Tutor Android App

Report

FROM

BTech CSE (P132L)

SUBMITTED TO

Dr. Subhita Menon Mam

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

SUBMITTED BY

G. Gopi Krishna

12115851

062 [Roll Number] – KO203

CSE227 : ADVANCED ANDROID APP DEVELOPMENT

GitHub Project Link:

https://github.com/gopi76/Tutor-App-Android-Studio-Kotlin-

Working Project Video Link:

https://github.com/gopi76/Tutor-App-Android-Studio-Kotlin-/blob/main/project_video.mp4

CSE224 (FUNDAMENTALS OF ANDROID):

- Toast
- Log
- Layouts : Linear, Relative and Constraint
- Alert Dialog
- Request App Permissions

CSE225 (DEVELOPING ANDROID APPS):

- Splash Screen
- Progress Bar
- Intents(both explicit and implicit intents)
- Loading
- Notification
- Navigation Drawer
- View Pager
- Date Picker Dialog
- Time Picker Dialog

- Rating Bar

CSE226:ANDROID APP DEPLOYMENT

- Recycler View
- Users Current Location
- Maps

CSE227 : ADVANCED ANDROID APP DEVELOPMENT

- Real Time Firebase
- Animation
- Proximity sensor

Others

- Alert Dialog Box
- Firebase Fire store (to store the tutor details)

Project Overview

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

1. **Sign In:** Users can log in with their credentials. This feature ensures that only registered users can access the app's functionalities, maintaining security and privacy.
2. **Sign Up:** New users can create a new account. The signup process collects essential information from users to create their profiles and grant them access to the app.
3. **Tutors List:** Users can view all available tutors. This feature provides a comprehensive list of tutors, including their profiles, expertise, and availability.

4. **Tutor Profile:** Users can view tutor profiles and book slots for classes. This feature allows users to explore detailed information about tutors, such as their qualifications, experience, and ratings from other users.

5. **Book Slot:** Users can also book slots for classes based on the tutor's availability.

6. **Nearby Tutors (Map View):** Utilizes OpenStreetMap to show nearby tutors in Map View. This feature uses geolocation to display tutors near the user's current location on a map, making it convenient for users to find local tutors.

7. **Profile Management:** Users can check and update their profiles. This feature enables users to edit their profile information, such as their name, contact details, and profile picture, ensuring that their information is up to date.

8. **Rating System:** Includes a rating bar for users to give feedback, with notifications for received ratings. This feature allows users to rate tutors based on their experience, providing valuable feedback to improve the quality of tutoring services. Users also receive notifications when they receive ratings from other users.

9. **Feedback Submission:** Users can submit feedback, stored in a real-time database. This feature allows users to share their feedback and suggestions about the app's functionality, user experience, and overall satisfaction, helping developers enhance the app based on user input.

10. **FAQ:** Users can access app details and necessary information. This feature provides a comprehensive FAQ section where users can find answers to common questions about the app's features.

11. **Sharing:** Users can share the app with others on their phones. This feature allows users to easily share the app with friends, family, or colleagues via messaging apps, social media platforms, or email, expanding the app's user base and visibility.

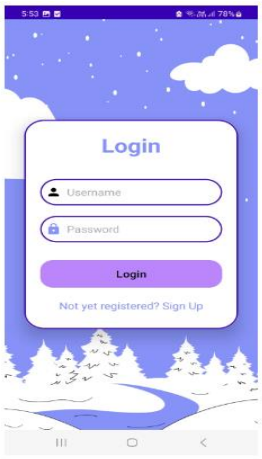
12. **Logout:** Users can log out of the app. This feature allows users to securely log out of their accounts, ensuring that their session is terminated and their data remains private.

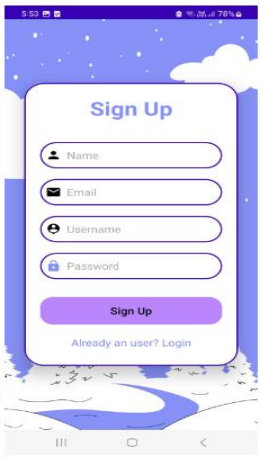
Project Images:


mytutor

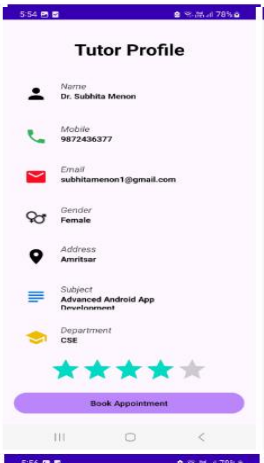

SEARCH TEACHER ON YOUR FINGERTIP
Our App has potenial to help Students!
Next

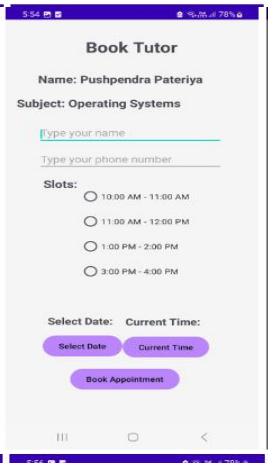

GET HAPPY WITH YOUR GRADES
Teacher's help you to achieve better Grades!
Start

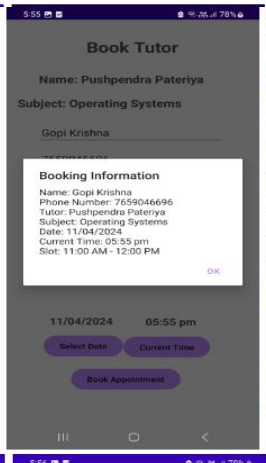

Login
Username
Password
Login
Not yet registered? Sign Up

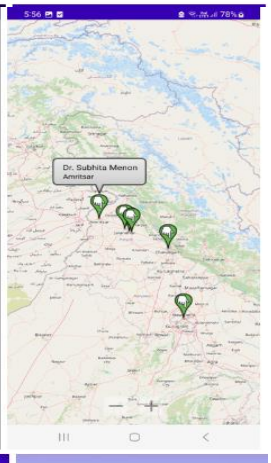

Sign Up
Name
Email
Username
Password
Sign Up
Already an user? Login

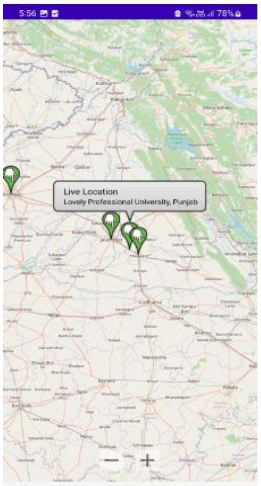

Available Tutors
Dr. Senthil Kumar J
Dr. Subhita Menon
Pushpendra Pateriya
Mr. Ravi Kant Sahu
Mr. Pranjal Kumar


Tutor Profile
Dr. Subhita Menon
9872436377
subhitamenon1@gmail.com
Female
Amritsar
CSE
Book Appointment


Book Tutor
Name: Pushpendra Pateriya
Subject: Operating Systems
Type your name
Type your phone number
Slots:
10:00 AM - 11:00 AM
11:00 AM - 12:00 PM
1:00 PM - 2:00 PM
3:00 PM - 4:00 PM
Select Date:
Current Time:
Book Appointment

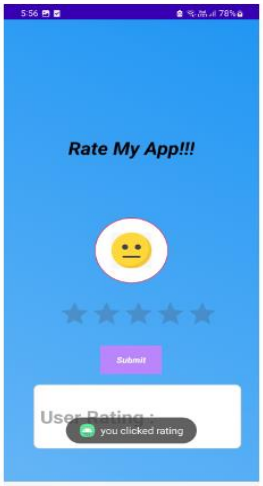

Book Tutor
Name: Pushpendra Pateriya
Subject: Operating Systems
Gopi Krishna
Booking Information
Name: Gopi Krishna
Phone Number: 7659046696
Tutor: Pushpendra Pateriya
Subject: Operating Systems
Date: 11/04/2024
Current Time: 05:55 pm
Slot: 11:00 AM - 12:00 PM
OK
11/04/2024
05:55 pm
Book Appointment

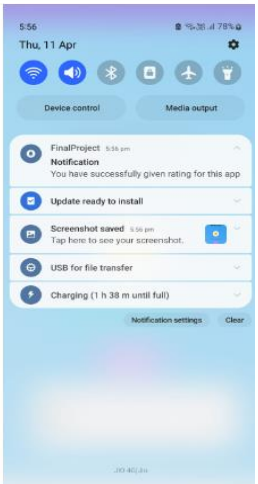

Dr. Subhita Menon
Amritsar

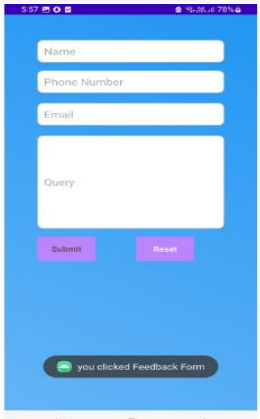

Live Location
Lovely Professional University, Punjab

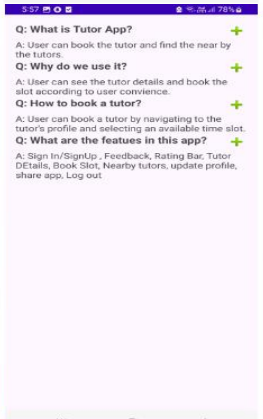

Edit Profile
Gopi Krishna G
gopikrishnaguntamukkala3
gopi76
Save


mytutor
Name
Gopi Krishna G
Email
Username
gopi76
Password
gopi76
Edit Profile


Rate My App!!!
Submit
User Rating :
you clicked rating


Thu, 11 Apr
Device control
Media output
FinalProject
Notification
You have successfully given rating for this app
Update ready to install
Screenshot saved
Tap here to see your screenshot.
USB for file transfer
Charging (1 h 38 m until full)
Notification settings
Clear


Name
Phone Number
Email
Query
Submit
Reset
you clicked Feedback Form


Q: What is Tutor App?
A: User can book the tutor and find the near by the tutors.
Q: Why do we use it?
A: User can see the tutor details and book the slot according to user convience.
Q: How to book a tutor?
A: User can book a tutor by navigating to the tutor's profile and selecting an available time slot.
Q: What are the featues in this app?
A: Sign In/SignUp , Feedback, Rating Bar, Tutor DEtails, Book Slot, Nearby tutors, update profile, share app, Log out


Tutor App
Logout
Are you sure you want to logout?
NO
YES

Login Activity Code :

```
package com.example.finalproject

import android.content.Intent
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
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 {
            Toast.makeText(this, "You clicked Login button", Toast.LENGTH_SHORT).show()
            if (validateInputs()) {
                checkUser()
            }
        }

        signupRedirectText.setOnClickListener {
            Toast.makeText(this, "You clicked SignUp button", Toast.LENGTH_SHORT).show()
            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)

        query.addListenerForSingleValueEvent(object : ValueEventListener {
            override fun onDataChange(snapshot: DataSnapshot) {
                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, MainActivity::class.java).apply {
                                putExtra("userDetails", userDetails)
                            }
                            startActivity(intent)
                            return
                        } else {
                            loginPassword.error = "Invalid Credentials"
                            return
                        }
                    }
                } else {
                    loginUsername.error = "User does not exist"
                }
            }
```

```
            override fun onCancelled(error: DatabaseError) {
                // Handle onCancelled event
            }
        })
    }
}
```

Main Activity Code:

```
package com.example.finalproject

import android.app.AlertDialog
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.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

    private lateinit var bottomNavigationView: BottomNavigationView

    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)

        bottomNavigationView = findViewById(R.id.bottomNavigationView)

        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()

        bottomNavigationView.setOnNavigationItemSelectedListener { item ->
            when (item.itemId) {
                R.id.tutors -> {
                    Toast.makeText(this, "you clicked nearby Tutors", Toast.LENGTH_SHORT).show()
                    val intent = Intent(this, NearbyTutorsActivity::class.java)
                    startActivity(intent)
                    true
                }
                R.id.settings -> {

                    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)
                    true
                }
                R.id.tutorsList -> {

                    Toast.makeText(this, "Tutors List clicked", Toast.LENGTH_SHORT).show()
                    val intent = Intent(this, TutorsActivity::class.java)
                    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.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.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, ratingstar::class.java)
                    startActivity(intent)
                }



                R.id.feedbackform11 -> {
                    Toast.makeText(this, "you clicked 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 the app", Toast.LENGTH_SHORT).show()
                    val intent = Intent().apply {
                        action = Intent.ACTION_SEND
                        putExtra(Intent.EXTRA_TEXT, "https://github.com/gopi76/Tutor-App-Android-Studio-Kotlin-")
                        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()
                    }
                }
            }

            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) {
                        showProximityAlert()
                    }
                }
            }
        }
    }



    override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {
        // Do nothing
    }

    private fun showProximityAlert() {
        val builder = AlertDialog.Builder(this)
        builder.setTitle("Alert!!")
        builder.setMessage("You are very near to your phone!")
        builder.setPositiveButton("OK", null)
        val dialog: AlertDialog = builder.create()
        dialog.show()
    }
}
```

Tutor Details Code:

```
package com.example.finalproject

import android.content.Intent
import android.os.Bundle
import android.widget.Button
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity

class TutorDetailsActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_tutor_details)
        if (intent.hasExtra("TUTOR_DETAILS")) {
            // Fetching tutor details passed from previous activity
            val tutor = intent.getParcelableExtra<Tutor>("TUTOR_DETAILS")
            if (tutor != null) {

                // Initializing views
                val tutorNameTextView: TextView = findViewById(R.id.tutorNameTextView)
                val phoneTextView: TextView = findViewById(R.id.teach_profile_mobileNumber)
                val emailTextView: TextView = findViewById(R.id.teach_profile_email_input)
                val subjectTextView: TextView = findViewById(R.id.subjectTextView)
                val genderTextView: TextView = findViewById(R.id.genderTextView)
                //val classTextView: TextView = findViewById(R.id.classTextView)
                val boardTextView: TextView = findViewById(R.id.boardTextView)
                val addressTextView: TextView = findViewById(R.id.addressTextView)
                val bookButton: Button = findViewById(R.id.bookButton)
                tutorNameTextView.text = tutor.name
                subjectTextView.text = "${tutor.subject}"
                genderTextView.text = "${tutor.gender}"
                phoneTextView.text = "${tutor.mobileNumber}"
                emailTextView.text = "${tutor.email}"
                boardTextView.text = "${tutor.board}"
                addressTextView.text = "${tutor.address}"
                bookButton.setOnClickListener {
                    // Create an Intent to start the BookingActivity
                    val intent = Intent(this@TutorDetailsActivity, BookingActivity::class.java)
                    intent.putExtra("TUTOR_DETAILS", tutor)
                    startActivity(intent)
                }
            } else {

                finish()
            }
        } else {

            finish()
        }
    }
}
```

Booking Activity Code:

```
package com.example.finalproject
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.RadioButton
import android.widget.RadioGroup
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
```

```
import java.text.SimpleDateFormat
import java.util.Calendar
import java.util.Date
import java.util.Locale

class BookingActivity : AppCompatActivity() {
    private var selectedDate: Date? = null
    private var selectedTime: Date? = null
    private lateinit var selectedDateTimeTextView: TextView
    private lateinit var selectedTimeTextView: TextView
    private lateinit var selectedRadioButton: RadioButton
    private val availableTimeSlots = mutableListOf<String>()
    private val calendar = Calendar.getInstance()
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_booking)
        val tutorNameTextView: TextView = findViewById(R.id.tutorNameTextView)
        val subjectTextView: TextView = findViewById(R.id.subjectTextView)
        val datePickerButton: Button = findViewById(R.id.datePickerButton1)
        val timePickerButton: Button = findViewById(R.id.timePickerButton1)
        val bookButton: Button = findViewById(R.id.bookButton)
        selectedDateTimeTextView = findViewById(R.id.selectedDateTextView1)
        selectedTimeTextView = findViewById(R.id.selectedTimeTextView1)
        val nameEditText: EditText = findViewById(R.id.name_edit)
        val phoneNumberEditText: EditText = findViewById(R.id.phone_number_edit)
        val radioGroup: RadioGroup = findViewById(R.id.radioGroupOptions)
        val tutor = intent.getParcelableExtra<Tutor>("TUTOR_DETAILS")


        tutorNameTextView.text = "Name: ${tutor?.name}"
        subjectTextView.text = "Subject: ${tutor?.subject}"
        availableTimeSlots.add("10:00 AM - 11:00 AM")
        availableTimeSlots.add("11:00 AM - 12:00 PM")
        availableTimeSlots.add("1:00 PM - 2:00 PM")
        availableTimeSlots.add("3:00 PM - 4:00 PM")

        radioGroup.setOnCheckedChangeListener { group, checkedId ->
            selectedRadioButton = findViewById<RadioButton>(checkedId)
        }
        datePickerButton.setOnClickListener {
            showDatePicker()
        }

        timePickerButton.setOnClickListener {
            showTimePicker()
        }

                bookButton.setOnClickListener {

            val name = nameEditText.text.toString()
            val phoneNumber = phoneNumberEditText.text.toString()


            if (selectedDate != null && selectedTime != null) {
                // Create an AlertDialog to confirm the booking
                val alertDialogBuilder = AlertDialog.Builder(this)
                alertDialogBuilder.setTitle("Confirm Booking")
                alertDialogBuilder.setMessage("Are you sure you want to book this appointment?")
                alertDialogBuilder.setPositiveButton("Yes") { _, _ ->

                    Toast.makeText(this, "Appointment booked successfully!", Toast.LENGTH_SHORT).show()
                    val bookingInfoDialogBuilder = AlertDialog.Builder(this)
                    bookingInfoDialogBuilder.setTitle("Booking Information")

                    bookingInfoDialogBuilder.setMessage("Name: $name\nPhone Number: $phoneNumber\nTutor: ${tutor?.name}\nSubject:
${tutor?.subject}\nDate: ${selectedDateTimeTextView.text}\nCurrent Time: ${selectedTimeTextView.text}\nSlot:
${selectedRadioButton.text}")
                    bookingInfoDialogBuilder.setPositiveButton("OK") { _, _ ->
                    }
                    val bookingInfoAlertDialog = bookingInfoDialogBuilder.create()
                    bookingInfoAlertDialog.show()
                }
                alertDialogBuilder.setNegativeButton("No") { _, _ ->
                                    Toast.makeText(this, "Booking canceled.", Toast.LENGTH_SHORT).show()
                }
                alertDialogBuilder.setNeutralButton("Cancel") { dialog, _ ->
                    dialog.dismiss()
                }

                alertDialogBuilder.show()
            } else {

                Toast.makeText(this, "Please select a date and time.", Toast.LENGTH_SHORT).show()
            }
        }
    }

    private fun showTimePicker() {
        val currentTime = Calendar.getInstance()
        val hour = currentTime.get(Calendar.HOUR_OF_DAY)
        val minute = currentTime.get(Calendar.MINUTE)

        val timePickerDialog = TimePickerDialog(
            this,
            TimePickerDialog.OnTimeSetListener { _, hourOfDay, minute ->

                val selectedTime = Calendar.getInstance()

                selectedTime.set(Calendar.HOUR_OF_DAY, hourOfDay)
                selectedTime.set(Calendar.MINUTE, minute)
                val timeFormat = SimpleDateFormat("hh:mm a", Locale.getDefault())
                val formattedTime = timeFormat.format(selectedTime.time)
                // Update the TextView to display the selected time with the "Selected Time: " prefix
```

```
                selectedTimeTextView.text = "$formattedTime"
                this.selectedTime = selectedTime.time
            },
            hour,
            minute,
            false
        )
        timePickerDialog.show()
    }

    private fun showDatePicker() {
        val datePickerDialog = DatePickerDialog(
            this, { _, year: Int, monthOfYear: Int, dayOfMonth: Int ->

                val selectedDate = Calendar.getInstance()
                selectedDate.set(year, monthOfYear, dayOfMonth)

                val dateFormat = SimpleDateFormat("dd/MM/yyyy", Locale.getDefault())
                val formattedDate = dateFormat.format(selectedDate.time)
                selectedDateTimeTextView.text = "$formattedDate"
                this.selectedDate = selectedDate.time
            },
            calendar.get(Calendar.YEAR),
            calendar.get(Calendar.MONTH),
            calendar.get(Calendar.DAY_OF_MONTH)
        )
        datePickerDialog.show()
    }
}
```

Near By Tutors Activity Code:

```
package com.example.finalproject

import android.Manifest
import android.content.pm.PackageManager
import android.location.Geocoder
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import com.google.android.gms.location.FusedLocationProviderClient
import com.google.android.gms.location.LocationServices
import com.google.firebase.FirebaseApp
import com.google.firebase.firestore.FirebaseFirestore
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 NearbyTutorsActivity : AppCompatActivity() {

    private lateinit var map: MapView
    private lateinit var fusedLocationClient: FusedLocationProviderClient
    private val firestore: FirebaseFirestore = FirebaseFirestore.getInstance()

    companion object {
        private const val REQUEST_LOCATION_PERMISSION = 1
    }

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

        Configuration.getInstance().load(this, getSharedPreferences("osmdroid", MODE_PRIVATE))

        setContentView(R.layout.activity_nearby_tutors)
        map = findViewById(R.id.map)
        map.setTileSource(TileSourceFactory.MAPNIK)
        map.setMultiTouchControls(true)

        // Set initial center to the Deoniwas Boys Hostel, Chaheru
        val initialCenter = GeoPoint(31.5018, 75.5728) // Coordinates for the Deoniwas Boys Hostel, Chaheru
        map.controller.setCenter(initialCenter)
        map.controller.setZoom(11.0) // Adjust the zoom level as needed

        // Check and request location permission if not granted
        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
            )
        }

        retrieveTutorInformation()
        initializeMap()
        FirebaseApp.initializeApp(this)

    }

    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 && ActivityCompat.checkSelfPermission(
            this,
            Manifest.permission.ACCESS_COARSE_LOCATION
        ) != PackageManager.PERMISSION_GRANTED
    ) {

        return
    }

    fusedLocationClient.lastLocation
        .addOnSuccessListener { location ->
            if (location != null) {
                val geocoder = Geocoder(this)
                try {
                    val addressList = geocoder.getFromLocation(location.latitude, location.longitude, 1)
                    if (addressList != null) {
                        if (addressList.isNotEmpty()) {
                            val address = addressList[0]
                            val currentLocation = GeoPoint(location.latitude, location.longitude)
                            addMarkerAtLocation(currentLocation, "My Location", address.getAddressLine(0))
                        } else {
                            addDefaultMarker()
                        }
                    }
                } catch (e: IOException) {
                    e.printStackTrace()
                    addDefaultMarker()
                }
            } else {

            }
        }
}



private fun retrieveTutorInformation() {

    val tutorList = mutableListOf<Pair<String, String>>()
    val defaultLocation = Pair("Live Location", "Lovely Professional University, Punjab")
    tutorList.add(defaultLocation)

    firestore.collection("tutors")
        .get()
        .addOnSuccessListener { documents ->
            for (document in documents) {
                val tutorName = document.getString("name") ?: "Tutor"
                val tutorAddress = document.getString("address") ?: ""
                if (tutorAddress.isNotEmpty()) {
                    tutorList.add(tutorName to tutorAddress)
                }
            }

            addMarkersForTutors(tutorList)
        }
        .addOnFailureListener { exception ->
                        }
}

private fun addDefaultMarker() {
    val defaultLocation = GeoPoint(31.2565, 75.6509) // Coordinates for Lovely Professional University
    val marker = Marker(map)
    marker.position = defaultLocation
    marker.title = "Default Location"
    marker.snippet = "Lovely Professional University"

    map.overlays.add(marker)
    marker.icon = ContextCompat.getDrawable(this, R.drawable.marker_red)
    // Show snippet immediately
    marker.showInfoWindow()
}
private fun addMarkersForTutors(tutorList: List<Pair<String, String>>) {
    for ((tutorName, tutorAddress) in tutorList) {
        val tutorLocation = getGeoPointFromAddress(tutorAddress)
        tutorLocation?.let {
            addMarkerAtLocation(it, tutorName, tutorAddress)
        }
    }
}
private fun addMarkerAtLocation(location: GeoPoint, title: String, snippet: String) {
    val marker = Marker(map)
    marker.position = location
    marker.title = title
    marker.snippet = snippet
    map.overlays.add(marker)

    marker.showInfoWindow()
}
private fun getGeoPointFromAddress(address: String?): GeoPoint? {
    val geocoder = Geocoder(this)
    return try {
        val locationList = address?.let { geocoder.getFromLocationName(it, 1) }
        if (locationList != null && locationList.isNotEmpty()) {
            val latitude = locationList[0].latitude
            val longitude = locationList[0].longitude
            GeoPoint(latitude, longitude)
```

```
            } else {
                null
            }
        } catch (e: IOException) {
            e.printStackTrace()
            null
        }
    }
    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()
    }
}
```

Edit Profile Activity Code:

```
package com.example.finalproject

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 Code:

```
package com.example.finalproject

import android.Manifest
import android.annotation.SuppressLint
import android.app.NotificationChannel
import android.app.NotificationManager
import android.content.Context
import android.content.pm.PackageManager
import android.os.Build
import android.os.Bundle
import android.util.Log
import android.widget.Button
import android.widget.ImageView
import android.widget.RatingBar
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.NotificationCompat
import androidx.core.app.NotificationManagerCompat

class ratingstar : AppCompatActivity() {
    private val CHANNEL_ID = "Channel_id_example_1"
    private val notificationid = 101

    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_ratingstar)

        createNotificationChannel()

        val rt: RatingBar = findViewById(R.id.ratingBar)
        val txtview: TextView = findViewById(R.id.textxviewrating)
        val submit: Button = findViewById(R.id.button)
        submit.setBackgroundColor(getResources().getColor(R.color.logo_secondary));

        var emoji: ImageView = findViewById(R.id.emoji)

        rt.onRatingBarChangeListener =
            RatingBar.OnRatingBarChangeListener { _, _, _ -> // Do something when the rating changes
                val rate = rt.rating
                if (rate <= 1) {
                    emoji.setImageResource(R.drawable.sad)
                } else if (rate <= 2) {
                    emoji.setImageResource(R.drawable.worried)
                } else if (rate <= 3) {
                    emoji.setImageResource(R.drawable.nuetral)
                } else if (rate <= 4) {
                    emoji.setImageResource(R.drawable.happy)
                } else {
                    emoji.setImageResource(R.drawable.sattisfied)
                }

                submit.setOnClickListener {
                    txtview.text = rt.rating.toString()
                    Toast.makeText(this, "Selected Rating: ${rt.rating}", Toast.LENGTH_SHORT).show()

                    sendNotification()
                }
            }
    }

    private fun createNotificationChannel() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            val name = "Notification"
            val descriptionText = "Notification Description"
```

```
            val importance = NotificationManager.IMPORTANCE_DEFAULT
            val channel = NotificationChannel(CHANNEL_ID,name, importance).apply {
                description = descriptionText
            }
            val notificationManager:NotificationManager = getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
            notificationManager.createNotificationChannel(channel)
        }
    }

    private fun sendNotification() {
        val builder = NotificationCompat.Builder(this, CHANNEL_ID)
            .setSmallIcon(R.drawable.baseline_circle_notifications_24)
            .setContentTitle("Notification")
            .setContentText("You have successfully given rating for this app")
            .setPriority(NotificationCompat.PRIORITY_DEFAULT)
        with(NotificationManagerCompat.from(this)) {
            notify(notificationid, builder.build())
        }

    }
}
```

Feedback form Code:

```
package com.example.finalproject

import android.annotation.SuppressLint
import android.app.AlertDialog
import android.app.Notification
import android.app.NotificationChannel
import android.app.NotificationManager
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 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 loadingDialog: DialogHelper


    // Reference to the Firebase Realtime Database
    private lateinit var databaseReference: DatabaseReference

    lateinit var notificationManager: NotificationManager
    lateinit var notificationChannel: NotificationChannel
    lateinit var builder: Notification.Builder
    private val channelId = "i.apps.notifications"
    private val description = "Test notification"

    @SuppressLint("MissingInflatedId")
    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)
        btnSubmit = findViewById(R.id.btnSubmit)
        editTextEmail = findViewById(R.id.editTextEmail)

        // Initialize the Firebase Database reference
        databaseReference = FirebaseDatabase.getInstance().reference.child("contactInfo")

        // Set onClickListener for btnSubmit

        val loadingdialog = DialogHelper(this)
        btnSubmit.setOnClickListener {
            onSubmitClick(it)
        }

        val resetBtn: Button = findViewById(R.id.btnReset)
        resetBtn.setOnClickListener {
            editTextName.text.clear()
            editTextEmail.text.clear()
            editTextPhone.text.clear()
            editTextQuery.text.clear()
        }

    }

    fun onSubmitClick(view: android.view.View) {
        val name = editTextName.text.toString()
        val phone = editTextPhone.text.toString()
        val query = editTextQuery.text.toString()
        val email = editTextEmail.text.toString()
        val builder = AlertDialog.Builder(this)

        builder.setTitle("Submit Query")
        builder.setMessage("Are you sure you want to submit your query?")
        builder.setPositiveButton("Yes") { dialog, _ ->
            val submissionKey = databaseReference.push().key
```

```
            val contactInfo = ContactInfo(name, phone, email, query)

            if (submissionKey != null) {
                databaseReference.child(submissionKey).setValue(contactInfo)
            }
            Toast.makeText(this, "Successfully submitted your query", Toast.LENGTH_SHORT).show()
            dialog.dismiss()
        }

        builder.setNegativeButton("No") { dialog, _ ->

            dialog.dismiss()
        }

        builder.setNeutralButton("Cancel") { dialog, _ ->

            dialog.dismiss()
        }
        val dialog = builder.create()
        dialog.show()
    }

}
```

FAQ activity Code:

```
package com.example.finalproject

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() {
        // Question 1
        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 Tutor App?"
        answer1.text = "A: User can book the tutor and find the near by the tutors."

        // Question 2
        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 do we use it?"
        answer2.text = "A: User can see the tutor details and book the slot according to user convience. "

        // Question 3
        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 tutor?"
        answer3.text = "A: User can book a tutor by navigating to the tutor's profile and selecting an available time slot."

        // Question 4
        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: Sign In/SignUp , Feedback, Rating Bar, Tutor DEtails, Book Slot, Nearby tutors, update profile, share app,
Log out"

        // Set click listeners for expand/collapse
        question1Layout.tag = answer1
        question2Layout.tag = answer2
        question3Layout.tag = answer3
        question4Layout.tag = answer4
    }

    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
            }
        }
    }
}
```

Bottom Navigation Menu Code:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/tutorsList"
```

```
        android:title="Tutors List"
        android:icon="@drawable/nav_teacher"/>
    <item
        android:id="@+id/tutors"
        android:title="nearby Tutors"
        android:icon="@drawable/baseline location on 24"/>
    <item
        android:id="@+id/settings"
        android:title="User Profile"
        android:icon="@drawable/baseline_settings_24"/>
</menu>
```

Navigation Drawer :

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:tools="http://schemas.android.com/tools"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <group
        android:checkableBehavior="single">
        <item
            android:id="@+id/row home"
            android:title="Home"
            android:icon="@drawable/baseline home 24">
        </item>

        <item android:title="Edit Profile"
            android:id="@+id/settings profile"
            android:icon="@drawable/baseline_edit_24"/>

        <item android:title="Rating"
            android:id="@+id/rating11"
            android:icon="@drawable/baseline star rate 24"/>

        <item android:title="Feedback Form"
            android:id="@+id/feedbackform11"
            android:icon="@drawable/baseline feedback 24"/>

        <item android:title="FAQ"
            android:id="@+id/faq"
            android:icon="@drawable/baseline_question_answer_24"/>

        <item
            android:title="Share"
            android:id="@+id/row share"
            android:icon="@drawable/baseline share 24">
        </item>

        <item
            android:title="Log Out"
            android:id="@+id/logOut"
            android:icon="@drawable/baseline_logout_24">
        </item>

    </group>

</menu>
```

----------The End----------