SMART FARMER-IOT ENABLED SMART FARMING APPLICATION

PROJECT REPORT

IBM-Project-13788-1659530472

TEAM ID: PNT2022TMID08469

Submitted by

KURUBA ASHOK	**(810419106028)**
BATHULA HARSHA KUMAR	**(810419106006)**
MULLANGI THIRUPATHI REDDY	**(810419106035)**
MUMMALANENI GIREESH NAG	**(810419106036)**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE

(AUTONOMOUS)

PERAMBALUR-621212

PROJECT REPORT

1. INTRODUCTION ………………………………………………………01

1.1 Project Overview

1.2 Purpose

2. LITERATURE SURVEY…………………………………………….02

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION ………………………...03

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS …………………………………….09

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN ……………………………………………….11

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING………………………16

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. CODING & SOLUTIONING ……………………………………..18

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

8. TESTING……………………………………………………………….21

8.1 Test Cases

8.2 User Acceptance Testing

9. RESULTS ………………………………………………………………24

9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES………………………….25

11. CONCLUSION ……………………………………………………….25

12. FUTURE SCOPE …………………………………………………….25

13. APPENDIX ………………………………………………………...26

Source Code And GitHub

Project Demo Link

1.INRODUCTION

1.1 Project Overview

IOT- Enabled Smart Farming agriculture system helps the farmer in monitoring different parameters of his field like soil moisture, temperature, humidity using some sensors. Farmer 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 task for the farmers. They can make the decision whether to water the crop or postpone it by monitoring the sensor parameters and control the motor pumps from the mobile application itself. All the sensor parameters are stored in the IBM Cloudant DB

IoT is network that connects physical objects or things embedded with electronics, software and sensors through network connectivity that collects and transfers data using cloud for communication. Data is transferred through internet without human to human or human to computer interaction. In this project we have not used any hardware. Instead of real soil and temperature conditions, sensors IBM IoT Simulator is used which can transmit soil moisture temperature as required..

Project requirements: Node-RED, IBM Cloud, IBM Watson IoT, Node.js, IBM Device, IBM IoT Simulator, Python 3.7, Open Weather API platform.

Project Deliverables: Application for IoT based Smart Agriculture System

1.2 Purpose

IoT based farming improves the entire agriculture system by monitoring the field in real-time. With the help of IoT in agriculture not only saves the time but also reduces the extravagant use of resources such as water and electricity. Sometimes due to over or less supply of water in the agricultural field crops may not grow proper. Using IoT supply of water and growth of plants can be satisfied to a greater extent. The flow of water can be controlled from the application.

Smart agriculture is a farming system which uses IoT technology. This emerging system increases the quantity and quality of agricultural products. IoT devices provide information about nature of farming fields and then take action depending on the farmer input**.**

The main goal of my project is to use IoT in the agriculture field in order to collect data instantly (soil Moister, temperature, humidity…), which will help one to monitor some environment conditions remotely, effectively and enhance tremendously the production and therefore the income of farmers. The present prototype is developed using Arduino technology, which comprise specific sensors, and a WIFI module that helps to collect instant data online. Worth mentioning the testing of this prototype generated, highly accurate data because while we were collecting them remotely any environmental changes were detected instantly and taking in consideration to make decisions.

2. LITERATURE SURVEY

2.1 Existing Problem

Watering the field is a difficult process, Farmers have to wait in the field until the water covers the whole farm field. Power Supply is also one of the problems. In Village Side, the power supply may vary. The Biggest Challenges Faced by IoT in the Agricultural Sector are Lack of Information, High Adoption, Cost and Security Concerns, etc The farmers do not have that much knowledge on the internet of things and good internet connection is required. So farmers don’t know how to use the web application and to make a connection if any component get failed.

2.2 References

[1] Divya J., Divya M.,Janani V.”IoT based Smart Soil Monitoring System for Agricultural Production” 2017.

[2] H.G.C.R.Laksiri, H.A.C.Dharmagunawardhana, J.V.Wijayakulasooriya ”Design and Optimization of loT Based Smart Irrigation System in Sri Lanka”2019 .

[3] Anushree Math, Layak Ali, Pruthviraj U ”Development of Smart Drip Irriga- tion System Using IoT”2018.

[4] Dweepayan Mishra1 ,Arzeena Khan2 Rajeev Tiwari3 , Shuchi Upadhay,”Automated Irrigation System-IoT Based Approach”,2018.

[5] R. Nageswara Rao, B.Sridhar,”IOT BASED SMART CROP-FIELD MONI- TORING AND AUTOMATION IRRIGATION SYSTEM”. 2018

[6] Shweta B. Saraf, Dhanashri H. Gawal,”IoT Based Smart Irrigation Monitoring And Controlling System”.2017

[7] Shrihari M, ”A Smart Wireless System to Automate Production of Crops and Stop Intrusion Using Deep Learning” 2020.

[8] G. Sushanth1, and S. Sujatha, ”IOT Based Smart Agriculture System”2018.

[9] Vaishali S, Suraj S, Vignesh G, Dhivya S and Udhayakumar S, ”Mobile Integrated Smart Irrigation Management and Monitoring System Using IOT”,2017

2.3 Problem Statement Definitions

The Biggest Challenges Faced by IoT in the Agricultural Sector are Lack of Information, High Adoption, Cost and Security. The farmers do not have that much knowledge on the internet of things and good internet connection is required. Power Supply is also one of the problems In Village Side, the power supply may vary. So farmers don’t know how to use the web application and to make a connection if any component get failed.

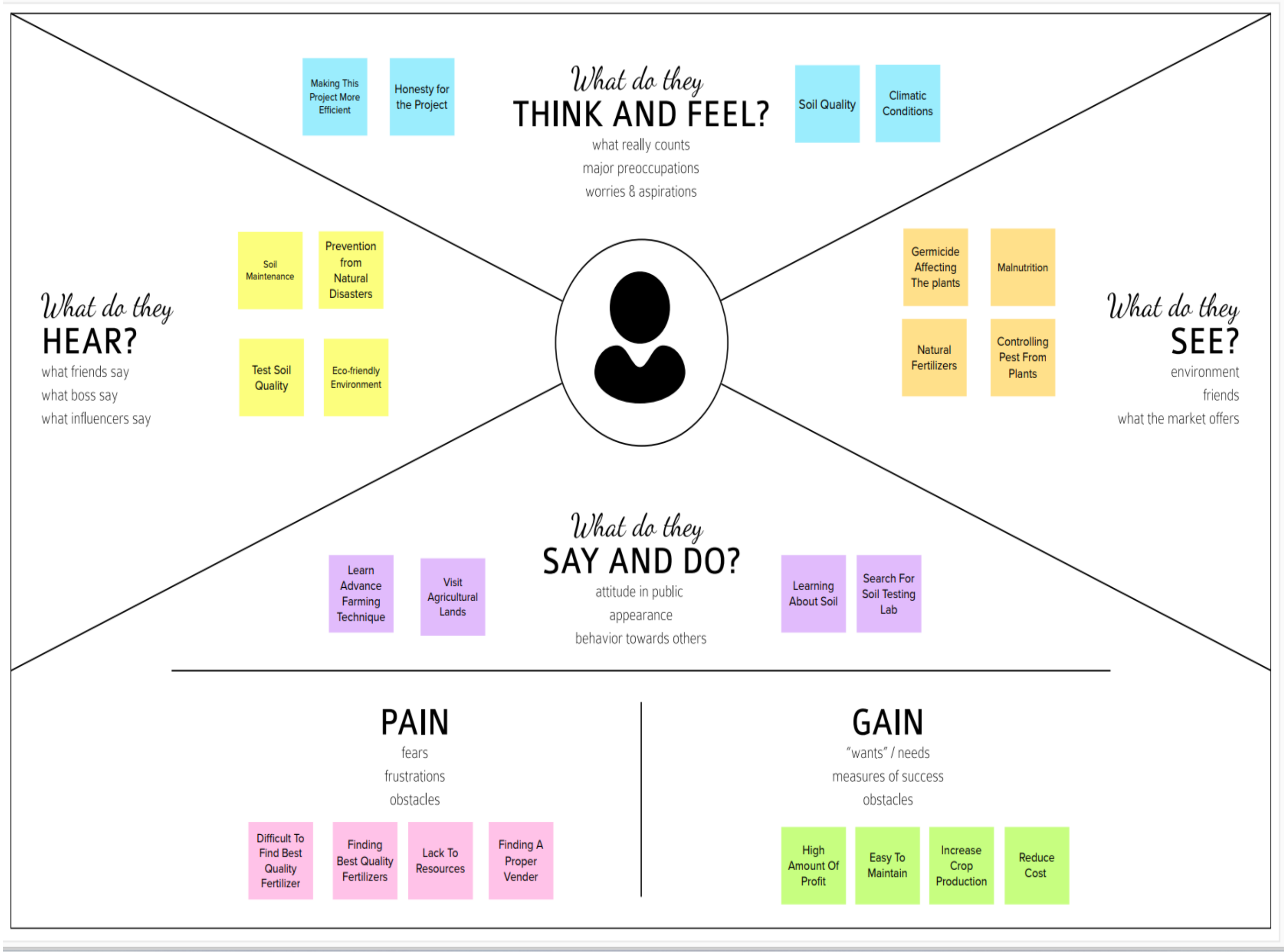
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:

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

It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges

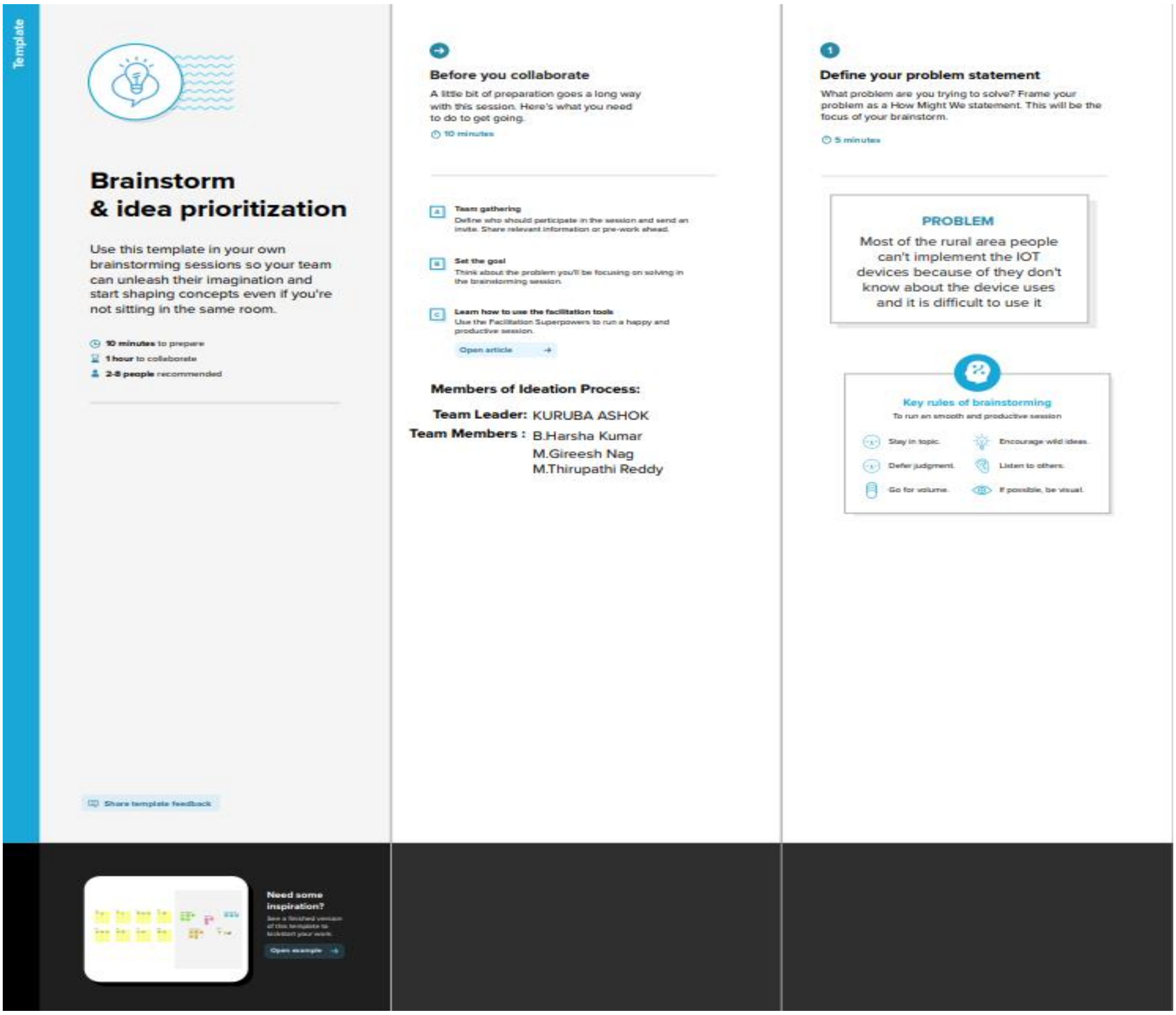
Empathy Map



3.2 Ideation and Brainstorming

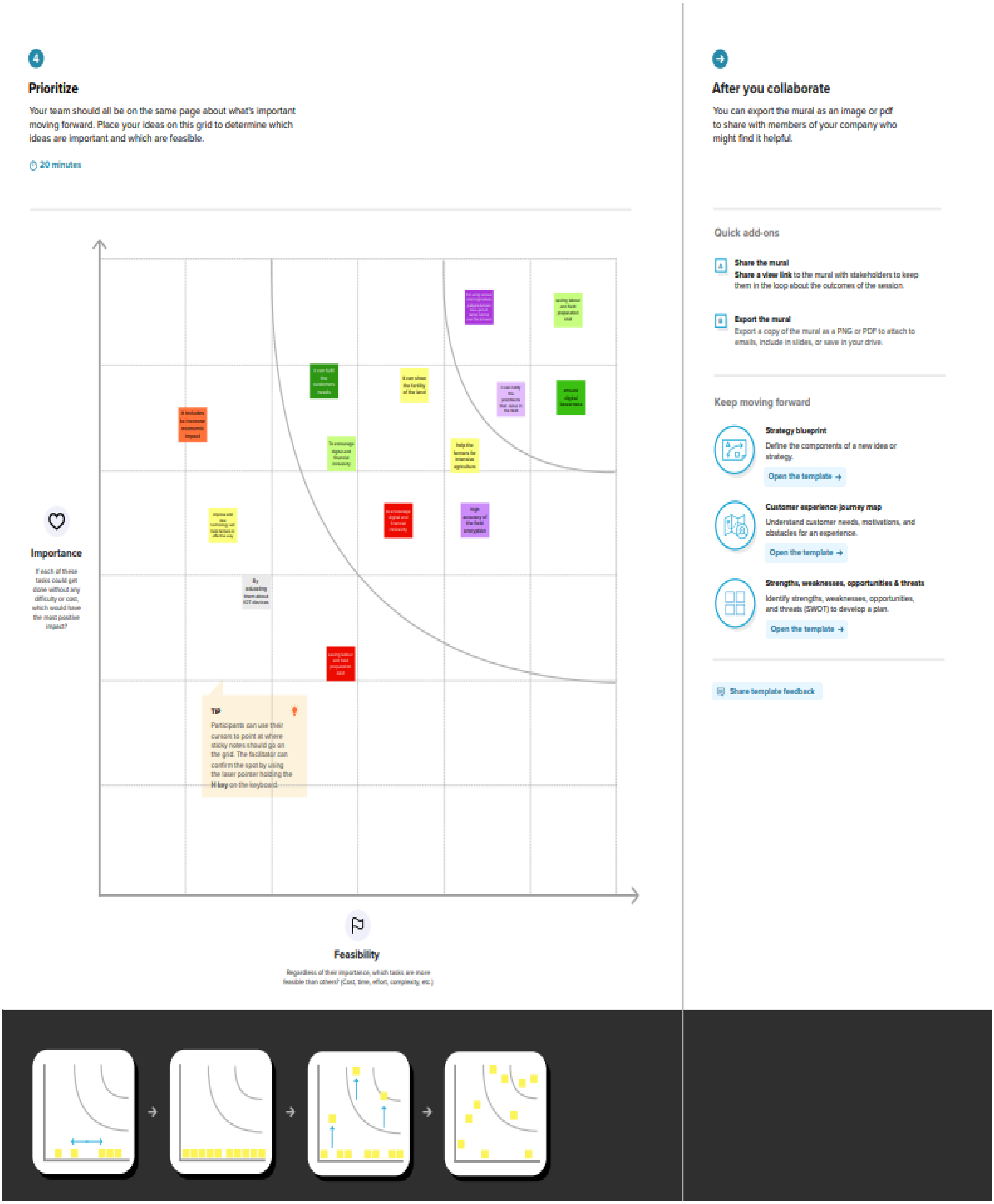
Reference :
https://app.mural.co/invitation/mural/dhanalakshmisrinivasanengine1186/1666587358105?sender=u24641282421e39fff7ab4248&key=929b2e42-e842-48c5-98db-1ae3735cd2bf

Step-1: Team Gathering, Collaboration and Select the Problem Statements:



Step-2: Brainstorm ,Idea Listing and Grouping

Step-3: Idea Prioritization

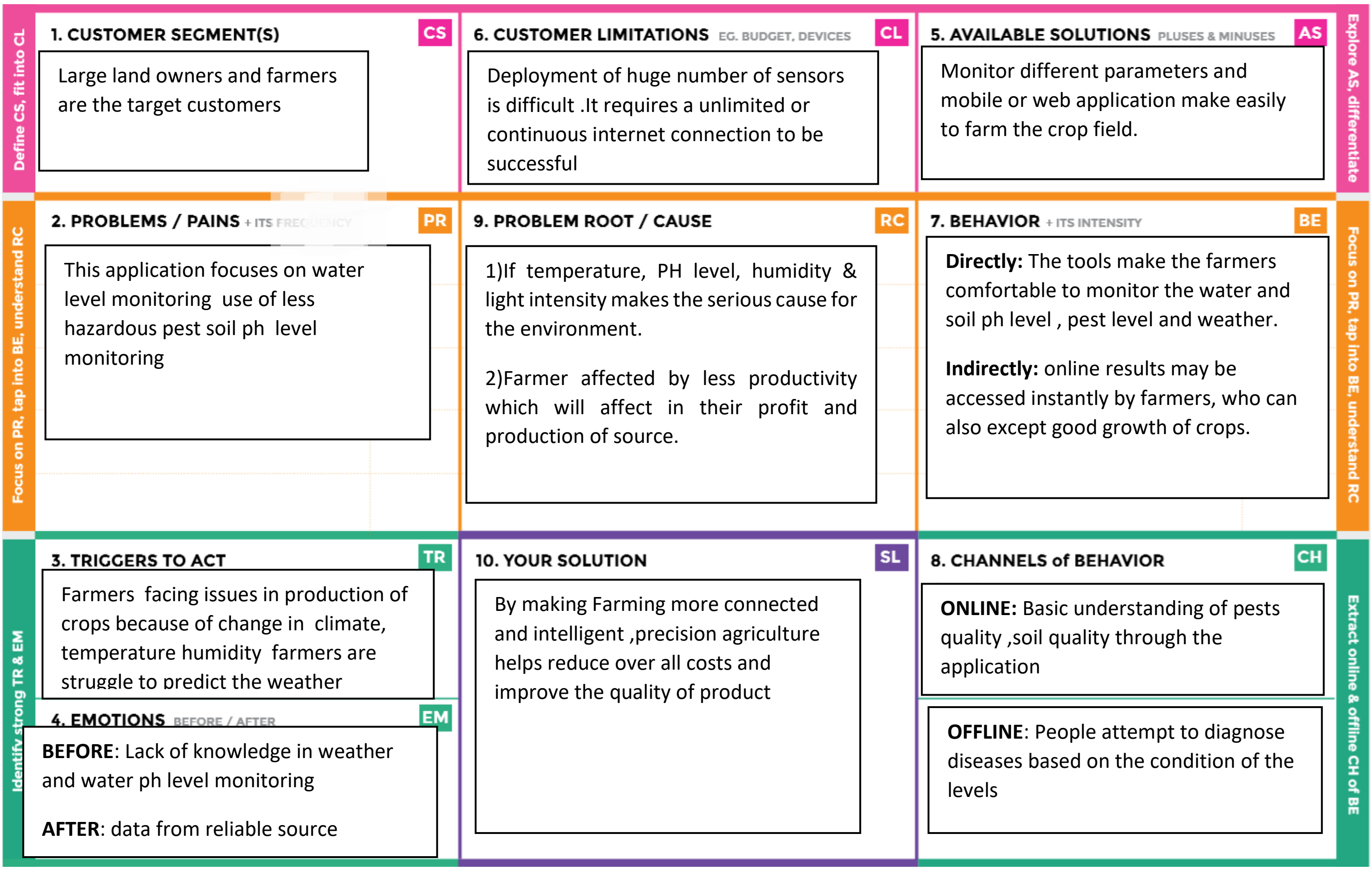


3.3 Proposed Solution:

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	• Watering the field is a difficult process, Farmers have to wait in the field until the water covers the whole farm field. • Power Supply is also one of the problems. In Village Side, the power supply may vary. • The Biggest Challenges Faced by IoT in the Agricultural Sector are Lack of Information, High Adoption, Cost and Security Concerns, etc
2.	Idea / Solution description	• As is the case of precision Agriculture Smart Farming Technique Enables Farmers better to monitor the fields and maintain the humidity level accordingly. • The Data collected by sensors, In terms of humidity, temperature, moisture, and dew detections help in determining the weather pattern in Farms. So cultivation is done for suitable crops.
3.	Novelty / Uniqueness	**ALERT MESSAGE** – IoT sensor nodes collect information from the farming environment, such as soil moisture, air humidity, temperature, nutrient ingredients of soil, pest images, and water quality, then transmit collected data to IoT backhaul devices. **REMOTE ACCESS –** It helps the farmer to operate the motor from anywhere.
4.	Social Impact / Customer Satisfaction	• Reduces the wages for labors who work in the agricultural field. • It saves a lot of time. • IoT can help improve customer relationships by enhancing the customer's overall experience. • Easily identify maintenance needs, build better products, send personalized communications, and more. • IoT can also help e-commerce businesses thrive and increase sales. • It make a wealthy society

5.	Business Model (Revenue Model)	The project involves Thermography sensors which is cheaper than the existing ideas
6.	Scalability of the Solution	Scalability in smart farming refers to the adaptability of a system to increase the capacity, for example, the number of technology devices such as sensors and actuators, while enabling timely analysis.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 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	Registration through Gmail Registration through phone number
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP Confirmation via verification link sent to registered mail id
FR-3	Roles and service	Choose roles (ex: farmer, student etc.) Enter the personal details. Choose the type of service or options (ex: irrigation, pest management, crop management etc.)
FR-4	Terms and conditions	Accepts the terms and condition for the chosen role and options
FR-5	Details of farm and plans	Enter the details of farming land and vegetation. Choose the crop you want to plant Choose the types of plans (ex: regular and premium)
FR-6	Details according to farm information	Check the weather information Enter the soil nutrient and pH value Click SAVE Soon the details will share to registered mail Exit

4.2 Non-Functional Requirements:

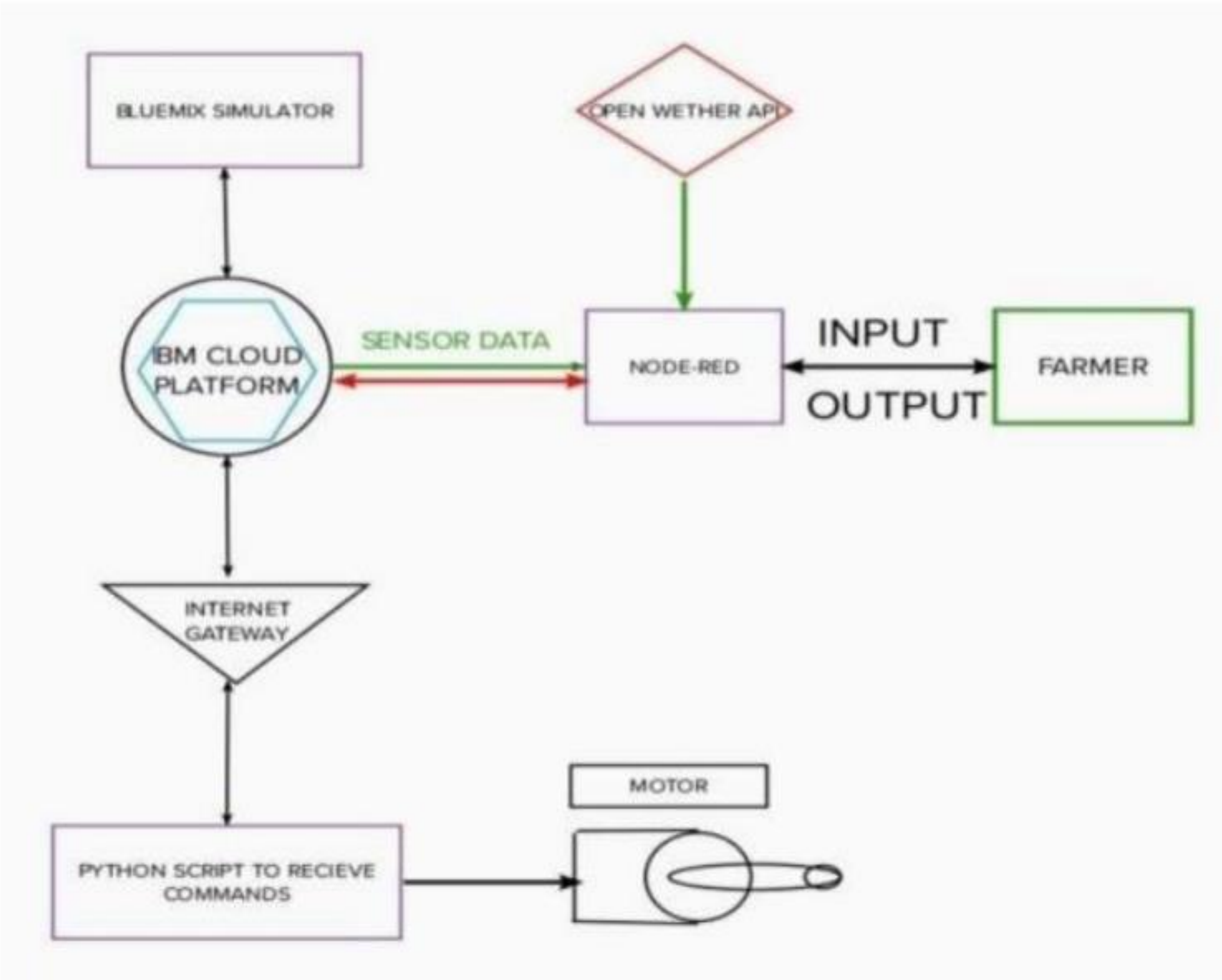
Following are the non-functional requirements of the proposed solution.

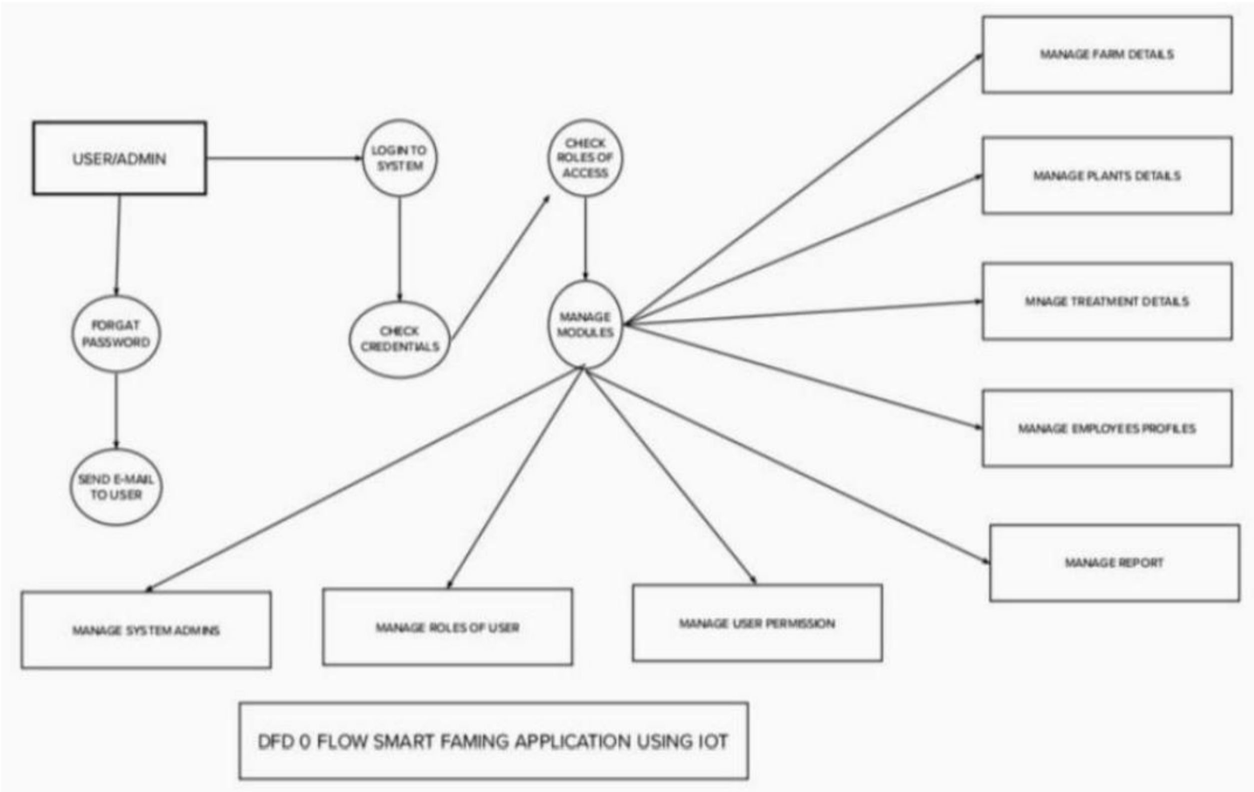
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	A system is built for monitoring the crop field with the help of sensors and automating the irrigation system and helps the farmer to understand the important aspects.
NFR-2	Security	Applications must be designed with the security of their use in mind. This includes personal data and their user’s well-being.
NFR-3	Reliability	It allows farmers to maximize yields using minimum resources such as water, fertilizers, seeds etc.
NFR-4	Performance	It increases efficiency and reduce the environmental impacts and to implement technology properly to minimize cost.
NFR-5	Availability	This concept focused on providing the agricultural industry with the infrastructure to leverage advanced technology.
NFR-6	Scalability	It provides the recognition of each object that makes up a solution and ensure communication. The system must remain operational regardless.

5.PROJECT DESIGN

5.1 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.





1. The different soil parameters temperature, soil moistures and then humidity are sensed using different sensors and obtained value is stored in the Ibm cloud.

2. Arduino UNO is used as a processing Unit that process the data obtained from the sensors and whether data from the weather API.

3. NODE-RED is used as a programming tool to write the hardware, software and APIs. The MQTT protocol is followed for the communication.

4. All the collected data are provided to the user through a mobile application that was developed using the MIT app inventor. The user could make a decision through an app, weather to water the crop or not depending upon the sensor values. By using the app they can remotely operate to the motor switch.

5.2 Solution & Technical Architecture

Solution Architecture:

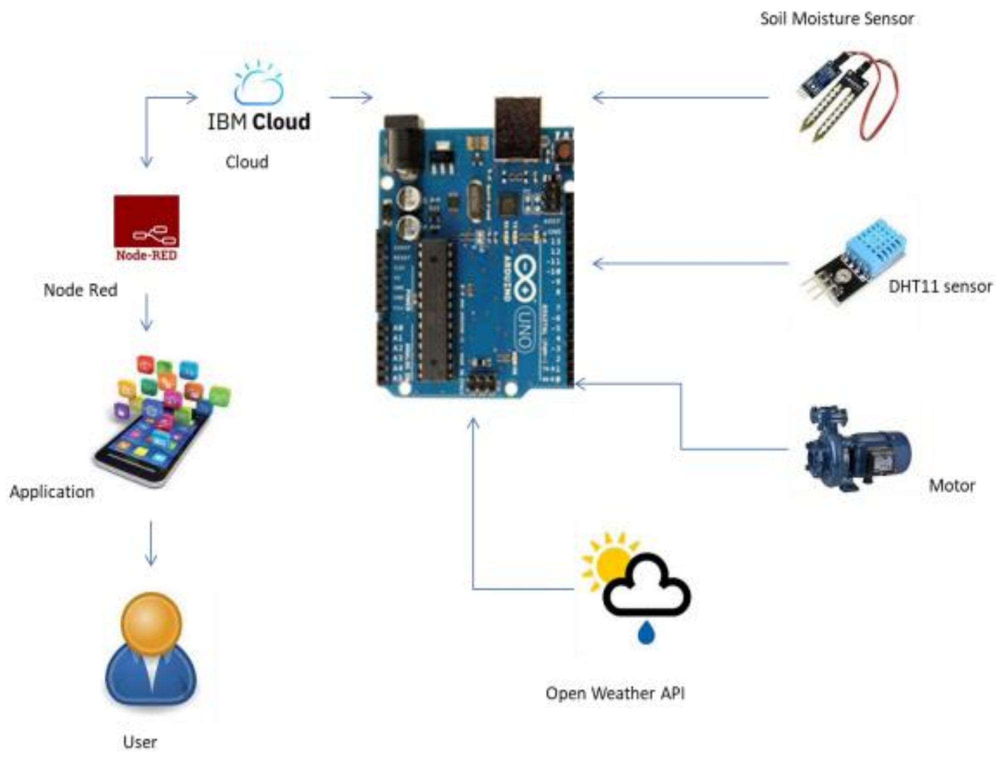
1.The different soil parameters (temperature, humidity, light intensity, pH level) are sensed using different sensors and the obtained value is stored in IBM cloud.

2. Arduino UNO is used as a processing unit which processes the data obtained fromsensors and weather data from weather API.

3. Node red is used as a programming tool to wire the hardware, software and APIs.The MQTT protocol is followed for communication.

4. All the collected data are provided to the user through a mobile application which was developed using MIT app inventor. The user could make decision through an app, whether to water the crop or not depending upon the sensor values.

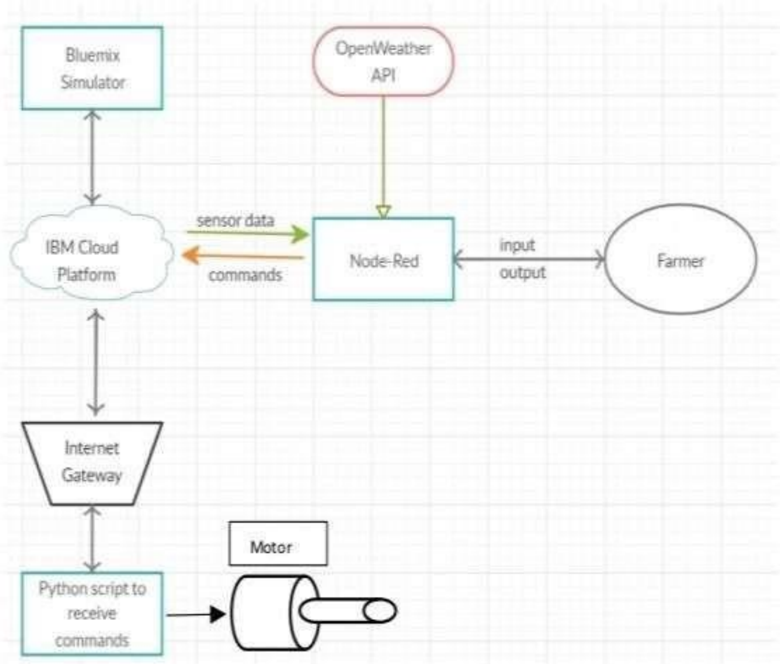
Solution Architecture Diagram:



Technical

Architecture

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2



The different soil parameters temperature, soil moistures and then humidity are sensed using different sensors and obtained value is stored in the IBM cloud.

Arduino 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 decide through an app, weather to water the crop ornot depending upon the sensor values. By using the app, they can remotely operate the motor switch.

5.3 User Stories

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Simulation creation and Python code development	USN-1	Connect Sensors Wi-fi Module with python code and pubsub python code.	2	High	Kuruba Ashok, Harsha Kumar,
Sprint-2	Connecting python code with IBM Watson platform and node-red work flow	USN-2	Creating device in the IBM Watson IoT platform, workflow for IoT scenarios using Node-Red	2	High	Gireesh Nag, Harsha Kumar.
Sprint-3	Creating MIT App Inventor and designing front end like username and password	USN-3	Developing an application for the Smart farmer project using MIT App Inventor	2	High	Kuruba Ashok, Thirupathi Reddy
Sprint-3	Developing the backend of the mit app using blocks	USN-3	Design the Modules and test the app	2	High	Kuruba Ashok, Gireesh Nag,
Sprint-4	Web UI	USN-4	To make the user to interact with software.	2	High	Kuruba Ashok' Harsha Kumar,

6.PROJECT PLANNING & SCHEDULING

6.1 Sprint planning & Estimation

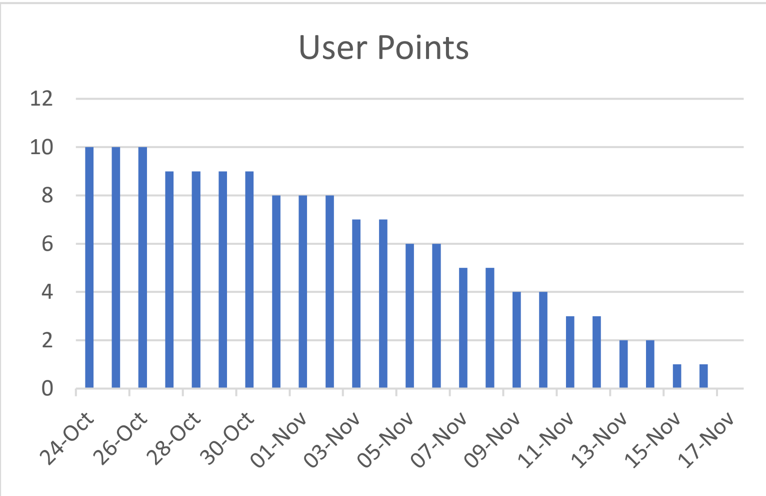
Title	Description	Duration
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers, research publications etc.	29 August-3rd September 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements.	5-10 th September 2022
Brainstorming ideas	List the ideas by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	12-17 September 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	19-24 September 2022
Problem Solution Fit	Prepare problem - solution Fit document.	26 September-01 October 2022
Solution Architecture	Prepare solution Architecture document.	26 September-01 October 2022
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application	03-08 October 2022
Data Flow Diagrams	Draw the data flow Diagrams and submit for review.	10-15 October 2022
Technology Architecture	Architecture diagram.	10-15 October 2022

Milestone & Activity List	Prepare the milestones & Activity list of the project.	17-22 October 2022
Sprint Delivery	Prepare the Sprint delivery on Number of Sprint planning meetings organized, Minutes of meeting recorded.	17-22 October 2022
Project Development Delivery of Sprint- 1,2,3&4	Develop & submit the developed code by testing it.	In Progress

6.2 Sprint Delivery Schedule:

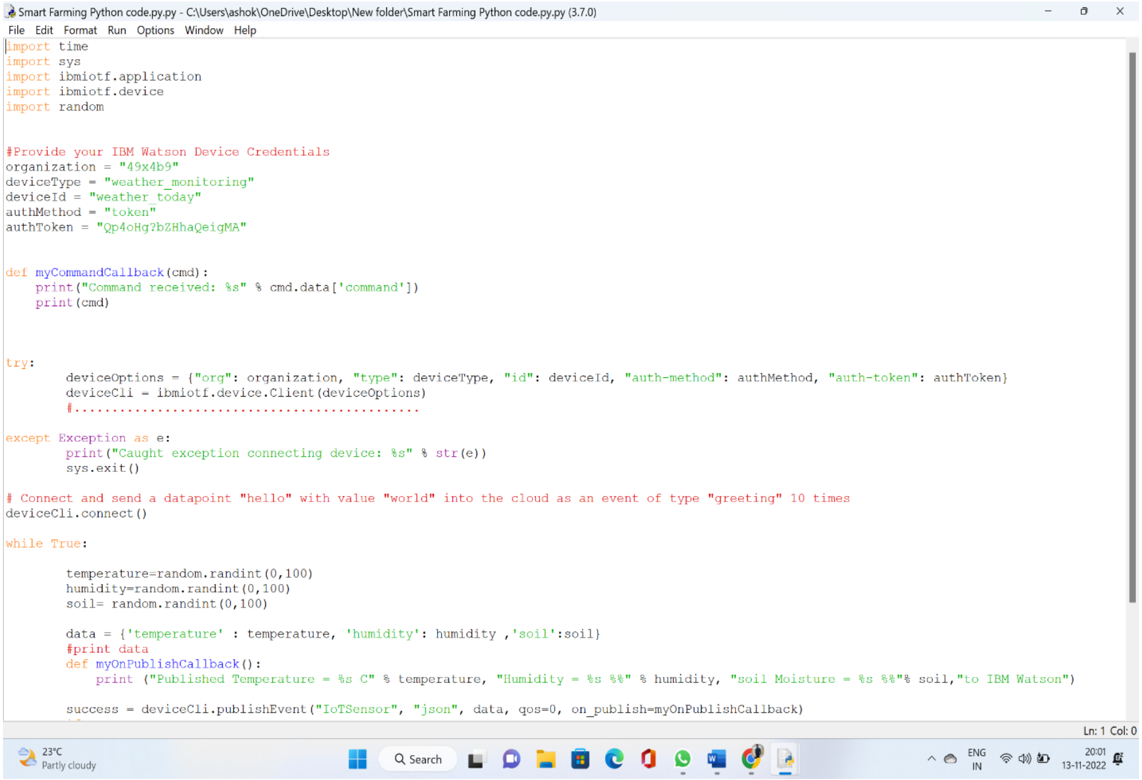
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed(as on Planned End Date)	Sprint Release Date(Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	28 Oct 2022
Sprint-2	20	5 Days	31 Oct 2022	04 Nov 2022	20	03 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	4 Days	14 Nov 2022	17 Nov 2022	20	16 Nov 2022

6.3 Report from JIRA



7. CODING & SOLUTIONING

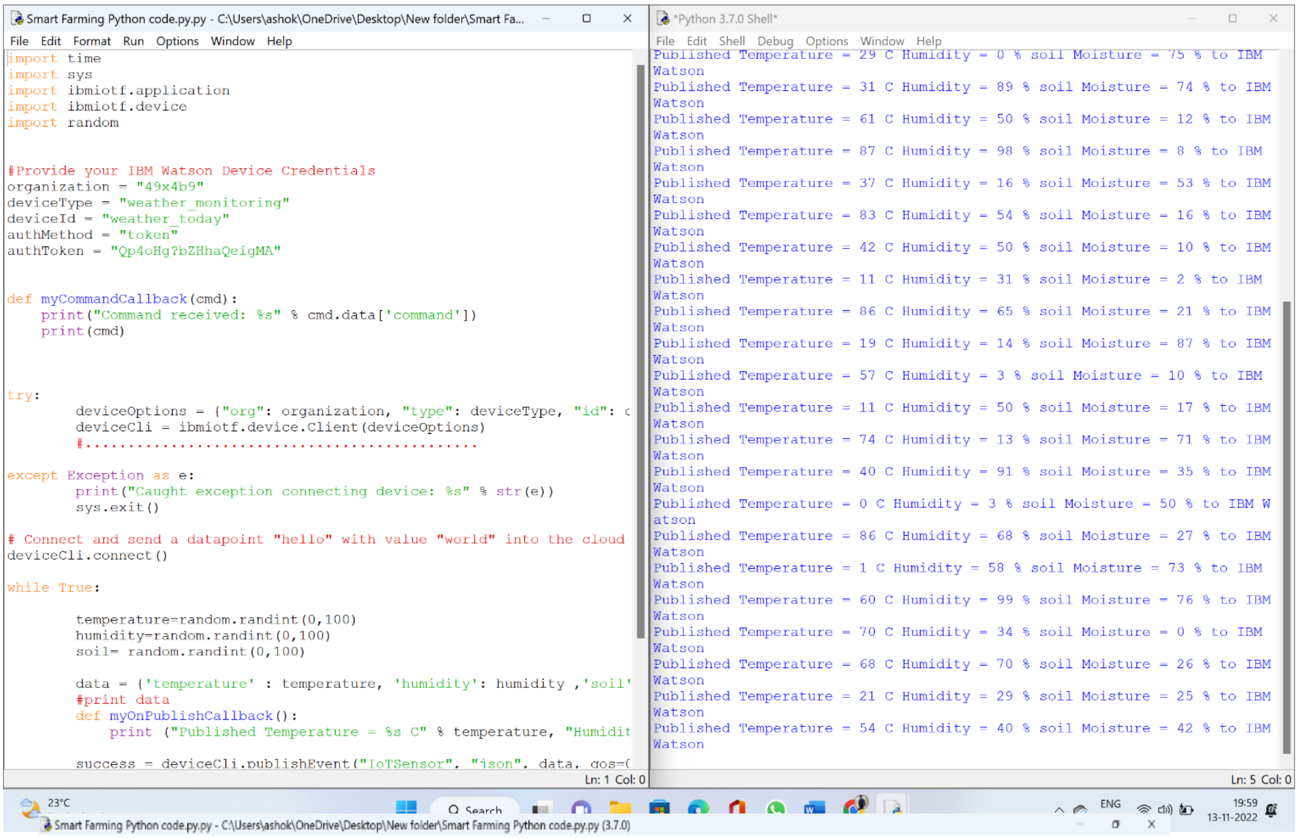
7.1 Feature 1

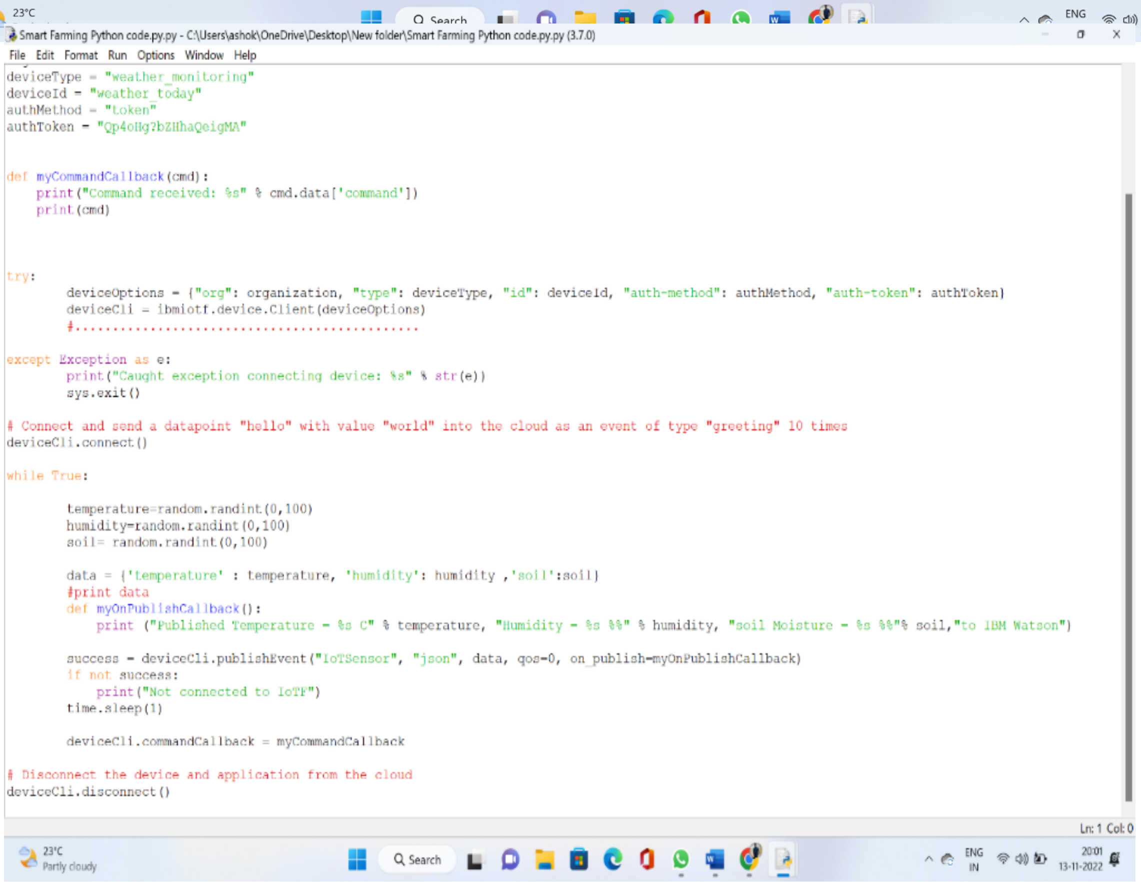

Smart Farming Python code.py.py - C:\Users\ashok\OneDrive\Desktop\New folder\Smart Farming Python code.py.py (3.7.0)

File Edit Format Run Options Window Help

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


#Provide your IBM Watson Device Credentials
organization = "49x4b9"
deviceType = "weather_monitoring"
deviceId = "weather_today"
authMethod = "token"
authToken = "Qp4oHg?bZHhaQeigMA"


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    print(cmd)



try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        #..............................................

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

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

while True:

        temperature=random.randint(0,100)
        humidity=random.randint(0,100)
        soil= random.randint(0,100)

        data = {'temperature' : temperature, 'humidity': humidity ,'soil':soil}
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temperature, "Humidity = %s %%" % humidity, "soil Moisture = %s %%"% soil,"to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
```

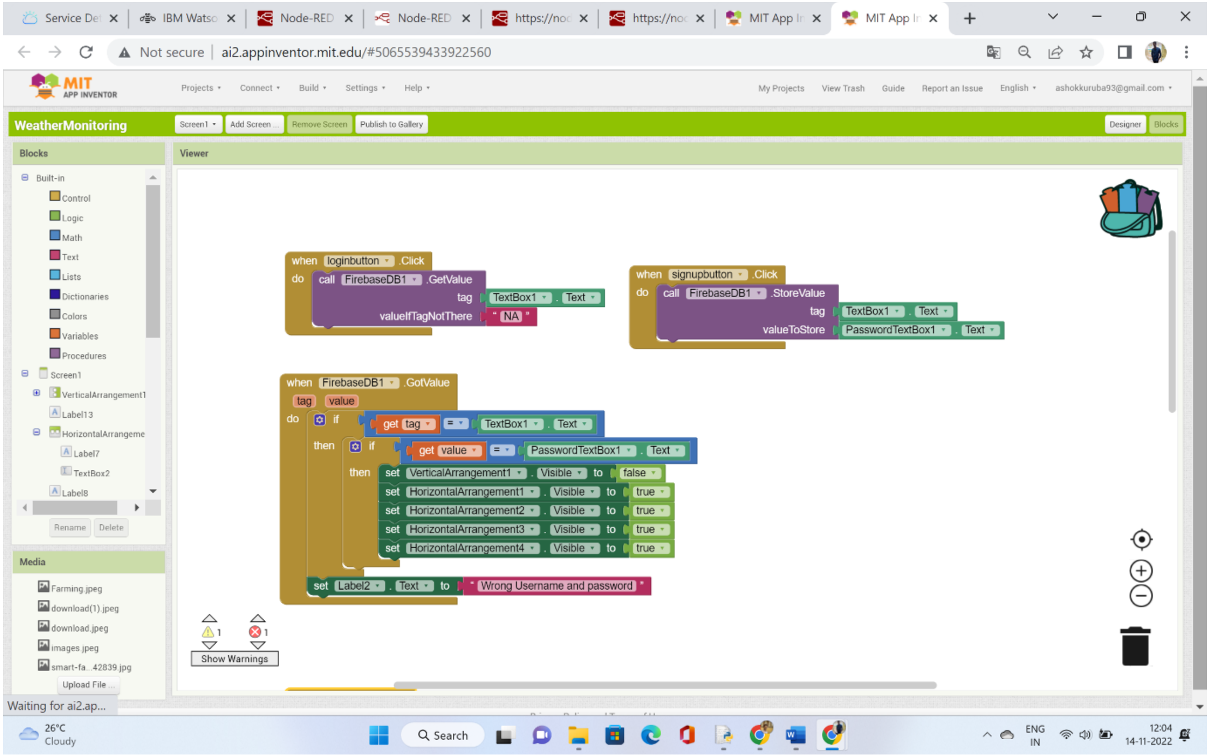
Ln: 1 Col: 0


Smart Farming Python code.py.py - C:\Users\ashok\OneDrive\Desktop\New folder\Smart Fa...
File Edit Format Run Options Window Help
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "49x4b9"
deviceType = "weather_monitoring"
deviceId = "weather_today"
authMethod = "token"
authToken = "Qp4oHg?bZHhaQeigMA"
def myCommandCallback(cmd):
print("Command received: %s" % cmd.data['command'])
print(cmd)
try:
deviceOptions = {"org": organization, "type": deviceType, "id": c
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
deviceCli.connect()
while True:
temperature=random.randint(0,100)
humidity=random.randint(0,100)
soil= random.randint(0,100)
data = {'temperature' : temperature, 'humidity': humidity ,'soil'
#print data
def myOnPublishCallback():
print ("Published Temperature = %s C" % temperature, "Humidit
success = deviceCli.publishEvent("IoTSensor", "json", data, qos=(
Ln: 1 Col: 0
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Published Temperature = 29 C Humidity = 0 % soil Moisture = 75 % to IBM Watson
Published Temperature = 31 C Humidity = 89 % soil Moisture = 74 % to IBM Watson
Published Temperature = 61 C Humidity = 50 % soil Moisture = 12 % to IBM Watson
Published Temperature = 87 C Humidity = 98 % soil Moisture = 8 % to IBM Watson
Published Temperature = 37 C Humidity = 16 % soil Moisture = 53 % to IBM Watson
Published Temperature = 83 C Humidity = 54 % soil Moisture = 16 % to IBM Watson
Published Temperature = 42 C Humidity = 50 % soil Moisture = 10 % to IBM Watson
Published Temperature = 11 C Humidity = 31 % soil Moisture = 2 % to IBM Watson
Published Temperature = 86 C Humidity = 65 % soil Moisture = 21 % to IBM Watson
Published Temperature = 19 C Humidity = 14 % soil Moisture = 87 % to IBM Watson
Published Temperature = 57 C Humidity = 3 % soil Moisture = 10 % to IBM Watson
Published Temperature = 11 C Humidity = 50 % soil Moisture = 17 % to IBM Watson
Published Temperature = 74 C Humidity = 13 % soil Moisture = 71 % to IBM Watson
Published Temperature = 40 C Humidity = 91 % soil Moisture = 35 % to IBM Watson
Published Temperature = 0 C Humidity = 3 % soil Moisture = 50 % to IBM Watson
Published Temperature = 86 C Humidity = 68 % soil Moisture = 27 % to IBM Watson
Published Temperature = 1 C Humidity = 58 % soil Moisture = 73 % to IBM Watson
Published Temperature = 60 C Humidity = 99 % soil Moisture = 76 % to IBM Watson
Published Temperature = 70 C Humidity = 34 % soil Moisture = 0 % to IBM Watson
Published Temperature = 68 C Humidity = 70 % soil Moisture = 26 % to IBM Watson
Published Temperature = 21 C Humidity = 29 % soil Moisture = 25 % to IBM Watson
Published Temperature = 54 C Humidity = 40 % soil Moisture = 42 % to IBM Watson
Ln: 5 Col: 0
23°C
Search
ENG
19:59
13-11-2022

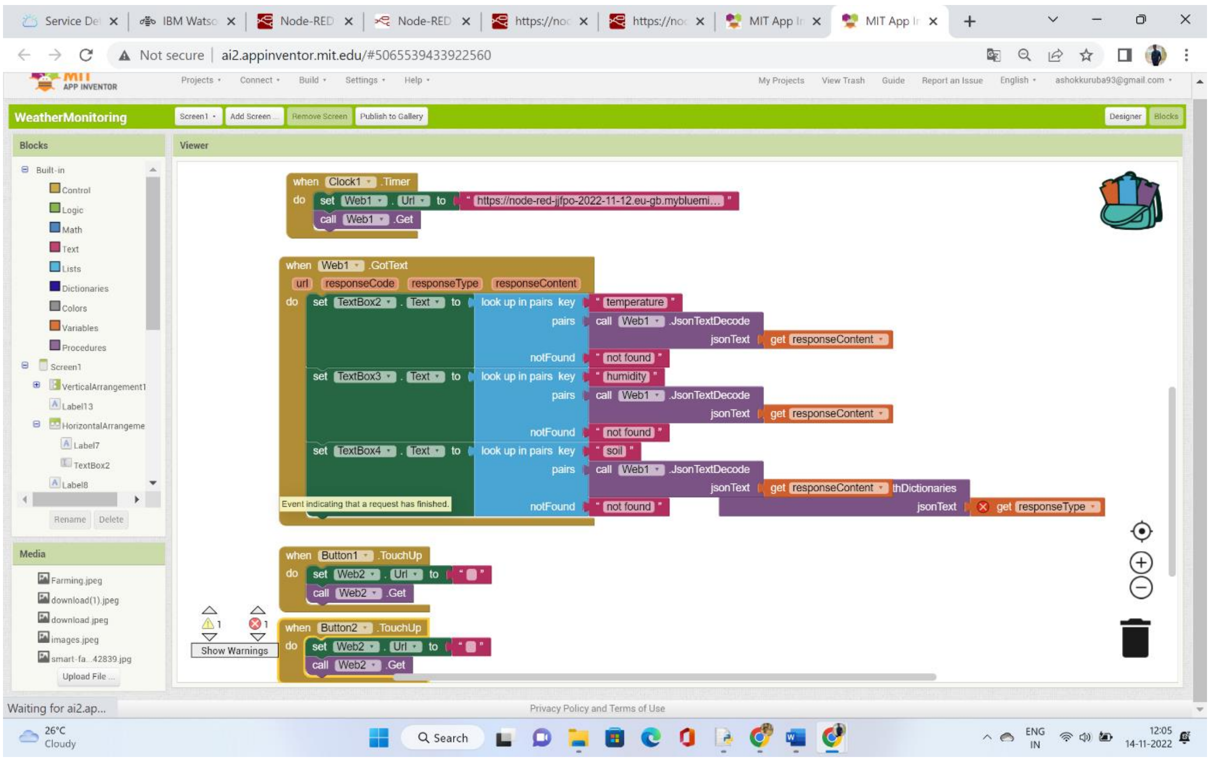

Smart Farming Python code.py.py - C:\Users\ashok\OneDrive\Desktop\New folder\Smart Farming Python code.py.py (3.7.0)
File Edit Format Run Options Window Help
deviceType = "weather_monitoring"
deviceId = "weather_today"
authMethod = "token"
authToken = "Qp4oHg?bZHhaQeigMA"
def myCommandCallback(cmd):
print("Command received: %s" % cmd.data['command'])
print(cmd)
try:
deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
deviceCli = ibmiotf.device.Client(deviceOptions)
#..
except Exception as e:
print("Caught exception connecting device: %s" % str(e))
sys.exit()
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()
while True:
temperature=random.randint(0,100)
humidity=random.randint(0,100)
soil= random.randint(0,100)
data = {'temperature' : temperature, 'humidity': humidity ,'soil':soil}
#print data
def myOnPublishCallback():
print ("Published Temperature = %s C" % temperature, "Humidity = %s %%" % humidity, "soil Moisture = %s %%"% soil,"to IBM Watson")
success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
if not success:
print("Not connected to IoTF")
time.sleep(1)
deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
Ln: 1 Col: 0
23°C
Partly cloudy
Search
ENG
IN
20:01
13-11-2022

7.2 Feature 2

These are the blocks of the login and signup page of mobile application.



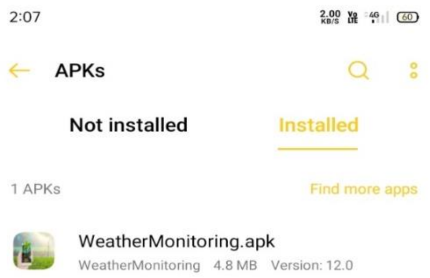
These are the blocks in the second page of the mobile application.



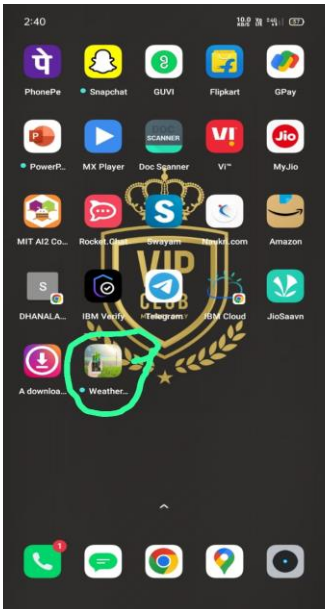
8.TESTING

8.1 Test Cases

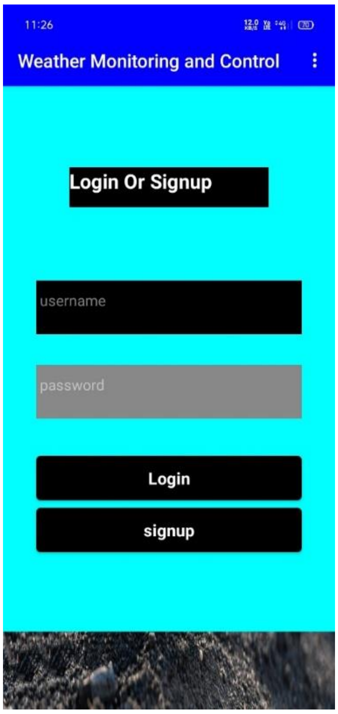
Step-1: First user need to download the android APK file from MIT app inventor where we developed our mobile application and install in their mobiles.



Step-2: After successful installation we can find app icon in our mobile as shown below.

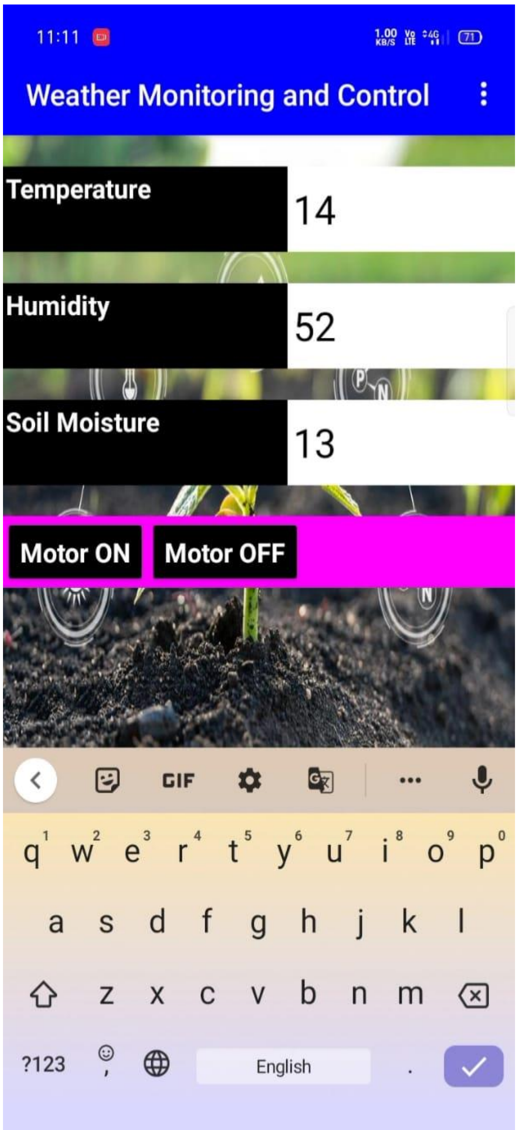


Step-3: After clicking the app icon it ask the user need to create username and password.so give username and password and click the signup button. The user can see interface like these as shown below.



8.2 User Acceptance Testing

After successful login. The next page will be open. In that page we can see the real time temperature , humidity and soil moisture reading and motor ON and motor OFF control button also as shown below.

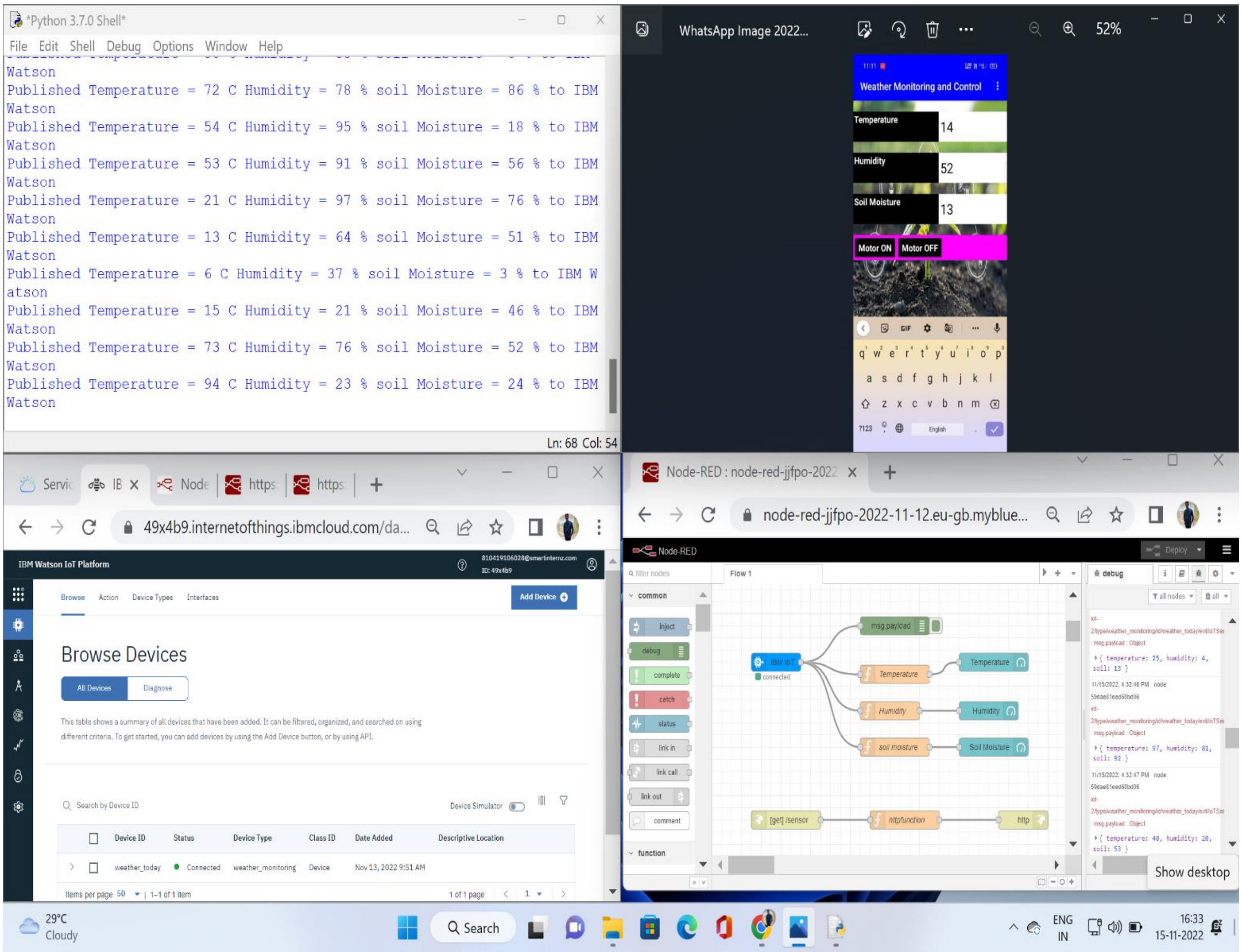


we are successfully created the IOT enabled smart farming application.

9. RESULTS

9.1 Performance Metrics

So finally when we run the python code it is going to connect the IBM Watson platform and connecting to the node-red after that is going to connect the mobile application.so we can see output in the fourth window.



10.ADVANTAGES & DISADVANTAGES

ADVANTAGES

All the data like climatic conditions and changes in them, soil or crop conditions everything can be easily monitored.

Risk of crop damage can be lowered to a greater extent.

Many difficult challenges can be avoided making the process automated and the quality of crops can be maintained.

The process included in farming can be controlled using the web applications from anywhere, anytime.

DISADVANTAGES:

Smart Agriculture requires internet connectivity continuously, but rural parts cannot fulfill this requirement.

Any faults in the sensors can cause great loss in the agriculture, due to wrong records and the actions of automated processes.

IoT devices need much money to implement.

11. CONCLUSION: So finally we build A IoT Web Application for smart agricultural system using Watson IoT platform, Watson simulator, IBM cloud and Node-RED and MIT app Inventor

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

13.APPENDIX

Source Code:

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


#Provide your IBM Watson Device Credentials
organization = "49x4b9"
deviceType = "weather_monitoring"
deviceId = "weather_today"
authMethod = "token"
authToken = "Qp4oHg?bZHhaQeigMA"


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    print(cmd)



try:
    deviceOptions = {"org": organization, "type":
deviceType, "id": deviceId, "auth-method": authMethod,
"auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #..............................................

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

# Connect and send a datapoint "hello" with value "world"
into the cloud as an event of type "greeting" 10 times
```

```
deviceCli.connect()

while True:

        temperature=random.randint(0,100)
        humidity=random.randint(0,100)
        soil= random.randint(0,100)

        data = {'temperature' : temperature, 'humidity': humidity ,'soil':soil}
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temperature, "Humidity = %s %%" % humidity, "soil Moisture = %s %%"% soil,"to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(1)

        deviceCli.commandCallback = myCommandCallback

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

GitHub:

Name	GitHub (User Name)
Team Leader(Kuruba Ashok)	Kuruba93
Team Member(M. Thirupathi Reddy)	Thirupathi70
Team Member(B. Harsha Kumar)	harshak1011
Team Member(M. Gireesh Nag)	gireesh1718

GitHub Link:

https://github.com/IBM-EPBL/IBM-Project-13788-1659530472.git

Project Demonstration Video Link:

https://drive.google.com/file/d/1UTgYIHR_PCnG7B5pMIROwCMwKqMcDafm/view?usp=share_link