FINAL CAPSTONE

APP: BOOKMYSLOT\_TAG

**Github link:** [**https://github.com/MdMastan/Finalcapstoneproject.git**](https://github.com/MdMastan/Finalcapstoneproject.git)

AndroidManifest

**<?xml version="1.0" encoding="utf-8"?>**

**<manifest xmlns:android="http://schemas.android.com/apk/res/android"**

**xmlns:tools="http://schemas.android.com/tools">**

**<application**

**android:allowBackup="true"**

**android:dataExtractionRules="@xml/data\_extraction\_rules"**

**android:fullBackupContent="@xml/backup\_rules"**

**android:icon="@mipmap/ic\_launcher"**

**android:label="@string/app\_name"**

**android:roundIcon="@mipmap/ic\_launcher\_round"**

**android:supportsRtl="true"**

**android:theme="@style/Theme.BookMySlot\_TAG"**

**tools:targetApi="31">**

**<activity**

**android:name=".MainActivity"**

**android:exported="true">**

**<intent-filter>**

**<action android:name="android.intent.action.MAIN" />**

**<category android:name="android.intent.category.LAUNCHER" />**

**</intent-filter>**

**</activity>**

**</application>**

**</manifest>**

**BUILD.GRADLE**

**plugins {**

**alias(libs.plugins.android.application)**

**alias(libs.plugins.kotlin.android)**

**id("androidx.navigation.safeargs")**

**id ("kotlin-parcelize")**

**}**

**android {**

**namespace = "com.example.bookmyslot\_tag"**

**compileSdk = 35**

**defaultConfig {**

**applicationId = "com.example.bookmyslot\_tag"**

**minSdk = 24**

**targetSdk = 34**

**versionCode = 1**

**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\_11**

**targetCompatibility = JavaVersion.VERSION\_11**

**}**

**kotlinOptions {**

**jvmTarget = "11"**

**}**

**viewBinding{**

**enable = true;**

**}**

**}**

**dependencies {**

**implementation(libs.androidx.core.ktx)**

**implementation(libs.androidx.appcompat)**

**implementation(libs.material)**

**implementation(libs.androidx.activity)**

**implementation(libs.androidx.constraintlayout)**

**testImplementation(libs.junit)**

**implementation("com.squareup.retrofit2:retrofit:2.9.0")**

**implementation("com.squareup.retrofit2:converter-gson:2.9.0")**

**implementation("androidx.navigation:navigation-fragment-ktx:2.5.3")**

**implementation ("androidx.recyclerview:recyclerview:1.2.1")**

**implementation ("androidx.room:room-runtime:2.5.0")**

**implementation("androidx.navigation:navigation-ui-ktx:2.5.3")**

**implementation("androidx.recyclerview:recyclerview:1.2.1")**

**androidTestImplementation(libs.androidx.junit)**

**androidTestImplementation(libs.androidx.espresso.core)**

**}**

**Activitymain 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:id="@+id/main"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**tools:context=".MainActivity">**

**<androidx.fragment.app.FragmentContainerView**

**android:id="@+id/nav\_host\_fragment"**

**android:name="androidx.navigation.fragment.NavHostFragment"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**app:navGraph="@navigation/nav\_graph"**

**app:defaultNavHost="true"/>**

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

**Dashboard xml**

**<?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"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:orientation="vertical"**

**android:padding="16dp">**

**<TextView**

**android:id="@+id/tvDashboardTitle"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Available Interview Slots"**

**android:textSize="18sp"**

**android:textStyle="bold"**

**android:gravity="center"**

**android:paddingBottom="10dp" />**

**<androidx.recyclerview.widget.RecyclerView**

**android:id="@+id/recyclerView"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"/>**

**</LinearLayout>**

**InterviewerDetailFragment**

**<?xml version="1.0" encoding="utf-8"?>**

**<LinearLayout**

**xmlns:android="http://schemas.android.com/apk/res/android"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:padding="16dp"**

**android:orientation="vertical">**

**<!-- Interviewer Name -->**

**<TextView**

**android:id="@+id/tvDetailName"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Interviewer Name"**

**android:textSize="18sp"**

**android:textStyle="bold" />**

**<!-- Interview Date -->**

**<TextView**

**android:id="@+id/tvDetailDate"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Date"**

**android:textSize="16sp" />**

**<!-- Interview Time -->**

**<TextView**

**android:id="@+id/tvDetailTime"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Time"**

**android:textSize="16sp" />**

**<!-- Interview Subject -->**

**<TextView**

**android:id="@+id/tvDetailSubject"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Subject"**

**android:textSize="16sp" />**

**<!-- Interview Status -->**

**<TextView**

**android:id="@+id/tvDetailStatus"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Status"**

**android:textSize="16sp"**

**android:textStyle="italic" />**

**<!-- Book Slot Button -->**

**<Button**

**android:id="@+id/btnBookSlot"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Book Slot" />**

**<!-- Release Slot Button -->**

**<Button**

**android:id="@+id/btnReleaseSlot"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Release Slot"**

**android:layout\_marginTop="8dp" />**

**</LinearLayout>**

**Login xml**

**<?xml version="1.0" encoding="utf-8"?>**

**<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:orientation="vertical"**

**android:gravity="center"**

**android:padding="20dp">**

**<EditText**

**android:id="@+id/etUsername"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:gravity="center"**

**android:hint="Username" />**

**<EditText**

**android:id="@+id/etPassword"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:gravity="center"**

**android:hint="Password"**

**android:inputType="textPassword" />**

**<Button**

**android:id="@+id/btnLogin"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Login" />**

**<TextView**

**android:id="@+id/tvRegister"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Don't have an account? Register"**

**android:textColor="@android:color/holo\_blue\_dark"**

**android:paddingTop="10dp"/>**

**</LinearLayout>**

**Register xml**

**<?xml version="1.0" encoding="utf-8"?>**

**<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**android:orientation="vertical"**

**android:gravity="center"**

**android:padding="20dp">**

**<EditText**

**android:id="@+id/etNewUsername"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:hint="New Username" />**

**<EditText**

**android:id="@+id/etNewEmail"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:hint="Enter Email"**

**android:padding="12dp"**

**android:inputType="textEmailAddress"**

**android:textSize="16sp"**

**android:layout\_marginTop="10dp" />**

**<EditText**

**android:id="@+id/etNewPassword"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:hint="New Password"**

**android:inputType="textPassword" />**

**<Button**

**android:id="@+id/btnRegister"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="Register" />**

**<TextView**

**android:id="@+id/tvBackToLogin"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Already have an account? Login"**

**android:textColor="@android:color/holo\_blue\_dark"**

**android:textSize="14sp"**

**android:layout\_marginTop="12dp"/>**

**</LinearLayout>**

**Item xml**

**<?xml version="1.0" encoding="utf-8"?>**

**<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"**

**android:layout\_width="match\_parent"**

**android:layout\_height="wrap\_content"**

**android:orientation="vertical"**

**android:padding="10dp"**

**android:background="@android:color/darker\_gray"**

**android:layout\_marginBottom="8dp">**

**<TextView**

**android:id="@+id/tvInterviewerName"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Interviewer Name"**

**android:textSize="16sp"**

**android:textStyle="bold"/>**

**<TextView**

**android:id="@+id/tvSubject"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Subject"**

**android:textSize="14sp"/>**

**<TextView**

**android:id="@+id/tvTime"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Time Slot"**

**android:textSize="14sp"/>**

**<TextView**

**android:id="@+id/tvDate"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Date"**

**android:textSize="14sp"/>**

**<TextView**

**android:id="@+id/tvStatus"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="Status"**

**android:textSize="14sp"/>**

**</LinearLayout>**

**mainActivity**

**package com.example.bookmyslot\_tag**

**import android.os.Bundle**

**import androidx.appcompat.app.AppCompatActivity**

**import androidx.navigation.findNavController**

**import androidx.navigation.fragment.NavHostFragment**

**import androidx.navigation.ui.setupActionBarWithNavController**

**class MainActivity : AppCompatActivity() {**

**override fun onCreate(savedInstanceState: Bundle?) {**

**super.onCreate(savedInstanceState)**

**setContentView(R.layout.activity\_main)**

**val navHostFragment = supportFragmentManager.findFragmentById(R.id.nav\_host\_fragment) as NavHostFragment**

**val navController = navHostFragment.navController**

**setupActionBarWithNavController(navController)**

**}**

**override fun onSupportNavigateUp(): Boolean {**

**val navController = findNavController(R.id.nav\_host\_fragment)**

**return navController.navigateUp() || super.onSupportNavigateUp()**

**}**

**}**

**adapter**

**interviewerAdapter**

**package com.example.bookmyslot\_tag.adapter**

**import android.view.LayoutInflater**

**import android.view.View**

**import android.view.ViewGroup**

**import android.widget.TextView**

**import androidx.recyclerview.widget.RecyclerView**

**import com.example.bookmyslot\_tag.R**

**import com.example.bookmyslot\_tag.model.Interviewer**

**class InterviewerAdapter(**

**private val interviewerList: List<Interviewer>,**

**private val onItemClick: (Interviewer) -> Unit**

**) : RecyclerView.Adapter<InterviewerAdapter.InterviewerViewHolder>() {**

**inner class InterviewerViewHolder(view: View) : RecyclerView.ViewHolder(view) {**

**private val tvName: TextView = view.findViewById(R.id.tvInterviewerName)**

**private val tvDate: TextView = view.findViewById(R.id.tvDate)**

**private val tvTime: TextView = view.findViewById(R.id.tvTime)**

**private val tvSubject: TextView = view.findViewById(R.id.tvSubject)**

**private val tvStatus: TextView = view.findViewById(R.id.tvStatus)**

**fun bind(interviewer: Interviewer) {**

**tvName.text = interviewer.name**

**tvDate.text = "Date: ${interviewer.date}"**

**tvTime.text = "Time: ${interviewer.timeFrom} - ${interviewer.timeTo}"**

**tvSubject.text = "Subject: ${interviewer.subject}"**

**tvStatus.text = "Status: ${interviewer.status}"**

**itemView.setOnClickListener { onItemClick(interviewer) }**

**}**

**}**

**override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): InterviewerViewHolder {**

**val view = LayoutInflater.from(parent.context)**

**.inflate(R.layout.item\_interviewer, parent, false)**

**return InterviewerViewHolder(view)**

**}**

**override fun onBindViewHolder(holder: InterviewerViewHolder, position: Int) {**

**holder.bind(interviewerList[position])**

**}**

**override fun getItemCount(): Int = interviewerList.size**

**}**

**database**

**DatabaseHelper**

**import android.content.ContentValues**

**import android.content.Context**

**import android.database.sqlite.SQLiteDatabase**

**import android.database.sqlite.SQLiteOpenHelper**

**import android.util.Log**

**import com.example.bookmyslot\_tag.database.UserModel**

**class DatabaseHelper(context: Context) :**

**SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {**

**init {**

**val dbPath = context.getDatabasePath(DATABASE\_NAME).absolutePath**

**Log.d("DatabaseHelper", "Database Path: $dbPath")**

**}**

**override fun onCreate(db: SQLiteDatabase?) {**

**val createTableQuery = """**

**CREATE TABLE $TABLE\_USERS (**

**$COLUMN\_USERNAME TEXT PRIMARY KEY,**

**$COLUMN\_EMAIL TEXT UNIQUE,**

**$COLUMN\_PASSWORD TEXT**

**)**

**""".trimIndent()**

**db?.execSQL(createTableQuery)**

**Log.d("DatabaseHelper", "User table created")**

**}**

**override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {**

**db?.execSQL("DROP TABLE IF EXISTS $TABLE\_USERS")**

**onCreate(db)**

**}**

**fun insertUser(user: UserModel): Boolean {**

**val db = this.writableDatabase**

**val checkQuery = "SELECT \* FROM $TABLE\_USERS WHERE $COLUMN\_USERNAME = ? OR $COLUMN\_EMAIL = ?"**

**val cursor = db.rawQuery(checkQuery, arrayOf(user.username, user.email))**

**if (cursor.count > 0) {**

**Log.d("DatabaseHelper", "User already exists: ${user.username}")**

**cursor.close()**

**db.close()**

**return false**

**}**

**val values = ContentValues().apply {**

**put(COLUMN\_USERNAME, user.username)**

**put(COLUMN\_EMAIL, user.email)**

**put(COLUMN\_PASSWORD, user.password) // TODO: Hash password before storing**

**}**

**val result = db.insert(TABLE\_USERS, null, values)**

**cursor.close()**

**db.close()**

**if (result == -1L) {**

**Log.d("DatabaseHelper", "User registration failed")**

**return false**

**}**

**Log.d("DatabaseHelper", "User registered: ${user.username}")**

**return true**

**}**

**fun checkUser(username: String, password: String): Boolean {**

**val db = this.readableDatabase**

**val query = "SELECT \* FROM $TABLE\_USERS WHERE $COLUMN\_USERNAME = ? AND $COLUMN\_PASSWORD = ?"**

**val cursor = db.rawQuery(query, arrayOf(username, password))**

**Log.d("DatabaseHelper", "Checking user: $username with password: $password")**

**val exists = cursor.count > 0**

**cursor.close()**

**db.close()**

**if (exists) {**

**Log.d("DatabaseHelper", "Login successful for $username")**

**} else {**

**Log.d("DatabaseHelper", "Invalid credentials for $username")**

**}**

**return exists**

**}**

**fun getAllUsers(): List<UserModel> {**

**val db = this.readableDatabase**

**val userList = mutableListOf<UserModel>()**

**val cursor = db.rawQuery("SELECT \* FROM $TABLE\_USERS", null)**

**while (cursor.moveToNext()) {**

**val username = cursor.getString(cursor.getColumnIndexOrThrow(COLUMN\_USERNAME))**

**val email = cursor.getString(cursor.getColumnIndexOrThrow(COLUMN\_EMAIL))**

**val password = cursor.getString(cursor.getColumnIndexOrThrow(COLUMN\_PASSWORD))**

**Log.d("DatabaseHelper", "Retrieved user - $username, $email, $password")**

**userList.add(UserModel(username, email, password))**

**}**

**cursor.close()**

**db.close()**

**return userList**

**}**

**companion object {**

**private const val DATABASE\_NAME = "bookmyslot.db"**

**private const val DATABASE\_VERSION = 2**

**private const val TABLE\_USERS = "users"**

**private const val COLUMN\_USERNAME = "username"**

**private const val COLUMN\_EMAIL = "email"**

**private const val COLUMN\_PASSWORD = "password"**

**}**

**}**

**database**

**Usermodel**

**package com.example.bookmyslot\_tag.database**

**data class UserModel(**

**val username: String,**

**val email: String,**

**val password: String**

**)**

**model**

**Interviewer**

**package com.example.bookmyslot\_tag.model**

**import java.io.Serializable**

**import android.os.Parcelable**

**import kotlinx.parcelize.Parcelize**

**@Parcelize**

**data class Interviewer(**

**val name: String,**

**val date: String,**

**val timeFrom: String,**

**val timeTo: String,**

**val subject: String,**

**var status: String**

**) : Parcelable**

**ui**

**DashboardFragment**

**package com.example.bookmyslot\_tag.ui**

**import android.os.Bundle**

**import android.view.LayoutInflater**

**import android.view.View**

**import android.view.ViewGroup**

**import androidx.fragment.app.Fragment**

**import androidx.navigation.fragment.findNavController**

**import androidx.recyclerview.widget.LinearLayoutManager**

**import androidx.recyclerview.widget.RecyclerView**

**import com.example.bookmyslot\_tag.R**

**import com.example.bookmyslot\_tag.adapter.InterviewerAdapter**

**import com.example.bookmyslot\_tag.model.Interviewer**

**class DashboardFragment : Fragment() {**

**private lateinit var recyclerView: RecyclerView**

**private lateinit var interviewerAdapter: InterviewerAdapter**

**private val interviewerList = mutableListOf<Interviewer>()**

**override fun onCreateView(**

**inflater: LayoutInflater, container: ViewGroup?,**

**savedInstanceState: Bundle?**

**): View? {**

**val view = inflater.inflate(R.layout.fragment\_dashboard, container, false)**

**recyclerView = view.findViewById(R.id.recyclerView)**

**recyclerView.layoutManager = LinearLayoutManager(requireContext())**

**populateInterviewerList()**

**interviewerAdapter = InterviewerAdapter(interviewerList) { selectedInterviewer ->**

**val action = DashboardFragmentDirections**

**.actionDashboardFragmentToInterviewerDetailFragment(selectedInterviewer)**

**findNavController().navigate(action)**

**}**

**recyclerView.adapter = interviewerAdapter**

**return view**

**}**

**private fun populateInterviewerList() {**

**interviewerList.add(Interviewer("Michael Johnson", "2025-03-01", "09:00 AM", "10:00 AM", "C++", "Available"))**

**interviewerList.add(Interviewer("Emma Williams", "2025-03-02", "10:30 AM", "11:30 AM", "Swift", "Booked"))**

**interviewerList.add(Interviewer("Robert Miller", "2025-03-03", "01:00 PM", "02:00 PM", "JavaScript", "Available"))**

**interviewerList.add(Interviewer("Sophia Wilson", "2025-03-04", "02:30 PM", "03:30 PM", "Go", "Available"))**

**interviewerList.add(Interviewer("David Anderson", "2025-03-05", "03:45 PM", "04:45 PM", "Ruby", "Booked"))**

**interviewerList.add(Interviewer("Olivia Martinez", "2025-03-06", "09:15 AM", "10:15 AM", "PHP", "Available"))**

**interviewerList.add(Interviewer("James Thomas", "2025-03-07", "11:00 AM", "12:00 PM", "C#", "Booked"))**

**interviewerList.add(Interviewer("Isabella Garcia", "2025-03-08", "01:45 PM", "02:45 PM", "Rust", "Available"))**

**interviewerList.add(Interviewer("William Robinson", "2025-03-09", "04:00 PM", "05:00 PM", "Perl", "Available"))**

**interviewerList.add(Interviewer("Mia Clark", "2025-03-10", "08:30 AM", "09:30 AM", "Scala", "Booked"))**

**interviewerList.add(Interviewer("Benjamin Lewis", "2025-03-11", "10:00 AM", "11:00 AM", "Dart", "Available"))**

**interviewerList.add(Interviewer("Charlotte Walker", "2025-03-12", "12:30 PM", "01:30 PM", "TypeScript", "Available"))**

**interviewerList.add(Interviewer("Ethan Hall", "2025-03-13", "03:00 PM", "04:00 PM", "Kotlin", "Booked"))**

**interviewerList.add(Interviewer("Ava Allen", "2025-03-14", "05:00 PM", "06:00 PM", "Python", "Available"))**

**interviewerList.add(Interviewer("Alexander Young", "2025-03-15", "09:45 AM", "10:45 AM", "R", "Available"))**

**interviewerList.add(Interviewer("Emily King", "2025-03-16", "11:15 AM", "12:15 PM", "Matlab", "Booked"))**

**interviewerList.add(Interviewer("Daniel Wright", "2025-03-17", "02:00 PM", "03:00 PM", "Shell", "Available"))**

**interviewerList.add(Interviewer("Harper Scott", "2025-03-18", "04:30 PM", "05:30 PM", "HTML/CSS", "Available"))**

**interviewerList.add(Interviewer("Henry Green", "2025-03-19", "07:00 AM", "08:00 AM", "SQL", "Booked"))**

**interviewerList.add(Interviewer("Abigail Adams", "2025-03-20", "08:45 AM", "09:45 AM", "Haskell", "Available"))**

**interviewerList.add(Interviewer("Matthew Nelson", "2025-03-21", "10:30 AM", "11:30 AM", "Lua", "Available"))**

**interviewerList.add(Interviewer("Ella Carter", "2025-03-22", "12:45 PM", "01:45 PM", "Objective-C", "Booked"))**

**interviewerList.add(Interviewer("Liam Baker", "2025-03-23", "03:15 PM", "04:15 PM", "F#", "Available"))**

**interviewerList.add(Interviewer("Madison Gonzalez", "2025-03-24", "05:45 PM", "06:45 PM", "Fortran", "Available"))**

**interviewerList.add(Interviewer("Noah Perez", "2025-03-25", "09:00 AM", "10:00 AM", "COBOL", "Booked"))**

**interviewerList.add(Interviewer("Lily Rivera", "2025-03-26", "10:30 AM", "11:30 AM", "VB.NET", "Available"))**

**interviewerList.add(Interviewer("Mason Torres", "2025-03-27", "01:00 PM", "02:00 PM", "Julia", "Available"))**

**interviewerList.add(Interviewer("Scarlett Ramirez", "2025-03-28", "02:30 PM", "03:30 PM", "Ada", "Booked"))**

**interviewerList.add(Interviewer("Lucas Flores", "2025-03-29", "03:45 PM", "04:45 PM", "Erlang", "Available"))**

**interviewerList.add(Interviewer("Grace Lee", "2025-03-30", "09:15 AM", "10:15 AM", "Prolog", "Available"))**

**interviewerList.add(Interviewer("Jackson Diaz", "2025-03-31", "11:00 AM", "12:00 PM", "Racket", "Booked"))**

**interviewerList.add(Interviewer("Sofia Hernandez", "2025-04-01", "01:45 PM", "02:45 PM", "Scheme", "Available"))**

**interviewerList.add(Interviewer("Aiden Martinez", "2025-04-02", "04:00 PM", "05:00 PM", "Elixir", "Available"))**

**interviewerList.add(Interviewer("Chloe Robinson", "2025-04-03", "08:30 AM", "09:30 AM", "D", "Booked"))**

**interviewerList.add(Interviewer("Carter Wilson", "2025-04-04", "10:00 AM", "11:00 AM", "Smalltalk", "Available"))**

**interviewerList.add(Interviewer("Zoe Scott", "2025-04-05", "12:30 PM", "01:30 PM", "Tcl", "Available"))**

**interviewerList.add(Interviewer("Logan Hall", "2025-04-06", "03:00 PM", "04:00 PM", "Lisp", "Booked"))**

**interviewerList.add(Interviewer("Penelope Adams", "2025-04-07", "05:00 PM", "06:00 PM", "Modula-2", "Available"))**

**interviewerList.add(Interviewer("Gabriel Wright", "2025-04-08", "09:45 AM", "10:45 AM", "OCaml", "Available"))**

**}**

**}**

**ui**

**InterviewerDetailFragment**

**package com.example.bookmyslot\_tag.ui**

**import android.app.AlertDialog**

**import android.content.Context**

**import android.content.Intent**

**import android.content.SharedPreferences**

**import android.net.Uri**

**import android.os.Bundle**

**import android.view.LayoutInflater**

**import android.view.View**

**import android.view.ViewGroup**

**import android.widget.Button**

**import android.widget.TextView**

**import android.widget.Toast**

**import androidx.fragment.app.Fragment**

**import androidx.activity.result.contract.ActivityResultContracts**

**import androidx.navigation.fragment.findNavController**

**import androidx.navigation.fragment.navArgs**

**import androidx.fragment.app.setFragmentResult**

**import com.example.bookmyslot\_tag.R**

**import com.example.bookmyslot\_tag.model.Interviewer**

**class InterviewerDetailFragment : Fragment() {**

**private val args: InterviewerDetailFragmentArgs by navArgs()**

**private lateinit var tvName: TextView**

**private lateinit var tvDate: TextView**

**private lateinit var tvTime: TextView**

**private lateinit var tvSubject: TextView**

**private lateinit var tvStatus: TextView**

**private lateinit var btnBookSlot: Button**

**private lateinit var btnReleaseSlot: Button**

**private lateinit var sharedPreferences: SharedPreferences**

**private lateinit var interviewerId: String**

**private var pendingStatusUpdate: String? = null**

**private lateinit var interviewer: Interviewer // Ensuring interviewer data is available**

**override fun onCreateView(**

**inflater: LayoutInflater, container: ViewGroup?,**

**savedInstanceState: Bundle?**

**): View? {**

**val view = inflater.inflate(R.layout.fragment\_interviewer\_details, container, false)**

**// Initialize UI elements**

**tvName = view.findViewById(R.id.tvDetailName)**

**tvDate = view.findViewById(R.id.tvDetailDate)**

**tvTime = view.findViewById(R.id.tvDetailTime)**

**tvSubject = view.findViewById(R.id.tvDetailSubject)**

**tvStatus = view.findViewById(R.id.tvDetailStatus)**

**btnBookSlot = view.findViewById(R.id.btnBookSlot)**

**btnReleaseSlot = view.findViewById(R.id.btnReleaseSlot)**

**// Retrieve interviewer object safely**

**interviewer = args.selectedInterviewer**

**interviewerId = interviewer.name // Unique identifier for SharedPreferences**

**sharedPreferences = requireContext().getSharedPreferences("BookingPrefs", Context.MODE\_PRIVATE)**

**// Load saved status or use default**

**val savedStatus = sharedPreferences.getString(interviewerId, null)**

**if (savedStatus != null) {**

**interviewer.status = savedStatus**

**}**

**updateUI(interviewer)**

**btnBookSlot.setOnClickListener {**

**if (interviewer.status == "Available") {**

**interviewer.status = "Booked"**

**saveStatus("Booked")**

**updateUI(interviewer)**

**sendResultToDashboard("Booked") // Send update to dashboard**

**showConfirmationDialog("Slot Booked Successfully!")**

**} else {**

**Toast.makeText(requireContext(), "Slot already booked!", Toast.LENGTH\_SHORT).show()**

**}**

**}**

**btnReleaseSlot.setOnClickListener {**

**if (interviewer.status == "Booked") {**

**AlertDialog.Builder(requireContext())**

**.setTitle("Confirmation")**

**.setMessage("Are you sure you want to release this slot?")**

**.setPositiveButton("Yes") { \_, \_ ->**

**pendingStatusUpdate = "Available"**

**sendEmail("Released", interviewer)**

**}**

**.setNegativeButton("No") { dialog, \_ -> dialog.dismiss() }**

**.show()**

**} else {**

**Toast.makeText(requireContext(), "Slot is not booked!", Toast.LENGTH\_SHORT).show()**

**}**

**}**

**return view**

**}**

**private fun saveStatus(status: String) {**

**sharedPreferences.edit().putString(interviewerId, status).apply()**

**}**

**private fun updateUI(interviewer: Interviewer) {**

**tvName.text = "Name: ${interviewer.name}"**

**tvDate.text = "Date: ${interviewer.date}"**

**tvTime.text = "Time: ${interviewer.timeFrom} - ${interviewer.timeTo}"**

**tvSubject.text = "Subject: ${interviewer.subject}"**

**tvStatus.text = "Status: ${interviewer.status}"**

**btnBookSlot.isEnabled = interviewer.status == "Available"**

**btnReleaseSlot.isEnabled = interviewer.status == "Booked"**

**}**

**// Email Intent Handling**

**private val emailLauncher = registerForActivityResult(ActivityResultContracts.StartActivityForResult()) { result ->**

**if (result.resultCode == android.app.Activity.RESULT\_OK && pendingStatusUpdate != null) {**

**saveStatus(pendingStatusUpdate!!)**

**interviewer.status = pendingStatusUpdate!! // Update local object**

**updateUI(interviewer)**

**sendResultToDashboard("Available") // Send update to dashboard**

**showConfirmationDialog("Slot Released Successfully!")**

**} else {**

**Toast.makeText(requireContext(), "Email canceled. Status remains the same.", Toast.LENGTH\_SHORT).show()**

**}**

**pendingStatusUpdate = null**

**}**

**private fun sendEmail(action: String, interviewer: Interviewer) {**

**val emailBody = "The interview slot with ${interviewer.name} on ${interviewer.date} from ${interviewer.timeFrom} to ${interviewer.timeTo} has been $action."**

**val intent = Intent(Intent.ACTION\_SENDTO).apply {**

**data = Uri.parse("mailto:")**

**putExtra(Intent.EXTRA\_SUBJECT, "Interview Slot $action")**

**putExtra(Intent.EXTRA\_TEXT, emailBody)**

**}**

**try {**

**emailLauncher.launch(Intent.createChooser(intent, "Send Email"))**

**} catch (e: Exception) {**

**Toast.makeText(requireContext(), "No email app found!", Toast.LENGTH\_SHORT).show()**

**}**

**}**

**private fun showConfirmationDialog(message: String) {**

**AlertDialog.Builder(requireContext())**

**.setTitle("Confirmation")**

**.setMessage(message)**

**.setPositiveButton("OK") { dialog, \_ -> dialog.dismiss() }**

**.show()**

**}**

**// 🔄 Sending data back to Dashboard for real-time status update**

**private fun sendResultToDashboard(updatedStatus: String) {**

**setFragmentResult("updateStatus", Bundle().apply {**

**putString("interviewerId", interviewerId)**

**putString("newStatus", updatedStatus)**

**})**

**}**

**}**

**ui**

**LoginFragment**

**package com.example.bookmyslot\_tag.ui**

**import DatabaseHelper**

**import android.os.Bundle**

**import android.view.LayoutInflater**

**import android.view.View**

**import android.view.ViewGroup**

**import android.widget.Button**

**import android.widget.EditText**

**import android.widget.TextView**

**import android.widget.Toast**

**import androidx.fragment.app.Fragment**

**import androidx.navigation.fragment.findNavController**

**import com.example.bookmyslot\_tag.R**

**class LoginFragment : Fragment() {**

**private lateinit var etUsername: EditText**

**private lateinit var etPassword: EditText**

**private lateinit var btnLogin: Button**

**private lateinit var tvRegister: TextView**

**private lateinit var dbHelper: DatabaseHelper**

**override fun onCreateView(**

**inflater: LayoutInflater, container: ViewGroup?,**

**savedInstanceState: Bundle?**

**): View? {**

**val view = inflater.inflate(R.layout.fragment\_login, container, false)**

**etUsername = view.findViewById(R.id.etUsername)**

**etPassword = view.findViewById(R.id.etPassword)**

**btnLogin = view.findViewById(R.id.btnLogin)**

**tvRegister = view.findViewById(R.id.tvRegister)**

**dbHelper = DatabaseHelper(requireContext())**

**btnLogin.setOnClickListener {**

**val username = etUsername.text.toString()**

**val password = etPassword.text.toString()**

**if (dbHelper.checkUser(username, password)) {**

**Toast.makeText(requireContext(), "Login Successful!", Toast.LENGTH\_SHORT).show()**

**findNavController().navigate(R.id.action\_loginFragment\_to\_dashboardFragment)**

**} else {**

**Toast.makeText(requireContext(), "Invalid Credentials!", Toast.LENGTH\_SHORT).show()**

**}**

**}**

**tvRegister.setOnClickListener {**

**findNavController().navigate(R.id.action\_loginFragment\_to\_registerFragment)**

**}**

**return view**

**}**

**}**

**ui**

**RegisterFragment**

**package com.example.bookmyslot\_tag.ui**

**import DatabaseHelper**

**import android.os.Bundle**

**import android.view.LayoutInflater**

**import android.view.View**

**import android.view.ViewGroup**

**import android.widget.Button**

**import android.widget.EditText**

**import android.widget.TextView**

**import android.widget.Toast**

**import androidx.fragment.app.Fragment**

**import androidx.navigation.fragment.findNavController**

**import com.example.bookmyslot\_tag.R**

**import com.example.bookmyslot\_tag.database.UserModel**

**class RegisterFragment : Fragment() {**

**private lateinit var etNewUsername: EditText**

**private lateinit var etNewEmail: EditText**

**private lateinit var etNewPassword: EditText**

**private lateinit var btnRegister: Button**

**private lateinit var tvBackToLogin: TextView // Back to Login TextView**

**private lateinit var dbHelper: DatabaseHelper**

**override fun onCreateView(**

**inflater: LayoutInflater, container: ViewGroup?,**

**savedInstanceState: Bundle?**

**): View? {**

**val view = inflater.inflate(R.layout.fragment\_register, container, false)**

**etNewUsername = view.findViewById(R.id.etNewUsername)**

**etNewEmail = view.findViewById(R.id.etNewEmail)**

**etNewPassword = view.findViewById(R.id.etNewPassword)**

**btnRegister = view.findViewById(R.id.btnRegister)**

**tvBackToLogin = view.findViewById(R.id.tvBackToLogin) //**

**dbHelper = DatabaseHelper(requireContext())**

**btnRegister.setOnClickListener {**

**val username = etNewUsername.text.toString().trim()**

**val email = etNewEmail.text.toString().trim()**

**val password = etNewPassword.text.toString().trim()**

**if (username.isNotEmpty() && email.isNotEmpty() && password.isNotEmpty()) {**

**val user = UserModel(username, email, password)**

**val isInserted = dbHelper.insertUser(user)**

**if (isInserted) { println(user.username.toString());**

**Toast.makeText(requireContext(), "Registration Successful!", Toast.LENGTH\_SHORT).show()**

**findNavController().navigate(R.id.action\_registerFragment\_to\_loginFragment)**

**} else {**

**Toast.makeText(requireContext(), "User already exists!", Toast.LENGTH\_SHORT).show()**

**}**

**} else {**

**Toast.makeText(requireContext(), "Please fill all fields!", Toast.LENGTH\_SHORT).show()**

**}**

**}**

**tvBackToLogin.setOnClickListener {**

**findNavController().navigate(R.id.action\_registerFragment\_to\_loginFragment)**

**}**

**return view**

**}**

**}**

**utils**

**EmailSender**

**package com.example.bookmyslot\_tag.utils**

**import android.content.Context**

**import android.content.Intent**

**import android.net.Uri**

**object EmailSender {**

**fun sendEmail(context: Context, subject: String, body: String) {**

**val intent = Intent(Intent.ACTION\_SENDTO).apply {**

**data = Uri.parse("mailto:") // Only email apps should handle this**

**putExtra(Intent.EXTRA\_SUBJECT, subject)**

**putExtra(Intent.EXTRA\_TEXT, body)**

**}**

**if (intent.resolveActivity(context.packageManager) != null) {**

**context.startActivity(Intent.createChooser(intent, "Send Email"))**

**}**

**}**

**}**

Appium Testing For Android

APP: BOOKMYSLOT\_TAG

// BaseTest.java

package com.example.bookmyslot\_tag.tests;

import io.appium.java\_client.AppiumDriver;

import io.appium.java\_client.android.AndroidDriver;

import io.appium.java\_client.remote.MobileCapabilityType;

import org.junit.After;

import org.junit.Before;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.remote.DesiredCapabilities;

import java.net.URL;

public class BaseTest {

protected AppiumDriver driver;

@Before

public void setUp() throws Exception {

DesiredCapabilities caps = new DesiredCapabilities();

caps.setCapability(MobileCapabilityType.PLATFORM\_NAME, "Android");

caps.setCapability(MobileCapabilityType.DEVICE\_NAME, "vivo 1933");

caps.setCapability(MobileCapabilityType.PLATFORM\_VERSION, "30");

caps.setCapability("appPackage", "com.example.bookmyslot\_tag");

caps.setCapability("appActivity", "com.example.bookmyslot\_tag.MainActivity");

caps.setCapability(MobileCapabilityType.AUTOMATION\_NAME, "UiAutomator2");

driver = new AndroidDriver(new URL("http://127.0.0.1:4723"), caps);

}

protected void performLogin() {

WebElement usernameField = driver.findElement(By.id("com.example.bookmyslot\_tag:id/etUsername"));

WebElement passwordField = driver.findElement(By.id("com.example.bookmyslot\_tag:id/etPassword"));

WebElement loginButton = driver.findElement(By.id("com.example.bookmyslot\_tag:id/btnLogin"));

usernameField.sendKeys("testuser123");

passwordField.sendKeys("password123");

loginButton.click();

}

@After

public void tearDown() {

if (driver != null) {

driver.quit();

}

}

}

// RegistrationTest.java

package com.example.bookmyslot\_tag.tests;

import org.junit.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import static org.junit.Assert.assertTrue;

public class RegistrationTest extends BaseTest {

@Test

public void testUserRegistration() throws InterruptedException {

WebElement registerLink = driver.findElement(By.id("com.example.bookmyslot\_tag:id/tvRegister"));

registerLink.click();

Thread.sleep(1000);

WebElement usernameField = driver.findElement(By.id("com.example.bookmyslot\_tag:id/etNewUsername"));

WebElement passwordField = driver.findElement(By.id("com.example.bookmyslot\_tag:id/etNewPassword"));

WebElement registerButton = driver.findElement(By.id("com.example.bookmyslot\_tag:id/btnRegister"));

usernameField.sendKeys("testuser123");

passwordField.sendKeys("password123");

registerButton.click();

Thread.sleep(1000);

assertTrue(driver.findElement(By.id("com.example.bookmyslot\_tag:id/btnLogin")).isDisplayed());

}

}

// LoginTest.java

package com.example.bookmyslot\_tag.tests;

import org.junit.Test;

import org.openqa.selenium.By;

import static org.junit.Assert.assertTrue;

public class LoginTest extends BaseTest {

@Test

public void testUserLogin() throws InterruptedException {

performLogin();

Thread.sleep(1000);

assertTrue(driver.findElement(By.id("com.example.bookmyslot\_tag:id/tvDashboardTitle")).isDisplayed());

}

}

// InterviewerListTest.java

package com.example.bookmyslot\_tag.tests;

import org.junit.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.interactions.PointerInput;

import org.openqa.selenium.interactions.Sequence;

import java.time.Duration;

import java.util.Arrays;

import static org.junit.Assert.\*;

public class InterviewerListTest extends BaseTest {

@Test

public void testViewInterviewerList() throws InterruptedException {

performLogin();

Thread.sleep(1000);

WebElement recyclerView = driver.findElement(By.id("com.example.bookmyslot\_tag:id/recyclerView"));

assertNotNull("RecyclerView not found!", recyclerView);

assertTrue(driver.findElements(By.id("com.example.bookmyslot\_tag:id/tvInterviewerName")).size() > 0);

}

@Test

public void testScrollInterviewerList() throws InterruptedException {

performLogin();

Thread.sleep(1000);

WebElement recyclerView = driver.findElement(By.id("com.example.bookmyslot\_tag:id/recyclerView"));

int startX = recyclerView.getLocation().getX() + (recyclerView.getSize().getWidth() / 2);

int startY = recyclerView.getLocation().getY() + (recyclerView.getSize().getHeight() \* 3 / 4);

int endY = recyclerView.getLocation().getY() + (recyclerView.getSize().getHeight() / 4);

PointerInput finger = new PointerInput(PointerInput.Kind.TOUCH, "finger");

Sequence scroll = new Sequence(finger, 1)

.addAction(finger.createPointerMove(Duration.ofMillis(0), PointerInput.Origin.viewport(), startX, startY))

.addAction(finger.createPointerDown(0))

.addAction(finger.createPointerMove(Duration.ofMillis(1000), PointerInput.Origin.viewport(), startX, endY))

.addAction(finger.createPointerUp(0));

driver.perform(Arrays.asList(scroll));

Thread.sleep(1000);

assertTrue("Scroll failed - new items not visible",

driver.findElements(By.id("com.example.bookmyslot\_tag:id/tvInterviewerName")).size() > 0);

}

}

// BookingTest.java

package com.example.bookmyslot\_tag.tests;

import org.junit.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import static org.junit.Assert.assertTrue;

public class BookingTest extends BaseTest {

protected void bookSlot() throws InterruptedException {

WebElement firstInterviewer = driver.findElement(By.id("com.example.bookmyslot\_tag:id/tvInterviewerName"));

firstInterviewer.click();

Thread.sleep(1000);

WebElement bookButton = driver.findElement(By.id("com.example.bookmyslot\_tag:id/btnBookSlot"));

bookButton.click();

Thread.sleep(1000);

}

@Test

public void testBookInterviewSlot() throws InterruptedException {

performLogin();

Thread.sleep(1000);

WebElement firstInterviewer = driver.findElement(By.id("com.example.bookmyslot\_tag:id/tvInterviewerName"));

firstInterviewer.click();

Thread.sleep(1000);

WebElement bookButton = driver.findElement(By.id("com.example.bookmyslot\_tag:id/btnBookSlot"));

bookButton.click();

Thread.sleep(1000);

WebElement status = driver.findElement(By.id("com.example.bookmyslot\_tag:id/tvDetailStatus"));

assertTrue(status.getText().contains("Booked"));

}

@Test

public void testReleaseInterviewSlot() throws InterruptedException {

performLogin();

bookSlot();

Thread.sleep(1000);

WebElement releaseButton = driver.findElement(By.id("com.example.bookmyslot\_tag:id/btnReleaseSlot"));

releaseButton.click();

Thread.sleep(1000);

WebElement confirmYes = driver.findElement(By.id("android:id/button1"));

confirmYes.click();

Thread.sleep(1000);

WebElement status = driver.findElement(By.id("com.example.bookmyslot\_tag:id/tvDetailStatus"));

assertTrue(status.getText().contains("Available"));

}

}