



**IOT ENABLED SMART FARMING APPLICATION  
NALAIYA THIRAN PROJECT BASED LEARNING  
ON  
PROFESSIONAL READINESS FOR INNOVATION,  
EMPLOYABILITY AND ENTREPRENEURSHIP**

**PROJECT REPORT**

*Submitted by*

**JAYASHARIMILA.K**

**(820619104018)**

**MONICA.U.M**

**(820619104032)**

**VIGNESHWARI.B**

**(820619104067)**

**ANUGRAHAA.R**

**(820619104302)**

**COMPUTER SCIENCE AND ENGINEERING  
ARASU ENGINEERING COLLEGE, KUMBAKONAM**

**ANNA UNIVERSITY: CHENNAI 600 025**

**NOVEMBER-2022**

# CONTENTS

## **1. INTRODUCTION**

- 1.1 Project Overview
- 1.2 Purpose

## **2. LITERATURE SURVEY**

- 2.1 Existing problem
- 2.2 References
- 2.3 Problem Statement Definition

## **3. IDEATION & PROPOSED SOLUTION**

- 3.1 Empathy Map Canvas
- 3.2 Ideation & Brainstorming
- 3.3 Proposed Solution
- 3.4 Problem Solution fit

## **4. REQUIREMENT ANALYSIS**

- 4.1 Functional requirement
- 4.2 Non-Functional requirements

## **5. PROJECT DESIGN**

- 5.1 Data Flow Diagrams
- 5.2 Solution & Technical Architecture
- 5.3 User Stories

## **6. PROJECT PLANNING & SCHEDULING**

- 6.1 Sprint Planning & Estimation
- 6.2 Sprint Delivery Schedule
- 6.3 Reports from JIRA

## **7. CODING & SOLUTIONING (Explain the features added in the project along with code)**

- 7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

## **8. TESTING**

8.1 Test Cases

8.2 User Acceptance Testing

## **9. RESULTS**

9.1 Performance Metrics

## **10.ADVANTAGES & DISADVANTAGES**

## **11.CONCLUSION**

## **12.FUTURE SCOPE**

## **13.APPENDIX**

Source Code

GitHub & Project Demo Link

# 1. Introduction

## 1.1 Project Overview

IoT-based agriculture system helps the farmer in monitoring different parameters of his field like soil moisture, temperature, and humidity using some sensors. Farmers can monitor all the sensor parameters by using a web or mobile application even if the farmer is not near his field. Watering the crop is one of the important tasks for the farmers. They can make the decision whether to water the crop or postpone it by monitoring the sensor parameters and controlling the motor pumps from the mobile application itself.

## 1.2 Purpose

Increasing control over production leads to better cost management and waste reduction. The ability to trace anomalies in crop growth or livestock health, for instance, helps eliminate the risk of losing yields. Additionally, automation boosts efficiency. Smart farming reduces the ecological footprint of farming. Minimized or site-specific application of inputs, such as fertilizers and pesticides, in precision agriculture systems will mitigate leaching problems as well as the emission of greenhouse gases.

## 2. Literature Survey

### 2.1 Existing Problem

IoT based Smart Farming improves the entire Agriculture system by monitoring the field in real-time. With the help of sensors and interconnectivity, the Internet of Things in Agriculture has not only saved the time of the farmers but has also reduced the extravagant use of resources such as Water and Electricity. Climate plays a very critical role for farming. And having improper knowledge about climate heavily deteriorates the quantity and quality of the crop production. Precision Agriculture/Precision Farming is one of the most famous applications of IoT in Agriculture. It makes the farming practice more precise and controlled by realizing smart farming applications such as livestock monitoring, vehicle tracking, field observation, and inventory monitoring. To make our greenhouses smart, IoT has enabled weather stations to automatically adjust the climate conditions according to a particular set of instructions. Adoption of IoT in Greenhouses has eliminated the human intervention, thus making entire process cost-effective and increasing accuracy at the same time.

### 2.2 References

1, Sustainable agriculture by the Internet of Things – A practitioner's approach to monitor sustainability progress. 2022, Computers and Electronics in Agriculture.

2, The Interplay between the Internet of Things and agriculture: A metric analysis and research agenda. 2022, International Journal of Intelligent Networks.

### 3, Agriculture 4.0 and its Barriers in the Agricultural Production Chain

Development in Southern Brazil. 2022, SSRN

4, IoT based Agriculture (IoTA): Architecture, Cyber Attack, Cyber Crime and Digital Forensics Challenges. 2022, Research Square.

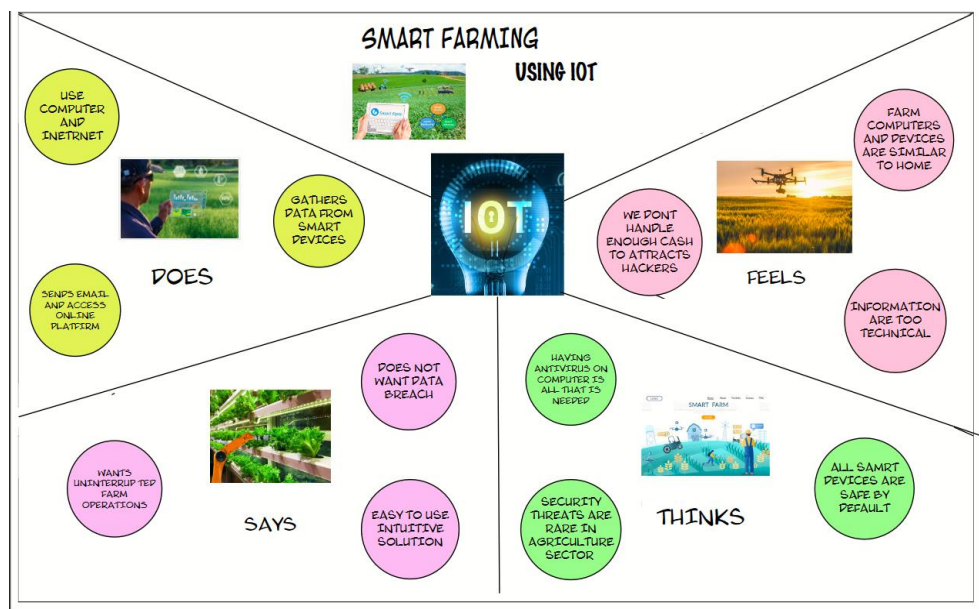
## 2.3 Problem Statement Solution

The traditional agriculture and allied sector cannot meet the requirements of modern agriculture which requires high-yield, high quality and efficient output. Thus, it is very important to turn towards modernization of existing methods and using the information technology and data over a certain period to predict the best possible productivity and crop suitable on the very particular land. The adoptions of access to high-speed internet, mobile devices, and reliable, low-cost satellites (for imagery and positioning) are few key technologies characterizing the precision agriculture trend. Precision agriculture is one of the most famous applications of IoT in the agricultural sector and numerous organizations are leveraging this technique around the world. Some products and services in use are VRI optimization, soil moisture probes, virtual optimizer PRO, and so on. VRI (Variable Rate Irrigation) optimization maximizes profitability on irrigated crop fields with topography or soil variability, improve yields, and increases water use efficiency. IoT has been making deep inroads into sectors such as manufacturing, health-care and automotive. When it comes to food production, transport and storage, it offers a breadth of options that can improve

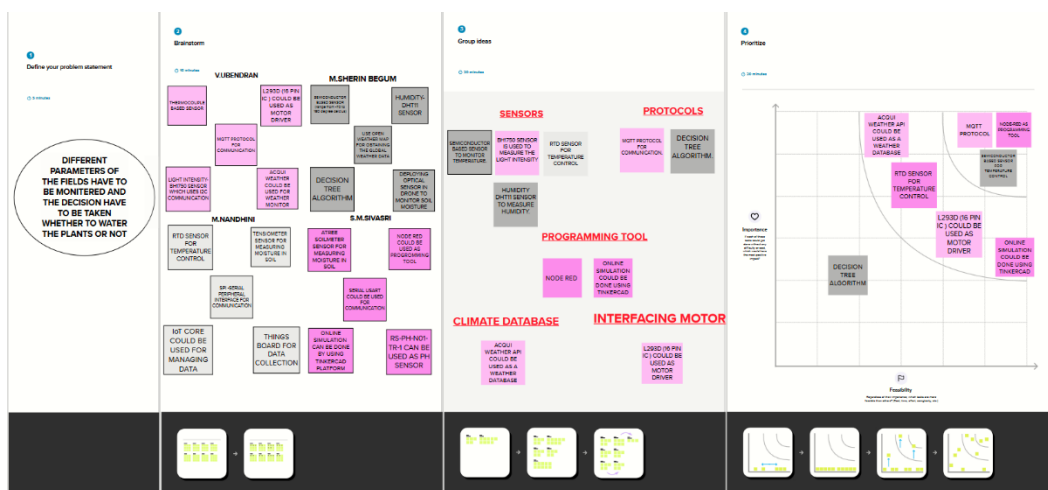
India's per capita food availability. Sensors that offer information on soil nutrient status, pest infestation, moisture conditions etc. which can be used to improve crop yields over time. Some of the sample problem statements related to Agriculture & allied sectors where IoT application will be beneficial are given below.

### 3.IDEATION &PROPOSED SYSTEM

#### 3.1 Prepare Empathy Map



#### 3.2 Ideation



### 3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Farmers should be in the farm field to monitor their crop field, if any emergency occurs for farmer to go outside there will be lack of irrigation in farm field which lead to damage in crops health
2.	Idea / Solution description	IoT-based agriculture system helps the farmer to monitoring different parameters of his field like soil moisture, temperature, and humidity using some sensors by using a web or mobile application
3.	Novelty / Uniqueness	when the farmer is not near his field, he can make the decision whether to water the crop or postpone it by monitoring the sensor parameters and controlling the motor pumps from the mobile application itself.
4.	Social Impact / Customer Satisfaction	A monthly subscription is charged to farmers for prediction and suggesting the irrigation timing based on sensors parameters like temperature ,humidity, soil moisture.
5.	Business Model (Revenue Model)	A monthly subscription is charged to farmers for prediction and suggesting the irrigation timing based on sensors parameters like temperature, humidity, soil moisture.
6.	Scalability of the Solution	Image recognition-based prediction of crops health Ai based automated irrigation using temperature, pressure, humidity, and soil moisture sensors



## 3.4 Proposed Solution Fit

Problem-Solution fit canvas 2.0			Purpose / Vision	
Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <b>CS</b> <i>The customers of this product are the farmers who cultivate crops. Our aim is to assist, aid and help them to monitor the field parameters remotely and to keep track of the parameters. This product saves the agriculture from extinction.</i>	<b>6. CUSTOMER CONSTRAINTS</b> <b>CC</b> <i>Deployment of huge number of sensors is difficult. It requires an unlimited or continuous internet connection to be successful.</i>	<b>5. AVAILABLE SOLUTIONS</b> <b>AS</b> <i>The irrigation process is automated using IoT. weather data and field parameters were obtained and processed to automate the process of irrigation. The drawbacks are high cost of installation, efficient only for short distance, difficulty in storing the data.</i>	Explore AS, differentiate
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <b>J&amp;P</b> <i>The objective of this product is to obtain the different field parameters using sensor and process it using a central processing system. Cloud is used to store and transmit the data by using IoT. Weather APIs are employed to assist the farmer in making decision. The farmer could take decision through a mobile application.</i>	<b>9. PROBLEM ROOT CAUSE</b> <b>RC</b> <i>The frequent change or unpredictable weather and climate, made it difficult for the farmers to do agriculture. These factors play a major role in making decision whether to water the plant or not. The monitoring of the field is hard when the farmer is out of station, thus leading to crop damage.</i>	<b>7. BEHAVIOUR</b> <b>BE</b> <i>Using proper drain system to overcome the effects of excess water due to heavy rain. Using hybrid varieties of crop that are resistant to pests.</i>	
Identify strong TR & EM	<b>3. TRIGGERS</b> <b>TR</b> <i>Farmers facing issues in providing proper irrigation. No proper supply of water leads to reduced production which affects the profit level of the farmer. Farmer's struggle to predict the weather.</i>	<b>10. YOUR SOLUTION</b> <b>SL</b> <i>Our product collects the data from different types of sensors and it sends the value to the main server. It also collects the weather data from the weather API. The ultimate decision, whether to water the crop or not is taken by the farmer using mobile application.</i>	<b>8. CHANNELS of BEHAVIOUR</b> <b>CH</b> <b>ONLINE:</b> <i>Providing online assistance to the farmer, in providing knowledge regarding the pH and moisture level of the soil. Online assistance to be provided to the user in using the product</i>	Extract online & offline CH of BE
	<b>4. EMOTIONS: BEFORE / AFTER</b> <b>EM</b> <b>BEFORE:</b> <i>Lack of knowledge in weather forecasting → Random decisions → low yield.</i> <b>AFTER:</b> <i>Data from reliable source → correct decision → high yield</i>		<b>OFFLINE:</b> <i>Awareness camps to be organized to teach the importance and advantages of the automation and IoT in the development of agriculture.</i>	

## 4.Requirement Analysis

### 4.1 Functional Requirement

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	End users can monitor and control their connected farm using IOT applications on their smartphones or tablets
NFR-2	Security	The software keeps the user's information more securely.
NFR-3	Reliability	The smart farm, embedded with IOT systems, could be called a connected farm, which can support a wide range of devices from diverse agricultural device manufactures.
NFR-4	Performance	It is a user-friendly software and have high performance.
NFR-5	Availability	Available for every user, visible for all users and farmer.

### 4.2 Non- Functional Requirements

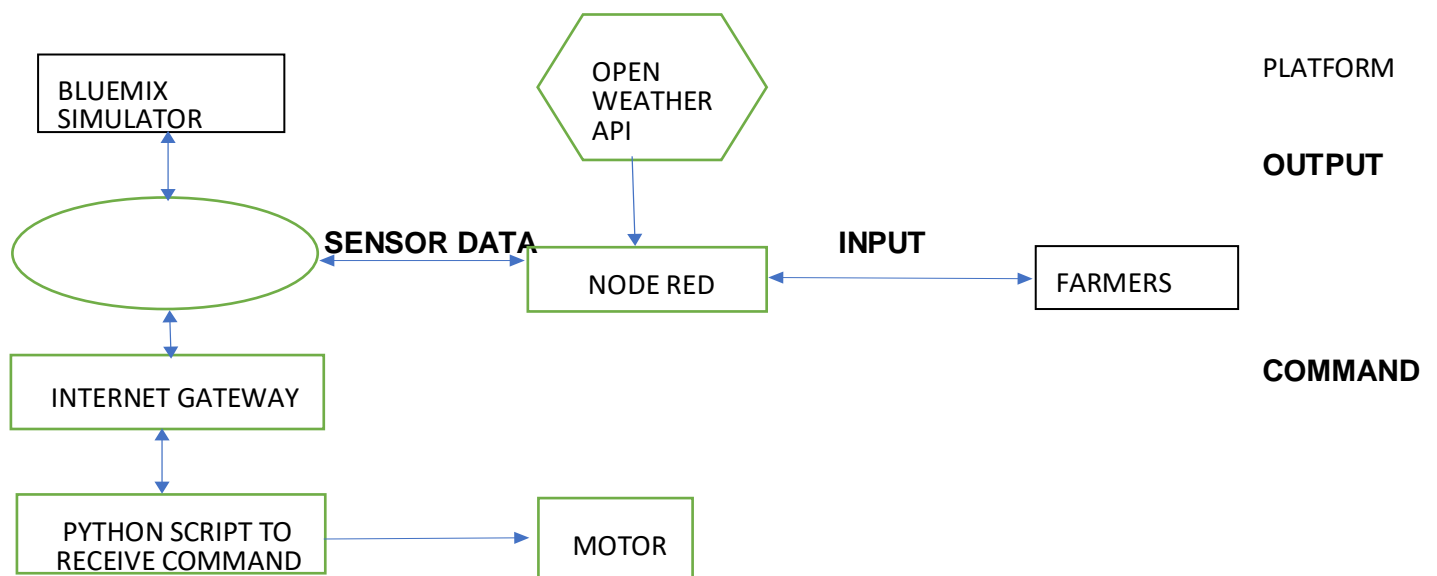
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Application.
FR-2	User Confirmation	Confirmation of registered user.
FR-3	User Profile	Log in Access the Profile
FR-4	Analyse	Data from smart sensors can be analysed for predictive analysis and automated decision making.
FR-5	Recommend	Based on the farming the software recommends the automated irrigation practices.
NFR-6	Scalability	The proposed precision farming structure allows the implementation of a flexible methodology that can be adopted to different types of crops.

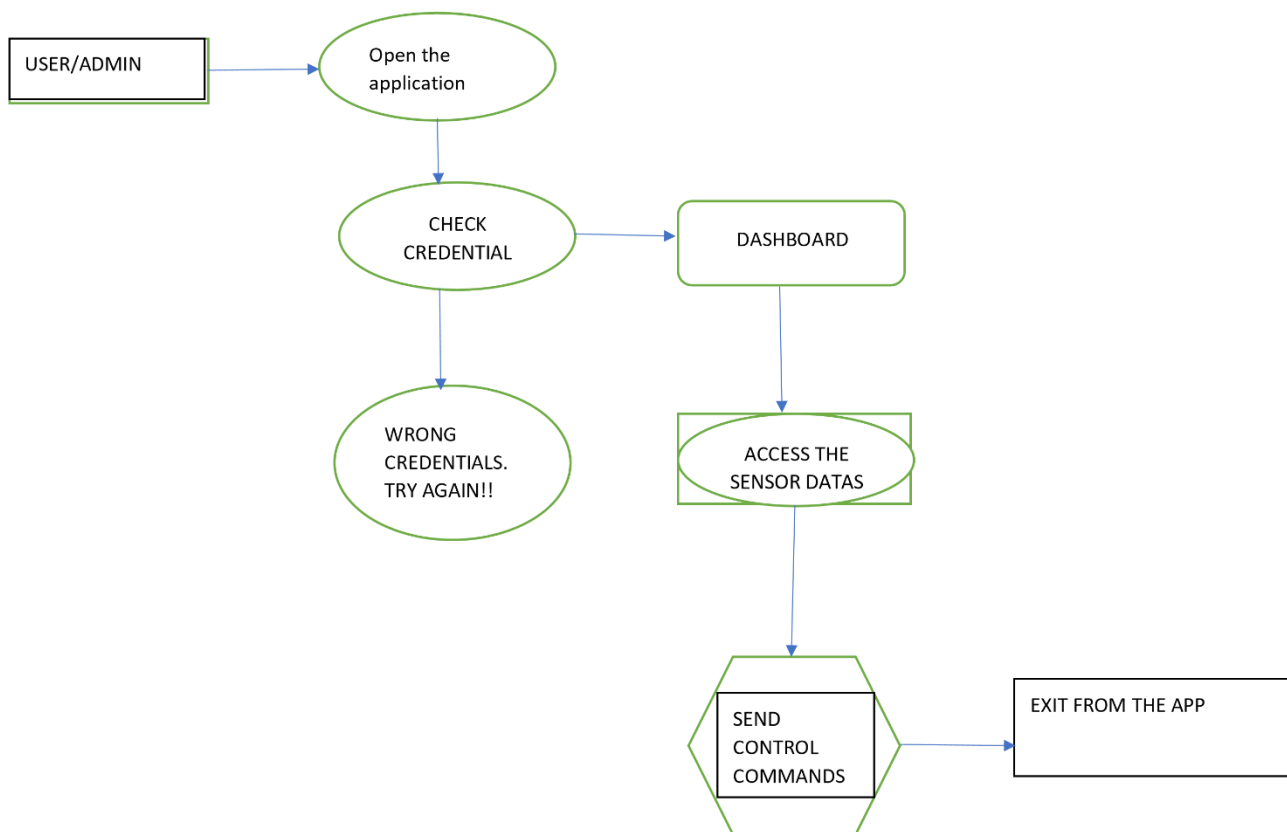
## 5.PROJECT DESIGN

### 5.1 Data Flow Diagrams

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.





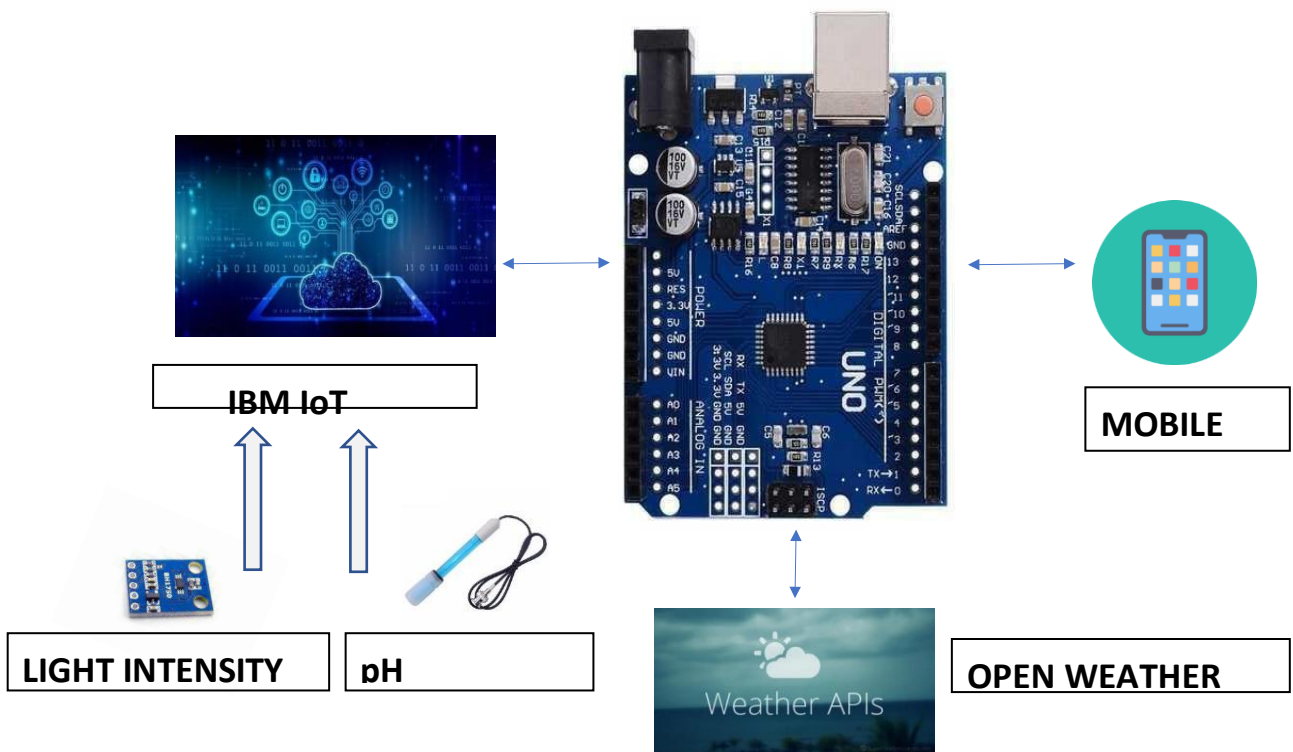
- The different soil parameters temperature, soil moistures and then humidity are sensed using different sensors and obtained value is stored in the ibm cloud.
- Aurdino UNO is used as a processing Unit that process the data obtained from the sensors and whether data from the weather API.
- NODE-RED is used as a programming tool to write the hardware, software and APIs. The MQTT protocol is followed for the communication.
- All the collected data are provided to the user through a mobile application that was developed using the MIT app inventor. The user could make a decision through an app, weather to water the crop or not depending upon the sensor values. By using the app they can remotely operate to the motor switch.

## 5.2 Solution & Technical Architecture

### Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.



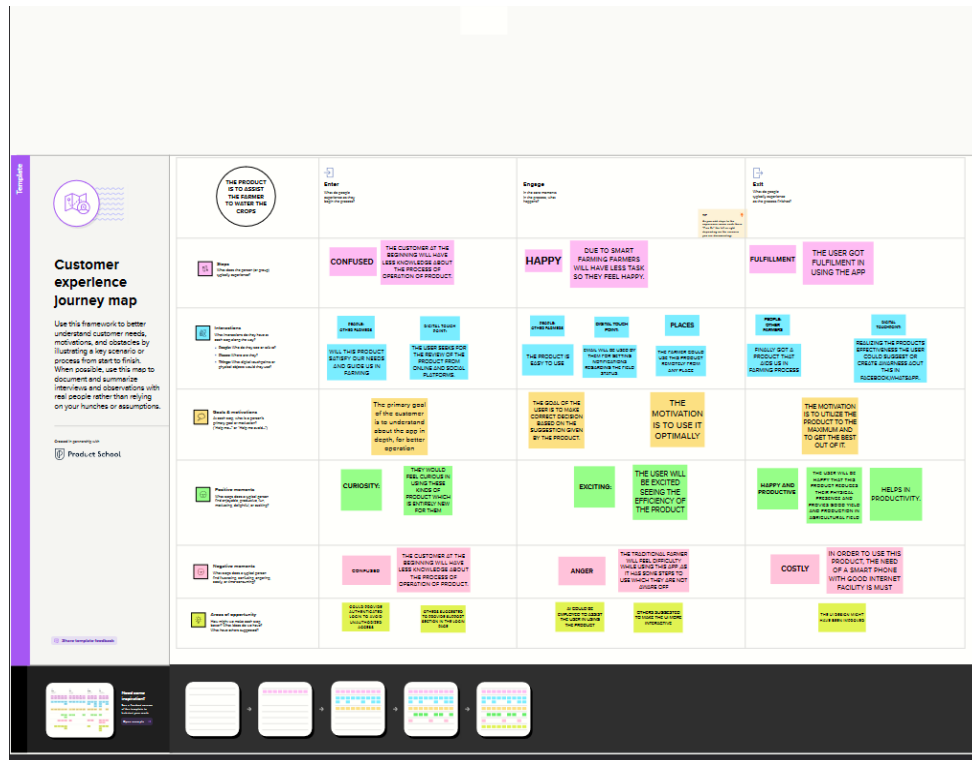
- Aurdino UNO is used as a processing Unit that process the data obtained from the sensors and whether data from the weather API.
- NODE-RED is used as a programming tool to write the hardware, software and APIs. The MQTT protocol is followed for the communication.
- All the collected data are provided to the user through a mobile application that was developed using the MIT app inventor. The user could make a decision through an app, weather to water the crop or not depending upon the sensor values. By using the app they can remotely operate to the motor switch.

**Table-1 : Components & Technologies:**

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson IOT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM Cloud
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
9.	Machine Learning Model	Purpose of Machine Learning Model	Object Recognition Model, etc.
10.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry, Kubernetes, etc.

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Opensource framework
2.	Security Implementations	Sensitive and private data must be protected from their production until the decision-making and storage stages.	e.g. Node-Red, Open weather App API, MIT App Inventor , etc
3.	Scalable Architecture	scalability is a major concern for IoT platforms. It has been shown that different architectural choices of IoT platforms affect system scalability and that automatic real time decision-making is feasible in an environment composed of dozens of thousand.	Technology used
S.No	Characteristics	Description	Technology
4.	Availability	Automatic adjustment of farming equipment made possible by linking information like crops/weather and equipment to auto-adjust temperature, humidity, etc.	Technology used
5.	Performance	The idea of implementing integrated sensors with sensing soil and environmental or ambient parameters in farming will be more efficient for overall monitoring.	Technology used

## 5.3 User Stories

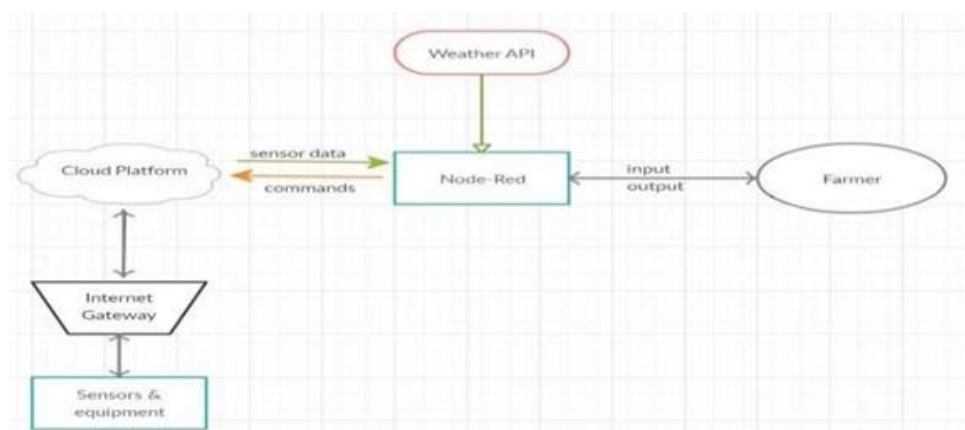


## 6. PROJECT PLANNING & SCHEDULING

## 6.1 Sprint Delivery planning & Estimation

## SPRINT DELIVERY OVERVIEW :

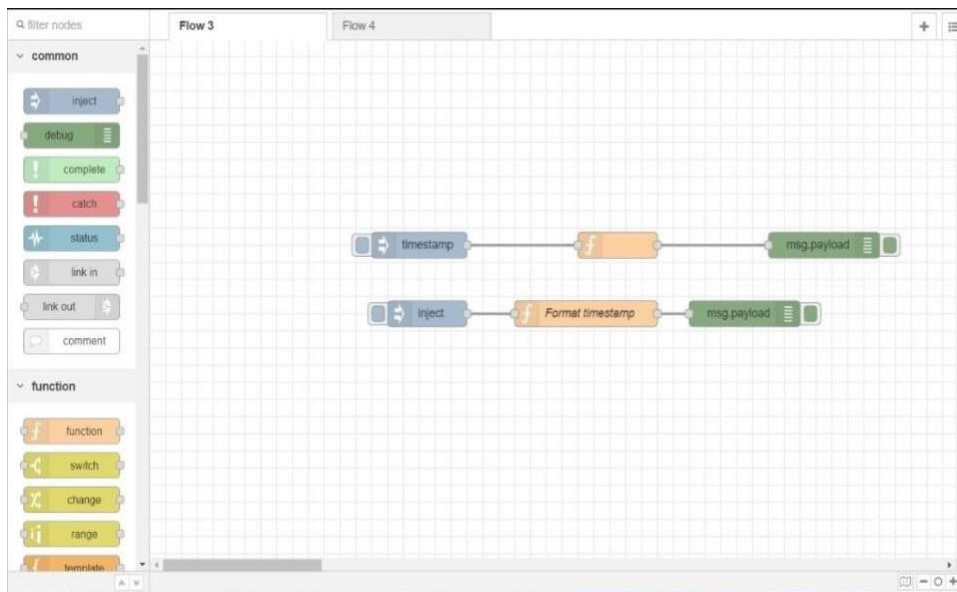
In order to implement the solution , the following approach as shown in the block diagram is used



## 1.Required Software Installation

### 1.1 A Node-Red

Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things. NodeRED provides a web browser-based flow editor, which can be used to create JavaScript functions.



Installation :

- First install npm/node.js
- Open cmd prompt
- Type => npm install node-red To run the application :
- Open cmd prompt
- Type=> node-red
- Then open <http://localhost:1880/> in browser

Installation of IBM IoT and Dashboard nodes for Node-Red

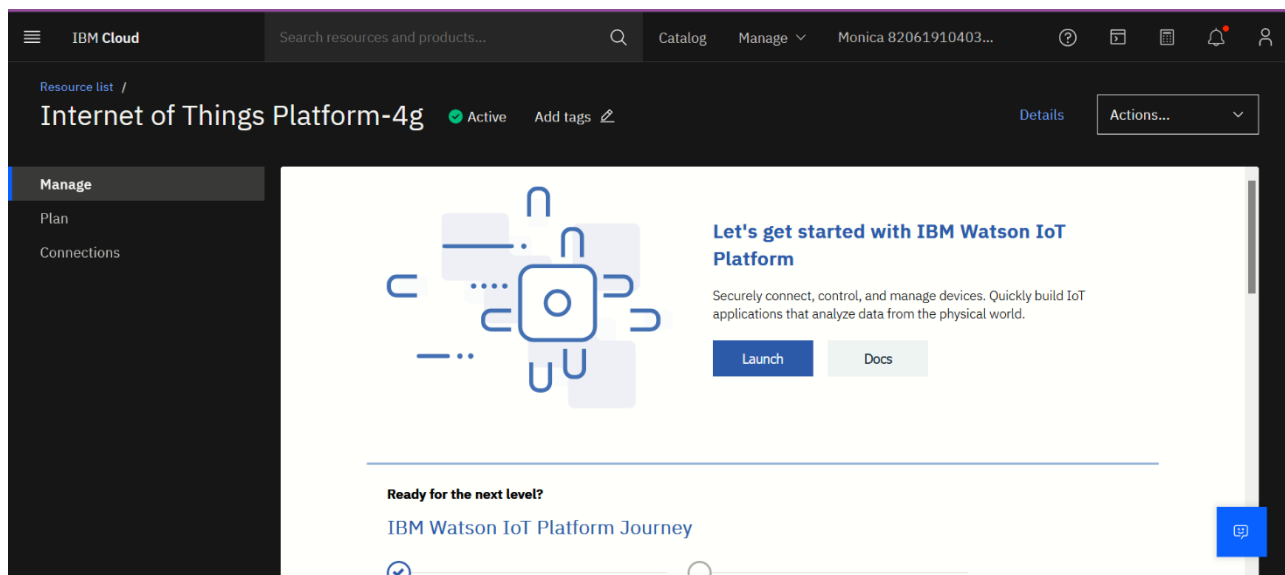
In order to connect to IBM Watson IoT platform and create the Web App UI these nodes are required



1. IBM IoT node
2. Dashboard node

## 1.2.B IBM Watson IoT Platform

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization and data storage. IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices.

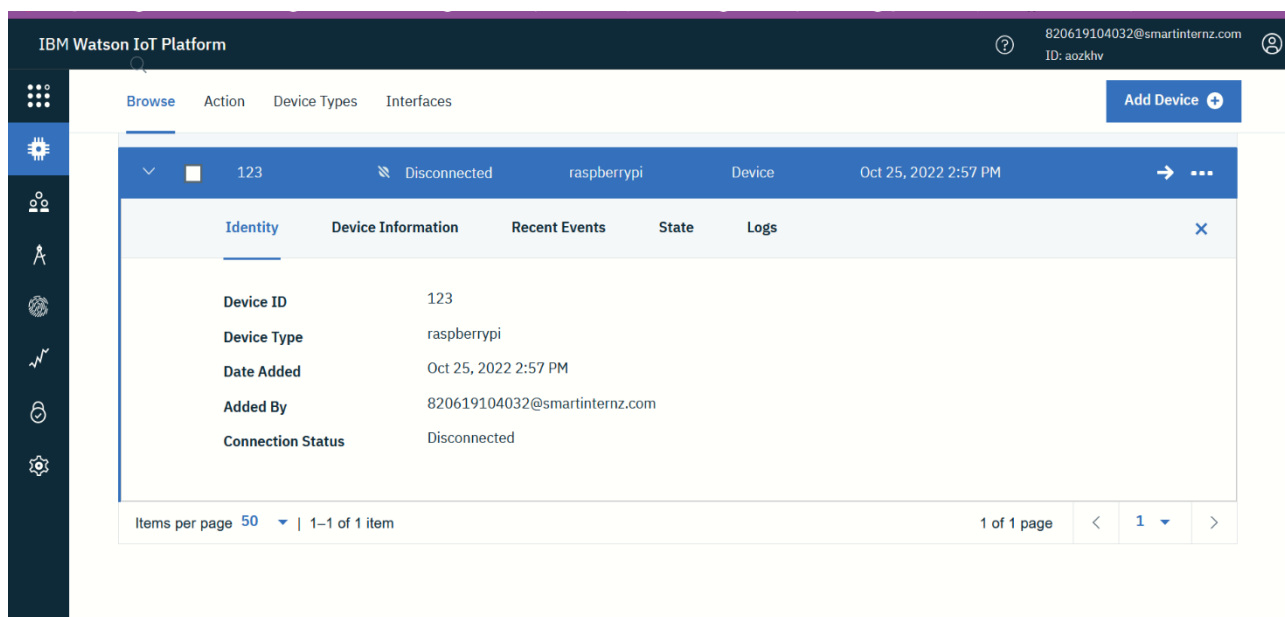


Steps to configure:

- Create an account in IBM cloud using your email ID
- Create IBM Watson Platform in services in your IBM cloud account
- Launch the IBM Watson IoT Platform
- Create a new device
- Give credentials like device type, device ID, Auth. Token • Create API key and store API key and token elsewhere.

### Steps to configure:

- Create an account in IBM cloud using your email ID
- Create IBM Watson Platform in services in your IBM cloud account
- Launch the IBM Watson IoT Platform
- Create a new device
- Give credentials like device type, device ID, Auth. Token • Create API key and store API key and token elsewhere.



## 1.2.C Python IDE

- Install Python3 compiler
- Install any python IDE to execute python scripts, in my case I used Spyder to execute the code.

```
Python 3.7 (64-bit)
Python 3.7.5 (tags/v3.7.5:5c82a94eb, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

## 1.3 IoT Simulator

- In our project in the place of sensors we are going to use IoT sensor simulator which give random readings to the connected cloud.
- The link to simulator: <https://watson-iot-sensor-simulator.mybluemix.net/>
- We need to give the credentials of the created device in IBM Watson IoT Platform to connect cloud to simulator.

## 1.4 Open Weather API

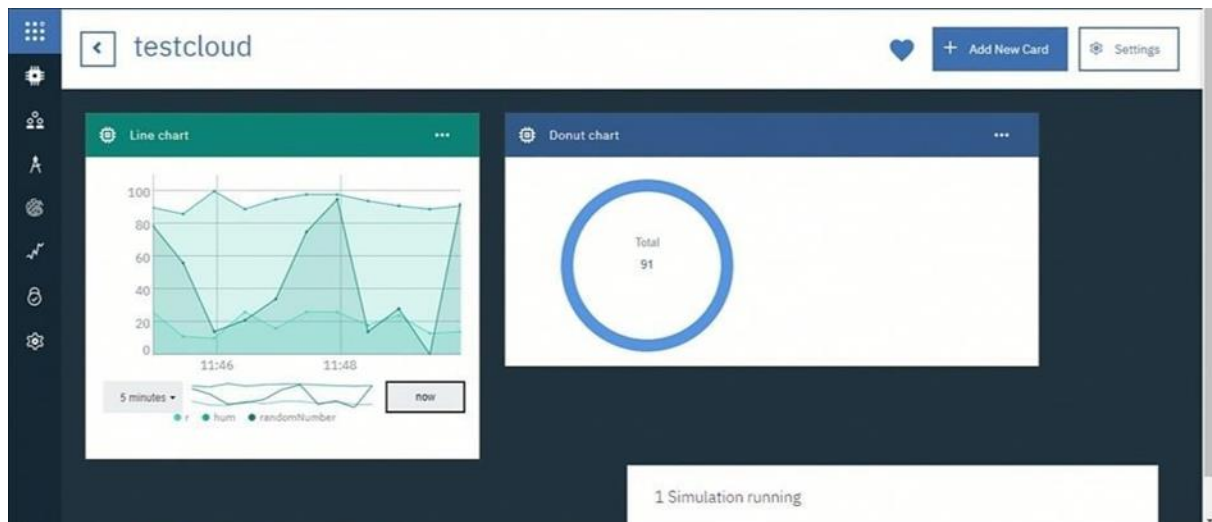
- OpenWeatherMap is an online service that provides weather data. It provides current weather data, forecasts and historical data to more than 2 million customer.
  - Website link: <https://openweathermap.org/guide>
  - Steps to configure:
    - o Create account in OpenWeather
    - o Find the name of your city by searching
    - o Create API key to your account
    - o Replace "city name" and "your api key" with your city and API key in below red text
- `api.openweathermap.org/data/2.5/weather?q={city name}&appid={your api key}` Link I used in my project:  
`http://api.openweathermap.org/data/2.5/weather?q=Gudur,in&appid=62354068e45f41ffa6a5b164714145fe`

## 2. Building Project

### 2.1 Connecting IoT Simulator to IBM Watson IoT Platform

Open link provided in above section 4.3

Give the credentials of your device in IBM Watson IoT Platform



Identity	Device Information	Recent Events	State	Logs
The recent events listed show the live stream of data that is coming and going from this device.				
Event	Value	Format	Last Received	
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu..."	json	a few seconds ago	
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu..."	json	a few seconds ago	
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu..."	json	a few seconds ago	
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu..."	json	a few seconds ago	
iotsensor	{"d":{"name":"qwerty123","temperature":17,"hu..."	json	a few seconds ago	

## 2.2 Configuration of Node-Red to collect IBM cloud data

The node IBM IoT App In is added to Node-Red workflow. Then the appropriate device credentials obtained earlier are entered into the node to connect and fetch device telemetry to Node-Red. Once it is connected, Node-Red receives data from the device.

Display the data using debug node for verification.

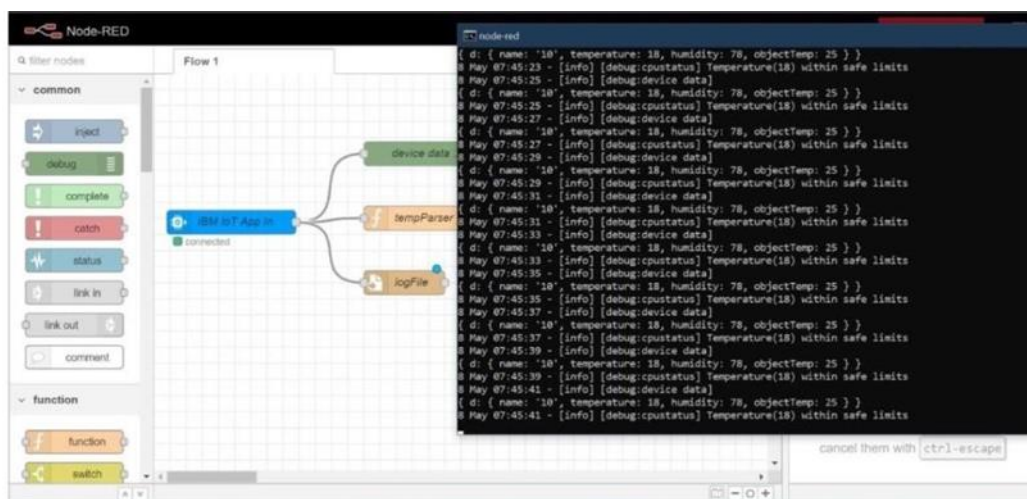
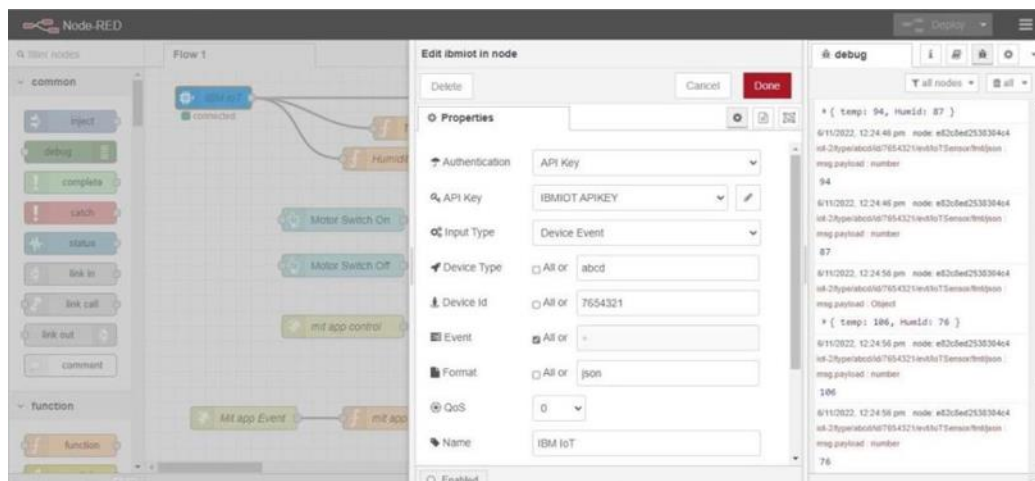
Connect function node and write the Java script code to get each reading separately.

The Java script code for the function node is:

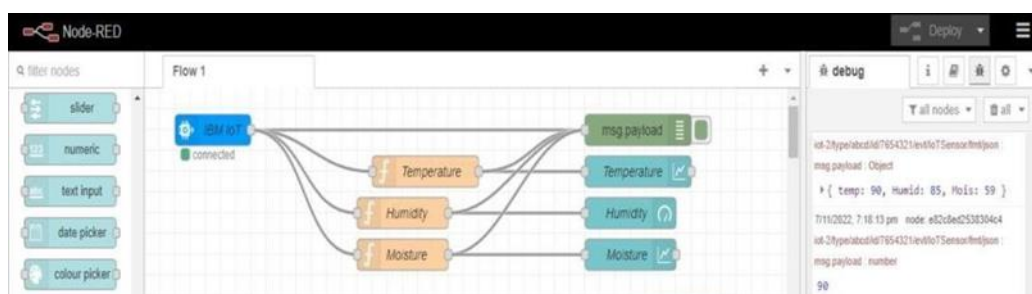
```
msg.payload=msg.payload.d.temperature return msg;
```

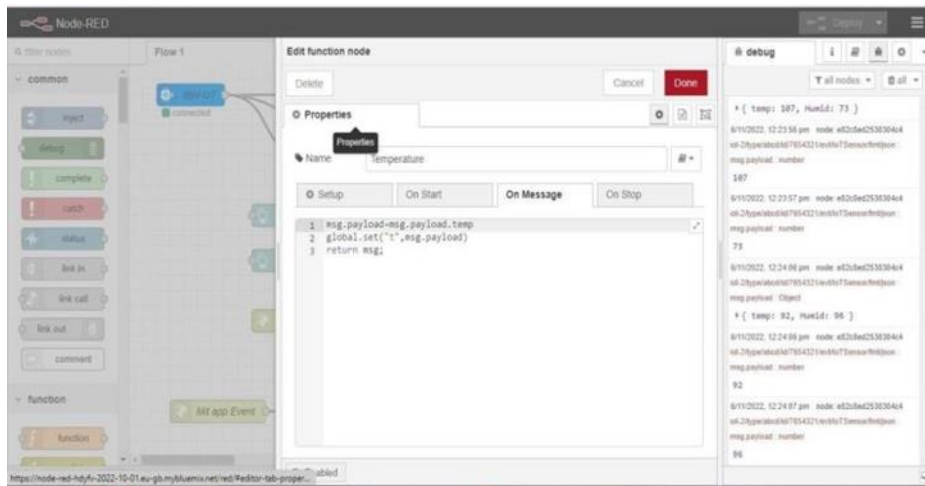
Finally, connect Gauge nodes from dashboard to see the data in UI.

Nodes connected in following manner to get each reading separately



Data received from the cloud in Node-RED console





### 3. Configuration of Node-Red to collect data from OpenWeather

The Node-Red also receive data from the OpenWeather API by HTTP GET request. An inject trigger is added to perform HTTP request for every certain interval.

HTTP request node is configured with URL we saved before in section 4.4

The data we receive from OpenWeather after request is in below JSON format:

```
{
  "coord": {
    "lon": 79.85,
    "lat": 14.13
  },
  "weather": [
    {
      "id": 803,
      "main": "Clouds",
      "description": "broken clouds",
      "icon": "04n"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 307.59,
    "feels_like": 305.5,
    "temp_min": 30
  },
  "temp_max": 307.59,
  "pressure": 1002,
  "humidity": 35,
  "sea_level": 1002,
  "grnd_level": 1000,
  "wind": {
    "speed": 6.23,
    "deg": 170
  },
  "clouds": {
    "all": 68
  },
  "dt": 1589991979,
  "sys": {
    "country": "IN",
    "sunrise": 15893553,
    "sunset": 1589979720
  },
  "timezone": 19800,
  "id": 1270791,
  "name": "Gūdūr",
  "cod": 200
}
```

7.

```
59, "temp_max": 307.59, "pressure": 1002, "humidity": 35, "sea_level": 1002, "grnd_level": 1000, "wind": {
```

```
{
```

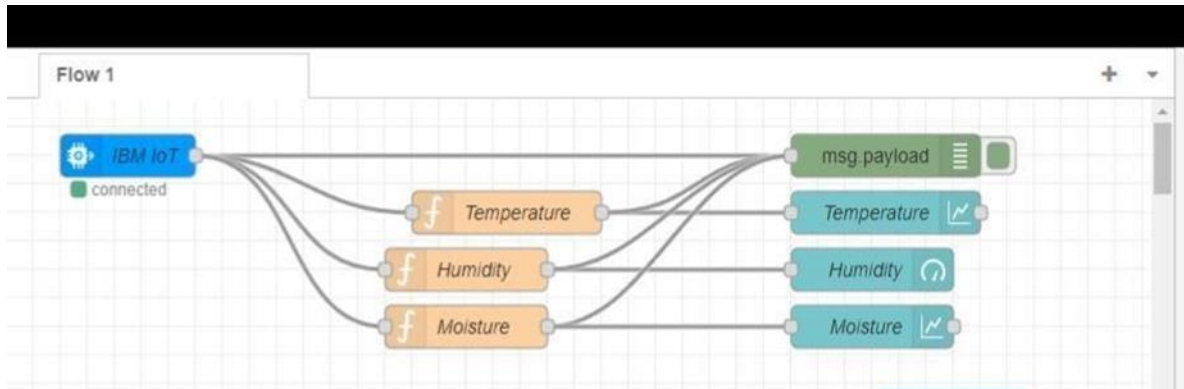
```
"speed": 6.23, "deg": 170, "clouds": { "all": 68 }, "dt": 1589991979, "sys": { "country": "IN", "sunrise": 15893553, "sunset": 1589979720 }, "timezone": 19800, "id": 1270791, "name": "Gūdūr", "cod": 200 }
```

In order to parse the JSON string we use Java script functions and get each parameters

```
var temperature = msg.payload.main.temp;
```

```
temperature = temperature-273.15; return {payload : temperature.toFixed(2)};
```

In the above Java script code we take temperature parameter into a new variable and convert it from kelvin to Celsius



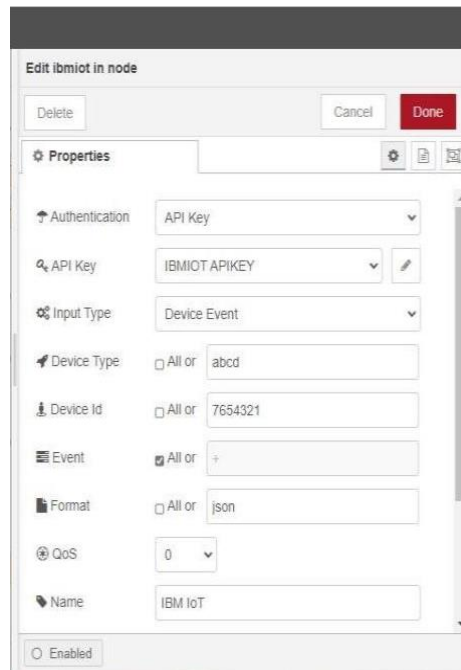
Then we add Gauge and text nodes to represent data visually in UI

The above image has the program flow for receiving data from OpenWeather

The above two images contain http request and function node data that needs to be filled.

## 2.4 Configuration of Node-Red to send commands to IBM cloud

ibmiot 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.



Here we add three buttons in UI which each sends a number 0,1 and 2.

0 -> for motor off

1 -> for motor on

2 -> for running motor continuously 30 minutes

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

The Java script code for the analyser is:

```
if(msg.payload===1) msg.payload={"command":"ON"}; else  
if(msg.payload===0)  
msg.payload={"command":"OFF"}; else
```



```
msg.payload={"command":"runfor30minutes"};
```

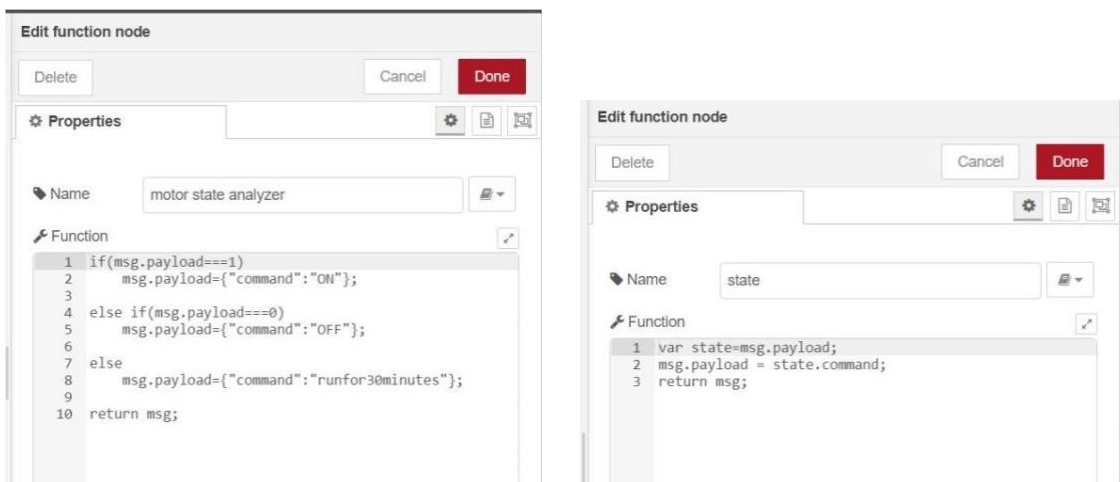
```
return msg;
```

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:

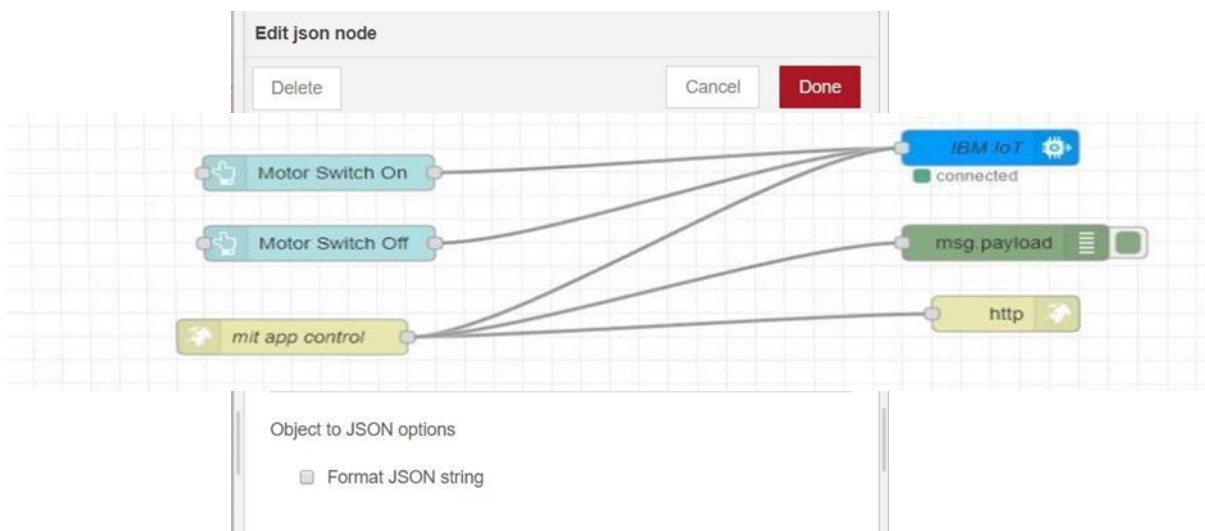
```
var state=msg.payload;
```

```
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.

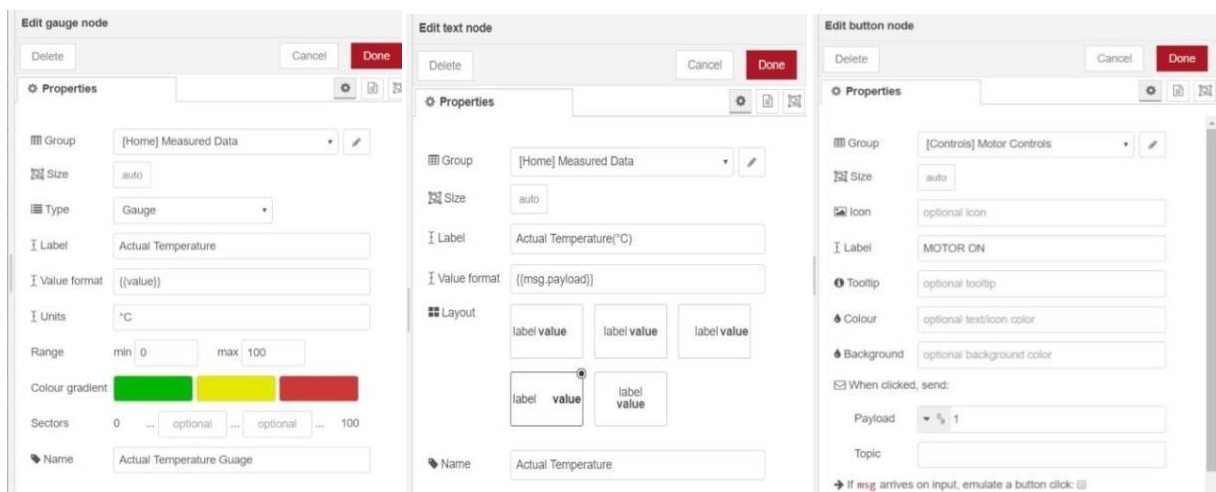


Edit JSON node needs to be configured like this

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

## 2.5 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



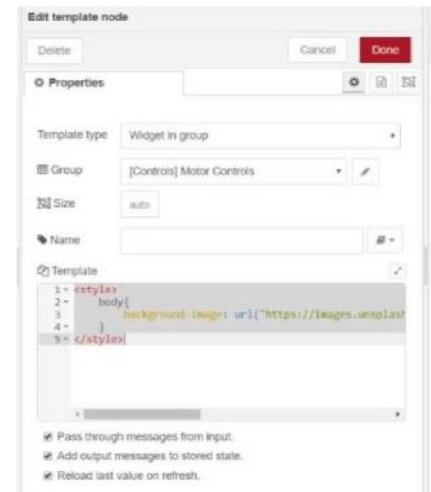
## Adding Background image to the UI

To add the background image we are going to add template node and configure it with below HTML code

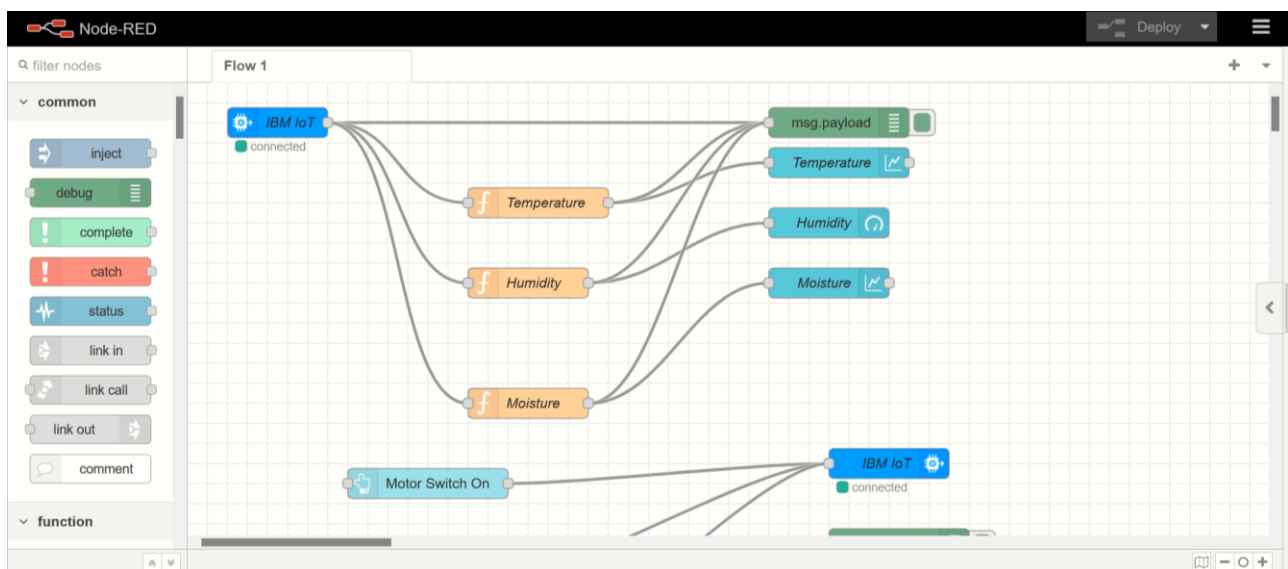
```
<style> body{ background-image: url("https://images.
    unsplash.com/photo-1563514227147-
    6d2ff665 a6a0?ixlib=rb-
    1.2.1&auto=format&fit=crop&w
    =1951&q=80");
}
```

We add URL of image that need to be displayed

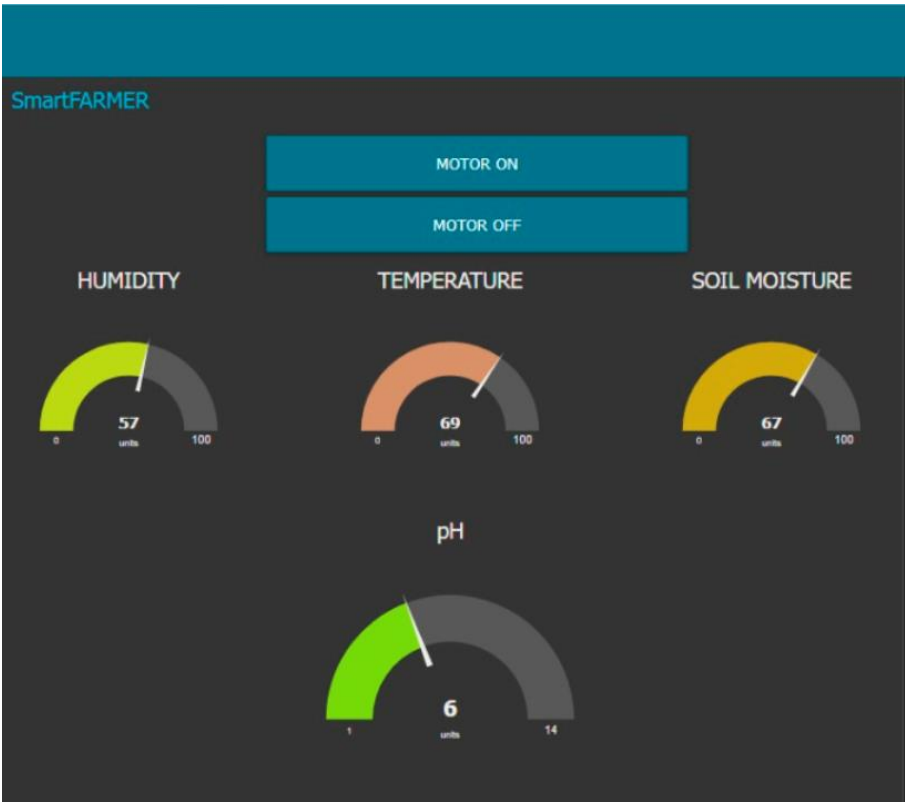
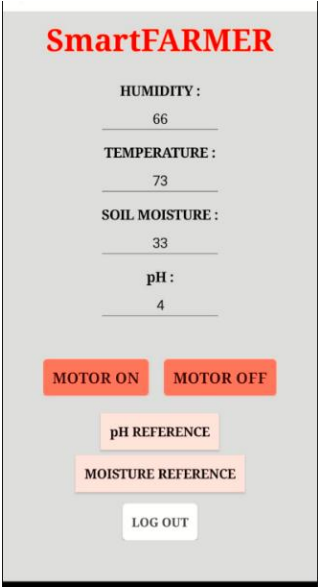
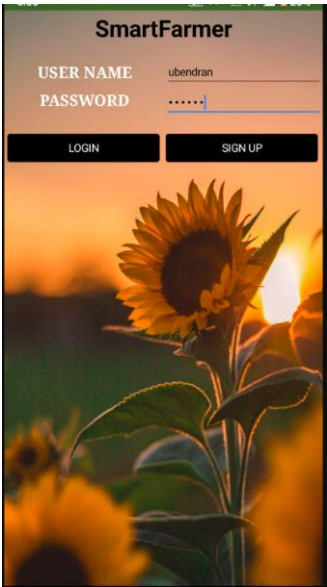
Configuration is shown in the beside image



### Complete Program Flow



Mobile app



```
===== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py =====
2022-11-07 20:01:24,074 ibmiotf.device.Client INFO Connected successfully: d:157uf3:abcd:7654321
Published Moisture = 90 deg C Temperature = 96 C Humidity = 76 % to IBM Watson
Published Moisture = 102 deg C Temperature = 110 C Humidity = 68 % to IBM Watson
Published Moisture = 45 deg C Temperature = 99 C Humidity = 100 % to IBM Watson
Command received: motoron
motor is on
Published Moisture = 77 deg C Temperature = 91 C Humidity = 85 % to IBM Watson
Published Moisture = 73 deg C Temperature = 94 C Humidity = 86 % to IBM Watson
Command received: motoroff
motor is off
Published Moisture = 101 deg C Temperature = 104 C Humidity = 87 % to IBM Watson
```

## 6.2 Sprint Delivery Planning Schedule

### Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Interfacing Sensors and Motor Pump and IBM cloud	USN-1	Develop a python code to Interface Sensors, Motor Pump and IBM cloud.	20	High	VIGNESHWARI.B
Sprint-2	Node-Red	USN-2	Develop a web Application Using a NodeRed.	20	High	JAYASHARIMILA.K
Sprint-3	Mobile Application	USN-3	Develop a mobile Application using MIT-App Inventor.	20	High	ANUGRAHAA.R
Sprint-4	Integration & Testing	USN-4	Integrating Python Script, Web application & Mobile App	20	Medium	MONICA U.M

## Project Tracker, Velocity & Burndown Chart: (4 Marks)

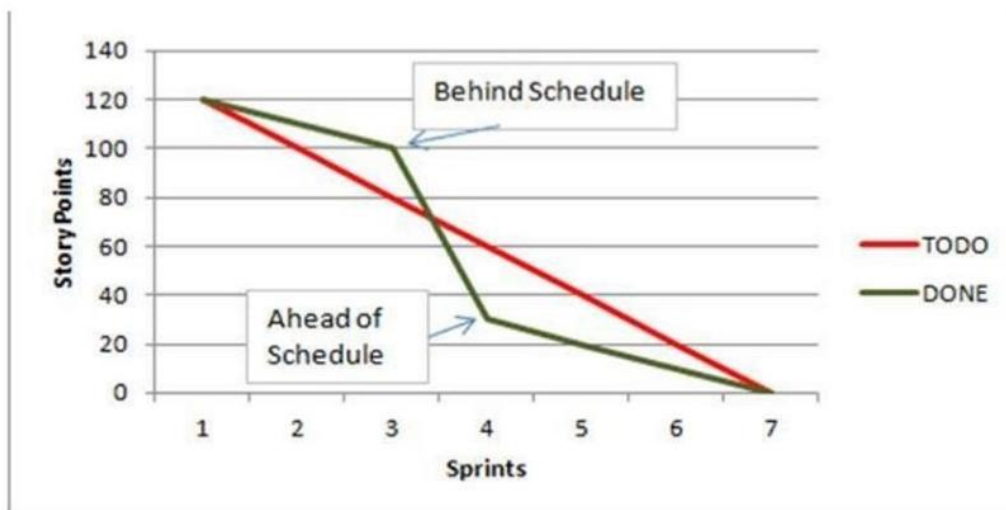
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	11 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	17 Nov 2022

### Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{\text{velocity}} = \frac{20}{10} = 2$$

Burndown Chart:



## 7.Coding & Solution

### 7.1 Feature - 1

Receiving commands from IBM cloud using Python program

This is the Python code to receive commands from cloud to any device like Raspberry Pi in the farm

```
import time import sys
```

```
import ibmiotf.application
```

```
import ibmiotf.device
```

```
import random
```

```
#IBM Watson Device Credentials organization = "re4wy2"
```

```
#replace with org ID deviceType = "abcd"
```

```
deviceId = "12"
```

```
authMethod = "token"
```

```
authToken = "12345678"
```

```
#Receives Command fro Node-RED
```

```
def myCommandCallback(cmd):
```

```
#gets data from ibm cloud to python print("Command receive: %s" %  
cmd.data['command'])
```

```

status=cmd.dataa['command']

if status=="motoron":

    print("motor is on")

elif status=="motoroff":

    print ("motor is off")

else :

    print("please send proper command")

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()

#Connect and send a datapoint deviceCli.connect()

while True:

    #Get sensor data from DHT11

    temp=random.randint(0,100) humidity=random.randint(0,100)
    randomNumber=random.randint(0,100)

```



```

data = {'temp':temp, 'humidity':humidity, 'randomNumber':randomNumber}

#print data
def myOnPublishCallback():

print("Published Temperature = %s" % temp, "Humidity = %s" % humidity,

"soilmoisture = %s" % randomNumber, "to IBM Watson")
success =
deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish =

myOnPublishCallback())

if not success:

print("NOT CONNECTED TO IoTF")

time.sleep(5)

deviceCli.commandCallback = myCommandCallback

#disconnect the device and application from the cloud
deviceCli.disconnect()

```



```

C:\WINDOWS\system32\cmd.exe
Published Temperature = 106 C Humidity = 88 % to IBM Watson
Published Temperature = 106 C Humidity = 99 % to IBM Watson
Published Temperature = 102 C Humidity = 65 % to IBM Watson
Published Temperature = 96 C Humidity = 74 % to IBM Watson
Published Temperature = 109 C Humidity = 64 % to IBM Watson
Published Temperature = 105 C Humidity = 86 % to IBM Watson
Published Temperature = 105 C Humidity = 83 % to IBM Watson
Published Temperature = 102 C Humidity = 86 % to IBM Watson
Published Temperature = 103 C Humidity = 68 % to IBM Watson
Published Temperature = 106 C Humidity = 83 % to IBM Watson
Published Temperature = 101 C Humidity = 85 % to IBM Watson
Published Temperature = 106 C Humidity = 84 % to IBM Watson
Published Temperature = 95 C Humidity = 74 % to IBM Watson
Published Temperature = 107 C Humidity = 73 % to IBM Watson
Published Temperature = 92 C Humidity = 96 % to IBM Watson
Published Temperature = 93 C Humidity = 82 % to IBM Watson
Published Temperature = 98 C Humidity = 88 % to IBM Watson
Published Temperature = 107 C Humidity = 71 % to IBM Watson
Published Temperature = 94 C Humidity = 87 % to IBM Watson
Published Temperature = 106 C Humidity = 76 % to IBM Watson
Published Temperature = 98 C Humidity = 81 % to IBM Watson
Published Temperature = 103 C Humidity = 95 % to IBM Watson
Published Temperature = 92 C Humidity = 66 % to IBM Watson
Published Temperature = 99 C Humidity = 76 % to IBM Watson
Published Temperature = 93 C Humidity = 68 % to IBM Watson
command received: motoron
Please send proper command
Published Temperature = 96 C Humidity = 62 % to IBM Watson
Published Temperature = 103 C Humidity = 68 % to IBM Watson

```

[illegible]

## 8.2 User Acceptance Testing

### 1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

Increasing control over production leads to **better cost management and waste reduction**. The ability to trace anomalies in crop growth or livestock health, for instance, helps eliminate the risk of losing yields. Additionally, automation boosts efficiency. Smart farming **reduces the ecological footprint of farming**. Minimized or site-specific application of inputs, such as fertilizers and pesticides, in precision agriculture systems will mitigate leaching problems as well as the emission of greenhouse gases.

### 2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

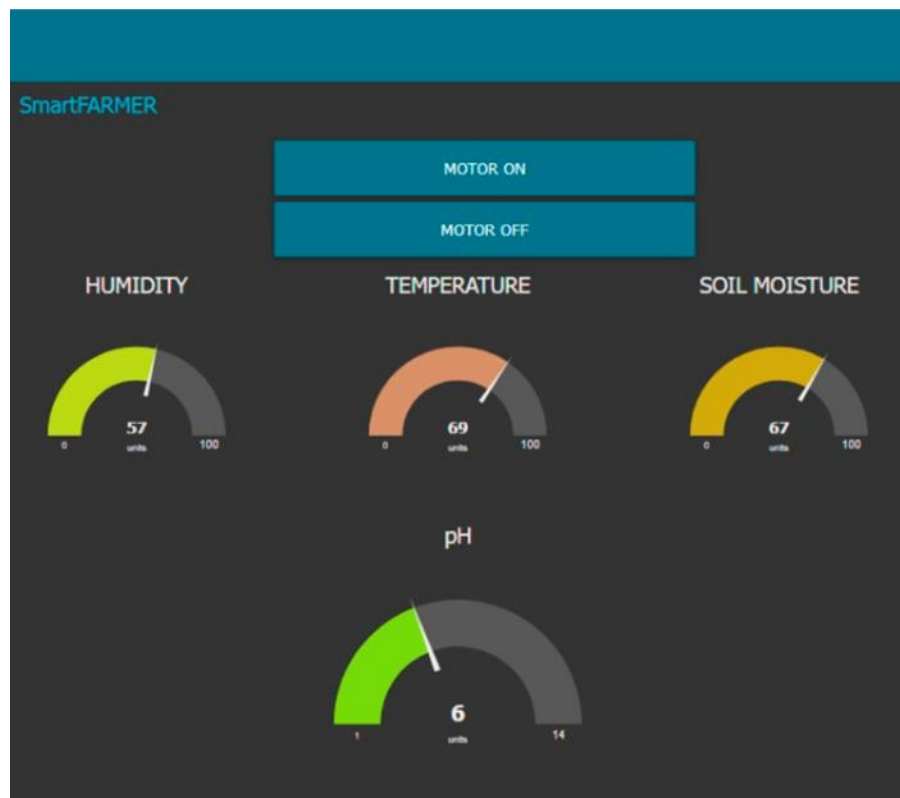
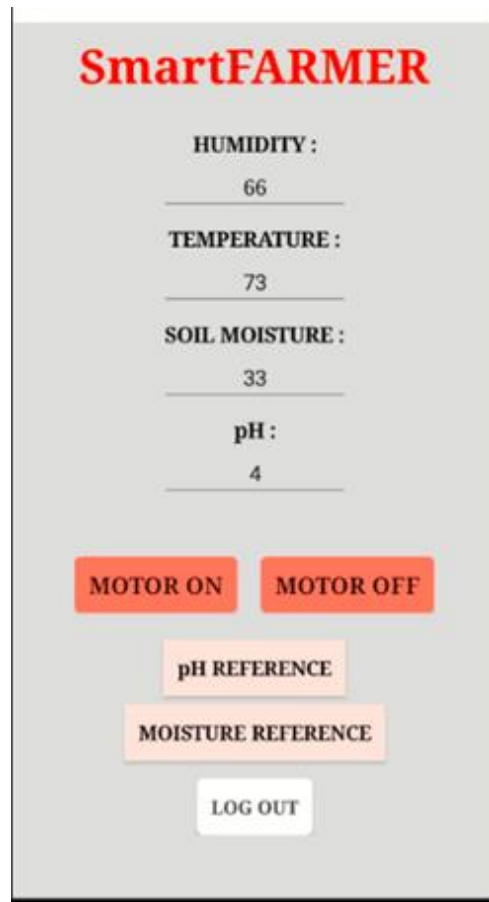
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	8	3	2	2	16
Duplicate	1	0	2	0	3
External	2	3	0	1	6
Fixed	9	2	3	17	31
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	1	4	1	1	7
Totals	21	12	9	22	66

### 3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	0	5
Client Application	30	0	0	30
Security	2	0	0	2
Outsource Shipping	2	0	0	2
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	1	0	0	1

## 9.Results :



```
===== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py =====
2022-11-07 20:01:24,074   ibmiotf.device.Client      INFO    Connected successfully: d:157uf3:abcd:7654321
Published Moisture = 90 deg C Temperature = 96 C Humidity = 76 % to IBM Watson
Published Moisture = 102 deg C Temperature = 110 C Humidity = 68 % to IBM Watson
Published Moisture = 45 deg C Temperature = 99 C Humidity = 100 % to IBM Watson
Command received: motoron
motor is on
Published Moisture = 77 deg C Temperature = 91 C Humidity = 85 % to IBM Watson
Published Moisture = 73 deg C Temperature = 94 C Humidity = 86 % to IBM Watson
Command received: motoroff
motor is off
Published Moisture = 101 deg C Temperature = 104 C Humidity = 87 % to IBM Watson
```

## 10. Advantages & Disadvantages

### Advantages:

- Farms can be monitored and controlled remotely.
- Increase in convenience to farmers.
- Less labour cost.
- Better standards of living.

### Disadvantages:

- Lack of internet/connectivity issues.
- Added cost of internet and internet gateway infrastructure.
- Farmers wanted to adapt the use of WebApp.

## 11 .Conclusion

Thus the objective of the project to implement an IoT system in order to help farmers to control and monitor their farms has been implemented successfully

## 12. Future Scope

Through collecting data from sensors using IoT devices, you will learn about the real-time state of your crops. The future of IoT in agriculture **allows predictive analytics to help you make better harvesting decisions**. Smart farming refers to **managing farms using modern Information and communication technologies to increase the quantity and quality of products while optimizing the human labor required**. Among the technologies available for present-day farmers are: Sensors: soil, water, light, humidity, temperature management.

IOT TECHNOLOGIES IN AGRICULTURE. IoT smart agriculture products are designed to **help monitor crop fields using sensors and by automating irrigation systems**. As a result, farmers and associated brands can easily monitor the field conditions from anywhere without any hassle.

## 10.Appendix :

Links:

IBM cloud reference: <https://cloud.ibm.com/>

Github link : <https://github.com/IBM-EPBL/IBM-Project-18992-1659691969>

IOT\_Watson simulator :

<https://aozkhv.internetofthings.ibmcloud.com/dashboard/devices/browse>

Node-Red :<https://node-red-hdyfv-2022-10-01.eu-gb.mybluemix.net/red/#flow/c7ddb1462b8a000c>

MIT App Inventor : <http://ai2.appinventor.mit.edu/#5147143820935168>