

## **Project Description**

In this project, we wanted to implement a two-factor authentication mechanism for smart phones. First, user will enter username and password to login the system, then the system should identify the user biometrically. We are allowed to identify user with voice detection, face detection, sign detection etc.

The second authentication method of the project may be done by using cryptographic method that based on previously shared secret value and hash algorithms.

To sum up, we are asked to design a smartphone application that has two different authentication factor. The first one is username and password authentication, the second one is biometrically identifying the user of the phone.

## **Project Implementation**

In this part, we are going to tell what we did for this project. First, we discuss about which authentication methods will be used for the second factor. Then, we searched different types of biometric built-in api. After some research, we decided to use fingerprint method. The user must identify the fingerprint to the smartphone which means only the owner of the smartphone can be login the system. After deciding the authentication method, we setup software environment for this project.

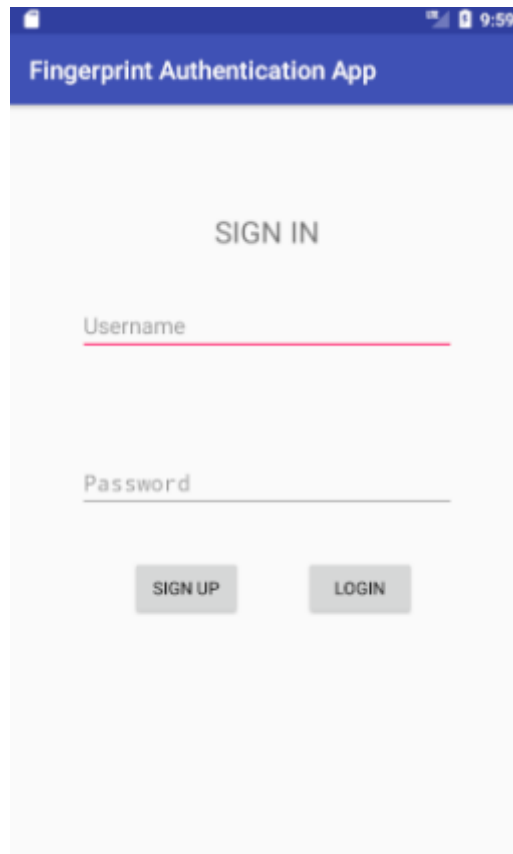
### **Software Environment**

IDE	Android Studio 2.3.3
JDK	Java JDK 1.8.0
Emulator	Android Virtual Device Manager
Devices	Samsung Galaxy J2 (Real Device)
	LG Nexus 5 (Virtual Device)
	Samsung S7
Operating System	Windows 10

We downloaded these software programs and installed all necessary things. After installation part, we read some documents, watched some tutorials about how to develop a smartphone application. We needed to know Java and XML language syntax to develop the project. Finally, we started to develop the project.

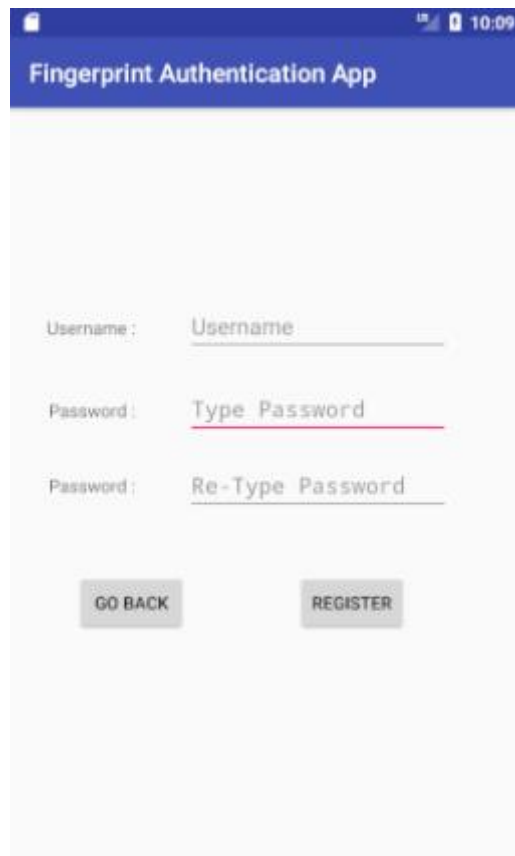
## **Application Develop Phases**

In this phase, we tried to design the application pages and layouts. Our application consists of 4 different pages. The first page is login screen. If the user have already registered the system then he or she can login, otherwise user can click the “SIGN UP” button to go the register page. You can see the login screen below:



Login button controls the entered username and password in the txt file. If user have already registered the system then the username and password are matched with the stored username and password in the txt file that named “userInformation.txt”.

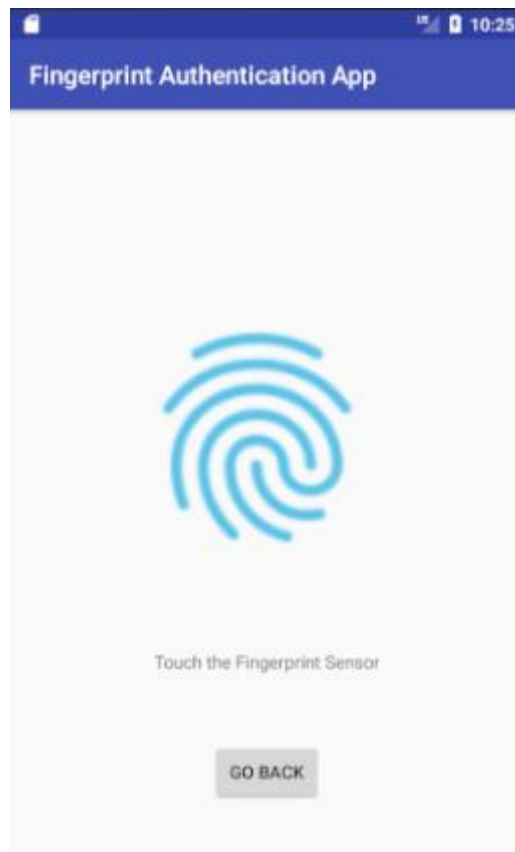
If the user want to register the system, then he or she should click the “SIGN UP” button to open the register page. You can see the register page below:



The screenshot shows a mobile application interface for a 'Fingerprint Authentication App'. The app's title is displayed in a blue header bar. The main content area is white and contains three input fields for registration: a 'Username' field with a placeholder 'Username', a 'Password' field with a placeholder 'Type Password' (which is underlined in red), and a second 'Password' field with a placeholder 'Re-Type Password'. At the bottom of the form, there are two buttons: 'GO BACK' and 'REGISTER'.

In the register page we want user to input username, password and password again. If some of these areas is blanked, then the user will see a message that indicate this area should be filled. If these areas are filled then user can register the system by clicking the “REGISTER” button. After registration the success message is shown to the user. We take these value and write them to the txt file so that we can use these information while user try to login the system. By the way, before writing these values to the txt file, we first send the entered password to the “encryptPassword” function. This function hashes the given string with “SHA-1” encryption method and returns us encrypted version of the string. So, we have the username and encrypted password, writing these strings to the end of the txt file.

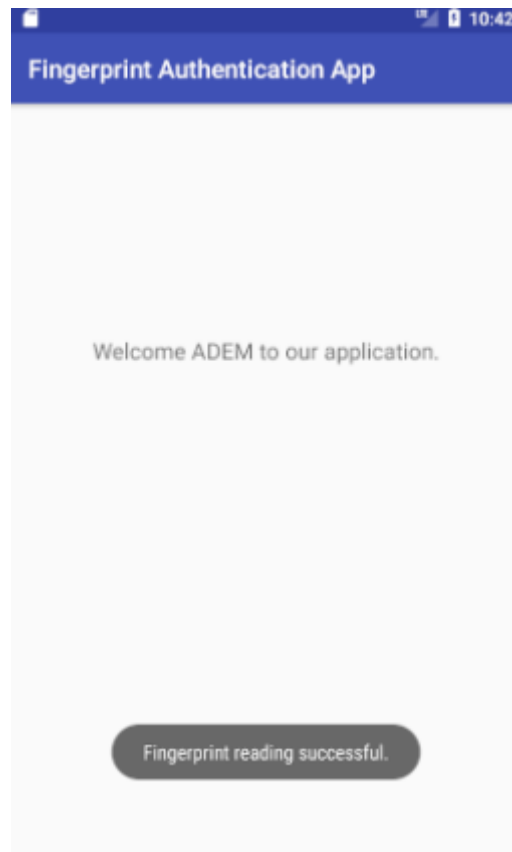
After registration, the user is directed to the main page that is login page so that the user can login the system. If the first authentication is passed successfully, then user will see second authentication page that asks user to touch for fingerprint. You can see this page below:



As you see we want user to touch the fingerprint sensor of the smartphone. In this page, if the smartphone has not a hardware sensor for the fingerprint then the program show an error message that indicate the smartphone has not fingerprint sensor, so user cannot progress anymore. The user fails second authentication method if the smartphone has not fingerprint sensor, so the user cannot access the system. Another error message will be shown if the smartphone has fingerprint sensor but the android release of that phone cannot support this fingerprint sensor. The user again cannot proceed the authentication and cannot access the system.

If the smartphone has a fingerprint sensor and the android release of the phone support this sensor, then user can touch this sensor. If the fingerprint matches with owner of the smartphone' fingerprint then the user passes the 2 factor authentication method and access the system. Otherwise, the program shows error message to user that indicate the fingerprints are not matched.

So far, we introduce our two-factor authentication mechanism. Finally, the user access the system and see the welcome message with its name. You can see the welcome page below:



Finally, the user access the system and see the welcome message. To sum up, we have created 4 different pages. First, the login page is created to login the system. Then, we developed the register page to register the system, also this phase is the first authentication method. Then we developed the second authentication method that wants user to touch the fingerprint sensor. Finally, we developed the welcome page for login successfully users to our system.

### **Application Codes**

We do not want to put all the code here since it will be really difficult for you to read and understand the codes and the pages will be more. Instead, we try to introduce necessary and important phase code of the project.

In the login page, we took the entered username and password then we try to match this information with the txt file. If the given information is found in the txt file, then user is directed to second authentication factor page. Otherwise, the user will be shown the error message that indicate "Login is failed.". You can see the login page code below:

```

public void onClick(View view){
    Intent SecondAuthenticationPage = new Intent(this, SecondAuthenticaton.class);

    EditText username = (EditText)findViewById(R.id.username);
    EditText password = (EditText)findViewById(R.id.password);

    String EnteredUsername = username.getText().toString();
    String EnteredPassword = password.getText().toString();

    try {
        BufferedReader br = new BufferedReader(new InputStreamReader(this.openFileInput("userInformation.txt")));
        String line;

        boolean isFound = false;
        while ((line = br.readLine()) != null) {
            String[] split = line.split("\\|");

            if(split[0].equals(EnteredUsername) && split[1].equals(EncryptionHelper.encryptPassword(EnteredPassword))){
                isFound = true;
                break;
            }
            else{
                // Toast.makeText(this, split[0] + " - " + EncryptionHelper.encryptPassword(EnteredPassword), Toast.LENGTH_SHORT).
                continue;
            }
        }

        if(!isFound){
            Toast.makeText(this, "Login is failed!", Toast.LENGTH_SHORT).show();
        }
        else{
            SecondAuthenticationPage.putExtra("WhoAmI", EnteredUsername);
            startActivity(SecondAuthenticationPage);
        }
    }
    catch (Exception ex){
        ex.printStackTrace();
    }
}

```

In the registration page, we took information entered by user and write them to the txt file. Before writing this information, we pass the entered password to the function that encrypt password and return us encrypted format. Then we write this information to the txt file. You can see the registration code below:

```

private void writeToFile(String username, String password, Context context){

    try {
        Writer writer = new BufferedWriter(new OutputStreamWriter(
            this.openFileOutput("userInformation.txt", MODE_APPEND), "utf-8"));

        writer.write(String.format("%s|%s\n", username, EncryptionHelper.encryptPassword(password)));
        writer.close();
    }
    catch (IOException e) {
        Toast.makeText(this, "Error: " + e.getMessage(), Toast.LENGTH_SHORT).show();
    }
}

```

As you see, we send the password to the encryptPassword function and write this username and encrypted password to the end of the txt file. You can see the EncryptionHelper class and its functions below:

```

public class EncryptionHelper {

    public static String encryptPassword(String password)
    {
        String sha1 = "";
        try
        {
            MessageDigest crypt = MessageDigest.getInstance("SHA-1");
            crypt.reset();
            crypt.update(password.getBytes("UTF-8"));
            sha1 = byteToHex(crypt.digest());
        }
        catch (NoSuchAlgorithmException e)
        {
            e.printStackTrace();
        }
        catch (UnsupportedEncodingException e)
        {
            e.printStackTrace();
        }

        catch (Exception ex) {

        }
        return sha1;
    }

    public static String byteToHex(final byte[] hash)
    {
        Formatter formatter = new Formatter();
        for (byte b : hash)
        {
            formatter.format("%02x", b);
        }
        String result = formatter.toString();
        formatter.close();
        return result;
    }
}

```

Until now, we explained the login and register page codes. After registration and login process, the user will access the second authentication page that wants user to touch the fingerprint sensor. You can see the code that belong the second authentication page below:

```

public String user;

FingerprintAuthHelper mFingerprintAuthHelper;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_second_authentication);

    Bundle ComingData = getIntent().getExtras();
    if(ComingData == null){
        return;
    }
    user = ComingData.getString("WhoAmI");
    // finger auth.
    mFingerprintAuthHelper = FingerprintAuthHelper.getHelper(this, this);
}

```

If the page is accessed, then we create the fingerprint helper method that is used the second authentication method. In case of other failure such as the device cannot have a fingerprint sensor we added this kind of controls in this page. You can see the some control of the fingerprint helper code below.

```

@Override
public void onNoFingerprintHardwareFound() {
    Toast.makeText(this, "Your device does not have fingerprint sensor.", Toast.LENGTH_SHORT).show();
}

@Override
public void onNoFingerprintRegistered() {
}

@Override
public void onBelowMarshmallow() {
    Toast.makeText(this, "This device does not support fingerprint verification.", Toast.LENGTH_SHORT).show();
}

```

If the user' fingerprint is matched with the owner of the smartphone fingerprint, then the user will be informed that second authentication is succeed and the welcome page will be opened. We also send the username to the welcome page. The codes belong the second authentication is below:

```

@Override
public void onAuthSuccess(FingerprintManager.CryptoObject cryptoObject) {
    Toast.makeText(this, "Fingerprint reading successful.", Toast.LENGTH_SHORT).show();
    Intent welcomePage = new Intent(this, welcomePage.class);
    welcomePage.putExtra("WhoAmI", user);
    startActivity(welcomePage);
}

```

In the welcome page, we only take username and assign it to text view to show the welcome message to the user. The code for this page is below:



```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_welcome_page);

    Bundle ComingData = getIntent().getExtras();
    if(ComingData == null){
        return;
    }

    String logged_user = ComingData.getString("WhoAmI");
    TextView username = (TextView)findViewById(R.id.textView5);
    username.setText("Welcome " + logged_user.toUpperCase() + " to our application.");
}

```

For this part, we introduced codes that we used for the project. After we present our project to the jury, we will upload it to the github.

## **Project Restrictions**

There are a few restrictions you need to know about the application. Check the requirement table below to run the application on your smartphone.

Supported OS	Minimum Android 2.3.3
Hardware	Fingerprint Sensor
	Your device should support fingerprint sensor
Disk Capacity	Minimum 2MB

## **Project Tests**

We tested our application both virtual device and real devices. While the application was running in real devices, we manage to test the application. We have tested application 2 real devices. The first device supports fingerprint sensor while the second one does not support. Since it is difficult to take screenshot while program running in real devices, we decided to put screen on virtual devices when the application is running. We use real devices in the presentation.

When application is started, user should enter username and password, if the user enters invalid arguments, then the user will see error message. You can see that error message below:



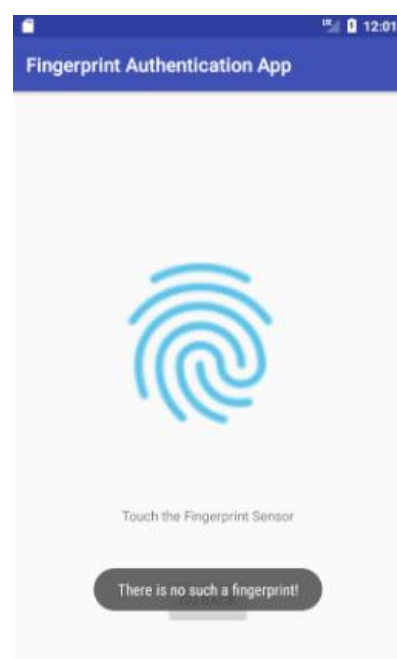
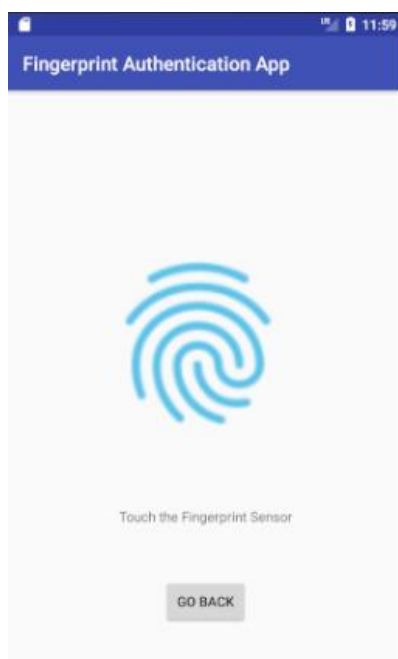
Then if the user users can register the system by clicking “SIGN UP” button. In this page, if the user enters different password, then the program will show error message. The message can be seen below:



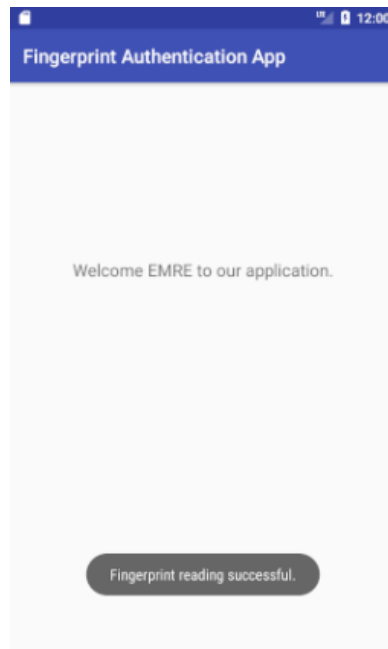
When the user enters passwords and these values are matched, then the user register the system and redirect to login page. You can see this process below:



After that the user can login the system that means the user successfully passes the first authentication mechanism and redirect to second authentication page. In this page user should touch the fingerprint sensor of the device, the left picture you can see that the application waits for this process. In the right picture, user touch the sensor but the application gives an error that indicates the user' fingerprint is not matched with the owner of the smartphone' fingerprint. You can see these 2 processes below:



If the fingerprints are matched, then the user passes the second authentication mechanism and can go access the system. In this case, the user will see a welcome message with his username.



## **Project Outcomes**

In this project, we learned how to develop a smartphone application and for this application we learned different types of authentication mechanism. We also learned how to build and run this application in real devices.