МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БАШКИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

**КАФЕДРА ПРОГРАММИРОВАНИЯ И ЭКОНОМИЧЕСКОЙ ИНФОРМАТИКИ**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ПО ПРОГРАММЕ БАКАЛАВРИАТА

**МАМАЛИМОВ ПАВЕЛ РАЛИФОВИЧ**

**РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ УПРАВЛЕНИЯ ЛИЧНЫМИ ФИНАНСАМИ**

|  |  |
| --- | --- |
|  | Выполнил:  Студент 4 курса очной формы обучения  Направление подготовки (специальность) 01.03.02 «Прикладная математика и информатика»  Направленность (профиль) Системное программирование и компьютерные технологии |
| Допущено к защите в ГЭК и проверено на объем заимствования:  Заведующий кафедрой  д.ф.-м.н., профессор  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Р.С. Юлмухаметов | Руководитель  к.ф.-м.н., доцент  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / М.Л. Бердникова |

УФА – 2021

Содержание

[Введение 3](#_Toc74938103)

[Глава 1. Программные решения, используемые для разработки мобильного приложения для управления финансами 5](#_Toc74938104)

[1.1. Мобильные приложения и их классификации 5](#_Toc74938105)

[1.2. Операционные системы для смартфонов. Android, IOS 8](#_Toc74938106)

[1.3. Анализ аналогичных Android приложений 8](#_Toc74938107)

[1.4. Анализ источников, необходимых для разработки Android приложения 11](#_Toc74938108)

[1.5. Среда разработки Android Studio 12](#_Toc74938109)

[1.6. Программирование в Android Studio на Java 14](#_Toc74938110)

[1.7. SQLite в Android Studio 15](#_Toc74938111)

[Вывод по первой главе 16](#_Toc74938112)

[Глава 2. Описание мобильного приложения 17](#_Toc74938113)

[2.1. Анализ требований мобильного приложения 17](#_Toc74938114)

[2.2. Архитектура приложения 18](#_Toc74938116)

[2.3. Список модулей и описание их функциональности 19](#_Toc74938117)

[Заключение 28](#_Toc74938118)

[Список литературы 29](#_Toc74938119)

[Приложение 31](#_Toc74938120)

# Введение

Управление личными финансами зачастую понимается людьми как экономия денежных средств. В свою очередь управление личными финансами – является распределением денег в зависимости от целей человека. Все большую актуальность в наше время имеет неправильное управление личными финансами. Многие люди сталкиваются с такой ситуацией, когда их доход уже долгое время стабильно повышается, а вот накоплений никак не получается сделать. Семейный бюджет также не изменяется, а у некоторых даже появляются долги. Люди, не задумываются или же вовсе не контролируют свою расходы и доходы. Причиной этого, отчасти, являются рекламы, акции в магазинах, общественное мнение. Под влиянием этих факторов человек зачастую совершает ненужные траты, не следит за своими расходами и тратит деньги не обдуманно.

К сожалению, многие люди еще со школьного возраста не понимают, что помимо знаний, которые они получают в школе, нужно расширять свой кругозор и обучаться навыкам ведения управлением личных финансов и денежных отношений. Как правило большинство просто видит неудовлетворенность определенной части общества своим финансовым положением, которые ничего не делают чтобы научится управлять своими финансами, приумножать их, правильно экономить и делать накопления, и продолжают поступать так же.

Управление личными финансами помогает выбрать план согласно вашим целям, учит с умом тратить деньги и организовывать семейный бюджет.

Управление личными финансами служит следующим целям:

- понять какие денежные средства используется неэффективно;

- выявить причины недостатка денег и найти способ их решения;

- проанализировать структуру расходов с целью повышения их эффективности.

**Объектом** выпускной квалификационной работы является управление личными финансами.

**Предметом** выпускной квалификационной работы является мобильное приложение для управления личными финансами

**Целью** выпускной квалификационной работы является создание мобильного приложения для управления личными финансами, которое будет простым в использовании и обеспечивает хранение в электронном виде данных о доходах и расходах пользователя.

Декомпозиция цели приводит к следующему списку задач исследования:

1. Проанализировать аналогичные приложения для управления личными финансами.
2. Рассмотреть среды разработки для мобильных приложений
3. Разработать мобильное приложение под управлением операционной системы Android.

# Глава 1. Программные решения, используемые для разработки мобильного приложения для управления финансами

## Мобильные приложения и их классификации

Мобильное приложение – является программным обеспечением, разработанным для работы на смартфонах, планшетах и других мобильных устройствах. Мобильное приложение специально разрабатывается под конкретные мобильные платформы такие как: Android, Windows Phone, iOS и др. Самыми крупными площадками являются AppStore и Google Play.

Виды мобильных приложений:

* + 1. Мобильные веб-приложения и сайты – клиент-серверные приложения, в котором клиент взаимодействует с веб-сервисом при помощи браузера

1. Нативные приложения – приложения, которые разработаны для использования на определенной платформе или устройстве
2. Гибридные приложения – является сочетанием веб и нативных приложения. Такие приложения можно загрузить только из маркетов вроде Google Play и App Store

На рисунке 1.1.1 изображена таблица сравнения видов мобильных приложений.



Рисунок 1.1.1 Сравнение видов мобильных приложений

Мобильные приложения можно разделить на несколько основных видов.

**Игровые приложения** (рисунок 1.1.2)– группа, в которую игры всех жанров. Мобильные игры занимают значительную часть на рынке приложений.



Рисунок 1.1.2. Пример игрового приложения

**Промо программы** (Рисунок 1.1.3) служат для продвижения бренда, для которого создано приложение. Главная цель разработчиков таких приложений обхватить максимальное количество клиентов, которые, в свою очередь, пользуются разными моделями мобильных телефонов. Через такие сервисы пользователи могут заказывать товары и услуги, оставлять отзывы, узнавать о скидках и акциях, получать бонусы.

****

Рисунок 1.1.3 Пример промо программы

**Контент-приложения** (Рисунок 1.1.4) – программы для быстрого доступа к определенному контенту. Прослушивать музыку, просматривать видеоролики, читать текстовых файлов, все это могут делать обладатели мобильных телефонов доступно и удобно благодаря контент-приложениям.

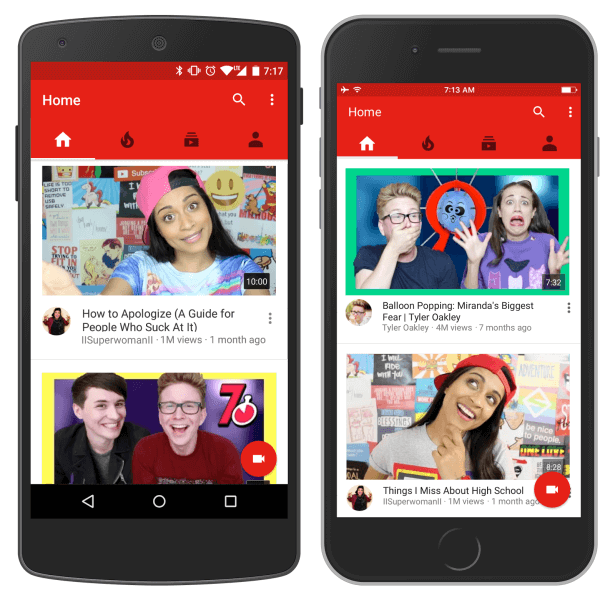


Рисунок 1.1.4 Пример контент-приложения

**Социальные сети** - приложения, которые позволяют общаться в социальных сетях через смартфон или планшет. Каждая популярная социальная сеть имеет свое мобильное приложение.

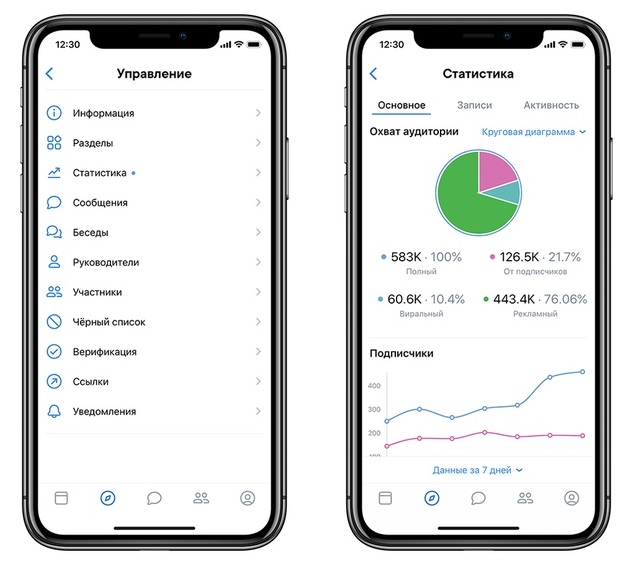


Рисунок 1.1.5 пример приложения социальной сети Вконтакте.

## Операционные системы для смартфонов. Android, IOS

На данный момент имеется большой выбор языков программирования и сред разработки, для разработки мобильных приложений. Такой обширный выбор потому, что для разных операционных систем нужно применять разные языки программирования. В процессе анализа операционных систем я ознакомился с такими технологиями, как IOS SDK, Android SDK.

**IOS SDK**

Наиболее подходящими языками разработки под IOS являются Objective-C и Swift. Так же можно программировать и на Python, C, C++, C# и Java для этого нужно просто компилировать приложение под IOS, но такие приложения работают гораздо хуже нативных.

**Android SDK**

Для разработки под операционную Android можно использовать такие среды разработки как Eclipse и Android Studio. IDE Eclipse на сегодняшний момент является устаревшей и в 2016 году в Google отказались от поддержки ее плагинов для Android-разработки. В свою очередь, IDE Android Studio поддерживается Google в качестве официальной среды разработки для Android-приложений. Она основана на IntelliJ IDEA, предлагает все нужные функции и заточена под проекты на Android.

Мною выбрана среда разработки Android Studio, так как она является официальной средой разработки Google и так же является более современной в отличие от Eclipse.

## Анализ аналогичных Android приложений

В магазине приложений Google Play Store есть много похожих приложений. Рассмотрим самые популярные и часто скачиваемые. Приложение **Monefy** одно из популярных приложений в данной категории.

Monefy (рисунок 1.5.1) – является простым приложением, которое позволяет пропустить этап настройки приложения и после установки сразу начать учитывать расходы и доходы.



Рисунок 1.5.1 интерфейс приложения Monefy

Приложение Monefy имеет простой и удобный для понимания интерфейс, на главном экране есть две кнопки, — плюс и минус — нужно нажать на одну из них, ввести сумму и выбрать нужную категорию. Так же у этого приложения есть платная версия (Monefy Pro), которая активирует такой функционал, как добавление собственных категории и учет денег в разных валютах.

Так же, было рассмотрено приложение **Spendee** (рисунок 1.5.2). Spendee – является бесплатным приложением для контроля бюджета. Оно помогает отслеживать и оптимизировать расходы и доходы.

Приложение Spendee имеет следующие важные функции:

1. Бюджеты - чтобы не сбиться с пути достижения финансовых целей
2. Управление разными валютами – чтобы вам было легче спланировать бюджет на отпуск
3. Маркеры - для более точного анализа транзакций и их маркировки
4. Веб-версия - для возможности анализировать ваши финансы на большем экране
5. Безопасная синхронизация данных – чтобы ваши данные оставались конфиденциальными
6. Бюджеты - чтобы не сбиться с пути достижения финансовых целей
7. Управление разными валютами – чтобы вам было легче спланировать бюджет на отпуск
8. Маркеры - для более точного анализа транзакций и их маркировки
9. Веб-версия - для возможности анализировать ваши финансы на большем экране
10. Безопасная синхронизация данных – чтобы ваши данные оставались конфиденциальными



Рисунок 1.5.2 приложение Spendee

Главной плюсом этих приложений это наличие русского интерфейса, который, кстати, имеет дружелюбный вид; возможность настройки категорий расходов "под себя" – есть выбор как среди уже готовых наименований с лаконичными пиктограммами, так и возможно создать свою категорию. Каждое приложение по-своему уникальны своим дизайном и функционалом, что позволяет внедрить новые идеи для разработки мобильного приложения.

## Анализ источников, необходимых для разработки Android приложения

Р. Майер, Программирование приложений для планшетных компьютеров и смартфонов. Эта книга, предназначенная для людей, которые уже имели опыт в программировании, но никогда не программировали для системы Android. Для работы с этой книгой читатель должен иметь опыт в программировании на любом языке, для того чтобы он мог понять основы алгоритмизации. В книге подробно описана технология разработки приложений.

Еще одним из источников является книга Д. Гриффитса "Head First. Программирование для Android. 2-е издание.". Она имеет простой и понятный язык изложения. Эта книга также предназначена для новичков в разработке ОС Android, но предполагает, что читатель уже немного знаком с основами языка Java.

Ян Ф. Дарвин "Android.Сборник рецептов. Задачи и решения для разработчиков приложений" является сборником из более чем двухсот рецептов по созданию приложений и ориентирован на опытных читателей, которые уже знакомы с языком Java, основы работы с платформой Android. В книге даны инструкции в виде пошагового решения задач с пользовательскими интерфейсами, механизмами определения местоположения, веб-службами и мультисенсорными жестами.

Интернет предоставляет много информации по теме разработки приложений. Преимущество этой информации заключается в том, что она часто является свежей и актуальной. Были рассмотрены некоторые из наиболее популярных ресурсов. Startandroid.ru является сайтом самой операционной системы Android. Есть много практических советов как для начинающих, так и для опытных пользователей. fandroid.info сайт, посвященный программированию на платформе Android, на нем можно найти видео уроки, в которых учат программировать языках программирования под Android наглядно разбирая примеры в видео роликах.

## Среда разработки Android Studio

Android Studio (рисунок 1.3.1) — является интегрированной средой разработки, с помощью которой пользователю доступны инструменты для создания приложений на платформе Android OS. Android Studio основана на базе IntelliJ IDEA от компании JetBrains. Android Studio доступна для Windows, macOS и GNU/Linux.

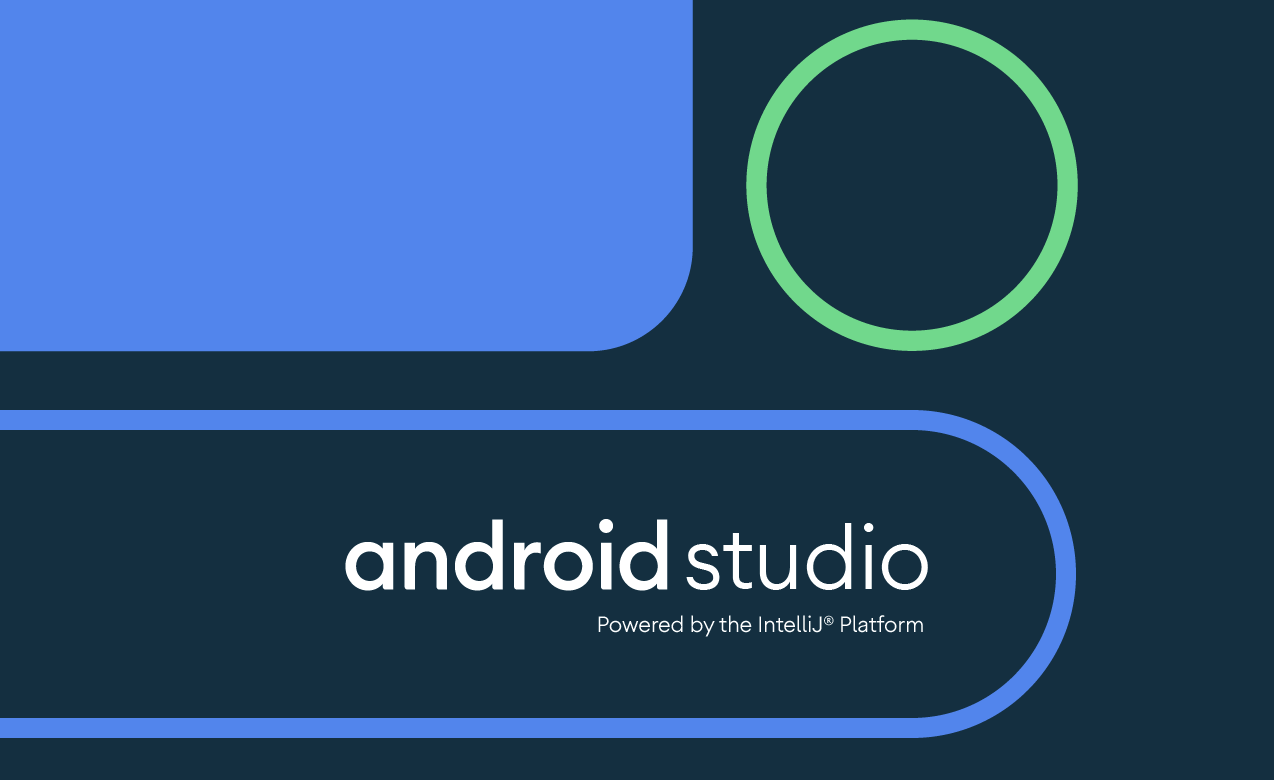


Рисунок 1.3.1 Android studio

Среда разработки Android Studio является бесплатной, то есть ее можно спокойно загрузить и сразу начать работать. Среда IDE не адаптирована под русский язык, поэтому интерфейс на английском. В Android Studio присутствуют макеты для создания пользовательского интерфейса, с которых можно начать разработку приложения. Также содержатся инструменты разработки решений для смартфонов и планшетов, а также поддерживается разработка приложений для Android Wear и Android TV.

В среде разработки Android Studio можно работать как одному разработчику, так и группе разработчиков с GIT или другими подобными системами управления версиями. Решения для Android разрабатываются в Android Studio с использованием Java, Kotlin или C++. В процесс разработки заложен концепт непрерывной интеграции, который позволяет сразу обнаружить имеющиеся проблемы.

Определять состояние файла с прикладными программами позволяют средства оценки производительности. С помощью визуализации графики можно проверить соответствие приложения ориентиру Google. А также с помощью инструмента для визуализации памяти разработчик может определять, когда его приложение использует много оперативной памяти и когда произойдет "сборка мусора". Инструменты для анализа батареи дают нам узнать, какая нагрузка приходится устройство.

Локализация приложений существенно проще с функцией SDK, которая также входит в перечень достоинств Android Studio.

Каждый проект в Android Studio содержит один или несколько модулей с файлами исходного кода и файлами ресурсов. Типы модулей включают:

1. Модули приложений Android
2. Модули библиотеки
3. Модули Google App Engine

По умолчанию Android Studio отображает файлы вашего проекта в представлении проекта Android, как показано на рисунке 1.3.2. Это представление организовано по модулям для обеспечения быстрого доступа к ключевым исходным файлам вашего проекта.

Все файлы сборки видны на верхнем уровне в Gradle Scripts, и каждый модуль приложения содержит следующие папки:

* + 1. manifests: содержит AndroidManifest.xml файл.
    2. java: содержит файлы с исходным кодом на языке Java, включая тестовый код JUnit.
    3. res: содержит все ресурсы, не связанные с кодом, такие как макеты XML, строки пользовательского интерфейса и растровые изображения.

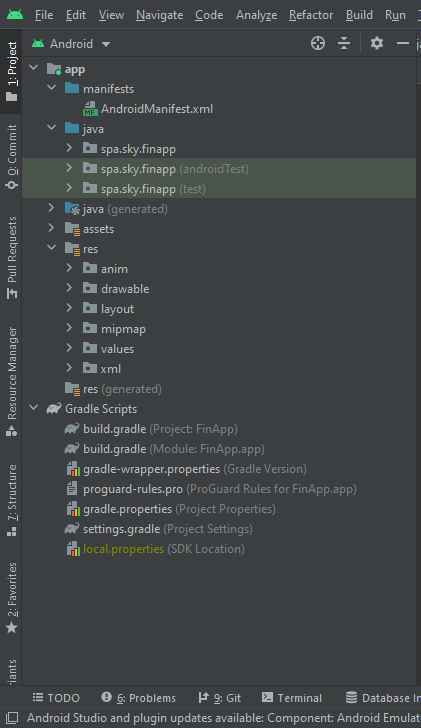


Рисунок 1.3.2 Файлы проекта в Android Studio

## Программирование в Android Studio на Java

Почему именно Java? Java - язык, который широко используется для разработки мобильных приложений. Существуют и другие языки, которые можно использовать для разработки приложений Android, например C и C ++, с использованием Native Development Kit (NDK), но для разработки приложений рекомендуется использовать Java. В доступной среде IDE, например Android Studio, Java является языком по умолчанию для кодирования, тогда как C и C ++ находятся в опциях. Он также поддерживает собственный код, поэтому C и C ++ применимы к коду для мобильного приложения. SDK (Software Development Kit) используется для разработки под Android, который имеет кучу библиотек вместо JVM.

Java и XML (eXtensible Markup Language) используются при разработке мобильных приложений. Java используется в разработке под Android для написания внутренней логики или бизнес-логики. Это язык высокого уровня. XML используется для разработки пользовательского интерфейса. Java объектно-ориентированный язык, но не чисто объектно-ориентированный из-за наличия встроенных типов данных в нем. Он называется объектно-ориентированным, потому что поддерживает все функции объектно-ориентированного языка.

Java играет важную роль в разработке приложений для Android, поскольку бизнес-логика написана на Java. Знание Java заранее является плюсом для разработки приложений для Android. Обладая глубокими знаниями Java, вы можете добавлять в приложение новые функции.

Java - объектно-ориентированный язык, что является серьезной причиной использования этого языка для разработки приложений. Он безопасен благодаря использованию классов и объектов и не поддерживает указатель и любое другое ключевое слово для управления памятью вручную, что снижает вероятность нежелательного вмешательства в приложение. Для управления памятью он использует сборщик мусора, который удаляет объекты, не используемые в течение длительного времени.

## SQLite в Android Studio

SQLite доступен на любом Android-устройстве и его не нужно устанавливать отдельно.

SQLite зарекомендовала себя в качестве чрезвычайно надёжной системы баз данных, которая используется во многих бытовых электронных устройствах и программах, включая некоторые MP3-проигрыватели, iPhone, iPod Touch, Mozilla Firefox и др.

С помощью SQLite можно создавать для своего приложения независимые реляционные базы данных. Android хранит базы данных в каталоге **/data/data/имя\_вашего\_пакета/databases** на эмуляторе, на устройстве путь может отличаться. Все базы данных по умолчанию являются закрытыми, доступ к ним могут получить только те приложения, которые их создали.

Библиотека Android содержит абстрактный класс SQLiteOpenHelper, с его помощью можно создавать, открывать и обновлять базы данных. Он требует обязательного переопределения методов, которые используются при инициализации, – методов создания, открытия и обновления базы данных. Метод onCreate будет вызван системой самостоятельно при первоначальной инициализации базы данных. В нем необходимо задать команды для создания таблиц внутри базы данных. Так как база данных точнее ее структура может меняться, то нужно реализовать метод onUpgrade, который стирает созданное ранее.

Для управления базой данных SQLite существует класс SQLiteDatabase. В классе SQLiteDatabase определены методы для чтения (query()), добавления( insert()), удаления(delete()) и изменения(update()) данных. Еще есть метод execSQL(), который  позволяет выполнять любой допустимый код на языке SQL, если вы хотите провести эти (или любые другие) операции вручную.

В приложении использовалась база данных SQLite, потому что проста в освоении и адаптирования для использования в среде разработки Android Studio.

## Вывод по первой главе

Были выявлены преимущества мобильного приложения после анализа предметной области. Было принято решение использовать среду разработки для Android под названием Android Studio и язык программирования Java после анализа похожих проектов и популярных сред разработки.

Так же проанализировав источники и приложения был сделан вывод, что многие приложения не имеют четкой классификации и структуры.

# Глава 2. Описание мобильного приложения

## Анализ требований мобильного приложения

Для того чтобы нам четко осознавать в каком направлении двигаться в дальнейшем, нам необходимо составить список требований, которым должно соответствовать наше приложение.

По данным на конец весны 2020 статистика (рисунок 2.1.1) в Android Studio показывает «кумулятивное распределение». Эти данные пригодятся разработчикам, которые хотят решить, какую версию Android выбрать как минимальную для установки своего приложения.

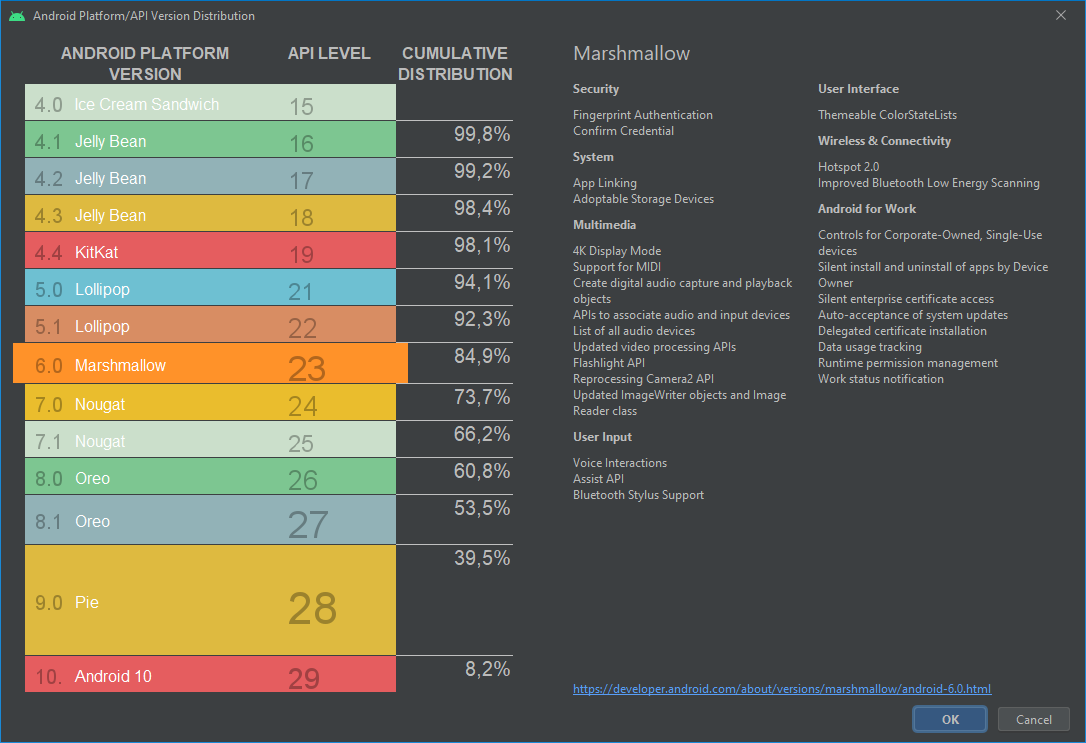


Рисунок 2.1.1 Статистика разных версий Android на конец весны 2020 года

Основываясь на статистике мною, была выбрана версия Android 4.1 Jelly Bean с API level 16 так как данная версия используется у 99,8% пользователей.

Технические требования:

1. Работать без серьезных ошибок
2. Работать с версии Android 4.1(API level 16)
3. Работать без визуальных задержек

Функциональные требования:

Пользователь заходит в приложение с обязательной авторизацией, если же пользователь не зарегистрирован, то он может пройти регистрацию в разделе Регистрация.

Для перехода в другие разделы должны быть кнопки, соответствующие дизайну внизу экрана.

В разделе транзакций и транзакций на интервале должна быть возможность прокрутки транзакций.

В разделе Добавить Расход должны быть реализованы иконки, нажимая на которые выбирается категория расхода.

Чтобы более глубже понять принципы разработки, нужно понять для какой программной платформы мы будем разрабатывать приложение для этого нам нужно познакомиться ближе с операционной системой Android.



## Архитектура приложения

Приложение для Android состоит из набора activity, каждой из которых соответствует экран приложения. Activity (в дальнейшем будем писать активность) – основной компонент в Android приложении. Активность – это экран то, что видит и с чем взаимодействует пользователь. Каждая активность представлена в проекте классом, написанном на языке java, который имеет расширение .java. Каждой активности соответствует xml файл, написанный на языке xml. Файл xml является файлом разметки в нем прописано расположение визуализированных объектов. Когда запускается активность система Android автоматически распознает разрешение экрана мобильного экрана и выводит контент в соответствии с разметкой. Так же в каждом приложении есть файл AndroidManifest.xml, который предоставляет подробную информацию о приложении. В AndroidManifest.xml прописаны минимальные требования к системе, а также активность, вызываемая при запуске приложения.

Приложение работает с базой данных SQLite (рисунок 2.1.1), которая прописана в самом приложении в виде класса DBHelper.java. В этом классе прописана база данных для приложения, а также функции с запросами к базе данных, которые в дальнейшем используются в приложении.

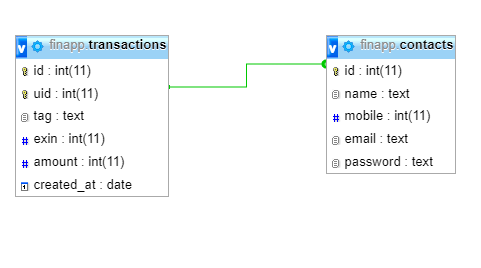


Рисунок 2.1.1 Схема Базы Данных приложения

## Список модулей и описание их функциональности

Функционально приложение состоит из приведенных ниже активностей. Activity является схемой представления Android приложения.

Каждый экран пользовательского интерфейса представлен классом activity и разметкой в виде XML-файлов. Android приложение способно состоять из одной или нескольких активностей и способно переключаться между ними во время выполнения приложения.

В процессе выполнения приложения первая активность, с которой столкнется пользователь, окажется Login\_Fragment, которая является activity авторизации (рисунок 2.2.1). Активность авторизации имеет два EditText (полей ввода) один из для ввода почты, а второй для ввода пароля, который при вводе текста в поле пароля скрывает его. Ещё имеется элемент CheckBox, который при первом нажатии на него показывает текст, введенный в поле пароля, при повторном нажатии текст в поле пароля опять скрывается. При нажатии на Button (кнопку) авторизоваться запускает функцию checkValidation(), которая сначала проверяет введены ли данные в поле почта и поле пароль, если оба или одно из полей является пустыми в приложении на экране появится уведомление, что оба поля должны быть заполнены, если же данные в поле почта введены не корректно, то на экране появится уведомление, что ваш Email недействителен. После проверки заполнения полей идет запрос в базу данных по введенным нами логину и паролю. Запускается запрос к классу DBHelper от введенным нами email - databaseHelper.searchPass(email). Функция searchPass() сравнивает введенный нами email с email в базе данных и возвращает нам List формата String (список). Если список окажется пустым, то на экране появится уведомление, чтобы мы зарегистрировались как пользователь для входа в систему. Если список не пустой, то начинается проверка введенных данных, если данные введены верно, то мы переходим на другую activity Dashboard, если какие-то из данных введены неверно, то на экране появится уведомление, что имя пользователя или пароль введены неверно.

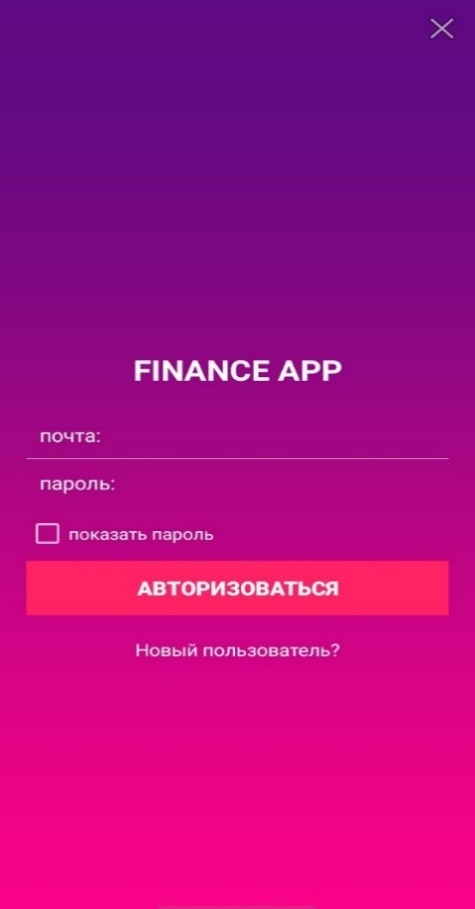


Рисунок 2.2.1 Activity авторизации

При нажатии на TextView "Новый пользователь?" нас переносит на активность (рисунок 2.2.2), которая называется SignUp\_Fragment. Эта активность регистрации нового пользователя здесь находится пять EditText (полей ввода), которые имеют свои значения для заполнения, один CheckBox для принятия условий пользования приложение и кнопка для отправки данных с EditText, чтобы создать нового пользователя в базе данных. После заполнения полей EditText и нажатия на кнопку "зарегистрироваться" запускается CheckValidation () в которой проверяется правильность введенных данных и при необходимости выводит на экран предупреждение, если что-то заполнено неверно или же вообще не заполнено. Если все данные введены верно, то внутри функции CheckValidation () запускается функция datebasehelper.insertContact (), которая создает нового пользователя в базе данных с учетом данных введенных в поля ввода. Когда новый пользователь создан пользователя переносит обратно на Activity авторизации.

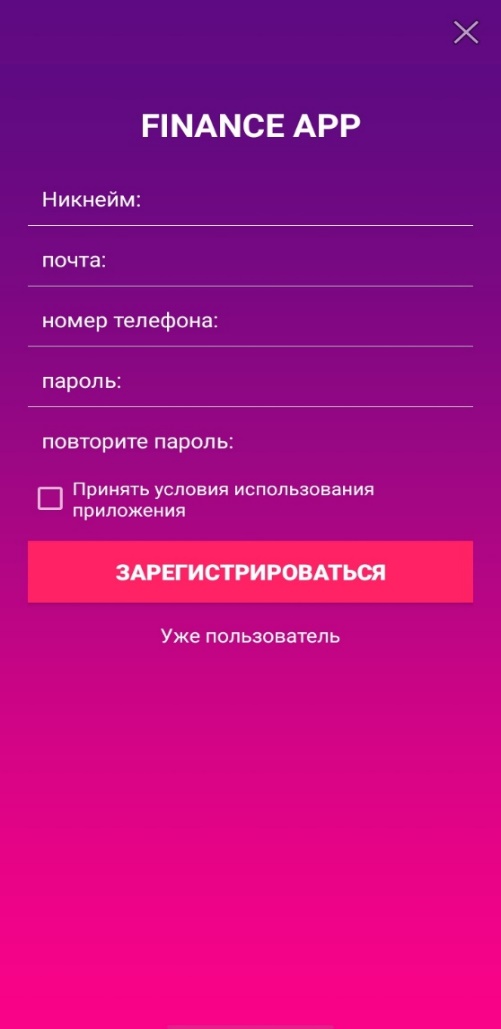


Рисунок 2.2.2 Activity регистрации

После того как пользователь зарегистрировался и авторизовался он попадает на Dashboard Activity (Рисунок 2.2.3). Эта активность является главным меню приложения. Пользователь может ознакомится с круговой диаграммой, которая строится на основе введенных им расходов и доходов с начала месяц, также видны последнее введенные две транзакции. Так же пользователь может увидеть общую сумму денег, которая у него осталась по подсчетам введенных им доходов и расходов с начала месяца, и сами расходы и доходы за этот период. При нажатии на текстовый элемент "показать все" пользователя переносит на activity всех транзакций, которые были совершены с начала месяца. При нажатии на кнопку "ДОХОДЫ" пользователя переносит на IncomeActivity, в которой пользователь может добавить новый доход. При нажатии на кнопку "РАСХОДЫ" пользователя переносит на ExpensesActivity, в которой пользователь может добавить новый доход. Так же при нажатии на одну из двух последних транзакций выйдет информации о том, когда была совершена транзакция.



Рисунок 2.2.3 Главное меню приложения

В IncomeActivity (рисунок 2.2.4) пользователю нужно ввести дату транзакции, сумму транзакции и нажать на кнопку принять. При нажатии на поле ввода даты пользователю на экран выводится календарь при помощи DatePicker в котором пользователю нужно выбрать дату транзакции и подтвердить ее, после выбора даты автоматически заполнится поле ввода даты. DatePicker это компонент интерфейса, позволяющий пользователю выбрать дату (date) и гарантирует, чтобы введенные пользователем данные были действительными. Далее пользователю нужно ввести сумму транзакции. После выбора даты и ввода суммы транзакции пользователю нужно подтвердить добавление транзакции нажатием кнопки "ПРИНЯТЬ". При нажатии кнопки "ПРИНЯТЬ" запускается функция addTake (), которая добавляет в базу данных новую транзакцию с тегом доход и после добавления переносит пользователя на activity всех транзакций TransactionsActivity. Круговая диаграмма в IncomeActivity и круговые диаграммы в других activity реализованны с использованием библиотеки MPAndroidChart. Библиотека MPAndroidChart – библиотека диаграмм, которую можно перетаскивать и масштабировать, включая графики, гистограммы и круговые диаграммы.

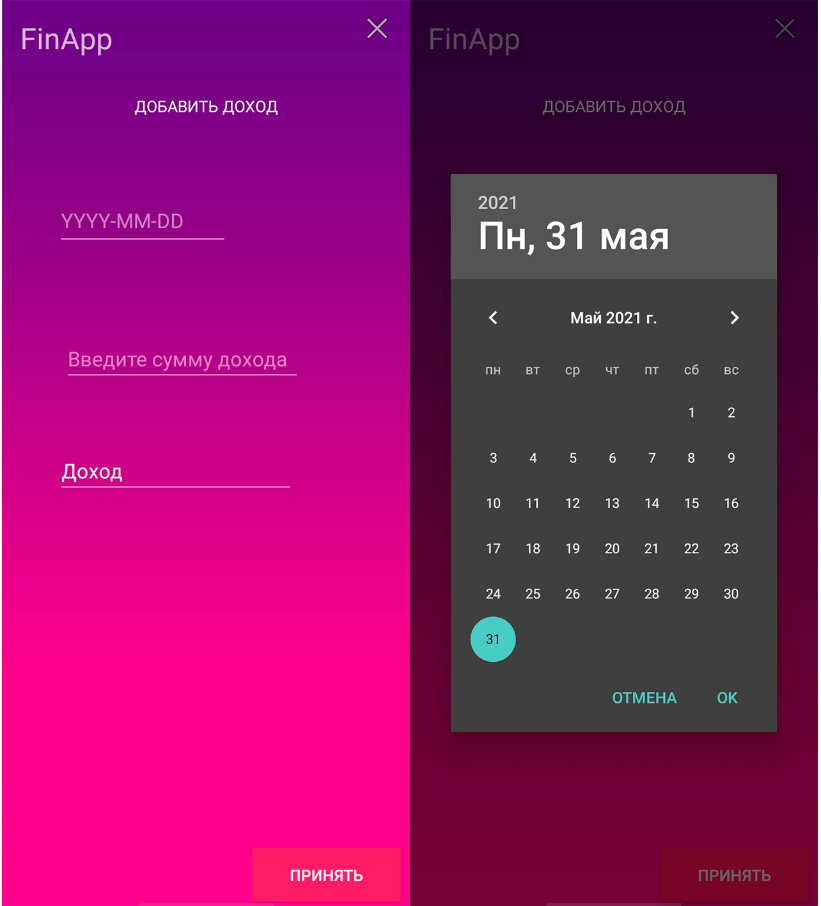


Рисунок 2.2.4 интерфейс IncomeActivity

В ExpensesActivity (рисунок 2.2.5) пользователю нужно ввести дату транзакции, сумму транзакции, выбрать тег расхода, нажав на иконку с изображением и текстовым пояснением, и нажать на кнопку принять. При нажатии на поле ввода даты пользователю на экран выводится календарь при помощи DatePicker в котором пользователю нужно выбрать дату транзакции и подтвердить ее, после выбора даты автоматически заполнится поле ввода даты. Далее пользователю нужно ввести сумму транзакции. После выбора даты и ввода суммы транзакции пользователю нужно выбрать тег транзакции. Пользователю просто нужно нажать на иконку с текстом пояснением и после нажатия заполнится текстовое поля тега расхода. При нажатии кнопки "ПРИНЯТЬ" запускается функция addPay (), которая добавляет в базу данных новую транзакцию с тегом расход и после добавления переносит пользователя на activity всех транзакций TransactionsActivity.

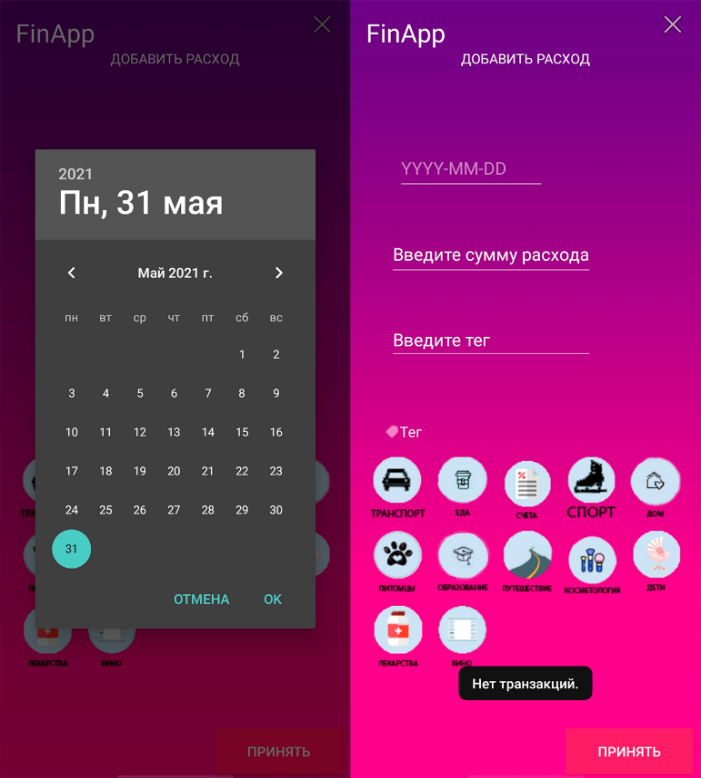


Рисунок 2.2.5 интерфейс ExpensesActivity

TransactionsActivity (рисунок 2.2.6) эта активность в которой пользователь может увидеть диаграмму расходов за период с начала месяца до текущего дня. Также пользователь может увидеть полный список совершенных им транзакций за период с начала месяца до текущего дня.

Пользователь так же может перейти на другие Activity. Например, нажав на текстовое поле "СКАЧАТЬ ОТЧЕТ" пользователя переносит на Pdf Activity, а нажимая на текстовое поле "СМОТРЕТЬ НА ИНТЕРВАЛЕ" пользователя переносит на DateCheck Activity.

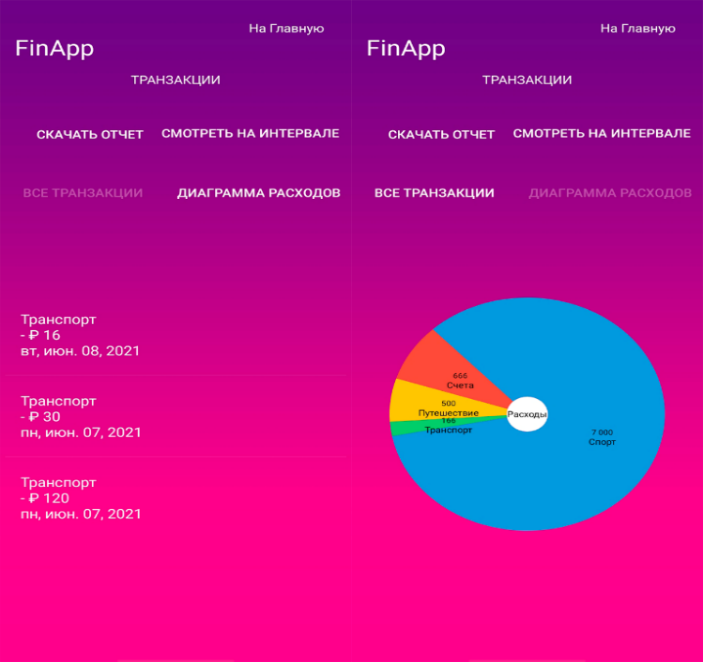


Рисунок 2.2.6 интерфейс TransactionsActivity

Pdf Activity (рисунок 2.2.7) представляет собой активность в которой при нажатии на значок загрузки запускается функция createPdfWrapper(), которая запрашивает у пользователя доступ к хранилищу данных, если же доступ уже получен, то запускается функция createPdf(), которая используя библиотеку iText создает pdf файл. iText — это библиотека классов для создания, анализа и изменения документов в форматах Portable Document Format (PDF), а также XML, HTML и RTF. В pdf файл записывается таблица, в которой описывается транзакции за все время использования приложения.

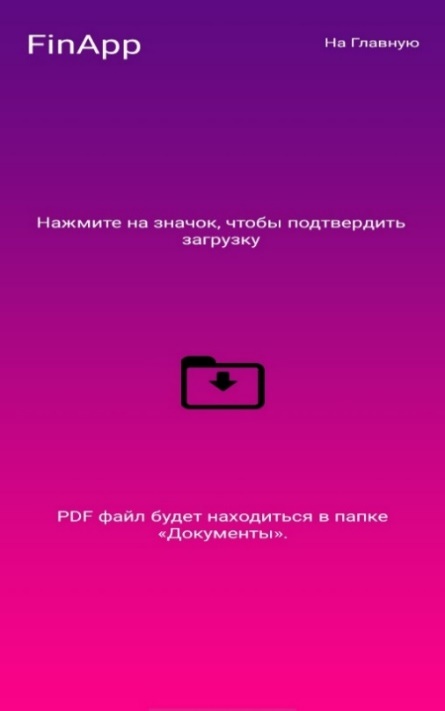


Рисунок 2.2.7 Pdf Activity

Так же после формирования pdf файла, записи туда таблицы и его закрытия, приложение переносит на SimpleMail Activity. SimpleMail Activity (Рисунок 2.2.8) это активность в которой пользователю предлагают выбрать хочет ли он отправить отчет созданный в Pdf Activity. Пользователь может вернуться в главное меню нажав на соответствующий элемент на экране. Нажав на кнопку "ОТПРАВИТЬ ФАЙЛ НА ПОЧТУ" пользователю предстоит выбрать из установленных у него почтовых клиентов. После выбора почтового клиента, приложение заполнит автоматически форму для отправки и пользователю остается только подтвердить отправку сообщения в почтовом клиенте. Для работы с другими приложениями используется специальный подкласс FileProvider, который по умолчанию включает создание URI контента для файлов. URI - последовательность символов, которая идентифицирует абстрактный или физический ресурс. Главное требование при использовании такого метода отправки сообщения на почту это наличие уже настроенного почтового клиента в мобильном телефоне пользователя.



Рисунок 2.2.8 SimpleMail Activity

DateCheck activity это активность в которой пользователь может посмотреть транзакции на интервале от одной даты и до другой. Пользователь выбирает нужную дату начала интервала, с которой он хочет начать просмотр статистики, и выбирает дату конца интервала, до которой он хочет просмотреть статистику. После нажатия на кнопку "РАССМОТРЕТЬ НА ИНТЕРВАЛЕ" появляется круговая диаграмма с расходами за выбранный период и которую можно сменить на список всех транзакций, а также появляется информация с суммой доходов за выбранный период и с суммой расходов за выбранный период.

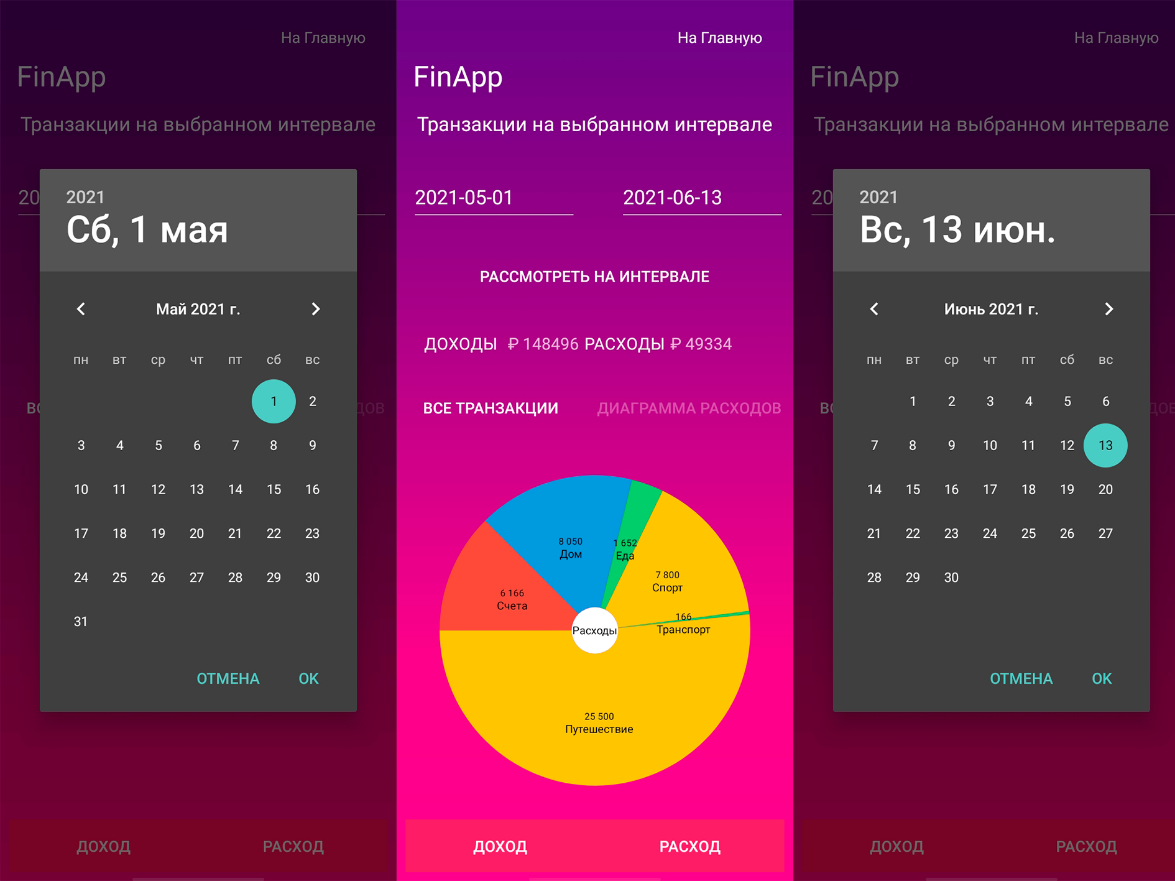


Рисунок 2.2.9 DateCheck activity

# Заключение

В рамках выпускной квалификационной работы было разработано мобильное приложение, позволяющее пользователю вести учет личных финансов.

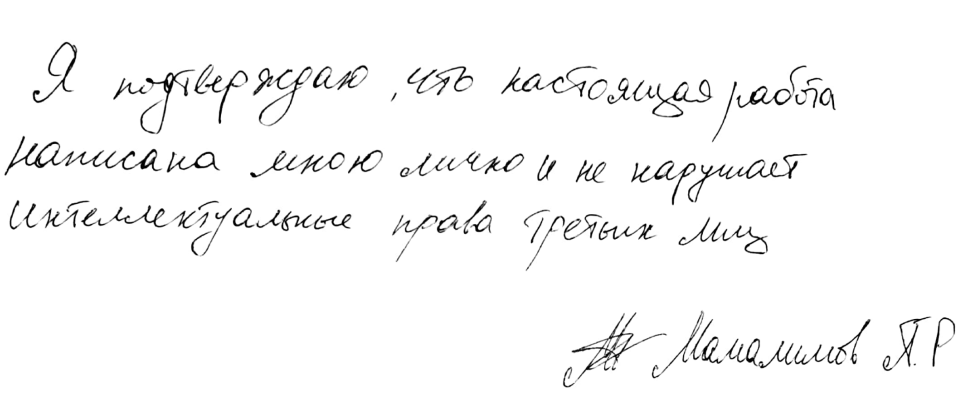
Были проанализированы технологии создания мобильного приложения, также была выбрана операционная система мобильного устройства для работы приложения и среда для разработки приложения.

В ходе выполнения выпускной квалификационной работы были решены следующие задачи исследования:

* проанализированы аналогичные проекты и предметная область
* были рассмотрены разные среды разработки мобильных приложений и была выбрана среда разработки Android Studio для разработки Android приложений.
* реализовано мобильное приложение на базе операционной системы Android

Результатом работы является создание мобильного приложения для управления личными финансами.

Разработанное приложение не имеет неисправностей и готово к использованию. Таким образом, были полностью решены задачи выпускной квалификационной работы и цель исследования была достигнута.



# Список литературы

1. Базы данных для Android [Электронный ресурс] – Режим доступа: <https://xakep.ru/2017/02/20/android-sqlite/#toc05> (дата обращения: 13.05.2021).
2. Библиотека для построения диаграмм – MPAndroidChart [Электронный ресурс] – Режим доступа: <https://github.com/PhilJay/MPAndroidChart> (дата обращения: 15.05.2021).
3. Варакин М.В. Разработка мобильных приложений под Android. УЦ «Специалист» при МГТУ им. Н. Э. Баумана, 2012
4. Гриффитс Д. Head First. Программирование для Android. 2-е издание. / Гриффитс Дон, Гриффитс Дэвид – Санкт-Петербург: Питер, 2018. – 704 с.
5. Дейтел П. Android для программистов: создаем приложения. Издательство Питер, 2012
6. Дэрси Л. Разработка приложений для Android-устройств. Базовые принципы /Л. Дэрси, Ш. Кондер – Том 1. – Москва: Эксмо, 2014. – 598 с
7. Класс FileProvider [Электронный ресурс]. – Режим доступа: <https://developer.android.com/reference/androidx/core/content/FileProvider#ProviderDefinition> (дата обращения: 15.05.2021).
8. Личные финансы. Как управлять и какие инструменты использовать [Электронный ресурс]. - Режим доступа: <http://vseprogroshi.com.ua/ru/lichnye-finansy-kak-upravlyat-i-kakie-instrumenty-ispolzovat.html> (дата обращения 13.05.2021)
9. Майер Р. Программирование приложений для планшетных компьютеров и смартфонов [Текст] / Р. Майер – Москва: Эксмо, 2013. – 816 с.
10. Марио Цехнер. Программирование игр под Android. Санкт-Петербург: Питер, 2013. 688 с.
11. Приложения в магазине «Google Play» – Monefy [Электронный ресурс]. – Режим доступа: <https://play.google.com/store/apps/details?id=com.monefy.app.lite&hl=ru&gl=US> (дата обращения: 28.05.2021).
12. Приложения в магазине «Google Play» – Spendee [Электронный ресурс]. – Режим доступа: <https://play.google.com/store/apps/details?id=com.cleevio.spendee&hl=ru&gl=US>(дата обращения: 28.05.2021).
13. Программирование под Android на Java [Электронный ресурс]. – Режим доступа: <https://metanit.com/java/android/> (дата обращения: 13.05.2021)
14. Руководство DatePicker [Электронный ресурс]. – Режим доступа: <https://betacode.net/12619/android-datepicker> (дата обращения: 20.05.2021)
15. Статистика версий Android [Электронный ресурс]. Режим доступа: <https://rozetked.me/news/10569-google-opublikovala-novuyu-statistiku-versiy-android-u-android-10-8-2> (дата обращения: 28.05.2021)
16. Якоб Нильсен, Ралука Будиу. Как создавать идеально удобные приложения для мобильных устройств. 2013г
17. Ян Ф. Дарвин "Android. Сборник рецептов. Задачи и решения для разработчиков приложений". - издательство Вильямс, 2017. – 768с.



# Приложение

**Класс главного меню приложения**

package spa.sky.finapp;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.Button;  
import android.widget.ListView;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.github.mikephil.charting.charts.PieChart;  
import com.github.mikephil.charting.data.PieData;  
import com.github.mikephil.charting.data.PieDataSet;  
import com.github.mikephil.charting.data.PieEntry;  
import com.github.mikephil.charting.utils.ColorTemplate;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class Dashboard extends AppCompatActivity {  
 int totals;  
 DBHelper databaseHelper;  
 Button take,pay;  
 TextView income, expense,total;  
 PieChart pieChart;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate (savedInstanceState);  
 setContentView (R.layout.activity\_dashboard);  
  
 income = findViewById (R.id.income);  
 expense = findViewById (R.id.expense);  
 take = findViewById (R.id.takeButton);  
 total = findViewById(R.id.total);  
 pay = findViewById (R.id.payButton);  
 pieChart = findViewById (R.id.pieChart);  
  
 databaseHelper = new DBHelper(this);  
  
 databaseHelper.setIncomeExpenses ();  
  
 income.setText ("₽ " + Utils.income);  
 expense.setText ("₽ " + Utils.expense);  
 totals = Utils.income - Utils.expense;  
  
 total.setText("₽ " + totals );  
  
 take.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(getApplicationContext (), IncomeActivity.class);  
 startActivity (intent);  
 finish ();  
 }  
 });  
  
 pay.setOnClickListener (new View.OnClickListener () {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(getApplicationContext (), ExpensesActivity.class);  
 startActivity (intent);  
 finish ();  
 }  
 });  
  
  
  
 TextView showAll = findViewById (R.id.showAllTextView);  
 showAll.setOnClickListener (new View.OnClickListener () {  
 @Override  
 public void onClick(View view) {  
 Intent intent = new Intent(getApplicationContext (), TransactionsActivity.class);  
 startActivity (intent);  
 finish ();  
  
 }  
 });  
  
 getTList();  
 getTPieChart();  
  
 }  
  
  
  
 public void onBackPressed() {  
 Intent intent = new Intent(this, MainActivity.class);  
 intent.setFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);  
 intent.putExtra("Выйти", true);  
 startActivity(intent);  
 finish();  
 }  
  
 public void getTList() {  
 ListView myListView = findViewById(R.id.transactionsListDash);  
  
 ArrayList<String> transact = databaseHelper.getTransactions ();  
 if (transact.size ( ) == 0) {  
 Toast.makeText (Dashboard.this, "Нет транзакций", Toast.LENGTH\_LONG).show ( );  
  
 }  
  
 else {  
 List<String> transactMini = new ArrayList<> ( );  
 transactMini.add (transact.get (0));  
 if (transact.size () >= 2)  
 transactMini.add (transact.get (1));  
  
 ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String> (this, android.R.layout.simple\_list\_item\_1, transactMini);  
  
 myListView.setAdapter (arrayAdapter);  
  
 myListView.setOnItemClickListener (new AdapterView.OnItemClickListener ( ) {  
  
 @Override  
 public void onItemClick(AdapterView<?> parent, View view, int position, long id) {  
  
 Toast.makeText (Dashboard.this, "Нажмите \* Показать все \*, чтобы просмотреть все транзакции.", Toast.LENGTH\_LONG).show ( );  
 }  
 });  
  
 }  
 }  
 public void getTPieChart() {  
  
 List<PieEntry> value = new ArrayList<> ( );  
 if (Utils.expense != 0 && Utils.income !=0) {  
  
 pieChart.setHoleRadius (15f);  
 pieChart.setTransparentCircleRadius (15f);  
  
 value.add (new PieEntry (Utils.expense, "Расходы"));  
 value.add (new PieEntry (Utils.income, "Доходы"));  
  
 PieDataSet pieDataSet = new PieDataSet (value, "Транзакции");  
 PieData pieData = new PieData (pieDataSet);  
 pieChart.setData (pieData);  
  
 pieChart.setContentDescription (null);  
 pieChart.getLegend ( ).setEnabled (false);  
 pieChart.getDescription ( ).setEnabled (false);  
 pieChart.setEntryLabelTextSize (10f);  
 pieChart.animateXY (1000, 1000);  
 pieDataSet.setColors (ColorTemplate.MATERIAL\_COLORS);  
 }  
 }  
}