужных доработок способствуют созданию надёжного, безопасного и функционального продукта, отвечающего ожиданиям заказчика.

# 5 Руководство пользователя

ФИО

Подпись

Дата

Лист

1

ДП 05.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

6 Технико-Экономическое обоснование проекта

Лит.

Листов

11

БГТУ 1-98 01 03, 2025

В данном разделе будет представлено руководство пользователя.

Руководство пользо

## 5.1 Общая характеристика разрабатываемого программного средства

## 6.2 Исходные данные для проведения расчётов

# 6 Технико-экономическое обоснование проекта

ФИО

Подпись

Дата

Лист

1

ДП 06.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

6 Технико-Экономическое обоснование проекта

Лит.

Листов

11

БГТУ 1-98 01 03, 2025

## 6.1 Общая характеристика разрабатываемого программного средства

В данном разделе внимание уделяется экономической стороне создания мобильного приложения для магазина спортивной обуви. В дипломной ставится задача показать, насколько оправдана разработка такого продукта и какие затраты потребуются на каждом этапе работы, включая расчёт себестоимости.

Объект дипломной — мобильное приложение, созданное специально для магазина спортивной обуви. Главное, что закладывается в идею приложения, — сделать процесс выбора и покупки кроссовок и другой спортивной обуви максимально простым и удобным для покупателей. Пользователь может спокойно листать каталог, подбирать обувь по нужным параметрам, добавлять понравившиеся модели в корзину, оформлять заказы и отслеживать их статус. Для сотрудников магазина приложение становится инструментом, с помощью которого можно оперативно обновлять ассортимент и работать с заказами.

Способ монетизации – продажа разработанного программного средства и прав на него заказчику. В разделе подробно рассматриваются все расходы, связанные с разработкой приложения для магазина спортивной обуви.

## 6.2 Исходные данные для проведения расчётов

Разработанное мобильное приложение нельзя считать абсолютно новым продуктом, так как на рынке уже существует множество решений для магазинов спортивной обуви. По уровню сложности данный проект относится ко второй категории программного обеспечения. Расчеты выполнены на основе действующих в данной сфере законов и нормативных актов. Все основные исходные данные, необходимые для определения стоимости разработки данного приложения, сведены в таблицу 6.1.

Таблица 6.1 – Исходные данные для расчёта

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование показателя | Единица измерения | Условные обозначения | Норматив |
| 1 | 2 | 3 | 4 |
| Численность разработчиков | чел | Чр | 1 |
| Норматив дополнительной заработной платы | % | Нд.з. | 15 |
| Ставка отчислений в Фонд социальной защиты населения | % | Нф.с.з.н. | 34,6 |
| Стоимость одного машино-часа | % | См.ч. | 0,05 |
| Ставка отчислений по обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний | % | Нб.г.с. | 0,2 |

Продолжение таблицы 6.1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Норматив прочих затрат | % | Нп.з. | 20 |
| Норматив общепроизводственных и общехозяйственных расходов | % | Нобп, обх | 100 |
| Норматив расходов на сопровождение и адаптацию | % | Нр.с.а. | 10 |
| Уровень рентабельности | % | Урент. | 30 |
| Ставка НДС | % | Нндс. | 20 |
| Коэффициент изменения скорости обработки информации | ед. | Кск. | 0,7 |
| Тарифная ставка I разряда в организации за месяц | руб. | ОТI | 780 |
| Тарифный коэффициент | ед. | Кт. | 3,04 |
| Коэффициент естественных потерь рабочего времени | ед. | Кп. | 1,2 |
| Коэффициент премирования | ед. | Кпр. | 1,3 |
| Норматив расхода машинного времени за отладку 100 строк исходного кода | маш. часов | Нм.в. | 12 |

Далее эти данные понадобятся для расчёта всех необходимых параметров в процессе экономического анализа приложения.

## 6.3 Методика обоснования цены

В условиях современной рыночной экономики программное обеспечение (ПО) выступает как продукт организаций, представляющий собой функционально завершённое решение, обладающее товарным видом и реализуемое покупателям по рыночным ценам. Все завершённые разработки относятся к категории научно-технической продукции.

Активное использование вычислительных технологий требует постоянного обновления и совершенствования ПО. Выбор наиболее эффективных проектов ПО связан с их экономической оценкой и расчётом ожидаемого экономического эффекта, который можно определить как для разработчика, так и для конечного пользователя.

Для разработчика экономический эффект выражается в виде чистой прибыли от продажи ПО, остающейся в распоряжении компании. Для пользователя же эффект проявляется в сокращении трудовых, материальных и финансовых затрат за счёт:

* снижения трудоёмкости расчётов, упрощения алгоритмизации программирования и ускорения отладки программ;
* сокращения расходов на оплату машинного времени, использования других ресурсов при отладке программных средств;
* снижения расходов на материалы;
* ускорение ввода в эксплуатацию новых систем;
* улучшения показателей основной деятельности в результате использования передовых программных средств.

Стоимостная оценка веб-приложения у разработчиков предполагает определение затрат, что включает следующие статьи:

* заработная плата исполнителей – основная и дополнительная;
* отчисления в Фонд социальной защиты населения и обязательному страхованию от несчастных случаев на производстве и профессиональных заболеваний;
* расходы на материалы и комплектующие;
* расходы на спецоборудование;
* расходы на оплату машинного времени;
* общепроизводственные и общехозяйственные расходы;
* уменьшения расходов, связанных с установкой, адаптацией и реализацией (реализационные или коммерческие расходы).

На основании затрат рассчитывается себестоимость и отпускная цена разработанного веб-приложения.

### 6.3.1 Определение объёма программного средства

Прежде всего, необходимо определить время, затраченное на разработку данного программного обеспечения. В таблице 6.2 приведены данные о затратах рабочего времени на создание программного продукта.

Таблица 6.2 – Затраты рабочего времени на разработку ПС

|  |  |
| --- | --- |
| Содержание работ | Затраты рабочего времени, дней |
| Рассмотрение аналогов | 1 |
| Проектирование базы данных | 2 |
| Проектирование серверной части приложения | 2 |
| Проектирование клиентской части приложения | 6 |
| Разработка серверной части приложения | 15 |
| Разработка клиентской части приложения | 42 |
| Тестирование приложения | 5 |
| Всего | 74 |

Самыми трудоёмкими этапами разработки, потребовавшими наибольших затрат рабочего времени, стали создание серверной и клиентской частей приложения. На их разработку в общей сложности было потрачено 57 дней, при этом клиентская часть включала работу по вёрстке всего приложения. В целом на выполнение всех этапов разработки ушло 74 дня.

### 6.3.2 Основная заработная плата

Для определения величины основной заработной платы было проведено исследование уровня заработных плат для специалистов, работающих с фреймворком Flutter и языком программирования Dart. Источниками данных послужили открытые веб-порталы, профессиональные форумы, отчётные документы, а также сведения о среднем уровне заработной платы в сфере информационных технологий в Республике Беларусь на 2025 год. По результатам анализа установлено, что средняя месячная зарплата для специалистов уровня junior/middle составляет 2000 рублей.

Основная заработная плата рассчитывается по формуле 6.1.

|  |  |
| --- | --- |
| Соз = Tраз ⋅ Краз⋅ Сзп, | (6.1) |

где Соз– основная заработная плата, руб.;

Траз – время разработки, месяцев;

Краз – количество разработчиков, человек;

Сзп – средняя месячная заработная плата.

Время разработки будет рассчитываться по формуле 6.2.

|  |  |
| --- | --- |
| Траз = Драз / 21, | (6.2) |

где Траз – время разработки, месяцев;

Драз – количество дней разработки, дней.

Проект разрабатывался одним человеком на протяжении трёх месяцев. Расчеты основной заработной платы и времени разработки представлены ниже:

Траз = 74 / 21 = 3,52 месяцев.

Соз = 3,52 ⋅ 1 ⋅ 2000 = 7040 руб.

В дальнейшем для других расчётов используется основная заработная плата, рассчитанная по указанной выше методике.

где Кпр – коэффициент премий и иных стимулирующих выплат (1,5);

*n* – категории исполнителей, занятых разработкой;

Зч*i* – часовой оклад исполнителя i-й категории, руб.;

### 6.3.3 Дополнительная заработная плата

Законодательство о труде предусматривает наличие выплат, определяющиеся по нормативу в процентах к основной заработной плате по формуле 6.3.

, (6.3)

где Соз – основная заработная плата, руб.;

Ндз – норматив дополнительной заработной платы, %.

Исходя из основной заработной платы, а также норматива дополнительной заработной платы, можно рассчитать сумму дополнительной заработной платы:

 руб.

Дополнительная заработная плата включает выплаты, предусмотренные законодательством о труде, и определяется по нормативу в процентах по отношению к основной заработной плате.

### 6.3.4 Расчёт отчислений на социальные цели

Отчисления в Фонд социальной защиты населения (ФСЗН) определяются в соответствии с действующими законодательными актами по нормативу в процентном отношении к фонду основной и дополнительной зарплаты исполнителей и вычисляются по формуле 6.4.

|  |  |
| --- | --- |
|  | (6.4) |

где Соз – основная заработная плата, руб.;

Сдз – дополнительная заработная плата, руб.;

Нфсзн – норматив отчислений в Фонд социальной защиты населения, %.

Исходя из полученных данных высчитаем отчисления в Фонд социальной защиты наcеления (ФСЗН):

801,216 руб.

Полученные выше значения будут использованы при подсчёте заработной платы программиста с отчислениями.

### 6.3.5 Расходы на материалы

Сумма расходов на материалы СМ определяется как итог таблицы 6.3, в которой собраны реально понесённые затраты на разработку программного средства

Таблица 6.3 – Материальные затраты на разработку ПС

|  |  |
| --- | --- |
| Вид материальных затрат | Фактические затраты, руб. |
| 1. Бумага, 1 пачка | 14,69 |
| 2. Картридж, 1 шт. | 40,20 |
| Всего | 54,89 |

Сумма расходов на материалы составляет 54,89 руб.

### 6.3.6 Расходы на специальное оборудование и платные услуги

Затраты на специальное оборудование и платные услуги, необходимые для разработки и отладки веб-приложения, оказались минимальными благодаря эффективному использованию бесплатных плагинов и библиотек. В ходе работы были выбраны надёжные и функциональные инструменты, предоставленные сообществом разработчиков на безвозмездной основе. К тому же всё необходимое оборудование для работы над проектом уже находилось в наличии, что исключило необходимость дополнительных закупок. Таким образом, общие расходы на специальное оборудование и платные услуги составили 0 рублей, что означает, что ССОПУ = 0 рублей.

### 6.3.7 Расчёт прочих прямых затрат

Расходы на конкретное программное средство Спз включают затраты на приобретение и подготовку специальной научно-технической информации и специальной литературы и рассчитываются по формуле 6.5.

|  |  |
| --- | --- |
|  | (6.5) |

где Нпз – норматив прочих затрат в целом по организации, %.

Норматив прочих затрат равен 20% – по исходным данным, приведённым в таблице 6.1.

= 7040 ⋅ 20/100 = 1408 руб.

Таким образом, прочие прямые затраты при разработке веб-приложенияравны 1408 рублей.

### 6.3.8 Расчёт общепроизводственных и общехозяйственных расходов

Сумма общепроизводственных, общехозяйственных расходов Собп,обх  – произведение основной заработной платы исполнителей на конкретное программное средство Соз на норматив накладных расходов в целом по организации Нобп,обх по формуле 6.6.

|  |  |  |
| --- | --- | --- |
|  |  | (6.6) |

Норматив накладных расходов по организации равен 100% – по исходным данным, приведённым в таблице 6.1.

Все данные необходимые для вычисления есть, поэтому можно определить сумму накладных расходов:

Собп,обх = ⋅ 100 / 100 = 7040 руб.

Получим, что накладные расходы, необходимые для разработки веб-приложения,равны 7040 рублей.

### 6.3.9 Расчёт суммы расходов на разработку программного средства

Сумма расходов на разработку программного средства Ср определяется как сумма основной и дополнительной заработных плат исполнителей на конкретное программное средство, отчислений на социальные нужды, расходов на материалы, расходов на оплату машинного времени, суммы прочих затрат и суммы накладных расходов по формуле 6.7.

|  |  |
| --- | --- |
| Ср = Соз + Сдз + Сфсзн + См + ССОПУ + Спз + Собп,обх | (6.7) |

Ср = 7040 + ++ 54,89 + 0 + 1408 + = 19400,11 руб.

Таким образом, получим сумму всех расходов на разработку веб-приложенияравной 19400,11 рублей.

### 6.3.10 Расходы на сопровождение и адаптацию

Сумма расходов на сопровождение и адаптацию программного средства Срса определяется как произведение суммы расходов на разработки на норматив расходов на сопровождение и адаптацию Нрса. Рассчитывается по формуле 6.8.

|  |  |  |
| --- | --- | --- |
|  |  | (6.8) |

где Ср.с.а – сумма расходов на сопровождение и адаптацию ПС, руб.;

Ср – общая сумма расходов на разработку ПС, руб.;

Нр.с.а – норматив расходов на сопровождение и адаптацию, %.

Основываясь на исходные данные, расположенные в таблице 6.1 и формулу 6.9, норматив расходов на сопровождение и адаптацию Нрса равен 10%. Сумма расходов на сопровождение и адаптацию ПС составляет:

руб.

Получим, что расходы, необходимые на сопровождение и адаптацию веб-приложения,равны рубля.

### 6.3.11 Расчет общей суммы расходов

Общая сумма расходов на разработку с затратами на сопровождение и адаптацию Сп определяется по формуле 6.9.

|  |  |  |
| --- | --- | --- |
|  |  | (6.9) |

Сп = + 940,01 = 21 340,12 руб.

Получим, что общая сумма расходов веб-приложения, выполняющего роль книжного магазина, равна 21 340,12 рубля.

### 6.3.12 Определение цены, оценка эффективности

Для оценки отпускной цены разработки необходимо рассмотреть цены разработки приложений, обладающих аналогичным функционалом. Для этого воспользуемся помощью трех сайтов-агрегаторов услуг по разработке программного обеспечения.

Первым ресурсом является сайт estimatemyapp.com. Выбрав все необходимые параметры, выходит, что на разработку схожего ресурса в среднем должно уйти 70 дней разработки, а также стоимость в 31500 $, что составит 94 613 руб (если брать по курсу 1$ = 3,0 руб). Наглядная демонстрация результатов работы ресурса представлена на рисунке 6.1.

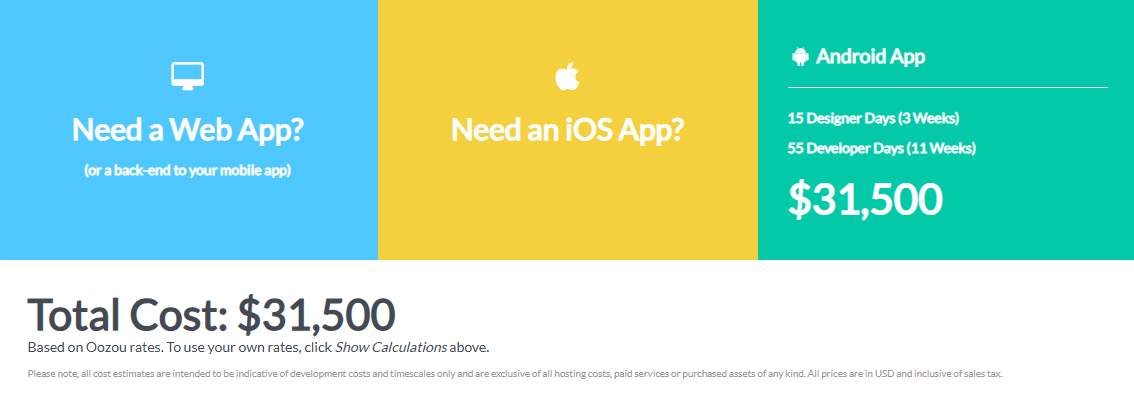


Рисунок 6.1 – Результат работы сайта estimatemyapp.com

Вторым ресурсом является сайт thebestapp.ru. Этот ресурс работает по аналогии с предыдущим, за исключением того, что итоговая цена будет представлена в российских рублях. На рисунке 6.2 представлен результат работы сайта-агрегатора:

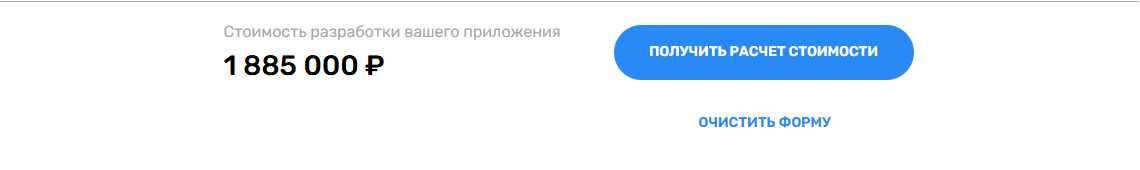


Рисунок 6.2 – Результат работы сайта thebestapp.ru

Как видно из результатов, примерная сумма похожего ресурса обойдется в 1 885 000 рублей, что в пересчёте на белорусские рубли составляет 69 843,02 рублей.

Последним ресурсом является сайт stfalcon.com. Выбрав все необходимые компоненты системы, приблизительная стоимость проекта составляет 16 343$, что в пересчёте составляет 49 087,83 белорусских рублей. Ниже на рисунке 6.3 приведён результат работы сайта.

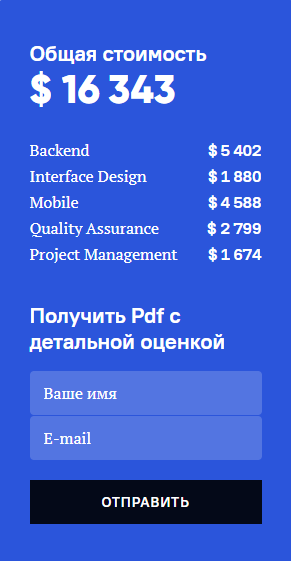


Рисунок 6.3 – Результат работы сайта stfalcon.com

Как видно из результатов, отпускная цена на сайтах-агрегаторах услуг по разработке программного обеспечения отличается друг от друга.

Средняя цена разработки приложения с аналогичным функционалом рассчитывается как среднее арифметическое указанных выше цен:

ЦАср = (94 613 + 69 843,02 + 49 087,83 ) / 3 = 71 181,28 руб.

Минимальная цена разработки приложения с аналогичным функционалом составляет 49 087,83 руб.

Т. к. полная себестоимость разработки и поддержки ПС составляет 21 340,12 руб., есть возможность установить цену соответствующую самой низкой представленной цене. Установим Цс НДС = 49 087,83 руб.

Цена ПС без НДС рассчитывается:

Цбез НДС = 49 087,83 \* (100 / 120) = 40 906,53 руб.

Прибыль от реализации ПС рассчитывается по формуле 6.10.

|  |  |  |
| --- | --- | --- |
|  | Прп = Цбез НДС - Сп | (6.10) |

Прп = 40 906,53 – 21 340,12 = 19 566,41 руб.

Рентабельность ПС рассчитывается по формуле 6.11.

|  |  |  |
| --- | --- | --- |
|  | РПС = | (6.11) |

РПС = ( 19 566,41 / 21 340,12 ) \* 100 = 91,69 %.

## 6.4 Вывод по разделу

В таблице 6.4 представлены результаты расчётов для основных показателей данного раздела.

Таблица 6.4 – Результаты расчётов

|  |  |
| --- | --- |
| Наименование показателя | Значение |
| Время разработки, мес. | 3,52 |
| Количество программистов, чел. | 1 |
| Основная заработная плата, руб. | 7040 |
| Дополнительная заработная плата, руб. | 1056 |
| Сумма отчислений на социальные цели, руб. | 801,216 |
| Сумма расходов на материалы, руб. | 54,89 |
| Расходы на специальное оборудование и платные услуги, руб. | 0 |
| Прочие прямые затраты, руб. | 1408 |
| Общепроизводственные и общехозяйственные расходы, руб. | 7040 |
| Сумма расходов на разработку программного средства, руб. | 19400,11 |
| Расходы на сопровождение и адаптацию ПС, руб. |  |
| Общая сумма расходов, руб. | 21340,12 |
| Отпускная цена с НДС, руб. | 49 087,83 |
| Цена без НДС | 40 906,53 |
| Прибыль от реализации, руб. | 19 566,41 |
| Рентабельность ПС, % | 91,69 |

Исходя из совокупности вышеупомянутых факторов, можно сделать вывод, что разработка данного мобильного приложения приносит положительный экономический эффект.

Необходимость создания мобильного приложения для продажи спортивной обуви продиктована стремлением обеспечить удобный и эффективный доступ к широкому ассортименту товаров. Это приложение является важной частью управления каталогом обуви и повышения качества обслуживания как для клиентов, так и для продавцов.

Разработка мобильного приложения была направлена на увеличение доступности и улучшение качества обслуживания в сфере торговли спортивной обувью. Использование приложения позволяет пользователям легко находить нужную обувь, изучать её характеристики, оставлять отзывы и рейтинги, а также быстро оформлять заказы и отслеживать их статус. Всё это способствует росту удовлетворённости клиентов и улучшению общего качества обслуживания.

Создание данного программного обеспечения, выполненное одним разработчиком за три месяца, при заданных условиях обойдётся компании в 21340,12 руб. Реализация данного продукта принесёт компании прибыль в размере 19 566,41 руб. Установленная цена на разработанное приложение соответствует нижнему диапазону рыночной стоимости для продуктов с аналогичным функционалом.

# 7 Анализ информационной безопасности приложения

ФИО

Подпись

Дата

Лист

1

ДП 07.00.ПЗ

Разраб.

*Ероховец И.А.*

Пров.

*Уласевич Н.И.*

Н. контр.

Николайчук А.Н.

Утв.

Блинова Е.А.

7 Анализ информационной безопасности приложения

Лит.

Листов

2

БГТУ 1-98 01 03, 2025

Важным аспектом при разработке мобильного приложения выступает обеспечение информационной безопасности. Приложение должно защищать данные пользователей от несанкционированного доступа, утечек и взлома. Надёжная безопасность формирует доверие к сервису и позволяет предотвратить большинство рисков, связанных с хранением и обработкой персональной информации.

В рамках анализа безопасности приложения были рассмотрены следующие ключевые направления:

* защита пользовательских данных: реализация мер по защите аккаунтов, личной информации, истории заказов;
* разграничение прав пользователей: внедрение ролевой модели доступа к функциям приложения;
* регулярное обновление и контроль актуальности данных: обеспечение целостности информации на клиенте и сервере.

## 7.1 Защита пользовательских данных

Для хранения и передачи конфиденциальных данных пользователей применяется облачный сервис Firebase, отличающийся высоким уровнем безопасности и современными средствами шифрования.

Пароли и аутентификационные данные никогда не сохраняются в открытом виде – для хранения и проверки используется сервис Firebase Auth, где все пароли проходят хэширование с использованием современных криптоалгоритмов.  
Данные профиля, история заказов, бонусные баллы хранятся в базе данных Firestore, где доступ возможен только для авторизованных пользователей через защищённые каналы передачи данных, протокол HTTPS и встроенное шифрование на стороне Firebase.

Для дополнительной защиты реализованы следующие меры:

* возможность восстановления доступа только по подтверждённому email;
* автоматическая проверка безопасности паролей при регистрации.

Вся передача данных между приложением и сервером осуществляется исключительно по защищённому протоколу HTTPS, что исключает возможность перехвата данных злоумышленниками.

Файлы и изображения пользователей загружаются во внешнее облачное хранилище с использованием временных защищённых токенов доступа.

Также в приложении реализована система автоматического обновления токенов доступа и регулярная сверка состояния пользователя с базой данных, которые обновляются после истечения срока годности.

## 7.2 Ограничение прав доступа пользователей

Для ограничения прав доступа пользователей реализована ролевая модель (RBAC – Role-Based Access Control). В приложении предусмотрены следующие основные роли:

* **пользователь** имеет доступ к поиску, покупкам, просмотру и редактированию личных данных, отзывам;
* **поставщик** имеет доступ к добавлению и редактированию собственных товаров;
* **администратор** имеет расширенные функции: модерация отзывов, управление заявками поставщиков, добавление новых категорий и адресов;
* **продавец** получает доступ к изменению статуса товара.

Управление доступом реализовано на двух уровнях:

* на клиенте скрываются элементы интерфейса, не предназначенные для текущей роли;
* на сервере и в Firebase Security Rules проводится дополнительная проверка прав при попытке выполнения чувствительных операций (например, добавление товара или модерация заявок).

Это позволяет минимизировать риски несанкционированного доступа к административным функциям и личным данным других пользователей.

## 7.3 Выводы по разделу

Благодаря внедрению современных средств информационной безопасности приложение практически не подвержено основным угрозам и будет защищать персональные данные пользователей на всех этапах работы. Использование Firebase обеспечило шифрование информации, надежную аутентификацию и очень строгий контроль доступа к данным. Для учетных записей реализовано хеширование паролей и восстановление доступа только через подтвержденный email. История заказов и персональные данные доступны только после авторизации, также блокируются попытки несанкционированного доступа как на уровне серверных правил, так и в клиентском интерфейсе.

Благодаря реализованной ролевой модели доступ к функциям строго ограничен: пользователь может просматривать и редактировать только свои данные, административные права назначаются отдельным ролям и проверяются также на стороне сервера.

Реализованные комплексные меры предоставляют прозрачное управление правами, усиленную защиту данных и соблюдение актуальных стандартов безопасности для мобильных приложений. Такой подход способствует укреплению доверия пользователей и гарантирует безопасность их информации независимо от сценария использования.

# Список использованных источников

1. Главная страница сайта Lamoda [Электронный ресурс] / lamoda.ru– Режим доступа https://www.lamoda.ru/men-home/?sitelink=topmenuM – Дата доступа: 30.03.2025.
2. Главная страница сайта Mark Formelle [Электронный ресурс] / markformelle.by – Режим доступа https://markformelle.by/magaziny/– Дата доступа: 31.03.2025.
3. Главная страница сайта Ostin [Электронный ресурс] / ostin.com – Режим доступа: https://ostin.com/?utm\_referrer=https%3A%2F%2Fwww.google.com%2F/– Дата доступа: 25.03.2025.
4. Главная страница сайта Android Studio [Электронный ресурс] / developer.android.com – Режим доступа https://developer.android.com/studio/intro – Дата доступа: 15.03.2025.
5. Android documentation for app developers. The activity lifecycle [Электронный ресурс] / developer.android.com – Режим доступа: <https://developer.android.com/guide/components/activities/activity-lifecycle> – Дата доступа: 11.03.2025.
6. Введение в язык Dart [Электронный ресурс] / Metanit – 2025. – Режим доступа: https://metanit.com/dart/tutorial/1.1.php – Дата доступа: 20.03.2025.

Приложение А

