Vel Tech Multi Tech
Dr.Rangarajan Dr.Sakunthala Engineering College
An Autonomous Institution

FACE DETECTION AND ALERTING SYSTEM USING PYTHON

A MINI-PROJECT REPORT

Submitted by

GIRISHUN KUMAR R (113119UG03027)

KISHORE M (113119UG03050)

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

VEL TECH MULTI TECH DR. RANGARAJAN DR. SAKUNTHALA

ENGINEERING COLLEGE, ALAMATHI ROAD, AVADI, CHENNAI-62

JUNE - 2022

BONAFIDE CERTIFICATE

Certified that this project report of title "**FACE DETECTION AND ALERTING SYSTEM USING PYTHON**" is the Bonafide work of **GIRISHUN KUMAR R (113119UG03027), KISHORE M (113119UG03050)** who carried out the project work under my supervision.

SIGNATURE

HEAD OF THE DEPARTMENT
Dr.N.Ganesh., M.E., MBA., M.L., Ph.D**,**
Dean and Professor,
Department of Computer Science and Engineering,
Vel Tech Multi Tech Dr. Rangarajan Dr. Sakunthala Engineering College,
Avadi, Chennai-600 062

SIGNATURE

SUPERVISOR
Ms.V.Vijayashanthi M.tech(IT),
Asst.Professor,
Department of Computer Science and Engineering,
Vel Tech Multi Tech Dr.Rangarajan Dr.Sakunthala Engineering College,
Avadi, Chennai-600 062

CERTIFICATE FOR EVALUATION

This is to certify that the project entitled **"FACE DETECTION AND ALERTING SYSTEM USING PYTHON"** is the bonafide record of work done by following students to carry out the project work under our guidance during the year 2021-2022 in partial fulfilment for the award of Bachelor of Engineering degree in Computer Science and Engineering conducted by Anna University, Chennai.

GIRISHUN KUMAR R **(113119UG03027)**

KISHORE M **(113119UG03050)**

This project report was submitted for viva voice held on _____

At Vel Tech Multi Tech Dr.Rangarajan and Dr.Sakunthala Engineering College.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We wish to express our sincere thanks to the almighty and the people who extended their help during the course of our work. We are greatly and profoundly thankful to our honourable Chairman, **Col. Prof.Vel. Shri Dr.R.Rangarajan B.E (ELEC)., B.E (MECH)., M.S (AUTO)., D.Sc.,** Vice Chairman, **Dr.Mrs.Sakunthala Rangarajan M.B.B.S.**, for facilitating us with this opportunity. We also record our sincere thanks to our honourable Principal, **Dr.V.Rajamani M.E., Ph.D.,** for his kind support to take up this project and complete it successfully. We would like to express our special thanks to our Head of the Department, **Dr.N.Ganesh M.E., MBA., M.L., Ph.D.** Department of Computer Science and Engineering and our project supervisor **Ms.V.Vijayashanthi M.tech(IT)** for their moral support by taking keen interest on our project work and guided us all along, till the completion of our project work and also by providing with all the necessary information required for developing a good system with successful completion of the same.

Further, the acknowledgement would be incomplete if we would not mention a word of thanks to our most beloved parents for their continuous support and encouragement all the way through the course that has led us to pursue the degree and confidently complete the project work.

(GIRISHUN KUMAR R) **(KISHORE M)**

ABSTRACT

Privacy and Security are two of the most important universal rights. To ensure security in our daily life through technology, a lot of research is going on. Among them facial recognition is a popular and well-established technology. In this technology, faces are detected and identified out of images and with the help of Machine Learning (ML), it becomes even more useful and precise. Using face recognition and Machine Learning , we aim to create a smart way, which secures the gateway on the basis of who we are. In our proof of the concept of such a smart security system, we have used Open-cv method to detect faces, trainer.yml method to recognize people and we used Pywhatkit method to alert the owner/admin through WhatsApp.

KEYWORDS USED: *Image Processing, Computer Vision, Machine Learning, Artificial Intelligence, Deep Learning, Computational Intelligence, OpenCv, Pywhatkit.*

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO.
	ABSTRACT	**V**
1.	**INTRODUCTION**	**1**
	1.1. OBJECTIVE	4
	1.2. SCOPE OF THE PROJECT	4
	1.3. LITERATURE SURVEY	5
2.	**SYSTEM ANALYSIS**	**13**
	2.1. EXISTING SYSTEM	14
	2.2. PROPOSED SYSTEM	15
	2.3. FEASIBILITY STUDY	15
3.	**SYSTEM SPECIFICATIONS**	**17**
	3.2. SOFTWARE SPECIFICATIONS	18
4.	**SOFTWARE DESCRIPTION**	**20**
5.	**MODULE DESCRIPTION**	**24**
	5.1. PROJECT DEFINITION	26
	5.2. OVERVIEW OF THE PROJECT	32
	5.3. METHODOLOGY	32
	5.4. ARCHITECTURE DIAGRAM	33
	5.5. MODULES	33
	5.6. ALERTING MESSAGE VIA WHATSAPP	45
6.	**SYSTEM TESTING**	**48**
	6.1. CODE REVIEW	49
	6.2. TESTING PROCESS	49
	6.3. EXPERIMENTAL OUTCOME	52
		53

7.	**SYSTEM IMPLEMENTATION**	
	7.1. IMPLEMENTATION PROCEDURE	54
	7.2. CONVOLUTION OPENCV	54
8.	**CONCLUSION AND FUTURE ENHANCEMENTS**	**57**
	8.1. CONCLUSION	58
	8.2. FUTURE ENHANCEMENTS	58
	APPENDICES	**59**
	APPENDIX-1 SCREENSHOTS	**60**
	APPENDIX-2 IMPLEMENTATION CODE	**63**
	REFERENCES	**72**

CHAPTER 1

INTRODUCTION

INTRODUCTION

In the last two decades, the facial recognition system has become one of the most important and interesting research in the technological field. Identifying a person with an image has been popularised through the mass media. Most of the biometric data have to be collected by using special hardware such as fingerprint scanners, palm print scanners, DNA analysers etc. But face recognition, in this particular area, has some advantages. It's hassle-free in a sense that one doesn't need to touch anything in order to be identified or recognized. As a result, in the field of biometrics, face recognition technology is one of the fastest growing fields.). Face recognition is a non-invasive identification system and faster than other systems since multiple faces can be analysed at the same time. The difference between face detection and identification is, face detection is to identify a face from an image and locate the face. Face recognition is making the decision" whose face is it?", using an image database.

Major areas of the commercial use of face recognition include biometrics, law enforcement and surveillance, human-computer interaction, multimedia management (for example, automatic tagging of a particular individual within a collection of digital photographs), smart cards, passport check, criminal investigations, access control etc. However, face detection can be a bit challenging at times owing to some unstable characteristics of a human face. For example, glasses and beard will affect the detecting accuracy. Moreover, different kinds and angles of lighting will generate uneven brightness on the face, which will have influence on the detection process.

To overcome these problems, the system used Viola Jones method for face detection and Eigenfaces method for face recognition. Eigenfaces are generated using a mathematical process called Principal Component Analysis (PCA). If a face is recognized, it is known and if not, it is unknown. Since PCA reduces the dimensions of facial images without losing important features, facial images for many people can be stored in the database without losing significant efficiency. Therefore, face recognition using PCA is more useful for home/office security systems than other face recognition techniques.

This recent interest in face recognition can be attributed to the increase of commercial interest and for the development of feasible technologies to support the development of face recognition. A facial recognition system is a software application for certifying an individual and recognizing him/her with images or videos from a source. Facial recognition can be done fast and accurately with the open-source platform called OpenCV and methodology to implement them using Python.

A path from a face and a picture database are favourite facial features. It is usually compared to biometrics such as fingerprints and eye investigation systems, and security systems and used in thumb detection systems. The OpenCV library makes programming easy to use. This comes up with advanced proficiencies like face detection, face tracking, facial recognition, and many more methods for artificial intelligence (AI). The main advantage of the OpenCV library is, it is a multi-platform framework; it supports Windows, Mac OS, Mac OS X.

Its challenging work includes face recognition on the lowest computing cost framework such as smartphones and embedded devices. For the person's testimony, facial recognition is used. Everyone has unique features that they do not share with another person. Currently, there are many devices and applications which use face detection technology to recognize and detect a face such as Facebook. Therefore, face detection is not new in the vision of computer science. In this letter, we have taken our research using OpenCV.

In other words, Security has become an important factor. Intruders have become prominent factors for all the data/property theft. The basic idea in this paper is to identify the intruder Whenever an unknown/unidentified person comes to the vicinity of the camera, all the above-said features get activated and the owner gets alerted and alert owner/administrator in different possible ways. This paper discusses different ways such as "WhatsApp message" and "intruder's image to owner's/administrator's WhatsApp" to alert owner/administrator. For identifying the intruder, a machine learning algorithm is used. A camera placed at the locality is trained such that it can identify the familiar people and it is "on" all the time.

1.1 OBJECTIVE

The major objectives of the project are as follows:

- We are proposing an Information Security Strategy.
- To identify the intruder using a machine learning algorithm.
- Image processing techniques are performed on this image and feature values are extracted.
- Detects and analyses faces.
- Converting it to an image, to find a match.
- If the face doesn't match with our data, alert through WhatsApp.
- Online monitoring.

1.2. SCOPE OF THE PROJECT

The scope of the project is more interactive and user-friendly. We are planning to integrate this machine learning project as a web application with an advanced python framework and the open-source platform called OpenCV. The OpenCV library makes programming easy to use. This comes up with advanced proficiencies like face detection, face tracking, facial recognition, and many more methods for artificial intelligence (AI).

By using this we are going to detect the faces of the persons passing through the camera, and check whether their photo matches with the data provided. If it matches it sends the name of the person entered or else sends unknown person entered / unknown person detected, then we can monitor through online using WIFI and smart devices.

1.3. LITERATURE SURVEY

1.3.1: G.T. Rado, H. Suhl, I.S. Jacobs and C.P. Bean, "Fine particles thin films and exchange anisotropy" in International Journal of Soft Computing and Artificial Intelligence, New York: Academic, vol. III, pp. 271-350, 1990, ISBN 2321-404X.

An application for tracking and detecting faces in videos and in cameras which can be used for multipurpose activities. The intention of the paper is deep study of face detection using open CV. A tabular comparison is performed in order to understand the algorithms in an easier manner. It talks about various algorithms like Adaboost, Haar-cascades. This paper aims to help in understanding the best prerequisites for face detection.

Face detection is the most popular area of research in the vision of computer science. It is a computer technology which is being used in a variety of applications that identifies human faces in digital images. The research under this field is expanding in many areas of science such as psychology. Face detection is one of the most talked about in technology. Localization of human faces is considered as the primary and the initial stage in study of face detection. For example, in home video surveillance etc. Face localization can be referred to as extraction of facial features using pattern recognition system. Both MATLAB and Open CV can be used for creating such prototypes and systems. In this paper we have carried out our research using Open CV. The Reasons for using open CV have been discussed further in this paper.

1.3.2: M.A. Turk and A.P. Pentland, "Face Recognition Using Eigenfaces", IEEE Conf. on Computer Vision and Pattern Recognition, pp. 586-591, 1991.

An approach to the detection and identification of human faces is presented, and a working, near-real-time face recognition system which tracks a subject's head and then recognizes the person by comparing characteristics of the face to those of known individuals is described. This approach treats face recognition as a two-dimensional recognition problem, taking advantage of the fact that faces are normally upright and thus may be described by a small set of 2-D characteristic views. Face images are projected onto a feature space ('face space') that best encodes the variation among known face images. The face space is defined by the 'eigenfaces', which are the eigenvectors of the set of faces; they do not necessarily correspond to isolated features such as eyes, ears, and noses. The framework provides the ability to learn to recognize new faces in an unsupervised manner.

1.3.3: KyungnamKim" Face Recognition using Principle Component Analysis, Conference paper-IEEE, vol.III, 2001.

A kernel principal component analysis (PCA) was previously proposed as a nonlinear extension of a PCA. The basic idea is to first map the input space into a feature space via nonlinear mapping and then compute the principal components in that feature space. This article adopts the kernel PCA as a mechanism for extracting facial features. Through adopting a polynomial kernel, the principal components can be computed within the space spanned by high-order correlations of input pixels making up a facial image, thereby producing a good performance.

1.3.4: G B Huang, H Lee and EL. Miller, "Learning hierarchical representation for Face verification with convolutional deep belief networks[C]", Proceedings of International Conference on Computer Vision and Pattern Recognition, pp. 223-226, 2012.

Most modern face recognition systems rely on a feature representation given by a hand-crafted image descriptor, such as Local Binary Patterns (LBP), and achieve improved performance by combining several such representations. In this paper, we propose deep learning as a natural source for obtaining additional, complementary representations. To learn features in high-resolution images, we make use of convolutional deep belief

networks. Moreover, to take advantage of global structure in an object class, we develop local convolutional restricted Boltzmann machines, a novel convolutional learning model that exploits the global structure by not assuming stationarity of features across the image, while maintaining scalability and robustness to small misalignments. We also present a novel application of deep learning to descriptors other than pixel intensity values, such as LBP. In addition, we compare performance of networks trained using unsupervised learning against networks with random filters, and empirically show that learning weights not only is necessary for obtaining good multilayer representations, but also provides robustness to the choice of the network architecture parameters. Finally, we show that a recognition system using only representations obtained from deep learning can achieve comparable accuracy with a system using a combination of hand-crafted image descriptors. Moreover, by combining these representations, we achieve state-of-the-art results on a real-world face verification database.

1.3.5: Paul Viola and Matthew Jones, "Rapid object detection using a boosted cascade of simple features", Conference paper-IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

This paper describes a machine learning approach for visual object detection which is capable of processing images extremely rapidly and achieving high detection rates. This work is distinguished by three key contributions. The first is the introduction of a new image representation called the "integral image" which allows the features used by our detector to be computed very quickly. The second is a learning algorithm, based on AdaBoost, which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers. The third contribution is a method for combining increasingly more complex classifiers in a "cascade" which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions. The cascade can be viewed as an object specific focus-of-attention mechanism which, unlike previous approaches, provides statistical guarantees that discarded regions are unlikely to contain the object of interest. In the domain of face detection the system yields detection rates comparable to the best previous systems. Used in real-time applications, the detector runs at 15 frames per second without resorting to image differencing or skin colour detection.

1.3.6: Ravishankara K, Dhanush, Vaisakh, Srajan I S "Whatsapp Chat Analyzer", Conference paper-IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol.III, 2011.

The most used and efficient method of communication in recent times is an application called WhatsApp. WhatsApp chats consist of various kinds of conversations held among groups of people. This chat consists of various topics. This information can provide lots of data for the latest technologies such as machine learning. The most important thing for a machine learning model is to provide the right learning experience which is indirectly affected by the data that we provide to the model. This tool aims to provide in-depth analysis of this data which is provided by WhatsApp. Irrespective of whichever topic the conversation is based on, our developed code can be applied to obtain a better understanding of the data. The advantage of this tool is that is implemented using simple python modules such as pandas, matplotlib, seaborn and sentiment analysis which are used to create data frames and plot different graphs, where then it is displayed in the flutter application which is efficient and less resources consuming algorithm, therefore it can be easily applied to largest dataset.

1.3.7: S. Jacobs and C. P. Bean, "Fine Particles, Thin Films and Exchange Anisotropy," In Magnetism, G. T. Rado and H. Suhl, Eds., Academic, New York, Vol. 3, 1963, pp. 271-350.

In a 2 × 2 MIMO antenna array system envelope correlation coefficient "ρ" shows the influence of different propagation paths of the RF signals that reach the antenna elements. The approximated value of this coefficient is based on a simple closed-form equation and also varies from 0 to 1. Quite perfect performance for MIMO applications is achieved when this parameter approximates to zero. In this paper, we evaluate an antenna diversity MIMO system by measuring the envelope correlation coefficient. The corresponding results in our antenna array configurations show that the measured "ρ" has very small values and approximates to zero. This observation indicates quite perfect behaviour and performance of our MIMO antenna array system.

1.3.8: Kruti Goyal, Kartikey Agarwal, Rishi Kumar "Face detection and tracking: Using OpenCV", Conference paper-IEEE, vol.IV, 2011.

An application for tracking and detecting faces in videos and in cameras which can be used for multipurpose activities. The intention of the paper is deep study of face detection using an open CV. A tabular comparison is performed in order to understand the algorithms in an easier manner. It talks about various algorithms like Adaboost, Haar cascades. This paper aims to help in understanding the best prerequisites for face detection.

CHAPTER 2

SYSTEM ANALYSIS

SYSTEM ANALYSIS

2.1. EXISTING SYSTEM

- In the existing system by using a CCTV camera, it records the video that can be viewed through a smartphone, tablet, or computer from anywhere over the internet.
- It captures the footage of the respective location and stores it in a SD card that can be viewed whenever needed.
- Loaded with some essential features only.

2.1.1. Disadvantages

- It can’t stop the theft.
- It doesn’t alert the property owner or anyone immediately.
- It is only used to identify the criminals only after notifying them of theft, burglary, or any other illegal or criminal activities

2.2. PROPOSED SYSTEM

- In the proposed system, initially the face of the users is detected. (registering to our database)
- After detection the faces are analysed and trained to get more accurate data of the users.
- When the collection of users' faces data is collected it is stored for the further identification of faces.
- Later when the user enters the visibility of the camera, the name of the user is sent as a message through WhatsApp, if it is an unknown person it sends a message as an unknown person entered.

2.2.1. ADVANTAGES

- Alerts the user immediately when an anonymous / unknown person enters.
- Don’t require more equipment.
- Can also be able to use the features of normal CCTV cameras along with this project.
- Completely software based.
- Cost as much as a normal CCTV camera.

2.3. FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and the project proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

2.3.1. Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.3.2. Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.3.3. Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER 3

SYSTEM SPECIFICATION

SYSTEM SPECIFICATIONS

3.1. SOFTWARE SPECIFICATION

- Client-Side PC
- Windows or Mac Platform PC
- Mac or Laptop with x86-64 (64-bit) compatible processors. **2 GHz or better processor** is recommended.
- At least **512 MB of free RAM** should be available for the application.
- A **camera or webcam** which is accessible using:
  - **DirectShow** interface on Microsoft Windows.
  - **GStreamer** interface on macOS or Linux.
- Internet connection required for managing the Face Verification transaction licenses.
- WhatsApp Application
- **Microsoft Windows specific requirements:**
  - Microsoft Windows **7 / 8 / 10**.
  - Microsoft **.NET framework 4.5** or newer (for .NET components usage).
  - One of following **development environments** for application development:
    - Microsoft Visual Studio 2012 or newer (for application development under C/C++, C#, Visual Basic .Net)
    - Java SE JDK 8 or newer
- **macOS specific requirements:**
  - **macOS 10.12.6** or newer.
  - XCode 6.x or newer (for application development)
  - GStreamer 1.10.x or newer with gst-plugin-base and gst-plugin-good is required for face capture using camera/webcam or rtsp video (for application development)

- GNU Make 3.81 or newer (to build samples and tutorials development)
- Java SE JDK 8 or newer (for application development)

CHAPTER 4

SOFTWARE DESCRIPTION

SOFTWARE DESCRIPTION

PYTHON CORE SOFTWARE

Python is an interpreter, high-level, general-purpose programming language. Created by Guido van Rossum released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python is an elegant and robust programming language that delivers both the power and general applicability of traditional compiled languages with the ease of use (and then some) of simpler scripting and interpreted languages. It allows you to get the job done, and then read what you wrote later. You will be amazed at how quickly you will pick up the language as well as what kind of things you can do with Python, not to mention the things that have already been done. Your imagination will be the only limit.

Although it has been around for well over fifteen years, some feel that Python is still relatively new to the general software development industry. We should,

however, use caution with our use of the word "relatively," as a few years seem like decades when developing on "Internet time."

There are three different ways to start Python. The simplest way is by starting the interpreter interactively, entering one line of Python at a time for execution. Another way to start Python is by running a script written in Python. This is accomplished by invoking the interpreter on your script application. Finally, you can run from a graphical user interface (GUI) from within an integrated development environment (IDE). IDEs typically feature additional tools such as an integrated debugger, text editor, and support for a wide range of source code control tools such as CVS.

You can run Python from a graphical user interface (GUI) environment as well. All you need is a GUI application on your system that supports Python. If you have found one, chances are that it is also an IDE (integrated development environment). IDEs are more than just graphical interfaces. They typically have source code editors and trace and debugging facilities.

Two primary ways that Python "does things" for you: statements and expressions (functions, equations, etc.). Most of you already know the difference between the two, but in case you need to review, a statement is a body of control which involves using keywords. It is similar to issuing a command to the interpreter. You ask Python to do something for you, and it will do it. Statements may or may not lead to a result or output.

CHAPTER 5

MODULE DESCRIPTION

MODULE DESCRIPTION

5.1. PROJECT DEFINITION

This project is about a security system that alerts the admin using face recognition. The proposed system integrated with OpenCV and machine learning techniques, this method decreases the needs of human effort developing its functionalities. It is easy to understand and fast to implement. It has the highest accuracy among all algorithms that predict images.

5.2. OVERVIEW OF THE PROJECT

In this section, the methodology of the proposed system for face detection, conversion to image and matching it with the data is described. The system will help significantly in the detection of multiple faces simultaneously without any inconvenience. The whole architecture can be divided into several modules consisting of pre-processing, feature extraction, and classification.

5.3. METHODOLOGY

Below are the methodology and descriptions of the applications used for data gathering, face detection, training and face recognition. The project was coded in Python using a mixture of IDLE and PyCharm IDEs.

- Face detection
- Face recognition process
- Face analysis
- Alerting

5.4. ARCHITECTURE DIAGRAM

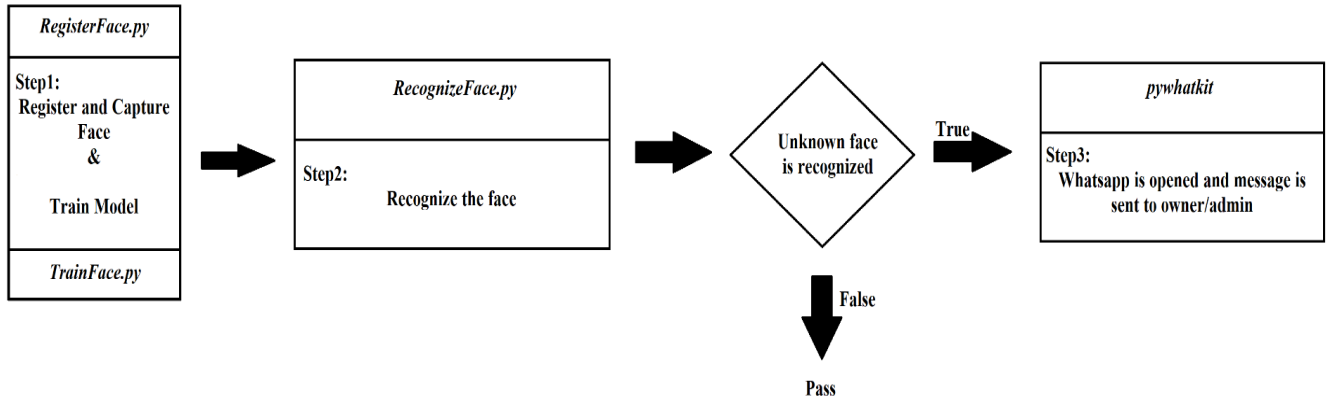


Fig. 5.1: Architecture diagram

5.5. MODULES

5.5.1. Face Recognition Process

For this project three algorithms are implemented independently. These are Eigenface, Fisher face and Linear binary pattern histograms respectively. All three can be implemented using OpenCV libraries. There are three stages for the face recognition as follows:

1. Collecting images IDs
2. Extracting unique features, classifying them and storing in XML files
3. Matching features of an input image to the features in the saved XML files and predict identity.

5.5.2 Collecting the image data

Collecting classification images is usually done manually using a photo editing software to crop and resize photos. Furthermore, PCA and LDA require the same number of pixels in all the images for the correct operation. This time consuming and a laborious task is automated through an application to collect 50 images with different expressions. The application detects suitable expressions between 300ms, straightens any existing tilt and saves them. The Flow chart for the application is shown in figure.

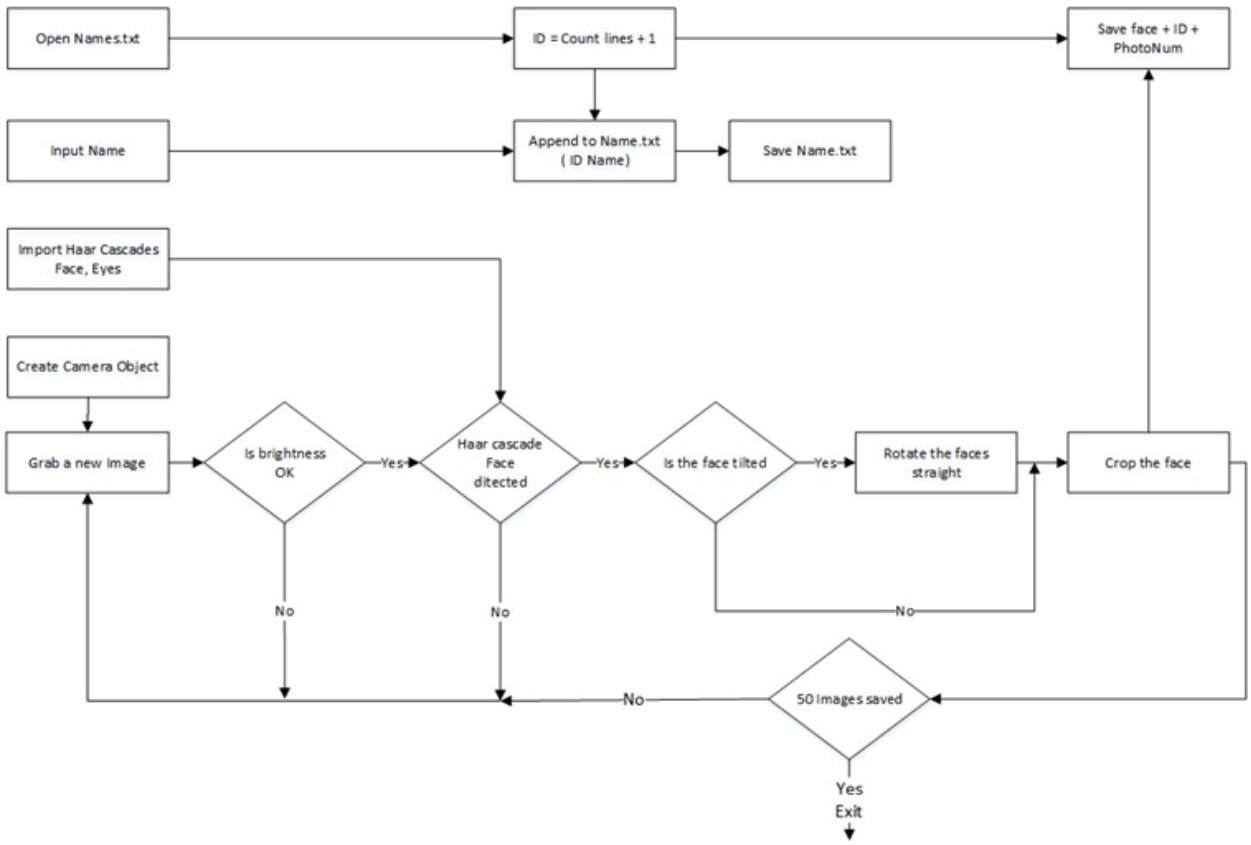


Fig. 5.2: The Flowchart for the image collection

Application starts with a request for a name to be entered to be stored with the ID in a text file. The face detection system starts the first half. However, before the

capturing begins, the application checks for the brightness levels and will capture only if the face is well illuminated. Furthermore, after the face is detected, the position of the eyes is analysed. If the head is tilted, the application automatically corrects the orientation. These two additions were made considering the requirements for the Eigenface algorithm. The Image is then cropped and saved using the ID as a filename to be identified later. A loop runs this program until 50 viable images are collected from the person. This application made data collection efficient.

5.5.2. Training the Classifiers

OpenCV enables the creation of XML files to store features extracted from datasets using the Face Recognizer class. The stored images are imported, converted to grayscale and saved with IDs in two lists with the same indexes. Face Recognizer objects are created using the face recogniser class. Each recogniser can take in parameters that are described below:

cv2.face.createEigenFaceRecognizer()

1. Takes in the number of components for the PCA for creating Eigenfaces. OpenCV documentation mentions 80 can provide satisfactory reconstruction capabilities.

2. Takes in the threshold in recognising faces. If the distance to the likeliest Eigenface is above this threshold, the function will return a -1, that can be used state the face is unrecognisable

cv2.face.createFisherfaceRecognizer()

1. The first argument is the number of components for the LDA for the creation of Fisher faces. OpenCV mentions it to be kept 0 if uncertain.

2. Similar to the Eigenface threshold. -1 if the threshold is passed.

cv2.face.createLBPHFaceRecognizer()

1. The radius from the centre pixel to build the local binary pattern.

2. The Number of sample points to build the pattern. Having a considerable number will slow down the computer.

3. The Number of Cells to be created in the X axis.

4. The number of cells to be created in the Y axis.

5. A threshold value similar to Eigenface and Fisher face. if the threshold is passed the object will return -1

Recogniser objects are created and images are imported, resized, converted into numpy arrays and stored in a vector. The ID of the image is gathered from splitting the file name, and stored in another vector. By using **FaceRecognizer.train(NumpyImage, ID)** all three of the objects are trained. It must be noted that resizing the images were required only for Eigenface and Fisher face, not for LBPH. Next, the configuration model is saved as an XML file using FaceRecognizer.save(FileName). In this project, all three are trained and saved through one application for convenience. The flow chart for the trainer is shown in figure.

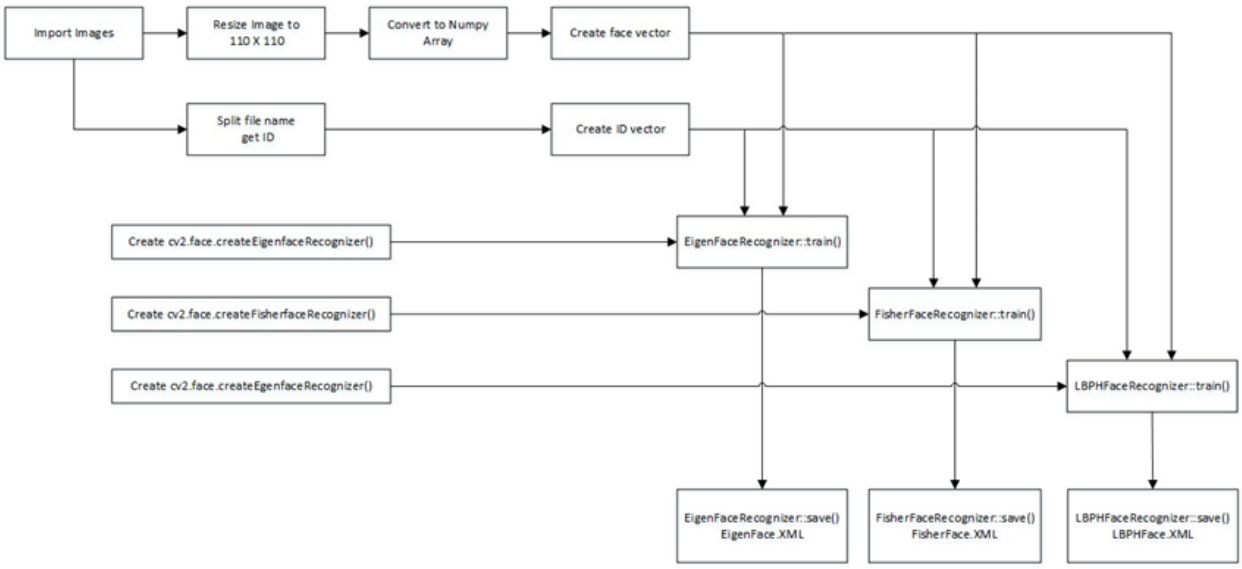


Figure 5.3: Flowchart of the training application

5.5.3. The Face Recognition

Face recogniser object is created using the desired parameters. Face detector is used to detect faces in the image, cropped and transferred to be recognised. This is done using the same technique used for the image capture application. For each face detected, a prediction is made using FaceRecognizer.predict() which returns the ID of the class and confidence. The process is the same for all algorithms and if the confidence is higher than the set threshold, ID is -1. Finally, names from the text file with IDs are used to display the name and confidence on the screen. If the ID is -1, the application will print an unknown face without the confidence level. The flow chart for the application is shown in figure.

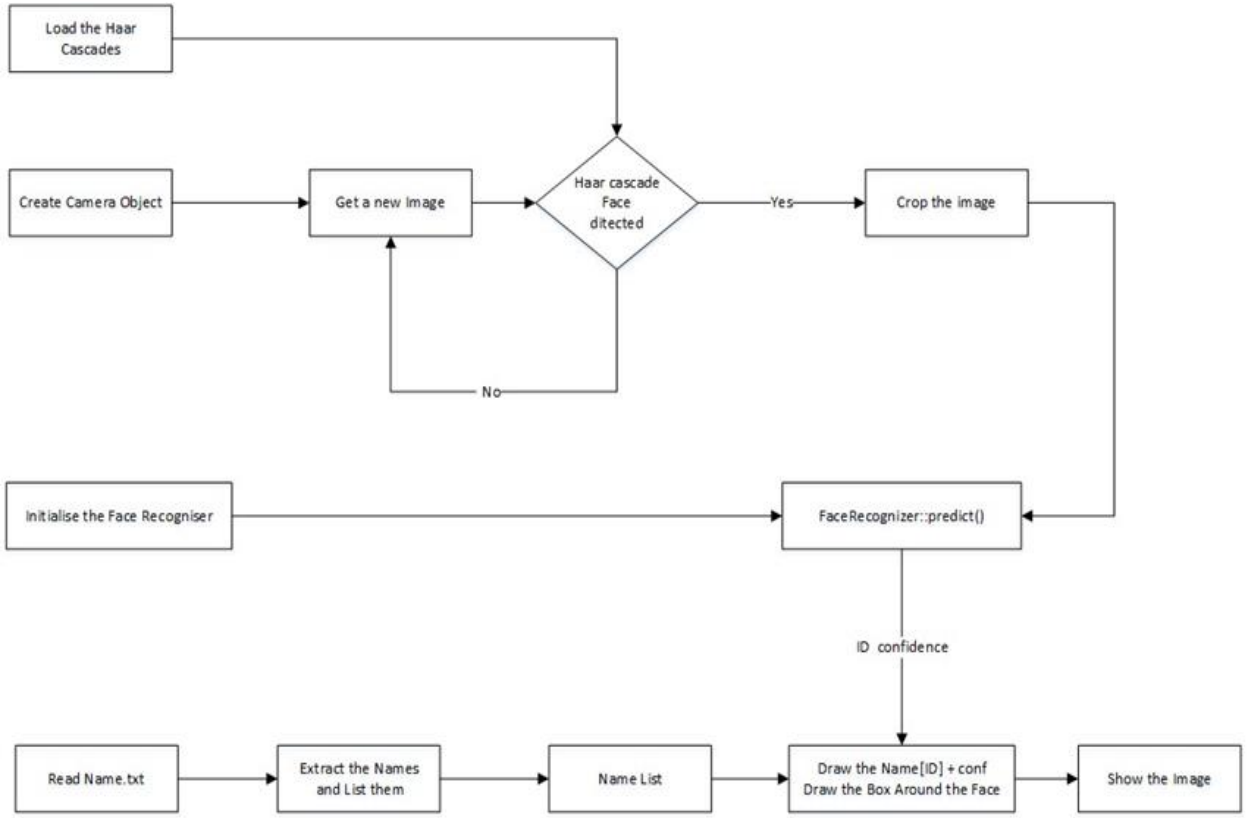


Fig:5.4: Flowchart of the face recognition application

5.5.4 Result

The collected images are shown below. Each face has 50 images. Three applications were written to iterate through the parameters of each algorithm. On each iteration, the algorithm is trained using different parameters and tested against a photo. The resulting data is plotted after finishing the tests. The applications are :

- **TestDataCollector EigenFace.py**
- **TestDataCollector FisherFace.py**
- **TestDataCollector LBPH.py**

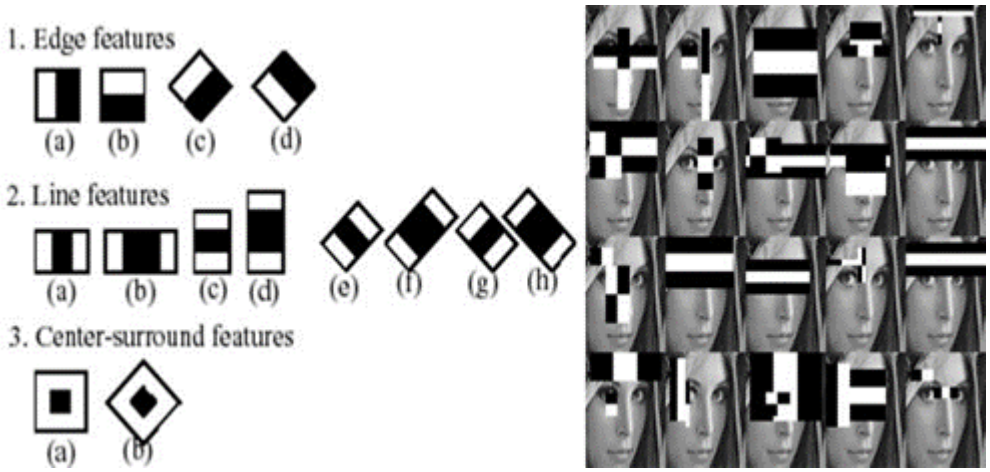
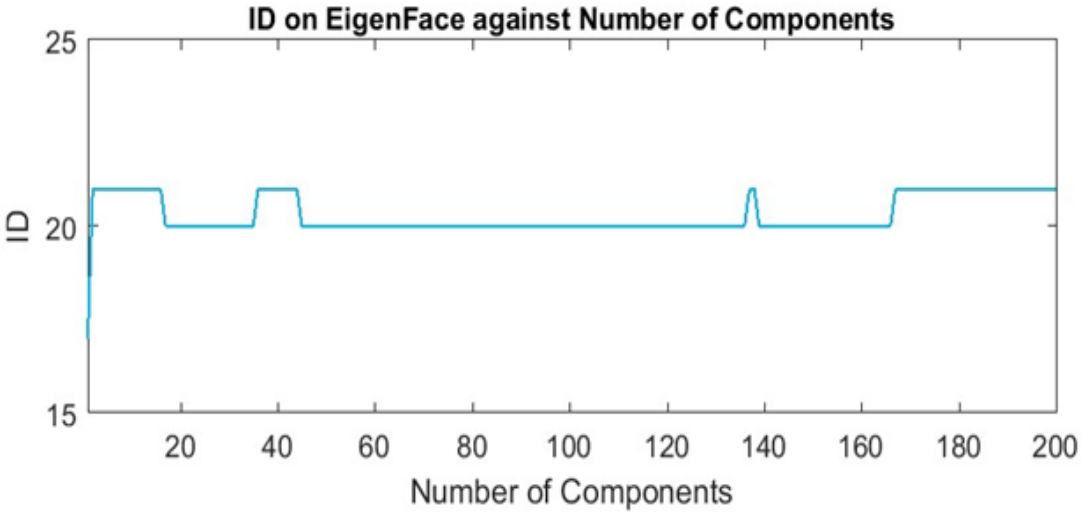


Fig:5.5: Dimensions of Face

The first test image is shown in figure 5.5 and the plots are analysed below. The resulting ID change is plotted below in figure 5.5. Note when components were 1, it identified the face as ID-1 and the rest as ID-2 and ID-21, which is the same person. The change of Confidence is plotted in figure, increasing with



components. From this plot it appears the best is when components are below 20

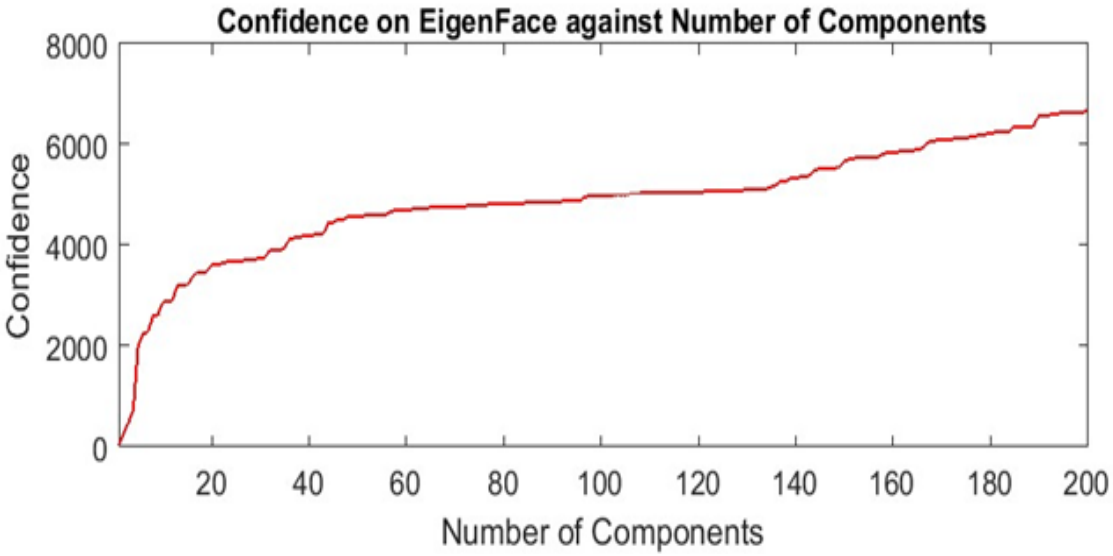


Fig:5.7: The Confidence increasing with No. of components

The ID results from Fisher face are more stable than Eigenface and are on ID-21 as seen in figure 5.8.

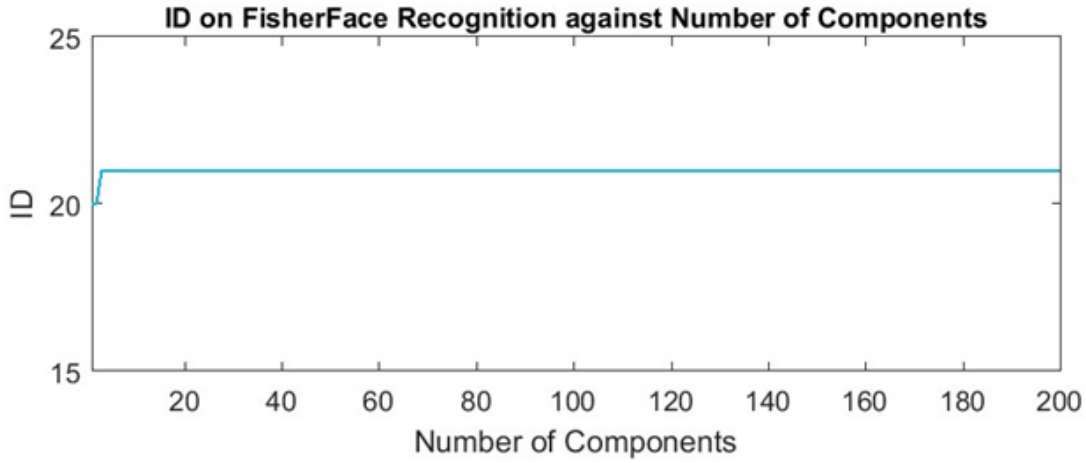
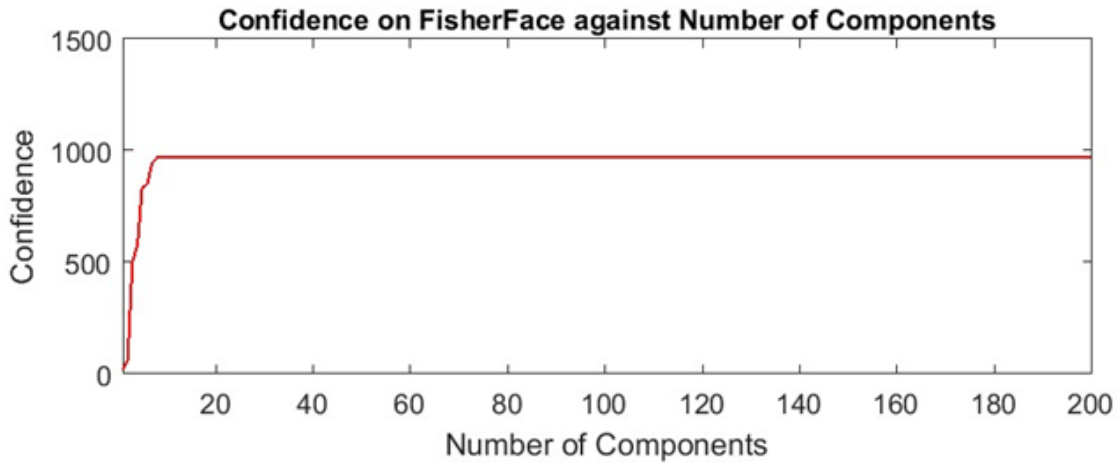


Fig:5.8: Fisherface ID is stable

Fisher face confidence increases in figure 16 until the number of components is 10 and will be used as the ideal value. LBPH has more than one parameter to change. All are incremented to the maximum



Fif:5.9: Stable confidence after 10 components

limit and the results are shown below. The first is the radius from the centre pixel and since the image size is 110 X 110, maximum radius is 54. The ID is steady all the way to 50 as can be seen in figure 17.

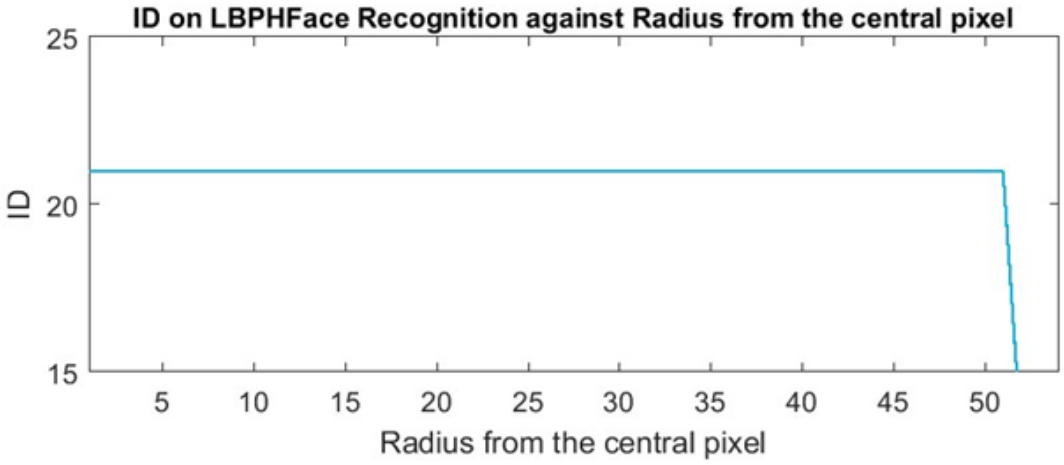


Fig:5.10 The ID returned from LBPH

Confidence level is graphed against the radius in figure 18. The confidence is fluctuating after 40. The lowest confidence level is at 2.

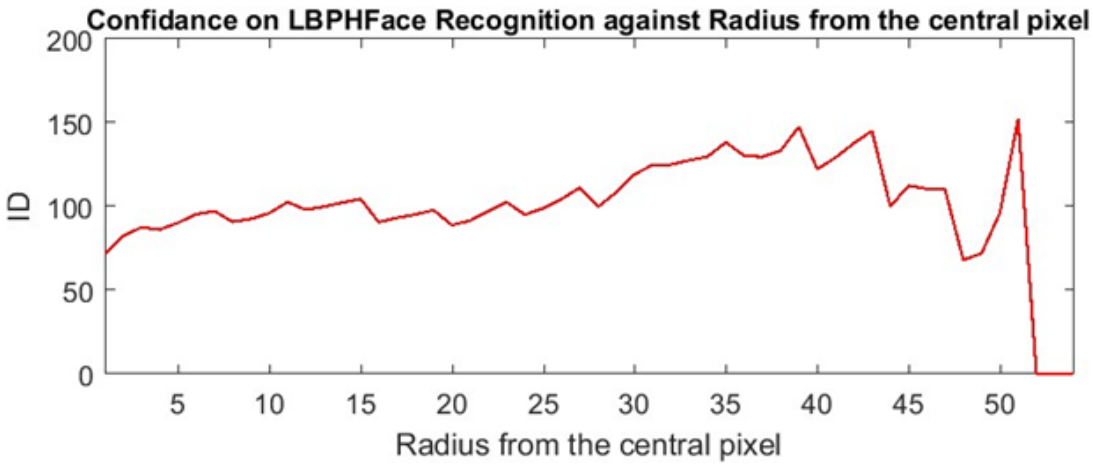


Figure 5.11: The confidence returned from LBPH

The number of neighbours are changed from 1 - 13. Further increase caused the computer to stall. The returned ID is plotted below in figure 19. ID steady until 9 neighbours and changed to ID-20.

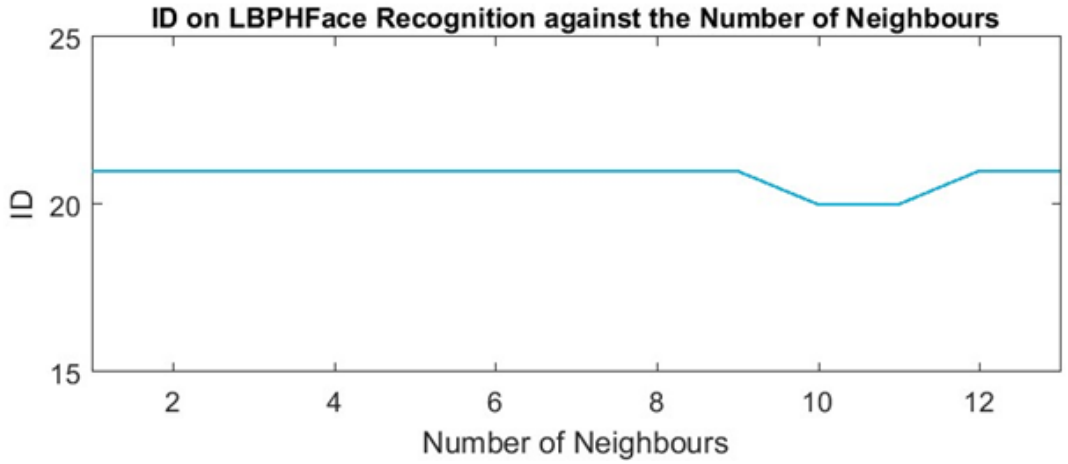


Fig:5.12: The ID returned from LBPH changing neighbours

The confidence continuously increased as can be seen in figure 20 and 1 neighbour will be included to the next test.

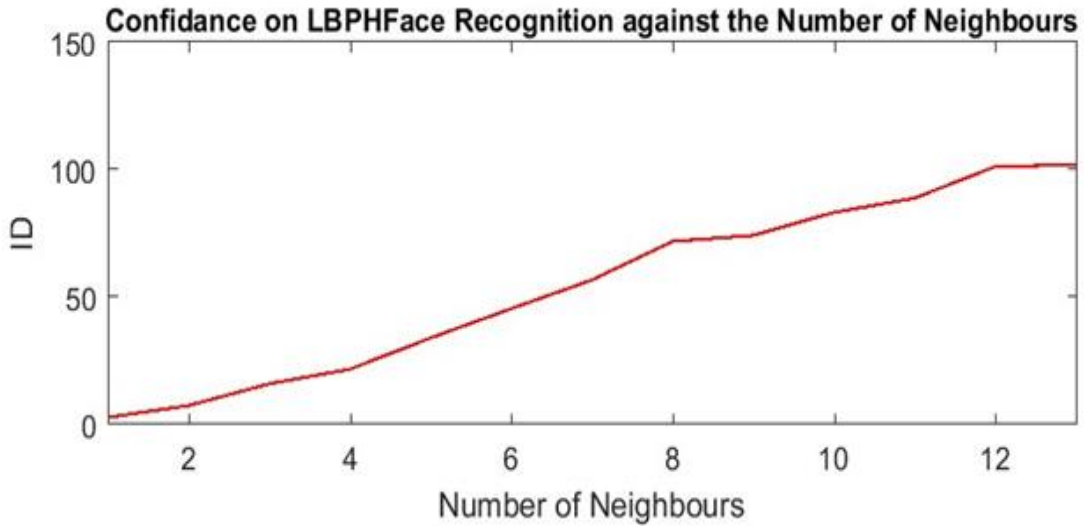


Fig:5.13: The Confidence returned from LBPH changing neighbours

The number of cells in X and Y directions are changed simultaneously. ID return is plotted below in figure 21. The ID changed from ID-20 to ID -21 and stayed steady.

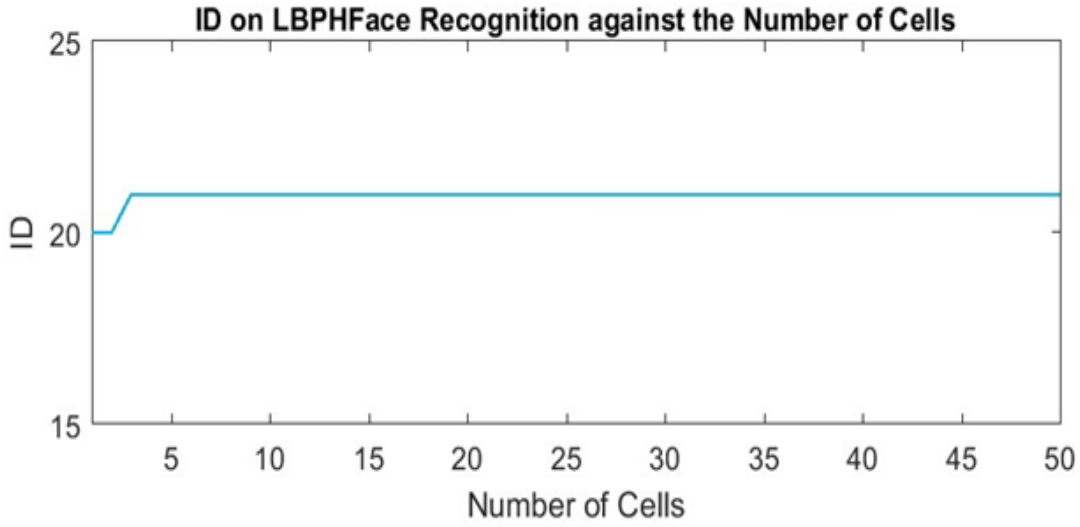


Fig:5.14: The ID returned from LBPH changing the number of cells

The returned confidence is plotted in figure 22. The confidence was low when cells were less than 8 per side and increased rapidly after 10 ending up more than 1700 at 50.

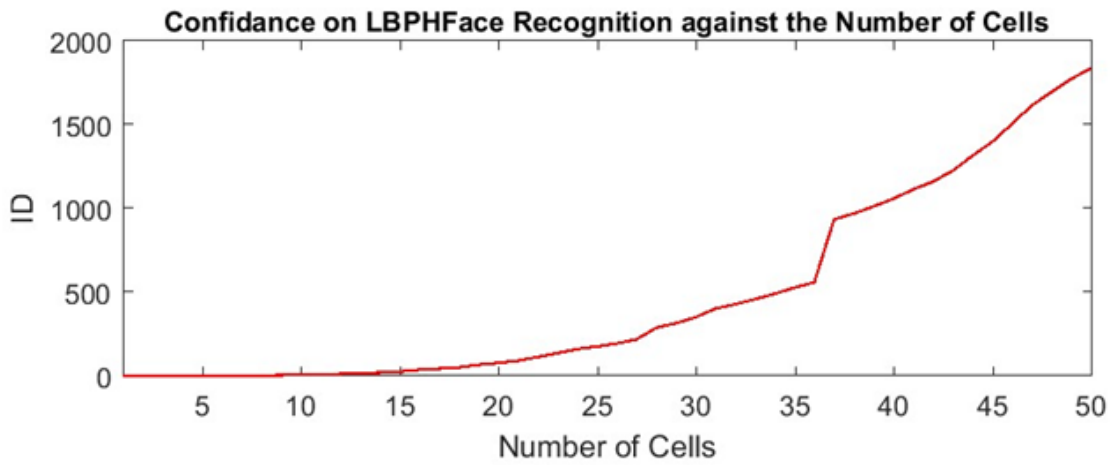


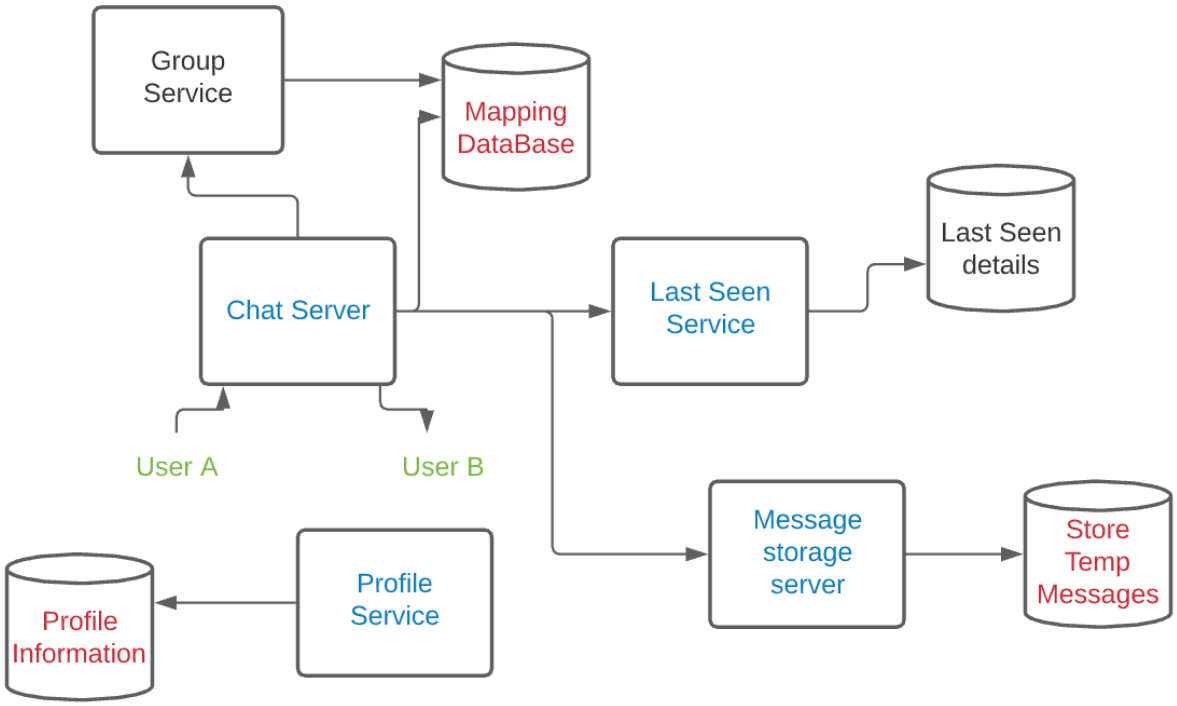
Fig:5.15: The confidence returned from LBPH changing the number of cells

5.6. Alerting message via WhatsApp

WhatsApp is free to download messenger apps for smartphones. WhatsApp uses the internet to send messages, images, audio or video. The service is very similar to text messaging services, however, because WhatsApp uses the internet to send messages, the cost of using WhatsApp is significantly less than texting. You can also use WhatsApp on your desktop, simply go to the WhatsApp website and download it to Mac or Windows. It is popular with teenagers because of features like group chatting, voice messages and location sharing.

The popularity of WhatsApp is because it's free, it doesn't have ads, and of course, it has a huge existing user base. It also features end-to-end encryption, so your messages are private and secure. However, security policy changes in early 2021 mean the app can share some of your data with Facebook. Learn more about that issue in our WhatsApp privacy explainer.

In addition to standard one-on-one and group chats, you can also do voice or video calls free of charge, even internationally. Remember that this will eat up your data when not connected to Wi-Fi.



This is how the server and owner/admin interact with each other.

CHAPTER 6

SYSTEM TESTING

SYSTEM TESTING

6.1. CODE REVIEW

After successful completion of coding, Code review was done with the objective of identifying and correcting deviations from standards, Identifying and fixing logical bugs and fall through and recording code walk through findings. The programs were checked and the code structure was made readable. The variable names were meaningful. It follows certain naming conventions, which makes the program readable.

- Variable names are prefixed with their scope and data type
- Checking out for the correct scope for various function
- All possible explanation for the code were given as comment
- Sufficient labels and comments were included in the code as the description of it for the benefit of the developer and other programmers who might examine it later.
- Checking out the connectivity of the database
- Code optimization was carried out

Software testing is a survey conducted to provide interested parties with information about the quality of the software product or service being tested. Software testing can also provide an objective, independent view of the software, allowing the business to understand and understand the risks associated with implementing software. Test methods include the process of running a program or application to detect software errors and verify that the software product is usable.

6.2. TESTING PROCESS

- Testing is the process of executing the program with the intent of finding an error.
- A good test has a high probability of finding an undiscovered error.
- A successful test is one that uncovers an as yet undiscovered error.

The objective is to design tests that systematically uncover different classes of error and do so with the minimum amount of time and effort. Testing cannot show the absence of defects, it can only show that the software defects are present.

6.2.1. System Testing

System testing is performed on the entire system in the context of a specification of functional requirements or a specification of system requirements, or both. It is also designed to test before and outside the limits specified in the specification of software or hardware requirements.

6.2.2. Unit Testing

Test modules are software test methods that test individual source code modules, set up one or more modules for computer programs, and control data, usage procedures, and work procedures to determine whether they are compatible to use. Unit tests are usually performed as part of a combination of code and the software life cycle module testing phase, although unusual coding and unit testing will be performed as two separate phases.

Test strategy and approach

Field testing will be performed manually

Features to be tested

- Able to register the face without any kind of issues.
- Once, the program is started it will to monitor the peoples' face
- If the face matches our database, then it will send a message as he/she is entered with their name. Else, it will send a message as Unknown is entered.

6.2.3. Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results

All the test cases mentioned above passed successfully. No defects encountered.

6.2.4. Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results

All the test cases mentioned above passed successfully. No defects encountered.

6.3. EXPERIMENTAL OUTCOMES

Table 6.1: Experimental Result

Images	**Face**	**Overall Image Count**	**Approx. Accuracy**
Trained	2	12	(70.00% and above)
Tested	10		

CHAPTER 6

SYSTEM IMPLEMENTATION

SYSTEM IMPLEMENTATION

7.1. IMPLEMENTATION PROCEDURE

Implementation is the stage of the project when the theoretical design is turned out into a working system. Each program is tested individually at the time of development using the data and has verified that this program linked together in the way specified in the programs specification, the computer system and its environment is tested to the satisfaction of the user. The final stage is to document the entire system which provides components and the operating procedures of the system.

7.2. CONVOLUTION OPENCV

The OpenCV tutorial provides basic and advanced concepts of OpenCV. Our OpenCV tutorial is designed for beginners and professionals.

OpenCV is an open-source library for computer vision. It provides the facility to the machine to recognize faces or objects. In this tutorial we will learn the concept of OpenCV using the Python programming language.

Our OpenCV tutorial includes all topics of Read and Save Image, Canny Edge Detection, Template matching, Blob Detection, Contour, Mouse Event, Gaussian blur and so on.

OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, a three-dimensional model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and act accordingly.

Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency. There are a two main task which are defined below:

- Object Classification - In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories.
- Object Identification - In the object identification, our model will identify a particular instance of an object

How does the computer recognize the image?

Human eyes provide lots of information based on what they see. Machines are facilitated with seeing everything, convert the vision into numbers and store in the memory. Here the question arises how computers convert images into numbers. So the answer is that the pixel value is used to convert images into numbers. A pixel is the smallest unit of a digital image or graphics that can be displayed and represented on a digital display device.

The picture intensity at the particular location is represented by the numbers. In the above image, we have shown the pixel values for a grayscale image consist of only one value, the intensity of the black colour at that location.

There are two common ways to identify the images:

1. Grayscale

Grayscale images are those images which contain only two colours black and white. The contrast measurement of intensity is black treated as the weakest intensity, and white as the strongest intensity. When we use the grayscale image, the computer assigns each pixel value based on its level of darkness.

2. RGB

An RGB is a combination of the red, green, blue colour which together makes a new colour. The computer retrieves that value from each pixel and puts the results in an array to be interpreted.

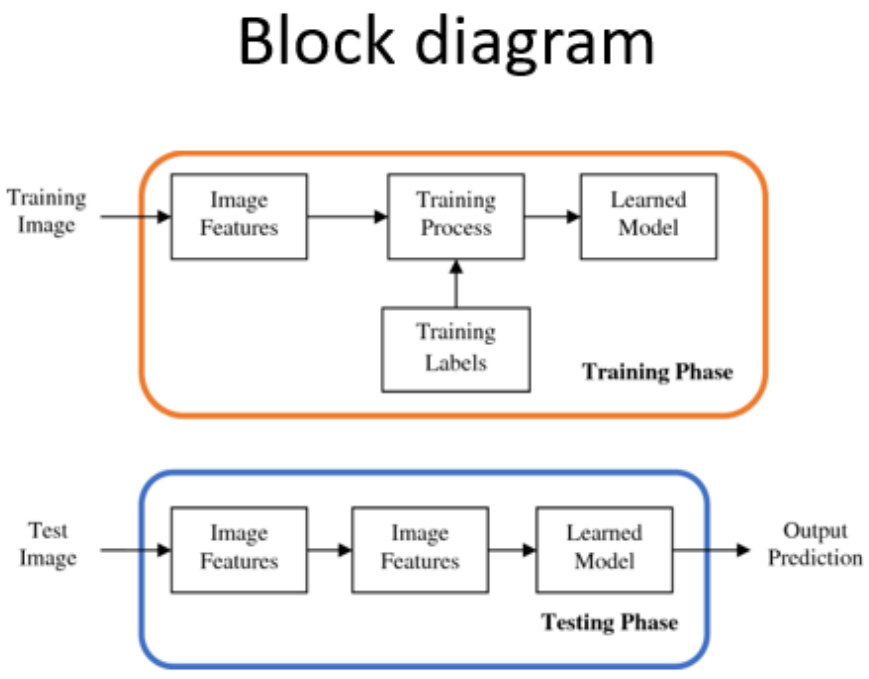


Fig. 7.1: Block Diagram

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION AND FUTURE ENHANCEMENTS

8.1. CONCLUSION

This paper describes the mini-project for visual perception and autonomy modules. Next, it explains the technologies used in the project and the methodology used. Finally, it shows the results, discusses the challenges and how they were resolved followed by a discussion. Using Haar-cascades for face detection worked extremely well even when subjects wore spectacles. Real time video speed was satisfactory as well devoid of noticeable frame lag. Considering all factors, LBPH combined with Haar-cascades can be implemented as a cost-effective face recognition platform. An example is a system to identify known troublemakers in a mall or a supermarket to provide the owner a warning to keep him alert or for entry of unfamiliar people's entry.

8.2. FUTURE ENHANCEMENTS

The scope of the project is more interactive and user-friendly. We are planning to integrate this machine learning project as a web application with an advanced python framework and the open-source platform called OpenCV. The OpenCV library makes programming easy to use. This comes up with advanced proficiencies like face detection, face tracking, facial recognition, and many more methods for artificial intelligence (AI).

By using this we are going to detect the faces of the persons passing through the camera, and check whether their photo matches with the data provided. If it matches it sends the name of the person entered or else sends unknown person entered / unknown person detected, then we can monitor through online using Wi-Fi and smart devices.

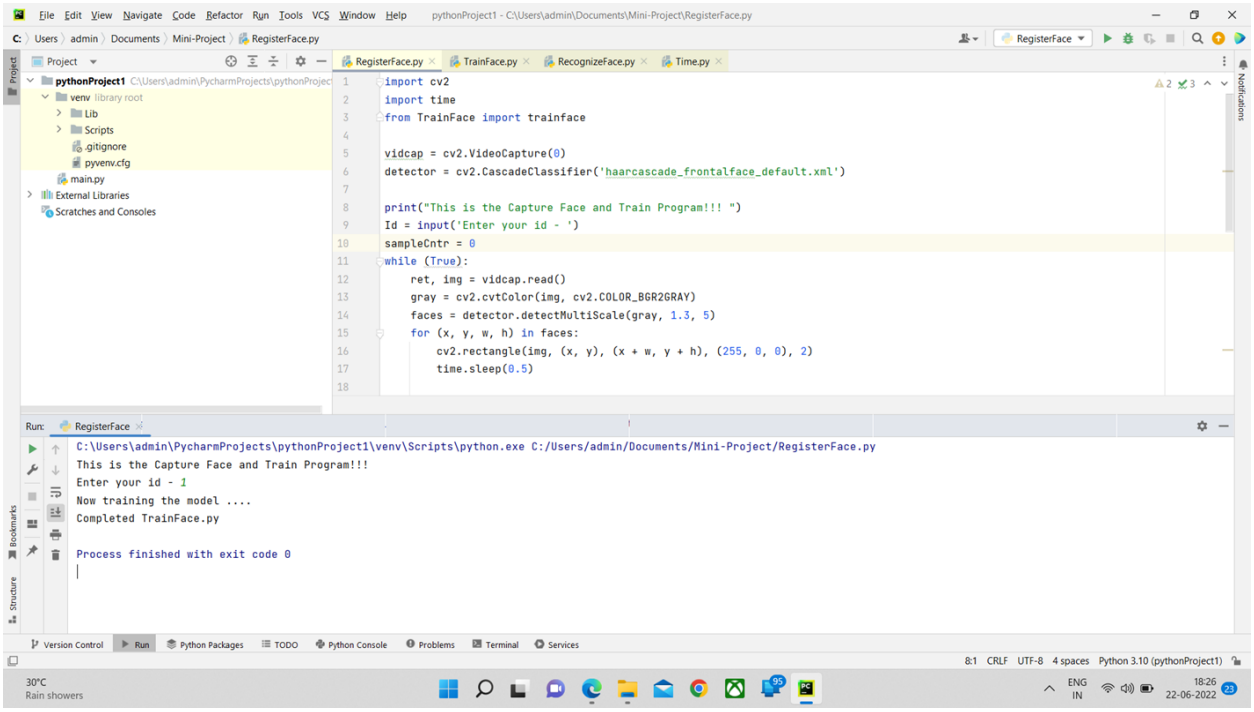
APPENDICES

APPENDIX - 1

SCREENSHOTS

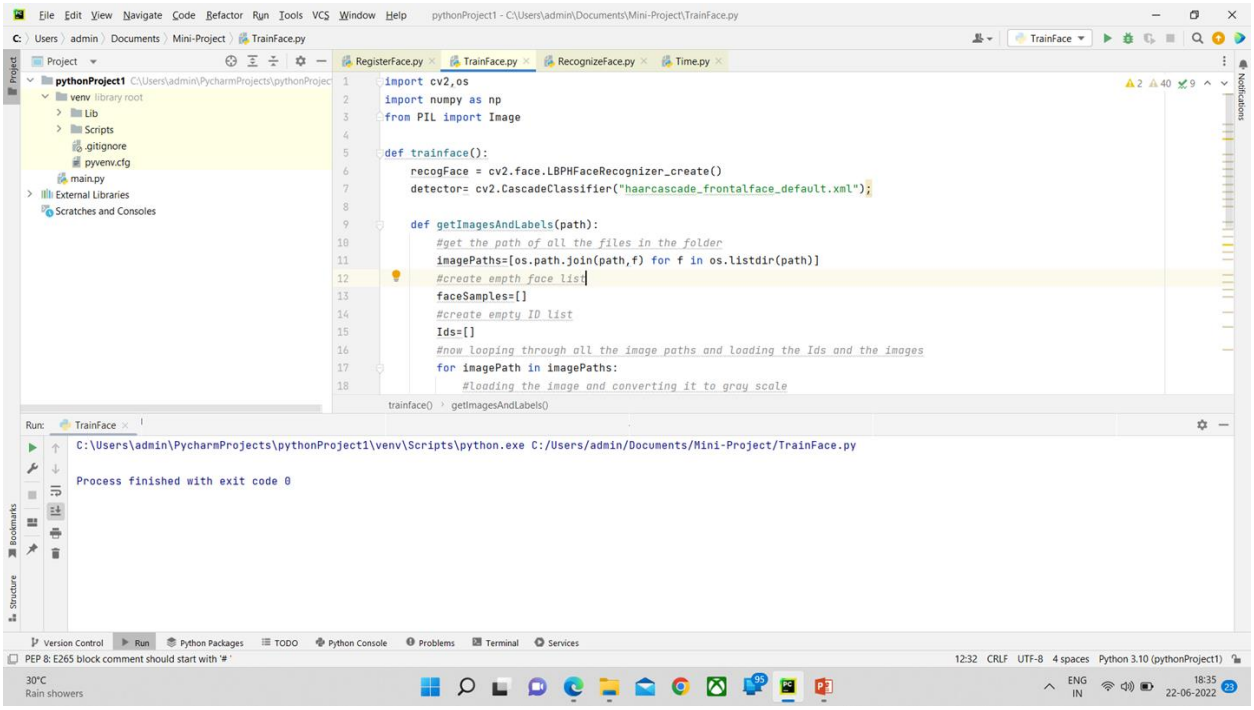
Step 1: Register the face to your dataset

This is the snapshot of the module -Face-Registration



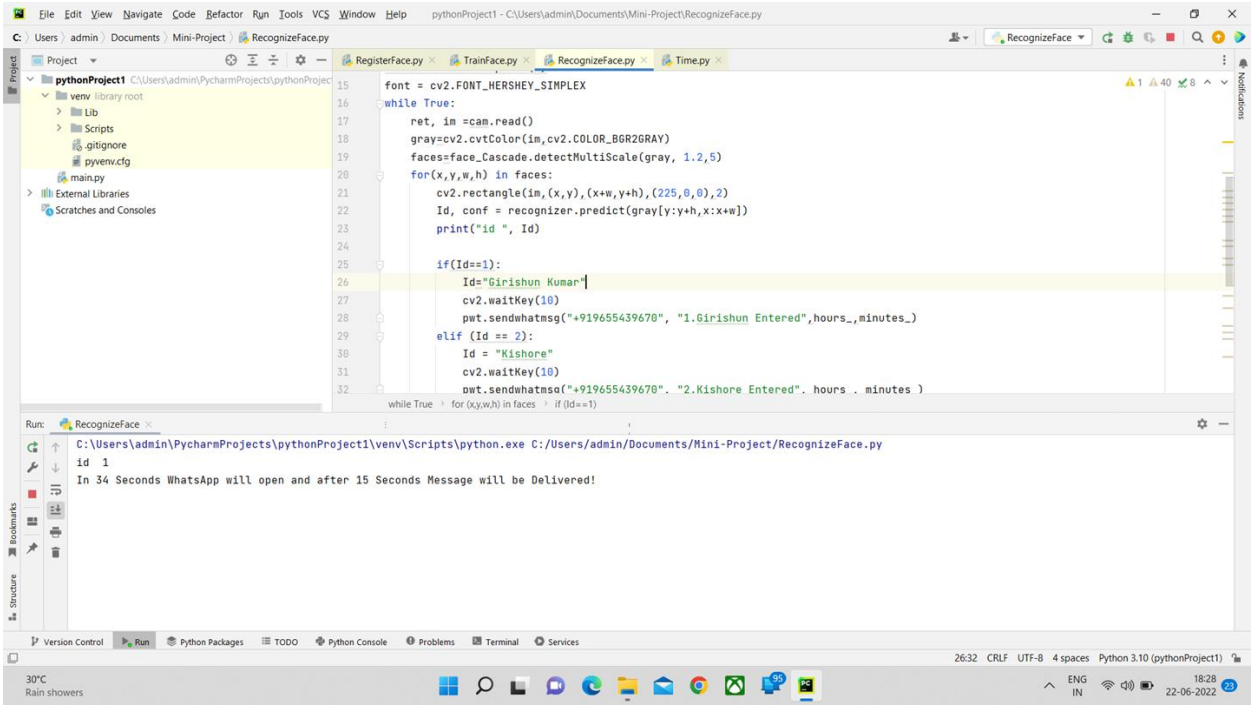
Step 2: Train your face in different modules

This is the snapshot of the module- Face-Trainninng.



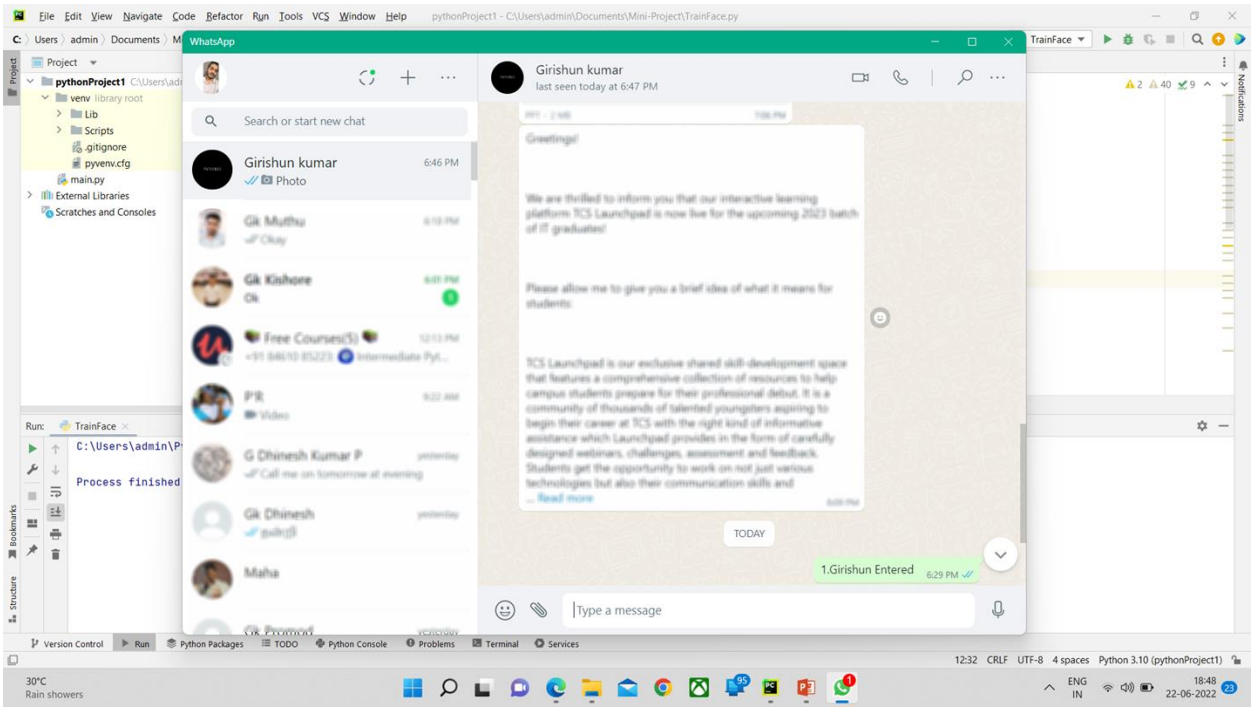
Step 3: Finally, the face recognition is started

This is the snapshot of the module – Face-Recognition



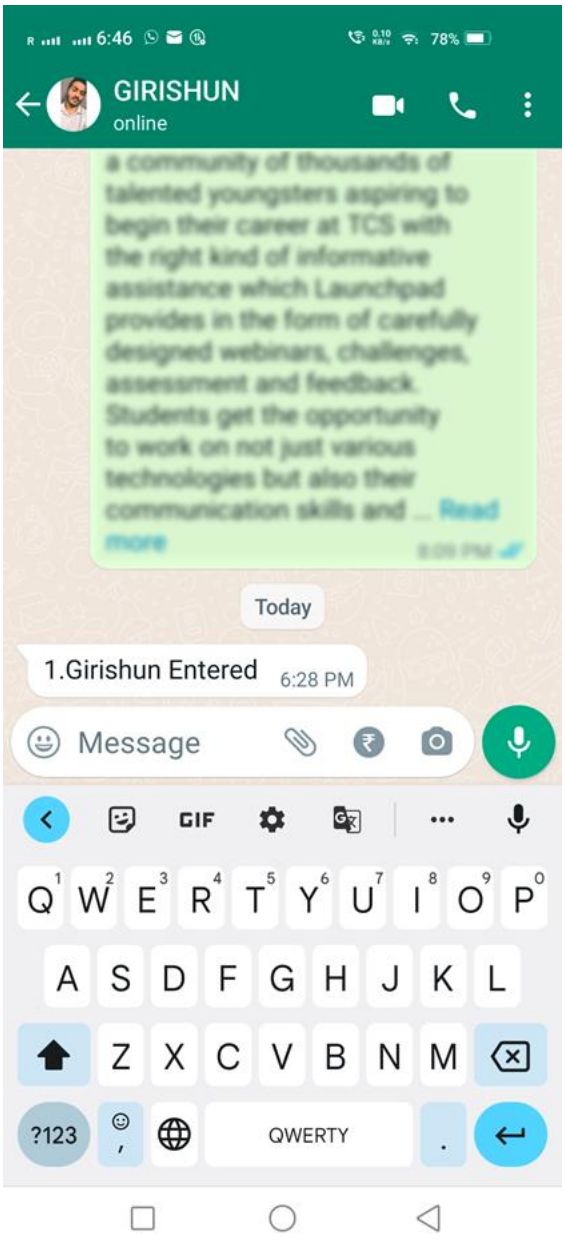
Step 4: Now, If the face matches our database, then it will send a message as he/she is entered with their name. Else, it will send a message as Unknown is entered.

This is the snapshot of the module –Alerting message through WhatsApp.



Step 5: Admin/Owners receiving the message.

This is the snapshot of the module –Admin/Owners' screenshot of receiving the message



APPENDIX – 2

IMPLEMENTATION CODE

RegisterFace.py(for registering face to database)

```
import cv2
import time
from TrainFace import trainface

vidcap = cv2.VideoCapture(0)
detector=
cv2.CascadeClassifier('haarcascade_frontalface_default.
xml')

print("This is the Capture Face and Train Program!!! ")
Id = input('Enter your id - ')
sampleCntr = 0
while (True):
    ret, img = vidcap.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)
        time.sleep(0.5)

        sampleCntr = sampleCntr + 1
```

```
        # saving the captured face in the dataset folder
        cv2.imwrite("dataSet/User."   +   Id   +   '.'   +
str(sampleCntr) + ".jpg", gray[y:y + h, x:x + w])

        cv2.imshow('frame', img)

    if cv2.waitKey(100) & 0xFF == ord('q'):
        break
    # Collect 10 samples
    elif sampleCntr > 11:
        break
vidcap.release()
cv2.destroyAllWindows()
time.sleep(1)
print("Now training the model .... ")
trainface()
```

TrainFace.py (for taining dataset and building the model)

```
import cv2,os
import numpy as np
from PIL import Image

def trainface():
    recogFace = cv2.face.LBPHFaceRecognizer_create()
    detector=
     cv2.CascadeClassifier("haarcascade_frontalface_defa
     ult.xml");

    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)]
    #create empth face list
    faceSamples=[]
    #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=detector.detectMultiScale(imageNp)
        #If a face is there then append that in the
list as well as Id of it
        for (x,y,w,h) in faces:
            faceSamples.append(imageNp[y:y+h,x:x+w])
            Ids.append(Id)
    return faceSamples,Ids
```

```
    faces,Ids = getImagesAndLabels('dataSet')
    recogFace.train(faces, np.array(Ids))
    recogFace.save('trainner/trainner.yml')
    print("Completed TrainFace.py")
```

RecognizeFace.py

```
import cv2
import numpy as np
import pywhatkit as pwt
from Time import *


recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainner/trainner.yml')
face_Cascade =
cv2.CascadeClassifier("haarcascade_frontalface_default.
xml");



cam = cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_SIMPLEX
while True:
    ret, im =cam.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces=face_Cascade.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])
        print("id ", Id)
```

```
        if(Id==1):
            Id="Girishun Kumar"
            cv2.waitKey(10)
            pwt.sendwhatmsg("+919655439670", "1.Girishun
Entered",hours_,minutes_)
        elif (Id == 2):
            Id = "Kishore"
            cv2.waitKey(10)
            pwt.sendwhatmsg("+919655439670", "2.Kishore
Entered", hours_, minutes_)
        else:
            Id="Unknown"
            cv2.waitKey(10)
            pwt.sendwhatmsg("+919655439670", "5.Unknown
Entered",hours_,minutes_)



        cv2.putText(im,str(Id),
(x,y+h),font,0.55,(255,255,255),1)
    cv2.imshow('im',im)
    if cv2.waitKey(10) & 0xFF==ord('q'):
        break
cam.release()
cv2.destroyAllWindows()
```

Time.py

```
import datetime



# datetime object
dt = datetime.datetime.now()
# printing in dd/mm/YY H:M:S format using strftime()
hours_ = dt.strftime("%H")
hours_=int(hours_)
minutes_=dt.strftime('%M')
minutes_=int(minutes_)
seconds_=dt.strftime('%S')
seconds_=int(seconds_)
if minutes_ == 58 or minutes_ == 59:
   hours_ += 1
if seconds_ >= 45 or seconds_ <= 60 :
   minutes_ += 1
```

REFERENCES

[1] G.T. Rado, H. Suhl, I.S. Jacobs and C.P. Bean, "*Fine particles thin films and exchange anisotropy*" in International Journal of Soft Computing and Artificial Intelligence, New York: Academic, vol. III, pp. 271-350, 1990, ISBN 2321-404X.

[2] M.A. Turk and A.P. Pentland, "*Face Recognition Using Eigenfaces*", IEEE Conf. on Computer Vision and Pattern Recognition, vol.II, 1991.

[3] KyungnamKim" *Face Recognition using Principle Component Analysis* Conference paper-IEEE, vol.III, 2001.

[4] G B Huang, H Lee and EL. Miller, "*Learning hierarchical representation for Face verification with convolutional deep belief networks[C]*", Proceedings of International Conference on Computer Vision and Pattern Recognition, pp. 223-226, 2012.

[5] Paul Viola and Matthew Jones, "*Rapid object detection using a boosted cascade of simple features*", Conference paper-IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol.III, 2006.

[6] Ravishankara K, Dhanush, Vaisakh, Srajan I S "*Whatsapp Chat Analyzer*", Conference paper-IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol.III, 2011.

[7] S. Jacobs and C. P. Bean, "*Fine Particles, Thin Films and Exchange Anisotropy*" In Magnetism, G. T. Rado and H. Suhl, Eds., Academic, New York, Vol. 3, 1963, pp. 271-350.

[8] Kruti Goyal, Kartikey Agarwal, Rishi Kumar "*Face detection and tracking: Using OpenCV*", Conference paper-IEEE, vol.IV, 2011.