Министерство образования Новосибирской области ГБПОУ НСО «Новосибирский авиационный технический колледж имени Б.С. Галущака»

Разработка мобильного приложения для цветочного МАГАЗИНА

Пояснительная записка к курсовому проекту

МДК.01.03 Разработка мобильных приложений

НАТКиГ.201400.010.000ПЗ

Разработала:

студентка группы ПР-21.101

Ляпунова Е. Д.

2024

**Содержание**

[Введение 3](#_Toc165140044)

[1 Исследовательский раздел 4](#_Toc165140045)

[1.1 Описание предметной области 4](#_Toc165140046)

[1.2 Образ клиента 5](#_Toc165140047)

[1.3 Сценарии 6](#_Toc165140048)

[1.4 Сбор и анализ прототипов 7](#_Toc165140049)

[2 Проектирования приложения 16](#_Toc165140050)

[2.1 UI/UX дизайн проекта 16](#_Toc165140051)

[2.2 Выбор технологии, языка и среды программирования 17](#_Toc165140052)

[3 Разработка мобильного приложения 19](#_Toc165140053)

[3.1 Разработка базы данных 19](#_Toc165140054)

[3.2 Разработка мультимедийного контента 21](#_Toc165140055)

[3.3 Описание используемых плагинов 22](#_Toc165140056)

[3.4 Описание разработанных процедур и функций 23](#_Toc165140057)

[4 Тестирование 27](#_Toc165140058)

[4.1 Протокол тестирования дизайна приложения 27](#_Toc165140059)

[4.2 Протокол тестирования функционала приложения 32](#_Toc165140060)

[Заключение 36](#_Toc165140061)

[Список используемых ресурсов 37](#_Toc165140062)

[Приложение А (обязательное) Техническое задание 39](#_Toc165140063)

[Приложение Б (справочное) Макет приложения 47](#_Toc165140064)

[Приложение В (справочное) Листинг 52](#_Toc165140065)

Введение

Цветочные магазины стали неотъемлемой частью нашей жизни, добавляя в нее красоту и эстетическое удовольствие. Однако в нашем современном и быстро меняющемся мире, требуется идти в ногу со временем и предлагать клиентам новые и удобные способы совершать покупки и получать услуги.

Актуальность данной темы заключается в необходимости адаптации цветочных салонов к современным технологиям и ожиданиям клиентов. Предложенное мобильное приложение представляет собой инновационный способ расширения клиентской базы и повышения удобства обслуживания.

Целью курсового проекта будет разработка мобильного приложения, специально созданного для цветочного салона. Это приложение предоставит возможность клиентам легко, быстро и удобно заказывать цветы, букеты и другие цветочные композиции, а также получать информацию о наличии товаров, дополнительных услугах, акциях и событиях, связанных с салоном.

Также будет учтена важность визуального впечатления и эстетического оформления. Интуитивно понятный и привлекательный дизайн приложения добавит комфорта при использовании, а также подчеркнет красоту самого процесса выбора и заказа цветов.

Задачами курсового проекта являются:

* рассмотреть предметную область;
* проанализировать существующие подобные приложения;
* определить этапы разработки мобильного приложения;
* смоделировать интерфейс приложения;
* разработать функционал приложения;
* провести тестирования мобильного приложения.

Реализуемый продукт – приложения для цветочного магазина.

1 Исследовательский раздел

## Описание предметной области

В современном мире цветочные магазины занимают особое место среди торговых точек, предлагающих свежие цветы и цветочные композиции. Распространенные в городских и сельских населенных пунктах, эти магазины являются не только местом продажи цветов, но и местом, где клиенты могут найти вдохновение, подарить радость и красоту своим близким.

Цветочный салон – это специализированный магазин, где предлагается широкий выбор готовых букетов для различных событий, таких как день рождения, свадьба, или просто для того, чтобы порадовать себя или близких. Кроме того, в салоне предлагается услуга оформления цветочных композиций для различных мероприятий и также продажа других декоративных элементов.

Цветочные магазины обычно тщательно подбирают своих поставщиков, чтобы гарантировать свежесть и качество предлагаемых цветов и растений. Сотрудники салона делают все возможное, чтобы удовлетворить потребности клиентов и создать особую атмосферу, которая делает посещение салон приятным и запоминающимся.

Важной частью работы цветочного салона является уход за растениями и цветами, чтобы сохранить их свежесть и красоту. Профессиональные флористы и садоводы следят за состоянием цветов, предлагая клиентам только лучшее качество. Регулярный полив, обрезка, удобрение и контроль за состоянием здоровья растений играют ключевую роль в поддержании уникальности цветочного салона.

Флористы цветочного салона творчески подходят к созданию букетов и композиций, учитывая вкусы и предпочтения каждого клиента. Эксперты по цветам подбирают сочетания цветов, форм и текстур, чтобы создать уникальные и элегантные композиции, которые будут радовать глаз и душу получателя.

Одной из особенностей цветочного салона является индивидуальный подход к каждому клиенту. Профессиональные консультанты помогают подобрать подходящие цветочные композиции для любого случая: от праздничного подарка до оформления свадебной церемонии. Клиенты ценят внимание к деталям и заботу, которую они получают в цветочном салоне.

С развитием онлайн – торговли цветами, важным аспектом для цветочного магазина становится создание электронного магазина с удобным интерфейсом для заказа букетов и доставки. Креативные решения в оформлении витрин, разнообразие предложений и услуг, а также забота о сохранении окружающей среды путем использования экологически чистых материалов – все это важные тенденции, влияющие на деятельность цветочных магазинов.

## Образ клиента

Всего мы рассмотрим 4 клиентских образов:

* Анне 30 лет – занятая работающая женщина. Она хочет найти удобный способ заказа красивого букета цветов для своей подруги, мамы и коллеги, а также следить за акциями и новинками в цветочном салоне в свободное от работы время;
* Михаил 25 лет – молодой бизнесмен. Он заинтересован в регулярных заказах цветов для офиса, проведения мероприятий и подарков для партнеров. Он ищет удобный способ выбора и заказа крупных партий цветов с доставкой;
* Елена 40 лет – домохозяйка, любительница цветов. Она увлечена уходом за растениями и хочет найти новые уникальные экземпляры, а также просто порадовать себя красивыми букетами. Она заинтересована в получении уведомлений о новых поступлениях и акциях;
* Денис 28 лет – романтик. Он часто делает подарки своей девушке и хочет иметь быстрый доступ к описанию букетов, возможности добавления поздравительной открытки или других персонализированных элементов заказа.

## Сценарии

В данном разделе мы рассмотрим сценарии с каждым образом клиента:

* Анна заметила, что у ее лучшей подруги скоро день рождения. Она хочет приготовить ей приятный сюрприз, но у Анны очень мало свободного времени из-за работы. Она открывает мобильное приложение цветочного салона, рассматривает разнообразные букеты и использовав функцию быстрого заказа, отправляет прекрасный букет цветов с персональной открыткой своей подруге прямо на дом;
* Михаил организует корпоративное мероприятие в своем офисе и хочет украсить его стильными цветами. Он запускает приложение цветочного салона, осматривает различные варианты для больших заказов цветов на свой вкус, выбирает несколько опций и сразу оформляет заказ на доставку в офис;
* Елена просыпается в выходной день и решает, что ей нужно освежить интерьер своего дома цветами. Выбирая новые растения для своей коллекции, она открывает приложение цветочного салона, просматривает каталог и использует возможность резервирования редких растений, чтобы затем забрать их в магазине;
* Денис понял, что скоро День святого Валентина, и он хочет сделать ей особенный подарок. Он заходит в приложение цветочного салона, выбирает стильный и романтичный букет, добавляет персональное послание и выбирает опцию доставки к указанному времени.

## Сбор и анализ прототипов

В данном сегменте разработано достаточно большое количество приложений. Воспользуемся Google Play для рассмотрения дизайна и функционала приложений.

Первым на рассмотрении будет приложение «Flor2u». Ниже на рисунке 1 представлен логотип приложения.

Рисунок 1 – Логотип приложения «Flor2u»

После запуска приложения открывает главный экран, который приведен на рисунке 2. В дизайне приложения преобладает белый и зеленый цвета. На карточке каждого букета приводиться следующая информация: рейтинг по отзывам пользователей, наименование, доступные цвета букета, минимальная стоимость и кнопка с переходом в карточку товара.

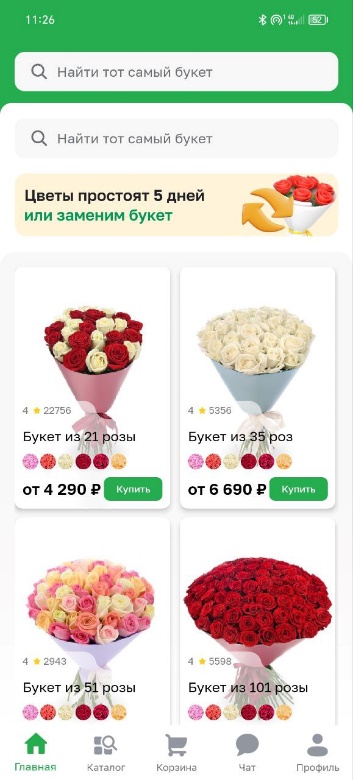


Рисунок 2 – Главный экран приложения «Flor2u»

На нижней панели расположено 5 кнопок с соответствующими иконками и надписями. Переходя с главного экрана на букет роз, открывается описание букета и все дополнительные услуги, данный экран приведен в рисунке 3.

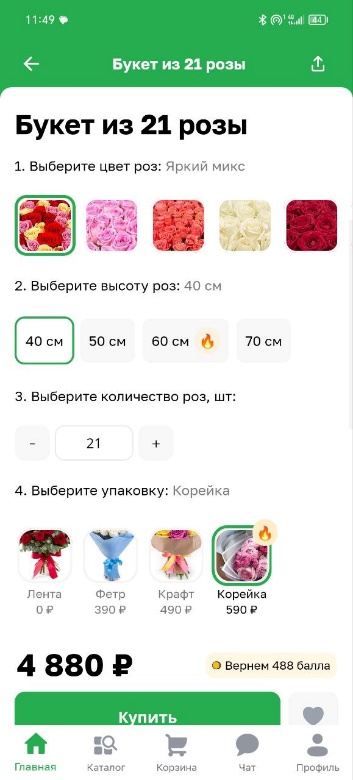


Рисунок 3 – Карточка букета роз

В карточке товара представлено много информации и выбора. Так пользователь может выбрать цвет, высоту роз, а также их количество и оформление букета. Для других видов цветов в приложении немного другой интерфейс, представленный в рисунке 4.

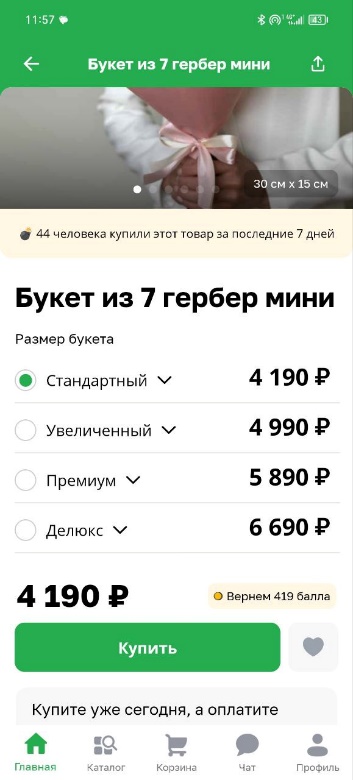


Рисунок 4 – Карточка букета гербер мини

В данной карточке уже отсутствует большой выбор, а дано 4 вида размера букета: стандартный, увеличенный, премиум и делюкс, а также их стоимость. Также открывает подробное описание количества цветов в выбранному размере.

Следующей страницей является каталог, представленный на рисунке 5.

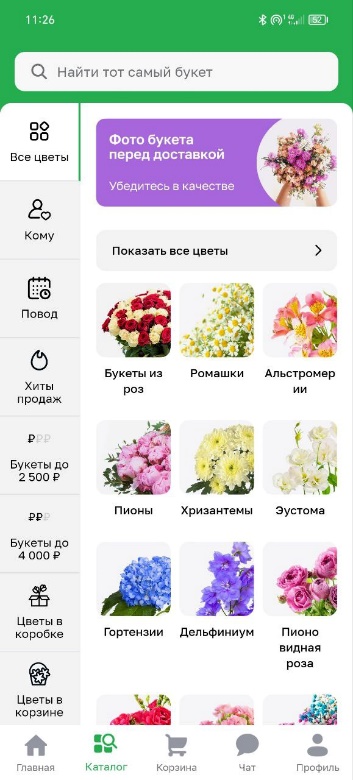


Рисунок 5 –Каталог приложения

Выбрав конкретную категорию нас переносит на экран с каталогом цветов и возможностью сортировки по возрастанию и убыванию цены, а также популярности. А также есть функция фильтра, где можно задать конкретный ценовой диапазон, выбрать конкретные цветы, категорию букета или же его высоту.

При переходе на вкладку корзина, где есть возможность добавить топперы к букету и оформить заказ, это всё представлено на рисунке 6.

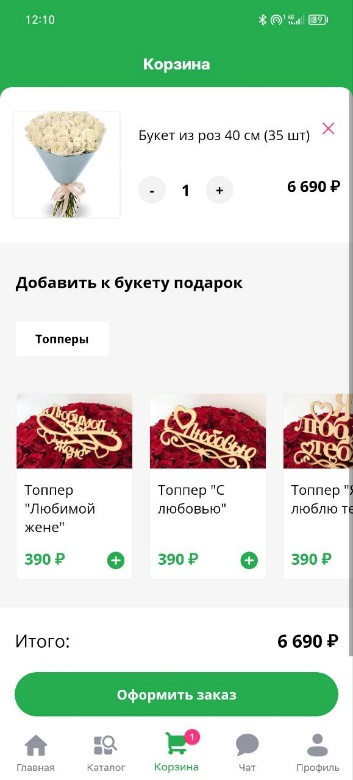


Рисунок 6 – Корзина приложения

В приложении также есть вкладки «Чат» и «Профиль». Вкладка «Профиль» приведен в рисунке 7.

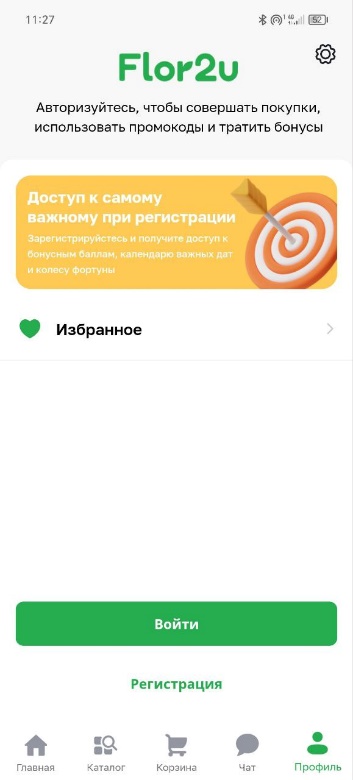


Рисунок 7 – Вкладка «Профиль»

Клиент может зарегистрироваться на выбор либо через номер телефона, либо через электронную почту. Данный функционал представлен на картинке 8.



Рисунок 8 – Регистрация

После внесения номер телефона, был получен код, этот процесс отображен на рисунке 9.

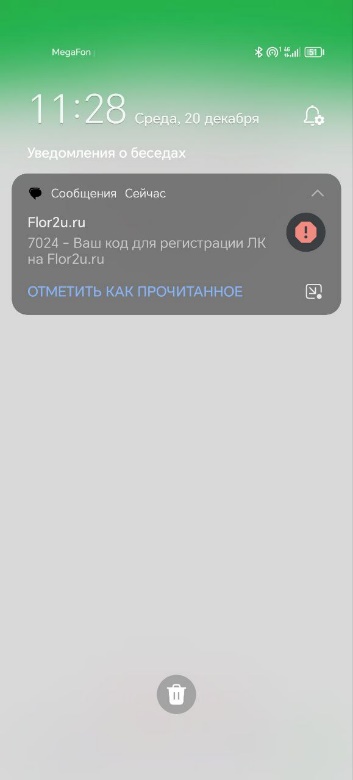


Рисунок 9 – Получение кода для регистрации в приложении

Но при внесении этого кода в поле, зарегистрироваться не получилось. Данная ошибка отражен в рисунке 10.



Рисунок 10 – Внесение кода из СМС

Вывод по анализу приложения.

Достоинства:

* хорошо прописаны карточки товара, множество информации;
* интуитивно приятный и понятный интерфейс.

Недостатки:

* излишнее количество информации на главной странице;
* не осуществляется проверка кода для регистрации пользователя.

Вторым на рассмотрении будет приложение «Flor2u». Ниже на рисунке 11 представлен логотип приложения.



Рисунок 11 – Логотип приложения «FloraOpt»

В приложении преобладает белый цвет с красными элементами. На рисунке 12 представлен главный экран приложения, на котором расположен выбор города, поиск, объявление и подборки букетов.

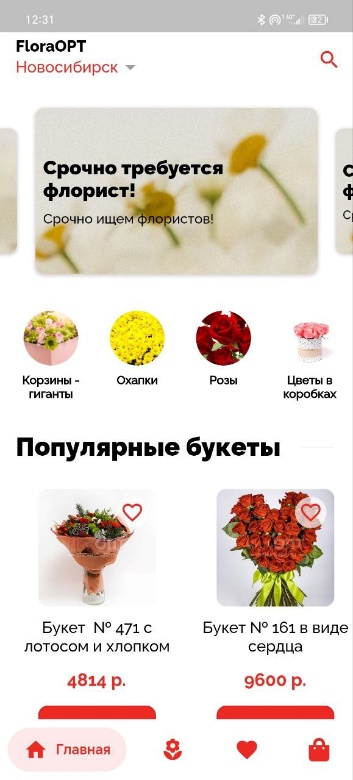


Рисунок 12 – Главный экран приложения

При переходе в карточку конкретного букета нас встречает несколько фотографий букета, его цена и кнопка добавления его в корзину. Это приведено на рисунке 13. Заметим, что в карточке представлена самая минимальная информация.



Рисунок 13 – Карточка товара

Далее переходим в каталог, который представлен на рисунке 14. В каталоге небольшое количество категорий. Не у всех категорий присутствуют иконки, что является визуально не красиво.

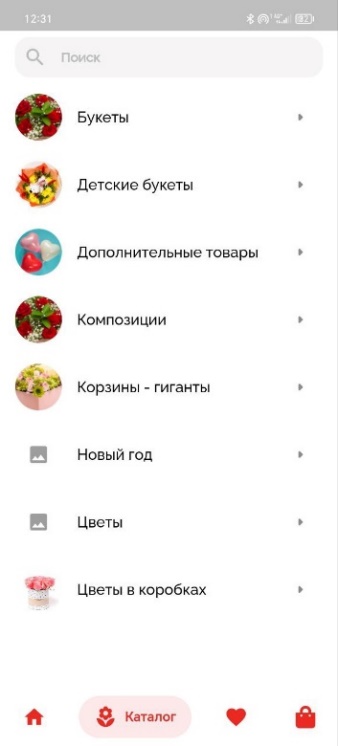


Рисунок 14 – Каталог приложения

При добавлении букета в корзину мы переходим во вкладку «Корзина», это представлено на рисунке 15. Видим, что вариативность действий небольшая, мы можем только увеличить количество букета и оформить заказ, а также ввести промокод.



Рисунок 15 – Вкладка «Корзина»

Перейдя во вкладку «Кабинет», которая отображена на рисунке 16, у нас есть возможность войти только по номеру телефона, с помощью получения кода через СМС. А также присутствует вкладка избранное даже для не авторизовавшихся клиентов, контактная информация и информация о приложении.

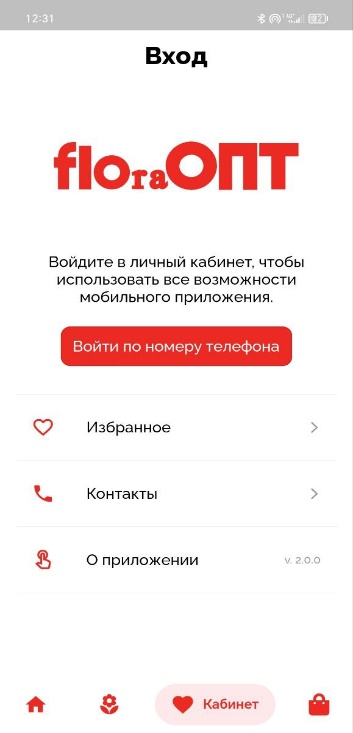


Рисунок 16 – Вкладка «Кабинет»

Во время анализа мобильного приложения «FloraOPT» были сделаны выводы и учтены некоторые моменты.

Достоинства:

* визуально приятный интерфейс;
* в меру представлена информация на главном экране.

Недостатки:

* мало дополнительной информации в карточке товара;
* отсутствие добавления украшений и другого декора при оформлении заказа.

# Проектирования приложения

## UI/UX дизайн проекта

Для проекта были определены основные экраны:

* главный экран;
* каталог;
* избранное;
* корзина;
* регистрация;
* авторизация;
* оформление заказа;
* личный кабинет.

Для подбора цветовой палитры важно учитывать взаимодействие различных оттенков и их влияние на общее восприятие дизайна, поэтому были выбраны следующие цвета:

* цвет #CBF0F1 является основным цветом, который придает ощущение свежести, нейтральности и спокойства. Этот оттенок голубого обычно ассоциируется с миром природы, легкостью и чистотой, что может создать атмосферу умиротворения и гармонии.
* цвет #F3D250 добавляет контраст и яркость к цветовой палитре. Желтый цвет символизирует энергию, радость и оптимизм, создавая живой акцент и привлекая внимание к деталям. В сочетании с голубым, желтый цвет придает дополнительную динамику и свежесть.
* цвет #7DA2A9 мягко дополняет палитру, придавая ей стиль и сдержанность. Серый цвет часто используется для контраста и создания гармонии между яркими и более нейтральными оттенками. Он способен уравновесить цветовую гамму и придать дизайну более спокойный характер.

Все три цвета вместе образуют удачную комбинацию, подчеркивающую стиль, элегантность и индивидуальность. Правильно подобранная цветовая палитра способствует созданию гармоничного и привлекательного дизайна, который будет привлекать внимание и вызывать положительные эмоции у зрителей.

Логотип приложения разработан в программе в программе «Photoshop», представлен на рисунке 17. Программа «Photoshop» была выбрана для разработки логотипа приложения из-за своего профессионализма, гибкости и возможности работы с изображениями высокого разрешения.



Рисунок 17 – Логотип

Дизайн проекта был разработан в «Figma» благодаря его удобному интерфейсу и широкому набору инструментов для создания пользовательских интерфейсов. «Figma» позволяет легко создавать прототипы и тестировать интерактивность элементов интерфейса.

В приложении Б представлен макет приложения. В макете приложения разработано 10 экранов. Взаимодействие между экранами происходит с помощью кнопок.

## Выбор технологии, языка и среды программирования

Для разработки проекта были выбраны следующие технологии, язык программирования и среда разработки:

* «Android Studio». «Android Studio» является официальной интегрированной средой разработки (IDE) для разработки мобильных приложений под платформу Android. Она предоставляет разработчикам широкий набор инструментов, редактор кода, эмулятор устройств и многое другое, что упрощает процесс разработки и отладки приложений под Android;
* Java. «Java» является одним из самых популярных языков программирования для разработки Android-приложений. Он обладает простым синтаксисом, обширным сообществом разработчиков и поддерживается Android SDK, что делает его отличным выбором для создания приложений под платформу Android;
* «Firebase Authentication» по номеру телефона. «Firebase Authentication» обеспечивает безопасную и простую аутентификацию пользователей в приложении. Использование аутентификации по номеру телефона позволяет пользователям войти в приложение с помощью их мобильных номеров, что обеспечивает удобство и безопасность процесса входа;
* «Firebase Storage» для хранения изображений букетов. «Firebase Storage» обеспечивает надежное хранилище для файлов, таких как изображения. Использование «Firebase Storage» для хранения изображений букетов позволяет эффективно управлять и загружать изображения в приложении, обеспечивая быстрый доступ к ним;
* «Cloud Firestore» для синхронизации и хранения данных. «Cloud Firestore» является гибкой и масштабируемой базой данных в облаке, предоставляемой Firebase. Использование «Cloud Firestore» для синхронизации данных позволяет хранить и синхронизировать информацию о букетах, заказах и других данным между различными устройствами пользователей, обеспечивая надежность и масштабируемость приложения.

Выбор данных технологий и инструментов обусловлен их популярностью, удобством использования, масштабируемостью и возможностью интеграции для создания функционального и производительного мобильного приложения для цветочного магазина под платформу Android.

# Разработка мобильного приложения

## Разработка базы данных

В проекте используется сервис «Firebase Authentication» по номеру телефона, который представлен на рисунке 18. Это обеспечивает удобный и безопасный способ аутентификации пользователей через их мобильные номера, повышая удобство для пользователей и защищая их данные от несанкционированного доступа.

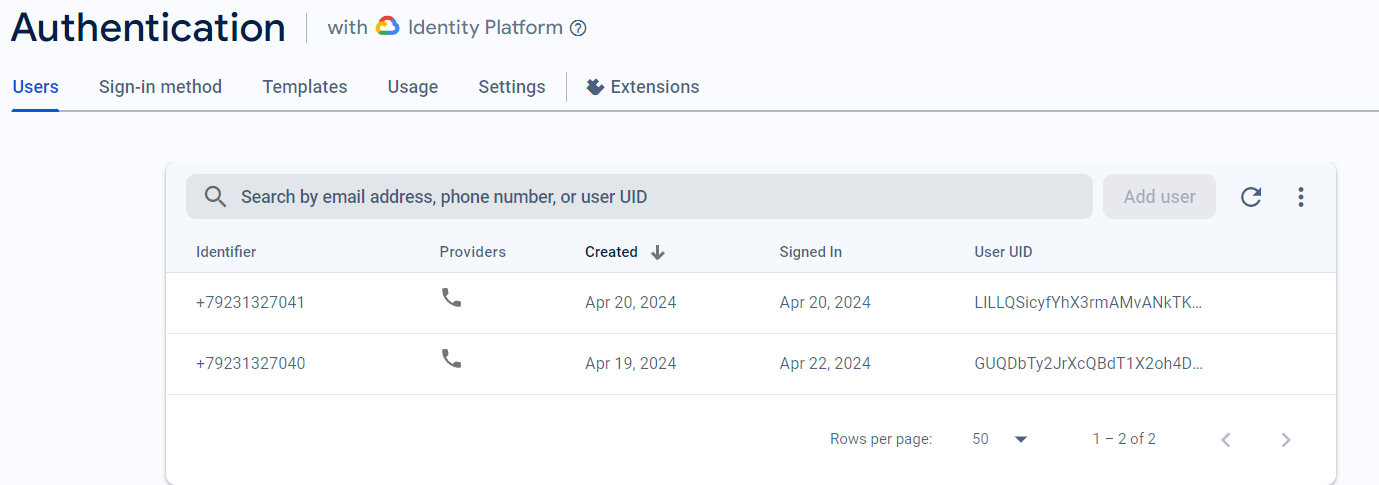


Рисунок 18 – Сервис «Firebase Authentication»

Выбор «Cloud Firestore» в качестве базы данных для приложения в «Android Studio» обусловлен:

* гибкой структурой базы данных и масштабируемостью;
* реальной синхронизацией данных и отзывчивым пользовательским опытом;
* автоматическим масштабированием, высокой доступностью и надежностью данных;
* интеграцией с «Android Studio» и поддержкой оффлайн-режима. Это обеспечивает надежное хранение данных цветочного магазина, включая информацию о букетах, пользователях, магазинах, изображениях и личных данных.

Исходя из анализа предметной области были созданы следующие сущности:

* «Пользователи», которая состоит из имени, номера телефона и даты рождения пользователя. Предназначена для хранения личных данных о пользователе;
* «Букеты», в которой содержится такая информация как: наименование, описание, стоимость за единицу товара, ссылка на изображение, которое храниться в «Firebase Storage», категорию и состав букета. Предназначена для хранение всей информацией о букетах;
* «Категории», которая содержит только наименование. Предназначена для распределения букетов;
* «Список цветов», в которой также содержится наименование цветка. Предназначена для формирования состава каждого букета;
* «Магазины», в которой представлен адрес каждого магазин и его контактная информация, такая как номер телефон и электронная почта, а также время работы. Предназначена для ознакомления клиента и возможностью оформить заказ с самовывозом в конкретный магазин;
* «Тип доставки», содержит в себе только наименование. Предназначена для хранения выбора клиента при оформлении заказа;
* «Время для самовывоза», содержит в себе информацию о магазине и времени самовывоза из него. Предназначена для оформления заказа клиентом с выбором нужного места и времени для самовывоза;
* «Время для доставки», которая хранит только информацию о интервалах доставки. Предназначена для выбора удобного времени доставки клиентом при оформлении заказа с доставкой;
* «Оформление заказа», содержит в себе такие данные как: тип доставки, данные пользователя, дата оформления заказа, итоговую стоимость заказа и комментарии от клиента. Предназначена для начального хранения всей общей информации о заказе;
* «Доставка», в которой содержится информация о сформированном ранее заказа, адресе клиента, желаемая дата и время доставки. Данная сущность будет хранить в себе подробную информацию о заказах с доставкой;
* «Самовывоз», содержит в себе ранее сформированный заказ, выбранный магазин клиентом и желаемая дата и время самовывоза. Предназначена для хранения подробной информации о заказ с самовывозом;
* «Состав заказа», в которой сохраняется информация о заказе и букете, который был выбран. Позволяет сохранять информацию о заказанных букетах.

Структура всех сущностей показа на рисунке 19.

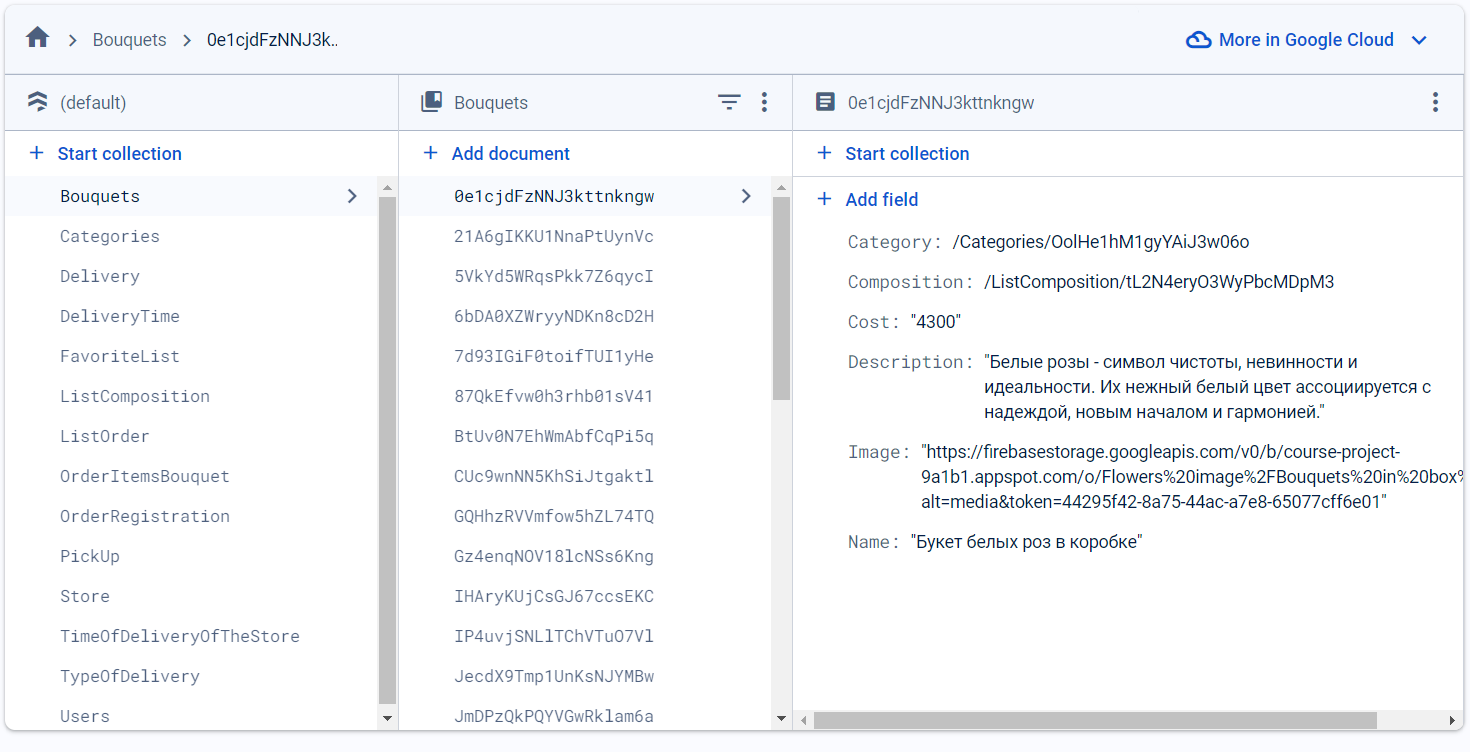


Рисунок 19 – Структура коллекция в «Cloud Firestore»

## Разработка мультимедийного контента

Для разработки мультимедийного контента в проекте было решено использовать язык разметки XML по следующим причинам:

* структурированность и удобство. XML обеспечивает простую структуру данных, что удобно для описания мультимедийного контента, такого как макеты интерфейса, цветовая палитра и другие элементы дизайна;
* поддержка различных устройств. XML позволяет создавать масштабируемые макеты и контент, что важно для адаптации приложения под разные разрешения экранов и устройств.

Определение основных цветов в отдельном файле «colors.xml» упрощает изменение цветовой схемы приложения, обеспечивает ее единообразие и удобство при внесении изменений.

Размещение всех изображений, иконок и графических ресурсов в папках «drawable» и «mipmap» делает управление этими ресурсами более удобным. Разделение по папкам помогает легко находить и использовать нужные изображения в приложении.

Проект реализован исключительно в светлой теме, что создает стильный и согласованный дизайн для пользователей приложения.

## Описание используемых плагинов

Для разработки были использованы встраиваемые плагины, которые создаются вместе с проектом, а также дополнительный плагины, которые нужны для функциональности приложения, представленные в таблице 1.

Таблица 1 – Библиотеки приложения и их описание

|  |  |
| --- | --- |
| Библиотека | Описание |
| 1 | 2 |
| com.google.android.gms:play-services-safetynet:18.0.1 | Библиотека используется для работы с SafetyNet API, который предоставляет функции безопасности на устройствах Android |
| com.google.firebase:firebase-auth:22.3.1 | Библиотека Firebase Authentication для аутентификации пользователей |
| com.google.firebase:firebase-firestore:24.11.0 | Библиотека Firebase Firestore для работы с облачной базой данных Firestore |
| com.squareup.picasso:picasso:2.71828 | Библиотека Picasso для загрузки и отображения изображений |
| com.google.firebase:firebase-analytics:21.6.1 | Библиотека Firebase Analytics для аналитики и отслеживания событий |
| androidx.recyclerview:recyclerview:1.3.2 | Библиотека RecyclerView для отображения списков данных. |
| com.google.code.gson:gson:2.10.1 | Библиотека Gson для работы с JSON в Java. |

Окончание таблицы 1

|  |  |
| --- | --- |
| 1 | 2 |
| androidx.constraintlayout:constraintlayout:2.1.4" | Библиотека ConstraintLayout для создания сложных макетов пользовательского интерфейса |
| androidx.legacy:legacy-support-v4:1.0.0 | Библиотека Legacy Support для обеспечения обратной совместимости |
| com.google.android.material:material:1.11.0 | Библиотека Material Design для создания стильного пользовательского интерфейса |
| androidx.appcompat:appcompat:1.6.1 | Библиотека для работы с AppCompat компонентами |

Выбранные библиотеки играют ключевую роль в разработке приложения: от обеспечения безопасности пользователей и работы с базами данных в реальном времени до улучшения пользовательского интерфейса и аналитики событий. Они помогают упростить разработку, повысить производительность и взаимодействие приложения с пользователем, делая его более удобным и функциональным.

## Описание разработанных процедур и функций

В приложении реализованы следующие методы, представленные в таблице 2.

Таблица 2 – Методы приложения

|  |  |
| --- | --- |
| Метод | Описание |
| 1 | 2 |
| addBouquetToFavorites() | Метод для добавления букета в избранное |
| removeBouquetFromFavorites() | Метод для удаления букета из избранного |
| fetchFavoriteBouquetsByUser() | Метод для получения содержимого вкладки избранного |
| addToOrderCart() | Метод для добавления букета в корзину |
| getCost() | Метод для получения стоимости букетов в корзине |
| fetchBouquetQuantitiesByUser() | Метод для получения содержимого корзины |
| navigateToFavorite() | Метод, с помощью которого перераспределяется пользователь во вкладки, в зависимости от того авторизирован ли он |
| deleteAccount() | Метод удаления профиля пользователя из базы данных |
| logoutUser() | Метод для выхода из аккаунта пользователя |

Окончание таблицы 2

|  |  |
| --- | --- |
| 1 | 2 |
| signInWithPhoneAuthCredential() | Метод проверки кода для регистрации пользователя и сохранения данных в базу данных |
| loadBouquetsByCategory() | Метод для определения выбранной пользователем категории букета и вывод списка букетов по этой категории |
| registation\_order() | Метод оформления заказа |
| getBouquetById() | Метод вывод информации в карточку товара по выбранному букету |
| PhoneAuthProvider.OnVerificationStateChangedCallbacks | Метод отправления кода по номеру телефона |
| handleVerificationFailure() | Метод обработки ошибок верификации телефона |

В приложении реализованы разнообразные методы, представленные в таблице 2, которые обеспечивают функциональность добавления и удаления букетов в избранное, управление содержимым корзины, авторизацию пользователя, обработку заказов и другие важные операции. Эти методы значительно обогащают пользовательский опыт, обеспечивая удобство использования приложения и эффективное взаимодействие с данными и функционалом.

В приложении реализованы адаптеры, часть которых прописаны в таблице 3.

Таблица 3 – Адаптеры приложения

|  |  |  |  |
| --- | --- | --- | --- |
| Адаптер | Назначение | Функциональность | Особенности |
| 1 | 2 | 3 | 4 |
| AdapterOrder | Адаптер, используемый для отображения списка букетов в корзине заказов. | Позволяет пользователям просматривать букеты в корзине, управлять количеством их приобретения, добавлять и удалять букеты из корзины, а также отображать общую стоимость заказа. | Включает в себя обработчики нажатий для кнопок увеличения, уменьшения количества и удаления букетов, а также отслеживание и обновление стоимости заказа. |

Окончание таблицы 3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | | 3 | | 4 | |
| BouquetsAdapter | | Адаптер для отображения списка букетов в магазине цветов. | | Обеспечивает отображение имени, изображения и стоимости каждого букета, а также позволяет пользователям добавлять букеты в корзину или отмечать их как избранные. | | Включает функционал добавления букетов в корзину и избранное, обработку кликабельных ссылок для набора телефона и отправки электронной почты, а также отображение статуса избранного для каждого букета. |
| StoreAdapter | | Адаптер для отображения списка магазинов цветов с их контактной информацией. | | Предоставляет информацию об адресе, телефоне, почте и времени работы каждого магазина, а также включает функционал для набора телефона или отправки электронной почты. | Включает обработку кликабельных ссылок для набора телефона и отправки электронной почты, что позволяет пользователям легко связываться с магазинами. | |

В приложении B представлен адаптер «BouquetsAdapter» для работы с «RecyclerView». Этот адаптер отображает список букетов. В адаптере реализованы следующие функции:

* инициализация списка букетов и списков избранных и купленных букетов;
* обработка нажатий на элементы списка, добавление букетов в корзину покупок и избранное;
* отслеживание состояния избранного для каждого букета;
* реализация функций добавления букетов в корзину и в избранное, удаления букетов из избранного;
* получение избранных букетов для текущего пользователя.

А также в адаптере реализован интерфейс «FavoritesListener» для обработки нажатий на избранные букеты.

Адаптеры способствуют эффективному отображению и управлению информацией, которая хранится в базе данных, в различных элементах пользовательского интерфейса. Они позволяют динамически загружать и отображать данные, обновлять содержимое списков и карточек при необходимости, а также обрабатывать действия пользователя, такие как добавление в корзину, сохранение в избранное или удаление позиции. Их использование обеспечивает удобство и гибкость взаимодействия пользователей с приложением и предоставляет возможность создания интерактивного и отзывчивого пользовательского опыта.

# Тестирование

## Протокол тестирования дизайна приложения

Для проведения тестирования дизайна приложения были взяты два эмулятора, представленные на рисунке 20.

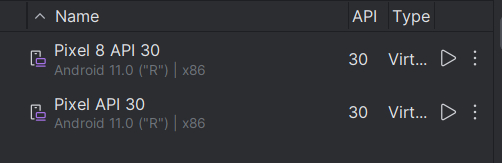


Рисунок 20 – Список эмуляторов для тестирования

Проводились тесты основных окон приложения, которые показаны на рисунках 21-29.

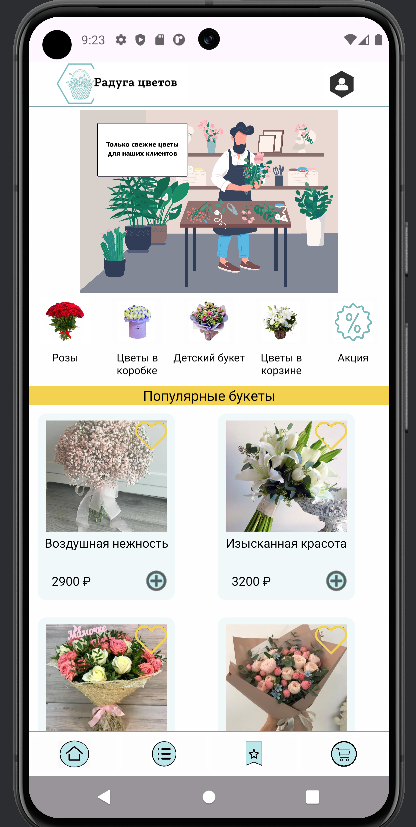
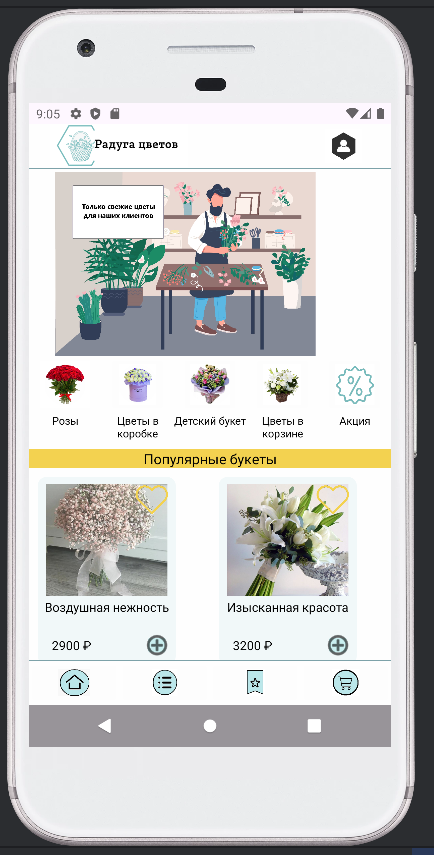


Рисунок 21 – Главный экран

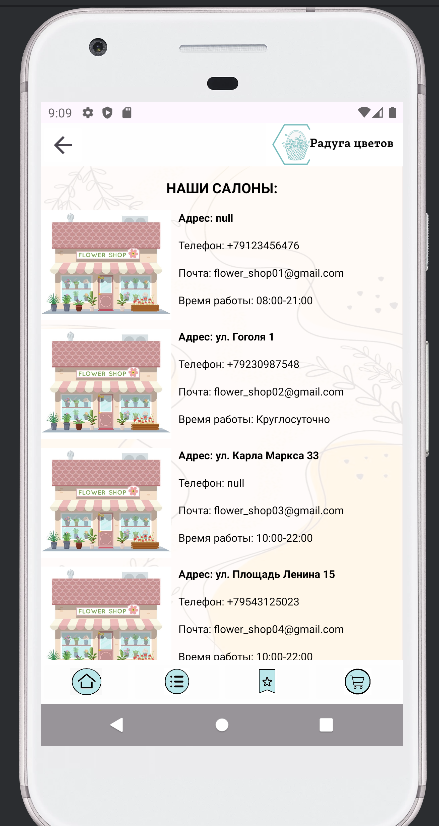


Рисунок 22 – Список магазинов

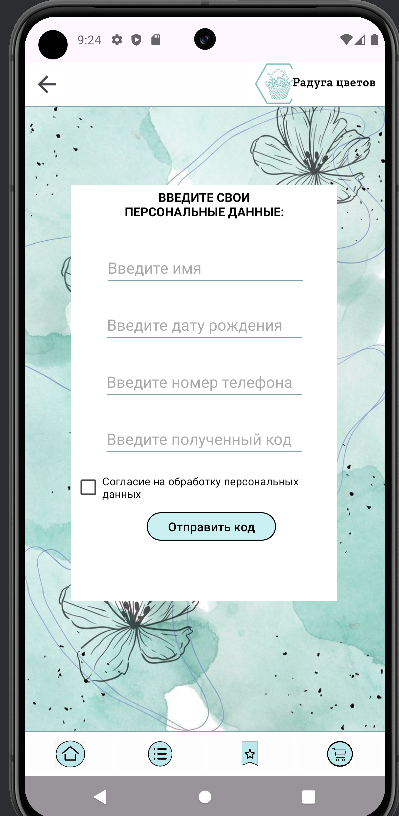
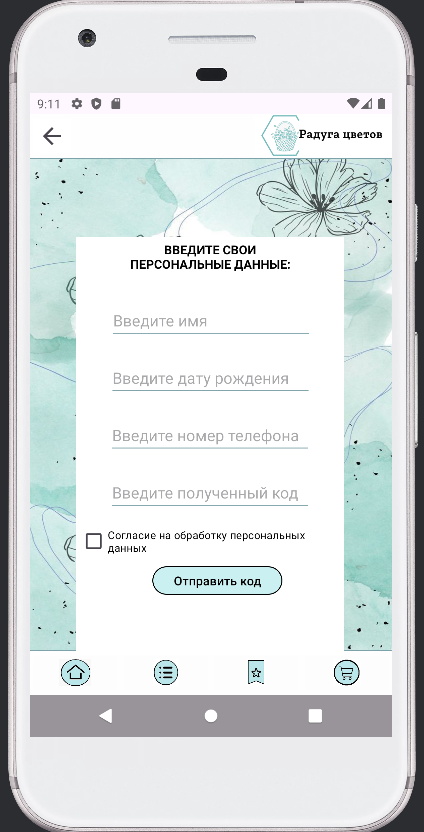


Рисунок 23 – Окно регистрации

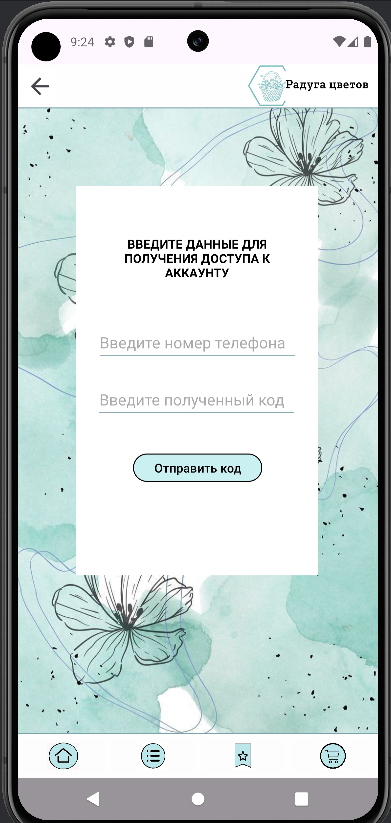
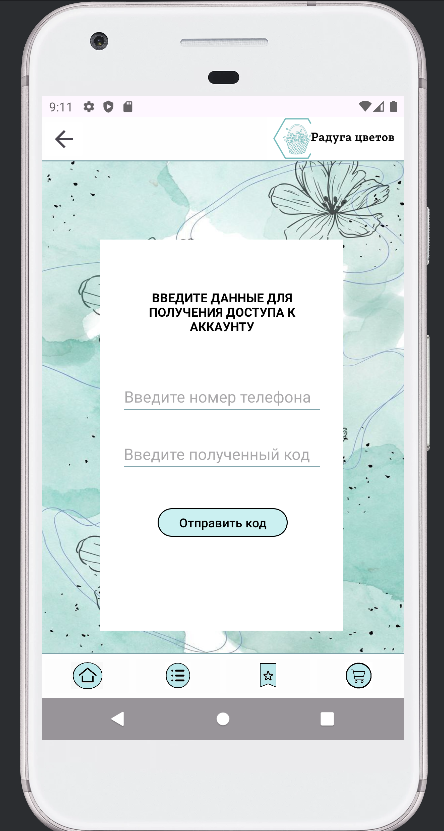


Рисунок 24 – Окно авторизации

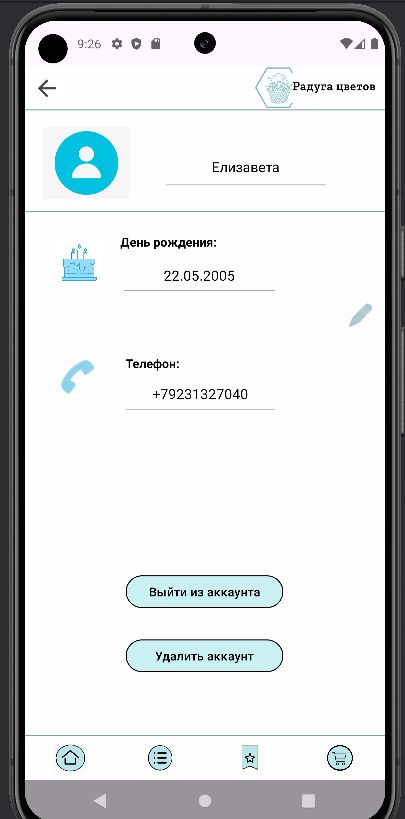
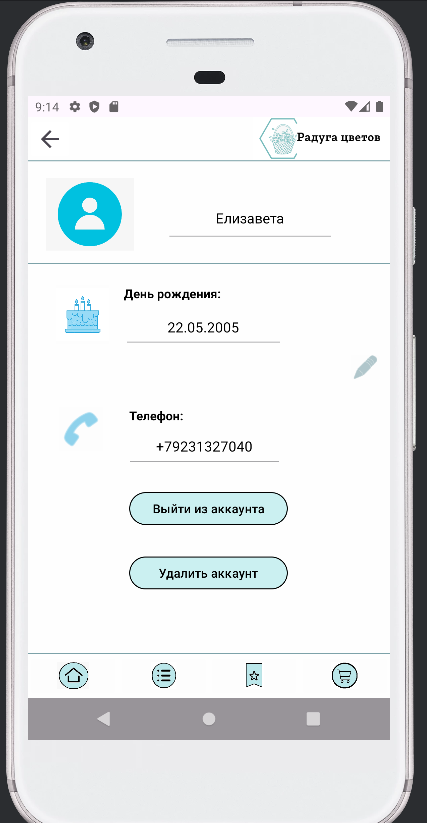


Рисунок 25 – Профиль клиента

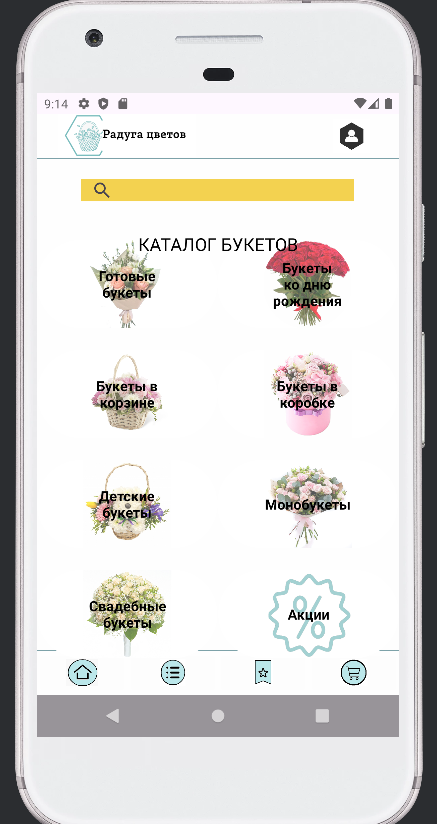


Рисунок 26 – Вкладка «Категории букетов»

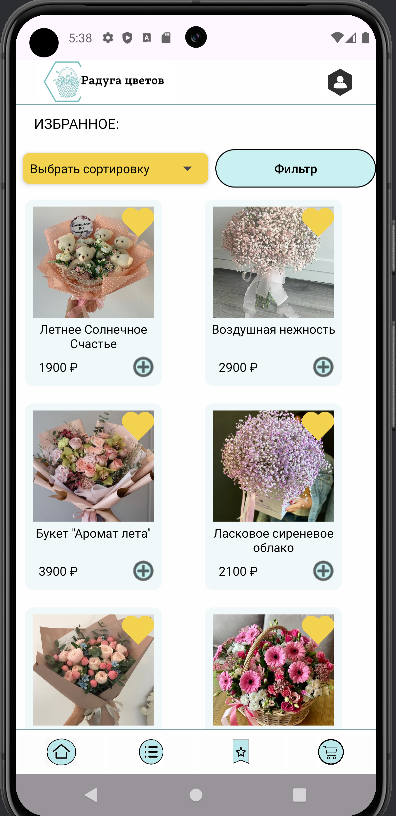
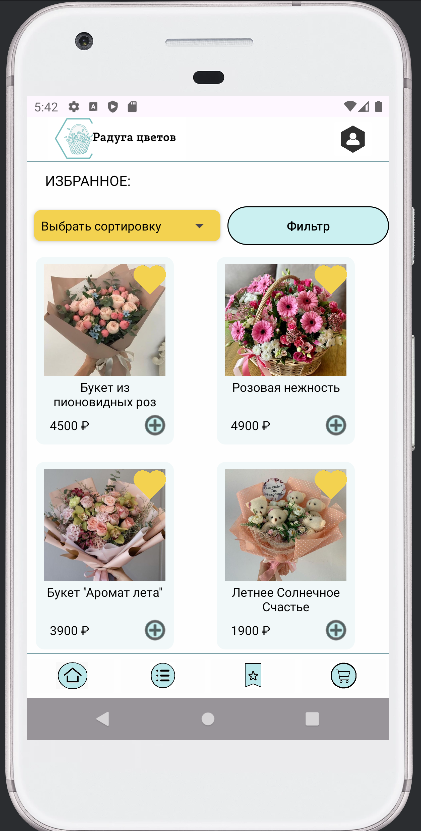


Рисунок 27 – Вкладка «Избранное»

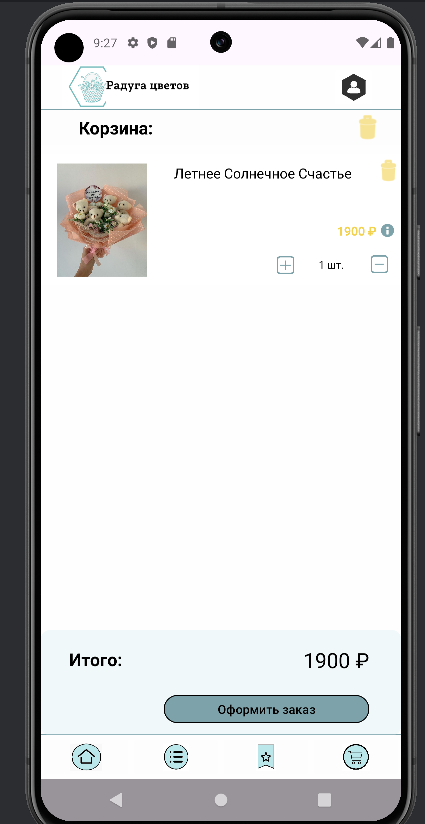
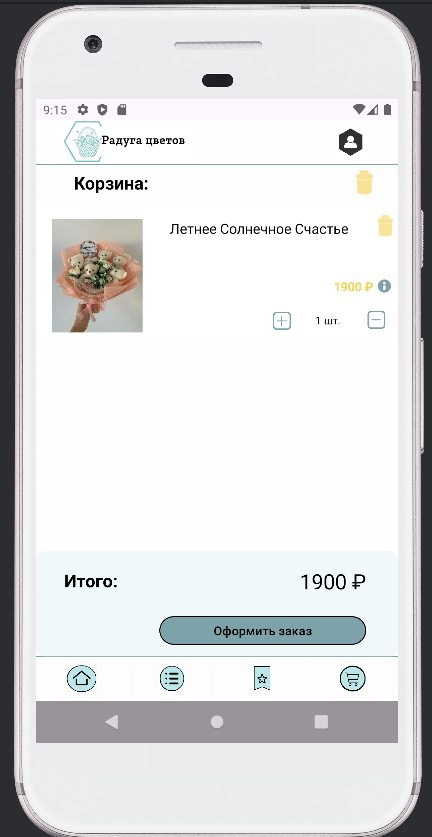


Рисунок 28 – Вкладка «Корзина»

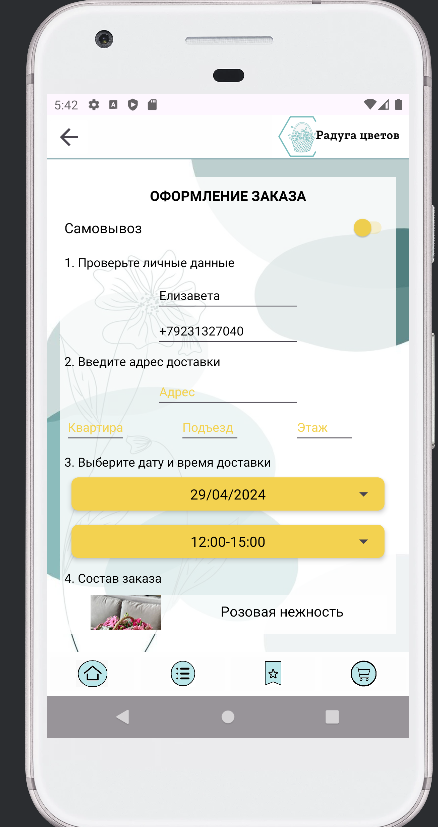


Рисунок 29 – Окно оформления заказа

## Протокол тестирования функционала приложения

Для тестирования функционала были подготовлены тест-кейсы, каждый из которых описывает набор шагов и ожидаемый результат при выполнении определенной функциональности. Эти тест-кейсы позволяют оптимально проверить функционал приложения с учетом различных сценариев использования и обнаружить возможные проблемы. Разработанные тест-кейсы представлены в таблицах с номерами от 1 до 5.

Таблица 1 – Test – case 1

|  |  |  |  |
| --- | --- | --- | --- |
| Название: | Дымовое тестирование | | |
| Функция: | Запуск приложения | | |
| Действие | | Ожидаемый результат | Результат теста:   * пройден * провален * заблокирован |
| Предусловие: | |  |  |
|  | |  |  |
| Шаги теста: | |  |  |
| Нажатие на ярлык приложения | | Корректное открытие приложения | пройден |
| Постусловие: | |  |  |
| Продолжение работы | | Приложение работает и функционирует | пройден |

Таблица 2 – Test – case 2

|  |  |  |  |
| --- | --- | --- | --- |
| Название: | Критическое тестирование | | |
| Функция: | Работа авторизации | | |
| Действие | | Ожидаемый результат | Результат теста:   * пройден * провален * заблокирован |
| 1 | | 2 | 3 |
| Предусловие: | |  |  |
| Запуск приложения | | Корректная работа приложения | пройден |
| Нажатие на кнопку «Профиль» | | Открытия страницы с выбором входа в профиль | пройден |
| Нажатие на кнопку «Зарегистрироваться» | | Открытия формы заполнения | пройден |
| Нажатие на текстовое поле «Номер телефона» | | Ввод номера телефона | пройден |
| Нажатие на кнопку «Отправить код» | | Открытие поля для ввода кода и отправления его на номер телефона | пройден |

Окончание таблицы 2

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Нажатие на текстовое поле «Пароль» | Ввод пароля | пройден |
| Нажатие на кнопку «Авторизоваться» | Открытие главного окна приложения | пройден |

Таблица 3 – Test – case 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название: | Критическое тестирование | | | |
| Функция: | Работа навигационного меню | | | |
| Действие | | Ожидаемый результат | Результат теста:   * пройден * провален * заблокирован | |
| Предусловие: | |  |  | |
| Запуск приложения | | Корректная работа приложения | пройден | |
| Пользователь авторизовался | | Предоставление доступа ко всем кнопкам меню | пройден | |
| Шаги теста: | |  |  | |
| Нажатие на кнопку меню «Главная» | | Перемещение на главную страницу | пройден | |
| Нажатие на кнопку меню «Каталог» | | Перемещение на каталог букетов | пройден | |
| Нажатие на кнопку меню «Избранное» | | Перемещение на страницу с избранными букетами | пройден | |
| Нажатие на кнопку меню «Корзина» | | Перемещение на страницу с корзинной | пройден | |
| Постусловие: | |  | |  |
| Продолжение работы | | Приложение работает и функционирует | | пройден |

Таблица 4 – Test – case 4

|  |  |  |  |
| --- | --- | --- | --- |
| Название: | Критическое тестирование | | |
| Функция: | Работа регистрации | | |
| Действие | | Ожидаемый результат | Результат теста:   * пройден * провален * заблокирован |
| 1 | | 2 | 3 |
| Предусловие: | |  |  |
| Запуск приложения | | Корректная работа приложения | пройден |
| Нажатие на кнопку «Профиль» | | Открытия страницы с выбором входа в профиль | пройден |
| Нажатие на кнопку «Зарегистрироваться» | | Открытия формы заполнения | пройден |
| Шаги теста: | |  |  |

Окончание таблицы 4

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Нажатие на текстовое поле «Имя» | Ввод имени | пройден |
| Нажатие на кнопку «Номер телефона» | Ввод даты рождения | пройден |
| Нажатие на текстовое поле «Номер телефона» | Ввод номера телефона | пройден |
| Нажатие на кнопку «Отправить код» | Открытие поля для ввода кода и отправления его на номер телефона | пройден |
| Нажатие на текстовое поле «Пароль» | Ввод пароля | пройден |
| Нажатие на кнопку «Авторизоваться» | Открытие главного окна приложения | пройден |

Таблица 5 – Test – case 5

|  |  |  |  |
| --- | --- | --- | --- |
| Название: | Критическое тестирование | | |
| Функция: | Работа регистрации | | |
| Действие | | Ожидаемый результат | Результат теста:   * пройден * провален * заблокирован |
| 1 | | 2 | 3 |
| Предусловие: | |  |  |
| Запуск приложения | | Корректная работа приложения | пройден |
| Нажатие на кнопку «Добавить в корзину» у выбранного букета | | Вывод сообщения об успешном добавлении в корзину | пройден |
| Нажатие на кнопку «Корзина» | | Открытия страницы с корзиной | пройден |
| Нажатие на кнопку «Оформить заказ» | | Открытия формы заполнения | пройден |
| Шаги теста: | |  |  |
| Нажатие на список доступных дат для доставки | | Вывод данных о доступных днях доставки | пройден |
| Нажатие на список доступного времени для доставки | | Вывод данных о доступном времени для доставки | пройден |
| Нажатие на текстовое поле «Адрес» | | Ввод адреса | пройден |
| Нажатие на кнопку «Этаж» | | Ввод этажа | пройден |
| Нажатие на текстовое поле «Подъезд» | | Ввод подъезда | пройден |
| Нажатие на текстовое поле «Квартира» | | Ввод номера квартиры | пройден |

Окончание таблицы 5

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Нажатие на текстовое поле «Комментарии» | Ввод комментария | пройден |
| Нажатие на кнопку «Оформить заказ» | Оформление заказа и уведомление о результате | пройден |

Заключение

В современном мире цветочные салоны играют значительную роль, не только в предоставлении красочных цветов и композиций, но и во внесении красоты и эстетического удовольствия в нашу повседневную жизнь. Однако, в условиях быстрого технологического развития, цветочным салонам необходимо активно адаптироваться к ожиданиям клиентов и современным технологиям для удобства и эффективности обслуживания.

В ходе разработки мобильного приложения для цветочного магазина учтены предпочтения и потребности пользователей. Функция оформления заказов, работа с вкладкой «Избранное» и «Корзина», а также регистрация и авторизация успешно авторизована.

Интерфейс приложения создан с фокусом на удобстве использования и минимальном числе шагов для достижения цели пользователей. Основные экраны разработаны с учетом стремления не перегружать информацией, при этом предоставляя полезные функции, включая просмотр трейлеров и возможность создания персонализированных списков просмотра и избранного контента.

Одной из ключевых функций является синхронизация всех данных между различными активностями приложения. Например, если определенный букет добавлен в список избранных, эта информация автоматически отображается на всех активностях, где присутствует список избранных букетов. Такая синхронизация обеспечивает консистентность данных и удобство для пользователей, позволяя им видеть актуальную информацию в любой части приложения.

Разработка мобильной версии цветочного салона позволила создать функциональное и удобное приложение, учитывающее потребности пользователей. Проделанная работа успешно достигла поставленной цели курсового проекта и является основой для дальнейшего развития и улучшения приложения.

Список используемых ресурсов

Нормативно – правовые акты:

1. ГОСТ Р 2.105-2019. ЕСКД. Общие требования к текстовым документам. – Москва: Стандартинформ, 2019. – 36 с
2. ГОСТ 19.503–79 ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению. – М.: Стандартинформ, 2010 – 6 с.

Электронные ресурсы:

1. Android Developers [Электронный ресурс]. – Документация Android Studio. – URL: https://developer.android.com/docs/ (дата обращения: 27.02.2024)
2. Developer documentation for Firebase [Электронный ресурс]. – Документация Firebase. – URL: https://firebase.google.com/docs?hl=ru (дата обращения: 05.03.2024)
3. Splash screens [Электронный ресурс]. – Документация Splash screens. – URL: https://developer.android.com/develop/ui/views/launch/splash-screen (дата обращения: 08.03.2024)
4. Firestore: глубокое знакомство с базой данных NoSQL Firebase [Электронный ресурс]. – Документация Firestore. – URL: https://appmaster.io/ru/blog/baza-dannykh-firestore-nosql (дата обращения: 15.03.2024)
5. Firebase для Android: хранилище файлов [Электронный ресурс]. – Документация Firebase Storage. – URL: https://code.tutsplus.com/ru/firebase-for-android-file-storage--cms-27376t (дата обращения: 18.03.2024)
6. Уроки по Android Studio [Электронный ресурс]. – Уроки Android Studio. – URL: https://startandroid.ru/ru/uroki/vse-uroki-spiskom.html (дата обращения: 28.02.2024)
7. Адаптеры и списки [Электронный ресурс]. – Глава 7. Адаптеры и списки. – URL: https://metanit.com/java/android/5.1.php (дата обращения: 15.04.2024)

Приложение А

(обязательное)

Министерство образования Новосибирской области ГБПОУ НСО «Новосибирский авиационный технический колледж имени Б.С. Галущака»

Разработка мобильного приложения для цветочного МАГАЗИНА

Техническое задание

НАТКиГ.201400.010.000ПЗ

Разработала:

студентка группы ПР-21.101

Ляпунова Е. Д.

2024

**Содержание**

Введение 41

1 Назначения разработки 42

2 Требования к мобильному приложению 43

2.1 Требования к функциональным характеристикам 43

2.2 Требования к надёжности 43

2.3 Условия эксплуатации 43

2.4 Требования к составу и параметрам технических средств 44

2.5 Требования к информационной и программной совместимости 44

2.6 Требования к защите информации 44

2.7 Требования к маркировке и упаковке 44

3 Требования к программной документации 45

4 Технико-экономические показатели 45

5 Стадии и этапы разработки 45

6 Порядок контроля и приёмки 45

Введение

Данное техническое задание распространяется на разработку мобильного приложения «Разработка мобильного приложения для цветочного магазина», используемого для управления продуктами, отображения каталога букетов, добавления букетов в избранное и корзину покупок, а также для обеспечения возможности оформления заказов.

Наименование приложения: «Радуга цветов».

Краткая характеристика области применения: мобильное приложение предоставляет возможность просматривать список букетов магазина, выбирать конкретную категорию букетов для просмотра списка, работа с избранным и оформлением заказа с выбором: самовывоз или доставка.

Основанием для проведения разработки является Протокол № Уч-29/4 от «22» Февраля 2024 г.

1 Назначение разработки

Основное назначение приложения заключается в:

* обеспечении удобным и эффективным интерфейсом приложения для пользователей;
* разработке работы функционала работы с избранным;
* разработке функционала оформления заказа.

Люди который могут работать с данными приложением – это пользователи приложения, которые могут просматривать список букетов, а при прохождении авторизации получат доступ к работе с избранным и оформлением заказов с выбором типа доставки.

2 Требования к мобильному приложению

**2.1 Требования к функциональным характеристикам**

Требование к составу выполняемых функций:

* авторизация и регистрация пользователей;
* изменение личной информации;
* удаление профиля;
* выход из аккаунта;
* ознакомление с категориями букетов;
* ознакомление со списками букетов;
* ознакомления с карточкой букета;
* просмотр списка магазинов с возможностью перехода в приложение при нажатии на контактную информацию – электронную почту, номер телефона;
* работа со списком избранных букетов;
* работа с корзиной;
* оформление заказа через доставку;
* оформление заказа через самовывоз.

**2.2 Требования к надёжности**

Обеспечение устойчивого функционирования должно выполняться несколькими действиями:

* стабильной работоспособностью без сбоев;
* обеспечением обнаружений и диагностики ошибок с выдачей соответствующих сообщений пользователю на уровне пользовательского интерфейса.

Приложение должно контролировать входную информацию.

**2.3 Условия эксплуатации**

Пользователь должен иметь практические навыки использования мобильного устройства под управлением операционной системы Android.

**2.4 Требования к составу и параметрам технических средств**

Для работы приложения необходимо мобильное устройство с установленной операционной системой Android не ниже версии 7.0.

**2.5 Требования к информационной и программной совместимости**

Проектирование взаимодействия с файловой системой должно быть выполнено в рамках разработки курсового проекта. При разработке взаимодействия с файловой системой должен быть использован язык программирования Java.

**2.6 Требования к защите информации**

Доступ к информации БД предоставляется только администратору базы данных.

**2.7 Требования к маркировке и упаковке**

Требования к маркировке и упаковке не предъявляются.

3 Требования к программной документации

Состав программной документации должен включать в себя:

* техническое задание;
* пояснительная записка.

**4 Технико-экономические показатели**

Экономические преимущества разработки и ориентировочная экономическая эффективность не рассчитывается.

**5 Стадии и этапы разработки**

Таблица 1 – Стадии разработки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Этапы разработки КП | | Сроки выполнения | Отчётность |
| 1 | 2 | | 3 | 4 |
| 1 | Определение цели и задач, объекта и предмета исследования | | 22.02.2024 | Пояснительная записка |
| 2 | Описание предметной области | | 28.02.2024 | Пояснительная записка |
| 3 | Выбор технологии, языка и среды программирования | | 05.03.2024 | Пояснительная записка |
| 4 | | Оформление технического задания | 16.03.2024 | Техническое задание |
| 5 | | Проектирование UI/UX дизайна | 30.03.2024 | Спецификации программного обеспечения |
| 6 | | Разработка базы данных | 01.04.2024 | Программный  продукт |
| 7 | | Разработка мобильного приложения | 02.04.2023 | Схема структурная системы и спецификации компонентов |
| 8 | | Отладка и тестирование приложения | 25.04.2024 | Тексты программных компонентов |
| 9 | | Оформление документации | 26.04.2024 | Программная  документация |
| 10 | | Защита | 29.04.2024 |  |

**6 Порядок контроля и приёмки**

Виды испытаний – защита курсового проекта.

Общее требования к приёмке:

* техническое задание;
* пояснительная записка;
* программный продукт.

Приложение Б

(справочное)

Макет приложения

Основные экраны приложения:

* главный экран, представлен на рисунке 1;
* каталог, представлен на рисунке 2;
* авторизация, представлен на рисунке 3;
* регистрация, представлен на рисунке 4;
* ввод кода для подтверждения регистрации или авторизации, представлен на рисунке 5;
* избранное, представлен на рисунке 6;
* корзина, представлен на рисунке 7;
* оформление заказа через доставку, представлен на рисунке 8;
* оформление заказ через самовывоз, представлен на рисунке 9.

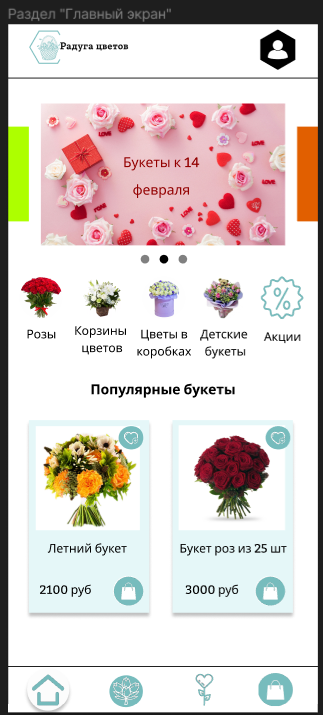


Рисунок 1 – Главный экран приложения

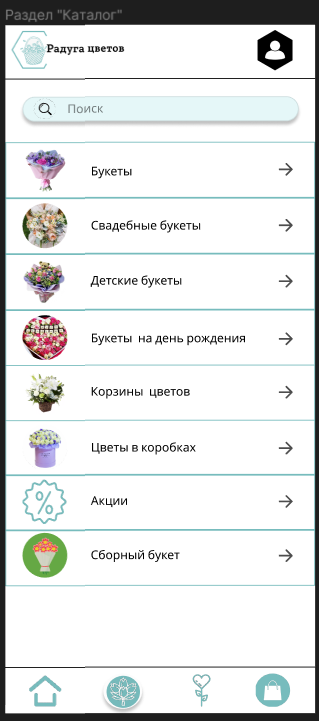


Рисунок 2 – Каталог приложения

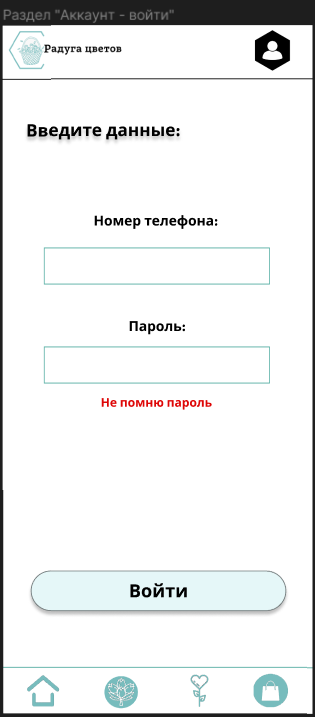


Рисунок 3 – Авторизация в приложении

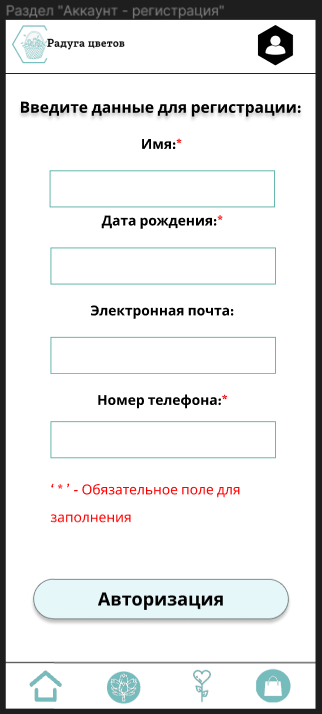


Рисунок 4 – Регистрация в приложении

****

Рисунок 5 – Экран ввода кода



Рисунок 6 – Вкладка «Избранное» в приложении

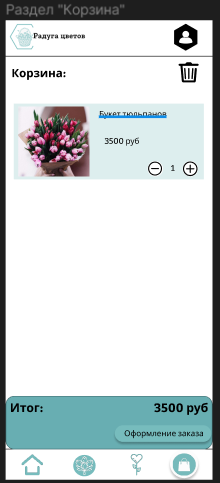


Рисунок 7 – Вкладка «Корзина» в приложении

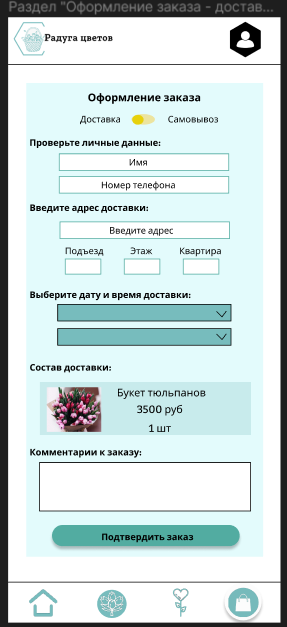


Рисунок 8 – Оформление доставки в приложении

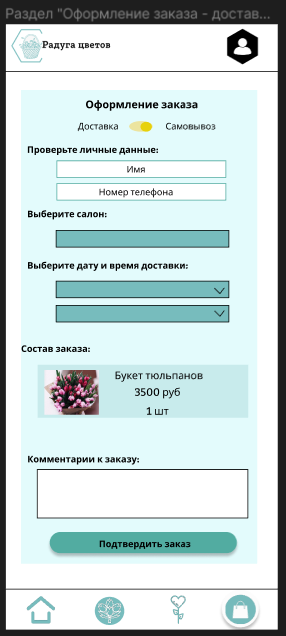


Рисунок 9 – Оформление самовывоза в приложении

Приложение В

(справочное)

Листинг 1 – Адаптер «BouquetsAdapter»

public class BouquetsAdapter extends RecyclerView.Adapter<BouquetsAdapter.BouquetsViewHolder> {  
 private static Context *context*;  
 private ArrayList<Bouquets> bouquetsList;  
 private ArrayList<Bouquets> favoriteBouquetsList = new ArrayList<>();  
 private ArrayList<Bouquets> buyBouquetsList = new ArrayList<>();  
  
 private FavoritesListener favoritesListener;  
 private static FirebaseFirestore *db*;  
 public BouquetsAdapter(Context context, ArrayList<Bouquets> bouquetsList, FavoritesListener favoritesListener) {  
 this.*context* = context;  
 this.bouquetsList = bouquetsList;  
 this.favoritesListener = favoritesListener;  
 this.*db* = FirebaseFirestore.*getInstance*();  
 }  
  
 @NonNull  
 @Override  
 public BouquetsViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(*context*).inflate(R.layout.*adapter\_bouquets*, parent, false);  
 return new BouquetsViewHolder(view);  
 }  
  
 public void setBouquetsListFavority(ArrayList<Bouquets> bouquets) {*//Назначаем полученный ищбранный список как основной лист* bouquetsList = bouquets;  
 }  
 public ArrayList<Bouquets> getFavoriteBouquetsListFavority() {  
 fetchFavoriteBouquetsByUser();  
 return favoriteBouquetsList;  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull BouquetsViewHolder holder, int position) {  
 Bouquets bouquet = bouquetsList.get(position);  
 holder.bind(bouquet);  
 View.OnClickListener onClickListener = v -> {  
 Intent intent = new Intent(*context*, Bouquets\_cards.class);  
 intent.putExtra("bouquet\_id", bouquet.getId());  
 *context*.startActivity(intent);  
 };  
 holder.itemView.setOnClickListener(onClickListener);  
 holder.imageView.setOnClickListener(onClickListener);  
 holder.productName.setOnClickListener(onClickListener);  
 holder.productPrice.setOnClickListener(onClickListener);  
  
 holder.btnBuy.setOnClickListener(v->{  
 if(FirebaseAuth.*getInstance*().getCurrentUser() != null){  
 if(bouquet!=null) {  
 addToOrderCart(bouquet);  
 } else {Toast.*makeText*(*context*, "Ошибка!", Toast.*LENGTH\_SHORT*).show();}  
  
 }else {Toast.*makeText*(*context*, "Войдите в аккаунт", Toast.*LENGTH\_SHORT*).show();}  
 });  
  
  
 holder.btnFavorite.setOnClickListener(v -> {  
 if (FirebaseAuth.*getInstance*().getCurrentUser() != null) {  
 if (bouquet != null) {  
 if (bouquet.getId() != null) {  
 *db*.collection("FavoriteList")  
 .whereEqualTo("userId", FirebaseAuth.*getInstance*().getCurrentUser().getUid())  
 .whereEqualTo("bouquetId", bouquet.getId())  
 .get()  
 .addOnSuccessListener(queryDocumentSnapshots -> {  
 if (!queryDocumentSnapshots.isEmpty()) {  
 removeBouquetFromFavorites(bouquet);  
 } else if (queryDocumentSnapshots.isEmpty()) {  
 addBouquetToFavorites(bouquet);  
 }  
 })  
 .addOnFailureListener(e -> {  
 Toast.*makeText*(*context*, "Ошибка!" + e, Toast.*LENGTH\_SHORT*).show();  
 });  
 } else if (bouquet.getId() == null) {  
 Toast.*makeText*(*context*, "Ошибка! Не найден ID букета!", Toast.*LENGTH\_SHORT*).show();  
 }  
 } else if (bouquet == null) {  
 Toast.*makeText*(*context*, "Ошибка! Букет не найден!", Toast.*LENGTH\_SHORT*).show();  
 }  
 } else if (FirebaseAuth.*getInstance*().getCurrentUser() == null) {  
 Toast.*makeText*(*context*, "Войдите в аккаунт", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 }  
 public int getItemCount() {  
 return bouquetsList.size();  
 }  
  
 public static class BouquetsViewHolder extends RecyclerView.ViewHolder {  
 ImageView imageView;  
 TextView productName;  
 TextView productPrice;  
 ImageButton btnFavorite;  
 ImageButton btnBuy;  
  
 public BouquetsViewHolder(@NonNull View itemView) {  
 super(itemView);  
 imageView = itemView.findViewById(R.id.*imageView*);  
 productName = itemView.findViewById(R.id.*ProductName*);  
 productPrice = itemView.findViewById(R.id.*ProductPrice*);  
 btnFavorite = itemView.findViewById(R.id.*btn\_favorite*);  
 btnBuy = itemView.findViewById(R.id.*btnbuy*);  
 }  
  
  
 public void bind(Bouquets bouquet) {  
 Picasso.*get*().load(bouquet.getImage()).into(imageView);  
 productName.setText(bouquet.getName());  
 productPrice.setText(String.*format*("%s ₽", bouquet.getCost()));  
  
 if (FirebaseAuth.*getInstance*().getCurrentUser() != null) {  
 *db*.collection("FavoriteList")  
 .whereEqualTo("userId", FirebaseAuth.*getInstance*().getCurrentUser().getUid())  
 .whereEqualTo("bouquetId", bouquet.getId())  
 .get()  
 .addOnSuccessListener(queryDocumentSnapshots -> {  
 if (!queryDocumentSnapshots.isEmpty()) {  
 bouquet.setFavorite(true);  
 } else {  
 bouquet.setFavorite(false);  
 }  
 if (bouquet.getFavorite()) {  
 btnFavorite.setImageResource(R.mipmap.*ikon\_favorite\_add*);  
 } else if (!(bouquet.getFavorite())) {  
 btnFavorite.setImageResource(R.mipmap.*ikon\_favorite*);  
 }  
 })  
 .addOnFailureListener(e -> {  
 Toast.*makeText*(*context*, "Ошибка!" + e, Toast.*LENGTH\_SHORT*).show();  
 });  
 }else{  
 btnFavorite.setImageResource(R.mipmap.*ikon\_favorite*);  
 }  
 }  
 }  
 public interface FavoritesListener {  
 void onFavoriteClick(Bouquets bouquet);  
 }  
  
 public void addToOrderCart(Bouquets bouquet) {  
 String userId = FirebaseAuth.*getInstance*().getCurrentUser().getUid();  
 String bouquetId = bouquet.getId();  
  
 if (bouquetId != null) {  
 OrderCart order = new OrderCart(userId, bouquetId);  
 *db*.collection("ListOrder")  
 .add(order)  
 .addOnSuccessListener(docRef -> {  
 notifyDataSetChanged(); *// Обновляем адаптер* Toast.*makeText*(*context*, "Букет добавлен в корзину!", Toast.*LENGTH\_SHORT*).show();  
 Log.*d*("ORDER", "Bouquet added to order successfully!");  
 })  
 .addOnFailureListener(e -> {  
 Log.*e*("ORDER", "Error adding bouquet to order", e);  
 Toast.*makeText*(*context*, "Букет не добавлен в корзину!", Toast.*LENGTH\_SHORT*).show();  
 });  
 } else {  
 Log.*d*("Debug", "Bouquet ID is null");  
 }  
 }  
  
 public void addBouquetToFavorites(Bouquets bouquet) {  
 String userId = FirebaseAuth.*getInstance*().getCurrentUser().getUid();  
 String bouquetId = bouquet.getId();  
  
 if (bouquetId != null) {  
 FavoriteItem favoriteItem = new FavoriteItem(userId, bouquetId);  
  
 *db*.collection("FavoriteList")  
 .add(favoriteItem)  
 .addOnSuccessListener(docRef -> {  
 favoriteBouquetsList.add(bouquet);  
 fetchFavoriteBouquetsByUser();  
 notifyDataSetChanged(); *// Обновляем адаптер* Toast.*makeText*(*context*, "Букет добавлен в избранное!", Toast.*LENGTH\_SHORT*).show();  
 Log.*d*("FAVORITES", "Bouquet added to favorites successfully!");  
 })  
 .addOnFailureListener(e -> {  
 Log.*e*("FAVORITES", "Error adding bouquet to favorites", e);  
 Toast.*makeText*(*context*, "Букет не добавлен в избранное!", Toast.*LENGTH\_SHORT*).show();  
 });  
 } else {  
 Log.*d*("Debug", "Bouquet ID is null");  
 }  
 }  
  
 public void removeBouquetFromFavorites(Bouquets bouquet) {  
 String userId = FirebaseAuth.*getInstance*().getCurrentUser().getUid();  
 String bouquetId = bouquet.getId();  
  
 *db*.collection("FavoriteList")  
 .whereEqualTo("userId", userId)  
 .whereEqualTo("bouquetId", bouquetId)  
 .get()  
 .addOnSuccessListener(queryDocumentSnapshots -> {  
 for (QueryDocumentSnapshot doc : queryDocumentSnapshots) {  
 *db*.collection("FavoriteList").document(doc.getId()).delete()  
 .addOnSuccessListener(aVoid -> {  
 favoriteBouquetsList.remove(bouquet);  
 fetchFavoriteBouquetsByUser();  
 notifyDataSetChanged();  
 Log.*d*("FAVORITES", "Bouquet removed from favorites successfully!");  
 Toast.*makeText*(*context*, "Букет удален из избранного!", Toast.*LENGTH\_SHORT*).show();  
 })  
 .addOnFailureListener(e -> {  
 Log.*e*("FAVORITES", "Error removing bouquet from favorites", e);  
 Toast.*makeText*(*context*, "Букет не удален из избранного!", Toast.*LENGTH\_SHORT*).show();  
 });  
 }  
 })  
 .addOnFailureListener(e -> Log.*e*("FAVORITES", "Error getting documents", e));  
 }  
  
  
 public void fetchFavoriteBouquetsByUser() {  
 favoriteBouquetsList.clear();  
  
 String userId = FirebaseAuth.*getInstance*().getCurrentUser().getUid();  
 *db*.collection("FavoriteList")  
 .whereEqualTo("userId", userId)  
 .get()  
 .addOnSuccessListener(queryDocumentSnapshots -> {  
 for (QueryDocumentSnapshot documentSnapshot : queryDocumentSnapshots) {  
 String bouquetId = documentSnapshot.getString("bouquetId");  
 if (bouquetId != null) {  
 *db*.collection("Bouquets")  
 .document(bouquetId)  
 .get()  
 .addOnSuccessListener(bouquetDocument -> {  
 if (bouquetDocument.exists()) {  
 Bouquets favoriteBouquet = bouquetDocument.toObject(Bouquets.class);  
 favoriteBouquet.setId(bouquetDocument.getId());  
 if (favoriteBouquet != null) {  
 favoriteBouquetsList.add(favoriteBouquet);  
 }  
 notifyDataSetChanged();  
 }  
 });  
 }  
 }  
 })  
 .addOnFailureListener(e -> Log.*e*("FAVORITES", "Error getting favorite bouquets for user", e));  
 }  
}