

IOT

1

ZEAL
INSTITUTES
www.zealeducation.com

Assignment: I

Aim:

- Design & implement IOT system using Arduino Uno / Raspberry Pi using 'Ultrasonic Sensor and Servo Motor' such as 'Door Opener in home automation'.

Objective:

- To connect & implement Raspberry Pi, Ultrasonic Sensor using Servo Motor used as Door Opener System.

Theory:

* Raspberry Pi :-

- USB ports: these are used to connect a mouse and keyboard. You can also connect other components, such as USB drive.
- SD card slot - you can slot the SD card in here. This is where the operating system software & your files are stored.
- Ethernet port - this is used to connect a mouse & keyboard. You can also connect other components, such as USB drive.

- Audio jack - you can connect headphones or speakers here.
- HDMI port - this is where you connect the monitor (or projector) that you're using to display output from Raspberry Pi, if your monitor has speakers, you can allow them to hear sound.
- Micro USB power connector - this where you connect a power supply. You should always do this last, after you have connected all your other components.
- GPIO ports - these allow you to connect electronic components such as LEDs & buttons to Raspberry pi.

Set up your SD card.

- If you have an SD card that doesn't have Raspberry Pi OS on it yet, or if you want to reset your Raspberry Pi, you can easily install Raspberry Pi OS yourself.
- The Raspberry Pi OS operating system via Raspberry Pi images.

- Using the Raspberry Pi OS via the Raspberry Pi imager, it is the easiest way to install Raspberry Pi OS on your SD card.
- + Interfaces:
 - You can link devices & components to your Raspberry Pi using a lot of different types of connections on or off, so that your Raspberry Pi recognises that you've linked something to it via a particular type of connection.
 - Camera - enable the Raspberry Pi Camera Module.
 - SSH - allow remote access to your Raspberry Pi desktop from another computer using VNC.
 - SPI - enable the SPI GPIO pins.
 - I2C - enable the I2C GPIO pins
 - Serial - enable the serial (Rx-Tx) GPIO pins.
 - 1-Wire - enable the 1-Wire GPIO pins
 - Remote GPIO - allows access to your raspberry pi's GPIO pins from another computer.

+ Performance:

- If you need to do so for the particular project you want to work on

You can change the performance settings of your raspberry pi in this tab.

+ Conclusion:

- We implemented IOT system using Arduino Uno | Raspberry Pi .

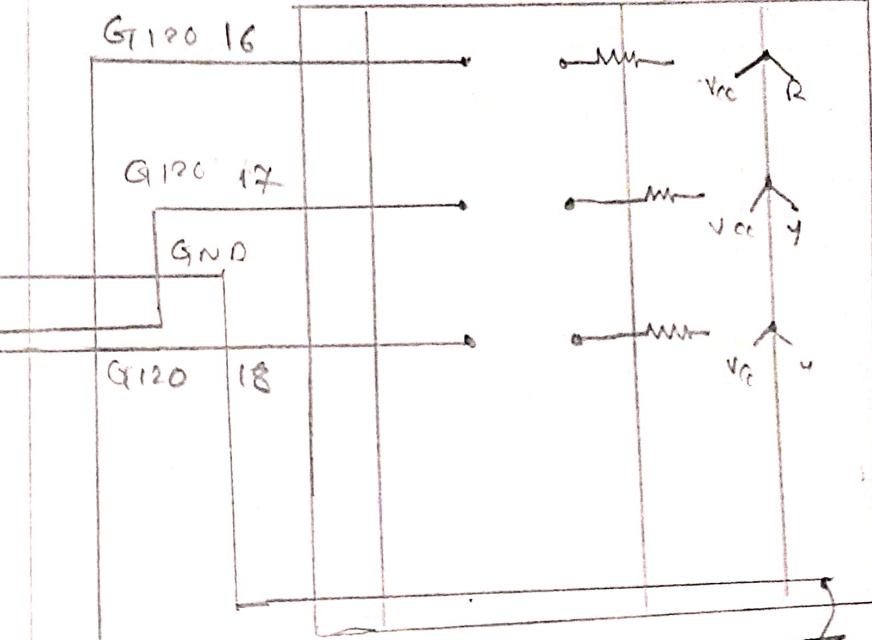
~~Ques~~

Diagram:

(3)

Circuit Diagram

•1	•2
•3	•4
•5	•6
•7	•8
•9	•10
•11	•12
•13	•14
•15	•16
•17	•18
•19	•20
•21	•22
•23	•24
•25	•26
•27	•28
•29	•30
•31	•32
•33	•34
•35	•36
•37	•38
•39	•40



* 36 = GP10 16 (R)

* 11 = GP20 17 (4)

* 12 = GP20 18 (G)

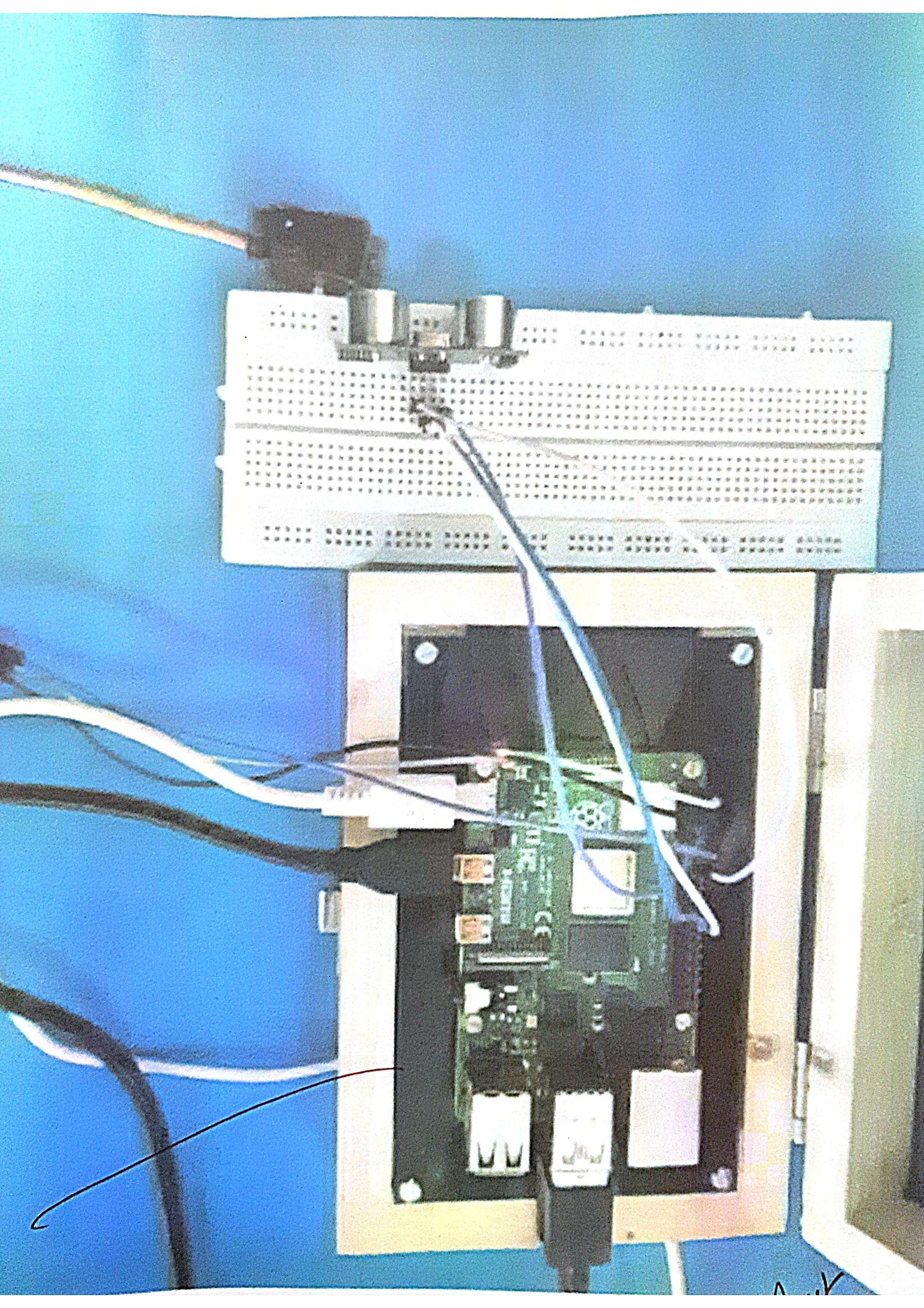
Practical No:1

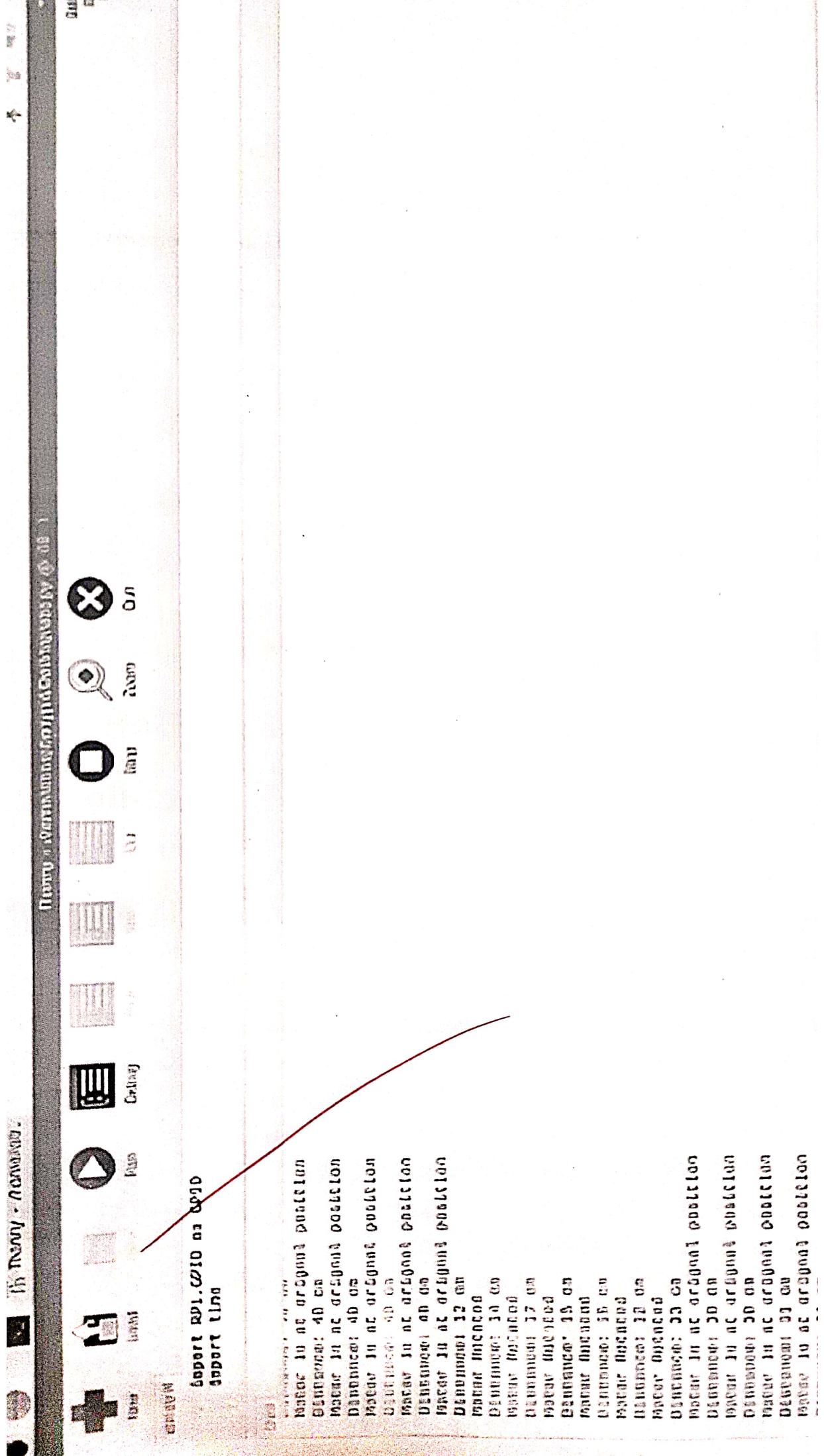
Program:

```
import RPi.GPIO as GPIO
import time
# Setup GPIO
GPIO.setwarnings(False)
TRIG = 11
ECHO = 8
servoPIN = 18
GPIO.setmode(GPIO.BCM)
# Ultrasonic sensor pin initialization
GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN) # Corrected line: Set ECHO pin as input
# Servo motor pin initialization
GPIO.setup(servoPIN, GPIO.OUT)
servo = GPIO.PWM(servoPIN, 50)
servo.start(2.5)
try:
    while True:
        GPIO.output(TRIG, False)
        time.sleep(0.000002)
        GPIO.output(TRIG, True)
        time.sleep(0.00001)
        GPIO.output(TRIG, False)
        startTime = time.time()
```

```
stopTime = time.time()
while GPIO.input(ECHO) == 0:
    startTime = time.time()
    while GPIO.input(ECHO) == 1:
        stopTime = time.time()
    GPIO.output(TRIG, True)
    timeElapsed = stopTime - startTime
    distance = (timeElapsed * 34300) / 2
    distance = int(distance)
    print("Distance: {} cm".format(distance))
if distance <= 20:
    duty_cycle = 12.5 # Adjust this value for desired servo position
    servo.ChangeDutyCycle(duty_cycle)
    print("Motor Rotated")
    time.sleep(0.1)
else:
    duty_cycle = 2.5 # Adjust this value for desired servo position
    servo.ChangeDutyCycle(duty_cycle)
    print("Motor is at original position")
    time.sleep(0.1)
except KeyboardInterrupt:
    servo.stop()
GPIO.cleanup()
```







Assignment: 2

+ Aim:

- Design and implement parameter monitoring IOT system keeping records on cloud such as environment humidity & temperature monitoring.

+ Objective:

- To design & implement environmental humidity & temp monitoring.

Theory:

- We are using the DHT11 sensor for sending temp. and humidity data to thing speak using arduino & esp8266. By this method, we can monitor other DHT11 sensor temperature and humidity data on the internet using the thing speak IOT server.

+ Component Required:

- Arduino Uno
- ESP8266 wifi module
- DHT11 sensor
- Breadboard
- Jumper wires

+ Step 1: Thingspeak setup for temperature and humidity monitoring;

- for executing your channel by thingspeak, you first need to sign up on a thingspeak. In case if you already have an account on thingspeak, just sign in using your id & password.

Step 2: Create a Channel for Your Data;

- Once you sign in after your account verification. Create a new channel by clicking "New Channel" button.
- Enter the name of your data "Temp." in Field 1 and "Humidity" in field 2.
- After this, click on save channel button to save your details.

Step 3: API Key;

- To send data to Thingspeak, we need a unique API key, which we will use later in our code to upload our sensor data to ThingSpeak website.
- Click on "API keys" button to get your unique API key for uploading your sensor data.
- Now copy your "Write API key". We will use this API key in our code.
- Upload it in Arduino UNO. If you successfully upload your program.

To enhance your IoT-based project on environment humidity and temperature monitoring DHT 11 Sector (Sensor) and esp8265, here are some additional points that can help extend your project and demonstration a deeper understanding.

- 1). Implement data visualization:
 - ↳ Graph customization.
 - ↳ multifield data analysis.
- 2) Add alert and notification
 - ↳ sms or email alerts
 - ↳ Integration with IFTTT
- 3). Optimize power consumption
 - ↳ Deep sleep mode for ESP8265
 - ↳ Power Efficiency in IoT Deployment
- 4) Data Backup and Redundancy:
 - ↳ Data Storage on SD card
 - ↳ Cloud Integration with google Sheets

P.T.O

Conclusion;

- This is we have program successfully thing speak.
we get temperature on

~~✓ Sanket~~

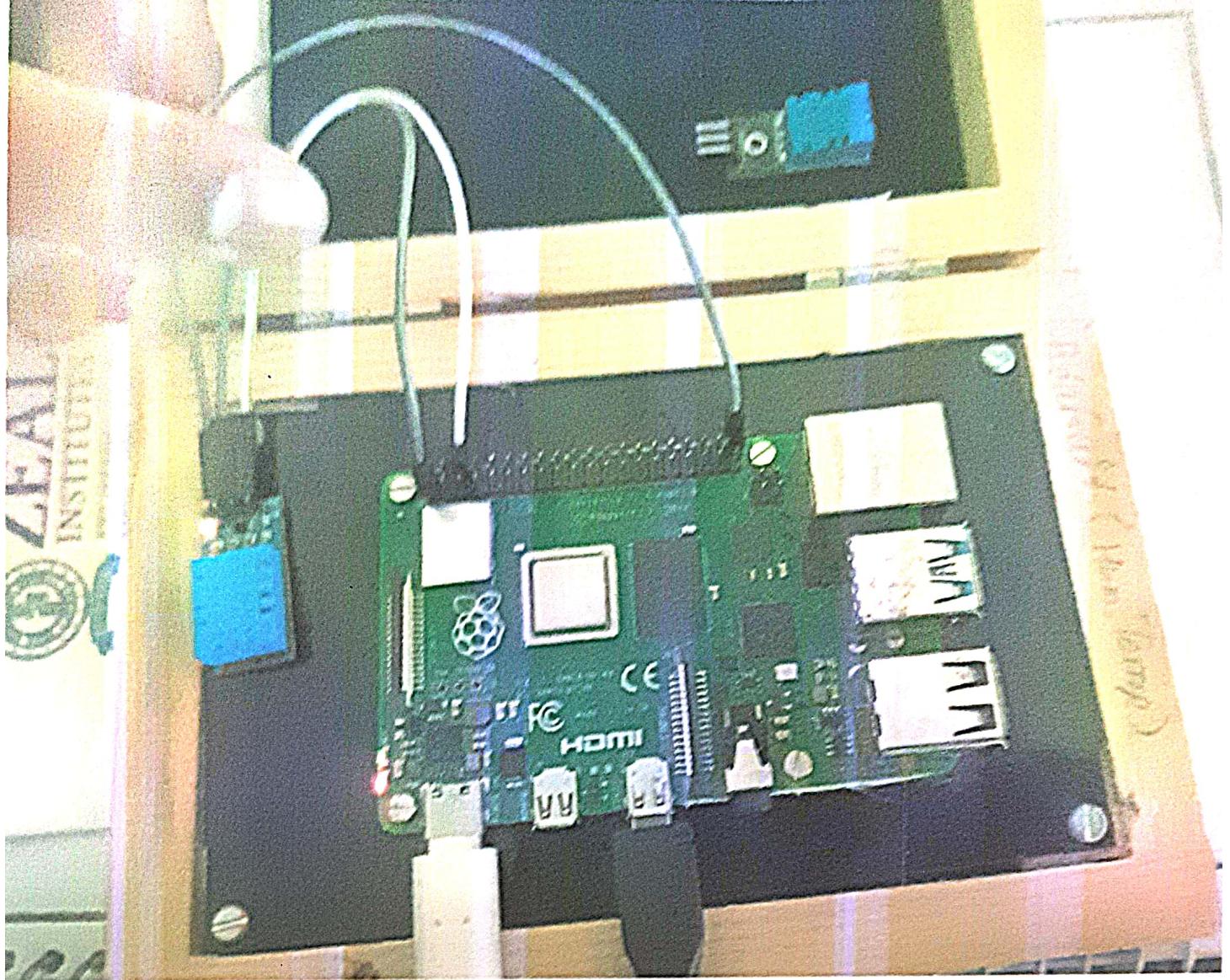
Practical No:- 02

Code:-

```
import Adafruit_DHT
import os, sys
from time import sleep
import requests
import RPi.GPIO as GPIO
baseURL = 'https://api.thingspeak.com/update?api_key=WRITE API KEY'
SensorPin = 4
while True:
    try:
        print("Program is Running")
        humi, temp = Adafruit_DHT.read(Adafruit_DHT.DHT11, SensorPin)
        if (humi > 0) and (temp > 0):
            print(humi)
            print(temp)
            print("Data received")
            x = '{}{{}}{{}}{{}}'.format(baseURL, '&field1=', temp, '&field2=', humi);
            y = requests.post(x)
            print(y.status_code)
            print(x);
            sleep(5)

    except:
        print("Data not received")
        sleep(5)
```

A handwritten red ink signature, likely belonging to the author or a witness, is written over a curved red line. The signature is fluid and cursive, appearing to read "J. Alvaro".



```
Shell X
Program is Running
Humidity: 8.0%
Temperature: 8.8°C
Data received
Response Code: 200
Request URL: http://192.168.1.11:8080/api/temperature
Program is Running
Humidity: 8.0%
Temperature: 8.8°C
Data received
Response Code: 200
Request URL: http://192.168.1.11:8080/api/temperature
Program is Running
Humidity: 8.0%
Temperature: 8.8°C
Data received
Response Code: 200
Request URL: http://192.168.1.11:8080/api/temperature
```

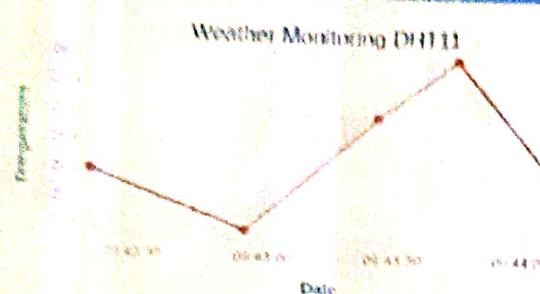
Great

Created: 3 minutes ago

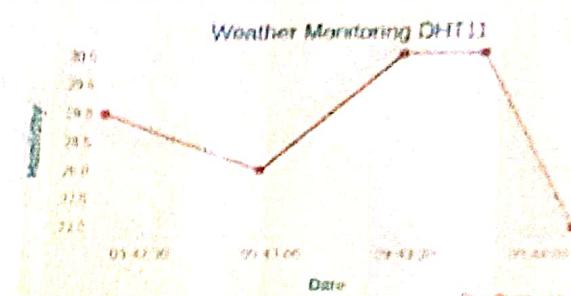
Last entry: less than a minute ago

Entries: 5

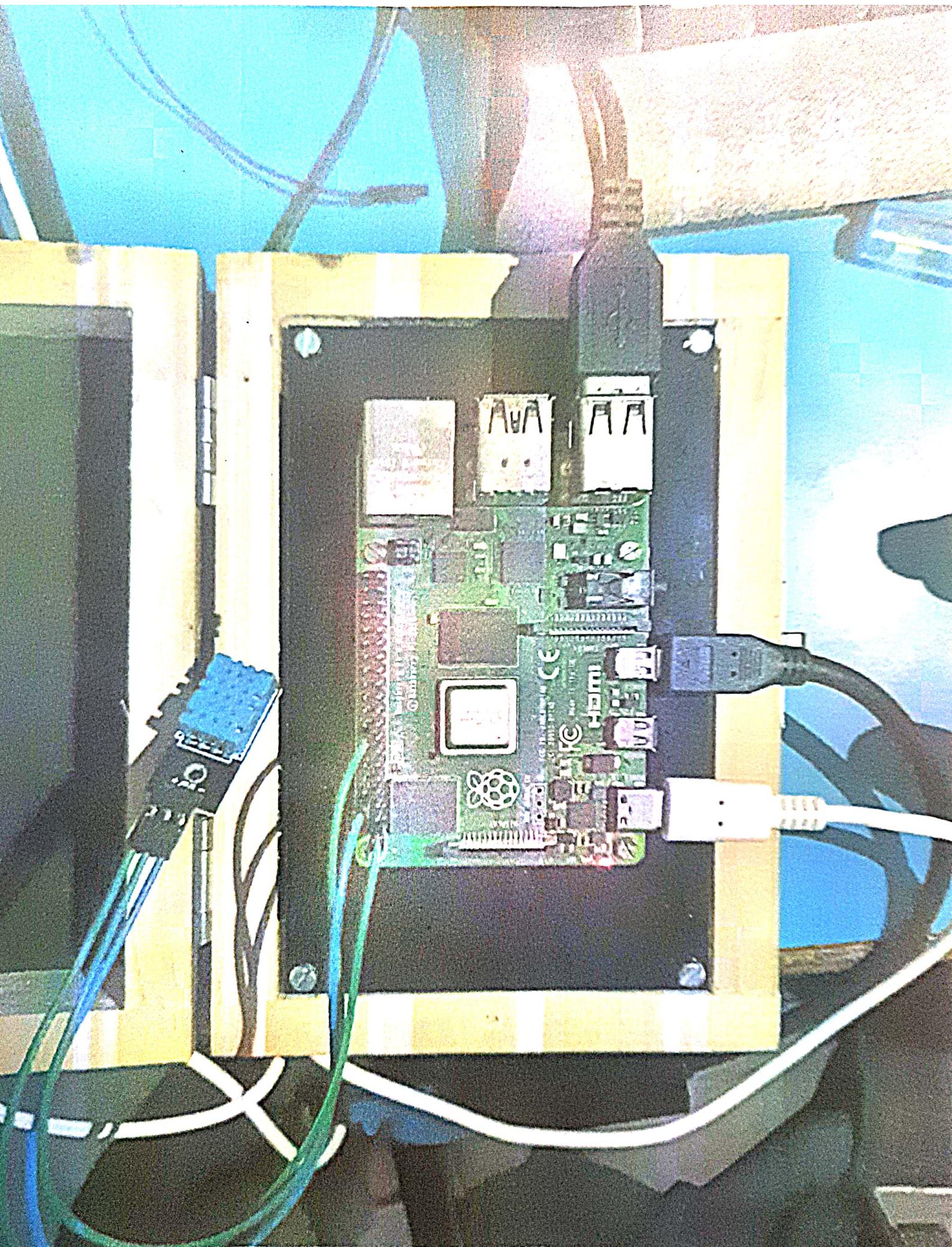
Field 1 chart



Field 2 chart



✓  ✓



Assignment: 3

Aim:

- Design and implement real time monitoring system using android phone (Blynk App). Such as 'Soil parameter monitoring'.

Objective:

- Real time Monitoring of Temperature on Android Phone using Soil Parameter.

Parameters:

- Raspberry Pi, BLYNK, Soil Moisture Sensor.

Theory:

- Agriculture has been the most imp. practice very beginning of human civilization. It has seen many iterations of development in technology with time.
- In last few decades changing weather conditions, increase in global temp. and pollution, has led to abnormal environment conditions like Raining.
- Good control over Environmental parameters like temperature, humidity and moisture plays imp. role in growth of plant. Temp. affects many of plant activities such as pollination and germination. etc.
- It is observed that higher temperature

respiration rate increases results in reduction of sugar contents of fruits and vegetables. At lower temperatures affects many of plant activities such is slowed down. Till date many methods have come into existence where water can be limitedly consumed.

- Wireless sensor networks is also called as wireless sensors & actor network, are distributed sparsely autonomous sensors to monitor physical or environmental conditions as temp, pressure, sound, moisture etc.
- Hence, the wireless sensor network works

Theory:

- Scalability & Monitoring (Systems):
- The wireless Sensor Network (WSN) utilized can be scaled depending on the farm's size. Small farms might require a few sensors to monitor key areas, while larger farms can deploy more sensors to cover a vast area, across the entire field.



Energy Efficiency:

- Raspberry Pi and other embedded systems are energy efficient devices capable of running continuously with low power consumption improving both yield and quality.

IOT Integration:

- The use of IOT in agriculture allows data monitoring for real time communication between devices like soil sensors and raspberry pi which can be accessed via Blynk app on Android integration nor only improves the precision of farming but also helps in reducing the manual effort and resources spent on monitoring field conditions.

Fast effective and easy to implement:

- The system, using Raspberry Pi and off the shelf of sensors such as soil moisture sensors, such as soil moisture sensor, offers cost effective solution that can be easily implemented by farmers with minimal technical knowledge, the Blynk app provides a user friendly interface to monitor and control farm activities in real time.

Conclusion:

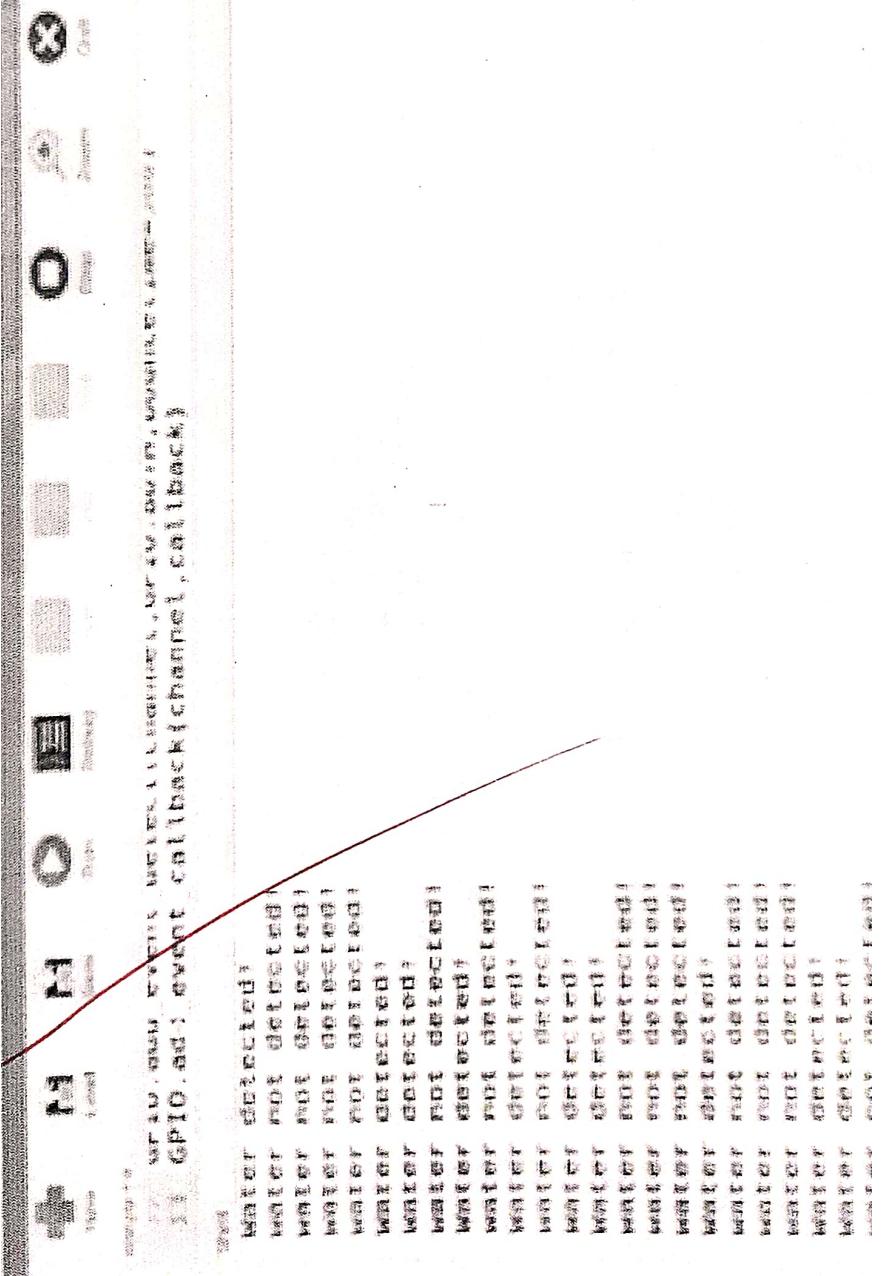
- The real time monitoring system using raspberry pi and soil moisture sensor provides and efficient for optimizing irrigation agriculture productivity.

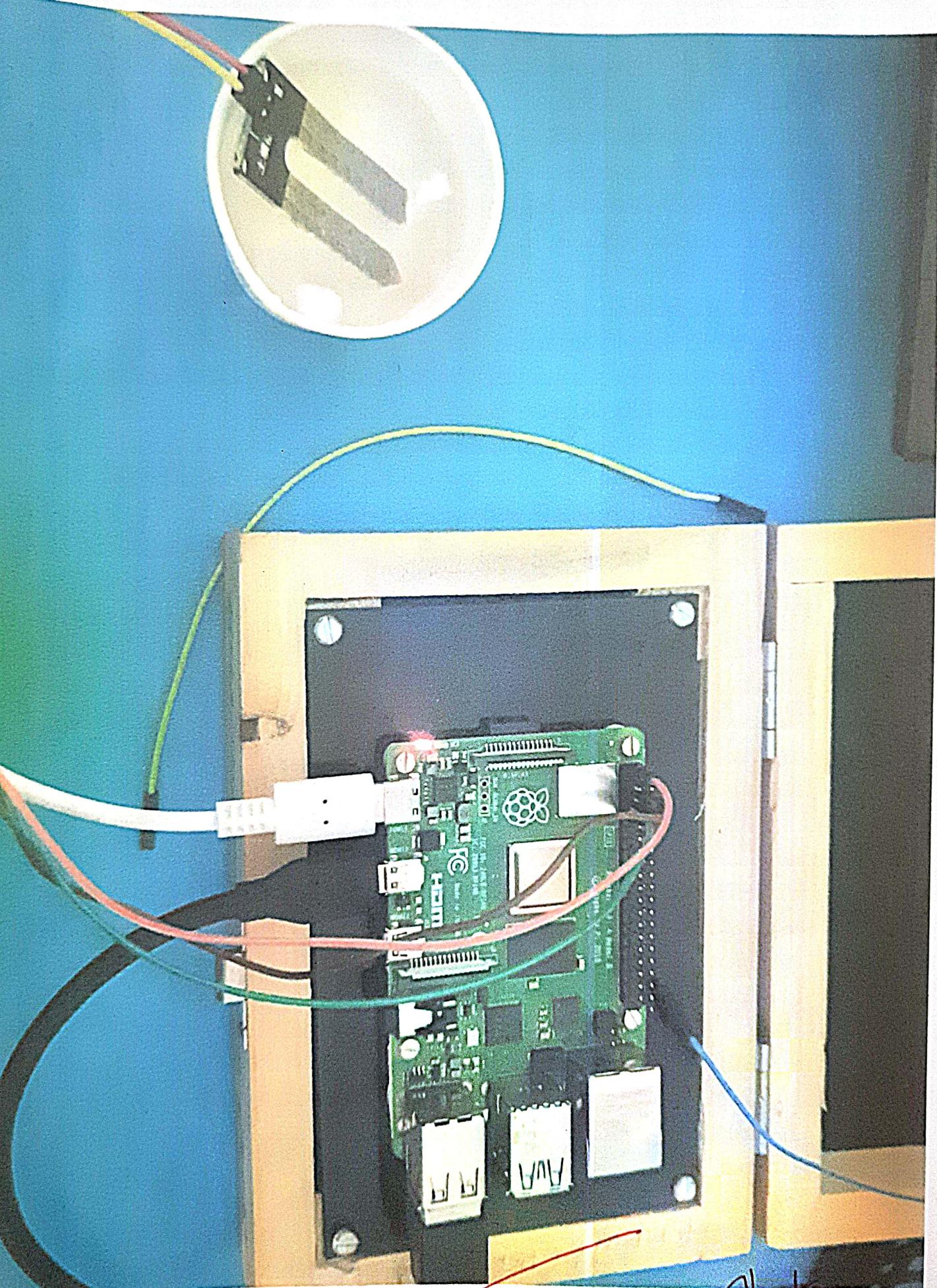
~~Blast~~

Practical No:3

Program:

```
import RPi.GPIO as GPIO
import time
#GPIO SETUP
channel =4
GPIO.setmode (GPIO.BCM)
GPIO.setup(channel, GPIO.IN)
def callback (channel):
    if GPIO.input (channel):
        print("Water detected!")
    else:
        print("water not detected!")
GPIO.add_event_detect (channel, GPIO.BOTH, bouncetime=300)
GPIO.add_event_callback(channel, callback)
while True:
    time.sleep(0)
```





Assignment: 4

Aim:

- Design and implement IOT system for one of application like:
- Traffic Application
- Medical / Health Application
- Social application etc.

Objective:

- Using for the manage Traffic light Signal.

+ Parameters:

- LED RGB,
- Breadboard
- Raspberry Pi - Kit.

Theory:

- First you need to understand how each comp. is connected: An LED requires GPIO pin 36, 11, 12. A ground pin 3 of Raspberry Pi.
- Each comp. requires its own individual GPIO pin, but components can share a ground pin.

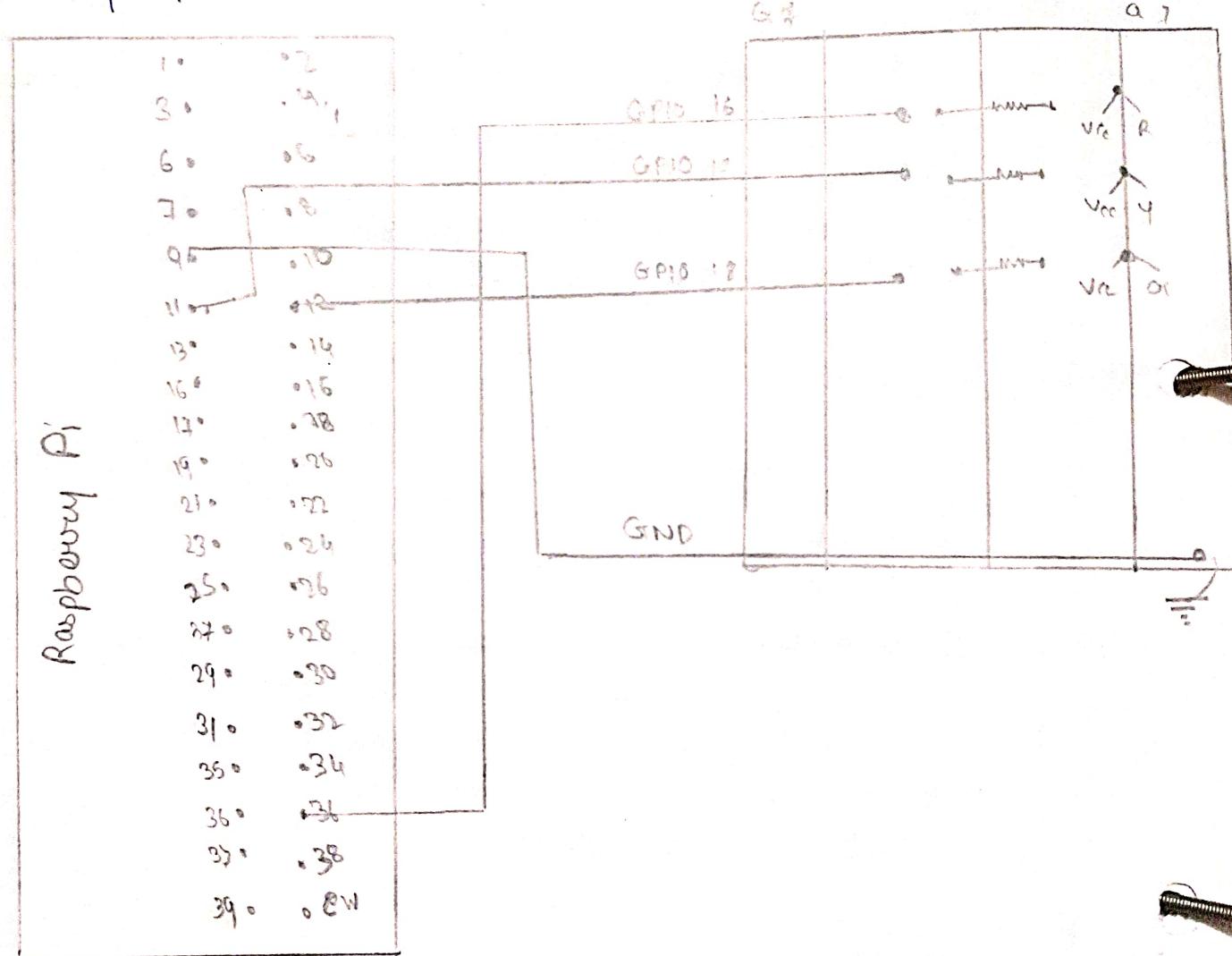
Component Pin:	Button	Red LED	Yellow LED	Green LED
GPIO Pin:	21	16	18	11

Breadboard.

Diagram:

Circuit Diagram:

To get started, you'll need to place all components on breadboard & connect them to appro. GPIO pins on Raspberry Pi.



- 36 = GPIO 10 (R)
- 11 = GPIO 19 (Y)
- 12 = GPIO 18 (A)

- # Use of GPIO Pins:
- The GPIO is general purpose input - output pins on Raspberry Pi, serve as an interface to connect and control whole electronic components such as LED. In this case diff. GPIO pins are used to control, the Red, yellow & Green LEDs.
 - Simulating a typical traffic light pattern.

Overview of Traffic System:

- A traffic light system plays role in managing traffic flow at intersection, by using Raspberry - Pi, to control LED's representing traffic lights.
- You can simulate how & actual traffic light works, this system can easily be scaled up and deal - world traffic.



Diagram:

Real Time Control (with IoT Integration):

- This traffic signal system could be further enhanced by adding real time control and monitoring capabilities.
- Hence, this is time control.

Scalability:

- This basic setup - can be extended by incorporating over traffic lights.
- pedestrian crossing, or even remote monitoring via the cloud using IoT technologies, these can send real time data to a remote server, allowing authorities to manage multiple, intersecting from a central location.

Future Enhancements:

- System can be expanded by integrating cameras to detect volume of traffic/signal

Conclusion:

- This practical demonstrates how IoT based systems can be used to control traffic signal providing foundation over/more advanced traffic management system.

~~Ques~~

```
1 from gpiozero import LED  
2 import time  
3 R=LED(16)  
4 Y=LED(18)  
5 G=LED(17)  
6 while(1):  
7     R.on()  
8     time.sleep(3)  
9     R.off()  
10    Y.on()  
11    time.sleep(2)  
12    Y.off()  
13    G.on()  
14    time.sleep(3)  
15    G.off()
```

