Social Networking Application using FaceNet and Firebase

Gautham Vijayaraj[*], AMJ Muthu Kumaran[*], K Mohanakrishnan[*]

[*]Department of Computer Science and Engineering, SRM Institute of Science and Technology, Potheri, Kattankulathur, Chengalpattu, Tamil Nadu 603203 ;gv1809@srmist.edu.in; muthua@srmist.edu.in; mp1762.srmust.edu.in

Abstract

As we all know most of our social lives are controlled by social networking sites. Social networking refers to the grouping of individuals and organizations together by the help of some medium, in order to share thoughts, interests, activities, etc. There are several web based social network service that are already available. But most of these sites are vulnerable to our personal information being exploited. The goal of this project is to explore and spearhead the use of face authentication in social media applications, which we believe will be a great addition to the already existing security features for social media platforms. Thus here we present our own social media platform(JC) that is authenticated using the FaceNet(Face Authentication) and made using the Google's Firebase platform .

***Keywords*: FaceNet; Firebase; Face Authentication.**

1. Introduction

In the recent years face recognition has gained considerable amount of attention from researchers in pattern recognition, biometrics, and computer vision groups. The following topics Anatomy of SOAP Security Performance from [2] and Social media security from [3] gives us reason to under the need for new security measures and also our own approach using the Face recognition techniques. There are countless security measures, and forensic applications that require the use of face recognition technologies. As you can see, the face detection system is important in our day to day life. Among all of the sorts of biometrics, face detection and recognition system can be considered as the most accurate one. In [4], Ashu Kumar et al[4] have presented a survey on the existing face detection techniques. With their help we have settled on the use of an API to make the face authentication process efficient and at the same time more accurate.[9] and [8] specifies the use of AI and API in with face authentication and also helps cement the process of using API in our project also. One of our main basis of belief that an application where face authentication can be a viable option can be brought to light with the help of [8]. The system proposed in ([8]) was designed to combine computation that is run on mobile devices and the advantages of cloud computing. The computations on mobile devices (on loading) will execute the augmented reality and face detection module which is then used to interact with the user. Other computations run on cloud servers (offloading) using the Google App Engine (GAE) and Face.com services. The system in this paper was designed in three main modules that is face detection on Android mobile device, augmented reality, and face recognition on cloud server using API as a result is system is gives inspiration for us to follow what we have done with ours. [8] Thus with the help of [4],[8] and [9] we have come to the conclusion that face authentication using an assisted API.

2. Proposed Method

The Facial Recognition method we are going to use is FaceNet. FaceNet is a unified facial recognition system that is based on learning a Euclidean embedding per image using Convolution Neural Networks that consists of 22 layers. The CNN that is used by FaceNet is trained so as to optimize the embedding that will then directly replace the intermediate bottleneck layer that is found in the previous deep learning approaches (Schroff, Kalenichenko & Philbin 2015). The FaceNet system can be recommended since the model has extensively been pre-trained using the VGGFace22 dataset that consists of about 3.3 million images[11]. From [11] it was analyzed that FaceNet showed the best accuracy results on a comparison with the other methods. *Why FaceNet?* From [11] we find that FaceNet can be used to achieve a precision that is close to the 100%. According to the figure we can find that the second best technique is scikit-learn implementation of Fisherface using KNN which gives a precision of 70.44%. Thus the results show that FaceNet clearly outperforms all other techniques where the minimum difference is about 29 percent points. Precision is here calculated by dividing the TPs with all positive predictions, whether they are correct or not, which mathematically is: $TP/(TP+FP)$.

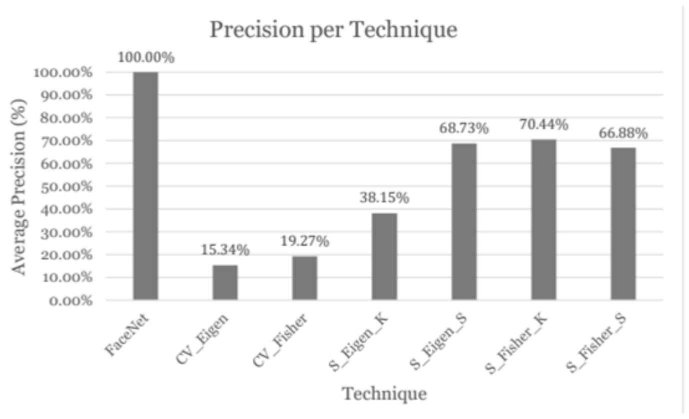


Fig.1. Comparison of precision between FaceNet and other facial recognition methods.

Training time is the time taken for the model to be trained on a specific technique using 400 training images in the dataset. The average training time taken by the FaceNet method was concluded to be 0.24 seconds

2.1. Implementation of FaceNet using Flutter and TensorFlow

It works with two computer vision models working together, the Firebase ML vision model to perform the face detection and preprocessing in the image, and the MobileFaceNet model to process, classify and transform into a data structure 'savable' by a database (an array of numbers).The process summary is as follows:

Sign up

(1) The user clicks a photo.
(2) The ML models processes it and create an output in the form array of numbers that is stored in a database.
(3) Name and a password are then requested.

Sign in

(1) The user clicks a photo.
(2) The ML models processes it and create the required output.
(3) The output will be compared against the outputs already stored in the database (it compares by proximity the closest one it finds). As condition, the proximity has to be under the *threshold* (minimum distance), if overcomes it, it will process it as a non-existent user.
(4) If the user exists (face already processed) it requests the password for that user, validates and authenticates it.[13]

How does it work? The TensorFlow provides a set of pre-trained models that are able to solve a plethora of machine learning related problems. All the models have been converted to be able work with TensorFlow and are ready to be used in applications. TensorFlow is designed to run various models efficiently on mobile devices and other embedded devices that have limited computing resources and memory.[12] Part of their efficiency is gained from using a special format to store the models. These models must be converted to the special format before TensorFlow

can be able to implement them. The process of running data through a model to obtain recognitions is called inference. It requires a model, input data, and an interpreter.[12] Converting FaceNet.pb to FaceNe.tflite. The pre-trained FaceNet model is quantized, where the embedding size can be chosen as either 512 or 128. The software requirements for this process are just Python (at least version *3.4).*After that we need to install the tensor flow package using the pip install command. The FaceNet model can be converted from (.pb) to (.tflite) file by following the steps from [12]How is the .tflite file implemented in flutter? With Face Detection API found in the ML kit, you can detect faces that exist in an image, obtain the contours and identify key facial features of detected faces. It works very well in the preprocessing of images to detect zones to be cropped and then processed by the MobileFaceNet model.
[14]The ML kits Face detection model detects existing human faces in the frames that the camera preview and the class Face contains the coordinates of the points that make up the frame around the face.

```
List<Face> faces = await _mlVisionService.getFacesFromImage(image);
```

This command detects a list of images from the picture taken from the camera of our mobile. The detected face of the last frame is captured, cropped and then pre-processed to be processed by the MobileFaceNet model.
The MobileFaceNet model returns an output (array of numbers).If it’s the sign up process , the application requests a name and a password, and later saves the three data (name, password and ML output). If it’s the Sign In process, the application performs a search in the database comparing the Euclidean distance between the ML output of the image and the ML outputs stored for each user, the one that matches or is close enough (accuracy above the threshold) the application brings the data and requests a password.

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$
$$= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

Fig.2 Euclidian distance formula

2.2. Frontend – (Flutter Framework) Flutter is an open-source User Interface SDK made by Google. It is used fir the development of IOS, Android, Windows, Mac, Linux, and the web applications from a single codebase. This is the platform we are going to use to design the front end of our application. The coding will be done in the dart language which was also developed by Google.

2.3. Backend – (Firebase and Node Js) Firebase is also is a platform developed by Google for storing data and accessing using the Cloud Firestore Database. The primary authentication is done through this platform for all the users to sign in. NodeJs is used for creating trigger functions in order to send push notifications and confirmation emails to all the users in real time.

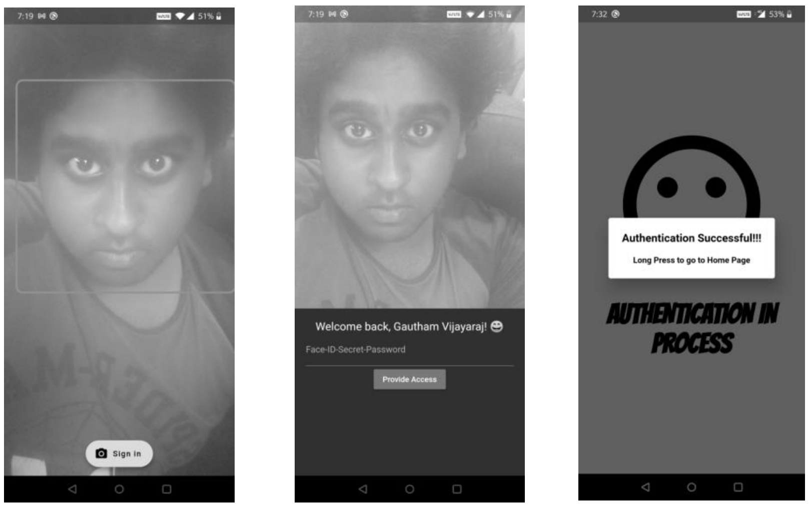

Sign in
Welcome back, Gautham Vijayaraj! 😊
Face-ID-Secret-Password
Provide Access
Authentication Successful!!!
Long Press to go to Home Page
AUTHENTICATION IN PROCESS

Fig.3.Working of Face Authentication in the JC app.

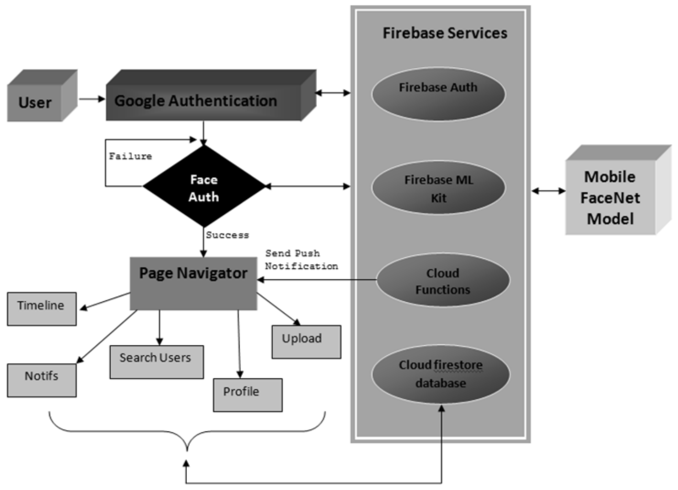


Fig.4. Architecture Diagram of the application

2.4. Cloud Functions

Once the user signs in we want to program 2 functions. One function to send a welcome message to the registered email and another to send push notifications if that user receives any new notifications. Since we are creating a social media application the need is imminent and it is specified by the figure 5 below. Modules required for the functions:

- Firebase-functions
- Firebase-Admin
- Node-Mailer
- Cors

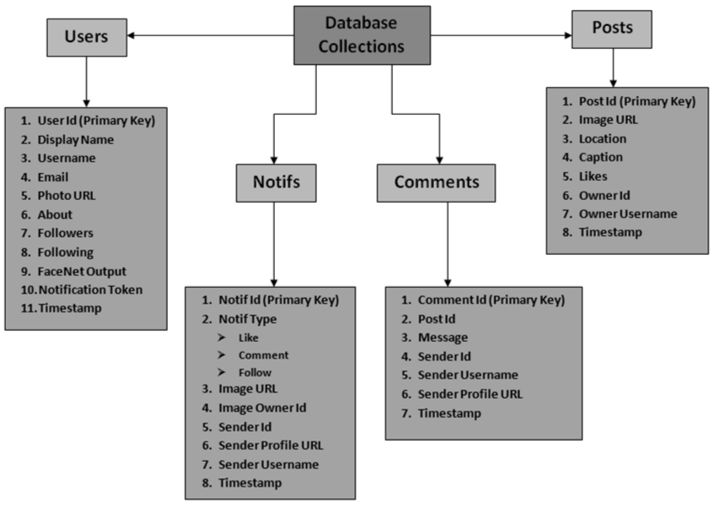


Fig.5.Block diagram of Cloud-Firestore Database

3. Messenger Feature

This Application wouldn't be a complete Social Networking App without the chatting feature so that's the next feature that is going to be focused on. The messages would be encrypted using the Triple DES algorithm and stored in the cloud firestore and few more additional collections in the database. The message which gets stored in the database looks like the figure 6,7 and the end output will be like the figure 8(both given below):

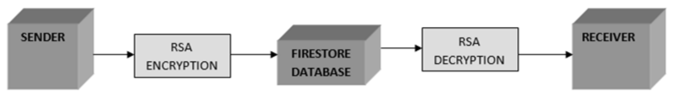
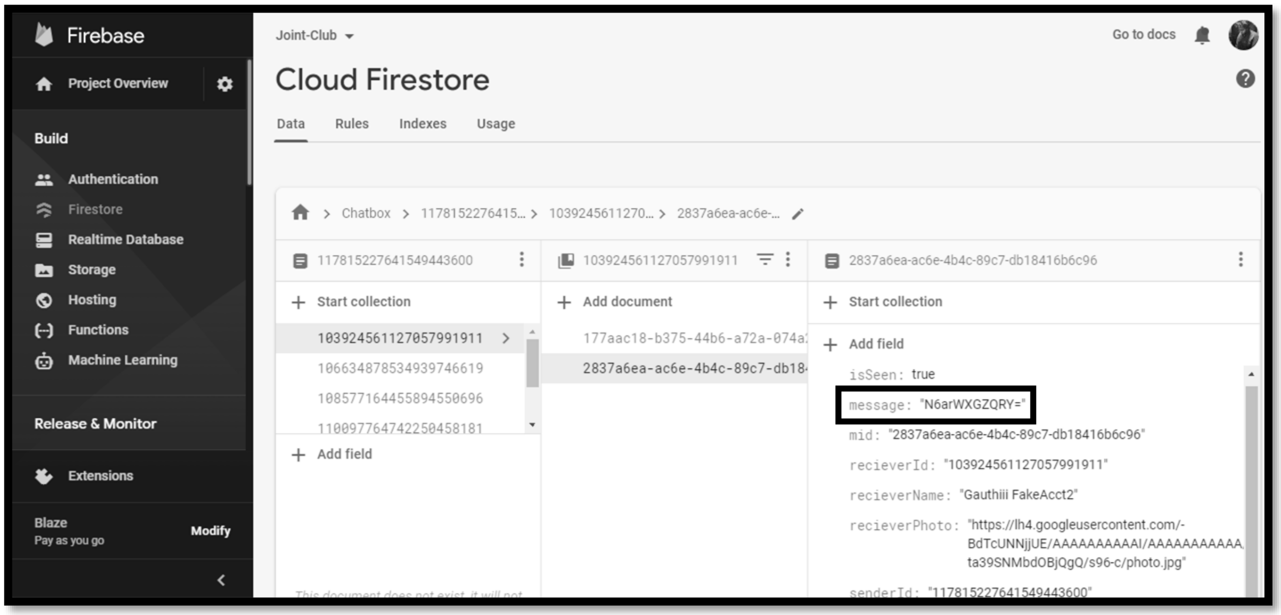


Fig.6.Block diagram of Sending and Receiving of messages



Chatbox > 1178152276415... > 1039245611270... > 2837a6ea-ac6e-...

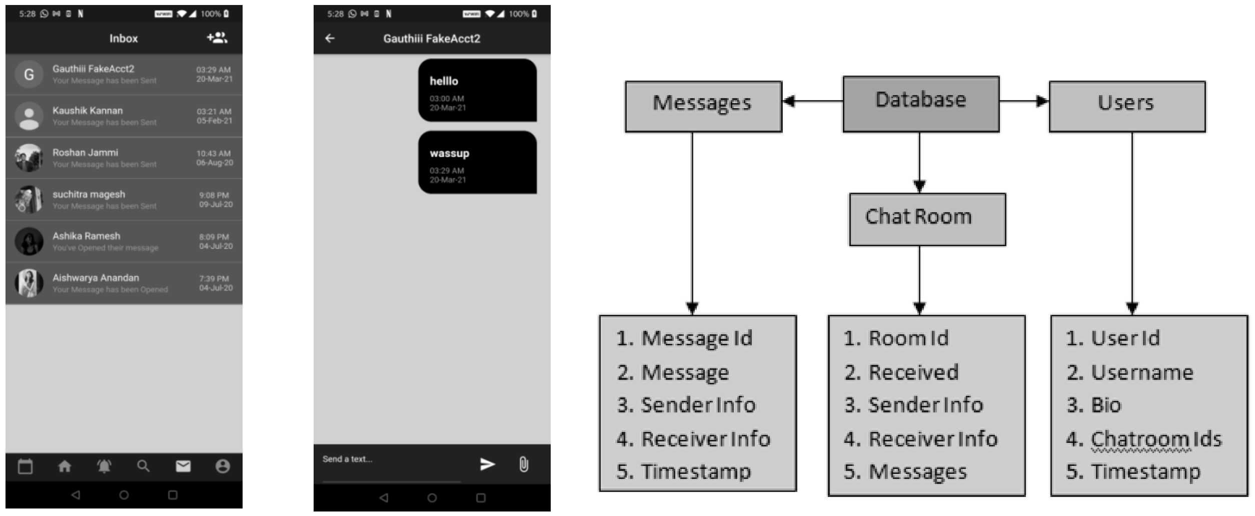
```
117815227641549443600
  103924561127057991911
  106634878534939746619
  108577164455894550696
  110097764742250458181

103924561127057991911
  177aac18-b375-44b6-a72a-074a
  2837a6ea-ac6e-4b4c-89c7-db18

2837a6ea-ac6e-4b4c-89c7-db18416b6c96
  isSeen: true
  message: "N6arWXGZQRY="
  mid: "2837a6ea-ac6e-4b4c-89c7-db18416b6c96"
  recieverId: "103924561127057991911"
  recieverName: "Gauthiii FakeAcct2"
  recieverPhoto: "https://lh4.googleusercontent.com/-
                 BdTcUNNjjUE/AAAAAAAAAAI/AAAAAAAAAAA
                 ta39SNMbdOBjQgQ/s96-c/photo.jpg"
  senderId: "117815227641549443600"
```

This document does not exist, it will not

Fig.7. This is how the messages are stored in the Google Firestore



Gauthiii FakeAcct2	Your Message has been Sent	03:29 AM 20-Mar-21
Kaushik Kannan	Your Message has been Sent	03:21 AM 05-Feb-21
Roshan Jammi	Your Message has been Sent	10:43 AM 06-Aug-20
suchitra magesh	Your Message has been Sent	9:08 PM 09-Jul-20
Ashika Ramesh	You've Opened their message	8:09 PM 04-Jul-20
Aishwarya Anandan	Your Message has been Opened	7:39 PM 04-Jul-20

Gauthiii FakeAcct2

helllo
03:00 AM 20-Mar-21

wassup
03:29 AM 20-Mar-21

Fig.8. Messaging in the app(left) and its underlying features(Block diagram to the right).

Fig.9. The application can be tested from the apk file that can be downloaded from the QR Code.

4. Conclusion

To sum up, JC App (Joint-Club – the name of the app) is different from the rest of the following apps because this app has Face-Recognition enabled authentication. Even if there are other social media apps which facial recognition, the JC uses FaceNet Model for process, classify and transform into a data structure 'savable' by a database (an array of numbers). The next additional feature that is implemented is regarding the security of the conversations. Data leaks are very common nowadays and the Chat-Screen is designed in such a way that the conversations can neither be saved by taking Screenshots nor Screen-Recorded by the users. As of now the messages are encrypted using the Triple DES Algorithms. However the application would be updated once new ideas are brainstormed to make this application more efficient and secure.

References

[1] Varsha R. Mouli, and K.P. Jevitha, "Web Services Attacks and Security- A Systematic Literature Review" in 6th International Conference On Advances In Computing & Communications(ICACC), 2016, pp.1-8.

[2] Hongbin Liu, Shrideep Pallickara, and Geoffrey Fox, "Performance of Web Services Security", Proceedings of the 13th Mardi Gras conference, 2005, pp.1-8.

[3] Zhiyong Zhang, and Brij B.Gupta, "Social media security and trust worthiness: Overview and new direction", Future Generation Computer Systems(FGCS), 2016, pp.914-925

[4] Ashu Kumar, Amandeep Kaur, and Munish Kumar," Face detection techniques: a review", Artificial Intelligence Review, 2018, pp.1-23.

[5] Priyadarshini Patila, Prashant Narayankarb, Narayan D G c, and Meena S Md, "A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish", International Conference on Information Security & Privacy(ICISP), 2015, pp.617-624.

[6] Nilanjan Chatterjee, Souvik Chakraborty, Aakash Decosta, Dr. Asoke Nath, "Real-time Communication Application Based on Android Using Google Firebase", International Journal of Advance Research in Computer Science and Management Studies(IJARCSMS), 2018, pp. 74-79.

[7] Prasetyawidi Indrawan , Slamet Budiyatno , Nur Muhammad Ridho , and Riri Fitri Sari, "Face Recognition for Social Media With Mobile Cloud Computing", International Journal on Cloud Computing: Services and Architecture (IJCCSA), 2013, pp. 23-35.

[8] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, Lior Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1701-1708.

[9] Levent Gokrem , Omer Faruk Nasip, "An Encrypted Messaging Application with Multi Fragmented Caesar Encryption Method between Mobile Devices", Journal of New Results in Science (JNRS), 2017, pp. 1-10.

[10] Timmy Schenkel, Oliver Ringhage, Nicklas Branding, "A Comparative Study of Facial Recognition Techniques: With focus on low computational power", Bachelor Degree Project in Information Technology Basic level 30 ECTS, 2019.

[11] Marcos Carlomagno ,"Real-time Image classification using Tensorflow Lite and Flutter", 2020
[12] Milind Deore , "FaceNet Architecture", 2020
[13] Marcos Carlomagno , "Face Recognition Authentication using Flutter and Tensorflow Lite , 2020