

# **1. Introduction**

## ***1.1 Introduction to Project :***

There are a number of businesses, government agencies and even churches that are employing facial recognition technology. Despite privacy concerns, it's hard to argue against the good things that the tech can be used for. Identifying missing children, helping the blind, catching shoplifters, tracking churchgoers, these are only a number of ways facial recognition is being used today and we are using this technology for identifying and find the missing children.



Fig. 1: Missing Children Data

## ***1.2 Project Category :***

It is an Application Development project which will detect person.

## ***1.3 Objectives :***

The study is aimed at attainment of following objectives:-

- An algorithm/model/library to detect face from the image.
- Crop the face and classify into different classes (Different Person)
- Make GUI for users so that they can use it easily and retrieve information about the uploader, which will use to collect more information about the child.

### ***1.4 Problem Formulation :***

There is still no web app that is using facial recognition to find and identify missing children. Also there is no hundred percent efficient algorithm that can detect and classify the face into different classes so we are trying to make a web app which will use facial recognition to find missing children. It is possible that our solution is not accurate but by this prototype we can have an idea and some research can benefit us a lot if we are succeed to make such app.

We are trying to solve these problems:-

- Measure characteristics of a face from image
- Classify it into different classes i.e. detect if the face matches a person already seen before
- Show details to the verifying party if there is a match

### ***1.5 Identification / Reorganization Need :***

As you can see in the figure, the number of lost children in US is pretty high therefore we can assume the condition in India leading to a lot of missing reports in police station but by using face recognition we can identifying and find missing children.

Face recognition has a very wide applications. Some of them are here:-

1. *Casinos use it to help addictive gamblers:* They can track them, identify them on the web, and let them know how often they're going to the casino. They claim they use it to help the addicts, but tracking those pesky card counters would come to no surprise.
2. *Dating sites match people with the same facial features:* Dating site findyourfacemate.com uses the theory that people are most attracted to those that look like them. Similarly, Doggleganger can also match you up with a DOG that looks like you based on the stereotype that people tend to look like their pets.
3. *Law enforcement and security use it to track down criminals:* This one comes as no surprise. The FBI and Homeland Security have been able to use this for a long time to identify threats. They've been known to snoop on crowds at sporting events and other places, so be good out there.
4. *Social Media can tag people automatically:* Facebook and its 140 billion photos of 600 million users uses Facial Recognition to identify everyone. There was initially a huge

backlash on the default tag settings, and there were quite a few complaints about privacy. But you do have the option to not be tagged automatically if you don't want to.

5. *Credit card companies will allow you to shop with your face:* MasterCard is about to drive Apple Pay into retirement along with checks and bartering. The company is researching ways to let their customers pay for things using a selfie, which prevents fraud and identity theft, and you don't have to remember passwords or put your credit card information on the web.
6. *Upscale hotels greet guests upon arrival:* Bellhops are really going for the tips trying to make every guest feel extra special. Everyone's a celebrity! As long as you're not totally off the grid, they'll know your name by the time you get to the door.
7. *Bars and restaurants pick out underage drinkers:* Sorry kiddos, you used to have a fake ID, but it might be harder to get a fake face on the black market. Or you could be responsible and wait until you're twenty-one, whichever you find easier.
8. *Schools in the UK have started using it to take attendance:* This has been going on in the UK for awhile, and will inevitably spread to other countries. Teachers can spend more time teaching and less time calling out names and searching through faces.

### ***1.6 Existing Systems :***

Latest example is IPHONE X using facial recognition for unlocking the phone. Therefore face recognition has a very wide applications.

### ***1.7 Proposed System :***

In our web app we first detect the face from the image and make a bbox over the detected area then crop that bbox and save that cropped image into disk. Then provide that cropped image as pipeline input and as a result we can find the class which has maximum confidence score related to the image. When we finalize the class then we return the name, number, email etc information about the uploader to the person who was validating the image that image is matched and your missing member can be found there. So it can become very useful to find and identifying missing children.

A latest post on 9 October 2017, is enough to show that India will also use deep learning applications in daily life:

***High Court asks Delhi to set up face recognition software to help trace and rescue missing children*** : The Delhi High Court has asked the AAP government and the police to coordinate with each other to procure at the earliest a facial recognition software which would help trace and rescue missing children. A bench of Justices G S Sistani and Chander Shekhar asked the police to forward, within three weeks, a proposal to the Delhi government regarding the specifications required in the software. The bench was hearing a plea regarding missing children in which it has been examining ways and means to address the issue of tracing them and restoring them to their families. Earlier, the court had observed that crowded places were often used by child traffickers to kidnap children and that the kids were vulnerable at the railway stations.

### ***1.8 Unique Features of the Systems :***

It will detect face of the person and match it with the existing dataset and find the best suitable match. But as we all know that it is very difficult to find hundred percent accuracy in detection and accuracy but we are trying our best to detect and classify the image correctly.

## **2. Requirement Analysis and System Specification**

### ***2.1 Feasibility Analysis***

A feasibility study is an evaluation and analysis of the potential of the proposed project which is based on extensive investigation and research to give full comfort to the decision makers. Feasibility studies aim to objectively and rationally uncover the strengths and weakness of an existing business or proposed venture, opportunities and threats as presented by the environment, the resources required to carry through, and ultimately the prospects for success.

In its simplest terms, the two criteria to judge feasibility are the cost required and value to be attained. As such, a well-designed feasibility study should provide a historical background of the business or project, description of the product or service, accounting statements, details of the operation and management, marketing research and policies, financial data, legal requirements and tax obligations. Generally, feasibility studies precede technical development and project implementation. When a new project is proposed, it normally goes through feasibility assessment. Feasibility study is carried out to determine whether the proposed system is possible to develop with available resources and what should be the cost consideration. Facts considered in the feasibility analysis were

- ☐ Technical feasibility
- ☐ Economic feasibility
- ☐ Operational feasibility

#### **Technical feasibility**

Technical feasibility centers on the existing computer system (hardware/software) and to what extent it can support the proposed addition also the organization already has sufficient high end machines to serve the processing requirements of the proposed system. The project is technically feasible as the technology involved in the project is easily available. To display the application, the only technical aspects needed are mentioned below:

Operating environment any operating system

Platform Python

Database Sqlite

For users: internet browser & internet connection

This project is technical feasible because in this project add information, search result, all these things are technically feasible. In our project, if we want to add new data, then simply we click on add menu and store into database.

### **Economic feasibility**

Economic feasibility is the most frequently used method for evaluating the effectiveness of the candidate system. More commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with the costs. If benefits outweigh the costs, then the decision is made to design and implement the system. The project is economically feasible as the only cost involved is having a computer with the minimum requirements mentioned earlier. For the users to access the application, the only cost involved will be in getting access to the internet. This project is economical feasible. In this project we use the JSP and for data storage we use MYSQL. It is economical feasible because Glassfish Server and MySQL database are both available free of cost.

### **Operational feasibility**

Operational feasibility is evaluation is to determine whether system is operationally is acceptable. During this study it was determine whether the system will operate in the way that user wants or not. This project is also operational feasible because in this project we provide the graphical user interface (GUI) which is easy to understand & operate. It also provides the user friendly interface. The user will easily use the system. In this project we use the buttons, text box, images which is easily understandable for end user.

The system will be used if it is developed well skill then be resistance for users that undetermined

1. No major training and new skills required as it is based on DBMS model.
2. It will help in the time saving and fast processing and applications.
3. New product will provide all the benefits of present system with better performance.
4. Improved information, better management and collection of the reports.

## ***2.2 Software Requirement Specification Document :***

### **Data Requirement**

In this web application, we need some input images for training and when training is done , a suspect picture will be uploaded. Linear classifier will classify the suspect in the available class level. So we need some input images and a suspect image. We need basically image data.

### **Functional Requirement**

In Software engineering and systems engineering, a functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define *what* a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases. Functional requirements are supported by non-functional requirements (also known as *quality requirements*), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). Generally, functional requirements are expressed in the form "system must do <requirement>", while non-functional requirements are "system shall be <requirement>". The plan for implementing *functional* requirements is detailed in the system *design*. The plan for implementing *non-functional* requirements is detailed in the system *architecture*.

It includes image data, align face, detect face, crop face, and generate embeddings functions.

### **Performance Requirement**

- To ensure system reliability.
- To ensure system accuracy.
- To ensure system flexibility.
- To ensure system maintainability.
- To improve the efficiency of system.

### **Maintainability Requirements**

- To ensure that bug fixing in the system takes place properly.
- To ensure that the quality of the system doesn't degrade with time.
- To ensure that the system is easy to maintain and update.

- To provide ease of operations.
- To provide easy adaptability.

### **Security Requirements**

- To maintain the integrity of system.
- To ensure system recovery in case of system crashes.

### **Look and Feel Requirements**

- The application should be easy and understandable for the users.

## ***2.3 Validation***

The face is detected appropriately in most of the cases leading to correct classification and hence leads to proper verification. The system is reliable and efficient as it provides a one-time investment. It also reduces a large amount of human effort and time required. The system uses easily available softwares and technology and hence leads to cost reduction. The system provides a user friendly application that is easily understandable to an individual. The application built is easy to update and maintain.

## ***2.4 Expected Hurdles***

1. Images are degraded.
2. If images of twins is uploaded then it will not differentiate between them.
3. If image with a black googles , then face will not detect properly.
4. Cost hurdles.
5. Effort Hurdles.
6. Hosting Hurdles.
7. Images that have high differences in intensity may not train properly.

## ***2.5 Software Development Life Cycle (SDLC) to be used***

The software development life cycle of the system would consist of the planning, design, development, testing and deployment phase.



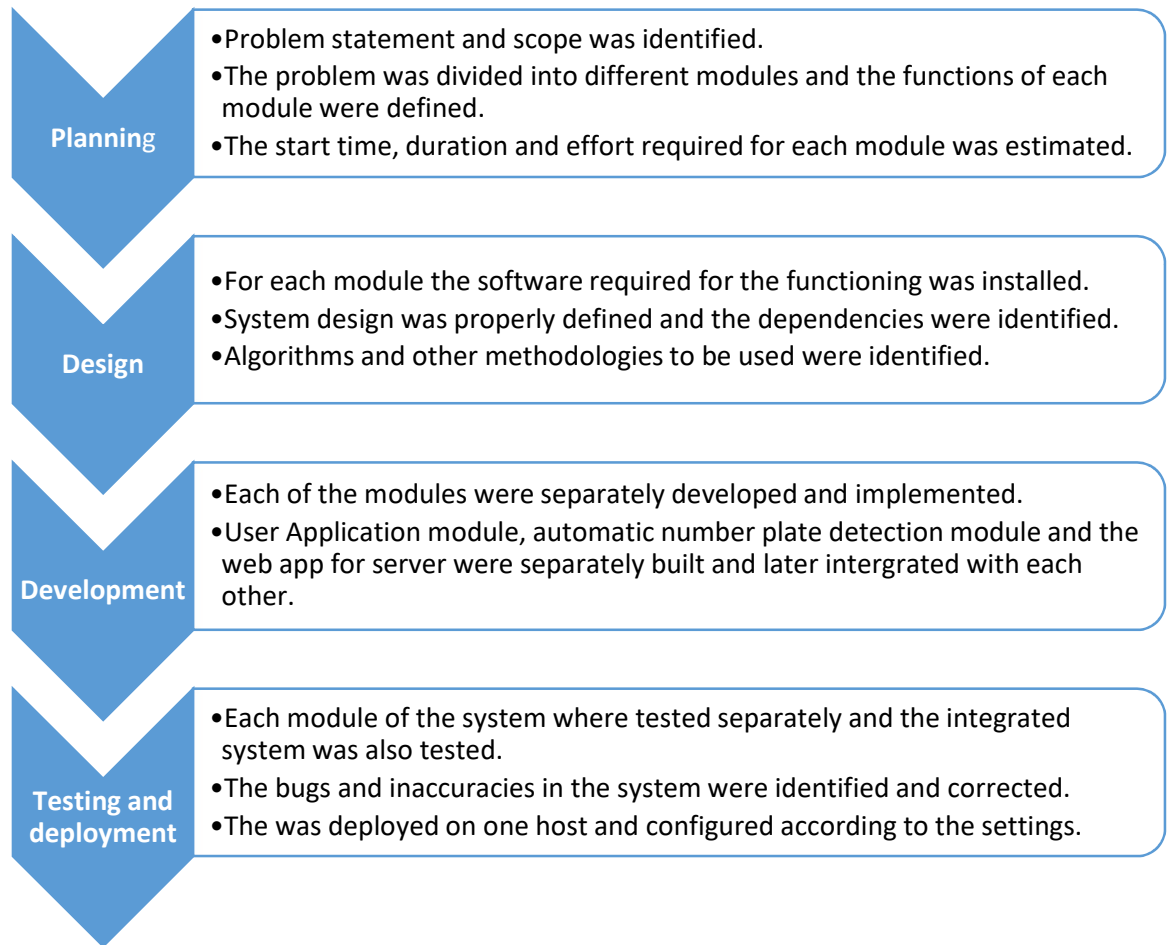


Fig2: SDLC of the proposed system

We use incremental approach where we make small modules of the project and at the end merge or embed all the modules to make a full application.

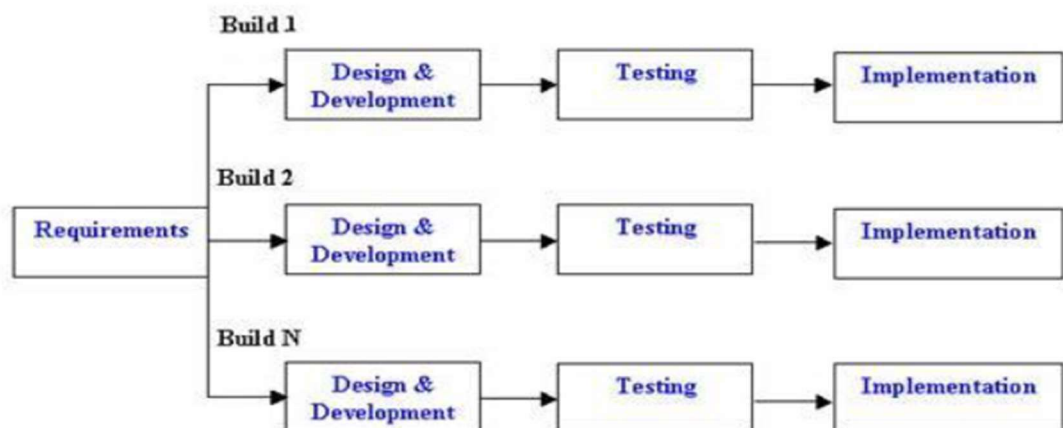


Fig3. Incremental Model

## **3. System Design**

### ***3.1 Design Approach***

The design approach used in creating the system is mainly object oriented as all the functions which are used in views.py have object-oriented property. All class methods take in self as their first positional argument, and all views take in request as their first positional argument. Taking in this argument allows access to state which would otherwise be difficult to access. Changing the order of URL patterns is equivalent to changing the polymorphic properties of a class and dynamic method lookup. The design approach used in creating the system is mainly object oriented. The code contains no use of functional programming approach. The system uses implementations of neural network, classifier & Django for front hand to detect a person.

### ***3.2 Detail Design***

**Finding a face** – This step mainly deals with detecting a face in an image

**Aligning the face** – Faces may not be aligned properly in an image. This step deals with warping each picture so that the eyes and lips are always facing in the sample place in the image

**Face Measurement** – The third step in the pipeline is to find measurements from a face.

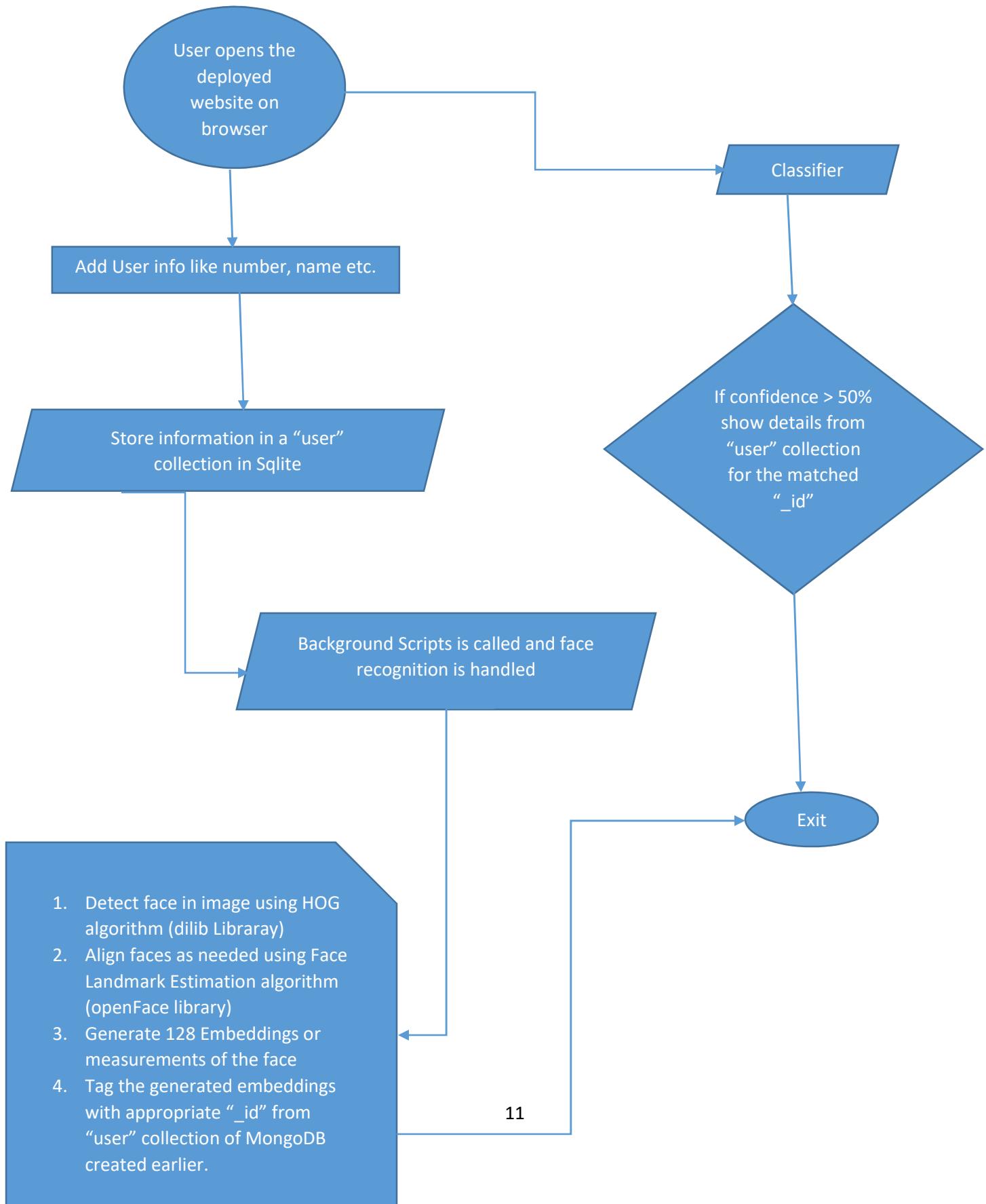
**Finding the person's face** – Here we check whether the test image is of a person we've already seen before.

Some Designing points:

- 1. Hypothesis Generation** – understanding the problem better by brainstorming possible factors that can impact the outcome.
- 2. Data Exploration** – looking at categorical and continuous feature summaries and making inferences about the data.
- 3. Data Cleaning** – imputing missing values in the data and checking for outliers
- 4. Feature Engineering** – modifying existing variables and creating new ones for analysis
- 5. Model Building** – making classifying models on the data.

### 3.3 Structured Analysis and Design Tool

#### Flowchart



### ***3.4 User Interface Design***

The front-end consists of two forms to be filled by users of the web app. The guardian whose child is lost needs to fill in the details which ask for his/her name, E-mail, Phone number. Apart from this, he/she needs to upload 6 images of the lost child which will be trained at the back-end using convolution neural networks. This was all about the first form.

The second form contains an option to upload an image. This image will be uploaded by someone who thinks he/she has seen the lost child before. Now this image will be trained against the previously uploaded images. The result will be prediction of a particular image with some confidence.

### ***3.5 User Interface Design***

#### ***3.5.1 ER Diagram***

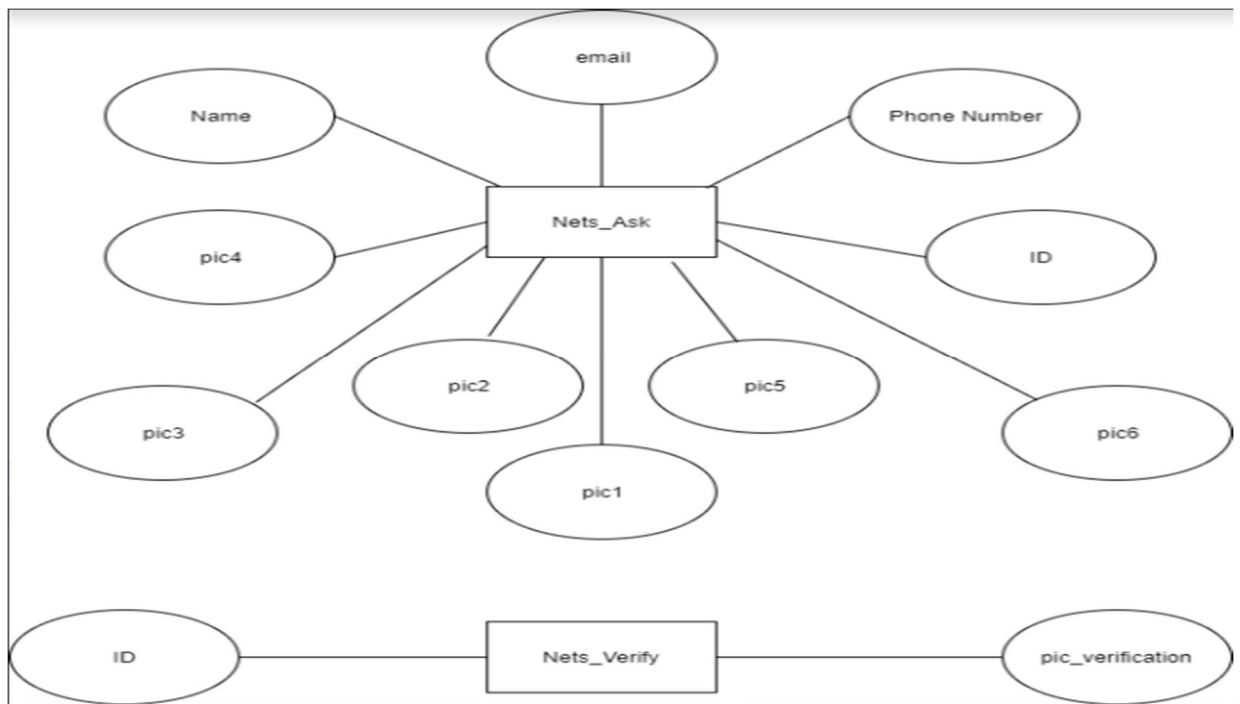


Fig4. ER Diagram

#### ***3.5.2 Normalization***

The tables in our projects are 2NF normalized as 2NF.

As per the Second Normal Form there must not be any partial dependency of any column on primary key. It means that for a table that has concatenated primary key, each column in the table that is not part of the primary key must depend upon the entire concatenated key for its existence. If any column depends only on one part of the concatenated key, then the table fails **Second normal form**.

### ***3.5.3 Database Manipulation***

We insert images, name, email and phone number in the table and when we find a match then we retrieve values (Name, Phone Number, Email etc.) of that matched or detected person.

### ***3.5.4 Database Connection Controls and Strings***

```
from django.db import models
```

```
class Ask(models.Model):
```

```
    name = models.CharField(max_length=500)
```

```
    phone = models.CharField(max_length=200, blank=True)
```

```
    email = models.CharField(max_length=500)
```

```
    pic = models.ImageField(upload_to="images/", blank=True)
```

```
    pic2 = models.ImageField(upload_to="images/", blank=True)
```

```
    pic3 = models.ImageField(upload_to="images/", blank=True)
```

```
    pic4 = models.ImageField(upload_to="images/", blank=True)
```

```
    pic5 = models.ImageField(upload_to="images/", blank=True)
```

```
    pic6 = models.ImageField(upload_to="images/", blank=True)
```

```
    def __str__(self):
```

```
        return self.name
```

```
class Verify(models.Model):
```

```
    pic_verification = models.ImageField(upload_to="images/", blank=True)
```

```
    def __str__(self):
```

```
        return self.name
```

### 3.6 Methodology

We have divided the project into two basic components, the backend and the frontend. Furthermore, different technologies and stacks are used in each of those. The frontend provides a way to both the user as well as the party verifying the person lost. The backend handles the logic that deals with facial recognition and categorization for the images.

#### Step 1: Finding a face

The first step in our pipeline is *face detection*. We're using a method invented in 2005 called **Histogram of Oriented Gradients (HOG)** for face detection.

- Make the image black and white, since we don't need any colour data
- For every pixel, we look at pixels directly surrounding it.
- Determine how dark the current pixel is, as compared to the pixels surrounding it. Then we draw an arrow along the direction of the image getting darker.
- The above representation would be called a HOG pattern.
- We might have to do max-pooling to handle the excess features from the HOG pattern.
- Compare the generated HOG pattern to a known HOG pattern. The known HOG pattern has been trained on a large dataset of human faces.

The **dlib** python library, handles the above HOG algorithm neatly. We have used it in our project.

#### Step 2: Aligning the face

This is the second step in our pipeline. Faces may not be aligned properly in an image. We as humans can perceive it easily, but for a computer to perceive a face reliably, it must be aligned properly.

To handle this, we try to warp each picture so that the eyes and lips are always facing in the sample place in the image.

We have used the **face landmark estimation** algorithm. We come up with 68 landmarks that exist on every face. We train a machine learning algorithm to be able to identify all these landmarks. Once we know where the landmarks, we'll apply affine transformations to the image (rotate, scale, shear).

We have used **dlib**, **opencv**, **openface** libraries in this step.

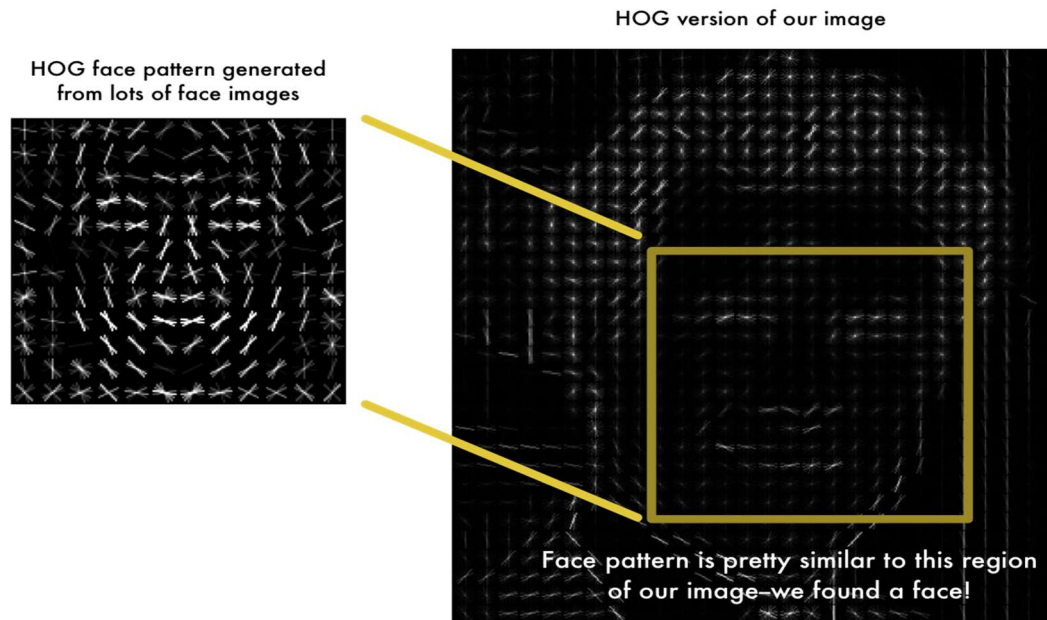


Fig 5: Face Pattern

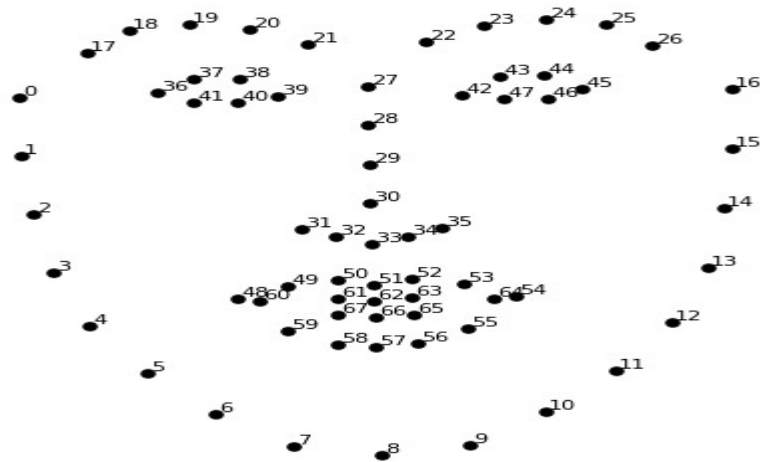


Fig 6: Image created by Brandon Amos, CMU (Openface)

### Step 3: Face measurements

The third step in the pipeline is to find measurements from a face. A deep convolutional neural network is trained to generate 128 measurements, or embeddings from a face.

The way this works can be described as follows:

- Take 3 images of person A, person B and person A'
- The neural network, at every step tweaks itself slightly so that A and A' are closer and A and B are much farther apart.
- Do this on a large enough data set to train the net.

We have used **openface** library for the above step

#### **Step 4: Finding the person's face**

This is the last step in the image recognition pipeline. Here we check if a test image is of a person we have already seen before.

Doing this over a database is very hard, as there could be many many possible images. Instead, we use a classifier.

After generating the 128 embeddings from the face in Step 3, we train our classifier with the said embeddings.

Using the test image on the trained classifier will neatly match it to someone's face with a confidence score.

We have used **openface** library in the above step.



## **4. Implementation, Testing and Maintenance**

### ***4.1 Introduction to Languages , IDE's , Tools and Technologies Used***

#### **Python**

**Python** is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

Python is used extensively for system administration (many vital components of Linux distributions are written in it); also, it is a great language to teach programming to novices. NASA has used Python for its software systems and has adopted it as the standard scripting language for its Integrated Planning System. Python is also extensively used by Google to implement many components of its Web Crawler and Search Engine & Yahoo! for managing its discussion groups.

Python within itself is an interpreted programming language that is automatically compiled into bytecode before execution (the bytecode is then normally saved to disk, just as automatically, so that compilation need not happen again until and unless the source gets changed). It is also a dynamically typed language that includes (but does not require one to use) object-oriented features and constructs.

The most unusual aspect of Python is that whitespace is significant; instead of block delimiters (braces → "{" in the C family of languages), indentation is used to indicate where blocks begin and end.

Another great feature of Python is its availability for all platforms. Python can run on Microsoft Windows, Macintosh and all Linux distributions with ease. This makes the programs very portable, as any program written for one platform can easily be used on another.

Python provides a powerful assortment of built-in types (e.g., lists, dictionaries and strings), a number of built-in functions, and a few constructs, mostly statements. For example, loop constructs that can iterate over items in a collection instead of being limited to a simple range of integer values. Python also comes with a powerful standard library, which includes hundreds of modules to provide routines for a wide variety of services including regular expressions and TCP/IP sessions.

## **Django**

**Django** is a free and open-source web framework, written in Python, which follows the model-view-template (MVT) architectural pattern. It is maintained by the Django Software Foundation (DSF), an independent organization established as a non-profit. Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "plug ability" of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models. Some well-known sites that use Django include the Public Broadcasting Service, Instagram, Mozilla, The Washington Times, Disqus, Bitbucket, and Nextdoor. It was used on Pinterest, but later the site moved to a framework built over Flask.

### *History*

Django was created in the fall of 2003, when the web programmers at the *Lawrence Journal-World* newspaper, Adrian Holovaty and Simon Willison, began using Python to build applications. It was released publicly under a BSD license in July 2005. The framework was named after guitarist Django Reinhardt. In June 2008, it was announced that a newly formed Django Software Foundation (DSF) would maintain Django in the future.

### *Features*

Despite having its own nomenclature, such as naming the callable objects generating the HTTP responses "views", the core Django framework can be seen as an MVC architecture. It

consists of an object-relational mapper (ORM) that mediates between data models(defined as Python classes) and a relational database ("Model"), a system for processing HTTP requests with a web templating system("View"), and a regular-expression-based URL dispatcher ("Controller").

Also included in the core framework are:

- a lightweight and standalone web server for development and testing
- a form serialization and validation system that can translate between HTML forms and values suitable for storage in the database
- a template system that utilizes the concept of inheritance borrowed from object-oriented programming
- a caching framework that can use any of several cache methods
- support for middleware classes that can intervene at various stages of request processing and carry out custom functions
- an internal dispatcher system that allows components of an application to communicate events to each other via pre-defined signals
- an internationalization system, including translations of Django's own components into a variety of languages
- a serialization system that can produce and read XML and/or JSON representations of Django model instances
- a system for extending the capabilities of the template engine
- an interface to Python's built-in unit test framework

#### *Bundled applications*

The main Django distribution also bundles a number of applications in its "contrib" package, including:

- an extensible authentication system
- the dynamic administrative interface
- tools for generating RSS and Atom syndication feeds
- a site's framework that allows one Django installation to run multiple websites, each with their own content and applications
- tools for generating Google Sitemaps

- built-in mitigation for cross-site request forgery, cross-site scripting, SQL injection, password cracking and other typical web attacks, most of them turned on by default
- a framework for creating GIS applications

## OpenFace

The following overview shows the workflow for a single input image of Sylvester Stallone from the publicly available LFW dataset.

1. Detect faces with a pre-trained models from dlib or OpenCV.
2. Transform the face for the neural network. This repository uses dlib's real-time pose estimation with OpenCV's affine transformation to try to make the eyes and bottom lip appear in the same location on each image.
3. Use a deep neural network to represent (or embed) the face on a 128-dimensional unit hypersphere. The embedding is a generic representation for anybody's face. Unlike other face representations, this embedding has the nice property that a larger distance between two face embeddings means that the faces are likely not of the same person. This property makes clustering, similarity detection, and classification tasks easier than other face recognition techniques where the Euclidean distance between features is not meaningful.
4. Apply your favorite clustering or classification techniques to the features to complete your recognition task. See below for our examples for classification and similarity detection, including an online web demo.

## SQLite Database

SQL is the most popular Open Source Relational SQL Database Management System. SQL is one of the best RDBMS being used for developing various web-based software applications. SQL is developed, marketed and supported by SQL AB, which is a Swedish company.

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds. Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems. Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of

data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as Foreign Keys.

**A Relational DataBase Management System (RDBMS)** is a software that –

- Enables you to implement a database with tables, columns and indexes.
- Guarantees the Referential Integrity between rows of various tables.
- Updates the indexes automatically.
- Interprets an SQL query and combines information from various tables.

### *RDBMS Terminology*

Before we proceed to explain the MySQL database system, let us revise a few definitions related to the database.

- **Database** – A database is a collection of tables, with related data.
- **Table** – A table is a matrix with data. A table in a database looks like a simple spreadsheet.
- **Column** – One column (data element) contains data of one and the same kind, for example the column postcode.
- **Row** – A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.
- **Redundancy** – Storing data twice, redundantly to make the system faster.
- **Primary Key** – A primary key is unique. A key value cannot occur twice in one table. With a key, you can only find one row.
- **Foreign Key** – A foreign key is the linking pin between two tables.
- **Compound Key** – A compound key (composite key) is a key that consists of multiple columns, because one column is not sufficiently unique.
- **Index** – An index in a database resembles an index at the back of a book.
- **Referential Integrity** – Referential Integrity makes sure that a foreign key value always points to an existing row.

## SQL

SQL is a fast, easy-to-use RDBMS being used for many small and big businesses. SQL is developed, marketed and supported by MySQL AB, which is a Swedish company. SQL is becoming so popular because of many good reasons –

- SQL is released under an open-source license. So you have nothing to pay to use it.
- SQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- SQL uses a standard form of the well-known SQL data language.
- SQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- SQL works very quickly and works well even with large data sets.
- SQL is very friendly to PHP, the most appreciated language for web development.
- SQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).

MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

## NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code and useful linear algebra, Fourier transform, and random number capabilities

## *4.2 Coding Standards of Language Used*

### **Puppet Coding Standard**

**Purpose:**

The purpose of this style guide is to promote consistent formatting in the Puppet Language, especially across modules, giving users and developers of Puppet modules a common pattern, design, and style to follow. Additionally, consistency in code and module structure makes continued development and contributions easier.

We recommend using `puppet-lint` and `metadata-json-lint` within your module to check for compliance with the style guide.

### **Terminology and guiding principles:**

Unless explicitly called out, everything discussed here applies specifically to Puppet (that is, Puppet modules, Puppet classes, etc.). The name ‘Puppet’ is not appended to every topic discussed.

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119.

We can never cover every circumstance you might run into when developing Puppet code. When you need to make a judgement call, keep in mind a few general principles.

1. **Readability matters.**

If you have to choose between two equal alternatives, pick the more readable one. This is subjective, but if you can read your own code three months from now, it’s a great start. In particular, code that generates readable diffs is highly preferred.

2. **Scoping and simplicity are key.**

When in doubt, err on the side of simplicity. A module should contain related resources that enable it to accomplish a task. If you describe the function of your module and you find yourself using the word “and,” consider splitting the module. You should have one goal, with all your classes and parameters focused on achieving it.

3. **Your module is a piece of software.**

At least, you should treat it that way. When it comes to making decisions, choose the option that is easier to maintain in the long term

### 4.3 Project Scheduling

The **project schedule** is the tool that communicates what work needs to be performed, which resources of the organization will perform the work and the timeframes in which that work needs to be performed.

#### Modules

- Research and Design
- Development
- Hypothesis generation
- Data exploration
- Data cleaning
- Model building

#### Gantt Chart

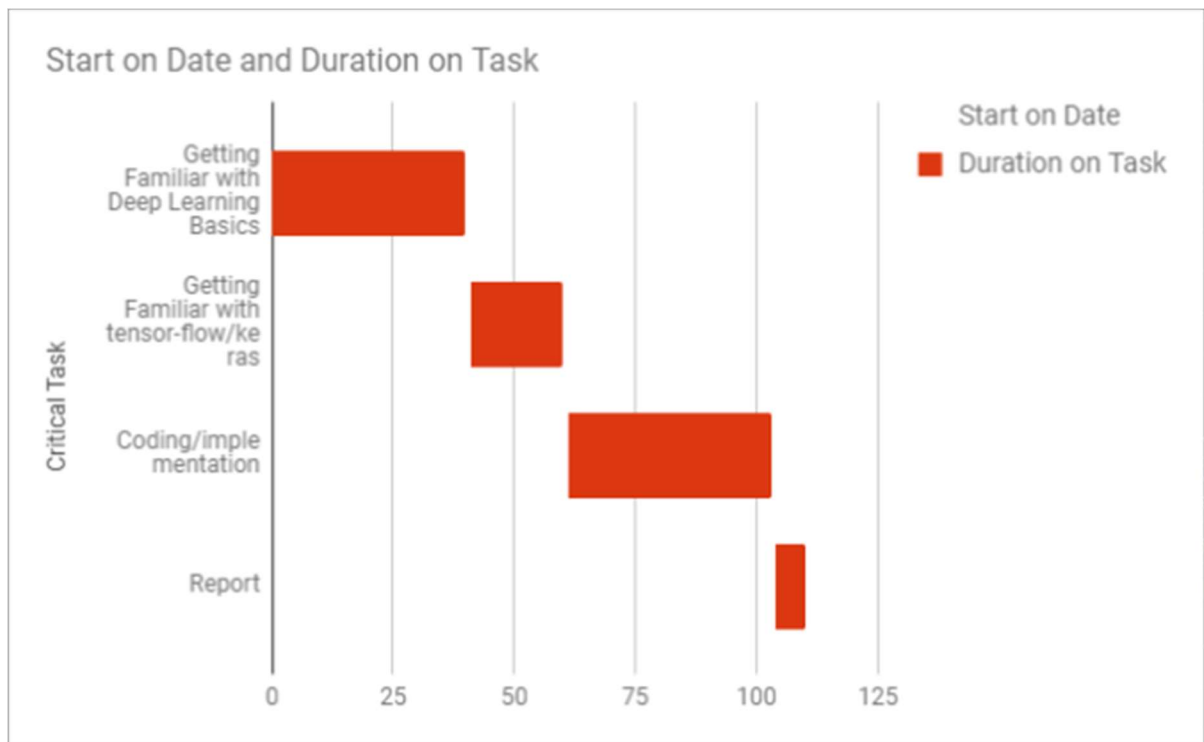


Fig7. Gantt Chart



## 4.4 Testing Techniques and Test Plans

### Unit Testing

In computer programming, **unit testing** is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use.

**Unit Testing** is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed.

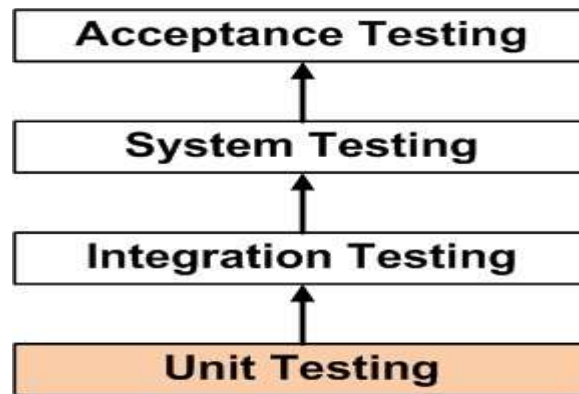


Fig7. Testing Phases

A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output. In procedural programming a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.)

Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing.

Classes may have references to other classes, testing a class can frequently spill over into testing another class. A common example of this is classes that depend on a database: in order to test the class, the tester often writes code that interacts with the database. This is a mistake, because a unit test should usually not go outside of its own class boundary, and especially should not cross such

process/network boundaries because this can introduce unacceptable performance problems to the unit test-suite. Crossing such unit boundaries turns unit tests into integration tests, and when test cases fail, makes it less clear which component is causing the failure. Instead, the software developer should create an abstract interface around the database queries, and then implement that interface with their own mock object. By abstracting this necessary attachment from the code (temporarily reducing the net effective coupling), the independent unit can be more thoroughly tested than may have been previously achieved. This results in a higher-quality unit that is also more maintainable.

### *Techniques*

Unit testing is commonly automated, but may still be performed manually. The IEEE does not favor one over the other. The objective in unit testing is to isolate a unit and validate its correctness. A manual approach to unit testing may employ a step-by-step instructional document. However, automation is efficient for achieving this, and enables the many benefits listed in this article. Conversely, if not planned carefully, a careless manual unit test case may execute as an integration test case that involves many software components, and thus preclude the achievement of most if not all of the goals established for unit testing.

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

### *ADVANTAGES*

#### **Find problems early**

Unit testing finds problems early in the development cycle. This includes both bugs in the programmer's implementation and flaws or missing parts of the specification for the unit. The process of writing a thorough set of tests forces the author to think through inputs, outputs, and error conditions, and thus more crisply define the unit's desired behavior. The cost of finding a bug before coding begins or when the code is first written is considerably lower than the cost of detecting, identifying, and correcting the bug later; bugs may also cause problems for the end-users of the software. Code can be impossible or difficult to test if poorly written, thus unit testing can force developers to structure functions and objects in better ways.

### **Facilitates change**

Unit testing allows the programmer to refactor code or upgrade system libraries at a later date, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes which may break a design contract.

### **Simplifies integration**

Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier.

### **Documentation**

Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface.

Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviors that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

### **Design**

When software is developed using a test-driven approach, the combination of writing the unit test to specify the interface plus the refactoring activities performed after the test is passing, may take the place of formal design. Each unit test can be seen as a design element specifying classes, methods, and observable behavior.

### **Integration Testing**

**Integration Testing** is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

### **Definition by ISTQB**

- **Integration testing:** Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. See also *component integration testing*, *system integration testing*.
- **Component integration testing:** Testing performed to expose defects in the interfaces and interaction between integrated components.
- **System integration testing:** Testing the integration of systems and packages; testing interfaces to external organizations (e.g. Electronic Data Interchange, Internet).

### *ANALOGY*

During the process of manufacturing a ballpoint pen, the cap, the body, the tail and clip, the ink cartridge and the ballpoint are produced separately and unit tested separately. When two or more units are ready, they are assembled and Integration Testing is performed. For example, whether the cap fits into the body or not.

### *METHOD*

Any of Black Box Testing, White Box Testing, and Gray Box Testing methods can be used. Normally, the method depends on your definition of 'unit'.

### *TASKS*

- Integration Test Plan
  - Prepare
  - Review
  - Rework
  - Baseline
- Integration Test Cases/Scripts
  - Prepare
  - Review
  - Rework
  - Baseline
- Integration Test
  - Perform

**When is Integration Testing performed?**

Integration Testing is performed after Unit Testing and before System Testing.

### *APPROACHES*

- *Top Down* is an approach to Integration Testing where top level units are tested first and lower level units are tested step by step after that. This approach is taken when top down development approach is followed. Test Stubs are needed to simulate lower level units which may not be available during the initial phases.
- *Bottom Up* is an approach to Integration Testing where bottom level units are tested first and upper level units step by step after that. This approach is taken when bottom up development approach is followed. Test Drivers are needed to simulate higher level units which may not be available during the initial phases.

### *TIPS*

- Ensure that you have a proper Detail Design document where interactions between each unit are clearly defined. In fact, you will not be able to perform Integration Testing without this information.
- Ensure that you have a robust Software Configuration Management system in place. Or else, you will have a tough time tracking the right version of each unit, especially if the number of units to be integrated is huge.
- Make sure that each unit is first unit tested before you start Integration Testing.
- As far as possible, automate your tests, especially when you use the Top Down or Bottom Up approach, since regression testing is important each time you integrate a unit, and manual regression testing can be inefficient.

### **Gray Box Testing**

Gray Box Testing is a software testing method which is a combination of Black Box Testing method and White Box Testing method. In Black Box Testing, the internal structure of the item being tested is unknown to the tester and in White Box Testing the internal structure is known. In Gray Box Testing, the internal structure is partially known. This involves having access to internal data

structures and algorithms for purposes of designing the test cases, but testing at the user, or black-box level.

Gray Box Testing is named so because the software program, in the eyes of the tester is like a gray/semi-transparent box; inside which one can partially see. Grey-box testing is a technique to test the application with having a limited knowledge of the internal workings of an application. In software testing, the phrase the more you know, the better carries a lot of weight while testing an application. Mastering the domain of a system always gives the tester an edge over someone with limited domain knowledge. Unlike black-box testing, where the tester only tests the application's user interface; in grey-box testing, the tester has access to design documents and the database. Having this knowledge, a tester can prepare better test data and test scenarios while making a test plan.

#### *EXAMPLE*

An example of Gray Box Testing would be when the codes for two units/ modules are studied (White Box Testing method) for designing test cases and actual tests are conducted using the exposed interfaces (Black Box Testing method).

#### *LEVELS APPLICABLE TO*

Though Gray Box Testing method may be used in other levels of testing, it is primarily useful in Integration Testing.

Advantages	Disadvantages
<ul style="list-style-type: none"><li>• Offers combined benefits of black-box and white-box testing wherever possible.</li><li>• Grey box testers don't rely on the source code; instead they rely on interface definition and functional specifications.</li></ul>	<ul style="list-style-type: none"><li>• Since the access to source code is not available, the ability to go over the code and test coverage is limited.</li><li>• The tests can be redundant if the software designer has already run a test case.</li></ul>

<ul style="list-style-type: none"><li>• Based on the limited information available, a grey-box tester can design excellent test scenarios especially around communication protocols and data type handling.</li><li>• The test is done from the point of view of the user and not the designer.</li></ul>	<ul style="list-style-type: none"><li>• Testing every possible input stream is unrealistic because it would take an unreasonable amount of time; therefore, many program paths will go untested.</li></ul>
---	--

## **5. Results and Discussions**

### ***5.1 User Interface Representation***

It contains backend and front end.

**Front End:** It means the technologies that are used to build the user friendly interface of a particular software or the technologies that are used to build that part of a software which we can see.

- HTML
- MongoDB/Sqlite (Databases)

**Back End:** It means the technologies that are responsible for the activity that are seen on the web page or a software.

- Python
- Opencv
- Neural Network Training (Keras and Tensorflow)
- Dilib
- openface

#### ***5.1.1 Brief Description of Various Modules of the system***

This project contains a web page which gives you 2 type of functionality. User will upload pictures of person that he/she wants to detect or want to search. Then a suspected picture will be uploaded by any citizen that has suspect on any person. Then user pictures will be trained and stored in our data base and when a suspect picture is uploaded, a classification function will operate, which will give the best match of class that has highest confidence.

### ***5.2 Snapshots of system with brief detail of each***



## Detect Face

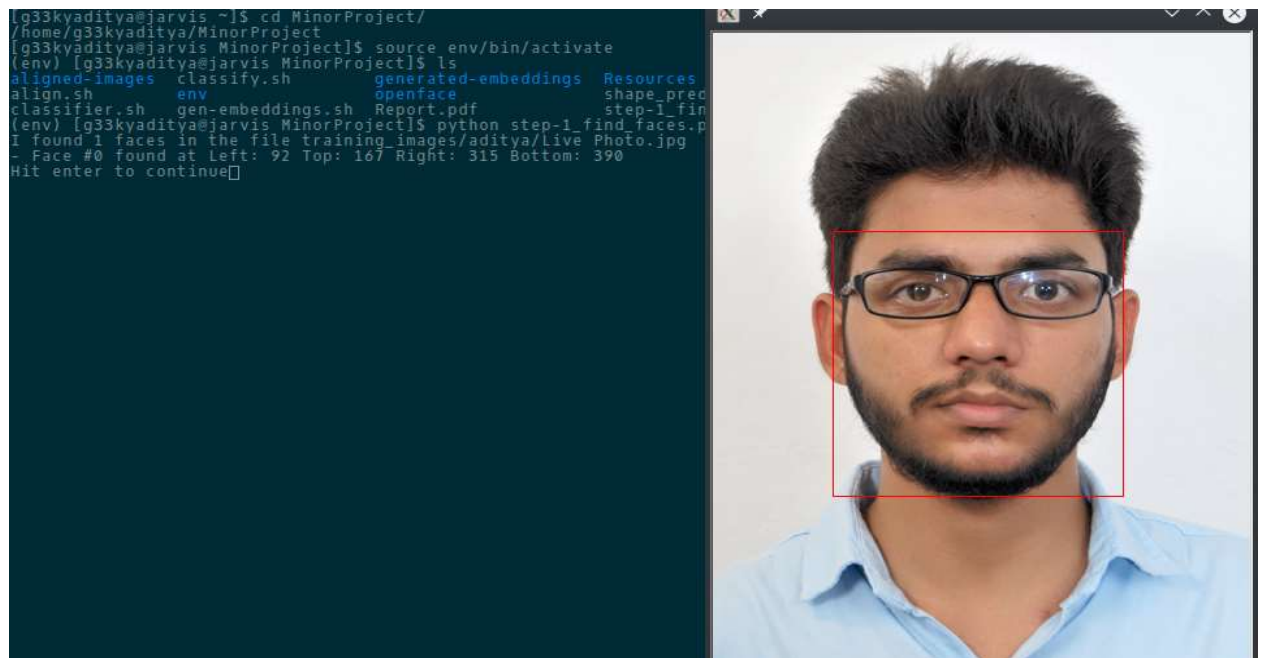


Fig 8: Detect the Face

When we train the image, detecting the face in the image is the first part that is needed for further tasks.

## Face Landmark Detection

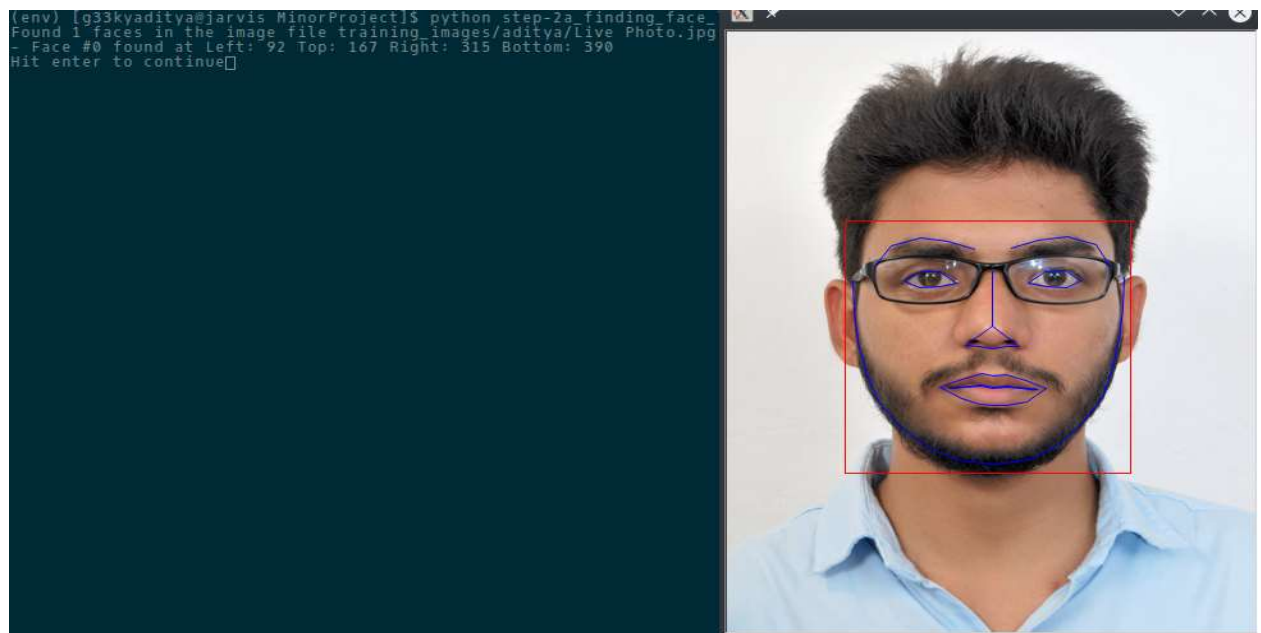


Fig 9: Face Landmark Detection

After detecting faces, we detect the landmarks in the image.

## Align the Face



Fig 10: Align the Face

This is the second step in our pipeline. Faces may not be aligned properly in an image. We as humans can perceive it easily, but for a computer to perceive a face reliably, it must be aligned.

## Generated Embeddings

```
(env) [g33kyaditya@jarvis MinorProject]$ bash gen-embeddings.sh
{
  data : ./aligned-images
  imgDim : 96
  model : /home/g33kyaditya/MinorProject/openface/models/openface/nn4.small2.v1.t7
  device : 1
  outDir : ./generated-embeddings
  cache : false
  cuda : false
  batchSize : 50
}
./aligned-images
cache location: /home/g33kyaditya/MinorProject/aligned-images/cache.t7
Creating metadata for cache.
{
  sampleSize :
  {
    1 : 3
    2 : 96
    3 : 96
  }
  split : 0
  verbose : true
  paths :
  {
    1 : ./aligned-images
  }
  samplingMode : balanced
  loadSize :
  {
    1 : 3
    2 : 96
    3 : 96
  }
}
running "find" on each class directory, and concatenate all those filenames into a single file containing all image paths for a given class
now combine all the files to a single large file
load the large concatenated list of sample paths to self.imagePath
24 samples found..... 0/24 ..... ETA: 0ms | Step: 0ms
Updating classList and imageClass appropriately
[===== 4/4 =====>] Tot: 15ms | Step: 3ms
Cleaning up temporary files
Splitting training and test sets to a ratio of 0/100
nImgs: 24
Represent: 24/24
(env) [g33kyaditya@jarvis MinorProject]$
```

Fig 11: Generated Embeddings

## Aligned Images

```
(env) [g33kyaditya@jarvis MinorProject]$ bash align.sh
=== ./training_images/aman/IMG_3850.JPG ===
=== ./training_images/aman/IMG_4201.JPG ===
=== ./training_images/aman/DSCN1625.JPG ===
=== ./training_images/akshit/DSC00130.JPG ===
=== ./training_images/akshit/DSC00352.JPG ===
=== ./training_images/aditya/Formal_Picture.JPG ===
=== ./training_images/anup/bpc - Copy.jpg ===
=== ./training_images/anup/DSC_0032.jpg ===
=== ./training_images/anup/IMG_20140913_143910.jpg ===
=== ./training_images/akshit/WhatsApp Image 2017-09-15 at 10.42.29 PM.jpeg ===
=== ./training_images/aman/DSCN1624.JPG ===
=== ./training_images/akshit/photo0353.jpg ===
=== ./training_images/akshit/IMG_20170914_094254_HDR-1-2.jpg ===
=== ./training_images/akshit/DSC00127.JPG ===
=== ./training_images/aditya/webcam-toy-photo1.jpg ===
=== ./training_images/anup/bpc3.jpg ===
=== ./training_images/anup/Pic.jpg ===
=== ./training_images/akshit/019.JPG ===
=== ./training_images/aditya/Live Photo.jpg ===
=== ./training_images/aditya/1965.jpg ===
=== ./training_images/aman/IMG_3872.JPG ===
=== ./training_images/aditya/IMG_20150917_191416.jpg ===
=== ./training_images/anup/bpc4.jpg ===
=== ./training_images/aman/IMG_3994.JPG ===
=== ./training_images/aman/IMG_3439.JPG ===
=== ./training_images/aman/IMG_3767.JPG ===
(env) [g33kyaditya@jarvis MinorProject]$
```

Fig 12: Aligned Images

## Classifier

```
(env) [g33kyaditya@jarvis MinorProject]$ bash classifier.sh
/home/g33kyaditya/MinorProject/env/lib/python2.7/site-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)
Loading embeddings.
Training for 4 classes.
Saving classifier to './generated-embeddings/classifier.pkl'
(env) [g33kyaditya@jarvis MinorProject]$
```

Fig 13: Classifier

## Classify into Different Classes

```
(env) [g33kyaditya@jarvis MinorProject]$ bash classify.sh ./test/DR.jpg
/home/g33kyaditya/MinorProject/env/lib/python2.7/site-packages/sklearn/lda.py:4: DeprecationWarning: lda.LDA has been moved to discriminant_analysis.LinearDiscriminantAnalysis in 0.17 and will be removed in 0.19
  "in 0.17 and will be removed in 0.19", DeprecationWarning)

=== ./test/DR.jpg ===
Predict anup with 0.91 confidence.
(env) [g33kyaditya@jarvis MinorProject]$
```

Fig 14: Classify into Different Makes

## GUI Prototype 1

The image shows a web browser window with the address bar displaying "127.0.0.1:8000/nets/login". The browser's bookmark bar includes links to "Apps", "interview tips", "GitHub - pradeepsh...", "Beautiful Soup docu...", "Introduction to twer...", "Getting started - tv...", "tweepy", "aricent", and "http://127.0.0.1:8000/". The main content area displays a login form with the following elements:

- Name:** A text input field with the placeholder text "Enter name".
- Phone Number:** A text input field with the placeholder text "Enter phone number".
- E-Mail:** A text input field with the placeholder text "Enter email".
- Upload File:** A section containing five "Choose File" buttons, each followed by the text "No file chosen".
- Submit:** A button located below the "Upload File" section.

Below the main form, there is a separate section with a label **Name:** and a text input field with the placeholder text "Enter circle name".

Fig 15: Graphical User Interface 1

## GUI Prototype 2

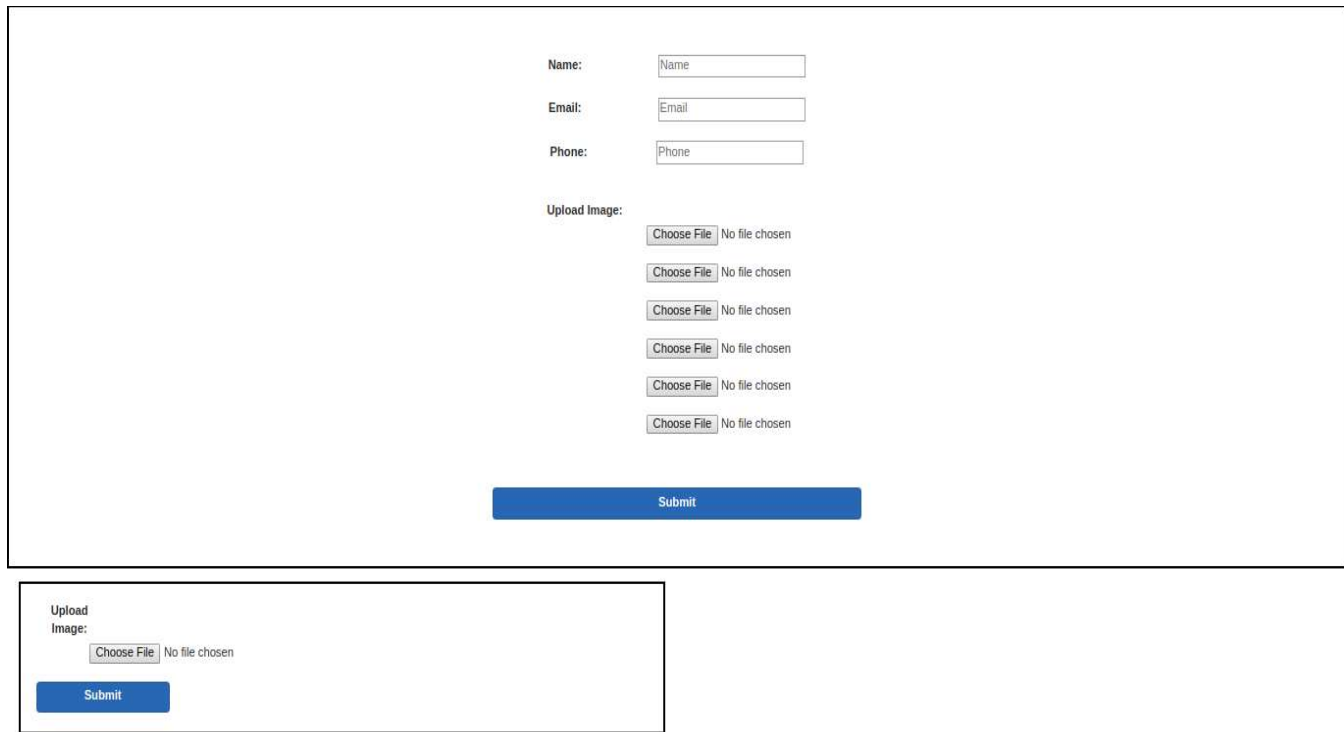
The image shows a web browser window with the address bar displaying "127.0.0.1:8000/nets/login". The browser's bookmark bar includes links to "Tide", "127.0.0.1:8000/nets/login", and "127.0.0.1:8000/nets/login". The main content area displays a login form with the following elements:

- Name:** A text input field containing the value "Aman".
- Email:** A text input field containing the value "jain.aman9464@gmail.com".
- Phone:** A text input field containing the value "987545651321".
- Upload Image:** A section containing a "Choose file" button followed by the text "DSCN1553.JPG".
- Submit:** A blue button located below the "Upload Image" section.

The Windows taskbar at the bottom of the screen shows the search bar with the text "Type here to search", several application icons (including Edge, Mail, and File Explorer), and the system tray displaying the language "ENG", the time "10:47 AM", and the date "10/13/2017".

Fig 16: Graphical User Interface 2

## A Complete GUI



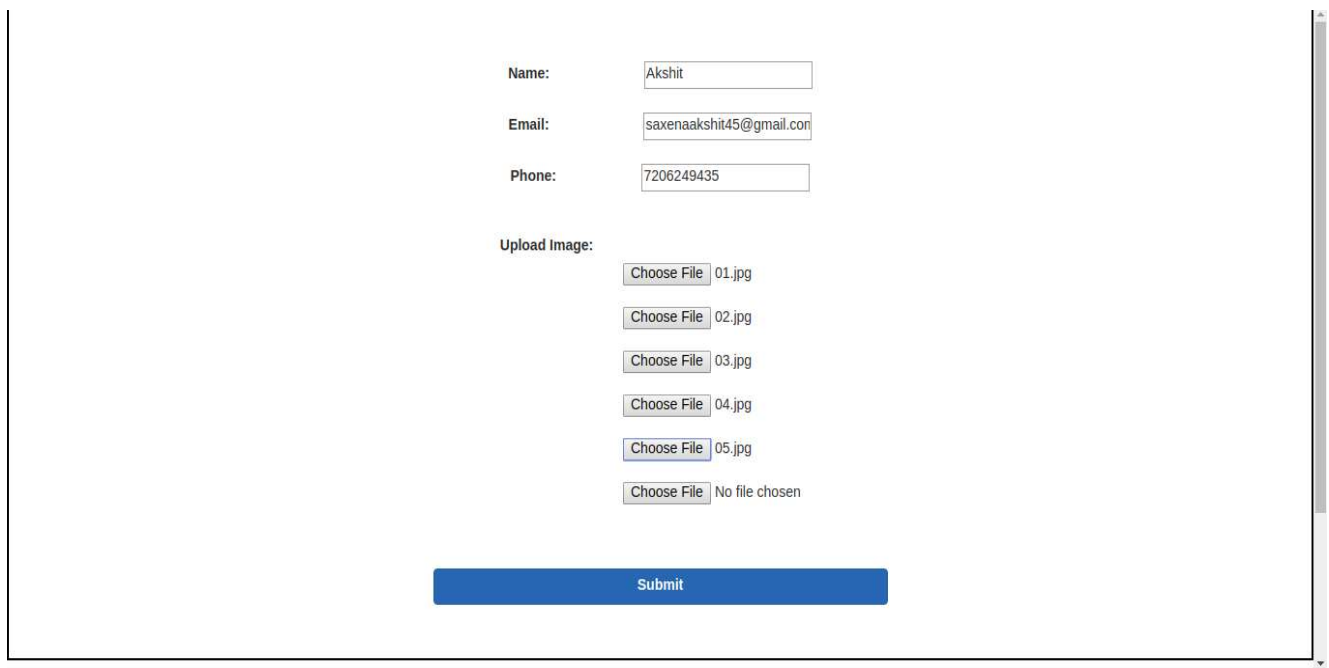
The figure shows a web form titled "A Complete GUI". It contains the following elements:

- Name:** A text input field with the placeholder text "Name".
- Email:** A text input field with the placeholder text "Email".
- Phone:** A text input field with the placeholder text "Phone".
- Upload Image:** A section with six "Choose File" buttons, each followed by the text "No file chosen".
- Submit:** A large blue button at the bottom center.

Below the main form, there is a smaller, identical version of the form, also titled "A Complete GUI", which includes the same input fields and buttons.

Fig 17: Complete GUI

## Adding Images



The figure shows a web form titled "Adding Images". It contains the following elements:

- Name:** A text input field with the value "Akshit".
- Email:** A text input field with the value "saxenaakshit45@gmail.com".
- Phone:** A text input field with the value "7206249435".
- Upload Image:** A section with six "Choose File" buttons. The first five buttons are followed by the file names "01.jpg", "02.jpg", "03.jpg", "04.jpg", and "05.jpg" respectively. The sixth button is followed by the text "No file chosen".
- Submit:** A large blue button at the bottom center.

Fig 18: Adding Images

### Verification Add Images

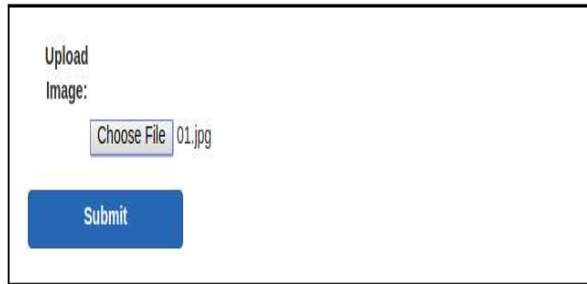


Fig 19: Uploading verification image

### Verification False Result

**Could not find a match for the person**

Fig 20: Verification False Result

### Verification True Result

Name	Phone	Email
Aditya	+91-7376109415	adsharmarocks@gmail.com

\*With a 82.0% confidence\*

Fig 21: Verification True Result

## 5.3 Back Ends Representation

### 5.3.1 Snapshot of Database table with Brief Description

#### DataSet

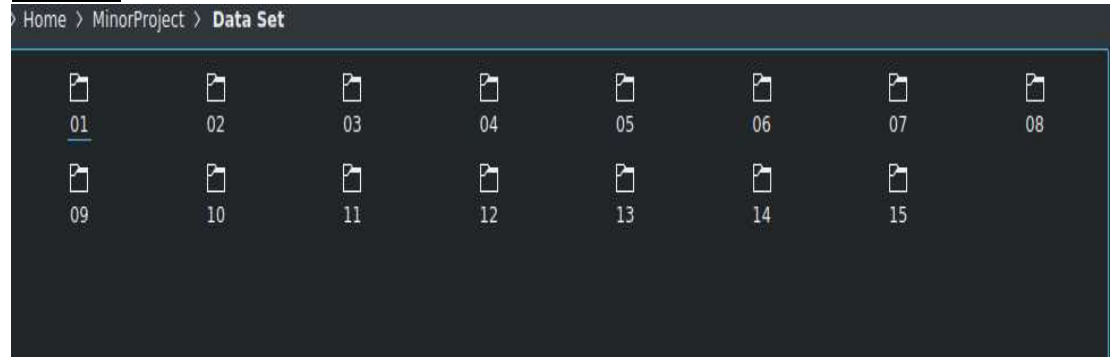


Fig22. DataSet

#### Sqlite DB

The screenshot shows a SQLite database viewer interface with tabs for 'Full View', 'Item View', and 'Script Output'. The 'Full View' tab is selected, displaying a table with the following data:

	phone	email	pic2	pic	pic4	pic3
1	+91-7376109415	adsharmarocks@gmail.com	images/02_tz5QB2X.jpg	images/01_cIFxA9.jpg	images/04_b2AsDDM.jpg	images/03_jvVmVBT.jpg
2	+91-7508582879	pratiksha@yahoo.com	images/03_mOw0Wzk.jpg	images/01_qxSfjyu.jpg	images/05_CeRWbS5.jpg	images/04_SlR8Pt5.jpg
3	+91-8219990196	chaitanyakadian@gmail.com	images/02_vVBtmOt.jpg	images/01_Lw2bT5E.jpg	images/04_3ewCS1s.jpg	images/03_bugPlnS.jpg
4	+91-9646261680	arshdeep@gmail.com	images/02_lKjwC3.jpg	images/01_vAt3wKs.jpg	images/04_ryzlGtd.jpg	images/03_0E2H4eQ.jpg

Fig23. Sqlite DB

#### Sqlite DB Verify

The screenshot shows a SQLite database viewer interface with tabs for 'Full View', 'Item View', and 'Script Output'. The 'Full View' tab is selected, displaying a table with the following data:

	id	pic_verification
1	1	images/IMG_20171213_201032390.jpg
2	2	images/IMG_20171213_201032390_lwb4Yxl.jpg
3	3	images/IMG_20171213_201032390_aurGHq8.jpg
4	4	images/me.jpg
5	5	images/me_Q3piF9W.jpg
6	6	images/arsh.jpg

Fig24. Sqlite DB Verify

## **6. Conclusion and Future Scope**

### ***6.1 Conclusion***

The backend image recognition pipeline works as expected. We trained a sample neural network for generating embeddings with a small data set to test the working. Furthermore, we will use a neural net by the openface team led by Brandon Amos (Carnegie Mellon University) for future workings. The backend needs to be connected to the frontend for which we intend to use Django Web Framework. We are learning it in the time being.

### ***6.2 Future Scope***

- In our project, we have a lot of loopholes like it cannot be used to detect the same person but with images of different age. When we use the pics such that a person with an image of age 10 and the same person with the age 25 will have his/her facial features changed a lot and our project fails in matching the person. So, in future we would be working on detecting the person with different age.
- Our project still have another loophole that is it is not able to differentiate between the twins. If the twins are lost and we add images to training set to classify them into different classes, but when we add an image to identify the missing twins. It would not be able to differentiate between the twins classes. So in the future we will try to change the program such that the twins could be verified.
- At the end we use a threshold value to check whether the confidence value is greater than the threshold then it can show the details of the particular class as the result but if the confidence value is less than the threshold it would not classify it. The problem is that we are choosing a random variable as the threshold. In the future we will be adding an algorithm to choose the threshold value.



## **References**

- [1]. <https://cmusatyalab.github.io/openface/>
- [2]. <https://www.learnopencv.com/tag/dlib/>
- [3]. [https://docs.opencv.org/trunk/d9/df8/tutorial\\_root.html](https://docs.opencv.org/trunk/d9/df8/tutorial_root.html)
- [4]. <https://thenewboston.com/page.php?pid=2951>
- [5]. <https://conda.io/docs/user-guide/getting-started.html>
- [6]. <http://www.codesofinterest.com/2016/11/setting-up-keras-anaconda-ubuntu.html>