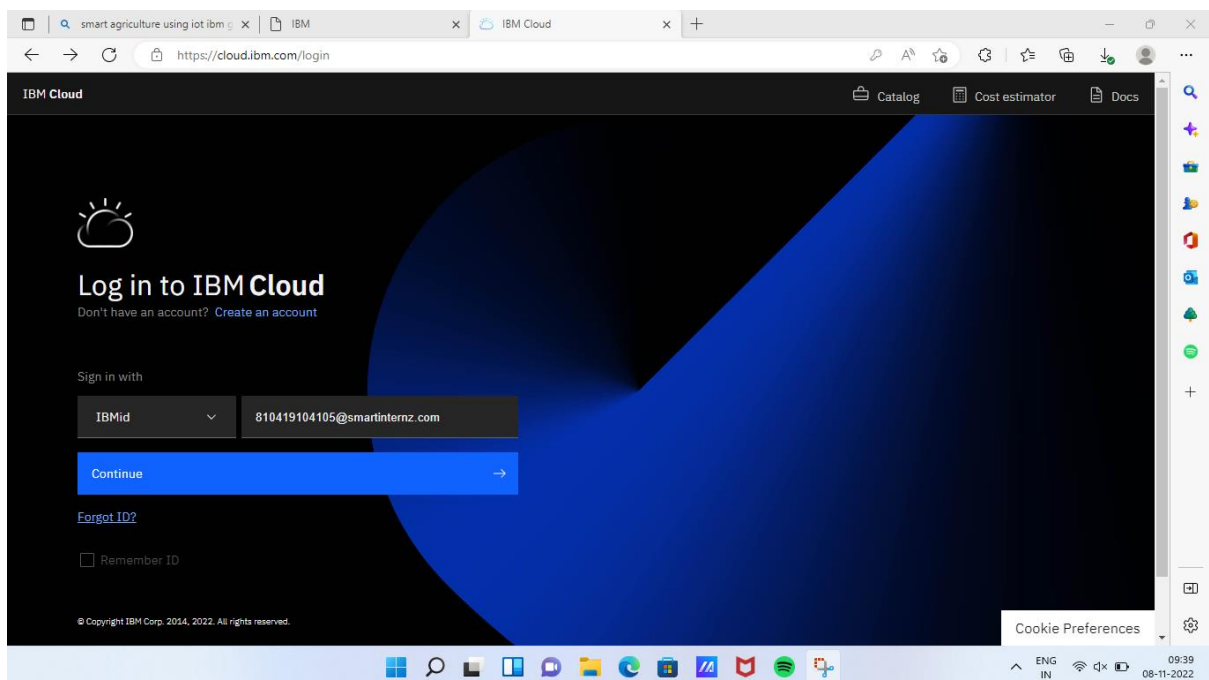


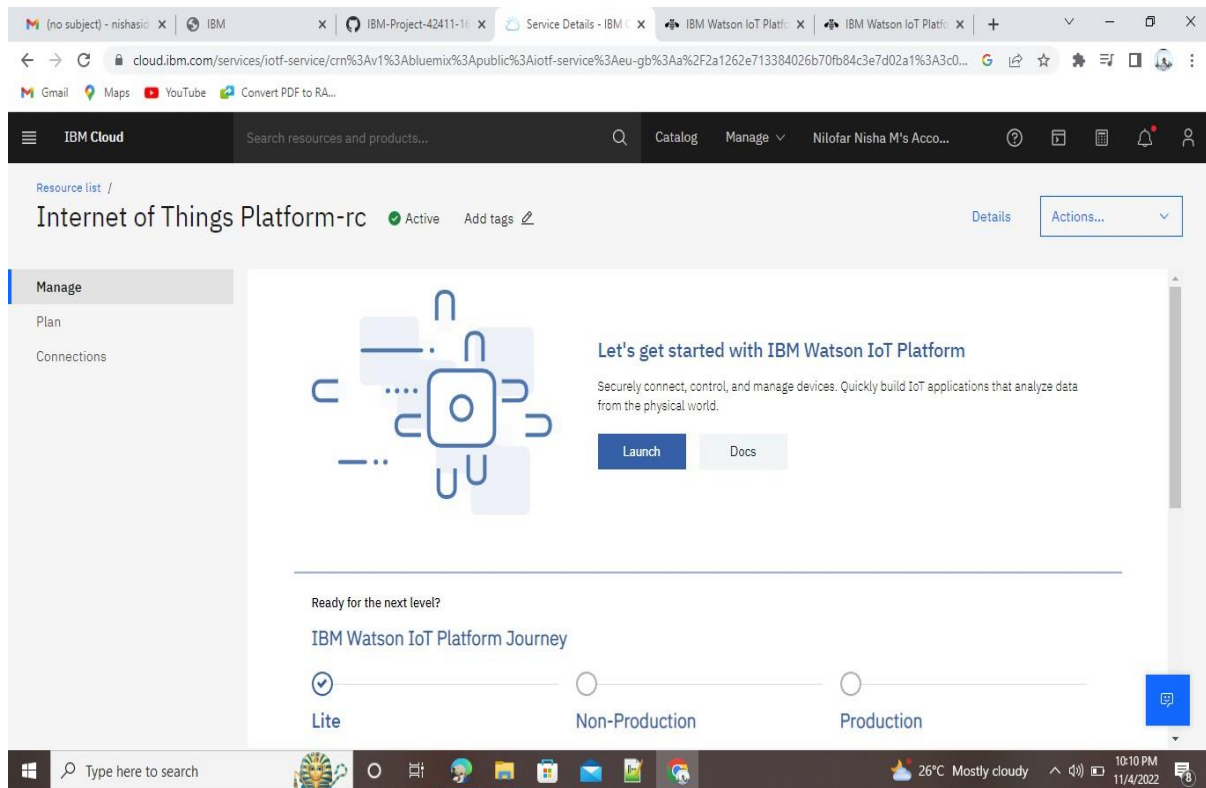
DEVELOP THE PYTHON SCRIPT

TEAM ID	PNT2022TMID08369
PROJECT NAME	IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

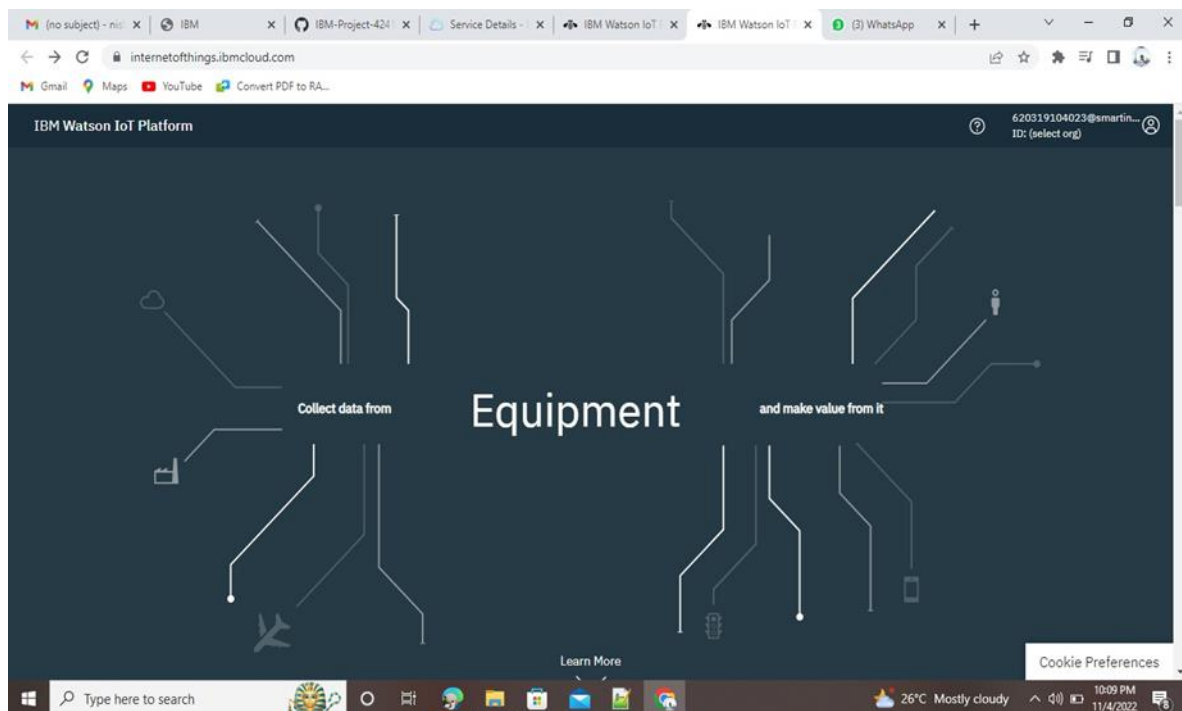
STEP 1: First login your IBM cloud account



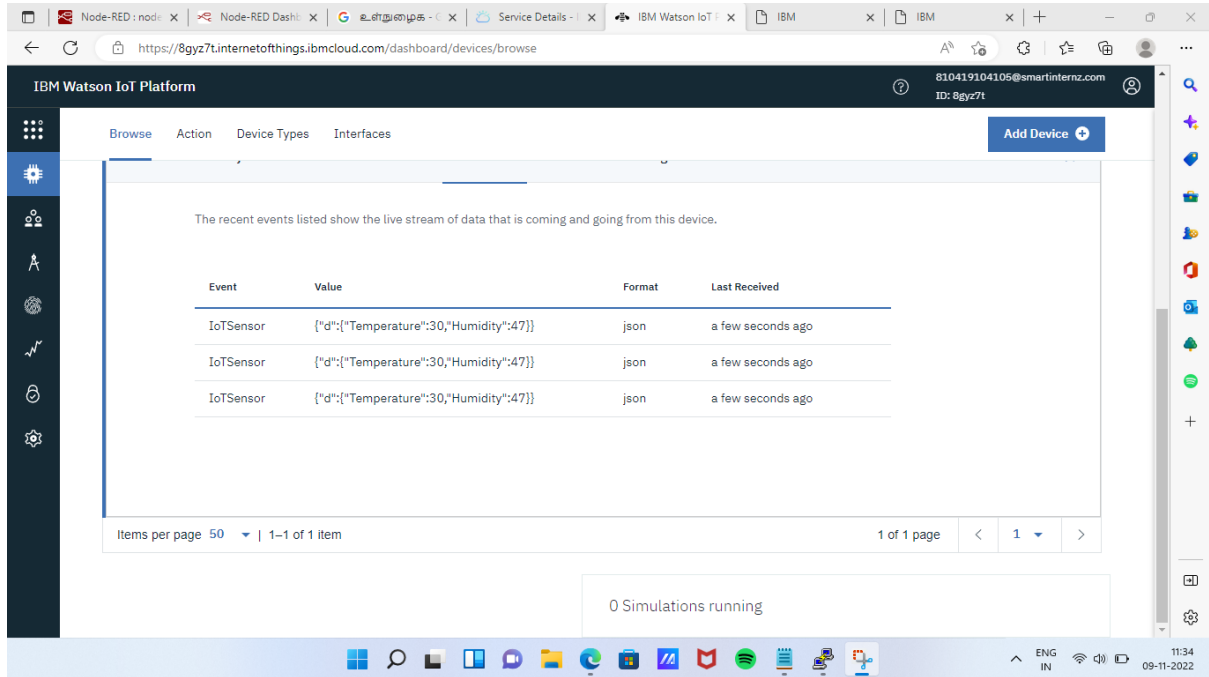
STEP 2: Internet of things platform smart crop protection will be created, where there are different options like manage, plan, and connection.



STEP 3: Clicking on the launch button in the manage tab, it will open to this.



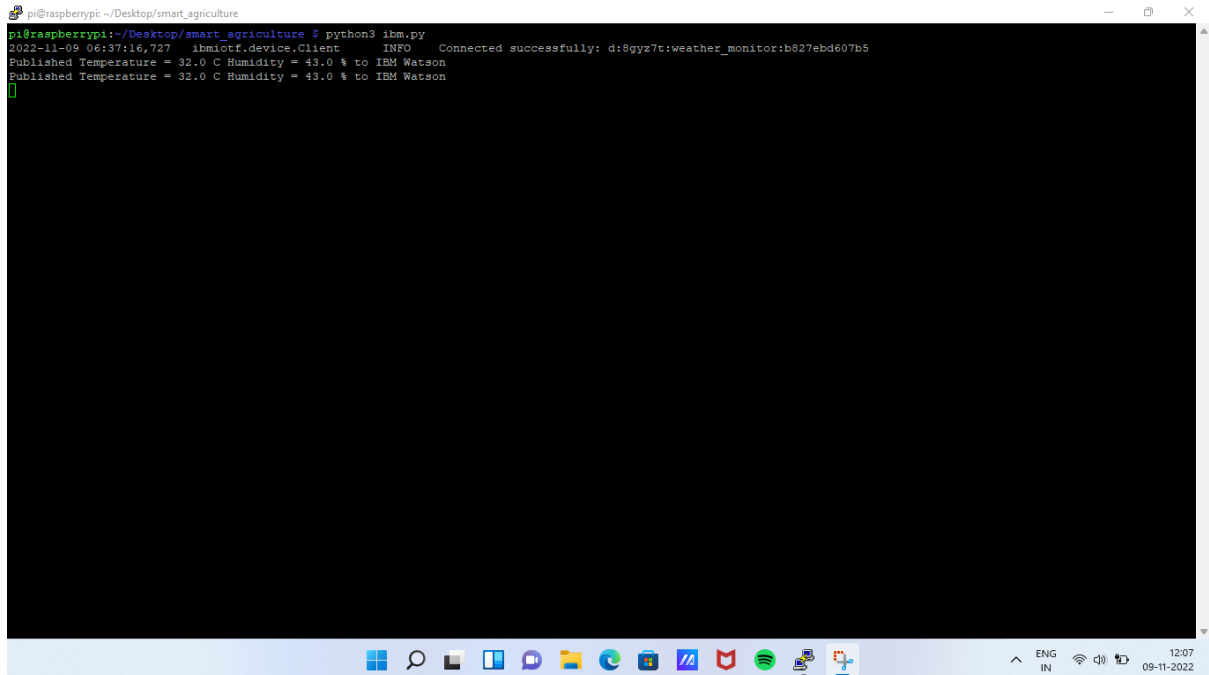
STEP 4: while running python code temperature and humidity value are published in IBM IoT Watson platform.



The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area displays a table of recent events. The table has four columns: 'Event', 'Value', 'Format', and 'Last Received'. The data shows three consecutive events from an 'IoT Sensor' with a value of '{"d":{"Temperature":30,"Humidity":47}}' in 'json' format, received 'a few seconds ago'. Below the table, it indicates 'Items per page 50' and '1-1 of 1 item'. At the bottom, a status bar shows '0 Simulations running'.

Event	Value	Format	Last Received
IoTSensor	{"d":{"Temperature":30,"Humidity":47}}	json	a few seconds ago
IoTSensor	{"d":{"Temperature":30,"Humidity":47}}	json	a few seconds ago
IoTSensor	{"d":{"Temperature":30,"Humidity":47}}	json	a few seconds ago

STEP 4: This is the python program output which is published in IBM IoT Watson platform.



The screenshot shows a terminal window on a Raspberry Pi. The prompt is 'pi@raspberrypi: ~/Desktop/smart_agriculture'. The user has run the command 'python3 ibm.py'. The output shows the script successfully connected to the IBM IoT Watson platform and published temperature and humidity data. The output text is as follows:

```
pi@raspberrypi:~/Desktop/smart_agriculture$ python3 ibm.py
2022-11-09 06:37:16,727 ibmiotf.device.Client INFO Connected successfully: d:8gyz7t:weather_monitor:b827ebd607b5
Published Temperature = 32.0 C Humidity = 43.0 % to IBM Watson
Published Temperature = 32.0 C Humidity = 43.0 % to IBM Watson
```

PYTHON CODE:

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

#Provide your IBM Watson Device Credentials
organization = "8gyz7t"
deviceType = "weather_monitor"
deviceId = "b827ebd607b5"
authMethod = "token"
authToken = "LWVpQPpVQ166HWN48f"

# Initialize GPIO and DHT11
sensor = Adafruit_DHT.DHT11
pin=4

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status = cmd.data['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 device: %s" % str(e))
```

```
    sys.exit()
```

```
deviceCli.connect()
```

```
while True:
```

```
    #Get Sensor Data from DHT11
```

```
    Humidity, Temperature = Adafruit_DHT.read_retry(sensor, pin)
```

```
    data = {"d":{"Temperature":Temperature, 'Humidity': Humidity}}
```

```
    #print data
```

```
    def myOnPublishCallback():
```

```
        print ("Published Temperature = %s C" % Temperature, "Humidity =  
%s %" % Humidity, "to IBM Watson")
```

```
        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,  
on_publish=myOnPublishCallback)
```

```
        if not success:
```

```
            print("Not connected to IoTF")
```

```
            time.sleep(1)
```

```
        deviceCli.commandCallback = myCommandCallback
```

```
# Disconnect the device and application from the cloud
```

```
deviceCli.disconnect()
```