FACE RECOGNITION BASED ATTENDANCE SYSTEM USING MACHINE LANUAGE

A Project Report

Submitted by

PRAVEEN KUMAR.T(411520104073)

NAVEEN KUMAR.E(411520104068)

RAGUL.P (411520104075)

RUTHRESH.P (411520104081)

in partial fulfilment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE

PERI INSTITUTE OF TECHNOLOGY, CHENNAI 600 048

ANNA UNIVERSITY: CHENNAI 600 025

JUNE 2023

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certificate that this project report "**FACE RECOGNITION BASED ATTENDANCE SYSTEM"** is the Bonifide work of "**PRAVEEN KUMAR.T (411520104073), NAVEEN KUMAR.E(411520104068), RAGUL.P(411520104075), RUTHRESH.P(411520104081)**" that carried out the project under my supervision.

SIGNATURE	**SIGNATURE**
Mrs. K. VARALAKSHMI, M.E.	**Mrs. LEKHA**
HEAD OF THE DEPARTMENT	**SUPERVISIOR**
Assistant Professor	Assistant Professor
Department of Computer Science	Department of Computer Science
PERI Institute of technology	PERI Institute of technology
Mannivakkam, Chennai 600 048	Mannivakkam, Chennai 600 048

Submitted for the University Project Viva-voce held on _____

INTERNAL EXAMINAR EXTERNAL EXAMINAR

ACKNOWLEDGEMENT

We express our deep sense of gratitude to our honorable and beloved Chairman **Mr. SARAVANAN PERIASMY** OUR chief Operating officer **Mr.SASIKUMAR VEERARAJAN** and other management members for providing the infrastructure needed.

We express our gratitude to our Principal **Dr. R. PALSON KENNEDY** for his wholehearted encouragement for completing this project.

We convey our thanks to **Mrs.K.VARALAKSHMI** Head of the Department, Department of Computer Science and Engineering, for her kind support and for providing necessary facilities to carry out the project work.

We thank **Mr.A.VIJAYANARAYANAN** the Project coordinators of the Department of Computer Science and Engineering, for their constant guidance and support in the reviews.

We would like to express our sincere thanks and deep sense of gratitude to our guide **Mrs.LEKHA** Department of Computer Science and Engineering for her valuable guidance, suggestions and constant support paved for the successful completion of the project work.

We also extend our heartfelt thanks to all the Staff Members of Computer Science and Engineering Department who have rendered their valuable help in making this project successful.

We are very indebted to our **PARENTS** and **FRIENDS** for the continuous support and encouragement throughout the project successfully.

ABSTRACT

The Python GUI Integrated Attendance System using Face Recognition is a cutting-edge solution for automating the process of taking attendance in academic, professional or industrial settings. This system leverages the power of face recognition algorithms to accurately identify individuals and track their attendance in real-time. The intuitive graphical user interface (GUI) is designed to make the system easy to use, even for those with little or no technical expertise. The attendance data is stored securely in a database, allowing administrators to easily manage, monitor, and analyse attendance patterns and trends. Attendance check plays an important role in classroom management. Checking attendance by calling names or passing around a sign-in sheet is time-consuming, and especially the latter is open to easy fraud. This paper presents the detailed implementation of a real-time attendance check system based on face recognition and its results. To recognize a student's face, the system must first take and save a picture of the student as a reference in a database. During the attendance check, the web camera takes face pictures for a student to be recognized, and then the computer automatically detects the face and identifies a student name who most likely matches the pictures, and finally an excel file will be updated for attendance record based on the face recognition results. In the system, a pre-trained Haar Cascade model is used to detect faces from web camera video. A Face Net, which has been trained by minimizing the triplet loss, is used to generate a 128-dimensional encoding for a face image. The similarity between the encodings of two face images determines whether the two face images coming from the same students. The system has been used for a class, and the results are very satisfactory. A survey has been conducted to investigate the pros and cons of the attendance system on college education management. Overall, this system

provides an efficient, reliable, and secure solution for taking attendance, making it a valuable tool for organizations looking to streamline their operations.

Face detection and face recognition are very important technologies these days, furthermore we noticed that they got have a variety of uses such as cellphones, army uses, and some high risk information offices. We decided to make a device that detects and recognize the face as a student attendance system and can be a substitute for the regular paper attendance system and finger print attendance system. The main function in our project is going to be done using LabVIEW because, LabVIEW is a very helpful programming tool in regards of facial uses and very helpful in other uses. Our project is based on a main program in LabVIEW that detects and recognize faces with giving scores and parameters, furthermore the subsystems are an Excel sheet that is integrated with the program, and a messaging device that is for either a message for absent students or to the student's parents. Components of our project are LabVIEW program as the main system and subsystems, Office Excel sheet to include students names, and a computer (or laptop) to integrate the programs together.

Keywords: Open CV, Biometric, Machine Learning, Python

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

Introduction

1.1Project Definition:

Design of an automaticclass attendance system using face detection algorithm of LabVIEW software. The system requires a video capture device and the running LabVIEW algorithm to be implemented successfully. It detects the faces and mark attendance accordingly. This system will prevent unnecessary wastage of time of classes that is usually wasted in form of class roll calls. his system leverages the power of face recognition algorithms to accurately identify individuals and track their attendance in real-time. The intuitive graphical user interface (GUI) is designed to make the system easy to use, even for those with little or no technical expertise.The attendance data is stored securely in a database, allowing administrators to easily manage, monitor, and analyze attendance patterns and trends.

1.2 Project Objectives:

1. Reducing time wastage during conventional class attendance.
2. Utilizing latest trends in machine vision to implement a feasible solution for class attendance system.
3. Automating the whole process so that we have digital environment.
4. Preventing fake roll calls as one to one attendance marking is possible only.
5. Encouraging the use of technology in daily lives.

1.3 Project Specifications:

a. Uses Pattern Matching algorithm for face detection.

b. Score of minimum 600 required to perfectly match a face.

c. Metric : Camera Resolution.

d. For prototype fixed to 10 users only but scalable design.

e. Requires good lighting condition for better camera capture capability.

f. Attendance sheet is .xlsx format and can be digitally distributed and maintained.

1.4 Product Architecture and Components

1.4.1 Functional Diagram

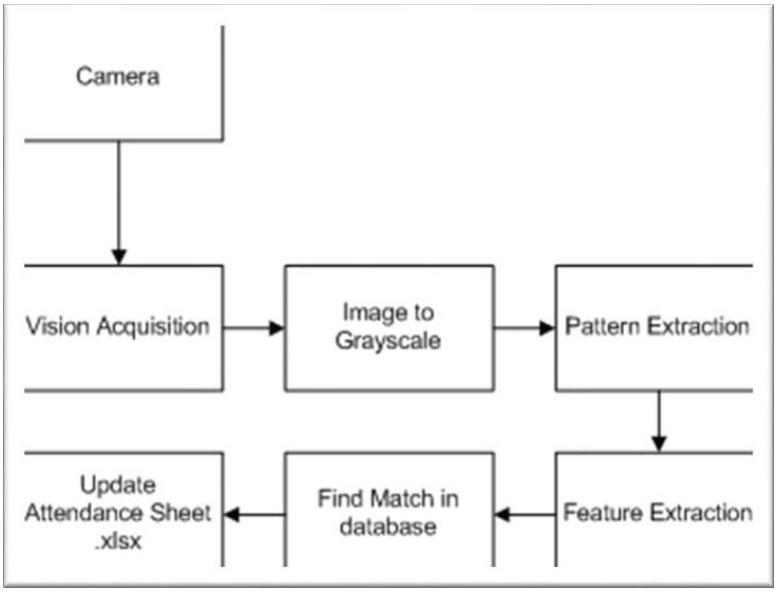


FIGURE 1.1: BLOCK DIAGRAM OF THE SYSTEM

The subsystem description is as follows:

Camera: The camera is the only hardware component required to capture live video feed of class.

Vision Acquisition: This module allows image to be captured by camera into LabVIEW for programming. It includes IMAQ submodules such as IMAQ Create, IMAQ dx Open, MAQ dx Grab. They all combine to provide Continuous Acquisition of video feed from camera module.

Image to Grayscale: This process is performed using IMAQ Extract Single Colour Plane VI to convert a 32/16bit image to 8bit image. This is a requirement for our pattern matching algorithm to work completely.

Pattern Extraction: This is included in Vision Assistant VI which deals with our face recognition algorithm. Pattern Extraction is feature in which the image inputted features are compared using Pattern Matching Algorithm.

Feature Extraction: This feature is used to extract important features out of image. It compares them with templates, saves in database and provides a score of comparison.

Find Match in database: Our database has preserved templates or images of students which we aim to recognize and mark attendance. This database can be updated or appended according to requirement. This database is used for comparison with extracted feature of image to confirm a successful hit.

Update Attendance Sheet.xlsx: If match is found our algorithm updates the attendance of user corresponding to his/her name in excel file of format .xlsx. If not, the system marks absent in front of his/her name in the same excel file.

1.4.2 User Interface Image.

This is the front panel of LabVIEW program that the user is going to be using. It shows the attendance sheet with the names of the students, and a live camera of the user in front of the camera.

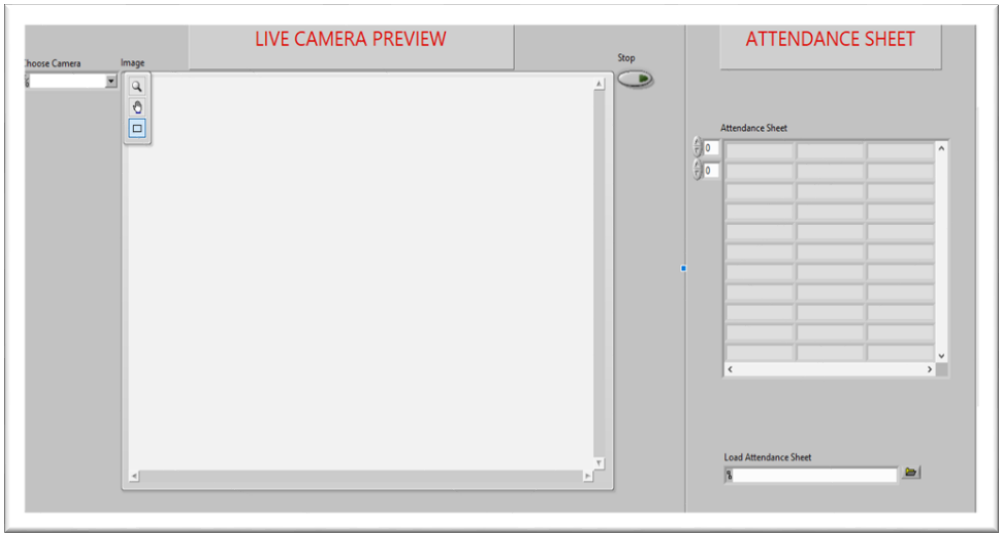
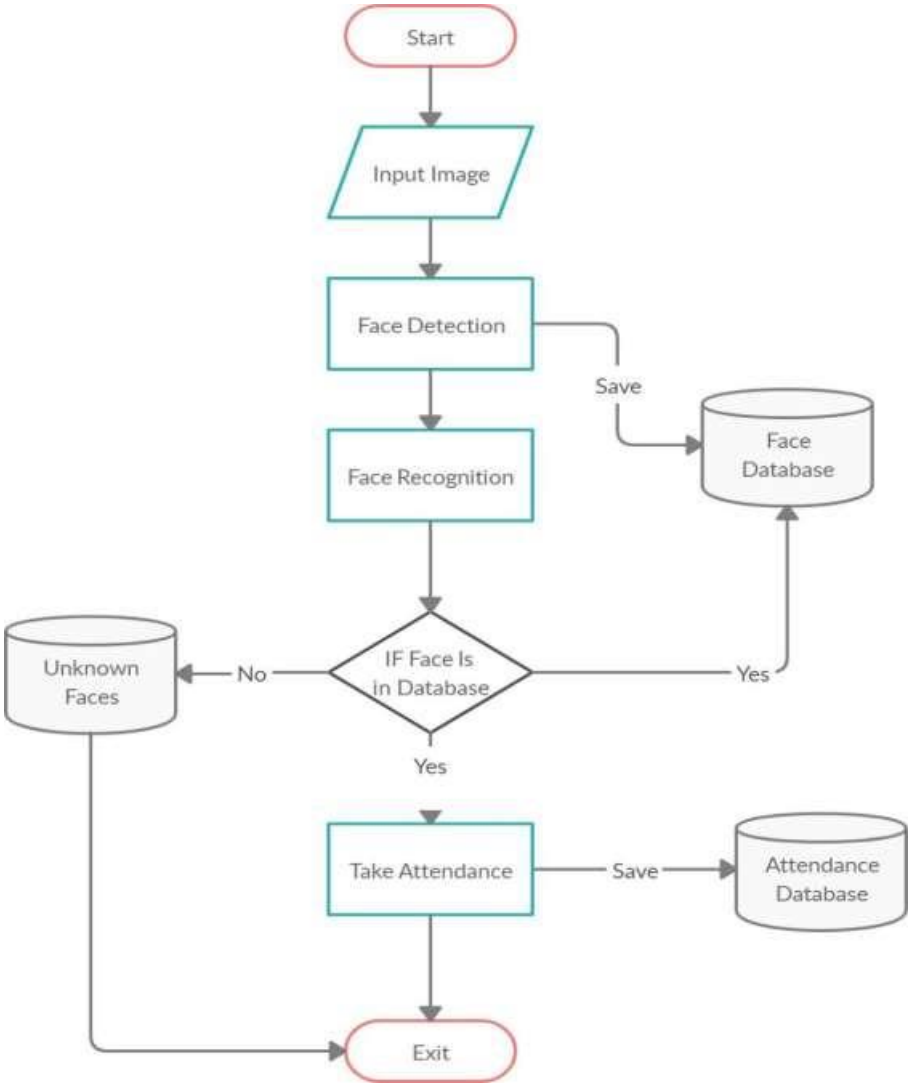


FIGURE 1.2: FRONT PANEL OF LABVIEW PROGRAM

1.5 FLOW CHART



1.3 Project Outline

1.6 Applications

a. Large application in institute attendance system where multiple attendances are carried out for different classes. The attendance will be short timed and reduce manual errors.

b. Large application of computer vision in field of Communication, Biomedical, Automatic Product Inspection.

CHAPTER 2

Literature Review

2.1 Project background

In the face detection and recognition system, the process flow is initiated by being able to detect the facial features from a camera or a picture store in a memory. The algorithm processes the image captured and identifies the number of faces in the image by analyzing from the learned pattern and compare them to filter out the rest. This image processing uses multiple algorithm that takes facial features and compare them with known database. The motivation behind this project is to simplify the means by which attendance is taken during lectures and how much time it takes. The use of ID cards or manually calling out attendance and writing it down on sheets is not productive and efficient. This system will detect the number of faces on the class and will also identify them from the store database. With the face detection and recognition system in place, it will be easy to tell if a student is actually present in the classroom or not.

2.2 Previous Work

PROJECT # 1

This is a project done by students as a final year project at Kingston University London in 2018.

The system will be presented an image either via camera or from memory and it must detect the number of faces on it automatically. After identifying faces, the system should crop the faces from the image and store them in memory for image recognition which will be done in the second step. The system should be able to automatically count the number of faces detected on the image. The

second step will be the recognition part where the system will be able to match faces from the stored dataset and compare it to the input data from the first step. A software will be used for this system which automatically sorts out the faces. The software will be inter-active so to facilitate interaction between multiple tasks as required. Because the system has two steps, the second phase of the system will involve the training of images on a dataset that are to be used for recognition.

The system behaviour has been explained in the following flowchart,

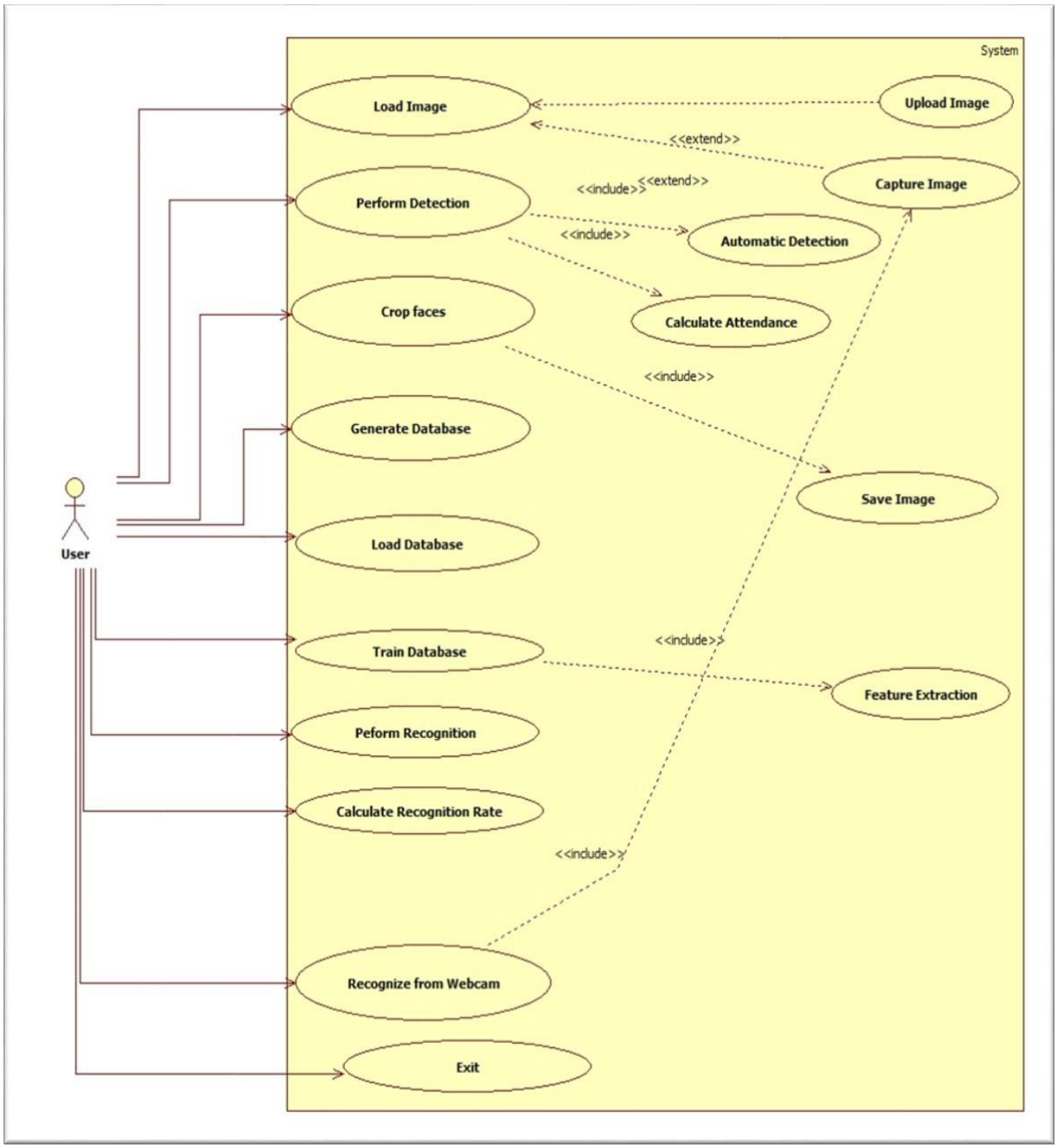


FIGURE 2.1: BLOCK DIAGRAM OF PREVIOUS PROJECT #1

Technology Used :

The key algorithms are Viola-Jones for face detection and Hidden Markov Model with SVD.

- The implementation of The Viola-Jones algorithm is available on softwares like MATLAB, OpenCV and Web Browsers (using adobe flash).
- The existing implementation of the Hidden Markov Model with SVD for face recognition are available on MATLAB, C++ and OpenCV libraries.

PROJECT # 2

This is a project done by students as a final year project at University of Nairobi in 2012.

The system will comprise of two modules. The first module a.k.a face detector is a mobile component, which is basically a camera that captures student faces and stores them in a file using computer vision face detection algorithms and face extraction techniques. The second module is a desktop application that does face recognition of the captured images (faces) in the file, marks the students register and then stores the results in a database for future analysis.

Following flowchart explains the process of the flow of information throughout the process.

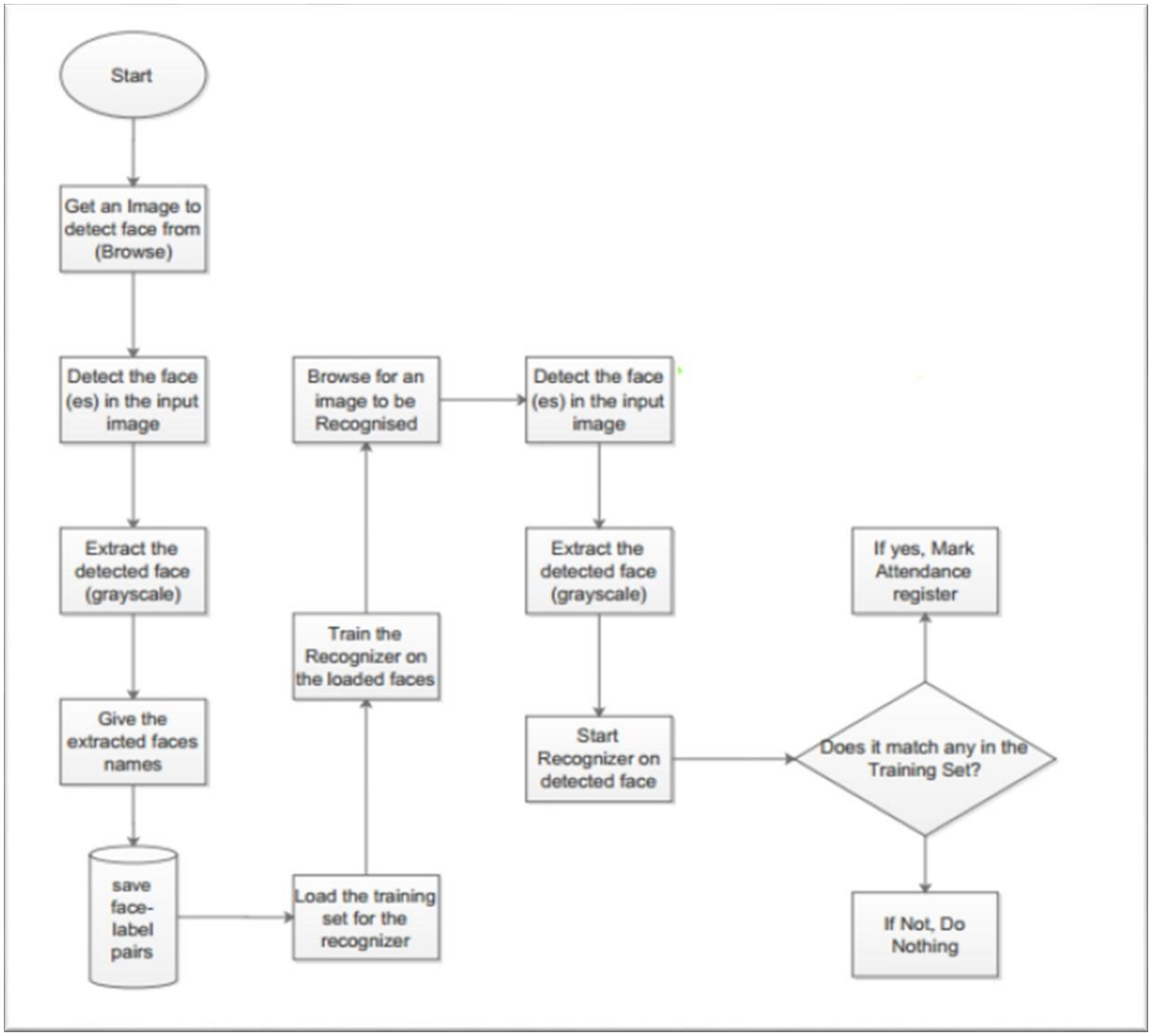


FIGURE 2.2: BLOCK DIAGRAM OF PREVIOUS PROJECT #2

Technology Used

The following tools will be used in the implementation of the designed system. They've been divided in to two categories; Mobile and Desktop tools.

- **Mobile Tools**

The face detection module will use OpenCV library for implementation by use of the frontal HaarCascade face detector in either Android studio.

OpenCV for Android Library - (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision.

Android Studio/ Eclipse IDE - Android Studio is the official IDE for Android application development, based on IntelliJ IDEA.

- **Desktop Tools**

EmguCV Library - EmguCV is a cross platform .Net wrapper to the OpenCV image processing library. OpenCV/EmguCV uses a type of face detector called a Haar Cascade. The Haar Cascade is a classifier (detector) trained on thousands of human faces.

Visual Studio - Visual Studio is able to build and run the solution examples after a proper configuration of EmguCV. The desktop software will implement the two sub-systems (Training set manager and Face recognizer) together with face detector in windows form.

PROJECT #3

This is a project done by students as a final year project at Universiti Tunku in 2018 .The approach performs face recognition-based student attendance system. This method is also similar to others and begins with the input of an image either loaded from memory or from camera. Then it pre-processes the facial features

and extracts it followed by subjective selecting and then the recognition of the facial images from known database. Both LBP and PCA feature extraction methods are studied in detail and computed in this approach in order to make comparisons. LBP is enhanced in this approach to reduce the illumination effect. An algorithm to combine enhanced LBP and PCA is also designed for subjective selection in order to increase the accuracy.

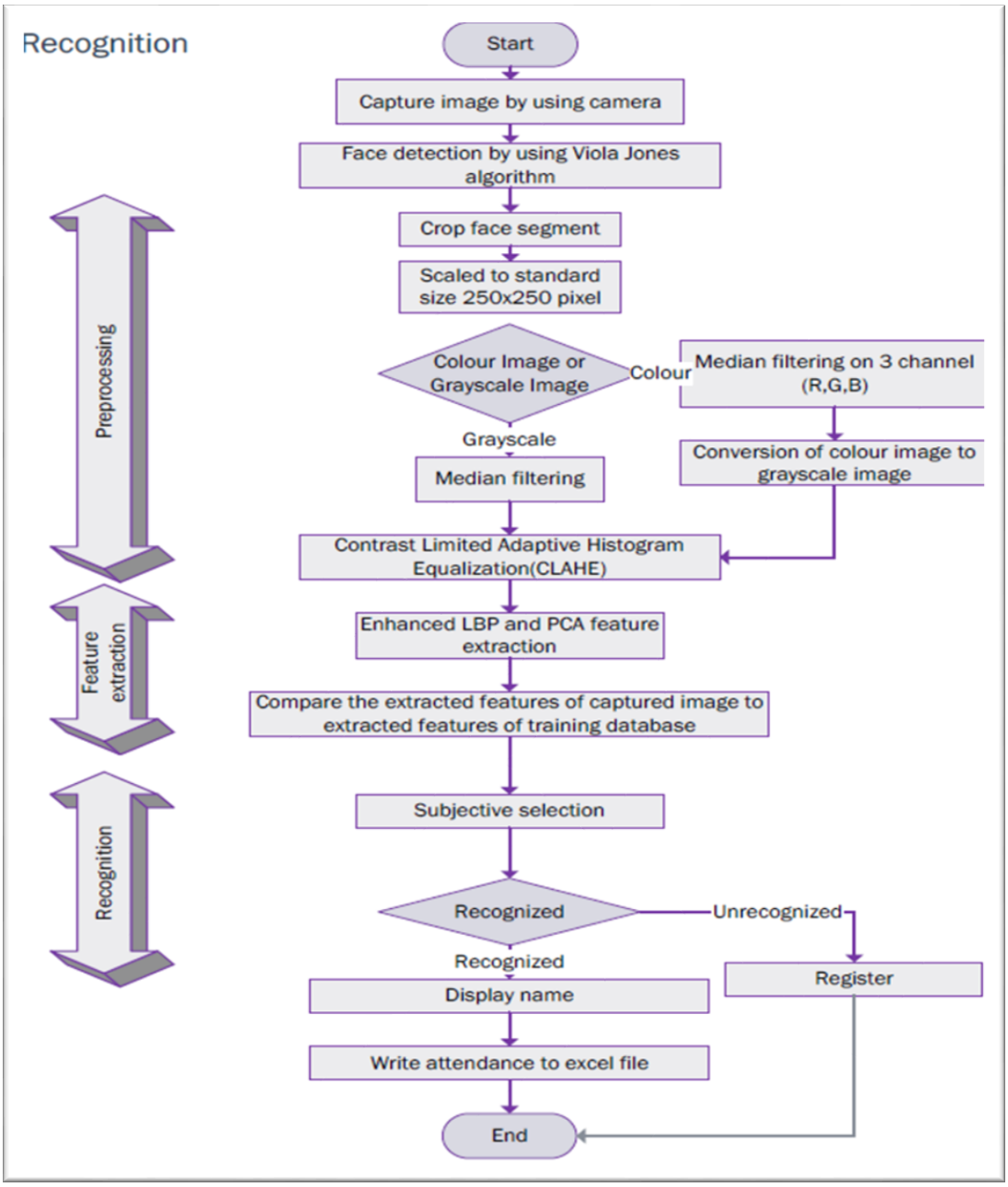


FIGURE 2.3 : BLOCK DIAGRAM OF PREVIOUS PROJECT #3

2.3 Comparative Study

Our project is different than all the previous projects made and mentioned above. They have purely used the core of machine vision to implement a face detection mechanism. None of the above mentioned projects have realized the power of LabVIEW programming and LabVIEW Vision modules in which not only pattern matching but other machine vision algorithms like edge tracking, geometric matching can be implemented with ease. Though the general mechanism and flow of events is similar in above projects and our current project however, the mechanism of face detection is completely unique and different.

Projects	1	2	3		Our Project
Face Recognition & Detection	√	√	√		√
Communication GSM, Zigbee, WiFi	GSM	GSM	Wi-fi		Wi-Fi
Time Saving	√				√
Market Demand	√	√	√		√
Local Usage in Schools		√	√		√
Data Saving in Record (Monitoring)	√				√

FIGURE 2.4: COMPARESSON BETWEEN ALL PROJECTS

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system for attendance tracking typically involves manual methods such as paper-based registers or sign-in sheets. In this system, individuals have to physically sign their names or mark their attendance, which can be time-consuming and prone to errors. It also requires manual effort to maintain and manage the attendance records.

3.1.1 DISADVANTAGES

- Implementing a face recognition-based attendance system requires adequate infrastructure, including cameras, hardware, and software.
- Facial recognition algorithms have been criticized for exhibiting bias, particularly in their performance across different demographics.

3.2 PROPOSED SYSTEM

The proposed system is a face recognition-based attendance system that leverages computer vision and machine learning techniques to automate the process of attendance tracking. The system captures images of individuals' faces during an initial enrollment phase. These images are then processed and used to create a unique face template or facial feature representation for each individual. The face templates are stored in a database for future reference.

When individuals arrive at the designated attendance area, cameras or other imaging devices capture their faces. The system analyzes the captured face images and extracts facial features. The extracted features are compared with the

stored face templates in the database to identify the individual. If a match is found, the attendance is marked for that person, and relevant records are updated.

3.2.1 OBJECTIVE

The objective of a face recognition-based attendance system is to automate the process of recording and tracking attendance in various settings, such as educational institutions, workplaces, or events, using facial recognition technology. The system aims to replace traditional methods of attendance-taking, such as manual roll calls or swipe cards, with a more efficient and accurate approach.

3.2.2 ADVANTAGES OF PROPOSED SYSTEM

- The system automates the attendance tracking process, eliminating the need for manual sign-in sheets or registers.
- The system can process attendance quickly, making it suitable for large groups of individuals.
- The system can provide real-time attendance updates, allowing administrators to track attendance instantly.

3.2.3 SYSTEM REQUIREMENTS

Hardware Requirements

- **Camera:** A high-resolution camera capable of capturing clear facial images is essential. Ideally, it should support HD or higher resolution.
- **Processor:** A powerful processor capable of handling the image processing and recognition algorithms efficiently. A multi-core processor with a clock speed of 2 GHz or higher is recommended.
- **Memory (RAM):** Sufficient RAM to accommodate the processing requirements of the face recognition algorithms and the operating system.

A minimum of 4 GB RAM is recommended, but more may be necessary for larger-scale deployments.

- **Storage:** Adequate storage to store the captured images and associated attendance records. The amount of storage required will depend on factors like the number of users, frequency of attendance, and image resolution. SSD storage is preferable for faster access.
- **Network:** A reliable network connection to facilitate real-time communication, especially if the attendance data needs to be stored or processed remotely.

Software Requirements

- **Operating System:**

The system should support the required operating system (e.g., Windows, macOS, Linux) and its associated dependencies.

Face Recognition Software:

A robust face recognition algorithm or software library that can accurately detect and recognize faces from captured images or video streams. Popular libraries include OpenCV, Dlib, and TensorFlow.

- **Database Management System:**

A database system to store and manage user information, attendance records, and associated data. Options include MySQL, PostgreSQL, or MongoDB.

- **Attendance Tracking Application**:

An application or software layer that integrates the face recognition algorithms, database management, and attendance tracking features. This application should provide a user-friendly interface for administrators and users to manage attendance.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Face detection involves separating image windows into two classes; one containing faces (turning the background (clutter). It is difficult because although commonalities exist between faces, they can vary considerably in terms of age, skin colour and facial expression. The problem is further complicated by differing lighting conditions, image qualities and geometries, as well as the possibility of partial occlusion and disguise. An ideal face detector would therefore be able to detect the presence of any face under any set of lighting conditions, upon any background. The face detection task can be broken down into two steps. The first step is a classification task that takes some arbitrary image as input and outputs a binary value of yes or no, indicating whether there are any faces present in the image. The second step is the face localization task that aims to take an image as input and output the location of any face or faces within that image as some bounding box with (x, y, width, height).After taking the picture the system will compare the equality of the pictures in its database and give the most related result.

4.2 Design Constraints

The constraints which were considered while designing on project are following.

4.2.1 Design Constraint: Engineering Standards

The samples for database should be increase, as to increase the efficiency of detection. Also, the more the expensive the camera, the easier its algorithm is likely detecting the person.

4.2.2 Design Constraint: Environmental

The camera should capture all the students present in the class. Each student present should be seated such that it is visible to camera, so that his/her attendance gets marked easily.

4.2.3 Design Constraint: Ethical

The second limitation which is faced include the person appearance by face, which a person changes his/her look and looks different from the picture in the database of the attendance system, then it may be difficult for his/her attendance to be marked.

4.3 Design Methodology

As we mentioned before in (Figure 1.1). The project process is:

- A camera will take continuous stream.
- In LABVIEW, IMAQ library for vision will be used.
- Convert the RGB image to grayscale image.
- Then perform Machine Vision Algorithm and match with patterns stored in our database.
- If pattern matches based on the score of how successful, decide to mark attendance or not.
- Update the marked attendance in a measurement file.

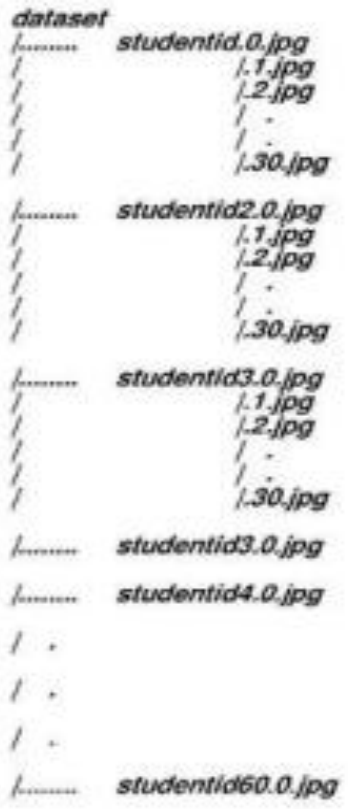
4.3.1 METHODOLOGY:

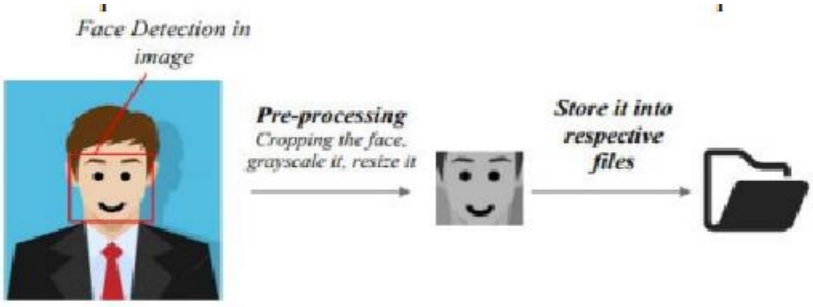
The proposed system is designed for automating the attendance of the different organization and reduces the flaws of existing manual system. he system calculate the attendance subject wise, that is the data of students and subjects are added manually by administrator, and whenever time for corresponding subject arrives the system automatically starts taking snaps and find whether human faces

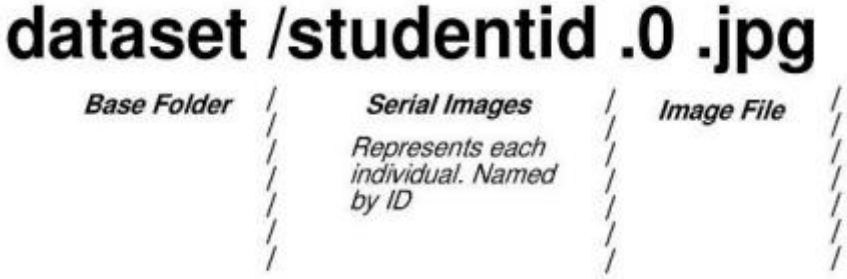
are appear in the given image or not. We have used Histogram of Oriented Gradient for face detection and deep learning techniques to calculate and compare 128-d face features for face recognition. Once faces are detected and recognize with the existing database, system calculate attendance for the recognize students with the respective subject id in real time. And an excel sheet generated and saved by the system automatically Before the attendance management system can work, there are a set of data needed to be inputted into the system which essentially consist of the individuals basic information which is their ID and their faces. The first procedure of portrait acquisition can be done by using the Camera to capture the faces of the individual. In this process the system will first detect the presence of a face in the captured image, if there are no face detected, the system will prompt the user to capture their face again until it meets certain number of portraits which will be 10 required portraits in this project for each student. The decision of storing only 10 portrait per student is due to the consideration of the limited storage space in the raspberry pi because the total amount of students in the university is considered heavy. Then, the images will undergo several pre-processing procedures to obtain a grayscale image and cropped faces of equal sized images because those are the prerequisites of using the Eigen Faces Recognizer. Both of the processes mentioned above can be represented in the diagram below.

4.3.2 Image Acquisition and Pre-processing procedures:

Hierarchy manner of the face database:







4.4 Product Subsystems and Components

4.4.1 Product Subsystem1: Vision Acquisition

This subsystem is used to acquire continuous stream of video from attached camera. It starts a camera session from desired camera and transmits its image feed to further processing. The feed captured is inline processed and then the next feed is captured as shown in (Figure 4.1).

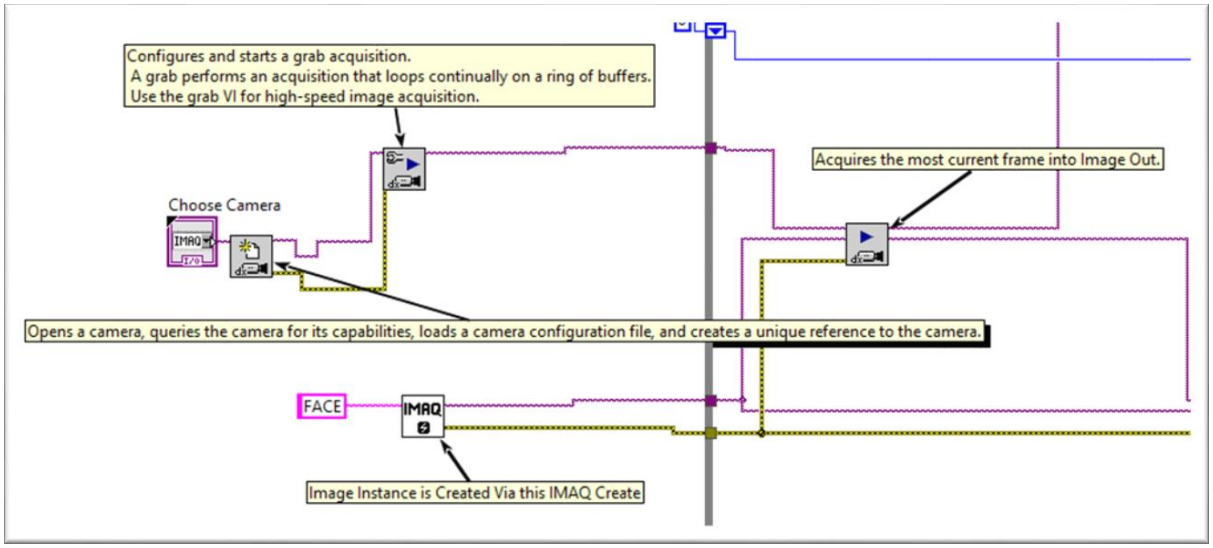


FIGURE 4.1: VISION ACQUISITION PROGRAMING ICONS

4.4.2 Product Subsystem2: Grayscale Conversion

This module converts an RGB (32bit) image to Grayscale (Unsigned 8 bit) image. This is requirement of Vision Assistant Processing whose image input must be 8 bits as shown in (Figure4.2)

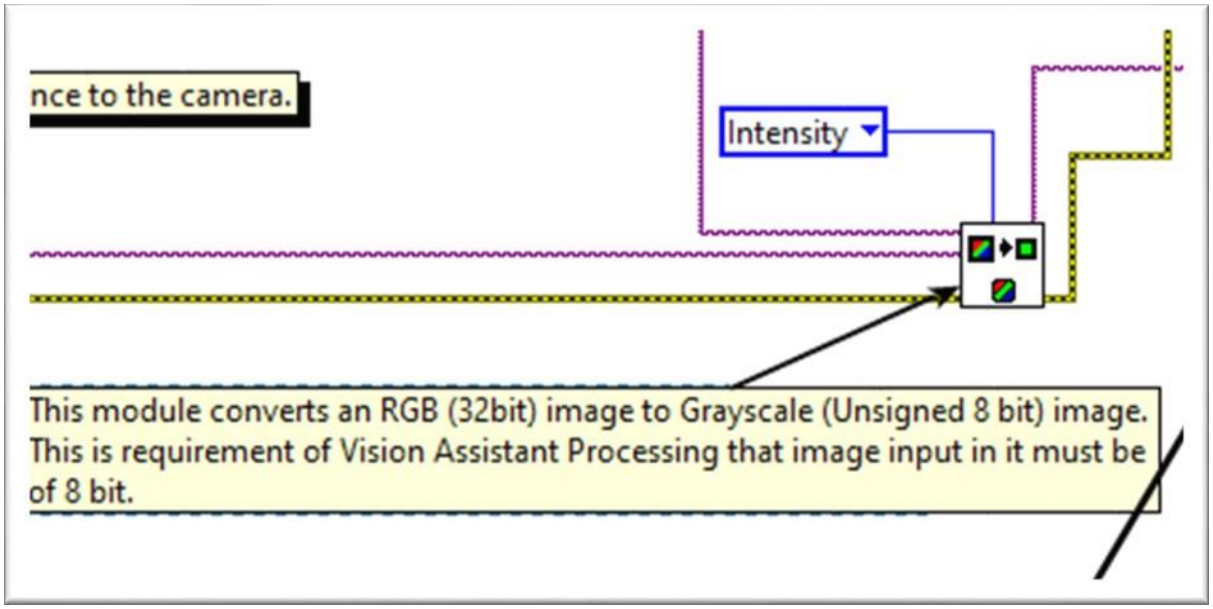


FIGURE 4.2: GRAYSCALE VISION CONVERSION ICON

4.4.3 Product Subsystem3: Vision Assistant

In (Figure 4.3) Vision Assistant helps us to perform Machine Vision Algorithm Pattern Matching on our image. This allows us to detect faces of student in a group of class. First one must add student faces as template in this program to create a database of images using reference images. The result is out puted which includes the information of score 0-1000 to tell how successful a match was, position of match occurred in image, angle of match occurred in image. This information together with number of matches for each user will be used to mark attendance of user in future progresses.

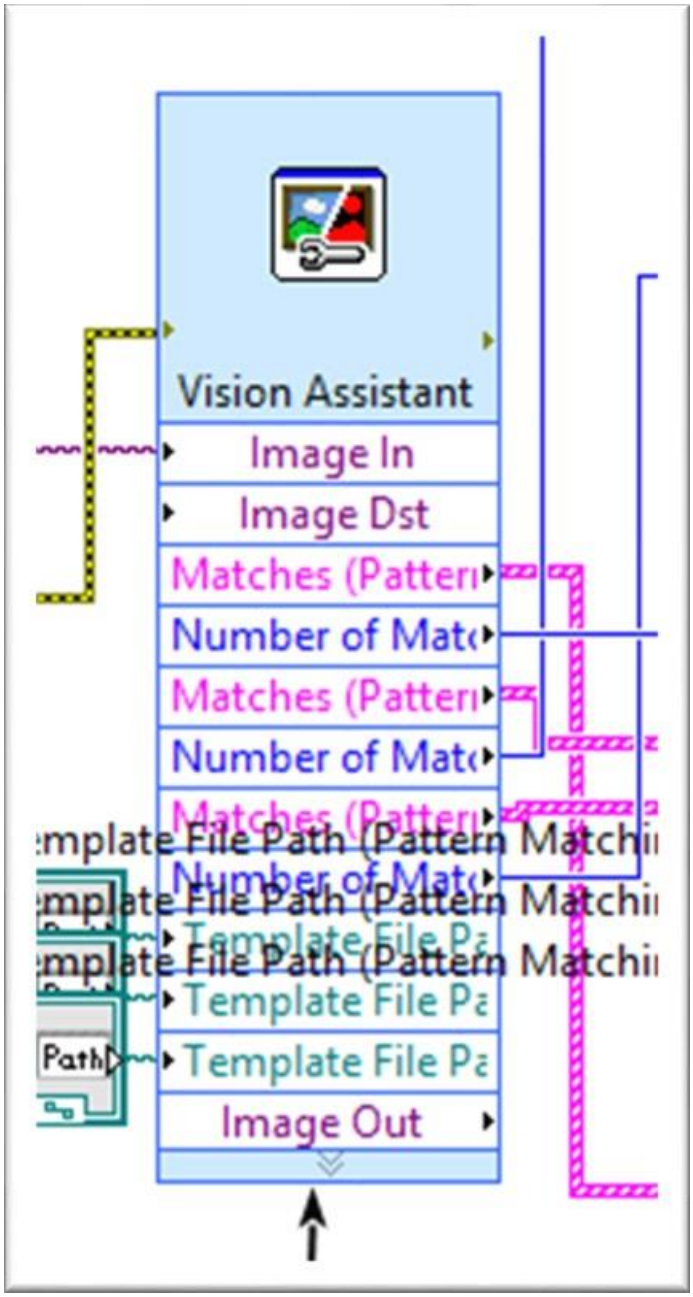


FIGURE 4.3: VISION ASSISTANT ICON

4.4.4 Product Subsystem4: Writing in a Measurement File

Each Student name and his/her attendance is marked if the case is true, else the attendance marked absent. The file path to the spread sheet delimiter is provided and the delimiter used to separate each student data is new line (new row). The system also caters a latch in form of a OR gate in which once a student is detected the system will maintain its value no matter if he/she gets out of frame and is undetected as demonstrated in (Figure 4.4)

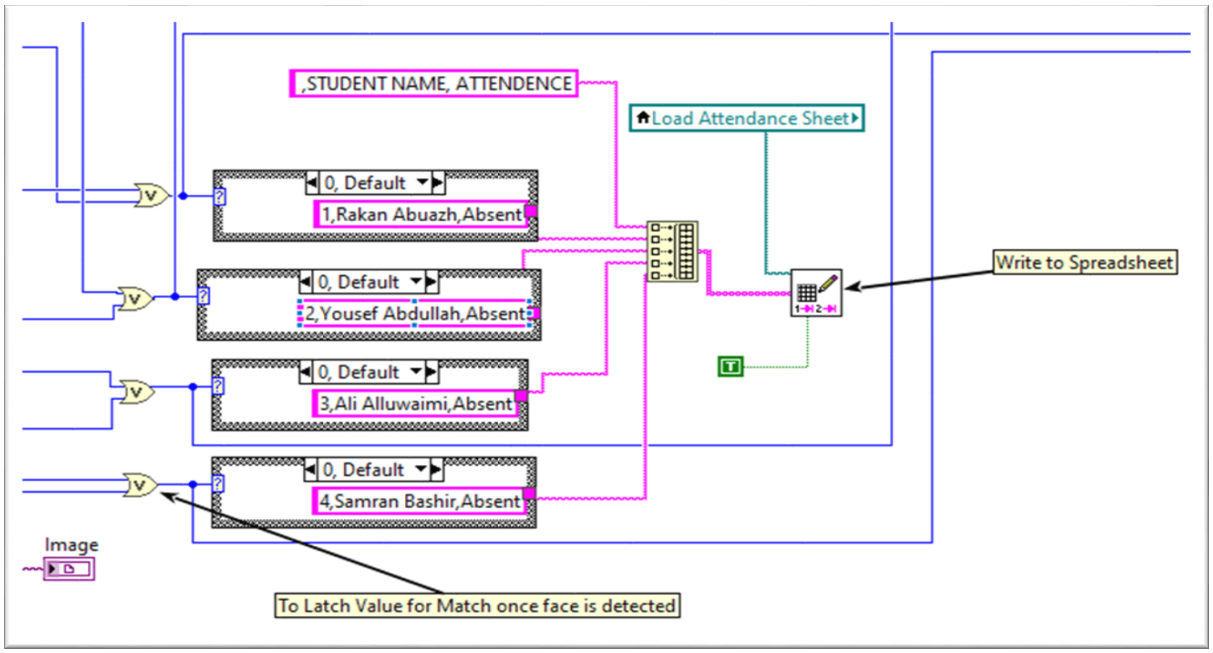


FIGURE 4.4: WRITING ON SPREADSHEET NAMES

4.5 Implementation

4.5.1 THE MAIN MENU:

This is the first prompt to appear when program is run. It will ask user to either proceed further for face detection or exit the program as demonstrated in (Figure 4.5).

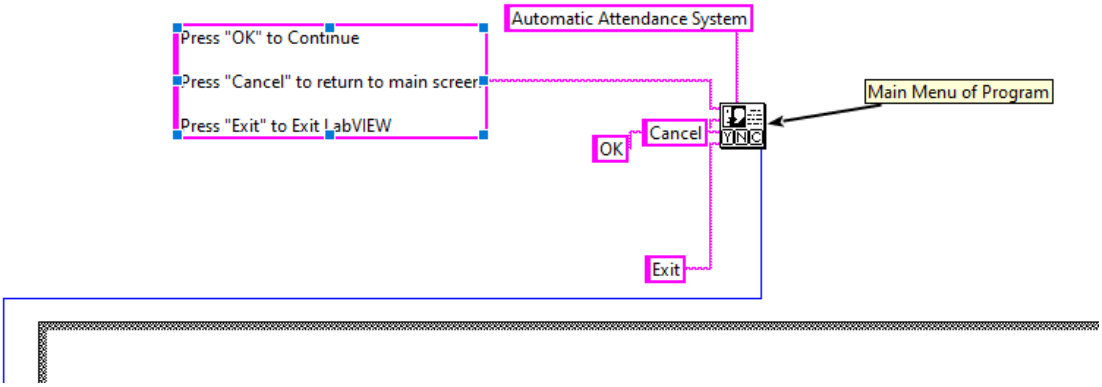


FIGURE 4.5: MAIN MENU ICONS

4.5.2 THE VISION ACQUISITON MODULES:

Vision acquisition is responsible of the live camera in front panel, changing the image to greyscale so the Vision Assistant can accept it, and the camera choice as shown in (Figure 4.6).

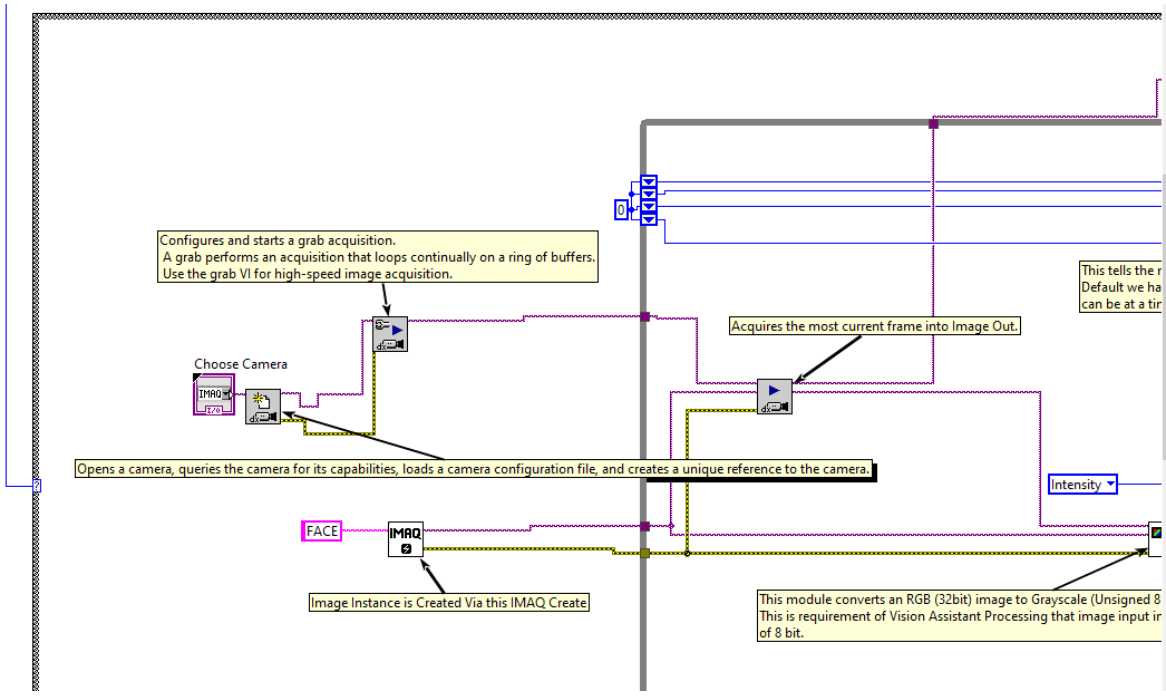


FIGURE4.6: VISION ACQUISITION MODULES

4.5.3 THE VISION ASSISTANT MODULE:

This is the Vision Assistant where processing is performed. But it is the requirement of Vision Assistant to first include the picture of each student for matching. We can add as many patterns as we want.

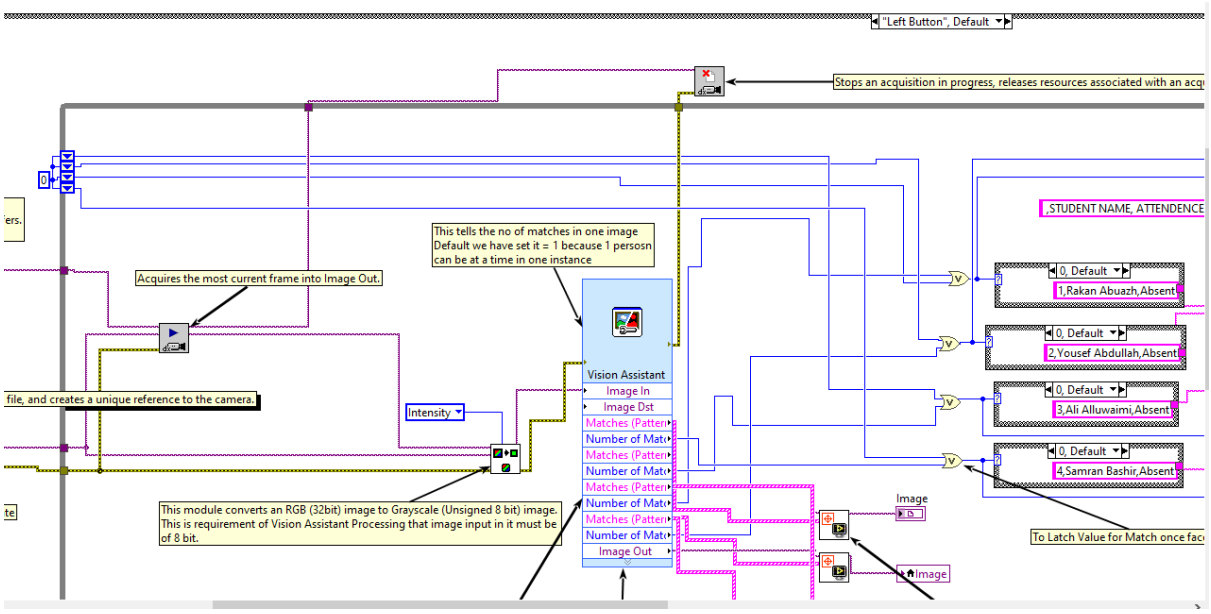


FIGURE 4.7: VISION ASSISTANT MODULES

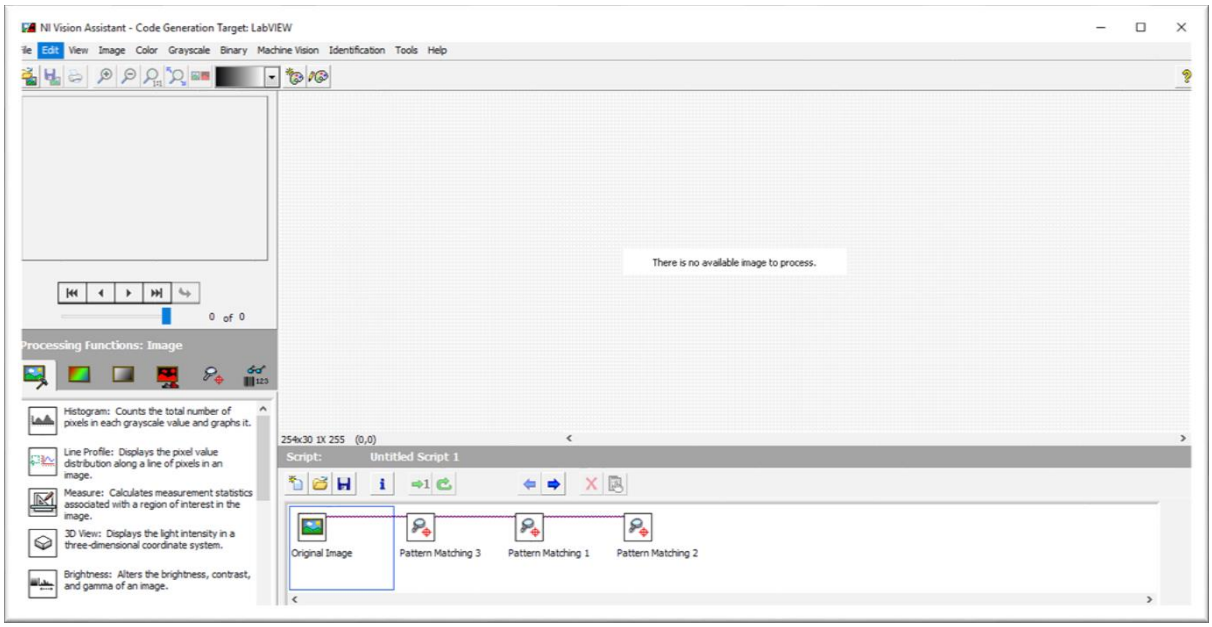


FIGURE4.8: INSIDE OF VISION ASSISTANT

This is the front panel where the capture picture of the class is shown in image out. We have to add path for the save attendance file where it needs to be stored and also the path of the pattern of each student, three cases are generated. You can increase the pattern from Vision Assistant block as already stated above. The marked attendance is updated in the file and also shown in the indicator in front panel as shown in (Figure 4.9).

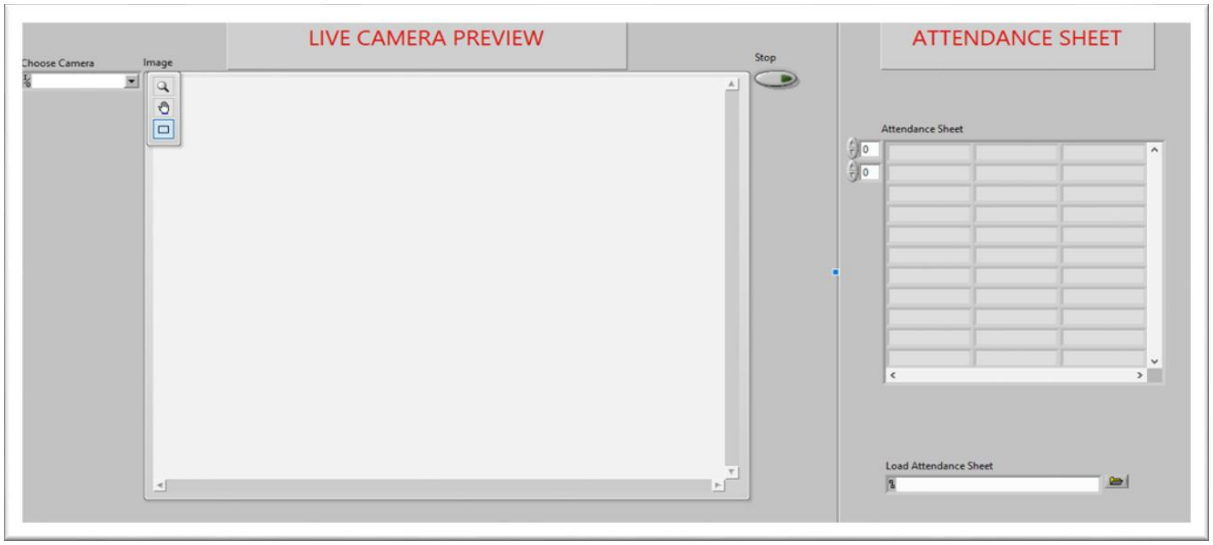


FIGURE4.9: FRONT PANEL

CHAPTER 5

ALGORITHMS

PRINCIPALCOMPONENTANALYSIS(PCA)

PCA is a useful statistical technique that has found application in fields such as face recognition and image compression, and is a common technique for finding patterns in data of high dimension. Eigen faces approach is a principal component analysis method, in which a small set of characteristic pictures, are used to describe the variation between face images. Eigen faces approach seems to be an adequate method to be used in face recognition due to its simplicity, speed and learning capability.

5.1 ALGORITHM STEPS

1. All training set images are resized and converted into a single vector and stored in the Database.
2. Then testing image is resized and converted into a single vector
3. Mean image of all training set images and testing images are calculated.
4. Then the mean image is subtracted from each image of the training set as well as from the test image. After subtraction we will get new images called as difference images.
5. All difference images of training set as well as testing image are converted into a column vector i.e. column-wise concatenation of all images.
6. Then using covariance matrix the eigenvector and eigenvalues are calculated. Each eigenvector belongs to one of the eigenface.
7. Using product of each eigen images with the difference images will get the weight vector of each class as well as the weight vector of the test image.
8. . Then the weight of the test image is subtracted from eachweight vector of the difference image

CHAPTER 6

SYSTEM IMPLEMENTATION

INTRODUCTION

An attendance system using Face Recognition feature with OpenCV library of Python. You can create a dataset of your face and train the system with that dataset, with this trained model we implemented attendance system to recognize the face and mark the attendance of user using provided user id.

UML diagrams

Using UML techniques, we built:

- Use Case diagram
- Activity diagram
- Status diagram
- Class diagram
- Component diagram
- Interaction diagram

6.1 Use Case Diagram:

The benefit of use case diagrams is mostly based on communication between the request team and the user group. A use case specification document should cover the following areas:

- **Actors -** participating in and interacting in this use case
- **Preconditions -** must be met for the use case to work
- **Unconditional -** defines the various states in which the system is expected to be after it is executed. The Use Case diagram lists the basic events that will Occur when the system is executed. It includes all the primary actions that the system must perform.

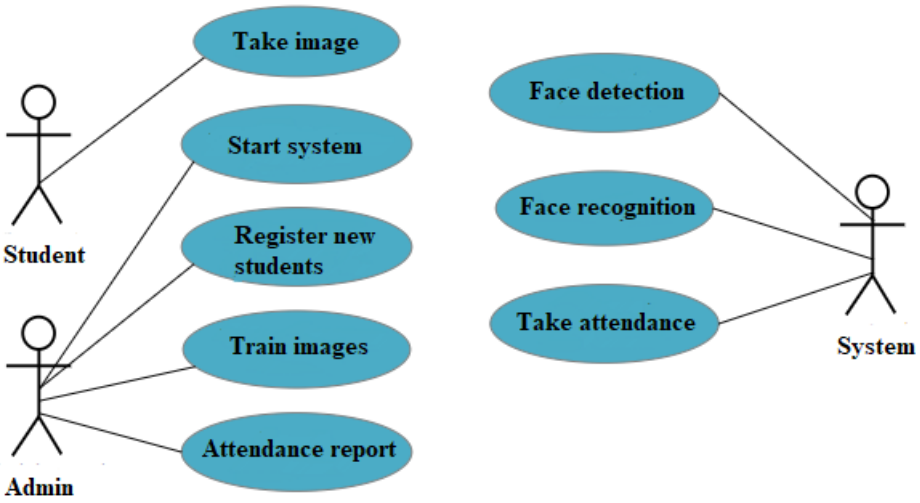


Figure 6.1. Use case diagram

6.2 Activity Diagram:

The activity diagram is a presentation in the form of diagrams keeping a hierarchy of activities. These diagrams are important to understand the way the system works as well as the flow of activities previously stated in the use case diagram. Activities are states of action that automatically switch to another state after the action is completed, but always starting with the "circles" that symbolize the beginning and the end as well as the "arrows" which display the transition from one activity to the next. For our system, we have presented three activity diagrams to create an overview of how the system functions flow. **Figure 6.2.1** shows the activity diagram, which shows the flow of events in the process when we are dealing with new students, who must first do the face registration in which case the system stores them, then training, and then the presence with these students can be noted. And finally, we have the generation of the report which is the result of the preliminary step respectively the activity of marking the presence. **Figure 6.2.2.** shows a simple diagram of the activity in which the operation of the whole system is presented in only 4 basic activities that the system performs. **Figure 6.2.3** illustrates the activity diagram for how the system works in the facial

features registration section, the processes which are performed before the presence marking process takes place. After going through the defined steps face image recognition can be performed successfully and, in this case, the assigned student is added to the presence preservation report otherwise, the error message is displayed on the screen, and the reason why it is not executed with order process success.

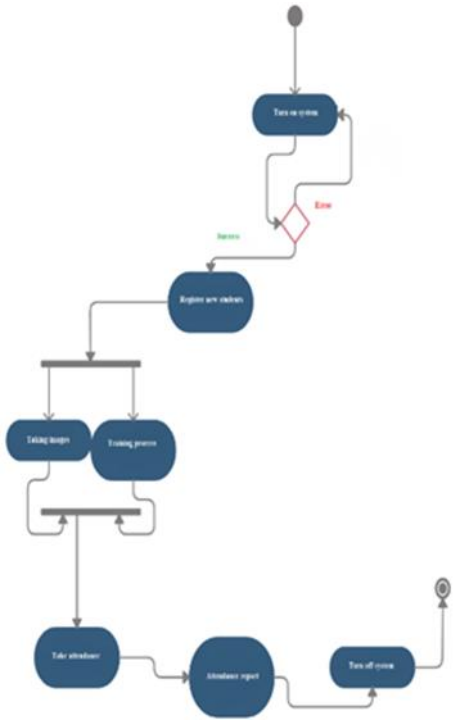


Figure 6.2.1. Activity diagram 1

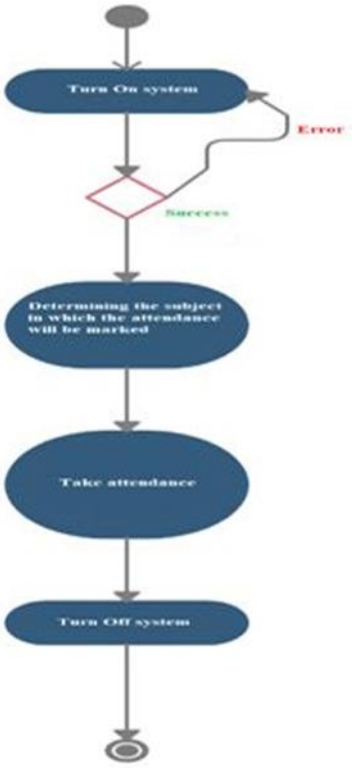


Figure 6.2.2. Activity diagram 2

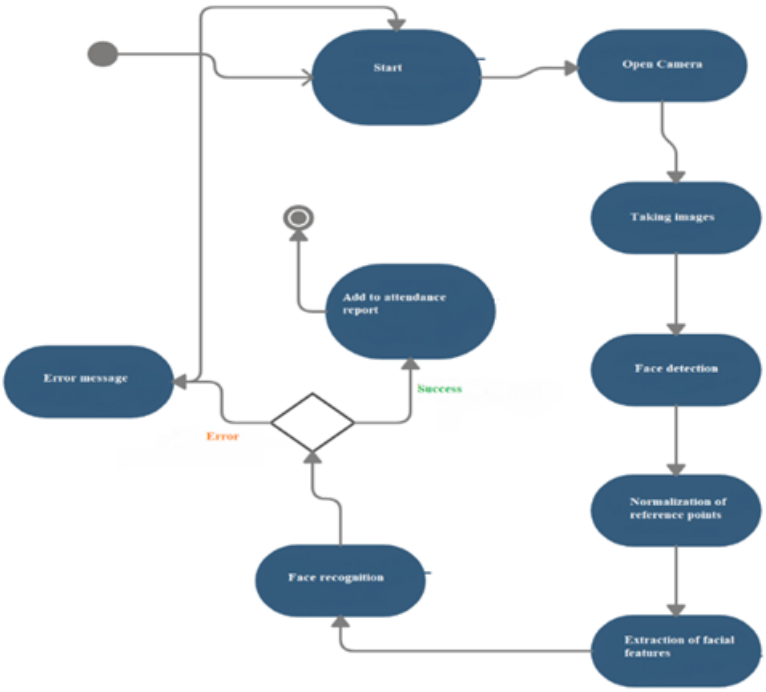


Figure 6.2.3. Activity diagram 3

6.3 State Diagram:

The state diagram helps us to better understand the simple functions as well as the complex ones that the system offers. By using the state diagram, we identify the dynamic behavior of the system as a whole or even of the subsystems. Exactly with the help of the state diagram, we determine the different states of the participants during the operation with the system. Figure 6 shows the situation diagram specifying all the necessary actions for the implementation of defined functions, as well as the states of the participants in the system.

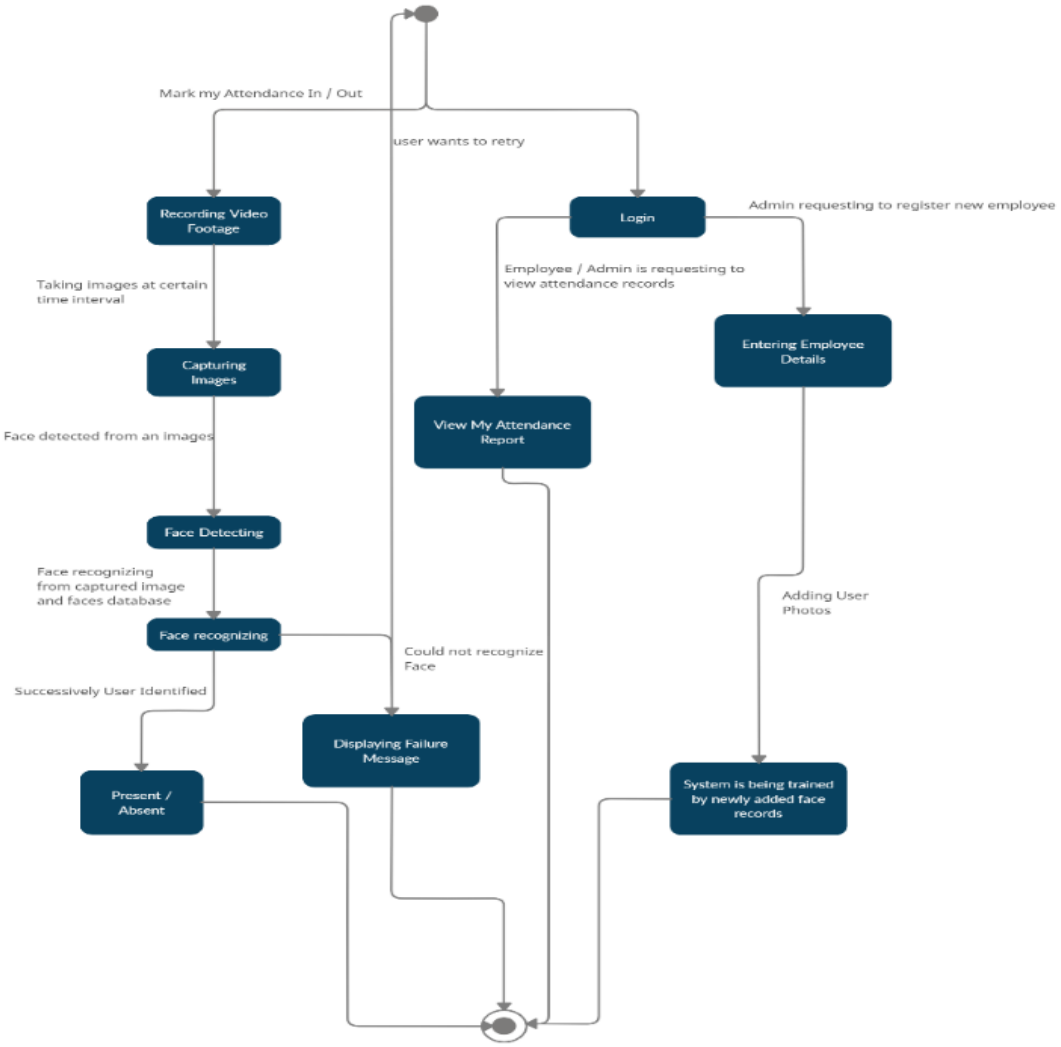


Figure 6.3 State diagram

6.4 Class Diagram:

The class diagram describes the structure of the system by presenting classes, attributes, operations, and relationships between objects. The class diagram is the main block of object-oriented modeling and is mainly used for general conceptual modeling of application structure and for detailed model translation modeling in coding. The class diagram can also be used for data modeling. In **figure 6.4** we have presented the class diagram in which we have managed to define four classes as well as the general one which is the system. Each class contains the name that characterizes it, each class has attributes as well as operations or functions. Most classes are interconnected with the system because there are functions that must be executed first by the system and then split between defined classes.

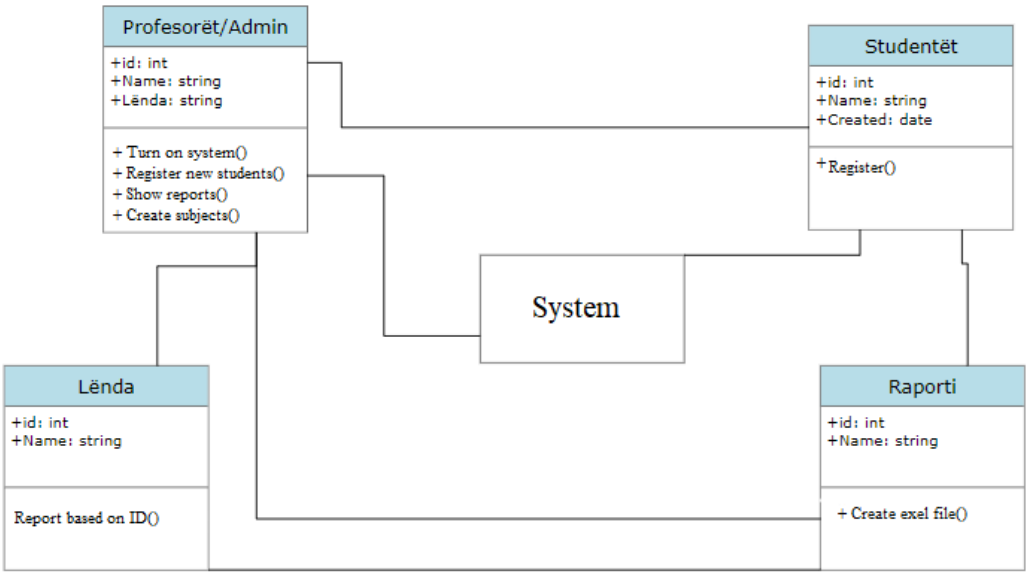


Figure 6.4 . Class diagram

6.5 Component Diagram:

The primary difference between a component diagram and other diagrams is that component diagrams represent the implementation perspective of a system. Therefore, the components in a component diagram reflect by grouping different designs of system elements, such as system classes. Firstly, the component must be replaceable, and secondly, the component must provide interfaces to enable other elements to interact and provide the services provided by the component. **Figure 6.5** shows the diagram which contains four components: connection, registration, presence marking, and the final report, where the diagram shows how the components interact with each other and their interaction with the database.

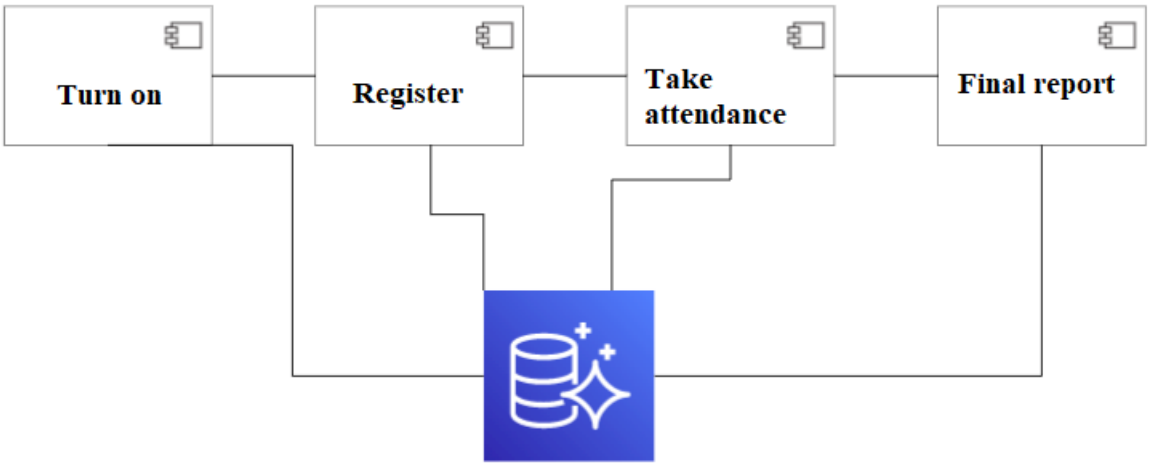


Figure 6.5 . Component diagram

6.6 Interaction Diagram:

The interaction diagram, otherwise known as the communication diagram, enables the illustration of connections and interactions between software objects. Otherwise, the interaction diagram represents the use case behavior describing how the set of objects interacts to complete the task. The types of interaction diagrams are sequential diagrams and cooperative diagrams. **Figure 6.6** shows the interaction diagram, in which I tried to show the communication between the components of the system together with the action sequences which start from the first sequence respectively from the "presence note". The ommunication diagram provides a clear illustration of the sequence of processes in sequential steps which define the functioning of the system.

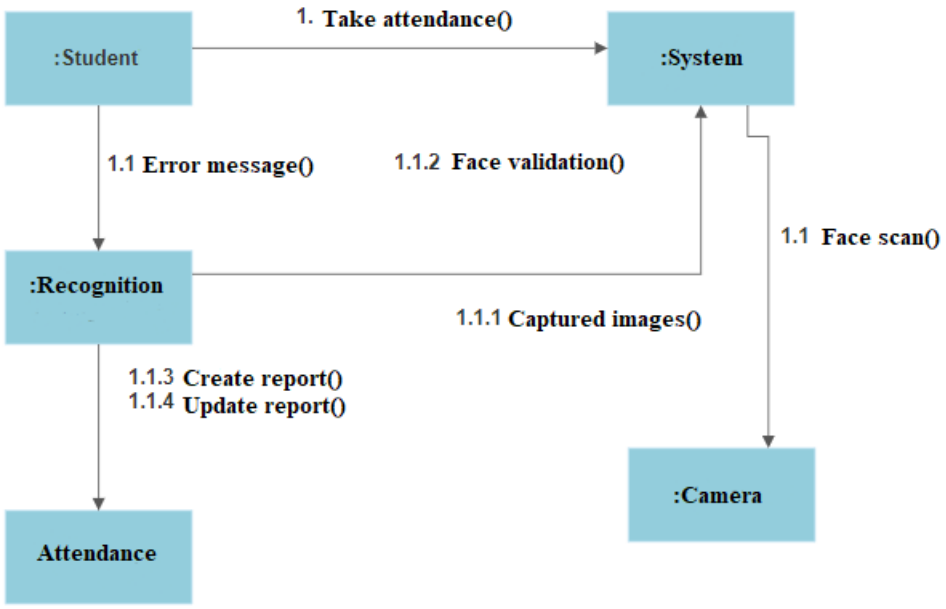


Figure 6.6. Interaction diagram

CHAPTER 7

SYSTEM TESTING

7.1 PERFORMANCE ANALYSIS

7.1.1 Digital Image Processing:

Digital Image Processing is the processing of images which are digital in nature by a digital computer. Digital image processing techniques are motivated by three major applications mainly:

- Improvement of pictorial information for human perception11
- Image processing for autonomous machine application
- Efficient storage and transmission.

7.1.2 Image Representation in a Digital Computer:

An image is a 2-Dimensional light intensity function

$$\mathbf{f}(\mathbf{x},\mathbf{y}) = \mathbf{r}(\mathbf{x},\mathbf{y}) \times \mathbf{i}(\mathbf{x},\mathbf{y}) \text{ -(2.0)}$$

Where, r (x, y) is the reflectivity of the surface of the corresponding image point. i (x,y) Represents the intensity of the incident light. A digital image f(x, y) is discretized both in spatial co-ordinates by grids and in brightness by quantization. Effectively, the image can be represented as a matrix whose row, column indices specify a point in the image and the element value identifies gray level value at that point. These elements are referred to as pixels or pels.

Typically following image processing applications, the image size which is used is$\mathbf{256 \times 256}$, elements, $\mathbf{640 \times 480}$ pels or $\mathbf{1024 \times 1024}$ pixels. Quantization of these matrix pixels is done at 8 bits for black and white images and 24 bits for coloured images (because of the three colour planes Red, Green and Blue each at 8 bits)

7.1.3 Steps in Digital Image Processing:

Digital image processing involves the following basic tasks:

- **Image Acquisition -** An imaging sensor and the capability to digitize the
- signal produced by the sensor.
- **Pre processing** – Enhances the image quality, filtering, contrast enhancement etc.
- **Segmentation** – Partitions an input image into constituent parts of Objects
- .**Description/feature Selection** – extracts the description of image objects suitable for further computer processing.
- **Recognition and Interpretation** – Assigning a label to the object based on the information provided by its descriptor. Interpretation assigns meaning to a set of labelled objects.
- **Knowledge Base** – This helps for efficient processing as well as inter module cooperation.

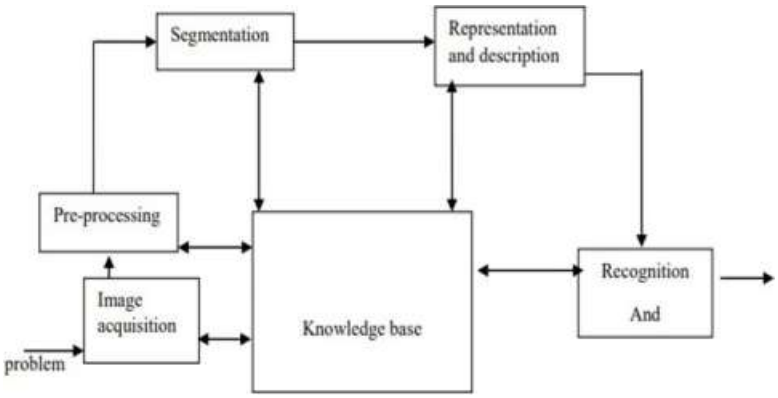


Figure 7.1 : A diagram showing the steps in digital image processing

7.2 Definition of Terms and History:

Face Detection

Face detection is the process of identifying and locating all the present faces in a single image or video regardless of their position, scale, orientation, age and expression. Furthermore, the detection should be irrespective of extraneous illumination conditions and the image and video content.

Face Recognition

Face Recognition is a visual pattern recognition problem, where the face, represented as a three dimensional object that is subject to varying illumination, 1213 pose and other factors, needs to be identified based on acquired images. Face Recognition is therefore simply the task of identifying an already detected face as a known or unknown face and in more advanced cases telling exactly whose face it Is.

Difference between Face Detection and Face Recognition :

Face detection answers the question, Where is the face? It identifies an object as a “face” and locates it in the input image. Face Recognition on the their hand answers the question who is this? Or whose face is it? It decides if the etected face is someone .It can therefore be seen that face detections output (the detected face) is the input to the face recognizer and the face Recognition’s output is the final decision i.e. face known or face unknown.

Haar Feature:

Viola-Jones algorithm which was introduced by P. Viola, M. J. Jones (2001) is the most popular algorithm to localize the face segment from static

images or video frame. Basically the concept of Viola-Jones algorithm consists of four parts. he first part is known as Haar feature, second part is where integral image is created, followed by implementation of Adaboost on the third part and lastly cascading process.

Figure 7.2 : Haar Feature

Viola-Jones algorithm analyses a given image using Haar features consisting of mutiple rectangles (Mekha Joseph et al., 2016).

In the fig shows several types of Haar features. The features perform as window function mapping onto the image. A single value result, which representing each feature can be computed by subtracting the sum of the white rectangle(s) from the sum of the black rectangle(s).

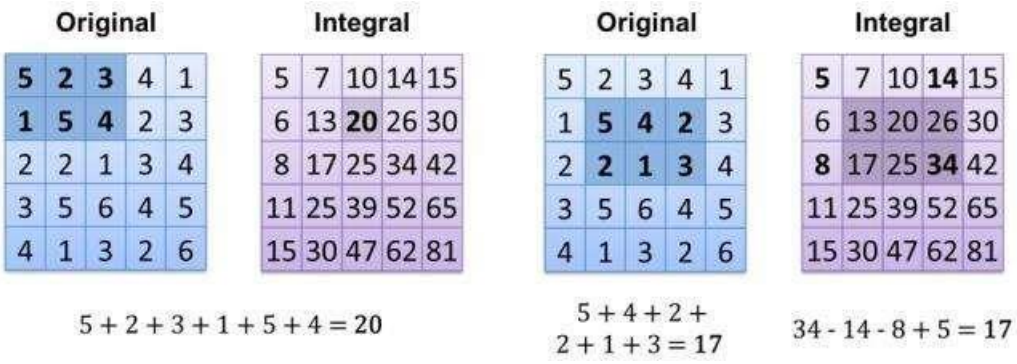


Figure 7.3 : Integral of Image

The value of integrating image in a specific location is the sum of pixels on the left and the top of the respective location. In order to illustrate clearly, the value of the integral image at location 1 is the sum of the pixels in rectangle A. The valuesof integral image at the rest of the locations are cumulative. For instance, the value at location 2 is summation of A and B, (A + B), at location 3 is summation of A and C, (A + C), and at location 4 is summation of all the regions, (A + B + C + D). Therefore, the sum within the D region can be computed with only addition and subtraction of diagonal at location 4 + 1 − (2 + 3) to eliminate rectangles A, B and C.

7.3 Local Binary Pattern Histogram

Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets. Using the LBP combined with histograms we can represent the face images with a simple data vector.

LBPH algorithm work step by step:

LBPH algorithm work in 5 steps.

1.**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.

● **Neighbours:** 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 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.

The image below shows this procedure:

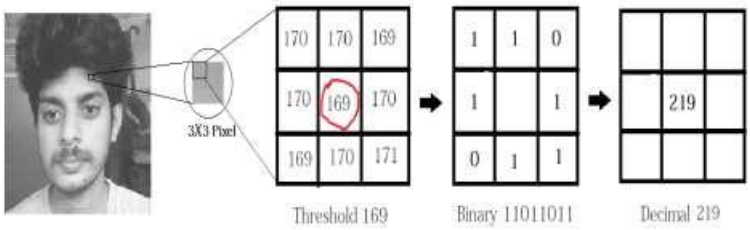


Figure 7.4: LBP Operation

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). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same.

● 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.

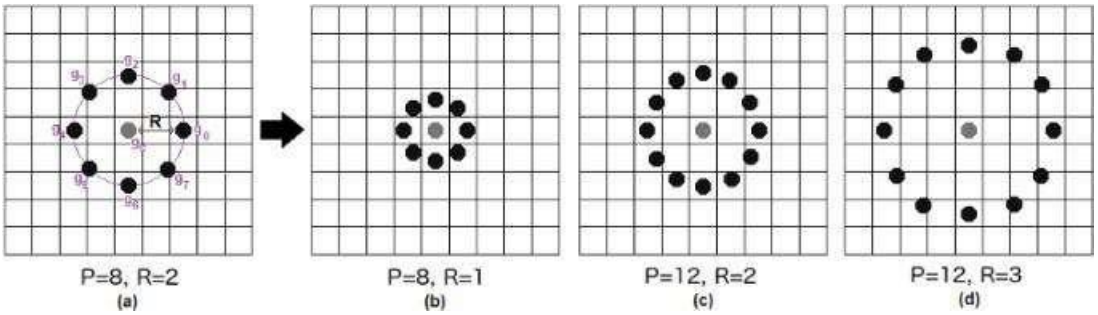


Figure 7.5: The LBP operation Radius Change

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 18value 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:

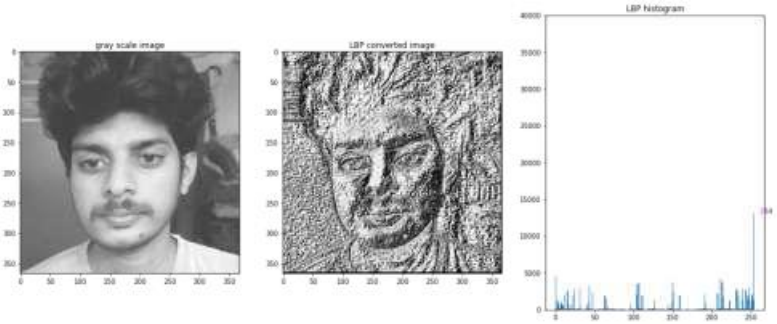


Figure 7.6: Extracting The Histogram

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 8x8x256=16.384 positions in the final histogram.

The final histogram represents the characteristics of the image original image.

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, etc.

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.

CHAPTER 8

SYSTEM ANALYSIS

8.1 Subsystem 2: Vision Acquisition

8.1.1 Objective

Verify that the camera can capture a live video feed and is able to convert image into grayscale.

8.1.2 Setup

We ran the main VI (Working Detection.vi) and checked whether the image of camera feed is being displayed in front panel or not.

8.1.3 Results

We were successfully able to capture live feed of video as shown in (Figure 8.1). Also, the image captured is in grayscale which passed our second test as well.

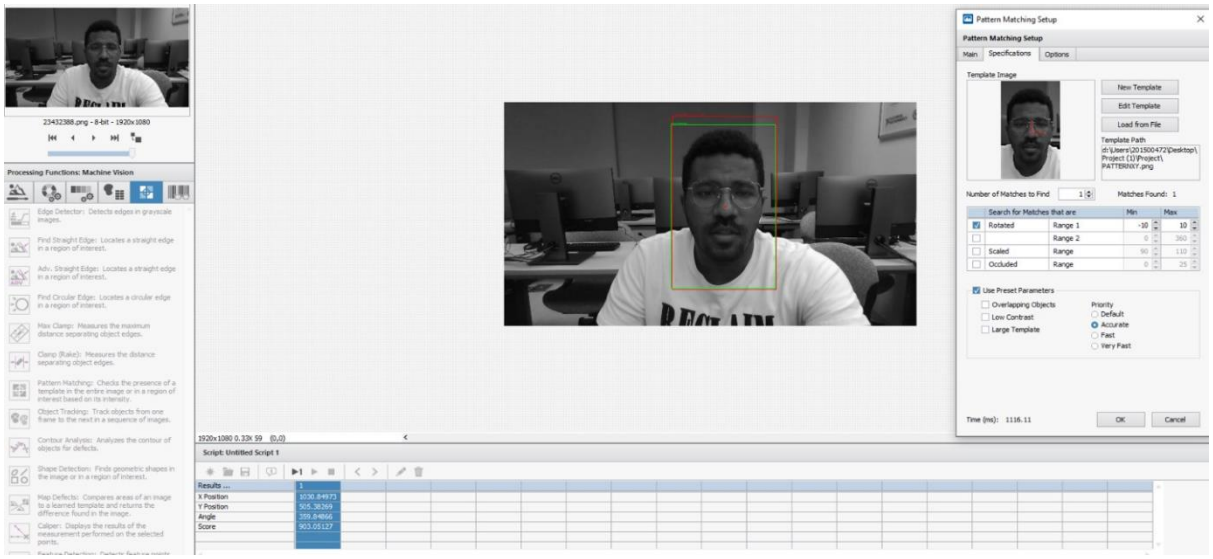


FIGURE 8.1: RESULTS OF FACE DETECTION

8.2 Subsystem 2: Vision Assistant

8.2.1 Objective

Verify that the image is captured from camera is processed and compared with saved templates for face recognition.

8.2.2 Setup

We ran the main VI (Working Detection.vi) in debugging mode to see whether the face was detected as well as to check number of matches ==1 for the student matched.

8.2.3 Results

We were successfully able to detect face as seen in value changes in debugging mode. The face corresponding to Yousef Abdullah caused number of matches to change to 1 which triggered the case structure to case = 1 and therefore passed the value further to Write to Spreadsheet module as shown in (Figure 8.1 and 8.2)

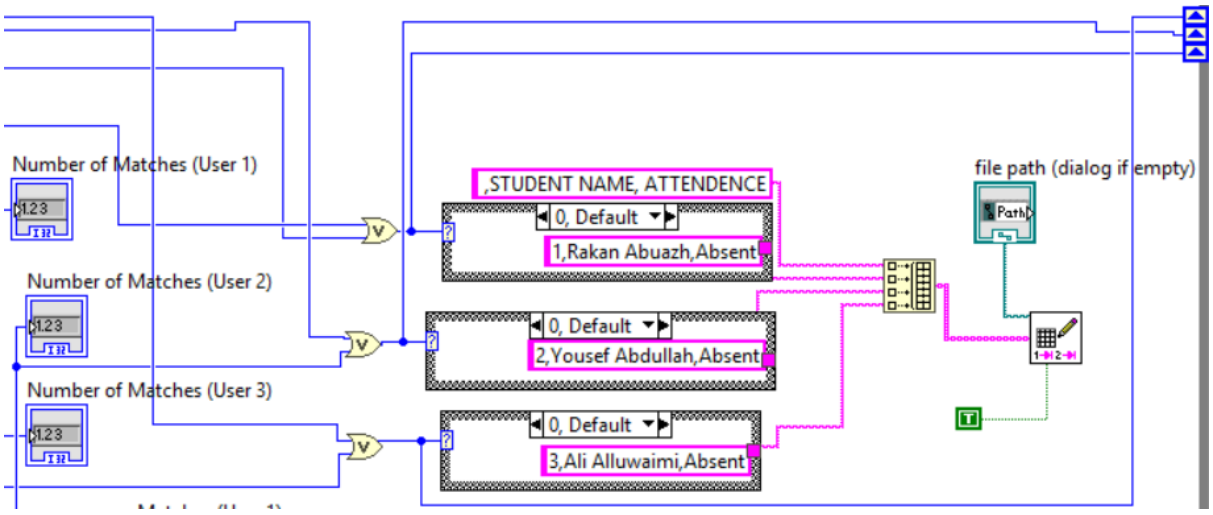


FIGURE 8.2: CONDITIONS OF ATTENDANCE

8.3 Overall Results, Analysis and Discussion

8.3.1 Objective

Verify that the spreadsheet is updated to corresponding present as soon as image is detected. The **"Present"** should be marked against the person detected only.

8.3.2 Setup

We ran the main VI (Working Detection.vi) and checked Attendance Sheet Section to change from **"Absent"** to "Present" on detection of person.

8.3.3 Results

We were successfully able to detect face and change the value of corresponding face from **Absent"** Before as shown in (Figure 8.3) to **"Present"** After as shown in (Figure 8.4).

BEFORE:

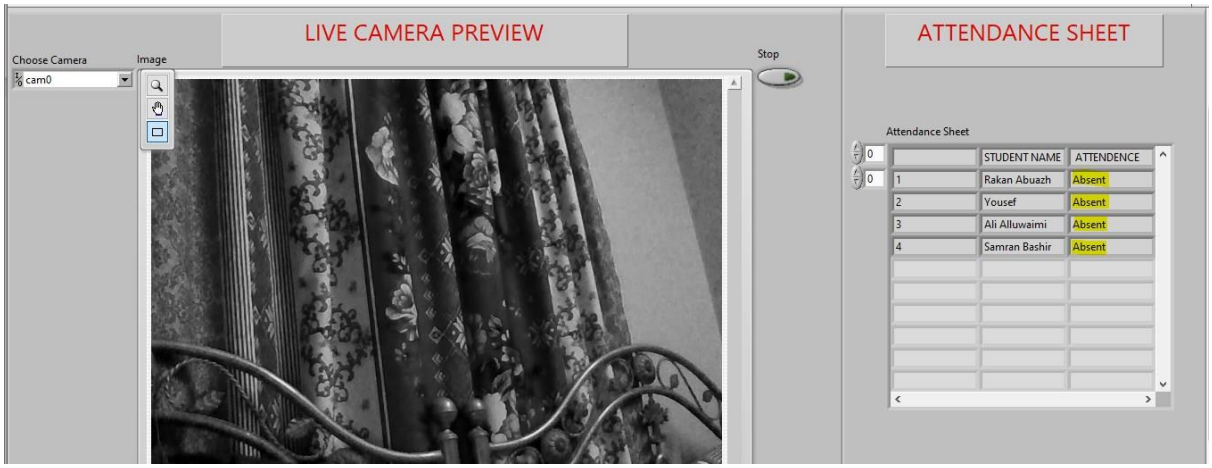


FIGURE 8.3: ATTENDANCE BEFORE FACE DETECTION

AFTER:

When we opened the attendance sheet on excel, we saw the file was updated correctly and Completely as shown in (Figure 8.4).

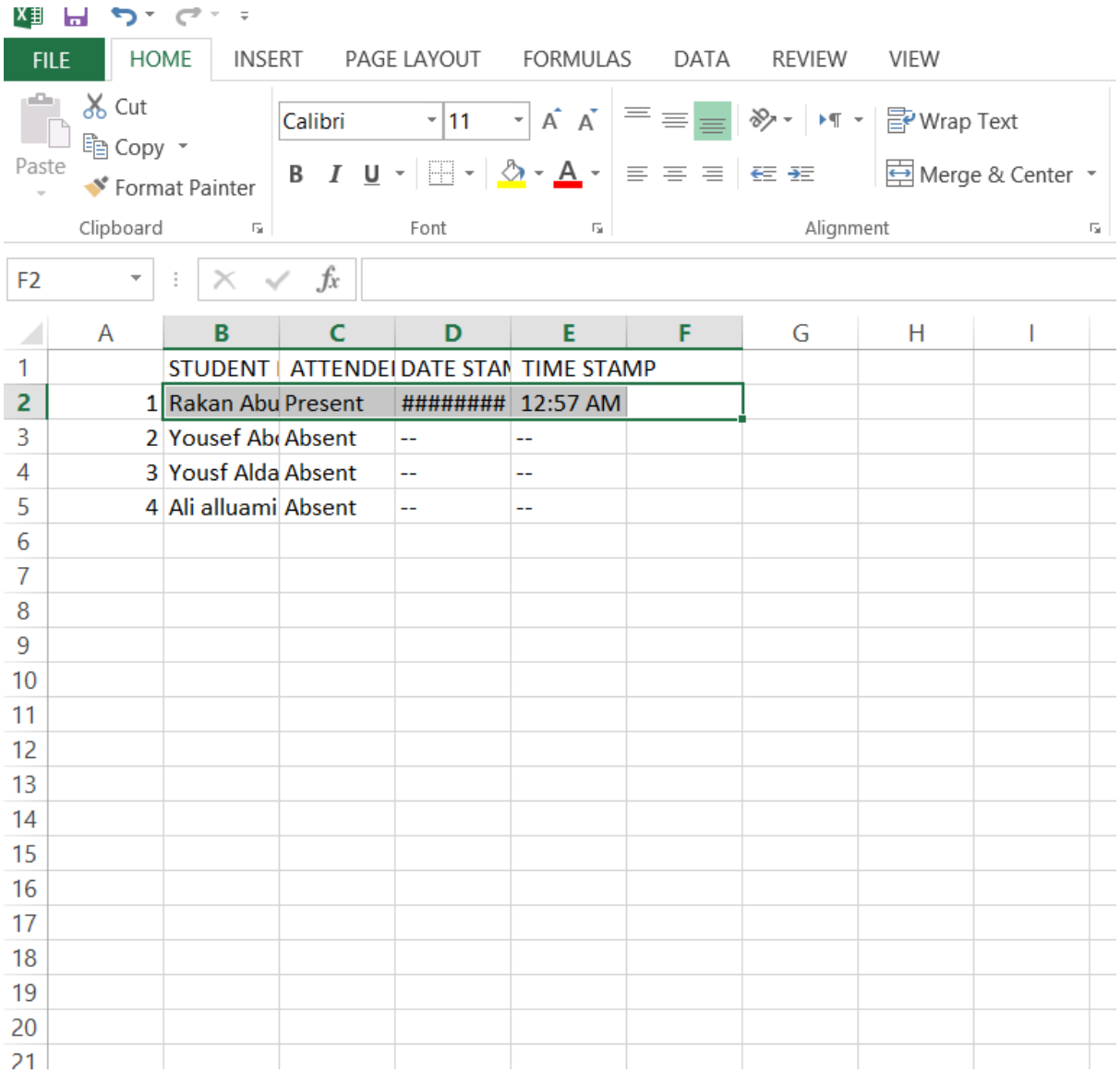


FIGURE8.4: ATTENDANCE AFTER FACE DETECTION

CHAPTER 9

MODULES

9.1 tkinter for whole GUI

Tkinter is Python's standard GUI package. It is an object-oriented layer on top of the open-source Tcl/Tk widget toolkit. While there are more feature complete packages available for creating a GUI, such as PyQt, Tkinter remains popular as it is simple, widely used by beginners and has a ton of references to get you started.

INSTALL TKINTER:

pip install tk

9.2 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

INSTALL OPENCV

Pip install opencv-python

9.3 CSV

A CSV (comma-separated values) file is a text file that has a specific format which allows data to be saved in a table structured format.

INSTALL CSV

Pip install python-csv

9.4 Numpy

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.

INSTALL NUMPY

Pip install numpy

9.5 DATETIME

Datetime in Python is the combination between dates and times. The attributes of this class are similar to both date and separate classes. These attributes include day, month, year, minute, second, microsecond, hour, and tzinfo.

INSTALL DATETIME

Pip install DateTime

9.6 PANDAS

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

INSTALL PANDAS

Pip install pandas

CHAPTER 10

CONCLUSION AND FUTURE ENHANCEMENTS

10.1 CONCLUSION

Face recognition systems are part of facial image processing applications and their significance as a research area are increasing recently. Implementations of system are crime prevention, video surveillance, person verification, and similar security activities. The face recognition system implementation can be part of Universities. Face Recognition Based Attendance System has been envisioned for the purpose of reducing the errors that occur in the traditional (manual) attendance taking system. The aim is to automate and make a system that is useful to the organization such as an institute. The efficient and accurate method of attendance in the office environment that can replace the old manual methods.This method is secure enough, reliable and available for use. Proposed algorithm is capable of detect multiple faces, and performance of system has acceptable good results.

10.2 Future Recommendations

The system can be made more flexible and scalable using these recommendations. Please note that the system implemented here is just a prototype of idea presented via this project. The recommendations are as follows:

- The system can be extended to more number of students with freedom to change list of
students according to class changes.
- The system can be made more flexible to allow updating of templates in case student incurs
significant amount of change in his facial features.

- The system can also be extended to allow better face recognition algorithm in which even rotational features of face can be detected efficiently.

In this paper, we have elaborated, researched, and developed a system for registration of presence, through which lecturers or professors can record student participation in electronic form. Using the software system saves time and effort, especially for large groups of students. The automated presence registration system is built to reduce the shortcomings and problems that appear in the traditional (manual) system. Thus, the purpose of the system and this paper is to build and clarify the system which enables the recognition of the facial expressions of the enrolled students, and the same preserves it and creates reports efficiently and with a high accuracy which the system achieves using image training. After extensive research into the Python programming language and libraries needed to build such systems, I began developing the system by performing all the steps required by software engineering for the system to be fully functional and effective. The completion of the project resulted in a software product that can automate the registration process of participation in the university facility. The system requires minimal human intervention, even only during the start-up and initial enrollment of students and then the process is completely automatic, saving the time consumed by the traditional system. In addition, the development of graphical interfaces makes the system very easy to use and understandable for all users who need to interact with it.

APPENDIX 1

SAMPLE CODE

HOME.HTML

```
<!doctype html>
<html lang="en">

<style type='text/css'>
    * {
        padding: 0;
        margin: 0;
        font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    }

    body {
        background-image: url('https://cutewallpaper.org/21/1920-x-1080-
gif/1920x1080-Wallpapercartoon-Wallpapers-Driverlayer-Search-.gif');
        background-size: cover;
        font-family: sans-serif;
        margin-top: 40px;
        height: 100vh;
        padding: 0;
        margin: 0;
    }

    table {
        border: 1px;
        font-family: arial, sans-serif;
        border-collapse: collapse;
        width: 86%;
        margin: auto;
    }

    td,
    th {
        border: 1px solid black !important;
        padding: 5px;
    }

    tr:nth-child(even) {
        background-color: #dddddd;
    }
```

```
</style>

<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet"
href="https://fonts.googleapis.com/icon?family=Material+Icons">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-
eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzKgwr
a6" crossorigin="anonymous">

    <title>Face Recognition Based Attendance System</title>
</head>

<body>

    <div class='mt-3 text-center'>
        <h1 style="width: auto;margin: auto;color: white;padding: 11px;font-size:
44px;">Face Recognition Based
            Attendance System</h1>
    </div>

    {% if mess%}
    <p class="text-center" style="color: red;font-size: 20px;">{{ mess }}</p>
    {% endif %}

    <div class="row text-center" style="padding: 20px;margin: 20px;">

        <div class="col"
            style="border-radius: 20px;padding: 0px;background-
color:rgb(211,211,211,0.5);margin:0px 10px 10px 10px;min-height: 400px;">
            <h2 style="border-radius: 20px 20px 0px 0px;background-color:
#0b4c61;color: white;padding: 10px;">Today's
                Attendance <i class="material-icons">assignment</i></h2>
            <a style="text-decoration: none;max-width: 300px;" href="/start">
                <button
                    style="font-size: 24px;font-weight: bold;border-radius:
10px;width:490px;padding: 10px;margin-top: 30px;margin-bottom: 30px;"
```

```
                type='submit' class='btn btn-primary'>Take Attendance <i
                    class="material-icons">beenhere</i></button>
            </a>
            <table style="background-color: white;">
                <tr>
                    <td><b>S No</b></td>
                    <td><b>Name</b></td>
                    <td><b>ID</b></td>
                    <td><b>Time</b></td>
                </tr>
                {% if l %}

                {% for i in range(l) %}
                <tr>
                    <td>{{ i+1 }}</td>
                    <td>{{ names[i] }}</td>
                    <td>{{ rolls[i] }}</td>
                    <td>{{ times[i] }}</td>
                </tr>
                {% endfor %}
                {% endif %}
            </table>

        </div>

        <div class="col"
            style="border-radius: 20px;padding: 0px;background-
color:rgb(211,211,211,0.5);margin:0px 10px 10px 10px;height: 400px;">
            <form action='/add' method="POST" enctype="multipart/form-data">
                <h2 style="border-radius: 20px 20px 0px 0px;background-color:
#0b4c61;color: white;padding: 10px;">Add
                    New User <i class="material-
icons">control_point_duplicate</i></h2>
                <label style="font-size: 20px;"><b>Enter New User
Name*</b></label>
                <br>
                <input type="text" id="newusername" name='newusername'
                    style="font-size: 20px;margin-top:10px;margin-bottom:10px;"
required>
                <br>
                <label style="font-size: 20px;"><b>Enter New User Id*</b></label>
                <br>
                <input type="number" id="newusereid" name='newuserid'
```

```
                style="font-size: 20px;margin-top:10px;margin-bottom:10px;"
required>
            <br>
            <button style="width: 232px;margin-top: 20px;font-size: 20px;"
type='submit' class='btn btn-dark'>Add
                New User
            </button>
            <br>
            <h5 style="padding: 25px;"><i>Total Users in Database:
{{totalreg}}</i></h5>
        </form>
    </div>

  </div>

</body>

</html>
```

APP.PY

```
import cv2
import os
from flask import Flask,request,render_template
from datetime import date
from datetime import datetime
import numpy as np
from sklearn.neighbors import KNeighborsClassifier
import pandas as pd
import joblib

#### Defining Flask App
app = Flask(_name_)

#### Saving Date today in 2 different formats
def datetoday():
    return date.today().strftime("%m_%d_%y")
def datetoday2():
    return date.today().strftime("%d-%B-%Y")

#### Initializing VideoCapture object to access WebCam
face_detector =
cv2.CascadeClassifier('static/haarcascade_frontalface_default.xml')
cap = cv2.VideoCapture(0)

#### If these directories don't exist, create them
if not os.path.isdir('Attendance'):
    os.makedirs('Attendance')
if not os.path.isdir('static/faces'):
    os.makedirs('static/faces')
if f'Attendance-{datetoday()}.csv' not in os.listdir('Attendance'):
    with open(f'Attendance/Attendance-{datetoday()}.csv','w') as f:
        f.write('Name,Roll,Time')

#### get a number of total registered users
def totalreg():
    return len(os.listdir('static/faces'))

#### extract the face from an image
def extract_faces(img):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    face_points = face_detector.detectMultiScale(gray, 1.3, 5)
```

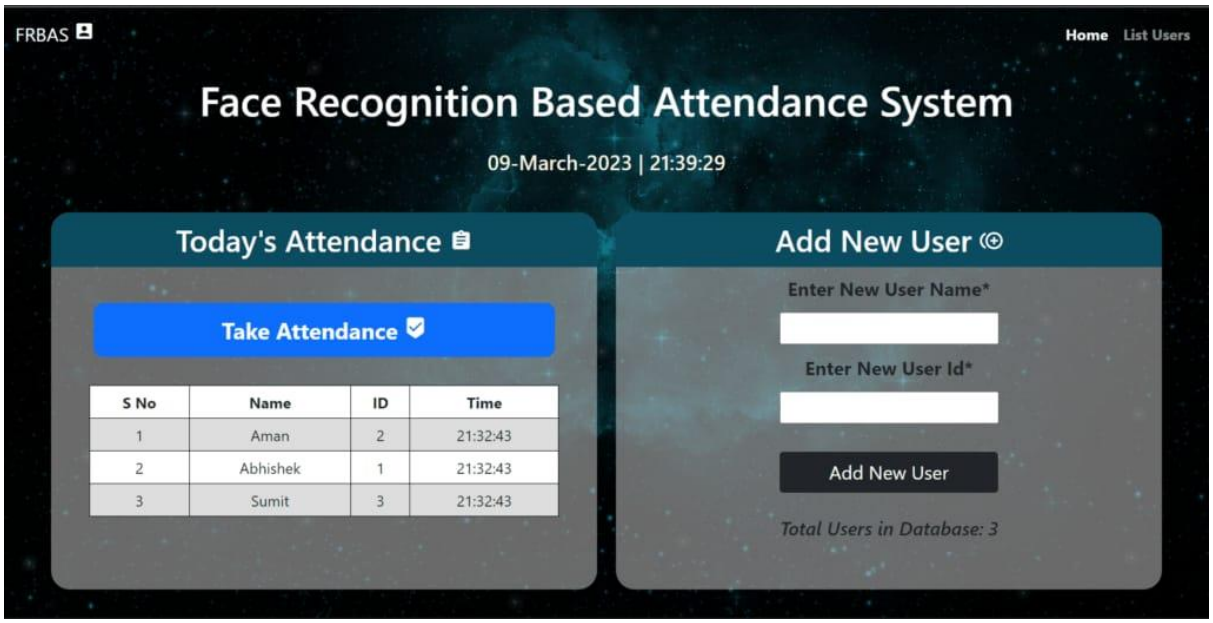
```
    return face_points

#### Identify face using ML model
def identify_face(facearray):
    model = joblib.load('static/face_recognition_model.pkl')
    return model.predict(facearray)

#### A function which trains the model on all the faces available in faces folder
def train_model():
    faces = []
    labels = []
    userlist = os.listdir('static/faces')
    for user in userlist:
        for imgname in os.listdir(f'static/faces/{user}'):
            img = cv2.imread(f'static/faces/{user}/{imgname}')
            resized_face = cv2.resize(img, (50, 50))
            faces.append(resized_face.ravel())
            labels.append(user)
    faces = np.array(faces)
    knn = KNeighborsClassifier(n_neighbors=5)
    knn.fit(faces,labels)
    joblib.dump(knn,'static/face_recognition_model.pkl')

#### Extract info from today's attendance file in attendance folder
def extract_attendance():
    df = pd.read_csv(f'Attendance/Attendance-{datetoday()}.csv')
    names = df['Name']
    rolls = df['Roll']
    times = df['Time']
    l = len(df)
    return names,rolls,times,l

#### Add Attendance of a specific user
def add_attendance(name):
    username = name.split('_')[0]
    userid = name.split('_')[1]
    current_time = datetime.now().strftime("%H:%M:%S")

    df = pd.read_csv(f'Attendance/Attendance-{datetoday()}.csv')
    if int(userid) not in list(df['Roll']):
        with open(f'Attendance/Attendance-{datetoday()}.csv','a') as f:
            f.write(f'\n{username},{userid},{current_time}')
```

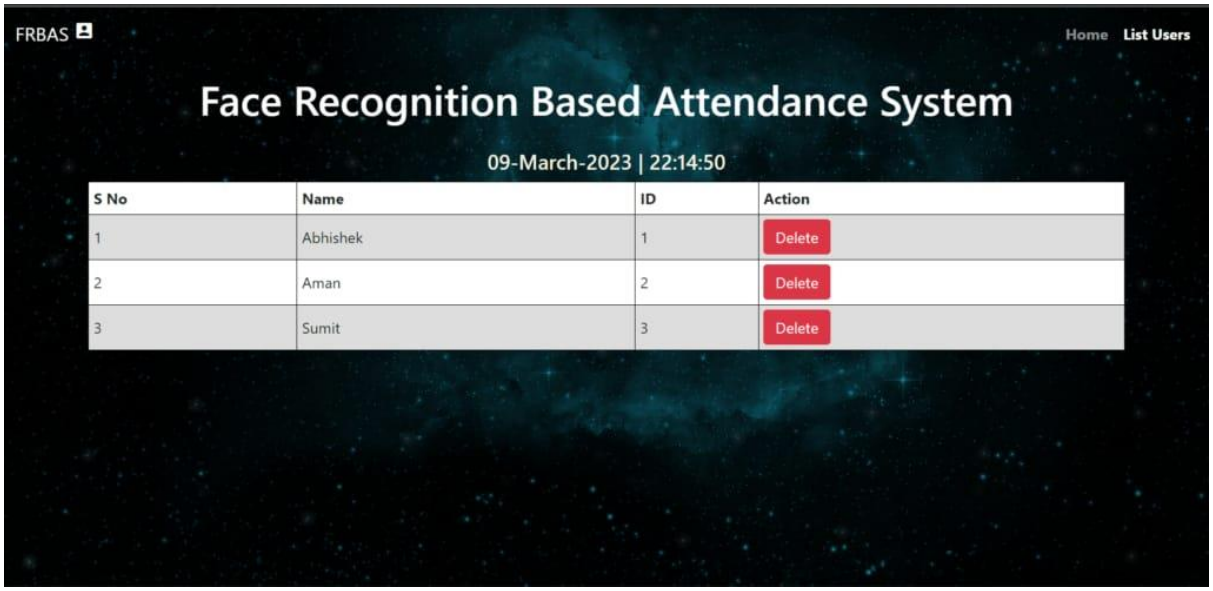
```
################## ROUTING FUNCTIONS
#########################

#### Our main page
@app.route('/')
def home():
    names,rolls,times,l = extract_attendance()
    return
render_template('home.html',names=names,rolls=rolls,times=times,l=l,totalreg=
totalreg(),datetoday2=datetoday2())

#### This function will run when we click on Take Attendance Button
@app.route('/start',methods=['GET'])
def start():
    if 'face_recognition_model.pkl' not in os.listdir('static'):
        return
render_template('home.html',totalreg=totalreg(),datetoday2=datetoday2(),mess=
'There is no trained model in the static folder. Please add a new face to
continue.')

    cap = cv2.VideoCapture(0)
    ret = True
    while ret:
        ret,frame = cap.read()
        if extract_faces(frame)!=():
            (x,y,w,h) = extract_faces(frame)[0]
            cv2.rectangle(frame,(x, y), (x+w, y+h), (255, 0, 20), 2)
            face = cv2.resize(frame[y:y+h,x:x+w], (50, 50))
            identified_person = identify_face(face.reshape(1,-1))[0]
            add_attendance(identified_person)

cv2.putText(frame,f'{identified_person}',(30,30),cv2.FONT_HERSHEY_SIMP
LEX,1,(255, 0, 20),2,cv2.LINE_AA)
        cv2.imshow('Attendance',frame)
        if cv2.waitKey(1)==27:
            break
    cap.release()
    cv2.destroyAllWindows()
    names,rolls,times,l = extract_attendance()
    return
render_template('home.html',names=names,rolls=rolls,times=times,l=l,totalreg=
totalreg(),datetoday2=datetoday2())
```

```
#### This function will run when we add a new user
@app.route('/add',methods=['GET','POST'])
def add():
    newusername = request.form['newusername']
    newuserid = request.form['newuserid']
    userimagefolder = 'static/faces/'+newusername+'_'+str(newuserid)
    if not os.path.isdir(userimagefolder):
        os.makedirs(userimagefolder)
    cap = cv2.VideoCapture(0)
    i,j = 0,0
    while 1:
        _,frame = cap.read()
        faces = extract_faces(frame)
        for (x,y,w,h) in faces:
            cv2.rectangle(frame,(x, y), (x+w, y+h), (255, 0, 20), 2)
            cv2.putText(frame,f'Images Captured:
{i}/50',(30,30),cv2.FONT_HERSHEY_SIMPLEX,1,(255, 0,
20),2,cv2.LINE_AA)
            if j%10==0:
                name = newusername+'_'+str(i)+'.jpg'
                cv2.imwrite(userimagefolder+'/'+name,frame[y:y+h,x:x+w])
                i+=1
            j+=1
        if j==500:
            break
        cv2.imshow('Adding new User',frame)
        if cv2.waitKey(1)==27:
            break
    cap.release()
    cv2.destroyAllWindows()
    print('Training Model')
    train_model()
    names,rolls,times,l = extract_attendance()
    return
render_template('home.html',names=names,rolls=rolls,times=times,l=l,totalreg=
totalreg(),datetoday2=datetoday2())

#### Our main function which runs the Flask App
if _name_ == '_main_':
    app.run(debug=True)
```

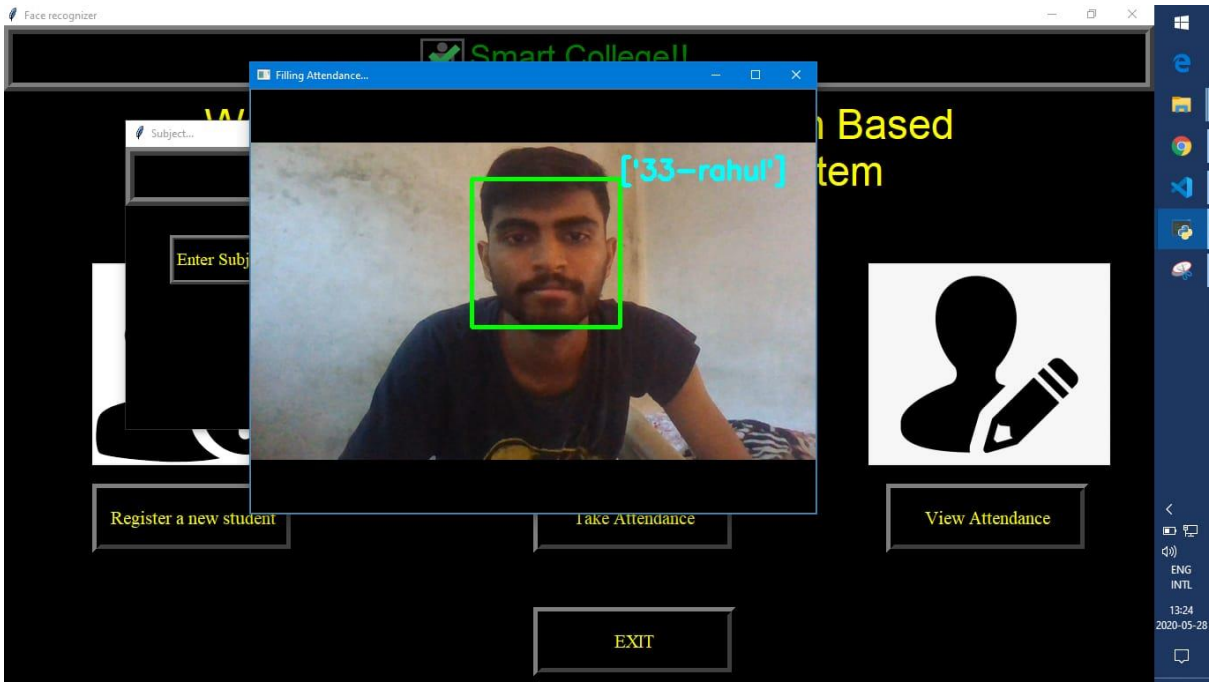
APPENDIX 2

OUTPUT SCREEN SHOT





	A	B	C	D
1	Name	Roll	Time	
2	Aman	2	21:32:43	
3	Abhishek	1	21:32:43	
4	Sumit	3	21:32:43	
5				
6				



REFERENCES

[1]. *A brief history of Facial Recognition*, NEC, New Zealand,26 May 2020.[Online]. Available: https://www.nec.co.nz/market-leadership/publications-media/a-brief-history-of-facial recognition/

[2]. *Face detection,TechTarget Network*, Corinne Bernstein, Feb, 2020.[Online]. Available: https://searchenterpriseai.techtarget.com/definition/face-detection

[3]. Paul Viola and Michael Jones, *Rapid Object Detection using a Boosted Cascade of Simple Features.* Accepted Conference on Computer Vision and Pattern Re cognition, 2001.

[4]. *Face Detection with Haar Cascade*,Towards Data Science-727f68dafd08,Girija Shankar Behera, India, Dec 24, 2020.[Online]. Available:https://towardsdatascience.com/face-detection with-haar-cascade-727f68dafd08

[5]. *Face Recognition: Understanding LBPH Algorithm,*Towards Data Science-90ec258c3d6b,Kelvin Salton do Prado, Nov 11, 2017.[Online]. Available :https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b

[6]. *What is Facial Recognition and how sinister is it*, Theguardian, IanSample, July, 2019. [Online]. Available: https://www.theguardian.com/technology/2019/jul/29/what-is-facial recognition-and-how-sinister-is-it

[7].Kushsairy Kadir , Mohd Khairi Kamaruddin, Haidawati Nasir, Sairul I Safie, Zulkifli Abdul Kadir Bakti,"A comparative study between LBP and Haar-like features for Face Detection using OpenCV*", 4th International Conference on Engineering Technology and Technopreneuship (ICE2T)*, DOI:10.1109/ICE2T.2014.7006273, 12 January 2015. O

[8].Senthamizh Selvi.R,D.Sivakumar, Sandhya.J.S , Siva Sowmiya.S, Ramya.S , Kanaga Suba Raja.S,"Face Recognition Using Haar - Cascade Classifier for Criminal Identification", *International Journal of Recent Technology and Engineering(IJRTE)*, vol.7, issn:2277-3878, , issue-6S5, April 2019.

[9]. Robinson-Riegler, G., & Robinson-Riegler, B. (2008). Cognitive psychology: applying the 64 science of the mind. Boston, *Pearson/Allyn and Bacon*..

[10]. Margaret Rouse, *What is facial recognition? - Definition from WhatIs.com*, 2012. [online] Available at: http://whatis.techtarget.com/definition/facial-recognition

[11]. Robert Silk, *Biometrics: Facial recognition tech coming to an airport near you: Travel Weekly*, 2017. [online] Available at: https://www.travelweekly.com/Travel-News/AirlineNews/Biometrics-Facial-recognition-tech-coming-airport-near-you

[12]. Sidney Fussell, *NEWS Facebook's New Face Recognition Features: What We Do (and Don't) Know*, 2018. [online] Available at: https://gizmodo.com/facebooks-new-face-recognition-fea tures-what-we-do-an-1823359911

[13]. Reichert, C. *Intel demos 5G facial-recognition payment technology | ZDNet*, 2017. [online] ZDNet. Available at: https://www.zdnet.com/article/intel-demos-5g-facial-recognition-payment

technology/#:~:text=Such%20%22pay%20via%20face%20identification,and%20artificial%20int elligence%20(AI). [Accessed 25 Mar. 2018].

[14]. Mayank Kumar Rusia, Dushyant Kumar Singh, Mohd. Aquib Ansari, “Human Face Identification using LBP and Haar-like Features for Real Time Attendance Monitoring”, *2019 Fifth International Conference on Image Information Processing (ICIIP)* ,Shimla, India, DOI: 10.1109/ICIIP47207.2019.8985867 10 February 2020.

[15]. V. a. R. Tokas, "Fast Face Recognition Using Eigen Faces," IJRITCC, vol. 2, no. 11, pp. 3615-3618, November 2014.