SMART ACCESS SYSTEM

Trisha Ashok
Vaishakh Gowda

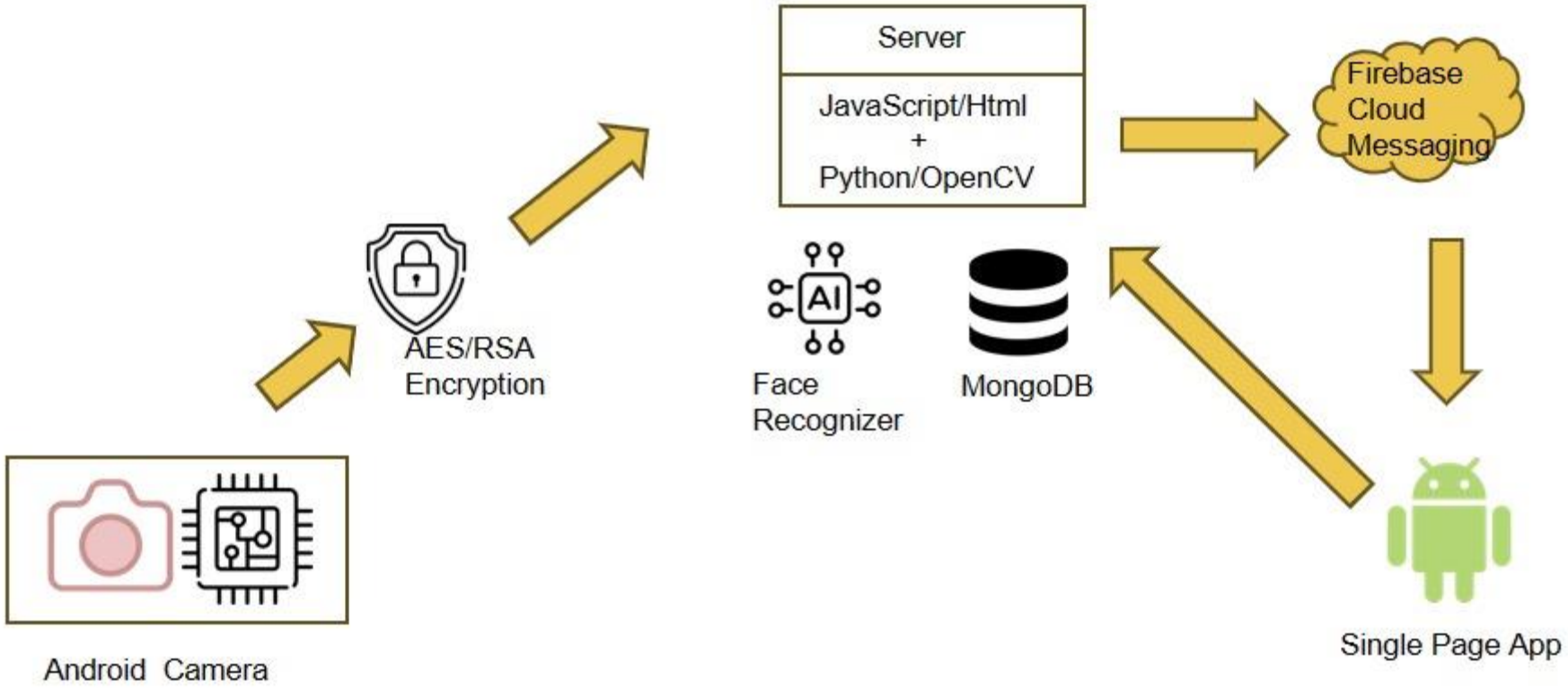
Introduction

Facial recognition technology is an inventive solution that improves convenience and home security.

- The system consists of following components :
- **A Client-side Android application** that connects to the camera to access, and
- **An Admin-side Android application** that can grant, store, or refuse access.
- A Python-OpenCV Face recognition package
- NodeJS supported backend

The facial recognition model is trained and tested by Python scripts in the system, while **Node.js** is used for **server-side activities** and OpenCV for facial recognition.

Architecture



Client-side Application

Hardware configuration of Android application and Camera Module:

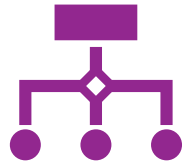
The camera module attached takes pictures in real time at the front door of the house.

Method of Facial Recognition:

The camera looks for faces all the time. The photograph is taken and uploaded to the server for facial recognition processing as soon as a face is detected.

Connectivity to the Home Access System:

The system initiates an action, like unlocking the door, if it recognizes the face and finds a match in the database.

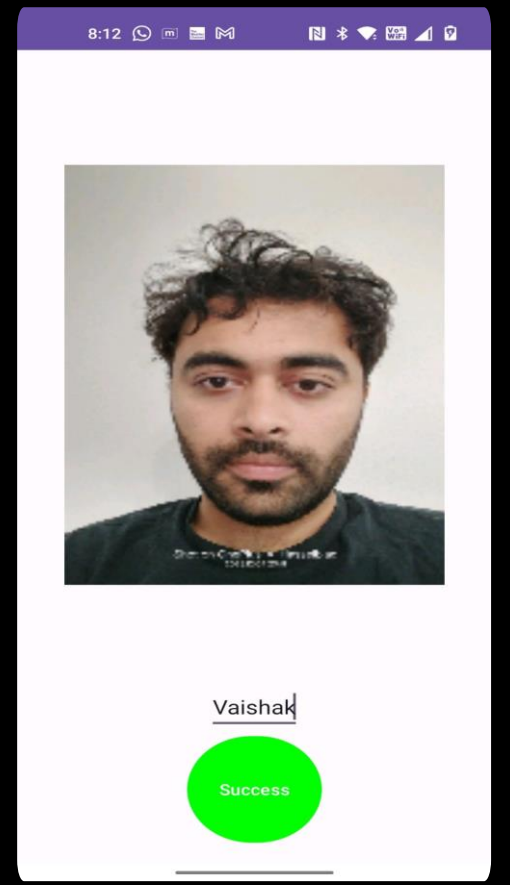


Un-recognized faces may set off an alert. Those datasets are immediately sent to the Server-side application where the admin can decide whether-or-not to grant access. This in return gets stored in the database for future review.

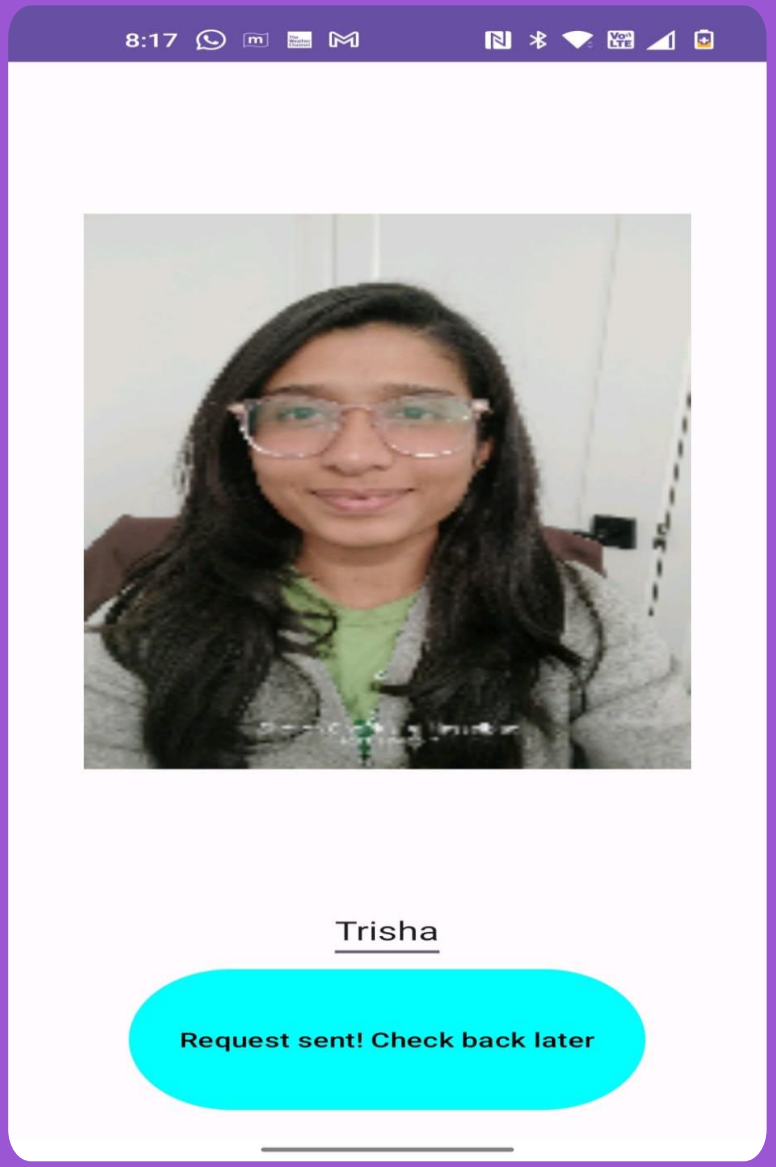
Admin-side Application

- The Admin Application (IoT-App in the repo) is an FCM registered app.
- Database Integration: To store admin settings and logs, we used Firebase or a comparable cloud database.
- As the server receives an unrecognized face from the clint application, the server sends an alert to the Admin App.

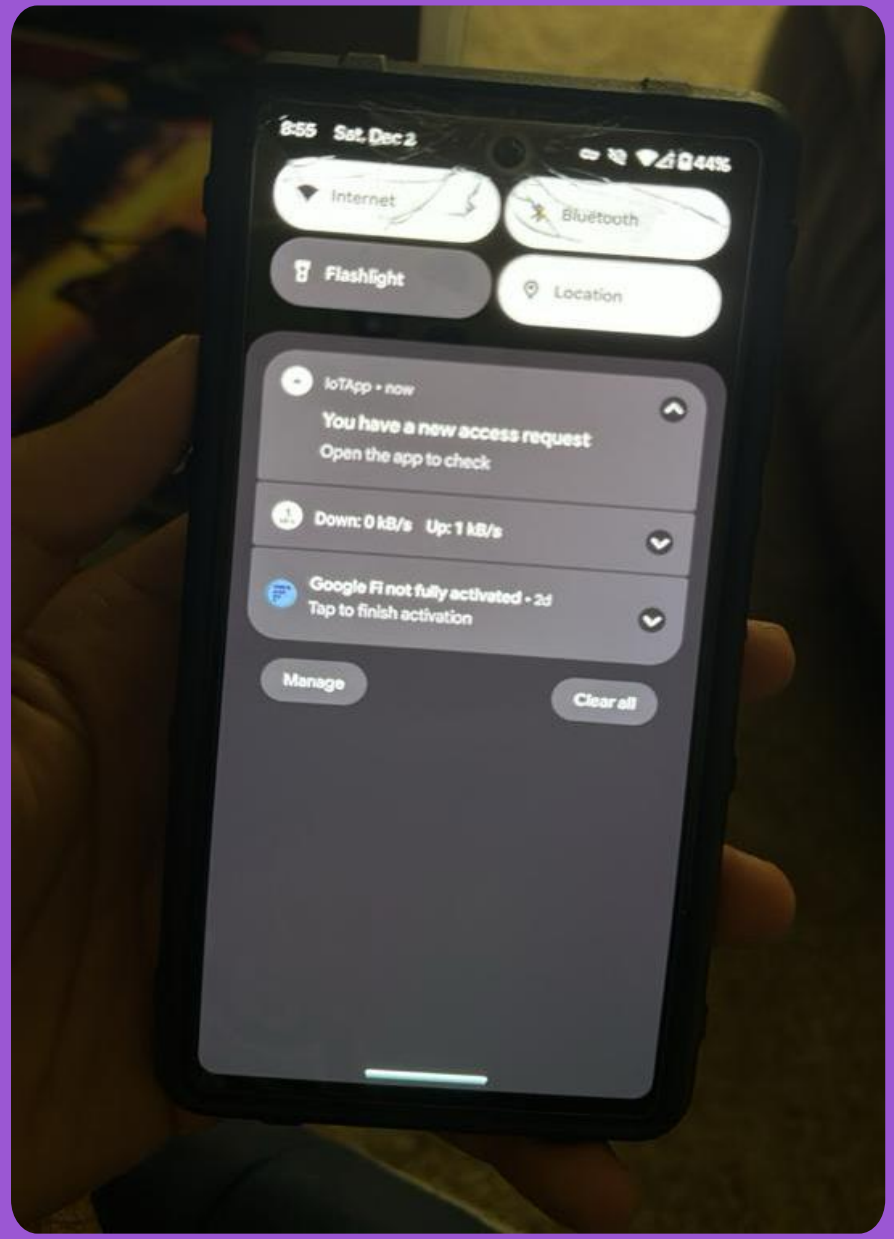
Registered face from Client-side



Requesting access from admin for new entry



Notification at Admin side



Server-side Application

- **Function of the Node.js Server:** Serves as a central point for handling requests from the Android application of client-side and admin-side.
- It responds to and manages requests for facial recognition.
- Data management includes keeping track of access logs and a database of recognized faces.
- It guarantees effective and safe data processing.
- **API Endpoints:** Offers access control, log retrieval, and image processing API endpoints.
- **Management and Retrieval:** Effective data retrieval for processing recognition.
- Permission management for users via the Android application.

Overall procedure

****Facial Image Capture and Server Communication**:**

- The client's facial image is captured and transmitted to the server.
- Python script `test.py is used to verify if the face is recognized.
- In case the face is unrecognized, the server sends an FCM message to the Admin App.

****Administrative Approval Process**:**

- Upon reception of the FCM notification, the admin app alerts the user that they have a new request.
- The Admin can then open the app and view the face of the image trying to access the system.
- On approval, the server makes 5 copies of the image and 5 copies of the flipped image and trains the model using ‘trainer.

Data Processing and Storage:

- The stored image undergoes the `trainer.py` process for validation.
- Upon approval, the validated data is sent and stored in MongoDB.
- Each entry is simultaneously stored in MongoDB for efficient future algorithmic utilization.

Verification and Confirmation:

- The client app can redo the process and gain access as the face will be recognized..

Facial Recognition for Subsequent Access:

- Subsequent visits by the client prompt direct recognition for access without the necessity of sending a request.

Use cases

** Finance and Banking **

Secure Entry and Exit to restricted places.

** Healthcare Industry **

Patient Record Verification

** Security and Access Control **

Biometric Authentication

The Design Gap

Initially, we planned to use raspberry pi and pi- cam as our hardware tool.

As we use the same image multiple times to train the model, we see that the AI is overfitted. This results in recognition errors frequently.

To be more user-friendly, we have implemented on android mobile using the camera module.

Conclusion

This entire process involves a sophisticated pipeline where facial images are scrutinized, verified, and stored, enabling efficient recognition and access for future interactions.