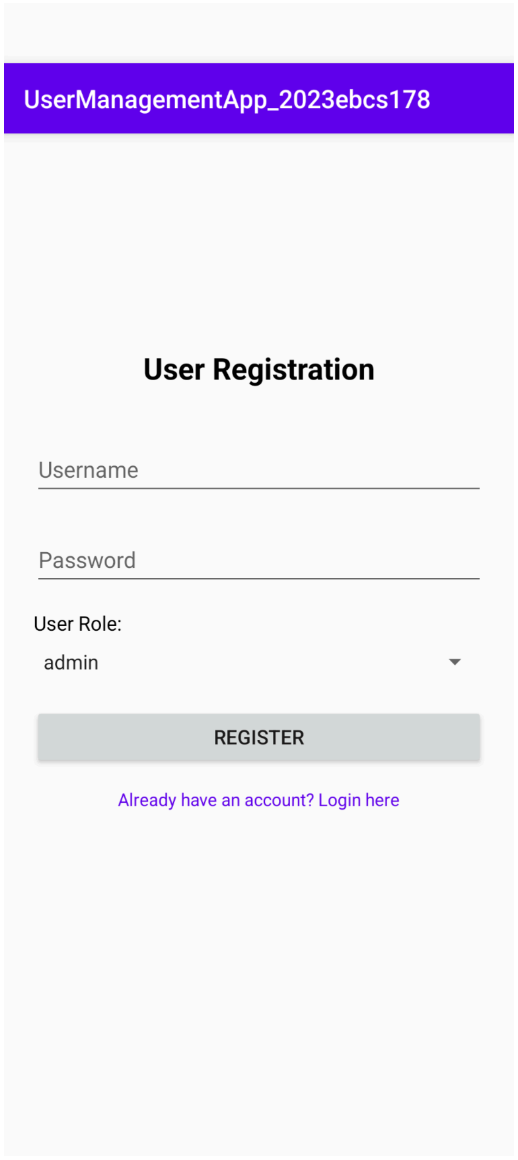
Prachi Tandel
Student ID: 2023ebcs178

A) User Login with Shared Preferences Implement a login screen using shared preferences
● Create a RegisterActivity with the following:
○ an EditText for the username
○ an EditText for password
○ a dropdown with values – 'admin' and 'normal' to identify the user
○ a button with name "Register"



UserManagementApp_2023ebcs178

User Registration

Username

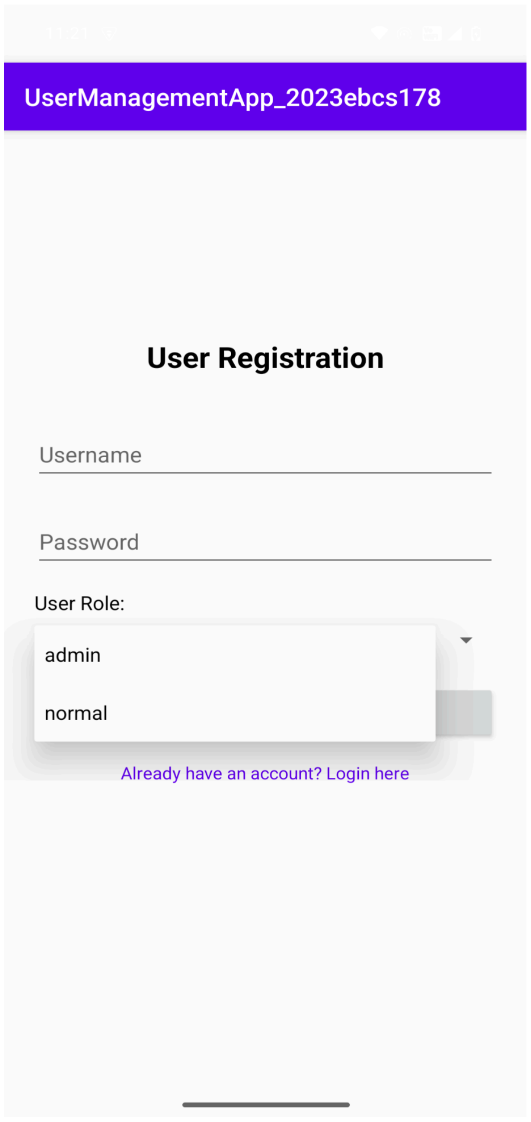
Password

User Role:

admin

REGISTER

Already have an account? Login here



UserManagementApp_2023ebcs178

User Registration

Username

Password

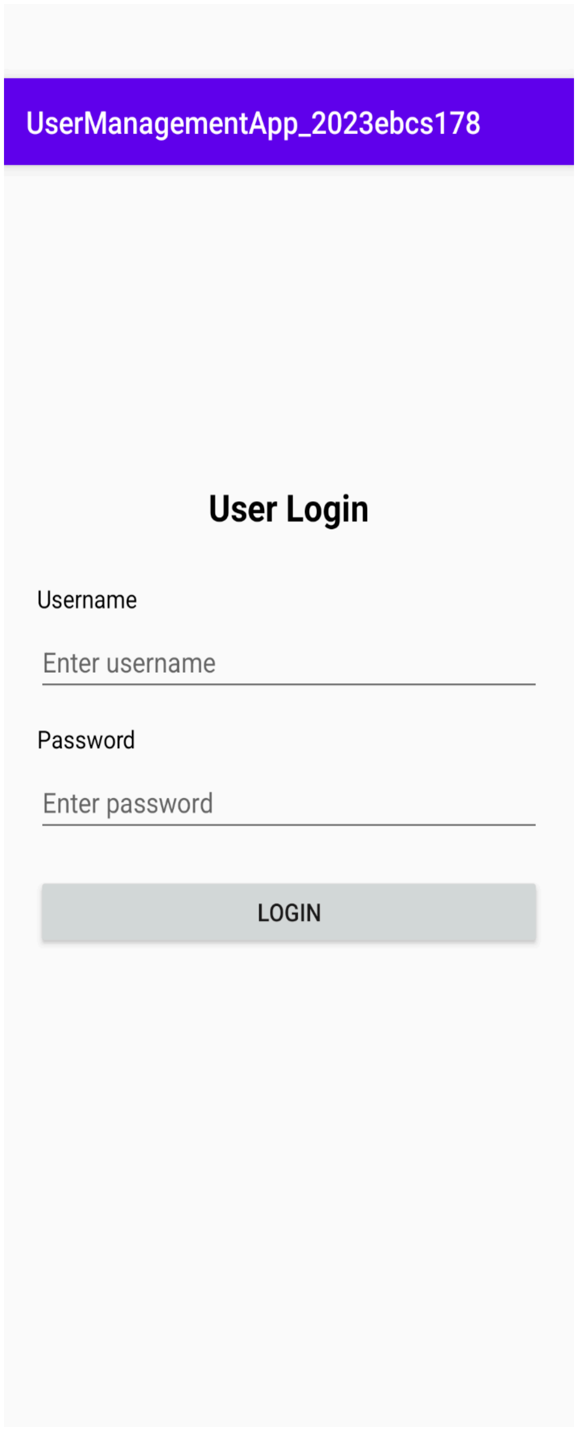
User Role:

admin

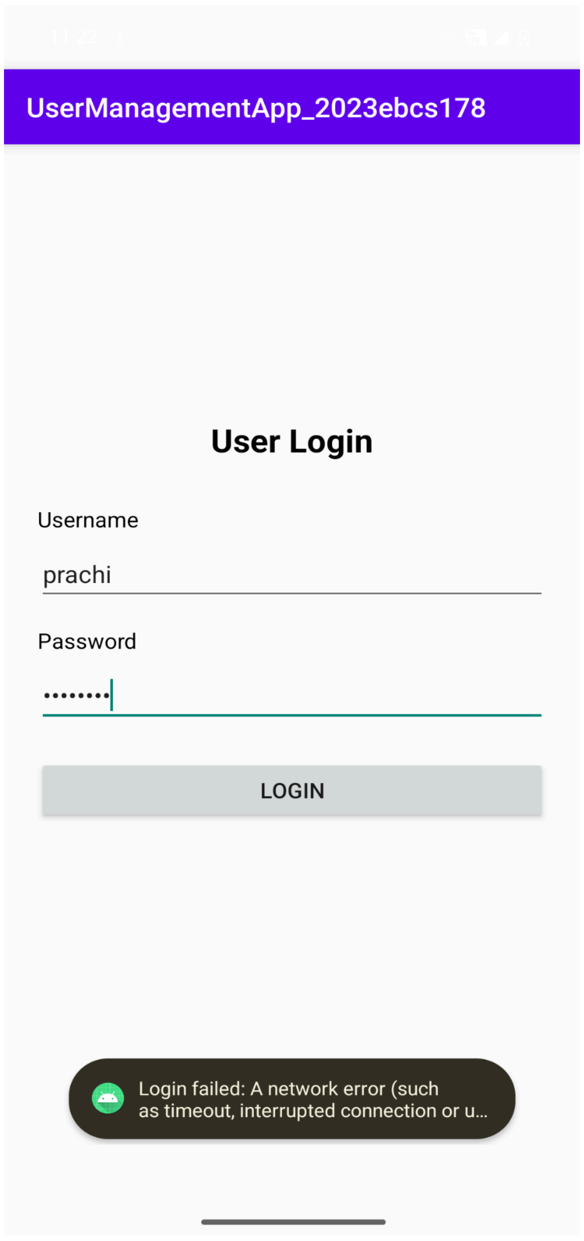
normal

Already have an account? Login here

- a label with link to start login activity with the following fields :
  - Username (TextView and EditText)
  - Password (TextView and EditText)
  - Login (Button)



● On Click of Register Button
○ Check for Network Connectivity in the Device, if it is connected, proceed further else ask the user to change the settings using a popup or toast


UserManagementApp_2023ebcs178

User Login

Username

prachi

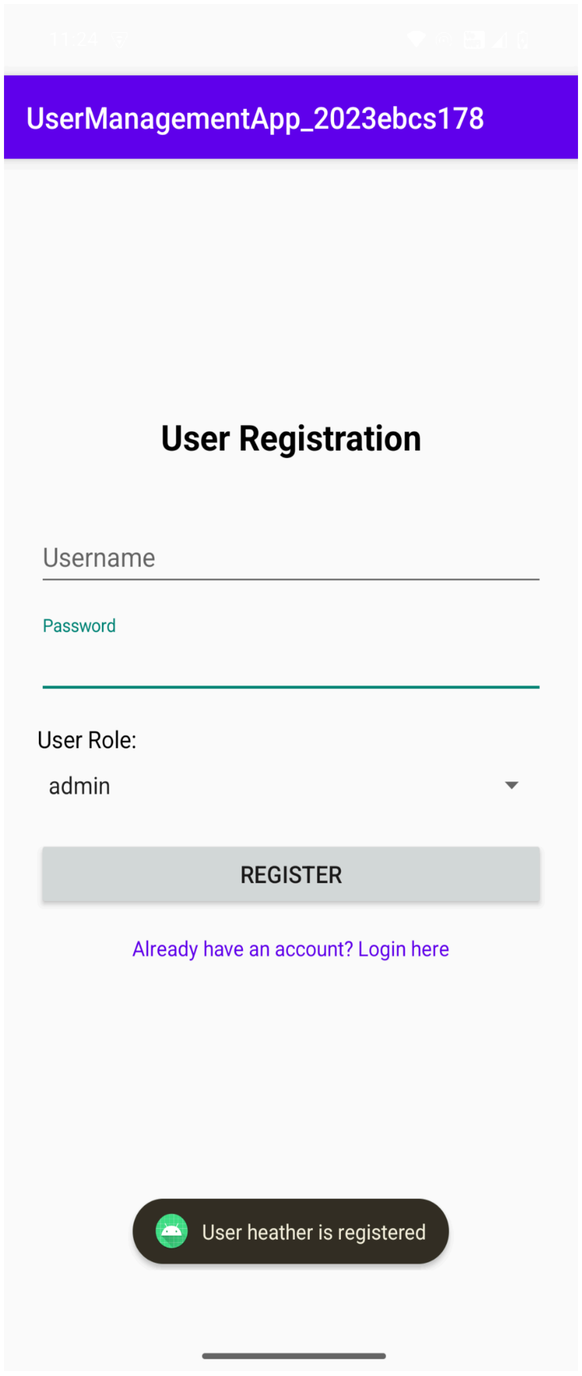
Password

••••••••

LOGIN

Login failed: A network error (such as timeout, interrupted connection or u...

○ The details are stored in the Firebase database using “createUserWithEmailAndPassword”, display a toast message “User [Username] is registered”.


UserManagementApp_2023ebcs178

User Registration

Username

Password

User Role:

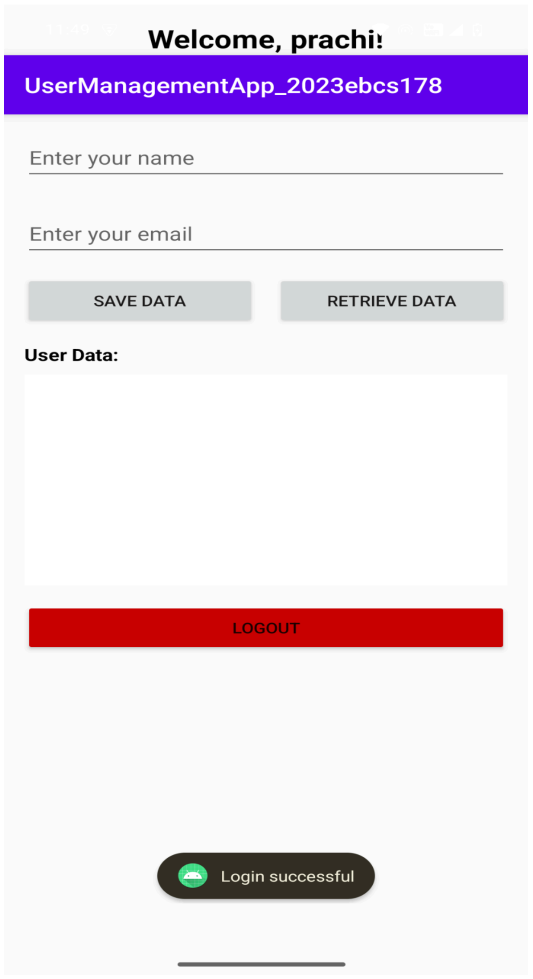
admin

REGISTER

Already have an account? Login here

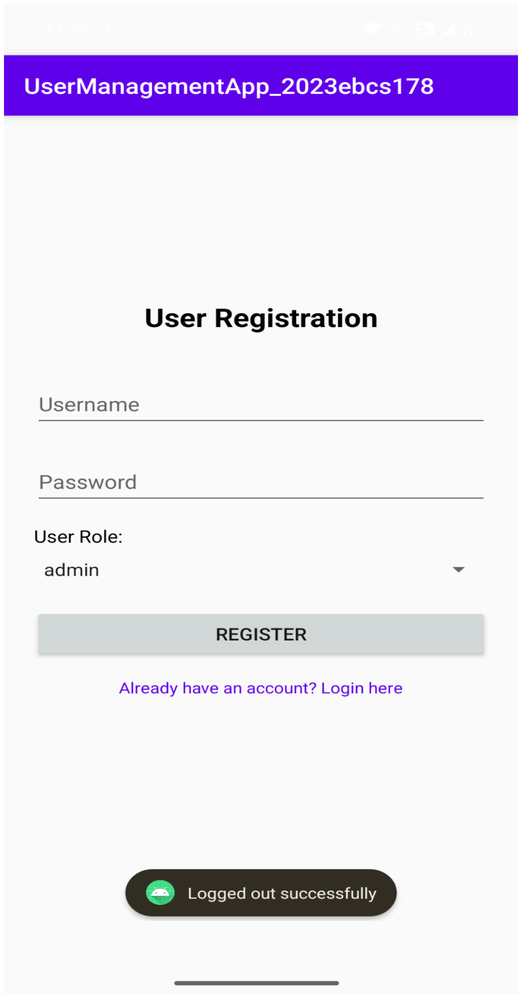
User heather is registered

- On click of Login link , LoginActivity should be displayed
- After entering the details(Username and Password), click on the “Login” button
- The details need to be Firebase authenticated using “signInWithEmailAndPassword”
- Save the entered username in Shared Preferences when the "Login" button is clicked.
- Navigate to a second screen (WelcomeActivity) that displays the username (e.g., "Welcome, [Username]!").
- WelcomeActivity should have “logout” button
- On click of Logout button – SharedPref should be deleted, user should be logged out and RegisterActivity screen should be displayed


Welcome, prachi!

UserManagementApp_2023ebcs178

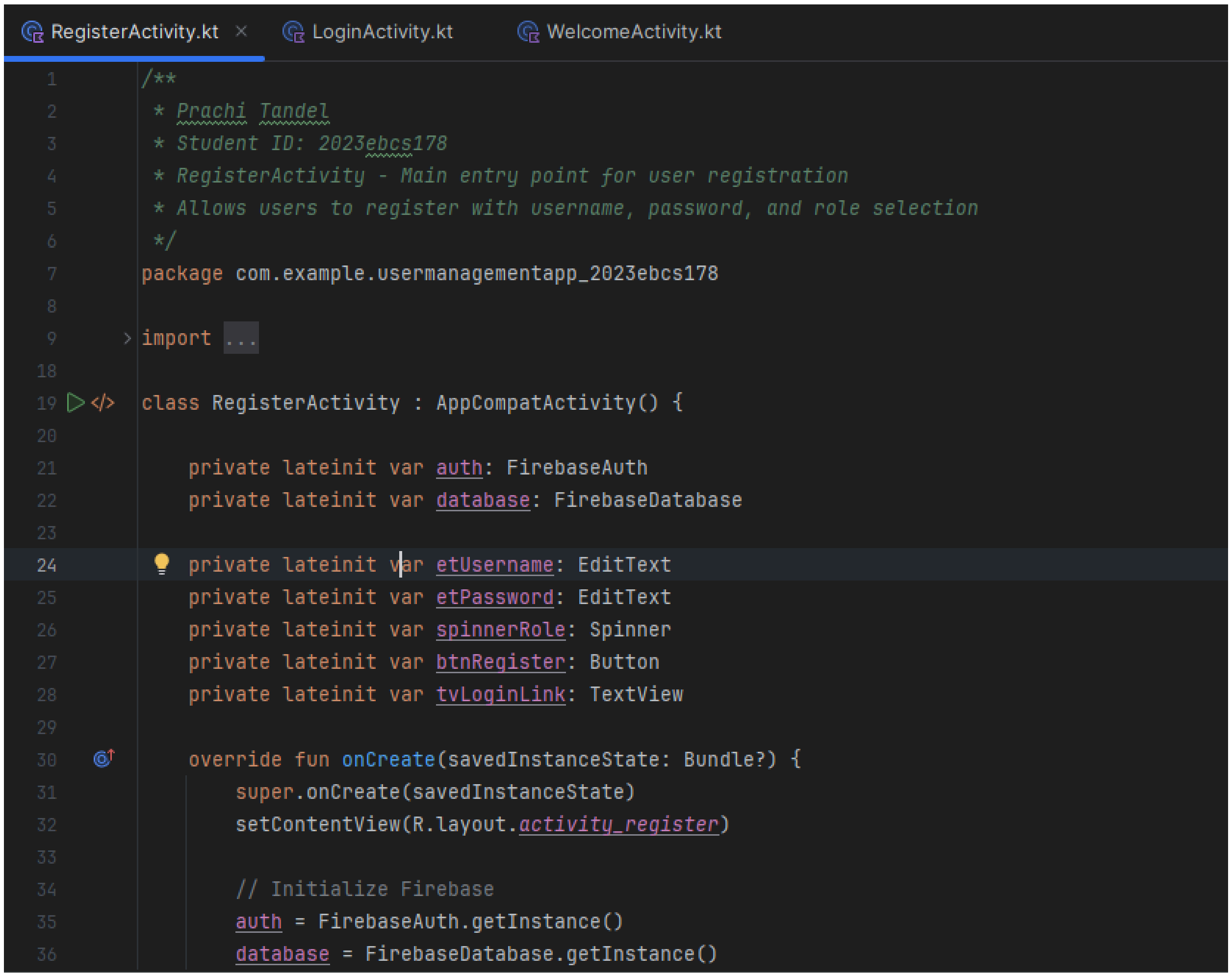
Login successful


UserManagementApp_2023ebcs178

User Registration

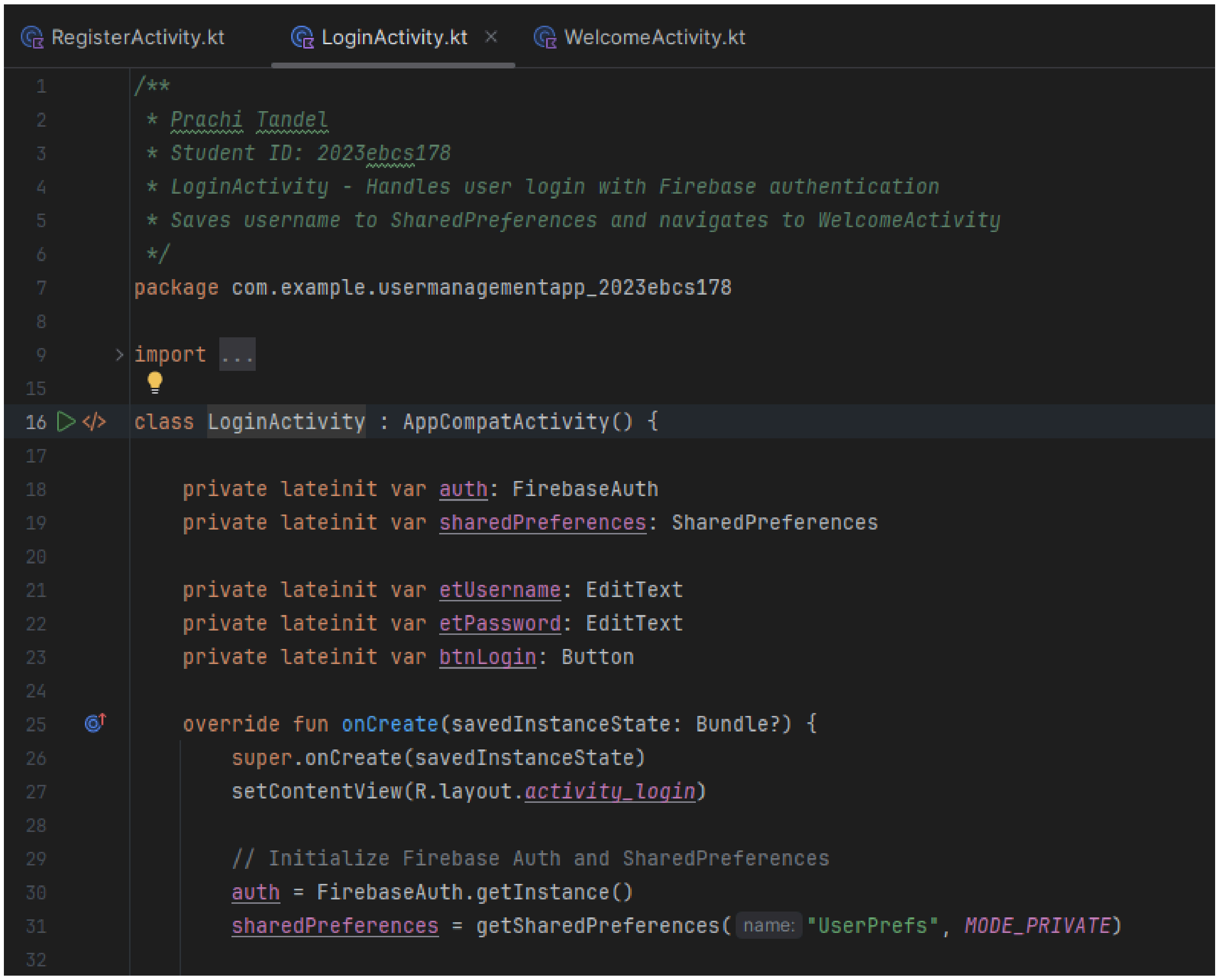
Logged out successfully

Deliverable:
● Source code for the register, login and welcome screens.
RegisterActivity.kt



```kotlin
/**
 * Prachi Tandel
 * Student ID: 2023ebcs178
 * RegisterActivity - Main entry point for user registration
 * Allows users to register with username, password, and role selection
 */
package com.example.usermanagementapp_2023ebcs178

import ...

class RegisterActivity : AppCompatActivity() {

    private lateinit var auth: FirebaseAuth
    private lateinit var database: FirebaseDatabase

    private lateinit var etUsername: EditText
    private lateinit var etPassword: EditText
    private lateinit var spinnerRole: Spinner
    private lateinit var btnRegister: Button
    private lateinit var tvLoginLink: TextView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_register)

        // Initialize Firebase
        auth = FirebaseAuth.getInstance()
        database = FirebaseDatabase.getInstance()
```

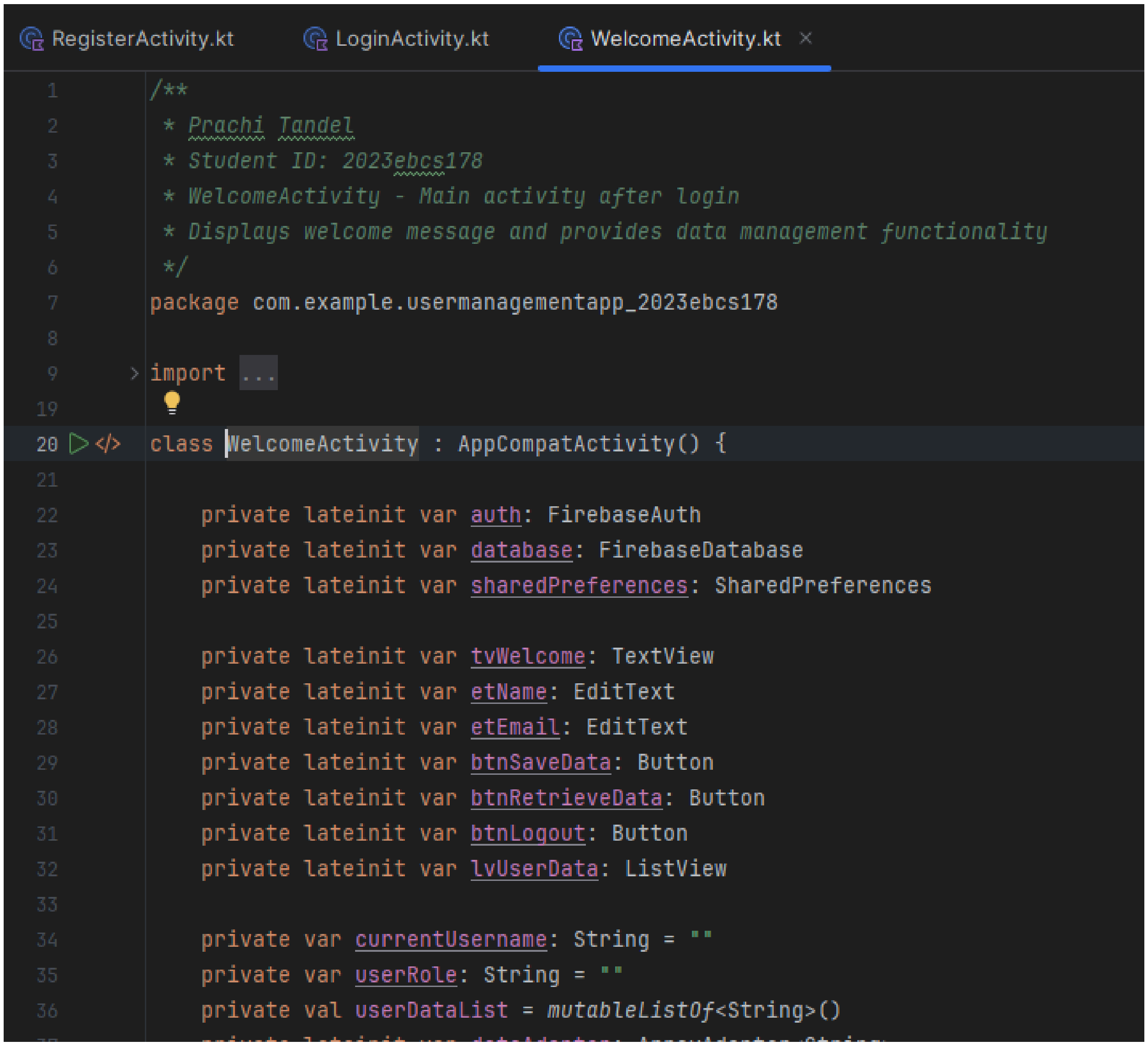
LoginActivity.kt



RegisterActivity.kt LoginActivity.kt WelcomeActivity.kt

```kotlin
/**
 * Prachi Tandel
 * Student ID: 2023ebcs178
 * LoginActivity - Handles user login with Firebase authentication
 * Saves username to SharedPreferences and navigates to WelcomeActivity
 */
package com.example.usermanagementapp_2023ebcs178

import ...

class LoginActivity : AppCompatActivity() {

    private lateinit var auth: FirebaseAuth
    private lateinit var sharedPreferences: SharedPreferences

    private lateinit var etUsername: EditText
    private lateinit var etPassword: EditText
    private lateinit var btnLogin: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)

        // Initialize Firebase Auth and SharedPreferences
        auth = FirebaseAuth.getInstance()
        sharedPreferences = getSharedPreferences( name: "UserPrefs", MODE_PRIVATE)

```

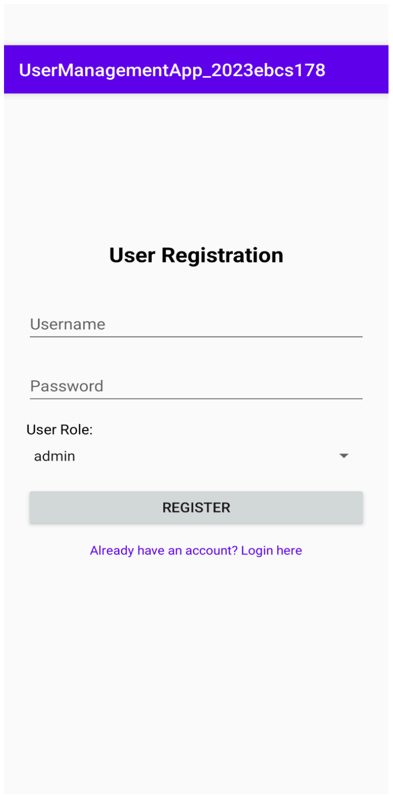
WelcomeActivity.kt



RegisterActivity.kt LoginActivity.kt WelcomeActivity.kt

```
/**
 * Prachi Tandel
 * Student ID: 2023ebcs178
 * WelcomeActivity - Main activity after login
 * Displays welcome message and provides data management functionality
 */
package com.example.usermanagementapp_2023ebcs178

import ...

class WelcomeActivity : AppCompatActivity() {

    private lateinit var auth: FirebaseAuth
    private lateinit var database: FirebaseDatabase
    private lateinit var sharedPreferences: SharedPreferences

    private lateinit var tvWelcome: TextView
    private lateinit var etName: EditText
    private lateinit var etEmail: EditText
    private lateinit var btnSaveData: Button
    private lateinit var btnRetrieveData: Button
    private lateinit var btnLogout: Button
    private lateinit var lvUserData: ListView

    private var currentUsername: String = ""
    private var userRole: String = ""
    private val userDataList = mutableListOf<String>()
```

- Screenshot of the register, login screen and the welcome screen.



UserManagementApp_2023ebcs178

User Registration

Username

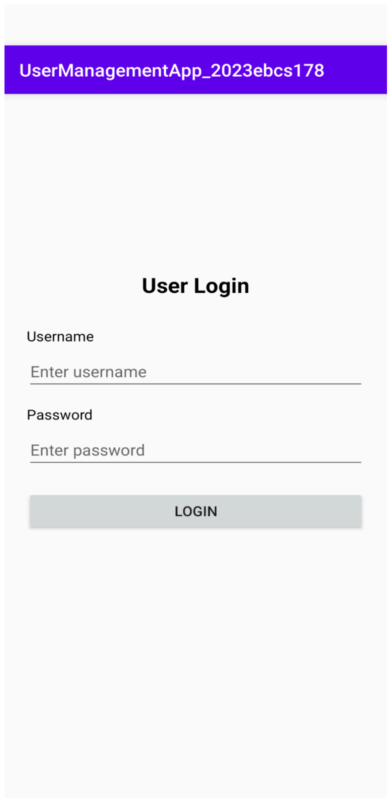
Password

User Role:

admin

REGISTER

Already have an account? Login here



UserManagementApp_2023ebcs178

User Login

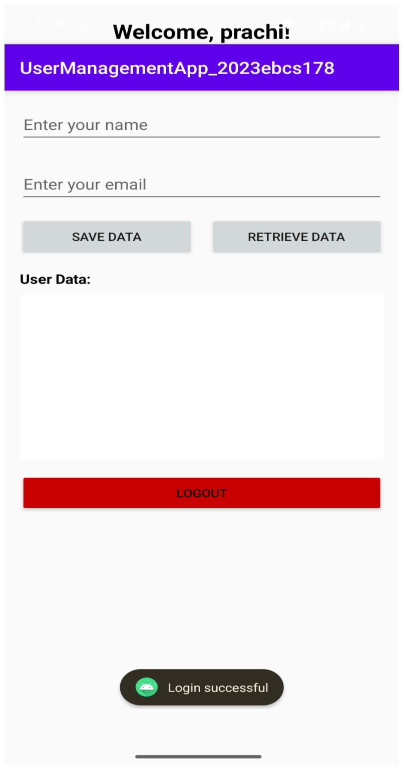
Username

Enter username

Password

Enter password

LOGIN



Welcome, prachi!

UserManagementApp_2023ebcs178

Enter your name

Enter your email

SAVE DATA

RETRIEVE DATA

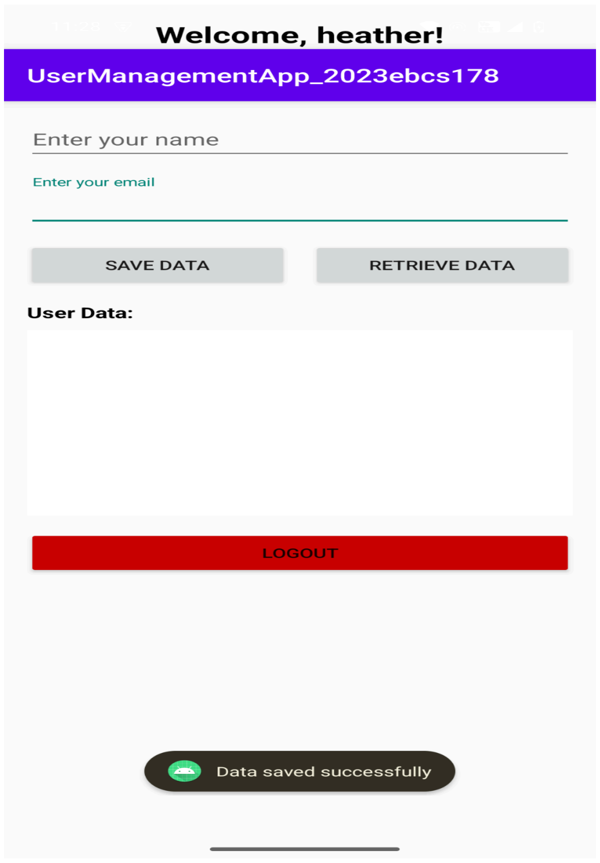
User Data:

LOGOUT

Login successful

B) Storing and Displaying Data Using Firebase
Allow the logged-in user to store and retrieve user details using Firebase Realtime Database.

- Check for Network Connectivity in the Device, if it is connected, proceed further else ask the user to change the settings using a popup or toast.
- If network connectivity is there, Save the entered details to Firebase under a node for the logged-in user.



Welcome, heather!

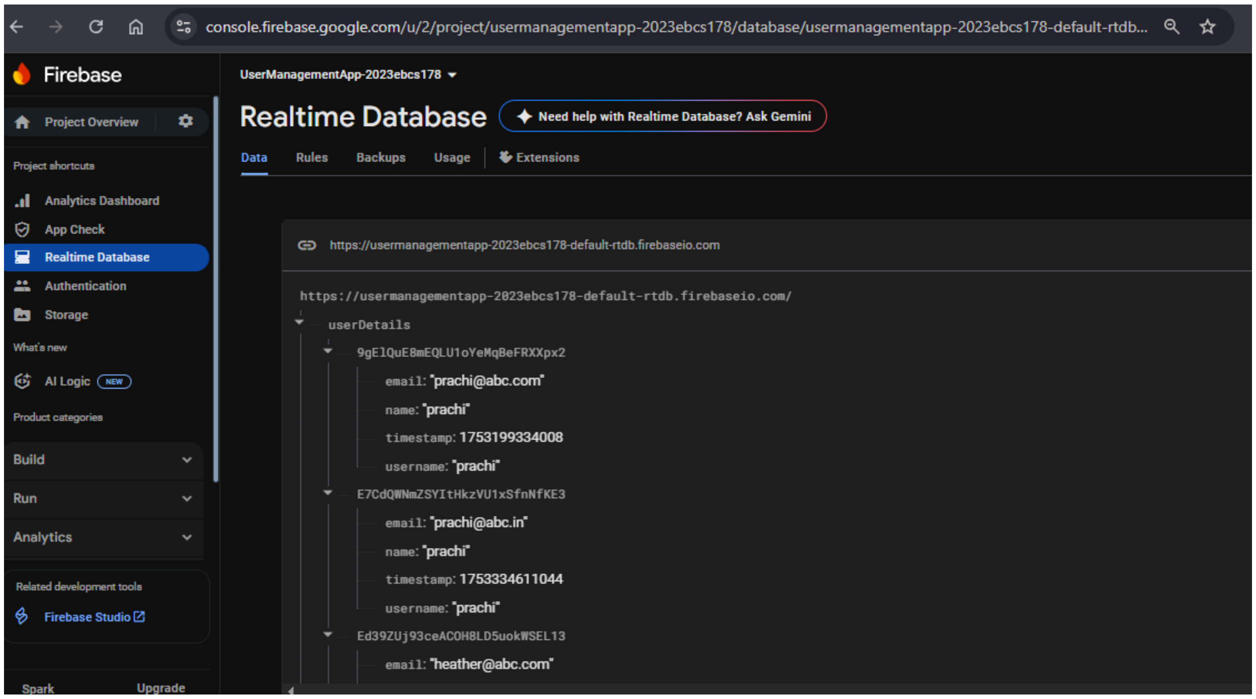
UserManagementApp_2023ebcs178

Enter your name

Enter your email

User Data:

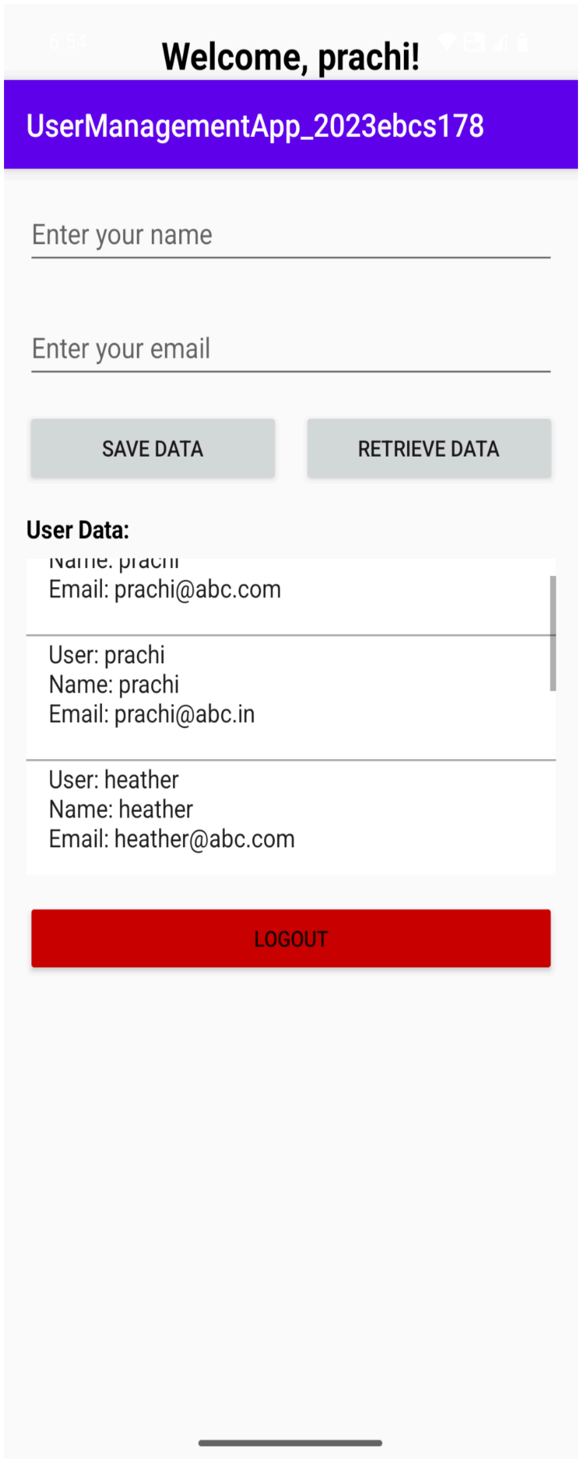
Data saved successfully



console.firebase.google.com/u/2/project/usermanagementapp-2023ebcs178/database/usermanagementapp-2023ebcs178-default-rtdb...
Firebase
Project Overview
Project shortcuts
Analytics Dashboard
App Check
Realtime Database
Authentication
Storage
What's new
AI Logic NEW
Product categories
Build
Run
Analytics
Related development tools
Firebase Studio
Spark
Upgrade
UserManagementApp-2023ebcs178
Realtime Database
Need help with Realtime Database? Ask Gemini
Data
Rules
Backups
Usage
Extensions
https://usermanagementapp-2023ebcs178-default-rtdb.firebaseio.com

```
https://usermanagementapp-2023ebcs178-default-rtdb.firebaseio.com/
  userDetails
    9gE1QuE8mEQLU1oYeMqBeFRXXpx2
      email: "prachi@abc.com"
      name: "prachi"
      timestamp: 1753199334008
      username: "prachi"
    E7CdQWNmZSYItHkzVU1xSfnNfKE3
      email: "prachi@abc.in"
      name: "prachi"
      timestamp: 1753334611044
      username: "prachi"
    Ed39ZUj93ceACOH8LD5uokWSEL13
      email: "heather@abc.com"
```

● If logged in as Admin - Retrieve all the user details for the logged-in user and display them in a ListView using scrollable list. Use Service to get all the details from Firebase.


Welcome, prachi!

UserManagementApp_2023ebcs178

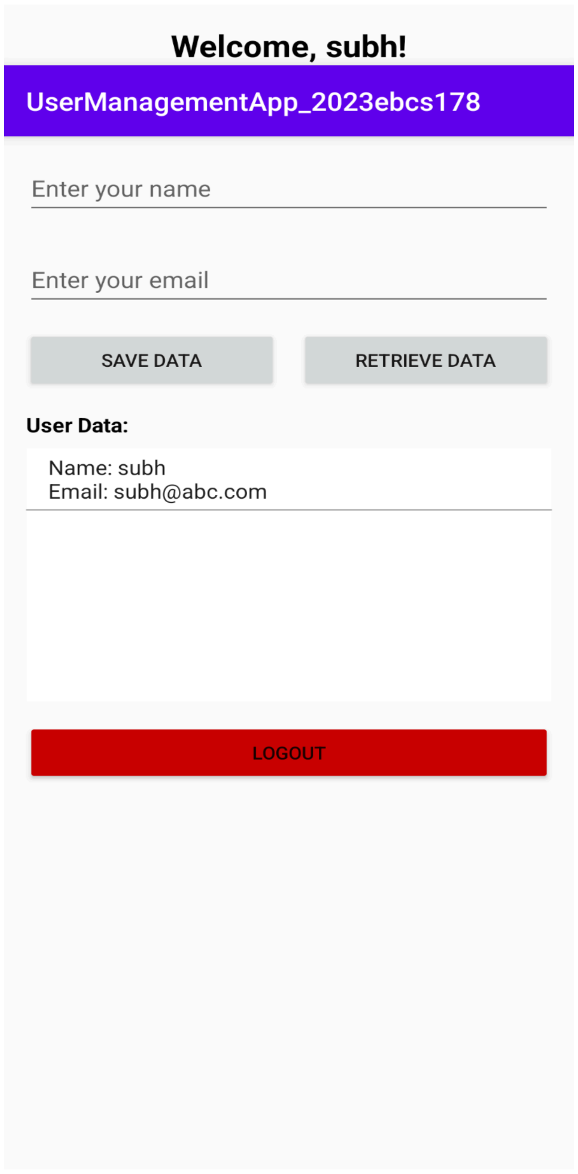
User Data:

Name: prachi
Email: prachi@abc.com

User: prachi
Name: prachi
Email: prachi@abc.in

User: heather
Name: heather
Email: heather@abc.com

- If logged in as a normal user – retrieve only the details which belongs to that user.



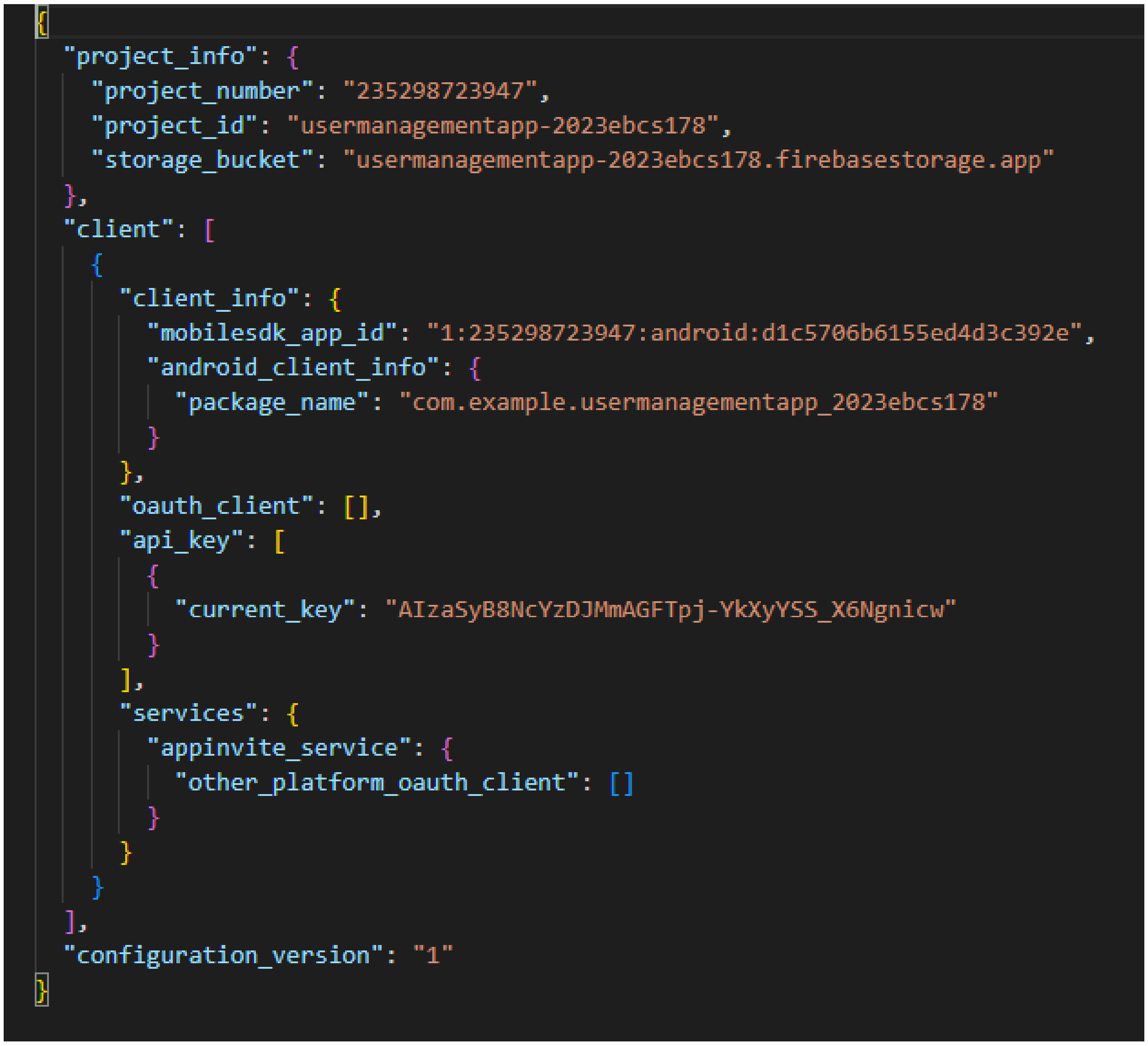
User Data:

Name: subh
Email: subh@abc.com

Deliverable:

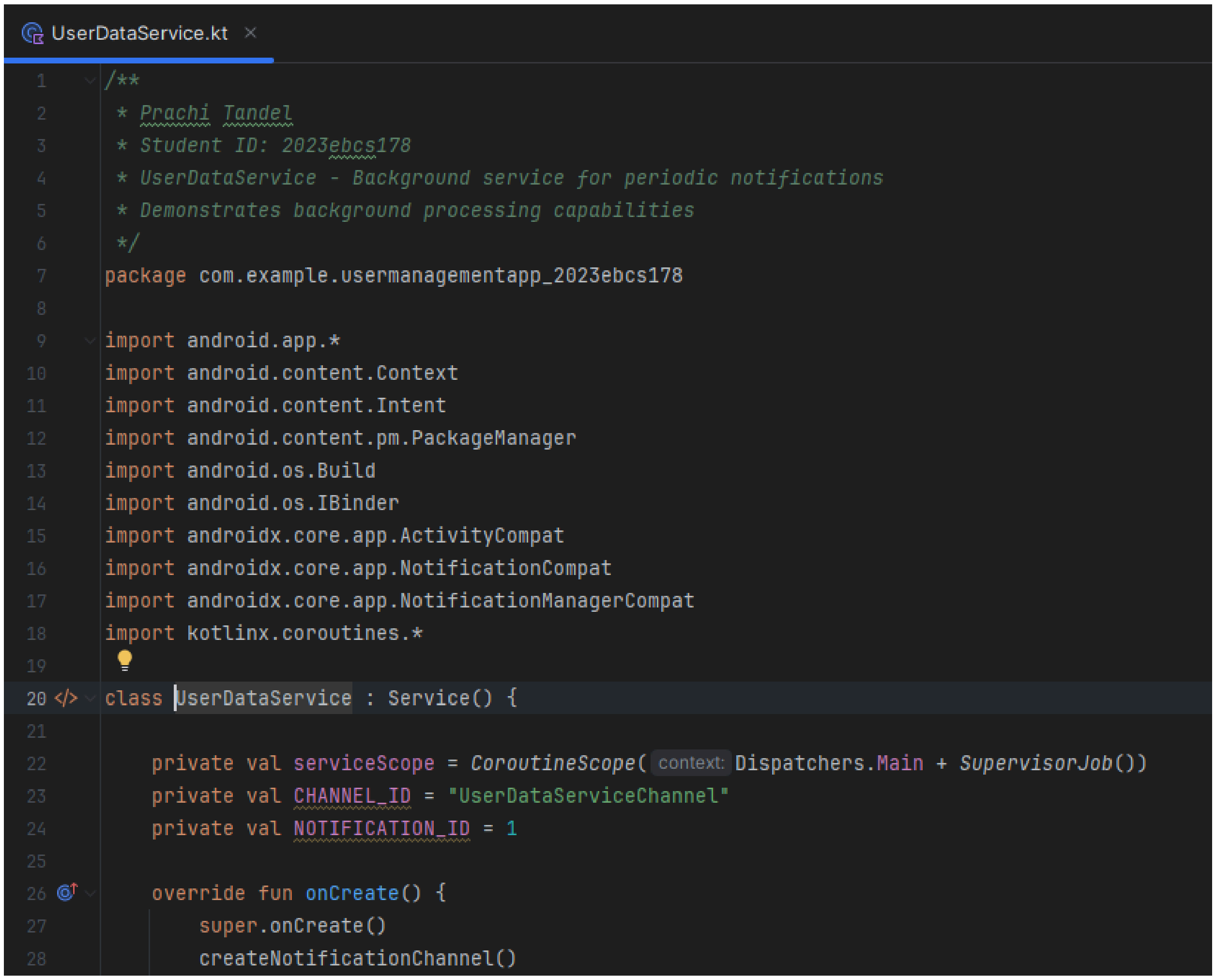
● Source code for Firebase integration and data retrieval.

google-services.json



```json
{
  "project_info": {
    "project_number": "235298723947",
    "project_id": "usermanagementapp-2023ebcs178",
    "storage_bucket": "usermanagementapp-2023ebcs178.firebasestorage.app"
  },
  "client": [
    {
      "client_info": {
        "mobilesdk_app_id": "1:235298723947:android:d1c5706b6155ed4d3c392e",
        "android_client_info": {
          "package_name": "com.example.usermanagementapp_2023ebcs178"
        }
      },
      "oauth_client": [],
      "api_key": [
        {
          "current_key": "AIzaSyB8NcYzDJMmAGFTpj-YkXyYSS_X6Ngnicw"
        }
      ],
      "services": {
        "appinvite_service": {
          "other_platform_oauth_client": []
        }
      }
    }
  ],
  "configuration_version": "1"
}
```

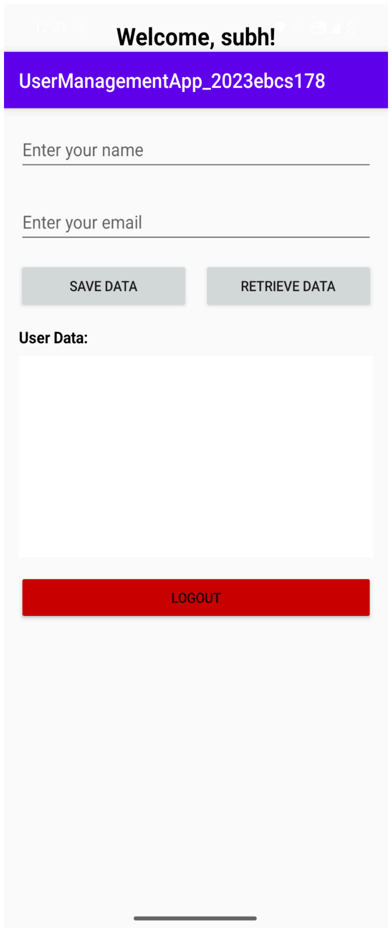
UserDataService.kt

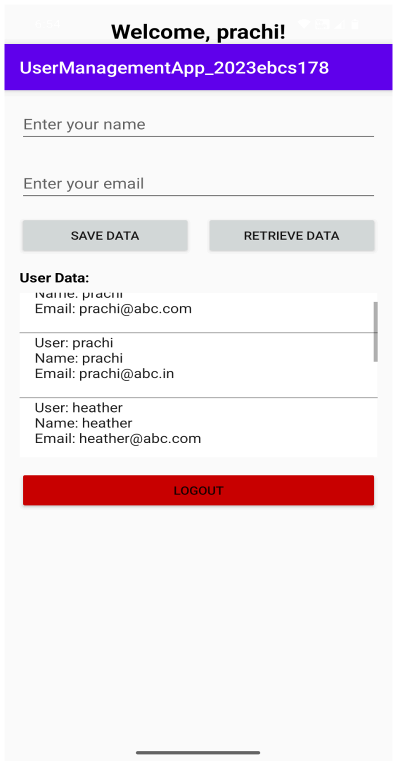


UserDataService.kt

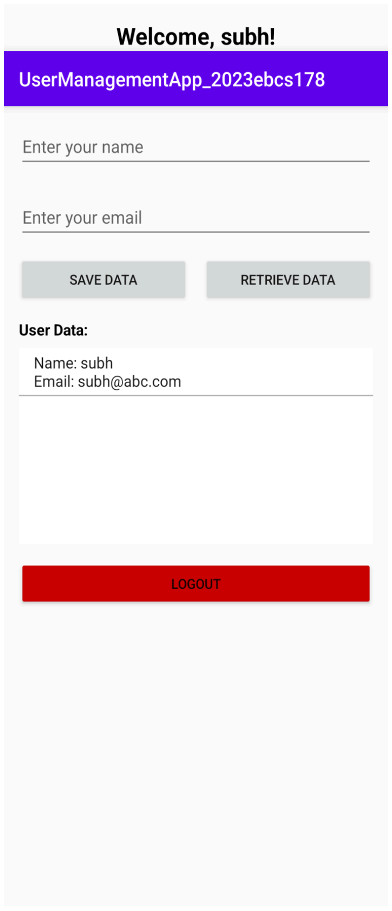
```kotlin
/**
 * Prachi Tandel
 * Student ID: 2023ebcs178
 * UserDataService - Background service for periodic notifications
 * Demonstrates background processing capabilities
 */
package com.example.usermanagementapp_2023ebcs178

import android.app.*
import android.content.Context
import android.content.Intent
import android.content.pm.PackageManager
import android.os.Build
import android.os.IBinder
import androidx.core.app.ActivityCompat
import androidx.core.app.NotificationCompat
import androidx.core.app.NotificationManagerCompat
import kotlinx.coroutines.*

class UserDataService : Service() {

    private val serviceScope = CoroutineScope( context: Dispatchers.Main + SupervisorJob())
    private val CHANNEL_ID = "UserDataServiceChannel"
    private val NOTIFICATION_ID = 1

    override fun onCreate() {
        super.onCreate()
        createNotificationChannel()
```

- Screenshot of the form and displayed data





```
Name: prachi
Email: prachi@abc.com

User: prachi
Name: prachi
Email: prachi@abc.in

User: heather
Name: heather
Email: heather@abc.com
```



```
Name: subh
Email: subh@abc.com
```