Phòng thí nghiệm An toàn Thông tin InsecLab
Trường Đại học Công nghệ Thông tin (UIT)

BÁO CÁO THỰC HÀNH

Môn học: Bảo mật web

Tên chủ đề: Lập trình an toàn ứng dụng Android cơ bản

GVHD: Ngô Đức Hoàng Sơn

Nhóm: 12

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT213.O21.ANTT.1

STT	Họ và tên	MSSV	Email
1	Nguyễn Triệu Thiên Bảo	21520155	21520155@gm.uit.edu.vn
2	Trần Lê Minh Ngọc	21521195	21521195@gm.uit.edu.vn
3	Huỳnh Minh Khuê	21522240	21522240@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Nội dung	Tình trạng	Trang
1	Yêu cầu 1	70%	2 - 4
2	Yêu cầu 2	100%	4 - 8
3	Yêu cầu 3	100%	9 – 10
4	Yêu cầu 4	100%	10 - 11
Điểm tự đánh giá			**?/10**

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

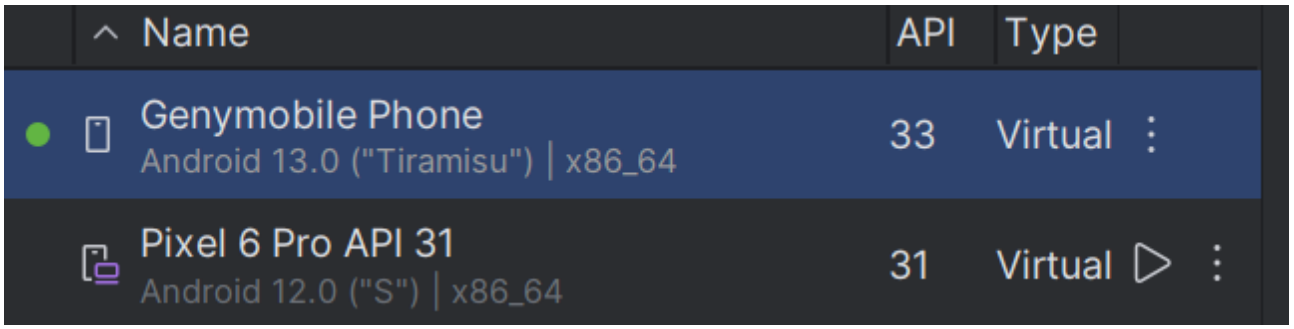
Yêu cầu 1: Sinh viên tiếp tục sửa lỗi Broadcast Receivers.

***Mục tiêu:** Tạo 1 ứng dụng có thể exploit ứng dụng InsecureBankv2 bằng cách để cho BroadcastReceiver của InsecureBankv2 bắt được lời gọi của của ứng dụng exploit, sau đó cho phép ứng dụng này gửi 1 tin nhắn vào trong máy nạn nhân (máy có cài đặt ứng dụng InsecureBankv2).

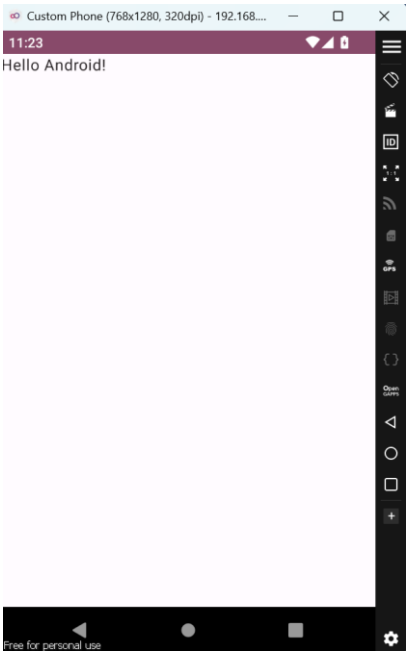
- Gửi intent cho bên phía BroadcastReceiver của app InsecureBankv2. BroadcastReceiver của app này sẽ lắng nghe intent mỗi khi có lời gọi yêu cầu thay đổi mật khẩu (từ lớp ChangePassword.class), cũng như lời gọi intent từ các ứng dụng khác.

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

    /*broadcastReceiver = MyBroadcastReceiver()
    val filter = IntentFilter("theBroadcast")
    registerReceiver(broadcastReceiver, filter)*/

    intent = Intent( action: "theBroadcast");
    intent.putExtra( name: "phonenumber", value: "+84900909090");
    intent.putExtra( name: "newpass", value: "Please give me a cup of coffee");
    sendBroadcast(intent);
```

- Dùng Genymobile để mở thiết bị ảo (do thiết bị này đã được cài đặt sẵn app InsecureBankv2)



- Giao diện của app:



***Vá lỗi:**

- Đổi trường **android:exported="true"** thành "**false**" để ngăn chặn lắng nghe intent từ các app khác.

```
<receiver android:exported="false" android:name="com.android.insecurebankv2.MyBroadCastReceiver">
    <intent-filter>
        <action android:name="theBroadcast"/>
    </intent-filter>
</receiver>
```

- Biên dịch mã nguồn:

```
PS C:\Users\HP\Downloads\Android-InsecureBankv2-master\Android-InsecureBankv2-master> .\apktool b InsecureBankv2 -o Inse
cureBankv3.apk
I: Using Apktool 2.9.3
I: Checking whether sources has changed...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: InsecureBankv3.apk
Press any key to continue . . .
PS C:\Users\HP\Downloads\Android-InsecureBankv2-master\Android-InsecureBankv2-master>
```

- Dùng keytool để tạo khóa ký ứng dụng:

```
PS C:\Users\HP\Downloads\Android-InsecureBankv2-master\Android-InsecureBankv2-master> keytool -genkey -v -keystore my-release-key.jks -keyalg RSA -keysize 2048 -validity 10000 -alias my-key-alias
Enter keystore password:
Re-enter new password:
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in braces.
What is your first and last name?
What is the name of your organizational unit?
What is the name of your organization?
What is the name of your City or Locality?
What is the name of your State or Province?
What is the two-letter country code for this unit?
Is CN=khue huynh, OU, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
  [no]:  yes
Generating 2,048 bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 10,000 days
        for: CN=khue huynh, OU, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
[Storing my-release-key.jks]
```

- Ký khóa ứng dụng:

```
PS C:\Users\HP\Downloads\Android-InsecureBankv2-master\Android-InsecureBankv2-master> jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.jks InsecureBankv3.apk my-key-alias
Enter Passphrase for keystore:
   adding: META-INF/MANIFEST.MF
   adding: META-INF/MY-KEY-A.SF
   adding: META-INF/MY-KEY-A.RSA
  signing: AndroidManifest.xml
  signing: classes.dex
  signing: res/anim/abc_fade_in.xml
  signing: res/anim/abc_fade_out.xml
  signing: res/anim/abc_grow_fade_in_from_bottom.xml
  signing: res/anim/abc_popup_enter.xml
  signing: res/anim/abc_popup_exit.xml
  signing: res/anim/abc_shrink_fade_out_from_bottom.xml
  signing: res/anim/abc_slide_in_bottom.xml
  signing: res/anim/abc_slide_in_top.xml
  signing: res/anim/abc_slide_out_bottom.xml
  signing: res/anim/abc_slide_out_top.xml
  signing: res/color/abc_primary_text_disable_only_material_dark.xml
  signing: res/color/abc_primary_text_disable_only_material_light.xml
  signing: res/color/abc_primary_text_material_dark.xml
  signing: res/color/abc_primary_text_material_light.xml
  signing: res/color/abc_search_url_text.xml
  signing: res/color/abc_secondary_text_material_dark.xml
  signing: res/color/abc_secondary_text_material_light.xml
  signing: res/color/common_signin_btn_text_dark.xml
  signing: res/color/common_signin_btn_text_light.xml
```

- Gỡ cài đặt InsecureBankv2 để cài InsecureBankv3:

```
PS C:\Program Files\Genymobile\Genymotion\tools> .\adb uninstall com.android.insecurebankv2
Success
PS C:\Program Files\Genymobile\Genymotion\tools> .\adb install "C:\Users\HP\Downloads\Android-InsecureBankv2-master\Android-InsecureBankv2-master\InsecureBankv3.apk"
Performing Streamed Install
Success
PS C:\Program Files\Genymobile\Genymotion\tools>
```

- Tiến hành tấn công lại và tiếp tục vá lỗi.

Yêu cầu 2: Sinh viên xây dựng ứng dụng Android gồm 3 giao diện chức năng chính:

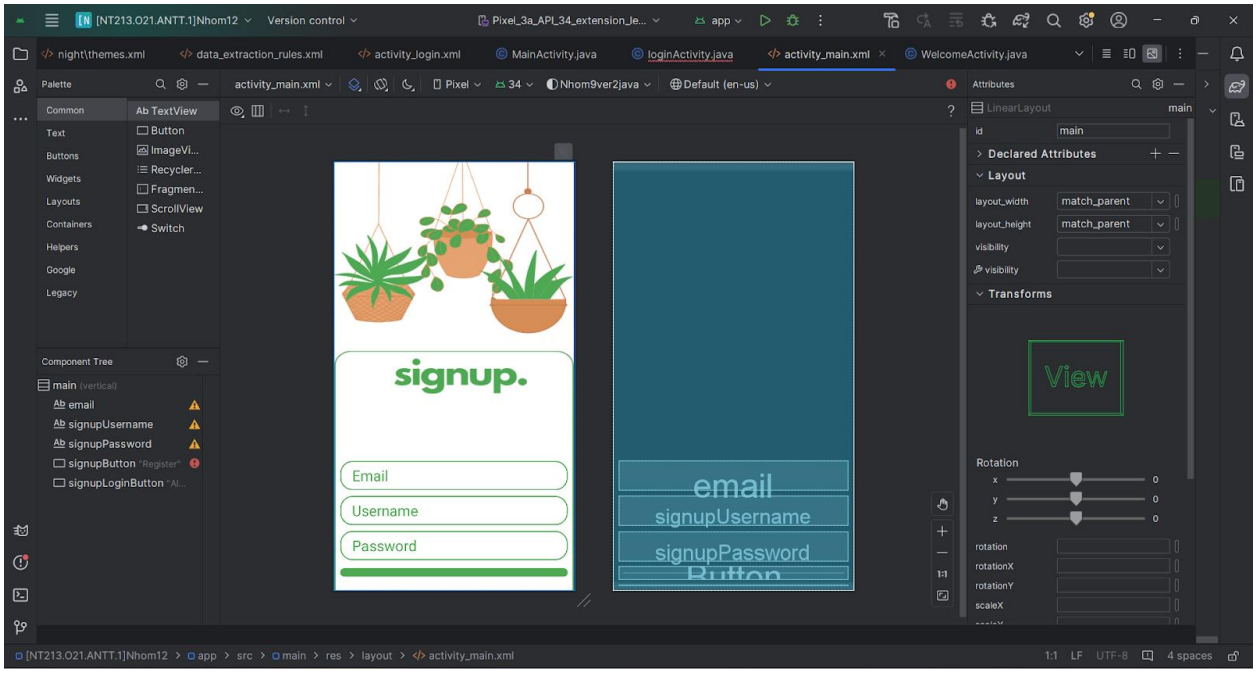
1) Register - Đăng ký thông tin với ứng dụng (email, username, password).

2) Login - Đăng nhập vào ứng dụng (username, password).

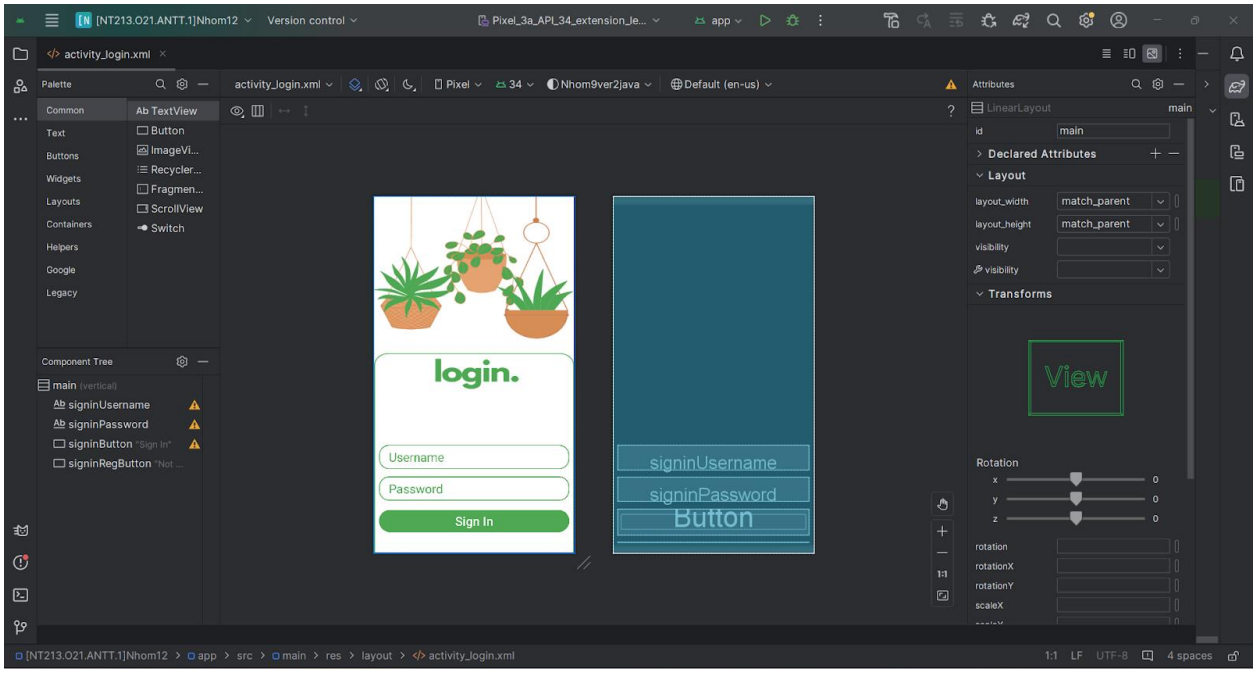
3) Hiển thị thông tin người dùng (một lời chào có tên người dùng).

Để tạo được ứng dụng đáp ứng các yêu cầu như trên thì ta lần lượt tạo các layout giao diện chức năng trước rồi code các chức năng sau

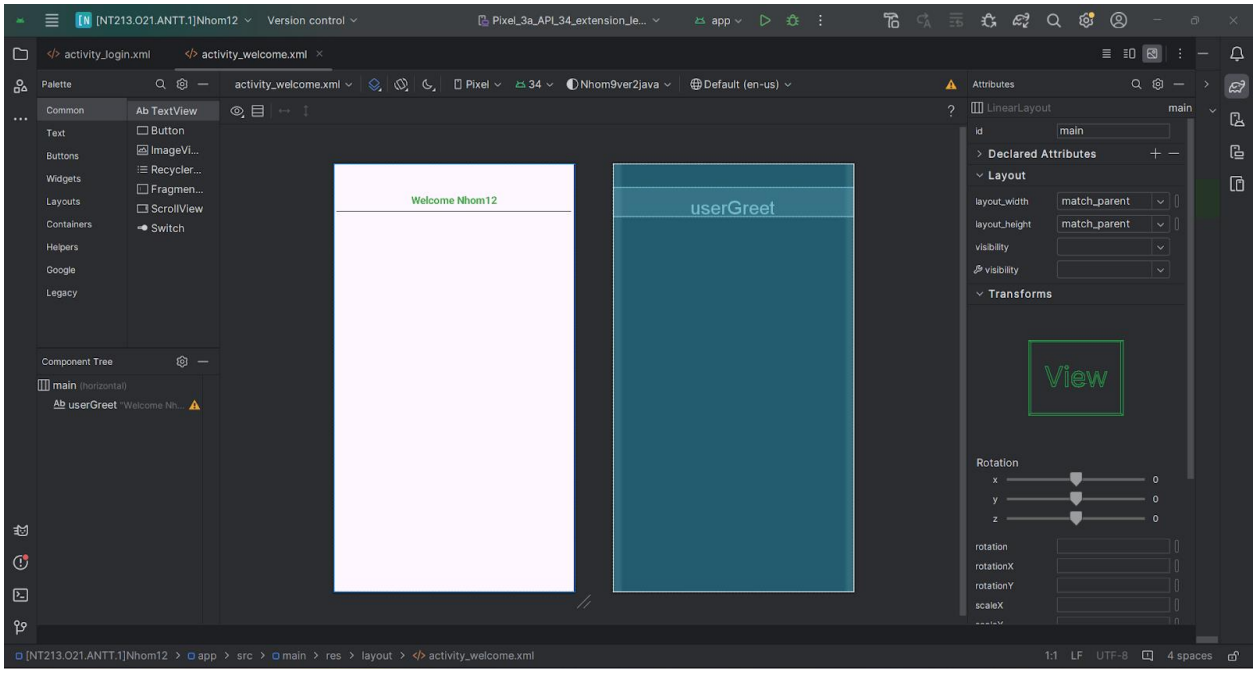
Layout **activity_main.xml** cũng chính là layout của MainActivity



Layout activity_login.xml là layout của LoginActivity



Layout activity_welcome.xml là layout của WelcomeActivity



Tiếp theo là xây dựng các Activity tương ứng với từng layout tương ứng
Đầu tiên là MainActivity có chức năng đăng kí
Khi được tạo, nó sẽ lấy những trường thông tin từ người dùng để dùng cho việc xử lý đăng ký

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable( $this$enableEdgeToEdge: this);
    setContentView(R.layout.activity_main);

    signupUsername=findViewById(R.id.signupUsername);
    signupPassword=findViewById(R.id.signupPassword);
    signupButton=findViewById(R.id.signupButton);
    signinButton=findViewById(R.id.signupLoginButton);
    email=findViewById(R.id.email);

    FirebaseDatabase db = FirebaseDatabase.getInstance();
    DatabaseReference ref = db.getReference( path: "users");
```

Khi ta nhấn nút đăng kí, hàm sẽ lấy các trường thông tin, nếu 1 trong 3 trường này bị thiếu chúng ta sẽ hiện thông báo là tất cả các trường phải được nhập vào

```
signupButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String user=signupUsername.getText().toString();
        String pass= signupPassword.getText().toString();
        String Email=email.getText().toString();

        if(TextUtils.isEmpty(user) || TextUtils.isEmpty(pass) || TextUtils.isEmpty(Email)){
            Toast.makeText( context: MainActivity.this, text: "All fields Required", Toast.LENGTH_SHORT).show();
        }
```

Sau đó chúng ta sẽ kiểm tra username có tồn tại hay không, nếu không tồn tại thì nó sẽ hash password người dùng vào insert người dùng đó vào database vào chuyển qua giao diện chính không thì sẽ thông báo là "Registration failed" hoặc là user đã tồn tại

```
                else{
                    User user1 = new User(Email, user, hashedPassword);
                    ref.child(user).setValue(user1).addOnSuccessListener(avoid ->{
                        Toast.makeText( context: MainActivity.this, text: "Registered successfully", Toast.LENGTH_SHORT).show();
                        Intent intent = new Intent(getApplicationContext(), WelcomeActivity.class);
                        intent.putExtra( name: "username", user);
                        startActivity(intent);
                    }).addOnFailureListener(e -> {
                        Toast.makeText( context: MainActivity.this, text: "Registration failed", Toast.LENGTH_SHORT).show();
                    });
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError error) {

            }
        });

    }
}
});
```

Ngoài nút đăng kí, ta còn 1 nút chuyển qua phần đăng nhập. Ở phần này ta chỉ đơn giản là khởi tạo một Intent mới và start nó

```
        signinButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(getApplicationContext(), loginActivity.class);
                startActivity(intent);
            }
        });

        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
            Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
            return insets;
        });
    }
}
```

Tiếp theo là xây dựng LoginActivity

Tương tự với đăng ký thì chúng ta lấy thông tin từ phía người dùng nhập, sau đó sẽ kiểm tra những cái thông tin mà người dùng nhập có đầy đủ không, nếu không thì sẽ hiện dòng chữ là "All fields required"

Nếu thỏa mãn, nó sẽ hash password và truyền vào hàm checkLogin của DBhelper nếu bằng true nghĩa là thông tin người dùng nhập vào đủ và sẽ thông báo login successful và chuyển qua màn hình chính nếu không thì sẽ báo là login failed

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable( $this$enableEdgeToEdge: this);
    setContentView(R.layout.activity_login);

    username = findViewById(R.id.signinUsername);
    password =findViewById(R.id.signinPassword);
    signin = findViewById(R.id.signinButton);
    signup = findViewById(R.id.signinRegButton);

    FirebaseDatabase db = FirebaseDatabase.getInstance();
    DatabaseReference ref = db.getReference( path: "users");
```

```
signin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String user=username.getText().toString();
        String pass=password.getText().toString();

        if(TextUtils.isEmpty(user) || TextUtils.isEmpty(pass)){
            Toast.makeText( context: loginActivity.this,  text: "All fields required", Toast.LENGTH_SHORT).show();
        }
        else{
            String hashedPassword = hashPassword(pass);
            ref.addListenerForSingleValueEvent(new ValueEventListener() {
                2 usages
                @Override
                public void onDataChange(@NonNull DataSnapshot snapshot) {
                    if(snapshot.hasChild(user)){
                        String passDB = snapshot.child(user).child( path: "password").getValue(String.class);
                        Log.i(hashedPassword, passDB);
                        if(passDB.equals(hashedPassword)){
                            Toast.makeText( context: loginActivity.this,  text: "Login successfully", Toast.LENGTH_SHORT).show();
                            Intent intent = new Intent(getApplicationContext(), WelcomeActivity.class);
                            intent.putExtra( name: "username", user);
                            startActivity(intent);
                        }
                        else{
                            Toast.makeText( context: loginActivity.this,  text: "Login failed", Toast.LENGTH_SHORT).show();
                        }
                    }
                    else{
                        Toast.makeText( context: loginActivity.this,  text: "User's Not Existed", Toast.LENGTH_SHORT).show();
                    }
```

Giống như phần đăng kí, trang đăng nhập cũng có 1 nút để chuyển tới phần đăng kí. Ta sẽ khởi tạo một Intent mới và start nó

```
    signup.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(getApplicationContext(), MainActivity.class);
            startActivity(intent);
        }
    });
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
        Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
        v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
        return insets;
    });
}
```

Yêu cầu 3: Sinh viên viết mã nguồn Java cho chức năng đăng nhập và đăng ký, sử dụng tập tin SQLiteConnector được giảng viên cung cấp để thực hiện kết nối đến cơ sở dữ liệu SQLite với các yêu cầu bên dưới.

2.1a. Thêm thông tin gồm email, username và password vào cơ sở dữ liệu khi người dùng dùng chức năng Đăng ký.

2.1b. Truy vấn thông tin username và password cho chức năng đăng nhập.

Ta sẽ xây dựng 1 HelperClass tên là DBHelper để kết nối tới SQLite và thực hiện truy vấn trong database

Đầu tiên là khởi tạo class với các phương thức onCreate sẽ khởi tạo một table users mới,

- onUpgrade sẽ drop table

```
no usages
public class DBHelper extends SQLiteOpenHelper {

    no usages
    public static final String DBNAME="login.db";
    no usages
    public DBHelper(Context context) {
        super(context, name: "login.db", factory: null, version: 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create table users(username TEXT primary key,email TEXT, password TEXT)");
    }

    10 usages
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("drop table if exists users");
    }
```

Tiếp tới là hàm insertData dùng cho chức năng đăng ký thêm một người dùng mới vào database với 3 tham số insert vào là email, username và password

```
no usages
public boolean insertData(String username,String email, String password){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues values = new ContentValues();


    values.put("username", username);
    values.put("email", email);
    values.put("password", password);

    long result = db.insert( table: "users",  nullColumnHack: null, values);
    if (result == 0 ) return false;
    else
        return true;
}
```

· Hàm checkUsername để kiểm tra username có tồn tại hay không dùng cho việc đăng nhập và đăng ký người dùng

```
no usages
public boolean checkUsername(String username ){
    SQLiteDatabase db = this.getWritableDatabase();
    Cursor cursor = db.rawQuery( sql: "select * from users where username=?", new String[] {username});
    if(cursor.getCount() > 0){
        return true;
    }
    else return false;
}
```

· Hàm checkLogin để kiểm tra việc đăng nhập của người dùng nếu người dùng nhập đúng sẽ trả về true không sẽ trả về false

```
no usages
public boolean checkLogin(String username, String password){
    SQLiteDatabase db = this.getWritableDatabase();
    Cursor cursor = db.rawQuery( sql: "select * from users where username=? and password=?",
            new String[] {username,password});
    if(cursor.getCount() > 0){
        return true;
    }
    else return false;
}
```

Yêu cầu 4: Điều chỉnh mã nguồn để password được lưu và kiểm tra dưới dạng mã hash thay vì plaintext.

Ở trong phần đăng ký, Sau khi đã nhận được password từ người dùng, chúng ta sẽ thực hiện hash password của người dùng nhập vào thông qua hàm hashPassword

```
public void onClick(View v) {
    String user=signupUsername.getText().toString();
    String pass= signupPassword.getText().toString();
    String Email=email.getText().toString();

    if(TextUtils.isEmpty(user) || TextUtils.isEmpty(pass) || TextUtils.isEmpty(Email)){
        Toast.makeText( context: MainActivity.this, text: "All fields Required", Toast.LENGTH_SHORT).show();
    }
    else{
        String hashedPassword = hashPassword(pass);
        ref.addListenerForSingleValueEvent(new ValueEventListener() {
```

Ở phần đăng nhập cũng thêm hàm tương tự trước khi truyền password vào hàm kiểm tra

```
signin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String user=username.getText().toString();
        String pass=password.getText().toString();

        if(TextUtils.isEmpty(user) || TextUtils.isEmpty(pass)){
            Toast.makeText( context: loginActivity.this, text: "All fields required", Toast.LENGTH_SHORT).show();
        }
        else{
            String hashedPassword = hashPassword(pass);
            ref.addListenerForSingleValueEvent(new ValueEventListener() {
```

Hàm hashPassword sẽ lấy tham số truyền vào là password ở dạng plaintext và hash với mã hash là SHA256

```
private String hashPassword(String password) {
    try {
        MessageDigest md = MessageDigest.getInstance( algorithm: "SHA-256");
        byte[] hashedBytes = md.digest(password.getBytes());

        // Convert the byte array to hexadecimal string
        StringBuilder sb = new StringBuilder();
        for (byte b : hashedBytes) {
            sb.append(String.format("%02x", b));
        }
        Log.i( tag: "test password", sb.toString());

        return sb.toString();
    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
    }
    return null;
}
```