МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет компьютерных наук

Кафедра программирования и информационных технологий

Продовольственный онлайн-супермаркет с доставкой на дом

Курсовой проект
09.03.04 Программная инженерия
Информационные системы и сетевые технологии

Допущено к защите в ГЭК __.__.2023

Зав. Кафедрой_____С.Д. Махортов, д. ф.-м. н., профессор

Обучающийся_____Н.В. Морозов*, 3 курс, д/о*

Обучающаяся_____С.А. Черникова*, 3 курс, д/о*

Обучающийся_____Р.Т.Халилов*, 3 курс, д/о*

Обучающийся_____Ж. Флорексил*, 3 курс, д/о*

Руководитель_____В.С. Тарасов, ст. преподаватель

Воронеж 2023

Содержание

Введение

Последние несколько лет онлайн-ритейл в России стремительно развивается.

Исследование потребительского поведения в ритейл-индустрии проводилось в августе 2021 года в формате панельного онлайн-опроса. Всего было опрошено 300 человек в возрасте от 18 до 55 лет из разных городов России с населением от 500 тысяч жителей.

Обратимся к факторам, обеспечивающим привлекательность приобретения продуктов в онлайне и офлайне. В числе бесспорных преимуществ онлайн-покупок участники опроса отметили безопасность совершения покупок на фоне пандемии. Отсутствие рисков заболеть стало особенно важным для покупателей, поэтому многие сервисы онлайн-покупки продуктов начали практиковать бесконтактную доставку до двери. Так, возможные последствия для здоровья покупателей снижаются практически до нуля (что положительно оценили 38% участников опроса), чем не могут похвастаться физические магазины.

Другая причина выбора e-grocery среди опрошенных людей — онлайн-покупки существенно экономят время покупателей. Это выражается хотя бы в отсутствии необходимости ожидания своей очереди на кассе. Помимо временных затрат, сокращаются и физические усилия, которые нужно приложить во время похода в магазин: не нужно складывать продукты и тащить пакет до дома. Достаточно сделать несколько движений рукой по экрану телефона и встретить курьера.

Так же участники опроса отметили простоту сравнения цен на продукты в разных сервисах доставок: для этого не нужно перемещаться между разными магазинами, а достаточно скачать несколько приложений, чтобы выбрать самый выгодный товар. В них же можно ознакомиться с актуальными акциями. Удобство такого устройства отметило 73% опрошенных.

Интернет диктует законы среде информационного обмена. В эру цифровизации и тотального распространения смартфонов знакомство с ассортиментом магазина и даже заказ товаров через мобильное приложение становится все более актуальным: об этом свидетельствует 41% анкетированных участников.

Почему люди положительно оценивают влияние мобильных приложений магазинов на свой привычный процесс шопинга? Клиентов привлекает система лояльности многих торговых сетей, доставка и удобство выбора продуктов. Некоторые из приложений даже предугадывают желания клиента и запоминают типичный набор его покупок. К тому же, в приложение с продуманным интерфейсом может быть просто интересно заходить время от времени, обновляя каталог в поисках новинок.

Целью данной работы является разработка мобильного приложения, а именно продовольственного онлайн-магазина под Android с возможностью как самовывоза из ближайших точек, так и доставки курьером. Особенностью данного приложения будет наличие внутренней карты магазина и наличие шагомера, обе доступны при выборе самовывоза. Первая функция поможет покупателю быстро найти в супермаркете товары, выбранные им для покупки. Вторая функция посчитает количество шагов, проделанных пользователем от его местоположения до магазина, и, исходя из этого количества, покупатель получит скидку при оплате продуктов на кассе.

1 Постановка задачи

1.1 Цели создания приложения

Целями создания мобильного приложения «The Shop» являются:

— Продажа продуктов питания;

— Сообщение пользователю актуальной информации.

1.2 Задачи приложения

Разрабатываемый проект должен решать следующие задачи:

— Просмотр каталога товаров. В нем представлены продукты, продаваемые заказчиком;

— Просмотр витрины. В ней размещается любая информация, которую необходимо разместить магазину: акции, советы, ссылки на внешние источники и т.п.;

— Заказ товара, отслеживание статуса заказа;

— Выбор способа оплаты и доставки;

— Администрация приложения.

1.3 Требования к разрабатываемой системе

1.3.1 Функциональные требования

Зарегистрированный покупатель обладает следующими возможностями:

— Определение магазина, из которого будет совершаться самовывоз заказа;

— Редактирование персональных данных (например, изменение ближайшего магазина для самовывоза);

— Выбор способа получения товаров (самовывоз или доставка курьером);

— Изменение содержимого корзины (добавление/удаление продуктов);

— Добавление товаров в избранное;

— Онлайн-оплата покупок;

— Авторизация;

— Использование шагомера;

— Использование внутренней карты магазина.

Незарегистрированный покупатель обладает следующими возможностями:

— Определение магазина, из которого будет совершаться самовывоз заказа;

— Способ получения товаров (самовывоз или доставка курьером);

— Изменение содержимого корзины (добавление/удаление продуктов);

— Онлайн-оплата покупки. При выборе самовывоза можно оплатить заказ только онлайн (для оплаты наличными на кассе нужно зарегистрироваться в приложении);

— Регистрация.

Сотрудник магазина (администратор) обладает следующими возможностями:

— Редактирование информации об акциях и выгодных предложениях магазина (например, добавить новую акцию);

— Получение API ключа;

— Обновление feed.

1.4 Требования к интерфейсу

Оформление и верстка экранов приложения должны соответствовать следующим требованиям:

— Все экраны приложения должны быть оформлены в едином стиле;

— Все экраны приложения должны быть оформлены в соответствии с принципами “Material Design”;

— Дизайн приложения должен быть адаптирован для корректного отображения при различных размерах экрана;

— Дизайн приложения должен поддерживать портретную ориентацию экрана.

1.5 Задачи, решаемые в процессе разработки

Были поставлены следующие задачи:

— Анализ предметной области;

— Выбор языка программирования для написания back-enda и front-enda;

— Написание технического задания в соответствии с ГОСТ 34.201-20;

— Написание курсовой работы по проекту;

— Анализ аналогов;

— Разработка дизайна приложения;

— Создание макетов дизайна приложения;

— Разработка основных функций приложения;

— Разработка особенностей приложения, которые бы выделяли его на рынке среди аналогов;

— Создание диаграмм: прецендентов, классов, активностей, последовательностей, развертывания, сотрудничества, объектов, состояний;

— Создание доски заданий и репозитория на GitHub;

— Реализация основных функций приложения с использованием REST API;

— Реализация интерфейса приложения с использованием Flutter;

— Описание процесса разработки и результата.

2 Анализ предметной области

2.1 Терминология (глоссарий) предметной области

Фреймворк – готовый набор инструментов, который помогает разработчику быстро создать продукт: сайт, приложение, интернет-магазин;

Ключ API – уникальный идентификатор, который используется для аутентификации запросов, связанных с вашим проектом.

Front-end – клиентская часть приложения. Отвечает за получение информации с программно-аппаратной части и отображение ее на устройстве пользователя. В нашем проекте, это само android приложение;

Back-end – программно-аппаратная часть приложения. Отвечает за функционирование внутренней части приложения;

Rest API – стиль архитектуры программного обеспечения для построения масштабируемых веб-приложений.

Material Design – дизайн-система для создания интерфейсов программного обеспечения и приложений, разработанная компанией Google.

Верстка экрана –корректное отображение интерфейса приложения на любых устройствах путем задания размеров и разрешения экрана для каждого макета.

Json – текстовый формат обмена данными, основанный на JavaScript.

Feed – список всех товаров магазина, данные о расположении магазинов, карта магазина и её разметка, представленные в виде json.

Неактуальная информация – информация, которая на текущий момент времени уже устарела.

2.2 Анализ рынка продуктовых онлайн-магазинов

Анализ рынка продуктовых онлайн-магазинов становится все более актуальным на фоне растущей конкуренции и увеличивающегося количества игроков в этом сегменте. Продуктовые онлайн-магазины имеют некоторые преимущества перед традиционными магазинами, но также сталкиваются с ограничениями и некоторыми минусами.

Одним из главных преимуществ продуктовых онлайн-магазинов является возможность заказа и доставки продуктов питания без необходимости выхода из дома. Это особенно актуально в условиях пандемии, когда многие люди избегают посещения магазинов и предпочитают оставаться дома. Кроме того, такие магазины обычно предлагают более широкий ассортимент товаров, чем традиционные магазины, и часто предлагают товары из региональных производств.

Однако у продуктовых онлайн-магазинов есть и ряд минусов. Одним из них является возможность получения продуктов низкого качества или с истекшим сроком годности. Это может произойти из-за проблем с хранением и транспортировкой продуктов. Кроме того, у некоторых продуктовых онлайн-магазинов могут быть ограничения по срокам доставки в том числе из-за отдаленности региона.

Продуктовые онлайн-магазины также сталкиваются с ограничениями в связи с тем, что рынок достаточно конкурентный. Они вынуждены проявлять творческий подход и усердно работать над предоставлением уникальных услуг своим клиентам. Конкурентной способностью для таких магазинов становится не только сумма цен, но и качество предоставления услуг, удобство работы с сайтом и организация работы курьеров.

Таким образом, продуктовые онлайн-магазины имеют свои преимущества и ограничения, но благодаря необходимости усложнения работы для организации продаж на рынке, такие магазины могут сохранять свою конкурентную способность. В будущем, с улучшением технологий и развитием сетей доставки, продуктовые онлайн-магазины, возможно, станут еще более популярными, для тех кто выбирает домашний комфорт и минимум рисков, связанных с пандемией.

2.3 Обзор аналогов

Этап обзора аналогов является важной частью процесса разработки мобильного приложения. Этот шаг включает в себя изучение приложений, аналогичных разрабатываемому, чтобы лучше понять ожидания

пользователей, отраслевые тенденции и передовые методы разработки мобильных приложений. Этот обзор позволит собрать информацию и идеи, полезные для разработки высококачественного приложения, которое будет соответствовать потребностям целевой аудитории и будет выделяться на рынке.

2.3.1 СберМегаМаркет

Это онлайн-платформа для супермаркетов, запущенная крупнейшим банком России Сбербанком. СберМегаМаркет предлагает широкий ассортимент продуктов питания, товаров для дома, косметических товаров и товаров для домашних животных. Пользователи могут выбрать удобное время доставки, а также отслеживать их доставку в режиме реального времени.

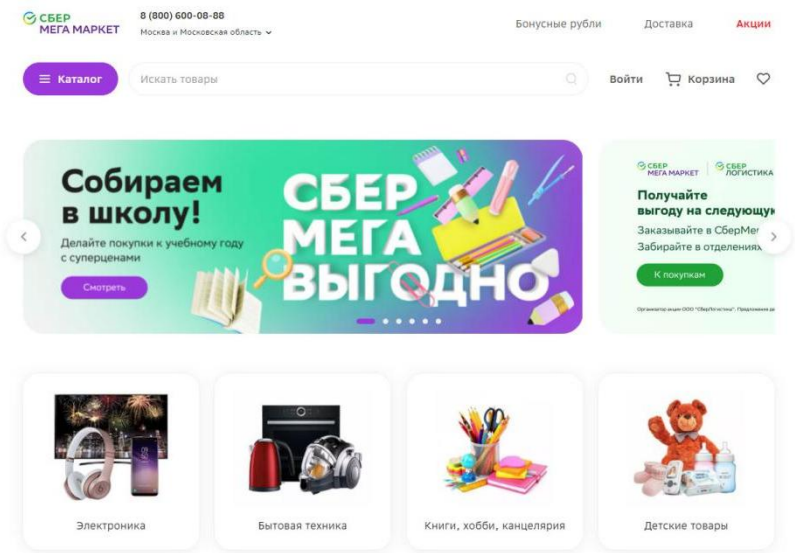

Рисунок 1 - интерфейс СберМегаМаркета

Плюсы:

— Автоматический импорт корзины из предыдущих заказов;

— Возможность отслеживания статуса заказа;

— Различные способы оплаты;

— Возможность заказа через голосовой ассистент;

— Можно использовать бонусы «Спасибо» и вход по «Сбер ID»;

— Бесплатная доставка для подписчиков «СберПрайм».

Минусы:

— Бывают проблемы с доставкой: пользователи или получают заказ частично, или вовсе не получают его;

— Иногда приложение тормозит.

2.3.2 Самокат

Это российская компания по онлайн-доставке продуктов, предлагающая продукты питания, аптечные товары, Товары для дома и туалетные принадлежности. Компания самокат гарантирует доставку за 15 минут и менее, что делает ее одним из самых быстрых вариантов доставки в России.

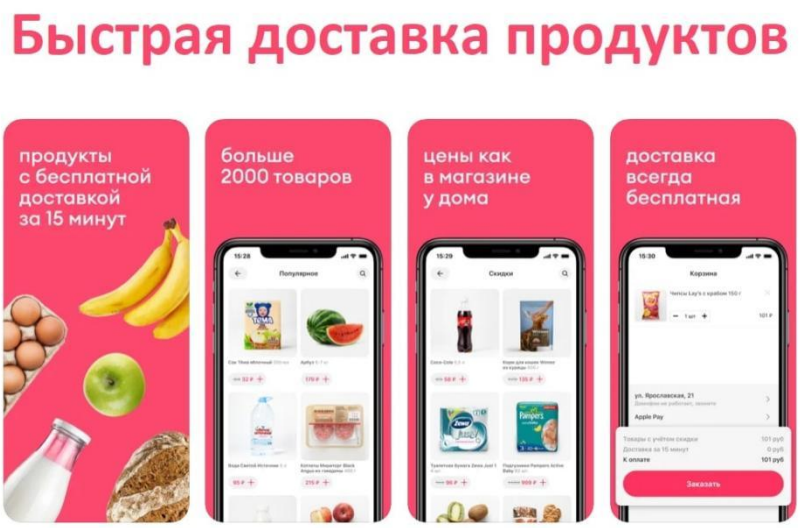

Рисунок 2 - интерфейс Самоката

Плюсы:

— Пользователь может выбрать время доставки;

— Возможность отслеживания статуса заказа в приложении.

Минусы:

— Высокая стоимость доставки;

— Иногда доставляют просроченный товар.

2.3.3 Магнит

Магнит - это сеть супермаркетов в России, предлагающая широкий ассортимент продуктов питания, а также товары для дома и косметики. Помимо своих обычных магазинов, Магнит также предлагает онлайн-платформу супермаркетов с доставкой на дом.

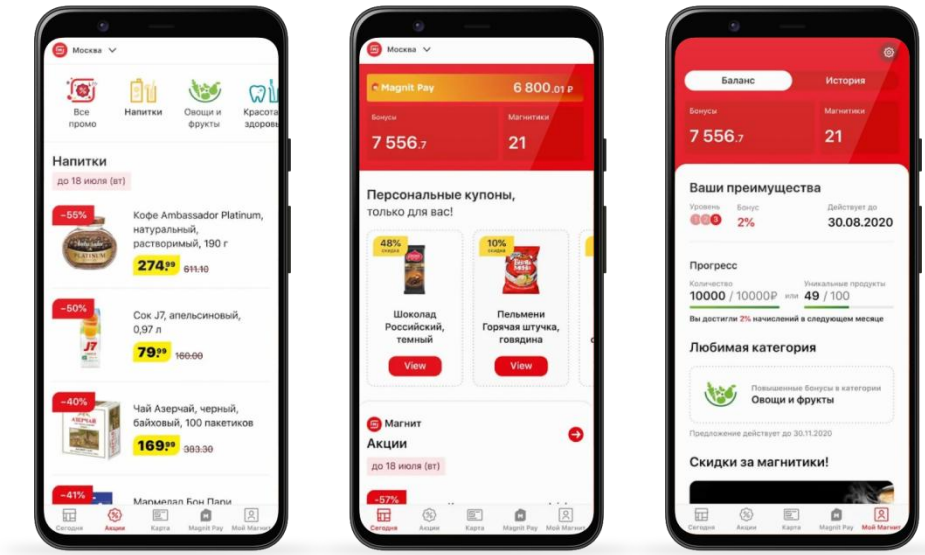

Рисунок 3 - интерфейс Магнита

Плюсы:

— Различные способы оплаты и доставки;

— Автоматический импорт корзины из предыдущих заказов.

Минусы:

— Приложение работает медленно и часто вылетает;

— Неактуальная информация о скидках и бонусах.

2.3.4 Пятерочка

Это еще одна сеть супермаркетов в России, предлагающая широкий ассортимент продуктов питания, а также товары для дома и косметики. Пятерочка, онлайн приложение которой представлено на рисунке 4) также предлагает онлайн-платформу супермаркетов с доставкой на дом.

Плюсы:

— Различные способы оплаты;

— Возможность отслеживания статуса заказа.

Минусы:

— Доставка продуктов занимает больше двух часов;

— Приложение часто зависает.

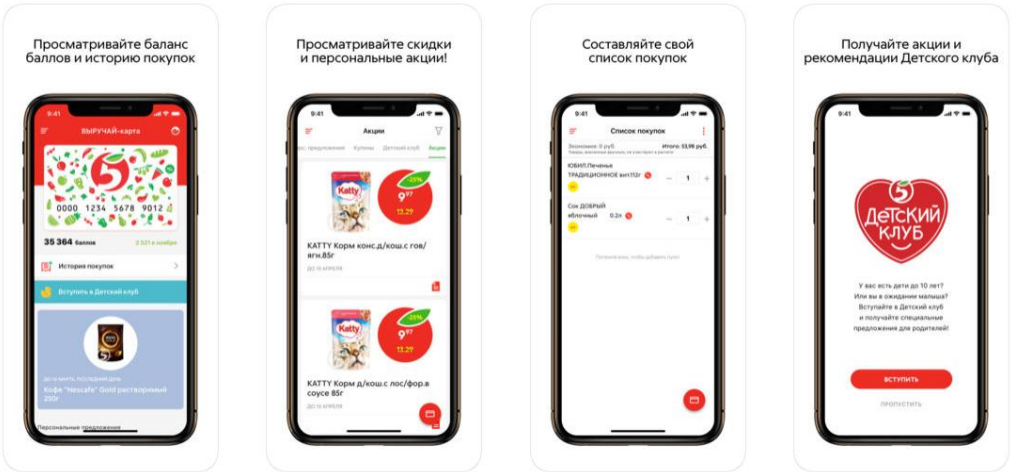

Рисунок 4 - интерфейс Пятерочки

3 Реализация

3.1 Средства реализации

Система должна состоять из сервера приложения, реляционной базы данных, клиентской части.

Основной используемый стек технологий:

Back-end (серверная часть):

— Java 17;

— Spring Framework;

— PostgreSQL, Liquibase;

— Elasticsearch;

— Система сборки Maven.

Язык Java был выбран, так как он не зависит от платформы. Можно создать Java-приложение на Windows, скомпилировать его в байт-код и запустить на любой другой платформе, поддерживающей JVM – виртуальную машину Java.

Основным преимуществом Spring Framework является большое количество реализованных внутренних библиотек, позволяющих быстро и качественно писать код.

В качестве базы данных была выбрана Postgres, т.к. поддерживает пользовательские объекты и их поведение, включая типы данных, функции, операции и другое. Это делает Postgres невероятно гибким и надежным. Среди прочего, он умеет создавать, хранить и извлекать сложные структуры данных.

Для управления базой данных и внесения изменений в нее будет использоваться библиотека Liquibase. Основным преимуществом является поддержка написания миграционных файлов в виде xml файлов.

Elasticsearch - это масштабируемый полнотекстовый поисковый движок с открытым исходным кодом, использующий библиотеку Lucene и написанный на Java. Он предназначен для сложных поисков по базе документов/файлов. Его преимуществом является скорость поиска и быстрая выдача результатов.

Front-end (клиентская часть):

— Flutter;

— Firebase (Crashlytics, Analitycs, Remote Config, A/B Testing, Authentication, Realtime Database);

— Google Maps;

— Elementary (фреймворк).

Главное преимущество Flutter - скорость разработки. Готовые решения данного фреймворка позволяют писать меньше кода, что значительно упрощает процесс создания приложений и существенно экономит время разработчиков.

Firebase разрабатывается Google. Предоставляет мощные базы данных для разработки веб- и мобильных приложений.

Elementary соответствует ряду требований: является комплексным решением и обеспечивает чётко разделённые по ответственностям слои. Они, в свою очередь, максимально изолированы и независимы друг от друга. Решение легко тестируемо. Реализация максимально близка к работе Flutter.

3.2 Диаграмма IDEF0

На рисунке 5 представлена IDEF0 диаграмма, иллюстрирующая процесс оформления и получения заказа.

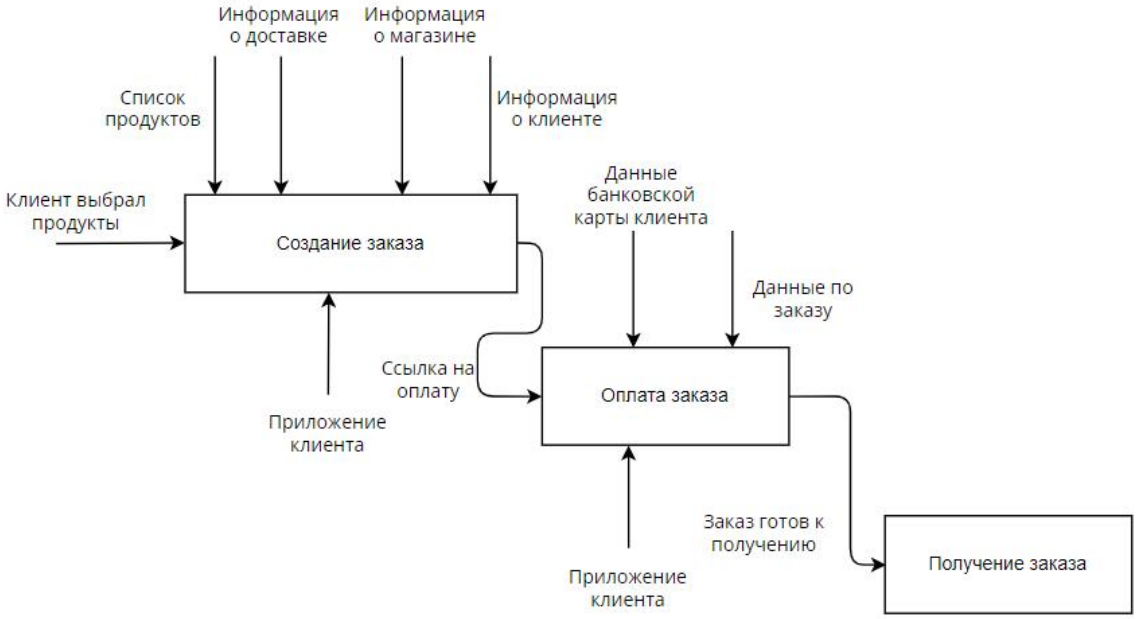

Рисунок 5 - Диаграмма IDEF0

3.3 Диаграмма вариантов использования

3.3.1 Диаграмма пользователя

На рисунке 6 представлена UML диаграмма вариантов использования, иллюстрирующая функционал покупателя.

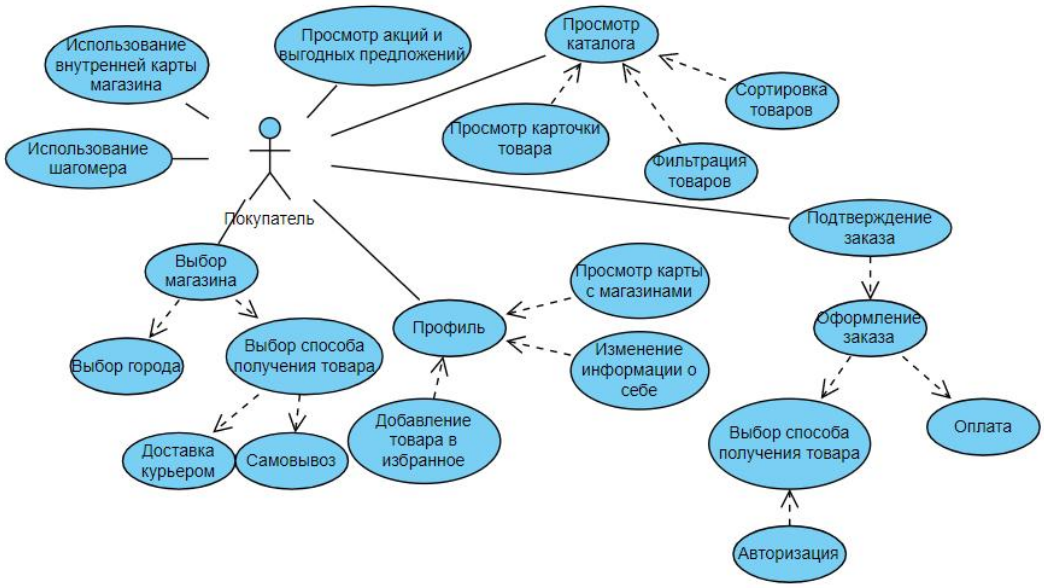

Рисунок 6 - Диаграмма пользователя

3.3.2 Диаграмма администратора

На рисунке 7 представлена UML диаграмма вариантов использования, иллюстрирующая функционал сотрудника магазина.

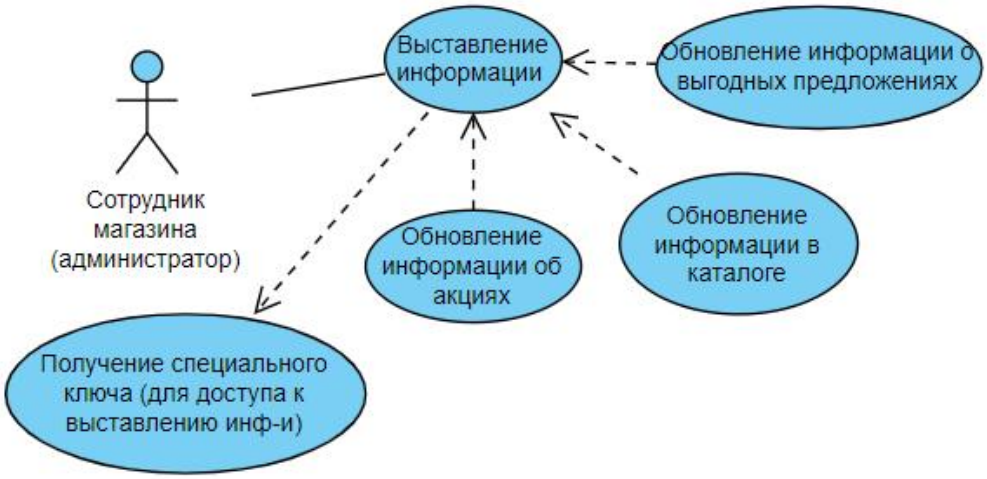

Рисунок 7 - Диаграмма сотрудника магазина

3.4 Диаграмма состояний

На рисунке 8 представлена UML диаграмма состояний, иллюстрирующая процесс создания и оплаты заказа.

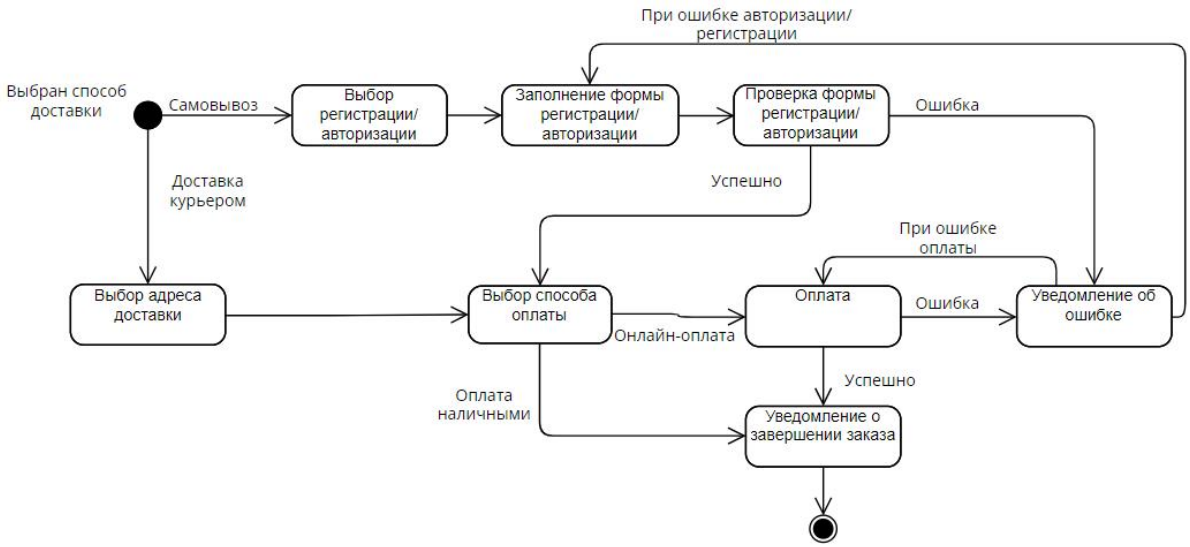

Рисунок 8 - Диаграмма состояний

3.5 Диаграмма классов сущностей

На рисунке 9 представлена UML диаграмма классов сущностей.

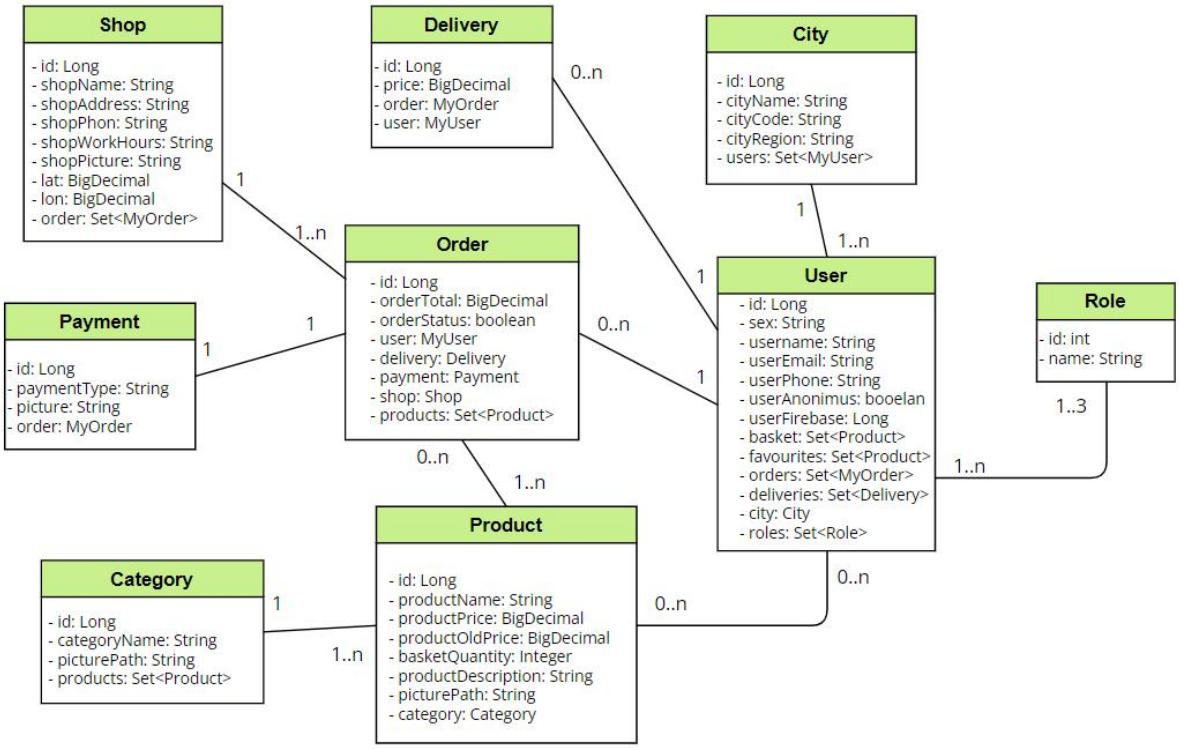

Рисунок 9 - Диаграмма классов сущностей

3.6 Диаграмма объектов

На рисунке 10 представлена UML диаграмма объекта Заказ.

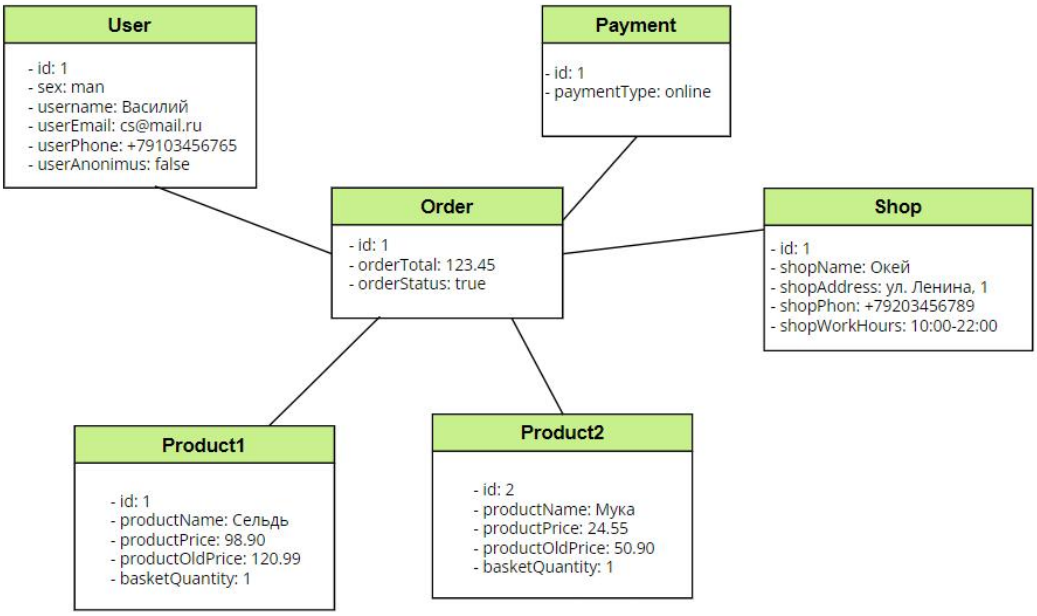

Рисунок 10 - Диаграмма объектов

3.7 Диаграмма развертываний

На рисунке 11 представлена UML диаграмма развертываний приложения.

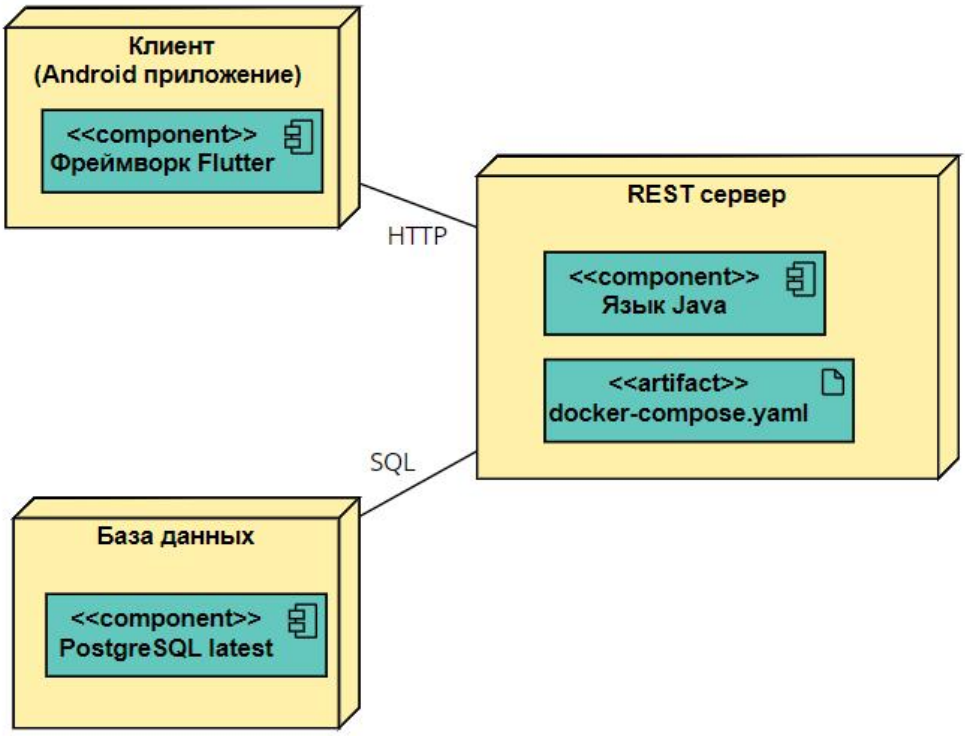

Рисунок 11 - Диаграмма развертываний

3.8 Архитектура приложения

На рисунке 12 представлена архитектура приложения

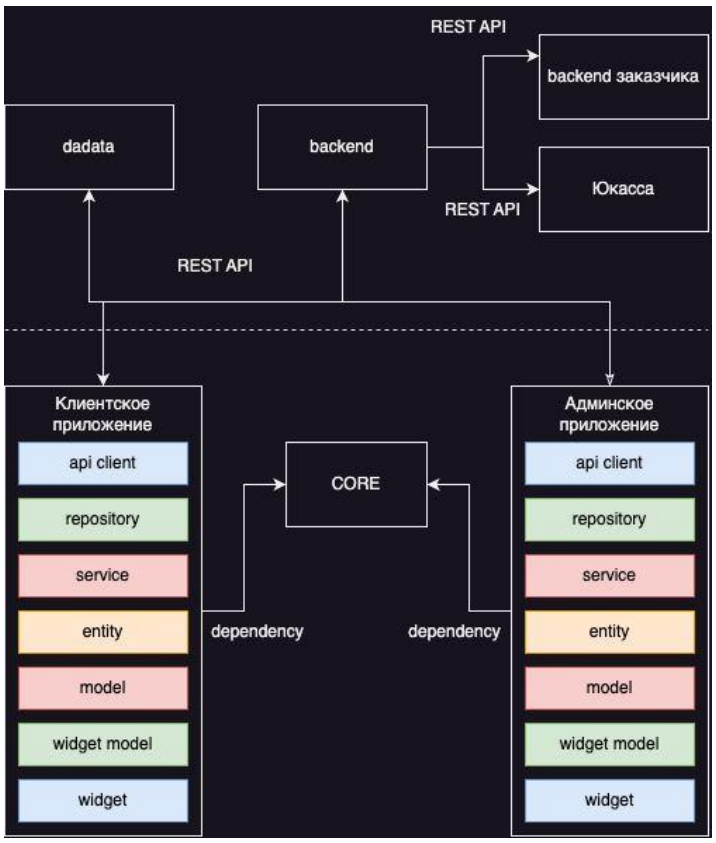

Рисунок 12 - Архитектура приложения

3.9 Архитектура серверной части

Для серверной части используется Spring Boot фреймворк для разработки веб-приложений, который облегчает создание приложений на основе архитектуры MVC (Model-View-Controller).

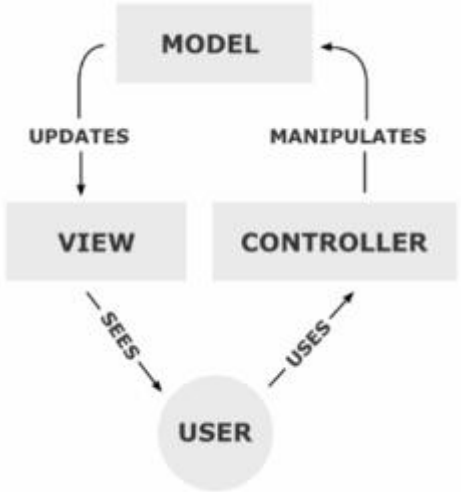

Рисунок 13 - Схема архитектуры MVC

MVC (Model-View-Controller) - это шаблон проектирования, используемый для разработки приложений, которые разделяют данные приложения, взаимодействие пользователя и управление взаимодействием между ними. Он разделяет приложение на три абстрактные составляющие: Model (Модель), View (Представление) и Controller (Контроллер).

Достоинства паттерна MVC следующие:

- Разделение ответственности: Модель содержит данные и логику бизнес-процессов. Представление отображает данные, а Контроллер обрабатывает входные данные и управляет взаимодействием между Моделью и Представлением. Обычно каждый компонент отвечает за свою функциональность, что делает код более читабельным и понятным;

- Возможность переиспользования кода: благодаря разделению различных компонентов приложения, каждый из них может быть переиспользован или заменен без влияния на другие. Обновление компонентов может происходить независимо, что сокращает время разработки и риски ошибок.

- Улучшение тестируемости: возможность тестирования каждого компонента независимо облегчает процесс тестирования и повышает качество приложения в целом;

- Повышение производительности: благодаря эффективному управлению потоками данных, производительность пользователя улучшается.

3.10 Архитектура клиентской части

Для клиентской части используется шаблон проектирования MVVM (Model-View-ViewModel). По своей сути он похож на шаблон MVC с отличиями в архитектуре (Рисунок 14).

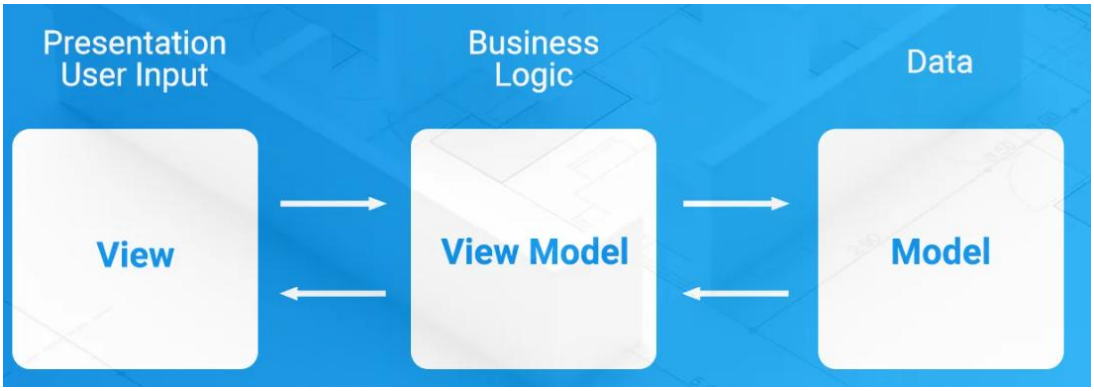

Рисунок 14 - Схема архитектуры MVVM

В данном шаблоне ViewModel, заменяющий Conroller в MVC, выполняет роль прослойки между Model - данными и View - отображением данных. В ViewModel хранится вся бизнес-логика экрана. Перед тем как попасть на экран данные из Model проходят обработку в ViewModel. Все действия пользователя на экране обрабатываются внутри ViewModel.

Преимущества шаблона MVVM:

— Возможность итеративного произвольного стиля программирования. Изолированные изменения менее опасны и более удобны для экспериментирования.

— Упрощение модульного тестирования. Блоки кода, изолированные друг от друга, можно тестировать отдельно и вне рабочих сред.

— Поддержка совместной работы команд. Несвязанный код хорошо спроектированных интерфейсов может быть разработан отдельными пользователями или командами и интегрирован позже.

3.11 Реализация серверной части приложения

3.11.1 Слой доступа к данным

Слой доступа к данным (Data Access Layer, DAL) - это слой абстракции, ответственный за управление доступом к данным в приложении. Он позволяет разработчикам концентрироваться на бизнес-логике, а не на том,

как данные хранятся и как к ним получать доступ. С помощью DAL можно развязать зависимости между кодом, работающим с данными, и представлением, что делает код более общим и переиспользуемым.

JPA Repository - это интерфейс, который предоставляет различные методы CRUD (Create, Read, Update, Delete) для работы с базой данных.

Использование JPA Repository позволяет упростить слой доступа к данным в приложении и сократить время разработки. Он автоматически генерирует SQL-запросы на основе методов, которые определены в репозитории, что уменьшает количество кода, который необходимо написать.

Реализация слоя доступа к данным представлена на рисунке 12:

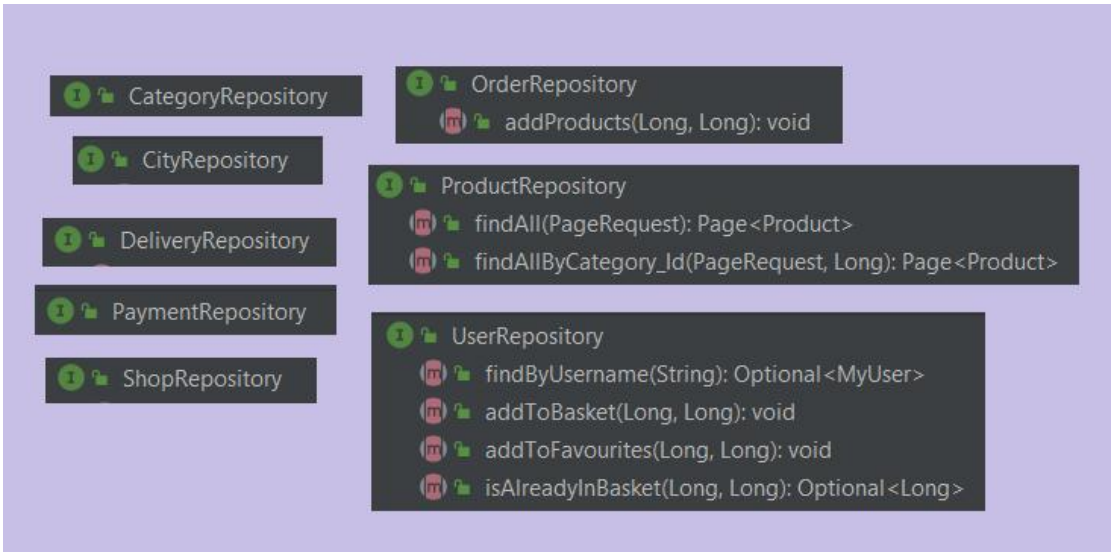

Рисунок 15 - Слой доступа к данным

3.11.2 Слой контроллеров

Слой контроллеров (Controller) - это один из трех компонентов архитектуры MVC в Spring Boot Framework.

Задачи слоя контроллеров в Spring Boot включают в себя:

— Обработка входящих запросов: контроллеры отлавливают запросы от клиентов и передают их в соответствующий метод обработки;

— Извлечение данных запроса: контроллеры извлекают данные, переданные в запросе, обрабатывают их и передают внутрь бизнес-логики приложения для дальнейшей обработки;

— Взаимодействие с бизнес-логикой: контроллеры вызывают нужные методы бизнес-логики, чтобы обработать полученные запросы и передать данные обратно в слой представления для отображения;

— Формирование и отправка ответа: контроллеры формируют ответ на запрос, который может быть представлен как HTML-страница, JSON-объект или любой другой тип данных, и отправляют его обратно клиенту.

Слой контроллеров позволяет разработчикам создавать высокопроизводительные и масштабируемые веб-приложения на основе архитектуры MVC. В целом, слой контроллеров является важным компонентом приложения, который помогает обеспечить гибкость, масштабируемость и управляемость проекта.

Реализация слоя контроллеров представлена на рисунке 13:

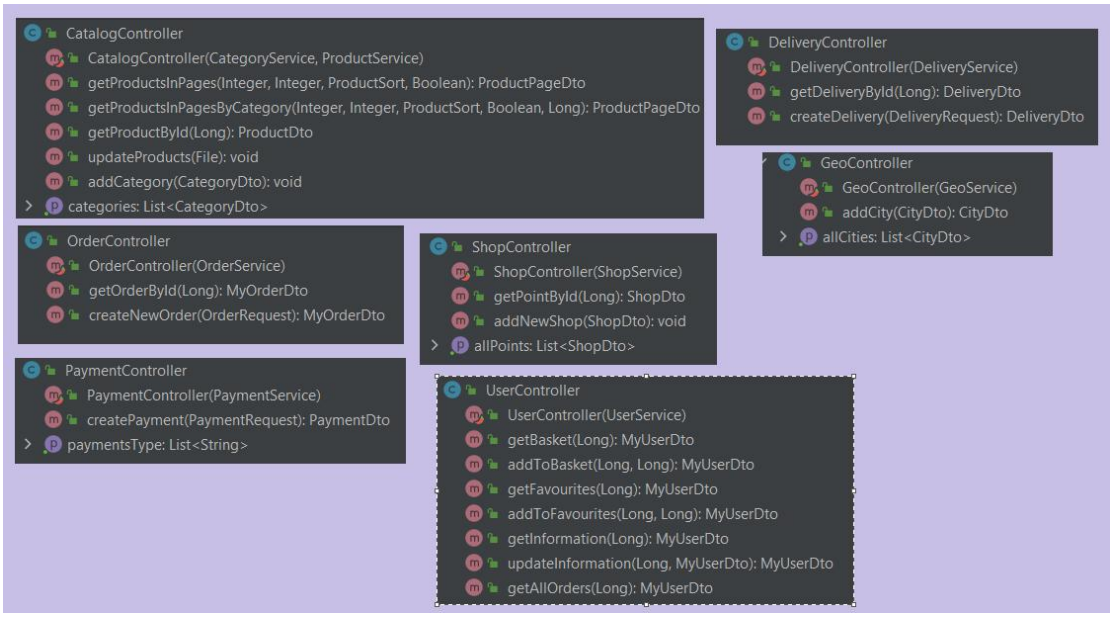

Рисунок 16 - Реализация слоя контроллеров

3.11.3 Слой моделей

Слой моделей (Model) - это один из трех компонентов архитектуры MVC в Spring Boot Framework. Он представляет бизнес-логику приложения и обычно содержит классы, которые представляют различные сущности в приложении.

Задачи слоя моделей в Spring Boot включают в себя:

- Определение структуры данных: модели предоставляют структуру данных для приложения, указывая, какие поля будут содержать каждый объект;
- Валидация данных: модели обеспечивают проверку данных перед сохранением или отправкой пользователю, чтобы гарантировать, что они соответствуют требованиям для данной сущности.;
- Взаимодействие с базой данных: модели содержат бизнес-логику и обеспечивают взаимодействие с базой данных через слой доступа к данным (DAL);
- Создание бизнес-логики: модели обеспечивают реализацию бизнес-логики приложения в отношении конкретных сущностей.

В слое моделей определяется структура и логика работа с бизнес-объектами, которые используются приложением. Это место, где пишется код для интерактивности и функциональности приложения, где описываются бизнес-правила и проверяется корректность данных перед сохранением их в базу данных. Правильно построенный слой моделей обеспечит целостность и надежность всей структуры приложения.

Реализация слоя моделей представлена на рисунках 14 и 15:

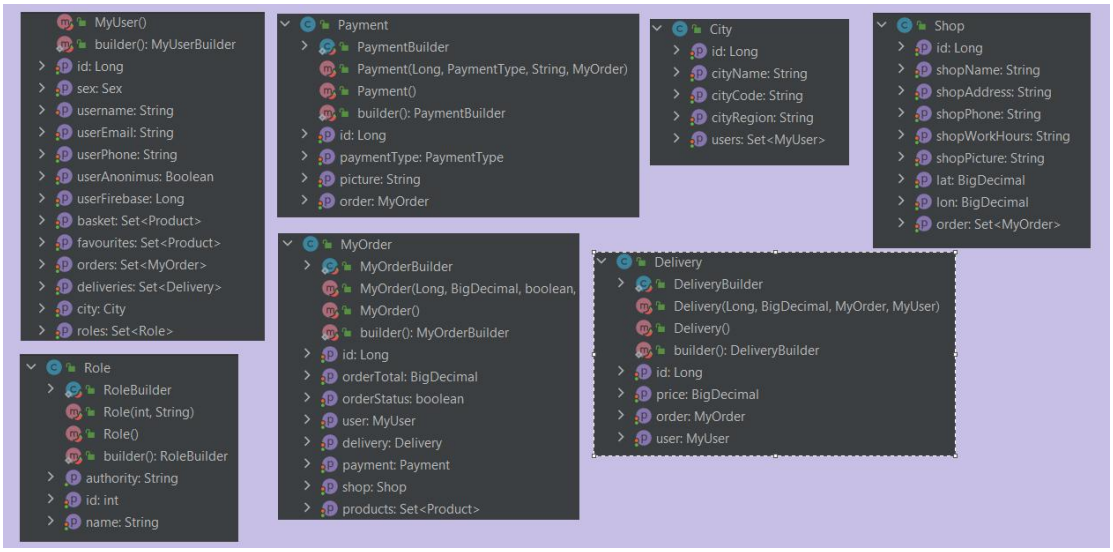

Рисунок 17 - Реализация слоя моделей

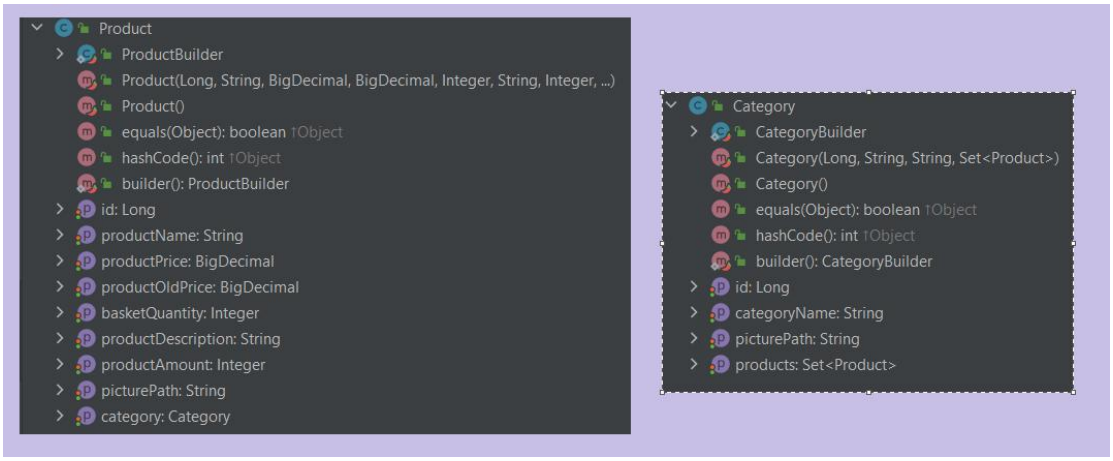

Рисунок 18 - Реализация слоя моделей

3.11.4 Слой бизнес-лоикики

Слой сервисов в бизнес-логике - это промежуточный слой, которые находится между слоем контроллеров и слоем моделей в MVC-архитектуре приложения. Слои контроллеров и моделей занимаются обработкой запросов и предоставлением данных, а слой сервисов обеспечивает бизнес-логику, которая определяет, как эти данные будут обработаны.

Задачи слоя сервисов в бизнес-логике включают в себя:

— Определение бизнес-правил: слой сервисов задает бизнес-правила, определяя, что именно должно происходить при выполнении различных операций;

— Обработка данных: слой сервисов получает данные из слоя моделей, обрабатывает их и возвращает результат в слой контроллеров;

— Управление транзакциями: слой сервисов отвечает за управление транзакциями между слоем моделей и слоем контроллеров, обеспечивая целостность данных в пределах приложения;

— Управление бизнес-логикой приложения: слой сервисов занимается управлением всей бизнес-логикой приложения, обеспечивая ее правильную работу.

Реализация слоя бизнес-логики представлена на рисунках 16 и 17:

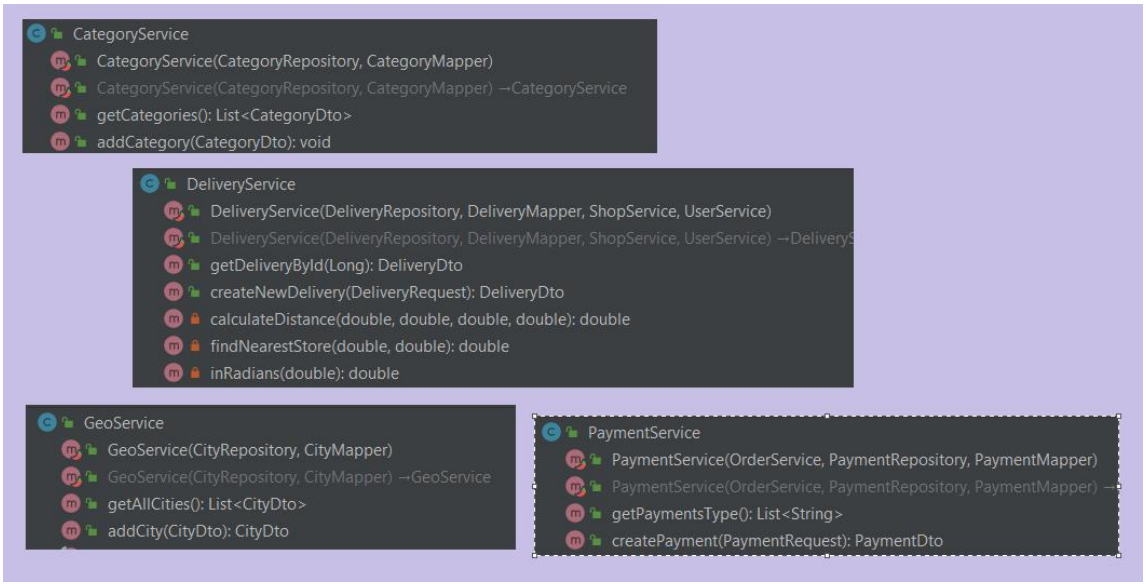

Рисунок 19 - Реализация слоя бизнес-логики

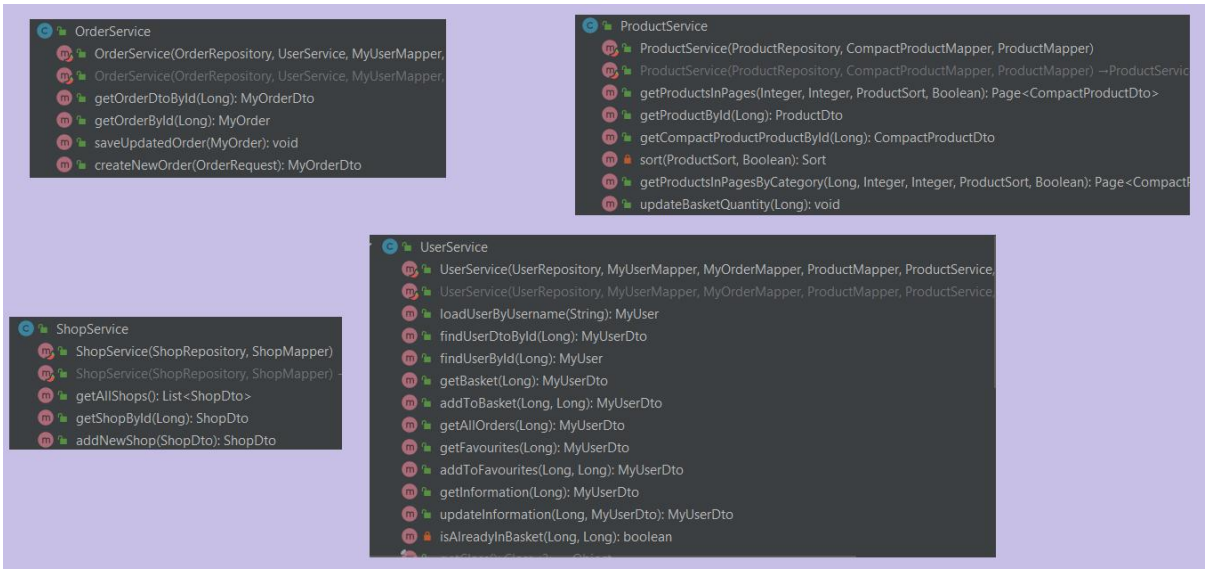

Рисунок 20 - Реализация слоя бизнес-логики

3.11.5 Механика работы приложения

Для работы приложения были реализованы следующие сервисы:

— UserService: имеет методы для формирования корзины пользователя по его id: добавлению и удалению товаров из нее, для формирования списка избранного пользователя по его id с теми же функциями, что и для корзины, получение списка заказов пользователя по его id. Так же предоставляет возможность удаления профиля и регистрации;

— UpdateFeedService: имеет метод, который позволяет сотруднику магазина обновить ассортимент товаров в онлайн магазине, отправив файл в формате json.

— ShopService: имеет методы для добавления нового магазина и демонстрации всех магазинов, откуда можно заказать доставку или забрать заказ самовывозом;

— ProductService: имеет методы для демонстрации карточки товара по его id, а так же методы для постраничного вывода продуктов с

возможностью сортировки по имени и по цене в порядке возрастания или убывания, а так же постраничного вывода продуктов, относящихся к одной категории с той же сортировкой;

— OrderService: имеет методы для создания нового заказа и демонстрции информации о конкретном заказе по его id;

— PaymentService: имеет метод по оплате заказа;

— JwtService: имеет методы для работы с security и firebase;

— GeoService: имеет методы по выводу списка городов России, в которых может проживать пользователь, а также метод для добавления нового города;

— DeliveryService: имеет метод для создания новой доставки, а также дополнительные функции для рассчета стоимости доставки в зависимости от расположения магазина;

— CategoryService: имеет методы для демонстрации всех категорий товаров, их добавления и удаления;

— BannerService: имеет методы для демонстрации баннеров: списки акций, выгодные предложения.

3.12 Реализация клиентской части приложения

3.12.1 Слой репозитеорией

Данный слой отвечает за дотсуп к данным, за взаимодействие с сервером. Этот слой делает все запросы на сервер: добавление или удаление из избранного, добавление или удаление из корзины, получение информации пользователя и т.п.

Классы данного слоя имеют постфикс «Client». Каждая модель имеет свой клиент, через которую делаются все запросы, описанные в сваггере.

3.12.2 Слой сущностей

Слой сущностей отвечает за хранение данных. Модели созданы для каждого возможно json-файла, который приходит с сервера, или отправляется на сервер. Json, полученный от сервера, обрабатывается в объект подходящей сущности или списка сущностей.

3.12.3 Слой сервисов

Данный слой отвечает за хранение данных в памяти устройства. Не все изменения сохраняются в базу данных. Выбранные пользователем город, метод доставки и пункт доставки (если пользователь выбрал самовывоз), сохраняются в память устройства.

3.12.4 Реализация MVVM архитектуры

Каждый экран состоит из 3 отдельных классов, выполняющих функции компонентов шаблона MVVM (Рисунок 21).

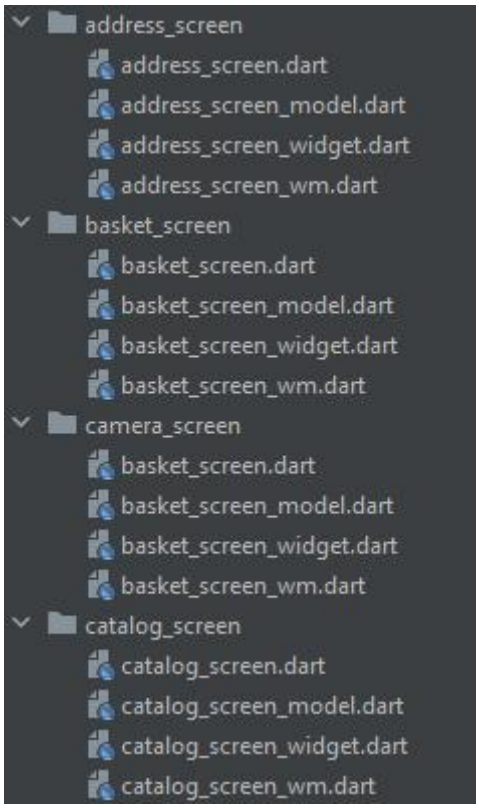

Рисунок 21 - Вид классов экранов в файловой системе

View слой шаблона представляется в виде класса с постфиксом «widget». Это класс отображения, в котором верстается экран.

ViewModel представляется в виде класса с постфиксом «wm» (от WidgetModel). Класс содержит в себе бизнес-логику.

Model слой шаблона представляется в виде класса с постфиксом «model». Внутри этого класса делается запрос в необходимый репозиторий для получения данных.

3.12.5 Слой core

Слой core содержит общие компоненты, используемые в различных частях приложения (Рисунок 22).

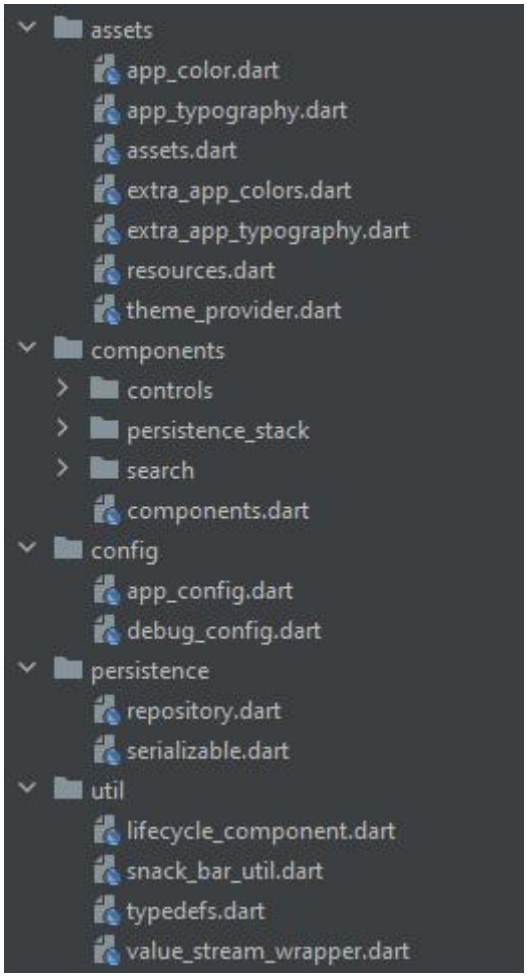

Рисунок 22 - Вид классов слоя core

Основные части слоя:

— Assets: цвета и шрифты приложения;

— Components: часто используемые виджеты, например виджет строки поиска;

— Config: конфигурация подключения к серверу;

— Util: служебные классы;

— Persistence: класс, сохраняющий данные в память устройства.

Заключение

В ходе выполнения данного курсового проекта были выполнены все поставленные задачи.

Все поставленные цели были выполнены:

— Продажа продуктов питания;

— Привлечение новых покупателей.

Список используемых источников

1. Документация Flutter [Электронный ресурс]. – Режим доступа: URL:https://docs.flutter.dev/– Заглавие с экрана. – (Дата обращения 04.05.2023).
2. Документация SpringBoot [Электронный ресурс]. – Режим доступа: URL: https://docs.spring.io/spring-boot/docs/current/reference/html/ – Заглавие с экрана. – (Дата обращения 22.04.2023).
3. Документация Swager [Электронный ресурс]. – Режим доступа: URL: https://swagger.io/docs/ – (Дата обращения 10.05.2023).
4. Документация Elementary [Электронный ресурс]. – Режим доступа: URL: https://pub.dev/documentation/elementary/latest/ – (Дата обращения 10.05.2023).