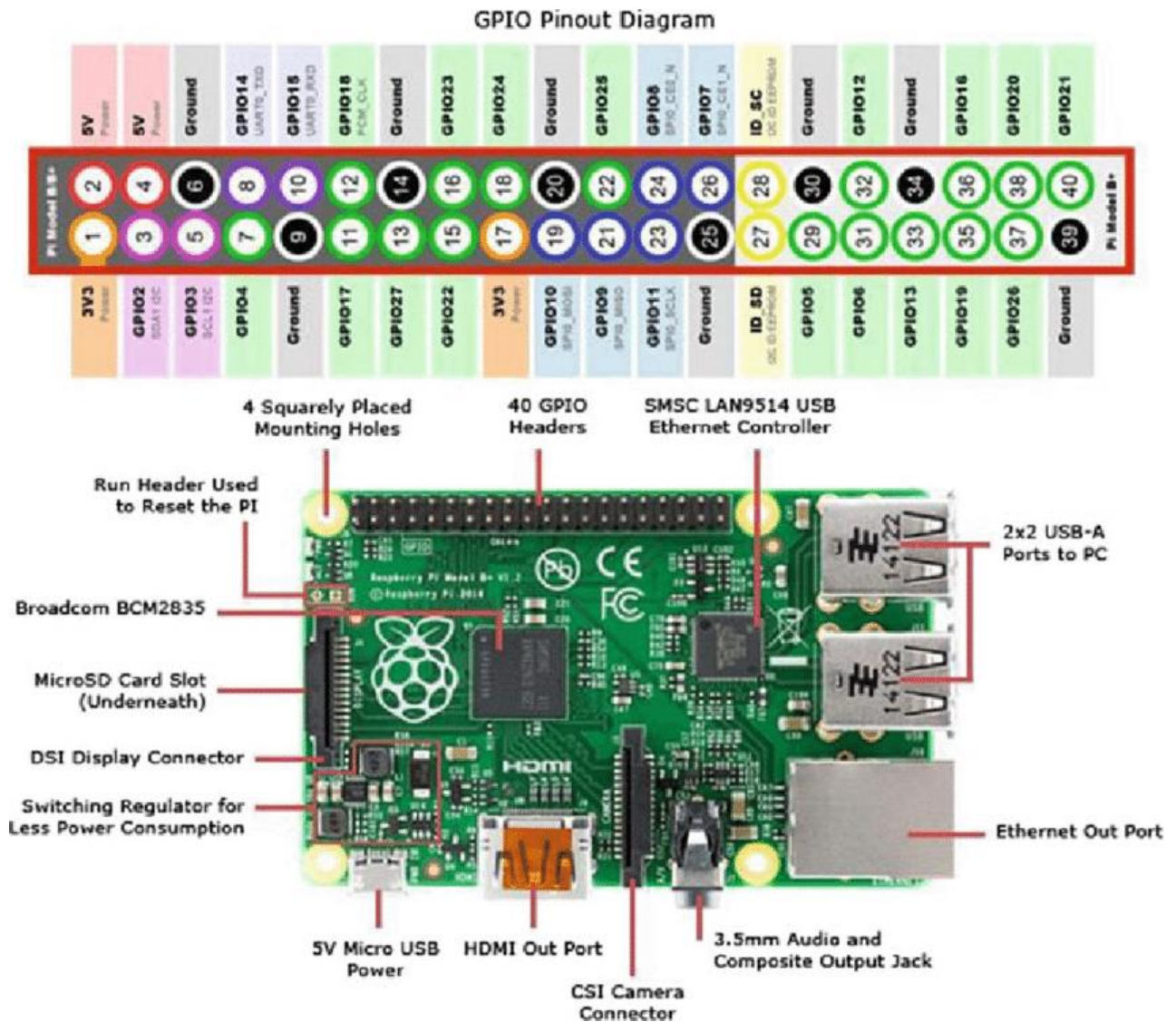


# INTERNET OF THINGS

## PRACTICAL 1: Starting Raspbian OS, Familiarizing with Raspberry Pi Components and Interface, Connecting to Ethernet, Monitor, USB.



Step 1: Download Raspberry pi.

<https://www.raspberrypi.org/downloads/raspbian>

### Install Raspberry Pi OS using Raspberry Pi Imager

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi. [Watch our 45-second video](#) to learn how to install an operating system using Raspberry Pi Imager.

Download and install Raspberry Pi Imager to a computer with an SD card reader. Put the SD card you'll use with your Raspberry Pi into the reader and run Raspberry Pi Imager.

[Download for Windows](#)

[Download for macOS](#)

[Download for Ubuntu for x86](#)

To install on **Raspberry Pi OS**, type `sudo apt install rpi-imager` in a Terminal window.



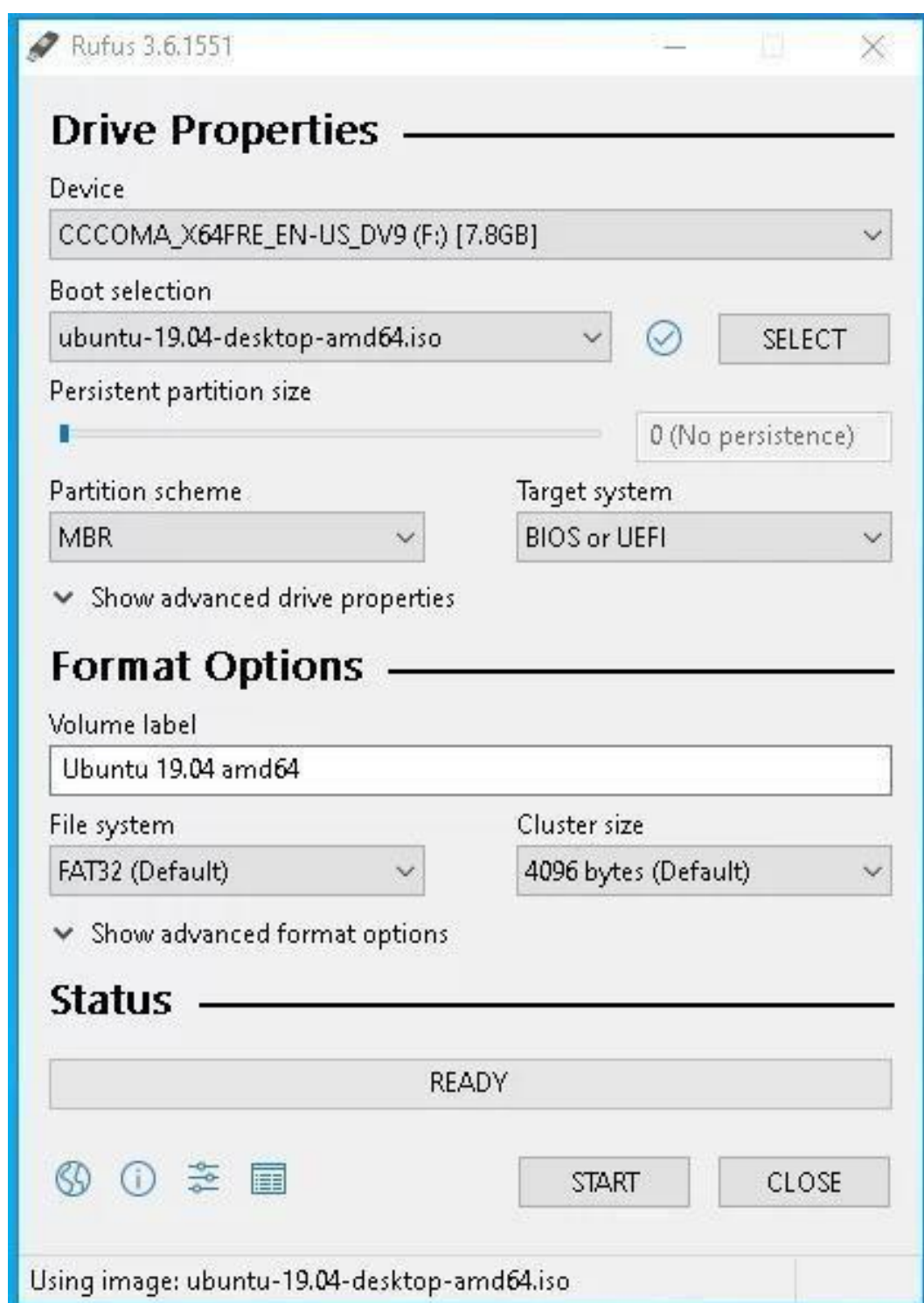
Step 2: Unzip the downloaded file.

Step 3: Start the downloaded file.

Step 4: Download Rufus.

Step 5: By using Rufus Install the Raspbian OS (copy the image file) in the SD Card.

After installing the OS, the name of the SD Card will change to boot drive. It means the OS is successfully installed in the SD Card.



## **PRACTICAL 2: Displaying different LED patterns with Raspberry Pi.**

### **Required Components:**

Raspberry Pi - 1

Power Supply 12 V/2 Amp - 1

HDMI Port - 1

USB Keyboard - 1

USB Mouse - 1

Micro SD Card - 1

LED Module - 1

Jumper (F to F) - 5

### **Connection:**

Connect Pin no.7 (GPIO 4) to LED1 of LED module

Connect Pin no.11 (GPIO 17) to LED2 of LED module

Connect Pin no.13 (GPIO 27) to LED3 of LED module

Connect Pin no.15 (GPIO 22) to LED4 of LED module

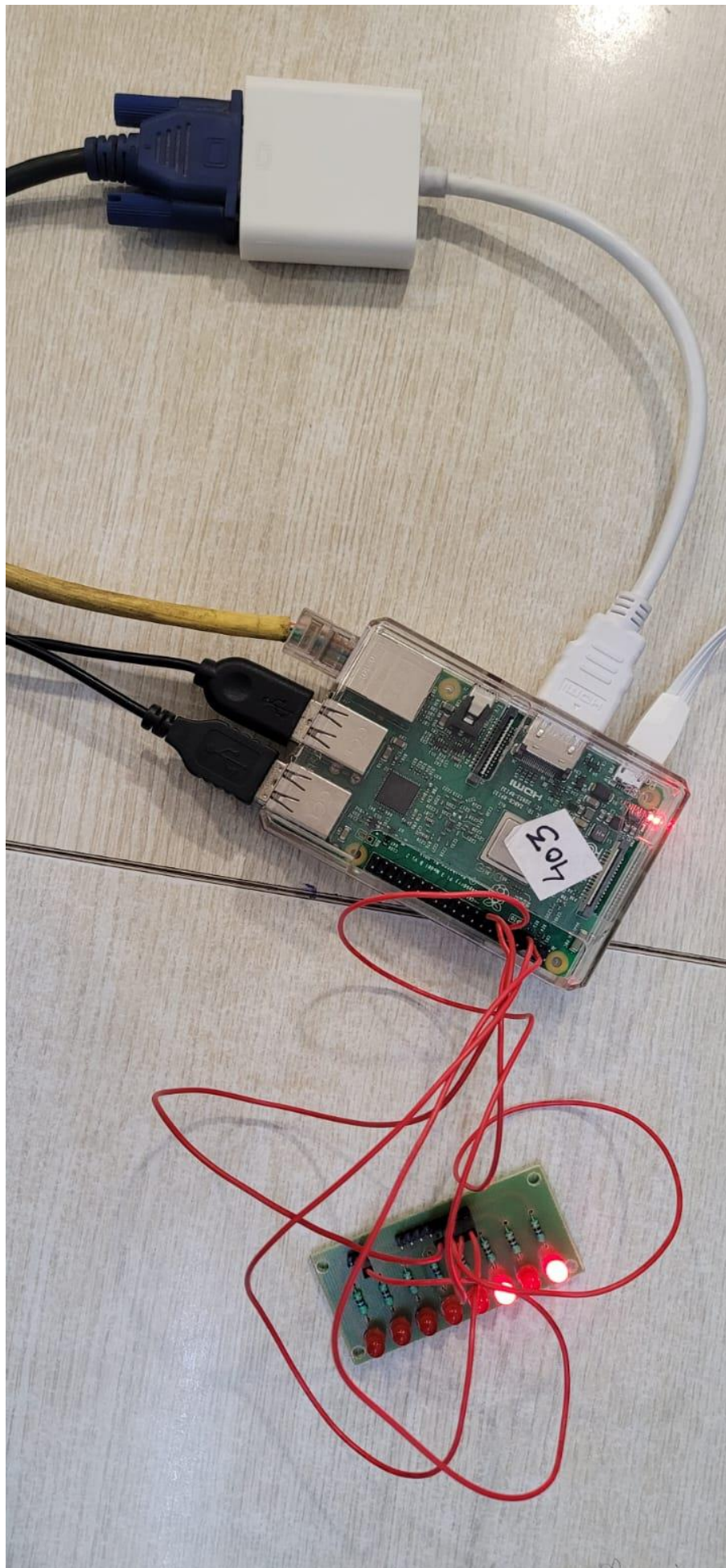
Connect Pin no.6 (GND) to GND of LED module

### **Code:**

```
import RPi.GPIO as GP
import time
GP.setmode(GP.BOARD)
GP.setup(7,GP.OUT)
GP.setup(11,GP.OUT)
GP.setup(13,GP.OUT)
GP.setup(15,GP.OUT)
while (1):
    GP.output(7,GP.HIGH)
    time.sleep(0.2)
    GP.output(11,GP.LOW)
    time.sleep(0.2)
    GP.output(13,GP.HIGH)
    time.sleep(0.2)
    GP.output(15,GP.LOW)
    time.sleep(0.2)
    print (" LED IS ON! ")
    GP.cleanup()
```



Output:



### **PRACTICAL 3: Displaying Time over 4-Digit 7-Segment Display using Raspberry Pi.**

#### **Required Components:**

Raspberry Pi - 1  
Power Supply 12 V/2 Amp - 1  
HDMI Port - 1  
USB Keyboard - 1  
USB Mouse - 1  
Micro SD Card - 1  
4-Digit 7-Segment Module - 1  
Jumper (F to F) - 4

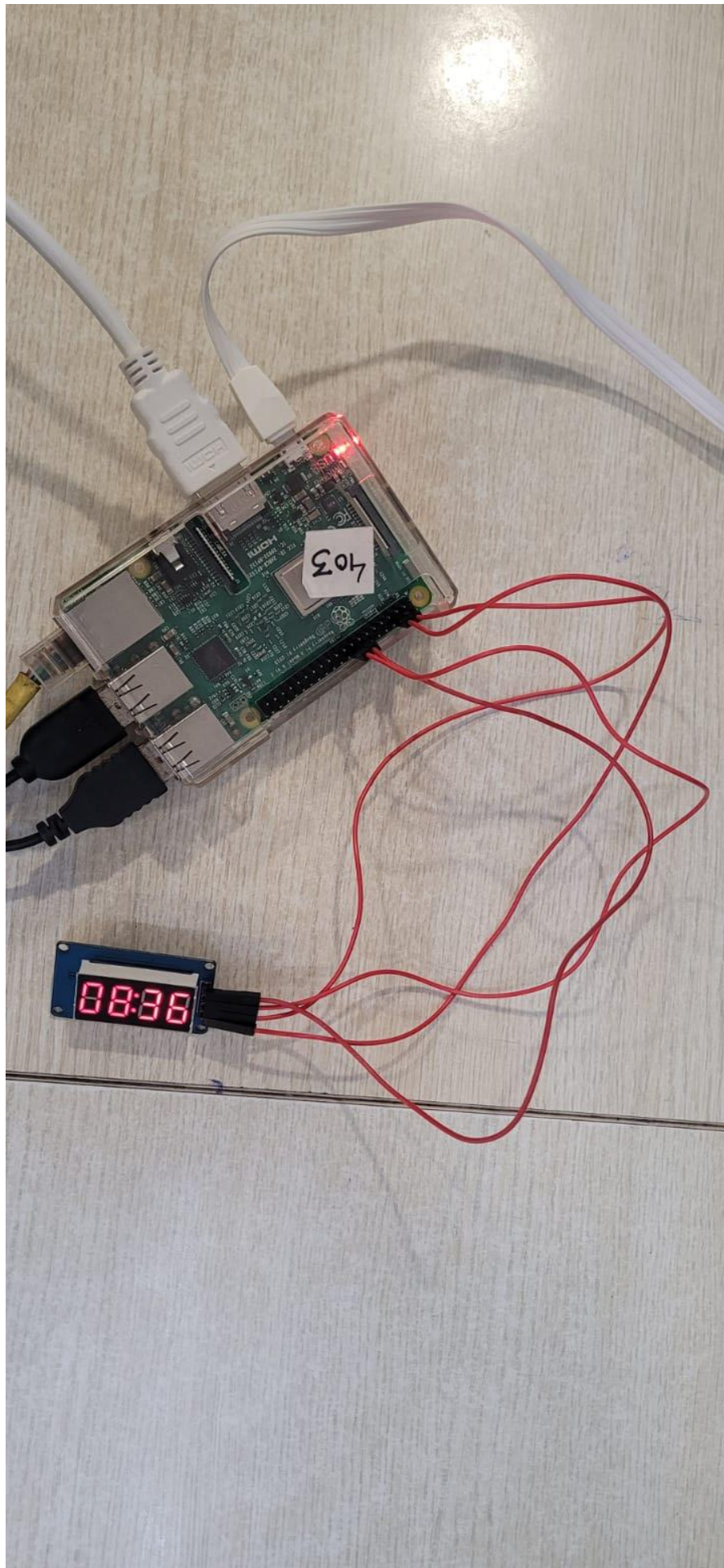
#### **Connection:**

Connect Pin no.16 (GPIO 23) to CLK of Timer  
Connect Pin no.18 (GPIO 24) to DIO of Timer  
Connect Pin no.4 (5V Power) to VCC of Timer  
Connect Pin no.6 (GND) to GND of Timer

#### **Code:**

```
import sys
import time
import datetime
import RPi.GPIO as GP
import tm1637
Display = tm1637.TM1637 (23,24, tm1637. BRIGHT_TYPICAL)
Display.Clear()
Display.SetBrightness (1)
while (True):
    now = datetime.datetime.now()
    hour = now.hour
    minute = now.minute
    second = now.second
    currenttime = [ int (hour / 10), hour % 10, int (minute / 10), minute % 10 ]
    Display.Show(currenttime)
    Display.ShowDoublepoint(second % 2)
    time.sleep(1)
```

Output:



## **PRACTICAL 4: Interfacing Raspberry Pi with RFID.**

### **Required Components:**

Raspberry Pi - 1

Power Supply 12 V/2 Amp - 1

Power Supply with Red & Black extension - 1

HDMI Port - 1

USB Keyboard - 1

USB Mouse - 1

Micro SD Card - 1

Ethernet - 1

RFID Card Reader - 1

USB to TTL Converter - 1

Jumper (F to F) - 4

### **Connection:**

Connect TX of Card Reader to RX of TTL

Connect GND of Card Reader to GND of TTL

Connect Red of Power Supply to VCC of Card Reader

Connect Black of Power Supply to GND of Card Reader

Connect TTL to Raspberry Pi

### **Code:**

```
import RPi.GPIO as GP
import time
import serial
GP.setmode(GP.BOARD)
def read_rfid():
    ser=serial.Serial("/dev/ttyUSB0")
    ser.baudrate=9600
    data=ser.read(12)
    ser.close()
    return data
try:
    while True:
        id=read_rfid()
        print(id)
        if id==b'1D00AF5623C7':
            print ("Access Granted")
            time.sleep(2)
        else:
```

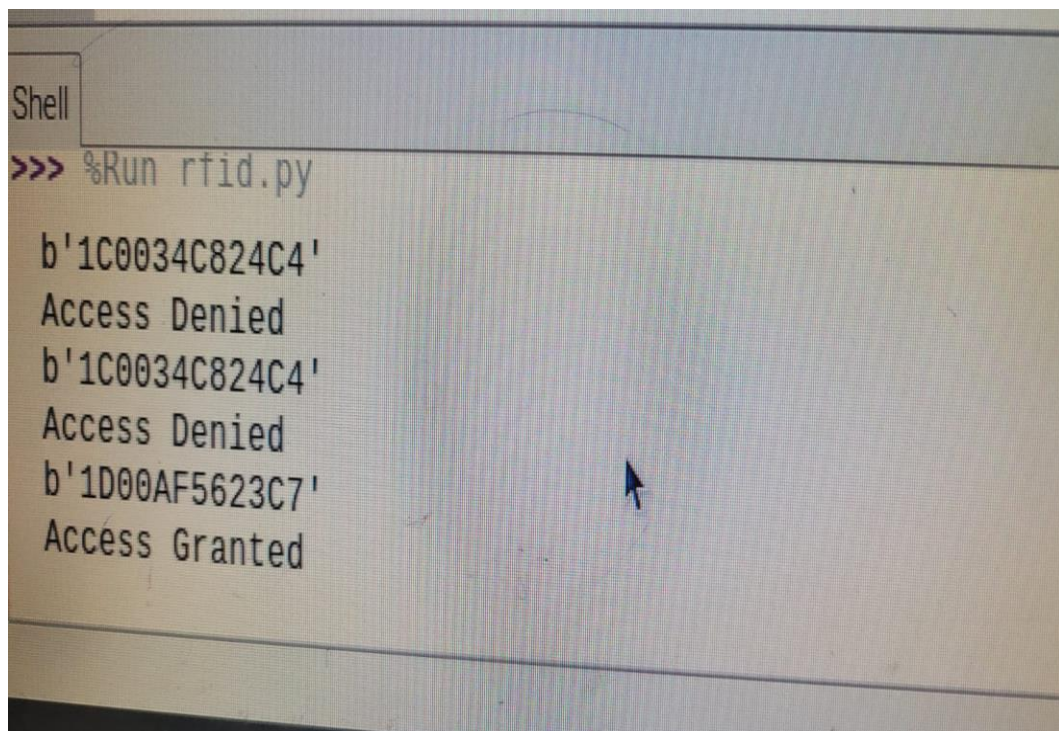


```
print ("Access Denied")
time.sleep(2)
finally:
GP.cleanup()
```

**Note** - If (/dev/ttyUSB0) error occurs run the following commands in the Terminal:

- sudo apt-get update
- sudo apt-get upgrade
- sudo raspi-config (in 'Interface' option 'enable spi')
- sudo reboot (compulsory)
- sudo apt-get install python3-dev python3-pip
- sudo pip3 install spidev

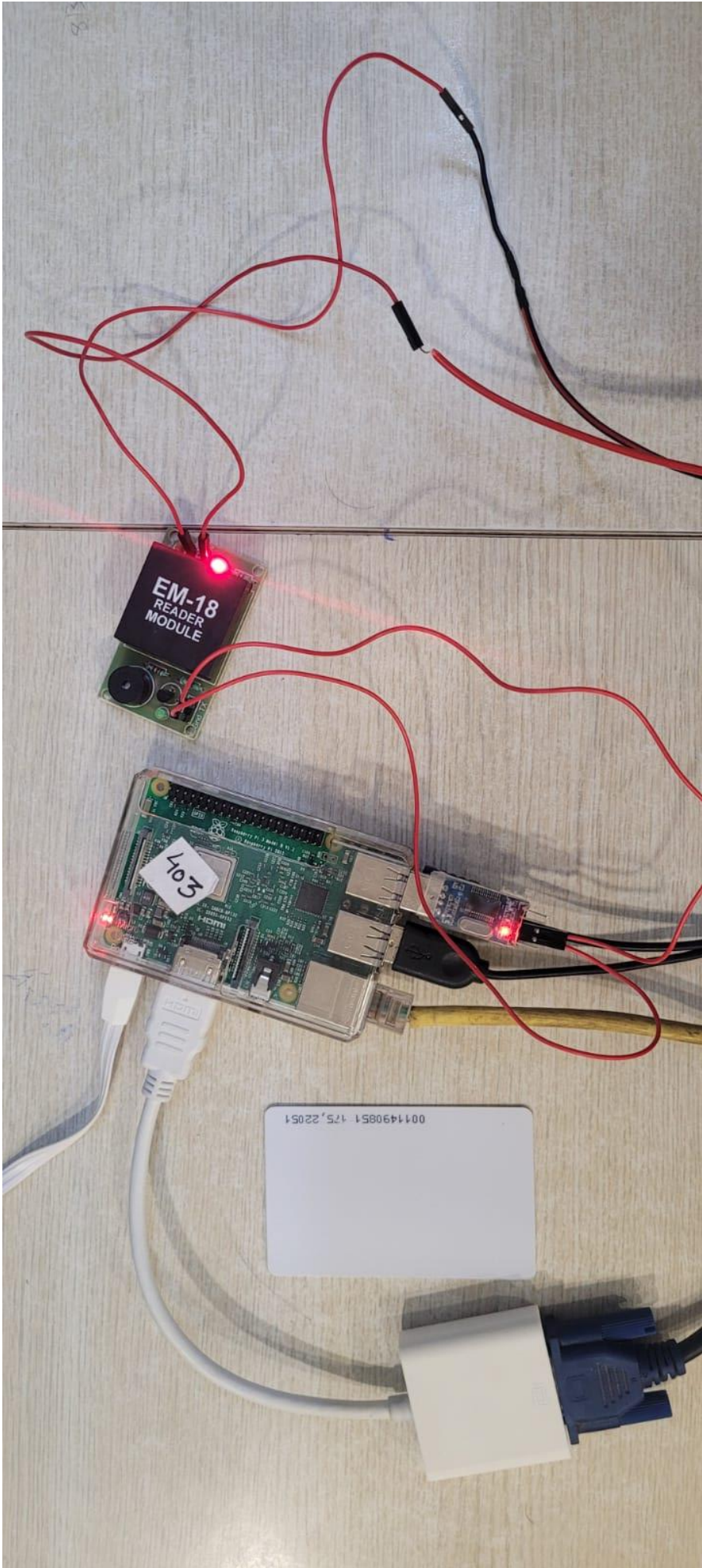
**Output:**



```
Shell
>>> %Run rfid.py

b'1C0034C824C4'
Access Denied
b'1C0034C824C4'
Access Denied
b'1D00AF5623C7'
Access Granted
```





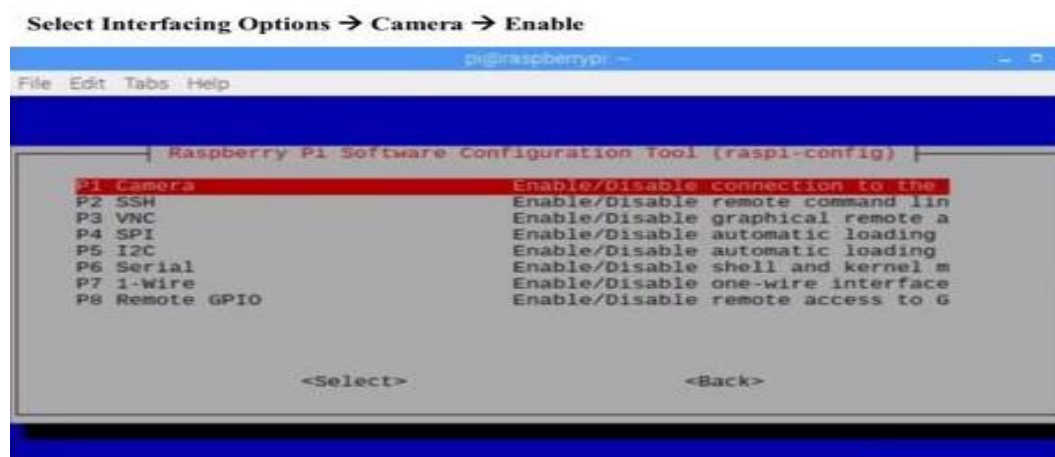
## **PRACTICAL 5: Visitor Monitoring with Raspberry Pi and Pi Camera.**

### **Required Components:**

Raspberry Pi - 1  
Power Supply 12 V/2 Amp - 1  
HDMI Port - 1  
USB Keyboard - 1  
USB Mouse - 1  
Micro SD Card - 1  
Pi Camera Module - 1

### **Connection:**

Connect Pi Camera Module to CSI Camera Port



After this reboot the system, using **sudo reboot** command.

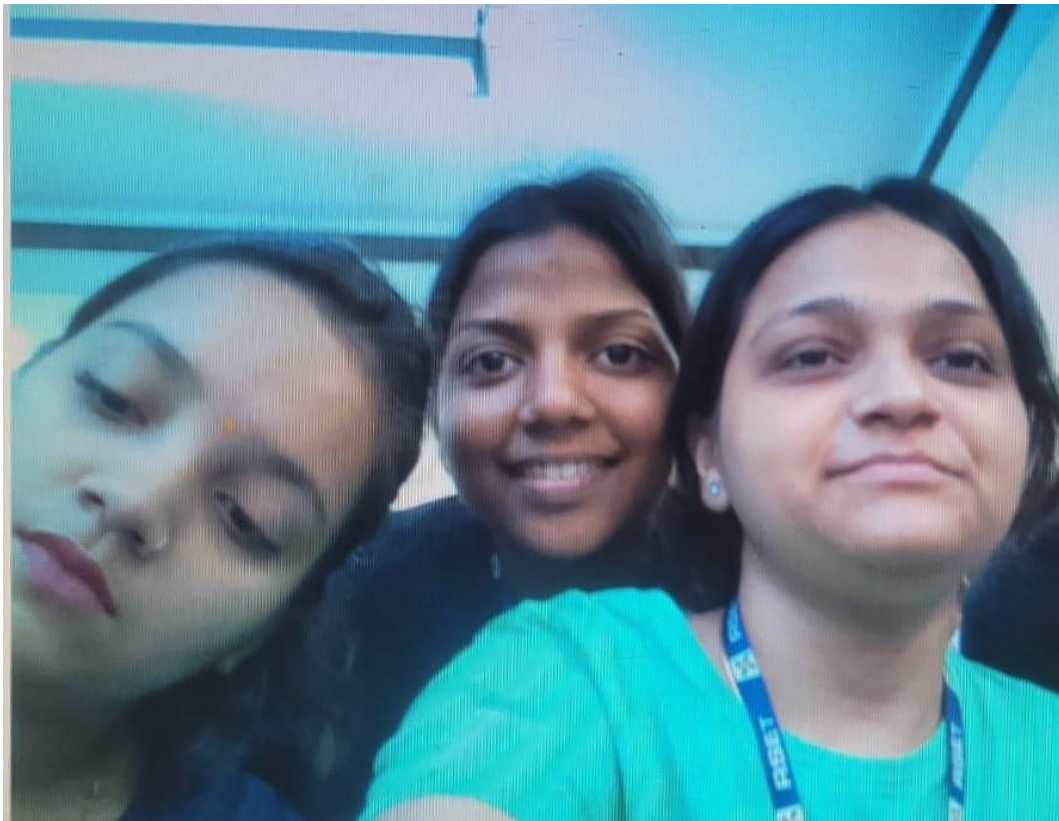
**Commands:** To install Pi-camera packages.

- `sudo apt-get install python-picamera`
- `sudo apt-get install python3-picamera`
- `sudo pip install picamera`

### **Code:**

```
import time
import picamera
camera=picamera.PiCamera()
camera.resolution=(1024,768)
camera.start_preview()
time.sleep(2)
camera.capture('/home/pi/Desktop/visitor/images%s.jpg')
camera.stop_preview
```

**Output:**



**Note** - Type 'exit()' in the shell to esc from the image screen.





## **PRACTICAL 6: IoT based Web Controlled Home Automation using Raspberry Pi.**

### **Required Components:**

Raspberry Pi - 1  
Power Supply 12 V/2 Amp - 1  
HDMI Port - 1  
USB Keyboard - 1  
USB Mouse - 1  
Micro SD Card - 1  
LED Module - 1  
Jumper (F to F) - 2

### **Connection:**

Connect Pin no.3 (GPIO 2) to LED1 of LED module  
Connect Pin no.6 (GND) to GND of LED module

### **Steps:**

- 1) Go to Web browser.
- 2) Type <http://webiopi.trouch.com/DOWNLOADS.html> & download it.
- 3) Extract the downloaded file.

**Commands:** It should be executed in Terminal.

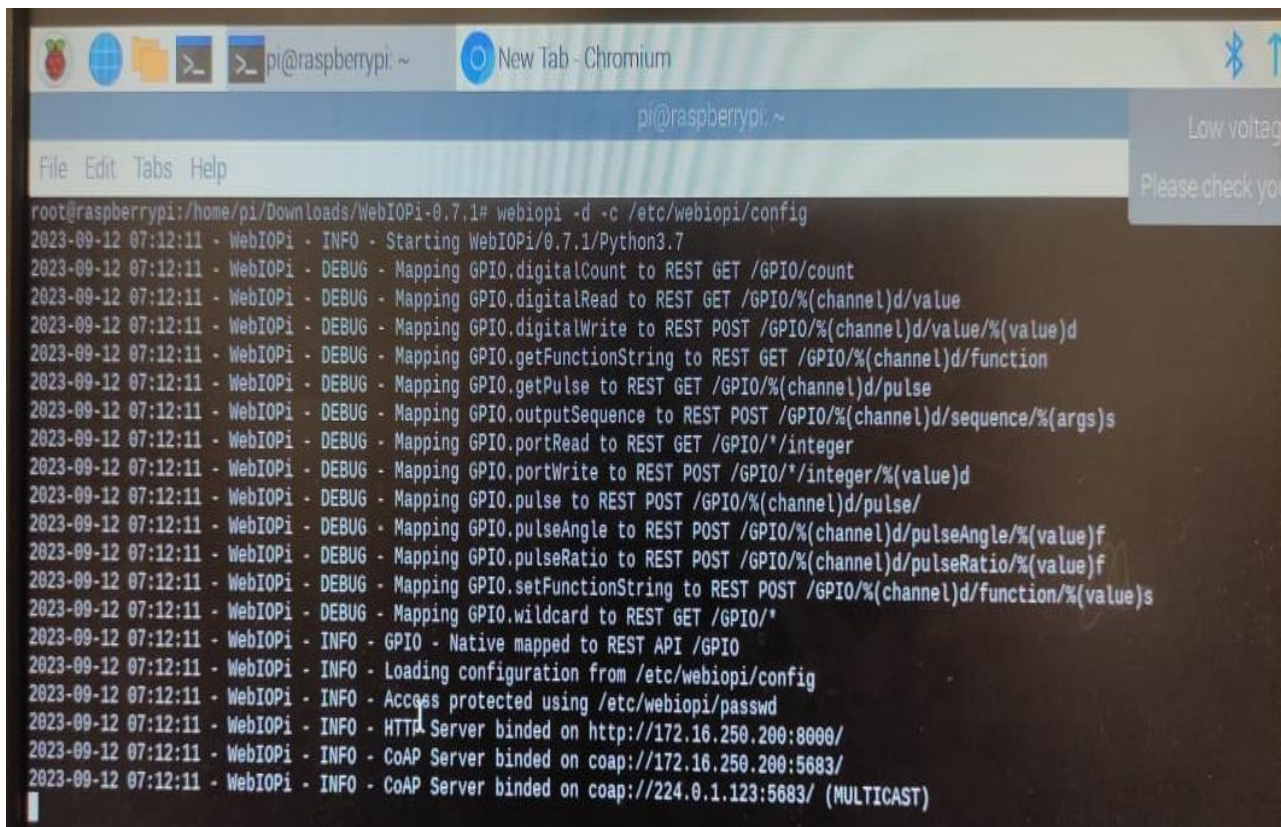
- `sudo apt-get update`
- `sudo apt-get upgrade`
- `cd/home/pi/Downloads` (Move to the downloaded folder)
- `tar xvzf WebIOPi-0.7.1.tar.gz`
- `cd WebIOPi-0.7.1`
- `wget`  
<https://raw.githubusercontent.com/doublebind/raspi/master/webiopi-pi2bplus.patch>
- `patch -p1 -i webiopi-pi2bplus.patch`
- `sudo ./setup.sh`
- `sudo reboot`
- `sudo webiopi -d -c/etc/webiopi/config`

Go to Web browser

Type `http://172.16.250.200:8000/`

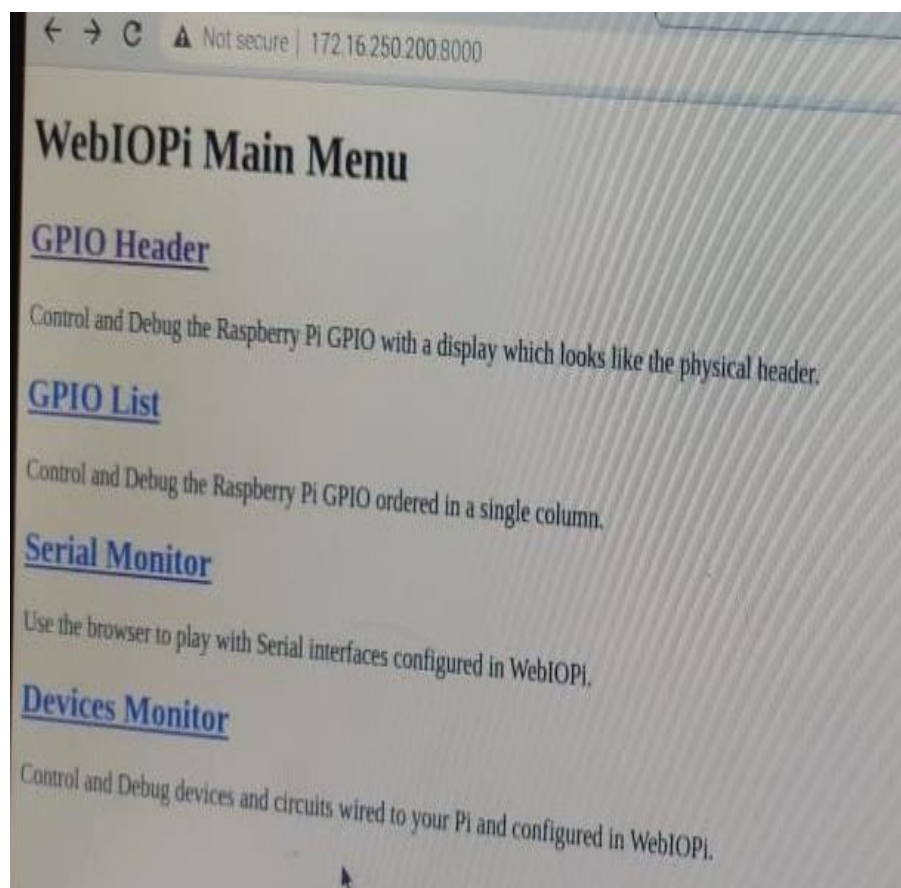


## Output:

A terminal window on a Raspberry Pi showing the output of the 'webiopi -d -c /etc/webiopi/config' command. The logs show the WebIOPi service starting, mapping various GPIO functions to REST API endpoints, and binding the HTTP and CoAP servers. The terminal title is 'pi@raspberrypi: ~' and the window is titled 'New Tab - Chromium'.

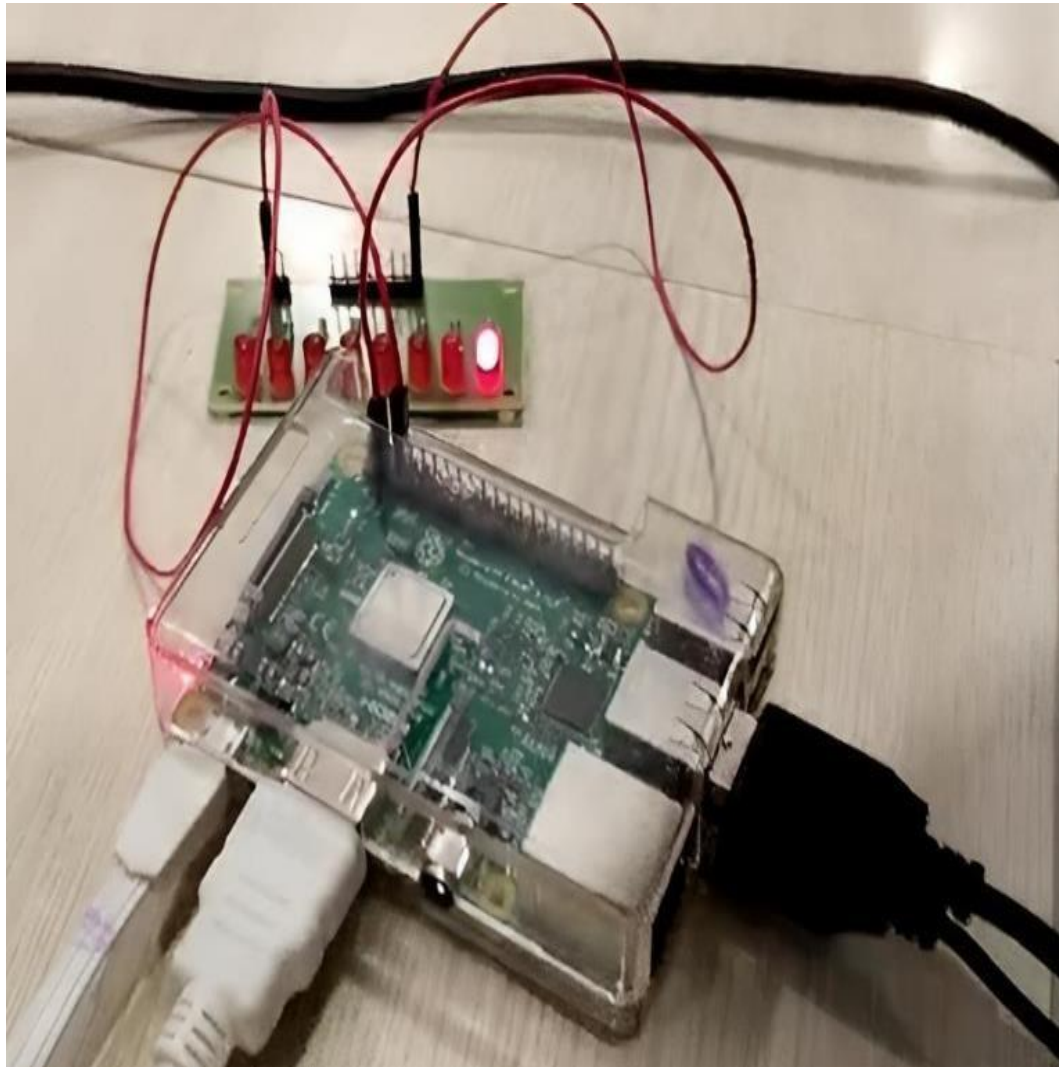
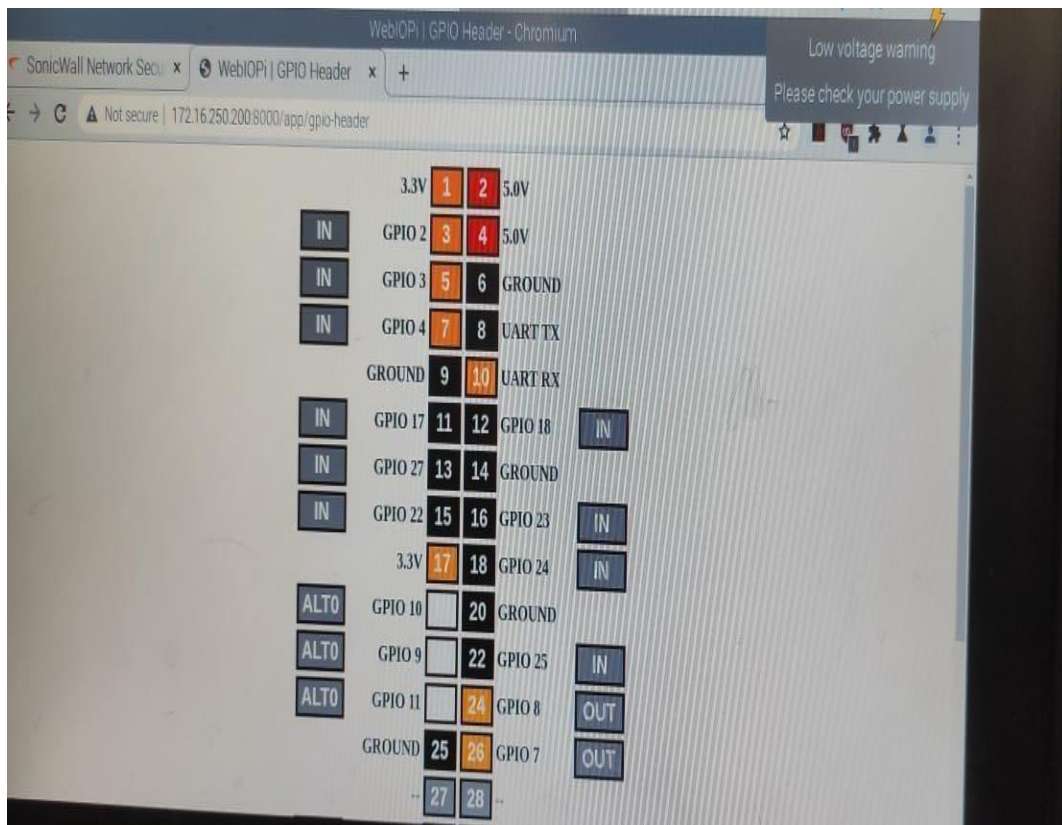
```
root@raspberrypi:/home/pi/Downloads/WebIOPi-0.7.1# webiopi -d -c /etc/webiopi/config
2023-09-12 07:12:11 - WebIOPi - INFO - Starting WebIOPi/0.7.1/Python3.7
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.digitalCount to REST GET /GPIO/count
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.digitalRead to REST GET /GPIO/(channel)d/value
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.digitalWrite to REST POST /GPIO/(channel)d/value/(value)d
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.getFunctionString to REST GET /GPIO/(channel)d/function
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.getPulse to REST GET /GPIO/(channel)d/pulse
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.outputSequence to REST POST /GPIO/(channel)d/sequence/(args)s
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.portRead to REST GET /GPIO/*/integer
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.portWrite to REST POST /GPIO/*/integer/(value)d
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.pulse to REST POST /GPIO/(channel)d/pulse/
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.pulseAngle to REST POST /GPIO/(channel)d/pulseAngle/(value)f
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.pulseRatio to REST POST /GPIO/(channel)d/pulseRatio/(value)f
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.setFunctionString to REST POST /GPIO/(channel)d/function/(value)s
2023-09-12 07:12:11 - WebIOPi - DEBUG - Mapping GPIO.wildcard to REST GET /GPIO/*
2023-09-12 07:12:11 - WebIOPi - INFO - GPIO - Native mapped to REST API /GPIO
2023-09-12 07:12:11 - WebIOPi - INFO - Loading configuration from /etc/webiopi/config
2023-09-12 07:12:11 - WebIOPi - INFO - Access protected using /etc/webiopi/passwd
2023-09-12 07:12:11 - WebIOPi - INFO - HTTP Server binded on http://172.16.250.200:8000/
2023-09-12 07:12:11 - WebIOPi - INFO - CoAP Server binded on coap://172.16.250.200:5683/
2023-09-12 07:12:11 - WebIOPi - INFO - CoAP Server binded on coap://224.0.1.123:5683/ (MULTICAST)
```

Click on the GPIO Header.



First click on 'IN' next to GPIO 2(Pin no.3) & turn it into 'OUT'.

Now click on Pin no.3 to turn LED “ On & Off ” and get the output.





## **PRACTICAL 7: Controlling Raspberry Pi with Telegram.**

### **Required Components:**

Raspberry Pi - 1  
Power Supply 12 V/2 Amp - 1  
HDMI Port - 1  
USB Keyboard - 1  
USB Mouse - 1  
Micro SD Card - 1  
LED Module - 1  
Jumper (F to F) - 2

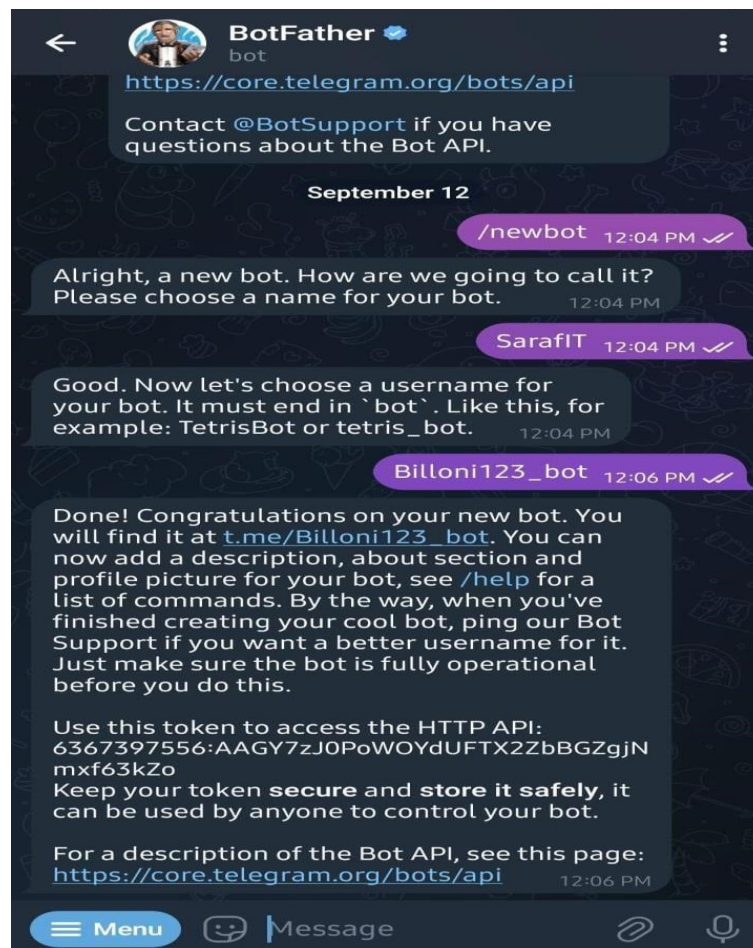
### **Connection:**

Connect Pin no.3 (GPIO 2) to LED1 of LED module  
Connect Pin no.6 (GND) to GND of LED module

Step 1: Install Telegram Bot on Raspberry Pi.

- `sudo apt-get update`
- `sudo apt-get upgrade`
- `sudo apt-get install python-pip`
- `sudo pip install telepot`

Step 2: Create a Bot in your Telegram App.



**Code:**

```
import time,datetime
import RPi.GPIO as GP
import telepot
import sys
def on(pin):
    GP.output(pin, GP.HIGH)
def off(pin):
    GP.output(pin, GP.LOW)
return
GP.setmode(GP.BOARD)
GP.setwarnings(False)
GP.setup(3, GP.OUT)
def handle(msg):
    chat_id = msg['chat']['id']
    command = msg['text']
    print ('Got command: %s' % command)
    if command == '/on':
        bot.sendMessage(chat_id, on(3))
    elif command == '/off':
        bot.sendMessage(chat_id, off(3))
    bot = telepot.Bot('Enter_your_bot_token_here')
    bot.message_loop(handle)
    print ("I am Listening...")
    while 1:
        try:
            time.sleep(10)
        except KeyboardInterrupt:
            print ('Program interrupted')
            GP.cleanup()
            exit()
        except:
            print ('Other error')
            GP.cleanup()
```



Output:

