**Федеральное государственное автономное образовательное учреждение**

**высшего образования**

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**ЛАБОРАТОРНАЯ РАБОТА**

Дисциплина: Программная инженерия

Студент: Кузнецов В.Ю.

Группа: НФИбд-02-21

Студенческий билет № 1032216436

Москва

2023

Маркетплейс Озон

* **Маркетплейс Озон** - это онлайн-платформа, которая объединяет продавцов и покупателей для удобной покупки и продажи различных товаров.

**Техническое задание**

1. Введение

Мобильное приложение "OZON" предоставляет пользователю возможность быстрого и удобного поиска, выбора и покупки товаров, доставка которых осуществляется на дом. Приложение должно быть доступно на платформах iOS и Android.

2. Описание функциональности

2.1 Регистрация и аутентификация пользователей

- Приложение должно предоставлять возможность пользователям зарегистрироваться с помощью электронной почты или аккаунта в социальной сети.

- После регистрации пользователи должны иметь возможность входить в приложение с помощью выбранного метода аутентификации.

2.2 Поиск товаров

- Приложение должно предоставлять возможность пользователям осуществлять поиск товаров по наименованию, категории, бренду и другим параметрам.

- Результаты поиска должны быть отображены в виде списка с возможностью сортировки по цене, рейтингу и другим параметрам.

- Пользователи должны иметь возможность просмотреть подробную информацию о товаре, включая описание, изображения, цену и наличие на складе.

2.3. Оформление заказов

- Приложение должно предоставлять возможность пользователям добавлять товары в корзину и оформлять заказ.

- При оформлении заказа пользователи должны указывать необходимую информацию, такую как адрес доставки, контактные данные и предпочтительный способ оплаты.

- Приложение должно поддерживать различные способы оплаты, включая оплату банковской картой, электронными кошельками и наличными при получении товара.

2.4 Отслеживание заказов

- Приложение должно предоставлять возможность пользователю отслеживать статус своих заказов.

- Пользователи должны получать уведомления о смене статуса заказа, такие как подтверждение заказа, готовность к отправке и доставка.

2.5 Рейтинг и отзывы

- Приложение должно предоставлять возможность покупателям оставлять рейтинг и отзывы о товарах.

- Пользователи должны иметь возможность просмотреть рейтинг и отзывы других пользователей перед покупкой товара.

3. Технологии и инструменты:

- Платформы: Android и iOS

- Языки программирования: Kotlin (для Android) и Swift (для iOS)

- Среда разработки: Android Studio (для Android) и Xcode (для iOS)

- База данных: Рекомендуется использование PostgreSQL или MySQL для хранения данных приложения.

- Уведомления: Реализация через Firebase Cloud Messaging (FCM) для отправки уведомлений на мобильные устройства пользователей.

4. Требования к развертыванию и поддержке:

- Приложение должно быть развернуто на соответствующих маркетплейсах для Android и iOS (Google Play и App Store), и обновляться вместе с новыми версиями.

- Поддержка должна включать регулярные обновления функционала приложения, исправление ошибок и обеспечение безопасности пользователей.

5. Требования к производительности

- Приложение должно обеспечивать быструю и отзывчивую работу при поиске товаров и оформлении заказов.

- Загрузка изображений товаров должна быть оптимизирована для быстрой загрузки.

- Приложение должно иметь масштабируемую архитектуру, чтобы обеспечивать надежную работу даже при большом количестве пользователей и одновременных запросах.

6. Требования к интерфейсу

- Приложение должно иметь привлекательный и интуитивно понятный пользовательский интерфейс.

- Интерфейс должен быть адаптивным и подстраиваться под разные размеры экранов мобильных устройств.

- Взаимодействие с элементами интерфейса должно быть удобным и интуитивно понятным для пользователя.

7. Требования к безопасности

- Приложение должно обеспечивать безопасность пользовательских данных, в том числе личной информации и данных о платежах.

- Аутентификация пользователей должна осуществляться с использованием надежных методов.

8. Требования к интеграции

- Приложение должно интегрироваться с платежными системами, чтобы обеспечивать возможность оплаты заказа.

- Приложение должно интегрироваться с платформами социальных сетей для регистрации пользователей через аккаунты в социальных сетях.

- Приложение должно обеспечивать интеграцию с базой данных товаров и информацией о товарных запасах.

9. Требования к развертыванию и поддержке

- Приложение должно быть развернуто на облачной платформе для обеспечения масштабируемости и надежности.

- Приложение должно иметь возможность обновления и поддержки на протяжении всего его жизненного цикла.

10. Требования к документации

- Разработчик должен предоставить документацию по установке и настройке приложения.

- Документация должна содержать описание функциональности и архитектуры приложения, API и методов интеграции.

11. Требования к тестированию

- Приложение должно быть протестировано на соответствие функциональным требованиям и стабильной работе.

- Должны быть проведены тесты безопасности для подтверждения соответствия требований безопасности и устранения уязвимостей.

12. Требования к срокам разработки

- Разработка приложения должна быть завершена через 6 месяцев.

**Архитектура**

Для разработки мобильного приложения "Озон" рекомендуется использовать архитектурный паттерн MVVM (Model-View-ViewModel). Этот паттерн обеспечит разделение логики приложения и пользовательского интерфейса, а также облегчит его тестирование и поддержку.

Основные компоненты архитектуры MVVM для мобильного приложения "Озон":

Модель (Model):

Отвечает за бизнес-логику приложения и обработку данных.

Включает в себя классы и компоненты, связанные с получением и обработкой данных из сети, кэшированием, базой данных и другими источниками данных.

Модель не зависит от пользовательского интерфейса и обновляет данные с помощью наблюдаемых объектов (observable objects) или хранилищ данных.

Реализуется с использованием классов или структур данных.

Представление (View):

Отвечает за отображение данных пользователю и взаимодействие с пользовательским вводом.

Включает в себя компоненты для отображения пользовательского интерфейса, такие как экраны, макеты, виджеты, элементы управления и др.

Реагирует на пользовательский ввод и передает соответствующие команды или данные во ViewModel для обработки.

Не содержит бизнес-логики и не обрабатывает данные напрямую.

Реализуется с использованием XML-макетов и классов пользовательского интерфейса.

ViewModel:

Связывает модель и представление, предоставляя данные и команды, необходимые для отображения и обработки в пользовательском интерфейсе.

Преобразует и форматирует данные из модели для отображения в представлении.

Реагирует на действия пользователя и обрабатывает их с помощью методов и команд модели.

Может содержать наблюдаемые объекты для обновления представления при изменении данных.

Реализуется с использованием классов или структур данных.

Слой сервисов (Services):

Отвечает за взаимодействие с внешними сервисами или компонентами, такими как API, базы данных, хранилища, авторизация и т.д.

Изолирует модель и ViewModel от низкоуровневых подробностей работы с сервисами.

Реализуется с использованием классов или интерфейсов.

Наблюдая за этой архитектурой, вы сможете достичь следующих преимуществ:

Четкое разделение ответственностей между компонентами.

Улучшенная тестируемость каждого компонента независимо.

Легкость поддержки и расширения приложения.

Возможность повторного использования ViewModel для разных представлений (например, на разных экранах).

Диаграмма Use Case

Изображение выглядит как текст, диаграмма, круг, снимок экрана

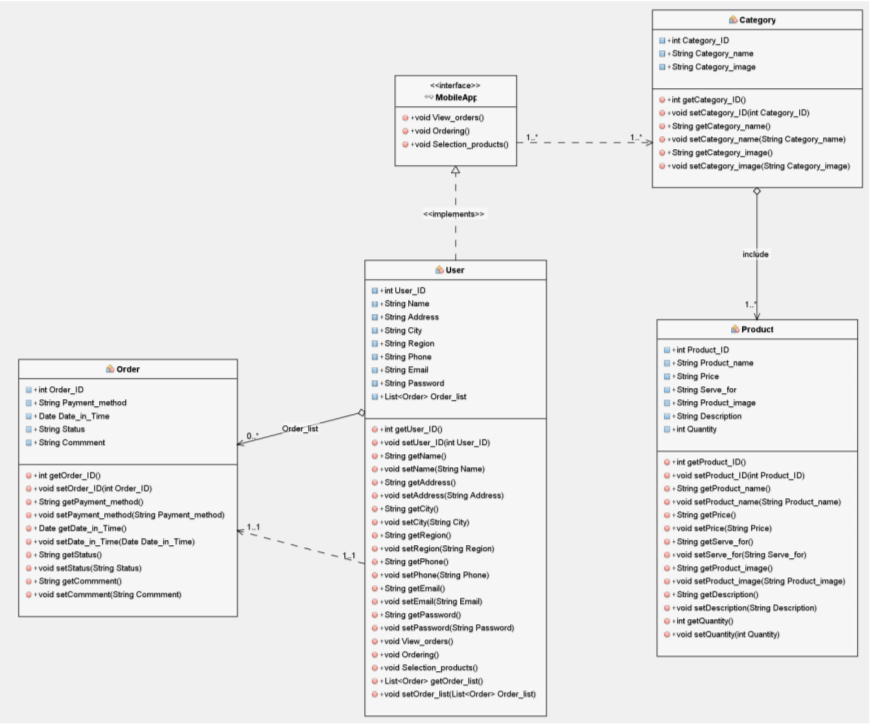
Автоматически созданное описание

Разработанная диаграмма вариантов использования для основных прецедентов проектируемой системы, где актеры, изображенные на ней – это группа лиц или другие системы, взаимодействующие с данной системой, а прецеденты (варианты использования) – это сервисы, обеспечиваемые системой.

Основные функции системы:

* Регистрация и аутентификация пользователей и продавцов
* Поиск и просмотр товаров
* Добавление товара в корзину и изменение её содержания
* Размещение и управление объявлениями о продаже товаров
* Оформление заказа
* Отслеживание статуса доставки
* Взаимодействие продавцов и покупателей
* Проставление оценок и отзывов о товарах и продавцах

Диаграмма классов



На диаграмме представлены основные классы:

User, Category, Product, Order, также интерфейс Mobile App.

* Класс User имеет возможность просматривать, выбирать товары и производить заказы через Mobile App, данный класс хранит в себе данные такие как: имя, адрес, город, регион, номер телефона, эл. почта, пароль для авторизации и список выбранных им товаров.
* Класс Category содержит виды продукции, а именно их наименование и изображение категории.
* Класс Product содержит данные о товаре, такие как: наименование, категорию, цену товара, наличие товара на складе, его количество и его описание.
* Mobile App предоставляет возможность клиенту просматривать товары, выбирать необходимую ему продукцию в нужном ему количестве, а также оформлять заказ.
* Order хранит в себе данные о заказе сделанным пользователем например: способ оплаты, дату доставки, статус заказа и комментарий.

Диаграмма последовательности

