CHAPTER 1. INTRODUCTION

1.1 Background:

Face detection is the most well-known space of exploration in the vision of computer engineering. It has become an exploring domain in current years due to the increase in security demands and its law implementation application. It is one of the major talked domains in computer technology which is being used in a variety of applications that recognizes (identifies) human faces in digital images. It is used for the automatic detection of a person with any images or video frame. Image classification involves activities such as predicting the class of one object in an image. It is required an hour to come up with a fast detection, efficiency, and tracking algorithm. When a user refers to the term "object recognition" they often mean "object detection". Object recognition refers to a collection of related tasks for identifying objects in digital photographs. Face recognition is a method of identifying or verifying the identity of an individual using their face. Face recognition systems can be used to identify people in photos, video, or in real-time.

Humans can detect and identify objects present in an image. The human visual system is fast and accurate and can also perform complex tasks like identifying multiple objects and detect obstacles with little conscious thought. With the availability of large sets of data, faster GPUs, and better algorithms, we can now easily train computers to detect and classify multiple objects within an image with high accuracy. Object detection is always more challenging and combines these two tasks and draws a bounding box around each object of interest in the image and assigns them a class label. Together, all these problems are referred to as object recognition.

Localization of human faces is considered as the primary and the initial stage in the study of face detection. It identifies the facial features by extracting features from an image of the subject's face and analyze the relative position, size, shape of eyes, nose, jaw, and cheekbones of the detected person. It is also referred to as the extraction of facial features using a recognition system. The resultant features are utilized to search for corresponding matching features in other images. OpenCV (Open-Source Computer Vision Library) can be used for creating such models, prototypes, and systems. Facial recognition systems span across several applications like Biometric systems, Psychology, Emigration checking (Passport checking), Banking, E-voting domain, and the gaming industry. A typical approach in face recognition is video content-based

searching. A retrieval system should return all videos containing specific actors upon a client's request.

For example, In YouTube, where a cast list may not be accessible, the visual content plays a significant role in achieving this task successfully. But the main disadvantages are the availability of annotated video face tracks. The use of the minimization way in this fashion is very computationally expensive. Other common issues in face detection systems are illumination changes between images, pose variation, occlusion (obstruction), and lighting conditions. This detecting and tracking algorithm will bring practical applications such as Smart captcha, Time tracking service, Outdoor surveillance camera, Teleconferencing, and Video chat service.

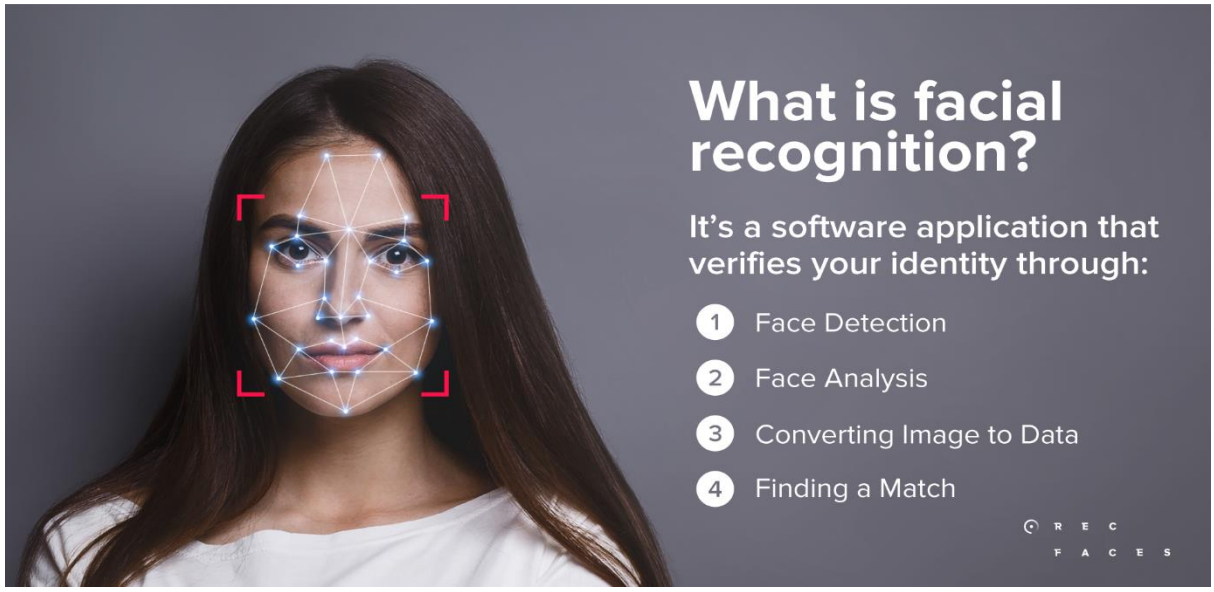


Figure1. Facial Based Recognition

1.2 Objectives:

1) The objective of the “Suspect Tracking System” is to detect and recognize the suspect from the frame.
2) Facial recognition is a way of using software to determine the similarity between two face images in order to evaluate a claim.
3) The proposed algorithm consists of three intermediate steps, the first step is to take an image of a person by using “Take an image” which captures the suspect images very fast.
4) The second step deals with training the captured images in step one using “Train image”. The third step deals with tracking the image called “Track image”, which classifies the images and tracks the suspect, and generates a report in CSV file format.

1.3 Purpose, Scope, and Applicability:

1.3.1 Purpose:

The main purpose of this project is to build an application that will help us to track the suspects. Facial recognition technology has been used countless times to identify, interdict, and capture criminal suspects. When a suspect gets caught in the camera, the report will be generated in excel file format. Thus, we can locate the suspect and the police can easily track the suspect. This system will reduce the workload for police. It provides an efficient way of tracking a suspect.

1.3.2 Scope:

The application provides an efficient "Suspect tracking system" along with tracking of the suspect from a real-time video. For this to it uses a Haar Cascades algorithm for object detection. The system finds its application at places where real-time Video Surveillance is required. It is used in Police stations, Hotels, Houses, and Colleges, etc. Managing the details of the suspect is also be efficient.

Limitations:

The following assumptions are to be made before using the developed system:

1) The background of the image that is given as the input to the system should remain the same for most of the time in case of filtering but it can be overcome by DWT (Discrete wavelet transform).
2) The size of the template of the object that is to be detected should be less than the actual object.
3) Proper illumination conditions should always be present.
4) There should not be any drastic change in the motion or light of room for the suspect to be detected.

1.3.3 Applicability:

1) The system is proposed to help agencies like CBI, CID and other such Bureau's to speed up investigation process and track status of multiple cases at a time.
2) End-users such as police station owners use it for identifying, controlling, and managing the suspect. Here I have used the Haar Cascade algorithm that trained the images and recognize them after that.
3) The software will be very useful for government agencies. The software allows for team work on solving complicated cases.

1.4 Achievements:

Based on the face you can identify the suspect. The system can be used by CBI, CID and other government agencies in several countries. You can achieve or detect the suspect that caught into the camera. It identifies the facial features from an image and analyses the relative position, shape and creates 60 images of 1 image. It trained the image and after that Track the People that value is the match from the image and generate CSV file in a specified folder with the Id, name, and time.

1.5 Organization of Report:

- In chapter 2, I am going to explain the technologies that I have used.
- In chapter 3, Planning and scheduling will be explained with the help of a block diagram, PERT Chart. This chapter also includes process models.
- In chapter 4, Procedural diagrams related to quick reports will be explained. This includes Data Flow Diagram.
- In chapter 5, I will explain the implementation of code along with the testing phases.
- In chapter 6, I will create test reports and User documentation.
- In chapter 7, I will explain the conclusion, significance, limitations, and future scope of the system.

CHAPTER 2. SURVEY OF TECHNOLOGIES

2.1 Problem statement-

1) This project base on Facial recognition is a way of using software to determine the similarity between two face images to evaluate a claim.
2) For this too I am using a Haar Cascades algorithm for object detection. The system finds its application at places where real-time Video Surveillance is required.
3) It is used in Police stations, Hotels, Houses, and Colleges, etc. Managing the details of the suspect is also be efficient.
4) Recognized the suspect and reduced the work of CBI, police station, etc.

2.2 Methodology-

2.2.1 Research Methodology-

Proposed image pre-processing techniques for face recognition using OpenCV-

Face recognition is limited mostly by light and poses factors. Often, input images needed to be correct to achieve efficient face recognition. This research discusses the enhanced pre-processing techniques such as illumination, pose, and illumination pose when combined with Eigen's face, fisher face and LBPH face recognition algorithms available in OpenCV. To identify the performance of combinations, relevant speed, identification rate, and threshold were measured. Experimental research was employed. Frontal faces from Yale Face Database and 20 individuals' faces were used as subjects.

The study found that there is a statistically significant difference among speed and threshold level of algorithms with the enhanced pre-processing techniques. While there is no statistically significant difference among identification rates. Further, a significant interaction effect in terms of speed and threshold level among face recognition algorithms and the enhanced pre-processing techniques is evident. While there is no significant interaction effect in identification rate.

2.2.2 Database-

My purpose is to make a facial recognition system that needs as little training data as possible. The main reason behind this constraint is the fact that it is more useful for a supervisor to have trained the model with one or few pictures for each student rather than having to make a large

Dataset with many images for the same person. We will be comparing two main face classification models, PCA dimensionality reduction, and pretrained CNNs. To perform face recognition, the following steps will be followed:

- Detecting all faces included in the image (face detection).
- Cropping the faces and extracting their features.
- Applying a suitable facial recognition algorithm to compare faces with the database of students and lecturers.
- Providing a file recording of the identified attendants.

2.2.3 Web-app Frontend-

For creating frontend and backend I am used Python. In this, I have used the OpenCV library of python that provides better recognition.

2.2.4 Web-app Backend-

In the backend, the Haar cascade algorithm is used. Haar cascade classifier is a machine learning object detection program that identifies objects in an image and video. A Haar feature is essentially calculations that are performed on adjacent rectangular regions at a specific location in a detection window.

2.2.5 Machine Learning-

The industry around facial recognition technology is rapidly maturing due to advances in AI, ML, and deep learning technologies. ML has defined the programming of the algorithm to learn from experience. Facial recognition is a technology that is capable of recognizing a person based on their face. It employs machine learning algorithms that find, capture, store and analyze facial features to match them with images of individuals in a pre-existing database.

2.3 Proposed System-

1. Face Recognition using Histogram of Oriented-

Various methods or experiments can be used for face recognition and detection however two of the main include an experiment that evaluates the impact of facial landmark localization in the face recognition performance and the second experiment evaluates the impact of extracting the HOG features from a regular grid and at multiple scales. Here the study is to question feature sets for robust visual object recognition. The Histogram of Oriented Gradients significantly outperforms other existing methods like edge and gradient-based descriptors. We study the influence of each stage of the computation on performance, concluding that fine-scale gradients, fine orientation binning, relatively coarse spatial binning, and high-quality local contrast normalization in overlapping descriptor blocks are all important for good results. Comparative experiments show that though HOG is a simple feature descriptor, the proposed HOG feature achieves amazing results with much lower computational time.

2. Implementation of Face Recognition algorithm for Biometrics based Time attendance system-

Face Recognition begins with extracting the coordinates of features such as the width of the mouth, the width of eyes, pupil, and compare the result with the measurements stored in the database, and return the closest record. Nowadays, there are a lot of face recognition techniques and algorithms found and developed around the world. Facial recognition becomes an interesting research topic. It is proven by numerous numbers of published papers related to facial recognition including facial feature extraction, facial algorithm improvements, and facial recognition implementations. The main purposes of this research are to get the best facial recognition algorithm provided by the Open CV by comparing the ROC (Receiver Operating Characteristics) curve and implement it in the attendance system as the main case study. Based on the experiments, the ROC curve proves that using the current training set, Eigen's face achieves better results than Fisher's face. Eigen face implemented inside the Attendance System returns between 70% to 90% similarity for genuine face images.

3. Face Recognition Based on a local binary pattern-

Facial analysis has been an important research field due to its wide range of applications like law enforcement, surveillance, entertainment like video games and virtual reality, information security, banking, human-computer interface, etc. The original interest in facial analysis relied on

face recognition, but later on the interest in the field was extended and research efforts were focused on the appearance of a model-based image, video coding, face tracking, pose estimation, facial expression, emotion analysis, and video indexing.

Face detection and recognition are still a very difficult challenge and there is no unique method that provides an efficient solution to all situations face processing may encounter. In this paper, a novel approach is presented to face recognition which considers both shape and texture information to represent the face. The face area is first divided into small regions from which Local Binary Pattern (LBP) histograms are extracted and concatenated into a single, spatially enhanced feature histogram efficiently representing the face image. Extensive experimental research proves the superiority of the proposed method in respect of its simplicity and efficiency in very fast feature extraction.

4. Face Recognition with partial face recognition and convolutional neural network-

Biometrics is a system in which we used to recognize humans based on their physical or behavioral characteristics. Face recognition has been a dynamic research area in the pattern recognition and computer vision domains. Each face in this world has uniqueness. Therefore, it is an identity of humans. This identity due to its uniqueness can be used for authentication and access control in different applications. In this review paper, the face recognition and facial feature is a key area of investigation. Therefore, this paper includes a survey of different recently developed face recognition techniques and methods that are claimed to provide an effective and accurate method of face recognition. In addition to that, a new model for recognizing a face is also introduced in this paper. That model is implemented in near future and their performance is compared with a similar approach.

2.4 Benefits-

1. The objective of the “Suspect Tracking System” is to detect and recognize the suspect from the frame.
2. We can detect the suspect and the police can easily track the suspect. This system will reduce the workload for police. It provides an efficient way of tracking a suspect.
3. The system is proposed to help agencies like CBI, CID, and other such Bureau’s to speed up the investigation process and track the status of multiple cases at a time.

2.5 Summarized Definition-

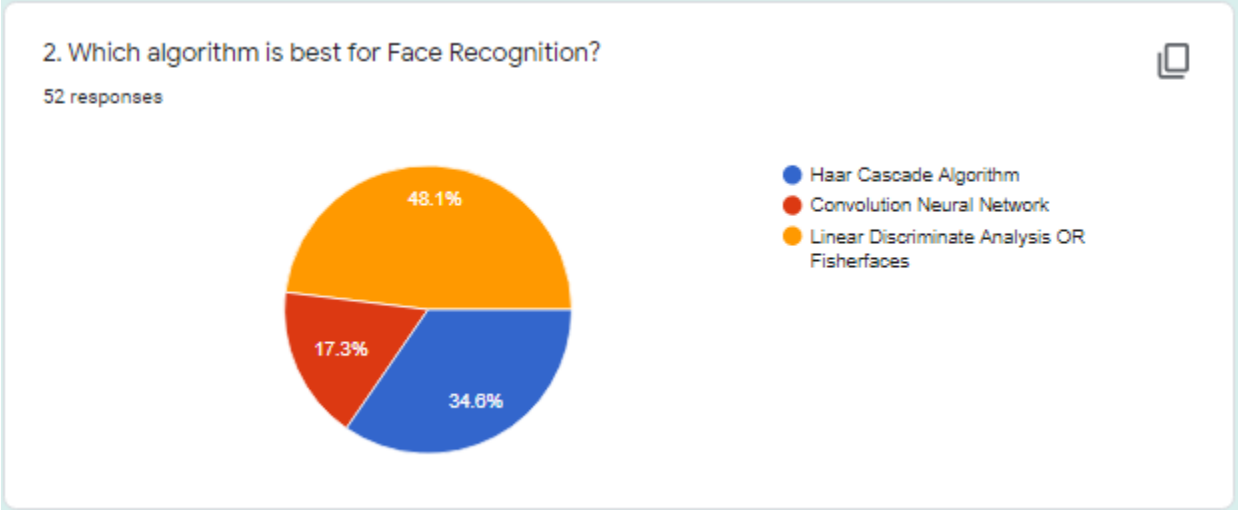
In this project, I recognize the suspect based on their face. The face is caught into the camera and they identify the suspect person. The proposed algorithm consists of three intermediate steps, the first step is to take an image of a person by using “Take an image” which captures the suspect images very fast. The second step deals with training the captured images in step one using “Train image”. The third step deals with tracking the image called “Track image”, which classifies the images and tracks the suspect, and generates a report in CSV file format.

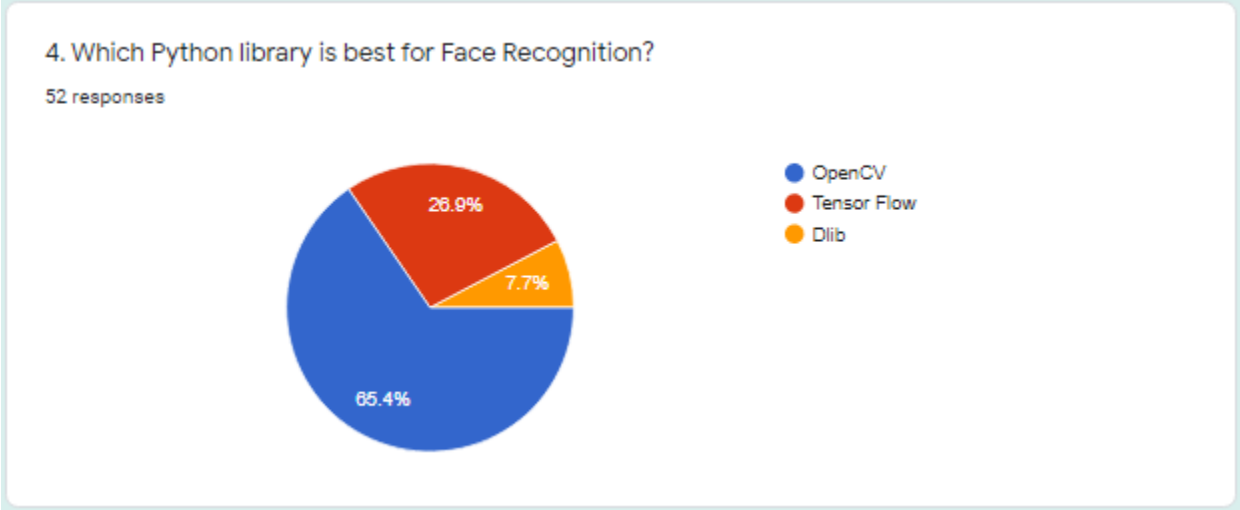
CHAPTER 3. RESEARCH KNOWLEDGE & LITERATURE REVIEW

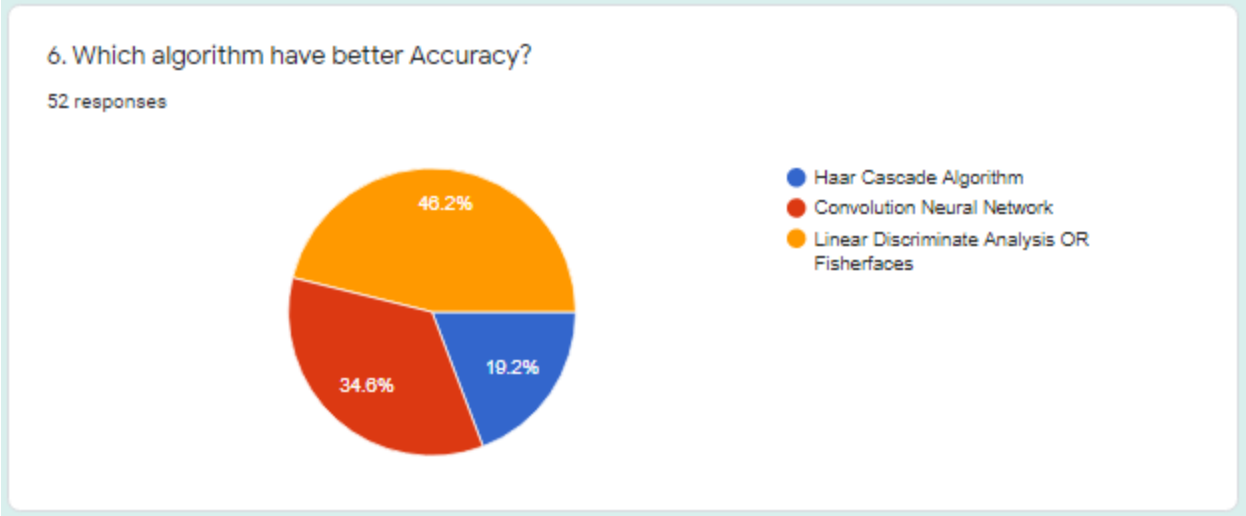
3.1 Primary Research-

- **Interviews:**

I have approached convenient way of collecting information from individuals or small groups of people. Due to COVID-19 I couldn't arrange any physical location for conducting an interview so I decided to take it virtually. I have approached some Thesis guides, PhD scholars, and faculties to ask them what's their perceptions on suspect tracking system.







- **Observation:**

  This research method is useful for gaining knowledge without the biased viewpoint sometimes present in interviews. I started observing people, occurrences, and other variables important to the study or research.

- **Focus Group:**

  I have made a group 8-10 people who had prior knowledge of suspect tracking system. This qualitative data gathering method was very useful because people from group has given their perception as well as guidance to understand the nature of gain the deeper knowledge of suspect tracking system.

3.1.1 Existing System-

Some facial recognition algorithms identify faces by extracting landmarks or features from an image. For example, an algorithm may analyze the relative position, size, and/or shape of the eyes, nose, cheekbones and jaw. These features are then used to search for other images with matching features. Other algorithms normalize a gallery of face images and then compress the face data, only saving the data in the image that is useful for face detection. A probe image is then compared with the face data. One of the earliest successful systems is based on template matching techniques applied to a set of salient facial features, providing a sort of compressed face representation.

Linear Discriminate Analysis (LDA) or Fisherfaces-

LDA is a method to find a linear combination of features which characterize or separate two or more classes of objects or events. The resulting combination may be used as a linear classifier. In computerized face recognition, each face is represented by a large number of pixel values. Linear discriminant analysis is primarily used here to reduce the number of features to a

more manageable number before classification. Each of the new dimensions is a linear combination of pixel values which form a template.

3.2 Secondary Research-

While conducting primary research I also continued the secondary research as it was really important to know the other side as well.

- Academic peer-reviewed journal.
- Published book and articles.
- Educational institutions.
- Commercial information sources.

I have read at least 5-6 paper for finding the useful information and qualitative data. Below is the same information which I used during our secondary research.

1) "Criminal Investigation Tracker with Suspect Identification", Mrs. S. Puvaneswati, Keerthana B, Pooja E.
2) "Evaluation of Haar Cascade Classifier for Face Detection", Rafael Padilla, Marly Costa.
3) https://towardsdatascience.com/face-detection-with-haar-cascade-727f68dafd08#:~:text=So%20what%20is%20Haar%20Cascade,Simple%20Features%E2%80%9D%20published%20in%202001.

3.2.1 Haar Cascade-

Object Detection using Haar feature-based cascade classifiers is an effective object detection method. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. First, a classifier is trained with a few hundred sample views of a particular object called positive examples that are scaled to the same size and negative examples - arbitrary images of the same size. After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a "1" if the region is likely to show the object, and "0" otherwise. To search for the object in the whole image one can move the search window across the image and check every location using the classifier.

The classifier is designed so that it can be easily "resized" in order to be able to find the objects of interest at different sizes, which is more efficient then resizing the image itself. So, to

find an object of an unknown size in the image the scan procedure should be done several times at different scales. The word “cascade” in the classifier name means that the resultant classifier consists of several simpler classifiers (*stages*) that are applied subsequently to a region of interest until at some stage the candidate is rejected or all the stages are passed. The word “boosted” means that the classifiers at every stage of the cascade are complex themselves and they are built out of basic classifiers using one of four different boosting techniques (weighted voting).

Currently Discrete Adaboost, Real Adaboost, Gentle Adaboost and Logitboost are supported. The basic classifiers are decision-tree classifiers with at least 2 leaves. Haar-like features are the input to the basic classifiers, and are calculated as described below. The current algorithm uses the following Haar-like features: The feature used in a particular classifier is specified by its shape (1a, 2b etc.), position within the region of interest and the scale (this scale is not the same as the scale used at the detection stage, though these two scales are multiplied).

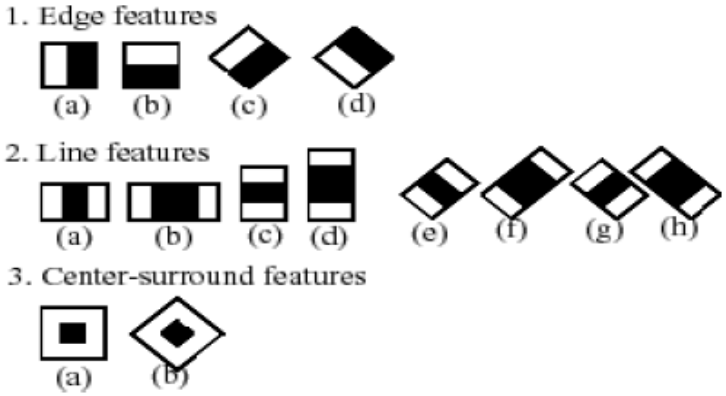


Figure 3.2.1. **Haar-Cascade Classifier Models**

For example, in the case of the third line feature (2c) the response is calculated as the difference between the sum of image pixels under the rectangle covering the whole feature (including the two white stripes and the black stripe in the middle) and the sum of the image pixels under the black stripe multiplied by 3 to compensate for the differences in the size of areas. The sums of pixel values over rectangular regions are calculated rapidly using integral images

3.2.2 Face Recognization with face recognition and convolutional neural networks [Shraddha Arya, Arpit Agrawal]

Biometrics is a system in which we used to recognize humans based on their physical or behavioral characteristics. Face recognition has been a dynamic research area in the pattern recognition and computer vision domains. Each face in this world has uniqueness. Therefore, it is

an identity of humans. This identity due to its uniqueness can be used for authentication and access control in a different application. In this review paper, the face recognition and facial feature is a key area of investigation. Therefore, this paper includes a survey of different recently developed face recognition techniques and methods that are claimed to provide an effective and accurate method of face recognition. In addition to that, a new model for recognizing a face is also introduced in this paper. That model is implemented in near future and their performance is compared with a similar approach.

3.3 Technology-

3.3.1 Python-

Python is an open-source programming language that was made to be easy-to-read and powerful. Python is an interpreted language. Interpreted languages do not need to be compiled to run. A program called an interpreter runs Python code on almost any kind of computer. This means that a programmer can change the code and quickly see the results. This also means Python is slower than a compiled language like C, because it is not running machine code directly. It is a high-level language, which means a programmer can focus on what to do instead of how to do it. Writing program in Python takes less time than in some other languages.

Python has very easy-to-read syntax. Python is used by hundreds of thousands of programmers and is used in many places. Sometimes only python code is used for a program, but most of the time it is used to do simple jobs while another programming language is used to do more complicated tasks. Its standard library is made up of many functions that come with Python when it is installed. On the internet there are many other libraries available that make it possible for the Python language to do more things. These libraries make it a powerful language; it can do many different things. Some things that Python is often used for are:

1) Web development
2) Scientific programming
3) Desktop GUIs
4) Network programming

5) Game programming.

Python's development is conducted largely through the Python Enhancement Proposal (PEP) process, the primary mechanism for proposing major new features, collecting community input on issues and documenting Python design decisions. Python's public releases come in three types, distinguished by which part of the version number is incremented: Backward-incompatible versions where code is expected to break and need to be manually ported. The first part of the version number is incremented. These releases happen infrequently-for example, version 3.0 was released 8 years after 2.0. Major or "feature" releases, about every 18 months, are largely compatible but introduce new features. The second part of the version number is incremented. Each major version is supported by bug fixes for several years after its release.

3.3.2 OpenCV-

OpenCV (Open-source computer vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez. The library is cross-platform and free for use under the open-source BSD license. The library is cross-platform and free for use under the open-source BSD license. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe. The first alpha version of OpenCV was released to the public at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. A version 1.1 "pre-release" was released in October 2008. OpenCV runs on the following desktop operating system: Window, Linux, macOS etc.

3.3.3 NumPy-

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

Inserting or appending entries to an array is not as trivially possible as it is with Python's lists. The np.pad(...) routine to extend arrays actually creates new arrays of the desired shape and padding values, copies the given array into the new one and returns it. NumPy's np.concatenate([a1,a2]) operation does not actually link the two arrays but returns a new one, filled

with the entries from both given arrays in sequence. Reshaping the dimensionality of an array with np.reshape(...) is only possible as long as the number of elements in the array does not change. These circumstances originate from the fact that NumPy's arrays must be views on contiguous memory buffers. A replacement package called Blaze attempts to overcome this limitation.

CHAPTER 4. DESCRIPTION OF METHODOLOGY

4.1 Problem Decision-

This project base on Facial recognition is a method of utilizing software to determine the similarity between two face images to evaluate a claim. It is used in Police stations, Hotels, Houses, and Colleges, etc. Managing the details of the suspect is also be efficient. For this too I am using a Haar Cascades algorithm for object detection. The system finds its application at places where real-time Video Surveillance is required. Recognized the suspect and reduced the work of CBI, police station, etc. It reduced the time for the identification and easily you can identify them.

4.2 Future Strategy-

Here I propose a system investigation tracker system that tracks the investigation status of criminal cases with logs and also predicts primary suspects. The system is proposed to help agencies like CBI, CID and other such bureaus to sped up investigation process and track status of multiple cases at a time. The system keeps logs of a case which includes people involved, past criminal history of those involved and other details. In future I am try to create system, realizes the type of case, allows admin to update the status of investigation, upload more images of crime, items found on scene etc.

4.3 Planning and Scheduling-

GANTT CHART:

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale. Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. This allows you to see at a glance:

- When each activity begins and end?
- How long each activity is scheduled to last?

- Where activities overlap with each other activities, and by how much?
- The start and the end of the project?
- What are the various activities?

Originally Gantt charts were prepared laboriously by hand; each time a project changed it was necessary to amend or redraw the chart and this limited their usefulness, continual change being a feature of most projects. Nowadays, however, with the advent of computers and project management software, Gantt charts can be created, updated and printed easily. Today, Gantt charts are most commonly used for tracking project schedules.

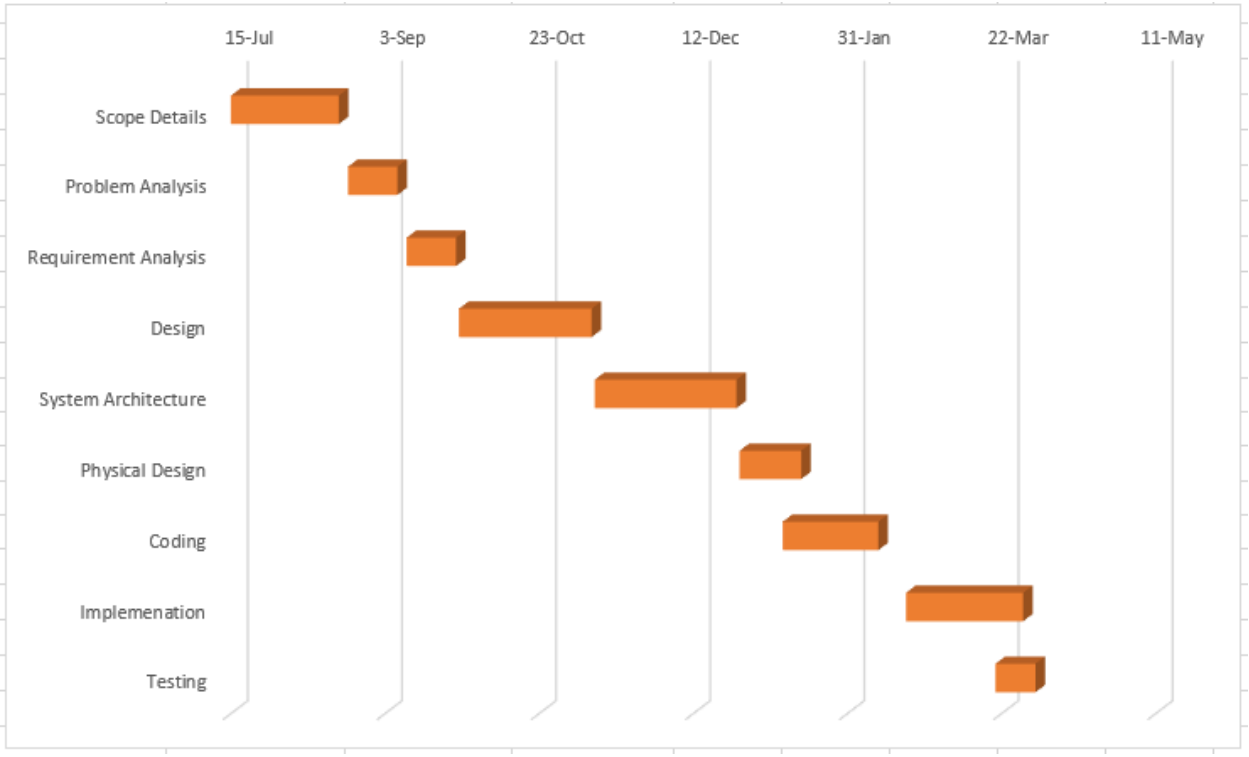


Figure 4.1 Gantt Chart

PERT CHART:

A PERT chart is a project management tool that provides a graphical representation of a project's timeline. PERT (Program Evaluation Review Technique) breaks down the individual tasks of a project for analysis. Although PERT charts are preferable to Gantt charts because they identify task dependencies, PERT charts are often more difficult to interpret.

Interpreting PERT Charts, A PERT chart is a visual representation of a series of events that must occur within a project’s lifetime. The direction of arrows indicates the flow and sequence of events required for project completion. Dotted activity lines represent dummy activities, which are

items located on another PERT path. Numbers and time allotments are assigned and shown inside each vector.

Benefits of PERT Charts:

A PERT chart allows managers to evaluate the time and resources required to manage a project. This includes the ability to track assets needed during any stage of production in the course of the entire project. PERT charts are useful for what-if analyses. Understanding the possibilities concerning the flow of project resources and milestones allows management to achieve the most efficient and useful project path. The time are given below.

- 15-Jul-21 Scope Details
- 22-Aug-21 Problem Analysis
- 10-Sep-21 Requirement Analysis
- 27-Sep-21 Design
- 10-Nov-21 System Architecture
- 27-Dec-21 Physical Design
- 18-Jan-22 Coding
- 19-Feb-22 Implementation
- 20-Mar-22 Testing

Disadvantages of PERT Charts:

1. The use of a PERT chart is highly subjective and subject to management's experience. For this reason, PERT charts may include unreliable data or unreasonable estimations for cost or time.
2. PERT charts are deadline-focused and may not fully communicate the financial positioning of the project.
3. Finally, because a PERT chart is labour intensive (large amount of work) the establishment and maintenance of the information require additional time and resources. Continual review of the information provided, as well as prospective positioning of the project, is required for PERT charts to be valuable.

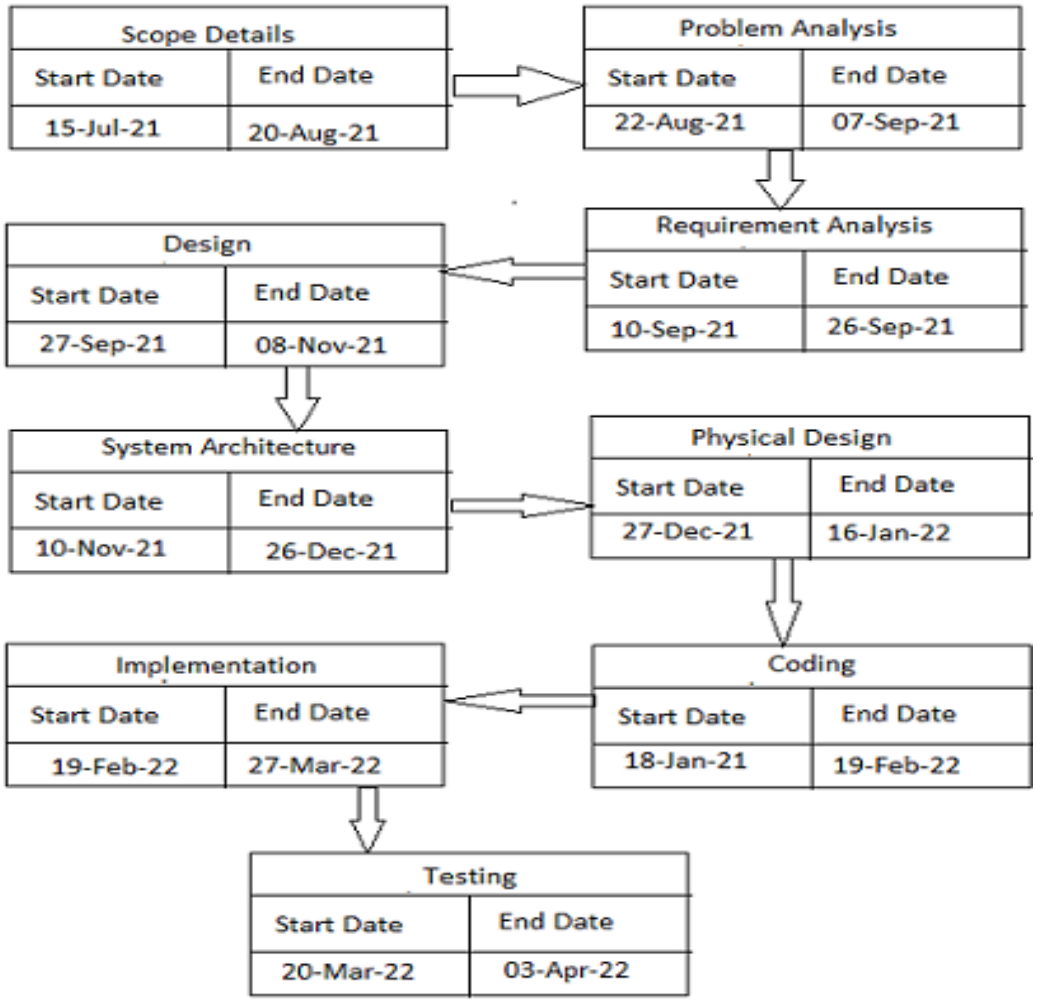


Figure 4.2 Pert Chart Diagram

4.4 Preliminary Product Description-

In this project, I create a system that identify suspect based on their face. The face is caught into the camera and they identify the suspect person that already present. The proposed algorithm consists of three intermediate steps, the first step is to take an image of a person by using “Take an image” which captures the suspect images very fast. The second step deals with training the captured images in step one using “Train image”. The third step deals with tracking the image called “Track image”, which classifies the images and tracks the suspect, and generates a report in CSV file format.

4.5 Conceptual Design-

4.5.1. Model:

Waterfall Model:

We are using waterfall model. The waterfall model is a breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks. Each phase is designed for performing specific activity during SDLC phase. It was introduced in 1970 by Winston Royce. The approach is typical for certain areas of engineering design. In software development, it tends to be among the less iterative and flexible approaches, as progress flows in largely one direction ("downwards" like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, deployment and maintenance.

The waterfall development model originated in the manufacturing and construction industries; where the highly structured physical environments meant that design changes became prohibitively expensive much sooner in the development process. When first adopted for software development, there were no recognized alternatives for knowledge-based creative work. You used these because of this.

- Requirements are not changing frequently
- Application is not complicated and big
- Project is short
- Requirement is clear
- Environment is stable
- Technology and tools used are not dynamic and is stable
- Resources are available and trained.

Different phases	**Activities performed by each stages**
Requirement Gathering stage	During this phase, detailed requirements of the software system to be developed are gathered from client
Design Stage	Plan the programming language, for Example Java, C++, .net or database like Oracle, MySQL, etc. Or other high-level technical details of the project
Coding Stage	After design stage, it is built stage, that is nothing but coding the software

Test Stage	In this phase, you test the software to verify that it is built as per the specifications given by the client.
Deployment stage	Deploy the application in the respective environment
Maintenance stage	Once your system is ready to use, you may later require change the code as per customer request

Table 4.1 Phase of waterfall model

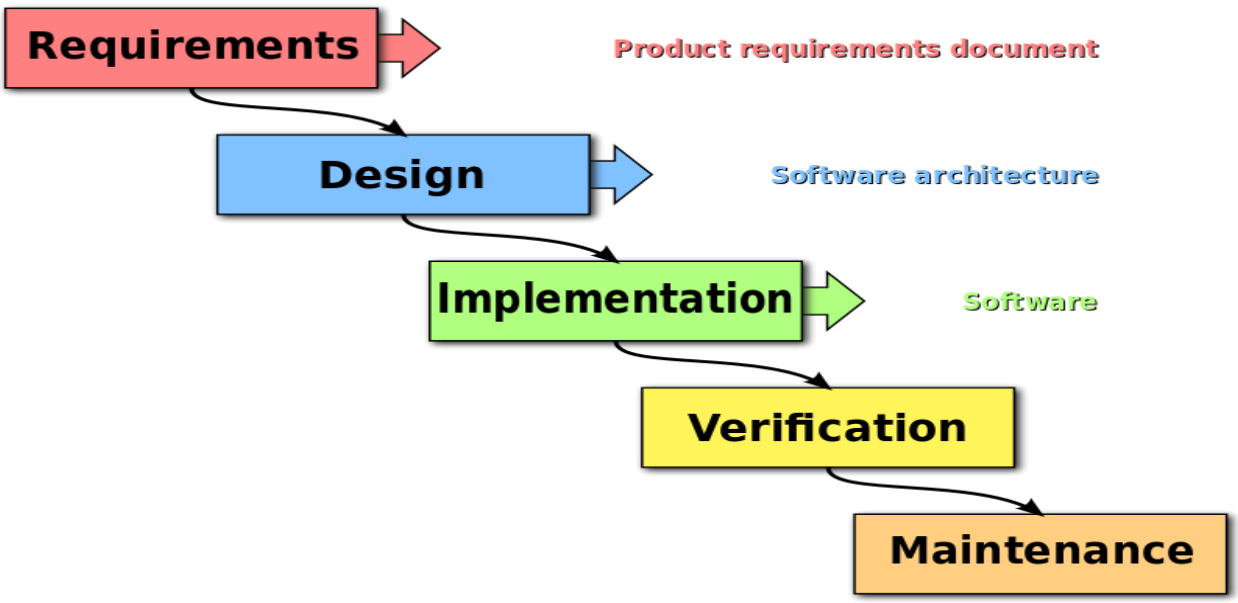


Figure 4.3 Waterfall model

Advantages	Dis-Advantages
• Before the next phase of development, each phase must be completed	• Error can be fixed only during the phase
• Suited for smaller projects where requirements are well defined	• It is not desirable for complex project where requirement changes frequently
• They should perform quality assurance test (Verification and Validation) before completing each stage	• Testing period comes quite late in the developmental process

• Elaborate documentation is done at every phase of the software's development cycle	• Documentation occupies a lot of time of developers and testers
• Project is completely dependent on project team with minimum client intervention	• Clients valuable feedback cannot be included with ongoing development phase
• Any changes in software is made during the process of the development	• Small changes or errors that arise in the completed software may cause a lot of problems

Table 4.2 Advantage and disadvantage of waterfall

4.5.2. Flow Diagram Architecture:

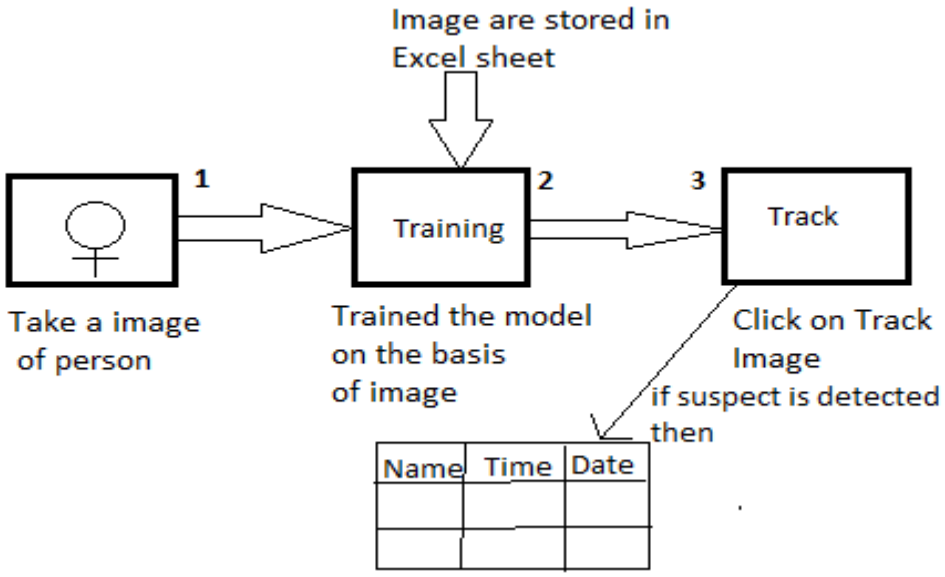


Figure 4.3 Flow Diagram

CHAPTER 5. IMPLEMENTATION & RESULTS

5.1 Implementation Approach-

I started with choosing the tools first, in terms of how I will build the applications. I started doing a feasibility study about that I found python will work well for me in terms of strong predictions and analyzing. I started building the datasets, training them, creating a module and started implementing my ideas.

I started finding some of the frontend development tools which can help me to give good look to feel for my applications, I discover some of the tools like PyCharm, Tkinter python library, etc. Python with Tkinter is the fastest and easiest way to create GUI applications. Tkinter was written by Steen Lumholt and Guido van Rossum, then later revised by Fredrik Lundh. Tkinter is free software released under a Python license.

I started deciding my module likes Live suspect tracking, capturing, training, etc. Now I had to choose my software development approach for my idea, I decided to choose waterfall model.

Waterfall Approach-

Development activities are performed in order, with possibly minor overlap, but with little or no iteration. Typically, the outcome of one phase acts as the input for the next phase sequentially. The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one. Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Figure 5.1 Water fall model

5.2 Code Efficiency-

Code which has been used in is efficient as compare to other languages. I have used python language for this project. Python can be treated in a procedural way, an object-oriented way or a functional way. It is possible to write Python in an Integrated Development Environment, such as Idle, Pycharm, Netbeans, or Eclipse which are particularly useful when managing larger collections of Python files.

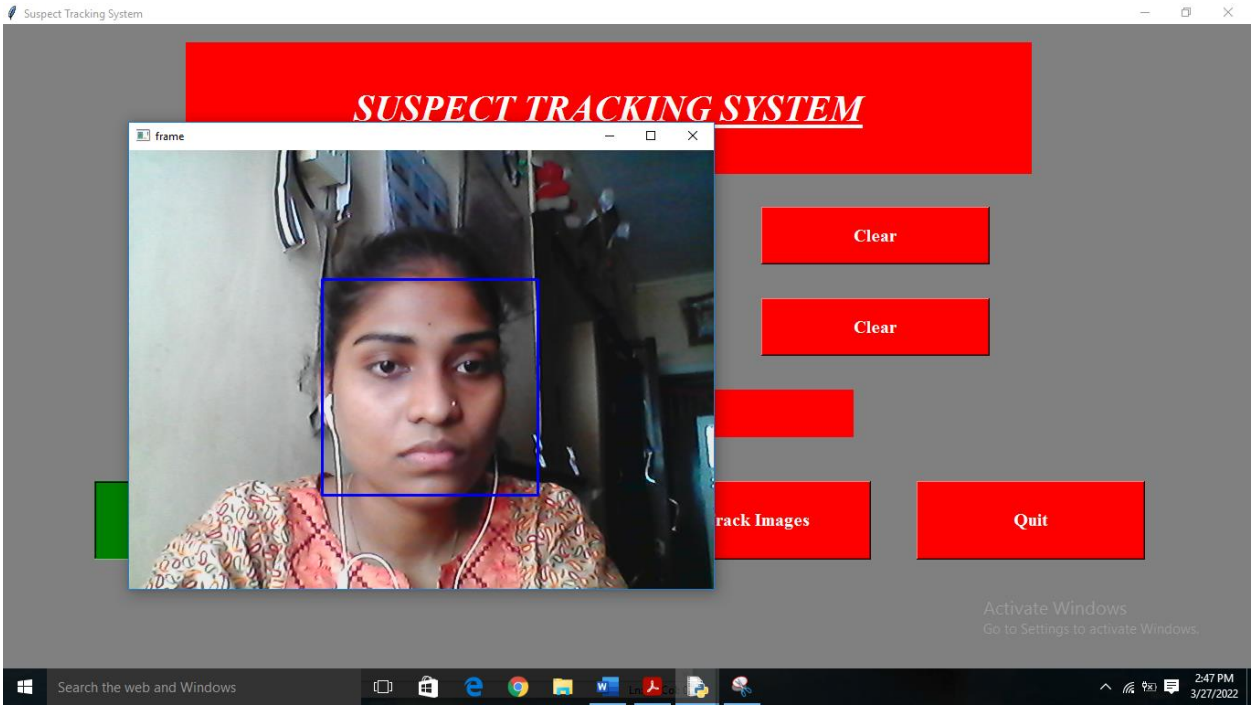
1) Python was designed for readability and has some similarities to the English language with influence from mathematics.
2) Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
3) Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions, and classes. Other programming languages often use curly brackets for this purpose.

5.3Testing-

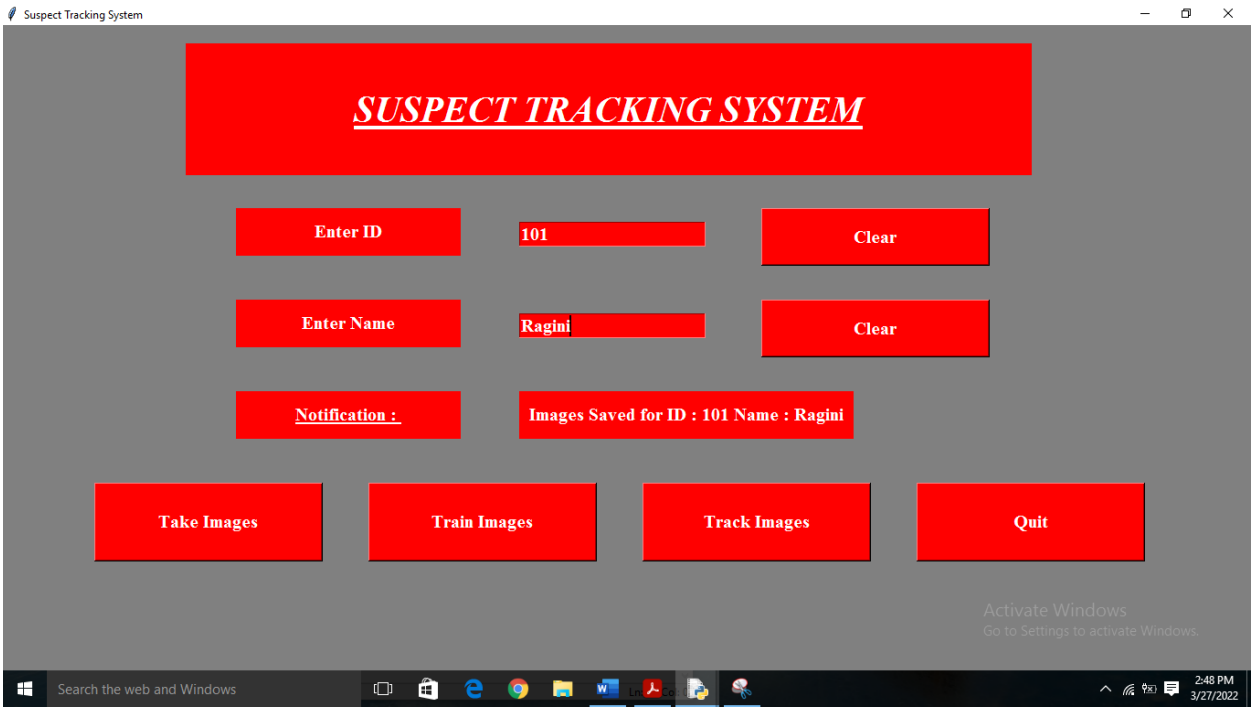
5.3.1 Unit testing-

Unit testing is a type of software testing where individual units or components of the software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

Capturing Image-



Output-



5.3.2 Integration Testing-

Integration testing is defined as a type of testing where software modules are integrated logically and tested as a group. A typical software project consists of multiple software modules, coded by different programmers. The purpose of this level of testing is to expose defects in the interaction between these software modules when they are integrated. Integration Testing focuses on checking data communication amongst these modules.

Train.py-

```
import tkinter as tk
from tkinter import Message ,Text
import cv2,os
import shutil
import csv
import numpy as np
from PIL import Image, ImageTk
import pandas as pd
import datetime
import time
import tkinter.ttk as ttk
import tkinter.font as font

window = tk.Tk()
#helv36 = tk.Font(family='Helvetica', size=36, weight='bold')
window.title("Suspect Tracking System")

dialog_title = 'QUIT'
dialog_text = 'Are you sure?'
#answer = messagebox.askquestion(dialog_title, dialog_text)

#window.geometry('1280x720')
window.configure(background='grey')
```

```
#window.attributes('-fullscreen', True)

window.grid_rowconfigure(0, weight=1)
window.grid_columnconfigure(0, weight=1)

#path = "profile.jpg"

#Creates a Tkinter-compatible photo image, which can be used everywhere Tkinter expects an image object.
#img = ImageTk.PhotoImage(Image.open(path))

#The Label widget is a standard Tkinter widget used to display a text or image on the screen.
#panel = tk.Label(window, image = img)
#panel.pack(side = "left", fill = "y", expand = "no")

#cv_img = cv2.imread("img541.jpg")
#x, y, no_channels = cv_img.shape
#canvas = tk.Canvas(window, width = x, height =y)
#canvas.pack(side="left")
#photo = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(cv_img))
# Add a PhotoImage to the Canvas
#canvas.create_image(0, 0, image=photo, anchor=tk.NW)

#msg = Message(window, text='Hello, world!')
# Font is a tuple of (font_family, size_in_points, style_modifier_string)

message = tk.Label(window, text="SUSPECT TRACKING SYSTEM" ,bg="red"  ,fg="white" ,width=40  ,height=3,font=('times', 30, 'italic bold underline'))
message.place(x=200, y=20)
```

```
lbl = tk.Label(window, text="Enter ID",width=20, height=2  ,fg="white"  ,bg="red" ,font=('times', 15, ' bold ') )
lbl.place(x=255, y=200)#300

txt = tk.Entry(window, width=20, bg="red", fg="white",font=('times', 15, ' bold '))
txt.place(x=565, y=215)

lbl2 = tk.Label(window, text="Enter Name",width=20,fg="white"  ,bg="red" , height=2 ,font=('times', 15, ' bold '))
lbl2.place(x=255, y=300)

txt2 = tk.Entry(window,width=20  ,bg="red"  ,fg="white",font=('times', 15, ' bold ')  )
txt2.place(x=565, y=315)

lbl3 = tk.Label(window, text="Notification : ",width=20  ,fg="white"  ,bg="red"  ,height=2 ,font=('times', 15, ' bold underline '))
lbl3.place(x=255, y=400)

message = tk.Label(window, text="" ,bg="red"  ,fg="white"  ,width=30  ,height=2, activebackground = "grey" ,font=('times', 15, ' bold '))
message.place(x=565, y=400)

#lbl3 = tk.Label(window, text="Suspect : ",width=20  ,fg="white"  ,bg="red"  ,height=2 ,font=('times', 15, ' bold  underline'))
#lbl3.place(x=400, y=650)

#message2 = tk.Label(window, text="" ,fg="white"  ,bg="red",activeforeground = "green",width=30  ,height=2  ,font=('times', 15, ' bold '))
#message2.place(x=700, y=650)

def clear():
```

```
    txt.delete(0, 'end')
    res = ""
    message.configure(text= res)
def clear2():
    txt2.delete(0, 'end')
    res = ""
    message.configure(text= res)

def is_number(s):
    try:
        float(s)
        return True
    except ValueError:
        pass

    try:
        import unicodedata
        unicodedata.numeric(s)
        return True
    except (TypeError, ValueError):
        pass

    return False

def TakeImages():
    Id=(txt.get())
    name=(txt2.get())
    if(is_number(Id) and name.isalpha()):
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector=cv2.CascadeClassifier(harcascadePath)
```

```
        sampleNum=0
        while(True):
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, 1.3, 5)
            for (x,y,w,h) in faces:
                cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
                #incrementing sample number
                sampleNum=sampleNum+1
                #saving the captured face in the dataset folder TrainingImage
                cv2.imwrite("TrainingImage\ "+name +"."+Id +'.'+ str(sampleNum) + ".jpg",
gray[y:y+h,x:x+w])
                #display the frame
                cv2.imshow('frame',img)
            #wait for 100 miliseconds
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
            # break if the sample number is morethan 100
            elif sampleNum>60:
                break
        cam.release()
        cv2.destroyAllWindows()
        res = "Images Saved for ID : " + Id +" Name : "+ name
        row = [Id , name]
        with open('SuspectDetails\SuspectDetails.csv','a+') as csvFile:
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        message.configure(text= res)
    else:
        if(is_number(Id)):
```

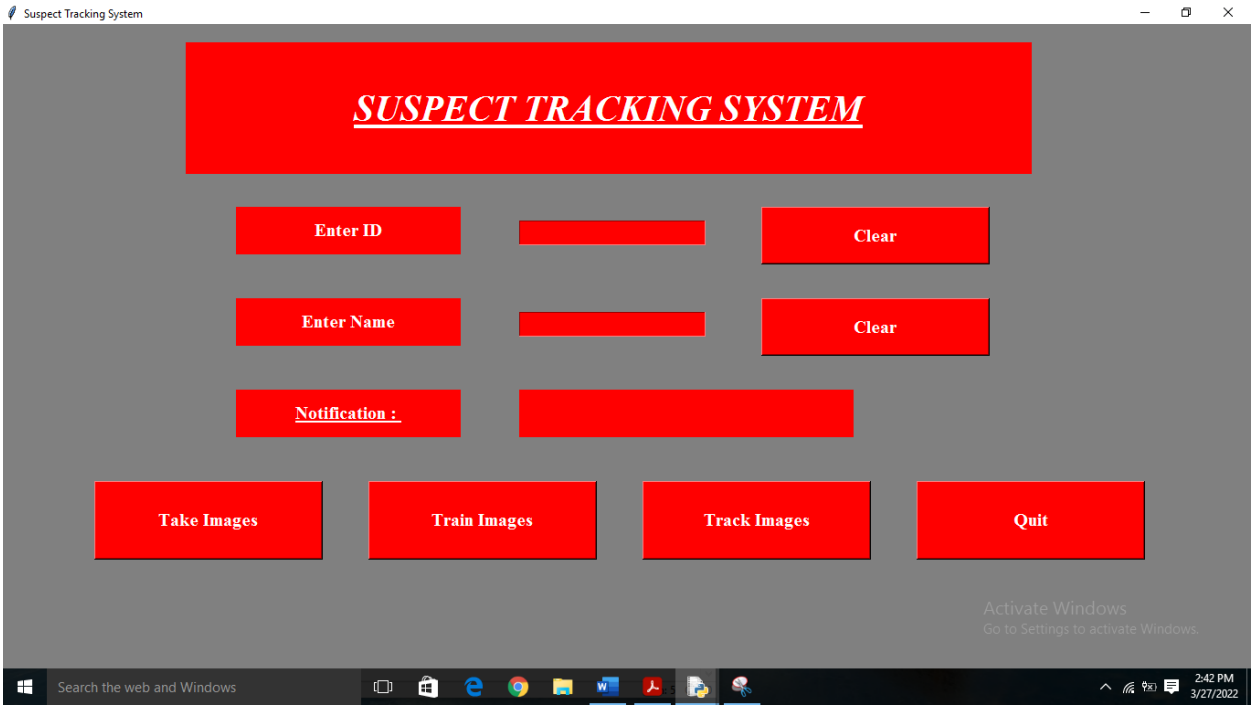
```
            res = "Enter Alphabetical Name"
            message.configure(text= res)
        if(name.isalpha()):
            res = "Enter Numeric Id"
            message.configure(text= res)

def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()#recognizer = cv2.face.LBPHFaceRecognizer_create()#$cv2.createLBPHFaceRecognizer()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector =cv2.CascadeClassifier(harcascadePath)
    faces,Id = getImagesAndLabels("TrainingImage")
    recognizer.train(faces, np.array(Id))
    recognizer.save("TrainingImageLabel\Trainner.yml")
    res = "Image Trained"#+",".join(str(f) for f in Id)
    message.configure(text= res)

def getImagesAndLabels(path):
    #get the path of all the files in the folder
    imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
    #print(imagePaths)

    #create empth face list
    faces=[]
    #create empty ID list
    Ids=[]
    #now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
        #loading the image and converting it to gray scale
        pilImage=Image.open(imagePath).convert('L')
        #Now we are converting the PIL image into numpy array
```

```
        imageNp=np.array(pilImage,'uint8')
        #getting the Id from the image
        Id=int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(Id)
    return faces,Ids

def TrackImages():
    recognizer = cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognizer()
    recognizer.read("TrainingImageLabel\Trainner.yml")
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath);
    df=pd.read_csv("SuspectDetails\SuspectDetails.csv")
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names =  ['Id','Name','Date','Time']
    attendance = pd.DataFrame(columns = col_names)
    while True:
        ret, im =cam.read()
        gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
        faces=faceCascade.detectMultiScale(gray, 1.2,5)
        for(x,y,w,h) in faces:
            cv2.rectangle(im,(x,y),(x+w,y+h),(225,0,0),2)
            Id, conf = recognizer.predict(gray[y:y+h,x:x+w])
            if(conf < 50):
                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa=df.loc[df['Id'] == Id]['Name'].values
                tt=str(Id)+"-"+aa
```

```
            attendance.loc[len(attendance)] = [Id,aa,date,timeStamp]


        else:
            Id='Unknown'
            tt=str(Id)
        if(conf > 75):
            noOfFile=len(os.listdir("ImagesUnknown"))+1
            cv2.imwrite("ImagesUnknown\Image"+str(noOfFile) + ".jpg", im[y:y+h,x:x+w])
        cv2.putText(im,str(tt),(x,y+h), font, 1,(255,255,255),2)
    attendance=attendance.drop_duplicates(subset=['Id'],keep='first')
    cv2.imshow('im',im)
    if (cv2.waitKey(1)==ord('q')):
        break
ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
Hour,Minute,Second=timeStamp.split(":")
fileName="Attendance\Attendance_"+date+"_"+Hour+"-"+Minute+"-"+Second+".csv"
attendance.to_csv(fileName,index=False)
cam.release()
cv2.destroyAllWindows()
#print(attendance)
res=attendance
message2.configure(text= res)

clearButton = tk.Button(window, text="Clear", command=clear, fg="white", bg="red", width=20,
height=2, activebackground = "Red" ,font=('times', 15, ' bold '))
clearButton.place(x=830, y=200)#950
clearButton2 = tk.Button(window, text="Clear", command=clear2 ,fg="white" ,bg="red"
,width=20 ,height=2, activebackground = "Red" ,font=('times', 15, ' bold '))
clearButton2.place(x=830, y=300)
```

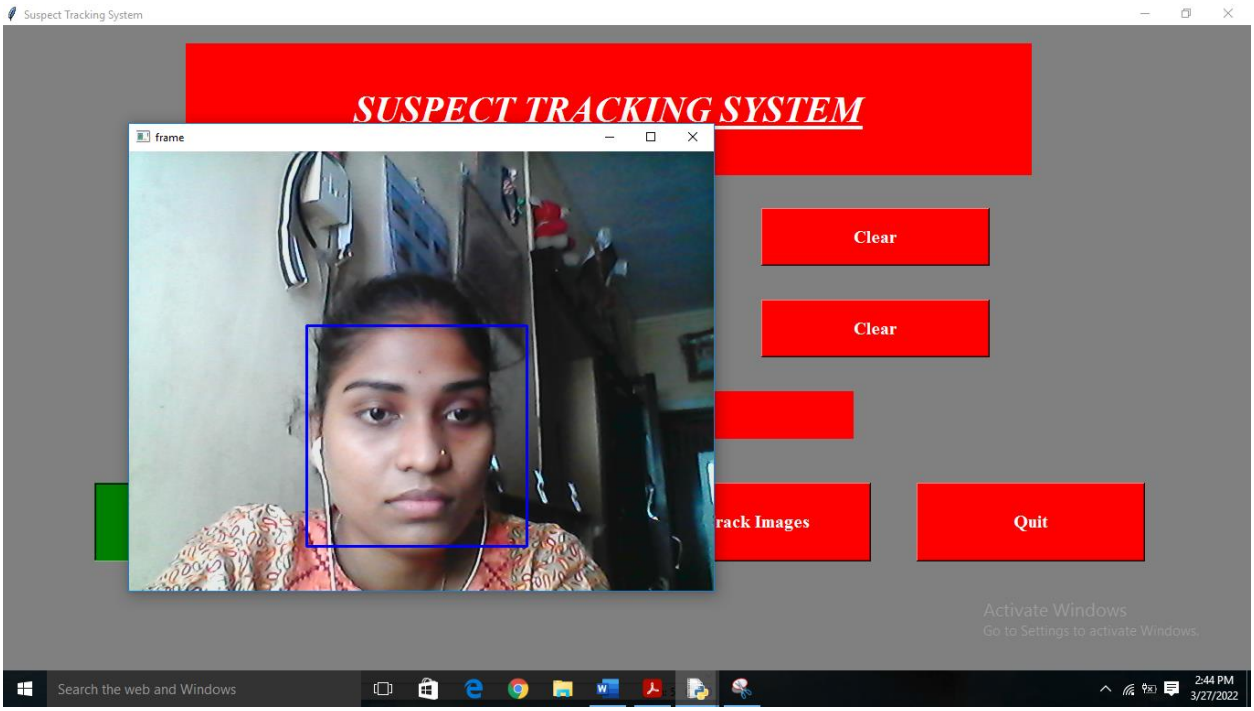
```
takeImg = tk.Button(window, text="Take Images", command=TakeImages ,fg="white"
,bg="red" ,width=20 ,height=3, activebackground = "green" ,font=('times', 15, ' bold '))
takeImg.place(x=100, y=500)
trainImg = tk.Button(window, text="Train Images", command=TrainImages ,fg="white"
,bg="red" ,width=20 ,height=3, activebackground = "Red" ,font=('times', 15, ' bold '))
trainImg.place(x=400, y=500)
trackImg = tk.Button(window, text="Track Images", command=TrackImages ,fg="white"
,bg="red" ,width=20 ,height=3, activebackground = "Red" ,font=('times', 15, ' bold '))
trackImg.place(x=700, y=500)
quitWindow = tk.Button(window, text="Quit", command=window.destroy ,fg="white"
,bg="red" ,width=20 ,height=3, activebackground = "Red" ,font=('times', 15, ' bold '))
quitWindow.place(x=1000, y=500)

window.mainloop()
```

Home Page-



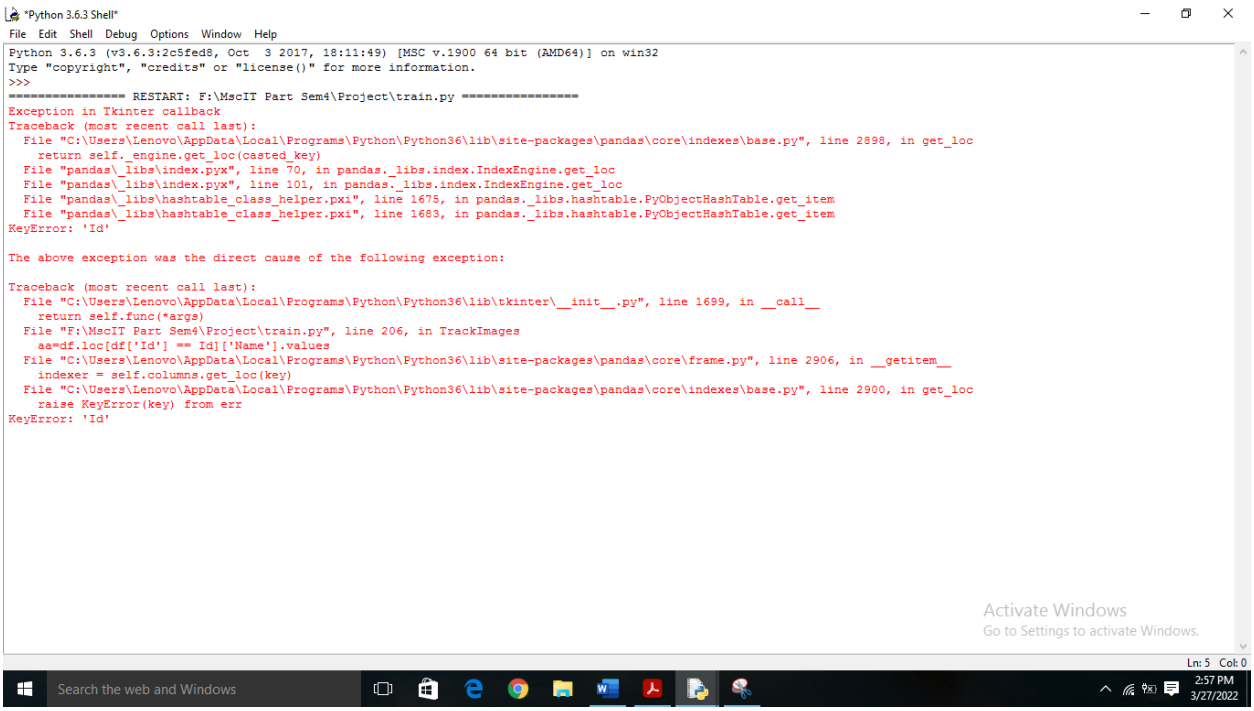
Filled in all the details and click on Take Image→After that One Tkinter window is open.

Output-



After that Click on Track Image→Again one window is open→It detect the suspect.

But in the my case it showing below error.



5.3.3 Beta Testing-

Beta Testing is one of the Acceptance Testing types, which adds value to the product as the end-user (intended real user) validates the product for functionality, usability, reliability, and compatibility.

Inputs provided by the end-users help in enhancing the quality of the product further and leads to its success. This also helps in decision making to invest further in the future products or the same product for improvisation. Since Beta Testing happens at the end user's side, it cannot be a controlled activity.

Challenges-

5.4Test Cases-

No.	Description	Test Data	Expected Output	Actual Output	Pass/Fail
1	The project is run on python idle	F5	Successfully executed	Same as excepted	Pass

2	Take image	Image clicked	Successfully click the image.	Same as expected	Pass
3	Train Image	Image trained	Trained	Trained notify	Pass
4	Quit	Quit	Close the window	Window closed	Pass
5	Track Image	Track Face	Not detected	Not got	Fail
6	Track Image	Track Face	Detected	Same as expected	Pass

Table 5.4.1 Test Cases

5.5Test Reports-

As mentioned above I have tested each and individual components or modules in my project. Every module working fine as expected. Refer table 5.4.1 Test Cases.

I had tested total 5 Modules (Mentioned below) all is working fine.

- Project run on python Idle and code run successfully.
- Take Image- One window is open and click the image.
- Train Image-Train the image and create 61 images of the image.
- Quit-Exit from the Window screen.
- Track Image- Detect the Suspect and show their name on the screen.

5.6Modification-

1) There was some gap between the expected and the actual result, I decided to provide the details of the suspect from the video but I am not able to implement that thing in my project.
2) But in the future, I am trying to implement that thing and from the video, we can detect the suspect.

5.7User Documentation-

CHAPTER 6. CONCLUSIONS AND PROPOSALS FOR THE FUTURE SCOPE

6.1 Conclusions-

The field of facial recognition is popular in the current and future years. Almost the previous year not everyone is aware of facial recognition, but the current release of many applications that are the beginning of a new era of innovation in Deep learning. Facial reorganization offers an efficient way of identifying and tracking the suspect based on the training datasets rules. The results showed the various possibilities in which it helps to investigate criminal and civil justice at the same level of high standards and with the same distinguished results.

Crime is an awful and illegal act against law for doing wrong things out of which someone can be punished by police authorities and the government. A criminal is a person who has committed or is involved with any kind of crime. Crimes or Violations are a social nuisance (problem) and cost our society nearly in several ways. In our society, the crime rate is growing very rapidly, especially women are facing many such crime problems. The reason for this might be the low pronouncement of guilt. About 10% of criminals commit 50% of crimes. This system helps track or identify them.

Through this system, we can detect the suspect, and the police can easily track the suspect. This system will reduce the workload of the police. It provides an efficient way of tracking a suspect. The system is proposed to help agencies like CBI, CID, and other such Bureau to speed up the investigation process and track the status of multiple cases at a time. This system is invariant to lighting conditions, background scenarios, and viewpoints. The machine learning algorithms Haar Cascade have a lot of positive and negative images that are used the train the classifier.

6.2 Limitations-

1) The background of the image that is given as the input to the system should remain the same for most of the time in case of filtering but it can be overcome by DWT (Discrete wavelet transform).
2) The size of the template of the object that is to be detected should be less than the actual object.
3) Proper illumination conditions should always be present.

4) There should not be any drastic change in the motion or light of the room for the suspect to be detected.

6.3 Future Scope-

1) In the future, you can use different algorithms for efficient and fast processing like a Convolutional neural network or LBP-based classifier then which gives better accuracy.
2) In my case, I have used my laptop camera but you can use a better camera for capturing the suspect it gives better clarification.
3) I have used Python IDLE but if you want you can use Pycharm for coding.
4) In the future, I can implement more features in this project such as In the Video it can track the suspect.
5) In the future Try to add a message feature when a suspect is detected.

CHAPTER 7. BIBLIOGRAPHY

1) https://ijesc.org/upload/a34318960524a381b0a311d882104ae8.Criminal%20Investigation%20Tracker%20with%20Suspect%20Prediction.pdf
2) https://www.spit.ac.in/wp-content/uploads/profkttalele/project/2009-10/(4)Suspicious%20Object%20Recognization/report%20Suspicious_Object_Detection.pdf
3) https://towardsdatascience.com/face-detection-with-haar-cascade-727f68dafd08#:~:text=So%20what%20is%20Haar%20Cascade,Simple%20Features%E2%80%9D%20published%20in%202001.
4) https://nevonprojects.com/criminal-investigation-tracker-with-suspect-prediction/