МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ
ИМЕНИ ГУМАРБЕКА ДАУКЕЕВА»
Кафедра IT-инжиниринг

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой
PhD, доцент
_____ А.А.Досжанова
«____» _____ 2020 г.

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка мобильного приложения для магазина «Unimag» на базе ОС «Android»

Специальность 5В070300 – «Информационные системы»

Выполнил Омаров И.О. Группа ИСу-17-3
Научный руководитель PhD, доцент, Картбаев Т.С.

Консультанты:
по экономической части: к.э.н., доцент _____ К.Р.Габелашвили
«____» _____ 2020 г.

по безопасности жизнедеятельности: к.т.н., доцент _____ Н. Г.Приходько
«____» _____ 2020 г.

по применению
вычислительной техники: ст. преп. _____ М.Н.Майкотов
«____» _____ 2020 г.

Нормоконтролер: ст. преп. _____ Б.Р.Абсатарова
«____» _____ 2020 г.

Рецензент: PhD _____
«____» _____ 2020 г.

Алматы 2020

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ
ИМЕНИ ГУМАРБЕКА ДАУКЕЕВА»

Институт систем управления и информационных технологий

Кафедра IT-инжиниринг

Специальность 5В070300 – «Информационные системы»

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Омарову Исламу Оркашевичу

Тема работы: Разработка мобильного приложения для магазина «Unimag» на базе ОС «Android»

Утверждена приказом по университету № _____ от «__» _____ 2020 г.

Срок сдачи законченного проекта «_____» _____2020 г.

Исходные данные к работе (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): структура предприятия, должностные обязанности, примеры базы данных.

Перечень вопросов, подлежащих разработке в дипломной работе, или краткое содержание дипломной работы:

а) исследование и анализ существующих приложений;
б) выбор программного обеспечения;
в) проектирование базы данных;
г) разработка интерфейса приложения;
д) программная реализация;
е) охрана труда и безопасность жизнедеятельности;
ж) экономическая эффективность.

Перечень графического материала (с точным указанием обязательных чертежей): представлены 13 таблиц, 83 иллюстраций.

Основная рекомендуемая литература:

1 Брайан Харди, Билл Филлипс, Крис Стюарт, Кристин Марсикано. Программирование под Android. 2-е издание (2016).

2 П. Дейтел, Х. Дейтел, А. Уолд. Android для разработчиков. 3-е издание (2016).

3 Дэвид Гриффитс, Дон Гриффитс. «Head First. Программирование для Android». Питер – 2018.

Консультации по работе с указанием относящихся к ним разделов работы

Раздел	Консультант	Сроки	Подпись
Экономическая часть	Габелашвили К.Р.	05.04.2020-23.04.2020	
Безопасности жизнедеятельности	Приходько Н.Г.	05.04.2020-22.04.2020	
Нормоконтролер	Абсатарова Б.Р.	13.05.2020-18.05.2020	
Программная часть	Майкотов М.Н.	14.05.2020-15.05.2020	

График
подготовки дипломной работы

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Исследование и анализ существующих приложений	13.01.2020-23.01.2020	
Выбор программного обеспечения	24.01.2020-01.02.2020	
Проектирование базы данных	02.02.2020-16.02.2020	
Разработка интерфейса приложения	17.02.2020-08.03.2020	
Программная реализация	09.03.2020-14.05.2020	

Дата выдачи задания «20» января 2020 г..

Заведующий кафедрой _____А.А. Досжанова

Научный руководитель работы _____Картбаев Т.С.

Задание принял к исполнению студент _____ И.О. Омаров

Аңдатпа

Берілген дипломдық жобаның мақсаты - Unimag ЖК үшін мобилдік қосымшаны құрастыру, дүкеннің коммерциялық жұмысын автоматтандыру және оның электрондық коммерция саласында әрі қарай дамуту.

Жобада өндіріс құрылымы, оның бизнес процестері және оларды оңтайландыру, құрастырылған қосымша үшін деректер базасының жобалау, оның интерфейсі, Google-дің Flutter фреймворктің көмегімен бағдарламалық жүзеге асыруы қарастырылды. Ол Android пен iOS платформаларында қосымшаларды құрастыруға көмегін тигізеді.

Сонымен қатар жобаның технико-экономикалық негіздеуі, еңбекті қорғау және қауіпсіздік мәселелері қарастырылды.

Аннотация

Данный дипломный проект нацелена на разработку мобильного приложения для ИП «Unimag» для автоматизированного ведения коммерческой деятельности магазина и его дальнейшего развития в сфере электронной коммерции.

В проекте рассмотрены структура предприятия, ее бизнес-процессы и их оптимизация, проектирование базы данных для разрабатываемого приложения, его интерфейс и программная реализация с помощью фреймворка Flutter от Google, который помогает разрабатывать приложения как на Android, так и на iOS платформы, а также web.

Также рассмотрено технико-экономическое обоснование проекта и вопросы безопасности жизнедеятельности и охраны труда.

Abstract

This thesis is aimed at developing a mobile application for IE “Unimag” for the automated conduct of commercial activities of the store and its further development in the field of electronic commerce.

The project considers the structure of the enterprise, its business processes and their optimization, database design for the developed application, its interface and software implementation using the Flutter framework from Google, which helps to develop applications on both Android and iOS platforms, as well as the web.

The feasibility study of the project and the issues of life safety and labor protection were also considered.

Содержание

Введение

Так как в современном мире количество пользователей смартфонов растет с каждым годом, а постоянная конкуренция на рынке заставляет компании развиваться и использовать различные инновации в своем бизнесе, чтобы идти наравне с быстро изменяющимися тенденциями, было принято решение использования смартфоны как площадку для реализации своих продуктов и услуг.

E-commerce («электронная торговля») за короткое время попадает в ряд с другими популярными видами предпринимательства, которые открываются пути реализации различных типов бизнеса.

Мобильные приложения позволяют мгновенно оповещать клиентов о заказах, проводить транзакции в один клик и автоматизировать продажи.

По результатам опубликованной статистики на 2018 год аналитической компании Pew Research Center, смартфоны имеют 59% населения мира [1]. Оставшийся 41% населения имеют либо простыми мобильными телефонами, известные как «простушки», либо нет ни того, ни другого.

На сегодняшний день Android и iOS являются самыми популярными платформами. Смартфоны на платформе Android имеют 85,9% населения планеты, оставшиеся 14,1%. Остальные платформы по оценкам Gartner составляет значение приближенное к 0%.

Пользователей смартфонов в Казахстане насчитывается 11,9 миллиона пользователей или же 64,9% населения [2].

Цель дипломного проекта разработать мобильное приложения для магазина “Unimag” с использованием современных программных инструментов: Flutter, MySQL. Для реализации данной цели необходимо выполнить следующие задачи:

- на основе анализа ПО мобильных приложений для e-commerce зарубежных и отечественных компаний выполнить их сравнительную характеристику;
- изучить структуру и назначение предприятия заказчика;
- обосновать выбор программных инструментов;
- разработать клиент-серверную архитектуру программного приложения, спроектировать диаграммы взаимодействия модулей приложения, смоделировать базу данных и выполнить программную реализацию мобильного приложения;
- рассчитать экономическую часть проекта;
- разработать мероприятия по охране труда и безопасности жизнедеятельности на предприятии.

В пояснительной записке данного проекта представлены все этапы разработки ПО мобильного приложения, приведены расчеты по экономической части и по разделу охраны труда, в приложении представлены листинги кодов.

1. Анализ предметной области и требования к системе

1.1 Анализ и сравнительная характеристика мобильных приложений для коммерческих целей.

Для дальнейшего проектирования и разработки необходимо провести анализ и сравнение существующих e-commerce приложений.

В качестве примеров представлены решения известных площадок для электронной коммерции, таких как Lamoda, WildBerries, Flip.kz.

Эти интернет-магазины имеют широкую аудиторию, более 1-2 миллионов клиентов, а также высокие оценки пользователей, в среднем 4,6 из 5 баллов.

Первым будет рассмотрен лидер в категории «Продажа одежды» Lamoda [3].

Приложение имеет минималистичный, но информативный и вполне дружелюбный интерфейс. Первым что отображает приложение это preloader, затем представит страницу с распродажей сезонной одежды и всем сопутствующим товарам, а также наиболее популярные товары по выбранной категории, рисунок 1.1.

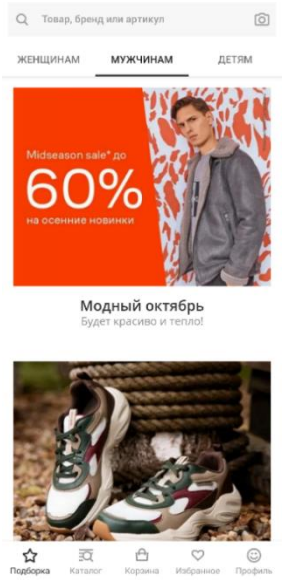


Рисунок 1.1 – Главная страница приложения Lamoda

Сверху размещены поисковая строка с возможностью искать по наименованию товара, бренду, артикулу и фотографии. Далее 3 категории, разделенные по гендерному признаку. Следом разделы под товар. В самом низу расположены основные вкладки приложения: «Подборка», «Каталог», «Корзина», «Избранное» и «Профиль».

Во вкладке «Каталог», на рисунке 1.2, представлены категории в которых товар разбит на более уточненные категории, к примеру категория «Одежда»

представляет следующий список категорий: блузы и рубашки, боди, брюки, верхняя одежда и т.д., представлены на рисунке 1.3. Помимо этого, имеется кнопка «Показать все товары» по нажатию которой открывается страница со всеми товарами в категории «Одежда».

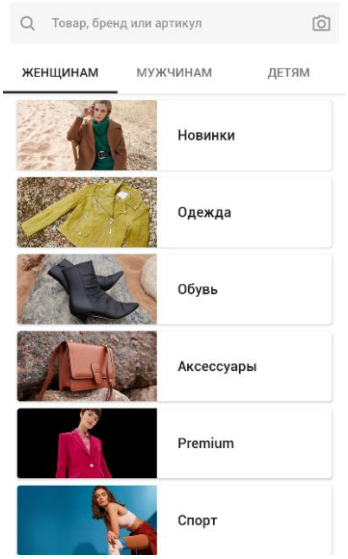


Рисунок 1.2 – Раздел «Каталог»

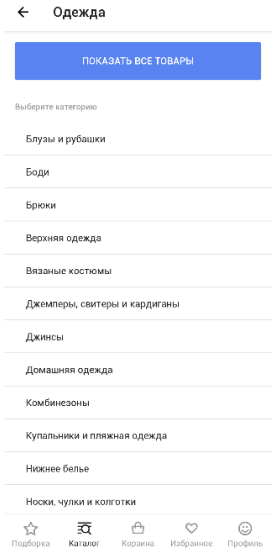


Рисунок 1.3 – Категории раздела «Одежды»

На странице виден весь товар, имеющийся в интернет-магазине из раздела «Одежда».

Основной элемент каталога – карточка товара. Как видно на рисунке 1.4, карточка имеет минимальный набор необходимой информации, такой как наименование товара, бренда, имеющиеся размеры и цену, а также фото. В правом верхнем углу карточки находится кнопка в виде «сердечка», она позволяет отметить товар как понравившийся и разместить его для быстрого поиска во вкладке «Избранный».

Страница включает в себя фильтр, который видно на рисунке 1.5 имеющий некоторый список атрибутов, по которым возможно отобрать интересующий товар: цвет, размер, материал и т.д. Также страница включает сортировку, она имеет значения: популярности, возрастания и убывания цены, новинки и скидки.

На рисунке 1.6 представлена страница товара, которая предоставляет более подробную информацию. Такие, как описание товара, его атрибуты, имеющиеся цвета и размеры, что видно рисунке 1.7. Помимо нее выходит информация о городе, сроке и цене доставки с условием. Город можно изменить, нажав на кликабельный текст. Также возможно посмотреть схожие товары, недавно просмотренные товары. Далее идет 3 раздела «Доставка и примерка», «Задайте нам вопрос» и «Центр поддержки».

Выше основных вкладок имеется кнопка добавления товара в «Избранное» и кнопка добавление товара с выбранными параметрами размера и цвета. Добавив некоторый товар, рассмотрим вкладку «Корзина».

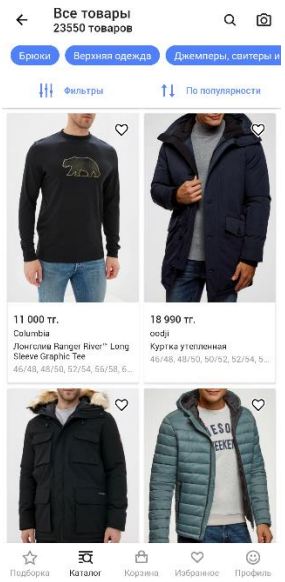


Рисунок 1.4 – Каталог товара раздела «Одежда»

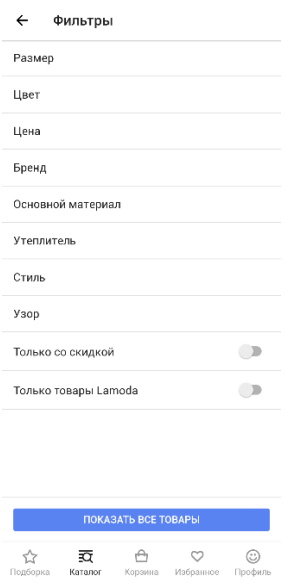


Рисунок 1.5 – Фильтры

Во вкладке «Корзина» располагаются вещи, которые были ранее добавлены, как потенциальные варианты к покупке. В корзине есть возможность редактировать список товаров и их атрибуты. Однако при изменении свойств, приходится вновь добавлять данный товар и удалять ранее выбранный.

Рисунок 1.6 – Страница товара

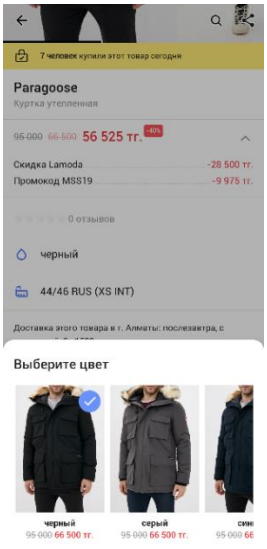


Рисунок 1.7 – Параметры товара

Внешний вид интерфейса вкладки «Корзина» представлен на рисунке 1.9, там же видны те товары, что были добавлены ранее.

Здесь же представлены цены товара: бывшая цена, новая цена с учетом скидки, сама скидка; поле промокода, если он имеется; количество товаров в заказе, общая сумма без учета скидок, вычет скидок, если таковые имеются и итоговая сумма к оплате.

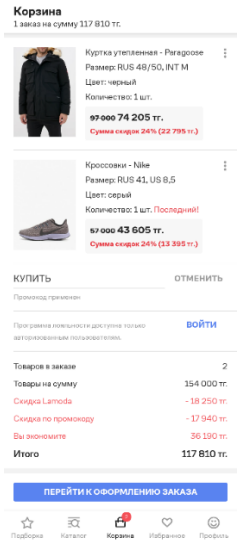


Рисунок 1.9 – вкладка «Корзина»

В конце страницы неявный сценарий транзакции предлагает перейти к оформлению заказа. Страница представлена на рисунке 1.10, и она предлагает заполнить контактные данные, а также подписаться на рассылку.

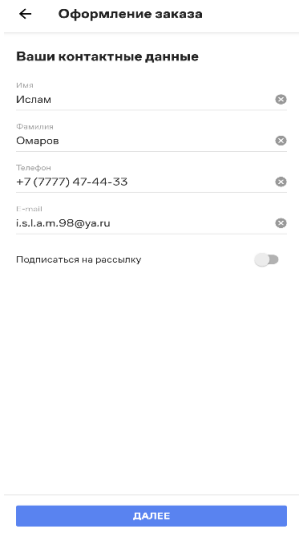


Рисунок 1.10 – Страница Оформления заказа

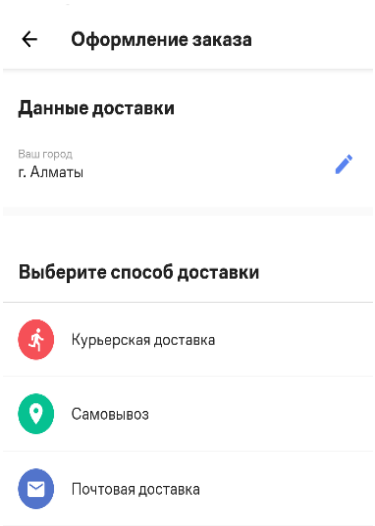


Рисунок 1.11 – Данные для доставки

При нажатии кнопки далее выполняется переход на продолжение сценария оформления заказа, где предлагают заполнить адрес доставки и выбрать способ доставки, страницу видно на рисунке 1.11.

Транзакция разбита на несколько частей для того, чтобы предоставлять информацию для пользователя порционно.

Рассмотрим варианты доставки: курьерскую доставку, самовывоз и почтовая доставка.

Нажав на курьерскую доставку, выполняется переход на страницу, на которой необходимо заполнить форму и выбрать условие доставки. На рисунке 1.12 представлена форма с его содержанием. Далее по сценарию идет выбор даты доставки и времени доставки, под этими пунктами имеется текст серого цвета содержащее условия доставки. Эти пункты изображены на рисунке 1.13.

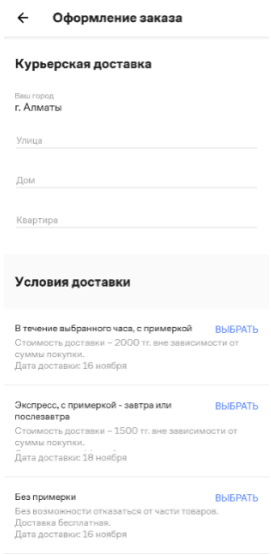


Рисунок 1.12 – Доставка заказа курьерской службой

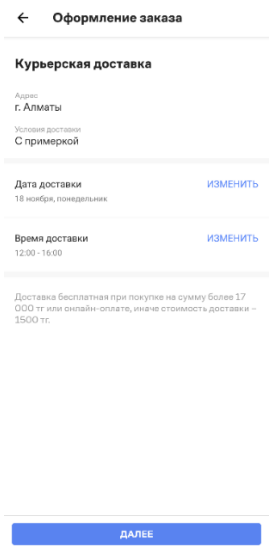


Рисунок 1.13 – Дата доставки

После нажатия кнопки «Далее», сценарий идет к заключительному моменту к способу оплаты и подтверждению заказа. Эта страница предлагает выбрать способ оплаты, с вариантами которых можно ознакомиться на рисунке 1.14. Так как страница является подтверждающей на ней выведена информация о заказе (количества товаров и цена за них, цена за доставку, способ оплаты, всевозможные скидки и итоговая сумма.

После нажатия кнопки «Оформить заказ» выходит страница об успешном оформлении заказа, но сначала необходимо ввести код, который придет на телефон.

Вариант «Самовывоза», представлен на рисунке 1.15. Он имеет только строку ввода города и вопрос о необходимости примерки, по сути это опция, которая будет указана у оператора в момент обработки заказа.

После выбора подходящей опции пользователя переносит на страницу с картой, на которой расположены точки вывоза Lamoda, над картой, которую

можно увидеть на рисунке 1.16 имеются две опции: «С примеркой» и «Оплата картой».

Это своего рода фильтры для точек. Ниже карты в виде списка представлены те же точки что и на карте. Однако, можно ввести в поиск нужную точку и путем отбора отфильтруют необходимое место.

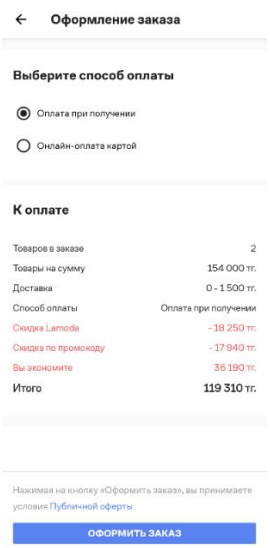


Рисунок 1.14 – Финальная страница оформления заказа

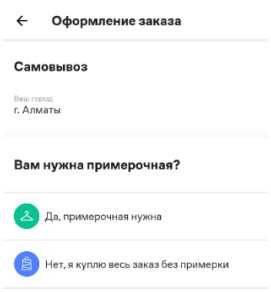


Рисунок 1.15 – Вариант самовывоза

Выбрав место для самовывоза список изменится на данные об отделении и представит варианты опций заказа. Это видно на рисунке 1.17, который изображен выше. После выбора опции выполняется переход на страницу подтверждения данных и необходимо указать удобную дату доставки заказа в пункт самовывоза.

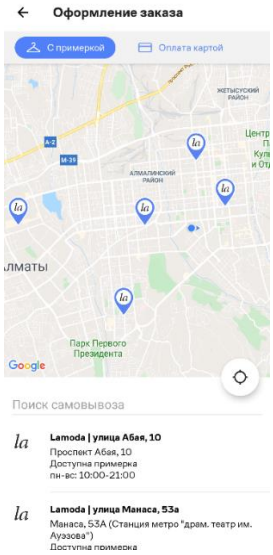


Рисунок 1.16 – Выбор точки

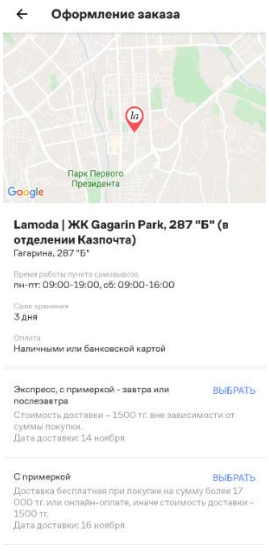


Рисунок 1.17 – Выбранное место

Следом за выбором времени следует информация, такая как: время работы пункта, осведомление о доставке товара и условия доставки. Более детально можно рассмотреть на рисунке 1.18.

После нажатия кнопки «Далее» идет страница оплаты и подтверждения.

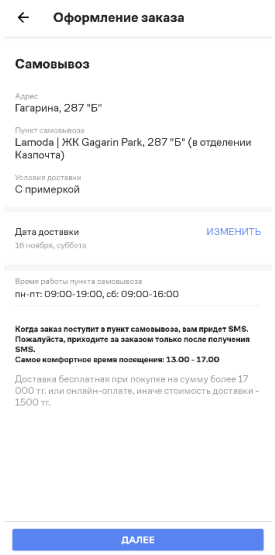


Рисунок 1.18 – Самовывоз

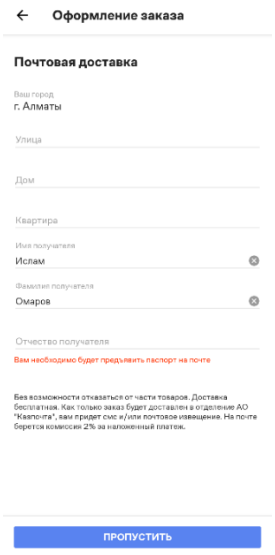


Рисунок 1.19 – Почтовая доставка

Со страницей почтовой доставки можно ознакомиться на рисунке 1.19. Тут предлагают заполнить форму, ознакомиться с условием и перейти к форме оплаты. Форма оплаты выглядит аналогично за исключением стоимости доставки.

Вкладка «Профиль» – это страница для входа в личный аккаунт, если он имеется, то необходимо авторизоваться, если же не имеется, то пользователю приложения предлагают зарегистрироваться. Также есть возможность авторизоваться в приложение с помощью социальных сетей, это удобная функция так как нет необходимости создавать очередной аккаунт и запоминать новые логин и пароль от аккаунта.

Содержимое «Профиля», изображенное на рисунке 1.20, будет рассмотрено далее. Зарегистрировавшись, осуществляется перенос на страницу личного профиля, что на ней находится видно на рисунке 1.21.

Первая строка связана со скидочными программами. При оплате заказов начисляются бонусы и чем больше этих бонусов, тем больший процент будет составлять скидка. Нажав на кнопку «О программе лояльности» происходит перенос на страницу изображенную на рисунке 1.22, где кратко объясняют, что это за программа, какие условия получения баллов и скидок, каким образом их можно получить и есть калькулятор баллов, эту страница изображена на рисунке 1.23.

«Ваши заказы» это страница с активными заказами, если их нет, то справа показывают 0. Однако, при наличии активных заказов их статус можно будет отслеживать именно в этой вкладке.

Далее кнопка с «Управлением подписок». Это грубо говоря рассылка об акциях, распродажах и прочих событиях.

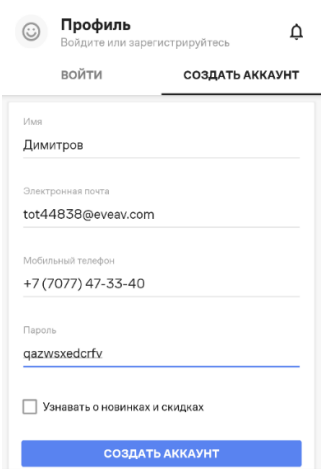


Рисунок 1.20 – Регистрация

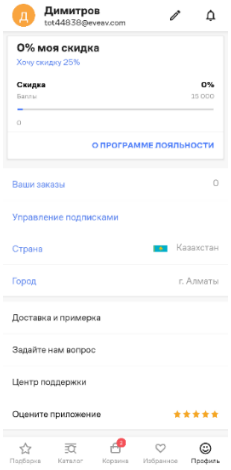


Рисунок 1.21 – Профиль

Затем идет «Страна». Здесь доступно 4 варианта: Беларусь, Украина, Россия и Казахстан. Это страны, в которых действует Lamoda и в которых имеются их пункты. И ниже кнопка «Город».

«Доставка и примерка», здесь нужно ввести населенный пункт для доставки. Кнопка «Задайте нам вопрос», эта кнопка для связи с операторами для уточнения интересующих вопросов, с которыми можно связаться с помощью мессенджера.

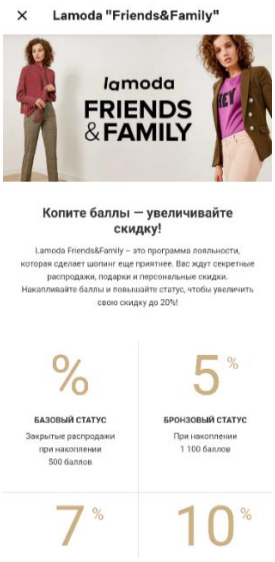


Рисунок 1.22 – Описание программы

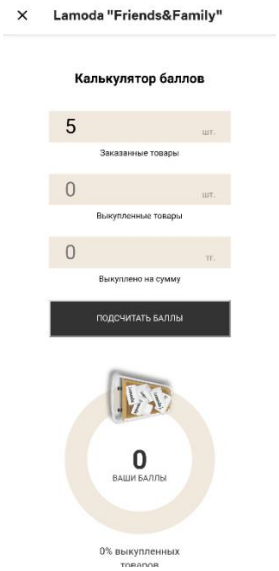


Рисунок 1.23 – Калькулятор баллов

Wildberries – это крупный международный интернет-магазин формата маркетплейс родом из России. Россия, Беларусь, Казахстан, Киргизия и Армения – это страны, в которых работает данный магазин.

При открытии приложения на смартфоне пользователя встречает главная страница, которая изображена на рисунке 1.24, на ней представлены в крупном окне скидки на ряд товаров и новые вещи разных категорий, а также строка поиска, где поиск осуществляется с помощью текста, аудио ввода, QR-кода и фото товара.

Внизу расположена панель со стандартными вкладками «Главная», «Каталог», «Корзина» и «Профиль». Страницы имеют много общего с аналогичными страницами прошлого приложения, однако они имеют свой определенный набор данных, который посчитали наиболее необходим.

Как выглядит каталог видно на рисунке 1.25. Он имеет практически такую же структуру, но из-за разницы ассортиментов имеет более массивный набор категорий и подкатегорий.

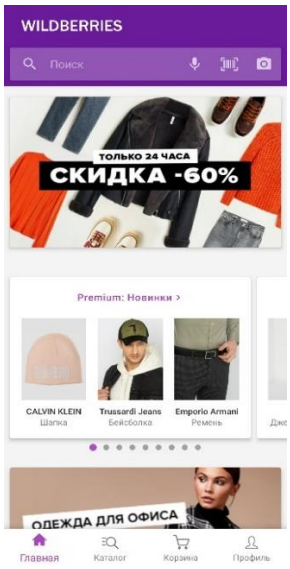


Рисунок 1.24 – Главная страница

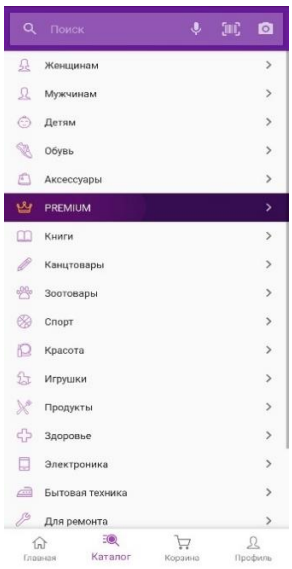


Рисунок 1.25 – Каталог

С интерфейсом списка товаров можно ознакомиться на рисунке 1.26. Карточки товара несут в себе изображение товара, цены (старую и со скидкой), название, оценку, кнопки добавить в избранное и/или корзину.

Выбрав любой товар выполняется переход на его страницу. На рисунке 1.27 представлена страница товара, где выводятся название, цена, артикул, цвет, состав и подробное описание которые можно рассмотреть на рисунке 1.28, таких как производитель, материалы подкладки, внешний и т.д. Имеются кнопка отзывов и вопросов по данному товару, а внизу расположена кнопка «в корзину». После добавления товара в корзину будет рассмотрен раздел «Корзина».

Корзина имеет шаблонный интерфейс: товар, его параметры, цена за единицу, количество, поле ввода промокода; затем информация по заказу:

количество товаров в заказе, на какую сумму, сумма с учетом скидки и общая сумма к оплате. Более детально на рисунке 1.29.

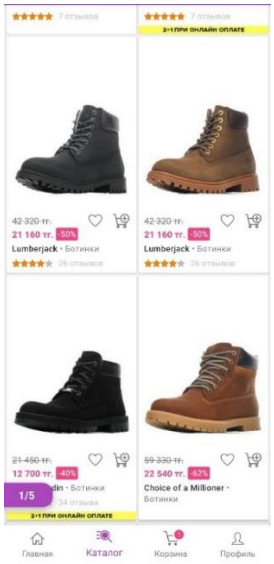


Рисунок 1.26 – Список товаров

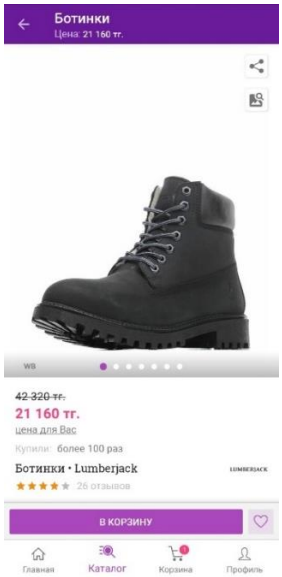


Рисунок 1.27 – Товар

Чтобы оформить заказ необходимо авторизоваться или зарегистрироваться. Форма регистрация изображена на рисунке 1.30 и имеет такие поля как: имя, почта, номер, пароль, чекбоксы для предложений от Wildberries.

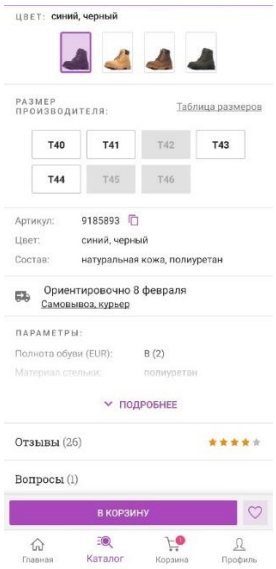


Рисунок 1.28 – Подробное описание

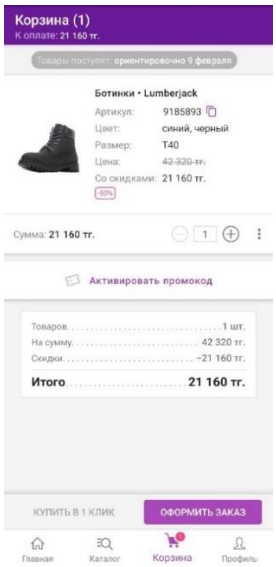
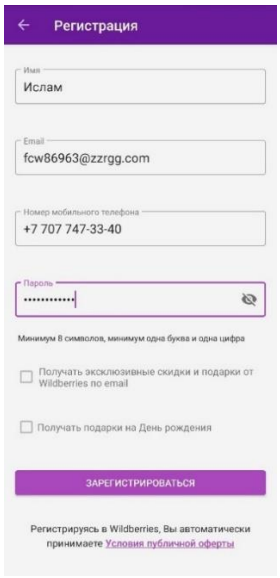


Рисунок 1.29 – Корзина

Нажав кнопку зарегистрироваться, пользователь соглашаемся с условиями публичной оферты. После нажатия просят ввести код, который

отправят на номер, указанный ранее, после успешного ввода появляется возможность полноценно пользоваться приложением.



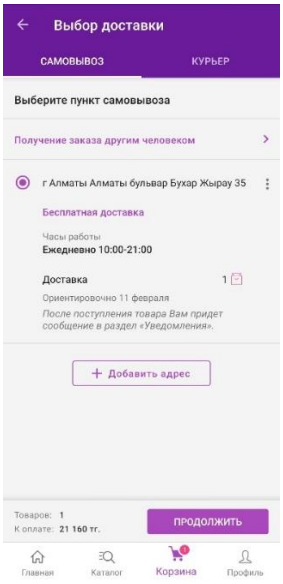


Рисунок 1.30 – Регистрация

Рисунок 1.31 – Самовывоз

По нажатию «оформить заказ» пользователя переносит на страницу выбора доставки. Имеется два варианта — это самовывоз и курьер.

Вариант самовывоз представлен на рисунке 1.31. Здесь необходимо выбрать пункт самовывоза, выбрать его можно тот который указывали ранее, если такого не имеется, то нужно добавить, нажав кнопку «+ Добавить адрес». После этого появится карта с расположениями точек самовывоза, нужно выбрать одну из точек, и она добавится в список.

Выбрав точку из списка, показывается время работы точки и время доставки товара туда. Нажав «Продолжить», происходит переход на страницу оплаты, рассмотреть можно на рисунке 1.32, где необходимо выбрать способ оплаты, сумму к оплате и нажать оформить заказ, где нужно будет ввести код сообщения для подтверждения и ожидать связи с оператором.

Доставка курьером. Внешне вкладка с этим вариантом выглядит аналогично самовывозу. По нажатию кнопки добавления выходит та же карта, которая изображена на рисунке 1.33, но уже без точек самовывоза. Здесь необходимо указать точку на карте куда необходимо доставить товар, после выбора точки выходит форма заполнения адреса, где адрес улицы уже введены, остается лишь ввести номер дома, квартиры, подъезд и этаж пометить частный дом это или нет. Далее нажать кнопку сохранить.

После выбора точки, выполнятся переход обратно и выходит оповещение, что доставка будет платной и предлагает вариант с бесплатным самовывозом. Здесь уже необходимо лишь подтвердить свой выбор и выполнится переход на страницу с оплатой.

Рисунок 1.32 – Форма оплаты

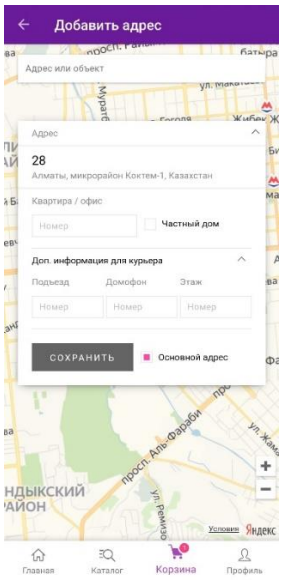


Рисунок 1.33 – Адрес доставки

Далее вкладка «Профиль». По составу, с которым можно ознакомиться на рисунках 1.34 и 1.35, она несколько отличается. Она имеет те же личные данные пользователя и информацию о заказах и справочный раздел. Однако помимо них имеется кнопка для перехода на страницу с избранными товарами. Затем «Лист ожидания» – это лист с заказами которые находятся в обработке. «Мои доставки» – эта вкладка для отображения доставляемых товаров, которые уже прошли обработку и уже отправлены клиенту. «Мои покупки» – уже купленные товары. И разделы сродни тем, что у прошлого приложения: скидочные программы и уведомления.

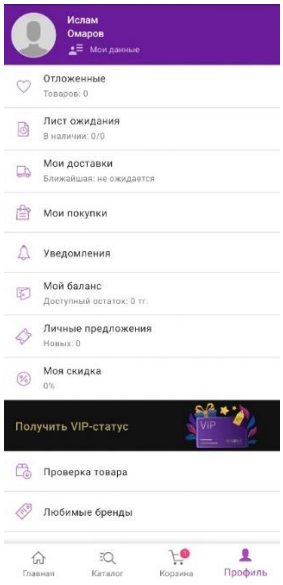


Рисунок 1.34 – Вкладка «Профиль»

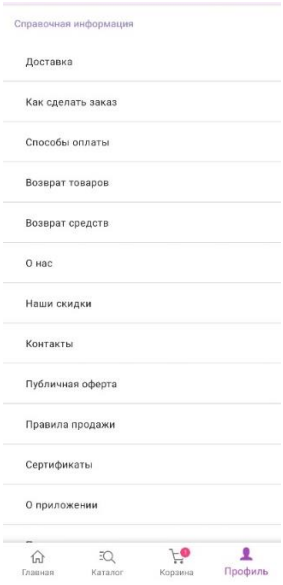


Рисунок 1.35 – FAQ

Flip.kz – это грубо говоря аналог Wildberries, он также является интернет-магазином формата маркетплейс, но созданный в Казахстане.

Интерфейс главной страницы, как можно заметить на рисунке 1.36, не такой шаблонный, как у приложений, рассмотренных ранее. Нет панели с вкладками в нижней части экрана, нет той строки поиска во всю ширину экрана, нет значка уведомлений в правом верхнем углу, вместо него здесь красуется корзина. В левом верхнем углу находится значок меню, рассмотреть можно на рисунке 1.37, в котором есть такие пункты как вход в профиль и регистрация, корзина, заказы, контакты и помощь или же справочная информация, однако тут она не столь обширная.

Если грубо прикинуть, то интерфейс больше напоминает сайт адаптированный под мобильную версию браузера.

Для нетипичного приложения нетипичное начало. Первым будут рассмотрены формы регистрации и авторизации, их можно увидеть на рисунке 1.38 и 1.39.

Форма регистрации имеет две вкладки: частные лица и юридические лица, формы имеют один вид, за исключением того, что во вкладке юридического лица нужно вписать название компании.

Форма входа, которая ранее не описывалась, имеет простой вид и везде он одинаковый по набору необходимых данных.

После авторизации/регистрации, пользователя переносит на страницу профиля, где имеются несколько пунктов с которыми можно ознакомиться на рисунке 1.40.

Так как это маркетплейс, как и Wildberries, то здесь довольно внушительный ассортимент товаров помимо одежды, поэтому здесь столь же внушительный набор категорий, подкатегорий и т.д., детально на рисунке 1.41.

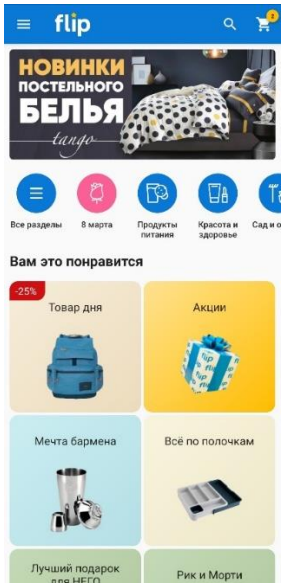


Рисунок 1.36 – Главная страница

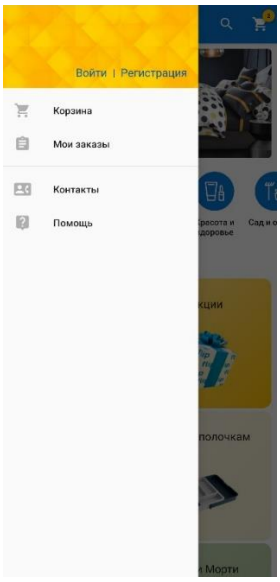


Рисунок 1.37 – Вкладка меню

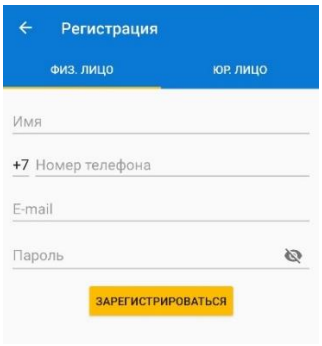


Рисунок 1.38 – Регистрация

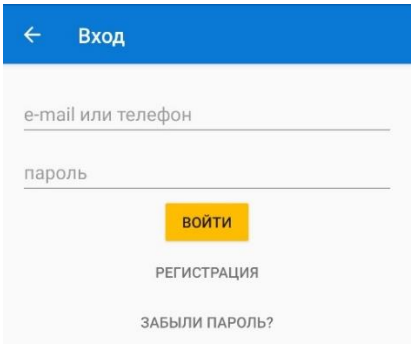


Рисунок 1.39 – Авторизация

Здесь карточки не столь богаты на функционал, что видно на рисунке 1.42, они имеют те же название и цену, однако нет кнопки добавить в корзину, кнопки «понравилось», рейтинга товара.

Выберем любой товар и посмотрим, как выглядит страница товара. На рисунке 1.43 видно, что состав также стандартен: фото, цвета, размеры.

Нажав добавить в корзину товаров, соответственно добавляется туда и появляется возможность перейти туда.

Перейдя на страницу корзины, видно довольно простое заполнение, что видно на рисунке 1.44. Имеется три пункта: адрес доставки, способ доставки и способ оплаты. Также сам товар в заказе и стоимость к оплате, комментарий и кнопка подтверждения товара.

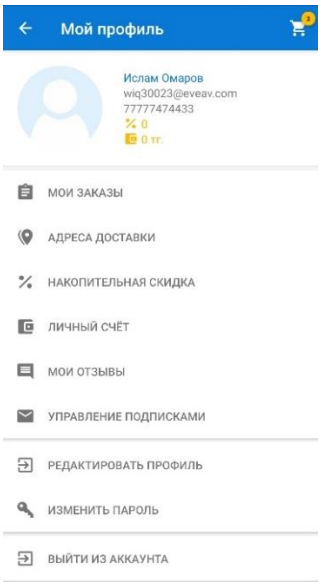


Рисунок 1.40 – Мой профиль

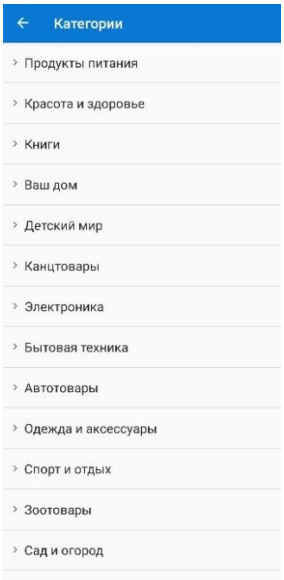
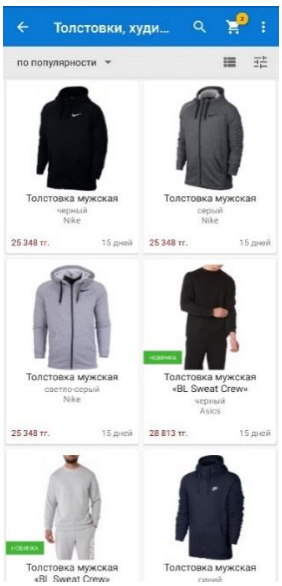


Рисунок 1.41 – Категории



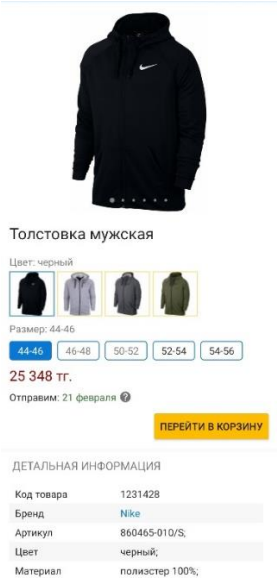


Рисунок 1.42 – Каталог товаров Рисунок 1.43 – Карта товара

Здесь нет пошагового заполнения данных, выбора различных опций и прочего. Просто и незамысловато. После подтверждения заказа необходимо будет ввести код, который придет на номер и после успешного заполнения кода заказ успешно оформляется и остается ожидать доставки.

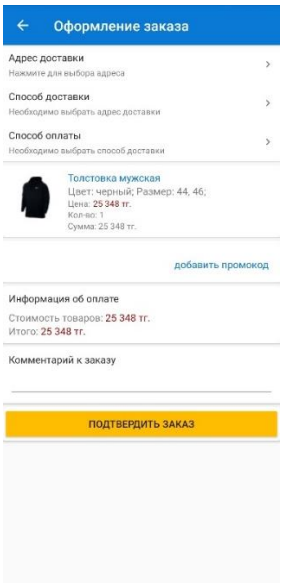


Рисунок 1.44 – Корзина

Рассмотрев аналогичные приложения, можно уже представить какой вид оно будет иметь и какой минимальный функционал должен быть у приложения для комфортной работы.

1.2. Структура и функциональные обязанности ИП “Unimag”

Организационная структура ИП «Unimag» представлена на рисунке 1.45.

Генеральный директор. В его подчинении находятся все подразделения, руководители. Определяет перспективы и принимает решение в какую сторону производить развитие предприятия.

Директор магазина - обеспечивает высокий уровень поддержания торгового зала, приём и расстановку продавцов, изучает и обобщает итоги работы продавцов. Анализирует причины увеличения и спадов продаж. Выявляет нарушения трудовой дисциплины.

Менеджер магазина - составляет заказ на товар, консультирует онлайн клиентов, ведут работу с покупателями, занимается претензионными клиентами.

Продавцы - ведут работу напрямую с клиентами, помогают определиться с выбором одежды, рассказывают о ее свойствах, консультируют обо всех нюансах одежды, провожают на кассу, оформляют продажу, пробивают чек.

Менеджер по закупкам - контроль наличия товара, своевременное обеспечение и пополнение товара. Выбор лучших поставщиков и цен. Контроль движения грузов и сроков поступления. Напрямую коммуницирует с директором магазина.

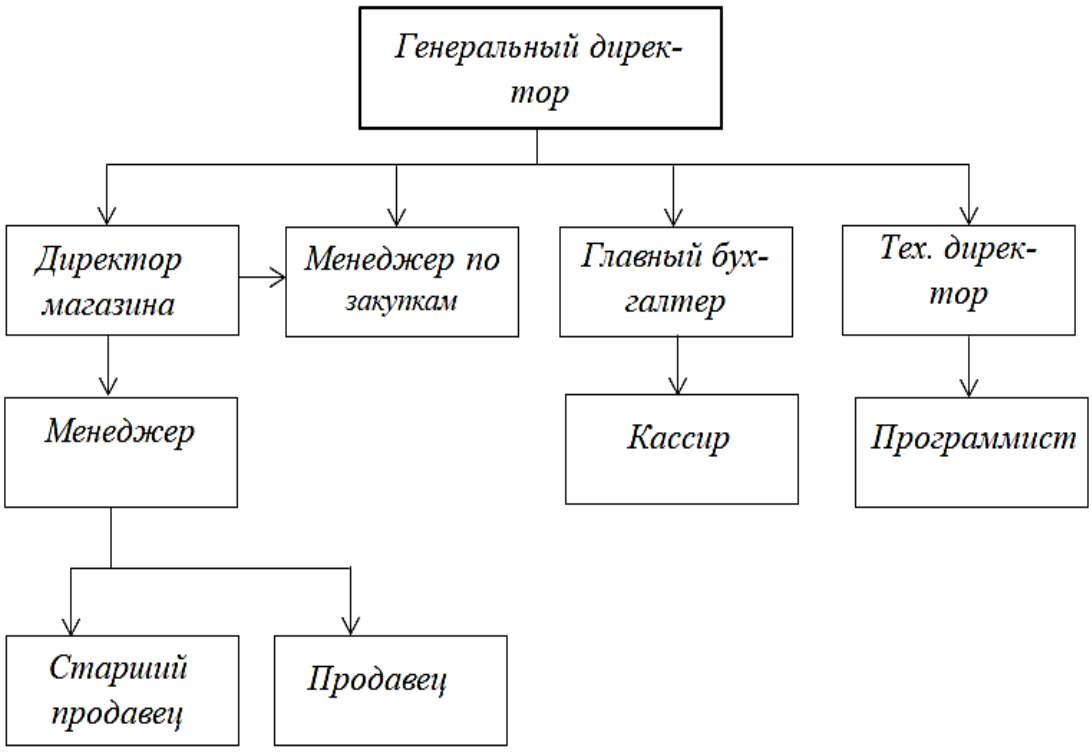


Рисунок 1.45 – Организационная структура

Технический директор - представляет руководству предложения по совершенствованию своей работы и работы компании. Сообщает руководству

предприятия о всех выявленных в процессе своей деятельности недостатках и вносит предложения по их устранению.

Программист - обеспечивает повседневную работу всех программ. Занимается устранением ошибок, разрабатывает сервисы, поддерживает аппаратное обеспечение в работоспособном состоянии.

Главный бухгалтер - осуществляет контроль за соблюдением порядка оформления первичных и бухгалтерских документов.

Бухгалтер - занимается расчетом платежных обязательств, расходованием фонда заработной платы, проведением инвентаризаций, товарно-материальных ценностей и денежных средств.

Кассир - осуществляет денежные операции, принимает и выдает деньги клиенту. Определяет подлинность купюр, ведёт учет и контроль денежной наличности. ведет кассовую отчетность. Проводит инкассацию денег и передает их инкассатору.

1.3 Анализ требования к разрабатываемому приложению

E-commerce проект подразумевает многопользовательский доступ. Исходя из этого следует разграничить роли для оптимального ведения бизнеса. Выделить можно такие роли как:

Администратор. Вносит сотрудников в систему и наделяет своей зоной ответственности, контролирует работу сотрудников. Может выполнять функции практически всех ролей.

Менеджер по заказам. Обрабатывает заказы и ведет их с момента принятия до выдачи покупателю. Отвечает на вопросы пользователей.

Менеджер базы. Вносит товар и необходимую информацию в базу. Создает новые категории при необходимости.

Менеджер по закупкам. Пополняет ассортимент в соответствии с листом заказов, в том числе из категории индивидуальных заказов.

Пользователь. Просматривает каталог, заполняет корзину, изменяет профиль, оформляет заказ.

Назначение ролей не только разгружает и оптимизирует человеческие ресурсы, но также несет в себе функцию защиты от несанкционированного доступа к данным, которые не находятся в их полномочиях.

Действия, которые разрешены для роли представлены в таблице 1.

Конвертировав таблицу 1 в диаграммы вариантов использования, можно увидеть какие возможны действия для определенной роли. Визуализация помогает в начале проектирования приложения тем, что наглядно показывает основной вектор разработки и задачи, которые необходимо выполнить.

На рисунке схематично изображены актеры же и блоки использования. Актеры – это субъекты, которые оказывают влияние на объекты, которыми являются блоки использования, обозначающие каким действием, можно оказывать влияние на внешнюю среду.

Таблица 1 – Разрешенные действия

Объект доступа	Действие	Роли
Сотрудники	Манипулирование списком сотрудников	Администратор
Финансовая сводка за период	Просмотр	Администратор
Заказы	Просмотр	Менеджер по заказам
Заказы	Изменение статуса	Менеджер по заказам
Каталог	Пополнение базы	Менеджер базы
Каталог	Изменение товара	Менеджер базы
Ассортимент	Контроль наличия	Менеджер по закупкам
Ассортимент	Пополнение запасов	Менеджер по закупкам
Лист заказов	Просмотр товаров индивидуального заказа	Менеджер по закупкам
Лист заказов	Пополнение товаров индивидуального заказа	Менеджер по закупкам
Каталог	Просмотр	Пользователь
Корзина	Пополнение	Пользователь
Корзина	Оформление заказа	Пользователь
Избранное	Добавление товара	Пользователь
Профиль	Просмотр и редактирование	Все

На рисунке 1.46 изображена диаграмма для администратора, тут изображен тот минимум, который можно представить на начальном этапе проектирования.

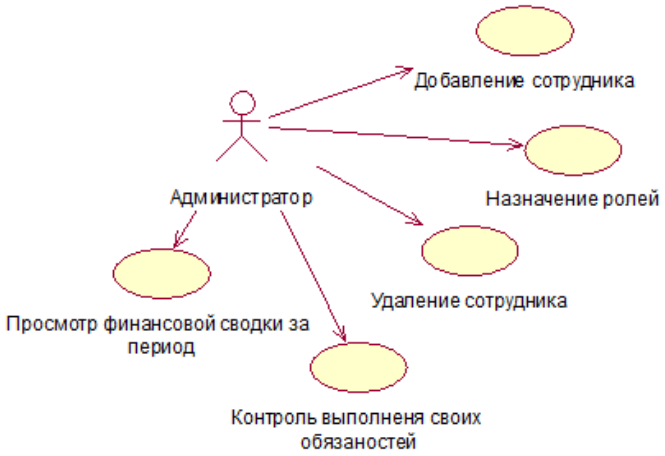


Рисунок 1.46 – Варианты использования Администратора

На рисунке 1.47 представлены деятельности менеджеров приложения.

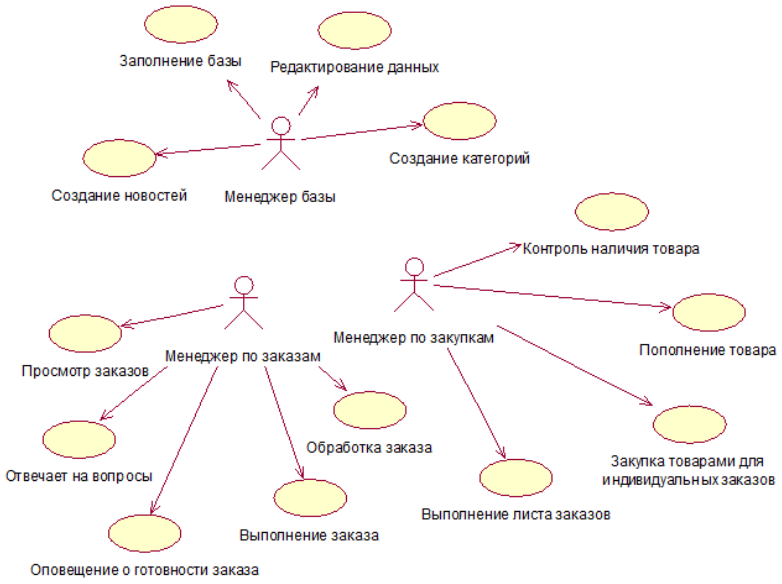


Рисунок 1.47 – Диаграмма для группы Менеджеров

Действия, которые может выполнять Пользователь представлены на рисунке 1.48.

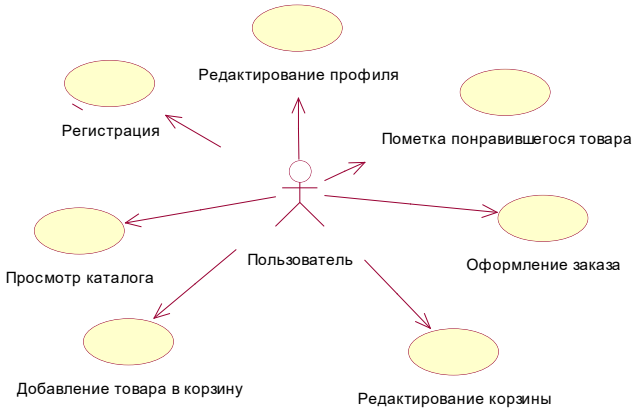


Рисунок 1.48 – Диаграмма действий для Пользователя

1.4 Описание структуры и основных модулей системы

Для упрощения проектирования будущей системы будет плюсом представить ее в виде комплекса модулей, которые взаимодействуют друг с другом. Клиентская часть должна быть дружелюбной, простой, красивой и эффективной. Она будет иметь такие модули как:

- модуль входа/регистрации. Это точка входа в систему и доступ к полноценному функционалу;
- модуль «Каталог». Модуль, который позволяет ознакомиться со всем ассортиментом, имеющимся в магазине. Начинается данный раздел обязательно с выбора категории;

- модуль «Корзина». Позволяет добавлять и редактировать свое наполнение и проводить приобретение товара;
- модуль «Профиль». Хранит некоторые данные о пользователе, доступ к которым имеется у администратора приложения. Эти данные пользователь может изменять по своему усмотрению;
- модуль уведомлений. Несет в себе информирующую роль, раздел который уведомляет об изменении статуса заказа, проходимых скидках и акциях, а также различных новостях;
- модуль «Избранное». Наличие такого модуля подразумевает что есть возможность добавлять любой товар в этот раздел.

1.5 Функциональная схема мобильного приложения

Исходя из проделанной ранее работы, можно описать следующую функциональную структуру мобильного приложения путем выделения в данной ИС подсистем:

- подсистема «Каталог». Основные функции: добавление/изменение существующих категорий товаров, включающих следующие параметры: название категории, порядком вывода категорий каталога, изображение категории товара, родительская категория каталога (при необходимости);
- подсистема «Корзина». Основные функции: в корзине покупатель может работать с изделиями: просматривать, изменять варианты и количество, удалять, добавлять; корзина должна выдавать стоимость каждого товара отдельно, а также общую сумму товара, способ доставки, способ оплаты товара и подтверждение заказа;
- подсистема «Уведомления». Основные функции: донесения информации до пользователей о различных событиях;
- подсистема «Профиль». Основные функции: предоставление личных данных пользователя, понравившиеся товары, заказы, статус пользователя (для сотрудников предприятия), смена пароля и информации, написанной ранее;
- подсистема «Управление заказами». Основные функции: просмотр и изменение статуса заказа, оплаты, места доставки и товаров, имеющихся в заказе, полученных через интерфейс мобильного приложения;
- подсистема «Управление товарами». Основные функции: добавление, изменение существующих товаров, включающее управление следующими параметрами: категория, название, изображение, описание, стоимость.

На рисунке 1.49 представлена функциональная схема мобильного приложения

Рисунок 1.49 – Функциональная схема

1.6 Постановка задачи

Проведя анализ деятельности магазина ИП “Unimag”, было принято решение о необходимости разработки мобильного приложения, которое оптимизирует деятельность магазина, а именно:

- ведение коммерческой деятельности в режиме 7/24;
- упрощение процесса реализации продукции магазина;
- повышение престижа магазина;
- администрирование каталога с помощью смартфона;
- поддержка клиентов в онлайн режиме.

Цель дипломного проекта разработать мобильное приложения для магазина “Unimag” с использованием современных программных инструментов: Flutter, MySQL. Для реализации данной цели необходимо выполнить следующие задачи:

- на основе анализа ПО мобильных приложений для e-commerce зарубежных и отечественных компаний выполнить их сравнительную характеристику, изучить структуру и назначение предприятия заказчика;
- обосновать выбор программных инструментов и разработать клиент-серверную архитектуру программного приложения, спроектировать диаграммы взаимодействия модулей приложения, смоделировать и спроектировать базу данных и выполнить программную реализацию мобильного приложения;
- рассчитать экономическую часть проекта и разработать мероприятия по охране труда и безопасности жизнедеятельности на предприятии;
- протестировать мобильное приложение “Unimag” и внедрить в производство.

2. Проектирование базы данных

2.1. Обоснование и выбор программного обеспечения для разработки мобильного приложения

В качестве системы управления базами данных была выбрана СУБД MySQL.

MySQL - компактный многопоточный сервер баз данных. MySQL характеризуется большой скоростью, устойчивостью и легкостью в использовании. Идеальное решение для малых и средних приложений.

Одной из существенных причин выбора является то, что MySQL имеет высокую скорость обработки данных из-за того, что ранее СУБД создавалась для обработки данных в промышленных масштабах. Также неоспоримым фактом является способ ее распространения, по GNU «General Public License», которой наделяются все «open-source» ПО.

Одной из особенностей является поддержка SQL. Этот аспект обеспечивает высокий уровень кроссплатформенности данных и кода, созданных на MySQL. Что позволяет спокойно перенести созданную ранее БД в любую другую СУБД, также поддерживающую SQL и применять хранимые процедуры, триггеры и запросы из этих платформ.

В MySQL существует система привилегий, что позволяет выдавать выбранной учетной записи права с определенным диапазонном действий с хранимыми данными. Тем самым пользователь получает данные из БД, система цела, неизменна и доступна для всех.

Еще одним преимуществом является хэширование паролей. Хэширование обеспечивает высокий уровень устойчивости к взлому. В связи с этим в MySQL восстановить пароль root является крайне сложной задачей.

Еще одним немаловажным фактором является выбор хостинга. Крайне тяжело найти оптимального поставщика, поддерживающего иные СУБД, а так как ранее было сказано, что MySQL очень популярен, то проблем с выбором хостинга, использующего данную СУБД не составит проблем.

Visual Studio Code – редактор кода, который может работать на Windows, Linux, OS X. Простой в освоении, имеет удобный интерфейс, наделен всеми функциями для написания приложений под Flutter. Правда придется до конфигурировать среду для последующего использования. Необходимы SDK Tools, Android Studio, плагин, который дает возможность разработки с помощью Flutter на языке Dart. В нем имеется возможность установки дополнительных расширений для удобства пользователя.

VS Code это неполноценная IDE, но пользуется большой популярностью у многих разработчиков, правда web. Имеется возможность интеграции с Git и отладчик. Имеет не слишком простой, но дружелюбный интерфейс.

Достоинства утилиты:

- редактор поддерживает работу со многими языками программирования, нужно лишь подключить соответствующие расширения;

- редактор кода, с которым удобно работать;
- изменения цветовой темы среды, мелочь, но приятно
- встроенные подсказки IntelliSence, на каком бы языке код не писали, всегда будет высвечиваться контекстное меню с вариантами кода;
- большая библиотека с расширениями и компонентами для разработки ПО;
- средства для тестирования и отладки каждого элемента приложения;
- для «зеленых» разработчиков имеется руководство по использованию редактора, размещенное на официальном сайте.
- Недостатки/спорные моменты
- все же он не является полноценной IDE;
- невозможно компилировать/интерпретировать созданные приложения;
- изначально он довольно функциональный редактор, но тем не менее без расширений он не более чем редактор.

Разработка приложения будет проводиться с помощью фреймворка от Google – Flutter.

Flutter относительно молодая, но многообещающая платформа, интересная тем, что позволяет создавать приложения с простотой веб-проектов, но с производительностью нативных приложений. Достигается это путем нескольких техник:

- не использует JavaScript ни в каком виде, в качестве языка программирования Flutter выбрали Dart, который компилируется в бинарный код, за счет чего достигается скорость выполнения операций сравнимая с Objective-C, Swift, Java, или Kotlin.
- Flutter не использует нативные компоненты, опять же, ни в каком виде, так что не приходится писать никаких прослоек для коммуникации с ними. Вместо этого, подобно игровым движкам, он отрисовывает весь интерфейс самостоятельно. Кнопки, текст, медиа-элементы, фон — все это отрисовывается внутри графического движка в самом Flutter.
- для построения UI во Flutter используется декларативный подход, вдохновленный веб-фреймворком ReactJS, на основе виджетов. Для еще большего прироста в скорости работы интерфейса виджеты перерисовываются по необходимости — только когда в них что-то изменилось;
- в фреймворк встроен Hot-reload, привычный для веба.

В качестве СУБД, как было указано в пункте описания информационного обеспечения, будет использована MySQL. MySQL – это одна из самых популярных и самых распространенных СУБД. MySQL отличатся хорошей скоростью работы, надежностью, гибкостью. Работа с ней, как правило, не вызывает больших трудностей.

2.2 Концептуальная модель

В настоящее время практически любой проект имеет базу данных, в которой он накапливает и хранит различного типа и важности информацию. Для ее построения необходимо предварительно провести анализ проекта или же предприятия, для которого разрабатывается программа с базой данных. Под анализом имеется ввиду структура проекта или компании, определение целей разработки, вырабатывается ряд требований, который должен выполнять проект. Очертив границы проекта процесс разработки упрощается. И первым делом конечно же разрабатывается механизм накопления, хранения и течения данных – это база данных.

Концептуальная модель данных является описанием основных сущностей и отношений между ними. Концептуальная модель схематично отражает рассматриваемую предметную область, для которой планируются работы по построению хранилища данных. При проектировании концептуальной модели структурируют данные и выявляют взаимосвязи между ними, без рассмотрения особенностей реализации и вопросов эффективности обработки. Для разработки концептуальной модели системы нужно выделить информационные объекты [3].

Для данного проекта были проведены работы по анализу и выделены следующие сущности:

- товар – фактически основной объект в любой коммерческой деятельности;
- цвет – это одна из опций товара;
- размер – опция;
- материал – опция;
- категория – один из видов сортировок по типу и применению;
- заказ – таблица, в которой отображены все заказы, сделанные через приложение;
- пользователь – таблица, которая содержит личные данные клиентов;
- адрес – содержит так же личную информацию пользователя;
- сотрудники – таблица аналогична пользователям, только ее изменяет администратор;
- роль – это таблица содержит роли, существующие в системе;
- новости – здесь будут записи о различных событиях магазина;
- статус – возможность изменять состояние заказов от «Обрабатывается» до «Заказ выполнен»;
- оплата или же тип оплаты, фактически опция для менеджера и системы;
- тип доставки – также является опцией для заказа.

На рисунке 2.1 представлена концептуальная модель базы данных.

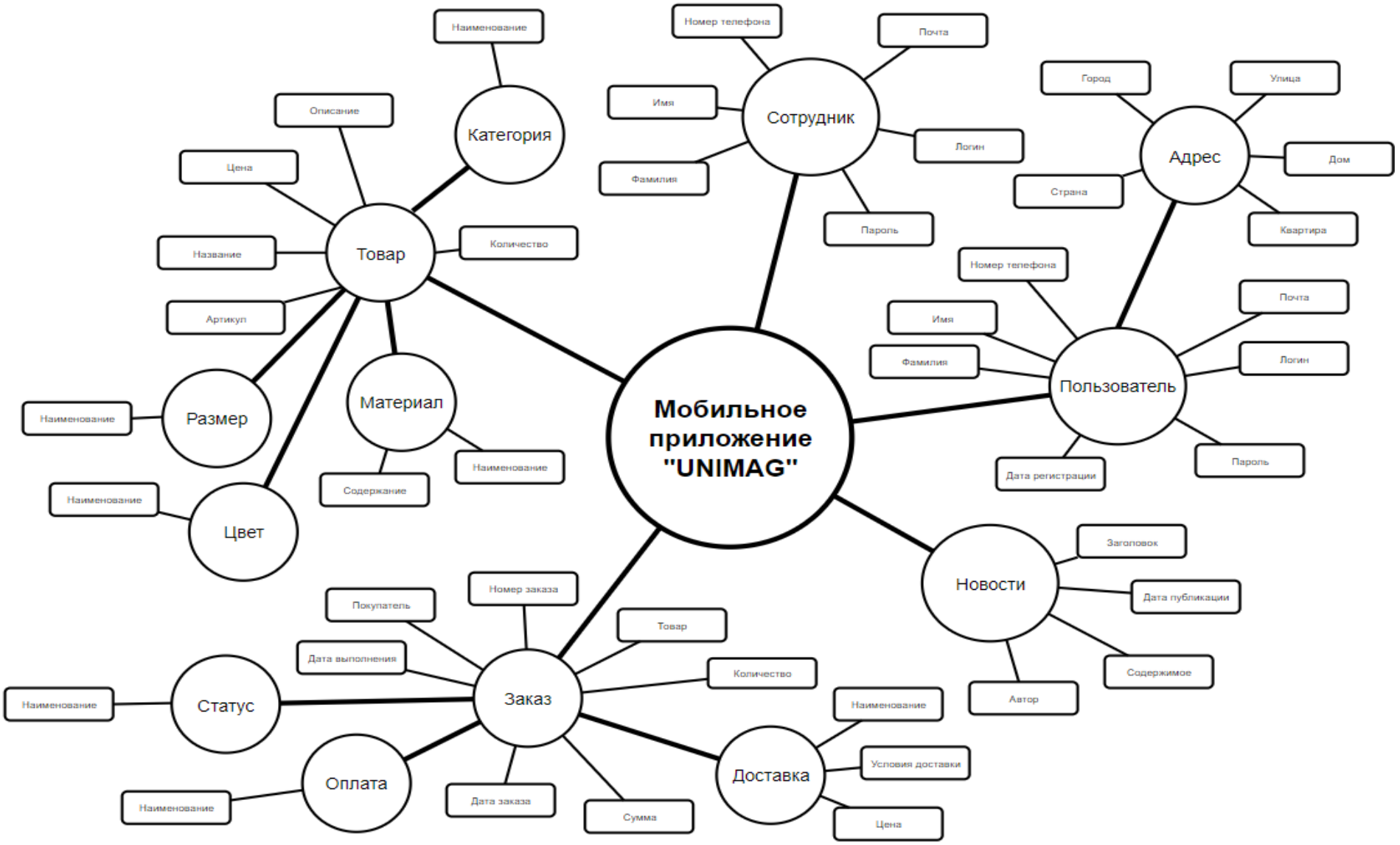


Рисунок 2.1 – Концептуальная модель

2.3 Логическая модель

Логическая модель показывает структуру исследуемой предметной области [4]. На ней изображены сущности исследуемой области, их атрибуты, а также показывает взаимоотношения между сущностями. Атрибутами являются свойства этих сущностей [5]. В таблице 2.1 приведен пример структуры сущности «Пользователь», в таблице 2.2 – сущность «Адрес».

Таблица 2.1 – Сущность «Пользователь»

id Пользователя	Идентификатор пользователя
Логин	Имя пользователя в системе
Пароль	Для доступа к аккаунту
Фамилия	Фамилия пользователя
Имя	Имя пользователя
Номер телефона	Номер телефона пользователя
email	Электронная почта пользователя
Дата регистрации	Дата регистрации пользователя

Таблица 2.2 – Сущность «Адрес»

id Адреса	Идентификатор адреса
Страна	Страна проживания пользователя
Город	Город, где проживает пользователь
Улица	Улица, где проживает пользователь
Дом	Номер дома пользователя
Квартира	Номер квартиры (опционально)

Связи определяют зависимости между сущностями. В реляционной базе данных связи осуществляются посредством внедрения внешнего ключа в таблицу. К примеру, в «Пользователь» перейдет первичный ключ из таблицы «Адрес». Таблица «Пользователь» изменила содержание, измененный вид представлен в таблице 2.3.

Таблица 2.3 – Сущность «Пользователь», измененная

id Пользователя	Идентификатор пользователя
…	…
id Адреса	Идентификатор адреса

Схема построена с помощью программы ERWin Data Modeler позволяющей моделировать бизнес-процессы. Главная задача ERWin Data Modeler моделирование данных [7]. Модели, которые строит данная программа разделяются на две категории: логические и физические.

Логическая модель данной работы проиллюстрирована на рисунке 2.2.

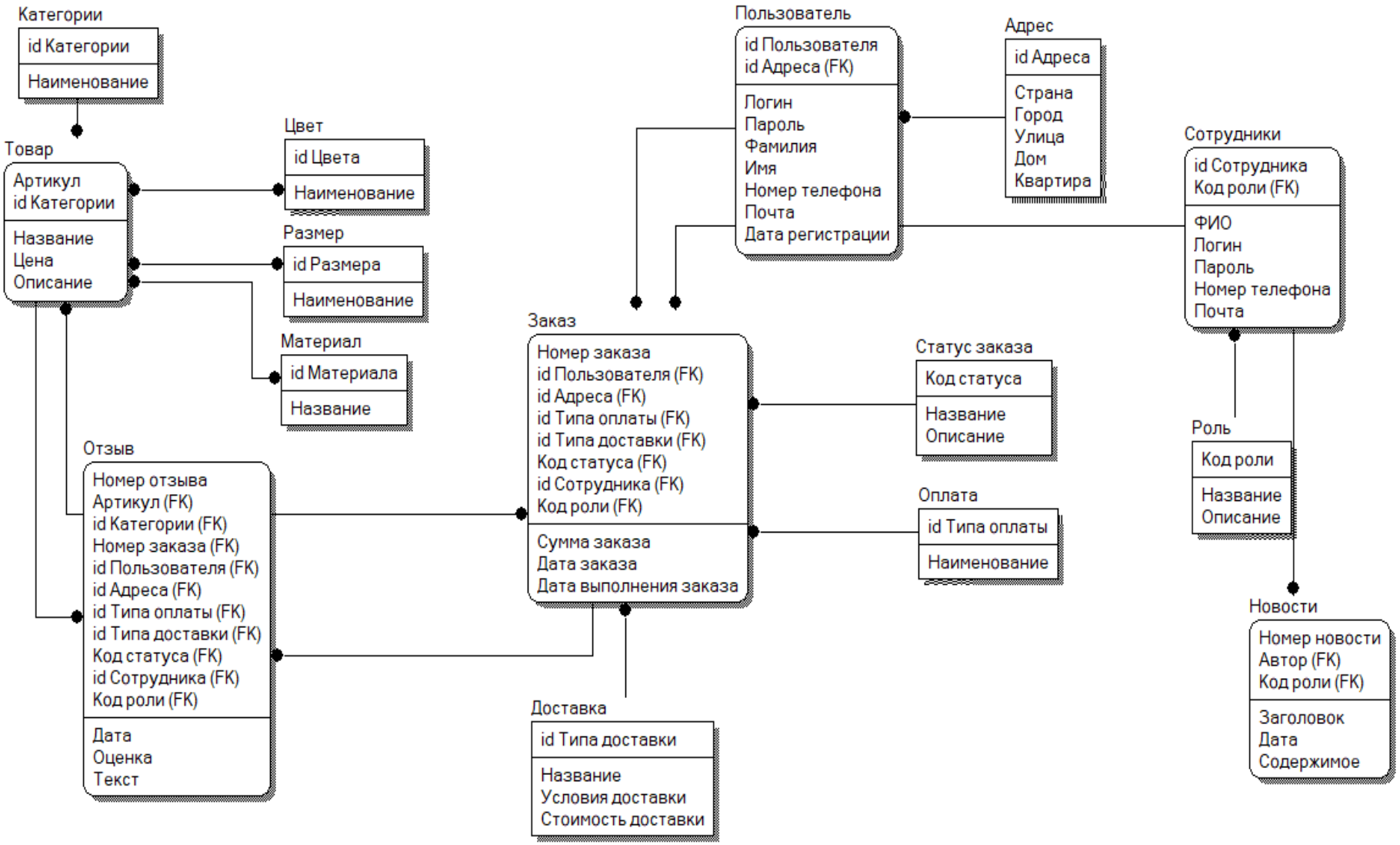


Рисунок 2.2 – Логическая модель

2.4 Физическая модель

Физическая модель данных описывает реализацию объектов логической модели на уровне объектов конкретной базы данных. Она описывает первичные ключи и внешние ключи таблиц, наименование таблиц, ее колонок и также нужно указать тип данных для колонок [6].

В прошлых моделях, связи указывались в виде соответствующих линий: «один-к-одному», «один-ко-многим» или «многие-ко-многим». Так нужно чтобы изобразить связи таблиц в общих чертах и этого хватает, но на этом уровне нужно продумать эти связи, особенно «многие-ко-многим» так как это реализуется не так просто. Для воссоздания связи «многие-ко-многим» необходимо использовать промежуточные таблицы, а чем заполнять решает проектировщик.

Для построения этой модели была использована программное обеспечение MySQL Workbech.

MySQL Workbech – представляет собой инструмент для визуального проектирования баз данных, интегрирующий проектирование, моделирование, создание и эксплуатацию БД в единое бесшовное окружение для системы баз данных MySQL [8].

Workbench очень удобен для реализации физической модели из-за возможности собрать свою базу данных визуально, что непременно удобнее и приятнее, чем список таблиц, а также она будет иметь функционал рабочей базы данных. Так же собрав визуальную физическую модель, Workbench может ее экспортировать в привычное представление [9].

Также он позволяет проводить манипуляции с базами данных с помощью встроенного SQL-редактора, такие как:

- создание таблиц;
- редактирование существующих;
- их заполнение и редактирование;
- работа с ключами, полями связями;
- создание запросов различной степени сложности;

Еще одним плюсом является подключение и синхронизация со схемой базы данных на локальном или удаленном сервере. Что существенно упрощает развертку рабочей базы данных на рабочем сервере.

Из-за того, что база данных должна быть доступна из любой точки, она должна быть расположена на хостинге. Все хостинги поддерживают СУБД MySQL, но для работы с ней используется веб-интерфейс PhpMyAdmin.

PhpMyAdmin имеет тот же функционал что и описанный Workbench, за исключением визуального проектирования.

Так как создана БД на локальном компьютере его следует импортировать на сервер. Делается это путем экспорта из Workbench всей базы данных и импорта файла в PhpMyAdmin.

Схема физической модели представлены на рисунке 2.3.

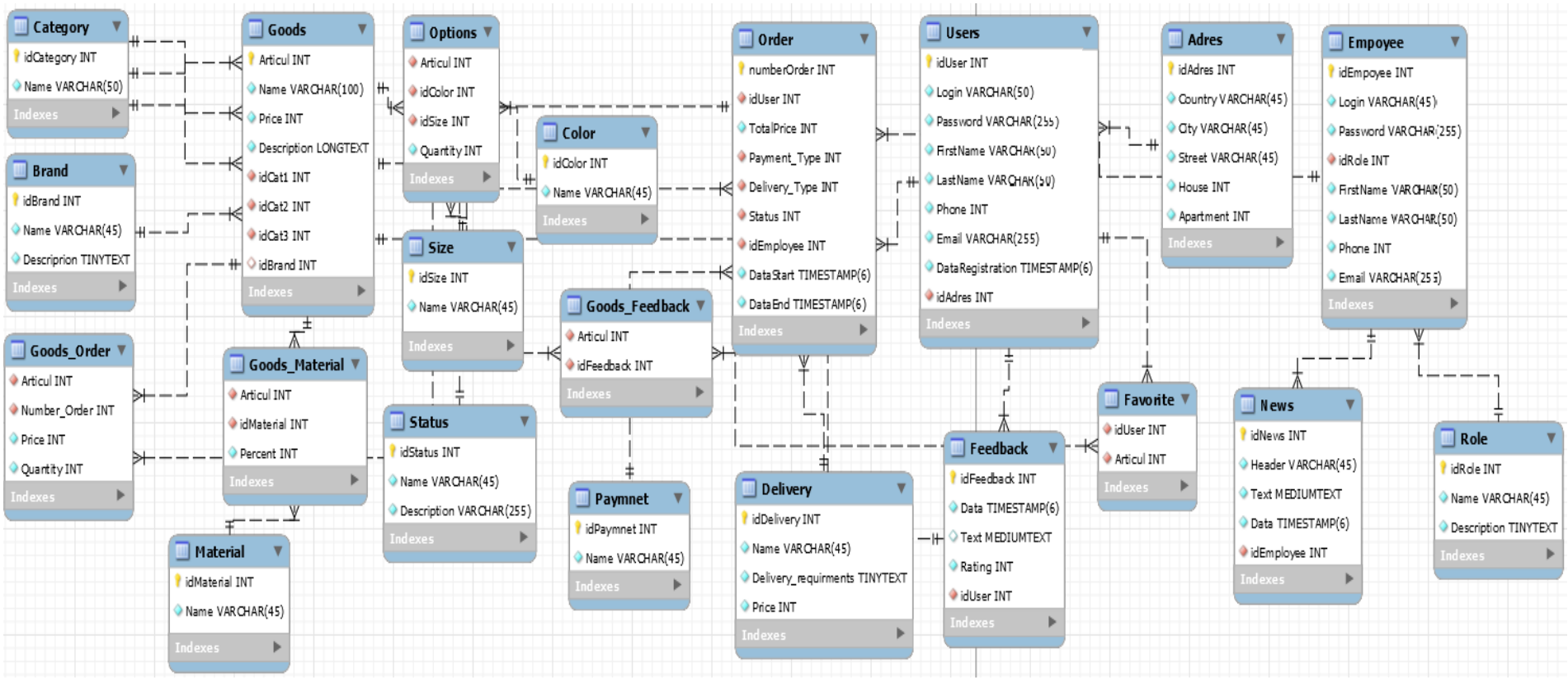


Рисунок 2.3 – Физическая модель

3 Проектирование и разработка мобильного приложения

3.1 Проектирование интерфейса

Перед тем как приступить к непосредственной разработке мобильного приложения, необходимо придать ему вид. На этом этапе отрисовываются основные экраны, детали интерфейса, формируется карта экранов. Если имеется готовый интерфейс, то это упрощает задачу.

В данном проекте будет выполнено проектирование интерфейса с нуля с помощью программы Adobe XD.

Adobe Experience Design (Adobe XD) — программа для разработки интерфейсов от Adobe Systems. Поддерживает векторную графику и веб-верстку и создает небольшие активные прототипы [10].

В процессе разработки интерфейс может изменить визуальную составляющую и быть отличной от прототипа интерфейса.

Главная страница мобильного приложения будет выглядеть как список категорий, для упрощения проекта и перехода пользователя к непосредственному ознакомлению с товарами. На рисунке 3.1 представлен прототип главной страницы.

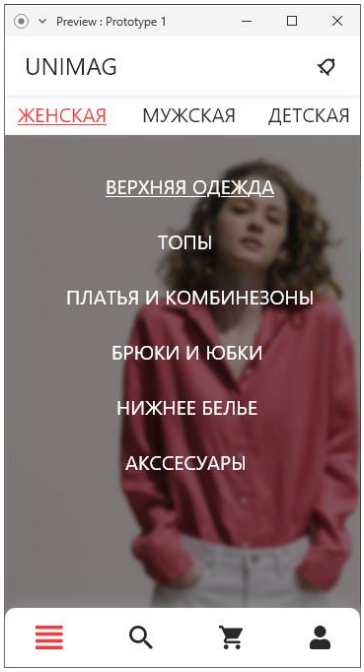


Рисунок 3.1 – Главная страница

После выбора заинтересовавшего раздела осуществляется переход к категориям этого раздела. На рисунке 3.2 представлен прототип экрана категорий.

После выбора соответствующей категории выполняется переход к самому основному модулю приложения типа интернет-магазин – каталог, который изображен на рисунке 3.3.

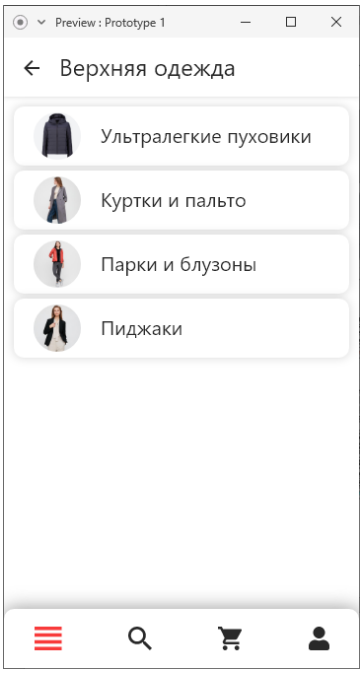


Рисунок 3.2 – Категории

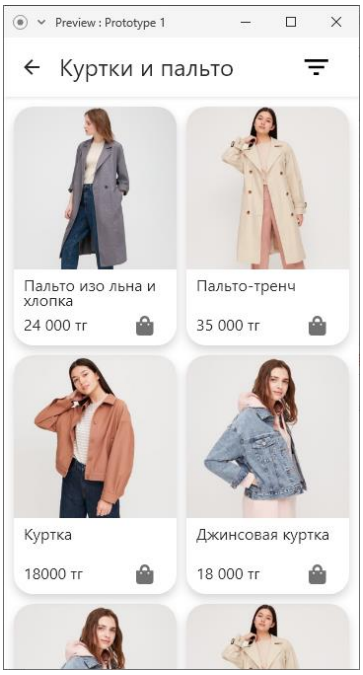


Рисунок 3.3 – Каталог товаров

Нажав на одну из карточек товара, будет осуществлен переход на страницу этого товара с более подробной информацией. Детальная страница изображена на рисунке 3.4.

Рисунок 3.4 – Детали товара

Здесь представлена такая информация как: размеры, цвет, описание, материалы, используемые в товаре. Также возможно добавить товар в избранное, в корзину и перейти в корзину. Далее идет корзина. Прототип страницы корзины представлен на рисунке 3.5.

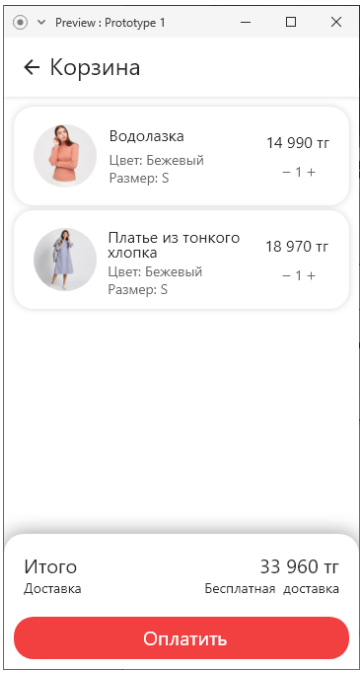


Рисунок 3.5 – Корзина

После оплаты товара, товар из корзины отображается на странице заказов, в которую можно попасть через страницу профиля.

С прототипом страницы профиля можно ознакомиться на рисунке 3.6, на ней представлены такие варианты как: заказы, избранное, раздел вопросов и ответов и кнопка выход.

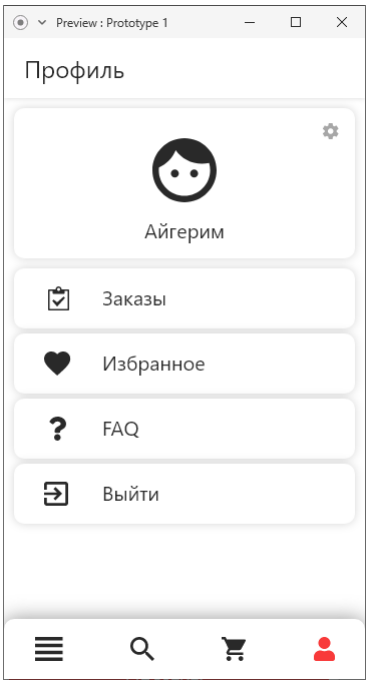


Рисунок 3.6 – Профиль

Прототип страницы с заказами изображен на рисунке 3.7. Он имеет весьма простой вид, но включает все необходимое.

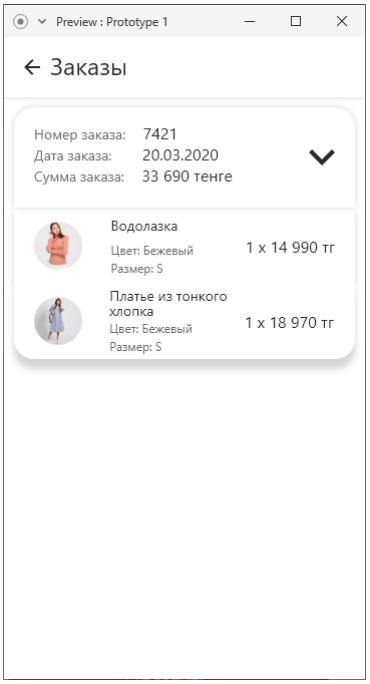


Рисунок 3.7 – Заказы

Раздел избранное представляет собой простой список товаров, которые были помечены как понравившиеся товары. Прототип страницы с понравившимися товарами представлен на рисунке 3.8.

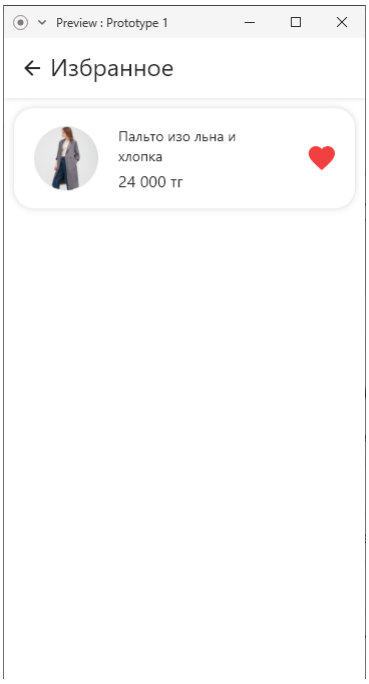


Рисунок 3.8 – Избранное

Последний раздел с FAQ должен выглядеть как смесь товаров, но со структурой вопрос-ответ. Прототип можно рассмотреть на рисунке 3.9.

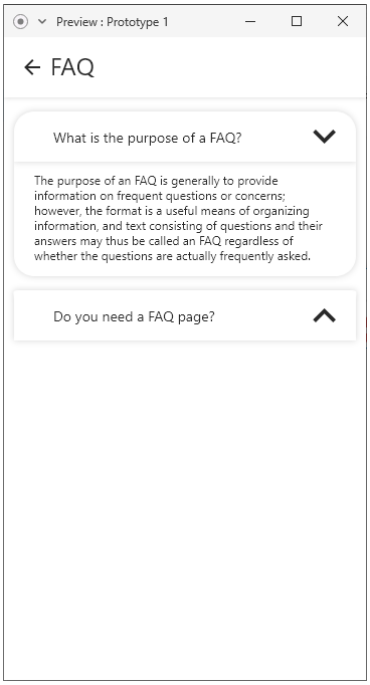


Рисунок 3.9 – FAQ

И наконец прототипы страниц регистрации и авторизации представлены на рисунке 3.10 и 3.11. Они представляют из себя простую форму ввода.

Рисунок 3.10 – Авторизация

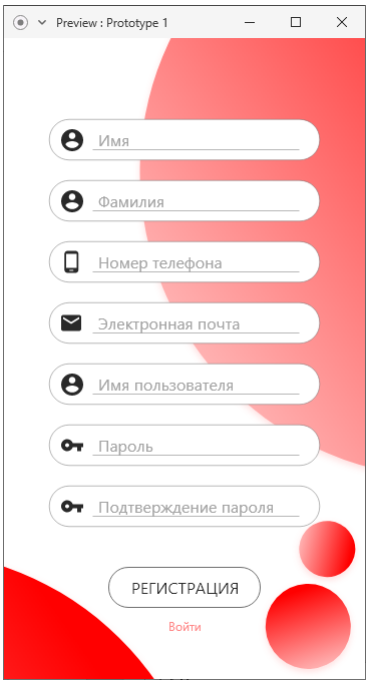


Рисунок 3.11 – Регистрация

Имея макет, на который можно опираться в процессе разработки интерфейса можно переходить к самой разработке.

3.2 Разработка серверной части

Для того чтобы мобильное приложение взаимодействовало с базой данных, необходимо разработать систему для работы с ней. Серверная сторона или же backend, необходима чтобы выполнять логику проекта, а клиентская сторона или же front-end, чтобы выводить туда информацию в приятном виде для пользователя.

Backend проекта будет выполняться с помощью языка PHP, так как он простой, удобный и оптимально подходит по техническим условиям хостинга, а также может взаимодействовать напрямую с базой данных. Dart также может взаимодействовать с базой данных и использоваться в backend, но в Казахстане довольно сложно найти хостинг провайдера который поддерживает данный язык.

С помощью PHP будет написана служба для взаимодействия с базой данных и frontend. Эта служба будет своего API (Application Programming Interface или программный интерфейс приложения)[11]. API – описание способов, которыми одна компьютерная программа может взаимодействовать с другой. Этот способ значительно упростит процесс разработки, так как он будет всего лишь выдавать данные из базы кодировать их в формат JSON и по запросу предоставлять их приложению.

JSON (JavaScript Object Notation) – текстовый формат обмена данными, основанный на JavaScript, структурирован и легко читается. Синтаксис JSON-текста выглядит в формате «ключ:значение», где ключ – регистрозависимая строка, а значение в любой форме: строка, число, массив, дата и т.д.[12].

Итак, для того чтобы выполнять манипуляции с базой данных необходимо сначала установить соединение с ней. На рисунке 3.12 представлен отрезок кода для соединения с базой данных, это ключевой файл всего интерфейса.

```php
<?php
// конфигурации
$host = 'localhost';
$user = 'TestUser';
$pass = 'testuser';
$dbname = 'unimag';
// соединиение с базой
$conn = new mysqli($host, $user, $pass, $dbname);

if ($conn->connect_error) {
    die("Ошибка соединения: " . $conn->connect_error .
    "Ошибка #" . $conn->connect_errno);
}
```

Рисунок 3.12 – Соединение с базой данных

Первые 4 переменные составляют простые конфигурации для подключения: хост – устройство, которое предоставляет возможность взаимодействовать со службами, пользователь – пользователь базы данных,

пароль – пароль аккаунта пользователя, база данных – выбор нужной базы данных.

Следом переменная, содержащая результат соединения, далее идет условие проверяющее соединения на ошибки, если во время соединения произошла ошибка, то выполняется вывод ошибки, ее номер и отключения процесса. Если же ошибок не произошло, то можно работать с базой данных.

Для того чтобы предоставить мобильному приложению на вывод товары хранящиеся товары нужно выполнить запрос. Сначала нужно проверить корректность его вывода в PhpMyAdmin. На рисунке 3.13 представлен результат запроса, описанного ниже.

```
SELECT
productId as 'Код',
productArticul as 'Артикул',
productName as 'Название',
productDescription as 'Описание',
productPrice as 'Цена',
productImage as 'Изображение'
FROM product
```

Код	Артикул	Название	Описание	Цена	Изображение
1	419777	Ультралегкая пуховая куртка	Ультралегкая парка — теперь еще удобнее.	36000	https://www.uniqlo.com/ru/es
2	424633	Пальто изо льна и хлопка	Идеально для погоды, когда нужна легкая верхняя од...	24000	https://www.uniqlo.com/ru/es
3	426692	Куртка из хлопка	Очень универсальная и практичная.	12000	https://www.uniqlo.com/ru/es
4	424636	Жакет	Легкая и удобная куртка, которую также можно носит...	12000	https://www.uniqlo.com/ru/es
5	422913	Свитер	Сочетание хлопка и пряжи делает свитер легким и мя...	6000	https://www.uniqlo.com/ru/es
6	422922	Легкий кардиган	Легкий и практичный кардиган станет прекрасным доп...	12000	https://www.uniqlo.com/ru/es
7	422706	Толстовка с капюшоном	Свободный и лаконичный стильный силуэт.	12000	https://www.uniqlo.com/ru/es
8	425679	Блузка JW Anderson	Легкая удлиненная сорочка из денима для создания с...	18000	https://www.uniqlo.com/ru/es
9	429693	Футболка в рубчик	Прекрасно сядет на любую фигуру благодаря эластичн...	4800	https://www.uniqlo.com/ru/es
10	418910	Зауженные селвидж джинсы	Традиционный селвидж-деним с добавлением эластичны...	18000	https://www.uniqlo.com/ru/es
11	426288	Брюки KANDO	Эти функциональные брюки прекрасно подходят даже д...	18000	https://www.uniqlo.com/ru/es
12	424151	Шорты	Благодаря укороченному силуэту очень удобны и не с...	9000	https://www.uniqlo.com/ru/es
13	418475	Ultra stretch комплект: джемпер, брюки	Невероятно эластичный и мягкий материал, который т...	15000	https://www.uniqlo.com/ru/es
14	426431	Футболка AIRism	Мягкий и приятный материал надолго сохраняет ощуще...	4800	https://www.uniqlo.com/ru/es
15	423844	Кожаный ремень	Текстура и поверхность кожи ремня станут более выр...	12000	https://www.uniqlo.com/ru/es
16	426947	Платье JW Anderson	Платье с красивыми оборками по низу подола.	12000	https://www.uniqlo.com/ru/es
17	424497	Трикотажная толстовка	Джемпер классического кроя с дизайнерской отделкой...	9000	https://www.uniqlo.com/ru/es
18	418655	Комплект одежды: джемпер, брюки	Благодаря отличной эластичности прекрасно походит ...	9000	https://www.uniqlo.com/ru/es
19	423596	Брюки для девочек	Актуальные цвета и фасон красиво подчеркнут линию ...	3000	https://www.uniqlo.com/ru/es
20	424417	Шорты из твила	Стильные и удобные шорты.	4800	https://www.uniqlo.com/ru/es
21	422490	Трикотажная юбка	Мягкий материал не стесняет движений и придает изд...	3000	https://www.uniqlo.com/ru/es

Рисунок 3.13 – Вывод товаров

Теперь выполнить запрос с помощью PHP. На рисунке 3.14 представлен отрывок кода запроса на вывод товара.

Сначала идет конструкция require (), которая необходима, чтобы перед выполнением всего кода, включить файл, который находится между скобок в сценарий. Эта конструкция включает файл соединения с сервером, она нужна чтобы не дублировать код.

Далее идет запрос, который помещен в виде строки в переменную. Следом переменная, в которой выполняется запрос. Потом идет условие, в

котором идет сравнение количество строк, хранимых в результирующей переменной, если она содержит хотя бы одну строку, то выполняется конструкция в теле условия, там идет цикл, который будет переводить результат каждой строки в ассоциативный массив и присваиваться в переменную, которая в свою очередь помещается в список. Затем этот список кодируется в JSON формат. В конце идет закрытие соединения с базой, так как необходимые данные уже были получены.

```php
<?php
require('connect.php');

$sql = "SELECT
    productId as 'Код',
    productArticul as 'Артикул',
    productName as 'Название',
    productDescription as 'Описание',
    productPrice as 'Цена',
    productImage as 'Изображение'
    FROM product";

$result = $conn->query($sql);
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        $db_data[] = $row;
    }
    echo json_encode($db_data, JSON_UNESCAPED_UNICODE);
}
$conn->close();
```

Рисунок 3.14 – Вывод товаров

На рисунке 3.15 можно посмотреть, что вывел этот отрывок кода.

[{"Код":"1","Артикул":"419777","Название":"Ультралегкая пуховая куртка","Описание":"Ультралегкая парка — теперь еще удобнее.","Цена":"36000","Изображение":"https:\/\/www.uniqlo.com\/ru\/estore\/hmall\/test\/u0000000001652\/main\/first\/561\/1.jpg"},
{"Код":"2","Артикул":"424633","Название":"Пальто изо льна и хлопка","Описание":"Идеально для погоды, когда нужна легкая верхняя одежда.","Цена":"24000","Изображение":"https:\/\/www.uniqlo.com\/ru\/estore\/hmall\/test\/u0000000003319\/main\/first\/561\/1.jpg"},
{"Код":"3","Артикул":"426692","Название":"Куртка из хлопка","Описание":"Очень универсальная и практичная.","Цена":"12000","Изображение":"https:\/\/www.uniqlo.com\/ru\/estore\/hmall\/test\/u0000000002784\/main\/first\/561\/1.jpg"},
{"Код":"4","Артикул":"424636","Название":"Жакет","Описание":"Легкая и удобная куртка, которую также можно носить в качестве кардигана.","Цена":"12000","Изображение":"https:\/\/www.uniqlo.com\/ru\/estore\/hmall\/test\/u0000000002929\/main\/first\/561\/1.jpg"},

Рисунок 3.15 – JSON результат

Однако, для магазина не нужно выводить весь товар, тогда исчезнет надобность в категориях и будет сложно найти то, что действительно интересно. Чтобы приблизить к реалиям, следует выполнить запрос с некоторыми условиями. К примеру, список категорий доступные для женщин.

Код запроса представлен ниже, а результат запроса на рисунке 3.16.

```
SELECT cattype.catTypeId as 'Код пола',
cattype.catTypeName as 'Для кого',
catsuper.catSuperId as 'Код типа',
catsuper.catSuperName as 'Тип',
catsub.catSubId as 'Код категории',
catsub.catSubName as 'Категория'
FROM category
INNER JOIN cattype ON category.catTypeId = cattype.catTypeId
```

INNER JOIN catsuper ON category.catSuperId = catsuper.catSuperId
INNER JOIN catsub ON category.catSubId = catsub.catSubId
INNER JOIN product ON category.productId = product.productId
WHERE cattype.catTypeId = 1
ORDER BY cattype.catTypeId
Здесь был использован многотабличный запрос.

Код пола	Для кого	Код типа	Тип	Код категории	Категория
1	Женщинам	1	Верхняя одежда	1	Пуховики
1	Женщинам	1	Верхняя одежда	3	Пальто
1	Женщинам	1	Верхняя одежда	2	Куртки
1	Женщинам	1	Верхняя одежда	4	Пиджаки
1	Женщинам	2	Топы	5	Свитеры
1	Женщинам	2	Топы	6	Кардиганы
1	Женщинам	2	Топы	7	Толстовки
1	Женщинам	2	Топы	10	Футболки
1	Женщинам	2	Топы	9	Блузки

Рисунок 3.16 – Категории доступные для женщин

PHP код на рисунке 3.17 аналогичен предыдущему коду за исключением содержимого переменной. Результат представлен на рисунке 3.18.

```php
<?php
require('connect.php');

$sql = "SELECT cattype.catTypeId as 'Код пола',
cattype.catTypeName as 'Для кого',
catsuper.catSuperId as 'Код типа',
catsuper.catSuperName as 'Тип',
catsub.catSubId as 'Код категории',
catsub.catSubName as 'Категория'
FROM category
INNER JOIN cattype ON category.catTypeId = cattype.catTypeId
INNER JOIN catsuper ON category.catSuperId = catsuper.catSuperId
INNER JOIN catsub ON category.catSubId = catsub.catSubId
INNER JOIN product ON category.productId = product.productId
WHERE cattype.catTypeId = 1";

$result = $conn->query($sql);
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        $db_data [] = $row;
    }
    echo json_encode($db_data, JSON_UNESCAPED_UNICODE);

}
$conn->close();
?>
```

Рисунок 3.17 – Вывод категорий

[{"Код пола":"1","Для кого":"Женщинам","Код типа":"1","Тип":"Верхняя одежда","Код категории":"1","Категория":"Пуховики"},
{"Код пола":"1","Для кого":"Женщинам","Код типа":"1","Тип":"Верхняя одежда","Код категории":"3","Категория":"Пальто"},
{"Код пола":"1","Для кого":"Женщинам","Код типа":"1","Тип":"Верхняя одежда","Код категории":"2","Категория":"Куртки"},
{"Код пола":"1","Для кого":"Женщинам","Код типа":"1","Тип":"Верхняя одежда","Код категории":"4","Категория":"Пиджаки"},

Рисунок 3.18 – JSON-текст

Следующим будет выбор товаров из категории женской верхней одежды.
SELECT

```
product.productName as 'Наименование',
product.productDescription as 'Описание',
product.productMaterial as 'Материал',
product.productPrice as 'Цена',
product.productImage as 'Изображение'
FROM category
INNER JOIN cattype ON category.catTypeId = cattype.catTypeId
INNER JOIN catsuper on category.catSuperId = catsuper.catSuperId
INNER JOIN catsub on category.catSubId = catsub.catSubId
INNER JOIN product ON category.productId = product.productId
WHERE cattype.catTypeName = 'Женщинам'
AND catsuper.catSuperName = 'Верхняя одежда'
AND catsub.catSubName = 'Пальто'
```

Результат запроса на рисунке 3.19.

Наименование	Описание	Материал	Цена	Изображение
Пальто изо льна и хлопка	Идеально для погоды, когда нужна легкая верхняя одежда.	52% Лён, 48% Хлопок	24000	https://www.uniqlo.com/ru/estore/hmall/test/u0000000003319/main/first/561/1.jpg

Рисунок 3.19 – Вывод товаров в категориях

PHP код для данного запроса представлен на рисунке 3.20 и его результат на рисунке 3.21

```php
<?php
require 'connect.php';

$sql = "SELECT
product.productName as 'Наименование',
product.productDescription as 'Описание',
product.productMaterial as 'Материал',
product.productPrice as 'Цена',
product.productImage as 'Изображение'
FROM category
INNER JOIN cattype ON category.catTypeId = cattype.catTypeId
INNER JOIN catsuper on category.catSuperId = catsuper.catSuperId
INNER JOIN catsub on category.catSubId = catsub.catSubId
INNER JOIN product ON category.productId = product.productId
WHERE cattype.catTypeName = 'Женщинам'
AND catsuper.catSuperName = 'Верхняя одежда'
AND catsub.catSubName = 'Пальто'";

$result = $conn->query($sql);
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        $db_data[] = $row;
    }
    echo json_encode($db_data, JSON_UNESCAPED_UNICODE);
}
$conn->close();
```

Рисунок 3.20 – Код вывода товара по категории

[{"Наименование":"Пальто изо льна и хлопка","Описание":"Идеально для погоды, когда нужна легкая верхняя одежда.","Материал":"52% Лён, 48% Хлопок","Цена":"24000","Изображение":"https:\/\/www.uniqlo.com\/ru\/estore\/hmall\/test\/u0000000003319\/main\/first\/561\/1.jpg"}]

Рисунок 3.21 – JSON-текст

3.3 Разработка клиентской части

Клиентская часть или же frontend это та часть, с которой взаимодействует пользователь, она представляет собой интуитивно понятной интерфейс с удобной визуальной составляющей. В проекте клиентская часть будет написана с помощью фреймворка Flutter, на языке Dart, а для взаимодействия с сервером будет использоваться архитектурный стиль для взаимодействия компонентов распределенного приложения в сети – REST (Representational State Transfer) [13]. Для веб-служб, построенных с помощью REST, применим термин «RESTful» [14].

Сообщение посредством этой архитектуры происходит путем передачи обычного HTTP-запроса, а необходимые данные передаются в качестве параметров запроса [15]. В проекте будут использованы 4 метода запросов:

- Get – метод, который запрашивает данные из ресурса. Запросы, выполненные с помощью этого метода, могут только получать данные;
- Post – метод, который отправляет данные на ресурс, также можно использовать для изменения данных на ресурсе;
- Put – метод, который заменяет данные ресурса данными хранящимися в запросе;
- Delete – метод с помощью которого происходит удаления данных на ресурсе или же в зависимости от запроса сам ресурс.

Хоть REST и не имеет официальных стандартов, но с помощью этой архитектуры можно адаптировать проект под любой другой API с минимальными правками.

Итак, начало приложение берет с формы авторизации или регистрации, если ранее пользователь не был зарегистрирован. Стандартная форма авторизации состоит из двух основных полей: логин и пароль. На рисунке 3.22 изображена форма авторизации.

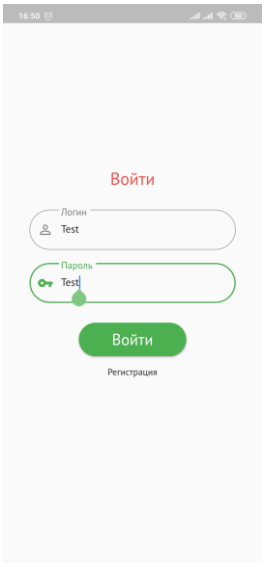


Рисунок 3.22 – Авторизация

Отрезки из кода, который составляет этот интерфейс приведены ниже. На рисунке 3.23 представлен класс LoginScreen который был расширен виджетом и классом StatelessWidget и представлены остальные элементы древа. Теперь по порядку. Для построения любой страницы необходимо использовать один из двух виджетов, первый описан ранее, второй – StatefullWidget. Их главное отличие в том, что первый не может самостоятельно изменять состояние, для того чтобы это осуществить нужно использовать конструкцию setState(), которая задает состояние для содержимого виджета.

Следующим идут переменные, который необходимы для работы с полями ввода, а именно со значениями, которые вводятся в поле для дальнейшего использования, оно будет рассмотрено позже.

Рисунок 3.23 – Отрезок кода

Так как все во фреймворке является виджетами, то необходимо всего лишь использовать предоставляемые шаблоны для элементов и использовать оптимально и креативно. Хоть набор шаблонов не велик, но их можно комбинировать и получать совершенно разные решения. Все зависит от креативности разработчика.

Далее по коду, как и говорилось ранее, все здесь виджеты. Здесь видно, что основной виджет предоставляет возможность собрать собственный виджет и перезаписать его, в конечном счете он будет возвращать собранные комбинации элементов интерфейса. На рисунке 3.23 видны несколько этих элементов, это: Scaffold, SafeArea, Container, SingleChildScrollView, Center, Column, Padding, Text.

Scaffold – это виджет, который отвечает за отображение элементов на экране смартфона. С этого виджета начинается вся графическая составляющая. Он имеет множество параметров, которые можно рассмотреть на рисунке 3.23, но зачастую используют небольшую часть, такие как appbar, body, drawer, floatActionButton. Первые два составляют основу страницы, точнее параметр body, его применение можно увидеть на рисунке выше. Appbar составляет собой панель в верхней части экрана, можно сказать заголовок. Drawer создает

список который выпадает слева. FloatActionButton – готовый шаблон кнопки, которая находится в нижней части экрана, но о ней позже.

SafeArea представляет собой виджет, который создает безопасные спуски с разрешенной в параметрах стороны. Из-за креативных решений дизайнеров смартфонов, этот виджет выглядит крайне полезным.

Container – виджет, название которого говорит за себя, это контейнер, который имеет множество параметров.

SingleChildScrollView – виджет, который позволяет прокручивать страницу, если не все попадает в видимую часть экрана. Очень удобно для дисплеев разных размеров.

Center – виджет, который центрирует содержимое в параметре child.

Column – виджет, который вмещает в себя список дочерних элементов в неограниченном количестве. Располагаются элементы соответственно в колонку друг под другом.

Padding – создает отступ, в зависимости от выбранного параметра, от своих краев.

Text – виджет, который собственно выводит текст.

По названиям можно понять за что отвечают виджеты и где их можно применить, что создает дружелюбную среду для разработчиков и упрощает процесс ознакомления с фреймворком.

Как выглядит форма регистрации, можно узнать на рисунке 3.24.

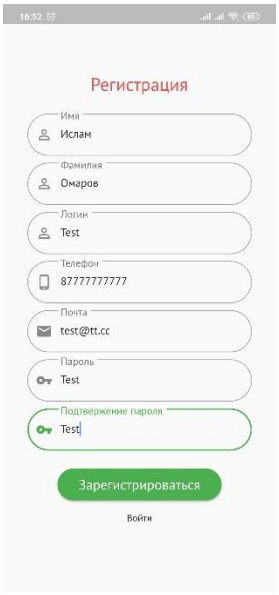


Рисунок 3.24 – Форма регистрации

Видно, что форм добавилось, но структура с предыдущим экраном не изменилась. Собственно, она практически везде будет одинаковой.

Рассмотрим теперь страницу с категориями. Это довольно простая в плане вида страница и то, как она выглядит можно рассмотреть на рисунке 3.25

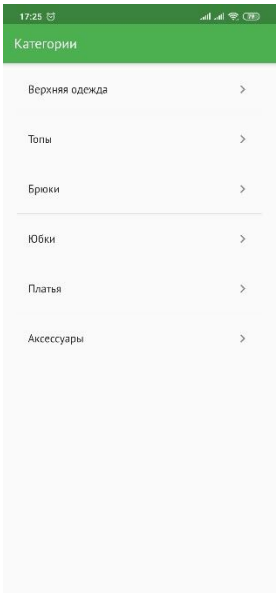


Рисунок 3.25 – Страница категорий

Страница представляет из себя простой список с категориями. Код этой страницы представлен на рисунке 3.26, но имеет простое содержание.

```
class CategoryScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Категории'),
      ), // AppBar
      body: Container(
        padding: EdgeInsets.symmetric(
          vertical: 10,
          horizontal: 20
        ), // EdgeInsets.symmetric
        child: Column(
          children: <Widget>[
            ListTile(
              title: Text('Верхняя одежда'),
              trailing: Icon(Icons.navigate_next),
            ), // ListTile
            Divider(),
```

Рисунок 3.26 – Отрезок кода

Как видно на скриншоте, структура простая и схожая с предыдущей страницей. Поэтому данная страница будет лишь дополнена новыми виджетами, а именно ListTile, Divider и Icon.

Divider – виджет, который отрисовывает линию, имеет параметр width для изменения толщины линии.

ListTile – это грубо говоря расширенная версия строки, которая с помощью определенных параметров дополняет список. Как видно на рисунке, параметр title это основная часть виджета, но также имеется подпись subtitle, а параметр trailing помогает вставить что-либо в конце строки. Аналогичный параметр – leading, он позволяет дополнить начало строки.

Icon – виджет, который отвечает за предоставления иконок. Иконки встроенные во фреймворк.

Вообще, все виджеты максимально приближены к философии Android по мнению Google – Material Design. Material Design можно назвать философией, так как большая часть разработчиков старается приблизить или привести в соответствие с ним свои продукты и предпочитает опираться на ряды советов по разработке продукта. Советом могут быть цвета, используемые в программе, шрифты, которые более читабельны и удобны, расположение элементов на экране, форма, вид интерфейса и др.

Дальше будет рассмотрена страница с товаром, тут набор виджетов практически не изменяется, но дополняется значительный функционал. На рисунке 3.27 можно увидеть малое сходство с предыдущими страницами.

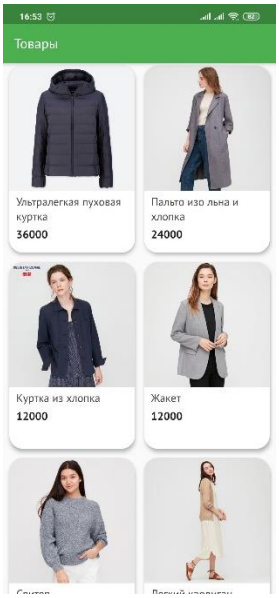


Рисунок 3.27 – Каталог

Как можно заметить, список заменился плитками, а текст дополнился изображениями. Ознакомиться с кодом страницы можно на рисунке 3.28.

```
@override
Widget build(BuildContext context) {
  return Container(
    padding: const EdgeInsets.symmetric(horizontal: 5),
    child: FutureBuilder(
        future: _getProductList(),
        builder: (BuildContext context, AsyncSnapshot snapshot) {
          if (snapshot.hasData) {
            return GridView.builder(
                itemCount: snapshot.data.length,
                gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
                    crossAxisCount: 2,
                    mainAxisSpacing: 5,
                    crossAxisSpacing: 5,
                    childAspectRatio: 0.68), // SliverGridDelegateWithFixedCrossAxisCount
                itemBuilder: (context, index) {
                  return Card(
                    shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(20)), // RoundedRectangleBorder
                    elevation: 4,
                    child: InkWell(
                      onTap: () {
                        Navigator.of(context).pushNamed('/productdetail', arguments: {'id':snapshot.data[index].productId});
                      },
                      child: Column(
                        crossAxisAlignment: CrossAxisAlignment.stretch,
```

Рисунок 3.28 – Отрезок кода

Здесь можно увидь немного иную структуру древа, а именно FututreBuilder. Этот виджет помогает выстраивать элементы в соответствии с содержимым, которое получается в параметре future. Вообще класс Future помогает выполнять задачи асинхронно и параллельно, то есть пока выполняется процесс в порядке очередности, функция, которая выполнена асинхронно, будет выполняться одновременно и как только появится возможность, вернет необходимые значения что видно в отрезке кода. В параметре builder используется конструкция, в которую передаются такие аргументы как context и snapshot. Первый аргумент ответственен за построение виджетов в структуре, а второй за предоставление результатов выполнения асинхронной функции. Далее в условии, используется свойство snapshot – hasData, который уведомляет о том, что аргумент имеет данные. Если такие имеются, то происходит возврат другой конструкции древа, а именно генерация элементов в виде сетки. Основными параметрами являются itemCount, gridDelegate, itemBuilder. Первый параметр необходим для генерации конечного количества плиток, второй – как разделять плитки между собой и третий параметр – возврат нового древа.

Конструкцию создания можно интуитивно понять, первый параметр отвечает за виджеты, второй за их индексирование. В теле конструкции представлен виджет Card, можно сказать, что это готовый шаблон для вывода какой-либо информации. Из «коробки» этот виджет достаточно стилизован и выглядит визуально приятно, но его можно расширить по мере необходимого, как и все остальное. Как видно в отрывке, с помощью shape задается форма для карточки, elevation отвечает за тень под карточкой, child – за следующий элемент.

InkWell – виджет, который отвечает за визуальное реагирование на действие, по-умолчанию это волны от места нажатия. Если писать свой собственный эффект, то это будет довольно громоздко и усложнит работу, но Flutter предлагает такие решения сразу же после установки. Теперь о действии и реагировании, как видно в коде по нажатию на карточку происходит навигация на другую страницу, а также передача данных на эту самую страницу.

Более подробно с кодом можно ознакомиться в Приложении А.

4 Экономическая часть

4.1 Резюме

Расчет экономической эффективности представляет собой обязательную часть, содержащую основные статьи расходов на выполнение проекта. Следует отметить, что затраты на разработку программного продукта являются в большинстве своем единоразовыми, а приходятся они именно на процесс разработки. Далее в зависимости от желания заказчика идут расходы на сопровождение программного продукта.

Дипломный проект представляет собой разработку мобильного приложения для электронной коммерции. Проект удобен для использования на смартфонах из-за их распространенности, автономности и доступа к интернету.

4.2 Трудоемкость разработки программного продукта

Для определения трудоемкости необходимо разбить задачу на несколько этапов, определить проводимые работы на данном этапе и определить нагрузку, тем самым будет получена трудоемкость разработки ПП.

В таблице 4.1 представлены этапы работ и их трудоемкость.

Таблица 4.1 – Распределение работ по этапам, видам и оценка их трудоемкости

№	Этап разработки ПП	Вид работы на данном этапе	Трудоемкость разработки ПП	
			Чел. х час	Час х день
1	Анализ требований	Анализ предметной области, установление целей и задач.	1 х 16	8 х 2
2	Анализ рынка	Анализ существующих мобильных приложений, детальное рассмотрение функционала, выявление преимуществ и недостатков. Определение необходимости в разработке программного продукта и выявление особенностей на фоне конкурентных решений.	1 х 16	8 х 2
3	Проектирование	Разработка технического задания. Определение набора средств для разработки. Проектирование базы данных, интерфейса мобильного приложения	1 х 40	8 х 5

Продолжение таблицы 4.1

4	Реализация	Построение диаграмм. Построение базы данных, ее заполнение. Реализация интерфейса и программного продукта.	1 х 120	8 х 15
5	Тестирование	Тестирование мобильного приложения, определение неполадок, исправление ошибок.	1 х 24	8 х 3
6	Внедрение и поддержка	Введение программного продукта в эксплуатацию, ознакомление персонала с продуктом и его сопровождение.	1 х 16	8 х 2
Итого трудоемкость выполненного проекта			1 х 232	8 х 29

4.3 Расчет затрат на разработку программного продукта

Расчет полных затрат на разработку проектного решения будет осуществлен по формуле:

$$C_{ni} = З_{тр} + З_{сзi} + М_i + Р_{сi} + П_{зi} + Р_{нi} \quad (4.1)$$

где $З_{тр}$ – затраты на оплату труда разработчика, тенге;

$З_{сзi}$ – отчисления по социальному налогу, тенге;

$М_i$ – затраты на материалы, тенге;

$Р_{сi}$ – затраты на специальные программные средства необходимые для разработки, тенге;

$П_{зi}$ – прочие затраты, тенге;

$Р_{нi}$ – накладные расходы, тенге.

Размер оплаты труда разработчиков рассчитывается по формуле:

$$З_{фот} = \sum_{i=1}^{n} ЧС_i \times T_i \quad (4.2)$$

где T_i – трудоемкость разработки;

$ЧС_i$ – часовая ставка, тенге/час.

Затраты на оплату труда записаны в таблицу 4.2.

Таблица 4.2 – Затраты на оплату труда

Исполнитель	Трудоемкость, чел. х час	Часовая ставка, тенге/час	Сумма, тенге
Разработчик - программист	1 х 232	700	162400
Итого затрат на оплату труда			162400

Социальный налог составляет 9,5% от заработной платы сотрудника и рассчитывается по формуле:

$$З_{сзi} = (З_{тр} - ПО - ВОСМС) \times 9{,}5\% - З_{соi} \quad (4.3)$$

где ПО – пенсионные отчисления, которые составляют 10% от заработной платы.

$$ПО = З_{тр} \times 10\% \quad (4.4)$$

Социальные отчисления составляют 3,5% от дохода разработчика, рассчитывается по формуле:

$$З_{соi} = (З_{тр} - ПО) \times 3{,}5\% \quad (4.5)$$

Отчисления на взнос по обязательному социальному медицинскому страхованию (ВОСМС) составляют 2% от заработной платы работника и рассчитываются по формуле:

$$ВОСМС = З_{тр} \times 2\% \quad (4.6)$$

Тогда,

$$ПО = 162400 \times 10\% = 16240 \text{ тенге};$$
$$ВОСМС = 162400 \times 2\% = 3248 \text{ тенге};$$
$$З_{соi} = (162400 - 16240) \times 3{,}5\% = 5116 \text{ тенге};$$
$$З_{сзi} = (162400 - 16240 - 1624) \times 9{,}5\% - 5116 = 8616 \text{ тенге}.$$

Сумма всех налогов равна 10,46% от заработной платы. Отсюда:

$$З_{сзi} = 162400 \times 10{,}46\% = 16987 \text{ тенге}$$

Затраты на материалы определятся по формуле:

$$M_i = \frac{З_{тр} \times Н_{мз}}{100\%} \quad (4.7)$$

где $Н_{мз}$ – норма расхода материалов от заработной платы (3-5%).
Поэтому затраты на материалы составляют,

$$M_i = \frac{162400 \times 5\%}{100\%} = 8120 \text{ тенге}.$$

Для разработки понадобятся средства, программное и техническое обеспечение, представленное в таблице 4.3

Таблица 4.3 – Спецоборудование»

Наименование	Описание	Кол-во, шт.	Цена за единицу, тг	Сумма, тг
Среда для разработки программного обеспечения	Visual Studio Code	1	0	0
Программа для проектирования интерфейса	Adobe XD	1	0	0
Программа для проектирования базы данных	MySQL Workbench	1	0	0
Фреймворк для разработки интерфейса	Flutter	1	0	0
Аккаунт разработчика	Google Play	1	10625	10625
Аккаунт разработчика	AppStore	1	42500	42500
Ноутбук Dell Vostro 5568	Core i5-7200U/RAM 8Gb/ SSD 515GB/ Nvidia 940MX 4Gb/ 1920x1080	1	229875	229875
			Итого	283000

Затраты на электроэнергию рассчитываются по формуле 4.8:

$$З_Э = \sum_{i=1}^{n} М_i \times К_i \times Т_i \times Ц \quad (4.8)$$

где $М_i$ – паспортная мощность электрооборудования, кВт;

$К_i$- коэффициент использования мощности, $К_i$равен 0,7;

$Т_i$ – время работы оборудования;

Ц – цена электроэнергии, тг/кВт*ч.

Для работы ноутбука по паспорту необходимо 65 Вт или же 0,065 кВт.

Цена электроэнергии с 1 января 2020 года составляет 19,17 тенге с НДС за 1 кВт/ч.

Следовательно затраты на электроэнергию равны:

$$З_Э = 0{,}065 \times 0{,}7 \times 232 \times 19{,}17 = 202 \text{ тенге.}$$

Сумма годовых амортизационных отчислений определяется методом уменьшающегося остатка по формуле:

$$А = \frac{Ф \times Н_а}{100} \quad (4.9)$$

где Ф – первоначальная стоимость основных производственных фондов;

$Н_а$ – норма амортизации.

При использовании метода уменьшающегося остатка применяется удвоенная ставка амортизации.

Годовые нормы амортизации основных фондов принимаются по налоговому кодексу РК или определяется, на основе срока полезного использования основных фондов по формуле:

$$Н_а = \frac{100}{Т} \quad (4.10)$$

где Т – возможный срок использования, год.

Для программного продукта срок полезного использования – 4 года.

$$Н_а = \frac{100}{4} \times 2 = 50\%$$

Амортизационные отчисления приведены в таблице 4.4

Таблица 4.4 – Амортизация основных фондов

Период	Норма амортизации	Амортизация за период	Накопленная амортизация	Балансовая стоимость на конец года
				283000
2020 год	50%	141500	141500	141500
2021 год	50%	70750	212250	70750
2022 год	50%	35375	247625	35375
2023 год	50%	17687,5	283000	0

Амортизация в последний период равен балансовой стоимости на конец предыдущего периода.

Для вычисления общих затрат на разработку программного продукта необходимо рассчитать амортизационные отчисления за период разработки по формуле:

$$З_{ам} = \frac{Ф \times Н_а \times N}{100\% \times 12 \times t} \quad (4.11)$$

где N – время использования программного продукта, дни;

t – количество рабочих дней в месяце.

Основные производственные фонды используются только на протяжении периода разработки и тестирования продукта, что значит время использования равно 232 часам или 29 дням.

Следовательно,

$$З_{ам} = \frac{283000 \times 50\% \times 29}{100\% \times 12 \times 21} = 16284 \text{ тенге.}$$

Прочие затраты» включает затраты на содержание аппарата управления, вспомогательных хозяйств и опытных (экспериментальных) производств, а также расходы на общехозяйственные нужды, относятся на конкретное ПО по нормативу в процентном соотношении к заработной плате исполнителей. Норматив устанавливается в целом организацией:

$$P_{нi} = З_{тр} \times \frac{Н_{нр}}{100} \quad (4.12)$$

где $P_{нi}$ – накладные расходы на конкретное ПО (тыс. тенге);

$Н_{нр}$ – норматив накладных расходов в целом по организации и равен 70%.

Следовательно,

$$P_{нi} = 162400 \times \frac{70}{100} = 113680 \text{ тенге.}$$

Расходы по статье «Прочие расходы» на конкретное ПО включает затраты на приобретение и подготовку специальной научно-технической информации и специальной литературы. Определяется по нормативу, разрабатываемому в целом по организации, в процентах к заработной плате:

$$П_{зi} = З_{тр} \times \frac{Н_{рнк}}{100} \quad (4.13)$$

где $Н_{рнк}$ – норматив прочих затрат в целом по организации и равен 20%.

Получается,

$$П_{зi} = З_{тр} \times \frac{20}{100} = 32480 \text{ тенге.}$$

Результаты всех выполненных расчетов записаны в таблице 4.5, по этим данным будет построена диаграмм затрат, которая проиллюстрирована ниже на рисунке 4.1.

Таблица 4.5 – Результаты выполненных расчетов

Затраты на разработку	Условное обозначение	Значение, тенге	В процентах от общей суммы
Заработная плата	$З_{тр}$	162400	26%
Налоги от заработной платы	$Зн$	16987	3%
Материалы	M_i	8120	1%
Спецоборудование	$Р_{сі}$	283000	45%
Затраты на электроэнергию	$З_Э$	202	0%
Амортизация основных фондов	$З_{ам}$	16284	3%
Прочие затраты	$П_{зі}$	32480	5%
Накладные расходы	$Р_{ні}$	113680	18%
Итого	$С_{пі}$	633153	100%

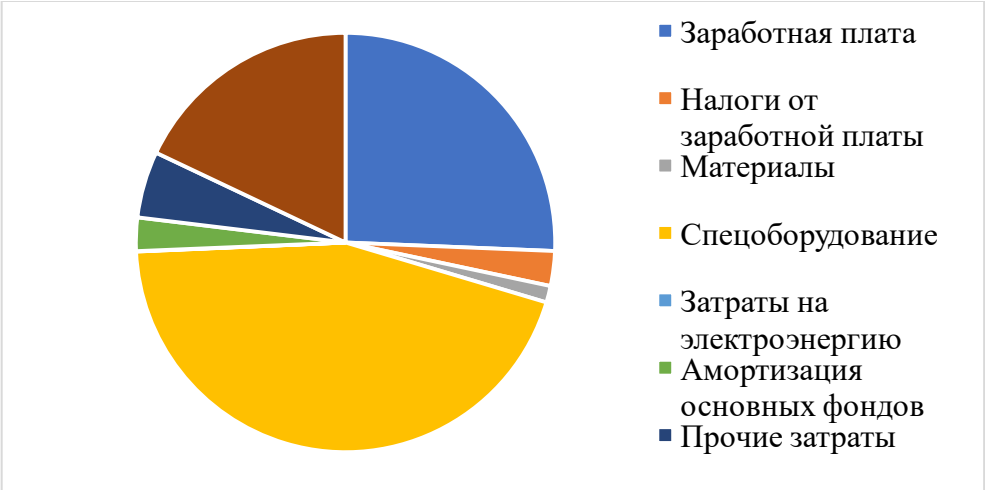


Рисунок 4.1 – Диаграмма затрат

Первоначальную цену продукта можно рассчитать по формуле

$$С_0 = 388522\left(1 \times \frac{50}{100}\right) = 949730 \text{ тенге.}$$

Цена готовой продукции с учетом НДС, которая равна 12%, рассчитывается по формуле:

$$С_р = С_0 + НДС \quad (4.14)$$

Отсюда, конечная цена программного продукта равна:

$$С_р = 949730 + (949730 \times 12\%) = 1063697 \text{ тенге.}$$

4.4 Расчет сравнительной экономической эффективности программного продукта

Разработка мобильного приложения формата интернет-магазина преследовала цель увеличения охвата аудитории и в последствии увеличения прибыли.

Так как физически сложно обслуживать каждого человека и нередко эти люди обращаются не с целью что-либо приобрести или наоборот, люди, знающие что именно они хотят купить будут ожидать в порядке очереди, чего не все готовы выждать. Данное приложение теоретически поможет сократить утечку потребителей, увеличить клиентскую базу и прибыль для предприятия, а также повысит его имидж, что тоже может принести свои результаты.

Величина ожидаемого годового эффекта от внедрения ПП рассчитывается по формуле

$$Э_г = Э_{уг} - К \times Е_н \quad (4.15)$$

где $Э_г$ – ожидаемый годовой экономический эффект, тенге;

$Э_{уг}$ – ожидаемая условно-годовая экономия, тенге;

К – капитальные вложения, тенге;

$Е_н$ – нормативный коэффициент эффективности капитальных вложений.

$Е_н$ рассчитывается по формуле:

$$Е_н = \frac{1}{Т_н} \quad (4.16)$$

где $Т_н$ – нормативный срок окупаемости капитальных вложений, лет.

Для программных продуктов срок окупаемости равен 4 годам.

Мобильное приложение поможет сократить персонал на 2 человек, который находятся в зале для обслуживания клиентов. Заработная плата консультанта за месяц составляет 120000 тенге, или же 1440000 тенге в год. На двоих приходится 2880000 тенге в год.

Следовательно,

$$Э_г = 2880000 - 1063697 \times (0{,}25) = 2614076 \text{ тенге.}$$

Расчетный коэффициент экономической эффективности капитальных вложений составляет:

$$Е_р = \frac{Э_{уг}}{К} \quad (4.17)$$

Расчетный срок окупаемости капитальных вложений происходит по формуле:

$$T_р = \frac{1}{E_р} \quad (4.18)$$

Отсюда,

$$E_р = \frac{2880000}{1063697} = 2{,}7;$$
$$T_р = \frac{1}{2{,}7} = 0{,}37 \text{ года.}$$

Срок окупаемости составляет 0,37 года или же больше 4 месяцев.

Результаты расчетов внесены в таблицу 4.6.

Таблица 4.6 – Показатели сравнительной экономической эффективности от внедрения программного продукта

Наименование показателей	Значение
Условная годовая экономия затрат, тенге	2880000
Коэффициент экономической эффективности капитальных вложений	2,7
Сроки окупаемости капитальных вложений, год	0,37

4.5 Вывод по экономическому разделу

В данном разделе были рассмотрены экономические вопросы по разрабатываемому мобильному приложению для магазина одежды «Unimag». Данное приложение было выбрано в рамках заказа от предприятия ИП “Unimag”. Были рассмотрены какие этапы проекта имеются и определили их трудоемкость эмпирическим путем. Были рассмотрены капитальные расходы на разработку и его стоимость. Был рассчитан коэффициент экономической эффективности с помощью которого было найден срок окупаемости проекта, который равен более 4 месяцев.

В некотором смысле мобильное приложение увеличивает посещаемость магазина и потенциальных покупателей, то прибыль соответственно увеличится, а следовательно, проект довольно быстро окупится. Также оно сократит затраты на содержание персонала, путем выполнения их функций.

Если исследовать цены на разработку мобильных приложений, то можно увидеть, что полученная стоимость будет выше средней рыночной цены. Однако цены представлены для одной платформы, а так как разрабатываемый проект является кроссплатформенным, то по цене разработки на одну платформу заказчик получает продукт, поддерживаемый на двух платформах.

5 Безопасность жизнедеятельности

5.1 Анализ потенциально опасных и вредных факторов воздействия на персонал в процессе работы

В данном разделе будет проведен анализ условий труда офиса, в котором будет разработан и будет использоваться программный продукт.

Офис не является местом, где встречаются опасные факторы для здоровья или, происходит тяжелая и травмоопасная работа, однако некоторый вред он все же может нанести. В офисах зачастую происходит работа в сидячем положении, в окружении множества источников электромагнитного излучения, нередки случаи работы без оптимального освещения или же температурного режима и плохо проветриваемого помещения.

Из-за постоянной работы в сидящем положении происходят перегрузки костно-мышечного аппарата, дистрофия мышц, при неправильном положении на рабочем месте возможны искривления позвоночника или же перегрузка кистей рук, также возможны более серьезные проблемы со здоровьем.

На фоне постоянного нахождения в офисе может произойти напряжения в коллективе, что приведет к эмоциональному напряжению, частой усталости из-за исчерпанного ментального ресурса, что приведет к снижению работоспособности и станет причиной частых заболеваний, в том числе серьезных.

В связи с тем, что в ограниченном помещении находится много источников электромагнитного излучения и персонал на постоянной основе вынужден взаимодействовать с этими источниками, то высок риск облучения электромагнитной радиацией, что может в худшем случае привести к развитию новообразований, также может повыситься раздраженность человека, ухудшиться его эффективность сна и прочие проблемы.

Из-за изменений климата и воздухообмена могут появиться респираторные заболевания, ухудшаться самочувствие, снижаться работоспособность, обостряться хронические заболевания или же аллергия.

Этот список можно продолжать, но для того, чтобы не усугублять ситуации и не допускать худшего развития событий были введены нормы. Эти нормы установлены в Кодексе Республики Казахстан «О здоровье народа и системе здравоохранения» статья 144 пункт 6, где представлены нормативные акты гигиенических нормативов физических факторов, оказывающих воздействие на человека, которые согласуются с министром здравоохранения и социального развития Республики Казахстан.

Все устройства, работающие с электричеством, создают определенное электромагнитное поле, которое в той или иной степени влияют на организм. И для того, чтобы минимизировать ущерб или избавиться от них вводятся нормы и правила для регулирования электрического воздействия на организм в рамках допустимого. Министерство Здравоохранения Республики Казахстан внесло некоторые правила и нормы на их эксплуатацию. Документ СанПиН

«Санитарно-эпидемиологические требования к эксплуатации радиоэлектронных средств и условиям работы с источниками электромагнитного излучения» регулирует вопросы эксплуатации и устанавливает требования к радиоэлектронным средствам [16].

Одними из требований являются:

- мощность станций передатчиков не должна превышать 100 Вт;
- коэффициент усиления антенны в пределах 10-16 дБ;
- мощность автомобильных станций 8-20 Вт;
- мощность радиотелефонов 0,8-5Вт;
- норма напряженности электромагнитного поля (ЭМП) – 10 мкВТ/см^2. Этот параметр описывает плотность потока энергии (ППЭ) проходящий через человека.
- И т.д.

В соответствии с рабочим диапазоном частот (400 – 1800 МГц) нормируемыми параметрами излучений систем сотовой связи являются ППЭ и энергетическая нагрузка (ЭН) на организм. ППЭ измеряется, в единицах поверхностной плотности мощности (Вт/м2, мВт/см2, мкВт/см2).

Человек в повседневной жизни практически везде сталкивается с воздействием электромагнитного поля на организм, а специалисты, которые работают со всевозможными источниками ЭМП подвержены постоянному облучению. Люди подвергающиеся облучению ЭМП замечают такие симптомы, как:

- изменения в эмоциональном состоянии;
- повышение раздражительности;
- ухудшение внимания;
- ухудшение памяти;
- снижение эффективности сна;
- и т.д.[16].

Возможны изменения биохимических показателей крови, появление головных болей и формирование злокачественных новообразований [17].

Общий характер воздействия ЭМП на организм человека можно выразить следующими факторами:

- опасность усиливается по мере возрастания частоты электромагнитных колебаний, поэтому предельно-допустимые уровни (ПДУ), рекомендованные санитарными службами и прописанные в нормативных документах, снижаются [17];
- опасность возрастает с повышением интенсивности излучения (мощности, амплитуды, напряженности, плотности потока). Например, человек, находящийся в непосредственной близости от антенн базовой станции сотовой связи мощностью 20 Вт получит больше вреда, чем от абонентского устройства – личного мобильного телефона мощностью 1 Вт. Но если базовая станция далеко и мощность ее передатчика в точке приема составляет, скажем,

1 мВт, то сотовый телефон сам станет источником повышенной угрозы (1 Вт = 1000 мВт) [17];

- опасность повышается при расширении спектрального состава (полосы рабочих частот, количества каналов). Для большинства бытовой электротехники, имеющей основное излучение в области низких частот и небольшой спектр излучения, действуют нормы СанПиН 2.1.2.1002-00. Для персональных компьютеров, имеющих сложную схему и широкую полосу генерации с частотами, достигающими нескольких гигагерц, разработан и введен отдельный нормативный документ СанПиН 2.2.2/2.4.1340-03. Здесь требования к ПДУ ЭМП гораздо жестче [17];
- опасность возрастает с увеличением времени воздействия ЭМП [17].

Чтобы защититься от воздействия ЭМП необходимо определить источники. В условиях дома или квартиры излучения, превышающие ПДУ, могут исходить из точек подвода квартирной электросети к системе энергообеспечения. Бытовые электроприборы находятся в области опасного излучения промышленных частот из-за близкого контакта с человеком [17].

В высокочастотной области спектра находятся персональный компьютер, смартфон, передатчики типа радиостанций, Wi-Fi роутеров или Bluetooth устройств.

Для более эффективной защиты следует ограничить время воздействия ЭМП на организм, находиться от источника на максимально возможном удалении, снизить время эксплуатации высокочастотной техники, снизить количество одновременно работающей бытовой техники, устанавливать защитные экраны, металлические щиты, фольгу. Также необходимо минимизировать воздействия ЭМП на места отдыха человека.

5.2 Расчетная часть

5.2.1 Воздействие ЭМП сотового диапазона на организм человека

Частоты сотового диапазона разбросаны от 800 МГц до 1800 МГц и иногда более 2000 МГц. Более подробные диапазоны частот представлены в таблице 5.1 [16].

Таблица 5.1 – Частотный диапазон сотовых операторов Казахстана

Оператор	2G	3G	4G
Altel	900МГц	850/900 МГц	1800 МГц
Beeline	900МГц	2100 МГц	800/1800/2100 МГц
Kcell/Activ	900МГц	2100 МГц	800/1800/2100 МГц
Tele2	900МГц	850/900 МГц	1800МГц

Один из важных параметров, который определяет эффект биологического воздействия на человека – плотность потока энергии, измеряемый в мкВт/см^2. Согласно требованиям, описанным в документ

СанПиН «Санитарно-эпидемиологические требования к эксплуатации радиоэлектронных средств и условиям работы с источниками электромагнитного излучения», это значение составляет 10 мкВт/см^2 или же 100 мВт/м^2 [17].

Как можно видеть по таблице частот сотовых операторов, нет определенных стандартов для типа соединения, также различные производители имеют свои поддерживаемые частоты для разных форматов сети. К примеру, распространенный iPhone X имеет ППЭ в 273 мкВт/см^2, а Redmi 8 Note – 247 мкВт/см^2.

Второй критерий воздействия ППЭ на организм это Specific Adsorption rate (SAR) или же удельная поглощенная мощность, выраженная на единицу массы тела или ткани, измеряемая в Вт/кг. Параметр определяет, на сколько градусов нагреваются ткани тела под воздействием радиоизлучения. В международной практике максимальный уровень SAR определяют от 1,6 или 2,0 Вт/кг [18].

Степень поглощения электромагнитного излучения организмом человека может зависеть от множества факторов, таких как мощность излучения смартфона (максимальная и средняя), расстояние и расположение аппарата относительно головы или тела пользователя, габариты и форма смартфона и антенны, замеряемые частоты и др. В связи с этим, уровень SAR оценивают несколькими способами, например, для головы, тела, в режиме одновременной передачи в разных частотных диапазонах (одновременное излучение, широкополосное излучение) и др. [18].

SAR вычисляется по формулам [19]:

1) сила поля в тканях [19]:

$$\mathrm{SAR} = \frac{\sigma \vec{E}^2}{\rho} \tag{5.1}$$

2) плотность тока в тканях [19]:

$$SAR = \frac{J^2}{\rho\sigma} \tag{5.2}$$

3) повышенная температура в тканях [19]:

$$SAR = c_i \frac{dT}{dt} \tag{5.3}$$

где E – напряженность электрического поля, В/м [19];

J – плотность тока, А/м^2 [19];

ρ – плотность человеческих тканей, кг/м^3 [19];

σ – электрическая проводимость человеческих тканей, См/м [19];

C_i – теплоемкость человеческих тканей, Дж/(кг*К) [19];

dT/dt – временная производная температуры человеческих тканей, в К/с [19].

В таблице 5.2 представлены некоторые модели телефонов и их SAR-уровни.

Таблица 5.2 – SAR уровни

Смартфон	SAR: Голова	SAR: Тело	SAR: Общее
Samsung Galaxy Note 9	0,27 Вт/кг	0,76 Вт/кг	1,59 Вт/кг
Xiaomi Pocophone F1	0,72 Вт/кг	0,75 Вт/кг	1,58 Вт/кг
Huawei Mate 20 Pro	0,58 Вт/кг	0,81 Вт/кг	1,53 Вт/кг
Apple iPhone XS	0,90 Вт/кг	0,99 Вт/кг	1,53 Вт/кг
Google Pixel 3	1,34 Вт/кг	1,34 Вт/кг	1,59 Вт/кг

Этот параметр вычисляется теоретически или оценивается экспериментально. Меньший SAR вызывает меньшее воздействие ЭМП мобильного устройства на человеческое тело.

5.2.2 Влияние экрана смартфона на зрение

Существует множество технологий изготовления матриц для смартфонов, такие как: LCD, TFT, IPS, OLED и т.д. Но основными технологиями считаются LCD и OLED, когда остальные являются их производнымир [20].

LCD (Liquid Crystal Display) – жидкокристаллические экраны. Жидкие кристаллы, которые лежат в основе технологии обладают двумя важнейшими свойствами: текучестью и анизотропностью. Анизотропность – это способность кристалла изменять свои свойства в зависимости от своего расположения в пространстве. В экранах эта особенность используется для управления светопроводимостью. С помощью транзисторов на ЖК-матрицу подается ток, который изменяет ориентацию кристаллов. Затем на них падает свет, проходящий через несколько фильтров, и в результате на экране появляется пиксель нужного цвета. Отметим, что для всех ЖК-экранов требуется источник подсветки: внешний (например, солнечные лучи) или встроенный (например, светодиоды) [20]. На рисунке 5.1 представлена схема LCD-экрана.

TFT (thin-film transistor) – это разновидность LCD-дисплеев, в которых для управления жидкими кристаллами используется активная матрица: в ее конструкцию входят тонкопленочные транзисторы [20]. На рисунке 5.2 изображена схема TFT экрана.

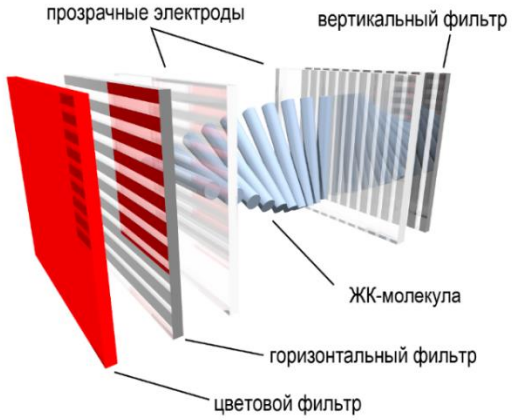


Рисунок 5.1 – Схема LCD

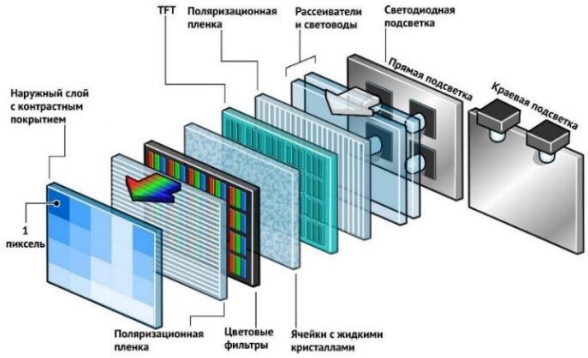


Рисунок 5.2 – Схема TFT

IPS (in-plane switching) – в таких экранах кристаллы при получении электрического импульса не скручиваются в спираль, а поворачиваются перпендикулярно своему начальному положению [20]. На рисунке 5.3 схема IPS экрана.

В OLED-матрицах (Organic light-emitting diode) вместо жидких кристаллов используются органические светодиоды, которые не требует подсветки. При подаче на них электрических импульсов они сами начинают светиться [20]. Схема OLED дисплея представлена на схеме 5.4.

При работе с любым экраном есть риск получить сухость глаза и близорукость. Так как при нормальных обстоятельствах человек моргает 18 раз в минуту, в следствии чего роговица увлажняется слезной жидкостью, но глядя в монитор можно заметить, как частота моргания заметно сокращается до 2-3 раз в минуту, что крайне мало, в последствии чего глаза сильно утомляются и появляется ощущение сухости и жжения в глазах [20].

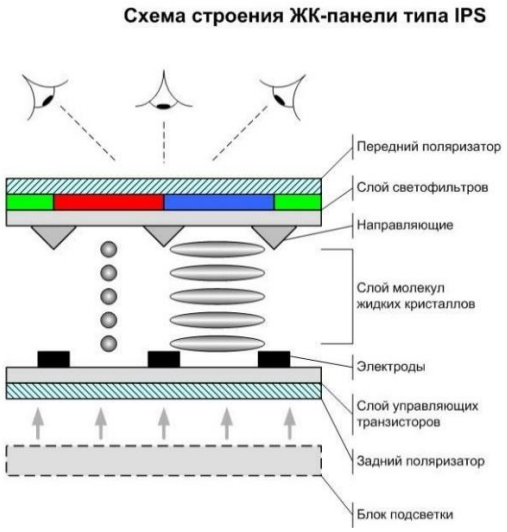


Рисунок 5.3 – Схема IPS

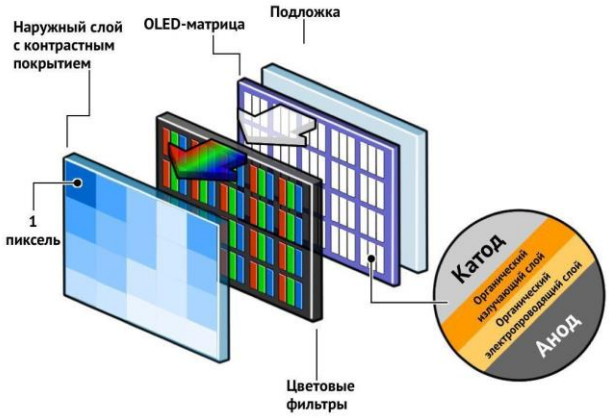


Рисунок 5.4 – Схема OLED

Вторая опасность — это миопия или же близорукость. При длительном контакте с экраном любого устройства глаза подвергаются постоянному напряжению и глазное яблоко начинает постепенно вытягиваться в следствии чего появляется близорукость. Однако есть еще так называемая ложная миопия. Она появляется в случаях, когда резко отрываешься от экрана и все вокруг начинает расплываться и мутнеть, со временем она проходит [20].

Офтальмологи отобрали ряд важных параметров дисплея смартфона. Это Количество точек на дюйм и яркость дисплея.

Первая характеристика — это соотношение размера экрана к его разрешению, обозначается как PPI (pixels-per-inch) [20].

Хорошим показателем является высокая плотность относительно экрана, к примеру, маленький экран с высоким разрешением будет более безопасен из-за более четкой картинки, чем большой экран с маленьким разрешением, так как PPI на маленьком экране будет больше, чем на большом. Если же картинка будет размыта, глаза начнут напрягаться для самостоятельной подстройки резкости, что приведет так же к близорукости [20].

Рассчитать PPI можно по формуле [21]:

$$PPI = \frac{d_p}{d_i} \quad (5.4)$$

где d_p – диагональное разрешение в пикселях;

d_i – размер диагонали в дюймах.

d_p рассчитывается по формуле [21]:

$$d_p = \sqrt{w_p^2 + h_p^2} \quad (5.5)$$

где w_p – ширина разрешения в пикселях;

h_p – высота разрешения в пикселях.

Посчитаем PPI для устройств стандартной диагонали в 6 дюймов и разрешением в FullHD (1920x1080).

w_p – 1080 пикселей;

h_p – 1920 пикселей;

d_p – 6 дюймов.

Тогда,

$$d_p = \sqrt{1080^2 + 1920^2} = 2202{,}9;$$

$$PPI = \frac{2202{,}9}{6} = 367{,}15.$$

Второй важный параметр – яркость. Человеческий глаз не приспособлен долго смотреть на яркий свет. Экран смартфона излучает яркий свет и на его фоне приходится различать текст или другую информацию, что заставляет организм неестественно реагировать на это, тем самым переставая моргать, что приводит к сухости, жжению и напряжению глаза или же к «синдрому сухого глаза». Чем ярче и резче свет, тем вреднее он для глаз. Параметр яркости зависит от разности освещения окружение и яркости экрана, а параметр резкости зависит от дисплея [22].

LCD используют технологию постоянной подсветки. Жидкие кристаллы подсвечиваются изнутри, за счет чего и формируется изображение. В TFT подсветка тусклее, чем в IPS, в которых применяется усиленная подсветка. Но эффект тот же: глаза постоянно подвергаются воздействию яркого света [22].

В OLED подсветка выборочная. Фактически, OLED-дисплей «всегда выключен», а светодиоды, составляющие основу экрана, загораются в зависимости от того, где и что нужно отобразить. Соответственно, световое воздействие этих экранов куда ниже, а свет намного мягче и безвреднее для глаз [22].

5.3 Расчет уровня шума

Шум является одним из негативных факторов, оказывающих влияние на человека, в условиях офиса этот параметр является одним из основных. Потому что мало кто сможет сосредоточиться и оставить эффективность на том же уровни при высоком уровне шума. Даже нормативное значение не идеально для персонала, но оно является компромиссным и оптимальным для офиса, где шум — это вполне обычное явление. Источниками шума могут быть персональные компьютеры, кондиционеры, телефоны и другое оборудование, даже разговор людей должен быть в пределах уровня обычной беседы.

Для решения вопросов о снижения шума требуется знать уровни шума на рабочем месте специалиста.

Уровень шума, возникающий от нескольких источников, работающих одновременно, рассчитываются на основании принципа энергетического суммирования излучений отдельных источников [23]:

$$L = 10lg \sum_{i=1}^{i=n} 10^{0.1Li}$$

где L_i – уровень звукового давления i-го источника шума;

n – количество источников шума.

Полученный результат сравнивается с допустимым значением, если уровень шума выше допустимого, то необходимы принять меры по снижению шума. Источники шума, окружающие разработчика и их уровень шума представлены в таблице 5.3, по ним будут проведены расчеты на соответствие нормам.

Таблица 5.3 Уровни звукового давления различных источников.

Источник шума	Уровень шума, дБ
Жесткий диск	28-35
Кондиционер	26-36
Ноутбук	31-43
Принтер	48
Разговор	40-45

Измерим уровень шума для нижних и высоких значений источников шума:

$$L = 10\text{lg}(10^{2,8} + 10^{2,6} + 10^{3,1} + 10^{4,8} + 10^{4}) = 48{,}7\text{дБ}$$
$$L = 10\text{lg}(10^{3,5} + 10^{3,6} + 10^{4,3} + 10^{4,8} + 10^{4,5}) = 50{,}9\text{дБ}$$

Полученные значения находятся ниже нормы, установленным в санитарных нормах, в 86дБ.

Если вычесть разговор, работу принтера и в определенные сезоны кондиционеры, то уровень шума значительно снизится. Однако эти значения находятся примерно на одном уровне, из-за чего окружение может сложиться в белый шум, что наоборот поможет отвлечься от всего вокруг и усилит сосредоточенное состояние, что повысит эффективность.

Заключение

В процессе выполнения данного проекта было разработано мобильное приложения на базе ОС Android для магазина одежды «Unimag». Мобильное приложение предоставляет возможность выполнять покупки и осуществлять коммерческую деятельность в дистанционном режиме и круглосуточно. Уникальностью данного проекта является разработка с использованием новых технологий равных по производительности нативным аналогам, а также возможность администрирования с помощью смартфона.

Разработка мобильного приложения была выполнена в среде разработки Visual Studio Code. Клиентская часть сделана с помощью фреймворка Flutter на языке Dart, серверная часть на языке PHP. В роли СУБД была использована MySQL с использованием веб-интерфейса PhpMyAdmin.

По мере выполнения проекта были достигнуты ранее поставленные задачи, такие как:

- исследование и анализ существующих приложений;
- выбор программного обеспечения;
- проектирование базы данных;
- разработка интерфейса приложения;
- программная реализация;

Также были выполнены разделы экономической части и безопасности жизнедеятельности.

По результатам расчета экономической части стоимость разработки мобильного приложения составила 1063697 тенге, коэффициент эффективности капитальных вложений равен 2,7, а срок окупаемости проекта будет не больше 5 месяцев.

В разделе безопасности жизнедеятельности было рассмотрено влияние ЭМП сотового диапазона на организм, влияние дисплея на зрение и произведен расчет шума.

Список литературы

1 Якобс П., Калдвелл Ю., Ханью Ч. //PEWRESEARCH.ORG: Social Media Use Continues to Rise in Developing Countries | Pew Research Center. URL: https://www.pewresearch.org/global/2018/06/19/2-smartphone-ownership-on-the-rise-in-emerging-economies/ (дата обращения 14.01.20).

2 Мусапирова А. // KURSIV.KZ: Какие смартфоны предпочитают казахстанцы. URL: https://kursiv.kz/news/tendencii-i-issledovaniya/2019-08/kakie-smartfony-predpochitayut-kazakhstancy (дата обращения 14.01.20).

3 Основы проектирования базы данных – пример. URL: https://function-x.ru/sql_prilozhenija_s_bazami_dannyh.html (дата обращения 02.02.20).

4 Логическая модель предметной области. URL: http://analyst.by/diagrams/logicheskaya-model-predmetnoy-oblasti (дата обращения 02.02.20).

5 Логическая модель данных. Понятие нормализации отношений. URL: https://studme.org/97320/informatika/logicheskaya_model_dannyh_ponyatie_normalizatsii_otnosheniy (дата обращения 02.02.20).

6 Ковтун М.В. // PRJ-EXP.RU: Типы моделей данных корпоративного хранилища данных. URL: https://www.prj-exp.ru/dwh/dwh_model_types.php (дата обращения 02.02.20).

7 Erwin. URL: https://www.kpms.ru/Automatization/ERwin.htm (дата обращения 05.02.20).

8 Графический клиент MySQL Workbench. URL: https://metanit.com/sql/mysql/1.3.php (дата обращения 07.02.20).

9 MySQL Workbench — Национальная библиотека им. Н. Э. Баумана. URL: https://ru.bmstu.wiki/MySQL_Workbench (дата обращения 08.02.20).

10 Adobe XD — Википедия. URL: https://ru.wikipedia.org/wiki/Adobe_XD (дата обращения 19.02.20).

11 API — Википедия. URL: https://ru.wikipedia.org/wiki/API (дата обращения 20.03.20).

12 JSON — Википедия" на сайте. URL: https://ru.wikipedia.org/wiki/JSON (дата обращения 22.03.20).

13 REST — Википедия. URL: https://ru.wikipedia.org/wiki/REST (дата обращения 28.03.20).

14 Архитектура REST. URL: https://habr.com/ru/post/38730/ (дата обращения 05.04.20).

15 Методы HTTP запроса. URL: https://developer.mozilla.org/ru/docs/Web/HTTP/Methods (дата обращения 09.04.20).

16 Частотные диапазоны 2G, 3G, 4G сотовых операторов Казахстана (Билайн, Алтел, Теле2, Кселл, Актив). URL: http://gadgetbox.kz/chastotnye-diapazony-sotovyh-operatorov (дата обращения 11.04.20).

17 СанПиН «Санитарно-эпидемиологические требования к эксплуатации радиоэлектронных средств и условиям работы с источниками электромагнитного излучения» (дата обращения 11.04.20).

18 Излучение сотовой связи: опасно? URL: https://habr.com/ru/company/yota/blog/311312/ (дата обращения 11.04.20).

19 Удельный коэффициент поглощения электромагнитной энергии. URL:https://ru.wikipedia.org/wiki/Удельный_коэффициент_поглощения_электромагнитной_энергии (дата обращения 12.04.20).

20 Михеев О. // HYPE.TECH: Виды экранов смартфонов: основные отличия матриц, преимущества и недостатки. URL: https://hype.tech/@id103/vidy-ekranov-smartfonov-osnovnye-otlichiya-matric-preimushchestva-i-nedostatki-9v8msqe7 (дата обращения 12.04.20).

21 PPI – Википедия. URL: https://ru.wikipedia.org/wiki/Ppi (дата обращения 12.04.20).

22 Типы экранов смартфонов: какой лучше для глаз? URL: https://ichip.ru/sovety/tipy-ehkranov-smartfonov-kakojj-luchshe-dlya-glaz-154602 (дата обращения 12.04.20)

23 Борьба с шумом на производстве: Справочник / Е.Я. Юдин, Л.А. Борисов, 2005. – 400с.

Приложение А
Листинг программы

Серверная часть:

```
<?php
// конфигурации
$host = 'localhost';
$user = 'TestUser';
$pass = 'testuser';
$dbname = 'unimag';
// соединиение с базой
$conn = mysqli_connect($host, $user, $pass, $dbname);
if ($conn->connect_error) {
    die("Ошибка соединения: " . $conn->connect_error .
    "Ошибка #" . $conn->connect_errno);}
require('connect.php');
$sql = "SELECT
     productId as 'Код',
     productName as 'Название',
     productPrice as 'Цена',
     productImage as 'Изображение'
     FROM product";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        $db_data[] = $row;
    }
    echo json_encode($db_data, JSON_UNESCAPED_UNICODE);
}
$conn->close();
require('connect.php');
    $login=$_POST['login'];
    $fN=$_POST['userfirstName'];
    $role=$_POST['role'];
$data = [
    'login' => $login,
    'userfirstName' => $fN,
    'role' => $role,
];
echo json_encode($data, JSON_UNESCAPED_UNICODE);
$sql = "SELECT
product.productName as 'Наименование',
product.productDescription as 'Описание',
product.productMaterial as 'Материал',
```

```
product.productPrice as 'Цена',
product.productImage as 'Изображение'
FROM category
INNER JOIN cattype ON category.catTypeId = cattype.catTypeId
INNER JOIN catsuper on category.catSuperId = catsuper.catSuperId
INNER JOIN catsub on category.catSubId = catsub.catSubId
INNER JOIN product ON category.productId = product.productId
WHERE cattype.catTypeName = 'Женщинам'
AND catsuper.catSuperName = 'Верхняя одежда'
AND catsub.catSubName = 'Пальто'";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
   while($row = $result->fetch_assoc()) {
      $db_data[] = $row;
   }
   echo json_encode($db_data, JSON_UNESCAPED_UNICODE);
}
$conn->close();
?>
require 'connect.php';
$id = $_GET['productId'];
$sql = "SELECT
product.productId AS 'Код товара' ,
product.productArticul AS 'Артикул',
product.productName AS 'Название',
product.productDescription as 'Описание',
product.productPrice  as 'Цена',
product.productImage as 'Изображение',
color.colorName as 'Цвет',
sizes.sizesName as 'Размер',
product.productMaterial as 'Материал'
FROM options
INNER JOIN sizes ON options.optionSizeId = sizes.sizesId
INNER JOIN product ON options.optionProductId = product.productId
INNER JOIN color ON options.optionColorId = color.colorId
WHERE product.productId =".$id;
$result = $conn->query($sql);
if ($result->num_rows > 0) {
   while ($row = $result->fetch_assoc()) {
         $db_data []= $row;
   }
   echo json_encode($db_data, JSON_UNESCAPED_UNICODE);}
```

```
$conn->close();
require('connect.php');
$sql = "SELECT cattype.catTypeId as 'Код пола',
cattype.catTypeName as 'Для кого',
catsuper.catSuperId as 'Код типа',
catsuper.catSuperName as 'Тип',
catsub.catSubId as 'Код категории',
catsub.catSubName as 'Категория'
FROM category
INNER JOIN cattype ON category.catTypeId = cattype.catTypeId
INNER JOIN catsuper ON category.catSuperId = catsuper.catSuperId
INNER JOIN catsub ON category.catSubId = catsub.catSubId
INNER JOIN product ON category.productId = product.productId
WHERE cattype.catTypeId = 1";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        $db_data [] = $row;
    }
    echo json_encode($db_data, JSON_UNESCAPED_UNICODE);
}
$conn->close();
?>
```

Клиентская часть:

```
import 'package:flutter/material.dart';
import './Screen/productScreen.dart';
void main() => runApp(MyApp());
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.green,
        accentColor: Colors.greenAccent,
        errorColor: Colors.red,
        fontFamily: 'PTSans',
      ),
      initialRoute: '/',
      routes: {ProductScreen.routeName: (context) => ProductScreen(),},);}}
```

```
import 'package:flutter/material.dart';
import '../Items/productsItem.dart';
class ProductScreen extends StatelessWidget {
  static const routeName = '/';
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Товары'),
      ),
      body: ProductsItem(),
    );
  }
}
import 'dart:convert';
import 'dart:async';
import 'package:http/http.dart' as http;
import 'package:flutter/material.dart';
import '../Model/products.dart';
class ProductsItem extends StatefulWidget {
  @override
  _ProductsItemState createState() => _ProductsItemState();
}
class _ProductsItemState extends State<ProductsItem> {
  Future<List<Products>> _getProductList() async {
    final data = await http.get('http://192.168.1.195/api.unimag/getdata.php');
    if (data.statusCode == 200) {
      final jsonData = json.decode(data.body);
      List<Products> products = [];
      for (var item in jsonData) {
        final product = Products(
          productId: item['Код'],
          productName: item['Название'],
          productDescription: item['Описание'],
          productPrice: item['Цена'],
          productImage: item['Изображение'],
        );
        products.add(product);
      }
      print('prod: ${products.length}');
      return products;
    }}
```

```
@override
Widget build(BuildContext context) {
 return Container(
  padding: const EdgeInsets.symmetric(horizontal: 5),
  child: FutureBuilder(
   future: _getProductList(),
   builder: (BuildContext context, AsyncSnapshot snapshot) {
    if (snapshot.hasData) {
     return GridView.builder(
      itemCount: snapshot.data.length,
      gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
       crossAxisCount: 2,
       mainAxisSpacing: 5,
       crossAxisSpacing: 5,
       childAspectRatio: 0.68),
      itemBuilder: (context, index) {
       return Card(
        shape: RoundedRectangleBorder(
         borderRadius: BorderRadius.circular(20)),
        elevation: 4,
        child: InkWell(
         onTap: () {
          Navigator.of(context).pushNamed('/productdetail',
arguments: {'id':snapshot.data[index].productId});
         },
         child: Column(
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: <Widget>[
           ClipRRect(
            borderRadius: BorderRadius.only(
             topLeft: Radius.circular(20),
             topRight: Radius.circular(20)),
            child: Image.network(
             snapshot.data[index].productImage,
             fit: BoxFit.cover,
            ),
           ),
           Padding(
            padding: const EdgeInsets.symmetric(
             vertical: 5,
             horizontal: 10,
            ),
```

```
                    child: Text(
                      snapshot.data[index].productName,
                      style: TextStyle(
                          fontSize: 16,
                          color: Colors.grey[800]),
                    ),
                  ),
                  Padding(
                    padding: const EdgeInsets.only(
                      left: 10,
                      right: 10,
                      bottom: 10,
                    ),
                    child: Text(
                      snapshot.data[index].productPrice,
                      style: TextStyle(
                        fontSize: 16,
                        fontWeight: FontWeight.w600,
                      ),
                    ),
                  ),
                ],
              ),
            ),
          );
        });
    } else if (snapshot.data == null) {
      return Center(child:
      CircularProgressIndicator(),
      );
    }
  }),
 );
 }
}
import 'package:flutter/material.dart';
class RegisterScreen extends StatelessWidget {
  TextEditingController _lname = TextEditingController();
  TextEditingController _fname = TextEditingController();
  TextEditingController _login = TextEditingController();
  TextEditingController _phone = TextEditingController();
  TextEditingController _email = TextEditingController();
```

```
TextEditingController _password = TextEditingController();
TextEditingController _passwordConfirm = TextEditingController();
@override
Widget build(BuildContext context) {
  return Scaffold(
    body: SafeArea(
        top: true,
        bottom: true,
        left: true,
        right: true,
        child: Container(
          padding: EdgeInsets.only(top: 50, left: 30, right: 30),
          child: SingleChildScrollView(
            child: Center(
              child: Column(
                children: <Widget>[
                  Padding(
                    padding: const EdgeInsets.all(20.0),
                    child: Text(
                      'Регистрация',
                      style: TextStyle(
                        color: Colors.red[400],
                        fontSize: 26,
                      ),
                    ),
                  ),
                  Padding(
                    padding: EdgeInsets.all(5),
                    child: TextField(
                      controller: _fname,
                      decoration: InputDecoration(
                          border: OutlineInputBorder(
                            borderSide:
                                BorderSide(color: Colors.grey, width: 0.1),
                            borderRadius: BorderRadius.circular(40),
                          ),
                          labelStyle: TextStyle(fontSize: 20),
                          labelText: 'Имя',
                          prefixIcon: Icon(Icons.perm_identity, size: 24),
                          hintText: 'Введите имя'),
                    ),
                  ),
```

```
Padding(
  padding: EdgeInsets.all(5),
  child: TextField(
    controller: _lname,
    decoration: InputDecoration(
        border: OutlineInputBorder(
          borderSide:
              BorderSide(color: Colors.grey, width: 0.1),
          borderRadius: BorderRadius.circular(40),
        ),
        labelStyle: TextStyle(fontSize: 20),
        labelText: 'Фамилия',
        prefixIcon: Icon(Icons.perm_identity, size: 24),
        hintText: 'Введите фамилию'),
  ),
),
Padding(
  padding: EdgeInsets.all(5),
  child: TextField(
    controller: _login,
    decoration: InputDecoration(
        border: OutlineInputBorder(
          borderSide:
              BorderSide(color: Colors.grey, width: 0.1),
          borderRadius: BorderRadius.circular(40),
        ),
        labelStyle: TextStyle(fontSize: 20),
        labelText: 'Логин',
        prefixIcon: Icon(Icons.perm_identity, size: 24),
        hintText: 'Введите логин'),
  ),
),
Padding(
  padding: EdgeInsets.all(5),
  child: TextField(
    controller: _phone,
    decoration: InputDecoration(
        border: OutlineInputBorder(
          borderSide:
              BorderSide(color: Colors.grey, width: 0.1),
          borderRadius: BorderRadius.circular(40),
        ),
```

```
        labelStyle: TextStyle(fontSize: 20),
        labelText: 'Телефон',
        prefixIcon: Icon(Icons.phone_android, size: 24),
        hintText: 'Введите ваш номер'),
  ),
),
Padding(
  padding: EdgeInsets.all(5),
  child: TextField(
    controller: _email,
    decoration: InputDecoration(
        border: OutlineInputBorder(
          borderSide:
              BorderSide(color: Colors.grey, width: 0.1),
          borderRadius: BorderRadius.circular(40),
        ),
        labelStyle: TextStyle(fontSize: 20),
        labelText: 'Почта',
        prefixIcon: Icon(Icons.email, size: 24),
        hintText: 'Введите вашу электронную почту'),
  ),
),
Padding(
  padding: EdgeInsets.all(5),
  child: TextField(
    controller: _password,
    decoration: InputDecoration(
        border: OutlineInputBorder(
          borderSide:
              BorderSide(color: Colors.grey, width: 0.1),
          borderRadius: BorderRadius.circular(40),
        ),
        labelText: 'Пароль',
        labelStyle: TextStyle(fontSize: 20),
        prefixIcon: Icon(
          Icons.vpn_key,
          size: 24,
        ),
        hintText: 'Введите пароль'),
  ),
),
Padding(
```

```
  padding: EdgeInsets.all(5),
  child: TextField(
    controller: _passwordConfirm,
    decoration: InputDecoration(
        border: OutlineInputBorder(
          borderSide:
              BorderSide(color: Colors.grey, width: 0.1),
          borderRadius: BorderRadius.circular(40),
        ),
        labelStyle: TextStyle(fontSize: 20),
        labelText: 'Подтвержение пароля',
        prefixIcon: Icon(Icons.vpn_key, size: 24),
        hintText: 'Подтвердите пароль'),
  ),
),
Padding(
  padding: EdgeInsets.only(top: 20),
  child: RaisedButton(
    padding:
        EdgeInsets.symmetric(vertical: 10, horizontal: 30),
    shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(40)),
    color: Theme.of(context).primaryColor,
    child: Text(
      'Зарегистрироваться',
      style: TextStyle(
        color: Colors.white,
        fontSize: 20,
      ),
    ),
    onPressed: () {},
  ),
),
FlatButton(
    onPressed: () {
      Navigator.of(context).push(
        MaterialPageRoute(
          builder: (context) => LoginScreen(),
        ),
      );
    },
    child: Text('Войти'))],),),),)),);}}
```

```
import 'package:flutter/material.dart';
class LoginScreen extends StatelessWidget {
  TextEditingController _login = TextEditingController();
  TextEditingController _password = TextEditingController();
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        top: true,
        bottom: true,
        left: true,
        right: true,
        child: Container(
          padding: EdgeInsets.only(top: 200, left: 30, right: 30),
          child: SingleChildScrollView(
            child: Center(
              child: Column(
                children: <Widget>[
                  Padding(
                    padding: const EdgeInsets.all(20.0),
                    child: Text(
                      'Войти',
                      style: TextStyle(
                        color: Colors.red[400],
                        fontSize: 26,
                      ),
                    ),
                  ),
                  Padding(
                      padding: EdgeInsets.all(10),
                      child: TextField(
                        controller: _password,
                        decoration: InputDecoration(
                            border: OutlineInputBorder(
                              borderSide:
                                  BorderSide(color: Colors.grey, width: 0.1),
                              borderRadius: BorderRadius.circular(40),
                            ),
                            labelStyle: TextStyle(fontSize: 20),
                            labelText: 'Логин',
                            prefixIcon: Icon(Icons.perm_identity, size: 24),
                            hintText: 'Введите логин'),)),
```

```
                Padding(
                  padding: EdgeInsets.all(10),
                  child: TextField(
                    controller: _password,
                    decoration: InputDecoration(
                        border: OutlineInputBorder(
                          borderSide:
                              BorderSide(color: Colors.grey, width: 0.1),
                          borderRadius: BorderRadius.circular(40),
                        ),
                        labelText: 'Пароль',
                        labelStyle: TextStyle(fontSize: 20),
                        prefixIcon: Icon(
                          Icons.vpn_key,
                          size: 24,
                        ),
                        hintText: 'Введите пароль'),
                  ),
                ),
                Padding(
                  padding: EdgeInsets.only(top: 20),
                  child: RaisedButton(
                    padding:
                        EdgeInsets.symmetric(vertical: 10, horizontal: 50),
                    shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(40)),
                    color: Theme.of(context).primaryColor,
                    child: Text(
                      'Войти',
                      style: TextStyle(color: Colors.white, fontSize: 24),
                    ),
                    onPressed: () {},
                  ),
                ),
                FlatButton(
                    onPressed: () {
                      Navigator.of(context).push(
                        MaterialPageRoute(
                          builder: (context) => RegisterScreen(),
                        ),);},
                    child: Text('Регистрация'))],),),),),),);}
}
```

```
import 'package:flutter/material.dart';
class CategoryScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Категории'),
      ),
      body: Container(
        padding: EdgeInsets.symmetric(
          vertical: 10,
          horizontal: 20
        ),
        child: Column(
          children: <Widget>[
            ListTile(
              title: Text('Верхняя одежда'),
              trailing: Icon(Icons.navigate_next),
            ),
            Divider(),
            ListTile(
              title: Text('Топы'),
              trailing: Icon(Icons.navigate_next),
            ),
            Divider(),
            ListTile(
              title: Text('Брюки'),
              trailing: Icon(Icons.navigate_next),
            ),
            Divider(),
            ListTile(
              title: Text('Юбки'),
              trailing: Icon(Icons.navigate_next),
            ),
            Divider(),
            ListTile(
              title: Text('Платья'),
              trailing: Icon(Icons.navigate_next),
            ),
            Divider(),
            ListTile(
              title: Text('Аксессуары'),
```

```
                trailing: Icon(Icons.navigate_next),
              ),
            ],
          ),
        ),
      );
    }
  }

  import 'package:flutter/material.dart';

  class ProductDetail extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
      return Scaffold(
        body: SingleChildScrollView(
          child: Column(
            children: <Widget>[
  Image.network("https://www.uniqlo.com/ru/estore/hmall/test/u000000000292
9/main/first/561/1.jpg"),
              SizedBox(
                height: 10,
              ),
              Container(
                margin: EdgeInsets.symmetric(horizontal: 20),
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.stretch,
                  children: <Widget>[
                    Text(
                      '424636 Жакет',
                      style: TextStyle(
                        fontSize: 24,
                      ),
                    ),
                    SizedBox(height: 10,),
                    Text(
                      'Описание',
                      style: TextStyle(
                        fontSize: 20,
                        color: Colors.black54
                      ),
                    ),
```

```
                SizedBox(height: 10,),
                Text(
                 'Легкая и удобная куртка, которую также можно носить в качестве кардигана.',
                  style: TextStyle(
                    fontSize: 16,
                  ),
                ),
                SizedBox(height: 10,),
                Text(
                  'Цена',
                  style: TextStyle(
                    fontSize: 20,
                    color: Colors.black54
                  ),
                ),
                SizedBox(height: 10,),
                Text(
                  '12 000 тенге',
                  style: TextStyle(
                    fontSize: 24,
                  ),
                ),
              ],
            ),
          ),
        ],
      ),
    ),
    floatingActionButton: FloatingActionButton(child: Icon(Icons.add_shopping_cart),onPressed: () {}),
  );
 }
}
```

Приложение Б
Акт внедрения

Утверждаю
Директор ИП «Uni»
Чупаченко М. Б.
«19» мая 2020г.

АКТ ВНЕДРЕНИЯ

Настоящий акт составлен о том, что результаты выпускной работы студента НАО АУЭС группы ИСу-17-3 очной формы обучения Омарова Ислама под руководством кандидата технических наук и доцента кафедры «IT-инжиниринг» Мамыровой А.К. на тему «Разработка мобильного приложения «Unimag» на базе ОС Android» была передана в эксплуатацию ИП «Uni» в мае 2020 года для использования в качестве программного комплекса.

Использование результата выпускной работы Омарова И. обеспечивает доступ к информации о магазине, а также позволяет совершать все ключевые функции (поиск товара, добавление в "избранное", сравнение, оформление заказа) без использования ПК.

Внедрение программного продукта «Unimag» решает также следующий ряд задач:

- автоматизацию работы администратора, а именно, управление всей базой данных магазина со смартфона;
- увеличение среднего чека и повышение клиентской лояльности;
- повышение эффективности коммуникаций с клиентами при помощи push-уведомлений.

Директор ИП «Uni» Чупаченко М. Б.

Исполнитель Омаров И.О.