

PROJECT REPORT

IOT BASED SMART CROP PROTECTION SYSTEM

FOR AGRICULTURE

TEAM ID: PNT2022TMID21131

| ROLL NUMBER | NAME |
|--------------------|-------------------|
| 714019205002 | ABIRAMI PAVISYA S |
| 714019205021 | JEEVIKAA SHRI G N |
| 714019205030 | NARMATA CR |
| 714019205042 | SNEHA M |

CONTENT

| SNO | TOPIC | PAGE NO |
|-----|--------------------------------------|---------|
| 1. | Introduction | 1 |
| 2. | Problem Statement | 2 |
| 3. | Proposed Solution | 3 |
| 4. | Theoretical Analysis | 3 |
| | 4.1 System Architecture | 3 |
| | 4.2 Software Requirements | 4 |
| | 4.21 Node-Red Services | 4 |
| | 4.22 IBM Watson IOT Platform | 5 |
| | 4.23 Python Ide | 5 |
| | 4.24 IBM Cloud ant | 6 |
| | 4.3 MIT-App Simulator | 6 |
| | 4.4 Open Weather API | 7 |
| 5. | Implementation Of System | 8 |
| | 5.1 IBM Watson IOT Device Deployment | 8 |
| | 5.2 Deployment Of Node-Red Services | 8 |
| | 5.3 Node-Red Connectivity to Python | 9 |
| | 5.4 Deployment Of MIT-App Simulator | 11 |
| 6. | Observations And Results | 12 |
| 7. | Advantages And Disadvantages | 15 |
| 8. | Conclusion | 16 |
| 9. | Reference | 17 |
| 10. | Appendix | 18 |

ABSTRACT

IOT Based Smart Crop-Protection for Agriculture monitoring is a system describes how to monitor crop field. It is developed by using sensors and according to the decision from a server based on sensed data, the irrigation and monitoring system is enhanced. Through wireless transmission the sensed data is forwarded to web server database. If the irrigation is automated, then the moisture and temperature fields are decreased below the potential range. The user can monitor and control the system remotely with the help of application which provides a web interface to user. By smart Agriculture monitoring system and one of the oldest ways in agriculture is the manual method of checking the parameters. In this method farmers by themselves verify all the parameter and calculate the reading. It aims at making agriculture smart using automation and IoT. The cloud computing devices are used at the end of the system that can create a whole computingsystem from sensors to tools that observe data from agriculture field. It proposes a novel methodology for smart farming by including a smart sensing system and smart irrigator system through wireless communication technology. This system is cheap at cost for installation. Here one can access and control the agriculture system in laptop, cell phone or a computer.

LIST OF FIGURES

| FIG. NO | NAME | PAGE NO |
|---------|--------------------------------|---------|
| 1 | NODE-RED DASHBOARD | 3 |
| 2 | DEVICE SIMULATION | 4 |
| 3 | PYTHON COMMAND PROMPT | 5 |
| 4 | CLOUD ANT DB | 5 |
| 5 | MIT-APP SIMULATOR | 6 |
| 6 | WEATHER API | 6 |
| 7 | IBM WATSON IOT CREDENTIALS | 7 |
| 8 | NODE-RED TO WATSON IOT | 8 |
| 9 | NODE-RED AND CLOUDANT CONNECT | 8 |
| 10 | TEMPERATURE READING | 8 |
| 11 | HUMIDITY READING | 8 |
| 12 | MOTOR READING | 9 |
| 13 | NODE-RED FLOW EDITOR | 9 |
| 14 | HTTP CONNECTION TO MIT-APP | 10 |
| 15 | SENSOR DATA DISPLAY IOT WATSON | 13 |
| 16 | NODE-RED WEB APPLICATION | 13 |
| 17 | HOME SCREEN | 14 |
| 18 | CONTROLLER SCREEN | 14 |
| 19 | DATA SCREEN | 14 |

Chapter – 1

INTRODUCTION

A system using sensors that monitor different conditions of environment like humidity, temperature etc., the processor and GUI module is used. The field condition is sent to the farmer via mobile text messages. With this system Soil moisture, humidity and energy efficiency are managed. A system is proposed for intelligent agriculture monitoring system based on IOT technology. The main aim of this project is to help farmers to automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature, humidity etc. and control the equipment like water motor and other devices remotely via internet without their actual presence in the field.

Need for Smart Framing

1. Simple, easy to install and configure.
2. Save energy and resources, so that they can be used efficiently and effectively.
3. Farmers will be able to apply the right amount of water at the right time by automatic irrigation.
4. To avoid irrigation at the wrong time of day, reduce the flow of excess waterlogged soil
5. improve plant performance.
6. The automatic irrigation system uses vales to turn on and off the engine.
Motors can be automated using IOT
7. Controls and no work need to shut down the engine.
8. It is an accurate irrigation system and an important tool for controlling soil moisture in a very special way
9. heat vegetable production.
10. Timesaving, eliminating human error in adjusting available soil moisture levels.

Chapter-2

PROBLEM STATEMENT

- Farmers are to be present at farm for maintenance irrespective of the weather conditions.
- They must ensure that the crops are well irrigated, and the routine activities of the field must be monitored by them physically.
- To get good, cultivate or yield farmers need to stay in the field for longer time for good yield
- Demand and supply are more if the field is to be monitored monotonously if it covers vast area
- Anytime crops may prone to various calamities which leads to poor yield of the crops

And these are some cons that is been faced by the farmers and which leads them to cultivate crops for less yield despite heavy work

The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature, humidity etc. and control the equipment like water motor and other devices remotely via internet without their actual presence in the field.

Chapter-3

PROPOSED SOLUTION

To reduce the cost-effective work, expenses and to work in an effective way, we introduce IoT services in which we use cloud services and internet to enable farmer to continue their work remotely via through internet. And they can also be able to monitor the field parameters and control the devices in farm to harvest higher yield with less consumption of manpower. The aim of the project is to protect fields to take account temperature and humidity of the fields to reduce the farming expenses and to save energy. Overall, the design realizes remote intelligent monitoring and control of fields and replaces the traditional wired technology to wireless, also reduces manpower cost.

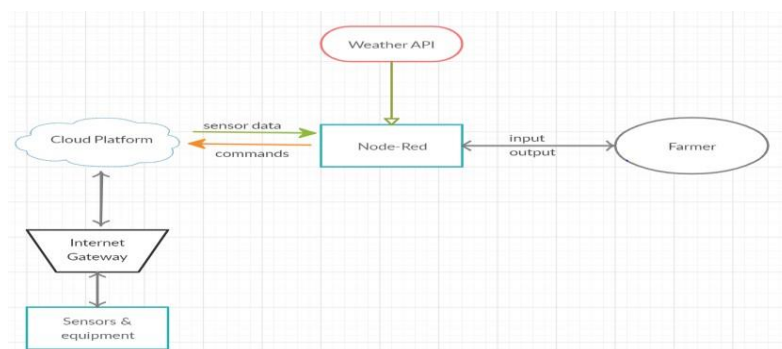
Chapter-4

THEORETICAL ANALYSIS

In this theoretical analysis we can be able to get detailed overview about the project and its implementation, deployment of the software used in each phase of the project

4.1 SYSTEM ARCHITECTURE

It depicts the flow of the project from initiation to the final phase of the project.



4.2 SOFTWARE REQUIREMENTS

The web-based application is deployed by consolidating the essential requirements to build IOT web-based platform. The following credential are needed to deploy the web application using NODE-RED which are stated as below

4.21 NODE-RED SERVICES

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. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.

Steps involved in configuring the Node-Red Services

1. Create an IBM cloud account
2. Select catalog → Node-Red Application
3. Give the required credentials to access the Node-Red like cloud foundry, Resource list and so on
4. Once you have finished the process just click on the URL to deploy the Node-Red dashboard

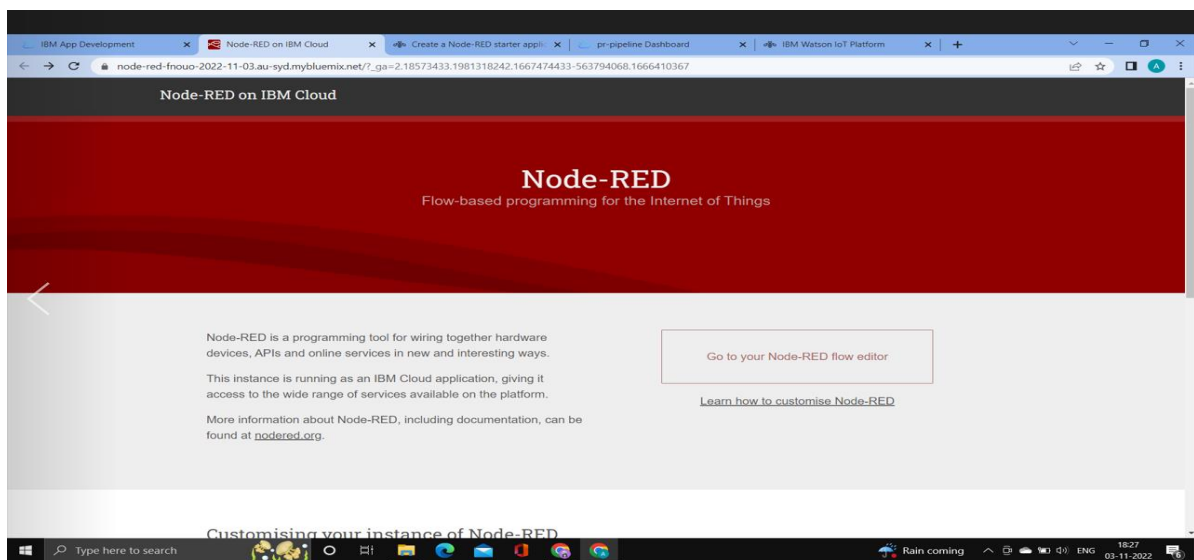


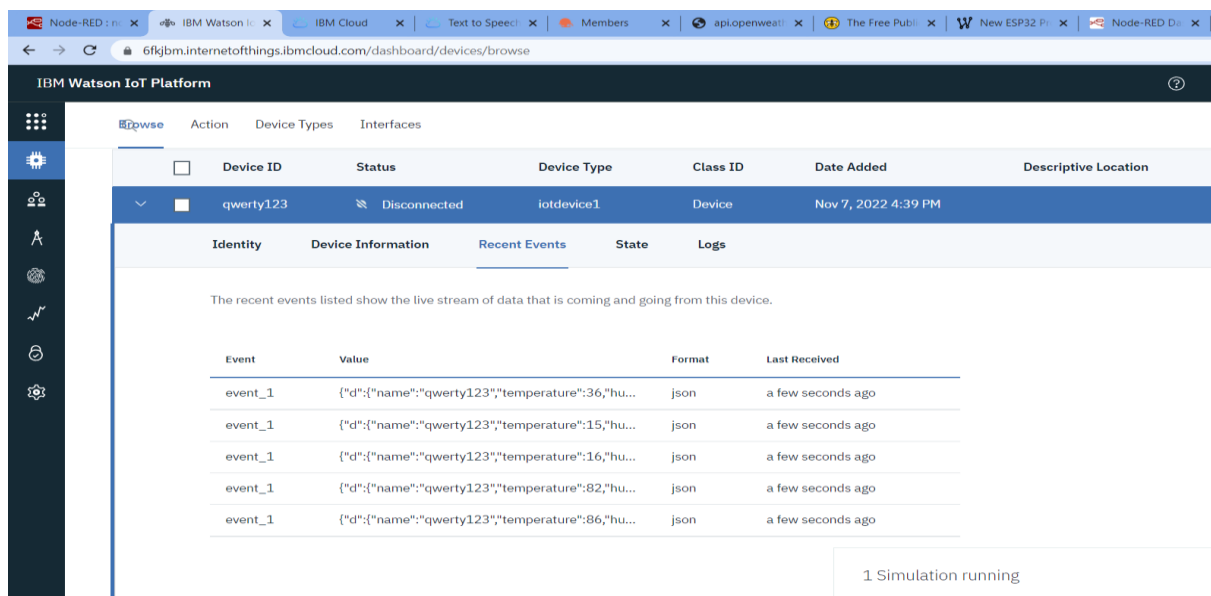
FIG-1 NODE-RED DASHBOARD

4.22 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 involved in configuring the IBM-Watson

1. Create an IBM cloud account
2. Select catalog → IBM-Watson IOT
3. Give the required credentials to access the IBM-Watson like cloud foundry, Resource list and so on
4. Once you have finished the process just click on the LAUNCH to deploy the IBM Watson and Sign into it
5. Under the IBM Watson click on “Bluemixfree” and you will direct to the IBM Watson-IOT where you can deploy your device



| Device ID | Status | Device Type | Class ID | Date Added | Descriptive Location |
|-----------|--------------|-------------|----------|---------------------|----------------------|
| qwerty123 | Disconnected | iotdevice1 | Device | Nov 7, 2022 4:39 PM | |

| Event | Value | Format | Last Received |
|---------|---|--------|-------------------|
| event_1 | {"d":{"name":"qwerty123","temperature":36,"hu..." | json | a few seconds ago |
| event_1 | {"d":{"name":"qwerty123","temperature":15,"hu..." | json | a few seconds ago |
| event_1 | {"d":{"name":"qwerty123","temperature":16,"hu..." | json | a few seconds ago |
| event_1 | {"d":{"name":"qwerty123","temperature":82,"hu..." | json | a few seconds ago |
| event_1 | {"d":{"name":"qwerty123","temperature":86,"hu..." | json | a few seconds ago |

1 Simulation running

FIG-2 DEVICE SIMULATION

4.23 Python Idle

Deployment of IOT platform is also initiated by the python interpreter that enables IOT platform to connect with devices. Few packages need to be installed to work in python interpreter so that it can be able to traverse between simulator and Node-Red many other services

```

C:\Windows\system32\cmd.exe
C:\Users\HP>pip install ibm-cos-sdk
Collecting ibm-cos-sdk
  Using cached ibm-cos-sdk-2.12.0.tar.gz (55 kB)
  Preparing metadata (setup.py) ... done
Collecting ibm-cos-sdk-core==2.12.0
  Downloading ibm-cos-sdk-core-2.12.0.tar.gz (956 kB)
    ----- 956.0/956.8 kB 44.9 kB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting ibm-cos-sdk-s3transfer==2.12.0
  Downloading ibm-cos-sdk-s3transfer-2.12.0.tar.gz (135 kB)
    ----- 135.7/135.7 kB 47.8 kB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting jmespath<1.0.0,>=0.10.0
  Downloading jmespath-0.10.0-py2.py3-none-any.whl (24 kB)
Collecting python-dateutil<3.0.0,>=2.8.2
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
    ----- 247.7/247.7 kB 65.0 kB/s eta 0:00:00
Collecting requests<3.0,>=2.27.1
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
    ----- 62.0/62.8 kB 26.1 kB/s eta 0:00:00
Collecting urllib3<1.27,>=1.26.9
  Downloading urllib3-1.26.12-py2.py3-none-any.whl (140 kB)
    ----- 140.0/140.4 kB 54.1 kB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Collecting charset-normalizer<3,>=2
  Downloading charset_normalizer-2.1.1-py3-none-any.whl (39 kB)
Collecting certifi<2017.4.17
  Downloading certifi-2022.9.24-py3-none-any.whl (161 kB)
    ----- 161.1/161.1 kB 94.7 kB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
    ----- 61.5/61.5 kB 84.2 kB/s eta 0:00:00
Building wheels for collected packages: ibm-cos-sdk, ibm-cos-sdk-core, ibm-cos-sdk-s3transfer
  Building wheel for ibm-cos-sdk (setup.py) ... done
  Created wheel for ibm-cos-sdk: filename=ibm_cos_sdk-2.12.0-py3-none-any.whl size=73910 sha256=9c7e17265f10
  Stored in directory: c:\users\hp\appdata\local\pip\cache\wheels\53\89\f8\ef16b3874af57e2253ee18a43804db68
  Building wheel for ibm-cos-sdk-core (setup.py) ... done
  Created wheel for ibm-cos-sdk-core: filename=ibm_cos_sdk_core-2.12.0-py3-none-any.whl size=562947 sha256=c
  Stored in directory: c:\users\hp\appdata\local\pip\cache\wheels\72\24\c4\b930b91571fe120c206726f2055a26c95
  Building wheel for ibm-cos-sdk-s3transfer (setup.py) ... done
  Created wheel for ibm-cos-sdk-s3transfer: filename=ibm_cos_sdk_s3transfer-2.12.0-py3-none-any.whl size=897
  Stored in directory: c:\users\hp\appdata\local\pip\cache\wheels\3f\bfb6\4c18868ed7b686342eab42be15f0c0ac
Successfully built ibm-cos-sdk ibm-cos-sdk-core ibm-cos-sdk-s3transfer
Installing collected packages: urllib3, six, jmespath, idna, charset-normalizer, certifi, requests, python-d
Successfully installed certifi-2022.9.24 charset-normalizer-2.1.1 ibm-cos-sdk-2.12.0 ibm-cos-sdk-core-2.12.0
urllib3-1.26.12
C:\Users\HP>

```

FIG-3 PYTHON COMMAND PROMPT

4.24 IBM Cloud ant

In order to store the NODE-RED value or any service provider we need to have a Database here the cloudant database is created inorder to store the values of the sensor reading of the devices

| Name | Size | # of Docs | Partitioned | Actions |
|----------------------|---------|-----------|-------------|--------------------------------|
| noderedfnouo20221103 | 36.7 KB | 4 | No | [Icons for edit, delete, etc.] |
| sample | 0 bytes | 0 | Yes | [Icons for edit, delete, etc.] |
| sample1 | 0.9 MB | 4076 | No | [Icons for edit, delete, etc.] |

FIG-4 CLOUDANT DB

4.3 MIT-APP SIMULATOR

Simulation is carried on by using MIT-App inventor for Node-Red web application we have created a simulator with three modules they are:

- HOME SCREEN
- CONTROLLER SCREEN
- DATA SCREEN

There will be a designer and block panel separately and features and styles are being done in designer panel where we use to drag and drop desired icons.

Similarly block panel here we give functionality to the features which are done by drag and drop method

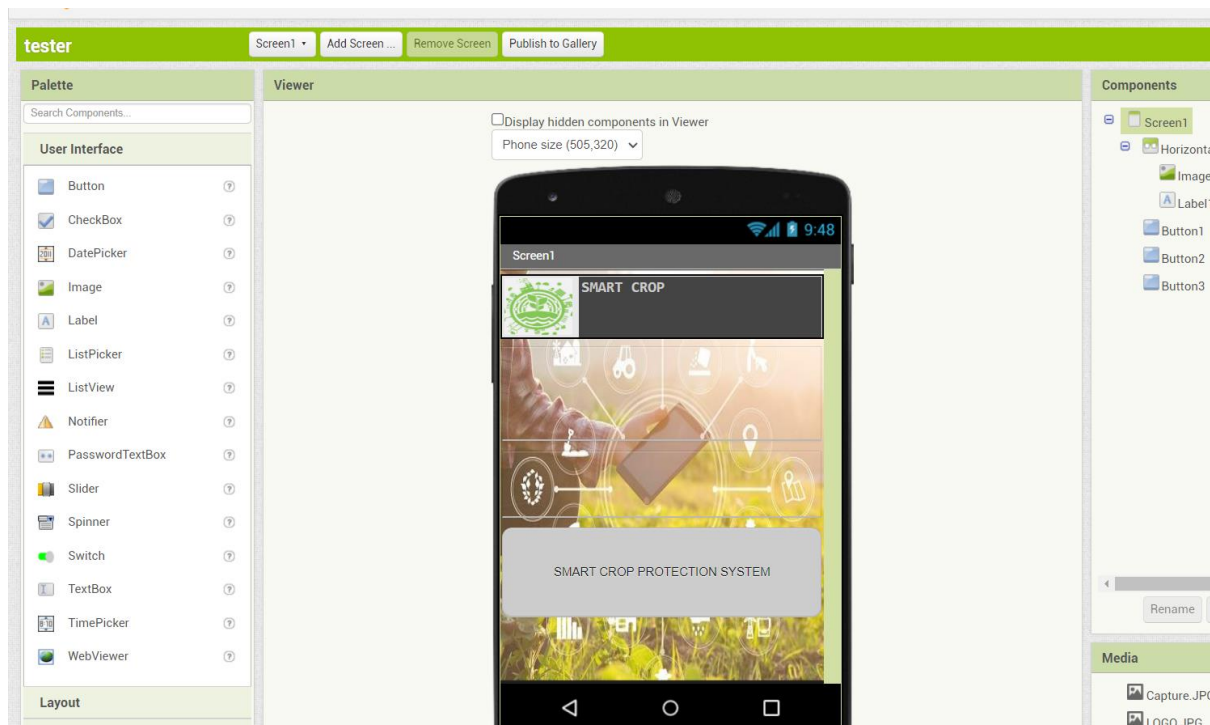


FIG-5 MIT-APP SIMULATOR

4.4 Open Weather API

It is an online service that provides weather data. And to configure we need to create account in Open Weather and find the name of our city by searching then create API key to our account after that replace “city name” and “API key” with your city and API key

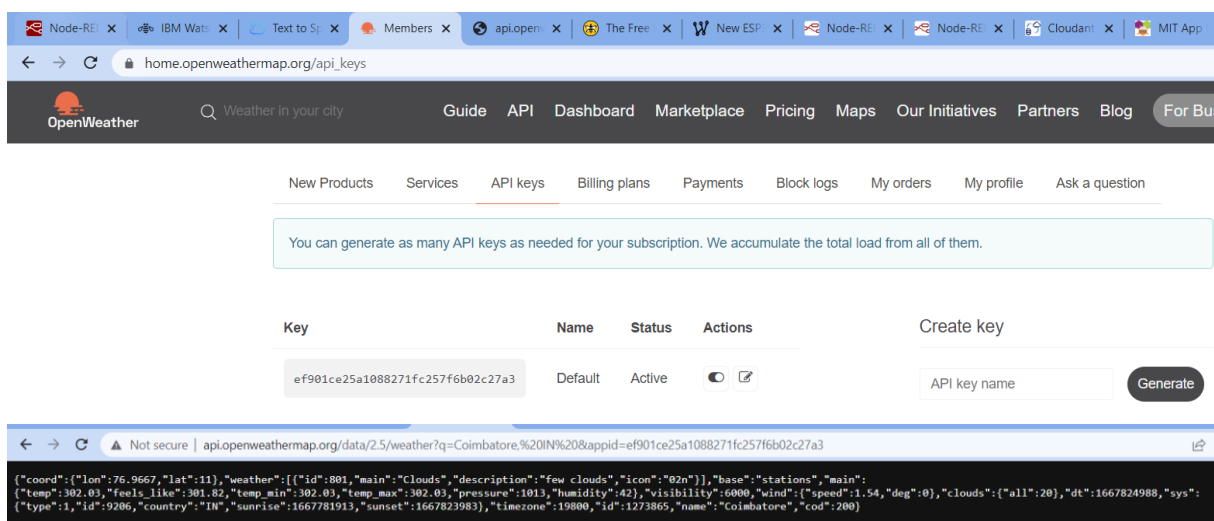


FIG – 6 WEATHER API

Chapter-5

IMPLEMENTATION OF THE SYSTEM

5.1 IBM Watson IoT Device Deployment

To deploy IBM-Watson the following credentials are important so that we can be able to connect our device to desired platform

1. OrgID
2. DeviceID
3. Device Type
4. Token
5. API

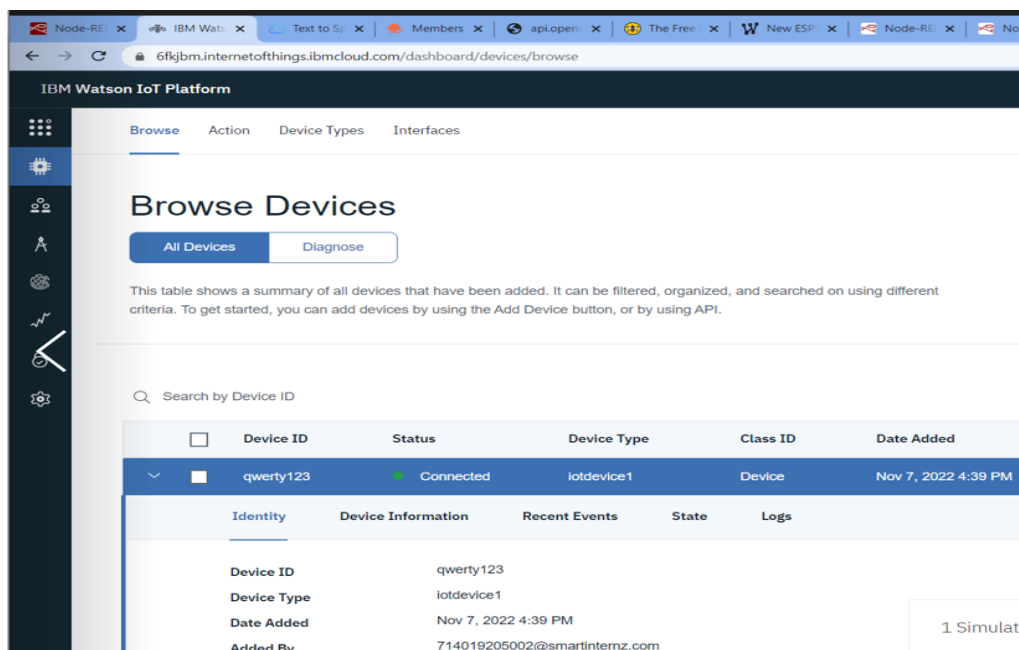


FIG-7 IBM WATSON IOT CREDENTIALS

5.2 Deployment Of Node-Red Services

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

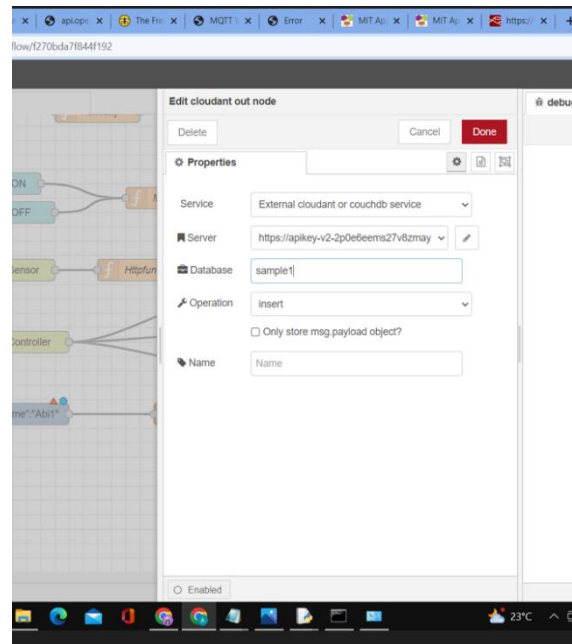
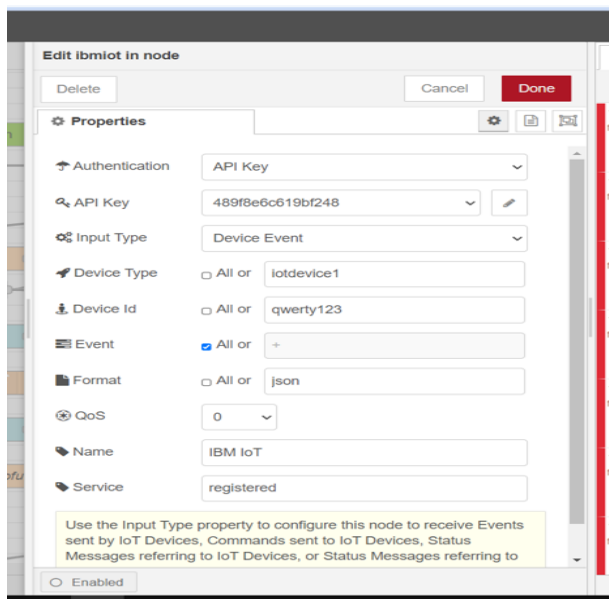


FIG-8 NODE-RED TO IOT WATSON FIG-9 NODE-RED TO CLOUDANT CONNECTIVITY

Once it is connected Node-Red receives data from the deviceDisplay the data using debug node for verification Then 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;
msg.payload = msg.payload.d.humidity;
return msg;
```

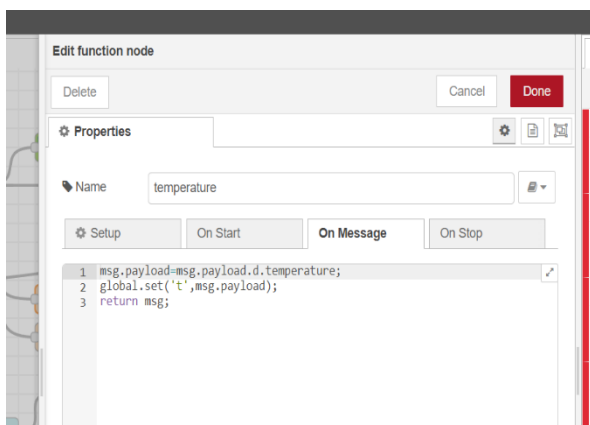


FIG-10 TEMPERATURE READING

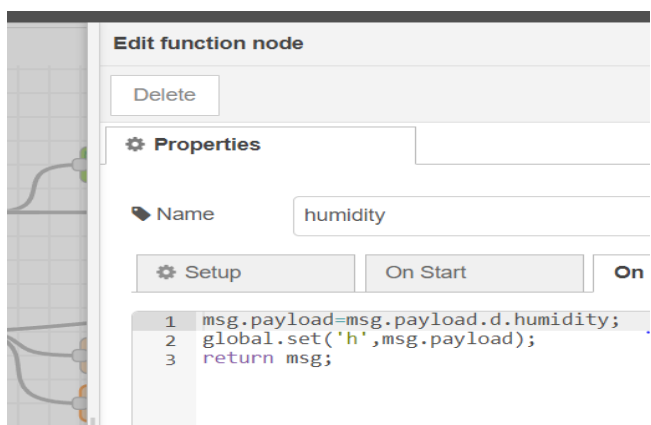


FIG -11 HUMIDITY READING

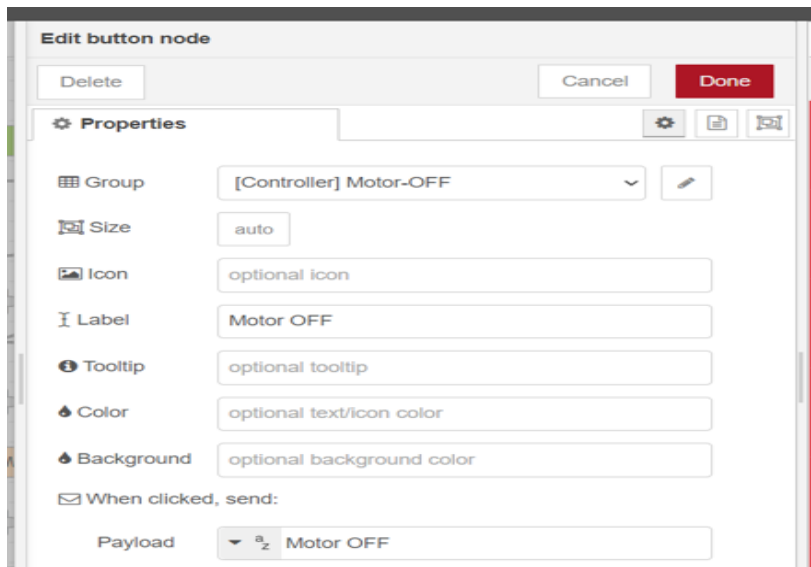


FIG -12 MOTOR READING

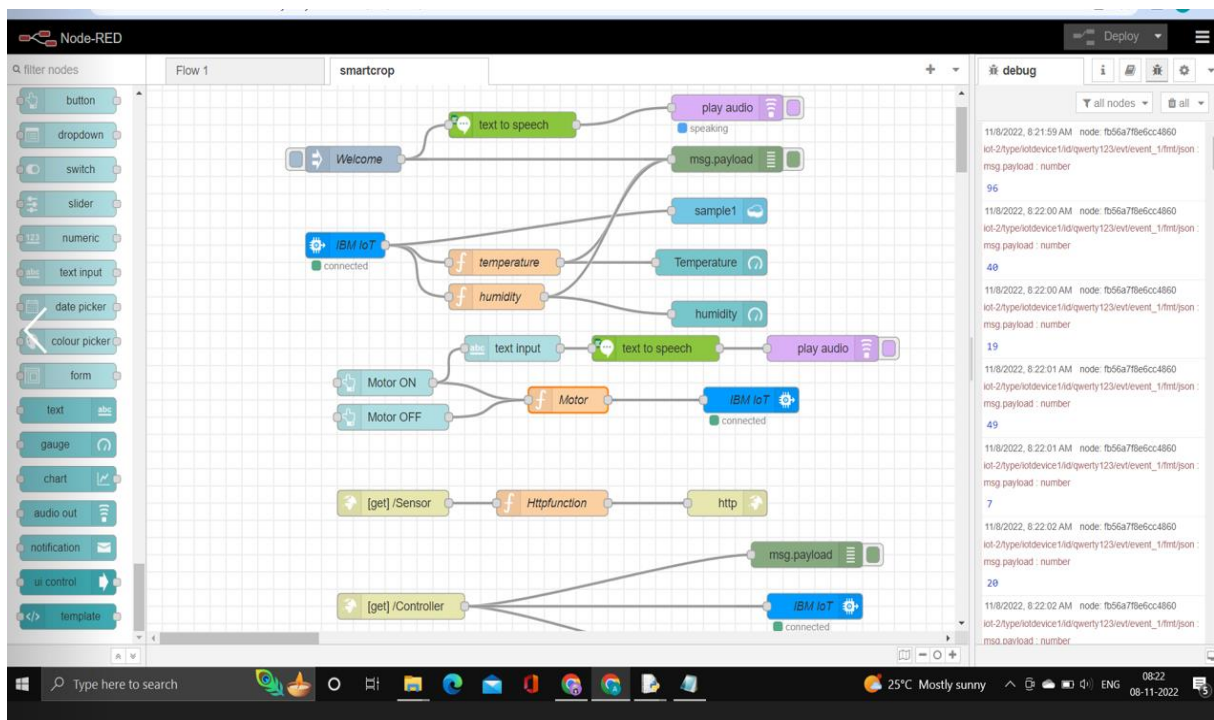


FIG- 13 NODE-RED FLOW EDITOR WITH TEXT TO SPEECH

5.3 Deployment Of MIT-App Simulator

MIT-APP Inventor has been deployed with the node red to render sensor data from the NODE-Red o the MIT-APP simulator

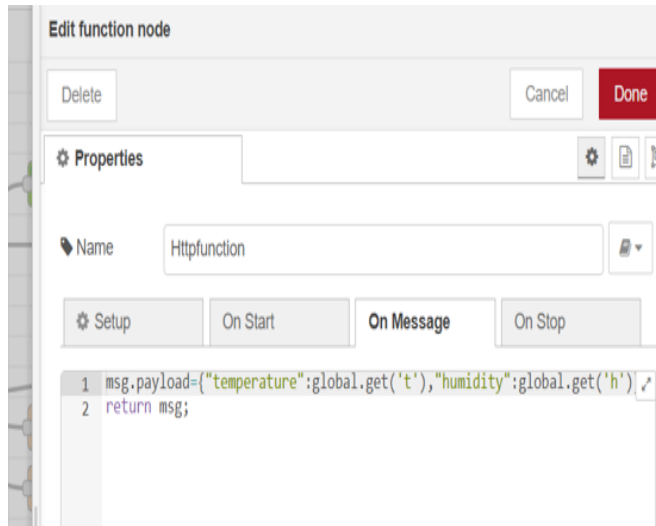
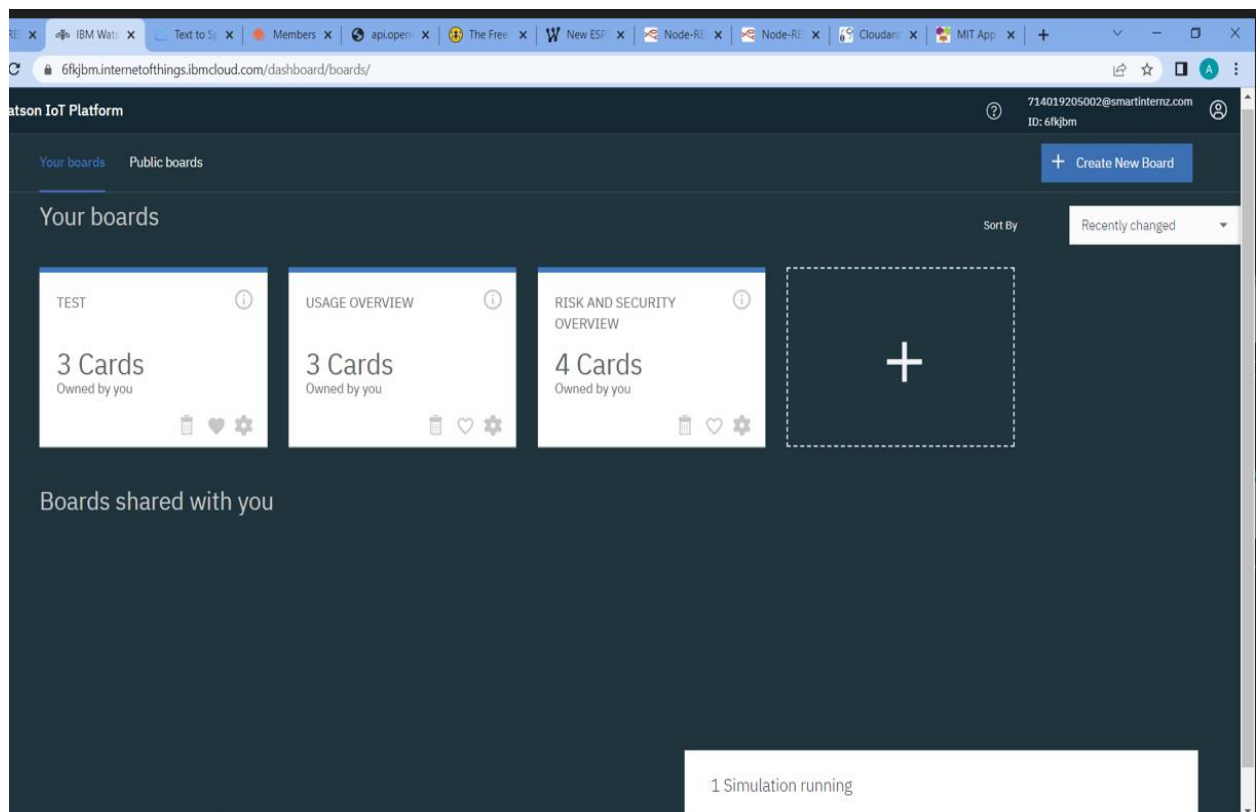
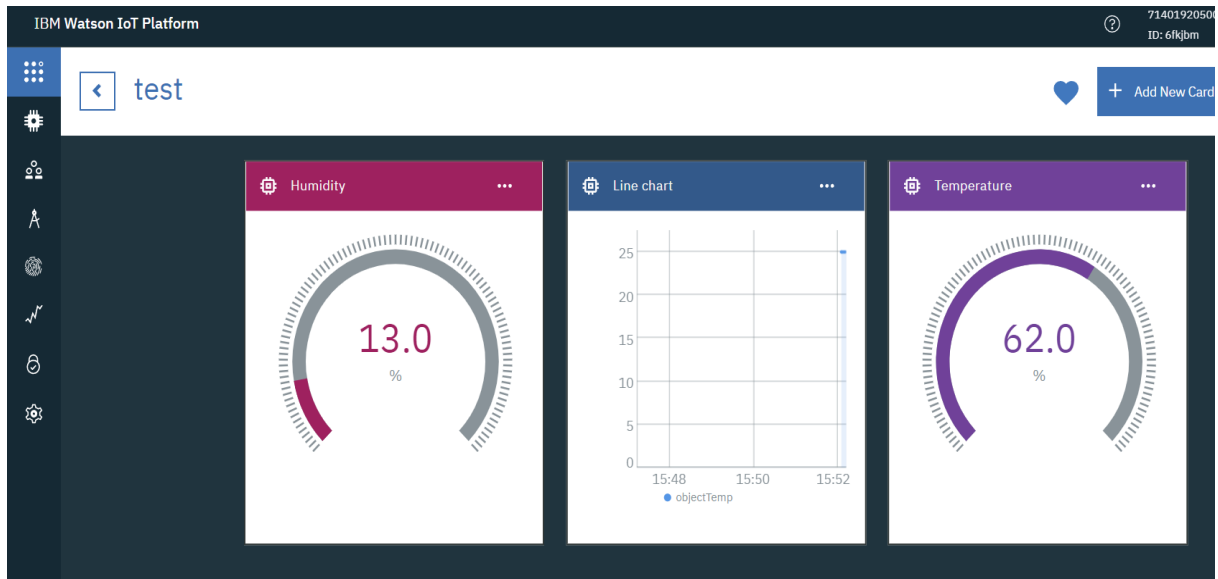


FIG – 14 HTTP CONNECTION TO MIT-APP

Chapter – 6

OBSERVATIONS AND RESULTS

Output Of Web-Based Application Using Node-Red are depicted as follows



| Device ID | Status | Device Type | Class ID | Date Added |
|-----------|-----------|-------------|----------|---------------------|
| qwerty123 | Connected | iotdevice1 | Device | Nov 7, 2022 4:39 PM |

| Identity | Device Information | Recent Events | State | Logs |
|-------------|-------------------------------|---------------|-------|------|
| Device ID | qwerty123 | | | |
| Device Type | iotdevice1 | | | |
| Date Added | Nov 7, 2022 4:39 PM | | | |
| Added By | 714019205002@smartinternz.com | | | |

FIG -15 Sensor Data Displayed In Device Boards As Separate Cards

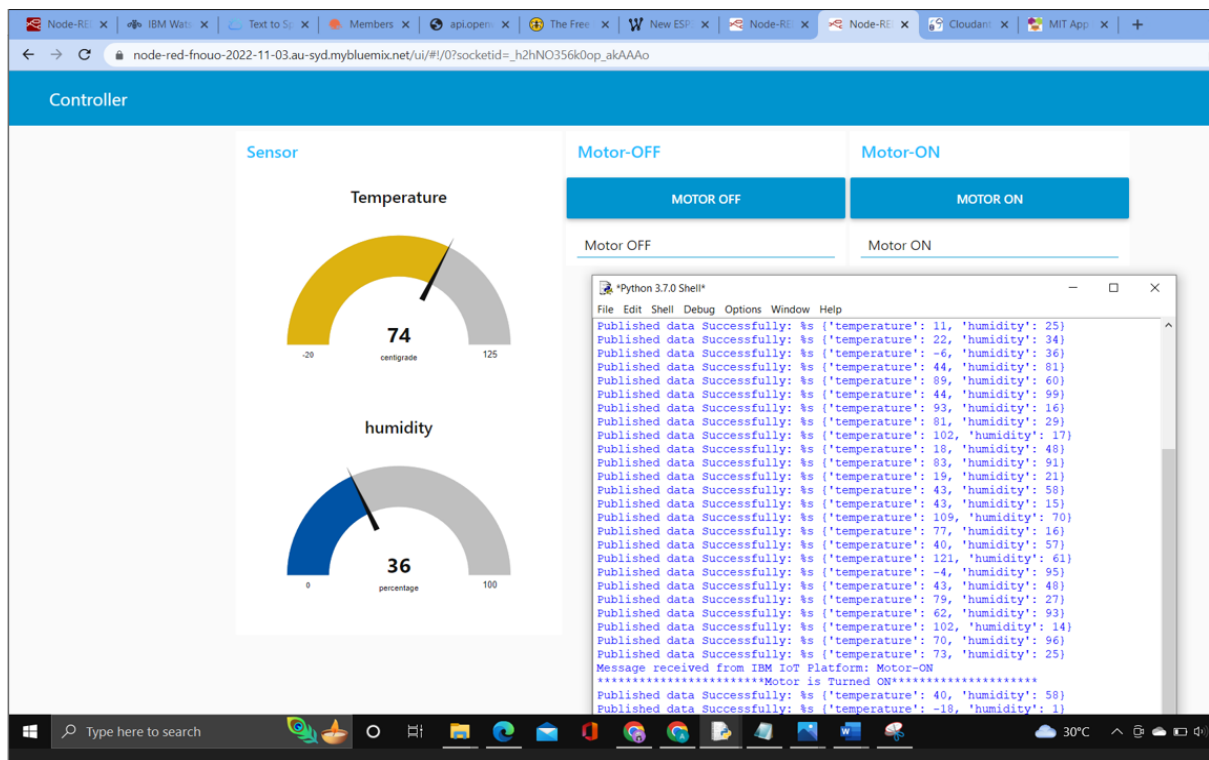


FIG-16 NODE-RED WEB GUI

SIMULATOR DEPLOYMENT OUTPUT



FIG-17 HOME SCREEN

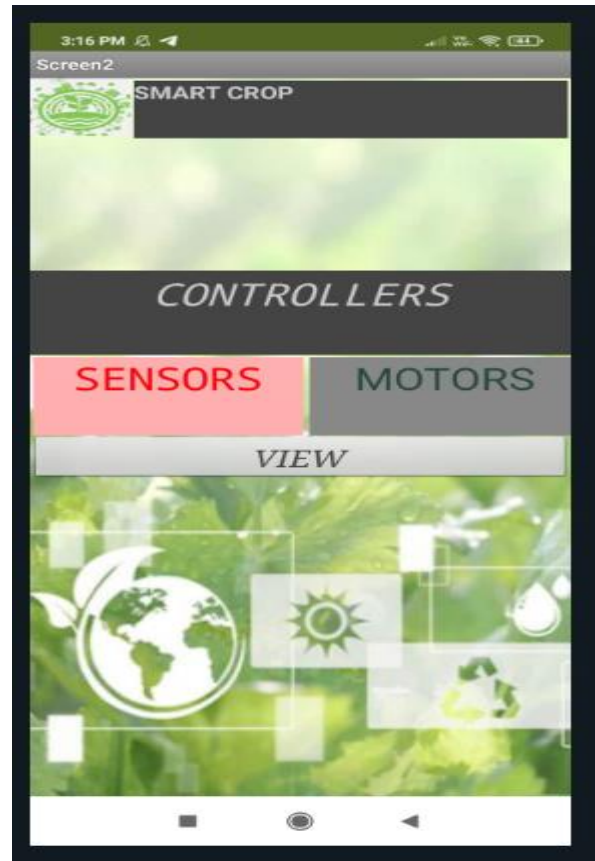


FIG-18 CONTROLLER SCREEN

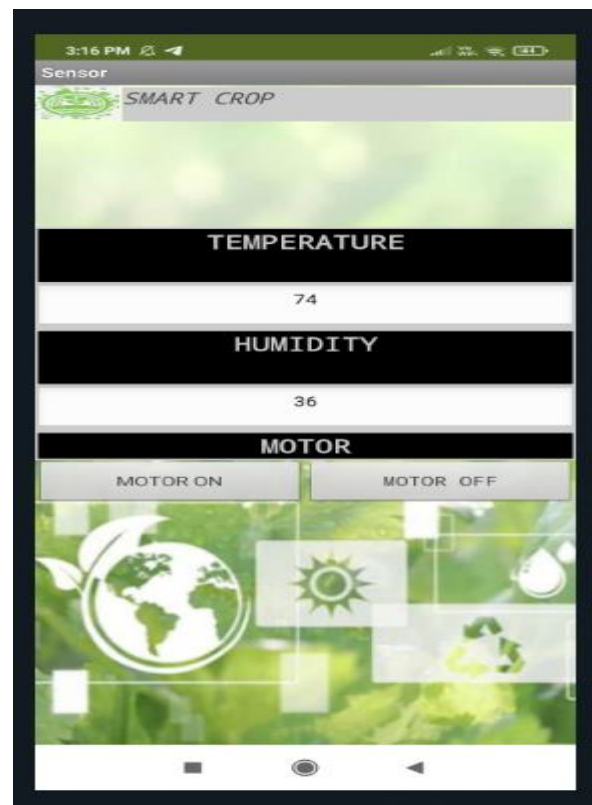


FIG-19 DATA SCREEN

Chapter- 7

ADVANTAGES AND DISADVANTAGES

The use of technology in farming and agriculture making it smart agriculture, is of course, a good initiative and a much needed one with the present increasing demand in the food supply. But there is the chance where smart farming will require certain skill sets to understand and operate the equipment. In the case of equipment like robots and computer-based intelligence for running the devices, it is highly unlikely that a normal farmer will be able to possess this knowledge or even develop them. Farmers are not used to these high-end technologies. They do not understand computer language or the artificial intelligence. For the smart agriculture, Internet of Things is essential which will require artificial intelligence and computer-based intelligence. This cannot be balanced here.

Advantages:

- ☐ Farms can be monitored and controlled remotely.
- ☐ Increase in convenience to farmers.
- ☐ Less Manpower.
- ☐ 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.

Chapter- 8

CONCLUSION

This system focuses on developing devices and tool to manage, display and alert the users using the advantages of a wireless sensor network system. It aims at making agriculture smart using automation and IoT. The cloud computing devices are used at the end of the system that can create a whole computing system from sensors to tools that observe data from agriculture field. It proposes a novel methodology for smart farming by including a smart sensing system and smart irrigator system through wireless communication technology. Thus, the objective of the project to implement an IoT system to help farmers to control and monitor their farms has been implemented successfully.

Mostly IoT in agriculture uses robots, drones, remote sensors, and computer imaging combined with continuously progressing machine learning and analytical tools for monitoring crops, surveying, and mapping the fields, and providing data to farmers for rational farm management plans to save both time and money.

Chapter- 9

REFERENCE

1. Balaji Bhanu, Raghava Rao, J.V.N. Ramesh and Mohammed Ali hussain, “Agriculture Field Monitoring and Analysis using Wireless Sensor Networks for improving Crop Production”, Eleventh International Conference on Wireless and Optical Communications Networks (WOCN).2014.
2. Harshal Meharkure, Pargyline, Sheetal Israni, “Application of IOT Based System for Advance Agriculture in India”, International Journal of Innovative Research in Computer and Communication Engineering(IJRCCE) Vol.
3. Issue 11, pp. 10831-10837, 2015. 3. LIU Dan, Cao Xin, Huang Chongwei, JI Liang Liang, “Intelligent agent greenhouse environment monitoring system based on IOT technology”, International Conference on Intelligent Transportation, Big Data & Smart City, 2015.
4. Mehdi Roopei, Paul Rad, Kim-Kwang Raymond Choo, “Cloud of Things in smart agriculture: Intelligent irrigation monitoring by Thermal Imaging” IEEE Cloud Computing, 2017.
5. IBM cloud reference: <https://cloud.ibm.com/>
6. Python code reference: <https://github.com/rachuriharish23/ibmsubscribe>
7. IoT simulator : <https://watson-iot-sensor-simulator.mybluemix.net/>

Chapter- 10

APPENDIX

CODE FOR PYTHON EDITOR

```
#IBM Watson IOT Platform
import wiotp.sdk.device
import time
import random
myConfig = {
    "identity": {
        "orgId": "6fkjbm",
        "typeId": "iotdevice1",
        "deviceId": "qwerty123"
    },
    "auth": {
        "token": "johnyjohnnyespapa"
    }
}

def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="Motor-ON"):
        print("*****Motor is Turned ON*****")
    else:
        print("*****Motor is Turned OFF*****")
client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

while True:
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'temperature':temp, 'humidity':hum}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
onPublish=None)
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
```

IBM TEXT TO SPEECH

```
from ibm_watson import TextToSpeechV1
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
authenticator =
IAMAuthenticator('M_u6yEvEGJylj_ysbL_pG0ZOKuRCQW1LgXUtv_IcBPCR')
text_to_speech = TextToSpeechV1(
    authenticator=authenticator
```

```

)
text_to_speech.set_service_url('https://api.au-syd.text-to-
speech.watson.cloud.ibm.com/instances/23724eb6-a096-4a3a-b914-da0e442c1c5f')
with open('hello_world.wav', 'wb') as audio_file:
    audio_file.write(
        text_to_speech.synthesize(
            'Alert',
            voice='en-US_AllisonV3Voice',
            accept='audio/wav'
        ).get_result().content)

```

CODE FOR MIT-APP INVENTOR (SCREEN 3)

