IoT LAB FILE

1. 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.

Raspberry Pi Imager

Step 2: Choose OS

OS Selection within Raspberry Pi Imager

Step 3: Choose SD Card

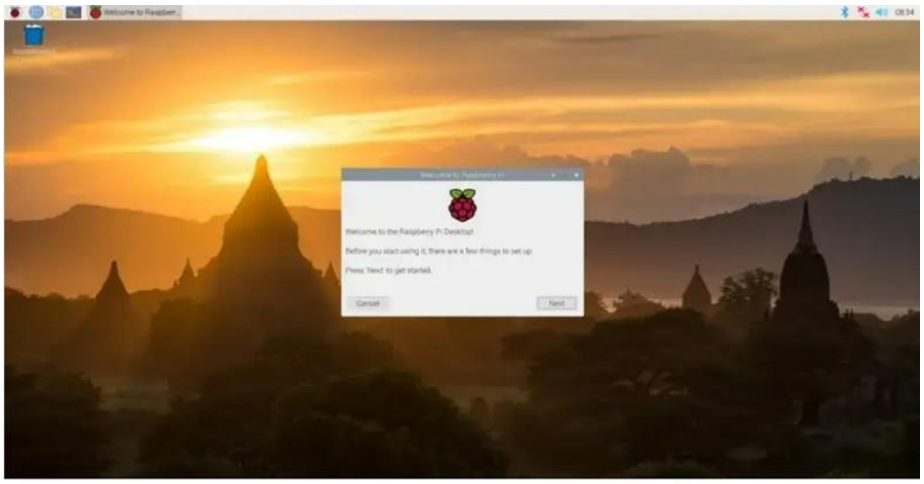
Select SD Card

Step 4: Write to SD Card

Write Process Started

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.



Welcome to Raspberry Pi

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

1. LDR Sensor

Code:

```
importRPI.GPIOasGPIO
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:

ValuesofLDRdependuponthelightexposedonit.Thesevaluesaredisplayedontheserial monitor.

2. 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()
                #timedifferencebetweenstartandarrival
                TimeElapsed=StopTime-StartTime
                 # multiply with the sonicspeed(34300 cm/s)#
                anddivideby2,becausethereand backdistance
                =(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.

3. IR Sensor

Code :

```
importRPi.GPIOasGPIO
importtime

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:

"ObjectDetected"messageisdisplayedwhenanIRsensor detectsanobject.

4. 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:
        if notno_rain.is_active:
                print("its raining - get the washing in!")
                buzz_now(5)
        sleep(1)
```

Output:

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

5. 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.

3. Uploading values of IoT sensors in the cloud

EnableSSH and I2C

1. ConnectPitothemonitor,keyboard,andmouse.
2. Start PiandthensignintoRaspberryPiOSbyusingpiastheusernameand raspberryas thepassword.
3. ClicktheRaspberryicon>Preferences>RaspberryPiConfiguration.
4. OntheInterfacestab,setSSHandI2CtoEnable,andthenclickOK.
5. Connecttheraspberrypitoasensor.
6. ConnectPiwithnetwork.

4. Implement IoT Core service to deploy sensor values on AWS Cloud

1. Browse AWS IoT Core

2.Register the Device in AWS

3. 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

4. Install Paho-MQTT Client Library
5. Testing MQTT Client on AWS
6. Setup MQTTFx Test Client
7. Reading data on MQTTFx from AWS IoT

5. Establish wireless communication between 2 Raspberry pi using the NRF module

Wireless communication using NRF

```
import RPi.GPIO as GPIO  # import gpio
import time      #import time library
import spidev
from lib_nrf24 import NRF24   #import NRF24 library
GPIO.setmode(GPIO.BCM)       # set the gpio mode
  # set the pipe address. this address shoeld be entered on the receiver alo
pipes = [[0xE0, 0xE0, 0xF1, 0xF1, 0xE0], [0xF1, 0xF1, 0xF0, 0xF0, 0xE0]]
radio = NRF24(GPIO, spidev.SpiDev())   # use the gpio pins
radio.begin(0, 25)   # start the radio and set the ce,csn pin ce= GPIO08, csn= GPIO25
radio.setPayloadSize(32)  #set the payload size as 32 bytes
radio.setChannel(0x76) # set the channel as 76 hex
radio.setDataRate(NRF24.BR_1MBPS)    # set radio data rate
radio.setPALevel(NRF24.PA_MIN)  # set PA level
radio.setAutoAck(True)       # set acknowledgement as true
radio.enableDynamicPayloads()
radio.enableAckPayload()
radio.openWritingPipe(pipes[0])     # open the defined pipe for writing
radio.printDetails()      # print basic detals of radio
sendMessage = list("Hi..Arduino UNO")  #the message to be sent
while len(sendMessage) < 32:
    sendMessage.append(0)
```

6. 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());

}
```

7. 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()))
```

8. 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 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.

9. 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. Theninstall 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.

10. 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.