

## BÁO CÁO THỰC HÀNH

Môn học: Bảo mật web và ứng dụng

### Lab 5: LẬP TRÌNH AN TOÀN ỨNG DỤNG ANDROID CƠ BẢN

Ngày báo cáo: 16/05/2024

#### 1. THÔNG TIN CHUNG:

Lớp: NT213.O22.ATCL.2

STT	Họ và tên	MSSV	Email
1	Hoàng Gia Bảo	21521848	21521848@gm.uit.edu.vn
2	Phạm Hoàng Phúc	21521295	21521295@gm.uit.edu.vn
3	Nguyễn Đại Nghĩa	21521182	21521182@gm.uit.edu.vn

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

## BÁO CÁO CHI TIẾT

### Bài tập 1:

Đọc file AndroidManifest.xml

```
For smali/baksmali info, see: https://github.com/google/smali

(bao@kali)-[~/Desktop/Baitapluuyentap]
$ java -jar apktool_2.9.3.jar d InsecureBankv2.apk
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
I: Using Apktool 2.9.3 on InsecureBankv2.apk
I: Loading resource table ...
I: Decoding file-resources ...
I: Loading resource table from file: /home/bao/.local/share/apktool/framework/1.apk
I: Decoding values */* XMLs ...
I: Decoding AndroidManifest.xml with resources ...
I: Regular manifest package ...
I: Baksmaling classes.dex ...
I: Copying assets and libs ...
I: Copying unknown files ...
I: Copying original files ...

(bao@kali)-[~/Desktop/Baitapluuyentap]
$
```

⇒ Từ đây có thể thấy rằng ta có thể dùng 1 app khác để thay đổi nội dung của Broadcast Receiver

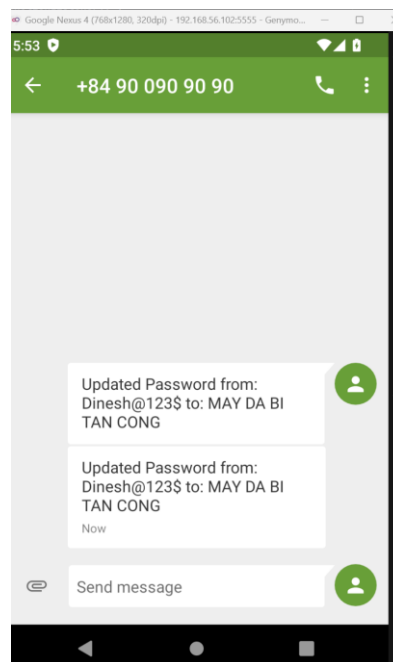
Ta Build 1 app có hàm như dưới :

```

20
21 @Override
22 public void onClick(View v) {
23     Log.i("tag: "Saigon", msg: "MAYDABITANCONG");
24     // Create an Intent object for the broadcast
25     Intent intent = new Intent(action: "theBroadcast");
26     intent.putExtra(name: "phonenumber", value: "+84900909090");
27     intent.putExtra(name: "newpass", value: "MAY DA BI TAN CONG");
28     // Send the broadcast
29     sendBroadcast(intent);

```

Kết quả :



Ta sửa lại value true-> False để vá lỗ hổng này

```

<activity android:exported="true" android:label="@string/title_activity_view_statement" android:name="com.android.insecurebankv2.view"
<provider android:authorities="com.android.insecurebankv2.TrackUserContentProvider" android:exported="true" android:name="com.android
<receiver android:exported="false" android:name="com.android.insecurebankv2.MyBroadCastReceiver">
<intent-filter>

```

Build lại app

```

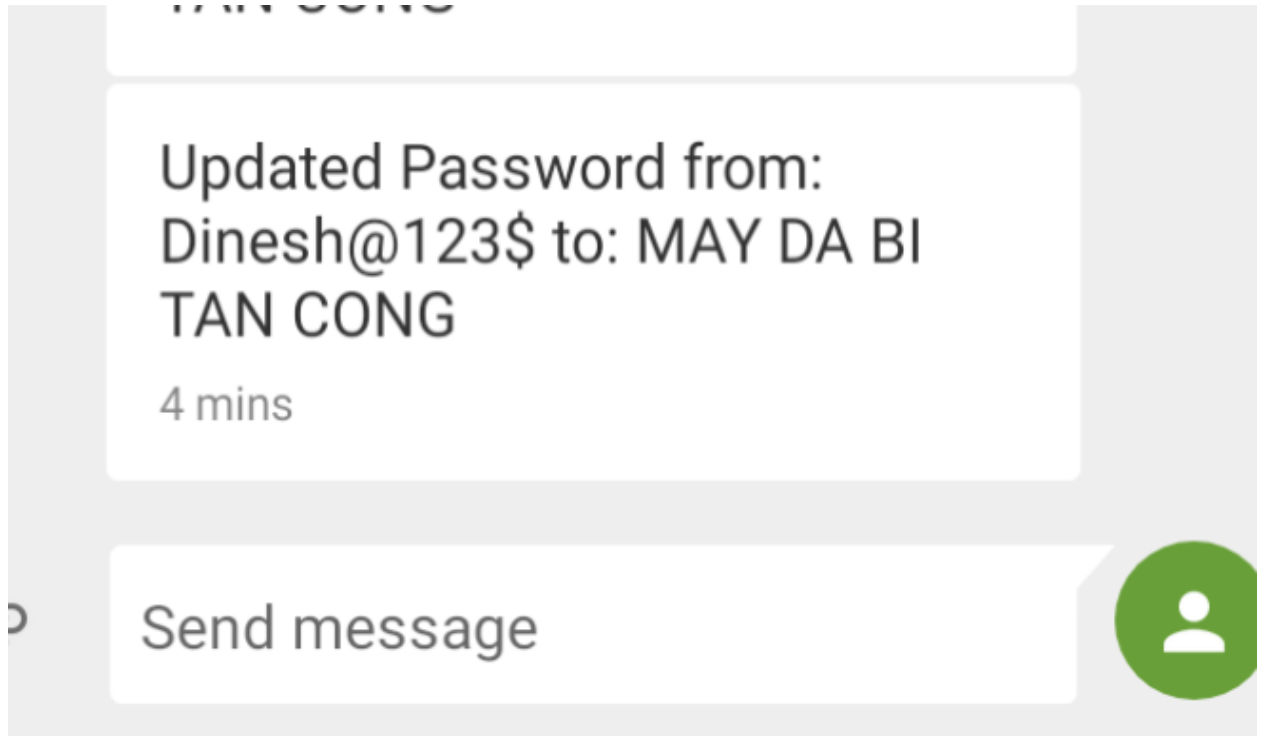
(bao@kali)-[~/Desktop/Baitapluuyentap]
$ java -jar apktool_2.9.3.jar b InsecureBankv2
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
I: Using Apktool 2.9.3
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: InsecureBankv2/dist/InsecureBankv2.apk
(bao@kali)-[~/Desktop/Baitapluuyentap]

```

Ký lại app

```
(bao@kali)-[~/Desktop/Baitapluventap]
$ apksigner sign --ks fivev2.keystore InsecureBankv2.apk
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Keystore password for signer #1:
(bao@kali)-[~/Desktop/Baitapluventap]
```

Kết quả :

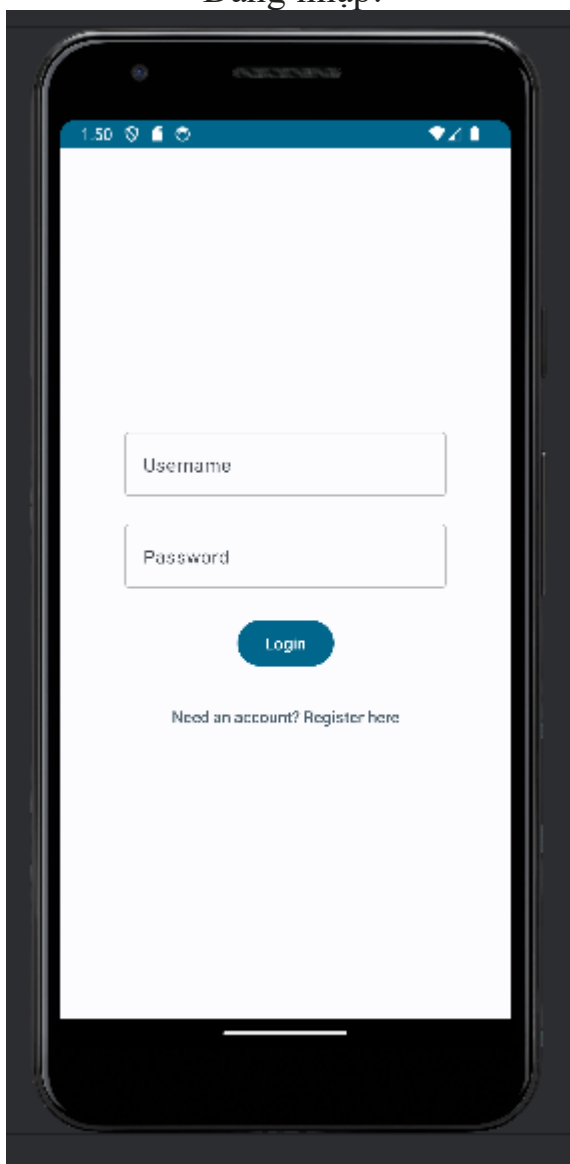


⇒ Sau khi chạy lại ứng dụng để gửi thông điệp này đến máy có ứng dụng InsecureBankv2 thì không được nữa!

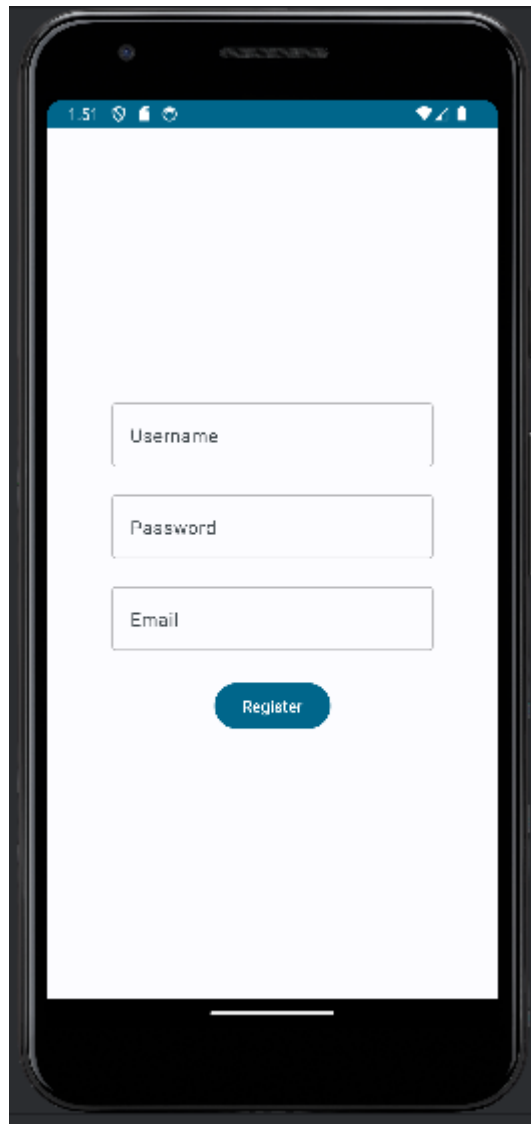
## **Bài tập 2:**

Code của ứng dụng này em sẽ đính kèm với bài nộp, sau đây sẽ là các giao diện của các chức năng.

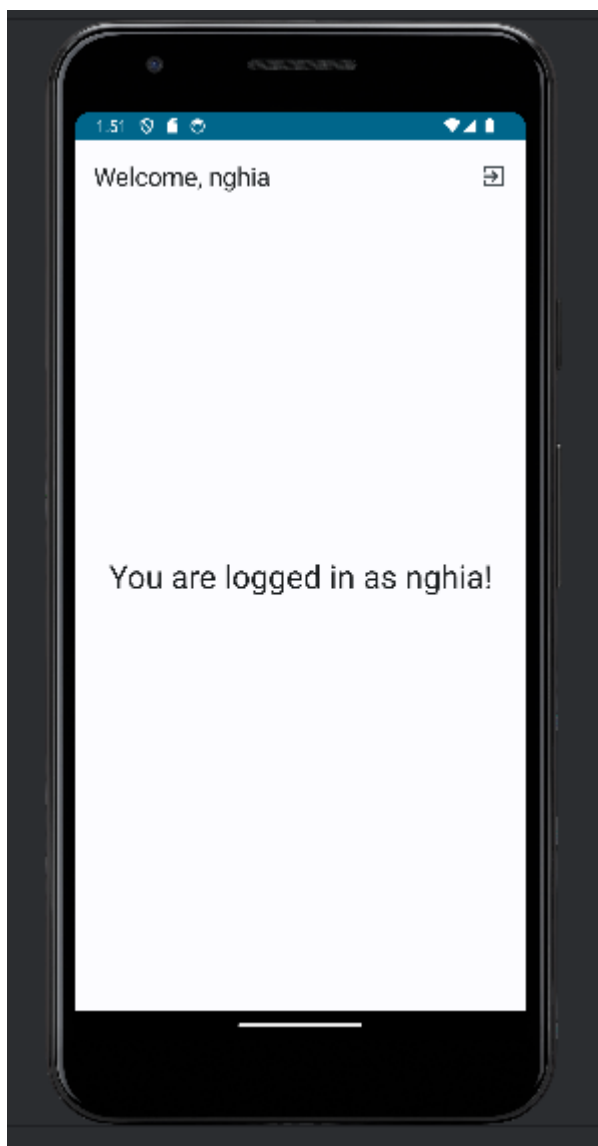
Đăng nhập:



Đăng kí:

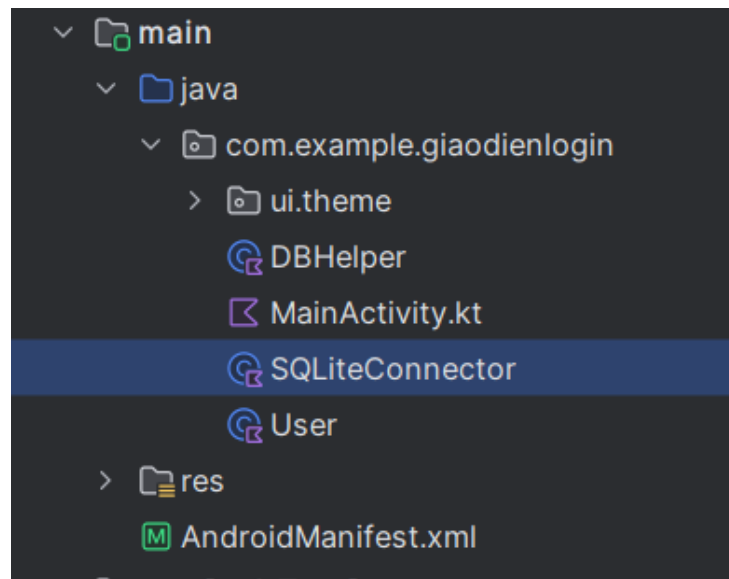


Sau khi đăng nhập thành công sẽ hiển thị như sau:



### **Bài tập 3:**

Import file SQLiteConnector



Tạo class User

```

1 package com.example.giaodienlogin
2 class User {
3     //ID
4     var id = 0
5
6     //USERNAME
7     var name: String? = null
8
9     //EMAIL
10    var email: String? = null
11
12    //PASSWORD
13    var password: String? = null
14 }
15

```

Code thực hiện chức năng lưu đăng ký, lưu thông tin vào cơ sở dữ liệu



```

Button(
    onClick = {
        if (username.isBlank() || password.isBlank() || email.isBlank()) {
            errorMessage = "All fields must be filled out."
            registerError = true
        } else if (!isValidEmail(email)) {
            errorMessage = "Please enter a valid email address."
            registerError = true
        } else {
            CoroutineScope(Dispatchers.Main).launch { this: CoroutineScope
                if (dbHelper.checkUserExists(username) || dbHelper.checkEmailExists(email)) {
                    if (dbHelper.checkUserExists(username)) {
                        errorMessage = "Username already exists. Please try another one."
                    } else {
                        errorMessage = "Email already exists. Please try another one."
                    }
                    registerError = true
                } else {
                    val added = dbHelper.addUser(username, password, email)

                    val newUser = User()
                    newUser.name = username
                    newUser.password = password
                    newUser.email = email
                    SQLiteConnector.addUser(newUser)

                    if (added) {
                        onRegisterSuccess()
                    } else {
                        errorMessage = "Registration failed. Please try again."
                        registerError = true
                    }
                }
            }
        }
    }
)

```

#### **Bài tập 4:**

Dưới đây là mã nguồn em đã thay đổi để mà có thể lưu và kiểm tra password dưới dạng mã hash thay vì plaintext:

```

suspend fun addUser(username: String, password: String, email: String): Boolean = withContext(Dispatchers.IO) { this: CoroutineScope
    val db = writableDatabase
    val hashedPassword = hashPassword(password)
    val contentValues = ContentValues().apply { this: ContentValues
        put("username", username)
        put("password", hashedPassword)
        put("email", email)
    }
    try {
        val result = db.insertOrThrow(table: "users", nullColumnHack: null, contentValues)
        return@withContext (result != -1L)
    } catch (e: Exception) {
        Log.e(tag: "DBHelper", msg: "Failed to add user: ${e.message}")
        Log.e(tag: "DBHelper", Log.getStackTraceString(e))
        return@withContext false
    } finally {
        db.close()
    }
}

new *
fun hashPassword(password: String): String {
    val digest = MessageDigest.getInstance(algorithm: "SHA-256")
    val hash = digest.digest(password.toByteArray(StandardCharsets.UTF_8))
    return hash.joinToString(separator: "") { "%02x".format(it) }
}

```

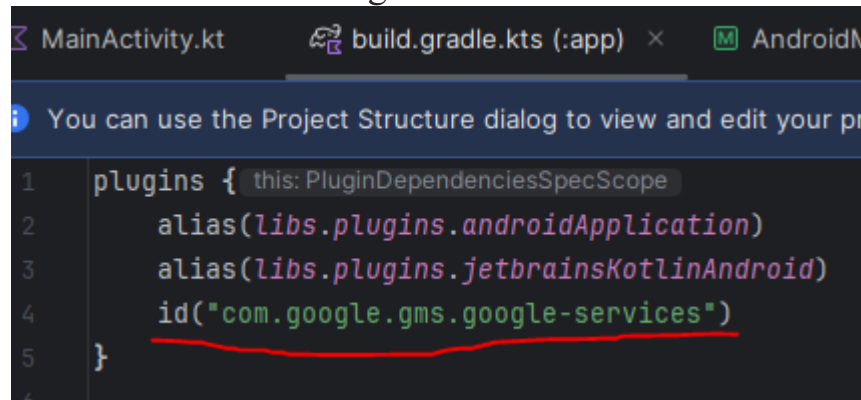
```

* Nghia *
fun validateUser(username: String, password: String): Boolean {
    val db = this.readableDatabase
    val cursor = db.rawQuery(sql: "SELECT password FROM users WHERE username=?", arrayOf(username))
    if (cursor.moveToFirst()) {
        val storedPassword = cursor.getString(cursor.getColumnIndexOrThrow("password"))
        cursor.close()
        db.close()
        return storedPassword == hashPassword(password)
    }
    cursor.close()
    db.close()
    return false
}

```

## Bài tập 5:

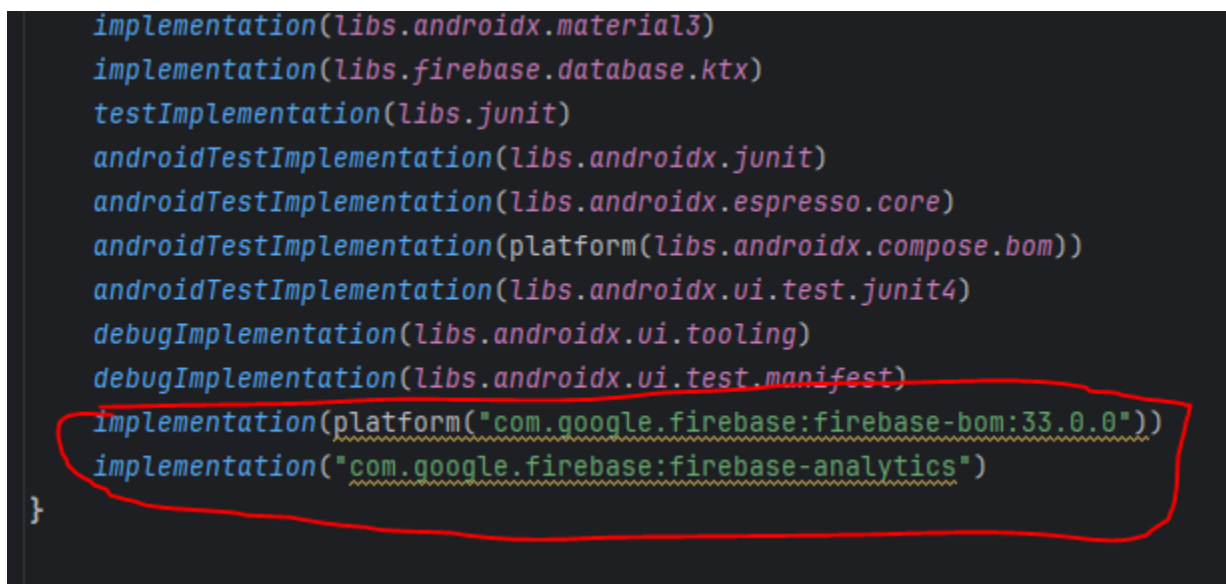
Em sử dụng firebase để làm CSDL, để có thể dùng được thì cần phải thêm code ở 2 file build.gradle.kts như sau:



```

1 plugins { this: PluginDependenciesSpecScope
2     alias(libs.plugins.androidApplication)
3     alias(libs.plugins.jetbrainsKotlinAndroid)
4     id("com.google.gms.google-services")
5 }

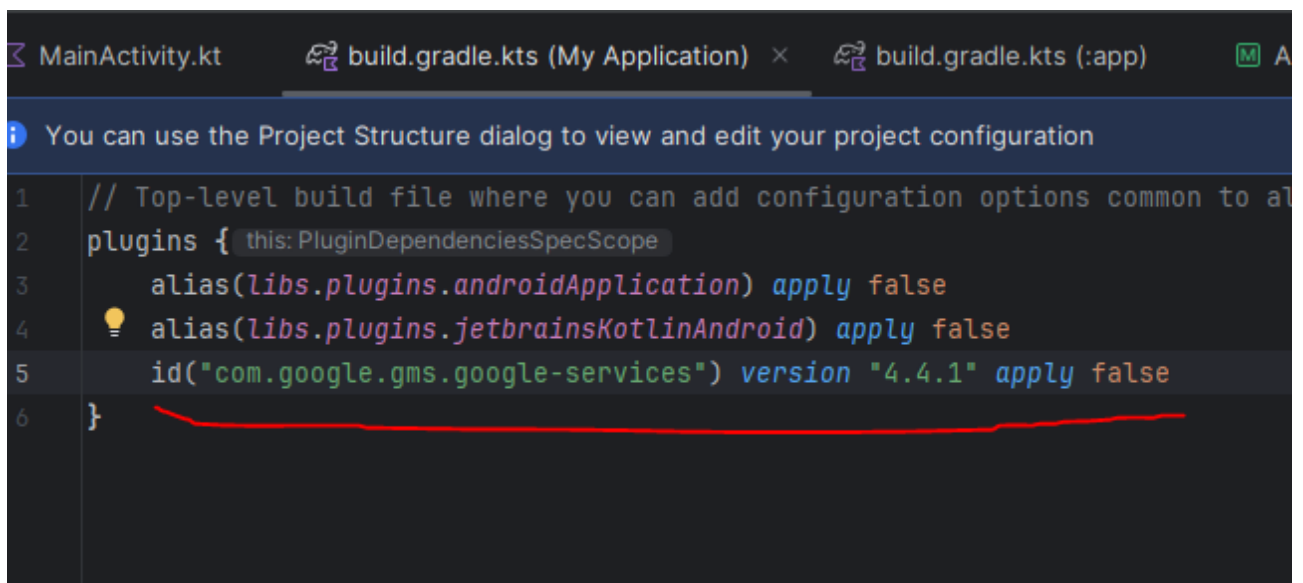
```



```

implementation(libs.androidx.material3)
implementation(libs.firebase.database.ktx)
testImplementation(libs.junit)
androidTestImplementation(libs.androidx.junit)
androidTestImplementation(libs.androidx.espresso.core)
androidTestImplementation(platform(libs.androidx.compose.bom))
androidTestImplementation(libs.androidx.ui.test.junit4)
debugImplementation(libs.androidx.ui.tooling)
debugImplementation(libs.androidx.ui.test.manifest)
implementation(platform("com.google.firebase:firebase-bom:33.0.0"))
implementation("com.google.firebase:firebase-analytics")
}

```



```

1 // Top-level build file where you can add configuration options common to all sub-projects/modules
2 plugins { this: PluginDependenciesSpecScope
3     alias(libs.plugins.androidApplication) apply false
4     alias(libs.plugins.jetbrainsKotlinAndroid) apply false
5     id("com.google.gms.google-services") version "4.4.1" apply false
6 }

```

Dưới đây là đoạn code để có thể lưu và truy xuất dữ liệu:

```

fun loginUser(username: String, password: String, onSuccess: () -> Unit, onError: () -> Unit) {
    val database = FirebaseDatabase.getInstance().getReference(path: "users")
    val query = database.orderByChild(path: "username").equalTo(username)

    query.addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            if (snapshot.exists() && snapshot.children.first().child(path: "password").value == password) {
                onSuccess()
            } else {
                onError()
            }
        }
        override fun onCancelled(error: DatabaseError) {
            onError()
        }
    })
}

fun registerUser(username: String, password: String, email: String, onSuccess: () -> Unit, onError: (String) -> Unit) {
    val database = FirebaseDatabase.getInstance().getReference(path: "users")
    val userId = database.push().key ?: ""
    val userMap = mapOf(
        "username" to username,
        "password" to password,
        "email" to email
    )

    database.child(userId).setValue(userMap).addOnSuccessListener { it: Void! }
        onSuccess()
    }.addOnFailureListener { it: Exception }
        onError("Failed to register. Please try again.")
}

```

## **Bài tập 6:**

File proguard em viết như sau:

```

23 # Giữ không đổi tên các lớp chính
24 -keep public class * extends android.app.Application
25 -keep public class * extends android.app.Activity
26 -keep public class * extends android.app.Service
27 -keep public class * extends android.content.BroadcastReceiver
28 -keep public class * extends android.content.ContentProvider
29
30 # Giữ các phương thức trong các lớp Activity, v.v.
31 -keepclassmembers class * extends android.app.Activity {
32     public void *(android.view.View);
33 }
34
35 # Giữ các enum
36 -keepclassmembers enum * {
37     public static **[] values();
38     public static ** valueOf(java.lang.String);
39 }
40
41 # Giữ getter/setter
42 -keepclassmembers class * {
43     void set*(**);
44     ** get*();
45 }
46
47 # Giữ các thành viên của các lớp được sử dụng bởi Parcelable
48 -keepclassmembers class * implements android.os.Parcelable {
49     static ** CREATOR;
50 }
51
52 # Giữ các lớp ViewModel
53 -keep class * extends androidx.lifecycle.ViewModel {
54     <init>(...);
55 }
56
57 # Giữ các views và setters của chúng được sử dụng trong các layout XML.
58 -keepclassmembers public class * extends android.view.View {
59     public <init>(android.content.Context);
60     public <init>(android.content.Context, android.util.AttributeSet);
61     public <init>(android.content.Context, android.util.AttributeSet, int);
62     public void set*(**);
63 }

```

```

64
65 # Giữ các liệt kê được sử dụng trong các giao tiếp được tuần tự hóa.
66 -keepclassmembers enum * {
67     public static **[] values();
68     public static ** valueOf(java.lang.String);
69 }
70
71 # Giữ Parcelable Creator
72 -keepclassmembers class * implements android.os.Parcelable {
73     public static final android.os.Parcelable$Creator CREATOR;
74 }
75
76 # Các lớp ViewModel không nên bị obfuscate để đảm bảo chúng có thể được khởi tạo một cách chính xác.
77 -keep class * extends androidx.lifecycle.ViewModel {
78     <init>(...);
79 }
80
81 # Các trường LiveData không nên bị đổi tên hoặc xóa để đảm bảo chức năng chính xác.
82 -keep class androidx.lifecycle.LiveData {
83     *;
84 }
85
86 # Giữ tất cả các lớp runtime của Compose
87 -keep class androidx.compose.runtime.** { *; }
88
89 # Giữ các hàm và lớp Composable
90 -keepclasseswithmembers class * {
91     @androidx.compose.runtime.Composable *;
92 }
93
94 # Giữ các lớp với lambda được tạo bởi Compose
95 -keepclassmembers class * {
96     void lambda$*(...);
97 }
98

```

Sau proguard thì các tên biến đã bị thay đổi đi so với không có, dưới đây là 1 vài ví dụ:

```
DBHelper.smali x
71     const-string v2, "user.db"
72
73     invoke-direct {p0, p1, v2, v0, v1}, Landroid/database/sqlite/SQLiteOpenHelper; -> <init>(Landroid/content/Context;Ljava/lang/String;Landroid/database/sqlite/SQLiteOpenHelper;)V
74
75     return-void
76 .end method
77
78
79 # virtual methods
80 .method public final addUser(Ljava/lang/String;Ljava/lang/String;Ljava/lang/String;Lkotlin/coroutines/Continuation;)Ljava/lang/Object;
81     .locals 8
82     .param p1, "username"    # Ljava/lang/String;
83     .param p2, "password"    # Ljava/lang/String;
84     .param p3, "email"       # Ljava/lang/String;
85     .param p4, "$completion" # Lkotlin/coroutines/Continuation;
86     .annotation system Ldalvik/annotation/Signature;
87         value = {
88             "(",
89             "Ljava/lang/String;",
90             "Ljava/lang/String;",
91             "Ljava/lang/String;",
92             "Lkotlin/coroutines/Continuation<",
93             "Ljava/lang/Boolean;",
94             ">;)",
95             "Ljava/lang/Object;"
96         }
97     .end annotation
98
99     .line 28
100     invoke-static {}, Lkotlinx/coroutines/Dispatchers; -> getIO()Lkotlinx/coroutines/CoroutineDispatcher;
101
102     move-result-object v0
103
104     check-cast v0, Lkotlin/coroutines/CoroutineContext;
```

```
DBHelper.smali x
136
137     const-string v0, "email"
138
139     invoke-static {p1, v0}, Lkotlin/jvm/internal/Intrinsics; -> checkNotNullParameter(Ljava/lang/Object;Ljava/lang/String;)V
140
141     .line 67
142     invoke-virtual {p0}, Lcom/example/giaodienlogin/DBHelper; -> getReadableDatabase()Landroid/database/sqlite/SQLiteDatabase;
143
144     move-result-object v0
145
146     .line 68
147     .local v0, "db":Landroid/database/sqlite/SQLiteDatabase;
148     const-string v1, "SELECT * FROM users WHERE email=?"
149
150     filled-new-array {p1}, [Ljava/lang/String;
151
152     move-result-object v2
153
154     invoke-virtual {v0, v1, v2}, Landroid/database/sqlite/SQLiteDatabase; ->.rawQuery(Ljava/lang/String;[Ljava/lang/String;)Landroid/database/Cursor;
155
156     move-result-object v1
157
158     .line 69
159     .local v1, "cursor":Landroid/database/Cursor;
160     invoke-interface {v1}, Landroid/database/Cursor; -> getCount()I
161
162     move-result v2
163
164     if-lez v2, :cond_0
165
166     const/4 v2, 0x1
167
168     goto :goto_0
```

```
MainActivityKt.smali x
141
142     move-result v5
143
144     if-eqz v5, :cond_4
145
146     const/4 v5, -0x1
147
148     const-string v7, "com.example.giaodienlogin.AuthenticationScreen (MainActivity.kt:44)"
149
150     invoke-static {v2, v4, v5, v7}, Landroidx/compose/runtime/ComposerKt;.->traceEventStart(IIILjava/lang/String;)V
151
152     .line 46
153     :cond_4
154     const/4 v2, 0x0
155
156     move v5, v2
157
158     .local v5, "$changed$iv":I
159     const/4 v7, 0x0
160
161     .local v7, "$if$remember":I
162     const v8, -0x1d58f75c
163
164     invoke-interface {v3, v8}, Landroidx/compose/runtime/Composer;.->startReplaceableGroup(I)V
165
166     const-string v9, "CC(remember):Composables.kt#9igjgp"
167
168     invoke-static {v3, v9}, Landroidx/compose/runtime/ComposerKt;.->sourceInformation(Landroidx/compose/runtime/Composer;Ljava/lang/String;)V
169
170     .line 266
171     const/4 v10, 0x0
172
173     .local v10, "invalid$iv$iv":Z
174     move-object v11, v3
175
176     .local v11, "$this$cache$iv$iv":Landroidx/compose/runtime/Composer;
177     const/4 v12, 0x0
```