IOT Material

COMMANDS

- Creating new file in VMWARE
  Nano filename.py
- Running file
  python3 filename.py
- Creating sqlite db
  sqlite3 databasename

Thingspeak

Sign in -> verification done in gmail -> continue (click in thingspeak) ->after successfully signed in ->channels ->my channel->new channel->give name and feild name

Save channel->copy chanel id value

Device tab->mqtt->select channel

-> Add device (last option) -> download credinetial in plain text
Support tab-> examples->search->mqtt basic->scroll down and select last to fourth(publish using web socket in raspberry pi)

For installing paho:
pip install paho-mqtt

```
import paho.mqtt.publish as publish
import psutil
import string

channel_ID = "1799852"

mqtt_host = "mqtt3.thingspeak.com"

mqtt_client_ID = "BDIiNR4CIxcPDBQzBBYLNwg"
mqtt_username = "BDIiNR4CIxcPDBQzBBYLNwg"
mqtt_password = "qbNDlRyj3xyUqbkEkg49okTV"

t_transport = "websockets"
t_port = 80

topic = "channels/" + channel_ID + "/publish"
```

```
while (True):
      # get the system performance data over 20 seconds
      cpu_percent = psutil.cpu_percent(interval=20)
      ram_percent = psutil.virtual_memory().percent
      #build the payload string
      payload = "field1=" + str(cpu_percent) + "&field2=" +
str(ram_percent)
      #attempt to publish this data to the topic
      try:
            print("Writing Paylod = ", payload," to host: ", mqtt_host, "
clientID= ",mqtt_client_ID, "username= ",mqtt_username, " PWD
",mqtt_password)
            publish.single(topic, payload, hostname=mqtt_host,
transport=t_transport, port=t_port, client_id=mqtt_client_ID,
auth={'username':mqtt_username,'password':mqtt_password})
      except (keyboardInterrupt):
            break
      except Exception as e:
            print (e)
```

Firebase.com

Sign in -> click on get started-> create/add project->write project name ->disable google analytics in the next step->continue->build.->realtime database

Create database->select singapore->

Click next->select start in test mode

Click Enable
(database is created on cloud)

For keys-: go to project overview->select web </> icon-> add name(same as project name)->register app->script will be displayed with id and keys.
.py program(shruti account)

- Run this command

sudo pip3 install pyrebase

```
import pyrebase
Config = {
        "apiKey": "AIzaSyDJ7Ir1QewmxyTZjXjO8meu6LtihcsrpuQ",
        "authDomain": "mydemo-af3e4.firebaseapp.com",
        "databaseURL": "https://mydemo-af3e4-default-rtdb.asia-
southeast1.firebasedatabase.app",
        "projectId": "mydemo-af3e4",
```

```
        "storageBucket": "mydemo-af3e4.appspot.com",
        "messagingSenderId": "628906293168",
        "appId": "1:628906293168:web:578a9db8b337b0f157c875",
        "measurementId": "G-TL6WMB3BQG"
};

firebase = pyrebase.initialize_app(Config);

storage =firebase.storage()
database =firebase.database()
a = 6
b = 60
print (a)
database.child("DB _object_name")
data = {"key1" :a,"key2" :b}
database.set(data)
no1 =33
no2 =33
database.child("DB _object_name2")
data1 = {"num1" :no1,"num2" :no2}
print (no1)
database.set(data1)
```

IOT_MQTT_to_rasbearyPI_publish_subscribe

```
import paho.mqtt.client as mqtt

def on_message(client, userdata, msg):
   print(msg.topic+" "+str(msg.payload))
   client.publish(topic="it", payload="TestingPayload", qos=1, retain=False)

#broker_url = "mqtt.eclipse.org"
broker_url = "broker.emqx.io"

broker_port = 1883

client = mqtt.Client()
client.on_message = on_message
client.connect(broker_url, broker_port)

client.subscribe("TestingTopic", qos=0)
#client.publish(topic="TestingTopic1", payload="TestingPayloadh1" ,qos-0, retain$
#print(msg.topic+" "+str(msg.payload))
client.loop_forever()
```

Program 2

```
import paho.mqtt.client as mqtt

def on_message(client,userdata,msg):
    print(msg.topic+" "+str(msg.payload))
    client.publish(topic="it",payload="Testing payload
7",qos=1,retain=False)

broker_url = "broker.emqx.io"
broker_port = 1883
client=mqtt.Client()
client.on_message=on_message
client.connect(broker_url,broker_port)

client.subscribe("TestingTopic",qos=0)

#client.publish(topic="TestingTopic1",payload="testing
payload",qos=1,retain=False)

client.loop_forever()
```

IOT_sqlite_to_mqtt_publish_and_sunscribe

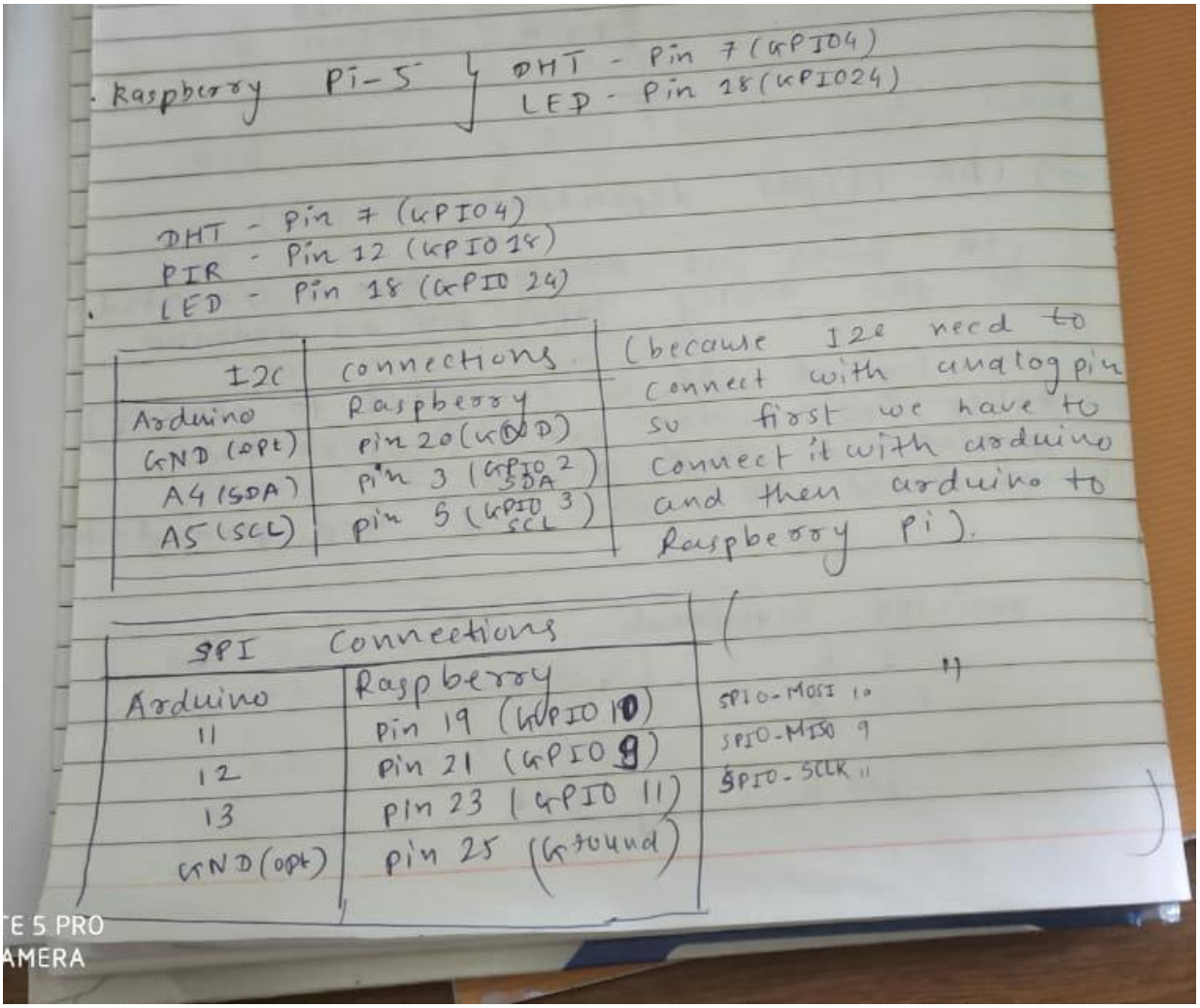
external_exam.py

```
import sqlite3
import time
import datetime
import paho.mqtt.client as mqtt

broker_url="broker.emqx.io"
broker_port=1883
client=mqtt.Client()
client.connect(broker_url,broker_port)

connection = sqlite3.connect("hospitalitydb.db")
crsr = connection.cursor()

sql_command = """ CREATE TABLE IF NOT EXISTS tbl_demo(id INTEGER PRIMARY
KEY AUTOINCREMENT,name VARCHAR(50),age INTEGER,date DATE) """
crsr.execute(sql_command)
print("table created...")

for i in range(1):
    print("Enter Name : ")
    name=input()
```

```
    print("Enter Age : ")
    age=input()
    date = datetime.datetime.now()
    print(name)
    crsr.execute("INSERT INTO tbl_demo(name,age,date)
values(?,?,?)",(name,age,date))
    connection.commit()

crsr.execute("select * from tbl_demo")
allrecord=crsr.fetchall()
print(allrecord)

crsr.execute("select name from tbl_demo")
ans=crsr.fetchall()
i=ans

for i in range (len(ans)):
    print(ans[i])
    client.publish(topic="it",
                payload="name: " + str(ans[i]),
                qos=0,
               retain=False)
    print("msg published...")
```

Displaydata.py

```
import sqlite3

connection = sqlite3.connect("hospitalitydb.db")
crsr = connection.cursor()

crsr.execute("select * from hospital")
ans=crsr.fetchall()
print(ans)
```



I2C code

Arduino code

```
#include<Wire.h>
int i2cData =0x56;
//LED on pin 13
const int ledPin=13;
int c;
void setup(){
//join I2C bus as slave with address 8
Wire.begin(0x8);
Serial.begin(9600);
Wire.onReceive(receiveEvent);
Wire.onRequest(sendData);
pinMode(ledPin,OUTPUT);
digitalWrite(ledPin,LOW);
```

```
}

void receiveEvent(int howMany)
{
while(Wire.available())
{
c=Wire.read();
digitalWrite(ledPin,c);


}
}

void loop()
{
delay(1000);
if(c<0x02){
Serial.println(c);
c=0x02;
}
}
void sendData()
{
Wire.write(i2cData);
}
```

Raspberry file

```
from smbus import SMBus
addr= 0x8
bus = SMBus(1)

numb = 1
print("Enter 1 for ON or 0 for OFF")
while numb==1:

 ledstate = input(">>>> ")


 if ledstate == "1":
    bus.write_byte(addr,0x1)
 elif ledstate == "0":
    bus.write_byte(addr,0x0)
 else:
    numb=0
```

SPI

Arduino code

```
#include <SPI.h>
char buf[100];
byte c=0,b=0;
volatile byte pos;
volatile boolean processing;
byte i;

void setup(void)
{
Serial.begin(115200); //uart speed
pinMode(MISO,OUTPUT);
pinMode(MOSI,INPUT);
SPCR |= _BV(SPE);
pos=0;
processing=false;
SPI.attachInterrupt();

}
ISR(SPI_STC_vect){
c= SPDR;
processing = true;
}
void loop(void){
if(processing){
Serial.println(c);
processing = false;
SPDR =i;
i= i+1;
}
}
```

Raspberry py code

```
import spidev
import time

spi = spidev.SpiDev(0,0)
spi.open(0,0)
msg = 0xAA
spi.max_speed_hz = 115200
```

```
while 1:
      spi.writebytes([2,4])
      y = spi.readbytes(1)
      print (y)
      time.sleep(0.5)
```

Ultrasonic

wire confi:
vcc => 5v
trig =>gpio21 =>40 no pin
echo =>gipio20 =>38no pin
gnd => 39 no pin

```
import RPi.GPIO as GPIO
import time
TRIG=21
ECHO=20
GPIO.setmode(GPIO.BCM)
while True:
    print("distance measurement in progress")
    GPIO.setup(TRIG,GPIO.OUT)
    GPIO.setup(ECHO,GPIO.IN)
    GPIO.output(TRIG,False)
    print("waiting for sensor to settle")
    time.sleep(0.2)
    GPIO.output(TRIG,True)
    time.sleep(0.00001)
    GPIO.output(TRIG,False)
    //high pulse=1
    //low pulse=0
    while GPIO.input(ECHO)==0:
        pulse_start=time.time()
    while GPIO.input(ECHO)==1:
        pulse_end=time.time()
    pulse_duration=pulse_end-pulse_start
    distance=pulse_duration*17150
    distance=round(distance,2)
    print("distance:",distance,"cm")
    time.sleep(2)
```

PIR

pin Config:
- pin =Grd
+ pin = 5 v
data (middle) = gpio18 =12 no
led = 1 ->grd
2=> gpio24 pin no18

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(24,GPIO.OUT)
GPIO.setup(18,GPIO.IN)

while(True):
        myin=GPIO.input(18)
        if myin==True:
                print("Motion Detected")
                time.sleep(1)
                GPIO.output(24,True)
                time.sleep(0.5)
                GPIO.output(24,False)
                time.sleep(0.5)
```

IR

GND-pin6
Vcc-pin4
Data-pin8 (GPIO14)

```
import RPi.GPIO as IO
#import time
IO.setwarnings(False)
IO.setmode(IO.BOARD)
IO.setup(3,IO.OUT)
IO.setup(8,IO.IN)

while(True):
      if(IO.input(8) == True):
            print ("Obstacle Detected")
            IO.output(3,True)
      else:
            print ("Obstacle not Detected")
            IO.output(3,False)
```

LDR

Arduino code

```
int ldr;
void setup() {
// put your setup code here, to run once:
pinMode(13,OUTPUT);
Serial.begin(9600);

}

void loop() {
// put your main code here, to run repeatedly:
        ldr = analogRead(A0);
        if(ldr < 50){
                digitalWrite(13,HIGH);
        }
        else{
                digitalWrite(13,LOW);
        }
        Serial.println(ldr);
        delay(100);
}
```

USB

Error commands

sudo chmod 666 /dev/ttys0
sudo chmod 666 dev/ttyACM0

rasberry-arduino
(8pin)UART tx-rx
(10pin)rx-tx
gnd-gnd

//single pc

```
import serial

#if __name__=='__main__':
ser=serial.Serial('/dev/ttyACM0',9600,timeout=1)
ser.flush()
while True:
if ser.in_waiting>0:
line=ser.readline().decode('utf-8').rstrip()
#rstrip for storing whole line in buffer 4-500 characters
print(line)

//2px tx_rx


import serial

#if __name__=='__main__':
ser=serial.Serial('/dev/ttyACM0',9600,timeout=1)
ser.flush()
while True:
if ser.in_waiting>0:
line=ser.readline().decode('utf-8').rstrip()
#rstrip for storing whole line in buffer 4-500 characters
print(line)

//Arduino code
void setup()

{

Serial.begin(9600);

}

3T75c8qU

void loop()

{

Serial.println("Hello From Arduino!");

delay(1000);

}
```

UART with tx-rx

Arduino	RPi
Tx	**8**
Rx	**10**
Gnd	**6**

```
import serial

from time import sleep
ser=serial.Serial('/dev/ttyS0',9600)
while True:

        received_data=ser.read()

        sleep(0.03)

        data_left=ser.inWaiting()

        received_data+=ser.read(data_left)

        print(received_data)
```

Note : run **sudo chmod 666 /dev/ttyS0** to give persmission to use ttyS0 before executing code

USB UART with RPi direct communication

Arduino	RPi
Tx	**Tx**
Rx	**Rx**
Gnd	**Gnd**

```
import serial

if name=='main':

        ser=serial.Serial('/dev/ttyACM0',9600,timeout=1)
        ser.flush()
        while True:

                if ser.in_waiting>0:

                        line=ser.readline().decode('utf-8').rstrip()

                        print(line)
```

Note : run **sudo chmod 666 /dev/ttyACM0** to give persmission to use ttyS0 before executing code

Arduino Code For USB UART

```
void setup()

{
        Serial.begin(9600);
}

void loop()

{
        Serial.println("Hello From Arduino!");
        delay(1000);

}
```

---

1. Connect IR sensor with RPi .sense object using IR sensor and update timing in SQLite db also publish msg “object detected” on MQTT topic name :: “IR_Sensor” and display time on screen

IR	RPi
Data	3 (without using LED) 8 (while using LED)
Vcc	2
Gnd	6

If using LED

LED	RPi
-	ground
+	3

```
import RPi.GPIO as IO

import time

import sqlite3

import paho.mqtt.client as mqtt



connection = sqlite3.connect('test.db')

cursor = connection.cursor()
```

```
cursor.execute('''CREATE TABLE irTB ( sensorData VARCHAR(100), currentTime
TIMESTAMP);''')

insertQuery = """INSERT INTO irTB  VALUES (?, ?, ?);"""


IO.setwarning(False)

IO.setmode(IO.BOARD)

IO.setup(8,IO.IN)

IO.setup(3,IO.OUT)


while 1:

        if(IO.input(8)==True):

                print("Obstacle Detected!!")

                cursor.execute(insertQuery, (“Obstacle Detected”, currentDateTime))

                connection.commit()


                broker_url = "broker.emqx.io"

                broker_port = 1883


                client = mqtt.Client()

                client.connect(broker_url,broker_port)


                client.publish(topic="IR_Sensor", payload=”object detected”, qos=0, retain=
                False)

                IO.output(3,True)

        else

                print("Obstacle Not Detected")
```

```
        IO.output(3,False)
cur.execute("select * from irTB ");
ans=cur.fetchall();
for row in ans:
        print(ans);
connection.close()
```

2. Sense light intensity using LDR on arduino board and send sensed light intensity value to RPi using i2c bus on RPi received content through i2c bus is update in SQLite DB and display on screen

Requirement : Arduino, LDR, RPi, cable
I2c :

Arduino	RPi
A4-SDA	3-SDA
A5-SCL	5-SCL
GND	20

LDR : connects with arduino because RPi has no analog convertor pin

LDR	Arduino
2 joint leg	A0
Register	GND
LDR	5v

LDR arduino code : (Slave)

```
#include<Wire.h>
int ldr; //set A0 (Analog Input) for LDR
void setup(){
        //join i2c bus as slave with address 8
        Wire.begin(0x8);
        Serial.begin(9600);
        //call receiveEvent when data received
```

```
        Wire.onReceive(receiveEvent);
        //call get request from master
        Wire.onRequest(sendData);
        pinMode(13,OUTPUT);
        digitalWrite(13,LOW);

    //fn that executes whenever data is received from master
    void receiveEvent (int howMany){
        while(Wire.available()){
            c=Wire.read();  //receive byte as a character
            digitalWrite(13,c);
        }
    }
    void loop(){
        ldr=analogRead(A0);     //reads the value of LDR(light)
        if(ldr<50)
        {       digitalWrite(13,HIGH);          }
        else
        {       digitalWrite(13,LOW);           }

        Serial.println(ldr);
        delay(100);

//prints the value of ldr to Serial monitor

    } //Serial.println(value); void
    sendData(){
```

```
        Wire.write(ldr);
}
```

I2CPY : (Master)

```
from smbus import SMBus
import time
import sqlite3
conn=sqlite3.connect("testDB.db");
cur=conn.cursor();

conn.execute('''CREATE TABLE ldrTB(LDRDATA TEXT NOT NULL);''')
conn.commit();

addr = 0x8
bus = SMBus(1)
print("Enter 1 for ON or 0 for OFF")
while 1:
        a.      =
        bus.read_byte_data(addr,0x1) print (y)

        cur.execute("'INSERT INTO ldrTB(LDRDATA) VALUES(?)'",(y));
        conn.commit()
cur.execute("'select * from ldrTB'");
ans=cur.fetchall();
for row in ans:
        print(ans);
conn.commit()
```

```
conn.close()
```

3. Connect ultrasonic sensor with RPi. Sense object distance and update distance and timing in

   SQLite DB also publish measured distance value on MQTT topic name :: “ultra_Sensor” and display on screen

Ultrasonic	RPi
ECHO	38(GPIO 20)
TRIG	40(GPIO 21)
VCC	2
GND	39

```
import RPi.GPIO as GPIO

import time

import sqlite3

import paho.mqtt.client as mqtt

TRIG=16

ECHO=20


connection = sqlite3.connect('test.db')

cursor = connection.cursor()

cursor.execute('''CREATE TABLE ultraTB (sensorData VARCHAR(100), currentTime
TIMESTAMP);''')

insertQuery = """INSERT INTO ultraTB VALUES (?, ?, ?);"""


GPIO.setmode(GPIO.BCM)


while True:

        print("distance measurement in progress");
```

```
GPIO.setup(TRIG,GPIO.OUT)

GPIO.setup(ECHO,GPIO.IN)


GPIO.output(TRIG,False)

print("waiting for sensor to settle");


time.sleep(0.2)

GPIO.output(TRIG,True)


time.sleep(0.00001)

GPIO.output(TRIG,False)


while GPIO.input(ECHO)==0:

        pulse_start=time.time()

while GPIO.input(ECHO)==1:

        pulse_end=time.time()


pulse_duration=pulse_end-pulse_start

dist=pulse_duration*17150


dist=round(dist,2)

print("distance:",dist,"cm")

time.sleep(0.1)


cursor.execute(insertQuery, (dist, currentDateTime))

connection.commit()
```

```
        broker_url = "broker.emqx.io"

        broker_port = 1883


        client = mqtt.Client()

        client.connect(broker_url,broker_port)


        client.publish(topic="UltraMP", payload=dist, qos=0, retain= False)
cur.execute("select * from ultraTB ");

ans=cur.fetchall();

for row in ans:

        print(ans);

connection.close()
```

4. Sense light intensity using LDR on arduino board and send sensed light intensity value to RPi using SPI bus on RPi received content through SPI bus is update in SQLite DB and display on screen

Requirement : Arduino, LDR, RPi, cable
SPI :

Arduino	RPi
11	19-MOSI
12	21-MISO
13	23-SCLK
GND	25-ground

LDR : connects with arduino because RPi has no analog convertor pin

LDR	Arduino
2 joint leg	A0
Register	GND
LDR	5v

5 . Use virtual box to create SQLite DB with name manualrollno DB. Insert following fields in DB table “Sequence no, name, age, gender” manually. Insert atleast 5 to 6 records in DB. Once

DB record entry completed then access it one by one and publish on topic name :: Virtual_A and display on screen

```
import paho.mqtt.client as mqtt

import sqlite3


connection=sqlite3.connect("manualrollno.db");

cur=connection.cursor();


create_tbl="""CREATE TABLE IF NOT EXISTS TEST(sequenceno INT, name TEXT, age INT, gender TEXT)"""

cur.execute(create_tbl);

connection.commit();

//insert 5-6 data manually

broker_url = "broker.emqx.io"

broker_port = 1883


client = mqtt.Client()

client.connect(broker_url, broker_port)


cur.execute("select * from TEST");

ans=cur.fetchall();

for row in ans:

        print(ans);

        client.publish(topic=" Virtual_A ",payload=ans,qos=0,retain=False)
```

```
connection.commit();

client.loop_forever()
```

6. Connect PIR sensor with RPi . sense object using PIR sensor and update timing in SQLite db also publish msg “PIR object detected” on MQTT topic name :: “PIR_Sensor” and display time on screen

PIR	RPi
GND	6
VCC	2
OUT(DATA)	12

If using LED

LED	RPi
-	Ground
+	Output-18

PIRpy :

```
import RPi.GPIO as GPIO

import time

import sqlite3

import paho.mqtt.client as mqtt


GPIO.setmode(GPIO.BCM)

GPIO.setup(24,GPIO.OUT)

GPIO.setup(18,GPIO.IN)


connection = sqlite3.connect('test.db')

cursor = connection.cursor()

cursor.execute('''CREATE TABLE pirTB (

   sensorData VARCHAR(100),
```

```
   currentTime TIMESTAMP);''')
insertQuery = """INSERT INTO pirTB
   VALUES (?, ?, ?);"""
while(True):
        myin=GPIO.input(18)
        if myin==True:
                print("Motion Detected")
                cursor.execute(insertQuery, (“Motion Detected”, currentDateTime))
                connection.commit()

                broker_url = "broker.emqx.io"
                broker_port = 1883

                client = mqtt.Client()
                client.connect(broker_url,broker_port)

                client.publish(topic="PIR_Sensor", payload=”motion detected”, qos=0,
                retain= False)

                time.sleep(1)
                GPIO.output(24,True)
                time.sleep(0.5)

                GPIO.output(24,False)
                time.sleep(0.5)
```

```
        cur.execute("select * from pirTB ");

        ans=cur.fetchall();

        for row in ans:

                print(ans);

        connection.close()
```

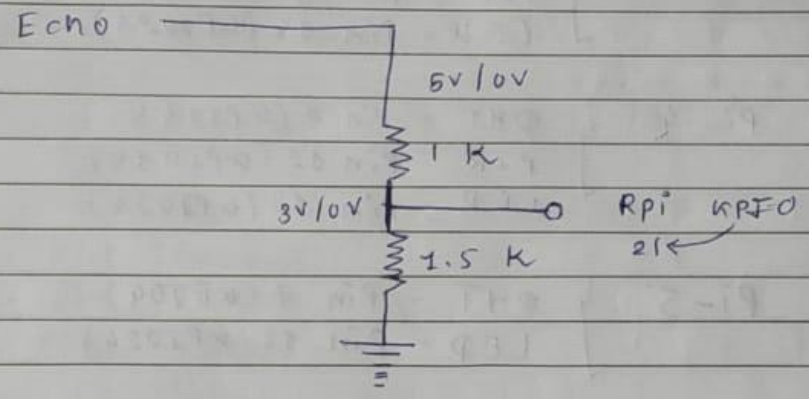
ADD Data in CSV File

```
import csv

data = [
      ['Albania',545822,'AL','ALB'],
      ['American Samoa',56335,'AS','ASM'],
      ['India',3524,'IN','IND']
]

header=['name','area','countrycode2','countrycode3']

with open('countries.csv','a',encoding='UTF8') as f:
      writer=csv.writer(f)
      #write the header
      writer.writerow(header)

      #write data
      #writer.writerow(data)
      for i in range(0, len(data)) :
            writer.writerow(data[i])

f.close()
rows=[]
with open('countries.csv','r') as file:
      csvreader=csv.reader(file)
      header1=next(csvreader)
      for row in csvreader:
            rows.append(row)
print(header1)
print(rows)

file.close()
```

→ Untra-sonic sensor connections

VCC - pin 4 (5V)
Echo - pin 38 (GPIO 20)
Trig - pin 40 (GPIO 21)
GND - Pin 6 (ground)

Now for Echo register connections



→ LDR (Light dependent resistor)

LDR gives out Analog voltage so connected to the analog input pin on arduino.

One leg of LDR is connected to VCC (5V) other to analog pin 0 (A0) on the Arduino. A 100K resistor is also connected to same leg and grounded

resistor required

1 K } untrasonic
1.5 K }

100 K } LDR

→ LDR connection to Arduino

LDR 2 connections ~~one~~ resistor 2 connections
1st with [1 K] resistor's 1 leg
2nd with Vcc (5 V)

LDR + resistor connection → A0 (Analog Pin)
Resistor's 2nd leg is grounded in Arduino.

⇒ It gives value in ~~se~~ serial monitor.

→ IR sensor connection with Raspberry Pi

IR sensor } Human body sense
obstuele "

PIR sensor } only human body sense

IR sensor have 3 - connections.
GND → Pin 6 (ground)
Vcc → Pin 4 (5V)
Data → Pin 8 (GPIO14)

We wing LED to indicate detection of obstacle
LED connections Anode (+) → Pin 3 (GPIO2)
Kathode (-) → Pin 9 (Ground)

PIR sensor have 3- connections.
GND → Pin 6 (ground)
Vcc → Pin 2 (Vcc) 5V
Data → Pin 12 (GPIO 18)

sqlLITE commands

---
sqlite 3 install and use
---

install: sudo apt-get install sqlite3

creating new database or entering into existing database: sqlite3 newDB.db

creating table: create table tableName (id integer, name text);

show table: .table

insert: insert into demo (demoID, demoName) values(1, "a1");

show: select * from demo;

.exit or .quit to exit

Node js with mqtt

/*install following command*/

npm init -y

npm install mqtt mosca

/*Replace the following condition in node_modules/jsonschema/lib/validator.js at line no 109*/

```
if((typeof schema == 'boolean' && typeof schema == 'object') || schema === null){
    throw new SchemaError('Expected `schema` to be an object or boolean');
}
```

/*Create broker.js*/

//============================broker.js=========================

//MQTT Broker

```
//Help us to create mqtt
var mosca = require("mosca");

var settings = {port:1234};

var broker = new mosca.Server(settings);

broker.on("ready",()=>{
   console.log("Broker is ready....");
});

============================================================
=============

//============================publisher.js=======================
=========

var mqtt = require("mqtt");
var client = mqtt.connect("mqtt://localhost:1234");

const TOPIC = "MY_TOPIC"

const message = "Hello there I am sending the message";

client.on("connect",()=>{
   setInterval(()=>{
      client.publish(TOPIC,message);
      console.log(`Message Received ${message}`)
   },3000);
});

============================================================
=============

//============================subscriber.js======================
=========

var mqtt = require("mqtt");
var client = mqtt.connect("mqtt://localhost:1234");

var TOPIC = "MY_TOPIC"

//client connect with broker then it will subscribe
client.on("connect",()=>{
```

```
    client.subscribe(TOPIC);
});

//Sending the message
client.on("message",(topic,message)=>{
    message = message.toString();
    console.log(`Message sent : ${message}`);
});
```

==========

mqtt settings:

mqtt client name: client (any)
protocol: mqtt / tcp
host: localhost:1234 (the one that is in the code)
normally host is: broker.emqx.io

when node broker.js will run, then only the connetion will eshtablish and it will show connected in mqtt box or else it will show connection error

//Run the files in different terminal

```
node broker.js
node publisher.js
node subscriber.js
```