

ANDROID APPLICATION DEVELOPMENT USING KOTLIN



GROCERY LISTER APP

NIKITHA P

SPS_APL_20220106112

1 INTRODUCTION

1.1 Overview

Whenever we go to shop something one of the things, we commonly do is make a list of items we intend to buy as it is common to forget something if we do not. But an issue with paper lists is that they can be spoiled, torn or lost easily. Also, we need to carry pen to mark what is bought already. This is both cumbersome and a waste of resources.

1.2 Purpose

This is a listing app that tries to resolve this problem. You can enter what you want to buy, in what quantity and its price. It also computes the total price that item will cost. This allows us to reduce clutter as we only need to conduct phone with us to have the list with us.

2 LITERATURE SURVEY

2.1 Existing problem

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. 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.

2.2 Proposed solution

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:

- i. Develop a system such that users can add item details like product name, product Quantity, and Product Price.
- ii. 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

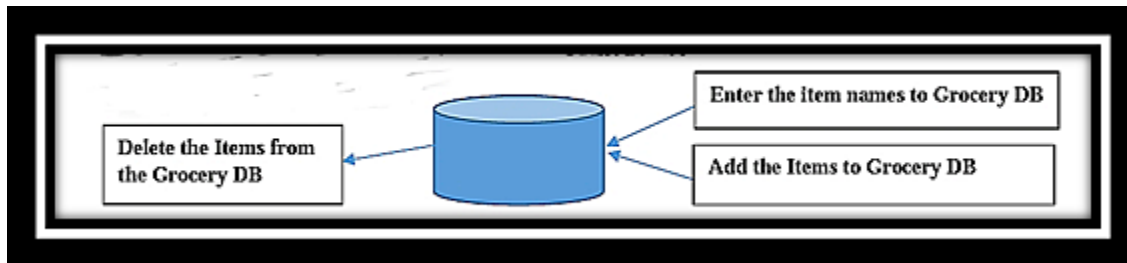
- cart.
- iii. Develop a good UI design that user friendly to the user.
- iv. Develop a good UI that is supported for all android devices.

3 THEORITICAL ANALYSIS

3.1 Block diagram Diagrammatic overview of the project.

The grocery cart application project will help the user or admin to store the list of items in proper sequence.

- i. User/Admin can add and remove the items in the list according to his/her will.
- ii. UI DESIGN IN THE ANDROID PLATFORM
- iii. ANDROID APPLICATION DEVELOPMENT
- iv. DATABASE CONNECTION TO STORE USER DATA



3.2 Hardware / Software designing Hardware and software requirements of the project

Software : Android studio

Emulator: Pixel 4 API 30

Operating System: Windows 11

Programming knowledge : Kotlin,basic SQL commands

Hardware:

ROM: 20 GB ROM

ROM: 20 GB ROM

4 EXPERIMENTAL INVESTIGATIONS

In this project, we are using MVVM (Model View ViewModel) for architectural patterns, Room for database, Co-routines 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.

RecyclerView

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. **Co-routines**

Co-routines are a lightweight thread, we use co-routines 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 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 (table name = "Name of table"). Entity class is a data class. And @Column information annotation is used to enter column variable name and datatype. We will also add PrimaryKey for auto-increment. Go to app > java > com.example.application-name. Right-click on com.example.applicationname go to new and create Kotlin file/class and name the file as GroceryEntities. See the code below to understand completely 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 using suspend function which is a co-routines 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.applicationname. 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.applicationname. Right-click on com.example.application-name go to new and create Kotlin file/class as GroceryDatabase.

Step 4: Implementing 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.applicationname 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 inherits 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.applicationname > 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 opens 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 RecyclerView. 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

on onBindViewHolder, we will also create an inner class called grocery view holder. Go to the app > java > com.example.applicationname. Right-click on com.example.applicationname 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.

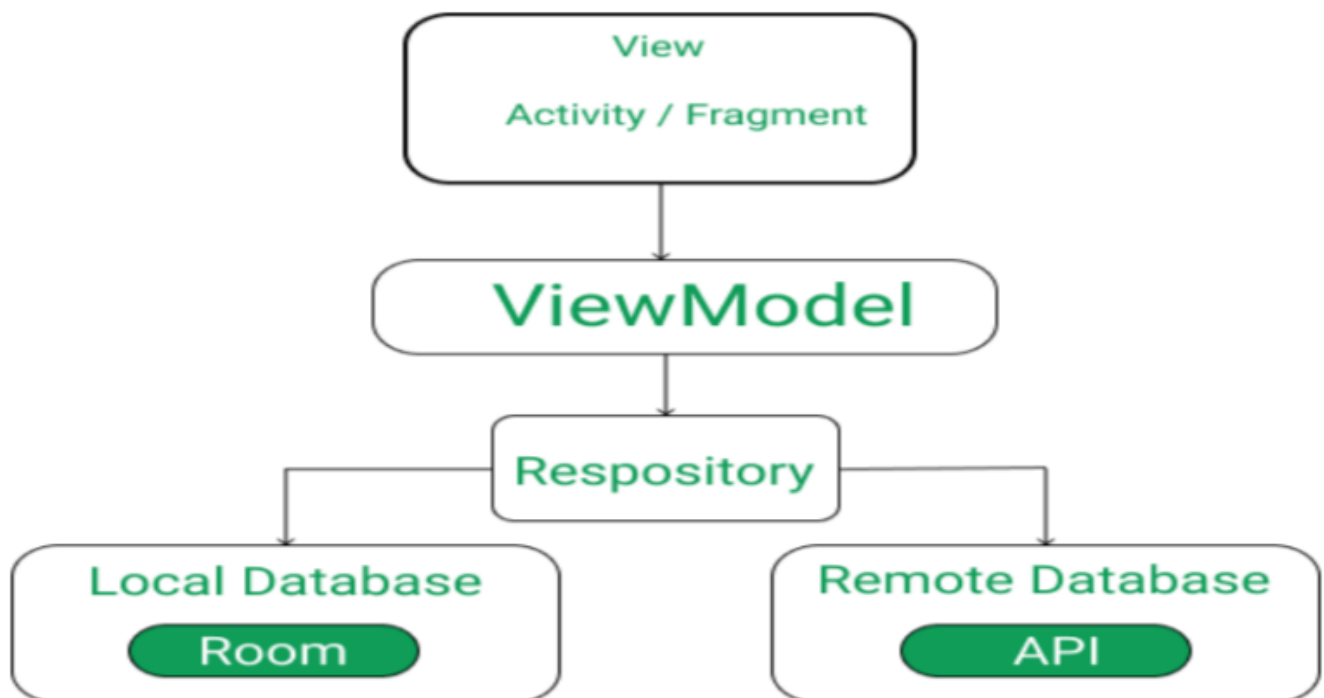
Step 7: To enter grocery item, quantity, and price from the user we have to create an interface.

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 click-listener on save text we have to create an interface first in which we create a function. Go to the app > java > com.example.applicationname > UI. Right-click on the UI package and create a Kotlin file/class and create an interface name it as DialogListener.

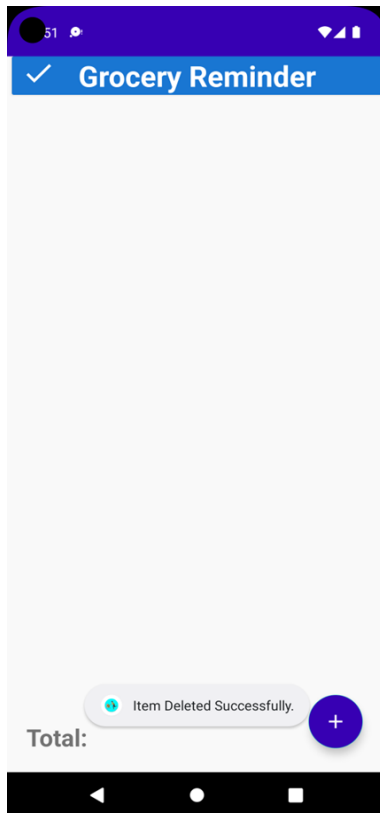
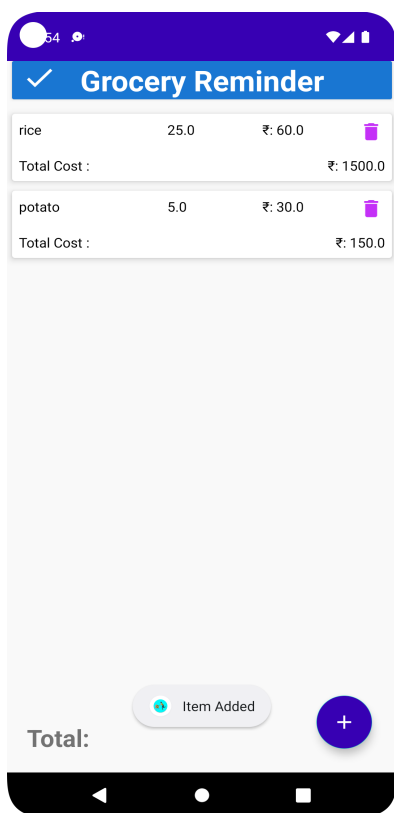
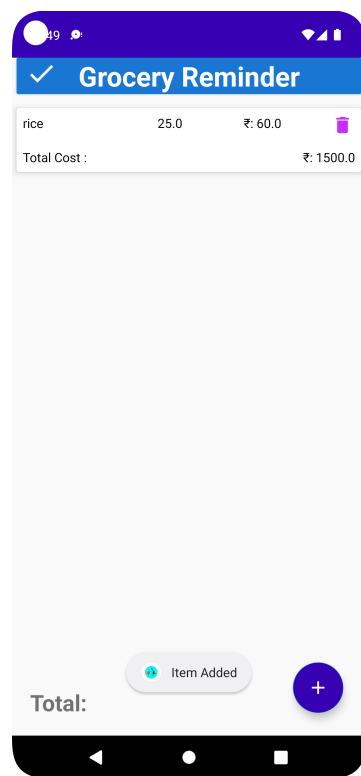
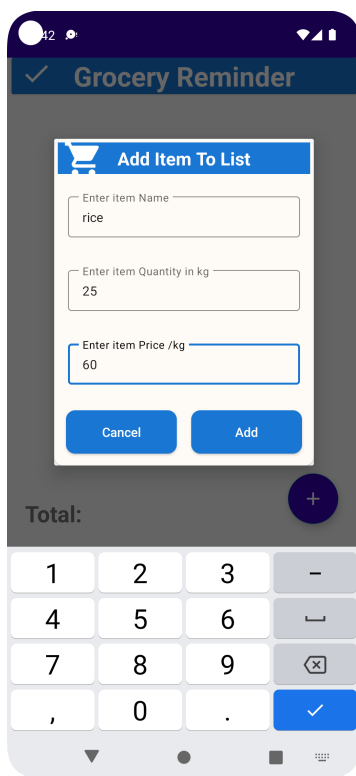
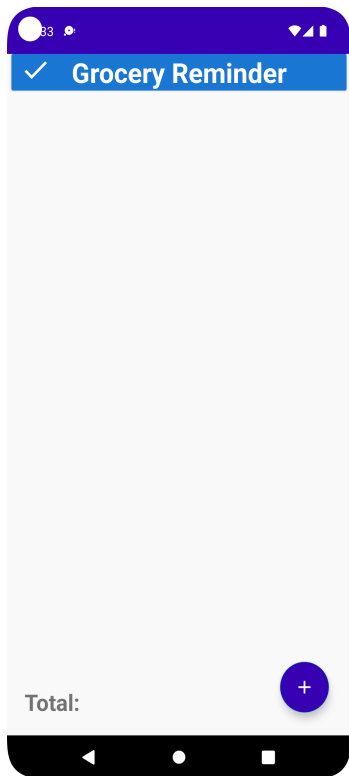
Step 8:

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.

5 FLOWCHART



6 RESULT



7 ADVANTAGES

- i. Easier to maintain item list compared to this form compared to pen-paper
- ii. Easier to tally money spent on each item if we are buying multiple of it as it tells total cost of item for the quantity.

DISADVANTAGES

- i. Currently only support single user's single list.
- ii. No authentication so anyone can access the list if they have access to list.
- iii. No categorization or history of past data.

8 APPLICATIONS

The app can be used to solve daily listing problem it maintains the grocery inventory.

9 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.

10 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:

- i. Add User Panel .
- ii. Add Admin Panel .
- iii. Provide Login Authentication .
- iv. Add Image to user Product and Rating.
- v. Provide data syncing between devices.
- vi. History to analyze spending habits .
- vii. Search and finding price for items added on the internet to suggest lower prices and saving money.
- viii. Note field to add some instructions to be followed while shopping.

11 BIBILOGRAPHY

Android studio download link :

<https://www.geeksforgeeks.org/guide-to-install-and-set-up-android-studio/>

Youtube video reference for building the app :

https://youtu.be/vdcLb_Y71lc

APPENDIX A.

Google drive project link :

<https://drive.google.com/drive/folders/1yMszy8PfAsR4gc0mWMJmilVqrxOdegYc?usp=sharing>

Google drive project demo link :

<https://drive.google.com/file/d/1hQGd11oXlsDg8v00XNnCR1yWAIUfAZqA/view?usp=sharing>

Github link :

<https://github.com/smartinternz02/SPSGP-96086-Virtual-Internship---Android-Application-Development-Using-Kotlin>

Google developer profile link :

<https://developers.google.com/profile/u/116631601124774768194>