



PROJECT REPORT

SMART FARMER IOT - ENABLED SMART FARMING APPLICATION

IBM TEAM ID : PNT2022IMID39877

TEAM MEMBERS

- 1. KUSHAL DARIRA R [TEAM-LEAD] – 511319104038**
- 2. BALAJI R – 511319104009**
- 3. HARI KRISHNA – 5113191044020**
- 4. KAMESH C – 511319104031**

FACULTY MENTOR

SAMUNDEEESWARI M - AP/CSE

Project Report

1. INTRODUCTION

- a. Project Overview
- b. Purpose

2. LITERATURE SURVEY

- a. Existing problem
- b. References
- c. Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

- a. Empathy Map Canvas
- b. Ideation & Brainstorming
- c. Proposed Solution
- d. Problem Solution fit

4. REQUIREMENT ANALYSIS

- a. Functional requirement
- b. Non-Functional requirements

5. PROJECT DESIGN

- a. Data Flow Diagrams
- b. Solution & Technical Architecture
- c. User Stories

6. PROJECT PLANNING & SCHEDULING

- a. Sprint Planning & Estimation
- b. Sprint Delivery Schedule
- c. Reports from JIRA

7. CODING & SOLUTIONING

- a. Feature 1
- b. Feature 2
- c. Database Schema (if Applicable)

8. TESTING

- a. Test Cases

b. User Acceptance Testing

9. **RESULTS**

a. Performance Metrics

10. **ADVANTAGES & DISADVANTAGES**

11. **CONCLUSION**

12. **FUTURE SCOPE**

13. **APPENDIX**

Source Code

GitHub & Project Demo Link

1. INTRODUCTION :

1.1 Project Overview

The Smart Farmer IOT – Enabled Smart Farming Application It Is About How a Farmer Can Initiate Farming Smartly Using Advance Technology Where The Farming Is Made Easy And Efficient Based On The Sensor's and Microcontrollers.

The Overall View of This Project Is About How One Can Detect Soil Moisture, Temperature, Humidity, And When A Motor Should Pump Out Water To The Soil.

A Smart Farming Application is Created Where The Application Collects The Data From The Sensors And Displays It To The User Through Cloud.

The Cloud Collect The Data Using A Node – Red Which Redirect's It To The IBM Watson IOT Platform And In Which The Data's From The Sensors Are Being Transferred Through Cloud.

The User Can Access From Any Part Of The World Connected Through "Internet" And Can Even On And Off Motor Based On The Climatic Conditions.

1.2 Purpose

The Main Purpose of This Project Is To Reduce Labor Cost, Man Power, Detection of Various Agricultural Soil Issues. The Smart Farmer Application Can Detect The Moisture, Temperature, Humidity, And Start Motor From And Parts of The World From One Touch.

The Smart Farmer App is Implemented Based On The Advance Technique of The IOT Domain Where The User Can Find It Easy And Responsive Application For Control And Usage.

2. LITERATURE SURVEY

2.1 Existing problem

The Existing Problem of Smart Famer Application Has Various Applications But Some Has There Own Properties And Implemented Based On The Usage.

There Are Some Smart Farming Application Which Are Made Based On The Client Requirements And The Cost Of The Product Is Competitively High When Compared With This Project.

This Major Cause Of This Project is The Accuracy and The Cost of the Project Is Comparatively Less When Compared With The Other

1. **“IOT Based Smart Irrigation Monitoring And Controlling System”**
2. **“IOT BASED SMART CROP-FIELD MONI- TORING AND AUTOMATION IRRIGATION SYSTEM”**
3. **“IOT Based Smart Agriculture System”**

They Are Some Of The Existing Solutions In The Smart Farmer And In Which The Improvements Have To Be Made In Accuracy And The Cost Which This Project Has Fulfilled All The Requirements With Respect To Cost And The Accuracy.

2.2 Reference

I. “IOT Based Smart Agriculture System” – 2018 – G. Sushanth1, S. Sujatha.

II. “IOT Based Smart Irrigation Monitoring And Controlling System” - 2017 – Shweta B. Saraf, Dhana shri, H. Gawal.

III. “IOT BASED SMART CROP-FIELD MONI- TORING AND Rao, AUTOMATION IRRIGATION SYSTEM” – 2018 – R. Nageswara B.Sridhar

IV. “Smart Agriculture Using Internet of Things with Raspberry Pi.” – 2020 – Zuraida Muhammad, Muhammad Azri Asyraf Mohd Hafez, Nor Adni Mat.

2.3 Problem Statement

(1) Kushal Has An Idea On Converting His Land To Agriculture Land So He Need Everything To Be Smart And Convenient To Use.

(2) Kushal Is Busy With His Professional Life So He Can't Maintain The Field.

(3) Hari Krishna Is a Doctor Were He Has An Idea To Invest some Of his Money In Farming But Has No Idea When And Where To Invest In the Equipment's.

(4) Hari Krishna Also Belongs From A Farmer's Family Were He can Monitor From His Mobile.

(5) Balaji Is Fond Of Travelling And His Mom Has an Plan To Set Up An Vegetable Harvesting in His Field So He Can Use And Maintain It from His Mobile.

(6) Balaji used To Travel a Lot So He Can Monitor It from Remote Location.

(7) Kamesh Is a Child Actor And got Inspired It from His Latest Travel Experience So He Is Planning to Harvest It Without Any Basic Knowledge.

(8) Kamesh Is Leaving In US And He can Monitor it from a remote place where he can access the watering and movement of soil.

3.IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

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

SmartFarmer

IoT Enabled Smart Farming Application

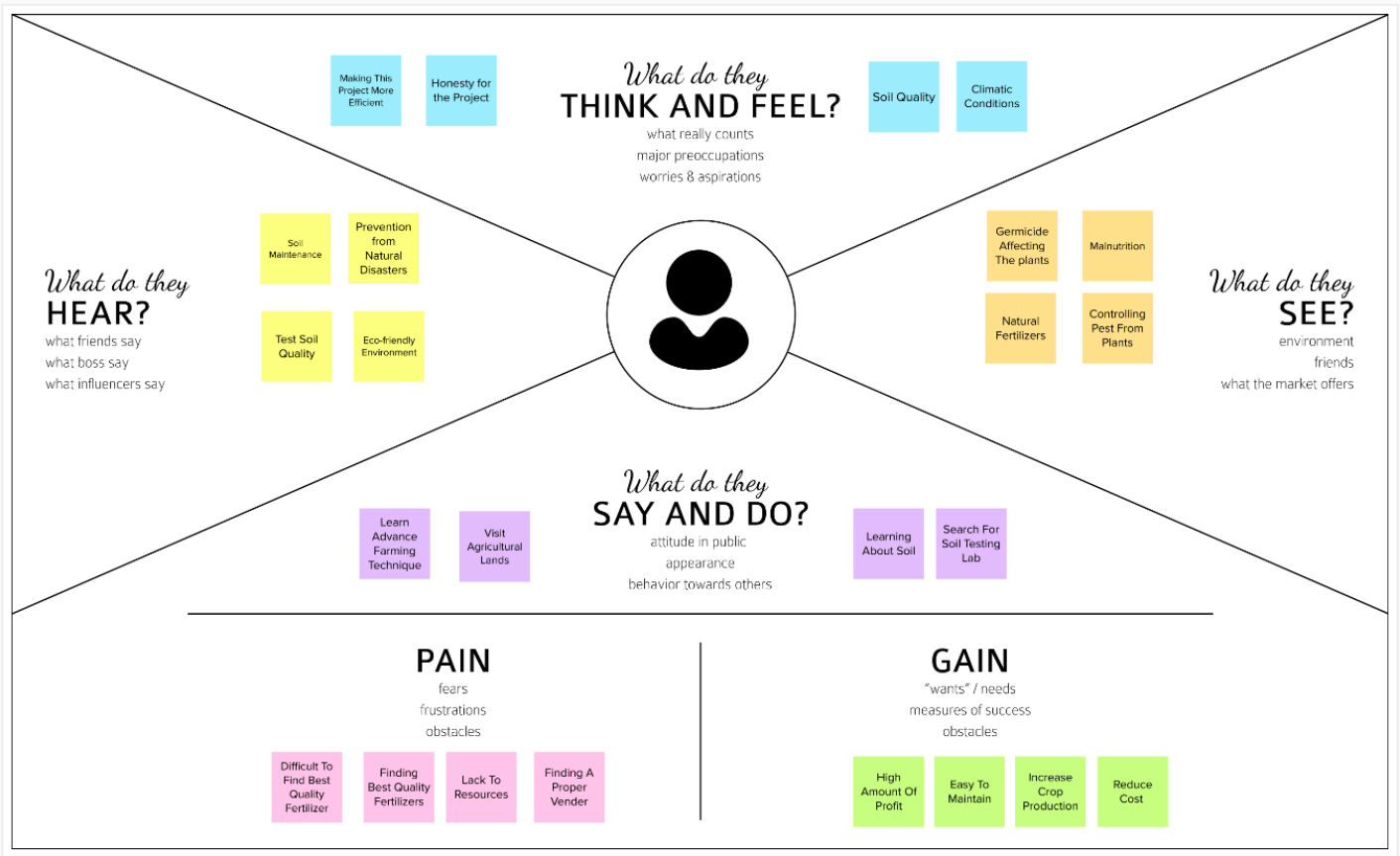


Fig 3.1.1 Empathy Map Canvas

Reference Link : <https://tinyurl.com/4846esrj>

3.2 Ideation & Brainstorming

Brainstorming is a **group problem-solving method that involves the spontaneous contribution of creative ideas and solutions**. This technique requires intensive, freewheeling discussion in which every member of the group is encouraged to think aloud and suggest as many ideas as possible based on their diverse knowledge.

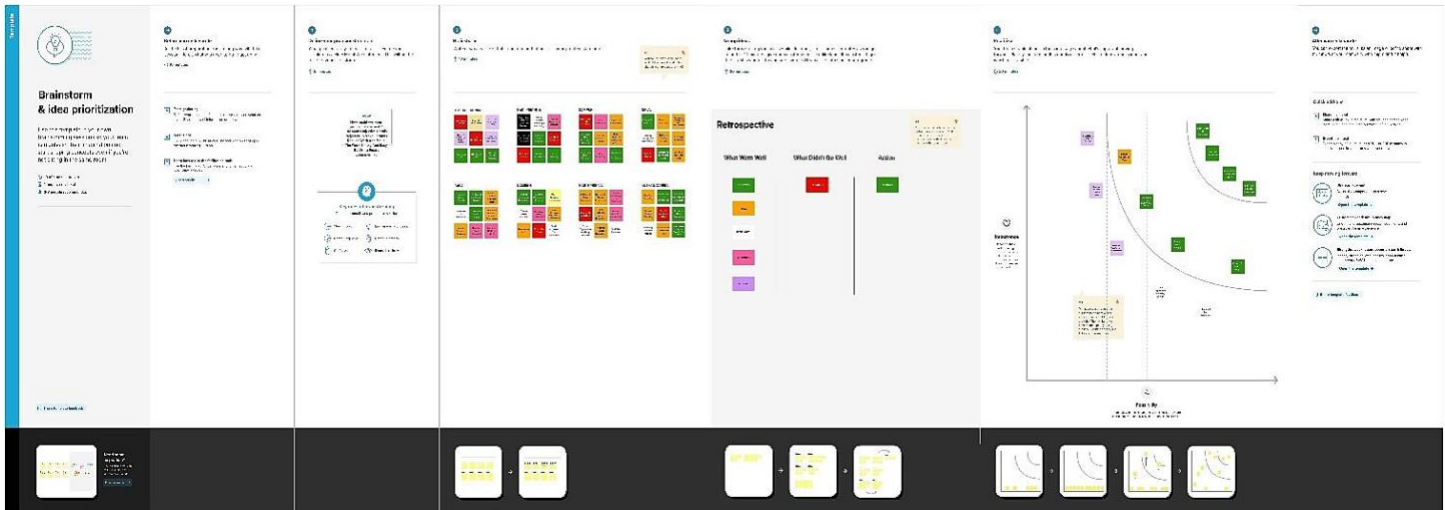
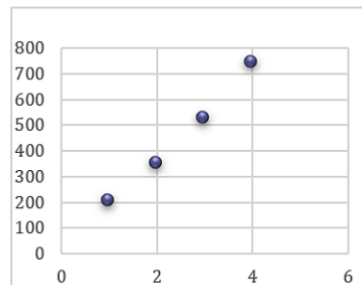


Fig 3.2.1 Brainstorming Diagram

Reference Link : <https://tinyurl.com/ymzt7zwx>

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	<ul style="list-style-type: none"> 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 / Solution description	<ul style="list-style-type: none">● 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	<p>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.</p> <p>REMOTE ACCESS – It helps the farmer to operate the motor from anywhere.</p>										
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none">● 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.	Business Model (Revenue Model)	<p>Revenue (No. of Users vs Months)</p> <p> </p>  <table><thead><tr><th>Months</th><th>User</th></tr></thead><tbody><tr><td>1</td><td>200</td></tr><tr><td>2</td><td>350</td></tr><tr><td>3</td><td>550</td></tr><tr><td>4</td><td>750</td></tr></tbody></table>	Months	User	1	200	2	350	3	550	4	750
Months	User											
1	200											
2	350											
3	550											
4	750											

6.	Scalability of the Solution	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

The Problem-Solution Fit simply means that **you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem**

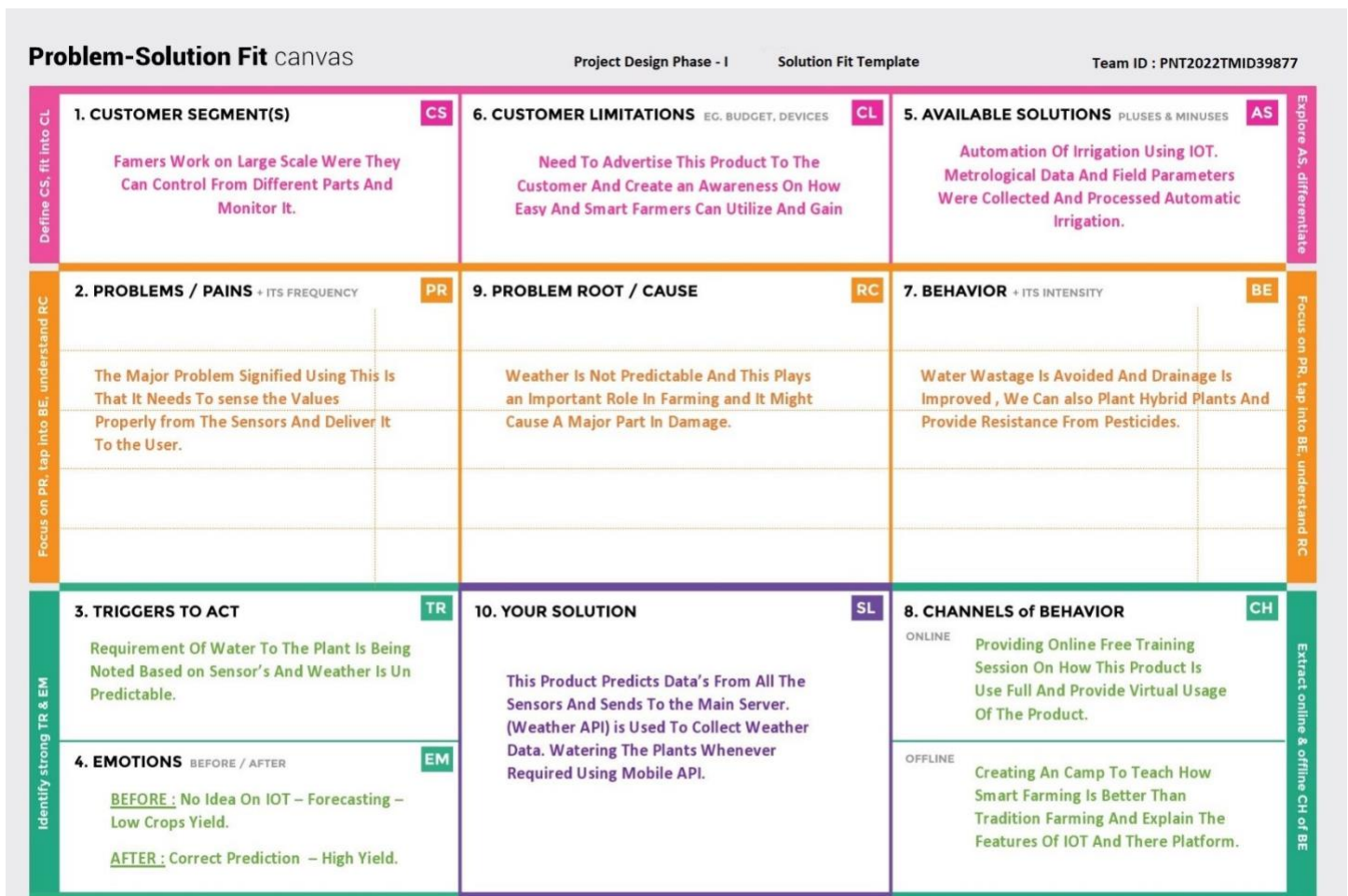


Fig 3.4.1 Problem Solution Fit Diagram

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Objective :

1. *It gives better control to the farmers for their livestock, growing crops, cutting costs, and resources.*
2. *The world's total population touched 6.60 billion in 2000 but is projected to grow to 9.32 billion by 2050. Hence, it is necessary to increase the yield on the limited farmland.*
3. *IoT in agriculture uses robots, drones, remote sensors, and computer imaging combined with continuously progressing machine learning and analytical tools for **monitoring crops, surveying, and mapping the fields, and providing data to farmers for rational farm management plans to save both time and money.***

Focus :

1. ***Sensors for soil scanning and water, light, humidity and temperature management.** Telecommunications technologies such as advanced networking and GPS. Hardware and software for specialized applications and for enabling IoT-based solutions, robotics and automation.*

4.2 Non-Functional requirements

A. Objectives :

1. *IoT smart agriculture products are designed to help monitor crop fields using sensors and by automating irrigation systems. As a result, farmers and associated brands can easily monitor the field conditions from anywhere without any hassle.*
2. *Internet of Things in Agriculture has come up as a second wave of green revolution. The benefits that the farmers are getting by adapting IoT are twofold. It has helped farmers to decrease their costs and increase yields at the same time by improving farmer's decision making with accurate data.*

B. Focus :

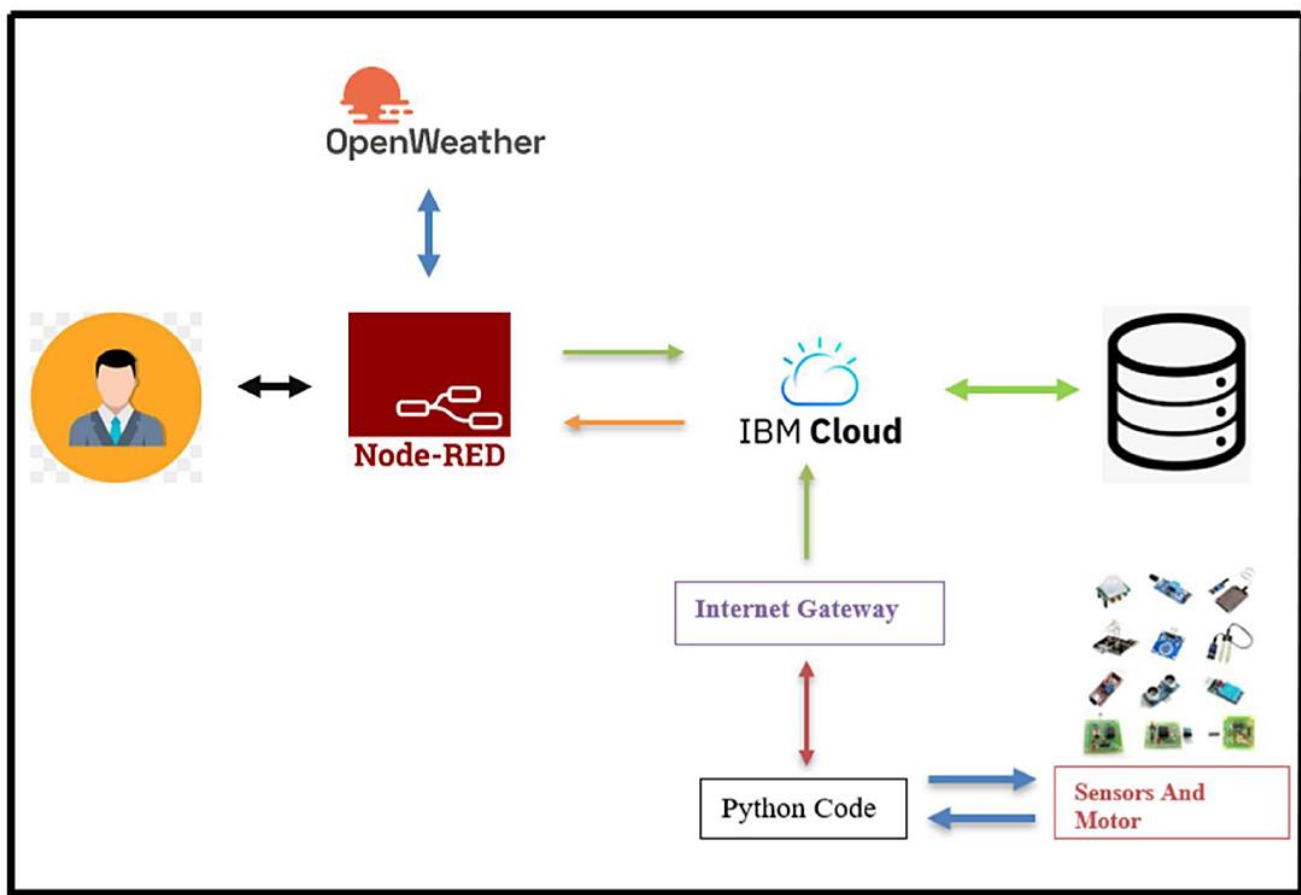
1. *Large landowners and small farmers must understand the potential of IoT market for agriculture by installing smart technologies to increase competitiveness and sustainability in their productions.*
2. *Smart farming is a management concept focused on providing the agricultural industry with the infrastructure to leverage advanced technology – including big data, the cloud and the internet of things (IoT) – for tracking, monitoring, automating and analyzing operations.*

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A data flow diagram (DFD) is a **graphical or visual representation** using a **standardized set of symbols and notations** to describe a business's operations through data movement.

They are often elements of a formal methodology such as Structured



Systems Analysis and Design Method (SSADM).

Fig 5.1.1. Smart Farmer Application And Data Flow Diagram :

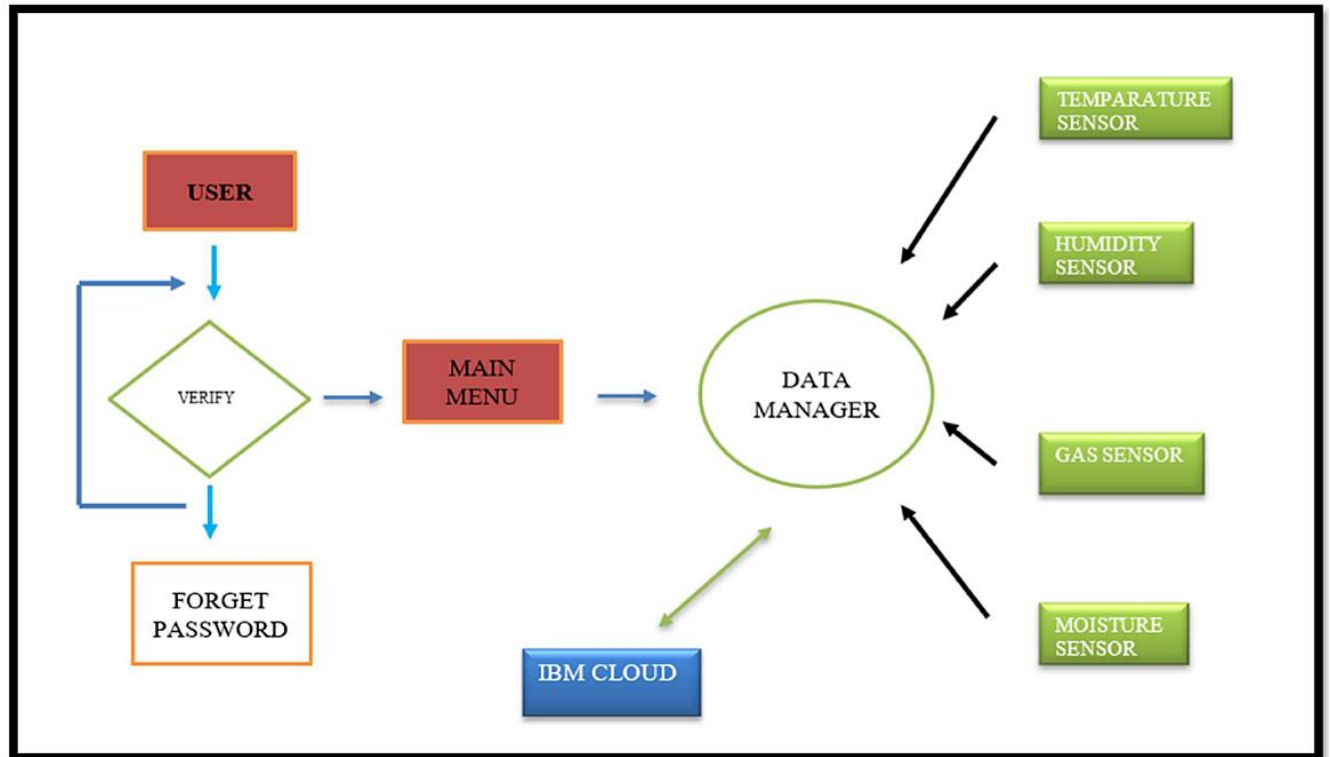


Fig 5.1.2. Data Flow Diagram :

5.2 Solution & Technical Architecture

a. Solution Architecture

Solution architecture is a practice to provide ground for software development projects by tailoring IT solutions to specific business needs and defining their functional requirements and stages of implementation. It is comprised of many subprocesses that draw guidance from various enterprise architecture viewpoints.

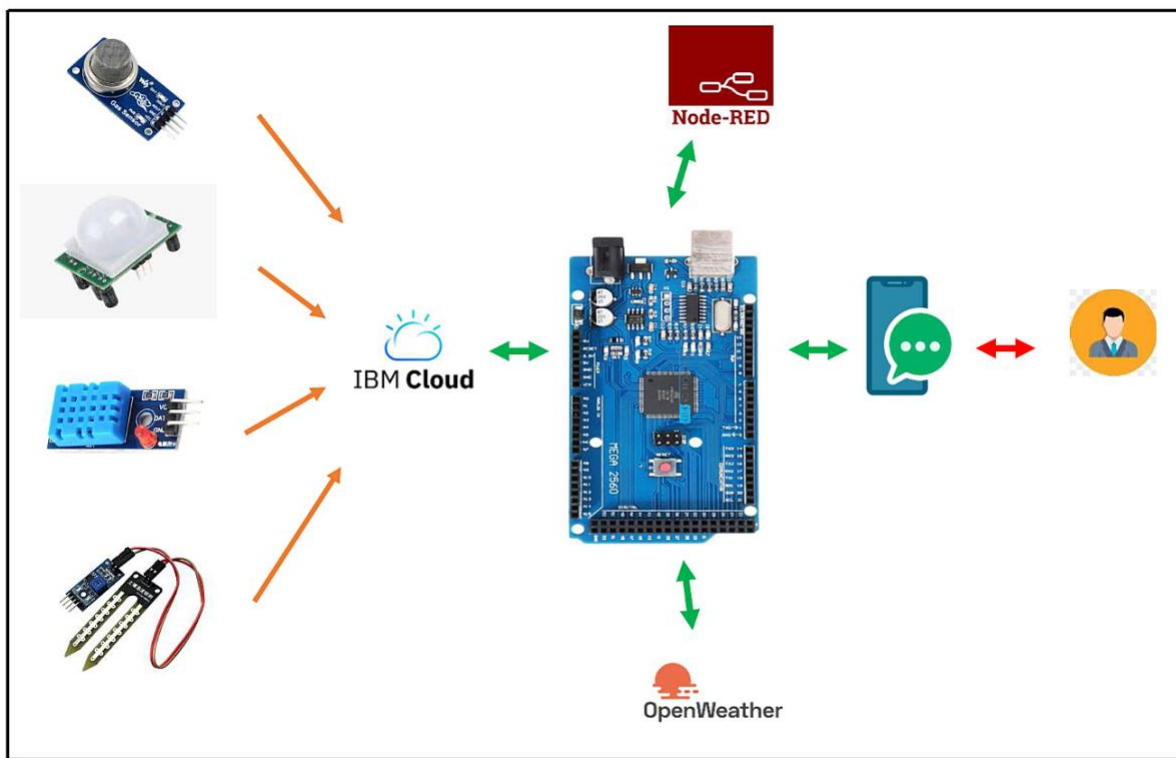
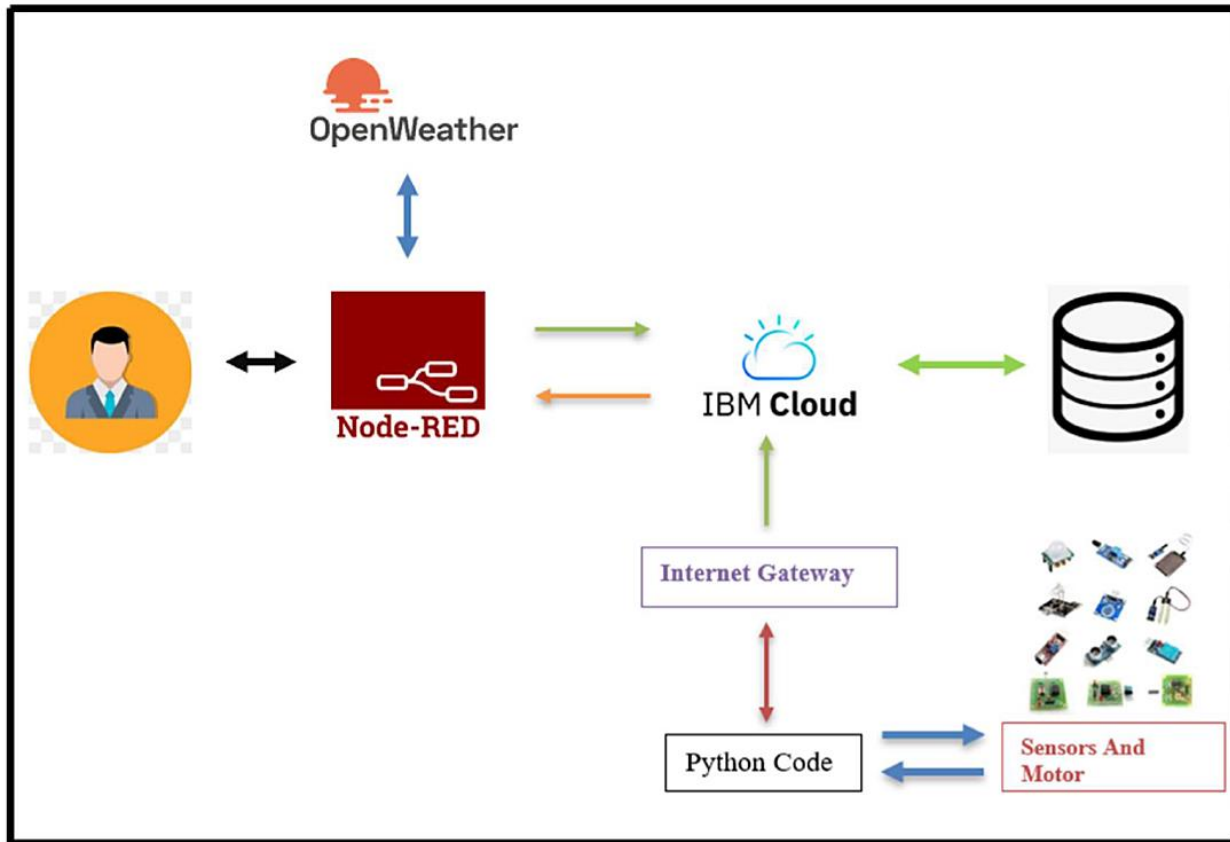


Fig 5.2.1 Solution Architecture Diagram

b. Technical Architecture

Technical Architecture (TA) is **a form of IT architecture that is used to design computer systems**. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.



5.3 User Stories

A user story is **an informal, general explanation of a software feature written from the perspective of the end user or customer**. The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer.

User Stories :

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard.	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm.	High	Sprint-1
		USN-3	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password		Medium	Sprint-1
Customer (Web user)	Dashboard	USN-5	Main Menu Is Apprised And Manages Modules.	Can View The Main Menu And Access The Mainframe.	High	Sprint-2
		USN-6	User Can Access All Sort Of Sensor Details And Access From All Over His Places.	Smart Farming Application.	High	Sprint-2
Administrator			User Can Manage And Access The Resources And Manages Using It.			Sprint-3

Fig 5.3.1 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint planning is a stage in Agile methodologies in which teams decide which tasks to complete in an upcoming sprint and how that work will be achieved. A sprint planning meeting is a meeting that is dedicated to planning the next sprint.

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Simulator Creator	USN-1	Create a Platform and Connect All The Sensors	2	High	Kushal Darira
Sprint-2	Software	USN-2	Device Creation in IBM Watson IOT Platform, Workflow Of IOT Scenarios using Node-Red	2	High	Kushal Darira Balaji
Sprint-3	MIT App Inventor	USN-3	Creating Virtual Application for Smart Farmer App Using MIT App Inventor	2	High	Kushal Darira Kamesh
Sprint-3	Dashboard	USN-3	Design Modules and Test the App	2	High	Kushal Darira Hari Krishna
Sprint-4	Web UI	USN-4	To Make the User To Interact With The Software	2	High	Kushal Darira

Fig 6.1.1 Sprint Planning

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		05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022		12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022		17 Nov 2022

Fig 6.1.2 Sprint Estimation

6.2 Sprint Delivery Schedule

The Development Team are accountable for the successful delivery of each Sprint Increment. No one single developer is ever individually accountable for that. There is no “delivery manager” role in

Scrum to be held accountable for anything concerning development, or to be held accountable.

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		05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022		12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022		17 Nov 2022

Fig 6.2.1 Sprint Delivery Plan

6.3 Reports from JIRA

Log in to continue - Log in with | virtual eye - Roadmap - Jira

https://abcd1234ef.atlassian.net/jira/software/projects/VE/boards/1/roadmap

Booking.com | b9 - Google Drive | Gmail | YouTube | Maps | Utomik Games | Booking.com

Other favorites

Your work | Projects | Filters | Dashboards | People | Apps | Create

Q Search

virtual eye
Software project

PLANNING

Roadmap

Board

DEVELOPMENT

Code

Project pages

Add shortcut

Project settings

You're in a team-managed project
Learn more

Projects / virtual eye
Roadmap

Give feedback | Share | Export

Search | KR | Status category | Epic

OCT | NOV | DEC

VE-1 sprint

VE-7 VE-2 sprint 2

VE-8 cloudant DB DONE

VE-9 VE-3 sprint-3

VE-11 coding TO DO

VE-10 VE-4 sprint 4

VE-12 applications building TO DO

Today | Weeks | Months | Quarters

27°C
Polluted air

Windows taskbar icons: File Explorer, Edge, Teams, Outlook, OneDrive, etc.

System tray: ENG IN, 20:55, 18/11/2022

Jira Software | Your work | Projects | Filters | Dashboards | People | Apps | Create

VirtualEye - Life Guard...
Software project

PLANNING

Roadmap

Backlog

Board

Reports

DEVELOPMENT

Code

Project pages

Add shortcut

Project settings

You're in a team-managed project
Learn more

Projects / VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning
Roadmap

Search | Status category | Epic

NOV

Sprints

VLGFSPSTD Sprint 1

VLGFSPSTD Sprint 2

VLGFSPSTD Sprint 3

VLGFSPSTD Sprint 4

+ Create Epic

Today | Weeks

Log in to continue - Log in with | VE board - Agile board - Jira | +

https://abcd1234ef.atlassian.net/jira/software/projects/VE/boards/1

Booking.com | b9 - Google Drive | Gmail | YouTube | Maps | Utomik Games | Booking.com | Other favorites

virtual eye
Software project

PLANNING

- Roadmap
- Board

DEVELOPMENT

- Code
- Project pages
- Add shortcut
- Project settings

You're in a team-managed project
Learn more

Projects / virtual eye

VE board

Search KR + Epic

GROUP BY None

TO DO 2 ISSUES

- login
SPRINT
VE-5
- coding
VE-3 SPRINT-3
VE-11

+ Create issue

IN PROGRESS

+ Create issue

DONE 4 ISSUES ✓

- cloudant DB
VE-2 SPRINT 2
VE-8
- registration
SPRINT
VE-3
- user confirmation

27°C Mostly clear

21:03 18/11/2022

Jira Software | Your work | Projects | Filters | Dashboards | People | Apps | Create

Search

VirtualEye - Life Guard...
Software project

PLANNING

- Roadmap
- Backlog
- Board
- Reports

DEVELOPMENT

- Code
- Project pages
- Add shortcut
- Project settings

You're in a team-managed project
Learn more

Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning

Backlog

Search Epic

Insights

Epic

Issues without epic

- Sprint-1
- Sprint-2
- Sprint-3
- Sprint-4

+ Create Epic

VLGFSPTD Sprint 1 7 Nov – 21 Nov (3 issues) 3 0 0 Complete sprint

- VLGFSPTDAD-11 Registration TO DO
- VLGFSPTDAD-12 User conformation TO DO
- VLGFSPTDAD-13 Login TO DO

+ Create issue

VLGFSPTD Sprint 2 21 Nov – 5 Dec (1 issue) 1 0 0 Start sprint

- VLGFSPTDAD-14 Cloudant DB TO DO

+ Create issue

Quickstart

7. CODING & SOLUTIONING

7.1 Arduino Code (Python)

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

#Provide your IBM Watson Device Credentials
organization = "157uf3"
deviceType = "abcd"
deviceId = "7654321"
authMethod = "token"
authToken = "87654321"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
        elif
    status == "motoroff":
        print ("motor is off")
    else :
        print ("please send proper command")
    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
    temp=random.randint(90,110)
    Humid=random.randint(60,100)
    Mois=random. Randint(20,120)
    data = { 'temp' : temp, 'Humid': Humid , 'Mois': Mois}
    #print data def myOnPublishCallback( ):
    print ("Published Temperature = %s C" % temp, "Humidity = %s %% " %Humid,
    "Moisture =%s deg c" % Mois "to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data,
    qos=0,on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoT")
    time.sleep(10)
    deviceCli.commandCallback = myCommandCallback
    # Disconnect the device and application from the cloud
    deviceCli.disconnect()
```

```

ibmiotpublishsubscribe.py - C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py (3.7.0)
File Edit Format Run Options Window Help

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

#Provide your IBM Watson Device Credentials
organization = "157uf3"
deviceType = "abcd"
deviceId = "7654321"
authMethod = "token"
authToken = "87654321"

# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
    elif status == "motoroff":
        print ("motor is off")
    else :
        print ("please send proper command")

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMe
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #.....

```

Ln: 22 Col: 21

```

Python 3.7.0 Shell
File Edit Shell Debug Options Window Help

Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py =====
2022-11-07 20:01:24,074  ibmiotf.device.Client      INFO      Connected successfully: d:157uf3:abcd:7654321
Published Moisture = 90 deg C Temperature = 96 C Humidity = 76 % to IBM Watson
Published Moisture = 102 deg C Temperature = 110 C Humidity = 68 % to IBM Watson
Published Moisture = 45 deg C Temperature = 99 C Humidity = 100 % to IBM Watson
Command received: motoron
motor is on
Published Moisture = 77 deg C Temperature = 91 C Humidity = 85 % to IBM Watson
Published Moisture = 73 deg C Temperature = 94 C Humidity = 86 % to IBM Watson
Command received: motoroff
motor is off
Published Moisture = 101 deg C Temperature = 104 C Humidity = 87 % to IBM Watson

```

7.2 Arduino Code (c++)

```
#include <WiFi.h>

#include <PubSubClient.h>

#include <ArduinoJson.h>

void setup() {

#define ORG "9g75cf"

#define DEVICE_TYPE "smart"

#define DEVICE_ID "sprint2"

#define TOKEN "<12345678>"

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

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

char publishTopic[] = "iot-2/evt/abcd_1/fmt/json";

char topic[] = "iot-2/cmd/home/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

PubSubClient client(server, 1883, WiFiClient);

void publishData();

const int trigpin=5;

const int echopin=18;

String command;

String data="";

String lat="14.167589";
```

```
String lon="80.248510";
```

```
String name="point2";
```

```
String icon="";
```

```
long duration;
```

```
int dist;
```

```
void setup()
```

```
{
```

```
  Serial.begin(115200);
```

```
  pinMode(trigpin, OUTPUT);
```

```
  pinMode(echopin, INPUT);
```

```
  wifiConnect();
```

```
  mqttConnect();
```

```
}
```

```
void loop() {
```

```
  publishData();
```

```
  delay(500);
```

```
  if (!client.loop()) {
```

```
    mqttConnect();
```

```
  }
```

```
}
```

```
void wifiConnect() {
```

```
Serial.print("Connecting to ");  
  
Serial.print("Wifi");  
  
WiFi.begin("Wokwi-GUEST", "", 6);  
while (WiFi.status() != WL_CONNECTED) {  
    delay(500);  
    Serial.print(".");  
}  
  
Serial.print("WiFi connected, IP address: ");  
  
}  
  
Serial.println(WiFi.localIP());  
}  
  
  
void mqttConnect() {  
    if (!client.connected()) {  
        Serial.print("Reconnecting MQTT client to ");  
        Serial.println(server);  
        while (!client.connect(clientId, authMethod, token)) {  
            Serial.print(".");  
            delay(1000);  
        }  
        initManagedDevice();  
        Serial.println();  
    }  
}
```

```

void initManagedDevice() {
    if (client.subscribe(topic)) {
        Serial.println(client.subscribe(topic));
        Serial.println("subscribe to cmd OK");
    }
    else {
        Serial.println("subscribe to cmd FAILED");
    }
}

void publishData()
{
    digitalWrite(trigpin,LOW);
    digitalWrite(trigpin,HIGH);
    delayMicroseconds(10);
    digitalWrite(trigpin,LOW);
    duration=pulseIn(echopin,HIGH);
    dist=duration*speed/2;
    if(dist<100){
        dist=100-dist;
        icon="fa-trash";
    }else{
        dist=0;
        icon="fa-trash-o";
    }

    DynamicJsonDocument doc(1024);

```

```
String payload;

doc["Name"]=name;

doc["Latitude"]=lat;

doc["Longitude"]=lon;

doc["Icon"]=icon;

doc["FillPercent"]=dist;

serializeJson(doc, payload);

delay(3000);

Serial.print("\n");

Serial.print("Sending payload: ");

Serial.println(payload);

if (client.publish(publishTopic, (char*) payload.c_str())){

    Serial.println("Publish OK");

    } else {

        Serial.println("Publish FAILED");

    }

}
```

IBM E??c\$SMART FARMER 2.0

```
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
Moisture : 1024 Humidity = 70% temperature = 30C Sent Data To IBM Watson
MOTOR ON
```

```
ROUGH_2 [Arduino 1.8.19 (Windows Store 1.8.37.0)]
File Edit Sketch Tools Help

ROUGH_2
#include <WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
void setup() {

#define ORG "9q75cf"
#define DEVICE_TYPE "smart"
#define DEVICE_ID "sprint2"
#define TOKEN "<12345678>"
//----- Customise the above values -----

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/abod_1/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, WiFiClient);
void publishData();

const int trigpin=5;
const int echopin=10;
String command;
String data="";
String lat="14.167589";
String lon="80.248510";
String name="point2";
String icon="";

long duration;
int dist;

void setup()
{
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
```


7.3 NODE – RED

Node-RED is a **programming tool for wiring together hardware devices, APIs and online services in new and interesting ways**. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

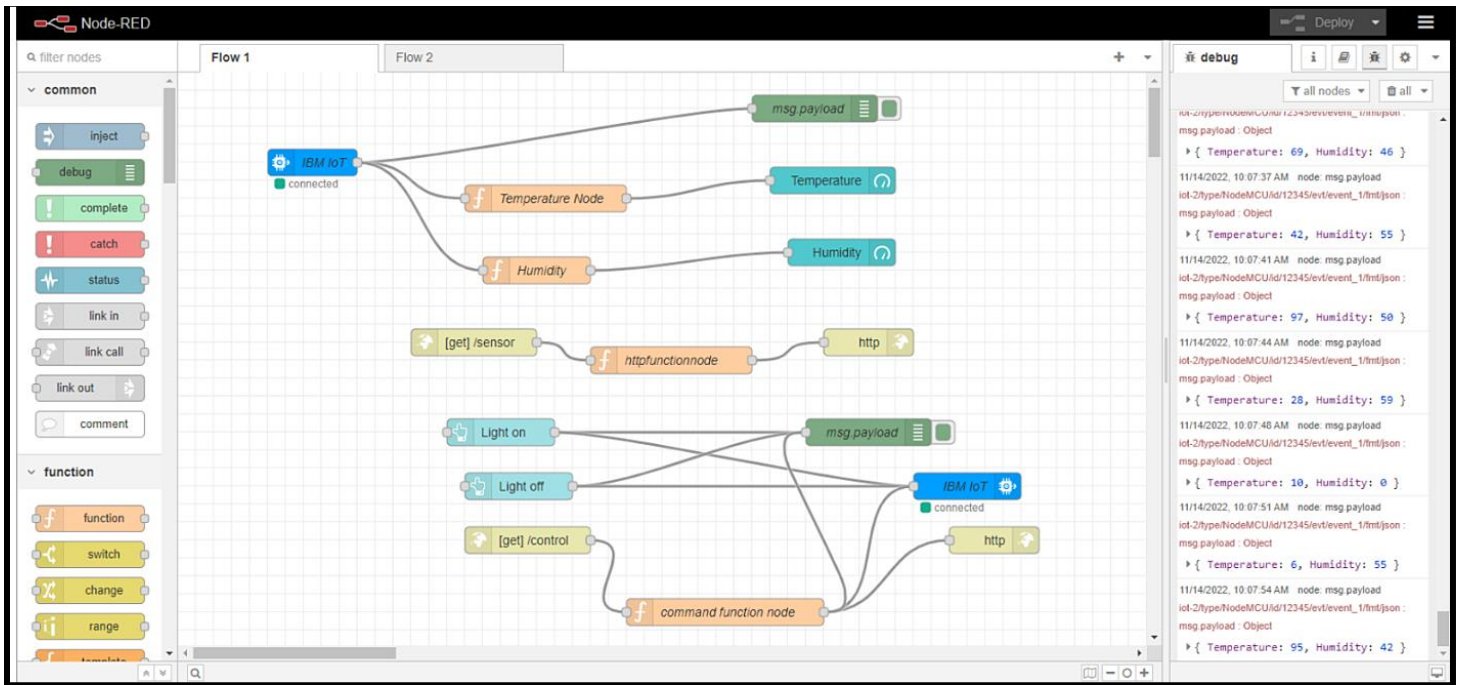


Fig 7.3.1 Build A Web Application In Node – Red .

Node-RED consists of a Node.js based runtime that you point a web browser at to access the flow editor. Within the browser you create your application

by dragging nodes from your palette into a workspace and start to wire them together.

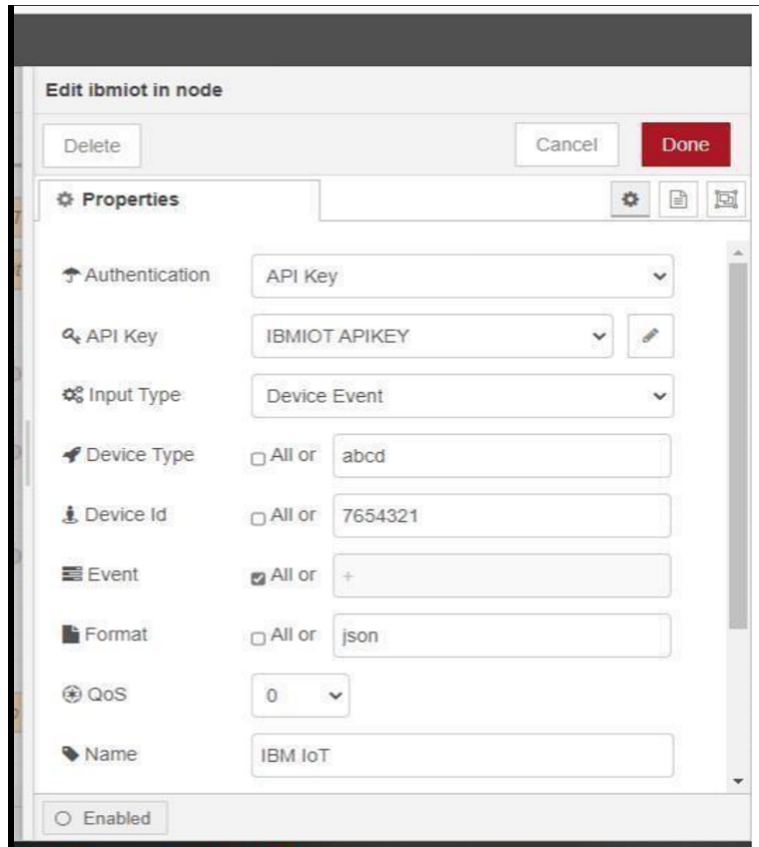


Fig 7.3.2 Build Using API Key .

1. We Added Two Buttons In The Application In Order To Access The Motor

1 = Motor On

0 = Motor Off

2. We Used A Function Node To Analyses And Receive And Analysis Of Data

The Node – Red Code :

```
if(msg.payload===1)
```

```
    msg.payload={"command":"ON"};
```

```
else if(msg.payload==0)
```

```
    msg.payload={"command":"OFF"};
```

JSON UI Interface:

1. In order To Use A Virtual Environment We Can Develop A Platform In Node – Red In Order To Show The Gauges, Text And Buttons To Control The Equipments

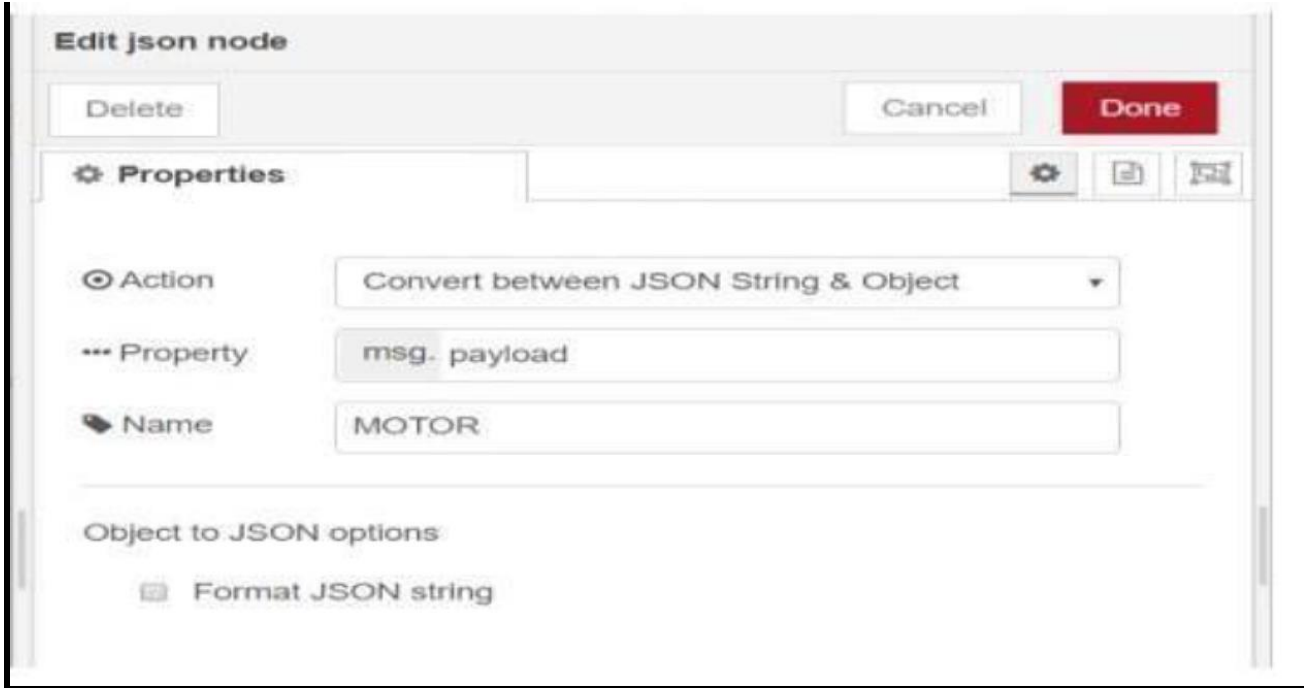


Fig 7.3.3 Editing Json File For Motor.



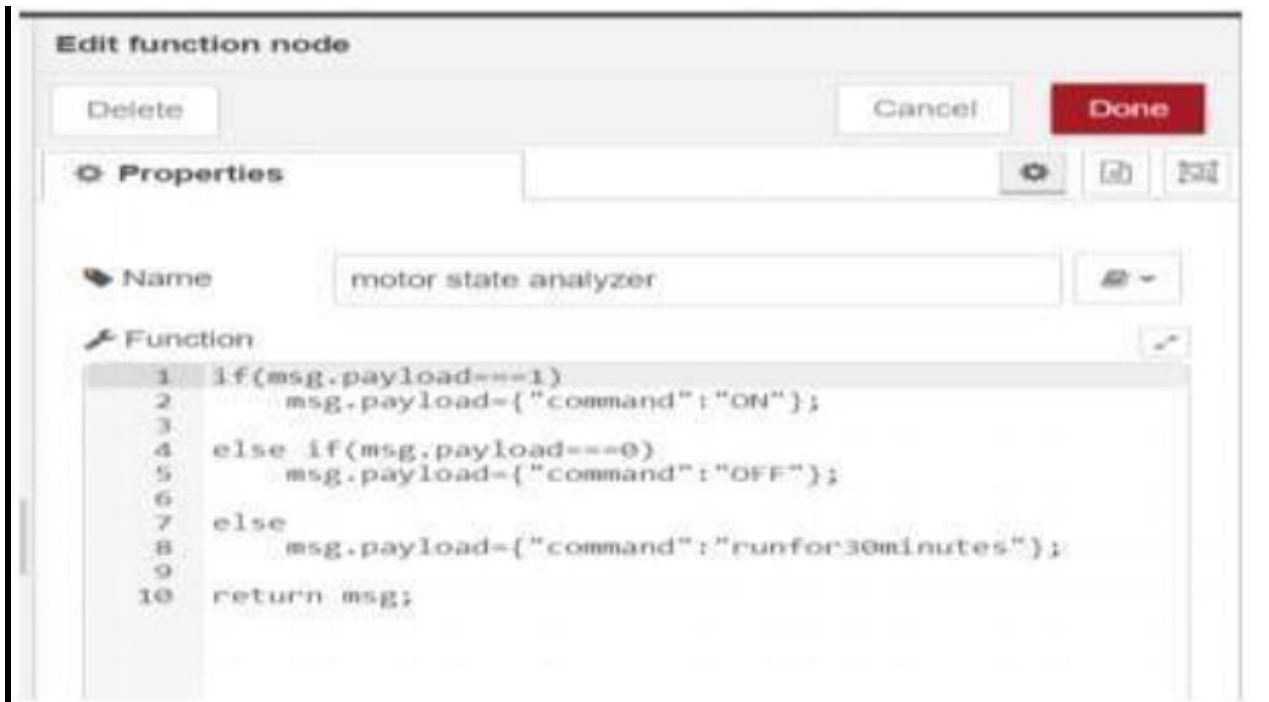


Fig 7.3.5 Message Payload For Motor On/Off

Web User Interface :

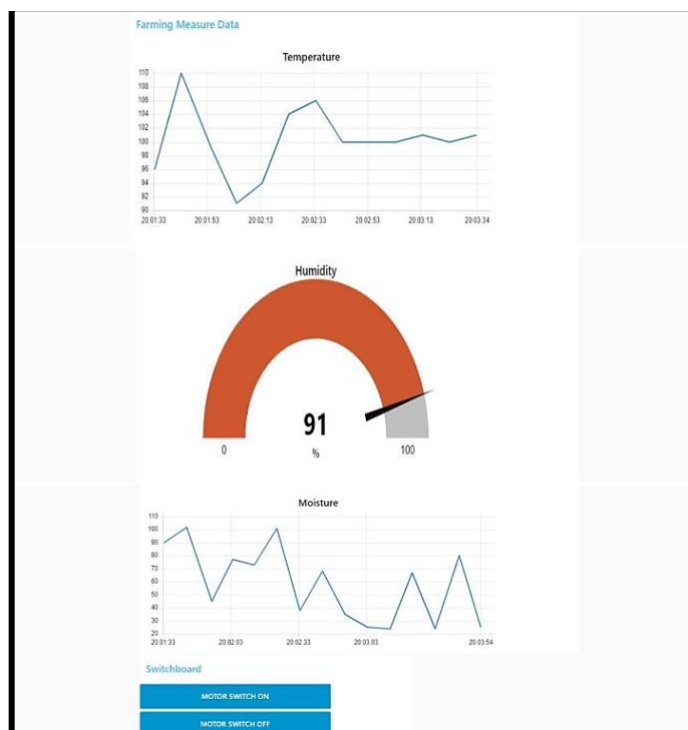


Fig 7.3.6 User Interface In Node - Red

7.4 IBM Cloud IOT Watson :

IBM Watson IoT Platform is a **fully managed, cloud-hosted service** that makes it simple to derive value from Internet of Things (IoT) devices.

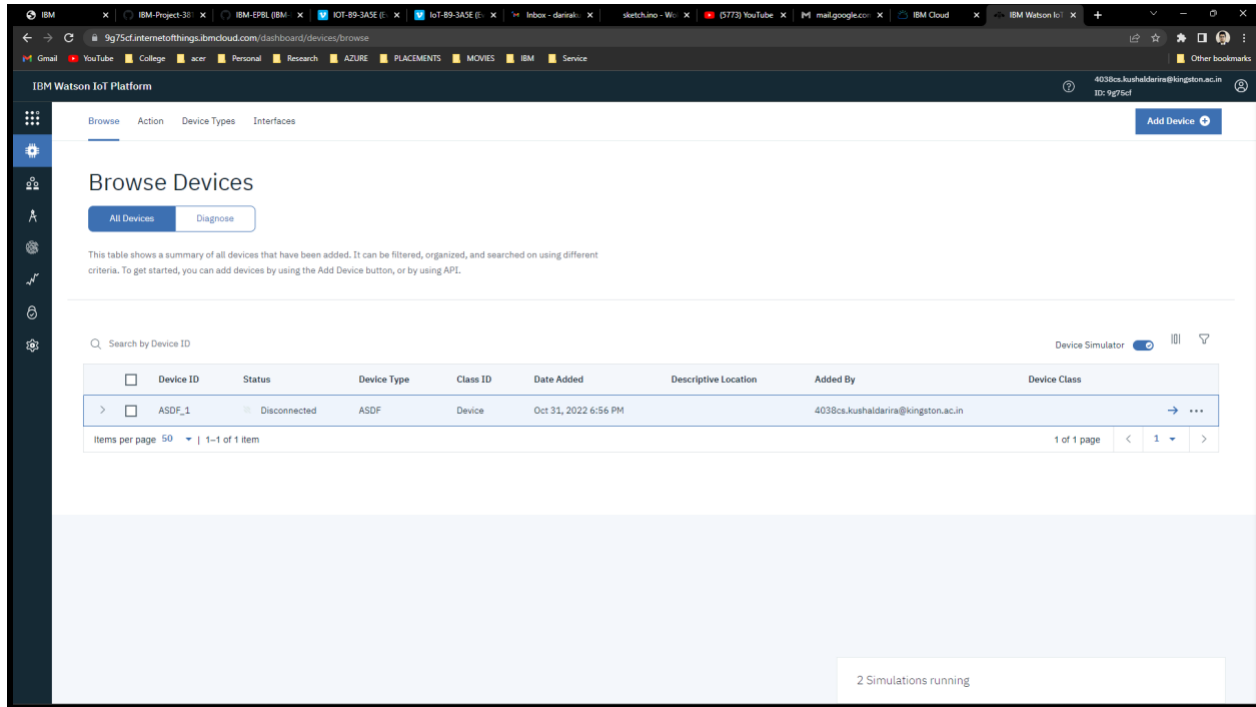


Fig 7.4.1 IBM Watson Platform

Recent Events

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

Event	Value	Format	Last Received
Sensor	{"temp":15,"hum":78,"moist":66}	json	a few seconds ago
Sensor	{"temp":41,"hum":90,"moist":15}	json	a few seconds ago
Sensor	{"temp":54,"hum":29,"moist":61}	json	a few seconds ago
Sensor	{"temp":46,"hum":38,"moist":77}	json	a few seconds ago
			1 Simulation running

Fig 7.4.2 Results For Program

8. Testing :

8.1 Test Cases :

Testing Black Boxes :

Black box testing involves testing a system with no prior knowledge of its internal

workings. A tester provides an input, and observes the output generated by the system under test.

No	Scenario	Test Results	Security	Conclusion
1	Provides A Power Source	Provides a Power Source	Read Automatically The Value Of The Gas Content In The room	Valid
2	Brings A Moisture And DHT 11 Data	Brings The Moisture And Temperature And Humidity	The Value Of Moisture And Log Credentials Are Verified	Valid

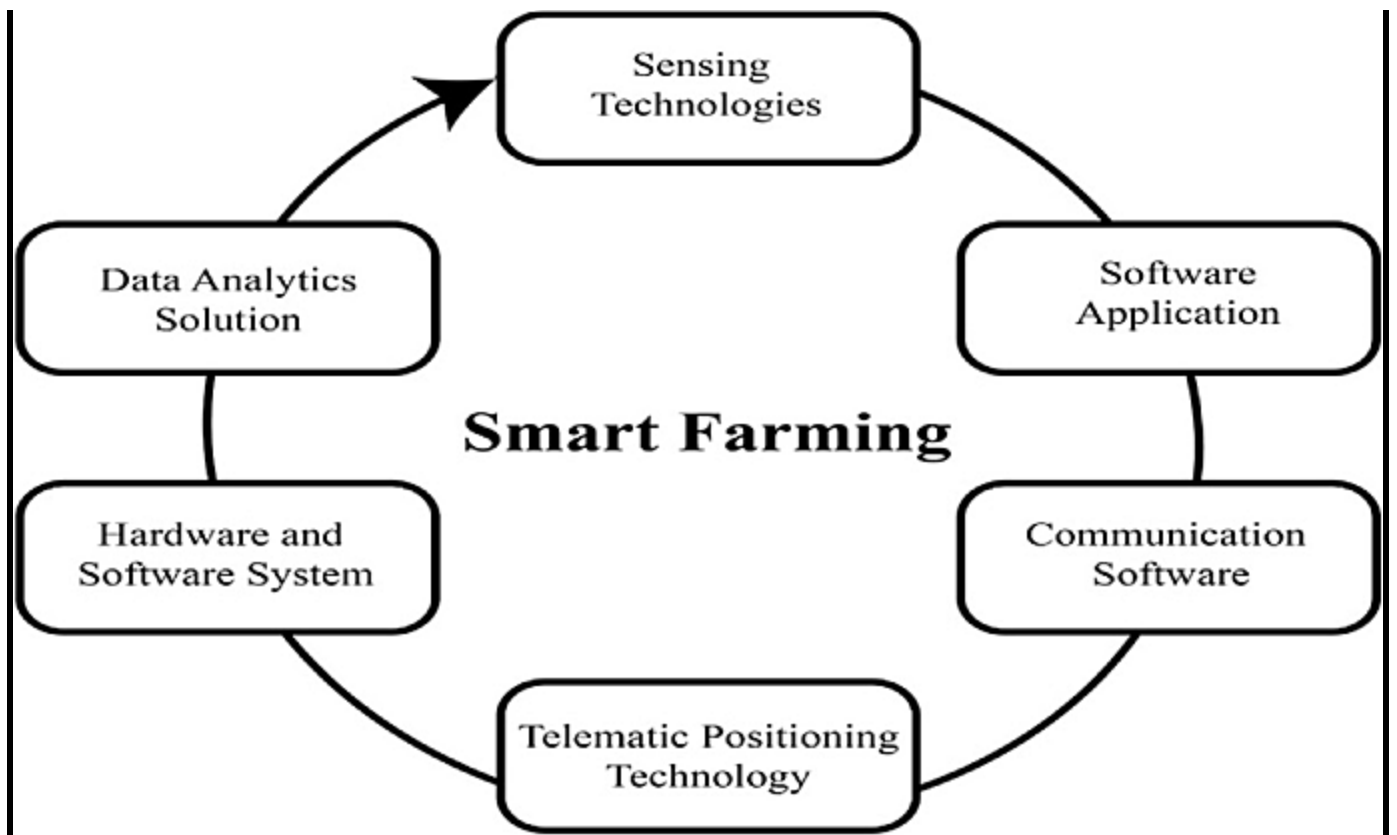
Testing The Moisture And Temperature Data

Analog Data	Moisture Light	Motor	App	Explanation
1	<=400>=1024	off	On	All The Data's Are Sent To the App.

2	≥ 400	On	On	Motor Is On And Data's Are Sent To The App.

8.2 User Acceptance Testing :

User Acceptance Testing (UAT), also known as beta or end-user testing, is defined as **testing the software by the user or client to determine whether it can be accepted or not**. This is the final testing performed once the functional, system and regression testing are completed.



10. ADVANTAGES & DISADVANTAGES

10.1 ADVANTAGES OF SMART FARMING

1. *Automatic adjustment of farming equipment made possible by linking information like crops/weather and equipment to auto-adjust temperature, humidity, etc.*

2. *In large farmland, Internet of Things equipped drone helps to receive the current state of crops and send the live pictures of farmland.*
3. *Analyzing farmland from the land using its Solutions you will know the current situation of fields and crops in.*
4. *86% of the studied farmers use some kind of “precision farming”.*
5. *95% acknowledged that “precision farming” is very helpful to use.*
6. *70% plan to expand their usage of "precision farming technologies".*

10. ADVANTAGES & DISADVANTAGES

10.1 ADVANTAGES OF SMART FARMING

1. *Automatic adjustment of farming equipment made possible by linking information like crops/weather and equipment to auto-adjust temperature, humidity, etc.*

2. *In large farmland, Internet of Things equipped drone helps to receive the current state of crops and send the live pictures of farmland.*
3. *Analyzing farmland from the land using its Solutions you will know the current situation of fields and crops in.*
4. *86% of the studied farmers use some kind of “precision farming”.*
5. *95% acknowledged that “precision farming” is very helpful to use.*
6. *70% plan to expand their usage of "precision farming technologies".*

11. 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.

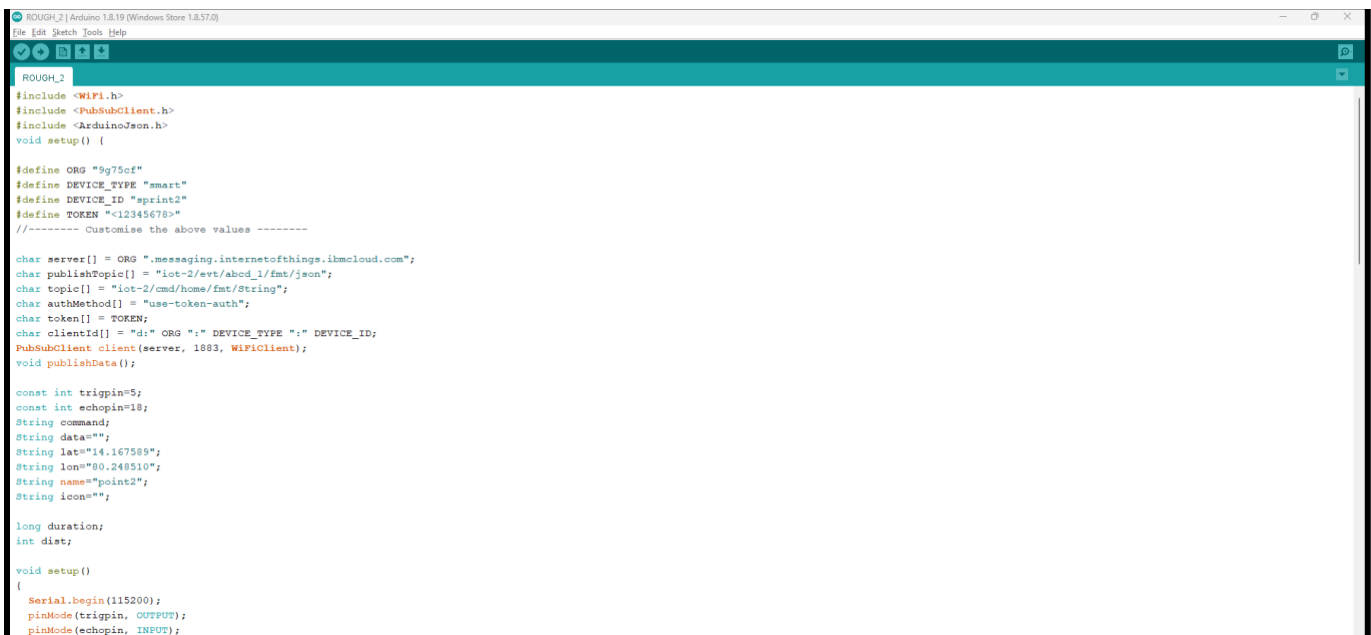
12. FUTURE SCOPE

Future work would be focused more on increasing sensors on this system. Through collecting data from sensors using IoT devices, you will learn about the real-time state of your crops. The future of IoT in agriculture **allows predictive analytics to help you make better harvesting decisions.**

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.

13. APPENDIX

a. Source Code



```
ROUGH_2 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

ROUGH_2
#include <WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
void setup() {

#define ORG "9g75cf"
#define DEVICE_TYPE "smart"
#define DEVICE_ID "sprint2"
#define TOKEN "<12345678>"
//----- Customise the above values -----

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/abcd_1/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/string";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, WiFiClient);
void publishData();

const int trigpin=5;
const int echopin=10;
String command;
String data="";
String lat="14.167585";
String lon="80.248510";
String name="point2";
String icon="";

long duration;
int dist;

void setup()
{
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
}
```

```
ROUGH_2 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

ROUGH_2
}

void loop() {
  publishData();
  delay(500);
  if (!client.loop()) {
    mqttConnect();
  }
}

void WifiConnect() {
  Serial.print("Connecting to ");
  Serial.print("Wifi");
  WiFi.begin("Wokwi-GUEST", "");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.print("WiFi connected, IP address: ");
  Serial.println(WiFi.localIP());
}

void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to ");
    Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(1000);
    }
    initManagedDevice();
    Serial.println();
  }
}

void initManagedDevice() {
  if (client.subscribe(topic)) {
    Serial.println(client.subscribe(topic));
  }
}

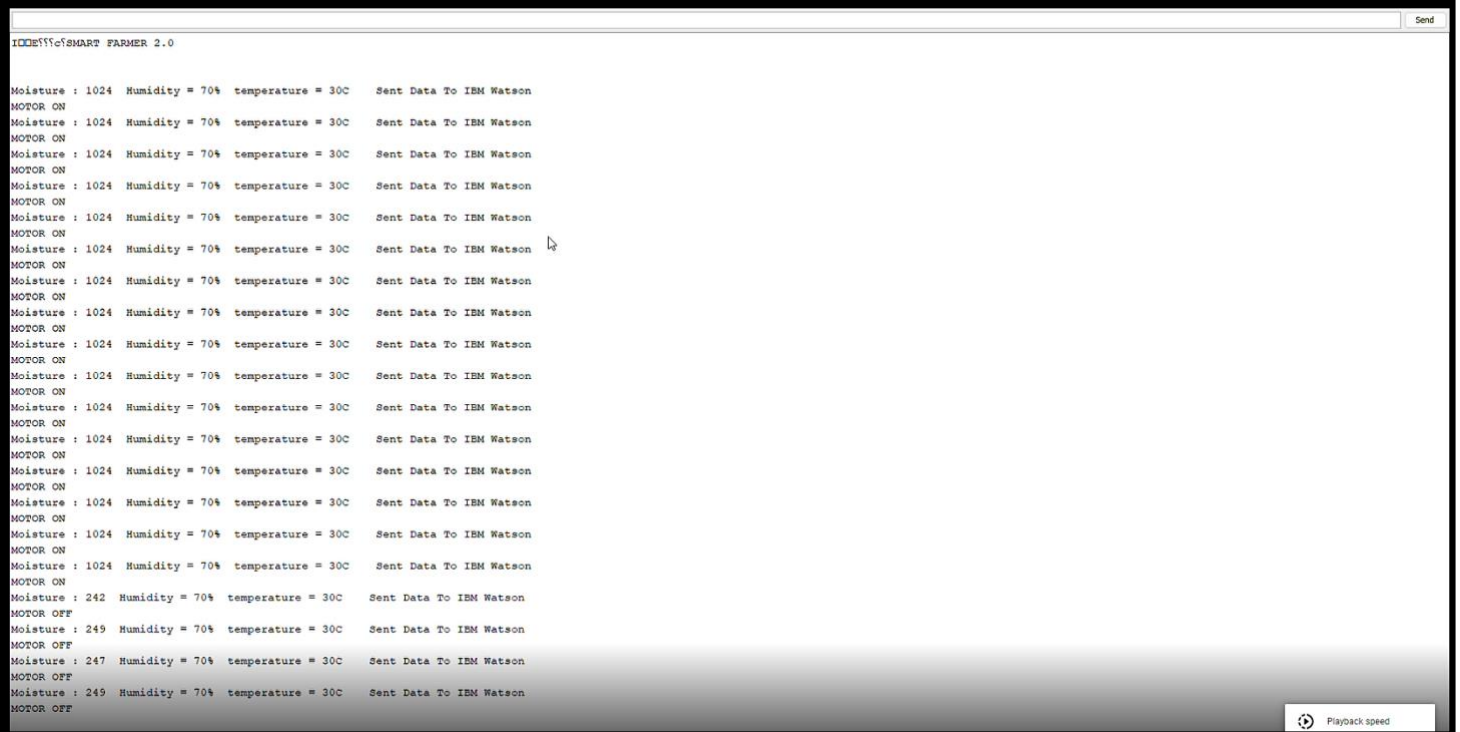
NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Disabled (new abort on oom), Disabled, All SSL cipher(s) (most compatible), 32kB cache + 32kB RAM (balanced), Use pgm_read macros for IRAM/PROGMEM, 4MB (FS/2MB OTA~1019KB), 2, v2, Lower Memory, Disabled, None, Only Sketch, 115200 on COM5
09:56 AM 19-11-2022
```

```
ROUGH_2 | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

ROUGH_2g
}
else {
  Serial.println("subscribe to cmd FAILED");
}
}

void publishData()
{
  digitalWrite(trigpin, LOW);
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin, LOW);
  duration=pulseIn(echopin, HIGH);
  dist=duration*speed/2;
  if (dist<100) {
    dist=100-dist;
    icon="fa-trash";
  } else {
    dist=0;
    icon="fa-trash-o";
  }
  DynamicJsonDocument doc(1024);
  String payload;
  doc["Name"]=name;
  doc["Latitude"]=lat;
  doc["Longitude"]=lon;
  doc["Icon"]=icon;
  doc["FillPercent"]=dist;
  serializeJson(doc, payload);
  delay(3000);
  Serial.print("\n");
  Serial.print("Sending payload: ");
  Serial.println(payload);
  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish OK");
  } else {
    Serial.println("Publish FAILED");
  }
}
}
```

OUTPUT SCREEN



b. GitHub & Project Demo Link

Project Demo Link :

