GROCERY LIST APPLICATION

SUBMITTED BY:

NADIMPALLI KAVYA DEEPIKA

SB20220202848

UNDER

Virtual Internship - Android Application Development Using Kotlin

SPS_APL_20220062867

Project Title

Build A GroceryAndroid App – Project

Problem Statement

As we can't remember everything, users frequently forget to buy the things they want to buy.However, with the assistance of this app,you can make a listof the groceries you intendto buy so that you don't forgetanything.

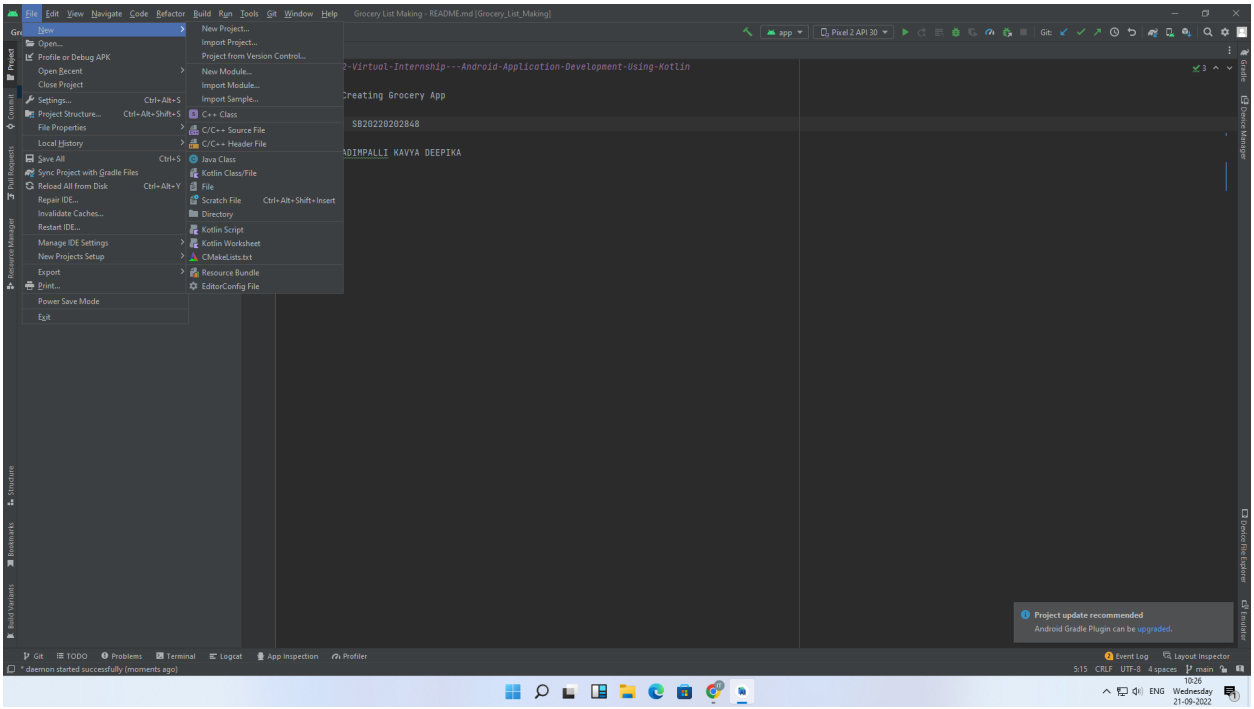
Introduction

In this project, we are usingMVVM (Model View ViewModel) for architectural patterns, Room for database, Coroutines and RecyclerView to display the listof items

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. Now we will create a function using suspend 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

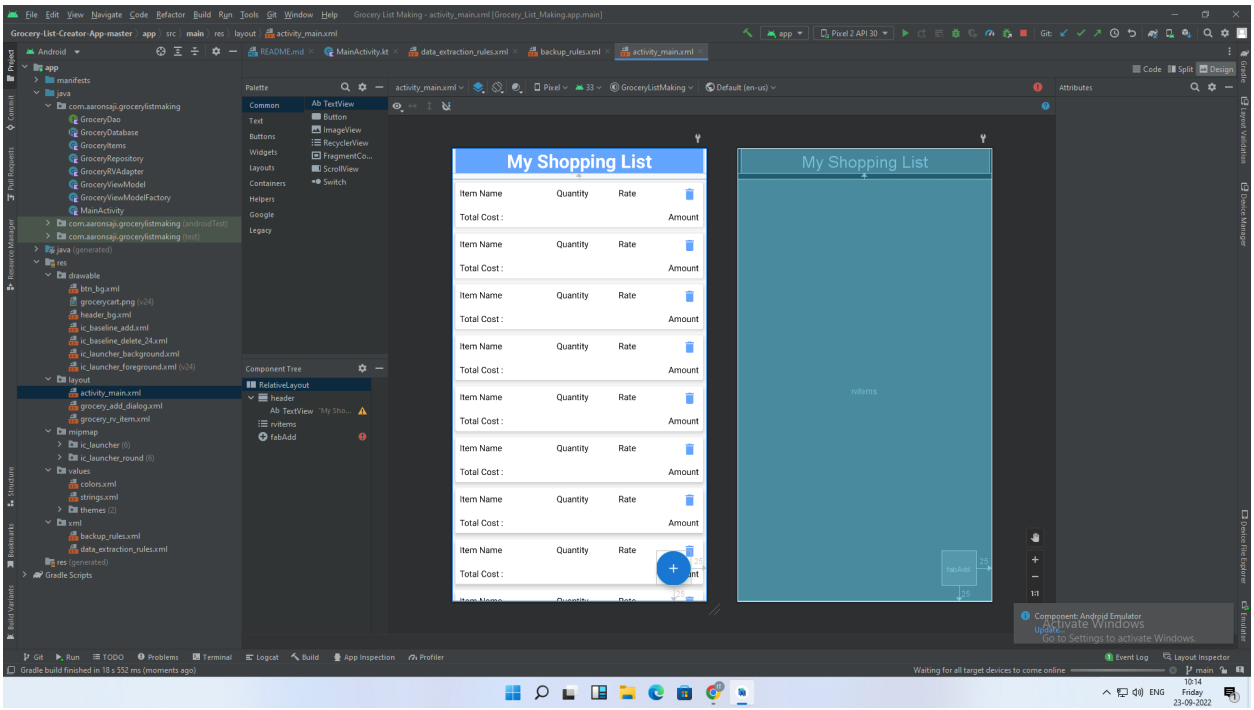
to **app > java > com.example.application-name > UI**. Right-click on the UI package and create a Kotlin file/class and name the file as **GroceryViewModel**.

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: 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:

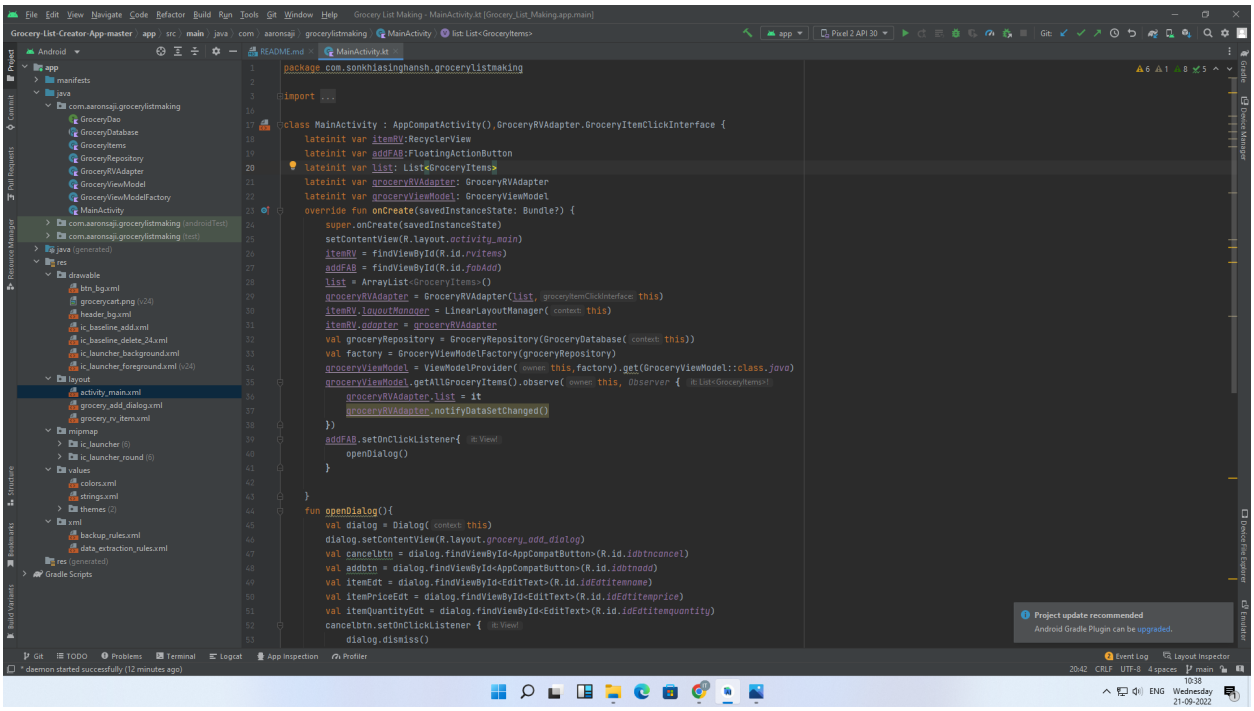
Let's implement **RecyclerView**. Now we will code the UI part of the row in the list. Go to **app > res > layout**. Right-click on layout, go to new, and then add a **Layout Resource File** and name it as **GroceryAdapter**.We will code adapter class for recycler view. In the GroceryAdapter class, we 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.



Technologies-Used

1. MVVM (ModelView ViewModel)

MVVM architecture in android is used to give structureto the project's code and understand code easily. MVVM is an architectural design pattern in android.MVVM treat Activityclasses and XML files as View. This design pattern separates UI from its logic. Here is an image to

quickly understand MVVM.

a. ROOM Database

Room persistence library is a database management library and it is used to store the data of apps like groceryitem name, groceryitem quantity, and grocery itemprice. Room is a cover layer on SQLite which helps to perform the operationon the databaseeasily.

b. RecycleView

RecyclerView is a containerand it is used to display the collection of data in a large amount of data set that can be scrolledvery effectively by maintaining alimited numberof views.

c. Coroutines

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

URL’s

✓ **GitHub URL: https://github.com/smartinternz02/SI-GuidedProject-54751-1661401499**

ACCOUNT ID’s

Demo Link: https://drive.google.com/file/d/180AdvtK423L-GK_8g61jOzwg1-vk6v04/view?usp=sharing

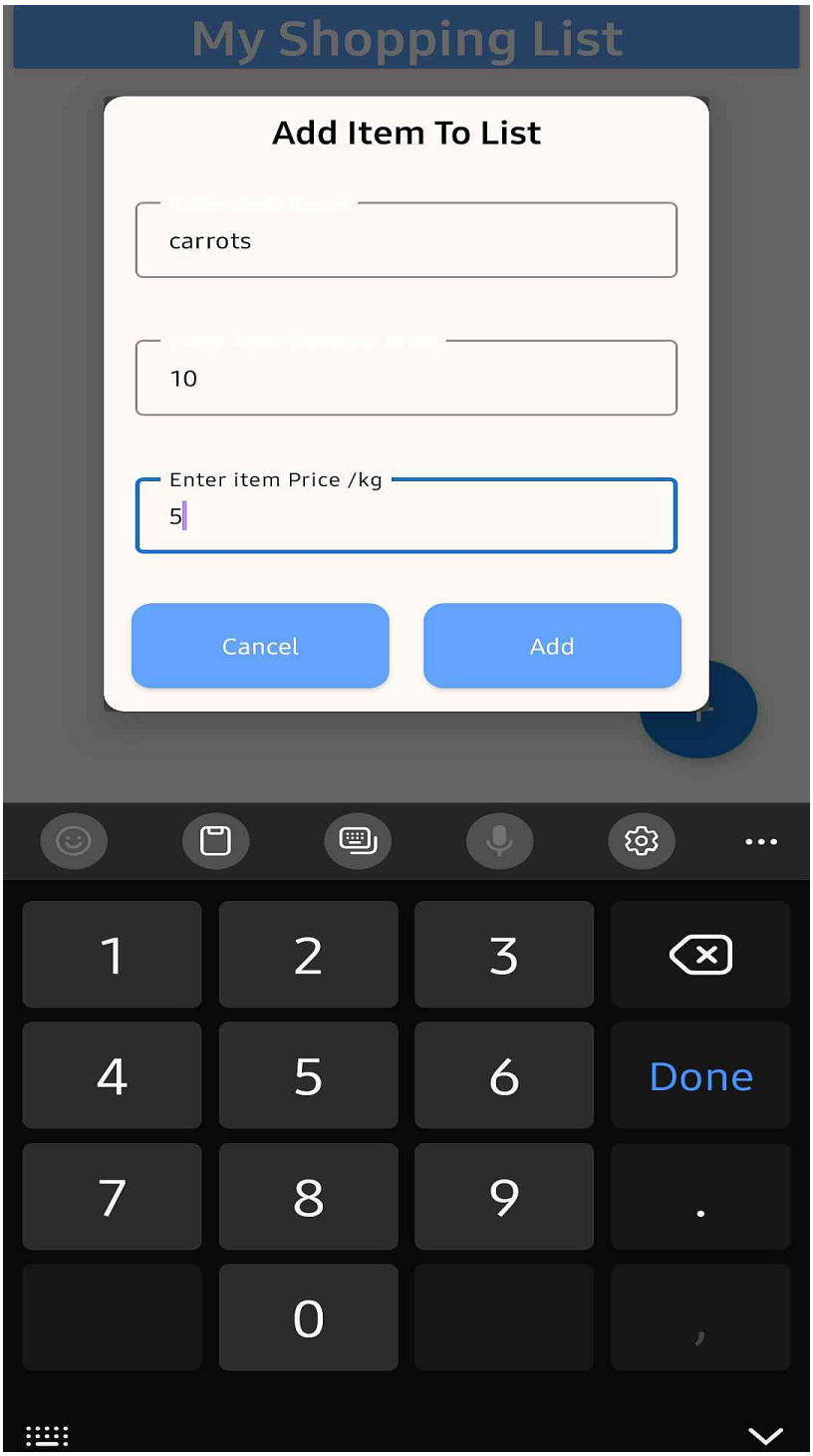
Smart Internz Registered ID: SB20220202848

Smartinternz registered email id: deepikanadimpalli163@gmail.com

Acknowledgements

I have taken much efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.
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 (Smart Internz) 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.

Screenshots of output:


My Shopping List
Add Item To List
carrots
10
Enter item Price /kg
5
Cancel
Add
1
2
3
4
5
6
Done
7
8
9
.
0

My Shopping List

carrots	10.0	₹: 5.0	
Total Cost :			₹: 50.0