**Group Project Documentation   
MOBILE APP WITH ANDROID STUDIO**

***CONNECT SQLServer to Android Studio***

**Prepared by Group 2  
Trần Minh Trí**

**Huỳnh Đức Gia Bảo**

**Nguyễn Trần Khánh Hà**

**Nguyễn Tuấn Kiệt**

**Ho Chi Minh City, 2025**

[**1. Cài đặt và Chuẩn bị**](#_sw9pyrieihez) **3**

[Tạo Dự án](#_r74bhaws8894) 3

[SQLServer](#_83bjzmmeb81v) 5

[Tạo user riêng cho Android:](#_fj4o0fbsnajk) 6

[Cấu hình file my.ini của SQLServer](#_1y41pklcp7xi) 6

[1. Thêm thủ công dòng bind-address](#_yy2ikj4ate9p) 6

[2. Lưu file và khởi động lại SQLServer](#_oqvcnf6y895m) 7

[**2. Kết nối JDBC trong Android**](#_3t2sx6971pvn) **9**

[Thêm file .jar driver SQLServer](#_gx8drw4na30f) 9

[Kết nối:](#_qezv81527fwa) 12

[**3. Mô hình Code**](#_irzkm2hie2kp) **12**

[a. Kiến trúc MVC](#_vkbridj9vxm4) 12

[b. Giao diện XML](#_uh4kmffj5ni0) 13

[c. Model: TelescopeProduct.java](#_ccp6defyzato) 15

[TelescopeProduct.java](#_27qvygepssk4) 15

[TelescopeProductDAO.java](#_il5vkib63dwi) 15

[ProductController.java](#_wyf5zz5o4y8x) 17

[**4. Giao diện & Activity**](#_q6rz4n7njl5v) **18**

[FormActivity.java](#_skhfll6b8h8x) 18

[MainActivity.java](#_hwrogm9tp5js) 20

[DetailActivity.java](#_fahrw911l8fv) 21

[activity\_main.xml](#_2g9mzwntgrhj) 22

[item\_product.xml](#_3ntdtojtcg44) 23

[activity\_form.xml](#_yzi3n8s2vac3) 25

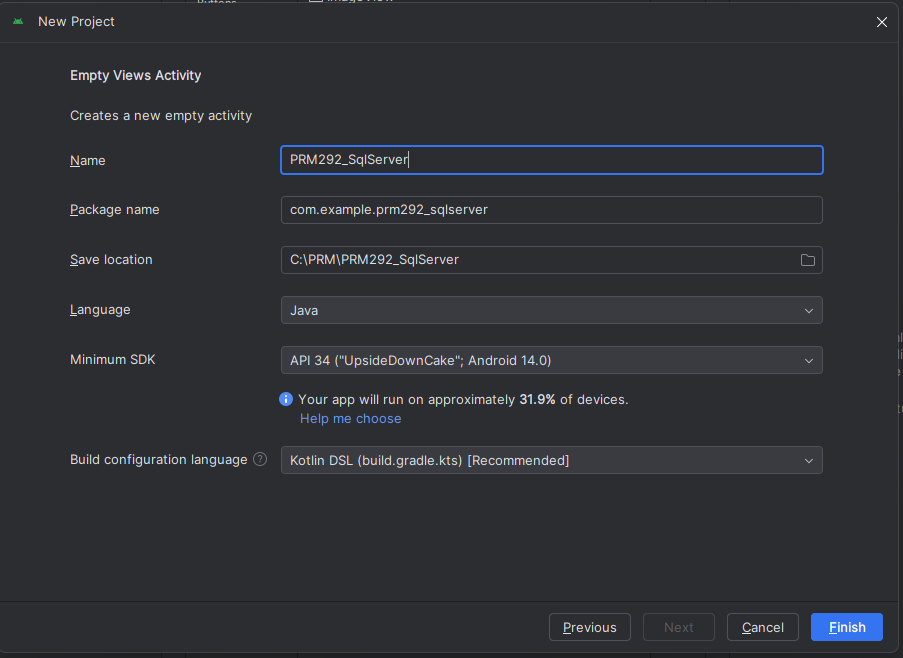
[activity\_detail.xml](#_42ynqv5o35g4) 25

[**5. AndroidManifest.xml**](#_9gngtph9j3y3) **26**

[**6. Sản phẩm hoàn thiện**](#_y7vto7ijyz26) **27**

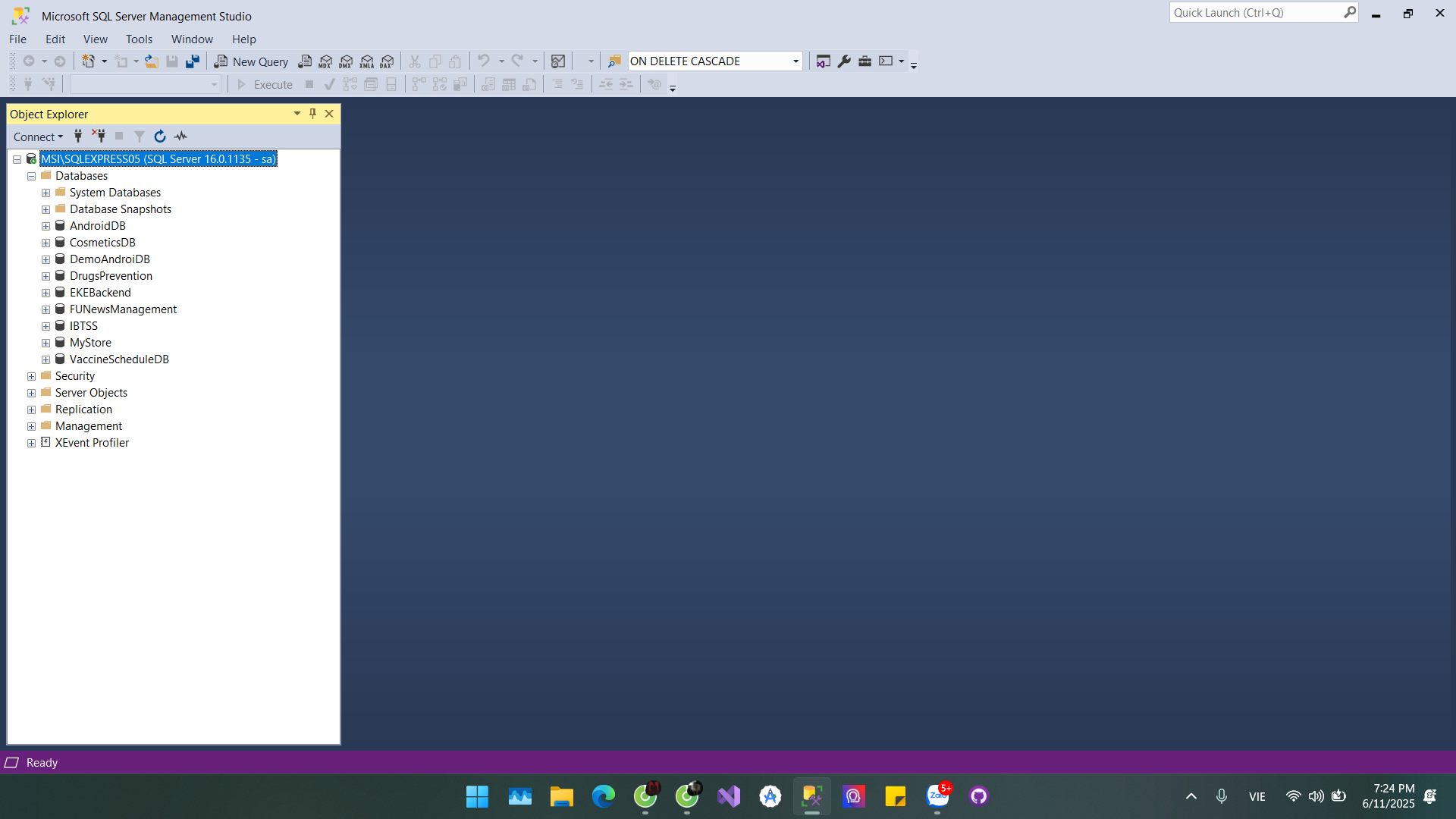
# **1. Cài đặt và Chuẩn bị**

## **Tạo Dự án**

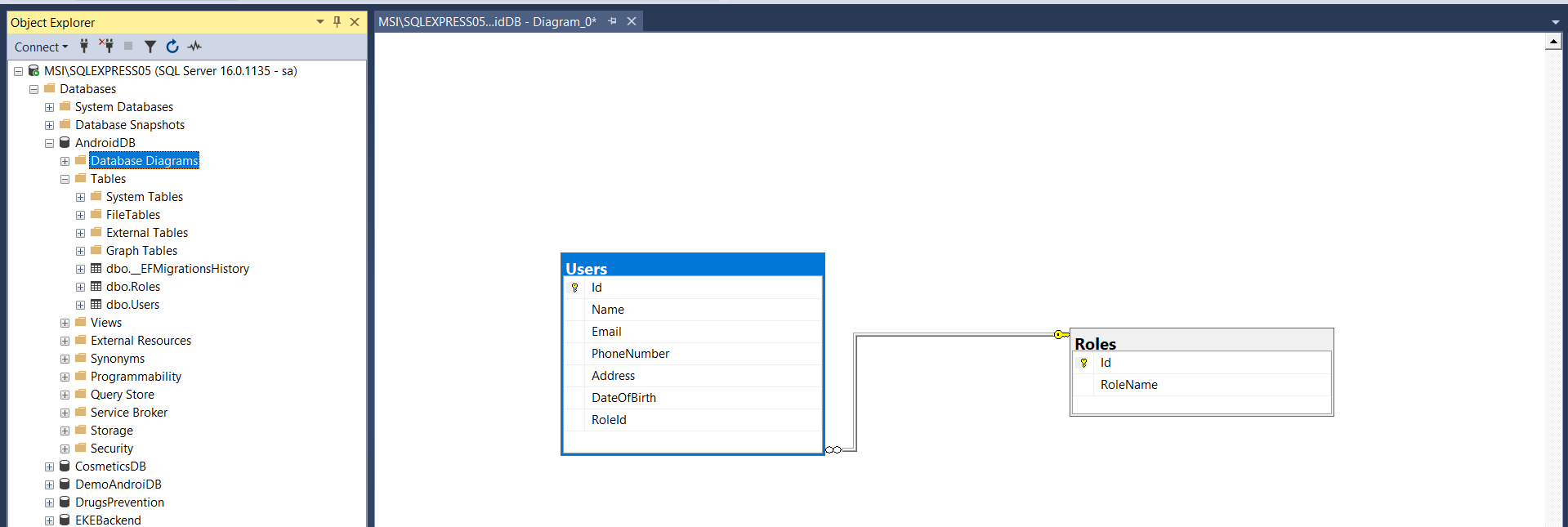
* Mở Android Studio chọn File trên thanh công cụ => New => New Project  
  
* Sau đó chọn Empty Views Activity  
  
* Sau khi chọn sẽ hiển thị ra bảng thông tin cấu hình của dự án  
    
  Lưu ý: Chọn Language và “ Java “

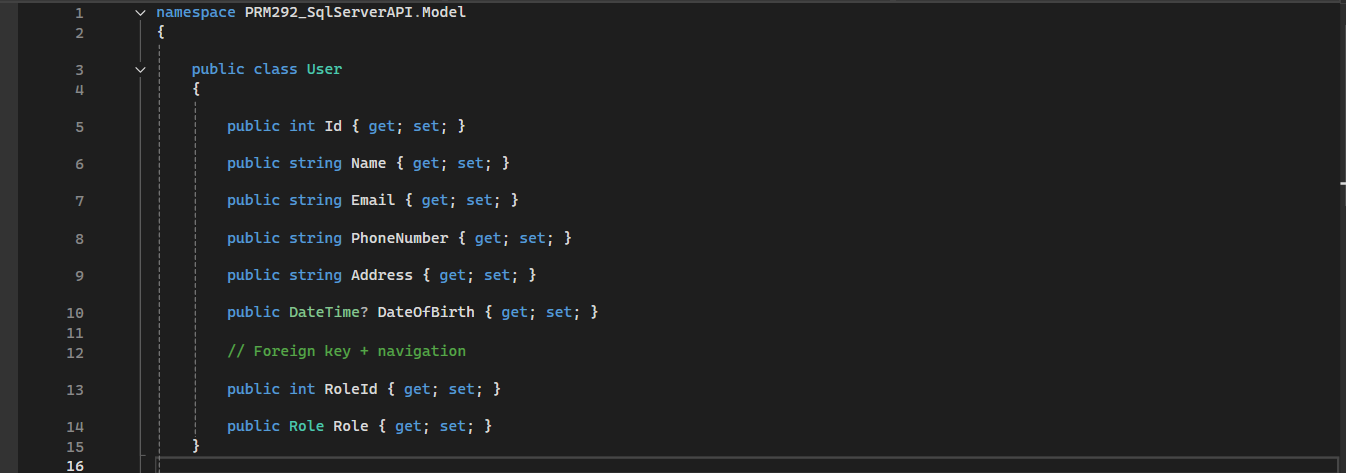
## **SQLServer**

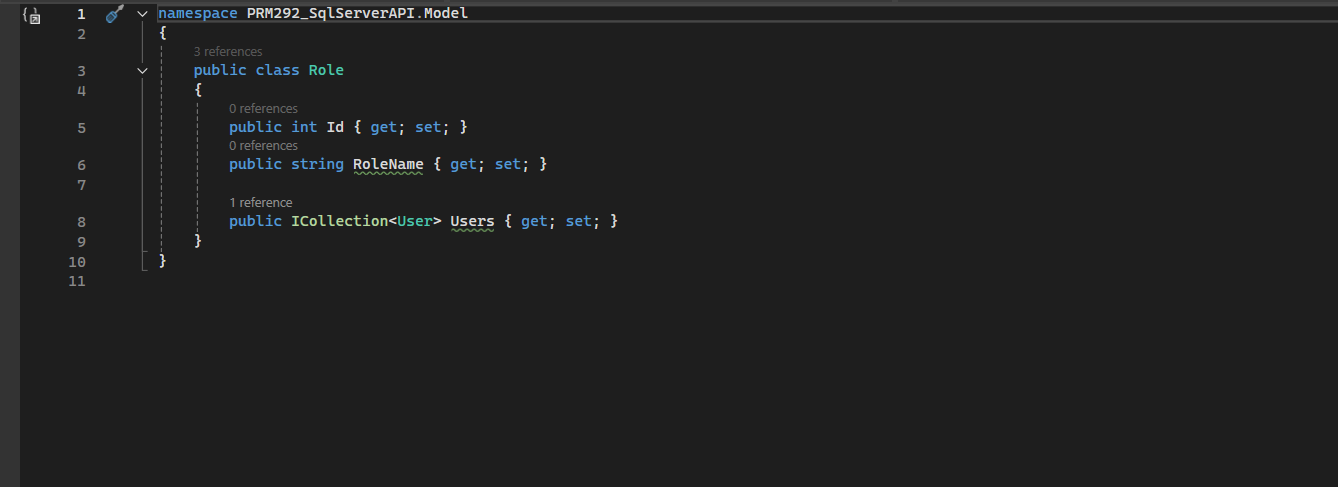
* Cài **SQLServer** trên Windows



* Tạo database: androiDandroiDB







* Tạo bảng: Chạy add-migration trong Pakage Manage Console trong Visual Studio để tạo Data Base trong SQL Server

## **Tạo user riêng cho Android:**

Sau khi tạo Migration thì thay code migration bằng

using System;

using Microsoft.EntityFrameworkCore.Migrations;

#nullable disable

namespace PRM292\_SqlServerAPI.Migrations

{

/// <inheritdoc />

public partial class CreateDB : Migration

{

/// <inheritdoc />

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.CreateTable(

name: "Roles",

columns: table => new

{

Id = table.Column<int>(type: "int", nullable: false)

.Annotation("SqlServer:Identity", "1, 1"),

RoleName = table.Column<string>(type: "nvarchar(max)", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Roles", x => x.Id);

});

migrationBuilder.CreateTable(

name: "Users",

columns: table => new

{

Id = table.Column<int>(type: "int", nullable: false)

.Annotation("SqlServer:Identity", "1, 1"),

Name = table.Column<string>(type: "nvarchar(max)", nullable: false),

Email = table.Column<string>(type: "nvarchar(max)", nullable: false),

PhoneNumber = table.Column<string>(type: "nvarchar(max)", nullable: false),

Address = table.Column<string>(type: "nvarchar(max)", nullable: false),

DateOfBirth = table.Column<DateTime>(type: "datetime2", nullable: true),

RoleId = table.Column<int>(type: "int", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Users", x => x.Id);

table.ForeignKey(

name: "FK\_Users\_Roles\_RoleId",

column: x => x.RoleId,

principalTable: "Roles",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateIndex(

name: "IX\_Users\_RoleId",

table: "Users",

column: "RoleId");

// Thêm dữ liệu vào bảng Roles

migrationBuilder.InsertData(

table: "Roles",

columns: new[] { "Id", "RoleName" },

values: new object[,]

{

{ 1, "Admin" },

{ 2, "User" },

{ 3, "Guest" }

});

// Thêm dữ liệu vào bảng Users

migrationBuilder.InsertData(

table: "Users",

columns: new[] { "Id", "Name", "Email", "PhoneNumber", "Address", "DateOfBirth", "RoleId" },

values: new object[,]

{

{ 1, "Nguyen Van A", "a@example.com", "0123456789", "Ha Noi", new DateTime(1990, 1, 1), 1 },

{ 2, "Tran Thi B", "b@example.com", "0987654321", "Ho Chi Minh", new DateTime(1995, 5, 10), 2 }

});

}

/// <inheritdoc />

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropTable(

name: "Users");

migrationBuilder.DropTable(

name: "Roles");

}

}

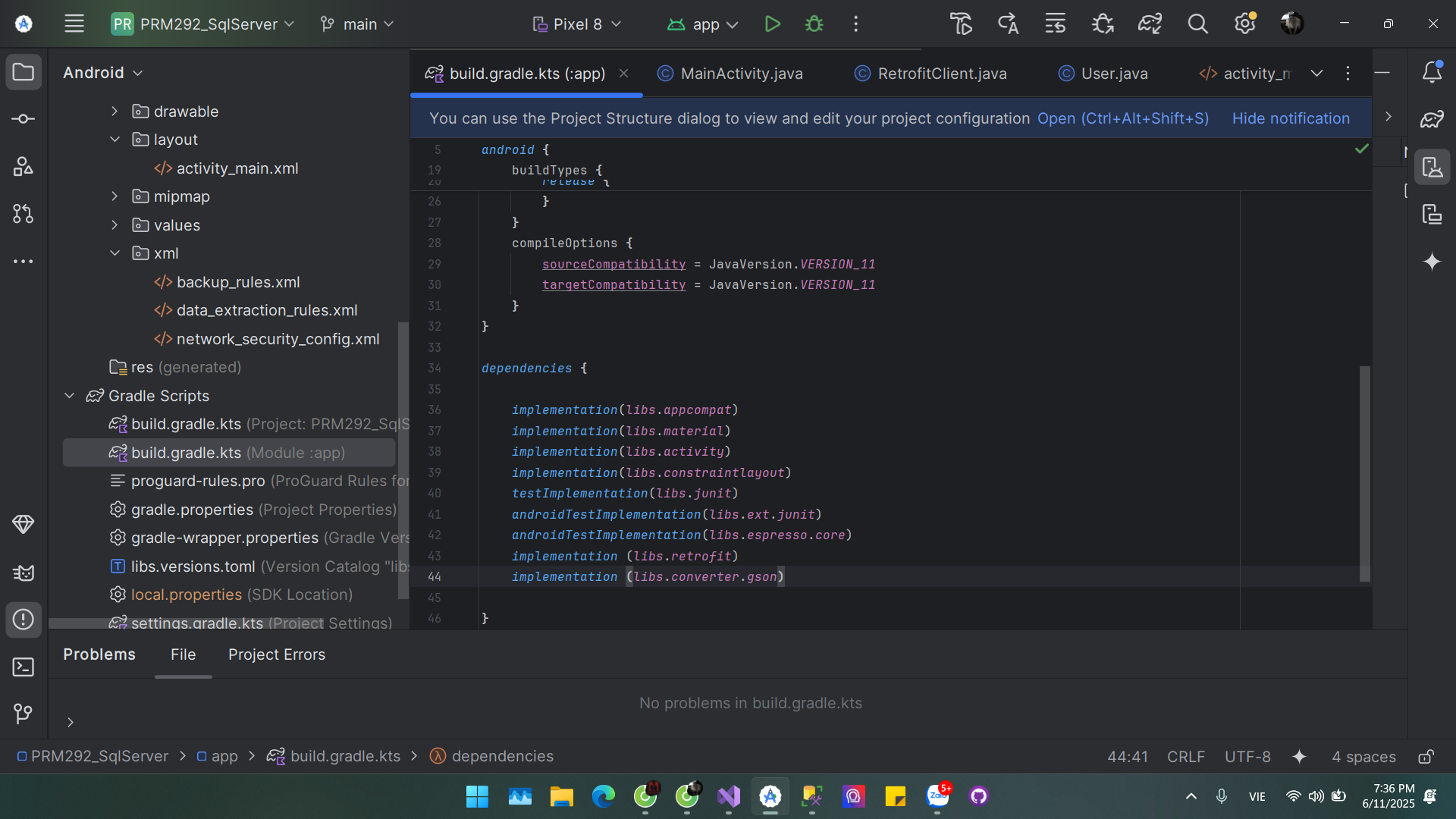
}

để tạo User mới

**Thêm vào build.gradle.kts:**

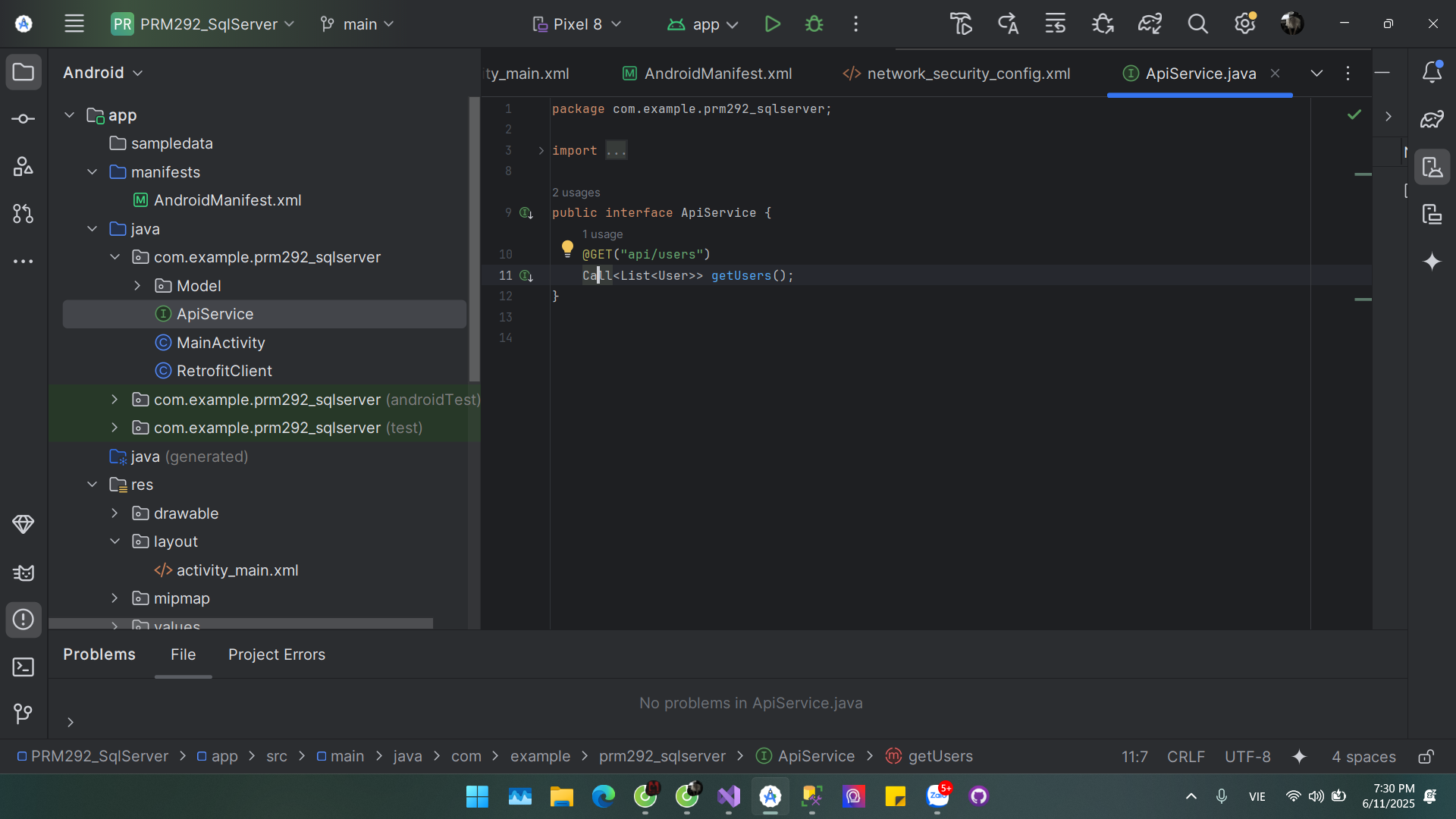
*implementation* (*libs*.*retrofit*)

*implementation* (*libs*.*converter*.*gson*)

* 

## **Kết nối:**

Tạo 3 class java trong thư mục java



Thêm code vào  
:  
package com.example.prm292\_sqlserver;

import java.util.List;

import com.example.prm292\_sqlserver.Model.User;

import retrofit2.Call;

import retrofit2.http.GET;

public interface ApiService {

@GET("api/users")

Call<List<User>> getUsers();

}

package com.example.prm292\_sqlserver;

import android.os.Bundle;

import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.example.prm292\_sqlserver.Model.User;

import java.util.List;

import retrofit2.Call;

import retrofit2.Callback;

import retrofit2.Response;

public class MainActivity extends AppCompatActivity {

private TextView txtResult;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

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

txtResult = findViewById(R.id.*txtResult*);

ApiService apiService = RetrofitClient.*getRetrofitInstance*().create(ApiService.class);

Call<List<User>> call = apiService.getUsers();

call.enqueue(new Callback<List<User>>() {

@Override

public void onResponse(Call<List<User>> call, Response<List<User>> response) {

if (response.isSuccessful() && response.body() != null) {

StringBuilder result = new StringBuilder();

for (User user : response.body()) {

result.append(user.getName())

.append(" - ")

.append(user.getEmail())

.append("\n");

}

txtResult.setText(result.toString());

} else {

txtResult.setText("No data received.");

}

}

@Override

public void onFailure(Call<List<User>> call, Throwable t) {

txtResult.setText("Error: " + t.getMessage());

}

});

}

}

package com.example.prm292\_sqlserver;

import retrofit2.Retrofit;

import retrofit2.converter.gson.GsonConverterFactory;

public class RetrofitClient {

private static Retrofit *retrofit*;

private static final String *BASE\_URL* = "http://10.0.2.2:5000/";

// private static final String BASE\_URL = "http://10.0.2.2:7257/";

public static Retrofit getRetrofitInstance() {

if (*retrofit* == null) {

*retrofit* = new Retrofit.Builder()

.baseUrl(*BASE\_URL*)

.addConverterFactory(GsonConverterFactory.*create*())

.build();

}

return *retrofit*;

}

}

\*\*Lưu ý private static final String *BASE\_URL* = "http://10.0.2.2:5000/";

nghĩa là: http://[ip của máy]:[port của url localhost]/’’

Tạo folder Model/User để khai báo Bảng User  
package com.example.prm292\_sqlserver.Model;

public class User {

private int id;

private String name;

private String email;

// Getters

public int getId() {

return id;

}

public String getName() {

return name;

}

public String getEmail() {

return email;

}

}

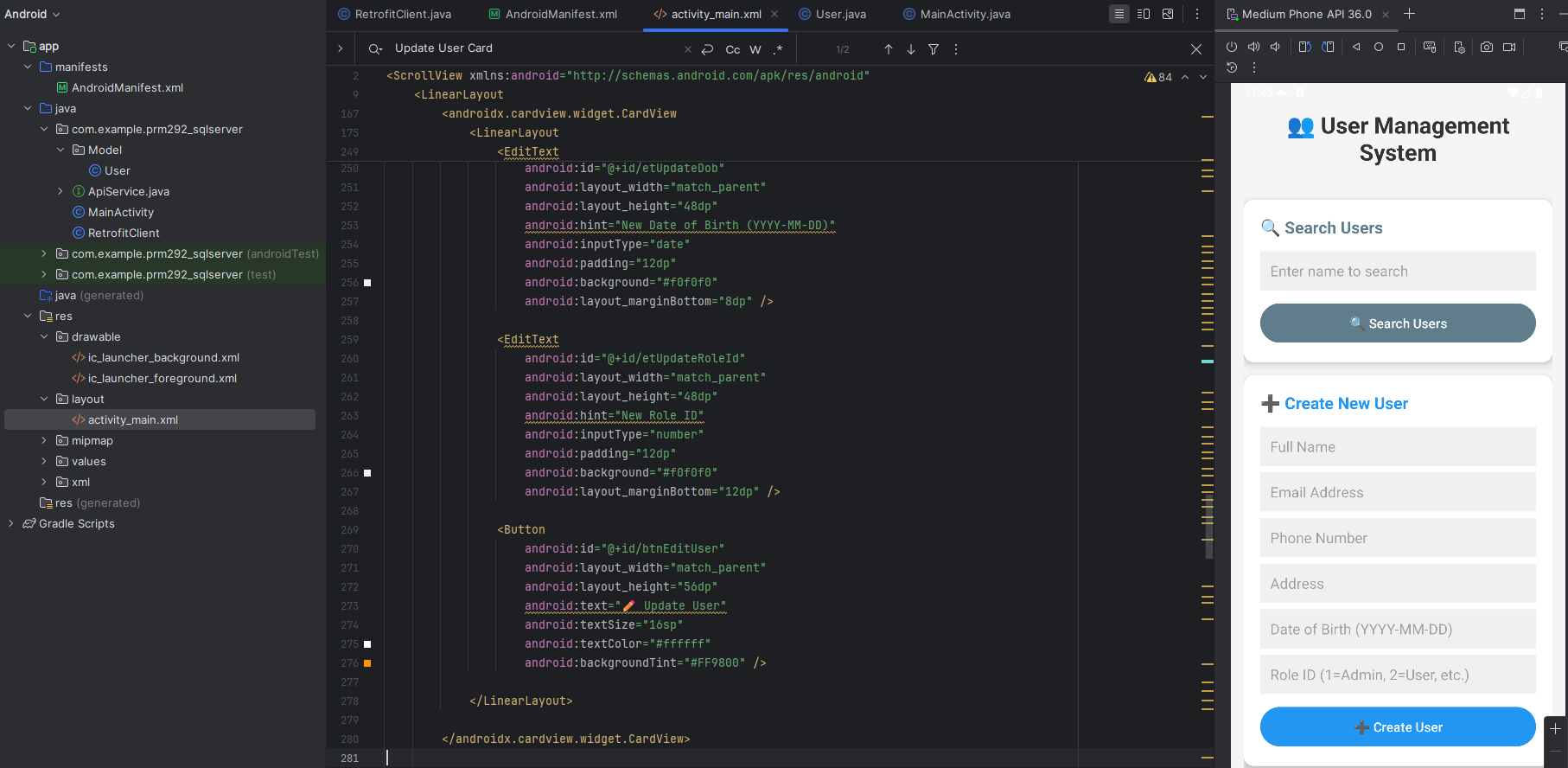
# **2.1 Mô hình Code**

## a.Setup ASP.net API

* Model: AppDbContext.cs, Role.cs, User.cs
* Controller: UserController.cs
* View: Android Studio
  + ApiService.java
  + MainActivity.java
  + RerofitClient.java
  + Model/User.java

## b. Giao diện XML

* activity\_main.xml: ListView , CRUD của form user



## c. Model: User.java

### User.java

* **Chức năng**: Lưu trữ dữ liệu tài khoản của User.
* Đại diện cho 1 dòng trong bảng User.
* Gồm các thuộc tính: id, Name, email,,..

## package com.example.prm292\_sqlserver.Model;

## 

## public class User {

## private int id;

## private String name;

## private String email;

## 

## // Getters

## public int getId() {

## return id;

## }

## 

## public String getName() {

## return name;

## }

## 

## public String getEmail() {

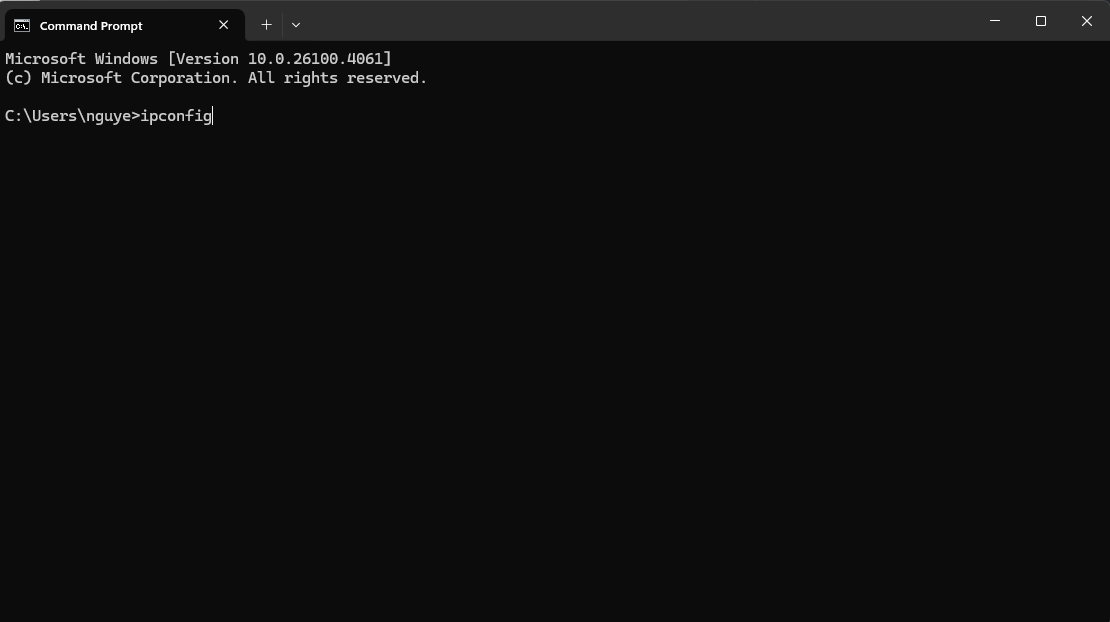
## return email;

## }

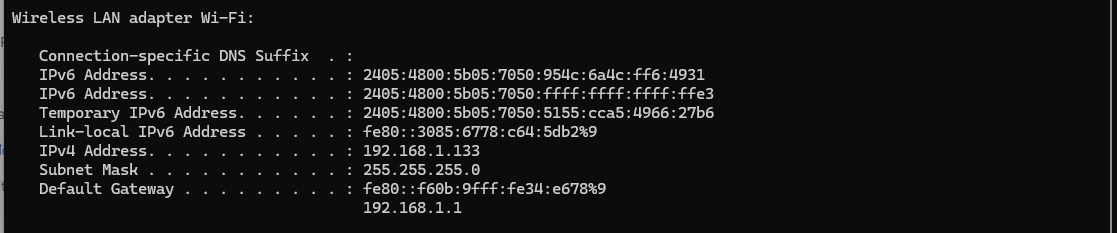
## }

# **2.2 Lấy IP address để config**

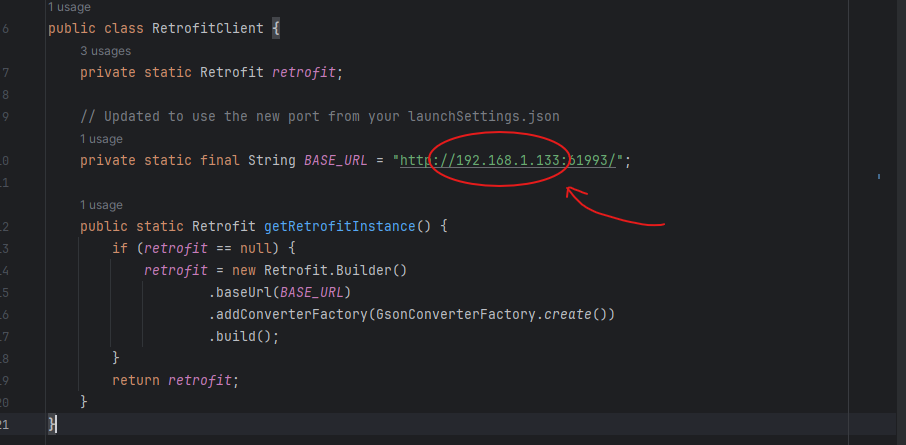
## Bước 1: Mở terminal (CMD) -> nhập “ipconfig”



## Bước 2: Tìm IPv4 , nếu bạn kết nối bằng wifi thì tìm đế địa chỉ dành cho wifi (nếu kết nối mạng LAN thì tìm đến mục Ethernet để lấy IP)



## Bước 3: Vào file “RetrofitClient.java” để set up IP của máy bạn

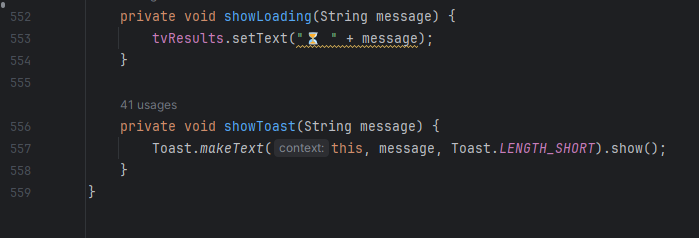
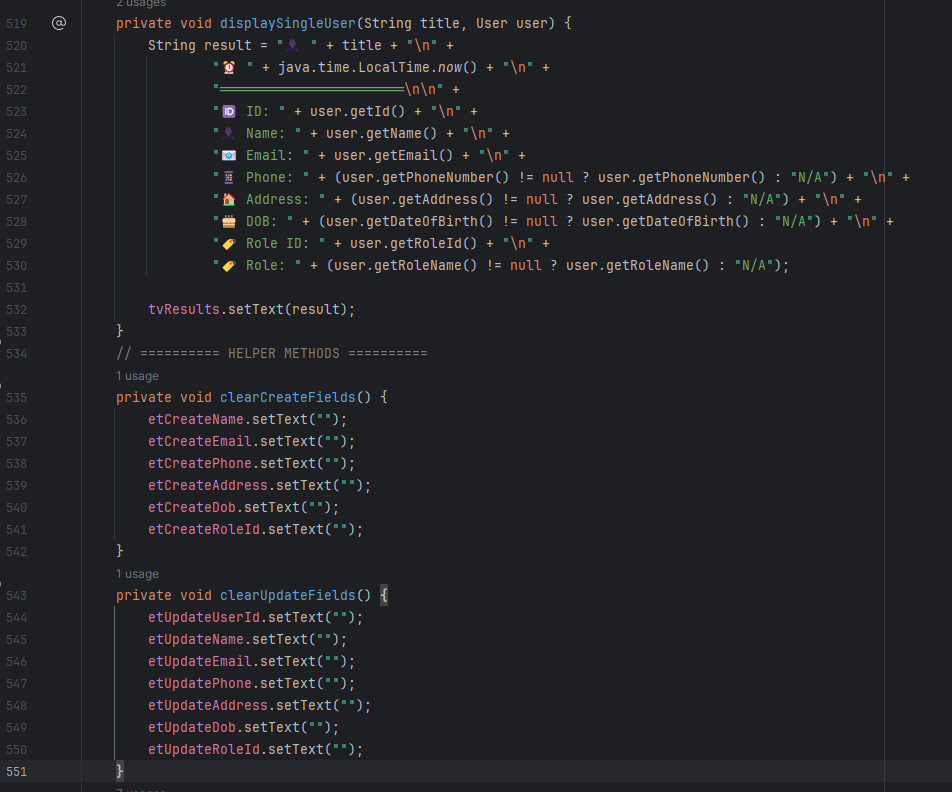
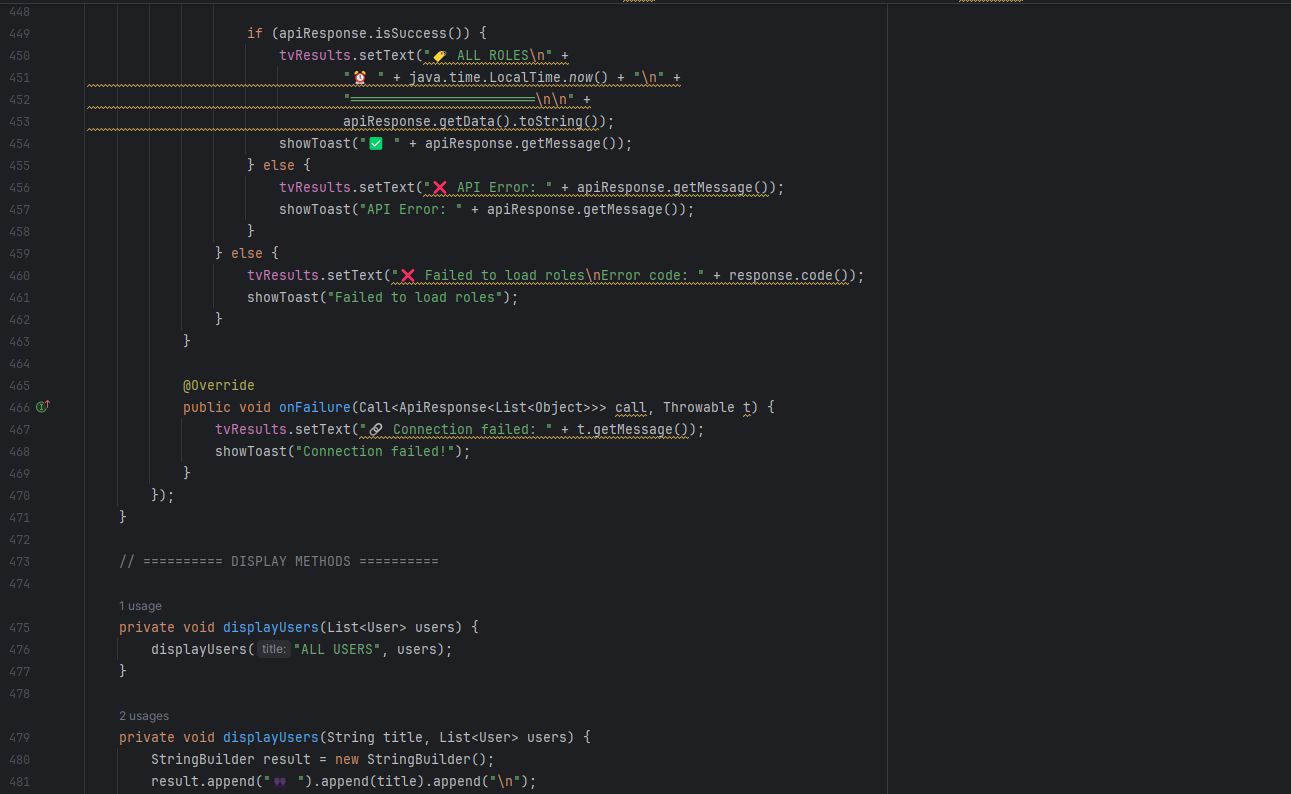
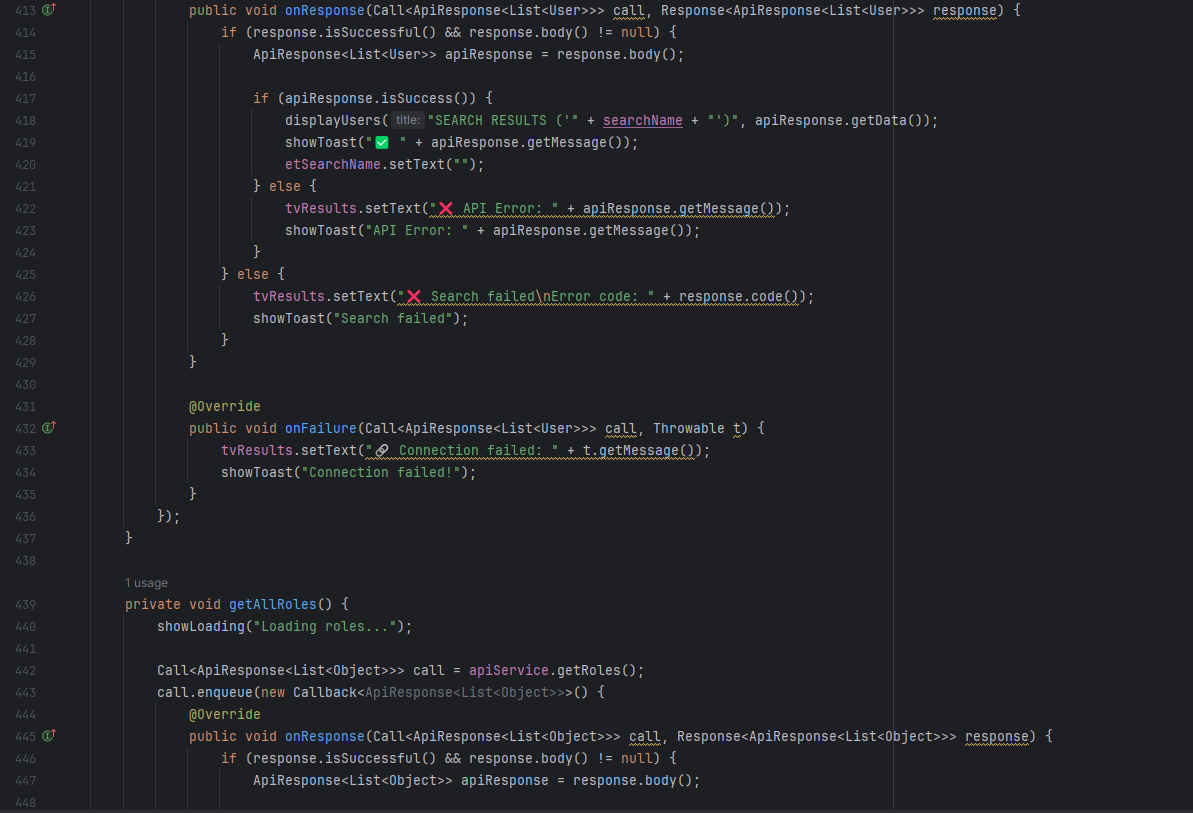
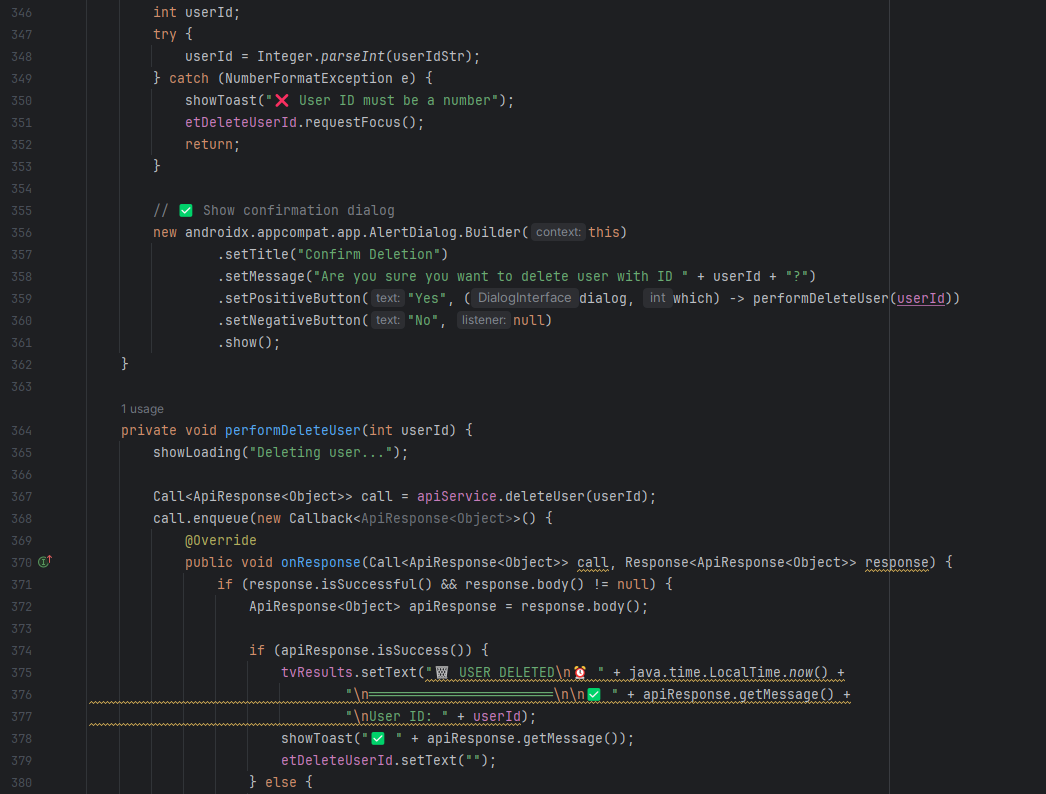
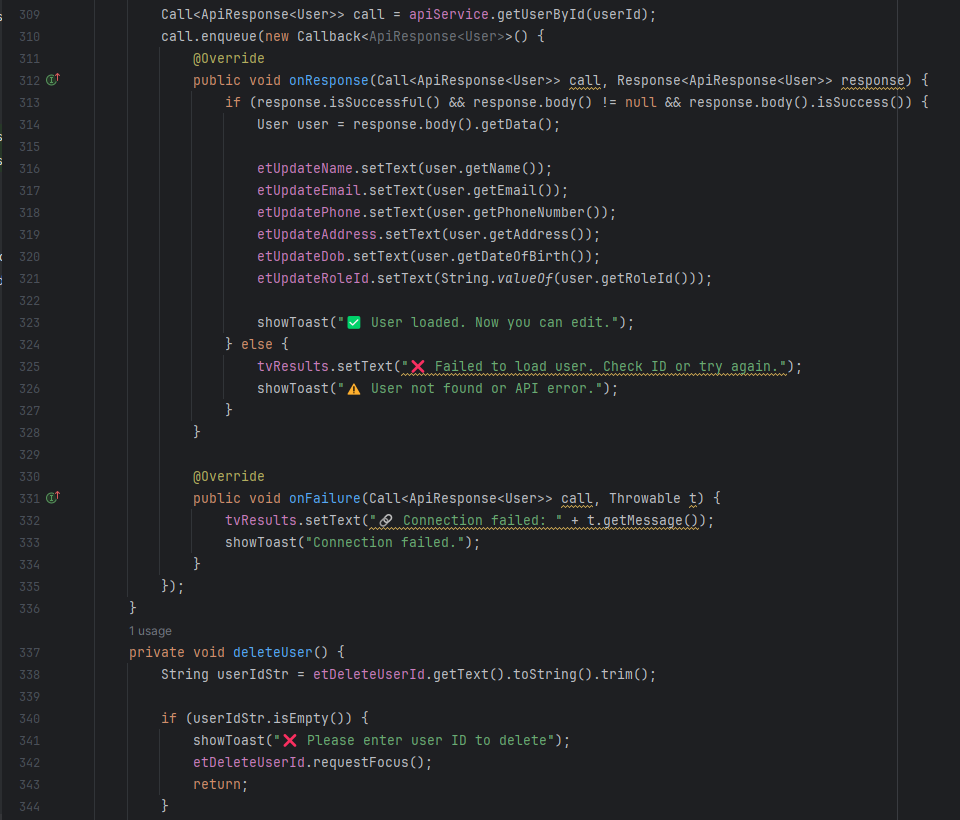
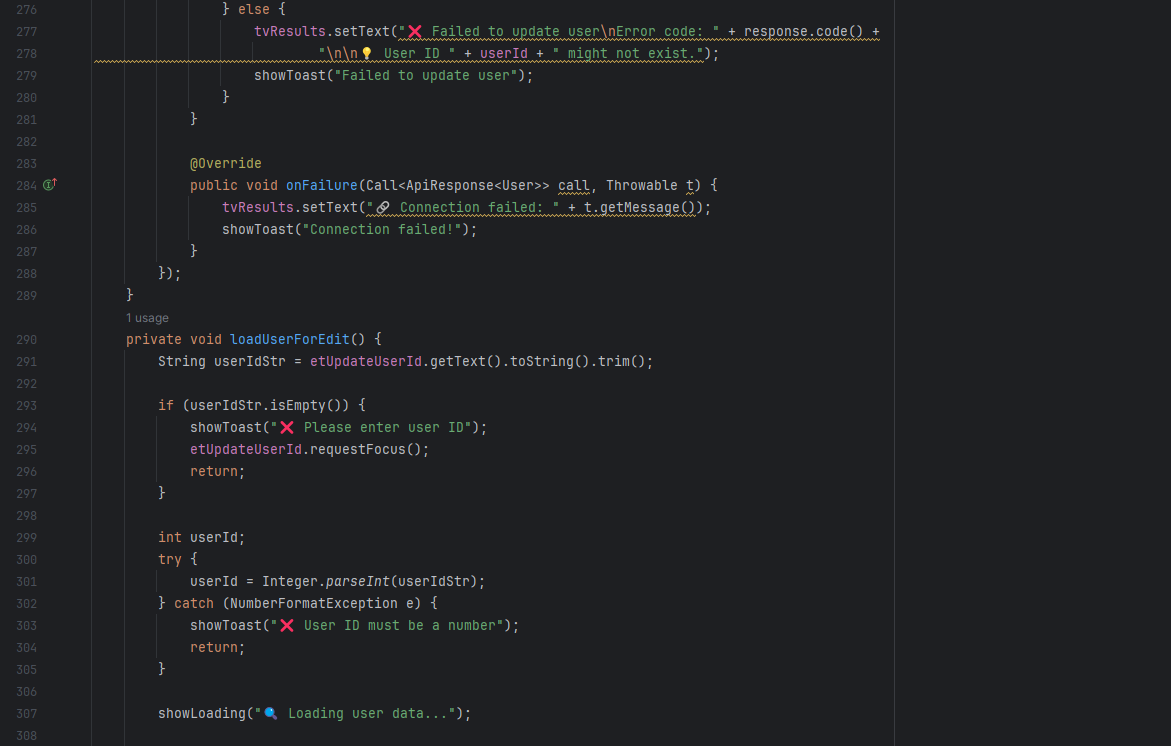
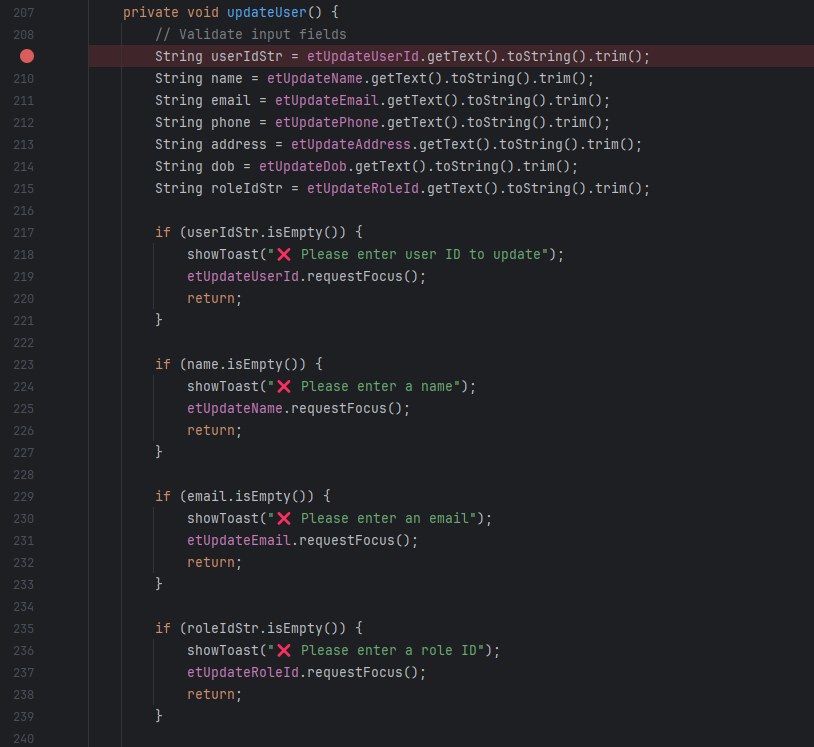
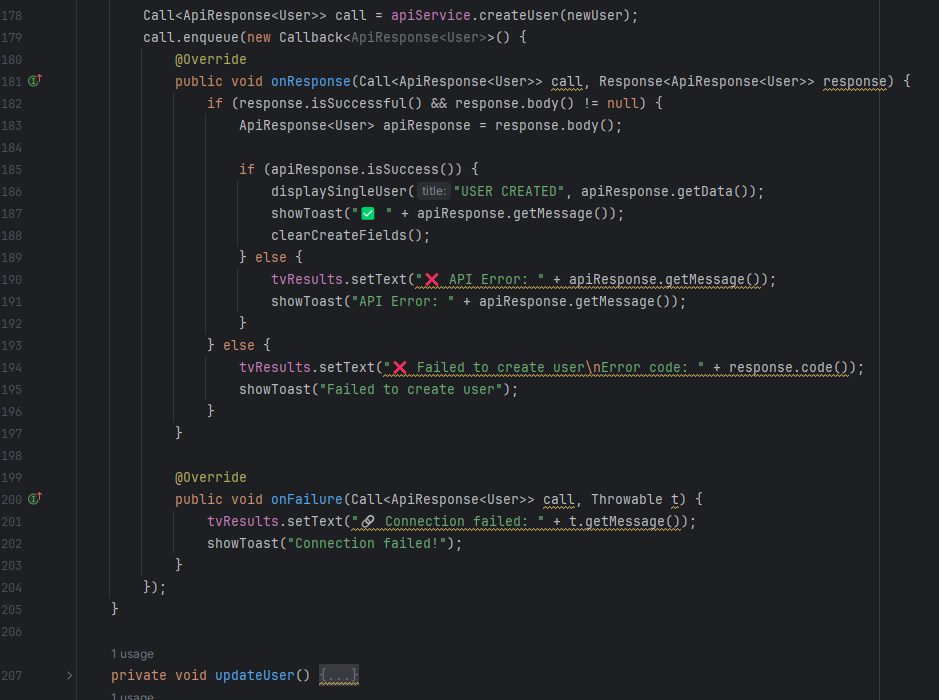
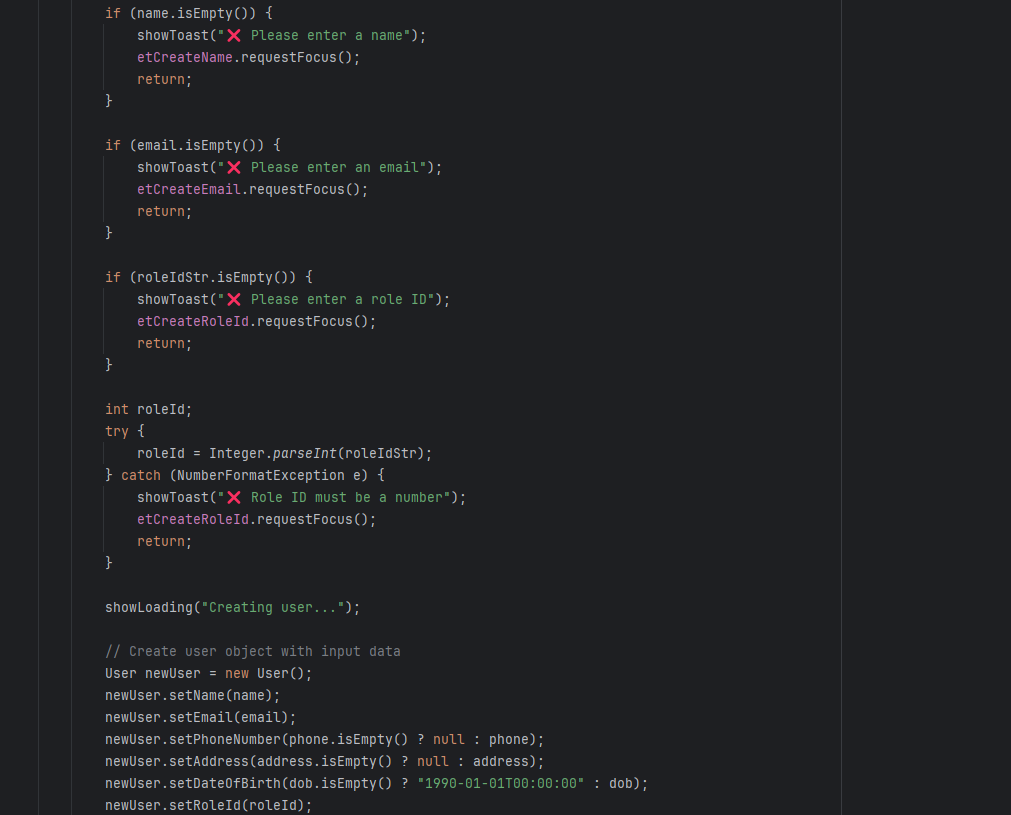
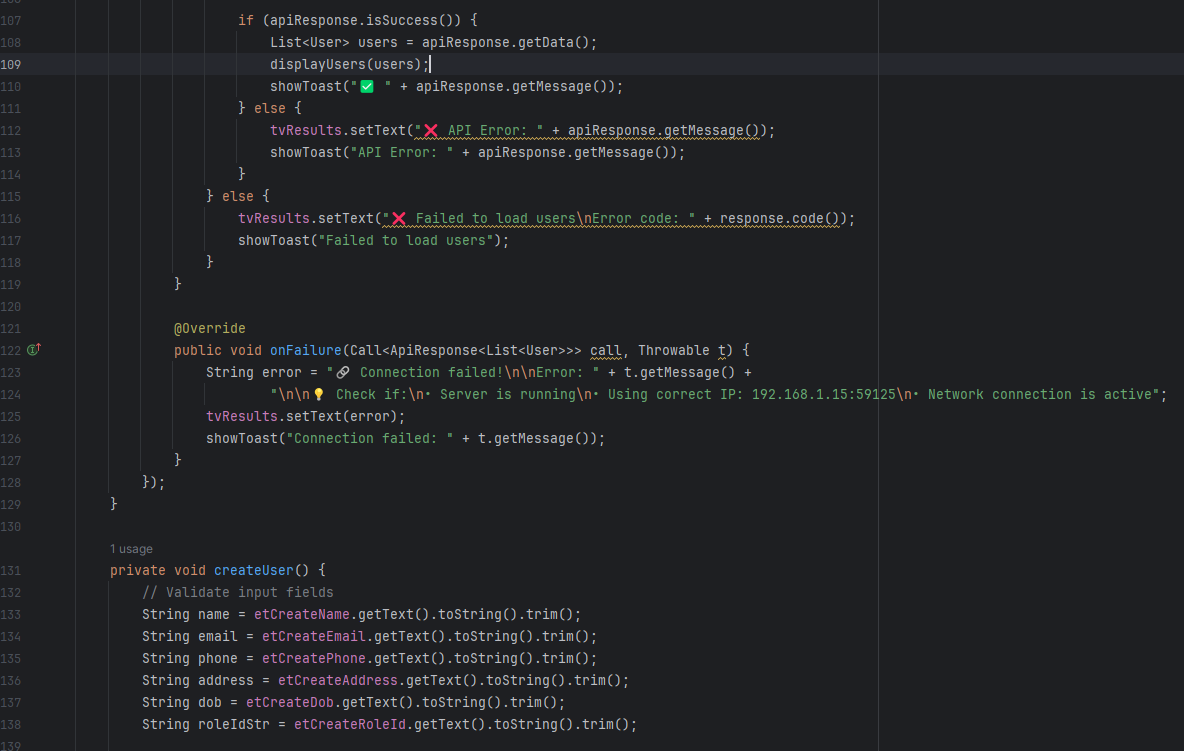
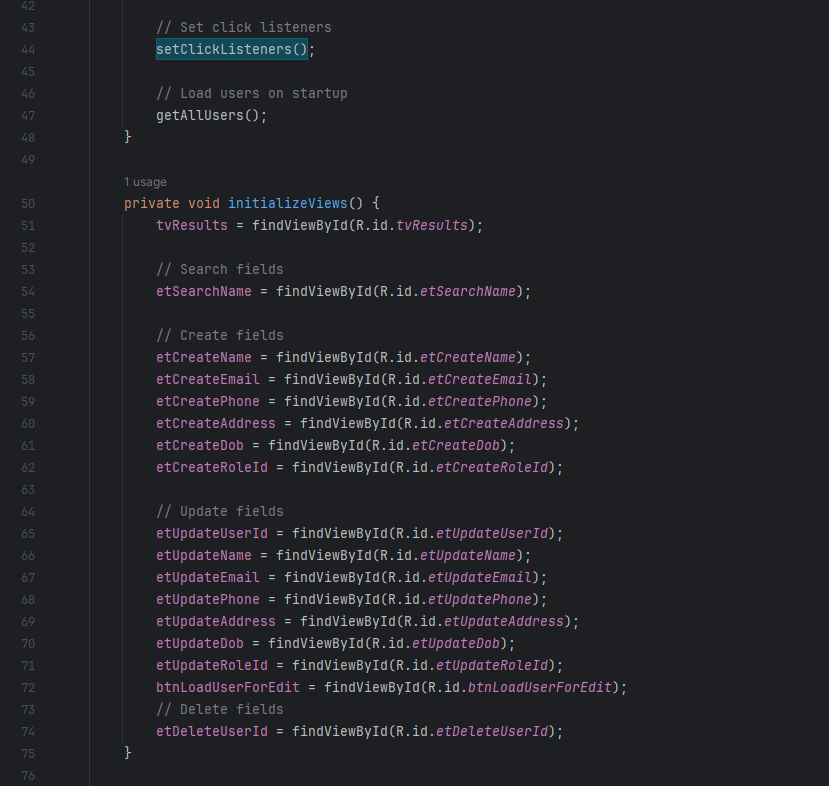
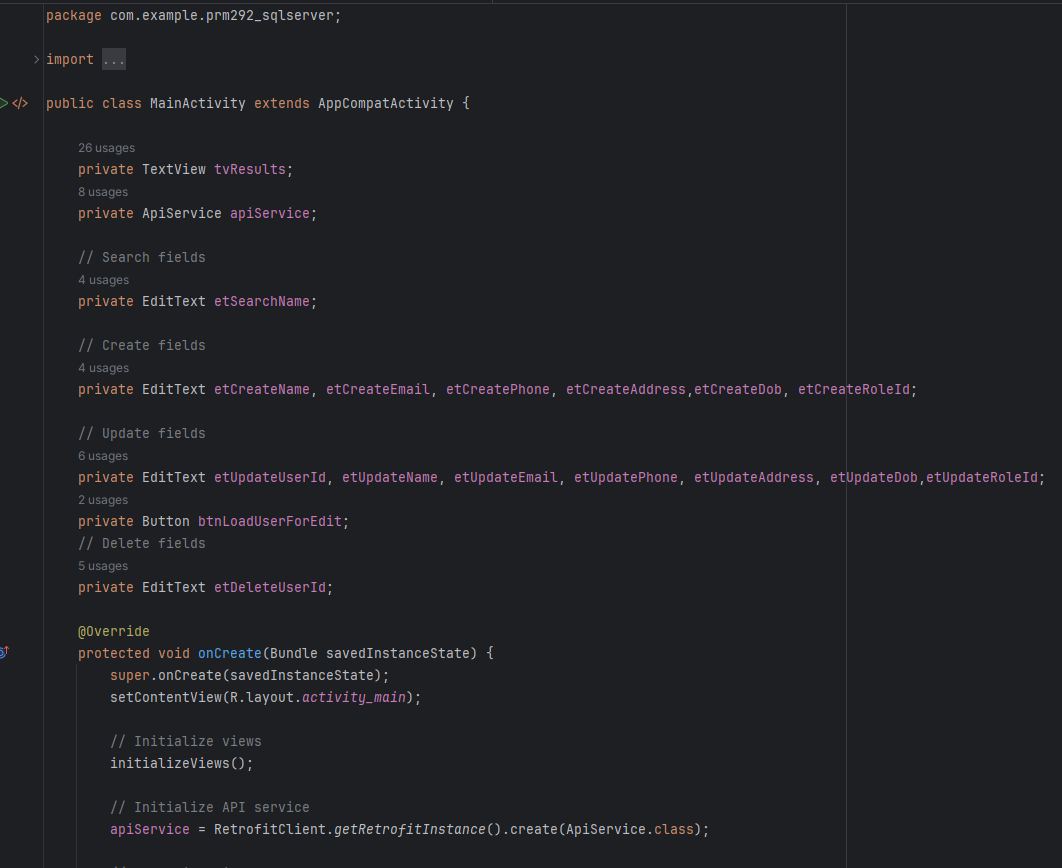


### 

# **3. Giao diện & Activity**

### **3.1 MainActivity.java**

* **Chức năng:** Quản lý toàn bộ hoạt động CRUD của người dùng:  
  + Hiển thị danh sách người dùng
  + Tìm kiếm theo tên
  + Thêm, chỉnh sửa và xóa người dùng
  + Hiển thị vai trò người dùng
* **Các chức năng chính:**
  + getAllUsers(): Lấy danh sách người dùng từ API
  + createUser(): Tạo người dùng mới từ form nhập liệu
  + loadUserForEdit(id): Tải dữ liệu người dùng theo ID để chỉnh sửa
  + updateUser(id): Cập nhật người dùng với thông tin mới
  + deleteUser(id): Xóa người dùng với xác nhận
  + searchUsers(name): Tìm kiếm người dùng theo tên
  + getRoles(): Hiển thị danh sách role hiện có



### 

### 

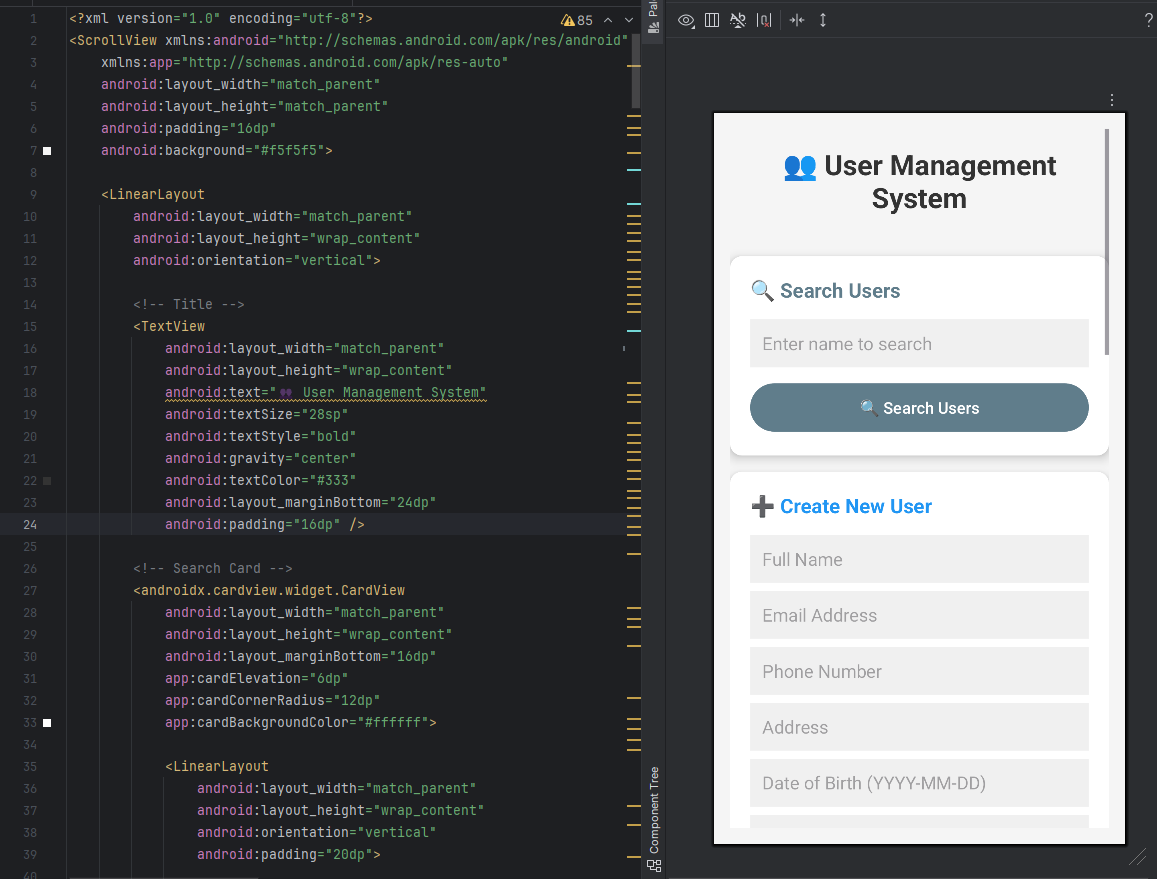
### 

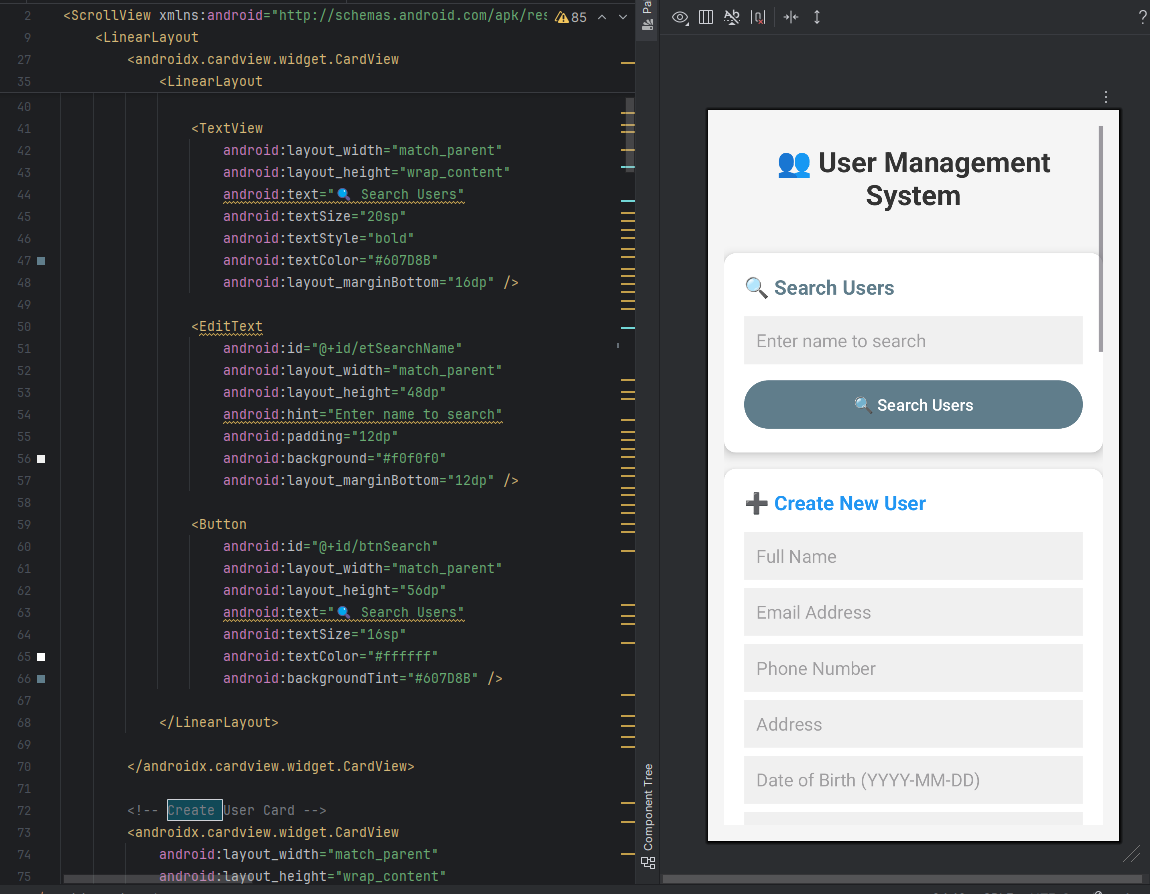
### **4.2 activity\_main.xml**

* **Giao diện chính (dưới dạng ScrollView)** gồm các phần:

#### a) Search Users

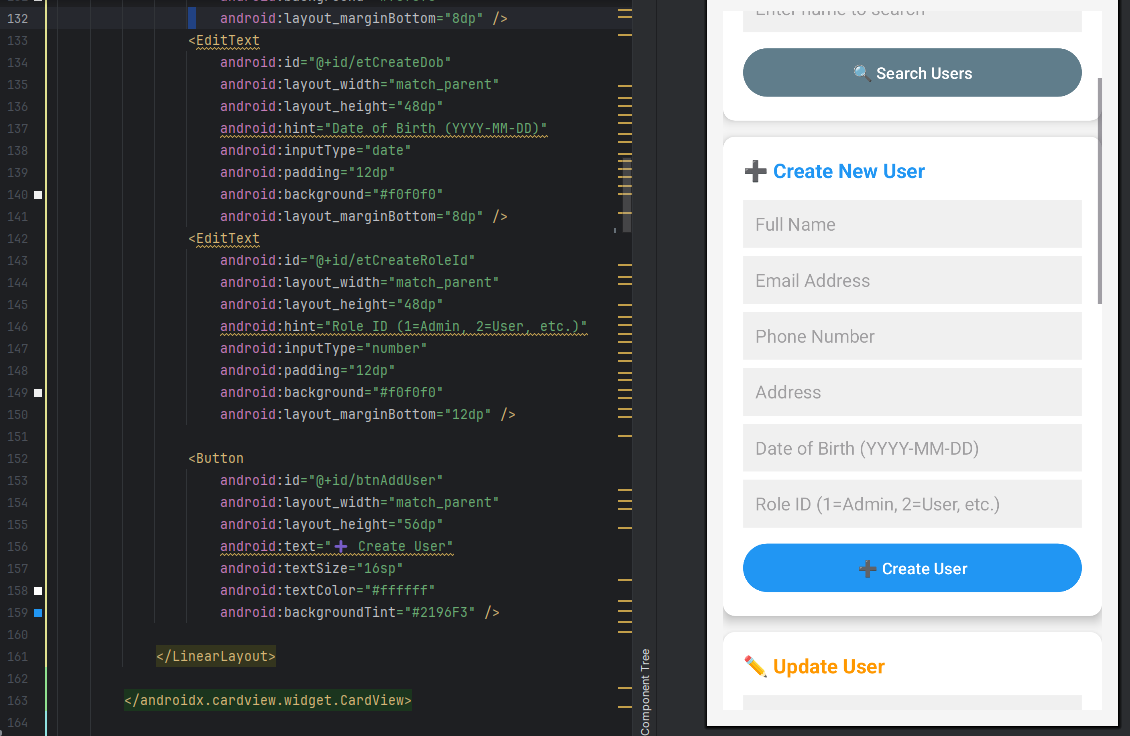
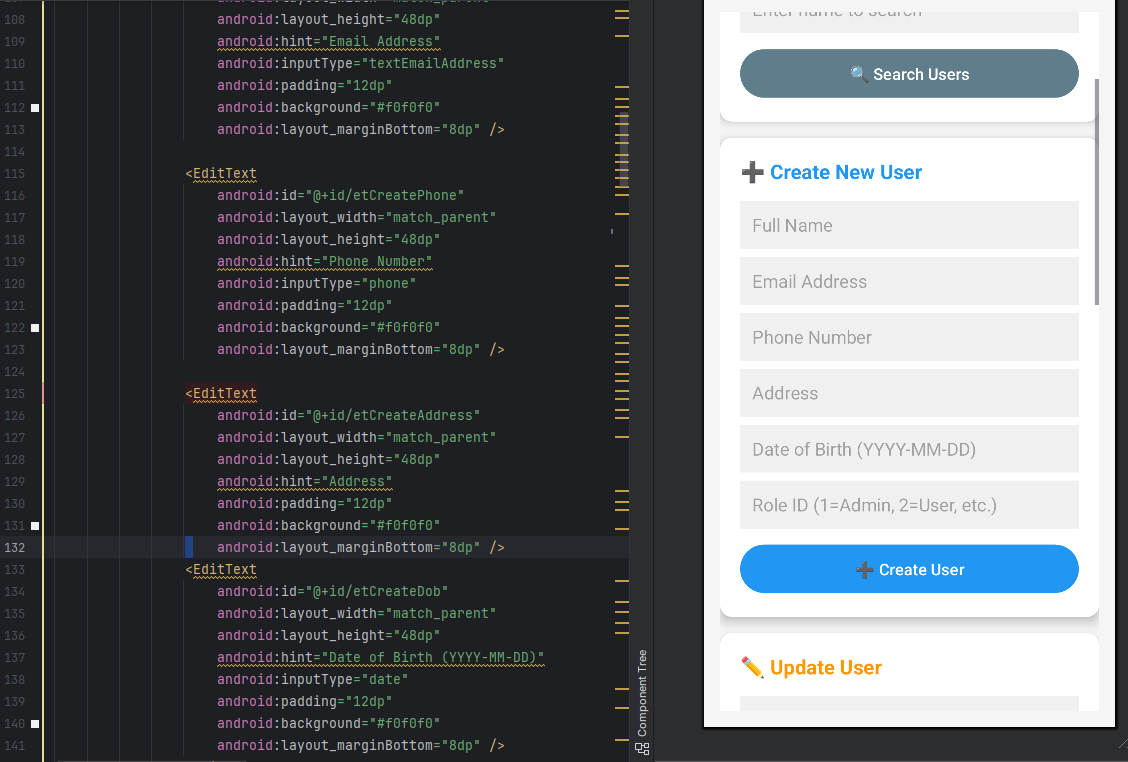
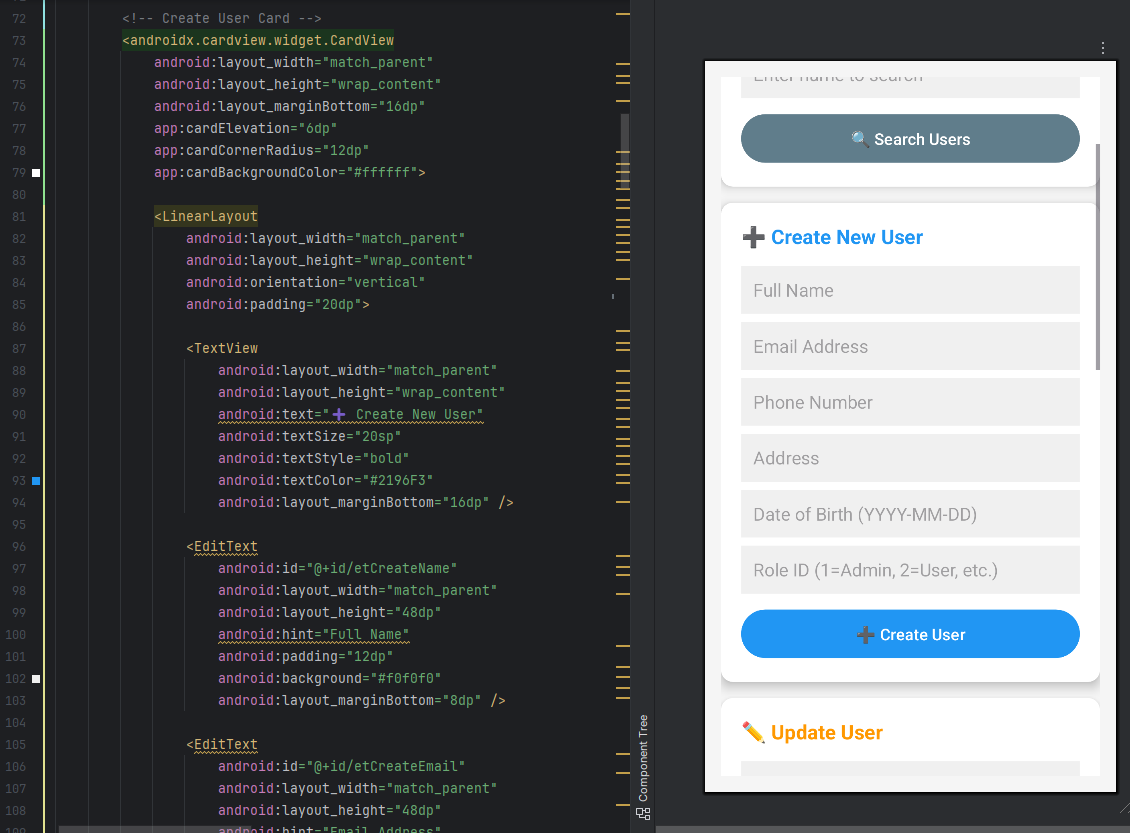
* Nhập tên và tìm kiếm người dùng từ cơ sở dữ liệu.





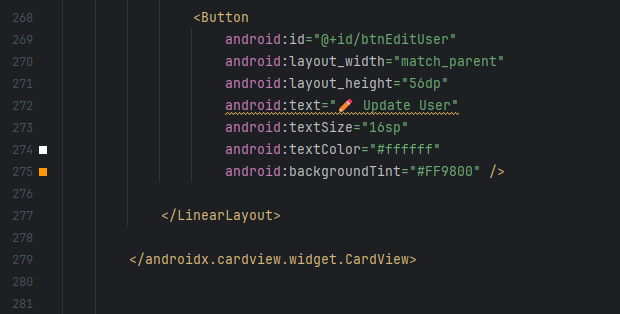
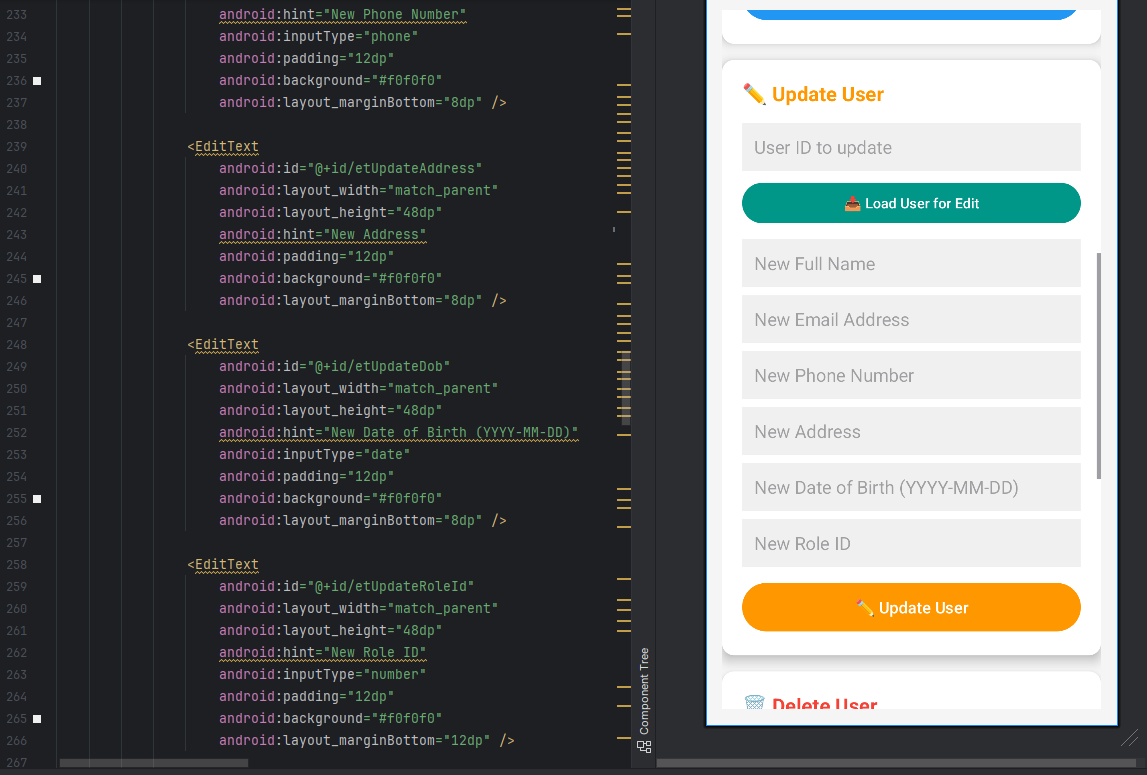
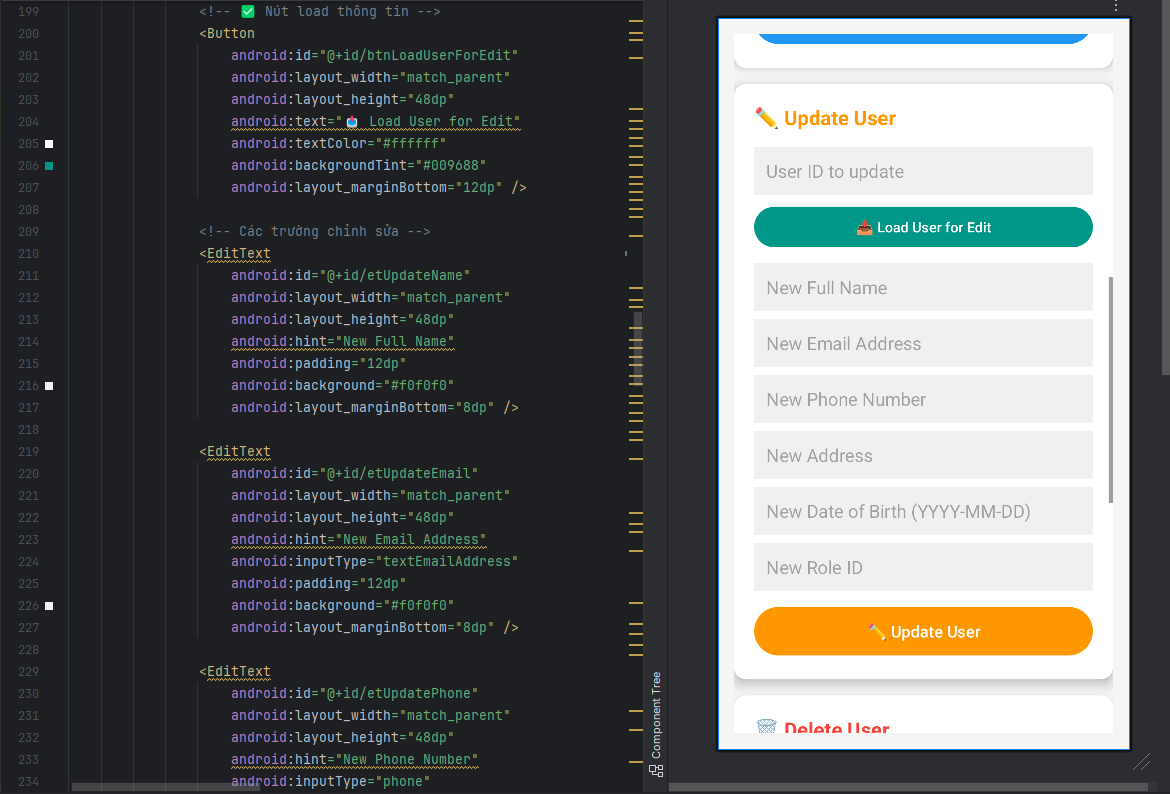
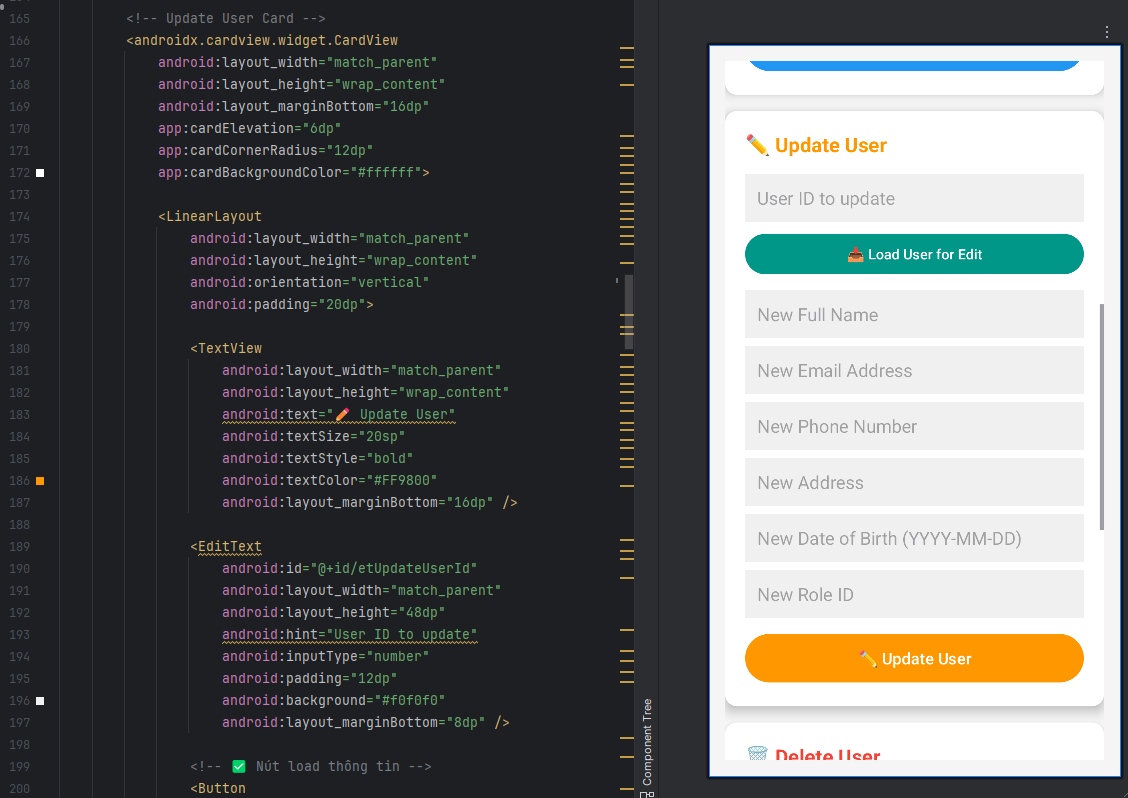
#### b) Create New User

* Các trường:  
  + Full Name
  + Email
  + Phone
  + Address
  + Date of Birth (YYYY-MM-DD)
  + Role ID
* Nút Create User: Gửi dữ liệu lên server để thêm người dùng mới.



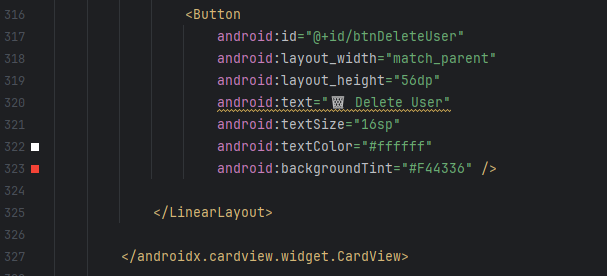
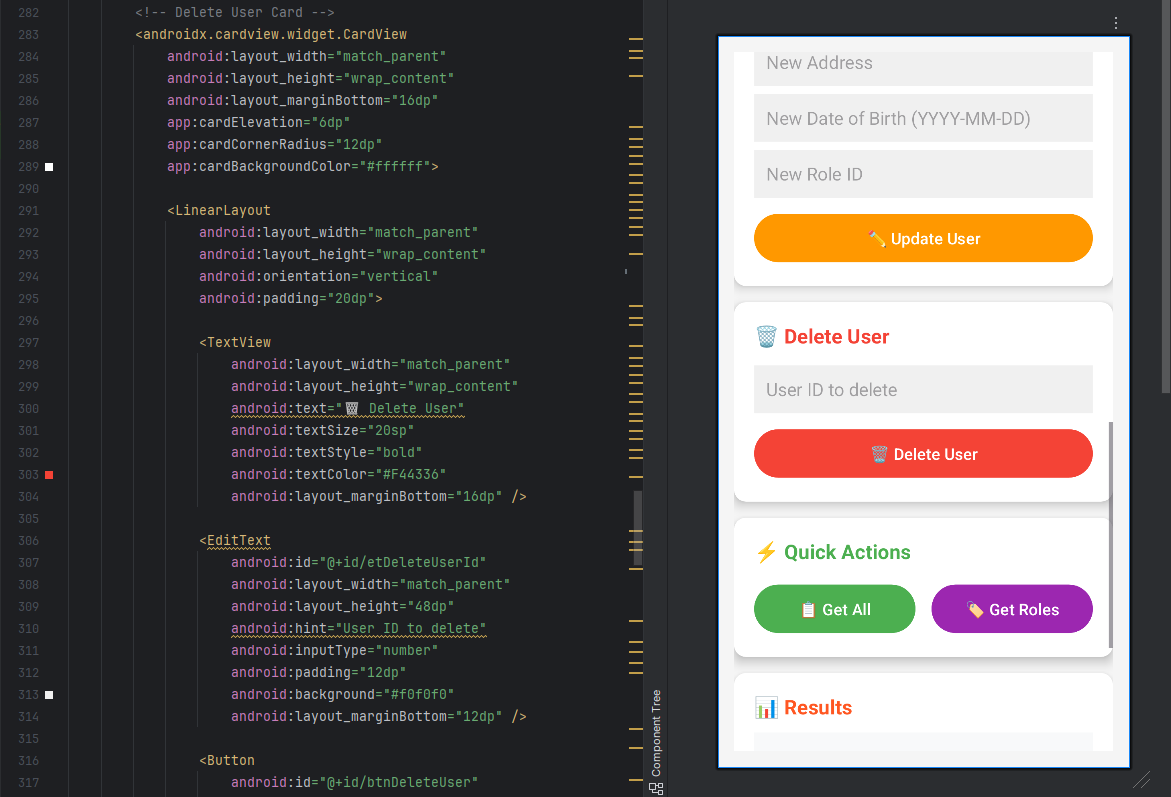
#### c) Update User

* Nhập User ID để tải dữ liệu.
* Nhập các trường mới nếu muốn cập nhật từng phần (các trường có thể để trống).
* Nút Load for Edit: Tải dữ liệu người dùng vào các ô nhập.
* Nút Update User: Gửi các trường được chỉnh sửa lên server để cập nhật.



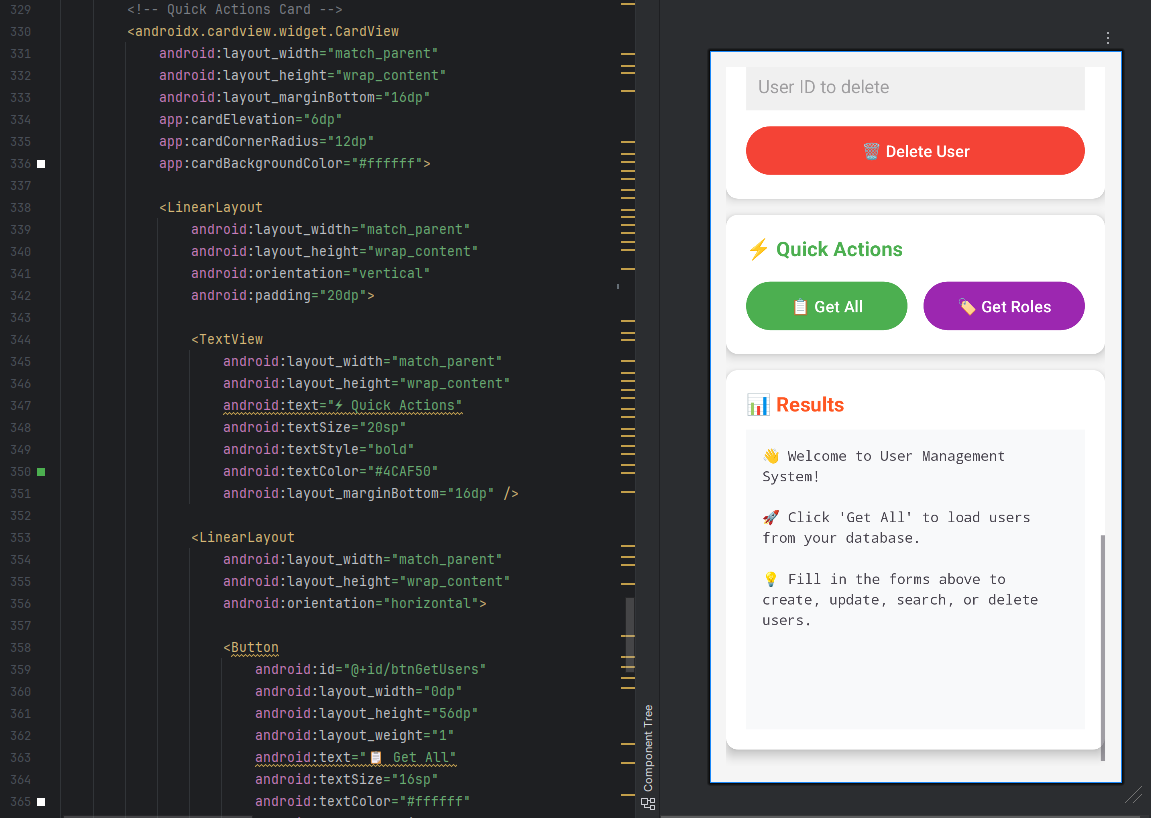
#### d) Delete User

* Nhập User ID và bấm Delete User, hệ thống sẽ hiển thị xác nhận trước khi xóa.



#### e) Quick Actions

* Get All: Tải lại toàn bộ danh sách người dùng.
* Get Roles: Hiển thị danh sách các vai trò.



#### e) Results

* Khu vực hiển thị kết quả tương tác: danh sách người dùng, kết quả tạo/sửa/xóa...

# 

# 

# **4. AndroidManifest.xml**



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

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

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

<!-- Permissions -->

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<application

android:usesCleartextTraffic="true"

android:allowBackup="true"

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

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

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

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

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

android:supportsRtl="true"

android:theme="@style/Theme.PRM292\_SqlServer"

tools:targetApi="31">

<activity

android:name=".MainActivity"

android:exported="true">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

# **5. Sản phẩm hoàn thiện Chức năng chính**

Ứng dụng Android quản lý người dùng với:

* Tạo người dùng mới
* Tìm kiếm theo tên
* Cập nhật từng trường riêng lẻ
* Xóa người dùng có xác nhận
* Hiển thị toàn bộ vai trò

Công nghệ sử dụng:

* Android Studio (Java)
* Retrofit2 – thư viện gọi API RESTful
* XML Layout – thiết kế giao diện theo Material Design
* ScrollView, CardView, LinearLayout – cấu trúc bố cục giao diện
* C# ASP.NET Web API – xây dựng hệ thống backend cung cấp dữ liệu JSON
* Localhost API hoặc Remote Server – triển khai API nội bộ hoặc qua IP cố định

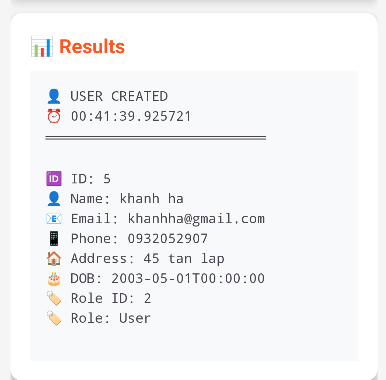
# 

* Khi search

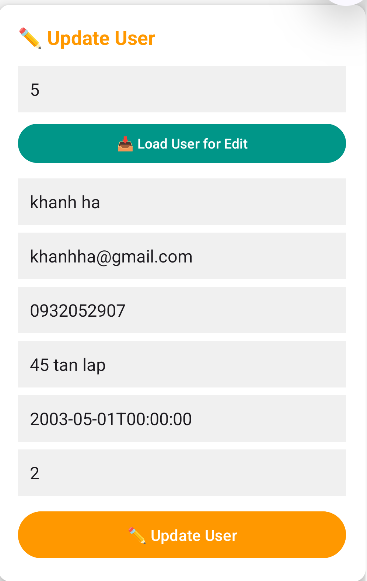
# 

* Khi thêm thông tin người dùng

# 



* Khi cập nhât thông tin người dùng



# 

* Khi thêm thông tin người dùng

# 

# Khi bấm coi danh sách thông tin

# 