Faculty of Computers
& Artificial Intelligence

Benha University

" CS2021A12"

Finding lost children

A senior project submitted in partial fulfillment of the requirements for the degree of Bachelor of Computers and Artificial Intelligence.

Computer Science Departement,

Project Team

1- Ahmed Nasser Mahmoud Ahmed
2- Arwa Essam Saad Abd El-Gouad
3- Mustafa Ahmed Khader Nassar
4- Mahmoud Yasser Mahmoud Taha
5- Mohamed Atia Abd El-Ghany Ibrahim
6- Mohamed Mahmoud El-Sayed Mahmoud

Under Supervision of

Dr. Ahemd Taha

Benha, July 2021

ACKNOWLEDGMENT

Firstly, the success of our project happens when we believe in ourselves, believe in doing the right things to help other people, and the help of many people around us.

Secondly, we would like to thank our supervisor, Dr. Ahmed Taha for his encouragement, patience, guidance, and support when our project starts until we finished the project; thank you, Dr. Ahmed Taha for helping us and for always wishing us the best.

DECLARATION

We hereby certify that this material, which we now submit for assessment on the program of study leading to the award of Bachelor of Computers and Artificial Intelligence in *(insert title of degree for which registered)* is entirely our own work, that we have exercised reasonable care to ensure that the work is original, and does not to the best of our knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of our work.

Signed: _____

Signed: _____

Signed: _____

Signed: _____

Signed: _____

Signed: _____

Date: 13, July 2021.

ABSTRACT

These days we face a big problem that impacts negatively our society. Most children being missed or kidnapped from their families. The New York Times reported that every hour, four children go missing in the country with close to half of them remaining untraced. Last year, 49% of the 36,740 children who went missing could not be traced. The percentage of untraced children for previous years indicates a worsening trend. In 2011, 38% of the 90,654 children who went missing remained untraced; the corresponding percentages were 41.35 in 2012 and 50.91 in 2013.

Our project aims to try to find these children in a local area and send them back to their families. We target phones as it's easy to use them and anyone these days has a smart phone. We made our app works in a local area and we selected the malls as a begin to our project, cause we can control the area of the mall instead of a public street where we don't have a lot of cameras.

Our project simply will be a mobile application which will help you to know the place of your child. The app can be used by parents to report missing their child or by any vigilant citizen who can report any vulnerable child on the mall. The app will take the child image, then detect the child face from it, then compare images in the database with the child image, trying to find the missing child by matching images. Also, the app works with surveillance systems if any camera detected any missing child and identifies him by comparing his face with the missing child, it will send a notification to his parents with his location.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

No table of contents entries found.

LIST OF ACRONYMS/ABBREVIATIONS

ACRONYM Definition of Acronym

Chapter One

1 INTRODUCTION

1.1 INTRODUCTION:

Nowadays we are surrounded by technology in all aspects of our life, and there are many examples of devices that we can't do without it such as smart phones, cameras, cars, and other devices which is being developed over the time to provide the best value for human being, so as a part of our duty is to learn how to use these technologies to facilitate the daily tasks that we do, in addition to finding solutions to intractable problems that require a great deal of time and effort to solve.

One of these problems is losing your child in a public place such as malls or hypermarkets, so in our project we used a combination of these technologies to face this worrying problems and facilitate finding your children. This project is a connected system consists of mobile application and the cameras of the mall based on machine learning algorithms such as face recognition, deep learning and age & gender detection instead of the old traditional surveillance systems. This new system makes it easier to search for your lost child automatically through cameras without any human intervention. We will discuss system details in the next chapters.

1.2 PROBLEM DEFINITION:

Around 8 million child worldwide are reported missing, losing your child commonly occurs in wide places such as malls and hypermarkets. Many children are kidnapped in malls because of parents' inattention, or they might lose their way away from their parents. Regardless of the reasons for the loss, but it still a problem that there is no innovative solutions to prevents its occurrence.

Often when a child is lost (in the mall, for example), the security men search for him through traditional methods that may succeed sometimes and fail at other times. It takes a lot of time and the probability of finding the child in the end is the same as not finding him. The cameras of the mall are usually emptied and workers are deployed

around the place to search for the child, and it may be too late and the child has been kidnapped and taken outside the mall or it may still be inside the mall, but it has not been found yet and there is no kind of safety in these methods.

1.2.1 What Is The Importance of This Problem?

As many children around the world gets lost every day, and no one wishes to lose his child especially in a crowded place like malls. With the traditional way if you lost your child there is no way but going to the security men of the mall to ask for help, a process that take time and effort and the child might have been kidnapped or left the current place. Considering all of these reasons, it was our responsibility to develop a system that reduce the possibility of losing your child and facilitate finding him in case of loss

1.2.2 What Are The Current Solutions?

Before starting to think about this project we searched for all the methods and techniques that already exist and help solving the same problem. There was a solution that has already been applied using a wrist bracelet.

The idea is the parent and child both should wear a wrist bracelet which monitors the distance between them, and if the child exceeded the specified distance the parent's bracelet give a warning that your child is away from you and you should take care, but one of the disadvantages of this bracelet is that the child could be sent by his parents to do something and in this case the bracelet will consider him lost and will give a warning, so it could be a little annoying if it applied to many people in the same place.

1.2.3 How Will Our Solution Solve The Problem? What Is New?

Our project's idea is elimination of human intervention and to make the process of searching for the child automatically through the cameras of the mall. The cameras at the entrance record an image to the child who enters the mall with another expected information like gender, age, and entrance date.

If the parents lost their child then they should use the application and upload an image of their child with some information like age and gender and expected lost date. As soon as the image gets uploaded, cameras start to search automatically through children and analyze them to capture similar faces and in the range of the child's age. Cameras at the exit of the mall will prevent the child from going outside. All these steps will be done automatically and in a record time instead of the traditional way of searching

1.3 SCOPE:

Our project targets the field of surveillance security and a specific age group. Through this project we aim to participate in the development of this field to decrease the possibility of children loss and make the local and crowded places safe for them.

1.4 SUMMARY:

At the end of this chapter, the reader will be able to form a basic idea about what is the problem that our project aims to solve, what is the importance of it, its current solutions and what we will add to our project to make it better than the traditional or existing methods to solve the same problem.

The reader will also understand the importance of modern technologies and how to use them to solve a worrying problem like the mentioned one.

Chapter Two

2 ANALYSIS AND DESIGN

2.1 INTRODUCTION:

In the previous chapter we formed a good idea about the functionality of the project, in this chapter we will dive into the analysis of the project such that what are the requirements of the user that the system should provide efficiently. The system also have requirements which must be available to work at its best. Those requirements can be divided into two types:

- Functional requirements
- Non-functional requirements

Which should be classified to understand what is necessary for the system and what is less important. In this chapter we will focus on stack holders, they are the category of people to whom this system is directed, and what are the benefits that will accrue to them after using the application?

2.2 USER AND SYSTEM REQUIRMENTS

2.2.1 User Requirements

In order for the system to be integrated, it must provide all the needs of the user, whether necessary or unnecessary, in order to reach, in the end, a state of satisfaction with it. In our project we provided all those requirements for the user to be able to use the application easily:

1. Interface of the application should be simple and easy to use.
2. User must have full accessibility to app settings and creating account.
3. User must be able to change his profile picture.
4. The app should be fast and reliable.

5. The app should include the notes and instructions about how to upload the correct image.

6. Connectivity between the app and back-end services should be efficient.

7. Notifications that user will receive must clear and arrive on time.

8. Security men must be able to see all lists in desktop application.

2.2.2 System Requirements

Each system needs some requirements in order to work properly. System requirements are classified as either functional requirements or non-functional requirements. A functional requirement specifies something that a user needs to perform their work, in other hand non-functional requirements specify all the remaining requirements not covered by the functional requirements.

2.2.2.1 Functional Requirements

- User must own a smartphone and to be connected to network.

- User must download the app and create an account to be able to access app features.

- Owner of the system (for example: malls) must own computers with high capabilities to be able to manage the servers.

- The system should capture and store information (face features, age, gender, and entrance date) about each child that enters.

- The quality of pictures captured by cameras of the mall should be high and clear.

- Connection between the app and the server requires good internet connection to cover the whole place.

- Cameras at the exits should delete the user from the database and store leaving date.

2.2.2.2 Non-Functional Requirements

- The system should be available during the working hours through the day.
- Accessibility of the system should be easy for the targeted category of people (parents).
- There should be a well-equipped and prepared control room in order to achieve best monitoring results.
- System results should be accurate to gain confidence and achieve the best results.
- System should consider backing up and restoring the lost information in case of failure.
- Security men should be prepared to use the app and the desktop application which is designed for the mall.
- System should be efficient and produce higher than 90% accuracy output.

2.3 STACK HOLDERS

In case of our projects there is no customers and there is no product to be sold. The project aims to target parents and any one responsible of a child inside the place.

Our project targets parents who lose their children in malls, our main goal is to help them to find their children as fast as we can with the help of the mall surveillance system and our project which uses deep learning and machine learning to find the child.

2.4 SYSTEM DESIGN

2.4.1 Activity Diagram

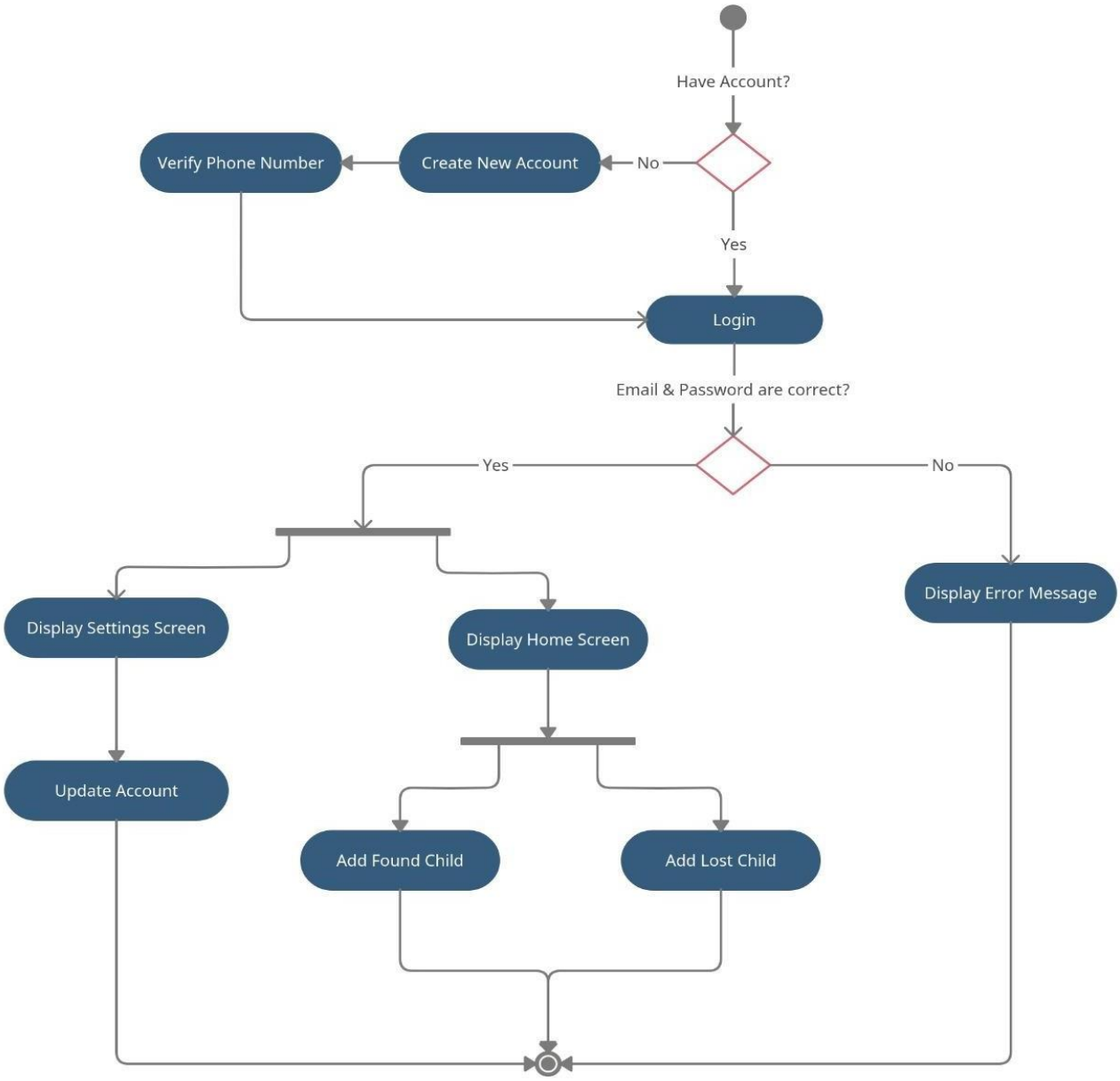


Figure 2.4.1.1: Activity Diagram

2.4.2 Use Case Diagram

Figure 2.4.2.1: Use Case Diagram

2.4.3 Sequence Diagram

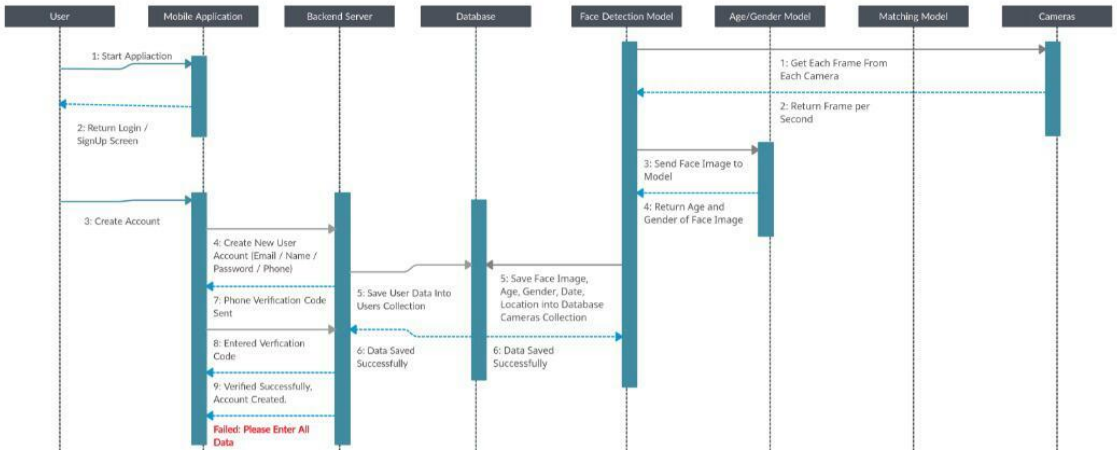


Figure 2.4.3.1: Sequence Diagram Part 1

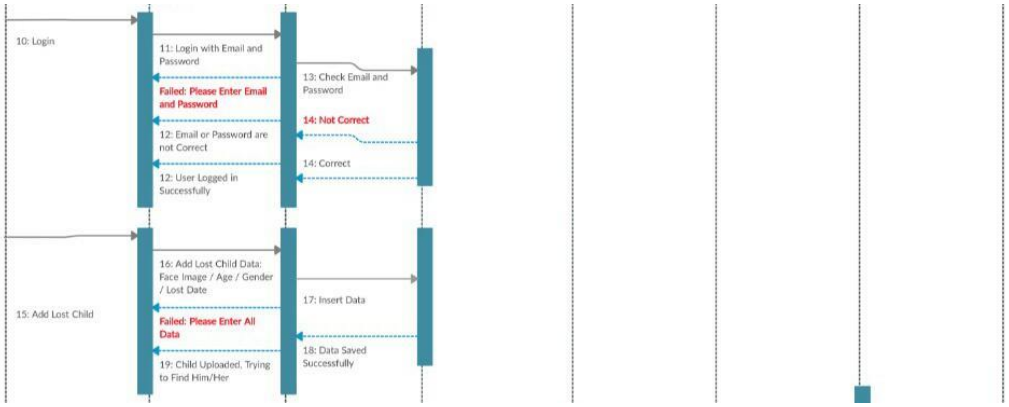


Figure 2.4.3.2: Sequence Diagram Part 2

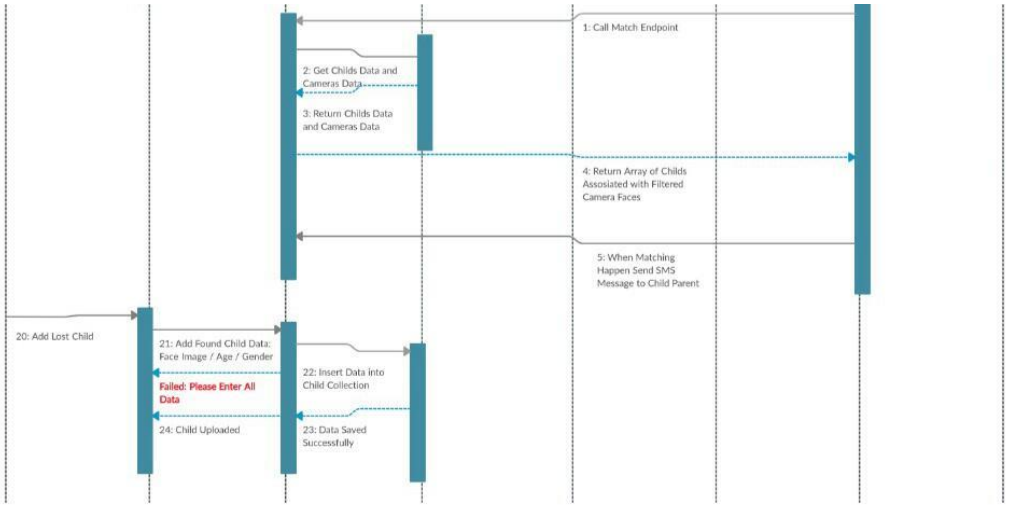


Figure 2.4.3.3: Sequence Diagram Part 3

2.4.4 Class Diagram

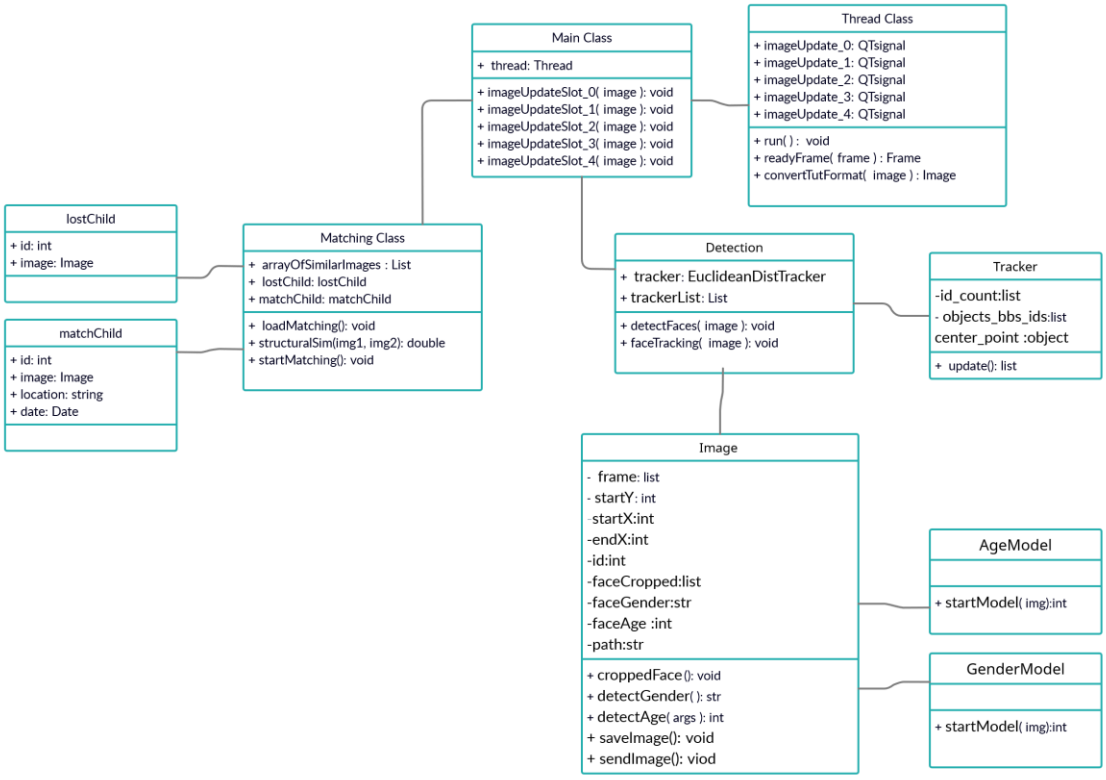


Figure 2.4.4.1: Class Diagram Part 1

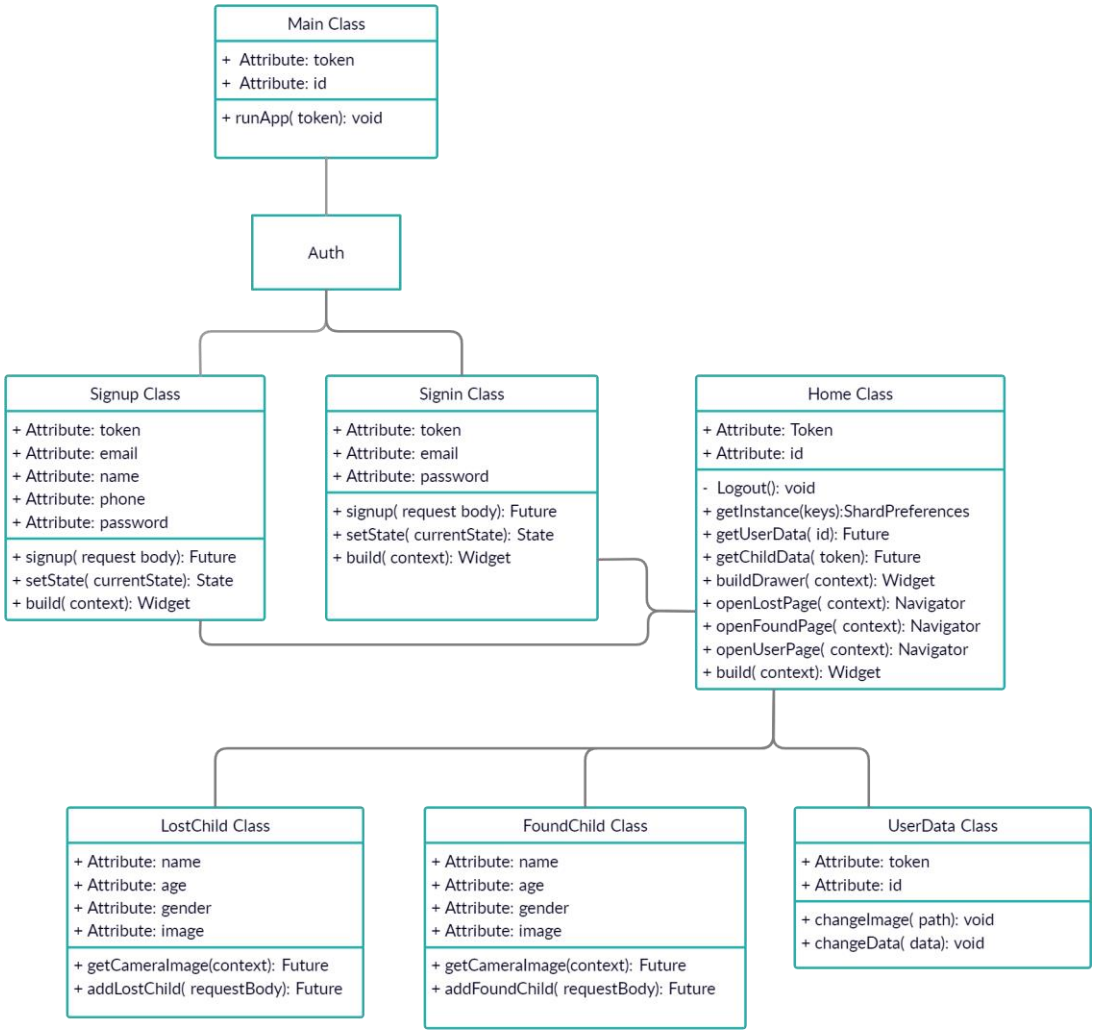


Figure 2.4.4.2: Class Diagram Part 2

2.4.5 Database Design

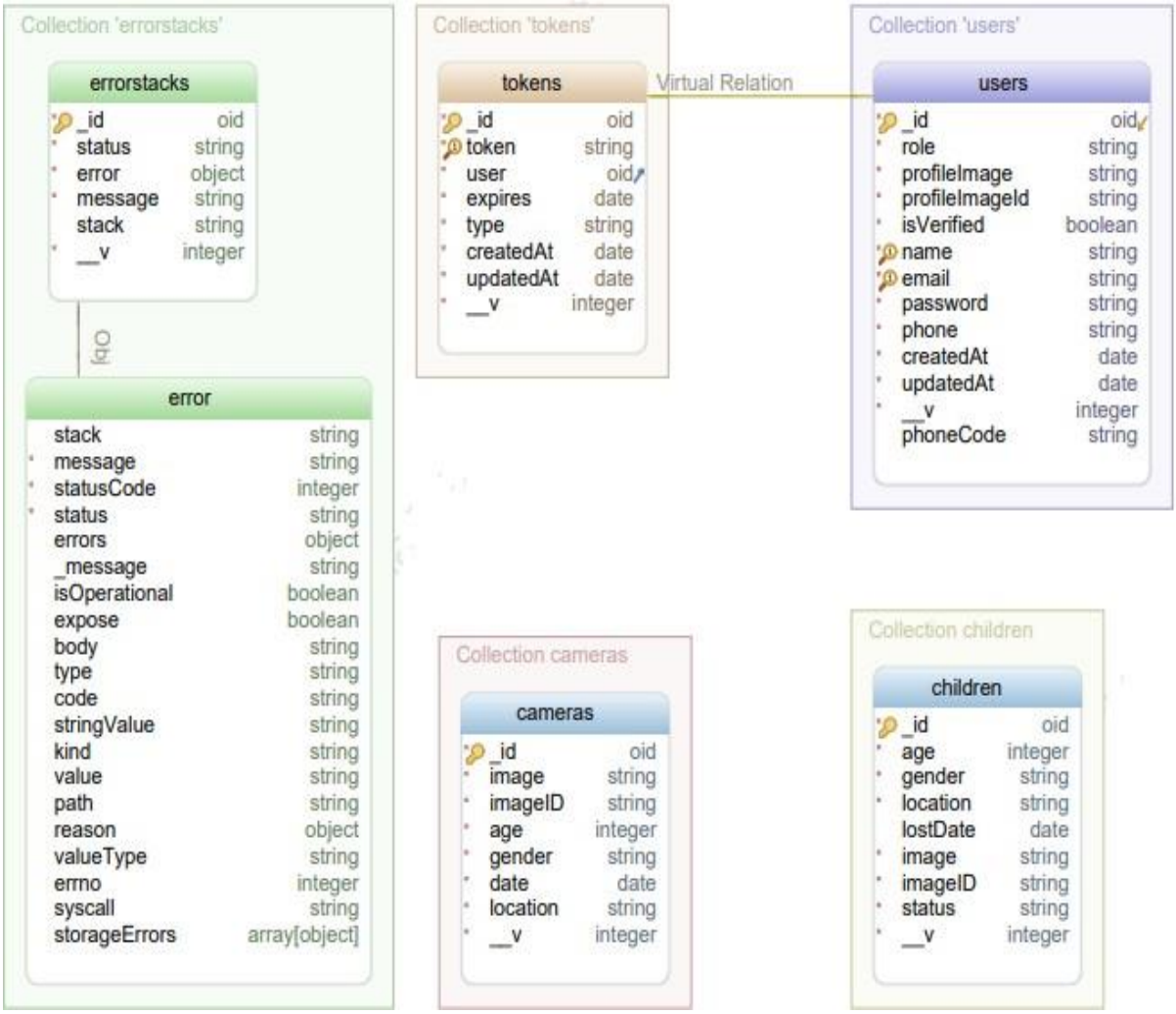


Figure 2.4.5.1: Database Diagram

2.5 USED TECHNOLOGIES AND TOOLS

2.5.1 Android Studio:

Is the official Android Application Development IDE based on IntelliJ IDEA, It supports development for all versions of Android, provided you download the respective SDKs from their servers and it's features:

- It is the official supported IDE by Google for Android Application Development. It provides code templates to build common app features.
- It is seamlessly integrated with the Google Cloud Platform, which allows you to easily set up Google Cloud Messaging for your application and includes the Google App Engine in your application.
- It has an advanced lint tool that helps you increase your app performance and helps you with usability, and version compatibility.
- It has a good-looking interface, which is the first and foremost thing I look for in any software or tool I use.
- It is available for Windows, Mac OS X and Linux as well, which means you can use your favourite Operating System while developing the applications on Android Studio.

The Android build system compiles app resources and source code and packages them into APKs that you can test, deploy, sign, and distribute. Android Studio uses an advanced build toolkit to automate and manage the build process while allowing you to define flexible custom build configurations. Each build configuration can define its own set of code and resources while reusing the parts common to all versions of your app.

The Android plugin for Gradle works with the build toolkit to provide processes and configurable settings that are specific to building and testing Android applications.

Gradle and the Android plugin run independently of Android Studio. This means that you can build your Android apps from within Android Studio, the command line on

your machine, or on machines where Android Studio is not installed (such as continuous integration servers).

If you are not using Android Studio. The output of the build is the same whether you are building a project from the command line, on a remote machine, or using Android Studio.

2.5.2 Language in Android:

- Dart – Dart is a client-optimized language for developing fast apps on any platform. Its goal is to offer the most productive programming language for multi-platform development, paired with a flexible execution runtime platform for app frameworks. We can use dart for building reliable and cross- platform mobile application through a toolkit designed by google called Flutter (Dart framework).

- Flutter – Flutter is a free and open-source mobile UI framework created by Google and released in May 2017. In a few words, it allows you to create a native mobile application with only one codebase. This means that you can use one programming language and one codebase to create two different apps (for IOS and Android). We used flutter for building mafkoud application to be available for all types of phone (Android or IOS).

Flutter consists of two important parts:

- An SDK (Software Development Kit): A collection of tools that are going to help you develop your applications. This includes tools to compile your code into native machine code (code for IOS and Android).

- A Framework (UI Library based on widgets): A collection of reusable UI elements (buttons, text inputs, sliders, and so on) that you can personalize for your own needs.

Advantage of the android studio:

1. **Faster Deployment of Fresh Builds:**

   Bringing incremental changes to an existing app code or resource is now easier and faster. Thanks to Instant Run. Code changes can be witnessed in the emulator or physical device in real-time without restarting the app or building a new APK (Android Application Package file) every time.

2. **More Accurate Programming:**

   Featuring an intelligent Code Editor equipped with the IntelliJ IDEA interface, Android Studio makes code writing and analysis faster, easier, and more accurate. The most challenging area has become a cakewalk now.

3. **Faster Programming and Testing:**

   The newly introduced emulator is 3x faster in CPU, RAM, & I/O in comparison to its predecessor. The virtual testing environment is faster than a real device and has a user-friendly UI.

   Sensor controls are effective in reading every move of the developers. Developers can drag and drop APKs for quick installation, resize and rescale the window, use multi-touch actions (pinch & zoom, pan, rotate, tilt), and much more.

4. **Inclusive App Development:**

   Making multiple builds is past now. Build for one and test on multiple devices using Cloud Test Lab Integration. Developers can check the compatibility and performance of an app on a wide range of physical Android devices from within Android Studio.

5. **Better App Indexing:**

   Promoting is an important component of app marketing, and Android Studio 2.0 takes it to a new high. The App Indexing feature available in the IDE helps in creating and adding indexable URL links to the app.

2.5.3 Node JS:

Everyone familiar with JavaScript, the most popular programming language, used as a client-side development tool in 95% of websites. But what about server- side programming?

Well, with the introduction of Node JS, JavaScript has become an all-purpose full-stack development language.

- Node JS is an open-source runtime environment for JavaScript. It's based on Chrome v8, an engine for chromium browsers.
- Node.JS allows your programs written in JavaScript to be executed on the server
- First written in 2009 to create dynamic web pages before they're sent to a browser.
- It soon became one of the most used tools in back-end web development.

There are many frameworks built for Node including such popular ones like Express JS, Meteor, Sails, and others, so, there are multiple reasons why Node JS became a standard for enterprise companies like Netflix, Uber and eBay.

Node.JS has opened the doors to JavaScript full stack development, inheriting the merits of JavaScript programming as well as allowing engineers to use its libraries and features. Lightweight JavaScript achieves high performance with fewer lines of code when compared to Java or C. Also, the frontend and backend are easier to keep in sync, because of a single language used on both sides of the application.

Developer-wise, it also became possible to share and reuse code. With the help of node modules, which are basically independent chunks of code, developers can use pre-built modules or reuse their own. Node.JS is highly scalable and lightweight. That's why it's a heavy favourite for micro service architectures.

Node.JS is considered fast thanks mostly to Chrome's v8 engine it’s used to compile JavaScript into machine code instead of using an interpreter.

JavaScript community notes constant improvements in the engine, as Google continues to invest heavily in it. While its event-based nature makes Node.JS highly

efficient for real-time apps that require constant data updates, the non-blocking input-output model solves performance issues.

Node JS Pros:

1. Node.js doesn't provide scalability. One CPU is not going to be enough; the platform provides no ability to scale out to take advantage of the multiple cores commonly present in today's server-class hardware.
2. Dealing with relational database is a pain if you are using Node.
3. Every time using a callback end up with tons of nested callbacks.
4. Without diving in depth of JavaScript if someone starts Node, he may face conceptual problem.
5. Node.js is not suited for CPU-intensive tasks. It is suited for I/O stuff only.

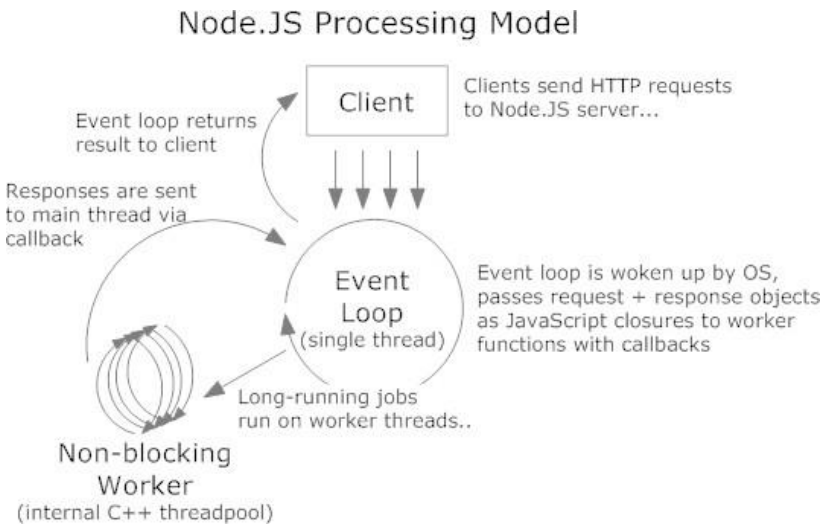


Figure 2.5.3.1: how Node JS works

2.5.4 Express JS (Node JS Framework):

Express.JS is an open-source Node.JS framework, it's used for designing and building web applications easily. Web applications are web apps that you can run on a web browser. Since Express.js only requires JavaScript, it becomes easier for programmers and developers to build web applications and API without any effort.

Express.js is a framework of Node.js which means that most of the code is already written for programmers to work with. You can build a single page, multi-page, or hybrid web applications using Express.js. Express.js is lightweight and helps to organize web applications on the server-side into a more organized MVC architecture.

The JavaScript library of Express.js helps the programmers to build efficient and fast web apps. Express.js enhances the functionality of the node.js. In fact, if you don't use Express.js, then you have to do a lot of complex programming to build an efficient.

API. It has made programming in node.js effortless and has given many additional features.

With the help of Express.js, you can easily build different kinds of web applications in a short period of time. Express.js provides a simple routing for requests made by clients. It also provides a middleware that is responsible for making decisions to give the correct responses for the requests made by the client.

Without Express.js, you have to write your own code to build a routing component which is a time consuming and tedious task. Express.js offers simplicity, flexibility, efficiency, minimalism, and scalability to the programmers. It also has the advantage of powerful performance as it is a framework of Node.js.

Node.js carries all the executions really fast with the help of Event Loop that avoids any kind of inefficiency. The powerful performance of Node.js and the ease of coding using Express.js are the most popular features loved by web application developers. Since Express.js is written in JavaScript, you can build websites, web applications, or even mobile apps using it.

Node.js is event-driven and thus has the ability to handle thousands of client requests at the same time which is not possible with PHP.

In today's world, real-time web apps and services are increasing in popularity. Node.js is designed exclusively to support real web applications. The most common example of a real-time web app would be live-chat. It involves thousands of users and real-time interaction that can be supported easily by Node.js.

Another asset of any business is money. It is important to use money efficiently to maximize profits. Since Express.js is an open-source and free web application that provides many great features, there is no reason left to not use it.

Features of Express JS:

1. Faster Server side development:

   Express.js provides many commonly used features of Node.js in the form of functions that can be readily used anywhere in the program. This removes the need to code for several hours and thus saves time.

2. Middleware:

   Middleware is a part of the program that has access to the database, client request, and the other middlewares. It is mainly responsible for the systematic organization of different functions of Express.js.

3. Routing:

   Express JS provides a highly advanced routing mechanism which helps to preserve the state of the webpage with the help of their URLs.

4. Templating:

   Express JS provides templating engines that allow the developers to build dynamic content on the web pages by building HTML templates on the server-side.

5. Debugging:

   Debugging is crucial for the successful development of web applications. Express JS makes debugging easier by providing a debugging mechanism that has the ability to pinpoint the exact part of the web application which has bugs.

2.5.5 MongoDB (NoSQL Database):

MongoDB is one of the most popular open-source NoSQL database written in C++. As of February 2015, MongoDB is the fourth most popular database management system. It was developed by a company 10gen which is now known as MongoDB Inc.

MongoDB is a document-oriented database which stores data in JSON-like documents with dynamic schema. It means you can store your records without worrying about the data structure such as the number of fields or types of fields to store values. MongoDB documents are similar to JSON objects.

MongoDB is a document-based database which is developed in the C++ programming languages. The word Mongo is basically derived from Humongous. MongoDB was first developed by a New York-based organization named 10gen in the year of 2007. Later 10gen changed the name and known as MongoDB Inc as of today. At the beginning, MongoDB is basically developed as a PAAS (Platform as a Service) database. But, in the year 2009, it was introduced as an open source database as named MongoDB 1.0. The below diagram demonstrates the release history of MongoDB to date. MongoDB 4.0 is the current stable version which is released in February, 2018.

In today's IT industry, there are large number of companies who are using MongoDB as a database service for the applications or data storage systems. As per the survey made by siftery on MongoDB, there are around 4000+ company confirmed that they are using MongoDB as Database. Some of the key names are:

1. Castlight Health
2. IBM
3. Citrix
4. Twitter
5. T-Mobile
6. Zendesk
7. Sony

8. BrightRoll
9. Foursquare
10. HTC
11. InVision

Since, MongoDB is a NoSQL database, so we need to understand when and why we need to use this type of database in the real-life applications. Since in normal circumstances, MongoDB always preferred by the developers or project managers when our main concern is the deal with large volume of data with a high performance. If we want to insert thousands of records in a second, then MongoDB is the best choice for that. Also, horizontal scaling (adding new columns) is not so easy process in any RDBMS systems. But in case of MongoDB, it is very much easy since it is a schema less database. Also, this type of work can be directly handled by the application automatically. There is no need to any type of administrative work for perform any type of horizontal scaling in the MongoDB. MongoDB is good for the below types of situations:

1. E-Commerce type of product-based applications.
2. Blog and Content Management systems.
3. High Speed logging, caching etc. in the Real time.
4. Need to maintain location wise Geospatial data.
5. For maintains data related to the Social and Networking types
6. If application is a loosely coupled mechanism – means design may change at any point of time

MongoDB Advantages:

1. MongoDB is a Schema less document type database.
2. MongoDB support field, range based query, regular expression or regex etc for searching the data from the stored data.
3. MongoDB is very easy to scale up or down.

4. MongoDB basically uses internal memory for storing the working temporary datasets for which it is much faster.
5. MongoDB support primary and secondary index on any fields.

6. MongoDB supports replication of database.
7. We can perform load balancing in the MongoDB by using Sharding. It scales the database horizontally by using Sharding.

8. MongoDB can be used as a file storage system which is known as a GridFS.
9. MongoDB provides the different ways to perform aggregation operations on the data like aggregation pipeline, map reduce or single objective aggregation commands.
10. MongoDB can store any type of file which can be any size without effecting our stack.
11. MongoDB basically use JavaScript objects in place of procedure.
12. MongoDB support special collection type like TTL (Time-To-Live) for data storage which expire at a certain time.

13. The dynamic database schema used in MongoDB is called the BSON.

2.5.6 Visual Studio Code

Visual Studio Code is a code editor in layman's terms. Visual Studio Code is "a free- editor that helps the programmer write code, helps in debugging and corrects the code using the intelli-sense method". In normal terms, it facilitates users to write the code in an easy manner. Many people say that it is half of an IDE and an editor, but the decision is up to the coders. Any program/software that we see or use work on the code that runs in the background. Traditionally coding used to do in the traditional editors or even in the basic editors like notepad! These editors used to provide basic support to the coders.

Visual Studio Code has some very unique features. They are listed as below:

1. **Support for multiple programming languages:**

   It supports multiple programming languages. So earlier, programmers needed Web-Support: a different editor for different languages, but it has built-in

multi-language support. This also means it easily detects if there's any fault or cross-language reference, it'll be able to detect it easily.

2. **Intelli-Sense:**

   It can detect if any snippet of code is left incomplete. Also, common variable syntaxes and variable declarations are made automatically. Ex: If a certain variable is being used in the program and the user has forgotten to declare, intelli- sense will declare it for the user.

3. **Cross-Platform Support:**

   Traditionally, editors used to support either windows, mac or Linux. But Visual Studio Code is cross-platform. So it can work on all three platforms. Also, the code works on all three platforms; else, the open-source and proprietary software codes used to be different.

4. **Extensions and Support:**

   Usually supports all the programming languages but, if the user/programmer wants to use the programming language which is not supported then, he can download the extension and use it. And performance-wise, the extension doesn't slow down the editor as it rums as a different process.

5. **Repository:**

   With the ever-increasing demand for the code, secure and timely storage is equally important. It is connected with git or can be connected with any other repository for pulling or saving the instances.

2.5.7 Anaconda

Anaconda is an open source distribution for Python. We used for machine learning. We almost use eight different libraries to develop our project. These libraries are OpenCV, Cvlib, Threading, Requests, Pyqt5 ui, Numpy, Tenserflow, and json.

We use OpenCV library with android and specially using cvlib to be able to make face detection and tracking and then take a picture for person face then compare this face

with other faces in database by using structural similarity algorithm that is used to make face recognition.

- Opencv (Open Source Computer Vision Library): is an open-source library that includes several hundreds of computer vision algorithms.

- Cvlib: A simple, high level, easy-to-use open source Computer Vision library for Python.

- Threading: C++ includes built-in support for threads, mutual exclusion, condition variables, and futures. Threads enable programs to execute across several processor cores.

- Requests: Is a Robot Framework library aimed to provide HTTP api testing functionalities by wrapping the well-known Python Requests Library. The quickest way to start is using the requests keywords and urls.

- PyQt5 ui o PyQt is a Python binding for Qt, which is a set of C++ libraries and development tools that include platform-independent abstractions for Graphical User Interfaces (GUI), as well as networking, threads, regular expressions, SQL databases, SVG, OpenGL, XML, and many

- Numpy: NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices.

- Tenserflow: TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.

- Json:

    - (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language Standard

- JSON is built on two structures:
    - A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
    - An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

2.5.8 PyCharm

Is the most widely used integrated development environment (IDE) for Python programming. This chapter provides an overview of PyCharm and describes its functionality.

PyCharm provides some features:

- Inspection and completion of the code
- Debugging on a higher level
- Web programming and frameworks such as Django and Flask are supported.

Chapter Three

3 DELIVERABLES AND EVALUATION

3.1 INTRODUCTION

On this chapter we will talk about real experiment with a real user. We tested our project in a real-world area and we asked random person to get in the experiment and he accepted that.

We will talk first about the user manual and what are the procedures the user should follow to find his/her child using our project. Then we will talk about our experiment and what did use made on a situation like this like losing his/her child. Finally we asked the user his/her opinions about the experiment and the project.

3.2 USER MANUAL

We will here talk about some procedures that user should follow when he/she lost his/her child in a mall.

1. When the user lose his/her child he/she should go to the security man and tell him the situation.

2. The security man will deal with this important situation by telling the user to download Mafkoud mobile application.

3. After the user download the application, he/she open the application and start creating new account.

4. The user should enter his/her email, name, password and phone number.

5. After that an SMS message will deliver to the user phone number with the verification code. The user should enter that code to verify his/her account.

6. Now the user after finishing the registration process he/she will see the home page and two buttons on it: Add Lost Child / Add Found Child.

7. The user should hit the Add Lost Child button.

8. Now he/she should upload his/her child image with age, gender, and lost date on this format (HH:MM PM/AM).

9. After submitting the child data the user should wait few minutes because our system now try to find the user child by searching for him on mall cameras and matching child face with camera faces.

10. When matching happens, the user will receive an SMS message on his phone number with his/her child location.

3.3 TESTING

We will test our project on real random user on mall and we will put him/her in a situation where he/she lose her/his child on the mall.

1. First thing user did he ran into the security room and told them what happened and how he/she lost her/his child.

2. The security man told the user to be patient and download the Mafkoud application which will help him to find his/her child as soon as possible and this app will reduce a lot of time on finding the child.

3. The user started downloading the application and created new account and verified the phone number.

4. The user opened the Add Lost Child screen on the application and filled the required data and submitted it.

5. The system started to find the child.

6. After almost 3 minutes the user received an SMS message on his phone number with the location of his/her child.

7. The user ran into the location and he/she found her/his child successfully.

3.4 EVALUATION (USER EXPERIMENT)

After testing our experiment on a real user we asked him/her about her/his opinions about our project, how he/she find/s it, and also what's his/her suggestions.

The user was glad with the project and how it helped him to find his/her child faster than the normal process which was reviewing all cameras videos and try to find the child which takes too much time.

The user said that he/she hope that this idea can be applied on the street cause it will help a lot of families in Egypt to find their children easily.

The user suggested to add interactive location finder, so he can see where the child now and how the system tracks the child.

3.5 SUMMARY

We saw a successful experiment with a real user and how he/she tested our project to find his/her child. We also saw user impression and how he liked the project too much.

This experiment target is to ensure that our project works perfectly on real-world situations on the mall and how the system deal with the problem and also how the security man should deal with it

Chapter four

4 DISCUSSION AND CONCLUSION

4.1 INTRODUCTION:

In this chapter we will discuss the importance of our project, how it will help reducing possibility of getting your child lost or kidnapped within a local or crowded area such as malls.

We will point at the implementation of each part of the project and each technology that has been used to achieve the targeted goal of the project, challenges that we faced during development of our idea which we have overcome. It will also include the possible feature recommendation that will be valuable and make the project applicable widely.at the end there will be a conclusion summary of all the project

4.2 MAIN FINDINGS:

4.2.1 Why is This Project Important?

The importance of our project lies in the fact that it solves a common and very worrying problem which is getting your child lost in crowded local places like malls. Many children gets lost every day from their parents and it may be by accident or it can be kidnapping incident. By the way, our project aims to make the best use of modern technologies to reduce the possibility of losing your child. And it does its best to keep your child closer to you even if you lost him.

So it is better than the traditional searching methods and participate in reducing the missing incidents.

4.2.2 Practical Implementations

4.2.2.1 Mafkoud Application

The application consists of:

- Authentication part Login, Register pages.
- Presentation for children data in the Home page.
- Add missing child data found or lost.
- User page.
- Notification page

Approach which is used in state management: setState, InheritedWidget. Used Packages:

- Shared preferences: used to save current authenticate by access token from the first login or register.
- Image picker: used to access device media from gallery app to upload image, or from device camera.
- http: for connect with REST API
- firebase core: used to initialize the project on firebase
- firebase_ml_vision: used machine learn toolkit from firebase vision package to detect child faces from coming media.

Application Flow:

1. **Landing Page**

   In That page we have function that check if user have login before, by check access token which saved in device shared preferences.

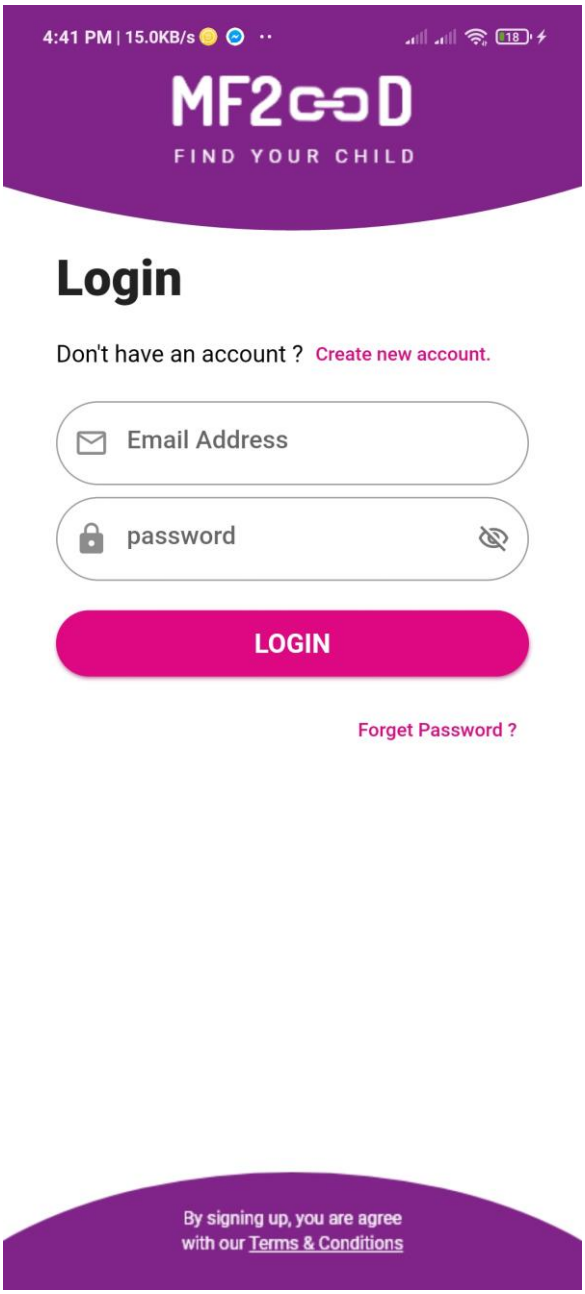
   If token not equal null so we already have user logged before then open Home Page.

Else if token equal null: the user need to login or register through login page or register page.

2. Login Page

User can login via email and password, we get data from custom input widgets by using TextEditingController to build request body.

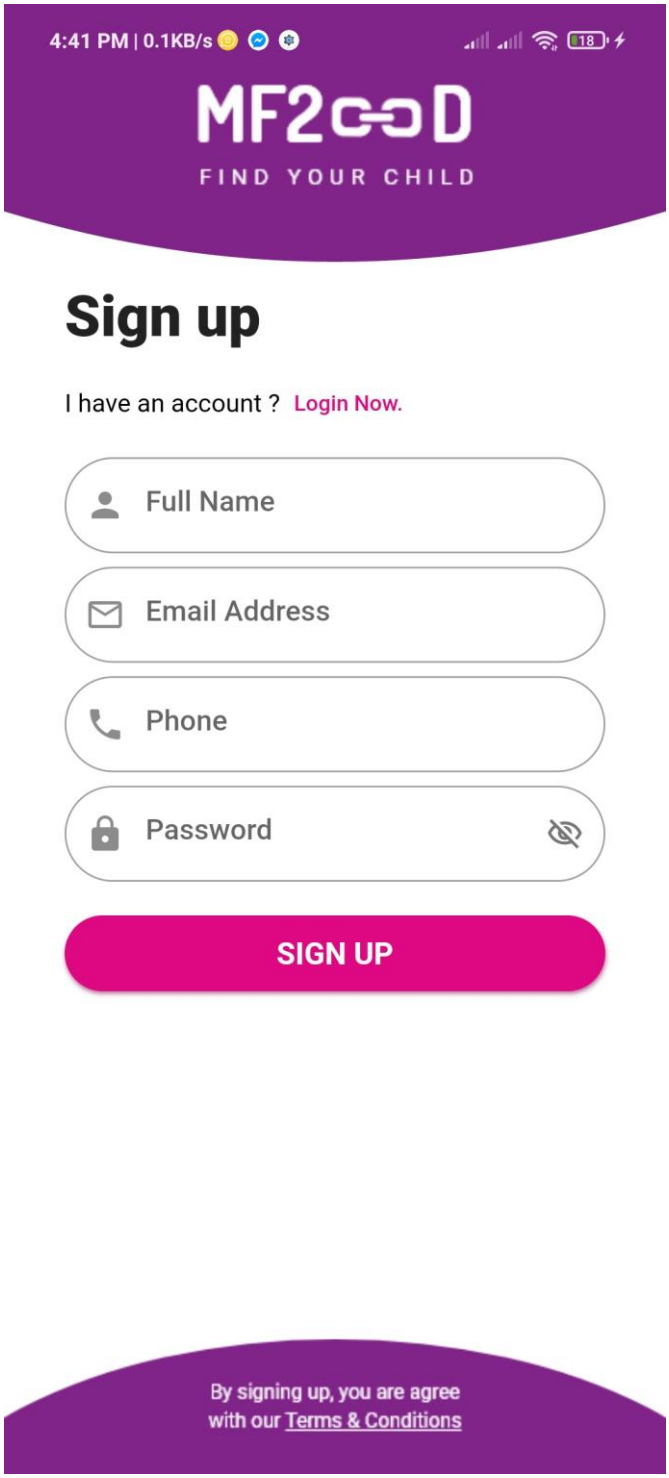
Then send post request, if the request return success status code: save the returned access token in shared preferences and open Home page.



3. Register Page

User can register via email by adding required data like email, name, phone number and password, we get data from custom input widgets by using Text Editing Controller to build request body.

Then send post request, if the request return success status code: save the returned access token in shared preferences and open Home page.



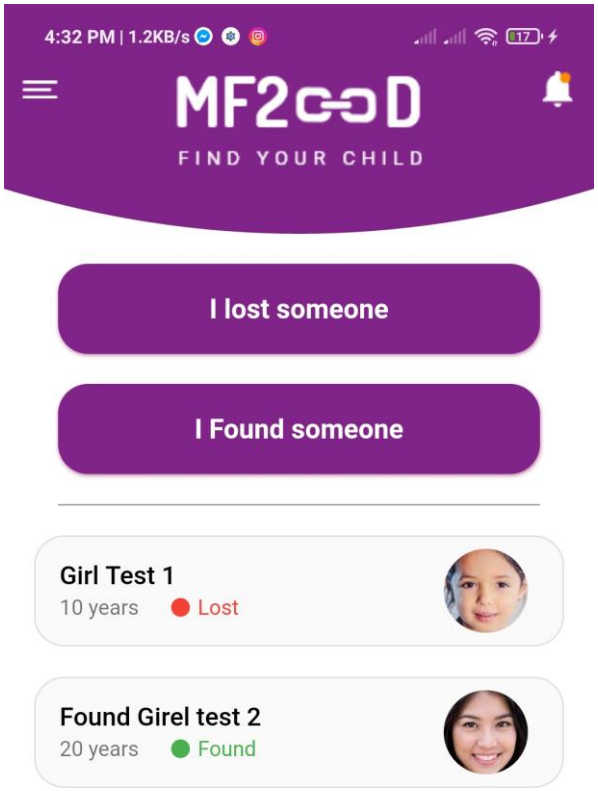
4. Home Page

In the home page we have two main buttons:

- Lost Child: used for open Lost page so can added required information about child.
- Found Child: used for open Found page.

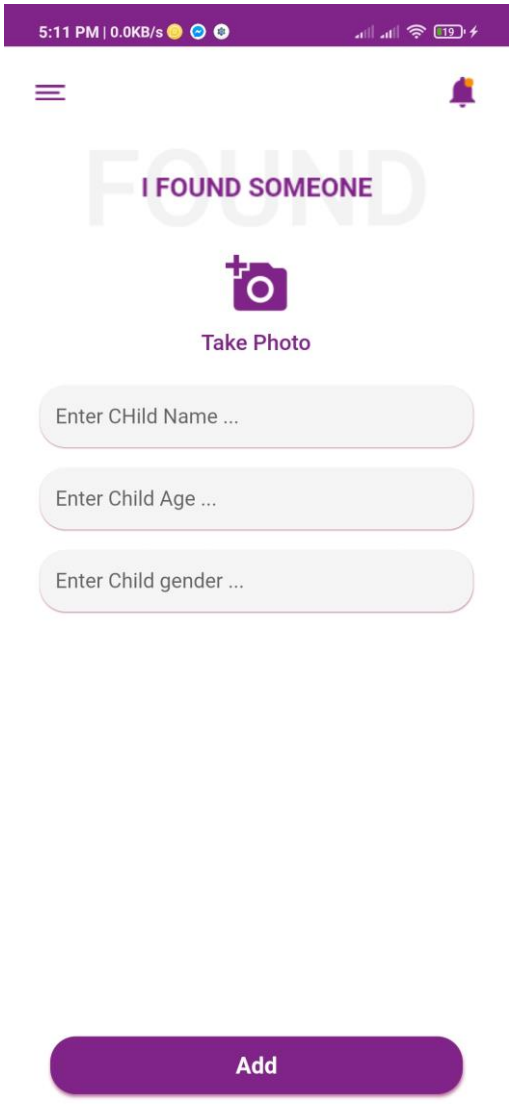
Second part is list of found and lost children every card have simple information about child.

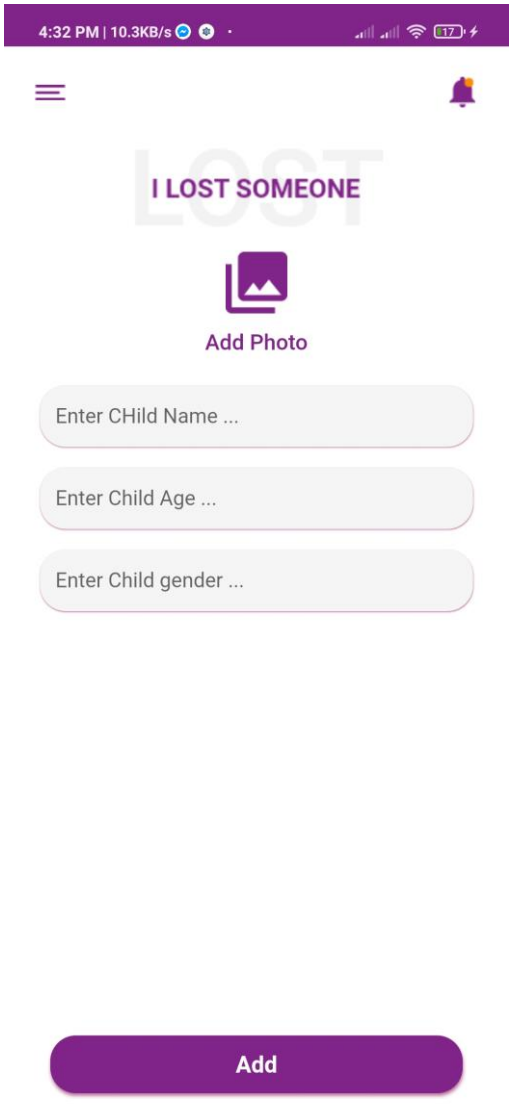
We get children data from post request and get the response list then map it to card widgets and push them to List view.



5. Add Lost/ Found Child

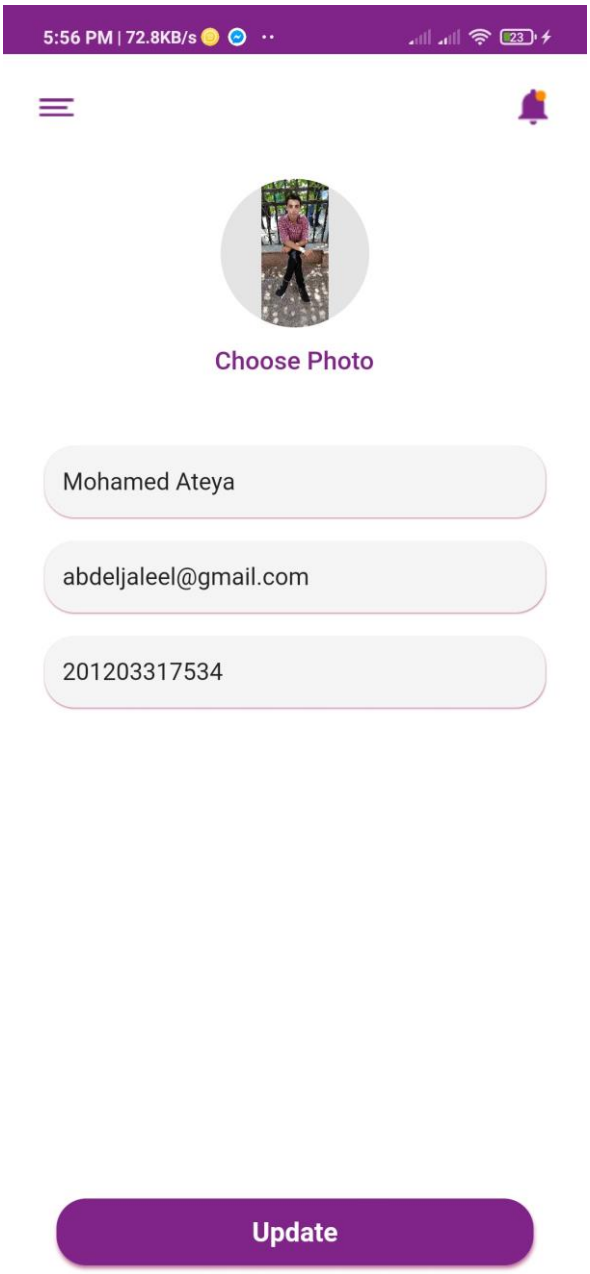
We get the required information like name, location, age, gender and the time child lost in. Then take the image which user uploaded to detect child face by using FaceDetector from FirebaseVisionImage package, then pass the face coordinates to other function to crop specified face from image. Then build post request using multi part request to upload the image.



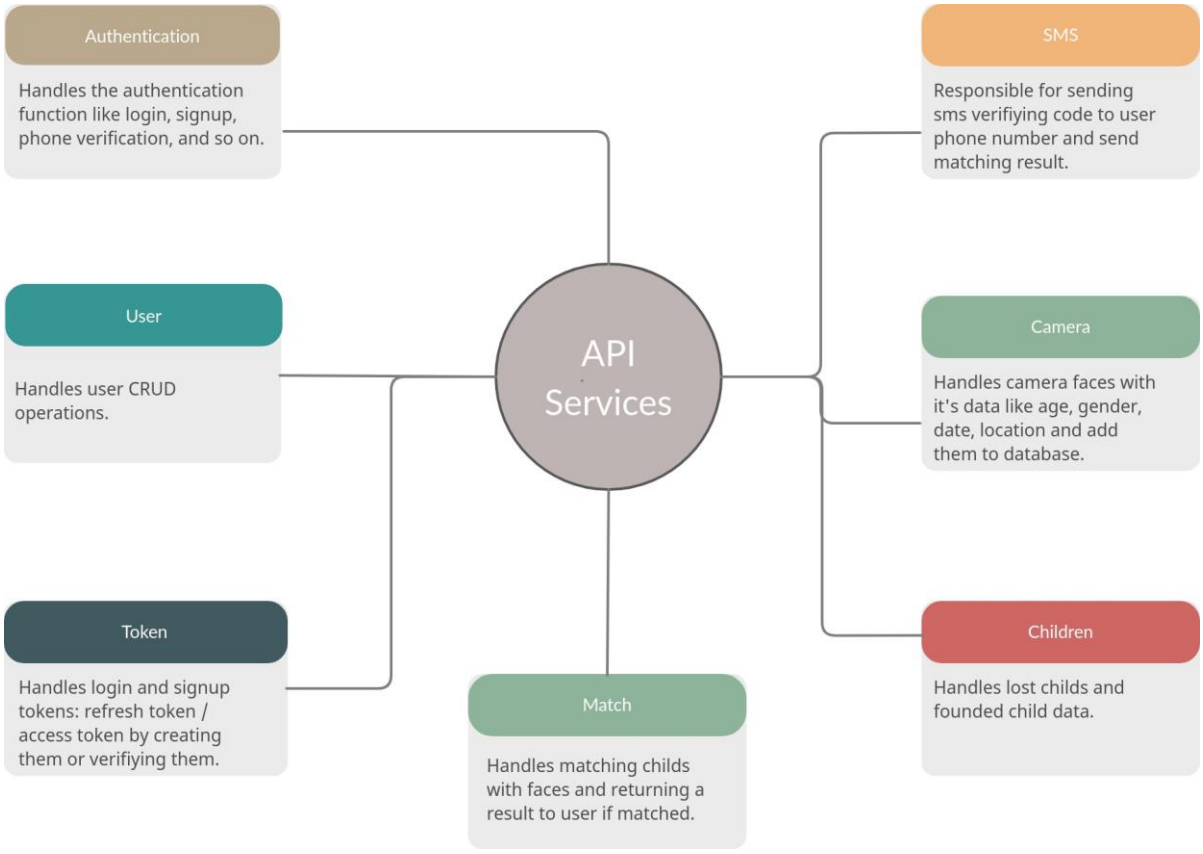


6. User Page

From that page user can edit name, phone and email.



4.2.2.2 Back-end Structure



The back-end API is divided into multiple services each service aims to achieve specific target in the process.

1. Authentication Services: These Services handles services like sign up and login for user and security man/woman, so any user or security man/woman can sign up for creating new account or can login into his/her existing account.

2. User Services: There are CRUD Services where you can create update delete any user if you are an admin or even you can get all user's data in the system. There are also some services are specific for the logged in users only like update his/her account details or update or add his/her profile image.

3. Tokens Services: These services are for the JWT tokens. There is a service for generating the JWT token for every user login or sign up. There is also another service for verifying the JWT token to prevent hackers from trying to access user account with invalid token.

4. Match Services: These are two services specific for the matching model one of them is to send the each child with the faces filtered for this child and the other for the matching result where the model send a request with the child id and the camera location to send a message to the parent with the camera location.

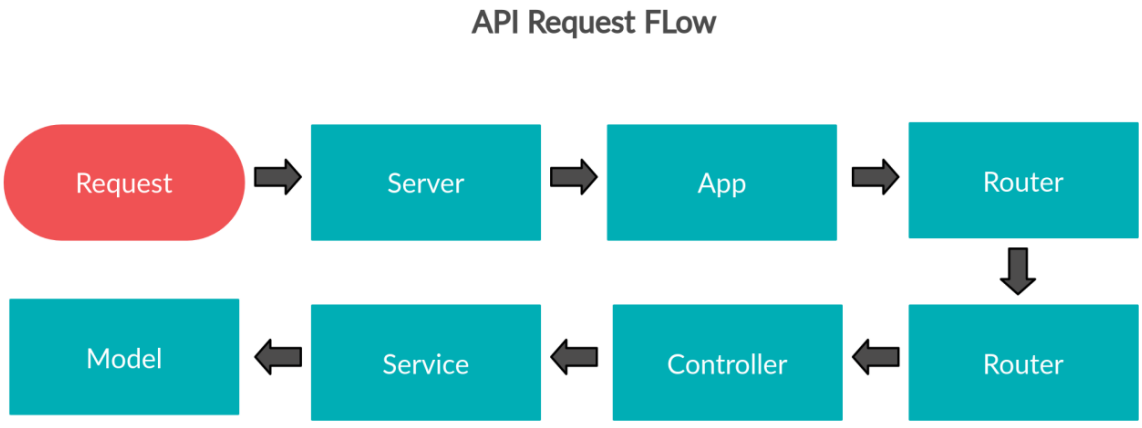
5. Children Services: These services are specific for lost or founded children, so the user can upload his lost child through the application or even anybody who found a child can upload the child data into the application.

6. Camera Services: These are services for the face detection model and age/gender model so they can send the camera faces with the predicted age and gender to the database.

7. SMS Services: These are services specific for verifying the phone verification code which sent to user phone number.

API Request Flow

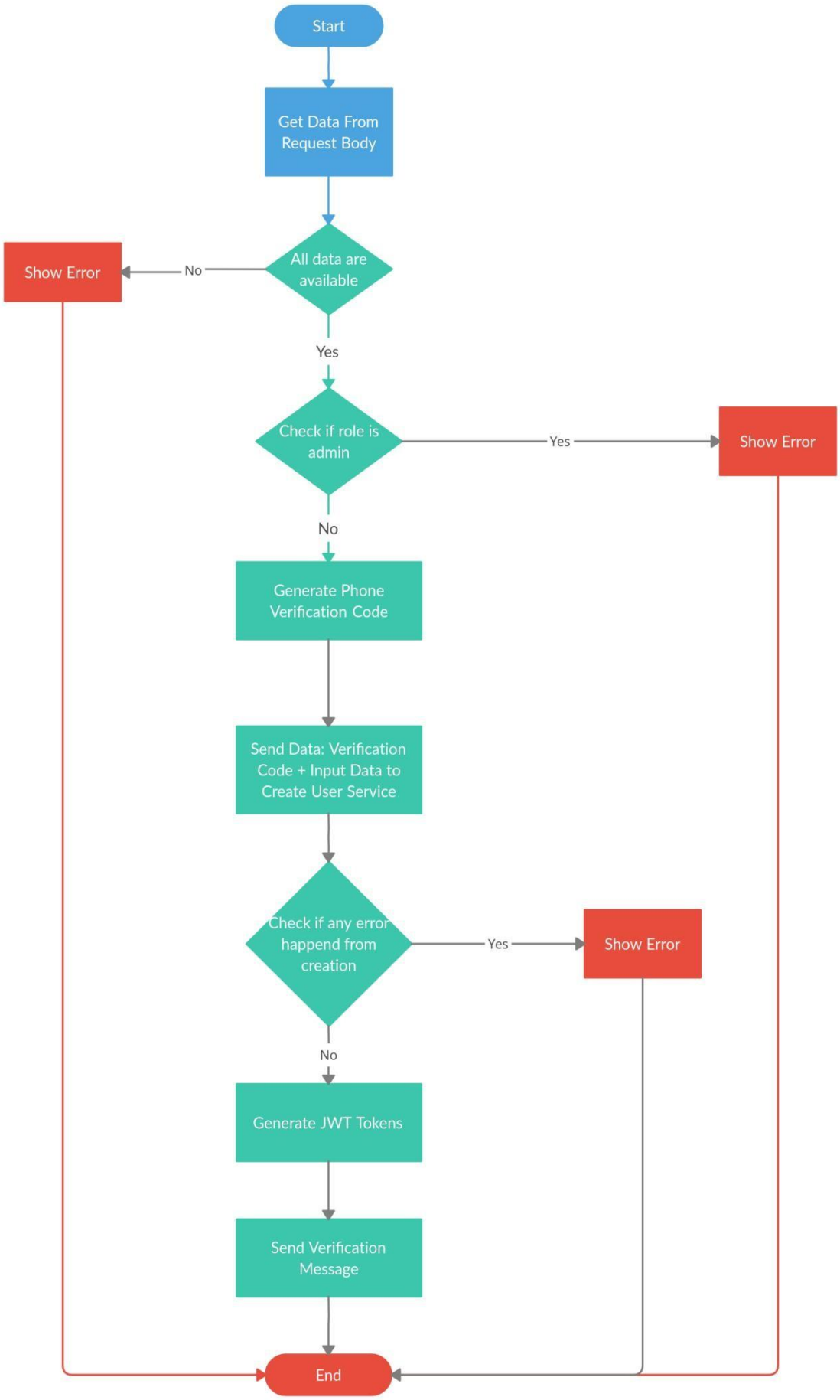
When someone try to make a request on any endpoint on the API, the server starts to take the request and sends it to the app server handler, next it see what is the main route to redirect the request to it and find the target route and hit it, next it finds the target controller and the request starts from there, now the controller make some processing on the request to send some data or even doesn't send anything to the service and the service return a result to us and the result usually from the database.



Authentication Services:

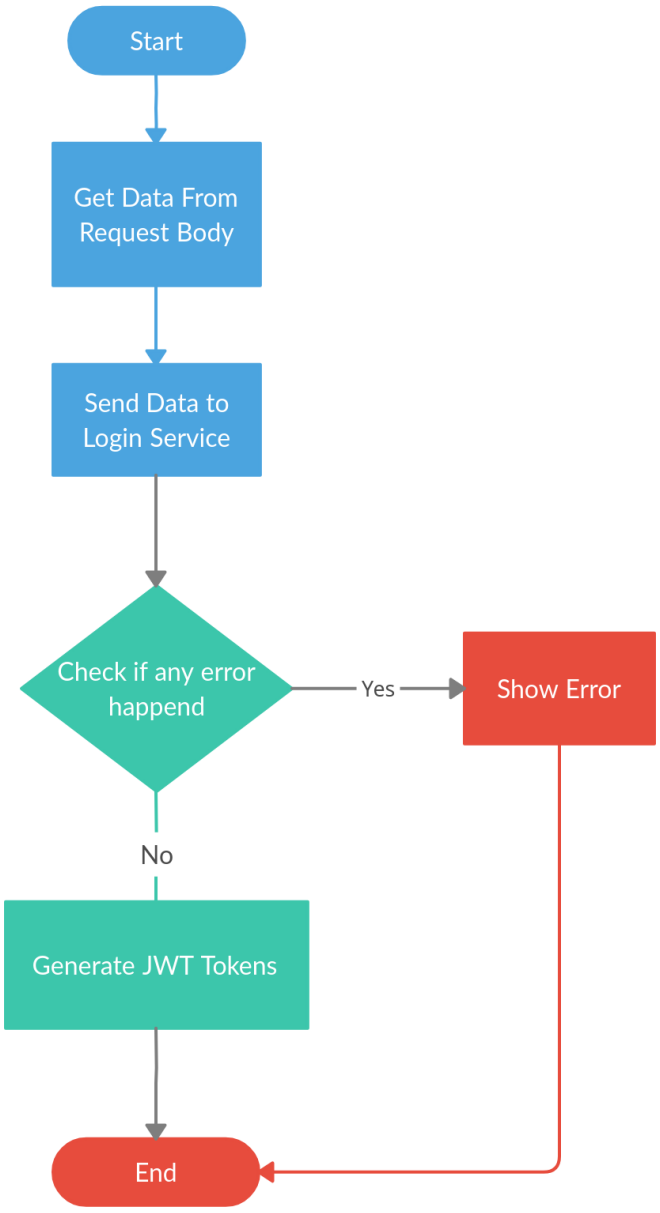
1. Registration Service: This Service let user sign up to create new account in the application. The required data are:
    - Name: it's a required string.
    - Email: it's a unique required string and must be in the email format.
    - Password: it's a required string a must contains letters and numbers and at least one capital letter.
    - Phone: it's a required string.
    - Role: it's a required string and must be user or security.

All these data are sent to the controller and the controller send the request data to the create user service for inserting user data into database. Then the controller generate a phone verification code and send the SMS message using SMS service to the user phone number and finally the controller calls the token service generateAuthTokens to generate authentication JWT tokens and send user data with tokens back in the response.


Start
Get Data From Request Body
All data are available
No
Show Error
Yes
Check if role is admin
Yes
Show Error
No
Generate Phone Verification Code
Send Data: Verification Code + Input Data to Create User Service
Check if any error happend from creation
Yes
Show Error
No
Generate JWT Tokens
Send Verification Message
End

2. Login Service: This Service let user to his/her account in the application. The required data:
    - Email: it's a required string.
    - Password: it's a required string.

All these data are sent throw the request to the authentication controller which calls the authentication service login to send the email and password to it, then it tries to find the user using it's email and match both passwords the entered password and the hashed password in the database.



SMS Services:

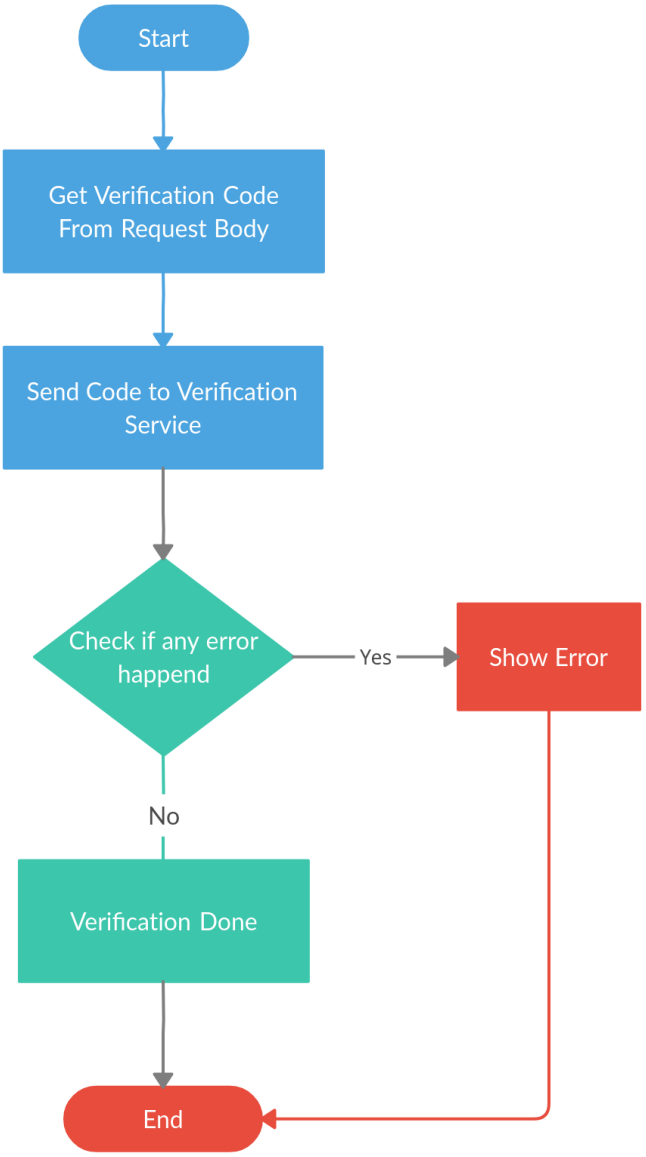
1. Send Verification Code Service:

   When user create new account, this service works by sending the verification code to the user phone number using provider called Vonage (nexmo).

2. Verify Verification Code Service:

   This service works when user enter the verification code so the service takes the code and search for user document in the user collection using it's id and then compare the phoneCode field with the verification code if both match then the service will delete the field and change the state of field isVerified from false to true and save the data

   .



User Services:

1. **Create User Service:**

   This service specific for creating user through the sign up or when the admin create new user because he/she have the permission to do that.

   The data which are passed to this service are:

   - name: it's a required string.
   - email : it's a required string.
   - password: it's a required string.
   - role: it's a required string.
   - phone: it's a required string.
   - phoneCode: it's a required string.

Then the service checks if all data are provided and check if the email already taken before if not the service will create the user data and insert it into database.

2. **Query Users Service:**

   This service allow you to get all user's data which save in the database and you can select any specific data instead of returning the whole data like just return users with the role security or you can sort data and even you can show specific fields to appear on these data.

3. **Query User Service:**

   This service allow you to get specific user using it's id.

4. **Update User Details Service:**

   This service allow admin to update user account details or even normal user to update his account details like name email phone. You cannot change you password through this service you have another service for doing that.

   The service works like the following:

   - Get user update data.
   - Search for user using his id in the database.

- If the user will change his email, then check if the inserted email is already taken or not.
- Finally update user data and save it.

5. Update User Profile Image Service:

This service allow admin to update user profile image or even normal user to update his account profile image.

The service works like the following:

- Search for user using his id in the database.
- Check if user uploaded the profile image or not.
- Destroy the user previous profile image from cloudinary.
- Upload the new profile image into cloudinary.
- Finally update user document fields (profileImage / profileImageId).

6. Add User Profile Image Service:

This service allow admin to add user profile image or even normal user to add his account profile image.

The service works like the following:

- Search for user using his id in the database.
- Check if user uploaded the profile image or not.
- Upload the new profile image into cloudinary.
- Finally update user document fields (profileImage / profileImageId).

7. Delete User Service:

This service allow admin to delete specific user using his id or even allow the user himself/herself to delete his account.

The service works like the following:

- Search for the user using his/her id.
- Destroy user profile image from cloudinary.
- Delete user data from database.

Token Services:

1. **Generate Authentication Tokens Service:**

   This service works only with the login or sign up services and it generate JWT tokens which will help the user to access the routes which are protected for only logged in users

2. **Verify Token Service:**

   This service takes the JWT token to check if it's a valid token or not by decoding the token using the JWT Secret and then check the token expiration time if it reached the expiration time or not. If it reached the time the the token is invalid and it logs out the user from the application and the user must login again to generate the new JWT token.

Camera Services:

1. **Add Face Service:**

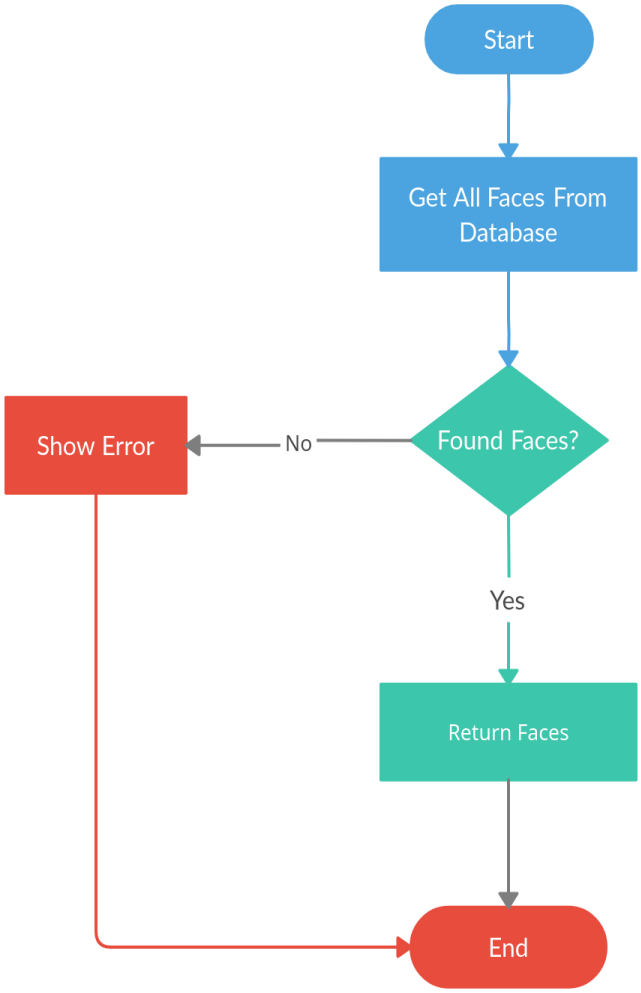
This service takes 5 inputs:

- age: it's a required number.
- gender: it's a required string.
- date: it's a required string (HH;MM AM/PM).
- location: it's a required string.

The service checks first if all data are provided then we uploads the face image into cloudinary and finally we create a new document in the camera collection in the database with the data inserted before.

2. Query Faces Service:

This service allows you to get all faces from database.



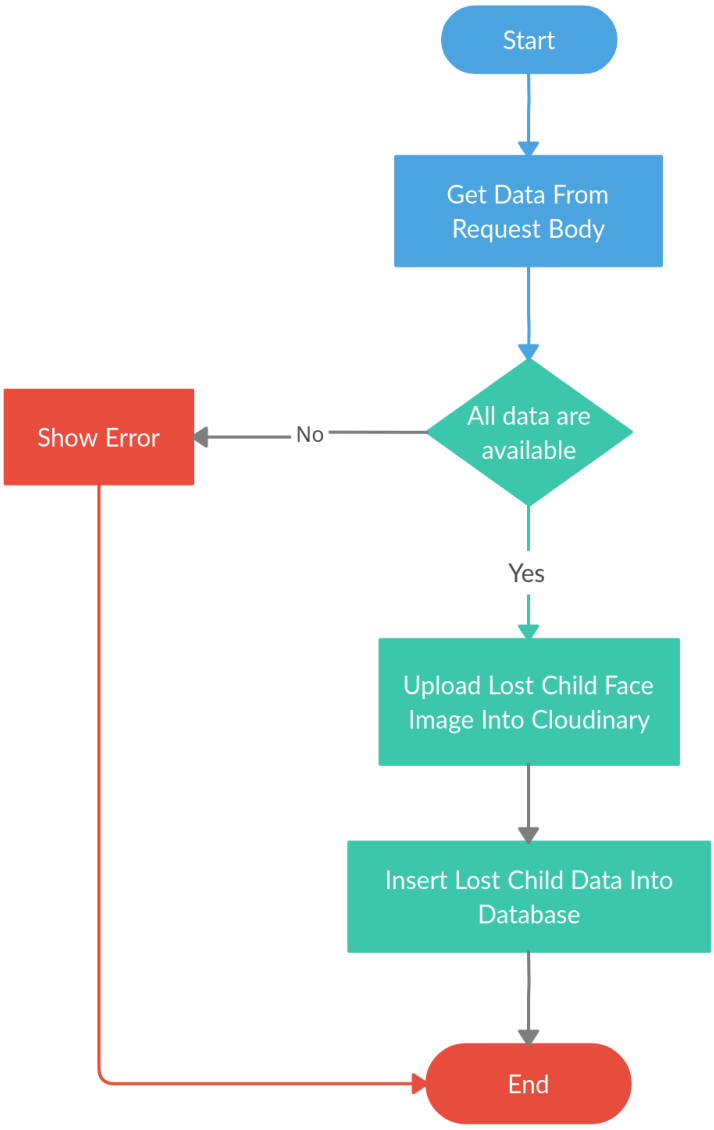
Child Services:

1. **Add Lost Child Service:**

This service allow you as a user to upload lost child data using application into the database to make the system find this child as soon as possible.

This service gets from the controller 5 pieces of data: age / gender / location / lostDate / image.

We check first if all these data are provided and nothing is missed, next we upload the lost child image into cloudinary and save his data as a new document in the database.

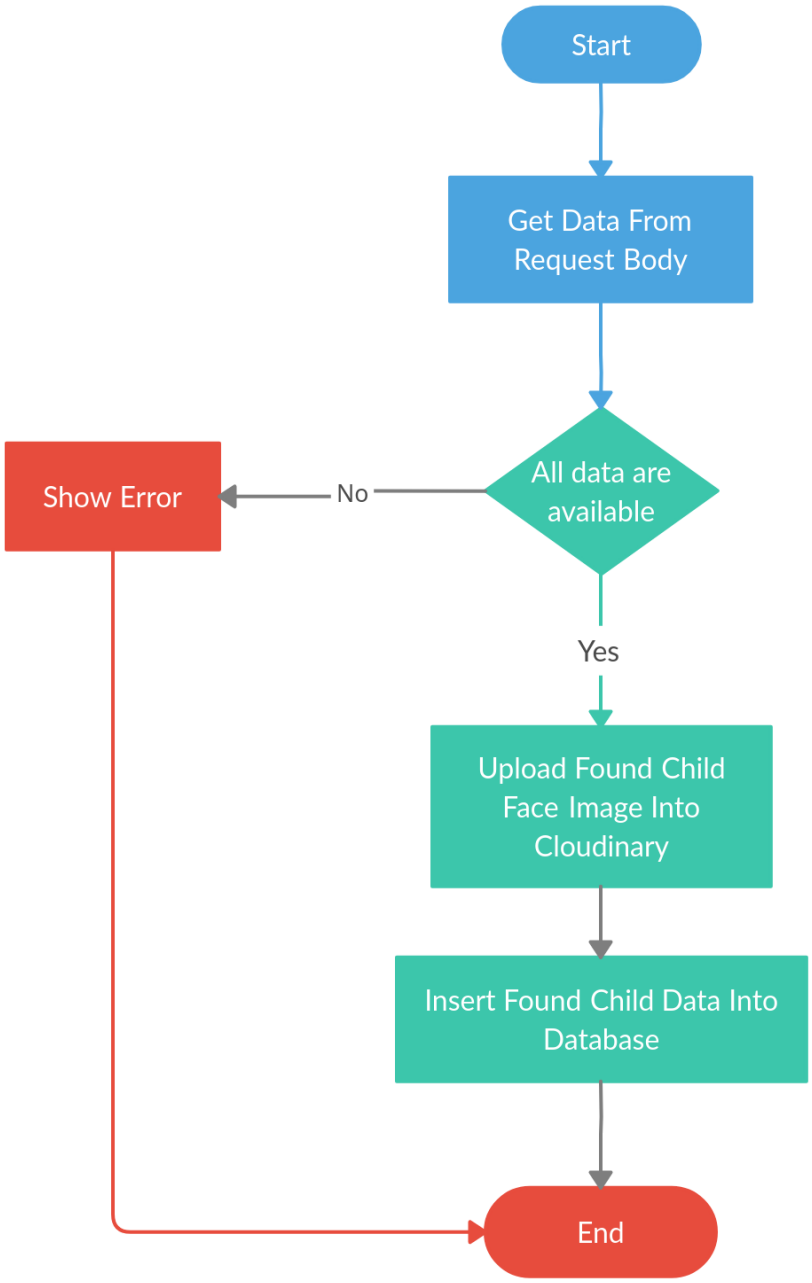


2. Add Found Child Service:

This service allow you as a user to upload founded child data using application into the database to help parents to find their child as soon as possible.

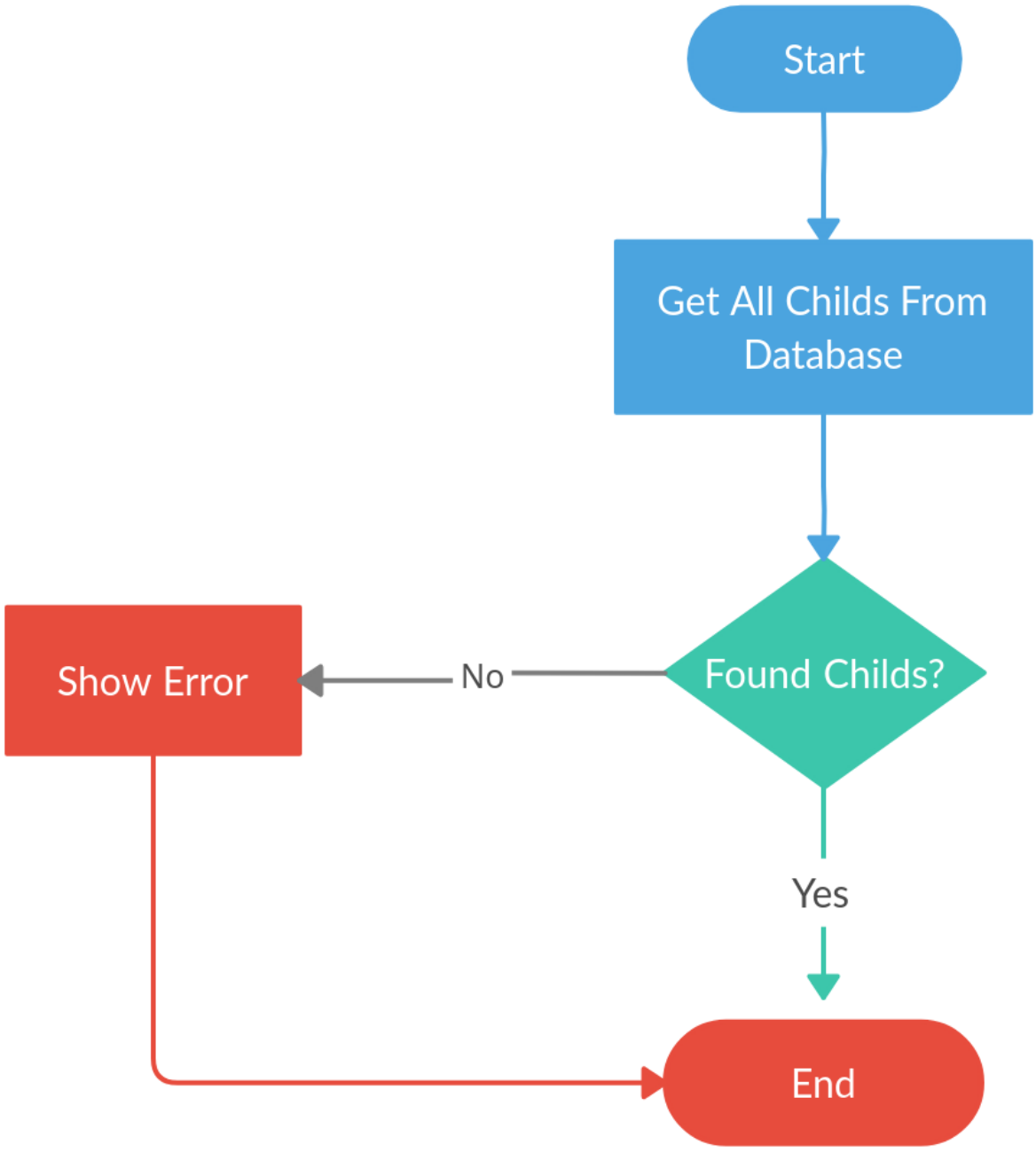
This service gets from the controller 5 pieces of data: age / gender / location / image.

We check first if all these data are provided and nothing is missed, next we upload the founded child image into cloudinary and save his data as a new document in the database.



3. Query Childs Service:

This service allow you to get all founded childs and lost childs together and return them to you.



Match Services:

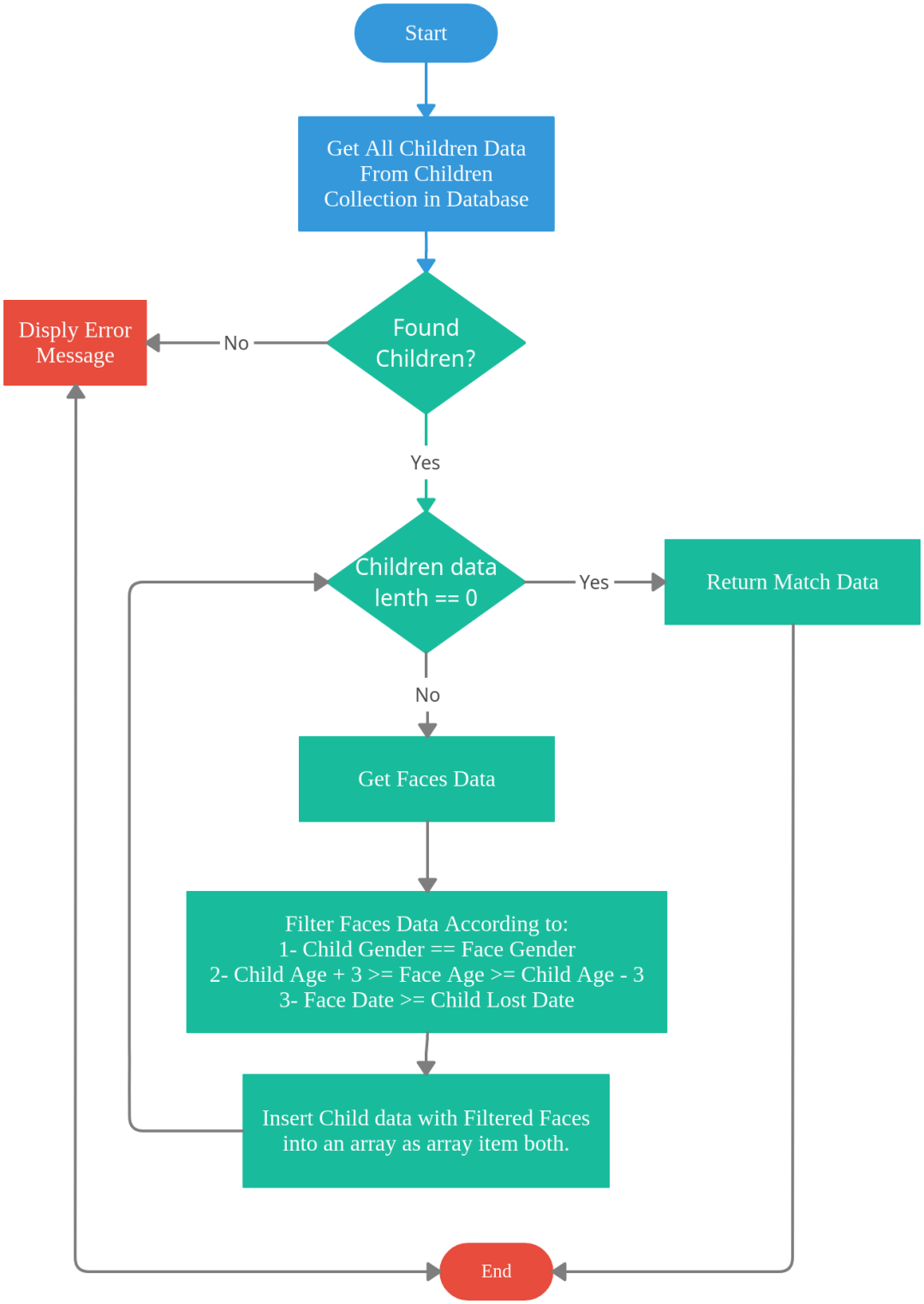
1. **Match Service:**

This service collects lost child data and find their nearly closed faces to them in gender age and date. Like when you have a female child with age 8 years old and lost in 9 AM so you will find all faces with gender female and age between 8+3 and 8-3 and also filter them to find only images with date greater than or equal 9 AM so you don't get faces before 9 AM.

We make these filter phases on every lost child object until we finish all child, so at the end we have array of arrays each element is a child data with it's faces so the first element contains two elements the child object and the faces array of objects.

```
[
    [
        {child data},
        [
        {face 1}, {face 2}, {face 3}, and so on
        ]
    ],
    [],
    [], and so on
]
```

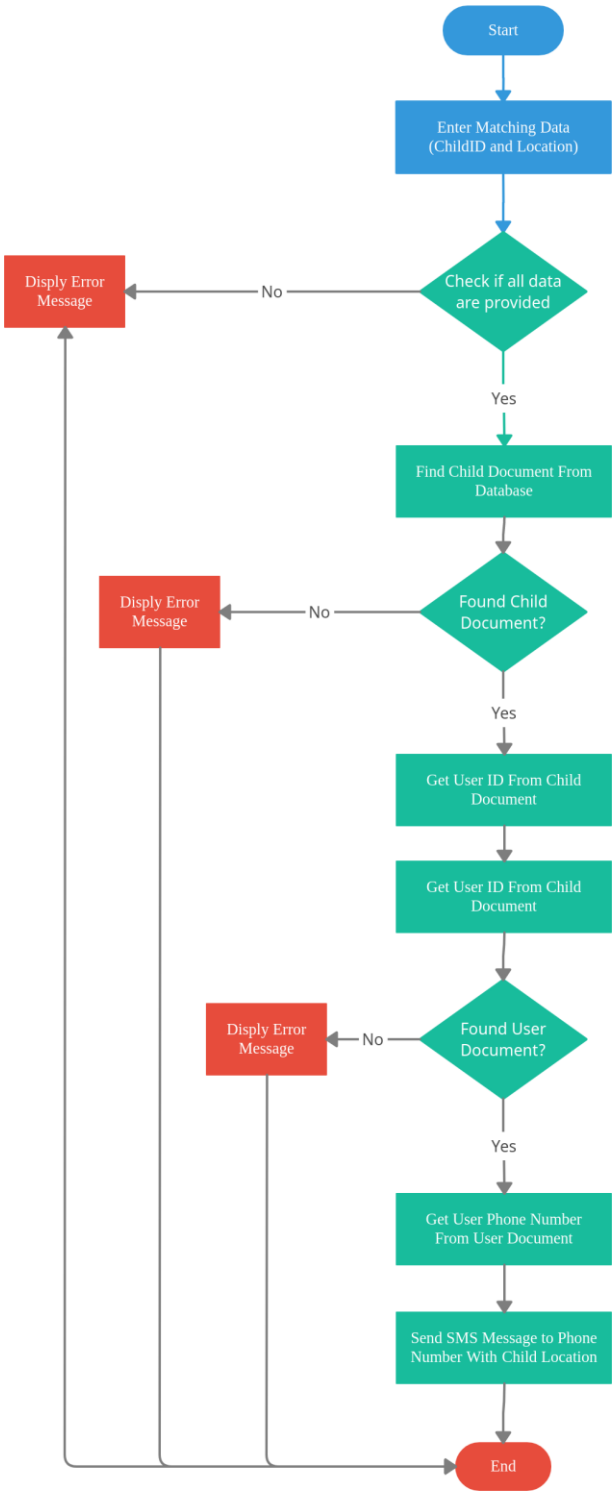
That's the final result of the matching service.


Start
Get All Children Data From Children Collection in Database
Found Children?
No
Disply Error Message
Yes
Children data lenth == 0
Yes
Return Match Data
No
Get Faces Data
Filter Faces Data According to:
1- Child Gender == Face Gender
2- Child Age + 3 >= Face Age >= Child Age - 3
3- Face Date >= Child Lost Date
Insert Child data with Filtered Faces into an array as array item both.
End

2. Match Result Service:

This service is for the matching model so when there is a similarity the model calls this service and send the child id and the camera location of the face to this service.

The service takes the child id and the camera location and send SMS message to the parent with his/her child location.



4.2.2.3 MACHINE LEARNING AND MODELS

4.2.2.3.1 face detection:

Face detection is a subset of object detection in which the goal is to locate and size all of the faces in a given image. Face detection may appear to be a simple process for humans, but it is a difficult for computers Because there are numerous challenges face computers (like scale, rotation, facial expressions, occlusion, or lighting condition).

We tried more than one model until we settled on the ideal model in terms of price and results and the perfect one was cvlib and we will discuss all below:

1. Face detection with haarcascade.
2. Face detection with Deep face Using Harcascade.
3. Face detection with Face detector (FP16): Floating-point 16 version of the original Caffe implementation.
4. Face detection with Face detector (UINT8): 8-bit quantized version using TensorFlow.
5. Face detection with dlib.
6. Face detection with cvlib.
7. Face detection with media pipe.

1. Haarcascade:

Paul Viola and Michael Jones introduced an effective object identification method utilising Haar feature-based cascade classifiers in their paper "Rapid Object Detection with a Boosted Cascade of Simple Features" in 2001. It's a machine-learning approach in which a cascade function is learned using a large number of positive and negative photos. Haar cascade detecting faces and object from photos. And we will use this algorithm in detecting faces. So this algorithm to work it needs to train the classifier a group of pictures containing faces (positive images) and a group of pictures without faces (negative images). Then it’s time for extracting features from images. The image below shows the haar features that were used. They are just similar to convolutional kernel. Each feature is a single value generated by subtracting the total of pixels beneath the white and black rectangles.

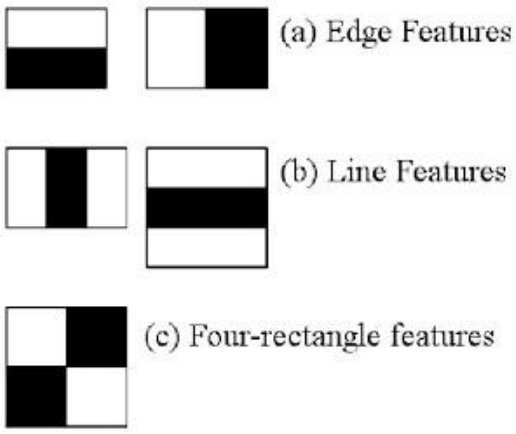


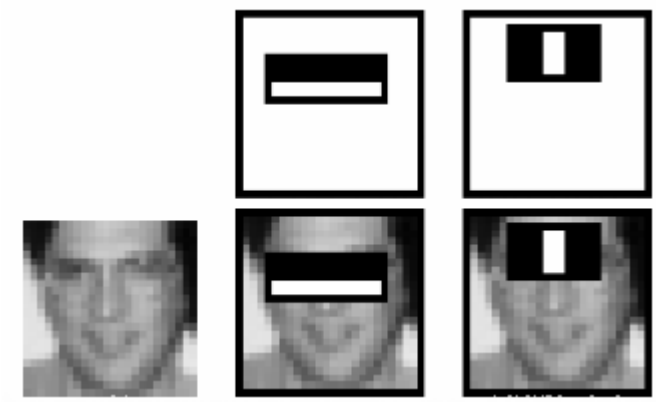
Figure 4.2.2.3.1.1: Haarcascade feature

Many features are now calculated using all potential sizes and positions of each kernel. (Can you image how much computing power is required? A 24x24 window yields more than 160000 features). We must find the total of pixels beneath white and black rectangles for each feature computation.

They used integral pictures to address the problem. It reduces the sum of pixels computation, regardless of how high the number of pixels is, to a four-pixel operation. Isn't it lovely? It speeds up the process. However, the majority of the characteristics we computed are irrelevant.

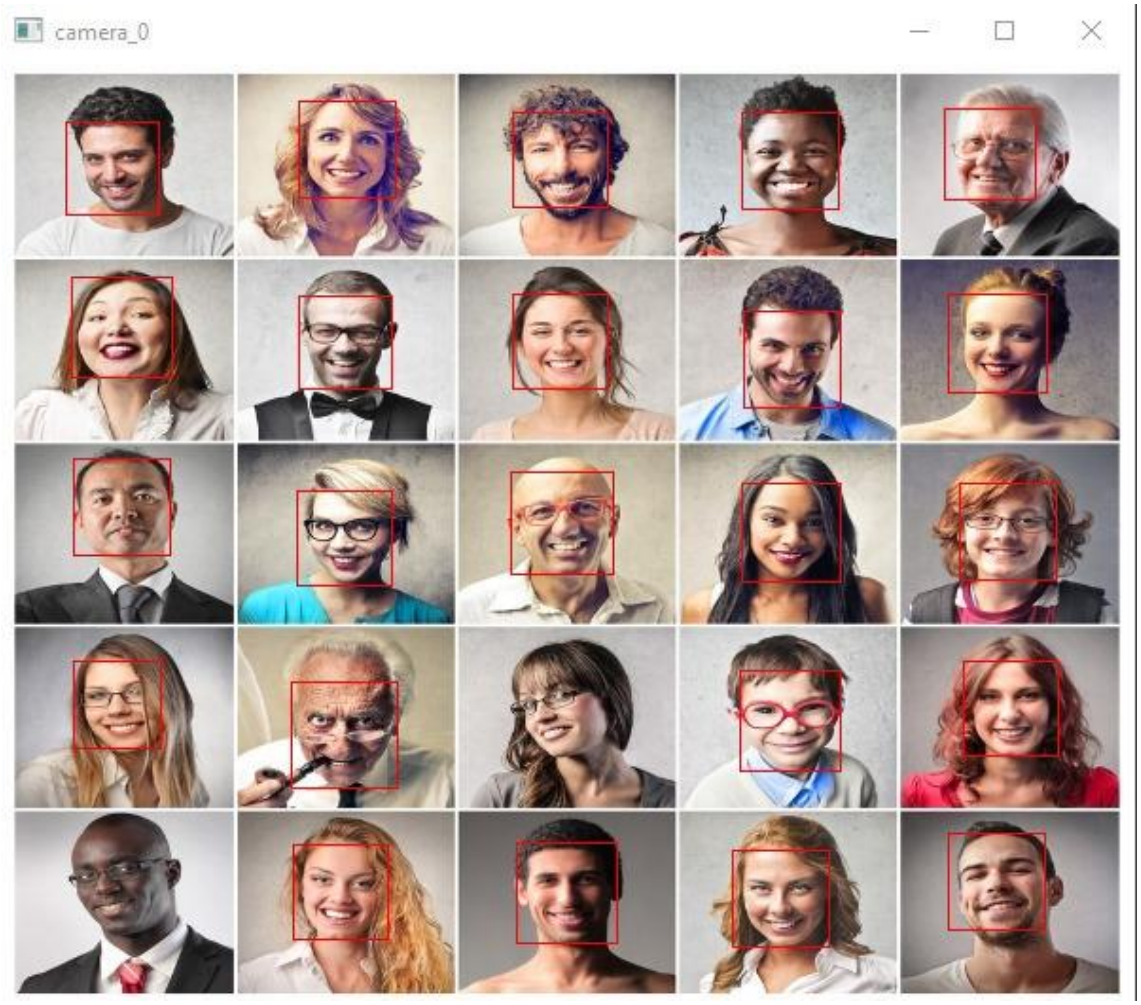
Two good features are shown in the top row image below. The first feature chosen appears to be the fact that the area around the eyes is generally darker than

the area around the nose and cheekbones. The second feature chosen is based on the fact that the eyes are darker than the nasal bridge. However, whether the same windows are used on the cheeks or anywhere else is immaterial. So, how do we pick the greatest features from a list of 160000+ options? Adaboost does this, as illustrated in Figure 4.2.2.3.1.1.



We do this by applying each feature to all of the training pictures. It calculates the optimal threshold for each characteristic to identify the faces as positive or negative. However, there will undoubtedly be mistakes or misclassifications. We choose characteristics with the lowest error rate, which implies they're the ones that best distinguish between face and non-facial pictures. (The procedure is not as straightforward as this.) At first, each image is assigned the same weight. The weights of misclassified pictures are raised after each categorization. Then the procedure is repeated. Error rates are computed at new levels. There are also new weights. The procedure is repeated until the desired accuracy or error rate is met, or until the required number of features is discovered). Weak classifiers collected in final classifier and weak here because they can’t classify alone image, but they can with a strong classifier. Even 200 features, according to the paper, give 95 percent accuracy in detection. Around 6000 features were included in their final setup. (Imagine a reduction in the number of features from 160000 to 6000 That's a significant gain). So now you're going to take a picture. then we should Take a 24x24 window and Appling 6000 to it. Check to see if it's a face or not. and it is a time consuming and inefficient. And there is a solution in paper. The non-face area of a picture makes up the majority of the image. As a result, having a simple technique to verify if a window is not a face region is a better approach. If it isn't, toss it out in one shot. It should not be processed again. Instead, concentrate on areas

where a face could appear. This manner, we'll have more time to examine a potential facial region. They came up with the idea of a Cascade of Classifiers to do this. Rather than applying all 6000 characteristics to a single window, divide them into various stages of classifiers and apply them one by one. (In most cases, the initial few phases will include a small number of features). If a window fails the first test, it should be discarded. We don't think about the remaining features. Apply the second stage of features and continue the procedure if it passes. A face region is a window that travels through all stages. The authors' detector contained around 6000 features and 38 stages, with the first five stages consisting of 1, 10, 25, 25, and 50 features. (The two features in the above picture were retrieved from Adaboost as the best two features.) According to the authors, 10 characteristics out of 6000+ are examined on average each sub-window. So, there you have it: a basic understanding of how Viola-Jones face detection works. For additional information, read the article or look up the references in the Additional Resources section.



```
while True:
    # parallel
    # camera 0
    check_0, frame_0 = vidoe_0.read()
    frame_0 = imread('1920_stock-photo-mosaic-of-satisfied-people-157248584.jpg')
    frame_0 = resize(frame_0, (600, 500))
    if not check_0:
        face_deteced_points_0 = face_deteced.detectMultiScale(cvtColor(frame_0, COLOR_BGR2GRAY), 1.05, 5)
        for x, y, w, h in face_deteced_points_0:
            face_save = frame_0[y:y + h, x:x + w]
            path = "faces saved/camera_0/camera_0_" + str(a) + ".png"
            imwrite(path, face_save)
            public_id = "camera_0_" + str(a)
            # response = network.uploadFun(path, "Camera0")
            # response2 = network.postUrl(response)
            frame_0 = rectangle(frame_0, (x, y), (x + w, y + h), (0, 0, 255), 1)
            # print(response2.text)
            a = a + 1
    else:
        frame_0 = imread('error\Error playing camrea.png')

    imshow("camera_0", frame_0)
```

advantage

- Works almost real-time on CPU
- Its Architecture is simple
- Detects faces at different scales

Cons

1. An old technique that is currently not the best technique in face detection
2. It depends on a scale number in order to be able to determine the noise in the picture, and with a high scale number, good results will appear, but it takes a lot of time, so we must specify a scale number that works for everyone
3. It gives large number of false predictions.
4. Doesn’t work on images that aren’t frontal images.
5. Doesn't operate in the presence of occlusion

2. Deep face

Deepface is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. It is a hybrid face recognition framework wrapping state-of-the-art models: VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepID, ArcFace and Dlib. Those models already reached and passed the human level accuracy. The library is mainly based on Keras and TensorFlow.

Face detection may be done with the OpenCV package and the classical feature-based cascade classifier.

In deep face: We have 4 common stages: detect, align, represent and verify.

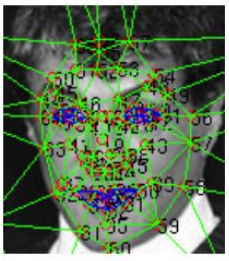
Alignment:

generating frontal face from the input image. The method used is 3D frontalization of faces based on the fiducial (face feature points) to extract the frontal face. The whole alignment process is done in the following steps:

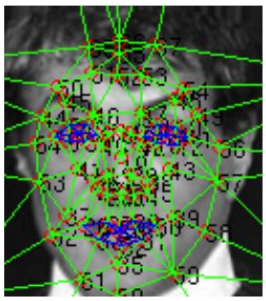
1. Given an input image, we first identify the face using six fiducial points. These six fiducial points are 2 eyes, tip of the nose and 3 points on the lips. These feature points are used to detect faces in the image.

2. 2D-face image cropped from the original image is generated using 6 fiducial points.

3. 67 fiducial point map with their corresponding Delaunay Triangulation are applied on the 2D-aligned cropped image. The out of plane rotations are being aligned. We generate a 3D-model using a generic 2D to 3D model generator.



4. Then we try to establish a relation between 2D and 3D



5. Frontalization of alignment. But first the residual component are added to x-y coordinates of 3D warp because it reduces corruption in 3D-warp. Frontalization is achieved by doing piece-wise affine on Delauney triangulation that we generated on 67-fiducial points.

Representation and Classification Architecture:

Taking input and passed it into 3D-aligned RGB image of 152*152. Then passed the image to convolution layer with 32 filters with size 11*11*3 and a 3*3 max-pooling layer. Then passed output to another convolution layer of 16 filters with size 9*9*16. Then we can extract low-level features from the image edges and textures.

The next three layers are connected which a type of fully connected layer that has different types of filters in a different feature map. This helps in improving the model because different regions of the face have different discrimination ability, so it is better to have different types of feature maps to distinguish these facial regions.

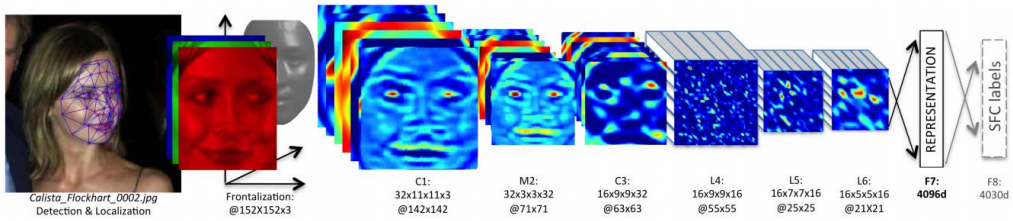


Figure 1DeepFace full architecture

The last two layers of the model are fully connected layers. These goals is establishing a correlation between two distant parts of the face. The output of the second last fully connected layer is used as a face representation and the output of the last layer is the softmax layer K classes for the classification of the face. Accuracy is 91.4% and it still the state-of-the-art accuracy at that time and reduces the error rate by more than 50%.

conclusion

It was one of the finest facial recognition models at the time of its release; today, models like Google-FaceNet and others give accuracy up to 99.6% on the LFW dataset.

The key challenge that DeepFace has solved is to create a model that is insensitive to light, posture, facial expression, and other factors. The model's accuracy was also improved because to the unique method of using 3D alignment.

3. Face detector (FP16): Floating-point 16 version of the original Caffe implementation

Here we are using pre-trained deep learning face detector model, to perform face detection which open cv provide this model. Opencv library included it. We should have the configuration file and the model file.

- First file contains: actual layer Weights

  res10_300x300_ssd_iter_140000_fp16.caffemodel

  https://github.com/opencv/opencv_3rdparty/raw/19512576c112aa2c7b6328cb0e8d589a4a90a26d/res10_300x300_ssd_iter_140000_fp16.caffemodel
- Second file defines: the model architecture

  deploy.prototxt

  https://github.com/opencv/opencv/blob/master/samples/dnn/face_detector/deploy.prototxt

```
# for video
vs = cv2.VideoCapture('Face detection.mp4')

while vs.isOpened():
    check, image = vs.read()
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    image = imutils.resize(gray, width=500)
    # detect faces in the grayscale image
    rects = detector(image, 1)

    # loop over the face detections
    for (i, rect) in enumerate(rects):
        shape = predictor(gray, rect)
        shape = face_utils.shape_to_np(shape)
        # convert dlib's rectangle to a OpenCV-style bounding box
        # [i.e., (x, y, w, h)], then draw the face bounding box
        (x, y, w, h) = face_utils.rect_to_bb(rect)
        cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
        # show the face number
        cv2.putText(image, "Face #{}".format(i + 1), (x - 10, y - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
        # loop over the (x, y)-coordinates for the facial landmarks
        # and draw them on the image
        # for (x, y) in shape:
        #     cv2.circle(image, (x, y), 1, (0, 0, 255), -1)
    # show the output image with the face detections + facial landmarks
    cv2.imshow("Output", image)
    cv2.waitKey(1)
cv2.destroyAllWindows()
```

His shortcomings were slow and separated a lot

4. Face detector (UINT8): 8-bit quantized version using TensorFlow

Here we are using pre-trained deep learning face detector model, to perform face detection which open cv provide this model. Opencv library included it. We should have the configuration file and the model file.

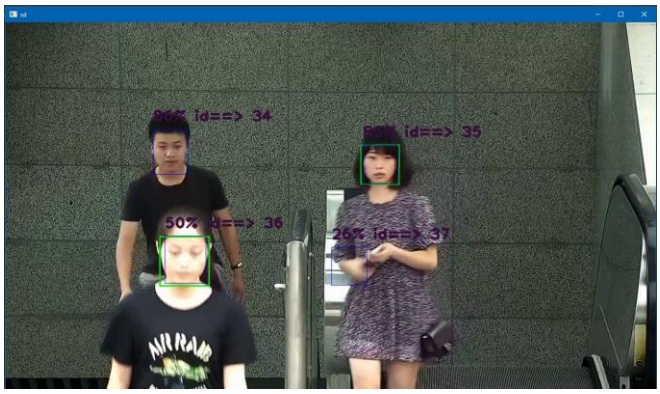
- First file contains: actual layer Weights

  opencv_face_detector_uint8.pb

- Second file defines: the model architecture

  opencv_face_detector.pbtxt

  https://github.com/opencv/opencv_extra/blob/master/testdata/dnn/opencv_face_detector.pbtxt



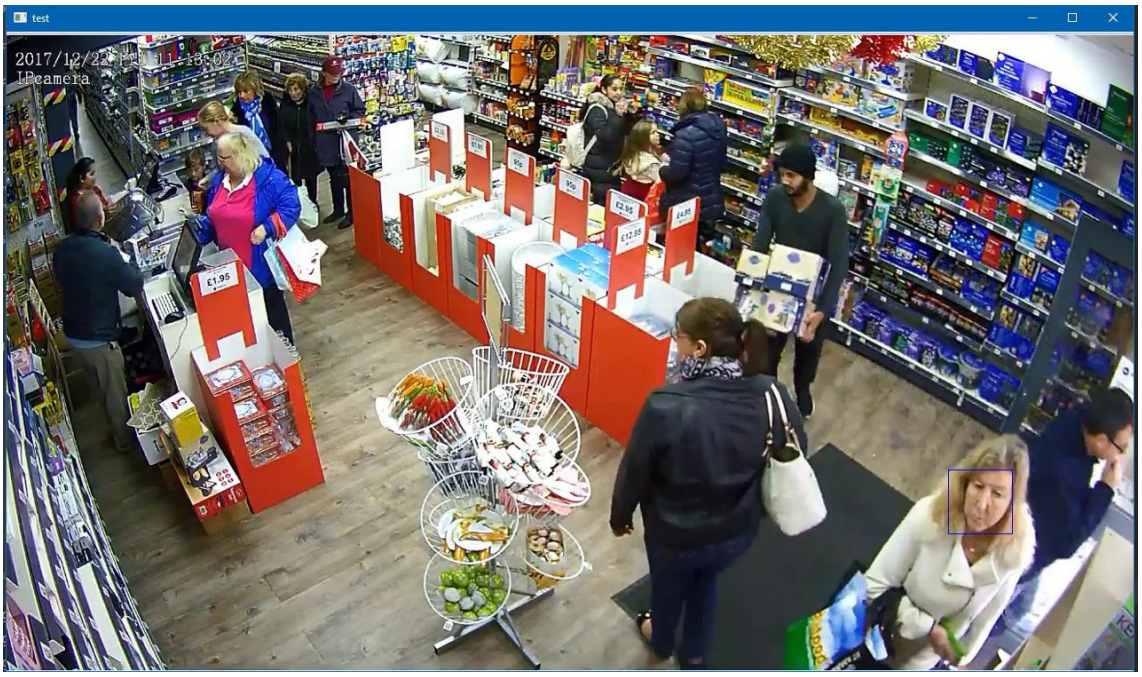
```
s = "opencv_face_detect.pbtxt"
d = "opencv_face_detector_uint8.pb"
# load
net = cv2.dnn.readNetFromTensorflow(d, s)
vs = cv2.VideoCapture("Face detection.mp4")
```

```
tracker = EuclideanDistTracker()
lisyt = []
last_acc = None
last_id = None
count = -1
while True:
    tr = []
    confidence_list = []
    check, image = vs.read()
    blob = cv2.dnn.blobFromImage(image=image, scalefactor=1.0, size=(300, 300), mean=[104, 117, 123])
    # Set the blob as input and obtain the detections:
    net.setInput(blob)
    detections = net.forward()
    detected_faces = 0
    (h, w) = image.shape[:2]
    for i in range(0, detections.shape[2]):
        # Get the confidence (probability) of the current detection:
        confidence = detections[0, 0, i, 2]
        # Only consider detections if confidence is greater than a fixed minimum confidence:
        if confidence > 0.25:  # Increment the number of detected faces:
            detected_faces += 1
            # Get the coordinates of the current detection:
            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            boxs = detections[0, 0, i, 2:7] * np.array([100, w, h, w, h])
            tr.append(list(boxs.astype("int")))
            (startX, startY, endX, endY) = box.astype("int")
            (startX, startY, endX, endY) = box.astype("int")
    boxes_ids = tracker.update(tr)
```

```
    for box_id in boxes_ids:
        acurcy, startX, startY, endX, endY, id = box_id
        text = str(acurcy) + "% " + "id==> " + str(id)
        cv2.putText(image, text, (startX, startY - 15), cv2.FONT_HERSHEY_PLAIN, 2, (60, 15, 56), 3)
        image = cv2.rectangle(image, (startX, startY), (endX, endY), (255, 0, 0), 1)
        if not lisyt:
            last_acc = acurcy
            last_id = id
            lisyt.append((id, acurcy))
            count += 1
        if (id == last_id) and (acurcy > last_acc):
            last_acc = acurcy
            last_id = id
            lisyt[count] = (id, acurcy)
            # print("replace if")
        elif id == last_id and acurcy < last_acc:
            continue
        elif id != last_id:
            last_acc = acurcy
            last_id = id
            lisyt.append((id, acurcy))
            count += 1

    cv2.imshow("sd", image)
    key = cv2.waitKey(1)
    if key == 'q':
        break
cv2.destroyAllWindows()
```

Good for far cameras, not near ones

5. Face detection with dlib




```
detector = dlib.get_frontal_face_detector()
vs = cv2.VideoCapture("HD CCTV Camera video 3MP 4MP iProx CCTV HDCCTVCameras.net retail store.mp4")
fps = round(vs.get(cv2.CAP_PROP_FPS))
width = vs.get(cv2.CAP_PROP_FRAME_WIDTH)
hight = vs.get(cv2.CAP_PROP_FRAME_HEIGHT)

while True:
    check, fame = vs.read()
    gray = cv2.cvtColor(fame, cv2.COLOR_BGR2GRAY)
    # gray = cv2.resize(gray, (300, 300))
    detect = detector(gray, 0)
    # print(detect)

    cv2.imshow("test", show_detection(fame, detect))
    key = cv2.waitKey(fps)
    if key & 0xff == ord('q'):
        break

cv2.destroyAllWindows()
```

```
import dlib
import cv2

cnn_face_detector = dlib.cnn_face_detection_model_v1("mmod_human_face_detector.dat")

# image = cv2.imread("1920_stock-photo-mosaic-of-satisfied-people-157248584.jpg")
vs = cv2.VideoCapture('Face detection.mp4')

while vs.isOpened():
    check, fame = vs.read()
    gray = cv2.cvtColor(fame, cv2.COLOR_BGR2GRAY)
    gray = cv2.resize(gray, (300, 300))
    detect = cnn_face_detector(fame, 0)
    for face in detect:
        cv2.rectangle(gray, (face.rect.left(), face.rect.top()), (face.rect.right(), face.rect.bottom()), (255, 0, 0),
                      1)
    cv2.imshow("xx", gray)
    cv2.waitKey(1)

cv2.destroyAllWindows()
```

Advantage:

- dlib is much accurate as compared to OpenCV Haar based face detector but still slow.
- Fastest on CPU
- Works very well for frontal and slightly non-frontal faces
- Works under small occlusion

Disadvantage:

- But it does not detect small sized faces (< 70x70).
- The bounding box frequently excludes a portion of the forehead and, on rare occasions, a portion of the chin.
- Doesn't operate well when there's a lot of occlusion.
- Doesn't function for side and non-frontal faces, such as gazing down or up.

6. media pipe

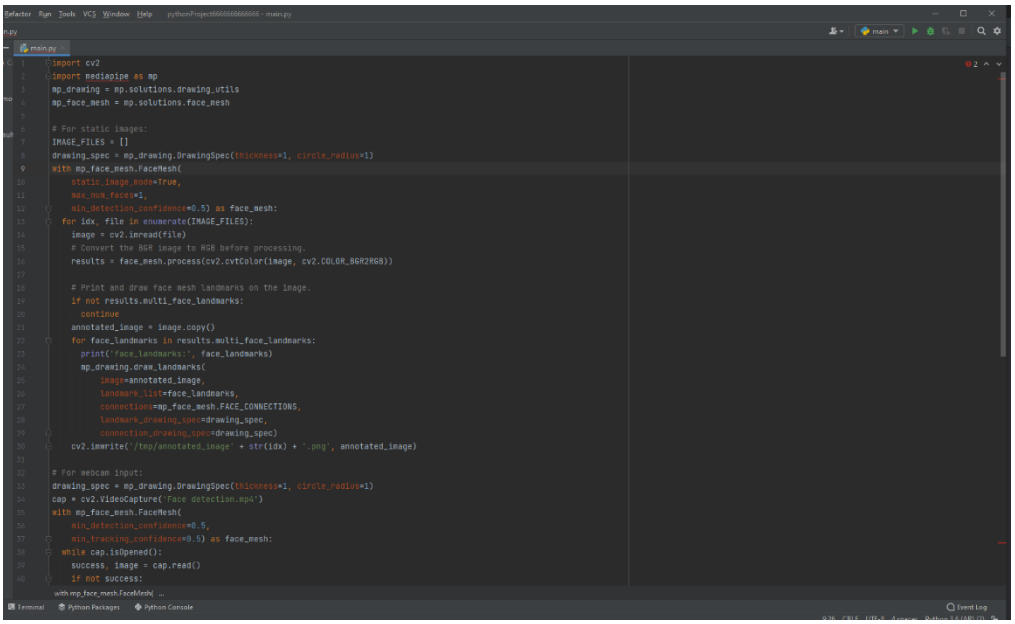
MediaPipe Face Mesh like face land marks, difference in number of points and shape, it is a face geometry solution that estimates 468 3D face landmarks in real-time even on mobile devices. It employs machine learning (ML) to infer the 3D surface geometry, requiring only a single camera input without the need for a dedicated depth sensor. Utilizing lightweight model architectures together with GPU acceleration throughout the pipeline, the solution delivers real-time performance critical for live experiences.

Additionally, the solution is bundled with the Face Geometry module that bridges the gap between the face landmark estimation and useful real-time augmented reality (AR) applications. It establishes a metric 3D space and uses the face landmark screen positions to estimate face geometry within that space. The face geometry data consists of common 3D geometry primitives, including a face pose transformation matrix and a triangular face mesh. Under the hood, a lightweight statistical analysis method called Procrustes Analysis is employed to drive a robust, performant and portable logic. The analysis runs on CPU and has a minimal speed/memory footprint on top of the ML model inference.

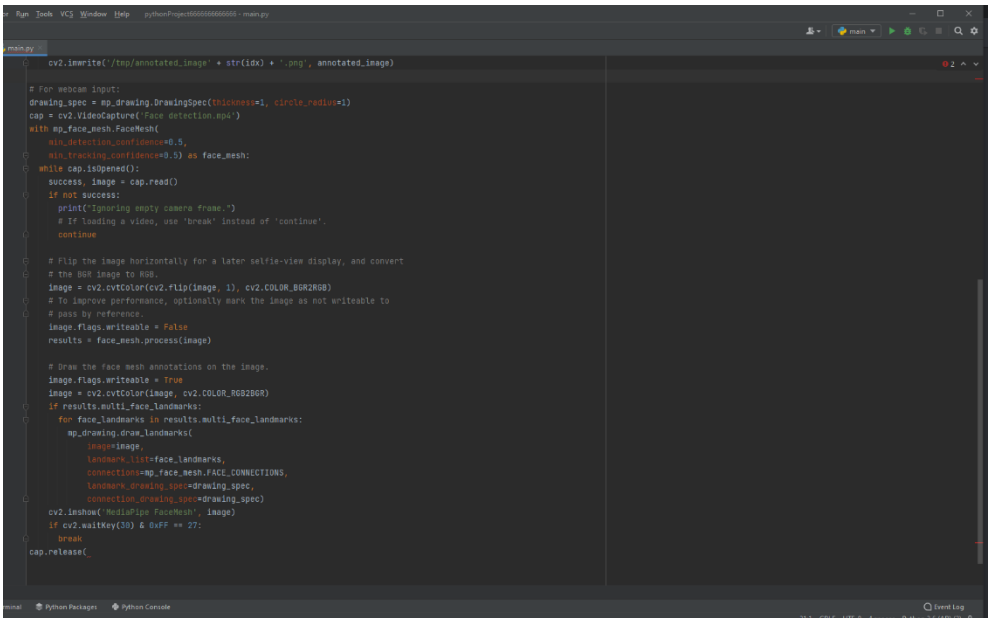
The facemesh package infers approximate 3D facial surface geometry from an image or video stream, requiring only a single camera input without the need for a depth sensor. This geometry locates features such as the eyes, nose, and lips within the face, including details such as lip contours and the facial silhouette.

Facemesh is a lightweight package containing only ~3MB of weights, making it ideally suited for real-time inference on a variety of mobile devices.

To install it we will need cv2 and mediapipe as mp and operating system



```
import cv2
import mediapipe as mp
mp_drawing = mp.solutions.drawing_utils
mp_face_mesh = mp.solutions.face_mesh

# For static images:
IMAGE_FILES = []
drawing_spec = mp_drawing.DrawingSpec(thickness=1, circle_radius=1)
with mp_face_mesh.FaceMesh(
    static_image_mode=True,
    max_num_faces=1,
    min_detection_confidence=0.5) as face_mesh:
  for idx, file in enumerate(IMAGE_FILES):
    image = cv2.imread(file)
    # Convert the BGR image to RGB before processing.
    results = face_mesh.process(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))

    # Print and draw face mesh landmarks on the image.
    if not results.multi_face_landmarks:
      continue
    annotated_image = image.copy()
    for face_landmarks in results.multi_face_landmarks:
      print('face_landmarks:', face_landmarks)
      mp_drawing.draw_landmarks(
          image=annotated_image,
          landmark_list=face_landmarks,
          connections=mp_face_mesh.FACE_CONNECTIONS,
          landmark_drawing_spec=drawing_spec,
          connection_drawing_spec=drawing_spec)
    cv2.imwrite('/tmp/annotated_image' + str(idx) + '.png', annotated_image)

# For webcam input:
drawing_spec = mp_drawing.DrawingSpec(thickness=1, circle_radius=1)
cap = cv2.VideoCapture('Face detection.mp4')
with mp_face_mesh.FaceMesh(
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5) as face_mesh:
  while cap.isOpened():
    success, image = cap.read()
    if not success:
```



```
    cv2.imwrite('/tmp/annotated_image' + str(idx) + '.png', annotated_image)

# For webcam input:
drawing_spec = mp_drawing.DrawingSpec(thickness=1, circle_radius=1)
cap = cv2.VideoCapture('Face detection.mp4')
with mp_face_mesh.FaceMesh(
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5) as face_mesh:
  while cap.isOpened():
    success, image = cap.read()
    if not success:
      print("Ignoring empty camera frame.")
      # If loading a video, use 'break' instead of 'continue'.
      continue

    # Flip the image horizontally for a later selfie-view display, and convert
    # the BGR image to RGB.
    image = cv2.cvtColor(cv2.flip(image, 1), cv2.COLOR_BGR2RGB)
    # To improve performance, optionally mark the image as not writeable to
    # pass by reference.
    image.flags.writeable = False
    results = face_mesh.process(image)

    # Draw the face mesh annotations on the image.
    image.flags.writeable = True
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
    if results.multi_face_landmarks:
      for face_landmarks in results.multi_face_landmarks:
        mp_drawing.draw_landmarks(
            image=image,
            landmark_list=face_landmarks,
            connections=mp_face_mesh.FACE_CONNECTIONS,
            landmark_drawing_spec=drawing_spec,
            connection_drawing_spec=drawing_spec)
    cv2.imshow('MediaPipe FaceMesh', image)
    if cv2.waitKey(30) & 0xFF == 27:
      break
cap.release(
```

Disadvantage

It requires high Gpu

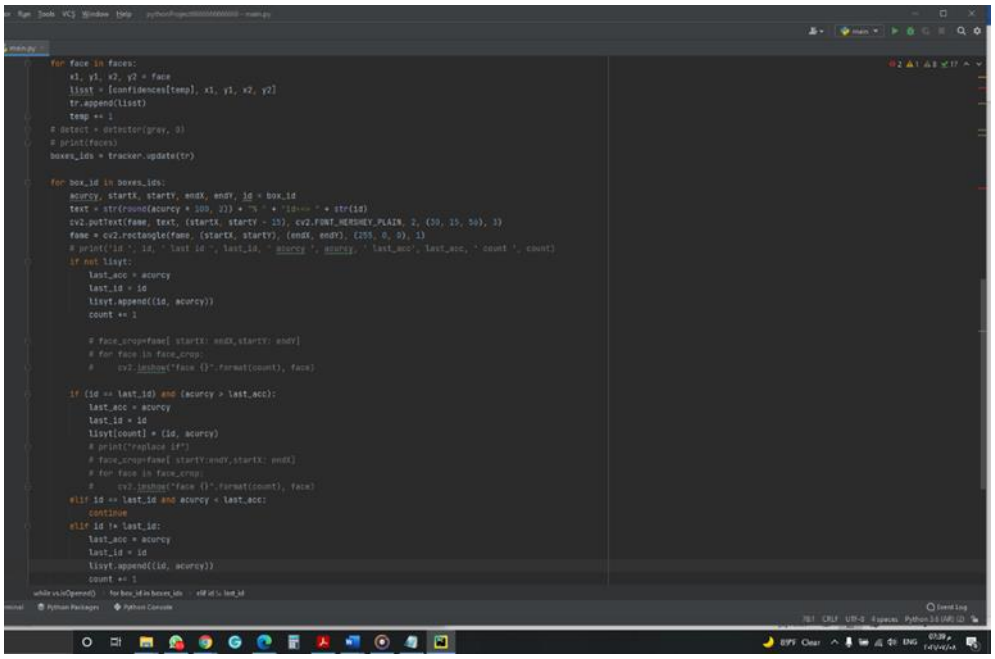
7. Cvlib

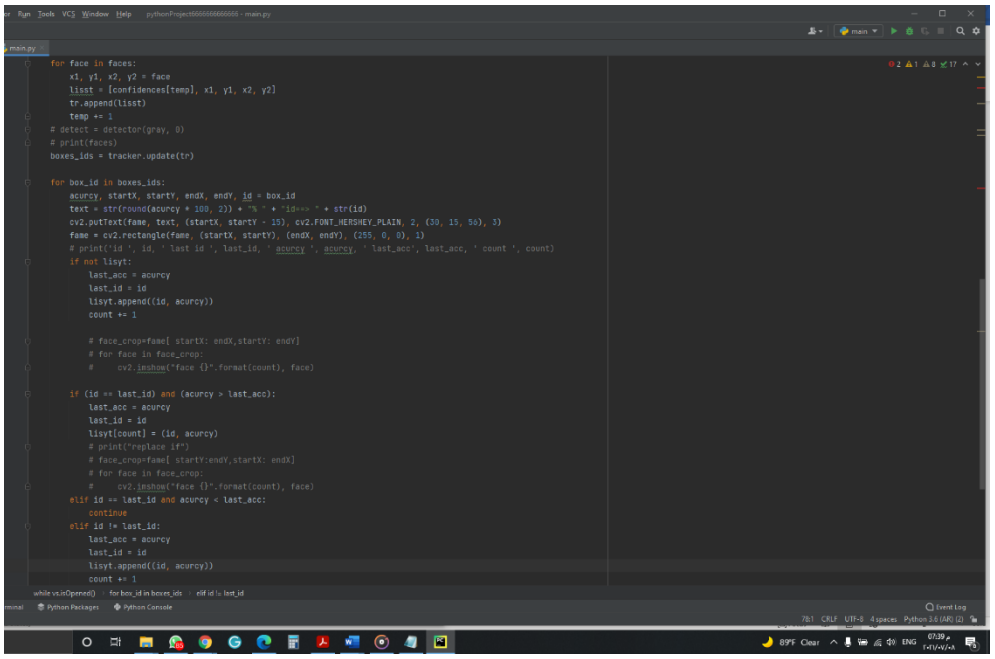
A simple, high level, easy-to-use open source Computer Vision library for Python.

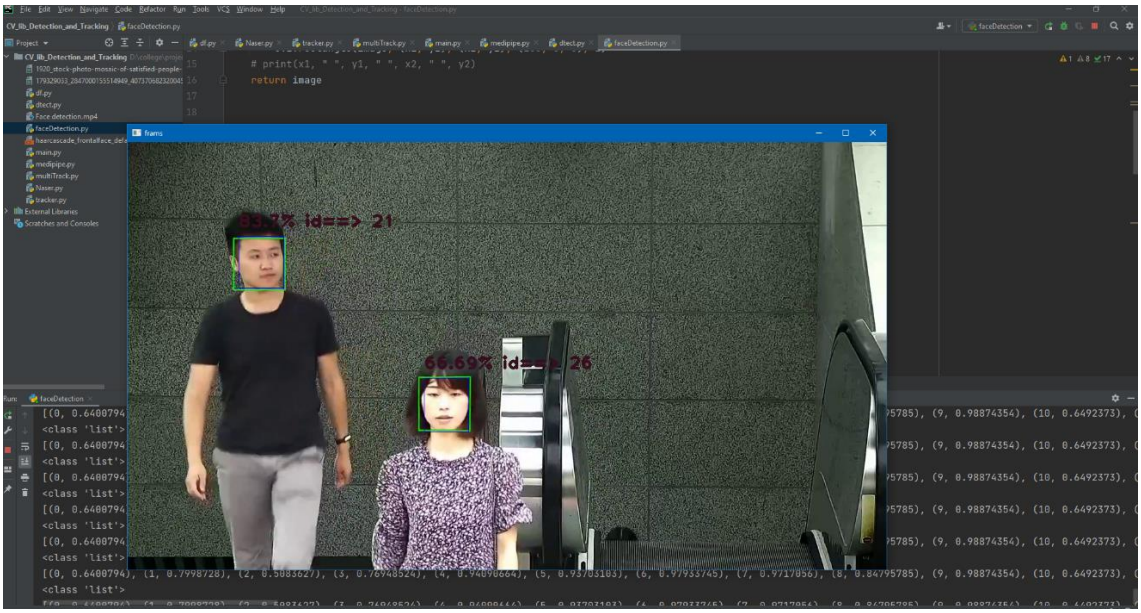
It was developed with a focus on enabling easy and fast experimentation. Being able to go from an idea to prototype with least amount of delay is key to doing good research.

To use it we should install

- OpenCV
- Tensorflow







Advantage:

- Simplicity
- user friendliness
- modularity and
- extensibility

4.2.2.3.2 Tracking

We have tried a lot of models of tracking models and the perfect one was algorithm with open cv and we will discuss all below:

1. TrackerCSRT
2. TrackerKCF
3. TrackerBoosting
4. TrackerMIL
5. TrackerTLD
6. TrackerMedianFlow
7. TrackerMOSSE
8. Opencv algorithm

1. TrackerCSRT

The Channel and Spatial Reliability Tracker CSRT, It's implementation is based on Discriminative Correlation Filter (DCF) with Channel and Spatial Reliability. It improves the DCF tracker by introducing spatial and channel reliability. The spatial reliability map is used to find out the optimal filter size. The ability to adjust filter size makes the CSRT tracker better than the traditional DCF algorithm by excluding unrealistic samples. Another benefit from the spatial reliability map is its ability to handle nonrectangular targets. The channel reliability is measured to weigh the importance of each channel filter, then combine them to get the final response map. Using only the HoGs and Colorname standard feature sets, the CSRT tracker can achieve an impressive accuracy with real-time speed.

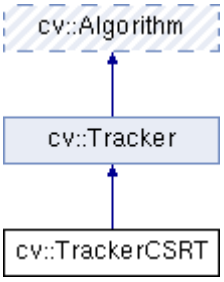
Advantage:

- detects target objects using the HoG features instead of raw pixels.
- It can adjust the size of target window dynamically as well.
- the CSRT achieves the best overall accuracy and is the most robust tracker on videos

Disadvantage:

- It still has a hard time recovering from a temporarily disappearing target or tracking a fast-moving
- It needs to detect the person on which the tracker will be tracked

Code: #include <opencv2/tracking/tracker.hpp>



2. TrackerKCF

the KCF (Kernelized Correlation Filter) tracker is a novel tracking framework that utilizes properties of circulant matrix to enhance the processing speed. This tracking method is an implementation of High-Speed Tracking with Kernelized Correlation Filters which is extended to KCF with color-names features. The KCF tracker emphasizes the importance of the negative samples and tends to use more samples for better training.

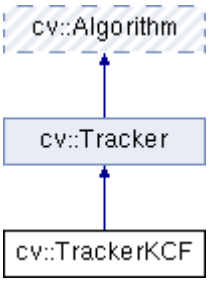
Advantage

- tracker performs well on ordinary videos

Disadvantage:

- It needs to detect the person on which the tracker will be tracked

Code: #include <opencv2/tracking/tracker.hpp>



3. TrackerBoosting

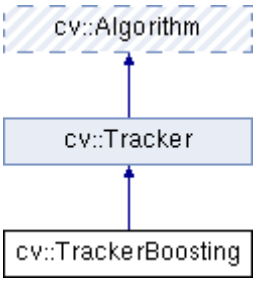
The Boosting tracker is a real-time object tracking based on a novel on-line version of the AdaBoost algorithm. The classifier uses the surrounding background as negative examples in update step to avoid the drifting problem. The online boosting algorithm maintains a global classifier pool of weak classifiers with multiple selectors. Each new training sample is used to update each weak classifier in the pool. A cascade system initializes the first selector with the current sample's importance, selects the best weak classifier with the least error, and passes the estimated importance to the next

selector until all selectors have been updated. In the end, a strong classifier is chosen from the best weak classifiers, and the worst weak classifier is replaced with a random one. The Boosting tracker utilizes the initial target area in the current frame as a positive example, and exploits other areas with the same size around the target as negative examples. Then, the online-trained classifier searches the neighborhood for potential targets in the next frame. The Boosting tracker can handle temporary occlusions as well as complex backgrounds.

Disadvantage:

- It needs to detect the person on which the tracker will be tracked

Code: #include <opencv2/tracking/tracker.hpp>



4. TrackerMIL

The Multiple Instance Learning (MIL) algorithm trains a classifier in an online manner to separate the object from the background. Multiple Instance Learning avoids the drift problem for a robust tracking. extends the online boosting algorithm by using a set of image patches (called a bag) instead of a single sample for training. A bag containing at least one positive example is called a positive bag, otherwise it is called a negative bag. The MIL tracker collects lots of small image patches centered at the tracking object as potential positive bags and chooses the best one to be the positive example. This strategy not only prevents the MIL tracker from losing important information but also avoids the mislabeling problem.

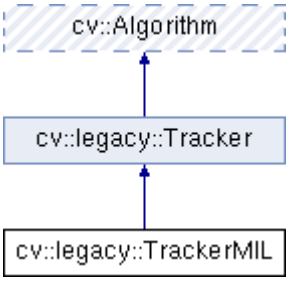
Advantage:

- The MIL tracker can properly handle most of the videos cases.

Disadvantage

- The problem of occlusion caused by change of viewpoints. The MIL tracker tends to fail in recovering the tracking target even after the occlusion.
- It needs to detect the person on which the tracker will be tracked

Code: #include <opencv2/tracking/tracking_legacy.hpp>



5. TrackerTLD

the TLD tracker is mainly composed of three parts: a tracker, a learner, and a detector. TLD is a novel tracking framework that explicitly decomposes the long-term tracking task into tracking, learning and detection. The tracker follows the object from frame to frame. The detector localizes all appearances that have been observed so far and corrects the tracker if necessary. The learning estimates detector's errors and updates it to avoid these errors in the future. The Median Flow algorithm was chosen as a tracking component in this implementation, following authors. The tracker is supposed to be able to handle rapid motions, partial occlusions, object absence etc. The TLD tracker is well known for its ability of failure recovery at the expense of instability. Compared to other online trackers struggling with the problem of accumulating errors, the combination of tracking and detecting modules makes the TLD tracker more reliable for long-term tracking.

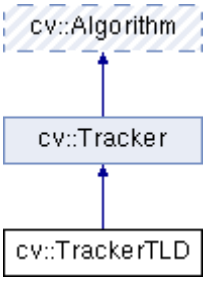
Advantage

- works well in the case of failure recovery.
- TLD tracker is a good choice to track a target that disappears from one place and reappears in another place in videos.

Disadvantage

- slow tracker with high false detect rate
- it is about 600 times slower
- It needs to detect the person on which the tracker will be tracked

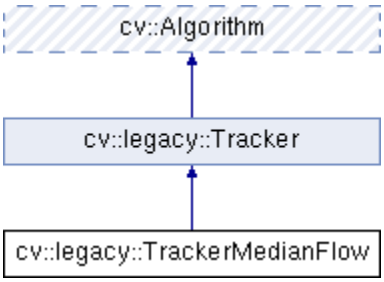
Code: #include <opencv2/tracking/tracker.hpp>



6. TrackerMedianFlow

The tracker is suitable for very smooth and predictable movements when object is visible throughout the whole sequence. It's quite and accurate for this type of problems (in particular, it was shown by authors to outperform MIL). is a bidirectional approach that combines forward and backward tracking. The forward and backward consistency is analyzed as a quality measure to assist the tracking. The MedianFlow tracker constructs both forward and backward trajectories at each time instant, and their corresponding errors are estimated. The trajectory with the minimum forward–backward error is chosen as the candidate for the succeeding tracking. As a result, the MedianFlow tracker is more reliable to follow objects with consistent movement.

Code: #include <opencv2/tracking/tracking_legacy.hpp>



7. TrackerMOSSE

the MOSSE (Minimum Output Sum of Squared Error) tracker. The implementation is based on Visual Object Tracking using Adaptive Correlation Filters. This tracker works with grayscale images, if passed bgr ones, they will get converted internally. It is a tracker based on correlation filters. It achieves high e_ciency by computing correlation in time domain. The MOSSE filter improves the ASEF filter to overcome the potential overfitting problem. The MOSSE tracker calculates the minimum output sum of square error to find out the most possible location of the tracking object. The benefits of using a correlation filter make the MOSSE tracker more robust to the problems of scaling, rotation, deformation, and occlusion compared to traditional approaches. Also, MOSSE is more flexible than other correlation-filter-based trackers because the target is not required to be in the center of the image in the beginning of tracking.

Advantage

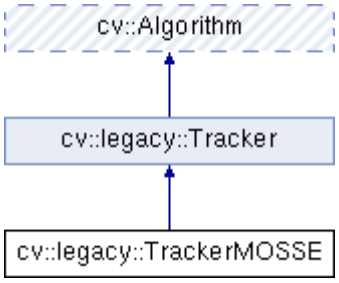
- For applications with high-speed demand or large motion, the MOSSE tracker is the best choice since its tracking speed is significantly higher than other trackers, though the fix-sized tracking window could be a problem for video sequences with huge scale change.
- Summerry the MOSSE is the most e_cient tracker in terms of speed.

- the MOSSE is the fastest tracker with an average of 3776 FPS.

Disadvantage

- It needs to detect the person on which the tracker will be tracked

Code: #include <opencv2/tracking/tracking_legacy.hpp>



Trackers Features	Principle Strength	Weakness	Improvement	Suggestion
TLD	Track, learn, and detect	Recovery from failure and occlusion	High false alarm	Combine with reliable filter
KCF	Kernelized correlation filter	Report tracking Failure	Fixed target size	Adaptable target size
BOOST	AdaBoost	Decent accuracy	Seldom report tracking failure	Adaptable tolerance
CSRT	Discriminative correlation filter	Robust and high accuracy	Long-term occlusion	Failure recovery with spatial relationship
MIL	Multi-instance learning	High accuracy	Long-term occlusion	Failure recovery with spatial relationship
MOSSE	Min square error	High tracking Speed	Fixed target size	Adaptable target size
MedianFlow	Min forward–backward error	Reliable on slow changing target	Fast-moving target	Support motion detection

8. Object Tracking with Opencv and Python

Object tracking does frame-by-frame tracking but keeps the history of where the object is at a time after time. We will talk first about object detection and then about how to apply object tracking to the detection.

Object tracking is the process of:

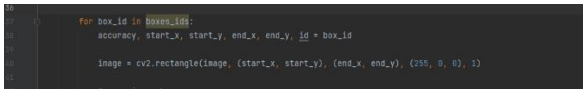
- Taking an initial set of object detections (such as an input set of bounding box coordinates)
- Creating a unique ID for each of the initial detections
- And then tracking each of the objects as they move around frames in a video, maintaining the assignment of unique IDs

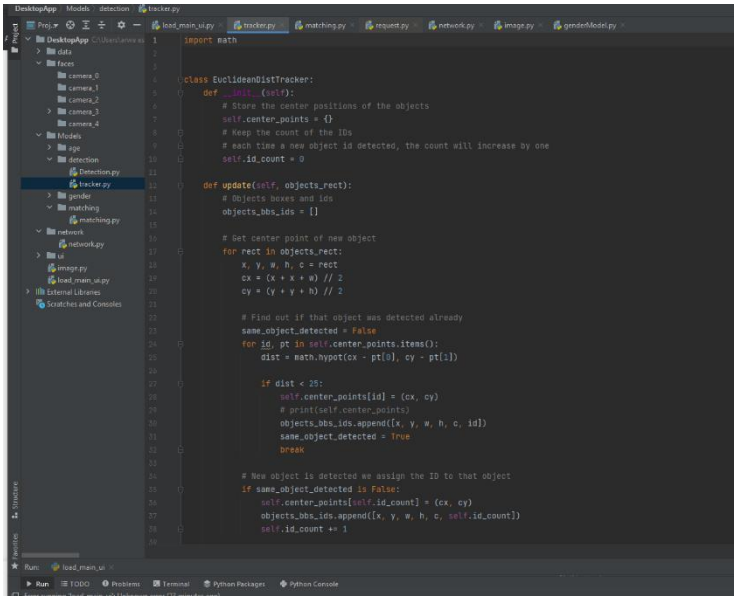
object tracking allows us to apply a unique ID to each tracked object, making it possible for us to count unique objects in a video. Object tracking is paramount to building a person counter (which we'll do later in this series).

How it works

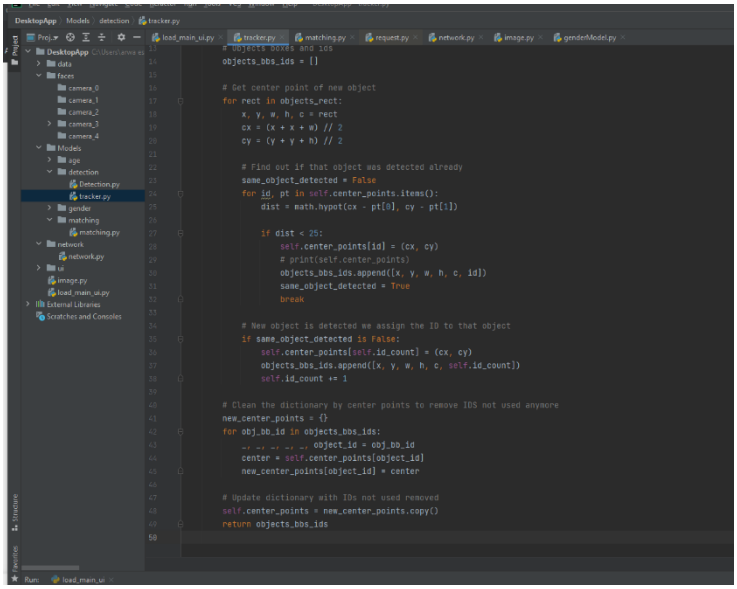
We will import and integrate the tracking functions. Then take position of the bounding box and insert them in a single array. This tracker takes all the bounding boxes of the objects so we need to save all the bounding boxes of the object into one array so it will be object detection, or we discard all the elements that we do not need, and we can easily do that by say detections it's an empty array each time we find the boxes we're going to put them inside detections.
Then pass our array with positions to tracker.update(). We will again get an array with the potions but in addition, a unique id will be assigned for each object.

As you can see from the code, we can analyze everything with a for a loop. At this point we just must draw the rectangle.
Code:





```
import math


class EuclideanDistTracker:
    def __init__(self):
        # Store the center positions of the objects
        self.center_points = {}
        # Keep the count of the IDs
        # each time a new object id detected, the count will increase by one
        self.id_count = 0

    def update(self, objects_rect):
        # Objects boxes and ids
        objects_bbs_ids = []

        # Get center point of new object
        for rect in objects_rect:
            x, y, w, h, c = rect
            cx = (x + x + w) // 2
            cy = (y + y + h) // 2

            # Find out if that object was detected already
            same_object_detected = False
            for id, pt in self.center_points.items():
                dist = math.hypot(cx - pt[0], cy - pt[1])

                if dist < 25:
                    self.center_points[id] = (cx, cy)
                    # print(self.center_points)
                    objects_bbs_ids.append([x, y, w, h, c, id])
                    same_object_detected = True
                    break

            # New object is detected we assign the ID to that object
            if same_object_detected is False:
                self.center_points[self.id_count] = (cx, cy)
                objects_bbs_ids.append([x, y, w, h, c, self.id_count])
                self.id_count += 1
```



```
        # Objects boxes and ids
        objects_bbs_ids = []

        # Get center point of new object
        for rect in objects_rect:
            x, y, w, h, c = rect
            cx = (x + x + w) // 2
            cy = (y + y + h) // 2

            # Find out if that object was detected already
            same_object_detected = False
            for id, pt in self.center_points.items():
                dist = math.hypot(cx - pt[0], cy - pt[1])

                if dist < 25:
                    self.center_points[id] = (cx, cy)
                    # print(self.center_points)
                    objects_bbs_ids.append([x, y, w, h, c, id])
                    same_object_detected = True
                    break

            # New object is detected we assign the ID to that object
            if same_object_detected is False:
                self.center_points[self.id_count] = (cx, cy)
                objects_bbs_ids.append([x, y, w, h, c, self.id_count])
                self.id_count += 1

        # Clean the dictionary by center points to remove IDS not used anymore
        new_center_points = {}
        for obj_bb_id in objects_bbs_ids:
            _, _, _, _, _, object_id = obj_bb_id
            center = self.center_points[object_id]
            new_center_points[object_id] = center

        # Update dictionary with IDs not used removed
        self.center_points = new_center_points.copy()
        return objects_bbs_ids
```

4.2.2.3.3 Age and gender Models

1. **Age Model**

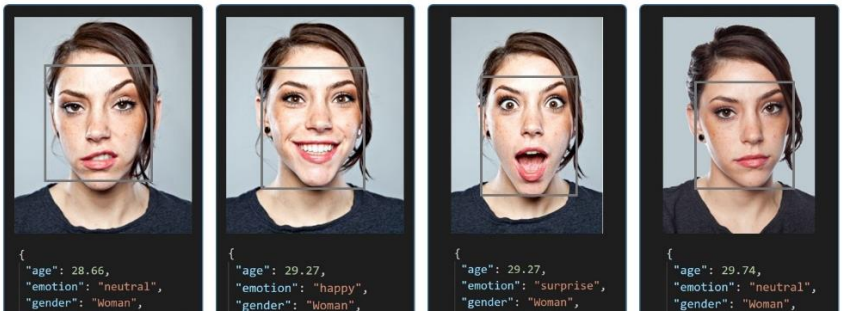
We used two models: deep face attribute, VGG net Caffe model(the one we use now) .

1. Deep face

So, we firstly use deep face attribute analysis in age model

We need test images for face detection. To keep things simple, we will use two test images: one with two faces, and one with many faces. We're not trying to push the limits of face detection, just demonstrate how to perform face detection with normal front-on photographs of people.

Running photos, we can see that many of the faces were detected correctly, but the result is not perfect.

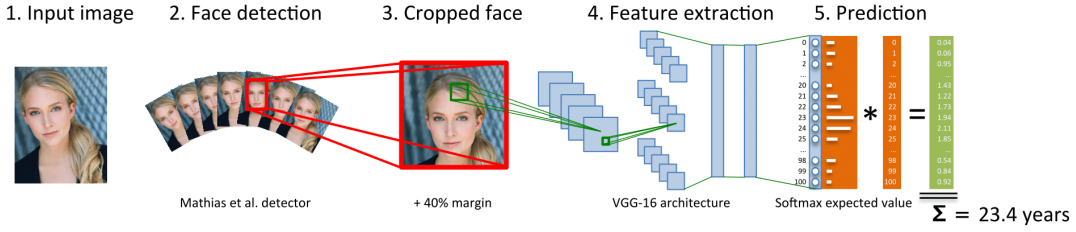


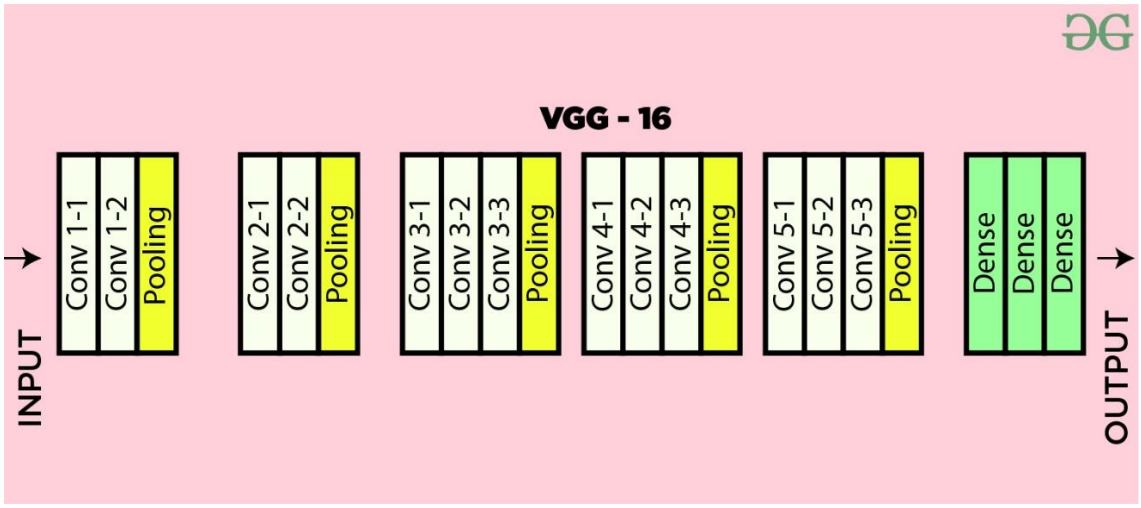
Problem of deep face:

- It The face must be recognized in advance, and this caused some problems because we give him the picture of the specific face and not the whole picture
- There are models now provide better accuracy up to 99.6% on.
- It is being implemented by haar cascade classifer

2. VGG net Caffe model.

we're going to use the image of a person then we're going to detect a face on this image then we will add some margin to the face and then we pass this a face with margin into the convolutional neural network to receive a prediction there will be two kinds of prediction age and gender and we will use VGG-16 architecture and we will take pre-trained models from this paper.

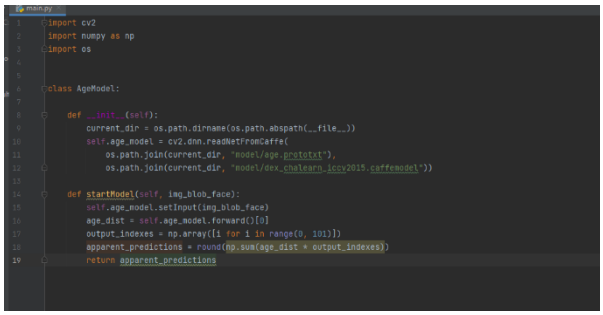




we tackle the estimation of apparent age in still face images with deep learning. Here the convolutional neural networks (CNNs) use the VGG-16 architecture and are pretrained on ImageNet for image classification.

In addition, due to the limited number of apparent age annotated images, we explore the benefit of finetuning over crawled Internet face images with available age. We crawled 0.5 million images of celebrities from IMDb and Wikipedia that we make public on this website.

This is the largest public dataset for age prediction to date. We pose the age regression problem as a deep classification problem followed by a softmax expected value refinement and show improvements over direct regression training of CNNs. Our proposed method, Deep EXpectation (DEX) of apparent age, first detects the face in the test image and then extracts the CNN predictions from an ensemble of 20 networks on the cropped face.



2. Gender Model

Gender classification can be considered as a binary class classification problem. We have used two models: deep face attribute, VGG net Caffe model (currently using).

1. Deep face

So, we firstly use deep face attribute analysis in age model

We need test images for face detection. To keep things simple, we will use two test images: one with two faces, and one with many faces. We're not trying to push the limits of face detection, just demonstrate how to perform face detection with normal front-on photographs of people.

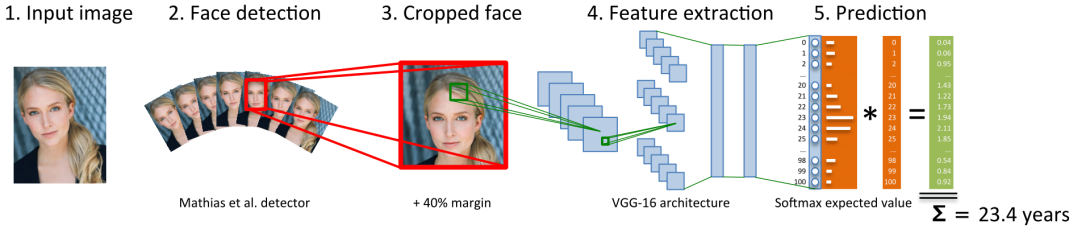
First, faces in the given images are detected and cropped out. Second, the facial embedding for each face are then calculated by passing it through a neural network. Third, treating these embeddings as feature vectors, these vectors are passed through various Machine Learning Models to predict gender.

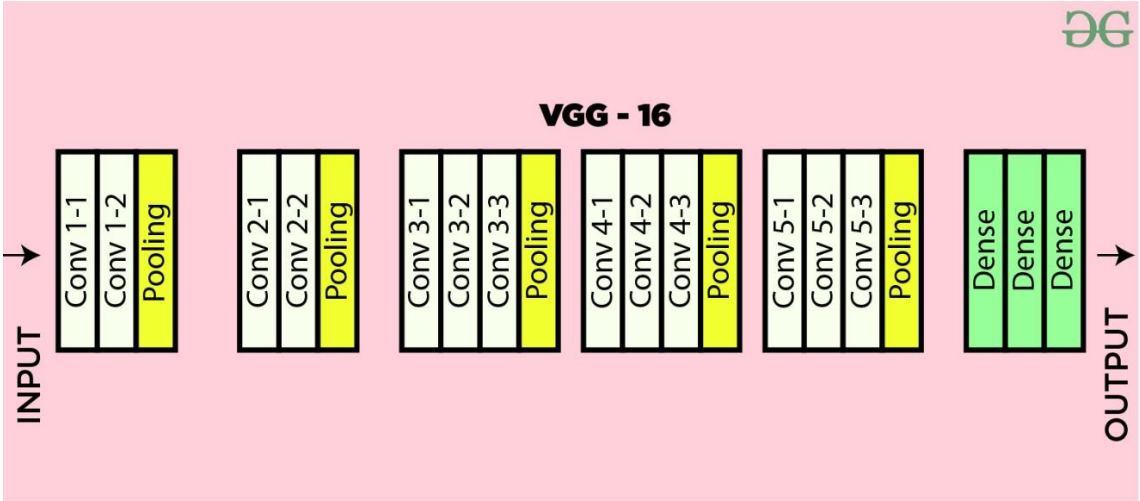
Problem of deep face:

- It The face must be recognized in advance, and this caused some problems because we give him the picture of the specific face and not the whole picture
- There are models now provide better accuracy up to 99.6% on.
- It is being implemented by haar cascade classifer

2. VGG net Caffe model.

we're going to use the image of a person then we're going to detect a face on this image then we will add some margin to the face and then we pass this a face with margin into the convolutional neural network to receive a prediction there will be two kinds of prediction age and gender and we will use VGG-16 architecture and we will take pre-trained models from this paper.

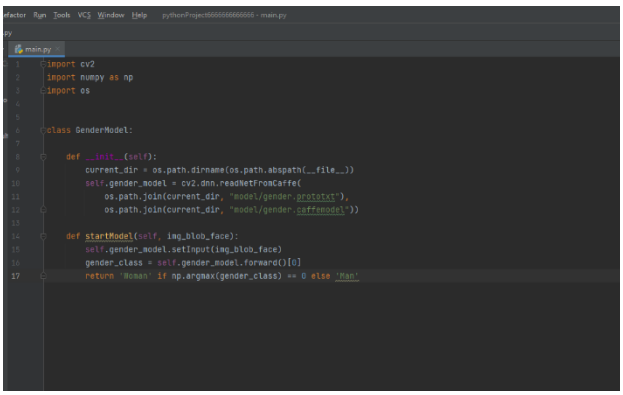




we tackle the estimation of apparent age in still face images with deep learning. Here the convolutional neural networks (CNNs) use the VGG-16 architecture and are pretrained on ImageNet for image classification.

In addition, due to the limited number of apparent age annotated images, we explore the benefit of finetuning over crawled Internet face images with available age. We crawled 0.5 million images of celebrities from IMDb and Wikipedia that we make public on this website.

This is the largest public dataset for age prediction to date. We pose the age regression problem as a deep classification problem followed by a softmax expected value refinement and show improvements over direct regression training of CNNs. Our proposed method, Deep EXpectation (DEX) of apparent age, first detects the face in the test image and then extracts the CNN predictions from an ensemble of 20 networks on the cropped face.



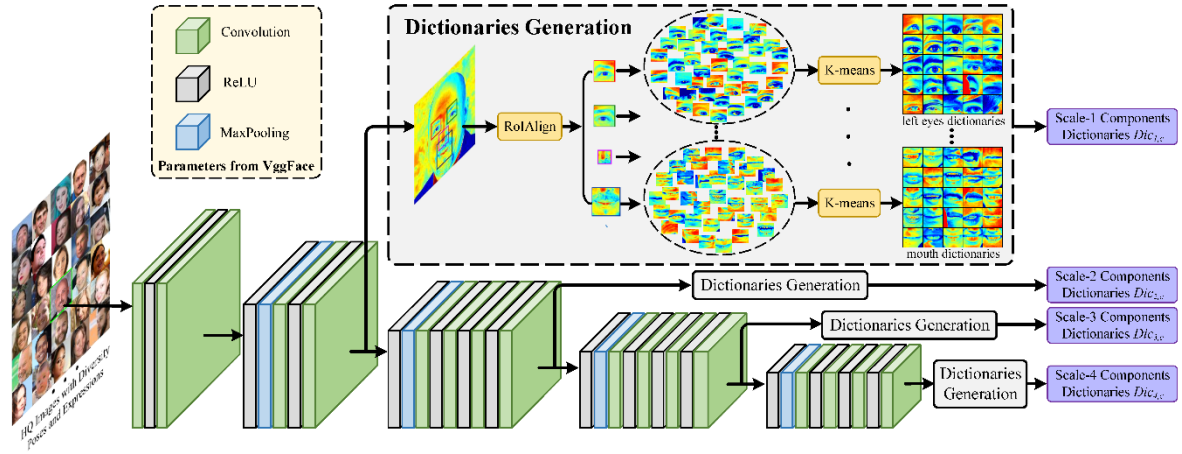
4.2.2.3.4 Enhancement

This model is responsible for enhancing the image it takes a low-quality image and returns a high-quality image so:

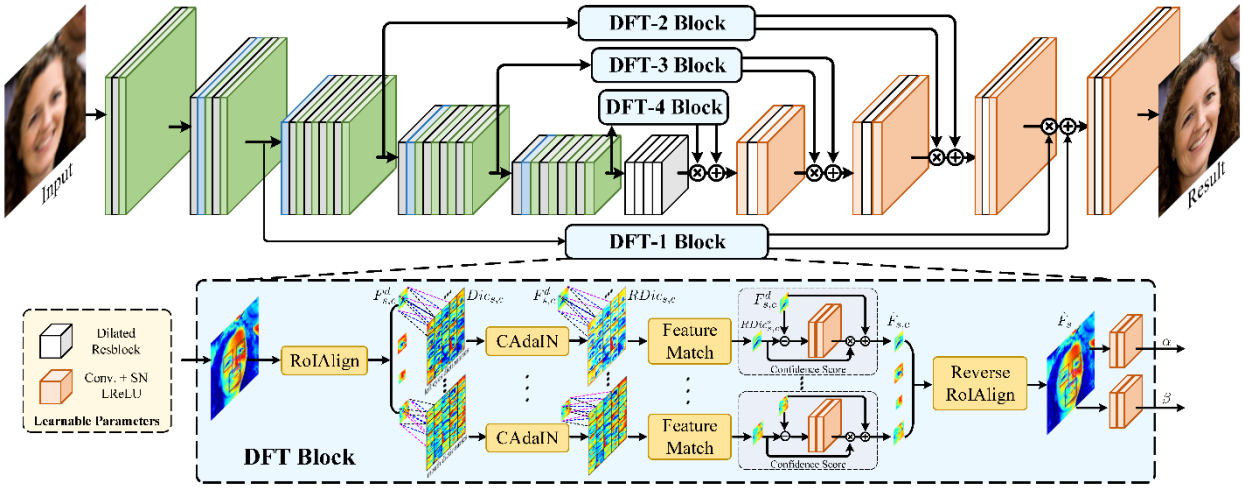
- 1 low-quality mage for input
- Output an enhanced Image

This model it mainly contains two parts:

1. the off-line generation of multi-scale component dictionaries from large amounts of high-quality images, which have diverse poses and expressions. K-means is adopted to generate K clusters for each component (i.e., left/right eyes, nose and mouth) on different feature scales.



2. The restoration process and dictionary feature transfer (DFT) block that are utilized to provide the reference details in a progressive manner. Here, DFT-i block takes the Scale-i component dictionaries for reference in the same



Blind face restoration aims at recovering realistic details from real low-quality (LQ) image to its high-quality (HQ) one, without knowing the degradation types or parameters. Compared with single image restoration tasks, e.g., image super-resolution, denoising, and deblurrin, blind image restoration suffers from more challenges, yet is of great practical value in restoring real LQ images.

We use deep component dictionaries as reference candidates to guide the degraded face restoration. The proposed DFDNet can generalize to face images without requiring the identity-belonging HQ reference, which is more applicative and efficient than those reference-based methods.

We suggest a DFT block by utilizing CAdaIN to eliminate the distribution diversity between the input and dictionary clusters for better dictionary feature transfer, and we also propose a confidence score to adaptively fuse the dictionary feature to the input with different degradation level.

We adopt a progressive manner for training DFDNet by incorporating the component dictionaries in different feature scales. This can make our DFDNet learn coarse-to-fine details.

Our proposed DFDNet can achieve promising performance on both synthetic and real degraded images, showing its potential in real applications.

Related Work

1. Single Image Restoration

   Besides the benefits brought by deep CNNs, single image recovery has been very successful in many tasks, for example, super image resolution, de-aliasing, de-aliasing and compression. Because of the specific facial structure, there are also several well-developed methods of facial hallucination. Among these methods, Huang et al. We propose to solve a very low resolution face image using neural networks to predict the wavelet coefficients of HQ images.

2. Reference-Based Image Restoration

   Due to the limitation of single image restoration methods on real-world LQ images, there are some works that use an additional image to guide the restoration process, which can bring the object structure details to the final result. As for natural image restoration, Zhang et al. utilize a reference image which has similar content with a LR image and then adopt a global matching scheme to search the similar content patches. These reference feature patches are then used to swap the texture feature of LR images. This method can achieve

great visual improvements. However, it is very time and memory consuming in searching similar patches from the global content.

3. Proposed Method

Inspired by the former reference-based image restoration methods, this work attempts to overcome the limitation of requiring reference image in face restoration. Given a LQ image Id, our proposed DFDNet aims to generate plausible and realistic HQ one ˆIh with the conducted component dictionaries. or each component of the degraded observation Id, our DFDNet selects the dictionary features that have the most similar structure with the input. Specially, we re-norm the whole dictionaries via component AdaIN(termed as CAdaIN) based on the input component to eliminate the distribution or style diversity. The selected dictionary features are then utilized to guide the restoration process via dictionary feature transformation. Furthermore, we introduce a confidence score on the selected dictionary feature to generalize different degradation levels through weighted feature fusion. The progressive manner from coarse to fine is also beneficial to the restoration process. In the following, we first describe the off-line generation of multi-scale deep component dictionaries. Then the details of our proposed DFDNet along with the dictionaries feature transfer (DFT) blocks are interpreted. The objective functions for training are finally presented.

- Off-line Generation of Component Dictionaries

  To build the deep component dictionaries that cover the most types of faces, we adopt FFHQ dataset due to its high-quality and considerable variation in terms of age, ethnicity, pose, expression, etc. We utilize Deep Pose and Face++1to recognize their poses and expressions (i.e., anger, disgust, fear, happiness, neutral, sadness and surprise), respectively, to balance the distribution of each attribute. Among these 70,000 high-quality images of FFHQ, we select 10,000 ones to build our dictionaries. Given a high-quality image Ih, we first use pre-trained VggFace to extract its features on different scales. With the facial landmarks Lh detected by dlib, we utilize RoIAlign to crop and re-sample these four components on each scale to a fixed size. We then adopt K-means to generate K clusters for each component,

$$Dic_{s,c} = \mathcal{F}_{Dic}\left(I^h | L^h; \Theta_{Vgg}\right),$$

resulting in our component dictionaries. In particular, for handling 256×256 images, the feature sizes of left/right eyes, nose and mouth on scale-1 are set to 40/40, 25, 55, respectively. The sizes are down-sampled one by one by two times for the following scale. These dictionary feature can be formulated as:

- Deep Face Dictionary Network

  After building the high-quality component dictionaries, our DFDNet is then proposed to transfer the dictionary features to the degraded input Id. The Blind Face Restoration via Deep Multi-Scale Component Dictionaries proposed DFDNet can be formulated as:

$$\hat{I} = \mathcal{F}(I^d | L^d, Dic; \Theta),$$

  Where Ld and Dic represent the facial landmarks of Id and the component dictionaries in Eqn. 1, respectively. Θ denotes the learnable parameters of DFDNet. To guarantee the features of Id and Dic in the same feature space, we take the pre-trained VggFace model as the encoder of DFDNet, which has the same network architecture and parameters in the dictionary generation network. Suppose that the encoder of DFDNet is different from VggFace or trainable in the training phase, it easily generates different features which are in consistent with the pre-conducted dictionaries. For better transferring the dictionary feature to the input components, we suggest a DFT block and use it in a progressive manner.

4. Experiments

Since the performance of reference-based methods are usually superior to other single image or face restoration methods, in this paper, we mainly compare our DFDNet with reference-based (i.e., GFRNet, GWAINet) and face prior-based methods (i.e., Shen et al., Kim et al.). We also report the results of single natural image (i.e., RCAN, ESRGAN) and face (i.e., WaveletSR) super-resolution methods. Among these methods, Shen et al. and Kim et al. can only handle 128×128 images, while others can restore256×256 images. For fair comparisons, our DFDNet is trained on

these two sizes (termed as DFDNet128 and DFDNet256). RCAN and ESRGAN were originally trained on the natural images, thus we retrain them using our training data for further fair comparison (termed as *RCAN and *ESRGAN). WaveletSR was also retrained by using our training data with their released training code (termed as *WaveletSR). Following, PSNR, SSIM and LPIPS are reported on the super-resolution task (×4 and×8) which also has the random injection of Gaussian noise and blur operation for quantitatively evaluating on the blind restoration task. In terms of qualitative comparison, we demonstrate the comparisons on the synthetic and real-world low-quality images. More visual results including high resolution restoration performance (i.e., 512×512) can be found in our supplemental materials.

- Training Details

  As mentioned in Section 3.1, we select 10,000 images from FFHQ to build our component dictionaries. We note that GFRNet, GWAINet and WaveletSR adopt VggFace2 as their training data, we also use it for training and validating our DFDNet for fair comparison. To evaluate the generality of our method, we build two test datasets, i.e., 2,000 test images from VggFace2 which are not overlapped with the training data, and another 2,000 images from CelebA. Each of them has a high-quality reference from the same identity for running GFRNet and GWAINet. To synthesize the training data that approximate to the real LQ images, we adopt the same degradation model suggested in GFRNet.

$$I^d = \left((I^h \otimes \mathbf{k})_{\downarrow_r} + \mathbf{n}_\sigma\right)_{JPEG_q}$$

  Where k denotes two common types of blur kernel,i.e., Gaussian blur with %∈ {1 : 0.1 : 5} and 32 motion blur kernels from. Down-sampler r, Gaussian noise nσ and JPEG compression quality q are randomly sampled from {1 : 0.1 : 8},{0 : 1 : 15}and{40 : 1 : 80}, respectively. The trade-off parameters for training DFDNet are set as follows: λl2= 100,λp,1= 0.5,λp,2= 1,λp,3= 2,λp,4= 4,λa,1= 4,λa,2= 2,λa,4= 1,λa,8= 1. The Adam optimizer [21] is adopted to train our DFDNet with learning rate lr= 2×10−4, β1= 0.5 andβ2= 0.999. lr is reduced by 2 times when the

reconstruction loss on validation set becomes non-decreasing. The whole model including the generation of multi-scale component dictionaries and the training of DFDNet are executed on a server with 128G RAM and 4 Tesla V100. It takes 4 days to train our DFDNet.

- Results on Synthetic Images

  Qualitative evaluation. The quantitative results of these competing methods on super-resolution task are shown in Table 1. We can have the following observations:1) Compared with all the competing methods, our DFDNet is superior to others by a large margin on two datasets and two super-resolution.

Table 1: Quantitative comparisons on two datasets and two tasks (×4 and ×8).

Methods	VggFace2 [3]						CelebA [27]					
	×4			×8			×4			×8		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Shen *et al.* [33]	20.56	.745	.080	18.79	.717	.126	21.04	.751	.079	18.64	.714	.131
Kim *et al.* [6]	-	-	-	20.99	.759	.095	-	-	-	20.72	.749	.104
DFDNet128	25.76	.893	.035	23.42	.841	.071	25.92	.899	.031	23.40	.839	.080
RCAN [46]	24.87	.889	.283	21.36	.819	.295	24.93	.892	.267	21.11	.814	.302
*RCAN	25.32	.896	.247	22.94	.836	.271	25.47	.901	.217	22.84	.831	.283
ESRGAN [36]	24.13	.876	.223	-	-	-	24.31	.878	.210	-	-	-
*ESRGAN	24.91	.891	.194	-	-	-	25.04	.896	.193	-	-	-
WaveletSR [15]	24.30	.878	.236	21.70	.823	.273	24.51	.884	.247	21.42	.820	.279
GFRNet [26]	27.13	.912	.132	23.37	.856	.269	27.32	.915	.124	23.12	.852	.273
GWAINet [7]	-	-	-	23.41	.860	.260	-	-	-	23.38	.859	.270
DFDNet256	**27.54**	**.923**	**.114**	**23.73**	**.872**	**.239**	**27.77**	**.925**	**.103**	**23.69**	**.872**	**.241**

Figure 2: Visual comparisons of these competing methods on ×4 SR task. Close-up in the right bottom of GFRNet is the required guidance.

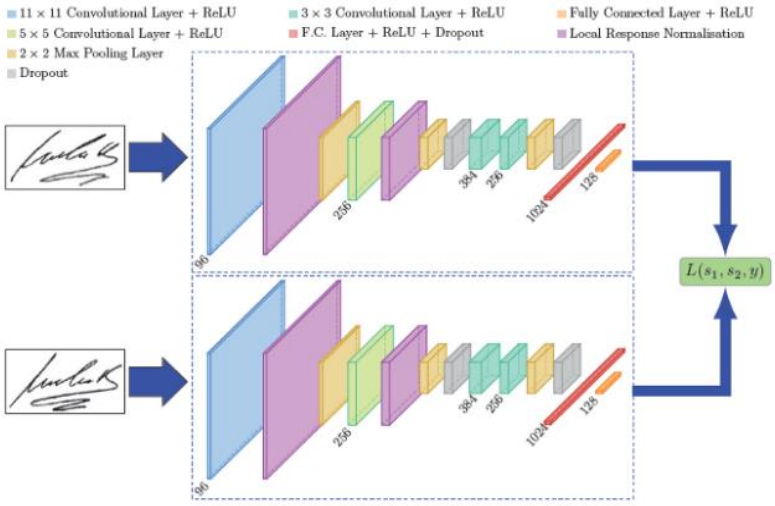
4.2.2.3.5 Matching

we have same requrments to get the match faces in the project

- 1 photo to every id (person)
- 1 photo to lost person

We have 1 photo to get matched in same list of ids , we can do this by using Architecture is called siamese network, siamese network is is an artificial neural network that uses the same weights while working in tandem on two different input vectors to compute comparable output vectors. Often one of the output vectors is precomputed, thus forming a baseline against which the other output vector is compared. This is similar to comparing fingerprints but can be described more technically as a distance function for locality-sensitive hashing.

siamese network sometimes called a **twin neural network.**



Siamese network used in Signet

A Siamese Neural Network is a class of neural network architectures that contain two or more identical subnetworks. 'identical' here means, they have the same configuration with the same parameters and weights. Parameter updating is mirrored across both sub-networks. It is used to find the similarity of the inputs by comparing its feature vectors, so these networks are used in many applications

Traditionally, a neural network learns to predict multiple classes. This poses a problem when we need to add/remove new classes to the data. In this case, we have to update the neural network and retrain it on the whole dataset. Also, deep neural networks need a large volume of data to train on. SNNs, on the other hand, learn a similarity function. Thus, we can train it to see if the two images are the same (which we will do here). This enables us to classify new classes of data without training the network again.

The main advantages of Siamese Networks are,

- **More Robust to class Imbalance:**

  With the aid of One-shot learning, given a few images per class is sufficient for Siamese Networks to recognize those images in the future.

- **Nice to an ensemble with the best classifier:**

  Given that its learning mechanism is somewhat different from Classification, simple averaging of it with a Classifier can do much better than average 2 correlated Supervised models (e.g. GBM & RF classifier)

- **Learning from Semantic Similarity:**

Siamese focuses on learning embeddings (in the deeper layer) that place the same classes/concepts close together. Hence, can learn semantic similarity.

The downsides of the Siamese Networks can be,

- **Needs more training time than normal networks:** Since Siamese Networks involves quadratic pairs to learn from (to see all information available) it is slower than normal classification type of learning(pointwise learning)
- **Doesn't output probabilities:** Since training involves pairwise learning, it won't output the probabilities of the prediction, but the distance from each class

Loss functions used in Siamese Networks:

Since training of Siamese networks involves pairwise learning usual, Cross entropy loss cannot be used in this case, mainly two loss functions are mainly used in training these Siamese networks, they are

Triplet loss is a loss function where a baseline (anchor) input is compared to a positive (truthy) input and a negative (falsy) input. The distance from the baseline (anchor) input to the positive (truthy) input is minimized, and the distance from the baseline (anchor) input to the negative (falsy) input is maximized.

$$\mathcal{L}\left(A, P, N\right) = \max\left(\| \mathrm{f}(A) - \mathrm{f}(P)\|^2 - \| \mathrm{f}(A) - \mathrm{f}(N)\|^2 + \alpha, 0\right)$$

In the above equation, alpha is a margin term used to "stretch" the distance differences between similar and dissimilar pairs in the triplet, fa, fa, fn are the feature embeddings for the anchor, positive and negative images.

During the training process, an image triplet (anchor image, negative image, positive image)(anchor image, negative image, positive image) is fed into the model as a single sample. The idea behind this is that distance between the anchor and positive images should be smaller than that between the anchor and negative images.

Contrastive Loss: is a popular loss function used highly nowadays, It is a distance-based loss as opposed to more conventional error-prediction losses. This loss is used to learn

embeddings in which two similar points have a low Euclidean distance and two dissimilar points have a large Euclidean distance.

And we defined Dw which is just the Euclidean distance as :

$$(1-Y)\frac{1}{2}(D_W)^2 + (Y)\frac{1}{2}\{max(0, m - D_W)\}^2$$

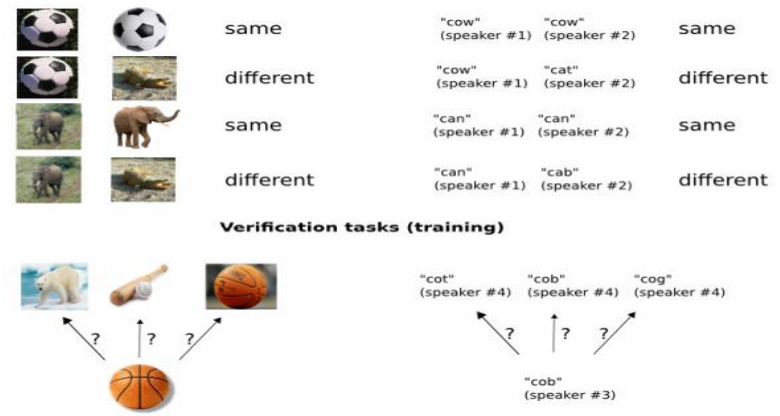
And we defined Dw which is just the Euclidean distance as :

Gw is the output of our network for one image.

$$\sqrt{\{G_W(X_1) - G_W(X_2)\}^2}$$

One-shot Image Recognition

The process of learning good features for machine learning applications can be very computationally expensive and may prove difficult in cases where little data is available. A prototypical example of this is the one-shot learning setting, in which we must correctly make predictions given only a single example of each new class. In this paper, we explore a method for learning siamese neural networks which employ a unique structure to naturally rank similarity between inputs. Once a network has been tuned, we can then capitalize on powerful discriminative features to generalize the predictive power of the network not just to new data, but to entirely new classes from unknown distributions. Using a convolutional architecture, we are able to achieve strong results which exceed those of other deep learning models with near state-of-the-art performance on one-shot classification tasks.



One shot task (test)

One-shot learning can be directly addressed by developing domain-specific features or inference procedures which possess highly discriminative properties for the target task. As a result, systems which incorporate these methods tend to excel at similar instances but fail to offer robust solutions that may be applied to other types of problems. In this paper, we present a novel approach which limits assumptions on the structure of the inputs while automatically acquiring features which enable the model to generalize successfully from few examples. We build upon the deep learning framework, which uses many layers of non-linearities to capture invariances to transformation in the input space, usually by leveraging a model with many parameters and then using a large amount of data to prevent overfitting (Bengio, 2009; Hinton et al., 2006). These features are very powerful because we are able to learn them without imposing strong priors, although the cost of the learning algorithm itself may be considerable.

Structural Similarity

SSIM is used as a metric to measure the similarity between two given images. As this technique has been around since 2004, a lot of material exists explaining the theory behind SSIM but very few resources go deep into the details, that too specifically for a gradient-based implementation as SSIM is often used as a loss function.
IM was first introduced in the 2004 IEEE paper, “Image Quality Assessment: From Error Visibility to Structural Similarity”. The abstract provides a good intuition into the idea behind the system proposed.

Objective methods for assessing perceptual image quality traditionally attempted to quantify the visibility of errors (differences) between a distorted image and a reference image using a variety of known properties of the human visual system. Under the assumption that human visual perception is highly adapted for extracting structural information from a scene, we introduce an alternative complementary framework for quality assessment based on the degradation of structural information**.**

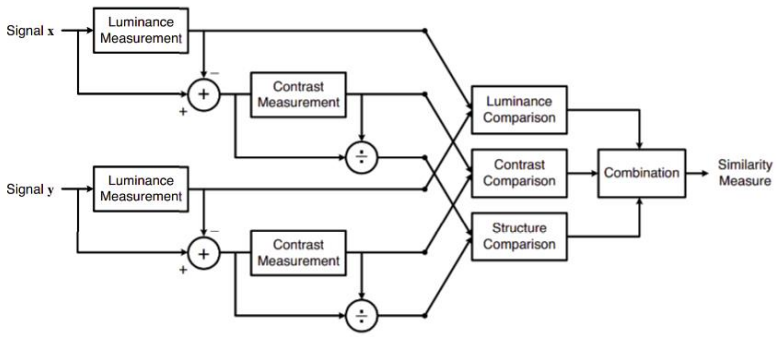
We have 2 essential points

- Most Image quality assessment techniques rely on quantifying errors between a reference and a sample image. A common metric is to quantify the difference in the values of each of the corresponding pixels between the sample and the reference images (By using, for example, Mean Squared Error).
- The Human visual perception system is highly capable of identifying structural information from a scene and hence identifying the differences between the information extracted from a reference and a sample scene. Hence, a metric that replicates this behavior will perform better on tasks that involve differentiating between a sample and a reference image.

The Structural Similarity Index (SSIM) metric extracts 3 key features from an image:

- Luminance
- Contrast
- Structure

The comparison between the two images is performed on the basis of these 3 features. given below shows the arrangement and flow of the Structural Similarity Measurement system. Signal X and Signal Y refer to the Reference and Sample Images.



But what does this metric calculate?

This system calculates the *Structural Similarity Index* between 2 given images which is a value between -1 and +1. *A value of +1* indicates that the 2 given images are **very similar or the same** while a *value of -1* indicates the 2 given images are **very different.** Often these values are adjusted to be in the range [0, 1], where the extremes hold the same meaning.

Now, let's explore briefly, how these features are represented mathematically, and how they contribute to the final SSIM score.

- **Luminance:** Luminance is measured by *averaging* over all the pixel values. Its denoted by μ (Mu) and the formula is given below.

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i. \qquad (2)$$

The luminance comparison function $l(\mathbf{x}, \mathbf{y})$ is then a function of μ_x and μ_y.

- **Structure:** The structural comparison is done by using a consolidated formula (more on that later) but in essence, we divide the input signal with its standard deviation so that the result has unit standard deviation which allows for a more robust comparison.

$$(\mathbf{x} - \mu_x)/\sigma_x$$

where x is the Input Image

- **Contrast**: It is measured by taking the standard deviation (square root of variance) of all the pixel values. It is denoted by σ (sigma) and represented by the formula below.

$$\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{\frac{1}{2}}. \qquad (4)$$

The contrast comparison $c(\mathbf{x}, \mathbf{y})$ is then the comparison of σ_x and σ_y.

So now we have established the mathematical intuition behind the three parameters. But hold on! We are not yet done with the math, a little bit more. What we lack now, are comparison functions that can compare the two given images on these parameters, and finally, a combination function that combines them all. Here, we define the comparison functions and finally the combination function that yields the similarity index value.

- **Luminance comparison function:** It is defined by a function, l(x, y) which is shown below. μ (mu) represents the mean of a given image. x and y are the two images being compared.

$$l(\mathbf{x},\mathbf{y}) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$$

- **Contrast comparison function:** It is defined by a function c(x, y) which is shown below. σ denotes the standard deviation of a given image. x and y are the two images being compared.

$$c(\mathbf{x},\mathbf{y}) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

- **Structure comparison function:** It is defined by the function s(x, y) which is shown below. σ denotes the standard deviation of a given image. x and y are the two images being compared.

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y).$$

And finally, the SSIM score is given by,

$$\mathrm{SSIM}(\mathbf{x},\mathbf{y}) = [l(\mathbf{x},\mathbf{y})]^{\alpha} \cdot [c(\mathbf{x},\mathbf{y})]^{\beta} \cdot [s(\mathbf{x},\mathbf{y})]^{\gamma}$$

But there's a plot twist!

For image quality assessment, it is useful to apply the SSIM index locally rather than globally. First, image statistical features are usually highly spatially nonstationary. Second, image distortions, which may or may not depend on the local image statistics, may also be space-variant. Third, at typical viewing distances, only a local area in the image can be perceived with high resolution by the human observer at one time instance (because of the foveation feature of the HVS [49], [50]). And finally, localized quality measurement can provide a spatially varying quality map of the image, which delivers more information about the quality degradation of the image and may be useful in some applications.

Summary:

Instead of applying the above metrics globally (i.e. all over the image at once) it's better to apply the metrics regionally (i.e. in small sections of the image and taking the mean overall).

This method is often referred to as the Mean Structural Similarity Index.

Due to this change in approach, our formulas also deserve modifications to reflect the same (it should be noted that this approach is more common and will be used to explain the code).

(**Note:** If the content below seems a bit overwhelming, no worries! If you get the gist of it, then going through the code will give you a much clearer idea.)

We have applied Similarity on photos.

0.40321971548712393

0.5640013065277749

0.944336993065277749

4.2.2.4 limitations

Let's talk now about things we tried to make in our project but we failed in some point so we decided to stop their until we learn something new.

1. For the backend part we tried to use microservice but we found that to make that we have to use something called load balancer and also a massege broker like RabbitMQ so we decided to build a normal Monolithic API until we learn the event-driven architecture with microservices architecture to use them perfectly together.
2. We use the free tier of MongoDB because MongoDB prices are high so we decided to stay on the free tier and also we decided to empty the camera collection every 24 hour so there is no overload happen cause we use a free tier plan and we have a lot of limitations on it.

3. We tried to use the face enhancement model in our project but we discovered that it will slow down the system too much so we decided to eliminate it until we deploy the models.
4. We couldn't deploy our models because of the high prices of AWS.

4.2.2.5 Future Works

Our future works are related to our limitations. We will use in the future the microservices architecture with the event-driven architecture to improke the API scalability and performance.

We will buy a MongoDB plan with a good space and memory and processing units to handle our data faster that it does now.

We will build our enhancement model and connect it with the other models and deploying them on AWS EC2

4.2.2.6 Conclusion Summary

This project is an application to facilitate finding lost children within a specified place(malls, hybermarkets, etc..) it is a combination of many different technlogies that serve each other to provide an easy to use, portable application for any kind of users by following a few steps :

1. cameras at the entrance take a high-quality image of the child's face
2. The image of the face is stored in database
3. if the child get lost, The person responsible for the child will upload an image, age, gender, and the expected lost date
4. Cameras at the exit prevent child from going outside the mall if he tried, then : Cameras of the mall start to search for the child by focusing on faces of the childrens by the help of age recognition algorithms, then a group of images that have been taken depending on the similarity of faces, range of age, and gender is processed and filtered using matching algorithms. if the filtered image matches the uploaded image, a notification is sent to the application saying that the child is found and where he is.

5. If the cameras cannot find the child inside the mall then the cameras at the exit will record the checkout date if the child left the mall. At the end, the purpose of the system is to make the process of finding lost children easier and faster as it happens without any human intervention and will dicrease the possibility of losing your child which is the main problem of our project.

5 REFERENCES

Gevorgyan, M., Mamikonyan, A., & Beyeler, M. (2020). OpenCV 4 with Python Blueprints: Build creative computer vision projects with the latest version of OpenCV 4 and Python 3, 2nd Edition. Packt Publishing.

Huang, K., Qu, X., Chen, S., Chen, Z., Zhang, W., Qi, H. and Zhao, F., 2020. Superb Monocular Depth Estimation Based on Transfer Learning and Surface Normal Guidance. Sensors, 20(17), p.4856.

Lin, L., Huang, G., Chen, Y., Zhang, L., & He, B. (2020). Efficient and High-Quality Monocular Depth Estimation via Gated Multi-Scale Network. IEEE Access, 8, 7709–7718.

Gupta, H. (2017, July 8). One Shot Learning with Siamese Networks in PyTorch. Hacker Noon. https://hackernoon.com/one-shot-learning-with-siamese-networks-in-pytorch-8ddaab10340e

Luo, Y. (2011). Midfrequency-based real-time blind image restoration via independent component analysis and genetic algorithms. Optical Engineering, 50(4), 047004. https://doi.org/10.1117/1.3567072

P. Viola and M. Jones, “Robust real-time object detection,” International Journal of Computer Vision, 57(2), , 2004.

Junguk Cho, Shahnam Mirzaei, Jason Oberg, Ryan Kastner, “FPGA-Based Face Detection System Using Haar Classifiers,” Proceeding of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays, , 2009.

Paper: Alan Lukezic, Tom'as Voj'ir, Luka Cehovin Zajc, Jir'i Matas, and Matej Kristan. Discriminative correlation filter tracker with channel and spatial reliability. International Journal of Computer Vision, 2018.

J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In proceedings of the European Conference on Computer Vision, 2012.

Helmut Grabner, Michael Grabner, and Horst Bischof. Real-time tracking via on-line boosting. In BMVC, volume 1, page 6, 2006.

Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 983–990. IEEE, 2009.

Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 34(7):1409–1422, 2012.

Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In Pattern Recognition (ICPR), 2010 20th International Conference on, pages 2756–2759. IEEE, 2010.

David S. Bolme, J. Ross Beveridge, Bruce A. Draper, and Man Lui Yui. Visual object tracking using adaptive correlation filters. In Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

https://data.vision.ee.ethz.ch/cvl/publications/papers/proceedings/eth_biwi_01229.pdf

https://data.vision.ee.ethz.ch/cvl/publications/papers/proceedings/eth_biwi_01229.pdf

https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf

https://research.fb.com/publications/deepface-closing-the-gap-to-human-levelperformance-in-face-verification/

https://github.com/opencv/opencv_3rdparty/raw/19512576c112aa2c7b6328cb0e8d589a4a90a26d/res10_300x300_ssd_iter_140000_fp16.caffemodel

https://github.com/opencv/opencv/blob/master/samples/dnn/face_detector/deploy.prototxt

https://github.com/opencv/opencv_extra/blob/master/testdata/dnn/opencv_face_detector.pbtxt

https://github.com/davisking/dlib