PROJECT REPORT

Project Name: SMART FARMER- IOT ENABLED SMART FARMING APPLICATION.

Team ID: PNT2022TMID18915

Team:

Glenda Lorelle Ritu E – Team Lead
Dinakaran C
Navin K
Jagan Raj

TABLE OF CONTENT

SMART FARMING

1.INTRODUCTION:

1.1 PROJECT OVERVIEW:

This is system that enables framers to monitor and their forms with a webbased application build with Node-RED. It uses the IBM IOT Watson cloud platform as its Backend.

1.2 Purpose:

Smart Farming reduce the ecological foodprint 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 ofgreenhouse gases.

2. LITERATURE SURVEY:

2.1 EXISTING PROBLEM:

The biggest challenges faced by IoT in the agricultural sector are lack of information, high adoption costs, and security concerns, etc. Most of the farmers are not aware of the implementationof IoT in agriculture.

2.2 REFERENCES:

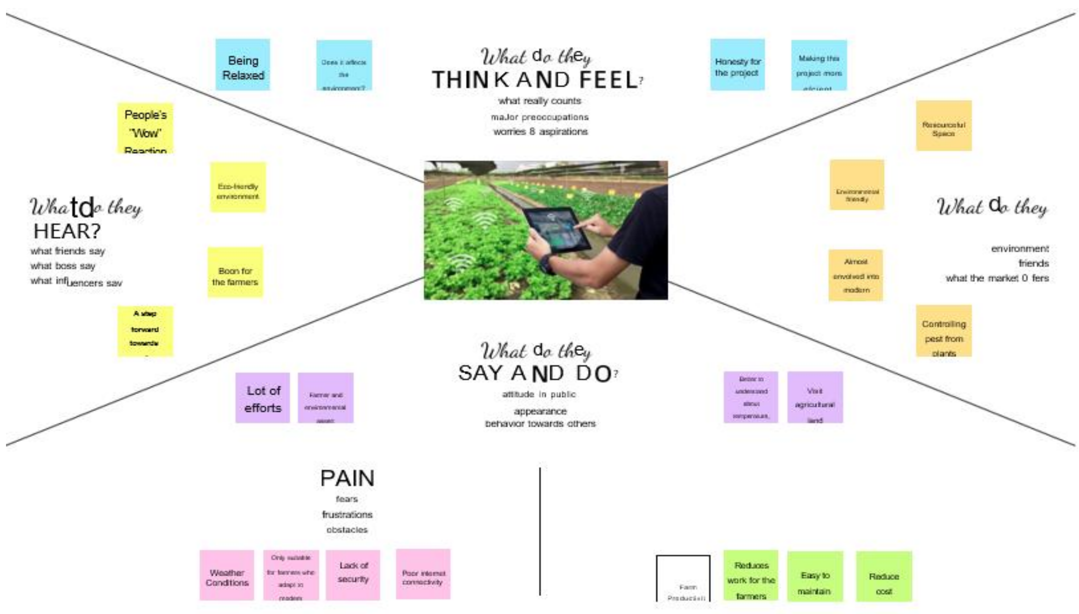
It is the application of modern ICT (Information and Communication Technologies) into agriculture. In IOT- based smart farming, a system is built for monitoring the crop field with the help of sensors (light, humidity, temperature, soil moisture, etc.). The farmers can monitor the field conditions from anywhere.

2.3 PROBLEM STATEMENT DEFINITION:

Overuse of pesticides and fertilizer in agricultural fields leads to destruction of the crop as well as reduces the efficiency of the field increasing the soil vulnerability toward pest. IoT applications may be used to update the farmer/ user about type & quantity of pesticide required by the crop.

3. IDEATION & PROPOSED SOLUTION:

3.1 EMPATHY MAP CANVAS:



3.2 IDEATION & BRAINSTORMING:

Ideation is the create process of generating, developing, and communicating new ideas, where an is idea understood as a basic element of thought that can be either visual, concrete, or abstract . **Brainstorming** is a group creative technique by which efforts are made to find a conclusion for a specific problem by gathering a list of ideas spontaneously contributed by its members.

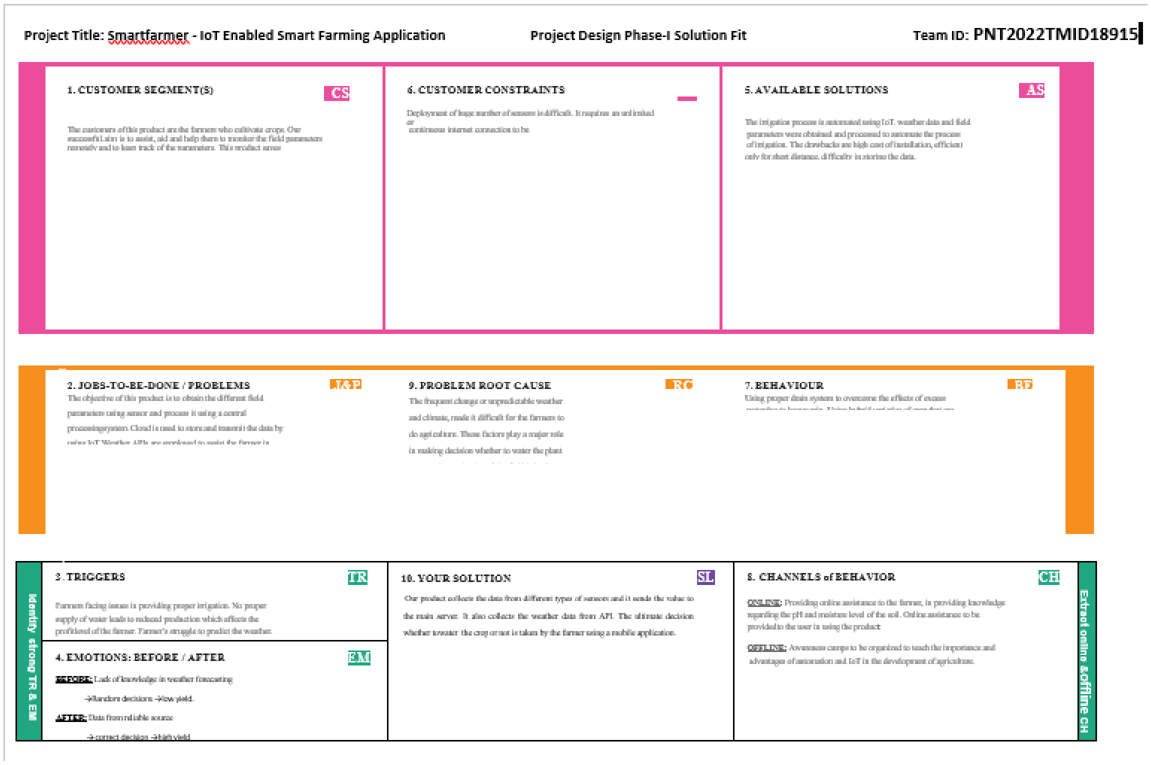
IDEATION PROCESS

3.3 Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To provide an effective decision support system employing a wireless sensor network that manages various agricultural activities and provides pertinent farm information on temperature, humidity, and soil moisture content. The weather is to blame for the rising water level. There are many distractions for farmers, which is bad for agriculture.
2.	Idea / Solution description	solutions for Smart Agricultural Systems offer an integrated IOT platform in the agricultural sector that enables farmers to use sensors, smart gateways, and monitoring systems to gather data, manage numerous farm characteristics, and analyze real- time data to make educated decisions.
3.	Novelty / Uniqueness	The use of IOT principles in agriculture has been the focus of several prominent researchers working toward smart farming. But there are still a number of difficulties that have not yet found an appropriate solution. This study attempts to highlight prior work and unresolved issues in IOT-based agriculture.
4.	Social Impact / Customer Satisfaction	Decreases thepay for workers who are employed in the agricultural sector. It helps people to save lots of time. By enriching the customer experience overall, and also IOT can help strengthen customer relationships.
5.	Business Model (Revenue Model)	Farmers are charged a monthly subscription for the prediction and recommendation of irrigation scheduling based on sensor metrics such as temperature, humidity, and soilmoisture.
6.	Scalability of the Solution	Scalability in smart farming refers to a system's ability to expand its capacity, such as the number of technological components like sensors and actuators, while enabling for quick analysis.

3.4 PROBLEM SOLUTIONS FIT:



4. REQUIREMENT ANALYSIS:

4.1 FUNCTIONAL ANALYSIS:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	EMAIL: Enter email addressPASSWORD: Enter password
FR-2	User Confirmation	Confirmation via Email. Thanks for your email.
FR-3	Log in to system	Serve authenticated content
FR-4	Manage Modules	Manage System Admins Manage Roles of User Manage User permission
FR-5	Check whether condition	Temperature monitoringstatus Humidity monitoringStatus
FR-6	Log out	Exit

4.2 NON FUNCTIONAL REQUIREMENTS:

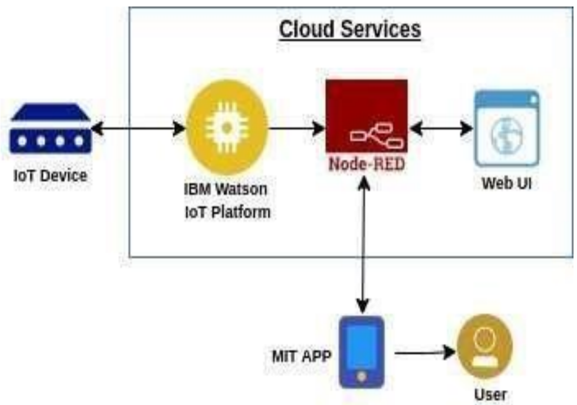
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Usability includes easy understanding and learn ability, efficiency in use,remember ability, lack of errors in operation and subjective pleasure.
NFR-2	**Security**	Sensitive and private data must be protected from their production until the decision-making and storage stages.
NFR-3	**Reliability**	The shared protection achieves a better trade-off between costs and reliability. The model uses dedicated and shared protection schemes to avoid farm service outages.
NFR-4	**Performance**	The idea of implementing integrated sensors with sensing soil and environmental parametersin farming will be more efficient.
NFR-5	**Availability**	Automatic adjustment of farming equipment made possible by linking information like crops/weather and equipment to auto-adjust temperature, humidity, etc.
NFR-6	**Scalability**	Scalability is a major concern for IoT platforms. It has shown that different architectural choices of IoT platforms affect system scalability,real time decision-making is feasible in an environment composed of dozens of thousand.

5. PROJECT DESIGN:

5.1 DATA FLOW DAIGRAMS AND USER STORIES:

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 inthe IBM cloud.
- Arduino UNO is used as a processing Unit that process the data obtainedfrom 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 applicationthat was developed using the MIT app inventor.
- The user could plan 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:

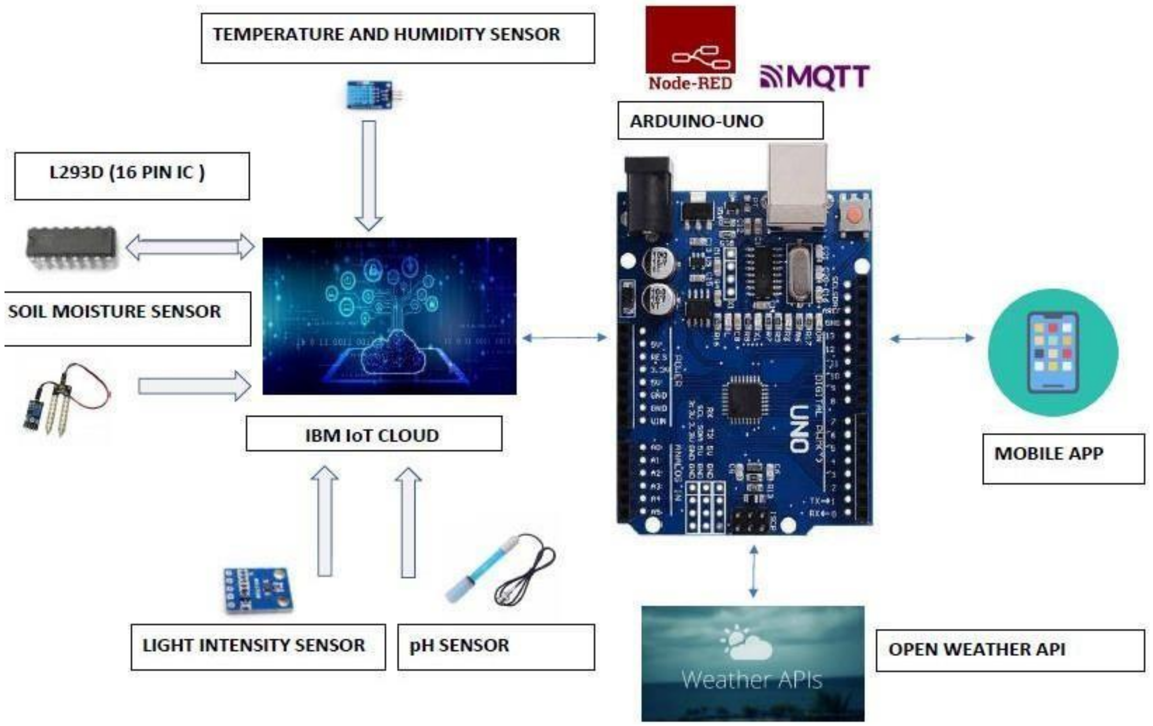
S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	MIT app
2.	Application Logic-1	Logic for a process in the application	Node red/IBM Watson/MIT app
3.	Application Logic-2	Logic for a process in the application	Node red/IBM Watson/MIT app
4.	Application Logic-3	Logic for a process in the application	Node red/IBM Watson/MIT app
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM cloud.
7.	Temperature sensor	Monitors the temperature of the crop	
8.	Humidity sensor	Monitors the humidity	
9.	Soil moisture sensor (Tensiometers)	Monitors the soil temperature	
10.	Weather sensor	Monitors the weather	.
11.	Solar panel		.
12.	RTC module	Date and time configuration	
13.	Relay	To get the soil moisture data	

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	MIT app,Node-Red	Software
2.	Scalable Architecture	Drone technology, pesticide monitoring ,Mineral identification in soil	Hardware

5.2 Solution Architecture Diagram:

➢ The different soil parameters (temperature, humidity, light intensity, pH level) are sensed using different sensors and the obtained value is storedin IBM cloud.



- The L293D is a 16-pin Motor Driver IC which can control a set of two DC motors simultaneously in any direction. The L293D is designed to provide bidirectional drive currents of up to 600 mA (per channel) at voltages from 4.5 V to 36 V (at pin 8!).

- Arduino UNO is used as a processing unit which processes the dataobtained from sensors and weather data from weather API.

- Node-RED is a programming tool for wiring together hardware devices,APIs and online services in new and interesting ways. It provides a browser-based editor.

- The MQTT protocol is followed for communication.

- All the collected data are provided to the user through a mobileapplication which was developed using MIT app inventor.

- Open Weather provides hyperlocal minute forecast, historical data, current state and from short-term to annual and forecasted weather data.All data is available via industry standard APIs.

- The user could make decision through an app, whether to water the cropor not, depending upon the sensor values.

6.PROJECT PLANNING AND SCHEDULING:

6.1 Sprint delivery plan:

Sprint	Functional Requirement (Epic)	User Story Number	User Story /Task	Story Points	Priority	Team Member

Sprint-1	Registration (Farmer Mobile User)	UNS- 1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	GLENDA LORELLE RITU (Leader)
Sprint-1	Login	UNS-2	As a user, I will receive confirmation email once I have registered for the application	1	High	JAGAN RAJ (Member 1)

Sprint-2	User Interface	UNS-3	As a user, I can register for the application through Facebook	3	Low	DINAKARAN C (Member 2)
Sprint-1	Data Visualization	UNS-4	As a user, I can register for the application through GMAIL	2	Medium	NAVIN (Member 3)
Sprint-3	Registration (Farmer - Web User)	USN - 1	As a user, I can log into the application by entering email and password	3	High	NAVIN

Sprint - 2	Login	USN - 2	As a registered user, I need to easily login log into my registered account via the web page in minimum time	3	High	GLENDA LORELLLE RITU (Leader)

Sprint - 4	Web UI	USN - 3	As a user, I need to have a friendly user interface to easily view and access the resources	3	Medium	JAGAN RAJ (Member 1)

Sprint - 1	Registration (Chemical Manufacturer - Web user)	USN - 1	As a new user, I want to first register using my organization email and create a password for the account.	2	High	DINAKARAN C (Member 2)
Sprint - 4	Login	USN - 2	As a registered user, I need to easily log in using the registered account via the web page.	3	High	GLENDA LORELLE RITU

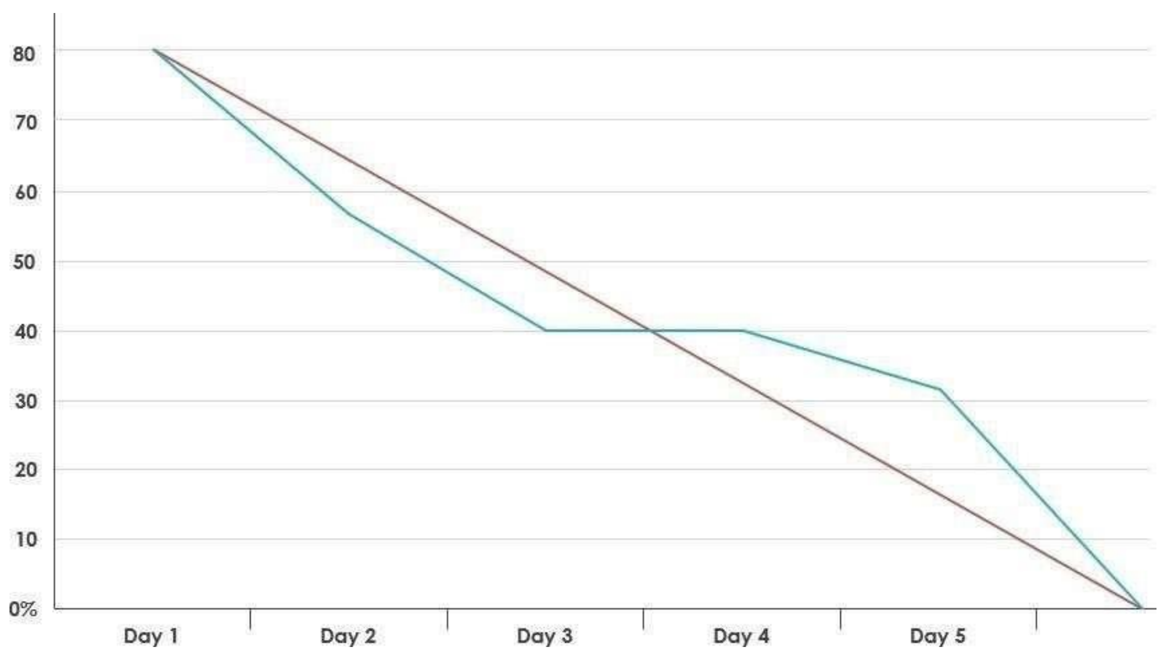
Sprint - 3	Web UI	USN - 3	As a user, I need to have a user friendly interface to easily view and access the resources.	3	Medium	JAGAN RAJ
Sprint - 1	Registration (Chemical Manufacture	USN - 1	As a user, I want to first register using my email and create a	1	High	DINAKARAN C

Sprint - 1	Login	USN - 2	As a registered user, I need to easily log in to the application.	2	Low	NAVIN

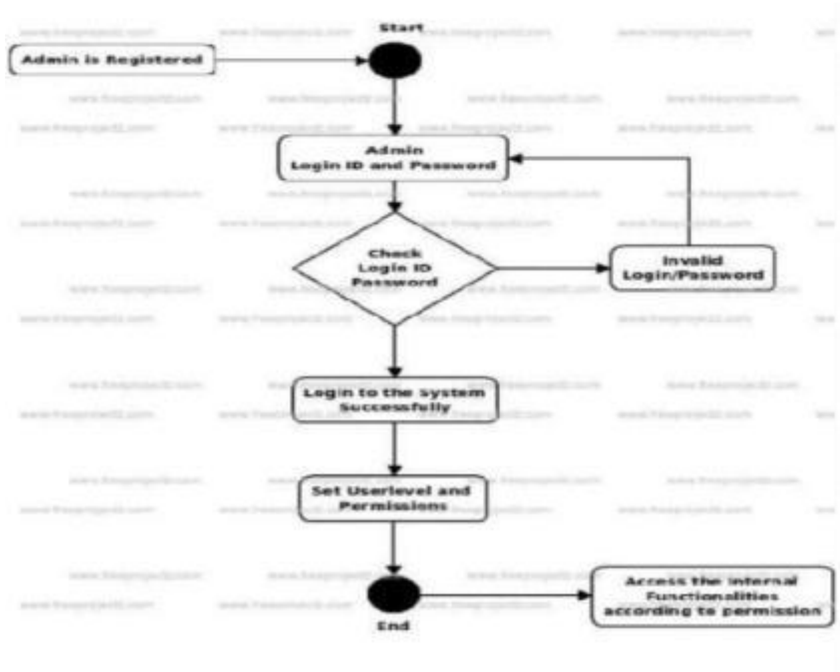
5.2 Sprint delivery schedule :

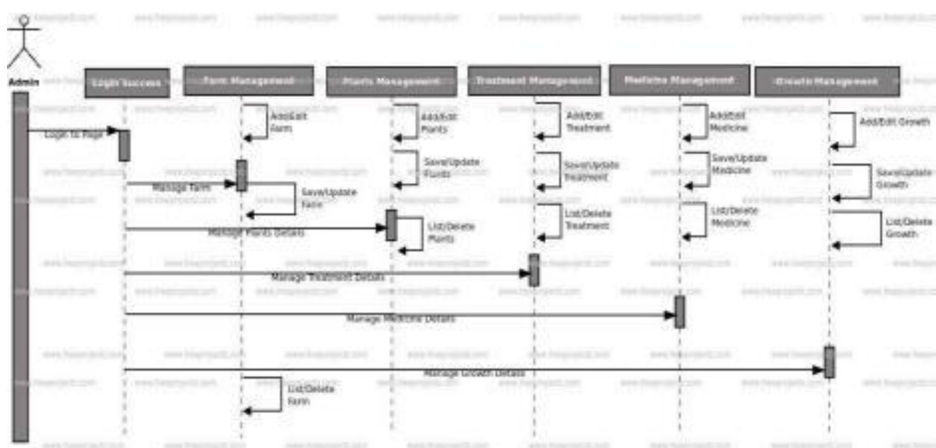
Sprint Delivery	Prepare the Sprint delivery on Number of Sprint planning meetings organized, Minutes of meeting recorded.	01 NOVEMBER 2022

Burndown Chart:



Develpoment phase:





Sequence Diagram of Farm Management System

7. CODING & SOLUTIONS:

FEATURES :

ibmiot.py - C:\Users\Priya\OneDrive\Desktop\users18\ibmiot.py (3.7.0)

File Edit Format Run Options Window Help

```python
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device
organization = "ck2tf0"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "token"
authToken ="87654321"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Commandreceived: %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 aneventof type "greeting" 10 times
deviceCli.connect()
while True:
#Get Sensor Data fromDHT11
   temp=random.randint(90,110)
   Humid=random.randint(60,100)
   Mois=random. randint(20,120)
   data = { 'temp' : temp, 'Humid': Humid ,'Mois': Mois}
#print data
def myOnPublishCallback():
  print ("Published Temperature = %s C" % temp, "Humidity = %s %%" %Humid, "Moisture =%s deg c" % Mois, "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()
```

Ln: 49 Col: 22

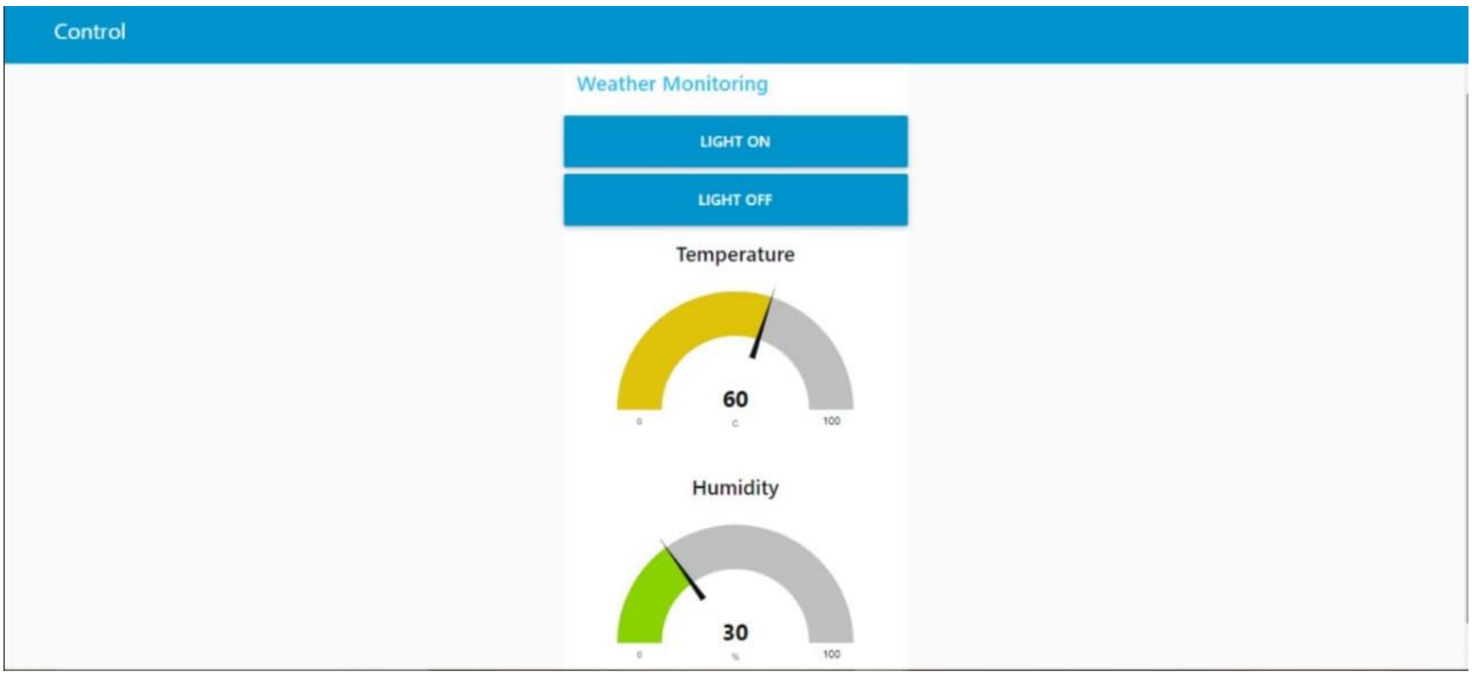
86°F Mostly sunny ENG IN 2:37 PM 11/17/2022

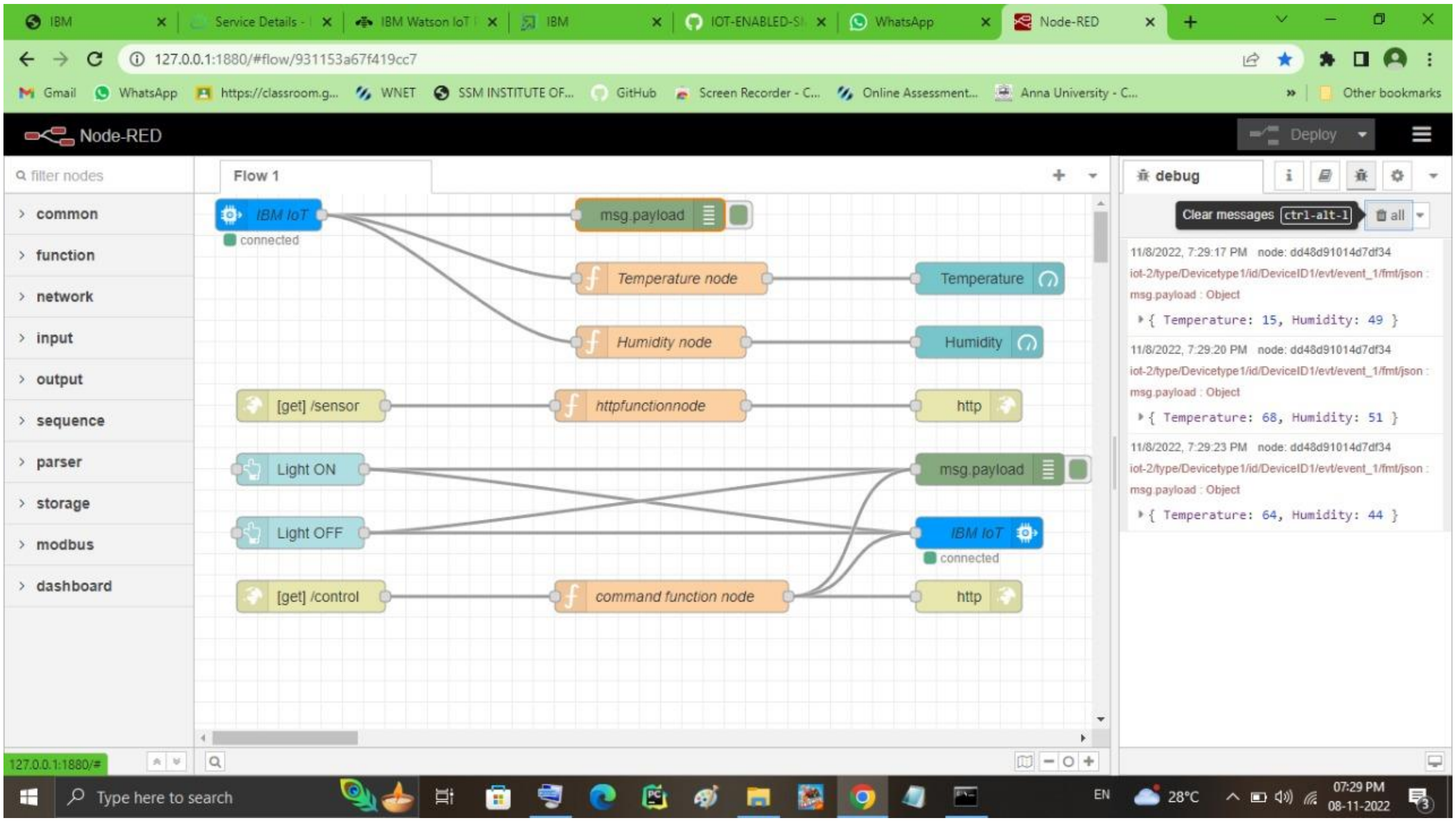
.

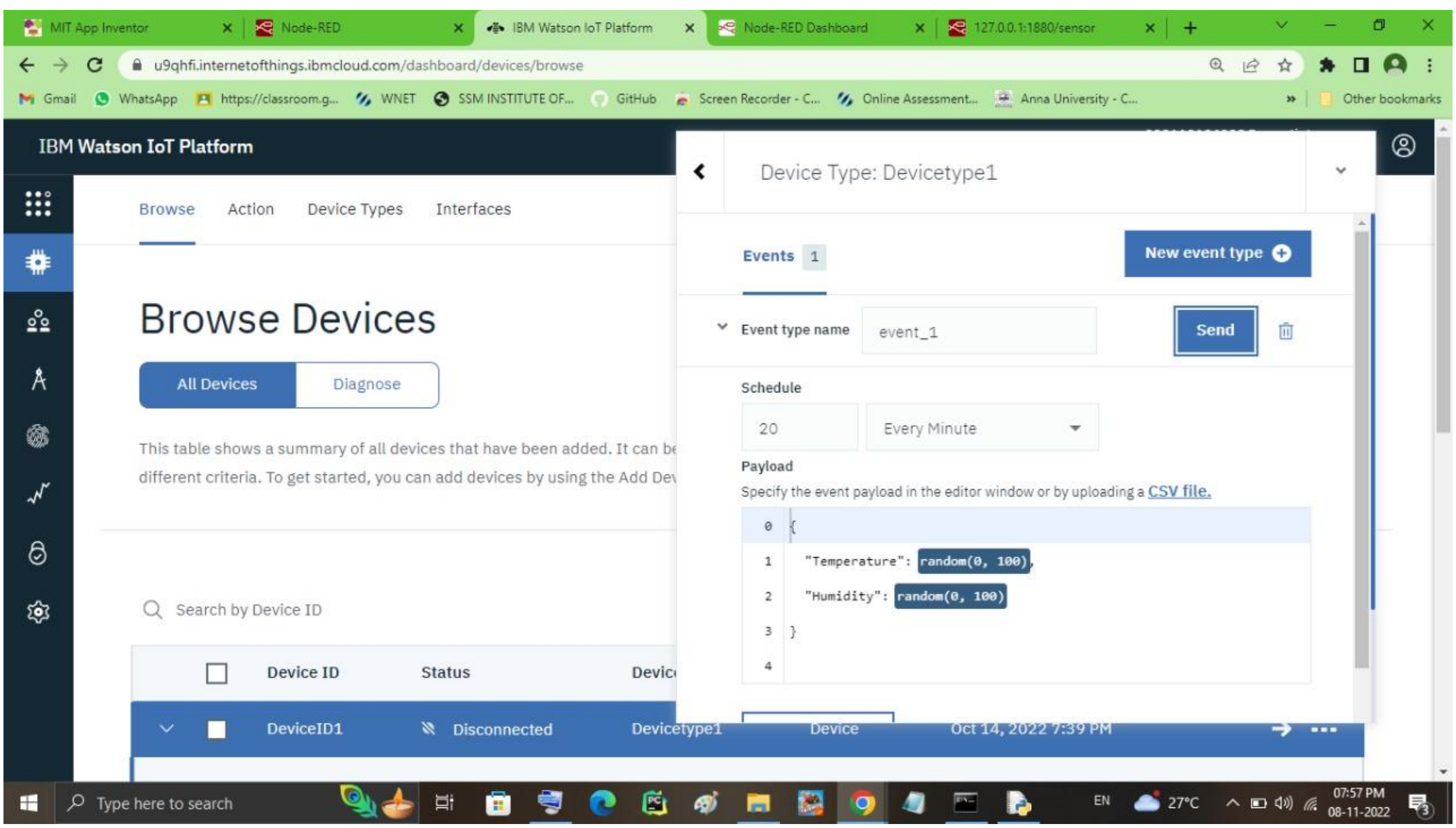
8. TESTING:

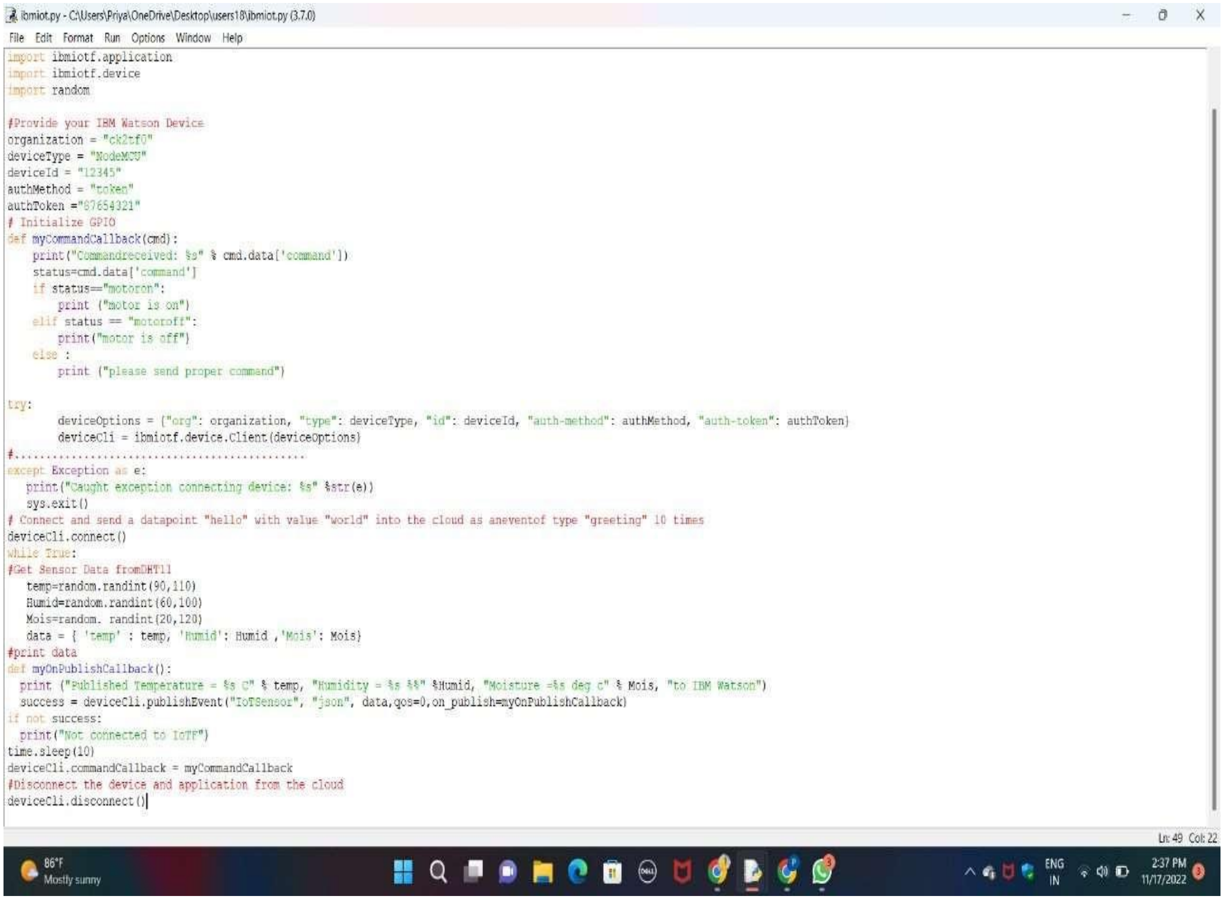
8.1 TEST CASE:

Web application using Node-RED.




Node-RED
Deploy
filter nodes
common
function
network
input
output
sequence
parser
storage
modbus
dashboard
Flow 1
IBM IoT
connected
msg.payload
Temperature node
Temperature
Humidity node
Humidity
[get] /sensor
httpfunctionnode
http
Light ON
Light OFF
msg.payload
IBM IoT
connected
[get] /control
command function node
http
debug
Clear messages ctrl-alt-l
all
11/8/2022, 7:29:17 PM node: dd48d91014d7df34
iot-2/type/Devicetype1/id/DeviceID1/evt/event_1/fmt/json : msg.payload : Object
{ Temperature: 15, Humidity: 49 }
11/8/2022, 7:29:20 PM node: dd48d91014d7df34
iot-2/type/Devicetype1/id/DeviceID1/evt/event_1/fmt/json : msg.payload : Object
{ Temperature: 68, Humidity: 51 }
11/8/2022, 7:29:23 PM node: dd48d91014d7df34
iot-2/type/Devicetype1/id/DeviceID1/evt/event_1/fmt/json : msg.payload : Object
{ Temperature: 64, Humidity: 44 }
127.0.0.1:1880/#flow/931153a67f419cc7
Type here to search
28°C
07:29 PM
08-11-2022


u9qhfi.internetofthings.ibmcloud.com/dashboard/devices/browse
IBM Watson IoT Platform
Browse
Action
Device Types
Interfaces
Browse Devices
All Devices
Diagnose
This table shows a summary of all devices that have been added. It can be
different criteria. To get started, you can add devices by using the Add Dev
Search by Device ID
Device ID
Status
DeviceID1
Disconnected
Devicetype1
Device
Oct 14, 2022 7:39 PM
Device Type: Devicetype1
Events 1
New event type
Event type name
event_1
Send
Schedule
20
Every Minute
Payload
Specify the event payload in the editor window or by uploading a CSV file.
{
"Temperature": random(0, 100),
"Humidity": random(0, 100)
}
Type here to search
27°C
07:57 PM
08-11-2022

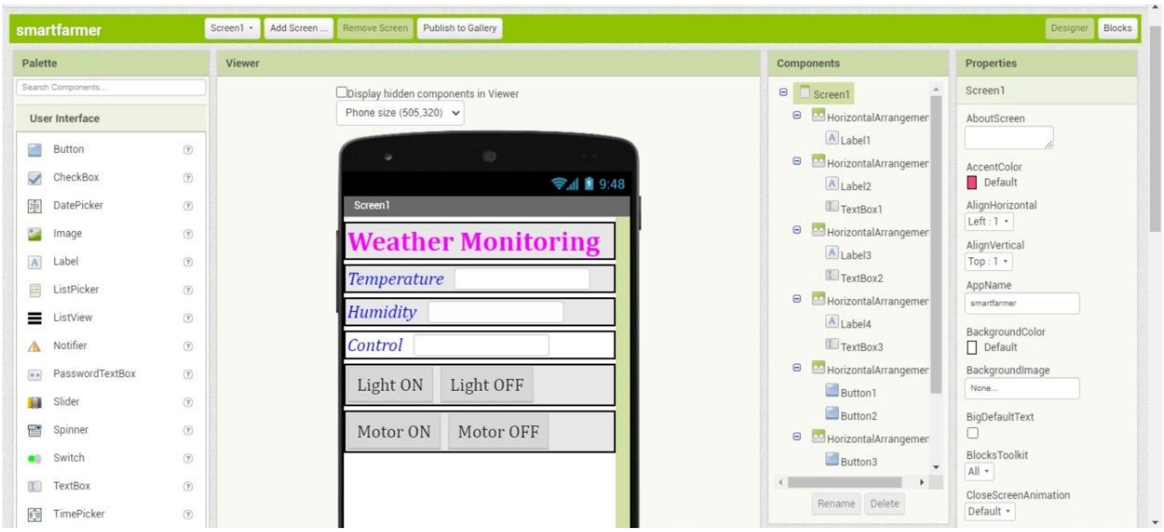


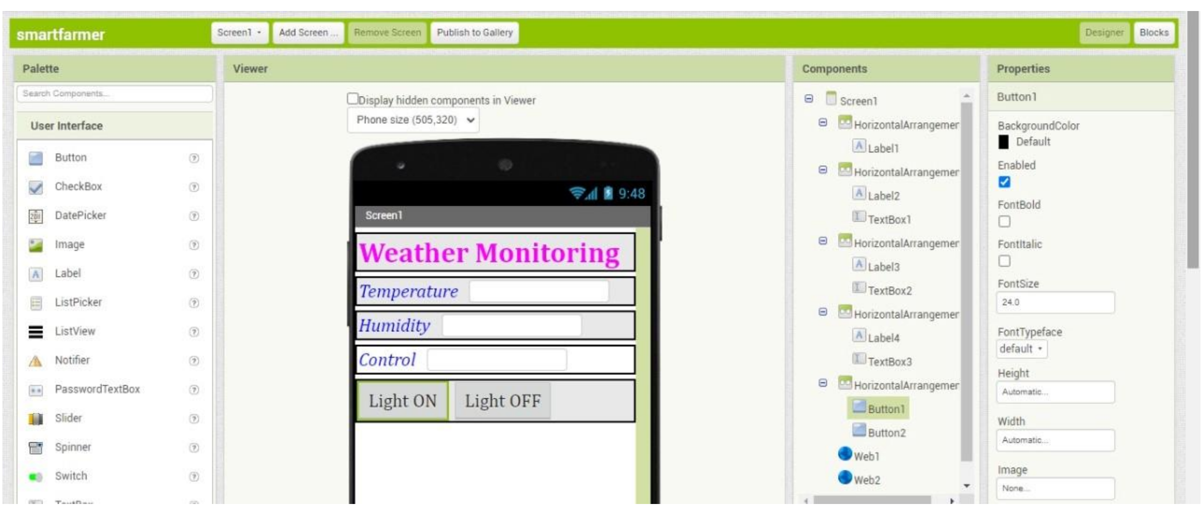

```python
import ibmiotf.application
import ibmiotf.device
import random

#Provide your IBM Watson Device
organization = "ck2tf0"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "token"
authToken ="87654321"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Commandreceived: %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 aneventof type "greeting" 10 times
deviceCli.connect()
while True:
#Get Sensor Data fromDHT11
   temp=random.randint(90,110)
   Humid=random.randint(60,100)
   Mois=random. randint(20,120)
   data = { 'temp' : temp, 'Humid': Humid ,'Mois': Mois}
#print data
def myOnPublishCallback():
  print ("Published Temperature = %s C" % temp, "Humidity = %s %%" %Humid, "Moisture =%s deg c" % Mois, "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()
```

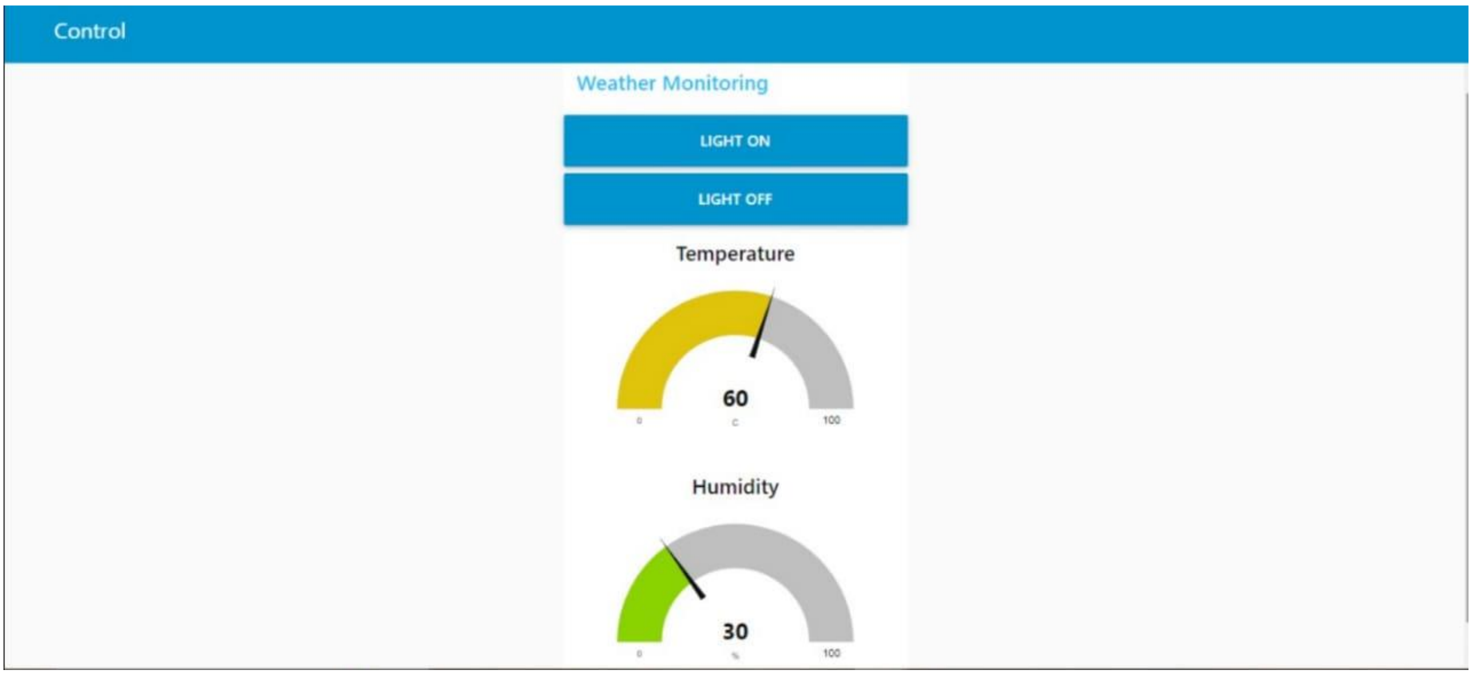
8.3 User Acceptance Testing





9. RESULT:

9.1 Performance Metrics



10. ADVANTAGES AND DISADVANTAGES:

10.1 ADVANTAGES:

- All the data like climatic conditions and changes in them, soil or crop conditions everything can be easily monitored.
- Risk of crop damage can be lowered to a greater extent.
- Many difficult challenges can be avoided making the process automated and the quality of crops can be maintained.
- The process included in farming can be controlled using the web applications from anywhere, anytime.

10.2 DISADVANTAGES:

- Smart Agriculture requires internet connectivity continuously, but rural parts cannot fulfil this requirement.
- Any faults in the sensors can cause great loss in the agriculture, due to wrong records and the actions of automated processes.
- IOT devices need much money to implement.

11. CONCLUSION:

An IOT based smart agriculture system using Watson IOT platform, Watson simulator, IBM cloud and Node-RED.

12. FUTURE SCOPE:

In future due to more demand of good and more farming in less time, for betterment of the crops and reducing the usage of extravagant resources like electricity and water IOT can be implemented in most of the places

13. APPENDIX:

SOURCE CODE:

```
import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig = {
"identity": {
"orgId": "u9qhfi",
"typeId": "Devicetypel",
"deviceId": "DeviceID1"
},
"auth": {
"token": ")hSb7_ZD+evl2fRhXi"
} }
client = wiotp.sdk.device.DeviceClient
(config=myConfig,logHandlers=None)
client.connect ()
def myCommandCallback (cmd) :
    print ("Message received from IBM IoT Platform: %s" %
cmd.data['command'])
    m=cmd.data['command']
    if (m=="motoron"):
        print ("Motor is switched on")
    elif (m=="motoroff"):
        print ("Motor is switched OFF")
    print (" ")
while True:
    soil=random.randint (0,100)
    temp=random.randint (-20, 125)
    hum=random.randint (0, 100)
    myData={'soil moisture': soil, 'temperature':temp, 'humidity':hum}
    client.publishEvent (eventId="status", msgFormat="json", data=myData,
```

```
qos=0 , onPublish=None)
    print ("Published data Successfully: %s", myData)
    time.sleep (2)
client.commandCallback = myCommandCallback
client.disconnect ()
```

<u>OUTPUT:</u>

GITHUB LINK: **https://github.com/IBM-EPBL/IBM-Project-26514-1660028804**

PROJECT DEMO: IN GITHUB LINK

THANK YOU…..