An Android Dev Project Report

On

GROCERY LIST APPLICATIONUSING KOTLIN

IN ANDROID STUDIO

SUBMITTED BY:

Thanushree H

UNDER

Application Id : SPS_APL_20220094464
SBID : SB20220232015

Virtual Internship - Android Application Development Using Kotlin

ABSTRACT

Android is an open-source operating system that runs on the linux kernel. With the advent of new mobile technologies, the mobile application industry is advancing rapidly. Consisting of several operating systems like symbian OS, iOS, blackberry, etc., Android OS is recognized as the most widely used, popular and user-friendly mobile platform. This open-source linux kernel-based operating system offers high flexibility due to its customization properties making it a dominant mobile operating system.

Android applications are developed using the java language. Google has its own Software Development Kit (SDK) which enables these java codes to control devices like mobile phones, tablets, etc. Android mobile application development provides a flexible platform for developers where they can use both java Integrated Development Environment (IDEs) and android java libraries.

Google android SDK delivers a special software stack that provides developers an easy platform to develop android applications. Moreover, developers can make use of existing java IDEs which provides flexibility to the developers. Java libraries are predominant in the process of third-party application development. Cross-platform approaches make sure that developers do not have to develop platform-dependent applications. With the help of these approaches, an application can be deployed to several platforms without the need for changes in coding. However, android is more prone to security vulnerabilities which the majority of the users do not take into account. Any android developer can upload their application on the android market which can cause a security threat to any android device. These applications do not have to go through rigorous security checks. In this report, a layered approach for android application development along with various cross-platform approaches is discussed.

ACKNOWLEDGEMENT

I consider it as a special privilege to express few word of gratitude and respect to all thosewho have guided and inspired us in completing Internship. The success depends largely on the encouragement and guideline of many other. I take this opportunity to express our gratitude to the people who have been instrumental in successful completion of the Internship.

I am highly indebted to SMARTINTERNZ (Experiential Learning & Remote Externship Platform to bring academia & industry very close for a common goal of talent creation) for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project. I would like to express my gratitude towards member of (SmartBridge) for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons for giving me such attention and time.

My thanks and appreciations also go to people who have willingly helped me out with their abilities.

Thanushree H.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Android is an open source and Linux-based **Operating System** for mobile devices such as smartphones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies.

Android Logo

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.

The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008.

On June 27, 2012, at the Google I/O conference, Google announced the next Android version, 4.1 **Jelly Bean**. Jelly Bean is an incremental update, with the primary aim of improving the user interface, both in terms of functionality and performance.

The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2.

1.1 ANDROID FEATURES AND APPLICATIONS

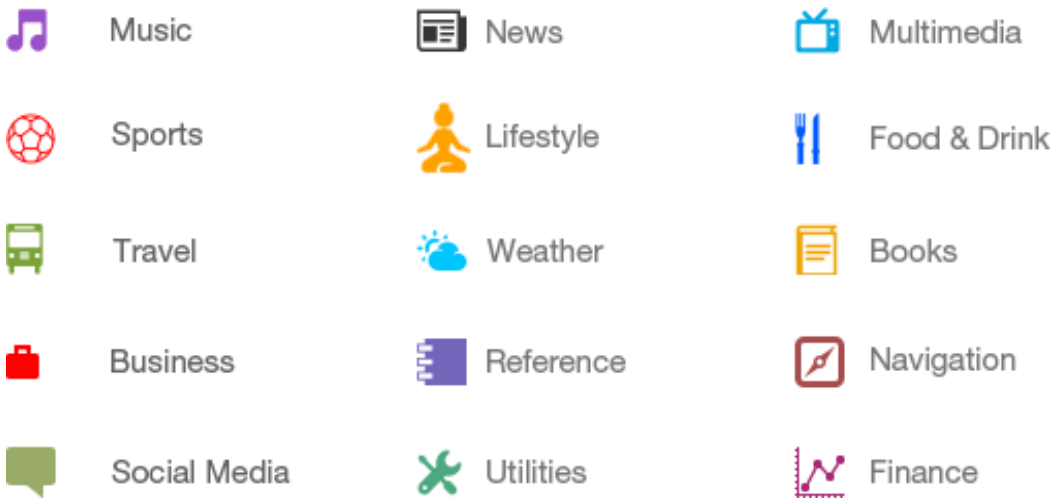
FEATURES:

- Android is a powerful operating system competing with Apple 4GS and supports great features. Few of them are listed below :
  - Beautiful UI
  - Connectivity
  - Storage
  - Messaging
  - Web browser
  - Resizable widgets
  - GCM
  - Wi-Fi Direct
  - Android Beam

APPLICATIONS:

- Android applications are usually developed in the Java language using the Android Software Development Kit. Once developed, Android applications can be packaged easily and sold out either through a store such as Google Play, SlideME,Opera Mobile Store, Mobango, F-droid and the Amazon Appstore.

CATEGORIES OF ANDROID APPLICATIONS



Android Applications

1.2 ANDROID ARCHITECTURE

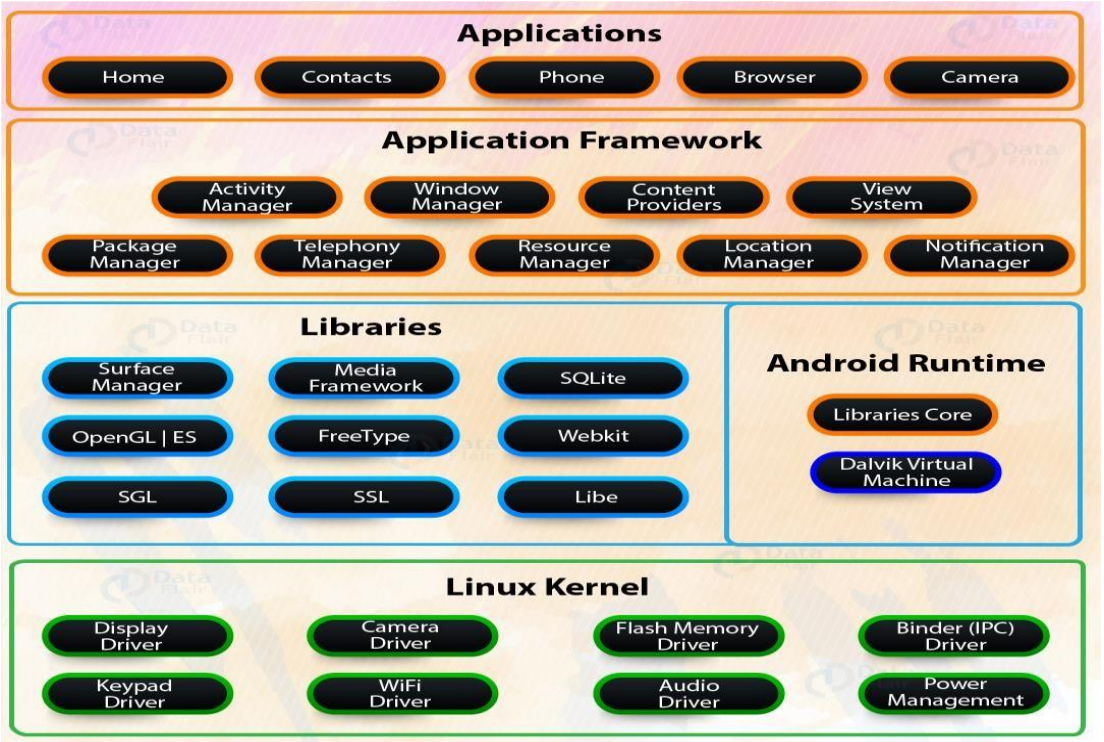


Figure 2.3 – Android Architecture

- Following are main components of android architecture those are
  - Applications
  - Android Framework
  - Android Runtime
  - Platform Libraries
  - Linux Kernel

- Linux kernel :
  - The bottom of the layers is Linux - Linux 3.6 with approximately 115 patches.

- Libraries :

  On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc. some key core Android libraries available to the Android developer is as follows –

  - android.app
  - android.content
  - android.database
  - android.opengl
  - android.os
  - android.text
  - android.view
  - android.widget
  - android.webkit

- Android Runtime:

  This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called **Dalvik Virtual Machine** which is a kind of Java Virtual Machine specially designed and optimized for Android. Application Framework: The Application Framework layer provides many higher-level services to applications in the form of Java classes.

- The Android framework includes the following key services –

  • Activity Manager

  • Content Providers

  • Resource Manager

  • Notifications Manager

  • View System

- Applications:

  You will find all the Android application at the top layer. You will write your application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, Games etc.

CHAPTER 2
ACTIVITIES PERFORMED

2.1 ANDROID APPLICATION COMPONENTS

- **Activities**
  They dictate the UI and handle the user interaction to the smart phone screen.

- **Services**
  They handle background processing associated with an application.

- **Broadcast Receivers**
  They handle communication between Android OS and applications.

- **Content Providers**
  They handle data and database management issues.

2.1.1 ACTIVITY

Activity represents a **single screen** with a user interface (UI) of an application and it will acts an entry point for users to interact with an app. From the multiple activities in android app, one activity can be marked as a main activity and that is the first screen to appear when we launch the application.

2.1.2 ANDROID ACTIVITY LIFECYCLE

Android Activity Lifecycle

2.1.3 LIFE CYCLE METHODS

All lifecycles methods must be call through to super method.

onCreate() : Called when the activity is created

onStart() : Called when an activity become visible

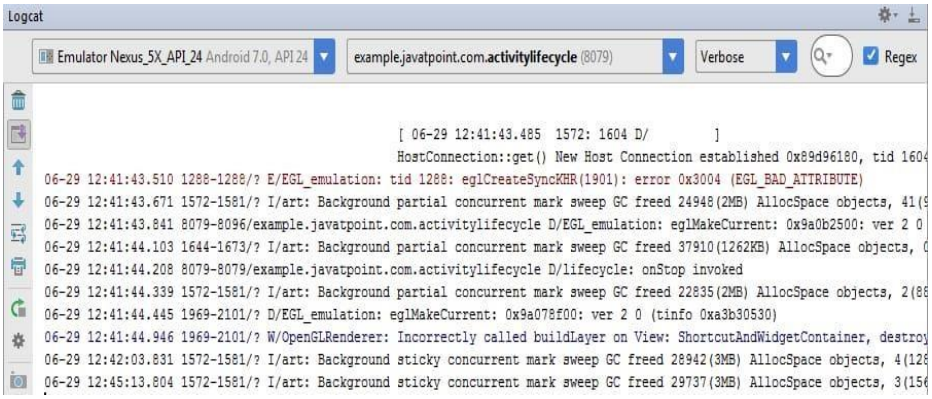
onResume() : Called when the activity is about to be ready for user interaction.

onPause() : Called when a previous activity is about to be restart

onStop() : Called when the activity becomes invisible

onRestart() : Called when an activity is about to be restarted after it has been stopped

onDestroy() : Called immediately before the activity is destroyed



Example for Activity Lifecycle

2.2 ANDROID UI LAYOUT

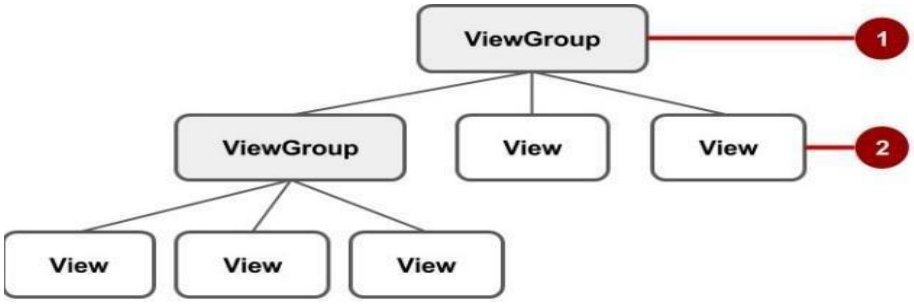
- **CONSTRAINT LAYOUT :** Is a view group which allows you to create large layouts.
- **LINEAR LAYOUT :** Is a view group that aligns all children in a single direction , vertically or horizontally.
- **RELATIVE LAYOUT :** View group that displays child views in relative positions.
- **TABLE LAYOUT :** View that groups views into rows and coloumns.
- **FRAME LAYOUT :** It is a place holder on screen that you can use to display a single view.

2.2.1 CONSTRAINT LAYOUT

Constraint Layout is a ViewGroup (i.e. a view that holds other views) which allows you to create large and complex layouts with a flat view hierarchy. Constraint Layout is very similar to Relative Layout. A group of child views using anchor points, edges, and guidelines to control how views are positioned relative to other elements in the layout

2.2.1.1 View & ViewGroup

The user interface in an android app is made with a collection of View and View Group objects. The ViewGroup is a subclass of View and provides invisible container that hold other Views or other ViewGroups and define their layout properties.



View & View Groups

2.2.2 LINEAR LAYOUT

Linear Layout is a ViewGroup subclass which is used to render all child View instances one by one either in Horizontal direction or Vertical direction based on the **orientation property.**

2.2.3 RELATIVE LAYOUT

Relative Layout is a View Group which is used to **specify the position** of child View instances relative to each other (Child A to the left of Child B) or relative to the parent (Aligned to the top of parent).

2.2.4 TABLE LAYOUT

Table Layout is a ViewGroup subclass that is used to display the child View elements in **rows and columns**.

2.2.5 FRAME LAYOUT

Frame Layout is a ViewGroup subclass that is used to specify the position of View instances it contains on the top of each other to display only **singleView** inside the Frame Layout.

2.3 ANDROID TOAST

An **Android Toast** is a small message displayed on the screen, similar to a tool tip or other similar popup notification. A **Toast** is displayed on top of the main content of an activity, and only remains visible for a short time period.

2.4 INTENT

Intent is a messaging object which is used to request an action from another app component such as activities, services, broadcast receivers and content providers.

- **Android Intent Types:**
  - **Data:** It specifies a type of data to an intent filter.
  - **Category:** We can specify a category for intent by using addCategory()

    The above properties (Component Name, Action, Data, and Category) will represent the characteristics of an intent.

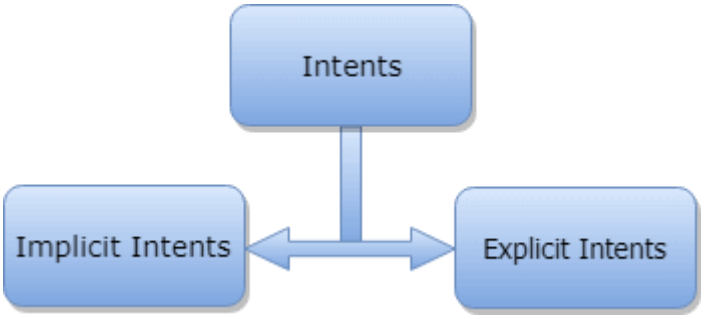


Figure 4.4 – Types of Intents

➢ **ANDROID INTENTS/FILTER**

- **IMPLICIT INTENT :** These intents do not name a target and the field for the component name is left blank. Implicit intents are often used to activate components in other applications.
- **EXPLICIT INTENT :** Explicit intent going to be connected internal world of application, suppose if you wants to connect one activity to another activity, we can do this quote by explicit intent.
- **INTENT FILTER :** OS uses filters to pinpoint the set of Activities, Services, and Broadcast receivers that can handle the Intent with help of specified set of action, categories, data scheme associated with an Intent.

2.5 ANDROIDUI CONTROLS

- **TEXT VIEW**

  Text View is a user interface control that is used to set and display the text to the user based on our requirements.
- **EDIT TEXT**

  Edit Text is a user interface control which is used to allow the user to enter or modify the text
- **AUTO COMPLETE TEXT VIEW**

  It is an TEXT VIEW editable text view which is used to show the list of suggestions based on the user typing text.
- **IMAGE BUTTON**

  Image Button is a user interface control that is used to display a button with an image and to perform an action when a user clicks or taps on it.

- **TOGGLE BUTTON**

  Toggle Button is a user interface control that is used to display ON(Checked) Or OFF(Unchecked) states as a button with a light indicator.

- **CHECK BOX**

  Check Box is a two states button that can be either checked (ON) or unchecked (OFF) and it will allow users to toggle between the two states (ON /OFF) based on the requirements.

- **RADIO GROUP**

  Radio Group is used to group one or more radio buttons into separate groups based on our requirements.

- **RADIO BUTTON**

  Radio Button is a two states button that can be either checked or unchecked and it's the same as Check Box control, except that it will allow only one option to select from the group of options.

- **PROGRESS BAR**

  Progress Bar is a user interface control that is used to indicate the progress of an operation.

- **SPINNER**

  Spinner is a view that allows a user to select one value from the list of values. The spinner in android will behave same as a dropdown list in other programming languages.

- **DATE AND TIME PICKER**

  Date Picker is a control that will allow users to select the date by a day, month and year in our application user interface.

  Time Picker is a widget for selecting the time of day, in either 24-hour or AM/PM mode.

CHAPTER 3

FINAL PROJECT

GROCERY APPLICATION

3.1 INTRODUCTION

Buying of groceries is a periodic task which has to be done on daily, weekly or monthly basis. This project helps in assisting these tasks by providing a user friendly interface to create list of items they want to buy and keep track of their purchases all within the android app. This will give users the convenience to buy groceries online in comfort of home. We are going to build a grocery application in android using Android Studio. Many times we forget to purchase things that we want to buy, after all, we can't remember all the items, so with the help of this app, you can note down your grocery items that you are going to purchase, by doing this you can't forget any items that you want to purchase. In this project, we are using (MVVM) for architectural patterns, Room for database, Recycler View and Coroutines to display the list of items.

3.1.1 OBJECTIVE

The main aim of this project is to list the items so that whenever users go to grocery stores, users will not be able to forget their items and this grocery application helps the users to tackle their day to day chaos more effortlessly.

3.1.2 PROBLEM TARGETED

It's not easy for the users to remember every item in this hectic lifestyle, they frequently can't recall their required necessity so we decided to build an app to store the items in the database for their future use. After buying the items users can delete the added items in the database.

3.1.3 PROBLEM'S PRIMARY GOALS

The goal of this project is to make an app that stores the user items in a cart and can modify and delete the added item in the list. To develop a reliable system, I have some specific goals such as:

- Develop a system such that users can add item details like product name, product Quantity, and Product Price.
- Develop a database room that is used to store the user data which already been added by the user in the cart and the user can also remove the previously added item in the cart.
- Develop a good UI design that user friendly to the user.
- Develop a good UI that is supported for all android devices.

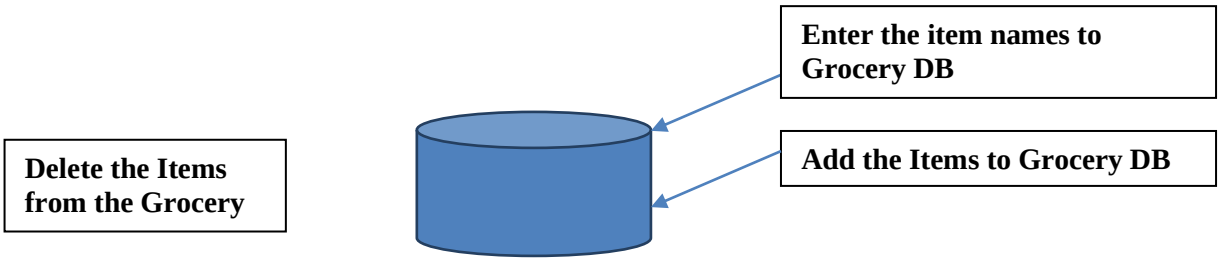
3.2 BACKGROUND & DIAGRAMS

3.2.1 BACKGROUND

The grocery cart application project will help the user or admin to store the list of items in proper sequence. User/Admin can add and remove the items in the list according to his/her will.

- UI DESIGN IN THE ANDROID PLATFORM
- ANDROID APPLICATION DEVELOPMENT
- DATABASE CONNECTION TO STORE USER DATA

3.2.2 CONTEXT DIAGRAM



3.3 TECHNICAL REQUIREMENTS

3.3.1 SOFTWARE

The Software Package is developed using Kotlin and Android Studio, basic SQL commands are used to store the database.

- Operating System: Windows 11
- Software: Kotlin and Java
- Emulator: Pixel 4 API 30

3.3.2 HARDWARE

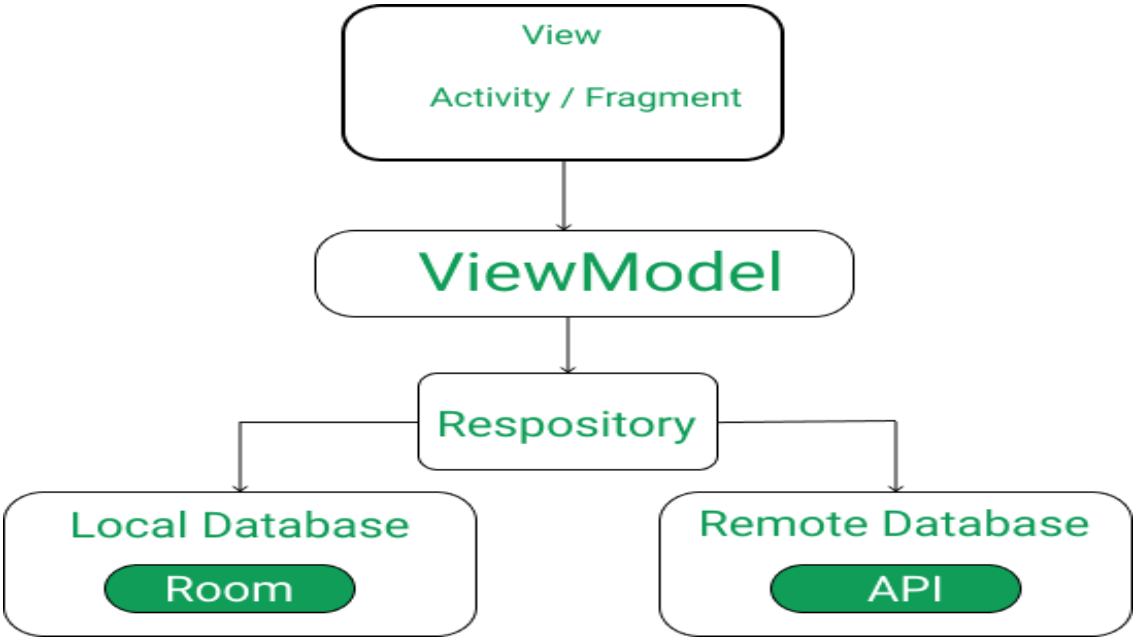
- RAM: 16GB RAM
- ROM: 20 GB ROM

3.4 IMPLEMENTATION AND DESIGNING

In this project, we are using MVVM (Model View ViewModel) for architectural patterns, **Room** for database, Coroutines and RecyclerView to display the list of items.

MVVM (Model View ViewModel)

MVVM architecture in android is used to give structure to the project's code and understand code easily. MVVM is an architectural design pattern in android. MVVM treat Activity classes and XML files as View. This design pattern completely separate UI from its logic. Here is an image to quickly understand MVVM.



ROOM DATABASE

Room persistence library is a database management library and it is used to store the data of apps like grocery item name, grocery item quantity, and grocery item price. Room is a cover layer on SQLite which helps to perform the operation on the database easily.

RecycleView

RecyclerView is a container and it is used to display the collection of data in a large amount of data set that can be scrolled very effectively by maintaining a limited number of views.

COROUTINES

Coroutines are a lightweight thread, we use coroutines to perform an operation on other threads, by this our main thread doesn't block and our app doesn't crash.

Step By Step Process

Step 1: Create a New Project

To create a new project in Android Studio please refer to How to Create/Start a New Project in Android Studio. Note that select **Kotlin** as the programming language.

Step 2: Before going to the coding section first you have to do some pre-task

Before going to the coding part first add these libraries in your gradle file and also apply the plugin as 'kotlin-kapt'. To add these library go to **Gradle Scripts >build.gradle (Module:app).**

Step 3: Implement Room Database

a) Entities class

The entities class contains all the columns in the database and it should be annotated with @Entity(tablename = "Name of table"). Entity class is a data class. And @Column info annotation is used to enter column variable name and datatype. We will also add Primary Key for auto-increment. Go to **app > java > com.example.application-name.** Right-click on **com.example.application-name** go to new and create Kotlin file/class and name the file as **GroceryEntities**. See the code below to completely understand and implement.

b) DAO Interface

The DAO is an interface in which we create all the functions that we want to implement on the database. This interface also annotated with @Dao. Now we will create a function usingsuspend function which is a coroutines function. Here we create three functions, First is the insert function to insert items in the database and annotated with @Insert, Second is for deleting items from the database annotated with @Delete and Third is for getting all items annotated with @Query. Go to the **app >java > com.example.application-name**. Right-click on **com.example.application-name** go to new and create Kotlin file/class and name the file as **GroceryDao**. See the code below to implement.

c) Database class

Database class annotated with @Database(entities = [Name of Entity class.class], version = 1) these entities are the entities array list all the data entities associating with the database and version shows the current version of the database. This database class inherits from the Room Database class. In **GroceryDatabase** class we will make an abstract method to get an instance of DAO and further use this method from the DAO instance to interact with the database. Go to the **app > java >**

com.example.application-name. Right-click on **com.example.application-name** go to new and create Kotlin file/class as **GroceryDatabase**.

Step 4: Now we will implement the Architectural Structure in the App

a) Repository Class

The repository is one of the design structures. The repository class gives the data to the ViewModel class and then the ViewModel class uses that data for Views. The repository will choose the appropriate data locally or on the network. Here in our Grocery Repository class data fetch locally from the Room database. We will add constructor value by creating an instance of the database and stored in the db variable in the Grocery Repository class. Go to the **app > java > com.example.application-name**. Right-click on **com.example.application-name** go to new and create Kotlin file/class as **GroceryRepository**.Go to **app > java > com.example.application-name**. Right-click on **com.example.application-name** go to new and create a new Package called **UI** and then right-click on UI package and create a Kotlin file/class.

b) ViewModel Class

ViewModel class used as an interface between View and Data. Grocery View Model class inherit from View Model class and we will pass constructor value by creating instance variable of Repository class and stored in repository variable. As we pass the constructor in View Model we have to create another class which is a Factory View Model class. Go to **app > java > com.example.application-name > UI**. Right-click on the UI package and create a Kotlin file/class and name the file as **GroceryViewModel**.

c) FactoryViewModel Class

We will inherit the Grocery ViewModel Factory class from ViewModelProvider.NewInstanceFactory and again pass constructor value by creating instance variable of GroceryRepository and return GroceryViewModel (repository). Go to the **app > java > com.example.application-name > UI**. Right-click on the **UI** package and create a Kotlin file/class name it **GroceryViewModelFactory**.

Step 5: Now let's jump into the UI part

In the **activity_main.xml** file, we will add two ImageView, RecyclerView, and Button after clicking this button a **DialogBox** open and in that dialog box user can enter the item name, item quantity, and item price.

Step 6:

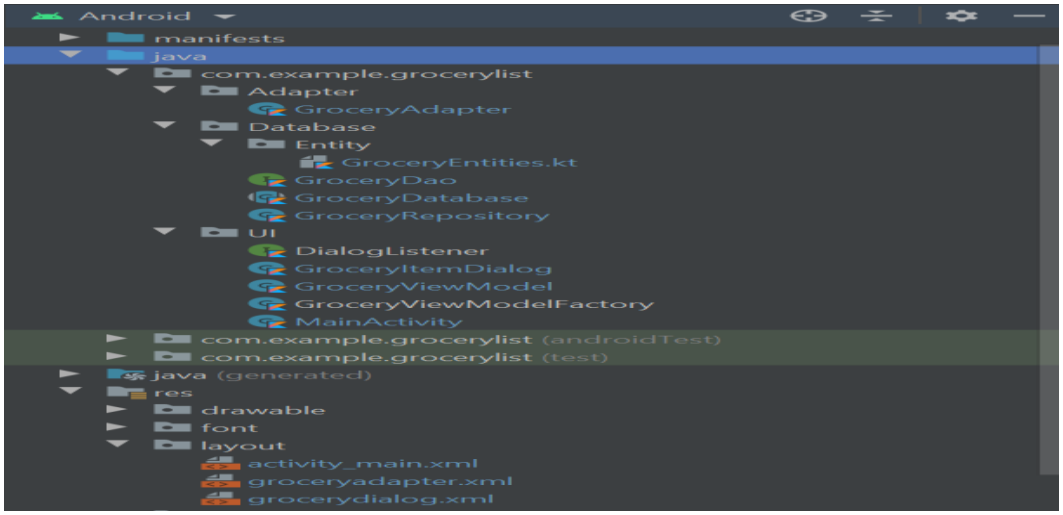
Let's implement **RecyclerView**. Now we will code the UI part of the row in the list. Go to **app > res > layout**. Right-click on layout, go to new, and then add a **Layout Resource File** and name it as **GroceryAdapter**.We will code adapter class for recycler view. In the GroceryAdapter class, we will add constructor value by storing entities class as a list in list variable and create an instance of the view model. In Grocery Adapter we will override three functions: onCreateViewHolder, getItemCount, and onbindViewHolder, we will also create an inner class called grocery view holder. Go to the **app > java > com.example.application-name**. Right-click on **com.example.application-name** go to new and create a new Package called **Adapter** and then right-click on Adapter package and create a Kotlin file/class name it **GroceryAdapter**.

Step 7:

To enter grocery item, quantity, and price from the user we have to create an interface. To implement this interface we will use DialogBox. First create UI of dialog box. In this dialog box we will add three edit text and two text view. Three edit text to enter grocery item name, quantity and price. Two text view one for save and other for cancel. After clicking the save text all data saved into the database and by clicking on the cancel text dialog box closes. Go to the **app > res > layout**. Right-click on **layout**, go to new and then add a **Layout Resource File** and name it as **GroceryDialog**.To add a clicklistener on save text we have to create an interface first in which we create a function. Go to the **app > java > com.example.application-name > UI.** Right-click on the **UI** package and create a Kotlin file/class and create an **interface** name it as **DialogListener**.

Step 8:

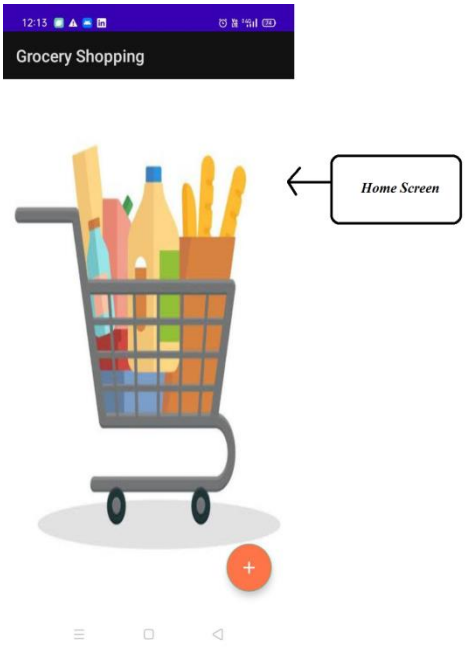
In this final step we will code in our **MainActivity**. In our **MainActivity**, we have to set up the recycler view and add click listener on add button to open the dialog box.

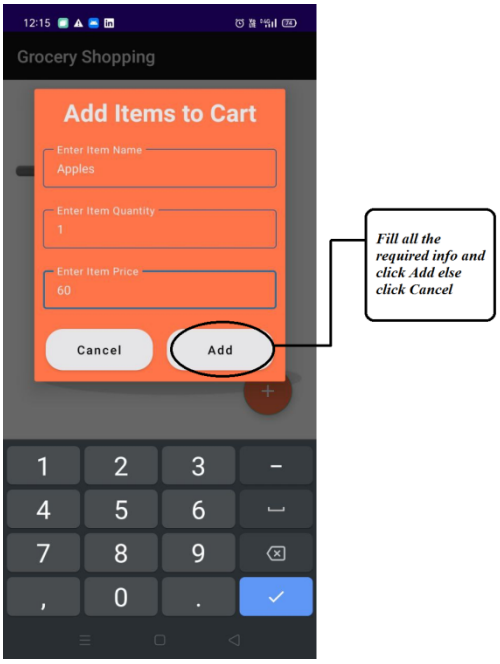


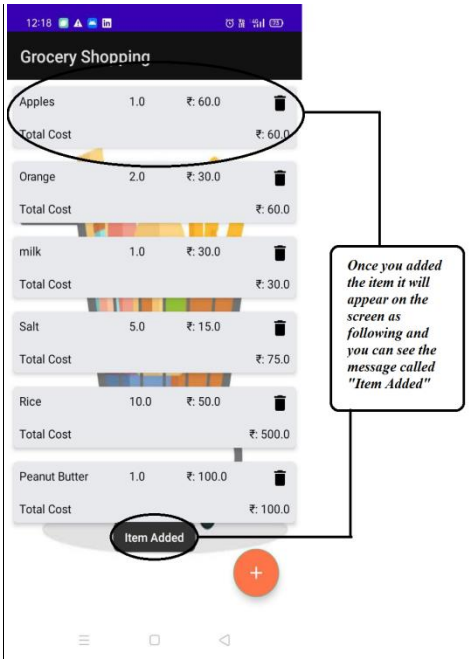
Complete Project Structure

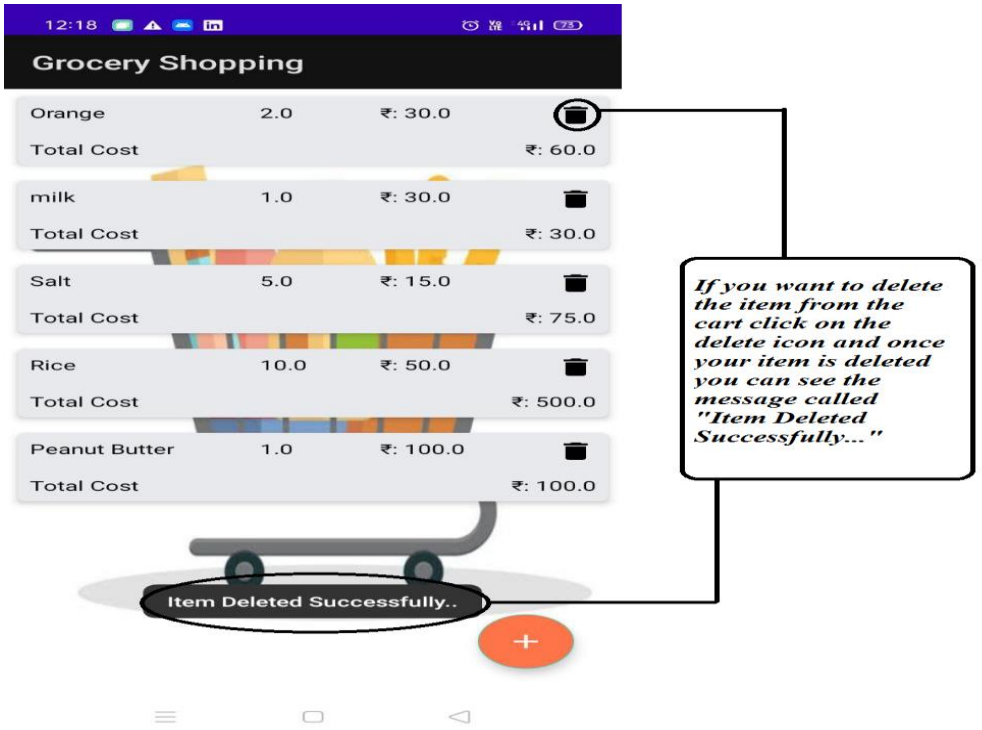
3.5 CONCLUSION AND FUTURE SCOPE

3.5.1 EXPECTED OUTPUT









3.5.2 CONCLUSION

This grocery application will help to store the list of data items include name of item, price and quantity required. Admins store his/her data in the list, the grocery application very helpful to users.

3.5.3 FUTURE SCOPE

This application helps to store the list of items by Admin. In Future we can also add scheduled addition of items according to requirement of user.
The Features are:

- Add User Panel
- Add Admin Panel
- Provide Login Authentication
- Add Image to user Product and Rating

CHAPTER 6

REFLECTION NOTES

I thoroughly enjoyed my internship and had a very valuable experience under my belt. I know this will help when looking for jobs and needing references.

I know that practical experience is the best, and internships give students that hands-on experience they need. I feel that quality internships are essential to develop key skills that we can't get in a classroom. Skills such as multitasking, communicating, learning to deal with diversity, and dealing with deadlines are different when you are working for someone else, not yourself like everyone do it college. Internships are also a great way to network with people in the industry. Our mentor and co-workers were great about giving us contacts and referring us to open positions in the industry.

I have learned that stressing over little things will not get us anywhere. I have learned to work well as a team and that without my counter parts the work would not get done. Another aspect that i learned throughout the internship is to never be afraid to ask lots of questions. By asking questions we get answers.

CHAPTER 7

CONCLUSION

Overall, internship was really good program and i surely recommend to my fellow friends. It helps to enhance and develop our skills, abilities and knowledge. It was a good experience and memories as not only we have gained experience, but also new friends and knowledge. SmartBridge is a good place to do internship since it provides numerous benefits and advantages. The treatment by the company was just equitable and professional. I have learned from different units and people. I am grateful and thankful to everyone in the company for the tutoring. They also helped me to handle some of my weaknesses and provided guidance whenever we were in need. I encourage students irrespective to there background to grab the opportunity to do internship as it will help you to identify your strength, abilities, weaknesses and more.