MINI – PROJECT
Android App Development
REPORT

on

Sub Shop

Submitted by

Ritikesh Kumar: 12114314
Ujjawal Kumar Gupta :12101047
Keshav Jha: 12100224

Program Name: B.Tech.(CSE)

Under the Guidance of

Dr. Om Prakash Yadav
Associate Professor

School of Computer Science and Engineering
Lovely Professional University. Phagwara,
Punjab, India

Description of Project

Sub Shop: Auto-Renew Essentials Marketplace

It is an innovative, user-centric e-commerce platform designed to simplify the way consumers manage and receive essential household items. The core objective of Sub Shop is to eliminate the inconvenience of running out of everyday necessities by offering a seamless **subscription-based delivery service**. Users can browse through a curated marketplace of essential products including toiletries, groceries, personal care items, and household supplies and opt into recurring deliveries based on their usage patterns and preferences.

Key Features:

- **Subscription-Based Delivery**
  Users can subscribe to essential items with customizable delivery intervals (weekly, bi-weekly, monthly, etc.) to ensure timely restocking without manual reordering.
- **Curated Essentials Marketplace**
  A well-organized catalog of high-quality household essentials, including groceries, toiletries, personal care, and cleaning supplies.
- **Flexible Delivery Date and Location Selection**
  Users can choose the **preferred date and delivery location** for each order or subscription cycle, allowing for maximum flexibility and convenience.
- **Multi-Item Subscription Bundles**
  Users can group multiple products into a single subscription bundle with shared delivery schedules, reducing packaging and shipping costs.
- **Custom Quantity Selection**
  Users can select the **exact quantity** of each product as per their requirements during subscription setup or manual ordering.
- **Mobile-Responsive and Intuitive UI**
  A clean, responsive interface ensures users can browse, subscribe, and manage orders effortlessly across devices.

Technical Highlights:

- **Platform & Language: Kotlin (Android)**
  The application is developed using Kotlin, the modern, concise, and powerful language for Android app development. It enables better performance, cleaner syntax, and seamless integration with Android SDKs.
- **User Interface: XML Layouts**
  The front-end user interface is designed using XML, following Android's UI framework. It ensures responsive, accessible, and device-compatible layouts, providing smooth user experience across various screen sizes.
- **Flexible Date and Location Selection**
  Users can select their preferred delivery date and location via custom date pickers and interactive map inputs.
- **Maps API Integration**
  Integrated with **Google Maps API** (or similar), enabling: Real-time location selection for delivery address-tagging and address autocomplete. Map-based UI for choosing delivery locations visually.

System Requirement

I. Hardware System Requirement:

1. **Processor:**

- ARM Cortex-A53 or better (Octa-core, 1.5 GHz or higher).

2. **RAM:**

- 2 GB or more for better multitasking and app stability.

3. **Operating System:**

- Android 6.0 (Marshmallow) or later for enhanced performance and security.

4. **Sensors and Hardware (if required by the app):**

- Touchscreen with multi-touch support.

II. Software System Requirement:

For Target Devices

1. **Operating System:**
   - Android 6.0 (Marshmallow) or higher
2. **API Level:**
   - API Level 21 or above to ensure compatibility
3. **Security and Permissions:**
   - Ensure proper configuration for runtime permissions (introduced in API 23) for target devices beyond Android 6.0.

Manifest code :-

Gradle.kts (Project)

```
plugins {
    alias(libs.plugins.android.application) apply false
    alias(libs.plugins.jetbrains.kotlin.android) apply false
    alias(libs.plugins.google.gms.google.services) apply false
}
```

Gradle.kts (Module)

```
plugins {
    alias(libs.plugins.android.application)
    alias(libs.plugins.jetbrains.kotlin.android)
    alias(libs.plugins.google.gms.google.services)
}

android {
    namespace = "com.shopping.subshop"
    compileSdk = 35

    defaultConfig {
        applicationId = "com.shopping.subshop"
        minSdk = 24
        targetSdk = 35
        versionCode = 2
        versionName = "1.0"
```

```
        testInstrumentationRunner =
"androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            isMinifyEnabled = false
            proguardFiles(
                getDefaultProguardFile("proguard-android-optimize.txt"),
                "proguard-rules.pro"
            )
        }
    }
    compileOptions {
        sourceCompatibility = JavaVersion.VERSION_1_8
        targetCompatibility = JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = "1.8"
    }
}

dependencies { // Use the latest version
    implementation("com.google.android.material:material:1.9.0")
    implementation ("com.github.bumptech.glide:glide:4.15.1")
    implementation ("com.google.android.libraries.places:places:3.3.0")
    implementation("com.google.android.gms:play-services-maps:18.2.0")
    implementation("com.google.android.libraries.places:places:3.3.0")
    implementation(libs.firebase.storage.ktx)
    annotationProcessor ("com.github.bumptech.glide:compiler:4.15.1")
    implementation ("com.google.firebase:firebase-auth:22.1.1")
    implementation ("com.google.firebase:firebase-firestore:24.9.1")
        implementation(libs.androidx.core.ktx)
        implementation(libs.androidx.appcompat)
        implementation(libs.material) // If managed via catalog, ensure it's
updated
        implementation(libs.androidx.activity)
        implementation(libs.androidx.constraintlayout)
    implementation(libs.firebase.auth)
    implementation(libs.firebase.firestore.ktx)
    implementation(libs.firebase.database)
    testImplementation(libs.junit)
```

```
    androidTestImplementation(libs.androidx.junit)
    androidTestImplementation(libs.androidx.espresso.core)
}
```

XML Code

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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:orientation="vertical"
    tools:context=".MainActivity"
    android:background="@color/red">

    <!-- Custom Toolbar/Navbar -->

    <!-- RecyclerView -->

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="@color/green"

android:theme="@style/ThemeOverlay.AppCompat.ActionBar"

app:popupTheme="@style/ThemeOverlay.AppCompat.Light"
        app:titleTextColor="@android:color/white">

        <!-- Logo -->

        <TextView
            android:id="@+id/tvTitle"
            android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:text="SubShop"
        android:textSize="24sp"
        android:textColor="@android:color/white"
        android:textStyle="bold"
        android:fontFamily="sans-serif-condensed"
      />

    <!-- Spacer to push login/signup to the right -->
    <View
        android:layout_width="150dp"
        android:layout_height="match_parent"
        android:layout_weight="1" />

    <Button
        android:id="@+id/btnAdmin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:background="?attr/selectableItemBackground"
        android:padding="8dp"
        android:text="Admin"
        android:textColor="@android:color/white" />
    <!-- Login Button -->
    <Button
        android:id="@+id/btnLogin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical"
        android:background="?attr/selectableItemBackground"
        android:padding="8dp"
        android:text="Login"
        android:textColor="@android:color/white" />

    <!-- Signup Button -->

</androidx.appcompat.widget.Toolbar>
```

```
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"

app:layoutManager="androidx.recyclerview.widget.GridLayout
Manager"
        tools:listitem="@layout/item_product"
        />

</LinearLayout>
```

.Kt Code

```
package com.shopping.subshop
import android.content.Intent
import android.os.Bundle
import android.util.Log
import android.widget.Button
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.GridLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.google.firebase.database.DataSnapshot
import com.google.firebase.database.DatabaseError
import com.google.firebase.database.DatabaseReference
import com.google.firebase.database.FirebaseDatabase
import com.google.firebase.database.ValueEventListener


class MainActivity : AppCompatActivity() {

    private lateinit var recyclerView: RecyclerView
    private lateinit var productList: ArrayList<Product>
    private lateinit var productAdapter: ProductAdapter
    private lateinit var database: DatabaseReference

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
```

```
        setContentView(R.layout.activity_main)
        var btnlogin = findViewById<Button>(R.id.btnLogin)
        var btnAdmin = findViewById<Button>(R.id.btnAdmin)

        btnlogin.setOnClickListener{
            startActivity(Intent(this, loginActivity::class.java))

        }

        btnAdmin.setOnClickListener{
            startActivity(Intent(this, Admin::class.java))

        }
        recyclerView = findViewById(R.id.recyclerView)
        recyclerView.layoutManager = GridLayoutManager(this,
2)  // 2 items per row
        productList = arrayListOf()

        database =
FirebaseDatabase.getInstance().getReference("Products")

        fetchProducts()
    }

    private fun fetchProducts() {
        database.addValueEventListener(object :
ValueEventListener {
            override fun onDataChange(snapshot: DataSnapshot) {
                productList.clear()
                for (productSnapshot in snapshot.children) {
                    val product =
productSnapshot.getValue(Product::class.java)
                    product?.let { productList.add(it) }
                }
                productAdapter = ProductAdapter(productList)
                recyclerView.adapter = productAdapter
            }
```

```
            override fun onCancelled(error: DatabaseError) {
                Log.e("Firebase", "Failed to load products:
${error.message}")
            }
        })
    }
}
```