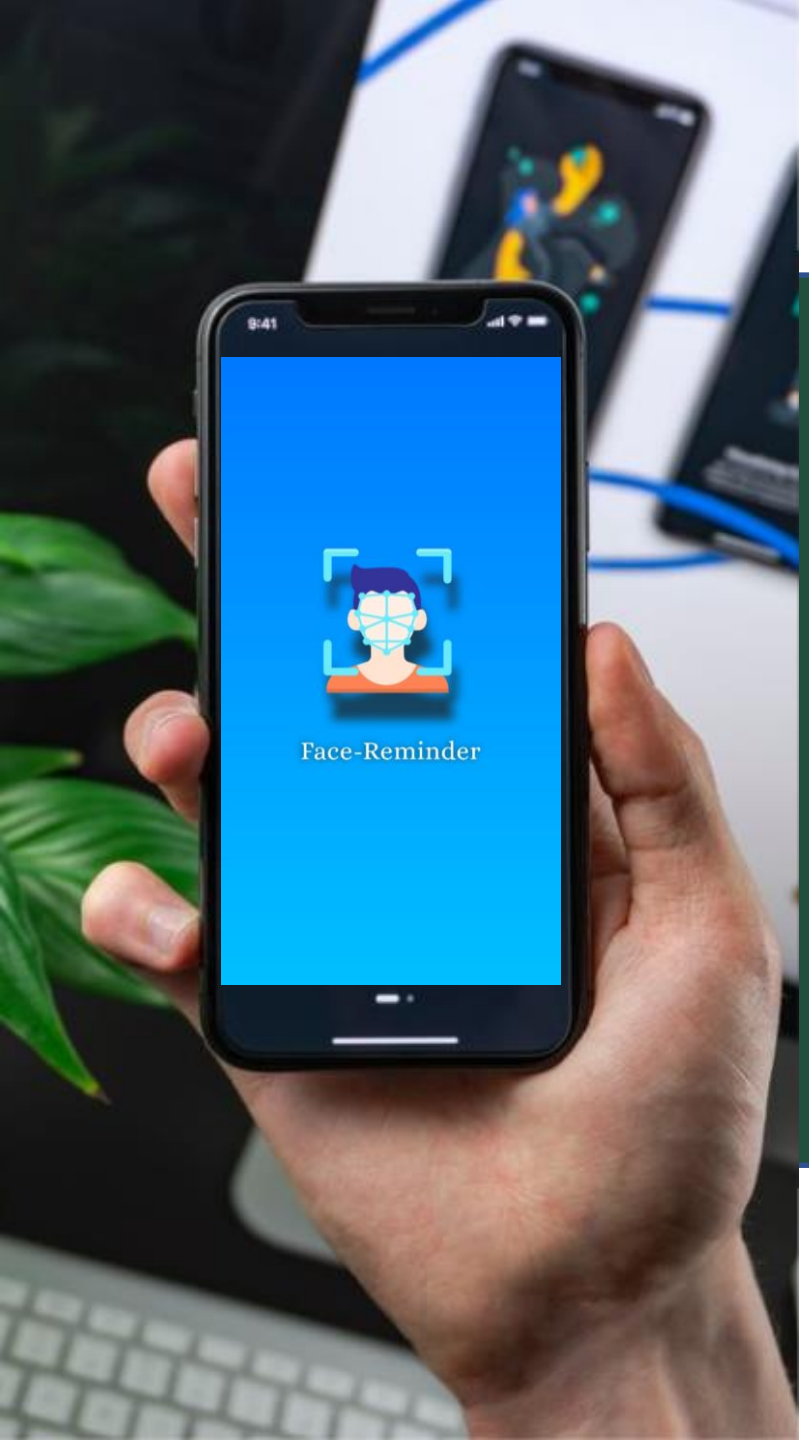


Assistive Mobile Application for Prosopagnosia Patients

1

The Team

Meet the team

Lamis Kamal

Mohamed Zakaria

Nada Elmaghraby

Suhaila Ahmed

Shimaa Mamdouh

Under the supervision of Dr. Sherif Sami

Lamis Kamal

Backend

Suhaila Ahmed

Nada Elmaghraby

Mohamed Zakaria

Frontend

Engine

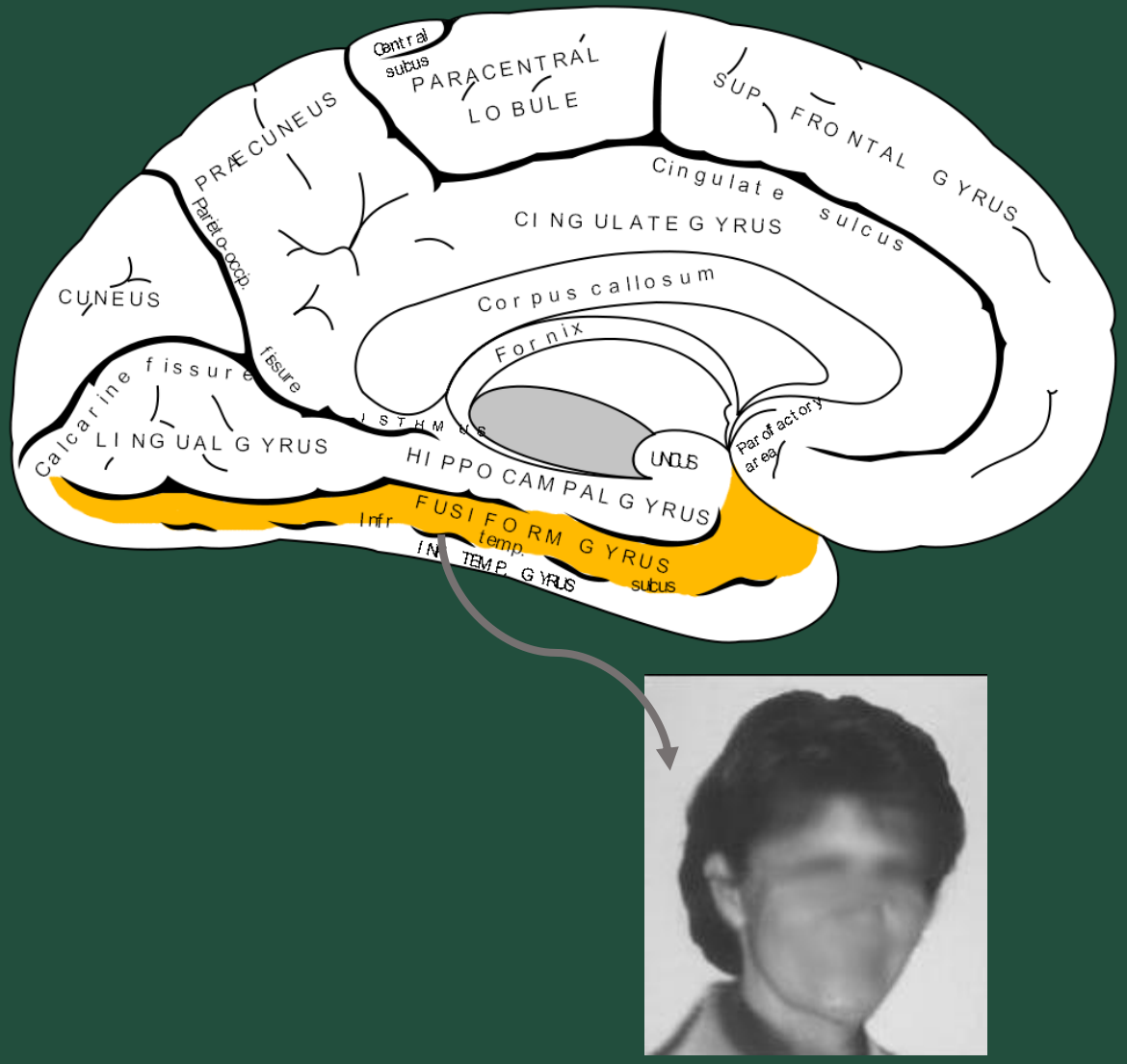
Shimaa Mamdouh

2

The Problem

Prosopagnosia (Face Blindness)

- Prosopagnosia is a result of abnormalities, damage, or impairment in the right fusiform gyrus.
- Which is a fold in the brain that coordinates the neural systems that control facial perception and memory.

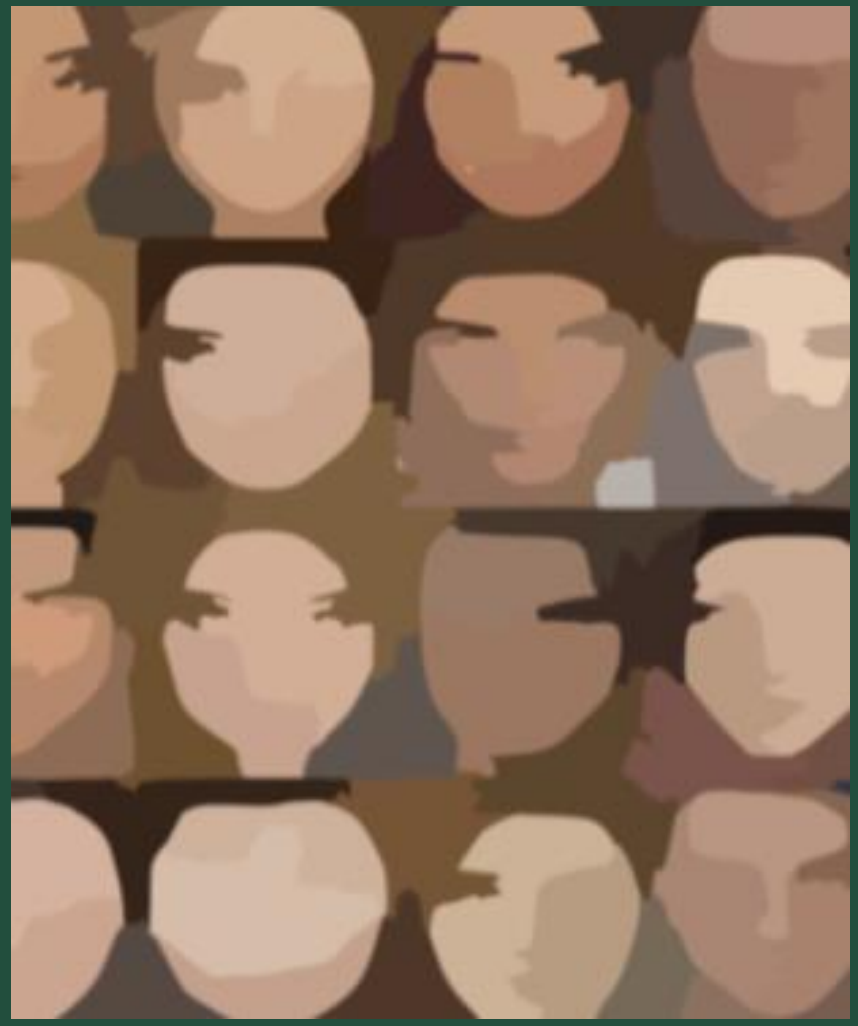


Prosopagnosia (Face Blindness)

- There are two types of prosopagnosia:
  1. Developmental
  2. Acquired

- Studies has indicated that 1 in 50 people may have developmental prosopagnosia.

- There is no treatment for prosopagnosia but patients adopt strategies for identifying people.



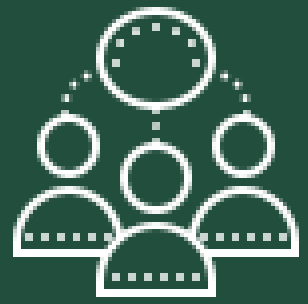
3

The Opportunity

The Opportunity

Statistics show there are over **six billion** smartphone worldwide, and expected to grow.

Artificial intelligence models in the field of facial recognition have become so advanced.



Communication is faster

4

Proposed Solution

Proposed Solution

- Using artificial intelligence algorithms to recognize faces and customize them according to our particular requirements.
- Considering the diverse range of mobile development platforms, we selected the most appropriate one for our requirements in order to construct a mobile application.

5

Market Survey

Market Survey

Orcam MyEye

1. Perform face recognition
2. Activated by voice or click
3. Costs around $4250 USD
4. External device

Social Recall

1. A mobile application for events
2. The information of attendees used during the event
3. Currently being expanded to include Prosopagnosia

6

FaceReminder

FaceReminder Mobile Application

The mobile application tools and platforms used to build face reminder app

Can be connected to an external camera via Wi-Fi, that is optional to each user

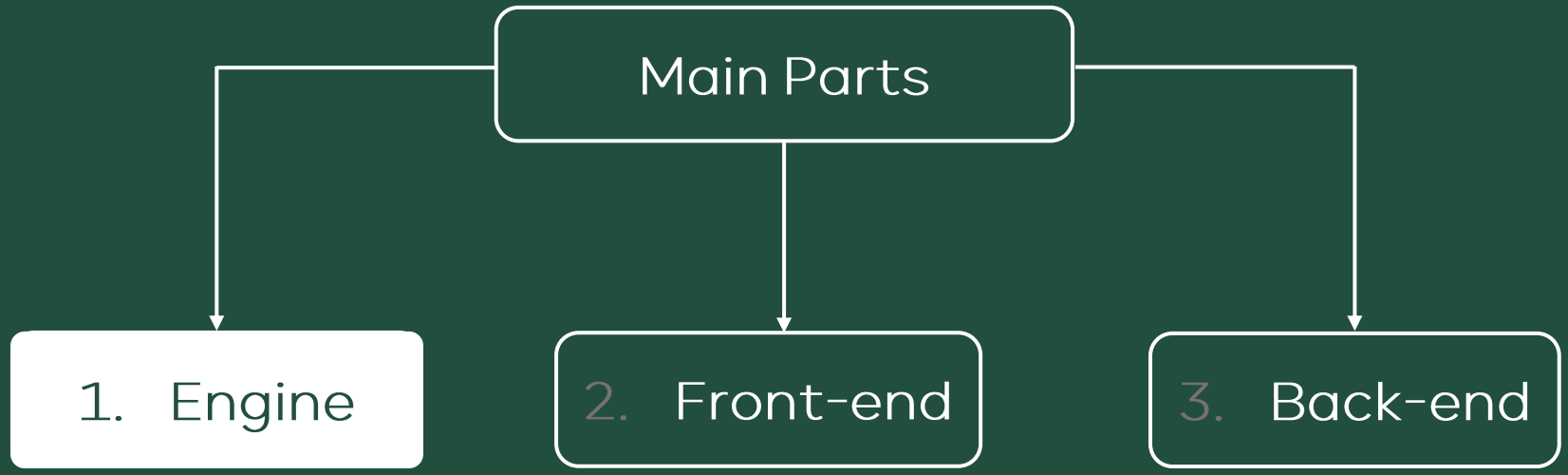
It’s cost effective compared to other similar existing products

7

Methodology

Methodology

In order to achieve the goal of the application, we divided it into



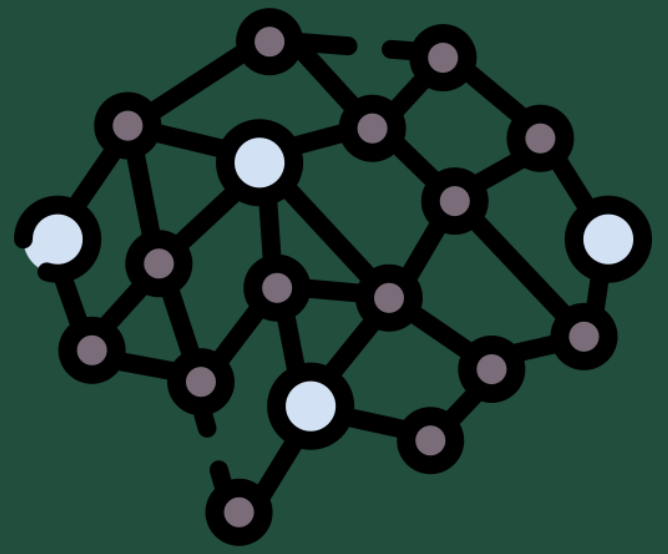
Deep learning model

1. Engine

A. Research Stage
    1. Stages of Recognition
    2. Big Names
    3. Technical Survey

B. Data Collection
    1. Validation

C. Approved Implementation



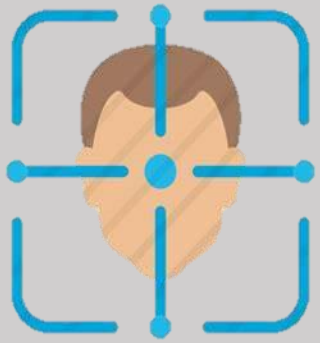
A

Research Stage

A.1 Stages of Recognition

Detection

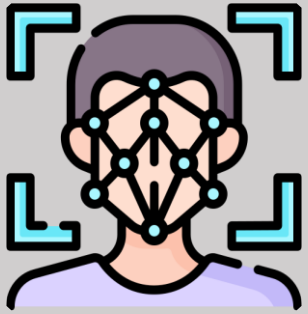
Checks if there's a face



- The detected faces are bounded by boxes
- Annotating facial landmarks

Alignment & Analysis

Orienting the face in a certain direction



- The output image is cropped tight around the face.
- Extracting the values of interest

Recognition

- Comparing the representation values to pre-saved representations of known faces

A.2 Big Names

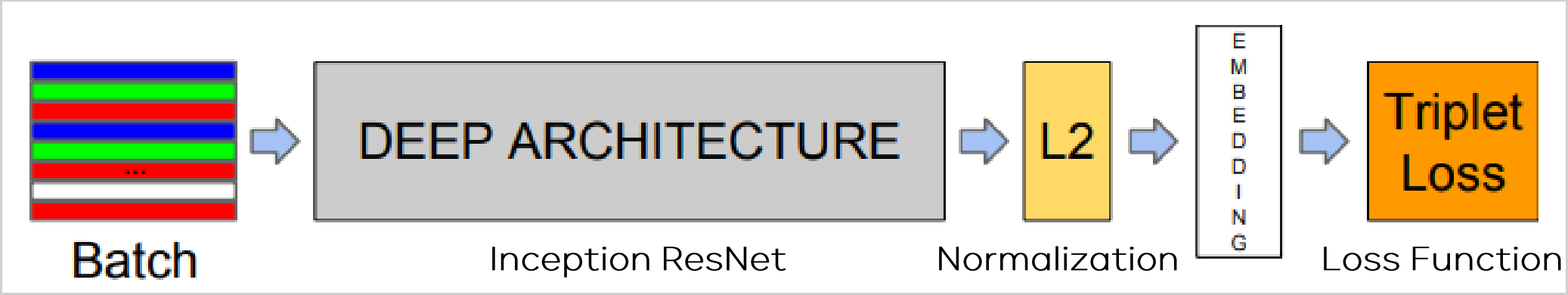
				
Used Technique	DeepFace	FaceNet	Vision Framework	Iris Recognition
Disadvantage	5x memory consumption	———	Require unavailable facilities	Different route
Size	511 M	92 M	———	———

A.3 Technical Survey
DeepFace Framework [3]

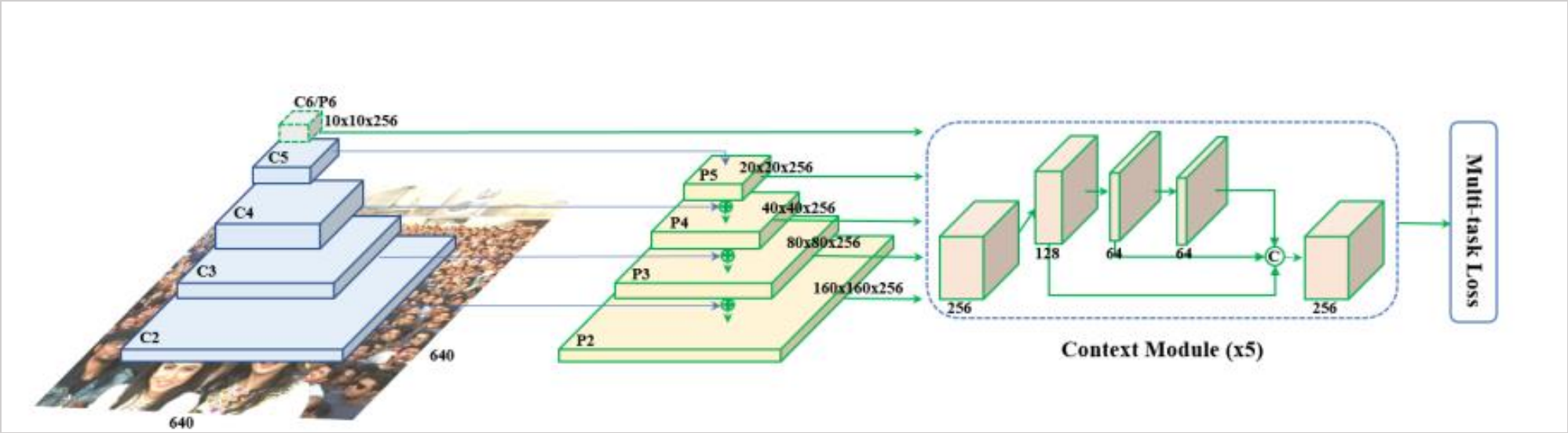
- **DeepFace framework**
  - Opensource lightweight framework for python
  - Licensed under MIT License
  - Provides different face detectors
    - MediaPipe, ssd, Dlib, Mtcnn, RetinaFace
  - Provides different models including **FaceNet** for the face recognition task
    - VGG-Face, FaceNet, FaceNet512, OpenFace, DeepFace, ArcFace, SFace

A.3 Technical Survey
FaceNet [2]

- **FaceNet** system developed by researchers at Google
- Based on Google Inc. the system accuracy on LFW Dataset is nearly 99%
- **The paper proposed two different archtitecture: The Zeiler&Fergus networks and Inception networks.**



A.3 Technical Survey
RetinaFace [1]



Feature Pyramid: to notice smaller faces

Context Module: for enhanced performance

Multi-task loss

B

Data Collection

B. Data Collection

Two Sources of Data Collection:

1. Celebrity dataset which is made up of professional images of 31 persons.

2. Images are collected using a google form of images taken by normal cell phones of 129 persons.

The two datasets then are merged and used for validating the models.

B. Data Collection: Google Forms

Graduation Project Support

We're working on graduation project which requires face recognition techniques, so we need a huge dataset of people images to train and test model.

nadamaghraby18@gmail.com Switch accounts

The name and photo associated with your Google Account will be recorded when you upload files and submit this form. Your email address is not part of your response.

*Required

Full Name *

Your answer

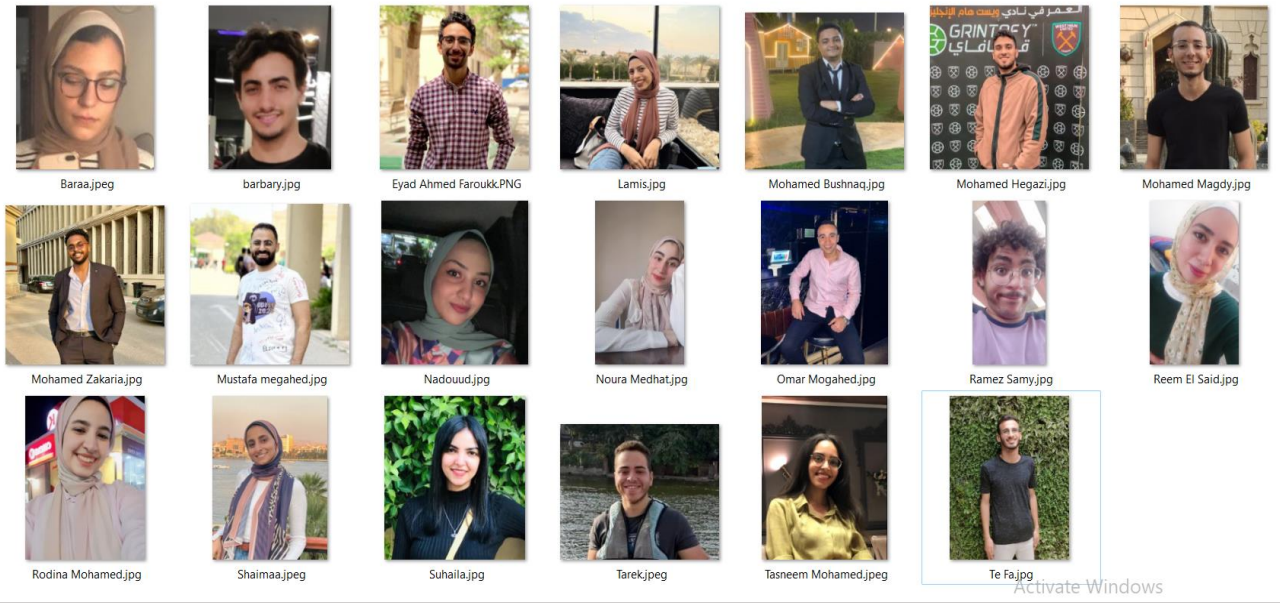
Image of you standing in front of camera like this one. *

My Drive > Graduation Project Support (Fil...

Folders

1 or 2 more optional photos of you...

Image of you standing in front of c...



B. Data Collection

Data Preparation

Before introducing the collected data to the model, a preprocessing step is performed. This involves:

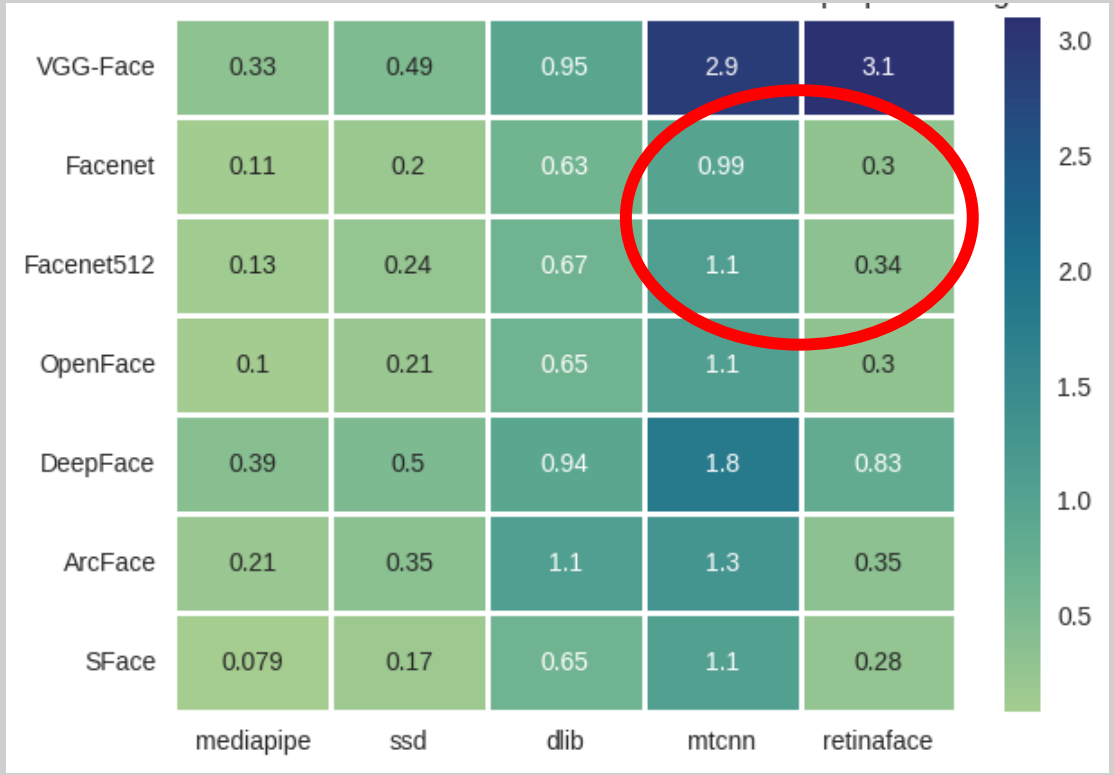
- Scaling all the gathered data to a size of 512x512 pixels
- Converting them to grayscale

B.1 Models Validation

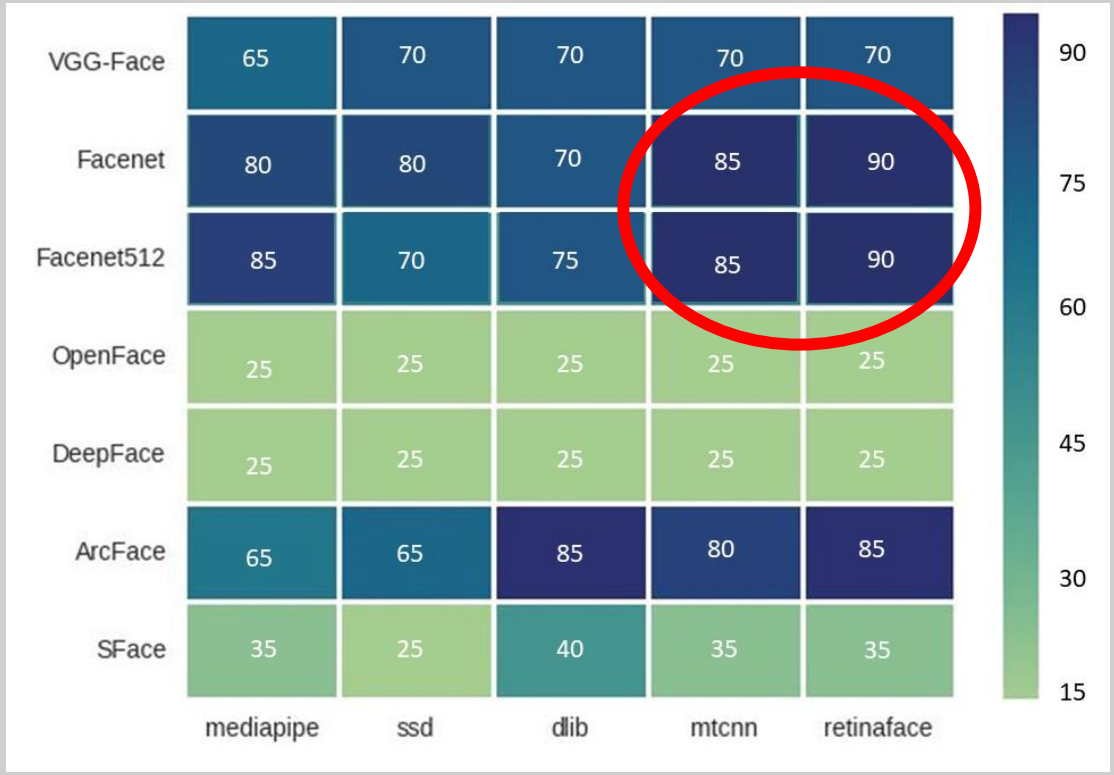
Efficiencies and Processing Time of Models vs Detectors

On subset from Data

Processing Time



Efficiency

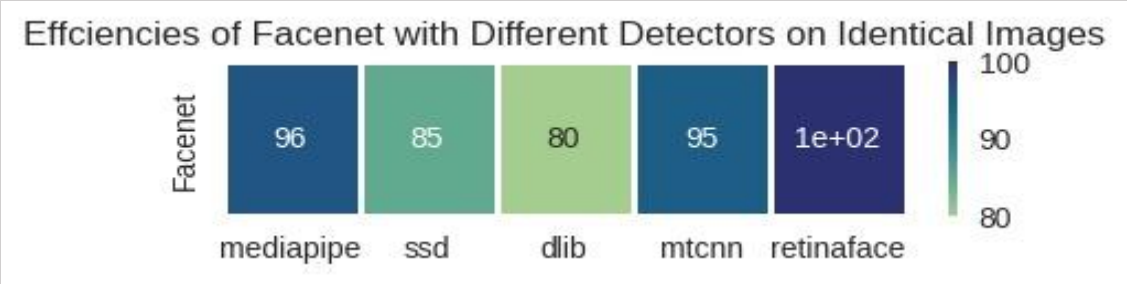


B.1 Models Validation

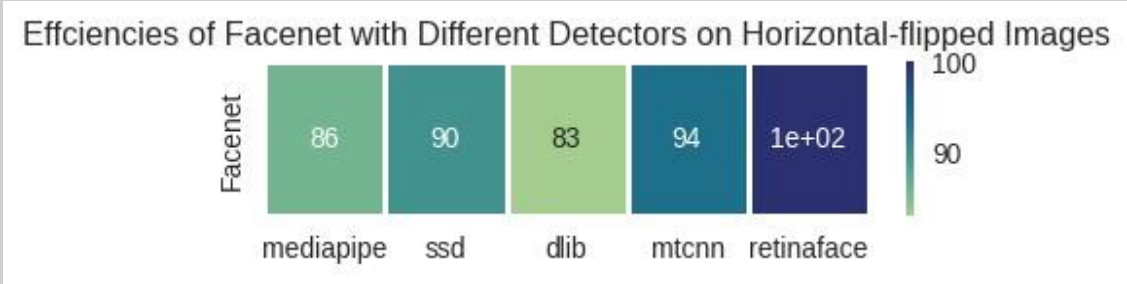
Efficiencies of Models vs Detectors

On our entire Data

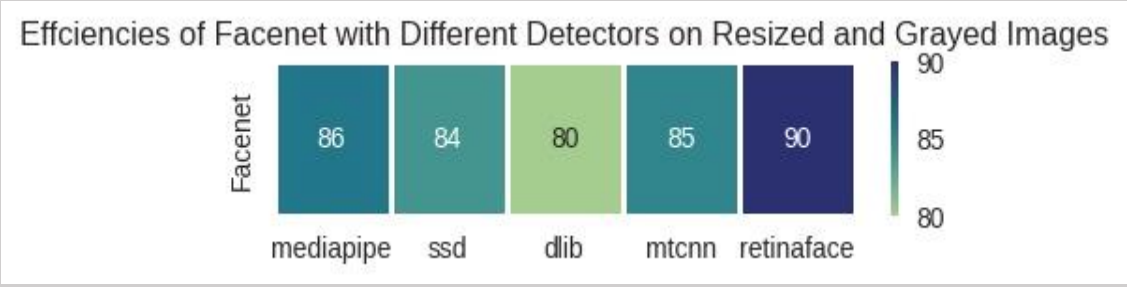
Identical Image



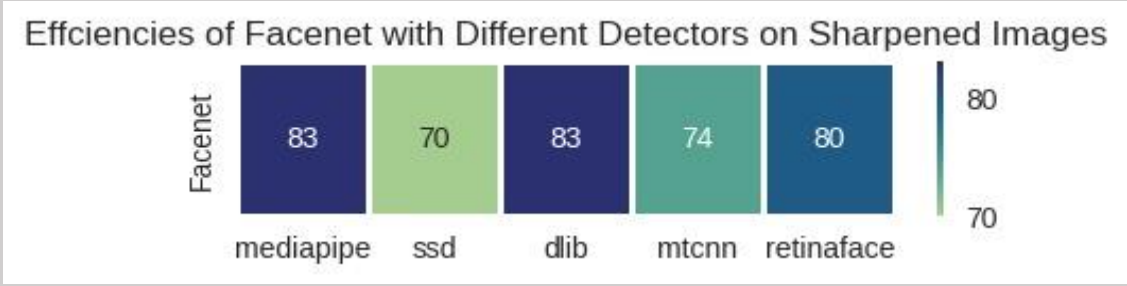
Horizontally-flipped Images



Resized & Grayed Images

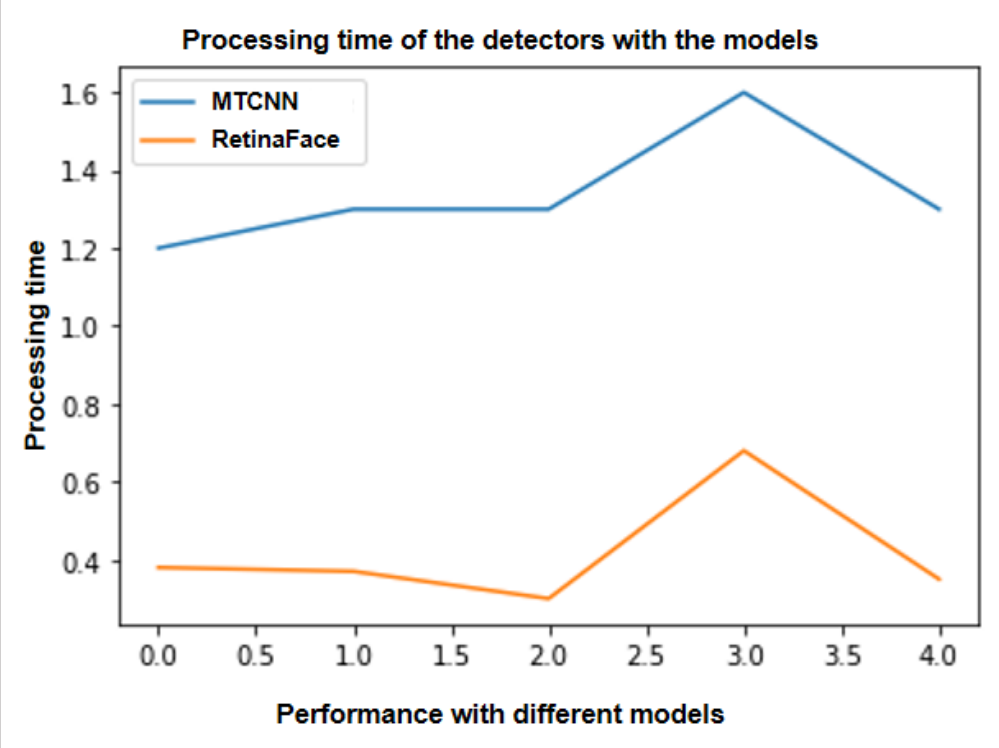


Sharpened Images



B.1 Validation Findings

1. RetinaFace and MTCNN most robust detectors among them.
2. MTCNN takes processing time approximately twice that of RetinaFace
3. FaceNet model is the most robust among them.



B.1 Validation Findings RetinaFace output

Challenge

It detects multiple faces in an image, but our objective is to identify a single face for the recognition task.

Solution

We iterate over the detected faces, select the largest one, and then input it into the model.

B.1 Validation Findings
Sample output

C

Approved Implementation

C. Approved Implementation

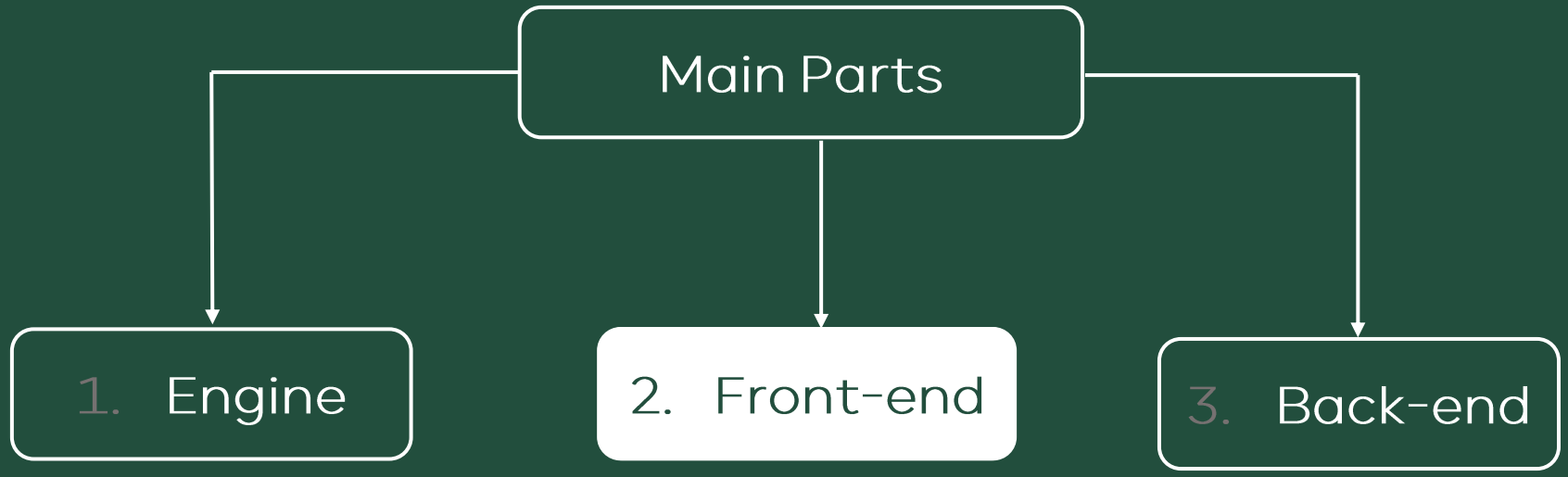
Framework
DeepFace

Detector
RetinaFace

Model
FaceNet

Methodology

In order to achieve the goal of the application, we divided it into



2. Front-end

A. Research Stage

B. Decision

C. Implementation

A. Research Stage

- Front-end Platforms

	React native	Flutter
Language	Java script	Dart
Community	Numerours open source libraries and resources	Less resources
Development time	Recompiled every time changes made	Changes can be seen in real time

B. Decision
React native platform

- It supports:
  - Redux toolkit: setup the store for user data
  - Axios: sending requests and handling responses
  - Vision-camera library: access mobile camera
  - Image-picker library: access mobile local storage

Redux toolkit

Vision-camera library

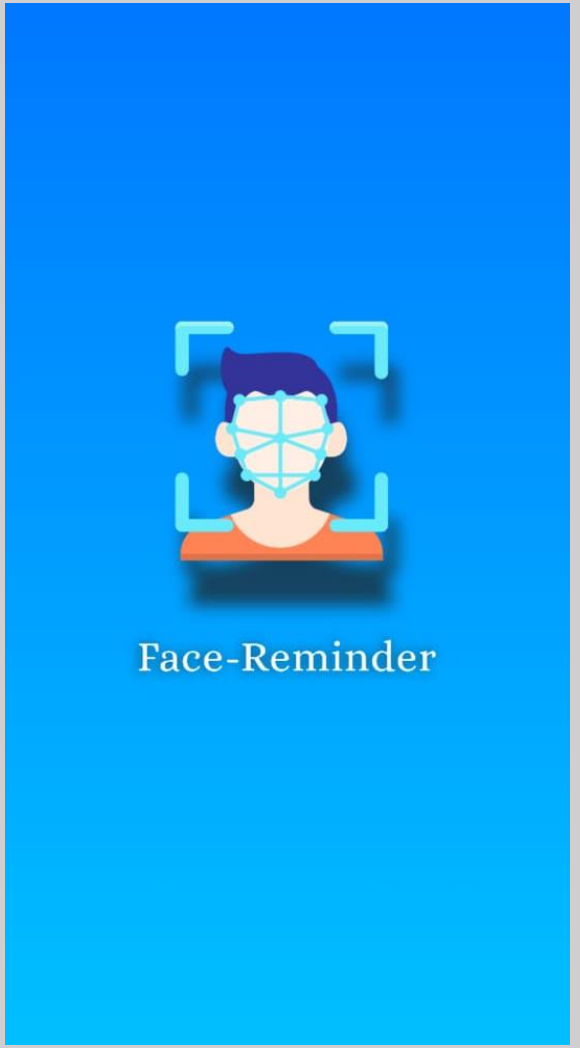
Axios

C. Implementation

UI/UX design:

In designing healthcare apps we have to consider some important aspects as:

- Functionality
- Ease of use
- Attractive UI design
- Security

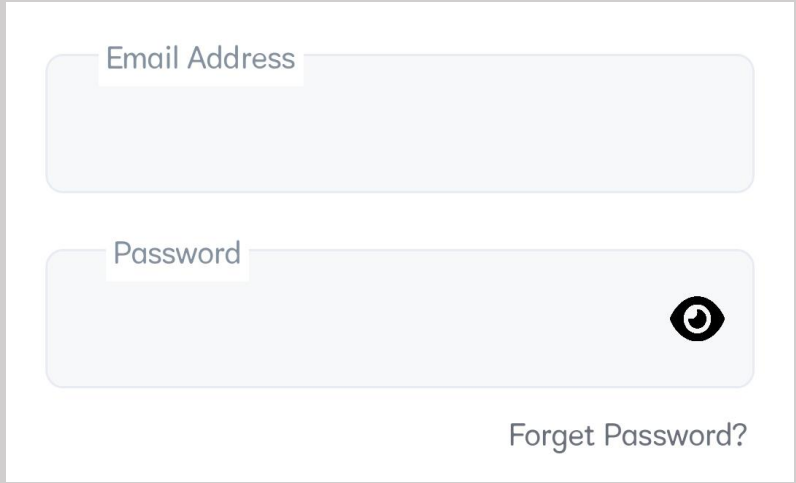


C. Implementation
UI/UX design

Functionality:

Two main components:

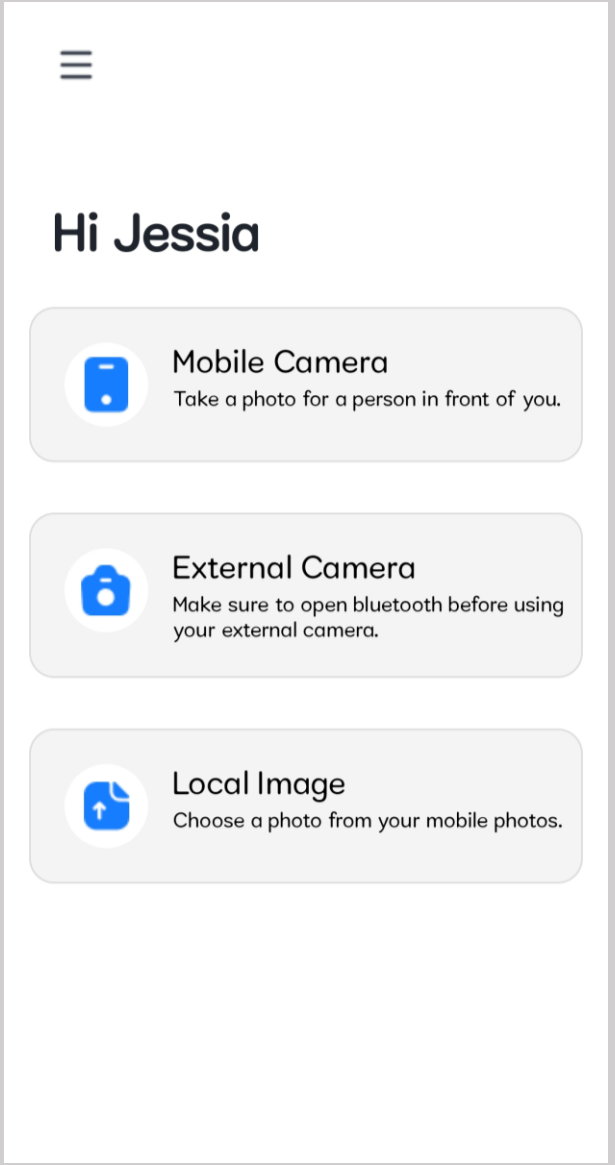
1. Button: To navigate between screens and sending the responses to the backend
1. Text Input: Used in login, sign up and adding data to database



C. Implementation
UI/UX design

Ease of use:

- Make the minimum number of screens for the user to reaching our goal.
- Focus on helping users to get through the interface when they are actually using it

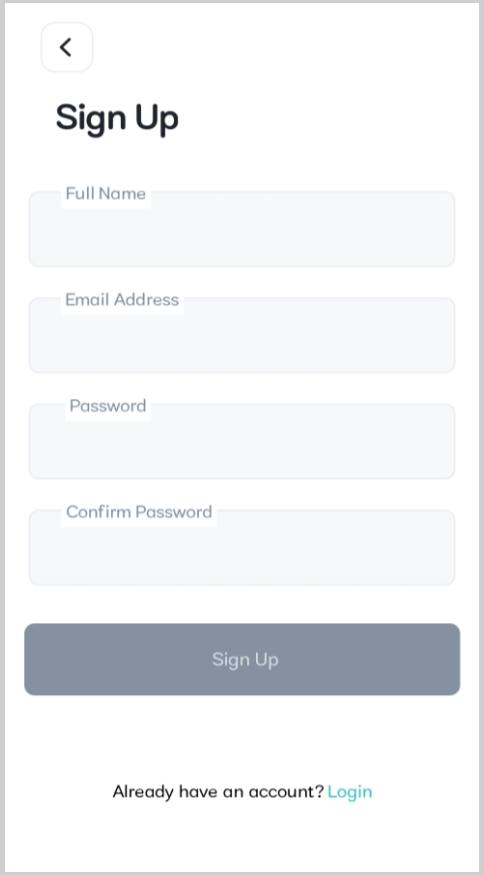


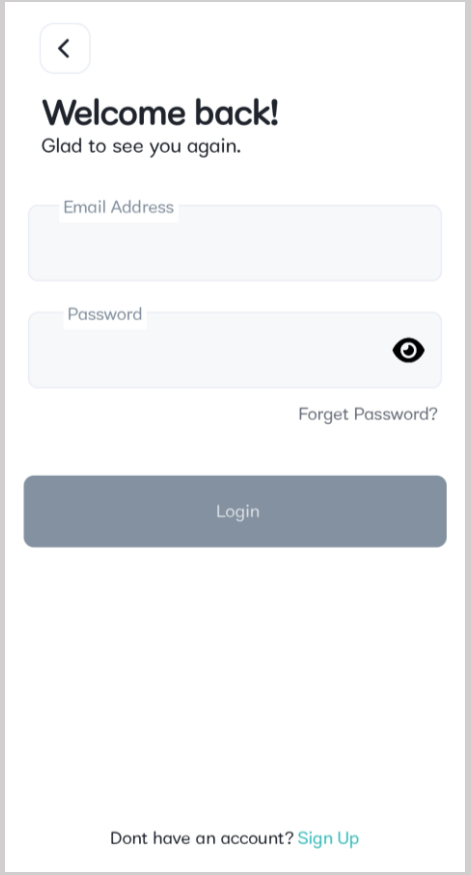
C. Implementation
UI/UX design

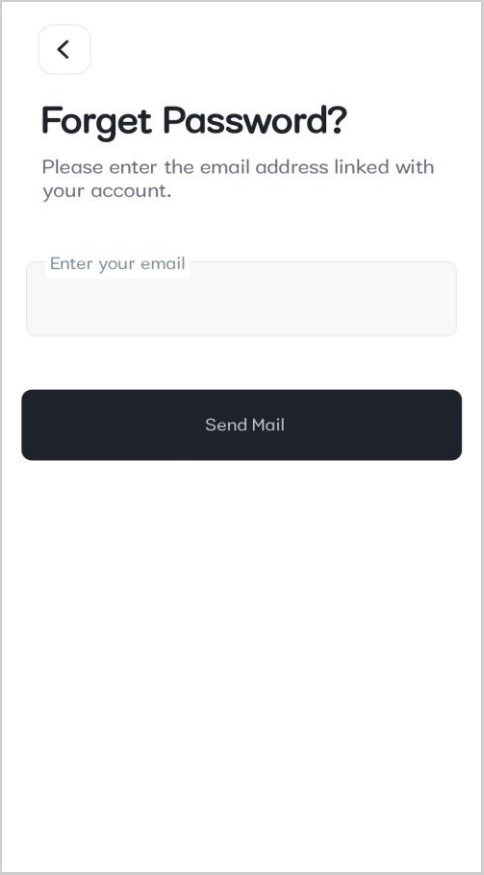
Security:

1. Login Screen
2. Sign Up Screen
3. Forget Password

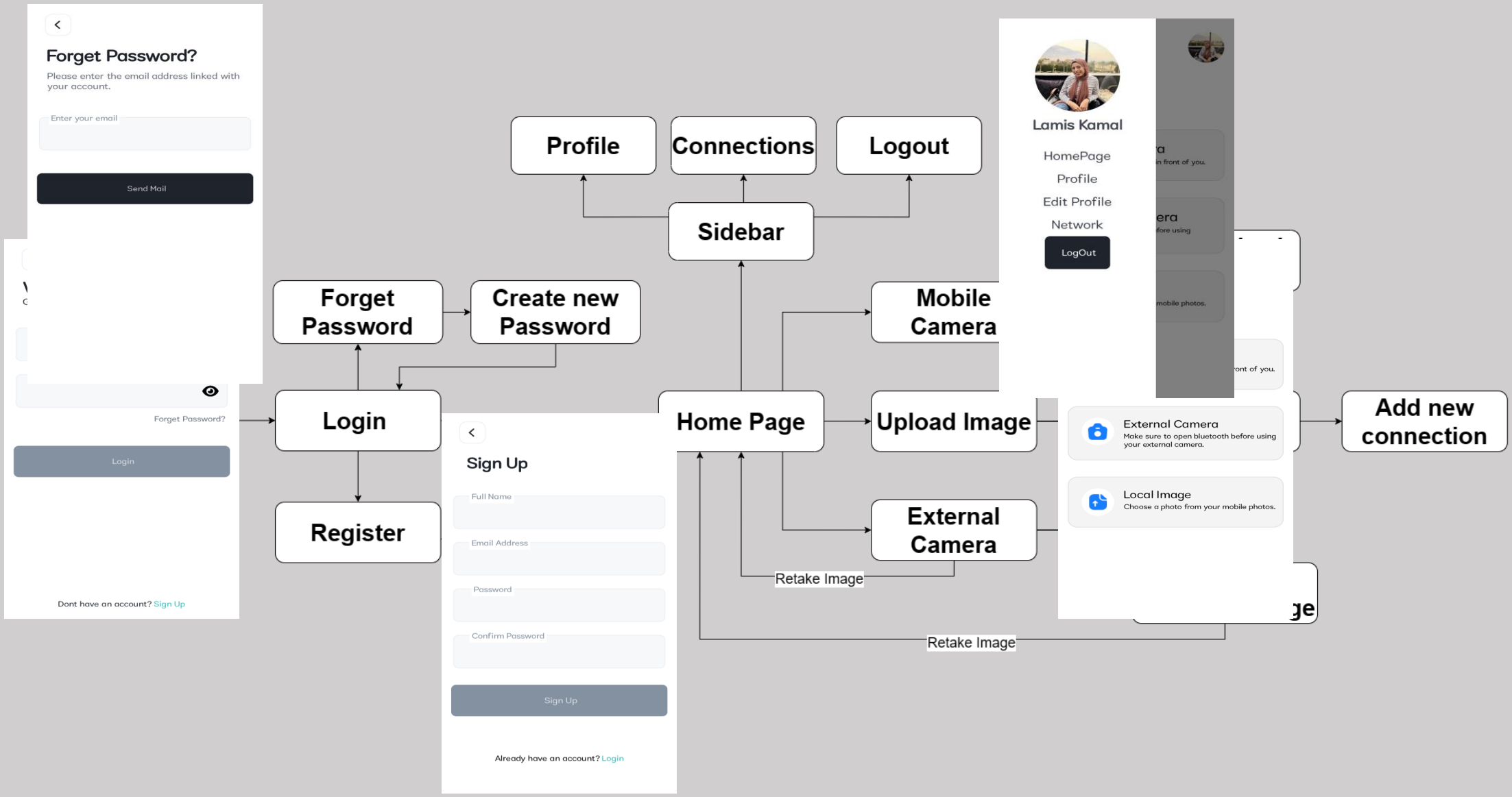
Every screen has its handling errors if one of the inputs incorrect







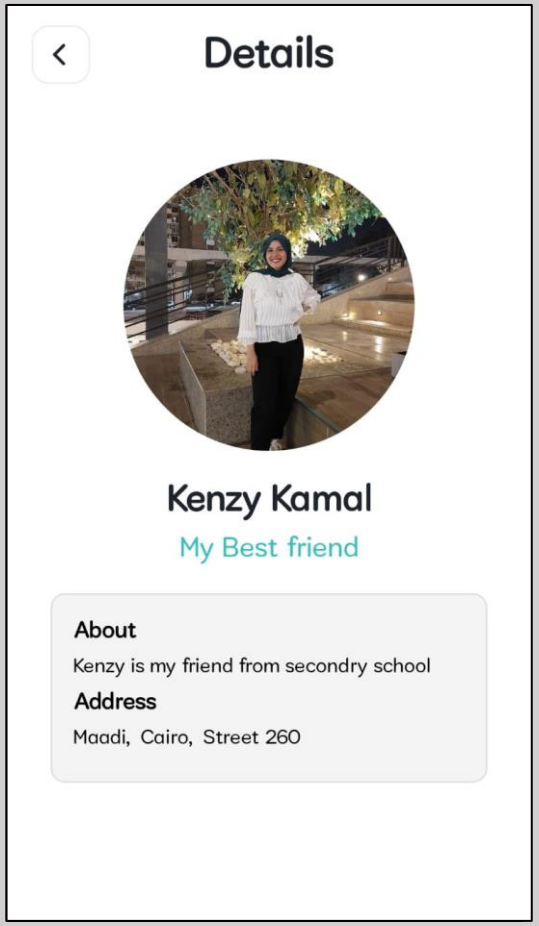
C. Implementation



C. Implementation

Sample screens from the application

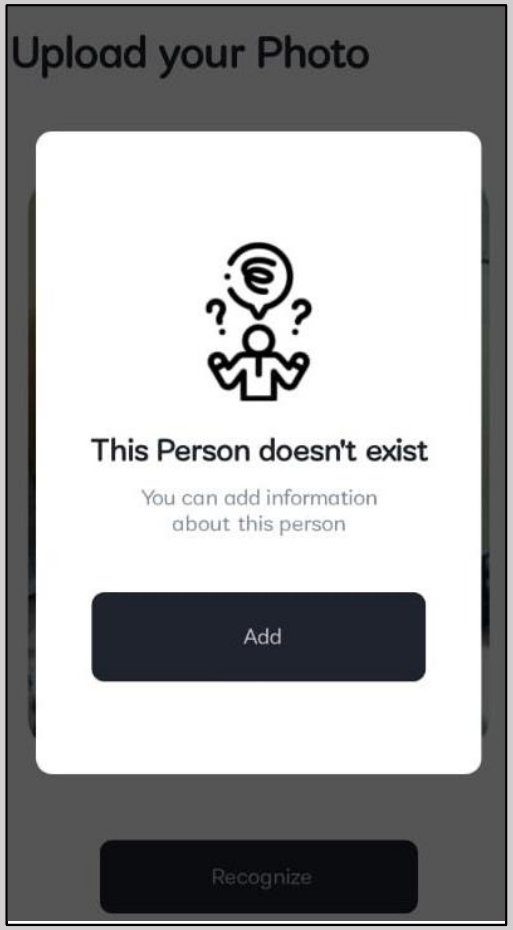
Once the image is uploaded, the application processes it and generates a response that includes the profile of the individual associated with the uploaded image.



C. Implementation

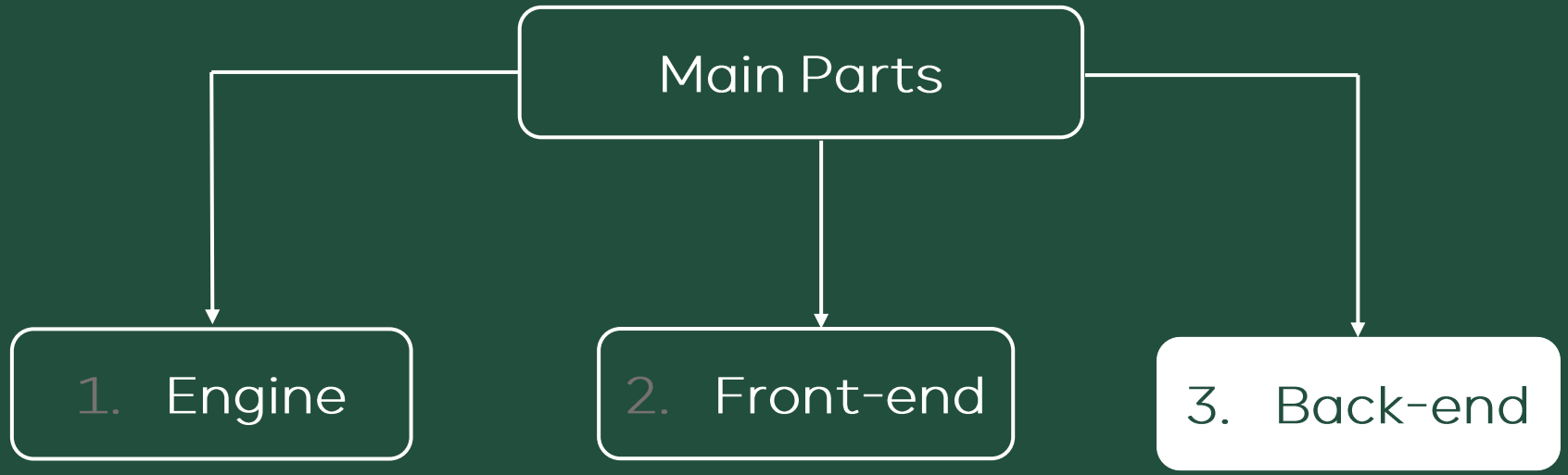
Sample screens from the application

In the event that the person is not found in the database, the application generates a response that presents the message "this person doesn't exist." Furthermore, it prompts the user to provide additional information about the person for inclusion in the database.



Methodology

In order to achieve the goal of the application, we divided it into



3. Back-end

A. Research Stage

B. Database

C. Architecture

D. Deployment

A. Research stage

- **Back-end platforms:**

a) We settled on Django and Flask because :

1. A Python language which we are familiar with.
2. To easily interact with AI models.

Flask

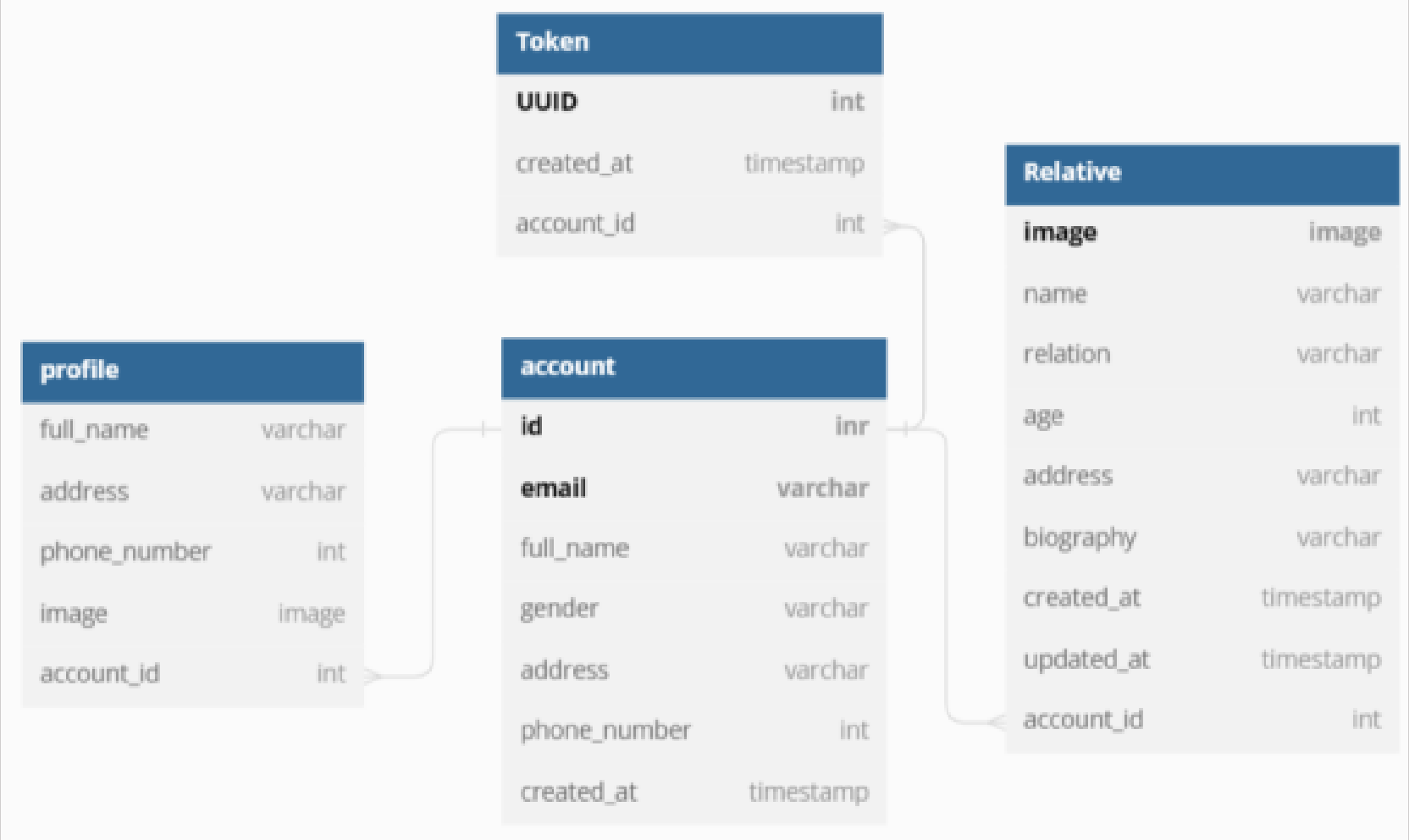
b) Then, we select Django because:

It has Django rest framework which is the best option to interact and send APIs to React Native.

Django

B. Database

- Consists of four models.
- The primary key in account model is “id”.
- Profile, Token, Relative models has a foreign key from account model.
- Used database is MySQL.



B. Database

```python
class Account(AbstractBaseUser, PermissionsMixin):
    email = models.EmailField(max_length=100, unique=True)
    password = models.CharField(max_length=100)

    USERNAME_FIELD = "email"

    is_staff = models.BooleanField(default=False)
    is_superuser = models.BooleanField(default=False)

    objects = CustomUserManager()
```

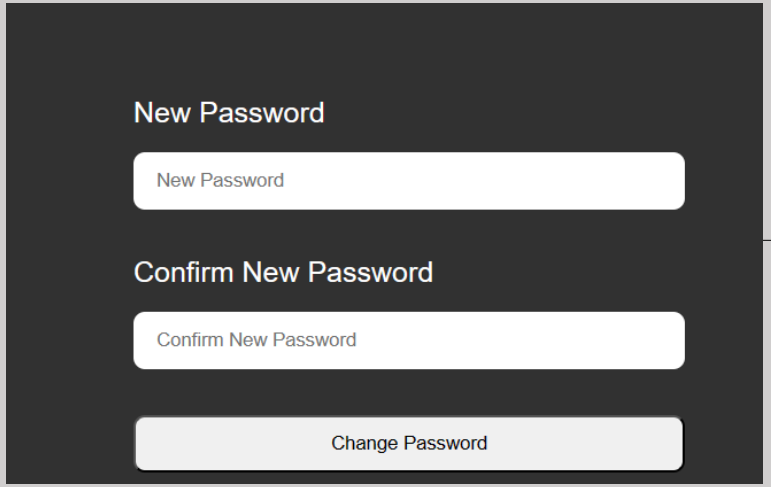
```python
class Profile(models.Model):
    account = models.OneToOneField("accounts.Account", on_delete=models.CASCAD
    fullname = models.CharField(max_length=100)
    phone = models.CharField(max_length=100, null=True, blank=True)
    address = models.CharField(max_length=250, null=True, blank=True)
    image = models.ImageField(upload_to="profiles/", null=True, blank=True)

    def __str__(self):
        return self.fullname
```

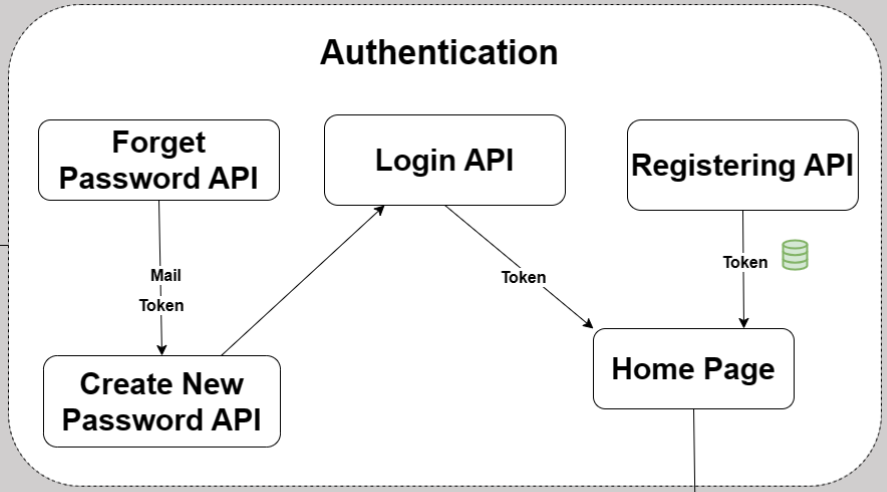
```python
class AccountToken(models.Model):
    account = models.OneToOneField(
        "accounts.Account", on_delete=models.CASCADE, related_n
    )
    uuid = models.UUIDField(default=uuid.uuid4, editable=False,
    created_at = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return str(self.uuid)
```

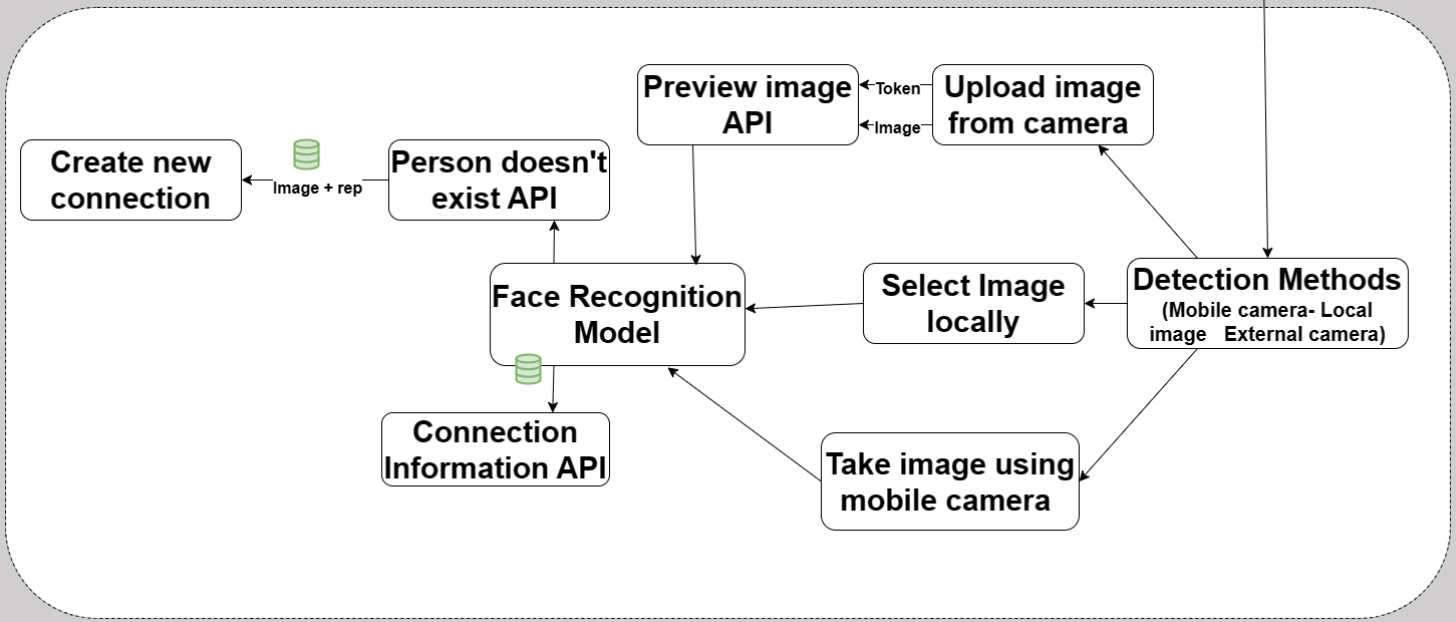
```python
class Connection(models.Model):
    account = models.ForeignKey("accounts.Account", on_delete=models.CASCADE,
    image = models.ImageField(upload_to=get_image_folder_path)
    name = models.CharField(max_length=100, null=True, blank=True)
    relation = models.CharField(max_length=100, null=True, blank=True)
    age = models.PositiveIntegerField(default=18)
    address = models.CharField(max_length=100, null=True, blank=True)
    phone_number = models.CharField(max_length=15, null=True, blank=True)
    biography = models.CharField(max_length=4000, null=True, blank=True)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
```

C. Architecture

- Tokens to achieve the security.
- Faster Recognition to achieve the ease of use.
- Two hours time limited token for forget password email.







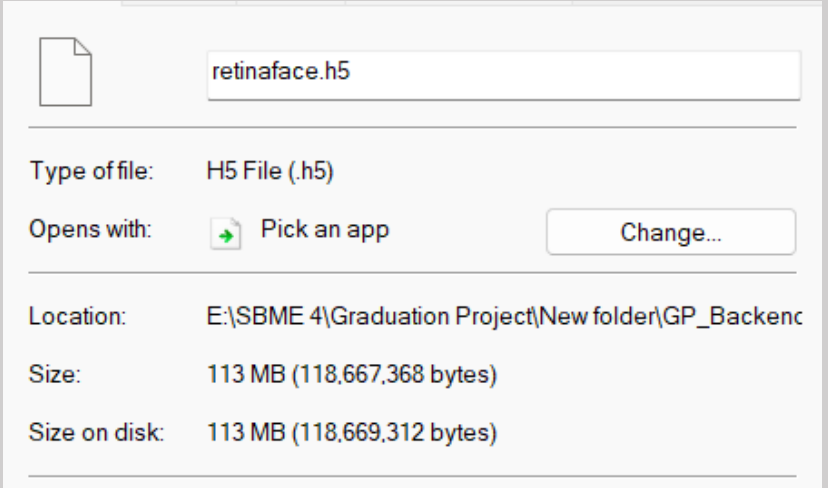
D. Deployment

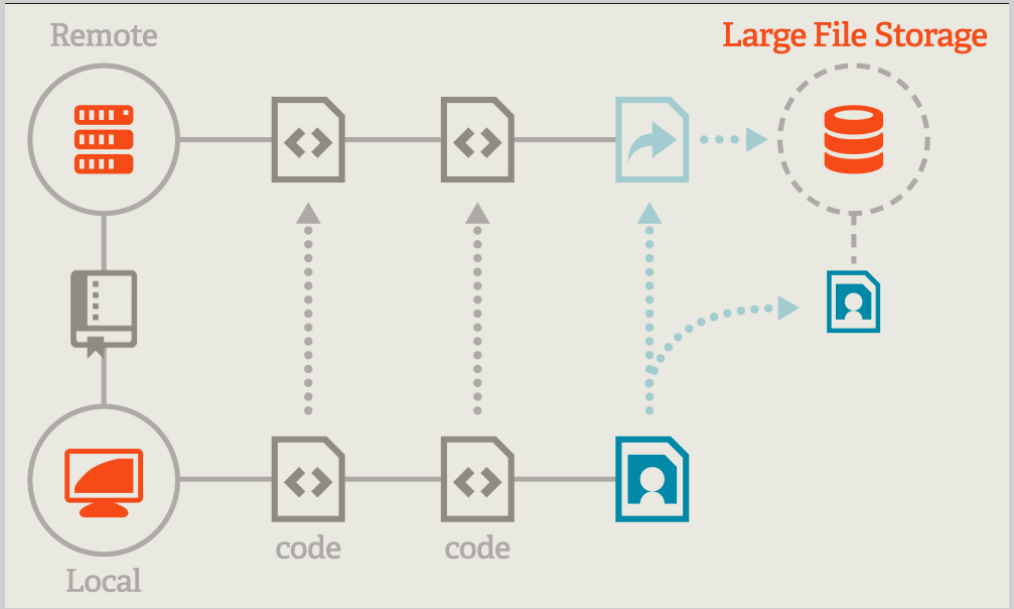
a) Challenge #1:

- Large deep learning files (ex. Facenet_model.h5, RetinaFace_model.h5)

Solution:

- Used git LFS (Large File Storage) to push files from local to remote repo





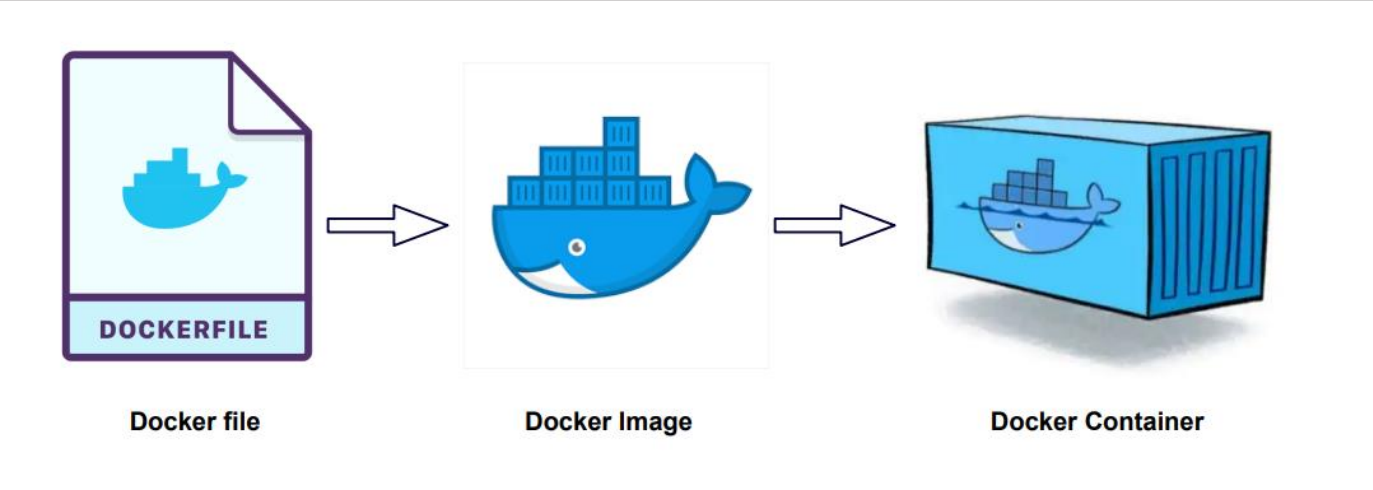
D. Deployment

b) Challenge #2:

- Large packages and libraries (ex. DeepFace, TensorFlow)

Solution:

- Create Docker file and Docker compose which is lightweight, executable container of software that includes everything needed to run an application. “Single file solution”



D. Deployment

Host Server:

- An AWS EC2 instance is used to complete the deployment as it provides cutting edge scalable computing capacity in the AWS cloud infrastructure.

c) Challenge #3:

- To generate APK, it requires a secured domain “Https” not Ip address of AWS domain “52.58.150.200”.

Solution:

- A certified domain “face_reminder.online” and obtain certification using ngnix configuration to run AWS on the new domain using certification “Https domain”.

8

Additional Feature

External Camera

It’s an Additional feature:

ESP32-CAM OV2640 Camera V2.0 for many reasons:

1. Supports both Bluetooth and Wi-Fi
2. It’s an microcontroller itself
3. Small size compared to other options
4. Compatible to send http requests to django apps
5. Affordable price (15 $)
6. Acceptable resolution (2 megapixels)

External Camera

Setup:

- Every user will have a specially setup code using:
  1. User's Wi-fi name and password.
  2. User's unique token which is created when register an account to ensure security and authorization of user.
- Connect the A0 port of camera to the ground and then upload the setup code through Arduino IDE.

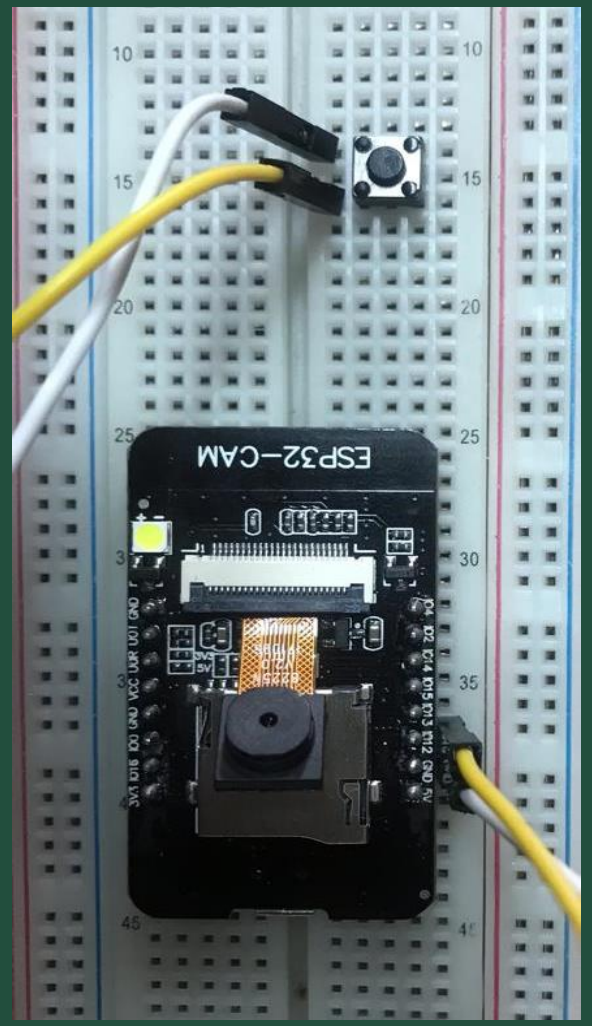
```
35  const int buttonPin = 12;
36  OneButton Button1(buttonPin, true);
37
38  //======================================== Insert your network credent
39  const char* ssid = "ahmed shaker";
40  const char* password = "ahmed@bebo123";
41  //========================================
42
43  //======================================== Variables for Timer/Millis.
44  unsigned long previousMillis = 0;
45  const int Interval = 20000; //--> Photo capture every 20 seconds.
46  //========================================
47
48  // Server Address or Server IP.
49  String serverName = "face-reminder.online";  //--> Change with your se
50  // The file path "upload_img.php" on the server folder.
51  String serverPath = "/accounts/upload/";
52  // Server Port.
53  const int serverPort = 80;
54
55  // Variable to set capture photo with LED Flash.
56  // Set to "false", then the Flash LED will not light up when capturing
57  // Set to "true", then the Flash LED lights up when capturing a photo.
58  bool LED_Flash_ON = true;
59
60  // Initialize WiFiClient.
61  WiFiClient client;
62
63  String authtoken = "d220e6e70998ef66feea0bbba43dcd93fd7abf7b";
64
65  //
66  void sendPhotoToServer() {
```

Output Serial Monitor ×

4. External Camera

How the camera working?

1. Connect camera to 5 volt power supply (using USB)
2. Click on the push button.
3. The flash is on then camera capture the image.
4. The image is sent in http request with the token of user to the backend.
5. The backend check the authentication of user and send image to front-end to perform recognition task.



9

Results

8.1 Accuracy

- Using all of the dataset classes to validate the model accuracy:
  - It gives 100% accuracy.
  - After performing some changes on the data (horizontal flipping), it gives accuracy of 100%
  - Resizing the data compromises the efficiency by 10%

8.1 Accuracy

- Using a 20 images to test the accuracy of the model
- It gives a 90% accuracy on recognizing them:
  - 18 images of them are recognized correctly including rotated images
  - 1 mismatched face
  - 1 unrecognized face

8.1 Accuracy
Sample of Rejected Data

supposed to be this

mismatched to this

There's similarity in eyebrows and lip line

8.1 Accuracy

Sample of Rejected Data

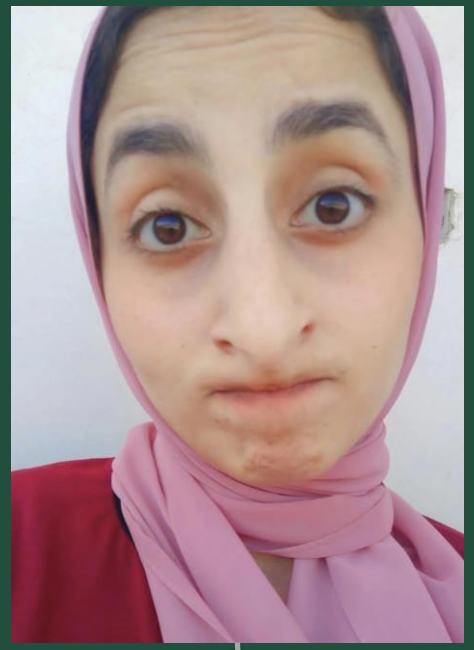
Supposed to be this

Instead it got rejected because it's an exaggerated expression

8.1 Accuracy
Sample of Rejected Data

- The unrecognizable face for exaggerated expression:

- The system's ability to recognize facial expressions has limitations.
- Exaggerated facial expressions pose a challenge for the recognition system.
- In such cases, the system often fails to accurately recognize the face.

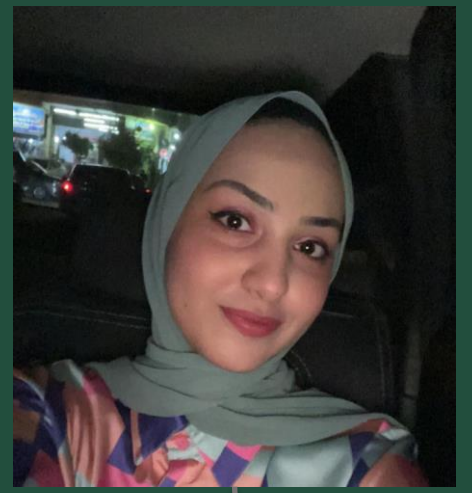


Failed to recognize

8.1 Accuracy

Sample of Accepted Data

matched to this



matched to this

8.2 Processing Time
Model Challenges

- Challenges:

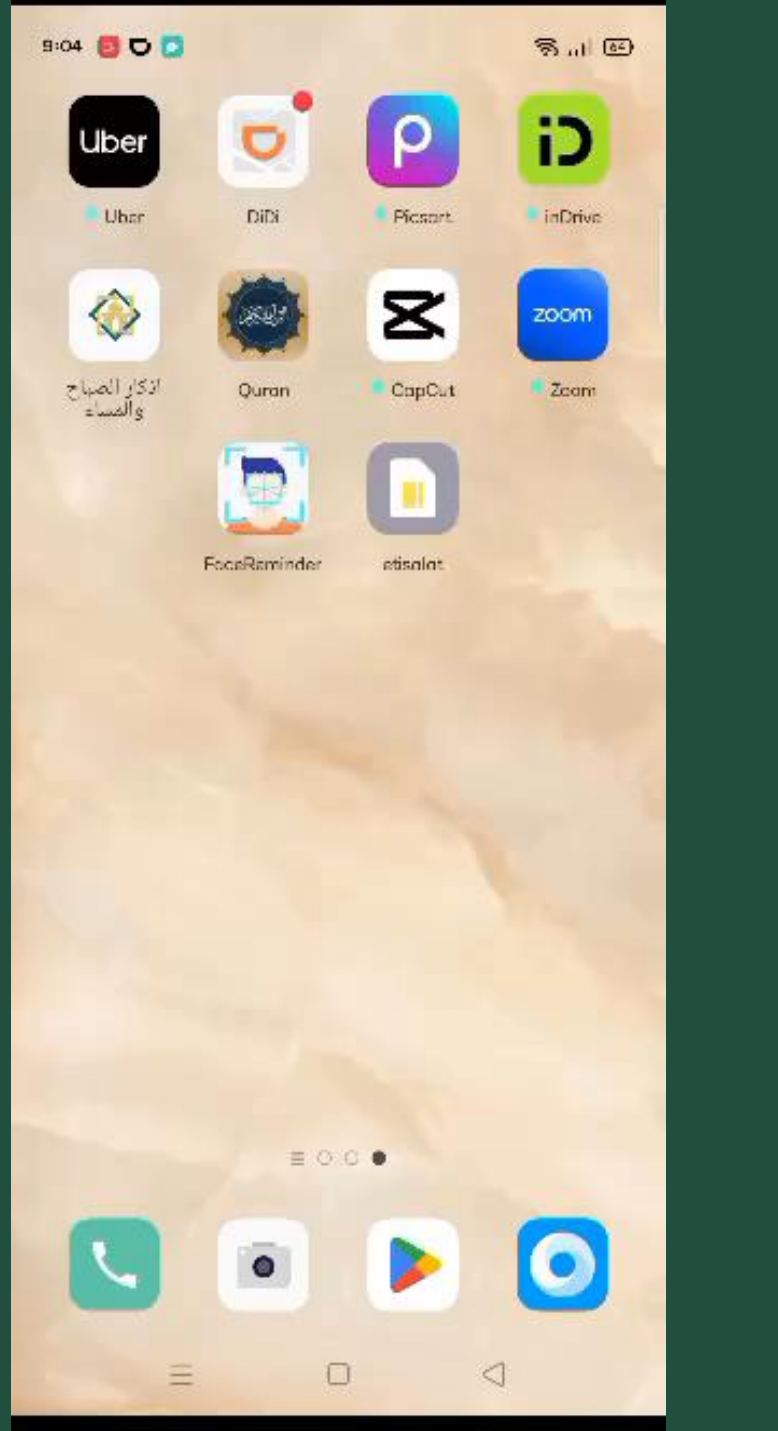
  Processing time for performing the recognition task was 7 minutes, after resizing the processing time is reduced to 3 minutes, however this decreased the efficiency of the model by 10%.

8.2 Processing Time
Justification and Modifications

- Justification:
  - The model process the entire dataset each time
- Modifications:
  - Storing the faces representation first time and then updating on it
- Results:
  - Performing the task in average of 10 seconds

10

App Demo


9:04
Uber
Uber
DiDi
Picsart
inDrive
اذكار الصباح والمساء
Quran
CapCut
zoom
Zoom
FaceReminder
etisalat

11

Future Work

Future Work

- Implementation of voice control specifically tailored to cater to the needs of visually impaired individuals.
- Enabling real-time recognition.
- Adding a feature that enables the grouping of individuals based on specific events, time, and locations.
- Connecting the camera to glasses

12

References

References

1. J. Deng, J. Guo, E. Ververas, I. Kotsia and S. Zafeiriou, "RetinaFace: Single-Shot Multi-Level Face Localisation in the Wild," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 5202-5211, doi: 10.1109/CVPR42600.2020.00525.
2. F. Schroff, D. Kalenichenko and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 815-823, doi: 10.1109/CVPR.2015.7298682.
3. S. I. Serengil and A. Ozpinar, "LightFace: A Hybrid Deep Face Recognition Framework," 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), Istanbul, Turkey, 2020, pp. 1-5, doi: 10.1109/ASYU50717.2020.9259802.

Questions?

Thank you