ВВЕДЕНИЕ

Целью данного курсового проекта является изучение, проектирование и написание приложения для ОС Android.

Android - операционная система для смартфонов, планшетов, электронных книг, цифровых проигрывателей, наручных часов, фитнес-браслетов, игровых приставок, ноутбуков, нетбуков, смартбуков, очков Google Glass, телевизоров и других устройств. Основана на ядре Linux и собственной реализации виртуальной машины Java от Google [4]. Изначально разрабатывалась компанией Android, Inc., которую затем выкупила Google.

Данная система была выбрана не случайно. На данный момент она является одной из самых популярных систем в мир, а также для нее есть написана хорошая документация, которая будет понятна даже новичку в данной области. Благодаря распространённости данной системы, сообщество разработчиков велико, что позволяет найти необходимый совет, при необходимости.

Для разработки было выбрано приложение под названием «Покормите студента». Приложение позиционируется как книга несложных кулинарных рецептов на любой случай жизни.

Существует определенное множество подобных приложений, но практически во всех, рецепты достаточно сложные, требует специфических инструментов или ингредиентов. Эта проблема и решена в приложении «Покормите студента». При составлении базы рецептов, важным критерием было то, что рецепт не должен требовать особых навыков в области кулинарии или специфических инструментов. Как следует из названия, целевая аудитория данного приложения – студенты, желающие разнообразить свой рацион.

1 АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ

Целью курсового проекта создание приложения для хранения кулинарных рецептов. Исходя из цели работы, к приложению были выдвинуты следующие требования:

- приложение содержит изначальную базу рецептов;
- пользователь имеет возможность добавить свой рецепт;
- пользователь может добавлять рецепты в «Избранное»;
- пользователь имеет возможность редактировать рецепты;
- рецепты сгруппированы по категориям;
- доступна сортировка рецептов по названия;
- доступна сортировка рецептов по ингредиентам.

При реализации интерфейса приложения должны быть учтены принципы проектирования UI Google Material Design. Также интерфейс приложения должен быть интуитивно понятен пользователю.

Приложение должно содержать следующие основные формы (экраны):

1. Главный экран. Содержит категории рецептов такие как «Завтрак», «Быстрый перекус», «Ужин» и др. Также группы «Мои рецепты» - рецепты, добавленные пользователем и «Любимые» - рецепты, которые понравились пользователю (рисунок 1.1).

Рисунок 1.1 – Главный экран

2. Список рецептов. Экран отображает рецепты для выбранной группы или категории. Также содержит возможность добавить рецепт (рисунок 1.2).

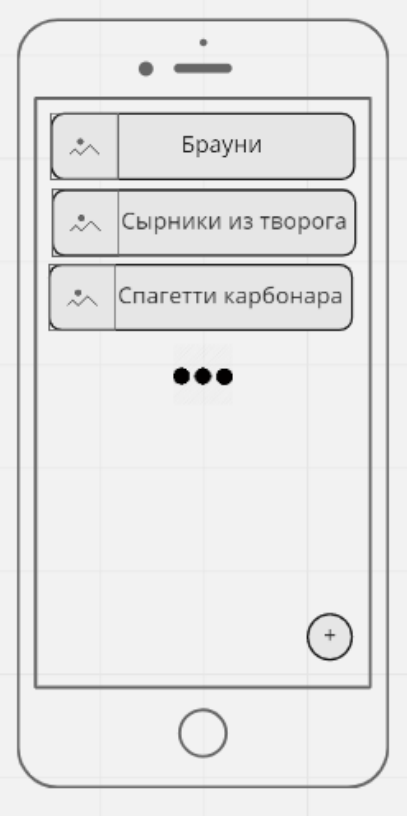

Рисунок 1.2 – Экран списка рецептов

3. Создание рецепта. Экран для добавления рецепта: ввода названия, описания, выбор картинки. Также позволяет добавлять этапы готовки и менять их порядок (рисунок 1.3).

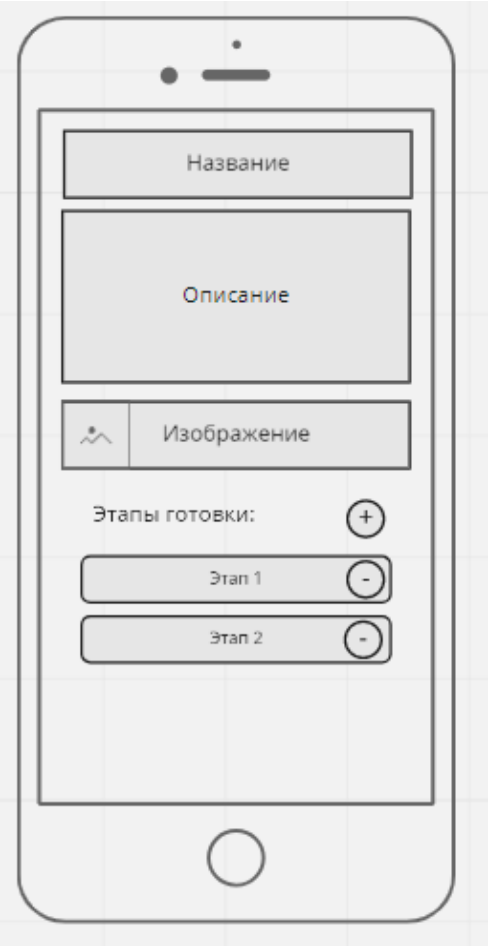

Рисунок 1.3 – Экран создания и редактирования рецепта

4. Редактирования рецепта. Аналогичен экрану создания рецепта (рисунок 1.3).
5. Просмотр рецепта. Экран для подробного просмотра рецепта, картинки рецепта и всех этапов готовки.

2 ПРОГРАММНОЕ ПРОЕКТИРОВАНИЕ

2.1 Функциональное проектирование

Компания Google не устанавливает строгие требования по использованию определенной архитектуры в Android приложении. Существуют различные способы реализации архитектуры в приложении, однако наиболее популярные из них основаны на таких паттернах как MVVM, MVP и MVC.

При проектировании приложения было принято использовать паттерн MVC, так как при его использовании компоненты приложения остаются максимально независимыми друг от друга, что способствует построению наиболее гибкой архитектору.

MVP – паттерн разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: model, view, presenter [4]. У каждого из них есть свои обязанности, связь между ними происходит через presenter.

Model содержит бизнес-логику приложения. Он контролирует как данные создаются, сохраняются и изменяются.

View это интерфейс, который отображает данные и направляет действия пользователя в Presenter.

Presenter выступает в роли посредника. Он извлекает данные из Model и показывает их в View (Activity, Fragment, Dialog и т.д.). Он также обрабатывает действия пользователя, полученные из View [6].

MVP является аналогом MVC, но с более современной парадигмой, которая создаёт лучшее разделение ролей и максимизирует тестируемость приложения. Схема работы MVP представлена на рисунке 2.1.1.

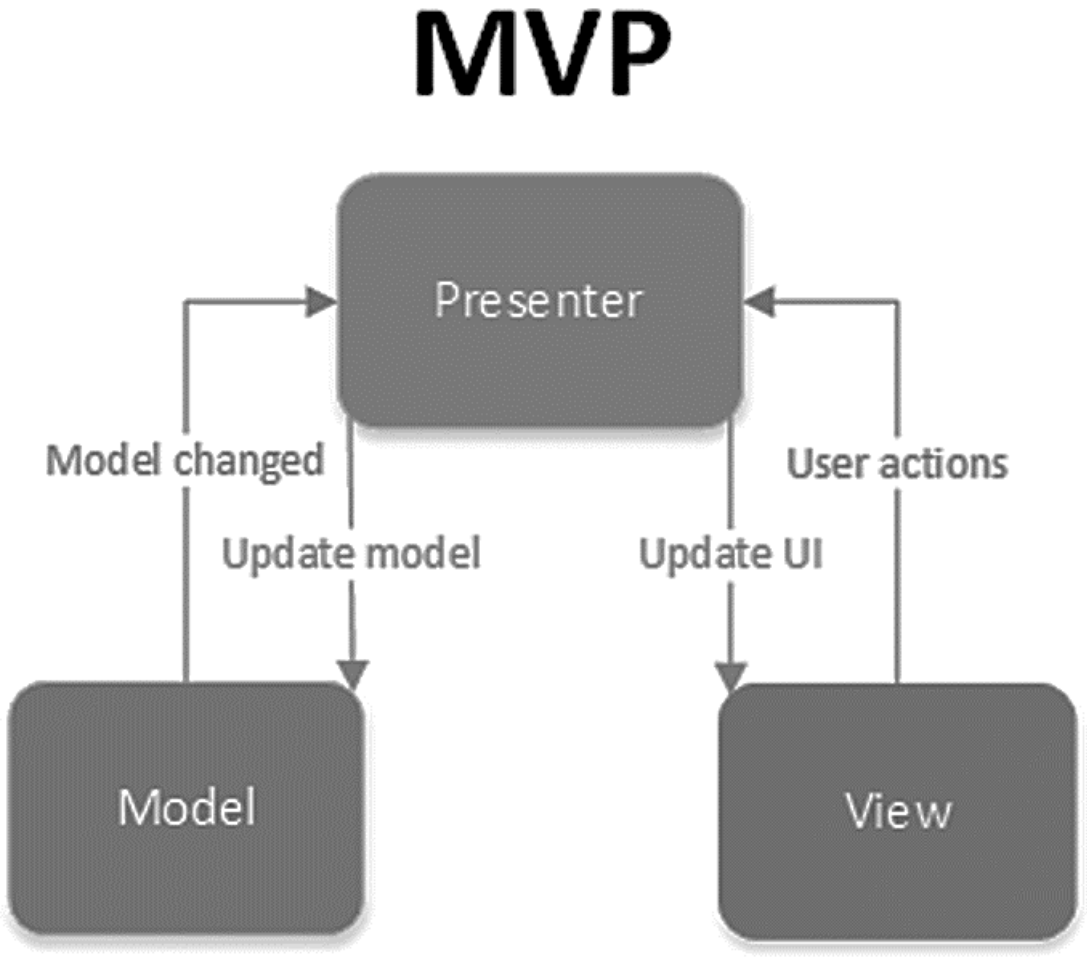

Рисунок 2.1.1 – Схема MVP

Применяя паттерн MVP к приложению Android, нужно определиться с зоной ответственности для каждой части.

Как было сказано ранее, класс Model содержит бизнес-логику приложения. В случае приложения рецептов можно сюда можно отнести классы взаимодействия с базой данных рецептов, классы сущностей рецептов и этапов готовки, класс чтения-сохранения картинок, а также класс для хранения настроек приложения (рисунок 2.1.2). Также к модели можно добавить класс посредник - репозиторирий, для агрегации всех данных о рецептах.

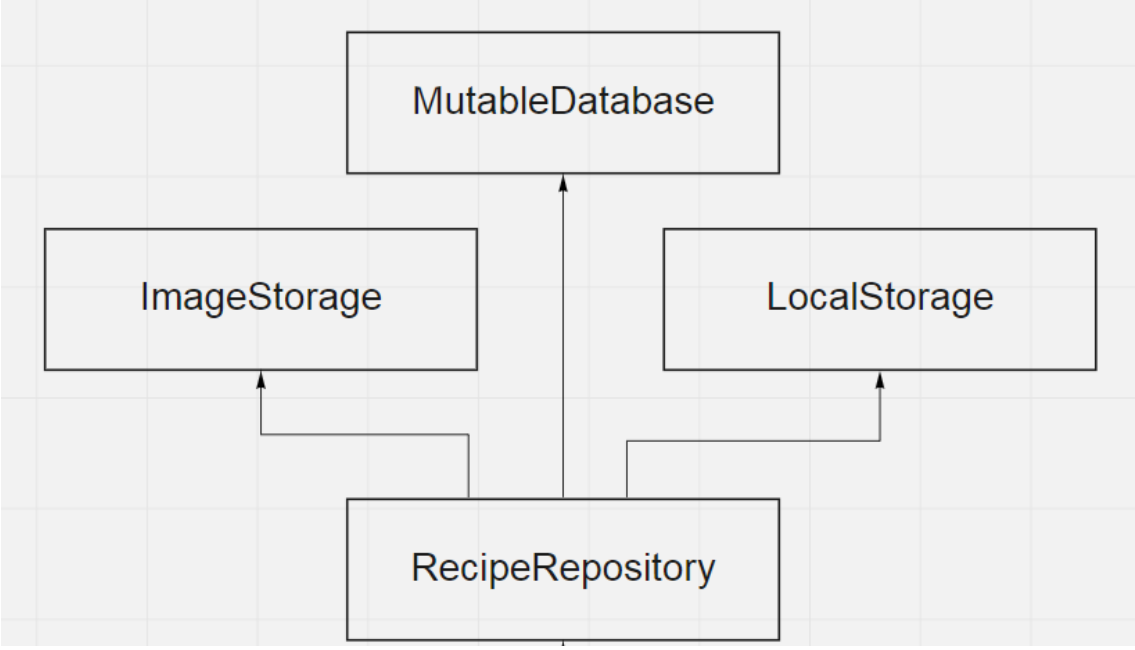

Рисунок 2.1.2 – Диаграмма роли модели в приложении рецептов

В Android приложение роль View играют классы форм. Такие как Activity и Fragment. Так как view-классы ответственны за взаимодействие с пользователем, то к ним также стоит отнести ListAdapter и классы диалогов (рисунок 2.1.3).

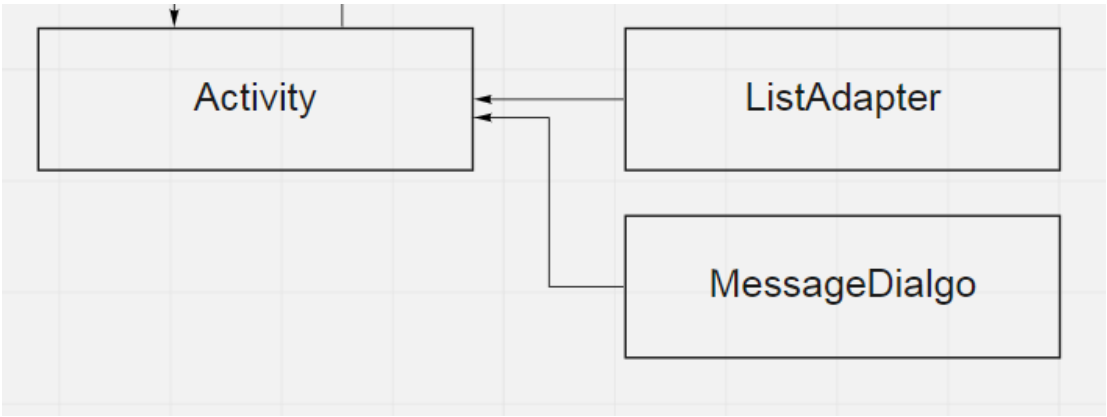

Рисунок 2.1.3 – Диаграмма роли View в Android приложении

Следовательно, класс Presenter берет обязанность за взаимодействие с репозиторием рецептов и контроль отображения рецептов пользователю. Таким образом общая схема MVP применённая к Android приложению изображена на рисунке 2.1.4.

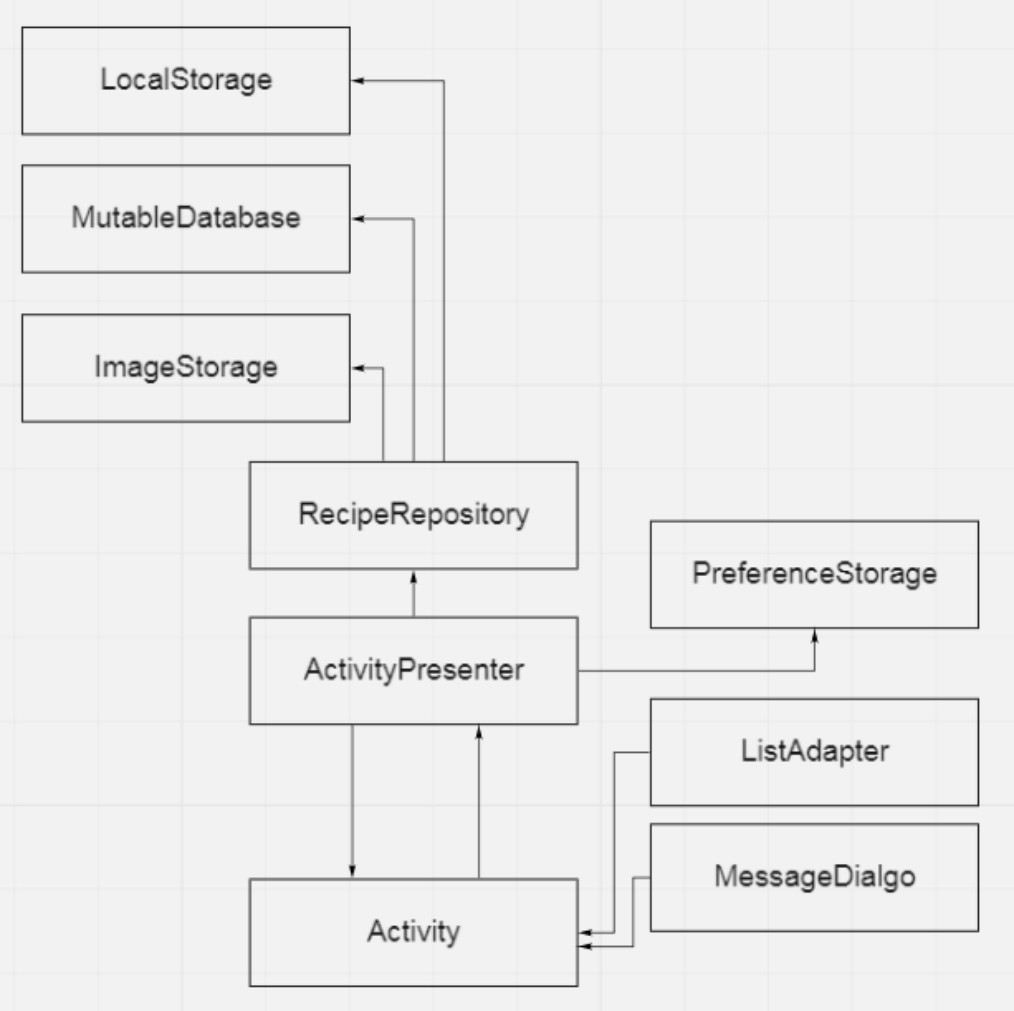

Рисунок 2.1.4 – общая схема MVP для Android приложения

Важным моментом при проектировании Android приложения является «жизненный цикл» view-элементов, в частности Activity и Fragment [1]. Это создает дополнительные трудности при реализации MVP паттерна, т к Android view элементы пересоздаются при изменении состояния устройства (например, смена ориентации). То есть, для корректной реализации MCP необходимо сохранять экземпляр Presenter класса при изменении состояния view.

Для сохранения состояний существуют множество способов, каждый из которых обладает своими достоинствами и недостатками. Одним из таких способов является использование Android класса - SupportLoaderManager. Данный класс не подвержен изменению состояния view, что дает возможность использовать его как хранилище для Presenter.

2.2 Проектирование базы данных

Для реализации функций добавления, редактировании и удаления рецептов необходимо использовать базу данных.

Изучив способы хранения крупных объемов данных для Android приложений в качестве базы данных было принято решение воспользоваться сторонней библиотекой Paper Database. Данная библиотека отлично подходит для быстрого сохранения несложных объектов в постоянную память Android устройства. Если сравнивать данную библиотеку со стандартным способом хранения больших объемов данных, который предоставляет Android SDK – SQLLite, то можно выделить следующие преимущества:

- скорость работы;
- простоту в использовании;
- возможность как синхронного доступа, так и асинхронного доступа;
- простоту обновления версий базы данных.

Кроме ранее перечисленных плюсов, библиотека имеет подробную документацию, которая даст ответы на всевозможные вопросы при реализации.

В первую очередь приложение рецептов должно хранить рецепты. Рассмотрим составные элементы каждого рецепта:

1. Название и описание рецепта. Существуют для каждого рецепта. Эти поля представлены типом String в базе данных.
2. Этапы готовки. Каждый этап готовки содержит краткое описание (представленное типом String) и время, требуемое для его выполнения (тип Long – миллисекунды, для удобной конвертации в любую единицу измерения). Этапов готовки в рецепте может содержаться неограниченное множество.

3. К описанию каждого рецепта прилагается фотография. Есть два способа хранения фотографии.
    a. Хранить фотографию непосредственно в объекте рецепта использую класс Bitmap. В таком случае фотографии будут подгружаться в оперативную память при первой загрузке рецепта, и храниться там до закрытия приложения.
    b. В объекте рецепта хранить идентификатор фотографии, и загружать нужную фотографию при необходимости.

Несмотря на то, что второй вариант требует большего количества кода при его реализации, он менее требователен к ресурсам устройства. Поэтому было принято решение использовать именно его.

Исходя из поставленных ранее требований, выделим следующие поля для каждой записи рецепта:

– id – уникальный идентификатор, для быстрого доступа к нужному рецепту (тип Integer). Поле недоступно для изменения пользователем, генерируется автоматически;

– name – название рецепта (тип String). Поле, доступное для изменения пользователем;

– description – текстовое описание рецепта (тип String). Поле, доступное для изменения пользователем;

– steps – список этапов готовки рецепта (тип List);

– imageName – название изображения, для отображения в заголовке рецепта (тип String). Поле недоступно для изменения пользователем. Значение присваивается автоматически, при выборе пользователем картинки.

Также в объекте этапа приготовления рецепта будут использоваться следующие поля:

– description – текстовое описание этапа (тип String);

– time – время, необходимое для выполнения этапа (тип Long).

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ

3.1 Детальная реализация функциональных частей ПО

Процесс реализации проекта начался с описания основных классов для форм, классов управляющей логики, и класса репозитория. В результате были реализованы следующие View:

– MainActivity – форма для отображения категорий;

– ListActivity – форма отображения списка рецептов;

– DescriptionActivity – форма просмотра конкретного рецепта;

– CreateEditActivity – форма создания и редактирования рецепта;

– CookStepEditDialog – диалог создания и редактирования этапа рецепта.

Подключение ранее перечисленных Acvitity классов к приложению можно наблюдать в AndroidManifest файле (листенинг 3.1.1) в строках 24, 28 32, 36 и 40.

Листинг 3.1.1 – AndroidManifest файл приложения

```
   1:  <?xml version="1.0" encoding="utf-8"?>
   2:  <manifest
xmlns:android="http://schemas.android.com/apk/res/android"
   3:      package="by.wiskiw.studentfood">
   4:
   5:      <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
   6:
   7:      <application
   8:          android:name=".di.FoodApp"
   9:          android:allowBackup="true"
   10:         android:icon="@mipmap/ic_launcher"
   11:         android:label="@string/app_name"
   12:         android:roundIcon="@mipmap/ic_launcher_round"
   13:         android:supportsRtl="true"
   14:         android:theme="@style/AppTheme">
   15:         <activity
   16:
android:name=".ui.activity.main.MainActivity"
   17:
android:theme="@style/AppTheme.SplashScreen">
```

```
   18:             <intent-filter>
   19:                 <action
android:name="android.intent.action.MAIN" />
   20:
   21:                 <category
android:name="android.intent.category.LAUNCHER" />
   22:             </intent-filter>
   23:         </activity>
   24:         <activity
   25:
android:name=".ui.activity.list.StaticRecipesListActivity"
   26:             android:label="@string/title_activity_list"
   27:             android:theme="@style/AppTheme" />
   28:         <activity
   29:
android:name=".ui.activity.list.FavoriteRecipesListActivity"
   30:             android:label="@string/title_activity_list"
   31:             android:theme="@style/AppTheme" />
   32:         <activity
   33:
android:name=".ui.activity.list.MyRecipesListActivity"
   34:             android:label="@string/title_activity_list"
   35:             android:theme="@style/AppTheme" />
   36:         <activity
   37:
android:name=".ui.activity.description.DescriptionActivity"
   38:
android:label="@string/title_activity_scrolling"
   39:             android:theme="@style/AppTheme.NoActionBar"
/>
   40:         <activity
   41:
android:name=".ui.activity.create.edit.CreateEditActivity"
   42:
android:label="@string/create_edit_activity_create_title"
   43:             android:screenOrientation="locked"
   44:             android:theme="@style/AppTheme" />
   45:     </application>
   46:
   47: </manifest>
```

Основная логика обработки отображения и взаимодействия с пользователем размещается в следующий классах:

– MainPresenter;

– ListPresenter;

– DescriptionPresenter;

– CreateEditPresenter.

Для сохранения и чтения рецептов, а также предоставления доступа к картинкам был реализован класс-репозитория рецептов – RecipesRepository. Репозиторий рецептов осуществляет взаимодействие со следующими классами:

– DummyRecipeReader – чтение первичного списка рецептов из XML файла;

– RecipeDao – предоставляет доступ к базе данных рецептов

– FoodFileManager – осуществляет контроль за сохранением и чтением картинок рецептов.

В результате была получена следующая схема взаимодействия классов, изображенная на рисунке 3.1.1 (для упрощения схемы соответствующие классы Activity и классы Presenter были объединены).

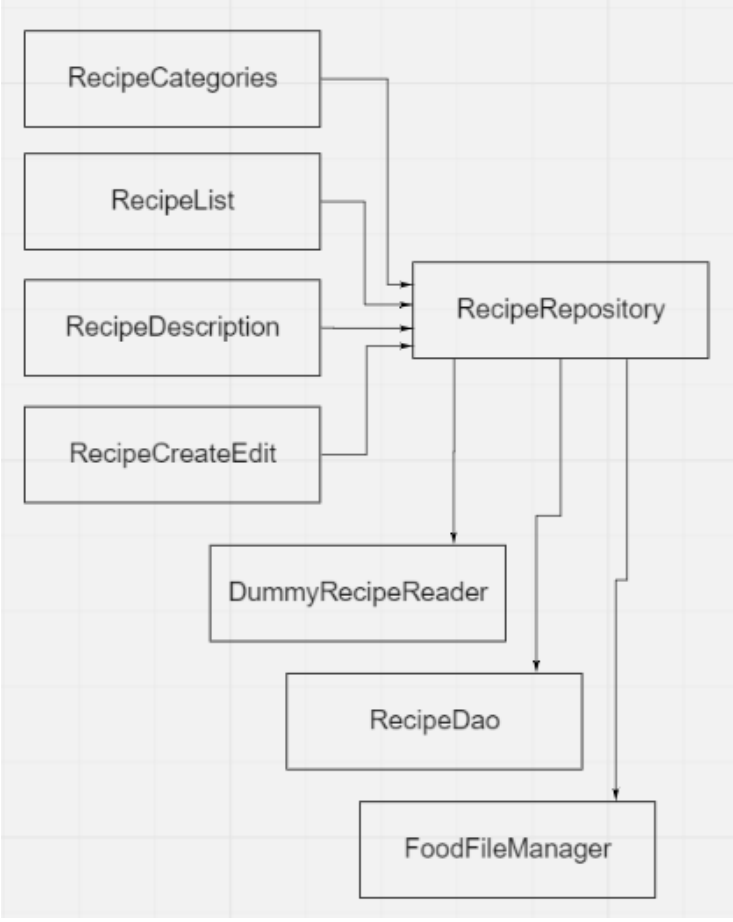

Рисунок 3.1.1 – общая схема MVP для Android приложения

Интерфейс взаимодействия с пользователем проектировался в XML файлах разметки. Пример верстки элемента для отображения рецепта в списке приведет в листинге 3.1.2.

Листинг 3.1.2 – Разметка элемента рецепта в списке

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:clickable="true"
    android:focusable="true"
    android:foreground="?android:attr/selectableItemBackground"
    app:cardCornerRadius="4dp">

    <android.support.constraint.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <ImageView
            android:id="@+id/recipe_image_view"
            android:layout_width="128dp"
            android:layout_height="128dp"
            android:scaleType="centerCrop"
            android:src="@drawable/template" />

        <TextView
            android:id="@+id/recipe_title_text_view"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:ellipsize="end"
            android:maxLines="1"
            android:padding="8dp"
            android:textStyle="bold"
            app:layout_constraintLeft_toRightOf="@+id/recipe_image_view"
            app:layout_constraintRight_toRightOf="parent" />

        <TextView
            android:id="@+id/recipe_description_text_view"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:ellipsize="end"
```

```
    84:             android:maxLines="3"
    85:             android:paddingStart="8dp"
    86:             android:paddingTop="4dp"
    87:             android:paddingEnd="8dp"
    88:             android:paddingBottom="8dp"
    89:
app:layout_constraintLeft_toRightOf="@+id/recipe_image_view"
    90:
app:layout_constraintRight_toRightOf="parent"
    91:
app:layout_constraintTop_toBottomOf="@+id/recipe_title_text_v
iew"/>
    92:
    93:         <TextView
    94:             android:id="@+id/cock_time_text_view"
    95:             android:layout_width="wrap_content"
    96:             android:layout_height="wrap_content"
    97:             android:gravity="bottom"
    98:             android:paddingStart="8dp"
    99:             android:paddingEnd="8dp"
    100:                 android:paddingBottom="4dp"
    101:
app:layout_constraintBottom_toBottomOf="parent"
    102:
app:layout_constraintRight_toRightOf="parent"
    103:
app:layout_constraintTop_toBottomOf="@+id/recipe_description_
text_view" />
    104:
    105:         </android.support.constraint.ConstraintLayout>
    106:
    107:     </android.support.v7.widget.CardView>
```

При реализации интерфейса использовались стандартные view-компоненты, такие как TextView, ImageView, EditText, Button и ImageButton.

Кроме компонентов, входящих в Android SDK, были использованы контейнеры CardView и ConstraintLayout из библиотеки Android Support.

На рисунке 3.1.2 изображены основные экраны приложения. На рисунке 3.1.3 изображен диалог добавления(редактирования) этапа рецепта.

Как и требовалось, все экраны и view-элементы были реализованы с использованием принципов проектирования UI – Material Design.

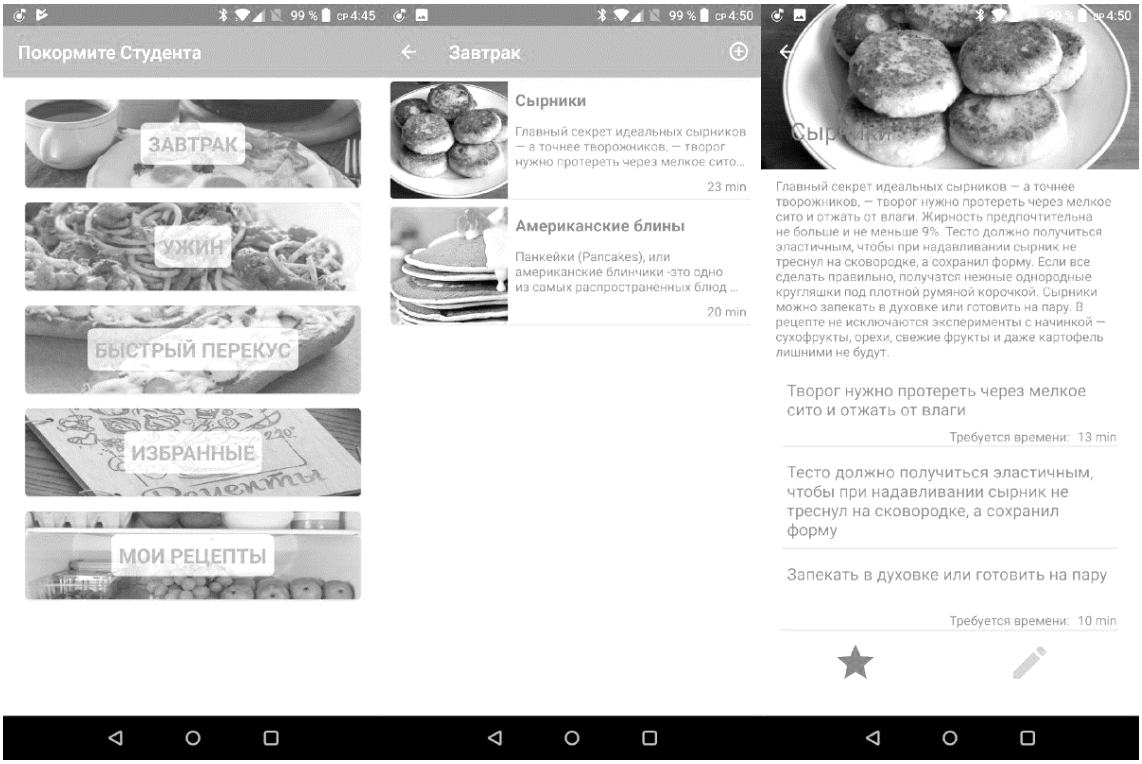

Рисунок 3.1.2 – основные экраны приложения

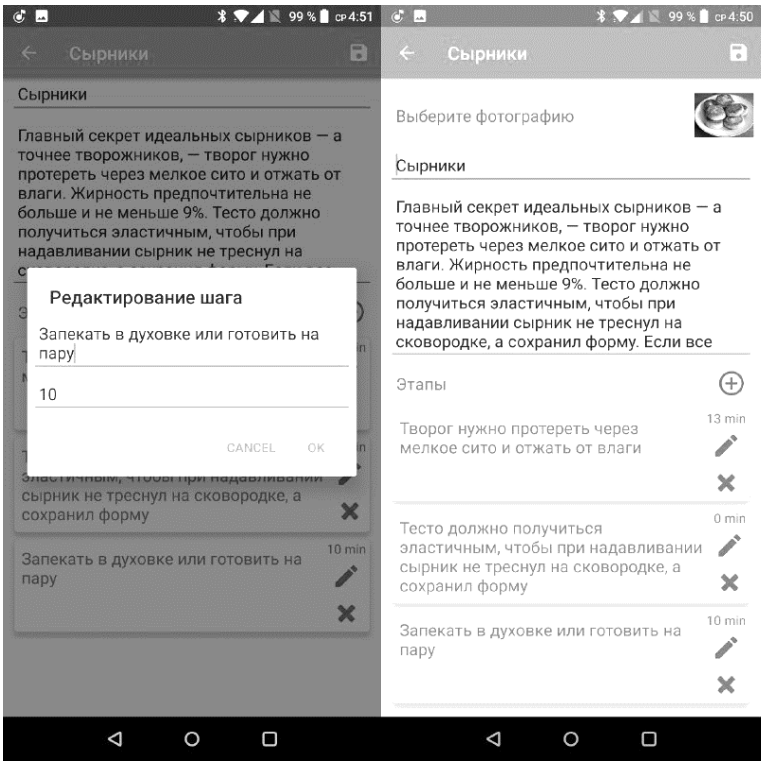

Рисунок 3.1.3 – экран редактирования рецепта

Начиная с Android 6.0 для корректной работы с базой данных и картинками приложению необходимо получить согласие от пользователя на использование внутренней памяти устройства. До версии Android 6.0, данные права предоставлялись приложениям по умолчанию. Для получения необходимых привилегий необходимо добавить запрос в AndroidManifest файл (листинг 3.1.3).

Листинг 3.1.3 – Запрашиваемые разрешения у пользователя

```
1: <uses-permission
   android:name="android.permission.WRITE_EXTERNAL_STORAGE"/
   >
2: <uses-permission
   android:name="android.permission.READ_EXTERNAL_STORAGE"/>
3: <uses-permission
   android:name="android.permission.CAMERA"/>
```

Кроме требования на использование внутренней памяти устройство, для возможности делать фотографию, необходимо получить разрешения на использование камеры. Запрос на использование камеры также прописывается в AndroidManifest - строка 3, листинг 3.1.3

3.2 Сопроводительная документация

Сопроводительная документация по разработанному мобильному приложению представлена в приложении А в виде технического задания.

3.3 Анализ ПО

Приложение было разработано, в соответствии современным тенденциям программирования. Были соблюдены все возможные рекомендации по проектированию и разработке Android приложений от Google, что повышает безопасность и делает приложение менее требовательным к ресурсам устройства.

Благодаря использованию паттерна проектирования MVP, поддержка данного ПО не доставит трудностей. А использование системы контроля версия VCS Git, упрощает поиск и исправление багов.

За счет вынесение всех строковых значений в файлы ресурсов, приложение может быть достаточно легко локализировано под нужный язык. В данный момент приложение доступно на двух языках: русский и английский, но при необходимости этот список может быть расширен. При отсутствии нужного файла локализации, приложение будет запущено на английском языке.

3.4 Тестирование программного обеспечения

Тестирование программного обеспечения – процесс исследования, испытания программного продукта. Целью тестирования является выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации.

Тестированием программного обеспечения проводилось в определённом порядке с целью получения информации о качестве работы.

Тестирование обеспечивает:

– Обнаружение ошибок.

– Демонстрацию соответствия функций программы ее назначению.

– Демонстрацию реализации требований к характеристикам.

Приложение было протестировано на нескольких устройствах с различными характеристиками. Основное тестирование проходилось при помощи смартфона OnePlus 5T со следующими техническими характеристиками:

- Операционная система: Android 8.1;
- Экран 5.4”;
- Разрешение экрана 720x1280;
- ОЗУ 8GB.
- 128 Gb встроенного хранилища.

Также финальная версия приложения была протестирована на следующих устройствах:

– Samsung Galaxy Nexus (2011 г.) Android 4.3

– Google Nexus 4 (2012 г.) Android 4.4

– Google Nexus 5 (2013 г.) Android 5.0

– Google Nexus 6P (2015 г.) Android 7.1

– Google Pixel (2016 г.) Android 9.0

Корректность выполнения определяется сравнением фактического и ожидаемого результатов. Если результаты совпали, то тестирование пройдено успешно. Если же результаты не равны, то произойдет сбой системы и пользователю выведется системное окно об ошибке. Все действия, проведённые в процессе тестирования, а также их результаты приведены в таблице 3.4.1.

Таблица 3.4.1 – Тестирование программы

Действие	**Ожидаемый результат**	**Результат**
Запуск программы	Открытие формы категории рецептов	Открытие формы категории рецептов
Выбор категории «Завтрак»	Открытие экрана со списком рецептов для завтрака	Открытие экрана со списком рецептов для завтрака
Выбор категории «Быстрый перекус»	Открытие экрана со списком рецептов для «Быстрого перекуса»	Открытие экрана со списком рецептов для «Быстрого перекуса»
Клик по кнопке «Выбрать фото»	Запускается экран выбора фотографии	Запускается экран выбора фотографии

Продолжение таблицы 3.4.1

Нажатие кнопки «Добавить новый рецепт»	Открытие экрана добавления рецепта	Открытие экрана добавления рецепта
Выбор опции «Удалить» для своего рецепта из списка	Удаление выбранного рецепта из списка	Удаление выбранного рецепта из списка
Выбор опции «Редактировать» для своего рецепта на экране просмотра	Запускается экран редактирования для выбранного экрана	Запускается экран редактирования для выбранного экрана
Клик по кнопке «Сохранить» на экране создания рецепта	Экране редактирования закрывается, рецепт появляется в списке	Экране редактирования закрывается, рецепт появляется в списке
Выбор опции «Сортировка по названию» на экране списка рецептов	Список сортируется по названию	Список сортируется по названию
Выбор опции «Сортировка по времени готовки» на экране списка рецептов	Список сортируется по времени готовки	Список сортируется по времени готовки

В ходе тестирования не было выявлено каких-либо ошибок и багов. Приложение работает стабильно и его функционал полностью дееспособное.

ЗАКЛЮЧЕНИЕ

В данной курсовой работе была поставлена цель, используя полученные знания за курс «Операционное системы и системное программирование», реализовать мобильное приложение для операционной системы Android. В результате было получено приложение, которое соответствует требованиям Google о проектировании интерфейсов приложений Material Design, а также принципам Clean Architecture за счет использования паттерна MVP.

Результатом выполнения курсового проекта является мобильное приложение «Покормите студента». В первую очередь рассчитанное на студентов, желающие разнообразить свой рацион.

В перспективе, использование паттернов MVP и «репозиторий» позволит расширять функционал без переписывания значительной части кода. Например, при необходимости добавить функцию синхронизации списка рецептом между несколькими устройствами, локальное хранилище рецептов можно заменить на удаленный сервер. Или заменить реализацию экранов с Activity на Fragment без переписывания бизнес логики.

В итоге приложение реализовано в соответствии всем заявленным требованиям, протестирована надлежащим образом, и может быть опубликовано в магазине приложений Google Play. Поставленная задача была выполнена.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Официальная документация Android [Электронный ресурс] – режим доступа: https://developer.android.com/. Дата доступа 10.11.18.
2. Официальная документация Firebase [Электронный ресурс] – режим доступа: https://firebase.google.com/docs/. Дата доступа 20.11.18.
3. Интернет портал “Java-Help” [Электронный ресурс] – режим доступа: http://java-help.ru/model-view-presenter-android-part-1/. Дата доступа 14.11.18.
4. Интернет портал «Википедия» [Электронный ресурс] – режим доступа: https://ru.wikipedia.org/. Дата доступа 20.11.18.
5. Интернет портал «Освой программирование играючи» [Электронный ресурс] – режим доступа: http://developer.alexanderklimov.ru/android/index.php. Дата доступа 20.11.18.
6. Интернет портал «Habr» [Электронный ресурс] – режим доступа: https://habr.com/. Дата доступа 20.11.18.

ПРИЛОЖЕНИЕ А

(обязательное)

Техническое задание

Введение

Мобильное приложение-книга рецептов для ОС Android "Накормите студента". Приложение предоставляет функционал для просмотра, сохранения и редактирования простых кулинарных рецептов. Целевая аудитория студенты со средним и малым достатком, проживающие в общежитии.

А.1 Основание для разработки

Приложение разрабатывалось в рамках курсового проекта студента учреждения образования «Полоцкий государственный университет» Яблонского А. С.

А.2 Назначение разработки

Разнообразить питание студентов. Обогатить их знания в кулинарном деле. Создать удобный интерфейс для быстрого поиска и сохранения рецептов.

А.3 Требования к программе или программному изделию

А.3.1 Требования к функциональным характеристикам

Требования к функциональным характеристикам приложения:

1. Группировка рецептов по категориям;
2. Сортировка рецептов по названия;
3. Сортировка рецептов по длительности готовки;
4. Поиск рецептов по ингредиентам;
5. Возможности сохранить рецепт в Избранные;

6. Возможность сохранить свой рецепт (текстовая информация, картинки);
7. Возможность изменить свой рецепт.

А.3.2 Требования к надежности

Приложение должно стабильно работать - без критических программных сбоев. Обрабатывать возможные ошибки пользователей.

А.3.3 Условия эксплуатации

Для полноценной эксплуатации данного приложения, нужен смартфон под управлением операционной системы Android версией не ниже 5.0.

А.3.4 Требования к составу и параметрам технических средств

Для обеспечения устойчивости работы данного приложения требуется:

1. Операционная система: Android;
2. Версия Android не ниже 5.0;
3. Базовые навыки пользования смартфоном.

А.3.5 Требования к информационной и программной совместимости

Для успешной работы приложения, необходим смартфон под управлением операционной системы Android.

А.3.6 Требования к маркировке и упаковке

Требования к маркировке и упаковке отсутствуют.

А.3.7 Требования к транспортированию и хранению

Программное средство должно храниться на компакт-диске в виде исполняемых файлов программного продукта.

А.4 Требования к программной документации

В качестве программной документации по приложению должно быть техническое задание согласно ГОСТ 19.201-78.

Требования к перечисленным программным документам устанавливаются государственными стандартами ЕСПД.

А.5 Стадии и этапы разработки

Разработка приложения заключается в следующем:

1. Анализ исходных данных и постановка задачи проектирования, разработка технического задания;
2. Разработка пользовательского интерфейса;
3. Разработка основного функционала;
4. Разработка дополнительного функционала;
5. Разработка системы сохранения рецептов;
6. Тестирование приложений;
7. Разработка программной документации.

А.6 Порядок контроля и приемки

Тестирование программного обеспечения состоит из проверки всех функциональных характеристик. Основным критерием оценки правильности работы, являлся визуальный контроль выполнения программой требующихся функций.

ПРИЛОЖЕНИЕ Б

Диаграмма вариантов использования

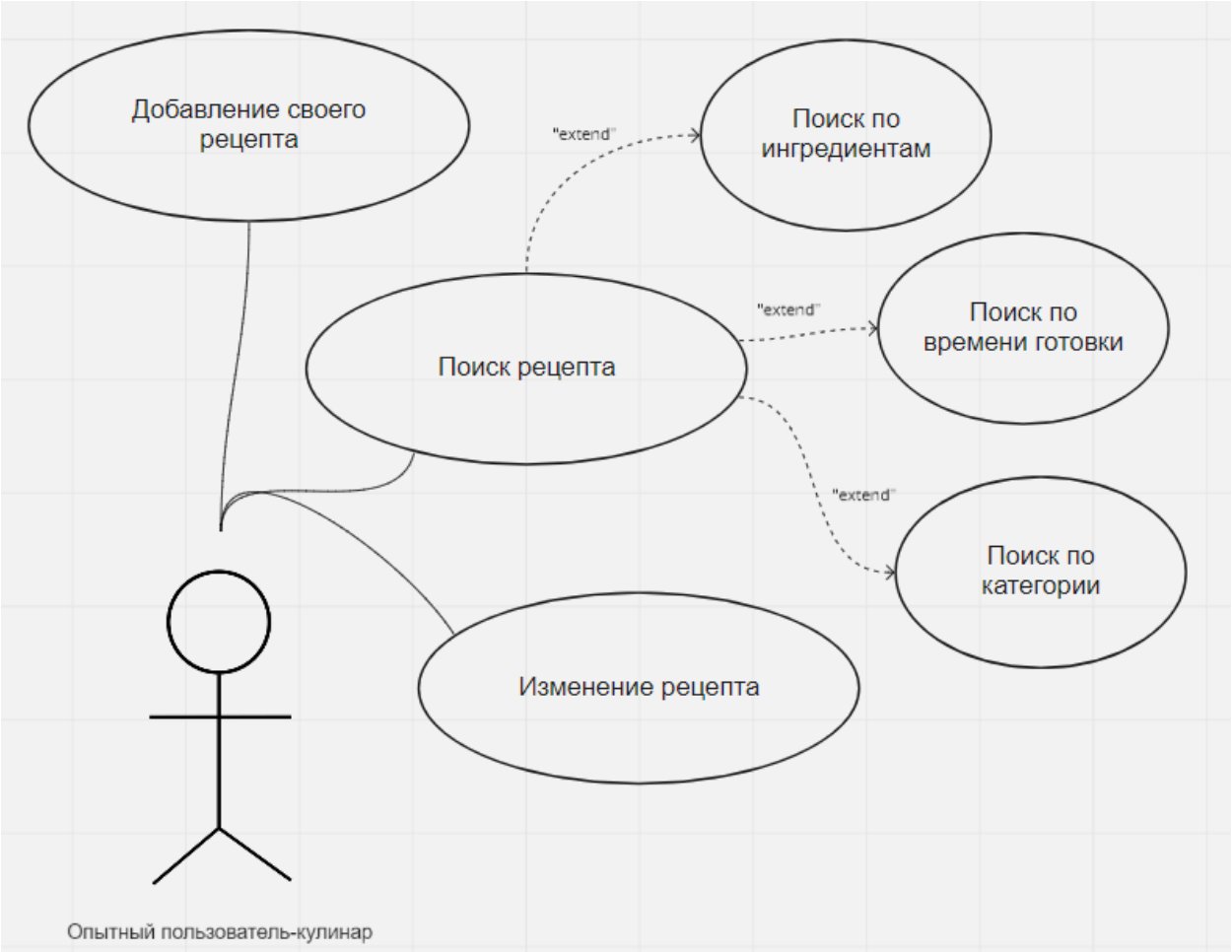

Рисунок Б.1 – Диаграмма вариантов использования для опытного пользователя