PROJECT DEVELOPMENT PHASE

SPRINT-1

Team ID	PNT2022TMID48285
Project Name	Industrial Specific Fire Management System

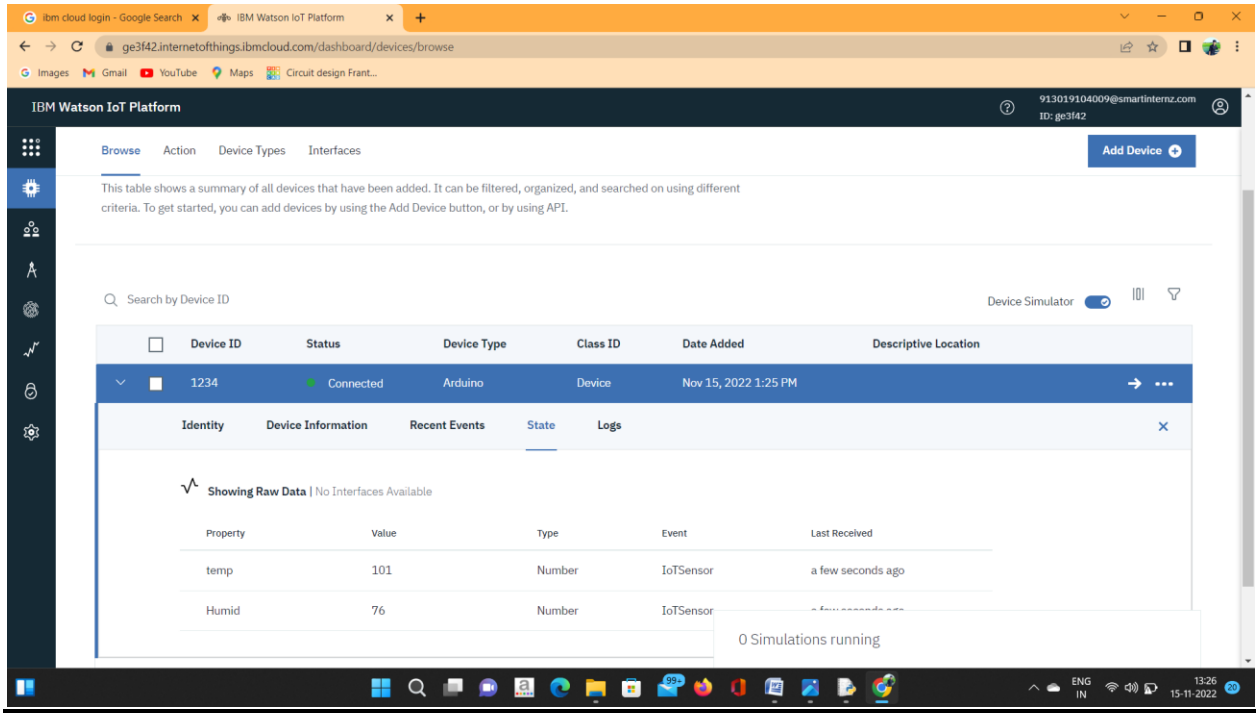
PYTHON CODE:

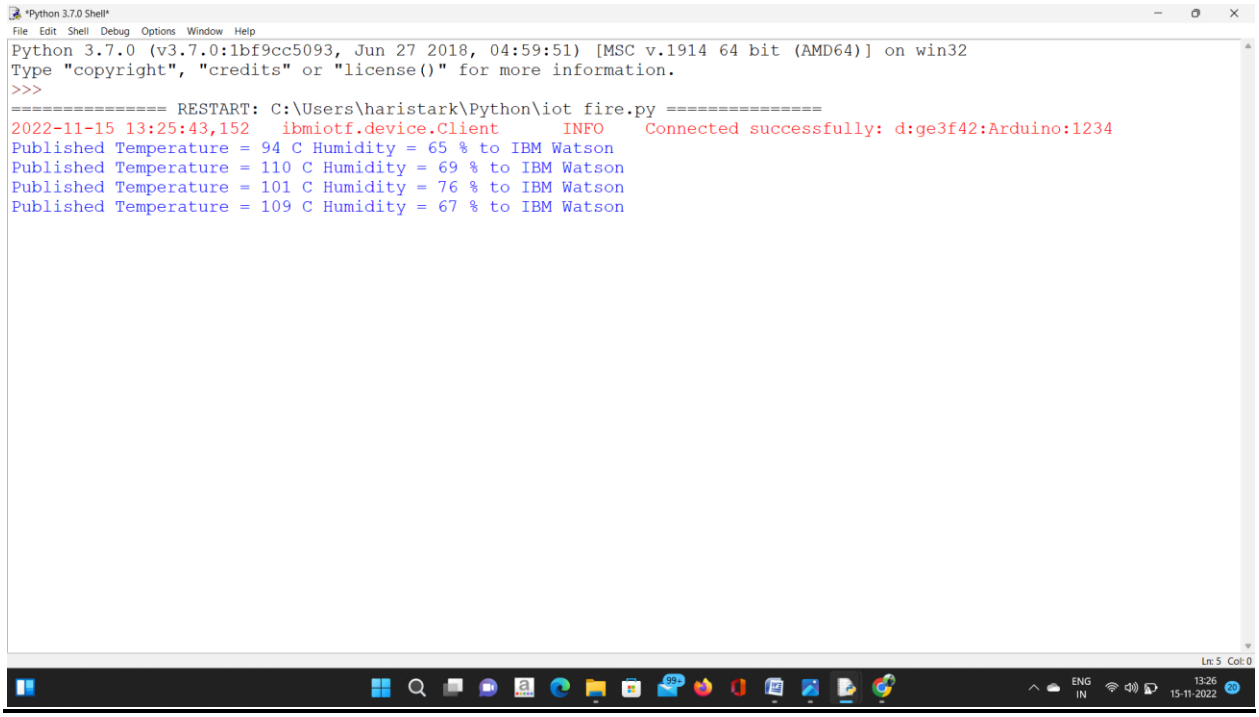
```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device Credentials
organization = "ge3f42"
deviceType = "Arduino"
deviceId = "1234"
authMethod = "token"
authToken = "FfR(Gr?Vsx?4c-*k45"

# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
```

```
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    elif status == "lightoff":
        print ("led is off")
    else :
        print ("please send proper command")



try:
        deviceOptions = {"org": organization, "type": deviceType, "id":
deviceId, "auth-method": authMethod, "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        #.....


except Exception as e:
            print("Caught exception connecting : %s" % str(e))
            sys.exit()


# Connect and send a datapoint "hello" with value "world" into the
cloud as an event of type "greeting" 10 times
deviceCli.connect()
```

```
while True:
        #Get Sensor Data from DHT11


        temp=random.randint(90,110)
        Humid=random.randint(60,100)


        data = { 'temp' : temp, 'Humid': Humid }
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp, "Humidity =
%s %%" % Humid, "to IBM Watson")


        success = deviceCli.publishEvent("IoTSensor", "json", data,
qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(10)


        deviceCli.commandCallback = myCommandCallback


# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

OUTPUT:





RESULT: The python code is executed successfully.