



**GOVERNMENT COLLEGE OF ENGINEERING
CHETTIKARAI, DHARMAPURI**



**SMART FARMER – IOT ENABLED SMART FARMING
APPLICATION**

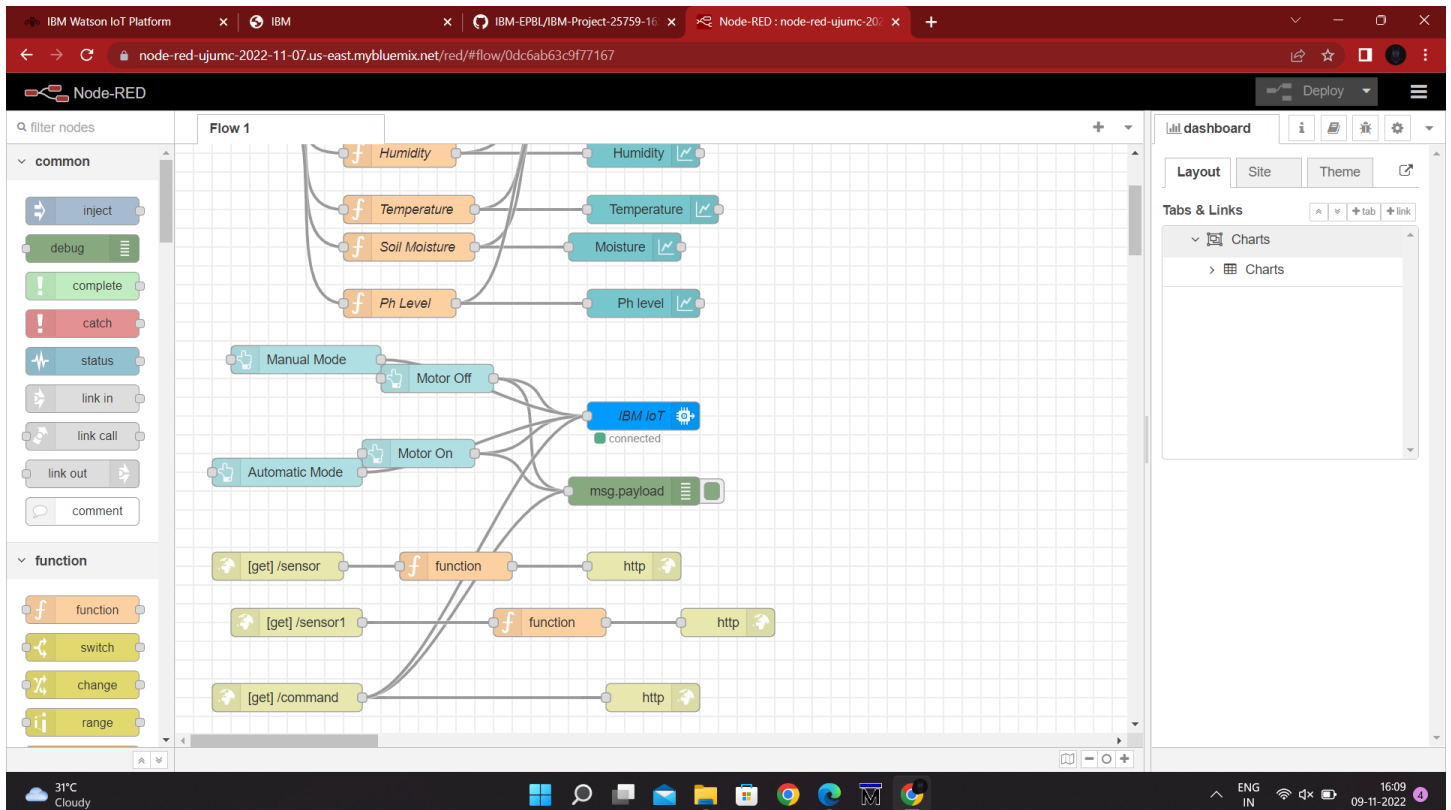
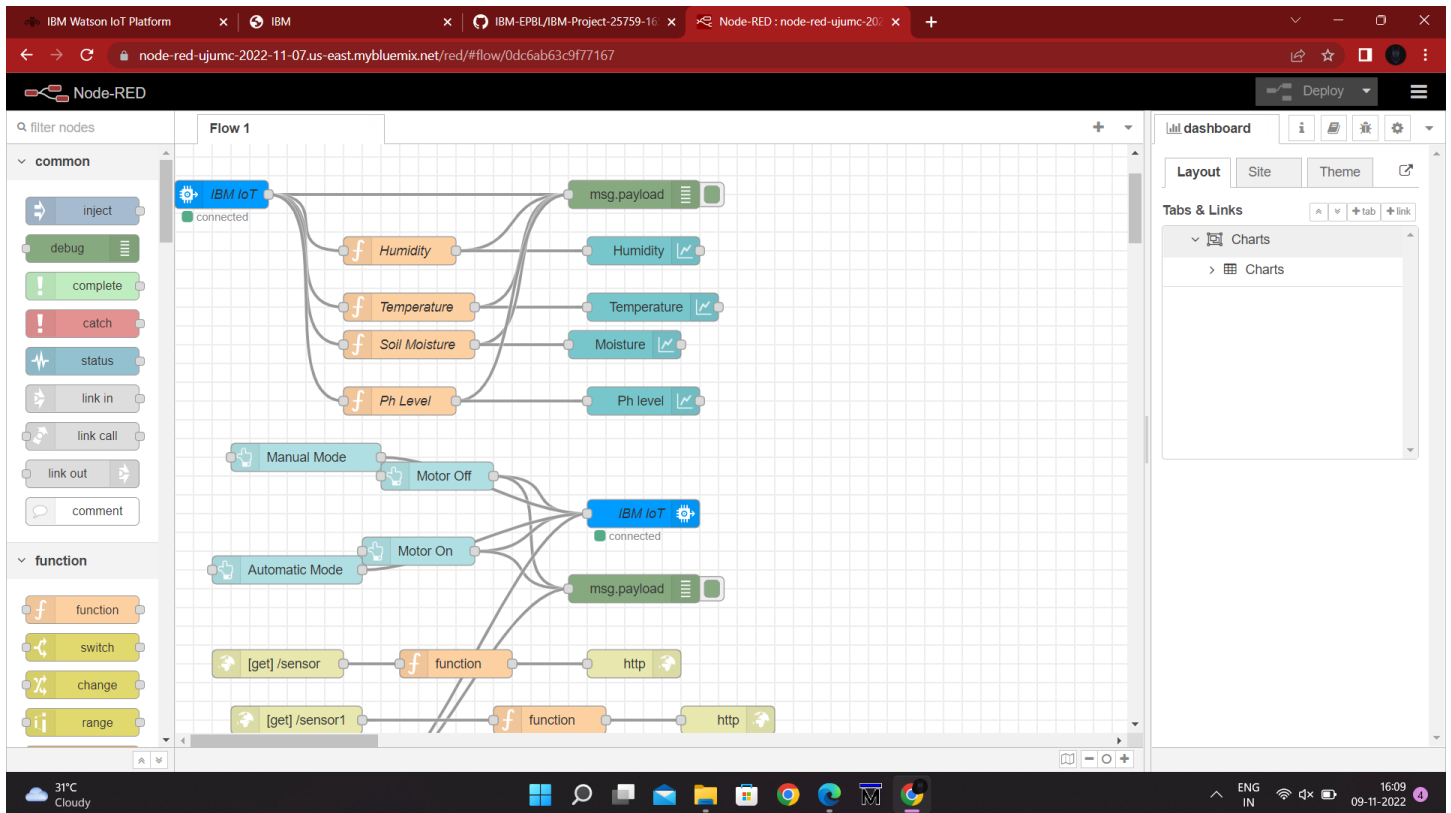
IBM NALAIYATHIRAN

Project Development-Delivery of Sprint 2

Creating Node-Red service and connect with IBM cloud and Web UI

TITLE	Smart Farmer IoT Enabled Smart Farming Application
DOMAIN NAME	INTERNET OF THINGS
TEAM ID	PNT2022TMID41287
TEAM LEADERNAME	MITHUN SRINIVASAN S
TEAM MEMBER NAME	ARUN KUMAR M AJITH KUMAR S RAVIN G
MENTOR NAME	Dr. DINESH G

Creating Node-Red service:



Connecting With IBM Cloud:

Using IBM IOT node through API key:

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes the IBM logo and a user profile with the email 613519106030@smartinternz.com and ID: kv09p4. A 'Generate API Key' button is visible in the top right. The main content area displays a table of API keys with columns for Key, Description, Role, and Expires. Two results are shown:

Key	Description	Role	Expires
a-kv09p4-ffvzkrsmu	API Key for the device simulator	Standard Application	-
a-kv09p4-kp3tgvwqjd	IBM	Standard Application	-

Below the table, the 'API Key Information' for the selected key is displayed:

Field	Value
Key	a-kv09p4-kp3tgvwqjd
Description	IBM
Date Added	Nov 7, 2022 4:26 PM
Last Update	Nov 7, 2022 4:26 PM
Last Edited By	613519106030@smartinternz.com
Expires	Never

Transferring Values from Python Code:

The screenshot shows a Python script in a text editor and its execution output in a terminal window.

Python Script (ibm_iot.py):

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

#Provide your IBM Watson Device Credentials
organization = "kv09p4"
deviceType = "Groot"
deviceId = "13"
authMethod = "token"
authToken = "12345678"
global y
# Initialize GPIO

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
    if status=="motoroff":
        print ("motor is off")
    if status=="manual":
        print ("Motor Control is in Manual Mode")
    if status=="automatic":
        print ("Motor control is in Automatic Mode")
        if soilmoisture > 600:
            print ("motor is on")

    #print(cmd)

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

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

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

Python 3.7.0 Shell Output:

```
Python 3.7.0 (tags/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: D:\IBM project\IBM iot.py =====
2022-11-09 16:20:43,237 ibmiotf.device.Client INFO Connected successfully: d:kv09p4:Groot:13
Published Temperature = 21 C Humidity = 80 % Soil Moisture is 14 % PH level is 13 to IBM Watson
Published Temperature = 51 C Humidity = 5 % Soil Moisture is 904 % PH level is 6 to IBM Watson
```

Node-Red:

The screenshot displays the Node-RED web interface in a browser. The top navigation bar includes links to IBM Watson IoT Platform, IBM, and the current project. The main workspace shows a flow named 'Flow 1' with the following components:

- Inputs:** An 'IBM IoT' node (connected) feeds into four function nodes: 'Humidity', 'Temperature', 'Soil Moisture', and 'Ph Level'.
- Outputs:** Each function node connects to a corresponding 'msg.payload' node, which then connects to a visualizer node (line graph).
- Control Logic:** A 'Manual Mode' button connects to a 'Motor Off' node. An 'Automatic Mode' button connects to a 'Motor On' node. Both motor nodes connect to an 'IBM IoT' node (connected).
- Data Retrieval:** Two '[get] /sensor' nodes connect to 'function' nodes, which then connect to 'http' nodes.

The right sidebar shows the 'debug' console with a list of messages. The messages are JSON objects containing sensor data, such as:

```
{ "temp": 16, "Humid": 23, "soilmoisture": 1016, "Phlevel": 6 }
```

The bottom status bar shows the system clock and weather information (31°C Cloudy).

Node-Red Dashboard:

The screenshot displays the Node-RED Dashboard web interface. The top navigation bar includes links to IBM Watson IoT Platform, IBM, and the current project. The main workspace shows three charts under the 'Charts' tab:

- Temperature:** A line graph showing temperature data over time. The y-axis ranges from 0 to 100. The x-axis shows time from 15:22:00 to 16:23:00.
- Moisture:** A line graph showing moisture data over time. The y-axis ranges from 0 to 1,023. The x-axis shows time from 15:22:00 to 16:23:00.
- Ph level:** A line graph showing pH level data over time. The y-axis ranges from 3.5 to 14. The x-axis shows time from 15:22:00 to 16:23:00.

The bottom status bar shows the system clock and weather information (31°C Cloudy).

Charts

