

Project Development Phase

Sprint – 2

Date	06 Nov 2022
Team ID	PNT2022TMID06701
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

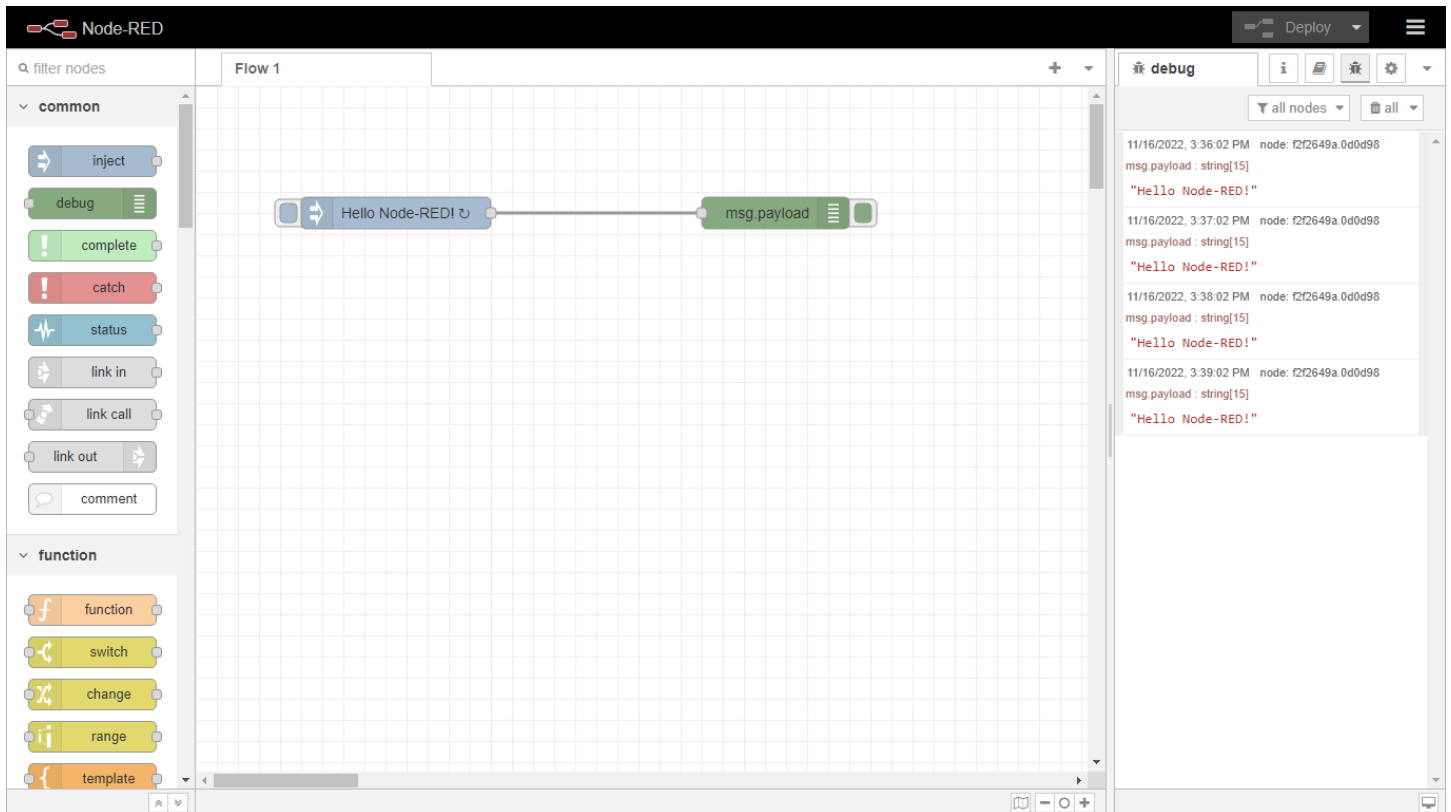
As a user, I can create the IBM Watson IoT Platform and integrate the microcontroller with it, to send the sensed data on Cloud.

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. The main content area is titled 'Browse Devices' and contains a table of devices. A device with ID 'iot' is listed as 'Disconnected'. A 'Simulations' panel is open on the right, showing '1/50 Simulations Running' and options to 'Create Simulated Device' or 'Use Registered Device'. The bottom status bar shows '29 events sent' and '1.27 KB sent'.

Device ID	Status	Device Type	Class ID	Date
iot	Disconnected	123	Device	Nov 1

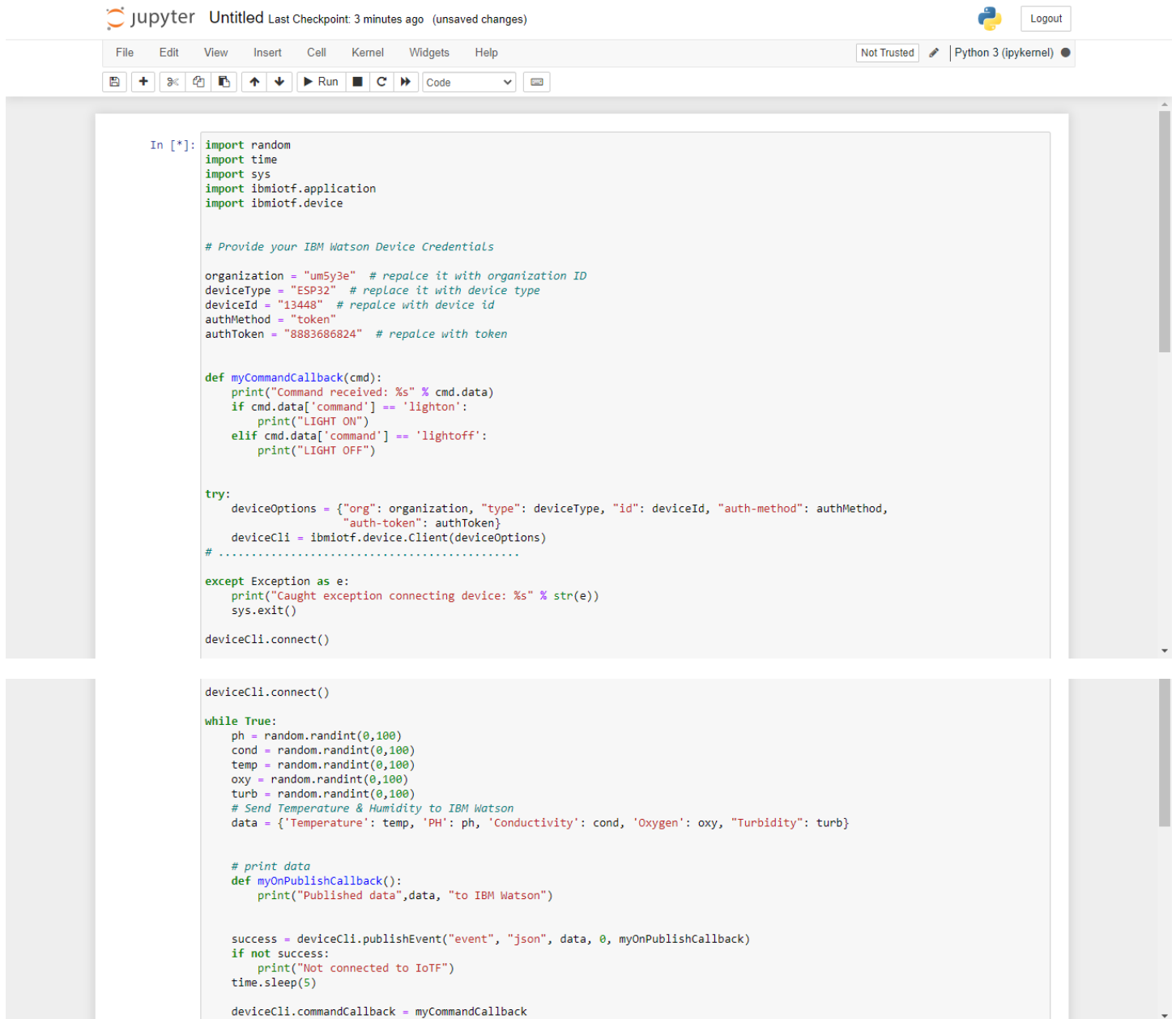
USN – 5: Create a node red service

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



USN – 6: To develop a Python code

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



The image shows a Jupyter Notebook interface with a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar. The notebook is titled "Untitled" and shows the last checkpoint was 3 minutes ago with unsaved changes. The code is written in Python 3 (ipykernel). The code defines a command callback function, sets up IBM Watson IoT credentials, connects to the device, and publishes random data to IBM Watson IoT.

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

# Provide your IBM Watson Device Credentials

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

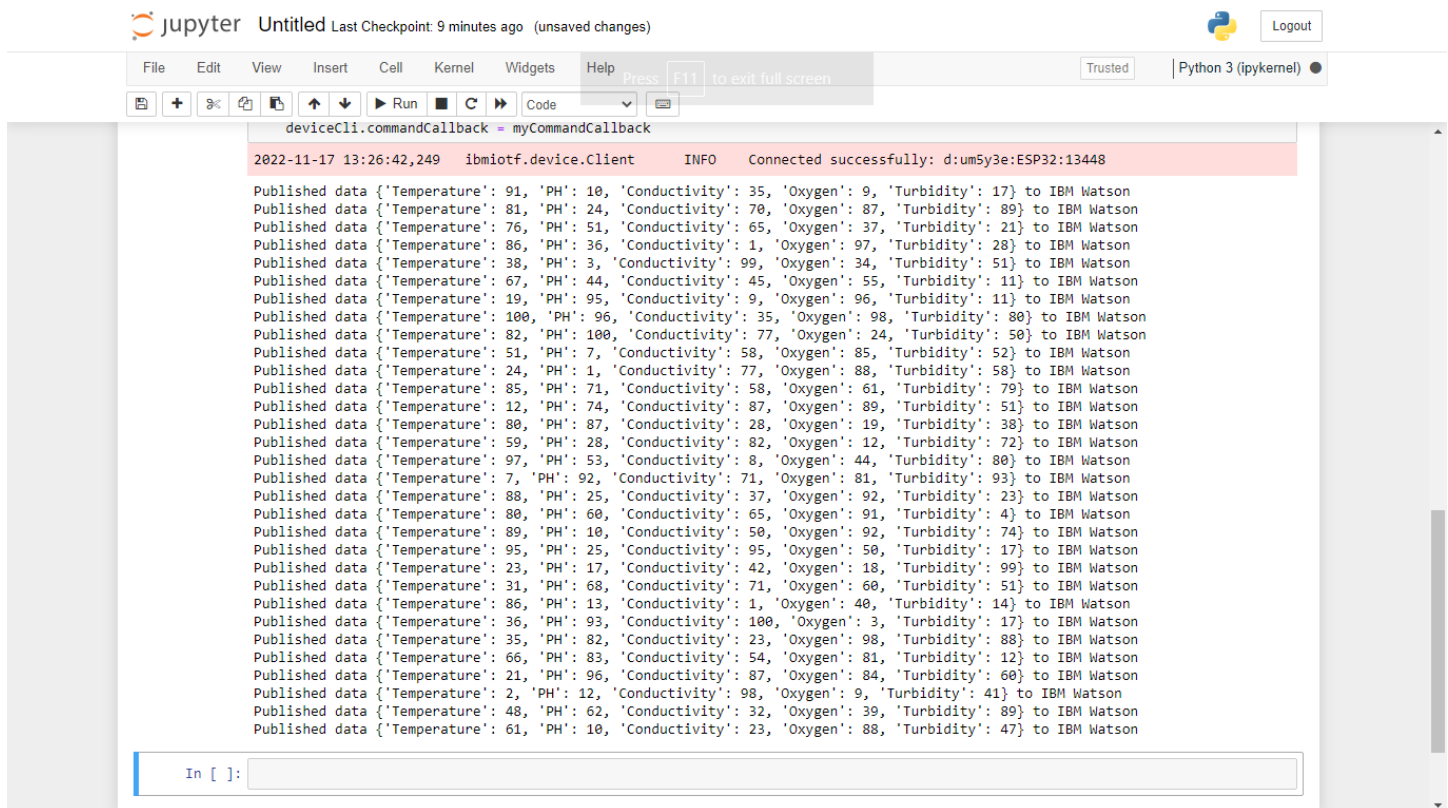
    # print data
    def myOnPublishCallback():
        print("Published data",data, "to IBM Watson")

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

    deviceCli.commandCallback = myCommandCallback
```

USN – 7: Publish Data to cloud

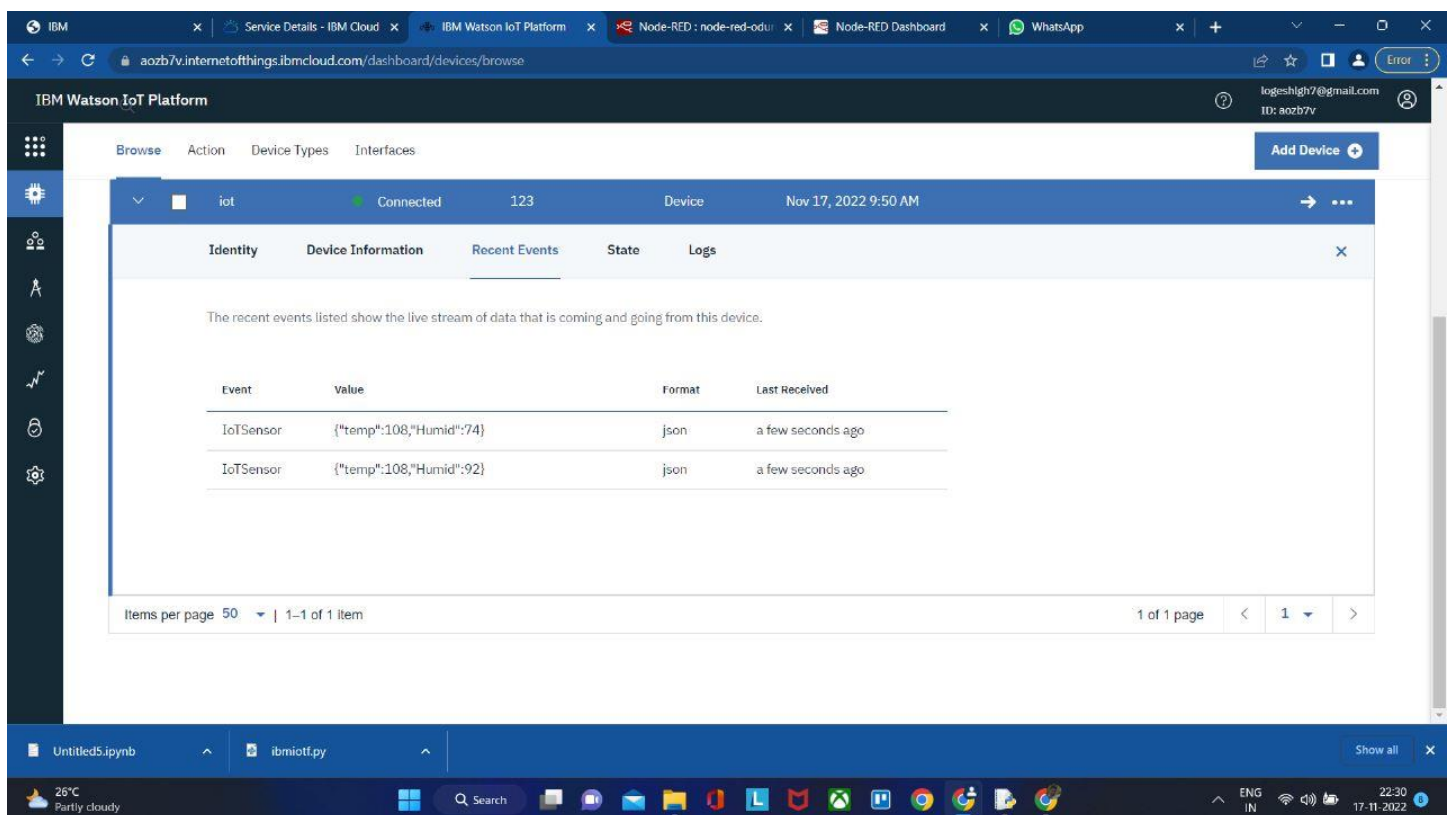
As a user, I can publish Data that is sensed by the microcontroller to the Cloud



The screenshot shows a Jupyter Notebook titled 'Untitled' with a 'Last Checkpoint: 9 minutes ago' and '(unsaved changes)'. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and code execution. The code cell displays the output of a device connection and a series of published data points to IBM Watson.

```
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 screenshot displays the IBM Watson IoT Platform dashboard. The top navigation bar includes links to 'Service Details - IBM Cloud', 'IBM Watson IoT Platform', 'Node-RED: node-red-odu', 'Node-RED Dashboard', and 'WhatsApp'. The main content area shows the 'Browse' tab selected, displaying a list of devices. A modal window titled 'Identity' is open, showing 'Recent Events' for a device. The events table lists two recent data points from an 'IoT Sensor'.

Event	Value	Format	Last Received
IoT Sensor	{"temp":108,"Humid":74}	json	a few seconds ago
IoT Sensor	{"temp":108,"Humid":92}	json	a few seconds ago