**Smart Farmer-IOT Enabled**

**Smart Farming Application**

**A NAALAIYA THIRAN PROJECT REPORT**

**TEAM ID : PNT2022TMID17579**

*Submitted by*

DEEPTI .S (713319EC019)

DHARANIDHARAN.S.V (713319EC022)

KIRUBAKARAN .V (713319EC051)

LATCHYA SHREE.G (713319EC055)

**BACHELOR OF ENGINEERING in Department of Electronics and Communication Engineering**

**SNS COLLEGE OF TECHNOLOGY**

###### (Autonomous)

**Saravanampatti , Coimbatore DT -641 035**

***NOVEMBER 2022***

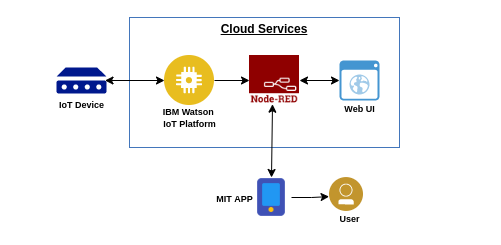
**TABLE OF CONTENTS**

1. **INTRODUCTION** 
   1. Project Overview
   2. Purpose
2. **LITERATURE SURVEY**
   1. Existing problem
   2. References
   3. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
   1. Empathy Map Canvas
   2. Ideation & Brainstorming
   3. Proposed Solution
   4. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
   1. Functional requirement
   2. Non-Functional requirements
5. **PROJECT DESIGN** 
   1. Data Flow Diagrams
   2. Solution & Technical Architecture
   3. User Stories
6. **PROJECT PLANNING & SCHEDULING** 
   1. Sprint Planning & Estimation
   2. Sprint Delivery Schedule
   3. Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)** 
   1. Feature 1
   2. Feature 2
   3. Database Schema (if Applicable)
8. **TESTING** 
   1. Test Cases
   2. User Acceptance Testing
9. **RESULTS** 
   1. Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX** Source Code GitHub & Project Demo Link

##### 1. INTRODUCTION

### 1.1 PROJECT OVERVIEW

IoT-based agriculture system helps the farmer in monitoring different parameters of his field like soil moisture, Temperature, humidity using some sensors. People who use the internet of things can live and work more intelligently and have total control over their life. 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. The system is implemented using an ultrasonic sensor which is connected to Arduino UNO as to monitor the water level. In this system, the depth level will be sent via Arduino Ethernet Shield with an Internet connection to the IBM Cloud.



##### 1.2 PURPOSE

* Need for technology to monitor important parameters like soil moisture, temperature, Humidity etc. to improve the cultivation process
* Need for technology to monitor weather of particular area with reliable source to save the crops at the time of natural calamities like flood, cyclone etc.
* 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.
* Traditional irrigation techniques, such as flood irrigation and overhead sprinkler irrigation, are not very effective.
* They waste a significant amount of water, and the excessive moisture in the soil can also encourage the growth of diseases like fungus.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. IoT is a new platform that is extremely beneficial to people all over the world.
* IoT is at the heart of such revolutionary growth engines. IoT is possible because of adequate power supply and internet connectivity.In large farmland, IoT equipped drone helps to receive the current state of crops and send the live pictures of farmland.

**2. LITERATURE SURVEY**

##### 2.1 EXISTING SYSTEM

1. The biggest challenges faced by IoT in the agricultural sector are lack of information, high adoption costs, and security concerns, etc. Most of the farmers are not aware of the implementation.
2. Getting the weather data from weather station.Transfer of node data to the gateway at faster rate. Unavailability of data’s such as PH level, potassium, Nitrogen etc related to the soil.

**2.2 REFERENCE**

[1] IoT Based Smart Irrigation Monitoring and Controlling System .Date Added to IEEE Xplore: 15January 2018. ISBN Information: Electronic ISBN: 978-1-5090.Date of conference: 19-20 May 2017.INSPEC Accession Number: 17504411 .

[2] International Journal of Innovations in Engineering and Science, Impact Factor Value 4.046 e-ISSN: 2456-3463 Vol.4, No. 5, 2019 “Solar Powered Smart Fencing System for Agriculture Protection using GSM & Wireless Camera”.

[3]International Journal of Management, Technology and Engineering ISSN NO: 2249-7455 Volume 8, Issue VII, JULY/2018”Protecting Crops from Birds, Using Sound Technology In Agriculture” [5] American Journal of Engineering Research (AJER) 2018 eISSN: 2320-0847 pISSN: 2320- 0936 Volume-7, Issue-7, pp-326-330 “Moisture Sensing Automatic Plant Watering System Using Arduino Uno”.

[4]A Systematic Review of IoT Solutions for Smart Farming:

Authors:Emerson Navarro, Nuno Costa, and António Pereira**.**Published: MDPI 2020

##### 2.3 PROBLEM STATEMENT DIFINITION

A strong customer problem statement should provide a detailed description of your customer’s current situation. Consider how they feel, the financial and emotional impact of their current situation, and any other important details about their thoughts or feelings. In upcoming years this demand is likely to increase because of increasing population. To meet this demand, we must adopt new techniques which will conserve need of water for irrigation process. . The system is implemented using an ultrasonic sensor which is connected to Arduino UNO as to monitor Farm Field level. In this system, Farm Field depth level will be sent via Arduino Ethernet Shield with an Internet connection to the IBM IoT Cloud.Creating a customer problem statement is easy with Miro. Using our collaborative online whiteboard, you can create an online problem statement that’s easy to follow and shareable with your team. All you have to do is sign up for free, select this template, and follow your template.

**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 behaviors and attitudes. It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem ant 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.

##### 

##### 3.2 Ideation & Brainstorming

**TEAM IDEAS:**

DEEPTI.S :

* Automate irrigation process using temperature of soil.
* Automate irrigation using measurement of moisture of soil.

DHARANIDHARAN.S.V :

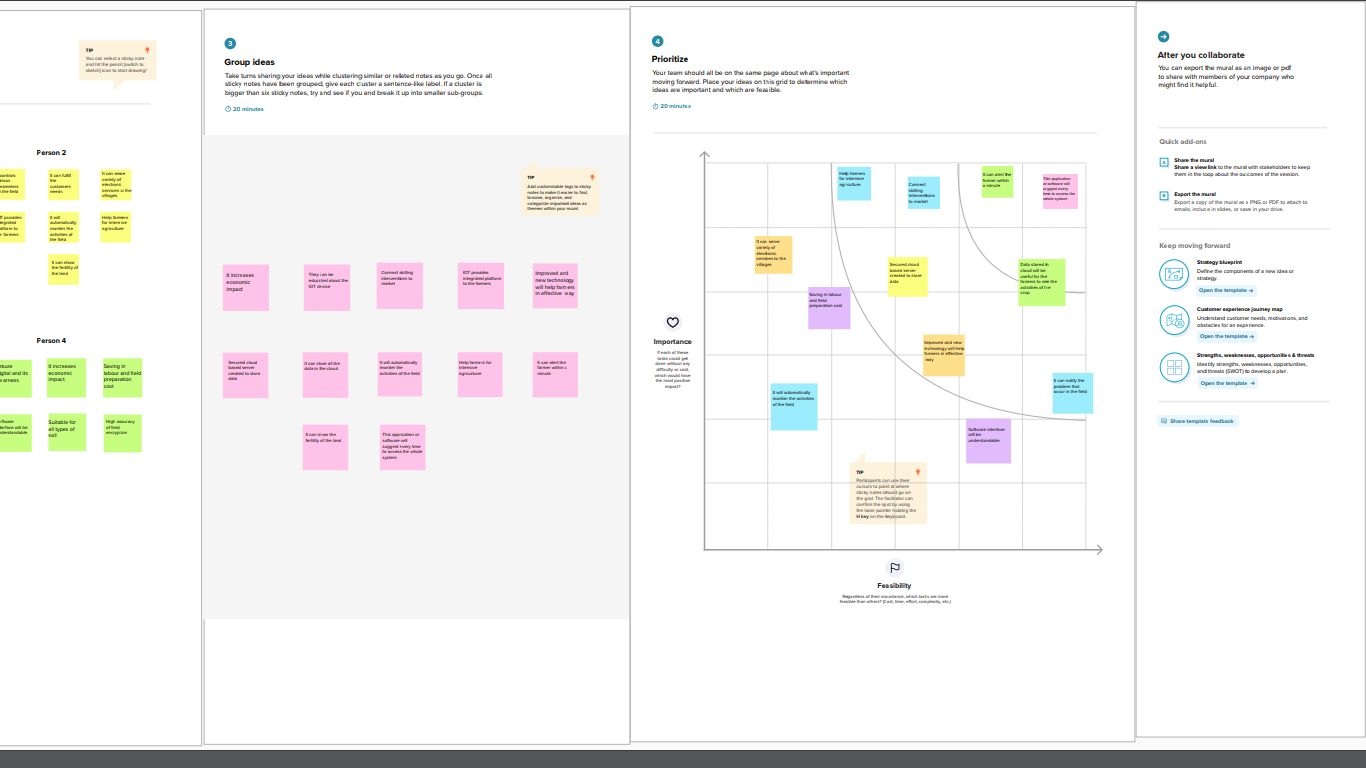
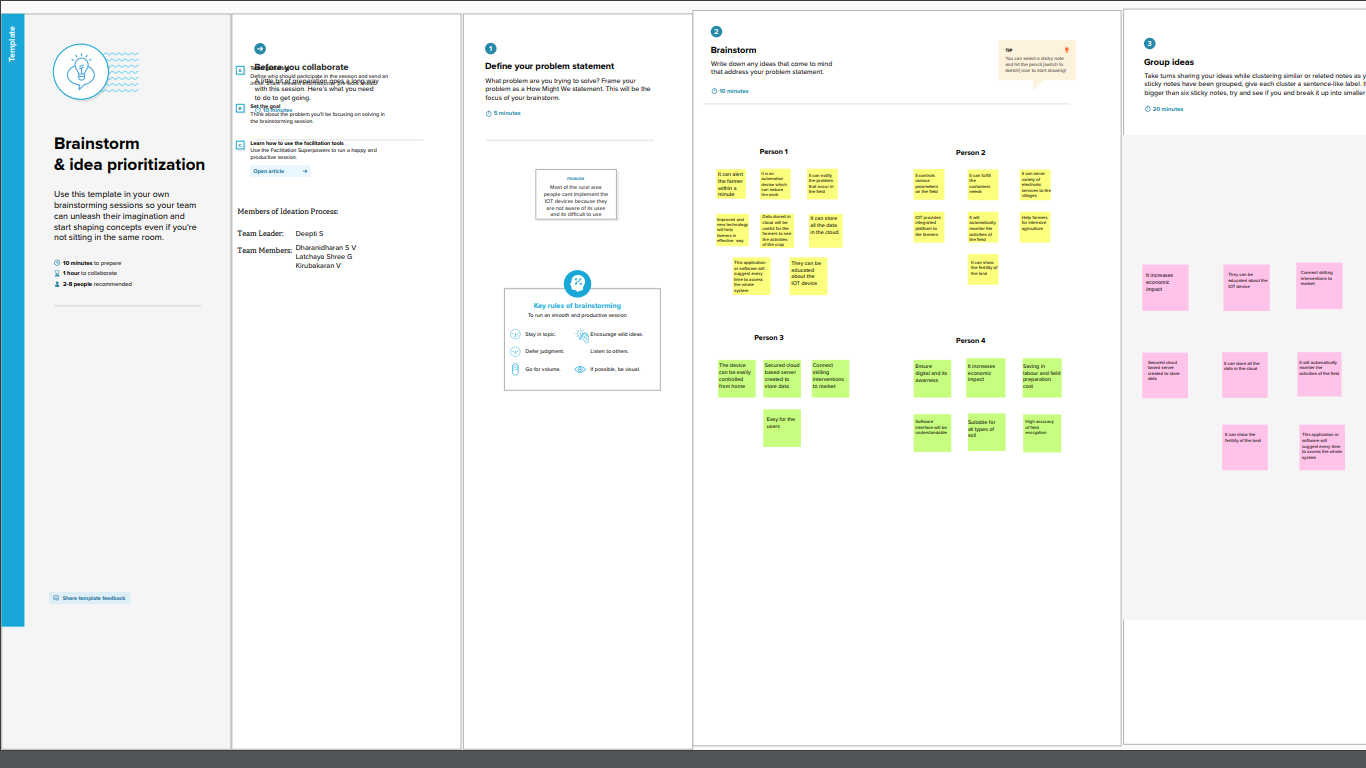
* We can use sensors on sensing.
* We can sense and program the moisture level.

KIRUBAKARAN.V:

* We can simplify the drip irrigation into time controlled irrigation.
* Automate irrigation using any Robots.

LATCHYA SHREE.G:

* We can automate and design Audino for programming.
* We can make good design and programming of soil moisture and temperature.



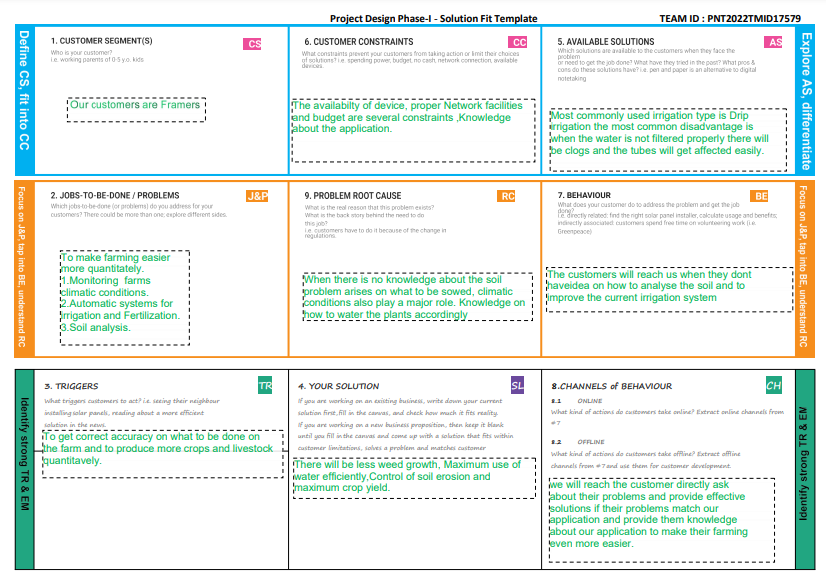
Best Three Ideas:-

* Automate irrigation using measurement of moisture of soil.
* We can sense and program the moisture level.
* We can automate and design Audino for programming.

##### 3.3 Proposed Solution

|  |  |  |
| --- | --- | --- |
| **S.No**  **.** | **Parameter** | **Description** |
| 1. | Problem Statement  (Problem to be solved) | To monitor the water level of crops during irrigation. |
| 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 | A mobile application is developed for monitoring and controlling the irrigation. |
| 4. | Social Impact / Customer Satisfaction | * Reduces the wages for labours 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. |
| 5. | Business Model (Revenue Model) | One time subscription. |
| 6. | Scalability of the Solution | The design scale of solution has been planned in a compact manner |

**3.4 Problem Solution fit**



**4.REQUIREMENT ANALYSIS**

### 4.1 Functional requirement

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional**  **Requirement (Epic)** | **Sub Requirement (Story / Sub-**  **Task)** |
| FR-1 | User Registration | Registration through Gmail |
| FR-2 | User Confirmation | Confirmation via Email & OTP |
| FR-3 | User Location | Share location through the mobile phones |
| FR-4 | User Enter the No. of Acres | Mention the number of acres through Smart farming applications |

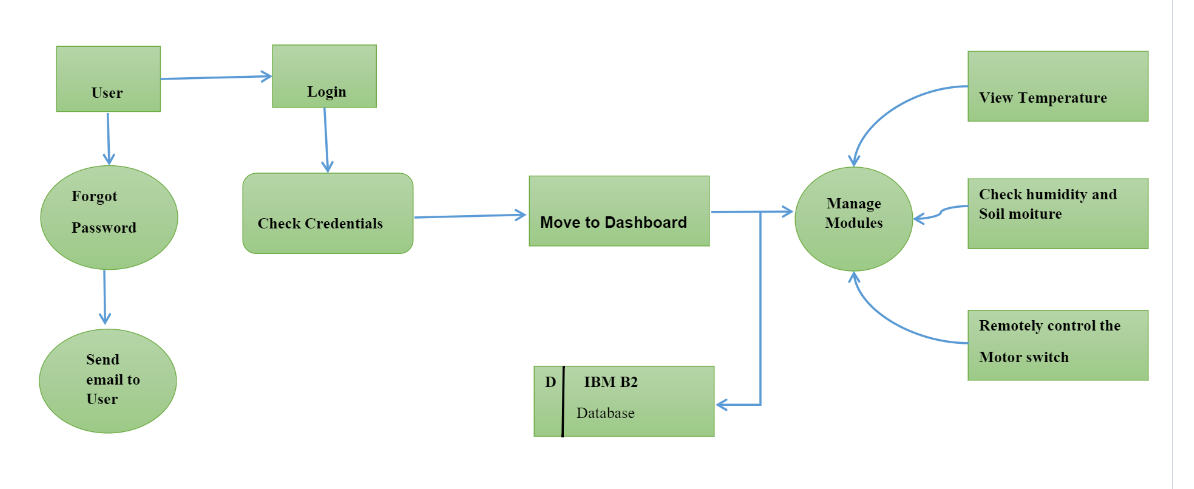
##### 4.2 NON-FUNCTIONAL REQUIREMENT

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

|  |  |  |
| --- | --- | --- |
| **FR.No** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | User Friendly Guidelines are available for the user in the application |
| NFR-2 | **Security** | Assures all data inside the system or its part will be protected against malware attacks or unauthorized access. |
| NFR-3 | **Reliability** | The shared protection achieves a better trade-off between costs and reliability. The model uses dedicated and shared protection schemes to avoid farm service outages. |
| NFR-4 | **Performance** | The Front-Page load time must be no more than 5 seconds for the user based on the network connection. |
| NFR-5 | **Availability** | The proposed product can be available and operable successfully all the time |
| NFR-6 | **Scalability** | The Website traffic limit must be scalable |

**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.

##### 5.2 Solution & Technical Architecture

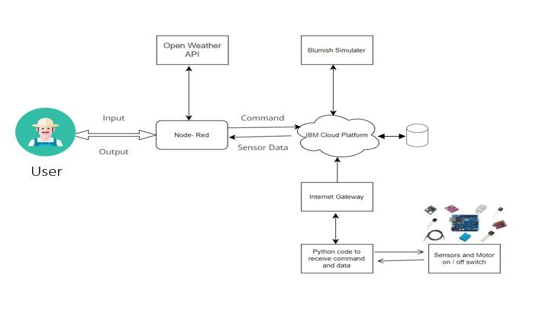
Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to;

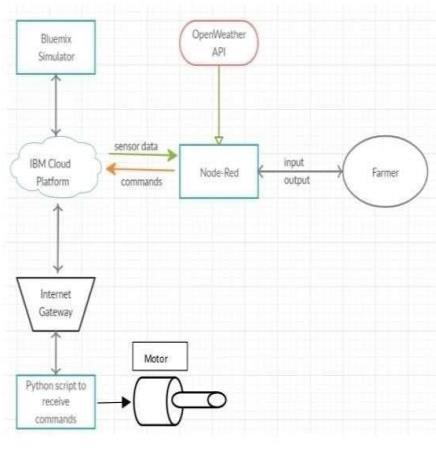
Find the best tech solution to solve existing business problems.

Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.

Define features, development phases, and solution requirements.

* Provide specifications according to which the solution is defined, managed, and delivered.





**Table-1:**

**Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | How user interacts with application e.g. Web UI, Mobile App, Chabot etc. | MIT app |
| 2. | Application Logic-1 | Logic for a process in the application | Node red/IBM Watson/MIT app |
| 3. | Application Logic-2 | Logic for a process in the application | Node red/IBM Watson/MIT app |
| 4. | Application Logic-3 | Logic for a process in the application | Node red/IBM Watson/MIT app |
| 5. | Database | Data Type, Configurations etc. | MySQL, NoSQL, etc. |
| 6. | Cloud Database | Database Service on Cloud | IBM cloud. |
| 7. | Temperature sensor | Monitors the temperature of the crop |  |
| 8. | Humidity sensor | Monitors the humidity |  |
| 9. | Soil moisture sensor  (Torsiometer’s) | Monitors the soil temperature |  |
| 10. | Weather sensor | Monitors the weather | . |
| 11. | Solar panel |  | . |
| 12. | RTC module | Date and time configuration |  |
| 13. | Relay | To get the soil moisture data |  |

**Table-2:**

**Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | MIT app, Node-Red | Software |
| 2. | Scalable Architecture | Drone technology, pesticide monitoring, Mineral identification in soil. | Hardware |

#### 5.3 USER STORIES

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functio nal**  **Require ment (Epic)** | **User**  **Story**  **Number** | **User Story / Task** | **Acceptance criteria** | **Prio**  **rity** | **Releas**  **e** |
| Custom er  (Mobile user) | Registra tion | USN-1 | As a user, I can register for the application by entering my email, password, and | I can access my account / dashboard | High | Sprint1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | confirming my password. |  |  |  |
|  |  | USN-2 | As a user, I will receive confirmatio n email once I have registered for the application | I can receive confirmation email & click confirm | High | Sprint1 |
|  |  | USN-3 | As a user, I can register for the application through Facebook | I can register & access the dashboard with Facebook  Login | Low | Sprint2 |
|  |  | USN-4 | As a user, I can register for the application through Gmail |  | Medi um | Sprint1 |
|  | Login | USN-5 | As a user, I can log into the application by entering email & password |  | High | Sprint1 |
|  | Dashboa  rd |  |  |  |  |  |
| Custo mer (Web  user) |  |  |  |  |  |  |
| Custom  er Care Executi |  |  |  |  |  |  |
| ve |  |  |  |  |  |  |
| Admini  strator |  |  |  |  |  |  |

#### 6. PROJECT PLANNING & SCHEDULING

#### 6.1 Sprint Planning & Estimation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement**  **(Epic)** | **User**  **Story**  **Number** | **User Story / Task** | **Story Points** | **Priority** | **Team** **Members** |
| Sprint1 | Hardware | USN-1 | Sensors and Wi-Fi module with python code. | 2 | High | DEEPTI.S  DHARANIDHARAN.S.V  KIRUBAKARAN.V  LATCHYA SHREE.G |
| Sprint2 | Software | USN-2 | IBM Watson IoT platform, Workflows for IoT scenarios using Node-red | 2 | High | DEEPTI.S  DHARANIDHARAN.S.V  KIRUBAKARAN.V  LATCHYA SHREE.G |
| Sprint3 | MIT app | USN-3 | To develop an mobile  application using MIT | 2 | High | DEEPTI.S  DHARANIDHARAN.S.V  KIRUBAKARAN.V  LATCHYA SHREE.G |
| Sprint4 | Web UI | USN-4 | To make the user to  interact with software. | 2 | High | DEEPTI.S  DHARANIDHARAN.S.V  KIRUBAKARAN.V  LATCHYA SHREE.G |

#### 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 | 29 Oct 2022 |
| Sprint-2 | 20 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 20 | 05 Nov 2022 |
| Sprint-3 | 20 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 20 | 12 Nov 2022 |
| Sprint-4 | 20 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 20 | 19 Nov 2022 |

**7. CODING & SOLUTIONING**

## **7.1 Feature 1**

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 ()

#### 7.1 Feature 2

/\*

Plant Watering System

The circuit:

* Water pump

Power supply: 4.5~12V DC Interface: Brown +; Blue - - Temperature/moisture sensor Power supply: 3.3-5v

* Moisture sensor Power supply: 3.3-5v

\*/

#include "DHT.h"

#define DHTPIN 2 // what digital pin we're connected to #define DHTTYPE

DHT22 // DHT 22 (AM2302), AM2321

DHT dht(DHTPIN, DHTTYPE);

const int SOIL\_MOISTURE\_SENSOR\_PIN = A0; const int WATER\_PUMP\_PIN = 4;

const int dry = 520; const int wet = 270; const int moistureLevels = (dry - wet) / 3;

// TODO: Should we have a counter so if it waters for X times, then take a break?

// OPTIMIZE: how dry to start watering and for how long. const int soilMoistureSartWatering = 400; const int soilMoistureStopWatering = 300;

// 60 seconds const long waterDuration = 1000L \* 60L;

// 60 seconds const long sensorReadIntervals = 1000L \* 60L;

// 2 hr

const long waterIntervals = 1000L \* 60L \* 60L \* 2; long lastWaterTime = waterIntervals - 1; boolean isWatering = false;

void setup()

{ Serial.begin(9600); pinMode(WATER\_PUMP\_PIN, OUTPUT); waterPumpOff(); dht.begin();

}

void loop()

{ mainLoop ();

}

void mainLoop() { float temperature = getTemperature(); float humidity = getHumidity(); long soilMoisture = analogRead(SOIL\_MOISTURE\_SENSOR\_PIN);

Serial.println("Soil Moisture: " + readableSoilMoisture(soilMoisture) + ", " + soilMoisture);

Serial.println("Temperature: " + String(temperature) + "

\*F");Serial.println("Humidity: " + String(humidity) + " %");

if (millis() - lastWaterTime > waterIntervals)

{waterPlants(soilMoisture); lastWaterTime = millis();

}

delay(sensorReadIntervals);

}

void waterPlants(int soilMoisture) {

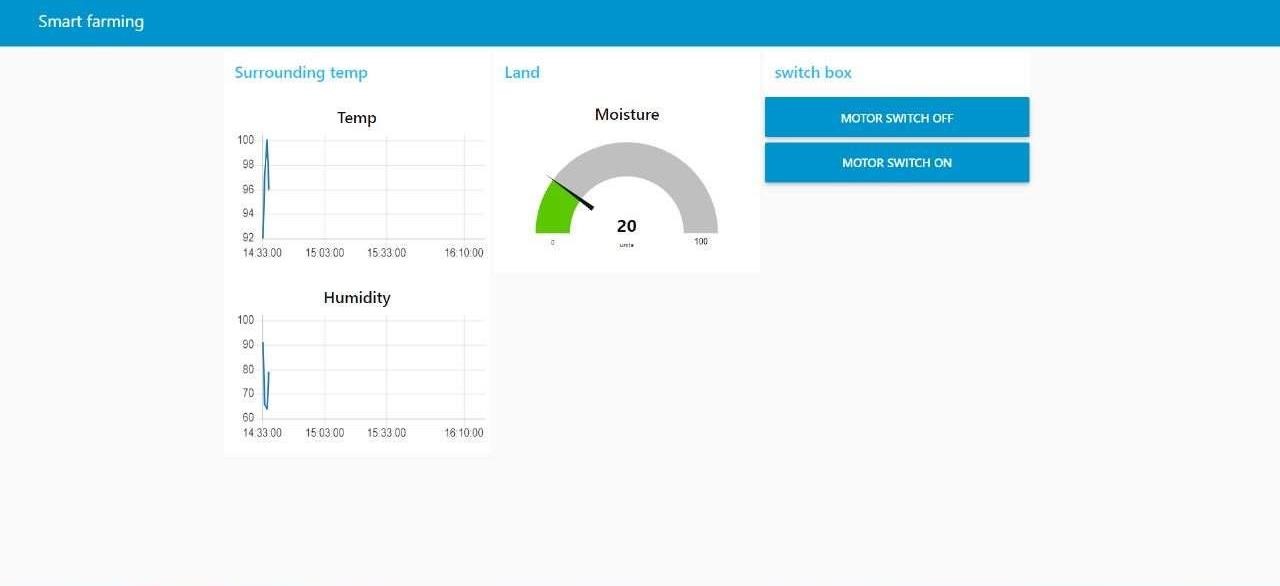
// Should this take a moving avg of the soilMoisture?

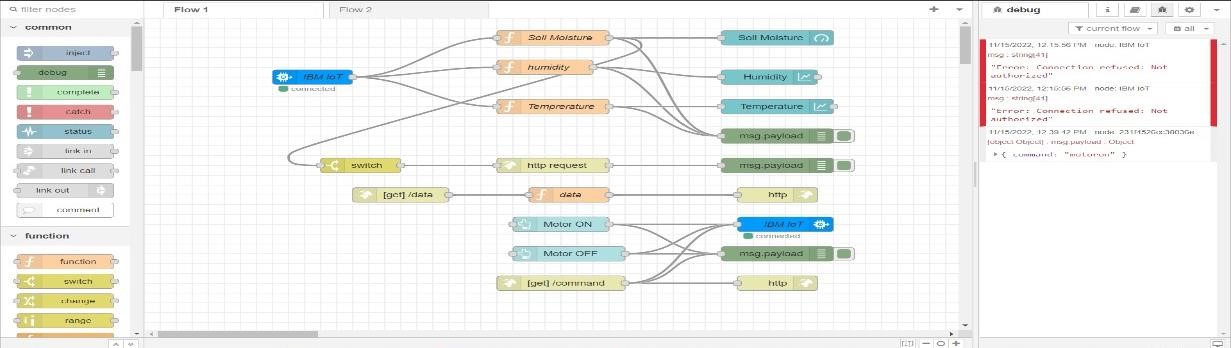
// Can get outliers on the right after watering. if (soilMoisture > soilMoistureSartWatering)

{ isWatering = true

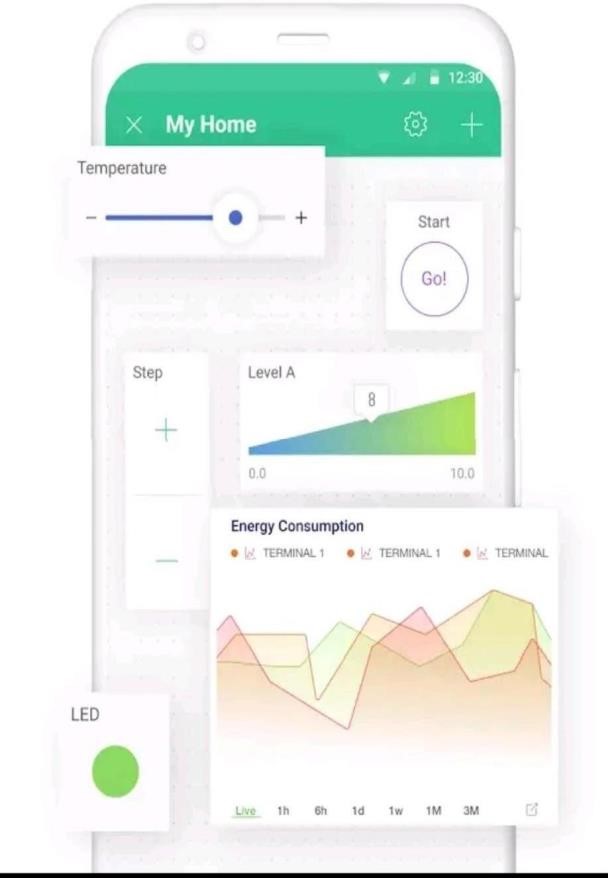
**8. TESTING**

## **8.1 Test Cases**



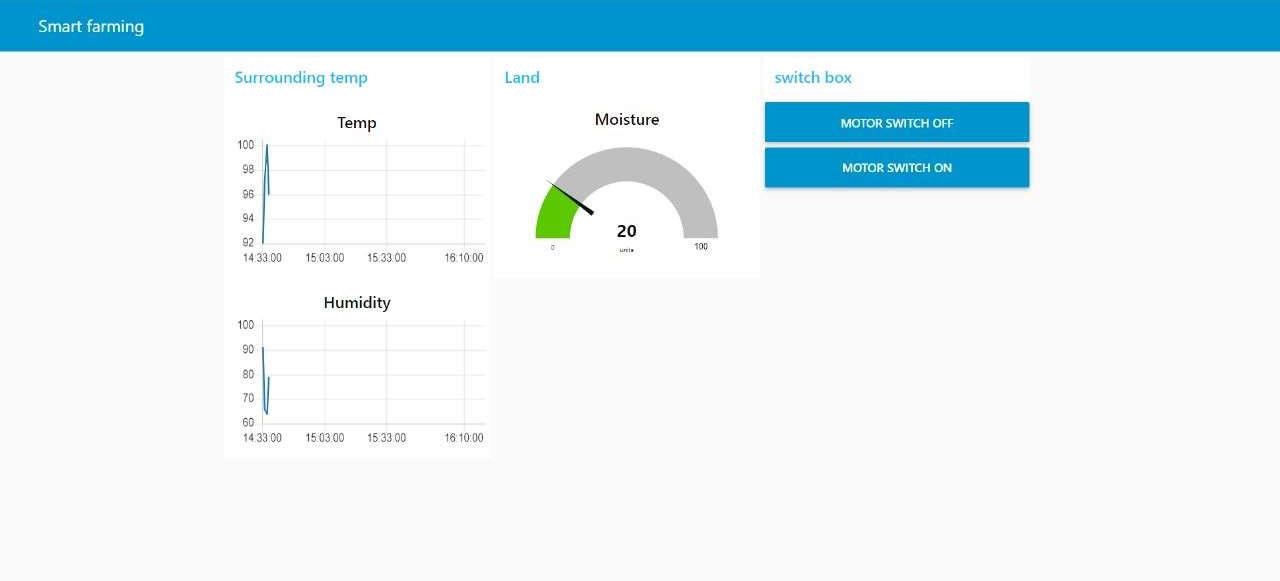


#### 8.2 User Acceptance Testing



**9. RESULTS**

#### 9.1 PERFORMANCE METRICS



**10. ADVANTAGES & 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 motordriven 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.

**DISADVANTAGES**:

* 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.

* When the network connectivity is poor the performance of the application will be affected.
* As it is platform dependent it cannot run on all devices.
* The application will produce inaccurate values when there is a fault or any change in API.
* The user should be more aware on the results produced.

**11. CONCLUSION**

Farmers can benefit greatly from an IoT-based smart agriculture system. As a result of the lack of Farming 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 Smart farming 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 on farmer’s phone.

**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.

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. When the network connectivity is poor the performance of the application will be affected. As it is platform dependent it cannot run on all devices. The application will produce inaccurate values when there is a fault or any change in API. The user should be more aware on the results produced. We can update 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 an 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"

}

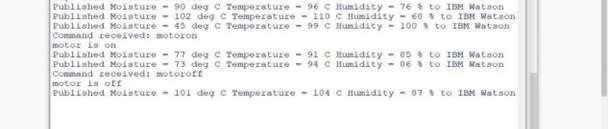
} client = wiotp.sdk.device.DeviceClient(config=myConfig,logHandlers=None) client.connect () def myCommandCallback (cmd) :

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 ()

**OUTPUT:**



**GitHub link:** **https:**[**https://github.com/IBM-EPBL/IBM-Project-45827-1660732610**](https://github.com/IBM-EPBL/IBM-Project-45827-1660732610)

**Project Demo link**: https://photos.app.goo.gl/G78qnFzn3oBhEdyZ6