Safety-Banking: Banking System with Image Authentication

Burak Ege Zagnus
ezagnus1@binghamton.edu

Ege Cihan Yavascan
eyavasc1@bingahmton.edu

Keywords — Android Phone, Image recognition, Android App, MySQL Database, Socket Programming.

I. Introduction

Today, software is evolving day by day, new things are emerging every day. How secure these applications or software are is a matter of debate. People register for these applications, create accounts. They enter personal information into applications such as banks, social media, etc. Apps also choose a way to protect this personal information. For example, a security question, send to the phone password, select pictures, extra code can be entered. In our practice we use image comparison. It is a special item that the person has identified, and we keep his photograph in the database first and we take it as soon as we enter the application and compare it with it. So, we use two-step verification with both password and personal property image.

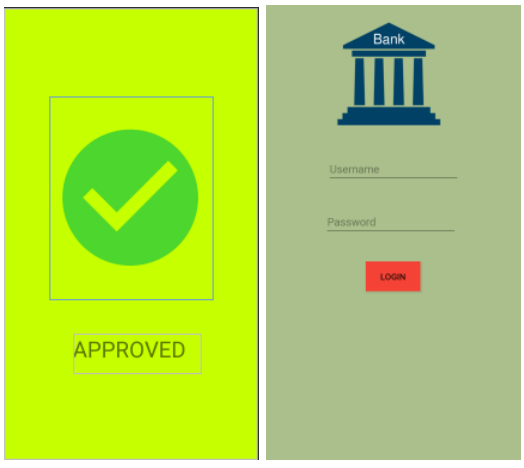
II. Architecture

Image Safety Banking design for the most part comprises of socket relations between close to home advanced mobile phone and PC so as to accomplish information exchange. Primary objective is that client needs to have the option to associate with his or her information in his or her PC with client secret word and user private credentials picture. Also, this data will keep user private information safe and provide two steps authentication. First, when we run our application, the login page appears. There is no registration page because the application is a bank application. Users are not required to register because they are bank customers. There are 3 cysts in this application that provide identification verification.

First, as with every application, first of all, the username and password control if the password is incorrect in the application warning goes out because your information is wrong. The second condition is that the username and password are entered correctly. In this case, there are two options from now on. If the user does not want a two-step authentication for the bank, the password and the username will be entered into the application when it is correct. If the user has requested a two-step authentication press login and camera turns on. After the camera is turned on, the user takes a picture of the object on the phone. If this photo matches the photo in the database, the user is entering the application. If it does not match, the user is unable to access this application.

III. Android Application

Our application is developed with android studio. We have 2 pages in the application, which one is login or main page and the other one is approving page pointing that if the user had access or not. When you open the application, you run into with login page and there are two textbox for the user to enter its credentials, which one is username, and the other one is password, and also we have one login button, after the credentials are entered, the user must click the login button. So, if the user credentials are correct, and if the user accepted image authentication while the user open a bank account, camera will be opened and here the user must take the photo which he/she specified while opening the bank account. But, if the user did not have image authentication done when opening the bank account, so the user can access just with username and password. After a user accepting image authentication shot the photo, then this photo will send to server-side written with Python language to compare the photo taken now with database one. After comparison, if image credential are correct, the approving page is displayed in smartphone. Otherwise, a textbox is displayed including error message.



Approving Page ***Login Page***

IV. MySQL Database

We used MySQL database as our bank database. We hold all information about the users here. We have

4 columns in database which is username, password, safety request, and image name. When the user enter the login button, credentials are taken here to be checked. Safety request is optional because the user can decide with itself about putting image authentication or not to his/her account. If the user accepts image authentication, so safety request will be “true”, otherwise it is going to be “false”. Also, we hold images on server, so, we hold images name (these names are unique for each user). And this name is sent from android application to server. Moreover, we use XAMPP application to use MySQL and Apache server for connection with MySQL database.

+ Options

Username	Password	safety	image
ezagnus1	12345	true	ezagnus1.png
eyavasc1	9876	false	eyavasc1.png

Users Credentials in MySQL

V. Socket Programming

We use two languages for socket programming. In client-side, we use Java language in Android Studio. Also, in server-side, we use Python language. First of all, for sending base64 string of an image taken with smartphone from Android phone to PC, we connect phone socket to the PC socket (server-side). So, we send the data with 7800 port. Server-side takes the string and convert to the image and do image processing to compare two photos. After the comparison, true or false string will be sent to Android application (client-side) again. Because this is another connection, we use another port number 7801 to send this data back to the application.

```python
import socket
import time
import base64
import io
from PIL import Image
from PIL import ImageFile
import cv2
import imutils
import argparse
from skimage.measure import compare_ssim
ImageFile.LOAD_TRUNCATED_IMAGES = True

listensocket = socket.socket()
listensocket2 = socket.socket()
PORT = 7800
PORT2 = 7801
maxConnection = 999
IP = socket.gethostname()

listensocket.bind(('',PORT))
listensocket2.bind(('',PORT2))
listensocket.listen(maxConnection)
listensocket2.listen(maxConnection)
print("Server started at " + IP + "on port"+ str(PORT))

(clientsocket,address) = listensocket.accept()
(clientsocket2,address) = listensocket2.accept()
print("new connection made")
```

Server-Side Socket Programming

```java
public class MessageHandling extends AsyncTask <String,Void,String> {
    Socket my_socket,my_socket2;
    DataOutputStream my_dos;
    PrintWriter my_pw;
    String in = "";
    public AsyncResponse delegate = null;

    @Override
    protected String doInBackground(String... voids) {
        try {
            my_socket = new Socket( host: "192.168.0.8", port: 7800);
            my_socket2 = new Socket( host: "192.168.0.8", port: 7801);
            my_pw = new PrintWriter(my_socket.getOutputStream());
            my_pw.println(voids[0] + " ");
            my_pw.println(voids[1]);
            my_pw.flush();
            my_pw.close();
            my_socket.close();


            BufferedReader stdIn = new BufferedReader(new InputStreamReader(my_socket2.getInputStream()));
            in = stdIn.readLine();
            my_socket2.close();
            return in;
```

Client-Side Socket Programming

VI. Software

This application is done in Android Studio using Java language. For database, we use MySQL database and Apache server. For socket programming, we use Java on client-side, and Python on server-side. Also, for image processing, we use Python language.

VII. IMAGE RECOGNITION SYSTEM

In Image Safe Banking, this section is the most crucial point because the images are compared here. We use SSIM to make comparisons in practice. SSIM is a probability calculation method. First, we set the dimensions of the two photos to a scale and equalize them. After this equation, we turn both photos Gray. This makes it easy to compare images. After turning gray, the similarity is calculated, and the same photo value finds the photos match and go into application. However, this step does not occur if users do not want to confirm two steps in the application.

```
# construct the argument parse and parse the arguments
image.save("phone_photo.png")
path = temp_split[0]
path2 = 'phone_photo.png'
# Using cv2.imread() method
imageA = cv2.imread(path)
imageB = cv2.imread(path2)
scale_percent = 50
#calculate the 50 percent of original dimensions
width = int(imageA.shape[1] * scale_percent / 100)
height = int(imageA.shape[0] * scale_percent / 100)
#dsize
dsize = (width, height)
# resize image
imageB = cv2.resize(imageB, dsize)
imageA = cv2.resize(imageA, dsize)
# convert the images to grayscale
grayA = cv2.cvtColor(imageA, cv2.COLOR_BGR2GRAY)
grayB = cv2.cvtColor(imageB, cv2.COLOR_BGR2GRAY)
# compute the Structural Similarity Index (SSIM) between the two
# images, ensuring that the difference image is returned
(score, diff) = compare_ssim(grayA, grayB, full=True)
```

SSIM Programming

VIII. CONCLUSION

As a result, security in applications is very important. Every day there are other threats against the software. These threats can steal personal information from applications. There are many precautions against these threats. For this reason, authentication is one of the most widely used methods. In this practice, we have taken the identity verification Act twice to a higher level of security. We do two-step identification verification with both password and image comparison**.**