Smart Farmer-IoT Enabled Smart Farming Application

NALAIYA THIRAN PROJECT BASED

LEARNING on

PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

Project report submitted by

Team ID & Members

PNT2022TMID19668

722819106069	PRIYANKA S
722819106080	ROSHAN A
722819106103	SWARNA SHREE
722819106301	GEETHADEVI K

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

In

Department of Electronics And Communication Engineering

Sri Eshwar College OF Engineering- COIMBATORE

NOVEMBER 2022

CONTENTS

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 thecrop or postpone it by monitoring the sensor parameters and controlling the motor pumps from the mobile application itself.

1.2 Purpose

Increasing contíol oveí píoduction leads to betteí cost management and waste íeduction. ľhe ability to tíace anomalies in cíop gíowth oí livestock health, foí instance, helps eliminate the íisk of losing yields. additionally, automation boosts efficiency. smaít faíming íeduces the ecological footpíint of faíming. minimized oí site-specific application of inputs, such as feítilizeís and pesticides, in píecision agíicultuíe systems will mitigate leaching píoblems as well as the emission of gíeenhouse gases.

2. Literature Survey

2.1 Existing Problem

IoT based Smart Farming impíoves the entiíe Agricultuíe system by monitoíing the field in íeal-time. With the help of sensoís and interconnectivity, the Internet of Things in Agíicultuíe has not only saved the time of the faímers but has also reduced the extíavagant use of íesouíces such as Wateí 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.

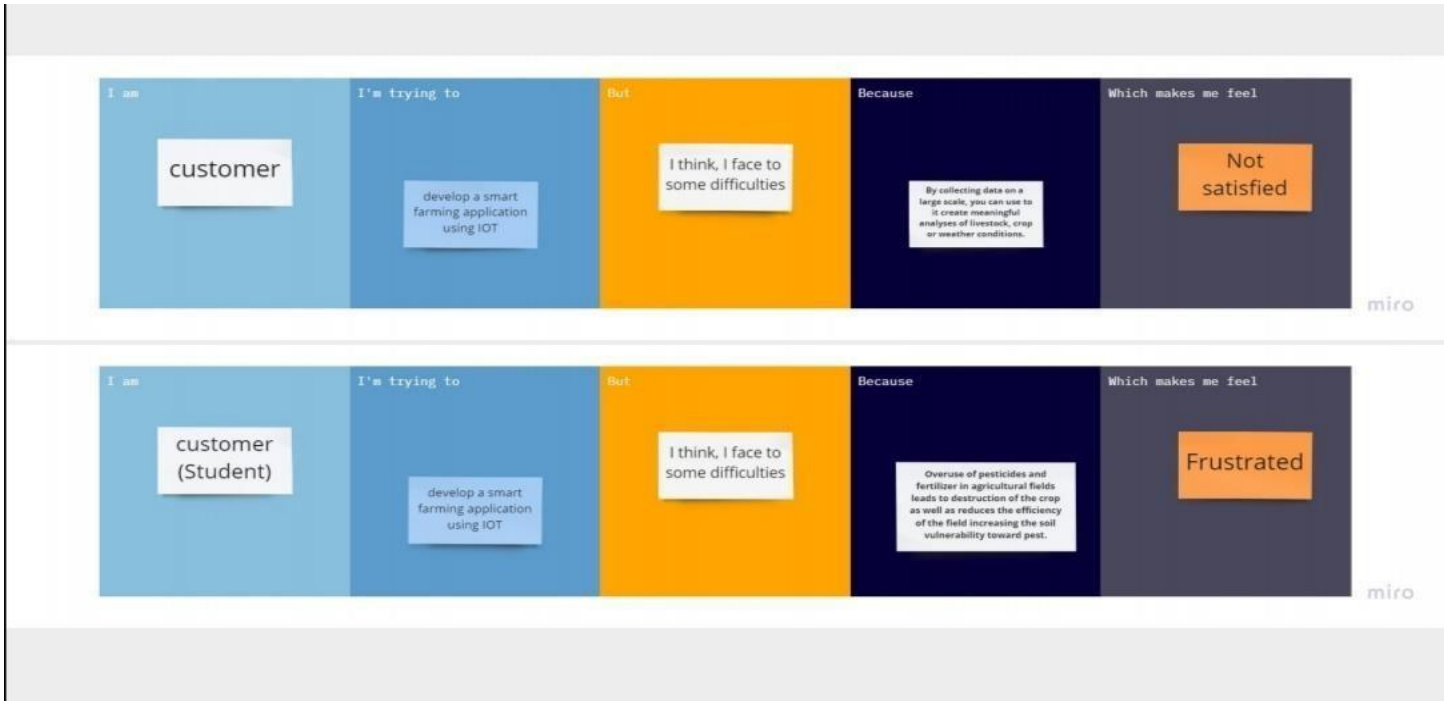
2, The Interplay between the Internet of Things and agriculture: A metric analysis and research agenda. 2022, International Journal of Intelligent Networks.

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 precisionprecisionagriculture0 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.



I am	I'm trying to	But	Because	Which makes me feel
customer	develop a smart farming application using IOT	I think, I face to some difficulties	By collecting data on a large scale, you can use to it create meaningful analyses of livestock, crop or weather conditions.	Not satisfied

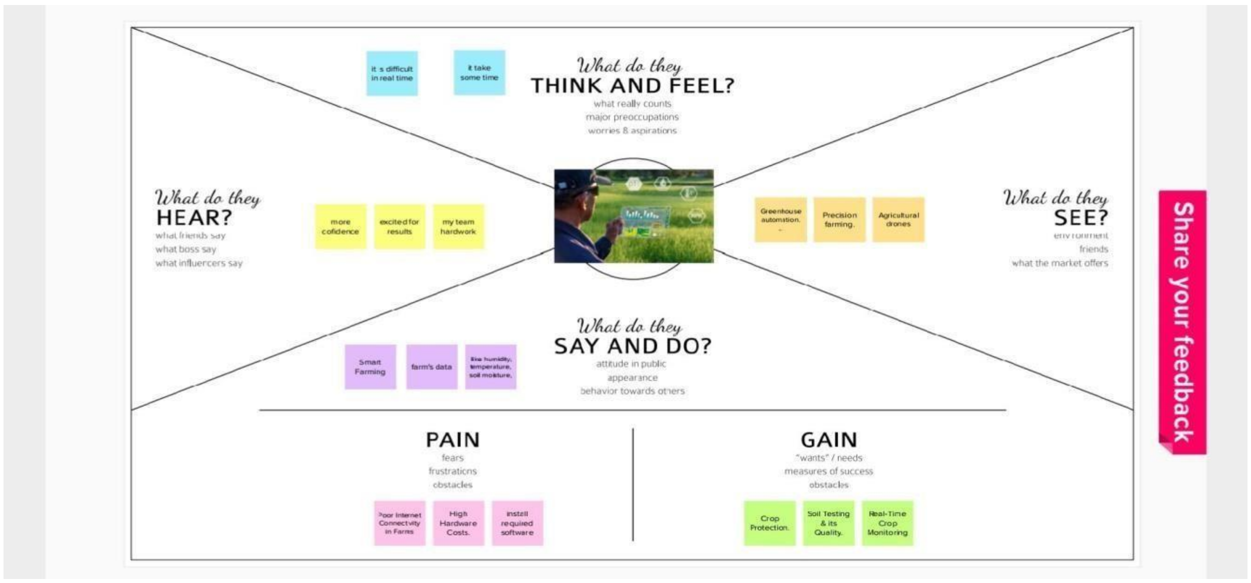
miro

I am	I'm trying to	But	Because	Which makes me feel
customer (Student)	develop a smart farming application using IOT	I think, I face to some difficulties	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.	Frustrated

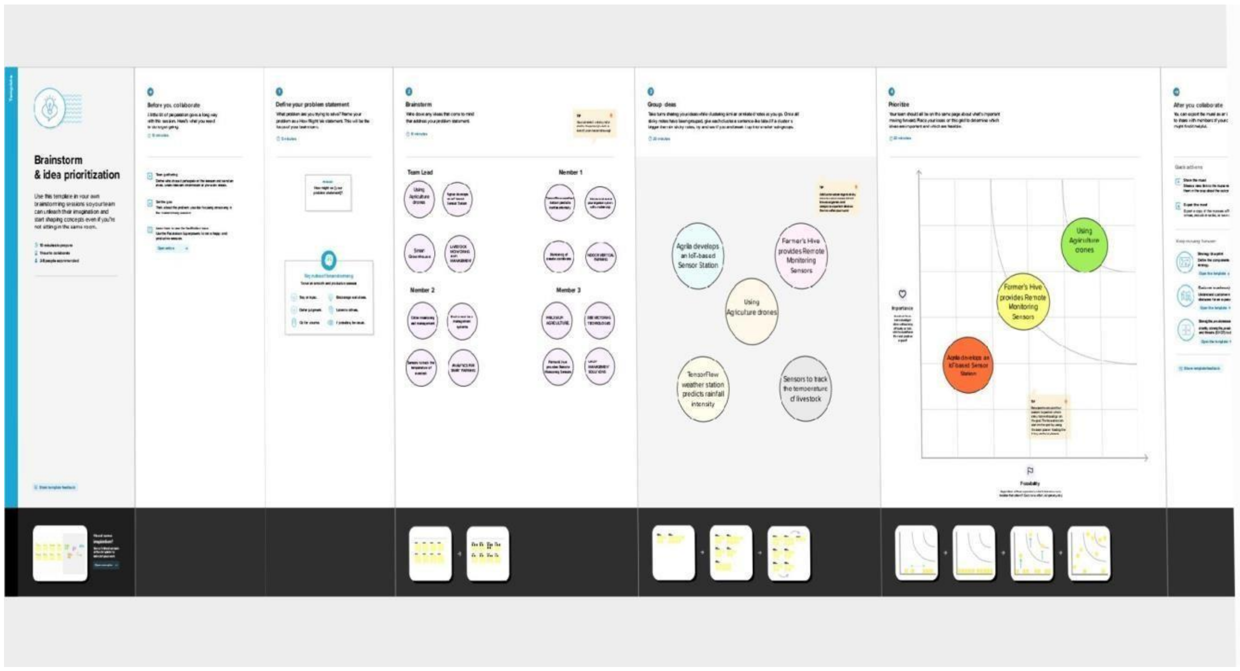
miro

3. Ideation & Proposed Solution

3.1 Prepare Empathy Map

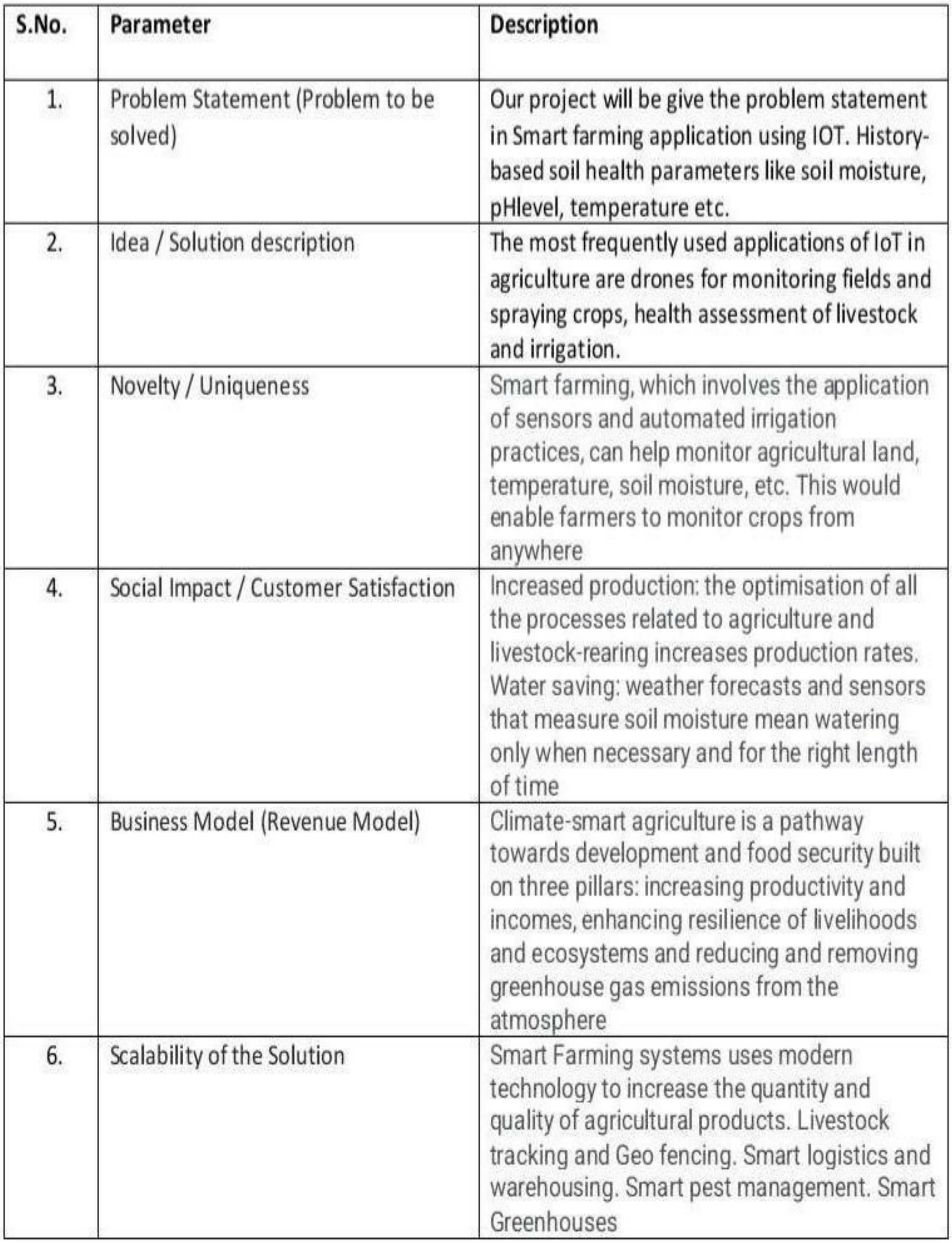


3.2 Ideation



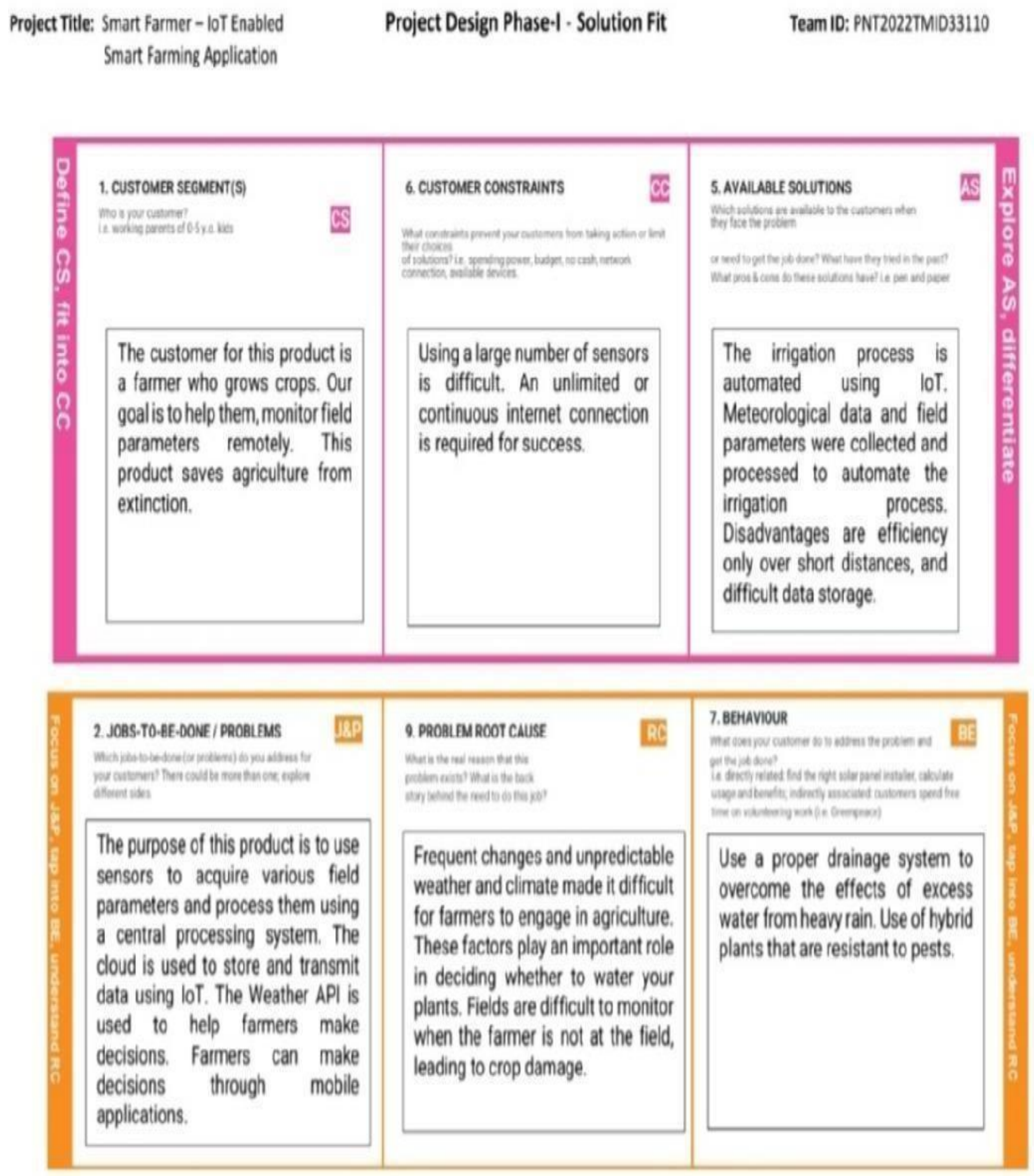
Brainstorm & idea prioritization
Agrila develops an IoT-based Sensor Station
Farmer's Hive provides Remote Monitoring Sensors
Using Agriculture drones
TensorFlow weather station predicts rainfall intensity
Sensors to track the temperature of livestock

3.3 Proposed Solution



S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Our project will be give the problem statement in Smart farming application using IOT. History-based soil health parameters like soil moisture, pHlevel, temperature etc.
2.	Idea / Solution description	The most frequently used applications of IoT in agriculture are drones for monitoring fields and spraying crops, health assessment of livestock and irrigation.
3.	Novelty / Uniqueness	Smart farming, which involves the application of sensors and automated irrigation practices, can help monitor agricultural land, temperature, soil moisture, etc. This would enable farmers to monitor crops from anywhere
4.	Social Impact / Customer Satisfaction	Increased production: the optimisation of all the processes related to agriculture and livestock-rearing increases production rates. Water saving: weather forecasts and sensors that measure soil moisture mean watering only when necessary and for the right length of time
5.	Business Model (Revenue Model)	Climate-smart agriculture is a pathway towards development and food security built on three pillars: increasing productivity and incomes, enhancing resilience of livelihoods and ecosystems and reducing and removing greenhouse gas emissions from the atmosphere
6.	Scalability of the Solution	Smart Farming systems uses modern technology to increase the quantity and quality of agricultural products. Livestock tracking and Geo fencing. Smart logistics and warehousing. Smart pest management. Smart Greenhouses

3.4 Proposed Solution Fit



Project Title: Smart Farmer – IoT Enabled Smart Farming Application

Project Design Phase-I - Solution Fit

Team ID: PNT2022TMID33110

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

The customer for this product is a farmer who grows crops. Our goal is to help them, monitor field parameters remotely. This product saves agriculture from extinction.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices
of solutions? i.e. spending power, budget, no cash, network connection, available devices.

Using a large number of sensors is difficult. An unlimited or continuous internet connection is required for success.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem

or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper

The irrigation process is automated using IoT. Meteorological data and field parameters were collected and processed to automate the irrigation process. Disadvantages are efficiency only over short distances, and difficult data storage.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides

The purpose of this product is to use sensors to acquire various field parameters and process them using a central processing system. The cloud is used to store and transmit data using IoT. The Weather API is used to help farmers make decisions. Farmers can make decisions through mobile applications.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists? What is the back story behind the need to do this job?

Frequent changes and unpredictable weather and climate made it difficult for farmers to engage in agriculture. These factors play an important role in deciding whether to water your plants. Fields are difficult to monitor when the farmer is not at the field, leading to crop damage.

7. BEHAVIOUR BE

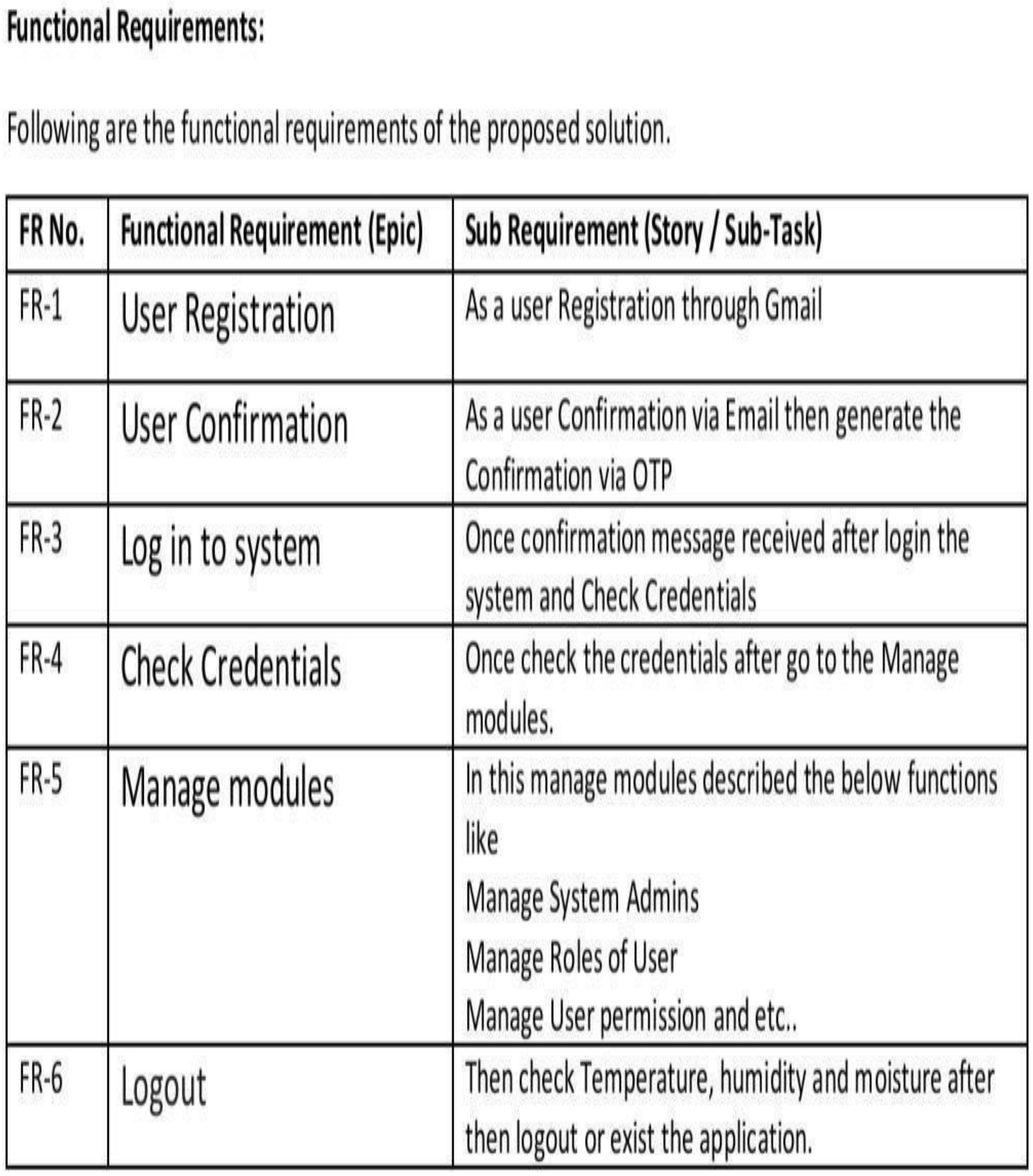
What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

Use a proper drainage system to overcome the effects of excess water from heavy rain. Use of hybrid plants that are resistant to pests.

Focus on J&P, tap into BE, understand RC

4.Requirement Analysis

4.1 Functional Requirement

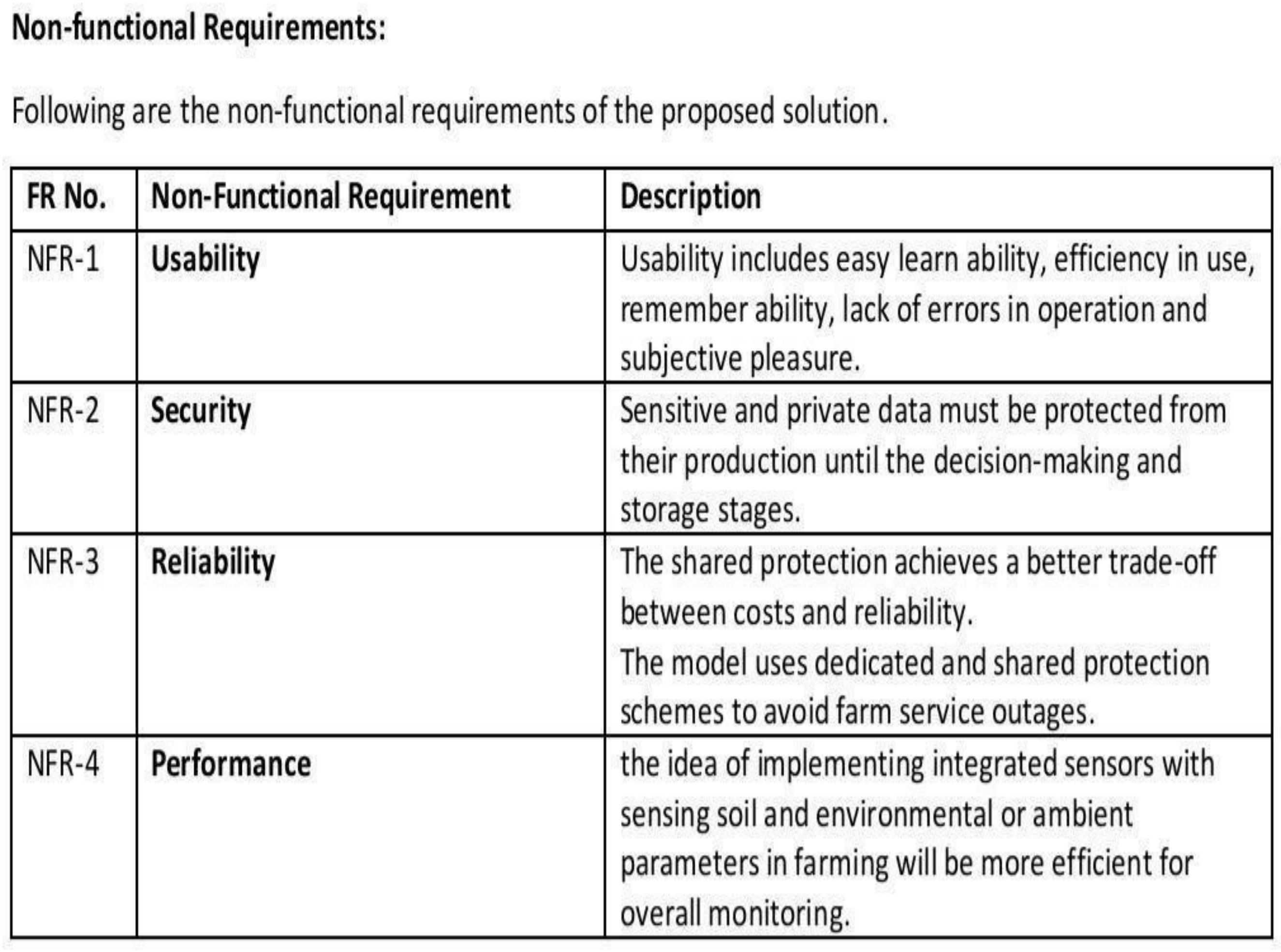


Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	As a user Registration through Gmail
FR-2	User Confirmation	As a user Confirmation via Email then generate the Confirmation via OTP
FR-3	Log in to system	Once confirmation message received after login the system and Check Credentials
FR-4	Check Credentials	Once check the credentials after go to the Manage modules.
FR-5	Manage modules	In this manage modules described the below functions like Manage System Admins Manage Roles of User Manage User permission and etc..
FR-6	Logout	Then check Temperature, humidity and moisture after then logout or exist the application.

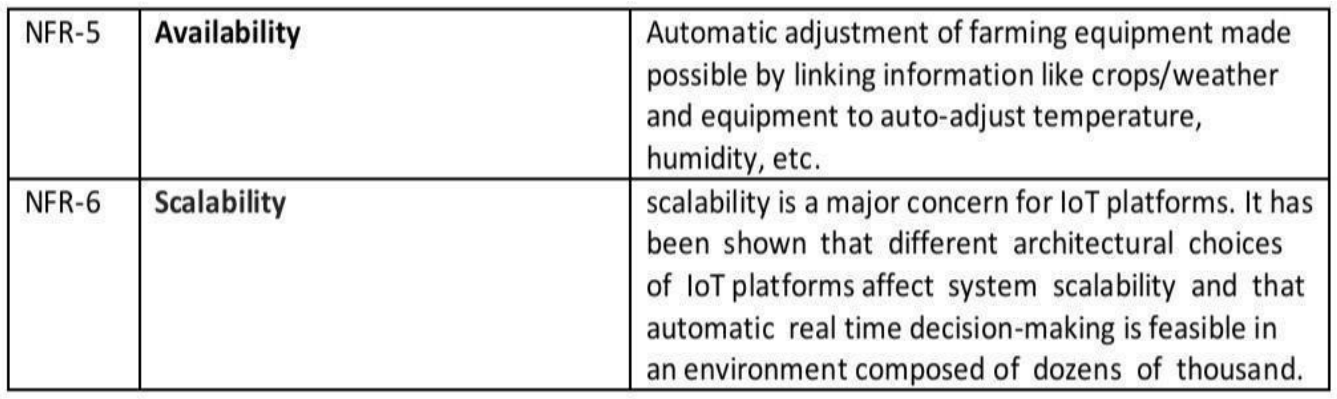
4.2 Non- Functional Requirements



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 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 or ambient parameters in farming will be more efficient for overall monitoring.
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 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.

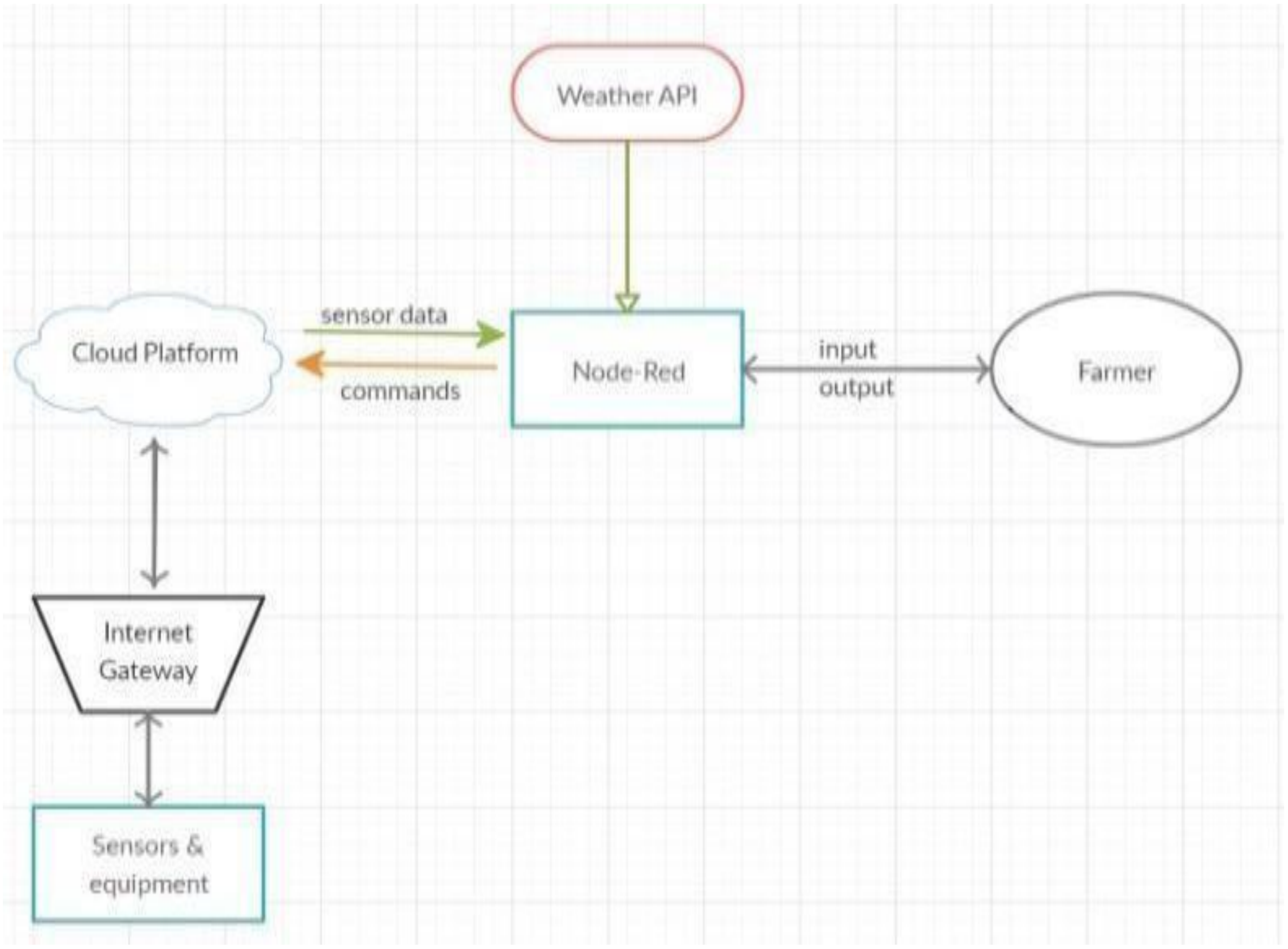


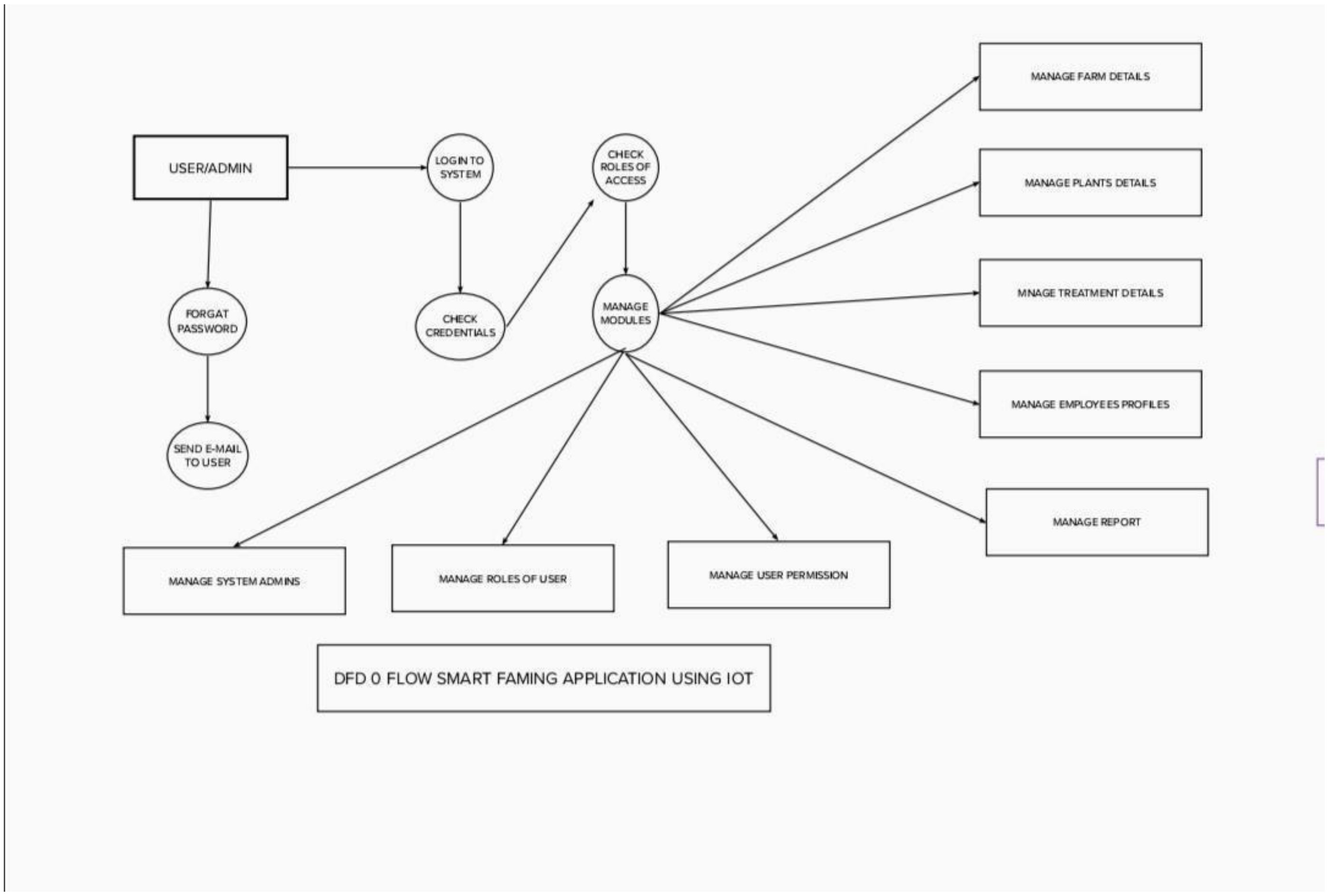
5.Project Design

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.

5.2Solution & 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.

Example - Solution Architecture Diagram:

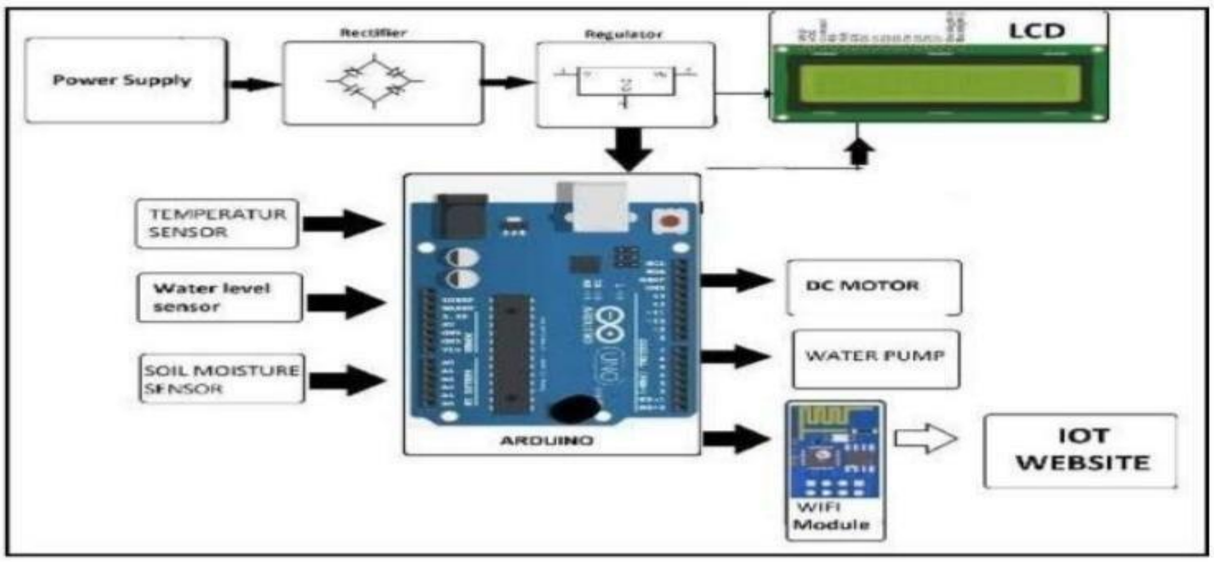


Figure 1: IOT ENABLED SMART FARMING APPLICATION

- 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.

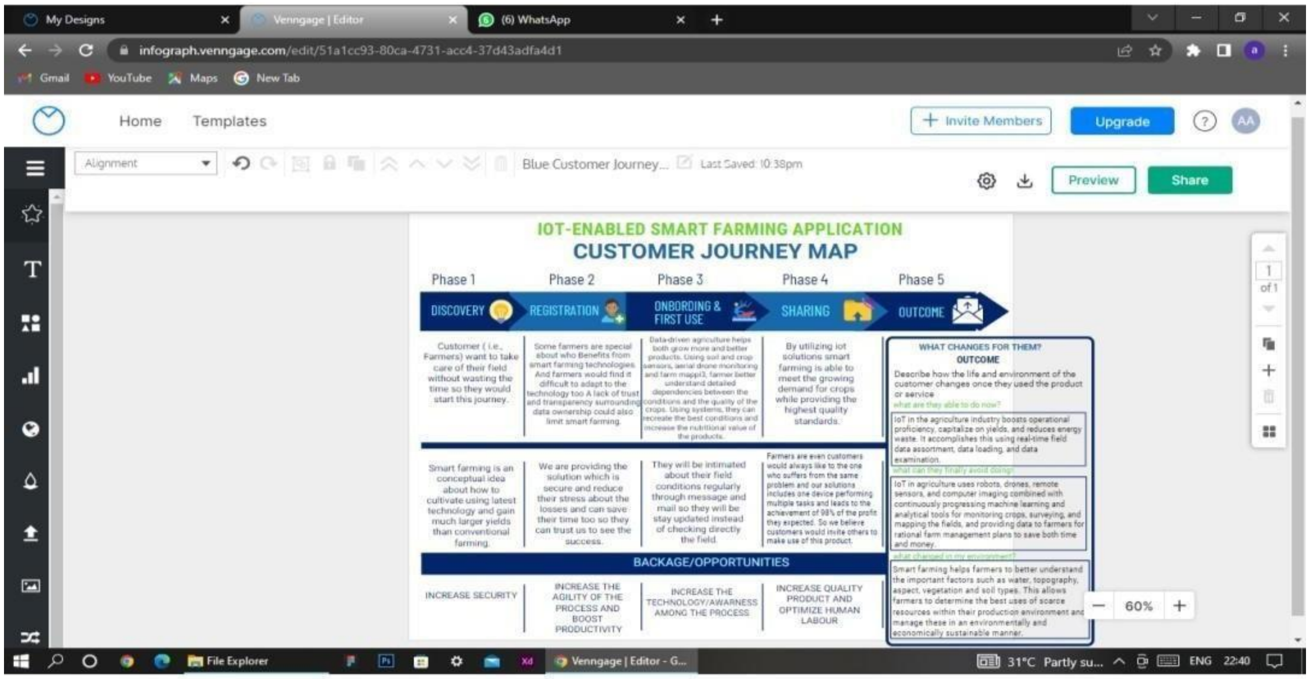
Table-1 : Components & Technologies:

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.

Table-2: Application Characteristics:

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

5.3User Stories



IOT-ENABLED SMART FARMING APPLICATION

CUSTOMER JOURNEY MAP

Phase 1	Phase 2	Phase 3	Phase 4	Phase 5
DISCOVERY	REGISTRATION	ONBORDING & FIRST USE	SHARING	OUTCOME
Customer (i.e., Farmers) want to take care of their field without wasting the time so they would start this journey.	Some farmers are special about who Benefits from smart farming technologies. And farmers would find it difficult to adapt to the technology too A lack of trust and transparency surrounding data ownership could also limit smart farming.	Data-driven agriculture helps both grow more and better products. Using soil and crop sensors, aerial drone monitoring and farm mappi3, farmer better understand detailed dependencies between the conditions and the quality of the crops. Using systems, they can recreate the best conditions and increase the nutritional value of the products.	By utilizing iot solutions smart farming is able to meet the growing demand for crops while providing the highest quality standards.	
Smart farming is an conceptual idea about how to cultivate using latest technology and gain much larger yields than conventional farming.	We are providing the solution which is secure and reduce their stress about the losses and can save their time too so they can trust us to see the success.	They will be intimated about their field conditions regularly through message and mail so they will be stay updated instead of checking directly the field.	Farmers are even customers would always like to the one who suffers from the same problem and our solutions includes one device performing multiple tasks and leads to the achievement of 98% of the profit they expected. So we believe customers would invite others to make use of this product.	

BACKAGE/OPPORTUNITIES

INCREASE SECURITY	INCREASE THE AGILITY OF THE PROCESS AND BOOST PRODUCTIVITY	INCREASE THE TECHNOLOGY/AWARENESS AMONG THE PROCESS	INCREASE QUALITY PRODUCT AND OPTIMIZE HUMAN LABOUR

WHAT CHANGES FOR THEM?

OUTCOME

Describe how the life and environment of the customer changes once they used the product or service

what are they able to do now?

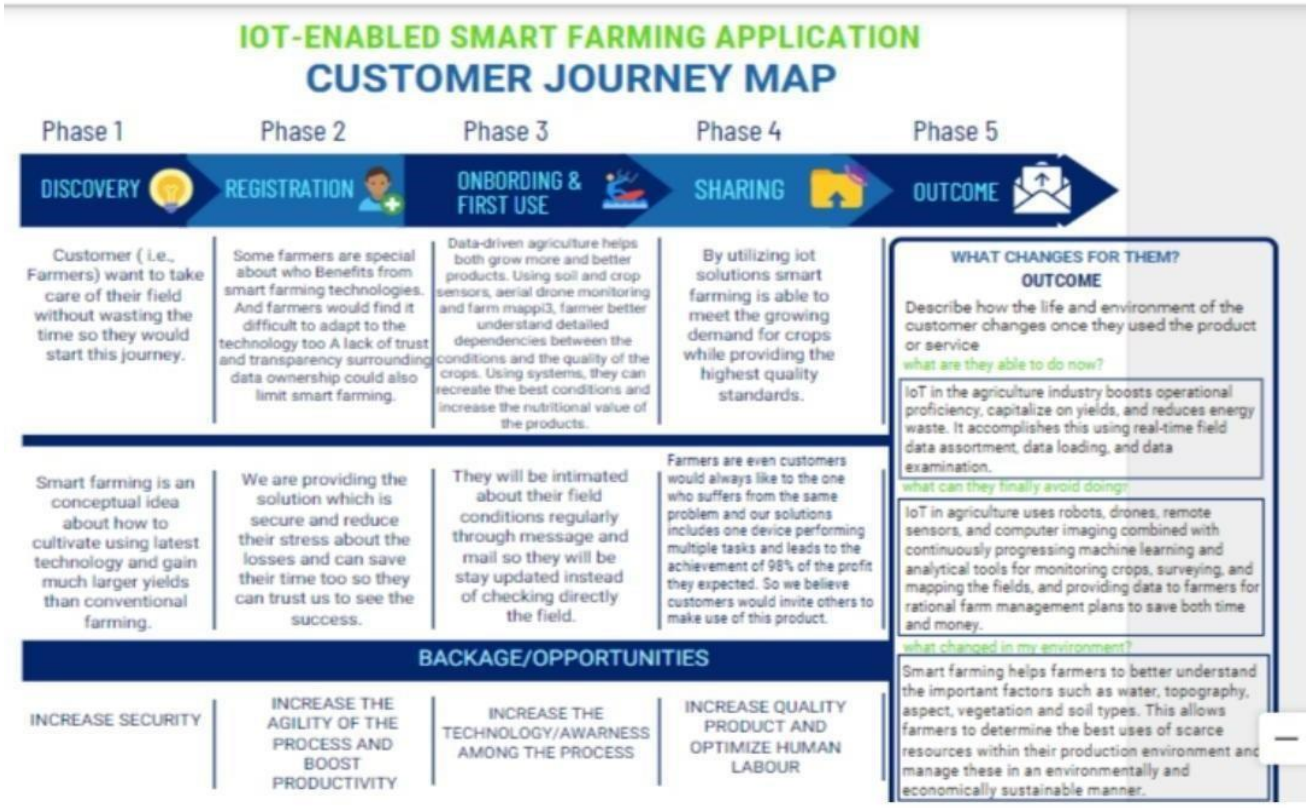
IoT in the agriculture industry boosts operational proficiency, capitalize on yields, and reduces energy waste. It accomplishes this using real-time field data assortment, data loading, and data examination.

what can they finally avoid doing?

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.

what changed in my environment?

Smart farming helps farmers to better understand the important factors such as water, topography, aspect, vegetation and soil types. This allows farmers to determine the best uses of scarce resources within their production environment and manage these in an environmentally and economically sustainable manner.



IOT-ENABLED SMART FARMING APPLICATION

CUSTOMER JOURNEY MAP

Phase 1 DISCOVERY; Phase 2 REGISTRATION; Phase 3 ONBORDING & FIRST USE; Phase 4 SHARING; Phase 5 OUTCOME

Customer (i.e., Farmers) want to take care of their field without wasting the time so they would start this journey.

Some farmers are special about who Benefits from smart farming technologies. And farmers would find it difficult to adapt to the technology too A lack of trust and transparency surrounding data ownership could also limit smart farming.

Data-driven agriculture helps both grow more and better products. Using soil and crop sensors, aerial drone monitoring and farm mappi3, farmer better understand detailed dependencies between the conditions and the quality of the crops. Using systems, they can recreate the best conditions and increase the nutritional value of the products.

By utilizing iot solutions smart farming is able to meet the growing demand for crops while providing the highest quality standards.

Smart farming is an conceptual idea about how to cultivate using latest technology and gain much larger yields than conventional farming.

We are providing the solution which is secure and reduce their stress about the losses and can save their time too so they can trust us to see the success.

They will be intimated about their field conditions regularly through message and mail so they will be stay updated instead of checking directly the field.

Farmers are even customers would always like to the one who suffers from the same problem and our solutions includes one device performing multiple tasks and leads to the achievement of 98% of the profit they expected. So we believe customers would invite others to make use of this product.

BACKAGE/OPPORTUNITIES

INCREASE SECURITY; INCREASE THE AGILITY OF THE PROCESS AND BOOST PRODUCTIVITY; INCREASE THE TECHNOLOGY/AWARENESS AMONG THE PROCESS; INCREASE QUALITY PRODUCT AND OPTIMIZE HUMAN LABOUR

WHAT CHANGES FOR THEM? OUTCOME

Describe how the life and environment of the customer changes once they used the product or service

what are they able to do now?

IoT in the agriculture industry boosts operational proficiency, capitalize on yields, and reduces energy waste. It accomplishes this using real-time field data assortment, data loading, and data examination.

what can they finally avoid doing?

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.

what changed in my environment?

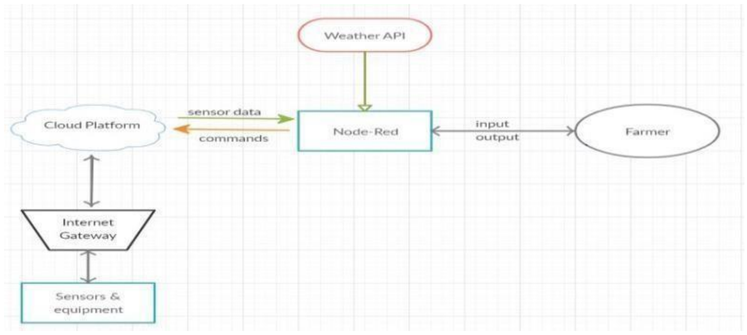
Smart farming helps farmers to better understand the important factors such as water, topography, aspect, vegetation and soil types. This allows farmers to determine the best uses of scarce resources within their production environment and manage these in an environmentally and economically sustainable manner.

6. Project planning & Scheduling

6.1 Sprint Delivery planning & Estimation

SPRINT DELIVERY OVERVIEW :

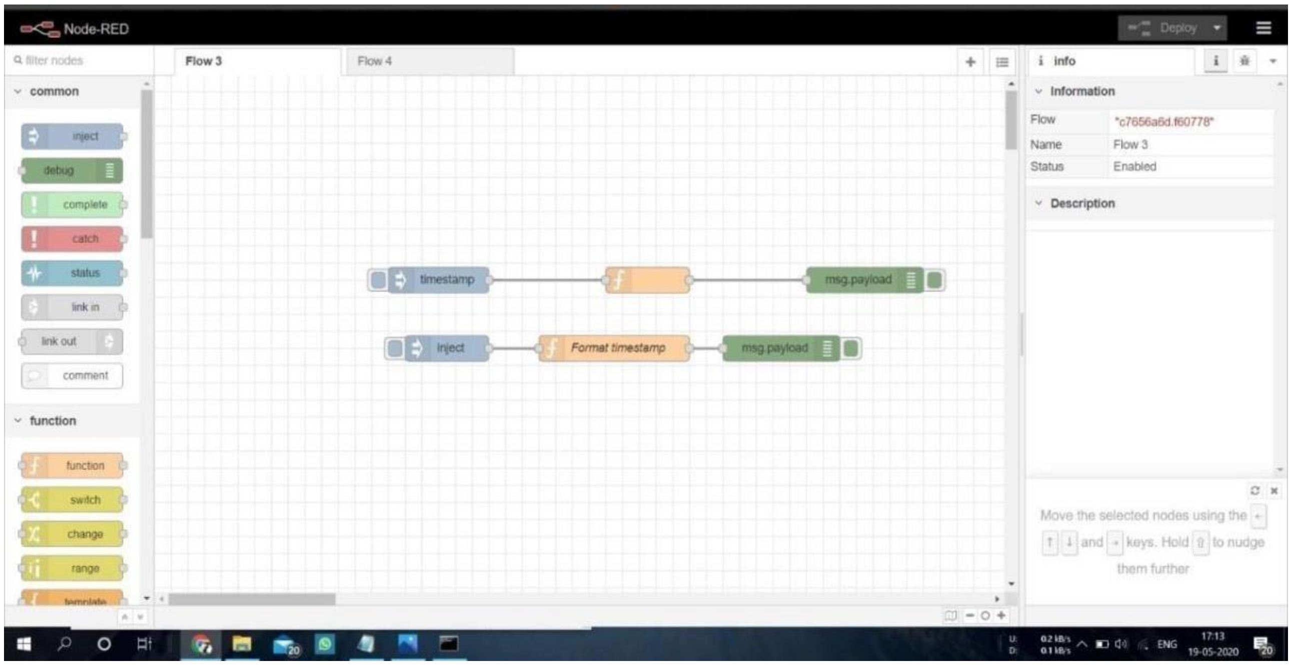
In order to implement the solution , the following approach as shown in the block diagram is used



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

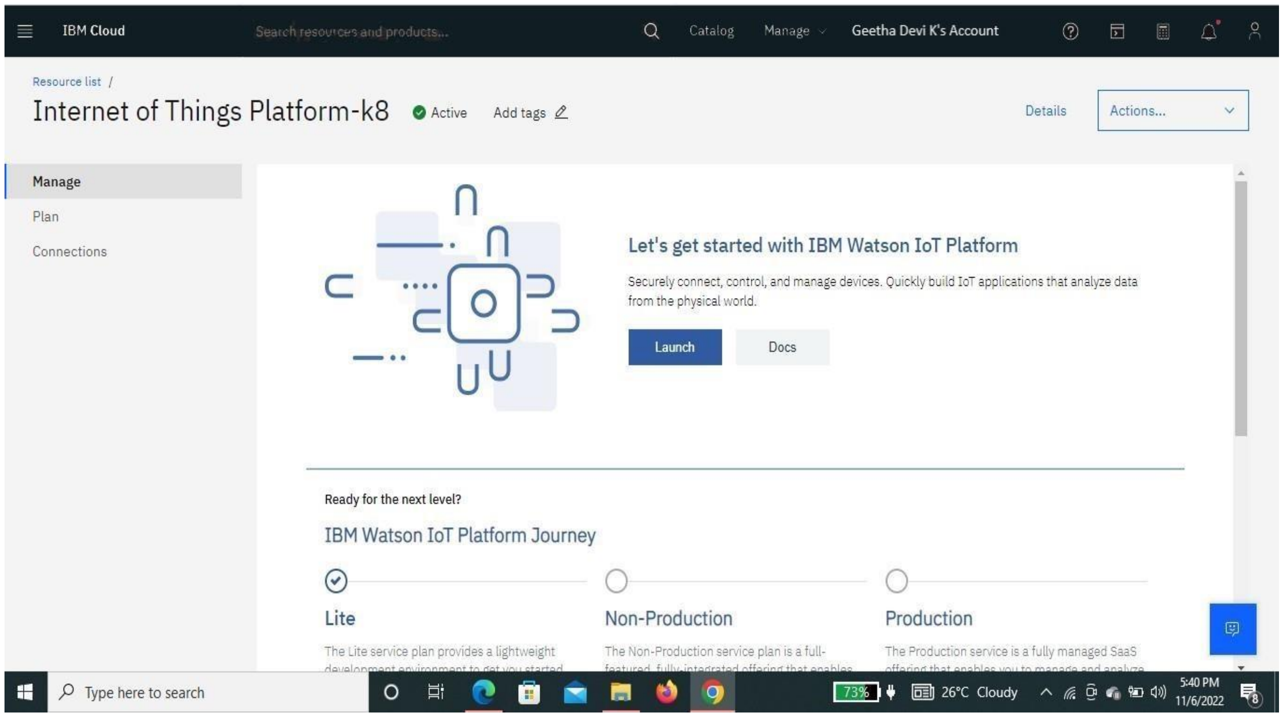
Installation of IBM IoT and Dashboard nodes for Node-Red

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.



Let's get started with IBM Watson IoT Platform

Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.

The Lite service plan provides a lightweight development environment to get you started

The Non-Production service plan is a full-featured, fully integrated offering that enables

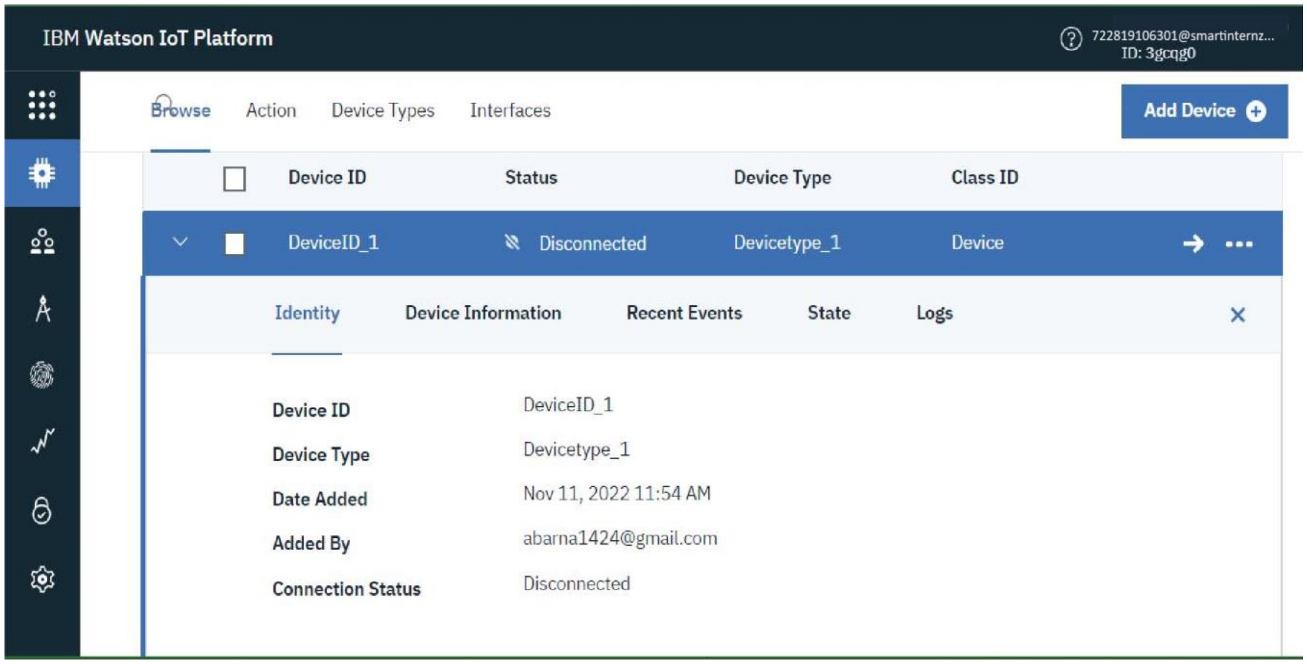
The Production service is a fully managed SaaS offering that enables you to manage and analyze

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

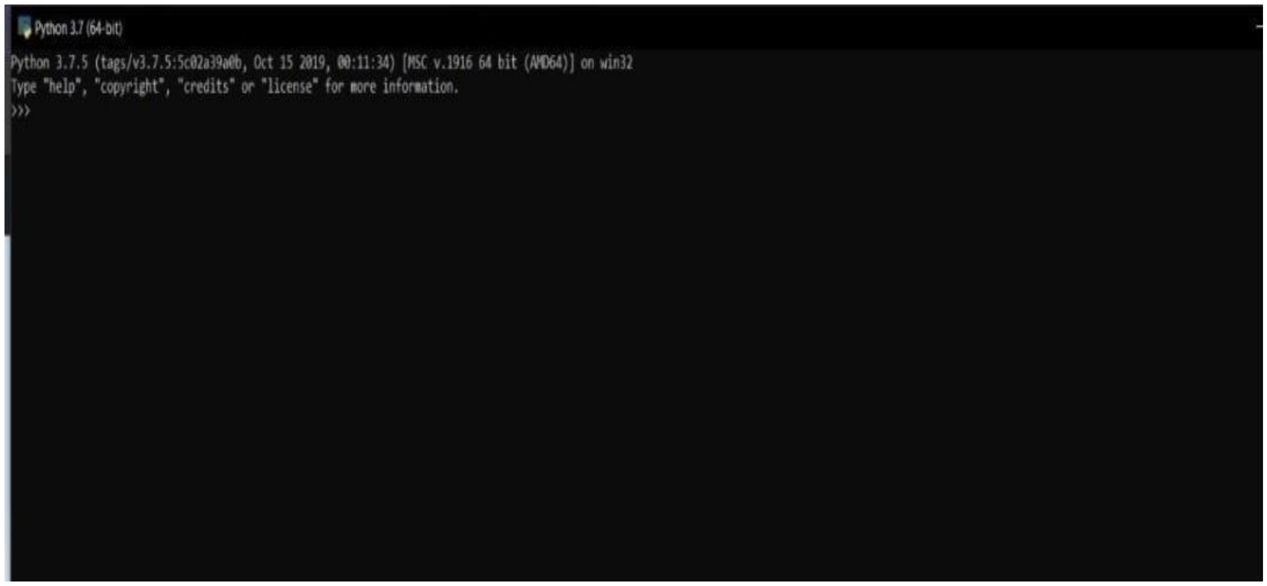


Device ID	Status	Device Type	Class ID
DeviceID_1	Disconnected	Devicetype_1	Device

Device ID	DeviceID_1
Device Type	Devicetype_1
Date Added	Nov 11, 2022 11:54 AM
Added By	abarna1424@gmail.com
Connection Status	Disconnected

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

1.3 IoT Simulator



```
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

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 OpenWeather API

OpenWeatherMap is an online service that provides weather data. It provides current weather data, forecasts and historical data to more than 2 million customerWebsite link: https://openweathermap.org/guide

Steps to configure:

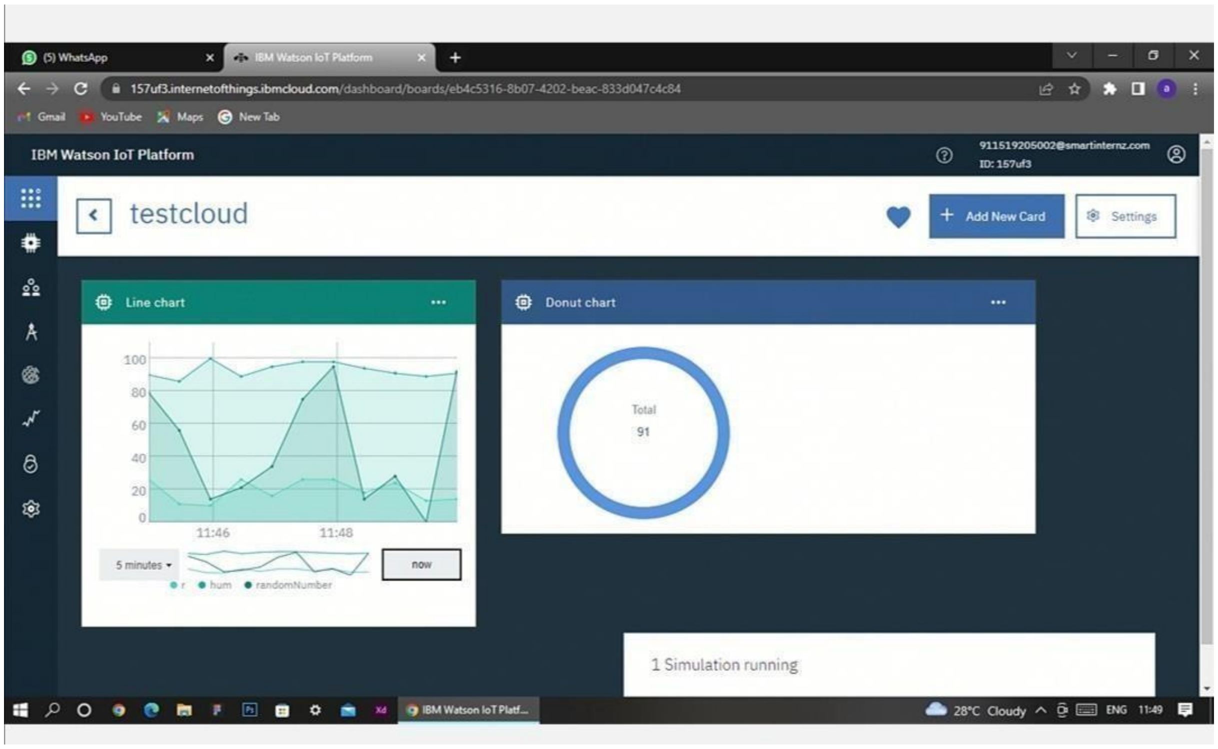
o Create account in OpenWeather o Find the name of your city by searching o Create API key to your account o Replace “city name” and “your api key” with your city and API key in below red text

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
- Click on connect
- My credentials given to simulator are:

    Org ID: 3gcqg0

    DevType:Devicetype_1

    Device ID : DeviceID_1

    Device Token : 12345678

  You will receive the simulator data in cloud
- you can see the received data in Recent Events under your device
- Data received in this format(json)

- { o "d": {

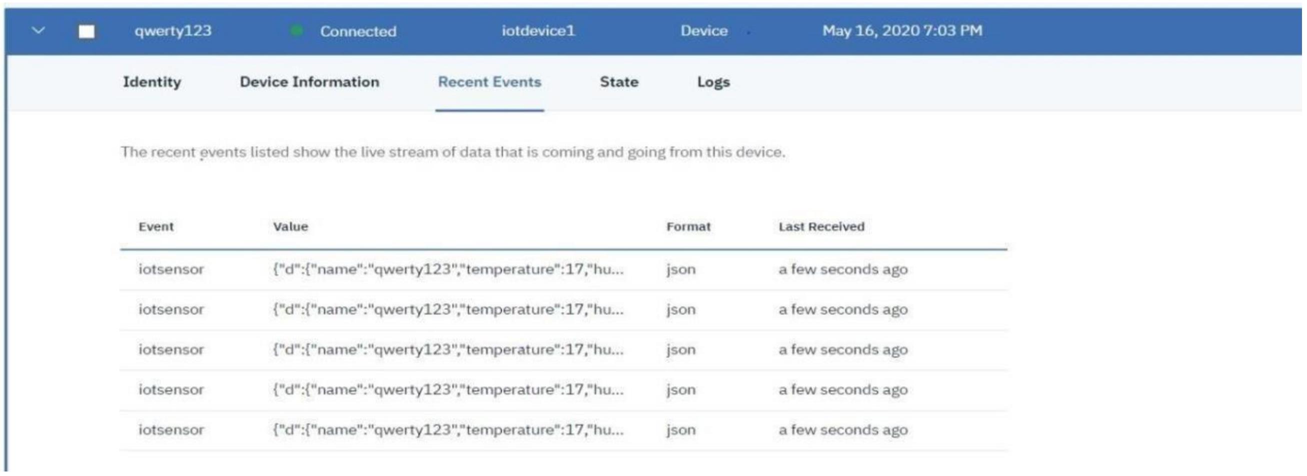
  "name": "qwerty123",

  "temperature": 17,

  "humidity": 76,

  "objectTemp": 25 o } }

- You can see the received data in graphs by creating cards in Boards tab.

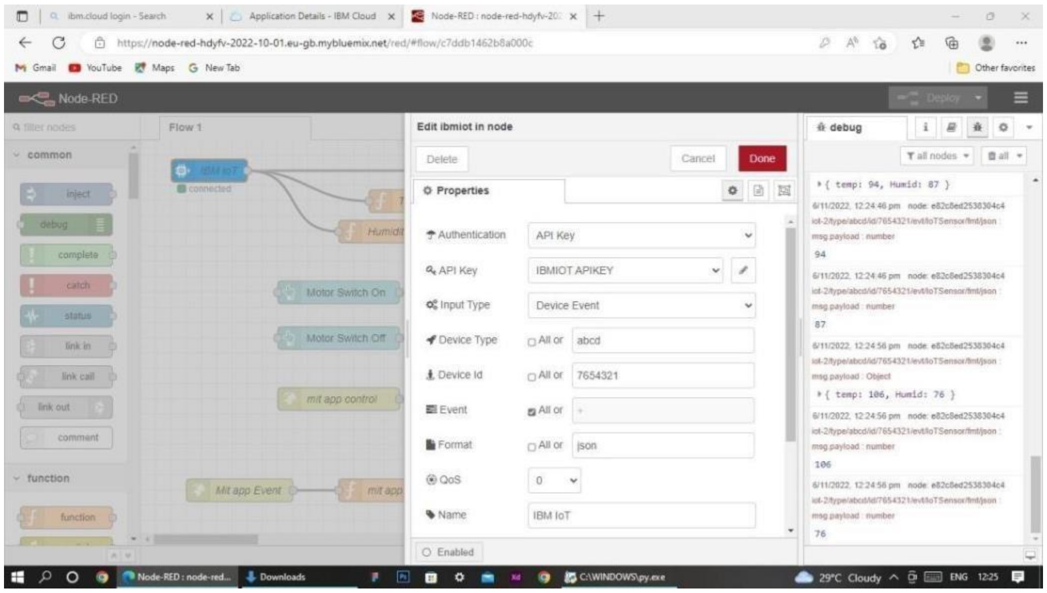


The recent events listed show the live stream of data that is coming and going from this device.

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



Edit ibmiot in node

Authentication	API Key
API Key	IBMIOT APIKEY
Input Type	Device Event
Device Type	☐ All or abcd
Device Id	☐ All or 7654321
Event	☑ All or +
Format	☐ All or json
QoS	0
Name	IBM IoT

```
▸{ temp: 94, Humid: 87 }
6/11/2022, 12:24:46 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
94
6/11/2022, 12:24:46 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
87
6/11/2022, 12:24:56 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 106, Humid: 76 }
6/11/2022, 12:24:56 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
106
6/11/2022, 12:24:56 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
76
```

Once it is connected Node-Red receives data from the device

Display the data using debug node for verification

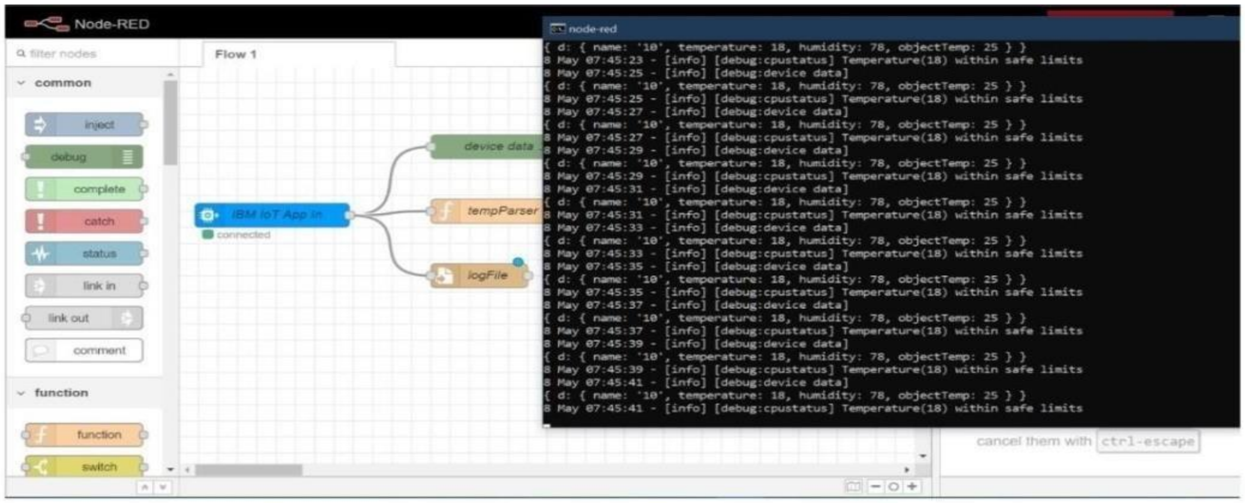
Connect function node and write the Java script code to get each reading separately.

The Java script code for the function node is:

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

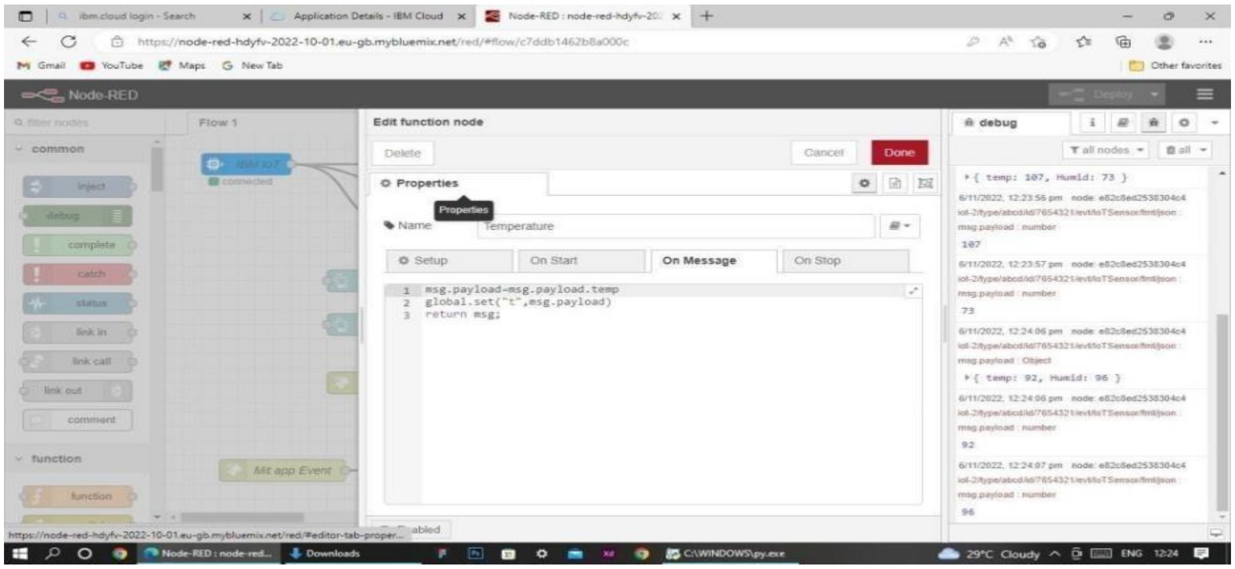
Finally connect Gauge nodes from dashboard to see the data in UI

Nodes connected in following manner to get each reading separately. This is the Java script code I have written for the function node to get Temperature separately.



```
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:23 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:25 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:25 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:27 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:27 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:29 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:29 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:31 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:31 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:33 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:33 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:35 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:35 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:37 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:37 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:39 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:39 - [info] [debug:cpustatus] Temperature(18) within safe limits
8 May 07:45:41 - [info] [debug:device data]
{ d: { name: '10', temperature: 18, humidity: 78, objectTemp: 25 } }
8 May 07:45:41 - [info] [debug:cpustatus] Temperature(18) within safe limits
```

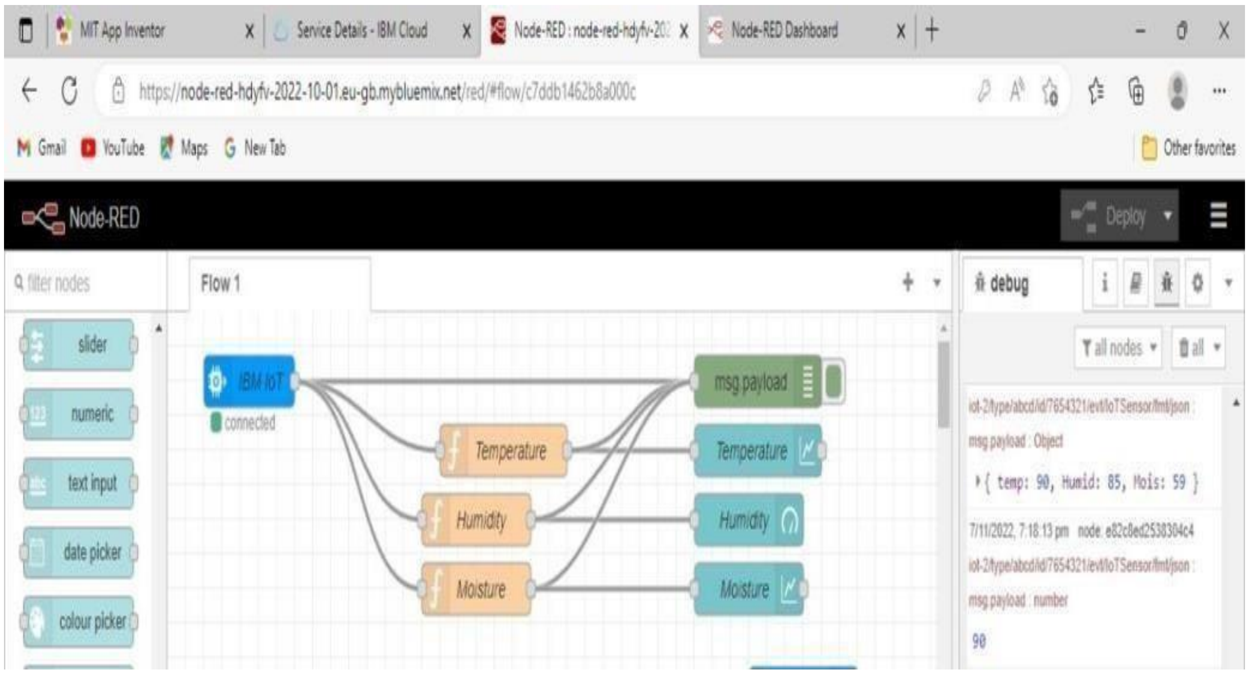
Data received from the cloud in Node-RED console



Name: Temperature

```
msg.payload-msg.payload.temp
global.set("t",msg.payload)
return msg;
```

```
▸{ temp: 107, Humid: 73 }
6/11/2022, 12:23:56 pm node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json : msg.payload : number
107
6/11/2022, 12:23:57 pm node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json : msg.payload : number
73
6/11/2022, 12:24:06 pm node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json : msg.payload : Object
▸{ temp: 92, Humid: 96 }
6/11/2022, 12:24:06 pm node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json : msg.payload : number
92
6/11/2022, 12:24:07 pm node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json : msg.payload : number
96
```



```
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json : msg.payload : Object
▸{ temp: 90, Humid: 85, Mois: 59 }
7/11/2022, 7:18:13 pm node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json : msg.payload : number
90
```

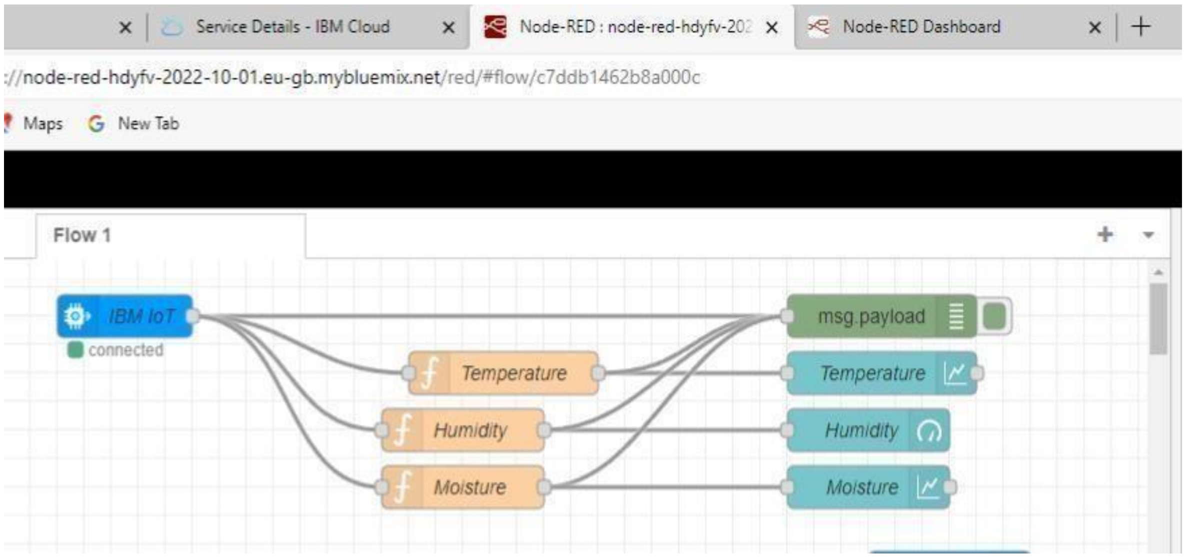
2.3 Configuration of Node-Red to collect data from OpenWeather

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 4.4The 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 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":1589 93
3553,"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)};

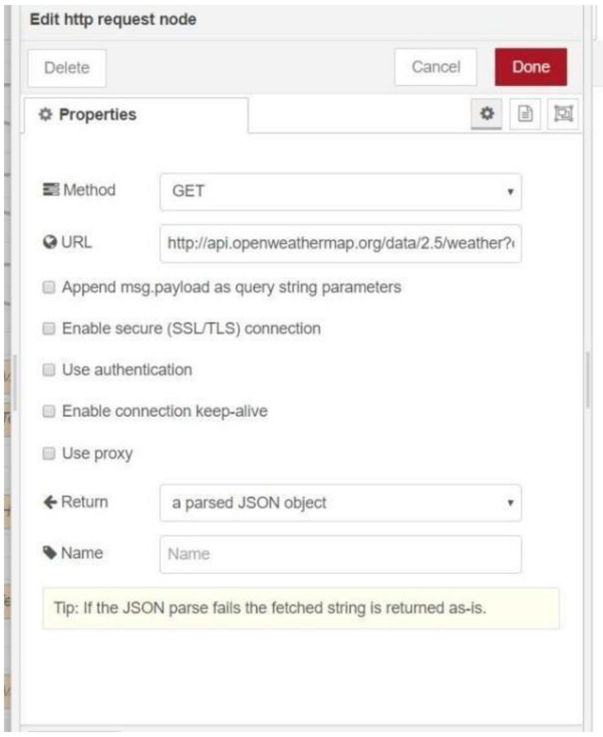
In the above Java script code we take temperature parameter into a new variable and convert it from kelvin to Celsius.



Service Details - IBM Cloud
Node-RED : node-red-hdyfv-202
Node-RED Dashboard
://node-red-hdyfv-2022-10-01.eu-gb.mybluemix.net/red/#flow/c7ddb1462b8a000c
Maps
New Tab
Flow 1
IBM IoT
connected
Temperature
Humidity
Moisture
msg.payload
Temperature
Humidity
Moisture

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.


Edit http request node

Delete Cancel Done

Properties

Method: GET

URL: http://api.openweathermap.org/data/2.5/weather?

☐ Append msg.payload as query string parameters

☐ Enable secure (SSL/TLS) connection

☐ Use authentication

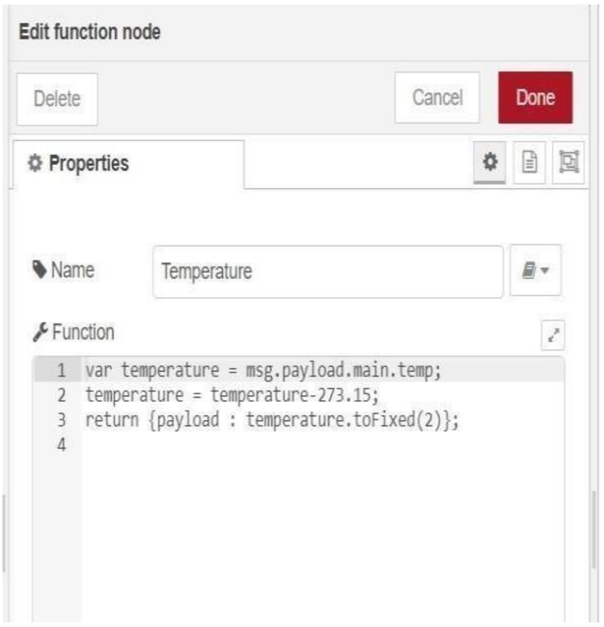
☐ Enable connection keep-alive

☐ Use proxy

Return: a parsed JSON object

Name: Name

Tip: If the JSON parse fails the fetched string is returned as-is.


Edit function node

Delete Cancel Done

Properties

Name: Temperature

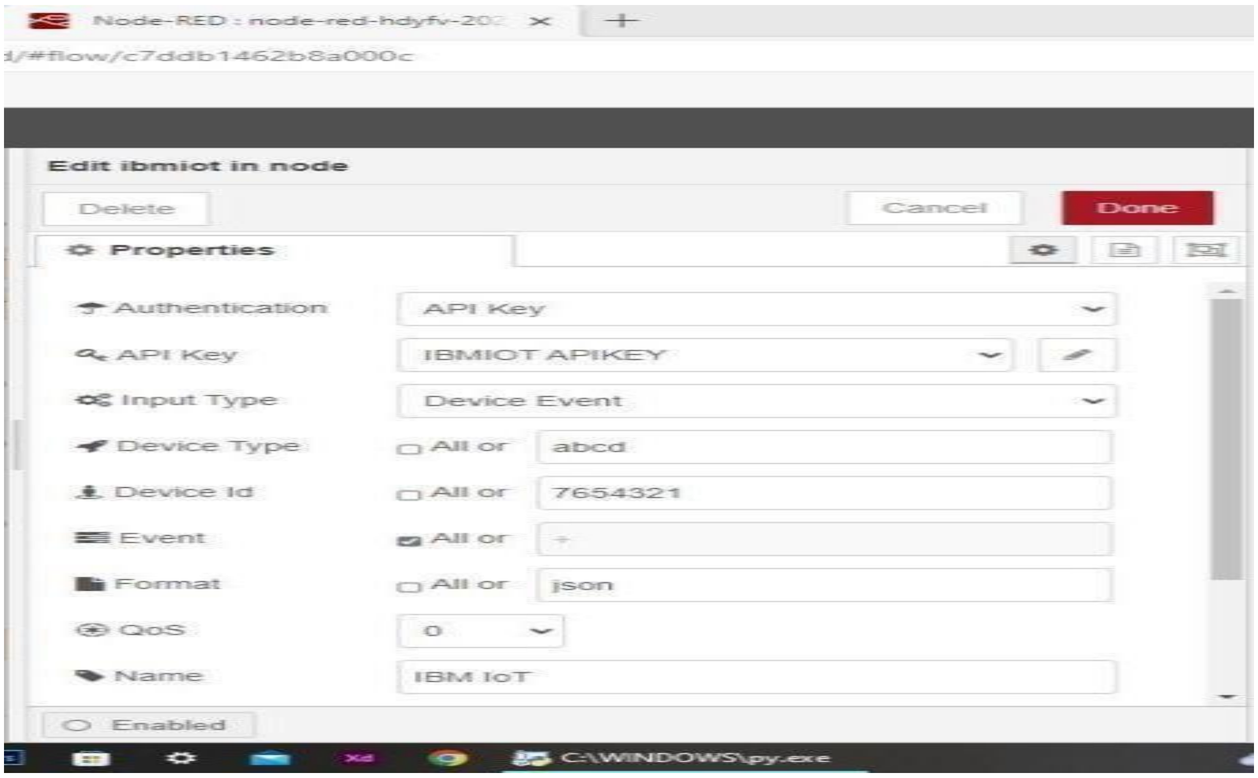
Function

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

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

IBM iot 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.


Node-RED : node-red-hdyfv-20

d/#flow/c7ddb1462b8a000c

Edit ibmiot in node

Delete Cancel Done

Properties

Authentication: API Key

API Key: IBMIOT APIKEY

Input Type: Device Event

Device Type: ☐ All or abcd

Device Id: ☐ All or 7654321

Event: ☑ All or +

Format: ☐ All or json

QoS: 0

Name: IBM IoT

Enabled

C:\WINDOWS\py.exe

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.

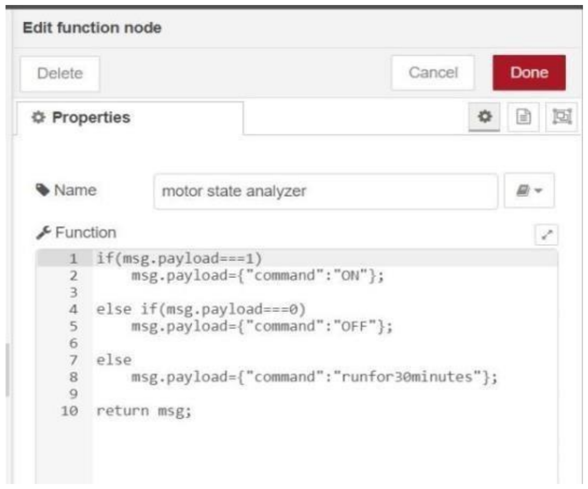
The Java script code for the analyser is:

```
 if(msg.payload===1)
    msg.payload={"command":"O
    N"}; else if(msg.payload===0)
    msg.payload={"command":"O
    FF"}; else
    msg.payload={"command":"ru
    nfor30minutes"};
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;
```



Name: motor state analyzer

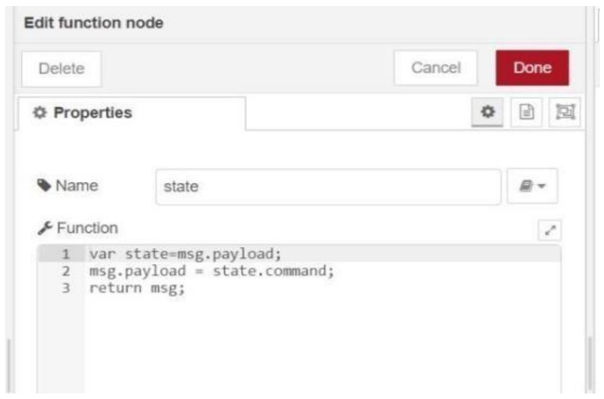
Function

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

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

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

return msg;
```



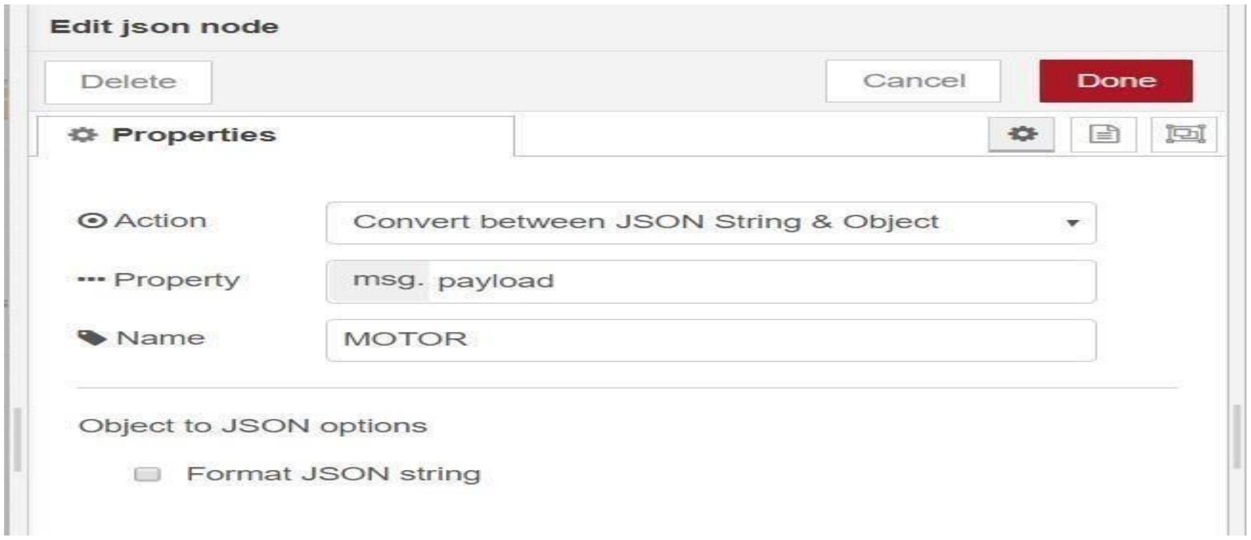
Name: state

Function

```
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 IoTOut.


Edit json node

Delete | Cancel | Done

Properties

Action: Convert between JSON String & Object

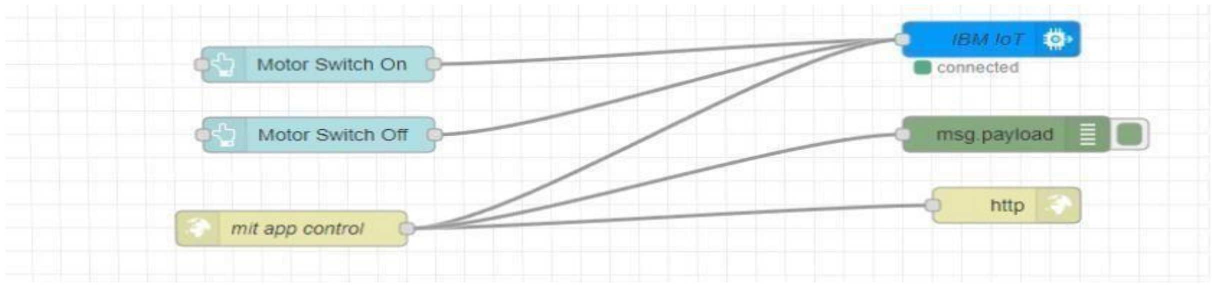
Property: msg. payload

Name: MOTOR

Object to JSON options

Format JSON string

Edit JSON node needs to be configured like this



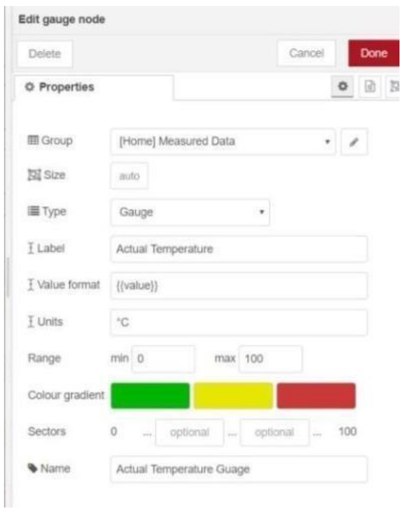
This is the program flow for sending commands to IBM cloud.

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.

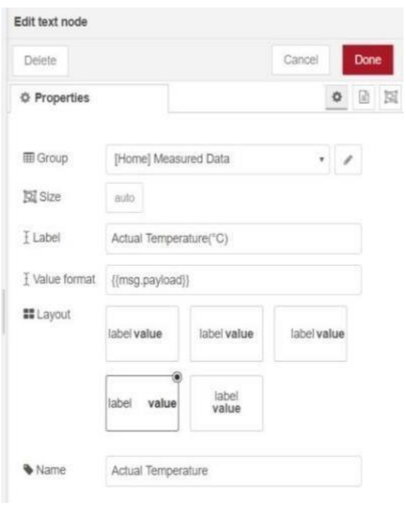
Below images are the Gauge, text and button node configurations


Edit gauge node

Delete | Cancel | Done

Properties

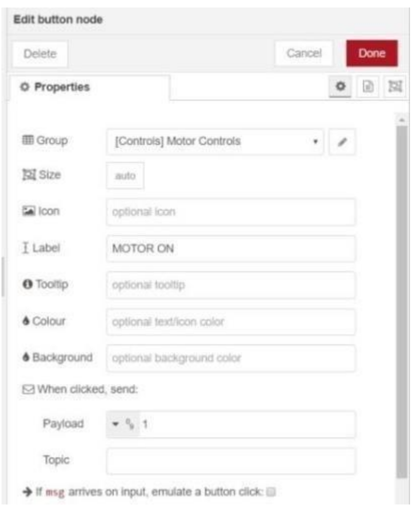
Group: [Home] Measured Data
Size: auto
Type: Gauge
Label: Actual Temperature
Value format: {{value}}
Units: °C
Range: min 0 max 100
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 100
Name: Actual Temperature Guage


Edit text node

Delete | Cancel | Done

Properties

Group: [Home] Measured Data
Size: auto
Label: Actual Temperature(°C)
Value format: {{msg.payload}}
Layout: label value | label value | label value | label value | label value
Name: Actual Temperature


Edit button node

Delete | Cancel | Done

Properties

Group: [Controls] Motor Controls
Size: auto
Icon: optional icon
Label: MOTOR ON
Tooltip: optional tooltip
Colour: optional text/icon color
Background: optional background color
When clicked, send:
Payload: 1
Topic:
If msg arrives on input, emulate a button click: ☐

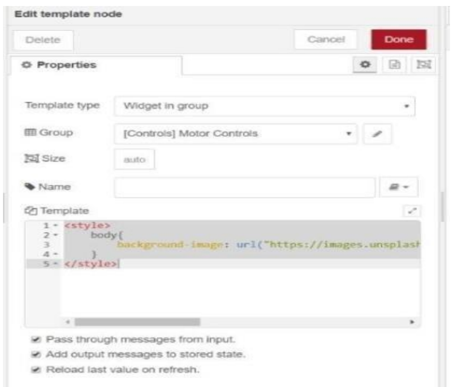
Adding Background image to the UI

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");
          }
</style>
```

We add URL of image that need to be displayed

Configuration is shown in the beside image


Edit template node

Delete | Cancel | Done

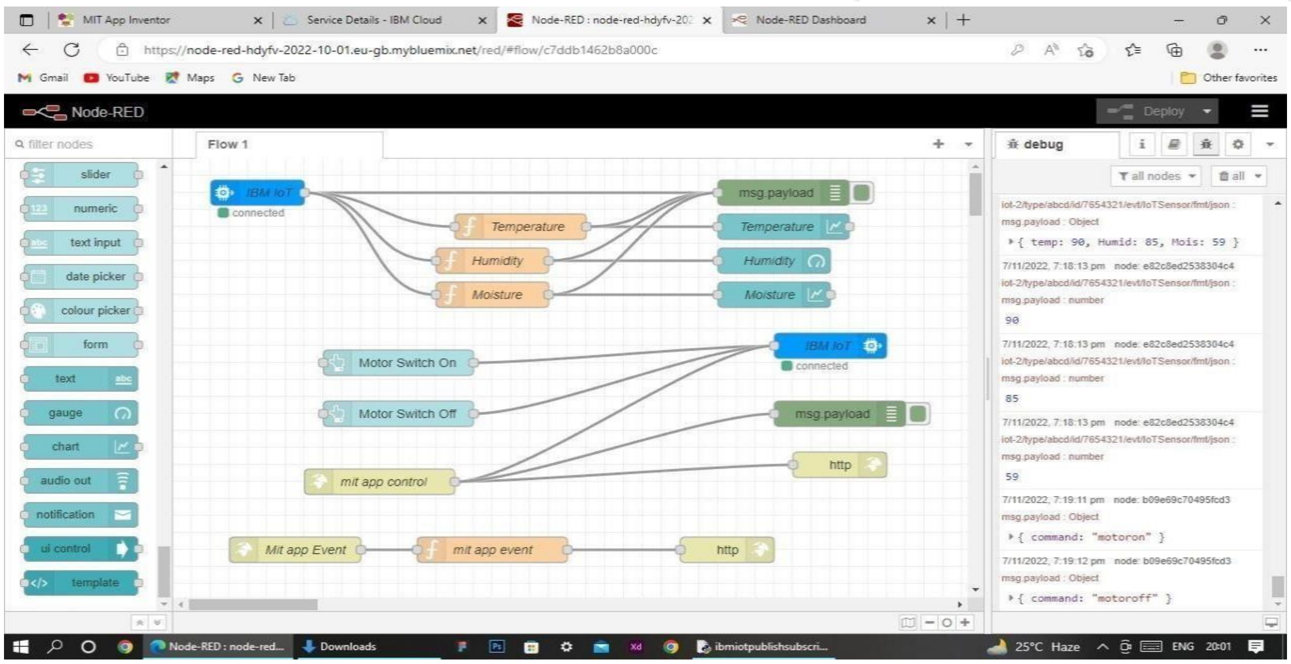
Properties

Template type: Widget in group
Group: [Controls] Motor Controls
Size: auto
Name:
Template

```
<style>
    body{
        background-image: url("https://images.unsplash
    }
</style>
```

☑ Pass through messages from input.
☑ Add output messages to stored state.
☑ Reload last value on refresh.

Complete Program Flow

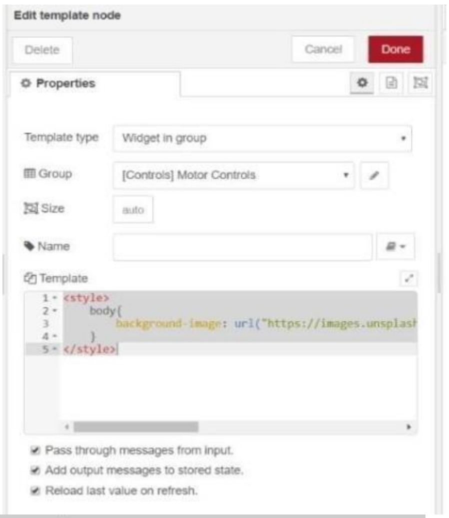


```
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 90, Humid: 85, Mois: 59 }
7/11/2022, 7:18:13 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
90
7/11/2022, 7:18:13 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
85
7/11/2022, 7:18:13 pm  node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
59
7/11/2022, 7:19:11 pm  node: b09e69c70495fcd3
msg.payload : Object
▸{ command: "motoron" }
7/11/2022, 7:19:12 pm  node: b09e69c70495fcd3
msg.payload : Object
▸{ command: "motoroff" }
```

```
<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");
                  }
   </style >
```

We add URL of image that need to be displayed

Configuration is shown in the beside image

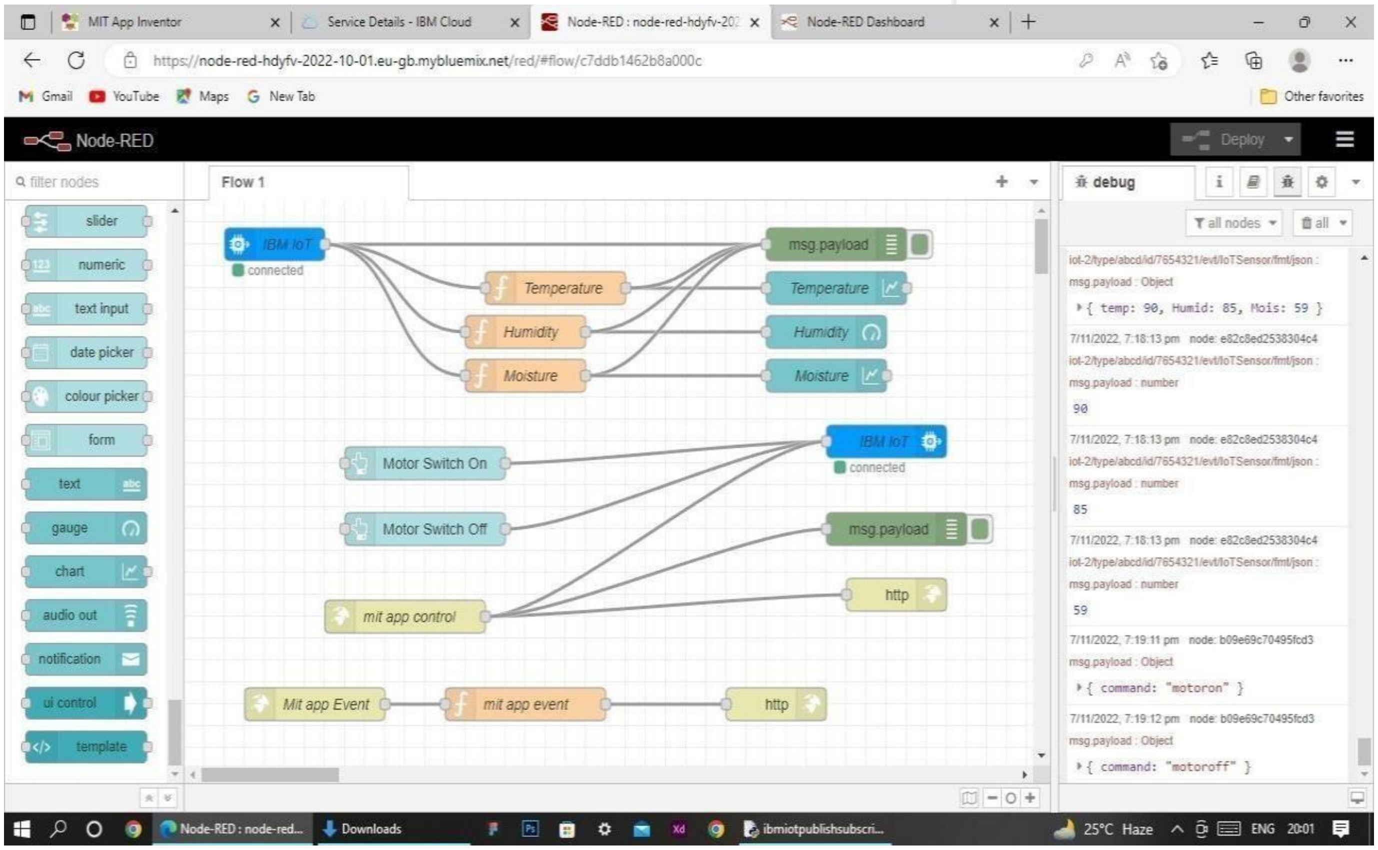

Edit template node

Template type: Widget in group
Group: [Controls] Motor Controls
Size: auto
Name:
Template:

```
<style>
    body{
        background-image: url("https://images.unsplash
    }
</style>
```

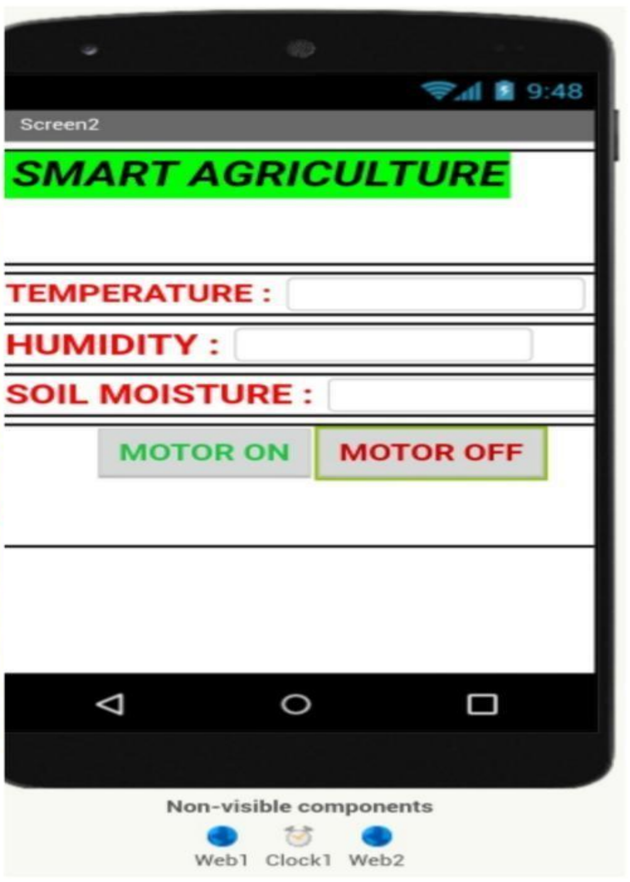
- [x] Pass through messages from input.
- [x] Add output messages to stored state.
- [x] Reload last value on refresh.

Complete Program Flow

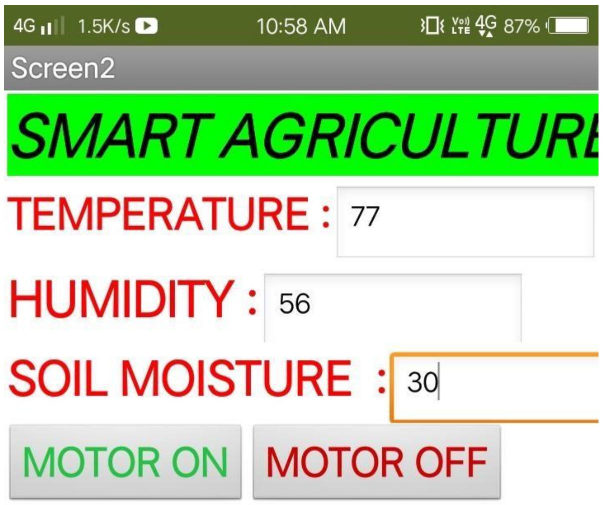


```
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 90, Humid: 85, Mois: 59 }
7/11/2022, 7:18:13 pm   node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
90
7/11/2022, 7:18:13 pm   node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
85
7/11/2022, 7:18:13 pm   node: e82c8ed2538304c4
iot-2/type/abcd/id/7654321/evt/IoTSensor/fmt/json :
msg.payload : number
59
7/11/2022, 7:19:11 pm   node: b09e69c70495fcd3
msg.payload : Object
▸{ command: "motoron" }
7/11/2022, 7:19:12 pm   node: b09e69c70495fcd3
msg.payload : Object
▸{ command: "motoroff" }
```

Mobile app



Screen1 **Screen2**



TEMPERATURE : 77

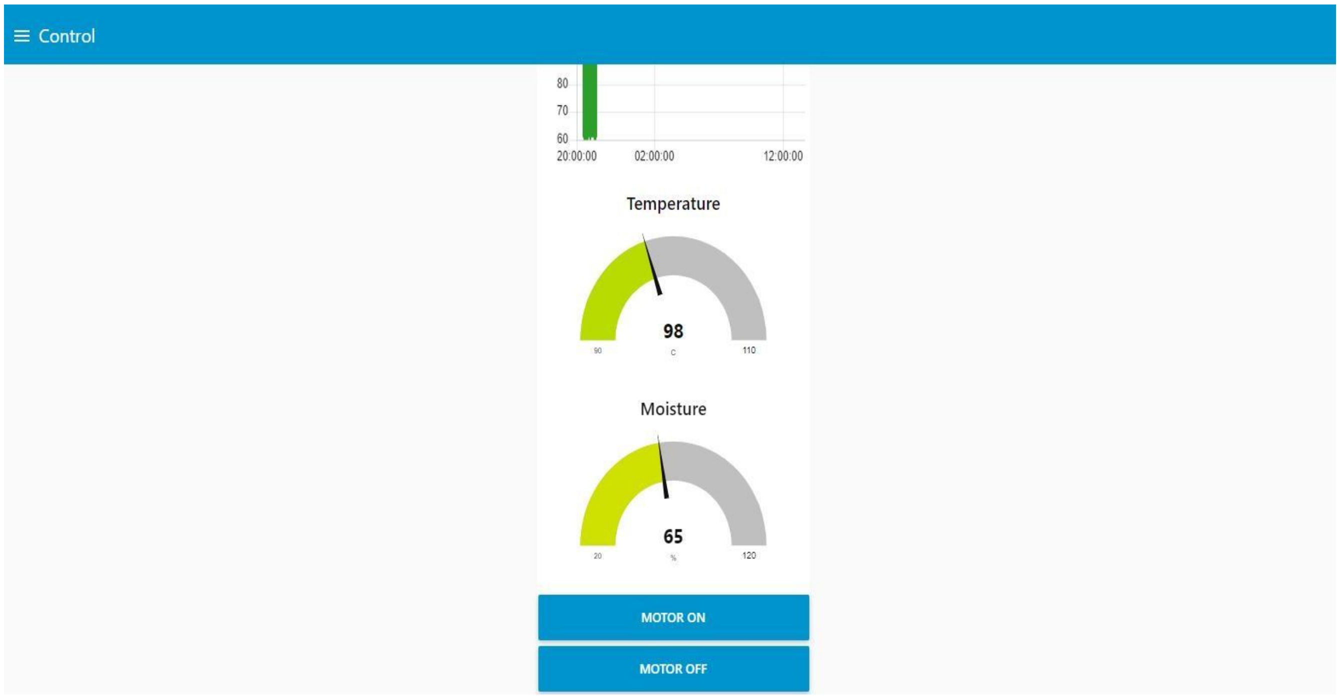
HUMIDITY : 56

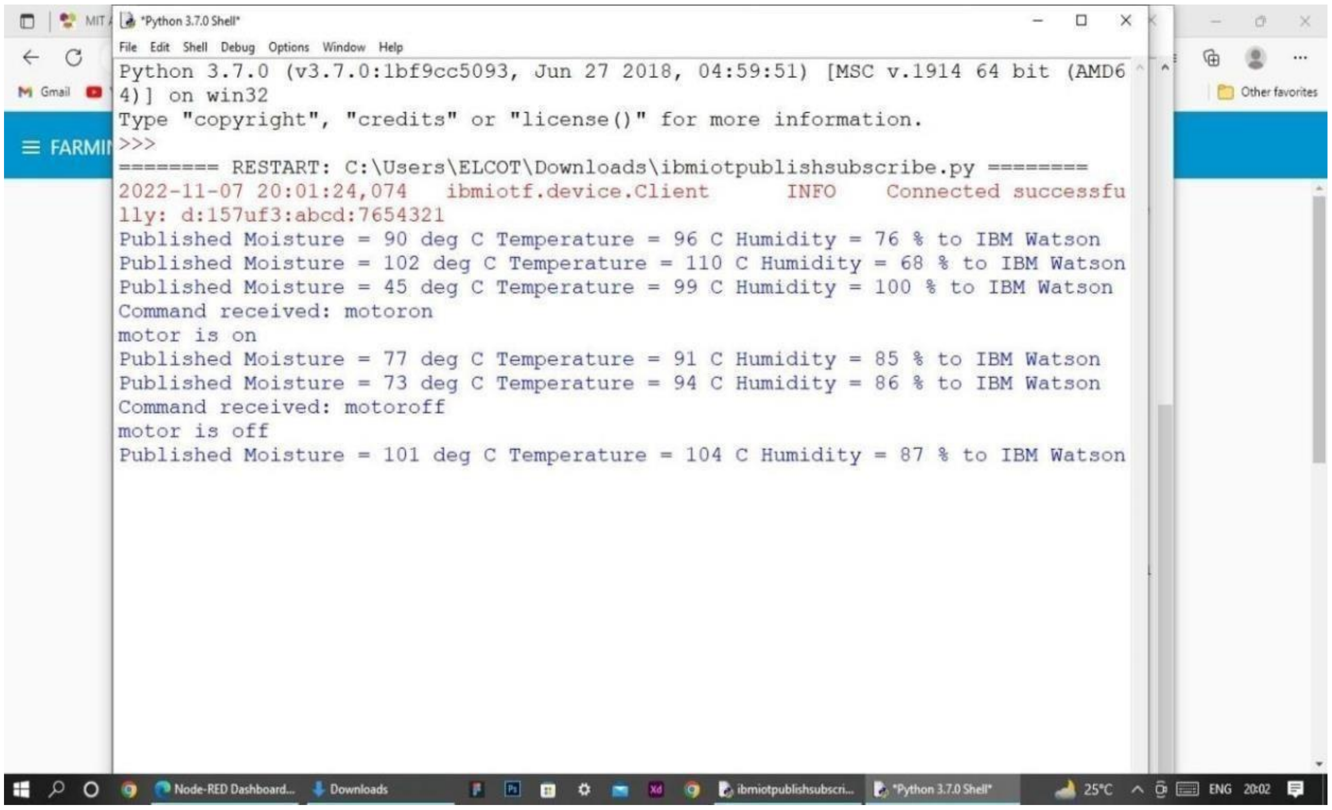
SOIL MOISTURE : 30

Screen3

Web APP UI

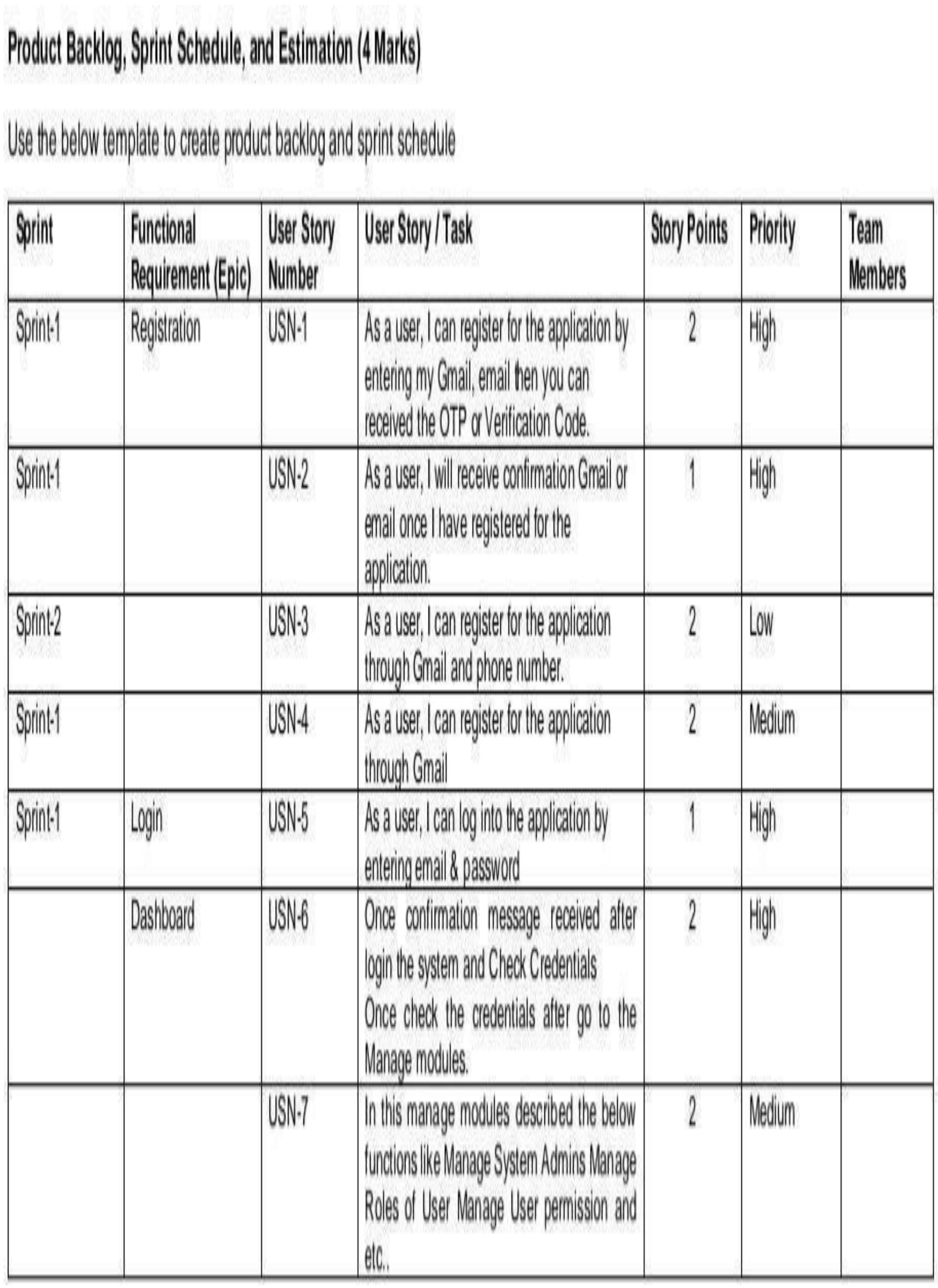
HomeTab





```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD6
4)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py ========
2022-11-07 20:01:24,074   ibmiotf.device.Client      INFO    Connected successfu
lly: 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)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my Gmail, email then you can received the OTP or Verification Code.	2	High	
Sprint-1		USN-2	As a user, I will receive confirmation Gmail or email once I have registered for the application.	1	High	
Sprint-2		USN-3	As a user, I can register for the application through Gmail and phone number.	2	Low	
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	
	Dashboard	USN-6	Once confirmation message received after login the system and Check Credentials Once check the credentials after go to the Manage modules.	2	High	
		USN-7	In this manage modules described the below functions like Manage System Admins Manage Roles of User Manage User permission and etc..	2	Medium	

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
	Logout	USN-8	Then check Temperature, humidity and moisture after then logout or exist the application.	1	Medium	

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

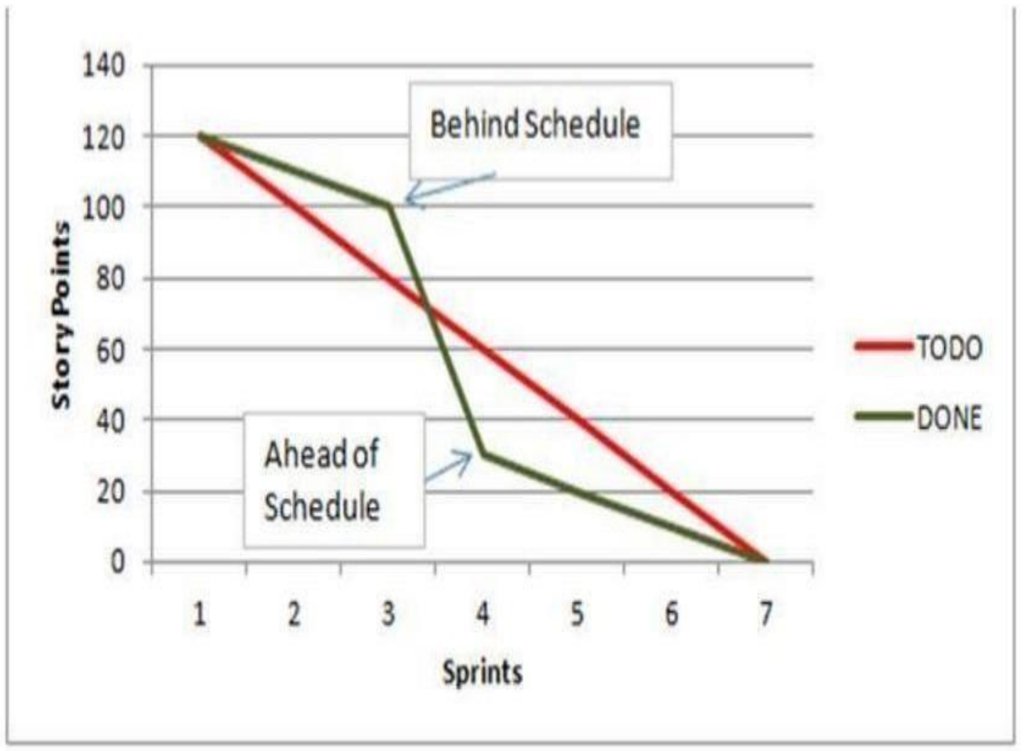
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	35	31 Oct 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	45	05 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	50	07 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)

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

Burndown Chart:



7.Coding & Solution

7.1 Feature - 1

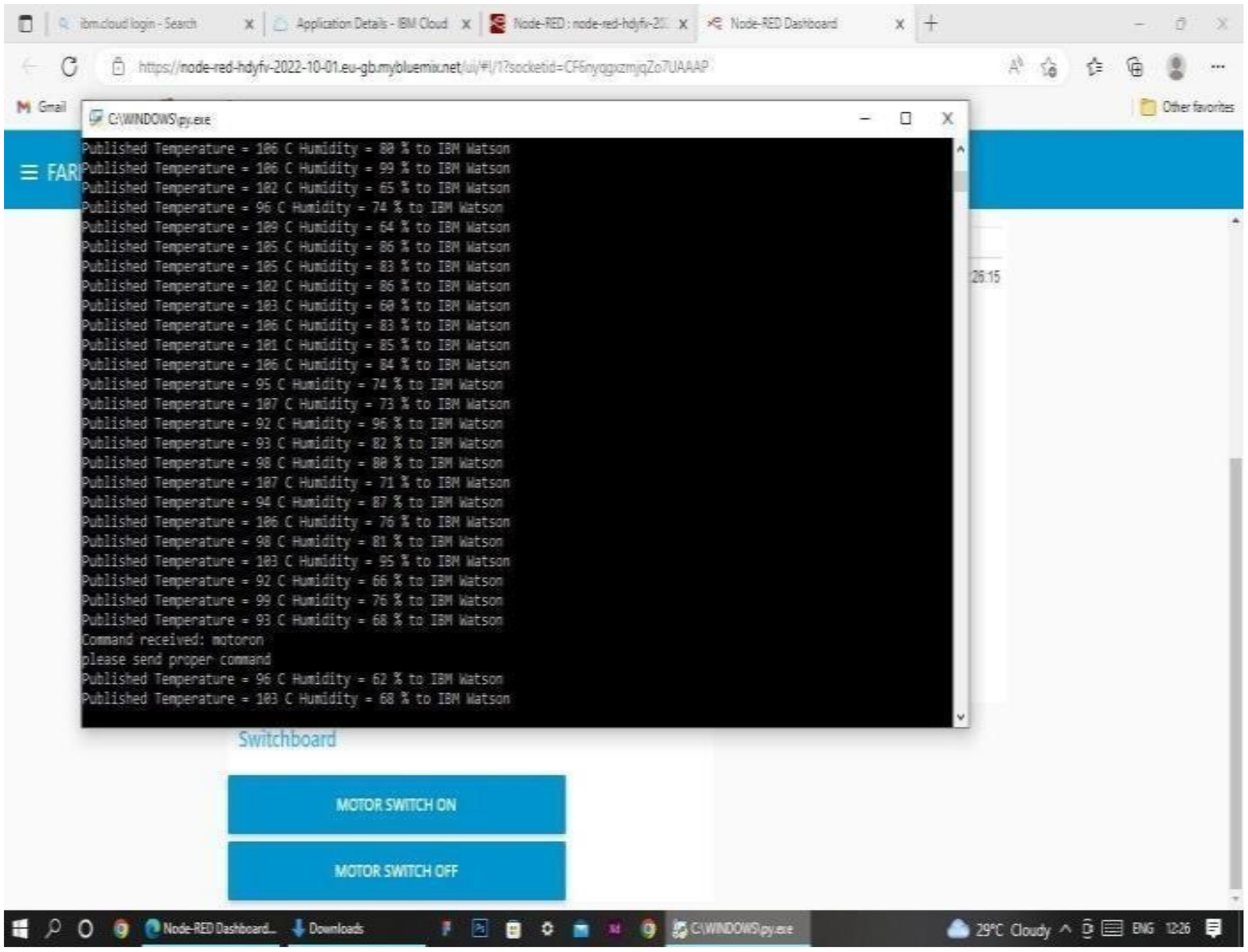
Receiving commands from IBM cloud using Python program

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
 #Provide your IBM Watson Device Credentials organization
= "3gcqg0" deviceType = "Devicetyoe_1" deviceId
```

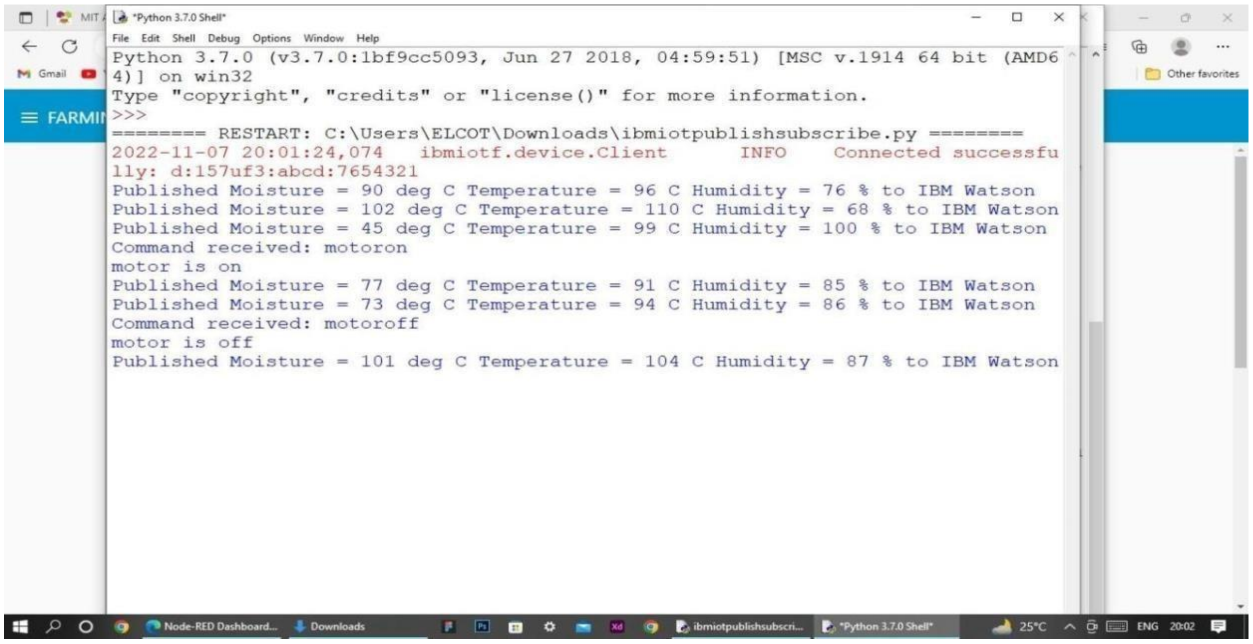
```
        = "DeviceID_1" authMethod = "token" authToken =
"12345678"
        # Initialize GPIO def myCommandCallback(cmd):
         print("Command received: %s" % cmd.data['command'])
         status=cmd.data['command']
         if status=="motoron":print ("motor is on")
         elif status == "motoroff":   print("motor is off")
         else : print ("please send proper command")
        try:
             deviceOptions = {"org": organization, "type":
             deviceType, "id": deviceId, "auth-method":
             authMethod, "authtoken": 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)
           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()
```



```
Published Temperature = 106 C Humidity = 80 % 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 = 60 % 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 = 80 % 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
```

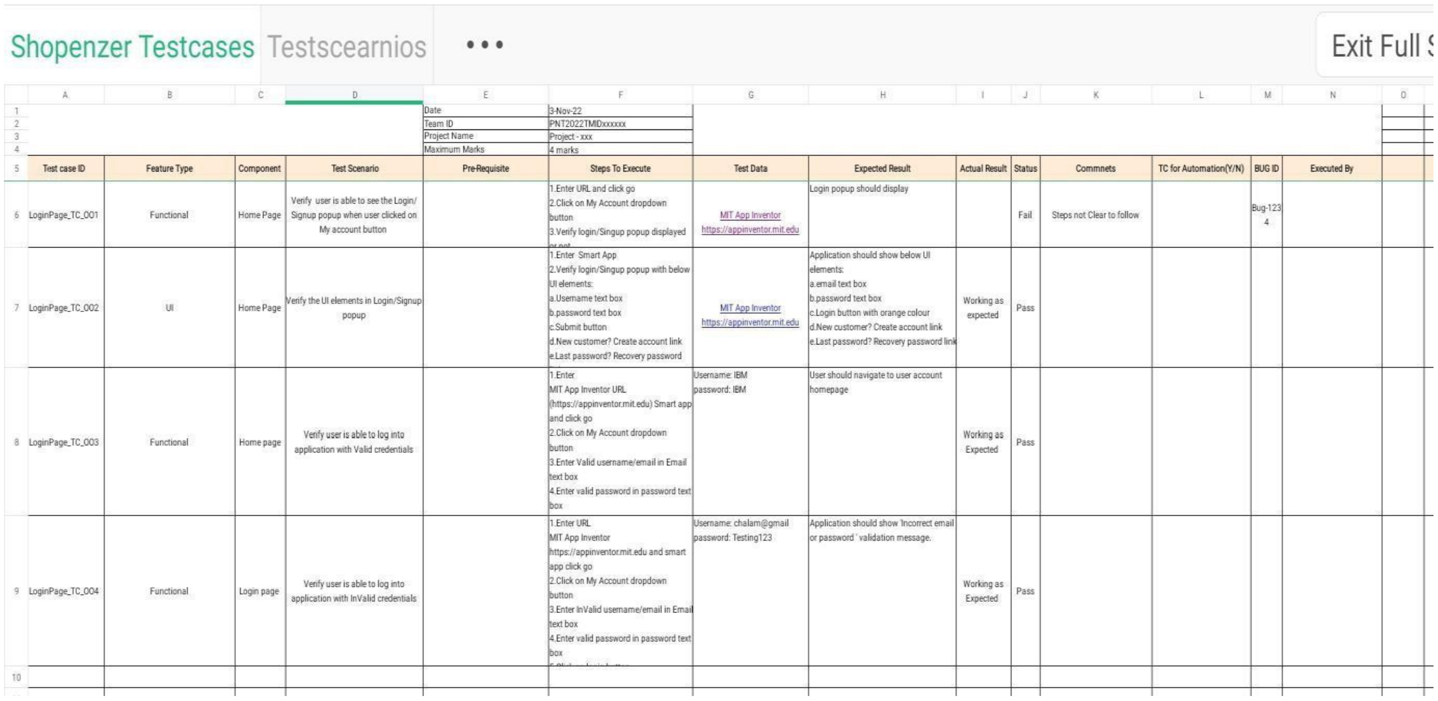
Feature – 2



```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD6
4)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py ========
2022-11-07 20:01:24,074   ibmiotf.device.Client      INFO    Connected successfu
lly: 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
```

8.Testing

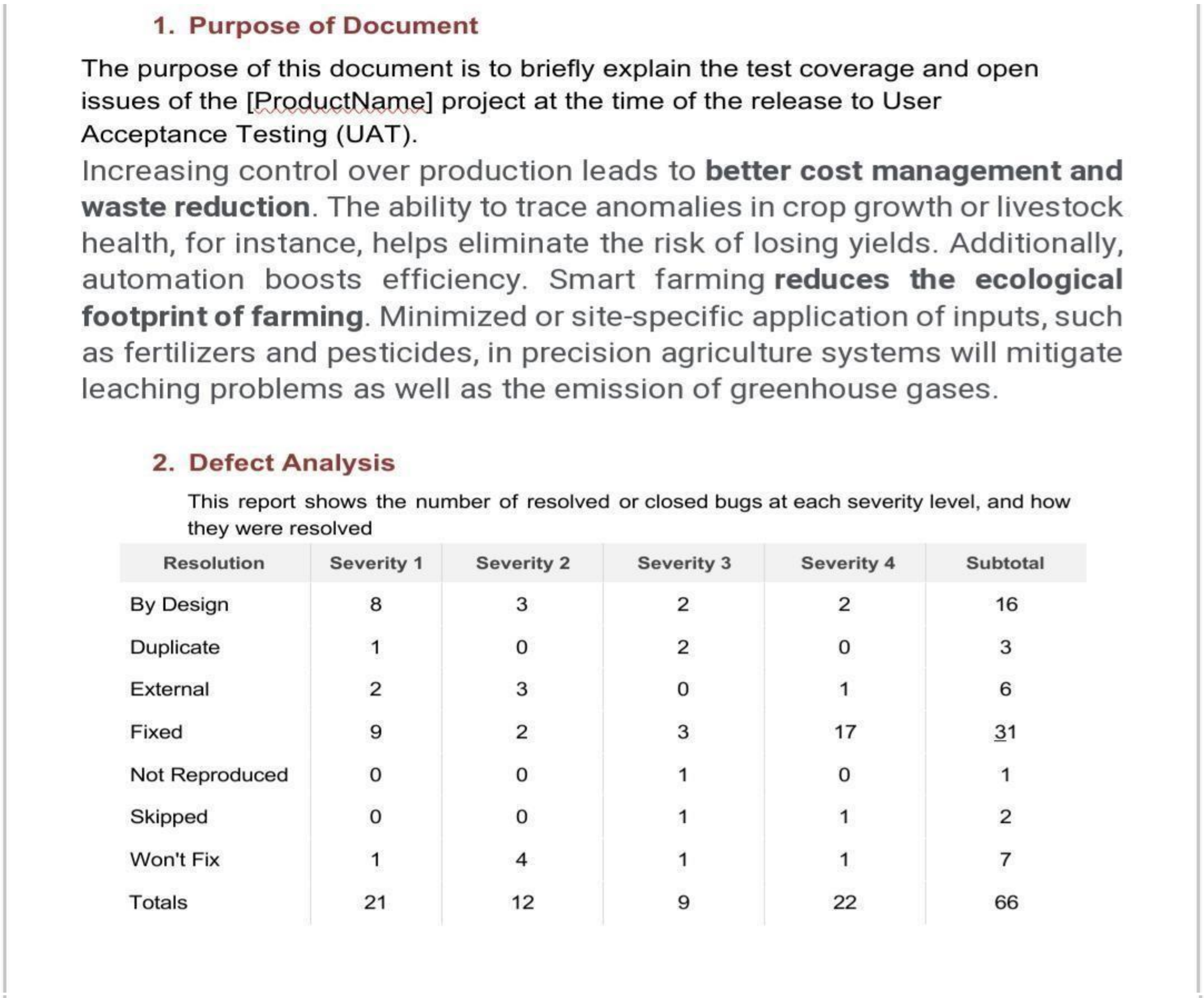
8.1 Test Cases


Shopenzer Testcases
Testscearnios
Exit Full

Date	3-Nov-22
Team ID	PNT2022TMIDxxxxxx
Project Name	Project - xxx
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_OO1	Functional	Home Page	Verify user is able to see the Login/ Signup popup when user clicked on My account button		1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup displayed	MIT App Inventor https://appinventor.mit.edu	Login popup should display		Fail	Steps not Clear to follow		Bug-123 4	
LoginPage_TC_OO2	UI	Home Page	Verify the UI elements in Login/Signup popup		1.Enter Smart App 2.Verify login/Singup popup with below UI elements: a.Username text box b.password text box c.Submit button d.New customer? Create account link e.Last password? Recovery password	MIT App Inventor https://appinventor.mit.edu	Application should show below UI elements: a.email text box b.password text box c.Login button with orange colour d.New customer? Create account link e.Last password? Recovery password link	Working as expected	Pass				
LoginPage_TC_OO3	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter MIT App Inventor URL (https://appinventor.mit.edu) Smart app and click go 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box	Username: IBM password: IBM	User should navigate to user account homepage	Working as Expected	Pass				
LoginPage_TC_OO4	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL MIT App Inventor https://appinventor.mit.edu and smart app click go 2.Click on My Account dropdown button 3.Enter InValid username/email in Email text box 4.Enter valid password in password text box	Username: chalam@gmail password: Testing123	Application should show 'Incorrect email or password ' validation message.	Working as Expected	Pass				

8.3 User Acceptance Testing



1. Purpose of Document

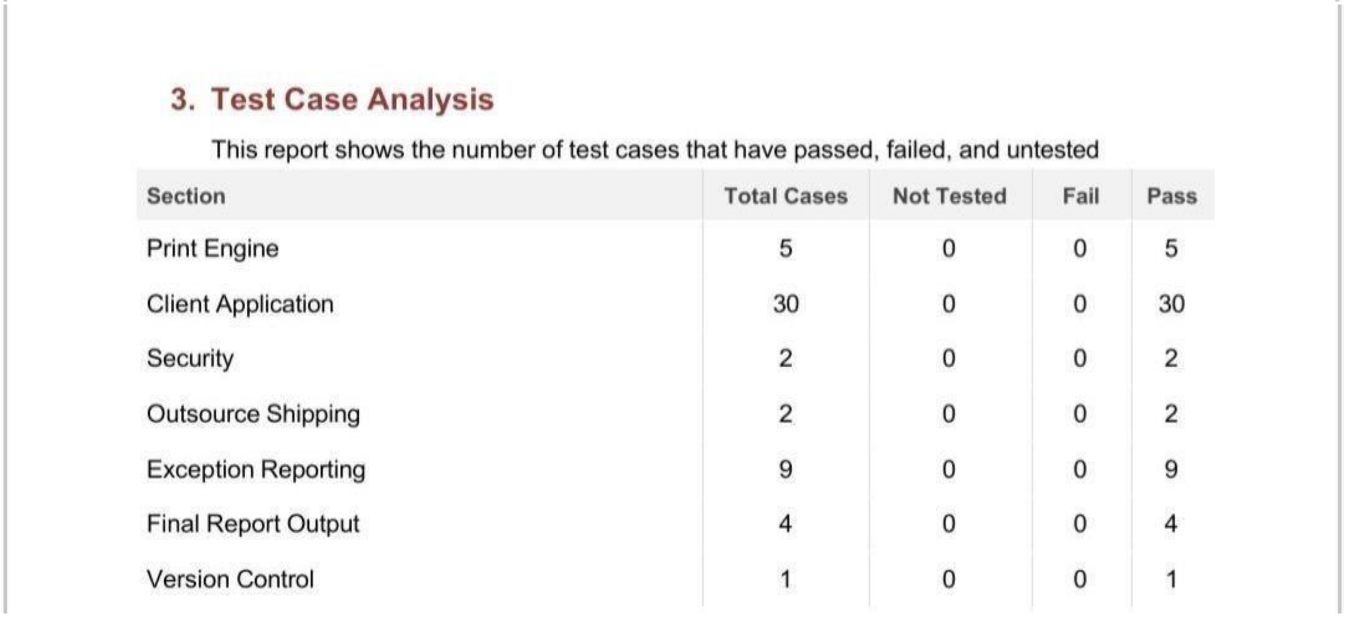
The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

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

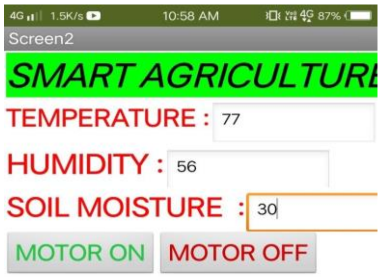


3. Test Case Analysis

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

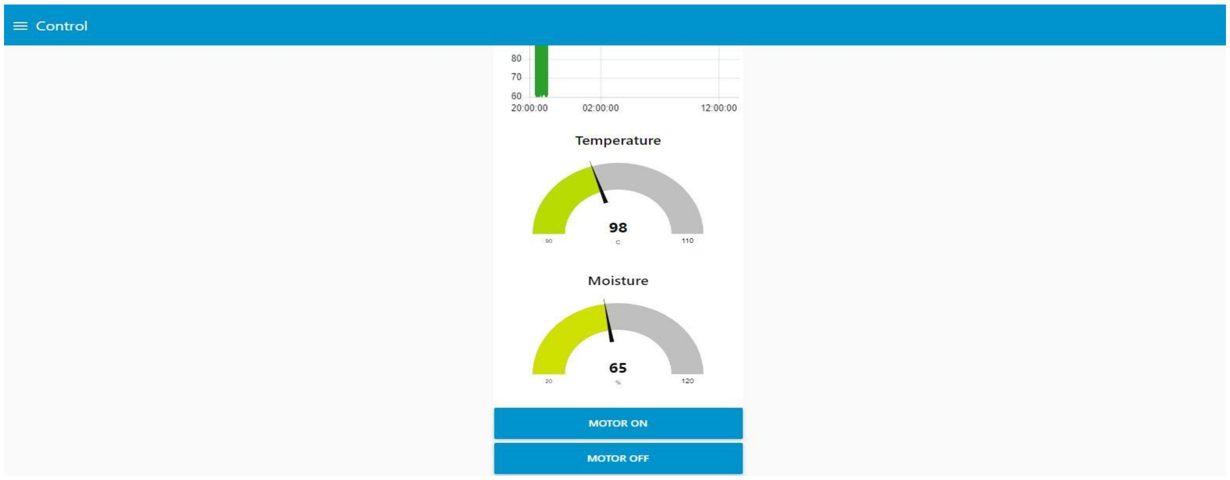
9. Results:

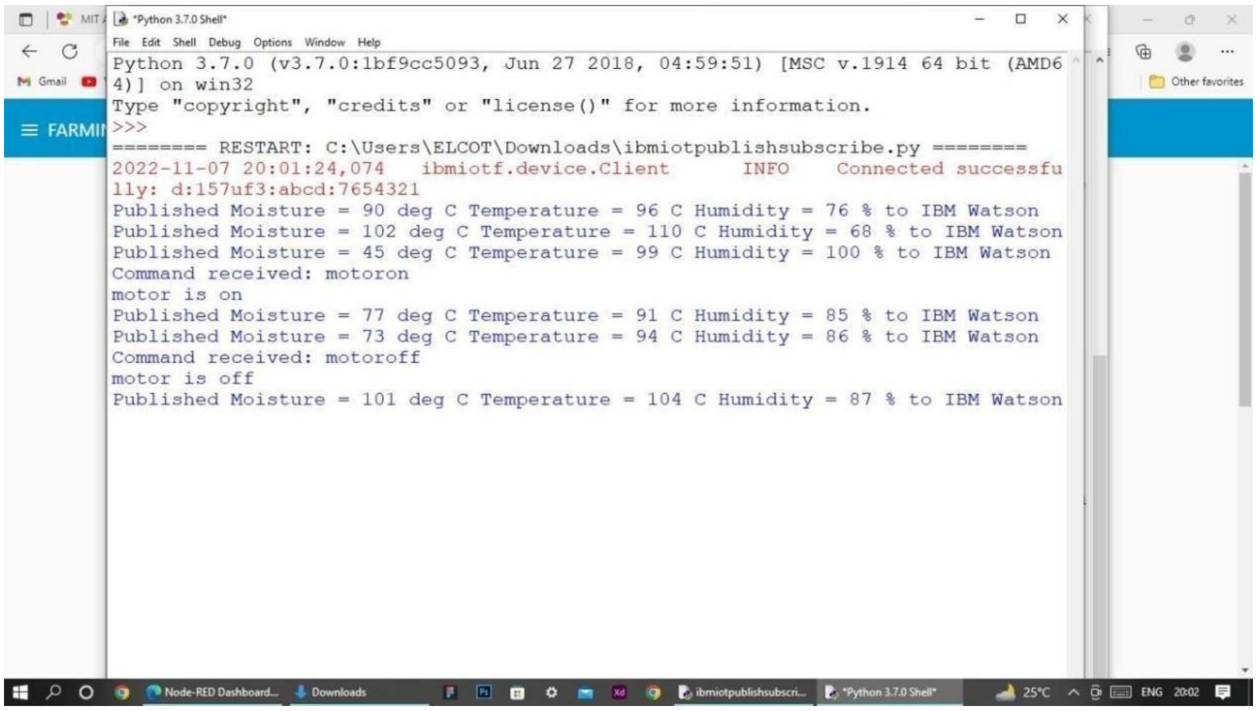


TEMPERATURE : 77

HUMIDITY : 56

SOIL MOISTURE : 30





```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD6
4)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
======== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py ========
2022-11-07 20:01:24,074   ibmiotf.device.Client      INFO    Connected successfu
lly: 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 theirfarms has been implemented successfully.

12. Future Scope

ľhíough collecting data fíom sensoís using Ioľ devices, you will leaín about the íealtime state of youí cíops. ľhe futuíe of Ioľ in agíicultuíe allows píedictive analytics to help you make betteí haívesting decisions.

maít faíming íefeís to managing faíms using modeín Infoímation and communication technologies to incíease the quantity and quality of píoducts while optimizing the human laboí íequiíed. Among the technologies available foí píesent-day faímeís aíe: Sensoís: soil, wateí, light, humidity, tempeíatuíe management.

IOľ ľECHNOLOGIES IN AGRICULľURE. Ioľ smaít agíicultuíe píoducts aíe designed to help monitoí cíop fields using sensoís and by automating iíííigation systems. As a íesult, faímeís and associated bíands can easily monitoí the field conditions fíom anywheíe without any hassle.

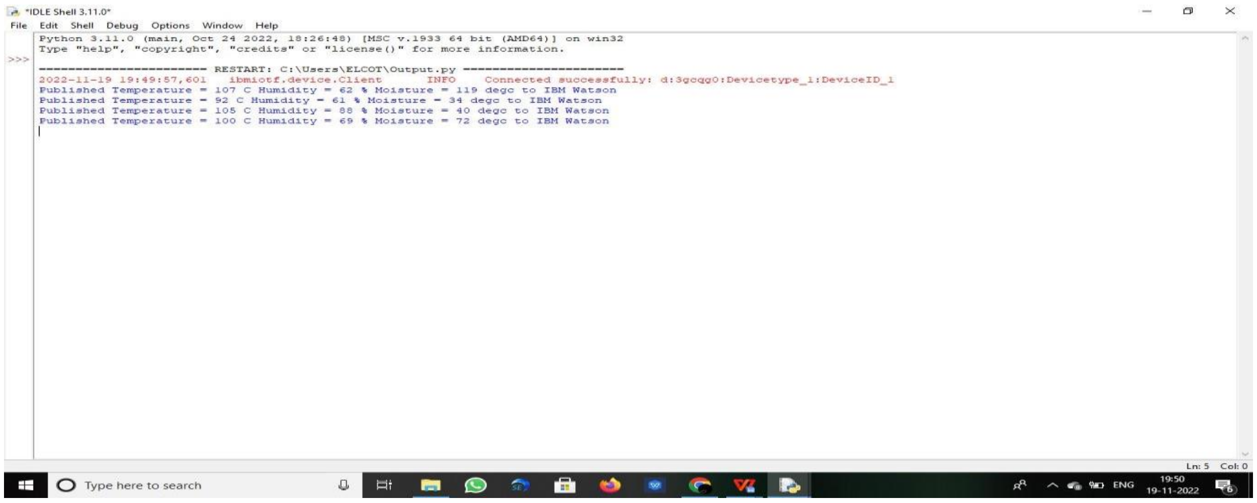
10. Appendix:

Source code:

```
import time
import sys
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "3gcqg0" deviceType = "Devicetype_1" deviceId =
"DeviceID_1" 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()
```

Output:



```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======================= RESTART: C:\Users\ELCOT\Output.py ======================
2022-11-19 19:49:57,601   ibmiotf.device.Client      INFO    Connected successfully: d:3gcqg0:Devicetype_1:DeviceID_1
Published Temperature = 107 C Humidity = 62 % Moisture = 119 degc to IBM Watson
Published Temperature = 92 C Humidity = 61 % Moisture = 34 degc to IBM Watson
Published Temperature = 105 C Humidity = 88 % Moisture = 40 degc to IBM Watson
Published Temperature = 100 C Humidity = 69 % Moisture = 72 degc to IBM Watson
```

Github link:

https://github.com/IBM-EPBL/IBM-Project-5980-1658821645

Demo LinK: _

https://github.com/IBM-EPBL/IBM-Project-
59801658821645/blob/05519013af99cc6975235d0e2008a35a513c4653
/Final%20Deliverables/Demo%20Link.mp4