FACE RECOGNITION BASED ATTENDANCE SYSTEM

A Project Report

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA

by

D. Bala Subrahmanyam	**(19X41A0413)**
G. Gayathri	**(20X45A0404)**
N. Mukunda Priya	**(19X41A0433)**
P. Sri Lakshmi Nadh	**(19X41A0441)**

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Under the Guidance of

Mr. B. Ravi

Assistant Professor

SRK INSTITUTE OF TECHNOLOGY,Enikepadu
Andhra Pradesh,Vijayawada-521108

April

2023

CERTIFICATE

This is to certify that the work done on the project report entitled "**FACE RECOGNITION BASED ATTENDANCE SYSTEM"** is a bonafied record of work carriedout by D. Bala Subrahmanyam (19X41A0413), G. Gayathri (20X45A0404) ,N. Mukunda Priya (19X41A0433), P.Sri Lakshmi Nadh (19X41A0441) in partial fulfilment of the requirements of the Degreeof Bachelor of Technology in Electronics and Communication Engineering, Jawaharlal Nehru Technological University, Kakinada.

Mr.B.Ravi	Dr.S.Sri Gowri
(PROJECT GUIDE)	(HEAD OF THE DEPARTMENT)

ACKNOWLEDGEMENTS

We are glad to express our deep sense of gratitude to **Mr. B. RAVI**, Professor, Electronics and Communication Engineering for his valuable guidance, motivation, constant inspiration and above all for his ever-co-operating attitude that enabled us in bringing this thesis in present form.

We express our heartful gratitude and deep indebtedness to our beloved Head of the Department, Electronics and Communication Engineering **Dr. S.SRI GOWRI**, for her great help and encouragement in doing our project successfully.

We also express our gratitude to our principal **Dr.M.EKAMBARAM NAIDU**, for his encouragement and facilities provided during the course of project.

We thank one and all who have rendered help to us directly or indirectly in the completion of this work.

D.BALA SUBRAHMANYAM (19X41A0413)
G. GAYATHRI (20X45A0404)
N. MUKUNDA PRIYA (19X41A0433)
P.SRI LAKSHMI NADH (19X41A0441)

CONTENTS

LIST OF FIGURES

ABSTRACT

The main purpose of this project is to build a face recognition-based attendance monitoring system for educational institution to enhance and upgrade the current attendance system into more efficient and effective as compared to before. The current old system has a lot of ambiguity that caused inaccurate and inefficient of attendance taking. Many problems arise when the authority is unable to enforce the regulation that exist in the old system. Thus, by means of technology, this project will resolve the flaws existed in the current system while bringing attendance taking to a whole new level by automating most of the tasks. The technology working behind will be the face recognition system. The human face is one of the natural traits that can uniquely identify an individual. Therefore, it is used to trace identity as the possibilities for a face to deviate or being duplicated is low. In this project, facedatabases will be created to pump data into the recognizer algorithm. Then, during the attendance taking session, faces will be compared against the database to seek for identity. When an individual is identified, its attendance will be taken down automatically saving necessary information into a database system. At the end of the day, the attendance informationregarding an individual can be accessed from a web server hosted by the raspberry pi .In short, this upgraded version of attendance monitoring system not only saved many resources, but also provide huge convenience to the authority as many process are automated.

Chapter 1
INTRODUCTION

A facial recognition attendance system uses facial recognition technology to identify and verify a person using the person's facial features and automatically mark attendance. The software can be used for different groups of people such as employees, students, etc. The system records and stores the data in real-time. Facial recognition is the most feasible option available in the current situation for organizations to make employee attendance and visitor entry contactless.

1.1 Objective

The proposed system will reduce the paper work where attendance will no longer involve any manual recording. The new system will also reduce thetotal time needed to do attendance recording. The new system will acquire individual attendance by means of facial-recognition to secure data accuracy of the attendance.

The followings are the objectives of this project:

- To ensure the speed of the attendance recording process is faster than the previoussystem which can go as fast as approximately 3 second for each student.
- Have sufficient memory space to store the database.
- Able to recognize the face of an individual accurately based on the face database.
- Develop a database for the attendance management system.
- Provide a user-friendly web interface for admins to access the attendance database andfor non-admins (parents) to check their child’s attendance.
- Allow new members to store their faces in the database .

1.2 Project Scope and Direction

The main intention of this project is to solve the issues encountered in the old attendance system while reproducing a brand new innovative smart system that can provide convenienceto the institution. In this project, a smart device will be developed which is capable of recognising the identity of each individuals and eventually record down the data into a database system. Apart from that, a website will be developed to provide visual access to the information.

The followings are the project scopes:

- The targeted groups of the attendance monitoring system are the students and staff ofan educational institution.
- The database of the attendance management system can hold above 500 individual'sinformation.
- The facial recognition process can only be done for 1 person at a time.
- The project has to work under a Wi-Fi coveraged area, as the system need to updatethe database of the attendance system constantly.
- The smart device is powered up by power bank to improve the portability of thedevice.

1.3 Impact ,Significance and Contributions

Many attendance management systems that exist nowadays are lack of efficiency and information sharing. Therefore, in this project, those limitations will be overcome and alsofurther improved.

The impact and the contribution of this project is as follow:

- Students will be more punctual on attending classes. This is due to the attendance of aparticular student can only be taken personally where any absentees will be noticed bythe system. This can not only train the student to be punctual as well as avoids any immoral ethics such as signing the attendance for their friends.
- The institution can save a lot of resources as enforcement are now done by means oftechnology rather than human supervision which will waste a lot of human resourcesfor an insignificant process.
- The smart device can operate at any location as long as there is Wi-Fi coverage whichmakes the attendance system to be portable to be placed at any intended location. For an example, the device can be placed at the entrance of the classroom to take the attendance.
- It saves a lot of cost in the sense that it had eliminated the paper work completely.
- The system is also time effective because all calculations are all automated.In short, the project is developed to solve the existing issues in the old attendance system.

Chapter 2
HISTORY OF IOT

The term ‘Internet of Things’ was coined in 1999 by the computer scientist Kevin Ashton. While working at Procter & Gamble, Ashton proposed putting radio-frequency identification (RFID) chips on products to track them through a supply chain. He reportedly worked the then-buzzword ‘internet’ into his proposal to get the executives’ attention. And the phrase stuck. Over the next decade, public interest in IoT technology began to take off, as more and more connected devices came to market.In 2000, LG announced the first smart refrigerator, in 2007 the first iPhone was launched and by 2008, the number of connected devices exceeded the number of people on the planet.In 2009, Google started testing driverless cars and in 2011, Google’s Nest smart thermostat hit the market, which allowed remote control of central heating.

2.1 Definition of IoT

The Internet of Things (IoT) describes the network of physical objects“things"that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. These devices range from ordinary household objects to sophisticated industrial tools. With more than 7 billion connected IoT devices today, experts are expecting this number to grow to 10 billion by 2020 and 22 billion by 2025. Oracle has a network of device partners.

2.2 Why We Use IoT?

Over the past few years, IoT has become one of the most important technologies of the 21st century. Now that we can connect everyday objects—kitchen appliances, cars, thermostats, baby monitors—to the internet via embedded devices, seamless communication is possible between people, processes, and things.By means of low-cost computing, the cloud, big data, analytics, and mobile technologies, physical things can share and collect data with minimal human intervention. In this hyperconnected world, digital systems can record, monitor, and adjust each

interaction between connected things. The physical world meets the digital world—and they cooperate

2.3 Features of IoT

Connectivity: In the case of IoT, the most important feature one can consider is connectivity. Without seamless communication among the interrelated components of the IoT ecosystems (i.e sensors, compute engines, data hubs, etc.) it is not possible to execute any proper business use case. IoT devices can be connected over Radio waves, Bluetooth, Wi-Fi, Li-Fi, etc. We can leverage various protocols of internet connectivity layers in order to maximize efficiency and establish generic connectivity across IoT ecosystems and Industry. There may be special cases where the IoT ecosystem is built on-premises or in an intranet.

Sensing: We humans can naturally understand and analyze our circumstances easily based on our past experiences with various things or situations. In the case of IoT in order to get the best of it, we need to read the analog signal, convert it in such a way that we can derive meaningful insights out of it. We use Electrochemical, gyroscope, pressure, light sensors, GPS, Electrochemical, pressure, RFID, etc. to gather data based on a particular problem. For example for automotive use cases, we use Light detection sensors along with pressure, velocity and imagery sensors. To make a use case successful we need to choose the proper sensing paradigm.

Active Engagements: IoT device connects various products, cross-platform technologies and services work together by establishing an active engagement between them. In general, we use cloud computing in blockchain to establish active engagements among IoT components. In the case of Industry grade, IoT solutions raw analog data need to be acquired, preprocessed and rescale as per business capacity. As per Google, only 50% of structured and 1% of unstructured data is used to make important business decisions. So while designing the IoT ecosystems carriers need to consider the future needs of manipulating such a huge scale of data to satisfy incremental business needs. One can confuse the need of active engagements with scale, practically it means your systems should be able to handle huge data across various technologies, platforms, products, and industries.

Scale: IoT devices should be designed in such a way that they can be scaled up or down easily on demand. In general, IoT is being used from smart home automation to automating large factories and work stations, so the use cases vary in scale. A carrier should design their IoT infrastructure depending upon their current and future engagement scale.

Dynamic Nature: For any IoT use case, the first and foremost step is to collecting and converting data in such a way that means business decisions can be made out of it. In this whole process, various components of IoT need to change their state dynamically. For example, the input of a temperature sensor will vary continuously based on weather conditions, locations, etc. IoT devices should be designed this keeping in mind**.**

Intelligence: In almost every IoT use cases in today's world, the data is used to make important business insights and drive important business decisions. We develop machine learning/ deep learning models on top of this massive data to obtain valuable insights. The analog signals are preprocessed and converted to a format on which machine-learning models are trained. We need to keep in mind the proper data infrastructure based on business needs.

Energy: From end components to connectivity and analytics layers, the whole ecosystems demand a lot of energy. While designing an IoT ecosystem, we need to consider design methodology such that energy consumption is minimal.

Safety: One of the main features of the IoT ecosystem is security. In the whole flow of an IoT ecosystem, sensitive information is passed from endpoints to the analytics layer via connectivity components. While designing an IoT system we need to adhere to proper safety, security measures, and firewalls to keep the data away from misuse and manipulations. Compromising any component of an IoT ecosystem can eventually lead to failure of the whole pipeline.

Integration: IoT integrates various cross-domain models to enrich user experience. It also ensures proper trade-off between infrastructure and operational costs.

2.4 Future of IoT

The IoT creates smart environments with digital technologies to optimise the way we live our lives. The rollout of over 41 billion IoT devices is expected by 2025 (International Data Corporation). This will lead to an exponential growth of data and push computing operations and data analytics to the edge.

Edge computing builds on a multi-layer technology that enables the management, and automation of connected IoT devices. It is the logical evolution of the dominant cloud computing model, avoiding the transfer of mission-critical data to the cloud, supporting resilience, real-time operations, and security, privacy and protection. At the same time, it reduces energy consumption and our carbon footprint. In edge computing, the processing moves from a centralised point, closer to (or even into) the IoT device itself: the 'edge' or periphery of a network.

The research, innovation and deployment of the next generation of IoT will need a strong computing capacity at the edge in order to support Europe’s digital autonomy in future ICT systems. This will be achieved by creating a computing continuum – from swarms of far edge devices to the cloud – and involves smart platform building driven by key European players.

Chapter 3

RASPBERRY PI

Raspberry Pi is defined as a minicomputer with the size of a credit card that is interoperablewith any input and output hardware device like a monitor, a television, a mouse, or akeyboard – effectively converting the set-up into a full-fledged PC at a low cost.

3.1 Introduction to Raspberry Pi

The Raspberry Pi is equipped with a quad-core 64-bit Broadcom BCM2837 ARM Cortex-A53 SoC processor running at 1.2 GHz**.** Which means the new Raspberry Pi 3 can be used for office applications and web browsing.

The great innovation in this third version is undoubtedly the addition of a **WI-FI** chip and **Bluetooth** Low Energy. This not only saves space (you no longer need to connect WIFI and Bluetooth dongles), but also frees up more USB ports for connecting otherdevices.

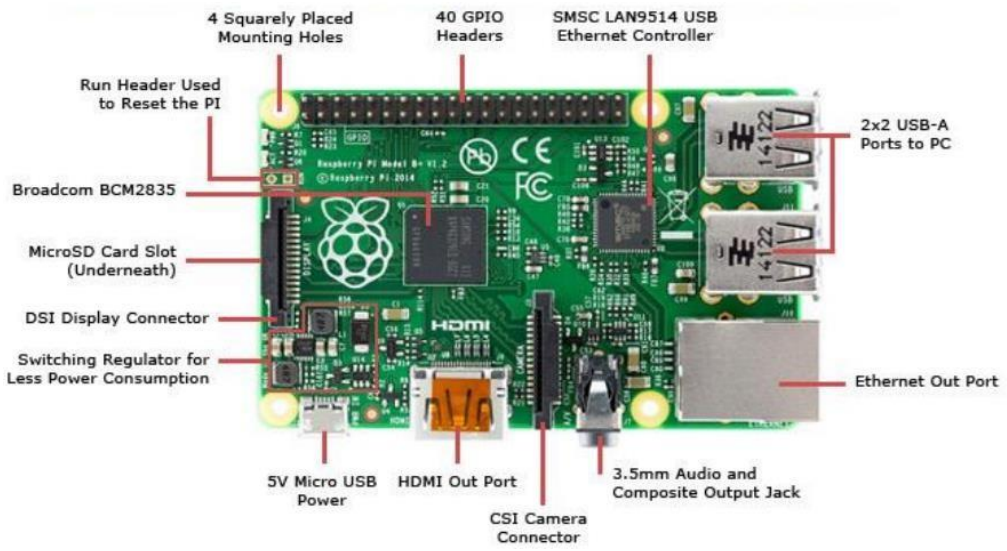

Figure 3.1 : Raspberry Pi Specifications

3.2 Pins of Raspberry Pi

Most models of the Raspberry Pi have a 40-pin header, as shown in the image above. Of the 40 pins, 26 are GPIO pins and the others are power or ground pins (plus two ID EEPROM pins, which you should not play with unless you know your stuff!). Any of the GPIO pins can be designated (in software) as an input or output pin and used for a wide range of purposes; whether it is turning on an LED, driving a motor, or sending data to another device, the possibilities are almost endless.

Early models of the Raspberry Pi A and B have a shorter header with 26 pins, as shown below.

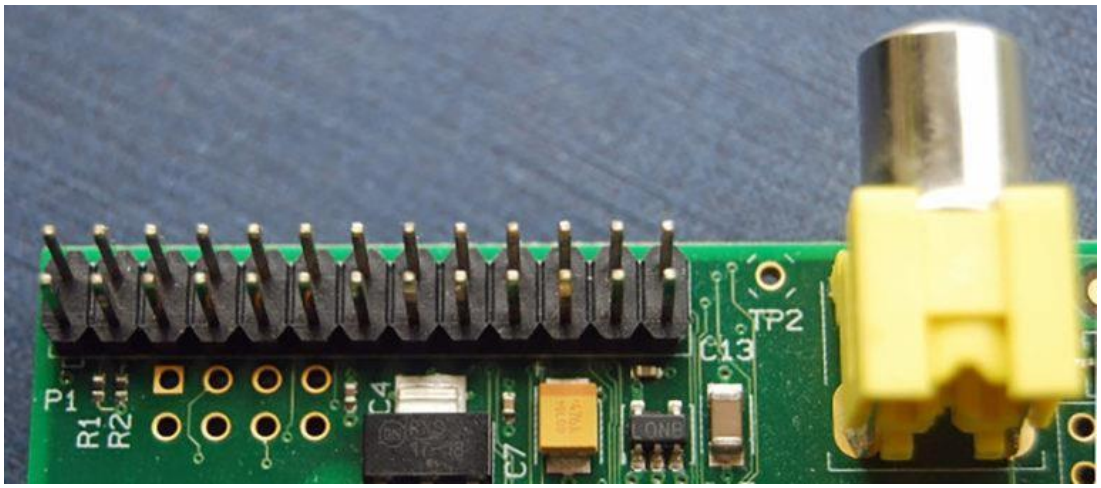

Figure 3.2 : GPIO Pins of Raspberry Pi

The Raspberry Pi Zero models have unpopulated pins (apart from the Raspberry Pi Zero WH) so there are holes where the GPIO header is located instead of physical pins. This means you need to add a header that includes the pins yourself.

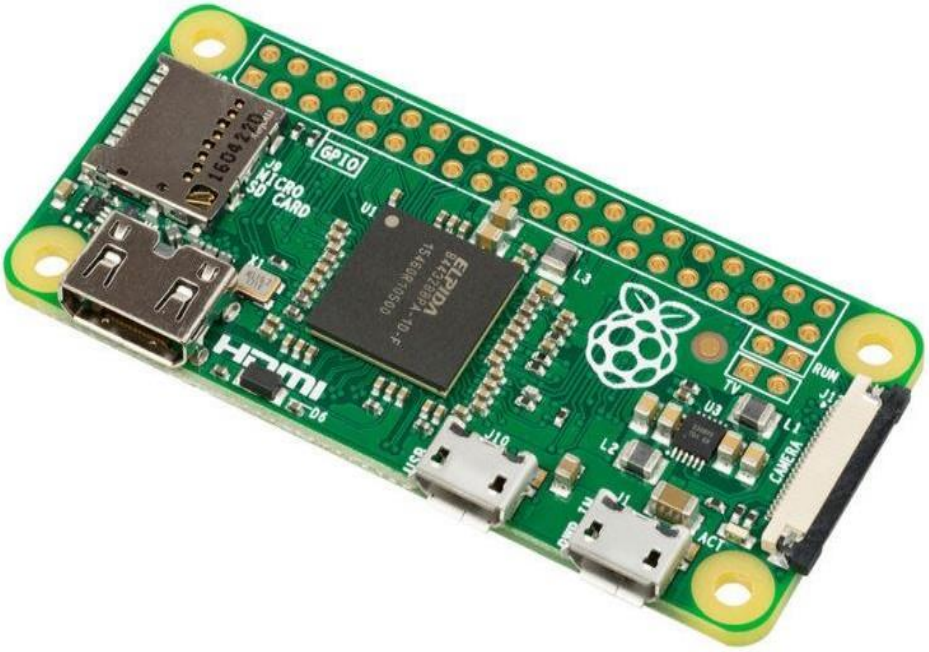

Figure 3.3 : Raspberry pi Layout

Although it is possible to create a robot buggy with most models of Raspberry Pi, I recommend using a Raspberry Pi 3B, 3B+, or 4. These models allow you to program the Raspberry Pi easily and connect it to another computer or even a smartphone by using the inbuilt WiFi or Bluetooth, rather than needing to plug the Pi physically into a screen or a keyboard and mouse.

GPIO pin numbering: When programming the GPIO pins, there are two different ways to refer to them: **GPIO numbering** and **physical numbering**. Throughout this course (and in all our resources) we will refer to the pins using the GPIO numbering scheme. Theseare the GPIO pins as the computer sees them.

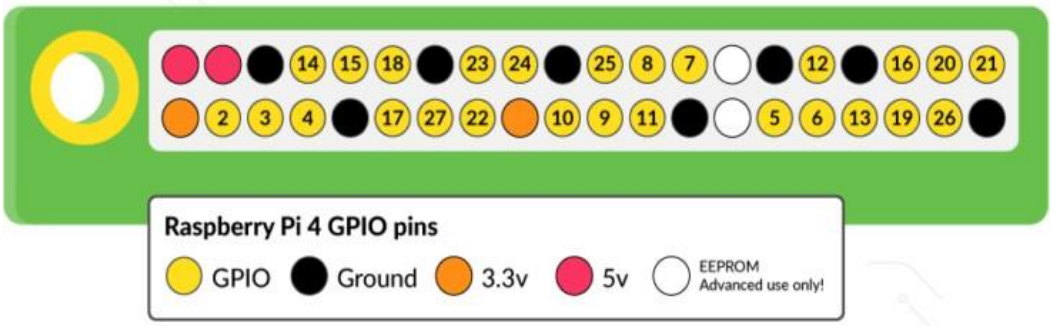

Figure 3.4 : GPIO Pins Layout

The numbering of the GPIO pins is not in numerical order, instead relating to the numbering on the CPU of the Raspberry Pi, so there is no easy way to remember them. However, you can use a reference board that fits over the pins, a printed reference (like the image above), or a website guide to the GPIO pins to help you.

Voltages: The voltage of a pin is labelled on the reference guide. There are two **5V** pins and two **3V** pins, as well as a number of ground pins (0V), which are unconfigurable. Theremaining pins are all general-purpose 3V3 pins, meaning that the outputs are set to

3.3 volts and the inputs are tolerant of 3.3 volts.

A GPIO pin designated as an **output** pin can be set to high (3.3V) or low (0V). Components are usually attached so that setting the output to high will allow current to flow to them, while setting the output to low won't.

A GPIO pin that is designated as an **input** will allow a signal to be received by the Raspberry Pi. The threshold between a high and a low signal is around 1.8V. A voltage between 1.8V and 3.3V will be read by the Raspberry Pi as high; anything lower than

1.8V will be read as low. Do not allow an input voltage above 3.3V, or else you will fry your Pi.

3.3 Pin Functionality

GPIO: GPIO pins are standard general-purpose pins that can be used for turning external devices, such as an LED, on or off.

Power: 5V and 3V3 pins are used to supply 5V and 3.3V power to external components.

I2C: I2C pins are used for connecting and hardware communication purposes with I2C compatible external modules.

SPI: SPI (Serial Peripheral Interface Bus) pins are also used for hardware communication, but with a different protocol.

UART: UART (Universal Asynchronous Receiver / Transmitter) pins are used for serial communication.

DNC: Use of DNC (Do Not Connect) pins should be avoided.

GND: GND (Ground) pins refer to pins that provide electrical grounding in your circuits.

3.4 Raspberry Pi Pin Configuration

Raspberry Pi is a series of small single-board computers developed in the UK by the Raspberry Pi Foundation with the aim of promoting teaching of basic computer science in schools and in developing countries. Today, Raspberry Pi is commonly used in robotics. In this blog, we shall take a look at the pin configuration of a Raspberry Pi 3 board.

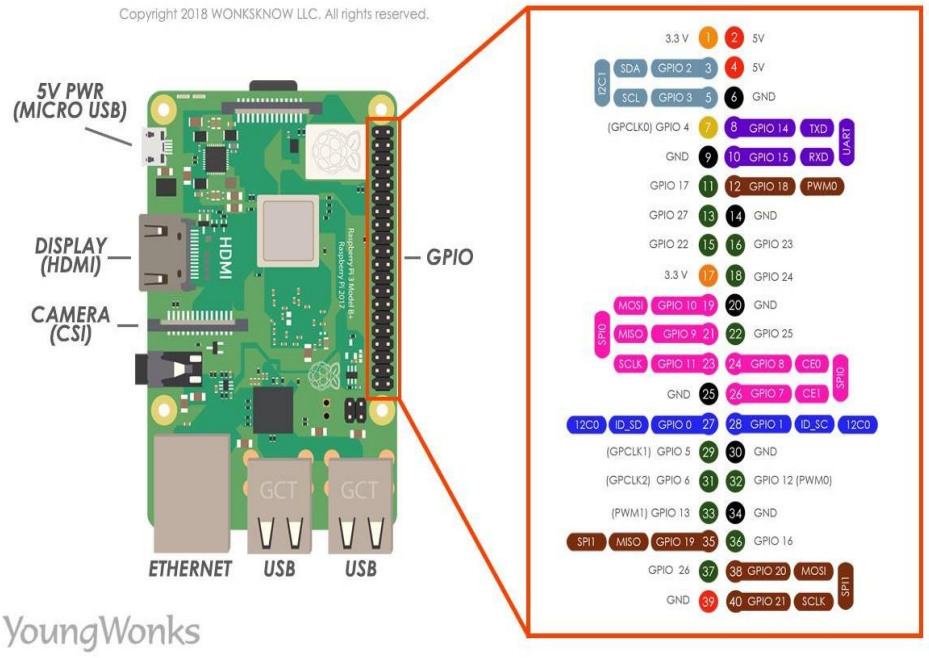

Figure 3.5 : Raspberry Pi Pin Configuration

A Raspberry Pi 3 board has 40 pins on it. Among these pins, we have four power pins on the Raspberry Pi, two of which are 5v pins and another two are 3.3v pins. The 5v power pins are connected directly to the Raspberry Pi's power input and we can use these pins to run low power applications.

Then there are the ground pins. There are eight ground pins and all of these are connected to each other; you can use any of these ground pins for your projects.
That leaves us with 28 GPIO pins, labeled starting from GPIO 0 and going up to GPIO 27.
The GPIO pins, as indicated by their full form, can be programmed to be output pins or input pins. So we can set values of output pins and we can even read values of input pins. The GPIO pins can be digitally programmed so that they can be turned ON or OFF. The output of any GPIO pin is 3.3v and can be used to control output components like an LED or a motor. These ON/OFF conditions can also be interpreted as a Boolean True/False, 1/0 or HIGH/LOW.
These are the common types of pins on a Raspberry Pi 3 board. Some of these pins also have a dual function. For example, pin 3 or GPIO 2 also acts like an I2C pin.

3.5 Applications of Raspberry Pi

Desktop PC

Wireless print server

Media Usage

Game Servers

Retro Gaming Machine

Robot Controller

Stop Motion Camera

Time-lapse

Chapter 4

SETTING UP THE RASPBERRY PI FOR USE

Virtual Network Computing (VNC) is a graphical desktop-sharing system that uses the Remote Frame Buffer protocol (RFB) to remotely control another computer. It transmits the keyboard and mouse input from one computer to another, relaying the graphical-screen updates, over a network.

4.1 Connecting to Raspberry Pi remotely

Before the raspberry pi can be accessed from a remote device through the connection of wifi, initial setup has to be done for future convenience. Therefore, when the desktop first appeared after the initialization process mentioned in the previous section, the Wi-Fi is turned on to obtain wifi connection from the hotspot provided by a mobile phone. Then, a remote access software such as *VNC Viewer* is installed onto the laptop. Upon connecting the raspberry pi to the remote viewer software (*VNC Viewer*) on the laptop, the raspberry pi has to be in the same network as the laptop, then the connection progress can be done by just entering the IP address of the raspberry pi into the *VNC Viewer*.

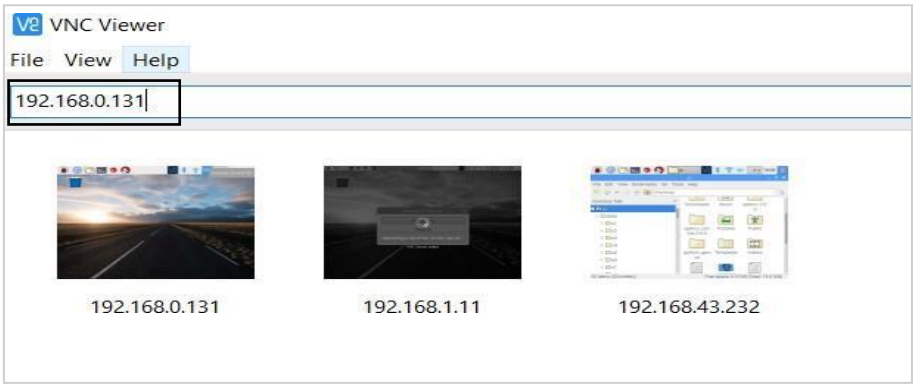

Figure 4.1 : Installing OpenCV into the Raspberry Pi

In this project, OpenCV is used to do facial recognition where the whole program will be coded in Python Language. The installation of OpenCV is merely not enough, therefore, OpenCV is installed with Python bindings to bridge Python and C++ in OpenCV. In short, the binding is crucial to enable the calling of a C++ function from Python. The following is a brief walkthrough on the installation process that had done.

4.2 Setting up the Pi Camera Module

After installing the Pi Camera Module onto the raspberry pi board, the next step is to enable the camera module. Before being able to do that, the raspberry pi firmware is being updated first. Then, the camera module is being enabled by going into the configuration menuat the terminal. Next, the raspberry pi is rebooted. In order for python to interact with the pi camera, a *pi camera* module with NumPy array support is installed. This is due to OpenCV takes images as NumPy arrays.

Chapter 5

OPERATING SYSTEM OF RASPBERRY PI

Raspberry Pi OS (formerly Raspbian) is a Unix-like operating system based on the Debian Linux distribution for the Raspberry Pi family of compact single-board computers. First developed independently in 2012, it has been produced as the primary operating system for these boards since 2013, distributed by the Raspberry Pi Foundation.

5.1 Raspbian

Raspberry Pi OS (formerly Raspbian) is a Unix-like operating system based on the Debian Linux distribution for the Raspberry Pi family of compact single-board computers. First developed independently in 2012, it has been produced as the primary operating system for these boards since 2013, distributed by the Raspberry Pi Foundation.

Raspberry Pi OS is highly optimized for the Raspberry Pi with ARM CPUs. It runs on every Raspberry Pi except the Pico microcontroller. Raspberry Pi OS uses a modified LXDE desktop environment with the Openbox stacking window manager, along with a unique theme. The default distribution is shipped with a copy of the computer algebra system Wolfram Mathematica,[4] VLC, and a lightweight version of the Chromium web browser.

5.2 History of Raspbian

Raspberry Pi OS was first developed by Mike Thompson and Peter Green as Raspbian, an independent and unofficial port of Debian to the Raspberry Pi. The first build was released on July 15, 2012. As the Raspberry Pi had no officially provided operating system at the time, the Raspberry Pi Foundation built on the work by the Raspbian project and began producing and releasing their own version of the software. The Foundation's first release of Raspbian, which now referred both to the community project as well as the official operating system, was announced on September 10, 2013.

On May 28, 2020, the Raspberry Pi Foundation announced a beta 64-bit version.

When the Foundation did not want to use the name Raspbian to refer to software that was not based on the Raspbian project, the name of the officially provided operating system was changed to Raspberry Pi OS. This change was also carried over to the 32-bit version, though it continued to be based on Raspbian. The 64-bit version of Raspberry Pi OS was officially released on February 2, 2022.

5.3 Features of Raspbian

Raspberry Pi OS has a desktop environment, PIXEL (short for Pi Improved Xwindows Environment, Lightweight), based on LXDE, which looks similar to many common desktops, such as macOS and Microsoft Windows. The desktop has a background image. A menu bar is positioned at the top and contains an application menu and shortcuts to a web browser (Chromium), file manager, and terminal. The other end of the menu bar shows a Bluetooth menu, Wi-Fi menu, volume control, and clock. The desktop can also be changed from its default appearance, such as repositioning the menu bar.

Chapter 6
PROGRAMMING LANGUAGE

Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming.

6.1 Python

Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. It supports multiple programming paradigms beyond object-oriented programming, such as procedural and functional programming. Python combines remarkable power with very clear syntax. It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible. It is also usable as an extension language for applications that need a programmable interface. Finally, Python is portable: it runs on many Unix variants including Linux and macOS, and on Windows.

6.2 History

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's "benevolent dictator for life", a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a five-member Steering Council to lead the project.

Python 2.0 was released on 16 October 2000, with many major new features

and Unicode support. Python 3.0, released on 3 December 2008, with many of its major features backported to Python 2.6.x and 2.7.x. Releases of Python 3 include the `2to3` utility, which automates the translation of Python 2 code to Python 3.

Python 2.7's end-of-life was initially set for 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. No further security patches or other improvements will be released for it. Currently only 3.7 and later are supported. In 2021, Python 3.9.2 and 3.8.8 were expedited as all versions of Python had security issues leading to possible remote code execution and web cache poisoning.

In 2022, Python 3.10.4 and 3.9.12 were expedited and 3.8.13, and 3.7.13, because of many security issues. When Python 3.9.13 was released in May 2022, it was announced that the 3.9 series (joining the older series 3.8 and 3.7) would only receive security fixes in the future. On September 7, 2022, four new releases were made due to a potential denial-of-service attack: 3.10.7, 3.9.14, 3.8.14, and 3.7.14.

As of November 2022, Python 3.11.0 is the current stable release. Notable changes from 3.10 include increased program execution speed and improved error reporting.

6.3 Indentation

Python uses whitespace indentation, rather than curly brackets or keywords, to delimit blocks. An increase in indentation comes after certain statements; a decrease in indentation signifies the end of the current block. Thus, the program's visual structure accurately represents its semantic structure. This feature is sometimes termed the off-side rule. Some other languages use indentation this way; but in most, indentation has no semantic meaning. The recommended indent size is four spaces.

6.4 Modules used in Program

Tkinter:Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interfaceto the Tk GUI toolkit, and is Python's de facto standard GUI.

Cv2 : OpenCV (Open Source Computer Vision Library) is an open source computer vision and python 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. Being an Apache 2 licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

Shutil: Shutil module offers high-level operation on a file like a copy, create, and remote operation on the file. It comes under Python's standard utility modules. This module helps in automating the process of copying and removal of files and directories.

Csv:A comma-separated values (CSV) file is a delimited text file that uses a comma to separate values. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format.

Numpy: NumPy is a Python library used for working with arrays. It also has functions forworking in domain of linear algebra, fourier transform, and matrices.

Pil:PIL stands for Python Imaging Library, and it's the original library that enabled Python to deal with images. It is the defacto image processing package for Python language. It is used for editing, creating and saving images .

Pandas :It is used for data security .It eases data analysis, data manipulation and cleaning of data. pandas support operations like sorting, re-indexing, iteration ,concatenation conversion of data, visualizations ,aggregations .

Datetime: The datetime module supplies classes for manipulating dates and times. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation. General calendar related functions.

Time : In Python, the time() function returns the number of seconds passed since epoch thepoint where time begins.

Urllib Request: The urllib.request module defines functions and classes which help in opening URLs(mostly HTTP) in a complex world basic and digest authentication, redirections, cookies and more. The Requests package is recommended for a higher-level HTTP client interface.

Chapter 7

HARDWARE AND SOFTWARE IMPLEMENTATION

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 individual's basic information which is their ID and their faces. The first procedure of portrait Acquisation can be done by using the Raspberry Pi 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 by using Hardware And Software Implementation.

7.1 Block Diagram

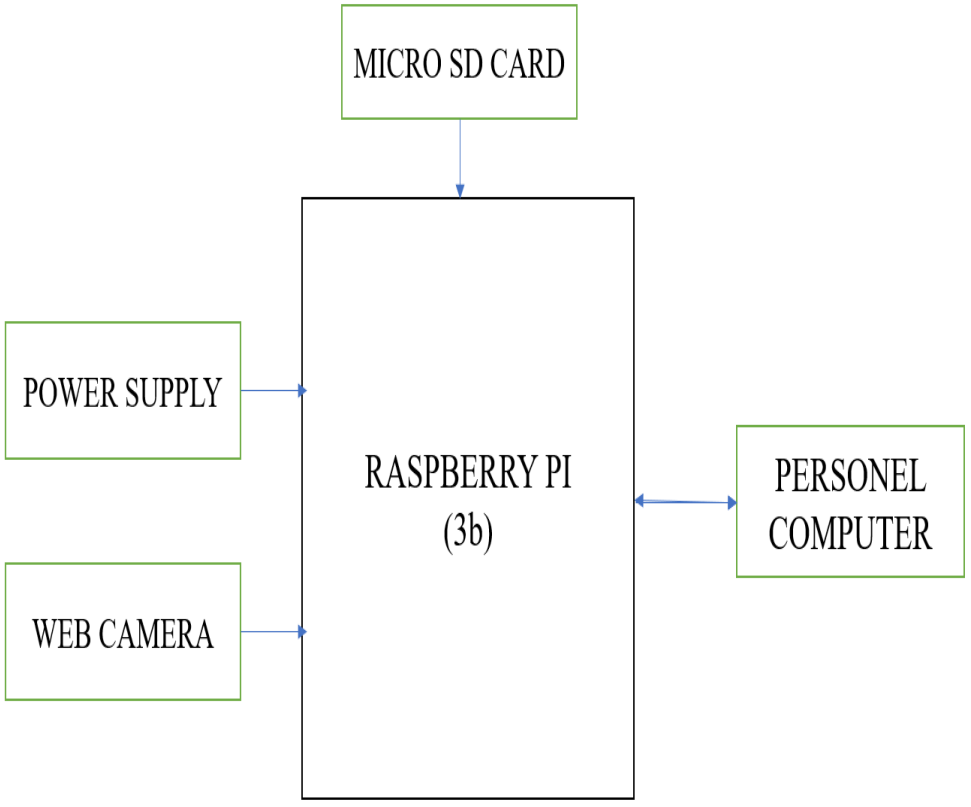

Figure 7.1 : Block Diagram of proposed system

7.1 Micro Sd Card

We need an SD card for your Raspberry Pi as it does not come with internal storage. Since all Raspberry Pi units have a dedicated SD card slot, this is the best, most compact option, and they are compatible with all versions of the Raspberry Pi, unlike flash drives or external hard drives. To run Raspberry Pi, you will need your SD card and load the operating system onto it. This will be done from your computer or laptop, or you can purchase an SD card with the Raspberry Pi operating system already loaded. Either way works as long as you have enough storage space and can fit the SD card into your Raspberry Pi.

7.2 Web Camera

A webcam is a video camera which is designed to record or stream to a computer or computer network they are primarily used in video telephony live streaming and social media and security webcams can be built in computer hardware or peripheral devices and a commonly used connected to a device using USB or wireless protocols.

Webcams have been used on the Internet as early as 199033 and the first widespread commercial one became available in 1994 early webcam usage on the Internet was primarily limited to stationary shots streamed to websites in the late 1990s and early 2000 instant messaging clients added support for webcam increasing their popularity in video conferencing computer manufacturers also started integrating webcam into laptop hardware in 2020 the COVID-19 pandemic kajaria shortage of webcams due to the increase in number of people working from home.

7.3 Power Supply

Power supply is a broad term but this lesson is restricted to discussion of circuits that generate a fixed or controllable magnitude dc voltage from the available form of input voltage. Integrated circuit (IC) chips used in dc supply for their proper operation. In majority of the cases the required voltages are of magnitudes varying between -18 to +18 volts. The 9V battery is an extremely common battery that was first used in transistor radios. It features a rectangular prism shape that utilizes a pair of snap connectors which are located at the top of the battery. A wide array of both large and small battery manufacturers produce versions of the 9V battery. Possible chemistries of primary (non-rechargeable) 9V batteries include Alkaline, Carbon-Zinc (Heavy Duty), Lithium. Possible chemistries of secondary (rechargeable) 9V batteries include nickel-cadmium (NiCd), nickel-metal hydride (NiMH), and lithium ion. The performance and application of the battery can vary greatly between different chemistries, meaning that some chemistries are better suited for some applications over others.

7.5: Specific Requirements

There will be several requirements to achieve the creation of the database.The below are the required software or packages needed to accompolish this objective.

Required software: OpenCV 3.4, Python 3

Required packages : tinker package - To provide user interface

Pi camera module – To interact with the raspberry pi's camera

7.6 Features Of Raspberry Pi

- Clock frequency: 1.2 GHz
- Chipset (SoC): Broadcom BCM2837
- Processor: 64-bit quad-core ARM Cortex-A53
- Graphics processor: Broadcom Dual Core Video Core IV (OpenGL ES 2.0, H.264Full HD @ 30 fps)
- Memory (SDRAM): 1 GB LPDDR2
- Number of USB 2.0 ports: 4
- Port extension: 40-pin GPIO
- Video outputs: HDMI and RCA, plus 1 CSI camera connector
- Audio outputs: 3.5 mm stereo jack or HDMI
- Data storage: MicroSD card
- Network connection: 10/100 Ethernet, 802.11n WiFi and Bluetooth 4.1 (BLE - LowEnergy)
- Peripherals: 17 x GPIO
- Supply: 5V 2.5A via micro USB
- Dimensions: 85.60 mm × 53.98 mm × 17 mm

7.7 Proposed work

All the Staff must register themselves by entering the required details and then theirimages will be captured and stored in the dataset. The faces detected will be comparedwith images present in the dataset. If match found, attendance will be marked marked for the respective member at the end of each session, list of absentees will besent to the respective PC destination. System Architecture Typically this process canbe divided into four stages[3],

7.8 Dataset Creation

Images of members are captured using a web cam. Multiple images of single member will be acquired with varied gestures and angles. These images undergo pre-processing. The images are cropped to obtain the Region of Interest (ROI) which will be further used in recognition process. Next step is to resize the cropped images to particular pixel position. Then these images will be converted from RGB to gray scale images. And then these images will be saved as the names of respective member in a folder.

7.9 Face Detection

This is required to create a rectangle around the faces in an image. It has got three parameters to consider- scaleFactor, minNeighbors, minSize. scaleFactor is used to indicate how much an image must be reduced in each image scale.

Figure 7.1: Face Detection

7.10 Face Recognition

Face recognition process can be divided into three steps prepare training data, train face recognizer, prediction[1]. Here training data will be the images present in the dataset. They will be assigned with a integer label of the member it belongs to. These images are then used for face recognition. the face to be recognized is calculated and then compared with the already computed histograms and returns the best matched label associated with the member it belongs[4].

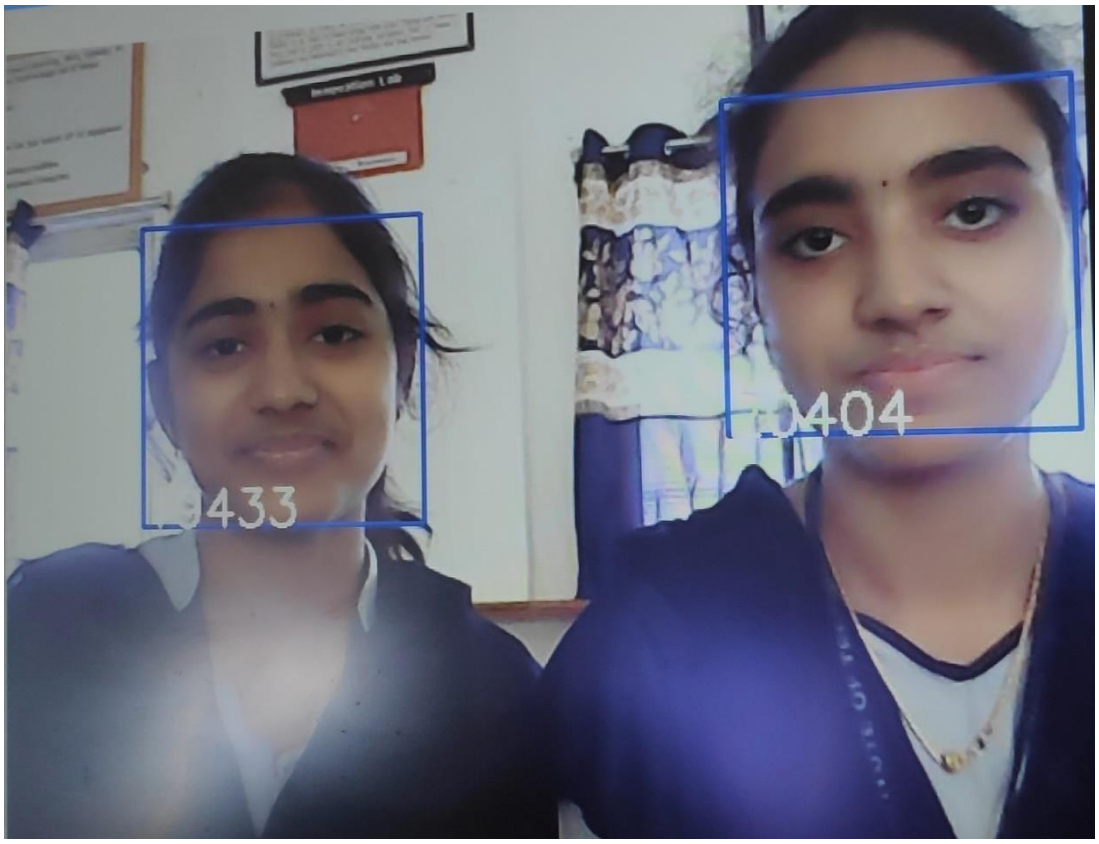

Figure 7.2 : FaceRecognition

7.11Attendance Updation

After face recognition process, the recognized faces will be marked as present in the excel sheet and the rest will be marked as absent and the list of absentees will be send to respective PC. In PC attendance will be updated with monthly attendance sheet at the end of every month[2].

7.12 Process of Attendance taking

1.Run the python program in the raspberry pi.

2.Face Recogniser Front end page will open.

3.For Creation of new database enter ID,name and click Take Images.

4.After that click on train images.

5.For attendance taking process click on track images .

6. After that camera window will open ,database created faces will appeared with Bounded box with ID Number

7.The attendace will check in the mobile phone or any other operating system with the help of link provided by the Thingspeak.

Chapter 8
RESULTS AND CONCLUSION

Here we are Creating database for each person by taking their name and their unique Id after that camera window will open and it take several images after completion of this process it will Train the images after that it will track the images by showing Id number on camera window.

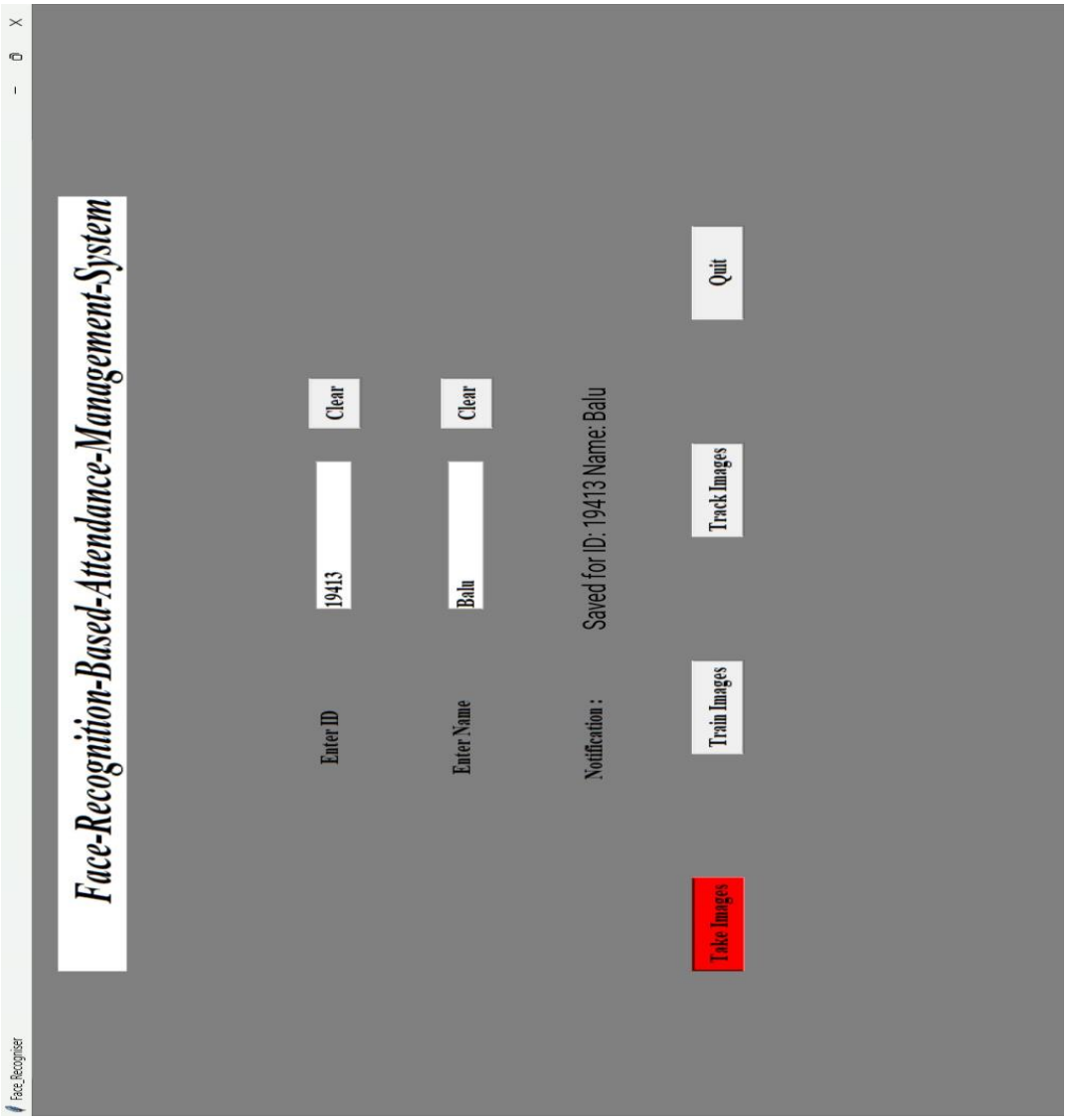

Figure 8.1 : Image Saved

Figure 8.2 : Image Trained

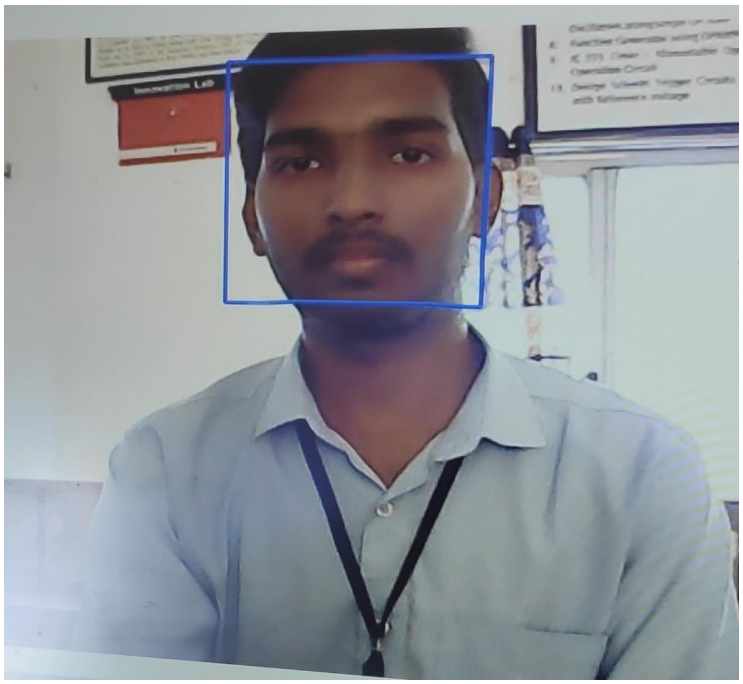

Figure 8.3 : Taking Image

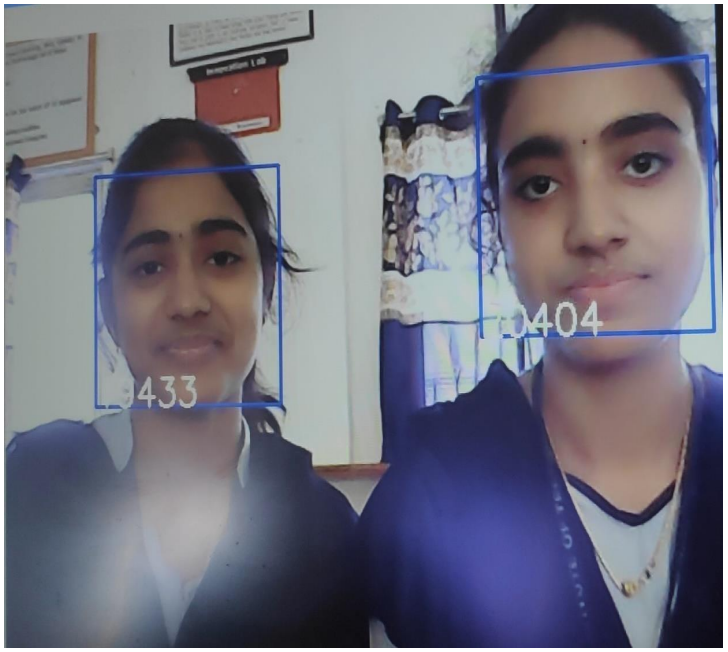

Figure 8.4 : Tracking Image

APPENDIX

SOURCE CODE:

```
import tkinter as tk
from tkinter import Message ,Text
import cv2,os
import shutil
import csv
import numpy as np
from PIL import Image, ImageTk
import pandas as pd
import datetime
import time
import tkinter.ttk as ttk
import tkinter.font as font
import urllib.request
window = tk.Tk()
#helv36 = tk.Font(family='Helvetica', size=36, weight='bold')
window.title("Face_Recogniser")
dialog_title = 'QUIT'
dialog_text = 'Are you sure?'
#answer = messagebox.askquestion(dialog_title, dialog_text)
window.geometry('1280x720')
window.configure(background='grey')
#window.attributes('-fullscreen', True)
window.grid_rowconfigure(0, weight=1)
window.grid_columnconfigure(0, weight=1)
#path = "profile.jpg"

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

```
#panel = tk.Label(window, image = img)
#panel.pack(side = "left", fill = "y", expand = "no")
#cv_img = cv2.imread("img541.jpg")
#x, y, no_channels = cv_img.shape
#canvas = tk.Canvas(window, width = x, height =y)
#canvas.pack(side="left")
#photo = PIL.ImageTk.PhotoImage(image = PIL.Image.fromarray(cv_img))
# Add a PhotoImage to the Canvas
#canvas.create_image(0, 0, image=photo, anchor=tk.NW)
#msg = Message(window, text='Hello, world!')
# Font is a tuple of (font_family, size_in_points, style_modifier_string)
message = tk.Label(window, text="Face-Recognition-Based-Attendance-
Management-System" ,fg="black", bg="grey" ,font=('times', 30, 'italic bold
underline'))
message.place(x=200, y=20)
lbl = tk.Label(window, text="Enter ID",width=20  ,height=2, bg="grey"
,font=('times', 15, ' bold ') )
lbl.place(x=400, y=200)
txt = tk.Entry(window,width=20 ,bg="white" ,font=('times', 15, ' bold '))
txt.place(x=700, y=215)
lbl2 = tk.Label(window, text="Enter Name",width=20  ,bg="grey"    ,height=2
,font=('times', 15, ' bold '))
lbl2.place(x=400, y=300)
txt2 = tk.Entry(window,width=20 ,bg="white" ,font=('times', 15, ' bold ') )
txt2.place(x=700, y=315)

lbl3 = tk.Label(window, text="Notification : ",width=20  ,bg="grey"  ,height=2
,font=('times', 15, ' bold '))
```

```
lbl3.place(x=400, y=400)
message = tk.Label(window, text="" ,bg="grey" ,width=30 ,height=2,
activebackground = "yellow" ,font=('times', 15, ' bold '))
message.place(x=700, y=400)
lbl3 = tk.Label(window, text="Attendance : ",width=20  ,bg="grey"  ,height=2
,font=('times', 15, ' bold underline'))
lbl3.place(x=400, y=600)
message2 = tk.Label(window, text="" ,bg="grey",activeforeground =
"green",width=30 ,height=2 ,font=('times', 15, ' bold '))
message2.place(x=700, y=600)
def clear():
    txt.delete(0, 'end')
    res = ""
    message.configure(text= res)
def clear2():
    txt2.delete(0, 'end')
    res = ""
    message.configure(text= res)
def is_number(s):
    try:
        float(s)
        return True
    except ValueError:
        pass
    try:
        import unicodedata
        unicodedata.numeric(s)
        return True
    except (TypeError, ValueError):
        pass
```

```
    return False
def TakeImages():
    Id=(txt.get())
    name=(txt2.get())
    if(is_number(Id) and name != ""):
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector=cv2.CascadeClassifier(harcascadePath)
        sampleNum=0
        while(True):
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, 1.3, 5)
            for (x,y,w,h) in faces:
                cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
                #incrementing sample number
                sampleNum=sampleNum+1
                #saving the captured face in the dataset folder TrainingImage
                cv2.imwrite("./TrainingImage/"+name +"."+Id +'.'+ str(sampleNum) +
".jpg", gray[y:y+h,x:x+w])
                #display the frame
                cv2.imshow('frame',img)
            #wait for 100 miliseconds
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
            # break if the sample number is morethan 100
            elif sampleNum>60:
                break
        cam.release()
        cv2.destroyAllWindows()
```

```
        res = "Images Saved for ID : " + Id +" Name : "+ name
        row = [Id , name]
        with open('./StudentDetails/StudentDetails.csv','a+') as csvFile:
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        message.configure(text= res)
    else:
        if(is_number(Id)):
            res = "Enter Alphabetical Name"
            message.configure(text= res)
        if(name.isalpha()):
            res = "Enter Numeric Id"
            message.configure(text= res)
def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()#recognizer =
cv2.face.LBPHFaceRecognizer_create()#$cv2.createLBPHFaceRecognizer()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector =cv2.CascadeClassifier(harcascadePath)
    faces,Id = getImagesAndLabels("TrainingImage")
    recognizer.train(faces, np.array(Id))
    recognizer.save("./TrainingImageLabel/Trainner.yml")
    res = "Image Trained"#+",".join(str(f) for f in Id)
    message.configure(text= res)
def getImagesAndLabels(path):
    #get the path of all the files in the folder
    imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
    #print(imagePaths)
    #create empth face list
    faces=[]
```

```
    #create empty ID list
    Ids=[]
    #now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
        #loading the image and converting it to gray scale
        pilImage=Image.open(imagePath).convert('L')
        #Now we are converting the PIL image into numpy array
        imageNp=np.array(pilImage,'uint8')
        #getting the Id from the image
        Id=int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(Id)
    return faces,Ids
def TrackImages():
    recognizer =
cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognizer()
    recognizer.read("./TrainingImageLabel/Trainner.yml")
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath);
    df=pd.read_csv("./StudentDetails/StudentDetails.csv")
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names =  ['Id','Name','Date','Time']
    attendance = pd.DataFrame(columns = col_names)
    ids =[0]
    while True:
        ret, im =cam.read()
        gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
        faces=faceCascade.detectMultiScale(gray, 1.2,5)
```

```
        for(x,y,w,h) in faces:
            cv2.rectangle(im,(x,y),(x+w,y+h),(225,0,0),2)
            Id, conf = recognizer.predict(gray[y:y+h,x:x+w])
            if(conf < 60):
                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,Id,date,timeStamp]
                if(Id not in ids):
                    ids.append(Id)
                    print('Updated')
                    wp =
urllib.request.urlopen("https://api.thingspeak.com/update?api_key=ZACOM7J7ZA3
DQ6QJ&field1=" +str(Id))
            else:
                Id='Unknown'
                tt=str(Id)
            # if(conf > 50):
            #     noOfFile=len(os.listdir("ImagesUnknown"))+1
            #     cv2.imwrite("ImagesUnknown\Image"+str(noOfFile) + ".jpg",
im[y:y+h,x:x+w])
            cv2.putText(im,str(tt),(x,y+h), font, 1,(255,255,255),2)
        attendance=attendance.drop_duplicates(subset=['Id'],keep='first')
        cv2.imshow('im',im)
        if (cv2.waitKey(1)==ord('q')):
            break
    ts = time.time()
    date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
```

```
    timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
    Hour,Minute,Second=timeStamp.split(":")
    fileName="./Attendance/Attendance_"+date+"_"+Hour+"-"+Minute+"-
"+Second+".csv"
    attendance.to_csv(fileName,index=False)
    cam.release()
    cv2.destroyAllWindows()
    # print(attendance)
    res=attendance
    message2.configure(text= res)
clearButton = tk.Button(window, text="Clear", command=clear  ,width=5  ,height=1
,activebackground = "Red" ,font=('times', 15, ' bold '))
clearButton.place(x=950, y=210)
clearButton2 = tk.Button(window, text="Clear", command=clear2  ,width=5
,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))
clearButton2.place(x=950, y=310)
takeImg = tk.Button(window, text="Take Images", command=TakeImages
,width=10 ,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))
takeImg.place(x=200, y=500)
trainImg = tk.Button(window, text="Train Images", command=TrainImages
,width=10 ,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))
trainImg.place(x=500, y=500)
trackImg = tk.Button(window, text="Track Images", command=TrackImages
,width=10 ,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))
trackImg.place(x=800, y=500)
quitWindow = tk.Button(window, text="Quit", command=window.destroy
,width=10 ,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))
quitWindow.place(x=1100, y=500)
window.mainloop()
```

References

[1] https://www.researchgate.net/publication/326261079_Face_detection_system_for_attendance_of_class'_students

[2] Lukas, Samuel, et al. "Student attendance system in classroom using face recognition technique." 2016 International Conference on Information and Communication Technology Convergence (ICTC). IEEE, 2016.

[3] https://becominghuman.ai/face-detection-using-opencv-with-haarcascade-classifiers-941dbb25177

[4] https://www.superdatascience.com/blogs/opencv-face-recognition