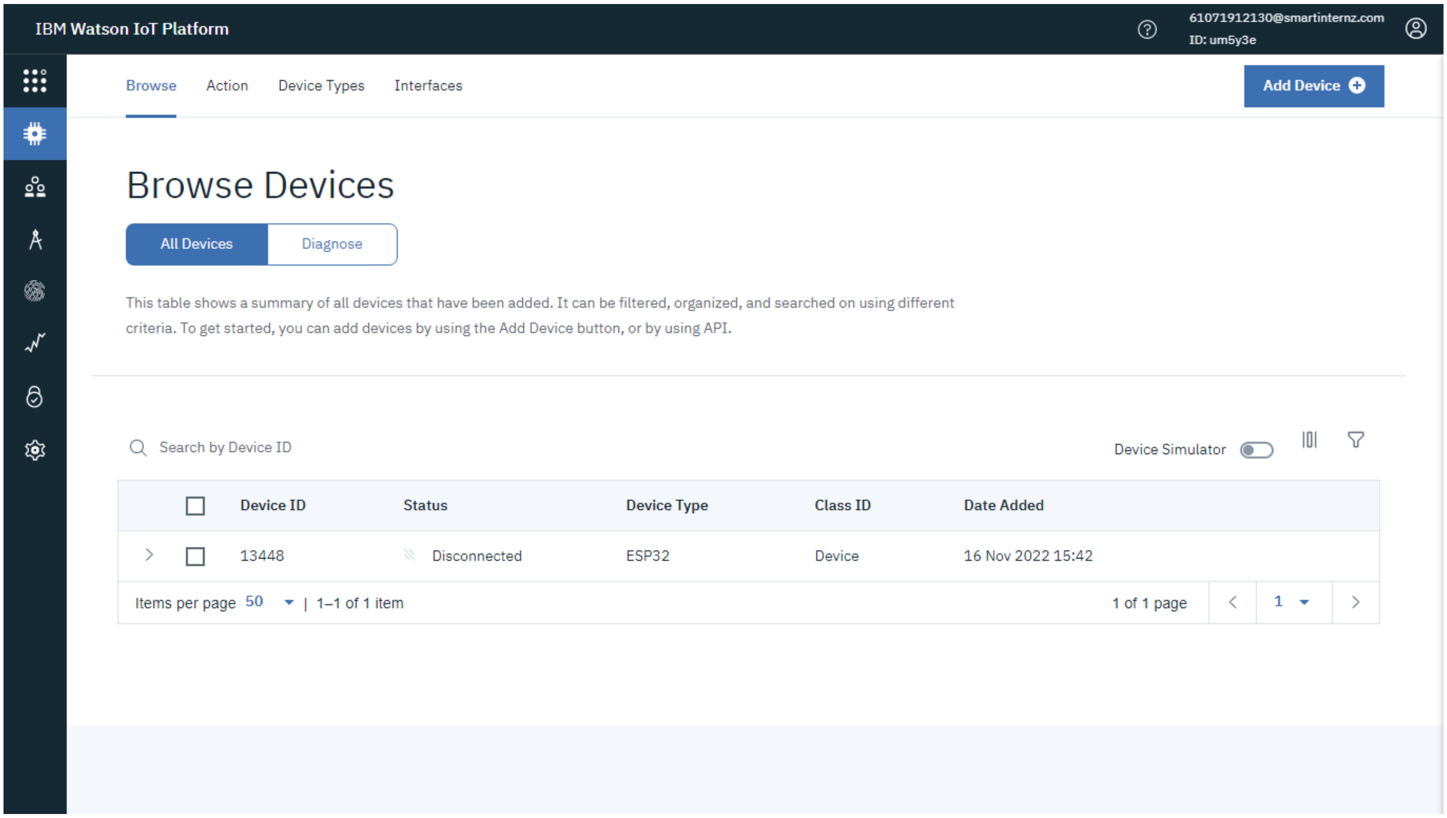
Project Development Phase

Sprint – 2

Date	06 Nov 2022
Team ID	PNT2022TMID06691
Project Name	Real-Time River Water Quality Monitoring and Control System
Maximum Marks	8 Marks

USN – 4: Create the IBM Watson IoT and device Settings

As a user, I can create the IBM Watson IoT Platform and integrate the microcontroller with it, to send the sensed data on Cloud.



Browse Devices

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

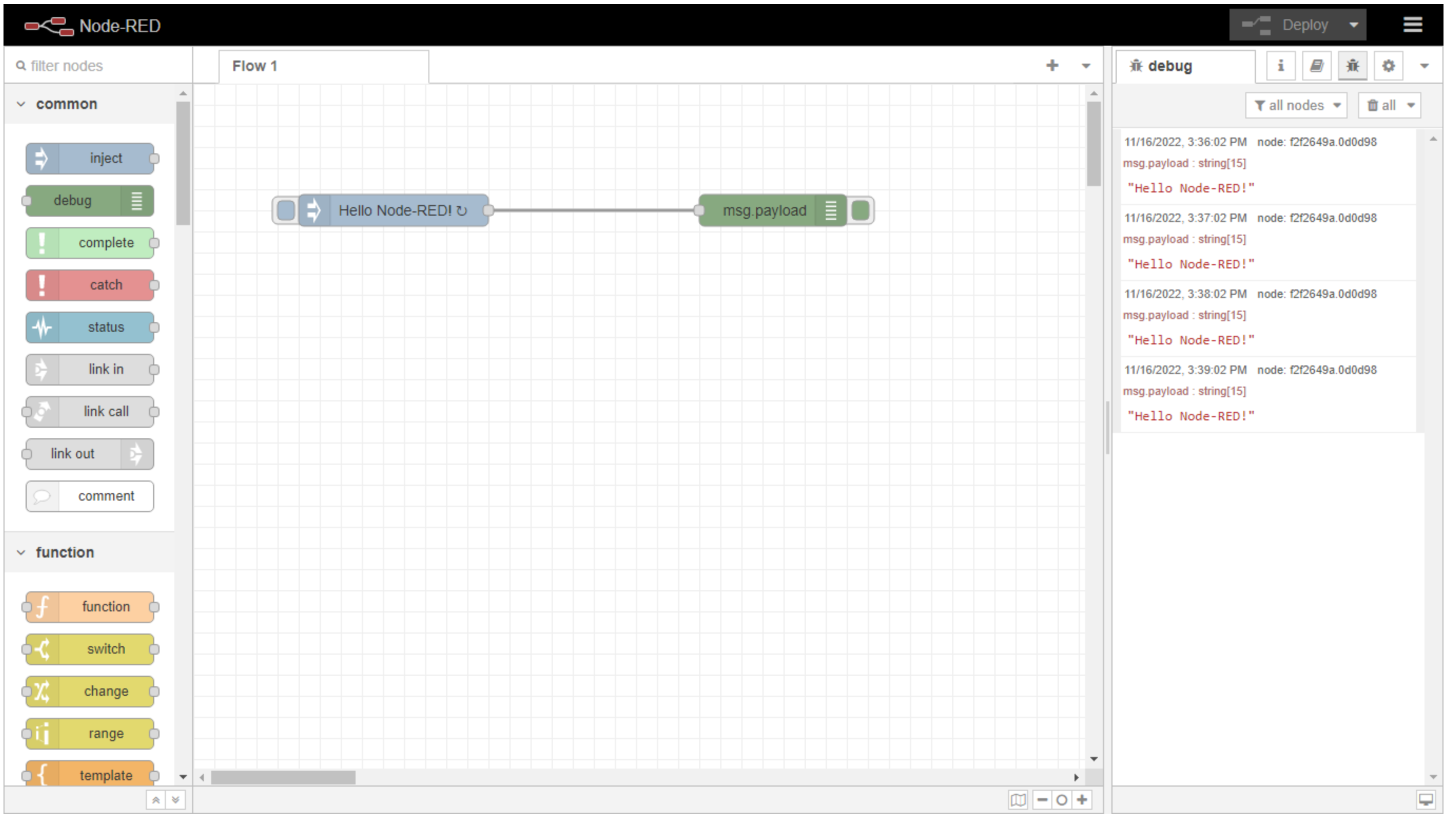
□	Device ID	Status	Device Type	Class ID	Date Added
□	13448	Disconnected	ESP32	Device	16 Nov 2022 15:42

Items per page 50 | 1–1 of 1 item

1 of 1 page 1

USN – 5: Create a node red service

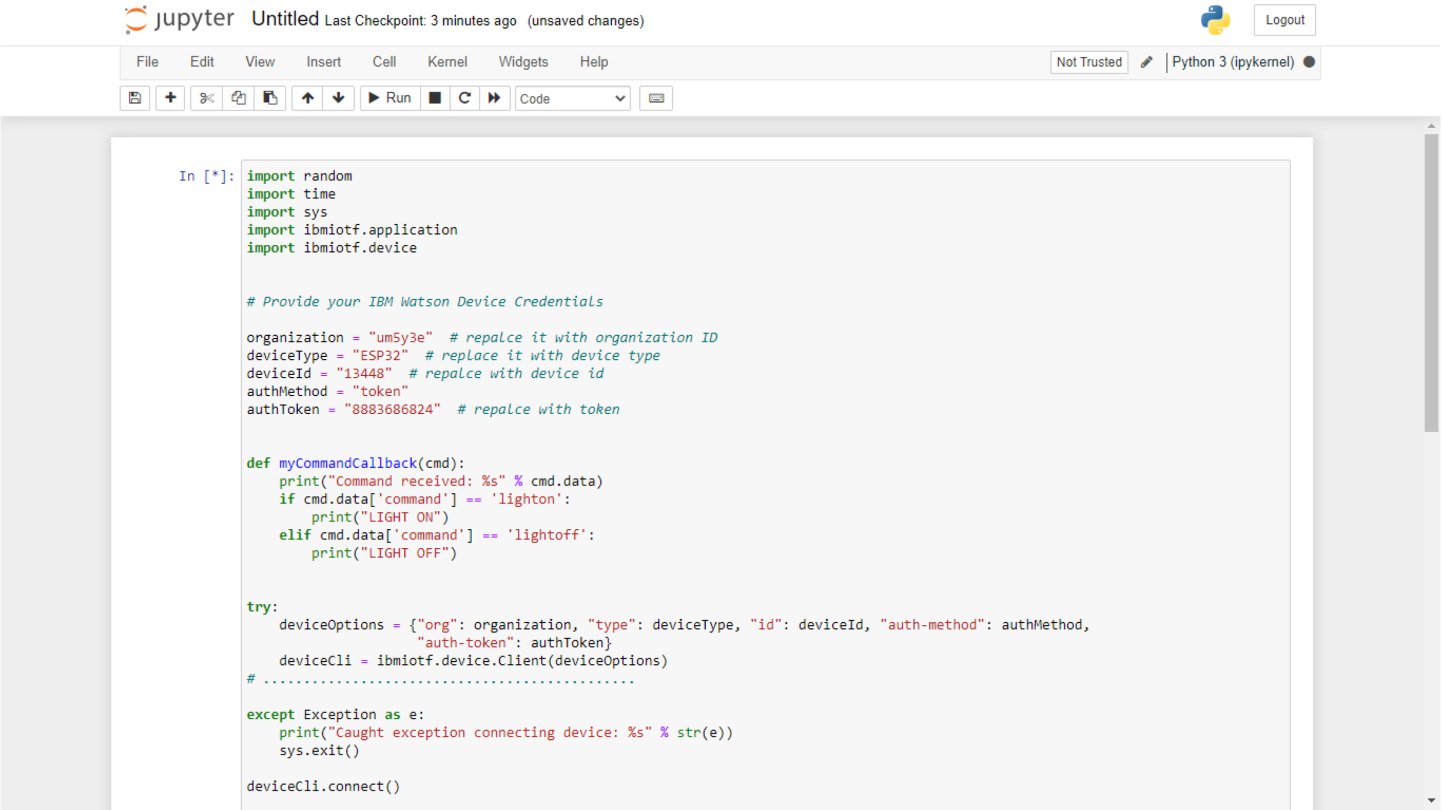
As a user, I can create a node red service to integrate the IBM Watson along with the Web UI



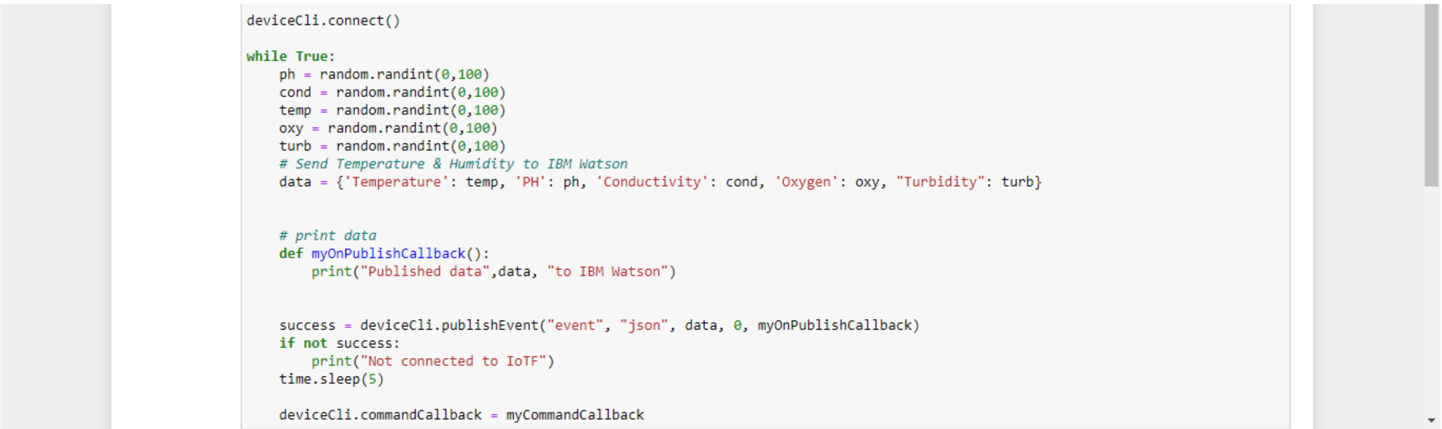
```
11/16/2022, 3:36:02 PM   node: f2f2649a.0d0d98
msg.payload : string[15]
"Hello Node-RED!"
11/16/2022, 3:37:02 PM   node: f2f2649a.0d0d98
msg.payload : string[15]
"Hello Node-RED!"
11/16/2022, 3:38:02 PM   node: f2f2649a.0d0d98
msg.payload : string[15]
"Hello Node-RED!"
11/16/2022, 3:39:02 PM   node: f2f2649a.0d0d98
msg.payload : string[15]
"Hello Node-RED!"
```

USN – 6: To develop a Python code

As a user, I can create a python code to sense the physical quantity and store data.



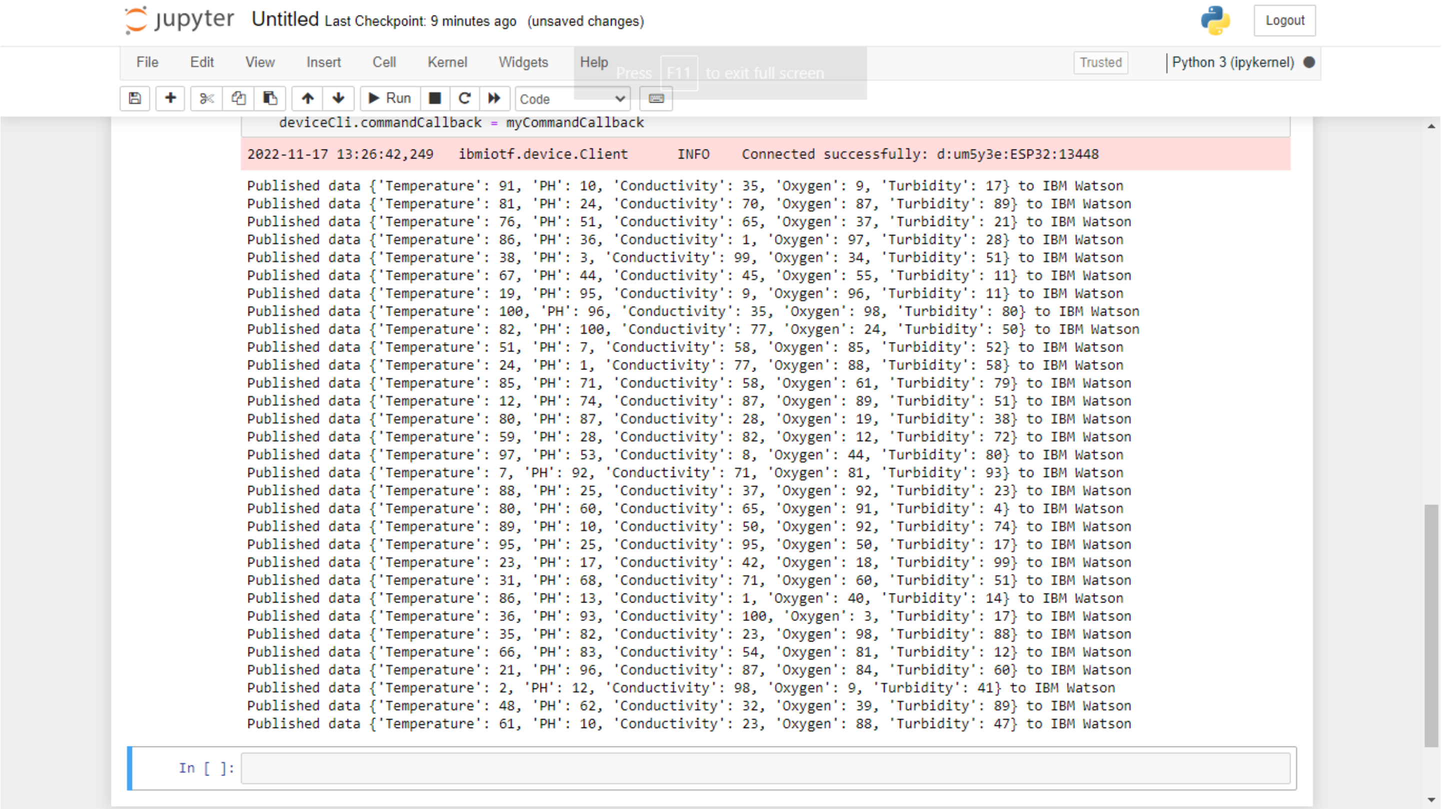
```
In [*]: import random
        import time
        import sys
        import ibmiotf.application
        import ibmiotf.device


        # Provide your IBM Watson Device Credentials

        organization = "um5y3e"  # repalce it with organization ID
        deviceType = "ESP32"  # replace it with device type
        deviceId = "13448"  # repalce with device id
        authMethod = "token"
        authToken = "8883686824"  # repalce with token


        def myCommandCallback(cmd):
            print("Command received: %s" % cmd.data)
            if cmd.data['command'] == 'lighton':
                print("LIGHT ON")
            elif cmd.data['command'] == 'lightoff':
                print("LIGHT OFF")


        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()
```



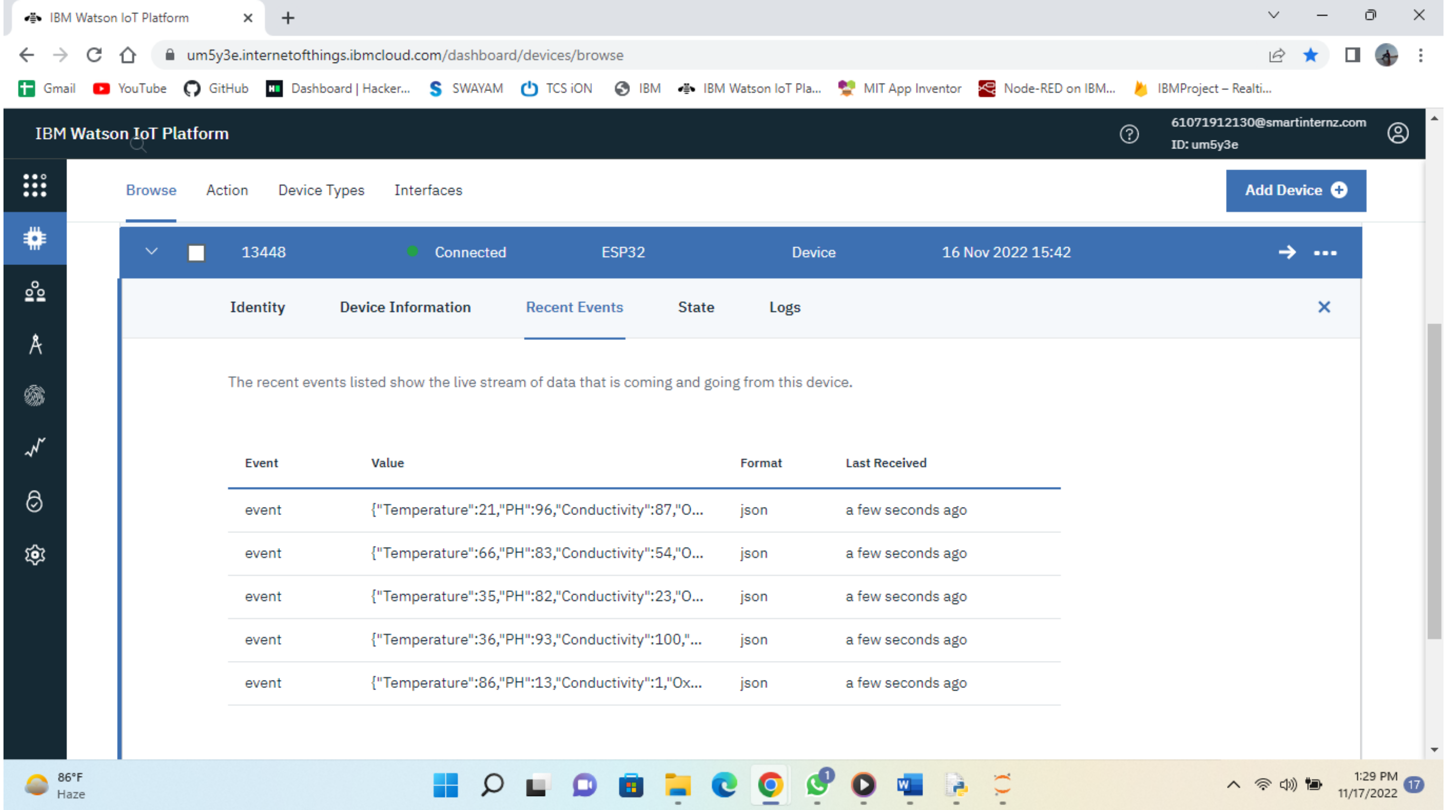
```
deviceCli.connect()

while True:
    ph = random.randint(0,100)
    cond = random.randint(0,100)
    temp = random.randint(0,100)
    oxy = random.randint(0,100)
    turb = random.randint(0,100)
    # Send Temperature & Humidity to IBM Watson
    data = {'Temperature': temp, 'PH': ph, 'Conductivity': cond, 'Oxygen': oxy, "Turbidity": turb}


    # print data
    def myOnPublishCallback():
        print("Published data",data, "to IBM Watson")


    success = deviceCli.publishEvent("event", "json", data, 0, myOnPublishCallback)
    if not success:
        print("Not connected to IoTF")
    time.sleep(5)

    deviceCli.commandCallback = myCommandCallback
```

USN – 7: Publish Data to cloud

As a user, I can publish Data that is sensed by the microcontroller to the Cloud



```
    deviceCli.commandCallback = myCommandCallback
2022-11-17 13:26:42,249   ibmiotf.device.Client      INFO    Connected successfully: d:um5y3e:ESP32:13448
Published data {'Temperature': 91, 'PH': 10, 'Conductivity': 35, 'Oxygen': 9, 'Turbidity': 17} to IBM Watson
Published data {'Temperature': 81, 'PH': 24, 'Conductivity': 70, 'Oxygen': 87, 'Turbidity': 89} to IBM Watson
Published data {'Temperature': 76, 'PH': 51, 'Conductivity': 65, 'Oxygen': 37, 'Turbidity': 21} to IBM Watson
Published data {'Temperature': 86, 'PH': 36, 'Conductivity': 1, 'Oxygen': 97, 'Turbidity': 28} to IBM Watson
Published data {'Temperature': 38, 'PH': 3, 'Conductivity': 99, 'Oxygen': 34, 'Turbidity': 51} to IBM Watson
Published data {'Temperature': 67, 'PH': 44, 'Conductivity': 45, 'Oxygen': 55, 'Turbidity': 11} to IBM Watson
Published data {'Temperature': 19, 'PH': 95, 'Conductivity': 9, 'Oxygen': 96, 'Turbidity': 11} to IBM Watson
Published data {'Temperature': 100, 'PH': 96, 'Conductivity': 35, 'Oxygen': 98, 'Turbidity': 80} to IBM Watson
Published data {'Temperature': 82, 'PH': 100, 'Conductivity': 77, 'Oxygen': 24, 'Turbidity': 50} to IBM Watson
Published data {'Temperature': 51, 'PH': 7, 'Conductivity': 58, 'Oxygen': 85, 'Turbidity': 52} to IBM Watson
Published data {'Temperature': 24, 'PH': 1, 'Conductivity': 77, 'Oxygen': 88, 'Turbidity': 58} to IBM Watson
Published data {'Temperature': 85, 'PH': 71, 'Conductivity': 58, 'Oxygen': 61, 'Turbidity': 79} to IBM Watson
Published data {'Temperature': 12, 'PH': 74, 'Conductivity': 87, 'Oxygen': 89, 'Turbidity': 51} to IBM Watson
Published data {'Temperature': 80, 'PH': 87, 'Conductivity': 28, 'Oxygen': 19, 'Turbidity': 38} to IBM Watson
Published data {'Temperature': 59, 'PH': 28, 'Conductivity': 82, 'Oxygen': 12, 'Turbidity': 72} to IBM Watson
Published data {'Temperature': 97, 'PH': 53, 'Conductivity': 8, 'Oxygen': 44, 'Turbidity': 80} to IBM Watson
Published data {'Temperature': 7, 'PH': 92, 'Conductivity': 71, 'Oxygen': 81, 'Turbidity': 93} to IBM Watson
Published data {'Temperature': 88, 'PH': 25, 'Conductivity': 37, 'Oxygen': 92, 'Turbidity': 23} to IBM Watson
Published data {'Temperature': 80, 'PH': 60, 'Conductivity': 65, 'Oxygen': 91, 'Turbidity': 4} to IBM Watson
Published data {'Temperature': 89, 'PH': 10, 'Conductivity': 50, 'Oxygen': 92, 'Turbidity': 74} to IBM Watson
Published data {'Temperature': 95, 'PH': 25, 'Conductivity': 95, 'Oxygen': 50, 'Turbidity': 17} to IBM Watson
Published data {'Temperature': 23, 'PH': 17, 'Conductivity': 42, 'Oxygen': 18, 'Turbidity': 99} to IBM Watson
Published data {'Temperature': 31, 'PH': 68, 'Conductivity': 71, 'Oxygen': 60, 'Turbidity': 51} to IBM Watson
Published data {'Temperature': 86, 'PH': 13, 'Conductivity': 1, 'Oxygen': 40, 'Turbidity': 14} to IBM Watson
Published data {'Temperature': 36, 'PH': 93, 'Conductivity': 100, 'Oxygen': 3, 'Turbidity': 17} to IBM Watson
Published data {'Temperature': 35, 'PH': 82, 'Conductivity': 23, 'Oxygen': 98, 'Turbidity': 88} to IBM Watson
Published data {'Temperature': 66, 'PH': 83, 'Conductivity': 54, 'Oxygen': 81, 'Turbidity': 12} to IBM Watson
Published data {'Temperature': 21, 'PH': 96, 'Conductivity': 87, 'Oxygen': 84, 'Turbidity': 60} to IBM Watson
Published data {'Temperature': 2, 'PH': 12, 'Conductivity': 98, 'Oxygen': 9, 'Turbidity': 41} to IBM Watson
Published data {'Temperature': 48, 'PH': 62, 'Conductivity': 32, 'Oxygen': 39, 'Turbidity': 89} to IBM Watson
Published data {'Temperature': 61, 'PH': 10, 'Conductivity': 23, 'Oxygen': 88, 'Turbidity': 47} to IBM Watson
```



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

Event	Value	Format	Last Received
event	{"Temperature":21,"PH":96,"Conductivity":87,"O...	json	a few seconds ago
event	{"Temperature":66,"PH":83,"Conductivity":54,"O...	json	a few seconds ago
event	{"Temperature":35,"PH":82,"Conductivity":23,"O...	json	a few seconds ago
event	{"Temperature":36,"PH":93,"Conductivity":100,"...	json	a few seconds ago
event	{"Temperature":86,"PH":13,"Conductivity":1,"Ox...	json	a few seconds ago