

PROJECT REPORT

IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

TEAM ID: PNT2022TMID31899

- **TEAM LEAD: YESUDAS.S**
- **TEAM MEMBER 1: THARUN KUMAR**
- **TEAM MEMBER 2: GOKULAVASAN.M**
- **TEAM MEMBER 3: PRAKASHKUMAR.S**

TABLE OF CONTENTS :

1 INTRODUCTION	3
1.1 OVERVIEW	
1.2 PURPOSE	
2 LITERATURE SURVEY	5
2.1 EXISTING PROBLEM	
2.2 REFERENCES	
2.3 PROBLEM STATEMENT	
3 IDEATION & PROPOSED SOLUTION	7
3.1 EMPATHY MAP CANVAS	
3.2 IDEATION & BRAINSTROMING	
3.3 PROPOSE SOLUTION	
4 REQUIREMENT ANALYSIS	10
4.1 FUNCTION REQUIREMENTS	
4.2 NON-FUNCTION REQUIREMENTS	
5 PROJECT DESIGN	12
5.1 DATA FLOW DIAGRAMS	
5.2 SOLUTION & TECHNICAL ARCHITECTURE	
5.3 USER STORIES	
6 PROJECT PLANNING & SCHEDULING	15
6.1 SPRINT PLANNING & ESTIMATION	
6.2 SPRINT DELIVERY SCHEDULE	
7 CODING & SOLUTIONING	18
7.1 FEATURE 1	
7.2 FEATURE 2	
8 TESTING	22
8.1 TEST CASES	
8.2 CODE TESTING	
9 RESULTS	24
10 ADVANTAGES & DISADVANTAGES	26
10.1 ADVANTAGES	
10.2 DISADVANTAGES	
11 CONCLUSION	27
12 FUTURE SCOPE	28
13 APPENDIX	29
13.1 SOURCE CODE	
13.2 GitHub & PROJECT DEMO LINK	

CHAPTER - 1

INTRODUCTION

1.1 OVERVIEW :

This is a Smart Agriculture System project based on Internet Of Things (IoT), that can measure soil moisture and temperature conditions for agriculture using Watson IoT services. 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. Wildlife tracking involves acquiring information about the behavior of animals in their natural habitat. This information is used both for scientific and conservation purposes. The primary form of information that needs to be obtained is the location of the animal at certain points in time and this is generally referred to as tracking or radio-tracking. However, due to the similarities in obtaining the information, the terms are frequently used interchangeably. There are remote methods that can be used to track and identify animals visually and through acoustic signals. It is meaningful to design a strategy to roughly localize mobile phones without a GPS by exploiting existing conditions and devices especially in environments without GPS availability. The availability of Bluetooth devices for most phones and the existence of a number of GPS equipped phones in a crowd of phone users enable us to design a Bluetooth aided mobile phone localization strategy.

➤ **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. The main aim of our project is to protect the crops from damage caused by animal as well as divert the animal without any harm. Crops in farms are many times ravaged by local animals like buffaloes, cows, goats, birds etc. This leads to huge losses for the farmers. It is not possible for farmers to barricade entire fields or stay on field 24 hours and guard it. So here we propose automatic crop protection system from animals. Animal detection system is designed to detect the presence of animal and offer a warning. In this project we used PIR and ultrasonic sensors to detect the movement of the animal and send signal to the controller. It diverts the animal by producing sound and signal further, this signal is transmitted to GSM and which gives an alert to farmers and forest department immediately. Index terms- PIR Sensor, Microcontroller, MATLAB, GPS Module, GSM Module

1.2.1 SCOPE OF WORK :

- Create a device in IBM Cloud Account.
- Install Node-RED and configure the nodes that we want to use in the project
- . ➤ Create the open weather map account and get the API key and the weather conditions using API key in the Node-RED.
- Create a web application for user interaction for observation and control actions.

CHAPTER - 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM :

- In agriculture water is needed for the crops for their growth. If the Soil gets dry it is necessary to supply water. But sometime if the farmer doesn't visit the field, it is not possible to know the condition of soil.

- Sometimes over supply of water or less supply of water affects the growth of crops

- Sometimes if the weather/temperature changes suddenly it is necessary to take certain actions.

- Specific crops grow better in specific conditions, they may get damaged due to bad weather.

2.2 REFERENCES :

[1]. Pavithra D.S, M. S .Srinath, "GSM based Automatic Irrigation Control System for Efficient Use of Resources and Crop Planning by Using an Android Mobile", IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE) Vol 11, Issue I, Jul-Aug 2014, pp 49-55.

[2].Shiraz Pasha B.R., Dr. B Yogesha, "Microcontroller Based Automated Irrigation System", The International Journal Of Engineering And Science (IJES), Volume3, Issue 7, pp 06-09, June 2014.

[3].Venkata Naga RohitGunturi, "Micro Controller Based Automatic Plant Irrigation System", International Journal of Advancements in Research & Technology, Volume 2, Issue4,

[4]. S.Gopinath, K.Govindaraju, T.Devika, N.SuthanthiraVanitha, "GSM based Automated Irrigation Control using Raingun Irrigation System", International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 2, February

[5]. Mahir Dursun and Semih Ozden, "A wireless application of drip irrigation automation supported by soil moisture sensors", Scientific Research and Essays, Volume 6(7), pp. 1573-1582, 4 April, 2011.

[6]. LaxmiShabadi, NandiniPatil, Nikita. M, Shruti. J, Smitha. P&Swati.C, "Irrigation Control System Using Android and GSM for Efficient Use of Water and Power", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 7, July 2014.

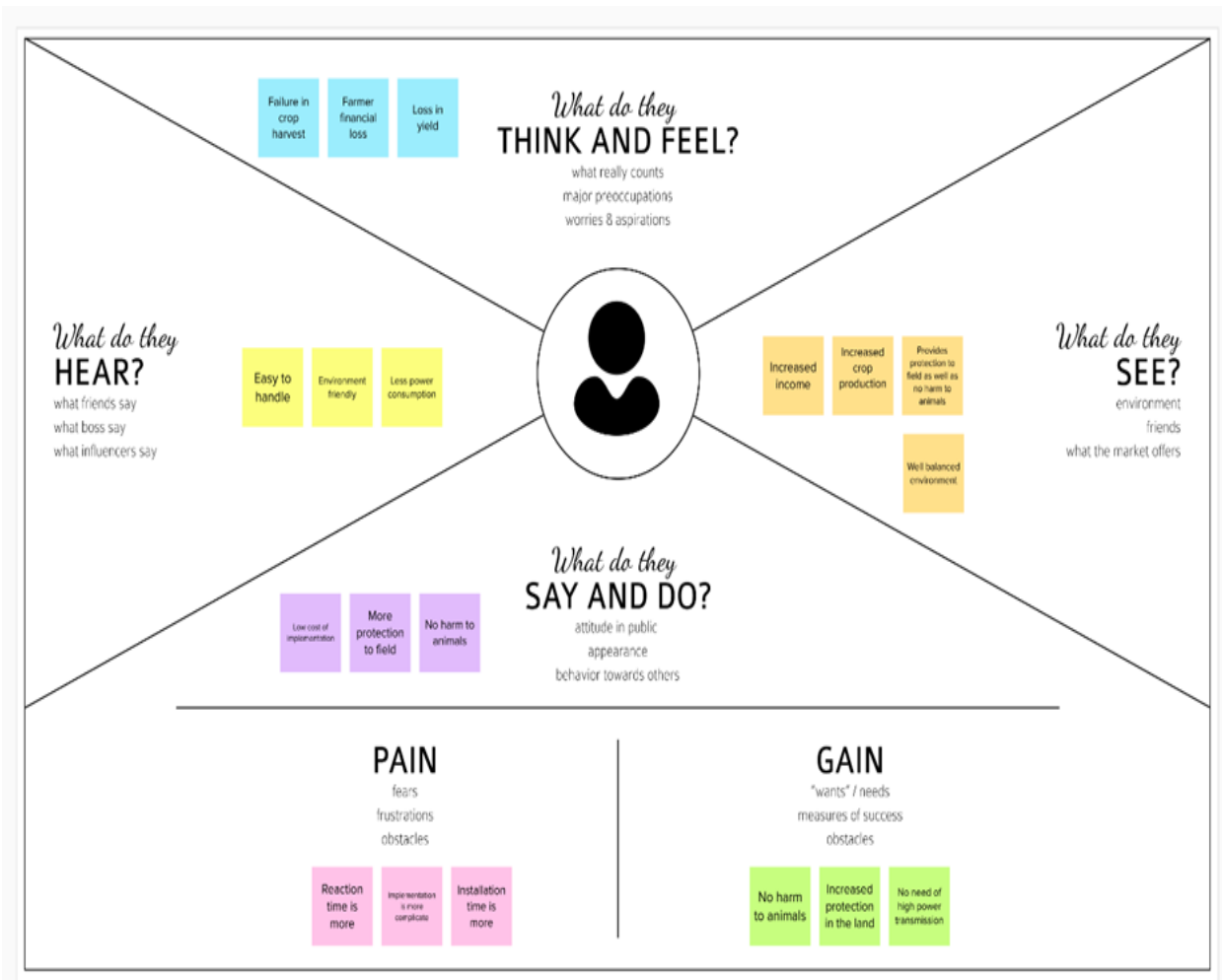
2.3 PROBLEM STATEMENT DEFINITION :

Most of the crop maintenance systems in our country are done manually. Farmers stay in the agricultural lands for longer duration for crops irrigation and field maintenance. The accurate value of the soil moisture level, crop wetness, Water level and growth level are not known. Present there is emerging global water crisis where managing scarcity of water has become a tedious job and there are conflicts between users of water. This is an era where human use and pollution of water resource have crossed the levels which lead to limit food production and low down the ecosystem. The major reason for these limitations is the growth of population which is increasing at a faster rate than the production of food and after a few years this population will sum up to 3-4 billion. Thus growth can be seen in countries which have shortage of uniform crop maintenance and are economically poor. Because of growth in population there is a huge demand to raise food production by 50% in the next half century to maintain the capita, based on an assumption that productivity of existing farm land does not decline. The crop water stress index called as CWSI existed around 30 years ago. This crop water stress index was then integrated using measurements of infrared canopy temperatures, ambient air temperatures, and atmospheric vapor pressure values to determine when to irrigate using drip irrigation. The management of these farms which are in greenhouses will require a data acquisition to be located in each greenhouse and the control room where a control unit is located. These are separated from the production area. At present, the data is transferred using wired communication called field bus. This data is transferred between greenhouses and control room. All the problems related here is presented using CAN and ZigBee protocols.

CHAPTER - 3

IDEATION AND PROPOSED SOLUTION


3.1 EMPATHY MAP CANVAS :



3.2 IDEATION & BRAINSTROMING :

Crops in farms are many times ravaged by local animals like buffaloes, cows, goats, birds, and fire etc. This leads to huge losses for the farmers. It is not possible for farmers to barricade entire fields or stay on field 24 hours and guard it. So here we propose automatic crop protection system from animals and fire. This is aarduino Uno based system using microcontroller. This system uses a motion sensor to detect wild animals approaching near the field and smoke sensor to detect the fire. In such a case the sensor signals the microcontroller to take action.If there is a smoke, it immediately turns ON the motor. This ensures complete safety of crops from animals and from fire thus protecting the farmer's loss. This is aarduino. Uno based system using microcontroller. This system uses a motion sensor to detect wild animals approaching near the field and smoke sensor to detect the fire. In such a case the sensor signals the7 microcontroller to take action.

Template



Brainstorm & idea prioritization

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

🕒 10 minutes to prepare
🕒 1 hour to collaborate
👤 2-6 people recommended

[Share template feedback](#)

➔

Before you collaborate

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

🕒 10 minutes

A

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

C

Learn how to use the facilitation tools

Use the Facilitation Supercpowers to run a happy and productive session.

[Open article](#) ➔

1

Define your problem statement

The main purpose of the project is to protect the crops from animals. Also it monitors the temperature and soil moisture level.

🕒 5 minutes

PROBLEMS

1. Improper maintenance of crops against various environmental factors such as temperature climate, topography and soil quality which results in crop destruction.
2. Requires protecting crops from Wild animals attacks, birds and pests.
3. Lack of Knowledge among farmers in usage of fertilizers and hence crops are affected due to high ammonia, urea, potassium and high PH level fertilizers.

8

3.3 PROPOSED SOLUTION :

Moisture sensor is interfaced with Arduino Microcontroller to measure the moisture level in soil and relay is used to turn ON and OFF the motor pump for managing the excess water level. It will be updated to authorities through IOT. Temperature sensor connected to 9 microcontroller is used to monitor the temperature in the field. The optimum temperature required for crop cultivation is maintained using sprinklers. IOT based fertilizing methods are followed, to minimize the negative effects on growth of crops while using fertilizers. The PIR sensor and UV sensors detect the motion of animals and birds for a particular arrange. The thermal radiation temperature of humans at different ages is fed to the system so there won't be any false alarm. If any invasion of animals is found, the camera focuses on the region and the processed image is sent to the farmer . After seeing the image of the animal that entered, they can decide to take any actions. A fence is built around the field to prevent large animals from entering where the sensors are placed at all the corners of the field fully covering the entire region.

Proposed Solution Template:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	<ul style="list-style-type: none">• Crops are not irrigated properly due to insufficient labour forces.• Improper maintenance of crops against various environmental factors such as temperature climate, topography and soil quality which results in crop destruction.• Lack of knowledge among farmers in usage of fertilizers and hence crops are affected due to high ammonia, urea, potassium and high PH level fertilizers.• Requires protecting crops from Wild animals attacks, birds and pests.
2.	Idea / Solution description	<ul style="list-style-type: none">• Moisture sensor is interfaced with Arduino Microcontroller to measure the moisture level in soil and relay is used to turn ON and OFF the motor pump for managing the excess water level. It will be updated to authorities through IOT.• Temperature sensor connected to microcontroller is used to monitor the temperature in the field. The optimum temperature required for crop cultivation is maintained using sprinklers.• IOT based fertilizing methods are followed, to minimize the negative effects on growth of crops while using fertilizers.• Image processing techniques with IOT is followed for crop protection against animal attacks.
3.	Novelty / Uniqueness	<ul style="list-style-type: none">• Automatic crop maintenance and protection using embedded and IOT technology.
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none">• This proposed system provides many facilities which helps the farmers to maintain the crop field without much loss.
5.	Business Model (Revenue Model)	<ul style="list-style-type: none">• This prototype can be developed as product with minimum cost with high performance .
6.	Scalability of the Solution	<ul style="list-style-type: none">• This can be developed to a scalable product by using sensors and transmitting the data through Wireless Sensor Network and Analysing the data in cloud and operation is performed using robots

CHAPTER - 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS :

By automatic crop monitoring process, farmers would be able to know the right amount of water and Nutritionists at the right time thus to maintain the growth of crop. The farmer can measure the water level, Water level and crop growth at any place. These values in agricultural land are measured by using sensors. It also assists the farmer to maintain the crop. The WSN for remote monitoring of crop field consists of set of wireless sensor nodes distributed in an area called end devices or sensor nodes. They have a stronger battery, a larger memory and more computation power, the sensor nodes collect the data from the field by using sensors and this data is send to the path between end devices. The collected data is send to the internet and pc through WSN. The main of this implementation was to demonstrate that the automatic irrigation system can be used to optimize /reduce water usage. This project describes details of the design and instrumentation of variable rate irrigation, a wireless sensor network, and software for real-time in-field sensing and control of a site-specific precision linear-move irrigation system. Field conditions were site-specifically monitored by six in-field sensor stations distributed across the field based on a soil property map, and periodically sampled and wirelessly transmitted to a base station. An irrigation machine was converted to be electronically controlled by a programming logic controller that updates georeferenced location of sprinklers from a differential Global Positioning System (GPS) and wirelessly communicates with a computer at the base station.

A software application was advanced by programming the verge values of soil moisture water level that was automated into a microcontroller. The proposed hardware of this system includes Raspberry pi, Temperature, humidity, Water level and soil moisture sensors, LCD. The system is low cost & low power consuming so that anybody can afford it. The data monitored is collected at the server. It can be used in precision farming. The system should be designed in such a way that even illiterate villagers can operate it. They themselves can check different parameters of the soil like salinity, acidity, moisture etc. from time to time. During irrigation period they have to monitor their distant pump house throughout the night as the electricity supply is not consistent.

4.2 NON-FUNCTIONAL REQUIREMENTS :

USABILITY : The system shall allow the users to access the system with pc using web application. The system uses a web application as an interface. The system is user friendly which makes the system easy.

AVAILABILITY : The system is available 100% for the user and is used 24 hrs a day and 365 days a year. The system shall be operational 24 hours a day and 7 days a week.

SCALABILITY : Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing demands.

SECURITY : A security requirement is a statement of needed security functionality that ensures one of many different security properties of software is being satisfied.

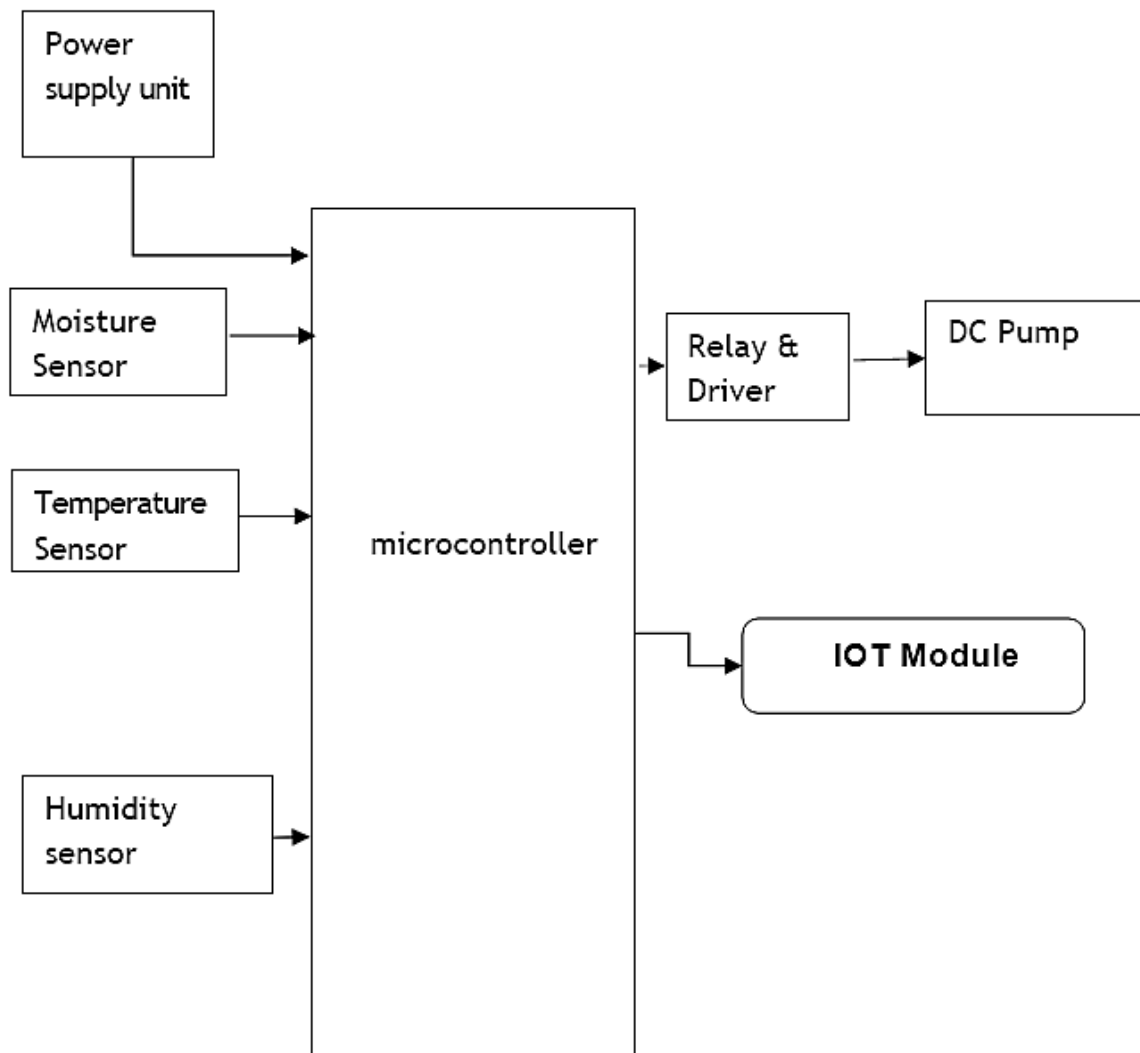
PERFORMANCE : The information is refreshed depending upon whether some updates have occurred or not in the application. The system shall respond to the member in not less than two seconds from the time of the request submittal. The system shall be allowed to take more time when doing large processing jobs. Responses to view information shall take no longer than 5 seconds to appear on the screen.

RELIABILITY : The system has to be 100% reliable due to the importance of data and the damages that can be caused by incorrect or incomplete data. The system will run 7 days a week. 24 hours a day.

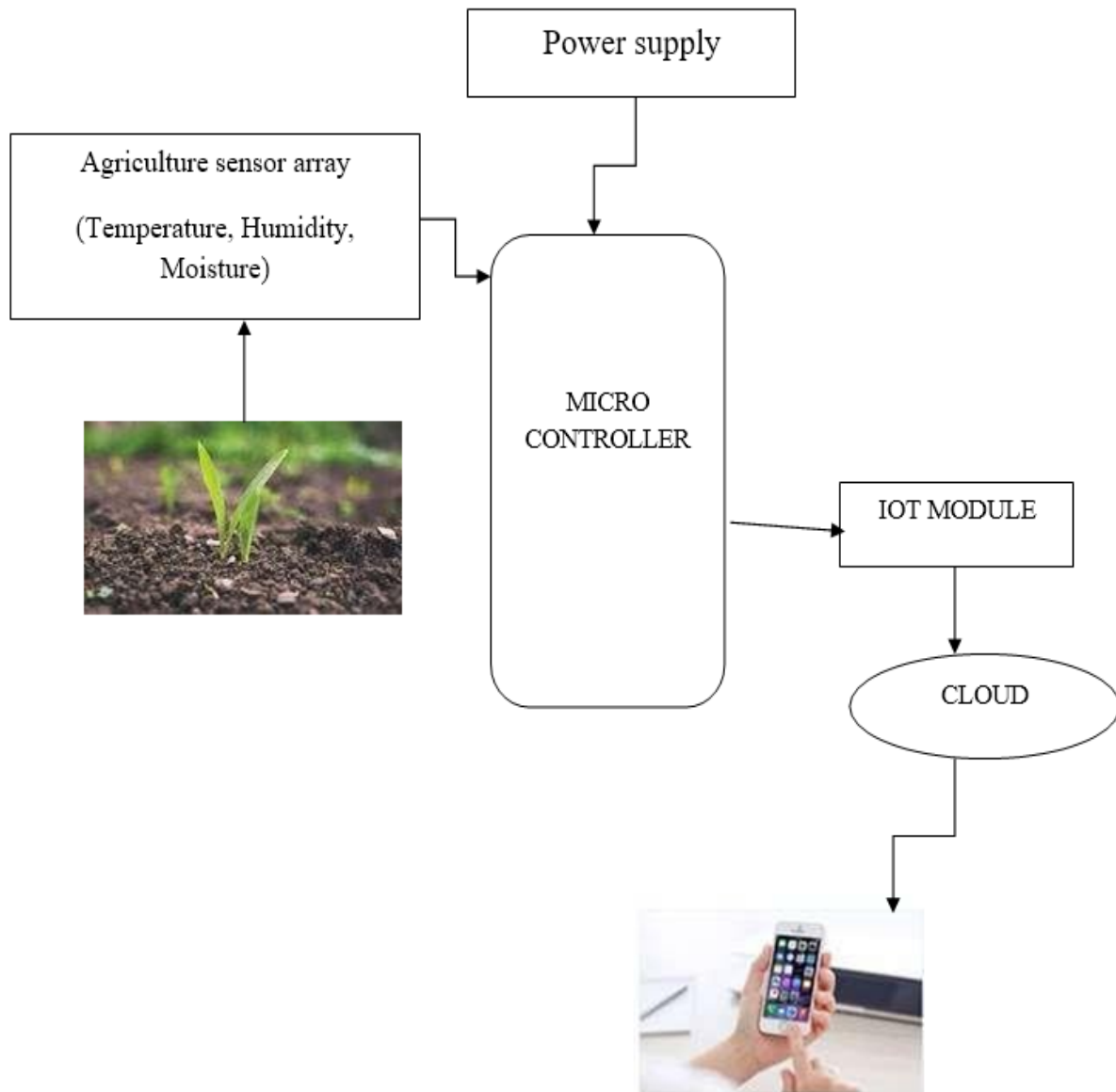
CHAPTER - 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS :



5.2 SOLUTION & TECHNICAL ARCHITECTURE :



5.3 USER STORIES :

FR No	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Visibility	Sensor nearing the crop field and to check the soil moisture and temperature with the help of humidity sensor
FR-2	User Reception	The Data like values of Temperature, Humidity, Soil moisture sensors are received via SMS
FR-3	User Understanding	Based on the sensor data value to get the information about present of farming land
FR-4	User Action	The user needs take action like destruction of crop residues, deep plowing, crop rotation, fertilizers, strip cropping, scheduled planting operations.

CHAPTER - 6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION :

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1		US-1	Create the IBM Cloud services which are being used in this project.	6	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-1		US-2	Configure the IBM Cloud services which are being used in completing this project.	4	Medium	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-2		US-3	IBM Watson IoT platform acts as the mediator to connect the web application to IoT devices, so create the IBM Watson IoT platform.	5	Medium	Sivasankar S, Nagulan N, Velmurugan A, Prasanth S

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
						Sivasankar S
Sprint-2		US-4	In order to connect the IoT device to the IBM cloud, create a device in the IBM Watson IoT platform and get the device credentials.	5	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-3		US-1	Configure the connection security and create API keys that are used in the Node-RED service for accessing the IBM IoT Platform.	10	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-3		US-2	Create a Node-RED service.	10	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-3		US-1	Develop a python script to publish random sensor data such as temperature, moisture, soil and humidity to the IBM IoT platform	7	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-3		US-2	After developing python code, commands are received just print the statements which represent the control of the devices.	5	Medium	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
						Sivasankar S
Sprint-4		US-3	Publish Data to The IBM Cloud	8	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-4		US-1	Create Web UI in Node- Red	10	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S
Sprint-4		US-2	Configure the Node-RED flow to receive data from the IBM IoT platform and also use Cloudant DB nodes to store the received sensor data in the cloudant DB	10	High	Sivasankar S, Nagulan N, Praveen kumar M, Velmurugan A, Prasanth S

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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

CHAPTER - 7

CODING & SOLUTIONING

CODING & SOLUTIONING:

1. Create a device in IBM Cloud.
2. Connect the device to IBM Simulator to get the weather conditions.
3. Build Node-RED flow to build a web application to display the weather conditions and control the devices.
4. Get the real time weather condition data from open weather map and integrate it in the Node-RED.
5. Control the working of the web application to the devices by python coding.

7.1 FEATURE 1 :

Develop the Python code for accessing and finding the Motor Status.If Excess of water given to crops then the crops are hardly affected and water also waste.According to prevention of crops, the below code runs for accessing motor status.

motor.py :

```
import time
import sys
import ibmiotf.application # to install pip install ibmiotf
import ibmiotf.device

# Provide your IBM Watson Device Credentials
organization = "8gyz7t" # replace the ORG ID
deviceType = "weather_monitor" # replace the Device type
deviceId = "b827ebd607b5" # replace Device ID
authMethod = "token"
authToken = "LWVpQPpVQ166HWN48f" # Replace the authtoken
```

```

def myCommandCallback(cmd): # function for Callback
    if cmd.data['command'] == 'motoron':
        print("MOTOR ON IS RECEIVED")
    elif cmd.data['command'] == 'motoroff':
        print("MOTOR OFF IS RECEIVED")
    if cmd.command == "setInterval":
        if 'interval' not in cmd.data:
            print("Error - command is missing required information:
'interval'")
        else:
            interval = cmd.data['interval']
    elif cmd.command == "print":
        if 'message' not in cmd.data:
            print("Error - command is missing required information:
'message'")
        else:
            output = cmd.data['message']
            print(output)

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:
    deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()

```

7.1 FEATURE 2 :

Develop the python code for generating the random value of temperature , humidity and soil moisture level.

monitor.py :

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

# Provide your IBM Watson Device Credentials
organization = "8gyz7t" # replace the ORG ID
deviceType = "weather_monitor" # replace the Device type
deviceId = "b827ebd607b5" # replace Device ID
authMethod = "token"
authToken = "LWVpQPpVQ166HWN48f" # Replace the authtoken
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:
    temp=random.randint(0,100)
    pulse=random.randint(0,100)
    soil=random.randint(0,100)
    data = { 'temp' : temp, 'pulse': pulse , 'soil':soil}

    #print data
    def myOnPublishCallback()
Print ("Published Temperature = %s C" % temp, "Humidity = %s %%" %
pulse,"Soil Moisture = %s %%" % soil,"to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoT")
    time.sleep(1)
    deviceCli.commandCallback = myCommandCallback

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

```

CHAPTER - 8

TESTING

8.1 TEST CASES :

The main purpose of a use case diagram is to show what system functions are performed for which actors in the system can be depicted. A use case diagram is a type of behavioral diagram created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

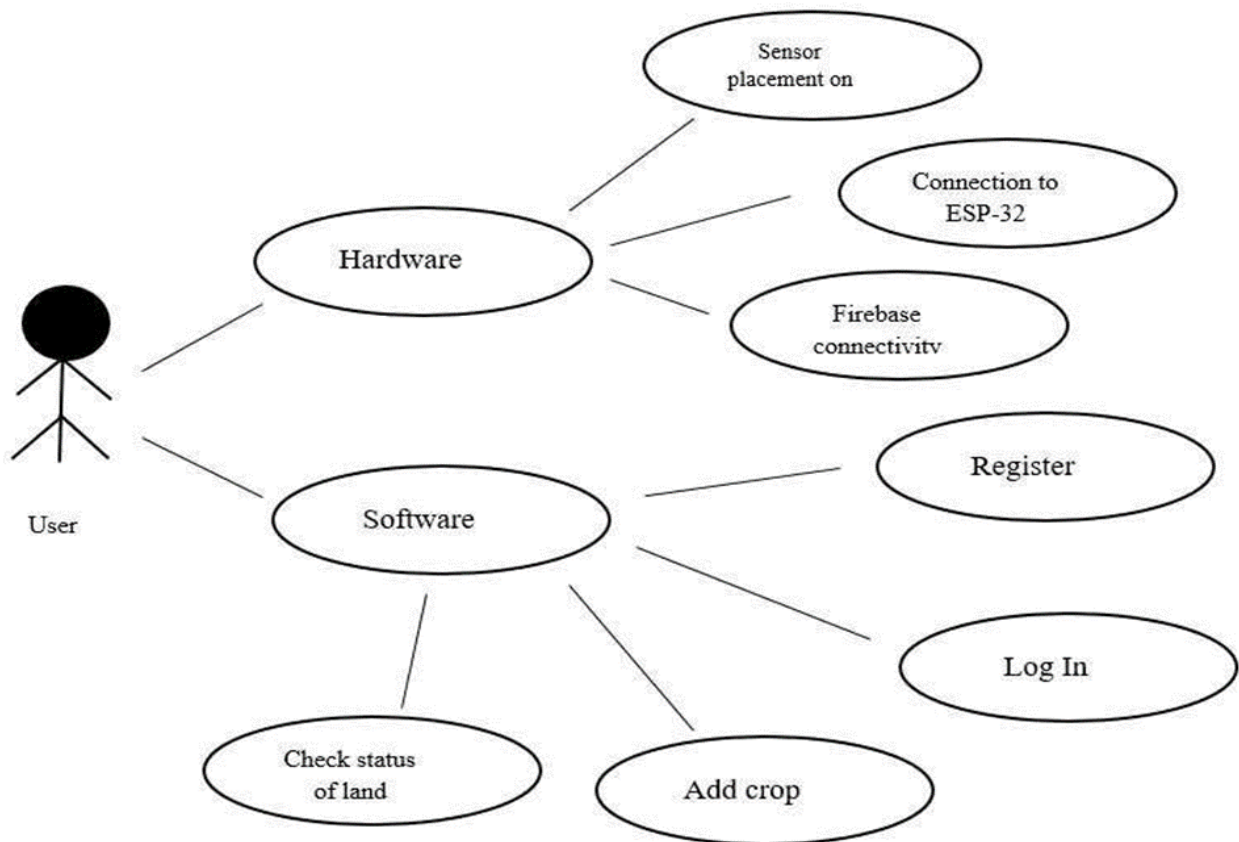


Fig. 3.5.1 Use case Diagram

8.2 CODE TESTING :

```
Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\darun\OneDrive\Documents\monitor.py =====
2022-11-18 14:55:56,531 ibmiotf.device.Client INFO Connected successfully: d:ebf2oy:Humidity:123456
Published Temperature = 50 C Humidity = 24 % Soil Moisture = 26 % to IBM Watson
Published Temperature = 20 C Humidity = 16 % Soil Moisture = 35 % to IBM Watson
Published Temperature = 7 C Humidity = 82 % Soil Moisture = 47 % to IBM Watson
Published Temperature = 92 C Humidity = 19 % Soil Moisture = 42 % to IBM Watson
Published Temperature = 21 C Humidity = 10 % Soil Moisture = 98 % to IBM Watson
Published Temperature = 61 C Humidity = 37 % Soil Moisture = 75 % to IBM Watson
Published Temperature = 55 C Humidity = 2 % Soil Moisture = 6 % to IBM Watson
Published Temperature = 31 C Humidity = 42 % Soil Moisture = 65 % to IBM Watson
Published Temperature = 48 C Humidity = 1 % Soil Moisture = 58 % to IBM Watson
Published Temperature = 53 C Humidity = 18 % Soil Moisture = 65 % to IBM Watson
Published Temperature = 90 C Humidity = 14 % Soil Moisture = 88 % to IBM Watson
Published Temperature = 61 C Humidity = 63 % Soil Moisture = 22 % to IBM Watson
Published Temperature = 68 C Humidity = 36 % Soil Moisture = 52 % to IBM Watson
Published Temperature = 92 C Humidity = 54 % Soil Moisture = 4 % to IBM Watson
Published Temperature = 41 C Humidity = 97 % Soil Moisture = 61 % to IBM Watson
Published Temperature = 33 C Humidity = 72 % Soil Moisture = 0 % to IBM Watson
Published Temperature = 68 C Humidity = 36 % Soil Moisture = 79 % to IBM Watson
|
```

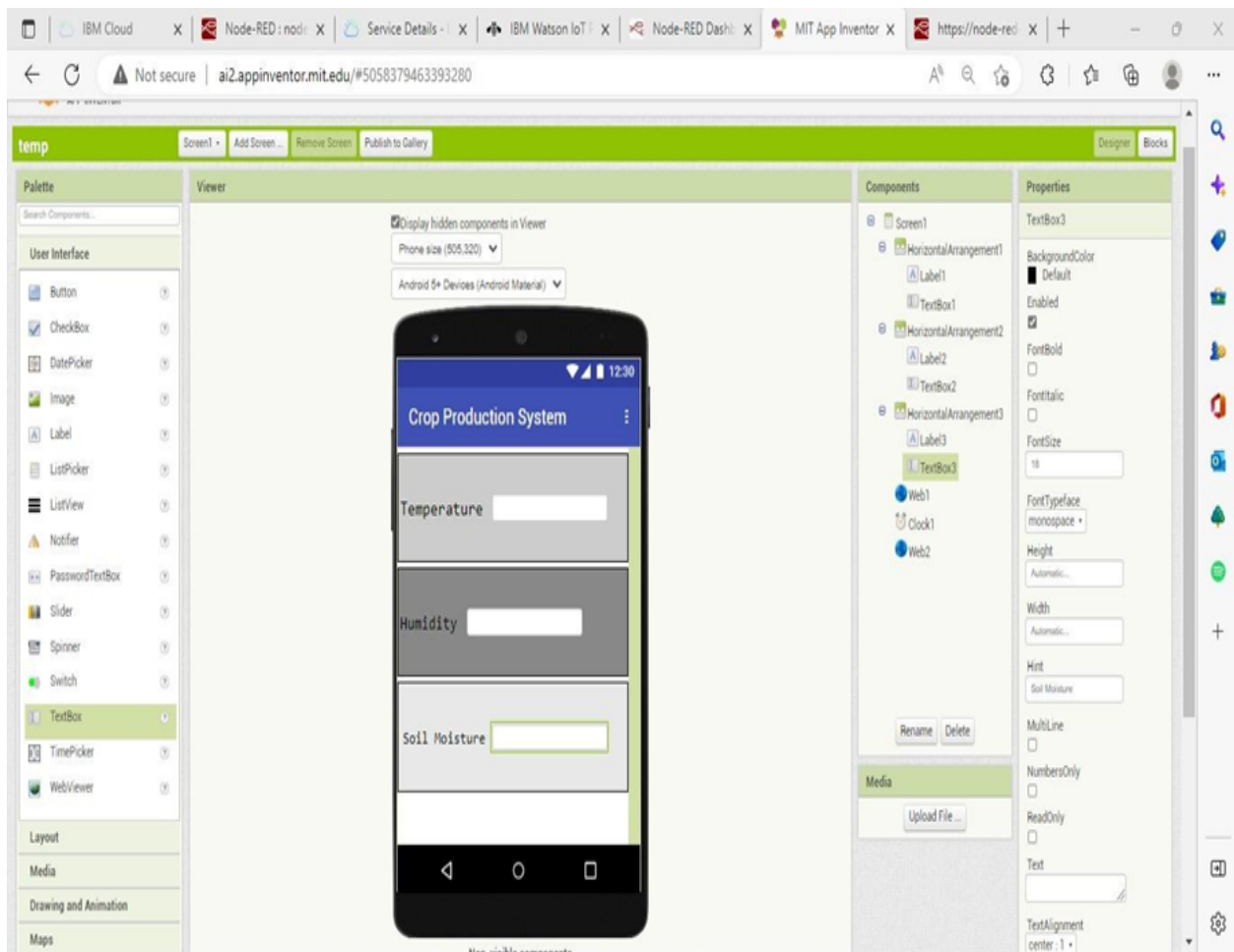
CHAPTER - 9

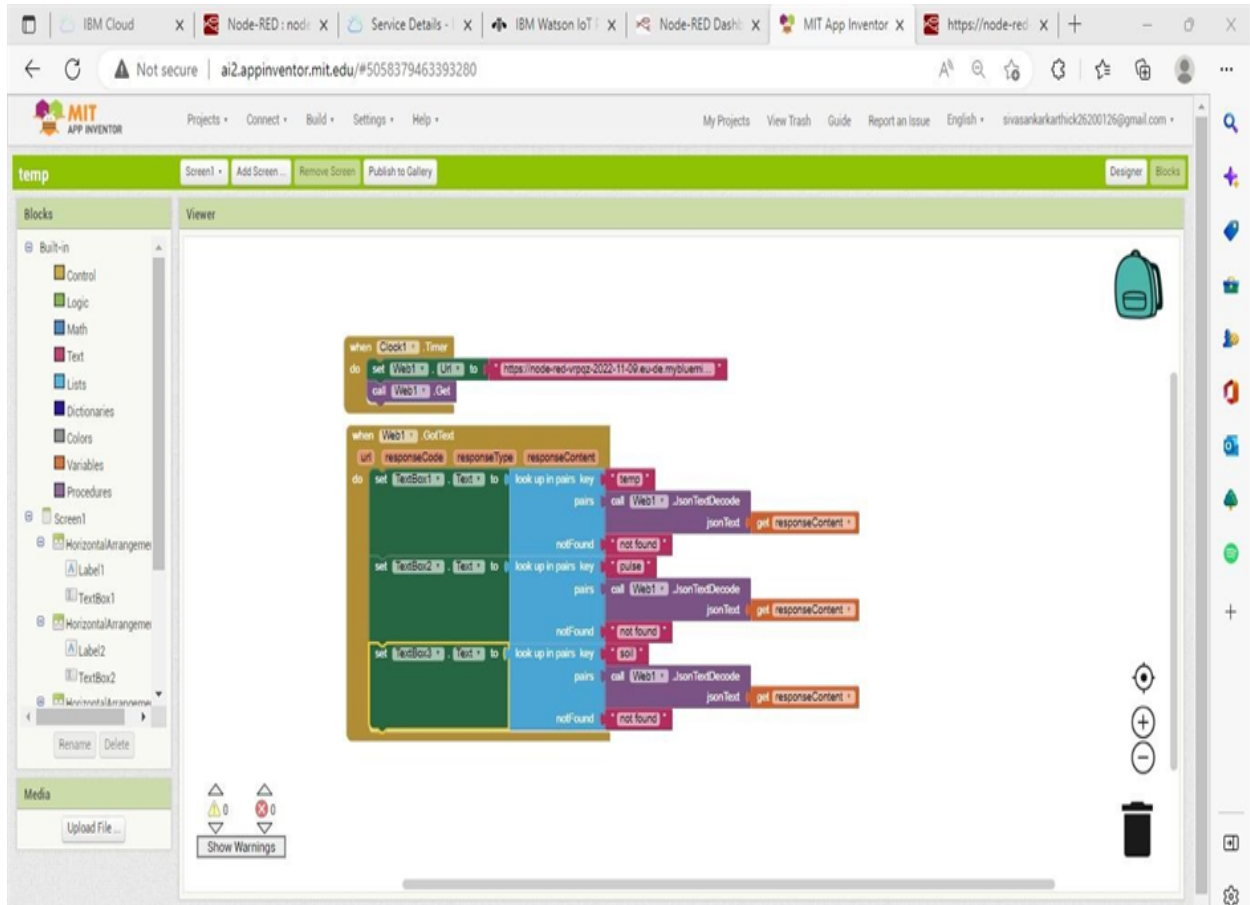
RESULTS

RESULTS :

We have successfully build a web based UI and integrated all the services using Node-RED.

Web Application : <https://node-red-aab.eu-gb.mybluemix.net/ui>





CHAPTER - 10

ADVANTAGES & DISADVANTAGES

10.1 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.

10.2 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.

APPLICATIONS :

- Precision Farming that is farming processes can be made more controlled and accurate.
- Live monitoring can be done of all the processes and the conditions on the agricultural field.
- All the controls can be made just on the click.
- Quality can be maintained.

CHAPTER - 11

CONCLUSION

IOT based crop field monitoring system serves as a reliable and efficient system for monitoring agricultural parameters. The corrective action can be taken. Wireless monitoring of field not only allows user to reduce the human power, but it also allows user to see accurate changes in it. It is cheaper in cost and consumes less power. The GDP per capita in Agri sector can be increased. Agriculture is a backbone of human civilization since man has started agriculture. As the generation evolved, man developed many methods of crop monitoring to provide growth to the crop. In the present scenario on conservation of water is of high importance. Present work is attempts to save the natural resources available for human kind. By continuously monitoring the status of the soil, we can control the flow of water and thereby reduce the wastage. This review is proposed to supports aggressive water management for the agricultural land. Microcontroller in the system promises about increase in systems life by reducing the power consumption resulting in lower power consumption.

After completing this project, it can be concluded that the system works perfectly as planned. It fulfills the two objectives stated at the beginning of this project which is to develop a smart farming application that can monitor the parameters such as temperature, humidity, soil moisture, and pH value, to optimize the use of water by using the controlling system and to analyze vegetative traits of plants using the suitable value of temperature, humidity, soil moisture and pH using Thing Speak. Furthermore, the smart farming system is an efficient method of applying 36 nutrient solutions in which the irrigation system is used as the carrier and the distributor for the plants. In a nutshell, the smart farming system using IoT is well suited for commercial agriculture to maximize profits and yields.

CHAPTER - 12

FUTURE SCOPE

- Future work would be focused more on increasing sensors on this system to 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.
- Smart farming refers to managing farms using modern Information and communication technologies to increase the quantity and quality of products while optimizing the human labor required.
- Among the technologies available for present-day farmers are: Sensors: soil, water, light, humidity, temperature management.
- Agriculture graduates can explore the sea of opportunities in both the public and private sectors. They can start their career as Agriculture Officer, Assistant Plantation Manager, Agricultural Research Scientist, Marketing Executive, Business development executive, and many more.
- 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" 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.

CHAPTER - 13

APPENDIX

13.1 SOURCE CODE :

```
import cv2
import numpy as np
import wiotp.sdk.device
import playsound
import random
import time
import datetime
import ibm_boto3
from ibm_botocore.client import Config, ClientError

#CloudantDB
from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel
from clarifai_grpc.grpc.api import service_pb2_grpc
stub = service_pb2_grpc.V2Stub(ClarifaiChannel.get_grpc_channel())
from clarifai_grpc.grpc.api import service_pb2, resource_pb2
from clarifai_grpc.grpc.api.status import status_code_pb2

#This is how you authenticate
metadata = (('authorization', 'key 5797d941-433e-436a-a480-680d9080a990'),)
COS_ENDPOINT = "https://s3.tok.ap.cloud-object-storage.appdomain.cloud"
COS_API_KEY_ID = "v9n8Zn4r5VpcMVz_HyRY0DrS13jSzph2IEFioVj4-vmT"
COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"
COS_RESOURCE_CRN = "crn:v1:bluemix:public:cloud-
objectstorage:global:a/3f060ee770d94e20a88f49f3da641d6d:f301cab2-2e94-48a1-
a8a05b4968527c54::"
```

```

clientdb = cloudbant("apikey-_pleLXPoaPpnOZ7SMoVKd6tZdsjf54X9LwkFEWB1a0T6",
"0165dca6-1176-4aa5-b0fe-81473e50e35d", url="https://47643860-3553-4211-
ba2ad8e26dd17c08-bluemix.cloudbantnosqlb.appdomain.cloud")
clientdb.connect()
#Create resource
cos = ibm_boto3.resource("s3", ibm_api_key_id=COS_API_KEY_ID,
ibm_service_instance_id=COS_RESOURCE_CRN,
ibm_auth_endpoint=COS_AUTH_ENDPOINT, config=Config(signature_version="oauth"),
endpoint_url=COS_ENDPOINT )
def multi_part_upload(bucket_name, item_name, file_path):

    try:
        print("Starting file transfer for {0} to bucket: {1}\n".format(item_name,
bucket_name))
        #set 5 MB chunks
        part_size = 1024 * 1024 * 5
        #set threshold to 15 MB
        file_threshold = 1024 * 1024 * 15
        #set the transfer threshold and chunk size
        transfer_config = ibm_boto3.s3.transfer.TransferConfig(
            multipart_threshold=file_threshold,
            multipart_chunksize=part_size
        )

        #the upload_fileobj method will automatically execute a multi-part upload
        #in 5 MB chunks size
        with open(file_path, "rb") as file_data:
            cos.Object(bucket_name, item_name).upload_fileobj(
                Fileobj=file_data,
                Config=transfer_config
            )

        print("Transfer for {0} Complete!\n".format(item_name))
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to complete multi-part upload: {0}".format(e))

```

```

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)
    command=cmd.data['command']
    print(command)
    if(command=="lighton"):
        print('lighton')
    elif(command=="lightoff"):
        print('lightoff')
    elif(command=="motoron"):
        print('motoron')
    elif(command=="motoroff"):
        print('motoroff')

myConfig = {
    "identity": {
        "orgId": "ebf2oy",
        "typeId": "Humidity",
        "deviceId": "123456"
    },
    "auth": {
        "token": "12345678"
    }
}

client = wiot.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

database_name = "sample"
my_database = clientdb.create_database(database_name)
if my_database.exists():
    print(f'({database_name})' successfully created.")
cap=cv2.VideoCapture("garden.mp4")
if(cap.isOpened()==True):
    print('File opened')
else:
    print('File not found')

```

```

while(cap.isOpened()):
    ret, frame = cap.read()
    gray = cv3.cvtColor(frame, cv2.COLOR_BGR@GRAY)
    imS= cv2.resize(frame, (960,540))
    cv2.imwrite('ex.jpg',imS)
    with open("ex.jpg", "rb") as f:
        file_bytes = f.read()

#This is the model ID of a publicly available General model. You may use any other
public or custom model ID.
    request = service_pb2.PostModeloutputsRequest(
        model_id='82eaf1c767a74869964531e4d9de5237',
inputs=[resources_pb2.Input(data=resources_pb2.Data(image=resources_pb2.Image(b
ase64=file_bytes)) )])
    response = stub.PostModelOutputs(request, metadata=metadata)
    if response.status.code != status_code_pb2.SUCCESS:
        raise Exception("Request failed, status code: " + str(response.status.code))
    detect=False
    for concept in response.outputs[0].data.concepts:
        #print('%12s: %.f' % (concept.name, concept.value))
        if(concept.value>0.98):
            #print(concept.name)
            if(concept.name=="animal"):
                print("Alert! Alert! animal detected")
                playsound.playsound('alert.mp3')
                picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")
                cv2.imwrite(picname+'.jpg',frame)
                multi_part_upload('Umamaheswari', picname+'.jpg', picname+'.jpg')

json_document={"link":COS_ENDPOINT+'/'+'Umamaheswari'+'/'+'picname+'.jpg'}
    new_document = my_database.create_document(json_document)
    if new_document.exists():
        print(f"Document successfully created.")
        time.sleep(5)
        detect=True

```



```

moist=random.randint(0,100)
humidity=random.randint(0,100)
myData={'Animal':detect,'moisture':moist,'humidity':humidity}
print(myData)
if(humidity!=None):
    client.publishEvent(eventId="status",msgFormat="json", daya=myData, qos=0,
onPublish=None)

    print("Publish Ok..")
client.commandCallback = myCommandCallback
cv2.imshow('frame',imS)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
client.disconnect()
cap.release()
cv2.destroyAllWindows()

```

13.2 GitHub & PROJECT DEMO LINK :

Git Repo : IBM-EPBL/**IBM-Project-39992-1660575428**

DEMO LINK : <https://drive.google.com/file/d/1-bXwHcFSPOjefddmlVe7TtsMvAnpVzDn/view>