IoT Security and Privacy

Assignment 2 – Raspberry Pi

10 points

Instructions:

1. Note: Blue text points to a web link. Ctrl + Click to follow link.
2. This is a team assignment. However, each member of the team has to submit the finished assignment. Those who do not submit will get zero for this assignment.
3. Answers to all questions must be put into **ONE** document. That is, every time, each student can only submit one report document, answering all questions of this assignment.
4. Students must put answers following each question in this assignment. The instructor will not grade a report with only answers in it and the student gets zero for such an assignment. An assignment report must include original questions.
5. Students MUST submit the finished assignment in either Microsoft Word or pdf format to Blackboard. The doc must be submitted as ONE standalone file and cannot be tarred or zipped into a container.
6. Refer to Print screen on how to take a screenshot. Pressing the Alt key in combination with PrtSc will capture the currently selected window.

Questions:

Question 1 (8 points)

In this assignment, students are required to use sensors on raspberry pi. If a group has n members, the group needs to try n different sensors. Ideally, each member works on one sensor and documents according to the requirements below.

Please refer to [1] for use of raspberry pi and [2] for various sensor manuals. Search “manual” in the page for manuals of the sensors.

Requirements:
For each sensor,

1. Introduce what the sensor does.
2. Include a photo of raspberry pi with the connected sensor. Explain how the sensor is connected to GPIO pins or the board of raspberry pi. Any accessories such as cameras are considered as sensors.
3. Include the results of your experiments controlling or communicating the sensor.
4. Include the code for controlling or communicating the sensor.

Question 2 (2 points)

1. Students will work in a team for the term project. Each team can have at most 4 students. Please identify all team members below.

   Team Members:

   1. Kiran Shettar
   2. Omkar Salunke
   3. Rohan Girase
   4. Tarun Jaykumar Moorjani

2. Please introduce the term project in mind right now.

   Project Idea:

   Title : Intelligent Door

   The idea of the project is to detect the position of the door i.e whether the door is closed or open. This can be used for varied purpose such as to maintain the temperature of the room, for security purposes etc. The main goal of the project is to use machine learning algorithm with a webcam integrated to raspberry pi to do rapid integration prototyping of IoT. We are going to train our model with a series of pictures(video) so that a single image is enough to predict the object present in the image. We are going to take a picture every 5 sec to detect the state of door whether it is open or closed. This picture will be processed using artificial neural network algorithm to detect the object present in the image. This idea can also be implemented in water sprinkler system in which the camera of the raspberry pi detects the intensity of the pixels around it to identify whether it’s day or night and then sprinkle water accordingly.

References

[1] Chao Gao, Use of Raspberry Pi, 2016

[2] SainSmart 37 in 1 Sensor Module Kit for Arduino UNO R3 Mega2560 Mega328 Nano Raspberry Pi, 2016.

Sensors Used:

1. Camera
2. PIR
3. Humidity
4. LED

1. LED Sensor:

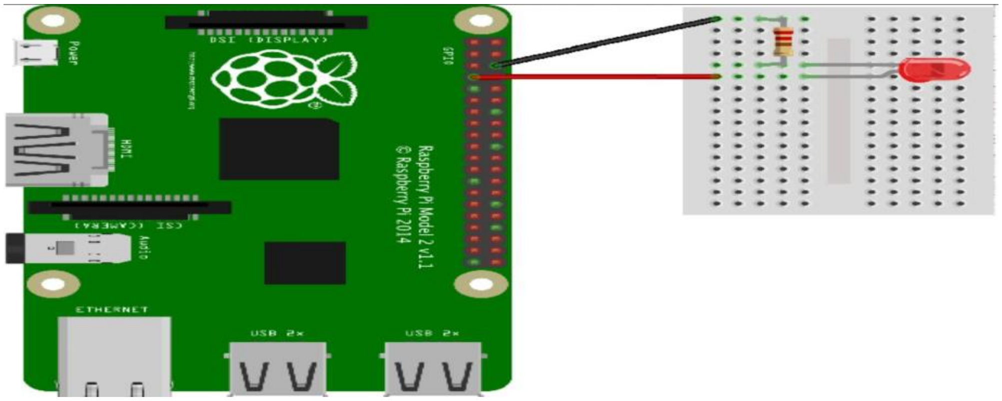
Team Member worked: Kiran Shettar

- **Introduce what the sensor does.**

  The LED(Light Emitting Diode) sensor converts electrical input signal into light, a closed optical channel (also called dielectrical channel), and a photosensor, which detects incoming light and either generates electric energy directly, or modulates electric current flowing from an external power supply.

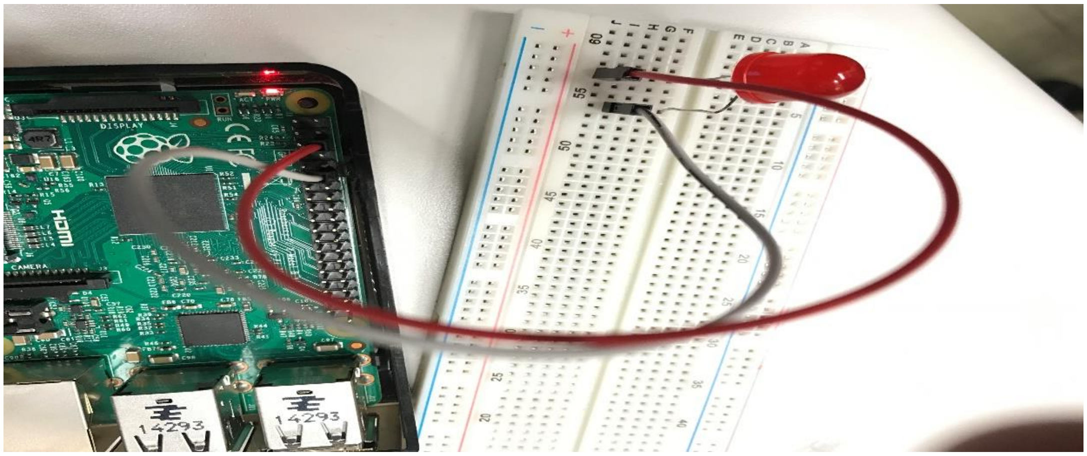
- **Include a photo of raspberry pi with the connected sensor. Explain how the sensor is connected to GPIO pins or the board of raspberry pi. Any accessories such as cameras are considered as sensors.**

  Below is the circuit followed to connect the LED sensor:



  Process:
  >> GPIO 18 is used as a output pin.
  >> LED CONNECTION: The longer leg (known as the ‘anode’), is always connected to the positive supply of the circuit. The shorter leg (known as the ‘cathode’) is connected to the negative side of the power supply, known as ‘ground’.
  >> Connections were done by connecting to a resistor and also without connecting to a resistor.

  CONNECTION BETWEEN IoT device and the LED sensor:

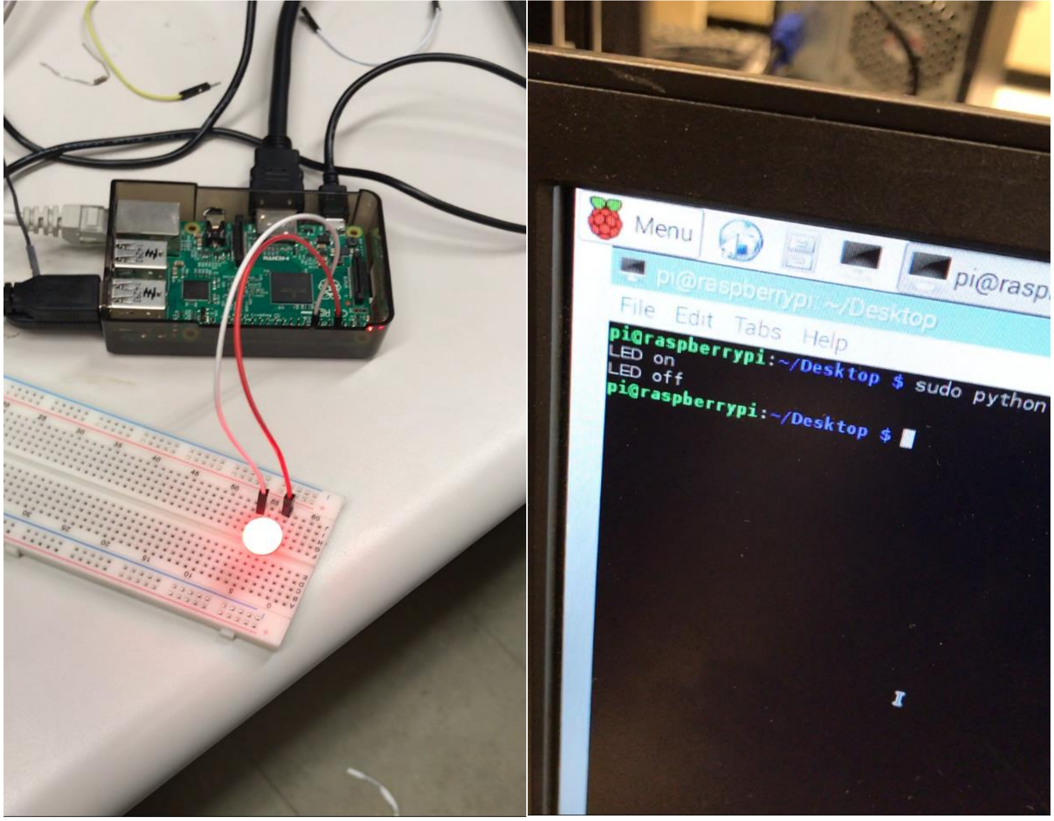


- **Include the results of your experiments controlling or communicating the sensor.**

  Results of the experiment:

  The sleep time is 5 seconds in my code which can be modified anytime. Once we run the python code the LED will turn on i.e it’ll be HIGH. After 5 seconds the LED will turn off i.e. it’ll be LOW.

If you terminate the code in between then LED will be on.

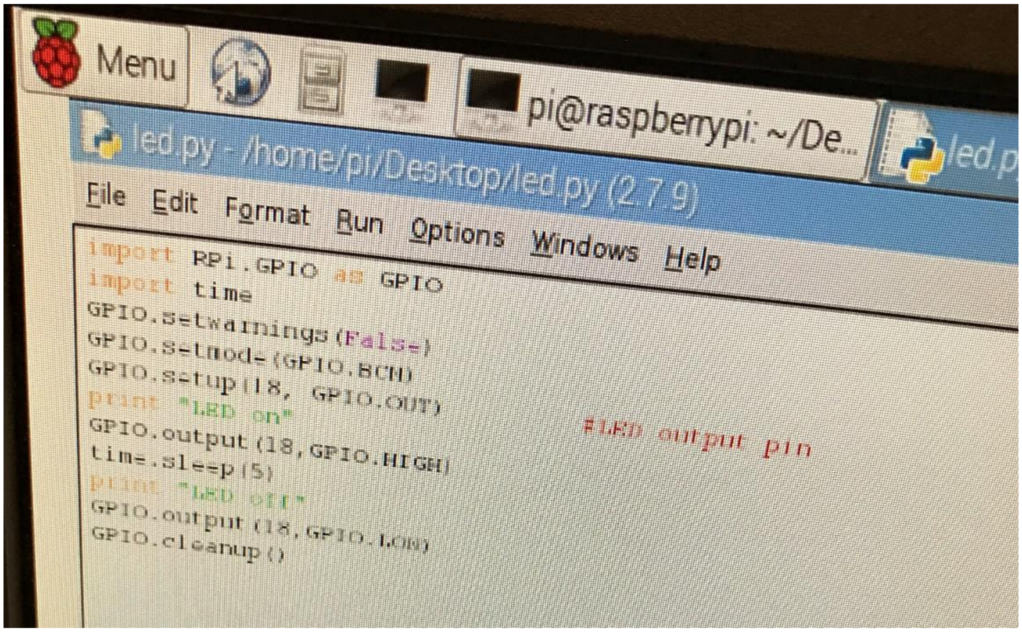


```
pi@raspberrypi:~/Desktop $ sudo python
LED on
LED off
pi@raspberrypi:~/Desktop $
```

Raspberry Pi connected with the LED sensor and successfully communicating with it.

- **Include the code for controlling or communicating the sensor.**

```
//CODE to communicate with LED sensor
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
print “LED on”
GPIO.output(18, GPIO.HIGH)
time.sleep(5)
print “LED off”
GPIO.output(18,GPIO.LOW)
GPIO.cleanup()
```


Menu
pi@raspberrypi: ~/De...
led.py - /home/pi/Desktop/led.py (2.7.9)
led.p
File Edit Format Run Options Windows Help

```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
print "LED on"
GPIO.output(18,GPIO.HIGH)                    #LED output pin
time.sleep(5)
print "LED off"
GPIO.output(18,GPIO.LOW)
GPIO.cleanup()
```

2. Humidity Sensor:

Team Member worked: Omkar Salunke

- Introduction:

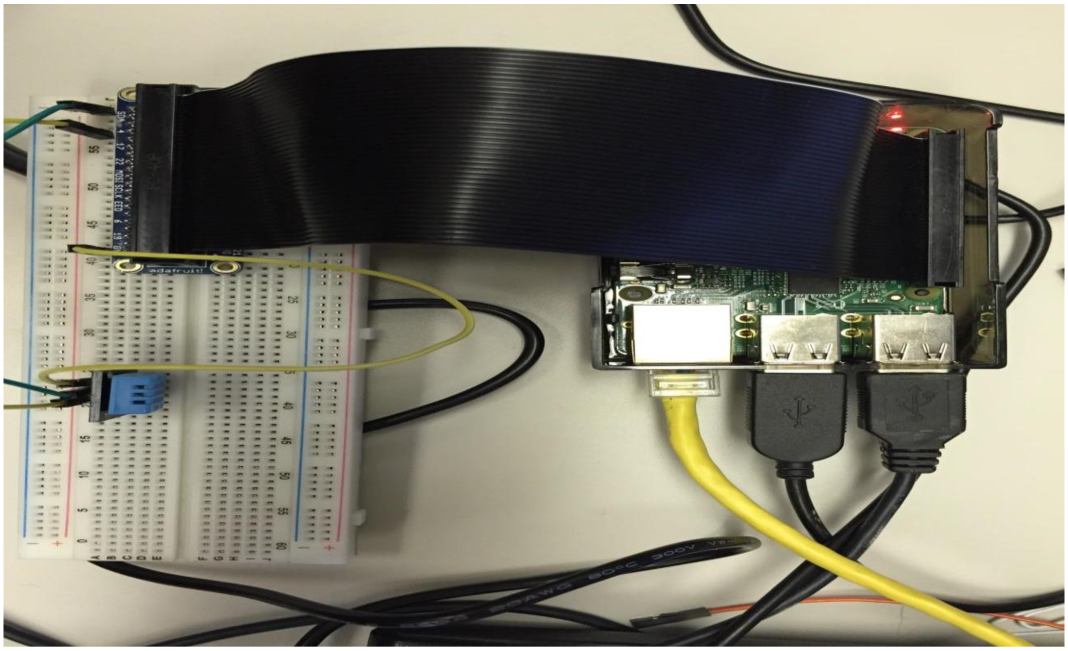
The Digital Temperature and Humidity sensor (DHT) senses the temperature and humidity around the raspberry pi and is capable of measuring the Temperature precise to 0.1 F, and relative Humidity upto 0.1 %.

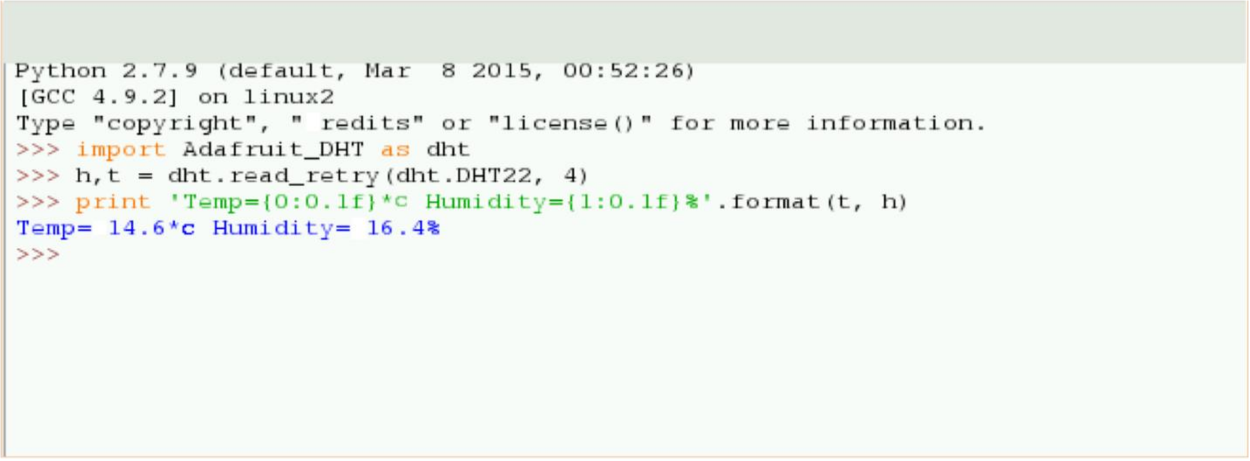
Connecting Sensor:

The sensor is installed in a breadboard. It has 3 connections, named V (high Voltage), G (Ground), and D (Data). The Data input is connected to GPIO-4.

Enabling the Sensor:

For this sensor the Python Library Adafruit was used. Downloaded from github, the Adafruit_python_DHT library allowed the sensor to pass its readings to the terminal.
Code for sensor:
“ h,t = dht.read.retry(dht.DHT22, 4) “
Images:





```
Python 2.7.9 (default, Mar  8 2015, 00:52:26)
[GCC 4.9.2] on linux2
Type "copyright", " redits" or "license()" for more information.
>>> import Adafruit_DHT as dht
>>> h,t = dht.read_retry(dht.DHT22, 4)
>>> print 'Temp={0:0.1f}*c Humidity={1:0.1f}%'.format(t, h)
Temp= 14.6*c Humidity= 16.4%
>>>
```

3. PIR sensor (Passive Infrared Motion Sensor):

Team Member Worked: Rohan Girase

Introduction:

The sensor adjusts itself to the infrared signature of the room it's in and then watches for any changes is regarded as passive because it doesn't send out any signal in order to detect movement. The output is digital. A change is caused by any object moving through the room which disturbs the infrared signature, and the change is noticed by the PIR module. The language used is python3.

Connection:

The sensor is connected to the raspberry pi through the breadboard using an interface.

The three pins of the sensor are connected to their respective ports in the breadboard: Top port (red wire) is connected to the 5v power supply, Middle port (yellow wire) to pin4 of the board and bottom port (black wire) to the ground connection of the board.

Working:

We first create a blank python file with the command: “*nano pir.py*”.

In the file we write the code:

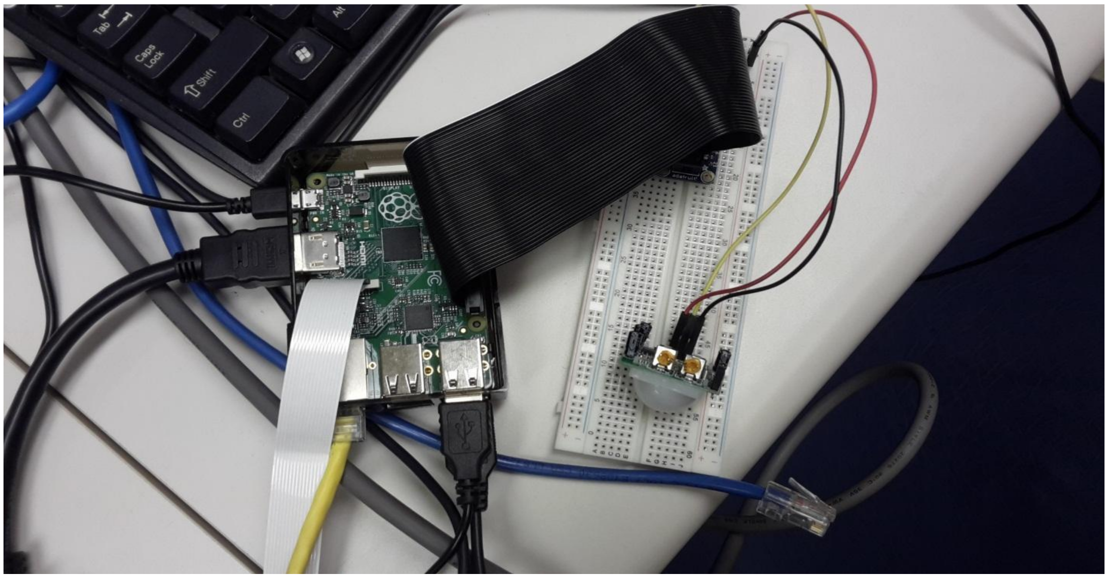
```
import RPi.GPIO as GPIO
import time
sensor = 4
GPIO.setmode(GPIO.BCM)
GPIO.setup(sensor, GPIO.IN, GPIO.PUD_DOWN)
previous_state = False
current_state = False
while True:
time.sleep(0.1)
previous_state = current_state
current_state = GPIO.input(sensor)
if current_state != previous_state:
new_state = "HIGH" if current_state else "LOW"
print("GPIO pin %s is %s" % (sensor, new_state))
```

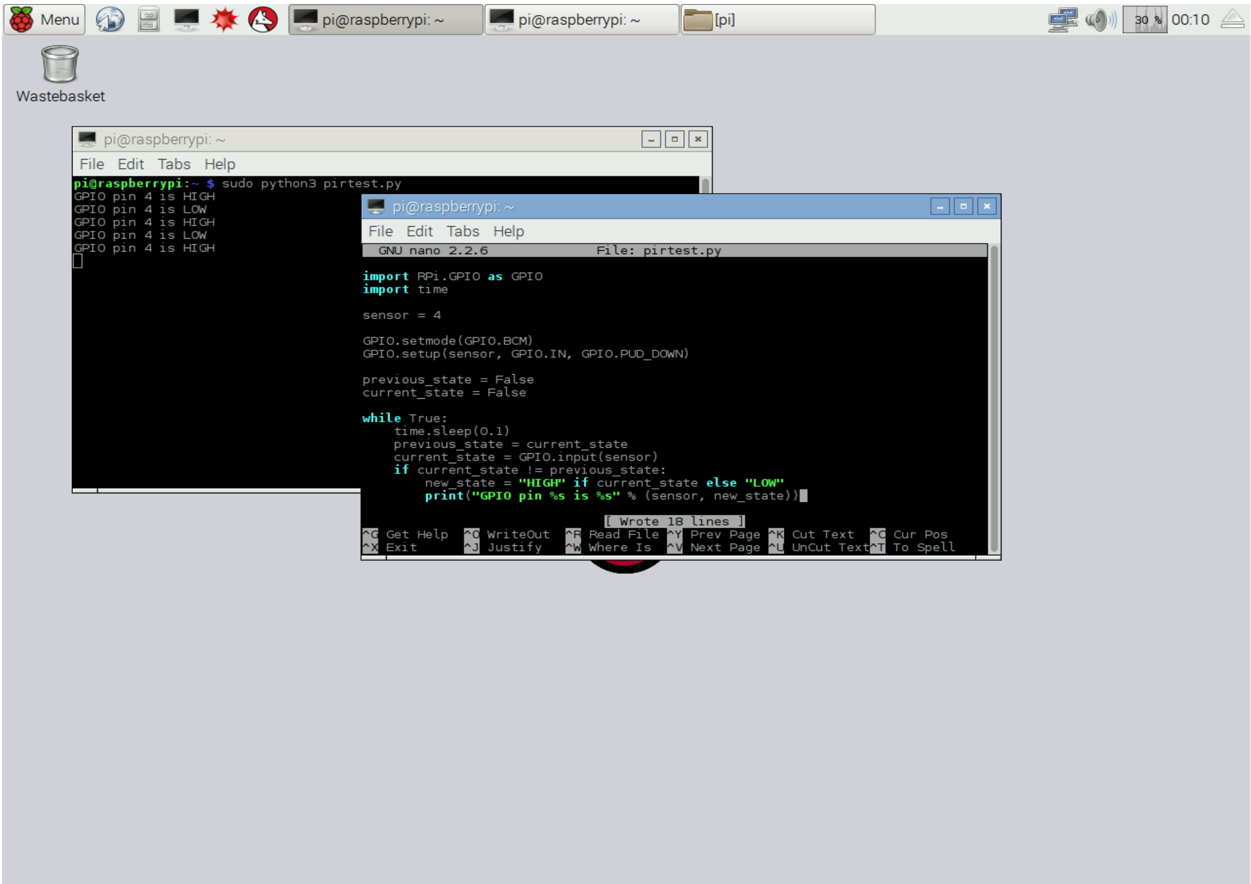
Save the code in the file by clicking “*ctrl+o*” and exit to terminal by typing “*ctrl+x*”.

Now type the following command to run the sensor: “*Sudo python3 pir.py*”

The output is high whenever motion is detected.

Images:





```
pi@raspberrypi:~ $ sudo python3 pirtest.py
GPIO pin 4 is HIGH
GPIO pin 4 is LOW
GPIO pin 4 is HIGH
GPIO pin 4 is LOW
GPIO pin 4 is HIGH
```

```
  GNU nano 2.2.6              File: pirtest.py

import RPi.GPIO as GPIO
import time

sensor = 4

GPIO.setmode(GPIO.BCM)
GPIO.setup(sensor, GPIO.IN, GPIO.PUD_DOWN)

previous_state = False
current_state = False

while True:
    time.sleep(0.1)
    previous_state = current_state
    current_state = GPIO.input(sensor)
    if current_state != previous_state:
        new_state = "HIGH" if current_state else "LOW"
        print("GPIO pin %s is %s" % (sensor, new_state))

                              [ Wrote 18 lines ]
^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text^T To Spell
```

4. Camera Sensor:

Team Member worked: Tarun Jaykumar Moorjani

Introduction:

The camera connected to Raspberry Pi is capable of taking 1080p photos and videos and controlled through programs. The language used for coding camera is python3.

Connecting camera:

With the silver connectors facing the HDMI port, the flex cable is inserted into the connector slot situated in between the Ethernet and HDMI ports. Insert the flex cable into the slot and push the tab down to connect the camera. The flex cable connector should be opened by pulling the tabs on the top of the connector upwards then towards the Ethernet port.

Enabling the camera:

Open terminal and type "*raspi-config*" and hit enter after physical connection. Select enable camera option and select the finish option. When prompted reboot the system.

Code for camera:

"*Raspivid –t 0*"

Images: