SmartInternz

Android App Dev Using Kotlin

HARSH GARG

Table Of Content

Analysis About the Project, Solution Previous Findings Etc.

Appendix

A. Source Code

1. Introduction

1.1 Overview:

Grocery lister app, android based application helps user to simplify the daily chaos of remembering the stuff they need to buy. This provides active database to list items out and its price in very efficient way. In this project, we are using mvvm (model view viewmodel) for architectural patterns, room for database, coroutines and recycler view to display the list of items.

1.2 Purpose:

Grocery lister app aims to provide a comfortable experience while shopping daily need items. This project helps in assisting them in their tasks by providing a user-friendly interface to create list of items they want to buy and keep track of their purchases. This helps them to simplify the calculations by totalling the amount also making it easier to help them remember by saving it in database unless deleted. Its functionality also includes the taking note of items with quantity and amount mention.

2. Literature survey

2.1 Existing problem:

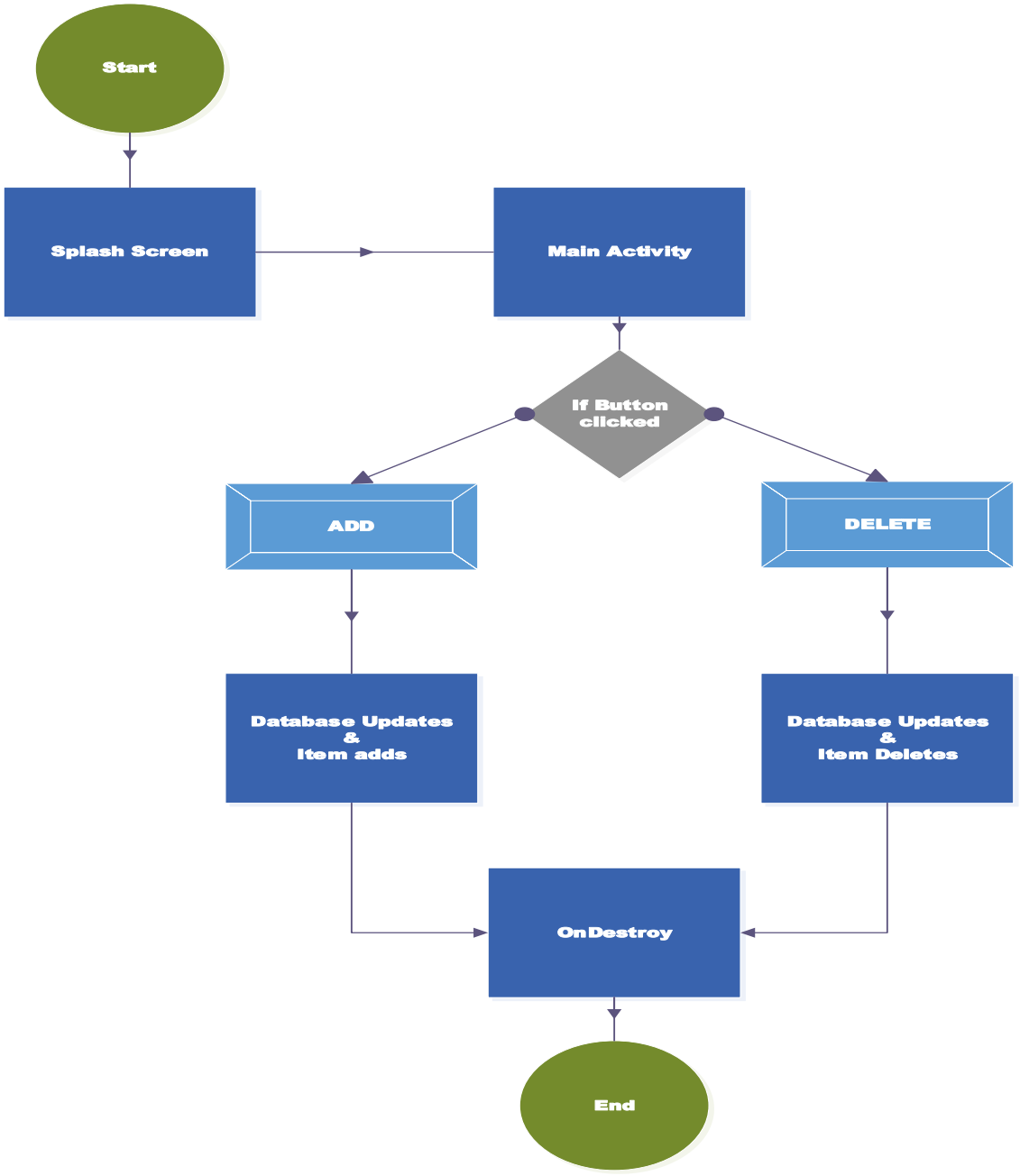
In general scenario most people go shopping while keeping notes in their head which they eventually end up forgetting about items they needed to buy, even while they keep it on paper note and carry it all along it has been seen that notes can actually get detroit while travelling due to bad weather and also it gets difficult to add items while its on your mind and you in real crowdy market. To deal with all scenarios for one and all there shall be something more reliable and durable which helps people to carry their list which can’t be damaged easily in addition to which also allows to add items far easily.

2.2Proposed solution:

For the easiness of people and making human mind less occupied over tiny stuff but also giving a new ease way to make their list of items in digital and efficient way, the grocery lister, an app with functionalities which allows adding and deleting of items in user friendly way also helps them note down price and do total with quantities provided.

3. Theoretical analysis

3.1Block diagram



On start of application the splash screen shows up which constitutes app icon, it will be shown for almost around 1.7 seconds. After the splash screen goes away main screen comes in notice consisting of two buttons namely add and delete. Add button by default is at bottom while delete button appears when item is already added. After all required operations get done and app is been closed then after just action has been taken place it leads to updating of database either deletion or addition of items from it.

3.2 Hardware / Software designing:

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 Languages:
Kotlin and Java

Emulator:
Pixel 4 API 30 3.2

Hardware:

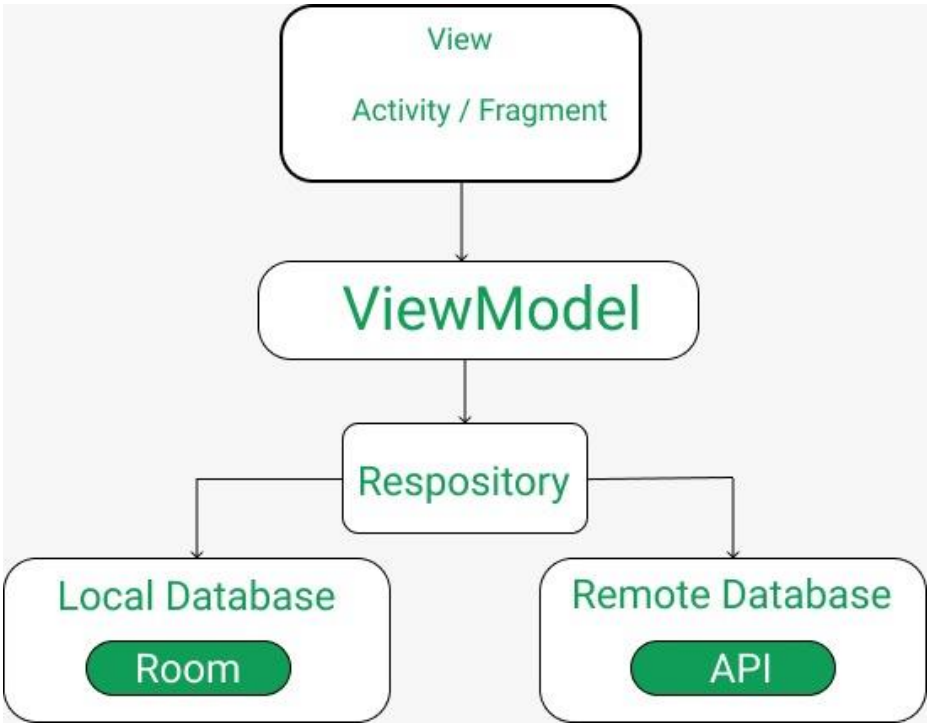
RAM: 16 GB RAM
ROM: 20 GB ROM

4. Theoretical analysis

- The project has been developed using Kotlin programming language.
- The software, Android Studio used to develop application.
- For other testing purposes the emulator has been used and also been tested upon Android phone (specific model: Samsung A8+).

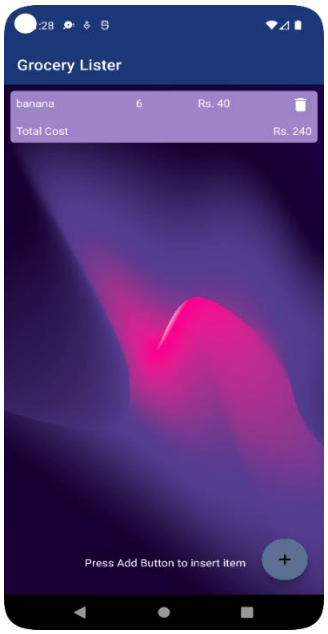
- MVVM architecture in android is used to give structure to the project's code and understand code easily. MVVM is an architectural design pattern in android. MVVM treat Activity classes and XML files as View. This design pattern separates UI from its logic. Here is an image to quickly understand MVVM.
- Room Database 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 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 are a lightweight thread, we use a coroutine to perform an operation on other threads, by this our main thread doesn't block and our app doesn't crash.

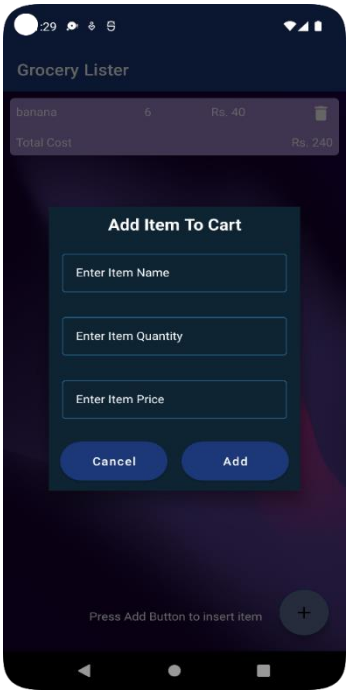
5. Flowchart

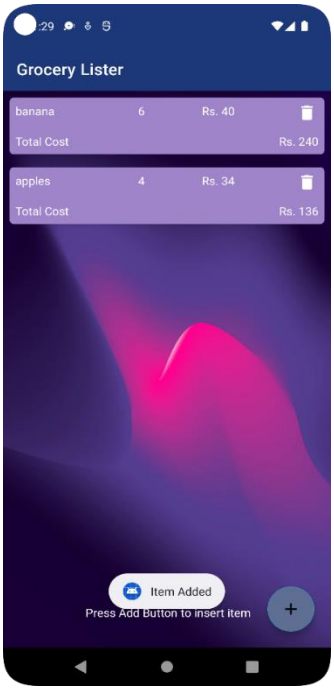


6. Result

Grocery Lister App, Android based application helps user to simplify the daily chaos of remembering the stuff they need to buy. This project helps in assisting them in their tasks by providing a user-friendly interface to create list of items they want to buy and keep track of their purchases. This helps them to simplify the calculations by totalling the amount also making it easier to help them remember by saving it in database unless deleted. Its functionality also includes the taking note of items with quantity and amount mention.







7. Advantages & Disadvantages

Advantages:

• Scope of this project includes giving users the ability to make lists of products they willing to purchase.

• Application helps user to simplify the calculations by totalling the amount while also making it easier to help them remember by saving it in database unless deleted.

• Its functionality also includes the taking note of items with quantity and amount mention.

Disadvantages:

• This project doesn't give users the ability to buy a product from a shop (online).

• It doesn't include information from real bank account to create an expense history.

8. Applications

Its application is but not limited to making items listing, managing items and price of data or items in the list. It can further be used to manage a total financial evaluation by total pricing it. Further, it can be applied to transaction history and which helps to evaluate it too.

9. Conclusion

This project helps in assisting them in their tasks by providing a user-friendly interface to create list of items they want to buy and keep track of their purchases. 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. But also can integrate the transaction management so one can actually evaluate their financial expenditure

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

11. Bibliography

https://www.geeksforgeeks.org/introduction-to-kotlin/

https://www.geeksforgeeks.org/introduction-to-android-development/#:~:text=Google%20first%20publicly%20announced%20Android,with%20the%20version%20Android%201.0

https://developer.android.com/courses/android-basics-kotlin/unit-1

https://developer.android.com/courses/android-basics-kotlin/unit-2

https://developer.android.com/courses/android-basics-kotlin/unit-3

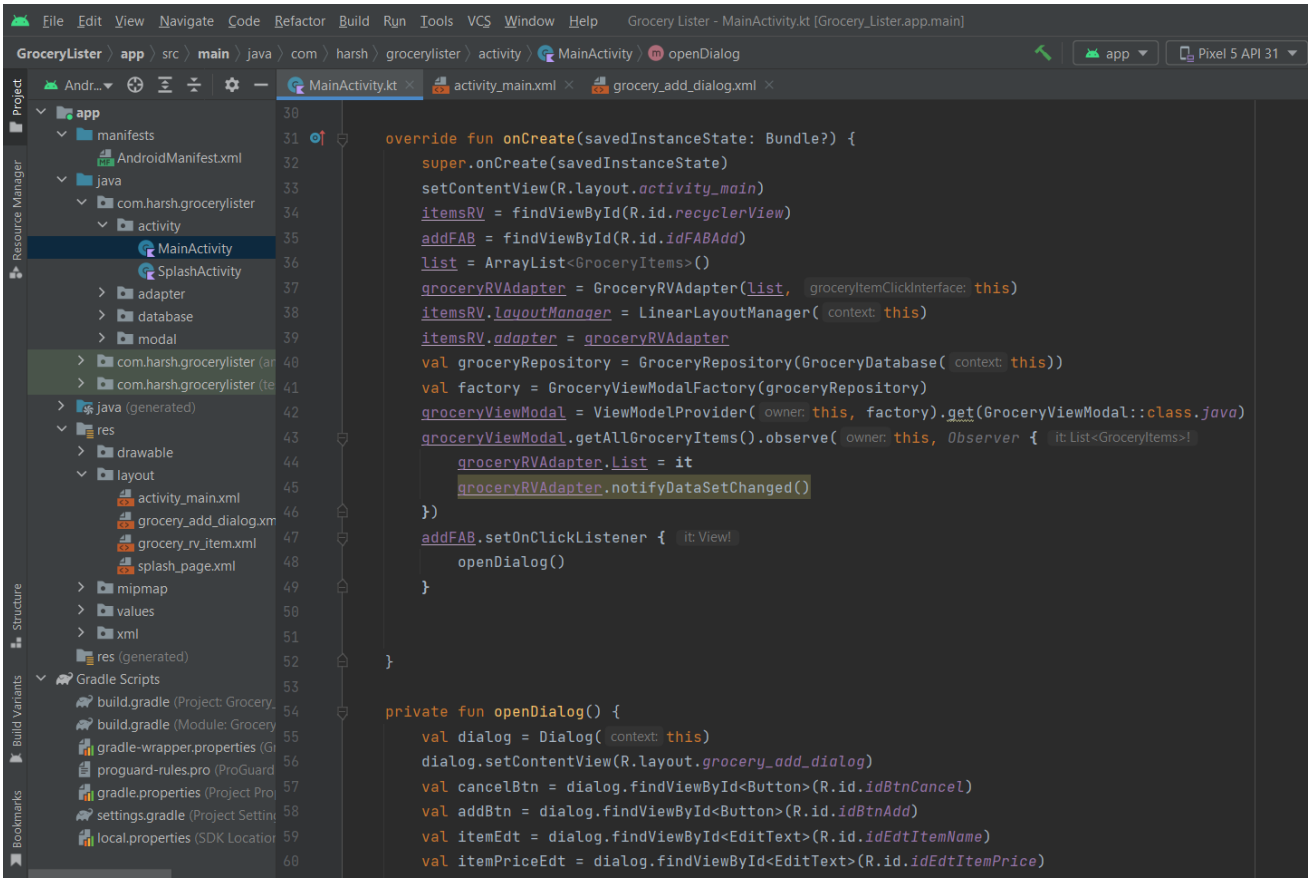
https://developer.android.com/courses/android-basics-kotlin/unit-4

https://developer.android.com/courses/android-basics-kotlin/unit-5

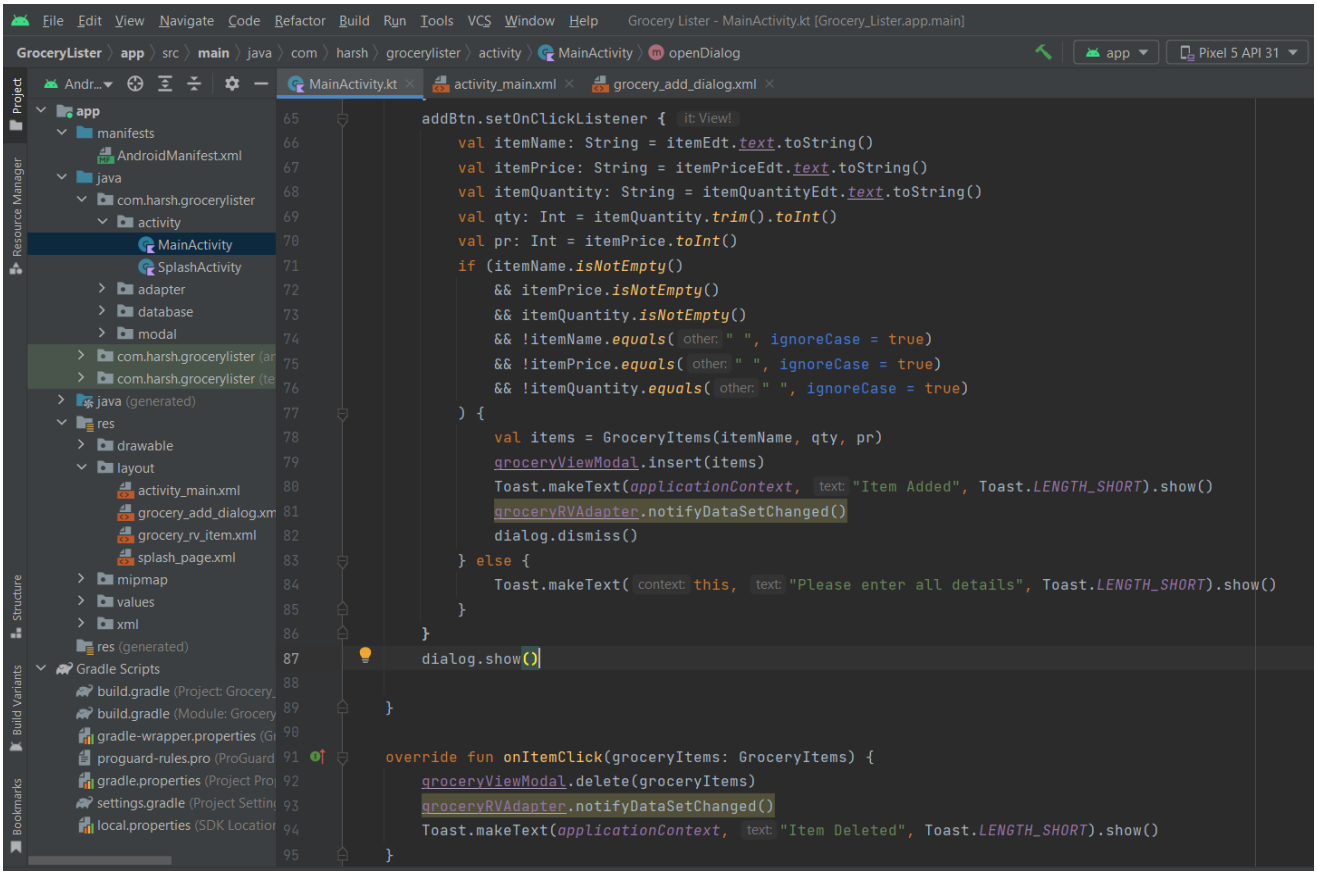
https://developer.android.com/courses/android-basics-kotlin/unit-6

APPENDIX

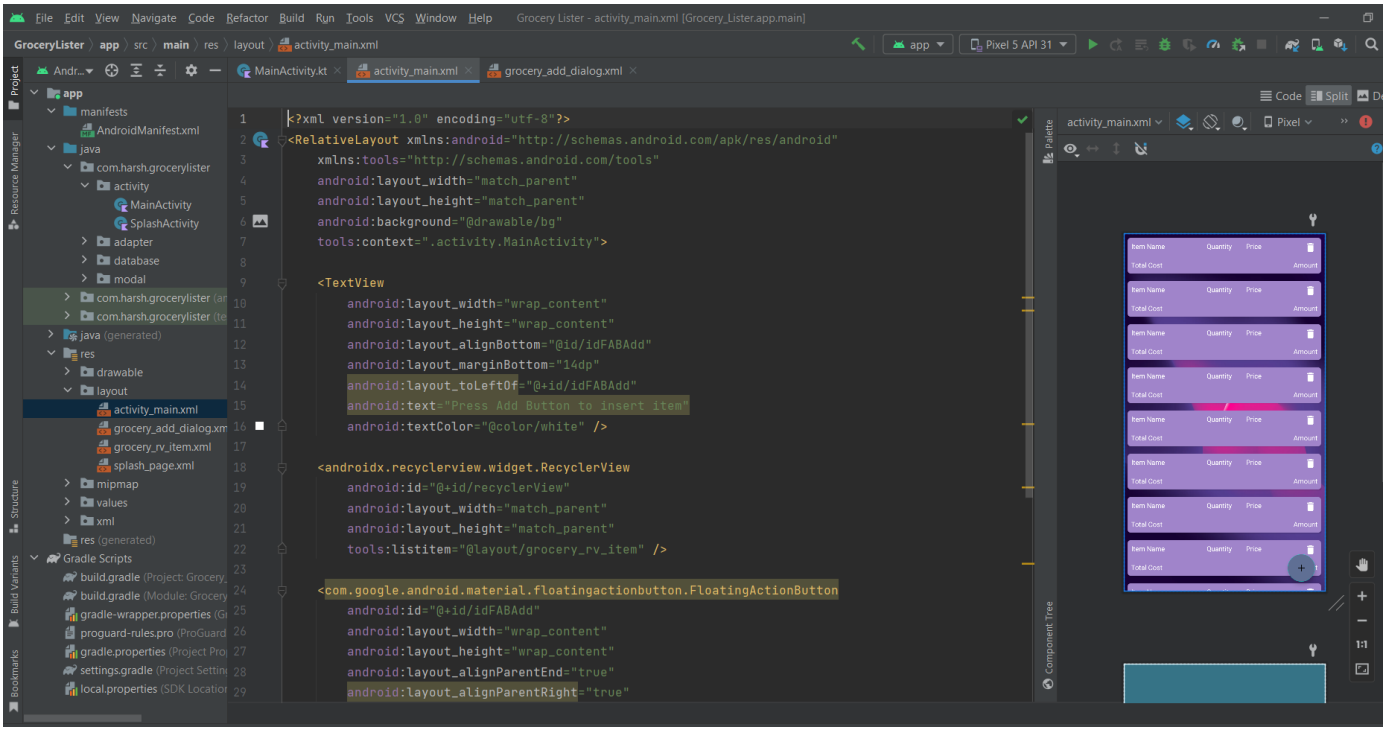
A. Source Code:

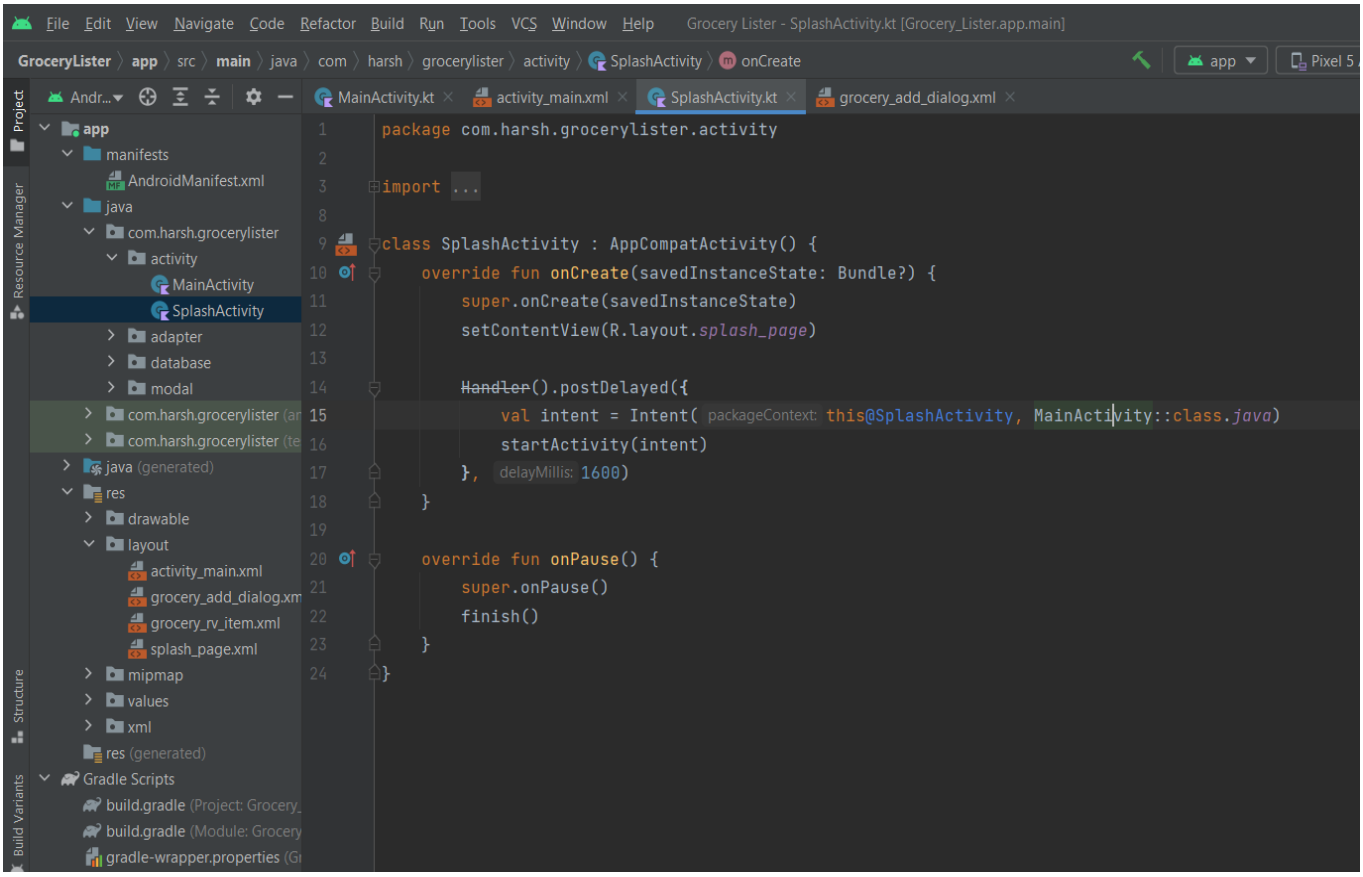


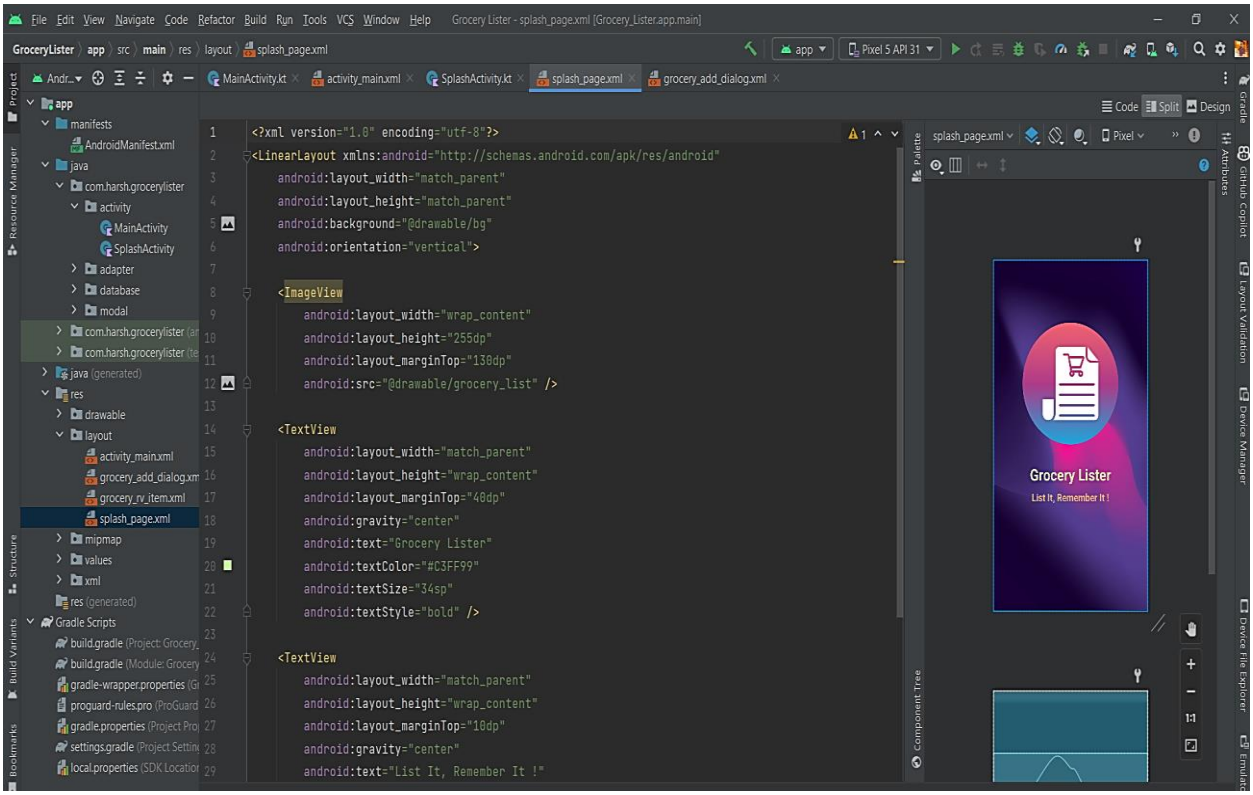
```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    itemsRV = findViewById(R.id.recyclerView)
    addFAB = findViewById(R.id.idFABAdd)
    list = ArrayList<GroceryItems>()
    groceryRVAdapter = GroceryRVAdapter(list, groceryItemClickInterface: this)
    itemsRV.layoutManager = LinearLayoutManager( context: this)
    itemsRV.adapter = groceryRVAdapter
    val groceryRepository = GroceryRepository(GroceryDatabase( context: this))
    val factory = GroceryViewModalFactory(groceryRepository)
    groceryViewModal = ViewModelProvider( owner: this, factory).get(GroceryViewModal::class.java)
    groceryViewModal.getAllGroceryItems().observe( owner: this, Observer { it: List<GroceryItems>!
        groceryRVAdapter.List = it
        groceryRVAdapter.notifyDataSetChanged()
    })
    addFAB.setOnClickListener { it: View!
        openDialog()
    }


}

private fun openDialog() {
    val dialog = Dialog( context: this)
    dialog.setContentView(R.layout.grocery_add_dialog)
    val cancelBtn = dialog.findViewById<Button>(R.id.idBtnCancel)
    val addBtn = dialog.findViewById<Button>(R.id.idBtnAdd)
    val itemEdt = dialog.findViewById<EditText>(R.id.idEdtItemName)
    val itemPriceEdt = dialog.findViewById<EditText>(R.id.idEdtItemPrice)
```

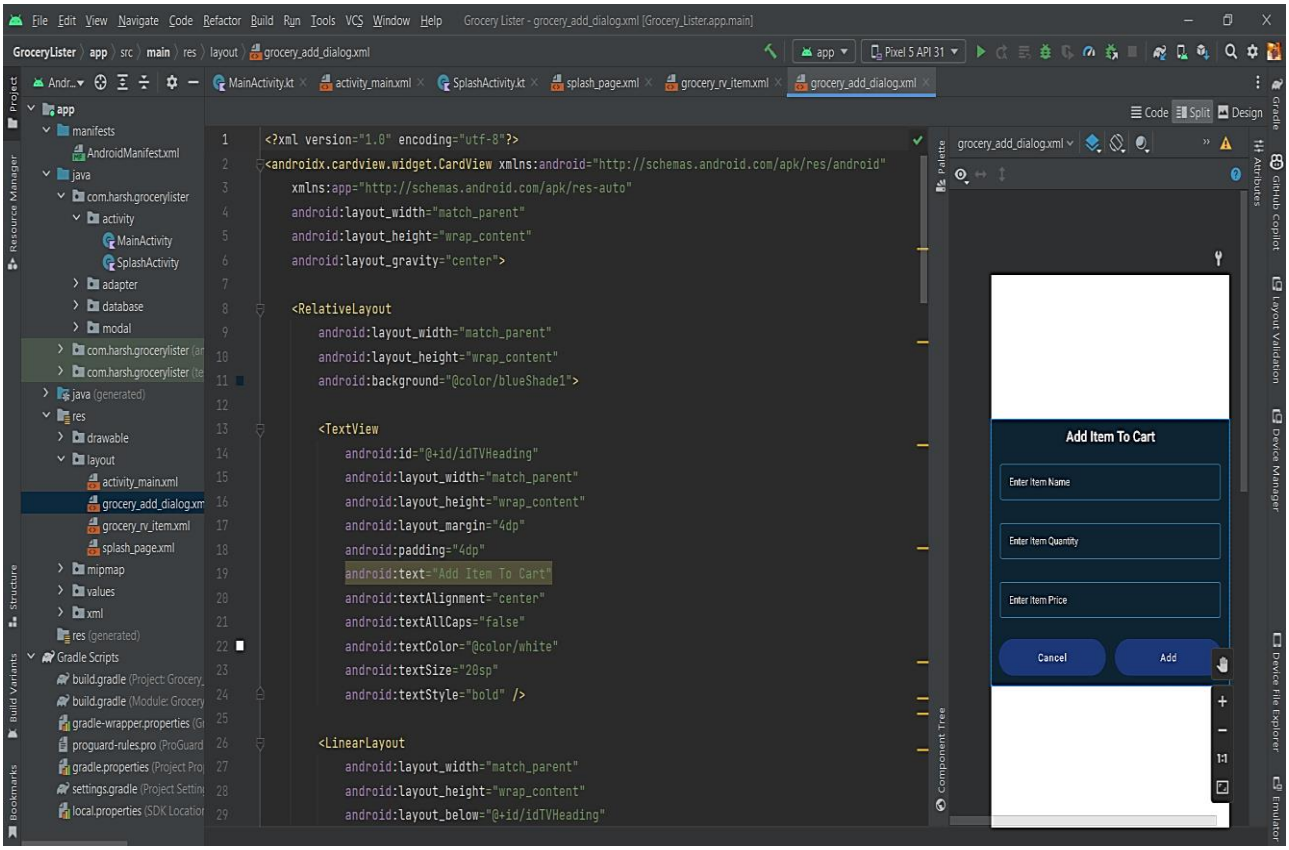


```
    addBtn.setOnClickListener { it: View!
        val itemName: String = itemEdt.text.toString()
        val itemPrice: String = itemPriceEdt.text.toString()
        val itemQuantity: String = itemQuantityEdt.text.toString()
        val qty: Int = itemQuantity.trim().toInt()
        val pr: Int = itemPrice.toInt()
        if (itemName.isNotEmpty()
            && itemPrice.isNotEmpty()
            && itemQuantity.isNotEmpty()
            && !itemName.equals( other: " ", ignoreCase = true)
            && !itemPrice.equals( other: " ", ignoreCase = true)
            && !itemQuantity.equals( other: " ", ignoreCase = true)
        ) {
            val items = GroceryItems(itemName, qty, pr)
            groceryViewModal.insert(items)
            Toast.makeText(applicationContext, text: "Item Added", Toast.LENGTH_SHORT).show()
            groceryRVAdapter.notifyDataSetChanged()
            dialog.dismiss()
        } else {
            Toast.makeText( context: this, text: "Please enter all details", Toast.LENGTH_SHORT).show()
        }
    }
    dialog.show()

}

override fun onItemClick(groceryItems: GroceryItems) {
    groceryViewModal.delete(groceryItems)
    groceryRVAdapter.notifyDataSetChanged()
    Toast.makeText(applicationContext, text: "Item Deleted", Toast.LENGTH_SHORT).show()
}
```


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
Grocery Lister - activity_main.xml [Grocery_Lister.app.main]
GroceryLister app src main res layout activity_main.xml
app
Pixel 5 API 31
MainActivity.kt
activity_main.xml
grocery_add_dialog.xml
Code Split
activity_main.xml
Pixel
Project
Resource Manager
Structure
Build Variants
Bookmarks
Palette
Component Tree
app
manifests
AndroidManifest.xml
java
com.harsh.grocerylister
activity
MainActivity
SplashActivity
adapter
database
modal
java (generated)
res
drawable
layout
activity_main.xml
grocery_add_dialog.xml
grocery_rv_item.xml
splash_page.xml
mipmap
values
xml
res (generated)
Gradle Scripts
build.gradle (Project: Grocery_
build.gradle (Module: Grocery
gradle-wrapper.properties
proguard-rules.pro
gradle.properties
settings.gradle
local.properties
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3 xmlns:tools="http://schemas.android.com/tools"
4 android:layout_width="match_parent"
5 android:layout_height="match_parent"
6 android:background="@drawable/bg"
7 tools:context=".activity.MainActivity">
8
9 <TextView
10 android:layout_width="wrap_content"
11 android:layout_height="wrap_content"
12 android:layout_alignBottom="@id/idFABAdd"
13 android:layout_marginBottom="14dp"
14 android:layout_toLeftOf="@+id/idFABAdd"
15 android:text="Press Add Button to insert item"
16 android:textColor="@color/white" />
17
18 <androidx.recyclerview.widget.RecyclerView
19 android:id="@+id/recyclerView"
20 android:layout_width="match_parent"
21 android:layout_height="match_parent"
22 tools:listitem="@layout/grocery_rv_item" />
23
24 <com.google.android.material.floatingactionbutton.FloatingActionButton
25 android:id="@+id/idFABAdd"
26 android:layout_width="wrap_content"
27 android:layout_height="wrap_content"
28 android:layout_alignParentEnd="true"
29 android:layout_alignParentRight="true"


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
Grocery Lister - SplashActivity.kt [Grocery_Lister.app.main]
GroceryLister app src main java com harsh grocerylister activity SplashActivity onCreate
app
Pixel 5
MainActivity.kt
activity_main.xml
SplashActivity.kt
grocery_add_dialog.xml
Project
Resource Manager
Structure
Build Variants
app
manifests
AndroidManifest.xml
java
com.harsh.grocerylister
activity
MainActivity
SplashActivity
adapter
database
modal
java (generated)
res
drawable
layout
activity_main.xml
grocery_rv_item.xml
splash_page.xml
mipmap
values
xml
res (generated)
Gradle Scripts
build.gradle (Project: Grocery_
build.gradle (Module: Grocery
gradle-wrapper.properties
1 package com.harsh.grocerylister.activity
2
3 import ...
8
9 class SplashActivity : AppCompatActivity() {
10 override fun onCreate(savedInstanceState: Bundle?) {
11 super.onCreate(savedInstanceState)
12 setContentView(R.layout.splash_page)
13
14 Handler().postDelayed({
15 val intent = Intent(packageContext: this@SplashActivity, MainActivity::class.java)
16 startActivity(intent)
17 }, delayMillis: 1600)
18 }
19
20 override fun onPause() {
21 super.onPause()
22 finish()
23 }
24 }


Grocery Lister - splash_page.xml [Grocery_Lister.app.main]
splash_page.xml
MainActivity.kt
activity_main.xml
SplashActivity.kt
splash_page.xml
grocery_add_dialog.xml
Code
Split
Design
Grocery Lister
List It, Remember It !


Grocery Lister - grocery_add_dialog.xml [Grocery_Lister.app.main]
grocery_add_dialog.xml
MainActivity.kt
activity_main.xml
SplashActivity.kt
splash_page.xml
grocery_rv_item.xml
grocery_add_dialog.xml
Code
Split
Design
Add Item To Cart
Enter Item Name
Enter Item Quantity
Enter Item Price
Cancel
Add

URL's:

Google Developer's Profile:

https://g.dev/harshgarg

GitHub Profile Link:

officialharshgarg (Harsh Garg) (github.com)

Drive Link of APK:

https://bit.ly/3UlSArc

Drive Link of Demo Video:

https://bit.ly/3f6lag7

GitHub Repository Link:

https://github.com/smartinternz02/SI-GuidedProject-93234-1662359121.git

SmartBridge ID or SBID:

SB20220240164

Registered Email:

harshgarg698@gmail.com