IoT LAB FILE

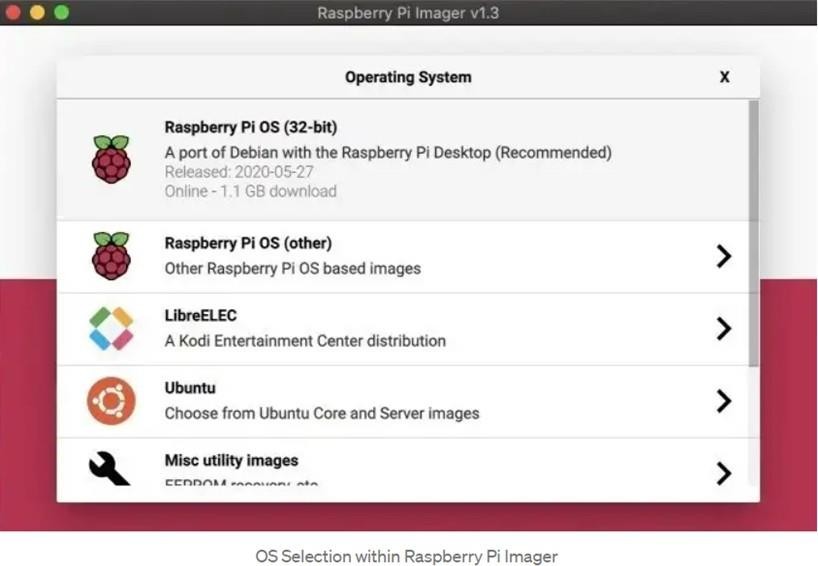
# Configure Raspberry Pi on the system and explore its GUI

Step 1: Download and Install Raspberry Pi Imager

Download the Raspberry Pi Imager for your operating system and follow the installation instructions. Launch Raspberry Pi Imager.



Step 2: Choose OS



Step 3: Choose SD Card



Step 4: Write to SD Card



Step 5: Booting Your Raspberry Pi

Insert your microSDHC card into your Raspberry Pi. Then, hook up your Raspberry Pi to power, keyboard, mouse, and monitor.



# Get Values for IR, LDR, PIR, Ultrasonic, Rain, and Sound sensors using Raspberry

## LDR Sensor

Code:

import RPI.GPIO as GPIO import time GPIO.setmode(GPIO.BOARD)

pin\_to\_circuit=22

Def rc\_time (Pin\_to\_circuit): count=0

GPIO.setup(pin\_to\_circuit, GPIO.OUT) GPIO.setup(pin\_to\_circuit, GPIO.LOW) time.sleep(0.1) GPIO.setup(pin\_to\_circuit, GPIO.IN)

While (GPIO.input(pin\_to\_circuit) ==GPIO.LOW): count +=1

Return count

try

While True: print(rc\_time(pin\_to\_circuit

)) except KeyboardInterrupt: pass

ﬁnally :

GPIO.cleanup()

Output:

Valuesof LDRdepend upon the light exposed on it. Thesevalues are displayed on the serial monitor.

## Ultrasonic Sound Sensor

Code:

import RPi.GPIO as GPIO

import time

#GPIO Mode (BOARD / BCM)

GPIO.setmode(GPIO.BCM)

#set GPIO Pins GPIO\_TRIGGER =18

GPIO\_ECHO =24

#set GPIO direction (IN / OUT) GPIO.setup(GPIO\_TRIGGER, GPIO.OUT) GPIO.setup(GPIO\_ECHO, GPIO.IN)

def distance():

# set Trigger to HIGH GPIO.output(GPIO\_TRIGGER, True) # set Trigger after 0.01ms to LOW time.sleep(0.00001) GPIO.output(GPIO\_TRIGGER, False)

StartTime = time.time() StopTime

=time.time()

# saveStartTime

while GPIO.input(GPIO\_ECHO) == 0: StartTime =time.time()

# save time of arrival

while GPIO.input(GPIO\_ECHO) == 1: StopTime =time.time()

# time diﬀerence betweenstartandarrival TimeElapsed = StopTime - StartTime

# multiply with the sonicspeed(34300 cm/s) # and divide by 2, because there and back distance =(TimeElapsed \* 34300) /2

return distance

if name == ' main ': try:

while True:

dist =distance()

print("Measured Distance =%.1fcm"% dist) time.sleep(1)

# Reset bypressing CTRL+ C except KeyboardInterrupt: print("Measurement stopped by User") GPIO.cleanup()

Output:

The distance is measured and displayed in the monitor.

## IR Sensor

Code :

import RPi.GPIO as GPIO import time

sensor=1 6

buzzer=1 8

GPIO.setmode(GPIO.BOARD)

GPIO.setup(sensor,GPIO.IN) GPIO.setup(buzzer,GPIO.OUT) GPIO.output(buzzer,False) Print“IRSensor Ready………”

Print ““

try:

While True:

if GPIO.input(sensor):

GPIO.output(buzzer,True) print “object detected” While GPIO.input(sensor):

time.sleep(0.2)

Else:

GPIO.output(buzzer,fals

e) except KeyboardInterrupt:

GPIO.cleanup()

Output:

“Object Detected” message is displayed when an IR sensor detects an object.

## Rain Sensor for Rain Detection

Code:

from time import sleep

from gpiozero import Buzzer, InputDevice

buzz=Buzzer(13) no\_rain=InputDevice(18)

defbuzz\_now(iterations): for x in

range(iterations)

: buzz.on() sleep(0.1) buzz.oﬀ() sleep(0.1)

While True:

Output:

if notno\_rain.is\_active:

print(“its raining - get the washing in!”)

buzz\_now(5)

sleep(1)

When water is sensed the message “Its raining - get the washing in!” is displayed.

## PIR Sensor

Code:

from gpiozero import MotionSensor pir = MotionSensor(4)

while True:

pir.wait\_for\_motion() print(“you moved”) pir.wait\_for\_no\_motion()

Output:

“You Moved” message is displayed when ever a movement is detected.

# Uploading values of IoT sensors in the cloud

Enable SSH and I2C

* 1. Connect Pito the monitor, keyboard, and mouse.
  2. Start Pi and then signinto Raspberry Pi OS by using pias the user name and raspberry as the password.
  3. Click the Raspberry icon> Preferences> Raspberry Pi Conﬁguration.
  4. On the Interfaces tab, set SSH and I2C to Enable, and then click OK.
  5. Connect the raspberry pitoasensor.
  6. Connect Piwith network .

# Implement IoT Core service to deploy sensor values on AWS Cloud

* 1. Browse AWS IoT Core 2.Register the Device in AWS
     1. Write Python Program for AWS IoT

import time

import paho.mqtt.client as mqtt import ssl

import json import thread

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM) GPIO.setup(21, GPIO.OUT)

def on\_connect(client, userdata, flags, rc): print("Connected with result code "+str(rc))

client = mqtt.Client() client.on\_connect = on\_connect

client.tls\_set(ca\_certs='./rootCA.pem', certfile='./123789c-certificate.pem.crt', keyfile='./123789c-private.pem.key', tls\_version=ssl.PROTOCOL\_SSLv23)

client.tls\_insecure\_set(True)

client.connect("p1345abcde-ats.iot.uk-west-1.amazonaws.com", 8883, 60) #Taken from REST API endpoint - Use your own.

def intrusionDetector(Dummy): while (1):

x=GPIO.input(21) if (x==0):

print "Just Awesome"

client.publish("device/data", payload="Hello from BinaryUpdates!!" , qos=0, retain=False)

time.sleep(5)

thread.start\_new\_thread(intrusionDetector,("Create intrusion Thread",)) client.loop\_forever()

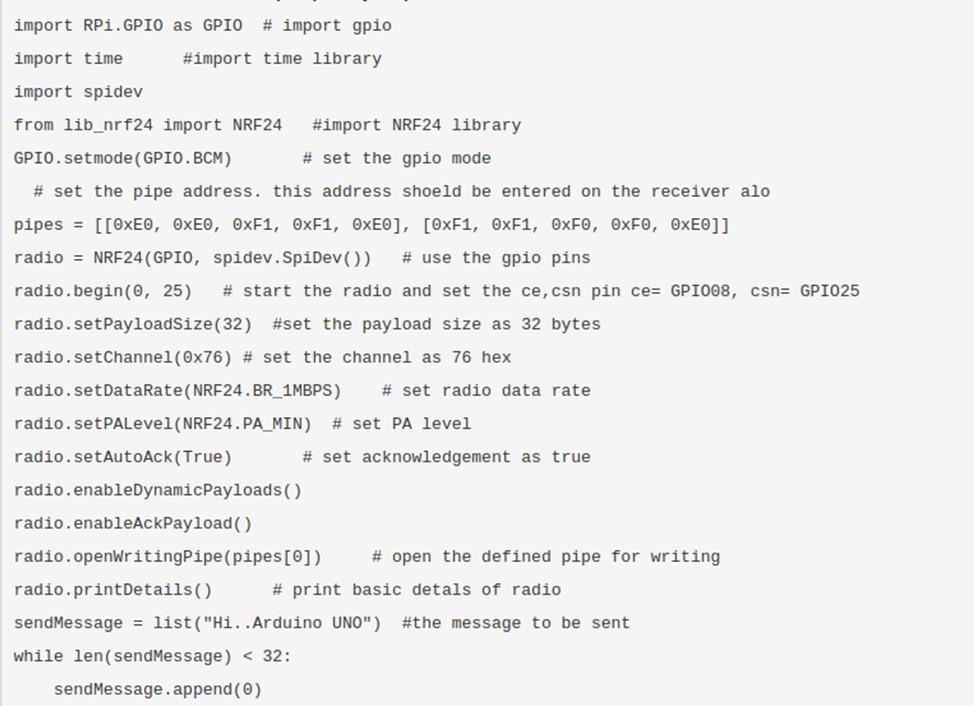
Insert AWS Credentials into Program

* + 1. Install Paho-MQTT Client Library 5.Testing MQTT Client on AWS

1. Setup MQTTFx Test Client
2. Reading data on MQTTFx from AWS IoT

# Establish wireless communication between 2 Raspberry pi using the NRF module

Wireless communication using NRF



# Establish Wireless communication between 2 Arduino using Bluetooth modules

For this experiment we have used an Arduino to interface a Bluetooth device with it. after that connect the Bluetooth to mobile and in Bluetooth console mobile app, send string data. which will be shown in serial console in Arduino.

Components:

Breadboard x 1, Raspberry Pi x 1, Jumper Wires

Source Code:

#include <SoftwareSerial.h> SoftwareSerial EEBlue(10, 11); // RX | TX void setup()

{

Serial.begin(9600);

EEBlue.begin(9600); //Default Baud for comm, it may be different for your Module.

Serial.println("The bluetooth gates are open.\n Connect to HC-05 from any other bluetooth device with 1234 as pairing key!.");

}

void loop()

{

// Feed any data from bluetooth to Terminal. if (EEBlue.available()) Serial.write(EEBlue.read());

// Feed all data from terminal to bluetooth if (Serial.available()) EEBlue.write(Serial.read());

}

# Establishing client server communication

Python-TCP-Server.py code: import socketserver

class Handler\_TCPServer(socketserver.BaseRequestHandler):

The TCP Server class for demonstration.

Note: We need to implement the Handle method to exchange data with TCP client.

def handle(self):

# self.request - TCP socket connected to the client self.data = self.request.recv(1024).strip()

print("{} sent:".format(self.client\_address[0])) print(self.data)

# just send back ACK for data arrival confirmation self.request.sendall("ACK from TCP Server".encode())

if name == " main ":

HOST, PORT = "localhost", 9999

# Init the TCP server object, bind it to the localhost on 9999 port

tcp\_server = socketserver.TCPServer((HOST, PORT), Handler\_TCPServer) # Activate the TCP server.

# To abort the TCP server, press Ctrl-C.

tcp\_server.serve\_forever()

Python-TCP-Client.py code: import socket

host\_ip, server\_port = "127.0.0.1", 9999 data = " Hello how are you?\n"

# Initialize a TCP client socket using SOCK\_STREAM

tcp\_client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) try:

# Establish connection to TCP server and exchange data tcp\_client.connect((host\_ip, server\_port))

tcp\_client.sendall(data.encode())

# Read data from the TCP server and close the connection received = tcp\_client.recv(1024)

finally:

tcp\_client.close()

print ("Bytes Sent: {}".format(data))

print ("Bytes Received: {}".format(received.decode()))

# Sniffing messages communicated between client and server

This can be done using Wireshark.

* + Open Wireshark
  + Select the network interface you want to sniff
  + Click on start button
  + Open your web browser and type in <http://www.techpanda.org/>
  + [The login email is admin@google.com](mailto:admin@google.com) and the password is Password2010
  + Click on submit button
  + A successful logon should give you the dashboard
  + Go back to Wireshark and stop the live capture
  + Filter for HTTP protocol results only using the filter textbox
  + Locate the Info column and look for entries with the HTTP verb POST and click on it
  + Just below the log entries, there is a panel with a summary of captured data. Look for the summary that says Line-based text data: application/x-www-form- urlencoded
  + You will be able to view the plaintext values of all the POST variables submitted to the server via HTTP protocol.

# Configure ESP8266 using Arduino IDE and configure GPIOs to Run the Hello World Program ( Using LED )

In order to use Arduino IDE to program the NodeMCU, you have to introduce it to the software at first.

To do this copy the following code and follow the steps below: <http://arduino.esp8266.com/stable/package_esp8266com_index.json> Step1:

Choose Preferences in the File menu and enter the copied code in Additional Board Manager URLs part. Then press OK.

Step2:

Search the word ESP8266 in Boards>boards manager from the Tools menu. Then install ESP8266 boards. After complete installation, you will see the INSTALLED label on ESP8266 boards. After these two steps, you can see ESP8266 based boards such

as NodeMCU in your Arduino IDE boards list, and you can choose your desired board to upload the code.

LED Blinking:

void setup() { pinMode(D0, OUTPUT);

}

void loop() { digitalWrite(D0, HIGH); delay(250); digitalWrite(D0, LOW); delay(250);

}

Output:

The LED will blink in a delayed manner. LED connected to the pin D0 is used in this experiment.

# Configure ESP8266 using Arduino IDE and display characters in the Seven Segment Display

void setup() { pinMode(D0, OUTPUT); pinMode(D1, OUTPUT); pinMode(D2, OUTPUT);

pinMode(D3, OUTPUT); pinMode(D4, OUTPUT);

}

void loop() { digitalWrite(D0, HIGH); digitalWrite(D1, HIGH); digitalWrite(D2, HIGH); digitalWrite(D3, HIGH); digitalWrite(D4, HIGH); delay(250); digitalWrite(D0, LOW); digitalWrite(D1, LOW); digitalWrite(D2, LOW); digitalWrite(D3, LOW); digitalWrite(D4, LOW); delay(250);

}

Output:

It will display number 6 on the 7 segment LED panel.