

## SPRINT-3

Date	13 November 2022
Project Name	Smart Farmer - IoT Enabled Smart Farming Application
TEAM ID	PNT2022TMID15139

### PROCEDURE:

- Open IBM Cloud and search Node-RED in catalog.
- Create Node-RED configuration and visit the app URL.
- Construct all nodes for Smart farm and enter all configuration in nodes.
- After configuring click deploy and see the output in python IDLE and debug
- Visit the api web and monitor the temperature and humidity reading.

### Creation of Node-RED

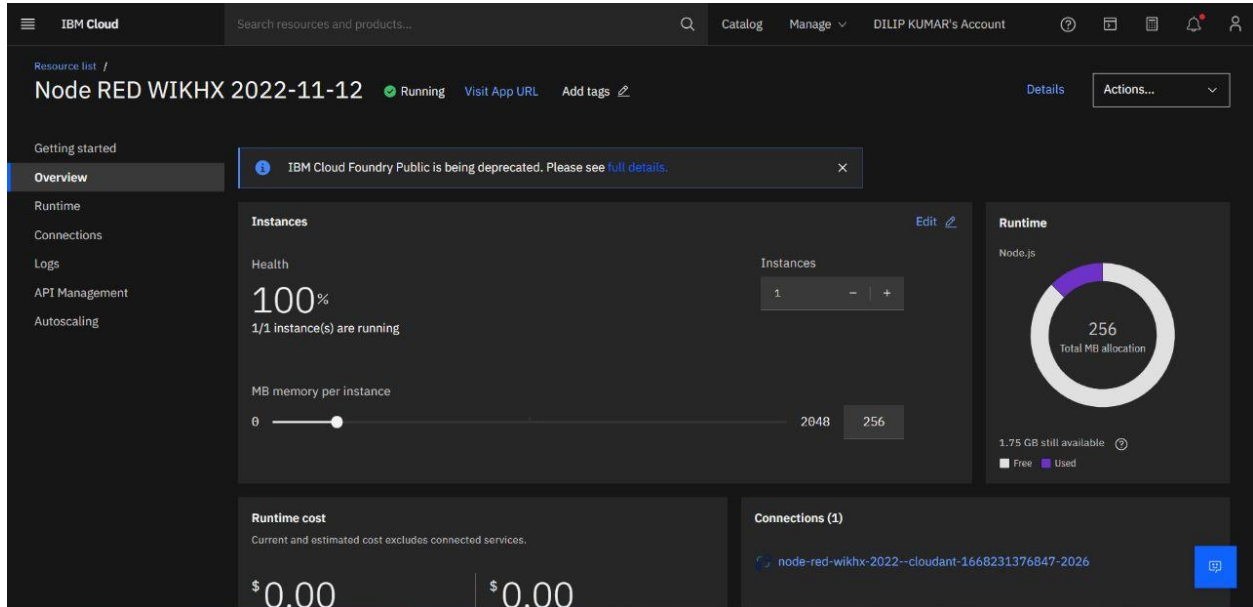
The screenshot displays the IBM Cloud console for a Node-RED application named 'Node RED WIKHX 2022-11-12'. The top navigation bar includes the IBM Cloud logo, a search bar, and user account information. The main content area is divided into several sections:

- Details:** Contains fields for App URL, Source (with a 'Download code' button), Resource group (set to 'Default'), Deployment target, and Created date (11/12/2022).
- Services:** Features a 'Cloudant' service card with links to 'Open dashboard', 'Documentation', and 'API reference', along with a 'Credentials' dropdown and buttons to 'Connect existing services' and 'Create service'.
- Deployment Automation:** Includes a 'Configure Continuous Delivery' section with a 'Deploy your app' button.
- Getting started quickly:** A sidebar with a list of steps for configuring the app, connecting services, and deploying.

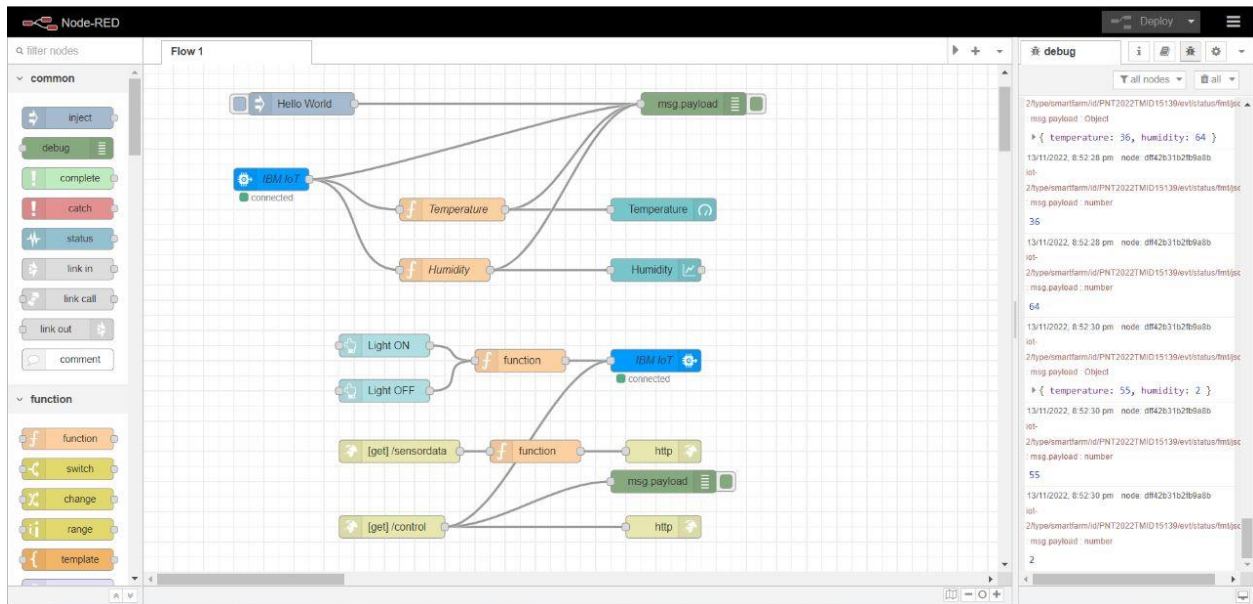
The 'Getting started quickly' sidebar contains the following steps:

1. Use the **Services** card to connect a service to your app. Select an existing service instance, or create a new one. [Learn more.](#)
2. If you want to view the code before your app is deployed, click **Download code** to obtain the .zip file.
3. Click **Deploy your app** in the **Deployment Automation** card to select the deployment target and configure the Continuous Delivery service. The deployment begins automatically.
4. After the deployment begins, you can view the status of the deployment, modify your app, view your repo, or...

## Run the Node-RED and launch the URL from IBM Cloud



## Construction of Nodes



## Configuring the IBM Nodes

The screenshot displays the Node-RED web interface. On the left, the 'common' and 'function' node palettes are visible. The main workspace shows a flow starting with a 'Hello World' node, followed by an 'IBM IoT' node (labeled 'connected'). This node is connected to two function nodes labeled 'Temperature' and 'Humidity'. These function nodes are further connected to 'Light ON' and 'Light OFF' nodes, which then connect to 'function' nodes. The 'Edit ibmiot out node' panel is open on the right, showing the following configuration:

- Authentication:** API Key
- API Key:** d53fad203f76f3a
- Output Type:** Device Command
- Device Type:** smartfam
- Device Id:** PNT2022TMID15139
- Command Type:** cmd
- Format:** json
- Data:** data
- QoS:** 0
- Name:** IBM IoT
- Service:** registered

A note at the bottom of the configuration panel states: "Note: if there is a property in the message that corresponds to any of the values entered above, then the property in the message takes precedence. See the Info tab for more details." The 'debug' console on the far right shows a series of messages, including objects like { temperature: 79, humidity: 13 } and numbers like 79, 13, 24, 12.

## Interfacing Node-RED and MIT App Inventor

The screenshot displays the Node-RED web interface. On the left, the 'common' and 'function' node palettes are visible. The main workspace shows a flow starting with a 'Hello World' node, followed by an 'IBM IoT' node (labeled 'connected'). This node is connected to two function nodes labeled 'Temperature' and 'Humidity'. These function nodes are further connected to 'Light ON' and 'Light OFF' nodes, which then connect to 'function' nodes. The 'Edit function node' panel is open on the right, showing the following configuration:

- Name:** Name
- Setup:** On Start, On Message, On Stop
- Code:**

```
1 = msg.payload;
2 "temperature":global.get('t'),
3 "humidity":global.get('h'),
4 = }
5 return msg;
```

The 'debug' console on the far right shows a series of messages, including objects like { temperature: 107, humidity: 45 } and numbers like 45, -8, 4.