

Grocery App Report

1. Introduction

1.1 Overview

This is an android app that helps you to make a list of grocery items along with its price and quantity.

1.2 Purpose

We are humans and we cannot remember everything. We sometimes forget the things that we want to buy. However, with the assistance of this app you can make a list of grocery items you intend to buy so that you don't forget anything and also have a track of your expenditure for budget maintenance.

2. Literature Survey.

2.1 Existing Problem

Users frequently forget items to buy because of which they have to run to shops again and again which is quite a frustrating and tiring situation and if our expenses crosses out budget while shopping that could be a matter of concern.

2.2 Proposed Solution

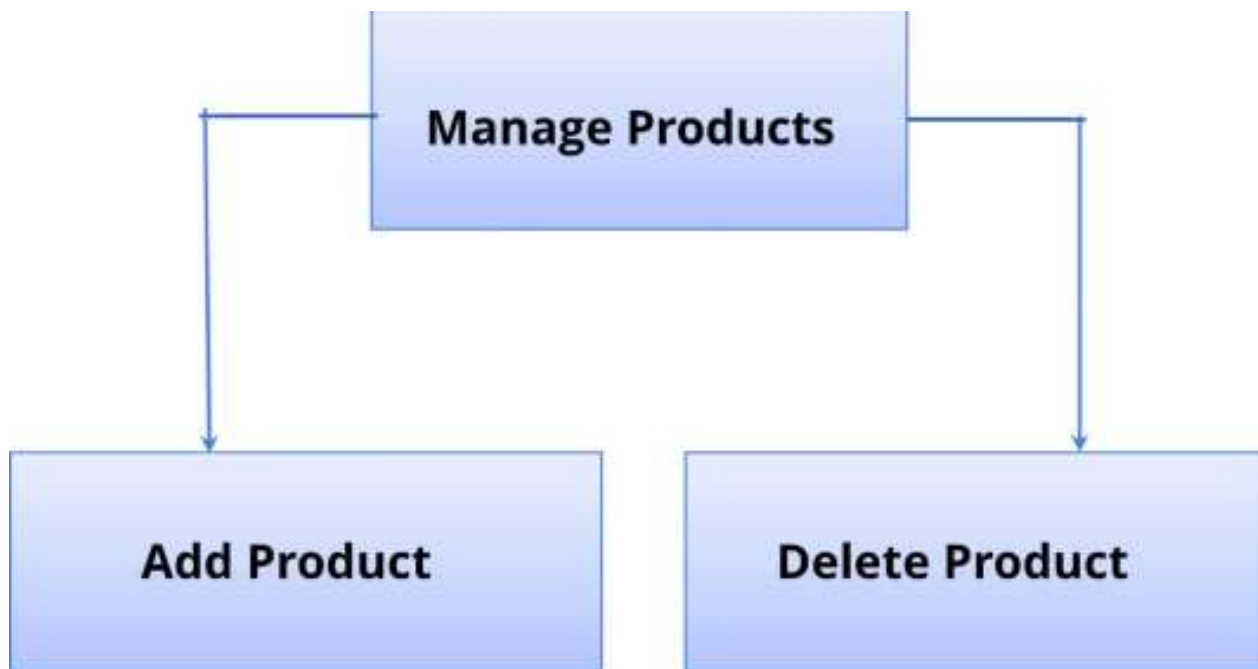
Smart

Internz

To overcome this problematic situation I built a grocery app which helps you to list down all the item that you need to buy along with its price.

3. Theoretical Analysis

3.1 Block Diagram



3.2 Hardware/Software designing

- Windows 10 OS
- Android Studio

Smart

Internz

4. Experimental Investigations

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

LiveData: A data holder class that can be observed. Always holds/caches the latest version of data, and notifies its observers when data has changed. LiveData is lifecycle aware. UI components just observe relevant data and don't stop or resume observation. LiveData automatically manages all of this since it's aware of the relevant lifecycle status changes while observing.

ViewModel: Acts as a communication center between the Repository (data) and the UI. The UI no longer needs to worry about the origin of the data. ViewModel instances survive Activity/Fragment recreation.

Repository: A class that you create that is primarily used to manage multiple data sources.

Entity: Annotated class that describes a database table when working with Room.

Room database: Simplifies database work and serves as an access point to the underlying SQLite database (hides SQLiteOpenHelper). The Room database uses the DAO to issue queries to the SQLite database.

SQLite database: On device storage. The Room persistence library creates and maintains this database for you.

DAO: Data access object. A mapping of SQL queries to functions. When you use a DAO, you call the methods, and Room takes care of the rest.

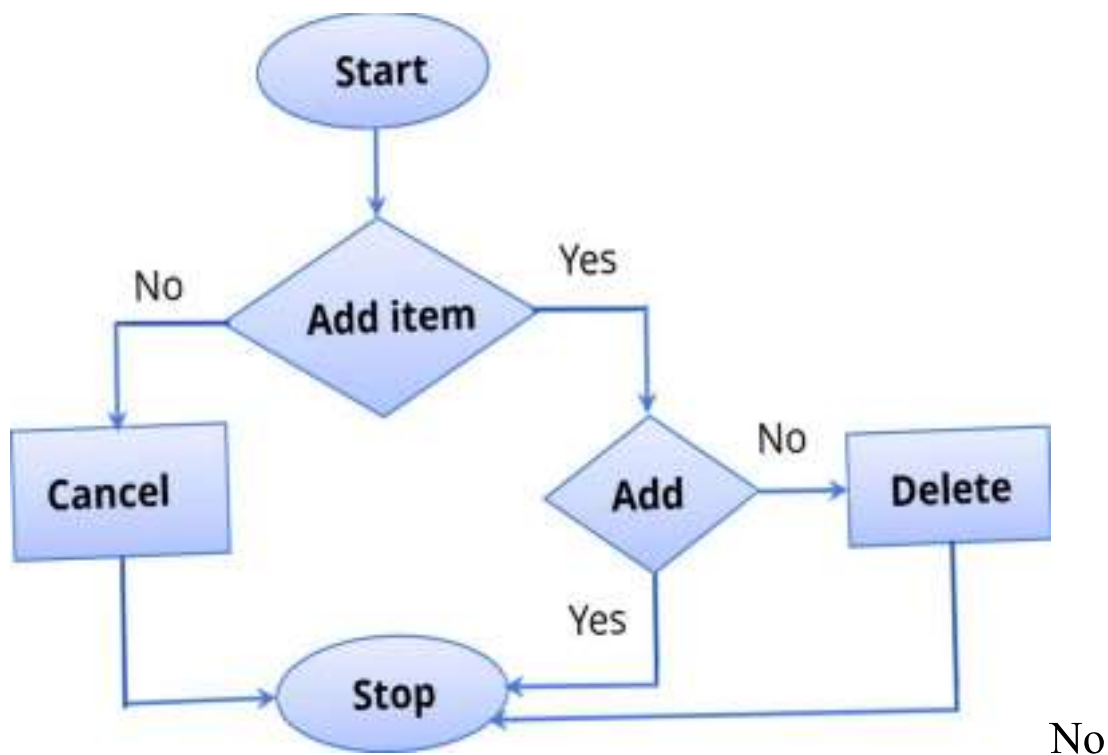
Smart

Internz

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

Coroutines: Coroutines are 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

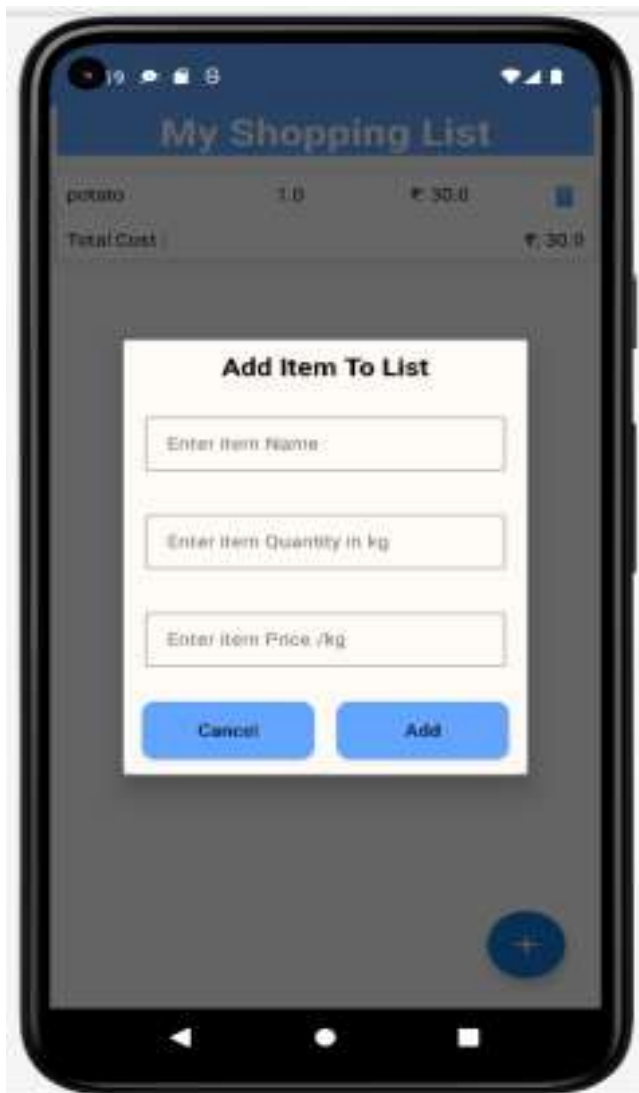


Smart
Internz

6.Result

Smart Internz

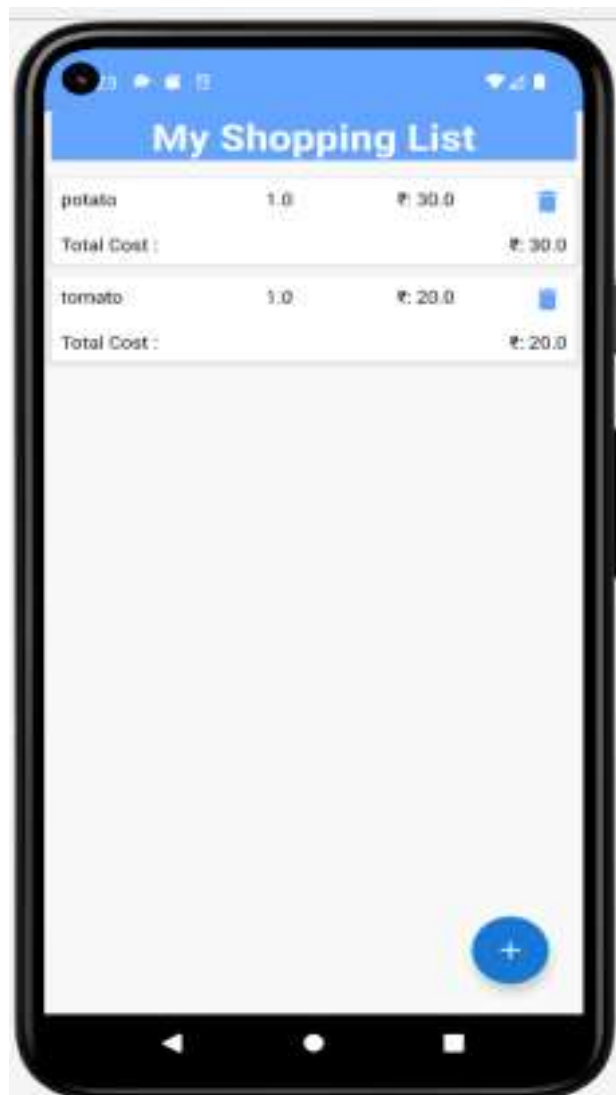
Emulator: 2 API 30



Event Log Layout inspector

Smart Internz

Emulator: 2 API 30



Smart Internz

Emulator: 2 API 30

Smart Internz

6. Conclusion

This project helped me to clear my concepts on Room Database, Coroutines, MVVM, etc. This project would help me not just as a developer to learn new and interesting things but also as a user we generally forgets items to purchase while shopping. Working on this project made me confident enough to apply my knowledge on android app development and create such an app. I have used Kotlin to build this application. All the functionality is coded in the classes and interfaces created and the layout is designed using xml.

8. Reference

- Google: <https://www.google.com/>
- Geeksforgeeks: <https://www.geeksforgeeks.org/how-to-build-a-grocery-androidapp-using-mvvm-and-room-database/>
- Android Developer:
<https://developer.android.com/codelabs/android-room-with-a-view-kotlin#0>
- YouTube: https://www.youtube.com/watch?v=vdcLb_Y71lc
- SmartInternz:
https://smartinternz.com/Student/guided_project_workspace/54944

9. Appendix

9.1 Source Code

MainActivity.kt

Smart

Internz



import androidx.lifecycle.Observer

import androidx.lifecycle.ViewModelProvider

import androidx.recyclerview.widget.LinearLayoutManager

import androidx.recyclerview.widget.RecyclerView

import com.google.android.material.floatingactionbutton.FloatingActionButton

on

FloatingActionButton

class MainActivity : AppCompatActivity(),

GroceryRVAdapter.GroceryItemClickListener {

lateinit var itemsRV: RecyclerView

lateinit var addFAB: FloatingActionButton

lateinit var list: List

lateinit var groceryRVAdapter: GroceryRVAdapter

lateinit var groceryViewModel: GroceryViewModel

onCreate(savedInstanceState:

override fun

Bundle?)

```

super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)

itemsRV = findViewById(R.id.rvItems)
addFAB = findViewById(R.id.fabAdd)
list = ()

groceryRVAdapter = GroceryRVAdapter(list, this) itemsRV.layoutManager =
LinearLayoutManager(this) itemsRV.adapter = groceryRVAdapter val groceryRepository
    = GroceryRepository
(GroceryDatabase(this)) val factory = GroceryViewModelFactory(groceryRepository)
groceryViewModel = ViewModelProvider(this, factory).get(GroceryViewModel::class.java)
groceryViewModel.getAllGroceryItems().observe(this, Observer {
groceryRVAdapter.notifyDataSetChanged()
})

```

```

addFAB.setOnClickListener {
    showDialog()
}

```

Smart Internz

```

fun showDialog() { val dialog
    Dialog(this) dialog
    setContentView(R.layout.dialog)
    cancelBtn = dialog.findViewById<Button>(R.id.btnCancel) val
    addBtn = dialog.findViewById<Button>(R.id.btnAdd) val itemEdit = dialog.findViewById<EditText>(R.id.itemEdit)
    itemQuantityEdit = dialog.findViewById<EditText>(R.id.itemQuantityEdit)
    cancelBtn.setOnClickListener { dialog.dismiss() }

    addBtn.setOnClickListener { val itemName: String = itemEdit.text.toString()
    val itemPrice: String = itemPriceEdit.text.toString()
    val itemQuantity: String = itemQuantityEdit.text.toString()
    val qty: Int = itemQuantity.toInt()
    val price: Int = itemPrice.toInt()
    if (itemName.isNotEmpty() && itemPrice.isNotEmpty() && itemQuantity.isNotEmpty()) {
        val items = GroceryItems(itemName, qty, price)
        groceryViewModel.insert(items)

        Toast.makeText(applicationContext, "Item Inserted.",
            Toast.LENGTH_SHORT).show()
        groceryRVAdapter.notifyDataSetChanged()
    }
}

```

```
dismiss ( )
```

```
Toast . makeText (
    applicationContext
    text,
    'Please Enter all data. ' + ,
    Toast . LENGTH—SHORT
) . show ( )
```

```
dialog . show ( )
```

```
override fun onItemClick (groceryItems: Grocery
    Items) groceryViewModal . delete (groceryItems
    ) groceryRVAdapter . notifyDataSetChanged ( )
    Toast . makeText (applicationContext,Item Deleted. .
    Toast.LENGTH—SHORT) . show ( )
```

Smart Internz

activity_main.xml

?xml 8

```
<RelativeLayout xmlns : android= 'http: //schemas . android.
com/apk/rets/android' xmlns : app= 'http: //schemas . android. com/apk/ res—autc
' xmlns: tools= 'http: //schemas . android. com/tools '

androic_d: layout _ '
android: layout _height— "mat
ch_parent ' android : background= @
color tools: context="
.MainActivity 'I >
```

```

lateinit var itemRV:RecyclerView
lateinit var addFAB:FloatingActionButton
lateinit var list: List<GroceryItems>
lateinit var groceryRVAdapter: GroceryRVAdapter
lateinit var groceryViewModel: GroceryViewModel />

```

<corn. google . android . material . float ingact ionbutt on . Float ingActionButton android:

```

id—" @+id/ idFABAdcl " android: layout_width—" wrap_content "

    android : wrap_ centert android : " t

    rue android. • "true "

    android: layout_ alignParentBottom—"

    I t rue " android: layout_ margin—" 2

    8dp android android: src= " "

```

/RelativeLayout>

<

Smart

Internz

<

.GroceryDaokt

```
import ...
```

```
@Dao
```

```
interface GroceryDao {  
    @Insert(onConflict = OnConflictStrategy.REPLACE)  
    suspend fun insert(item: GroceryItems)  
    @Delete  
    suspend fun delete(item: GroceryItems)  
    @Query("SELECT * FROM Grocery_items")  
    fun getAllGroceryItems(): LiveData<List<GroceryItems>>
```

Smart

Internz

GroceryItems.kt

```

import ...

@Entity(tableName = "Grocery_items")
data class GroceryItems (
    @ColumnInfo(name = "itemName")
    var itemName:String,

    @ColumnInfo(name = "itemQuantity")
    var itemQuantity:Double,

    @ColumnInfo(name = "itemPrice")
    var itemPrice:Double,
)
{
    @PrimaryKey(autoGenerate = true)
    var id:Int?=null
}

```

GroceryRepository.kt

```
package com.divyanshu.groceryapp

class GroceryRepository(private val db:GroceryDatabase) {
    suspend fun insert(items: GroceryItems) =
        db.getGroceryDao().insert(items)
    suspend fun delete(items: GroceryItems) =
        db.getGroceryDao().delete(items)

    fun getAllItems() = db.getGroceryDao().getAllGroceryItems() }
```

Smart

Internz

GroceryDatabase.kt

```
package com.divyanshu.groceryapp

import ...

@Database(entities = [GroceryItems::class], version = 1)

abstract class GroceryDatabase : RoomDatabase() {

    abstract fun getGroceryDao() : GroceryDao
    companion object {
        @Volatile
        private var instance: GroceryDatabase? = null
        private val LOCK = Any()

        operator fun invoke(context: Context) = instance ?: synchronized(
            instance?: createDatabase(context).also {
                instance = it
            }
        )

        private fun createDatabase(context: Context) =
            Room.databaseBuilder(
                context.applicationContext,
```

GroceryRVAdapter.kt

Package

com.example.groceryappsunidhi

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import

android.widget.ImageView

return

Smart

Internz

```
import android.widget . TextView
import androidx . recyclerview . widget .
    RecyclerView
--•lass GroceryRVAdapter
    ( var list:
    val groceryItemClickListener: GroceryItemClickListener
    RecyclerView . Adapter<GroceryRVAdapter . GroceryViewHolder> ( )

    inner class GroceryViewHolder (itemView: View)
RecyclerView . ViewHolder (itemView) (
    val nameTV — itemView . (R. id.idTVItemName) val quantityTV — itemView
    . (R. id.idTVQuantity)
    val rateTV = itemView.findViewById<TextView> (R.id. idTVRate)
    id. idTVTotalAmt)

    val deleteTV = itemView . (R. id.
                                idTVDelete)

    interface GroceryItemClickListener { fun
        onItemClick (groceryItems : Grocery
        Items)
```

```
        override fun onCreateView (parent: ViewGroup, viewType : Int):
GroceryViewHolder {
    val view

    LayoutInflater . from (parent . context) . inflate (R. layout . tern, oarent , false) return
    GroceryViewHolder (view)
```

```
override fun onBindViewHolder (holder: GroceryViewHolder, position: Int) holder . nameTV
    . text = list. get (position) . itemName holder . itemTV . text = list . get (position) . i
    temQuantity . toString()
```

Smart

Internz

strings.xml

```
resources for all resources in the translations folder. Open Editor Hide Notifications  
<resources> ✓  
  <string name="app_name">Grocery App</string>  
</resources>
```

Note :- Since the page limit is exceeding I can't put the whole source code here. Please check the drive link or the github link below for full code.

Drive link:

<https://drive.google.com/file/d/1JbBKwASgGicXqYgrzSk1Jbf4swBvvBpV/view?usp=sharing>

Github link: <https://github.com/smartinternz02/SPSGP-54944-Virtual-Internship---Android-Application-Development-Using-Kotlin>

