

Configuration Manual

MSc Internship
MSc in Cyber Security

Sumana Ponnasamudra Boraiah
X18100147

School of Computing
National College of Ireland

Supervisor: Mr Ben Fletcher

National College of Ireland
MSc Project Submission Sheet
School of Computing



Student Name: Sumana Ponnasamudra Boraiah
Student ID: X18100147
Programme: MSc in Cyber Security **Year:** 2018/2019
Module: Academic Internship
Lecturer: Ben Fletcher
Submission Due Date: 15/12/19
Project Title: Secure Cardless Transaction Android Application using ECC Algorithm and QR code

Word Count: 423

Page Count: 8

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

Signature:

Date:

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Sumana Ponnasamudra Boraiah
X18100147

1 Introduction

The configuration manual provides the software and hardware configuration techniques of the “Secure Cardless Transaction Android Application using ECC Algorithm and QR code”. The secure application is an android application for handling the secure transaction at ATM. As a solution to the cardless transaction, the application includes the bank details to interact and follows with additional features. The code mentioned is to support the application features explanation.

2 Environment

2.1 Hardware Requirements

Operating System: Windows 7, 8 or the above version

Processor: Intel Core i5 8th Generation

RAM: 8 GB

Storage 256 GB SSD

Android Phone: minimum Android 5.0 Lollipop

2.2 Software

Android Studio is the integrated development environment to develop android applications. To develop creative GUI and functionalities, it provides editor tools and emulators with different versions. For implementing user interface, it uses XML tags and properties. The other functionality could be implemented using java programming by creating activity classes. The software is available to download in its official website- (“Download Android Studio and SDK tools,” n.d.)

3 Running/Accessing application

3.1 Installing Secure Application

Download the APK file of the application into smart phone. Run the APK file and allow the application to access phone gallery and internal storage.

4 Operation of the system

- The user must register with the application and continue to use the application.

The screenshot shows a registration form titled "Register" under the "My Application" header. It contains four input fields: "Name" with the value "sumana", "Mobile Number" with "0899634525", "Email" with "sumanapb12@gmail.com", and "Password" with masked characters ".....". A blue "SIGNUP" button is at the bottom right, and a link "Don't have an account? Sign Up" is below it.

(1)

The screenshot shows a main menu with two buttons: "Genrate KeyPair" and "User List".

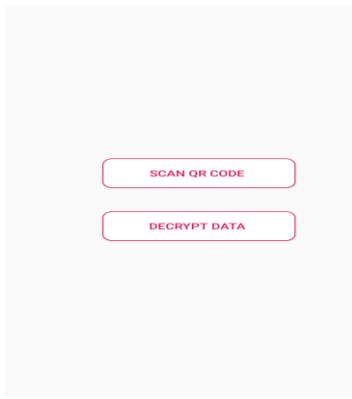
(2)

The screenshot shows the "HOME" screen with a list of users. Each user entry includes a profile icon, a name, and an email address. The first user is "kapil" with email "k Kapoor1392@gmail.com". The second user is "nayana" with email "nayanasadudra@gmail.com".

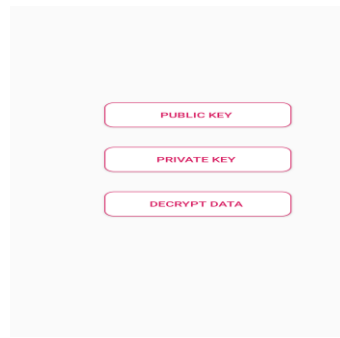
(3)

The screenshot shows a main menu with four buttons: "SMS", "QR CODE", "QR CODE SCAN", and "DOWNLOAD QR".

(4)



(5)



(6)

5 Code

The ECDSA and ECIES standards of ECC algorithm are implemented using java security APIs and SpongyCastle provider. The code implemented referring (262588213843476, n.d.) The folder structure of the software includes Manifest File, Java (includes java activity class files), res folder includes the layouts where GUI design files are coded using XML tags and properties. Gradle includes the whole application module, the required library must be included over here.

5.1 Key Generation

- This function used to generate the unique keypair to employ asymmetric encryption.

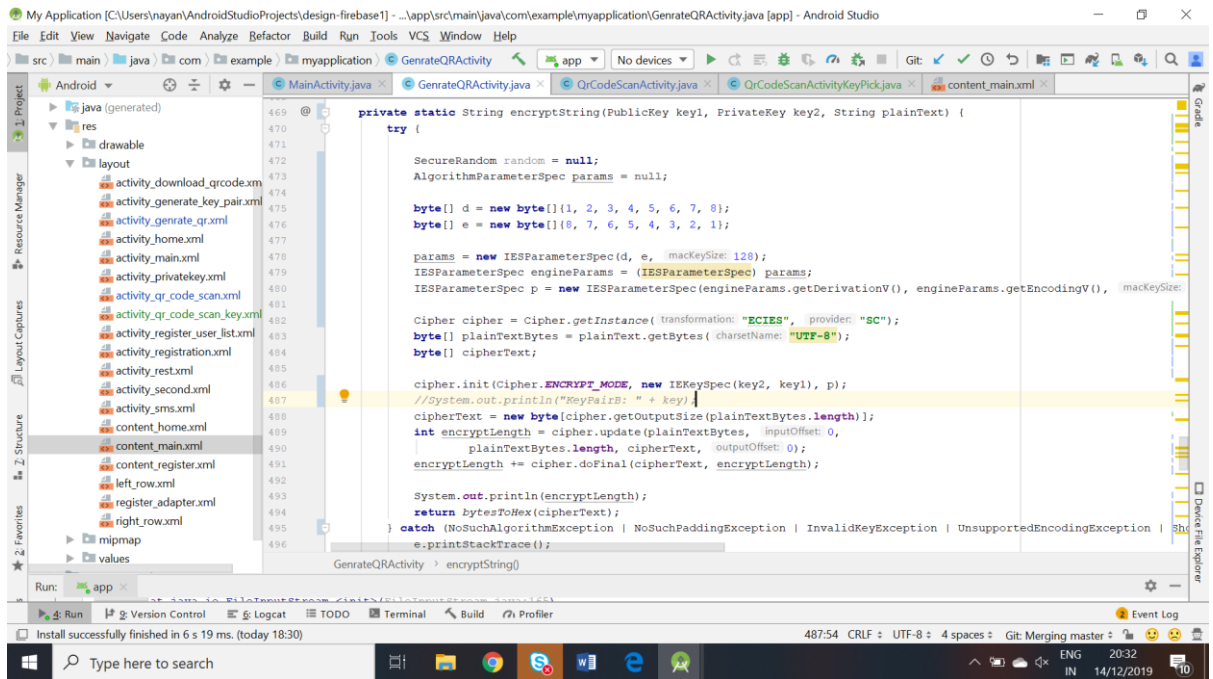
```

161
162
163 private static KeyPair generateECCKeys() {
164     try {
165         KeyPairGenerator kpg;
166         kpg = KeyPairGenerator.getInstance( algorithm: "ECDSA", provider: "SC");
167         ECGenParameterSpec ecsp;
168         ecsp = new ECGenParameterSpec( stdName: "secp192r1");
169         kpg.initialize(ecsp, new SecureRandom());
170
171         KeyPair kp = kpg.genKeyPair();
172         PrivateKey privateKey = kp.getPrivate();
173         PublicKey publicKey = kp.getPublic();
174
175         return kp;
176     } catch (NoSuchAlgorithmException | InvalidAlgorithmParameterException | NoSuchProviderException e) {
177         e.printStackTrace();
178         return null;
179     }
180 }
181
182
183
184

```

Figure 1: ECDSA Key generation

5.2 Encryption



The screenshot shows the Android Studio interface with the MainActivity.java file open. The code defines a static method encryptString that takes a PublicKey, a PrivateKey, and a String plaintext as input. It uses the ECIES encryption scheme with a SecureRandom instance and an IESParameterSpec. The method returns the encrypted string as a hex string.

```
private static String encryptString(PublicKey key1, PrivateKey key2, String plaintext) {
    try {
        SecureRandom random = null;
        AlgorithmParameterSpec params = null;

        byte[] d = new byte[] {1, 2, 3, 4, 5, 6, 7, 8};
        byte[] e = new byte[] {8, 7, 6, 5, 4, 3, 2, 1};

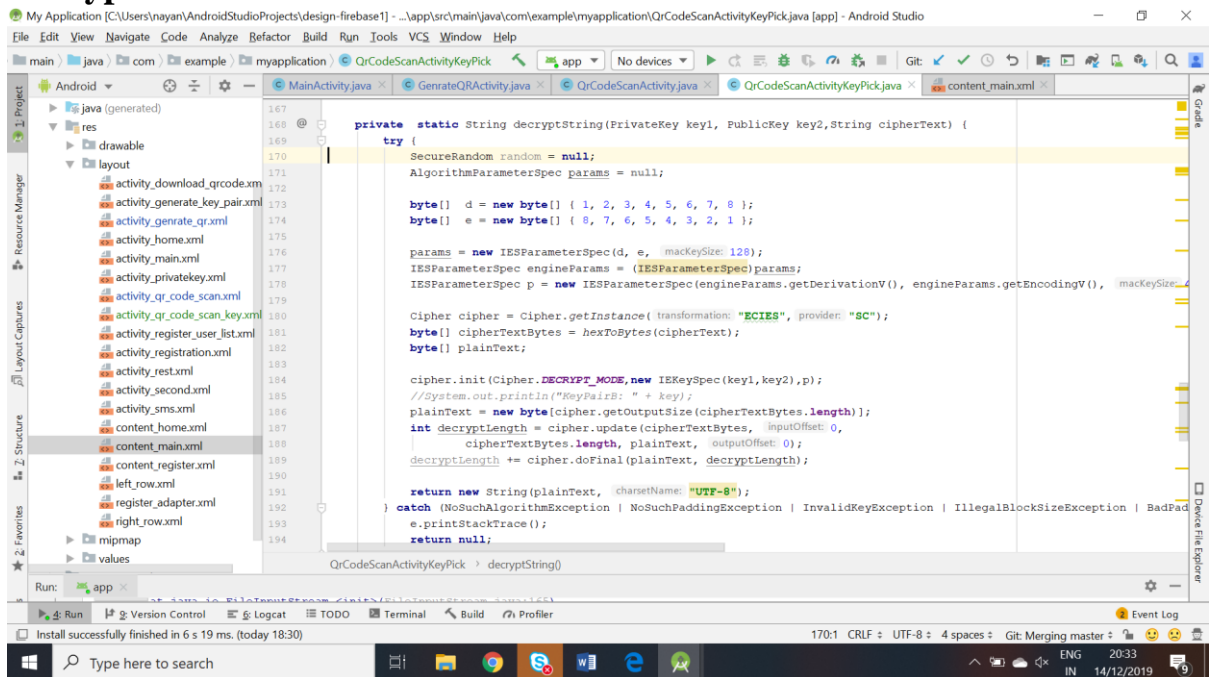
        params = new IESParameterSpec(d, e, macKeySize: 128);
        IESParameterSpec engineParams = (IESParameterSpec) params;
        IESParameterSpec p = new IESParameterSpec(engineParams.getDerivationV(), engineParams.getEncodingV(), macKeySize);

        Cipher cipher = Cipher.getInstance("ECIES", provider: "SC");
        byte[] plaintextBytes = plaintext.getBytes(charsetName: "UTF-8");
        byte[] cipherText;

        cipher.init(Cipher.ENCRYPT_MODE, new IESKeySpec(key2, key1), p);
        //System.out.println("KeyPair: " + key);
        cipherText = new byte[cipher.getOutputSize(plaintextBytes.length)];
        int encryptLength = cipher.update(plaintextBytes, inputOffset: 0,
            plaintextBytes.length, cipherText, outputOffset: 0);
        encryptLength += cipher.doFinal(cipherText, encryptLength);

        System.out.println(encryptLength);
        return bytesToHex(cipherText);
    } catch (NoSuchAlgorithmException | NoSuchPaddingException | InvalidKeyException | UnsupportedEncodingException | IOException) {
        e.printStackTrace();
    }
}
```

5.3 Decryption



The screenshot shows the Android Studio interface with the QrCodeScanActivityKeyPick.java file open. The code defines a static method decryptString that takes a PrivateKey, a PublicKey, and a String cipherText as input. It uses the ECIES decryption scheme with a SecureRandom instance and an IESParameterSpec. The method returns the decrypted string as a hex string.

```
private static String decryptString(PrivateKey key1, PublicKey key2, String cipherText) {
    try {
        SecureRandom random = null;
        AlgorithmParameterSpec params = null;

        byte[] d = new byte[] {1, 2, 3, 4, 5, 6, 7, 8};
        byte[] e = new byte[] {8, 7, 6, 5, 4, 3, 2, 1};

        params = new IESParameterSpec(d, e, macKeySize: 128);
        IESParameterSpec engineParams = (IESParameterSpec) params;
        IESParameterSpec p = new IESParameterSpec(engineParams.getDerivationV(), engineParams.getEncodingV(), macKeySize);

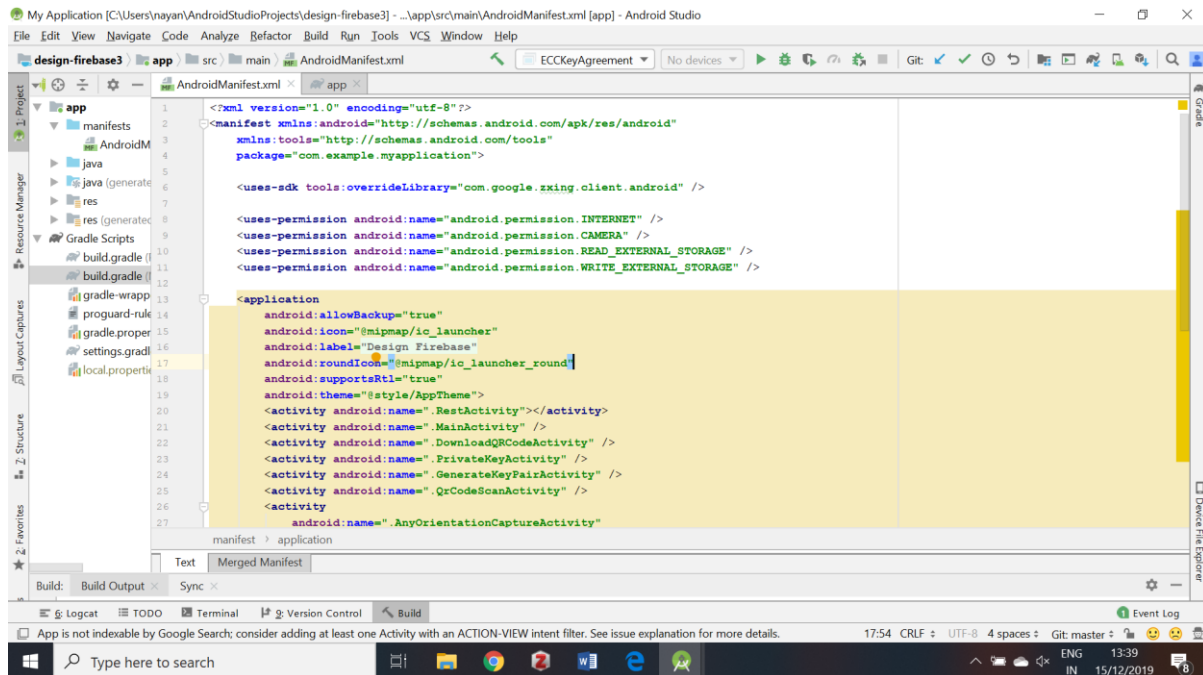
        Cipher cipher = Cipher.getInstance("ECIES", provider: "SC");
        byte[] cipherTextBytes = hexToBytes(cipherText);
        byte[] plaintext;

        cipher.init(Cipher.DECRYPT_MODE, new IESKeySpec(key1, key2), p);
        //System.out.println("KeyPair: " + key);
        plaintext = new byte[cipher.getOutputSize(cipherTextBytes.length)];
        int decryptLength = cipher.update(cipherTextBytes, inputOffset: 0,
            cipherTextBytes.length, plaintext, outputOffset: 0);
        decryptLength += cipher.doFinal(plaintext, decryptLength);

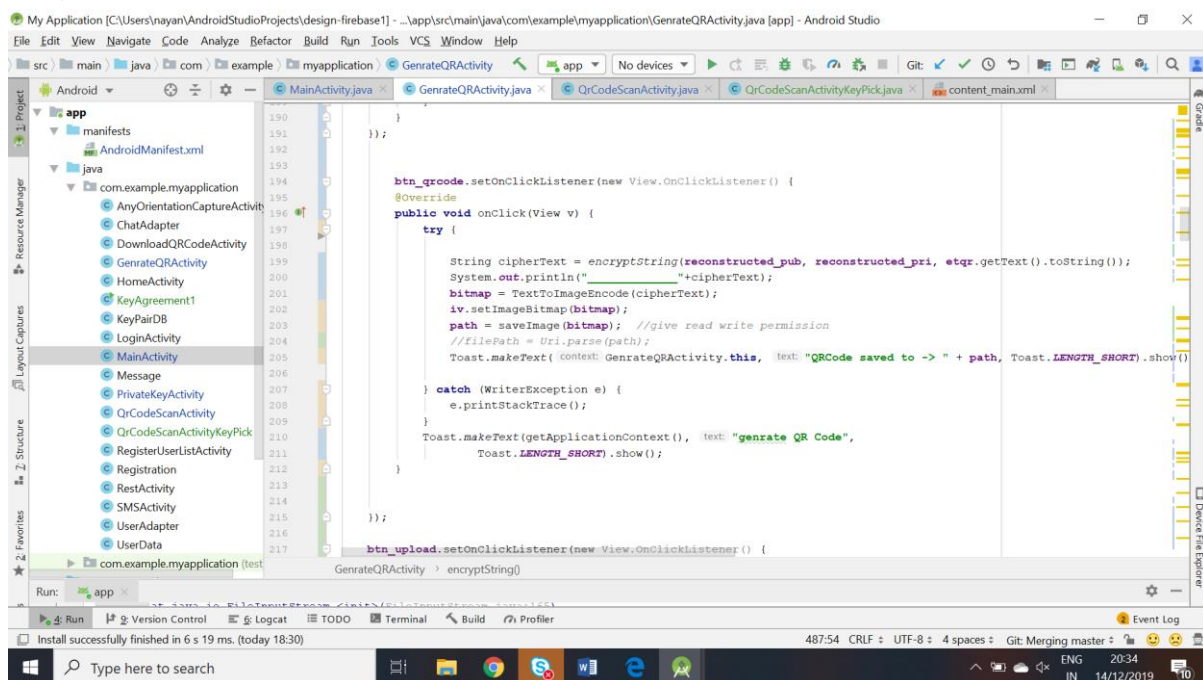
        return new String(plaintext, charsetName: "UTF-8");
    } catch (NoSuchAlgorithmException | NoSuchPaddingException | InvalidKeyException | IllegalBlockSizeException | BadPaddingException) {
        e.printStackTrace();
    }
}
```

5.4 Configuration File

- Android manifest file is the first file to execute when the application runs.
- In this file, all the activity classes used in application are registered in order to support redirection from one activity to another.
- The permission required from android device to the application for accessing the internal storage is registered here.



5.5 QR code



References

- 262588213843476, n.d. Encryption using Elliptic Curves and Diffie-Hellman key exchanges [WWW Document]. Gist. URL <https://gist.github.com/zcdziura/7652286> (accessed 12.15.19).
- Download Android Studio and SDK tools [WWW Document], n.d. . Android Dev. URL <https://developer.android.com/studio> (accessed 12.15.19).