

ATTENDANCE CAPTURE SYSTEM USING FACE RECOGNITION

*A project report submitted to the Faculty of Engineering of
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA*

In Partial fulfilment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY In COMPUTER SCIENCE AND ENGINEERING

Submitted By

E.POOJITHA (20NH1A0521)

K.SANJEEV (20NH1A0549)

K.SASHANK ROY (20NH1A0546)

L.NAGA RAJU (20NH1A0556)

Under the Esteemed Guidance of

Mr. G.Venkata Ratnam M.Tech., (Ph.D)

Associate Professor



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

V.K.R, V.N.B & A.G.K COLLEGE OF ENGINEERING

(An ISO 9001:2015 certified institution, Accredited by NAAC with 'A' grade)

(Sponsored by General & Technical Education Society, Gudivada)

(Approved by AICTE-New Delhi, Recognized by Govt. of A.P & Affiliated to JNTU - KAKINADA)

GUDIVADA-521301, Krishna District, Andhra Pradesh.

MAY-2024

ATTENDANCE CAPTURE SYSTEM USING FACE RECOGNITION

*A project report submitted to the Faculty of Engineering of
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA
In Partial fulfilment of the requirement for the award of degree of*

BACHELOR OF TECHNOLOGY In COMPUTER SCIENCE AND ENGINEERING

Submitted By

E.POOJITHA (20NH1A0521)

K.SANJEEV (20NH1A0549)

K.SASHANK ROY (20NH1A0546)

L.NAGA RAJU (20NH1A0556)

Under the Esteemed Guidance of

Mr. G.Venkata Ratnam M.Tech., (Ph.D)

Associate Professor



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

V.K.R, V.N.B & A.G.K COLLEGE OF ENGINEERING

(An ISO 9001:2015 certified institution, Accredited by NAAC with 'A' grade)

(Sponsored by General & Technical Education Society, Gudivada)

(Approved by AICTE-New Delhi, Recognized by Govt. of A.P & Affiliated to JNTU - KAKINADA)

GUDIVADA-521301, Krishna District, Andhra Pradesh.

MAY-2024

V.K.R, V.N.B & A.G.K COLLEGE OF ENGINEERING

(An ISO 9001:2015 certified institution, Accredited by NAAC with 'A' grade)

(Sponsored by General & Technical Education Society, Gudivada)

(Approved by AICTE-New Delhi, Recognized by Govt. of A.P & Affiliated to JNTU - KAKINADA)

GUDIVADA-521301, Krishna District, Andhra Pradesh.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that **E.POOJITHA(20NH1A0521)**, **K.SANJEEV(20NH1A0549)**, **K.SASHANK ROY (20NH1A0546)**, **L.NAGA RAJU (20NH1A0556)** of final year computer science and Engineering has submitted MAJOR PROJECT report entitled **“Attendance Capture System Using Face Recognition ”** in partial fulfilment for the award of Bachelor of Technology Degree of JNTUK Kakinada in session 2023-24. It has been found to be satisfactory and here by approved for submission.

Mr. G.VENKATA RATNAM, M.Tech, (Ph.D)

(HoD of CSE Dept & Project Guide)

ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We extend our sincere gratitude to our esteemed project guide for his unwavering support to **Mr. G.VENKATA RATNAM M.Tech, (Ph.D)**, Assistant Professor for his innovative idea, dedicated support and encouraging us constantly throughout this project venture. We are also grateful for his constant availability and detailed supervision. Furthermore, we are also grateful to his keen interest in this project.

We feel elated to extend our heartfelt gratitude, akin to a bouquet of vibrant flowers blooming with appreciation to **Mr. G.VENKATA RATNAM M.Tech, (Ph.D)** Head of the Department of Computer Engineering, for his encouragement all the way during the analysis of the project. His annotations and criticisms are the key behind the successful completion of the project work.

We would like to take this opportunity to express our profound sense of gratitude to our beloved Principal **Dr. S.H.V.PRASADA RAO M.Tech, Ph.D** for providing us all the required facilities.

We thank the Teaching and Non-Teaching staff of Computer Science and Engineering Department who helped directly and indirectly in completing of our Project Work.

SUBMITTED BY:

E.POOJITHA (20NH1A0521)

K.SANJEEV (20NH1A0549)

K.SASHANK ROY (20NH1A0546)

L.NAGA RAJU (20NH1A0556)

INDEX

CONTENT	PGNO
ABSTRACT	i
List Of Figures	ii
1. INTRODUCTION	1-8
2. LITERATURESURVEY	9-12
3. OVERALL DESCRIPTION	13-14
3.1. Feasibility Report	
4. SYSTEM ANALYSIS	15-16
4.1 Existing System	
4.2 Proposed System	
4.3 Software Requirements	
4.4 Hardware Requirement	
5. SYSTEM DESIGN	17-20
5.1 System Architecture	
5.2 UML Diagrams	
6. IMPLEMENTATION	21-49
7. CODING	50-58
8. TESTING	59-60
9. RESULTS	61-64
10. CONCLUSION	65
11 FUTURE ENHANCEMENTS	66
12 REFERENCES	67

Abstract

The one important asset of our country is Bank currency and to create discrepancies of money miscreants introduce the fake notes which resembles to original note in the financial market. During demonetization time it is seen that so much of fake currency is floating in market. In general, by a human being, it is very difficult to identify forged note from the genuine not instead of various parameters designed for identification as many features of forged note are similar to original one. To discriminate between fake bank currency and original note is a challenging task. So, there must be an automated system that will be available in banks or in ATM machines. To design such an automated system there is need to design an efficient algorithm which is able to predict whether the banknote is genuine or forged bank currency as fake notes are designed with high precision.

LIST OF FIGURES

FIG.NO	CONTENTS	PGNO
5.1	System Architecture	17
5.2.1	Usecase Diagram	18
5.2.2	Sequence Diagram	19
5.2.3	Activity Diagram	20
6.1.2.1	Python Installation	22
6.1.2.2	Python Installation	23
6.1.2.3	Python Advanced Options	23
6.1.2.4	Setup Progress	24
6.1.2.5	Setup was Successful	25
6.1.2.6	Opening Python	25
6.1.2.7	Python Prompt	26
6.1.2.8	Checking Python Installation	26
6.2.5	Display Image In Window	30
6.10.2.1	Identifying Parts in The Face	42
6.10.2.2	Identifying Face in a Image	43
6.11.1	Pixels Value Calculation in Matrix	44
6.11.2	Form Circular Local Binary Pattern	45
6.11.3	Grid Conversion	46
9.1-9.10	Project Stages Login To Capture	61-64

1. INTRODUCTION

1.1 Purpose of this Project:

One of the foremost reminiscences everyone has about college is the morning roll call that the lecturers would in person call upon our names, and we tend to reply in affirmation to prove our attendance. It's a long and tedious routine in educational institutions. Attendance being a very important side of administration might usually become a time constraint, repetitive job, loaning itself to inaccuracies. Managing student's attendance at lecture periods has become a tough challenge. The ability to work out the attendance proportion becomes a significant task as manual computation produces errors, and wastes a great deal of our time. The basis of developing an automatic attendance management system is to computerize the standard method of taking attendance. The proposed system strives to outgrow the constraints of the existing systems and provides features such as detection of faces, extraction of features, detection of extracted features and analysis of student's attendance.

1.2 Scope of this Project

In this project, we will keep track about the attendance of students using their Faces, Generally attendance will be taken by calling roll numbers or using Biometric Systems. It is easy to proxy them. But in our project, there is a very minute chance for proxy and mistakes. It is a fully automated system for taking attendance and storing the reports from time to time.

1.3 Intended Audience and Reading Suggestions and Document Overview

The information contained in this document is intended for general distributions. However, it is especially important to keep the details like student name and student roll number. The document as the responsibility rests with them to ensure that the guidelines contained in it are followed. The guidelines should also be brought to the attention of all staff whose work involves the handling of student data.

1.4 Introduction To Anaconda

1.4.1 Introduction

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository.

1.4.2 Anaconda Distribution

Anaconda is a package manager, an environment manager, a Python/R data science distribution, and a collection of over 7,500+ open-source packages. Anaconda is free and easy to install, and it offers free community support. Get the Anaconda Cheat Sheet and then download Anaconda. Want to install conda and use conda to install just the packages you need? Get Miniconda.

After you install Anaconda or Miniconda, if you prefer a desktop graphical user interface (GUI) then use Navigator. If you prefer to use Anaconda prompt (or terminal on Linux or macOS), then use that and conda. You can also switch between them.

To try Navigator, after installing Anaconda, click the Navigator icon on your operating system's program menu, or in Anaconda prompt (or terminal on Linux or macOS), run the command `anaconda-navigator`. To try conda, after installing Anaconda or Miniconda, take the 30-minute conda test drive and download a conda cheat sheet.

- Over 250 packages are automatically installed with Anaconda.
- Over 7,500 additional open-source packages (including R) can be individually installed from the Anaconda repository with the `conda install` command.
- Thousands of other packages are available from Anaconda Cloud.
- You can download other packages using the `pip install` command that is installed with Anaconda. Pip packages provide many of the features of conda packages and in some cases they can work together. However, the preference should be to install the conda package if it is available.

- You can also make your own custom packages using the conda build command, and you can share them with others by uploading them to Anaconda Cloud, PyPI, or other repositories.

1.4.3 Installation

- **License:** Free use and redistribution under the terms of the End User License Agreement.
- **Operating system:** Windows 7 or newer, 64-bit macOS 10.13+, or Linux, including Ubuntu, RedHat, CentOS 6+, and others.
- If your operating system is older than what is currently supported, you can find older versions of the Anaconda installers in our archive that might work for you. See Using Anaconda on older operating systems for version recommendations.
- **System architecture:** Windows- 64-bit x86, 32-bit x86; MacOS- 64-bit x86; Linux- 64-bit x86, 64-bit Power8/Power9.
- Minimum 5 GB disk space to download and install.

On Windows, macOS, and Linux, it is best to install Anaconda for the local user, which does not require administrator permissions and is the most robust type of installation. However, if you need to, you can install Anaconda system wide, which does require administrator permissions. You can use silent mode to automatically accept default settings and have no screen prompts appear during installation.

We recommend upgrading your operating system. Most OS that are no longer supported in the latest Anaconda are no longer getting security updates. To use Anaconda on older operating systems, download from our archive. You will not be able to use conda to update or install packages beyond the Anaconda version noted in the table below, unless you limit it to versions available at the time that particular version of Anaconda was released. You can see what was available by checking the package table archives.

Operating System	How to install Anaconda
macOS 10.10-10.12	Use the command line or graphical installers for Anaconda versions 2019.10 and earlier. Download from our archive.
macOS 10.9	Use the command line or graphical installers for Anaconda versions 5.1 and earlier.
macOS 10.5 and 10.6	Use the command line installers for Anaconda versions 1.8 and earlier
Windows XP	Use Anaconda versions 2.2 and earlier.
Centos5 (or equivalent)	Use Anaconda versions 4.3 and earlier.

Table 1.4.3.1 Using Anaconda on older operating systems

- Obtain a local copy of the appropriate Anaconda installer for the non-networked machine. You can copy the Anaconda installer to the target machine using many different methods including a portable hard drive, USB drive, or CD.
- After copying the installer to the non-networked machine, follow the detailed installation instruction.

1.4.4 User guide

If you're new to Anaconda, follow the directions at Getting started with Anaconda to write your first Python project using Anaconda. Use the navigation tabs at the bottom of the page to go through the user guide in order. Read the Navigator user guide to learn how to use Anaconda's graphical user interface. Get an overview of the interface and take your first steps

using Navigator at Getting started with Navigator. Conda is an open source package management system and environment management system included in Anaconda and Miniconda. Learn how to get started with conda.

Explore what you can do with Anaconda. The tasks page shows you how to install conda packages, switch between environments, use IDEs, and more. See How to contribute to Anaconda by helping resolve issues, improve documentation, add to feedstocks, and contribute code.

1.4.5 Getting Started With Anaconda

Anaconda Distribution contains conda and Anaconda Navigator, as well as Python and hundreds of scientific packages. When you installed Anaconda, you installed all these too. Conda works on your command line interface such as Anaconda Prompt on Windows and terminal on macOS and Linux.

Navigator is a desktop graphical user interface that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands.

You can try both conda and Navigator to see which is right for you to manage your packages and environments. You can even switch between them, and the work you do with one can be viewed in the other.

1.4.6 Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, macOS, and Linux.

- To get Navigator, get the Navigator Cheat Sheet and install Anaconda.

- The Getting started with the Navigator section shows how to start Navigator from the shortcuts or from a terminal window.
- The command-line program conda is both a package manager and an environment manager. This helps data scientists ensure that each version of each package has all the dependencies it requires and works correctly. Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages, and update them – all inside Navigator.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- Spyder
- VSCode
- Glueviz
- Orange 3
- RStudio

The simplest way is with Spyder. From the Navigator Home tab, click Spyder, and write and execute your code. You can also use Jupyter Notebooks the same way. Jupyter Notebooks are an increasingly popular system that combines your code, descriptive text, output, images, and interactive interfaces into a single notebook file that is edited, viewed, and used in a web browser.

1.5 Introduction to Spyder

1.5.1 Introduction

Spyder is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib,

pandas, IPython, SymPy and Cython, as well as other open source software. It is released under the MIT license.

Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community. Spyder is extensible with first- and third-party plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and Rope. It is available cross-platform through Anaconda, on Windows, on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.

1.5.2 Overview

Spyder is a powerful scientific environment written in Python, for Python, and designed by and for scientists, engineers and data analysts. It offers a unique combination of the advanced editing, analysis, debugging, and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection, and beautiful visualization capabilities of a scientific package.

1.5.3 Core components

Work efficiently in a multi-language editor with a function/class browser, real-time code analysis tools (pyflakes, pylint, and pycodestyle), automatic code completion (jedi and rope), horizontal/vertical splitting, and go-to-definition.

Harness the power of as many IPython consoles as you like with full workspace and debugging support, all within the flexibility of a full GUI interface run your code by line, cell, file.

Render documentation in real-time with Sphinx for any class or function, whether external or user-created, from either the Editor or a Console.

Inspect any variables, functions or objects created during your session. Editing and interaction is supported with many common types, including numeric/strings/booleans, Python lists/tuples/dictionaries, dates/timedeltas, Numpy arrays, Pandas index/series/dataframes, PIL/Pillow images, and more.

Examine your code with the static analyzer, trace its execution with the interactive debugger, and unleash its performance with the profiler. Keep things organized with project support and a builtin file explorer, and use find in files to search across entire projects with full regex support.

1.5.4 Installation

The easiest way to install Spyder on any of our supported platforms is to download it as part of the Anaconda distribution, and use the conda package and environment manager to keep it and your other packages installed and up to date.

If in doubt, you should always install Spyder via this method to avoid unexpected issues we are unable to help you with; it generally has the least likelihood of potential pitfalls for non-experts, and we may be able to provide limited assistance if you do run into trouble.

Other install options exist, including:

The WinPython distribution for Windows

- The MacPorts project for macOS
- Your distribution's package manager (i.e. apt-get, yum, etc) on Linux
- The pip package manager, included with most Python installations

However, we lack the resources to provide individual support for users who install via these methods, and they may be out of date or contain bugs outside our control, so we recommend the Anaconda version instead if you run into issues.

1.5.5 Features

- An editor with syntax highlighting, introspection, code completion
- Support for multiple IPython consoles
- The ability to explore and edit variables from a GUI
- A Help pane able to retrieve and render rich text documentation on functions, classes and methods automatically or on-demand
- A debugger linked to IPdb, for step-by-step execution

2. LITERATURE SURVEY

There were many approaches used for dealing with disparity in images subject to illumination changes and these approaches were implemented in object recognition systems and also by systems that were specific to faces. A method for dealing with such variations was using gray-level information to extract a face or an object from shading approach. The main reason why gray scale representations are used for extracting descriptors instead of operating on colour images directly is that gray scale simplifies the algorithm and reduces computational requirements. Here in our case, colour is of limited benefit and introducing unnecessary information could increase the amount of training data required to achieve good performance. Being an ill-posed problem, these proposed solutions assumed either the object shape and reflectance properties or the illumination conditions.

2.1 Borisagar, V. H., & Zaveri, M. A. (2014). Disparity Map Generation from Illumination Variant Stereo Images Using Efficient Hierarchical Dynamic Programming. The Scientific World Journal, 2014, Article ID 513417.

There were many approaches used for dealing with disparity in images subject to illumination changes and these approaches were implemented in object recognition systems and also by systems that were specific to faces. A method for dealing with such variations was using gray-level information to extract a face or an object from a shading approach. The main reason why gray-scale representations are used for extracting descriptors instead of operating on color images directly is that gray scale simplifies the algorithm and reduces computational requirements. Here in our case, color is of limited benefit and introducing unnecessary information could increase the amount of training data required to achieve good performance. Being an ill-posed problem, these proposed solutions assumed either the object shape and reflectance properties or the illumination conditions.

These assumptions made are too strict for general object recognition and therefore it didn't prove to be sufficient for face recognition. The second approach is the edge map of the image which is a useful object representation feature that is insensitive to illumination changes to a certain event. Edge images could be used for recognition and to achieve similar accuracy as gray-level pictures. The edge map information approach possesses the advantage of feature-based approaches, such as invariance to illumination and low memory requirement. It integrates the structural information is

grouping pixels of face edge map to line segments. After thinning the edge map, a polygonal line fitting process is applied to generate the edge map of a face. There is one another approach through which the image disparities due to illumination differences are handled; it is by using a model of several images of the same face which is taken under various illumination conditions. Herein, the images captured can be used as independent models or as a combined model-based recognition system.

2.2 Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94

Gray-scale representations simplify algorithms and reduce computational requirements. When dealing with object recognition, color information may not always be beneficial. In fact, introducing unnecessary color data could increase the amount of training data required for good performance. However, relying solely on gray scale assumes that the object shape and reflectance properties are invariant across illumination conditions, which is often not the case. Relying solely on gray scale simplifies the algorithm, as you rightly pointed out. However, it assumes certain invariances

Shape Invariance performs The object's shape remains consistent across different lighting conditions.

Reflectance Invariance is The material properties (reflectance) of the object do not change significantly. Unfortunately, these assumptions are often violated in real-world scenarios.

Handling Illumination Variations is To address illumination changes, researchers explore various techniques

Multiple Illumination Models Capturing images of the same face under different lighting conditions (as we discussed earlier).

Invariant Descriptors Extracting features that are less sensitive to illumination changes (e.g., SIFT, SURF).

Color Constancy Algorithms Correcting color variations caused by different light sources.

2.3 Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 679–698. DOI: 10.1109/TPAMI.1986.4767851

Edge maps provide a useful representation feature that is less sensitive to illumination changes. By extracting structural information from the edges of an image, we can achieve similar accuracy as with gray-level pictures. The advantage lies in invariance to illumination variations and low memory requirements. Grouping pixels from the edge map into line segments and applying

polygonal line fitting allows us to generate an edge map representation of a face. Edge maps serve as a powerful feature for object recognition. They are invariant to global illumination changes, making them robust in varying lighting conditions. By focusing on local structures, edge-based descriptors enhance recognition accuracy. Image Segmentation is an Edge maps aid in segmenting an image into meaningful regions. Boundaries defined by edges help separate objects from the background. Shape Analysis can be a Edge-based shape descriptors are widely used in shape matching and retrieval tasks. Curvature information along edges provides discriminative features. Some methods create models from multiple images of the same face captured under various illumination conditions. These models serve as independent representations or can be combined to build a robust recognition system.

2.4 Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), 711–720. DOI: 10.1109/34.598228

Another approach involves capturing several images of the same face under different illumination conditions. These images can serve as independent models or be combined to create a more robust recognition system. By considering multiple lighting scenarios, we can handle image disparities effectively. Collect several images of the same individual from different lighting conditions. These images should cover a wide range of illumination scenarios, including variations in intensity, direction, and color temperature. Independent Models or Combined Model contains These images can serve as independent models or be combined to create a more robust recognition system. Let's explore both options.

Independent Models are used to Treat each image as an independent model. During recognition, compare the input face with each model separately. The final decision can be based on voting or weighted scores from individual models. Combined Model are contains Fuse information from all images into a single model. Techniques like Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA) can be used to create a unified representation. This combined model captures the variations due to illumination while maintaining discriminative features. Texture-based methods focus on capturing patterns and variations in pixel intensities. These approaches analyze local texture properties, such as co-occurrence matrices, Gabor filters, or local binary patterns. By

considering texture information, these methods can be more robust to illumination changes. Histogram equalization and normalization techniques aim to adjust the pixel intensity distribution in an image. These methods enhance contrast and reduce the impact of varying illumination conditions. Histogram equalization redistributes pixel values to achieve a uniform histogram, while normalization scales pixel values to a common range.

Shadows often introduce significant disparities in images. Shadow detection and removal algorithms can help mitigate this issue. Techniques include shadow detection based on color information, shadow segmentation, and shadow compensation during feature extraction. Convolutional neural networks (CNNs) and other deep learning architectures have shown promising results in handling illumination variations. These models learn hierarchical features directly from raw pixel data, allowing them to adapt to different lighting conditions. To improve model robustness, data augmentation techniques can artificially create variations in the training dataset. By introducing augmented images with different illumination levels.

3. OVERALL DESCRIPTION

3.1 Feasibility Report

3.1.1 Technical Feasibility

Evaluating technical feasibility is the trickiest part of a feasibility study. This is because, at this point in time, not too many detailed designs of the system, making it difficult to access issues like performance, costs on (on account of the kind of the technology to be developed)etc. A number of issues have to be considered while doing a technical analysis.

1. Understand the different technologies involved in the proposed system:

Before commencing the project, we have to be very clear about what are the technologies that are to be required for the development of the new system.

2. Find out whether the organisation currently possesses the required technologies:

- It is the required technology available with the organization?
- If so is the capacity sufficient?

3.1.2 Operational Feasibility

Proposed projects are beneficial only if they can be turned into an information system that will meet the organizations operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to implement?

Here, the questions that will help test the operational feasibility of a project:

- Is there sufficient support for the project from management from users? If the current system is well liked and used to the extent that a person will not be able to see the reasons for changes, there may be resistance.
- Are the current business methods acceptable to the users? If they are not, users may welcome a change that will bring about a more operational and useful system.
- Have the user been involved in the planning the development of the project?

- Early involvement reduces the chances of the resistance to the system and in
- General and increases the likelihood of successful project

3.1.3 Economic Feasibility

Economic feasibility attempts to weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place this feasibility study will give the top management and economic justification for the new system.

A simple economic analysis which gives the actual comparison of costs and benefits are must more meaningful in this case. In addition, this proves to be a useful point of reference to compare actual costs as the project progresses this could be various types of intangible benefits on account of automation this could include increased customer satisfaction improvement in product quality better decision making timelines of information, expediting activities, improved accuracy of operations, better documentation and record keeping faster retrieval of information, better employee morale.

4. SYSTEM ANALYSIS

4.1 Existing System

In the present system all work is done on paper. The whole session attendance is stored in the register and at the end of the semester the reports are generated and it takes more time for calculation.

Initially, attendance will be manually stored in registers by calling roll numbers. Later, Biometric attendance comes into play. Biometrics has decreased the time taken for attendance in traditional way but still it has some problems like proxy and some network connection issues. Following are the disadvantages of existing system:

- We require more calculations to generate the report.
- It is a time taking process.
- Work needs to be done manually.
- Lots of paperwork needed to generate reports.
- By using Bio-Metric attendance time has been reduced but it can be easily manipulated.
- Chance for proxy attendance.
- To overcome the drawbacks from manual and Biometric, we proposed a system.

4.2 Proposed System

In this Attendance System, images of students are collected to recognize the faces of the students. The system is initially trained with the student's faces which is collectively known as student database. The system uses user friendly User interface to maximize the user experience while both training and testing which are collecting student images and taking attendance with the system. This project can be used for many other applications where face recognition can be used for authentication. The system will automatically generate reports at the end of semester. Following are the advantages of proposed system. By using our proposed system, attendance will be taken by recognizing the human face.

- It is an easy process and a quick process.
- Fully automated process for attendance and generating reports.
- Time can be saved during generating reports, calculations and taking attendance.

4.3 Software Requirements

- IDE : Spyder
- Back-End Language : Open CV using Python
- Front-End Language : Tkinter
- Database : Excel

4.4 Hardware Requirements

- RAM : 4 GB
- Processor : Intel i5
- Hard Disk : 15 GB

5. SYSTEM DESIGN

5.1 System Architecture

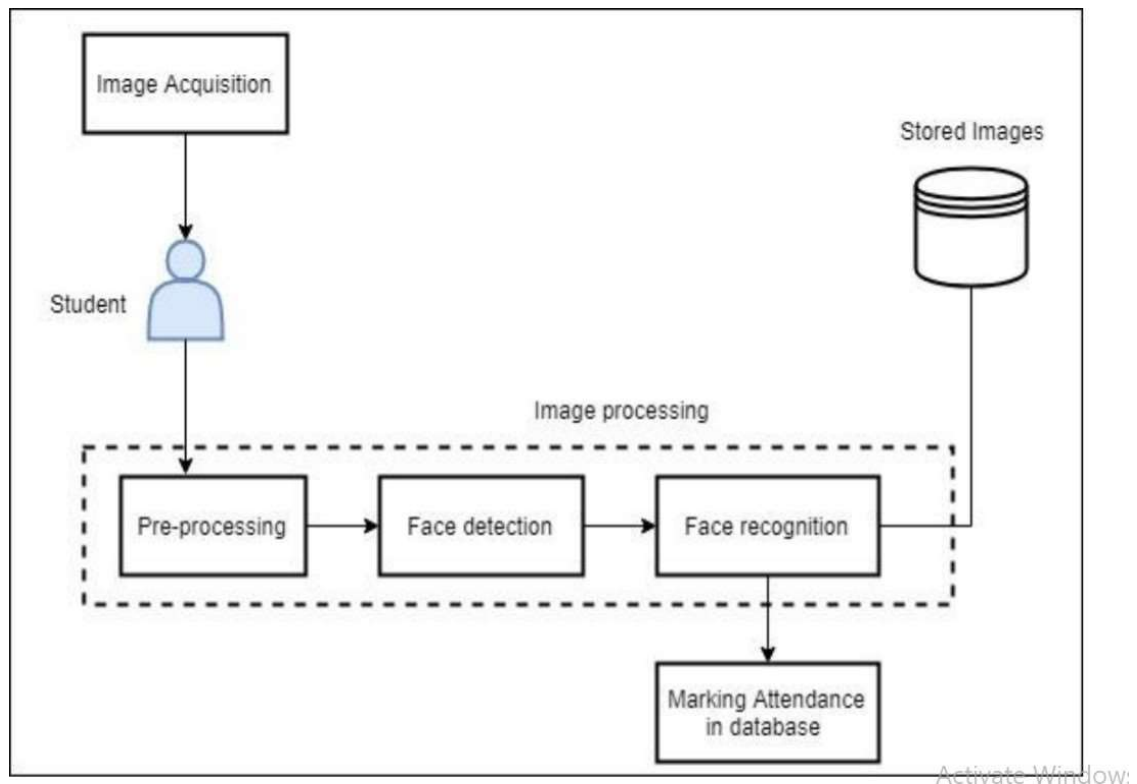


Fig 5.1 System Architecture

5.2 UML Diagrams

5.2.1 Usecase Diagram

Use Case Diagram captures the system's functionality and requirements by using actors and use cases. Use Cases model the services, tasks, function that a system needs to perform. Use cases represent high-level functionalities and how a user will handle the system.

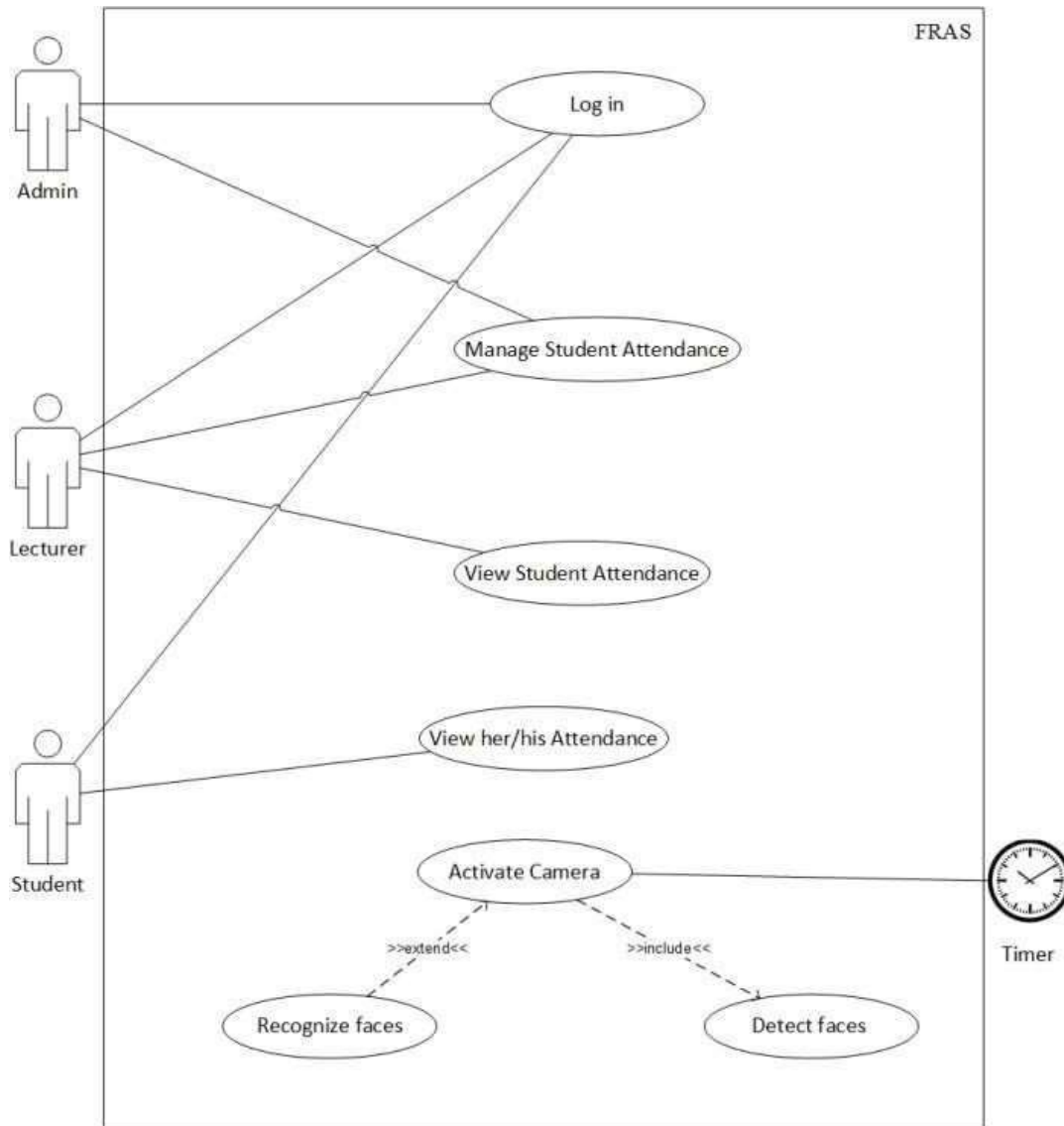


Fig 5.2.1 usecase diagram

5.2.2 Sequence Diagram

UML Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

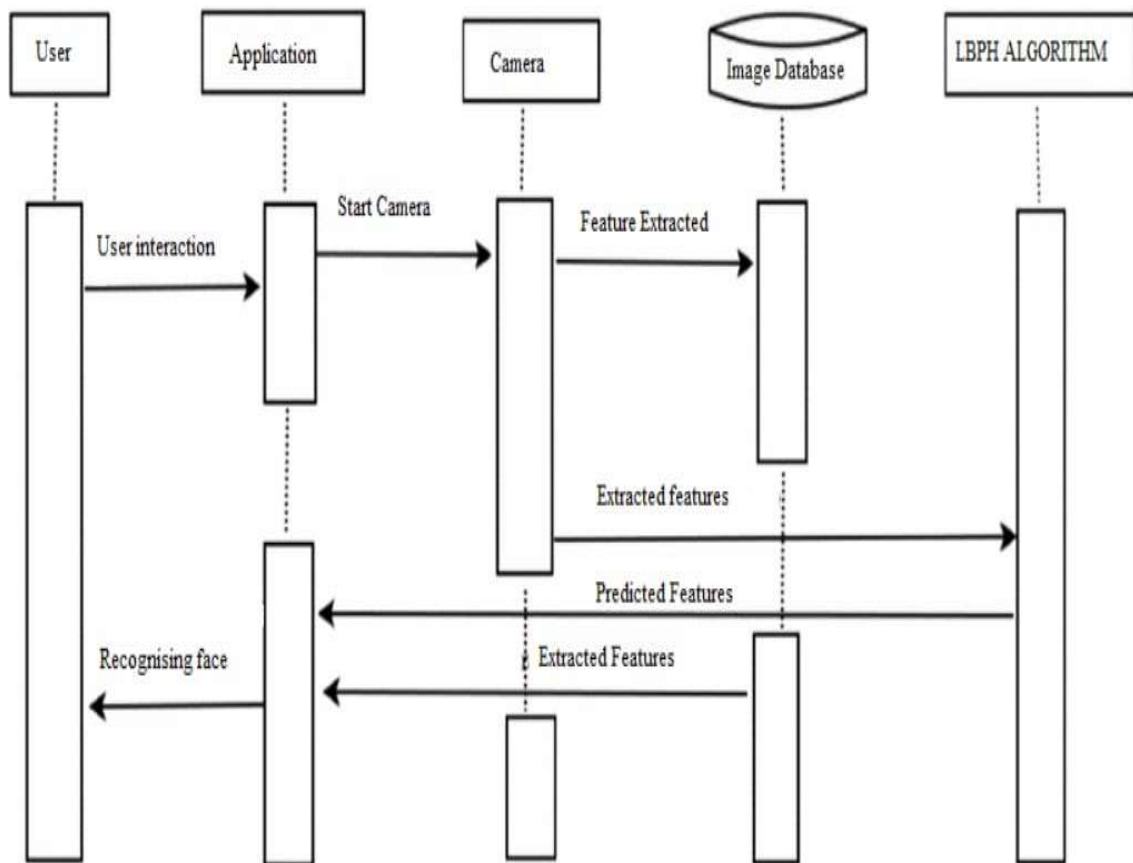


Fig 5.2.2 sequence diagram

5.2.3 Activity Diagram

Activity diagram is defined as a UML diagram that focuses on the execution and flow of the behavior of a system instead of implementation. It is also called object-oriented flowchart. Activity diagrams consist of activities that are made up of actions which apply to behavioral modeling technology.

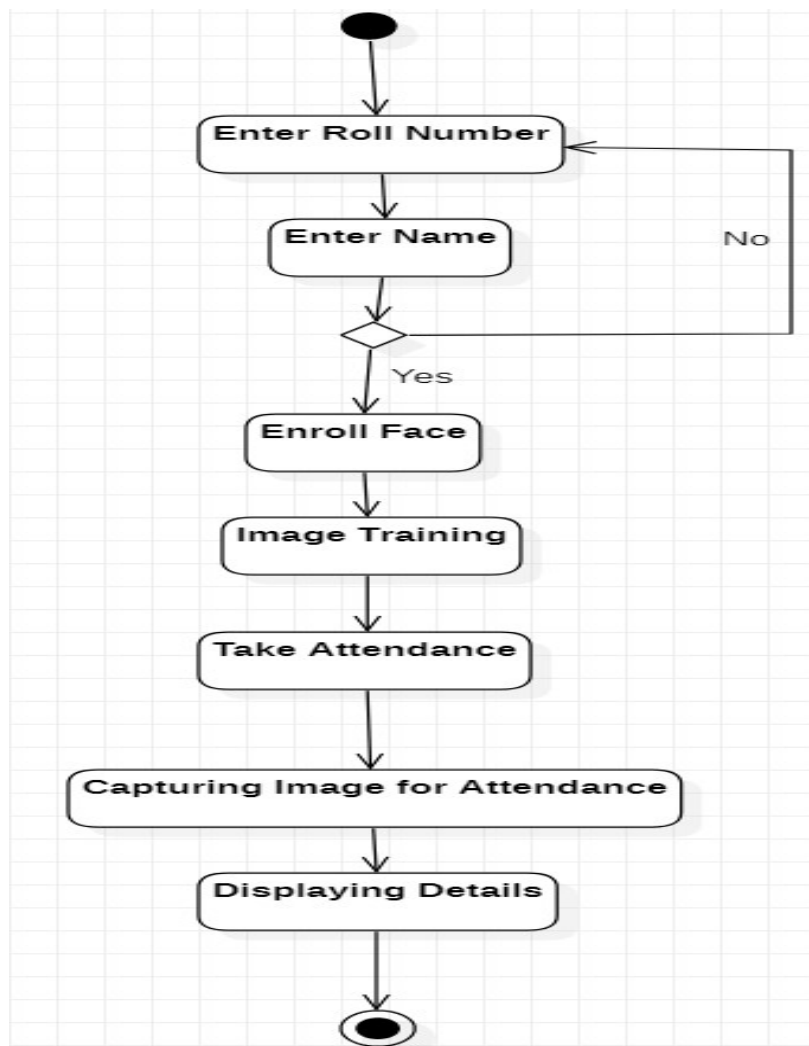


Fig 5.2.3 Activity Diagram

6. IMPLEMENTATION

6.1 Introduction To Python

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, <https://www.python.org/>, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

For a description of standard objects and modules, see The Python Standard Library. The Python Language Reference gives a more formal definition of the language. To write extensions in C or C++, read Extending and Embedding the Python Interpreter and Python/C API Reference Manual. There are also several books covering Python in depth.

6.1.1 History of the software

Python was created in the early 1990s by Guido van Rossum at Stichting Mathematisch Centrum (CWI, see <https://www.cwi.nl/>) in the Netherlands as a successor of a language called ABC. Guido remains Python's principal author, although it includes many contributions from others.

In 1995, Guido continued his work on Python at the Corporation for National Research Initiatives (CNRI, see <https://www.cnri.reston.va.us/>) in Reston, Virginia where he released several versions of the software.

In May 2000, Guido and the Python core development team moved to BeOpen.com to form the BeOpen PythonLabs team. In October of the same year, the PythonLabs team moved to Digital Creations (now Zope Corporation; see <https://www.zope.org/>). In 2001, the Python Software Foundation (PSF, see <https://www.python.org/psf/>) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

All Python releases are Open Source (see <https://opensource.org/> for the Open Source Definition). Historically, most, but not all, Python releases have also been GPL-compatible; the table below summarizes the various releases.

6.1.2 How To Install Python 3.8 On Windows

Python has emerged as one of the major programming languages used to develop different types of applications including Web Applications, Desktop Applications, Numeric and Scientific Applications, etc. It soon entered into mainstream programming and is majorly popular in data science engineers. It's a high-level programming language used for general-purpose programming.

This tutorial provides all the steps required to install the most recent version of Python 3 i.e. Python 3.8 on Windows 10. The steps should be similar for other versions of Python and Windows.

- **Step 1 - Download**

Open the Downloads Page for Windows and choose the most recent version. You can directly start downloading the most recent version by clicking the yellow button or choose the other download options from the downloads. I have clicked on the Download Link of Python 3.8.1 while writing this tutorial.

- **Step 2 - Install**

After completing the download, double click the installer to start the installation. The welcome screen shows the default installation location and provides options to Install the launcher for all users and to add Python 3.8 to the system path as shown in Fig 6.1.2.1.



Fig 6.1.2.1 Python Installation

After checking the checkboxes for Launcher and System Path, I have clicked the Customize installation Link to choose the installation components and to change the installation path. It shows the components including Documentation, pip, tcl/tk and IDLE, Python test suite, py launcher, for all users as shown in Fig 6.1.2.2.

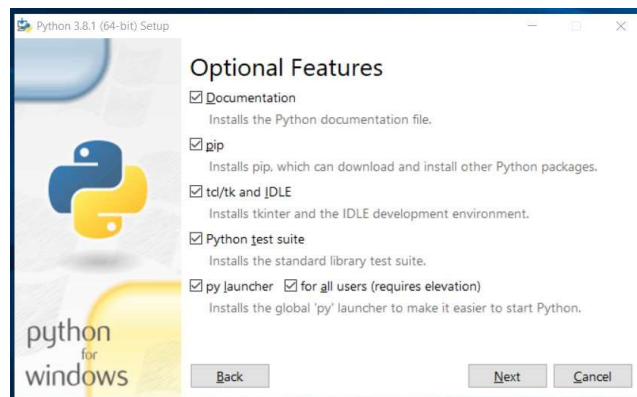


Fig 6.1.2.2 Python Installation

Click the Next Button to choose Advanced Options as shown in Fig 6.1.2.3.

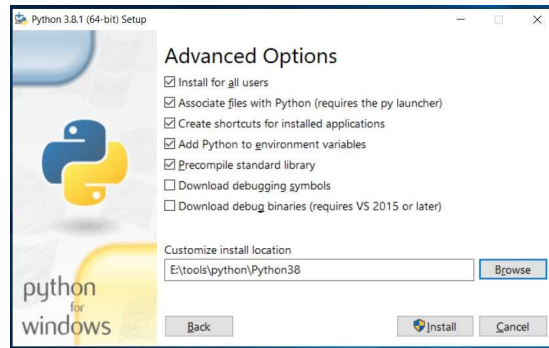


Fig 6.1.2.3 Python Advanced Options

I have selected the option to install it for all the users and also provided the custom installation location as shown in Fig 5. Now click the Install Button to start the installation. It will also ask for system permission to continue with the installation. Allow the installer to complete the installation. It will show the installation progress as shown in Fig 6.1.2.4.

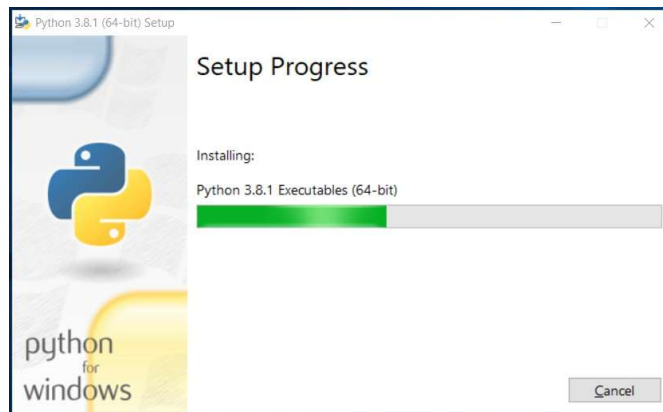


Fig 6.1.2.4 Setup Progress

It will show the success screen after completing the installation as shown in Fig 6.1.2.5.



Fig 6.1.2.5 Setup was Successful

Now click the Close Button to close the installer. This completes the installation step to install Python.

- **Step 3 - Verify Installation**

In this step, we will verify the installation of Python. Search for Python on system tray as shown in Fig 8. It will show all the available options.

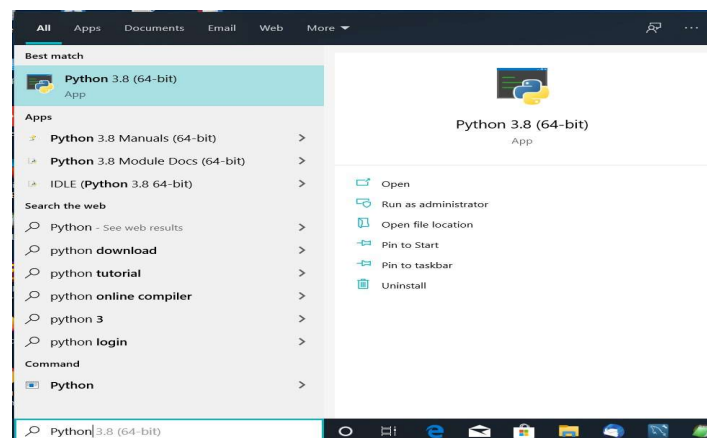


Fig 6.1.2.6 Opening Python

Now execute the Python command-line option as highlighted in Fig 8. It will start the Python command line window as shown in Fig 6.1.2.7.

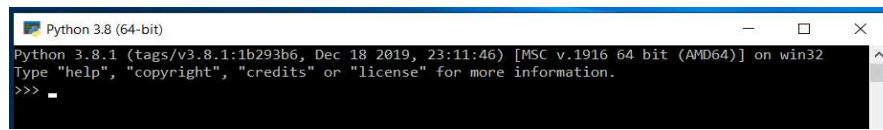


Fig 6.1.2.7 Python Prompt

It shows that Python 3.8.1 is installed as part of this tutorial. We can also verify the version on Windows Command Prompt as shown in Fig 6.1.2.8.

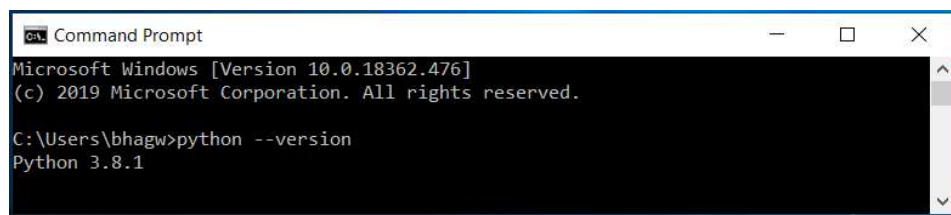


Fig 6.1.2.8 Checking Whether Python is Installed or Not

6.2 Introduction To Tkinter

Python provides various options for developing graphical user interfaces (GUIs). The most important features are listed below.

Python has a lot of GUI frameworks, but Tkinter is the only framework that's built into the Python standard library. Tkinter has several strengths. It's cross-platform, so the same code works on Windows, macOS, and Linux. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they're run.

Although Tkinter is considered the de-facto Python GUI framework, it's not without criticism. One notable criticism is that GUIs built with Tkinter look outdated. If you want a shiny, modern interface, then Tkinter may not be what you're looking for.

However, Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python,

especially for applications where a modern sheen is unnecessary, and the top priority is to build something that's functional and cross-platform quickly.

6.2.1 Tkinter Programming

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps

- Import the Tkinter module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

6.2.2 Tkinter Widgets

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

There are currently 15 types of widgets in Tkinter. We present these widgets as well as a brief description in the following table –Sr.No.Operator & Description

1. Button

The Button widget is used to display the buttons in your application.

2. Entry

The Entry widget is used to display a single-line text field for accepting values from a user.

3. Label

The Label widget is used to provide a single-line caption for other widgets. It can also contain images.

4. Message

The Message widget is used to display multiline text fields for accepting values from a user.

5. Text

The Text widget is used to display text in multiple lines.

6. LabelFrame

A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts.

7. tkMessageBox

This module is used to display message boxes in your applications

6.2.3 Standard attributes

Let us look at how some of their common attributes, such as sizes, colors and fonts are specified.

- Dimensions
- Colors
- Fonts
- Anchors
- Bitmaps
- Cursors

6.2.4 Tkinter Colors

Tkinter represents colors with strings. There are two general ways to specify colors in Tkinter

- You can use a string specifying the proportion of red, green and blue in hexadecimal digits. For example, "#fff" is white, "#000000" is black, "#000fff000" is pure green, and "#00ffff" is pure cyan (green plus blue).

- You can also use any locally defined standard color name. The colors "white", "black", "red", "green", "blue", "cyan", "yellow", and "magenta" will always be available.

Color Options

The common color options are

- **background** – Background color for the widget. This can also be represented as *bg*.
- **disabledforeground** – Foreground color for the widget when the widget is disabled.
- **foreground** – Foreground color for the widget. This can also be represented as *fg*.
- **highlightcolor** – Foreground color of the highlight region when the widget has focus.
- **selectbackground** – Background color for the selected items of the widget.

6.3 Introduction To OpenCV

OpenCV was started at Intel in 1999 by Gary Bradsky and the first release came out in 2000. Vadim Pisarevsky joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle who won 2005 DARPA Grand Challenge. Later its active development continued under the support of Willow Garage, with Gary Bradsky and Vadim Pisarevsky leading the project. Right now, OpenCV supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-by-day.

6.3.1 OpenCV-Python

Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability.

So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

6.3.2 Read an image

Use the function `cv2.imread()` to read an image. The image should be in the working directory or a full path of image should be given.

- **cv2.IMREAD_COLOR** : Loads a color image. Any transparency of image will be neglected. It is the default flag.
- **cv2.IMREAD_GRAYSCALE** : Loads image in grayscale mode
- **cv2.IMREAD_UNCHANGED** : Loads image as such including alpha channel

6.3.3 Display an image

Use the function `cv2.imshow()` to display an image in a window. The window automatically fits the image size.

First argument is a window name which is a string. second argument is our image. You can create as many windows as you wish, but with different window names.

```
cv2.imshow('image',img)
```

```
cv2.waitKey(0)
```

```
cv2.destroyAllWindows()
```

A screenshot of the window will look like this (in Fedora-Gnome machine):



Fig 6.2.5 Display Image In Window

`cv2.waitKey()` is a keyboard binding function. Its argument is the time in milliseconds. The function waits for specified milliseconds for any keyboard event. If you press any key in that time, the program continues. If 0 is passed, it waits indefinitely for a keystroke. It can also be set to detect specific key strokes like, if key 'a' is pressed etc which we will discuss below. `cv2.destroyAllWindows()` simply destroys all the windows we created. If you want to destroy any specific window, use the function `cv2.destroyWindow()` where you pass the exact window name as the argument.

6.3.4 Write an image

Use the function `cv2.imwrite()` to save an image.

First argument is the file name, second argument is the image you want to save.

Below program loads an image in grayscale, displays it, save the image if you press 's' and exit, or simply exit without saving if you press ESC key.

```
import numpy as np

import cv2

img = cv2.imread('messi5.jpg',0)

cv2.imshow('image',img)

k = cv2.waitKey(0)

if k == 27:    # wait for ESC key to exit

    cv2.destroyAllWindows()

elif k == ord('s'): # wait for 's' key to save and exit

    cv2.imwrite('messigray.png',img)

    cv2.destroyAllWindows()
```

6.4 Introduction To NumPy

6.4.1 Python Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.

6.4.2 Arrays in Numpy

Array in Numpy is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In Numpy, the number of dimensions of the array is called rank of the array. A tuple of integers giving the size of the array along each dimension is known as shape of the array. An array class in Numpy is called ndarray. Elements in Numpy arrays are accessed by using square brackets and can be initialized by using nested Python Lists.

6.4.3 Creating a Numpy Array

Arrays in Numpy can be created in multiple ways, with various numbers of Ranks, defining the size of the Array. Arrays can also be created with the use of various data types such as lists, tuples, etc. The type of the resulting array is deduced from the type of the elements in the sequences.

6.4.4 Accessing the array Index

In a numpy array, indexing or accessing the array index can be done in multiple ways. To print a range of an array, slicing is done. Slicing of an array is defining a range in a new array which is used to print a range of elements from the original array. Since, sliced array holds a range of elements of the original array, modifying content with the help of sliced array modifies the original array content.

6.4.5 Basic Array Operations

In numpy, arrays allow a wide range of operations which can be performed on a particular array or a combination of Arrays. These operations include some basic Mathematical operations as well as Unary and Binary operations.

6.4.6 More on Numpy Arrays

- Basic Array Operations in Numpy
- Advanced Array Operations in Numpy
- Basic Slicing and Advanced Indexing in NumPy Python

6.4.7 Data Types in Numpy

Every Numpy array is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. Every ndarray has an associated data type (dtype) object. This data type object (dtype) provides information about the layout of the array. The values of an ndarray are stored in a buffer which can be thought of as a contiguous block of memory bytes which can be interpreted by the dtype object. Numpy provides a large set of numeric data types that can be used to construct arrays. At the time of Array creation, Numpy tries to guess a data type, but functions that construct arrays usually also include an optional argument to explicitly specify the datatype.

6.4.8 Constructing a Data Type Object

In Numpy, datatypes of Arrays need not be defined unless a specific data type is required. Numpy tries to guess the datatype for Arrays which are not predefined in the constructor function.

6.5 Introduction To Pandas

6.5.1 What is Pandas

Pandas is an open source library that allows to you perform data manipulation in Python. Pandas library is built on top of Numpy, meaning Pandas needs Numpy to operate. Pandas provide an easy way to create, manipulate and wrangle the data. Pandas is also an elegant solution for time series data.

6.5.2 Why use Pandas

Data scientists use Pandas for its following advantages:

- Easily handles missing data

- It uses Series for one-dimensional data structure and DataFrame for multi-dimensional data structure
- It provides an efficient way to slice the data
- It provides a flexible way to merge, concatenate or reshape the data
- It includes a powerful time series tool to work with

6.5.3 What is a data frame

A data frame is a two-dimensional array, with labeled axes (rows and columns). A data frame is a standard way to store data.

Data frame is well-known by statistician and other data practitioners. A data frame is a tabular data, with rows to store the information and columns to name the information. For instance, the price can be the name of a column and 2,3,4 the price values.

Below a picture of a Pandas data frame:

	Item	Price
0	A	2
1	B	3

6.6 Introduction To DateTime Module

6.6.1 How to get current date and time in Python

In this article, you will learn to get today's date and current date and time in Python. We will also format the date and time in different formats using strftime() method.

There are a number of ways you can take to get the current date. We will use the date class of the datetime module to accomplish this task.

Example 1: Python get today's date

```
from datetime import date  
  
today = date.today()
```

```
print("Today's date:", today)
```

Run Code

Here, we imported the date class from the datetime module. Then, we used the date.today() method to get the current local date.

Example 2: Get the current date and time

```
from datetime import datetime

# datetime object containing current date and time

now = datetime.now()

print("now =", now)

# dd/mm/YY H:M:S

dt_string = now.strftime("%d/%m/%Y %H:%M:%S")

print("date and time =", dt_string)
```

6.7 Introduction To CV2 Module

6.7.1 what is CV2 in Python

Python OpenCV | cv2. OpenCV-Python is a library of Python bindings designed to solve computer vision problems. cv2.imread() method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.

6.7.2 How do import cv2 in Anaconda

To use OpenCV fully with Anaconda (and Spyder IDE), we need to: Download the OpenCV package from the official OpenCV site. Copy and paste the cv2.pyd to the Anaconda site-packages directory.

6.7.3 cv2.imread() method

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. cv2.imread() method loads an image from the specified file. If the image cannot be read.

Syntax: cv2.imread(path, flag)

Parameters:

path: A string representing the path of the image to be read.

flag: It specifies the way in which image should be read.

Return Value: This method returns an image that is loaded from the specified file. All three types of flags are described below:

1. **cv2.IMREAD_COLOR:** It specifies to load a color image. Any transparency of image will be neglected. It is the default flag. Alternatively, we can pass integer value 1 for this flag.
2. **cv2.IMREAD_GRAYSCALE:** It specifies to load an image in grayscale mode. Alternatively, we can pass integer value 0 for this flag.
3. **cv2.IMREAD_UNCHANGED:** It specifies to load an image as such including alpha channel. Alternatively, we can pass integer value -1 for this flag.

6.7.4 cv2.imwrite() method

cv2.imwrite() method is used to save an image to any storage device. This will save the image according to the specified format in the current working directory.

Syntax: cv2.imwrite(filename, image)

Parameters:

filename: A string representing the file name. The filename must include image format like .jpg, .png, etc.

image: It is the image that is to be saved.

Return Value: It returns true if the image is saved successfully.

6.7.5 cv2.imshow() method

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. cv2.imshow() method is used to display an image in a window. The window automatically fits the image size.

Syntax: cv2.imshow(window_name, image)

Parameters:

window_name: A string representing the name of the window in which image to be displayed.

image: It is the image that is to be displayed.

Return Value: It doesn't return anything.

6.8 Introduction To OS Module

6.8.1 Python - OS Module

It is possible to automatically perform many operating system tasks. The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc.

6.8.2 List Files and Sub-directories

The **listdir()** function returns the list of all files and directories in the specified directory.

```
>>>os.listdir("c:\python37")

['DLLs', 'Doc', 'fantasy-1.py', 'fantasy.db', 'fantasy.py', 'frame.py', 'gridexample.py', 'include', 'pythonw.exe', 'sclst.py', 'Scripts', 'tcl', 'test.py', 'Tools', 'tooltip.py', 'vcruntime140.dll', 'virat.jpg', 'virat.py']
```

6.8.3 os.path.join() method

os.path.join() method in Python joins one or more path components intelligently. This method concatenates various path components with exactly one directory separator ('/') following each non-empty part except the last path component. If the last path component to be joined is empty then a directory separator ('/') is put at the end.

If a path component represents an absolute path, then all previous components joined are discarded and joining continues from the absolute path component.

Syntax: `os.path.join(path,*paths)`

6.9 Introduction To PIL

6.9.1 How to use Pillow, a fork of PIL

In last post I was writing about PIL, also known as Python Imaging Library, this library can be used to manipulate images quite easily. PIL hasn't seen any development since 2009. Therefore, the kind users of this site suggested to take a look at Pillow. This article will tell you how to use Pillow.

6.9.2 What is Pillow

Pillow is a fork of PIL (Python Image Library), started and maintained by Alex Clark and Contributors. It was based on the PIL code, and then evolved to a better, modern and more friendly version of PIL. It adds support for opening, manipulating, and saving many different image file formats. A lot of things work the same way as the original PIL.

6.9.3 Verify that Pillow is installed

To verify that Pillow is installed, open up a Terminal and type in the following line:

```
$ python
Python 2.7.5 (default, Aug 25 2013, 00:04:04)
[GCC 4.2.1 Compatible Apple LLVM 5.0 (clang-500.0.68)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from PIL import Image
```

If the system comes back with a ">>>", the Pillow modules are properly installed.

6.9.4 How to use Pillow to manipulate an image

Since we are going to work with images, let us first download one. If you already have a picture to use, go ahead and skip this step. In our example we will use a standard test image called "Lenna" or "Lena".

6.9.5 Using Pillow

Let us look at the possible uses for this library. The basic functions are found in the Image module. You can create instances of this class in several ways either by loading images from files, processing other images, or creating images from scratch. Import the Pillow modules you want to use.

```
from PIL import Image
```

You can then access functions as usual, e.g.

```
myimage = Image.open(filename)
myimage.load()
```

6.9.6 Load an Image

To load an image from your computer, you can use the "open" method to identify the file, and then load the identified file using `myfile.load()`. Once the image is loaded, you can do a number of things with it. I often use the try/except block when dealing with files. To load our image using try/except:

```
from PIL import Image, ImageFilter

try:
    original = Image.open("Lenna.png")
except:
    print "Unable to load image"
```

When we read files from disk using the `open()` function, we don't have to know the format of the file to. The library automatically determines the format based on the contents of the file. Now when you have an Image object, you can use the available attributes to examine

the file. For example, if you want to see the size of the image, you can call the "format" attribute.

In this Attendance System, images of students are collected to recognize the faces of the students. The system is initially trained with the student's faces which is collectively known as student database. The system uses user friendly User interface to maximize the user experience while both training and testing which are collecting student images and taking attendance with the system. This project can be used for many other applications where face recognition can be used for authentication. The system will automatically generate reports at the end of semester.

Below are the Modules in our project:

- Registering Face
- Image Training
- Face Recognition
- Database
- Reports Generation

6.10 Registering Face

In this module, first students need to enter their roll number and name. It verifies roll number and name, if provided details does not meet our criteria then students need to re enter their details. Otherwise the system will prompt the Enroll Face button. By clicking on that button, it opens the camera for taking 15 images of the students. While taking the images of a student, we need to recognise the face in the image. For that, here we use haarcascade_frontalface_default.xml file. During registration, the details of the students will be stored in a .CSV File.

6.10.1 Introduction To Haarcascade

A Haar Cascade is basically a classifier which is used to detect particular objects from the source. The haarcascade_frontalface_default.xml is a haar cascade designed by OpenCV

to detect the frontal face. This haar cascade is available on github. A Haar Cascade works by training the cascade on thousands of negative images with the positive image superimposed on it. The haar cascade is capable of detecting features from the source.

6.10.2 Basics

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.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, haar features shown in the image below are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

Now all possible sizes and locations of each kernel is used to calculate plenty of features. (Just imagine how much computation it needs? Even a 24x24 window results over 160000 features). For each feature calculation, we need to find sum of pixels under white and black rectangles. To solve this, they introduced the integral images. It simplifies calculation of sum of pixels, how large may be the number of pixels, to an operation involving just four pixels. Nice, isn't it? It makes things super-fast.

But among all these features we calculated, most of them are irrelevant. For example, consider the image below. Top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applying on cheeks or any other place is irrelevant. So how do we select the best features out of 160000+ features? It is achieved by Adaboost.



Fig 6.10.2.1 Identifying Parts In Face

For this, we apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. But obviously, there will be errors or misclassifications. We select the features with minimum error rate, which means they are the features that best classifies the face and non-face images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then again the same process is done. New error rates are calculated. Also new weights. The process is continued until required accuracy or error rate is achieved or required number of features are found).

Final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95% accuracy. Their final setup had around 6000 features. (Imagine a reduction from 160000+ features to 6000 features. That is a big gain).

So now you take an image. Take each 24x24 window. Apply 6000 features to it. Check if it is face or not. Wow.. Wow.. Isn't it a little inefficient and time consuming? Yes, it is. Authors have a good solution for that.

In an image, most of the image region is non-face region. So it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot. Don't process it again. Instead focus on the region where there can be a face. This way, we can find more time to check a possible face region.

For this they introduced the concept of Cascade of Classifiers. Instead of applying all the 6000 features on a window, group the features into different stages of classifiers and apply one-by-one. (Normally the first few stages will contain very few features). If a window fails the first stage, discard it. We don't consider remaining features on it. If it passes, apply the second stage of features and continue the process.



Fig 6.10.2.2 Identifying Face in a Image

6.11 Image Training

In this module, it takes the 15 images stored in the local computer and train them in a way that it matches, if it comes across the same face again. For image training, here we used Local Binary Pattern Histogram Algorithm (LBPH). It takes images of a students as input and extracts features from them.

6.11.1 Introduction Local Binary Pattern Histogram Algorithm (LBPH)

- **Parameters:** the LBPH uses 4 parameters:
- **Radius:** the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
- **Neighbors:** the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.

- **Grid X:** the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- **Grid Y:** the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

2. Training the Algorithm: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

3. Applying the LBP operation: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors.

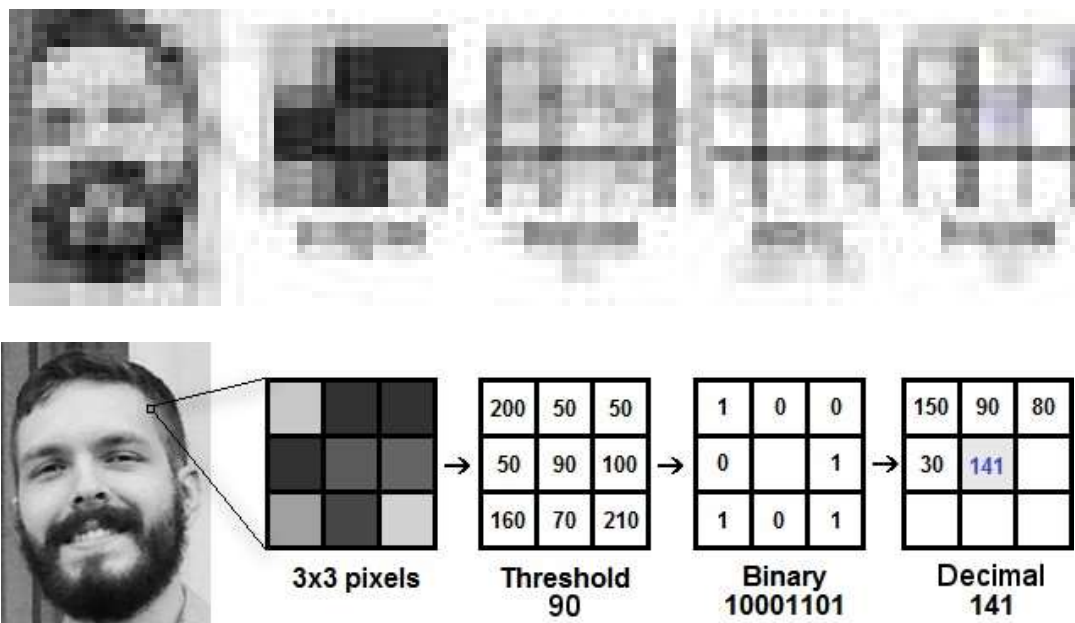


Fig 6.11.1 Pixels Value Calculation in Matrix Form

Based on the image above, let's break it into several small steps so we can understand it easily:

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101).
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.

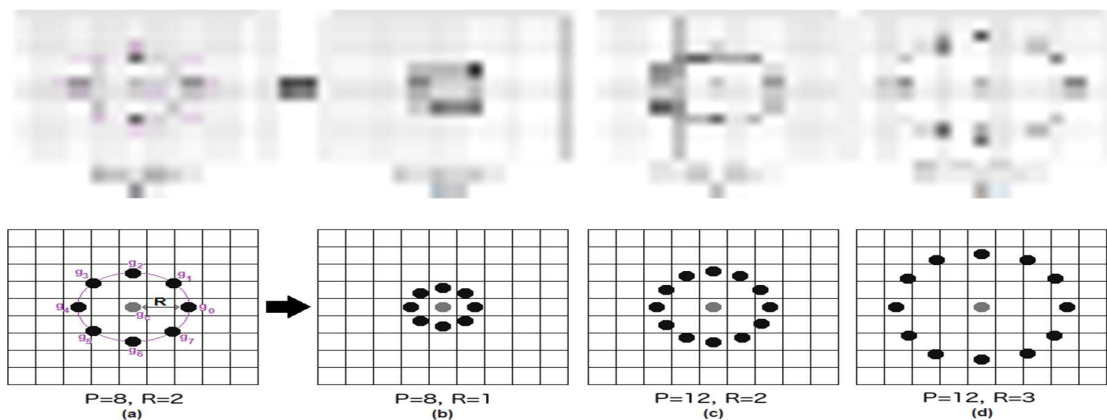


Fig 6.11.2 Circular Local Binary Pattern

It can be done by using bilinear interpolation. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

4. Extracting the Histograms: Now, using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids, as can be seen in the following image:

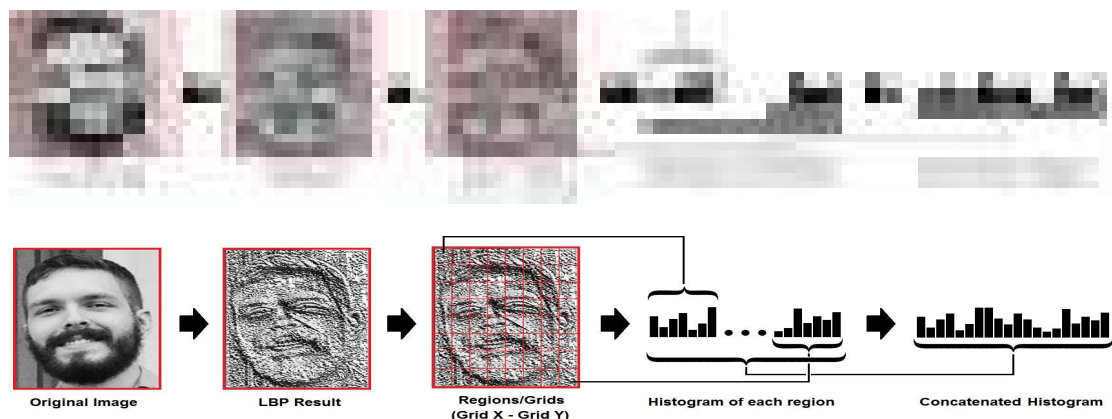


Fig 6.11.3 Grid Conversion

Based on the image above, we can extract the histogram of each region as follows:

- As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.
- Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have $8 \times 8 \times 256 = 16,384$ positions in the final histogram.

5. Performing the face recognition: In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

- So to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.

- We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: euclidean distance, chi-square, absolute value. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^n (hist1_i - hist2_i)^2}$$

- So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a 'confidence' measurement.
- We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

6.12 Face Recognition

In this module, for taking attendance again we need to identify the face of a student in the image and need to present a rectangular box around the face. For that again we will use haarcascade file. The features which are extracted from the images by using LBPH Algorithm are saved in the Trainer.yml file. During attendance time, it will extract the features of the present image and matches them with already existing features, if it finds any match attendance will be taken for that particular student.

6.12.1 Introduction To Trainer.yml file

The YML file format is used for scripts and source code files that follow human-readable data serialization standards, which were developed for a data-centric markup called YAML. The .yaml file extension is used for these files, which are also known as YAML documents. YAML stands for YAML Ain't Markup Language. Formerly known as Yet Another Markup Language, YAML is based on languages like C, Python, Perl and XML.

These .yaml files are also based on the RFC 2822 electronic mail format. To indicate that YAML is more of a data-centric markup instead of a document markup, Yet Another Markup Language was changed to YAML Ain't Markup Language. These YML files can be

opened by using Notepad and other standard text editing applications. These text editing tools can also be used to create and edit the code stored in a .yaml file.

6.12.2 Trainer .yaml file

First create a python “trainer.py” file in the same folder where we saved out dataset generator script in the previous post, and then create a folder in the same directory name it “trainer”, this is the folder where we are going to save our recognizer after training.

6.12.3 Load The Training Data

Ok, now we will be going to create a function which will grab the training images from the dataset folder, and will also get the corresponding Ids from its file name, So I am going to name this function “**getImagesAndLabels**” . We need the path of the dataset folders so we will provide the folder path as an argument. So the function will be like this

```
def getImagesAndLabels(path):
```

So now inside this function we are going to do the following

- Load the training images from dataset folder
- capture the faces and Id from the training images
- Put them In a List of Ids and FaceSamples and return it

To load the image we need to create the paths of the image. This will get the path of each images in the folder. Now we need to create two lists for faces and Ids to store the faces and Ids. Now we will loop the images using the image path and will load those images and Ids, we will add that in your lists

To get the Id we split the image path and took the first from the last part (which is “-1” in python) and that is the name of the imagefile. Now here is the trick, we saved the file name in our previous program like this “**User.Id.SampleNumber**” so if we split this using “.” the we will get 3 token in a list “User”, “Id”, “SampleNumber”. Now we are using the detector to extract the faces and append them in the faceSamples list with the Id

6.13 Database

- The details provided by students during registration are stored in a .CSV File.
- Student's Roll Number and Name will be taken during registration Process.
- Images of a student are stored in a folder in the Local System.
- Images will be stored with the name of student's roll number and name.
- The attendance details of students are also stored in a .CSV File.
- In this file, student's name, roll number, time and date during attendance will be stored.
- Unknown Images during attendance are also stored in a separate folder in Local Computer.
- Trainer.yml will be also saved in a separate file.

6.14 Reports Generation

In this module, the attendance details of the students will be stored in a .CSV File and also shown on the user interface along with student's roll number, name, time and date.

7. CODING

```
import cv2

import csv

import os

import tkinter as t

from tkinter import messagebox as msg

import pandas as pd

import numpy as np

from PIL import Image

import time

import datetime

userent=rolent=nament=""

pasent=clr=clr1=clr2=clr3=lab=r=n=0

clg name="SAI TIRUMALA NVR ENGINEERING COLLEGE"

def verify():

    global r,n

    r=rolent.get()

    n=nament.get()

    if len(r)==10 and len(n)>=1:

        msg.showinfo("info","Verification Successful")

        enr=t.Button(home,text="ENROLL FACE",command=enroll,bg='snow',height=2)

        enr.place(x=230,y=320)
```

```
elif len(n) == 0:

    msg.showwarning("warning", "Please enter your name")

else:

    msg.showwarning("warning", "Wrong Roll Number")

def enroll():

    global n, r

    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)

            sampleNum = sampleNum + 1

            cv2.imwrite("TrainingImage\ "+n+"."+r+"'+ str(sampleNum) + ".jpg",
gray[y:y+h, x:x+w])

            cv2.imshow('Frame', img)

            if cv2.waitKey(100) & 0xFF == ord('q'):

                break

        elif sampleNum > 15:
```

```
        break

    cam.release()

    cv2.destroyAllWindows()

    res = "Images Saved for ID: "+r + "Name: "+n

    row = [r,n]

    with open('StudentDetails\studentDetails.csv', 'a+') as csvFile:

        writer=csv.writer(csvFile)

        writer.writerow(row)

    csvFile.close()

    msg.showinfo("Registered Successfully", "Name:"+n+"\nRoll NUmber:"+r)

def train():

    recognizer = cv2.face_LBPHFaceRecognizer.create()

    harcascadePath = "haarcascade_frontalface_default.xml"

    detector =cv2.CascadeClassifier(harcascadePath)

    faces,Id = getImagesAndLabels("TrainingImage")

    recognizer.train(faces, np.array(Id))

    recognizer.save("TrainingImageLabel\Trainer.yml")

    res = "Image Trained"#+" ".join(str(f) for f in Id)

    msg.showinfo("info",res)

def getImagesAndLabels(path):

    imagePaths=[os.path.join(path,f) for f in os.listdir(path)]

    faces=[], Ids=[]

    for imagePath in imagePaths:
```

```
pillImage=Image.open(imagePath).convert('L')

imageNp=np.array(pillImage,'uint8')

Id=int(os.path.split(imagePath)[-1].split(".")[1])

faces.append(imageNp)

Ids.append(Id)

return faces,Ids

def clear():

    userent.delete(first=0,last=len(userent.get()))

def clear1():

    pasent.delete(first=0,last=len(pasent.get()))

def clear2():

    rolent.delete(first=0,last=len(rolent.get()))

def clear3():

    nament.delete(first=0,last=len(nament.get()))

def track():

    recognizer = cv2.face_LBPHFaceRecognizer.create()

    recognizer.read("TrainingImageLabel\Trainer.yml")

    harcascadePath = "haarcascade_frontalface_default.xml"

    faceCascade = cv2.CascadeClassifier(harcascadePath)

    df=pd.read_csv("StudentDetails\studentDetails.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,1.3,5)

    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(conf)

        if(conf < 75):

            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(20)==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()

att=attendance

def display(att):

    home=t.Tk()

    home.title("ATTENDANCE")

    home.geometry('500x250')

    home.configure(bg='springgreen')

    lab=t.Label(home,text=att,width=50,height=2,bg='snow')

    lab.place(x=50,y=50)

def register():

    global rolent,nament,home
```

```
home=t.Tk()

home.title("REGISTRATION")

home.geometry('700x500')

home.configure(bg='spring green')

roll=t.Label(home,text="ROLL NUMBER",width=15,height=1,bg='snow',font='Arial 12')

name=t.Label(home,text="NAME",width=15,height=1,bg='snow',font='Arial 12')

rolent=t.Entry(home,width=20,font='Arial 16')

nament=t.Entry(home,width=20,font='Arial 16')

clr2=t.Button(home,text="CLEAR",command=clear2,width=10,height=2,bg='snow')

clr3=t.Button(home,text="CLEAR",command=clear3,width=10,height=2,bg='snow')

ver=t.Button(home,text="VERIFY",command=verify,width=10,height=2,bg='snow')

tra=t.Button(home,text="TRAIN FACES",command=train,width=10,height=2,bg='snow')

roll.place(x=70,y=150)

name.place(x=70,y=200)

rolent.place(x=230,y=150)

nament.place(x=230,y=200)

clr2.place(x=510,y=150)

clr3.place(x=510,y=200)

ver.place(x=100,y=320)

tra.place(x=350,y=320)

def attendance():

    home=t.Tk()

    home.title("ATTENDANCE")
```

```
home.geometry('500x250')

home.configure(bg='turquoise1')

tra=t.Button(home,text="TRACK ATTENDANCE",command=track)

tra.place(x=180,y=140)

def homescreen():

    home=t.Tk()

    home.title("HOME SCREEN")

    home.geometry('500x250')

    home.configure(bg='cyan')

    reg=t.Button(home,text="REGISTER",command=register)

    att=t.Button(home,text="ATTENDANCE",command=attendance)

    reg.place(x=140,y=85)

    att.place(x=230,y=85)

def login1():

    global userent,pasent

    a='pj', p='123'

    if userent.get()==a:

        if pasent.get()==p:

            login.destroy()

            homescreen()

        else:

            msg.showwarning("warning","Wrong Password")

            clear1()
```



```
else:

    msg.showwarning("warning","Invalid Credintals")

    clear()

    clear1()

login=tk.Tk()

login.title("LOGIN")

login.geometry('500x300')

login.configure(bg='lightslateblue')

user=tk.Label(login,text="USERNAME",height=1,width=17).place(x=15,y=70)

pas=tk.Label(login,text="PASSWORD",height=1,width=17).place(x=15,y=120)

clg=tk.Label(login,text=clgname,height=2,width=50,font='BOLD',bg='lightslateblue')

clr=tk.Button(login,text="CLEAR",height=1,width=10,command=clear1)

clr.place(x=350,y=120)

clr1=tk.Button(login,text="CLEAR",height=1,width=10,command=clear)

clr1.place(x=350,y=70)

userent=tk.Entry(login,width=15,font='Roman')

pasent=tk.Entry(login,width=15,show='*',font='Roman')

sub=tk.Button(login,text="SUBMIT",command=login1,width=10,height=1)

userent.place(x=180,y=70)

pasent.place(x=180,y=120)

sub.place(x=210,y=180)

clg.place(x=0,y=250)

login.mainloop()
```

8. TESTING

8.1 Testing Methodologies

- Black box Testing
- White box Testing
- Grey box Testing

BLACK BOX TESTING is defined as a testing technique in which functionality of the Application Under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications. In BlackBox Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

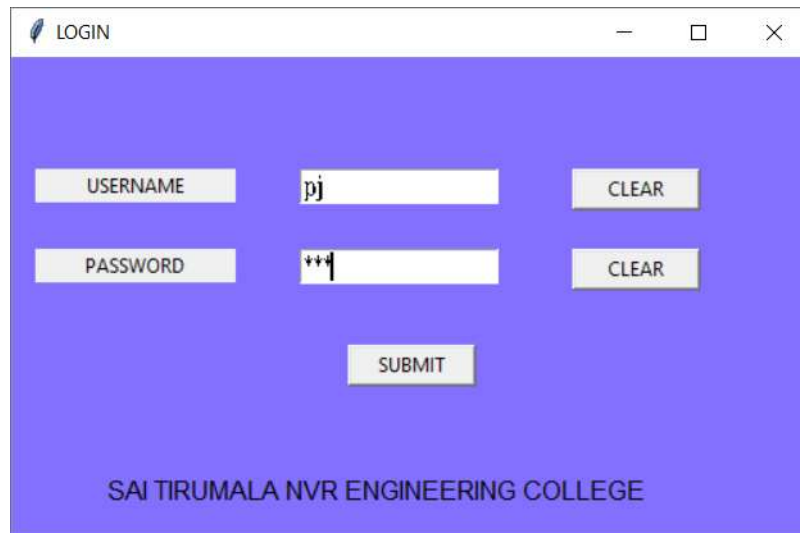
WHITE BOX TESTING is testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers.

GRAY BOX TESTING is a technique to test the software product or application with partial knowledge of the internal workings of an application. The purpose of this testing is to search for defects due to improper code structure or improper functioning usage of an application. In this process, context-specific errors that are related to web systems are commonly identified. It increases the testing coverage by concentrating on all of the layers of any complex system. Gray Box Testing is a software testing method, which is a combination of both White Box Testing and Black Box Testing method.

Test Case No.	Description	Expected output	Actual Output	Result
1	Functionality 1: Open Cam	Camera window being opened on the screen	Camera window opened	Pass
2	Functionality 2: Images Stored	Images being stored	Images being stored	Pass
3	Functionality 3: Open Cam and Detect	Face being Detected/ Identified by the system	Identified Face	Pass
4	Functionality 4: Recognising Face for attendance	Match with existing face for attendance	Match with existing face for attendance	Pass
5	Functionality 5: Recognising twins as different persons	Twins are to be recognised as two separate students	Twins are not recognised as two separate students	fail
6	Functionality 6: Displaying Attendance	Attendance need to displayed on GUI	Attendance displayed.	Pass

Table 8.1 Test Cases

9. RESULTS



LOGIN

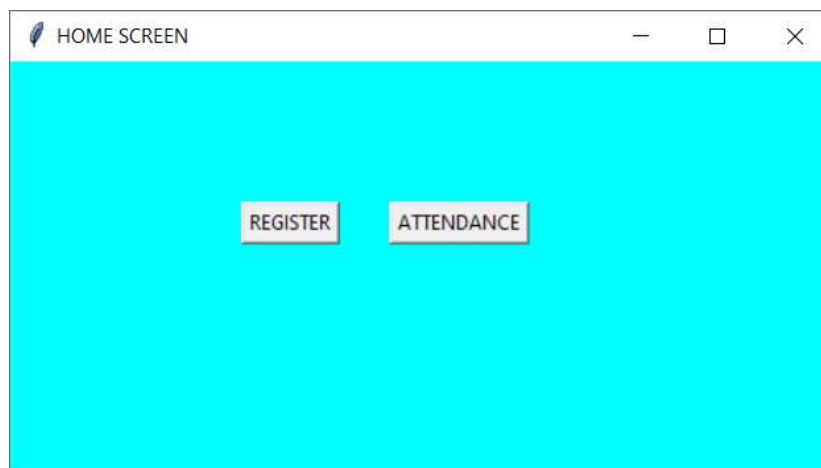
USERNAME CLEAR

PASSWORD CLEAR

SUBMIT

SAI TIRUMALA NVR ENGINEERING COLLEGE

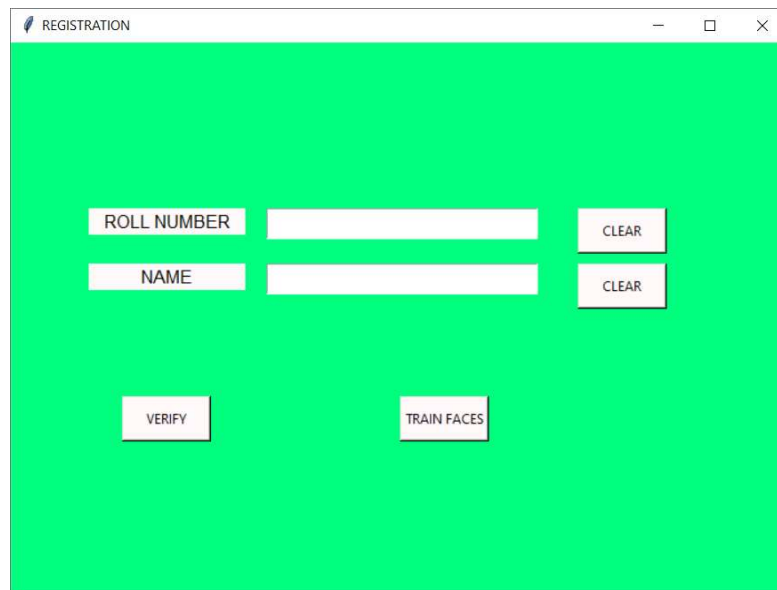
Fig 9.1 Login Screen



HOME SCREEN

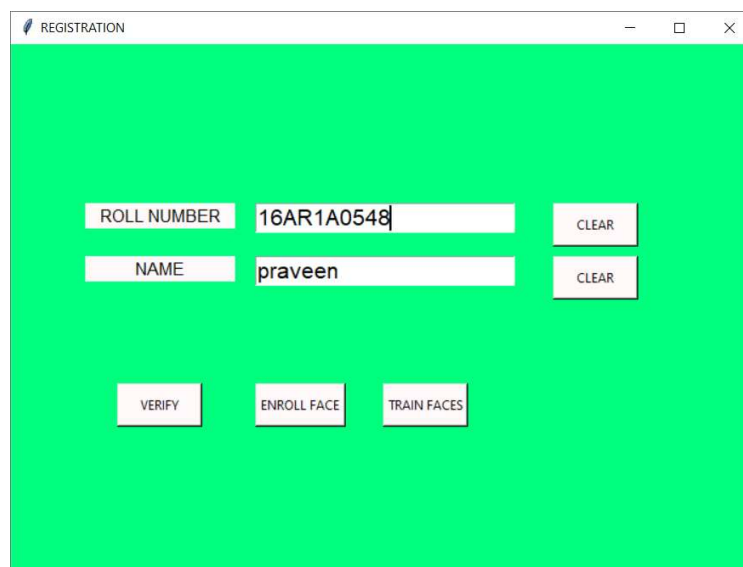
REGISTER ATTENDANCE

Fig 9.2 Home Screen



A screenshot of a web application window titled "REGISTRATION". The window has a light blue background. It contains two input fields for "ROLL NUMBER" and "NAME", each with a "CLEAR" button to its right. Below these fields are two buttons: "VERIFY" and "TRAIN FACES".

Fig 9.3 Registration Screen



A screenshot of the same "REGISTRATION" window, now with data entered. The "ROLL NUMBER" field contains "16AR1A0548" and the "NAME" field contains "praveen". The "CLEAR" buttons are still present. The buttons at the bottom are now "VERIFY", "ENROLL FACE", and "TRAIN FACES".

Fig 9.4 Providing Details

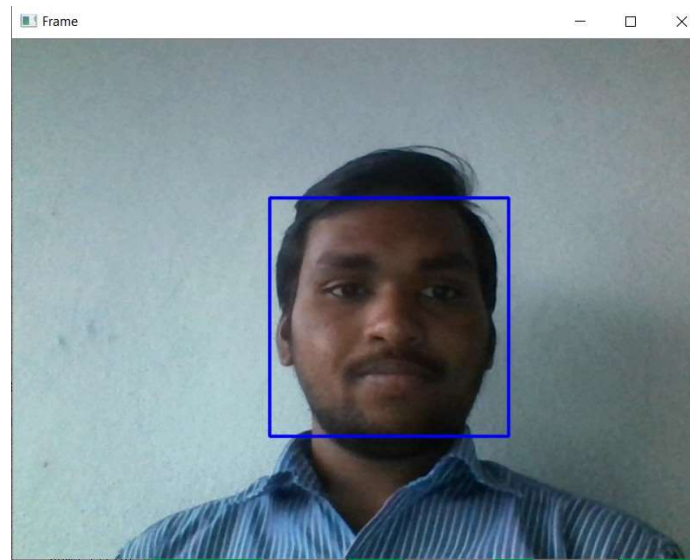


Fig 9.5 Student registering his Face



Fig 9.6 Registered Successfully

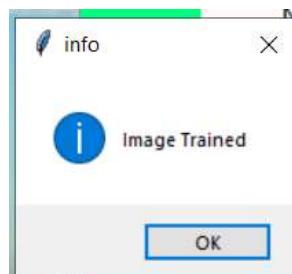


Fig 9.7 Image Trained



Fig 9.8 Attendance Screen



Fig 9.9 Attendance of a student

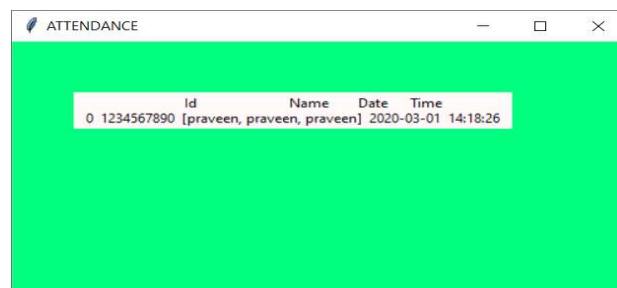


Fig 9.10 Showing Attendance on Screen

10. CONCLUSION

The purpose of reducing the errors that occur in the traditional attendance taking system has been achieved by implementing this automated attendance system. In this project, face recognition system have been presented using deep learning which exhibits robustness towards recognition of the users with accuracy of 98.3% .

The result shows the capability of the system to cope with the change in posing and projection of faces. From face recognition with deep learning, it has been determined that during face detection, the problem of illumination is solved as the original image is turned into a HOG representation that captures the major features of the image regardless of image brightness. In the face recognition method, local facial landmarks are considered for further processing. After which faces are encoded which generates 128 measurements of the captured face and the optimal face recognition is done by finding the person's name from the encoding. The result is then used to generate an excel sheet.

11. FUTURE ENHANCEMENTS

In our proposed system, every student needs to place themselves in front of the webcam or IP camera for attendance. But, in the future we can develop a system like, we will place a High Definition camera in a classroom and it will take a snap of whole students present in the classroom and identify each and every person present in the classroom and will grant attendance for them. We can take images in some intervals and finally at the end of the day, we can grant attendance for students in the classroom. Whole, process would be completed a lot quicker than the proposed system.

12. REFERENCES

- B.K.P. Horn and M. Brooks, Seeing Shape from Shading. Cambridge, Mass.: MITPress, 1989.
- A.F. Abate, M. Nappi, D. Riccio, and G. Sabatino, "2D and 3D face recognition: A survey", Pattern Recognition Letters, vol.28, issue 15, pp.1885-1906, Oct 2007.
- Yael Adini, Yael Moses, and Shimon Ullman, "Face Recognition: The Problem of Compensating for Changes in Illumination Direction".
- Kanan C, Cottrell GW (2012) Color-to-Grayscale: Does the Method Matter in Image Recognition? <https://doi.org/10.1371/journal.pone.0029740>.
- Grundland M, Dodgson N (2007) Decolorize: Fast, contrast enhancing, color to grayscale conversion. Pattern Recognition 40: 2891-2896.
- F. Ibikunle, Agbetuvi F. and Ukpere G. "Face Recognition Using Line Edge Mapping Approach." American Journal of Electrical and Electronic Engineering 1.3(2013): 52-59.
- T. Kanade, Computer Recognition of Human Faces. Basel and Stuttgart: Birkhauser Verlag 1997.
- K. Wong, H. Law, and P. Tsang, "A System for Recognising Human Faces," Proc. ICASSP, pp. 1,638- 1,642, 1989.
- V. Govindaraju, D.B. Sher, R. Srihari, and S.N. Srihari, "Locating Human Faces in Newspaper Photographs," Proc. CVPR 89, pp. 549-554; 1989.
- N. Dalal, B. Triggs "Histograms of oriented gradients for Human Detection", IEEE Computer Society Conference on Computer Vision and Pattern Recognition , Vol. 1, 2005, pp. 886 – 893.