# *Aim* ---> Build A Grocery Android App

***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 list of the groceries you intend to buy so that you don't forget anything.

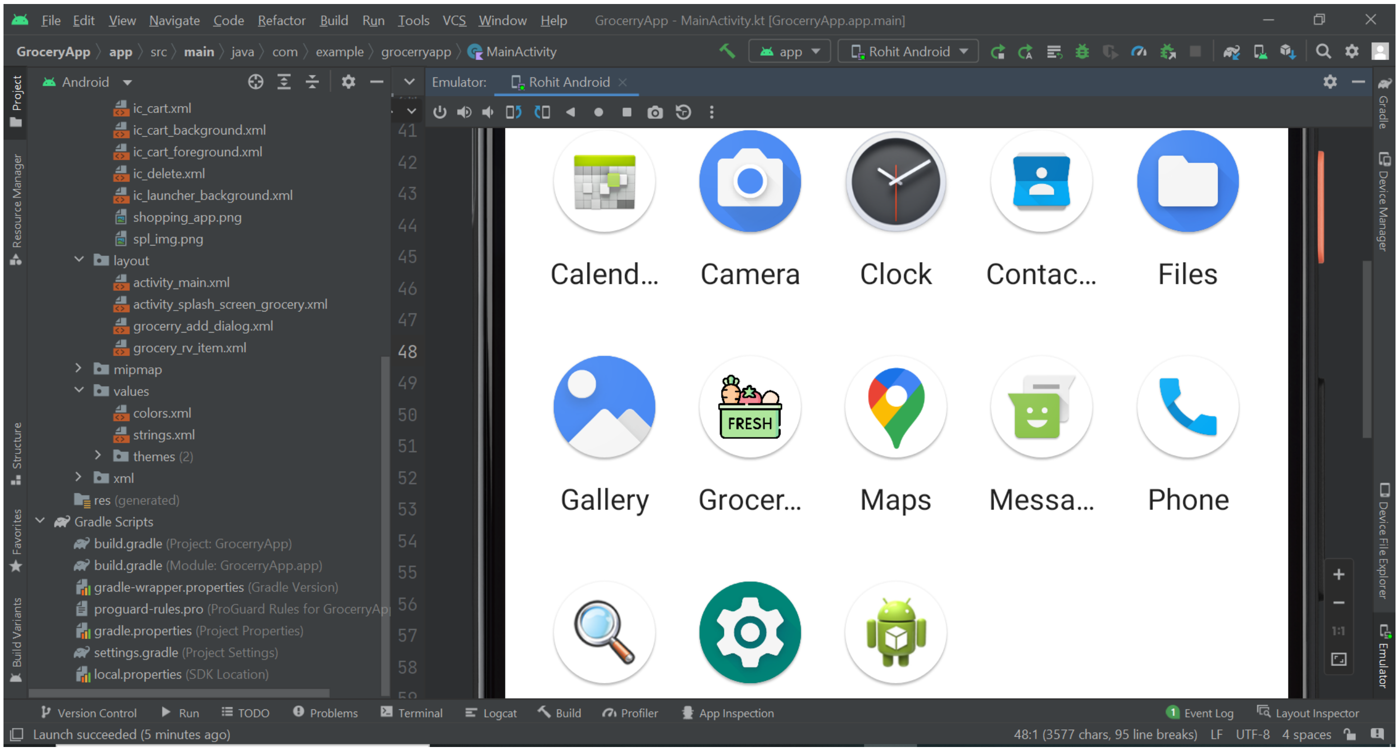
***Solution  -->***

Build a grocery application in android using android studio. Many times we forget to purchase things that we want to buy, after all, we can't remember all the items, so with the help of this app, you can note down your grocery items that you are going to purchase, by doing this you can't forget any items that you want to purchase.

- In this project, we are using MVVM(Model View ViewModel) for architectural patterns, **Room** for database,Coroutines and RecyclerView to display the list of items.
- Language used --- **Kotlin**

## ***GROCERY APP***

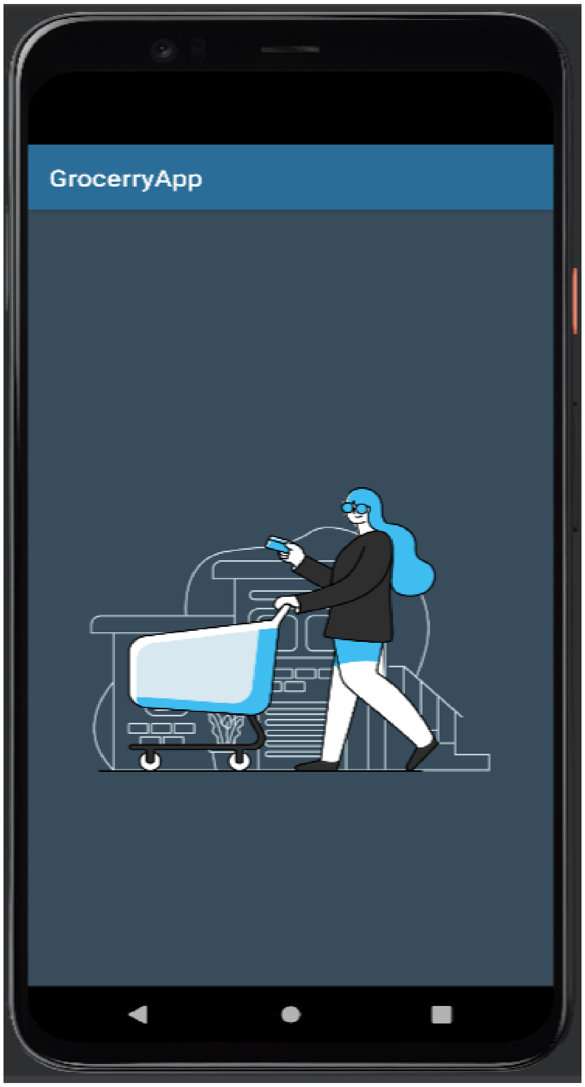
ICON



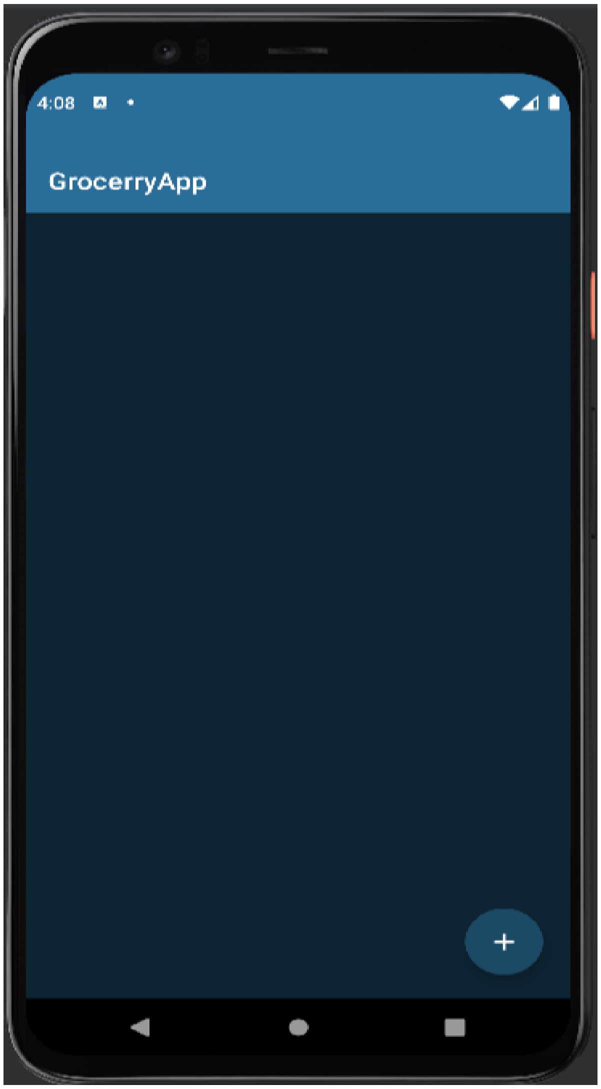
Launch succeeded (5 minutes ago)

48:1 (3577 chars, 95 line breaks) LF UTF-8 4 spaces

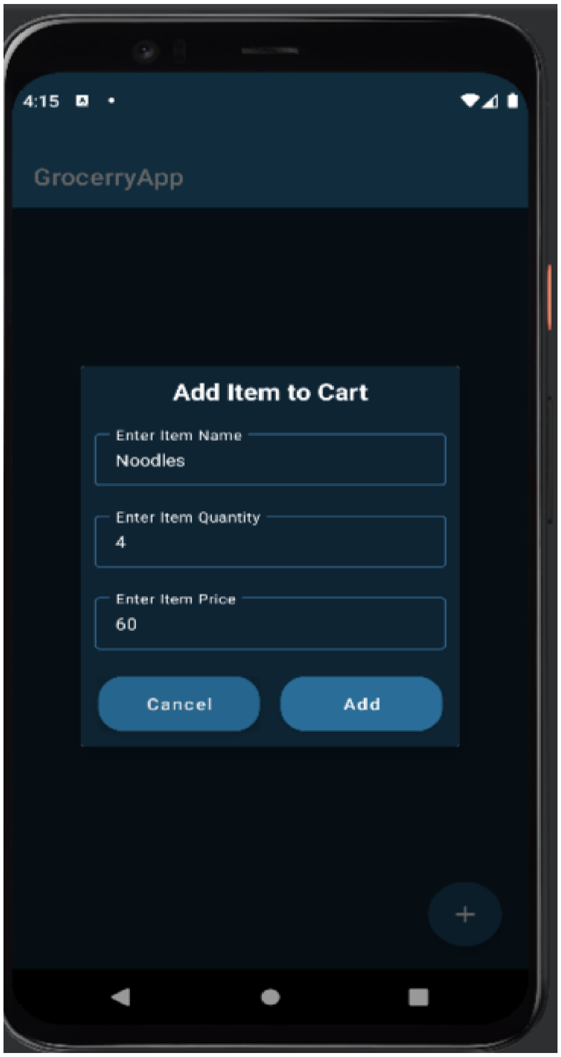
Splash Screen



After Splash Screen



Dialog Box



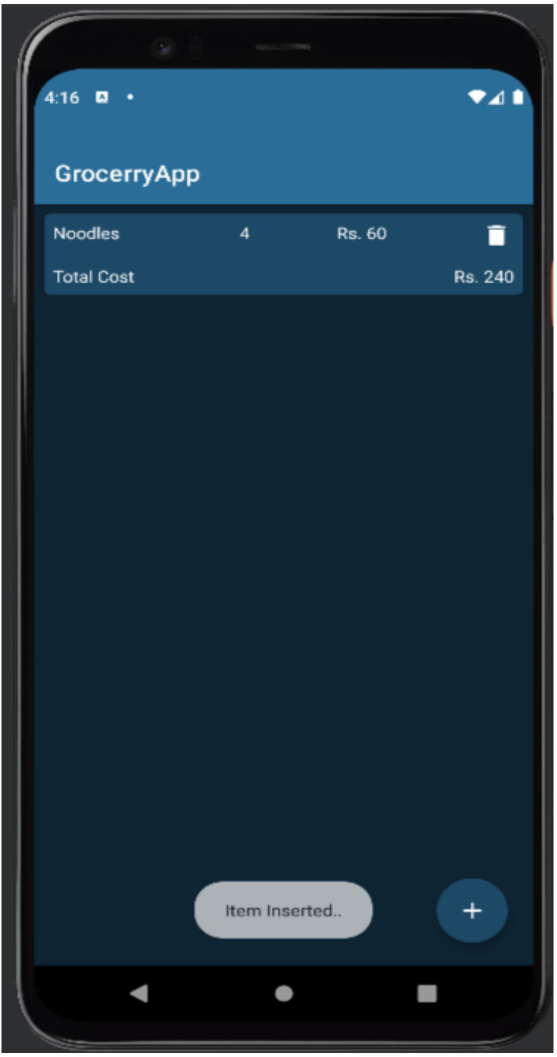
Add Item to Cart

Enter Item Name
Noodles

Enter Item Quantity
4

Enter Item Price
60

Item Inserted

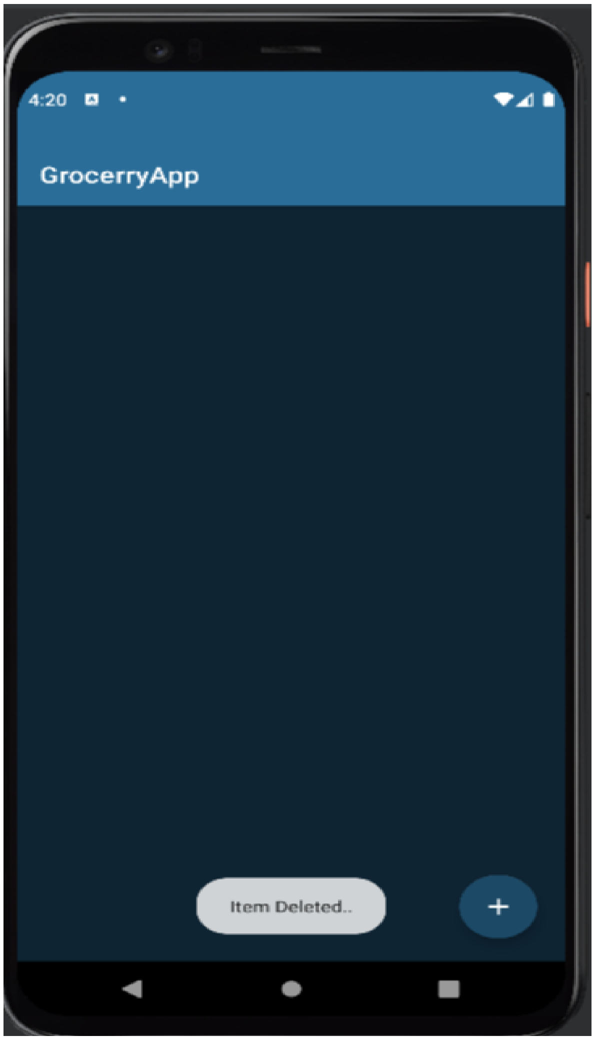


| | | | |
|---|---|---|---|
| Noodles | 4 | Rs. 60 | 🗑 |
| Total Cost | | | Rs. 240 |

Item Inserted..

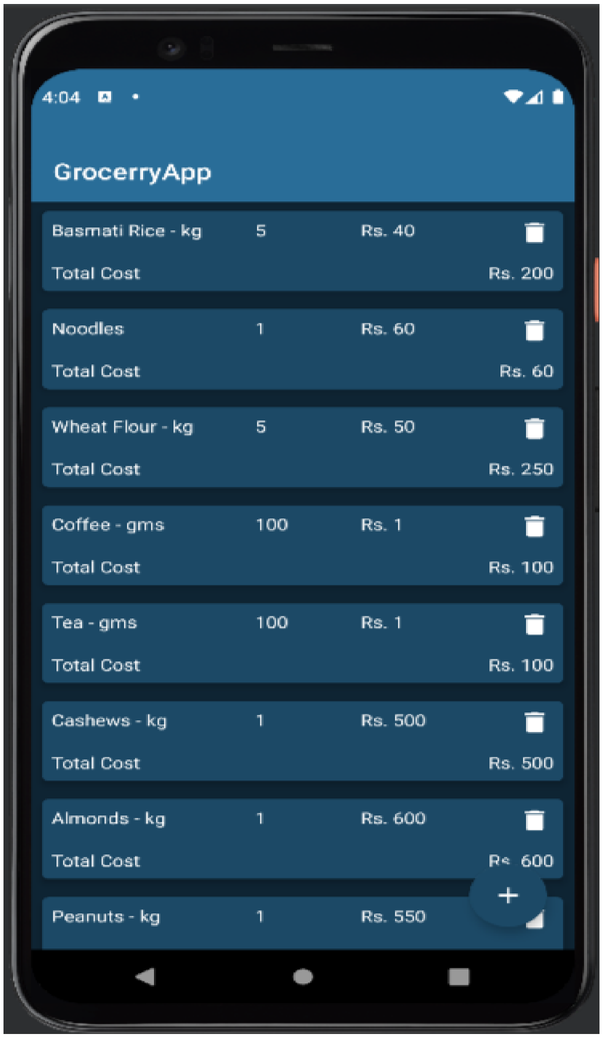
It

Item Deleted



Item Deleted..

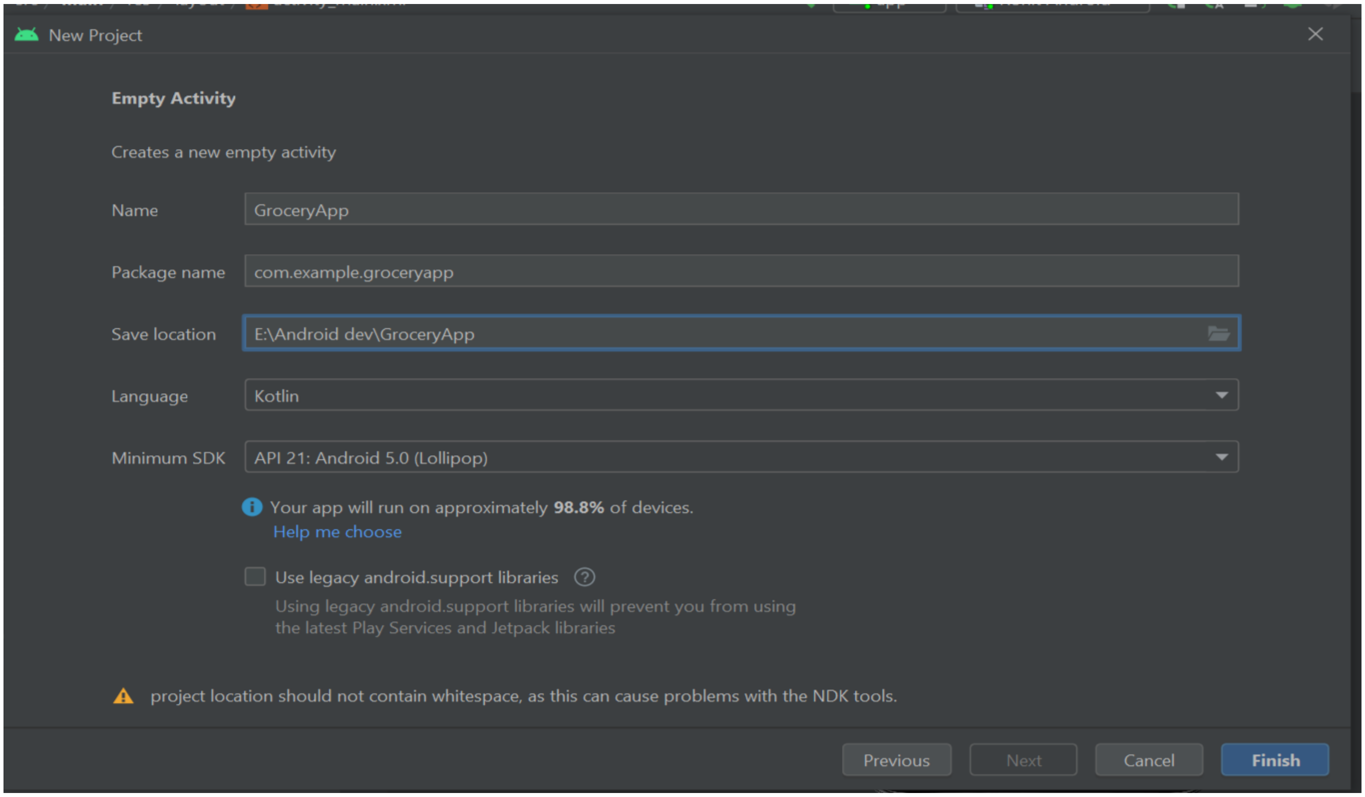
List of Grocery Items



| | | | |
|---|---|---|---|
| Basmati Rice - kg | 5 | Rs. 40 | |
| Total Cost | | | Rs. 200 |
| Noodles | 1 | Rs. 60 | |
| Total Cost | | | Rs. 60 |
| Wheat Flour - kg | 5 | Rs. 50 | |
| Total Cost | | | Rs. 250 |
| Coffee - gms | 100 | Rs. 1 | |
| Total Cost | | | Rs. 100 |
| Tea - gms | 100 | Rs. 1 | |
| Total Cost | | | Rs. 100 |
| Cashews - kg | 1 | Rs. 500 | |
| Total Cost | | | Rs. 500 |
| Almonds - kg | 1 | Rs. 600 | |
| Total Cost | | | Rs. 600 |
| Peanuts - kg | 1 | Rs. 550 | |

### *Step by Step Implementation -->*

- **Create a New Project - Select Kotlin as a programming language**



New Project

**Empty Activity**

Creates a new empty activity

Name: GroceryApp

Package name: com.example.groceryapp

Save location: E:\Android dev\GroceryApp

Language: Kotlin

Minimum SDK: API 21: Android 5.0 (Lollipop)

Your app will run on approximately **98.8%** of devices.
Help me choose

□ Use legacy android.support libraries
Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

⚠ project location should not contain whitespace, as this can cause problems with the NDK tools.

- **Add libraries in the gradle file and also apply the plugin as ‘kotlin-kapt’** To add these library go to **Gradle Scripts > build.gradle(Module:app).**

```
plugins {
    id 'com.android.application'
    id 'org.jetbrains.kotlin.android'
    id 'kotlin-kapt'
}

android {
    compileSdk 32

    defaultConfig {
        applicationId "com.example.grocerryapp"
        minSdk 16
        targetSdk 32
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-

        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    packagingOptions {
        exclude 'META-INF/atomicfu.kotlin_module'
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
}

dependencies {
    implementation "androidx.appcompat:appcompat:1.5.1"
    implementation "androidx.activity:activity-ktx:1.5.1"

    // Dependencies for working with Architecture components
    // You'll probably have to update the version numbers in build.gradle (Project)

    // Room components
    implementation "androidx.room:room-ktx:2.4.3"
    kapt "androidx.room:room-compiler:2.4.3"
    androidTestImplementation "androidx.room:room-testing:2.4.3"

    // Lifecycle components
    implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:2.5.1"
    implementation "androidx.lifecycle:lifecycle-livedata-ktx:2.5.1"
    implementation "androidx.lifecycle:lifecycle-common-java8:2.5.1"

    // Kotlin components
//    implementation 'org.jetbrains.kotlin:kotlin-stdlib-jdk7:1.7.10'
    api "org.jetbrains.kotlinx:kotlinx-coroutines-core:$rootProject.coroutines"
    api "org.jetbrains.kotlinx:kotlinx-coroutines-android:$rootProject.coroutines"

    // UI
    implementation "androidx.constraintlayout:constraintlayout:2.1.4"
    implementation "com.google.android.material:material:1.6.1"

    // Testing
    testImplementation "junit:junit:$rootProject.junitVersion"
    androidTestImplementation "androidx.arch.core:core-testing:$rootProject.coreTestingVersion"
    androidTestImplementation ("androidx.test.espresso:espresso-

        exclude group: 'com.android.support', module: 'support-annotations'
    })
    androidTestImplementation "androidx.test.ext:junit:$rootProject.androidxJunitVersion"
}
```

➤ ***Implement room database***

## a) Entities class

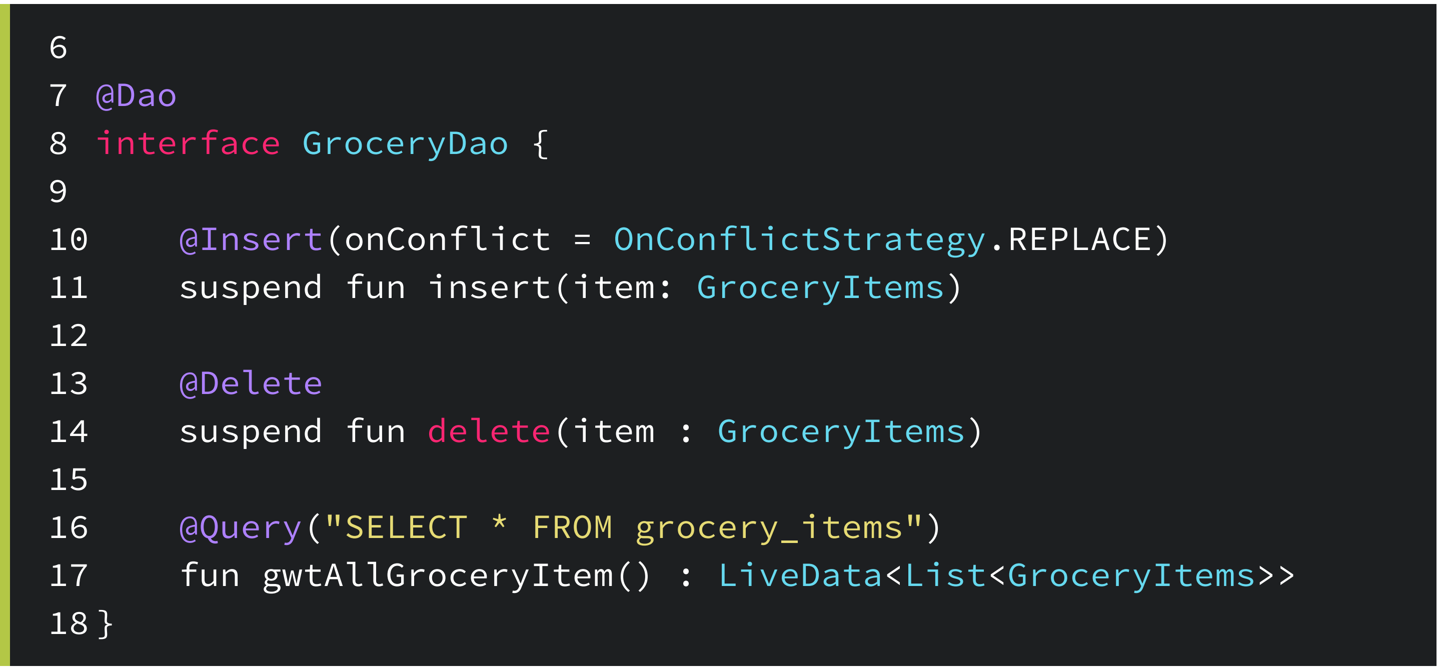
The entities class contains all the columns in the database and it should be annotated with @Entity(tablename = “Name of table")

```
package com.example.grocerryapp

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 : Int,

    @ColumnInfo(name = "itemPrice")
    var itemPrice : Int,

){

    @PrimaryKey(autoGenerate = true)
    var id:Int? = null
}
```

### b) Dao Interface

The Dao is an interface in which we create all the functions that we want to implement on the database.

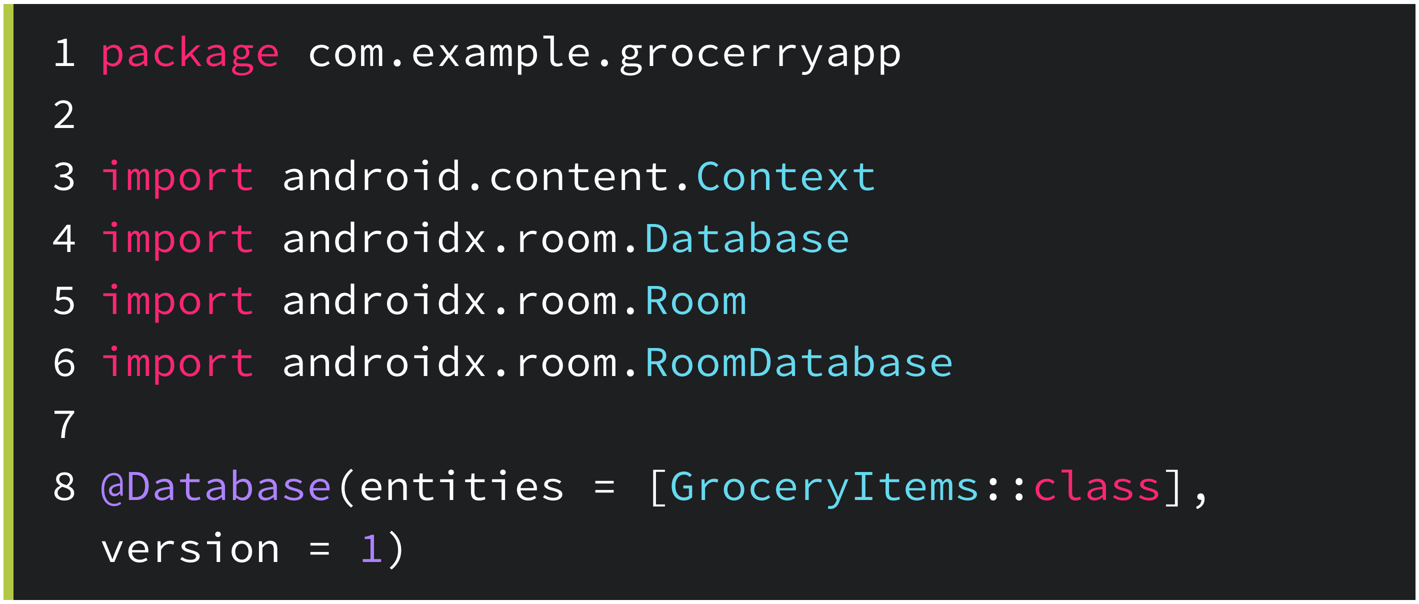
```
package com.example.grocerryapp


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



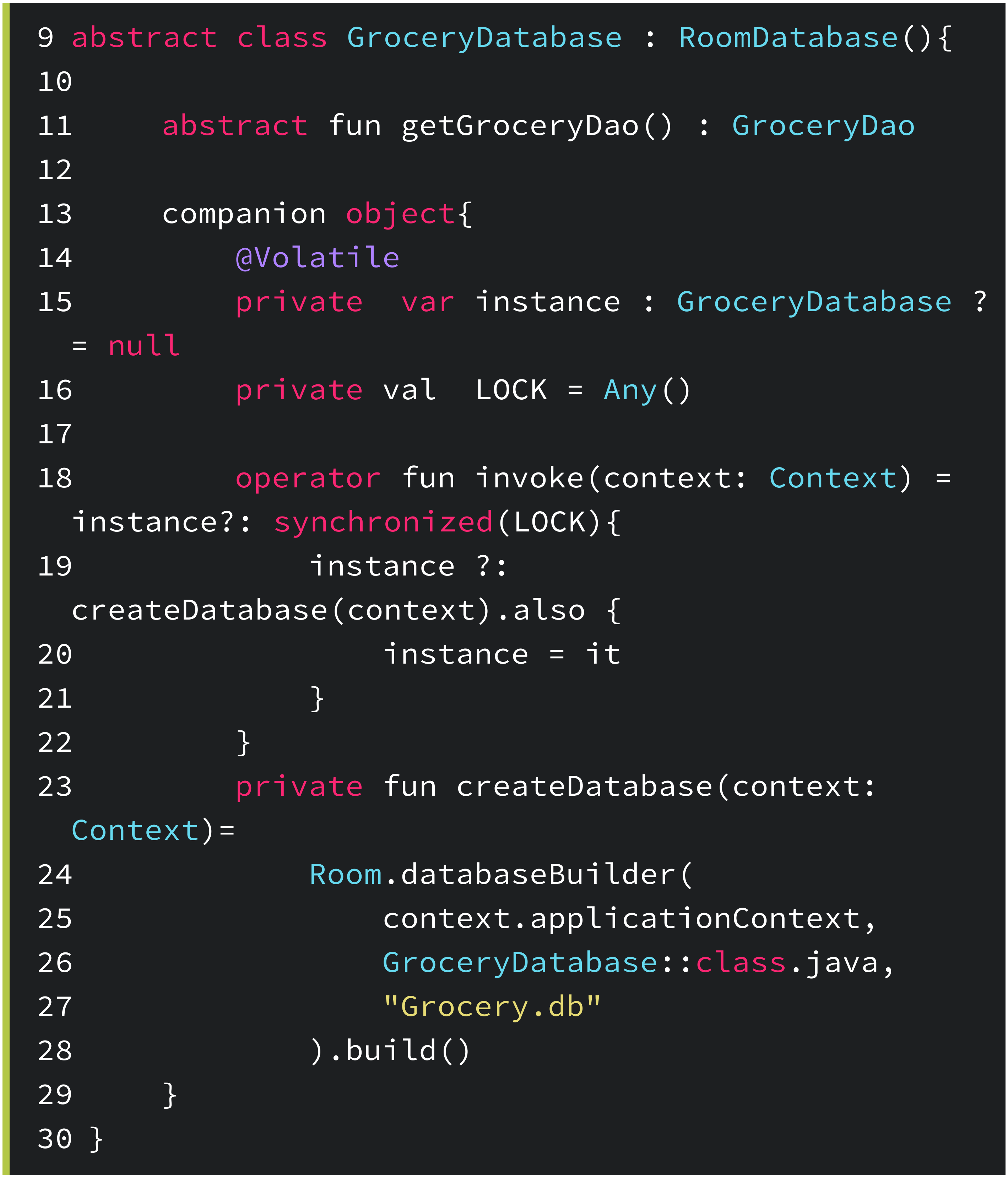
```

@Dao
interface GroceryDao {

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insert(item: GroceryItems)

    @Delete
    suspend fun delete(item : GroceryItems)

    @Query("SELECT * FROM grocery_items")
    fun gwtAllGroceryItem() : LiveData<List<GroceryItems>>
}
```

### 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.



```
package com.example.grocerryapp

import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase

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



```kotlin
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,
                "Grocery.db"
            ).build()
    }
}
```

## ➤ 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

```
package com.example.grocerryapp

data 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().gwtAllGroceryItem()
}
```

### b) ViewModel class

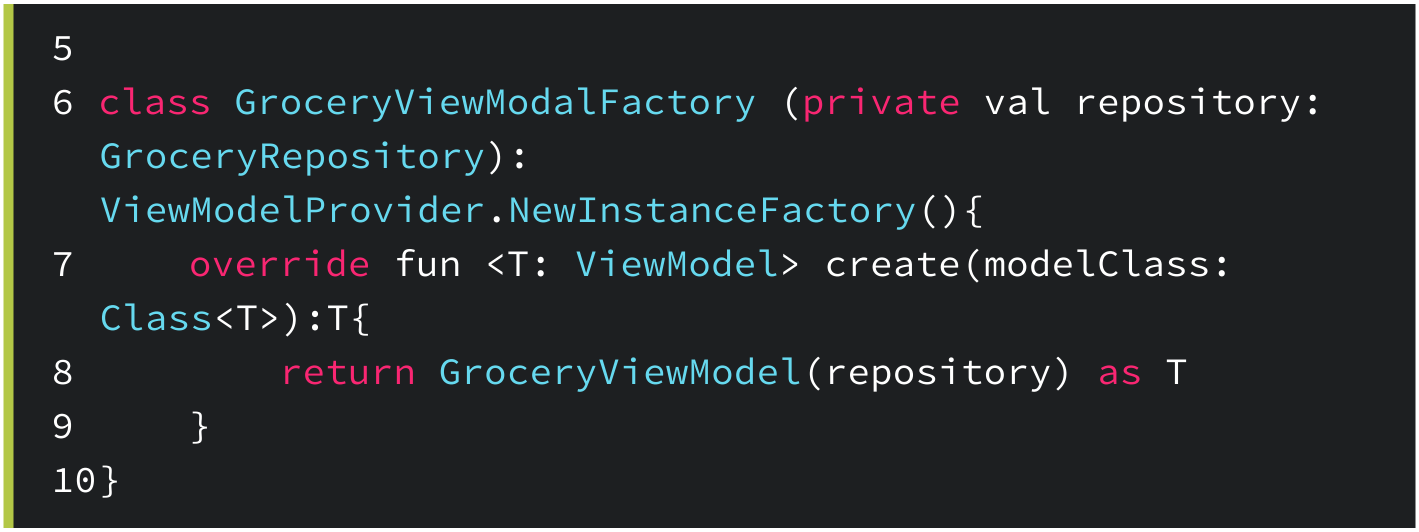
ViewModel class used as an interface between View and Data. Grocery View Model class inherit from View Model class and we will pass constructor value by creating instance variable of Repository class and stored in repository variable. As we pass the constructor in View Model we have to create another class which is a Factory View Model class.

```
package com.example.grocerryapp

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()
}
```

### c) Factory ViewModel class

We will inherit the Grocery ViewModel Factory class from ViewModelProvider.NewInstanceFactory and again pass constructor value by creating instance variable of Grocery Repository and return GroceryViewModel(repository)

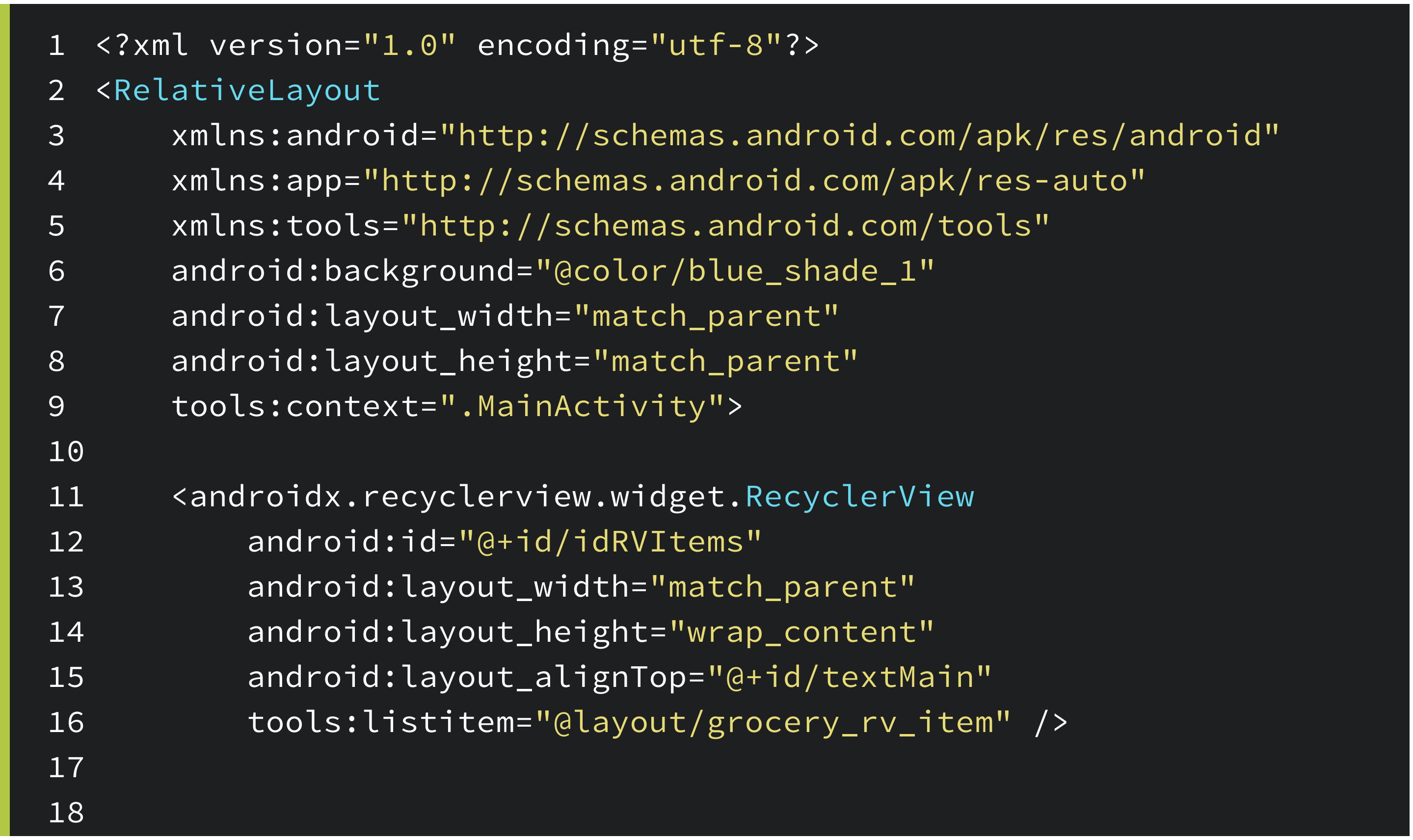
```
package com.example.grocerryapp

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



```kotlin

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

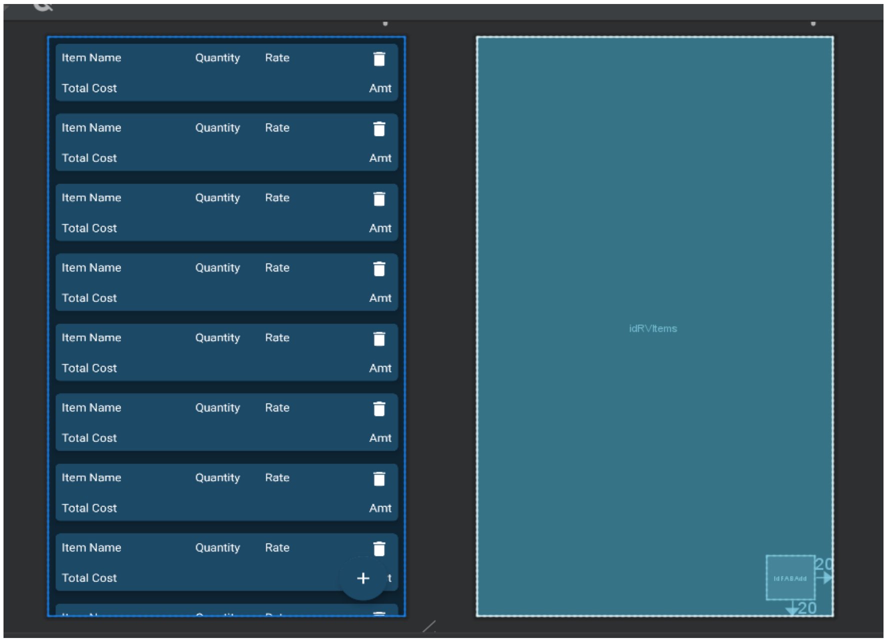
### ➤ Now let’s jump into the UI part

- **activity_main.xml -** Add RecyclerView and FloatingActionButton after clicking this button a **DialogBox** open and in that dialog box user can enter the item name, item quantity, and item price



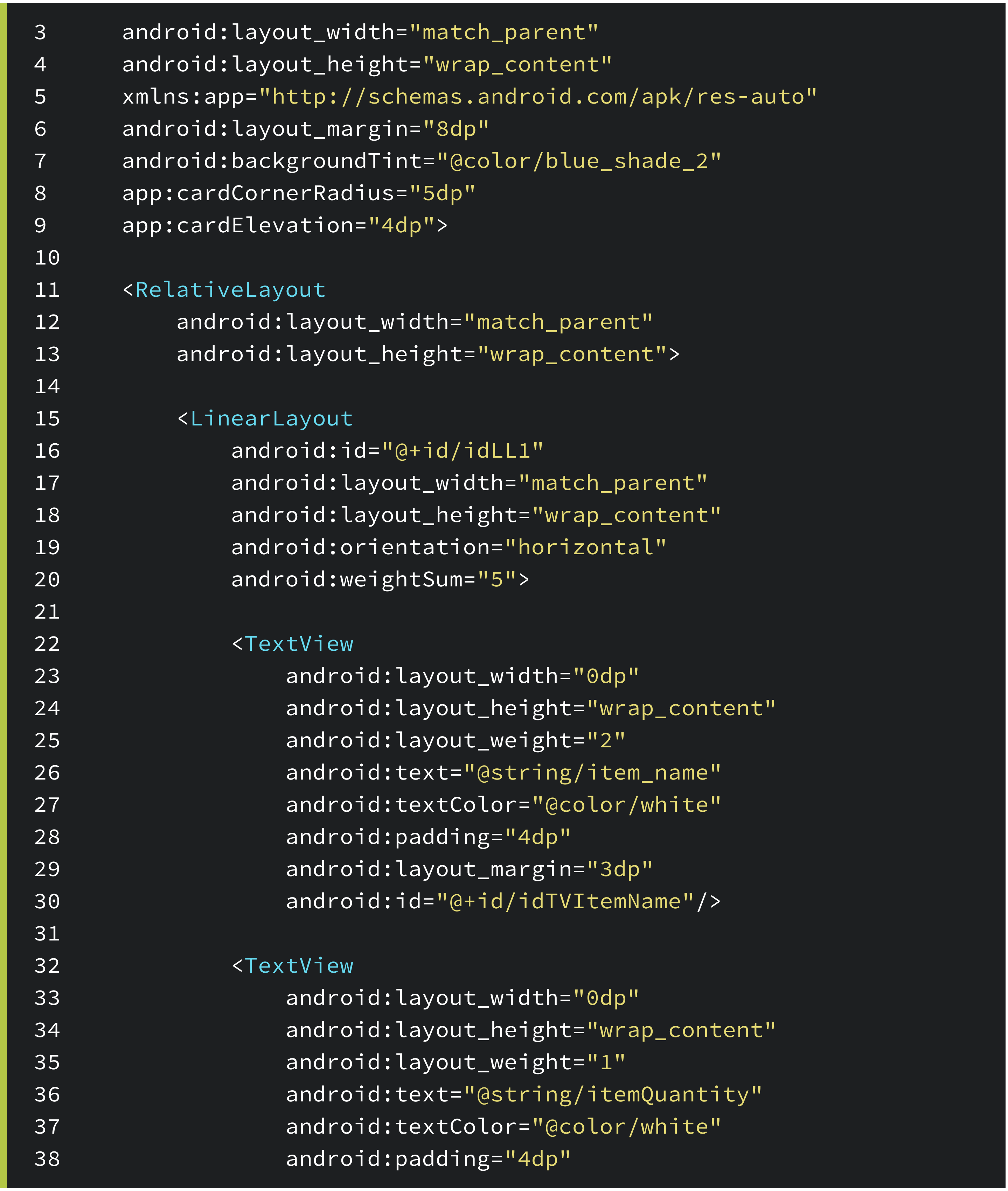
```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:background="@color/blue_shade_1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/idRVItems"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/textMain"
        tools:listitem="@layout/grocery_rv_item" />


```

```
<com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/idFABAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_margin="20dp"
        app:backgroundTint="@color/blue_shade_2"
        android:layout_alignParentRight="true"
        android:src="@drawable/ic_add"
        app:tint="@color/white"/>

</RelativeLayout>
```

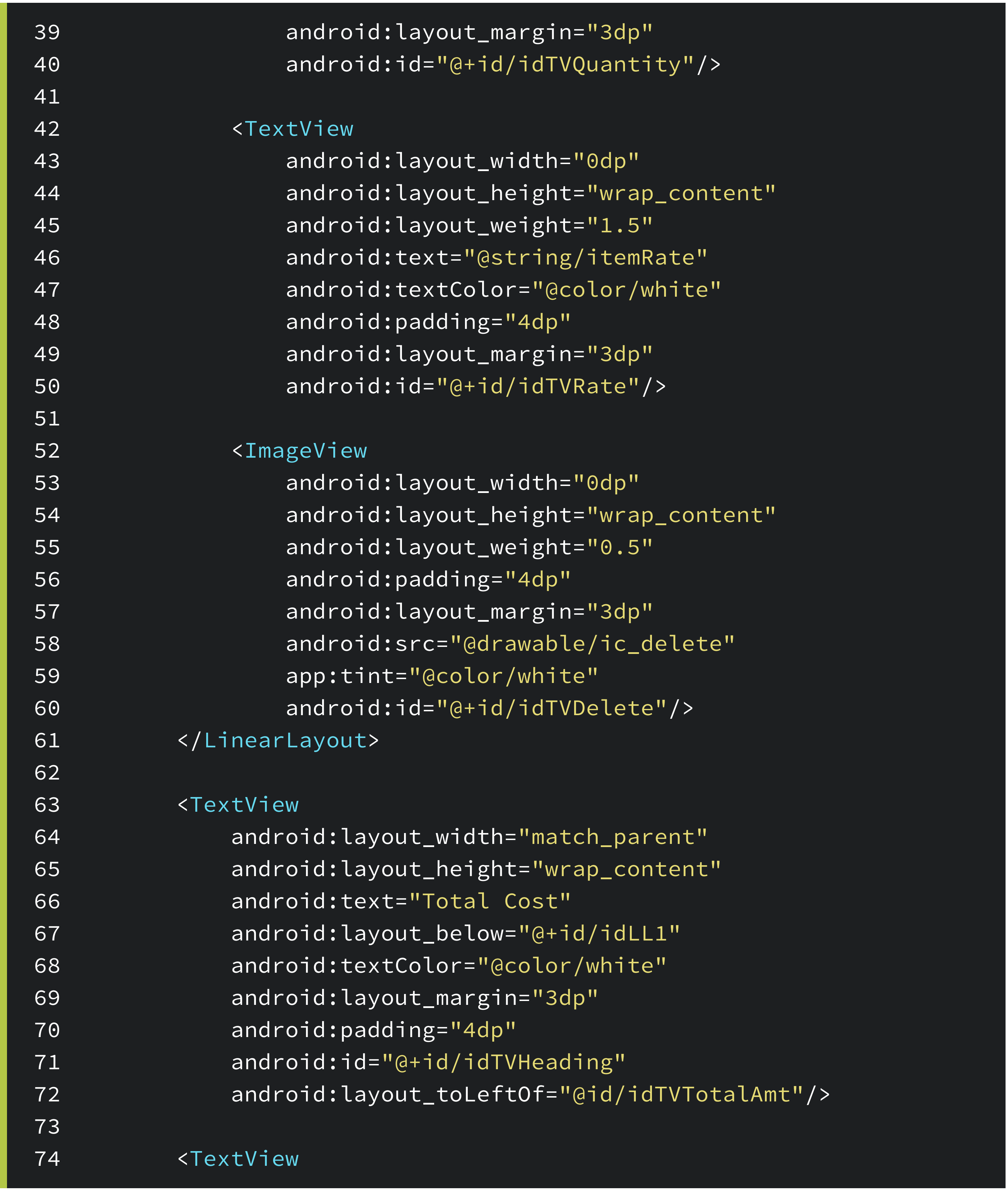


- **RecyclerView --** Now we will code the UI part of the row in the list.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"
```

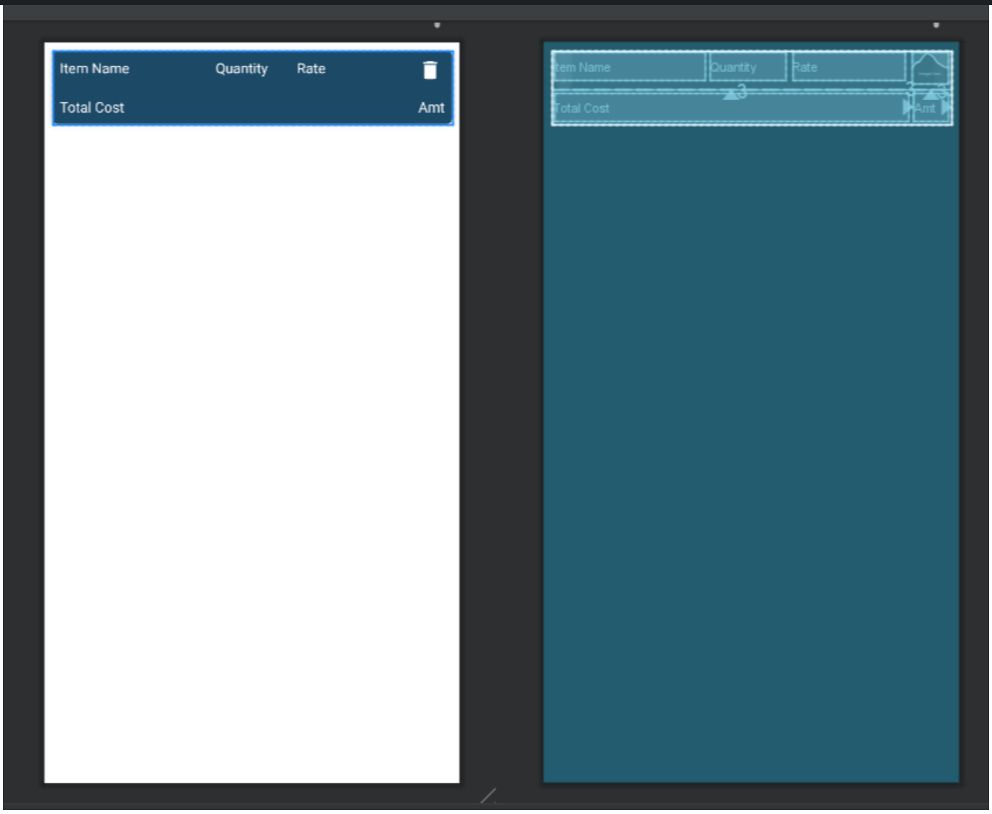


```xml
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_margin="8dp"
    android:backgroundTint="@color/blue_shade_2"
    app:cardCornerRadius="5dp"
    app:cardElevation="4dp">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <LinearLayout
            android:id="@+id/idLL1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:weightSum="5">

            <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="2"
                android:text="@string/item_name"
                android:textColor="@color/white"
                android:padding="4dp"
                android:layout_margin="3dp"
                android:id="@+id/idTVItemName"/>

            <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="@string/itemQuantity"
                android:textColor="@color/white"
                android:padding="4dp"
```



```xml
                android:layout_margin="3dp"
                android:id="@+id/idTVQuantity"/>

            <TextView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1.5"
                android:text="@string/itemRate"
                android:textColor="@color/white"
                android:padding="4dp"
                android:layout_margin="3dp"
                android:id="@+id/idTVRate"/>

            <ImageView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="0.5"
                android:padding="4dp"
                android:layout_margin="3dp"
                android:src="@drawable/ic_delete"
                app:tint="@color/white"
                android:id="@+id/idTVDelete"/>
        </LinearLayout>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Total Cost"
            android:layout_below="@+id/idLL1"
            android:textColor="@color/white"
            android:layout_margin="3dp"
            android:padding="4dp"
            android:id="@+id/idTVHeading"
            android:layout_toLeftOf="@id/idTVTotalAmt"/>

        <TextView
```

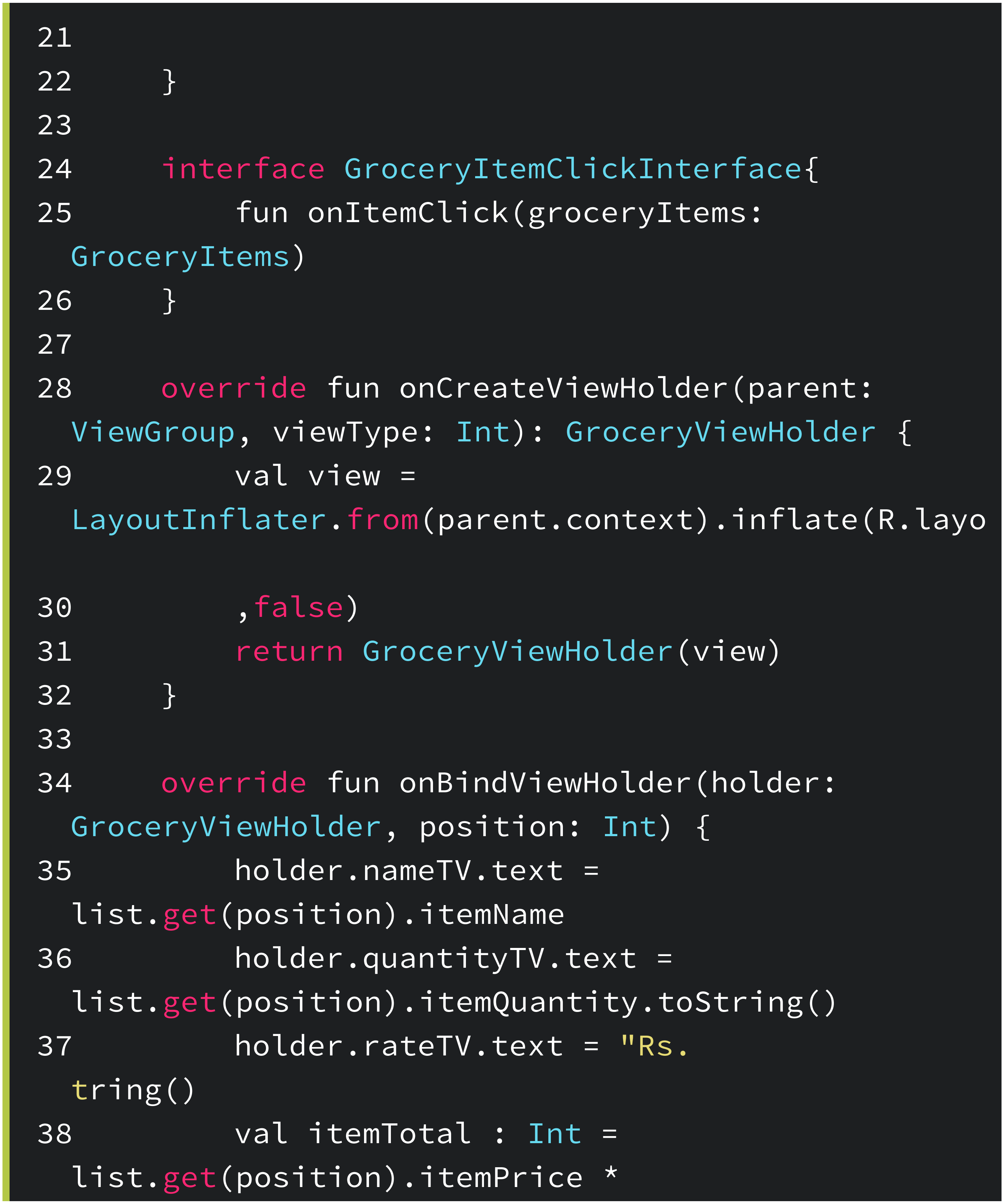
```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Amt"
            android:layout_below="@id/idLL1"
            android:textColor="@color/white"
            android:layout_margin="3dp"
            android:padding="4dp"
            android:id="@+id/idTVTotalAmt"
            android:layout_alignParentEnd="true"
            android:layout_alignParentRight="true"/>

    </RelativeLayout>

</androidx.cardview.widget.CardView>
```



***Adapter class for recycler view.***

```
package com.example.grocerryapp

```

```
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 : GroceryItemClickInterface)
    : RecyclerView.Adapter<GroceryRVAdapter.GroceryViewH

    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 amountTV = itemView.findViewById<TextView>(R.id.idTVTotalAmt)
        val deleteTV = itemView.findViewById<ImageView>(R.id.idTVDelete)
```

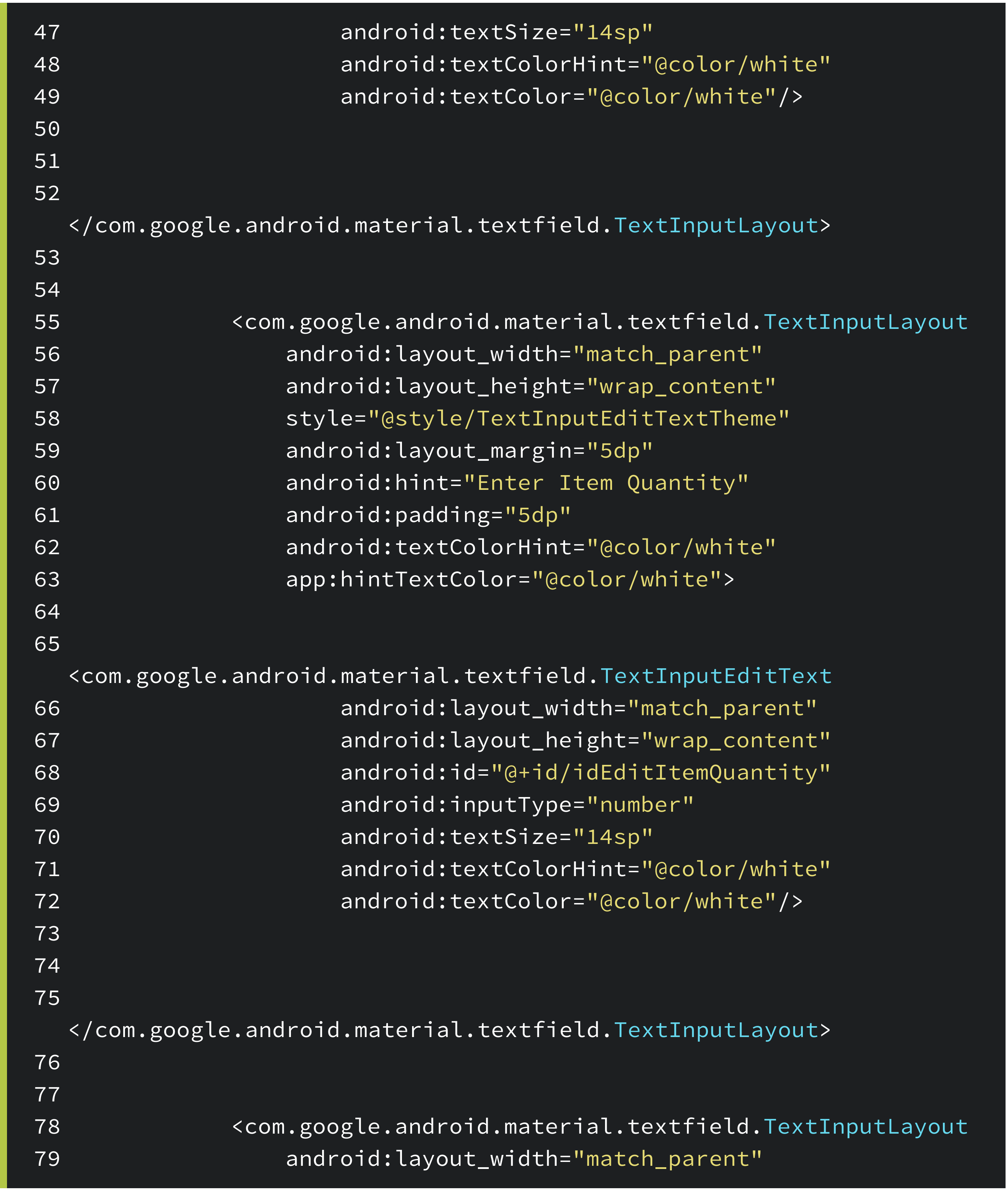


```

    }

    interface GroceryItemClickInterface{
        fun onItemClick(groceryItems: GroceryItems)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): GroceryViewHolder {
        val view = LayoutInflater.from(parent.context).inflate(R.layo

        ,false)
        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 = "Rs.
tring()
        val itemTotal : Int = list.get(position).itemPrice *
```

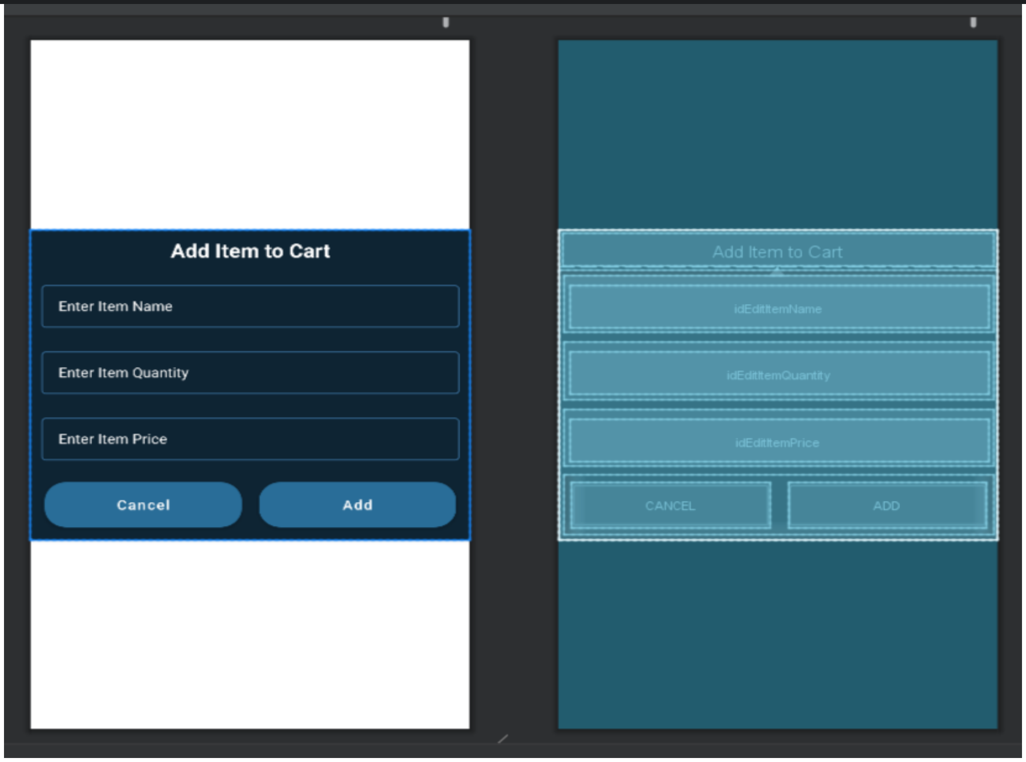
```
list.get(position).itemQuantity
        holder.amountTV.text = "Rs.

        holder.deleteTV.setOnClickListener {
groceryItemClickInterface.onItemClick(list.get(pos

        }

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

- 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**

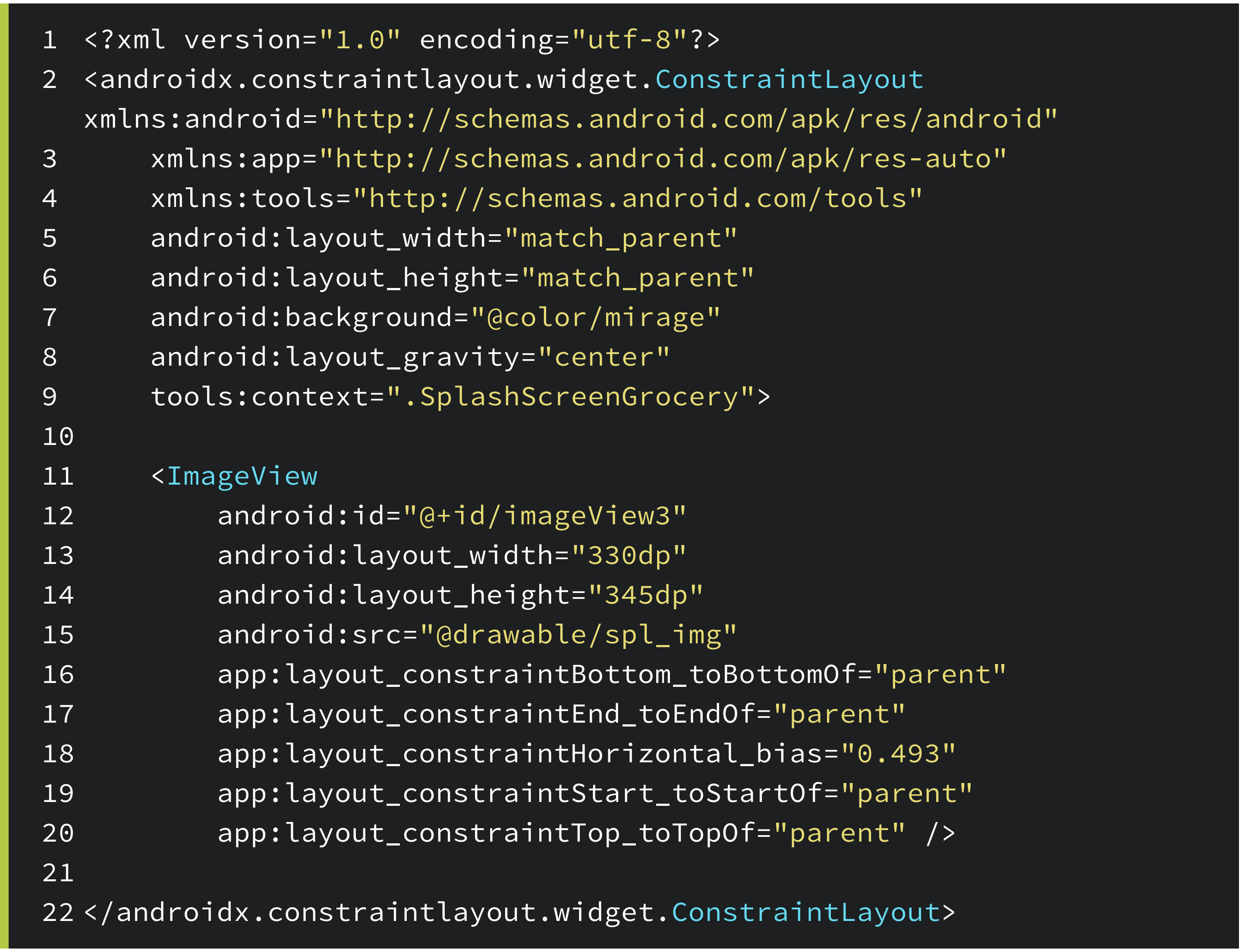
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_gravity="center">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/blue_shade_1">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Add Item to Cart"
            android:textAllCaps="false"
            android:textAlignment="center"
            android:gravity="center"
            android:textColor="@color/white"
            android:textStyle="bold"
            android:textSize="20sp"
            android:padding="4dp"
            android:layout_margin="4dp"
            android:id="@+id/idTVHeading"/>
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:layout_below="@+id/idTVHeading">

            <com.google.android.material.textfield.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                style="@style/TextInputEditTextTheme"
                android:layout_margin="5dp"
                android:hint="Enter Item Name"
                android:padding="5dp"
                android:textColorHint="@color/white"
                app:hintTextColor="@color/white">


<com.google.android.material.textfield.TextInputEditText
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:id="@+id/idEditItemName"
                    android:inputType="text"
```



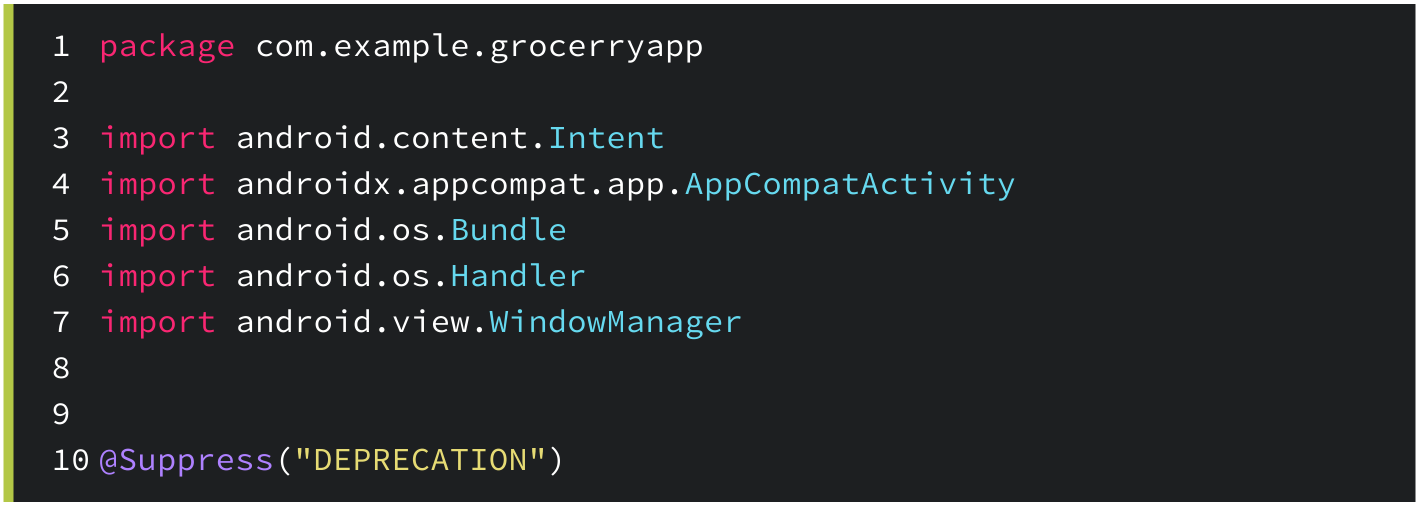
```
                    android:textSize="14sp"
                    android:textColorHint="@color/white"
                    android:textColor="@color/white"/>



</com.google.android.material.textfield.TextInputLayout>


            <com.google.android.material.textfield.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                style="@style/TextInputEditTextTheme"
                android:layout_margin="5dp"
                android:hint="Enter Item Quantity"
                android:padding="5dp"
                android:textColorHint="@color/white"
                app:hintTextColor="@color/white">


<com.google.android.material.textfield.TextInputEditText
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:id="@+id/idEditItemQuantity"
                    android:inputType="number"
                    android:textSize="14sp"
                    android:textColorHint="@color/white"
                    android:textColor="@color/white"/>



</com.google.android.material.textfield.TextInputLayout>


            <com.google.android.material.textfield.TextInputLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                style="@style/TextInputEditTextTheme"
                android:layout_margin="5dp"
                android:hint="Enter Item Price"
                android:padding="5dp"
                android:textColorHint="@color/white"
                app:hintTextColor="@color/white">

                <com.google.android.material.textfield.TextInputEditText
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:id="@+id/idEditItemPrice"
                    android:inputType="number"
                    android:textSize="14sp"
                    android:textColorHint="@color/white"
                    android:textColor="@color/white"/>

            </com.google.android.material.textfield.TextInputLayout>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal"
                android:layout_margin="5dp"
                android:weightSum="2">

                <Button
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:layout_margin="8dp"
                    android:id="@+id/idBtnCancle"
                    android:text="Cancel"
                    android:background="@drawable/custom_button_back"
                    android:textAllCaps="false"/>

                <Button
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:layout_margin="8dp"
                    android:id="@+id/idBtnAdd"
                    android:text="Add"
                    android:background="@drawable/custom_button_back"
                    android:textAllCaps="false"/>

            </LinearLayout>
        </LinearLayout>
    </RelativeLayout>

</androidx.cardview.widget.CardView>
```



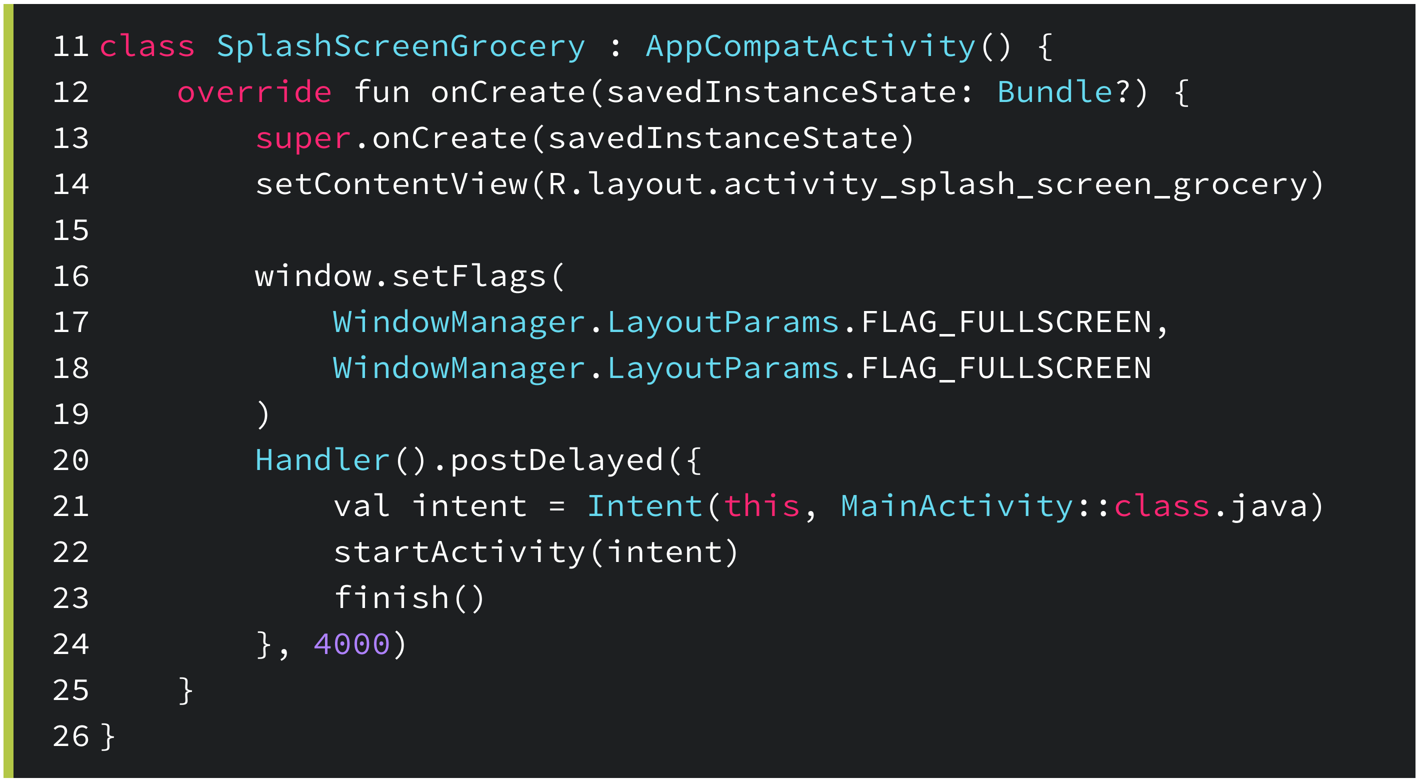
- **Splash Screen**



```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/mirage"
    android:layout_gravity="center"
    tools:context=".SplashScreenGrocery">

    <ImageView
        android:id="@+id/imageView3"
        android:layout_width="330dp"
        android:layout_height="345dp"
        android:src="@drawable/spl_img"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.493"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

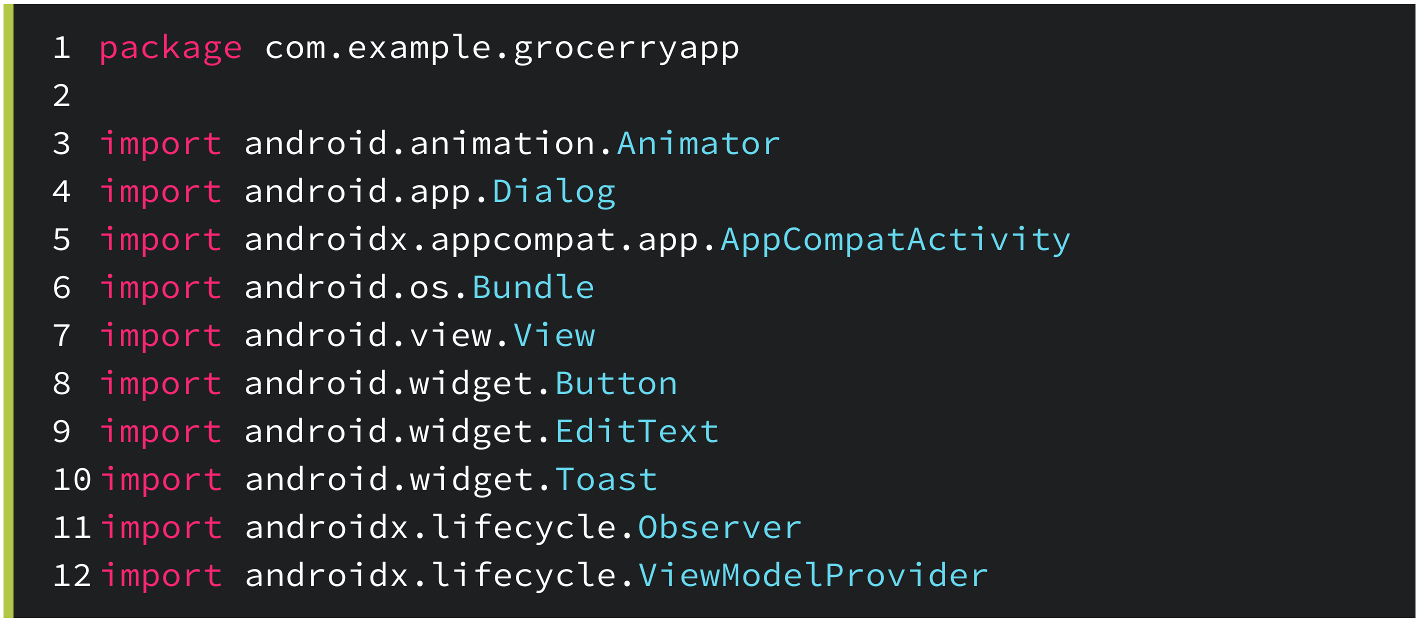


```kotlin
package com.example.grocerryapp

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.os.Handler
import android.view.WindowManager


@Suppress("DEPRECATION")
```

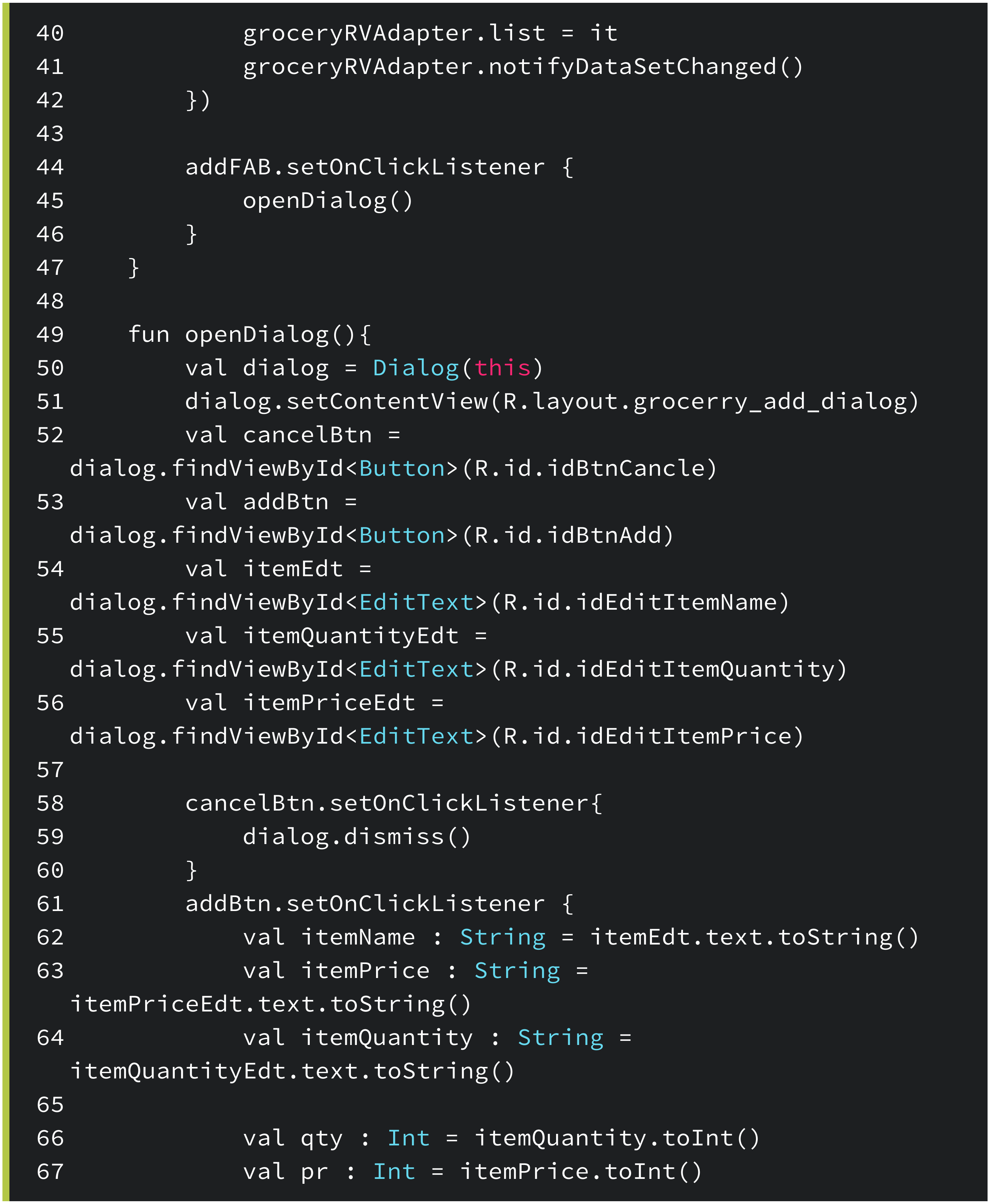


```kotlin
class SplashScreenGrocery : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_splash_screen_grocery)

        window.setFlags(
            WindowManager.LayoutParams.FLAG_FULLSCREEN,
            WindowManager.LayoutParams.FLAG_FULLSCREEN
        )
        Handler().postDelayed({
            val intent = Intent(this, MainActivity::class.java)
            startActivity(intent)
            finish()
        }, 4000)
    }
}
```

- **In this step finally we will code in our MainActivity. In our main activity, we have to set up the recycler view and add click listener on add button to open the dialog box.**

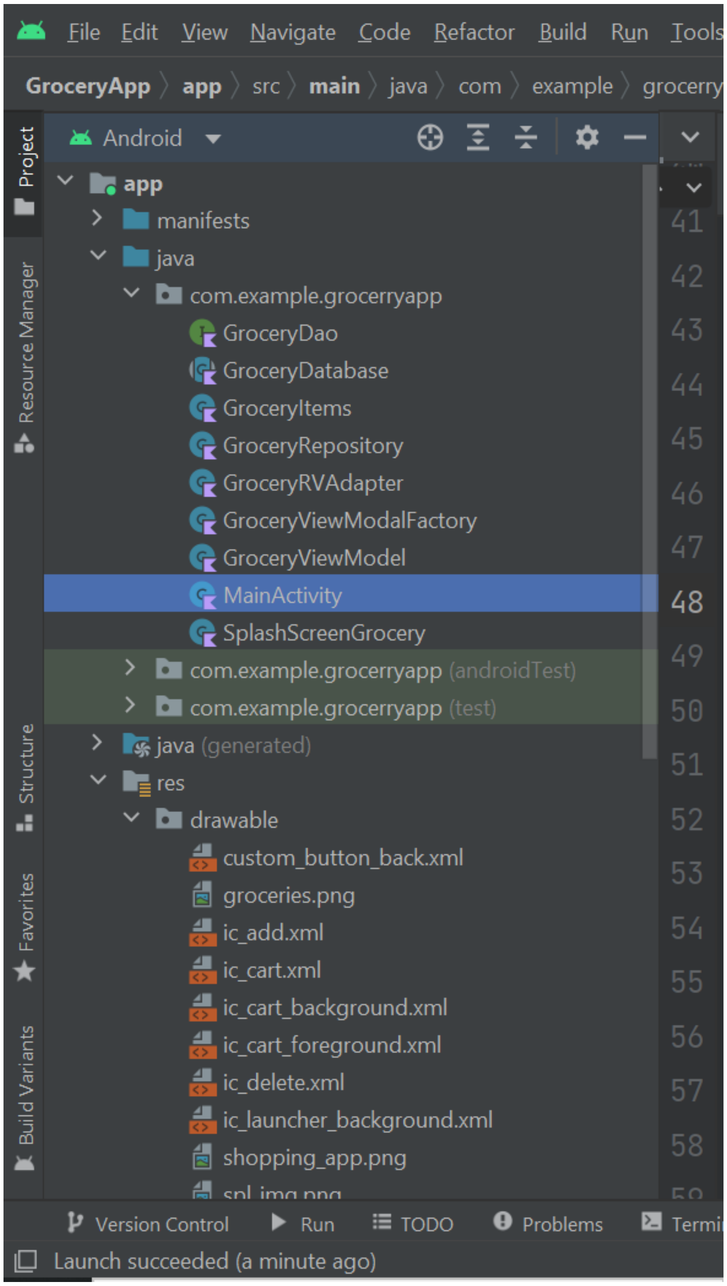


```kotlin
package com.example.grocerryapp

import android.animation.Animator
import android.app.Dialog
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
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

class MainActivity : AppCompatActivity(), GroceryRVAdapter.GroceryItemClickInterface {
    lateinit var itemsRv : 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)

        itemsRv = findViewById(R.id.idRVItems)
        addFAB = findViewById(R.id.idFABAdd)

        list = ArrayList<GroceryItems>()
        groceryRVAdapter = GroceryRVAdapter(list,this)
        itemsRv.layoutManager = LinearLayoutManager(this)
        itemsRv.adapter = groceryRVAdapter

        val groceryRepository = GroceryRepository(GroceryDatabase(this))
        val factory = GroceryViewModalFactory(groceryRepository)
        groceryViewModel = ViewModelProvider(this,factory).get(GroceryViewModel::class.ja

        groceryViewModel.getAllGroceryItems().observe(this, Observer{
```



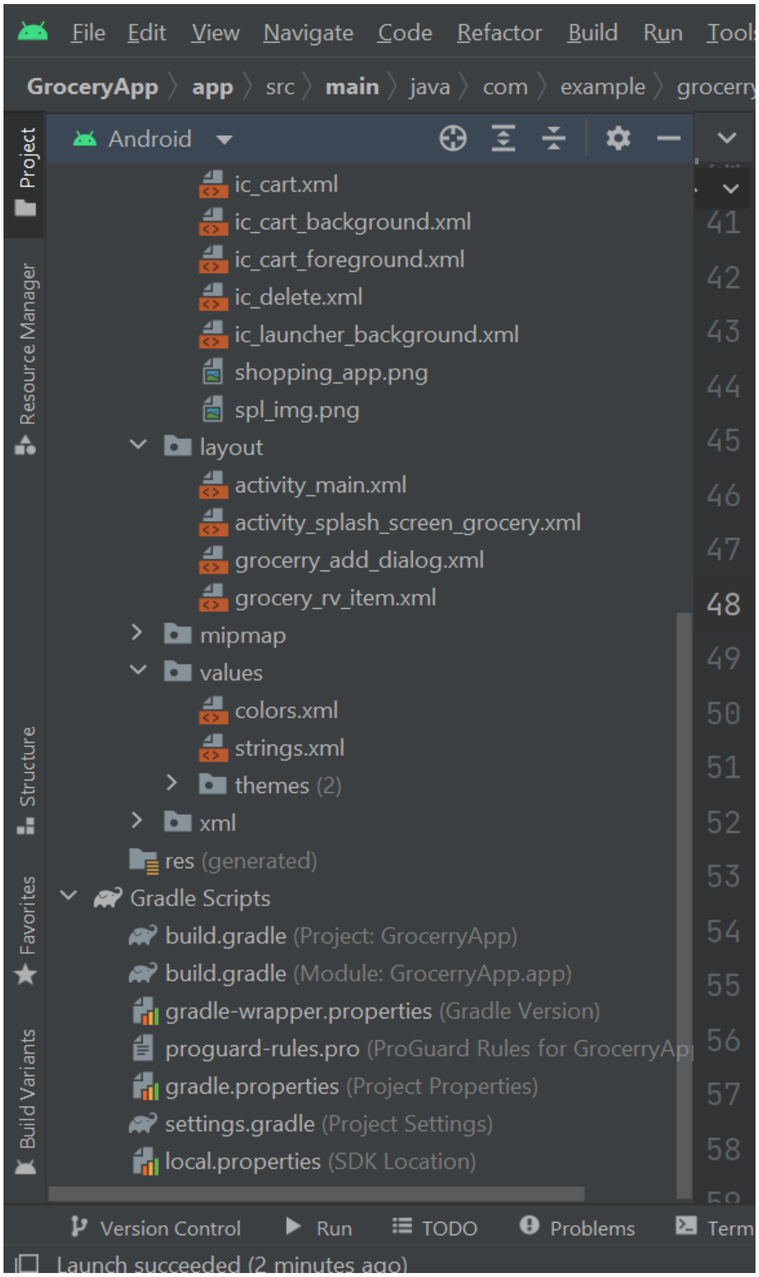
```
            groceryRVAdapter.list = it
            groceryRVAdapter.notifyDataSetChanged()
        })

        addFAB.setOnClickListener {
            openDialog()
        }
    }

    fun openDialog(){
        val dialog = Dialog(this)
        dialog.setContentView(R.layout.grocerry_add_dialog)
        val cancelBtn = dialog.findViewById<Button>(R.id.idBtnCancle)
        val addBtn = dialog.findViewById<Button>(R.id.idBtnAdd)
        val itemEdt = dialog.findViewById<EditText>(R.id.idEditItemName)
        val itemQuantityEdt = dialog.findViewById<EditText>(R.id.idEditItemQuantity)
        val itemPriceEdt = dialog.findViewById<EditText>(R.id.idEditItemPrice)

        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 : Int = itemQuantity.toInt()
            val pr : Int = itemPrice.toInt()

            if(itemName.isNotEmpty() && itemPrice.isNotEmpty() && itemQuantity.isNotEmpty()){
                val items = GroceryItems(itemName,qty,pr)
                groceryViewModel.insert(items)
                Toast.makeText(applicationContext, "Item

                groceryRVAdapter.notifyDataSetChanged()
                dialog.dismiss()
            }else{
                Toast.makeText(
                    applicationContext,
                    "Please Enter all the data..",
                    Toast.LENGTH_SHORT
                ).show()
            }
        }

        dialog.show()
    }
    override fun onItemClick(groceryItems: GroceryItems) {
        groceryViewModel.delete(groceryItems)
        groceryRVAdapter.notifyDataSetChanged()
        Toast.makeText(applicationContext,"Item

    }
}
```

- ***This is how the complete project structure looks like.***


File Edit View Navigate Code Refactor Build Run Tools
GroceryApp 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 grocerry
Android
Project
Resource Manager
Structure
Favorites
Build Variants
Version Control
Run
TODO
Problems
Termi

```
app
  manifests
  java
    com.example.groceryapp
      GroceryDao
      GroceryDatabase
      GroceryItems
      GroceryRepository
      GroceryRVAdapter
      GroceryViewModalFactory
      GroceryViewModel
      MainActivity
      SplashScreenGrocery
    com.example.groceryapp (androidTest)
    com.example.groceryapp (test)
  java (generated)
  res
    drawable
      custom_button_back.xml
      groceries.png
      ic_add.xml
      ic_cart.xml
      ic_cart_background.xml
      ic_cart_foreground.xml
      ic_delete.xml
      ic_launcher_background.xml
      shopping_app.png
      spl_img.png
```

Launch succeeded (a minute ago)


File Edit View Navigate Code Refactor Build Run Tool
GroceryApp 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 grocerr
Android
Project
Resource Manager
Structure
Favorites
Build Variants
Version Control
Run
TODO
Problems
Term

```
            ic_cart.xml
            ic_cart_background.xml
            ic_cart_foreground.xml
            ic_delete.xml
            ic_launcher_background.xml
            shopping_app.png
            spl_img.png
        layout
            activity_main.xml
            activity_splash_screen_grocery.xml
            grocerry_add_dialog.xml
            grocery_rv_item.xml
        mipmap
        values
            colors.xml
            strings.xml
            themes (2)
        xml
    res (generated)
Gradle Scripts
    build.gradle (Project: GrocerryApp)
    build.gradle (Module: GrocerryApp.app)
    gradle-wrapper.properties (Gradle Version)
    proguard-rules.pro (ProGuard Rules for GrocerryAp
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

Launch succeeded (2 minutes ago)