

BÁO CÁO THỰC HÀNH

Môn học: Bảo mật Web và ứng dụng

Lab 5: Lập trình an toàn ứng dụng Android cơ bản

GVHD: Ngô Đức Hoàng Sơn

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT132.012.ATCL.2- Nhóm 4

STT	Họ và tên	MSSV	Email
1	Đỗ Thị Yến Ly	21520337	21520337@gm.uit.edu.vn
2	Lê Hoàng Oanh	21521253	21521253@gm.uit.edu.vn
3	Nguyễn Đại Bảo Duy	21520772	21520772@gm.uit.edu.vn

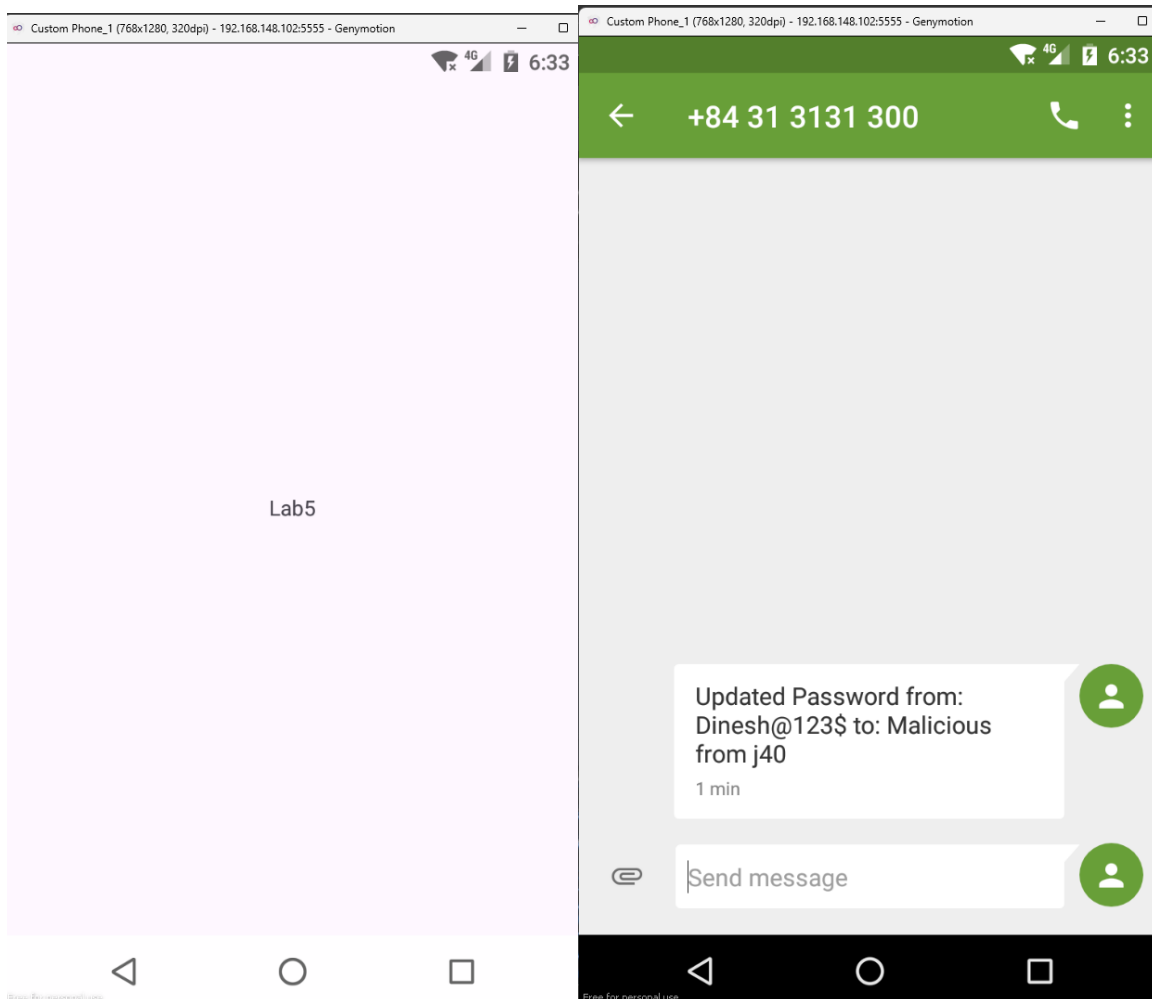
2. NỘI DUNG THỰC HIỆN:

STT	Công việc	Kết quả tự đánh giá
1	Yêu cầu 1	100%
2	Yêu cầu 2	100%
3	Yêu cầu 3	100%
4	Yêu cầu 4	100%
5	Yêu cầu 5	100%
6	Yêu cầu 6	100%
7	Yêu cầu 7	100%

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

BÁO CÁO CHI TIẾT

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



Build app thành công và tấn công BroastCast

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable( $this$enableEdgeToEdge: this);
    setContentView(R.layout.activity_main);
    Intent intent = new Intent( action: "theBroadcast");
    intent.putExtra( name: "phonenumber", value: "+84313131300");
    intent.putExtra( name: "newpass", value: "Malicious from j40");
    sendBroadcast(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;
    });
}

```

Mã nguồn app tấn công

```

<activity android:label="@string/title_activity_do_login" android:name="com.android.insecurebankv2.DoLogin"/>
<activity android:exported="false" android:label="@string/title_activity_post_login" android:name="com.android.insecurebankv2.PostLogin"/>
<activity android:label="@string/title_activity_wrong_login" android:name="com.android.insecurebankv2.WrongLogin"/>
<activity android:exported="true" android:label="@string/title_activity_do_transfer" android:name="com.android.insecurebankv2.DoTransfer"/>
<activity android:exported="true" android:label="@string/title_activity_view_statement" android:name="com.android.insecurebankv2.ViewStatement"/>
<provider android:authorities="com.android.insecurebankv2.TrackUserContentProvider" android:exported="true" android:name="com.android.insecurebankv2.TrackUserContentProvider"/>
<receiver android:exported="false" android:name="com.android.insecurebankv2.MyBroadCastReceiver">
    <intent-filter>
        <action android:name="theBroadcast"/>
    </intent-filter>
</receiver>

```

Để khắc phục ta tắt chế độ Broadcast của ứng dụng đổi giá trị thành false

```

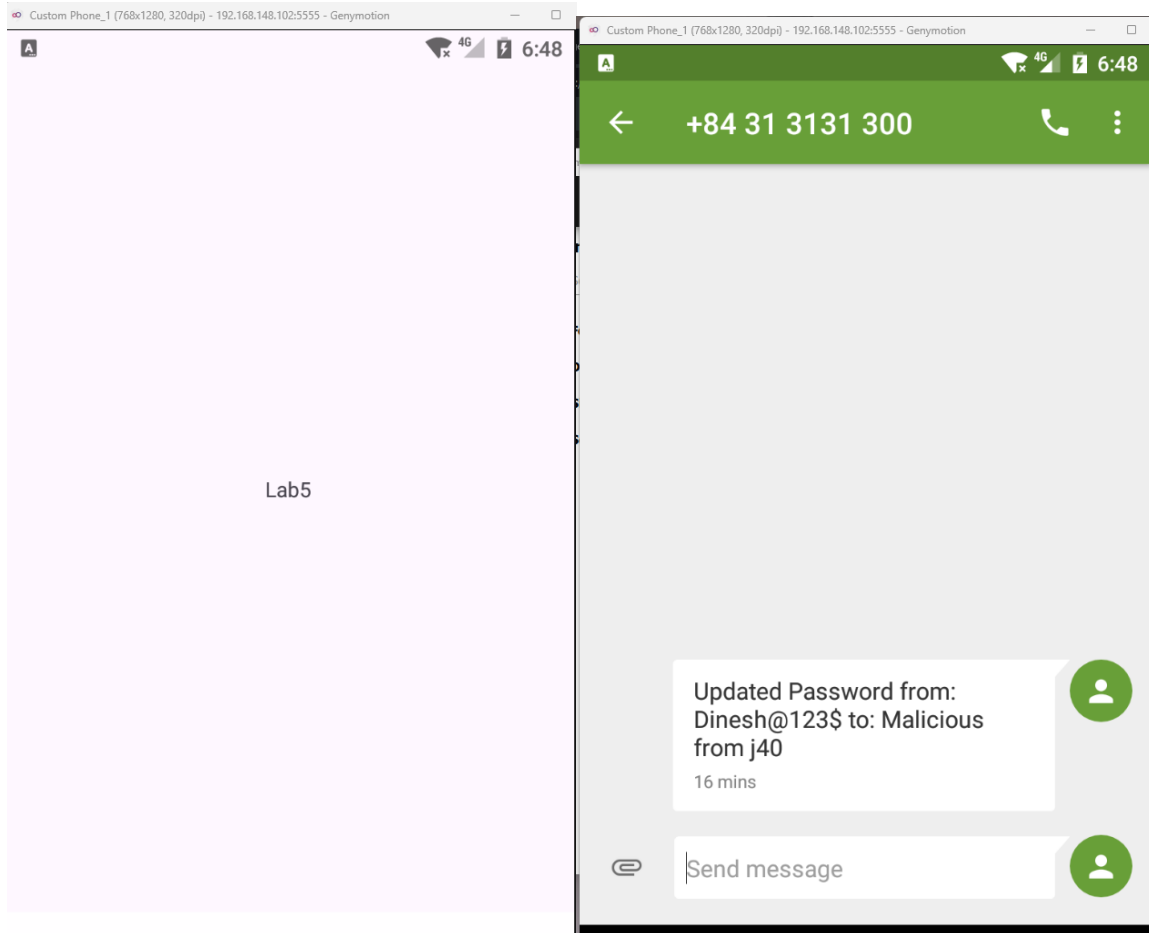
C:\Users\21520\Downloads\Lab5\Lab5>C:\Users\21520\AppData\Local\Android\Sdk\build-tools\34.0.0\apksigner sign --ks InsecureBankv3.key store InsecureBankv3.apk
Keystore password for signer #1:

C:\Users\21520\Downloads\Lab5\Lab5>C:\Users\21520\Downloads\platform-tools-latest-windows\platform-tools\adb install InsecureBankv3.apk
Performing Streamed Install
Success

C:\Users\21520\Downloads\Lab5\Lab5>

```

Build lại ứng dụng InsecureBank và cài đặt

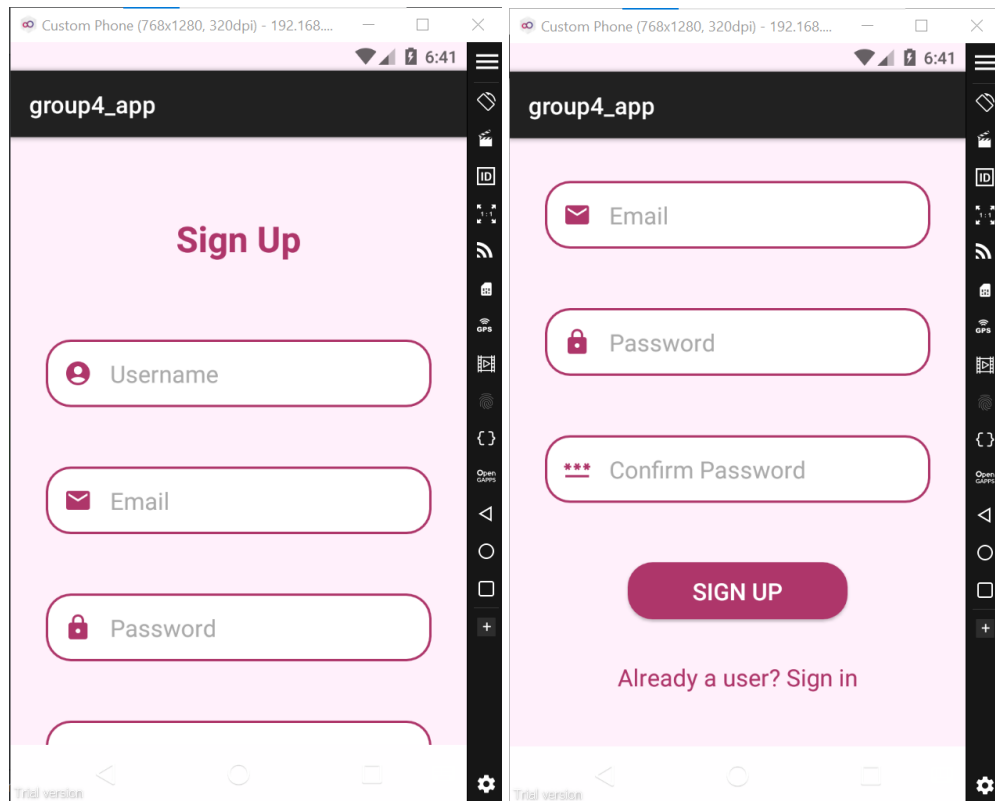


Không còn nhận được thông điệp từ attacker nữa

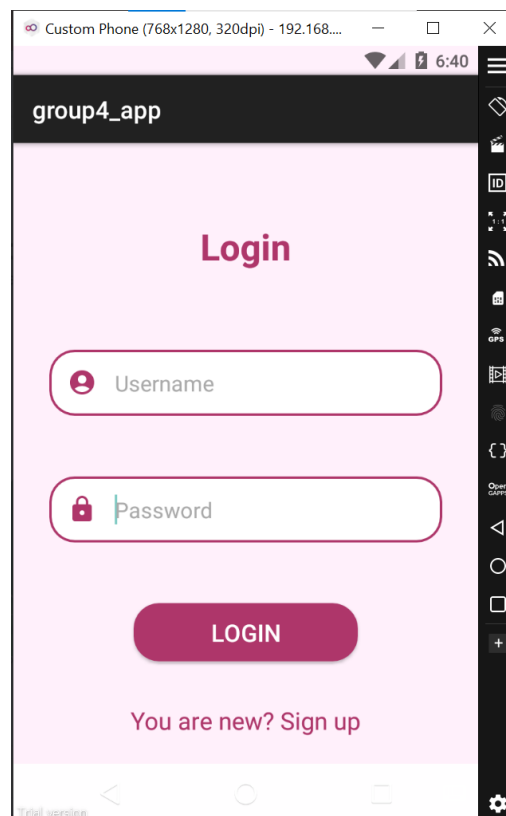
Yêu cầu 2. Sinh viên xây dựng ứng dụng Android gồm 3 giao diện chức năng chính:

- 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).**

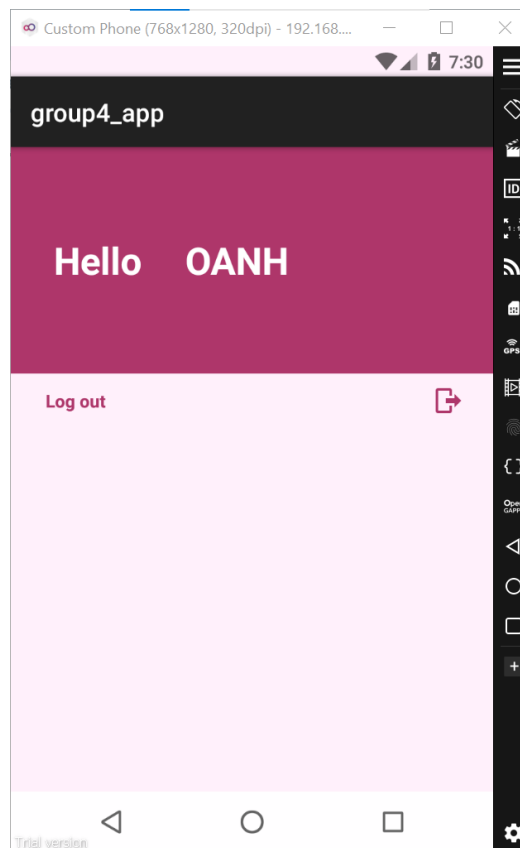
1) Register



2) Login

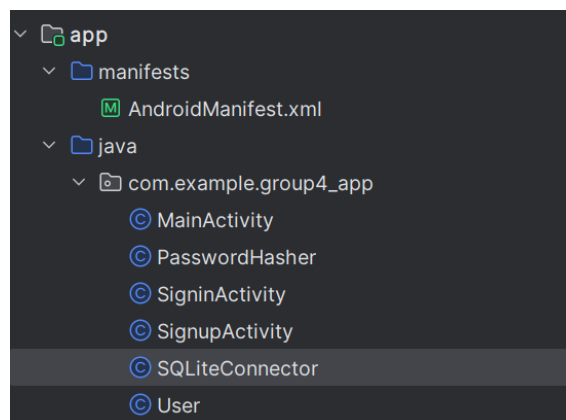


3) Một lời chào có tên người dùng



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.

Ta thấy file SQLiteConnector đã được thêm vào



Tạo file .java định nghĩa một class User gồm các trường tối thiểu id, username, password, email cùng các phương thức get và set giá trị của các trường này.

```

1 package com.example.group4_app;
2
3 public class User {
4     private int id;
5     private String name;
6     private String email;
7     private String password;
8
9     public int getId(){return id;}
10    public void setId (int id){this.id=id;}
11
12    public String getName() { return name; }
13
14    public void setName(String name){ this.name = name; }
15
16    public String getEmail() { return email; }
17
18    public void setEmail(String email){ this.email = email; }
19
20    public String getPassword() { return password; }
21
22    public void setPassword(String password){ this.password = password; }

```

Mã nguồn Java cho chức năng đăng nhập

```

15 // Lớp hoạt động đăng nhập
16 public class SigninActivity extends AppCompatActivity {
17
18     // Biến binding cho layout signin
19     private ActivitySigninBinding binding;
20
21     // Biến connector để kết nối với cơ sở dữ liệu SQLite
22     private SQLiteConnector sqliteConnector;
23
24     // Phương thức onCreate được gọi khi hoạt động được tạo
25     @Override
26     protected void onCreate(Bundle savedInstanceState) {
27         // Kích hoạt chế độ toàn màn hình
28         EdgeToEdge.enable(this);
29         super.onCreate(savedInstanceState);
30
31         // Inflate layout signin và set nội dung cho hoạt động
32         binding = ActivitySigninBinding.inflate(getLayoutInflater());
33         setContentView(binding.getRoot());
34
35         // Tạo đối tượng connector để kết nối với cơ sở dữ liệu SQLite
36         sqliteConnector = new SQLiteConnector(context, this);

```

```

38 // Đặt sự kiện click cho nút đăng nhập
39 binding.signInButton.setOnClickListener(view -> {
40     // Lấy giá trị của trường tên đăng nhập và mật khẩu
41     String username = binding.signInUser.getText().toString();
42     String password = binding.signInPassword.getText().toString();
43
44     // Kiểm tra xem cả hai trường đều có giá trị hay không
45     if (username.isEmpty() || password.isEmpty()) {
46         // Hiển thị thông báo lỗi nếu có trường nào đó trống
47         Toast.makeText(context, this, text: "All fields are mandatory", Toast.LENGTH_SHORT).show();
48     } else {
49         // Kiểm tra thông tin đăng nhập với cơ sở dữ liệu
50         boolean checkCredentials = sqlLiteConnector.checkUser(username, password);
51
52         // Nếu thông tin đăng nhập hợp lệ
53         if (checkCredentials) {
54             // Hiển thị thông báo thành công
55             Toast.makeText(context, this, text: "Sign in successfully", Toast.LENGTH_SHORT).show();
56             // Tạo intent để chuyển đến MainActivity
57             Intent intent = new Intent(packageContext, this, MainActivity.class);
58             // Đưa tên đăng nhập vào intent
59             intent.putExtra(name: "UserName", username);
60             // Khởi động hoạt động chính
61             startActivity(intent);
62         } else {
63             // Hiển thị thông báo lỗi nếu thông tin đăng nhập không hợp lệ
64             Toast.makeText(context, this, text: "Invalid Credentials", Toast.LENGTH_SHORT).show();
65         }
66     }
67 });

```

```

69 // Đặt sự kiện click cho văn bản đăng ký
70 binding.SignUpRedirectText.setOnClickListener(view -> {
71     // Tạo intent để chuyển đến hoạt động đăng ký
72     Intent intent = new Intent(packageContext, this, SignupActivity.class);
73     // Khởi động hoạt động đăng ký
74     startActivity(intent);
75 });
76
77 // Đặt sự kiện áp dụng các khoảng cách cho layout
78 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
79     // Lấy các khoảng cách của hệ thống
80     Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
81     // Áp dụng các khoảng cách cho layout
82     v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
83     return insets;
84 });
85 }
86 }

```

Mã nguồn Java cho chức năng đăng ký


```
public class SignupActivity extends AppCompatActivity {  
    16  
    17    // Biến binding để truy cập các thành phần trong layout  
    8 usages  
    18    private ActivitySignupBinding binding;  
    19  
    20    // Biến connector để thao tác với cơ sở dữ liệu SQLite  
    3 usages  
    21    private SQLiteConnector sqliteConnector;  
    22  
    23    *hoanh123  
    24    @Override  
    25    protected void onCreate(Bundle savedInstanceState) {  
    26        // Kích hoạt chế độ toàn màn hình  
    EdgeToEdge.enable( $this$enableEdgeToEdge: this);  
    27        super.onCreate(savedInstanceState);  
    28  
    29        // Tạo binding để truy cập các thành phần trong layout  
    30        binding = ActivitySignupBinding.inflate(getLayoutInflater());  
    31        setContentView(binding.getRoot());  
    32  
    33        // Tạo connector để thao tác với cơ sở dữ liệu SQLite  
    34        sqliteConnector = new SQLiteConnector( context: this);  
    35  
    36        // Đặt sự kiện click cho nút đăng ký  
    37        binding.signupButton.setOnClickListener(v -> {  
    38            // Lấy giá trị của các trường nhập liệu  
    39            String username = binding.signupUser.getText().toString();  
    40            String email = binding.signupEmail.getText().toString();  
    41            String password = binding.signupPassword.getText().toString();  
    42            String confirmPassword = binding.signupConfirm.getText().toString();
```

```

44 // Kiểm tra xem tất cả các trường nhập liệu có được điền đầy đủ không
45 if (username.isEmpty() || email.isEmpty() || password.isEmpty() || confirmPassword.isEmpty()) {
46     // Nếu có trường nào đó chưa được điền, hiển thị thông báo lỗi
47     Toast.makeText(context, SignupActivity.this, text: "All fields are mandatory", Toast.LENGTH_SHORT).show();
48 } else {
49     // Kiểm tra xem mật khẩu và xác nhận mật khẩu có trùng nhau không
50 if (password.equals(confirmPassword)) {
51     // Kiểm tra xem email đã được đăng ký chưa
52 if (!sqliteConnector.checkUser(email)) {
53     // Tạo đối tượng User mới
54     User user = new User();
55     user.setName(username);
56     user.setEmail(email);
57     user.setPassword(password);
58
59     // Thêm user vào cơ sở dữ liệu
60     sqliteConnector.addUser(user);
61
62     // Hiển thị thông báo thành công
63     Toast.makeText(context, SignupActivity.this, text: "Signup Successfully", Toast.LENGTH_SHORT).show();
64
65     // Chuyển sang màn hình đăng nhập
66     Intent intent = new Intent(packageContext, SignupActivity.this, SigninActivity.class);
67     startActivity(intent);
68 } else {
69     // Hiển thị thông báo lỗi nếu email đã được đăng ký
70     Toast.makeText(context, SignupActivity.this, text: "User already exists. Please sign in", Toast.LENGTH_SHORT).show();
71 }
72 } else {
73     // Hiển thị thông báo lỗi nếu mật khẩu và xác nhận mật khẩu không trùng nhau
74     Toast.makeText(context, SignupActivity.this, text: "Invalid password", Toast.LENGTH_SHORT).show();
75 }
76 }
77 });

```

```

80 // Đặt sự kiện click cho nút chuyển sang màn hình đăng nhập
81 binding.SignInRedirectText.setOnClickListener(v -> {
82     // Chuyển sang màn hình đăng nhập
83     Intent intent = new Intent(packageContext, SignupActivity.this, SigninActivity.class);
84     startActivity(intent);
85 });
86
87 // Đặt sự kiện áp dụng các khoảng cách cho màn hình
88 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
89     // Lấy các khoảng cách của màn hình
90     Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
91     // Áp dụng các khoảng cách cho màn hình
92     v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
93     return insets;
94 });
95 }

```

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

```

1 package com.example.group4_app;
2 import java.security.MessageDigest;
3 import java.security.NoSuchAlgorithmException;
4
5 2 usages
6 public class PasswordHasher {
7     2 usages
8     @ public static String hashPassword(String password) {
9         try {
10             // Create MessageDigest instance for SHA-256
11             MessageDigest md = MessageDigest.getInstance("SHA-256");
12
13             // Add password bytes to digest
14             md.update(password.getBytes());
15
16             // Get the hash's bytes
17             byte[] bytes = md.digest();
18
19             // Convert bytes to hexadecimal format
20             StringBuilder sb = new StringBuilder();
21             for (byte aByte : bytes) {
22                 sb.append(Integer.toString((aByte & 0xff) + 0x100, radix: 16).substring(beginIndex: 1));
23             }
24
25             // Get complete hashed password in hexadecimal format
26             return sb.toString();
27         } catch (NoSuchAlgorithmException e) {
28             e.printStackTrace();
29             return null;
30         }
31     }
32 }

```

```

public void addUser(User user) {
    SQLiteDatabase db = this.getWritableDatabase();

    ContentValues values = new ContentValues();
    values.put(COLUMN_USER_NAME, user.getName());
    values.put(COLUMN_USER_EMAIL, user.getEmail());
    // Hash the password before storing it
    String hashedPassword = PasswordHasher.hashPassword(user.getPassword());
    values.put(COLUMN_USER_PASSWORD, hashedPassword);

    // Inserting Row
    db.insert(TABLE_USER, nullColumnHack: null, values);
    db.close();
}

```

```

public boolean checkUser(String username, String password) {

    // array of columns to fetch
    String[] columns = {
        COLUMN_USER_ID
    };

    SQLiteDatabase db = this.getReadableDatabase();
    // Hash the password
    String hashedPassword = PasswordHasher.hashPassword(password);
    // selection criteria
    String selection = COLUMN_USER_NAME + " = ?" + " AND " + COLUMN_USER_PASSWORD + " = ?";

    // selection arguments
    String[] selectionArgs = {username, hashedPassword};

    // query user table with conditions
    /**
     * Here query function is used to fetch records from user table this function works like we use sql query.
     * SQL query equivalent to this query function is
     * SELECT user_id FROM user WHERE user_email = 'jack@androidtutorialshub.com' AND user_password = 'qwerty';
     */
    Cursor cursor = db.query(TABLE_USER, //Table to query
        columns, //columns to return
        selection, //columns for the WHERE clause
        selectionArgs, //The values for the WHERE clause
        null, //group the rows
        null, //filter by row groups
        null); //The sort order

    int cursorCount = cursor.getCount();

    cursor.close();
    db.close();
    if (cursorCount > 0) {
        return true;
    }
}

```

Yêu cầu 5. Tạo một cơ sở dữ liệu tương tự bên ngoài thiết bị, viết mã nguồn thực hiện kết nối đến CSDL này để truy vấn thay vì sử dụng SQLite.

- Ta tạo database trên mysql

Server: 127.0.0.1 » Database: usermanager » Table: user

Browse Structure SQL Search Insert Export Import Privileges Operations Trigger

Showing rows 0 - 3 (4 total, Query took 0.0002 seconds.) [user_email: 2... - ABCD2...]

SELECT * FROM `user` ORDER BY `user_email` ASC

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	user_id	user_name	user_email	user_password
<input type="checkbox"/> Edit Copy Delete	3	1	2	3
<input type="checkbox"/> Edit Copy Delete	4	a	a@gmail.com	1
<input type="checkbox"/> Edit Copy Delete	1	abcd	abcd	abcd
<input type="checkbox"/> Edit Copy Delete	2	abcd1	abcd2	abcd

☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

- Lần lượt tạo các code để checkUser.php, checkEmail.php, addUser.php và php để kết nối tới cơ sở dữ liệu

File Explorer view of the directory: This PC > OS (C:) > xampp > htdocs > platform

Name	Date modified	Type	Size
addUser.php	5/15/2024 9:56 PM	PHP Source File	2 KB
checkEmail.php	5/15/2024 10:17 PM	PHP Source File	1 KB
checkUser.php	5/15/2024 10:14 PM	PHP Source File	1 KB
dbConnect.php	5/15/2024 9:50 PM	PHP Source File	1 KB

Code editor view of dbConnect.php:

```
1 <?php
2 $servername = "localhost";
3 $username = "root";
4 $password = "";
5 $dbname = "usermanager";
6
7 $conn = mysqli_connect($servername, $username, $password, $dbname);
8
9 if (!$conn) {
10     die("Connection failed: " . mysqli_connect_error());
11 }
12
13
```

```

1  <?php
2  include 'dbConnect.php';
3
4  $email = $_POST['email'];
5
6  $query = "SELECT * FROM user WHERE user_email = '$email'";
7  $result = mysqli_query($conn, $query);
8
9  if(mysqli_num_rows($result) > 0) {
10     echo json_encode(array("success" => true));
11 } else {
12     echo json_encode(array("success" => false));
13 }
14
15 mysqli_close($conn);
16
17

```

```

1  <?php
2  include 'dbConnect.php';
3
4  $username = $_POST['username'];
5  $password = $_POST['password'];
6
7  $query = "SELECT * FROM user WHERE user_name = '$username' AND user_password = '$password'";
8  $result = mysqli_query($conn, $query);
9
10 if(mysqli_num_rows($result) > 0) {
11     echo json_encode(array("success" => true));
12 } else {
13     echo json_encode(array("success" => false));
14 }
15 echo json_encode($response);
16 mysqli_close($conn);
17

```

- Ta tạo MySQLConnector.java

```

4 usages
17 public class MySQLConnector {
    3 usages
18     private static final String BASE_URL = "http://172.31.0.2/platform/";
    4 usages
19     private RequestQueue requestQueue;
20
    2 usages
21     public MySQLConnector(RequestQueue requestQueue) {
22         this.requestQueue = requestQueue;
23     }
24
    // Hàm kiểm tra thông tin đăng nhập
    1 usage
26     public void checkUser(String username, String password, final VolleyCallback callback) {
27         String url = BASE_URL + "checkUser.php";
28         StringRequest stringRequest = new StringRequest(Request.Method.POST, url,
29             new Response.Listener<String>() {
30                 @Override
31                 public void onResponse(String response) {
32                     try {
33                         JSONObject jsonObject = new JSONObject(response);
34                         boolean success = jsonObject.getBoolean("name: success");
35                         callback.onSuccess(success);
36                     } catch (JSONException e) {
37                         e.printStackTrace();
38                         callback.onSuccess(result: false);
39                     }
40                 }
41             },
42             new Response.ErrorListener() {
no usages
                @Override
                public void onErrorResponse(VolleyError error) {
                    error.printStackTrace();
                    callback.onSuccess(result: false);
                }
            }) {
    2 usages
    @Override
    protected Map<String, String> getParams() {
        Map<String, String> params = new HashMap<>();
        params.put("username", username);
        params.put("password", password);
        return params;
    }
};
requestQueue.add(stringRequest);
}

// Hàm thêm user vào cơ sở dữ liệu
1 usage
public void addUser(User user, final VolleyCallback callback) {
    String url = BASE_URL + "addUser.php";
    StringRequest stringRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONObject jsonObject = new JSONObject(response);
                    boolean success = jsonObject.getBoolean("name: success");
                    callback.onSuccess(success);
                } catch (JSONException e) {
                    e.printStackTrace();
                    callback.onSuccess(result: false);
                }
            }
        }
    );
    requestQueue.add(stringRequest);
}

```



```

    },
    new Response.ErrorListener() {
        no usages
        @Override
        public void onErrorResponse(VolleyError error) {
            error.printStackTrace();
            callback.onSuccess(result: false);
        }
    }) {
        @Override
        protected Map<String, String> getParams() {
            Map<String, String> params = new HashMap<>();
            params.put("username", user.getName());
            params.put("email", user.getEmail());
            params.put("password", user.getPassword());
            return params;
        }
    };
    requestQueue.add(stringRequest);
}

// Hàm kiểm tra email đã được đăng ký chưa
no usages
public void checkEmail(String email, final VolleyCallback callback) {
    String url = BASE_URL + "checkEmail.php";
    StringRequest stringRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONObject jsonObject = new JSONObject(response);
                    boolean success = jsonObject.getBoolean(name: "success");
                    callback.onSuccess(success);
                } catch (JSONException e) {

```

```

public void checkEmail(String email, final VolleyCallback callback) {
    String url = BASE_URL + "checkEmail.php";
    StringRequest stringRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    JSONObject jsonObject = new JSONObject(response);
                    boolean success = jsonObject.getBoolean("success");
                    callback.onSuccess(success);
                } catch (JSONException e) {
                    e.printStackTrace();
                    callback.onSuccess(result: false);
                }
            }
        },
        new Response.ErrorListener() {
            no usages
            @Override
            public void onErrorResponse(VolleyError error) {
                error.printStackTrace();
                callback.onSuccess(result: false);
            }
        }) {
        @Override
        protected Map<String, String> getParams() {
            Map<String, String> params = new HashMap<>();
            params.put("email", email);
            return params;
        }
    };
    requestQueue.add(stringRequest);
}

protected Map<String, String> getParams() {
    Map<String, String> params = new HashMap<>();
    params.put("email", email);
    return params;
}

};
requestQueue.add(stringRequest);
}

// Interface callback để xử lý kết quả trả về từ yêu cầu Volley
3 usages
public interface VolleyCallback {
    9 usages
    void onSuccess(boolean result);
}
}

```

- SignIn

```
e (group4_app)  AndroidManifest.xml  MySQLConnector.java  User.java  PasswordHasher.java  SignInActivity.java x v
1 package com.example.group4_app;
2
3 > import ...
11
12 public class SignInActivity extends AppCompatActivity {
13
14     6 usages
15     private ActivitySignInBinding binding;
16     2 usages
17     private MySQLConnector mySQLConnector;
18
19     @Override
20     protected void onCreate(Bundle savedInstanceState) {
21         super.onCreate(savedInstanceState);
22         binding = ActivitySignInBinding.inflate(getLayoutInflater());
23         setContentView(binding.getRoot());
24
25         // Creating RequestQueue with Volley
26         mySQLConnector = new MySQLConnector(Volley.newRequestQueue( context: this));
27
28         binding.signInButton.setOnClickListener(view -> {
29             String username = binding.signInUser.getText().toString();
30             String password = binding.signInPassword.getText().toString();
31
32             if (username.isEmpty() || password.isEmpty()) {
33                 Toast.makeText( context: this, text: "All fields are mandatory", Toast.LENGTH_SHORT).show();
34             } else {
35                 mySQLConnector.checkUser(username, password, result -> {
36                     if (result) {
37                         Toast.makeText( context: SignInActivity.this, text: "Sign in successfully", Toast.LENGTH_SHORT).show();
38                         Intent intent = new Intent( packageContext: SignInActivity.this, MainActivity.class);
39                         intent.putExtra( name: "UserName", username);
40                         startActivity(intent);
41                     } else {
42                         Toast.makeText( context: SignInActivity.this, text: "Invalid Credentials", Toast.LENGTH_SHORT).show();
43                     }
44                 });
45             }
46         });
47     }
48 }
```

```

31         Toast.makeText( context: this, text: "All fields are mandatory", Toast.LENGTH_SHORT).show();
32     } else {
33         mySQLConnector.checkUser(username, password, result -> {
34             if (result) {
35                 Toast.makeText( context: SigninActivity.this, text: "Sign in successfully", Toast.LENGTH_SHORT).show();
36                 Intent intent = new Intent( packageContext: SigninActivity.this, MainActivity.class);
37                 intent.putExtra( name: "UserName", username);
38                 startActivity(intent);
39             } else {
40                 Toast.makeText( context: SigninActivity.this, text: "Invalid Credentials", Toast.LENGTH_SHORT).show();
41             }
42         });
43     }
44 }
45
46 binding.SignUpRedirectText.setOnClickListener(view -> {
47     Intent intent = new Intent( packageContext: this, SignupActivity.class);
48     startActivity(intent);
49 });
50 }
51 }
52

```

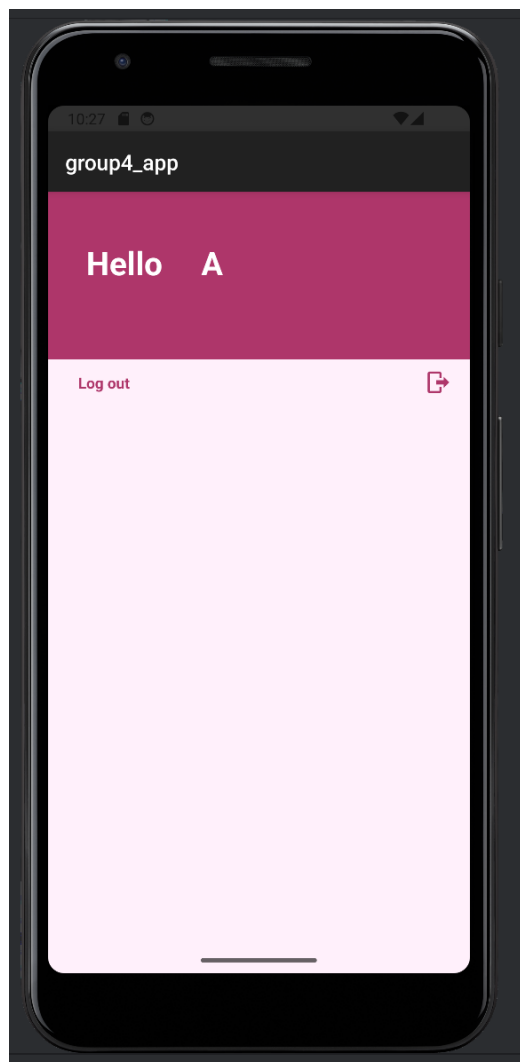
- Code SignUp

```



3  > import ...
15
16 public class SignupActivity extends AppCompatActivity {
17
18     8 usages
19     private ActivitySignupBinding binding;
20
21     @Override
22     protected void onCreate(Bundle savedInstanceState) {
23         super.onCreate(savedInstanceState);
24
25         binding = ActivitySignupBinding.inflate(getLayoutInflater());
26         setContentView(binding.getRoot());
27
28         RequestQueue requestQueue = Volley.newRequestQueue( context: this);
29         MySQLConnector mySQLConnector = new MySQLConnector(requestQueue);
30
31         binding.signupButton.setOnClickListener(v -> {
32             String username = binding.signupUser.getText().toString();
33             String email = binding.signupEmail.getText().toString();
34             String password = binding.signupPassword.getText().toString();
35             String confirmPassword = binding.signupConfirm.getText().toString();
36
37             if (username.isEmpty() || email.isEmpty() || password.isEmpty() || confirmPassword.isEmpty()) {
38                 Toast.makeText( context: this, text: "All fields are mandatory", Toast.LENGTH_SHORT).show();
39             } else {
40                 if (password.equals(confirmPassword)) {
41                     mySQLConnector.addUser(new User(username, email, password), result -> {
42                         if (result) {
43                             Toast.makeText( context: this, text: "Signup Successfully", Toast.LENGTH_SHORT).show();
44                             Intent intent = new Intent( packageContext: this, SigninActivity.class);
45                             startActivity(intent);
46                         } else {
47                             Toast.makeText( context: this, text: "Signup Failed", Toast.LENGTH_SHORT).show();
48                         }
49                     });
50                 } else {
51                     Toast.makeText( context: this, text: "Password does not match", Toast.LENGTH_SHORT).show();
52                 }
53             }
54         });
55     }
56 }
57

```



















```
48         });
49     } else {
50         Toast.makeText(context: this, text: "Passwords do not match", Toast.LENGTH_SHORT).show();
51     }
52 }
53 };
54
55 binding.SignInRedirectText.setOnClickListener(v -> {
56     Intent intent = new Intent(packageContext: this, SigninActivity.class);
57     startActivity(intent);
58 });
59
60 ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
61     Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
62     v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
63     return insets;
64 });
65 }
66 }
67 }
```



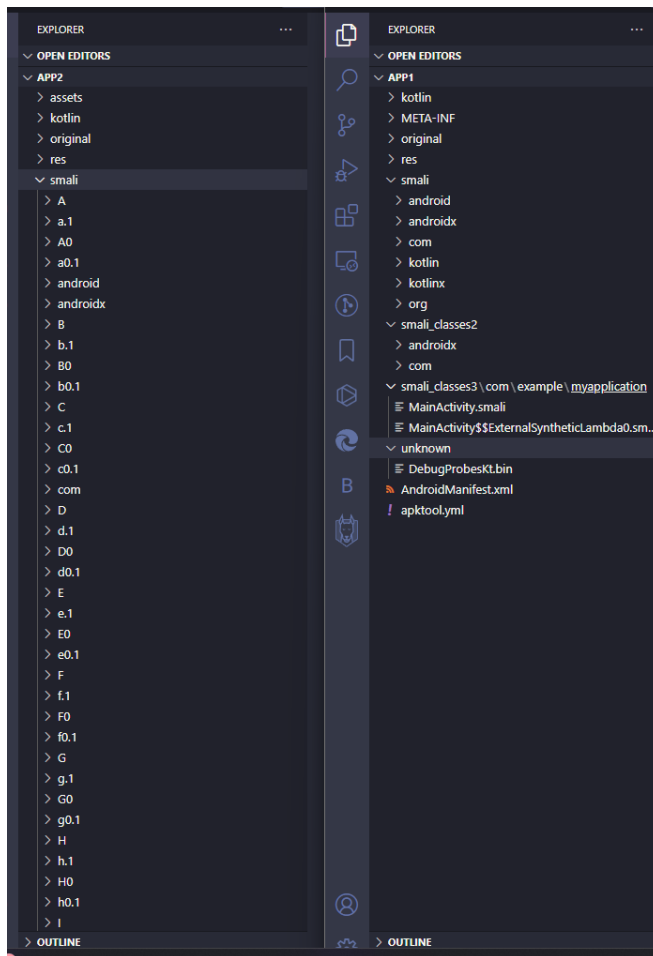
Yêu cầu 6. Với ứng dụng đã xây dựng, tìm hiểu và sử dụng công cụ ProGuard để tối ưu hóa mã nguồn. Trình bày khác biệt trước và sau khi sử dụng?

 app2	5/15/2024 2:35 PM	BlueStacks Androi...	1,617 KB
 app1	5/15/2024 2:11 PM	BlueStacks Androi...	5,506 KB

Sau khi sử dụng ProGuard thì dung lượng file đã giảm đi

Name	Date modified	Type	Size		Name	Date modified	Type	Size	
▼ Today					▼ Today				
 apktool	5/15/2024 2:41 PM	Yaml Source File	1 KB		 apktool	5/15/2024 2:20 PM	Yaml Source File	3 KB	
 AndroidManifest	5/15/2024 2:41 PM	XML Source File	3 KB		 AndroidManifest	5/15/2024 2:20 PM	XML Source File	3 KB	
 assets	5/15/2024 2:41 PM	File folder			 META-INF	5/15/2024 2:20 PM	File folder		
 kotlin	5/15/2024 2:41 PM	File folder			 kotlin	5/15/2024 2:20 PM	File folder		
 original	5/15/2024 2:41 PM	File folder			 original	5/15/2024 2:20 PM	File folder		
 smali	5/15/2024 2:41 PM	File folder			 smali	5/15/2024 2:20 PM	File folder		
 unknown	5/15/2024 2:41 PM	File folder			 smali_classes2	5/15/2024 2:20 PM	File folder		
 res	5/15/2024 2:41 PM	File folder			 smali_classes3	5/15/2024 2:20 PM	File folder		
					 unknown	5/15/2024 2:20 PM	File folder		
					 res	5/15/2024 2:20 PM	File folder		

Các thư mục cũng ít hơn



Các file được xóa bớt

```

! apktool.yml
! apktool.yml
1 version: 2.9.3
2 apkFileName: app2.apk
3 isFrameworkApk: false
4 usesFramework:
5   ids:
6     - 1
7   tag: null
8 sdkInfo:
9   minSdkVersion: 26
10  targetSdkVersion: 34
11 packageInfo:
12   forcedPackageId: 127
13   renameManifestPackage: null
14 versionInfo:
15   versionCode: 1
16   versionName: 1.0
17 resourcesAreCompressed: false
18 sharedLibrary: false
19 sparseResources: false
20 unknownFiles:
21   DebugProbesKt.bin: 8
22 doNotCompress:
23   - resources.arsc
24   - assets/dexopt/baseline.prof
25   - assets/dexopt/baseline.profm
26   - webp
27   - png
28
! apktool.yml
! apktool.yml
1 version: 2.9.3
2 apkFileName: app1.apk
3 isFrameworkApk: false
4 usesFramework:
5   ids:
6     - 1
7   tag: null
8 sdkInfo:
9   minSdkVersion: 26
10  targetSdkVersion: 34
11 packageInfo:
12   forcedPackageId: 127
13   renameManifestPackage: null
14 versionInfo:
15   versionCode: 1
16   versionName: 1.0
17 resourcesAreCompressed: false
18 sharedLibrary: false
19 sparseResources: false
20 unknownFiles:
21   DebugProbesKt.bin: 8
22 doNotCompress:
23   - resources.arsc
24   - META-INF/androidx.activity_activity.version
25   - META-INF/androidx.annotation_annotation-experimental.vers
26   - META-INF/androidx.appcompat_appcompat-resources.version
27   - META-INF/androidx.appcompat_appcompat.version
28   - META-INF/androidx.cardview_cardview.version
29   - META-INF/androidx.coordinatorlayout_coordinatorlayout.ver
30   - META-INF/androidx.core_core-ktx.version
31   - META-INF/androidx.core_core.version
32   - META-INF/androidx.cursoradapter_cursoradapter.version
33   - META-INF/androidx.customview_customview.version
34   - META-INF/androidx.documentfile_documentfile.version
35   - META-INF/androidx.drawerlayout_drawerlayout.version
36   - META-INF/androidx.dynamicanimation_dynamicanimation.versi
37   - META-INF/androidx.emoji2_emoji2-views-helper.version
38   - META-INF/androidx.emoji2_emoji2.version
39   - META-INF/androidx.fragment_fragment.version
40   - META-INF/androidx.interpolator_interpolator.version
41   - META-INF/androidx.legacy_legacy-support-core-utils.versio
42   - META-INF/androidx.loader_loader.version
43   - META-INF/androidx.localbroadcastmanager_localbroadcastman
44   - META-INF/androidx.print_print.version
45   - META-INF/androidx.profileinstaller_profileinstaller.versi
46   - META-INF/androidx.recyclerview_recyclerview.version
47   - META-INF/androidx.savedstate_savedstate.version
48   - META-INF/androidx.startup_startup-runtime.version
49   - META-INF/androidx.tearing_tearing.version

```

Code được rút gọn đi

Rút gọn, làm rối mã nguồn và tối ưu hoá ứng dụng

- Rút gọn mã (hoặc loại bỏ mã chết (tree-shaking)): nhận diện và loại bỏ một cách an toàn các lớp, trường, phương thức và thuộc tính không sử dụng khỏi ứng dụng cũng như các phần phụ thuộc thư viện của ứng dụng. Đây là một công cụ hữu ích giúp tạm thời khắc phục giới hạn tham chiếu 64k. Ví dụ: nếu bạn chỉ sử dụng một số API của phần phụ thuộc thư viện, tính năng rút gọn này có thể nhận diện mã thư viện mà ứng dụng không sử dụng và xoá chính mã đó khỏi ứng dụng. Để tìm hiểu thêm, hãy tham khảo phần nội dung về cách rút gọn mã.
- Rút gọn tài nguyên: xoá tài nguyên không sử dụng khỏi ứng dụng đóng gói, bao gồm cả tài nguyên không sử dụng đến trong phần phụ thuộc thư viện của ứng dụng. Tính năng này được sử dụng kết hợp với tính năng rút gọn mã sao cho khi

xoá mã không sử dụng, những tài nguyên không được tham chiếu đến nữa cũng được xoá một cách an toàn. Để tìm hiểu thêm, hãy tham khảo phần nội dung về cách rút gọn tài nguyên.

- Tối ưu hoá: kiểm tra và viết lại mã để cải thiện hiệu suất trong thời gian chạy và giảm hơn nữa kích thước của các tệp DEX của ứng dụng. Điều này giúp cải thiện hiệu suất thời gian chạy của mã thêm đến 30%, cải thiện đáng kể thời gian khởi động và kết xuất khung hình. Ví dụ: nếu R8 phát hiện lệnh rẽ nhánh `else {}` của một câu lệnh `if/else` nào đó không bao giờ được sử dụng thì R8 sẽ xoá mã của lệnh rẽ nhánh `else {}` đó. Để tìm hiểu thêm, hãy tham khảo phần nội dung tối ưu hoá mã.
- Làm rối mã nguồn (hoặc giảm kích thước giá trị nhận dạng): rút ngắn tên của các lớp và thành phần, nhờ đó làm giảm kích thước tệp DEX. Để tìm hiểu thêm, hãy chuyển đến phần về cách làm rối mã nguồn.