Lab 4

**Getting started with Open Source Cloud Server Upload any Sensor Data to cloud Server ThingSpeak Complete Tutorial Read, Write with Example**

## **ThingSpeak:**

There are two ways one is hard way without library and another with library

### **Video Tutorial can be Found Below Link:**

1. Hard way



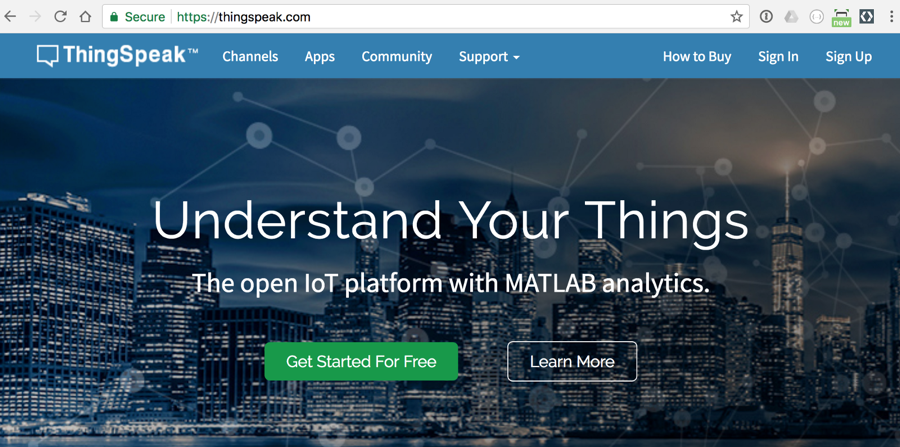
[**https://www.youtube.com/watch?v=jYeUS7JxmA8**](https://www.youtube.com/watch?v=jYeUS7JxmA8)

1. **Easy way:**

****

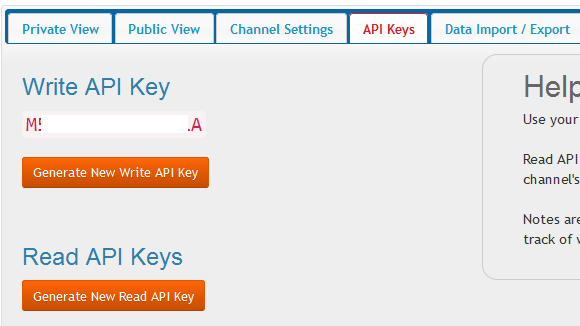
<https://www.youtube.com/watch?v=HM7_Urqoric>

# Step 1: Go thingspeak website and Create your Account



After creating your Account we ned to copy Read and Write API Keys shown in image below

# Step 2: Copy Credentials all all API keys



# We are Going to Learn Both ways

1. Upload the Data with 3 lines of code using Library that i have developed
2. Upload using URLlib and request Module

# Method 1: Harder way

Write Data

import urllib.request

import requests

import threading

import json

import random

def thingspeak\_post():

threading.Timer(15,thingspeak\_post).start()

val=random.randint(1,30)

URl='https://api.thingspeak.com/update?api\_key='

KEY=' -------WRITE KEY----------'

HEADER='&field1={}&field2={}'.format(val,val)

NEW\_URL = URl+KEY+HEADER

print(NEW\_URL)

data=urllib.request.urlopen(NEW\_URL)

print(data)

if \_\_name\_\_ == '\_\_main\_\_':

thingspeak\_post()

# Method 2 Easy way with Three Line of Code

## **important Note:**

Assume your name of your python file is main and the code that i am about to show you is the library developed to do jon in 3 lines of code. Paster the Entire 500 Line of code in your File called main and all way down start writing your code

or

copy the library ie python file in your working Directory and you can simple say from nameofpythonfile import \*

# Assume your Work Directory is

Foldername ---- IoTclass

> main.py

> masterclass.py

in your main Python File you will say from masterclass import \*

if that dosent work for some reason you can also copy entire 500 Line of library code in main.py and all the way down start writing your code

## Code

from masterclass import \*

temperature = 22

humidity = 11

w\_key = 'AZRMN9ZP5FKLXYLR'

r\_key = '17FSCQ4FTX6V4VWM'

channel\_id = 83234

ob = Thingspeak(write\_api\_key=w\_key, read\_api\_key=r\_key, channel\_id=channel\_id)

ob.post\_cloud(value1=temperature,value2=humidity)

# Sample Code using Master class.

### Use this Template for Coding !

try:

import numpy as np

from masterclass import \*

except:

print("Library not Found ! ")

x = []

y1 = []

y2 = []

def sensor\_random\_value():

"""

Write you Raspberry pi Sensor Code Inside this

:return: Comma Seperated Values

"""

x = np.random.randint(0, 20, None) # create random value which resenble to sensor data Live

y = np.random.randint(0, 20, None) # create a another random variable to store Data Sensor 2

my\_sensor = "{},{}".format(x, y) # create a Fake Data which represent the hardware data

return my\_sensor

def mymain():

x,y = sensor\_random\_value().split(',')

w\_key = 'AZRMN9ZP5FKLXYLR'

r\_key = '17FSCQ4FTX6V4VWM'

channel\_id = 83234

ob = Thingspeak(write\_api\_key=w\_key, read\_api\_key=r\_key, channel\_id=channel\_id)

ob.post\_cloud(value1=x,value2=y)

if \_\_name\_\_ == "\_\_main\_\_":

mymain()

**Task 1: Upload DHT-11 Sensor Data to Cloud Server !**

try:

import Adafruit\_DHT

from masterclass import \*

except:

print("Library not Found !")

def sensor\_value():

# Example using a Raspberry Pi with DHT sensor

# connected to GPIO23.

pin = 4

sensor = Adafruit\_DHT.DHT22

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, pin)

if humidity is not None and temperature is not None:

print('Temp={0:0.1f}\*C Humidity={1:0.1f}%'.format(temperature, humidity))

data = '{},{}'.format(temperature,humidity)

return data

else:

print('Failed to get reading. Try again!')

def my\_main():

x,y = sensor\_value().split(',')

w\_key = 'AZRMN9ZP5FKLXYLR'

r\_key = '17FSCQ4FTX6V4VWM'

channel\_id = 83234

ob = Thingspeak(write\_api\_key=w\_key, read\_api\_key=r\_key, channel\_id=channel\_id)

ob.post\_cloud(value1=x,value2=y)

if \_\_name\_\_ == "\_\_main\_\_":

my\_main()

# Reading Data From Cloud

# print(ob.read\_cloud(result=3))

## **Task: Try to interface Distance Sensor and Upload the Data to Cloud Server.**

# For Those who want to see my Library Code

# <https://github.com/soumilshah1995/IoTMaster>

|  |
| --- |
| """"  Author : Soumil shah  Email : soushah@my.bridgeport.edu  Version 1.0.1  update: Changes made to class added private var and url and read url  cannot be changed !  """  try: # import the important library  from urllib import request  from urllib.request import urlopen  import threading # import threadding  import json # import json  import random # import random  import requests # import requests for web API  import ssl  import geocoder # for Locations  import datetime # for date time  from twilio.rest import Client # for Sms  from serial import Serial # for arduino  import numpy as np  from Adafruit\_IO import Client # for Adafruit IO  import os  from os import system # For Text to speech  import paho.mqtt.publish as publish  import sqlite3 # for Database  import paho.mqtt.publish as publish  import urllib # for web Api  except:  print("No Library Found")  class Thingspeak(object): # define a class called Thingspeak  def \_\_init\_\_(self, write\_api\_key = None, read\_api\_key=None, channel\_id=0):  """  :param write\_key: takes a string of write api key  :param timer: can take integer values  """  # self.url = 'https://api.thingspeak.com/update?api\_key='  # self.read\_url = 'https://api.thingspeak.com/channels/{}/feeds.json?api\_key='.format(channel\_id)  self.write\_key = write\_api\_key  self.channel\_id = channel\_id  self.read\_api\_key = read\_api\_key  # Private Var cannot change  self.\_\_url = 'http://api.thingspeak.com/update?api\_key'  self.\_\_read\_url = 'https://api.thingspeak.com/channels/{}/feeds.json?api\_key='.format(channel\_id)  self.feild1 = []  self.feild2 = []  def post\_cloud(self, value1, value2):  try:  """  :param value1: can be interger or float  :param value2: can be interger or float  :return: updated to cloud storage  """  URL = self.\_\_url  KEY = self.write\_key  HEADER = '&field1={}&field2={}'.format(str(value1), str(value2))  NEW\_URL = str(URL) + "=" + str(KEY) + str(HEADER)  print(NEW\_URL)  context = ssl.\_create\_unverified\_context()  data = request.urlopen(NEW\_URL,context=context)  print(data)  except:  print('could not post to the cloud server ')  def read\_cloud(self, result=1):  try:  """  :param result: how many data you want to fetch accept interger  :return: Two List which contains Sensor data  """  URL\_R = self.\_\_read\_url  read\_key = self.read\_api\_key  header\_r = '&results={}'.format(result)  new\_read\_url = URL\_R + read\_key + header\_r  data = requests.get(new\_read\_url).json()  field1 = data['feeds']  for x in field1:  self.feild1.append(x['field1'])  self.feild2.append(x['field2'])  return self.feild1, self.feild2  except:  print('read\_cloud failed !!!! ')  class IfTTT(object):  def \_\_init\_\_(self, eventname='', key=''):  self.eventname = eventname  self.Key = key  self.\_\_Url = 'https://maker.ifttt.com/trigger/{}/with/key/'.format(self.eventname)  self.New\_Url = self.\_\_Url + self.Key  print(self.New\_Url)  def iftt\_post(self, data1=10, data2=11):  try:  """  :param data1: pass your sensor value only integer  :param data2: pass your data as interger only  :return: True if it was successful  """  URl = self.New\_Url  Key = self.Key  payload = {'value1': data1,  'value2': data2}  requests.post(self.New\_Url, data=payload)  print("Done posted on IFTTT")  return True  except:  print('failed to post to cloud sever ! ')  class Location(object):  def \_\_init\_\_(self):  pass  def get\_locations(self):  """  :return: Lat and Long  """  try:  g = geocoder.ip('me')  my\_string=g.latlng  longitude=my\_string[0]  latitude=my\_string[1]  return longitude,latitude  except:  print('Error make sure you have Geo-Coder Installed ')  class DateandTime(object):  def \_\_init\_\_(self):  pass  @ staticmethod  def get\_time\_date():  try:  """  :return: date and time  """  my = datetime.datetime.now()  data\_time = '{}:{}:{}'.format(my.hour,my.minute,my.second)  data\_date = '{}/{}/{}'.format(my.day,my.month,my.year)  return data\_date,data\_time  except:  print('could now get date and time ')  def convert\_timestamp(self,timestamp):  timestamp = 1554506464  dt\_object = datetime.fromtimestamp(timestamp)  return dt\_object  class Weather\_details(object):  def \_\_init\_\_(self,key='', city=''):  self.city = city  self.key = key  def get\_weather\_data(self):  try:  city = self.city  key = self.key  URL='http://api.openweathermap.org/data/2.5/weather?appid={}&q={}'.format(key,city)  print(URL)  data = requests.get(URL).json()  long = data['coord']['lon']  lat = data['coord']['lat']  humidity = data['main']['humidity']  wind\_speed = data['wind']['speed']  wind\_degree = data['wind']['deg']  sunrise = data['sys']['sunrise']  sunset = data['sys']['sunset']  m1 = data['weather'][0]['description']  m2 = data['weather'][0]['main']  body = '{} {}'.format(m1, m2)  return long, lat, humidity, wind\_speed, wind\_degree, sunrise, sunset,body  except:  print('Error occured ')  class Arduino(object):  def \_\_init\_\_(self,comport='/dev/cu.usbmodem14101', baudrate=9600):  self.comport = comport  self.baudrate = baudrate  def read\_data(self):  try:  """  :return: data after reading from serial object  """  arduino = Serial(self.comport, self.baudrate,timeout=1)  data = arduino.readline().decode('ascii')  return data  except:  print("please check your com port and baud rate ")  def write\_data(self,data='1'):  try:  """    :param data: pass string 1 or 0 or any character and write if statment in Arduino  to compare and trigger events  :return: NONE  """  arduinodata = Serial(self.comport, self.baudrate,timeout=1)  data\_send = data.encode('utf-8')  arduinodata.write(data\_send)  except:  print("cannot write data please check com port and baud rate ")  class Adafruit\_cloud():  def \_\_init\_\_(self, username='', Aio\_key=''):  self.username = username  self.Aio\_key = Aio\_key  self.aio = Client(self.username,self.Aio\_key)  self.feed\_name = []  def adafruit\_send(self,feed\_name='',data=''):  try:  sensor = self.aio.feeds(feed\_name)  self.aio.send\_data(sensor.key,data)  print('Data was uploaded ')  except:  print("cannot send Data !")  def adafruit\_get(self, feedname=''):  try:  data = self.aio.receive(feedname)  return data.value  except:  print('cannot get data ! ')  def adafruit\_feed\_list(self):  try:  feeds = self.aio.feeds()  for f in feeds:  print(f)  self.feed\_name.append(f.name)  return self.feed\_name  except:  print("cannot get feed list ! ")  class TextSpeech\_Mac(object):  def \_\_init\_\_(self):  """  This script works for mac only  """  pass  def speak(self,text):  try:  """  :param text: Takes String as Input  :return: Return None  """  system("say {}".format(text))  except:  print("This works for mac only ! ")  class Mqtt(object):  def \_\_init\_\_(self, data):  self.data= data  self.hostname="test.mosquitto.org"  self.topic = ''  def post(self):  """  POst teh  :return:  """  publish.single(self.topic, "{}".format(self.data), hostname=self.hostname)  class YoutubeSub(object):  def \_\_init\_\_(self,name ='',google\_key=''):  """  :param name: Name of Youtube Person , Google Cloud API key  :param google\_key:  """  self.name = name  self.google\_key = google\_key  self.context = ssl.\_create\_unverified\_context()  def get\_subscriber(self):  """  :return: Youtube Sub Count  """  self.url = 'https://www.googleapis.com/youtube/v3/channels?part=statistics&forUsername=' \  + self.name+'&key='+self.google\_key  data = urllib.request.urlopen(self.url, context=self.context).read()  subs = json.loads(data)["items"][0]["statistics"]["subscriberCount"]  subs = int(subs)  return subs  class Text\_MP3\_converter(object):  def \_\_init\_\_(self):  self.\_\_url = 'https://text-to-speech-demo.ng.bluemix.net/api/v1/synthesize?t'  self.\_\_header ={  'Accept-Encoding': 'gzip, deflate, sdch',  'Accept-Language': 'en-US,en;q=0.8',  'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_10\_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36',  'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8',  'Referer': 'http://www.wikipedia.org/',  'Connection': 'keep-alive'}  self.\_\_params = {  'text': 'hello everyone i am going to teach you python',  'voice': 'en-US\_AllisonV2Voice',  'download': True,  'accept': 'audio/mp3'}  def text\_audio(self,name ='test', text='hello world'):  """  :param name\_file:  :param text:  :return: saves MP3 File on your computer  """  try:  response = requests.get(self.\_\_url, headers=self.\_\_header,params=self.\_\_params)  with open("{}.mp3".format(name),'wb') as f:  f.write(response.content)  print("File has been downloaded on your computer with name {}".format(name))  except:  print('Error cannot convert File')  class Spotifypy(object):  def \_\_init\_\_(self,client\_id='', client\_secret='',oauth\_token=''):  self.client\_id = client\_id  self.client\_secret = client\_secret  self.oauth\_token = oauth\_token  self.\_\_header = {'Authorization': self.oauth\_token,  'Accept':'application/json',  'Content-Type':'application/json'}  def play\_song(self):  self.\_\_url\_play = 'https://api.spotify.com/v1/me/player/play'  response = requests.put(self.\_\_url\_play, headers=self.\_\_header)  print(response)  def pause\_song(self):  self.\_\_url\_pause = 'https://api.spotify.com/v1/me/player/pause'  response = requests.put(self.\_\_url\_pause, headers=self.\_\_header)  print(response)  def get\_recommendation(self):  self.\_\_url\_recommendation = 'https://api.spotify.com/v1/recommendations/available-genre-seeds'  response = requests.get(self.\_\_url\_recommendation, headers=self.\_\_header)  # print(response.json())  return response.json()  def get\_device(self):  self.\_\_url\_device = 'https://api.spotify.com/v1/me/player/devices'  response = requests.get(self.\_\_url\_recommendation, headers=self.\_\_header)  return response.json()  class Converter(object):  def \_\_init\_\_(self):  self.\_\_url = "https://www.binaryhexconverter.com/hesapla.php"  self.\_\_params = {  'fonksiyon': 'dec2hex',  'deger': 12,  'pad': False,  'v': 2  }  def bin2hex(self,binnum=1):  self.\_\_params['deger'] = binnum  self.\_\_params['fonksiyon'] = 'bin2hex'  response = requests.get(self.\_\_url,params=self.\_\_params)  return response.text  def bin2dec(self, binnum=1):  self.\_\_params['deger'] = binnum  self.\_\_params['fonksiyon'] = 'bin2dec'  response = requests.get(self.\_\_url, params=self.\_\_params)  return response.text  def bin2asc(self,binnum=1):  self.\_\_params['deger'] = binnum  self.\_\_params['fonksiyon'] = 'bin2asc'  response = requests.get(self.\_\_url, params=self.\_\_params)  return response.text  def hex2bin(self, hexnum ='A'):  self.\_\_params['deger'] = hexnum  self.\_\_params['fonksiyon'] = 'hex2bin'  response = requests.get(self.\_\_url, params=self.\_\_params)  return response.text  def hex2dec(self,hexnum ='A'):  self.\_\_params['deger'] = hexnum  self.\_\_params['fonksiyon'] = 'hex2dec'  response = requests.get(self.\_\_url, params=self.\_\_params)  return response.text  def dec2hex(self, decnum='10'):  self.\_\_params['deger'] = decnum  self.\_\_params['fonksiyon'] = 'dec2hex'  response = requests.get(self.\_\_url, params=self.\_\_params)  return response.text  def dec2bin(self, decnum='10'):  self.\_\_params['deger'] = decnum  self.\_\_params['fonksiyon'] = 'dec2bin'  response = requests.get(self.\_\_url, params=self.\_\_params)  return response.text  def dec2oct(self, decnum='10'):  self.\_\_params['deger'] = decnum  self.\_\_params['fonksiyon'] = 'dec2oct'  response = requests.get(self.\_\_url, params=self.\_\_params)  return response.text |

# Getting Started with Twilio and Raspberry pi for SMS Notification [Lab 4A]

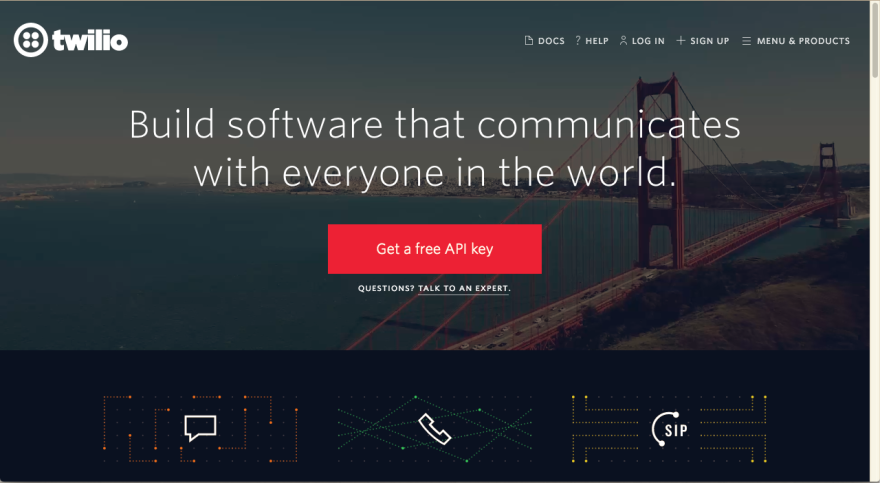
## **objective:**

In this we are going to Learn how to use Twilio API with Python and Raspberry to Send SMS when Event is Occured. Event can be When Temperature Exceeds more than Specified Value it shall send a SMS to a contact numnber.

# Step 1: Create a Account on Twilio

Go to Twilio Website and Create a Accout.

<https://www.twilio.com>

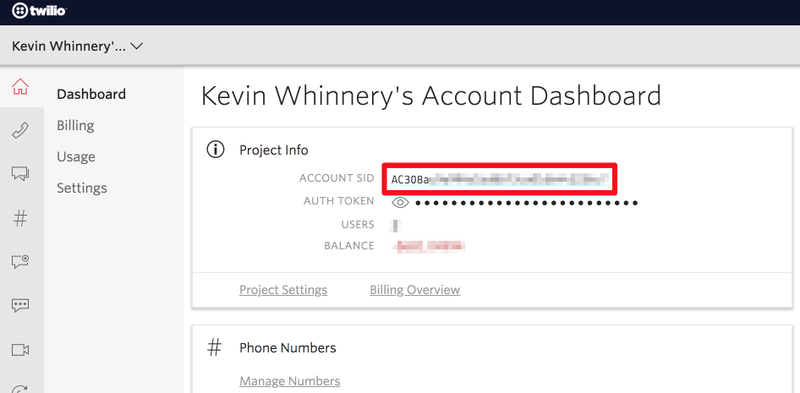


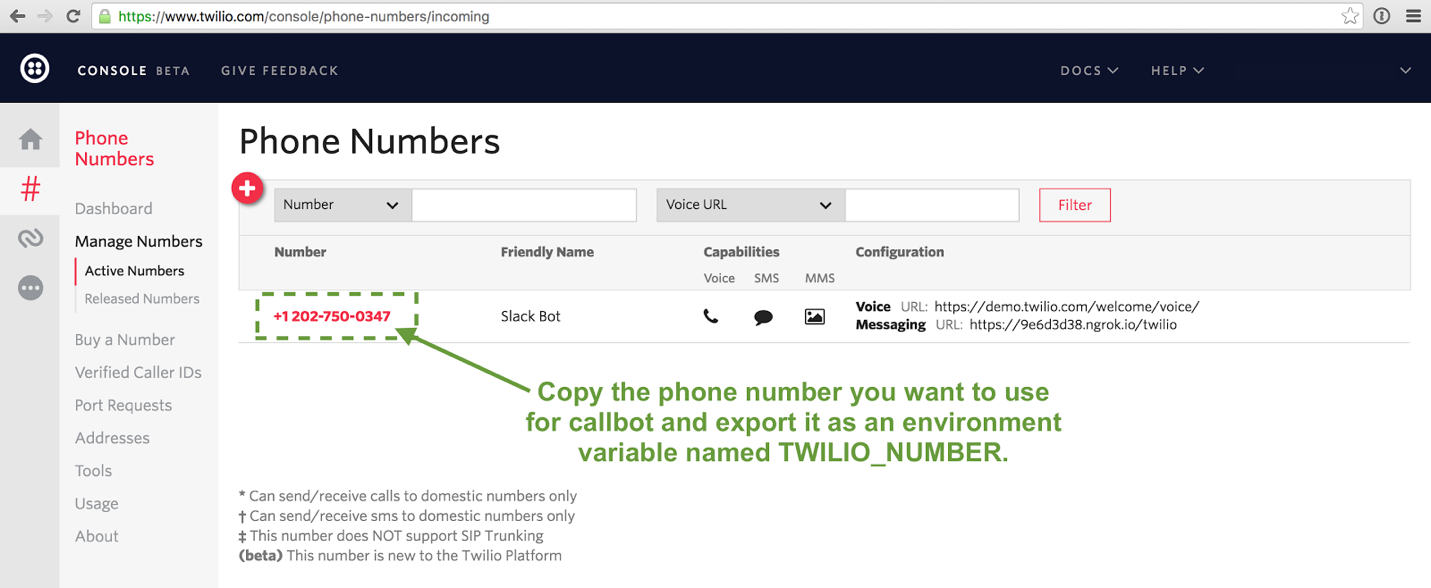
# Step 2: Get Twilo Phone Number

After you have created a account. go ahead and get a Twilio Phone Number and also we would require Twilio AUTH KEY and SSID go ahead and get the Twilio Number.



## After you get Twilo Number you should see a screen like this





# Step 3: Download Python Library for Twilio

pip install twilio.rest if pip install didnt work for you go ahead to google and and read the doccumentaion on downloading TWILIO for Python 3 +

# Step 4 : Lets Code .........

### Import the library

from twilio.rest import Client

### Define a Function to send SMS

Make sure to use try and except Block to Avoid Errors Add you SSID and AUTH Token.

# def send\_sms\_alert():

# try:

# # Define your body

# my\_body='Yo'

# # define client

# client = Client('SSID GOES HERE XXX ','AUTH TOKEN XX ')

# client.messages.create(to='+ TO XXXX ',

# from\_= '+TWILIO NUMBER XXX',

# body=my\_body)

# except:

# print('Cannot send Sms!')

### Putting all Together

# from twilio.rest import Client

# def send\_sms\_alert():

# try:

# # Define your body

# my\_body='Yo'

# # define client

# client = Client('SSID GOES HERE XXX ','AUTH TOKEN XX ')

# client.messages.create(to='+ TO XXXX ',

# from\_= '+TWILIO NUMBER XXX',

# body=my\_body)

# except:

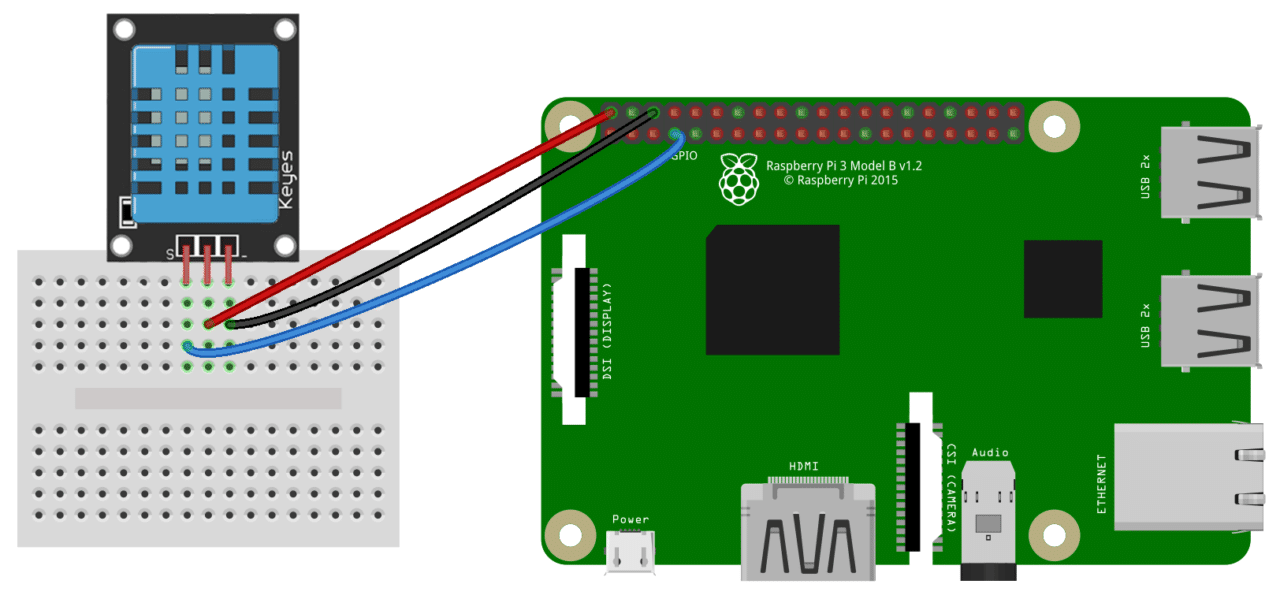
# print('Cannot send Sms!')

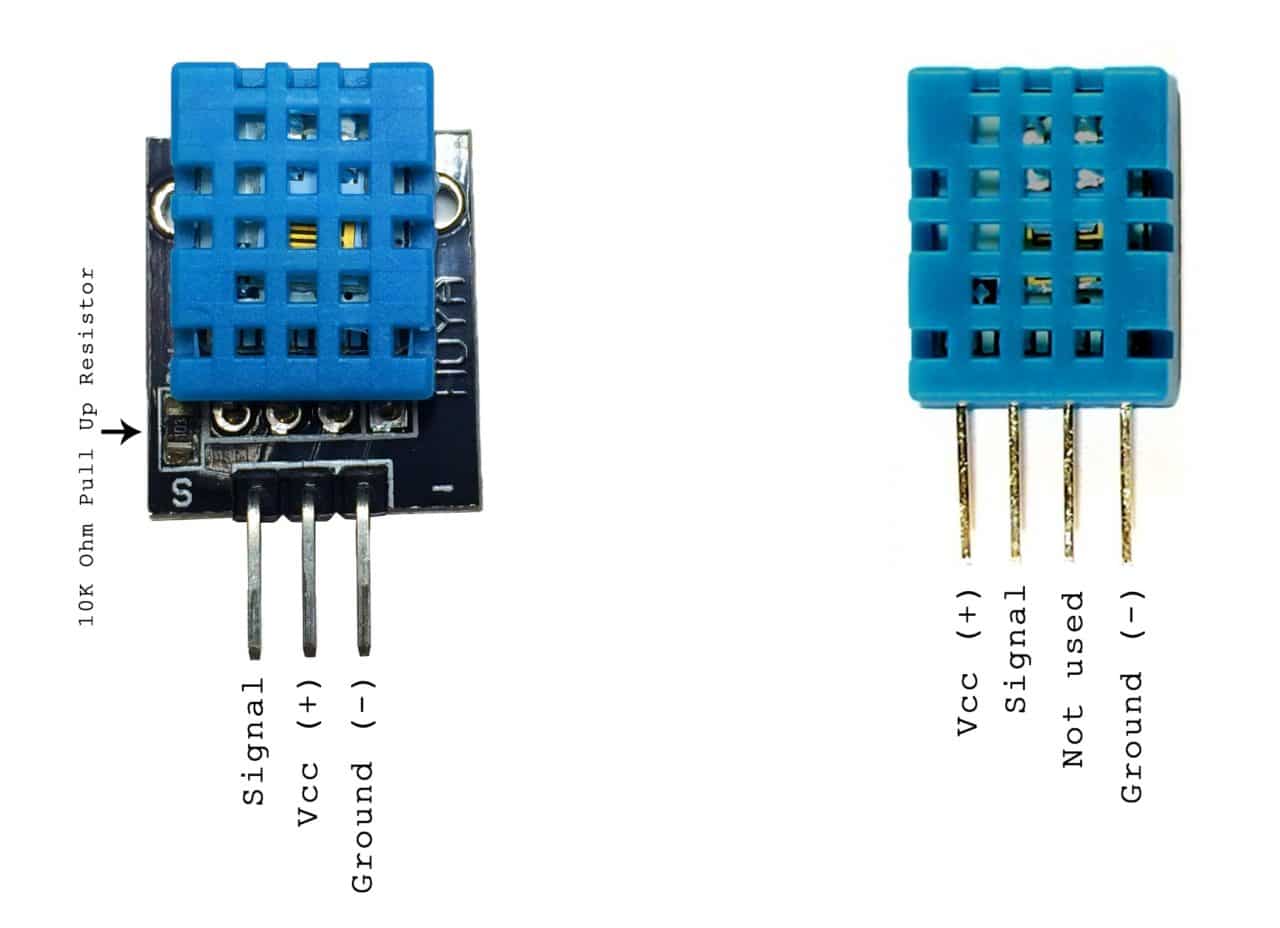
# send\_sms\_alert()

If you have Error make sure you have entered correct SSID and AUTH key. After that Make sure you have entered correct Phone Number Make sure to senter your number in to as we are suing Trial Account wont be able to send sms to other people as its a premuim Features.

## **Congrulations You just sent a SMS using Python**[¶](http://localhost:8888/notebooks/Desktop/Important%20Documents%20/IoT%20final/Combined/Lab%204/Lab%204A%20(Twilio).ipynb#Congrulations-You-just-sent-a-SMS-using-Python)

# Practical Example





# Please follow complete Steps on how to install the Library on following link

Installation Steps: -

Python library to read the DHT series of humidity and temperature sensors on a Raspberry Pi or Beaglebone Black. Designed specifically to work with the Adafruit DHT series sensors ----> <https://www.adafruit.com/products/385> Currently the library is tested with Python 2.6, 2.7, 3.3 and 3.4. It should work with Python greater than 3.4, too Dependencies

For all platforms (Raspberry Pi and Beaglebone Black) make sure your system is able to compile and download Python extensions with pip:

On Raspbian or Beaglebone Black's Debian/Ubuntu image you can ensure your system is ready by running one or two of the following sets of commands

Python 2: sudo apt-get update sudo apt-get install python-pip sudo python -m pip install --upgrade pip setuptools wheel

Python 3: sudo apt-get update sudo apt-get install python3-pip sudo python3 -m pip install --upgrade pip setuptools wheel

Install with pip Use pip to install from PyPI.

Python 2: sudo pip install Adafruit\_DHT

Python 3: sudo pip3 install Adafruit\_DHT

Compile and install from the repository First download the library source code from the GitHub releases page, unzipping the archive, and execute: Python 2: cd Adafruit\_Python\_DHT sudo python setup.py install

Python 3: cd Adafruit\_Python\_DHT sudo python3 setup.py install

You may also git clone the repository if you want to test an unreleased version: git clone <https://github.com/adafruit/Adafruit_Python_DHT.git>

import Adafruit\_DHT

from twilio.rest import Client

pin = 4

sensor = Adafruit\_DHT.DHT22

def main():

humidity, temperature = Adafruit\_DHT.read\_retry(sensor, pin)

if humidity is not None and temperature is not None:

print('Temp={0:0.1f}\*C Humidity={1:0.1f}%'.format(temperature, humidity))

if humidity >34:

print("Alert ")

send\_sms\_alert(' High Temperature Alert Warning ')

else:

print('Failed to get reading. Try again!')

def send\_sms\_alert(text='Python'):

try:

# Define your body

my\_body = text

# define client

client = Client('SSID GOES HERE XXX ','AUTH TOKEN XX ')

client.messages.create(to='+ TO XXXX ',

from\_= '+TWILIO NUMBER XXX',

body=my\_body)

except:

print('Cannot send Sms!')

if \_\_name\_\_ == "\_\_main\_\_":

while True:

main()

Application

Here is an example where I used Twilio

<https://www.youtube.com/watch?v=hT4zp-mGFkE>

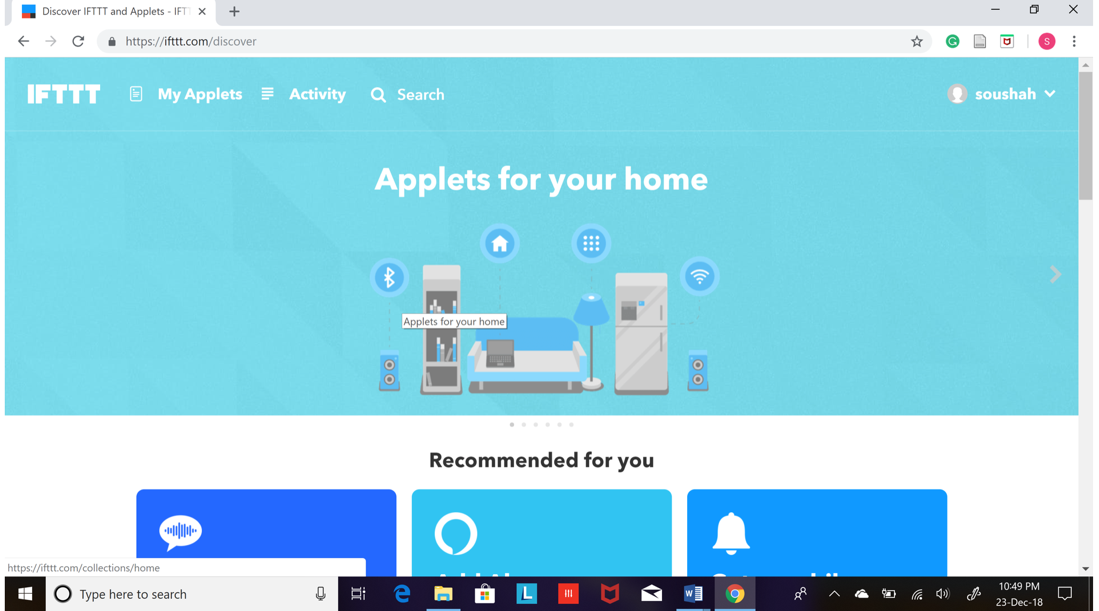
<https://www.youtube.com/watch?v=lrSUts2hNlw>

Getting started with IFTTT Lab 4B

# Objective:

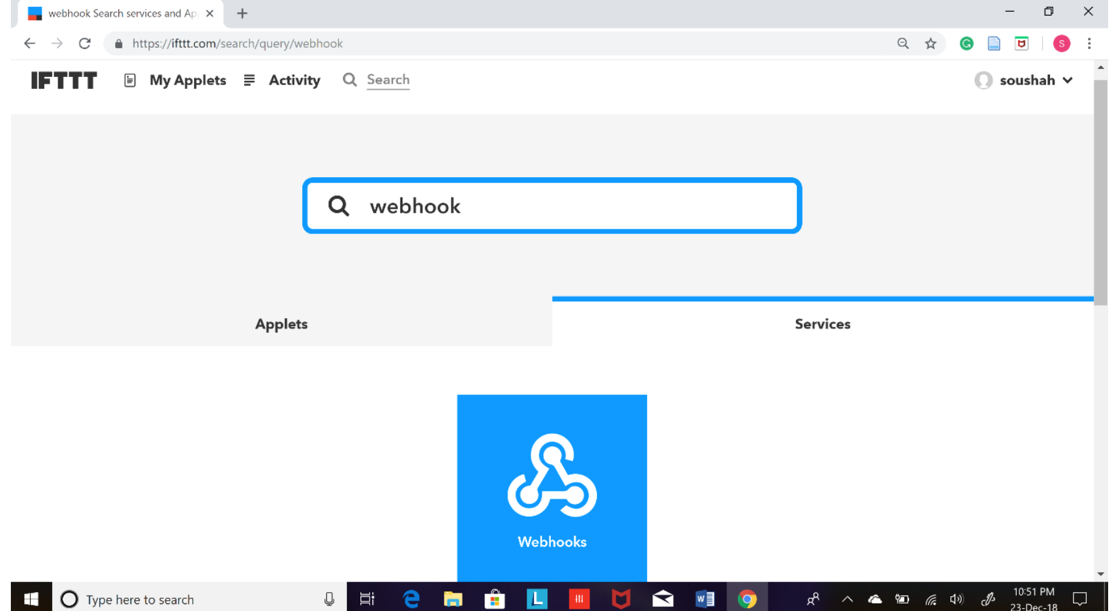
our goal is to learn the Most popular Cloud services that is IFTTT. IFTTT stands for if this than that. it provides lot of services from Phone, SMS, Gmail , Drive and much more possibilities are endless. it also allows you to connect your IoT device get and notidfied when an event occurs through webhook.

### **Step 1: Create a Account on IFTTT**

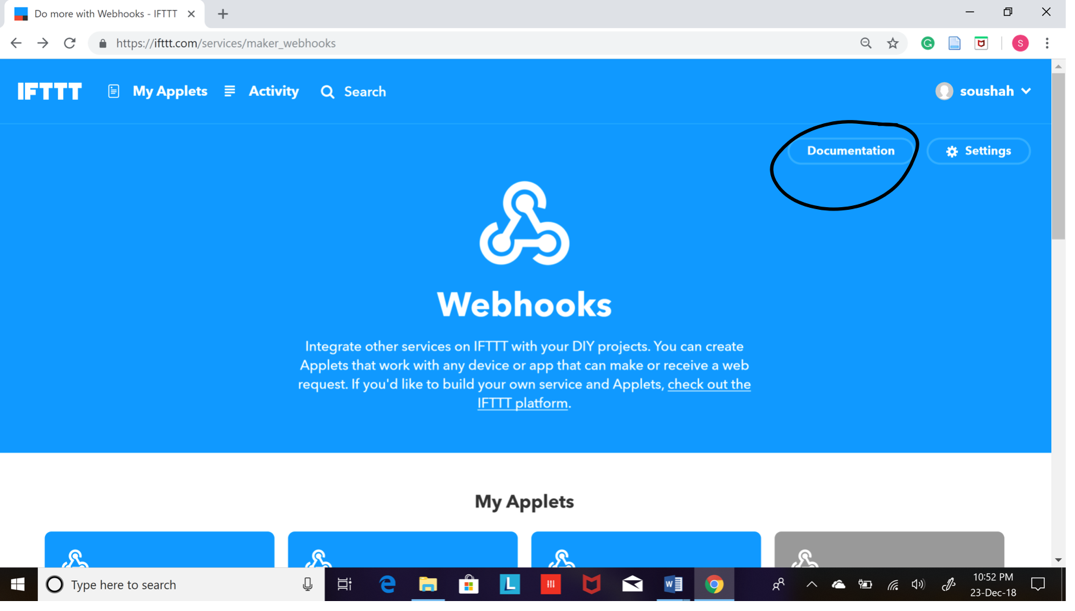


once you create a accout you should see a screen something like this

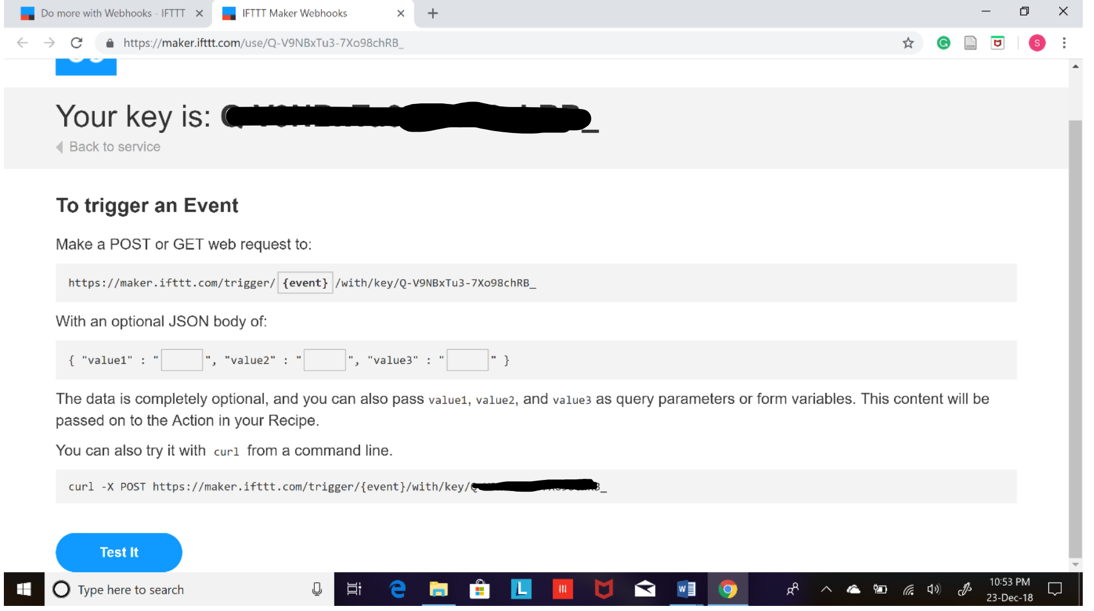
### **Step 2: Search for WebHook on Search Bar**



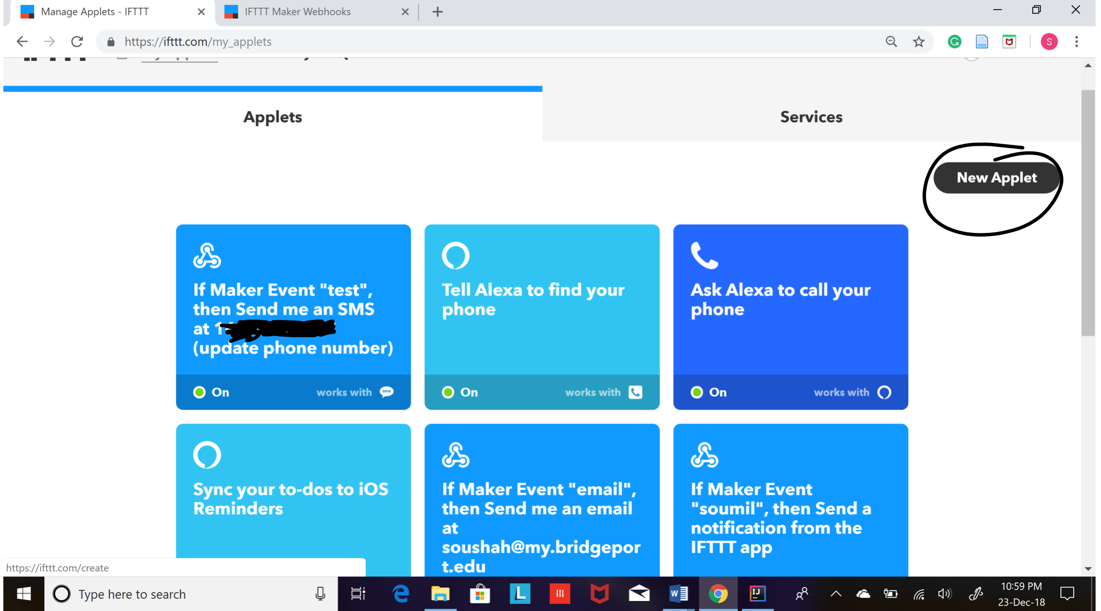
### Step 3: Click on Documentation

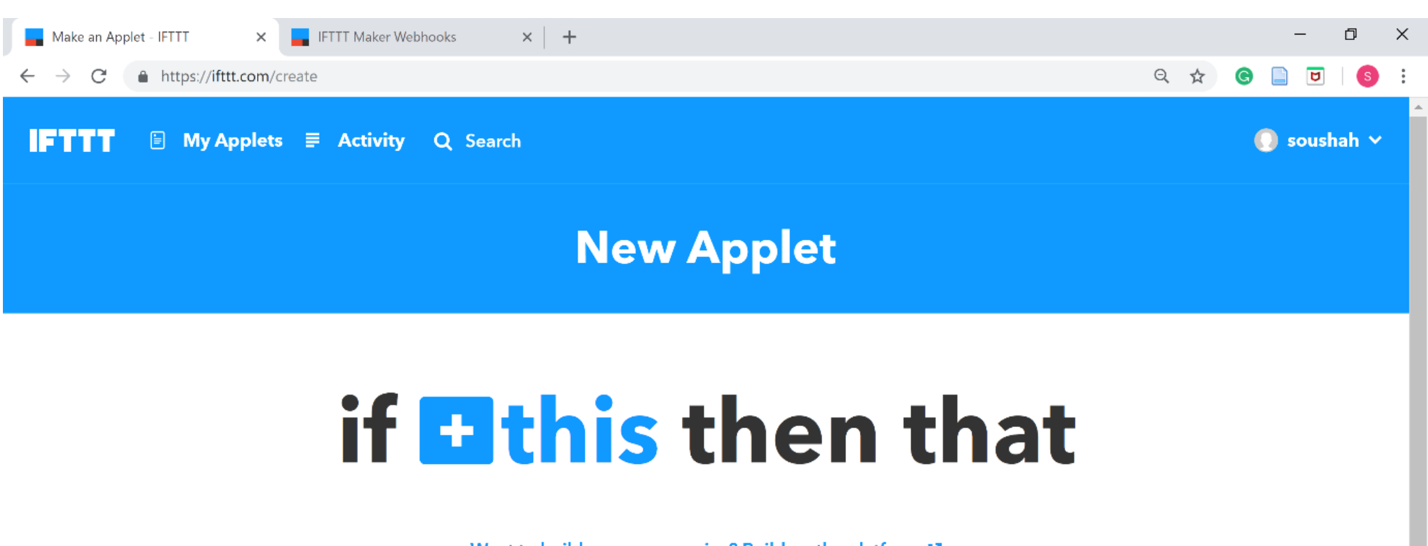


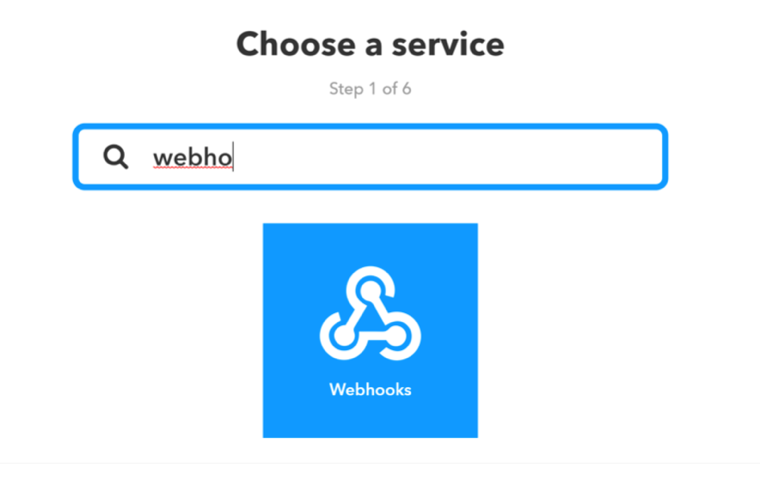
### Step 4: Copy the Secret Key



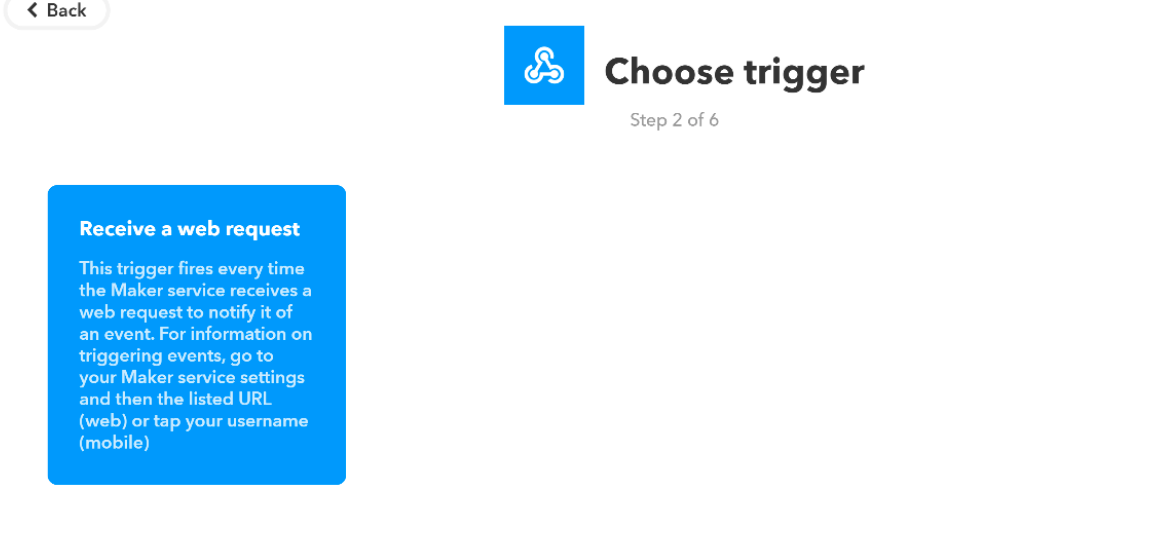
### Step 5: create a New Applete

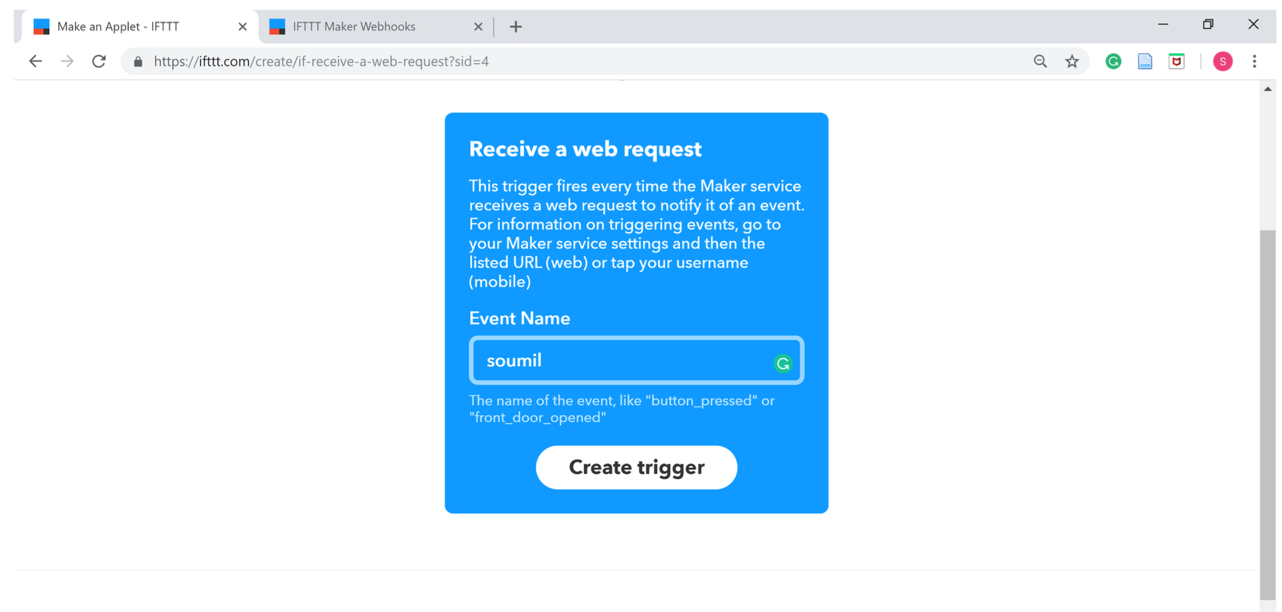






Choose receive web requests option form menu





Create Trigger

Note:- Use the IoT Master Library which was discussed above

Code:

#iftt\_k = 'Q-XXXXXX-7Xo98chRB\_'  
#feed = 'sensor'  
  
#ob1 = IfTTT(eventname=feed, key=iftt\_k)  
#ob1.iftt\_post(data1=temperature,data2=humidity)