# PROJECT REPORT

## IBM-IoT ENABLED SMART FARMING APPLICATION

## SMART FARMER-SMART FARMING APPLICATION USING INTERNET OF THINGS

## TABLE OF CONTENT


| S.NO | PG. NO. | TITLE |
|---|---|---|
| 1 | 3 | INTRODUCTION |
| 1.1 | | PROJECT |
| 1.2 | | OVERVIE |
| | | W |
| | | PURPOSE |
| 2 | 4 | LITERATURE SURVEY |
| 2.1 | | EXISTING PROBLEM |
| 2.2 | | REFERENCES |
| 2.3 | | PROBLEM STATEMENT DEFINITION |
| 3 | 6 | IDEATION AND PROPOSED SOLUTION |
| 3.1 | | EMPATHY MAP CANVAS |
| 3.2 | | IDEATION AND BRAINSTORMING |
| 3.3 | | PROPOSED SOLUTION |
| 3.4 | | PROBLEM SOLUTION FIT |
| 4 | 10 | REQUIREMENT ANALYSIS |
| 4.1 | | FUNCTIONAL REQUIREMENT |
| 4.2 | | NON-FUNCTIONAL REQUIREMENT |
| 5 | 11 | PROJECT DESIGN |
| 5.1 | | DATA FLOW DIAGRAMS |
| 5.2 | | SOLUTION AND TECHNICAL ARCHITECTURE |
| 5.3 | | USER STORIES |
| 6 | 15 | PROJECT PLANNING AND SCHEDULING |
| 6.1 | | SPRINT PLANNING AND ESTIMATION |
| 6.2 | | SPRINT DELIVERY SCHEDULE |
| 7 | 17 | CODING AND SOLUTIONING |
| 7.1 | | FEATURE 1 |
| 7.2 | | FEATURE 2 |

| | | |
|---|---|---|
| 8.1<br>8.2 | | TEST CASES<br>USER ACCEPTANCE TESTING |
| 9<br>9.1 | 23 | RESULTS<br>PERFORMANCE METRICS |
| 10 | 27 | ADVANTAGES AND DISADVANTAGES |
| 11 | 27 | CONCLUSION |
| 12 | 29 | FUTURE SCOPE |
| 13 | 30 | APPENDIX<br>SOURCE CODE<br>GITHUB AND PROJECT DEMO LINK |

# 1.Introduction:

## 1.1 Project overview:

Internet of Things Smart technology enables new digital agriculture. Today technology has become a necessity to meet current challenges and several sectors are using the latest technologies to automate their tasks. Advanced agriculture, based on Internet of Things technologies, is envisioned to enable producers and farmers to reduce waste and improve productivity by optimizing the usage of fertilizers to boost the efficiency of plants. It gives better control to the farmers for their livestock, growing crops, cutting costs, and resources.

## 1.2 Purpose:

We have tried to focus on different scientific applications which could be put together in the agricultural field for better accuracy with better productivity using less manpower. Moreover, we include a method for monitoring the agricultural fields from any remote location and assessing the basic condition of the field. This is the project from the motivation of the farmers working in the farmlands who are solely dependent on the rains and bore wells for irrigation of their land. In recent times, the farmers have been using irrigation techniques through manual control in which the farmers irrigate the land at regular intervals by turning the water-pump ON/OFF when required.

# 2.literature survey:

## 2.1 Existing Problem:

Agriculture is the foundation of our Nation. In the past, agriculturists used to figure out the ripeness of soil and influenced presumptions to develop which kind of product. They didn't think about the dampness, level of water and especially climate conditions which are more horrifying to an agriculturist. They utilize pesticides in view of a few suspicions which lead a genuine impact to the yield if the supposition isn't right. Profitability relies upon the last phase of the harvest on which agriculturists depend.

## 2.2 References:

[1] S. S. Gill, I. Chana, C. Science, and R. Buyya, “IoT Based Agriculture as a Cloud and Big Data Service: The Beginning of Digital India,” vol. 29, no. 4, pp. 1–23, 2017.

[2] R. K. Kodali and A. Sahu, “An IoT based Weather Information Prototype Using WeMos,” no. 1, pp. 612–616, 2016.

[3] I. Mat, M. Rawidean, M. Kassim, A. N. Harun, I. M. Yusoff, and M. Berhad, “Smart Agriculture Using Internet of Things,” 2018 IEEE Conf. Open Syst., pp. 54–59, 2018.

[4] S. R. Prathibha, A. Hongal, and M. P. Jyothi, “IOT BASED MONITORING SYSTEM IN SMART AGRICULTURE,” pp. 5–8, 2017.

[5] Y. Bo and H. Wang, “The Application of Cloud Computing and The Internet of Things in Agriculture and Forestry,” pp. 168–172, 2011.

[6] Tien Cao Huang and Can Nguyen Duy, “Environment Monitoring for Agricultural Application Based on Wireless Sensor Network.”, April 16-17.

[7] M. K. Gayatri, “Providing Smart Agricultural Solutions to Farmers for better yielding using IoT,” no. Tiar, pp. 40–43, 2015.

[8] S. V. Mukherji, R. Sinha, S. Basak, and S. P. Kar, “Smart Agriculture using Internet of Things and MQTT Protocol,” 2019 Int. Co

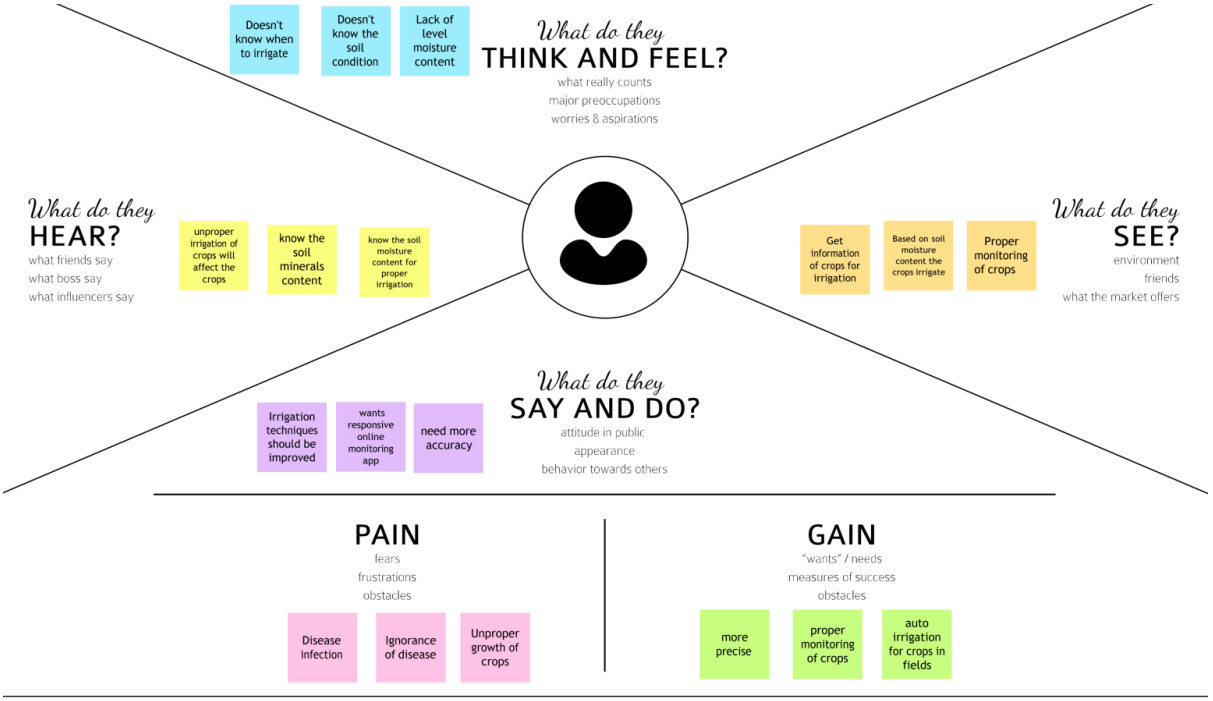
## 2.3 Problem Statement Definition:

The researcher is supposed to implement an "IoT Based Smart Farming System" with various sensors, which will help to collect the data and analyze it. The proposed system collects information about different agricultural parameters (temperature, humidity, moisture) using an IoT sensor. These values collected are then sent over the mobile via SMS. Farmers can view all the parameters required for a smart farming system through the webpage.

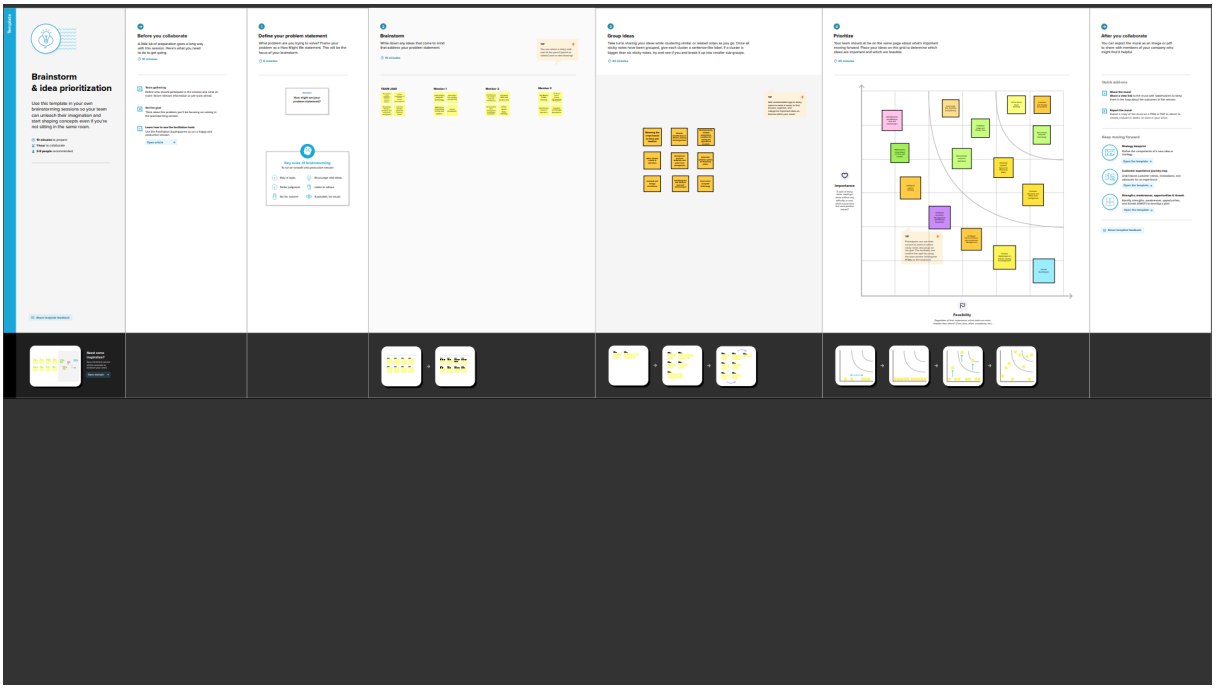
# 3. IDEATION AND PROPOSED SOLUTION:

## 3.1 Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes.



# 3.2 Ideation & Brainstorming:



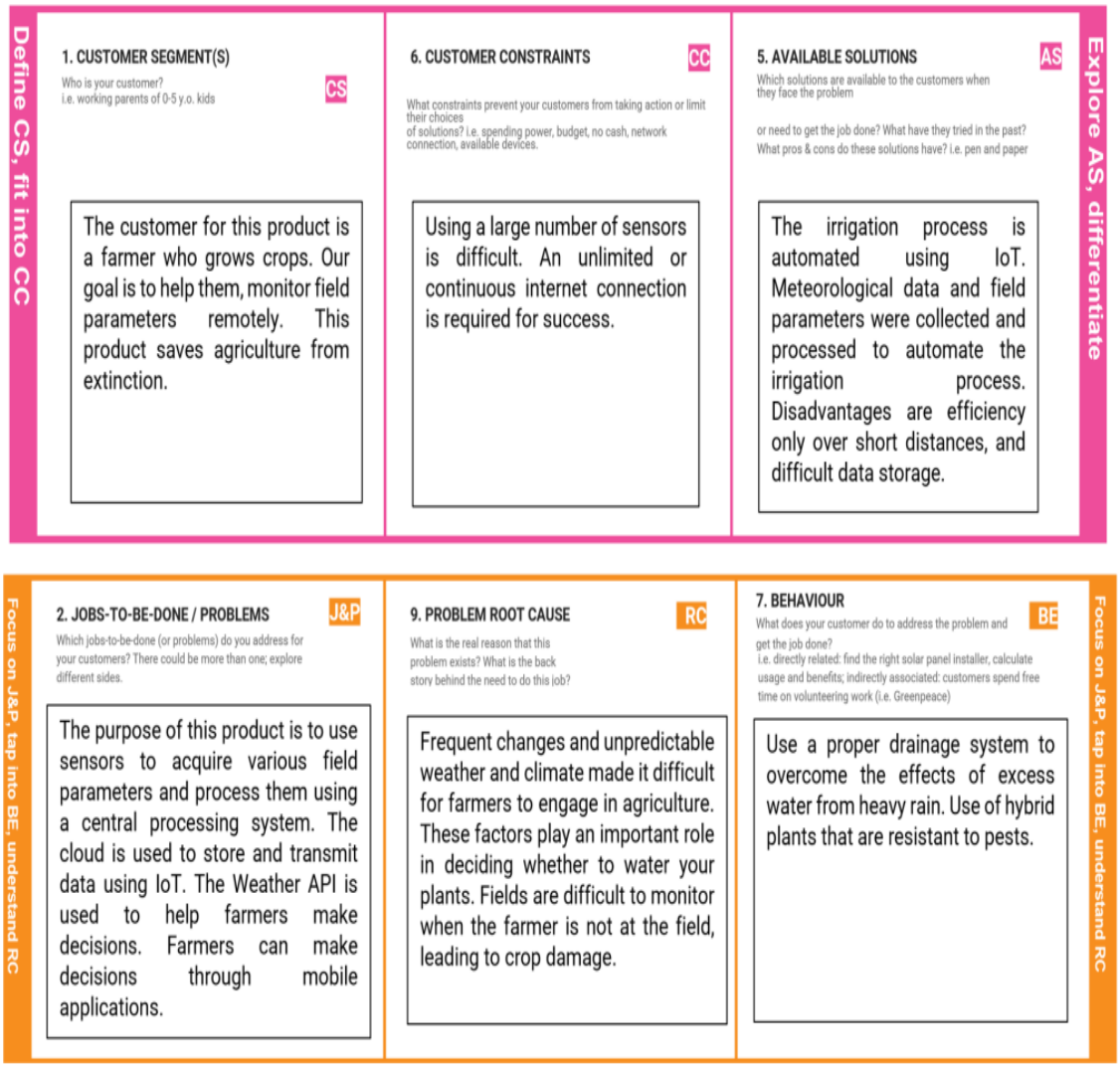
## 3.3 Proposed Solution:

To improve the efficiency of the product thereby supporting both rancher and country we need to utilize the innovation which appraises the nature of harvest and gives recommendations. The Internet of things (IOT) is revamping agribusiness, engaging farmers by a broad assortment of techniques, for instance, accuracy and conservative cultivation to go up against challenges in the field.

In this project, on a farm, management can monitor different environmental parameters effectively using sensor devices such as temperature sensor, relative humidity sensor and soil moisture sensor. Periodically (30 seconds) the sensors are collecting information of the agriculture field area and are being logged and stored online using cloud computing and Internet of Things. By using wireless transmission, the sensed data is forwarded towards the web server database. If irrigation is automated, then that means if the moisture and temperature fields fall below the potential range. The user can monitor and control the system remotely with the help of an application which provides a web interface to the user.

# 3.4 Problem Solution fit:

**Project Title:** Smart Farming apllication | **Project Design Phase-I – Problem solution Fit** | **Team ID: PNT2022TMID16477**



# 4. REQUIREMENT ANALYSIS

## 4.1 Functional requirement:

Following are the functional requirements of the proposed solution.

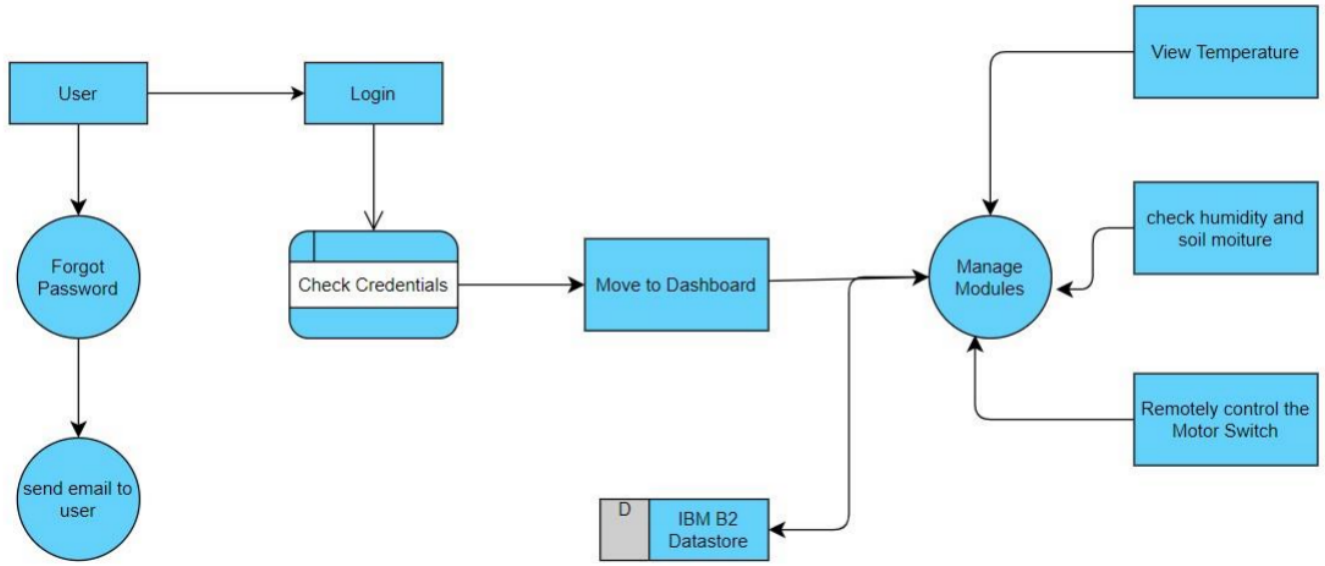
| FR No. | Functional Requirement (Epic) | Sub Requirement (Story / Sub-Task) |
|---|---|---|
| FR-1 | User Registration | Registration through Form Registration<br>through Gmail |
| FR-2 | User Confirmation | Confirmation via Email Confirmation<br>via OTP |
| FR-3 | Sensor Function for framing System | Measure the Temperature and Humidity<br>Measure the Soil Monitoring Check the crop<br>diseases |
| FR-4 | Manage Modules | Manage Roles of User<br>Manage User permission |
| FR-5 | Check whether details | Temperature details Humidity<br>details |
| FR-6 | Data Management | Manage the data of weather conditions<br>Manage the data of crop conditions<br>Manage the data of live stock conditions |

## 4.2 Non-Functional requirements:

| FR No. | Non-Functional Requirement | Description |
|---|---|---|
| FR-1 | Usability | The application must be useful to all sorts of people, Its complexity level should be low and should be usable by uneducated farmers. It should be simple rather than confusing |
| FR-2 | Security | Since it involves cloud storage of gathered sensor data, which could be misused, Data handling must be highly secure. |
| FR-3 | Reliability | Since it is used for remote monitoring, It can be used in cases where a single farmer is managing the entire farm, Data should be more accurate and should not be misleading. |
| FR-4 | Performance | Highly effective monitoring, tracking, and recovery of farm assets, tracking range should be greater than at least 5km. Continuous readings on temperature,gas,humidity,pH,smoke detection ,water and fuel levels are necessary. |
| FR-5 | Availability | It should monitor water level, fuel level, electric fence-theft monitoring, temperature, humidity, tractor guidance, GPS tags, soil moisture, and toxic gases. |
| FR-6 | Scalability | It should be made used in remote areas where technological |

# 5. PROJECT DESIGN

## 5.1 Data Flow Diagrams:




## 5.2 Solution & Technical Architecture:

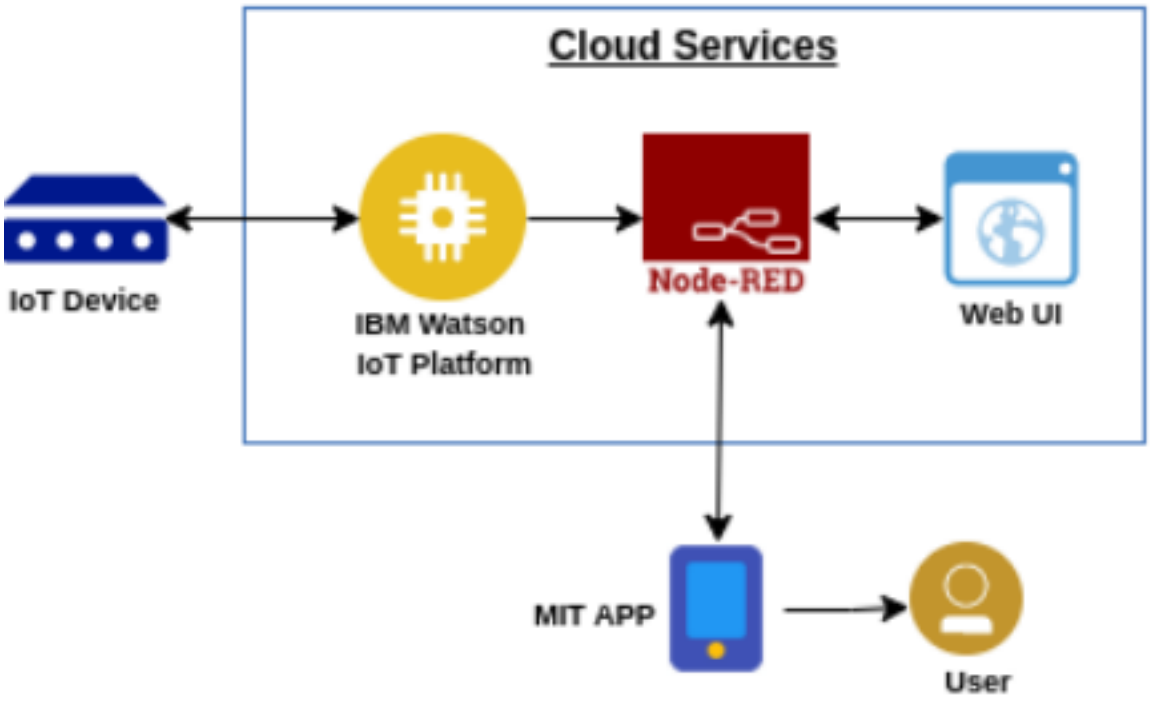
● IoT-based agriculture systems help the farmer in monitoring different parameters of his field like soil moisture, temperature, and humidity using some sensors.

● 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 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.

## Technical Architecture

## Project Flow:




● The parameters like temperature, humidity, and soil moisture are updated to the Watson IoT platform

● The device will subscribe to the commands from the mobile application and control the motors accordingly

● APIs are developed using Node-RED service for communicating with Mobile Application

● A mobile application is developed using the MIT App inventor to monitor the sensor parameters and control the motors.

● To accomplish this, we must complete all the activities and tasks listed below:

● Create and configure IBM Cloud Services

● Create IBM Watson IoT Platform

● Create a device & configure the IBM IoT Platform

● Create Node-RED service

● Create a database in Cloudant DB to store all the sensor parameters

● Develop a python script to publish and subscribe to the IBM IoT platform

● Configure the Node-RED and create APIs for communicating with mobile application

● Develop a mobile application to display the sensor parameters and control the motors

## 5.3 User Stories:

**List of all the user stories for the Smart farming application are as follows;**

| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Release |
|---|---|---|---|---|---|---|
| Farmer -Mobile user | End User-Mobile app | USN-1 | I have to check the temperature value of my field. | Click on the button to view the temperature. | High | Sprint-1 |
| | | USN-2 I | I have to monitor the soil moisture content near my crops. | Click on the button to view the moisture content | High | Sprint-1 |
| | | USN-3 I | I have to measure the humidity conditions for crop storage. | Click on the button to view the crop storage.. | Low | Sprint-2 |
| | | USN-4 I | I have to measure the current temperature and compare it with previous temperature. | Click on the option to compare results. | Medium | Sprint-1 |
| | | USN-5 | I have to visualize the graph of crop production. | Click on the graph button to visualize the results. | High | Sprint-1 |

# 6. PROJECT PLANNING & SCHEDULING

## 6.1 Sprint Planning & Estimation:

| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority | Team Members |
|---|---|---|---|---|---|---|
| Sprint-1 | Simulation creation | USN-1 | Connect Sensors and Arduino with python code | 2 | High | Kanagaraj Akash |
| Sprint-2 | Software | USN-2 | Creating device in the IBM Watson IoT platform, workflow for IoT scenarios usingNode-Red | 2 | High | Kanagaraj Akash Manojkumar Mouniesh |
| Sprint-3 | MIT App Inventor | USN-3 | Develop an application for the Smart farming project using MIT App Inventor | 2 | High | Kanagaraj Akash Manojkumar Mouniesh |
| Sprint-3 | Dashboard | USN-3 | Design the Modules and test the app | 2 | High | Kanagaraj Akash Manojkumar Mouniesh |
| Sprint-4 | Web UI | USN-4 | To make the user to interact with software. | 2 | High | Kanagaraj Akash Manojkumar Mouniesh |

## 6.2 Sprint Delivery Schedule:

| S.NO | ACTIVITY TITLE | ACTIVITY DESCRIPTION | DURATION |
|---|---|---|---|
| 1 | Understanding the project | Assign ed the team Members after that create repository in the GitHub and then assign task to each member and guide them how to access the GitHub while submitting the assignments | 1 week |
| 2 | Staring The Project | We the Team Members were Assigned all the Tasks Based on Sprints and Work on It Accordingly | 1 week |
| 3 | Completing Every Task | Team Leader should ensure that whether every team member have completed the assigned task or not | 1 week |
| 4 | Stand Up Meetings | Team Lead Must Have a Stand-Up Meeting with The Team and Work on The Updates and Requirement Session | 1 week |
| 5 | Deadline | Ensure that team members are completing every task within the deadline | 1 week |
| 6 | Budget and Scope of project | Analyze the overall budget which must be 1 week within certain limit it should be favorable to every person | 1 week |

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

**PYTHON CODE**:

To develop the python code to publish and subscribe to the commands from the IBM cloud.

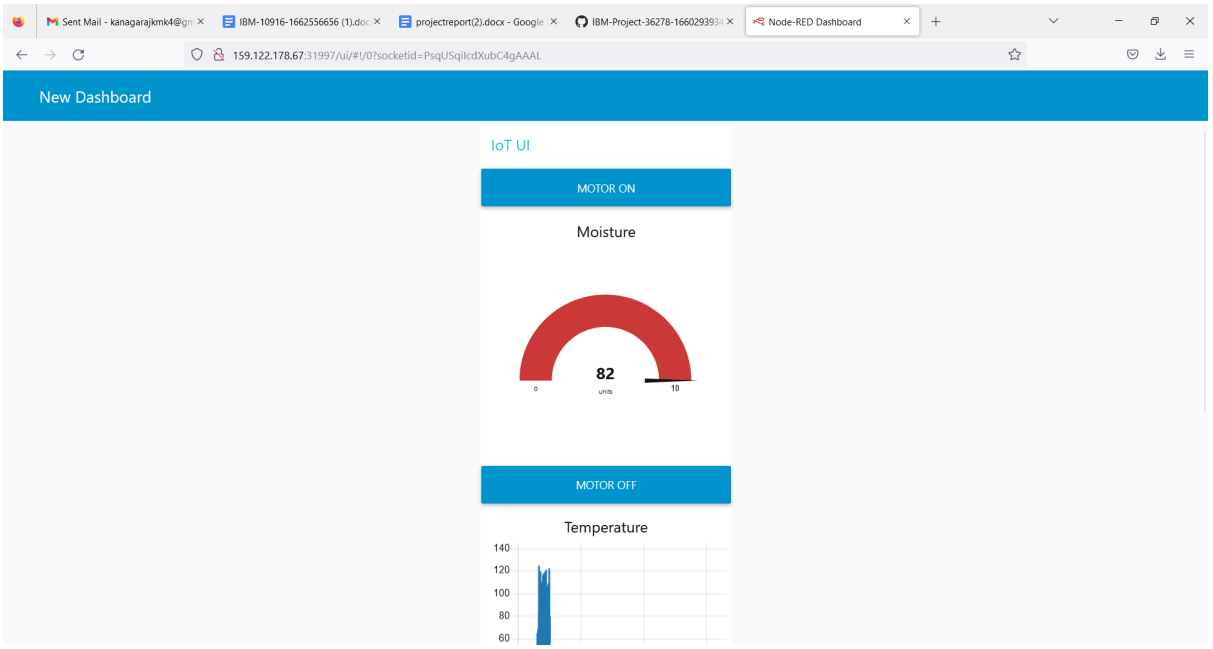
**PROGRAM:**

```
import time                                  # Team Id-PNT2022TMID16477
import sys
import ibmiotf.application
import ibmiotf.device
import random


#Provide your IBM Watson Device Credentials
organization = "mwjyar"
deviceType = "abcd"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"


# Initialize GPIO
def myCommandCallback(cmd):
        print("Command received: %s" % cmd.data['command'])
        m=cmd.data['command']
```

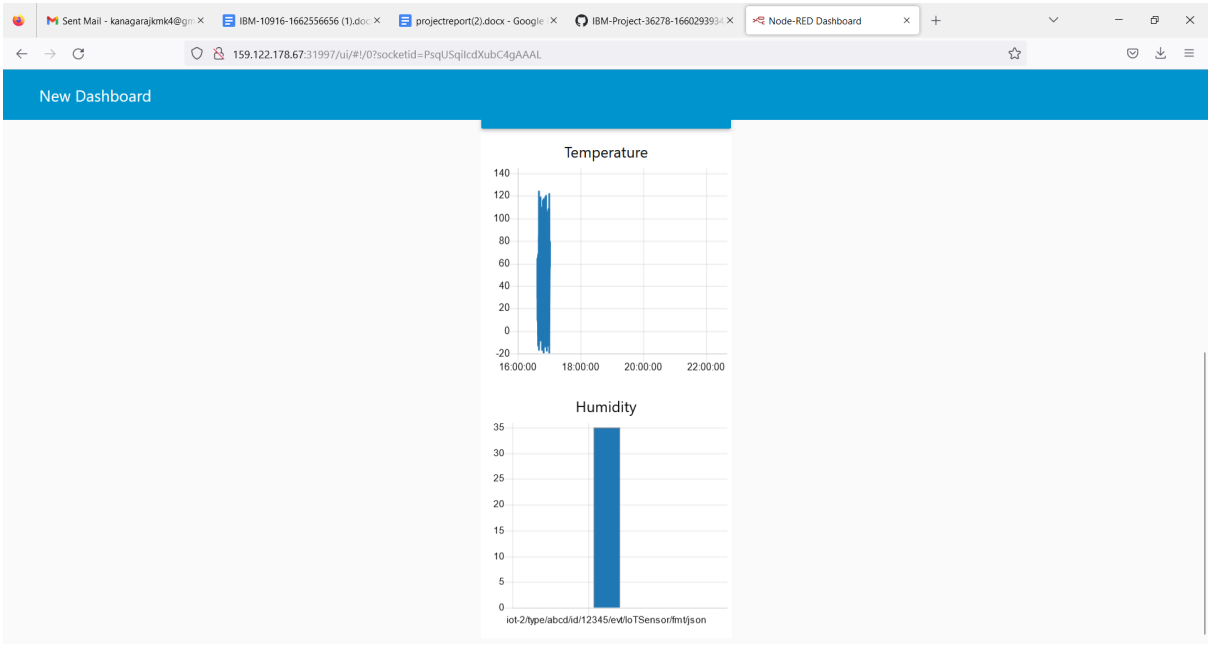
```
        if (m=="motoron"):
        print ("motor is switched on")
        elif (m=="motoroff"):
        print ("motor is switched off")
        else :
        print ("please send proper command")

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 "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:
        #Get Sensor Data from DHT11
        moist=random.randint(0,100)
```

```
        temp=random.randint(-20,125)
        Humid=random.randint(0,100)
        data = { 'moist' : moist , 'temp' : temp ,'Humid': Humid}
        #print data
        def myOnPublishCallback():
        print ("Published moist = %s C" % moist, "temp= %s %%" % temp, "Humid = %s %%" % Humid, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
        print("Not connected to IoTSensor")
        time.sleep(10)

        deviceCli.commandCallback = myCommandCallback

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

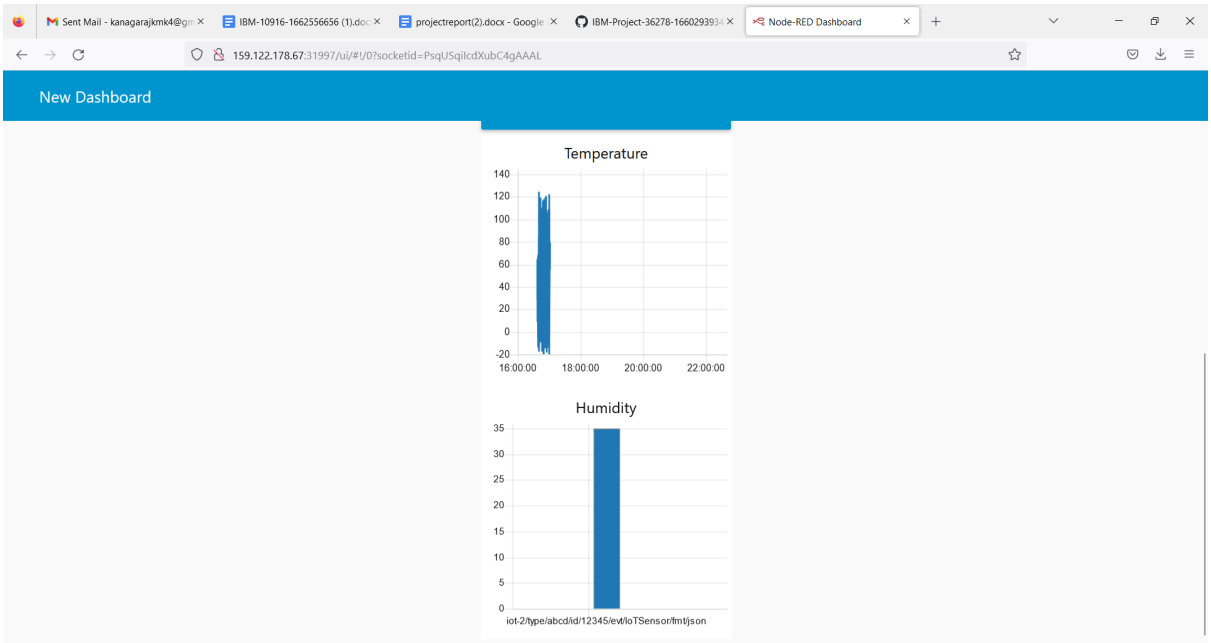
## 7.1 Feature 1- SOIL MOISTURE DETECTION



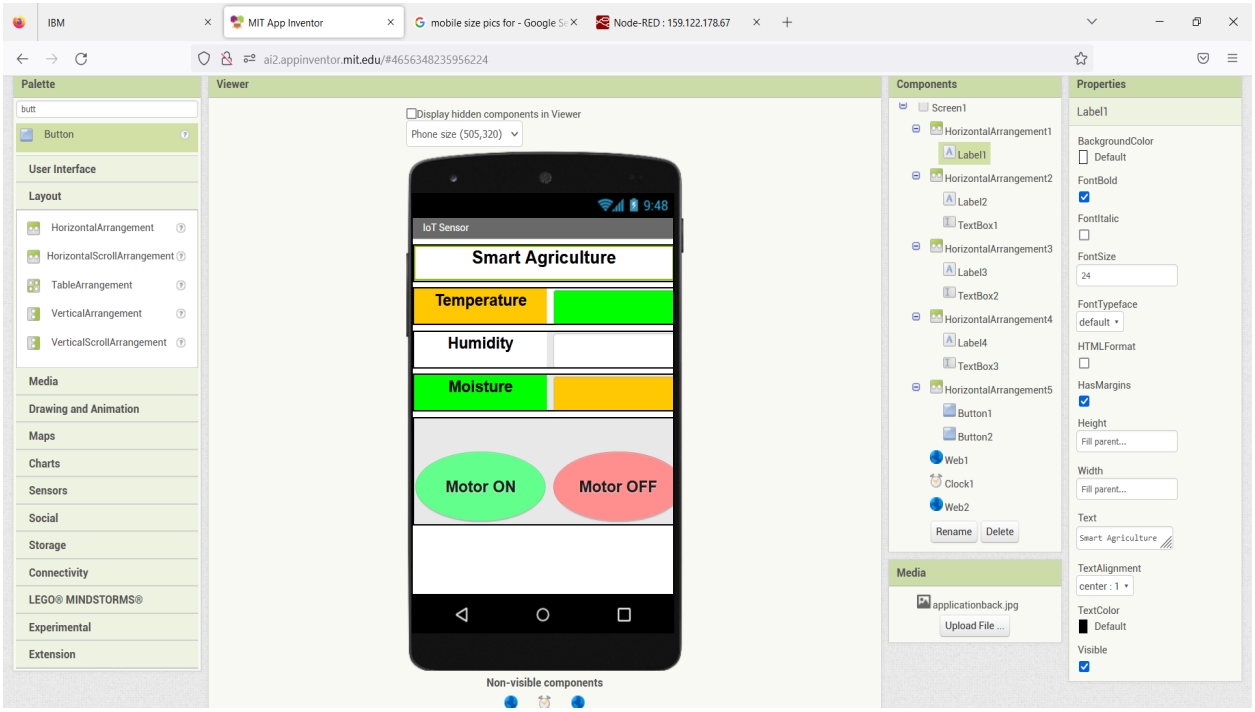

## 7.2 Feature 2- HUMIDITY DETECTION



## 7.3 Feature 3- TEMPERATURE DETECTION




## 7.4 Feature 4- MOTOR ON AND OFF



# 8. TESTING

## 8.1 Test Cases:

### 1. IBM Watson IOT service:

To create an IBM Watson IOT service Steps to create an IBM Watson IOT service: and create a device using it.

- Click on catalog in IBM cloud account.
- Click on services.
- Enter as an Internet of things platform.
- Enter region and pricing plan.
- Enter service name and click create.
- Click on launch.
- Then IBM Watson OT platform opens.
- Click on sign in.
- Enter IBM Id.
- Enter Password.
- Then you can access IBM Watson IOT platform.

## Steps to create a device:

- Click on devices in IBM Watson IOT platform.
- Choose to create a device
- Enter the device type as abcd.
- Enter the device ID as 12345.

- Click next.
- Enter device credentials (optional).
- Click next.
- Enter the authentication token as 12345678.
- Click on continue.
- Click on next.
- Click finish Device is created successfully, and we can see device credential

**2. Creation of Node – Red Service:**

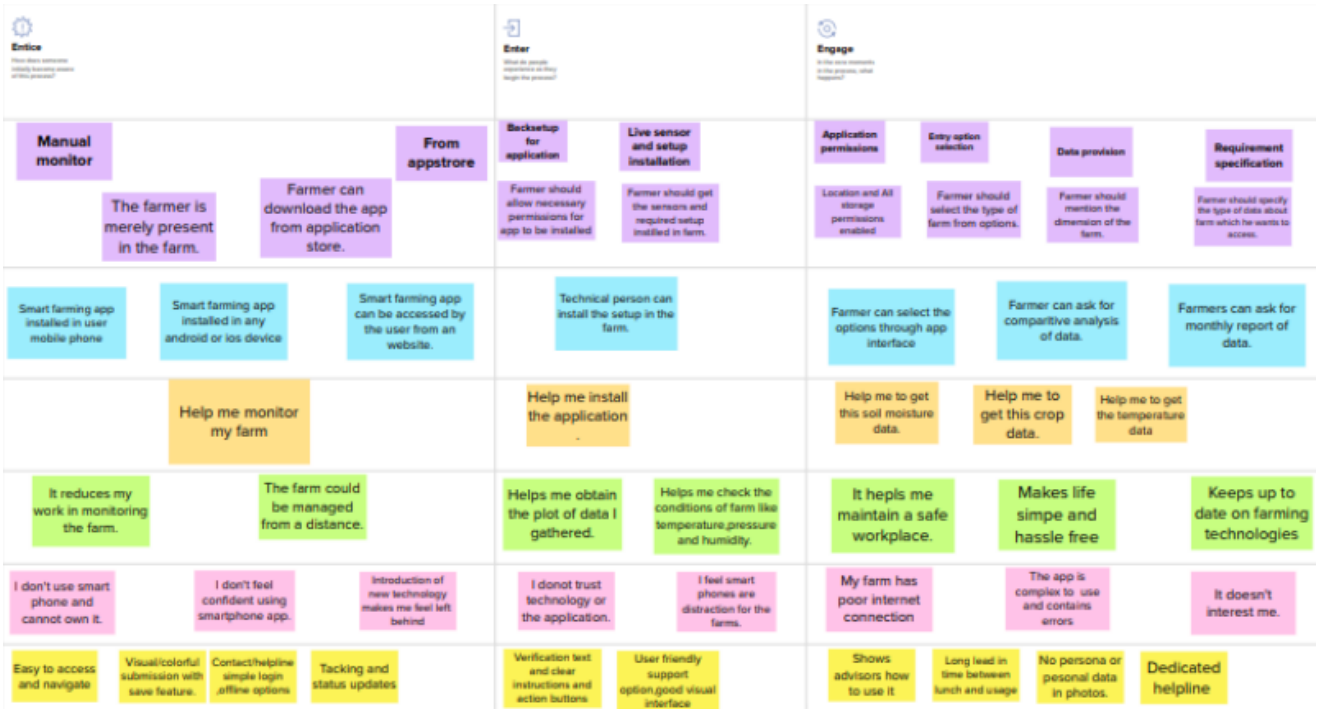
To create a Node Red service.

Steps to create a Node-Red service:

- Click on catalog in IBM cloud account.
- Click on services.
- Enter the Node red service.
- Node red app opens click on get started.
- Enter app name as default.
- Enter the region as London.
- Choose a pricing plan as lite.
- Click create.
- You will be redirected to a new page.

- Click on deploy your app.
- Choose cloud foundry.
- Enter IBM API key (by clicking new+).
- Choose memory size as default.
- Enter the region as London.
- Click next.
- Click create.
- Status will be updated after creation.
- Click on App URL.
- Click next.
- Choose not recommended.
- Click next.
- You will see the Node red page.
- Go to your node red flow editor.
- In the left panel choose nodes

## 8.2 User Acceptance Testing:

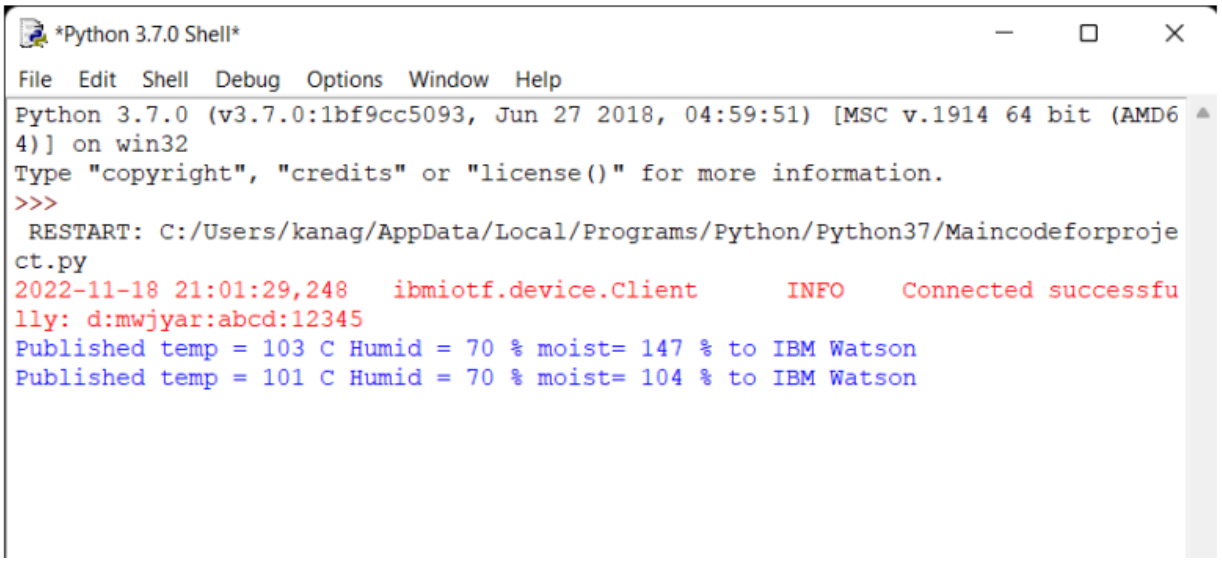



## 9. RESULTS:

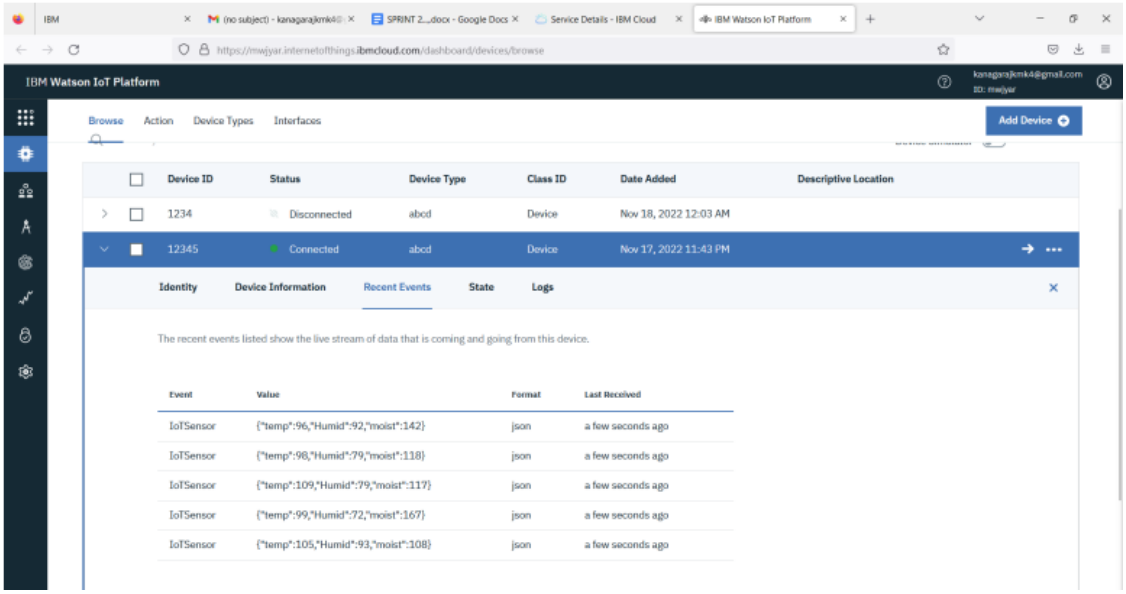
The yield appearing beneath signifies the temperature, soil moisture and humidity data received from the IoT simulator sensor and open weather Api. The web app displays all this data for the past one hour. There are a set of buttons on the web application that can be used to control the motor and light on the farm to turn them ON/OFF remotely.

## 9.1 Performance Metrics:

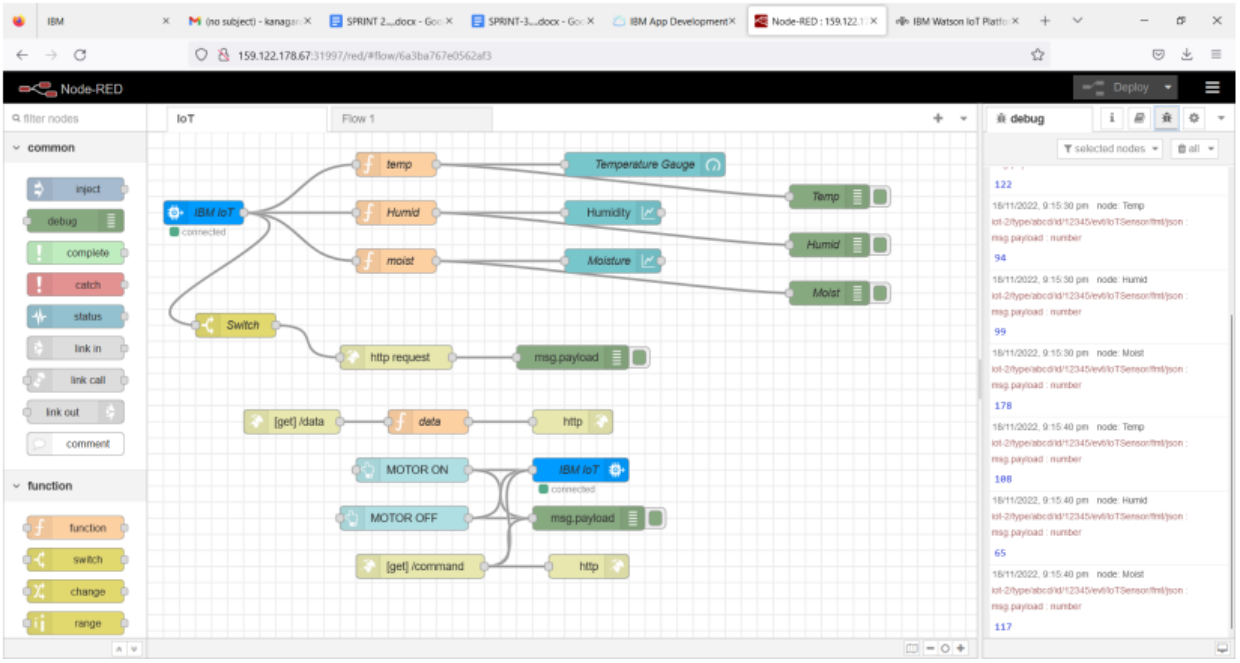
### Python code Results:



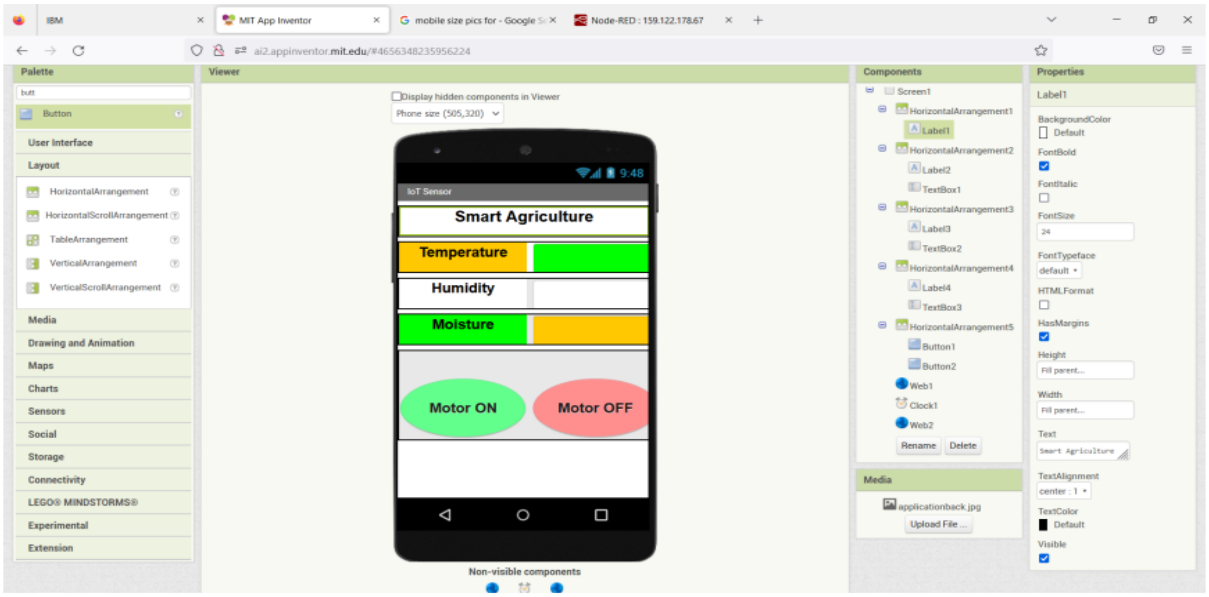
# IBM Watson-IOT Platform:



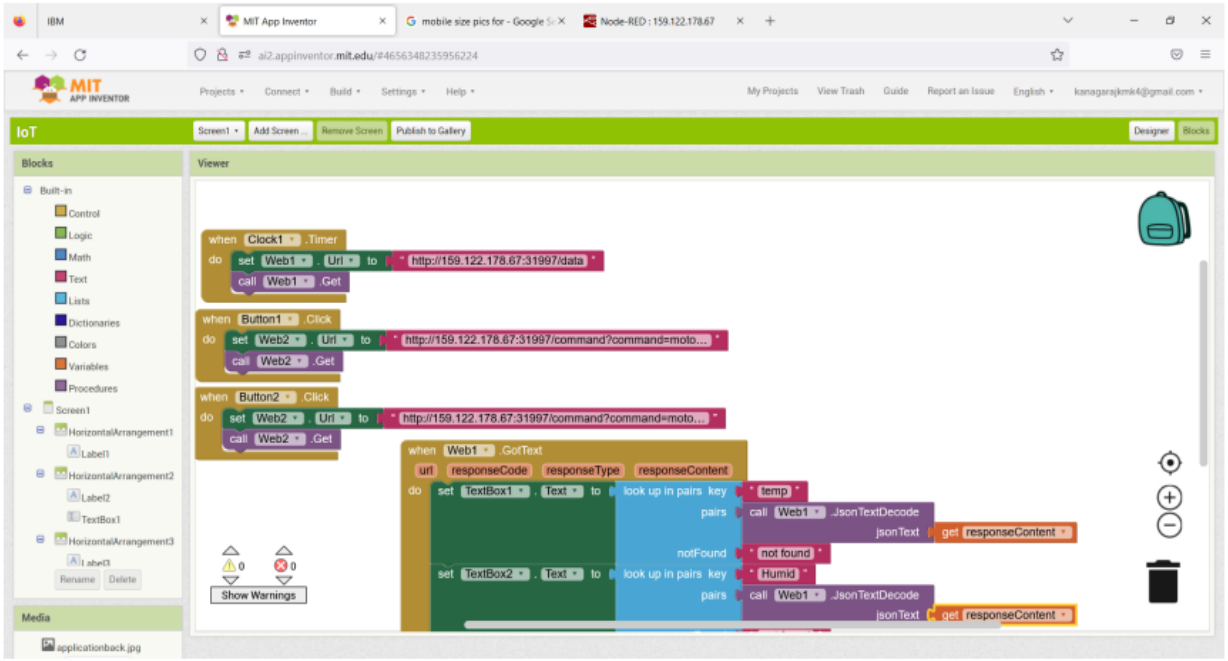

# NODE-RED Flow:

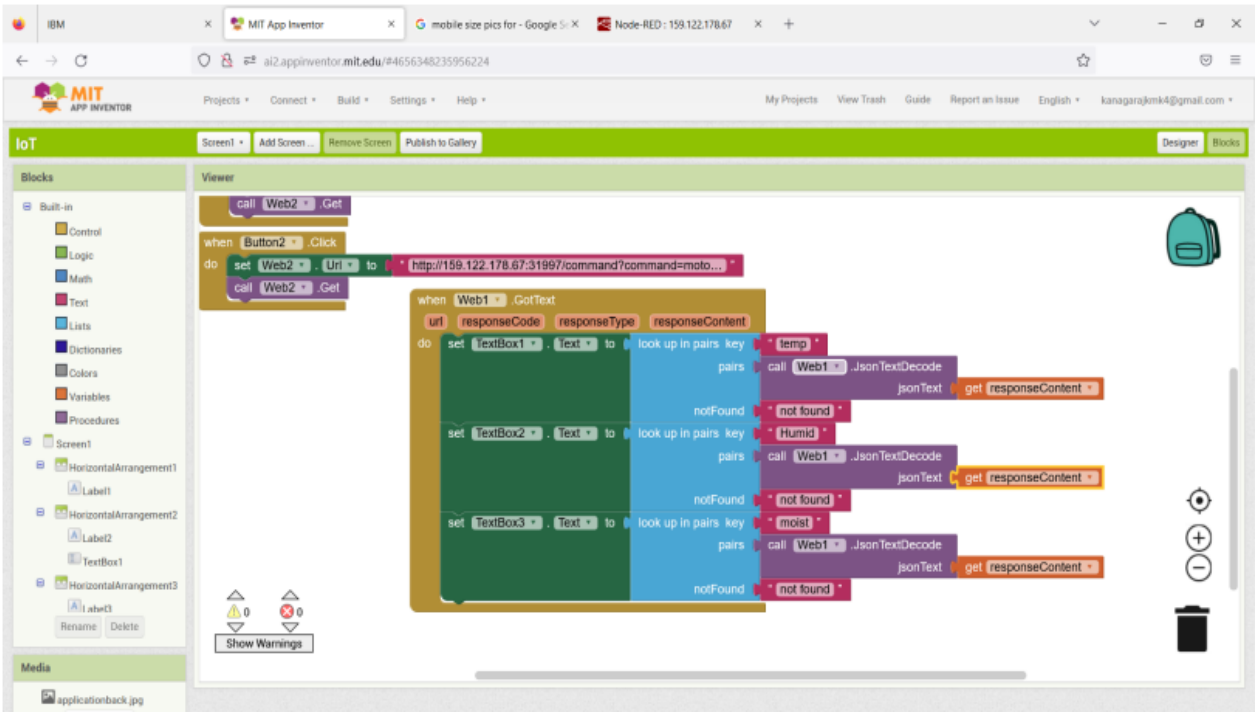


## MIT App Inventor:




## Blocks:






## 10. ADVANTAGES & DISADVANTAGES:

### ADVANTAGES:

1. Communicating the device at a larger distance through web application. It will play an important role in reducing the manpower and traveling expenses of a farmer.

2. Monitoring parameters like temperature, humidity etc. will play an important role in improving the growth of the plant.

3. Integrating the weather station to the web browser will provide the details of status of the cloud, wind speed etc. It will allow the farmer to protect their plants from natural calamities.

## DISADVANTAGE:

1. Since the real time sensor will be connected to the controller, the controller requires continuous supply of the internet to transfer the data.

2. Non availability of weather prediction for a long period of time. Since the long weather prediction requires additional payment to open weather.

## 11. CONCLUSION:

IoT based SMART AGRICULTURE SYSTEM for Live Monitoring of Temperature and Soil Moisture and to control motor and light remotely have been proposed using Node Red and IBM Cloud Platform. The System has high efficiency and accuracy in fetching the live data of temperature and soil moisture. The IoT based smart farming System being proposed via this project will assist farmers in increasing the agriculture yield and take efficient care of food production as the System will always provide a helping hand to farmers for getting accurate live feed of environmental temperature and soil moisture with more than 99% accurate results. Therefore, the project proposes a thought of consolidating the most recent innovation into the agrarian field to turn the customary techniques for water systems to current strategies in this way making simple profitable and temperate trimming.

## 12. FUTURE SCOPE:

The project has vast scope in developing the system and making it more user friendly and the additional features of the system like:

- By installing a webcam in the system, photos of the crops can be captured and the data can be sent to a database.
- Speech based options can be implemented in the system for the people who are less literate.
- GPS (Global Positioning System) can be integrated to provide specific location of the farmer and more accurate weather reports of agriculture fields and gardens.
- Regional language features can be implemented to make it easy for the farmers who are aware of only their regional language.

## 13. APPENDIX:

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

#Provide your IBM Watson Device Credentials
organization = "mwjyar"
deviceType = "abcd"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['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)

#.....................................
```

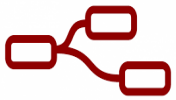
```
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()

# Connect and send a datapoint "hello" with value "world" into the
cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:
        #Get Sensor Data from DHT11

        temp=random.randint(90,110)
        Humid=random.randint(60,100)
        moist=random.randint(10,80)

        data = { 'temp' : temp, 'Humid': Humid, 'moist' : moist}
        #print data
        def myOnPublishCallback():
        print ("Published Temperature = %s C" % temp, "Humidity = %s
%%"                % Humid, "moisture = %s %%" % moist, "to IBM
Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data,
qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(10)

        deviceCli.commandCallback = myCommandCallback

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

## Technologies used in this project:

### Node-Red:

http://159.122.178.67:31997/red/#flow/6a3ba767e0562af3/ui

### Node-Red DashBoard:



http://159.122.178.67:31997/ui/#!/0?socketid=PsqUSqiIcdXubC4gAAAL

### IBM Cloud:

https://cloud.ibm.com/login

### MIT APP:

http://ai2.appinventor.mit.edu/#4656348235956224

## GitHub:

https://github.com/IBM-EPBL/IBM-Project-36278-1660293934

**Drive Link:Demo Video Link**

https://drive.google.com/file/d/1ahVBzdcLIj1E8bOtE5nzYm4NfDMQ3YYt/view?usp=share_link