Smart Internz

Build A Grocery Android App

1. INTRODUCTION

1.1 Overview

This is an androidapp that helps you to make a list of grocery items along with its priceand quantity.

1.2 Purpose

As we can't remember everything, we frequently forget to purchase things that we want to buy. However, with the aid of this app, you may make a list of the groceries you intend to buy so that you don't forget anything. Kotlin was used to implement this project.

2. LITERATURE SURVEY

2.1 Existing Problem

Users frequently forget what they need to buy, forcing them to rush back and forth to the store, which is rather unpleasant and exhausting. If our spending go over our budget while shopping, that could be a cause for concern.

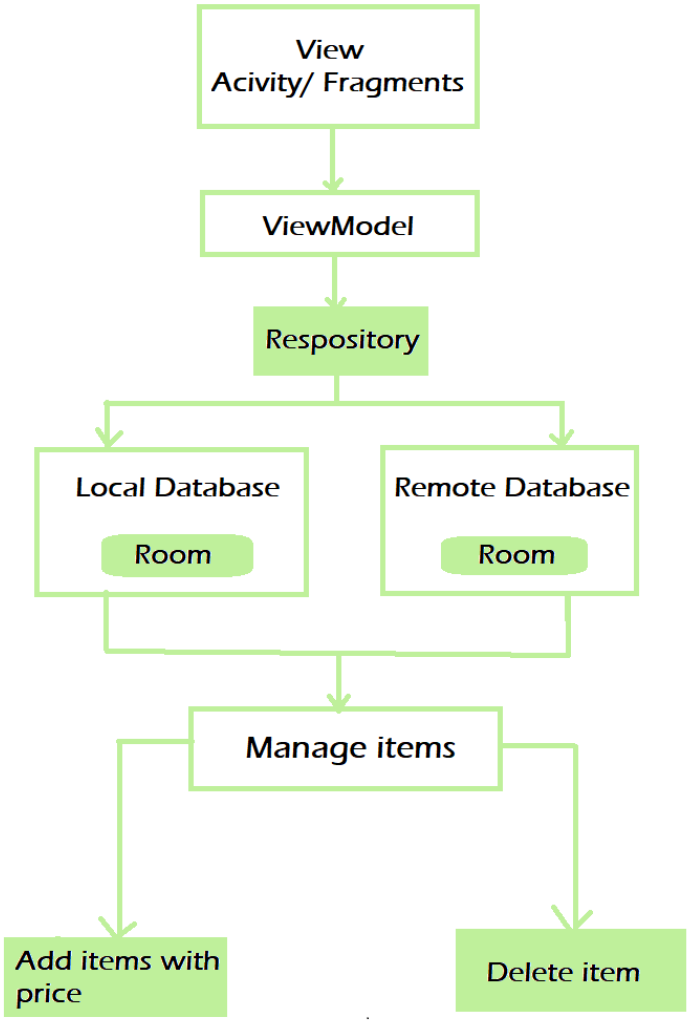
2.2 Proposed Solution

To overcome the above challenging circumstance A grocery app I created allows to list every item you need to buy, along with its price.

Smart Internz

3. THEORETICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware/Software designing

i. Windows 10 OS

ii. Android Studio

iii. Emulator/ physical device

4. EXPERIMENTAL INVESTIGATIONS

Coroutines, Model View ViewModel (MVVM) architectural patterns, Room for the database, and RecyclerView to display the list of objects are all used in this project.

LiveData

A class that holds observable data. The latest version of the data is always held or cached, and it alerts observers when the data has changed. LiveData is aware of the lifespan. UI elements don't pause or resume observation; they only keep an eye on the relevant data. Since LiveData is aware of the relevant lifecycle status changes while monitoring, it handles all of this automatically.

ViewModel

Serves as a conduit for information between the UI and the Repository (data). The source of the data is no longer a concern for the UI. ViewModel instances are preserved during Activity/Fragment playback.

MVVM (Model View ViewModel)

Android uses MVVM architecture to arrange project code and make it simpler to understand. An architectural design pattern used in Android is MVVM. XML files and Activity classes are treated as Views by MVVM. With this design approach, UI and logic are entirely separated. Here is a visual representation of MVVM.

DataBase ROOM

The data of apps, such as the name, amount, and price of groceries, are stored in the Room persistence library, a database management library. Room is a cover layer for SQLite that makes it easier to operate on databases.

RecycleView

RecyclerView is a container that is used to show a collection of data in a sizable data set that can be scrolled efficiently by keeping the number of views to a minimum.

Coroutines

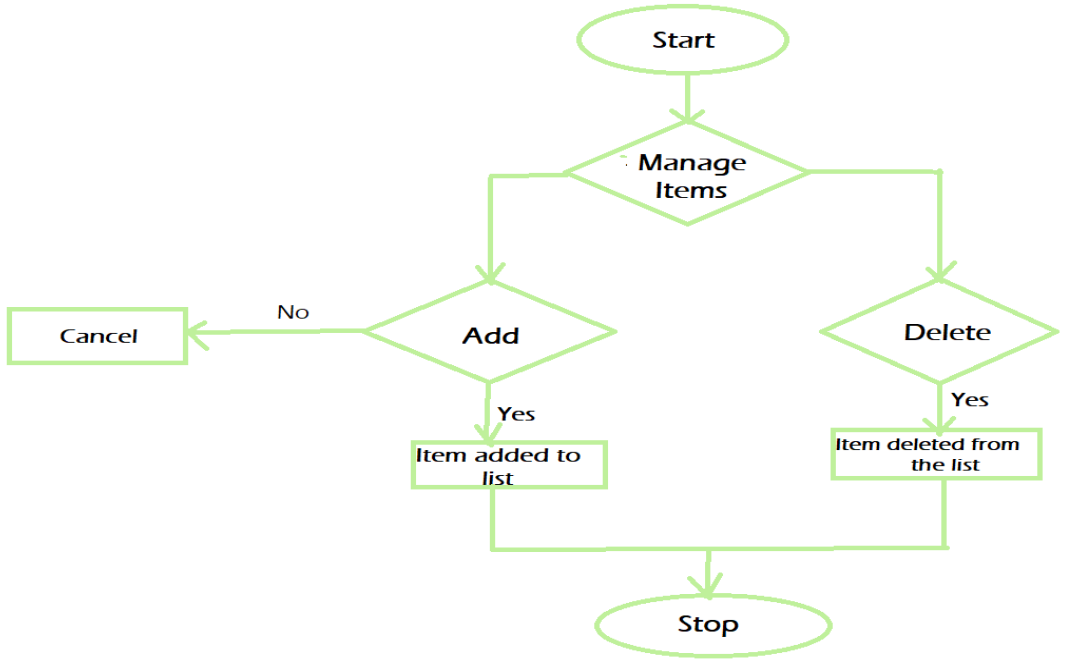
Since coroutines are lightweight threads, we utilise them to execute operations on other threads so that our app doesn't crash or cause our main thread to halt.

Entity: Annotated classthat describes a database table when workingwith Room.

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

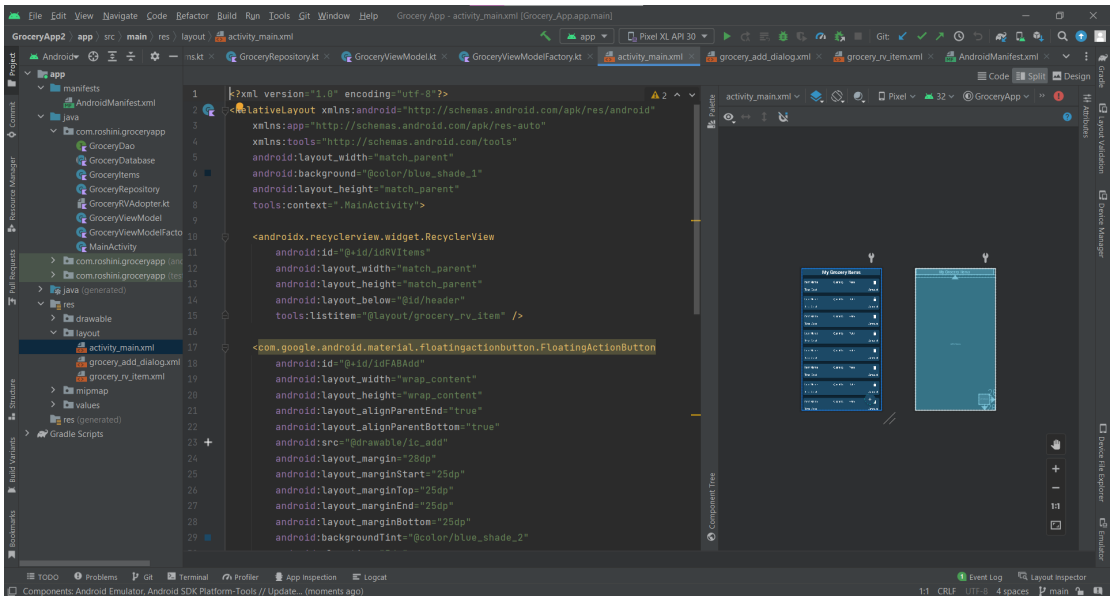
DAO: Data access object.A mapping of SQL queriesto functions. When you use a DAO,you call the methods, and Room takes care of the rest.

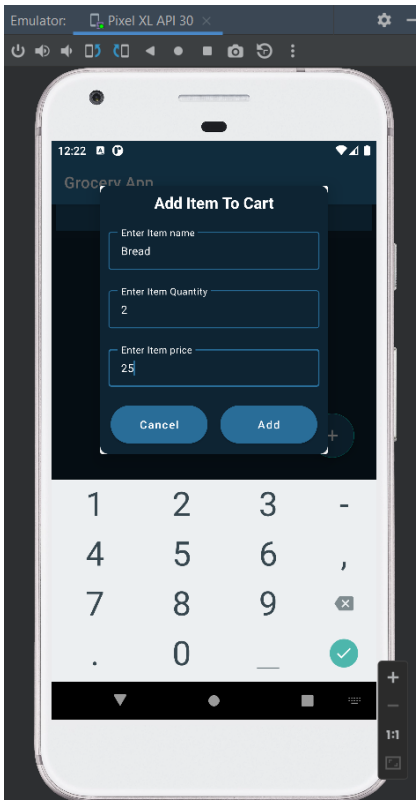
5. FLOWCHART

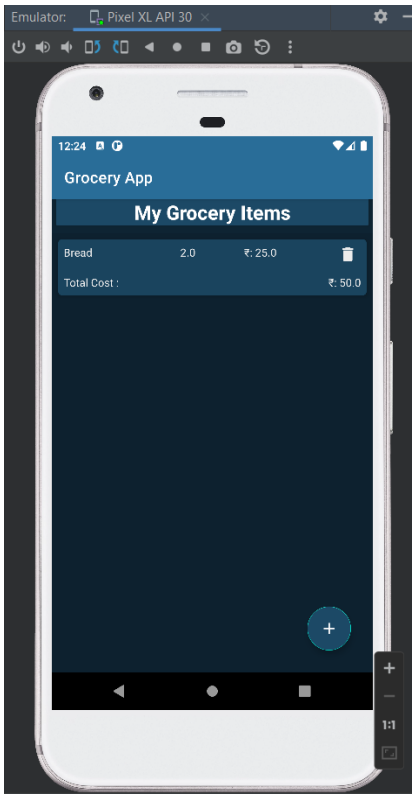


Smart Internz

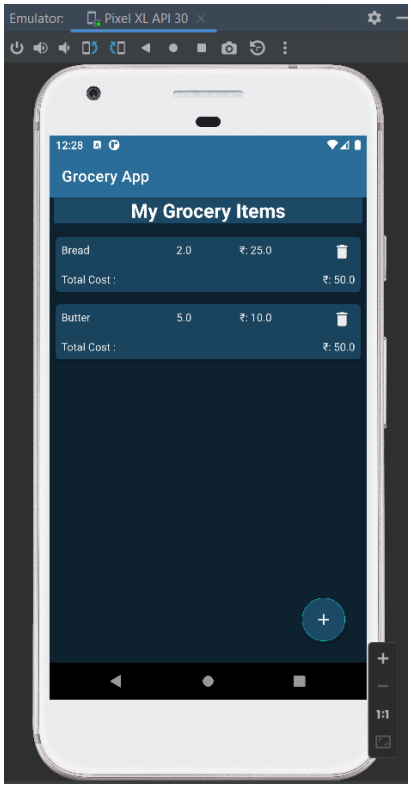
6. RESULT

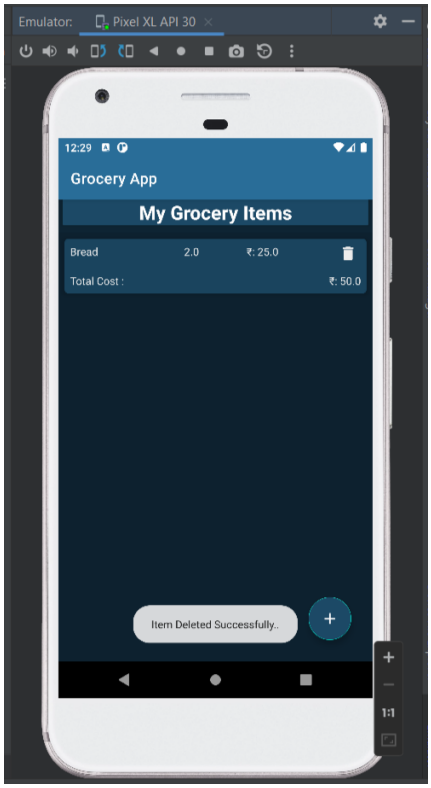






Adding new item into the list





Deleting existing item from the cart

7. ADVANTAGES & DISADVANTAGES

ADVANTAGES

- simple price notes on the things
- The total budget will be made available in advance.
- user pleasant and simple for everyone to update

DISADVANTAGE

- Difficult to know the prices ahead of time
- the quantity of things may change depending on customer preference

8. APPLICATIONS

Make a list of the items you want to buy with this app so we don't forget anything. The program also adds up each item's total along with its quantity. this facilitates budgeting and buying.

The application can be widely used in shopping, listing the items necessary to buy and delete the items once purchased.

9. CONCLUSION

I was able to understand room databases, coroutines, MVVM, and other ideas because to this project. This project would help me learn new and fascinating things as a user as well as a developer because we frequently forget what we need to get while shopping.

My confidence to use my understanding of Android app development and produce such an app increased as a result of working on this project. To create this application, I used Kotlin. The layout is constructed using xml, and all the functionality is coded in the created classes and interfaces.

10. FUTURE SCOPE

- We can add up the total value of items in list.
- The app can be connected to any online grocery shopping app to purchase it online.
- Voice can be enabled in the app.
- An intelligent grocery app that uses AI to make suggestions to users based on previously added products to cart can be presented.

11. BIBILOGRAPHY

References

- **Geeksforgeeks:** https://www.geeksforgeeks.org/how-to-build-a-grocery-android-app-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_Y71Ic
- **SmartInternz:** https://smartinternz.com/Student/guided_project_info/82262#

APPENDIX

A. Source Code

GroceryDao.kt

```
package com.roshini.groceryapp

import androidx.lifecycle.LiveData
import androidx.room.*

@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>>
```

GroceryDatabase.kt

```
package com.roshini.groceryapp

import android.content.Context
```

Smart Internz

```
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase


@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(LOCK)
{
            instance?: createDatabase(context).also {
                instance = it
            }

        }


        private fun createDatabase(context: Context) =
            Room.databaseBuilder(
                context.applicationContext,
                GroceryDatabase::class.java,
                "GroceryApp.db"
            ).build()
    }
}
```

GroceryItems.kt

```
package com.roshini.groceryapp

import androidx.room.ColumnInfo
import androidx.room.Entity
import androidx.room.PrimaryKey

@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.roshini.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()
}
```

GroceryRAVAdopter.kt

```
package com.roshini.groceryapp

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView

class GroceryRVAdapter(
    var list: List<GroceryItems>,
    val groceryItemClickInterface: MainActivity
)
    : RecyclerView.Adapter<GroceryRVAdapter.GroceryViewHolder>() {

    inner class
GroceryViewHolder(itemView:View):RecyclerView.ViewHolder(itemView){
```

```
        val nameTV = itemView.findViewById<TextView>(R.id.idTVItemName)
        val quantityTV = itemView.findViewById<TextView>(R.id.idTVQuantity)
        val rateTV = itemView.findViewById<TextView>(R.id.idTVRate)
        val totalTV = itemView.findViewById<TextView>(R.id.idTVTotalAmt)
        val deleteIV = itemView.findViewById<ImageView>(R.id.idTVDelete)
    }


    interface GroceryItemClickInterface{
        fun onItemClick(groceryItems: GroceryItems)
    }


    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
GroceryViewHolder {
        val view =
LayoutInflater.from(parent.context).inflate(R.layout.grocery_rv_item,parent,fals
e)
        return GroceryViewHolder(view)
    }

    override fun onBindViewHolder(holder: GroceryViewHolder, position: Int) {
        holder.nameTV.text = list.get(position).itemName
        holder.quantityTV.text = list.get(position).itemQuantity.toString()
        holder.rateTV.text = "₹: "+list.get(position).itemPrice.toString()
        val itemTotal :Double = list.get(position).itemQuantity *
list.get(position).itemPrice
        holder.totalTV.text = "₹: "+itemTotal.toString()
        holder.deleteIV.setOnClickListener {
            groceryItemClickInterface.onItemClick(list.get(position))
        }
    }

    override fun getItemCount(): Int {
        return list.size
    }
}
```

GroceryViewModel.kt

```
package com.roshini.groceryapp

import androidx.lifecycle.ViewModel
import kotlinx.coroutines.GlobalScope
```

```
import kotlinx.coroutines.launch

class GroceryViewModel(private val repository: GroceryRepository):ViewModel()
{
    fun insert(items: GroceryItems) = GlobalScope.launch {
        repository.insert(items)
    }
    fun delete(items: GroceryItems) = GlobalScope.launch {
        repository.delete(items)
    }
    fun getAllGroceryItems() = repository.getAllItems()
}
```

GroceryViewModelFactory.kt

```
package com.roshini.groceryapp

import androidx.lifecycle.ViewModel
import androidx.lifecycle.ViewModelProvider

class
GroceryViewModelFactory(private val repository:
GroceryRepository):ViewModelProvider.NewInstanceFactory() {
    override fun <T : ViewModel> create(modelClass: Class<T>): T {
        return GroceryViewModel(repository) as T
    }
}
```

MainActivity.kt

```
package com.roshini.groceryapp

import android.app.Dialog
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.widget.AppCompatButton
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
import com.google.android.material.snackbar.Snackbar
import java.util.*
import androidx.appcompat.app.AppCompatDelegate as AppCompatDelegate

class MainActivity :
AppCompatActivity(),GroceryRVAdapter.GroceryItemClickInterface {
    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(savedInstanceState)
        setContentView(R.layout.activity_main)
        itemRV = findViewById(R.id.idRVItems)
        addFAB = findViewById(R.id.idFABAdd)
        list = ArrayList<GroceryItems>()
        groceryRVAdapter = GroceryRVAdapter(list,this)
        itemRV.layoutManager = LinearLayoutManager(this)
        itemRV.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.list = it
            groceryRVAdapter.notifyDataSetChanged()
        })
        addFAB.setOnClickListener{
            openDialog()
        }

    }
    fun openDialog(){
```

```
        val dialog = Dialog(this)
        dialog.setContentView(R.layout.grocery_add_dialog)
        val cancelbtn =
dialog.findViewById<AppCompatButton>(R.id.idBtnCancel)
        val addbtn =
dialog.findViewById<AppCompatButton>(R.id.idBtnAdd)
        val itemEdt = dialog.findViewById<EditText>(R.id.idEdtItemName)
        val itemPriceEdt =
dialog.findViewById<EditText>(R.id.idEdtItemPrice)
        val itemQuantityEdt =
dialog.findViewById<EditText>(R.id.idEdtItemQuantity)
        cancelbtn.setOnClickListener {
            dialog.dismiss()
        }
        addbtn.setOnClickListener {
            val itemName:String = itemEdt.text.toString()
            val itemPrice:String = itemPriceEdt.text.toString()
            val itemQuantity:String = itemQuantityEdt.text.toString()
            val qty : Double = itemQuantity.toDouble()
            val pr : Double = itemPrice.toDouble()
            if (itemName.isNotEmpty() && itemPrice.isNotEmpty() &&
itemQuantity.isNotEmpty()){
                val items = GroceryItems(itemName,qty,pr)
                groceryViewModel.insert(items)
                Toast.makeText(applicationContext,"Item
Added",Toast.LENGTH_SHORT).show()
                groceryRVAdapter.notifyDataSetChanged()
                dialog.dismiss()
            }
            else{
                Toast.makeText(applicationContext,"Please fill all
details",Toast.LENGTH_SHORT).show()
            }

        }
        dialog.show()
    }

    override fun onItemClick(groceryItems: GroceryItems) {
        groceryViewModel.delete(groceryItems)
        groceryRVAdapter.notifyDataSetChanged()
```

```
        Toast.makeText(applicationContext,"Item Deleted
Successfully..",Toast.LENGTH_SHORT).show()
    }
}
```

Since the page limit is exceeding I can't put the whole source code here. Please check the demo link and the github link below for complete code.

URL's

- Github URL: https://github.com/smartinternz02/SPSGP-82262-Virtual-Internship---Android-Application-Development-Using-Kotlin
- Google Developer Profile: https://g.dev/roshinir
- Demo link: https://drive.google.com/file/d/1tWAHzdpk3nEu9mgZtr8yjqa7odM16qhx/view?usp=sharing
- smartinternz id: SB20220191959
- smartinternz registered email id: rroshini3@gmail.com