

# PROJECT REPORT

**Project Name** : Smartfarmer-IoT enabled smart  
Farming application

**Team ID** : PNT2022TMID06254

**Team Leader** : BHARATH A

**Team Members** : ABDUL AJEEZ S.B

**Team Members** : DATCHANA MOORTHY T

**Team Members** : KANNAN T

# SMART FARMER – IoT ENABLED SMART FARMING APPLICATION

## 1. INTRODUCTION

### 1.1 PROJECT OVERVIEW

IOT is a system that is built for monitoring the crop field with the help of sensors (light, humidity, temperature, soil moisture, crop health etc..) and automating the irrigation system. The farmers can monitor the field condition from anywhere. The IOT in farming where we used in this project is used to collect the data from various components like sensors. Then the user can change the system of behavior of supply water which will help to increase the production of farming.

### 1.2 PURPOSE

IoT-based agriculture system helps the farmer in monitoring different parameters of his field like soil moisture, Temperature, 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 the crop or postpone it by monitoring the sensor parameters and controlling the motor pumps from the mobile application itself. IoT based Smart Farming **improves the entire Agriculture system by monitoring the field in real-time**. With the help of sensors and interconnectivity, the Internet of Things in Agriculture has not only saved the time of the farmers but has also reduced the extravagant use of resources such as Water and Electricity.

## **2. LITERATURE SURVEY**

### **2.1 EXISTING PROBLEM**

Deploying a huge number of IoT devices for smart agriculture can cause interference to different network systems, especially some IoT networks using short spectrum bands such as ZigBee, Wi-Fi, Sigfox, and LoRa . Interference can degrade system performance as well as reduce the reliability of IoT ecosystems. IoT networks that use cognitive technology to reuse unlicensed spectra increase the cost of the device. In our opinion, the advent of the 6G network will allow a huge number of devices to connect to the Internet with an extremely high access speed and extremely large bandwidth. The full interference problem of IoT networks will be solved. Security and Privacy: One of the most important problems of applying IoT in smart agriculture is the security problem, including the protection of data and systems from attacks on the Internet. In regard to system security, IoT devices' limited capacity and ability led to complex encryption algorithms that are impossible to implement on IoT devices. As a result, IoT systems can be attacked using the Internet to gain system control rights; IoT gateways are also attacked via denial of service . In addition, cloud servers can be attacked by data spoofing to perform unauthorized tasks that affect the autonomous farming processes of farms. Cloud infrastructures can also be controlled by attackers . Several issues of detailed IoT data privacy and security measures have been discussed . According to Neshenko et al., the IoT data security issue is one of the biggest problems slowing down IoT adoption in smart agriculture . Regarding data security, the obtained information from IoT systems in farms is collected, processed, and commercially exploited by service providers to varying degrees. Therefore, one of the most important problems of policies regards the validity and legal status of farm data . In reality, these data are of great value when aggregated and analyzed for large-scale agricultural activities. Consequently, without policies, the data privacy and security of farms can affect the competitive advantage of farmers/farm owners. In our opinion, using cryptography coupled with access keys is a possible solution to solve this problem.

In our opinion, the security problems of IoT systems will be an exciting research topic and garner attention for both academia and industry research. An in-depth survey of threats and solutions to improve robustness, trust, and privacy for future IoT systems is presented .

## 2.2 REFERENCES

- [1] Pradayumna Gokuale, Omkar Bharth, Sagar babu , "Introduction to IoT and Devices", International Advanced Research Journal in Engineering Science and Technology (IJEST), Vol. 3, Issue 7, April 2017.
- [2] Drian Thomos, "The advanced technology in Internet Evolution: The IoT and its applications", Research journal in Technology of IoT.
- [3] P.John, P.Raj kumar "design and implementation new trends in agriculture" International Research of science and technology, Vol. 7, Issue 3, January 2016.
- [4] Muthunoori Suresh, K.kiran rathod," Smart Agriculture using IoT Technology", International Journal of science and Technology, ISSN: 972-1011, Volume-9 Issue-4, March 2018.
- [5] Nikesh Gondchawar, Prof. Dr. R. S. Kawitkar, "IOT based smart agriculture", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 6, June 2016.
- [6] Anand Nayyar, Er. Vikram Puri," Smart Farming: IoT Based Smart Sensors Agriculture Stick for Live Temperature and Moisture Monitoring using Arduino, Cloud Computing & Solar Technology", November 2017.
- [7] Sweksha Goyal, Unnathi Mundra, Prof. Sahana Shetty," SMART AGRICULTURE USING IOT", International Journal of Computer Science and Mobile Computing, Vol.8 Issue.5, pg. 143-148, May 2019.
- [8] K.Jagan mohan rao, P.suvarna Raju "New trends in agriculture farming", International journal of science and technology, ISSN:4545-4556, Volume-3, Issue-7,2015.
- [9] A.Anusha, A.Guptha, G.Sivanageswar Rao, Ravi Kumar Tenali, "A Model for Smart Agriculture Using IOT", International Journal of Innovative Technology and Exploring Engineering (IJITEE),ISSN:2278-3075, Volume-8 Issue-6, April 2019.

[10] J.Kiran Raj sekhar, P.Rajesh , "Advanced technology in agriculture farming", research in International journal, Volume 1, Issue 10,page no.151-156, March 2015.

[11] Akira Suyama and Ushio Inoue, "Novel Concept for a Long Lifetime Wireless Geo-fencing System with an Integrated Sub-10  $\mu$ A Wake-Up Receiver", ICIS, pp. 26-29, June 2016.

[12] Nik Hisham, Ahmad Rizan, Ibrahim and Ahmad Nizar, "IR 4.0 Using IoT and LORAWAN to Accelerate Lentinula Edodes Growth", ICSSA2018, 2018.

[13] L. Mainetti, L. Patrono, A. Secco and I. Sergi, "An IoT-aware AAL system for elderly people", 2016 Int. Multidiscip. Conf. Comput. Energy Sci. Split. 2016, 2016.

[14] F. J. Ferrández-Pastor, J. M. García-Chamizo, M. Nieto-Hidalgo and J. Mora-Martínez, "Precision Agriculture Design Method Using a Distributed Computing Architecture on Internet of Things Context", Sensors, vol. 18, no. 6, pp. 1731, 2018.

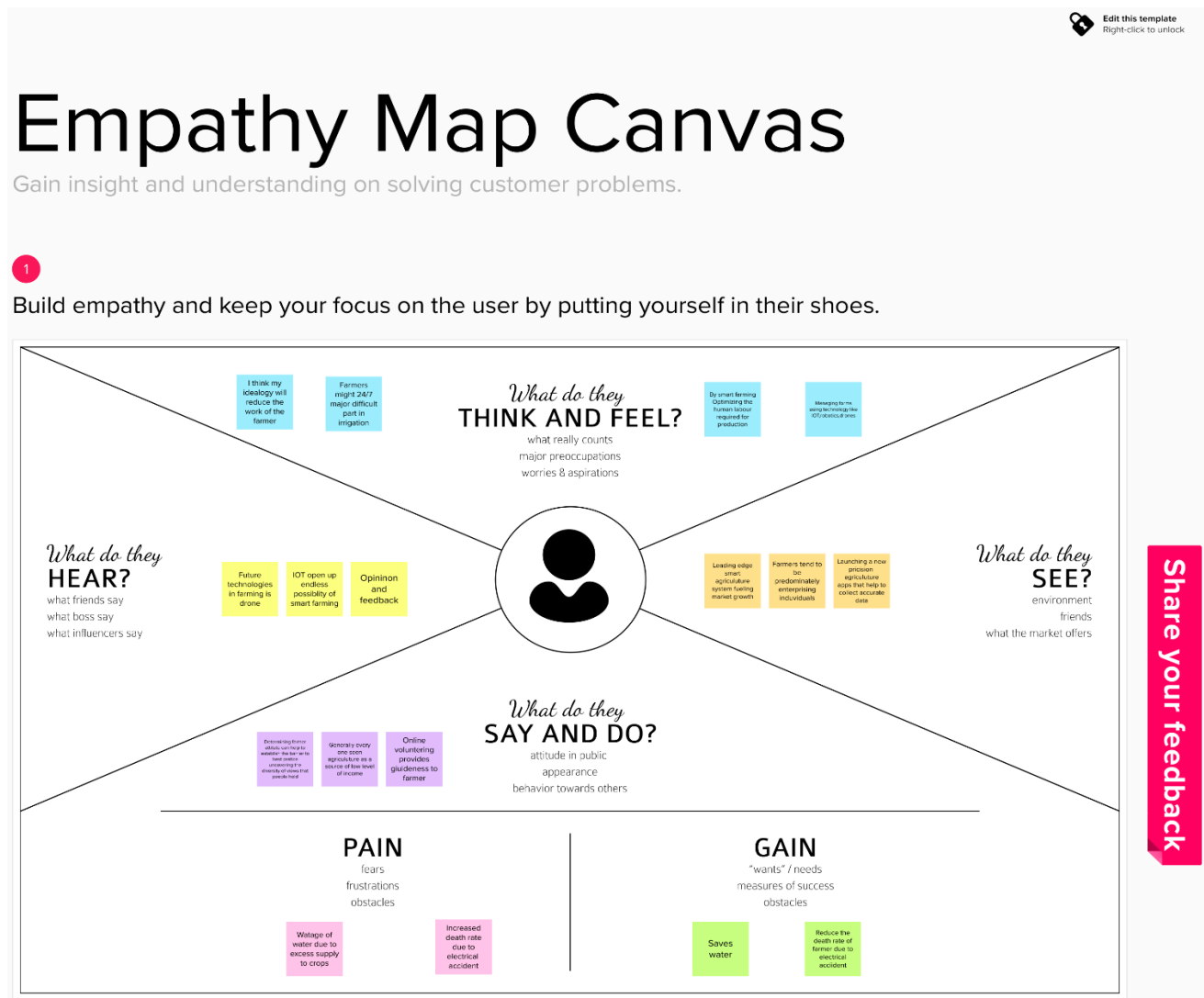
## **2.3 PROBLEM STATEMENT DEFINITION**

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 precision 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.

## 3.IDEATION AND PROPOSED SOLUTION

### 3.1 EMPATHY MAP CANVAS

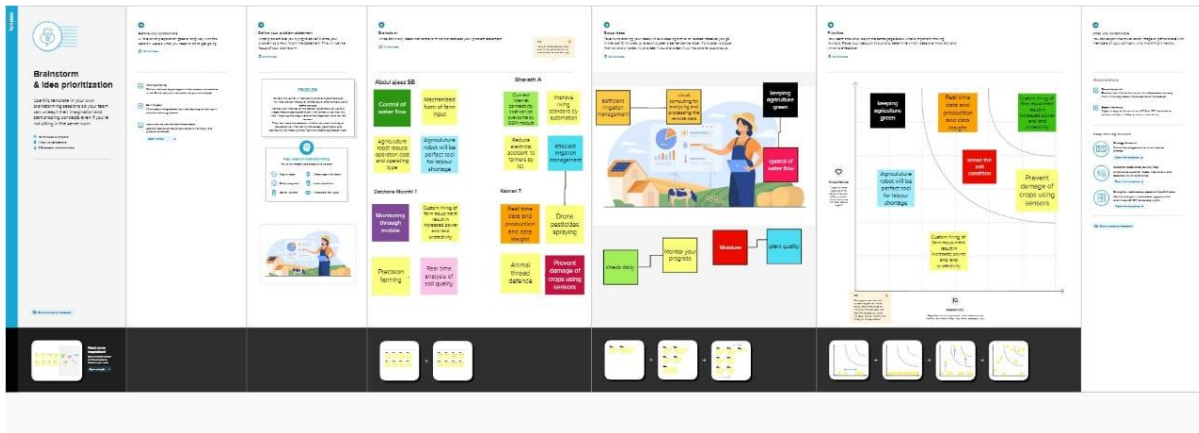
An empathy map is a **collaborative tool teams can use to gain a deeper insight into their customers**. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community.



### 3.2 IDEATION AND BRAINSTROMING

**Brainstorming** is one of the primary methods employed during the Ideation stage of a typical Design Thinking process. Brainstorming is a great way to

generate many ideas by leveraging the collective thinking of the group, engaging with each other, listening, and building on other ideas.



### 3.3 PROPOSED SOLUTION

In this activity we are expected to prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.

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

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	Factors such as climate changes, Population growth, Food security concerns should be protected and improve crop yield.
2.	Idea / Solution description	Farmers can monitor the crops from the analytical dashboard and take actions on insights using IoT Based remote sensing.
3.	Novelty / Uniqueness	All the required features along with the CAMERA facility which enables us to monitor the crops virtually in a more precise way.
4.	Social Impact / Customer Satisfaction	Doubles the Farmer income, Higher Production, Food safety

5.	Business Model (Revenue Model)	Government Driven Models, Contract farming models.
6.	Scalability of the Solution	Reliable with increased Performance and Efficiency and affordable cost

### 3.4 PROBLEM SOLUTION FIT

Problem-Solution Fit - **this occurs when you have evidence that customers care about certain jobs, pains, and gains.** At this stage you've proved the existence of a problem and have designed a value proposition that addresses your customers' jobs, pains and gains.

Project Design phase – I		
Problem Solution fit		
Project name : moisture control irrigation system		
Team Id : PNT2022TMID06254		
<b>1.Customer segments:-</b> the customers who are going to adapt this project contains of <ul style="list-style-type: none"> <li>• large scale farmers</li> <li>• remote farmers</li> </ul>	<b>6.Customer constrains:-</b> The customer wants a device which could solve the problems in irrigation when he is remote or absence of humans and that device should fulfill all the following constrains <ul style="list-style-type: none"> <li>• cost efficient</li> <li>• space efficient</li> <li>• time efficient</li> <li>• resource efficient</li> </ul>	<b>5.Available solutions</b> The moisture controlled irrigation system could be the best solution for this problem statement that has been provided by the farmers and also it specifically satisfies the customer constrains also
<b>2.Jobs to be done :-</b> the customers want to automate the process of irrigation in cost, energy and reduced power consumption and also reliable manner	<b>9.Problem route cause:-</b> The problem has its route stabled at the rate of the fast moving world since people move most of the times and since they have their work to be stagnated similarly farmers face the inability in the process of irrigation	<b>7.Behavior:-</b> The customer wants to make the revolutionary propagation in the rating of the irrigation through the reliability of amount of water availability on the land
<b>3.Triggers:-</b> The reliability and easy accessibility of this finished projects yields the peoples attraction have this project installed in their fields <b>4.Emotions:-</b> The customers feel happy and comfortable since our project reduces their work burden	<b>10.Solution -</b> Our solution for this project is to initiate the reliability of the irrigation system using the sensor sensed information from the field and also make the automation is on and off of water pump	<b>8.Channels of behavior:-</b> <ul style="list-style-type: none"> <li>• The channels of behavior recombines the ration of the following</li> <li>• Online</li> <li>• offline</li> </ul>



## 4. REQUIREMENT ANALYSIS

### 4.1 FUNCTIONAL REQUIREMENT

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	Data base management	Datas are collected in IBM Cloud Stored in the database
FR-4	Functionality	Designed for farmers with all standard requirements
FR-5	User Interface	System is provided with easy enabling via App
FR-6	Purpose	Control and Improve crop yield

### 4.2 NON-FUNCTIONAL REQUIREMENTS

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

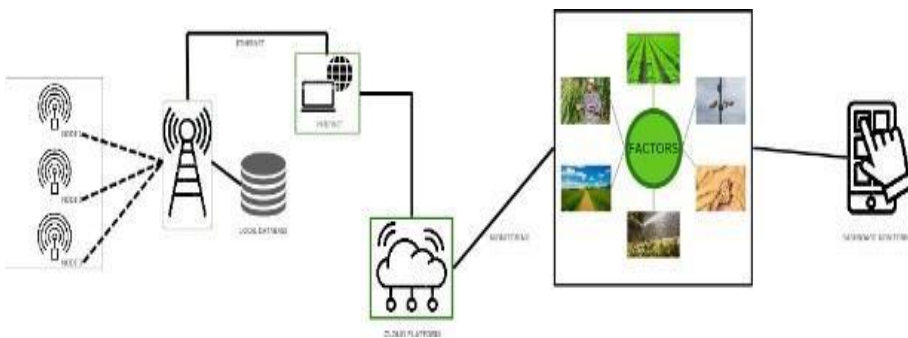
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	This application can be used by the farmers who seek for better solution in farming
NFR-2	Security	Every nodes and its data will be protected.
NFR-3	Reliability	Large number of services will be provided with trusted end to end connection to all the farmers
NFR-4	Performance	Optimized based on time and space complexity
NFR-5	Availability	For Farmers who have passion towards improving the good quality yield
NFR-6	Scalability	Scaled using microservice architecture

## 5.PROJECT DESIGN

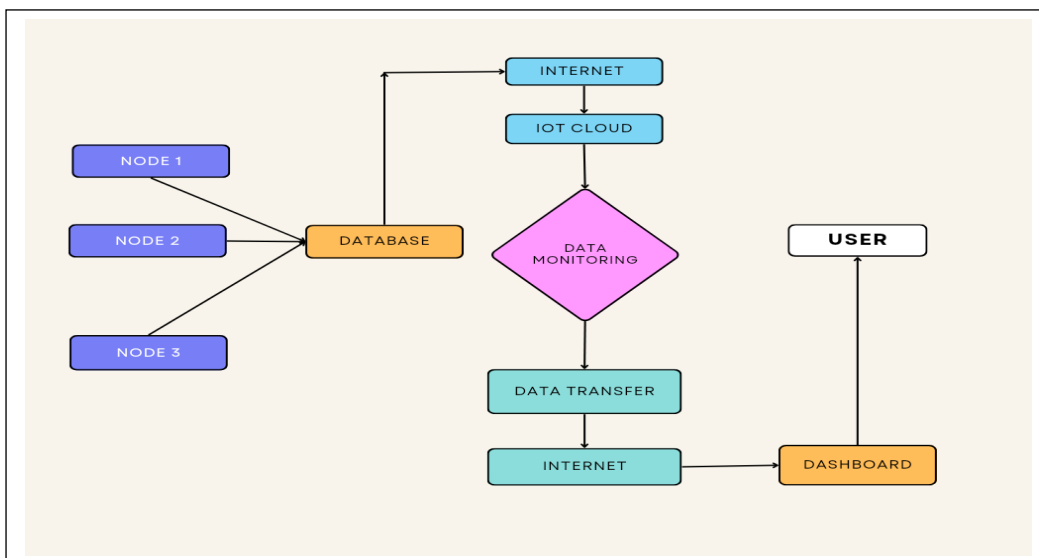
### 5.1 DATA FLOW DIAGRAM

A data flow diagram (DFD) is **a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement**. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method (SSADM).

#### SMART FARMER-ARCHITECTURE

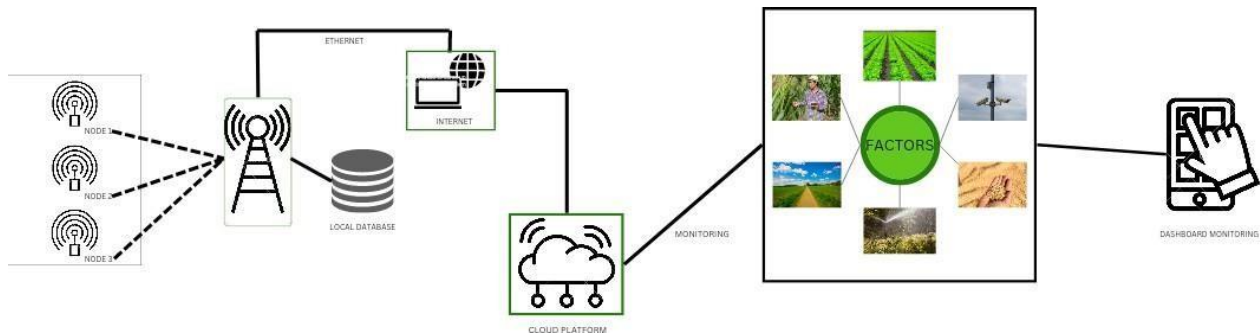


### FLOW DIAGRAM

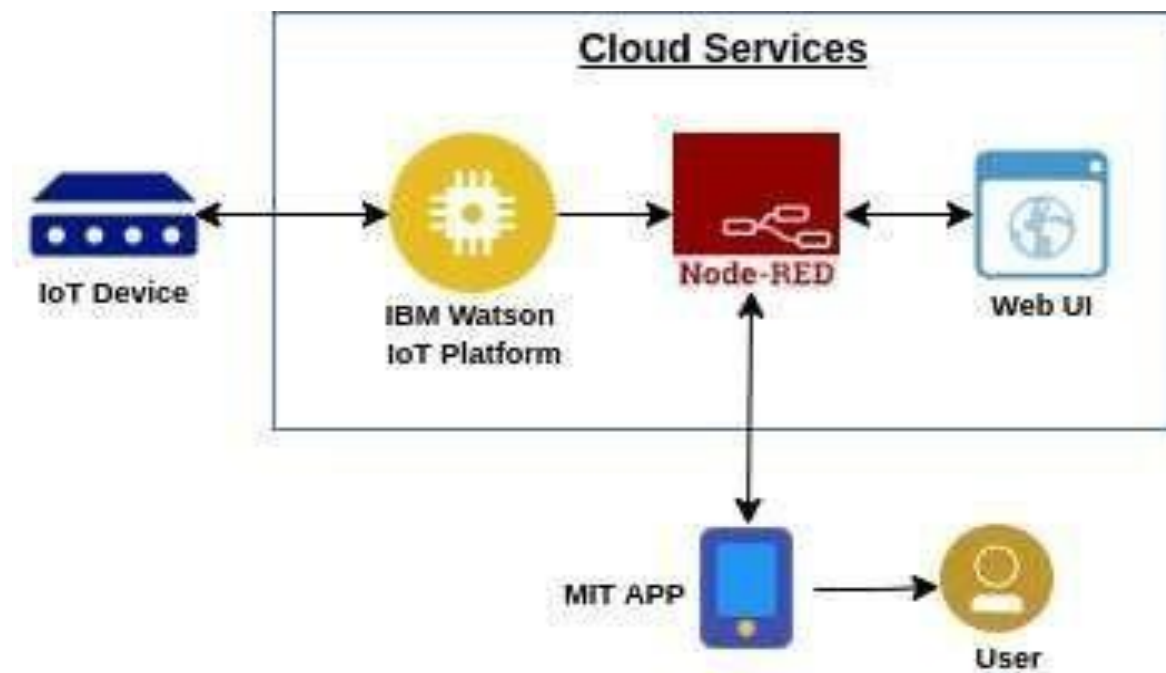


## 5.2 SOLUTION AND TECHNICAL ARCHITECTURE

### SOLUTION ARCHITECTURE



### TECHNICAL ARCHITECTURE



## COMPONENTS AND TECHNOLOGIES

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App	Python, IBM Cloud ,IBM IoT Platform,I BM Nodered, IBM Cloudant DB
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM IoT platform
4.	Application Logic-3	Logic for a process in the application	IBM Nodered , MIT App
5.	Database	Data Type, Configurations etc.	Python
6.	Cloud Database	Database Service on Cloud	IBM Cloud, IBM Cloudant DB .
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API	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.

### 5.3 USER STORIES

User stories **act as the common language between all participants in the development process**: product owners, architects, designers, and developers must share a common understanding of stories. You can achieve this common language by focusing on the value that each story provides to users.

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	I can login successfully	High	Sprint-1
	Dashboard	USN-5	As a user, I can access my dashboard after sign in	I can access my dashboard	High	Sprint-2
Customer (Web user)	Access	USN-6	As a user, I can register for the website by entering my email, password, and confirming my password and sign in anytime	I can access the account	High	Sprint-2
Customer Care Executive	Alert	USN-7	As a User, I get alerts and notifications	I can view alerts and notifications	Medium	Sprint-2
Administrator	Maintenance	USN-8	As an admin, I can view the progress of the application	I can modify application behaviour	Medium	Sprint-2

## **6.PROJECT PLANNING AND SCHEDULING**

### **6.1 SPRINT PLANNING AND ESTIMATION**

The purpose of sprint planning is **for the team to commit to complete a collection of stories that add new functionality to the product by the end of the sprint**. The team plans and estimates, and then decides how much work can fit into the sprint.

#### **Pre - Requisites**

- ☐ IBM Cloud Services
- ☐ MIT App Inventor
- ☐ Software
- ☐ Create An Account in Fast2smsDashboard

#### **Project Objectives**

- ☐ Abstract
- ☐ Brainstorming

#### **Create and Configure IBM Cloud Services**

- ☐ Create IBM Watson IOT Platform and Device
- ☐ Create Node- RED Service

#### **Develop the Python Script and Subscribe to IBM IOTPlatform**

- ☐ Develop A Python code

#### **Develop a Web Application Using Node - RED Service**

- ☐ Develop The Web Application Using Node - RED

#### **Develop a Mobile Application**

- ☐ Develop a Mobile Application

#### **Ideation Phase**

- ☐ Literature Survey on The Selected Project & Information Gathering
- ☐ Prepare Empathy Map
- ☐ Ideation

#### **Project Design Phase - 1**

- ☐ Proposed Solution
- ☐ Prepare Solution Fit
- ☐ Solution Architecture

#### **Project Design Phase - 2**

- ☐ Customer Journey
- ☐ Functional Requirement
- ☐ Data Flow Diagram
- ☐ Technology Architecture

## **Project Planning Phase**

- ☐ Prepare Milestones & Activity List
- ☐ Sprint Delivery Plan

## **Project Development Phase**

- ☐ Project Development - Delivery of Sprint - 1
- ☐ Project Development - Delivery of Sprint - 2
- ☐ Project Development - Delivery of Sprint - 3

## 6.2 SPRINT DELIVERY SCHEDULE

The deliverables of a sprint aren't as predictable as they are for other projects. Sprint participants have produced **sketches and drawings, writing, photographs, comic strips, videos and fully coded working prototypes**. The answer is whatever's right to answer the problem.

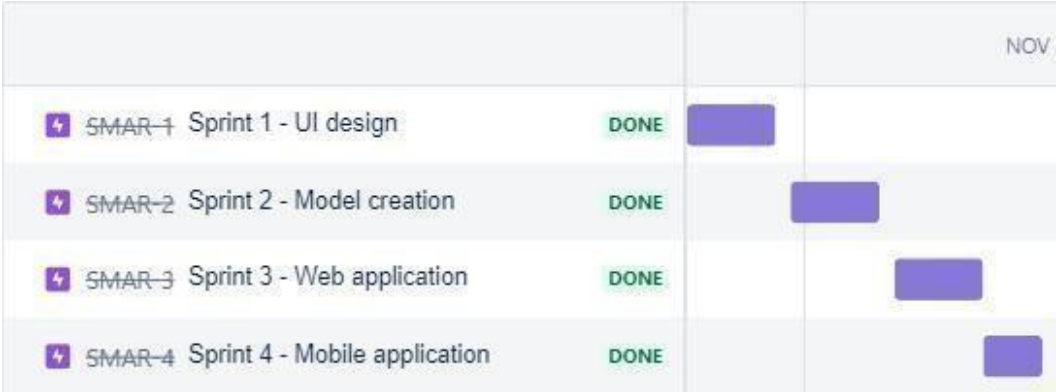
Sprint	Functional Requirement (Epic)	User story number	User Story / Task	Priority	Team Members
Sprint – 1	Creating Hardware Simulation	USN - 1	Connect Sensors and Wi - Fi modules by using Python code	High	Bharath A Abdul ajeez SB Datchana moorthi T Kannan T
Sprint – 2	Using Software	USN - 2	Creating device in the IBM Watson IOT platform, to making workflow of IOT scenarios using Node - RED service	High	Bharath A Abdul ajeez SB Datchana moorthi T Kannan T
Sprint - 3	MIT App Inventor	USN - 3	Develop a mobile application for the Smart Farmer project using MIT App Inventor	High	Bharath A Abdul ajeez SB Datchana moorthi T Kannan T
Sprint - 4	Web UI	USN - 4	To make the user to interact with software	High	Bharath A Abdul ajeez SB Datchana moorthi T Kannan T

## 6.3 REPORTS FROM JIRA

**JIRA is integrated with IBM Control Desk** and using IBM Control Desk capabilities, it is easier to manage the software projects and issues. Jira **helps teams plan, assign, track, report, and manage work and brings teams together for everything from agile software development and customer support to start-ups and enterprises**. Software teams build better with Jira Software, the #1 tool for agile teams. **Document Generation is a reporting app for Jira that supports templates**. Features: Templates in DOC, DOCX, RTF or ODT files. Single and multiple issue export to PDF, DOC, and PNG. Jira Software is **an agile project management tool that**

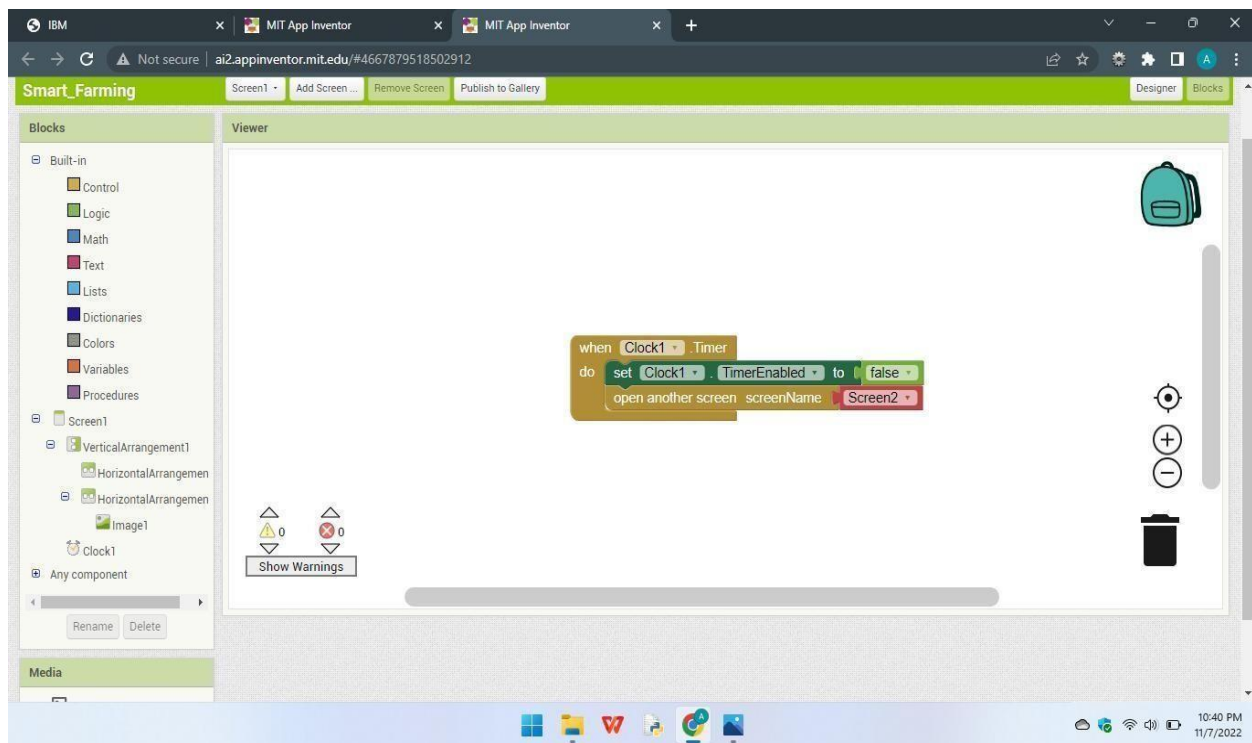
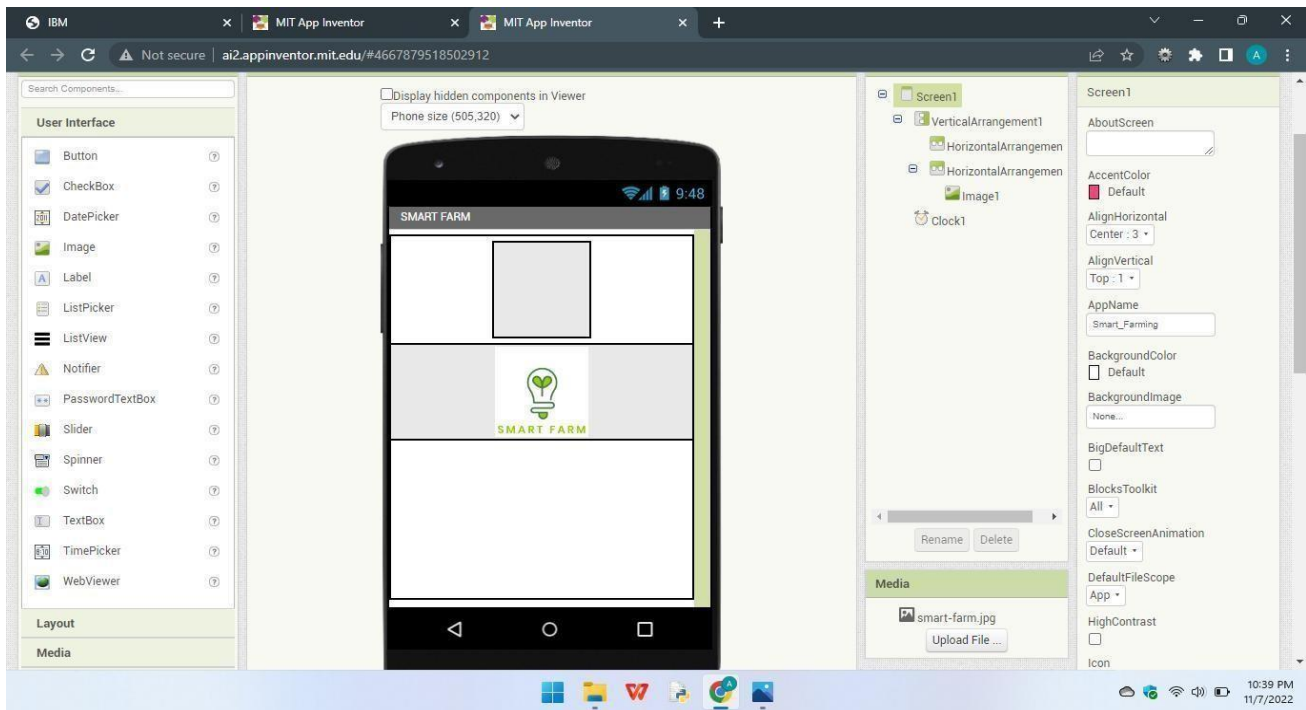


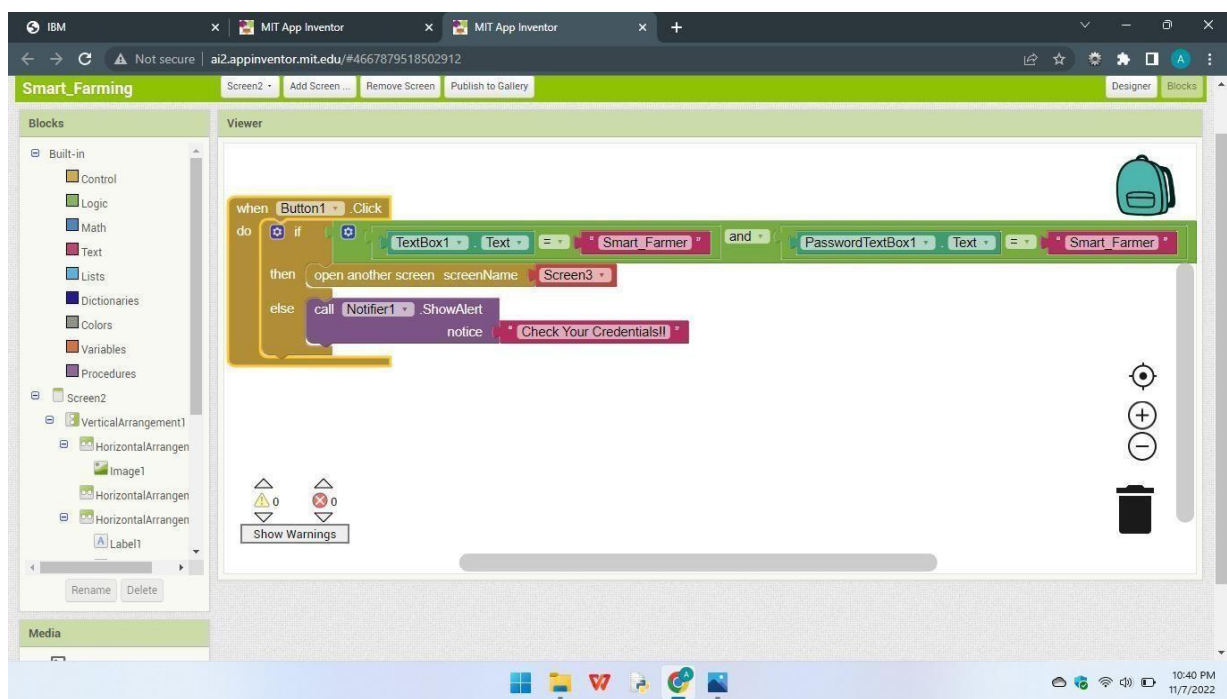
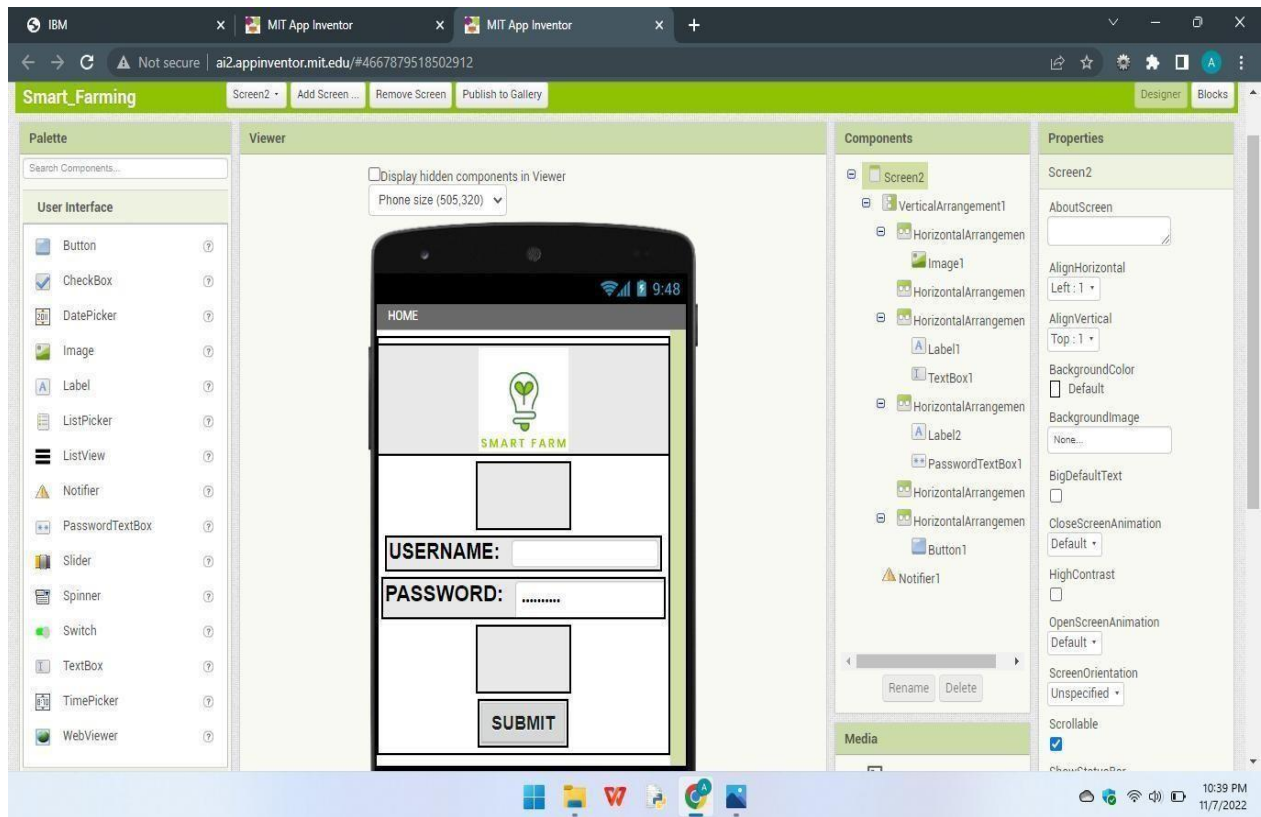
**supports any agile methodology, be it scrum, kanban, or your own unique flavor.** From agile boards, backlogs, roadmaps, reports, to integrations and add-ons you can plan, track, and manage all your agile software development projects from a single tool.

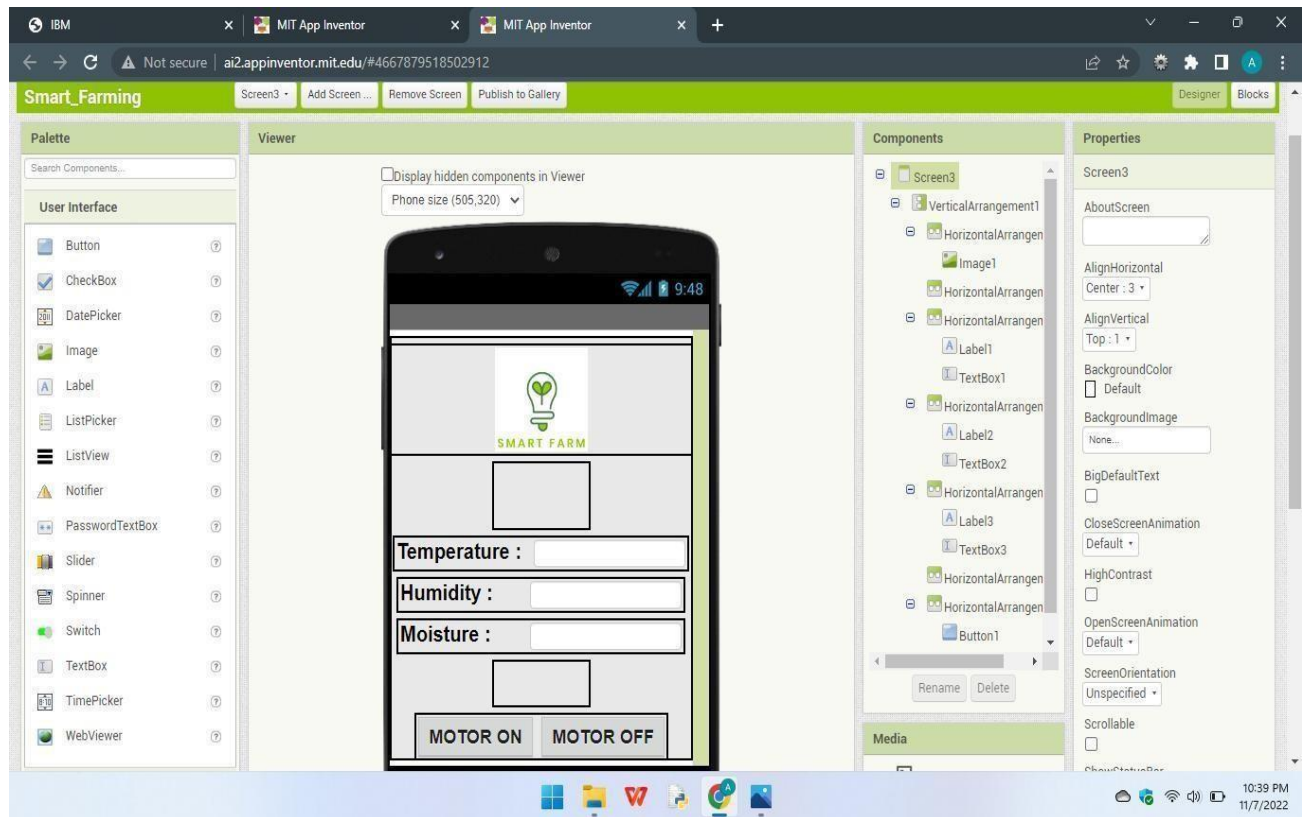


## 7.CODING AND SOLUTIONING

### 7.1Features - Development of Sprint-1 (User Interface Design)







USERNAME: Smart\_Farmer

PASSWORD: .....

Temperature :

Humidity :

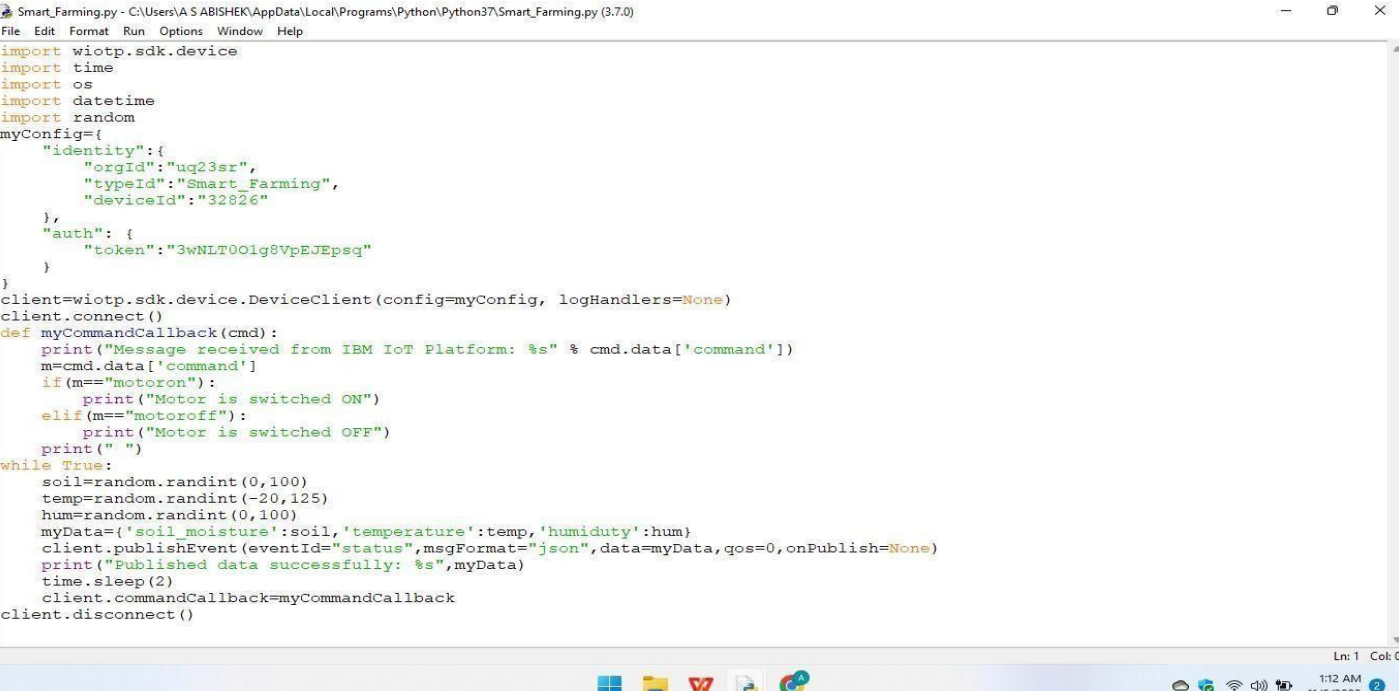
Moisture :

MOTOR ON MOTOR OFF

SUBMIT



## Development of Sprint-2 (Python Code for Publish & Subscribe)



```
Smart_Farming.py - C:\Users\A S ABISHEK\AppData\Local\Programs\Python\Python37\Smart_Farming.py (3.7.0)
File Edit Format Run Options Window Help
import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig={
    "identity":{
        "orgId":"ug23sr",
        "typeId":"Smart_Farming",
        "deviceId":"32826"
    },
    "auth": {
        "token":"3wNLT00Ig8VpEJEpsq"
    }
}
client=wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="motoron"):
        print("Motor is switched ON")
    elif(m=="motoroff"):
        print("Motor is switched OFF")
    print(" ")
while True:
    soil=random.randint(0,100)
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'soil_moisture':soil,'temperature':temp,'humidity':hum}
    client.publishEvent(eventId="status",msgFormat="json",data=myData,qos=0,onPublish=None)
    print("Published data successfully: %s",myData)
    time.sleep(2)
    client.commandCallback=myCommandCallback
client.disconnect()
```

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\A S ABISHEK\AppData\Local\Programs\Python\Python37\Smart_Farming.py
2022-11-06 01:11:23.500 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:uq23sr:Smart_Farming:32826Published
data successfully: %s
{'soil_moisture': 83, 'temperature': 29, 'humidity': 36}
Published data successfully: %s {'soil_moisture': 47, 'temperature': 96, 'humidity': 25}
Published data successfully: %s {'soil_moisture': 45, 'temperature': 5, 'humidity': 49}
Published data successfully: %s {'soil_moisture': 82, 'temperature': 29, 'humidity': 80}
Published data successfully: %s {'soil_moisture': 92, 'temperature': 49, 'humidity': 86}
Published data successfully: %s {'soil_moisture': 60, 'temperature': 11, 'humidity': 98}
Published data successfully: %s {'soil_moisture': 40, 'temperature': -18, 'humidity': 100}
Published data successfully: %s {'soil_moisture': 17, 'temperature': 80, 'humidity': 33}
Published data successfully: %s {'soil_moisture': 60, 'temperature': 93, 'humidity': 95}
Published data successfully: %s {'soil_moisture': 22, 'temperature': 15, 'humidity': 70}
Published data successfully: %s {'soil_moisture': 78, 'temperature': 72, 'humidity': 15}
Published data successfully: %s {'soil_moisture': 42, 'temperature': 41, 'humidity': 63}
Published data successfully: %s {'soil_moisture': 21, 'temperature': 61, 'humidity': 50}
Published data successfully: %s {'soil_moisture': 36, 'temperature': 95, 'humidity': 56}
Published data successfully: %s {'soil_moisture': 14, 'temperature': 102, 'humidity': 74}
Published data successfully: %s {'soil_moisture': 34, 'temperature': 31, 'humidity': 44}
Published data successfully: %s {'soil_moisture': 95, 'temperature': 1, 'humidity': 37}
Published data successfully: %s {'soil_moisture': 58, 'temperature': 61, 'humidity': 97}
Published data successfully: %s {'soil_moisture': 94, 'temperature': 86, 'humidity': 14}
Published data successfully: %s {'soil_moisture': 56, 'temperature': 89, 'humidity': 82}
Published data successfully: %s {'soil_moisture': 100, 'temperature': -12, 'humidity': 61}
Published data successfully: %s {'soil_moisture': 82, 'temperature': 63, 'humidity': 41}
Published data successfully: %s {'soil_moisture': 30, 'temperature': 53, 'humidity': 84}
Published data successfully: %s {'soil_moisture': 42, 'temperature': 83, 'humidity': 71}
Published data successfully: %s {'soil_moisture': 80, 'temperature': 107, 'humidity': 38}
Published data successfully: %s {'soil_moisture': 64, 'temperature': 14, 'humidity': 73}
Published data successfully: %s {'soil_moisture': 95, 'temperature': 0, 'humidity': 100}
Published data successfully: %s {'soil_moisture': 67, 'temperature': 124, 'humidity': 21}
Published data successfully: %s {'soil_moisture': 51, 'temperature': 109, 'humidity': 5}
Published data successfully: %s {'soil_moisture': 25, 'temperature': 35, 'humidity': 58}
```

Browse

Action

Device Types

Interfaces

Add Device +

32826

Connected

Smart\_Farming

Device

Oct 28, 2022 11:29 PM

→

...

Identity

Device Information

Recent Events

State

Logs

×

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

Event	Value	Format	Last Received
status	{"soil_moisture":89,"temperature":80,"humiduty...	json	a few seconds ago
status	{"soil_moisture":44,"temperature":93,"humiduty...	json	a few seconds ago
status	{"soil_moisture":34,"temperature":59,"humiduty...	json	a few seconds ago
status	{"soil_moisture":18,"temperature":76,"humiduty...	json	a few seconds ago
status	{"soil_moisture":71,"temperature":123,"humidut...	json	a few seconds ago

0 Simulations running

11/6/2022

1:13 AM

# Development of Sprint-3 (Web Application Using Node-Red)

```
Smart_Farming.py - C:\Users\A S ABISHEK\AppData\Local\Programs\Python\Python37\Smart_Farming.py (3.7.0)
File Edit Format Run Options Window Help

import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig={
    "identity":{
        "orgId":"ug23sr",
        "typeId":"Smart_Farming",
        "deviceId":"32826"
    },
    "auth": {
        "token":"3wNLT001g8VpEJEpsq"
    }
}
client=wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="motoron"):
        print("Motor is switched ON")
    elif(m=="motoroff"):
        print("Motor is switched OFF")
    print(" ")
while True:
    soil=random.randint(0,100)
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'soil_moisture':soil,'temperature':temp,'humidity':hum}
    client.publishEvent(eventId="status",msgFormat="json",data=myData,qos=0,onPublish=None)
    print("Published data successfully",myData)
    time.sleep(2)
    client.commandCallback=myCommandCallback
client.disconnect()
```

```
*Python 3.7.0 Shell*
File Edit Shell Debug Options Window Help

Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\A S ABISHEK\AppData\Local\Programs\Python\Python37\Smart_Farming.py
2022-11-10 16:04:42,463 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:ug23sr:Smart_Farming:32826Published
data successfully
{'soil_moisture': 37, 'temperature': 1, 'humidity': 35}
Published data successfully {'soil_moisture': 89, 'temperature': 94, 'humidity': 24}
Published data successfully {'soil_moisture': 57, 'temperature': 28, 'humidity': 90}
Published data successfully {'soil_moisture': 65, 'temperature': -18, 'humidity': 4}
Published data successfully {'soil_moisture': 87, 'temperature': 81, 'humidity': 92}
Published data successfully {'soil_moisture': 62, 'temperature': -16, 'humidity': 33}
Published data successfully {'soil_moisture': 99, 'temperature': 105, 'humidity': 62}
Published data successfully {'soil_moisture': 41, 'temperature': 114, 'humidity': 78}
Published data successfully {'soil_moisture': 26, 'temperature': -15, 'humidity': 49}
Published data successfully {'soil_moisture': 55, 'temperature': 84, 'humidity': 87}
```

Service Details - IBM Cloud

IBM Watson IoT Platform

+

← → ↻ y13urg.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform

ravisubir73@gmail.com  
ID: y13urg

Browse

Action

Device Types

Interfaces

Add Device

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device Simulator

<input type="checkbox"/>	Device ID	Status	Device Type	Class ID	Date Added
> <input type="checkbox"/>	12345	Disconnected	Arduino	Device	Nov 3, 2022 2:15 PM
> <input type="checkbox"/>	Arduino_1	Disconnected	Arduino	Device	Nov 3, 2022 2:16 PM
> <input type="checkbox"/>	TEST	Connected	ESP8266	Device	Nov 19, 2022 8:53 PM

Items per page 50 | 1-3 of 3 items

1 of 1 page

Type here to search

22°C Haze

21:28  
19-11-2022

Browse

Action

Device Types

Interfaces

Add Device

<input type="checkbox"/>	Device ID	Status	Device Type	Class ID	Date Added
▼ <input type="checkbox"/>	32826	Connected	Smart_Farming	Device	Oct 28, 2022 11:29 PM

Identity Device Information Recent Events State Logs

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

Event	Value	Format	Last Received
status	{"soil_moisture":29,"temperature":99,"humidity"...	json	a few seconds ago
status	{"soil_moisture":2,"temperature":42,"humidity"...	json	a few seconds ago
status	{"soil_moisture":2,"temperature":37,"humidity"...	json	a few seconds ago
status	{"soil_moisture":94,"temperature":106,"humidit...	json	a few seconds ago
status	{"soil_moisture":71,"temperature":-6,"humidity"...		

0 Simulations running

4:07 PM  
11/10/2022



Service Details - IBM Cloud x IBM Watson IoT Platform x Node-RED: node-red-zncis x Getting Started with MIT App Inventor x MIT App Inventor

node-red-zncis-2022-11-04-au-syd.mybluemix.net/red/#flow/f23f5cad061e8487

### Node-RED

filter nodes

Flow 1

common

- inject
- debug
- complete
- catch
- status
- link in
- link call
- link out
- comment

function

- function
- switch

debug

msg.payload : number

95

10/11/2022, 16:09:05 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evl/status/fmt/json :  
msg.payload : number  
71

10/11/2022, 16:09:05 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evl/status/fmt/json :  
msg.payload : number  
93

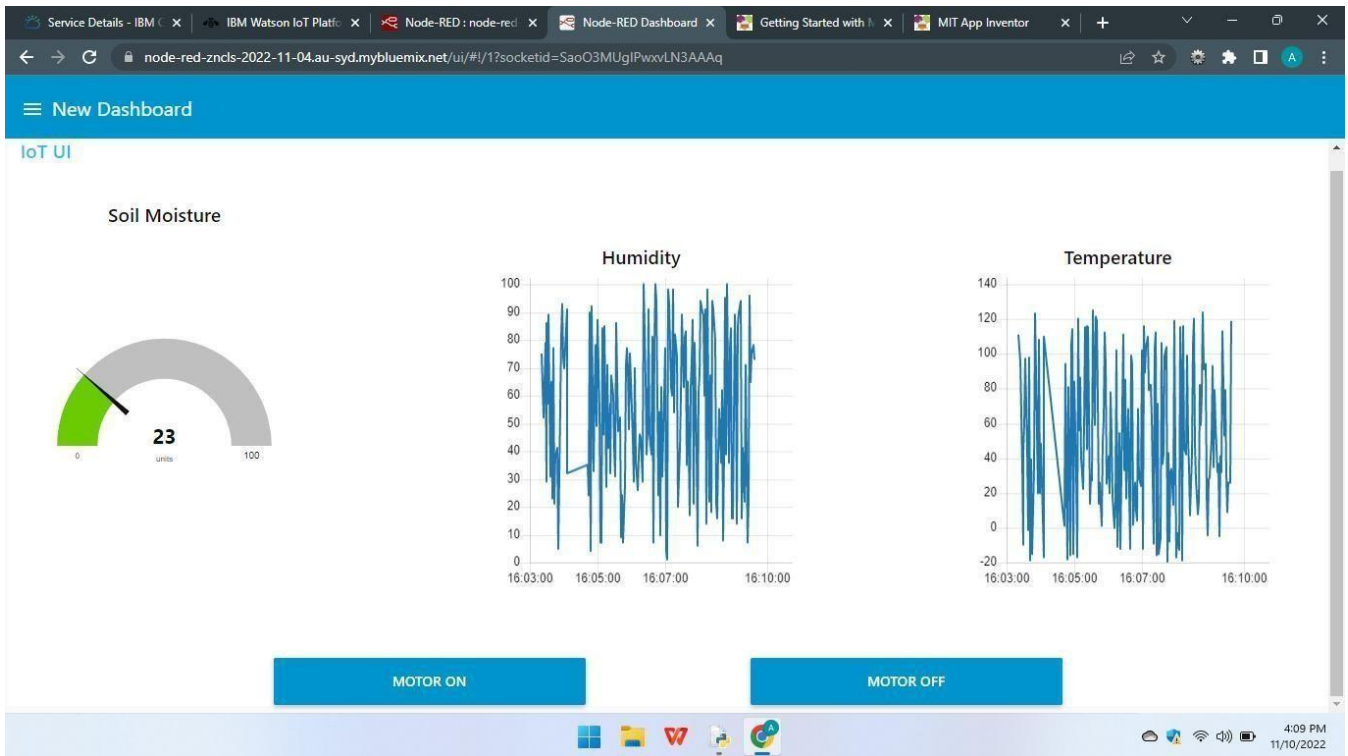
10/11/2022, 16:09:07 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evl/status/fmt/json :  
msg.payload : number  
16

10/11/2022, 16:09:07 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evl/status/fmt/json :  
msg.payload : number  
14

10/11/2022, 16:09:07 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evl/status/fmt/json :  
msg.payload : number  
35

Flow 1 diagram:

```
graph LR
    IBM IoT[IBM IoT] --> Soil Moisture[Soil Moisture]
    IBM IoT --> Humidity[Humidity]
    IBM IoT --> Temperature[Temperature]
    Soil Moisture --> Soil Moisture Out[Soil Moisture]
    Humidity --> Humidity Out[Humidity]
    Temperature --> Temperature Out[Temperature]
    Temperature --> msg payload 1[msg payload]
    msg payload 1 --> http 1[http]
    [get] /data --> data[data]
    data --> http 2[http]
    MOTOR ON[MOTOR ON] --> IBM IoT
    MOTOR OFF[MOTOR OFF] --> IBM IoT
    MOTOR ON --> msg payload 2[msg payload]
    MOTOR OFF --> msg payload 2
    msg payload 2 --> http 3[http]
    [get] /command --> http 3
```



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Published data successfully {'soil_moisture': 50, 'temperature': 112, 'humidity': 74}
Published data successfully {'soil_moisture': 54, 'temperature': 112, 'humidity': 79}
Published data successfully {'soil_moisture': 37, 'temperature': -6, 'humidity': 74}
Published data successfully {'soil_moisture': 69, 'temperature': 96, 'humidity': 83}
Published data successfully {'soil_moisture': 57, 'temperature': 88, 'humidity': 84}
Published data successfully {'soil_moisture': 25, 'temperature': 54, 'humidity': 99}
Published data successfully {'soil_moisture': 27, 'temperature': 16, 'humidity': 7}
Published data successfully {'soil_moisture': 28, 'temperature': -1, 'humidity': 100}
Published data successfully {'soil_moisture': 64, 'temperature': 69, 'humidity': 19}
Published data successfully {'soil_moisture': 9, 'temperature': 39, 'humidity': 86}
Published data successfully {'soil_moisture': 8, 'temperature': -3, 'humidity': 61}
Published data successfully {'soil_moisture': 41, 'temperature': 61, 'humidity': 49}
Published data successfully {'soil_moisture': 87, 'temperature': 92, 'humidity': 8}
Published data successfully {'soil_moisture': 84, 'temperature': 92, 'humidity': 84}
Message received from IBM IoT Platform: motoron
Motor is switched ON
Published data successfully {'soil_moisture': 80, 'temperature': 26, 'humidity': 99}
Message received from IBM IoT Platform: motoroff
Motor is switched OFF
Published data successfully {'soil_moisture': 31, 'temperature': 108, 'humidity': 46}
Published data successfully {'soil_moisture': 36, 'temperature': 86, 'humidity': 69}
Published data successfully {'soil_moisture': 49, 'temperature': 99, 'humidity': 34}
Published data successfully {'soil_moisture': 91, 'temperature': 90, 'humidity': 15}
Published data successfully {'soil_moisture': 99, 'temperature': 75, 'humidity': 2}
Published data successfully {'soil_moisture': 25, 'temperature': 2, 'humidity': 99}
Published data successfully {'soil_moisture': 61, 'temperature': 7, 'humidity': 61}
Published data successfully {'soil_moisture': 17, 'temperature': 39, 'humidity': 85}
Published data successfully {'soil_moisture': 89, 'temperature': 51, 'humidity': 61}
Published data successfully {'soil_moisture': 72, 'temperature': 18, 'humidity': 7}
Published data successfully {'soil_moisture': 7, 'temperature': 42, 'humidity': 36}
Published data successfully {'soil_moisture': 67, 'temperature': -4, 'humidity': 94}
Published data successfully {'soil_moisture': 21, 'temperature': 41, 'humidity': 74}
Published data successfully {'soil_moisture': 26, 'temperature': 114, 'humidity': 71}
Published data successfully {'soil_moisture': 89, 'temperature': -2, 'humidity': 48}
Published data successfully {'soil_moisture': 10, 'temperature': -12, 'humidity': 2}
Ln: 406 Col: 0
4:18 PM 11/10/2022
```

Service Details - IBM x IBM Watson IoT Platf x Node-RED: node-red x Node-RED Dashboard x Getting Started with x MIT App Inventor x

node-red-zncds-2022-11-04.au-syd.mybluemix.net/red/#flow/f23f5cad061e8487

Node-RED

Flow 1

common

- inject
- debug
- complete
- catch
- status
- link in
- link call
- link out
- comment

function

- function
- switch

debug

msg.payload: number  
92

10/11/2022, 16:15:47 node: 47237c95f9032d61  
msg.payload: Object  
{ command: "motoron" }

10/11/2022, 16:15:48 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evt/status/fml/json:  
msg.payload: number  
80

10/11/2022, 16:15:49 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evt/status/fml/json:  
msg.payload: number  
99

10/11/2022, 16:15:50 node: c7bc968f68e5d4e5  
iot-2/type/Smart\_Farming/id/32826/evt/status/fml/json:  
msg.payload: number  
26

10/11/2022, 16:15:50 node: 47237c95f9032d61  
msg.payload: Object  
{ command: "motoroff" }

10/11/2022, 16:15:51 node: c7bc968f68e5d4e5

The flow diagram in Node-RED consists of several interconnected nodes:

- IBM IoT** (connected) feeds into three function nodes: **Soil Moisture**, **Humidity**, and **Temperature**.
- Each of these function nodes connects to a corresponding **msg.payload** node (e.g., **Soil Moisture** connects to **Soil Moisture**).
- The **msg.payload** nodes connect to a **data** function node.
- The **data** function node connects to an **http** node.
- There are also **MOTOR ON** and **MOTOR OFF** nodes that connect to an **IBM IoT** (connected) node and a **msg.payload** node.
- A **[get] /command** node connects to an **http** node.

# Development of Sprint-4 (MIT Application and its Execution)

```
Smart_Farming.py - C:\Users\A S ABISHEK\AppData\Local\Programs\Python\Python37\Smart_Farming.py (3.7.0)
File Edit Format Run Options Window Help

import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig={
    "identity":{
        "orgId":"ug23sr",
        "typeId":"Smart_Farming",
        "deviceId":"32826"
    },
    "auth": {
        "token":"3wNLT00lg8VpEJEpsq"
    }
}
client=wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="motoron"):
        print("Motor is switched ON")
    elif(m=="motoroff"):
        print("Motor is switched OFF")
    print(" ")
while True:
    soil=random.randint(0,100)
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'soil_moisture':soil,'temperature':temp,'humidity':hum}
    client.publishEvent(eventId="status",msgFormat="json",data=myData,qos=0,onPublish=None)
    print("Published data successfully",myData)
    time.sleep(2)
    client.commandCallback=myCommandCallback
client.disconnect()
```

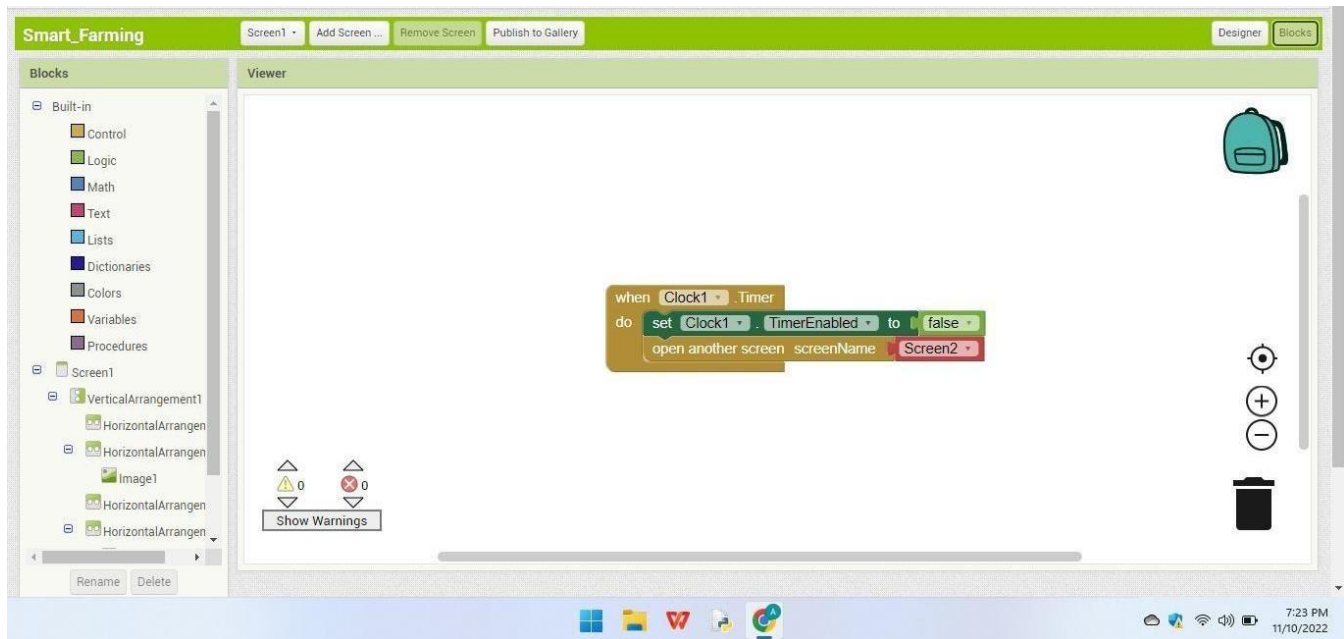
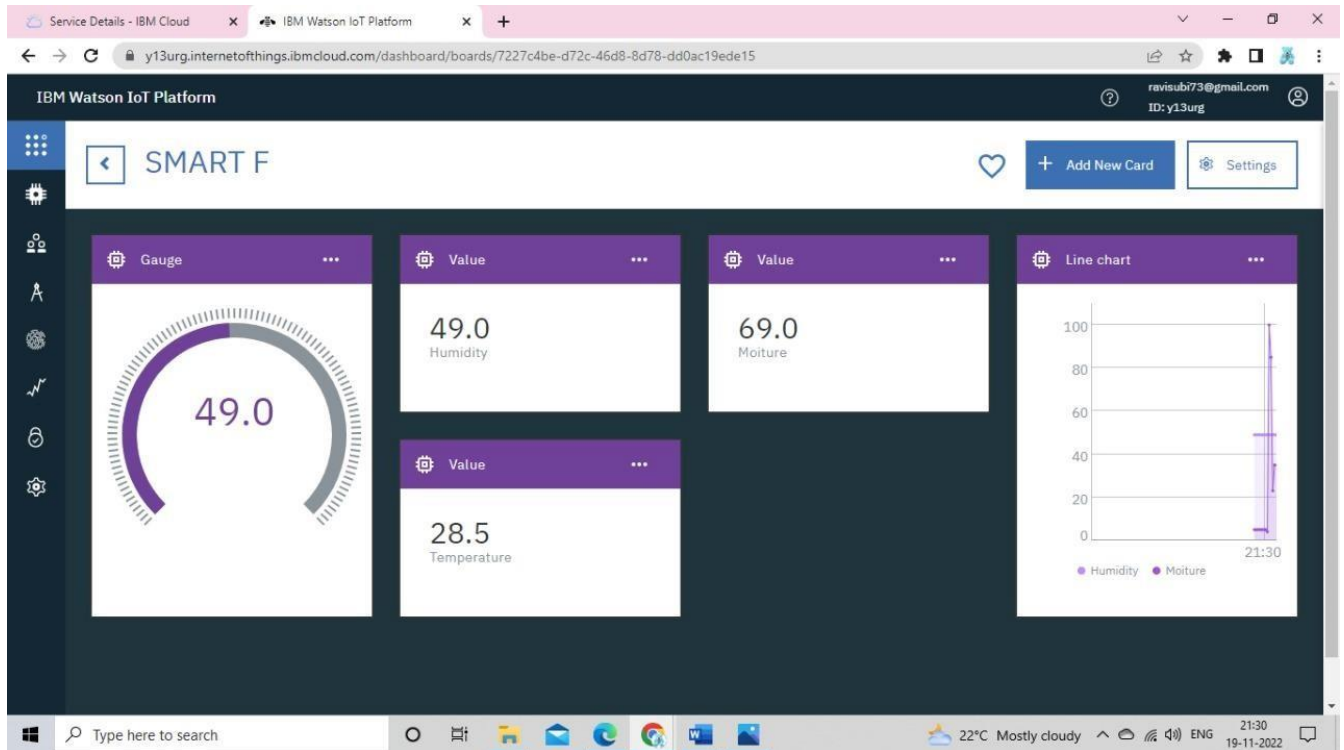
Ln: 32 Col: 38  
4:04 PM  
11/10/2022

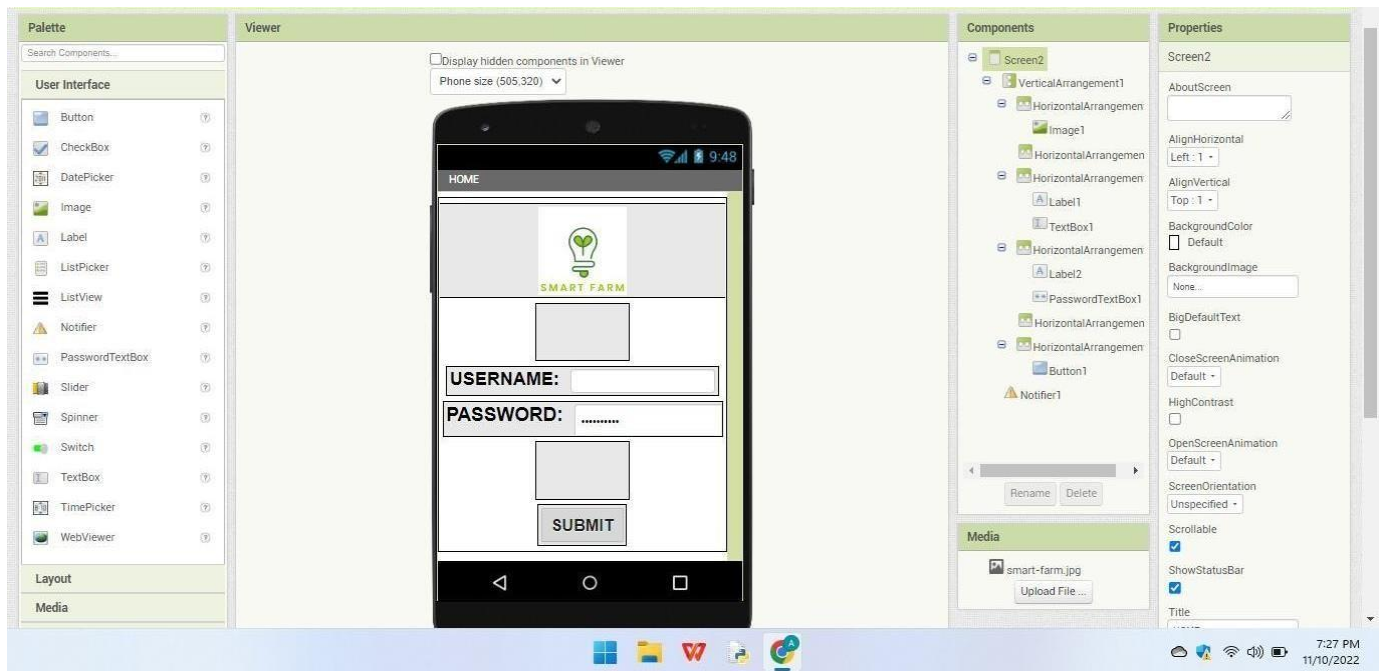
```
*Python 3.7.0 Shell*
File Edit Shell Debug Options Window Help

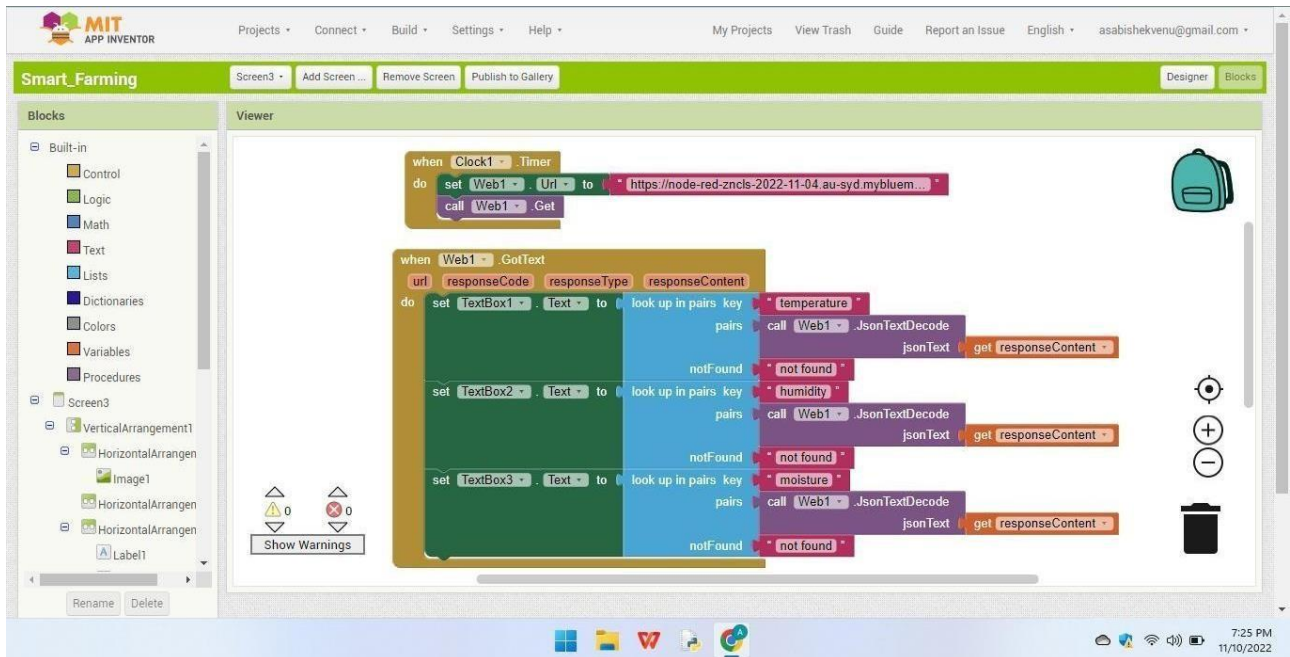
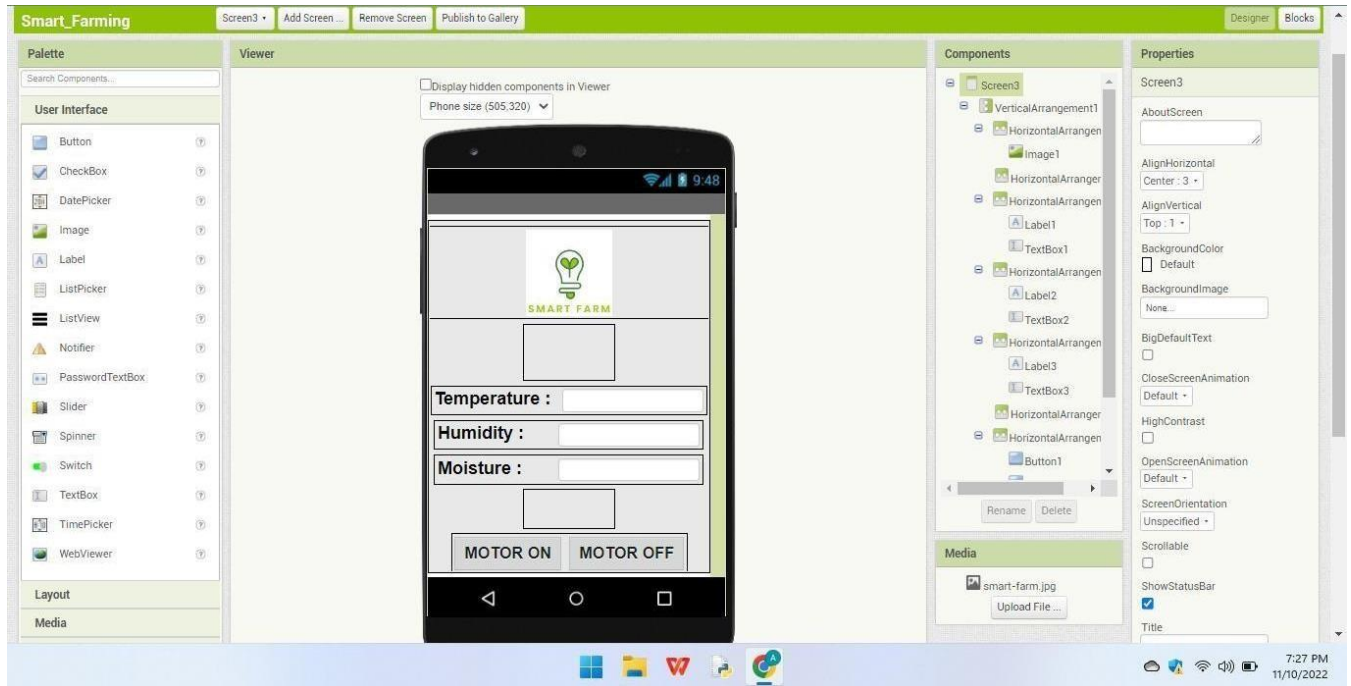
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\A S ABISHEK\AppData\Local\Programs\Python\Python37\Smart_Farming.py
2022-11-10 16:04:42,463 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:ug23sr:Smart_Farming:32826Published
data successfully
{'soil_moisture': 37, 'temperature': 1, 'humidity': 35}
Published data successfully {'soil_moisture': 89, 'temperature': 94, 'humidity': 24}
Published data successfully {'soil_moisture': 57, 'temperature': 28, 'humidity': 90}
Published data successfully {'soil_moisture': 65, 'temperature': -18, 'humidity': 4}
Published data successfully {'soil_moisture': 87, 'temperature': 81, 'humidity': 92}
Published data successfully {'soil_moisture': 62, 'temperature': -16, 'humidity': 33}
Published data successfully {'soil_moisture': 99, 'temperature': 105, 'humidity': 62}
Published data successfully {'soil_moisture': 41, 'temperature': 114, 'humidity': 78}
Published data successfully {'soil_moisture': 26, 'temperature': -15, 'humidity': 49}
Published data successfully {'soil_moisture': 55, 'temperature': 84, 'humidity': 87}
```

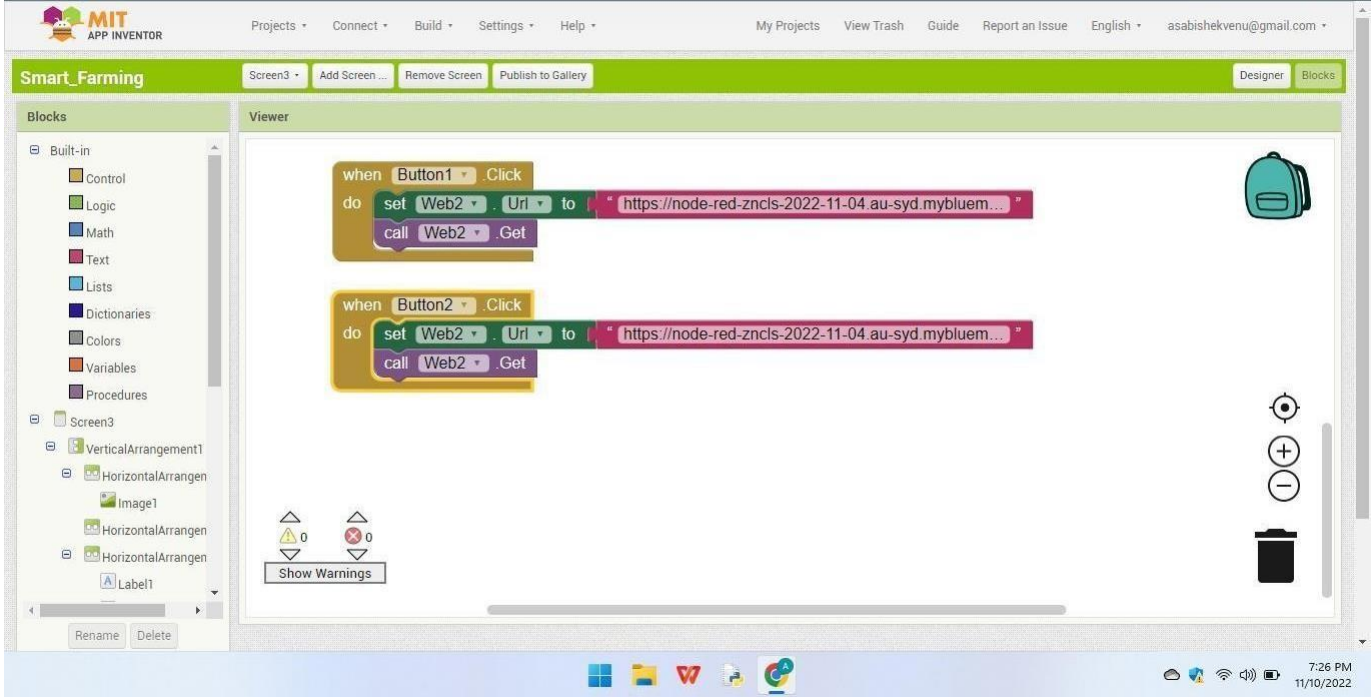
Ln: 5 Col: 0  
4:05 PM  
11/10/2022















SMART FARM

USERNAME:

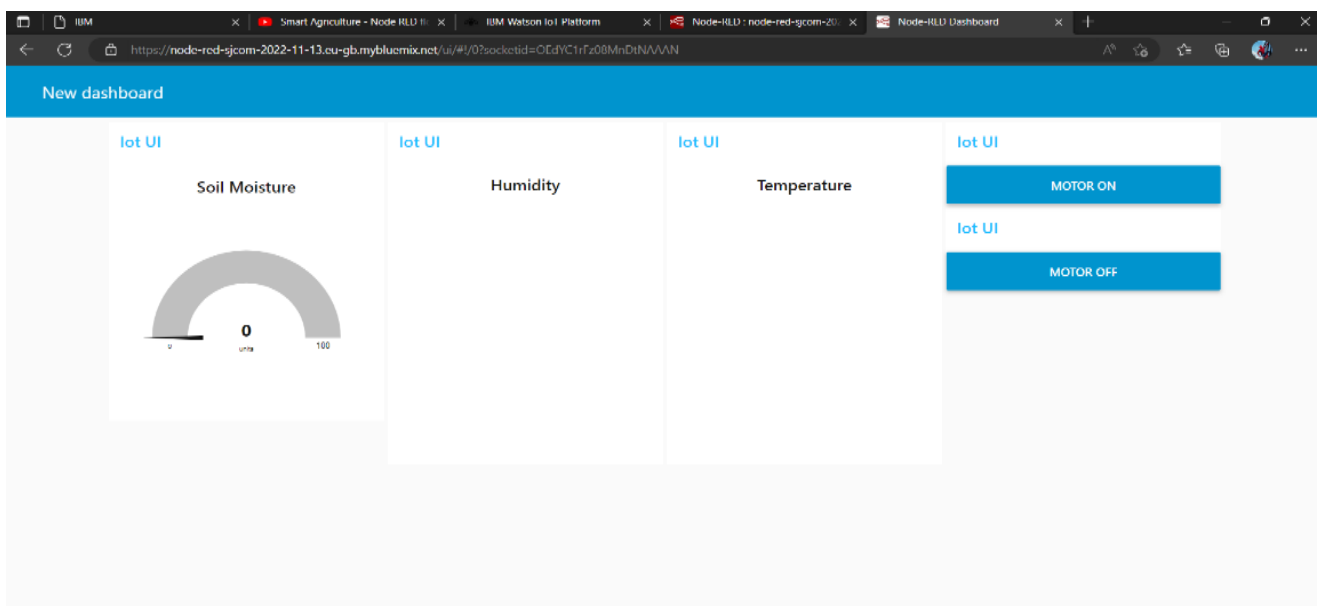
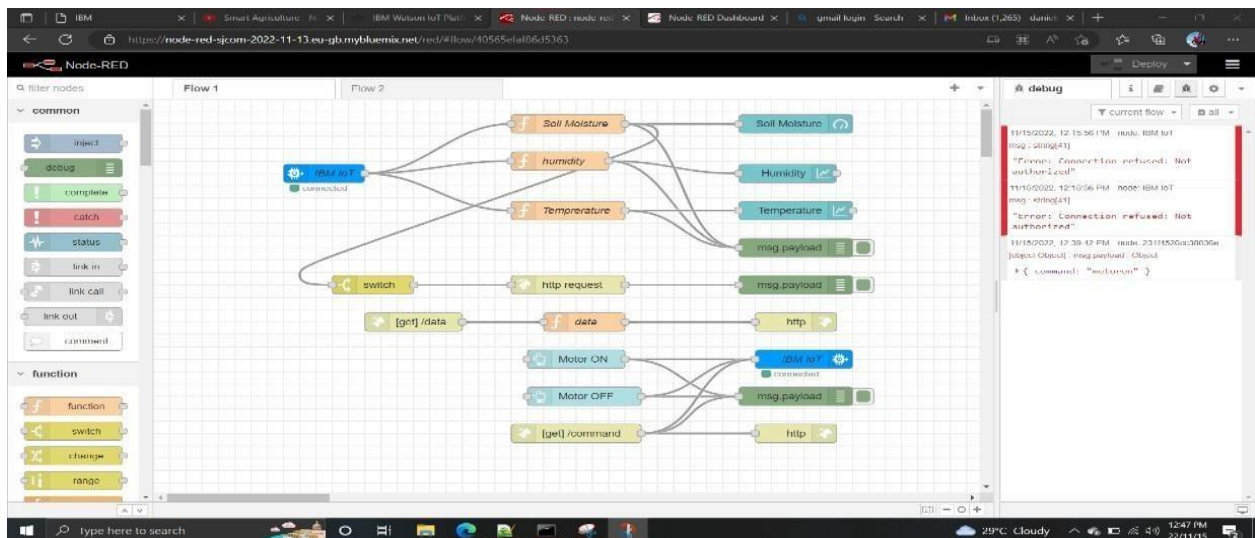
PASSWORD:

SUBMIT

## 8.TESTING

### 8.1 TEST CASES

In software engineering, a **test case** is a specification of the inputs, execution conditions, testing procedure, and expected results that define a single test to be executed to achieve a particular software testing objective, such as to exercise a particular program path or to verify compliance with a specific requirement. Test cases underlie testing that is methodical rather than haphazard. A battery of test cases can be built to produce the desired coverage of the software being tested. Formally defined test cases allow the same tests to be run repeatedly against successive versions of the software, allowing for effective and consistent regression testing.



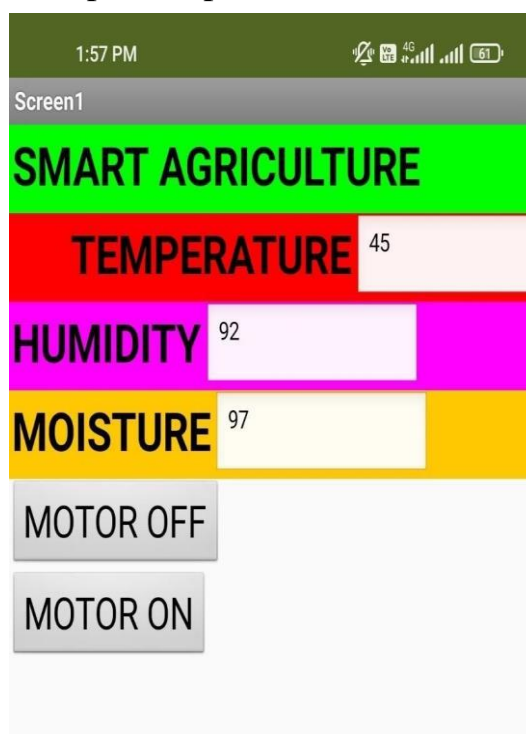
```

C:\Users\hp\OneDrive\Desktop\JavaPrograms\smartfarmer.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Python file
16 client = wiotsdk.device.DeviceClient(config=myConfig,logHandlers=None)
17 client.connect ()
18 def mycommandcallback (cmd) :
19     print ("Message received from IBM IoT Platform: %s" %cmd.data['command'])
20     mycmd.data['command']
21     if (cmd=="motoron"):
22         print ("Motor is switchedon")
23     elif (cmd=="motortoff"):
24         print ("Motor is switchedOFF")
25     print (" ")
26 while True:
27     soil=random.randint (0,100)
28     temp=random.randint (-20, 125)
29     hum=random.randint (0, 100)
30     mydata={'soil moisture':soil,'temperature':temp,'humidity':hum}
31     client.publishEvent (eventId="status", msgFormat="json", data=myData, qos=0 , onPublish=None)
32     print ("Published data Successfully: %s",myData)
33     time.sleep (2)
34     client.commandcallback =myCommandCallback
35 client.disconnect ()
length: 1,071 lines: 35 Ln: 27 Col: 32 Pos: 691 Windows (CR LF) UTF-8 INS
Type here to search 29°C Cloudy 12:48 PM 22/11/15

```

## 8.2 USER ACCEPTANCE TESTING

User acceptance testing (UAT), also called application testing *or* end-user testing, is a phase of software development in which the software is tested in the real world by its intended audience. UAT is often the last phase of the software testing process and is performed before the tested software is released to its intended market. The goal of UAT is to ensure software can handle real-world tasks and perform up to development specifications. In UAT, users are given the opportunity

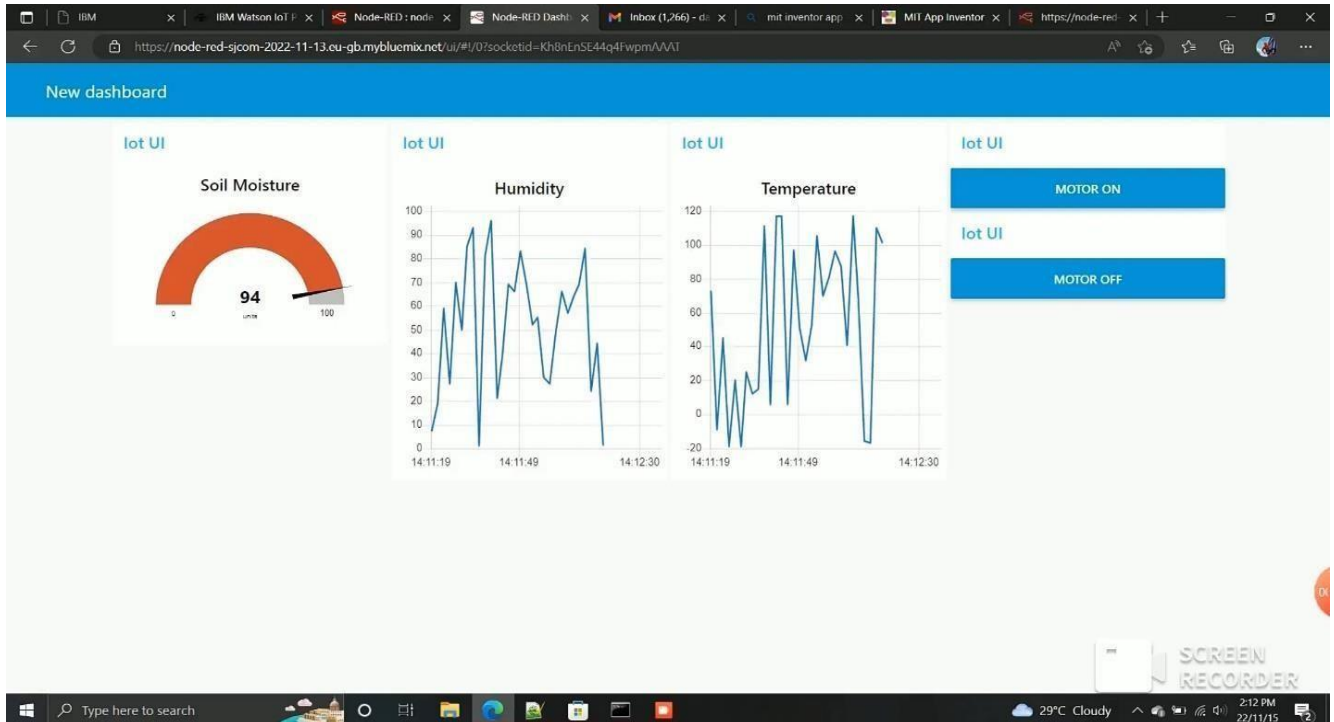


to interact with the software before its official release to see if any features have been overlooked or if it contains any bugs. UAT can be done in-house with volunteers, by paid test subjects using the software or by making the test version available for download as a free trial. The results from the early testers are forwarded to the developers, who make final changes before releasing the software commercially.

## 9.RESULTS

### 9.1 PERFORMANCE METRICS

You can monitor system performance using metrics in IBM® Cognos® Administration, which allows you to diagnose and fix problems quickly. For example, you may want to know if there are more than 50 items in a queue or if any item has been waiting in a queue for longer than a specified amount of time. You must have the required permissions to access **IBM Cognos Administration Capabilities**. Using metrics, you can assess the status of the system as a whole, along with the status of individual servers, dispatchers, and services. You can view the attributes for each metric score, set the threshold values that are used to calculate metric scores, and reset metrics. You may want to refresh report service connections if a PowerCube has been rebuilt. You can also perform functions such as starting and stopping dispatchers or services **Stopping and starting dispatchers and services**, and unregistering dispatchers **Removing dispatchers from the environment**. You can use log files to analyze long-range performance and usage **Setting up logging**. You can create a metric dump file for troubleshooting purposes.



## **10. ADVANTAGES AND DISADVANTAGES**

### **ADVANTAGES**

- A remote control system can help in working irrigation system valves dependent on schedule. Irrigating remote farm properties can be exceptionally troublesome and labor- intensive. It gets hard to comprehend when the valves were started and whether the ideal measure of water was distributed.
- For situations where a quick reaction is required, manual valve actuation may not be conceivable constantly. Thus, remote observing and control of irrigation systems, generators or wind machines or some other motor-driven hardware become the next logical step.
- Various solutions are available to monitor engine statistics and starting or stopping the engine. When the client chooses to begin or stop the motor, the program transmits a sign to the unit within seconds by means of a mobile phone system.
- Submersible weight sensors or ultrasonic sensors can screen the degree of tanks, lakes, wells and different kinds of fluid stockpiling like fuel and compost. The product figures volume dependent on the tank or lake geometry after some time. It conveys alarms dependent on various conditions.

### **DISADVANTAGES**

- One huge disadvantage of smart farming is that it requires an unlimited or continuous internet connection to be successful. This means that in rural communities, especially in the developing countries where we have mass crop production, it is completely impossible to operate this farming method.

- In places where internet connections are frustratingly slow, smart farming will be an impossibility. As pointed out earlier, smart farming makes use of high techs that require technical skill and precision to make it a success.
- It requires an understanding of robotics and ICT. However, many farmers do not have these skills. Even finding someone with this technical ability is difficult or even expensive to come by, at most. And, this can be a discouraging factor hindering a lot of promising farmers from adopting it.
- The smart agriculture needs availability of internet continuously. Rural part of most of the developing countries do not fulfil this requirement. Moreover internet connection is slower.
- The smart farming based equipment require farmers to understand and learn the use of technology. This is major challenge in adopting smart agriculture farming at large scale across the countries

## **11.CONCLUSION**

Farmers can benefit greatly from an IoT-based smart agriculture system. As a result of the lack of irrigation, agriculture suffers. Climate factors such as humidity, temperature, and moisture can be adjusted dependent on the local environmental variables. This technology also detects animal invasions, which are a major cause of crop loss. This technology aids in the scheduling of irrigation based on present data from the field and records from a climate source. It helps in deciding the farmer to whether to do irrigation or not to do. Continuous internet connectivity is required for continuous monitoring of data from sensors. This also can be overcome by using GSM unit as an alternative of mobile app. By GSM, SMS can be sent to farmers phone.



## **12. FUTURE SCOPE**

- In the current project we have implemented the project that can protect and maintain the the crop. In this project the farmer monitor and control the field remotely. In future we can add or update few more things to this project
- We can create few more models of the same project ,so that the farmer can have information of a entire.
- We can update the this project by using solar power mechanism. So that the power supply from electric poles can be replaced with solar panels. It reduces the power line cost. It will be a one time investment. We can add solar fencing technology to this project.
- We can use GSM technology to this project so that the farmers can get the information directly to his home through SMS. This helps the farmer to get information if there is a internet issues.
- We can add camera feature so that the farmer can monitor his field in real time. This helps in avoiding thefts.

## 13.APPENDIX

### SOURCE CODE

```
Import wiotp.sdk.device
import time
import os import
datetime import
random myConfig
={
    "identity": {
        "orgId": "0hzydu",
        "typeId": "NodeMCU",
        "deviceId": "12345"
    },
    "auth": {
        "token": "12345678"
    }
}

client =
wiotp.sdk.device.DeviceClient(config=myConfig,logHandlers=None)
client.connect ()
def myCommandCallback (cmd) :

    print("Message received from IBM IoT Platform: %s"
    %cmd.data['command']) m=cmd.data['command']
    if (m=="motoron"):
        print("Motor is switchedon")
    elif (m=="motoroff"):

        print ("Motor is switchedOFF")
    print (" ")
while True:

    moist =random.randint (0,100)

    temp=random.randint (-20, 125)

    hum=random.randint (0, 100)
```

```
myData={'moisture':moist,'temperature':temp,'humidity':hum}  
client.publishEvent (eventId="status", msgFormat="json", data=myData, qos=0 ,  
    onPublish=None)  
  
    print ("Published data Successfully:  
    %s",myData) time.sleep (2)  
    client.commandCallback =myCommandCallback  
client.disconnect ()
```

### **GITHUB LINK**

**<https://github.com/IBM-EPBL/IBM-Project-1293-1658383415>**

### **DEMO VIDEO LINK**

**[https://drive.google.com/file/d/1PhOKYFKBQ1M3w-KYf5aoCf2VztyCZo3\\_/view?usp=drivesdk](https://drive.google.com/file/d/1PhOKYFKBQ1M3w-KYf5aoCf2VztyCZo3_/view?usp=drivesdk)**





