**SMART FARMER-IOT ENABLED SMART FARMING APPLICATION**

**PROJECT REPORT**

**Submitted by**

**KAARUNYA C R (737819ECR055)**

**JHANANI K R (737819ECR054)**

**MANO RANJITH A (737819ECR081)**

**KAJENDRAN D (737819ECR056)**

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

**of**

**BACHELOR OF ENGINEERING**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**VELALAR COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(AUTONOMOUS)**

**THINDAL,ERODE.**

**PROJECT REPORT**

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

1. **TESTING…………………………………………………………….21** 
   1. Test Cases
   2. User Acceptance Testing
2. **RESULTS ……………………………………………………………24**
   1. Performance Metrics
3. **ADVANTAGES & DISADVANTAGES………………………….25**
4. **CONCLUSION …………………………………………………….25**
5. **FUTURE SCOPE ………………………………………………….25**
6. **APPENDIX ………………………………………………………...26**

Source Code And GitHub

Project Demo Link

**1.INRODUCTION**

**1.1 Project Overview**

IOT- Enabled Smart Farming agriculture system helps the farmer in monitoring different parameters of his field like soil moisture, temperature, humidity using some sensors. Farmer can monitor all the sensor parameters by using a web or mobile application even if the farmer is not near his field. Watering the crop is one of the Important task for the farmers. They can make the decision whether to water the crop or postpone it by monitoring the sensor parameters and control the motor pumps from the mobile application itself. All the sensor parameters are stored in the IBM Cloudant DB

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

**Project requirements:**

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

**Project Deliverables**: Application for IoT based Smart Agriculture System

**1.2 Purpose**

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

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

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

1. **LITERATURE SURVEY**

* 1. **Existing Problem**

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

* 1. **References**

1. Divya J., Divya M.,Janani V.”IoT based Smart Soil Monitoring System for Agricultural Production” 2017.
2. H.G.C.R.Laksiri, H.A.C.Dharmagunawardhana, J.V.Wijayakulasooriya ”Design and Optimization of loT Based Smart Irrigation System in Sri Lanka”2019 .
3. Anushree Math, Layak Ali, Pruthviraj U ”Development of Smart Drip Irriga- tion System Using IoT”2018.
4. Dweepayan Mishra1 ,Arzeena Khan2 Rajeev Tiwari3 , Shuchi Upadhay,”Automated Irrigation System-IoT Based Approach”,2018.
5. R. Nageswara Rao, B.Sridhar,”IOT BASED SMART CROP-FIELD MONI- TORING AND AUTOMATION IRRIGATION SYSTEM”. 2018
6. Shweta B. Saraf, Dhanashri H. Gawal,”IoT Based Smart Irrigation Monitoring And Controlling System”.2017
7. Shrihari M, ”A Smart Wireless System to Automate Production of Crops and Stop Intrusion Using Deep Learning” 2020.
8. G. Sushanth1, and S. Sujatha, ”IOT Based Smart Agriculture System”2018.
9. Vaishali S, Suraj S, Vignesh G, Dhivya S and Udhayakumar S, ”Mobile Integrated Smart

Irrigation Management and Monitoring System Using IOT”,2017

**2.3 Problem Statement Definitions**

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

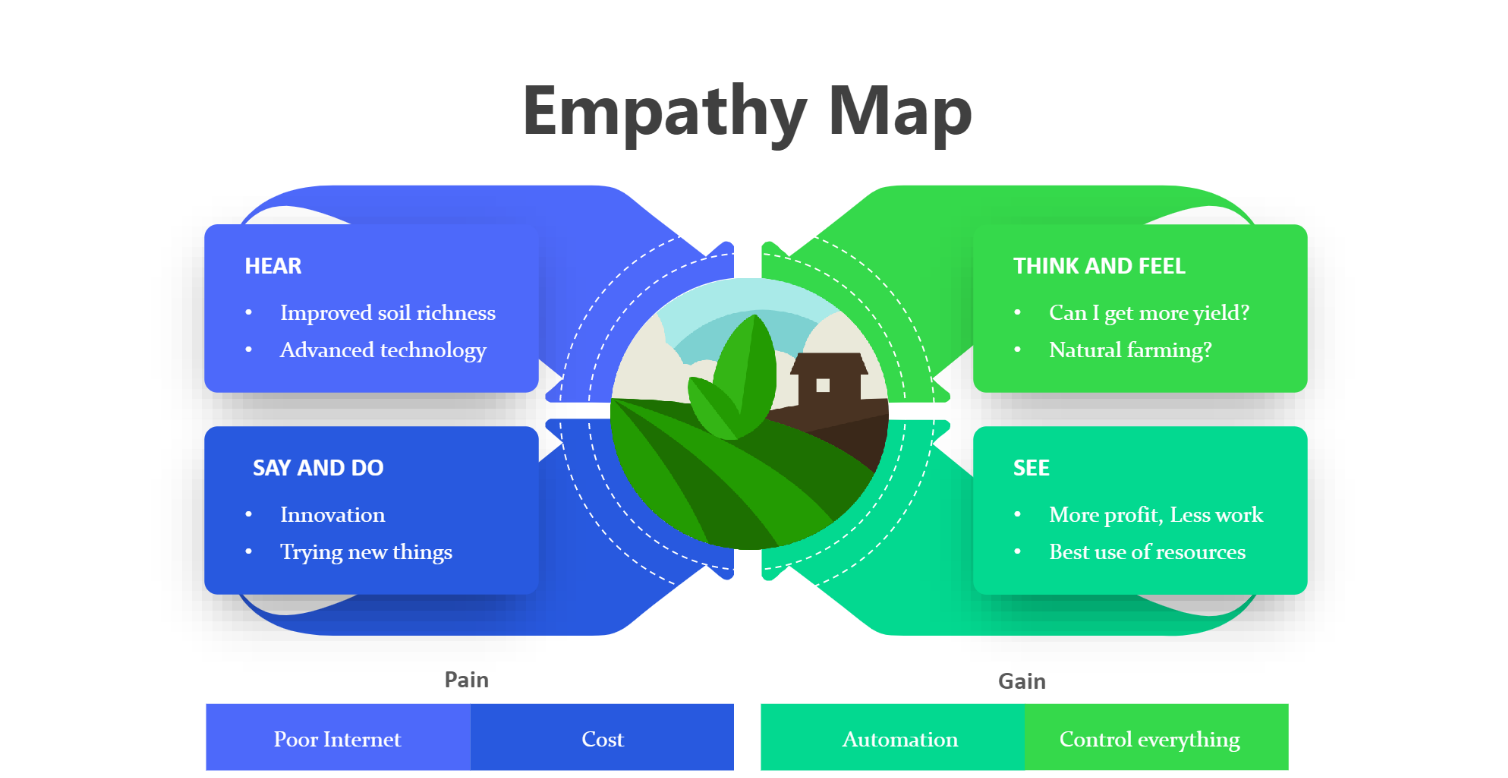
1. **IDEATION & PROPOSED SOLUTION**

* 1. **Empathy Map Canvas:**

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

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

**Empathy Map**

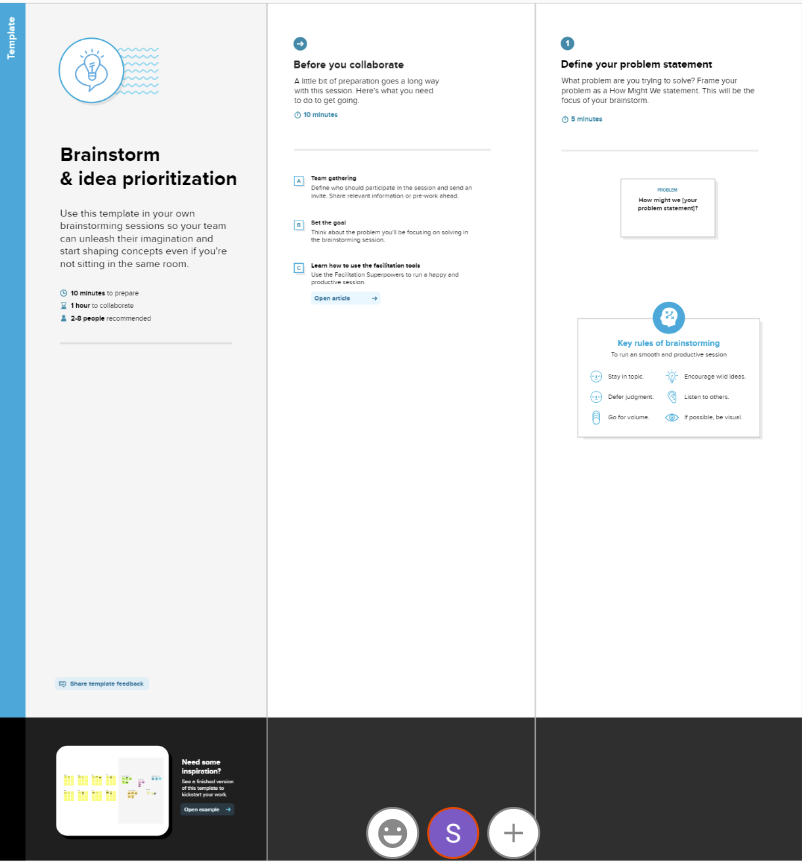


* 1. **Ideation and Brainstorming**

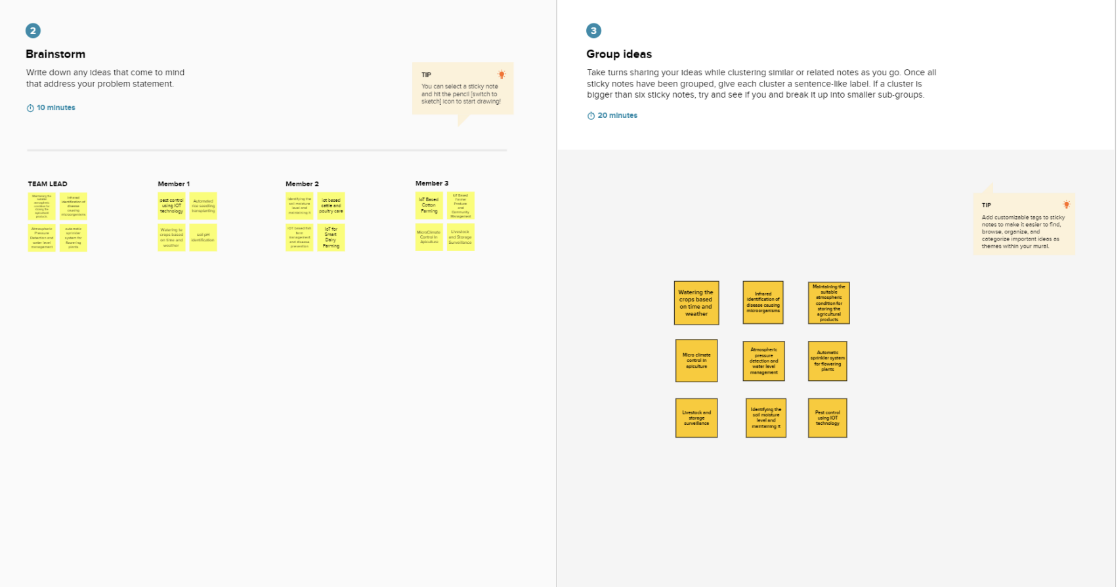
**Reference :**

<https://app.mural.co/t/sivabalan7158/m/sivabalan7158/1666065800436/7b1e33e04dbcf54716d63ecefff45f5a8b8545b5?sender=u373bfdf6cdd38c88e9152869>

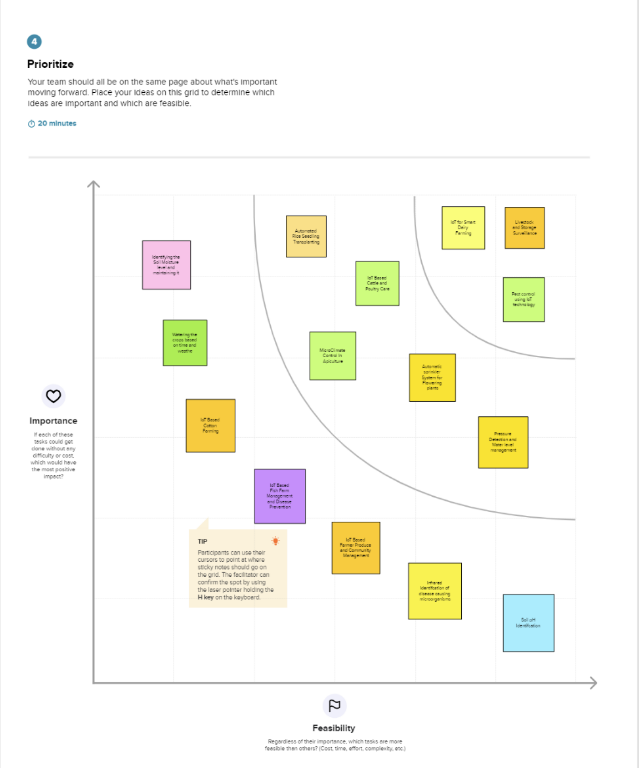
Step-1: Team Gathering, Collaboration and Select the Problem Statements:



Step-2: Brainstorm ,Idea Listing and Grouping



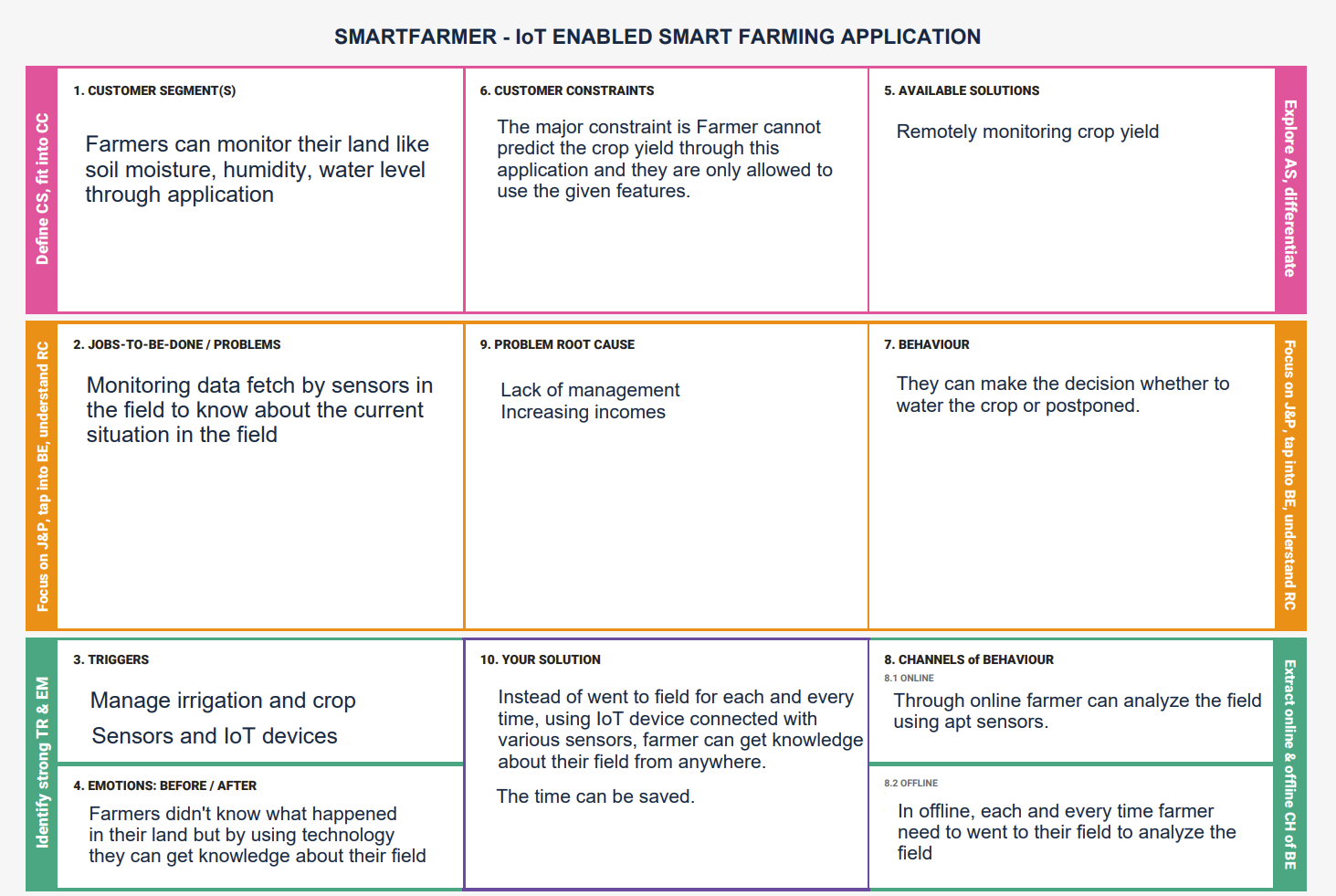
Step-3: Idea Prioritization



* 1. **Proposed Solution:**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to be solved) | * The act of watering a field is challenging; farmers must wait in the field until the entire farm field is submerged in water. * One of the issues is the power supply. Power availability in Village Side may be variable. * The IoT in Agriculture Faces the Following Major Challenges * High Lack of Information ● Security, Cost, and Adoption ● worries, etc. |
| 2. | Idea / Solution description | * As with smart farming and   precision agriculture   * Farmers are better able to keep an eye on their fields and adjust the humidity level as needed thanks to technology. * The information gathered by sensors—which includes information on humidity, temperature, wetness, and dew detections—helps forecast the weather in farms. So, cultivation for suitable crops is carried out. |
| 3. | Novelty / Uniqueness | **ALERT MESSAGE** – IoT sensor nodes  gather data from the agricultural environment, including soil moisture, air humidity, temperature, the nutrients in the soil, pest images, and water quality, and then send the gathered information to IoT backhaul devices. |
| 4. | Social Impact / Customer Satisfaction | * Reduces the pay for workers in the agricultural sector. * It helps you save lots of time. ● By boosting the consumer   experience overall, IoT can help strengthen customer relationships.   * Identify maintenance   requirements quickly, create better products, provide tailored communications, and more.   * IoT may also boost sales and make ecommerce companies successful. It creates a prosperous society. |
| 5. | Business Model (Revenue Model) | Revenue (No. of Users vs Months)    Months |
| 6. | Scalability of the Solution | Scalability in smart farming refers to a system's ability to expand its capacity, such as the number of technological components like sensors and actuators, while allowing for prompt analysis. |

* 1. **Problem Solution fit**

****

1. **REQUIREMENT ANALYSIS** 
   1. **Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR**  **No.** | **Functional Requirement**  **(Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through gmail/LinkedIn Mobile Application Via wifi |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | Authentication | Using biometrics or PIN authentication to carry out some delicate app operations. |
| FR-4 | Cloud Database | Database service on cloud |
| FR-5 | Cloud Connectivity | Connecting Hardware to cloud |

* 1. **Non-Functional Requirements:**

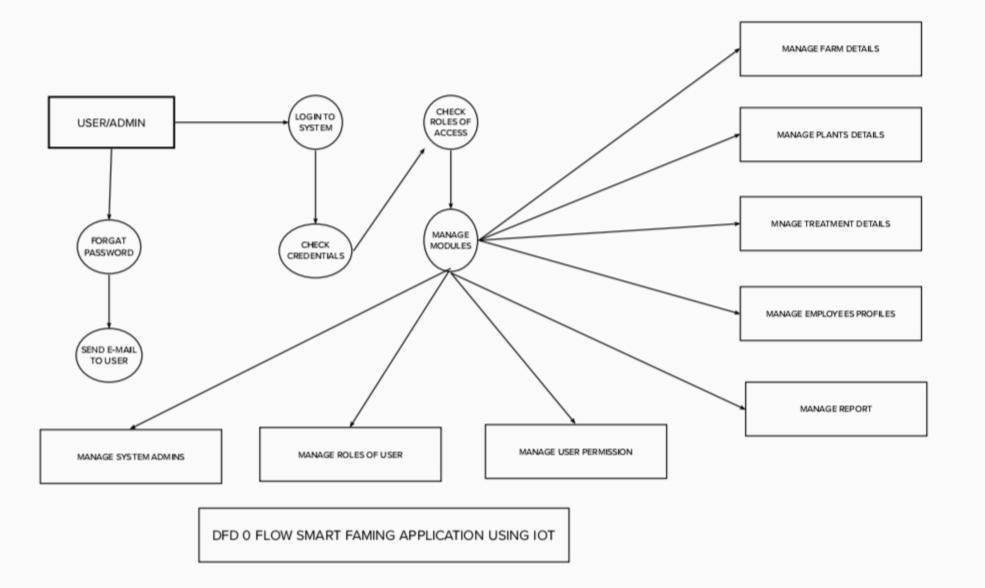
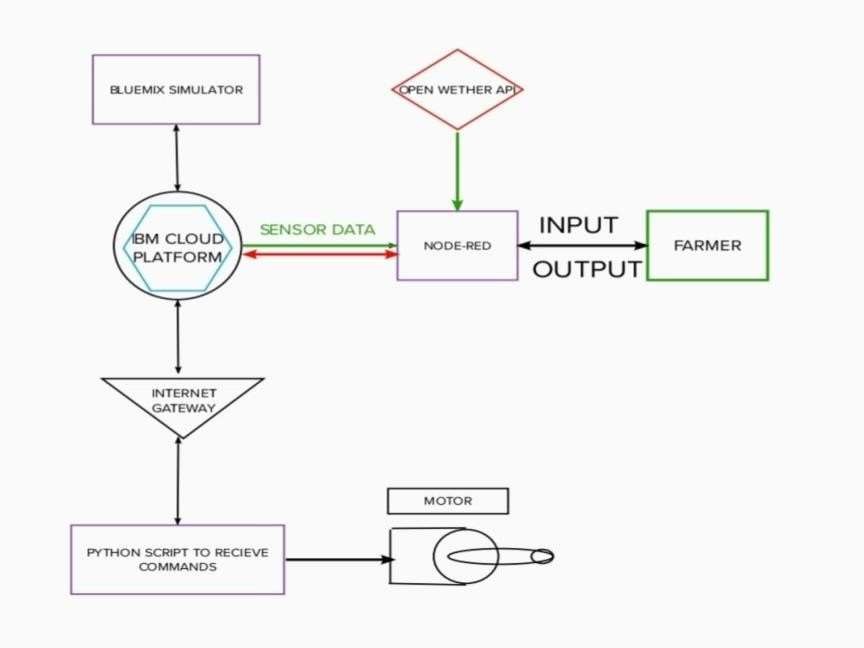
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | The dashboard needs to be clear, uncluttered, and adaptable. |
| NFR-2 | **Security** | Authentication - The user have a private dashboard for secured access. |
| NFR-3 | **Reliability** | High-quality sensors were employed to provide long-lasting high precision and accuracy. |
| NFR-4 | **Performance** | Performance can be enhanced by employing efficient sensors and developing efficient code. |

**5.PROJECT DESIGN**

**5.1 Data Flow Diagrams:**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



* 1. The different soil parameters temperature, soil moistures and then humidity are sensed using different sensors and obtained value is stored in the Ibm cloud.
  2. Arduino UNO is used as a processing Unit that process the data obtained from the sensors and whether data from the weather API.

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

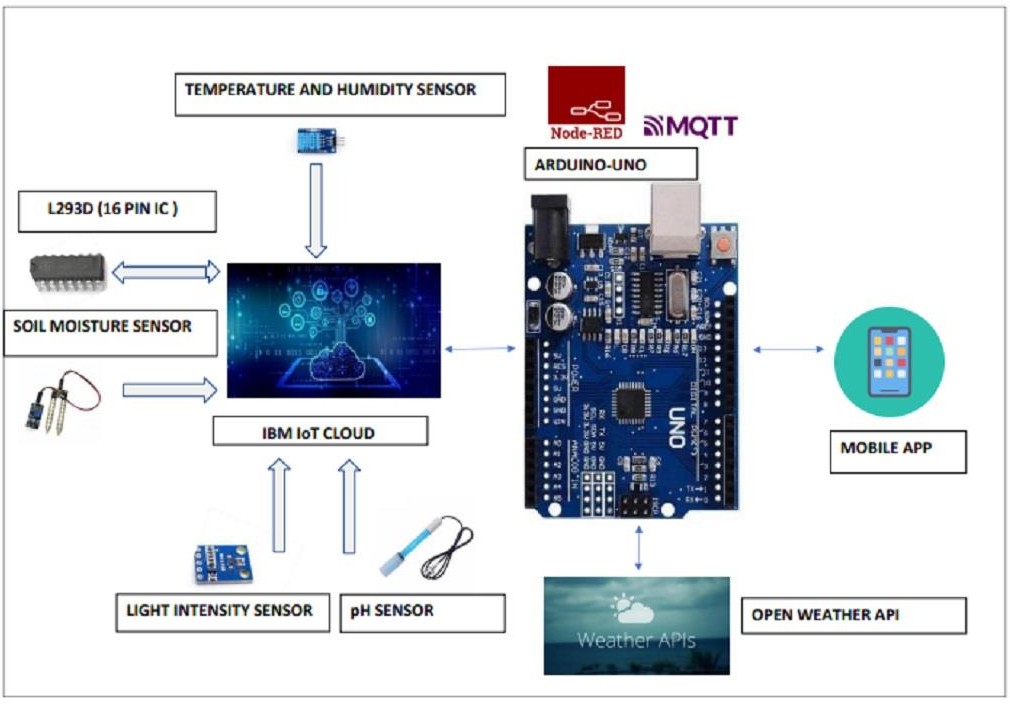
* 1. All the collected data are provided to the user through a mobile application that was developed using the MIT app inventor. The user could make a decision through an app, weather to water the crop or not depending upon the sensor values. By using the app they can remotely operate to the motor switch.

**5.2 Solution & Technical Architecture**

**Solution Architecture:**

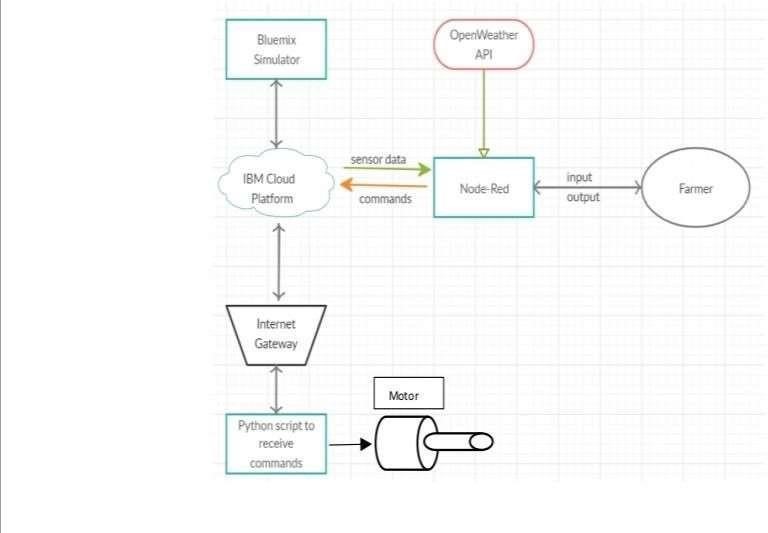
1. Different sensors are used to measure the various soil characteristics (temperature, humidity, and soil moisture), and the results are saved in the IBM cloud.
2. The data from sensors and weather data from weather API are processed using an Arduino UNO as a processing unit.
3. Node-red is employed as a programming tool to connect the APIs, hardware, and software. It uses the MQTT protocol for communication.
4. A mobile application that was created utilising the MIT app inventor gives the user access to all the collected data. Depending on the sensor results, the user might decide via an app whether to irrigate the crop or not. They are able to remotely control the motor switch by utilising the app.

**Solution Architecture Diagram:**



**Technical Architecture**

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



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

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

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

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

**5.3 User Stories**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User Story**  **Number** | **User Story / Task** | **Story Points** | **Priority** | **Team**  **Members** |
| Sprint-1 | Simulation creation | USN-1 | Connect  Sensors Wi-fi Module with python code | 2 | High | KAARUNYA C R  JHANANI K R  MANO RANJITH A  KAJENDRAN D |
| Sprint-2 | Software | USN-2 | Creating device in the IBM Watson IoT platform, workflow for IoT scenarios using NodeRed | 2 | High | KAARUNYA C R  JHANANI K R  MANO RANJITH A  KAJENDRAN D |
| Sprint-3 | MIT App  Inventor | USN-3 | To Develop an application for the Smart  farmer project using MIT App  Inventor | 2 | High | KAARUNYA C R  JHANANI K R  MANO RANJITH A  KAJENDRAN D |
| Sprint-4 | Web UI | USN-5 | To make the user to interact with software. | 2 | High | KAARUNYA C R  JHANANI K R  MANO RANJITH A  KAJENDRAN D |

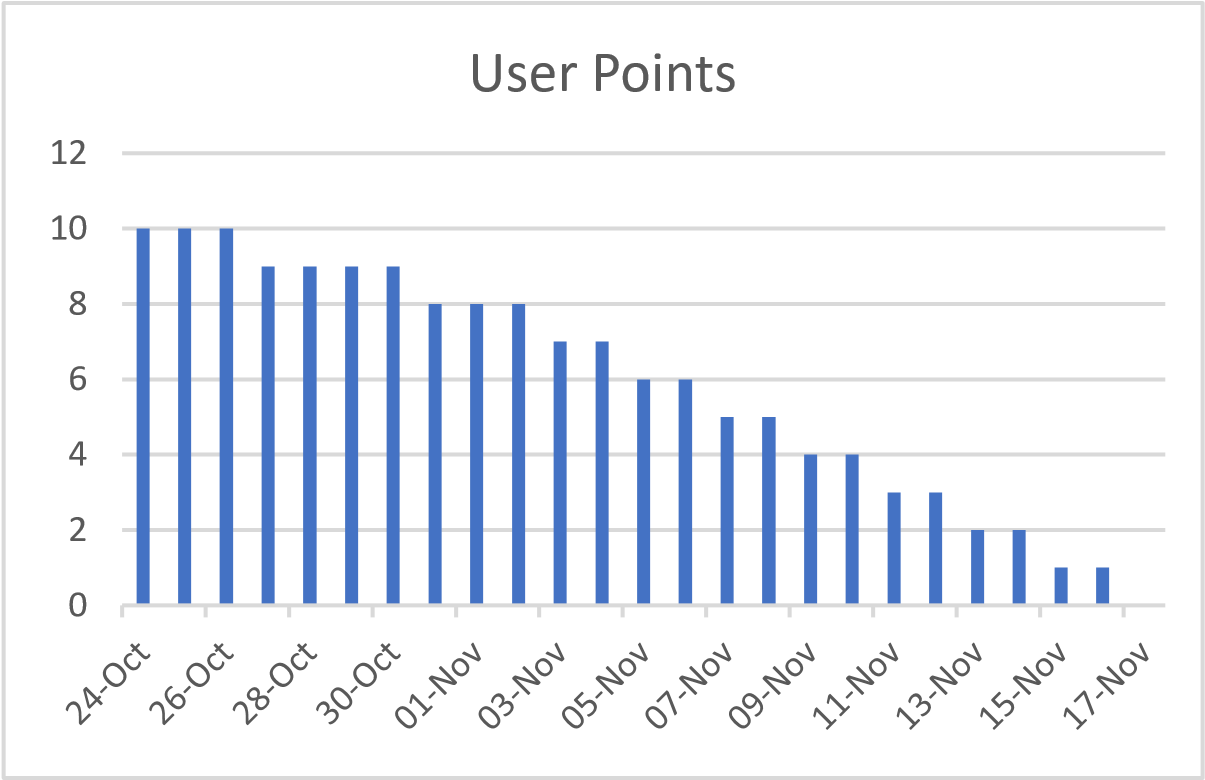
**6.PROJECT PLANNING & SCHEDULING**

**6.1 Sprint planning & Estimation**

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

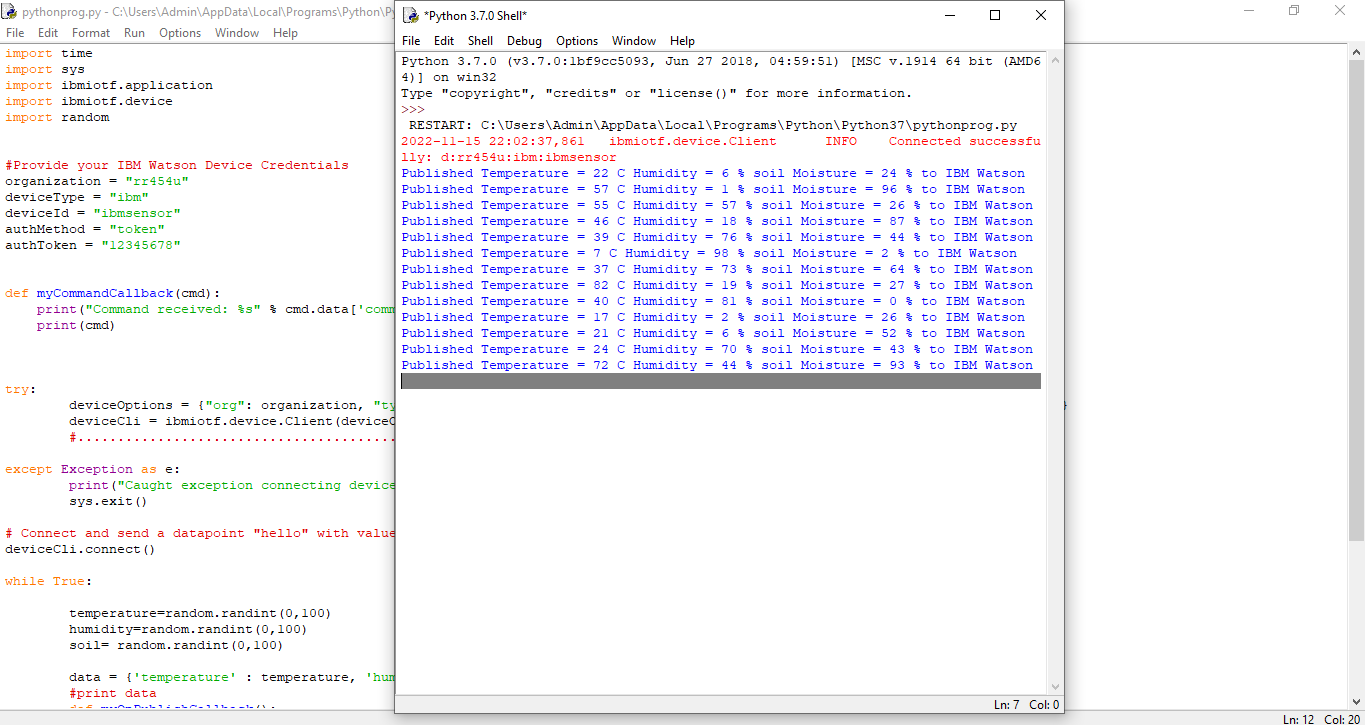
|  |  |
| --- | --- |
| Milestone & Activity List | Prepare the milestones & Activity list of the project. |
| Sprint Delivery | Prepare the Sprint delivery on Number of Sprint planning meetings organized, Minutes of meeting recorded. |
| Project Development Delivery of Sprint- 1,2,3&4 | Develop & submit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

**6.2 Report from JIRA**



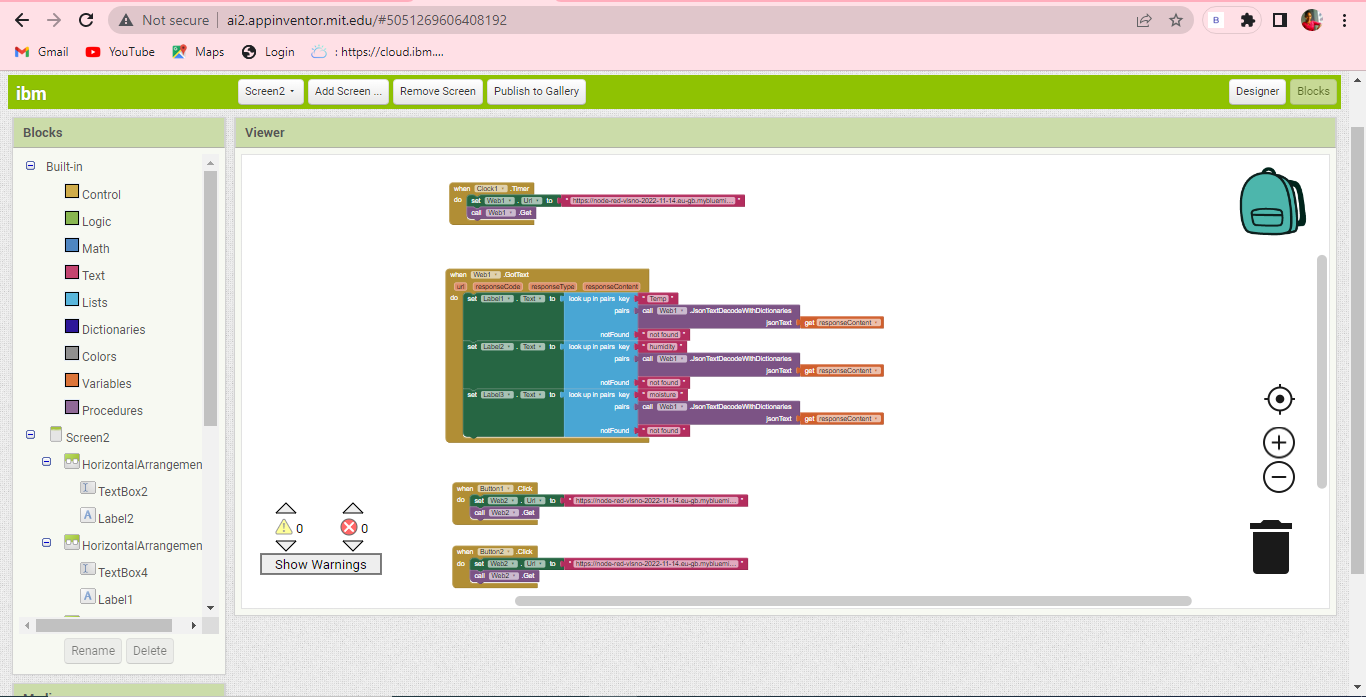
1. **CODING & SOLUTIONING 7.1 Feature 1**





**7.2 Feature 2**

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

****

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

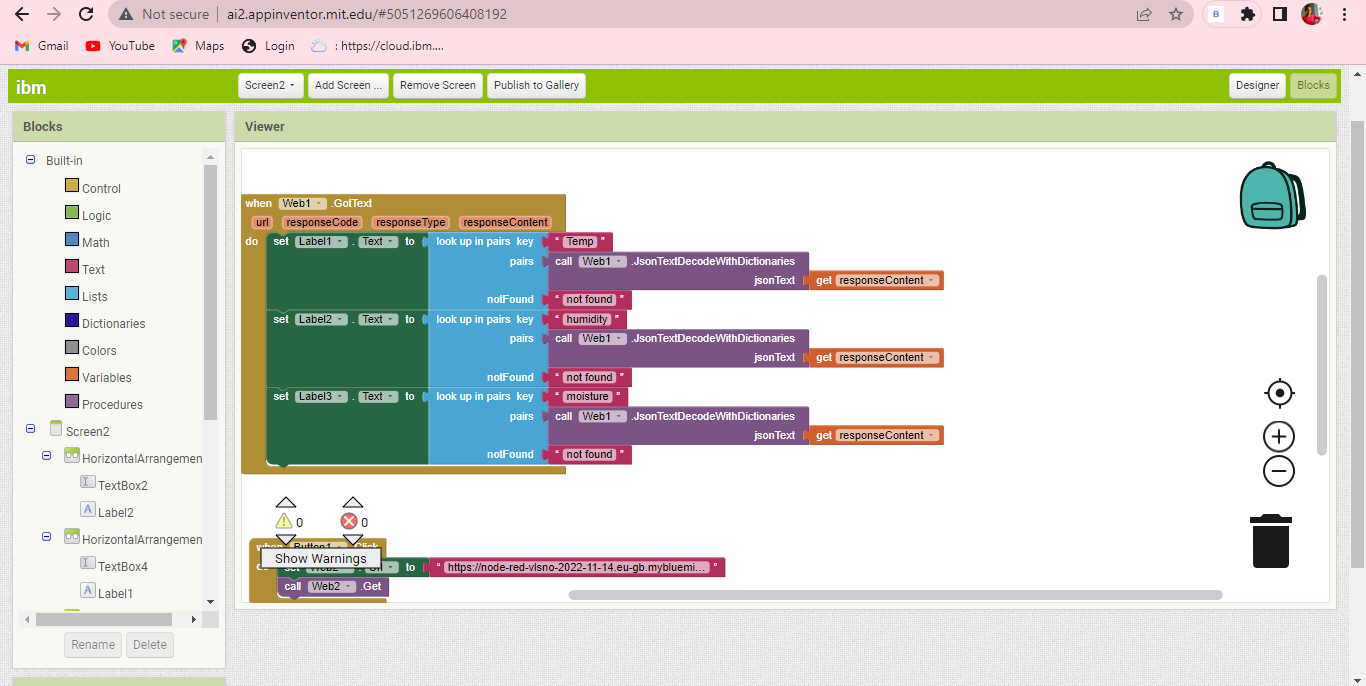
**8.TESTING**

* 1. **Test Cases**

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

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

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

****

**8.2 User Acceptance Testing**

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

we are successfully created the IOT enabled smart farming application.

**9. RESULTS 9.1 Performance Metrics**

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

**10.ADVANTAGES & DISADVANTAGES**

**ADVANTAGES**

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

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

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

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

**DISADVANTAGES:**

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

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

IoT devices need much money to implement.

1. **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

1. **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:**

#include <WiFi.h>//library for wifi

#include <PubSubClient.h>//library for MQtt

#include "DHT.h"// Library for dht11

#define DHTPIN 15     // what pin we're connected to

#define DHTTYPE DHT22   // define type of sensor DHT 11

#define LED 2

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht connected

void callback(char\* subscribetopic, byte\* payload, unsigned int payloadLength);

//-------credentials of IBM Accounts------

#define ORG "rr454u"//IBM ORGANITION ID

#define DEVICE\_TYPE "sensor\_1"//Device type mentioned in ibm watson IOT Platform

#define DEVICE\_ID "sensor"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "12345678"     //Token

String data3;

float h, t;

//-------- Customise the above values --------

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name

char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in which data to be send

char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING

char authMethod[] = "use-token-auth";// authentication method

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;//client id

//-----------------------------------------

WiFiClient wifiClient; // creating the instance for wificlient

PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by passing parameter like server id,portand wificredential

void setup()// configureing the ESP32

{

**Serial**.begin(115200);

  dht.begin();

  pinMode(LED,OUTPUT);

  delay(10);

**Serial**.println();

  wificonnect();

  mqttconnect();

}

void loop()// Recursive Function

{

  h = dht.readHumidity();

  t = dht.readTemperature();

**Serial**.print("temperature:");

**Serial**.println(t);

**Serial**.print("Humidity:");

**Serial**.println(h);

  PublishData(t, h);

  delay(1000);

  if (!client.loop()) {

    mqttconnect();

  }

}

/\*.....................................retrieving to Cloud...............................\*/

void PublishData(float temperature, float humidity) {

  mqttconnect();//function call for connecting to ibm

  /\*

     creating the String in in form JSon to update the data to ibm cloud

  \*/

  String payload = "{\"temperature\":";

  payload += temperature;

  payload += "," "\"Humidity\":";

  payload += humidity;

  payload += "}";

**Serial**.print("Sending payload: ");

**Serial**.println(payload);

  if (client.publish(publishTopic, (char\*) payload.c\_str())) {

**Serial**.println("Publish ok");// if it sucessfully upload data on the cloud then it will print publish ok in Serial monitor or else it will print publish failed

  } else {

**Serial**.println("Publish failed");

  }

}

void mqttconnect() {

  if (!client.connected()) {

**Serial**.print("Reconnecting client to ");

**Serial**.println(server);

    while (!!!client.connect(clientId, authMethod, token)) {

**Serial**.print(".");

      delay(500);

    }

     initManagedDevice();

**Serial**.println();

  }

}

void wificonnect() //function defination for wificonnect

{

**Serial**.println();

**Serial**.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection

  while (WiFi.status() != WL\_CONNECTED) {

    delay(500);

**Serial**.print(".");

  }

**Serial**.println("");

**Serial**.println("WiFi connected");

**Serial**.println("IP address: ");

**Serial**.println(WiFi.localIP());

}

void initManagedDevice() {

  if (client.subscribe(subscribetopic)) {

**Serial**.println((subscribetopic));

**Serial**.println("subscribe to cmd OK");

  } else {

**Serial**.println("subscribe to cmd FAILED");

  }

}

void callback(char\* subscribetopic, byte\* payload, unsigned int payloadLength)

{

**Serial**.print("callback invoked for topic: ");

**Serial**.println(subscribetopic);

  for (int i = 0; i < payloadLength; i++) {

    //Serial.print((char)payload[i]);

    data3 += (char)payload[i];

  }

**Serial**.println("data: "+ data3);

  if(data3=="lighton")

  {

**Serial**.println(data3);

digitalWrite(LED,HIGH);

  }

  else

  {

**Serial**.println(data3);

digitalWrite(LED,LOW);

  }

data3="";

}

**GitHub:**

|  |  |
| --- | --- |
| **Name** | **GitHub (User Name)** |
| Team Leader(KAARUNYA C R) | KaarunyaCR |
| Team Member(JHANANI K R) | Jhanani-KR |
| Team Member(MANO RANJITH A) | 19ECR081 |
| Team Member(KAJENDRAN D) | kajendran-D |

**GitHub Link:**

https://github.com/IBM-EPBL/IBM-Project-36055-1660292223