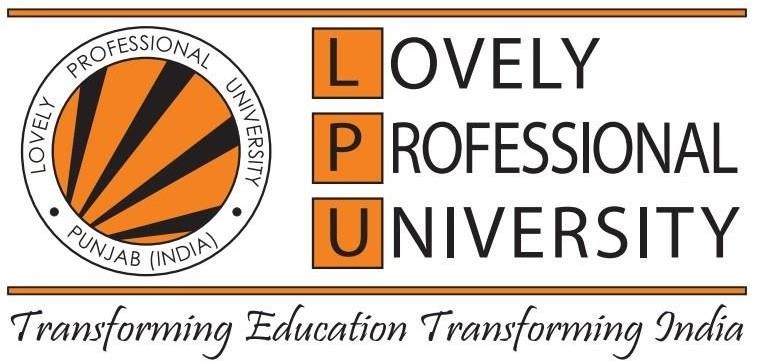
# Car Pooling Android App

**Report**

FROM

**BTech CSE (P132L)**

SUBMITTED TO



**LOVELY PROFESSIONAL UNIVERSITY PHAGWARA, PUNJAB**

SUBMITTED BY

**Full Name : Guntamukkala Gopi Krishna**

**Registration No : 12115851**

**Roll Number : 062 – KO203**

**Subject Code : CSE227**

**Subject Name : ADVANCED ANDROID APP DEVELOPMENT**

# GitHub Project Link: [ Included All ]

# Project Source Code

# Project Images

# Project Apk app file

# Attached Working Video

# Project Overview of features used

# Project Link : <https://github.com/gopi76/Car-Pooling-App>

# Topics Covered :

# CSE224 (FUNDAMENTALS OF ANDROID):

* Request App Permissions ( for locations )
* Log ( used to see errors specifically in app if user faced )
* Toast
* Layouts : Linear, Relative and Constraint
* Alert Dialog ( for showing the user information for confirmation)
* Menu

# CSE225 (DEVELOPING ANDROID APPS):

* Splash Screen (used at starting activity )
* Progress Bar (used when user clicked on login button )
* Intents( used both explicit and implicit intents)
* Loading
* Notification( user when clicked rating it will show rating given by them )
* Navigation Drawer
* View Pager ( user can view all the features of this app at starting )
* Date Picker Dialog
* Time Picker Dialog
* Bottom Navigation bar
* Rating Bar ( User can give rating based upto max 5 stars)

# CSE226: ANDROID APP DEPLOYMENT

* Recycler View
* Card View
* Floating Action Buttons ( FAB )
* Users Current Location
* Maps

**CSE227 : ADVANCED ANDROID APP DEVELOPMENT**

* Real Time Firebase
* Phone Verification
* Connected Wifi Information
* Animation ( zoom – in, zoom - out, rotate, fade in animations )
* Wifi Check
* Proximity sensor
* Web View

**Others**

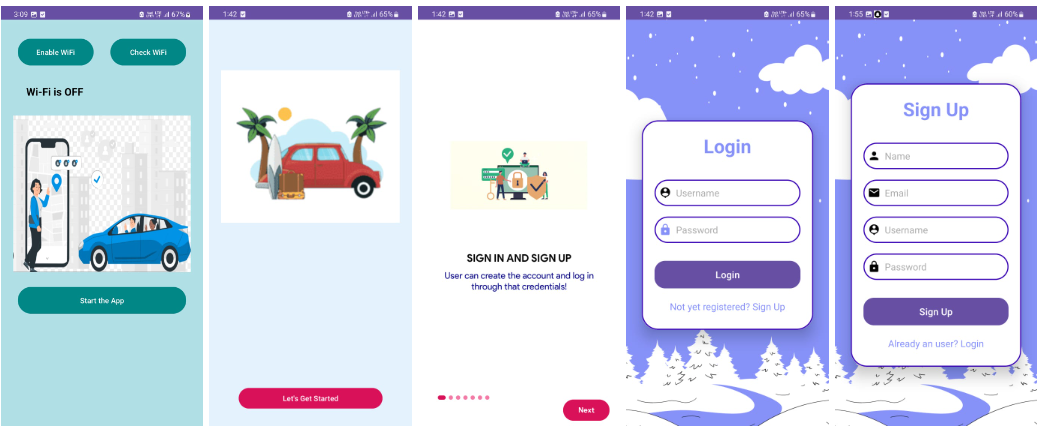
* Used Email Confirmation (for both car owner and rider) – through SMTP
* Category Selection ( based on user needs )
* Text To Speech
* Speech To Text

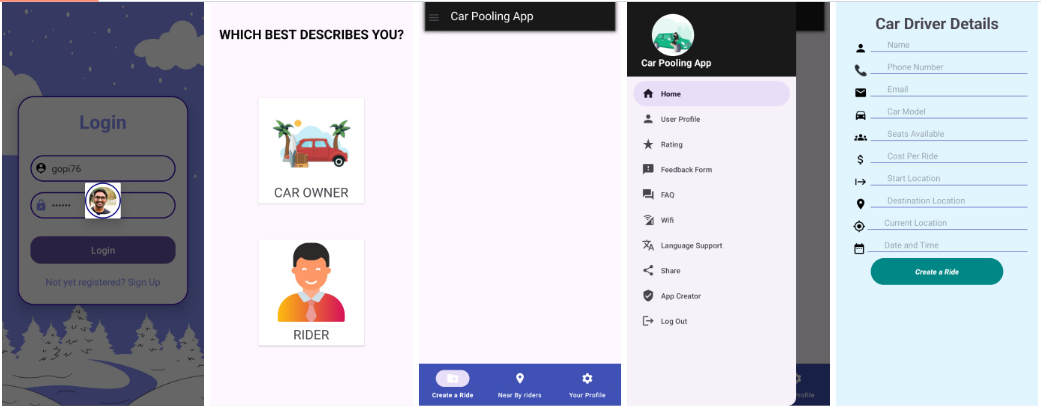
## **Project Overview**

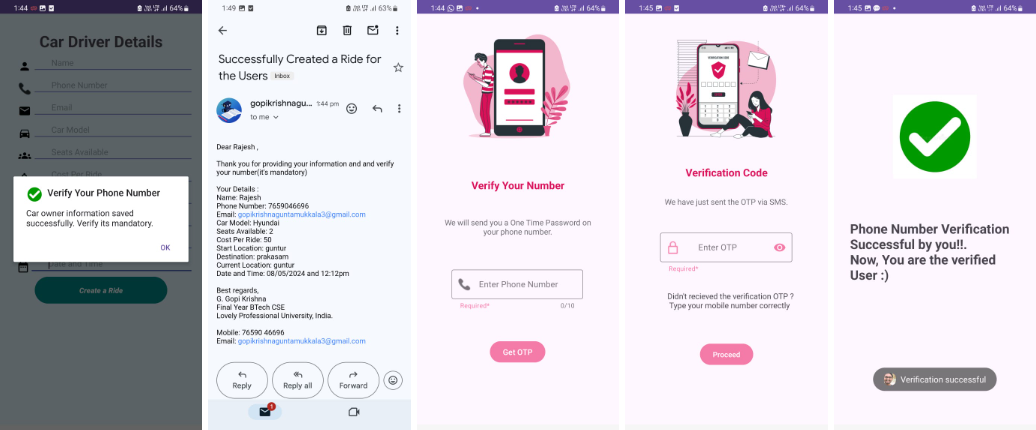
This project aims to create a Car Pooling App using Android Studio and Kotlin. It includes the following features:

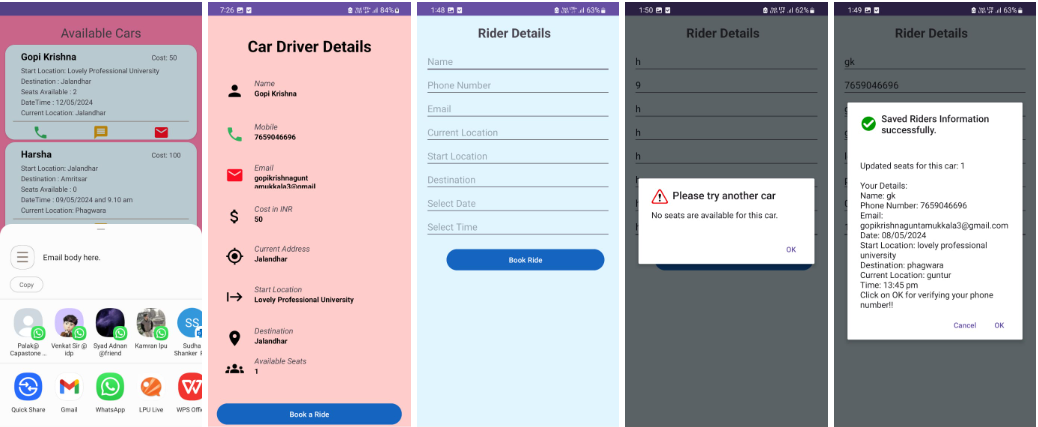
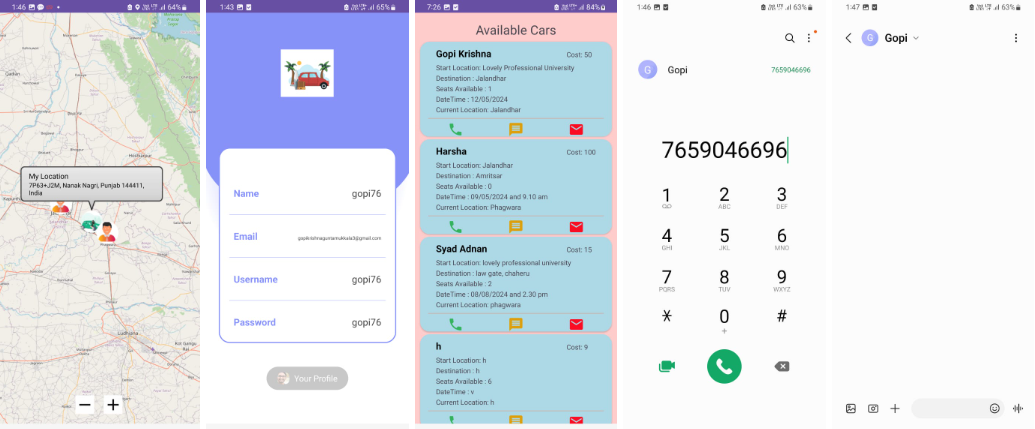
* **Sign In :** By entering their credentials, users can log in. By limiting access to the app's features to registered users alone, this feature preserves privacy and security.
* **Sign Up :** To access the app's features, users must first register for a new account and then log in. New users can use the app's tailored offerings after registering.
* **View and Update Profile :** Users are able to see and modify the information in their profile, including their name, email address, etc. This guarantees the accuracy and currency of user data.
* **Rating System :** Users have the option to review and comment on their experiences, and the rating is kept in the database. Users can also see the average rating given by all users and receive notifications regarding ratings.
* **Feedback Form :** Through this form, users can send in questions or feedback. A little check dialog box appears as confirmation after submission, making the user experience pleasant.
* **Category Selection :** Depending on their requirements, users can choose to be either a rider or an owner of a car. This classification, which takes into account personal preferences, simplifies the user experience.
* **Offer Rides :** Users are able to peruse and observe every ride that is available, together with information about the start and end times.
* **Owner Profile :** Users have access to comprehensive profiles of automobile owners that include facts about seats that are accessible, contact information (such as an email address or phone number), and other pertinent data.
* **Proximity Sensor ( Alert System ) :** To lessen eye strain and possible discomfort, the app uses a proximity sensor to warn users when they are getting too close to the screen. This feature makes using the app safer and more pleasant, which improves user experience.
* **Create a Ride :** Through the app, users can generate rides. For further protection, they must go through phone verification and obtain a confirmation email.
* **Book a Ride :** Through the app, users can suggest rides. Users receive emails confirming their reservations, and for extra security, they must verify over the phone.
* **Near by Riders ( Map View ) :** On the map view, users can see nearby riders, who are shown as person icons. This feature improves user visibility and makes ride selection easier by using Open Street Map view to display nearby riders' real-time positions.
* **Near by Cars ( Map View ) :** Using the car symbols on the map view, users can see the locations of nearby cars.
* **Search :** Users can go through all of the available riders and rides that other users have provided by using the search functionality. To identify particular rides based on user preferences, the search feature contains filtering options.
* **Share the App :** Through a Web View, users can obtain the app code and distribute it to other users. With social media sharing and other means, users can use this feature to spread the word about the app and increase its user base.
* **Logout :** The ability to log out of an account is available to users. When you click the logout button, a confirmation dialog box shows up.
* **Frequently Asked Questions, or FAQs** **:** it provide users with a thorough rundown of frequently asked questions about the app, its features.
* **Wifi Info :** Users are able to get comprehensive details about the Wifi network they are currently connected to.
* **Details About the App Creator :** Users can obtain information about the App Creator, such as background information, methods of contact (phone, email, or SMS), and ways to report issues with the app's usability or operation.
* **Turn on Wifi :** With this function, users can easily turn on Wifi right from within the app. Because internet access is necessary for the proper operation of apps, users can assure seamless and uninterrupted app usage by tapping the "Enable Wifi" option on their smartphone.
* **Text-to-Speech (TTS) :** This feature improves accessibility and user interaction by allowing users to convert text input into spoken language output in one of five languages.
* **Speech-to-Text (STT) :** Users can speak to transmit data; the speech is converted to text and shown in the application's text view ( only English ).

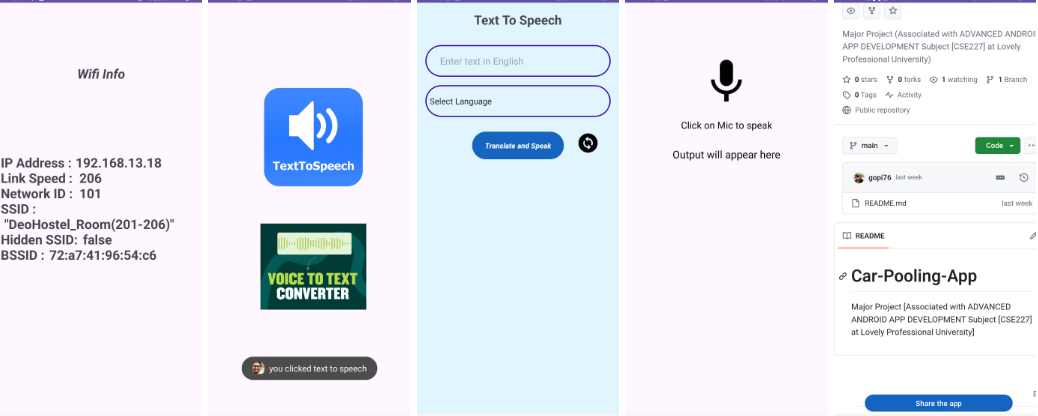
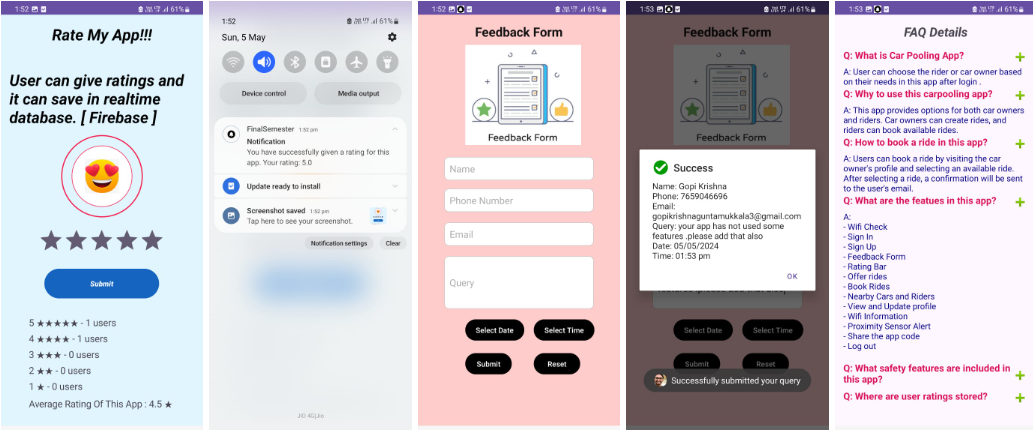
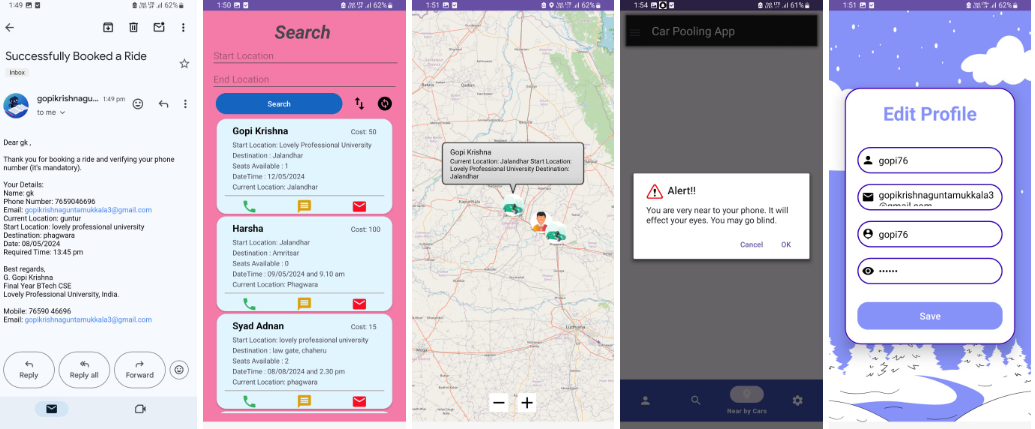
**Project Images:**

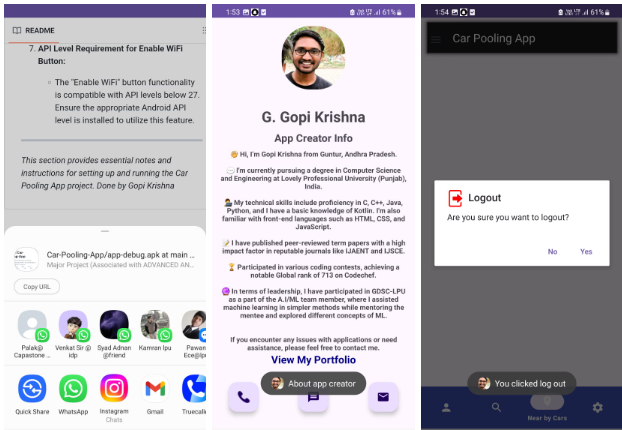
****

****

****

****

****

****

**Wifi Check Code**

package com.example.finalsemester  
import android.annotation.SuppressLint  
import android.content.Context  
import android.content.Intent  
import android.net.wifi.WifiManager  
import android.os.Build  
import android.os.Bundle  
import android.os.Handler  
import android.view.animation.AlphaAnimation  
import android.view.animation.AnimationUtils  
import android.widget.Button  
import android.widget.ImageView  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
class EnableDisableWifi : AppCompatActivity() {  
 private lateinit var enableButton: Button  
 private lateinit var checkButton: Button  
 private lateinit var carpoolapp: Button  
 private lateinit var statusTextView: TextView  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_enable\_disable\_wifi*)  
 enableButton = findViewById(R.id.*enablebtn*)  
 checkButton = findViewById(R.id.*checkbtn*)  
 carpoolapp = findViewById(R.id.*app*)  
 statusTextView = findViewById(R.id.*statusTextView*)  
 val fadeInAnimation = AnimationUtils.loadAnimation(this, R.anim.*fade\_in*)  
 enableButton.setOnClickListener **{** val wifiManager = *applicationContext*.getSystemService(Context.*WIFI\_SERVICE*) as WifiManager  
 if (!wifiManager.*isWifiEnabled*) {  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*Q*) {  
 wifiManager.startScan()  
 }  
 wifiManager.*isWifiEnabled* = true  
 }  
 **}** carpoolapp.setOnClickListener **{** val splashImageView = ImageView(this)  
 splashImageView.setImageResource(R.drawable.*splash11*)  
 val fadeInAnimation = AlphaAnimation(0f, 1f)  
 fadeInAnimation.*duration* = 1000  
 carpoolapp.startAnimation(fadeInAnimation)  
 Handler().postDelayed(**{** val intent = Intent(this, FirstActivity::class.*java*)  
 startActivity(intent)  
 **}**, 4000)  
 **}** checkButton.setOnClickListener **{** val wifiManager = *applicationContext*.getSystemService(Context.*WIFI\_SERVICE*) as WifiManager  
 val status = if (wifiManager.*isWifiEnabled*) "Wi-Fi is ON" else "Wi-Fi is OFF"  
 statusTextView.*text* = status  
 **}** }  
}

**First Activity (used animation in this code)**

package com.example.finalsemester  
import android.content.Intent  
import android.os.Bundle  
import android.view.animation.AnimationSet  
import android.view.animation.AnimationUtils  
import android.widget.Button  
import android.widget.ImageView  
import androidx.appcompat.app.AppCompatActivity  
import com.example.finalsemester.onboarding.OnBoardingActivity  
import java.util.Timer  
import kotlin.concurrent.timerTask  
class FirstActivity : AppCompatActivity() {  
 private var imageView: ImageView? = null  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_first*)  
 imageView = findViewById(R.id.*app\_name*)  
 val button: Button = findViewById(R.id.*button2*)  
 button.setOnClickListener **{** animateAndNavigate()  
 **}** }  
 private fun animateAndNavigate() {  
 val animationSet = AnimationSet(true)  
 val zoomOutAnimation = AnimationUtils.loadAnimation(this, R.anim.*zoom\_out*)  
 animationSet.addAnimation(zoomOutAnimation)  
 val zoomInAnimation = AnimationUtils.loadAnimation(this, R.anim.*zoom\_in*)  
 zoomInAnimation.*startOffset* = 1000  
 animationSet.addAnimation(zoomInAnimation)  
 val rotateAnimation = AnimationUtils.loadAnimation(this, R.anim.*rotate*)  
 rotateAnimation.*startOffset* = 2000  
 animationSet.addAnimation(rotateAnimation)  
 imageView?.startAnimation(animationSet)  
 Timer().schedule(*timerTask* **{** startActivity(Intent(this@FirstActivity, OnBoardingActivity::class.*java*))  
 finish()   
 overridePendingTransition(R.anim.*slide\_in\_right*, R.anim.*slide\_out\_left*)  
 **}**, 4000)  
 }  
}

**OnBoarding Activty**

package com.example.finalsemester.onboarding  
import android.content.Intent  
import android.os.Bundle  
import android.view.View  
import android.view.ViewGroup  
import android.widget.ImageView  
import android.widget.LinearLayout  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.content.ContextCompat  
import androidx.viewpager2.widget.ViewPager2  
import androidx.viewpager2.widget.ViewPager2.OnPageChangeCallback  
import com.example.finalsemester.LoginActivity  
import com.example.finalsemester.R  
import com.google.android.material.button.MaterialButton  
class OnBoardingActivity : AppCompatActivity() {  
 private lateinit var onboardingAdapter: OnboardingAdapter  
 private lateinit var layoutOnboardingIndicator: LinearLayout  
 private lateinit var buttonOnboardingAction: MaterialButton  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_on\_boarding*)  
 layoutOnboardingIndicator = findViewById(R.id.*layoutOnboardingIndicators*)  
 buttonOnboardingAction = findViewById(R.id.*buttonOnBoardingAction*)  
 setOnboardingItem()  
 val onboardingViewPager = findViewById<ViewPager2>(R.id.*onboardingViewPager*)  
 onboardingViewPager.*adapter* = onboardingAdapter  
 setOnboadingIndicator()  
 setCurrentOnboardingIndicators(0)  
  
 onboardingViewPager.registerOnPageChangeCallback(object : OnPageChangeCallback() {  
 override fun onPageSelected(position: Int) {  
 super.onPageSelected(position)  
 setCurrentOnboardingIndicators(position)  
 }  
 })  
 buttonOnboardingAction.setOnClickListener **{** view: View? **->** if (onboardingViewPager.*currentItem* + 1 < onboardingAdapter.*itemCount*) {  
 onboardingViewPager.*currentItem* = onboardingViewPager.*currentItem* + 1  
 } else {  
 moveToLogin()  
 finish()  
 }  
 **}** }  
 private fun setOnboadingIndicator() {  
 val indicators = *arrayOfNulls*<ImageView>(onboardingAdapter.*itemCount*)  
 val layoutParams = LinearLayout.LayoutParams(  
 ViewGroup.LayoutParams.*WRAP\_CONTENT*, ViewGroup.LayoutParams.*WRAP\_CONTENT* )  
 layoutParams.setMargins(8, 0, 8, 0)  
 for (i in indicators.*indices*) {  
 indicators[i] = ImageView(*applicationContext*)  
 indicators[i]!!.setImageDrawable(  
 ContextCompat.getDrawable(  
 *applicationContext*, R.drawable.*onboarding\_indicator\_inactive* )  
 )  
 indicators[i]!!.*layoutParams* = layoutParams  
 layoutOnboardingIndicator.addView(indicators[i])  
 }  
 }  
 private fun setCurrentOnboardingIndicators(index: Int) {  
 val childCount = layoutOnboardingIndicator.*childCount* for (i in 0 *until* childCount) {  
 val imageView = layoutOnboardingIndicator.getChildAt(i) as ImageView  
 imageView.setImageDrawable(  
 if (i == index) ContextCompat.getDrawable(*applicationContext*, R.drawable.*onboarding\_indicator\_active*)  
 else ContextCompat.getDrawable(*applicationContext*, R.drawable.*onboarding\_indicator\_inactive*)  
 )  
 }  
 buttonOnboardingAction.*text* = if (index == onboardingAdapter.*itemCount* - 1) "Start" else "Next"  
 }  
 private fun setOnboardingItem() {  
 val onBoardingItems: MutableList<OnBoardingItem> = ArrayList()  
 val loginpageinfo = OnBoardingItem()  
 loginpageinfo.setTitle("Sign In and Sign Up")  
 loginpageinfo.setDescription("User can create the account and log in through that credentials!")  
 loginpageinfo.setImage(R.drawable.*login*)  
 val carownerAndRider = OnBoardingItem()  
 carownerAndRider.setTitle("Car owner or Rider")  
 carownerAndRider.setDescription("User can select either they were car owner or rider in the app")  
 carownerAndRider.setImage(R.drawable.*category*)  
 val rides = OnBoardingItem()  
 rides.setTitle("Offer Rides")  
 rides.setDescription("User can see all the available rides in the app")  
 rides.setImage(R.drawable.*rides*)  
 val locations = OnBoardingItem()  
 locations.setTitle("Near by Cars")  
 locations.setDescription("User can see all the near by cars in the app by using Open Street Map")  
 locations.setImage(R.drawable.*location*)  
 val profile = OnBoardingItem()  
 profile.setTitle("User Profile Information")  
 profile.setDescription("User can view their profile and update their details if user wants too!")  
 profile.setImage(R.drawable.*updateprofile*)  
 val rating11 = OnBoardingItem()  
 rating11.setTitle("Rating Star")  
 rating11.setDescription("User can give the rating and it is stored in firebase")  
 rating11.setImage(R.drawable.*rating*)  
 val feedback11 = OnBoardingItem()  
 feedback11.setTitle("Rating and Feedback Form")  
 feedback11.setDescription("User can fill their queries through feedback form and these details are stored in realtime firebase")  
 feedback11.setImage(R.drawable.*feedback*)  
 onBoardingItems.add(loginpageinfo)  
 onBoardingItems.add(rides)  
 onBoardingItems.add(carownerAndRider)  
 onBoardingItems.add(locations)  
 onBoardingItems.add(profile)  
 onBoardingItems.add(rating11)  
 onBoardingItems.add(feedback11)  
 onboardingAdapter = OnboardingAdapter(onBoardingItems)  
 }  
 private fun moveToLogin() {  
 startActivity(Intent(*applicationContext*, LoginActivity::class.*java*))  
 finish()  
 }  
}

**Login Activity**

package com.example.finalsemester  
import android.app.Dialog  
import android.content.Intent  
import android.content.res.ColorStateList  
import android.os.Bundle  
import android.util.Log  
import android.widget.Button  
import android.widget.EditText  
import android.widget.ImageView  
import android.widget.ProgressBar  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.content.ContextCompat  
import com.google.firebase.database.DataSnapshot  
import com.google.firebase.database.DatabaseError  
import com.google.firebase.database.FirebaseDatabase  
import com.google.firebase.database.ValueEventListener  
class LoginActivity : AppCompatActivity() {  
 private lateinit var loginUsername: EditText  
 private lateinit var loginPassword: EditText  
 private lateinit var loginButton: Button  
 private lateinit var signupRedirectText: TextView  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_login*)  
 loginUsername = findViewById(R.id.*login\_username*)  
 loginPassword = findViewById(R.id.*login\_password*)  
 signupRedirectText = findViewById(R.id.*signupRedirectText*)  
 loginButton = findViewById(R.id.*login\_button*)  
 loginButton.setOnClickListener **{** if (validateInputs()) {  
 checkUser()  
 }  
 **}** signupRedirectText.setOnClickListener **{** startActivity(Intent(this@LoginActivity, SignupActivity::class.*java*))  
 **}** }  
 private fun validateInputs(): Boolean {  
 val username = loginUsername.*text*.toString().*trim*()  
 val password = loginPassword.*text*.toString().*trim*()  
 if (username.*isEmpty*()) {  
 loginUsername.*error* = "Username cannot be empty"  
 return false  
 }  
 if (password.*isEmpty*()) {  
 loginPassword.*error* = "Password cannot be empty"  
 return false  
 }  
 return true  
 }  
 private fun checkUser() {  
 val userUsername = loginUsername.*text*.toString().*trim*()  
 val userPassword = loginPassword.*text*.toString().*trim*()  
 val reference = FirebaseDatabase.getInstance().getReference("users")  
 val query = reference.orderByChild("username").equalTo(userUsername)  
 val progressDialog = Dialog(this@LoginActivity)  
 progressDialog.setContentView(R.layout.*custom\_progress\_dialog*)  
 val imageView: ImageView = progressDialog.findViewById(R.id.*imageView*)  
 val loadingProgressBar: ProgressBar = progressDialog.findViewById(R.id.*loadingProgressBar*)  
 imageView.setImageResource(R.drawable.*me*)  
 loadingProgressBar.*indeterminateTintList* = ColorStateList.valueOf(ContextCompat.getColor(this, R.color.*dark\_blue*))  
 progressDialog.setCancelable(false)  
 progressDialog.show()  
 query.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(snapshot: DataSnapshot) {  
 progressDialog.dismiss()  
 if (snapshot.exists()) {  
 for (childSnapshot in snapshot.*children*) {  
 val passwordFromDB = childSnapshot.child("password").getValue(String::class.*java*)  
 if (passwordFromDB == userPassword) {  
 val nameFromDB = childSnapshot.child("name").getValue(String::class.*java*)  
 val emailFromDB = childSnapshot.child("email").getValue(String::class.*java*)  
 val usernameFromDB = childSnapshot.child("username").getValue(String::class.*java*)  
  
 val userDetails = HashMap<String, String>().*apply* **{** put("name", nameFromDB!!)  
 put("email", emailFromDB!!)  
 put("username", usernameFromDB!!)  
 put("password", passwordFromDB!!)  
 **}** val intent = Intent(this@LoginActivity, CategorySelectionActivity::class.*java*).*apply* **{** putExtra("userDetails", userDetails)  
 **}** startActivity(intent)  
  
 Log.d("LoginActivity", "User logged in successfully: $userUsername")  
 return  
 } else {  
 loginPassword.*error* = "Invalid Credentials"  
 return  
 }  
 }  
 } else {  
 loginUsername.*error* = "User does not exist"  
 }  
 }  
 override fun onCancelled(error: DatabaseError) {  
 progressDialog.dismiss()  
 }  
 })  
 }  
}

**Sign up Activity**

package com.example.finalsemester  
import android.content.Intent  
import android.os.Bundle  
import android.view.View  
import android.widget.Button  
import android.widget.EditText  
import android.widget.TextView  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import com.example.finalsemester.HelperClass  
import com.example.finalsemester.LoginActivity  
import com.google.firebase.database.DatabaseReference  
import com.google.firebase.database.FirebaseDatabase  
class SignupActivity : AppCompatActivity() {  
 lateinit var signupName: EditText  
 lateinit var signupEmail: EditText  
 lateinit var signupUsername: EditText  
 lateinit var signupPassword: EditText  
 lateinit var loginRedirectText: TextView  
 lateinit var signupButton: Button  
 var database: FirebaseDatabase? = null  
 var reference: DatabaseReference? = null  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_signup*)  
 signupName = findViewById<EditText>(R.id.*signup\_name*)  
 signupEmail = findViewById<EditText>(R.id.*signup\_email*)  
 signupUsername = findViewById<EditText>(R.id.*signup\_username*)  
 signupPassword = findViewById<EditText>(R.id.*signup\_password*)  
 signupButton = findViewById<Button>(R.id.*signup\_button*)  
 loginRedirectText = findViewById<TextView>(R.id.*loginRedirectText*)  
 signupButton.setOnClickListener(View.OnClickListener **{** database = FirebaseDatabase.getInstance()  
 reference = database!!.getReference("users")  
 val name = signupName.getText().toString()  
 val email = signupEmail.getText().toString()  
 val username = signupUsername.getText().toString()  
 val password = signupPassword.getText().toString()  
 if (name.*isEmpty*() || email.*isEmpty*() || username.*isEmpty*() || password.*isEmpty*()) {  
 Toast.makeText(this@SignupActivity, "All details are mandatory to fill", Toast.*LENGTH\_SHORT*).show()  
 return@OnClickListener  
 }  
 val helperClass = HelperClass(name, email, username, password)  
 reference!!.child(username).setValue(helperClass)  
 Toast.makeText(this@SignupActivity, "You have signup successfully!", Toast.*LENGTH\_SHORT*)  
 .show()  
 val intent = Intent(this@SignupActivity, LoginActivity::class.*java*)  
 startActivity(intent)  
 **}**)  
 loginRedirectText.setOnClickListener(View.OnClickListener **{** val intent = Intent(this@SignupActivity, LoginActivity::class.*java*)  
 startActivity(intent)  
 **}**)  
 }  
}

**Category Selection Activity**

package com.example.finalsemester  
import android.content.Intent  
import android.os.Bundle  
import android.view.View  
import androidx.appcompat.app.AppCompatActivity  
import androidx.cardview.widget.CardView  
class CategorySelectionActivity : AppCompatActivity(), View.OnClickListener {  
 private lateinit var userDetails: HashMap<String, String>  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_category\_selection*)  
 userDetails = *intent*.getSerializableExtra("userDetails") as? HashMap<String, String>  
 ?: HashMap()   
 val rider = findViewById<CardView>(R.id.*rider*)  
 val car\_owner = findViewById<CardView>(R.id.*car\_owner*)  
 car\_owner.setOnClickListener(this)  
 rider.setOnClickListener(this)  
 }  
 override fun onClick(view: View?) {  
 when (view?.*id*) {  
 R.id.*car\_owner* -> {  
 val intent = Intent(this, MainActivity::class.*java*).*apply* **{** putExtra("userDetails", userDetails)  
 **}** startActivity(intent)  
 }  
 R.id.*rider* -> {  
 val intent = Intent(this, MainActivity2::class.*java*).*apply* **{** putExtra("userDetails", userDetails)  
 **}** startActivity(intent)  
 }  
 }  
 }  
}

**Data Classes : (for both riders and car owners)**

package com.example.finalsemester  
data class CarOwner(  
 var name: String = "",  
 var phoneNumber: String = "",  
 var email: String = "",  
 var carModel: String = "",  
 var seatsAvailable: String = "",  
 var costPerRide: String = "",  
 var startLocation: String = "",  
 var destination: String = "",  
 var currentLocation: String = "",  
 var dateTime: String = ""  
) {  
 constructor() : this(  
 "", "", "", "", "", "", "", "", "", ""  
 )  
}

package com.example.finalsemester  
data class Rider(  
 val email: String = "",  
 val displayName: String = "",  
 val phoneNumber: String = "",  
 val date: String = "",  
 val startLocation: String = "",  
 val destination: String = "",  
 val currentLocation: String = "",  
 val requiredTime: String = ""  
) {  
 constructor() : this(  
 email = "",  
 displayName = "",  
 phoneNumber = "",  
 date = "",  
 startLocation = "",  
 destination = "",  
 currentLocation = "",  
 requiredTime = ""  
 )  
}

**Main Activity 1 (for car\_owners)**

package com.example.finalsemester  
import android.content.Context  
import android.content.Intent  
import android.hardware.Sensor  
import android.hardware.SensorEvent  
import android.hardware.SensorEventListener  
import android.hardware.SensorManager  
import android.os.Bundle  
import android.view.MenuItem  
import android.widget.Toast  
import androidx.appcompat.app.ActionBarDrawerToggle  
import androidx.appcompat.app.AlertDialog  
import androidx.appcompat.app.AppCompatActivity  
import androidx.appcompat.widget.Toolbar  
import androidx.core.view.GravityCompat  
import androidx.drawerlayout.widget.DrawerLayout  
import com.google.android.material.bottomnavigation.BottomNavigationView  
import com.google.android.material.navigation.NavigationView  
class MainActivity : AppCompatActivity(), SensorEventListener {  
 private lateinit var layDL: DrawerLayout  
 private lateinit var vNV: NavigationView  
 private lateinit var toolbar: Toolbar  
 private lateinit var sensorManager: SensorManager  
 private var proximitySensor: Sensor? = null  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 layDL = findViewById(R.id.*layDL*)  
 vNV = findViewById(R.id.*vNV*)  
 toolbar = findViewById(R.id.*toolbar*)  
 sensorManager = getSystemService(Context.*SENSOR\_SERVICE*) as SensorManager  
 proximitySensor = sensorManager.getDefaultSensor(Sensor.*TYPE\_PROXIMITY*)  
 setSupportActionBar(toolbar)  
 val toggle = ActionBarDrawerToggle(  
 this, layDL, toolbar,  
 R.string.*open\_drawer*, R.string.*close\_drawer* )  
 layDL.addDrawerListener(toggle)  
 toggle.syncState()  
 if (savedInstanceState == null) {  
 vNV.setCheckedItem(R.id.*row\_home*)  
 }  
 navClick()  
 val bottomNavigationView11 = findViewById<BottomNavigationView>(R.id.*bottomNavigationView11*)  
 bottomNavigationView11.setOnNavigationItemSelectedListener **{** item **->** when (item.*itemId*) {  
 R.id.*nearbyriders* -> {  
 Toast.makeText(this, "you clicked nearByCars", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, NearByRiders::class.*java*)  
 startActivity(intent)  
 true  
 }  
 R.id.*createRide* -> {  
 Toast.makeText(this, "Create a Ride", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, CarOwnerProfileSetupActivity::class.*java*)  
 startActivity(intent)  
 true  
 }  
 R.id.*userprofilecarowner* -> {  
 Toast.makeText(this, "Your Profile", Toast.*LENGTH\_SHORT*).show()  
 val userDetails = *intent*.getSerializableExtra("userDetails") as HashMap<String, String>  
 val intent = Intent(this, UserProfile::class.*java*).*apply* **{** putExtra("userDetails", userDetails)  
 **}** startActivity(intent)  
 true  
 }  
 else -> false  
 }  
 **}** }  
 private fun navClick() {  
 vNV.setNavigationItemSelectedListener **{** item: MenuItem **->** when (item.*itemId*) {  
 R.id.*row\_home* -> Toast.makeText(this, "Home", Toast.*LENGTH\_SHORT*).show()  
 R.id.*settings\_profile* -> {  
 Toast.makeText(this, "Profile", Toast.*LENGTH\_SHORT*).show()  
 val userDetails = *intent*.getSerializableExtra("userDetails") as HashMap<String, String>  
 val intent = Intent(this, ProfileActivity::class.*java*).*apply* **{** putExtra("userDetails", userDetails)  
 **}** startActivity(intent)  
 }  
 R.id.*texttospeech* -> {  
 Toast.makeText(this, "you clicked text to speech", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, SelectionActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*appcreator* -> {  
 Toast.makeText(this, "About app creator", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, AppAdminInformationActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*logOut* -> {  
 Toast.makeText(this, "You clicked log out", Toast.*LENGTH\_SHORT*).show()  
 val builder = AlertDialog.Builder(this)  
  
 builder.setTitle("Logout")  
 builder.setMessage("Are you sure you want to logout?")  
 builder.setIcon(R.drawable.*logout*)  
  
 builder.setPositiveButton("Yes") **{** \_, \_ **->** val intent = Intent(this@MainActivity, LoginActivity::class.*java*)  
 startActivity(intent)  
 finish()  
 **}** builder.setNegativeButton("No") **{** dialog, \_ **->** dialog.dismiss()  
 **}** val dialog: AlertDialog = builder.create()  
 dialog.show()  
 }  
 R.id.*rating11* -> {  
 Toast.makeText(this, "you clicked rating", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, RatingStarActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*wifi* -> {  
 Toast.makeText(this, "wifi information", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, WifiInfoDemo::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*feedbackform11* -> {  
 Toast.makeText(this, "Feedback Form", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, FeedbackForm::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*faq* -> {  
 Toast.makeText(this, "you clicked FAQ", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, FAQActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*row\_share* -> {  
 Toast.makeText(this, "Share", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, ShareTheApp::class.*java*)  
 startActivity(intent)  
 }  
 }  
 layDL.closeDrawer(GravityCompat.*START*)  
 true  
 **}** }  
 override fun onBackPressed() {  
 if (layDL.isDrawerOpen(GravityCompat.*START*)) {  
 layDL.closeDrawer(GravityCompat.*START*)  
 } else {  
 super.onBackPressed()  
 }  
 }  
 override fun onResume() {  
 super.onResume()  
 proximitySensor?.*let* **{** sensorManager.registerListener(this, **it**, SensorManager.*SENSOR\_DELAY\_NORMAL*)  
 **}** }  
 override fun onPause() {  
 super.onPause()  
 proximitySensor?.*let* **{** sensorManager.unregisterListener(this, **it**)  
 **}** }  
 override fun onSensorChanged(event: SensorEvent?) {  
 event?.*let* **{** sensorEvent **->** if (sensorEvent.sensor == proximitySensor) {  
 val distance = sensorEvent.values.*getOrNull*(0)  
 val maxRange = proximitySensor?.*maximumRange* ?: 0f  
 distance?.*let* **{** if (**it** < maxRange) {  
 val errorMessage = "You are very near to your phone. It will effect your eyes. You may go blind."  
 showProximityAlert(errorMessage)  
 }  
 **}** }  
 **}** }  
 override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {  
 }  
 private fun showProximityAlert(errorMessage: String) {  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Alert!!")  
 builder.setMessage(errorMessage)  
 builder.setIcon(R.drawable.*error*)  
 builder.setPositiveButton("OK") **{** dialog, \_ **->** dialog.dismiss()  
 **}** builder.setNegativeButton("Cancel") **{** dialog, \_ **->** dialog.dismiss()  
 **}** val dialog: AlertDialog = builder.create()  
 dialog.show()  
 }  
}

**Main Activity 2 (for riders)**

package com.example.finalsemester  
import android.annotation.SuppressLint  
import android.content.Context  
import android.content.Intent  
import android.hardware.Sensor  
import android.hardware.SensorEvent  
import android.hardware.SensorEventListener  
import android.hardware.SensorManager  
import android.os.Bundle  
import android.view.MenuItem  
import android.widget.Toast  
import androidx.appcompat.app.ActionBarDrawerToggle  
import androidx.appcompat.app.AlertDialog  
import androidx.appcompat.app.AppCompatActivity  
import androidx.appcompat.widget.Toolbar  
import androidx.core.view.GravityCompat  
import androidx.drawerlayout.widget.DrawerLayout  
import com.google.android.material.bottomnavigation.BottomNavigationView  
import com.google.android.material.navigation.NavigationView  
class MainActivity2 : AppCompatActivity(), SensorEventListener {  
 private lateinit var layDL: DrawerLayout  
 private lateinit var vNV: NavigationView  
 private lateinit var toolbar: Toolbar  
 private lateinit var sensorManager: SensorManager  
 private var proximitySensor: Sensor? = null  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main2*)  
 layDL = findViewById(R.id.*layDL*)  
 vNV = findViewById(R.id.*vNV*)  
 toolbar = findViewById(R.id.*toolbar*)  
 sensorManager = getSystemService(Context.*SENSOR\_SERVICE*) as SensorManager  
 proximitySensor = sensorManager.getDefaultSensor(Sensor.*TYPE\_PROXIMITY*)  
 val bottomNavigationView1 = findViewById<BottomNavigationView>(R.id.*bottomNavigationView*)  
 bottomNavigationView1.setOnNavigationItemSelectedListener **{** item **->** when (item.*itemId*) {  
 R.id.*availablecars* -> {  
 Toast.makeText(this, "you clicked available Cars", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, CarOwnersActivity::class.*java*)  
 startActivity(intent)  
 true  
 }  
 R.id.*nearbycars* -> {  
 Toast.makeText(this, "you clicked nearByCars", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, NearByCars::class.*java*)  
 startActivity(intent)  
 true  
 }  
  
 R.id.*userprofilerider* -> {  
 Toast.makeText(this, "Your Profile", Toast.*LENGTH\_SHORT*).show()  
 val userDetails = *intent*.getSerializableExtra("userDetails") as HashMap<String, String>  
 val intent = Intent(this, UserProfile::class.*java*).*apply* **{** putExtra("userDetails", userDetails)  
 **}** startActivity(intent)  
 true  
 }  
 R.id.*searchcars* -> {  
 Toast.makeText(this, "You clicked Search", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, SearchCarsActivity::class.*java*)  
 startActivity(intent)  
 true  
 }  
 else -> false  
 }  
 **}** setSupportActionBar(toolbar)  
 val toggle = ActionBarDrawerToggle(  
 this, layDL, toolbar,  
 R.string.*open\_drawer*, R.string.*close\_drawer* )  
 layDL.addDrawerListener(toggle)  
 toggle.syncState()  
 if (savedInstanceState == null) {  
 vNV.setCheckedItem(R.id.*row\_home*)  
 }  
 navClick()  
 }  
 private fun navClick() {  
 vNV.setNavigationItemSelectedListener **{** item: MenuItem **->** when (item.*itemId*) {  
 R.id.*row\_home* -> Toast.makeText(this, "Home", Toast.*LENGTH\_SHORT*).show()  
 R.id.*settings\_profile* -> {  
 handleProfileNavigation()  
 true  
 }  
 R.id.*logOut* -> {  
 Toast.makeText(this, "You clicked log out", Toast.*LENGTH\_SHORT*).show()  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Logout")  
 builder.setMessage("Are you sure you want to logout?")  
 builder.setIcon(R.drawable.*logout*)  
  
 builder.setPositiveButton("Yes") **{** \_, \_ **->** val intent = Intent(this@MainActivity2, LoginActivity::class.*java*)  
 startActivity(intent)  
 finish()  
 **}** builder.setNegativeButton("No") **{** dialog, \_ **->** dialog.dismiss()  
 **}** val dialog: AlertDialog = builder.create()  
 dialog.show()  
 }  
  
 R.id.*rating11* -> {  
 Toast.makeText(this, "you clicked rating", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, RatingStarActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*wifi* -> {  
 Toast.makeText(this, "wifi information", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, WifiInfoDemo::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*texttospeech* -> {  
 Toast.makeText(this, "you clicked text to speech", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, SelectionActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*appcreator* -> {  
 Toast.makeText(this, "About app creator", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, AppAdminInformationActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*feedbackform11* -> {  
 Toast.makeText(this, "Feedback Form", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, FeedbackForm::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*faq* -> {  
 Toast.makeText(this, "you clicked FAQ", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, FAQActivity::class.*java*)  
 startActivity(intent)  
 }  
 R.id.*row\_share* -> {  
 Toast.makeText(this, "Share", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, ShareTheApp::class.*java*)  
 startActivity(intent)  
 }  
 }  
 layDL.closeDrawer(GravityCompat.*START*)  
 true  
 **}** }  
 private fun handleProfileNavigation() {  
 val userDetails = *intent*.getSerializableExtra("userDetails") as? HashMap<\*, \*>  
 if (userDetails != null && userDetails is HashMap<\*, \*>) {  
 val intent = Intent(this, ProfileActivity::class.*java*).*apply* **{** putExtra("userDetails", userDetails as HashMap<String, String>)  
 **}** startActivity(intent)  
 } else {  
 Toast.makeText(this, "User details not found", Toast.*LENGTH\_SHORT*).show()  
 }  
 }  
  
 override fun onBackPressed() {  
 if (layDL.isDrawerOpen(GravityCompat.*START*)) {  
 layDL.closeDrawer(GravityCompat.*START*)  
 } else {  
 super.onBackPressed()  
 }  
 }  
 override fun onResume() {  
 super.onResume()  
 proximitySensor?.*let* **{** sensorManager.registerListener(this, **it**, SensorManager.*SENSOR\_DELAY\_NORMAL*)  
 **}** }  
 override fun onPause() {  
 super.onPause()  
 proximitySensor?.*let* **{** sensorManager.unregisterListener(this, **it**)  
 **}** }  
 override fun onSensorChanged(event: SensorEvent?) {  
 event?.*let* **{** sensorEvent **->** if (sensorEvent.sensor == proximitySensor) {  
 val distance = sensorEvent.values.*getOrNull*(0)  
 val maxRange = proximitySensor?.*maximumRange* ?: 0f  
 distance?.*let* **{** if (**it** < maxRange) {  
 val errorMessage = "You are very near to your phone. It will effect your eyes. You may go blind."  
 showProximityAlert(errorMessage)  
 }  
 **}** }  
 **}** }  
 override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {}  
 private fun showProximityAlert(errorMessage: String) {  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Alert!!")  
 builder.setMessage(errorMessage)  
 builder.setIcon(R.drawable.*error*)  
 builder.setPositiveButton("OK") **{** dialog, \_ **->** dialog.dismiss()  
 **}** builder.setNegativeButton("Cancel") **{** dialog, \_ **->** dialog.dismiss()  
 **}** val dialog: AlertDialog = builder.create()  
 dialog.show()  
 }  
}

**Car Owner Profile Setup Activity**

package com.example.finalsemester  
import android.content.Intent  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AlertDialog  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.database.DatabaseReference  
import com.google.firebase.database.FirebaseDatabase  
import kotlinx.coroutines.Dispatchers  
import kotlinx.coroutines.GlobalScope  
import kotlinx.coroutines.launch  
import java.util.Properties  
import javax.mail.Authenticator  
import javax.mail.Message  
import javax.mail.PasswordAuthentication  
import javax.mail.Session  
import javax.mail.Transport  
import javax.mail.internet.InternetAddress  
import javax.mail.internet.MimeMessage  
class CarOwnerProfileSetupActivity : AppCompatActivity() {  
 private lateinit var nameEditText: EditText  
 private lateinit var phoneNumberEditText: EditText  
 private lateinit var emailEditText: EditText  
 private lateinit var carModelEditText: EditText  
 private lateinit var seatsEditText: EditText  
 private lateinit var costPerRideEditText: EditText  
 private lateinit var startLocationEditText: EditText  
 private lateinit var destinationEditText: EditText  
 private lateinit var currentLocationEditText: EditText  
 private lateinit var dateTimeEditText: EditText  
 private lateinit var saveButton: Button  
 private lateinit var database: FirebaseDatabase  
 private lateinit var reference: DatabaseReference  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_car\_owner\_profile\_setup*)  
 nameEditText = findViewById(R.id.*editTextName*)  
 phoneNumberEditText = findViewById(R.id.*editTextPhoneNumber*)  
 emailEditText = findViewById(R.id.*editTextEmail*)  
 carModelEditText = findViewById(R.id.*editTextCarModel*)  
 seatsEditText = findViewById(R.id.*editTextSeats*)  
 costPerRideEditText = findViewById(R.id.*editTextCost*)  
 startLocationEditText = findViewById(R.id.*editTextStartLocation*)  
 destinationEditText = findViewById(R.id.*editTextDestinationLocation*)  
 currentLocationEditText = findViewById(R.id.*editTextCurrentLocation*)  
 dateTimeEditText = findViewById(R.id.*editTextDateTime*)  
 saveButton = findViewById(R.id.*buttonSave*)  
 database = FirebaseDatabase.getInstance()  
 reference = database.*reference*.child("car\_owners")  
 saveButton.setOnClickListener **{** saveCarOwnerInformation()  
 **}** }  
 private fun saveCarOwnerInformation() {  
 val name = nameEditText.*text*.toString().*trim*()  
 val phoneNumber = phoneNumberEditText.*text*.toString().*trim*()  
 val email = emailEditText.*text*.toString().*trim*()  
 val carModel = carModelEditText.*text*.toString().*trim*()  
 val seatsAvailable = seatsEditText.*text*.toString().*trim*()  
 val costPerRide = costPerRideEditText.*text*.toString().*trim*()  
 val startLocation = startLocationEditText.*text*.toString().*trim*()  
 val destination = destinationEditText.*text*.toString().*trim*()  
 val currentLocation = currentLocationEditText.*text*.toString().*trim*()  
 val dateTime = dateTimeEditText.*text*.toString().*trim*()  
 if (name.*isEmpty*() || phoneNumber.*isEmpty*() || email.*isEmpty*() ||  
 carModel.*isEmpty*() || seatsAvailable.*isEmpty*() || costPerRide.*isEmpty*() ||  
 startLocation.*isEmpty*() || destination.*isEmpty*() || currentLocation.*isEmpty*() || dateTime.*isEmpty*()  
 ) {  
 Toast.makeText(this, "Please fill in all required fields", Toast.*LENGTH\_SHORT*).show()  
 return  
 }  
 val carOwnerInfo = *mapOf*(  
 "email" *to* email,  
 "name" *to* name,  
 "phoneNumber" *to* phoneNumber,  
 "carModel" *to* carModel,  
 "seatsAvailable" *to* seatsAvailable,  
 "costPerRide" *to* costPerRide,  
 "startLocation" *to* startLocation,  
 "destination" *to* destination,  
 "currentLocation" *to* currentLocation,  
 "dateTime" *to* dateTime  
 )  
 sendEmail(email, carOwnerInfo)  
 reference.push().setValue(carOwnerInfo)  
 .addOnSuccessListener **{** AlertDialog.Builder(this)  
 .setTitle("Verify Your Phone Number")  
 .setIcon(R.drawable.*tick*)  
 .setMessage("Car owner information saved successfully. Verify its mandatory.")  
 .setPositiveButton("OK") **{** dialog, which **->** val intent = Intent(this, PhoneAuthenticationActivity::class.*java*)  
 startActivity(intent)  
 **}** .show()  
 clearEditTextFields()   
 **}** .addOnFailureListener **{** e **->** Toast.makeText(this, "Failed to save car owner information", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
 private fun sendEmail(recipientEmail: String, userDetails: Map<String, String>) {  
 GlobalScope.*launch*(Dispatchers.IO) **{** try {  
 val props = Properties()  
 props["mail.smtp.host"] = "smtp.gmail.com"  
 props["mail.smtp.port"] = "587"  
 props["mail.smtp.auth"] = "true"  
 props["mail.smtp.starttls.enable"] = "true"  
 val session = Session.getInstance(props, object : Authenticator() {  
 override fun getPasswordAuthentication(): PasswordAuthentication {  
 return PasswordAuthentication("gopikrishnaguntamukkala3@gmail.com", "zpuebnvfffveyfou")  
 }  
 })  
 val message = MimeMessage(session)  
 message.setFrom(InternetAddress("gopikrishnaguntamukkala3@gmail.com"))  
 message.addRecipient(Message.RecipientType.*TO*, InternetAddress(recipientEmail))  
 message.*subject* = "Successfully Created a Ride for the Users"  
 val messageBody = *buildString* **{** append("Dear ${userDetails["name"]} ,\n\n")  
 append("Thank you for providing your information and and verify your number(it's mandatory) \n\n")  
 append("Your Details : \n")  
 append("Name: ${userDetails["name"]}\n")  
 append("Phone Number: ${userDetails["phoneNumber"]}\n")  
 append("Email: ${userDetails["email"]}\n")  
 append("Car Model: ${userDetails["carModel"]}\n")  
 append("Seats Available: ${userDetails["seatsAvailable"]}\n")  
 append("Cost Per Ride: ${userDetails["costPerRide"]}\n")  
 append("Start Location: ${userDetails["startLocation"]}\n")  
 append("Destination: ${userDetails["destination"]}\n")  
 append("Current Location: ${userDetails["currentLocation"]}\n")  
 append("Date and Time: ${userDetails["dateTime"]}\n\n")  
 append("Best regards,\nG. Gopi Krishna\nFinal Year BTech CSE\nLovely Professional University, India.\n\nMobile: 76590 46696\nEmail: gopikrishnaguntamukkala3@gmail.com")  
 **}** message.setText(messageBody)  
 Transport.send(message)  
 runOnUiThread **{** Toast.makeText(this@CarOwnerProfileSetupActivity, "Email sent successfully", Toast.*LENGTH\_SHORT*).show()  
 **}** } catch (e: Exception) {  
 e.printStackTrace()  
 runOnUiThread **{** Toast.makeText(this@CarOwnerProfileSetupActivity, "Failed to send email: ${e.message}", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
 **}** }  
 private fun clearEditTextFields() {  
 nameEditText.*text*.clear()  
 phoneNumberEditText.*text*.clear()  
 emailEditText.*text*.clear()  
 carModelEditText.*text*.clear()  
 seatsEditText.*text*.clear()  
 costPerRideEditText.*text*.clear()  
 startLocationEditText.*text*.clear()  
 destinationEditText.*text*.clear()  
 currentLocationEditText.*text*.clear()  
 dateTimeEditText.*text*.clear()  
 }  
}

**Near By Riders Activity**

package com.example.finalsemester  
  
import android.Manifest  
import android.content.pm.PackageManager  
import android.graphics.Bitmap  
import android.graphics.drawable.BitmapDrawable  
import android.location.Geocoder  
import android.os.Bundle  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.app.ActivityCompat  
import androidx.core.content.ContextCompat  
import androidx.core.graphics.drawable.RoundedBitmapDrawableFactory  
import com.google.android.gms.location.FusedLocationProviderClient  
import com.google.android.gms.location.LocationServices  
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  
import org.osmdroid.config.Configuration  
import org.osmdroid.tileprovider.tilesource.TileSourceFactory  
import org.osmdroid.util.GeoPoint  
import org.osmdroid.views.MapView  
import org.osmdroid.views.overlay.Marker  
import org.osmdroid.views.overlay.mylocation.GpsMyLocationProvider  
import org.osmdroid.views.overlay.mylocation.MyLocationNewOverlay  
import java.io.IOException  
  
class NearByRiders : AppCompatActivity() {  
 private lateinit var map1: MapView  
 private lateinit var fusedLocationClient: FusedLocationProviderClient  
 private lateinit var database: DatabaseReference  
 companion object {  
 private const val REQUEST\_LOCATION\_PERMISSION = 1  
 }  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_near\_by\_riders*)  
 Configuration.getInstance().load(this, getSharedPreferences("osmdroid", *MODE\_PRIVATE*))  
 map1 = findViewById(R.id.*map1*)  
 map1.setTileSource(TileSourceFactory.*MAPNIK*)  
 map1.setMultiTouchControls(true)  
 val initialCenter = GeoPoint(31.5018, 75.5728)  
 map1.*controller*.setCenter(initialCenter)  
 map1.*controller*.setZoom(11.0)  
 if (ContextCompat.checkSelfPermission(this, Manifest.permission.*ACCESS\_FINE\_LOCATION*) == PackageManager.*PERMISSION\_GRANTED*) {  
 initializeMap()  
 } else {  
 ActivityCompat.requestPermissions(  
 this,  
 *arrayOf*(Manifest.permission.*ACCESS\_FINE\_LOCATION*),  
 REQUEST\_LOCATION\_PERMISSION  
 )  
 }  
 database = FirebaseDatabase.getInstance().getReference("riders")  
 retrieveRiderInformation()  
 }  
 private fun initializeMap() {  
 val locationOverlay = MyLocationNewOverlay(GpsMyLocationProvider(this), map1)  
 locationOverlay.enableMyLocation()  
 map1.*overlays*.add(locationOverlay)  
 fusedLocationClient = LocationServices.getFusedLocationProviderClient(this)  
 if (ActivityCompat.checkSelfPermission(this, Manifest.permission.*ACCESS\_FINE\_LOCATION*) == PackageManager.*PERMISSION\_GRANTED*) {  
 fusedLocationClient.*lastLocation* .addOnSuccessListener **{** location **->** if (location != null) {  
 val geocoder = Geocoder(this)  
 try {  
 val addressList = geocoder.getFromLocation(location.*latitude*, location.*longitude*, 1)  
 if (addressList != null && addressList.*isNotEmpty*()) {  
 val address = addressList[0]  
 val currentLocation = GeoPoint(location.*latitude*, location.*longitude*)  
 addMarkerAtLocation(currentLocation, "My Location", address.getAddressLine(0), true)  
 } else {  
 addDefaultMarker()  
 }  
 } catch (e: IOException) {  
 e.printStackTrace()  
 addDefaultMarker()  
 }  
 }  
 **}** }  
 }  
 private fun addDefaultMarker() {  
 val defaultLocation = GeoPoint(31.2565, 75.6509)  
 val marker = Marker(map1)  
 marker.*position* = defaultLocation  
 marker.*title* = "Default Location"  
 marker.*snippet* = "Lovely Professional University"  
 marker.*icon* = ContextCompat.getDrawable(this, R.drawable.*marker\_red*)  
 map1.*overlays*.add(marker)  
 }  
 private fun retrieveRiderInformation() {  
 database.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(dataSnapshot: DataSnapshot) {  
 if (dataSnapshot.exists()) {  
 for (riderSnapshot in dataSnapshot.*children*) {  
 val rider = riderSnapshot.getValue(Rider::class.*java*)  
 rider?.*let* **{** val currentLocation = rider.currentLocation ?: ""  
 if (currentLocation.*isNotBlank*()) {  
 val geocoder = Geocoder(this@NearByRiders)  
 try {  
 val addressList = geocoder.getFromLocationName(currentLocation, 1)  
 if (addressList != null && addressList.*isNotEmpty*()) {  
 val address = addressList[0]  
 val riderLocation = GeoPoint(address.*latitude*, address.*longitude*)  
 val snippet = *buildString* **{** append("Current Location: $currentLocation\n")  
 append("Start Location: ${rider.startLocation ?: ""}\n")  
 append("Destination: ${rider.destination ?: ""}")  
 **}** addMarkerAtLocation(riderLocation, rider.displayName ?: "", snippet, false)  
 }  
 } catch (e: IOException) {  
 e.printStackTrace()  
 }  
 }  
 **}** }  
 }  
 }  
 override fun onCancelled(databaseError: DatabaseError) {  
 databaseError.toException().printStackTrace()  
 }  
 })  
 }  
 private fun addMarkerAtLocation(location: GeoPoint, title: String, snippet: String, isMyLocation: Boolean) {  
 val marker = Marker(map1)  
 marker.*position* = location  
 marker.*title* = title  
 marker.*snippet* = snippet  
 val markerDrawable = if (isMyLocation) {  
 ContextCompat.getDrawable(this, R.drawable.*round1*)  
 } else {  
 ContextCompat.getDrawable(this, R.drawable.*round111*)  
 }  
 markerDrawable?.*let* **{** val width = *resources*.getDimension(R.dimen.*marker\_size*).toInt()  
 val height = *resources*.getDimension(R.dimen.*marker\_size*).toInt()  
 val resizedBitmap = Bitmap.createScaledBitmap((**it** as BitmapDrawable).*bitmap*, width, height, false)  
 val roundedBitmapDrawable = RoundedBitmapDrawableFactory.create(*resources*, resizedBitmap)  
 roundedBitmapDrawable.*isCircular* = true  
 marker.*icon* = BitmapDrawable(*resources*, roundedBitmapDrawable.*bitmap*)  
 **}** map1.*overlays*.add(marker)  
 marker.showInfoWindow()  
 }  
  
 override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out String>, grantResults: IntArray) {  
 super.onRequestPermissionsResult(requestCode, permissions, grantResults)  
 if (requestCode == REQUEST\_LOCATION\_PERMISSION) {  
 if (grantResults.*isNotEmpty*() && grantResults[0] == PackageManager.*PERMISSION\_GRANTED*) {  
 initializeMap()  
 } else {  
 }  
 }  
 }  
 override fun onResume() {  
 super.onResume()  
 map1.onResume()  
 }  
 override fun onPause() {  
 super.onPause()  
 map1.onPause()  
 }  
}

**Phone Authentication Activity**

package com.example.finalsemester  
  
import android.content.Intent  
import android.os.Bundle  
import android.view.View  
import android.widget.Button  
import android.widget.ProgressBar  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import com.google.android.material.textfield.TextInputEditText  
import com.google.firebase.FirebaseException  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.auth.PhoneAuthCredential  
import com.google.firebase.auth.PhoneAuthProvider  
import java.util.concurrent.TimeUnit  
  
class PhoneAuthenticationActivity : AppCompatActivity() {  
 private lateinit var phoneNumberInput: TextInputEditText  
 private lateinit var progressBar: ProgressBar  
 private var verificationId: String? = null  
 private lateinit var auth: FirebaseAuth  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_phone\_authentication*)  
 auth = FirebaseAuth.getInstance()  
 phoneNumberInput = findViewById(R.id.*phone\_verify\_input*)  
 progressBar = findViewById(R.id.*phone\_verify\_progressBar*)  
 val getOtpButton: Button = findViewById(R.id.*phone\_verify\_get\_otp*)  
 getOtpButton.setOnClickListener **{** val phoneNumber = phoneNumberInput.*text*.*toString*().*trim*()  
 if (validatePhoneNumber(phoneNumber)) {  
 progressBar.*visibility* = View.*VISIBLE* sendOtp(phoneNumber)  
 } else {  
 phoneNumberInput.*error* = "Invalid phone number"  
 }  
 **}** }  
 private fun validatePhoneNumber(phoneNumber: String): Boolean {  
 return phoneNumber.length == 10 && phoneNumber.*all* **{ it**.*isDigit*() **}** }  
 private fun sendOtp(phoneNumber: String) {  
 PhoneAuthProvider.getInstance().verifyPhoneNumber(  
 "+91$phoneNumber",  
 60,  
 TimeUnit.*SECONDS*,  
 this,  
 object : PhoneAuthProvider.OnVerificationStateChangedCallbacks() {  
 override fun onVerificationCompleted(credential: PhoneAuthCredential) {  
 progressBar.*visibility* = View.*GONE* signInWithPhoneAuthCredential(credential)  
 }  
 override fun onVerificationFailed(e: FirebaseException) {  
 progressBar.*visibility* = View.*GONE* Toast.makeText(this@PhoneAuthenticationActivity, "Verification Failed: ${e.message}", Toast.*LENGTH\_SHORT*).show()  
 }  
 override fun onCodeSent(  
 verificationId: String,  
 token: PhoneAuthProvider.ForceResendingToken  
 ) {  
 super.onCodeSent(verificationId, token)  
 this@PhoneAuthenticationActivity.verificationId = verificationId  
 navigateToOtpVerification()  
 }  
 })  
 }  
 private fun signInWithPhoneAuthCredential(credential: PhoneAuthCredential) {  
 auth.signInWithCredential(credential)  
 .addOnCompleteListener(this) **{** task **->** if (task.*isSuccessful*) {  
 navigateToOtpVerification()  
 } else {  
 Toast.makeText(  
 this@PhoneAuthenticationActivity,  
 "Authentication failed: ${task.*exception*?.message}",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 **}** }  
 private fun navigateToOtpVerification() {  
 val intent = Intent(this@PhoneAuthenticationActivity, OtpVerifyActivity::class.*java*)  
 intent.putExtra("verificationId", verificationId)  
 startActivity(intent)  
 finish()   
 }  
}

**Otp Verify Activity**

package com.example.finalsemester  
  
import android.content.Intent  
import android.os.Bundle  
import android.util.Log  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.auth.PhoneAuthProvider  
  
class OtpVerifyActivity : AppCompatActivity() {  
 private lateinit var verificationId: String  
 private lateinit var phoneNumber: String  
 private val TAG = "OtpVerifyActivity"  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_otp\_verify*)  
 verificationId = *intent*.getStringExtra("verificationId") ?: ""  
 phoneNumber = *intent*.getStringExtra("phoneNumber") ?: ""  
 val otpVerifyButton: Button = findViewById(R.id.*otp\_verify\_button*)  
 val otpInput: EditText = findViewById(R.id.*otp\_verify\_input*)  
 otpVerifyButton.setOnClickListener **{** val otpCode = otpInput.*text*.toString().*trim*()  
 verifyOtp(verificationId, otpCode)  
 **}** }  
 private fun verifyOtp(verificationId: String, otpCode: String) {  
 try {  
 val credential = PhoneAuthProvider.getCredential(verificationId, otpCode)  
 FirebaseAuth.getInstance().signInWithCredential(credential)  
 .addOnCompleteListener(this) **{** task **->** if (task.*isSuccessful*) {  
 Toast.makeText(this, "Verification successful", Toast.*LENGTH\_SHORT*).show()  
 startActivity(Intent(this, SuccessfulPhoneVerificationActivity::class.*java*))  
 finish()  
 } else {  
 Toast.makeText(this, "Verification failed: ${task.*exception*?.message}", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}** } catch (e: Exception) {  
 Log.e(TAG, "Error verifying OTP", e)  
 Toast.makeText(this, "Error verifying OTP: ${e.message}", Toast.*LENGTH\_SHORT*).show()  
 }  
 }  
 private fun sanitizePhoneNumber(phoneNumber: String): String {  
 return phoneNumber.*replace*("[^0-9]".*toRegex*(), "")  
 }  
}

**Offers Rides Code**

package com.example.finalsemester  
  
import android.content.Intent  
import android.net.Uri  
import android.os.Bundle  
import android.util.Log  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.ImageView  
import android.widget.TextView  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import androidx.recyclerview.widget.LinearLayoutManager  
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 CarOwnersActivity : AppCompatActivity() {  
 private lateinit var database: DatabaseReference  
 private lateinit var adapter: CarOwnerAdapter  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_car\_owners*)  
 database = FirebaseDatabase.getInstance().getReference().child("car\_owners")  
 val recyclerView = findViewById<RecyclerView>(R.id.*carOwnersRecyclerView*)  
 recyclerView.*layoutManager* = LinearLayoutManager(this)  
 adapter = CarOwnerAdapter()  
 recyclerView.*adapter* = adapter  
 loadCarOwners()  
 }  
 private fun loadCarOwners() {  
 database.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(dataSnapshot: DataSnapshot) {  
 if (dataSnapshot.exists()) {  
 val carOwnersList = *mutableListOf*<CarOwner>()  
 for (ownerSnapshot in dataSnapshot.*children*) {  
 val carOwner = ownerSnapshot.getValue(CarOwner::class.*java*)  
 carOwner?.*let* **{** carOwnersList.add(**it**)  
 **}** }  
 adapter.setCarOwners(carOwnersList)  
 }  
 }  
 override fun onCancelled(databaseError: DatabaseError) {  
 Log.e(TAG, "Error loading car owners", databaseError.toException())  
 Toast.makeText(  
 this@CarOwnersActivity,  
 "Failed to load car owners",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 })  
 }  
 companion object {  
 private const val TAG = "CarOwnersActivity"  
 }  
 inner class CarOwnerAdapter : RecyclerView.Adapter<CarOwnerAdapter.CarOwnerViewHolder>() {  
 private var carOwners: List<CarOwner> = *emptyList*()  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): CarOwnerViewHolder {  
 val itemView = LayoutInflater.from(parent.*context*)  
 .inflate(R.layout.*item\_owner*, parent, false)  
 return CarOwnerViewHolder(itemView)  
 }  
 override fun onBindViewHolder(holder: CarOwnerViewHolder, position: Int) {  
 val currentCarOwner = carOwners[position]  
 holder.bind(currentCarOwner)  
 }  
 override fun getItemCount(): Int {  
 return carOwners.size  
 }  
 fun setCarOwners(newCarOwners: List<CarOwner>) {  
 carOwners = newCarOwners  
 notifyDataSetChanged()  
 }  
 inner class CarOwnerViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 private val nameTextView: TextView = itemView.findViewById(R.id.*card\_name*)  
 private val costTextView: TextView = itemView.findViewById(R.id.*cost*)  
 private val sourceTextView: TextView = itemView.findViewById(R.id.*card\_source*)  
 private val destinationTextView: TextView = itemView.findViewById(R.id.*card\_student\_board*)  
 private val seatsTextView: TextView = itemView.findViewById(R.id.*card\_seats*)  
 private val dateTimeTextView: TextView = itemView.findViewById(R.id.*card\_date*)  
 private val currentLocationTextView: TextView = itemView.findViewById(R.id.*card\_current\_address*)  
 private val emailImageView: ImageView = itemView.findViewById(R.id.*card\_email\_button*)  
 private val phoneNumberImageView: ImageView = itemView.findViewById(R.id.*card\_call\_button*)  
 private val smsImageView: ImageView = itemView.findViewById(R.id.*card\_sms\_button*)  
 fun bind(carOwner: CarOwner) {  
 nameTextView.*text* = carOwner.name  
 costTextView.*text* = "Cost: ${carOwner.costPerRide}"  
 sourceTextView.*text* = carOwner.startLocation  
 destinationTextView.*text* = carOwner.destination  
 seatsTextView.*text* = "${carOwner.seatsAvailable}"  
 dateTimeTextView.*text* = "${carOwner.dateTime}"  
 currentLocationTextView.*text* = "Current Location: ${carOwner.currentLocation}"  
 emailImageView.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_SEND*).*apply* **{** *type* = "message/rfc822"  
 putExtra(Intent.*EXTRA\_EMAIL*, *arrayOf*(carOwner.email))  
 putExtra(Intent.*EXTRA\_SUBJECT*, "Subject Here")  
 putExtra(Intent.*EXTRA\_TEXT*, "Email body here.")  
 **}** itemView.*context*.startActivity(Intent.createChooser(intent, "Send Email"))  
 **}** phoneNumberImageView.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_DIAL*).*apply* **{** *data* = Uri.parse("tel:${carOwner.phoneNumber}")  
 **}** itemView.*context*.startActivity(intent)  
 **}** smsImageView.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_SENDTO*).*apply* **{** *data* = Uri.parse("smsto:${carOwner.phoneNumber}")  
 **}** itemView.*context*.startActivity(intent)  
 **}** nameTextView.setOnClickListener **{** val intent = Intent(itemView.*context*, CarOwnerProfileDetails::class.*java*)  
 intent.putExtra("carOwnerName", carOwner.name)  
 intent.putExtra("carOwnerCost", carOwner.costPerRide)  
 intent.putExtra("carOwnerSource", carOwner.startLocation)  
 intent.putExtra("carOwnerDestination", carOwner.destination)  
 intent.putExtra("carOwnerSeats", carOwner.seatsAvailable)  
 intent.putExtra("carOwnerDateTime", carOwner.dateTime)  
 intent.putExtra("carOwnerCurrentLocation", carOwner.currentLocation)  
 intent.putExtra("carOwnerEmail", carOwner.email)  
 intent.putExtra("carOwnerPhoneNumber", carOwner.phoneNumber)  
 itemView.*context*.startActivity(intent)  
 **}** }  
 }  
 }  
}

**Details of Car owner information**

package com.example.finalsemester  
  
import android.content.Intent  
import android.os.Bundle  
import android.widget.Button  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
  
class CarOwnerProfileDetails : AppCompatActivity() {  
 private lateinit var nameTextView: TextView  
 private lateinit var profile\_mobileNumber: TextView  
 private lateinit var profileEmailTextView: TextView  
 private lateinit var costTxtView: TextView  
 private lateinit var currentLocationTextView: TextView  
 private lateinit var startlocation: TextView  
 private lateinit var destinationTextView: TextView  
 private lateinit var seatsTextView: TextView  
 private lateinit var bookButton: Button  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_car\_owner\_profile\_details*)  
 nameTextView = findViewById(R.id.*nameTextView*)  
 profileEmailTextView = findViewById(R.id.*profile\_email\_input*)  
 profile\_mobileNumber = findViewById(R.id.*profile\_mobileNumber*)  
 costTxtView = findViewById(R.id.*costtxtview*)  
 currentLocationTextView = findViewById(R.id.*currentlocation*)  
 startlocation = findViewById(R.id.*startlocation*)  
 destinationTextView = findViewById(R.id.*destination*)  
 seatsTextView = findViewById(R.id.*seats*)  
 bookButton = findViewById(R.id.*bookButton*)  
 val carOwnerName = *intent*.getStringExtra("carOwnerName") ?: ""  
 val carOwnerCost = *intent*.getStringExtra("carOwnerCost") ?: ""  
 val carOwnerMobile = *intent*.getStringExtra("carOwnerPhoneNumber") ?: ""  
 val carOwnerEmail = *intent*.getStringExtra("carOwnerEmail") ?: ""  
 val carOwnerCurrentLocation = *intent*.getStringExtra("carOwnerCurrentLocation") ?: ""  
 val carOwnerStartLocation = *intent*.getStringExtra("carOwnerSource") ?: ""  
 val carOwnerDestination = *intent*.getStringExtra("carOwnerDestination") ?: ""  
 val carOwnerSeats = *intent*.getStringExtra("carOwnerSeats") ?: ""  
  
 nameTextView.*text* = carOwnerName  
 costTxtView.*text* = carOwnerCost  
 profile\_mobileNumber.*text* = carOwnerMobile  
 profileEmailTextView.*text* = carOwnerEmail  
 currentLocationTextView.*text* = carOwnerCurrentLocation  
 startlocation.*text* = carOwnerStartLocation  
 destinationTextView.*text* = carOwnerDestination  
 seatsTextView.*text* = carOwnerSeats  
   
 bookButton.setOnClickListener **{** val intent = Intent(this, RiderProfileSetupActivity::class.*java*)  
 intent.putExtra("carOwnerName", nameTextView.*text*.toString())  
 intent.putExtra("carOwnerCost", costTxtView.*text*.toString())  
 intent.putExtra("carOwnerPhoneNumber", profile\_mobileNumber.*text*.toString())  
 intent.putExtra("carOwnerEmail", profileEmailTextView.*text*.toString())  
 intent.putExtra("carOwnerCurrentLocation", currentLocationTextView.*text*.toString())  
 intent.putExtra("carOwnerSource", startlocation.*text*.toString())  
 intent.putExtra("carOwnerDestination", destinationTextView.*text*.toString())  
 intent.putExtra("carOwnerSeats", seatsTextView.*text*.toString())  
 startActivity(intent)  
 **}** }  
}

**Book a ride Code**

package com.example.finalsemester  
  
import android.content.Intent  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AlertDialog  
import androidx.appcompat.app.AppCompatActivity  
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  
import kotlinx.coroutines.Dispatchers  
import kotlinx.coroutines.GlobalScope  
import kotlinx.coroutines.launch  
import java.util.Properties  
import javax.mail.Authenticator  
import javax.mail.Message  
import javax.mail.PasswordAuthentication  
import javax.mail.Session  
import javax.mail.Transport  
import javax.mail.internet.InternetAddress  
import javax.mail.internet.MimeMessage  
  
class RiderProfileSetupActivity : AppCompatActivity() {  
 private lateinit var displayNameEditText: EditText  
 private lateinit var phoneNumberEditText: EditText  
 private lateinit var emailEditText: EditText  
 private lateinit var dateEditText: EditText  
 private lateinit var startLocationEditText: EditText  
 private lateinit var destinationEditText: EditText  
 private lateinit var currentLocationEditText: EditText  
 private lateinit var requiredTimeEditText: EditText  
 private lateinit var ridersRef: DatabaseReference  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_rider\_profile\_setup*)  
 displayNameEditText = findViewById(R.id.*displayNameEditText*)  
 phoneNumberEditText = findViewById(R.id.*phoneNumberEditText*)  
 emailEditText = findViewById(R.id.*emailEditText*)  
 dateEditText = findViewById(R.id.*dateEditText*)  
 startLocationEditText = findViewById(R.id.*startLocationEditText*)  
 destinationEditText = findViewById(R.id.*destinationEditText*)  
 currentLocationEditText = findViewById(R.id.*currentLocationEditText*)  
 requiredTimeEditText = findViewById(R.id.*requiredTimeEditText*)  
 val carOwnerName = *intent*.getStringExtra("carOwnerName") ?: ""  
 val carOwnerCost = *intent*.getStringExtra("carOwnerCost") ?: ""  
 val carOwnerPhoneNumber = *intent*.getStringExtra("carOwnerPhoneNumber") ?: ""  
 val carOwnerEmail = *intent*.getStringExtra("carOwnerEmail") ?: ""  
 val carOwnerCurrentLocation = *intent*.getStringExtra("carOwnerCurrentLocation") ?: ""  
 val carOwnerStartLocation = *intent*.getStringExtra("carOwnerSource") ?: ""  
 val carOwnerDestination = *intent*.getStringExtra("carOwnerDestination") ?: ""  
 val carOwnerSeats = *intent*.getStringExtra("carOwnerSeats") ?: ""  
 val carOwnerDateTime = *intent*.getStringExtra("carOwnerDateTime") ?: ""  
 ridersRef = FirebaseDatabase.getInstance().getReference("riders")  
 findViewById<Button>(R.id.*saveButton*)?.setOnClickListener **{** saveRiderInformation()  
 **}** }  
 private fun saveRiderInformation() {  
 val displayName = displayNameEditText.*text*.toString().*trim*()  
 val phoneNumber = phoneNumberEditText.*text*.toString().*trim*()  
 val email = emailEditText.*text*.toString().*trim*()  
 val date = dateEditText.*text*.toString().*trim*()  
 val startLocation = startLocationEditText.*text*.toString().*trim*()  
 val destination = destinationEditText.*text*.toString().*trim*()  
 val currentLocation = currentLocationEditText.*text*.toString().*trim*()  
 val requiredTime = requiredTimeEditText.*text*.toString().*trim*()  
  
 if (displayName.*isEmpty*() || phoneNumber.*isEmpty*() || email.*isEmpty*() ||  
 date.*isEmpty*() || startLocation.*isEmpty*() || destination.*isEmpty*() ||  
 currentLocation.*isEmpty*() || requiredTime.*isEmpty*()  
 ) {  
 Toast.makeText(this, "Please fill in all required fields", Toast.*LENGTH\_SHORT*).show()  
 return  
 }  
  
 val riderInfo = *mapOf*(  
 "displayName" *to* displayName,  
 "phoneNumber" *to* phoneNumber,  
 "email" *to* email,  
 "date" *to* date,  
 "startLocation" *to* startLocation,  
 "destination" *to* destination,  
 "currentLocation" *to* currentLocation,  
 "requiredTime" *to* requiredTime  
 )  
  
 sendEmail(email, riderInfo)  
  
 val riderId = ridersRef.push().*key* ?: ""  
 ridersRef.child(riderId).setValue(riderInfo)  
 .addOnSuccessListener **{** showConfirmationDialog(riderId)  
 **}** .addOnFailureListener **{** e **->** showToast("Error saving rider information: ${e.message}")  
 clearEditTextFields()  
 **}** }  
  
 private fun showConfirmationDialog(riderId: String) {  
 val carOwnerName = *intent*.getStringExtra("carOwnerName") ?: ""  
 val carOwnerSeatsStr = *intent*.getStringExtra("carOwnerSeats") ?: "0"  
 val carOwnerSeats = carOwnerSeatsStr.*toIntOrNull*() ?: 0  
  
 val updatedSeats = carOwnerSeats - 1  
  
 val displayName = displayNameEditText.*text*.toString().*trim*()  
 val phoneNumber = phoneNumberEditText.*text*.toString().*trim*()  
 val email = emailEditText.*text*.toString().*trim*()  
 val date = dateEditText.*text*.toString().*trim*()  
 val startLocation = startLocationEditText.*text*.toString().*trim*()  
 val destination = destinationEditText.*text*.toString().*trim*()  
 val currentLocation = currentLocationEditText.*text*.toString().*trim*()  
 val requiredTime = requiredTimeEditText.*text*.toString().*trim*()  
  
 if (updatedSeats < 0) {  
 val dialogBuilder = AlertDialog.Builder(this)  
 dialogBuilder.setMessage("No seats are available for this car.")  
 .setTitle("Please try another car")  
 .setCancelable(false)  
 .setIcon(R.drawable.*error*)  
 .setPositiveButton("OK") **{** dialog, \_ **->** dialog.dismiss()  
 clearEditTextFields()  
 **}** val alertDialog = dialogBuilder.create()  
 alertDialog.show()  
 } else {  
  
 val dialogMessage = "\n\n" +  
 "Updated seats for this car: $updatedSeats\n\n" +  
 "Your Details:\n" +  
 "Name: $displayName\n" +  
 "Phone Number: $phoneNumber\n" +  
 "Email: $email\n" +  
 "Date: $date\n" +  
 "Start Location: $startLocation\n" +  
 "Destination: $destination\n" +  
 "Current Location: $currentLocation\n" +  
 "Time: $requiredTime\n" +  
 "Click on OK for verifying your phone number!!"  
  
 val dialogBuilder = AlertDialog.Builder(this)  
 dialogBuilder.setMessage(dialogMessage)  
 .setTitle("Saved Riders Information successfully.\n")  
 .setCancelable(false)  
 .setIcon(R.drawable.*tick1*)  
 .setPositiveButton("OK") **{** dialog, \_ **->** val intent = Intent(this, PhoneAuthenticationActivity::class.*java*)  
 intent.putExtra("displayName", displayName)  
 intent.putExtra("phoneNumber", phoneNumber)  
 startActivity(intent)  
  
 val carOwnersRef = FirebaseDatabase.getInstance().getReference("car\_owners")  
 val query = carOwnersRef.orderByChild("name").equalTo(carOwnerName)  
  
 query.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(snapshot: DataSnapshot) {  
 if (snapshot.exists()) {  
 for (carOwnerSnapshot in snapshot.*children*) {  
 val carOwnerId = carOwnerSnapshot.*key* ?: ""  
 val carOwnerRef = carOwnersRef.child(carOwnerId)  
 carOwnerRef.child("seatsAvailable").setValue(updatedSeats.toString())  
 }  
 }  
 dialog.dismiss()  
  
 }  
 override fun onCancelled(error: DatabaseError) {  
 dialog.dismiss()  
 showToast("Failed to update seat count")  
 }  
 })  
 **}** .setNegativeButton("Cancel") **{** dialog, \_ **->** dialog.dismiss()  
 clearEditTextFields()  
 **}** val alertDialog = dialogBuilder.create()  
 alertDialog.show()  
 }  
 }  
 private fun showToast(message: String) {  
 Toast.makeText(this, message, Toast.*LENGTH\_SHORT*).show()  
 }  
 private fun sendEmail(recipientEmail: String, userDetails: Map<String, String>) {  
 GlobalScope.*launch*(Dispatchers.IO) **{** try {  
 val props = Properties()  
 props["mail.smtp.host"] = "smtp.gmail.com"  
 props["mail.smtp.port"] = "587"  
 props["mail.smtp.auth"] = "true"  
 props["mail.smtp.starttls.enable"] = "true"  
 val session = Session.getInstance(props, object : Authenticator() {  
 override fun getPasswordAuthentication(): PasswordAuthentication {  
 return PasswordAuthentication("gopikrishnaguntamukkala3@gmail.com", "zpuebnvfffveyfou")  
 }  
 })  
 val message = MimeMessage(session)  
 message.setFrom(InternetAddress("gopikrishnaguntamukkala3@gmail.com"))  
 message.addRecipient(Message.RecipientType.*TO*, InternetAddress(recipientEmail))  
 message.*subject* = "Successfully Booked a Ride"  
  
 val messageBody = *buildString* **{** append("Dear ${userDetails["displayName"]} ,\n\n")  
 append("Thank you for booking a ride and verifying your phone number (it's mandatory).\n\n")  
 append("Your Details:\n")  
 append("Name: ${userDetails["displayName"]}\n")  
 append("Phone Number: ${userDetails["phoneNumber"]}\n")  
 append("Email: ${userDetails["email"]}\n")  
 append("Current Location: ${userDetails["currentLocation"]}\n")  
 append("Start Location: ${userDetails["startLocation"]}\n")  
 append("Destination: ${userDetails["destination"]}\n")  
 append("Date: ${userDetails["date"]}\n")  
 append("Required Time: ${userDetails["requiredTime"]}\n\n")  
 append("Best regards,\nG. Gopi Krishna\nFinal Year BTech CSE\nLovely Professional University, India.\n\nMobile: 76590 46696\nEmail: gopikrishnaguntamukkala3@gmail.com")  
 **}** message.setText(messageBody)  
 Transport.send(message)  
 runOnUiThread **{** Toast.makeText(this@RiderProfileSetupActivity, "Email sent successfully", Toast.*LENGTH\_SHORT*).show()  
 **}** } catch (e: Exception) {  
 e.printStackTrace()  
 runOnUiThread **{** Toast.makeText(this@RiderProfileSetupActivity, "Failed to send email: ${e.message}", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
 **}** }  
 private fun clearEditTextFields() {  
 displayNameEditText.*text*.clear()  
 phoneNumberEditText.*text*.clear()  
 emailEditText.*text*.clear()  
 dateEditText.*text*.clear()  
 startLocationEditText.*text*.clear()  
 destinationEditText.*text*.clear()  
 currentLocationEditText.*text*.clear()  
 requiredTimeEditText.*text*.clear()  
 }  
}

**Near By Cars Code**

package com.example.finalsemester  
  
import android.Manifest  
import android.content.pm.PackageManager  
import android.graphics.Bitmap  
import android.graphics.drawable.BitmapDrawable  
import android.location.Geocoder  
import android.os.Bundle  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.app.ActivityCompat  
import androidx.core.content.ContextCompat  
import androidx.core.graphics.drawable.RoundedBitmapDrawableFactory  
import com.google.android.gms.location.FusedLocationProviderClient  
import com.google.android.gms.location.LocationServices  
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  
import org.osmdroid.config.Configuration  
import org.osmdroid.tileprovider.tilesource.TileSourceFactory  
import org.osmdroid.util.GeoPoint  
import org.osmdroid.views.MapView  
import org.osmdroid.views.overlay.Marker  
import org.osmdroid.views.overlay.mylocation.GpsMyLocationProvider  
import org.osmdroid.views.overlay.mylocation.MyLocationNewOverlay  
import java.io.IOException  
  
class NearByCars : AppCompatActivity() {  
 private lateinit var map: MapView  
 private lateinit var fusedLocationClient: FusedLocationProviderClient  
 private lateinit var database: DatabaseReference  
 companion object {  
 private const val REQUEST\_LOCATION\_PERMISSION = 1  
 }  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_near\_by\_cars*)  
 Configuration.getInstance().load(this, getSharedPreferences("osmdroid", *MODE\_PRIVATE*))  
 map = findViewById(R.id.*map*)  
 map.setTileSource(TileSourceFactory.*MAPNIK*)  
 map.setMultiTouchControls(true)  
 val initialCenter = GeoPoint(31.5018, 75.5728)  
 map.*controller*.setCenter(initialCenter)  
 map.*controller*.setZoom(11.0)  
 if (ContextCompat.checkSelfPermission(this, Manifest.permission.*ACCESS\_FINE\_LOCATION*) == PackageManager.*PERMISSION\_GRANTED*) {  
 initializeMap()  
 } else {  
 ActivityCompat.requestPermissions(  
 this,  
 *arrayOf*(Manifest.permission.*ACCESS\_FINE\_LOCATION*),  
 REQUEST\_LOCATION\_PERMISSION  
 )  
 }  
 database = FirebaseDatabase.getInstance().*reference*.child("car\_owners")  
 retrieveCarOwnerInformation()  
 }  
 private fun initializeMap() {  
 val locationOverlay = MyLocationNewOverlay(GpsMyLocationProvider(this), map)  
 locationOverlay.enableMyLocation()  
 map.*overlays*.add(locationOverlay)  
 fusedLocationClient = LocationServices.getFusedLocationProviderClient(this)  
 if (ActivityCompat.checkSelfPermission(this, Manifest.permission.*ACCESS\_FINE\_LOCATION*) == PackageManager.*PERMISSION\_GRANTED*) {  
 fusedLocationClient.*lastLocation* .addOnSuccessListener **{** location **->** if (location != null) {  
 val geocoder = Geocoder(this)  
 try {  
 val addressList = geocoder.getFromLocation(location.*latitude*, location.*longitude*, 1)  
 if (addressList != null && addressList.*isNotEmpty*()) {  
 val address = addressList[0]  
 val currentLocation = GeoPoint(location.*latitude*, location.*longitude*)  
 addMarkerAtLocation(currentLocation, "My Location", address.getAddressLine(0), true)  
 } else {  
 addDefaultMarker()  
 }  
 } catch (e: IOException) {  
 e.printStackTrace()  
 addDefaultMarker()  
 }  
 }  
 **}** }  
 }  
 private fun addDefaultMarker() {  
 val defaultLocation = GeoPoint(31.2565, 75.6509)  
 val marker = Marker(map)  
 marker.*position* = defaultLocation  
 marker.*title* = "Default Location"  
 marker.*snippet* = "Lovely Professional University"  
 marker.*icon* = ContextCompat.getDrawable(this, R.drawable.*marker\_red*)  
 map.*overlays*.add(marker)  
 }  
 private fun retrieveCarOwnerInformation() {  
 database.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(dataSnapshot: DataSnapshot) {  
 if (dataSnapshot.exists()) {  
 for (ownerSnapshot in dataSnapshot.*children*) {  
 val carOwner = ownerSnapshot.getValue(CarOwner::class.*java*)  
 carOwner?.*let* **{** val currentLocation = carOwner.currentLocation ?: ""  
 if (currentLocation.*isNotBlank*()) {  
 val geocoder = Geocoder(this@NearByCars)  
 try {  
 val addressList = geocoder.getFromLocationName(currentLocation, 1)  
 if (addressList != null && addressList.*isNotEmpty*()) {  
 val address = addressList[0]  
 val carOwnerLocation = GeoPoint(address.*latitude*, address.*longitude*)  
 val snippet = "Current Location: $currentLocation\n" +  
 "Start Location: ${carOwner.startLocation ?: ""}\n" +  
 "Destination: ${carOwner.destination ?: ""}"  
 addMarkerAtLocation(carOwnerLocation, carOwner.name ?: "", snippet, false)  
 }  
 } catch (e: IOException) {  
 e.printStackTrace()  
 }  
 }  
 **}** }  
 }  
 }  
 override fun onCancelled(databaseError: DatabaseError) {  
 }  
 })  
 }  
 private fun addMarkerAtLocation(location: GeoPoint, title: String, snippet: String, isMyLocation: Boolean) {  
 val marker = Marker(map)  
 marker.*position* = location  
 marker.*title* = title  
 marker.*snippet* = snippet  
 val markerDrawable = if (isMyLocation) {  
 ContextCompat.getDrawable(this, R.drawable.*round111*)  
 } else {  
 ContextCompat.getDrawable(this, R.drawable.*round1*)  
 }  
 markerDrawable?.*let* **{** val width = *resources*.getDimension(R.dimen.*marker\_size*).toInt()  
 val height = *resources*.getDimension(R.dimen.*marker\_size*).toInt()  
 val resizedBitmap = Bitmap.createScaledBitmap((**it** as BitmapDrawable).*bitmap*, width, height, false)  
 val roundedBitmapDrawable = RoundedBitmapDrawableFactory.create(*resources*, resizedBitmap)  
 roundedBitmapDrawable.*isCircular* = true  
 marker.*icon* = BitmapDrawable(*resources*, roundedBitmapDrawable.*bitmap*)  
 **}** map.*overlays*.add(marker)  
 marker.showInfoWindow()  
 }  
 override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out String>, grantResults: IntArray) {  
 super.onRequestPermissionsResult(requestCode, permissions, grantResults)  
 if (requestCode == REQUEST\_LOCATION\_PERMISSION) {  
 if (grantResults.*isNotEmpty*() && grantResults[0] == PackageManager.*PERMISSION\_GRANTED*) {  
 initializeMap()  
 } else {  
 }  
 }  
 }  
 override fun onResume() {  
 super.onResume()  
 map.onResume()  
 }  
 override fun onPause() {  
 super.onPause()  
 map.onPause()  
 }  
}

**Search (for riders)**

package com.example.finalsemester  
  
import android.content.Intent  
import android.net.Uri  
import android.os.Bundle  
import android.util.Log  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.Button  
import android.widget.EditText  
import android.widget.ImageView  
import android.widget.TextView  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import androidx.recyclerview.widget.LinearLayoutManager  
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.Query  
import com.google.firebase.database.ValueEventListener  
  
class SearchCarsActivity : AppCompatActivity() {  
 private lateinit var database: DatabaseReference  
 private lateinit var adapter: CarOwnerAdapter  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_search\_cars*)  
 database = FirebaseDatabase.getInstance().getReference().child("car\_owners")  
 val recyclerView = findViewById<RecyclerView>(R.id.*carOwnersRecyclerView*)  
 recyclerView.*layoutManager* = LinearLayoutManager(this)  
 adapter = CarOwnerAdapter()  
 recyclerView.*adapter* = adapter  
 loadCarOwners()  
 val editTextStartLocation = findViewById<EditText>(R.id.*editTextStartLocation*)  
 val editTextEndLocation = findViewById<EditText>(R.id.*editTextEndLocation*)  
 val buttonSearch = findViewById<Button>(R.id.*buttonSearch*)  
 buttonSearch.setOnClickListener **{** val startLocation = editTextStartLocation.*text*.toString().*trim*()  
 val endLocation = editTextEndLocation.*text*.toString().*trim*()  
 if (startLocation.*isEmpty*() || endLocation.*isEmpty*()) {  
 Toast.makeText(this@SearchCarsActivity, "Please enter both locations", Toast.*LENGTH\_SHORT*).show()  
 } else {  
 searchCarOwners(startLocation, endLocation)  
 }  
 **}** val imageViewSwap = findViewById<ImageView>(R.id.*imageViewSwap*)  
 val imageClear = findViewById<ImageView>(R.id.*imageClear*)  
 imageViewSwap.setOnClickListener **{** val startText = editTextStartLocation.*text*.toString()  
 val endText = editTextEndLocation.*text*.toString()  
 editTextStartLocation.setText(endText)  
 editTextEndLocation.setText(startText)  
 **}** imageClear.setOnClickListener **{** editTextStartLocation.setText("")  
 editTextEndLocation.setText("")  
 **}** }  
 private fun searchCarOwners(startLocation: String, endLocation: String) {  
 val query: Query = database.orderByChild("startLocation").equalTo(startLocation)  
 query.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(dataSnapshot: DataSnapshot) {  
 val carOwnersList = *mutableListOf*<CarOwner>()  
 for (ownerSnapshot in dataSnapshot.*children*) {  
 val carOwner = ownerSnapshot.getValue(CarOwner::class.*java*)  
 carOwner?.*let* **{** if (**it**.destination == endLocation) {  
 carOwnersList.add(**it**)  
 }  
 **}** }  
 if (carOwnersList.*isNotEmpty*()) {  
 adapter.setCarOwners(carOwnersList)  
 } else {  
 Toast.makeText(  
 this@SearchCarsActivity,  
 "No car owners found for the specified locations",  
 Toast.*LENGTH\_SHORT* ).show()  
 loadCarOwners()  
 }  
 }  
 override fun onCancelled(databaseError: DatabaseError) {  
 Log.e(TAG, "Error searching car owners", databaseError.toException())  
 Toast.makeText(  
 this@SearchCarsActivity,  
 "Failed to search car owners",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 })  
 }  
 private fun loadCarOwners() {  
 database.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(dataSnapshot: DataSnapshot) {  
 if (dataSnapshot.exists()) {  
 val carOwnersList = *mutableListOf*<CarOwner>()  
 for (ownerSnapshot in dataSnapshot.*children*) {  
 val carOwner = ownerSnapshot.getValue(CarOwner::class.*java*)  
 carOwner?.*let* **{** carOwnersList.add(**it**)  
 **}** }  
 adapter.setCarOwners(carOwnersList)  
 } else {  
 Toast.makeText(  
 this@SearchCarsActivity,  
 "No car owners found",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 }  
 override fun onCancelled(databaseError: DatabaseError) {  
 Log.e(TAG, "Error loading car owners", databaseError.toException())  
 Toast.makeText(  
 this@SearchCarsActivity,  
 "Failed to load car owners",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 })  
 }  
 companion object {  
 private const val TAG = "SearchCarsActivity"  
 }  
 inner class CarOwnerAdapter : RecyclerView.Adapter<CarOwnerAdapter.CarOwnerViewHolder>() {  
 private var carOwners: List<CarOwner> = *emptyList*()  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): CarOwnerViewHolder {  
 val itemView = LayoutInflater.from(parent.*context*)  
 .inflate(R.layout.*item\_owner*, parent, false)  
 return CarOwnerViewHolder(itemView)  
 }  
 override fun onBindViewHolder(holder: CarOwnerViewHolder, position: Int) {  
 val currentCarOwner = carOwners[position]  
 holder.bind(currentCarOwner)  
 }  
 override fun getItemCount(): Int {  
 return carOwners.size  
 }  
 fun setCarOwners(newCarOwners: List<CarOwner>) {  
 carOwners = newCarOwners  
 notifyDataSetChanged()  
 }  
 inner class CarOwnerViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 private val nameTextView: TextView = itemView.findViewById(R.id.*card\_name*)  
 private val costTextView: TextView = itemView.findViewById(R.id.*cost*)  
 private val sourceTextView: TextView = itemView.findViewById(R.id.*card\_source*)  
 private val destinationTextView: TextView = itemView.findViewById(R.id.*card\_student\_board*)  
 private val seatsTextView: TextView = itemView.findViewById(R.id.*card\_seats*)  
 private val dateTimeTextView: TextView = itemView.findViewById(R.id.*card\_date*)  
 private val currentLocationTextView: TextView = itemView.findViewById(R.id.*card\_current\_address*)  
 private val emailImageView: ImageView = itemView.findViewById(R.id.*card\_email\_button*)  
 private val phoneNumberImageView: ImageView = itemView.findViewById(R.id.*card\_call\_button*)  
 private val smsImageView: ImageView = itemView.findViewById(R.id.*card\_sms\_button*)  
 fun bind(carOwner: CarOwner) {  
 nameTextView.*text* = carOwner.name  
 costTextView.*text* = "Cost: ${carOwner.costPerRide}"  
 sourceTextView.*text* = carOwner.startLocation  
 destinationTextView.*text* = carOwner.destination  
 seatsTextView.*text* = "${carOwner.seatsAvailable}"  
 dateTimeTextView.*text* = "${carOwner.dateTime}"  
 currentLocationTextView.*text* = "Current Location: ${carOwner.currentLocation}"  
 emailImageView.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_SEND*).*apply* **{** *type* = "message/rfc822"  
 putExtra(Intent.*EXTRA\_EMAIL*, *arrayOf*(carOwner.email))  
 putExtra(Intent.*EXTRA\_SUBJECT*, "Query regarding booking a ride")  
 putExtra(Intent.*EXTRA\_TEXT*, "Hi I would like to tell something about ride... continue")  
 **}** itemView.*context*.startActivity(Intent.createChooser(intent, "Send Email"))  
 **}** phoneNumberImageView.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_DIAL*).*apply* **{** *data* = Uri.parse("tel:${carOwner.phoneNumber}")  
 **}** itemView.*context*.startActivity(intent)  
 **}** smsImageView.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_SENDTO*).*apply* **{** *data* = Uri.parse("smsto:${carOwner.phoneNumber}")  
 **}** itemView.*context*.startActivity(intent)  
 **}** nameTextView.setOnClickListener **{** val intent = Intent(itemView.*context*, CarOwnerProfileDetails::class.*java*)  
 intent.putExtra("carOwnerName", carOwner.name)  
 intent.putExtra("carOwnerCost", carOwner.costPerRide)  
 intent.putExtra("carOwnerSource", carOwner.startLocation)  
 intent.putExtra("carOwnerDestination", carOwner.destination)  
 intent.putExtra("carOwnerSeats", carOwner.seatsAvailable)  
 intent.putExtra("carOwnerDateTime", carOwner.dateTime)  
 intent.putExtra("carOwnerCurrentLocation", carOwner.currentLocation)  
 intent.putExtra("carOwnerEmail", carOwner.email)  
 intent.putExtra("carOwnerPhoneNumber", carOwner.phoneNumber)  
 itemView.*context*.startActivity(intent)  
 **}** }  
 }  
 }  
}

**Edit Profile Activity**

package com.example.finalsemester  
  
import android.app.Activity  
import android.content.Intent  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.database.DatabaseReference  
import com.google.firebase.database.FirebaseDatabase  
  
class EditProfileActivity : AppCompatActivity() {  
 private lateinit var editName: EditText  
 private lateinit var editEmail: EditText  
 private lateinit var editUsername: EditText  
 private lateinit var editPassword: EditText  
 private lateinit var saveButton1: Button  
 private lateinit var nameUser: String  
 private lateinit var emailUser: String  
 private lateinit var usernameUser: String  
 private lateinit var passwordUser: String  
 private lateinit var reference: DatabaseReference  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_edit\_profile*)  
 reference = FirebaseDatabase.getInstance().getReference("users")  
 editName = findViewById(R.id.*editName*)  
 editEmail = findViewById(R.id.*editEmail*)  
 editUsername = findViewById(R.id.*editUsername*)  
 editPassword = findViewById(R.id.*editPassword*)  
 saveButton1 = findViewById(R.id.*saveButton*)  
 val userDetails = *intent*.getSerializableExtra("userDetails") as HashMap<String, String>  
 nameUser = userDetails["name"].*toString*()  
 emailUser = userDetails["email"].*toString*()  
 usernameUser = userDetails["username"].*toString*()  
 passwordUser = userDetails["password"].*toString*()  
 editName.setText(nameUser)  
 editEmail.setText(emailUser)  
 editUsername.setText(usernameUser)  
 editPassword.setText(passwordUser)  
  
 saveButton1.setOnClickListener **{** saveChanges()  
 **}** }  
 private fun saveChanges() {  
 if (isNameChanged() || isEmailChanged() || isPasswordChanged()) {  
 Toast.makeText(this, "Saved", Toast.*LENGTH\_SHORT*).show()  
 val updatedUserDetails = HashMap<String, String>().*apply* **{** put("name", editName.*text*.toString())  
 put("email", editEmail.*text*.toString())  
 put("username", editUsername.*text*.toString())  
 put("password", editPassword.*text*.toString())  
 **}** val resultIntent = Intent()  
 resultIntent.putExtra("updatedUserDetails", updatedUserDetails)  
 setResult(Activity.*RESULT\_OK*, resultIntent)  
 } else {  
 Toast.makeText(this, "No Changes Found", Toast.*LENGTH\_SHORT*).show()  
 }  
 finish()  
 }  
 private fun isNameChanged(): Boolean {  
 val newName = editName.*text*.toString()  
 return if (newName != nameUser) {  
 reference.child(usernameUser).child("name").setValue(newName)  
 true  
 } else {  
 false  
 }  
 }  
 private fun isEmailChanged(): Boolean {  
 val newEmail = editEmail.*text*.toString()  
 return if (newEmail != emailUser) {  
 reference.child(usernameUser).child("email").setValue(newEmail)  
 true  
 } else {  
 false  
 }  
 }  
 private fun isPasswordChanged(): Boolean {  
 val newPassword = editPassword.*text*.toString()  
 return if (newPassword != passwordUser) {  
 reference.child(usernameUser).child("password").setValue(newPassword)  
 true  
 } else {  
 false  
 }  
 }  
}

**Rating Star Activity**

package com.example.finalsemester  
  
import android.annotation.SuppressLint  
import android.app.NotificationChannel  
import android.app.NotificationManager  
import android.content.ContentValues.*TAG*import android.os.Build  
import android.os.Bundle  
import android.provider.Settings  
import android.util.Log  
import android.widget.\*  
import androidx.appcompat.app.AppCompatActivity  
import androidx.core.app.NotificationCompat  
import androidx.core.app.NotificationManagerCompat  
import com.google.firebase.database.\*  
  
class RatingStarActivity : AppCompatActivity() {  
 private val CHANNEL\_ID = "Channel\_id\_example\_1"  
 private val notificationid = 101  
 private lateinit var starsContainer: LinearLayout  
 private lateinit var database: DatabaseReference  
 private lateinit var deviceId: String  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_rating\_star*)  
 database = FirebaseDatabase.getInstance().*reference* deviceId = Settings.Secure.getString(*contentResolver*, Settings.Secure.*ANDROID\_ID*)  
 createNotificationChannel()  
 val rt: RatingBar = findViewById(R.id.*ratingBar*)  
 val submit: Button = findViewById(R.id.*button*)  
 val emoji: ImageView = findViewById(R.id.*emoji*)  
 starsContainer = findViewById(R.id.*starsContainer*)  
 fetchAndDisplayStarRatings()  
 val ratingRef = database.child("ratings").child(deviceId)  
 ratingRef.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(snapshot: DataSnapshot) {  
 if (snapshot.exists()) {  
 val previousRating = snapshot.getValue(Float::class.*java*)  
 previousRating?.*let* **{** rt.*rating* = **it  
 }** }  
 }  
 override fun onCancelled(error: DatabaseError) {  
 Log.e(*TAG*, "Database error: ${error.*message*}")  
 }  
 })  
 rt.*onRatingBarChangeListener* = RatingBar.OnRatingBarChangeListener **{** \_, \_, \_ **->** val rate = rt.*rating* when {  
 rate <= 1 -> emoji.setImageResource(R.drawable.*sad*)  
 rate <= 2 -> emoji.setImageResource(R.drawable.*worried*)  
 rate <= 3 -> emoji.setImageResource(R.drawable.*nuetral*)  
 rate <= 4 -> emoji.setImageResource(R.drawable.*happy*)  
 else -> emoji.setImageResource(R.drawable.*sattisfied*)  
 }  
 submit.setOnClickListener **{** Toast.makeText(this@RatingStarActivity, "Selected Rating: $rate", Toast.*LENGTH\_SHORT*).show()  
 sendNotification(rate)  
 **}  
 }** }  
 private fun createNotificationChannel() {  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*) {  
 val name = "Channel Name"  
 val descriptionText = "Channel Description"  
 val importance = NotificationManager.*IMPORTANCE\_DEFAULT* val channel = NotificationChannel(CHANNEL\_ID, name, importance).*apply* **{** *description* = descriptionText  
 **}** val notificationManager = getSystemService(NotificationManager::class.*java*)  
 notificationManager.createNotificationChannel(channel)  
 }  
 }  
 private fun sendNotification(rating: Float) {  
 val builder = NotificationCompat.Builder(this, CHANNEL\_ID)  
 .setSmallIcon(R.drawable.*notification*)  
 .setContentTitle("Notification")  
 .setContentText("You have successfully given a rating for this app. Your rating: $rating")  
 .setPriority(NotificationCompat.*PRIORITY\_DEFAULT*)  
 try {  
 val notificationManager = NotificationManagerCompat.from(this)  
 notificationManager.notify(notificationid, builder.build())  
 } catch (e: SecurityException) {  
 Log.e(*TAG*, "Failed to send notification: ${e.message}", e)  
 Toast.makeText(this@RatingStarActivity, "Failed to send notification", Toast.*LENGTH\_SHORT*).show()  
 }  
 val database = FirebaseDatabase.getInstance()  
 val ratingRef = database.*reference*.child("ratings").child(deviceId)  
 ratingRef.setValue(rating)  
 .addOnSuccessListener **{** Log.d(*TAG*, "Rating stored successfully: $rating")  
 **}** .addOnFailureListener **{** e **->** Log.e(*TAG*, "Error storing rating: $rating", e)  
 **}** }  
 private fun fetchAndDisplayStarRatings() {  
 val ratingsRef = database.child("ratings")  
 ratingsRef.addListenerForSingleValueEvent(object : ValueEventListener {  
 override fun onDataChange(snapshot: DataSnapshot) {  
 val ratingCounts = *mutableMapOf*<Int, Int>()  
 for (ratingSnapshot in snapshot.*children*) {  
 val rating = ratingSnapshot.getValue(Float::class.*java*)  
 rating?.*let* **{** val starCount = **it**.toInt()  
 if (starCount in 1..5) {  
 if (ratingCounts.containsKey(starCount)) {  
 ratingCounts[starCount] = ratingCounts[starCount]!! + 1  
 } else {  
 ratingCounts[starCount] = 1  
 }  
 }  
 **}** }  
 displayStarRatingCounts(ratingCounts)  
 }  
 override fun onCancelled(error: DatabaseError) {  
 Toast.makeText(this@RatingStarActivity, "Database error: ${error.*message*}", Toast.*LENGTH\_SHORT*).show()  
 }  
 })  
 }  
 private fun displayStarRatingCounts(ratingCounts: Map<Int, Int>) {  
 starsContainer.removeAllViews()  
 val maxStars = 5  
 val starSymbols = *listOf*("★★★★★", "★★★★", "★★★", "★★", "★")  
 var totalUsers = 0  
 var totalRating = 0  
 for (star in maxStars *downTo* 1) {  
 val count = ratingCounts[star] ?: 0  
 totalUsers += count  
 totalRating += star \* count  
 val starRatingText = "$star ${starSymbols[maxStars - star]} - $count users"  
 val textView = TextView(this@RatingStarActivity)  
 textView.*text* = starRatingText  
 textView.*textSize* = 18f  
 textView.setPadding(8, 8, 8, 8)  
 starsContainer.addView(textView)  
 }  
 val averageRating = if (totalUsers > 0) {  
 totalRating.toFloat() / totalUsers  
 } else {  
 0f  
 }  
 val averageRatingText = "Average Rating: ${String.*format*("%.1f", averageRating)} ★"  
 val averageTextView = TextView(this@RatingStarActivity)  
 averageTextView.*text* = averageRatingText  
 averageTextView.*textSize* = 18f  
 averageTextView.setPadding(8, 16, 8, 8)  
 starsContainer.addView(averageTextView)  
 }  
}

**Feedback Form Activity**

package com.example.finalsemester  
import android.app.AlertDialog  
import android.app.DatePickerDialog  
import android.app.TimePickerDialog  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.database.DatabaseReference  
import com.google.firebase.database.FirebaseDatabase  
import java.text.SimpleDateFormat  
import java.util.Calendar  
import java.util.Locale  
class FeedbackForm : AppCompatActivity() {  
 private lateinit var editTextName: EditText  
 private lateinit var editTextPhone: EditText  
 private lateinit var editTextQuery: EditText  
 private lateinit var editTextEmail: EditText  
 private lateinit var btnSubmit: Button  
 private lateinit var btnSelectDate: Button  
 private lateinit var btnSelectTime: Button  
 private lateinit var databaseReference: DatabaseReference  
 private var selectedDate: String? = null  
 private var selectedTime: String? = null  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_feedback\_form*)  
 editTextName = findViewById(R.id.*editTextName*)  
 editTextPhone = findViewById(R.id.*editTextPhone*)  
 editTextQuery = findViewById(R.id.*editTextQuery*)  
 editTextEmail = findViewById(R.id.*editTextEmail*)  
 btnSubmit = findViewById(R.id.*btnSubmit*)  
 btnSelectDate = findViewById(R.id.*btnSelectDate*)  
 btnSelectTime = findViewById(R.id.*btnSelectTime*)  
 databaseReference = FirebaseDatabase.getInstance().*reference*.child("contactInfo")  
 btnSubmit.setOnClickListener **{** onSubmitClick()  
 **}** btnSelectDate.setOnClickListener **{** showDatePicker()  
 **}** btnSelectTime.setOnClickListener **{** showTimePicker()  
 **}** val resetBtn: Button = findViewById(R.id.*btnReset*)  
 resetBtn.setOnClickListener **{** clearFields()  
 **}** }  
 private fun onSubmitClick() {  
 val name = editTextName.*text*.toString().*trim*()  
 val phone = editTextPhone.*text*.toString().*trim*()  
 val query = editTextQuery.*text*.toString().*trim*()  
 val email = editTextEmail.*text*.toString().*trim*()  
 if (name.*isEmpty*() || phone.*isEmpty*() || query.*isEmpty*() || email.*isEmpty*()) {  
 Toast.makeText(this, "Please fill in all the fields", Toast.*LENGTH\_SHORT*).show()  
 return  
 }  
 if (!isValidEmail(email)) {  
 Toast.makeText(this, "Please enter a valid email address", Toast.*LENGTH\_SHORT*).show()  
 return  
 }  
 if (selectedDate.*isNullOrEmpty*() || selectedTime.*isNullOrEmpty*()) {  
 Toast.makeText(this, "Please select date and time", Toast.*LENGTH\_SHORT*).show()  
 return  
 }  
 val contactInfo = ContactInfo(name, phone, email, query)  
 val alertDialogBuilder = AlertDialog.Builder(this)  
 alertDialogBuilder.setTitle("Success")  
 .setIcon(R.drawable.*tick*)  
 .setMessage(  
 "Name: $name\n" +  
 "Phone: $phone\n" +  
 "Email: $email\n" +  
 "Query: $query\n" +  
 "Date: $selectedDate\n" +  
 "Time: $selectedTime"  
 )  
 .setPositiveButton("OK") **{** dialog, which **->** dialog.dismiss()  
 **}** .create()  
 .show()  
 saveFeedback(contactInfo)  
 }  
 private fun isValidEmail(email: String): Boolean {  
 return android.util.Patterns.*EMAIL\_ADDRESS*.matcher(email).matches()  
 }  
 private fun showDatePicker() {  
 val calendar = Calendar.getInstance()  
 val datePickerDialog = DatePickerDialog(  
 this,  
 **{** \_, year, month, dayOfMonth **->** val selectedCalendar = Calendar.getInstance()  
 selectedCalendar.set(year, month, dayOfMonth)  
 val dateFormat = SimpleDateFormat("dd/MM/yyyy", Locale.getDefault())  
 selectedDate = dateFormat.format(selectedCalendar.*time*)  
 **}**,  
 calendar.get(Calendar.*YEAR*),  
 calendar.get(Calendar.*MONTH*),  
 calendar.get(Calendar.*DAY\_OF\_MONTH*)  
 )  
 datePickerDialog.show()  
 }  
 private fun showTimePicker() {  
 val calendar = Calendar.getInstance()  
 val timePickerDialog = TimePickerDialog(  
 this,  
 **{** \_, hourOfDay, minute **->** val selectedCalendar = Calendar.getInstance()  
 selectedCalendar.set(Calendar.*HOUR\_OF\_DAY*, hourOfDay)  
 selectedCalendar.set(Calendar.*MINUTE*, minute)  
 val timeFormat = SimpleDateFormat("hh:mm a", Locale.getDefault())  
 selectedTime = timeFormat.format(selectedCalendar.*time*)  
 **}**,  
 calendar.get(Calendar.*HOUR\_OF\_DAY*),  
 calendar.get(Calendar.*MINUTE*),  
 false  
 )  
 timePickerDialog.show()  
 }  
 private fun saveFeedback(contactInfo: ContactInfo) {  
 val submissionKey = databaseReference.push().*key* if (submissionKey != null) {  
 contactInfo.date = selectedDate ?: ""  
 contactInfo.time = selectedTime ?: ""  
 databaseReference.child(submissionKey).setValue(contactInfo)  
 .addOnSuccessListener **{** Toast.makeText(this, "Successfully submitted your query", Toast.*LENGTH\_SHORT*).show()  
 **}** .addOnFailureListener **{** Toast.makeText(this, "Failed to submit your query", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
 }  
 private fun clearFields() {  
 editTextName.*text*.clear()  
 editTextPhone.*text*.clear()  
 editTextQuery.*text*.clear()  
 editTextEmail.*text*.clear()  
 selectedDate = null  
 selectedTime = null  
 }  
}

**FAQ Activity**

package com.example.finalsemester  
  
import android.os.Bundle  
import android.view.View  
import android.widget.LinearLayout  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
  
class FAQActivity : AppCompatActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_faqactivity*)  
 populateFAQ()  
 }  
 private fun populateFAQ() {  
 val question1Layout = findViewById<LinearLayout>(R.id.*question1Layout*)  
 val question1 = findViewById<TextView>(R.id.*question1*)  
 val answer1 = findViewById<TextView>(R.id.*answer1*)  
 question1.*text* = "Q: What is Car Pooling App?"  
 answer1.*text* = "A: User can choose the rider or car owner based on their needs in this app after login ."  
 val question2Layout = findViewById<LinearLayout>(R.id.*question2Layout*)  
 val question2 = findViewById<TextView>(R.id.*question2*)  
 val answer2 = findViewById<TextView>(R.id.*answer2*)  
 question2.*text* = "Q: Why to use this carpooling app?"  
 answer2.*text* = "A: This app provides options for both car owners and riders. Car owners can create rides, and riders can book available rides."  
 val question3Layout = findViewById<LinearLayout>(R.id.*question3Layout*)  
 val question3 = findViewById<TextView>(R.id.*question3*)  
 val answer3 = findViewById<TextView>(R.id.*answer3*)  
 question3.*text* = "Q: How to book a ride in this app?"  
 answer3.*text* = "A: Users can book a ride by visiting the car owner's profile and selecting an available ride. After selecting a ride, a confirmation will be sent to the user's email."  
 val question4Layout = findViewById<LinearLayout>(R.id.*question4Layout*)  
 val question4 = findViewById<TextView>(R.id.*question4*)  
 val answer4 = findViewById<TextView>(R.id.*answer4*)  
 question4.*text* = "Q: What are the featues in this app?"  
 answer4.*text* = "A:\n" +  
 "- Wifi Check\n" +  
 "- Sign In\n" +  
 "- Sign Up\n" +  
 "- Feedback Form\n" +  
 "- Rating Bar\n" +  
 "- Offer rides\n" +  
 "- Book Rides\n" +  
 "- Nearby Cars and Riders\n" +  
 "- View and Update profile\n" +  
 "- Wifi Information\n" +  
 "- Proximity Sensor Alert \n" +  
 "- Share the app code\n" +  
 "- Log out\n"  
 val question5Layout = findViewById<LinearLayout>(R.id.*question5Layout*)  
 val question5 = findViewById<TextView>(R.id.*question5*)  
 val answer5 = findViewById<TextView>(R.id.*answer5*)  
 question5.*text* = "Q: What safety features are included in this app?"  
 answer5.*text* = "A: Users must verify their phone number to create or book a ride in the carpooling app."  
 val question6Layout = findViewById<LinearLayout>(R.id.*question6Layout*)  
 val question6 = findViewById<TextView>(R.id.*question6*)  
 val answer6 = findViewById<TextView>(R.id.*answer6*)  
 question6.*text* = "Q: Where are user ratings stored?"  
 answer6.*text* = "A: User ratings are stored in Realtime Firebase. Users can view all user ratings at the end of this activity and find the average rating for this car pooling app."  
 question1Layout.*tag* = answer1  
 question2Layout.*tag* = answer2  
 question3Layout.*tag* = answer3  
 question4Layout.*tag* = answer4  
 question5Layout.*tag* = answer5  
 question6Layout.*tag* = answer6  
 }  
 fun toggleAnswer(view: View) {  
 val answerView = view.*tag* as? View  
 answerView?.*let* **{** if (**it**.*visibility* == View.*VISIBLE*) {  
 **it**.*visibility* = View.*GONE* } else {  
 **it**.*visibility* = View.*VISIBLE* }  
 **}** }  
}

**Wifi Info Demo Activity**

package com.example.finalsemester  
  
import android.annotation.SuppressLint  
import android.content.Context  
import android.net.wifi.WifiManager  
import android.os.Bundle  
import android.text.format.Formatter  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
  
class WifiInfoDemo : AppCompatActivity() {  
 @SuppressLint("MissingInflatedId")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_wifi\_info\_demo*)  
 val wifiInformationTv: TextView = findViewById(R.id.*textViewWifi*)  
 val wifiManager = *applicationContext*.getSystemService(Context.*WIFI\_SERVICE*) as WifiManager  
 val wInfo = wifiManager.*connectionInfo* val ipaddress = Formatter.formatIpAddress(wInfo.*ipAddress*)  
 val linkSpeed = wInfo.*linkSpeed* val networkID = wInfo.*networkId* val ssid = wInfo.*ssid* val hssid = wInfo.*hiddenSSID* val bssid = wInfo.*bssid* wifiInformationTv.*text* = "IP Address : \t$ipaddress\n"+  
 "Link Speed : \t$linkSpeed\n"+  
 "Network ID : \t$networkID\n"+  
 "SSID : \t$ssid\n"+  
 "Hidden SSID: \t$hssid\n"+  
 "BSSID : \t$bssid\n"  
 }  
}

**Text To Speech Activity**

package com.example.finalsemester  
import android.os.Bundle  
import android.speech.tts.TextToSpeech  
import android.util.Log  
import android.widget.ArrayAdapter  
import android.widget.Button  
import android.widget.EditText  
import android.widget.ImageView  
import android.widget.Spinner  
import androidx.appcompat.app.AppCompatActivity  
import java.util.Locale  
class TextToSpeechActivity : AppCompatActivity(), TextToSpeech.OnInitListener {  
 private lateinit var tts: TextToSpeech  
 private lateinit var editText: EditText  
 private lateinit var spinnerLanguages: Spinner  
 private lateinit var btnTranslate: Button  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_text\_to\_speech*)  
 tts = TextToSpeech(this, this)  
 editText = findViewById(R.id.*editText*)  
 spinnerLanguages = findViewById(R.id.*spinnerLanguages*)  
 btnTranslate = findViewById(R.id.*btnTranslate*)  
 val imageClear:ImageView = findViewById(R.id.*imageClear*)  
 imageClear.setOnClickListener **{** editText.setText("")  
 **}** val languages = *listOf*("Select Language","English (US)", "English (UK)", "French", "German", "Italian")  
 val adapter = ArrayAdapter(this, android.R.layout.*simple\_spinner\_item*, languages)  
 adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*)  
 spinnerLanguages.*adapter* = adapter  
 btnTranslate.setOnClickListener **{** val selectedLanguage = spinnerLanguages.*selectedItem*.toString()  
 val textToSpeak = editText.*text*.toString()  
 translateAndSpeak(textToSpeak, selectedLanguage)  
 **}** }  
 override fun onInit(status: Int) {  
 if (status == TextToSpeech.*SUCCESS*) {  
  
 } else {  
 Log.e("TTS", "TextToSpeech initialization failed.")  
 }  
 }  
 private fun translateAndSpeak(text: String, language: String) {  
 val locale = when (language) {  
 "English (US)" -> Locale.*US* "English (UK)" -> Locale.*UK* "French" -> Locale.*FRANCE* "German" -> Locale.*GERMANY* "Italian" -> Locale.*ITALY* "Arabic" -> Locale("ar", "SA")  
 else -> Locale.*US* }  
 val result = tts.setLanguage(locale)  
 if (result == TextToSpeech.*LANG\_MISSING\_DATA* || result == TextToSpeech.*LANG\_NOT\_SUPPORTED*) {  
 Log.e("TTS", "The language $language is not supported.")  
 } else {  
 tts.speak(text, TextToSpeech.*QUEUE\_FLUSH*, null, null)  
 }  
 }  
 override fun onDestroy() {  
 super.onDestroy()  
 tts.stop()  
 tts.shutdown()  
 }  
}

**Speech To Text Activity**

package com.example.finalsemester  
import android.content.Intent  
import android.os.Bundle  
import android.speech.RecognizerIntent  
import android.widget.ImageView  
import android.widget.TextView  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import java.util.Locale  
class SpeechToTextActivity : AppCompatActivity() {  
 private lateinit var outputTV: TextView  
 private lateinit var micIV: ImageView  
 private val REQUEST\_CODE\_SPEECH\_INPUT = 1  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_speech\_to\_text*)  
 outputTV = findViewById(R.id.*idTVOutput*)  
 micIV = findViewById(R.id.*idIVMic*)  
 micIV.setOnClickListener **{** val intent = Intent(RecognizerIntent.*ACTION\_RECOGNIZE\_SPEECH*)  
 intent.putExtra(  
 RecognizerIntent.*EXTRA\_LANGUAGE\_MODEL*,  
 RecognizerIntent.*LANGUAGE\_MODEL\_FREE\_FORM* )  
 intent.putExtra(  
 RecognizerIntent.*EXTRA\_LANGUAGE*,  
 Locale.getDefault()  
 )  
 intent.putExtra(RecognizerIntent.*EXTRA\_PROMPT*, "Speak to text")  
 try {  
 startActivityForResult(intent, REQUEST\_CODE\_SPEECH\_INPUT)  
 } catch (e: Exception) {  
 Toast.makeText(this@SpeechToTextActivity, "Error: ${e.message}", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}** }  
 override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {  
 super.onActivityResult(requestCode, resultCode, data)  
 if (requestCode == REQUEST\_CODE\_SPEECH\_INPUT) {  
 if (resultCode == *RESULT\_OK* && data != null) {  
 val results: ArrayList<String>? = data.getStringArrayListExtra(RecognizerIntent.*EXTRA\_RESULTS*)  
 if (results != null && results.*isNotEmpty*()) {  
 val inputText = results[0]  
 outputTV.*text* = inputText  
 } else {  
 outputTV.*text* = "No speech input detected"  
 }  
 }  
 }  
 }  
}

**Share the App Activity**

package com.example.finalsemester  
import android.content.Intent  
import android.os.Bundle  
import android.webkit.WebView  
import android.webkit.WebViewClient  
import android.widget.Button  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
class ShareTheApp : AppCompatActivity() {  
 private lateinit var webView: WebView  
 private lateinit var shareapp: Button  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_share\_the\_app*)  
 webView = findViewById(R.id.*webView*)  
 shareapp = findViewById(R.id.*shareapp*)  
 webView.*webViewClient* = WebViewClient()  
 webView.loadUrl("https://github.com/gopi76/Car-Pooling-App")  
 webView.*settings*.*javaScriptEnabled* = true  
 webView.*settings*.setSupportZoom(true)  
 shareapp.setOnClickListener **{** val intent = Intent().*apply* **{** *action* = Intent.*ACTION\_SEND* putExtra(Intent.*EXTRA\_TEXT*, "https://github.com/gopi76/Car-Pooling-App/blob/main/app-debug.apk")  
 *type* = "text/plain"  
 **}** if (intent.resolveActivity(*packageManager*) != null) {  
 startActivity(Intent.createChooser(intent, "Share via"))  
 } else {  
 Toast.makeText(  
 this,  
 "No app can handle this action",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 **}** }  
 override fun onBackPressed() {  
 if (webView.canGoBack()) {  
 webView.goBack()  
 } else {  
 super.onBackPressed()  
 }  
 }  
}

**App Admin Information Activity**

package com.example.finalsemester  
import android.content.Intent  
import android.net.Uri  
import android.os.Bundle  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
import com.google.android.material.floatingactionbutton.FloatingActionButton  
class AppAdminInformationActivity : AppCompatActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.activity\_app\_admin\_information)  
 val nameTextView = findViewById<TextView>(R.id.nameTextView)  
 val aboutTextView = findViewById<TextView>(R.id.aboutTextView)  
 val aboutTextView2 = findViewById<TextView>(R.id.aboutTextView2)  
 val portfolioLinkTextView = findViewById<TextView>(R.id.portfolioLinkTextView)  
 val callFab = findViewById<FloatingActionButton>(R.id.callFab)  
 val messageFab = findViewById<FloatingActionButton>(R.id.messageFab)  
 val emailFab = findViewById<FloatingActionButton>(R.id.emailFab)  
 nameTextView.text = "G. Gopi Krishna"  
 aboutTextView.text = "App Creator Info"  
 aboutTextView2.text = """  
 👋 Hi, I’m Gopi Krishna from Guntur, Andhra Pradesh.  
   
 💬 I'm currently pursuing a degree in Computer Science and Engineering at Lovely Professional University (Punjab), India.  
   
 👨‍💻 My technical skills include proficiency in C, C++, Java, Python, and I have a basic knowledge of Kotlin. I'm also familiar with front-end languages such as HTML, CSS, and JavaScript.  
   
 📝 I have published peer-reviewed term papers with a high impact factor in reputable journals like IJAENT and IJSCE.  
   
 🏆 Participated in various coding contests, achieving a notable Global rank of 713 on Codechef.  
   
 🔮 In terms of leadership, I have participated in GDSC-LPU as a part of the A.I/ML team member, where I assisted machine learning in simpler methods while mentoring the mentee and explored different concepts of ML.  
  
   
 If you encounter any issues with applications or need assistance, please feel free to contact me.  
""".trimIndent()  
 portfolioLinkTextView.setOnClickListener **{** val portfolioUrl = "https://gopi76.github.io/Portfolio.github.io/"  
 openUrl(portfolioUrl)  
 **}** callFab.setOnClickListener **{** val intent = Intent(Intent.ACTION\_DIAL)  
 intent.data = Uri.parse("tel:+917659046696")  
 startActivity(intent)  
 **}** messageFab.setOnClickListener **{** val intent = Intent(Intent.ACTION\_SENDTO)  
 intent.data = Uri.parse("smsto:+917659046696")  
 startActivity(intent)  
 **}** emailFab.setOnClickListener **{** val intent = Intent(Intent.ACTION\_SENDTO)  
 intent.data = Uri.parse("mailto:gopikrishnaguntamukkala3@gmail.com")  
 startActivity(intent)  
 **}** }  
 private fun openUrl(url: String) {  
 val intent = Intent(Intent.ACTION\_VIEW, Uri.parse(url))  
 startActivity(intent)  
 }  
}

**---------------The End-------------------**