Faculty of Computer Applications & Information Technology
***i*MSc(IT) Programme, Ahmedabad.**

Embedded Systems 2 Project (Sem – V)

"SMART DOORBELL USING RASPBERRY PI"

BY:~

Sakshi Bhavsar A-08

Mishri Dave A-12

Aishwarya Shah A-55

Under the guidance of Prof. Poonam Dang

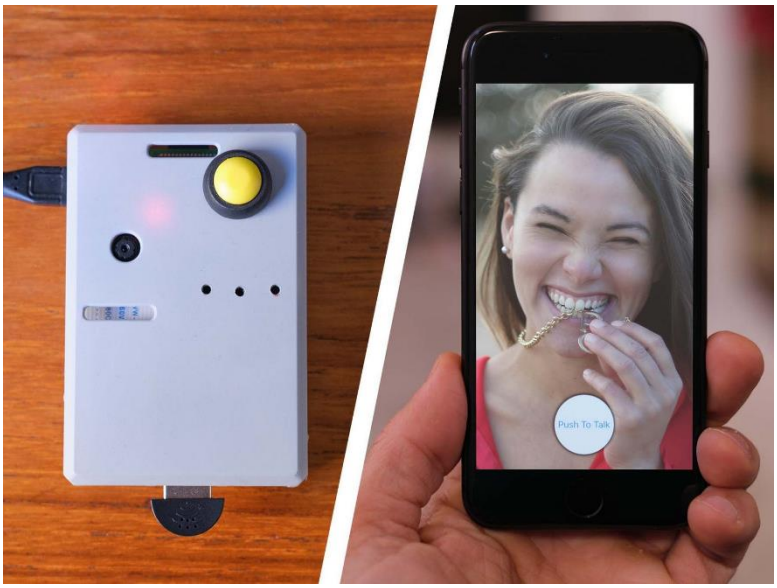
ACKNOWLEDGEMENT

We are over helmed in all humbleness and gratefulness to acknowledge our depth to GLS UNIVERSITY who has helped us to put these ideas, well above the level of simplicity and into something concrete. We would like to express our special thanks of gratitude to our guide- Prof. Poonam Dang who gave us the golden opportunity to do this wonderful project on the topic Smart Doorbell, which also helped our group in doing a lot of Research and we came to know about so many new things. We are really thankful to them. Any attempt at any level can ‘t be satisfactorily completed without the support and guidance of my parents and friends.

INDEX

INTRODUCTION

Our system connects WiFi-enabled Android devices with firebase server using Raspberry Pi and enables user to answer the door when the doorbell is pressed.There will be two-way communication i.e. on both sides -user and arriver audio and videos can be seen. Moreover the door can be opened and closed with the help of mobile itself.



OBJECTIVE

Starting from small houses to huge industries, surveillance plays very vital role to fulfill safety aspects as Burglary and theft have always been a problem. In daily life, people have the need to know, Identify of a visitor who comes to their homes, regardless of they are there at that time.Sometimes there occurs a situation where disable people are alone at home or they stay alone; could not get up from bed; old age people who stay alone and are more vulnerable to risks need something extra to boost comfort and security at their main doors.Thus, the main objective is to render comfort and security to user.

COMPONENTS/HARDWARE PART

1.Raspberry pi 3B+ module

The Raspberry Pi 3 Model B+ is the final revision in the Raspberry Pi 3 range.

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- Extended 40-pin GPIO header
- Full-size HDMI
- 4 USB 2.0 ports
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- 4-pole stereo output and composite video port
- Micro SD port for loading your operating system and storing data
- 5V/2.5A DC power input
- Power-over-Ethernet (PoE) support (requires separate PoE HAT)

2. CAMERA MODULE

The Pi camera module is a portable light weight camera that supports Raspberry Pi. It communicates with Pi using the MIPI camera serial interface protocol. The Pi camera module when purchased comes along with a ribbon cable, this cable has to be connected to the CSI (Camera Serial Interface) port of the Pi**.** This port can be found near the HDMI port.

3. MINI USB MICROPHONE

It's the world's **smallest USB microphone**! ... This plug-and-play Mini USB Microphone needs no driver, and its USB interface means you can use it with any computer, laptop, SBC, etc

4. SERVO MOTOR

The control of Raspberry Pi servo motors is very simple and thanks to the small size and weight they are used in many areas.Unlike stepper motors, servomotors can be controlled with a single GPIO. Because servo motors use feedback to determine the position of the shaft, you can control that position very precisely. As a result, servo motors are used to control the position of objects, rotate objects, move legs, arms or hands of robots, move sensors etc. with high precision.

5. OTHER COMPONENTS

JUMPER WIRES

BLUETOOTH SPEAKER

PUSH BUTTON

RASPBERRY PI CASE

ASSEMBLY

STEP 1:

Connect camera module and enable it in raspberry pi using configuration->interface.

STEP 2:

Connect and enable mini-usb microphone and put its percentage to 100 using alsa mixer.

STEP 3:

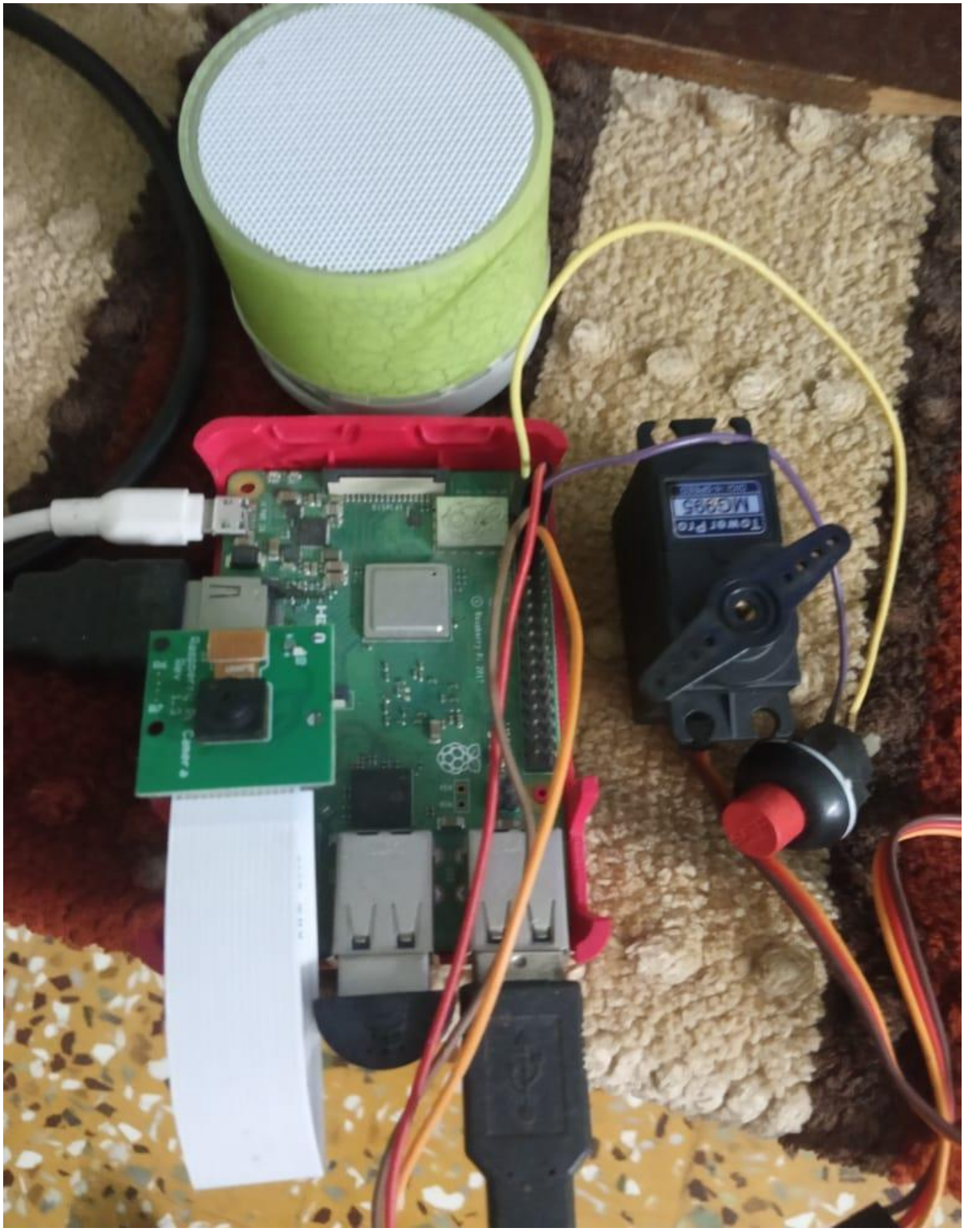
Connect bluetooth speaker to raspberry pi by connecting it clicking on bluetooth icon and selecting it as audio output device via right clicking on sound icon

STEP 4:

Servo motor has three wires - orange,red and brown. Red wire indicates power and needs to be connected to 5v pin on raspberry pi i.e first on top right[pin-2] as indicated in image. Brown wire is meant for grounding and needs to be connected to 5th pin from left side[pin-9-GND] . The orange wire is for connecting to gpio pin and in our case we will connect this wire to gpio pin 17[pin-11]; 6th from left.

STEP 5:

To connect push button attach one wire to GND pin on raspberry ; in our case pin-6 i.e. 3rd pin from the top right. Attach another wire to gpio pin 2; i.e. second from top left[pin-3]. We will use this push button to trigger all the actions.


Tower Pro
MG995

CODING:

1.DOORBELL.PY [FOR RUNNING DOORBELL]

(IT MAKES RASPBERRY PI FUNCTION AS DOORBELL INFINITELY UNTIL ITS POWER IS PUT OFF)

```
#############
# User Parameters
#############

# Doorbell pin
DOORBELL_PIN = 2
# Number of seconds to keep the call active
DOORBELL_SCREEN_ACTIVE_S = 300
# ID of the JITSI meeting room
JITSI_ID = None  # If None, the program generates a random UUID
# JITSI_ID = "hackershackdoorbellexample"
# Path to the SFX file
RING_SFX_PATH = "tune.wav"  # If None, no sound effect plays
# RING_SFX_PATH = "/home/pi/ring.wav"
# Enables email notifications
ENABLE_EMAIL = True
# Email you want to send the notification from (only works with
gmail)
FROM_EMAIL = 'aishu072001@gmail.com'
# You can generate an app password here to avoid storing your
password in plain text
# this should also come from an environment variable
# https://support.google.com/accounts/answer/185833?hl=en
FROM_EMAIL_PASSWORD = '1234'
# Email you want to send the update to
TO_EMAIL = 'aishu072001@gmail.com'



#############
# Program
#############

import time
import os
import signal
import subprocess
import smtplib
import uuid

from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.image import MIMEImage

try:
    import RPi.GPIO as GPIO
except RuntimeError:
```

```
    print("Error importing RPi.GPIO. This is probably because you
need superuser. Try running again with 'sudo'.")


def show_screen():
    os.system("tvservice -p")
    os.system("xset dpms force on")


def hide_screen():
    os.system("tvservice -o")


def send_email_notification(chat_url):
    if ENABLE_EMAIL:
        sender = EmailSender(FROM_EMAIL, FROM_EMAIL_PASSWORD)
        email = Email(
            sender,
            'Video Doorbell',
            'Notification: A visitor is waiting',
            'A video doorbell caller is waiting on the virtual
meeting room.  You can open the door at: http://192.168.1.6:5000/
Meet them at %s' % chat_url
        )
        email.send(TO_EMAIL)


def ring_doorbell(pin):
    SoundEffect(RING_SFX_PATH).play()

    chat_id = JITSI_ID if JITSI_ID else str(uuid.uuid4())
    video_chat = VideoChat(chat_id)
    send_email_notification(video_chat.get_chat_url())

    show_screen()

    video_chat.start()
    time.sleep(DOORBELL_SCREEN_ACTIVE_S)
    video_chat.end()

    hide_screen()


class SoundEffect:
    def __init__(self, filepath):
        self.filepath = filepath

    def play(self):
        if self.filepath:
            subprocess.Popen(["aplay", self.filepath])


class VideoChat:
    def __init__(self, chat_id):
        self.chat_id = chat_id
        self._process = None
```

```
    def get_chat_url(self):
        return "http://meet.jit.si/%s" % self.chat_id

    def start(self):
        if not self._process and self.chat_id:
            self._process = subprocess.Popen(["chromium-browser", "-
kiosk", self.get_chat_url()])
        else:
            print("Can't start video chat -- already started or
missing chat id")

    def end(self):
        if self._process:
            os.kill(self._process.pid, signal.SIGTERM)


class EmailSender:
    def __init__(self, email, password):
        self.email = email
        self.password = password


class Email:
    def __init__(self, sender, subject, preamble, body):
        self.sender = sender
        self.subject = subject
        self.preamble = preamble
        self.body = body

    def send(self, to_email):
        msgRoot = MIMEMultipart('related')
        msgRoot['Subject'] = self.subject
        msgRoot['From'] = self.sender.email
        msgRoot['To'] = to_email
        msgRoot.preamble = self.preamble

        msgAlternative = MIMEMultipart('alternative')
        msgRoot.attach(msgAlternative)
        msgText = MIMEText(self.body)
        msgAlternative.attach(msgText)

        smtp = smtplib.SMTP('smtp.gmail.com', 587)
        smtp.starttls()
        smtp.login(self.sender.email, self.sender.password)
        smtp.sendmail(self.sender.email, to_email,
msgRoot.as_string())
        smtp.quit()


class Doorbell:
    def __init__(self, doorbell_button_pin):
        self._doorbell_button_pin = doorbell_button_pin

    def run(self):
        try:
```

```
            print("Starting Doorbell...")
            hide_screen()
            self._setup_gpio()
            print("Waiting for doorbell rings...")
            self._wait_forever()

        except KeyboardInterrupt:
            print("Safely shutting down...")

        finally:
            self._cleanup()

    def _wait_forever(self):
        while True:
            time.sleep(0.1)

    def _setup_gpio(self):
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(self._doorbell_button_pin, GPIO.IN,
pull_up_down=GPIO.PUD_DOWN)
        GPIO.add_event_detect(self._doorbell_button_pin,
GPIO.RISING, callback=ring_doorbell, bouncetime=2000)

    def _cleanup(self):
        GPIO.cleanup(self._doorbell_button_pin)
        show_screen()


if __name__ == "__main__":
    doorbell = Doorbell(DOORBELL_PIN)
    doorbell.run()
```

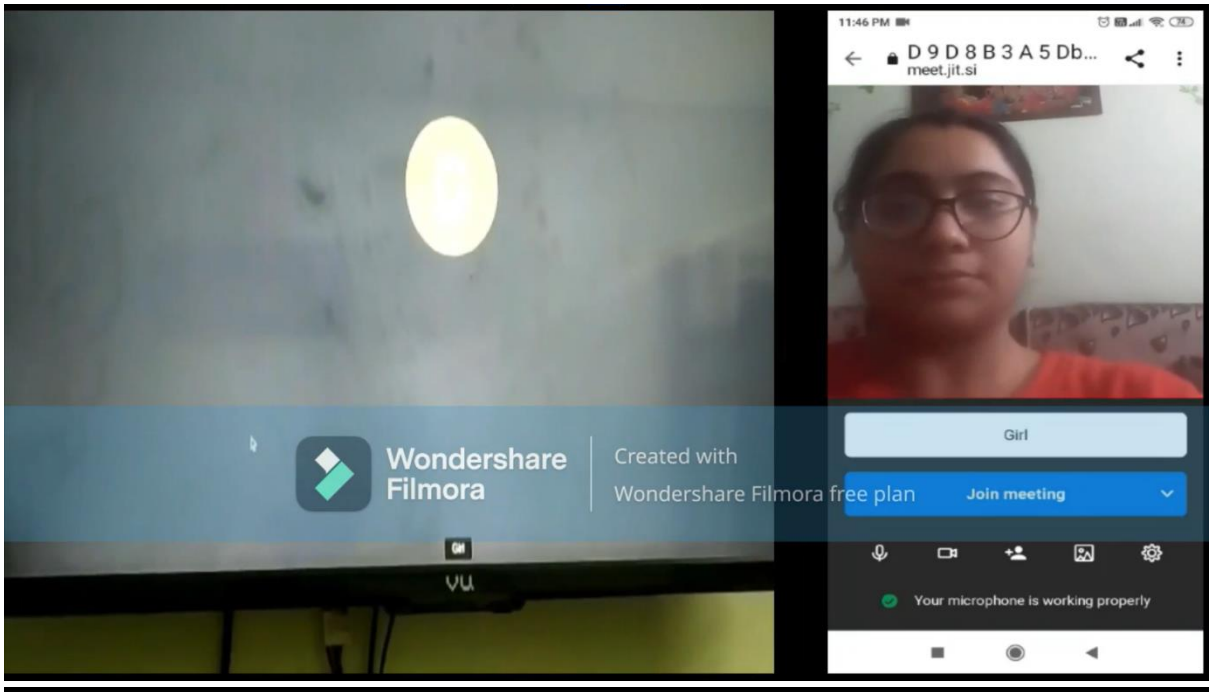
1.SERVO2.PY [FOR RUNNING SERVO MOTOR]

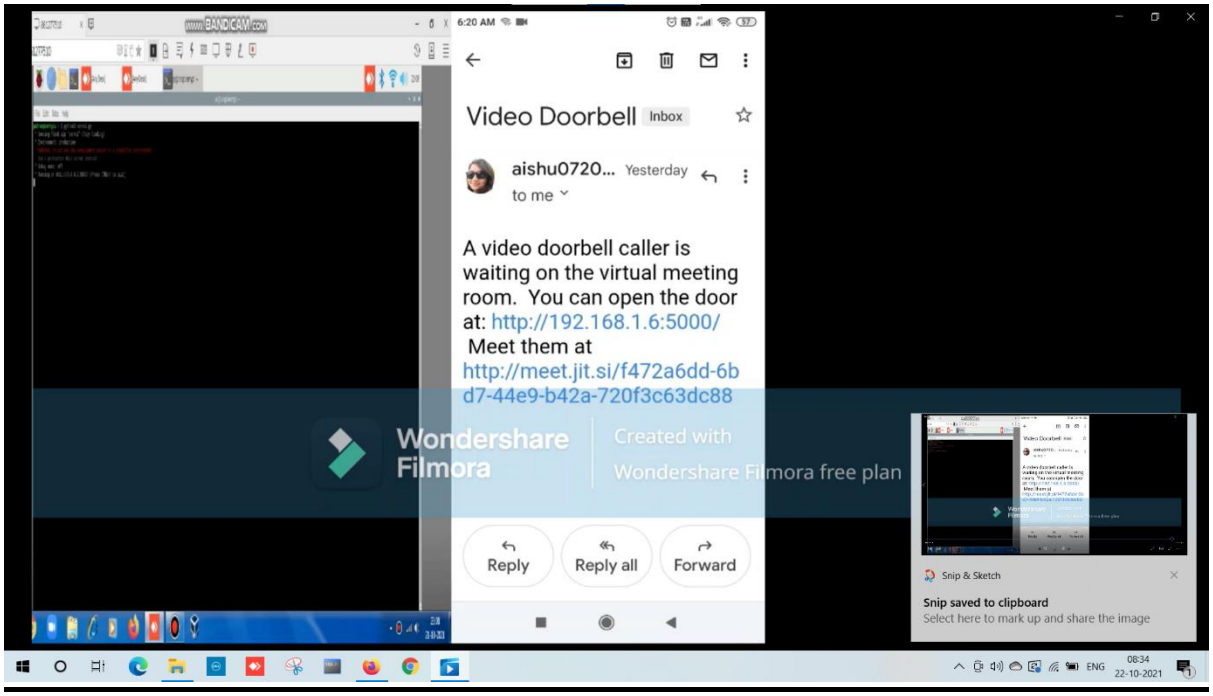
(IT WILL OPEN OR CLOSE THE DOOR BASED ON POSITION OF WEBPAGE SLIDER)

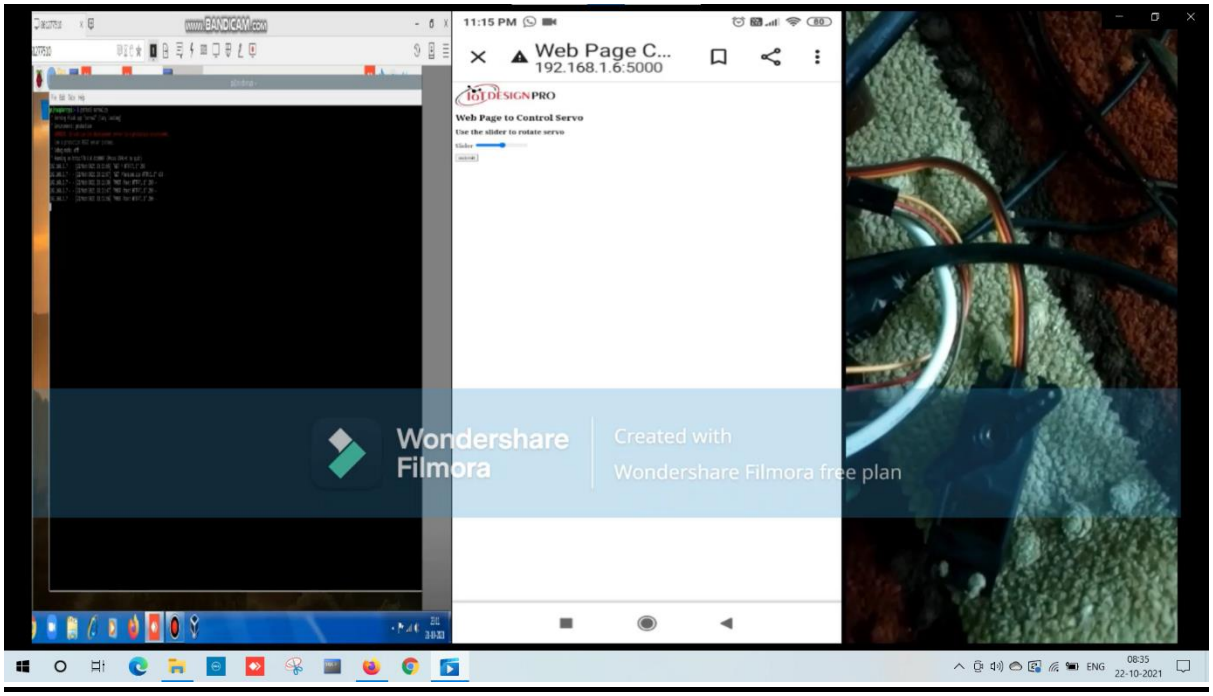
```
from flask import Flask, render_template_string, request   # Importing the
Flask modules
import RPi.GPIO as GPIO     # Importing the GPIO library
from time import sleep      # Import sleep module from time library
servo_pin = 17 # GPIO Pin where sero is connected
GPIO.setmode(GPIO.BCM)      # Define the Pin numbering type. Here we are
using BCM type
# Defing Servo Pin as output pin
GPIO.setup(servo_pin, GPIO.OUT)
p = GPIO.PWM(servo_pin, 100)  # PWM channel at 50 Hz frequency
p.start(0) # Zero duty cycle initially
app = Flask(__name__)
#HTML Code
TPL = '''
<html>
```

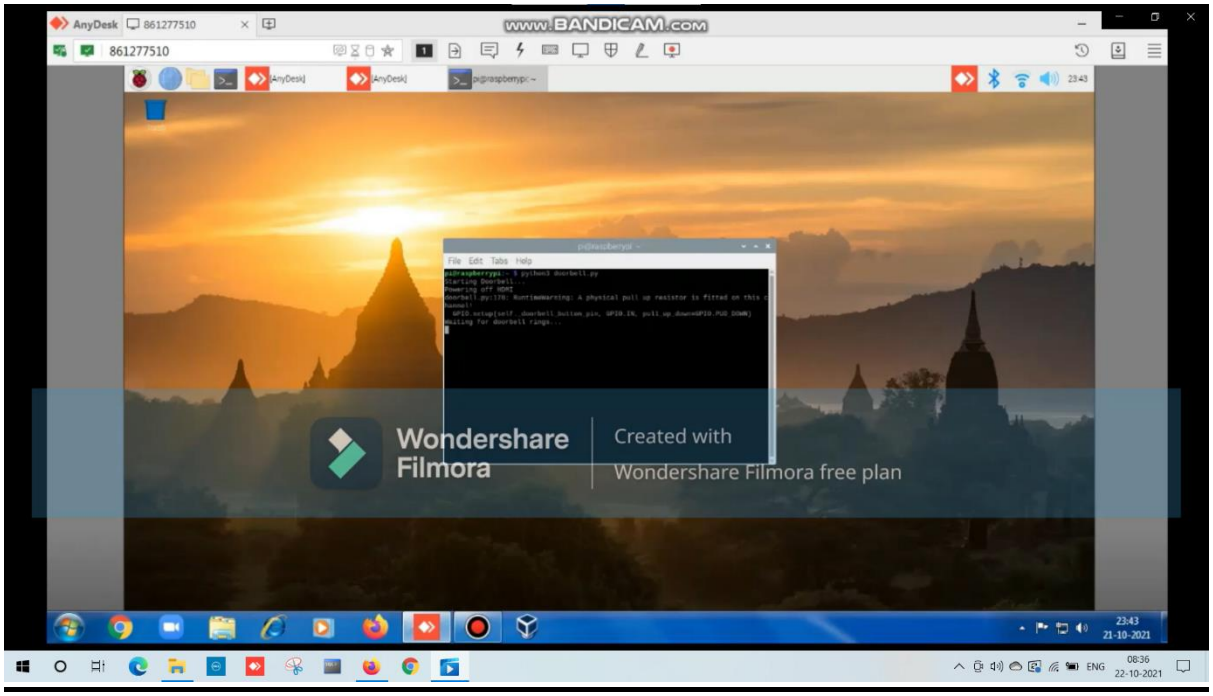
```
     <img
src="https://iotdesignpro.com/sites/default/files/Iot%20Design%20Pro%20Logo
_0.png" alt="">
    <head><title>Web Page Controlled Servo</title></head>
    <body>
    <h2> Web Page to Control Servo</h2>
        <form method="POST" action="test">
            <h3> Use the slider to rotate servo  </h3>
            <p>Slider   <input type="range" min="1" max="12.5"
name="slider" /> </p>
            <input type="submit" value="submit" />
        </form>
    </body>
</html>

'''
@app.route("/")
def home():
    return render_template_string(TPL)
@app.route("/test", methods=["POST"])
def test():
    # Get slider Values
    slider = request.form["slider"]
    # Change duty cycle
    p.ChangeDutyCycle(float(slider))
    sleep(1)
    # Pause the servo
    p.ChangeDutyCycle(0)
    return render_template_string(TPL)
# Run the app on the local development server
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=False)
```

WORKING:




6:20 AM
Video Doorbell
Inbox
aishu0720... Yesterday
to me
A video doorbell caller is waiting on the virtual meeting room. You can open the door at: http://192.168.1.6:5000/
Meet them at http://meet.jit.si/f472a6dd-6bd7-44e9-b42a-720f3c63dc88
Reply
Reply all
Forward
Wondershare Filmora
Created with Wondershare Filmora free plan
Snip & Sketch
Snip saved to clipboard
Select here to mark up and share the image
ENG
08:34
22-10-2021


11:15 PM
Web Page C...
192.168.1.6:5000
IOT DESIGN PRO
Web Page to Control Servo
Use the slider to rotate servo
Wondershare Filmora
Created with Wondershare Filmora free plan
ENG
08:35
22-10-2021



CONCLUSION:

Hence as this technology offers a better safety there has been a boom in use of this kind of devices. Our system will combine many of the existing technologies into one so as to develop a better solution. A new feature has been added i.e. automatic door opening and locking.