

Smart Farmer-IOT Enabled Smart Farming Application

SPRINT DELIVERY – 3

TITLE	Smart Farmer-IOT Enabled Smart Farming Application
DOMAIN NAME	INTERNET OF THINGS
TEAM ID	PNT2022TMID22828
LEADER NAME	KOWSALYA D
TEAM MEMBER NAME	KAMALAKANNAN R KARTHICK S NITHEEN V P

Configuration of Node-Red to send commands to IBM cloud

Ibm iot out node I used to send data from Node-Red to IBM Watson device. So, after adding it to the flow we need to configure it with credentials of our Watson device.

The screenshot shows the 'Edit ibmiot in node' configuration window in Node-RED. The window has a title bar with 'Node-RED : node-red-hdyfv-202' and a close button. Below the title bar is the URL 'ed/#flow/c7ddb1462b8a000c'. The main content area is titled 'Edit ibmiot in node' and contains several configuration options:

- Authentication:** A dropdown menu set to 'API Key'.
- API Key:** A text input field containing 'IBMIOT APIKEY' with a search icon on the left and an edit icon on the right.
- Input Type:** A dropdown menu set to 'Device Event'.
- Device Type:** A checkbox labeled 'All or' followed by a text input field containing 'abcd'.
- Device Id:** A checkbox labeled 'All or' followed by a text input field containing '7654321'.
- Event:** A checkbox labeled 'All or' followed by a text input field containing '+'. The checkbox is checked.
- Format:** A checkbox labeled 'All or' followed by a text input field containing 'json'.
- QoS:** A dropdown menu set to '0'.
- Name:** A text input field containing 'IBM IoT'.

At the bottom of the window, there is a checkbox labeled 'Enabled' which is checked. The window also has 'Delete', 'Cancel', and 'Done' buttons at the top right.

Here we add two buttons in UI

1 -> for motor on

2 -> for motor off

We used a function node to analyse the data received and assign command to each number.

The Java script code for the analysis is:

```
if(msg.payload===1)
```

```
msg.payload={"command": "ON"}; else
```

```
if(msg.payload===0)
```

```
msg.payload={"command": "OFF"};
```

Then we use another function node to parse the data and get the command and represent it visually with text node.

The Java script code for that function node is:

```
var state=msg.payload;  
msg.payload = state.command;  
return msg;
```

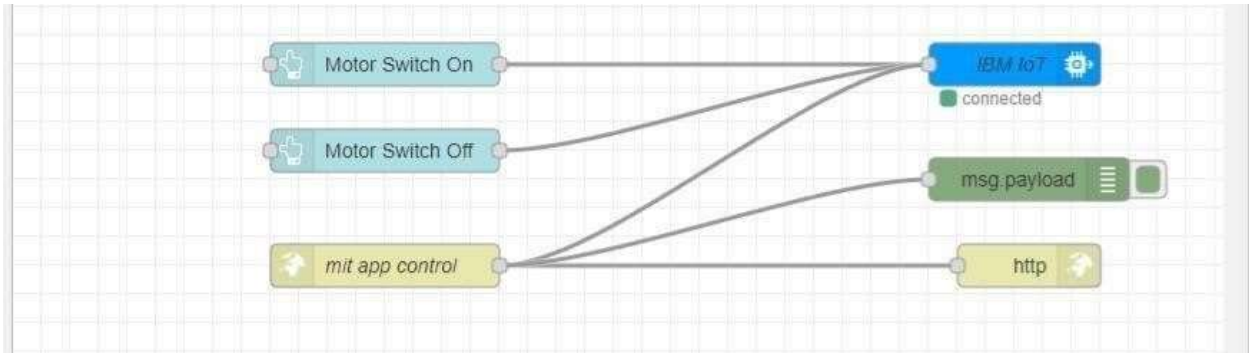


The above images show the java script codes of analyser and state function nodes.

Then we add edit Json node to the conversion between JSON string & object and finally connect it to IBM IoT Out.



Edit JSON node needs to be configured like this



This is the program flow for sending commands to IBM cloud.

5.5 Adjusting User Interface

In order to display the parsed JSON data a Node-Red dashboard is created

Here we are using Gauges, text and button nodes to display in the UI and helps to monitor the parameters and control the farm equipment.

Below images are the Gauge, text and button node configurations.

Complete Program Flow

The screenshot displays the Node-RED web interface with a workflow designed to publish sensor data to the IBM IoT Platform. The workflow consists of the following nodes:

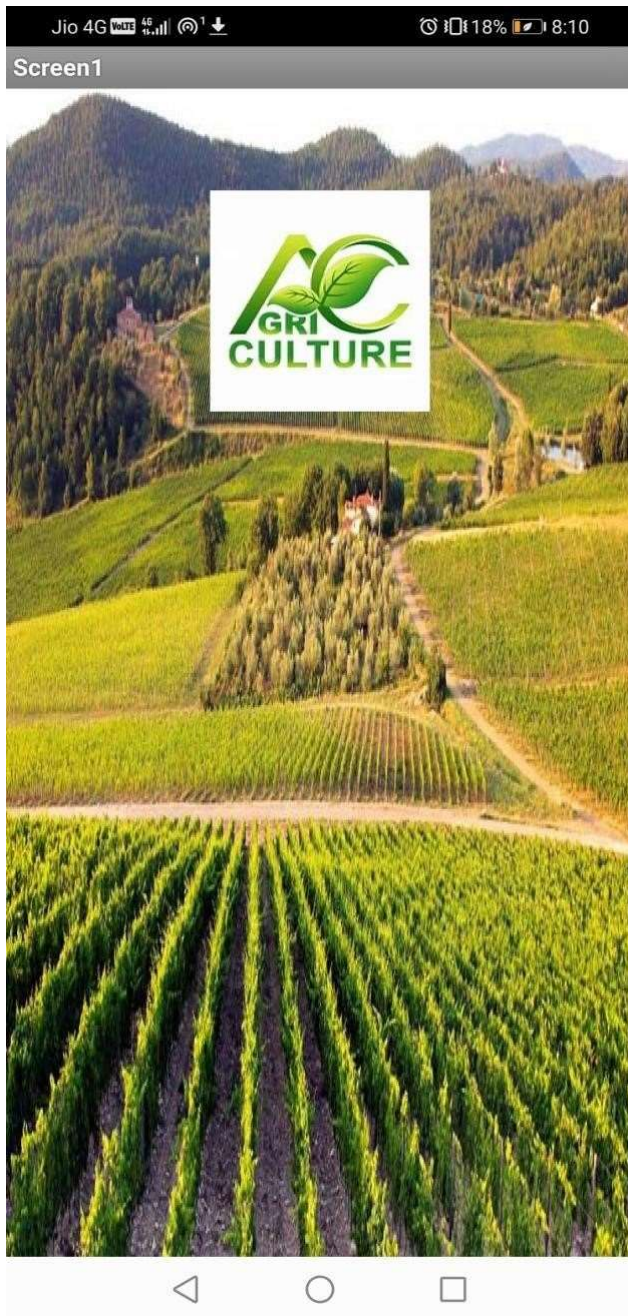
- msg.payload**: The starting node for the message.
- Temperature**: A function node that processes the temperature data.
- humidity**: A function node that processes the humidity data.
- ph**: A function node that processes the pH data.
- IBM IoT**: The final node responsible for publishing the data to the IBM IoT Platform.

The terminal window on the left shows the execution output, indicating successful data publication:

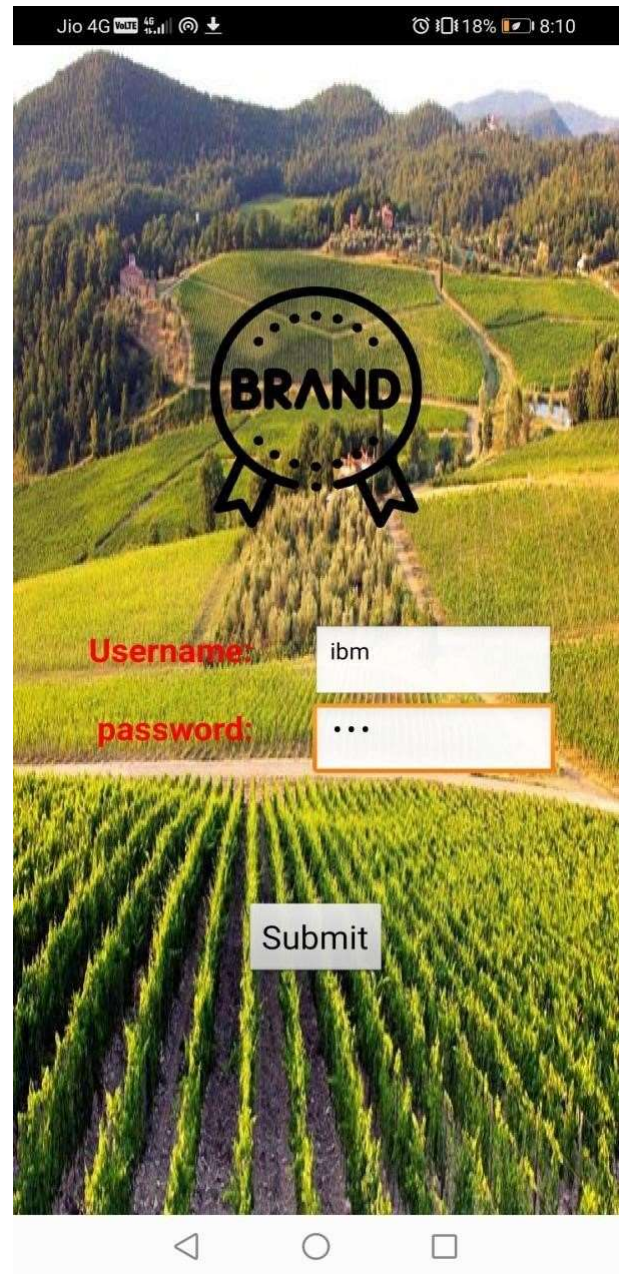
```
Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\rkama\AppData\Local\Programs\Python\Python39\ibm code.py ==
2022-11-19 13:52:11,429 wiotp.sdk.device.client.DeviceClient INFO Connecte
d successfully: d:tdo49a:NodeMCU:12345
Published data Successfully: %s ('soil_moisture': 75, 'temperature': 104, 'humid
ity': 82)
Published data Successfully: %s ('soil_moisture': 84, 'temperature': 69, 'humid
ity': 39)
Published data Successfully: %s ('soil_moisture': 42, 'temperature': 4, 'humidit
y': 95)
```

The debug console on the right shows the message payload being sent to the IBM IoT Platform, including the following data:

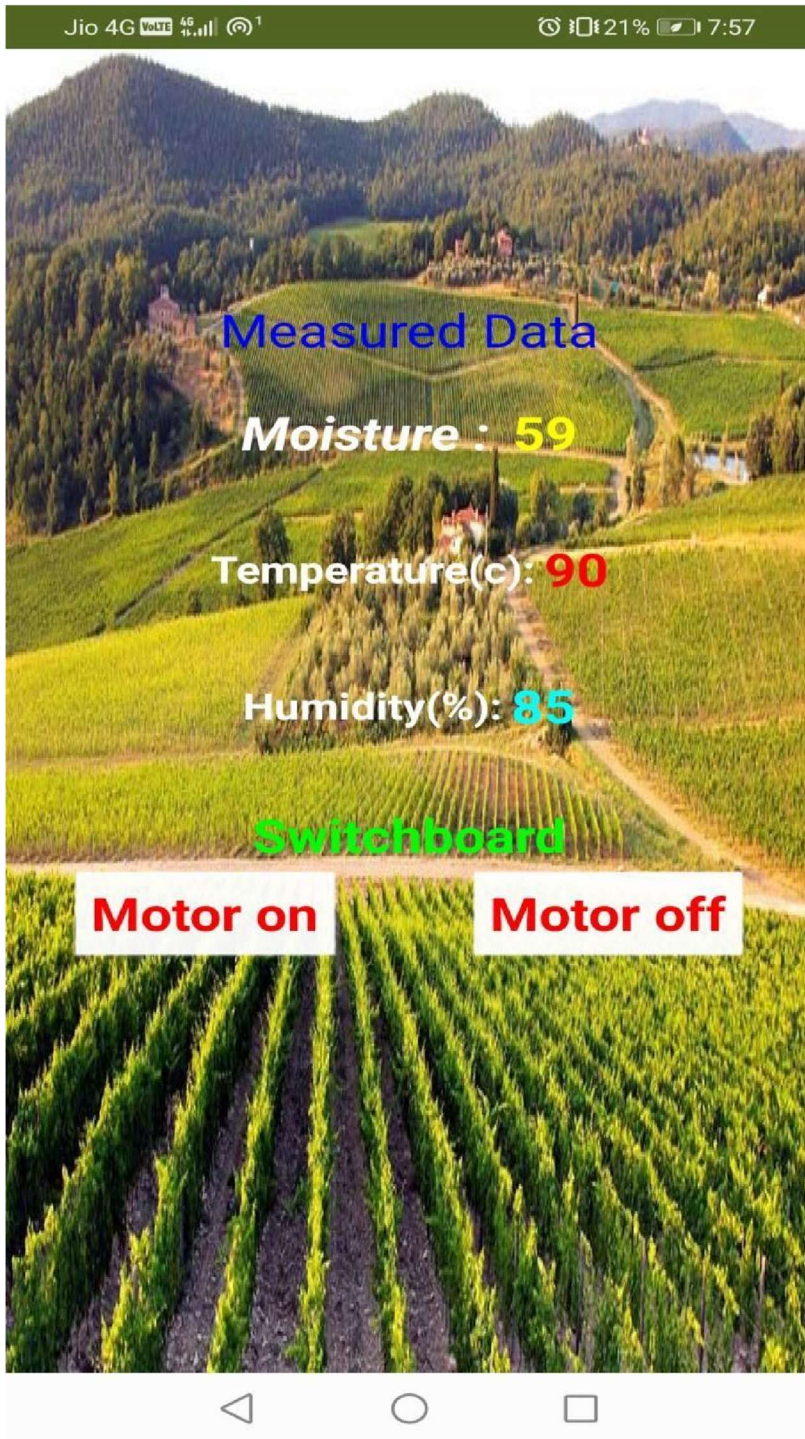
```
{
  "soil_moisture": 42,
  "temperature": 4,
  "humidity": 95
}
```

SCREEN – 1



SCREEN - 2



SCREEN - 3 Web APP UI Home Tab

PH



LIGHT ON

