IOT Material

COMMANDS

* Creating new file in VMWARE

Nano filename.py

* Running file

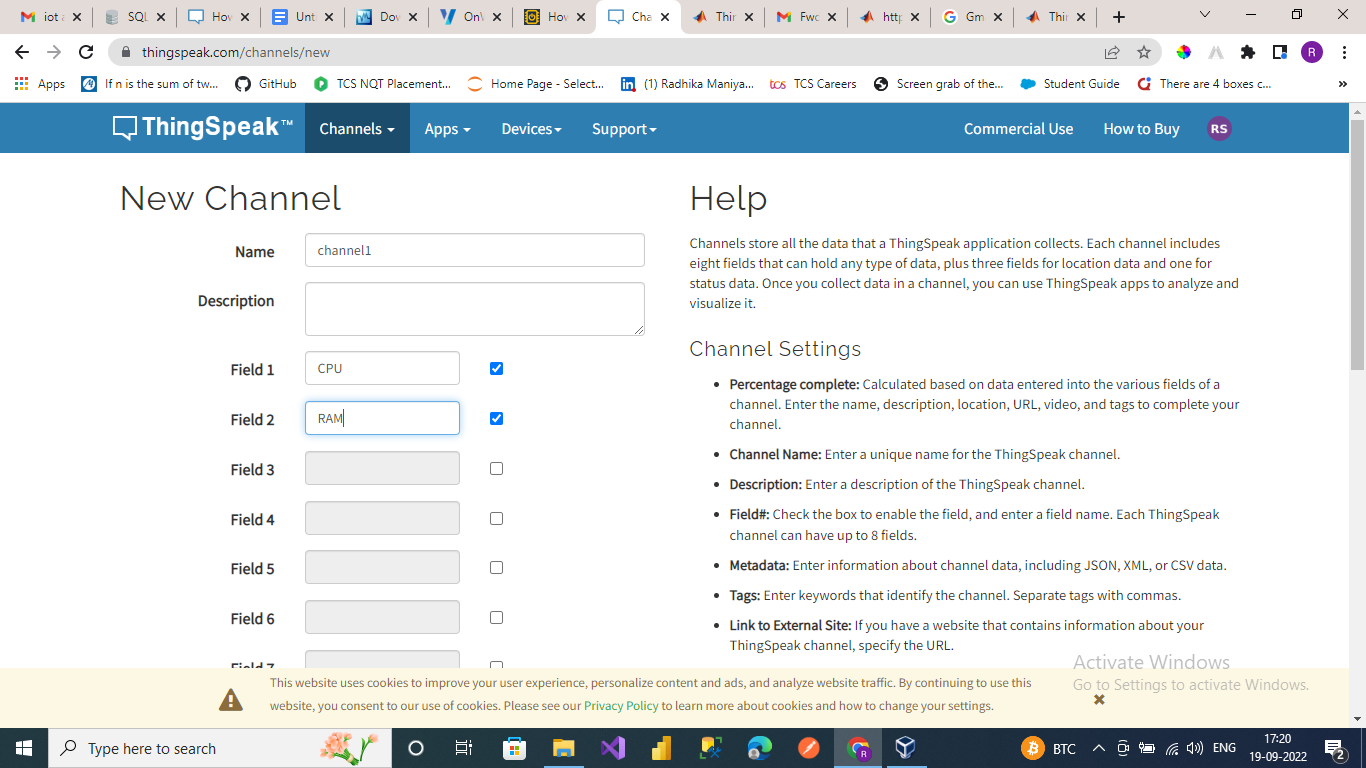
python3 filename.py

* Creating sqlite db

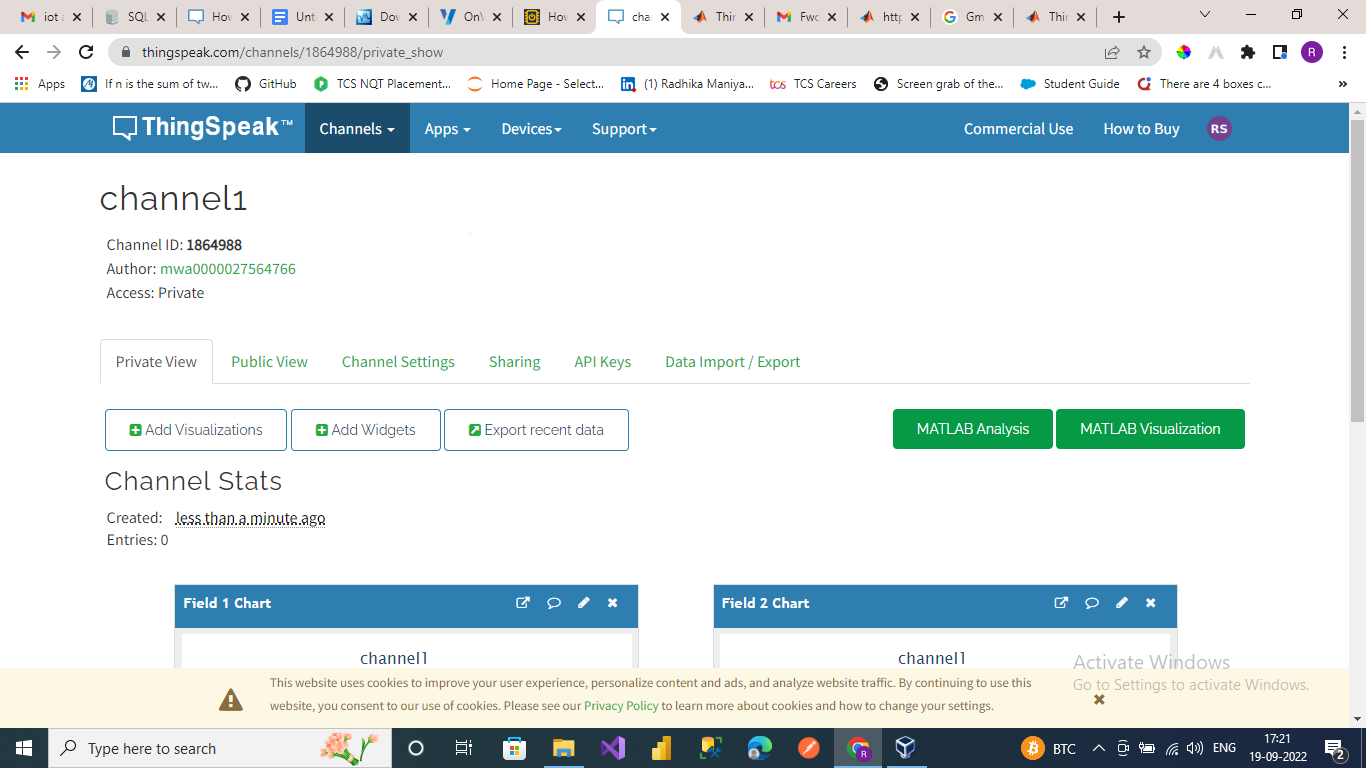
sqlite3 databasename

Thingspeak

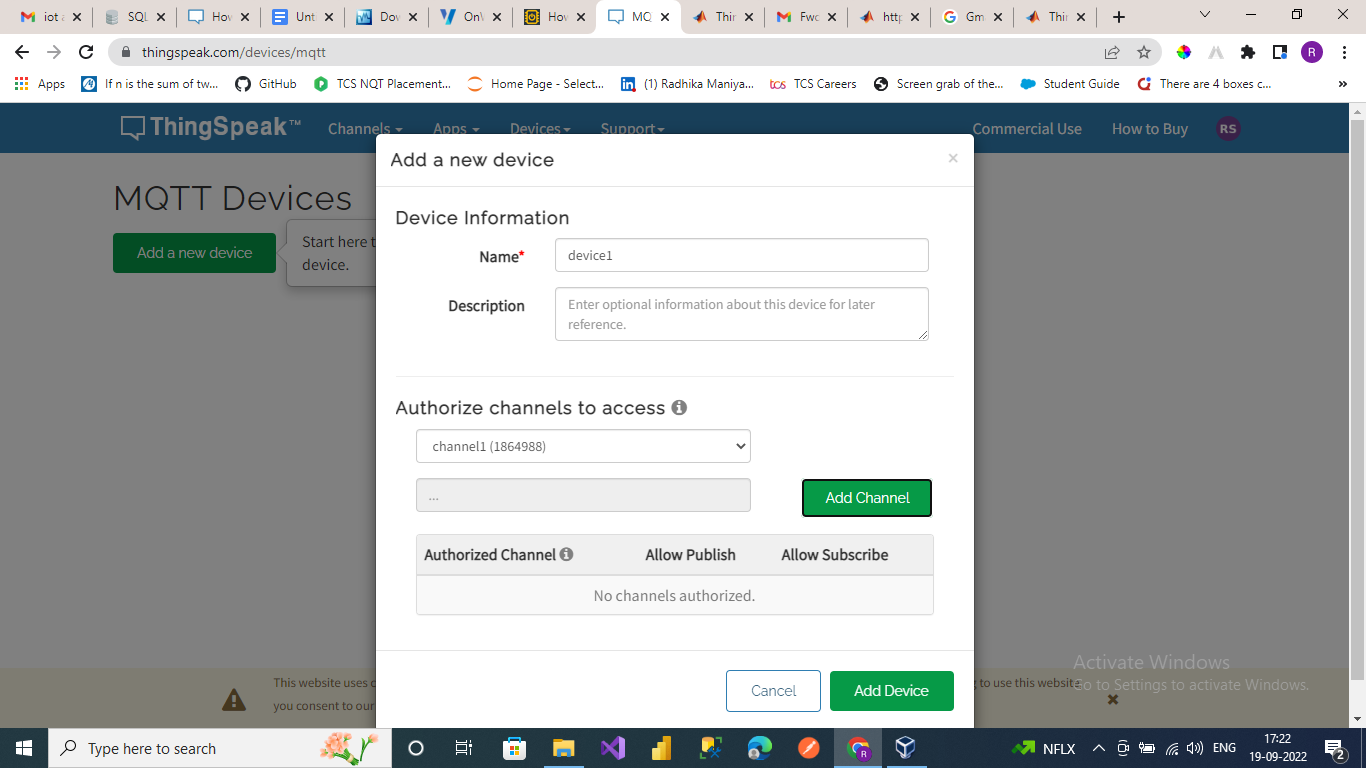
Sign in -> verification done in gmail -> continue (click in thingspeak) ->after successfully signed in ->channels ->my channel->new channel->



Save channel->

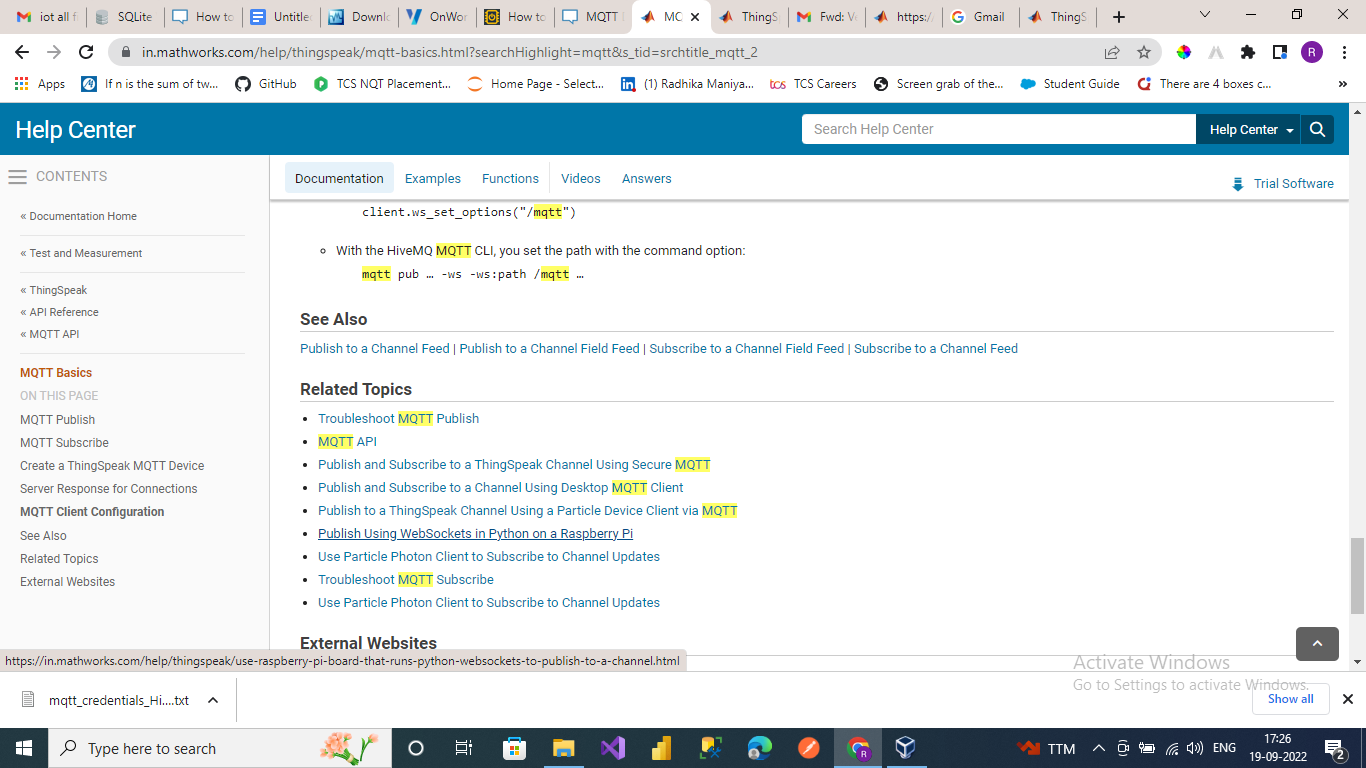


Device tab->mqtt->



-> Add device (last option) -> download credinetial in plain text

Support tab-> examples->search->mqtt basic->scroll down and select



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):

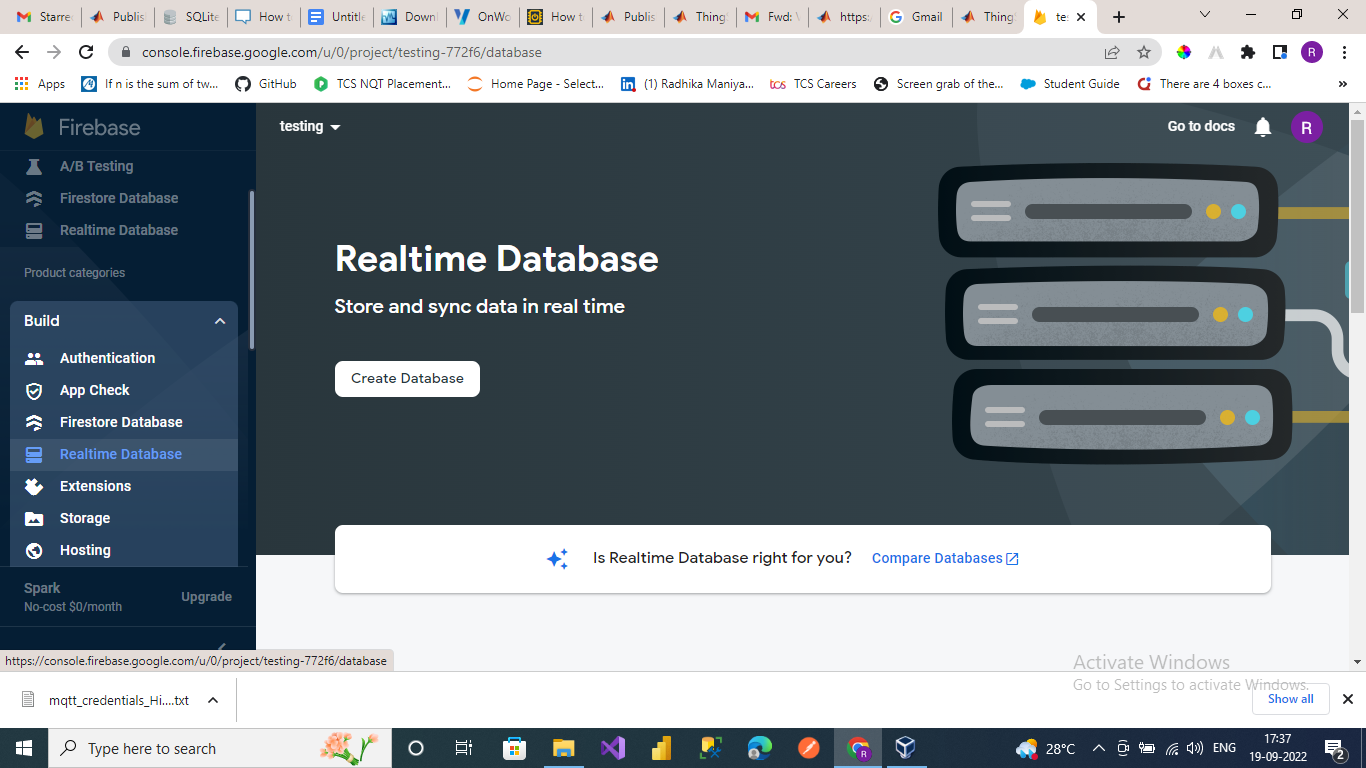
break

except Exception as e:

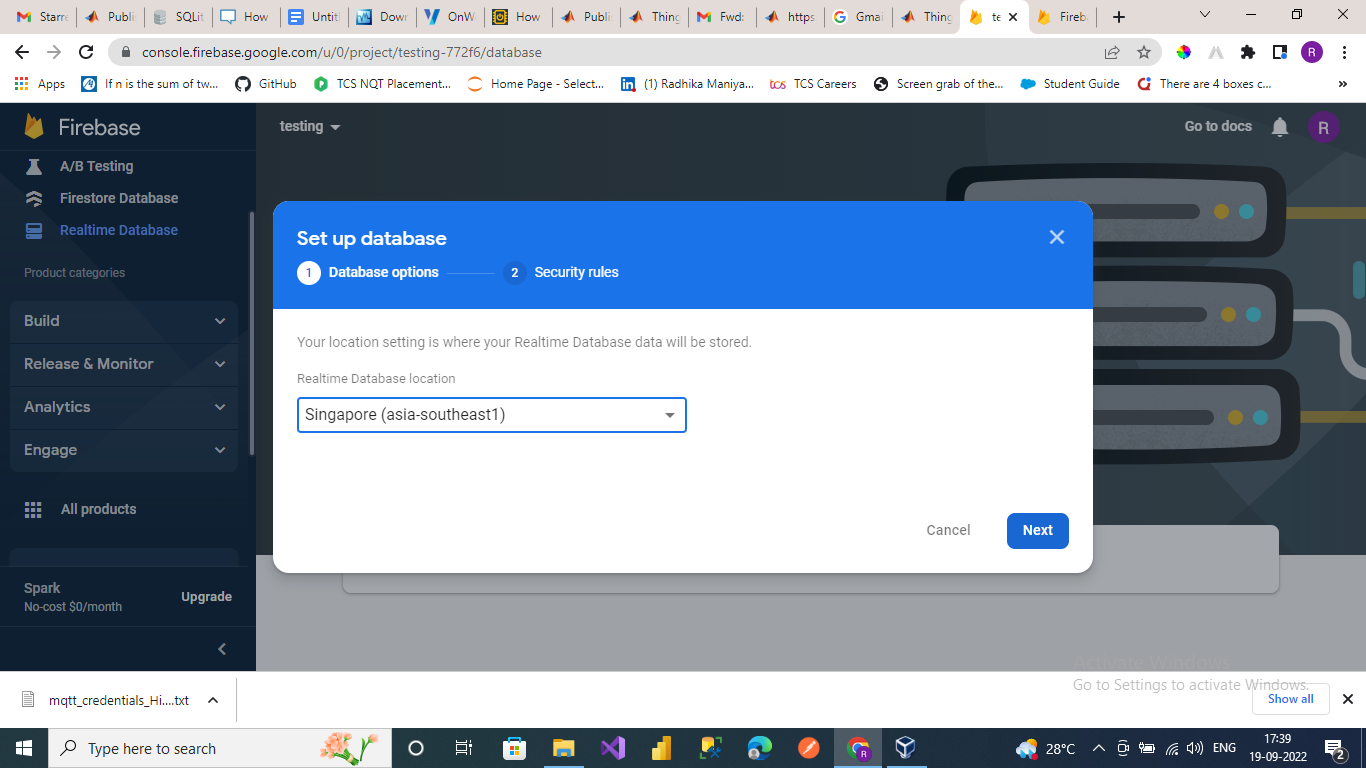
print (e)

Firebase.com

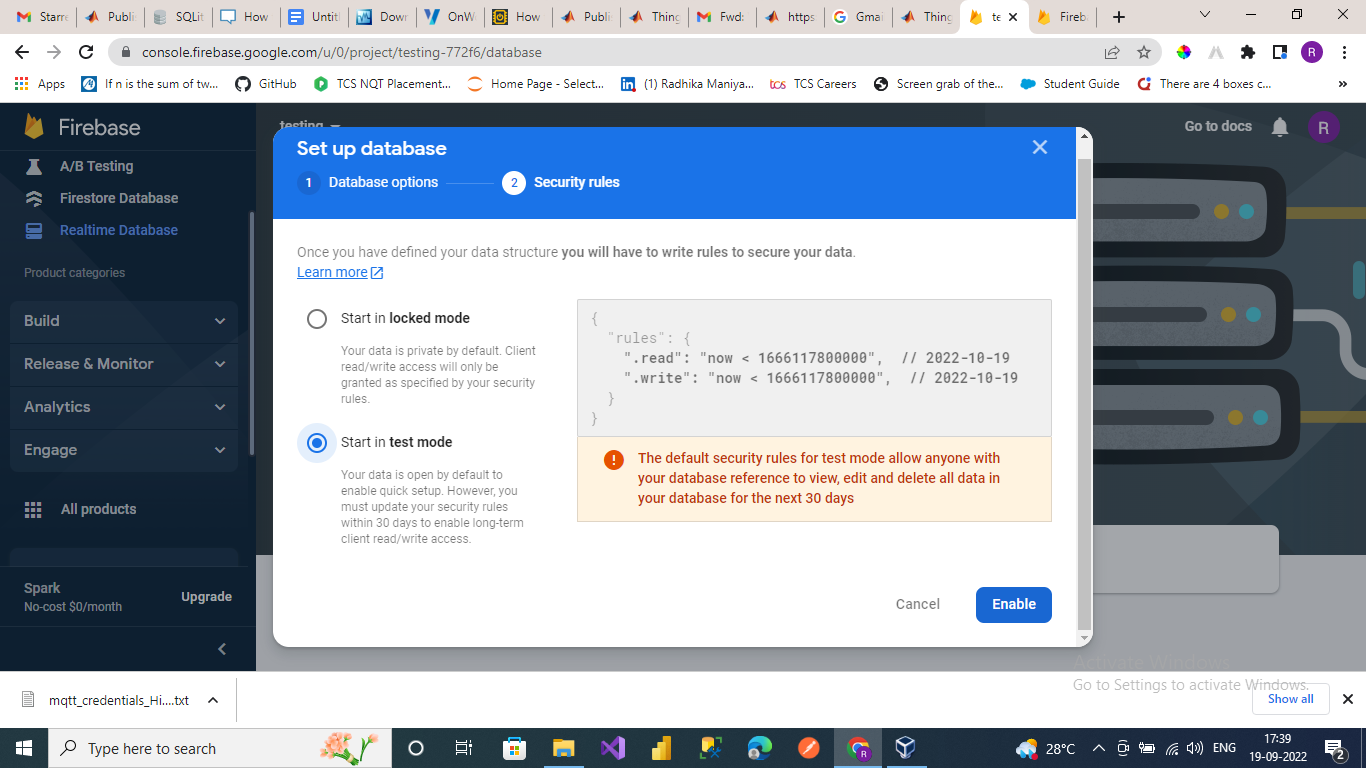
Sign in -> click on get started-> create/add project->write project name ->disable google analytics in the next step->continue->build.



Create database



Click next



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

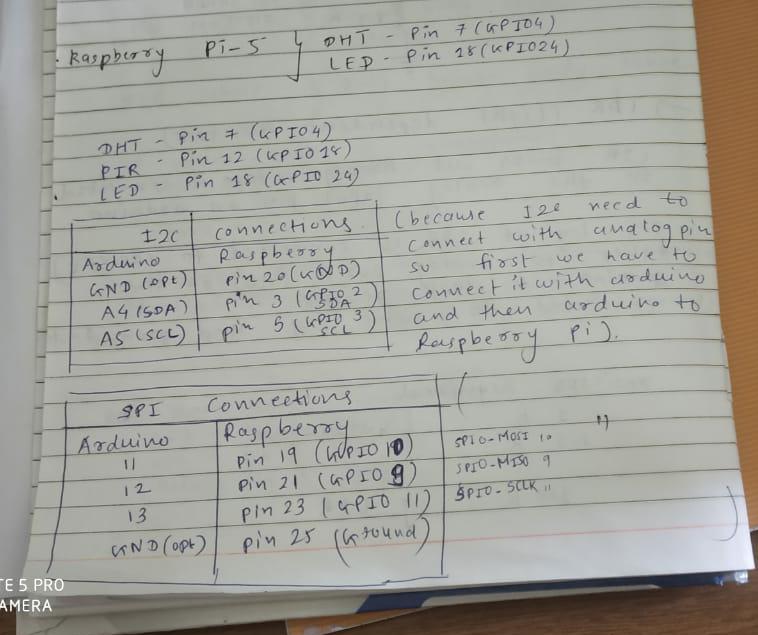
connection = sqlite3.connect("hospitalitydb.db")

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](http://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](http://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);

}

**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)

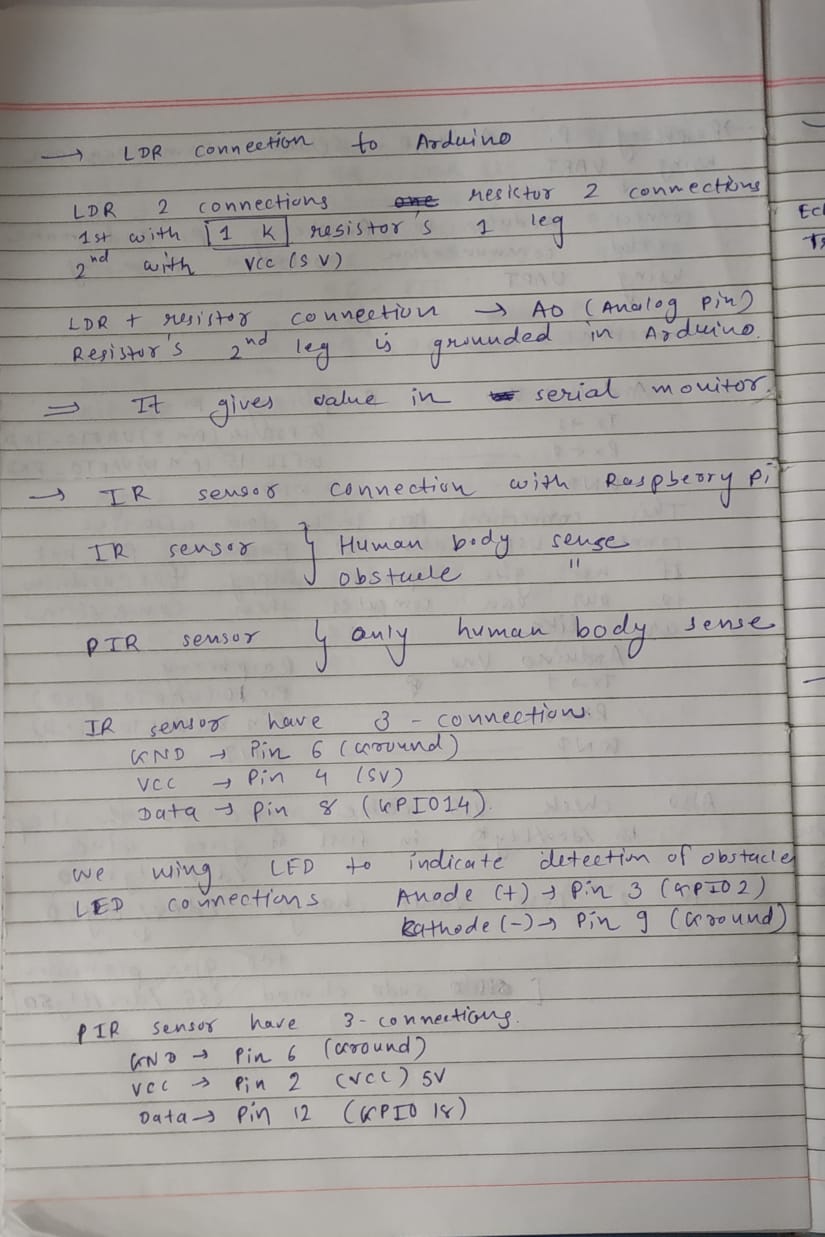
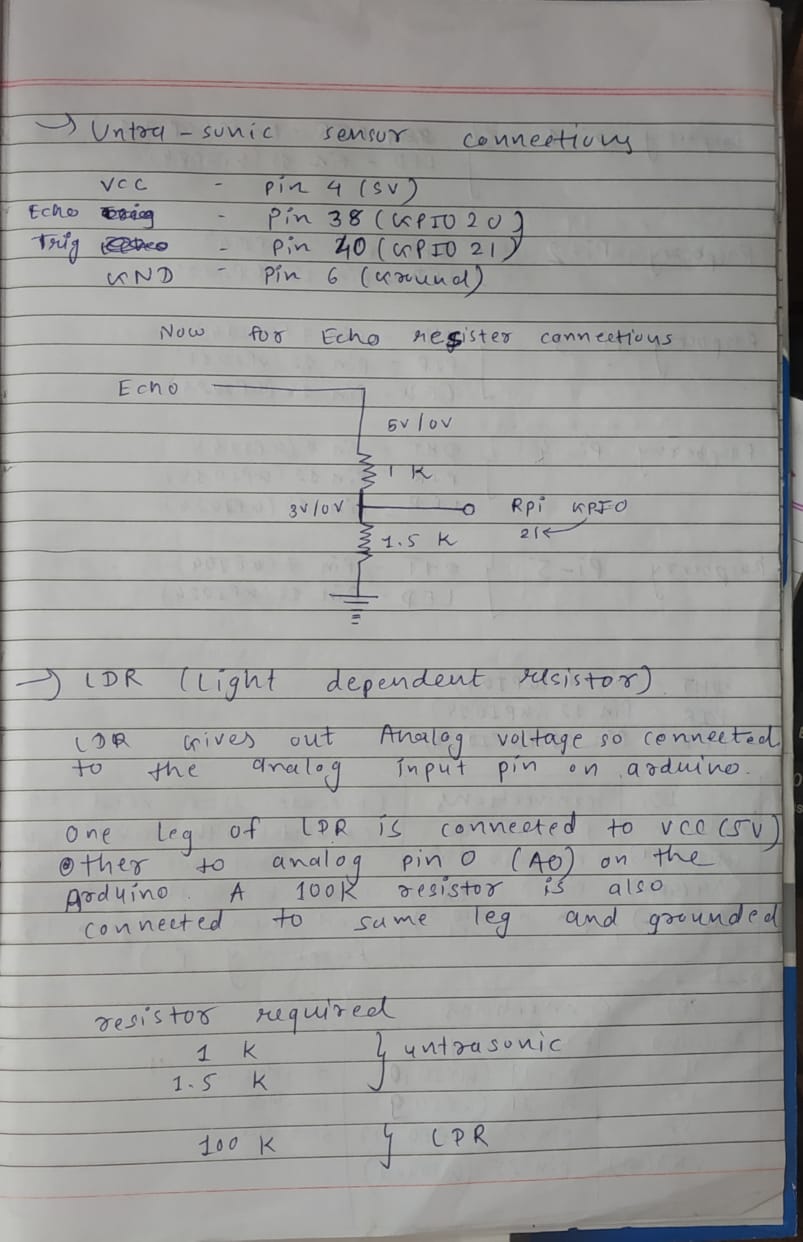
for row in csvreader:

rows.append(row)

print(header1)

print(rows)

file.close()



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