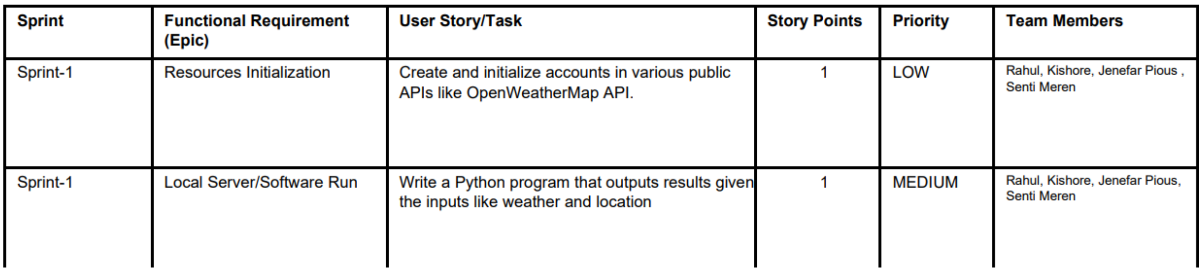
SPRINT 1



Sprint	Functional Requirement (Epic)	User Story/Task	Story Points	Priority	Team Members
Sprint-1	Resources Initialization	Create and initialize accounts in various public APIs like OpenWeatherMap API.	1	LOW	Rahul, Kishore, Jenefar Pious , Senti Meren
Sprint-1	Local Server/Software Run	Write a Python program that outputs results given the inputs like weather and location	1	MEDIUM	Rahul, Kishore, Jenefar Pious, Senti Meren

details.py

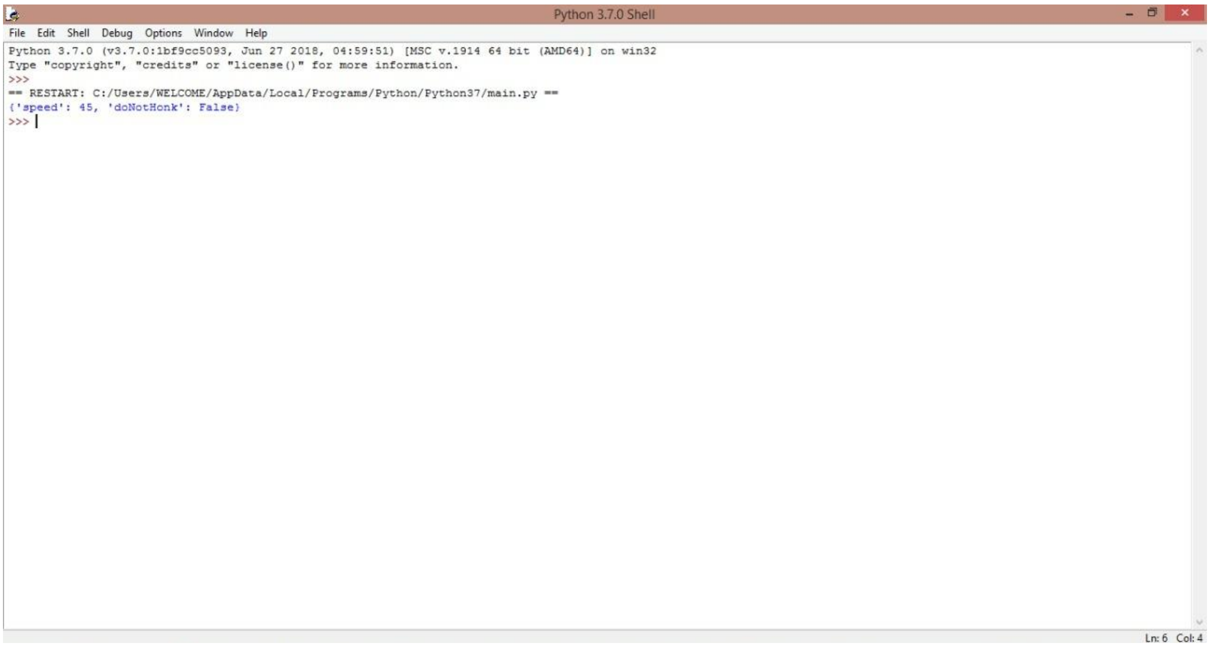
```
import weatherAPI
from datetime import datetime as dt
def processConditions(myLocation,APIKEY,localityInfo):
    weatherData = weatherAPI.get(myLocation,APIKEY)


    finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in weatherData else localityInfo["usualSpeedLimit"]/2
    finalSpeed = finalSpeed if weatherData["visibility"]>35 else finalSpeed/2


    if(localityInfo["hospitalsNearby"]):
        # hospital zone
        doNotHonk = True
    else:
        if(localityInfo["schools"]["schoolZone"]==False):
            doNotHonk = False
        else:

            now = [dt.now().hour,dt.now().minute]
            activeTime = [list(map(int,_.split(":"))) for _ in localityInfo["schools"]["activeTime"]]
doNotHonk = activeTime[0][0]<=now[0]<=activeTime[1][0] and
activeTime[0][1]<=now[1]<=activeTime[1][1]

    return({
        "speed" : finalSpeed,
        "doNotHonk" : doNotHonk
    })
```

main.py

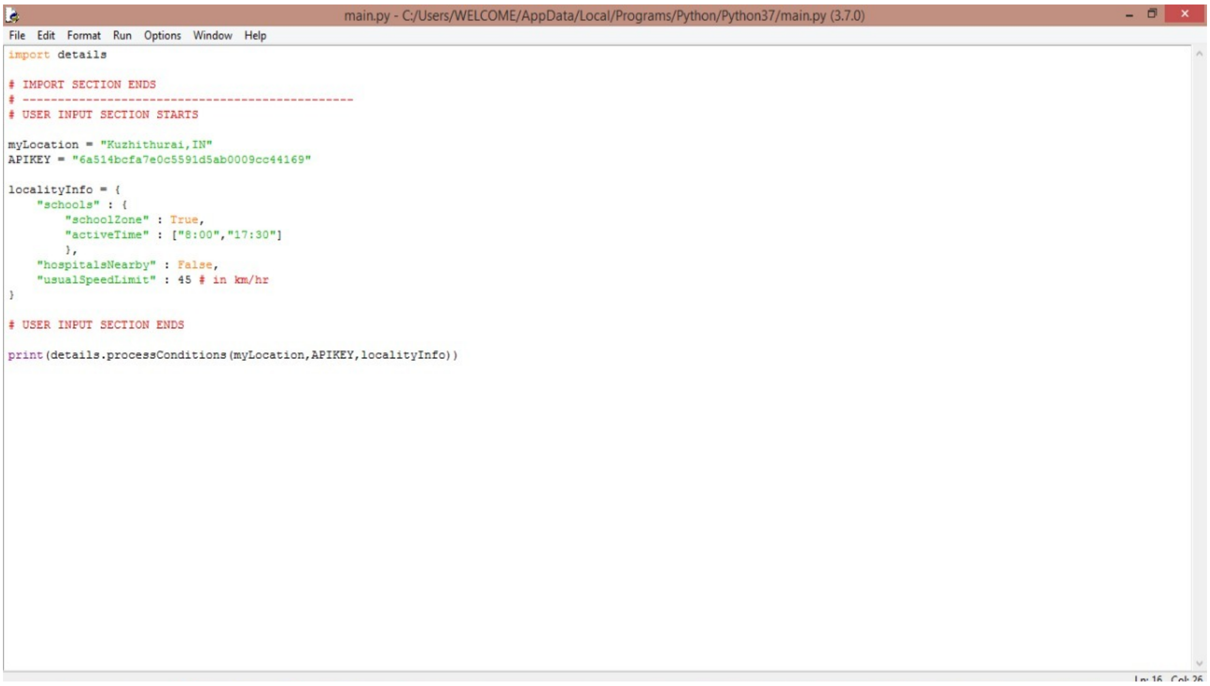
```
import details
myLocation = "Kuzhithurai"
APIKEY = "9cd610e5fd400c74212074c7ace0d62c"

localityInfo = {
    "schools" : {
        "schoolZone" : True,
        "activeTime" : ["8:00","17:30"]
        },
    "hospitalsNearby" : False,
    "usualSpeedLimit" : 45 # in km/hr
}



print(details.processConditions(myLocation,APIKEY,localityInfo))
```

weatherAPI.py

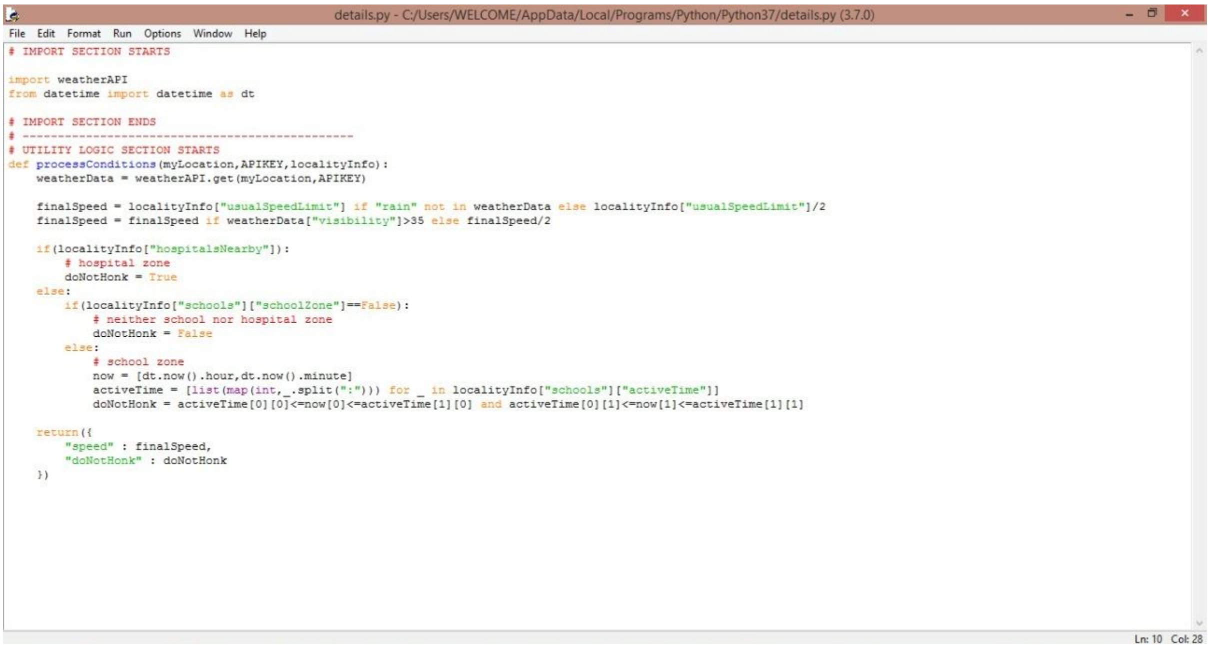
```
import requests as reqs

def get(myLocation,APIKEY):

   apiURL = f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&appid={APIKEY}"

   responseJSON = (reqs.get(apiURL)).json()

   returnObject = {

      "temperature" : responseJSON['main']['temp'] - 273.15,

      "weather" : [responseJSON['weather'][_]['main'].lower() for _ in range(len(responseJSON['weather']))],

      "visibility" : responseJSON['visibility']/100,

   }

   if("rain" in responseJSON):

      returnObject["rain"] = [responseJSON["rain"][key] for key in responseJSON["rain"]]

   return(returnObject)
```



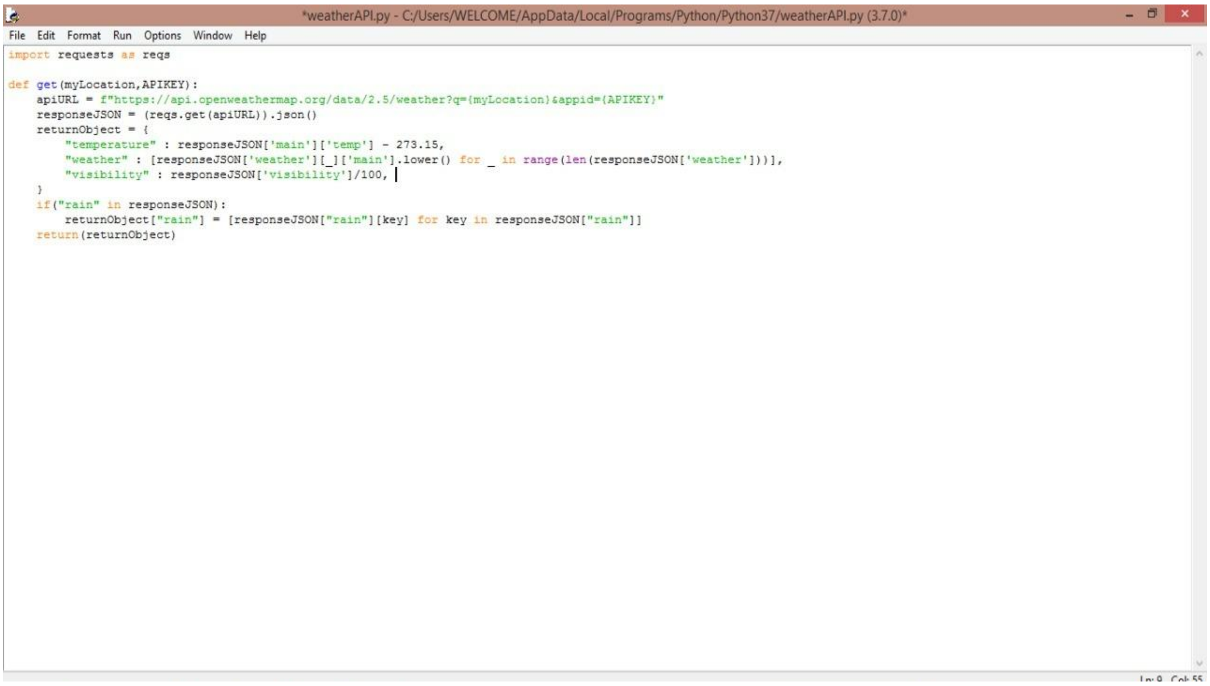
```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/WELCOME/AppData/Local/Programs/Python/Python37/main.py ==
{'speed': 45, 'doNotHonk': False}
>>> 
```


main.py - C:/Users/WELCOME/AppData/Local/Programs/Python/Python37/main.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 16 Col: 26

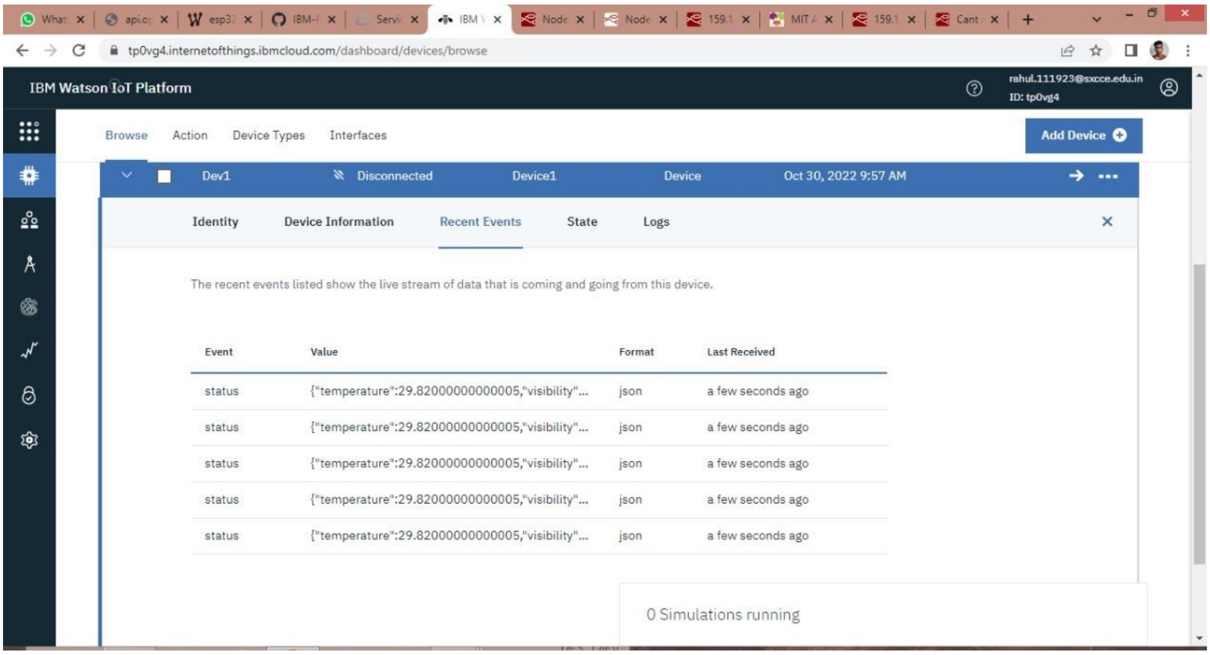
```python
import details

# IMPORT SECTION ENDS
# ----------------------------------------------
# USER INPUT SECTION STARTS

myLocation = "Kuzhithurai,IN"
APIKEY = "6a514bcfa7e0c5591d5ab0009cc44169"

localityInfo = {
    "schools" : {
        "schoolZone" : True,
        "activeTime" : ["8:00","17:30"]
        },
    "hospitalsNearby" : False,
    "usualSpeedLimit" : 45 # in km/hr
}

# USER INPUT SECTION ENDS

print(details.processConditions(myLocation,APIKEY,localityInfo))
```


details.py - C:/Users/WELCOME/AppData/Local/Programs/Python/Python37/details.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 10 Col: 28

```python
# IMPORT SECTION STARTS

import weatherAPI
from datetime import datetime as dt

# IMPORT SECTION ENDS
# ----------------------------------------------
# UTILITY LOGIC SECTION STARTS
def processConditions(myLocation,APIKEY,localityInfo):
    weatherData = weatherAPI.get(myLocation,APIKEY)

    finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in weatherData else localityInfo["usualSpeedLimit"]/2
    finalSpeed = finalSpeed if weatherData["visibility"]>35 else finalSpeed/2

    if(localityInfo["hospitalsNearby"]):
        # hospital zone
        doNotHonk = True
    else:
        if(localityInfo["schools"]["schoolZone"]==False):
            # neither school nor hospital zone
            doNotHonk = False
        else:
            # school zone
            now = [dt.now().hour,dt.now().minute]
            activeTime = [list(map(int,_.split(":"))) for _ in localityInfo["schools"]["activeTime"]]
            doNotHonk = activeTime[0][0]<=now[0]<=activeTime[1][0] and activeTime[0][1]<=now[1]<=activeTime[1][1]

    return({
        "speed" : finalSpeed,
        "doNotHonk" : doNotHonk
    })
```

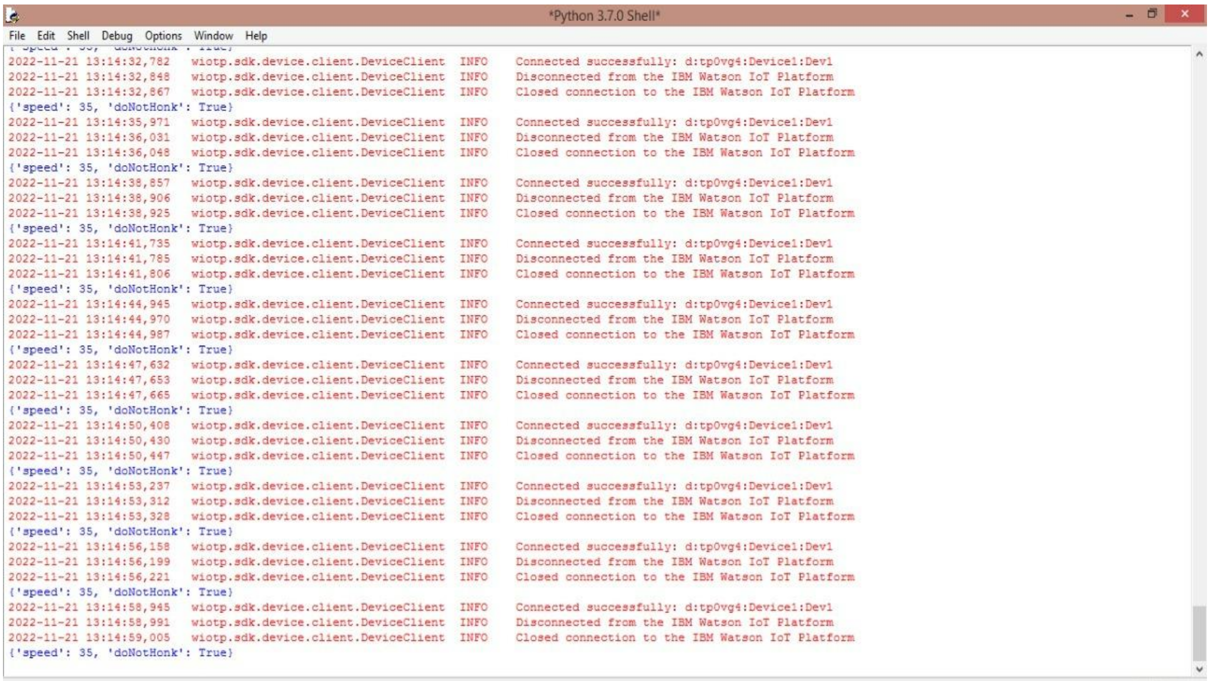

weatherAPI.py - C:/Users/WELCOME/AppData/Local/Programs/Python/Python37/weatherAPI.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 9 Col: 55

```
import requests as reqs

def get(myLocation,APIKEY):
    apiURL = f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&appid={APIKEY}"
    responseJSON = (reqs.get(apiURL)).json()
    returnObject = {
        "temperature" : responseJSON['main']['temp'] - 273.15,
        "weather" : [responseJSON['weather'][_]['main'].lower() for _ in range(len(responseJSON['weather']))],
        "visibility" : responseJSON['visibility']/100,
    }
    if("rain" in responseJSON):
        returnObject["rain"] = [responseJSON["rain"][key] for key in responseJSON["rain"]]
    return(returnObject)
```

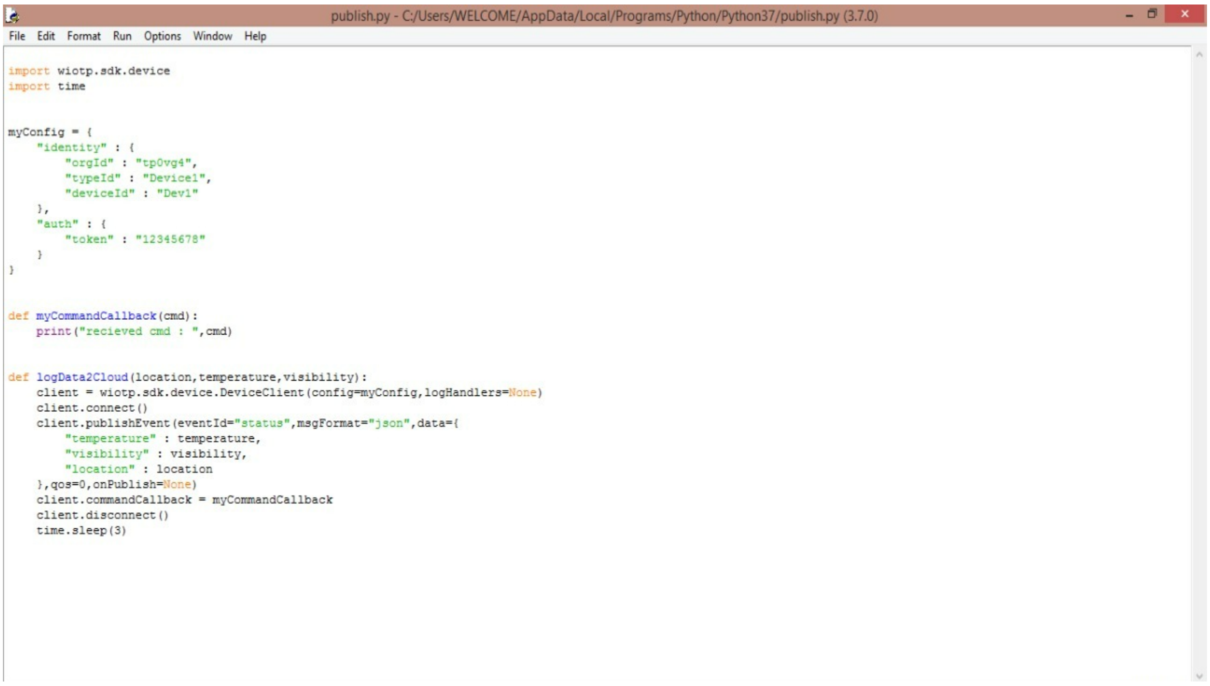


The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
status	{"temperature":29.82000000000005,"visibility"...	json	a few seconds ago
status	{"temperature":29.82000000000005,"visibility"...	json	a few seconds ago
status	{"temperature":29.82000000000005,"visibility"...	json	a few seconds ago
status	{"temperature":29.82000000000005,"visibility"...	json	a few seconds ago
status	{"temperature":29.82000000000005,"visibility"...	json	a few seconds ago



```
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:32,782   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:32,848   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:32,867   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:35,971   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:36,031   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:36,048   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:38,857   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:38,906   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:38,925   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:41,735   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:41,785   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:41,806   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:44,945   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:44,970   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:44,987   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:47,632   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:47,653   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:47,665   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:50,408   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:50,430   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:50,447   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:53,237   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:53,312   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:53,328   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:56,158   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:56,199   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:56,221   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
2022-11-21 13:14:58,945   wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:tp0vg4:Device1:Dev1
2022-11-21 13:14:58,991   wiotp.sdk.device.client.DeviceClient  INFO    Disconnected from the IBM Watson IoT Platform
2022-11-21 13:14:59,005   wiotp.sdk.device.client.DeviceClient  INFO    Closed connection to the IBM Watson IoT Platform
{'speed': 35, 'doNotHonk': True}
```


publish.py - C:/Users/WELCOME/AppData/Local/Programs/Python/Python37/publish.py (3.7.0)
File Edit Format Run Options Window Help

```python
import wiotp.sdk.device
import time


myConfig = {
    "identity" : {
        "orgId" : "tp0vg4",
        "typeId" : "Device1",
        "deviceId" : "Dev1"
    },
    "auth" : {
        "token" : "12345678"
    }
}


def myCommandCallback(cmd):
    print("recieved cmd : ",cmd)


def logData2Cloud(location,temperature,visibility):
    client = wiotp.sdk.device.DeviceClient(config=myConfig,logHandlers=None)
    client.connect()
    client.publishEvent(eventId="status",msgFormat="json",data={
        "temperature" : temperature,
        "visibility" : visibility,
        "location" : location
    },qos=0,onPublish=None)
    client.commandCallback = myCommandCallback
    client.disconnect()
    time.sleep(3)
```