

SMART FARMER-IOT ENABLED SMART FARMING APPLICATION

PROJECT REPORT

Submitted by

TINI J MERCY (211419205173)

BRINDHA S (211419205028)

SNEHA N (211419205158)

SAMYUKTHAA V.G (211419205141)

**in partial fulfilment for the award of the degree
of**

**BACHELOR OF
TECHNOLOGY IN**

INFORMATION TECHNOLOGY

PANIMALAR ENGINEERING COLLEGE(AUTONOMOUS)

PROJECT REPORT

1. INTRODUCTION.....	01
Project Overview	
Purpose	
2. LITERATURE SURVEY	02
Existing problem	
References	
Problem Statement Definition	
3. IDEATION & PROPOSED SOLUTION.....	03
Empathy Map Canvas	
Ideation & Brainstorming	
Proposed Solution	
Problem Solution fit	
4. REQUIREMENT ANALYSIS.....	09
Functional requirement	
Non-Functional requirements	
5. PROJECT DESIGN	11
Data Flow Diagrams	
Solution & Technical Architecture	
User Stories	
6. PROJECT PLANNING & SCHEDULING	16
Sprint Planning & Estimation	
Sprint Delivery Schedule	
Reports from JIRA	
7. CODING & SOLUTIONING	18
Feature 1	
Feature 2	
Database Schema (if Applicable)	

8. TESTING.....	21
Test Cases	
User Acceptance Testing	
9. RESULTS	24
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

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 Farming System

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 Moisture, 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

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.

References

- [1] Tharindu Madushan Bandara, Hinayana Mudiyansele, Mansoor Raza. "Smart farm and monitoring system for measuring the environment using wireless sensor network-IoT technology in farming" 2020.
- [2] Siwaporn Meadthaisong, Thiang Meathaisong "Smart Farming Using Internet of Thing(IoT) in Agriculture by Tangible Programming for Children"2020 .
- [3] Anushree Math, Layak Ali, Pruthviraj U "Development of Smart Drip Irrigation System Using IoT"2018.
- [4] Sen g-Kyoun Jo Dae-Haon Pary Hyeon Park Se-Han Kim" Smart Livestock Farms Using Digital Twin:Feasibility",2018.
- [5] Zahid Khan, Muhammad Zahid Khan, Irshad Ahmed Abbasi." Internet of things based smart farming monitoring system for blotting reduction in onion farms". 2021
- [6] Xiaofan Jiang, Jose Fernando Waimin, Hongjie Jiang, Charilaos Mousoulis, Nithin Raghunathan, Ra him Brahmins and Diminutions peroulis" Wireless Sensor Network Utilizing Flexible Nitrate Sensors for Smart Farming".2021
- [7] Anusha Vangala, Anil Kumar Sutrala, Ashok Kumar Das, Minho Jo "Smart Contract-Based Blockchain-Envisioned Authentication Scheme for smart Farming " 2021.
- [8] Md Masum Billah, Zulkhari Mohd Yusof, Kushsairy Kadir, Abdul Malik Mohd Ali, Izanoordina Ahmad"Real-time Monitoring of water Quality in animal Farm :An IoT Application "2019.
- [9] Sai sree laya appalling, Sid-up mittal, Maanak Gupta "Gerontologist and Artificial Intelligence Systemsfor the cooperative Smart Farming Ecosystem",2020

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 faile

3.IDEATION & PROPOSED SOLUTION

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




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:



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 🕒 10 minutes to prepare
- 🕒 1 hour to collaborate
- 👤 2-8 people recommended

➔

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

- A Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- B Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.
- C Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article ➔

1


Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕒 5 minutes

PROBLEM

How might we [your problem statement]?



Key rules of brainstorming

To run an smooth and productive session

- 🗣️ Stay in topic.
- 💡 Encourage wild ideas.
- ⏸️ Defer judgment.
- 👂 Listen to others.
- 🗣️ Go for volume.
- 👁️ If possible, be visual.

Step-2: Brainstorm ,Idea Listing and Grouping

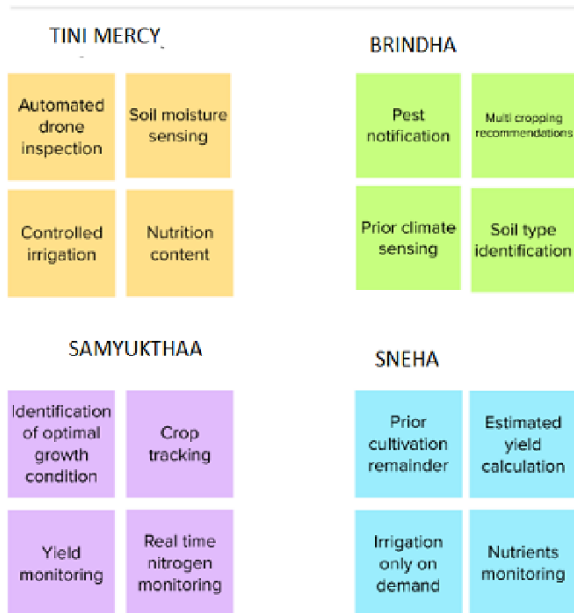
Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes



3

Group Ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes



Step-3: Idea Prioritization

Step-3: Idea Prioritization

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

🕒 30 minutes



Proposed Solution:

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To save natural resources such as water Simultaneously increase the
2.	Idea / Solution description	By controlling and monitoring the farm using sensors which help them in forecasting the yields, improving efficiency in farming methods..
3.	Novelty / Uniqueness	To implement the soil moisture measuring system to measure the moisture level in the farm. With the help of temperature monitoring, it can maintain and control the farm. Can control the motor pumps by setting time using the mobile application.
4.	Social Impact / Customer Satisfaction	By controlling and monitoring the farm using the mobile application can save farmer's time and money, at the same time can provide more yield efficiently.
5.	Business Model (RevenueModel)	Water and money will be saved. Increased work efficiency. Reduce consumables. Increase yields. Easy of recording and reporting. Easy of use.
6.	Scalability of the Solution	Automated sensing of farm.Improved accuracy rate. Reduced time and work burden.

Problem Solution fit

Project Title: Smart Farmer-IoT Enabled Smart Farmer Application

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMID01408

Define CS, fit into	1. CUSTOMER SEGMENT(S) Farmer is our customer	6. CUSTOMER CONSTRAINTS The main limitation to our customer is the lack of connectivity in rural areas and	5. AVAILABLE SOLUTIONS Right Now, Irrigation is managed remotely. We are trying to track Greenhouse Gases, Predicting crops requirement before.	Explore AS, fit into
	2. JOBS-TO-BE-DONE / PROBLEMS It helps in monitoring crops remotely and to take control of motor and other sensor-based devices	9. PROBLEM ROOT CAUSE Agriculture prediction is hard because to predict we must consider lot of parameters and some parameters are not sensed by sensors	7. BEHAVIOUR Use of Proper drainage system to overcome the effects of excess water from heavy rain. Use of hybrid plants that are resistant to pests	
Identify success factors	3. TRIGGERS Explaining the use and need of this product in the government agricultural training centers	10. YOUR SOLUTION Using cellular network for connectivity can solve the connectivity issues, Using Edge-computing can unlock computing problems, making it cost efficient can attract more customers.	8. CHANNELS of BEHAVIOUR ONLINE Youtube as the online communication channel OFFLINE Newspaper Advertisement as offline communication medium.	Predict if you can reach
	4. EMOTIONS BEFORE / AFTER Affrighted about weather, Fear of diseases affecting plants and pesticides > Now they are in 24/7 monitoring remotely and they also get alerted frequently.			

3. REQUIREMENT ANALYSIS

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 Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	System login	Check authorization Check access
FR-4	Manage schedule	Manage system admins Manage user consent Manage user
FR-5	Check details	Moist details Temperature details
FR-6	Log out	Exit

Non-Functional Requirements:

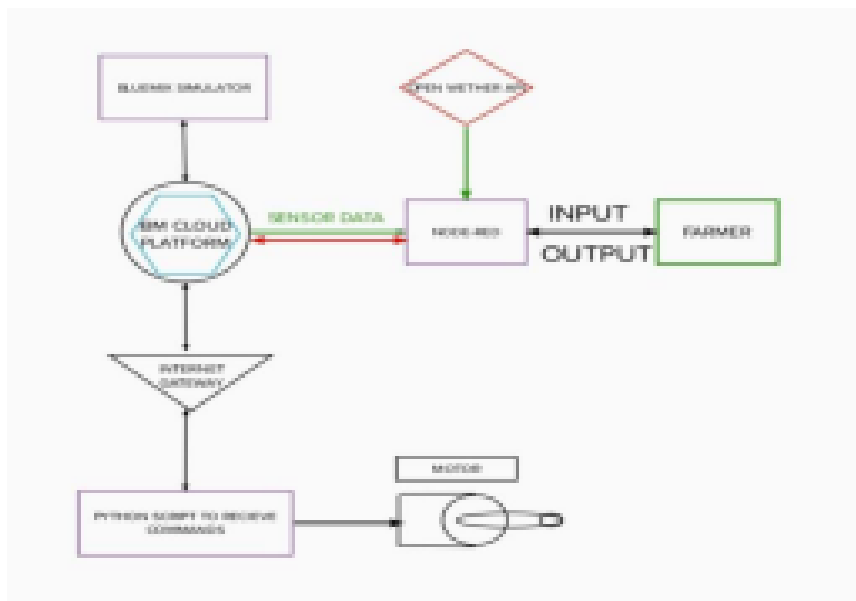
Following are the non-functional requirements of the proposed solution.

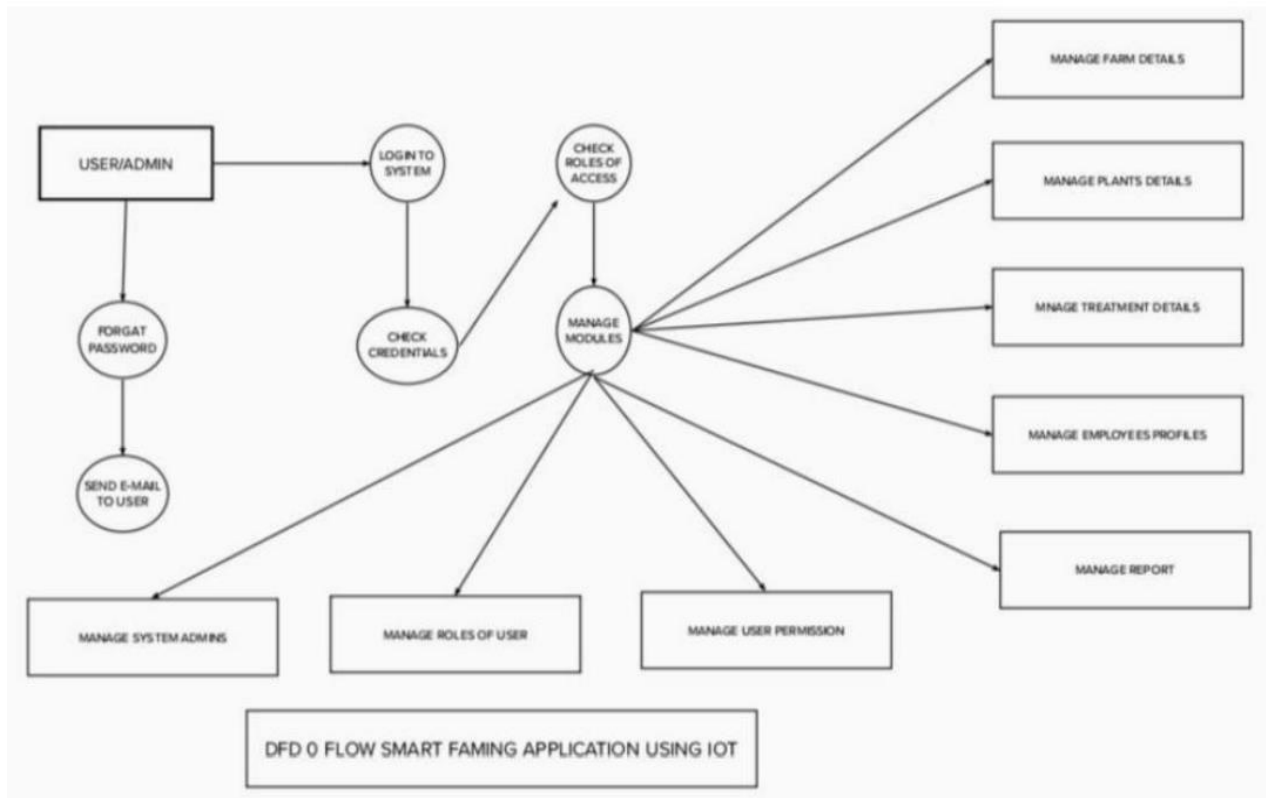
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Usability specify the quality attributes of system. This requirement can be the speed with which a system must perform to satisfy user expectations
NFR-2	Security	Sensitive and individual data must be secured by their proffering until the decision-making storing stages
NFR-3	Reliability	This focused on the promise dataset is used. The model uses diligence and shared protection to neglect farm service
NFR-4	Performance	It requires low power consumption and low data transmission rates. This idea of implementation combined sensors with soil and environmental parameters.
NFR-5	Availability	These quality characteristics considered are cost, sensitivity, design complexity, storage capacity, development process, response criteria and environmental impact and farming equipment made possible like crops, weather , humidity, etc
NFR-6	Scalability	It is the major concern for IoT platform. It has different choices of IOT platform affect system and real time accountability in an environment.

4.PROJECT DESIGN

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, whether to water the crop or not depending upon the sensor values. By using the app they can remotely operate to the motor switch.

5. User Stories

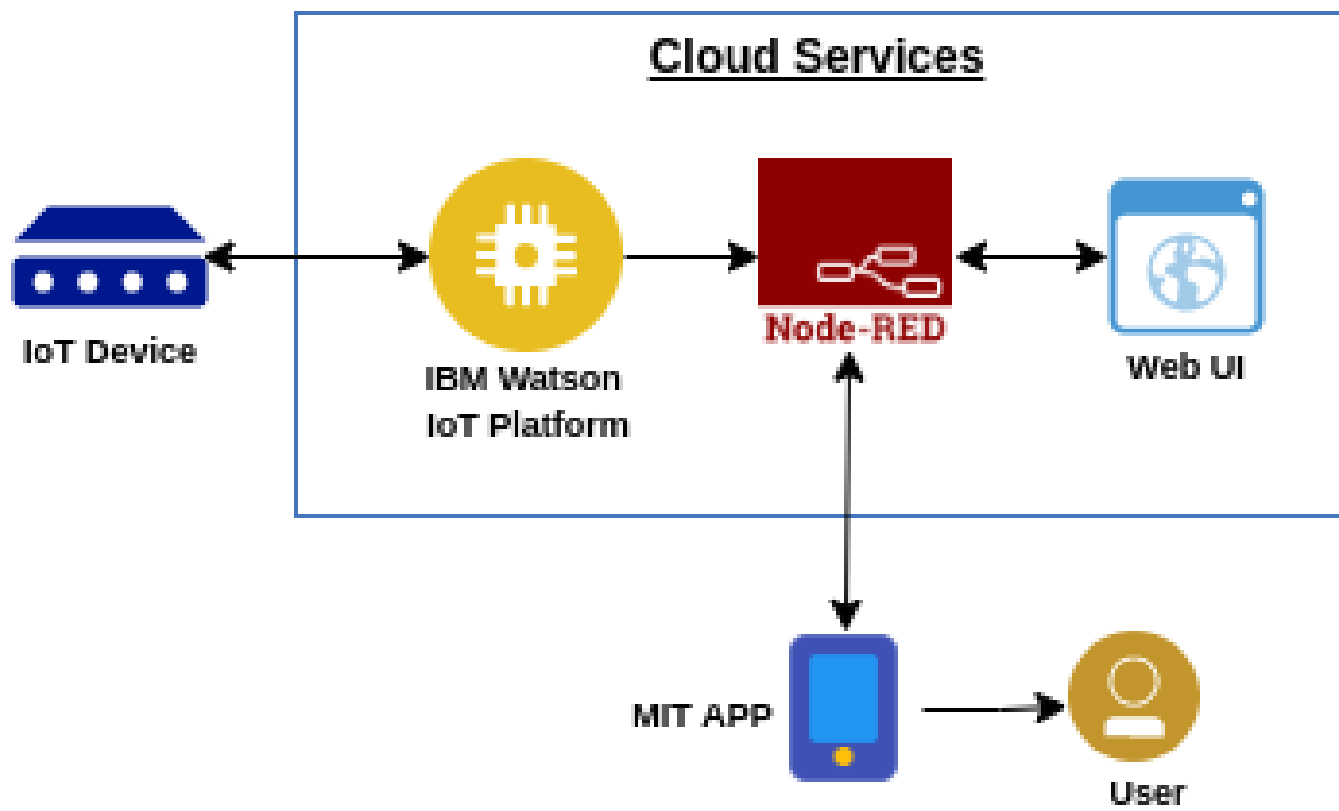
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web user)						
Customer Care Executive						
Administrator						

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 from sensors 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:



User Stories

Sprint	Functional Requirement(Epic)	User Story Number	User Story /Task	Story Points	Priority
Sprint-1	Simulationcreation	USN-1	Connect Sensors and Arduino with pythoncode	2	High
Sprint-2	Software	USN-2	Creating device in the IBMWatson IoT platform,workflow for IoT scenario susing Node-Red	2	High
Sprint-3	MITAppInventor	USN-3	Develop an application for the Smart farmer project using MITApp Inventor	2	High
Sprint-3	Dashboard	USN-3	Design the Modules and test the app	2	High
Sprint-4	WebUI	USN-4	To make the user to interact with software	2	High

ProjectTracker, Velocity&BurndownChart:(4Marks)

Sprint	Total StoryPoints	Duration	SprintStartDate	SprintEndDate(Planned)	Story PointsCompleted on PlannedEnd
Sprint-1	20	6Days	24Oct2022	29Oct2022	20
Sprint-2	20	6Days	31Oct2022	05Nov2022	20
Sprint-3	20	6Days	07Nov2022	12Nov2022	20
Sprint-4	20	6Days	14Nov2022	19Nov2022	20

Velocity:

Imagine we have a 10-days 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)

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

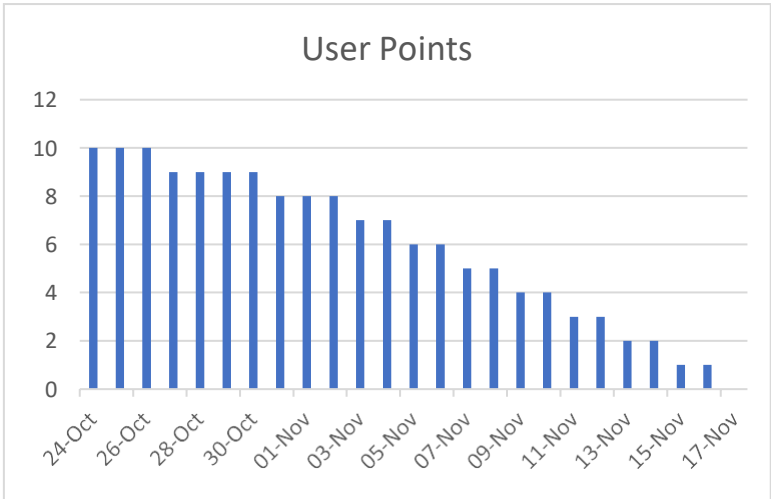
6.PROJECT PLANNING & SCHEDULING

Sprint planning & Estimation

S.NO	ACTIVITY TITLE	ACTIVITY DESCRIPTION	DURATION
1	Understanding the project	Assign the team members after that create repository in the GitHub and then assign task to each member and guide them how to access the GitHub while submitting the assignments	1 week
2	Staring The Project	Team Members to Assign All the Tasks Based on Sprints and Work on It Accordingly.	1 week
3	Completing Every Task	Team Leader should ensure that whether every team member have completed the assigned task or not	1 week
4	Stand Up Meetings	Team Lead Must Have a Stand-Up Meeting with The Team and Work on The Updates and Requirement Session	1 week

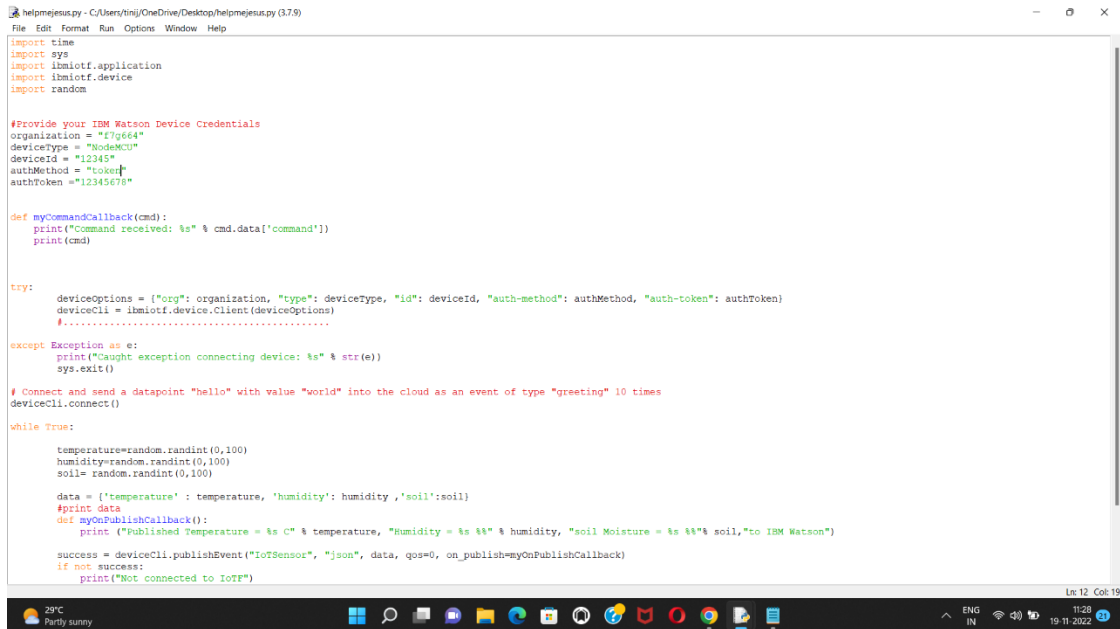
5	Deadline	Ensure that team members are completing every task within the deadline	1 week
6	Budget and Scope of project	Analyze the overall budget which must be within certain limit it should be favorable to every person	1 week

Report from JIRA



7. CODING & SOLUTIONING

Feature 1



```
helpmejenus.py - C:/Users/tmj/OneDrive/Desktop/helpmejenus.py (3.7.9)
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 = "f7g664"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"

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 IoT")
```

Ln: 12 Col: 19

29°C Partly sunny

ENG IN 11:28 19-11-2022


```
helpmejesus.py - C:/Users/tinij/OneDrive/Desktop/helpmejesus.py (3.7.9)
File Edit Format Run Options Window Help

#Provide your IBM Watson Device Credentials
organization = "f7g664"
deviceType = "NodeMCU"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"

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 IoT")
    time.sleep(1)

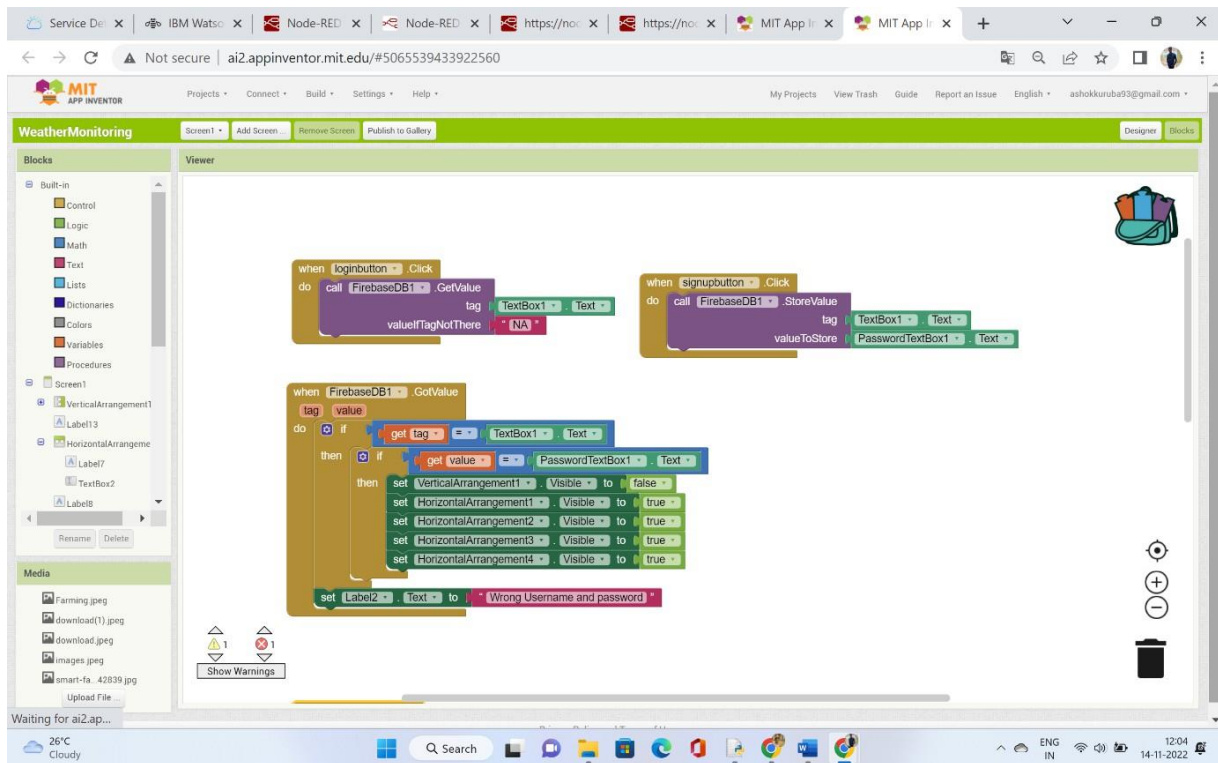
    deviceCli.commandCallback = myCommandCallback

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

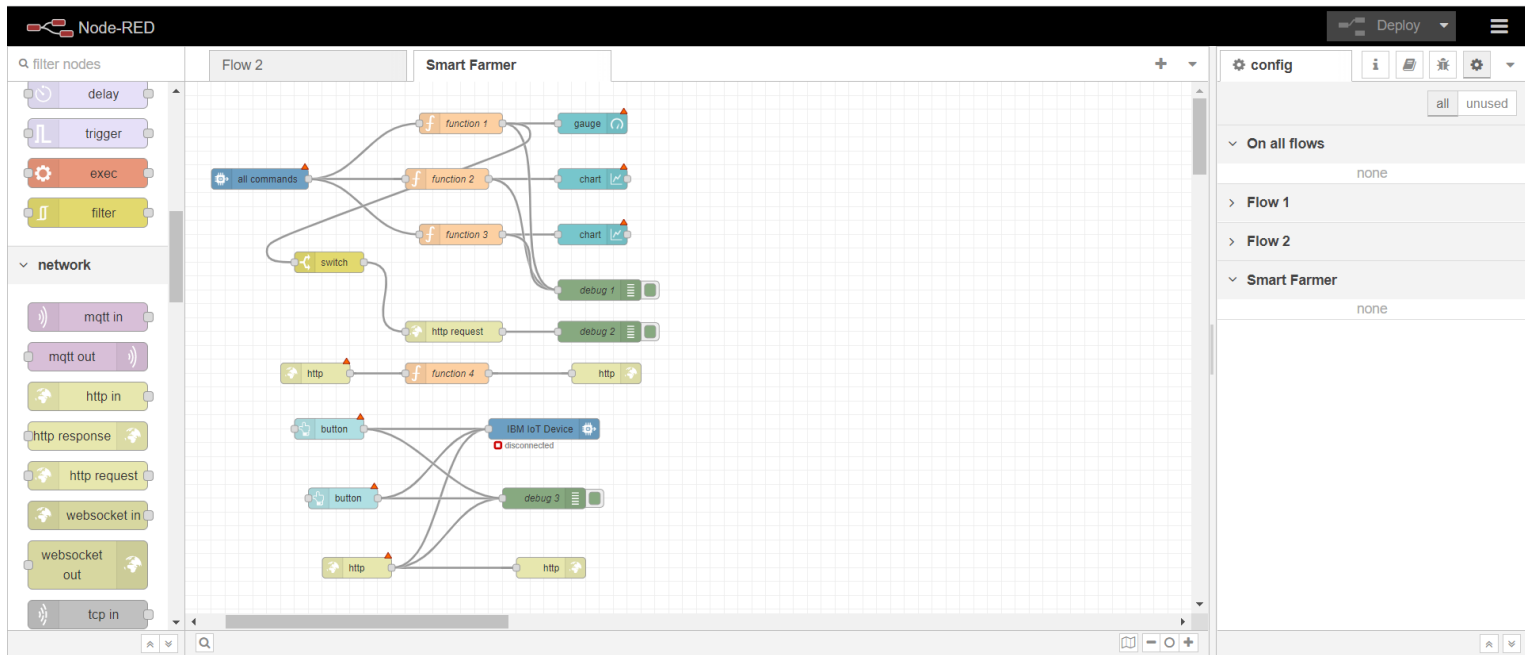
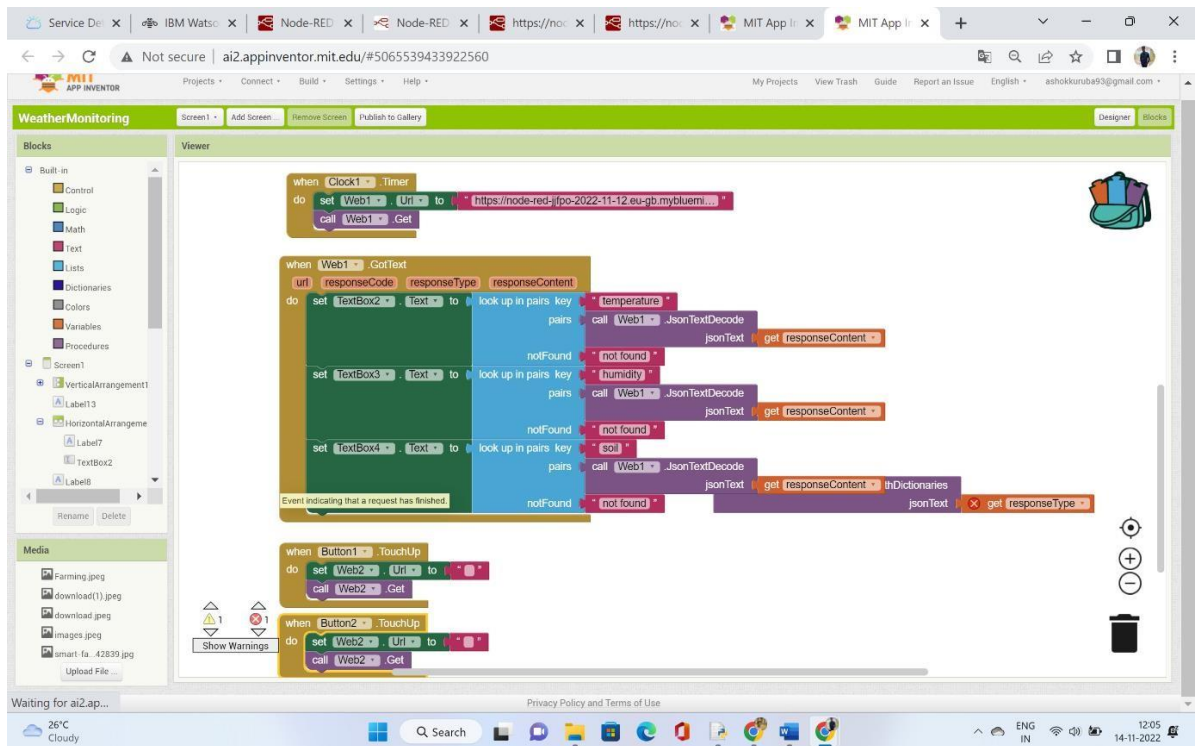
```
*Python 3.7.9 Shell*
File Edit Shell Debug Options Window Help
===== RESTART: C:/Users/tinij/OneDrive/Desktop/helpmejesus.py =====
2022-11-19 11:23:02,125 ibmiotf.device.Client INFO Connected successfully: d:f7g664:NodeMCU:12345
Published Temperature = 98 C Humidity = 10 % soil Moisture = 29 % to IBM Watson
Published Temperature = 91 C Humidity = 30 % soil Moisture = 99 % to IBM Watson
Published Temperature = 63 C Humidity = 64 % soil Moisture = 14 % to IBM Watson
Published Temperature = 74 C Humidity = 66 % soil Moisture = 89 % to IBM Watson
Published Temperature = 14 C Humidity = 39 % soil Moisture = 74 % to IBM Watson
Published Temperature = 55 C Humidity = 28 % soil Moisture = 29 % to IBM Watson
Published Temperature = 87 C Humidity = 89 % soil Moisture = 91 % to IBM Watson
Published Temperature = 25 C Humidity = 76 % soil Moisture = 70 % to IBM Watson
Published Temperature = 11 C Humidity = 15 % soil Moisture = 90 % to IBM Watson
Published Temperature = 26 C Humidity = 74 % soil Moisture = 1 % to IBM Watson
Published Temperature = 59 C Humidity = 35 % soil Moisture = 25 % to IBM Watson
Published Temperature = 62 C Humidity = 56 % soil Moisture = 10 % to IBM Watson
Published Temperature = 98 C Humidity = 13 % soil Moisture = 30 % to IBM Watson
Published Temperature = 40 C Humidity = 9 % soil Moisture = 48 % to IBM Watson
Published Temperature = 5 C Humidity = 59 % soil Moisture = 51 % to IBM Watson
Published Temperature = 100 C Humidity = 12 % soil Moisture = 4 % to IBM Watson
Published Temperature = 13 C Humidity = 94 % soil Moisture = 29 % to IBM Watson
Published Temperature = 7 C Humidity = 38 % soil Moisture = 21 % to IBM Watson
Published Temperature = 4 C Humidity = 75 % soil Moisture = 39 % to IBM Watson
Published Temperature = 7 C Humidity = 16 % soil Moisture = 17 % to IBM Watson
Published Temperature = 55 C Humidity = 49 % soil Moisture = 28 % to IBM Watson
Published Temperature = 21 C Humidity = 31 % soil Moisture = 32 % to IBM Watson
Published Temperature = 21 C Humidity = 83 % soil Moisture = 82 % to IBM Watson
Published Temperature = 98 C Humidity = 13 % soil Moisture = 8 % to IBM Watson
Published Temperature = 22 C Humidity = 64 % soil Moisture = 24 % to IBM Watson
Published Temperature = 11 C Humidity = 86 % soil Moisture = 33 % to IBM Watson
Published Temperature = 91 C Humidity = 65 % soil Moisture = 18 % to IBM Watson
Published Temperature = 87 C Humidity = 51 % soil Moisture = 40 % to IBM Watson
Published Temperature = 21 C Humidity = 48 % soil Moisture = 29 % to IBM Watson
Published Temperature = 83 C Humidity = 22 % soil Moisture = 96 % to IBM Watson
Published Temperature = 97 C Humidity = 62 % soil Moisture = 35 % to IBM Watson
Published Temperature = 67 C Humidity = 46 % soil Moisture = 43 % to IBM Watson
Published Temperature = 13 C Humidity = 69 % soil Moisture = 65 % to IBM Watson
Published Temperature = 88 C Humidity = 39 % soil Moisture = 90 % to IBM Watson
Published Temperature = 46 C Humidity = 10 % soil Moisture = 14 % to IBM Watson
Published Temperature = 84 C Humidity = 63 % soil Moisture = 91 % to IBM Watson
Published Temperature = 69 C Humidity = 22 % soil Moisture = 20 % to IBM Watson
Published Temperature = 46 C Humidity = 80 % soil Moisture = 39 % to IBM Watson
Published Temperature = 40 C Humidity = 67 % soil Moisture = 0 % to IBM Watson
Published Temperature = 28 C Humidity = 91 % soil Moisture = 96 % to IBM Watson
Published Temperature = 89 C Humidity = 84 % soil Moisture = 72 % to IBM Watson
Published Temperature = 52 C Humidity = 56 % soil Moisture = 56 % to IBM Watson
Published Temperature = 38 C Humidity = 12 % soil Moisture = 23 % to IBM Watson
Published Temperature = 47 C Humidity = 6 % soil Moisture = 26 % to IBM Watson
Published Temperature = 67 C Humidity = 40 % soil Moisture = 0 % to IBM Watson
```

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

Test Cases

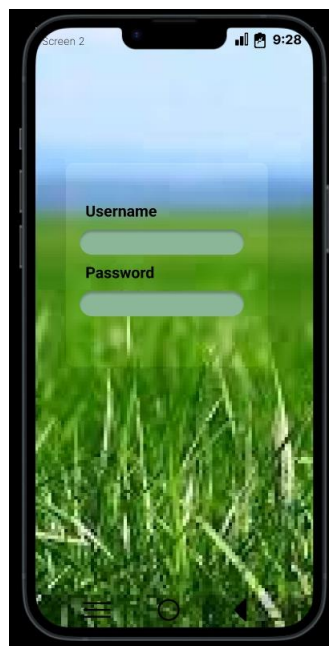
Step-1: First user need to download the android APK file from MIT app inventor where we developed our mobile application and install in their mobiles.



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

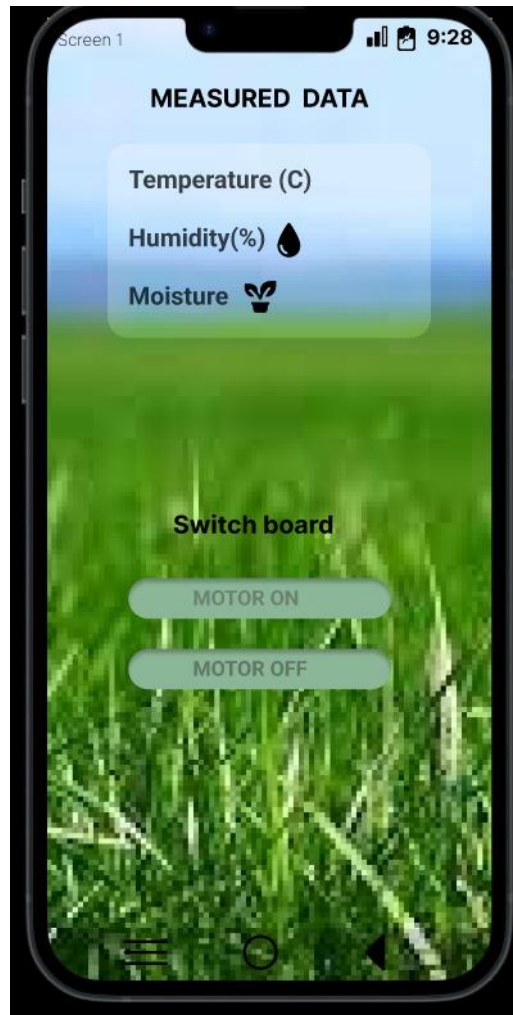


Step-3: After clicking the app icon it ask the user need to create username and password.so give username and password and click the signup button. The user can see interface like these as shown below.



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.

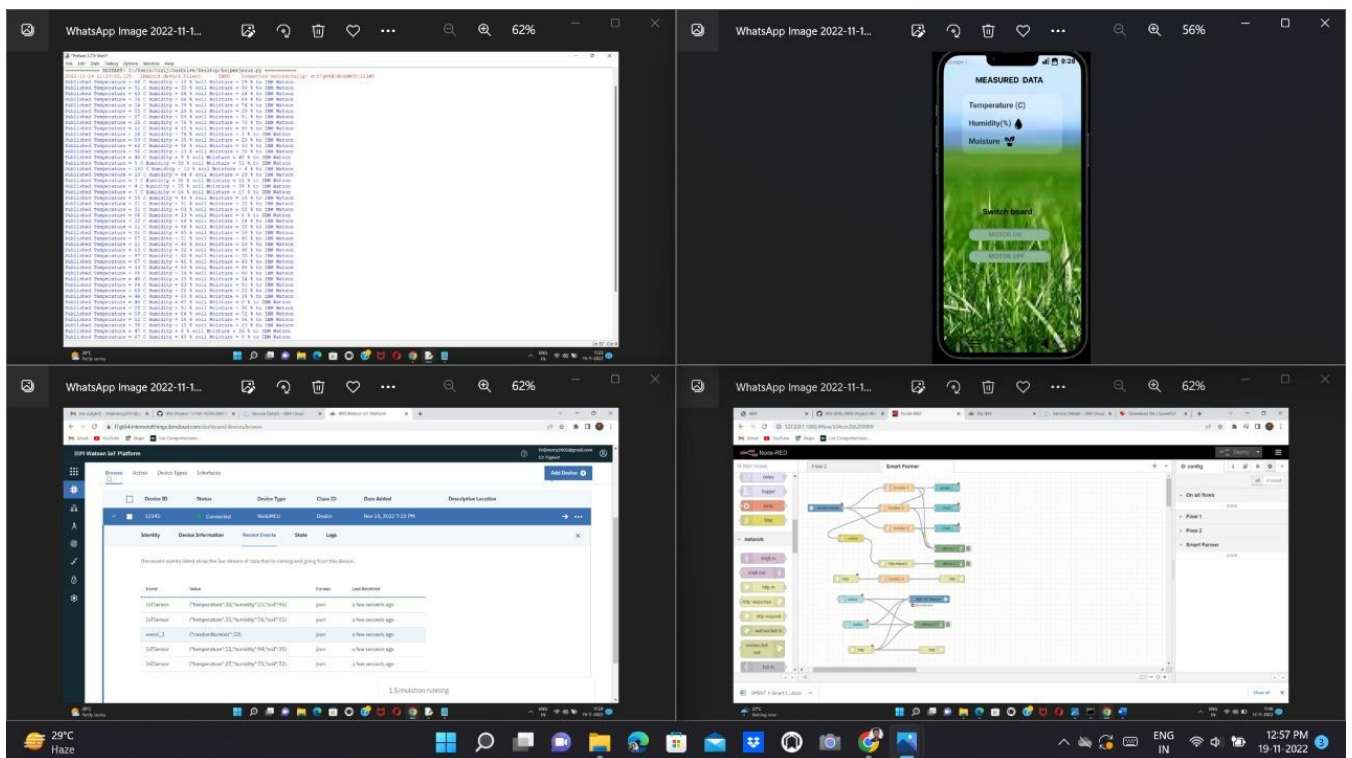


we are successfully created the IOT enabled smart farming application.

9. RESULTS

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 IoTTF")
        time.sleep(1)

    deviceCli.commandCallback = myCommandCallback

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

```

GitHub:

Name	GitHub (User Name)
Team Leader(Tini J Mercy)	Tini-j-Mercy
Team Member(Brindha S)	brindha00
Team Member(Sneha N)	sneha-1710
Team Member(Samyukthaa V.G)	Samyu-2001

GitHub Link:

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

Project Demonstration Video Link:

https://drive.google.com/file/d/189spH1IzN2MQ1UnKwQwwPsJP9JeMn3l_/view?usp=drivesdk