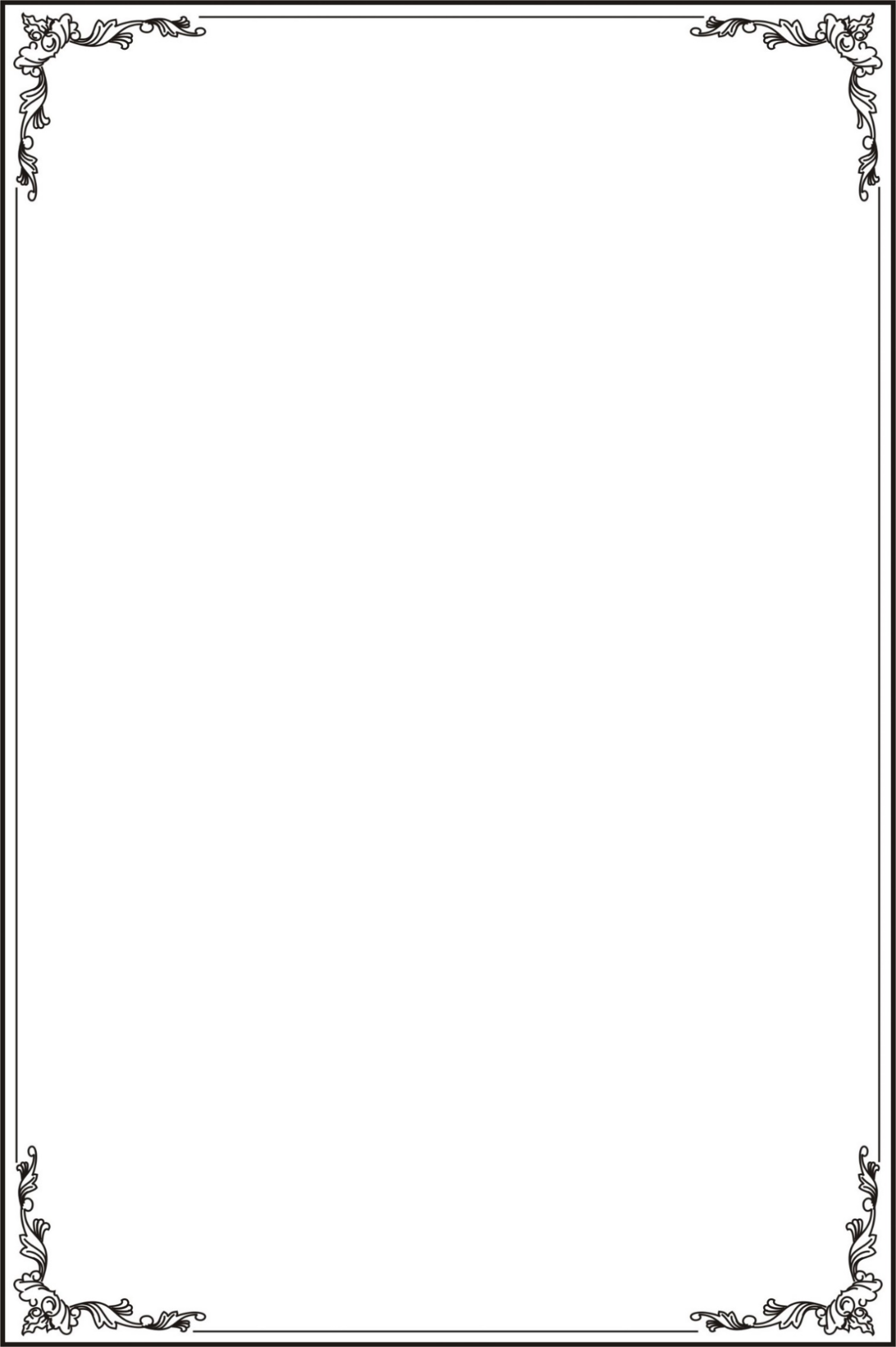
**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

🙞🕮🙜



**BÁO CÁO**

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG DI ĐỘNG**

**LÀM TRẮC NGHIỆM**

**GVHD**: ThS. Nguyễn Hữu Trung

**SVTH:** Nhóm 23

1. Nguyễn Văn An 22162002
2. Đỗ Phương Thùy 22162045
3. Phạm Nguyễn Minh Trí 22162050
4. Nguyễn Thành Được 22162011

**Môn học**: Lập trình di động

**Mã lớp học:** MOPR331279\_23\_2\_02

**Thành phố Hồ Chí Minh, Tháng 5 năm 20**

**MỤC LỤC**

[CHƯƠNG 1. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG](#_Toc167107931)

[1.1. Biểu đồ mô tả hệ thống](#_Toc167107932)

[1.2. Sơ đồ usecase](#_Toc167107933)

[1.2.1. Biểu đồ usecase tổng quát](#_Toc167107934)

[1.2.2. Đặc tả use case Đăng nhập](#_Toc167107935)

[1.2.3. Đặc tả usecase đăng ký](#_Toc167107936)

[1.2.4. Đặc tả usecase thi trắc nghiệm](#_Toc167107937)

[1.3. Sơ đồ hoạt động](#_Toc167107938)

[1.3.1. Sơ đồ hoạt động đăng nhập](#_Toc167107939)

[1.3.2. Sơ đồ hoạt động đăng ký](#_Toc167107940)

[1.3.3. Sơ đồ hoạt động thi trắc nghiệm](#_Toc167107941)

[1.4. Sơ đồ tuần tự](#_Toc167107942)

[1.4.1. Sơ đồ tuần tự đăng nhập](#_Toc167107943)

[1.4.2. Sơ đồ tuần tự đăng ký](#_Toc167107944)

[1.4.3. Sơ đồ tuần tự trả lời câu hỏi](#_Toc167107945)

[1.5. Thiết kế cơ sở dữ liệu](#_Toc167107946)

[1.5.1. Bảng User](#_Toc167107947)

[1.5.3. Bảng Câu hỏi](#_Toc167107948)

[CHƯƠNG 2. XÂY DỰNG GIAO DIỆN CHO ỨNG DỤNG](#_Toc167107949)

[2.1 Giao diện đăng nhập](#_Toc167107950)

[2.2 Giao diện đăng ký](#_Toc167107951)

[2.3 Giao diện chọn avatar khi đăng nhập](#_Toc167107952)

[2.4 Giao diện khi quên mật khẩu và lấy lại mật khẩu](#_Toc167107953)

[2.5 Giao diện trang chủ](#_Toc167107954)

[2.6 Giao diện chọn chủ đề](#_Toc167107955)

[2.7 Giao diện bắt đầu làm trắc nghiệm](#_Toc167107956)

[2.8 Giao diện xem kết quả khi đã nộp](#_Toc167107957)

**CHƯƠNG 3. XÂY DỰNG CHỨC NĂNG CHO ỨNG DỤNG**

**3.1. Code**

**3.2 API**

**CHƯƠNG 1. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

*Chương 1: trình bày về các biểu đồ use case, đặc tả, các sơ đồ hoạt động, tuần tự và thiết kế cơ sở dữ liệu cho đề tài*

**1.1. Biểu đồ mô tả hệ thống**

Ảnh có chứa văn bản, biểu đồ, ảnh chụp màn hình, Kế hoạch

Mô tả được tạo tự động

* + - **Hệ thống**
      * **Người dùng:**
* **Đăng ký:** người dùng nhập thông tin cá nhân và đăng ký tài khoản.
* **Đăng nhập/ xóa tài khoản:** người dùng đăng nhập vào game với tài khoản đã tạo hoặc có thể xóa tài khoản cá nhân.
* **Chọn chủ đề:** người dùng chọn chủ đề câu hỏi theo các tiêu chí mà người quản trị thiết lập.
* **Trả lời câu hỏi:** người dùng trả lời câu hỏi tương ứng với chủ đề trong một khoản thời gian nhất định.
* **Xem điểm, kết quả:** người dùng hoàn thành 10 câu hỏi có thể xem điểm, thưởng (nếu có).
* **Xem bảng xếp hạng:** người dùng xem bảng xếp hạng điểm số so với người chơi khác
  + - * **Người quản trị:**
* **Quản lý chủ đề:** người quản trị được quyền quản lý chủ đề của các câu đố với khả năng cập nhật, thêm, xóa, sửa các danh mục.
* **Quản lý câu hỏi:** người quản trị được quyền quản lý các câu đố tương ứng từng chủ đề với khả năng cập nhật, thêm, xóa, sửa các danh mục.
* **Quản lý bảng xếp hạng:** người quản trị được quyền quản lý bảng xếp hạng người chơi với khả năng cập nhật, thêm, xóa, sửa các danh mục.

**1.2. Sơ đồ usecase**

***1.2.1. Biểu đồ usecase tổng quát***

***Ảnh có chứa hình vẽ, biểu đồ, bản phác thảo, Nghệ thuật vẽ nét đơn

Mô tả được tạo tự động***

***Danh sách tác nhân mô tả***

|  |  |
| --- | --- |
| **Tác nhân** | **Mô tả tác nhân** |
| Người chơi | Người chơi có thể sử dụng các chức năng có sẵn trong  ứng dụng. |
| Người quản trị | Người quản trị có thể cập nhật: thêm, xóa, sửa các dữ liệu. |

### *1.2.2. Đặc tả use case Đăng nhập*

|  |  |
| --- | --- |
| **Use Case – Đăng Nhập** | |
| Định nghĩa | Người dùng đăng nhập vào hệ thống |
| Actors | Người dùng |
| Điều kiện tiên quyết | Người dùng đã có tài khoản. |
| **Dòng sự kiện** | |
| Dòng sự kiện chính | 1. Người dùng vào trang đăng nhập tài khoản  2. Người dùng nhập thông tin (tên đăng nhập, mật khẩu) nếu khớp thông tin, đăng nhập thành công  3. Trang chủ của ứng dụng hiện ra |
| Dòng sự kiện thay thế | Hiển thị thông báo thông tin không hợp lệ |
| Điều kiện sau | Sai mật khẩu:  Hệ thống kiểm tra không khớp thông tin tài khoản.  Hệ thống thông báo đăng nhập thất bại.  Quay lại bước 2 dòng sự kiện chính. |

***Biểu đồ usecase đăng nhập***

***Ảnh có chứa biểu đồ, vòng tròn, hàng, văn bản

Mô tả được tạo tự động***

***1.2.3. Đặc tả usecase đăng ký***

|  |  |
| --- | --- |
| **Use Case – Đăng Ký** | |
| Định nghĩa | Người dùng đăng ký tài khoản thành viên |
| Actors | Người dùng |
| Điều kiện tiên quyết | Người dùng chưa tạo tài khoản |
| **Dòng sự kiện** | |
| Dòng sự kiện chính | 1. Người dùng vào trang đăng ký tài khoản  2. Người dùng nhập thông tin (tên đăng nhập, mật khẩu, số điện thoại, avata) và nhấn nút đăng ký  3. Hệ thống kết nối Database và cập nhật lại dữ liệu  4. Trang đăng nhập hiện ra |
| Dòng sự kiện thay thế | Hiển thị thông báo thông tin nhập liệu không hợp lệ |
| Điều kiện sau | 1. Sai yêu cầu về trường thông tin:    1. Hệ thống kiểm tra trường thông tin không đúng yêu cầu về ký tự.    2. Hệ thống thông báo không hợp lệ.    3. Quay lại bước 2 dòng sự kiện chính.   2. Lỗi kết nối API  2.1. Hệ thống gặp lỗi từ máy chủ  2.2. Hệ thống thông báo lỗi kết nối máy chủ |

***Biểu đồ usecase đăng ký***

***Ảnh có chứa biểu đồ, vòng tròn, bản phác thảo

Mô tả được tạo tự động***

***1.2.4. Đặc tả usecase thi trắc nghiệm***

|  |  |
| --- | --- |
| **Use Case – Quiz** | |
| Định nghĩa | Trả lời câu hỏi trên ứng dụng |
| Actors | Người dùng |
| Điều kiện tiên quyết | Người dùng truy cập vào bộ câu hỏi trắc nghiệm xếp theo chủ đề |
| **Dòng sự kiện** | |
| Dòng sự kiện chính | 1. Người dùng đăng nhập vào hệ thống 2. Người dùng chọn chủ đề câu hỏi 3. Các câu hỏi sẽ được hiện thị. 4. Người dùng chọn 1 trong 4 phương án, đáp án hiện ra, hệ thống chuyển đến câu tiếp theo 5. Hoàn thành hết các câu hỏi, hệ thống chuyển sang màn hình kết quả |
| Dòng sự kiện thay thế | Không |
| Điều kiện sau |  |

***Biểu đồ usecase thi trắc nghiệm***

***Ảnh có chứa biểu đồ, văn bản, vòng tròn, hàng

Mô tả được tạo tự động***

**1.3. Sơ đồ hoạt động**

***1.3.1. Sơ đồ hoạt động đăng nhập***

***Ảnh có chứa văn bản, biểu đồ, ảnh chụp màn hình, hàng

Mô tả được tạo tự động***

***1.3.2. Sơ đồ hoạt động đăng ký***

***Ảnh có chứa văn bản, biểu đồ, hàng, Song song

Mô tả được tạo tự động***

***1.3.3. Sơ đồ hoạt động thi trắc nghiệm***

***Ảnh có chứa văn bản, biểu đồ, hàng, Kế hoạch

Mô tả được tạo tự động***

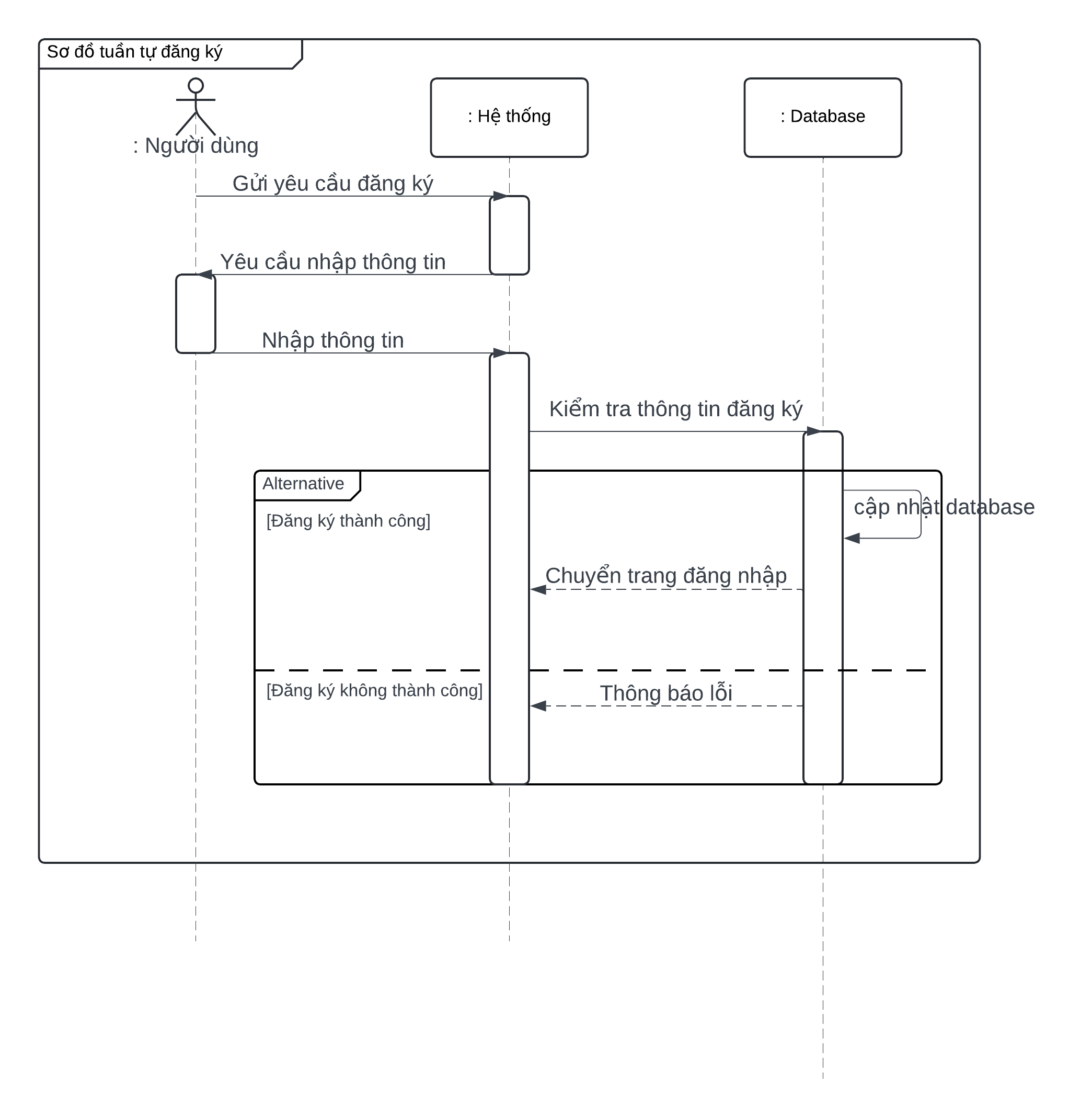
**1.4. Sơ đồ tuần tự**

***1.4.1. Sơ đồ tuần tự đăng nhập***

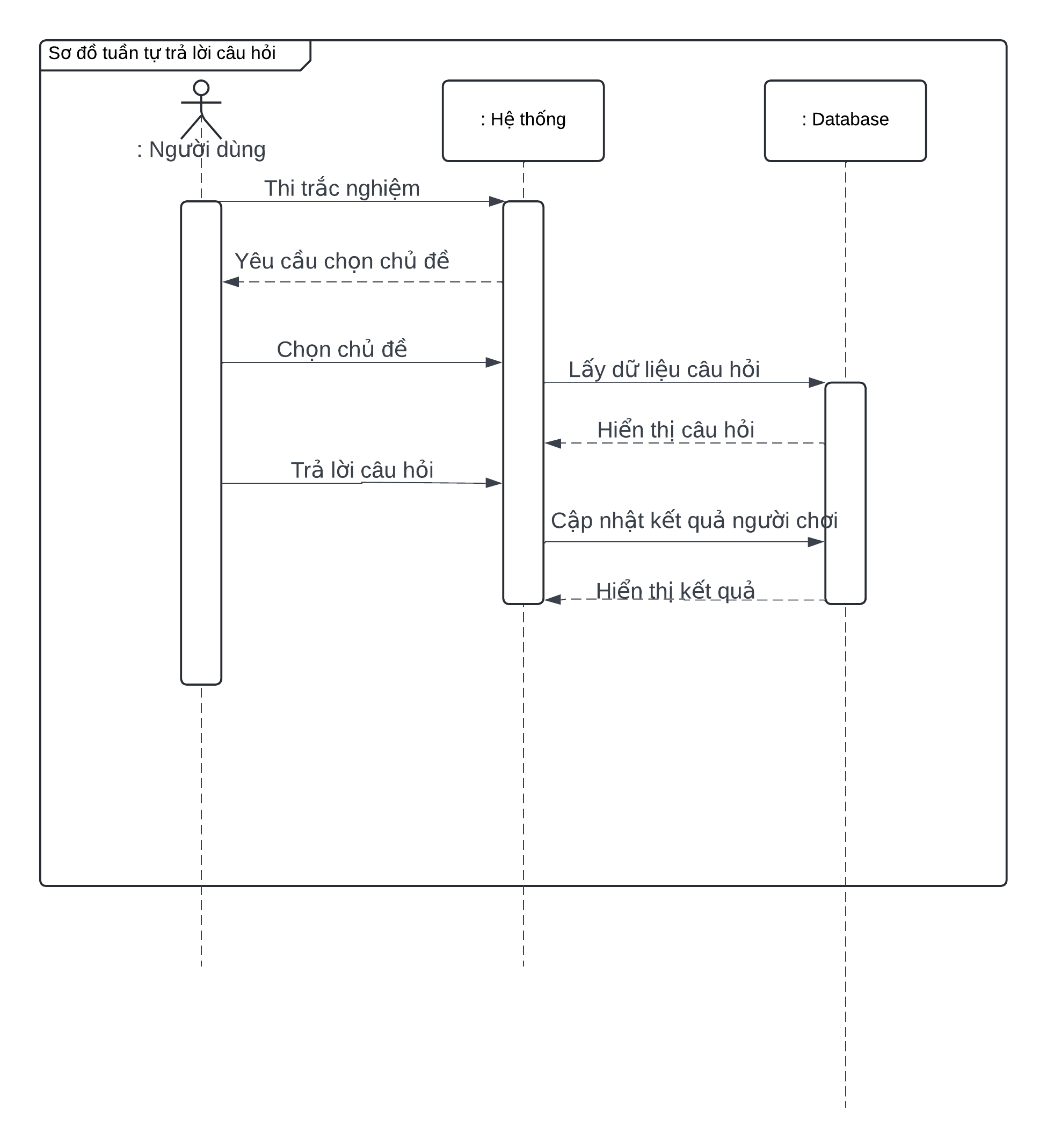
***Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Phông chữ

Mô tả được tạo tự động***

***1.4.2. Sơ đồ tuần tự đăng ký***

******

***1.4.3. Sơ đồ tuần tự trả lời câu hỏi***

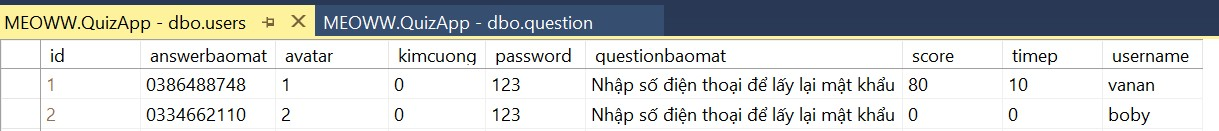
******

## 1.5. Thiết kế cơ sở dữ liệu

Trong phần này, chúng ta sẽ thiết kế cơ sở dữ liệu cho ứng dụng Quiz và triển khai nó bằng Java Spring Tool Suite, đồng thời sử dụng SQL Server để lưu trữ các bảng dữ liệu.

Cơ sở dữ liệu được thiết kế làm 2 bảng: bảng User và bảng Câu hỏi

***1.5.1. Bảng User***

******

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên thuộc tính** | **Kiểu dữ liệu** | **Extra** | **Chức năng** |
| id | int | AUTO\_INCREMENT | Mã số ID của người dùng |
| answerbaomat | string |  | Số điện thoại để lấy lại mật khẩu |
| questionbaomat | string |  | Yêu cầu để lấy lại mật khẩu |
| avatar | int |  | ID ảnh người dùng |
| kimcuong | int |  | Điểm thưởng |
| username | string |  | Tên đăng nhập |
| password | string |  | Mật khẩu của người dùng |
| score | int |  | Điểm số |
| timep | int |  | Số câu trả lời |

***1.5.3. Bảng Câu hỏi***

***Ảnh có chứa văn bản, ảnh chụp màn hình, số, tài liệu

Mô tả được tạo tự động***

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên thuộc tính** | **Kiểu dữ liệu** | **Extra** | **Chức năng** |
| id | int | AUTO\_INCREMENT | Mã số ID của câu hỏi |
| a, b, c, d | nvarchar |  | Đáp án |
| answer | nvarchar |  | Câu trả lời đúng |
| difficulty | nvarchar |  | Phân loại độ khó |
| ques | nvarchar |  | Nội dung câu hỏi |
| type | nvarchar |  | Chủ đề câu hỏi |

**CHƯƠNG 2. XÂY DỰNG ỨNG DỤNG DI ĐỘNG**

*Chương 2 trình bày về giao diện cũng như chức năng của ứng dụng di động*

**2.1 Giao diện đăng nhập**

Người dùng nhập username và mật khẩu để đăng nhập vào hệ thống khi đã đăng ký tài khoản.

Ảnh có chứa văn bản, ảnh chụp màn hình, thiết kế đồ họa, thiết kế

Mô tả được tạo tự động

**2.2 Giao diện đăng ký**

Nếu người dùng chưa có tài khoản, thì đăng ký mới tài khoản ở trang này.

Ảnh có chứa văn bản, ảnh chụp màn hình, phim hoạt hình, thiết kế đồ họa

Mô tả được tạo tự động

**2.3 Giao diện chọn avatar khi đăng nhập**

Chọn avatar để bắt đầu đăng nhập vào app.

Ảnh có chứa ảnh chụp màn hình, phim hoạt hình, minh họa, khung ảnh

Mô tả được tạo tự động

**2.4 Giao diện khi quên mật khẩu và lấy lại mật khẩu**

Nhập thông tin tài khoản để lấy lại mật khẩu.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

**2.5 Giao diện trang chủ**

Sau khi đăng nhập thành công sẽ đưa người dùng đến trang chính của ứng dụng và tương tác với các tùy chọn.

Ảnh có chứa văn bản, ảnh chụp màn hình, đa phương tiện

Mô tả được tạo tự động

**2.6 Giao diện chọn chủ đề**

Sau khi nhấp vào “Chơi ngay” sẽ đưa người dùng đến màn hình chủ đề câu hỏi của ứng dụng.

Ảnh có chứa văn bản, ảnh chụp màn hình, Hệ điều hành

Mô tả được tạo tự động

**2.7 Giao diện bắt đầu làm trắc nghiệm**

Hiễn thị các thông tin gồm username, số thứ tự câu hỏi, câu hỏi, các tùy chọn đáp án và thời gian thi.

Ảnh có chứa văn bản, ảnh chụp màn hình, Hệ điều hành, phần mềm

Mô tả được tạo tự động

**2.8 Giao diện xem kết quả khi đã nộp**

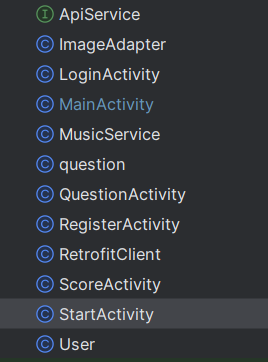
Ảnh có chứa văn bản, ảnh chụp màn hình, Hệ điều hành, phần mềm

Mô tả được tạo tự động

**CHƯƠNG 3. XÂY DỰNG CHỨC NĂNG CHO ỨNG DỤNG**

*Chương 3 trình bày về code và api của ứng dụng di động*

**3.1. Code : gồm những class sau :**



**3.1a . MainActivity :**

package com.example.quizzapp;

import android.app.Dialog;

import android.content.Intent;

import android.media.MediaPlayer;

import android.net.Uri;

import android.os.Bundle;

import android.view.View;

import android.view.ViewGroup;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.GridView;

import android.widget.ImageView;

import android.widget.ListView;

import android.widget.TextView;

import android.widget.Toast;

import android.widget.VideoView;

import androidx.annotation.Nullable;

import androidx.appcompat.app.AppCompatActivity;

import java.text.DecimalFormat;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

import retrofit2.Call;

import retrofit2.Callback;

import retrofit2.Response;

public class MainActivity extends AppCompatActivity {

private List<User> lu, luCopy;

public boolean isMusicPlaying = true;

private TextView u, sokc;

private User x;

private Button choingay, thongtincanhan, xephang, setting, cuahang;

Dialog dialog;

private List<String> luNew;

List<Integer> lavatar = new ArrayList<>();

List<Integer> lchude = new ArrayList<>();

private Intent musicServiceIntent;

private ViewGroup layout;

private int bg;

private int avatar;

private ImageView ava;

public int z ;

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

layout = findViewById(R.id.ques);

{

lavatar.add(R.drawable.img\_1);

lavatar.add(R.drawable.img\_2);

lavatar.add(R.drawable.img\_3);

lavatar.add(R.drawable.img\_4);

lavatar.add(R.drawable.img\_5);

lavatar.add(R.drawable.img\_6);

lavatar.add(R.drawable.img\_7);

lavatar.add(R.drawable.img\_8);

lavatar.add(R.drawable.img\_9);

lavatar.add(R.drawable.img\_10);

}

musicServiceIntent = new Intent(MainActivity.this, MusicService.class);

{

lchude.add(R.drawable.t0);

lchude.add(R.drawable.t1);

lchude.add(R.drawable.t2);

lchude.add(R.drawable.t3);

// Thêm các hình ảnh khác vào danh sách

}

lu = new ArrayList<>();

z = getIntent().getIntExtra("chude",z);

setMainActivityBackground(z);

getListUsers();bg=0;

xephang = findViewById(R.id.button3);

u = findViewById(R.id.textView5);

sokc = findViewById(R.id.textView8);

choingay = findViewById(R.id.button1);

thongtincanhan = findViewById(R.id.button2);

cuahang = findViewById(R.id.button4);

cuahang.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

ch();

}

});

thongtincanhan.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

hientt();

}

});

x = getIntent().getParcelableExtra("tk");

u.setText(x.getUsername());

sokc.setText(String.valueOf(x.getKimcuong()));

avatar = x.getAvatar();

ava = findViewById(R.id.image\_logo);

ava.setImageResource(avatar);

choingay.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent tostart = new Intent(MainActivity.this, StartActivity.class);

tostart.putExtra("tk", x);

tostart.putExtra("cd",z);

tostart.putExtra("bg",bg);

startActivity(tostart);

}

});

xephang.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

xh();

}

});

setting = findViewById(R.id.button5);

setting.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

setting();

}

});

Button dangxuat = findViewById(R.id.button6);

dangxuat.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent backlogin = new Intent(MainActivity.this, LoginActivity.class);

startActivity(backlogin);

}

});

}

//end

private void getListUsers() {

ApiService apiService = RetrofitClient.getClient().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) {

lu = response.body();

// Gọi phương thức để xử lý dữ liệu người dùng sau khi lấy dữ liệu từ API

handleUserData();

}

@Override

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

Toast.makeText(MainActivity.this, "Network error", Toast.LENGTH\_SHORT).show();

}

});

}

// Thêm phương thức này để sao chép và sắp xếp danh sách người dùng sau khi lấy dữ liệu từ API

private void handleUserData() {

luCopy = new ArrayList<>(lu);

Collections.sort(luCopy, new Comparator<User>() {

@Override

public int compare(User user1, User user2) {

return user2.getScore() - user1.getScore();

}

});

luNew = new ArrayList<>();

int i = 0;

int t =0;

int k=0;

for (User q : luCopy) {

if(i==0){

luNew.add("Hạng "+(i+1)+" "+q.getUsername()+" điểm: "+ q.getScore());

t=i+1;

} else {

if(q.getScore() == k){

luNew.add("Hạng "+t+" "+q.getUsername()+" điểm: "+ q.getScore());

} else {

luNew.add("Hạng "+(i+1)+" "+q.getUsername()+" điểm: "+ q.getScore());

t=i+1;

}

}

k = q.getScore();

i++;

if (i == 49) {

break;

}

}

}

private void hientt() {

dialog = new Dialog(MainActivity.this);

dialog.setContentView(R.layout.activity\_profile);

dialog.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.75));

TextView t1, t2, t3, t4, t5, t6;

t1 = dialog.findViewById(R.id.user\_label);

t2 = dialog.findViewById(R.id.tv16);

t3 = dialog.findViewById(R.id.tv2);

t4 = dialog.findViewById(R.id.tv3);

t5 = dialog.findViewById(R.id.tv4);

t6 = dialog.findViewById(R.id.tv1);

t1.setText("Người chơi " + x.getUsername());

t2.setText("Số câu đã trả lời " + x.getTimep());

t3.setText("Với số câu đúng " + x.getScore() / 10);

t4.setText("Số câu trả lời sai " + (x.getTimep() - x.getScore() / 10));

if (x.getTimep() != 0) {

DecimalFormat df = new DecimalFormat("#.##"); // Số # biểu thị cho một chữ số có thể có hoặc không.

double rounded = Double.parseDouble(df.format((x.getScore() \*10) / x.getTimep())); // rounded = 5.68

t5.setText("Tỉ lệ trả lời đúng " + rounded + "%");

} else {

t5.setText("Tỉ lệ trả lời đúng là 0");

}

t6.setText("Tổng số điểm " + x.getScore());

dialog.show();

Button btb = dialog.findViewById(R.id.return\_button);

btb.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dialog.dismiss();

}

});

}

private void xh() {

dialog = new Dialog(MainActivity.this);

dialog.setContentView(R.layout.activity\_ranking);

dialog.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.75));

ListView lv;

lv = dialog.findViewById(R.id.leaderboardListView);

// Tạo CustomAdapter

ArrayAdapter adapter = new ArrayAdapter(MainActivity.this, R.layout.dapan, luNew);

// Thiết lập CustomAdapter cho ListView

lv.setAdapter(adapter);

dialog.show();

Button backmain = dialog.findViewById(R.id.backToMainButton);

backmain.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dialog.dismiss();

}

});

}

private void setting(){

dialog = new Dialog(MainActivity.this);

dialog.setContentView(R.layout.setting);

dialog.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.6));

Button b1,b2,b3,b4,b5;

b1= dialog.findViewById(R.id.buttonSound);

if(MusicService.isRunning()){

b1.setText("Đang bật nhạc");

} else {

b1.setText("Nhạc đã tắt");

}

b2= dialog.findViewById(R.id.buttonTheme);

b3= dialog.findViewById(R.id.buttonAvatar);

b4= dialog.findViewById(R.id.buttonSnow);

b5= dialog.findViewById(R.id.buttonDeleteAccount);

b1.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if(MusicService.isRunning()){

b1.setText("Nhạc đã tắt");

isMusicPlaying = false;

stopService(musicServiceIntent);

} else {

b1.setText("Đang bật nhạc");

startService(musicServiceIntent);

}

}

});

b2.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

chont();

}

});

dialog.show();

b4.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dialog.dismiss();

}

});

b3.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

chonlogo();

}

});

b5.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Dialog dialog2 = new Dialog(MainActivity.this);

dialog2.setContentView(R.layout.bottom\_sheet\_layout);

dialog2.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.4));

GridView gr = dialog2.findViewById(R.id.gridViewAvatars);

List<String> ln = new ArrayList<>();

ln.add("Chưa");

ln.add("Xóa");

ArrayAdapter aaa = new ArrayAdapter(MainActivity.this, R.layout.dapan,ln);

gr.setAdapter(aaa);

dialog2.show();

gr.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

String S = aaa.getItem(position).toString();

if(S.equals("Chưa")){

dialog2.dismiss();

} else {

del();

Intent tologin = new Intent(MainActivity.this, LoginActivity.class);

startActivity(tologin);

}

}

});

}

});

}

private void chonlogo(){

Dialog dialog1 = new Dialog(MainActivity.this);

dialog1.setContentView(R.layout.bottom\_sheet\_layout);

dialog1.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.75));

// Lấy GridView từ layout

GridView gridView = dialog1.findViewById(R.id.gridViewAvatars);

// Tạo danh sách hình ảnh

List<Integer> imageList = new ArrayList<>(lavatar);

// Tạo adapter cho GridView

ImageAdapter adapter = new ImageAdapter(imageList);

// Đặt adapter cho GridView

gridView.setAdapter(adapter);

// Xử lý sự kiện khi người dùng chọn một hình ảnh từ GridView

gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

// Xử lý khi người dùng chọn một hình ảnh từ GridView

int selectedImage = adapter.getItem(position);

// Thực hiện các hành động khác với hình ảnh được chọn

}

});

// Xử lý sự kiện khi người dùng nhấn nút "Đóng"

// Hiển thị dialog

dialog1.show();

gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

// Xử lý khi người dùng chọn một hình ảnh từ GridView

int selectedImage = adapter.getItem(position);

// Gán giá trị của selectedImage cho biến avatar

avatar = selectedImage;

// Thay đổi ImageView ở màn hình đăng nhập thành ảnh được chọn

ImageView loginAvatar = findViewById(R.id.image\_logo);

loginAvatar.setImageResource(avatar);

x.setAvatar(avatar);

updateUser();

// Đóng dialog sau khi chọn hình ảnh

dialog1.dismiss();

}

});

}

private void chont() {

Dialog ndialog = new Dialog(MainActivity.this);

ndialog.setContentView(R.layout.bottom\_sheet\_layout);

ndialog.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.75));

// Lấy GridView từ layout

GridView gridView = ndialog.findViewById(R.id.gridViewAvatars);

// Tạo danh sách hình ảnh

List<Integer> imageList = new ArrayList<>(lchude);

// Tạo adapter cho GridView

ImageAdapter adapter = new ImageAdapter(imageList);

// Đặt adapter cho GridView

gridView.setAdapter(adapter);

ndialog.show();

gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

// Set the clicked image as the background of MainActivity

int selectedImageResourceId = imageList.get(position);

setMainActivityBackground(selectedImageResourceId);

z = selectedImageResourceId;

// Dismiss the dialog after selecting an image

ndialog.dismiss();

}

});

}

private void setMainActivityBackground(int imageResourceId) {

findViewById(R.id.ques).setBackgroundResource(imageResourceId);

bg = imageResourceId;

}

private void updateUser() {

// Khởi tạo RetrofitClient và ApiService

ApiService apiService = RetrofitClient.getClient().create(ApiService.class);

// Tạo một yêu cầu PUT để cập nhật thông tin người dùng

Call<User> call = apiService.updateUser(x.getId(),x);

// Gửi yêu cầu PUT đến máy chủ

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

@Override

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

if (response.isSuccessful()) {

// Xử lý khi cập nhật thành công

User updatedUser = response.body();

Toast.makeText(MainActivity.this, "cập nhật avatar thành công", Toast.LENGTH\_SHORT).show();

} else {

// Xử lý khi gặp lỗi từ máy chủ

Toast.makeText(MainActivity.this, "Cập nhật avatar thất bại", Toast.LENGTH\_SHORT).show();

}

}

@Override

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

// Xử lý khi gặp lỗi kết nối hoặc lỗi khác

Toast.makeText(MainActivity.this, "Không thể kết nối đến máy chủ", Toast.LENGTH\_SHORT).show();

}

});

}

public void del(){

ApiService apiService = RetrofitClient.getClient().create(ApiService.class);

Call<User> call = apiService.deleteUser(x.getId());

// Gửi yêu cầu PUT đến máy chủ

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

@Override

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

if (response.isSuccessful()) {

// Xử lý khi cập nhật thành công

User updatedUser = response.body();

Toast.makeText(MainActivity.this, " Đã xóa tài khoản", Toast.LENGTH\_SHORT).show();

} else {

// Xử lý khi gặp lỗi từ máy chủ

Toast.makeText(MainActivity.this, "Không xóa được tài khoản", Toast.LENGTH\_SHORT).show();

}

}

@Override

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

// Xử lý khi gặp lỗi kết nối hoặc lỗi khác

Toast.makeText(MainActivity.this, "Không thể kết nối đến máy chủ", Toast.LENGTH\_SHORT).show();

}

});

}

public void ch(){

Dialog dc = new Dialog(MainActivity.this);

dc.setContentView(R.layout.shop);

dc.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.75));

Button s1,s2,s3,s4;

dc.show();

s1= dc.findViewById(R.id.button9);

s2= dc.findViewById(R.id.button10);

s3 = dc.findViewById(R.id.button11);

s4 = dc.findViewById(R.id.button12);

s1.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Toast.makeText(MainActivity.this, "Chưa hỗ trợ trên app ", Toast.LENGTH\_SHORT).show();

}

});

s2.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Toast.makeText(MainActivity.this, "Chưa hỗ trợ trên app ", Toast.LENGTH\_SHORT).show();

}

});

s3.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Toast.makeText(MainActivity.this, "Chưa hỗ trợ trên app ", Toast.LENGTH\_SHORT).show();

}

});

s4.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dc.dismiss();

}

});

}

}

**3.1b LoginActivity:**

package com.example.quizzapp;

import android.app.Dialog;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageButton;

import android.widget.TextView;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;

import java.util.List;

import retrofit2.Call;

import retrofit2.Callback;

import retrofit2.Response;

public class LoginActivity extends AppCompatActivity {

private EditText Name, Pass;

private Button bl;

private Dialog dialog;

private List<User> lu;

private TextView breg, lp;

String A,B;

private ImageButton avatar;

private User user;

private Intent musicServiceIntent;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

Name = findViewById(R.id.editText\_email);

Pass = findViewById(R.id.editText\_password);

bl = findViewById(R.id.button);

breg= findViewById(R.id.button\_re);

lp = findViewById(R.id.textView6);

musicServiceIntent = new Intent(LoginActivity.this, MusicService.class);

startService(musicServiceIntent);

lu = new ArrayList<>();

getListUsers();

bl.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

login();

}

});

breg.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent tosignup = new Intent(LoginActivity.this,RegisterActivity.class);

startActivity(tosignup);

}

});

lp.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

ser();

}

});

}

private User tkuser(List<User> lu,String Name){

User x = new User();

for(User u : lu){

if(u.getUsername().equals(Name)){

x=u;

break;

}

}

return x;

}

private void login() {

String namelogin = Name.getText().toString().trim();

String passlogin = Pass.getText().toString().trim();

boolean isLoggedIn = false;

for (User x : lu) {

if (x.getUsername().equals(namelogin) && x.getPassword().equals(passlogin)) {

isLoggedIn = true;

break;

}

}

if (isLoggedIn) {

Intent toapp = new Intent(LoginActivity.this, MainActivity.class);

User u = tkuser(lu, namelogin);

toapp.putExtra("tk", u);

int z =0;

toapp.putExtra("pn",z);

toapp.putExtra("chude",R.drawable.t0);

startActivity(toapp);

} else {

Toast.makeText(LoginActivity.this, "Incorrect username or password", Toast.LENGTH\_SHORT).show();

}

}

private void getListUsers() {

ApiService apiService = RetrofitClient.getClient().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) {

lu = response.body();

}

@Override

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

Toast.makeText(LoginActivity.this, "Network error", Toast.LENGTH\_SHORT).show();

}

});

}

private void ser(){

dialog = new Dialog(LoginActivity.this);

dialog.setContentView(R.layout.ser);

dialog.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.85));

Button b1,b2;

EditText a,b;

b1 = dialog.findViewById(R.id.button8);

b2 = dialog.findViewById(R.id.button7);

a = dialog.findViewById(R.id.editTextText3);

b = dialog.findViewById(R.id.textView13);

b2.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

A = a.getText().toString();

B = b.getText().toString();

if(A.isEmpty() || B.isEmpty()){

Toast.makeText(LoginActivity.this, "Chưa nhập tài khoản hoạc số điện thoại", Toast.LENGTH\_SHORT).show();

} else {

int k=0;

for(User i : lu){

if(i.getUsername().equals(A) && i.getAnswerbaomat().equals(B)){

a.setText("Đây là mật khẩu của bạn "+i.getPassword());

k=0;

break;

} else {

k = -1;

}

}

if(k==-1){

Toast.makeText(LoginActivity.this, "Nhập sai tài khoản hoặc số điện thoại", Toast.LENGTH\_SHORT).show();

}

}

}

});

b1.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dialog.dismiss();

}

});

dialog.show();

}

}

**3.1c RegisterActivity :**

package com.example.quizzapp;

import android.app.Dialog;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.view.ViewGroup;

import android.widget.AdapterView;

import android.widget.Button;

import android.widget.EditText;

import android.widget.GridView;

import android.widget.ImageButton;

import android.widget.ImageView;

import android.widget.TextView;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;

import java.util.List;

import retrofit2.Call;

import retrofit2.Callback;

import retrofit2.Response;

public class RegisterActivity extends AppCompatActivity {

private EditText Name, Pass, sdt;

private TextView bl;

private User user;

private Button su;

private int avatar;

private List<User> lu;

private int k;

private ImageView logo;

Dialog dialog;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_register);

Name = findViewById(R.id.editText\_email);

Pass = findViewById(R.id.editText\_password);

bl = findViewById(R.id.backlogin);

su = findViewById(R.id.button\_login);

logo= findViewById(R.id.image\_logo);

logo.setImageResource(R.drawable.img\_1);

sdt = findViewById(R.id.editTextText2);

lu = new ArrayList<>();

avatar = 1;

getListUsers();

k=0;

//dangki

logo.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

chonlogo();

}

});

su.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dangky(); return;

}

});

// sang login

bl.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent tologin = new Intent(RegisterActivity.this,LoginActivity.class);

startActivity(tologin);

}

});

}

private void chonlogo(){

dialog = new Dialog(RegisterActivity.this);

dialog.setContentView(R.layout.bottom\_sheet\_layout);

dialog.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 0.75));

// Lấy GridView từ layout

GridView gridView = dialog.findViewById(R.id.gridViewAvatars);

// Tạo danh sách hình ảnh

List<Integer> imageList = new ArrayList<>();

imageList.add(R.drawable.img\_1);

imageList.add(R.drawable.img\_2);

imageList.add(R.drawable.img\_3);

imageList.add(R.drawable.img\_4);

imageList.add(R.drawable.img\_5);

imageList.add(R.drawable.img\_6);

imageList.add(R.drawable.img\_7);

imageList.add(R.drawable.img\_8);

imageList.add(R.drawable.img\_9);

imageList.add(R.drawable.img\_10);

// Thêm các hình ảnh khác vào danh sách

// Tạo adapter cho GridView

ImageAdapter adapter = new ImageAdapter(imageList);

// Đặt adapter cho GridView

gridView.setAdapter(adapter);

// Xử lý sự kiện khi người dùng chọn một hình ảnh từ GridView

gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

// Xử lý khi người dùng chọn một hình ảnh từ GridView

int selectedImage = adapter.getItem(position);

// Thực hiện các hành động khác với hình ảnh được chọn

}

});

// Xử lý sự kiện khi người dùng nhấn nút "Đóng"

// Hiển thị dialog

dialog.show();

gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

// Xử lý khi người dùng chọn một hình ảnh từ GridView

int selectedImage = adapter.getItem(position);

// Gán giá trị của selectedImage cho biến avatar

avatar = selectedImage;

// Thay đổi ImageView ở màn hình đăng nhập thành ảnh được chọn

ImageView loginAvatar = findViewById(R.id.image\_logo);

loginAvatar.setImageResource(avatar);

// Đóng dialog sau khi chọn hình ảnh

dialog.dismiss();

}

});

}

private void getListUsers() {

ApiService apiService = RetrofitClient.getClient().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) {

lu = response.body();

}

@Override

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

Toast.makeText(RegisterActivity.this, "Network error", Toast.LENGTH\_SHORT).show();

}

});

}

private void dangky(){

String ten = Name.getText().toString();

String mk = Pass.getText().toString();

String sp = sdt.getText().toString();

if(sp.isEmpty()){

Toast.makeText(RegisterActivity.this, "Nhập số điện thoại", Toast.LENGTH\_SHORT).show();

return;

}

if(mk.isEmpty()){

Toast.makeText(RegisterActivity.this, "Nhập mật khẩu để đăng ký", Toast.LENGTH\_SHORT).show();

return;

}

if(ten.isEmpty()){

Toast.makeText(RegisterActivity.this, "Nhập tên để đăng ký", Toast.LENGTH\_SHORT).show();

return;

}

if(ten.length()<3){

Toast.makeText(RegisterActivity.this, "tên phải hơn 3 kí tự", Toast.LENGTH\_SHORT).show();

return;

}

if(mk.length()<3){

Toast.makeText(RegisterActivity.this, "mật khẩu phải hơn 3 kí tự", Toast.LENGTH\_SHORT).show();

return;

}

if(sp.length()<8){

Toast.makeText(RegisterActivity.this, "Sai định dạng số điện thoại", Toast.LENGTH\_SHORT).show();

return;

}

if(lu.stream().count()==0){

Toast.makeText(RegisterActivity.this, "Tài khoản đang được tạo", Toast.LENGTH\_SHORT).show();

user = new User(lu.stream().count()+1, Name.getText().toString(), Pass.getText().toString(),0,sdt.getText().toString(), "Nhập số điện thoại để lấy lại mật khẩu", 0,avatar,0);

updateUser();

Intent toapp = new Intent(RegisterActivity.this,LoginActivity.class);

toapp.putExtra("tk", user);

startActivity(toapp);

return;

}

for(User u: lu){

if(ten.equals(u.getUsername())){

Toast.makeText(RegisterActivity.this, "Tài khoản đã tồn tại", Toast.LENGTH\_SHORT).show();

return;

} else {

k=1;

}

}

if(k == 1){

Toast.makeText(RegisterActivity.this, "Tài khoản đang được tạo", Toast.LENGTH\_SHORT).show();

user = new User(lu.stream().count()+1, Name.getText().toString(), Pass.getText().toString(),0,sdt.getText().toString(), "Nhập số điện thoại để lấy lại mật khẩu", 0,avatar,0);

updateUser();

Intent toapp = new Intent(RegisterActivity.this,LoginActivity.class);

toapp.putExtra("tk", user);

startActivity(toapp);

}

}

private void updateUser() {

// Khởi tạo RetrofitClient và ApiService

ApiService apiService = RetrofitClient.getClient().create(ApiService.class);

// Tạo một yêu cầu PUT để cập nhật thông tin người dùng

Call<User> call = apiService.createUser(user);

// Gửi yêu cầu PUT đến máy chủ

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

@Override

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

if (response.isSuccessful()) {

// Xử lý khi cập nhật thành công

User updatedUser = response.body();

Toast.makeText(RegisterActivity.this, "Cập nhật thông tin người dùng thành công", Toast.LENGTH\_SHORT).show();

} else {

// Xử lý khi gặp lỗi từ máy chủ

Toast.makeText(RegisterActivity.this, "Cập nhật thông tin người dùng thất bại", Toast.LENGTH\_SHORT).show();

}

}

@Override

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

// Xử lý khi gặp lỗi kết nối hoặc lỗi khác

Toast.makeText(RegisterActivity.this, "Không thể kết nối đến máy chủ", Toast.LENGTH\_SHORT).show();

}

});

}

}

**3.1d QestionActivity**

package com.example.quizzapp;

import android.app.Dialog;

import android.content.Intent;

import android.os.Bundle;

import android.os.CountDownTimer;

import android.os.Handler;

import android.os.Parcelable;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.ListView;

import android.widget.TextView;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;

import java.util.List;

import java.util.Random;

import retrofit2.Call;

import retrofit2.Callback;

import retrofit2.Response;

public class QuestionActivity extends AppCompatActivity {

private List<question> lu,lu10,luchude;

private Random ran = new Random();

List<String> lis;

private String chude;

private List<question> filteredQuestions;

private ListView lv;

private Handler mHandler = new Handler();

private static final int DELAY\_TIME = 2000; // Thời gian hiển thị đáp án (2 giây)

private int currentQuestionIndex = 0,max,i;

private CountDownTimer countDownTimer;

private int sco;

private User u;

private TextView ten,kc;

private ImageView ava;

int y;

private int min ;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_question);

lis = new ArrayList<>();

lv = findViewById(R.id.listView);

if (lv == null) {

Toast.makeText(this, "ListView not found", Toast.LENGTH\_SHORT).show();

return;

}

Intent intent = getIntent();

if (intent == null) {

Toast.makeText(this, "Intent not found", Toast.LENGTH\_SHORT).show();

return;

}

u = intent.getParcelableExtra("tk");

if (u == null) {

Toast.makeText(this, "User data not found in intent", Toast.LENGTH\_SHORT).show();

return;

}

chude = intent.getStringExtra("chude");

if (chude == null) {

Toast.makeText(this, "Chude data not found in intent", Toast.LENGTH\_SHORT).show();

return;

}

getListQuestion(); // Call the method to fetch questions from the API

}

private void getListQuestion() {

ApiService apiService = RetrofitClient.getClient().create(ApiService.class);

Call<List<question>> call = apiService.getquestions();

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

@Override

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

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

lu = response.body();

updateListView();

} else {

Toast.makeText(QuestionActivity.this, "Failed to get questions from the API", Toast.LENGTH\_SHORT).show();

}

}

@Override

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

Toast.makeText(QuestionActivity.this, "Network error", Toast.LENGTH\_SHORT).show();

}

});

}

private void updateListView() {

if (lu == null) {

Toast.makeText(this, "No questions received from the API", Toast.LENGTH\_SHORT).show();

return;

}

filteredQuestions = new ArrayList<>();

for (question x : lu) {

if (x.getType().equals(chude)) {

filteredQuestions.add(x);

}

}

if (filteredQuestions.size() < 10) {

Toast.makeText(this, "Not enough questions for this topic", Toast.LENGTH\_SHORT).show();

} else {

// Display the first question

min = 0;

max = filteredQuestions.size() - 15;

int randomIndex = getRandomNumberInRange(max);

min = randomIndex;

displayQuestion(filteredQuestions.get(randomIndex));

lis.add("" + filteredQuestions.get(randomIndex).getQues() + " đáp án là " + filteredQuestions.get(randomIndex).getAnswer());

}

}

private void displayQuestion(final question q) {

// Hiển thị câu hỏi

TextView tv = findViewById(R.id.textView9);

tv.setText(q.getQues());

// Tạo danh sách đáp án

final List<String> dapan = new ArrayList<>();

dapan.add(q.getA());

dapan.add(q.getB());

dapan.add(q.getC());

dapan.add(q.getD());

// Hiển thị danh sách đáp án

ArrayAdapter adap = new ArrayAdapter<>(QuestionActivity.this,R.layout.dapan, dapan);

lv.setAdapter(adap);

// Bắt đầu đếm ngược

startCountDownTimer();

lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

countDownTimer.cancel(); // Hủy bỏ đếm ngược

String selectedAnswer = dapan.get(position);

// Hiển thị đáp án đúng

String correctAnswer = q.getAnswer();

String message;

if (selectedAnswer.equals(correctAnswer)) {

message = "Bạn đã trả lời đúng. Đáp án đúng là: " + correctAnswer;

sco = sco +1;

} else {

message = "Bạn đã trả lời sai. Đáp án đúng là: " + correctAnswer;

}

showDialogWithDelay(message);

}

});

}

private void startCountDownTimer() {

countDownTimer = new CountDownTimer(10000, 1000) { // 10 giây, cập nhật mỗi giây

public void onTick(long millisUntilFinished) {

// Cập nhật số giây còn lại lên TextView

long secondsRemaining = millisUntilFinished / 1000;

TextView textViewCountdown = findViewById(R.id.textViewCountdown);

textViewCountdown.setText(String.valueOf(secondsRemaining));

}

public void onFinish() {

// Hiển thị dialog khi hết thời gian

showDialogWithDelay("Hết thời gian. Đáp án đúng là: " + lu.get(currentQuestionIndex).getAnswer());

}

}.start();

}

private void showDialogWithDelay(final String message) {

// Hiển thị dialog

final Dialog dialog = new Dialog(QuestionActivity.this);

dialog.setContentView(R.layout.chude);

TextView tvCorrectAnswer = dialog.findViewById(android.R.id.text1);

tvCorrectAnswer.setText(message);

dialog.show();

// Tắt dialog sau 2 giây và hiển thị câu hỏi tiếp theo

mHandler.postDelayed(new Runnable() {

@Override

public void run() {

dialog.dismiss();

currentQuestionIndex++;

if (currentQuestionIndex < 10) {

TextView cau = findViewById(R.id.textView14);

cau.setText("Câu số "+ (currentQuestionIndex+1));

max = max+1;

int randomIndex = getRandomNumberInRange( max);

min= randomIndex;

displayQuestion(filteredQuestions.get(randomIndex));

lis.add(""+filteredQuestions.get(randomIndex).getQues()+" đáp án là " + filteredQuestions.get(randomIndex).getAnswer());

} else {

// Kết thúc hoạt động nếu đã hiển thị đủ 10 câu hỏi

Intent tokq = new Intent(QuestionActivity.this, ScoreActivity.class);

tokq.putExtra("kq", sco);

tokq.putExtra("tk", u);

y = getIntent().getIntExtra("cd",y);

tokq.putExtra("chude",y);

tokq.putStringArrayListExtra("list", (ArrayList<String>) lis);

startActivity(tokq);

}

}

}, DELAY\_TIME);

}

private int getRandomNumberInRange(int max) {

if (min >= max) {

min =0;

return ran.nextInt(filteredQuestions.size()-1);

}

Random random = new Random();

return random.nextInt(max - min + 1) + min;

}

}

**3.1e StartActivity**

package com.example.quizzapp;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.ImageView;

import android.widget.ListView;

import android.widget.TextView;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;

import java.util.List;

import retrofit2.Call;

import retrofit2.Callback;

import retrofit2.Response;

public class StartActivity extends AppCompatActivity {

private List<String> li;

private List<question> lu;

private ListView lv;

private User u;

TextView ten,kc;

private ImageView ava;

public int z ;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_startquiz);

li = new ArrayList<>();

lu = new ArrayList<>();

u = getIntent().getParcelableExtra("tk",User.class);

lv = findViewById(R.id.lv);

ArrayAdapter<String> adapter = new ArrayAdapter<>(this, R.layout.chude, li);

lv.setAdapter(adapter);

ten = findViewById(R.id.textView5);

kc = findViewById(R.id.textView8);

ten.setText(u.getUsername());

kc.setText(""+u.getKimcuong());

ava = findViewById(R.id.image\_logo);

ava.setImageResource(u.getAvatar());

getListquestion(); // Gọi phương thức để lấy danh sách câu hỏi từ API

lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

// Truy cập TextView từ view được chọn

TextView textView = view.findViewById(android.R.id.text1);

// Kiểm tra textView có khác null không

if (textView != null) {

// Lấy văn bản từ TextView

String selectedText = textView.getText().toString();

// Chuyển sang màn hình câu hỏi và chuyền dữ liệu

Intent toques = new Intent(StartActivity.this, QuestionActivity.class);

toques.putExtra("chude", selectedText);

toques.putExtra("tk",u);

z= getIntent().getIntExtra("cd",z);

toques.putExtra("cd",z);

startActivity(toques);

} else {

// TextView không được tìm thấy trong item layout

Toast.makeText(StartActivity.this, "TextView not found", Toast.LENGTH\_SHORT).show();

}

}

});

}

private void getListquestion() {

ApiService apiService = RetrofitClient.getClient().create(ApiService.class);

Call<List<question>> call = apiService.getquestions();

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

@Override

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

lu = response.body();

updateListView(); // Sau khi nhận được dữ liệu từ API, cập nhật ListView

}

@Override

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

Toast.makeText(StartActivity.this, "Network error", Toast.LENGTH\_SHORT).show();

}

});

}

private void updateListView() {

for (question x : lu) {

int k = 0;

for (String i : li) {

if (x.getType().equals(i)) { // So sánh chuỗi sử dụng phương thức equals()

k = 1;

break;

}

}

if (k == 0) {

li.add(x.getType());

}

}

ArrayAdapter<String> adapter = (ArrayAdapter<String>) lv.getAdapter();

adapter.notifyDataSetChanged(); // Thông báo cho adapter rằng dữ liệu đã thay đổi

}

}

**3.1f ScoreActivity**

package com.example.quizzapp;

import android.app.Dialog;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.ImageView;

import android.widget.ListView;

import android.widget.TextView;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.List;

import retrofit2.Call;

import retrofit2.Callback;

import retrofit2.Response;

public class ScoreActivity extends AppCompatActivity {

private List<String> li;

private List<question> lu;

private ListView lv;

private User u;

private int sco;

List<String> lis;

private TextView ten,kc,kq;

private Button bac ;

private ImageView ava;

public int y;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_score);

bac = findViewById(R.id.back);

u = getIntent().getParcelableExtra("tk",User.class);

sco = getIntent().getIntExtra("kq",sco);

u.setScore(u.getScore()+sco\*10);

u.setTimep(u.getTimep()+10);

lis = getIntent().getStringArrayListExtra("list");

if(sco == 10){

u.setKimcuong(u.getKimcuong()+1);

Toast.makeText(ScoreActivity.this, "Trả lời đúng 10 câu liên tiếp được thêm 1 kim cương", Toast.LENGTH\_SHORT).show();

}

ten = findViewById(R.id.textView5);

kq = findViewById(R.id.correctAnswersTextView);

ten.setText(u.getUsername());

kq.setText(""+sco\*10+"/"+100);

bac.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent backmain = new Intent(ScoreActivity.this,MainActivity.class);

backmain.putExtra("tk",u);

int z=1;

backmain.putExtra("pn",z);

y= getIntent().getIntExtra("chude",y);

backmain.putExtra("chude",y);

startActivity(backmain);

}

});

updateUser();

Button coilaide = findViewById(R.id.coilaide);

coilaide.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Dialog dialog = new Dialog(ScoreActivity.this);

dialog.setContentView(R.layout.de);

dialog.getWindow().setLayout(ViewGroup.LayoutParams.MATCH\_PARENT, (int) (getResources().getDisplayMetrics().heightPixels \* 1));

ListView lv;

lv = dialog.findViewById(R.id.lvlv);

// Tạo CustomAdapter

ArrayAdapter adapter = new ArrayAdapter(ScoreActivity.this, R.layout.chude, lis);

// Thiết lập CustomAdapter cho ListView

lv.setAdapter(adapter);

dialog.show();

ImageView backmain = dialog.findViewById(R.id.imageView4);

backmain.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

dialog.dismiss();

}

});

}

});

}

private void updateUser() {

// Khởi tạo RetrofitClient và ApiService

ApiService apiService = RetrofitClient.getClient().create(ApiService.class);

// Tạo một yêu cầu PUT để cập nhật thông tin người dùng

Call<User> call = apiService.updateUser(u.getId(),u);

// Gửi yêu cầu PUT đến máy chủ

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

@Override

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

if (response.isSuccessful()) {

// Xử lý khi cập nhật thành công

User updatedUser = response.body();

} else {

// Xử lý khi gặp lỗi từ máy chủ

Toast.makeText(ScoreActivity.this, "Cập nhật thông tin người dùng thất bại", Toast.LENGTH\_SHORT).show();

}

}

@Override

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

// Xử lý khi gặp lỗi kết nối hoặc lỗi khác

Toast.makeText(ScoreActivity.this, "Không thể kết nối đến máy chủ", Toast.LENGTH\_SHORT).show();

}

});

}

}

**3.1g MucsicService**

package com.example.quizzapp;

import android.app.Service;

import android.content.Intent;

import android.media.MediaPlayer;

import android.os.IBinder;

import android.app.Service;

import android.content.Intent;

import android.media.MediaPlayer;

import android.os.Binder;

import android.os.IBinder;

public class MusicService extends Service {

private MediaPlayer mediaPlayer;

private static boolean isRunning = false;

public static boolean isRunning() {

return isRunning;

}

public class LocalBinder extends Binder {

MusicService getService() {

return MusicService.this;

}

}

private final IBinder binder = new LocalBinder();

@Override

public void onCreate() {

super.onCreate();

mediaPlayer = MediaPlayer.create(this, R.raw.nhacapp);

mediaPlayer.setLooping(true);

isRunning = true;

}

@Override

public int onStartCommand(Intent intent, int flags, int startId) {

if (!mediaPlayer.isPlaying()) {

mediaPlayer.start();

}

return START\_STICKY;

}

@Override

public void onDestroy() {

super.onDestroy();

if (mediaPlayer != null) {

mediaPlayer.stop();

mediaPlayer.release();

isRunning = false;

}

}

@Override

public IBinder onBind(Intent intent) {

return binder;

}

}

**3.1h User**

package com.example.quizzapp;

import android.os.Parcel;

import android.os.Parcelable;

public class User implements Parcelable {

private Long id;

private String username;

private String password;

private int score;

private String answerbaomat;

private String questionbaomat;

private int kimcuong;

private int avatar;

private int timep; // Thêm trường timep

// Constructors

public User(Long id, String username, String password, int score, String answerbaomat, String questionbaomat, int kimcuong, int avatar, int timep) {

this.id = id;

this.username = username;

this.password = password;

this.score = score;

this.answerbaomat = answerbaomat;

this.questionbaomat = questionbaomat;

this.kimcuong = kimcuong;

this.avatar = avatar;

this.timep = timep; // Khởi tạo trường timep

}

public User(){

}

// Getters and Setters

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getScore() {

return score;

}

public void setScore(int score) {

this.score = score;

}

public String getAnswerbaomat() {

return answerbaomat;

}

public void setAnswerbaomat(String answerbaomat) {

this.answerbaomat = answerbaomat;

}

public String getQuestionbaomat() {

return questionbaomat;

}

public void setQuestionbaomat(String questionbaomat) {

this.questionbaomat = questionbaomat;

}

public int getKimcuong() {

return kimcuong;

}

public void setKimcuong(int kimcuong) {

this.kimcuong = kimcuong;

}

public int getAvatar() {

return avatar;

}

public void setAvatar(int avatar) {

this.avatar = avatar;

}

public int getTimep() {

return timep;

}

public void setTimep(int timep) {

this.timep = timep;

}

// Parcelable implementation

protected User(Parcel in) {

id = in.readLong();

username = in.readString();

password = in.readString();

score = in.readInt();

answerbaomat = in.readString();

questionbaomat = in.readString();

kimcuong = in.readInt();

avatar = in.readInt();

timep = in.readInt(); // Đọc giá trị của timep

}

public static final Parcelable.Creator<User> CREATOR = new Parcelable.Creator<User>() {

@Override

public User createFromParcel(Parcel in) {

return new User(in);

}

@Override

public User[] newArray(int size) {

return new User[size];

}

};

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeLong(id);

dest.writeString(username);

dest.writeString(password);

dest.writeInt(score);

dest.writeString(answerbaomat);

dest.writeString(questionbaomat);

dest.writeInt(kimcuong);

dest.writeInt(avatar);

dest.writeInt(timep); // Ghi giá trị của timep

}

@Override

public int describeContents() {

return 0;

}

}

**3.1i question:**

package com.example.quizzapp;

import android.os.Parcel;

import android.os.Parcelable;

import android.widget.TextView;

public class question implements Parcelable {

private Long id;

private String ques;

private String type;

private String difficulty;

private String a; // Thay đổi tên thành a

private String b; // Thay đổi tên thành b

private String c; // Thay đổi tên thành c

private String d; // Thay đổi tên thành d

private String answer;

public question(Long id, String ques, String type, String difficulty, String a, String b, String c, String d, String answer) {

this.id = id;

this.ques = ques;

this.type = type;

this.difficulty = difficulty;

this.a = a; // Thay đổi tên thành a

this.b = b; // Thay đổi tên thành b

this.c = c; // Thay đổi tên thành c

this.d = d; // Thay đổi tên thành d

this.answer = answer;

}

protected question(Parcel in) {

id = in.readLong();

ques = in.readString();

type = in.readString();

difficulty = in.readString();

a = in.readString(); // Thay đổi tên thành a

b = in.readString(); // Thay đổi tên thành b

c = in.readString(); // Thay đổi tên thành c

d = in.readString(); // Thay đổi tên thành d

answer = in.readString();

}

public static final Creator<question> CREATOR = new Creator<question>() {

@Override

public question createFromParcel(Parcel in) {

return new question(in);

}

@Override

public question[] newArray(int size) {

return new question[size];

}

};

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getQues() {

return ques;

}

public void setQues(String ques) {

this.ques = ques;

}

public String getType() {

return type;

}

public void setType(String type) {

this.type = type;

}

public String getDifficulty() {

return difficulty;

}

public void setDifficulty(String difficulty) {

this.difficulty = difficulty;

}

public String getA() {

return a;

}

public void setA(String a) {

this.a = a;

}

public String getB() {

return b;

}

public void setB(String b) {

this.b = b;

}

public String getC() {

return c;

}

public void setC(String c) {

this.c = c;

}

public String getD() {

return d;

}

public void setD(String d) {

this.d = d;

}

public String getAnswer() {

return answer;

}

public void setAnswer(String answer) {

this.answer = answer;

}

@Override

public int describeContents() {

return 0;

}

@Override

public void writeToParcel(Parcel dest, int flags) {

dest.writeLong(id);

dest.writeString(ques);

dest.writeString(type);

dest.writeString(difficulty);

dest.writeString(a); // Thay đổi tên thành a

dest.writeString(b); // Thay đổi tên thành b

dest.writeString(c); // Thay đổi tên thành c

dest.writeString(d); // Thay đổi tên thành d

dest.writeString(answer);

}

}

**3.1j ImageAdapter**

package com.example.quizzapp;

import android.view.View;

import android.view.ViewGroup;

import android.widget.BaseAdapter;

import android.widget.GridView;

import android.widget.ImageView;

import java.util.List;

public class ImageAdapter extends BaseAdapter {

private List<Integer> imageList;

public ImageAdapter(List<Integer> imageList) {

this.imageList = imageList;

}

@Override

public int getCount() {

return imageList.size();

}

@Override

public Integer getItem(int position) {

return imageList.get(position);

}

@Override

public long getItemId(int position) {

return position;

}

@Override

public View getView(int position, View convertView, ViewGroup parent) {

ImageView imageView;

if (convertView == null) {

imageView = new ImageView(parent.getContext());

imageView.setLayoutParams(new GridView.LayoutParams(400, 400)); // Đặt kích thước 150x150

imageView.setScaleType(ImageView.ScaleType.CENTER\_CROP);

} else {

imageView = (ImageView) convertView;

}

int imageResource = getItem(position);

imageView.setImageResource(imageResource);

return imageView;

}

}

**3.1k RetrofitClient**

package com.example.quizzapp;

import retrofit2.Retrofit;

import retrofit2.converter.gson.GsonConverterFactory;

public class RetrofitClient {

private static Retrofit retrofit = null;

private static final String BASE\_URL = "http://192.168.1.6:8052/api/";

public static Retrofit getClient() {

if (retrofit == null) {

retrofit = new Retrofit.Builder()

.baseUrl(BASE\_URL)

.addConverterFactory(GsonConverterFactory.create())

.build();

}

return retrofit;

}

}

**3.1l ApiService:**

package com.example.quizzapp;

import retrofit2.Call;

import retrofit2.http.Body;

import retrofit2.http.DELETE;

import retrofit2.http.GET;

import retrofit2.http.POST;

import retrofit2.http.PUT;

import retrofit2.http.Path;

import java.util.List;

public interface ApiService {

// GET all users

@GET("users")

Call<List<User>> getUsers();

// POST a new user

@POST("users")

Call<User> createUser(@Body User user);

// DELETE a user by ID

@DELETE("users/{id}")

Call<User> deleteUser(@Path("id") long id);

// PUT (update) a user by ID

@PUT("users/{id}")

Call<User> updateUser(@Path("id") long id, @Body User user);

@GET("questions")

Call<List<question>> getquestions();

// DELETE a user by ID

// PUT (update) a user by ID

}

**3.2 API**

**3.2a question**

package com.example.quizapp;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "question")

public class question {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(columnDefinition = "nvarchar(255)")

private String ques;

@Column(columnDefinition = "nvarchar(255)")

private String type;

@Column(columnDefinition = "nvarchar(255)")

private String difficulty;

@Column(columnDefinition = "nvarchar(255)")

private String A;

@Column(columnDefinition = "nvarchar(255)")

private String B;

@Column(columnDefinition = "nvarchar(255)")

private String C;

@Column(columnDefinition = "nvarchar(255)")

private String D;

@Column(columnDefinition = "nvarchar(255)")

private String answer;

public String getType() {

return type;

}

public void setType(String type) {

this.type = type;

}

public String getDifficulty() {

return difficulty;

}

public void setDifficulty(String difficulty) {

this.difficulty = difficulty;

}

public String getAnswer() {

return answer;

}

public void setAnswer(String answer) {

this.answer = answer;

}

public question(Long id, String ques, String type, String difficulty, String a, String b, String c, String d,

String answer) {

super();

this.id = id;

this.ques = ques;

this.type = type;

this.difficulty = difficulty;

A = a;

B = b;

C = c;

D = d;

this.answer = answer;

}

// Constructors

public question() {

// Hàm khởi tạo mặc định

}

public question(Long id, String ques, String a, String b, String c, String d) {

super();

this.id = id;

this.ques = ques;

A = a;

B = b;

C = c;

D = d;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getQues() {

return ques;

}

public void setQues(String ques) {

this.ques = ques;

}

public String getA() {

return A;

}

public void setA(String a) {

A = a;

}

public String getB() {

return B;

}

public void setB(String b) {

B = b;

}

public String getC() {

return C;

}

public void setC(String c) {

C = c;

}

public String getD() {

return D;

}

public void setD(String d) {

D = d;

}

}

**3.2b User**

package com.example.quizapp;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "users")

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(columnDefinition = "nvarchar(255)")

private String username;

@Column(columnDefinition = "nvarchar(255)")

private String password;

private int score;

@Column(columnDefinition = "nvarchar(255)")

private String questionbaomat;

@Column(columnDefinition = "nvarchar(255)")

private String answerbaomat;

private int timep;

private int kimcuong;

private int avatar;

public User(Long id, String username, String password, int score, String questionbaomat, String answerbaomat,

int timep, int kimcuong, int avatar) {

super();

this.id = id;

this.username = username;

this.password = password;

this.score = score;

this.questionbaomat = questionbaomat;

this.answerbaomat = answerbaomat;

this.timep = timep;

this.kimcuong = kimcuong;

this.avatar = avatar;

}

public User() {

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public int getScore() {

return score;

}

public void setScore(int score) {

this.score = score;

}

public String getQuestionbaomat() {

return questionbaomat;

}

public void setQuestionbaomat(String questionbaomat) {

this.questionbaomat = questionbaomat;

}

public String getAnswerbaomat() {

return answerbaomat;

}

public void setAnswerbaomat(String answerbaomat) {

this.answerbaomat = answerbaomat;

}

public int getTimep() {

return timep;

}

public void setTimep(int timep) {

this.timep = timep;

}

public int getKimcuong() {

return kimcuong;

}

public void setKimcuong(int kimcuong) {

this.kimcuong = kimcuong;

}

public int getAvatar() {

return avatar;

}

public void setAvatar(int avatar) {

this.avatar = avatar;

}

}

**3.2c UserRepository**

package com.example.quizapp;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface UserRepository extends JpaRepository<User, Long> {

User findByUsername(String username);

}

**3.2d questionRepository**

package com.example.quizapp;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface questionRepository extends JpaRepository<question,Long> {

question findByques(String ques);

}

**3.2e questionController**

package com.example.quizapp;

import java.util.List;

import org.springframework.dao.EmptyResultDataAccessException;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import jakarta.persistence.EntityNotFoundException;

@RestController

@RequestMapping("/api/questions")

public class questionController {

private final questionRepository QuestionRepository;

public questionController(questionRepository QuestionRepository) {

this.QuestionRepository = QuestionRepository;

}

@GetMapping

public List<question> getAllquestion() {

return QuestionRepository.findAll();

}

@PostMapping

public question createUser(@RequestBody question Question) {

return QuestionRepository.save(Question);

}

@PutMapping("/{id}")

public question updateUser(@PathVariable Long id, @RequestBody question Question) {

question existingQuestion = QuestionRepository.findById(id)

.orElseThrow(() -> new EntityNotFoundException("Question not found"));

existingQuestion.setA(Question.getA());

existingQuestion.setB(Question.getB());

existingQuestion.setC(Question.getC());

existingQuestion.setD(Question.getD());

existingQuestion.setQues(Question.getQues());

existingQuestion.setType(Question.getType());

existingQuestion.setDifficulty(Question.getDifficulty());

existingQuestion.setAnswer(Question.getAnswer());

return QuestionRepository.save(existingQuestion);

}

// Các phương thức xóa và các phương thức tùy chỉnh khác

@DeleteMapping("/{id}")

public ResponseEntity<?> deleteQuestion(@PathVariable Long id) {

try {

QuestionRepository.deleteById(id);

return ResponseEntity.ok().build();

} catch (EmptyResultDataAccessException e) {

throw new EntityNotFoundException("Question not found");

}

}

@GetMapping("/{id}")

public ResponseEntity<question> getQuestionById(@PathVariable Long id) {

question Question = QuestionRepository.findById(id)

.orElseThrow(() -> new EntityNotFoundException("Question not found"));

return ResponseEntity.ok().body(Question);

}

@PostMapping("/batch")

public ResponseEntity<List<question>> createQuestions(@RequestBody List<question> questions) {

List<question> savedQuestions = QuestionRepository.saveAll(questions);

return ResponseEntity.ok().body(savedQuestions);

}

}

**3.2f UserController**

package com.example.quizapp;

import java.util.List;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import jakarta.persistence.EntityNotFoundException;

@RestController

@RequestMapping("/api/users")

public class UserController {

private final UserRepository userRepository;

public UserController(UserRepository userRepository) {

this.userRepository = userRepository;

}

@GetMapping

public List<User> getAllUsers() {

return userRepository.findAll();

}

@PostMapping

public User createUser(@RequestBody User user) {

// Trước khi lưu user, bạn có thể đặt giá trị cho timep nếu cần

// Ví dụ: user.setTimep(0);

return userRepository.save(user);

}

@GetMapping("/{id}")

public ResponseEntity<User> getUserById(@PathVariable Long id) {

User user = userRepository.findById(id)

.orElseThrow(() -> new EntityNotFoundException("User not found"));

// Bạn có thể kiểm tra và thực hiện xử lý với trường timep ở đây nếu cần

return ResponseEntity.ok().body(user);

}

@PutMapping("/{id}")

public User updateUser(@PathVariable Long id, @RequestBody User user) {

User existingUser = userRepository.findById(id)

.orElseThrow(() -> new EntityNotFoundException("User not found"));

existingUser.setUsername(user.getUsername());

existingUser.setPassword(user.getPassword());

existingUser.setScore(user.getScore());

existingUser.setQuestionbaomat(user.getQuestionbaomat());

existingUser.setAnswerbaomat(user.getAnswerbaomat());

existingUser.setKimcuong(user.getKimcuong());

existingUser.setAvatar(user.getAvatar());

existingUser.setTimep(user.getTimep()); // Cập nhật trường timep

return userRepository.save(existingUser);

}

@DeleteMapping("/{id}")

public ResponseEntity<?> deleteUser(@PathVariable Long id) {

userRepository.findById(id)

.orElseThrow(() -> new EntityNotFoundException("User not found"));

userRepository.deleteById(id);

return ResponseEntity.ok().build();

}

}