# INTRODUCTION

* 1. Project Overview
  2. Purpose

# LITERATURE SURVEY

* 1. Existing problem
  2. References
  3. Problem Statement Definition

# IDEATION & PROPOSED SOLUTION

* 1. Empathy Map Canvas
  2. Ideation & Brainstorming
  3. Proposed Solution
  4. Problem Solution Fit

# REQUIREMENT ANALYSIS

* 1. Functional requirement
  2. Non-Functional requirement

# PROJECT DESIGN

* 1. Data Flow Diagrams
  2. Solution & Technical Architecture
  3. User Stories

# PROJECT PLANNING & SCHEDULING

* 1. Sprint Planning & Estimation
  2. Sprint Delivery Schedule

# CODING & SOLUTIONING

* 1. Temperature Sensor
  2. IBM Watson IoT Platform

# TESTING

* 1. Test Cases

# RESULTS

* 1. Performance Metrics

# ADVANTAGES & DISADVANTAGES

1. **CONCLUSION**

# FUTURE SCOPE

1. **APPENDIX**

SourceCode GitHub Link

1. **INTRODUCTION**

# PROJECT OVERVIEW:

Smart farming is about using the new technologies which have arisen at the dawn of the Fourth Industrial Revolution in the areas of agriculture and cattle production to increase production quantity and quality, by making maximum use of resources and minimizing the environmental impact. 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. Smart farming is a management concept focused on providing the agricultural industry with the infrastructure to leverage advanced technology – including big data, the cloud and the internet of things (IoT) – for tracking, monitoring, automating and analyzing operations. Smart farming reduces the ecological footprint of farming. Minimized or site-specific application of inputs, such as fertilizers and pesticides, in precision agriculture systems will mitigate leaching problems as well as the emission of greenhouse gases (6)."Smart farming" is an emerging concept that refers to managing farms using technologies like IoT, robotics, drones and AI to increase the quantity and quality of products while optimizing the human labor required by production. IoT in agriculture is designed to help farmers monitor vital information like humidity, air temperature and soil quality using remote sensors, and to improve yields, plan more efficient irrigation, and make harvest forecasts. One of the largest livelihood providers in India is Agriculture.

Agriculture plays an essential role in support in human life. The rise in population is proportional to the increase in agriculture production. Basically, Agriculture production depends upon the seasonal situations which do not have enough water sources. To get beneficial results in agriculture and to overcome the problems, IoT based smart agriculture system is employed. Global and regional scale agricultural monitoring systems aim to provide up-to-date information regarding food production. In IoT-based smart farming, a system is built for monitoring the crop field with the help of sensors like light, humidity, temperature, soil moisture, etc. The farmers can monitor the field conditions from anywhere. IoT-based smart farming is highly efficient when compared with the conventional approach.

# PURPOSE:

By making farming more connected and intelligent, precision agriculture helps reduce overall costs and improve the quality and quantity of products, the sustainability of agriculture and the experience for the consumer. Increasing control over production leads to better cost management and waste reduction. The ability to trace anomalies in crop growth or livestock health, for instance, helps eliminate the risk of losing yields. Additionally, automation boosts efficiency. With smart devices, multiple processes can be activated at the same time, and automated services enhance product quality and volume by better controlling production processes.

Smart farming systems also enable careful management of the demand forecast and delivery of goods to market just in time to reduce waste. Precision agriculture is focused on managing the supply of land and, based on its condition, concentrating on the right growing parameters – for example, moisture, fertilizer or material content – to provide production for the right crop that is in demand. The types of precision farming systems implemented depend on the use of software for the management of the business. Control systems manage sensor input, delivering remote information for supply and decision support, in addition to the automation of machines and equipment for responding to emerging issues and production support. Smart farming and IoT- driven agriculture are laying the groundwork for a “third green revolution,” which refers to the combined application of information and communications technologies. This includes devices such as precision equipment, IoT sensors and actuators, geo- positioning systems, unmanned aerial vehicles (UAVs) and robots. IoT technology helps better control agricultural processes to reduce production risks and enhances the ability to foresee production results, which helps farmers better plan and distribute product. Data about exact batches of crops and the quantity of crops to harvest can help farmers cut down on labor and waste, for example. Additionally, in a number of sectors.

Due to the recent advances in sensors for the smart farming systems for agriculture and the evolution of WSN and IoT technologies, these can be applied in the development of automatic irrigation systems. The system will determine the parameters that are monitored in irrigation systems regarding water quantity and quality, soil characteristics, weather conditions, and fertilizer usage and provide an overview of the most utilized nodes and wireless technologies employed to implement WSN and IoT based smart farming system.

# LITERATURE SURVEY

* 1. **EXISTING PROBLEM**

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 of IoT in agriculture. Even if the farmers adopt IoT technology they won’t be able to take benefit of this technology due to poor communication infrastructure. Farms are located in remote areas and are far from access to the internet. A farmer needs to have access to crop data reliably at any time from any location, **so** connection issues would cause an advanced monitoring system to be useless. Equipment needed to implement IoT in agriculture is expensive. However sensors are the least expensive component, yet outfitting all of the farmers’ fields to be with them would cost more than a thousand dollars. Automated machinery cost more than manually operated machinery as they include cost for farm management software and cloud access to record data. To earn higher profits, it is significant for farmers to invest in these technologies however it would be difficult for them to make the initial investment to set up IoT technology at their farms. Since IoT devices interact with older equipment they have access to the internet connection, there is no guarantee that they would be able to access drone mapping data or sensor readouts by taking benefit of public connection. An enormous amount of data is collected by IoT agricultural systems which is difficult to protect. Someone can have unauthorized access IoT providers database and could steal and manipulate the data.

# REFERENCES:

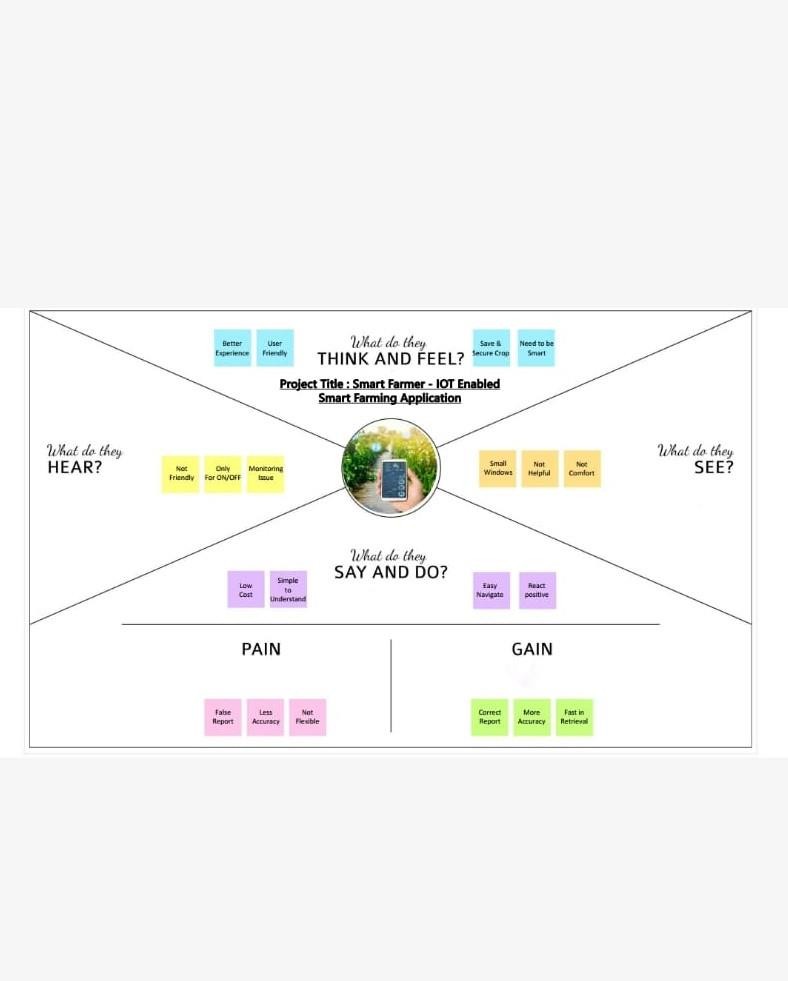
1. Rajalakshmi.P and S. Devi Mahalakshmi, “IOT BasedCrop Field Monitoring and Irrigation Automation”, 10thInternational conference on Intelligent systems andcontrol (ISCO), 2016.
2. Joaquin Gutierrez, Juan Francisco Villa-Medina et.al,“Automated Irrigation System Using a Wireless SensorNetwork and GPRS Module”, IEEE Transactions onInstrumentation and Measurement, 2013.
3. Dr. V. Vidya Devi and G. Meena Kumari, “Real Time Automation and Monitoring System for Modernized Agriculture”, International Journal of Review and Research in Applied Sciences and Engineering, Vol3 no.1. pp 7-12, 2013.
4. Basha, Elizabeth, and Daniela Rus, “Design of early warning flood detection systems for developing countries”, International Conference on Information and Communication Technologies and Development, 2007.
5. K. Jyostsna Vanaja, Aala Suresh et.al, “IOT based Agriculture System Using NodeMCU”, International Research Journal of Engineering and Technology, Vol.05.
6. T. Rajesh, Y. Thrinayana and D. Srinivasulu “IoT based smart agriculture monitoring system”, International Research Journal of Engineering and Technology

# PROBLEM STATEMENT DEFINITION:

Agriculture is the Backbone of Our Country. Traditional methods that are used for irrigation. They results in a lot of wastage of water. About 85% of total available water resources across the world are solely used for the irrigation purpose. 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. In this paper proposed system is based on IoT that uses real time input data. This Water Level Monitoring Irrigation system the excess availability of water in crop is monitored through sensors and reduces the water consumption. This idea is also to focus on parameters such as temperature and soil moisture. The main objective of this project is to control reduce the water supply, save the crops and monitor the plants. 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. The IBM Cloud store the collected Farm field level data into IoT database and display the Farm Field depth level on online dashboard for real-time visualization. The IBM Event manager invoke a notification alert to the Owner of the farmer mobile phone via a SMS when the farm field is nearly filled and It automatically Switch Off the Water Motor. Therefore, the Irrigation became more effective and systematic. To provide efficient decision web using wireless sensor network which handle different activities of farm and provides useful information associated with farm. Information associated with Soil moisture, Temperature and Humidity content. To provide efficient decision web using wireless sensor network which handle different activities of farm and provides useful information associated with farm. Information associated with Soil moisture, Temperature and Humidity content. Due to the atmospheric condition, water level increasing Farmers get lot of distractions which isn't good for Agriculture. Water level is managed by farmers in both Automatic/Manual using that mobile application. it'll make easier to farmers. Performing agriculture is incredibly much time consuming, It should utilize minimum resources in terms of hardware and value. This overcomes the manual operations required to observe and maintain the agricultural farms in both automatic and manual modes. It should be able to measure the rise or decrease in level of water yet as moisture within the soil.

# IDEATION &PROPOSED SOLUTION

* 1. **EMPATHY MAP CANVAS:**



# IDEATION &BRAINSTORMING:

* + 1. **DEFINE PROBLEM STATEMENT:**

It should utilize minimum resources in terms of hardware and value. This overcomes the manual operations required to observe and maintain the agricultural farms in both automatic and manual modes. It should be able to measure the rise or decrease in level of water yet as moisture within the soil.

# BRAINSTORM:

**SURYA**

* + - * Using Agriculture drones
      * Smart Greenhouse
      * Developing an IoT based Sensor Station
      * Livestock monitoring and management

# KOUSHIK

* + - * End to end form management systems
      * Sensors to track the temperature of livestock
      * Analytics for smart farming
      * Cattle monitoring and management

# RAJARAMAN

* + - * Sensor flow weather station predicts rainfall intensity
      * Monitor and control your irrigation system with a mobile app
      * Monitoring of climate conditions
      * Indoor vertical farming

# HARIHARAN

* + - * Precision Agriculture
      * Bee vectoring technologies
      * Farmer’s Hive provide Remote Monitoring sensors

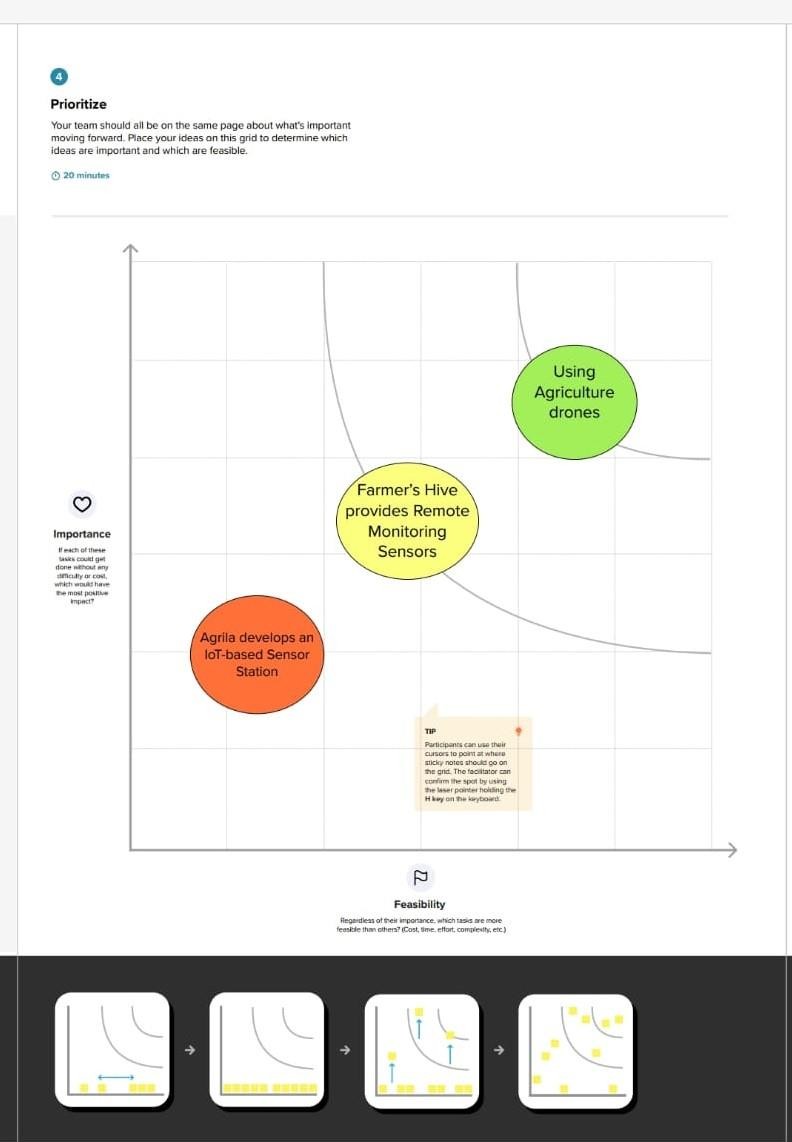
# HARIPRASAD

* + - * Crop Management solutions
      * Using remote monitoring
      * Reducing wastes and increasing productivity

# GROUP IDEAS

* + - * Develops an Iot based sensor Station
      * Farmers Hive provides Remote monitoring sensors
      * Using Agriculture Drones
      * Sensor Flow Weather Station Predicts Rainfall Intensity
      * Sensors to track the temperature of livestock

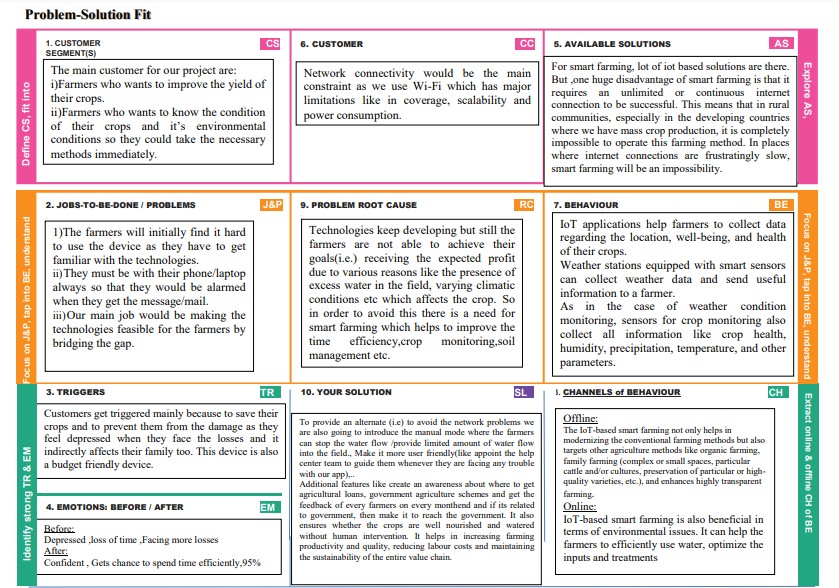
# PRIORITIZE:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **3.3 PROPOSED SOLUTION:** | | | |  |
|  | **S.NO** | **PARAMETER** | **DESCRIPTION** |  |
| 1. | ProblemStatement | * Watering the field is a difficult process, Farmers have to wait in the field until the water covers the whole farm field. * Power Supply is also one of the problems. In Village Side, the power supply may vary. * The Biggest Challenges Faced by IoT in the Agricultural Sector are Lack of Information, High Adoption, Cost and Security Concerns, etc |  |
| **2.** | Idea/Solutiondescription | As is the case of precision Agriculture Smart Farming Technique Enables Farmers better to monitor the fields and maintain the humidity level accordingly.   * The Data collected by sensors, In terms of humidity, temperature, moisture, and dew detections help in determining the weather pattern in Farms. So cultivation is done for   suitable crops.. |  |
| **3.** | Novelty/Uniqueness | ALERT MESSAGE – IoT sensor nodes collect information from the farming environment, such as soil moisture, air humidity, temperature, nutrient ingredients of soil, pest images, and water quality, then transmit collected data to IoT backhaul  devices. REMOTE ACCESS – It helps the farmer to operate the motor from anywhere. |  |
| **4.** | SocialImpact/CustomerSatisfaction | * Reduces the wages for labors who work in the agricultural field. * It saves a lot of time. * IoT can help improve customer relationships by enhancing the customer's overall experience. * Easily identify maintenance needs, build better products, send personalized communications, and more. * IoT can also help e-commerce businesses thrive and increase sales. * It make a wealthy society. |  |
|  | | | |  |

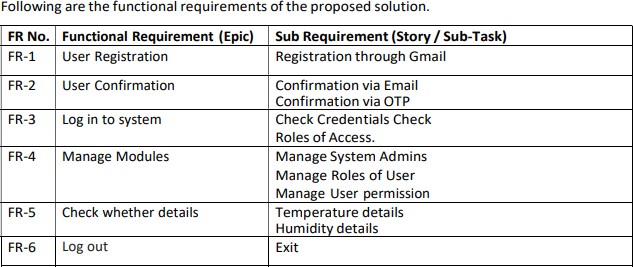
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | |  |
|  | **5.** | BusinessModel(RevenueModel) | Revenue (No. of Users vs Months)  Screenshot (2).png |  |
| **6.** | ScalabilityoftheSolution | Scalability in smart farming refers to the adaptability of a system to increase the capacity, for example, the number of technology devices such as sensors and  actuators, while enabling timely analysis. |  |
|  | | | |  |

**3.4 PROBLEM SOLUTION FIT:**



1. **REQUIREMENT ANALYSIS:**

# FUNCTIONAL REQUIREMENT:

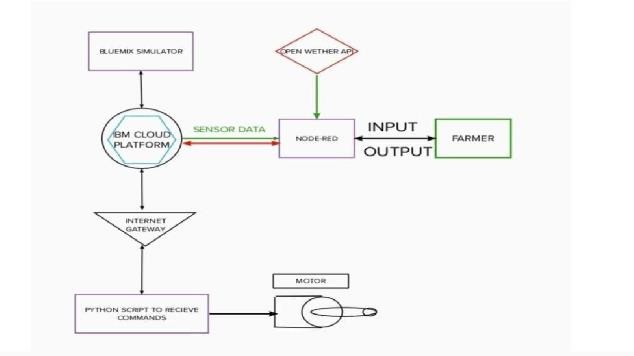


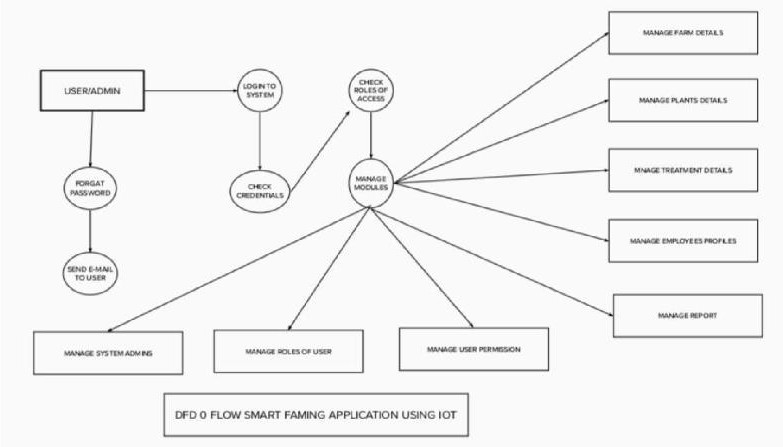
* 1. **NON-FUNCTIONAL REQUIREMENT:**

|  |  |  |
| --- | --- | --- |
| **FR NO** | **NON-FUNCTIONAL REQUIREMENT** | **DESCRIPTION** |
| **NFR-1** | Usability | Usability includes easy learn ability,efficiency in use,remember ability, lack of errors in operation  and subjective pleasure. |
| **NFR-2** | Security | Sensitive and private data must be protected fromtheir production until the decision-making and  storage stages. |
| **NFR-3** | Reliability | The shared protection achieves a better trade-off between costs and reliability.The model uses dedicated and shared protectionschemes to avoid farm  service outages. |
| **NFR-4** | Performance | The idea of implementing integrated sensors with sensing soil and environmental or ambient parameters in farming will be more efficient for overall  monitoring. |
| **NFR-5** | Availability | Automatic adjustment of farming equipment made possible by linking information like crops/weather and equipment to auto-adjust temperature,  humidity, etc. |
| **NFR-6** | Scalability | Scalability is a major concern for IoT platforms. It has shown that different architectural choices of IoT platforms affect system scalability and that automatic real time decision-making is feasible inan environment composed of  dozens of thousand. |

# PROJECT DESIGN:

* 1. **DATA FLOW DIAGRAM:**





# USER STORIES:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **USER TYPE** | **FUNCTIONAL REQUIREMENT** | **USER STORY NUMBER** | **USER STORY,TASK** | **ACCEPTANCE CRITERIA** | **Priority** |
| Customer (Mobile user) and (Web user) | Registration | USN-1 | As a user, I can register my account by entering my email, password, and  confirming my password. | I can access my account /  dashboard | High |
|  |  | USN-2 | As a user, I will receive confirmation email once I have  registered myself | I can receive confirmation email & click confirm | High |
|  |  | USN-3 | As a user, I can register for the application  through apple facebook | I can register & access the dashboard with facebook | low |
|  |  | USN-4 | As a user, I can log into the  application by entering user id &  password |  | Medium |
|  | Login | USN 5 | As a user,I can log into the  application by  entering email and password |  | High |
| Customer Care Executive | Login |  | As I enter I can view the working of the application and scan for any glitchesand monitor the  operation and check if all the users are  authorized. | I can login only with my  provided credentials | High |
| Administrator | Login |  | Maintaining and making sure the databasecontaining the locations are secure and  accurate | I can login only with my  provided credentials | High |

1. **PROJECT PLANNING & SCHEDULING**

# SPRINT PLANNING &ESTIMATION:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SPRINT** | **FUNCTIONAL REQUIREMENT** | **USER**  **STORY NUMBER** | **USER STORY,TASK** | **POINT** | **PRIORITY** |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email, and password, and confirming my  password | 4 | High |
| Sprint-1 | Confirmation Email | USN-2 | As a user, I will receive a  confirmation email once I have registered for the  application | 4 | High |
| Sprint-1 | Authentication | USN-3 | As a user, I can register for the application through  Gmail and mobile app. | 4 | High |
| Sprint-1 | Login | USN-4 | As a user, I can log into the application by entering email  & password | 4 | High |
| Sprint-1 | Dashboard | USN-5 | As a user, I need to be able to view the functions that I can  perform | 4 | High |
| Sprint-2 | Notification | USN-1 | As a user, I should be able to notify my parent and guardian in emergency  situations | 10 | High |
| Sprint-2 | Store data | USN-2 | As a user, I need to continuously store my location data  into the database. | 10 | Medium |
| Sprint-3 | Communication | USN-3,1 | I should be able to  communicate with my parents | 6 | Low |
| Sprint-3 | IoT Device – Watson  communication | USN-1,4 | The data from IoT device should reach  IBM Cloud | 7 | Medium |
| Sprint-3 | Node RED- | USN-5,2 | The data stored in | 7 | High |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Cloudant DB communication |  | IBM Cloud should be properly integrated with  Cloudant DB |  |  |
| Sprint-4 | User – WebUI interface | USN-1,4 | The Web UI should get inputs from the  user | 6 | High |
| Sprint-4 | Geofencing | USN-2,3,5 | The geofencing of the child should be done based on the  geographical coordinates | 7 | High |

* 1. **SPRINT DELIVERY SCHEDULE:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SPRINT** | **TOTAL STORY**  **POINT** | **DURATION** | **SPRINT START**  **DATE** | **SPRINT END DATE** |
| Sprint-1 | 20 | 6 days | 24 Oct 2022 | 29 Oct 2022 |
| Sprint-2 | 20 | 6 days | 05 Nov 2022 | 10 Nov 2022 |
| Sprint-3 | 20 | 9 days | 06 Nov 2022 | 14 Nov 2022 |
| Sprint-4 | 20 | 9 days | 06 Nov 2022 | 14 Nov 2022 |

# CODING & SOLUTIONING:

* 1. **FEATURE-1:**

# TEMPERATURE SENSOR CODING IN ARDUINO:

const int sensor=A5; // Assigning analog pin A5 to variable 'sensor'

float tempc; //variable to store temperature in degree Celsius

float tempf; //variable to store temperature in Fahreinheit float vout; //temporary variable to hold sensor reading void setup() {

pinMode(sensor,INPUT); // Configuring sensor pin as input Serial.begin(9600);

}

void loop() {

vout=analogRead(sensor); //Reading the value from sensor vout=(vout\*500)/1023;

tempc=vout; // Storing value in Degree Celsius tempf=(vout\*1.8)+32; // Converting to Fahrenheit Serial.print("in DegreeC="); Serial.print("\t");

Serial.print(tempc); Serial.print(" "); Serial.print("in Fahrenheit="); Serial.print("\t"); Serial.print(tempf);

Serial.println();

delay(500); //Delay of 1 second for ease of viewing }

**Connect the IBM Watson Iot platform.**

import time import sys

import ibmiotf.application import ibmiotf.device import random

#Provide your IBM Watson Device Credentials organization = "bnsfkk"

deviceType ="Weather\_Monitor" deviceId = "weather" authMethod = "token" authToken = "weatherravi"

# Initialize GPIO

temp=random.randint(0,100) pulse=random.randint(0,100) oxygen= random.randint(0,100) lat = 17

lon = 18

def myCommandCallback(cmd):

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

try:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}

deviceCli = ibmiotf.device.Client(deviceOptions) #..............................................

except Exception as e:

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

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

deviceCli.connect()

while True:

#Get Sensor Data from DHT11

data = {"d":{ 'temp' : temp, 'pulse': pulse ,'oxygen': oxygen,"lat":lat,"lon":lon}} #print data

def myOnPublishCallback():

print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % pulse, "to IBM Watson")

success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF") time.sleep(1)

deviceCli.commandCallback = myCommandCallback

# TESTING

* 1. **TEST CASE: SPRINT-1**

Weathering Monitor-80%

# SPRINT-2

Temperature Monitor-90%

# SPRINT-3

Monitoring the environment condition-80%

# SPRINT-4

Receiving the notification from the cloud-80%

# 9.RESULTS

* 1. **PERFORMANCE METRICS:**

By making farming more connected and intelligent, precision agriculture helps reduce overall costs and improve the quality and quantity of products, the sustainability of agriculture and the experience for the consumer. Increasing control over production leads to better cost management and waste reduction. The ability to trace anomalies in crop growth or livestock health, for instance, helps eliminate the risk of losing yields. Additionally, automation boosts efficiency. With smart devices, multiple processes can be activated at the same time, and automated services enhance product quality and volume by better controlling production processes.

Smart farming systems also enable careful management of the demand forecast and delivery of goods to market just in time to reduce waste. Precision agriculture is focused on managing the supply of land and, based on its condition, concentrating on the right growing parameters – for example, moisture, fertilizer or material content – to provide production for the right crop that is in demand. The types of precision farming systems implemented depend on the use of software for the management of the business. Control systems manage sensor input, delivering remote information for supply and decision support, in addition to the automation of machines and equipment for responding to emerging issues and production support.

# ADVANTAGES:

1. **Increase productivity:** The use of Smart Agriculture helps in producing more and better quality nutritious food and helps in increasing the income and employment rate by 60 percent for those who live in rural areas and are

completely dependent on agriculture for their livelihood.

1. **Reduce harmful emissions:** Smart agriculture helps in reducing the impacts of harmful gases, avoids deforestation, and absorbs carbon dioxide from the atmosphere.
2. **Sensors:** Various sensors involved in the system helps in monitoring the soil moisture, soil temperature, solar radiation, atmospheric pressure, air temperature, air humidity, soil oxygen level, soil water potential, luminosity, etc.
3. More benefits of using smart agriculture are **Fast response, User-friendly, Efficient, Low-cost design, etc.**
4. **DISADVANTAGES:**
   1. These days the topic of the Internet of Things is a trending one. But many are not familiar with this concept.
   2. Internet of things is basically the internet connection between things, people, process, animals, surrounding etc. in a virtual way.
   3. In the concept of the Internet of Things, almost everything in our surroundings will be able to communicate with one another without the help of humans.
   4. This concept will be highly beneficial in various sectors. But like any other technology, this concept has its own challenges.
   5. It also has some issues which have to be tracked properly in order to attain the full benefit of it.
   6. Some disadvantages of using this modern technology in the field of agriculture are listed below.

# CONCLUSION:

IoT based SMART FARMING SYSTEM for Live Monitoring of Temperature and Soil Moisture has been proposed using Arduino and Cloud Computing . The System has high efficiency and accuracy in fetching the live data of temperature and soil moisture. The IoT based smart farming System being proposed via this report will assist farmers

in increasing the agriculture yield and take efficient care of food production as the System will always provide helping hand to farmers for getting accurate live feed of environmental temperature and soil moisture with more than 99% accurate results. IoT will help to enhance smart farming. Using IoT thesystem can predict the soil moisture level and humidity so that the irrigation system can be monitored and controlled.

IoT works in different domains of farming to improve time efficiency, water management, crop monitoring, soil management and control of insecticides and pesticides. This system also minimizes human efforts, simplifies techniques of farming and helps to gain smart farming. Besides the advantages provided by this system, smart farming can also help to grow the market for farmer with single touch and minimum effect.

# FUTURE SCOPE:

Fetch more data especially with regard to Pest Control and by also integrating GPS module in this system to enhance this Agriculture IoT Technology to full-fledged Agriculture Precision ready product. IoT will help to enhance smart farming. Using IoT the system can predict the soil moisture level and humidity so that the irrigation system can be monitored and controlled.

The project has vast scope in developing the system and making it more user friendly and the additional features of the system like:

1. By installing a webcam in the system, photos of the crops can be captured and the data can be sent to database.
2. Speech based option can be implemented in the system for the people who are less literate.
3. GPS (Global Positioning System) can be integrated to provide specific location of the farmer and more accurate weather reports of agriculture field and garden.
4. Regional language feature can be implemented to make it easy for the farmers who are aware of only their regional

# APPENDIX

**SOURCE CODE:**

# FOR REGISTRATION HTML CODE FOR LOGIN PAGE:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>smart.com</title>

<style>

@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display=s wap');

\*{

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: "Poppins", sans-serif;

}

body{

width: 100%;

height: 100vh; display: flex;

align-items: center; justify-content: center; background: #5372F0;

}

::selection{ color: #fff;

background: #5372F0;

}

.wrapper{ width: 380px;

padding: 40px 30px 50px 30px; background: #fff;

border-radius: 5px; text-align: center;

box-shadow: 10px 10px 15px rgba(0,0,0,0.1);

}

.wrapper header{ font-size: 35px;

font-weight: 600;

}

.wrapper form{ margin: 40px 0;

}

form .field{ width: 100%;

margin-bottom: 20px;

}

form .field.shake{

animation: shake 0.3s ease-in-out;

}

@keyframes shake { 0%, 100%{

margin-left: 0px;

}

20%, 80%{

margin-left: -12px;

}

40%, 60%{

margin-left: 12px;

}

}

form .field .input-area{ height: 50px;

width: 100%; position: relative;

}

form input{ width: 100%;

height: 100%; outline: none; padding: 0 45px; font-size: 18px; background: none;

caret-color: #5372F0; border-radius: 5px; border: 1px solid #bfbfbf; border-bottom-width: 2px; transition: all 0.2s ease;

}

form .field input:focus, form .field.valid input{ border-color: #5372F0;

}

form .field.shake input, form .field.error input{ border-color: #dc3545;

}

.field .input-area i{ position: absolute; top: 50%;

font-size: 18px; pointer-events: none;

transform: translateY(-50%);

}

.input-area .icon{ left: 15px; color: #bfbfbf;

transition: color 0.2s ease;

}

.input-area .error-icon{ right: 15px;

color: #dc3545;

}

form input:focus ~ .icon, form .field.valid .icon{ color: #5372F0;

}

form .field.shake input:focus ~ .icon, form .field.error input:focus ~ .icon{ color: #bfbfbf;

}

form input::placeholder{ color: #bfbfbf;

font-size: 17px;

}

form .field .error-txt{ color: #dc3545;

text-align: left;

margin-top: 5px;

}

form .field .error{ display: none;

}

form .field.shake .error, form .field.error .error{ display: block;

}

form .pass-txt{ text-align: left;

margin-top: -10px;

}

.wrapper a{ color: #5372F0;

text-decoration: none;

}

.wrapper a:hover{

text-decoration: underline;

}

form input[type="submit"]{ height: 50px;

margin-top: 30px; color: #fff; padding: 0; border: none;

background: #5372F0; cursor: pointer;

border-bottom: 2px solid rgba(0,0,0,0.1); transition: all 0.3s ease;

}

form input[type="submit"]:hover{ background: #2c52ed;

}

</style>

</head>

<body>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login Form validation in HTML & CSS | CodingNepal</title>

<link rel="stylesheet" href="style.css">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font- awesome/5.15.3/css/all.min.css"/>

</head>

<body>

<div class="wrapper">

<header>Login Form</header>

<form action="#">

<div class="field email">

<div class="input-area">

<input type="text" placeholder="Email Address">

<i class="icon fas fa-envelope"></i>

<i class="error error-icon fas fa-exclamation-circle"></i>

</div>

<div class="error error-txt">Email can't be blank</div>

</div>

<div class="field password">

<div class="input-area">

<input type="password" placeholder="Password">

<i class="icon fas fa-lock"></i>

<i class="error error-icon fas fa-exclamation-circle"></i>

</div>

<div class="error error-txt">Password can't be blank</div>

</div>

<div class="pass-txt"><a href="#">Forgot password?</a></div>

<input type="submit" value="Login">

</form>

<div class="sign-txt">Not yet member? <a href="#">Signup now</a></div>

</div>

<script>

const form = document.querySelector("form"); eField = form.querySelector(".email"),

eInput = eField.querySelector("input"), pField = form.querySelector(".password"), pInput = pField.querySelector("input");

form.onsubmit = (e)=>{

e.preventDefault(); //preventing from form submitting

//if email and password is blank then add shake class in it else call specified function (eInput.value == "") ? eField.classList.add("shake", "error") : checkEmail(); (pInput.value == "") ? pField.classList.add("shake", "error") : checkPass();

setTimeout(()=>{ //remove shake class after 500ms eField.classList.remove("shake"); pField.classList.remove("shake");

}, 500);

eInput.onkeyup = ()=>{checkEmail();} //calling checkEmail function on email input keyup

pInput.onkeyup = ()=>{checkPass();} //calling checkPassword function on pass input keyup

function checkEmail(){ //checkEmail function

let pattern = /^[^ ]+@[^ ]+\.[a-z]{2,3}$/; //pattern for validate email if(!eInput.value.match(pattern)){ //if pattern not matched then add error and remove

valid class

eField.classList.add("error"); eField.classList.remove("valid");

let errorTxt = eField.querySelector(".error-txt");

//if email value is not empty then show please enter valid email else show Email can't be blank

(eInput.value != "") ? errorTxt.innerText = "Enter a valid email address" : errorTxt.innerText = "Email can't be blank";

}else{ //if pattern matched then remove error and add valid class eField.classList.remove("error");

eField.classList.add("valid");

}

}

function checkPass(){ //checkPass function

if(pInput.value == ""){ //if pass is empty then add error and remove valid class pField.classList.add("error");

pField.classList.remove("valid");

}else{ //if pass is empty then remove error and add valid class pField.classList.remove("error"); pField.classList.add("valid");

}

}

//if eField and pField doesn't contains error class that mean user filled details properly if(!eField.classList.contains("error") && !pField.classList.contains("error")){

window.location.href = form.getAttribute("action"); //redirecting user to the specified url which is inside action attribute of form tag

}

}

</script>

</body>

</html>

**Connecting Sensors with wokwi using C++**

#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 "i3869j"//IBM ORGANITION ID

#define DEVICE\_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform #define DEVICE\_ID "1234"//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](http://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("temp:"); **Serial**.println(t); **Serial**.print("Humid:"); **Serial**.println(h);

PublishData(t, h); delay(1000);

if (!client.loop()) { mqttconnect();

}

}

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

void PublishData(float temp, float humid) { mqttconnect();//function call for connecting to ibm

/\*

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

\*/

String payload = "{\"temp\":"; payload += temp;

payload += "," "\"Humid\":"; payload += humid;

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="";

}

# CODING FOR NODEMCU:

#include <ESP8266WiFi.h> #include <ESP8266HTTPClient.h> #include <Adafruit\_ADS1015.h> WiFiClient client;

String thingSpeakAddress= "[http://api.thingspeak.com/update?](http://api.thingspeak.com/update)"; String writeAPIKey;

String tsfield1Name; String request\_string; HTTPClient http; Adafruit\_ADS1115 ads; void setup()

{

Serial.begin(115200);

delay(3000);

WiFi.disconnect();

Serial.println("START");

WiFi.begin("DESKTOP","asdfghjkl"); // Wifi ("ID","Password")

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

delay(300);

Serial.println("...");

}

Serial.println("I AM CONNECTED");

Serial.println("Hello!");

Serial.println("Getting single-ended readings from AIN0..3");

Serial.println("ADC Range: +/- 6.144V (1 bit = 3mV/ADS1015,

0.1875mV/ADS1115)");

ads.begin();

}

void loop()

{

int16\_t adc0, adc1, adc2, adc3;

Serial.println(" ");

adc0 = ads.readADC\_SingleEnded(0);

adc0 = adc0 / 25;

adc1 = ads.readADC\_SingleEnded(1);

adc1 = adc1 / 25;

adc2 = ads.readADC\_SingleEnded(2);

adc2 = adc2 / 25;

adc3 = ads.readADC\_SingleEnded(3);

adc3 = adc3 / 25;

Serial.print("SOIL MOISTURE in persent 1% : ");

Serial.println(adc0);

Serial.print("SOIL MOISTURE in persent 2% : ");

Serial.println(adc1);

Serial.print("SOIL MOISTURE in persent 3% : ");

Serial.println(adc2);

Serial.print("SOIL MOISTURE in persent 4% : ");

Serial.println(adc3);

Serial.println(" ");

if (client.connect("api.thingspeak.com",80))

{

request\_string = thingSpeakAddress;

request\_string += "key=";

request\_string += "2YGO2FHN3XI3GFE7";

request\_string += "&";

request\_string += "field1";

request\_string += "=";

request\_string += adc0;

http.begin(request\_string);

http.GET();

http.end();

}

delay(10);

if (client.connect("api.thingspeak.com",80))

{

request\_string = thingSpeakAddress;

request\_string += "key=";

request\_string += "2YGO2FHN3XI3GFE7";

request\_string += "&";

request\_string += "field2";

request\_string += "=";

request\_string += adc1;

http.begin(request\_string);

http.GET();

http.end();

}

delay(10);

if (client.connect("api.thingspeak.com",80))

{

request\_string = thingSpeakAddress;

request\_string += "key=";

request\_string += "2YGO2FHN3XI3GFE7";

request\_string += "&";

request\_string += "field3";

request\_string += "=";

request\_string += adc2;

http.begin(request\_string);

http.GET();

http.end();

}

delay(10);

if (client.connect("api.thingspeak.com",80))

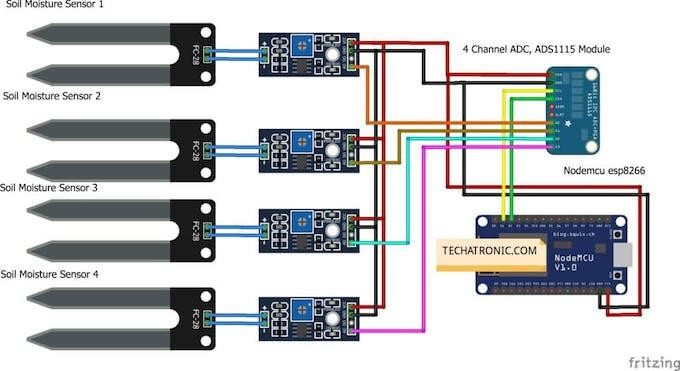
{

request\_string = thingSpeakAddress; request\_string += "key="; request\_string += "2YGO2FHN3XI3GFE7"; request\_string += "&"; request\_string += "field4"; request\_string += "="; request\_string += adc3; http.begin(request\_string); http.GET();

http.end();

}

delay(10);}



CODING FOR SECOND NODEMCU:

#include <ESP8266WiFi.h> #include <ESP8266HTTPClient.h>

String thingSpeakAddress= ["http://api.thingspeak.com/update?"](http://api.thingspeak.com/update); String writeAPIKey;

String tsfield1Name;

String request\_string,request\_string1; HTTPClient http;

#include <DHT.h> // Including library for dht

// library

// https://github.com/adafruit/DHT-sensor-library #include <ESP8266WiFi.h>

String apiKey = "77921LPMGM2OAGQE"; // Enter your Write API key

from ThingSpeak

const char \*ssid = "DESKTOP";

// replace with your wifi ssid

and wpa2 key

const char \*pass = "asdfghjkl";

// WIFI Password

const char\* server = "api.thingspeak.com";

#define DHTPIN 0

//pin D3 where the dht11 is connected

DHT dht(DHTPIN, DHT11);

WiFiClient client; void setup()

{ dht.begin(); Serial.begin(115200); delay(3000); WiFi.disconnect(); Serial.println("START");

WiFi.begin("DESKTOP","asdfghjkl");

while ((!(WiFi.status() == WL\_CONNECTED))){ delay(300);

Serial.println("...");

}

Serial.println("I AM CONNECTED");

}

void loop()

{

if (client.connect("api.thingspeak.com",80))

{

request\_string = thingSpeakAddress; request\_string += "key="; request\_string += "77921LPMGM2OAGQE"; request\_string += "&";

request\_string += "field3"; request\_string += "="; request\_string += analogRead(A0); http.begin(request\_string); http.GET();

http.end();

}

delay(10);

float h = dht.readHumidity();

float t = dht.readTemperature(); if (isnan(h) || isnan(t))

{

Serial.println("Failed to read from DHT sensor!"); return;

}

if (client.connect(server,80)) // "184.106.153.149" or api.thingspeak.com

{

String postStr = apiKey; postStr +="&field1="; postStr += String(t); postStr +="&field2="; postStr += String(h); postStr += "\r\n\r\n";

client.print("POST /update HTTP/1.1\n"); client.print("Host: api.thingspeak.com\n"); client.print("Connection: close\n"); client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");

client.print("Content-Type: application/x-www-form-urlencoded\n"); client.print("Content-Length: ");

client.print(postStr.length()); client.print("\n\n"); client.print(postStr); Serial.print("Temperature: "); Serial.print(t);

Serial.print(" degrees Celcius, Humidity: "); Serial.print(h);

Serial.print(" Soil Sensor "); Serial.print(A0);

Serial.println("%. Send to Thingspeak.");

}

client.stop(); Serial.println("Waiting...");

// thingspeak needs minimum 15 sec delay between updates delay(10);

}

**GITHUBCODE-**[**https://github.com/IBM-EPBL/IBM-Project-24905-1659950712**](https://github.com/IBM-EPBL/IBM-Project-24905-1659950712)