“FACE GRAB”

Minor Project

Submitted by: -

Name: **PRIYANSH VERMA**	Roll No: 0206cs181**118**
Name: **TULIKA JAIN**	Roll No: 0206cs181**174**
Name: **UMEED CHANDEL**	Roll No: 0206cs181**177**

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GYAN GANGA INSTITUTE OF TECHNOLOGY & SCIENCES
JABALPUR (M.P.)
RAJIV GANDHI PRODYOGIKI VISHWAVIDYALAYA,
BHOPAL (M.P.)
June- 2021

TABLE OF CONTENTS

1. INTRODUCTION

Purpose of Project

The purpose of FACE GRAB web app is to automate the surveillance process. We don’t need to go through each and every footage recorded by our installed cameras to find a person anymore, FACE GRAB will do that for us.

Project and Product Overview

FACE GRAB is a web application which identifies individuals with their unique id using computer vision, face recognition library and stores all the details in a database. When the admin wants to search a particular individual, they can access his/her details using the person's unique id. The details include name, location, camera id, date, time and photo.

Intended Audience

FACE GRAB’s intended audience are the individuals who are authorized to have access to the data of all the members in an organization and monitor the activities happening within an organization.

Team Architecture

Our team’s name is TRIPOD as it consists of three individuals namely PRIYANSH VERMA, TULIKA JAIN and UMEED CHANDEL.

We have divided all our tasks into three categories each handled by one individual.

1. Backend: It involves the developing main script and logic behind the application using languages like python.

2. Frontend: This includes designing the user interface which is the bridge between user and our application.

3. Database and Storage: It’s about storing and managing the applications data for effective and efficient use

Overall Description

FACE GRAB stores the details captured by the installed cameras in the particular premises in the database and outputs the information when the admin searches the individual using a unique id. Thus, automating the surveillance process.

2. DURATION

Timeline

Our project was majorly divided into three tasks, the backend took around 70 hours. Frontend and database we’re summed up together in around 90 hours. The project still requires additional hours to be optimized and upgraded.

Overall: 160 hours approximately.

3. REQUIREMENTS

Functional Requirements

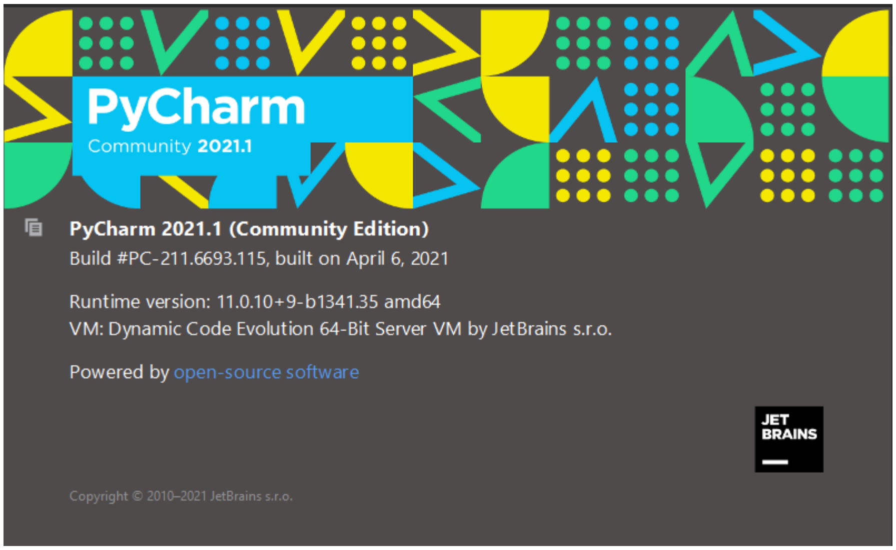
Hardware Interface:

1. Computer system
2. High quality camera.
3. Adequate Bandwidth
4. Operating system
5. RAM
6. Disk Space
7. Optimal speed of the processor

Software Interface:

1. Backend- **PyCharm IDE 2021.1** (Community Edition)

Python version **3.9.5**
PyCharm is an integrated development environment used specifically for the Python language. Written in java and python.



PyCharm 2021.1 (Community Edition)
Build #PC-211.6693.115, built on April 6, 2021

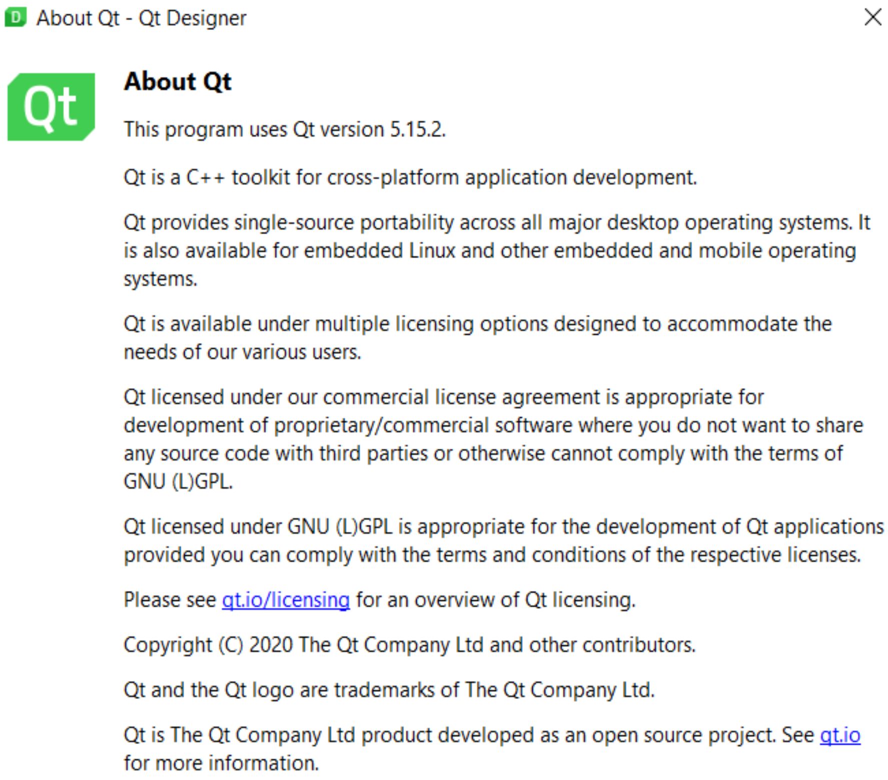
Runtime version: 11.0.10+9-b1341.35 amd64
VM: Dynamic Code Evolution 64-Bit Server VM by JetBrains s.r.o.

Powered by open-source software

Copyright © 2010–2021 JetBrains s.r.o.

2. Frontend - **Qt Designer 5.15**

Qt Designer is a Qt tool that provides you with a what-you-see-is-what-you-get (WYSIWYG) user interface to create GUIs for your PyQt applications productively and efficiently.



About Qt - Qt Designer

About Qt

This program uses Qt version 5.15.2.

Qt is a C++ toolkit for cross-platform application development.

Qt provides single-source portability across all major desktop operating systems. It is also available for embedded Linux and other embedded and mobile operating systems.

Qt is available under multiple licensing options designed to accommodate the needs of our various users.

Qt licensed under our commercial license agreement is appropriate for development of proprietary/commercial software where you do not want to share any source code with third parties or otherwise cannot comply with the terms of GNU (L)GPL.

Qt licensed under GNU (L)GPL is appropriate for the development of Qt applications provided you can comply with the terms and conditions of the respective licenses.

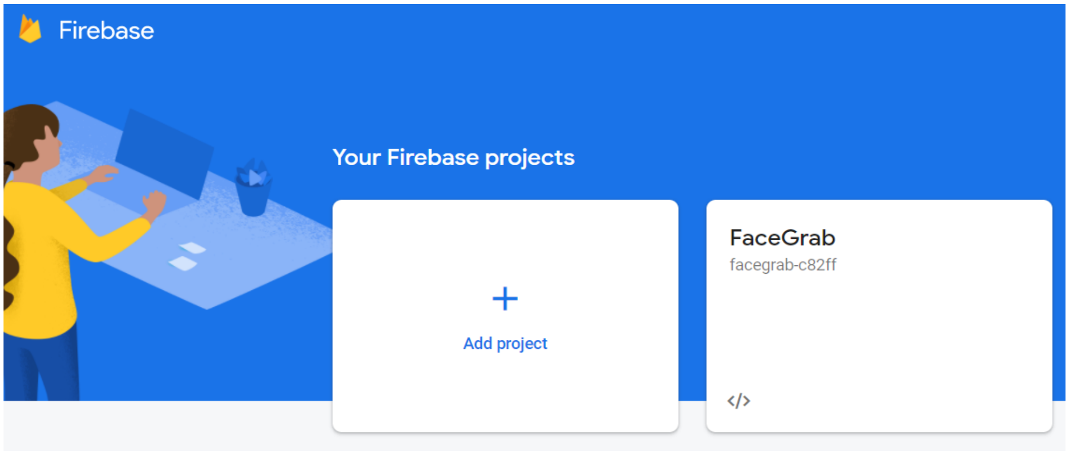
Please see qt.io/licensing for an overview of Qt licensing.

Copyright (C) 2020 The Qt Company Ltd and other contributors.

Qt and the Qt logo are trademarks of The Qt Company Ltd.

Qt is The Qt Company Ltd product developed as an open source project. See qt.io for more information.

3. Database and Storage - **Firebase**



Firebase is a platform developed by Google for creating mobile and web applications.

Nonfunctional Requirements

1. Usability requirement

a) User friendly Interface - Easy to navigate as it's designed in a simplified manner and returns the surveillance details.

b) Information Scent - Admin should provide the correct reference id of a person to access his details.

c) Effective - It's a data-driven web App, as its effectiveness depends on accuracy and availability of data.

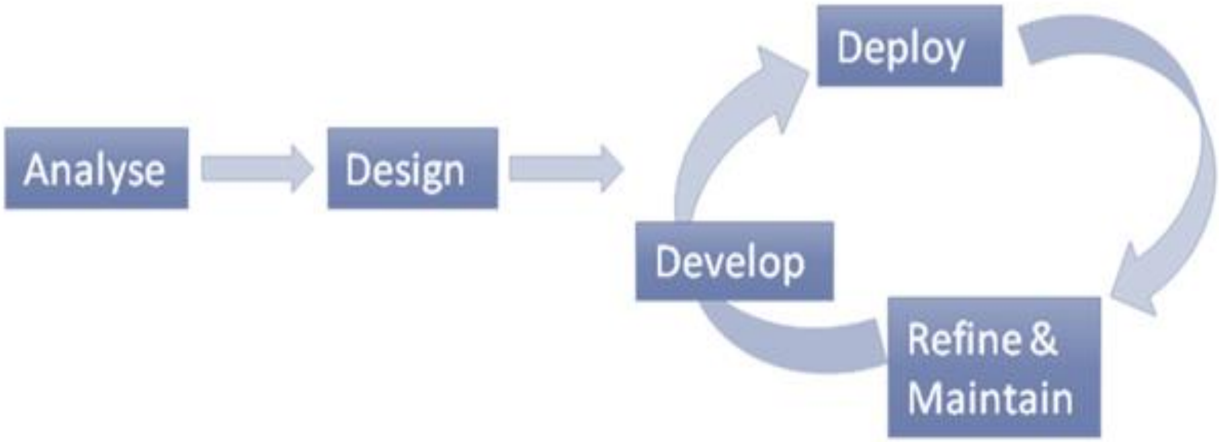
d) Error tolerant - it's design and execution process are so keenly observed and done that it can handle any occurrence of error.

e) Flow - Flow within the Face Grab is smooth and easy, movement from one page to another is uniquely planned and defined with no ambiguity.

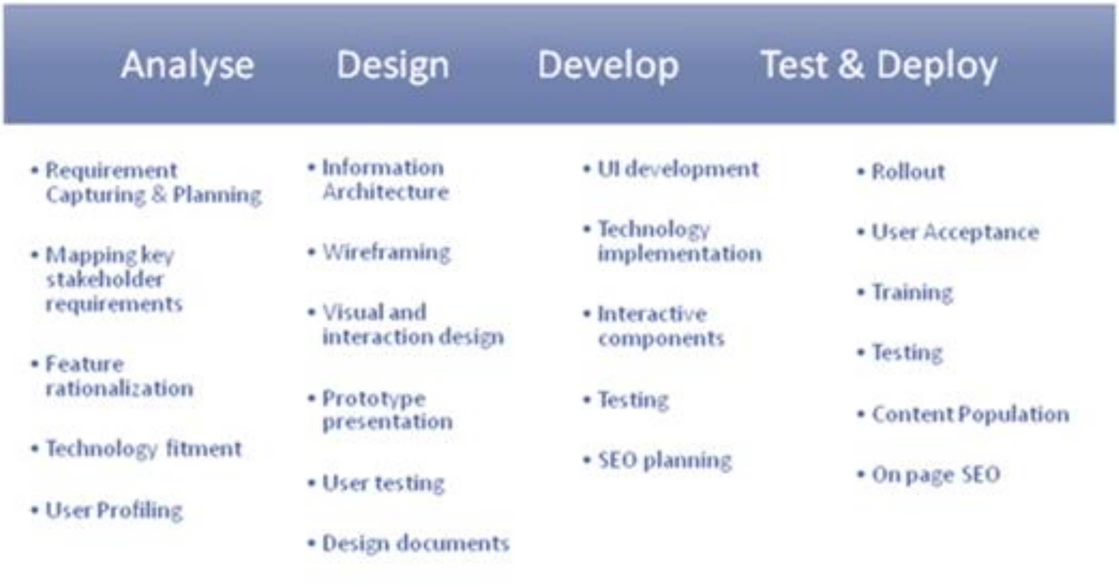
2. Security and control requirements

The access of our web application "Face Grab" will only be assigned to a single user, the Admin. Who'll be able to use the web App and access it's linked Firebase.

3. Project methodology



The phases of the project will be as stated below:



Analyse	Design	Develop	Test & Deploy
• Requirement Capturing & Planning • Mapping key stakeholder requirements • Feature rationalization • Technology fitment • User Profiling	• Information Architecture • Wireframing • Visual and interaction design • Prototype presentation • User testing • Design documents	• UI development • Technology implementation • Interactive components • Testing • SEO planning	• Rollout • User Acceptance • Training • Testing • Content Population • On page SEO

4. DESIGN TECHNIQUES

QT DESIGNER

Qt Designer is the Qt tool for designing and building graphical user interfaces (GUIs) with Qt Widgets. You can compose and customize your windows or dialogs in a what-you-see-is-what-you-get (WYSIWYG) manner, and test them using different styles and resolutions.

PYTHON

Python is an interpreted object-oriented high level programming language with dynamic semantics. It's high-level built-in data structure, combined with dynamic typing and dynamic binding make it very attractive for Rapid Application Development, as well as for use as a scripting for glue language to connect existing components together.

FIREBASE

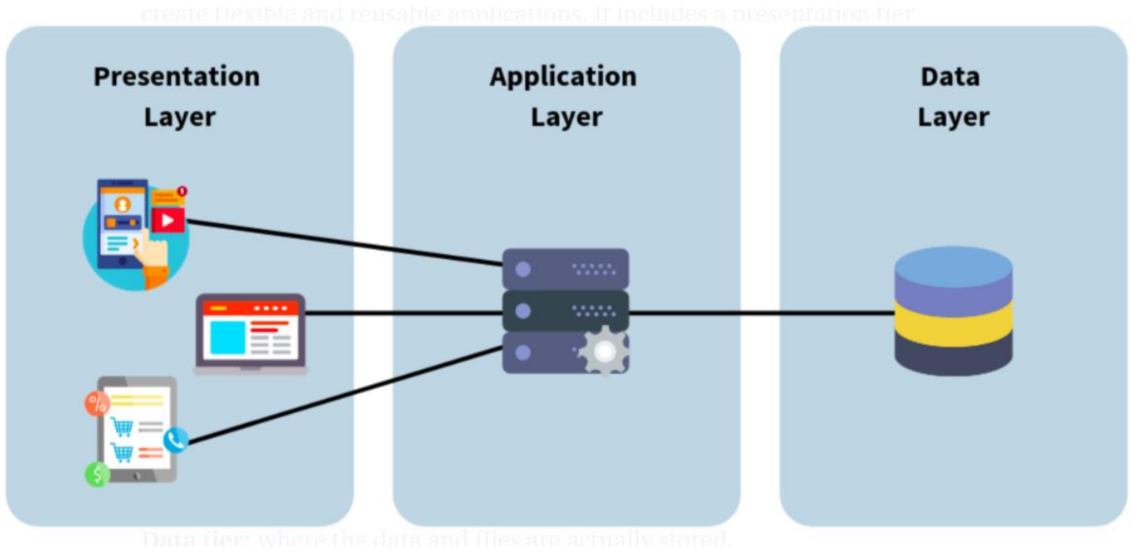
Google Firebase is a Google-backed application development software that enables developers to develop iOS, Android and Web apps. Firebase provides tools for tracking, analytics, reporting and many more.

5. TIER ARCHITECTURE

The FACE GRAB web app can be identified as three tier architecture the three layers being:

Presentation Layer

In this layer the code defines the basic designing of a front-end view of the web applications as well as calling of the functions of other layers so that they can be integrated with each other. (Code generated using QT Designer)


create flexible and reusable applications. It includes a presentation tier
Presentation Layer
Application Layer
Data Layer
Data tier: where the data and files are actually stored.

Application Layer

In this the function of the Application layer which accepts the data from the application layer and passes it to the data layer. All the logic that acts as an interface between Client layer and Data Access Layer. (Functions and classes made using PyCharm)

Database Layer

This is the data layer function, which receives the data from the Application layer and performs the necessary operation into the database. (Database and storage on Firebase)

6. SOFTWARE PROCESS MODEL

Why not Evolutionary models?

As it is a combination of Iterative and Incremental model of software development life cycle, it is better suited for large as well as mission critical projects. The FACE GRAB web app is identified as a minor project which may expand with time but can't be categorized into mission critical projects.

Why not the Waterfall model?

This was the first Process Model to be introduced and is linear and sequential which indicated no turning back. Once an application is in the testing stage, it is very difficult to go back and change something. As requirements of The FACE GRAB web app are at a moderate to high risk of changing, the waterfall model is also not well suited.

Why Incremental RAD model?

The Rapid Application Development model is a type of incremental model. In these components or functions are developed in parallel as if they were mini projects, inapplicable for small projects as cost of modeling and automated code generation is very high. And rectifying a problem in one unit requires correction in all the units and consumes a lot of time.

Observation

Keeping in mind the changing requirement and moderate scale of the "FACE GRAB" web app as a minor project, to be completed within a short time span and as Agile stands as one of the most popular approaches to project management. To get better control and improved project predictability, for reduced risks and increased flexibility we've pursued The Agile Methodology.

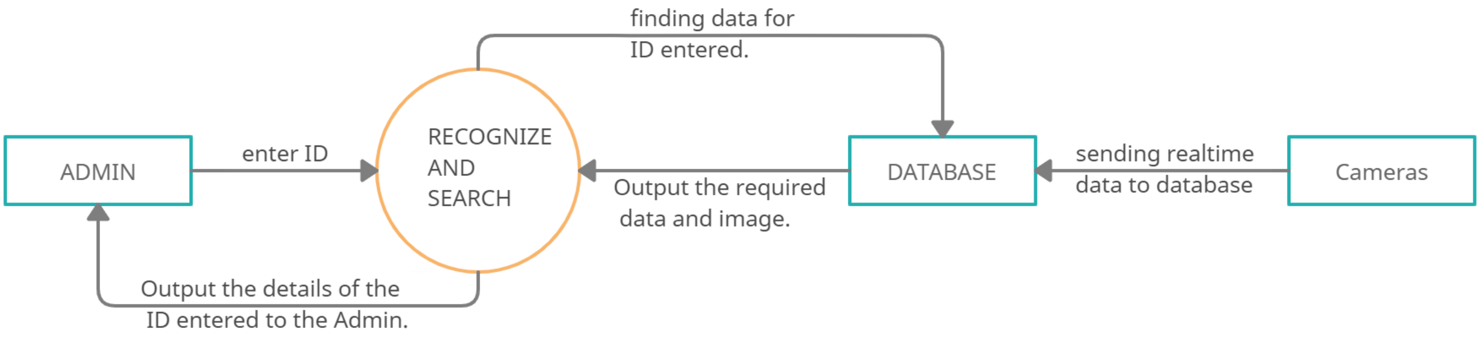
Determining project feasibility

- FACEGRAB is the automation process of surveillance so it focuses on a particular task rather than multitasking at a time.

- The platform is easy to be dealt with and suitable for the cause. Its navigation and design planning makes it more feasible for the users to use.

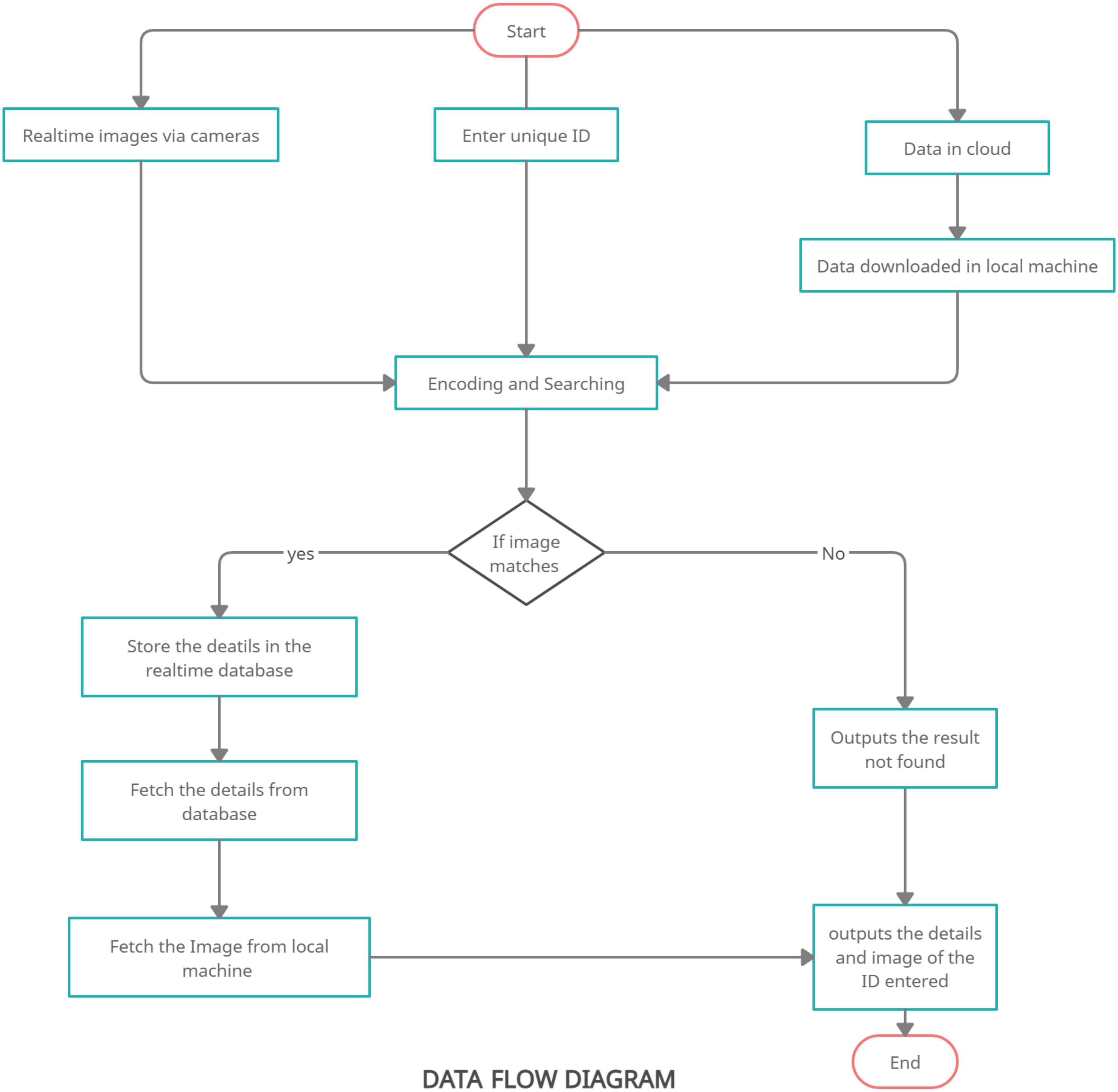
7. DESIGN

DATA FLOW DIAGRAM



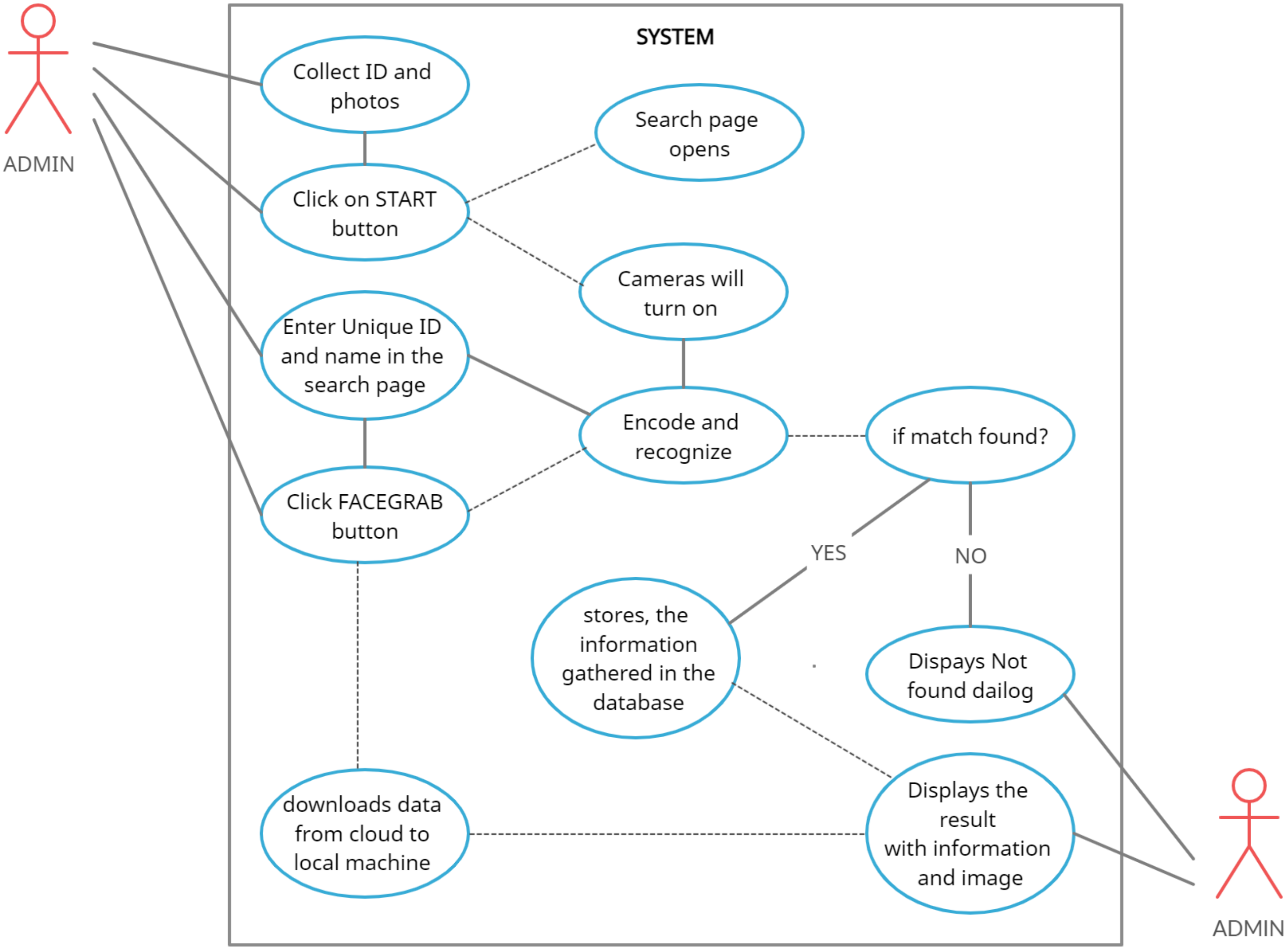
**DATA FLOW DIAGRAM
LEVEL-0**

A data-flow diagram is a way of representing a flow of data through a process or a system.


Start
Realtime images via cameras
Enter unique ID
Data in cloud
Data downloaded in local machine
Encoding and Searching
If image matches
yes
No
Store the deatils in the realtime database
Fetch the details from database
Fetch the Image from local machine
Outputs the result not found
outputs the details and image of the ID entered
End

DATA FLOW DIAGRAM
LEVEL - 1

USE CASE DIAGRAM

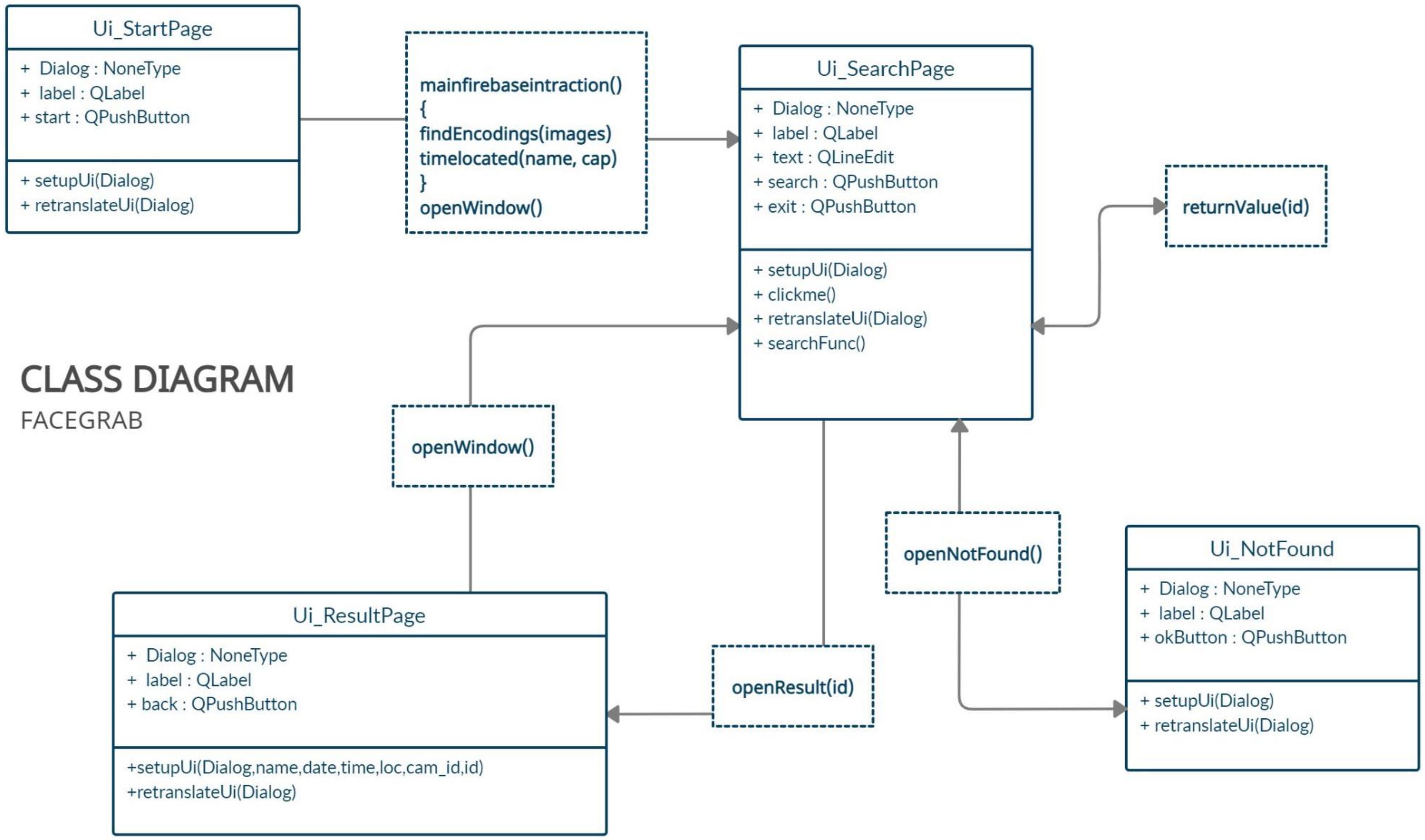


USE CASE DIAGRAM

A use case diagram is a way to summarize details of a system and the users within that system. It is generally shown as a graphic depiction of interactions among different elements in a system

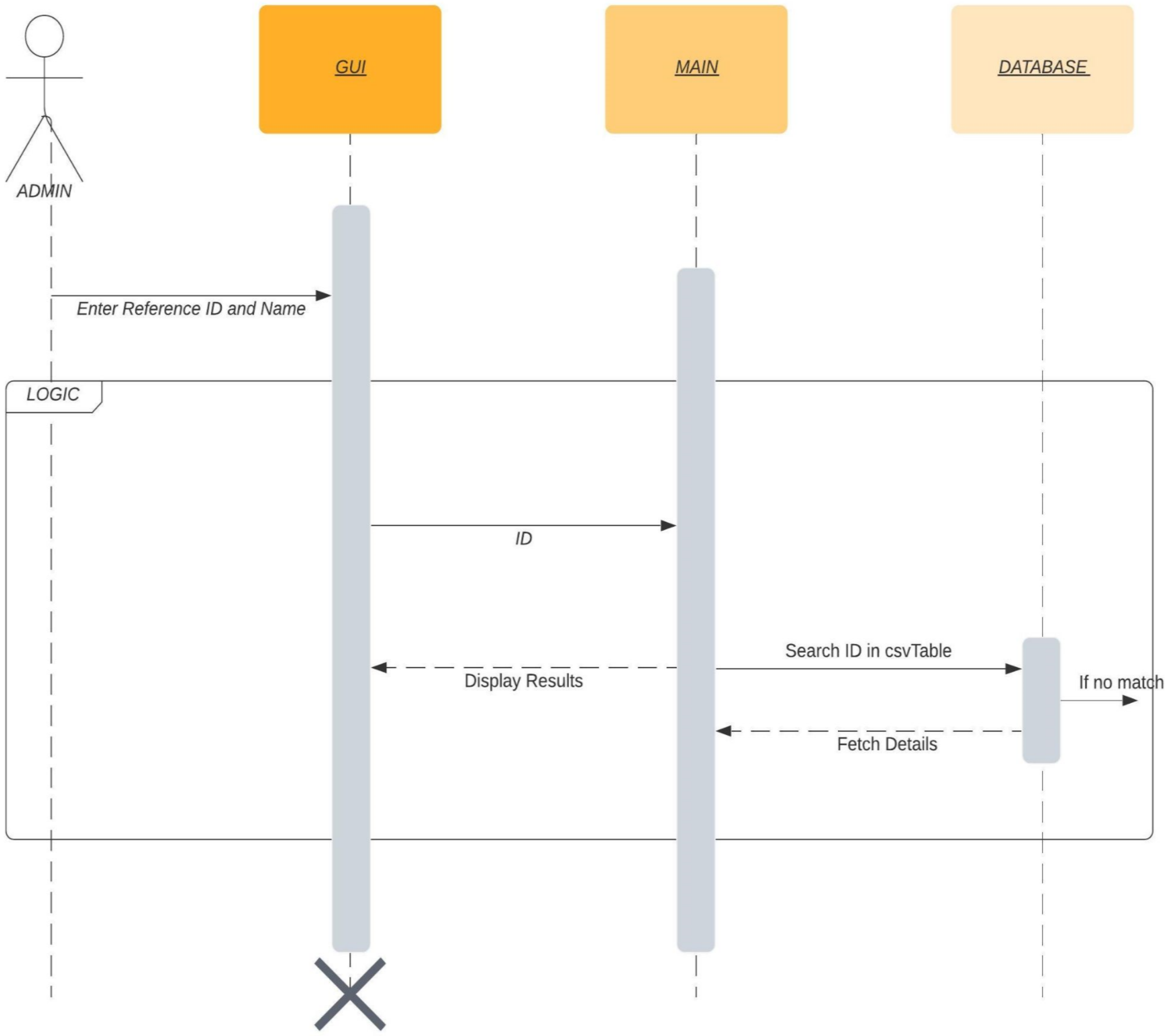
CLASS DIAGRAM

It represents the static view of an application



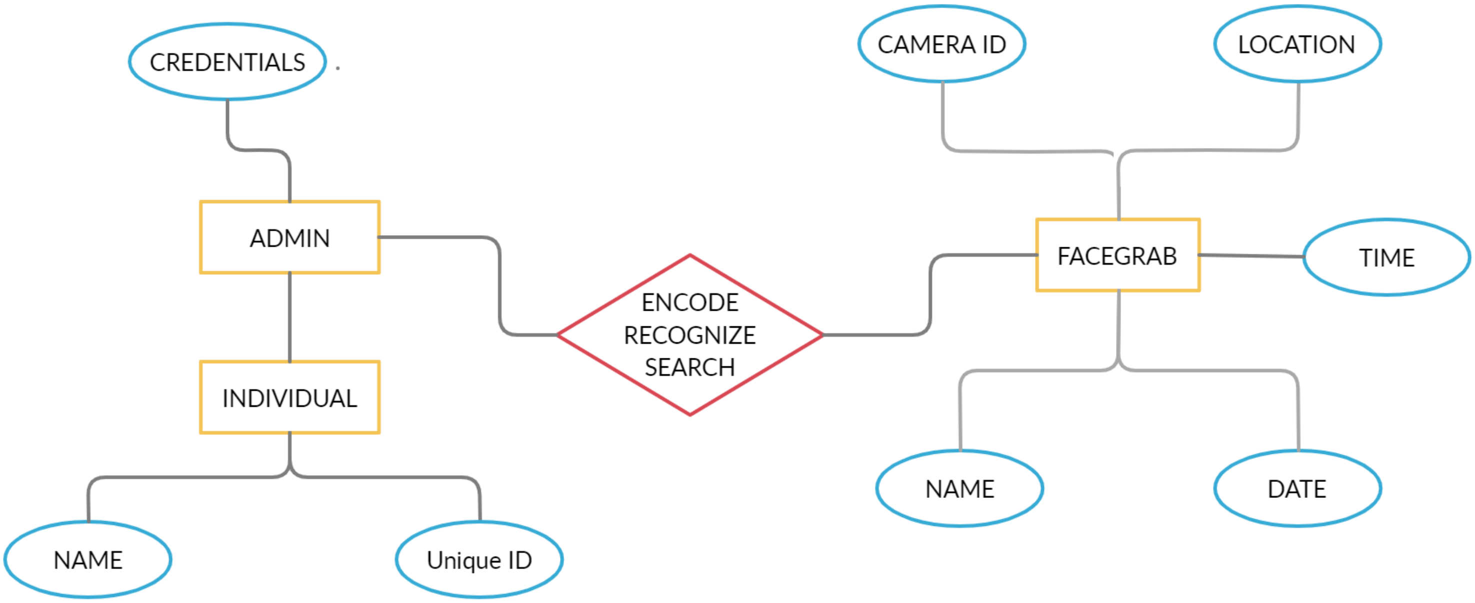
SEQUENCE DIAGRAM

It’s an interaction diagram because it describes how—and in what order—a group of objects works together.



E-R DIAGRAM

ER diagram is a diagram that displays the relationship of entity sets.ER diagrams are created based on three basic concepts: entities, attributes and relationships.



E-R DIAGRAM

8. FRONTEND

‘Front-end" typically means the parts of the project a user interacts with--such as the graphical user interface or command line.

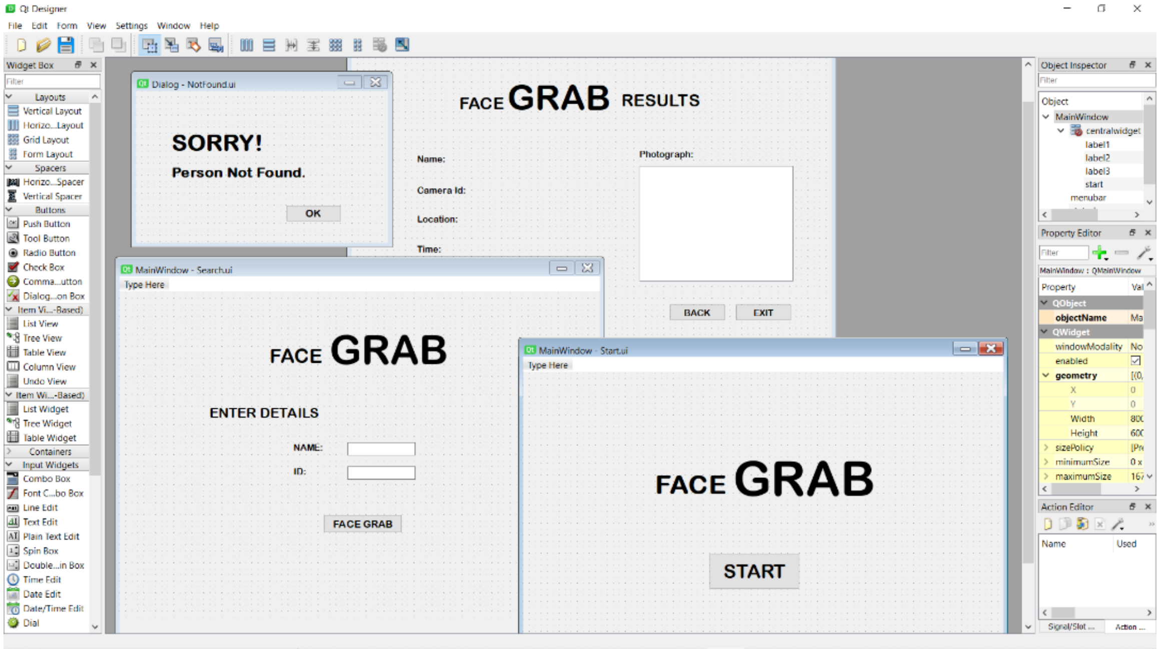
Front-end of Face Grab is designed using the pyqt5 tool i.e., QT designer. **PyQt** is one of the most popular **Python** bindings for the **Qt** cross-platform C++ framework.

PyQt5 has provided a tool called ‘QT Designer’ to design the front-end by drag and drop method so that development can become faster.

Command to install pyqt5 and its tools for python interpreter:

```
$ pip install pyqt5
$ pip install pyqt5 -tools
```

In Qt designer, designing was initiated using drag and drop method, which created a .ui file which is an XML file, a basic layout for the front-end pages.



By using the below commands the .ui file was converted to .py (inside your .ui file location)

```
$ pyuic5 -x Start.ui -o Start.py
```

Images included in the frontend design are compiled inside a .qrc file, which are converted to python code using below command (inside your .ui file location):

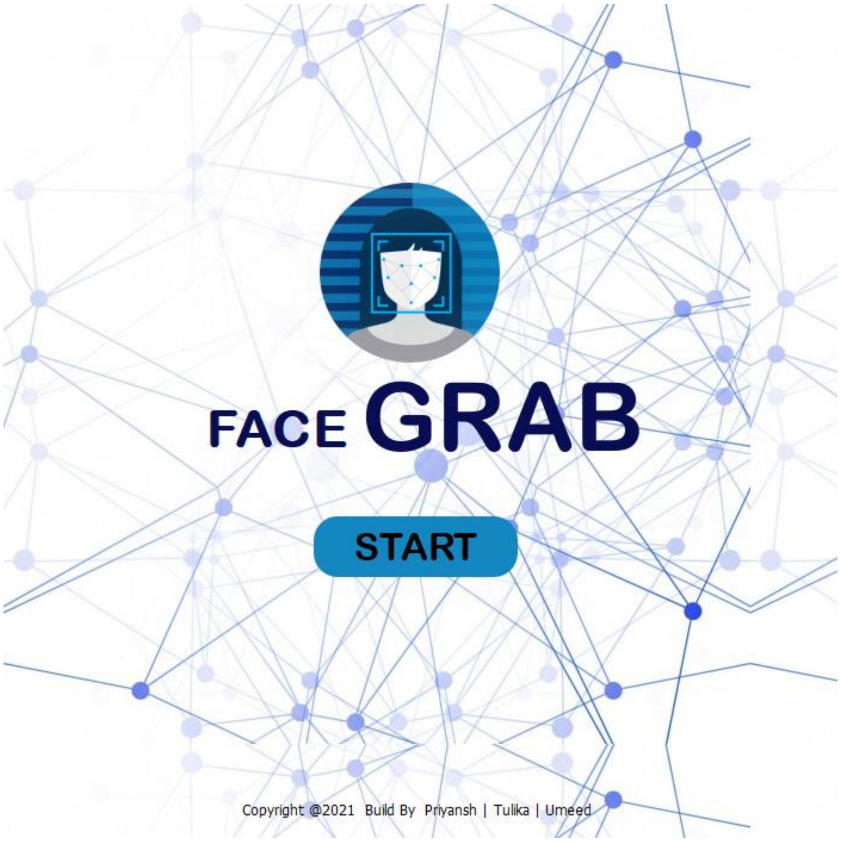
```
$ pyrcc5 -x imgs.qrc -o imgs_rc.py
```

Face Grab consists of 4 modules: Start Dialog, Search Dialog, Result Dialog, Not Found Dialog.

All the Dialog boxes include common attributes which are QLabel, QPushButton, QTextFeild, Background images and a logo.

1. *START PAGE*:

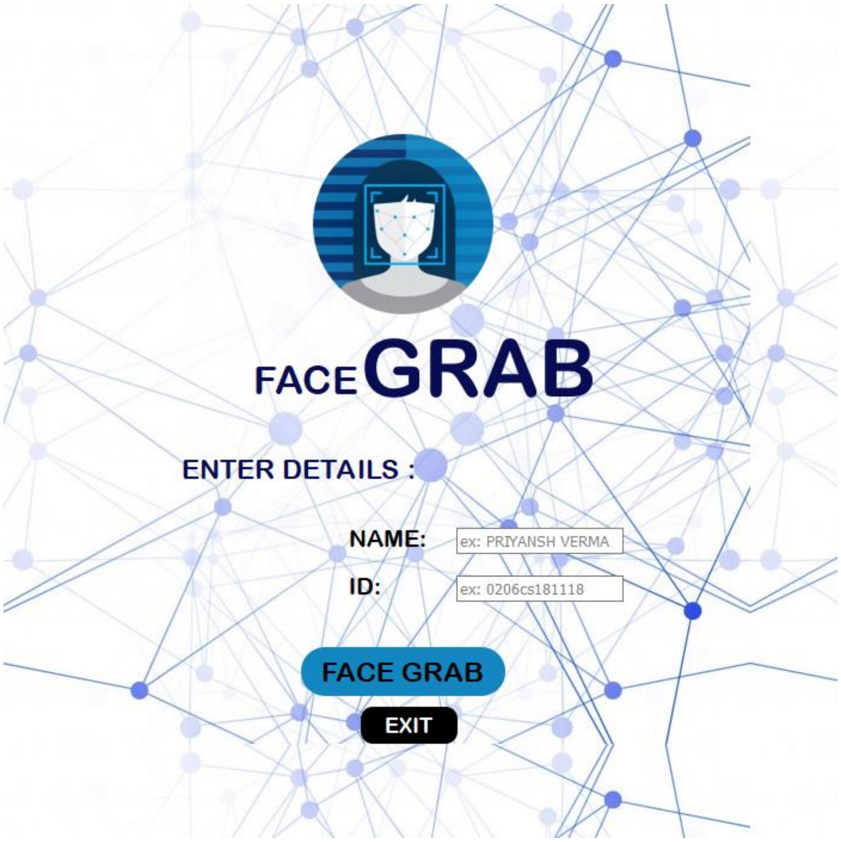
Clicking Start button inside the start page initiates the main event of the web app, cameras installed turns on and start fetching data, encoding starts and Search page opens.



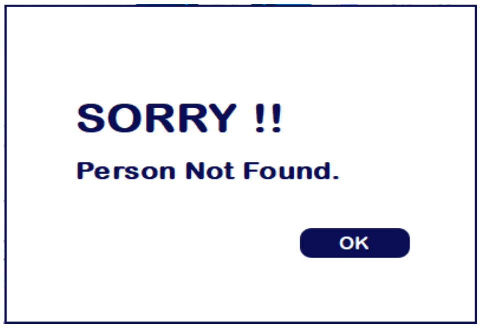
Copyright @2021 Build By Priyansh | Tulika | Umeed

2.SEARCH PAGE

- In the search window, Admin needs to enter the unique reference ID and name to find the particular individual and click the FACEGRAB button.
- By this Face Grab will search the person’s Id entered in the database
- and fetch the name and photo of that person, opening the result page.
- Clicking the exit button in this page will close the whole app.



3.NOT FOUND PAGE



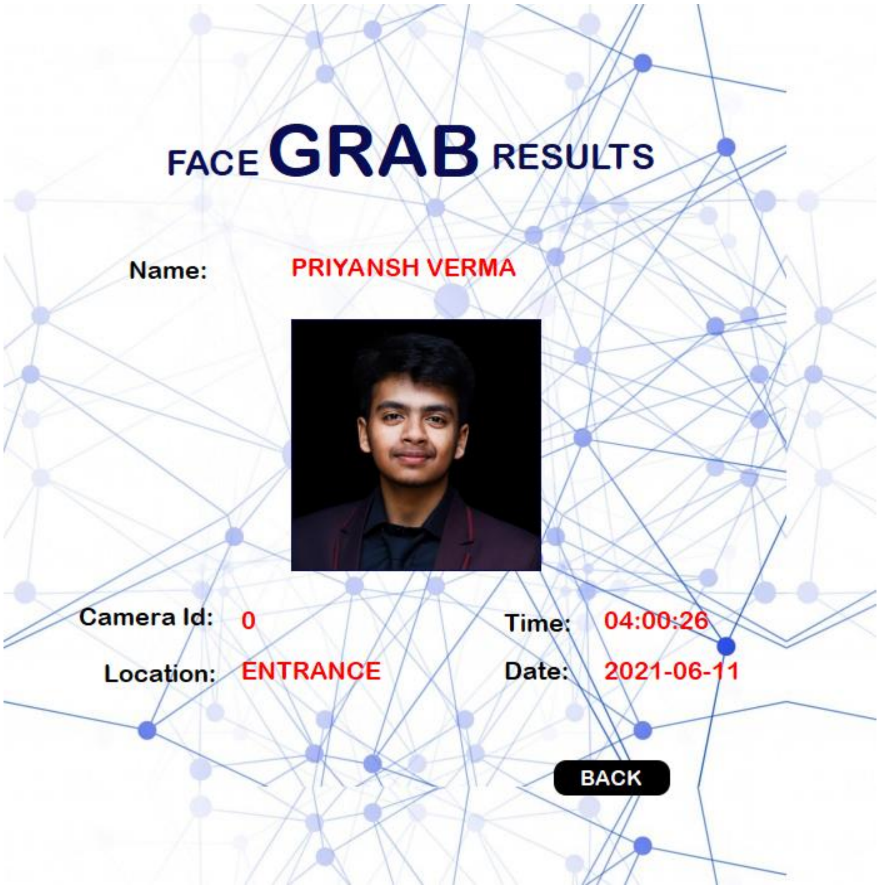
SORRY !!

Person Not Found.

If the entered reference id in the search page doesn't match the data in the database, not found window will pop-up

4.RESULT PAGE

Result page displays the details of the person’s ID entered with the photograph. Exit button on this page will bring you back to the search page.



FACE GRAB RESULTS

Name: PRIYANSH VERMA

Camera Id: 0

Time: 04:00:26

Location: ENTRANCE

Date: 2021-06-11

Inside the generated python code, various style sheets are included for the styling.

Below are the modules and classes that are required while designing the frontend.

MODULE and CLASSES	DESCRIPTION
Pyqt5	Core non-graphical classes used by other modules.
QtCore	Base classes for graphical user interface (GUI) components. Includes OpenGL.
QtGui	Classes to extend Qt GUI with C++ widgets.
QtWidgets	Qt Widgets Module provides a set of UI elements to create classic desktop-style user interfaces.
QRegExp	QRegExp is a class of QtCore module which provides pattern matching using regular expressions.
QRegExpValidator	QRegExpValidator is a class inside QtGui module which is used to check a string against a regular expression
QPixmap	QPixmap class inside QtGui module which is an off-screen image representation that can be used as a paint device.

9. BACKEND

"Back-end" means the parts that do the work, but the user is unaware of or cannot see.

FACE GRAB's backend is written in python (version **3.9.5**) programming language on **PyCharm IDE 2021.1** (Community Edition).

At first, we installed some packages either on PyCharm or on the command line using **pip** command which is a package installer for Python.

Packages names and version and description is described below:

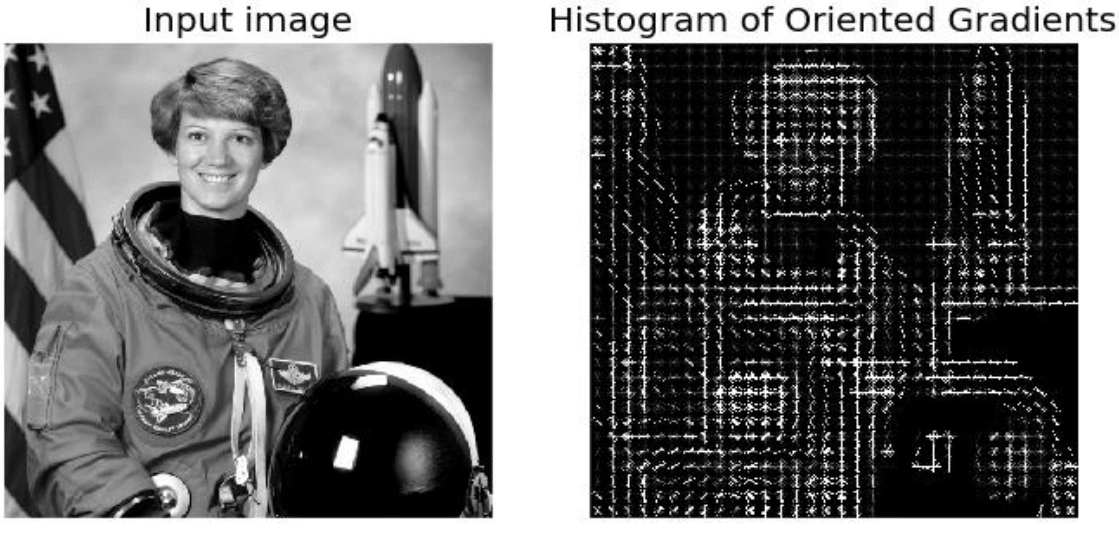
VERSION	PACKAGES	DESCRIPTION
6.1.0	pillow	Python Imaging Library (PIL) adds support for opening, manipulating, and saving many different image file formats
19.1.1	pip	package-management system written in Python used to install and manage software package
3.20.3	cmake	cross-platform build system generator that make/project files
19.22.0	dlib	Digital Library - C++ library implementing a variety of machine learning algorithms
1.16.4	numpy	Numerical Python - fundamental package for scientific computing

4.5.2.52	opencv-python	open computer vision - library of Python bindings designed to solve computer vision problems
1.3.0 0.3.0	face-recognition face-recognition-models	Recognize and manipulate faces, Built using dlib’s state-of-the-art face recognition built with deep learning.
5.15.4	pyqt5	Python Quasar Toolkit - used building GUI apps in Python

Basic **Face Recognition** through machine learning involves four basics steps:

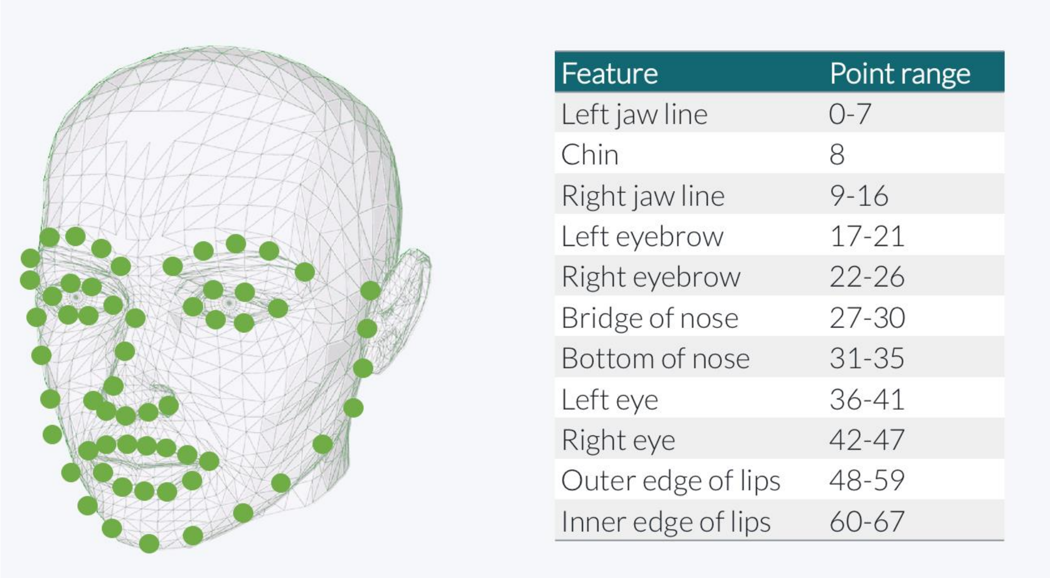
Step 1: *Finding all the Faces*

By using the method called Histogram of Oriented Gradients - or just HOG for short.
The original image is then turned into a HOG representation that captures the major features of the image regardless of image brightness and color.



Step 2: *Posing and Projecting Faces*

Use an algorithm called face landmark estimation.
The basic idea is we will come up with 68 specific points (called landmarks) that exist on every face — the top of the chin, the outside edge of each eye, the inner edge of each eyebrow, etc.



Feature	Point range
Left jaw line	0-7
Chin	8
Right jaw line	9-16
Left eyebrow	17-21
Right eyebrow	22-26
Bridge of nose	27-30
Bottom of nose	31-35
Left eye	36-41
Right eye	42-47
Outer edge of lips	48-59
Inner edge of lips	60-67

Step 3: *Encoding Faces*

Pass the centered face image through a neural network to train the model to generate 128 measurements for each face.

```
[-0.16597123  0.11949642  0.10122138 -0.12254387 -0.11838274 -0.02759848
  0.01384261 -0.07288168  0.06527785 -0.13963827  0.17559825 -0.08519795
 -0.25020906 -0.0102928  -0.01079375  0.21948609 -0.21553843 -0.19072869
 -0.13214052 -0.10856642  0.01477439  0.07908501 -0.05431245  0.11583221
 -0.1715932  -0.23868942 -0.10927209 -0.01475592  0.01510147 -0.08837818
 -0.00252565  0.03029939 -0.17696191  0.00717495  0.04937807  0.12862331
 -0.01091681 -0.09640069  0.1934087   0.01389313 -0.23863317  0.01766105
  0.15954995  0.26412117  0.21739453 -0.00336392 -0.03702538 -0.05539587
  0.10163558 -0.28075284  0.05129996  0.20316276  0.02803321  0.16820049
 -0.02042791 -0.12237336 -0.00515839  0.22946236 -0.21929839  0.03012473
  0.10334508 -0.19385433  0.010464   -0.1046315   0.159848    0.14616895
 -0.19181257 -0.22495562  0.19514428 -0.23470432 -0.11755507  0.07461189
 -0.0839375  -0.09592333 -0.35081255  0.03915626  0.32692659  0.20959754
 -0.10889891 -0.01162777 -0.09144997 -0.02747694  0.00839157  0.15324235
 -0.02081387 -0.0425766  -0.06370837  0.0559925   0.19876704  0.04837478
  0.00898859  0.25267416  0.04914283  0.0009048   0.00353496  0.14037418
 -0.1516469  -0.07702    -0.10442869  0.03603458 -0.10455432 -0.05006842
  0.01399595  0.18746354 -0.10759938  0.24044091 -0.03882578 -0.01190661
 -0.12068785  0.03638472 -0.02094681 -0.02857181  0.10984001 -0.22919109
  0.14735889  0.20415246 -0.00772358  0.09734803  0.10852756  0.12523061
 -0.03349054 -0.08732443 -0.17586364 -0.09488015  0.04729999 -0.0654182
  0.10575557  0.07663471]
```

Machine learning people call the 128 measurements of each face an embedding

Step 4: *Finding the person's name from the encoding*

Find the person in our database of known people who has the closest measurements to our test image.

What the main python script is doing can be concluded as follows:

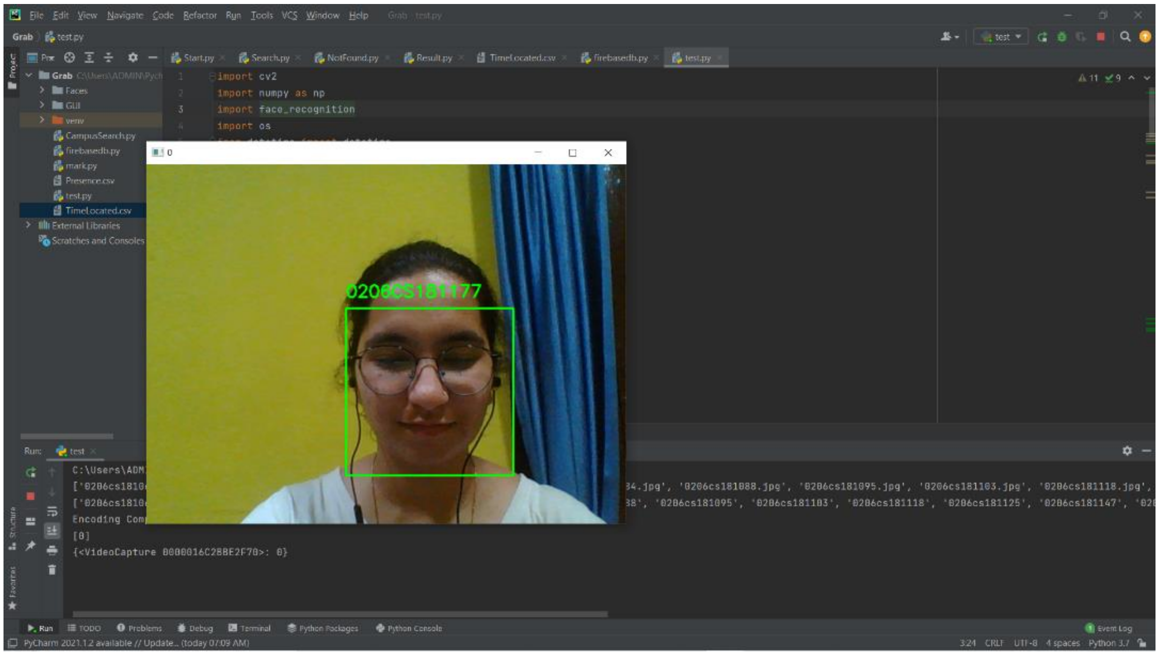
1. Reading the camera live stream from a number of cameras attached to the system.

2. Converting the image from the stream to RGB format.

3. Get all the faces from the camera live stream from all the cameras connected to the system

4. Encoding the faces in the stream and storing all 128 measurements.

5. Comparing the encodings from the live stream cameras with the encodings of known faces from the ImageSourceDirectory that is downloaded from the Firebase storage using Pyrebase package.



```
import cv2
import numpy as np
import face_recognition
import os
```

```
C:\Users\ADM
['0206cs1810                                    34.jpg', '0206cs181088.jpg', '0206cs181095.jpg', '0206cs181103.jpg', '0206cs181118.jpg',
['0206cs1810                                    38', '0206cs181095', '0206cs181103', '0206cs181118', '0206cs181125', '0206cs181147', '020
Encoding Com
[0]
{<VideoCapture 0000016C28BE2F70>: 0}
```

6. Comparing the facial features distance with faces from all the live streams cameras with the faces already encoded from the ImageSourceDirectory facial features distance.

7. If the faces match, the timelocated () function is called with the id of the face and the stream in which the face is found as arguments.

10. DATABASE

1. The FACEGRAB web app uses google Firebase platform for Database services. **Firebase** developed by Google is one of the leading BaaS (Backend-as-a-Service) solutions on the market.

2. Firebase provides two database services, Firestore and Realtime database.

a) **Firestore** is a NoSQL document database built for automatic scaling application development. It describes relationships between data objects.

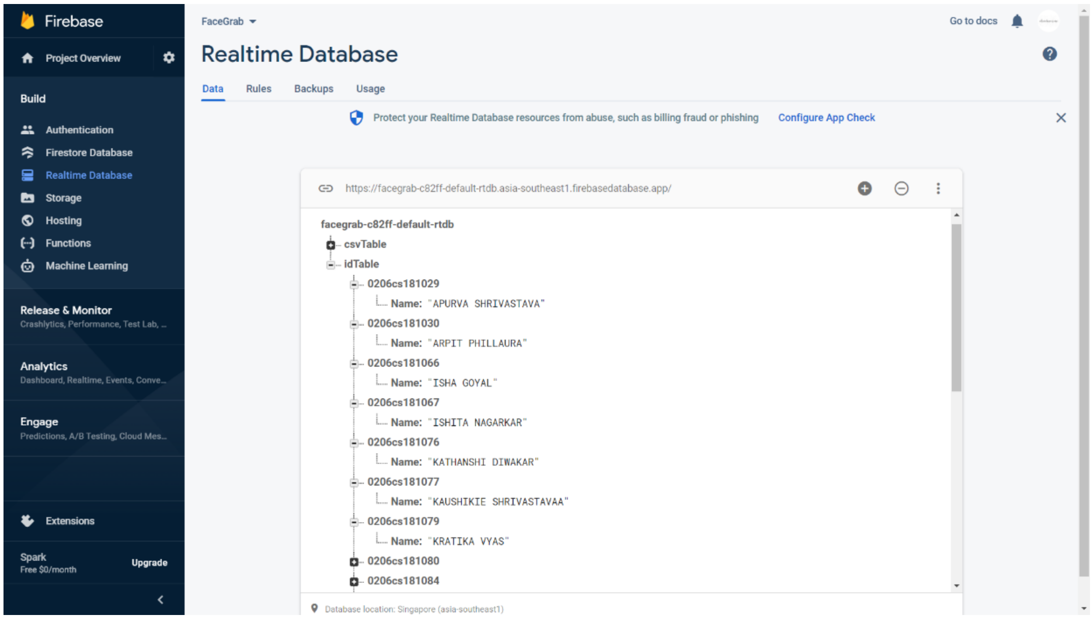
b) Firebase **Realtime** Database: The Firebase Realtime Database is a cloud-hosted NoSQL database that lets you store and sync data between your users in real-time.

FACEGRAB uses the **Realtime** database.

3. FACEGRAB Realtime database includes 2 tables:

a) **csvTable**: this table stores the person's details i.e., name, date, location, camera id, and time.

b) **idTable**: This table stores the **Unique** reference ID and names associated with it.



Protect your Realtime Database resources from abuse, such as billing fraud or phishing

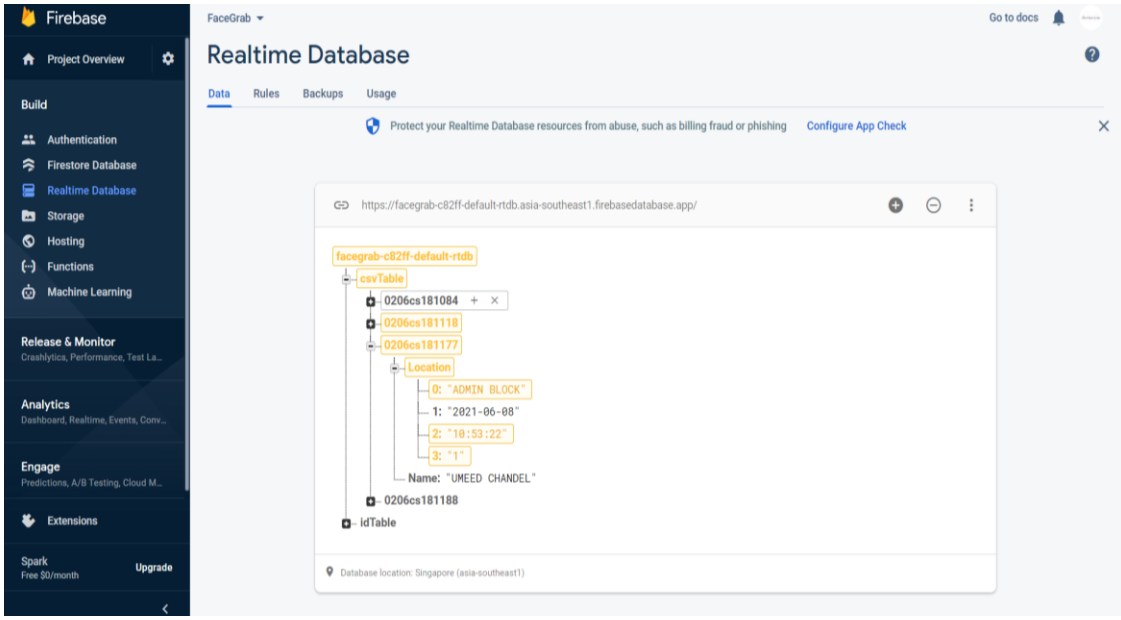
https://facegrab-c82ff-default-rtdb.asia-southeast1.firebasedatabase.app/

```
facegrab-c82ff-default-rtdb
  csvTable
  idTable
    0206cs181029
      Name: "APURVA SHRIVASTAVA"
    0206cs181030
      Name: "ARPIT PHILLAURA"
    0206cs181066
      Name: "ISHA GOYAL"
    0206cs181067
      Name: "ISHITA NAGARKAR"
    0206cs181076
      Name: "KATHANSHI DIWAKAR"
    0206cs181077
      Name: "KAUSHIKIE SHRIVASTAVAA"
    0206cs181079
      Name: "KRATIKA VYAS"
    0206cs181080
    0206cs181084
```

Database location: Singapore (asia-southeast1)

4. **csvTable** stores the real-time data from cameras and **idTable** has names associated with reference ID to be searched.

5. When Admin uses face grab to search a person, the real-time cameras start sending the information to the csvtable and the details of the particular reference ID and name from the idTable is displayed on the Result Page.

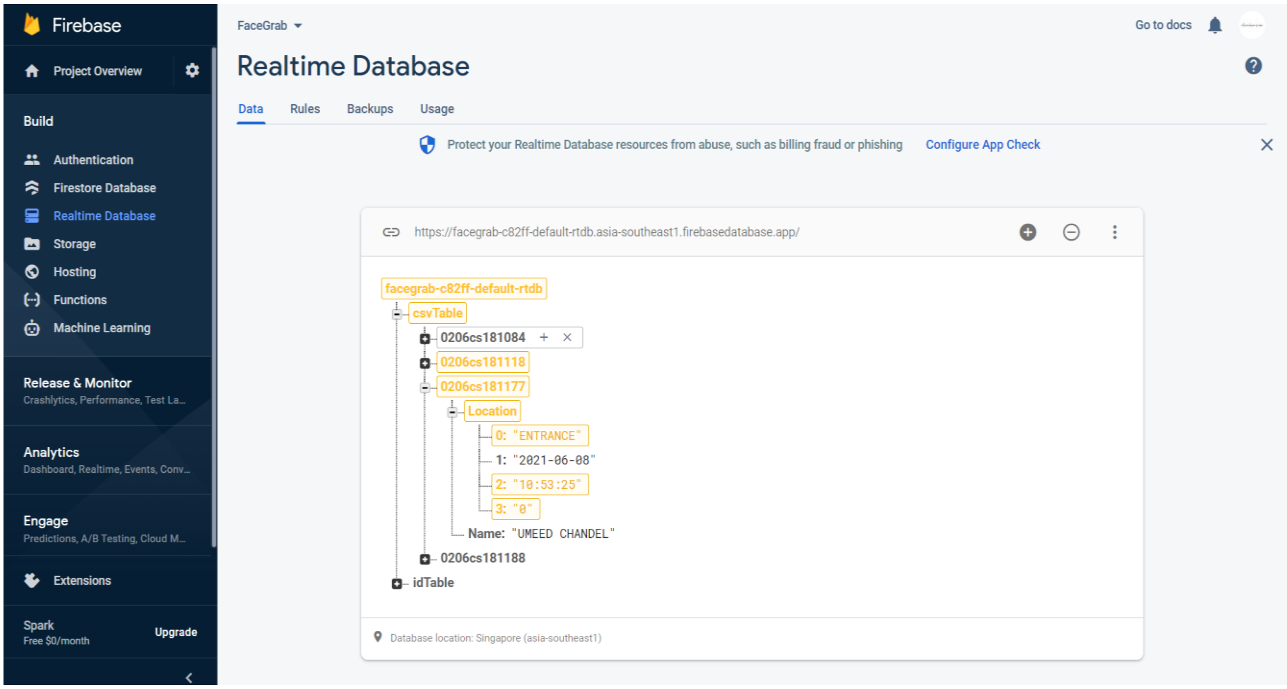


Protect your Realtime Database resources from abuse, such as billing fraud or phishing

https://facegrab-c82ff-default-rtdb.asia-southeast1.firebasedatabase.app/

```
facegrab-c82ff-default-rtdb
  csvTable
    0206cs181084  +  ×
    0206cs181118
    0206cs181177
      Location
        0: "ADMIN BLOCK"
        1: "2021-06-08"
        2: "10:53:22"
        3: "1"
      Name: "UMEED CHANDEL"
    0206cs181188
  idTable
```

Database location: Singapore (asia-southeast1)



Protect your Realtime Database resources from abuse, such as billing fraud or phishing

https://facegrab-c82ff-default-rtdb.asia-southeast1.firebasedatabase.app/

```
facegrab-c82ff-default-rtdb
  csvTable
    0206cs181084
    0206cs181118
    0206cs181177
      Location
        0: "ENTRANCE"
        1: "2021-06-08"
        2: "10:53:25"
        3: "0"
      Name: "UMEED CHANDEL"
    0206cs181188
  idTable
```

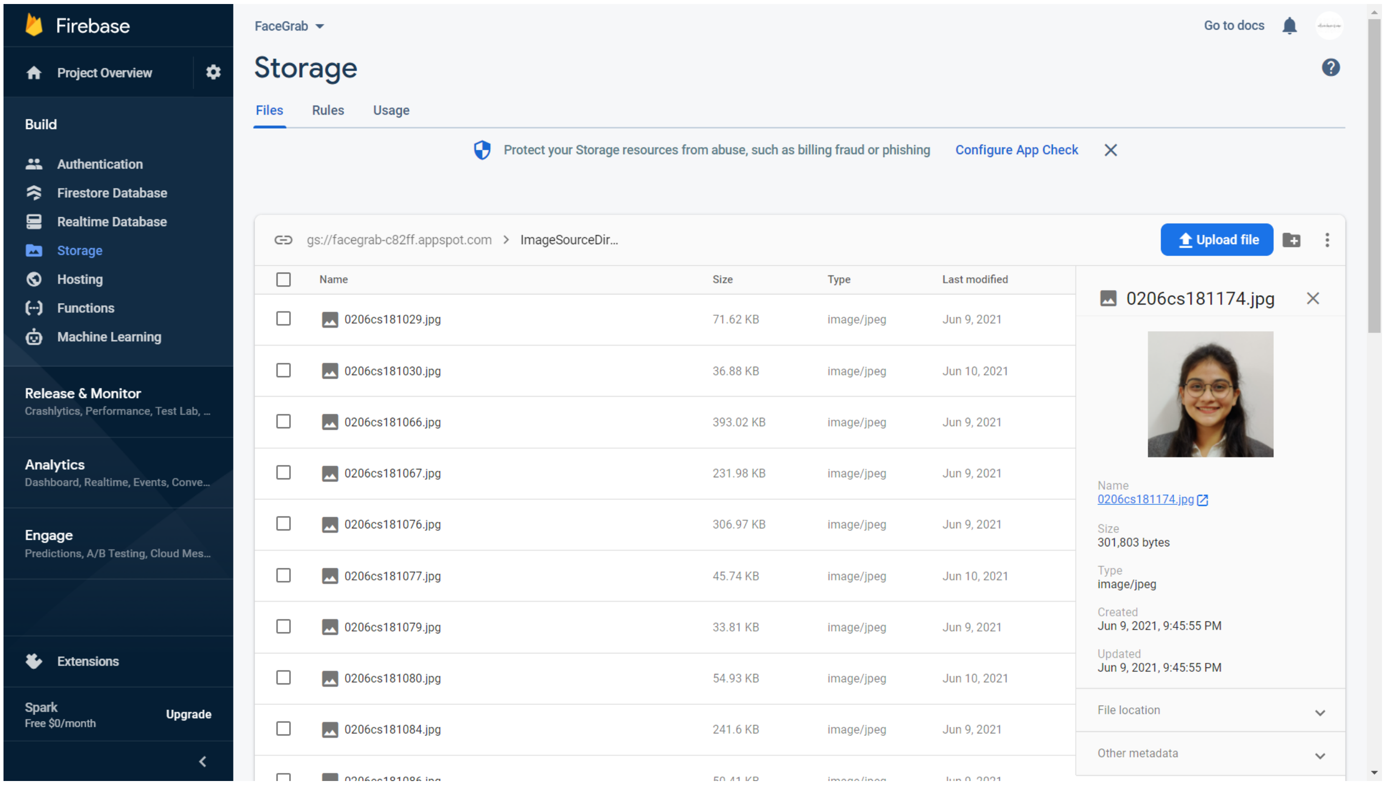
Database location: Singapore (asia-southeast1)

11. STORAGE

1. Cloud Storage on Firebase lets you upload and share user generated content, such as images and videos, which allows you to build rich media content into your applications.

2. Your data is stored in a Google cloud Storage bucket - an exabyte scale object storage solution with high availability and global redundancy.

3. Cloud Storage on Firebase lets you securely upload these files directly from mobile devices and web browsers, handling spotty networks with ease.



Protect your Storage resources from abuse, such as billing fraud or phishing

gs://facegrab-c82ff.appspot.com > ImageSourceDir…

□	Name	Size	Type	Last modified
□	0206cs181029.jpg	71.62 KB	image/jpeg	Jun 9, 2021
□	0206cs181030.jpg	36.88 KB	image/jpeg	Jun 10, 2021
□	0206cs181066.jpg	393.02 KB	image/jpeg	Jun 9, 2021
□	0206cs181067.jpg	231.98 KB	image/jpeg	Jun 9, 2021
□	0206cs181076.jpg	306.97 KB	image/jpeg	Jun 9, 2021
□	0206cs181077.jpg	45.74 KB	image/jpeg	Jun 10, 2021
□	0206cs181079.jpg	33.81 KB	image/jpeg	Jun 9, 2021
□	0206cs181080.jpg	54.93 KB	image/jpeg	Jun 10, 2021
□	0206cs181084.jpg	241.6 KB	image/jpeg	Jun 9, 2021
□	0206cs181086.jpg	50.41 KB	image/jpeg	Jun 9, 2021

0206cs181174.jpg

Name
0206cs181174.jpg

Size
301,803 bytes

Type
image/jpeg

Created
Jun 9, 2021, 9:45:55 PM

Updated
Jun 9, 2021, 9:45:55 PM

File location

Other metadata

We used the following packages to connect to firebase database and storage:

VERSION	PACKAGES	DESCRIPTION
3.0.27	pyrebase	Python interface to Firebase's REST API
4.3.3	pyrebase4	to use Python to manipulate your Firebase database

12. TEST CASES

A test case is usually a single step, or occasionally a sequence of steps, to test the correct behavior/functionality, features of an application.

1. As mentioned earlier FACE GRAB the webapp will be accessed by only one individual, the admin.

2. He's the 1st actor who'll be interacting with the GUI (Graphic User Interface) which has a total of 4 modules or pages.

    a. START PAGE - The START module has only one event occurrence i.e., to click the **Start** button and move on to the next module.

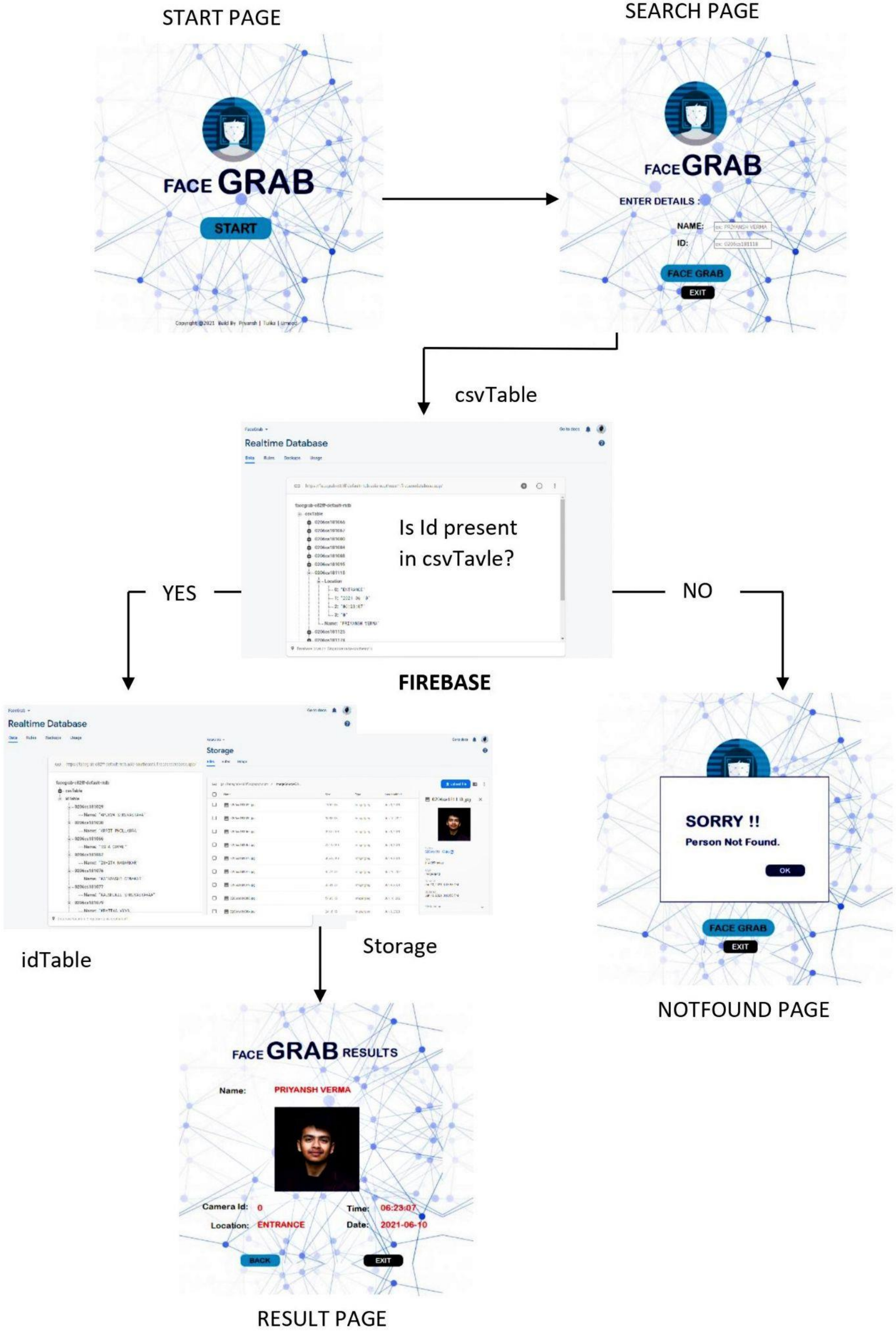
    b. SEARCH PAGE - The SEARCH module has two attributes **Name** and **Id** of the person to be found**,** to be entered by the admin.

    c. RESULT PAGE - The RESULT module will display the details of the individual if the individual is present in the storage and database of our firebase project and came in front of any surveillance camera installed on the premises. To move back it has a **back** button on it, which will bring you back to the SEARCH module.

        The details shown on the RESULT module are:

        a) Name
        b) Photo
        c) Camera id
        d) Time
        e) Location
        f) Date

    d. NOT FOUND PAGE - If the conditions haven't met then simply the NOT FOUND module will appear. It will have an **ok** button to get rid of the dialog box.



START PAGE

FACE GRAB

START

Copyright @2021 Build By Priyansh | Tulika | Umeed

SEARCH PAGE

FACE GRAB

ENTER DETAILS :

NAME: ex: PRIYANSH VERMA

ID: ex: 0206cs181118

FACE GRAB

EXIT

csvTable

Realtime Database

Data Rules Backups Usage

```
facegrab-c82ff-default-rtdb
  csvTable
    0206cs181066
    0206cs181067
    0206cs181080
    0206cs181084
    0206cs181088
    0206cs181095
    0206cs181118
      Location
        0: "ENTRANCE"
        1: "2021-06-10"
        2: "06:23:07"
        3: "0"
      Name: "PRIYANSH VERMA"
    0206cs181125
    0206cs181126
```

Is Id present in csvTavle?

FIREBASE

YES

NO

Realtime Database

Data Rules Backups Usage

```
facegrab-c82ff-default-rtdb
  csvTable
  idTable
```

Storage

idTable

Storage

RESULT PAGE

FACE GRAB RESULTS

Name: PRIYANSH VERMA

Camera Id: 0

Time: 06:23:07

Location: ENTRANCE

Date: 2021-06-10

BACK

EXIT

NOTFOUND PAGE

SORRY !!

Person Not Found.

OK

FACE GRAB

EXIT

3. Meanwhile the installed surveillance cameras on the premises are working and gathering the details of the individuals that are present in the project firebase and storing it in the Firebase Realtime Database. Child name “**csvTable”.**

4. The pictures in the firebase **“storage”** are stored by their unique Id name. And the names to the id’s are allotted in the **“idTable”** firebase database.

5. The entries in **“csvTable”** are the deciding factor whether RESULT module will be displayed or NOT FOUND module as it’s the table which stores the individuals details when they appear in front of the cameras.

13. CONCLUSION

Hereby we conclude that the FACE GRAB, web application fulfills the basic functional requirements as stated in the report i.e., surveillance automation. It successfully displays the details of the person to be searched by using his reference id while reducing labor work, and also, time required.

At present FACE GRAB has accomplished the core functionalities of surveillance automation and is capable of expanding its services to further extent.

14. BIBLIOGRAPHY

Blogs, articles, documentations, stack overflow, YouTube and git.

- PyCharm: https://www.jetbrains.com/pycharm/learn/
- OpenCV: https://docs.opencv.org/4.5.2/d6/d00/tutorial_py_root.html
- Face Recognition: https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78
- QT Designer: https://doc.qt.io/qt-5/qtdesigner-manual.html
- Pyrebase: https://github.com/thisbejim/Pyrebase
- Firebase: https://firebase.google.com/docs?authuser=0