PROJECT REPORT

Project Name: SMARTFARMER- IOT ENABLED SMART FARMING APPLICATION

Team ID:PNT2022TMID53765

Team:

Sanjay T

Saran A

Saravanakumar D

Siva sudharsanam K

INDEX

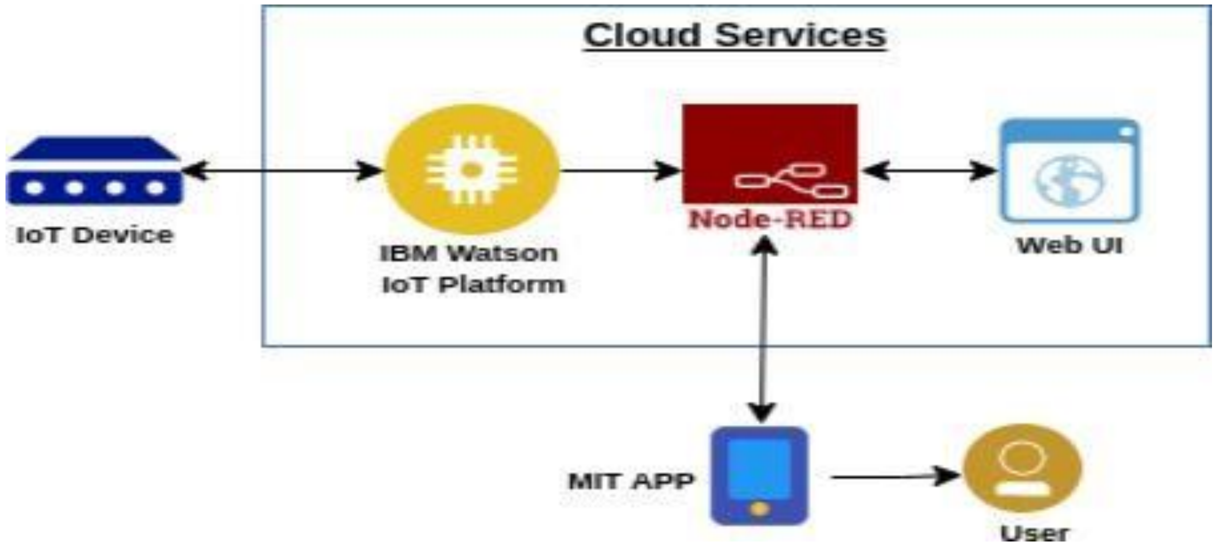
GitHub & Project Demo Link

SMART FARMING

1.**INTRODUCTION:**

1.1 PROJECT OVERVIEW:

- IoT-based agriculture system helps the farmer in monitoring different parameters of his field like soil moisture, Temperature, humidity using some sensors.
- This is system that enables framers to monitor and control their forms with a web-based application build with Node-RED.
- It uses the IBM IOT Watson cloud platform as its Backend.



1.2 PURPOSE:

Smart Farming reduce the ecological foot print 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.

2. LITERATURE SURVEY:

2.1 EXISTING PROBLEM:

The biggest challenges faced by IoT in the agricultural sector are lack of information, high adoption costs, and security concers etc. Most of the farmers are not aware of the implementation of IoT in agriculture.

To successfully deploy a smart agriculture system, consider setting up a communications network that can integrate a limited number of sensors across a large area of farmland. This will require third-party network provisioning or setting up a private network consisting of access points and uplinks to a private backhaul network, which channels all the data traffic to centralized monitoring software or an analytics head-end system • It is not a secure system.

- There is no motion detection for protection of agriculture field.
- Automation is not available.

2.2 REFERENCES:

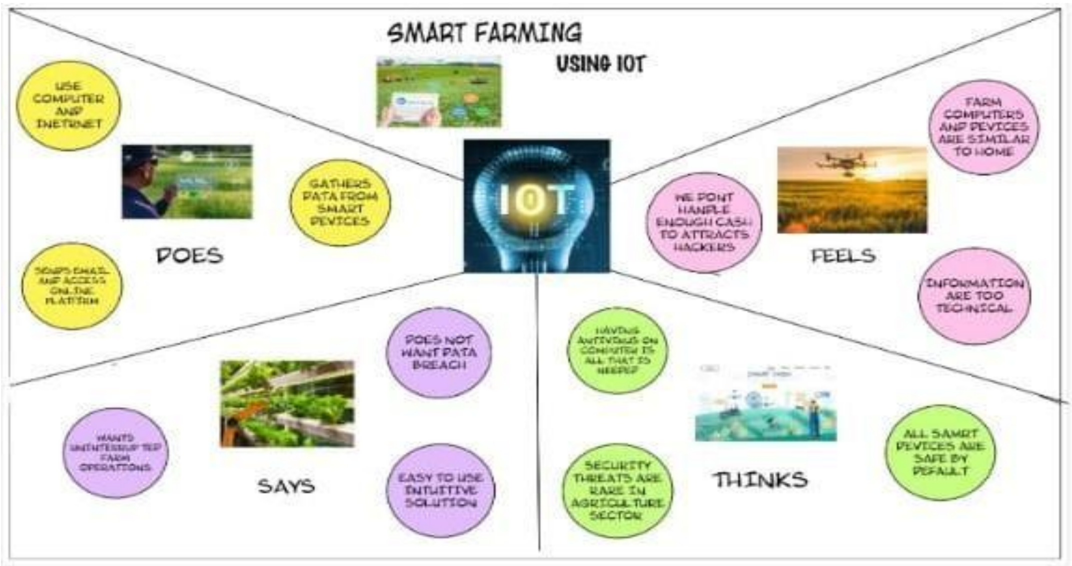
It is the application of modern ICT (Information and Communication Technologies) into agriculture. In IOT- based smart farming, a system is built for monitoring the crop field with the help of sensors (light, humidity, temperature, soil moisture, etc.). The farmers can monitor the field conditions from anywhere.

2.3 PROBLEM STATEMENT DEFINITION:

Overuse of pesticides and fertilizer in agricultural fields leads to destruction of the crop as well as reduces the efficiency of the field increasing the soil vulnerability toward pest. IoT applications may be used to update the farmer/user about type & quantity of pesticide required by the crop.

3. IDEATION & PROPOSED SOLUTION:

3.1 EMPATHY MAP CANVAS:



3.2 IDEATION AND BRAINSTORMING:

Introduction on Internet of Things (IoT), application of IoT in agricultural field to improve the yield and quality by reducing the cost is provided. The sensors which are used in the architecture are discussed briefly and the process of transmission of data from the agriculture field to the central system is explained. The proposed system advantages are included. In addition, open research issues, challenges, and future of IoT in agricultural field are highlighted. The concept is basically developed on an idea, where there are numerous things or objects - such as Arduino, sensors, GSM models, LCD display, etc., that are connected with the Internet. Each of the objects has a different address and is able to interact with other items. The things or objects co-operate with each other to reach a common goal.

We are going to construct a smart agricultural monitoring system which can collect crucial agricultural data and send it to an IoT platform called Thing speak in real time where the data can be logged and analyzed. The logged data on Thing speak is in graphical format, a botanist or a reasonably knowledged farmer can analyze the data (from anywhere in the world) to make sensible changes in the supplied resources (to crops) to obtain high quality yield.

Smart agriculture monitoring system or simply smart farming is an emerging technology concept where data from several agricultural fields ranging from small to large scale and its surrounding are collected using smart electronic sensors. The collected data are analyzed by experts and local farmers to draw short term and long-term conclusion on weather pattern, soil fertility, current quality of crops, amount of water that will be required for next week to a month etc.

We can take smart farming a step further by automating several parts of farming, forexample smart irrigation and water management. We can apply predictive algorithms on microcontrollers or SoC to calculate the amount of water that will be required today for a particular agriculture field. Say, if there was rain yesterday and the quantity of water required today is going to be less. Similarly, if humidity was high the evaporation of water at upper ground level is going to be less, so water required will be less than normal, thus reducing water usage.

3.3 PRPOSED SOLUTION:

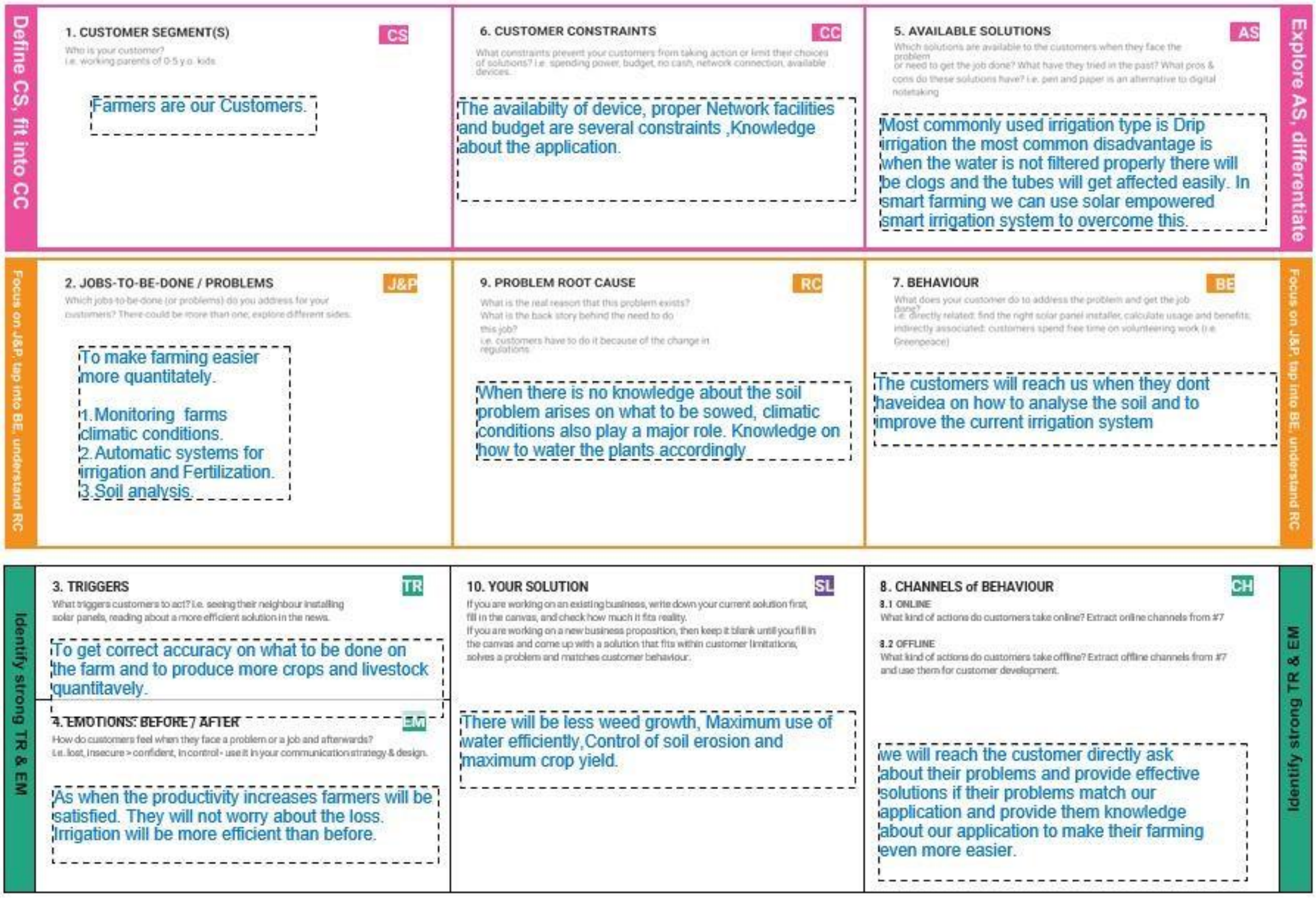
- To develop a Smart Agricultural System based on IOT which can give real time data and can help farmers in a very efficient manner.
- Soil Moisture can be checked by using the sensors that can sense the soil condition and send the data (moisture content in the soil) over the cloud services to the web application.

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To make farming easier by choosing several constraints in agriculture and to overcome those constraints, to increase production quality and quantity using IOT.
2.	Idea / Solution description	Using smart techniques like monitoring farms climate, smart irrigation and soil analysis.
3.	Novelty / Uniqueness	Solar power smart irrigation system which helps you to monitor temperature, moisture ,humidity using smart sensors.
4.	Social Impact / Customer Satisfaction	It is better than the present modern irrigation system by using this method we can control soil erosion. There will be better production yield.
5.	Business Model (Revenue Model)	As the productivity increases customer satisfaction also increases and hence need for the application also increases, which increases the revenue of the business.
6.	Scalability of the Solution	It is definetly scalable we ca increase the constraints when the problem arises.

3.4 PROBLEM SOLUTION FIT:

- With the help of the IoT devices, you can know the real-time status of the crops by capturing the data from sensors.
- Using predictive analytics, you can get an insight to make better decisions related to harvesting.



4.REQUIREMENT ANALYSIS:

4.1 FUNCTIONAL REQUIREMENTS:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	IoT devices	Sensors and Wifi module.
FR-2	Software	Web UI, Node-red, IBM Watson, MIT app

4.2 NON -FUNCTIONAL REQUIREMENTS:

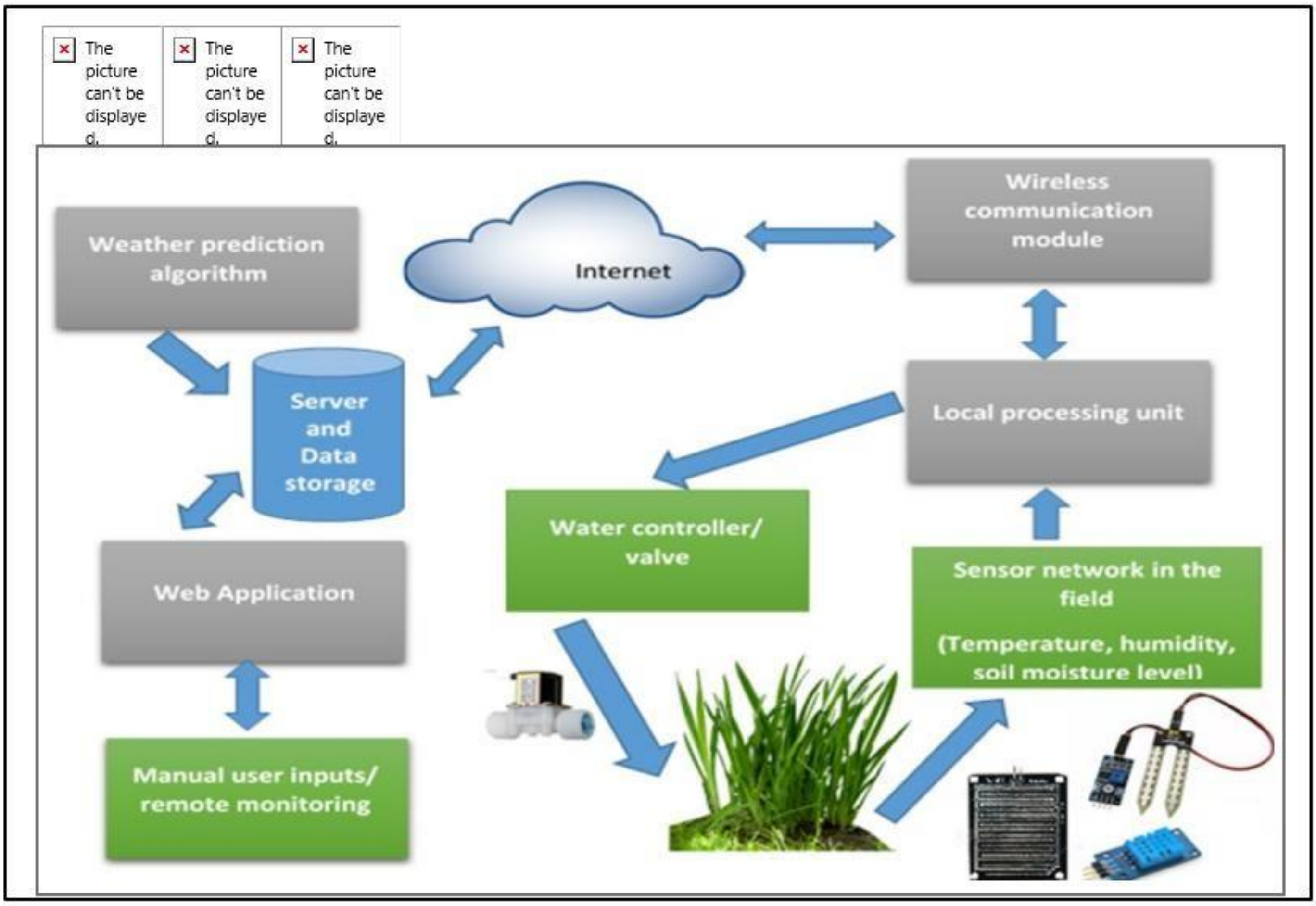
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Time consumability is less, Productivity is high.
NFR-2	**Security**	It has low level of security features due to integration of sensor data.
NFR-3	**Reliability**	Accuracy of data and hence it is Reliable.
NFR-4	**Performance**	Performance is high and highly productive.
NFR-5	**Availability**	With permitted network connectivity the application is accessible
NFR-6	**Scalability**	It is perfectly scalable many new constraints can be added

5 PROJECT DESIGN:

5.1 DATA FLOW DIAGRAM:

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



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	IoT devices	USN-1	Sensors and wi-fi module		High	Sprint-1
Customer	Software	USN-2	IBM Watson IoT platform, Workflows for IoT scenarios using Node-red		High	Sprint-2
Customer	MIT app	USN-3	To develop an application using MIT		High	Sprint-3
Customer	Web UI	USN-4	To make the user to interact with the software.	User can access the app for the services.	High	Sprint-4

5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

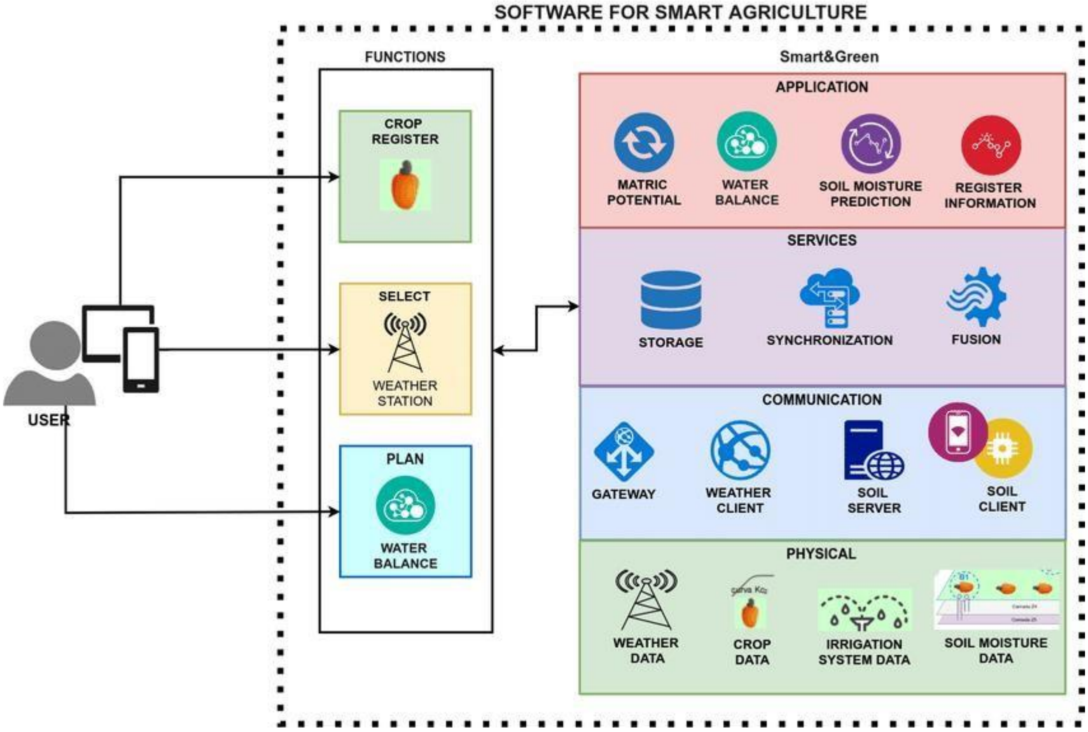


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	MIT app
2.	Application Logic-1	Logic for a process in the application	Node red/IBM Watson/MIT app
3.	Application Logic-2	Logic for a process in the application	Node red/IBM Watson/MIT app
4.	Application Logic-3	Logic for a process in the application	Node red/IBM Watson/MIT app
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM cloud.

7.	Temperature sensor	Monitors the temperature of the crop	
8.	Humidity sensor	Monitors the humidity	
9.	Soil moisture sensor (Tensiometers)	Monitors the soil temperature	
10.	Weather sensor	Monitors the weather	.
11.	Solar panel		.
12.	RTC module	Date and time configuration	
13.	Relay	To get the soil moisture data	

Table-2: Application Characteristics:

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

6.PROJECT PLANNING AND SCHEDULING :

6.1 SPRINT PLANNING AND ESTIMATION:

S.NO	ACTIVITY TITLE	ACTIVITY DESCRIPTION	DURATION
1	**Understanding the project**	Assign the team members after that create repository in the GitHub and then assign task to each member and guide them how to access the GitHub while submitting the assignments	1 week
2	**Staring The Project**	Team Members to Assign All the Tasks Based on Sprints and Work on It Accordingly.	1 week
3	**Completing Every Task**	Team Leader should ensure that whether every team member have completed the assigned task or not	1 week
4	**Stand Up Meetings**	Team Lead Must Have a Stand-Up Meeting with The Team and Work on The Updates and Requirement Session	1 week
5	**Deadline**	Ensure that team members are completing every task within the deadline	1 week
6	**Budget and Scope of project**	Analyze the overall budget which must be within certain limit it should be favorable to every person	1 week

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Simulation creation	USN-1	Connect Sensors and Arduino with python code	2	High	Sanjay.T Saran.A Saravanakumar.D Sivasudharsanam.k
Sprint-2	Software	USN-2	Creating device in the IBM Watson IoT platform, workflow for IoT scenarios using Node-Red	2	High	Sanjay.T Saran.A Saravanakumar.D Sivasudharsanam.k
Sprint-3	MIT App Inventor	USN-3	Develop an application for the Smart farmer project using MIT App Inventor	2	High	Sanjay.T Saran.A Saravanakumar.D Sivasudharsanam.k
Sprint-3	Dashboard	USN-3	Design the Modules and test the app	2	High	Sanjay.T Saran.A Saravanakumar.D Sivasudharsanam.k
Sprint-4	Web UI	USN-4	To make the user to interact with software.	2	High	Sanjay.T Saran.A Saravanakumar.D Sivasudharsanam.k

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

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	4 nov 2022	10 NOV 2022	20	10 Nov 2022
Sprint-2	20	6 Days	6 Oct 2022	12 Nov2022		12 Nov 2022
Sprint-3	20	6 Days	9 Nov 2022	15 Nov 2022		15 Nov 2022
Sprint-4	20	6 Days	12 Nov 2022	18 Nov 2022		18 Nov 2022

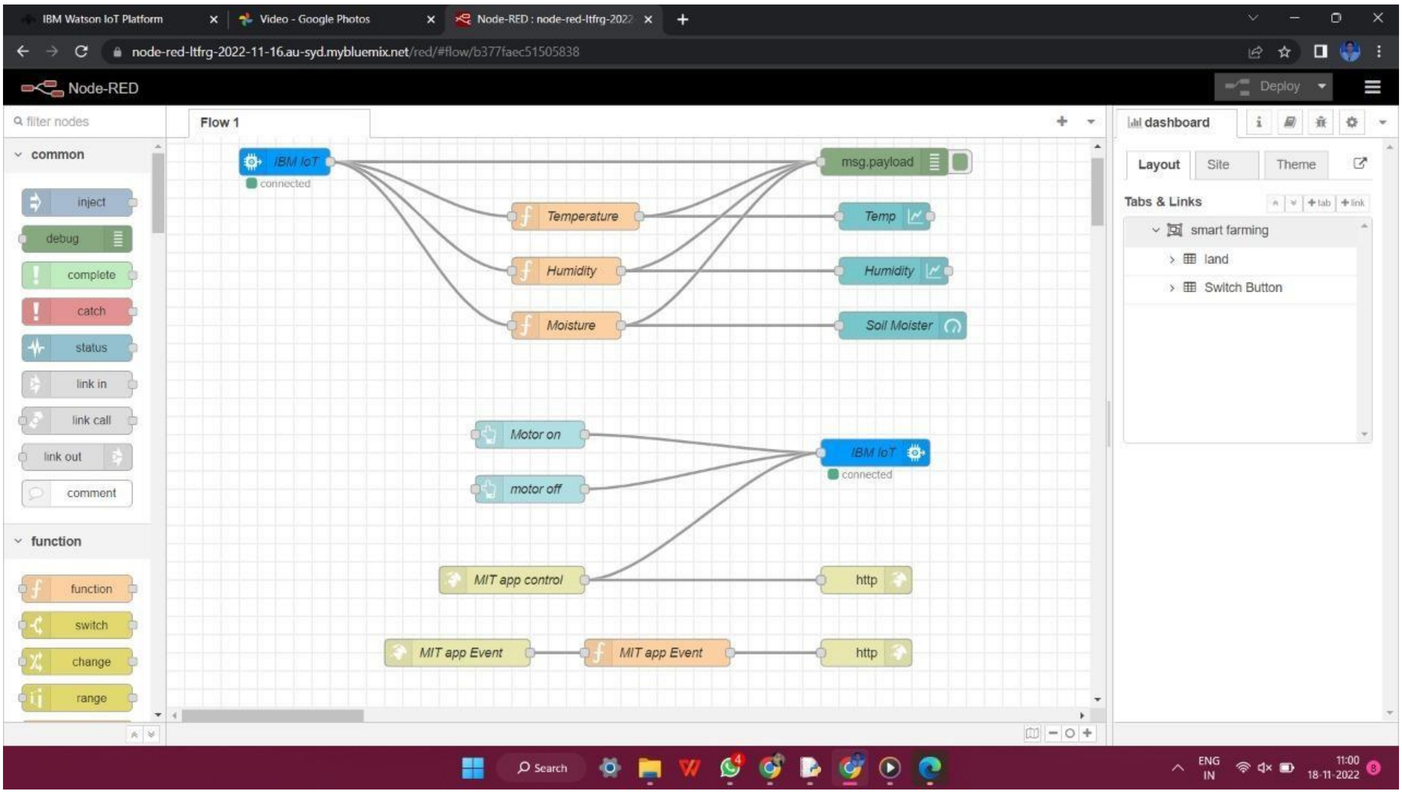
7. CODING AND SOLUTIONING:

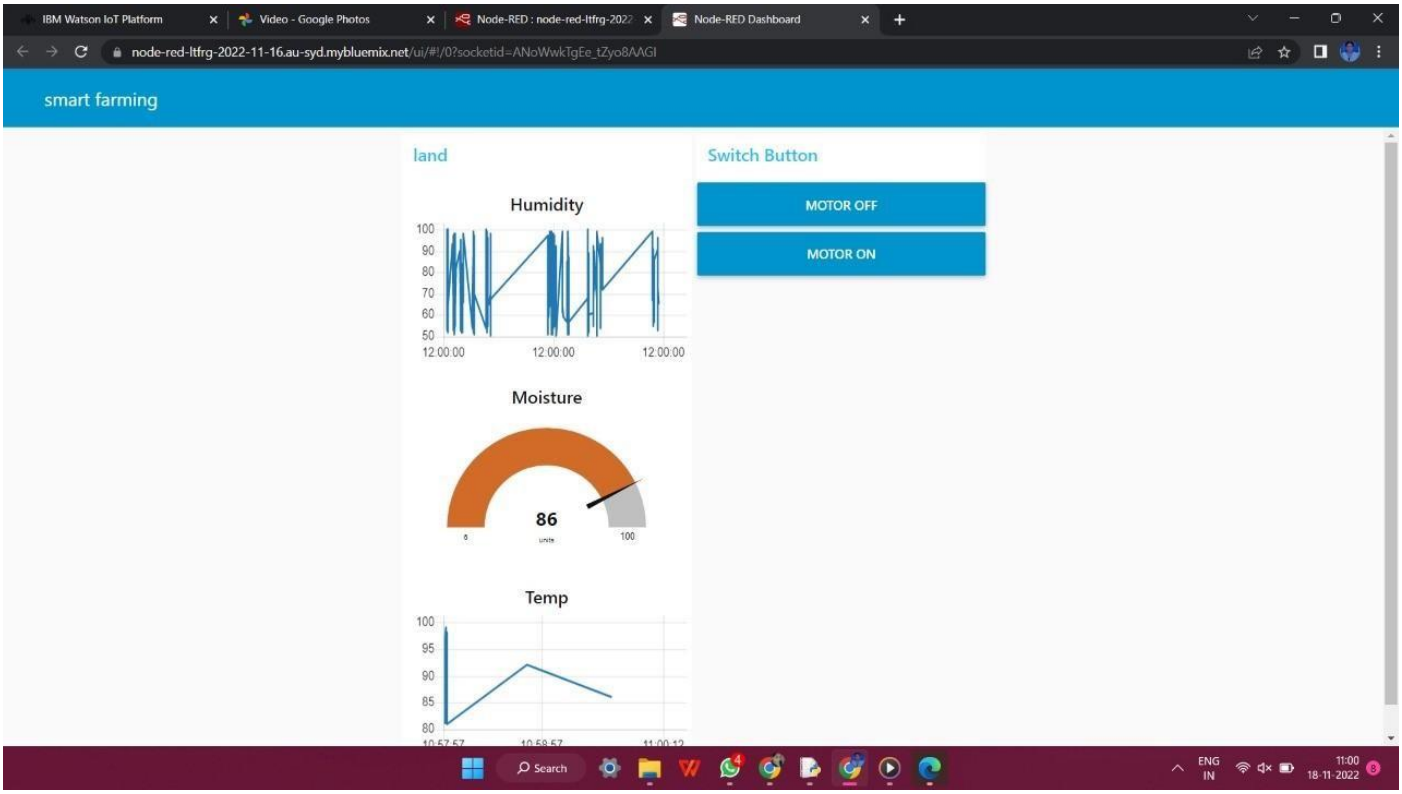
7.1 CODING:

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

#Provide your IBM Watson Device organization =
"r8cpvf" deviceType = "farming" deviceId = "12345"
authMethod = "token" authToken ="87654321" #
Initialize GPIO def myCommandCallback(cmd):
print("Commandreceived: %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
aneventof type "greeting" 10 times deviceCli.connect() while True:
#Get Sensor Data fromDHT11
   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 IoTF")
time.sleep(10) deviceCli.commandCallback =
myCommandCallback #Disconnect the device and
application from the cloud deviceCli.disconnect()
```

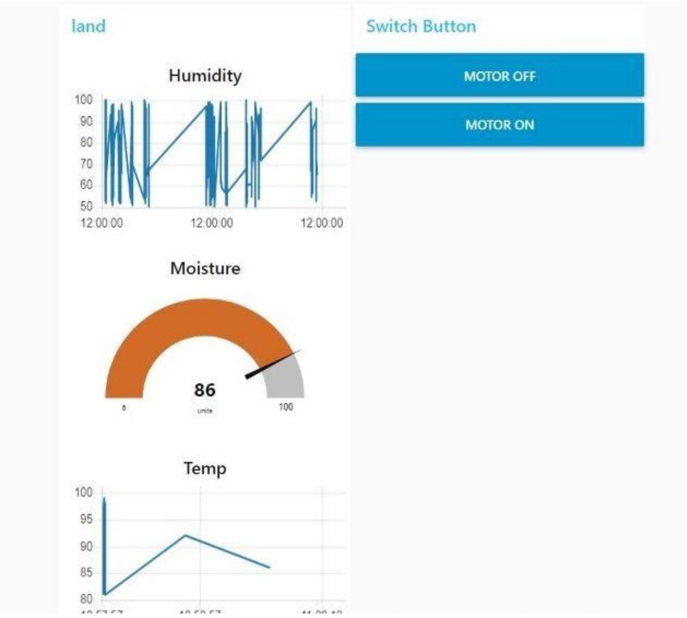
7.2 SOLUTION:


Node-RED
Flow 1
IBM IoT
connected
msg.payload
Temperature
Temp
Humidity
Humidity
Moisture
Soil Moister
Motor on
motor off
IBM IoT
connected
MIT app control
http
MIT app Event
MIT app Event
http
dashboard
Layout
Site
Theme
Tabs & Links
smart farming
land
Switch Button


smart farming
land
Switch Button
Humidity
MOTOR OFF
MOTOR ON
Moisture
86
Temp

8. RESULTS:

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



9. ADVANTAGES AND DISADVANTAGES:

9.1 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 extend
- 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.

9.2 DISADVANTAGES:

- Smart Agriculture requires internet connectivity continuously, but rural parts can not fulfill this requirement.
- IoT devices need much money to implement.

10. CONCLUSION:

An IOT based smart agriculture system using Watson IOT Platfrom , Watson simulator , IBM cloud and Node-RED.

11. FUTURE SCOPE :

In future due to more demand of good and more farming in less time, for betterment of the crops and reducing the usage of extravagant resources like electricity and water IoT can be implemented in most of the places.

12. APPENDIX :

12.1 SOURCE CODE:

```
import time import sys

import ibmiotf.application

import ibmiotf.device

import random


#Provide your IBM Watson Device organization =

"r8cpvf" deviceType = "farming" deviceId = "12345"

authMethod = "token" authToken ="87654321" #

Initialize GPIO def myCommandCallback(cmd):

print("Commandreceived: %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
aneventof type "greeting" 10 times deviceCli.connect() while True:
#Get Sensor Data fromDHT11
   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 IoTF")
time.sleep(10) deviceCli.commandCallback =
myCommandCallback
#Disconnect the device and application from the cloud deviceCli.disconnect()
```