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.

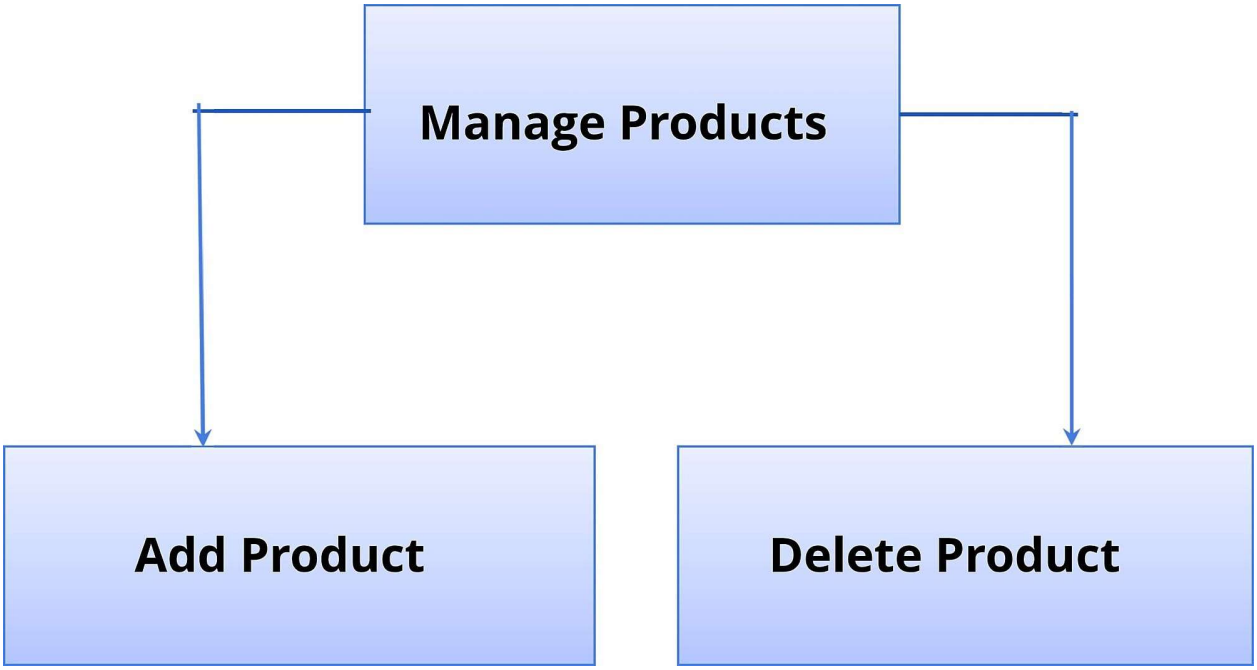
2.2 Proposed Solution

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

i. Windows 10 OS

ii. Android Studio

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.

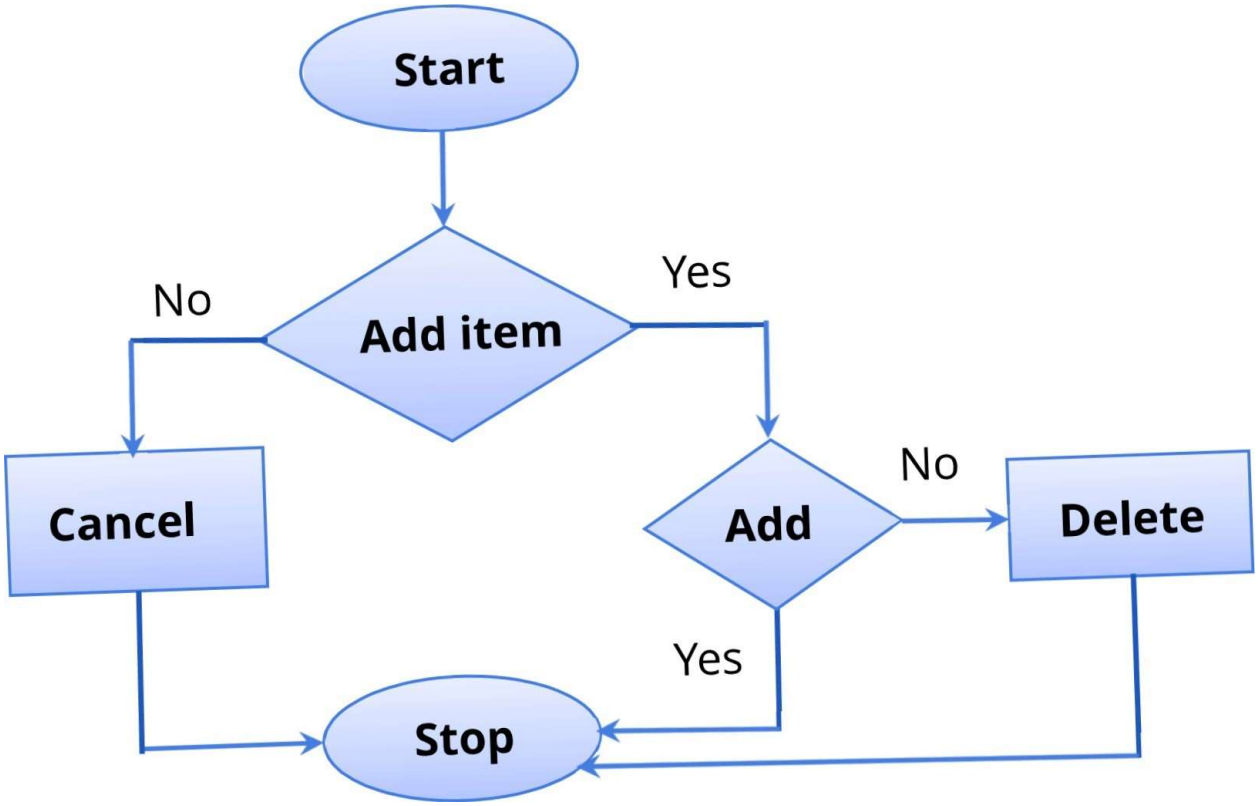
SQLite database: On device storage. The Room persistence library creates and maintains this database for you.

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.

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.

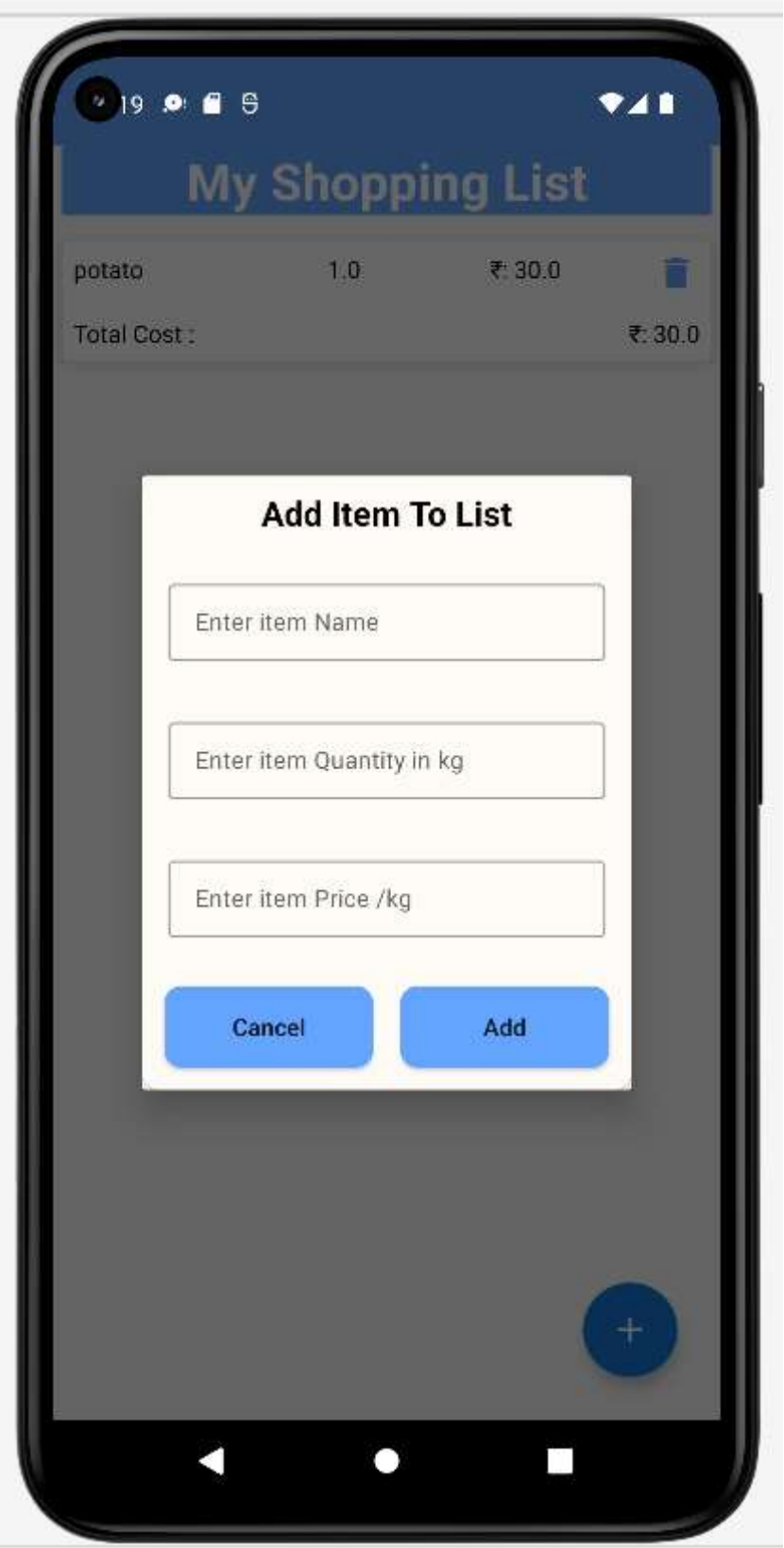
Coroutines: Coroutines are lightweight thread, we use a coroutine to perform an operation on other threads, by this our main thread doesn't block and our app doesn't crash.

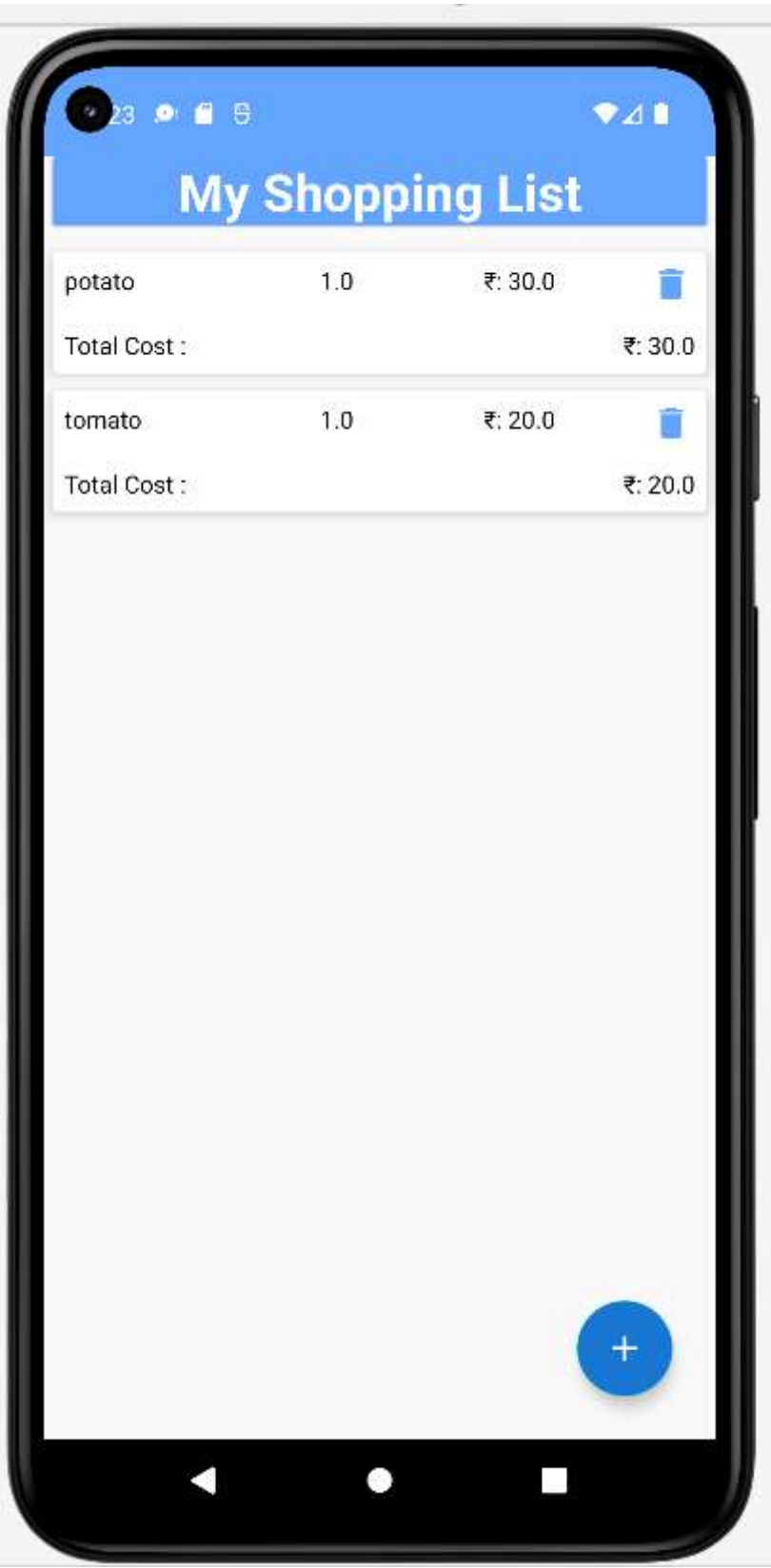
5. Flowchart



No

6.Result


My Shopping List
potato
1.0
₹: 30.0
Total Cost :
₹: 30.0
Add Item To List
Enter item Name
Enter item Quantity in kg
Enter item Price /kg
Cancel
Add


My Shopping List
potato
1.0
₹: 30.0
Total Cost :
₹: 30.0
tomato
1.0
₹: 20.0
Total Cost :
₹: 20.0

7. 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

1. Google: https://www.google.com/
2. Geeksforgeeks: https://www.geeksforgeeks.org/how-to-build-a-grocery-androidapp-using-mvvm-and-room-database/
3. Android Developer: hÜps://developer.android.com/codelabs/android-room-witha-view-kotlin#0
4. YouTube: hÜps://www.youtube.com/watch?v=vdcLb_Y71 lc
5. SmartInternz: hÜps://smartinternz.com/Student/guided project workspace/55908

9.Appendix

9.1 Source Code

MainActivity.kt

```
pleInstrumentedTest.kt   MainActivity.kt
                                                    3  1  1  5
class MainActivity : AppCompatActivity(),GroceryRVAdapter.GroceryItemCli
    lateinit var itemRV:RecyclerView
    lateinit var addFAB:FloatingActionButton
    lateinit var list: List<GroceryItems>
    lateinit var groceryRVAdapter: GroceryRVAdapter
    lateinit var groceryViewModel: GroceryViewModel
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(sav
```

```
t11porL arid r Oidx. 11 recycle . Observer
rnport androidx . li fecycle . ViewMode1Provider
import androidx . recyclerview . widget . LinearLayoutManager
import androidx . recyclerview . widget . RecyclerView
import com.google . android . material . float ingactionbutton .        on
       FloatingAct
•lass MainActivity : AppCompatActivity ( ) ,
GroceryRVAdapter . GroceryTtemClickTnterface {
    lateinit var itemsRV: RecyclerView
    lateinlt var adclFAB: FloatingActionButton
    lateinlt var list : List
    lateinlt var groceryRVAdapter: GroceryRVAdapter
    lateinit var groceryViewModal: GroceryViewMode1
                onCreate        (savedInstanceState:
    override fun Bundle?)
        super    .    onCreate    (savedInstanceState)
        setContentView (R. layout . acti vity_main)

        itemsRV  =  findViewById  (R.id.  idRVItems)
        addFAB = findViewByTd (R. id. idFABAdd)
        list -                        ( )
        graceryRVAdapter = GroceryRVAdapter (1 i st, this) itemsRV .
layoutManager  -  LinearLayoutManager  (this)  itemsRV.  adapter  -
groceryRVAdapter   val   groceryRepository   -   GroceryRepository
(GroceryDatabase (this) ) val factory = GroceryViewMode1Factory (groce
ryRepository) groceryViewMoclal = ViewMode1Provider (this, factory) .
get   (GroceryViewModeI   :   :class   .   java)   groceryvlewmodal   .
```

```
getA11GroceryIterns ( ) . observe (this, Observer {
groceryRVAdapter.1ist - groceryRVAdapter . notifyDataSetChanged ( )


        addFAB . set OnC1ickListener
            openDia10g ( )



    fun openDiaIog ( ) val dialog
        Dialog (this) dialog .
        setConter,tView (R. layout .
            cancelBtn dialog .findViewById<Button> (R. id. idBtnCance1)
        val addBtn = dialog . (R. id. idBtnAdd) val itemEdt -
        dialog . (R. id. idEdt Tternmame) dxalog . (R. ce)
        val itemQuantityEdt = dialog . findViewById<EditText > (R .
        icl. idEdtItemQuanti ty) cancelBtn . setOnC1ickListener {
        dialog . dismiss ()

        addBtn . setOnC1ickListener val itemName: String = itemEdt.
text . tostring ( ) val itemPrice: String - itemPriceEdt . text .
toString ( ) val itemQuantity: String = itemQuantityEdt . text .
toString ( ) val qty: Int = itemQuantity . tolnt ( ) v al pr: Int
itemPrice . tornt ( ) i f (itemName . isNotEmpty() S S itemPrice.
isNotEmpty() itemQuantity . isNotEmpty() ) val items = Groceryltems
(itemName, qty, pr) groceryViewModa1. insert (items)
            Toast . makeText (applicationCont ext, Item Inserted. .
roast.LENGTH_SHORT) . show ( ) groceryRVAdapter .
                notifyDataSetChanged ( ) dialog .
                dismiss ( )

                Toast . makeText (
                    applicationCon
                    text,
                    '1 Please Enter all data. . '1 ,
                    Toast . LENGTH-SHORT
                ) . show ( )




        dialog . show ()
```

```
override fun onltemC1ick (groceryltems: Grocery
    Items) groceryV1ewModa1 . delete (groceryltems
    ) groceryRVAdapter . not ifyDat aSetChanged (
    )
  Toast . makeText (applicationContext, Item Deleted. .
roast.LENGTH-SHORT) . show ( )
```

activity_main.xml

```
<Re1ativeLayout xmlns : android= 'l http: / /schemas . android.
com/apk/rets/android" xmlns : app="       / / schemas . android . com/apk/
res-autc ' xmlns: tools= 'l http: //schemas . android. com/tools '
   androic_d: layout _                    '
   android:    layout    _height-    "mat
   ch_parent  '  android  :  background=  @
   c•olor  tools:  context="  .MainActivity
   'I >
```

```
lateinit var itemRV:RecyclerView
lateinit var addFAB:FloatingActionButton
lateinit var list: List<GroceryItems>
lateinit var groceryRVAdapter: GroceryRVAdapte
lateinit var groceryViewModel: GroceryViewMode
```

```
<corn. google . android . material . float ingact ionbutt on . Float
    ingActionButton android: id- " @+id/ idFABAdcl " android:
    layout_width- t ' wrap_content '
    android :  wrap_ccntert android :  " t rue
    android.•  "true '
    android: layout_a1ignParentBottorn-  ' I  t
    rue  "  android:  layout_margin-  "  2  8dp
```

```
android android: src= "      "

<RelativeLayout>
```

GroceryDao.kt

```
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>>
```

GroceryItems.kt

```kotlin
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()
 }
```

<

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

 mport android. view . Layout Inflater
import android. view . View
import android. view . ViewGroup
 mpo rtandroid. widget . ImageView

 rnpo rt android. widget . TextView
 mport androidx . recyclerview . widget .
       RecyclerView
--•lass GroceryRVAdapter
    ( var list:
```

```
    val groceryItemClickInterface: GroceryItemClickInterface
    RecyclerView . Adapter<GroceryRVAdapter . GroceryViewE01der> ( )

    inner class GroceryViewH01der (itemView: View)
RecyclerView . ViewH01der (itemView) (
        val  nameTV  -  itemView  .          (R.  id.iciTVItemlVame)  val
        quantityTV - itemView .      (R. id.idTVQuantity)
        val rateTV = iternView.findViewByTd<TextView> (R.id. idTVRate)
        val amount TV - itemView .                    (R.            id.
                                                      idTVTota1Amt)
        val deleteTV = itemView .                      (R. id.
                                                       idTVDe1ete)
    interface   GroceryItemClickInterface   {   fun
        onltemClick   (groceryIterns   :   Grocery
        Items)


    override fun onCreateViewH01der (parent: ViewGroup, viewType : Int):
GroceryViewHoIder   {
        val view
  Layout Inflater . from (parent . context) . inflate (R. layout .
tern,
oarent      ,      false)      return
        GroceryViewH01der (view)


    override  fun  onBindViewH01der  (holder:  GroceryViewH01der,  position:
        Int)  holder  .  nameTV  .  text  =  list.  get  (position)  .  itemName
        holder  .  i  tv  TV  .  text  l  i  st  .  get  (position)  .  i  temQu:anti
        t.v .toStrinq()
```

strings.xml

```
<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/SI-GuidedProject-55908-1663222170