

CONTACTLESS DOORBELL SYSTEM

PROJECT REPORT

Submitted by

Boyapati Yagnavi Chowdari (185002020)

Ghanta Sandhya Rani (185002029)

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in Information Technology



Department of Information Technology

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

Rajiv Gandhi Salai (OMR), Kalavakkam – 603 110

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	iv
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	
1.	INTRODUCTION	
	1.1 Problem Statement	
	1.2 Problem Description	
	1.3 Motivation	
	1.4 Objective	
	1.5 Report Organization	
2.	LITERATURE SURVEY	
	2.1 Related Works	
	2.2 Existing Systems	
3.	HARDWARE DESCRIPTION	
	3.1 Raspberry pi	
	3.1.1 Hardware Specifications	
	3.1.2 Model B Raspberry Pi	
	3.1.3 Mounting OS	
	3.1.4 Starting Up Raspberry Pi	

3.2 Buzzer

3.3 LCD 16x2 Display

4. INTERFACING WITH RASPBERRY PI

4.1 Pi interfacing with webcam

4.2 Pi interfacing with buzzer

4.3 Pi interfacing with LCD using i2c

5. PROPOSED SYSTEM

5.1 Algorithms for face detection and
face recognition

5.2 Email notification

5.3 System Design

6. IMPLEMENTATION AND RESULTS

6.1 Train_model.py

6.2 Face.py

6.3 Results

7. CONCLUSION

ABSTRACT

A contactless doorbell as well as an IoT-based safety system that recognizes visitors and alerts the homeowner automatically is being proposed. This is especially important during the Covid pandemic and for disabled persons who have trouble reaching the bell. The contactless doorbell uses a raspberry pi controller to work along with a camera module to perform automatic operations. This system will help the homeowner know who has arrived at his/her door as well as it will act as a security system when the owner is not at home .

The system's operation is controlled by a Raspberry Pi controller. A camera module is used to record video and photos of anyone who comes close. There is no need for the person to press any buttons. The camera detects anyone approaching the entrance and uses face recognition to determine whether or not the person is registered in the system. If the person is registered in the system the display greets the person and alerts the owner over an email about who has arrived along with an image. If the visitor is not in the registered faces the house owner is alerted.

Face detection is done using Histogram of Oriented Gradients (HOG) and Linear SVM classifier. It is also combined with an image pyramid and a sliding window detection scheme. To build face recognition system, face detection is performed first, extracting face embeddings from each face in the dataset using deep learning and saving it. If they match, it gives the name of the stored images dataset to the owner. If they don't match it identified the visitor as an unknown person and alerts the user using buzzer and email.

LIST OF FIGURES

FIGURE NO	TITLE
3.1.1	Raspberry Pi Hardware Specifications
3.1.2	Pin Configuration of Raspberry Pi
3.2	Buzzer
3.3	LCD 16x2 Pin Configuration
4.3	Enabling I2C on Raspberry Pi
5.2	Output of Email Notification
5.3	Flowchart of Process
6.3	Results

LIST OF ABBREVIATIONS

ABBREVIATION	DESCRIPTION
LCD	Liquid Crystal Display
USB	Universal Serial Bus
SVM	Support Vector Machine
CNN	Convolution Neural Network
GPIO	General Purpose Input Output
SDA	Serial Data
SCL	Serial Clock
HOG	Histogram of Oriented Gradients

CHAPTER 1

INTRODUCTION

1.1 General

Doorbells have been playing an important role in protecting the security of modern homes since they were invented. A doorbell allows visitors to announce their presence and request entry into a building as well as enables the occupant to verify the identity of the guests to help prevent home robbery or invasion at a moment's notice. There are two types of doorbells depending on the requirement of wall wiring: the wired doorbells and the wireless doorbells. The former requires a wire to connect both the front door button and the back door button to a transformer, while the latter transfer the signal wirelessly using telephone technology. Modern buildings are typically equipped with wireless doorbell systems that employ radio technology to signal doorbells and answer the doors remotely. Although these doorbells are much more convenient than wired ones, they do not always satisfy the demands of modern homes for the following three reasons. First, the answering machines are normally located at a fixed place (often near to the door), if an occupant wants to answer the doorbell, he/she has to go to the answering machines. Second, if the occupant would like to see the visitors outside, he/she has to go to door. Third, the occupant has no way to answer or admit guests when he/she is not at home, nor to keep a record of guests.

This contactless doorbell system is composed of doorbell interfaced with raspberry pi and a camera module. A camera module is used to capture video and images of any approaching person. The person doesn't need to press any button, the camera is used to detect any person approaching the door and use face recognition to check if the person is registered in the system or not. If a person is registered int the system the doorbell greets the person and alerts the owner over IoT about who has arrived along with an image and doorbell ring. If a person is

not registered in the system, the system shows an image over IoT interface(mobile) and alerts the owner.

1.2 Problem statement

- The main aim of this project is to design a contactless doorbell which plays an important role in home security as well as improve the life quality by automating the doorbell concept.
- This project focuses on IoT related contactless doorbell system which is designed to notify the house owner automatically when a visitor approaches.
- The system implements face detection and face recognition along with a user interface and display unit.

1.3 Motivation

- A system that gives importance to home security and automation of doorbell.
- To greatly improve people's life quality and contribute to evolution of smart device.
- This is highly cost effective and can be implemented to improve the home security.

1.4 Objectives

- To implement efficient algorithm for face detection and face recognition.
- To alert the house owner when a face is detected.
- Displaying welcome message to the recognized visitor.

- To provide live video for user upon his choice.

1.5 Report Organization

Chapter1 : Introduction

Chapter2 : Literature survey

Chapter3 : Hardware Description

Chapter4 : Interfacing with raspberry pi

Chapter5 : Proposed System

Chapter6 : Implementation & Results

Chapter7 : Conclusion

+

..+

CHAPTER 2

LITERATURE REVIEW

2.1 Human face detection algorithm via Haar cascade classifier combined with three additional classifiers

Authors: Li Cuimei, Qi Zhiliang, Jia Nan, Wu Jianhua

Techniques Used: Human skin hue histogram matching, eyes and mouth detection

Human face detection has been a challenging issue in the areas of image processing and pattern recognition. A new human face detection algorithm by primitive Haar cascade algorithm combined with three additional weak classifiers is proposed in this paper. The three weak classifiers are based on skin hue histogram matching, eyes detection and mouth detection. First, images of people are processed by a primitive Haar cascade classifier, nearly without wrong human face rejection (very low rate of false negative) but with some wrong acceptance (false positive). Secondly, to get rid of these wrongly accepted non-human faces, a weak classifier based on face skin hue histogram matching is applied and a majority of non-human faces are removed. Next, another weak classifier based on eyes detection is appended and some residual non-human faces are determined and rejected. Finally, a mouth detection operation is utilized to the remaining non-human faces and the false positive rate is further decreased. With the help of OpenCV, test results on images of people under different occlusions and illuminations and some degree of orientations and rotations, in both training set and test set show that the proposed algorithm is effective and achieves state-of-the-art performance. Furthermore, it is efficient because of its easiness and

simplicity of implementation.

2.2 Home Security system for alone elderly people

Authors: Jarupath Kulsiriruangyos, Varanya Rattanawutikul, Patcharaporn Sangsartra, Damras Wongsawang

Techniques Used: Sinch, OpenCv

This paper has introduced a method to develop a home security system for elderly who live alone which help elderly people to identify to-be identified person without face-to-face communication to reduce an abuse that could be occurred to elderly people and help them broadcast an emergency message to notice their family that there are something wrong with their parent. The purposes of this project is to improve a security in elderly people by avoid face-to-face communication with unknown people. In addition, to help elderly people to get timely care in an emergency situation, this project provided an emergency broadcast message to ensure that elderly's family would be notice about dangerous that occurred with their parent. Moreover, this project aims to improve comfortability in elderly people. This project uses a mobile application to provide an easier and faster way to identify to-be identify people who come to contact them with less movement is needed. The system consists of 2 main components which are Raspberry Pi set and Android application. Raspberry Pi set is the component that setup at elderly's front door area and Android application is used as an interface of the system for elderly to use this system easily. Next, to create an interface that would fit with elderly people use, the application interface was designed to be very simple as it could be by concern all of the elderly's eye conditions. However, to maintain elderly satisfaction in using this system, this system has to provide clear instruction to ensure that elderly people will understand clearly what to do to complete the task and reduce fear to use this system.

2.3 An Efficient Face Detection and Recognition Method for Surveillance

Authors: K.V.Arya, Abhinav Adarsh

Techniques used: PCA verification algorithm

Identification of a person in surveillance area using face information has many applications in real life. The face recognition in the images got from surveillance camera is challenging task due to the presence of multiple faces in the given area. In this paper a method has been proposed where the algorithm has been modified for the detection of the faces, extraction of the feature information and matching the features. The detection has been carried out by combining the skin models, Haar like filter for each individual, which gives the information of multiple faces for single individual. Then PCA has been used to identify the desired single face. The feature extraction has been carried out by estimating and selection of sub-space from MMP.

The experimental results show that the proposed method detects and recognize faces efficiently, giving recognition accuracy from 83% to 99% by varying the number of vector space. There is a trade-off between recognition rate and timing efficiency. The average time taken for only comparing in 100 dimensions for 97.3% accuracy for 700 test images is 169.4 sec. The work can further be extended for improving the recognition accuracy as well as time for large face databases.

2.4 Analysis of Different Face Recognition Algorithms

Authors: Rakesh Saini, Abhishek Saini, Deepak Agarwal

Techniques used: PCA, Linear Discriminant Analysis, Independent Component Analysis

The paper presented different algorithms which can be used to develop a face recognition system. A comparison of these algorithms and discussed the merits and demerits. In this paper they compared individual algorithms and algorithms that are used in combination of PCA. On the basis of analysis there is no specific algorithm which suited well for good recognition rate but according to user requirement of specification different algorithms can be combined for good recognition and results. The technique which suits well for face recognition is incremental PCA with LDA. This method can process face images (including training and identifying) in high speed and obtain good results. The incremental PCA –LDA is very efficient in memory usage and it is very efficient in calculation of the first basis vectors. This algorithm gives an acceptable face recognition success rate

CHAPTER 3

HARDWARE DESCRIPTION

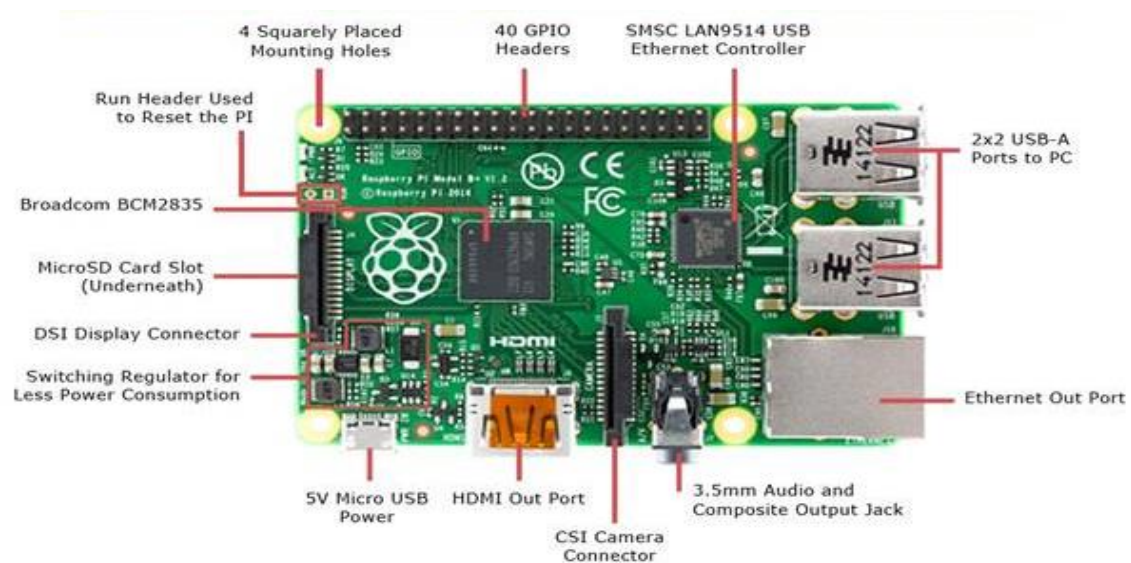
3.1 Raspberry Pi

The raspberry pi comes in two models, they are model A and model B. The main difference between model A and model B is USB port. Model A board will consume less power and that does not include an Ethernet port. But, the model B board includes an Ethernet port and designed in China. The raspberry pi comes with a set of opensource technologies, i.e. communication and multimedia web technologies. In the year 2014, the foundation of the raspberry pi board launched the computer module, that packages a model B raspberry pi board into module for use as a part of embedded systems, to encourage their use.

3.1.1 Hardware specifications of Raspberry Pi :

The raspberry pi board comprises a program memory (RAM), processor and graphics chip, CPU, GPU, Ethernet port, GPIO pins, Xbee socket, UART, power source connector. And various interfaces for other external devices. It also requires mass storage, for that we use an SD flash memory card. So that raspberry pi board will boot from this SD card similarly as a PC boots up into windows from its hard disk.

Essential hardware specifications of raspberry pi board mainly include SD card containing Linux OS, US keyboard, monitor, power supply and video cable. Optional hardware specifications include USB mouse, powered USB hub, case, internet connection, the Model A or B: USB Wifi adaptor is used and internet connection to Model B is LAN cable.



Memory

The raspberry pi model Aboard is designed with 256MB of SDRAM and model B is designed with 512MB. Raspberry pi is a small size PC compare with other PCs. The normal PCs RAM memory is available in gigabytes. But in raspberry pi board, the RAM memory is available more than 256MB or 512MB. We are using model B.

CPU (Central Processing Unit)

The Central processing unit is the brain of the raspberry pi board and that is responsible for carrying out the instructions of the computer through logical and mathematical operations. The raspberry pi uses ARM11 series processor, which has joined the ranks of the Samsung galaxy phone.

GPU (Graphics Processing Unit)

The GPU is a specialized chip in the raspberry pi board and that is designed to speed up the operation of image calculations. This board designed with a Broadcom video core IV and it supports OpenGL.

Ethernet Port

The Ethernet port of the raspberry pi is the main gateway for communicating with additional devices. The raspberry pi Ethernet port is used to plug your home router to access the internet.

GPIO Pins

The general purpose input & output pins are used in the raspberry pi to associate with the other electronic boards. These pins can accept input & output commands based on programming raspberry pi. The raspberry pi affords digital GPIO pins. These pins are used to connect other electronic components. For example, you can connect it to the temperature sensor to transmit digital data.

XBee Socket

The XBee socket is used in raspberry pi board for the wireless communication purpose.

Power Source Connector

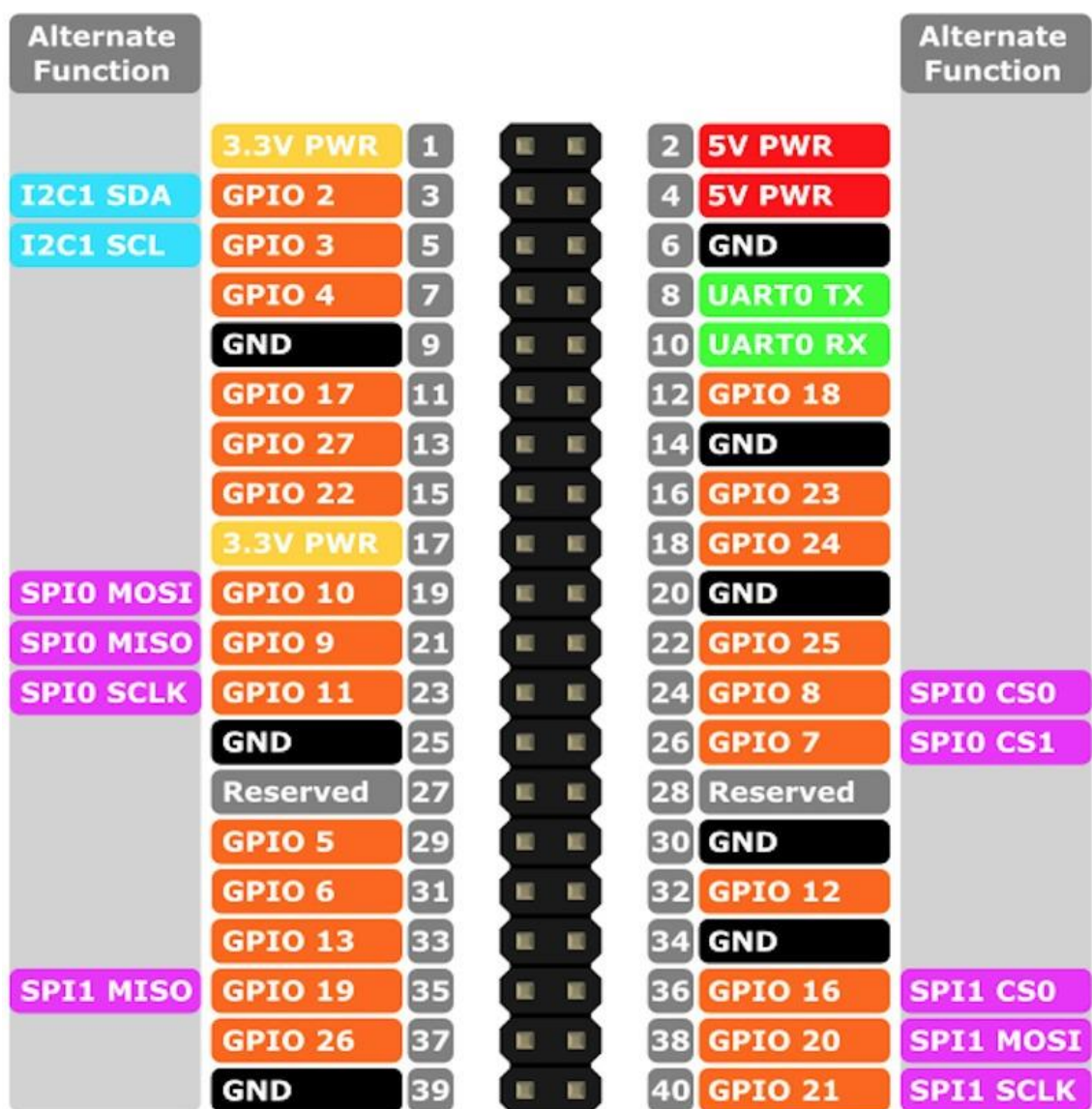
The power source cable is a small switch, which is placed on side of the shield. The main purpose of the power source connector is to enable an external power source.

UART

The Universal Asynchronous Receiver/ Transmitter is a serial input & output port. That can be used to transfer the serial data in the form of text and it is useful for converting the debugging code.

Display

The connection options of the raspberry pi board are two types such as HDMI and Composite. Many LCD and HD TV monitors can be attached using an HDMI male cable and with a low-cost adaptor.



Pin configuration

3.1.2 Model B Raspberry pi Board

The Raspberry Pi is a Broadcom BCM2835 SOC (system on chip board). It comes equipped with a 700 MHz, 512 MB of SDRAM and ARM1176JZF-S core CPU. The USB 2.0 port of the raspberry pi boards uses only external data connectivity options. The Ethernet in the raspberry pi is the main gateway to interconnect with other devices and the internet in model B. This draws its power from a micro USB adapter, with a minimum range of 2.5 watts(500 MA). The graphics, specialized chip is designed to speed up the manipulation of image calculations. This is in built with Broadcom video core IV cable, that is useful if you want to run a game and video through your raspberry pi.

Features of Raspberry PI Model B

- 512 MB SDRAM memory
- Broadcom BCM2835 SoC full high definition multimedia processor
- Dual Core Video Core IV Multimedia coprocessor
- Single 2.0 USB connector
- HDMI (rev 1.3 and 1.4) Composite RCA (PAL & NTSC) Video Out
- 3.5 MM Jack, HDMI Audio Out

Raspberry Pi 3 Model B

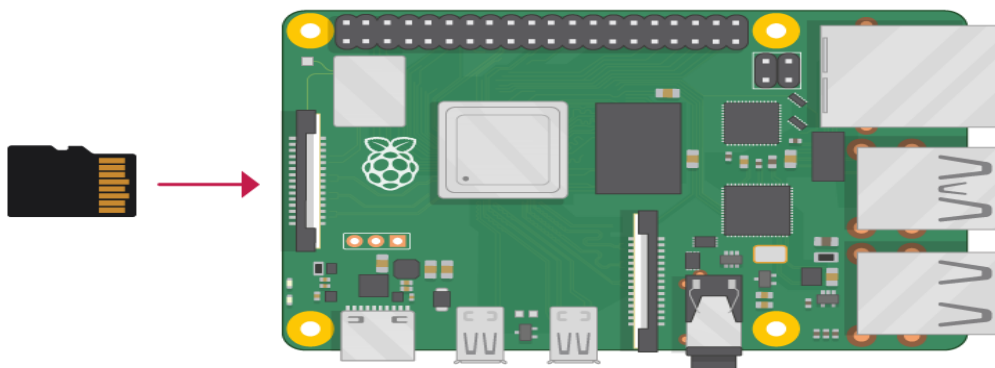


Model B Raspberry pi Board

- MMC, SD, SDIO Card slot on board storage
- Linux Operating system
- Dimensions are 8.6cm*5.4cm*1.7cm
- On board 10/100 Ethernet RJ45 jack

Raspberry Pi OS

- Raspberry Pi OS is a free operating system based on Debian, optimized for the Raspberry Pi hardware, and is the recommended operating system for normal use on a Raspberry Pi. The OS comes with over 35,000 packages: precompiled software bundled in a nice format for easy installation on your Raspberry Pi.
- Raspberry Pi needs an SD card to store all its files and the Raspberry Pi OS operating system. A microSD card with a capacity of at least 8GB is needed.
- Many sellers supply SD cards for Raspberry Pi that are already set up with Raspberry Pi OS and ready to go.

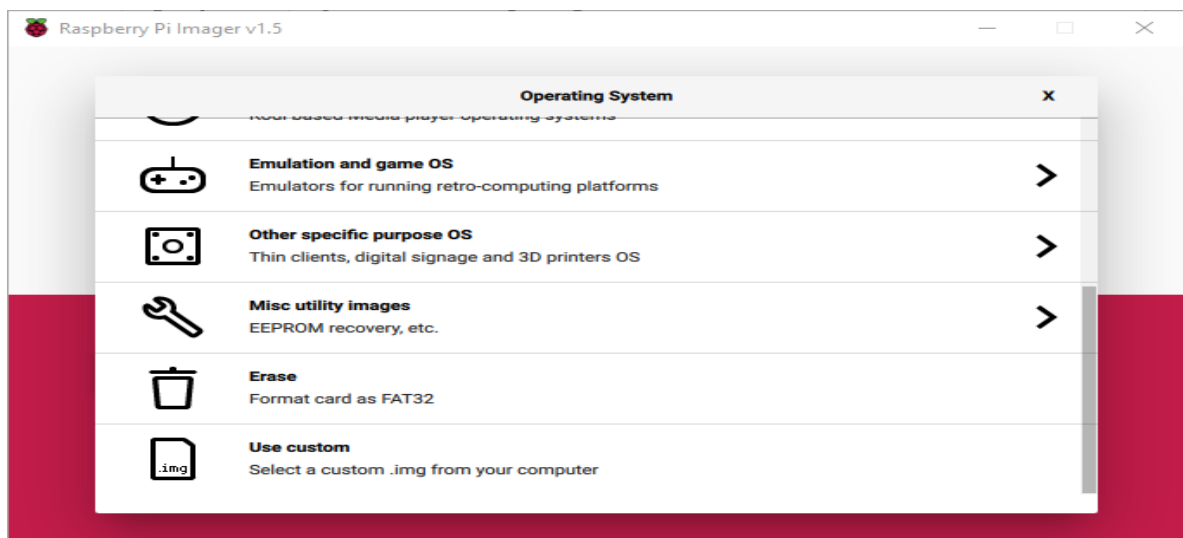


3.1.3 Mounting OS onto raspberry pi:

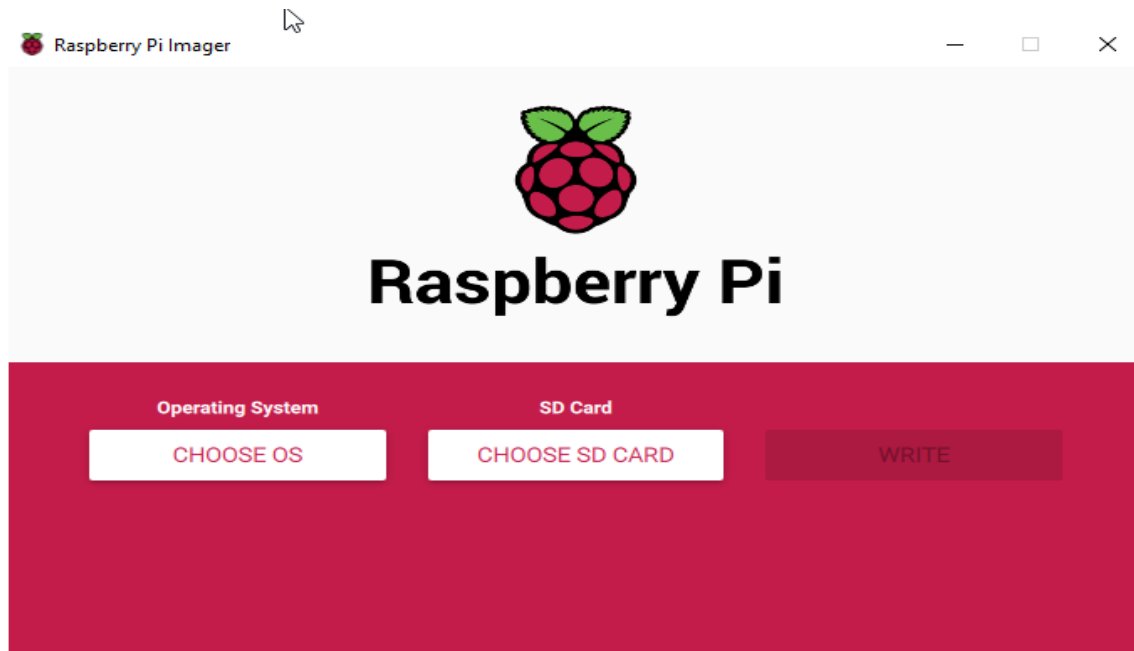
Step 1: Download and Install Raspberry Pi Imager

Step 2: Run Raspberry Pi Imager and Format the SD Card

- With the target SD Card inserted into computer or external card reader, open Raspberry Pi Imager.
- When Raspberry Pi Imager is ran you are presented with two options: Choose OS and Choose SD Card.
- Scroll down to the bottom of the list and click on the Erase option.
- Now click the "Choose SD Card" Option and select the target SD Card. Note that any data still on this SD card will be erased forever, and you should ensure that you have a backup of any files you would like to keep.
- With the SD Card Selected the "Write" option becomes available. Click Write and wait for the process to complete. Once the process is complete, a notification window will open letting you know that it's now ok to remove the SD card from the reader. Remove the SD Card and reinsert it into the reader to make it available again.



Step 3: Burn the Raspberry Pi OS Image to the SD Card



- With a freshly formatted SD card, we can now move on to installing the operating system.
- Click the "Choose OS" button, and select one of the available operating systems. For the purpose of this tutorial we will be using the top option, Raspberry Pi OS (32-Bit)
- Now select the SD card we just formatted in the previous step.
- The "Write" option will become available. Click "Write" to begin burning the image to the SD card.
- It can take anywhere from a few minutes to upwards of half an hour for the process to complete depending on the quality and speed of the SD card, card reader, and computer.
- Once Raspberry Pi Imager has finished writing the files to the SD card, it will verify that the image on the SD card is identical to the image file used

to burn the image. This usually takes less than a minute but could take longer.

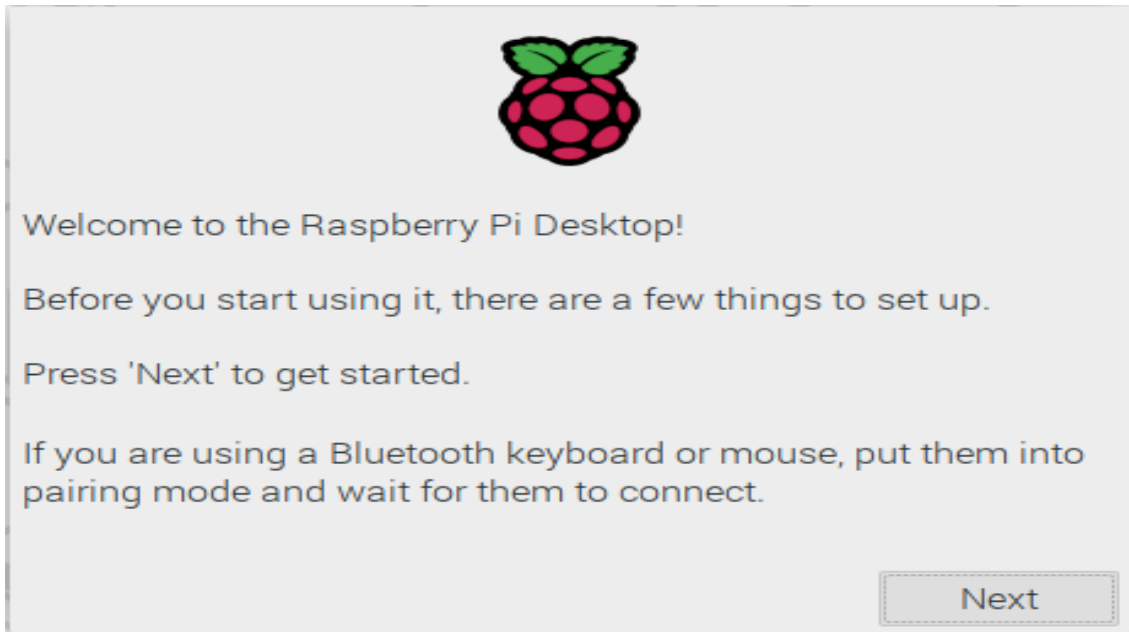
- When the verification process is complete, a notification window will open letting you know that the write was successful and that it's now safe to remove the SD card.

3.1.4 Starting up Raspberry Pi:

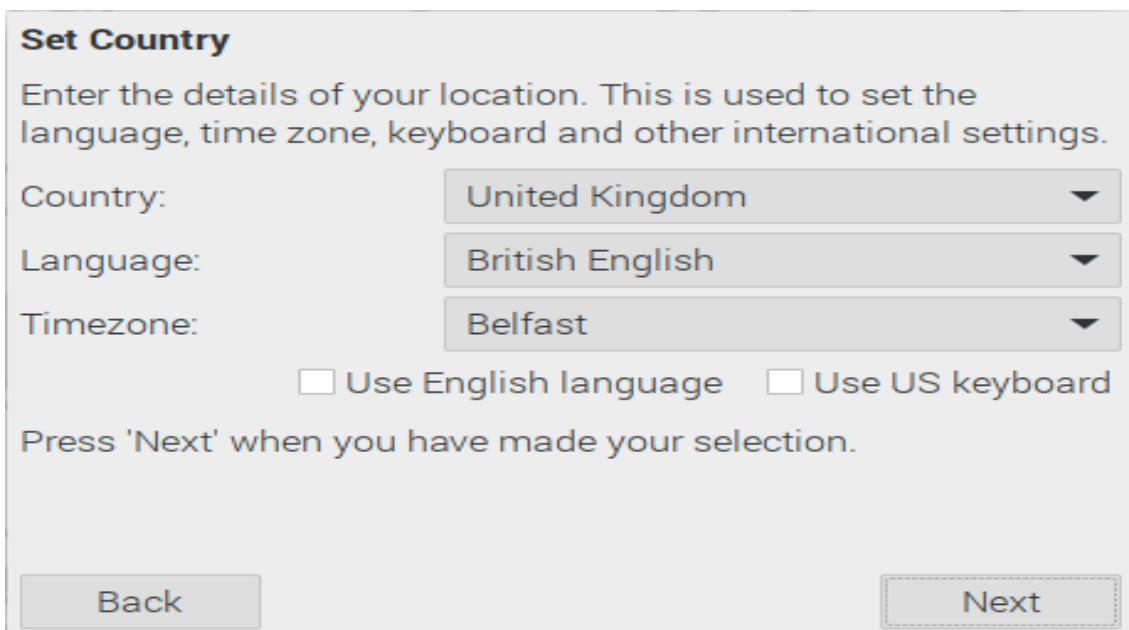
- Plug the power supply into a socket and connect it to Raspberry Pi's power port.
- A red LED light up on the Raspberry Pi, which indicates that Raspberry Pi is connected to power. As it starts up (this is also called booting), will see raspberries appear in the top left-hand corner of the screen.
- After a few seconds the Raspberry Pi OS desktop will appear.



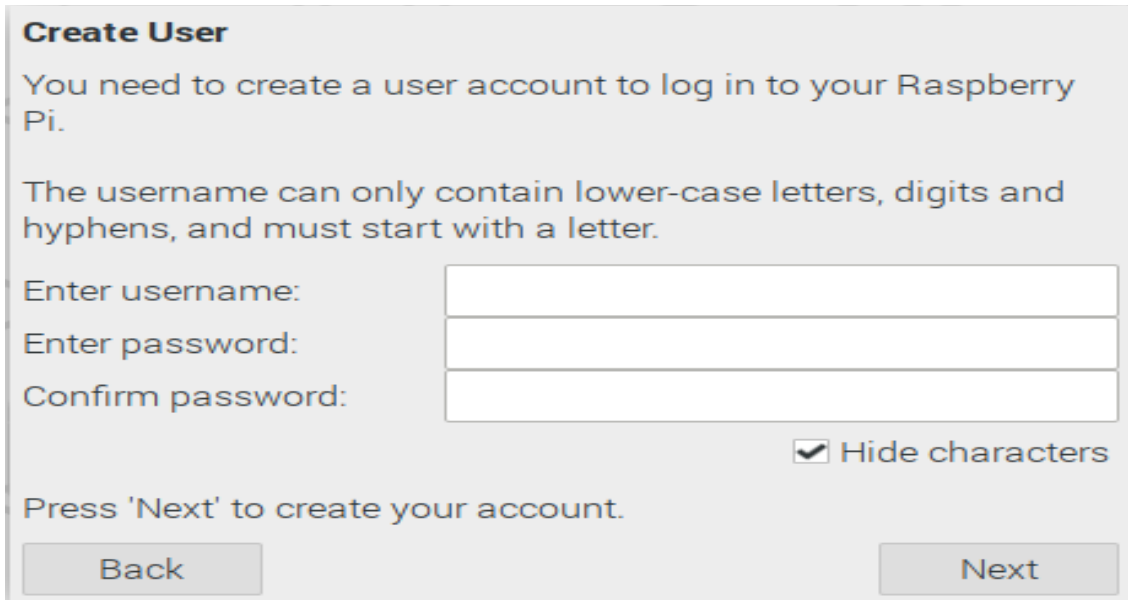
- When you start your Raspberry Pi for the first time, the Welcome to Raspberry Pi application will pop up and guide you through the initial setup.



- Click on Next to start the setup.
- Set your Country, Language, and Time zone, then click on Next again.



- Enter a new username and password for your Raspberry Pi and click on Next.



Create User

You need to create a user account to log in to your Raspberry Pi.

The username can only contain lower-case letters, digits and hyphens, and must start with a letter.

Enter username:

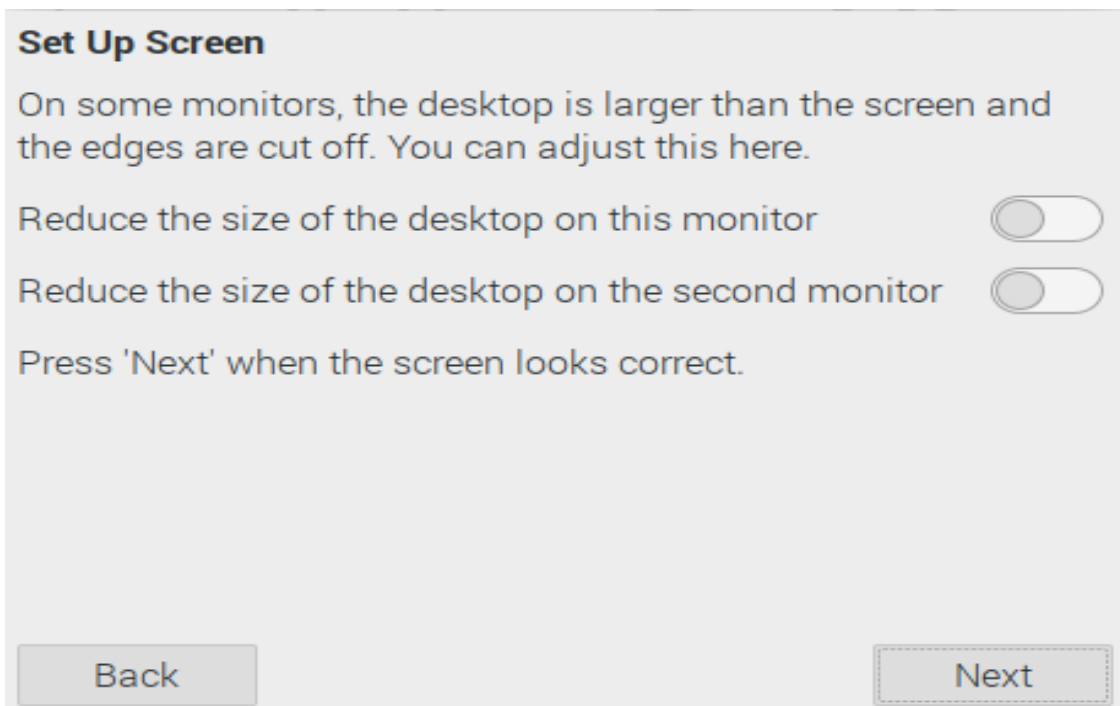
Enter password:

Confirm password:

☒ Hide characters

Press 'Next' to create your account.

- Set up your screen so that the Desktop completely fills your monitor.



Set Up Screen

On some monitors, the desktop is larger than the screen and the edges are cut off. You can adjust this here.

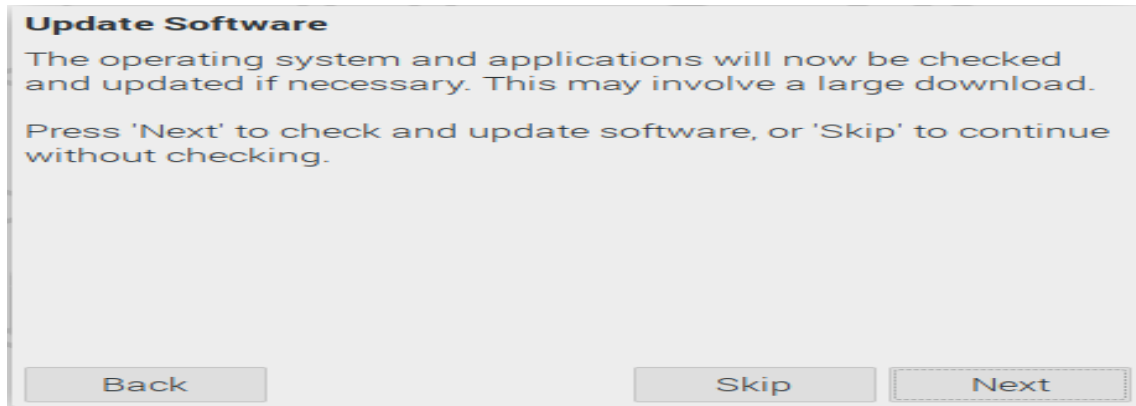
Reduce the size of the desktop on this monitor ☐

Reduce the size of the desktop on the second monitor ☐

Press 'Next' when the screen looks correct.

- Connect to your wireless network by selecting its name, entering the password, and clicking on Next.

- Click on Next, and let the wizard check for updates to Raspberry Pi OS and install them (this might take a little while).



- Click on Restart to finish the setup.

3.2 Buzzer

An audio signalling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren

Types of Buzzer

A buzzer is available in different types which include the following.

- Piezoelectric
- Electromagnetic
- Mechanical
- Electromechanical
- Magnetic

The **specifications of the buzzer** include the following.

- The frequency range is 3,300Hz
- Operating Temperature ranges from -20°C to $+60^{\circ}\text{C}$
- Operating voltage ranges from 3V to 24V DC
- The sound pressure level is 85dBA or 10cm
- The supply current is below 15mA

Magnetic buzzers utilize an electric charge instead of depending on piezo materials to generate a magnetic field, after that it permits another element of the buzzer to vibrate & generate sound. The applications of magnetic buzzers are similar to the piezo type in household devices, alarms such as watches, clocks & keyboards.

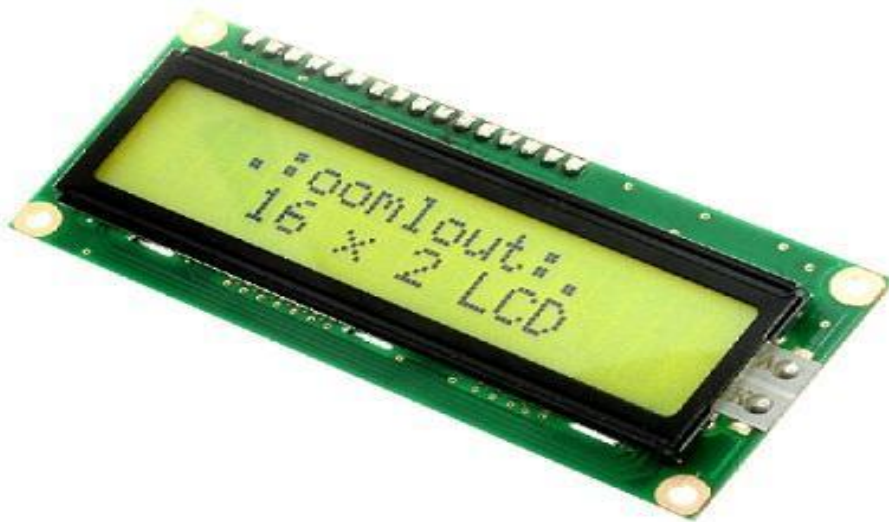
An electromagnetic buzzer consists of an oscillator, solenoid coil, magnet, vibration diaphragm, housing etc. Pretty much works the same as a magnetic buzzer, where they produce sound through magnetism, with a frequency of 2 kHz.



It includes two pins namely positive and negative. The positive terminal of this is represented with the '+' symbol or a longer terminal. This terminal is powered through 6Volts whereas the negative terminal is represented with the '-' symbol or short terminal and it is connected to the GND terminal.

3.3 LCD 16x2 Display

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.

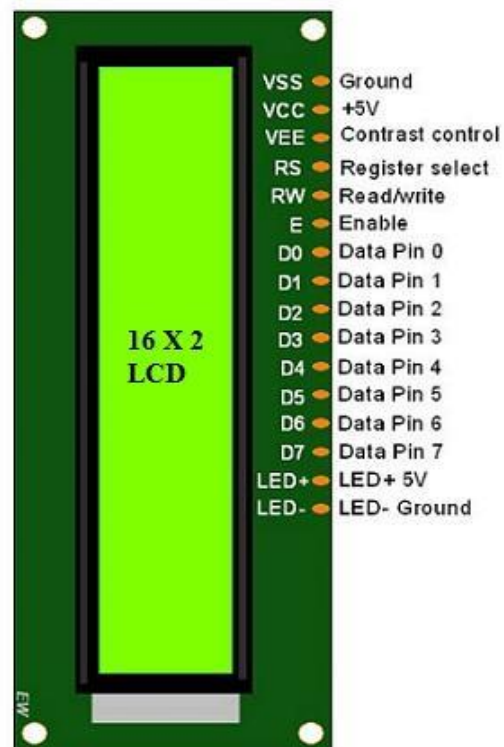


LCD 16×2 Pin Diagram

The 16×2 LCD pinout is shown below.

- Pin1 (Ground/Source Pin): This is a GND pin of display, used to connect the GND terminal of the microcontroller unit or power source.
- Pin2 (VCC/Source Pin): This is the voltage supply pin of the display, used to connect the supply pin of the power source.
- Pin3 (V0/VEE/Control Pin): This pin regulates the difference of the display, used to connect a changeable POT that can supply 0 to 5V.

- Pin4 (Register Select/Control Pin): This pin toggles among command or data register, used to connect a microcontroller unit pin and obtains either 0 or 1 (0 = data mode, and 1 = command mode).
- Pin5 (Read/Write/Control Pin): This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1 = Read Operation).
- Pin 6 (Enable/Control Pin): This pin should be held high to execute Read/Write process, and it is connected to the microcontroller unit & constantly held high.
- Pins 7-14 (Data Pins): These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to microcontroller unit like 0 to 7.
- Pin15 (+ve pin of the LED): This pin is connected to +5V
- Pin 16 (-ve pin of the LED): This pin is connected to GND.



Features of LCD16x2

The features of this LCD mainly include the following.

- The operating voltage of this LCD is 4.7V-5.3V
- It includes two rows where each row can produce 16-characters.
- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8 pixel box
- The alphanumeric LCDs alphabets & numbers
- Its display can work on two modes like 4-bit & 8-bit
- These are obtainable in Blue & Green Backlight
- It displays a few custom generated characters

Registers of LCD

A 16×2 LCD has two [registers](#) like data register and command register. The RS (register select) is mainly used to change from one register to another. When the register set is '0', then it is known as command register. Similarly, when the register set is '1', then it is known as data register.

Command Register

The main function of the command register is to store the instructions of command which are given to the display. So that predefined tasks can be performed such as clearing the display, initializing, set the cursor place, and display control. Here commands processing can occur within the register.

Data Register

The main function of the data register is to store the information which is to be exhibited on the LCD screen. Here, the ASCII value of the character is the information which is to be exhibited on the screen of LCD. Whenever we send the information to LCD, it transmits to the data register, and then the process will be starting there. When register set =1, then the data register will be selected.

16×2 LCD Commands

The commands of LCD 16X2 include the following.

- For Hex Code-01, the LCD command will be the clear LCD screen
- For Hex Code-02, the LCD command will be returning home
- For Hex Code-04, the LCD command will be decrement cursor
- For Hex Code-06, the LCD command will be Increment cursor
- For Hex Code-05, the LCD command will be Shift display right
- For Hex Code-07, the LCD command will be Shift display left

- For Hex Code-08, the LCD command will be Display off, cursor off
- For Hex Code-0A, the LCD command will be cursor on and display off
- For Hex Code-0C, the LCD command will be cursor off, display on
- For Hex Code-0E, the LCD command will be cursor blinking, Display on
- For Hex Code-0F, the LCD command will be cursor blinking, Display on
- For Hex Code-10, the LCD command will be Shift cursor position to left
- For Hex Code-14, the LCD command will be Shift cursor position to the right
- For Hex Code-18, the LCD command will be Shift the entire display to the left
- For Hex Code-1C, the LCD command will be Shift the entire display to the right
- For Hex Code-80, the LCD command will be Force cursor to the beginning (1st line)
- For Hex Code-C0, the LCD command will be Force cursor to the beginning (2nd line)
- For Hex Code-38, the LCD command will be 2 lines and 5×7 matrix

CHAPTER 4

INTERFACING WITH RASPBERRY PI

4.1 Raspberry Pi interfacing with webcam

Raspberry Pi persists in its pride stage due to its relevance in performing image processing applications. Real time image processing schemes can be developed using a Raspberry Pi, as it supports the webcam interface, and there by we can process the algorithms for detection, recognition, segmentation, surveillance etc.



Hardware Modules:

1. Raspberry Pi
2. USB Webcam

Additional Software Packages:

1. fswebcam package

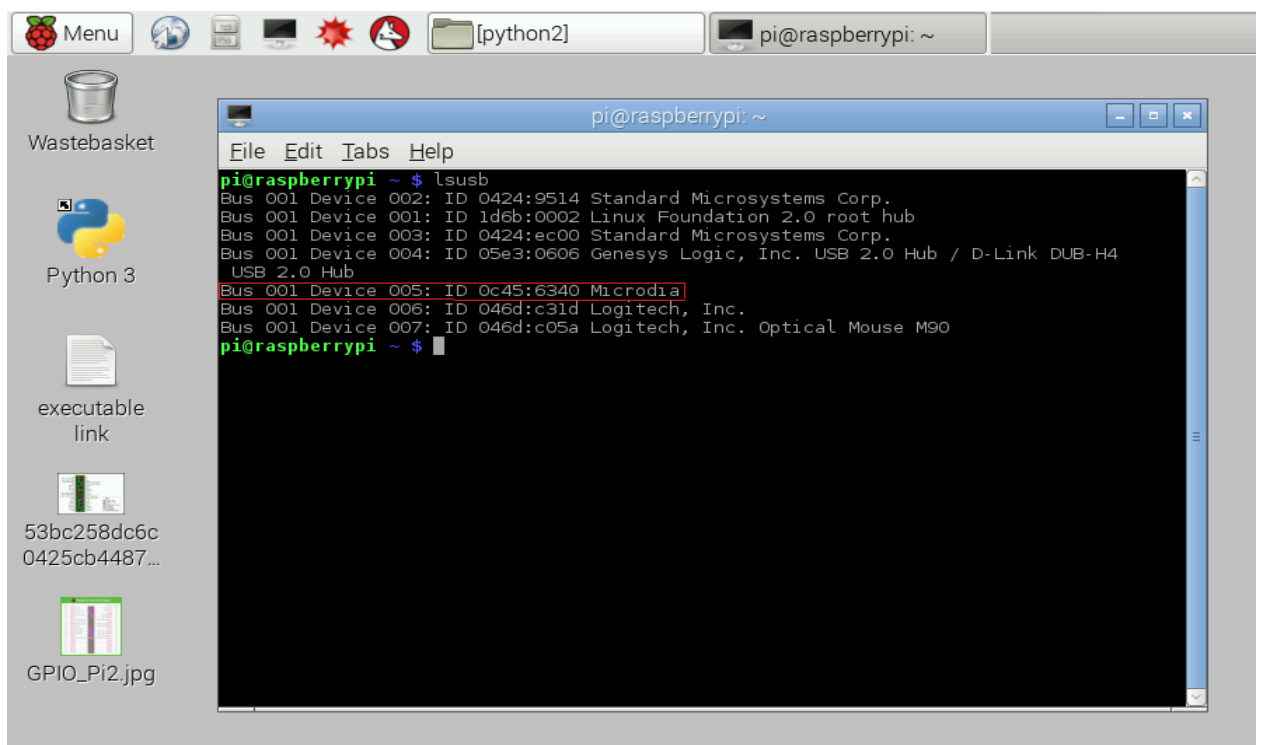
The webcam should be connected to the USB port of the Raspberry Pi. after connecting open the LX Terminal and type in **lsusb**.

If the camera is detected then a message will be displayed, as shown in the image above (marked in red). Now install fswebcam to start using the camera. To install the fswebcam software package, type in as follows:

```
sudo apt-get install fswebcam
```

After the installation enter the command fswebcam followed with a desired file name for saving the image.

```
fswebcam testimage.
```



For selecting proper resolution commands given below are followed

```
fswebcam -r 640x480 testimage.jpg
```

The camera will capture the snapshot and it will be saved in the main folder using the preferred name.

4.2 Interfacing Raspberry Pi with bell (buzzer)

Buzzers are often used in DIY projects to create simple sounds for alarms and alerts.

A buzzer is used in this project to generate a buzzing sound when a face is detected at the door.

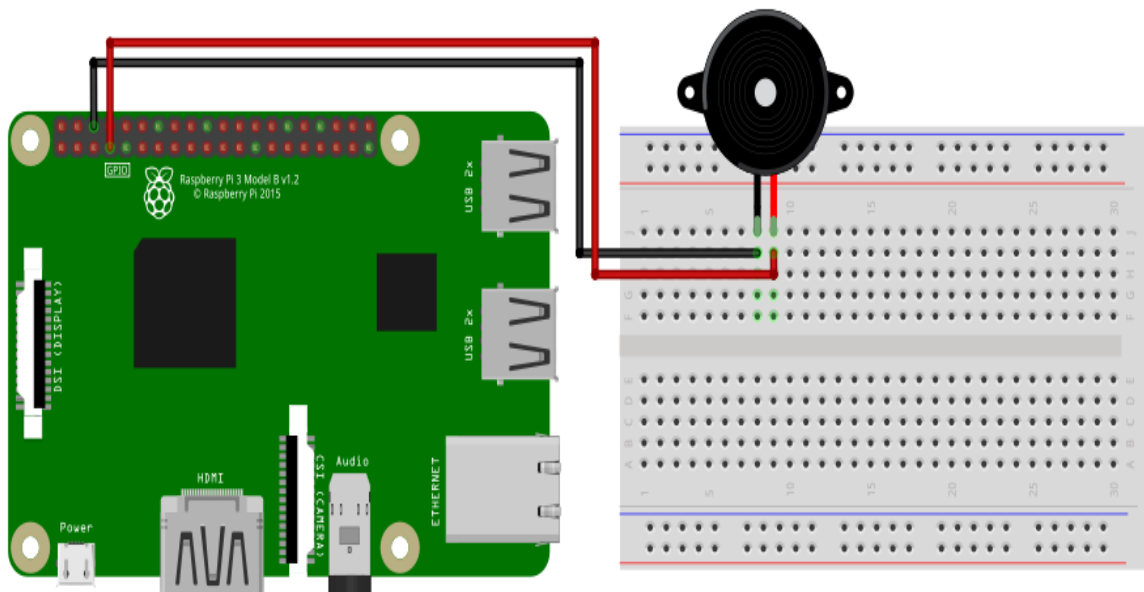
Setting up a buzzer on Raspberry Pi :

These are the parts we will need:

- Raspberry Pi
- Breadboard
- Jumper wires
- Buzzer

Use two socket-socket jumper wires to attach your buzzer to your Raspberry Pi device. The long leg of the buzzer must be wired to 3V3 and the short leg of the buzzer should be wired to a GND (ground) pin.

The buzzer should sound straight away, so you know that it works.



Code:

When the algorithm detects a face, execute the following code :

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(17,GPIO.OUT)

/*on face detection make the pin high*/
GPIO.output(17,GPIO.HIGH)
time.sleep(5)
GPIO.output(17,GPIO.LOW)
time.sleep(5)
GPIO.cleanup()
```

4.3 Interfacing 16x2 LCD display with Raspberry Pi using I2C

I2C Module

In the module left side, we have 4 pins, and two are for power (Vcc and GND), and the other two are the interface I2C (SDA and SCL) . The plate pot is for display contrast adjustment, and the jumper on the opposite side allows the back light is controlled by the program or remain off for power saving.

LCD

LCD stands for Liquid crystal display. 16×2 LCD is named so because; it has 16 Columns and 2 Rows. There are a lot of combinations available like 8×1, 8×2, 10×2, 16×1, etc. but the most used one is the 16×2 LCD. So, it will have 16×2 = 32 characters in total and each character will be made of 5×8 Pixel Dots.

Circuit Diagram

I2C Module is designed suitably to connect all 17 pins of 16X2 LCD Module from Backside to make interfacing easy. I2C Module is having only 4pins which we need connect with Raspberry PI and they are:

- GND >> to >> GND
- VCC >> to >> 5V
- SDA >> to >>GPIO
- SCL >> to >>GPIO

Also, I2C module featured with onboard potentiometer to adjust the bright as per requirement

Step by step process for Interfacing 16X2 LCD Display with Raspberry Pi using I2C Module, Python and GPIO's.

Step-1: Enable i2c using raspi-config utility

First, Update and upgrade your Raspberry PI with below command

sudo apt-get update

Sudo apt-get upgrade

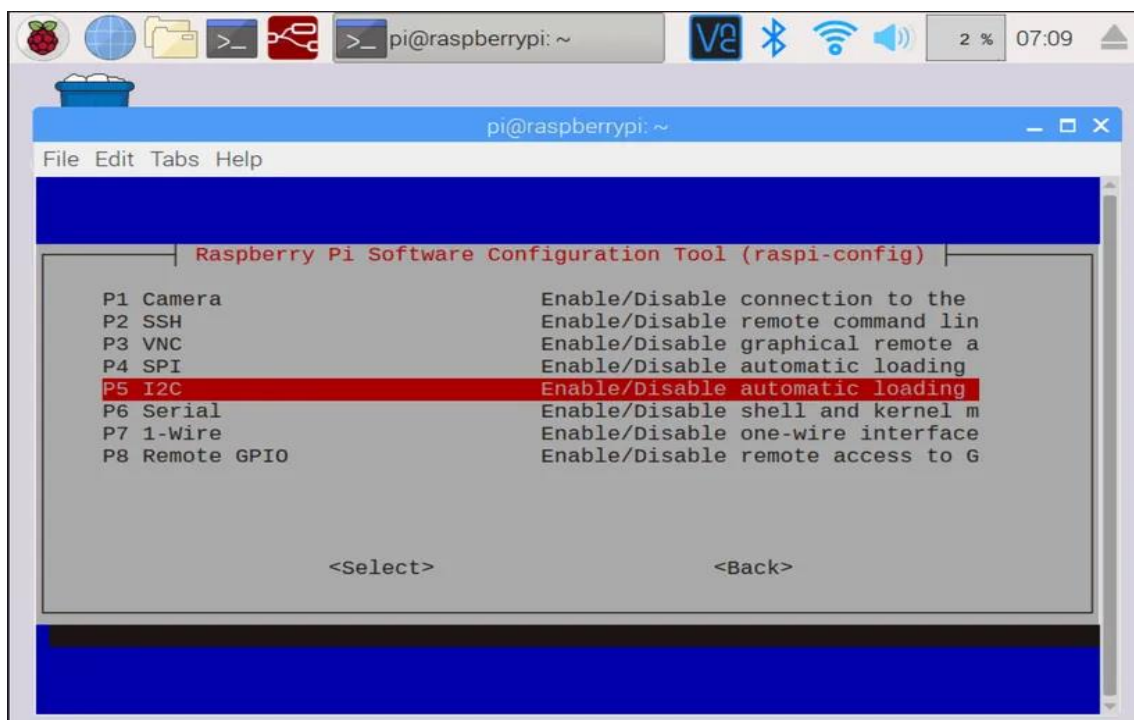
Then we need to make sure that the I2C protocol is enabled on your Raspberry Pi.

use *sudo raspi-config* command to enable *raspi-config* utility to enable I2C protocol on Raspberry Pi.

sudo raspi-config

After you click the Finish Button just reboot your pi with

sudo reboot



Step-2: Detect I2C Bus and Device address

Once the system is back you can check whether the I2C bus is active, I2C protocol supports multiple devices connected to the same bus, and it identifies each device by their hardware address. The *i2cdetect* command can be used to see what is connected and what the addresses are:

```
sudo i2cdetect -y 1
```

device address I2C Module connected with Raspberry PI is found at 0x27. The same Address is used in Python script to enable I2C communication between Raspberry Pi and I2C module.

Step-3: Installing rpi_lcd library

The easiest way to program this 16×2 I2C LCD display in Python is by using a dedicated library rpi_lcd.

Commands used:

```
sudo pip3 install rpi_lcd
```

```
sudo find /usr/local -name rpi_lcd 2> /dev/null
```

```
cd /usr/local/lib/python3.7/dist-packages/rpi_lcd
```

Code:

```
from signal import signal, SIGTERM, SIGHUP, pause
from rpi_lcd import LCD
lcd = LCD()
def safe_exit(signum, frame):
    exit(1)
try:
    signal(SIGTERM, safe_exit)
    signal(SIGHUP, safe_exit)
    lcd.text("Hello,", 1)
    lcd.text("Raspberry Pi!", 2)
    pause()
except KeyboardInterrupt:
    pass
finally:
    lcd.clear()
```

CHAPTER 5

PROPOSED SYSTEM

5.1 Algorithms for face detection and face recognition

The dlib library is arguably one of the most utilized packages for face recognition. A Python package appropriately named *face_recognition* wraps dlib's face recognition functions into a simple, easy to use API.

This dlib includes two face detection methods built into the library:

1. A **HOG + Linear SVM face detector** that is accurate and computationally efficient.
2. A **Max-Margin (MMOD) CNN face detector** that is both *highly accurate* and *very robust*, capable of detecting faces from varying viewing angles, lighting conditions, and occlusion.

Best of all, the MMOD face detector can run on an NVIDIA GPU, making it super fast! Since, CNN algorithm requires high computational power which can only be computed by higher raspberry pi models which are cost inefficient. So , we use HOG+Linear SVM algorithm here which also gives pretty accurate results to detect and recognize faces.

Face detection using HOG and Linear SVM

- Installing Dlib's Python package.
- A brief about Dlib's HOG and Linear SVM face detector.
- Exploring the directory structure and test data that we will use.

- Face detection with Dlib using HOG and Linear SVM.
 - Detecting faces in images.
 - Detecting faces in videos.

Installing Dlib's Python package:

```
pip install dlib
```

Dlib's HOG and Linear SVM Face Detector:

As the name suggests, it uses **Histogram of Oriented Gradients (HOG)** and Linear SVM classifier for face detection. It is also combined with an image pyramid and a sliding window detection scheme.

All of this suggests that the detection might be slow, mostly because of the sliding window scheme. This means that we may not get real-time performance when trying to detect faces in videos. Well, instead of speculating, we will in fact carry out face detection in both images and videos in this post, and see for ourselves.

Also, it is worth noting that it is really convenient to load the HOG and Linear SVM face detector. Dlib provides the

```
get_frontal_face_detector()
```

function which instantiates the face detector for us with just one line of code. We will see all of this in detail while coding our way through the post.

5.2 Setup Email Notifications

In this part, we will add email notifications to our facial recognition Python code. We are using mailgun service for sending email notification on face detection at the

door. This mailgun service is one of the efficient and simple services to use.

Mailgun

Mailgun (www.mailgun.com) is a transactional email service used by developers and information technology professionals to send, receive, and track emails using its powerful API. Developers can use it to send out transactional emails, monitor their performance, and optimize emails for higher engagement rates .

Transactional emails are “triggered/automated emails” that are usually sent to the customer after a specific action has been performed on an app or website. For example, an email sent to the customer after creating an account on an app can be called a transactional email.

Mailgun also supports various programming languages like Python, Java, Ruby, and the PHP framework to help developers set up Mailgun in a familiar programming language.

Mailgun is a secure tool. It doesn't share customer information with unauthorized third parties, and their server infrastructure is stored at top-tier data centers.

You can use the Mailgun email API for inbound mail processing, and even filter emails for spam by using a spam filter. However, if you're looking to connect your personal mail address to Mailgun, it's not recommended.

This is chiefly because Mailgun is designed specifically for developers instead of for general use.

Three Key Features of Mailgun

1. Email Validation

It lets you filter out inactive or undeliverable mail addresses before sending mail, and that keeps your bounce rates low. This way, your sender IP reputation isn't adversely affected.

There are two ways you can use the Mailgun API to validate your emails:

A. Real-Time Email Validation API

Mailgun's real-time email validation API identifies invalid emails at the time of capture (when an email address is inserted into your database). This way, no user will be able to insert an invalid mail address into Mailgun.

B. Bulk List Validation Tool

Mailgun has a bulk email validation tool that makes it super easy to clean out inactive emails from your list. All you have to do is upload your contacts as a CSV list, and Mailgun will take care of the rest — identifying and removing invalid accounts.

2. Burst Sending Feature

Email clients like Gmail and Outlook have daily limits on how many emails you can send out. And most of the time, these limits aren't enough for businesses that need to send thousands of emails in one go.

If you need to send a mass email to thousands of people on Gmail, it can take hours or even days for the mail to reach your last recipient. And that's if your account isn't blocked for spam before that!

This is why tools like Mailgun are specially designed for super-fast, high-volume email delivery. Mailgun is focused on sending out any transactional email as quickly as possible.

With Mailgun's Rapid Fire feature, you can choose to send your emails in bursts at predetermined times, or send out high volume emails around the clock.

Mailgun also features intelligent queue management features, which promises to send out 99% of emails within the first five minutes.

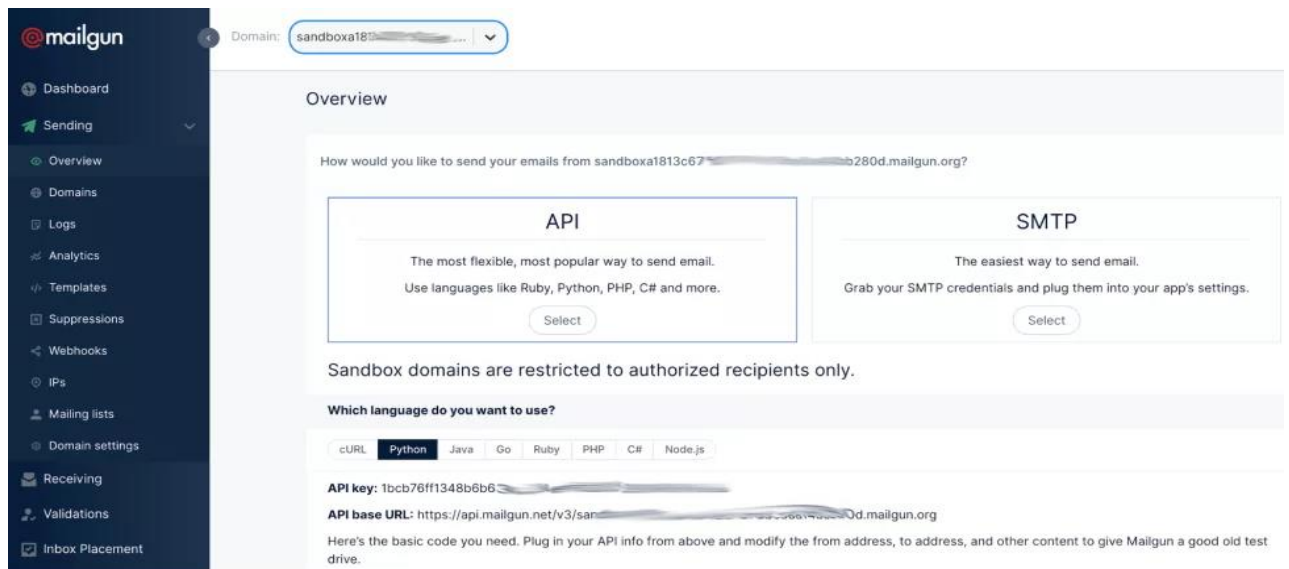
3. Email Performance Prediction

Mailgun has a feature called Inbox Placement, which can predict email deliverability issues and help in resolving them before sending out emails. This

way, there can be increase in recipient engagement and ROI from emails.

Setup Email Notifications for Raspberry Pi Facial Recognition

1. Navigate to mailgun.com in browser.
2. Create and/or Login to Mailgun account.
3. Navigate to sandbox domain and click API and then Python to reveal API credentials.



5. "https://api.mailgun.net/v3/YOUR_DOMAIN_NAME/messages" replace "YOUR_DOMAIN_NAME" with Mailgun domain.
6. Replace "YOUR_API_KEY" with API key from Mailgun.
7. Add email address from Mailgun account.

```
#function for setting up emails
def send_message(name):
    return requests.post(
        "https://api.mailgun.net/v3/sandbox7dcadbb6fb4d43138e56da272505abb8.mailgun.org/messages",
        auth=("api", "c85843aa0398b42874eb829491706292-53ce4923-ace23f4c"),
        files = [("attachment", ("image.jpg", open("image.jpg", "rb").read()))],
        data={"from": 'hello@example.com',
            "to": ["yagnavi0310@gmail.com", "sandhyaghanta10@gmail.com"],
            "subject": "You have a visitor",
            "html": "<html>" + name + " is at your door. </html>"})
```

8. Run the code **send_test_email.py**. If a status code 200 is received and “Message: Queued” message, check your email. This email may be delivered to Spam folder.

----- Forwarded message -----

From: <hello@example.com>

Date: Mon, 2 May 2022, 6:35 pm

Subject: You have a visitor

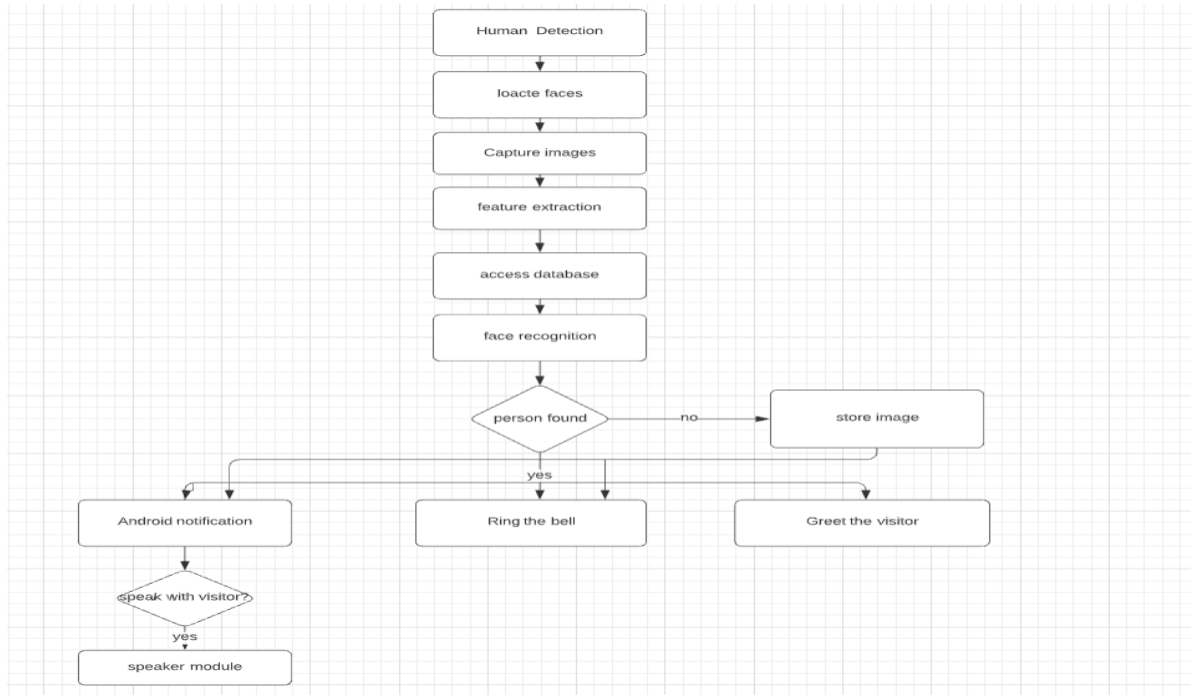
To: <yagnavi0310@gmail.com>, <sandhyaghanta10@gmail.com>

unknown is at your door.



5.3 SYSTEM DESIGN

Data Flow diagram of the system



1) Face Detection and Face Recognition

Face Detection

The first task that performed is detecting faces in the video stream and extract coordinates of the face for further processing and these coordinates are used to draw rectangles where face is found.

Face detection in dlib uses Histogram of Oriented Gradients (HOG) and Linear SVM classifier. It is also combined with an image pyramid and a sliding window detection scheme.

This is based on the HOG (Histogram of Oriented Gradients) feature descriptor with a linear SVM machine learning algorithm to perform face detection. The basic idea of HOG is dividing the image into small connected cells. It computes histogram for each cell and Combines small histograms into one

histogram which is a final feature vector. These feature vectors are called face embeddings.

Face Recognition

The libraries we used to implement facial recognition are

1) OpenCv 2) dlib 3) Face_recognition

The dlib library contains our implementation of deep metric learning which is used to construct our face embeddings used for the actual recognition process. To build face recognition system, we first performed face detection, extract face embeddings from each face in the dataset using deep learning and saved it into a .pickles file.

we run the same network over the images captured from the video stream at the door and convert those faces into face embeddings and see if these vector values matches or are similar with the embeddings of stored dataset's embeddings. If they match, it gives the name of the stored images dataset to the owner. If they don't match it identified the visitor as an unknown person and alerts the user.

2) Alert Generation :

Mailgun provides APIs (as well as others like Logging and Mailing List APIs) for external clients – when it comes to email, ease and efficiency are key. All email APIs work in a similar way. When you use them, you can provide more consistent email messaging.

So, whenever a person arrives at the door, the system starts detecting the face and recognizing it. After identification of Human face the system clicks the image of the person standing before the door. Buzzer is activated and will be on for few seconds. If person is Unknown(not registered in database) , the mailgun sends an email to the specified owners with the image along with the message saying “Unknown person is at the door!” . If the person is known, it send a mail saying “{Name of that person} is at the door!”.

CHAPTER 6

IMPLEMENTATION

Code:

Train_model.py

```
#!/usr/bin/python

# import the necessary packages
from imutils import paths
import face_recognition
#import argparse
import pickle
import cv2
import os

# our images are located in the dataset folder.
print("[INFO] start processing faces...")
imagePaths = list(paths.list_images("dataset"))

# initialize the list of known encodings and known names
knownEncodings = []
knownNames = []

# loop over the image paths
for (i, imagePath) in enumerate(imagePaths):
```

```
# extract the person name from the image path
print("[INFO] processing image {}/{}".format(i + 1, len(imagePaths)))
name = imagePath.split(os.path.sep)[-2]

# loading input image and convert it from RGB (Opencv ordering) to dlib
ordering (RGB)
image = cv2.imread(imagePath)
rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# detect the (x, y)-coordinates of the bounding boxes
# corresponding to each face in the input image
boxes = face_recognition.face_locations(rgb,
model="hog")

# computing the facial embedding for the face
encodings = face_recognition.face_encodings(rgb, boxes)

for encoding in encodings:
# add each encoding + name to our set of known names and encodings
knownEncodings.append(encoding)
knownNames.append(name)

# dump the facial encodings + names to disk
print("[INFO] serializing encodings...")
data = {"encodings": knownEncodings, "names": knownNames}
f = open("encodings.pickle", "wb")
f.write(pickle.dumps(data))
```

```
f.close()
```

Face.py

```
# import the necessary packages
from imutils.video import VideoStream
from imutils.video import FPS
import face_recognition
import imutils
import pickle
import time
import cv2
import requests

#Initialize 'currentname' to trigger only when a new person is identified.
currentname = "unknown"

#Determine faces from encodings.pickle file model created from
train_model.py
encodingsP = "encodings.pickle"

#use this xml file
cascade = "haarcascade_frontalface_default.xml"

#function for setting up emails
def send_message(name):
    return requests.post(
```



```

"https://api.mailgun.net/v3/sandbox7dcadbb6fb4d43138e56da272505abb8.mailg
un.org/messages",

auth=("api", "c85843aa0398b42874eb829491706292-53ce4923-ace23f4c"),

files = [("attachment", ("image.jpg", open("image.jpg", "rb").read()))],

data={"from": 'hello@example.com',

"to": ["yagnavi0310@gmail.com", "sandhyaghanta10@gmail.com"],

"subject": "You have a visitor",

"html": "<html> + name + " is at your door. </html>"}))

# load the known faces and embeddings along with OpenCV's Haar
# cascade for face detection

print("[INFO] loading encodings + face detector...")

data = pickle.loads(open(encodingsP, "rb").read())

detector = cv2.CascadeClassifier(cascade)

# initialize the video stream and allow the camera sensor to warm up
print("[INFO] starting video stream...")

vs = VideoStream(src=0).start()

# vs = VideoStream(usePiCamera=True).start()

time.sleep(2.0)

# start the FPS counter

fps = FPS().start()

# loop over frames from the video file stream
while True:

# grab the frame from the threaded video stream and resize it

```

```
# to 500px (to speedup processing)

frame = vs.read()

frame = imutils.resize(frame, width=500)

# convert the input frame from (1) BGR to grayscale (for face
# detection) and (2) from BGR to RGB (for face recognition)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

# detect faces in the grayscale frame
rects = detector.detectMultiScale(gray, scaleFactor=1.1,
minNeighbors=5, minSize=(30, 30),
flags=cv2.CASCADE_SCALE_IMAGE)

# OpenCV returns bounding box coordinates in (x, y, w, h) order
# but we need them in (top, right, bottom, left) order, so we
# need to do a bit of reordering
boxes = [(y, x + w, y + h, x) for (x, y, w, h) in rects]

# compute the facial embeddings for each face bounding box
encodings = face_recognition.face_encodings(rgb, boxes)
names = []

# loop over the facial embeddings
for encoding in encodings:
    # attempt to match each face in the input image to our known
    # encodings
```

```

matches = face_recognition.compare_faces(data["encodings"],
encoding)

name = "Unknown"

# check to see if we have found a match
if True in matches:

# find the indexes of all matched faces then initialize a
# dictionary to count the total number of times each face
# was matched

matchedIdxs = [i for (i, b) in enumerate(matches) if b]

counts = {}

# loop over the matched indexes and maintain a count for
# each recognized face face
for i in matchedIdxs:
name = data["names"][i]
counts[name] = counts.get(name, 0) + 1

# determine the recognized face with the largest number
# of votes (note: in the event of an unlikely tie Python
# will select first entry in the dictionary)
name = max(counts, key=counts.get)

#If someone in your dataset is identified, print their name on the screen
if currentname != name:
currentname = name
print(currentname)

```

```
#Take a picture to send in the email

img_name = "image.jpg"
cv2.imwrite(img_name, frame)
print('Taking a picture.')

#Now send me an email to let me know who is at the door
request = send_message(name)
print ('Status Code: '+format(request.status_code)) #200 status code means
email sent successfully
else:
    print("unknown")
    img_name = "image.jpg"
    cv2.imwrite(img_name, frame)
    print('Taking a picture.')

#Now send me an email to let me know who is at the door
request = send_message("unknown")
print ('Status Code: '+format(request.status_code)) #200 status code means
email sent successfully

# update the list of names
names.append(name)

# loop over the recognized faces
for ((top, right, bottom, left), name) in zip(boxes, names):
```

```
# draw the predicted face name on the image - color is in BGR
cv2.rectangle(frame, (left, top), (right, bottom),
(0, 255, 225), 2)
y = top - 15 if top - 15 > 15 else top + 15
cv2.putText(frame, name, (left, y), cv2.FONT_HERSHEY_SIMPLEX,
.8, (0, 255, 255), 2)

# display the image to our screen
cv2.imshow("Facial Recognition is Running", frame)
key = cv2.waitKey(1) & 0xFF

# if the `q` key was pressed, break from the loop
if key == ord("q"):
    break

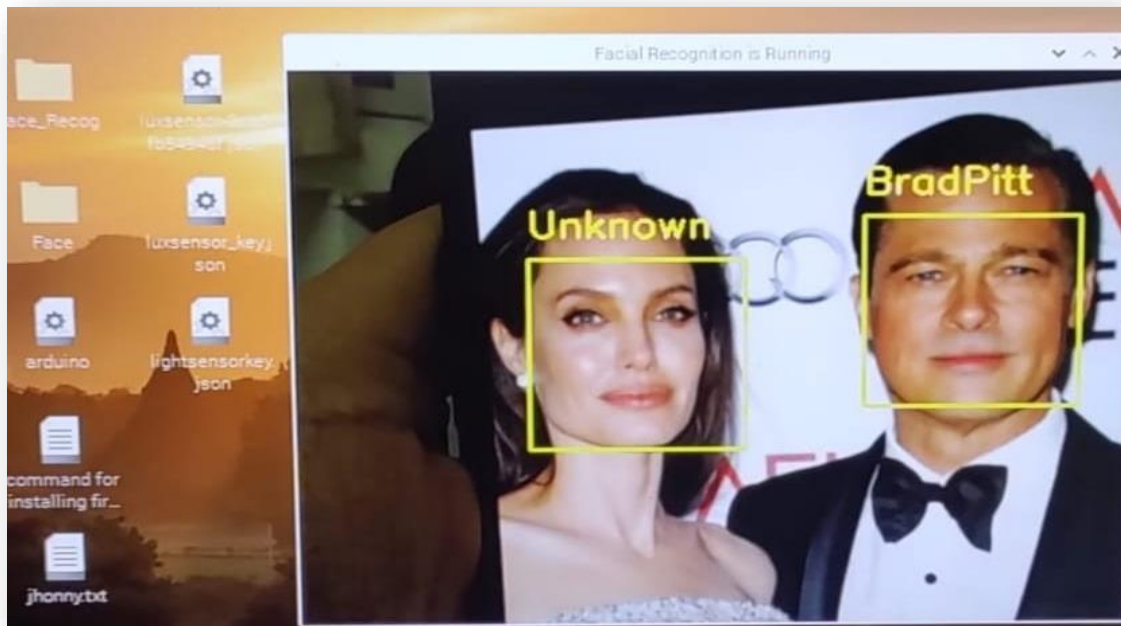
# update the FPS counter
fps.update()

# stop the timer and display FPS information
fps.stop()
print("[INFO] elapsed time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))

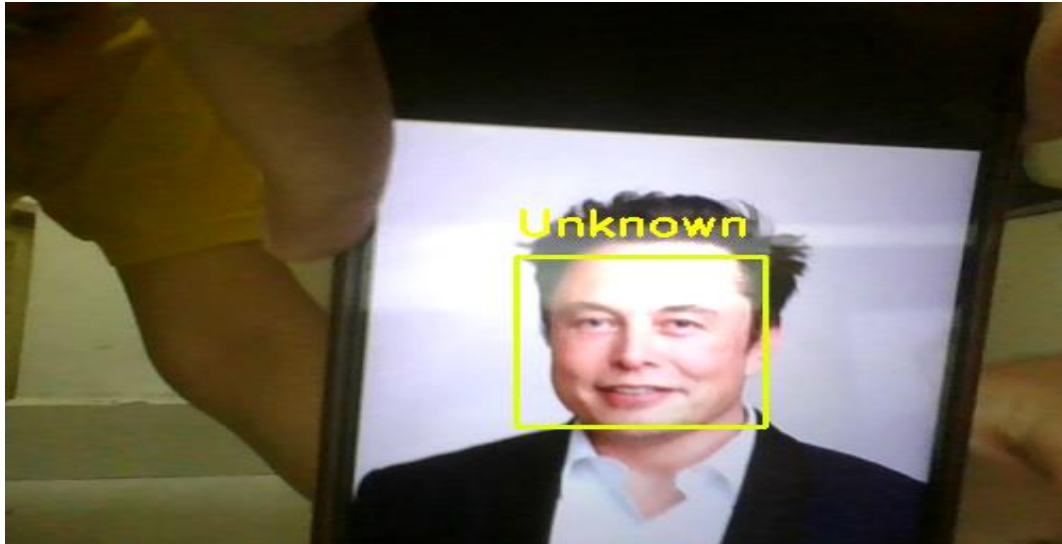
# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```

OUTPUT

Face recognition



Alert



----- Forwarded message -----

From: <hello@example.com>

Date: Mon, 2 May 2022, 6:35 pm

Subject: You have a visitor

To: <yagnavi0310@gmail.com>, <sandhyaghanta10@gmail.com>

unknown is at your door.



sandhya ghanta

to me ▾



----- Forwarded message -----

From: <hello@example.com>

Date: Fri, 29 Apr 2022, 10:45 am

Subject: You have a visitor

To: <yagnavi0310@gmail.com>, <sandhyaghanta10@gmail.com>

Yagnavi is at your door.

