PROJECT REPORT

IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

TEAM ID: PNT2022TMID51806

Team Leader

Ajisha K (961619205004)

Team Members

Subitha S (961619205016)

Akash margin (961619205301

Bachelor of Technology

In

Information technology

Marthandam College of Engineering And Technology Anna University, Chennai.

TABLE OF CONTENTS

1. INTRODUCTION:

1.1 Project Overview

The Smart protection system defines that this project help to farmer for the protection of a farm. We have designed this project for the only secure from animals but we this project have the provision to secure from the human begins also. This can achieve by the help of IOT device that we are discuss in this paper. The SCPS work on the battery so that this project can be easily portable and also we are add solar panels and converter modules this can help the battery to charge from solar energy. The IOT device is used to indicate the farmer by a message while someone enter into the farm and we are used SD card module that helps to store a specified sound to fear the animals. This project is smart crop protection system for protect the farm from animals as well as unknown person. This projects contents ardiuno UNO, Nodemcu, LCD display, PIR sensor, flame sensor, sd card module, solar panel, solar charges converter. This whole project is work on 12v dc supply from battery. We used solar panel to charge the battery. This project is based on Internet Of Things (IoT), that can measure soil moisture, Humidity and temperature conditions for agriculture and crop protection 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 moisture, Humidity and Temperature data obtained from sensors we make use of IBM IoT Simulator which can transmit these parameters as required.

2.2 Purpose

An intelligent crop protection system helps the farmers in protecting the crop from the animals and birds which destroy the crop. This system also helps farmers to monitor the soil moisture levels in the field and also the temperature and humidity values near the field. The motors and sprinklers in the field can be controlled using the mobile application.

2. LITERATURE SURVEY:

2.1 Existing Problem

● Agriculture is a field which forms the basis of our economy. Yet it faces a lot of problems in terms of availability of resources, Irrigation, increasing rate of Pesticides, Climatic disasters, Insects which ruin the crops and makes a huge loss this sector.

● 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

● https://smartinternz.com/assets/docs/Smart%20Home%
20Automation%20using%20IBM%20cloud%20Service s%20(1).pdf

● https://smartinternz.com/assets/docs/Smart%20Home%
20Automation%20using%20IBM%20cloud%20Service s%20(1).pdf

● https://openweathermap.org/

● https://smartinternz.com/assets/docs/Sending%2 0Http%
20request%20to%20Open%20weather%20map%20 web
site%20to%20get%20the%20weather%20forecast.pdf

● https://www.youtube.com/watch?v=cicTw4SEdxk

● https://smartinternz.com/assets/docs/Smart%20Home%
20Automation%20using%20IBM%20cloud%20Service s%20(1).pdf

● https://github.com/rachuriharish23/ibmsubscribe

2.3 Problem Statement Definition

● Smart Crop Protection System based on IoT can monitor soil moisture and climatic conditions to grow and yield a good crop.

● The farmer can also get the real time weather forecasting data by using external platforms like Open Weather API.

● Farmer is provided a mobile app using which he can monitor the temperature, humidity and soil moisture parameters along with weather forecasting details.

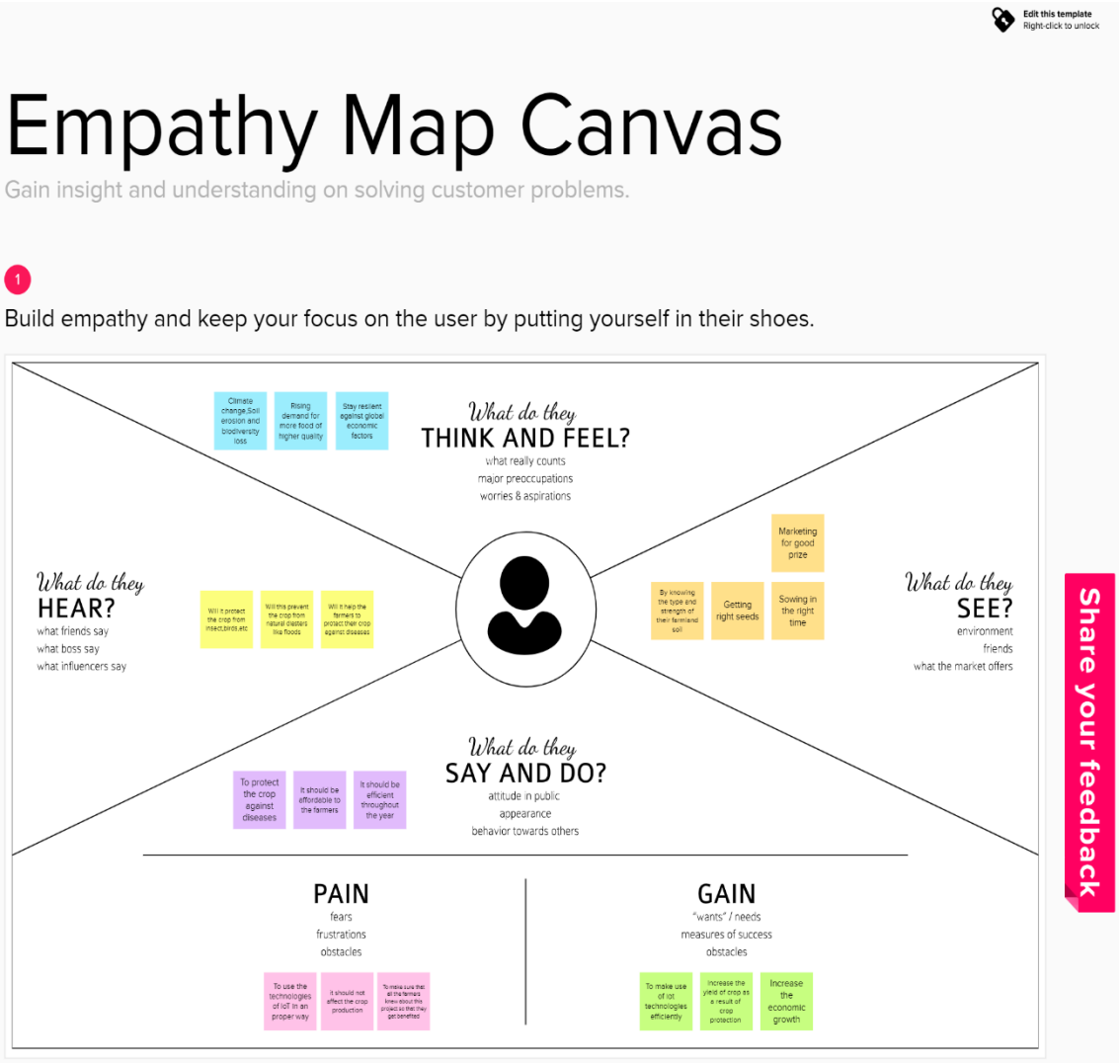
● Based on all the parameters he can water his crop by controlling the motors using the mobile application.

● Even if the farmer is not present near his crop he can water his crop by controlling the motors using the mobile application from anywhere.

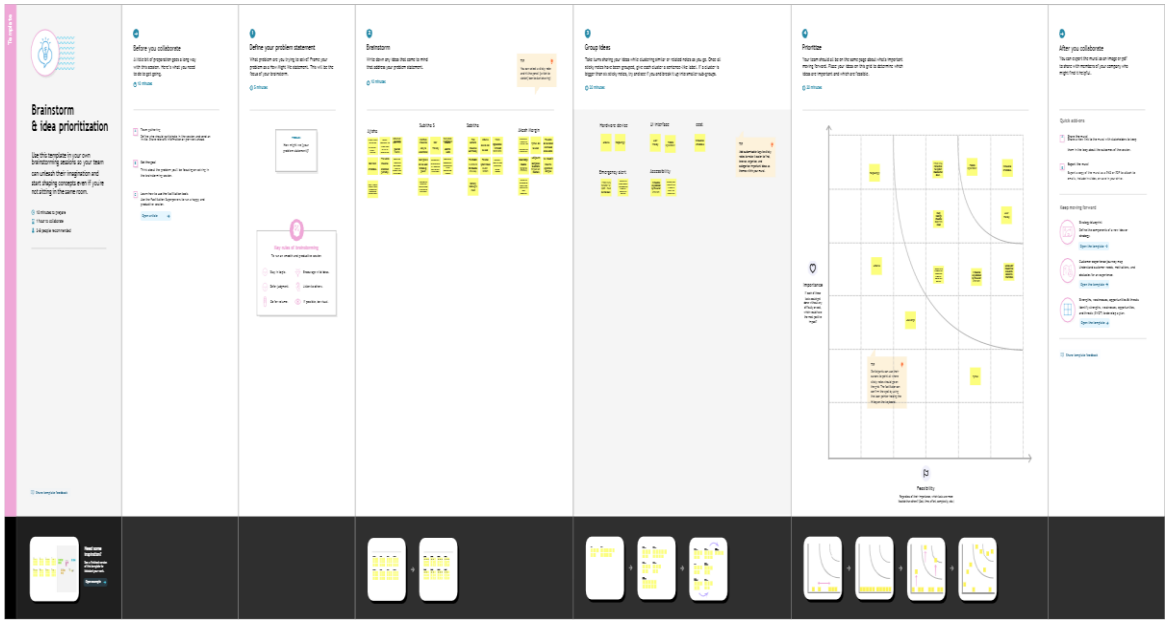
● Here we are using the Online IoT simulator for getting the Temperature, Humidity and Soil Moisture values.

3. IDEATION & PROPOSED SOLUTION:

3.1 Empathy Map Canvas



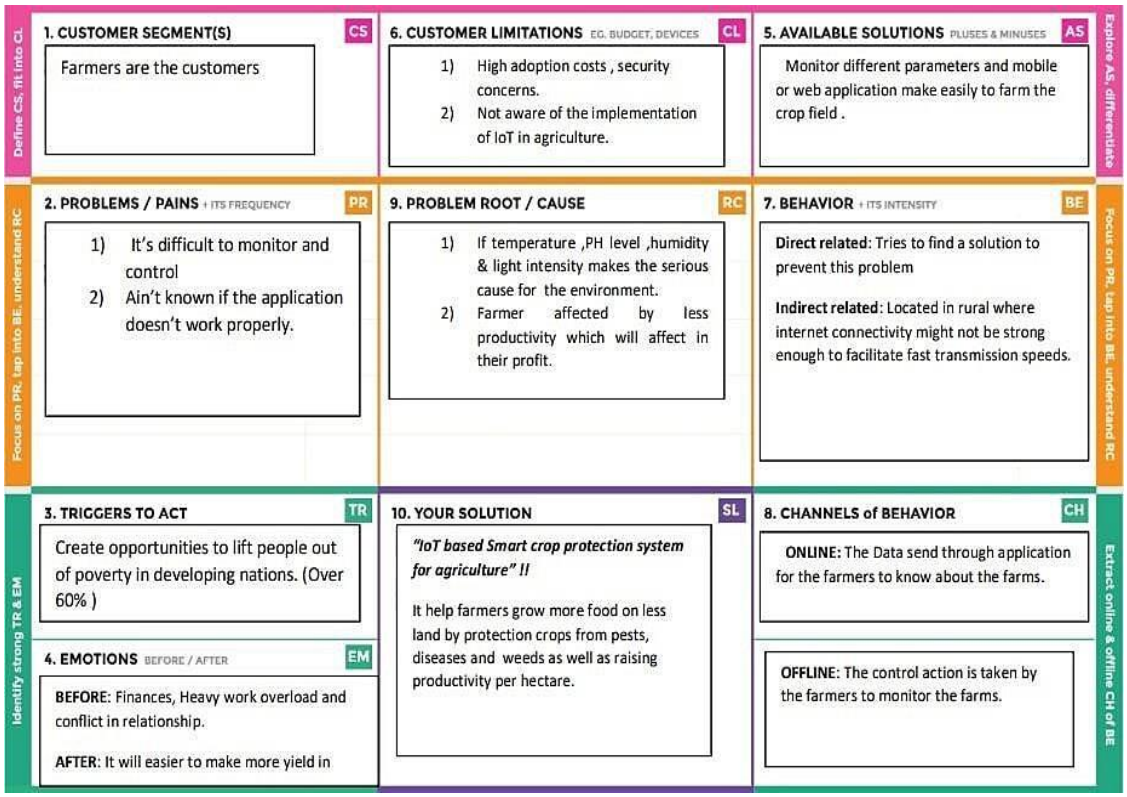
3.2 Ideation & Brain Storming



3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	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.
2.	Idea / Solution description	Here we propose an automatic crop protection system from animals and fire. This is an arduino 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.
3.	Novelty / Uniqueness	Fastest alert to the farmers through SMS.
4.	Social Impact / Customer Satisfaction	Real time data and production insight. Remote monitoring.
5.	Business Model (Revenue Model)	Help farmers in protecting their orchards and fields and save them from significant financial losses and will save them from the unproductive efforts that they endure for the protection their fields. This will also help them in achieving better crop yields thus leading to their economic wellbeing.
6.	Scalability of the Solution	Alerts the farmers immediately through an SMS.

3.4 Problem Solution Fit



4.REQUIREMENT ANALYSIS:

4.1.Functional Requirements

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Install the app. Signing up with Gmail or phone numbers. Creating a new profile. Understand the guidelines which we given
FR-2	User Confirmation	Email or phone number verification required via OTP.
FR-3	Accessing datasets	The data like values of temperature, data sensor, humidity, soil moisture are received by alert SMS.
FR-4	Interface sensor	Connect the sensor and the application When animals enter the field, the alarm is generated.
FR-5	User action	The user needs to take action like detecting through crop rotation, fertilizer, strip cropping.

4.2.Non Functional Requirements

Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This project's contributors to the farm protection through the smart protection system and use new technologies and also increase the quality of its crop.

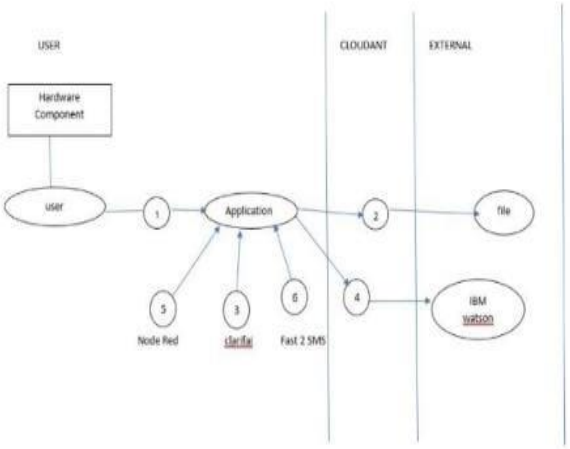
NFR-2	**Security**	It was created to protect the crops from animals.
NFR-3	**Reliability**	Farmers are able to safeguard their lands by help of this technology. They get some good benefits from higher crop yields.
NFR-4	**Performance**	When animals attempt to enter the crop field, IOT devices and sensors alert the farmer via message and maintain good yields.
NFR-5	**Availability**	Agriculture fences are quite an effective wild animal protection system.
NFR-6	**Scalability**	The develop system will not harmful and injurious to animals as well as human beings through the system.

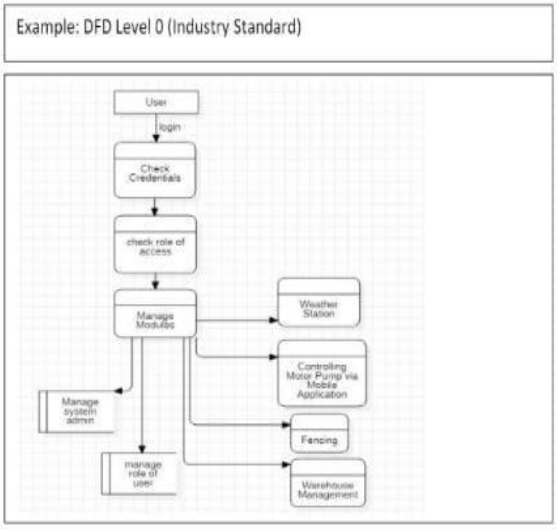
5.PROJECT DESIGN:

5.1Data Flow Diagram

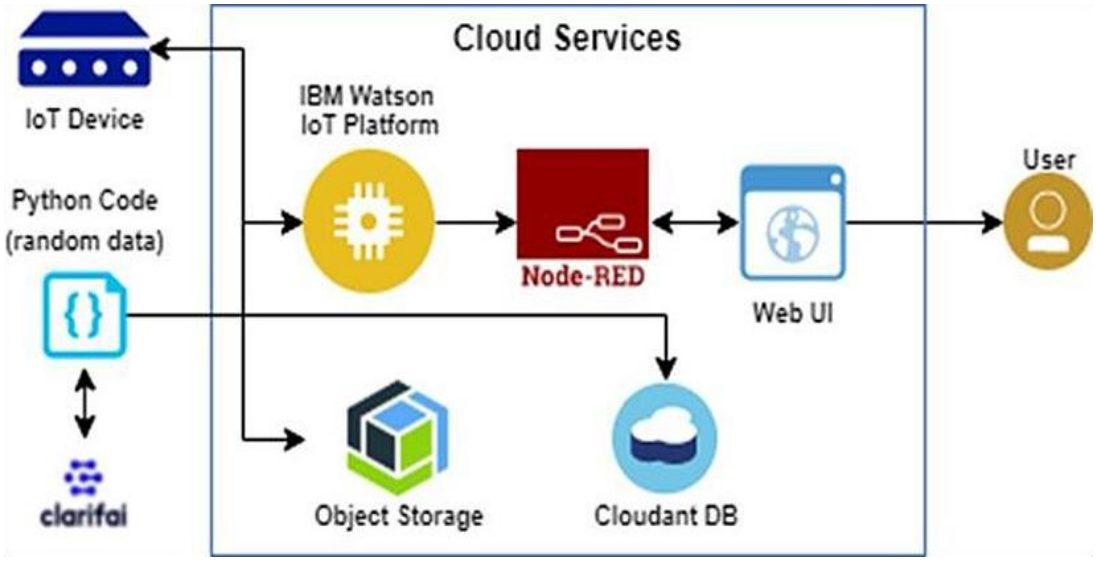
Data Flow Diagrams:

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





5.2.Solution & Technical Architecture



5.3.User Stories

User Type	Functional requirement(Epic)	User Story number	User Story/Task	Acceptancecriteria	Priority	Release
Customer (Mobil user)	Registration	USN-1	User can enter into the web application	I can accessmy account /dashboard	High	Sprint 1
		USN-2	User can register their credentials likeemail id and password	I can receive confirmation email & click confirm	High	Sprint 1
	Login	USN-3	User can log into theapplication by entering email & password	I can login tomy account	High	Sprint 1
	Dashboard	USN-4	User can view the temperature	I can view the data given by the device	High	Sprint 2
		USN-5	User can view thelevel of sensor monitoring value	I can view the data given by the device	High	Sprint 2
Customer(Web user)	Usage	USN-1	User can view the web page and get theinformation	I can view the data given by the device	High	Sprint 3
Customer	Working	USN-1	User act according to the alert given by thedevice	I can get thedata work according to it	High	Sprint 3
		USN-2	User turns ON Buzzer/Sound Alarm when the disturbancewill occur on field.	I can get thedata work according toit		Sprint 4
Administration	Administration	USN-1	User store every information	I can store the gained information	High	Sprint 4

6.PROJECT PLANNING& SCHEDULING

6.1 Sprint Planning & Estimation

TITLE	DESCRIPTION	DATE
Literature Survey on The Selected Project and Information Gathering	A Literature Survey is a compilation summary of research done previously in the given topic. Literature survey can be taken from books, research paper online or from any source.	20 September 2022
Prepare Empathy Map	Empathy Map is a visualization tool which can be used to get a better insight of the customer	22 September 2022
Ideation-Brainstorming	Brainstorming is a group problem solving session where ideas are shared, discussed and organized among the team members.	28 September 2022
Define Problem Statement	A Problem Statement is a concise description of the problem or issues a project seeks to address. The problem statement identifies the current state, the desired future state and any gaps between the two.	20 September 2022
Problem Solution Fit	This helps us to understand the thoughts of the customer their likes, behaviour, emotions etc.	01 October 2022
Proposed Solution	Proposed solution shows the current solution and it helps is going towards the desired result until it is achieved.	18 October 2022
Solution Architecture	Solution Architecture is a very complex process I.e it has a lot of sub-processes and branches. It helps in understanding the components and features to complete our project.	18 October 2022
Customer Journey	It helps us to analyse from the perspective of a customer, who uses our project.	01 November 2022
Functional Requirement	Here functional and nonfunctional requirements are briefed. It has specific features like usability, security, reliability, performance, availability and scalability.	01 November 2022
Data Flow Diagrams	Data Flow Diagram is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement.	03 November 2022
Technology Architecture	Technology Architecture is a more well defined version of solution architecture. It helps us analyze and understand various technologies that needs to be implemented in the project.	03 November 2022
Prepare Milestone & Activity List	It helps us to understand and evaluate our own progress and accuracy so far.	06 November 2022
Spring Delivery Plan	Sprint planning is an event in scrum that kicks off the sprint. The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved.	06 November 2022

6.2 Sprint Delivery Schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
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	Akash Selvin S Derish Kenimer Joffin V Rana Prathap
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	Akash Selvin S Derish Kenimer Joffin V Rana Prathap
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	Akash Selvin S Derish Kenimer Joffin V Rana Prathap
Sprint-3		US-2	Create a Node-RED service.	10	High	Akash Selvin S Derish Kenimer Joffin V Rana Prathap

Sprint-3		US-1	Develop a system which will sensor the animals entry into the fields and intimate the farmers.	7	High	Akash Selvin S Derish Kenimer Joffin V Rana Prathap
Sprint-3		US-2	After developing python code, commands are received just print the statements which represent the control of the devices.	5	Medium	Akash Selvin S Derish Kenimer Joffin V Rana Prathap
Sprint-4		US-3	Publish Data to The IBM Cloud	8	High	Akash Selvin S Derish Kenimer Joffin V Rana Prathap
Sprint-4		US-1	Create Web UI in Node- Red	10	High	Akash Selvin S Derish Kenimer Joffin V Rana Prathap
Sprint-4		US-2	Configure the Node-RED flow to receive	10	High	Akash Selvin S Derish Kenimer Joffin V

6.3 Reports from JIRA

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	8	6 Days	24 Oct 2022	29 Oct 2022	22	29 Oct 2022
Sprint-2	1	6 Days	31 Oct 2022	05 Nov 2022	10	05 Nov 2022
Sprint-3	2	6 Days	07 Nov 2022	12 Nov 2022	12	12 Nov 2022
Sprint-4	1	6 Days	14 Nov 2022	19 Nov 2022	9	19 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

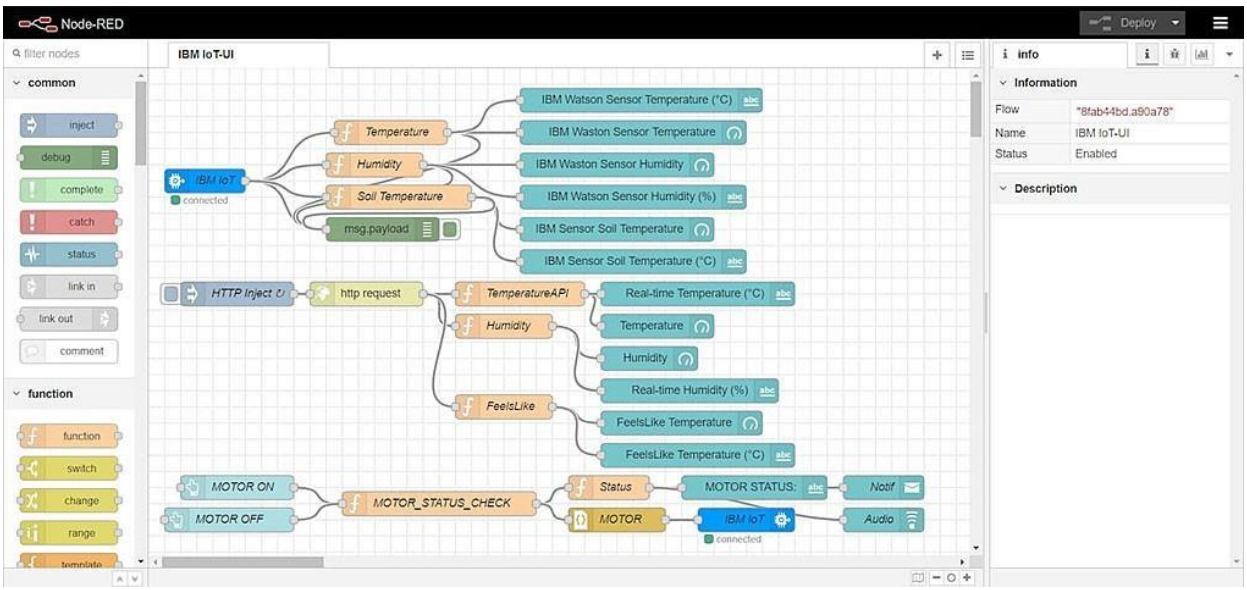
$$AV = \frac{sprint\ duration}{velocity}$$

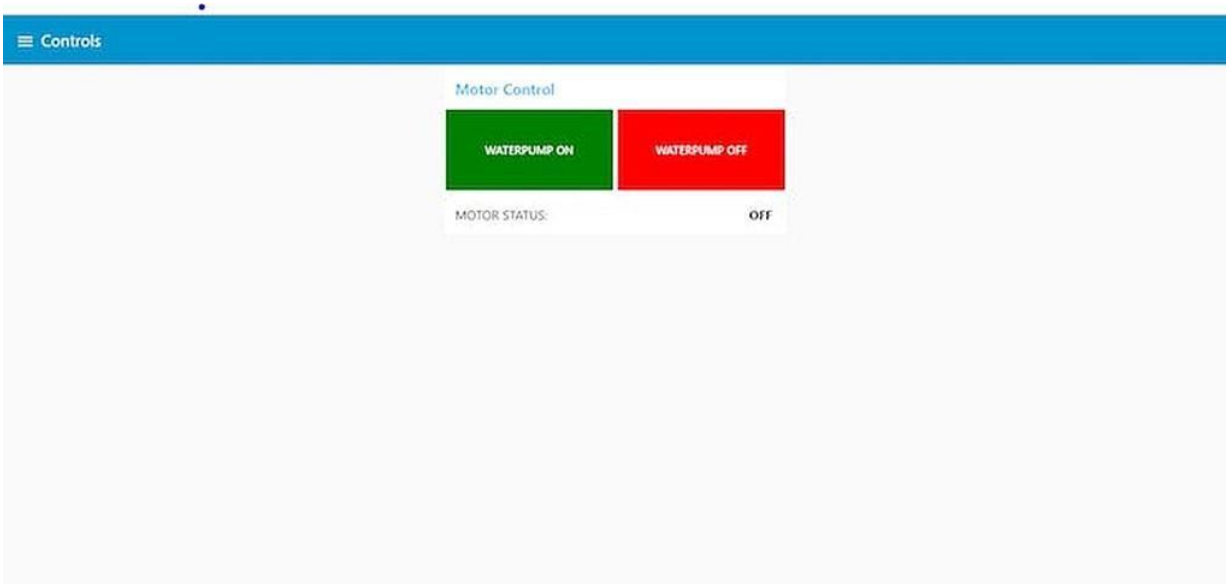
=6/13.25

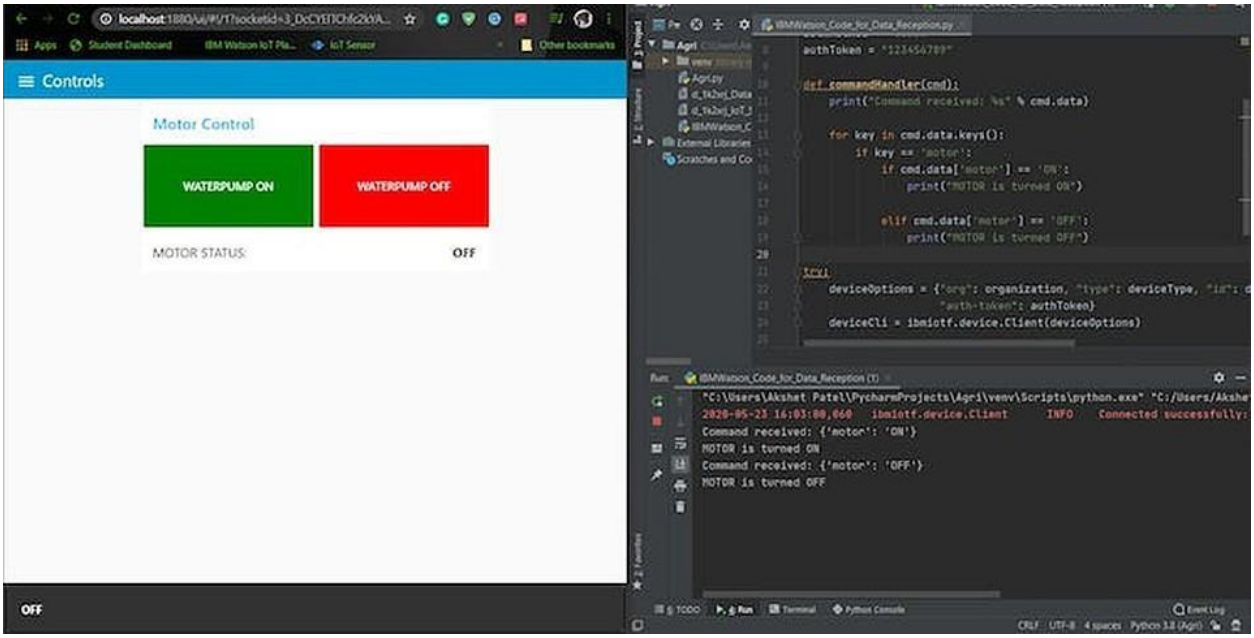
=0.45

7.CODING & SOLUTIONING:

7.1.Feature 1



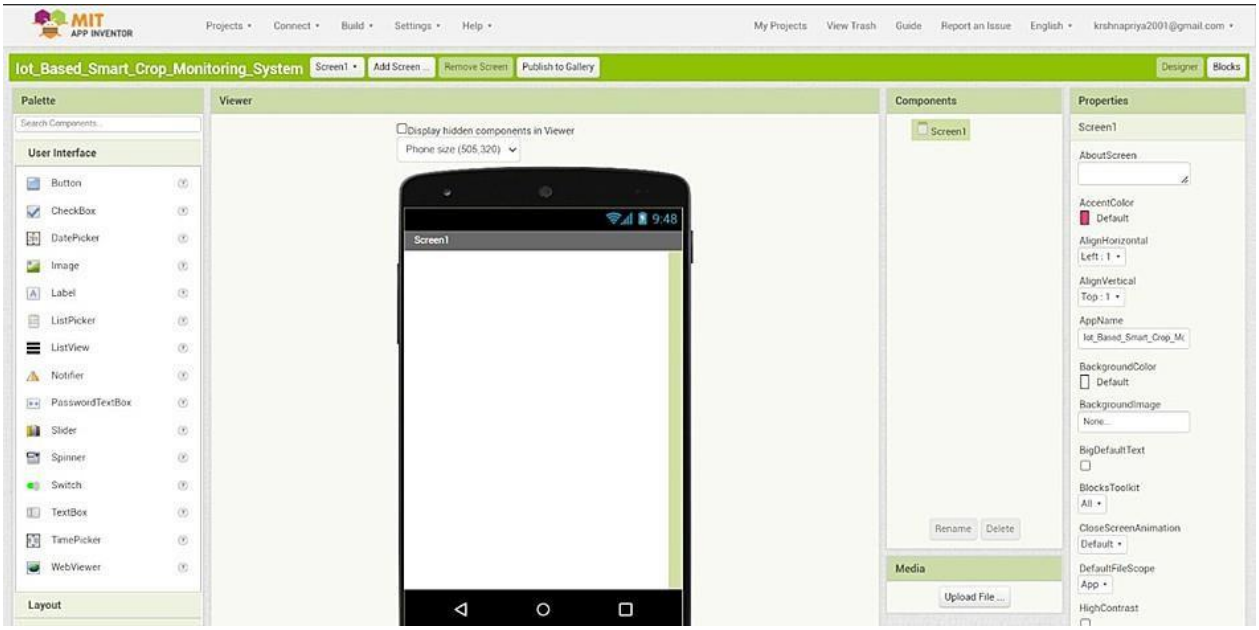



Controls
Motor Control
WATERPUMP ON
WATERPUMP OFF
MOTOR STATUS:
OFF
authToken = "123456789"
def commandHandler(cmd):
print("Command received: %s" % cmd.data)
for key in cmd.data.keys():
if key == 'motor':
if cmd.data['motor'] == 'ON':
print("MOTOR is turned ON")
elif cmd.data['motor'] == 'OFF':
print("MOTOR is turned OFF")
deviceOptions = {"org": organization, "type": deviceType,
"auth-token": authToken}
deviceCli = ibmiotf.device.Client(deviceOptions)
Command received: {'motor': 'ON'}
MOTOR is turned ON
Command received: {'motor': 'OFF'}
MOTOR is turned OFF

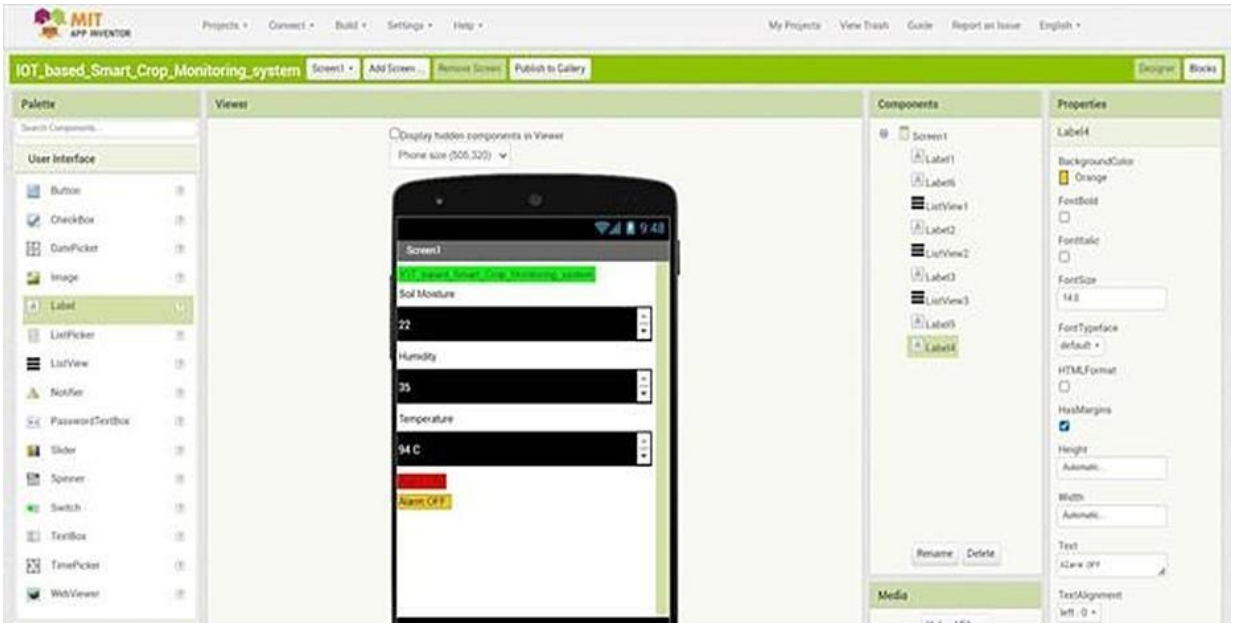

IBM Watson
IBM Watson Sensor Temperature (°C) 15
IBM Watson Sensor Humidity (%) 75
IBM Sensor Soil Temperature (°C) 24
Realt-time Weather
Real-time Temperature (°C) 31
Real-time Humidity (%) 70
FeelsLike Temperature (°C) 34.82
IBM Sensor Soil Temperature
24
Temperature
31
IBM Waston Sensor Temperature
15
FeelsLike Temperature
34.82
IBM Waston Sensor Humidity
75
Humidity
70

7.2.Feature 2

MIT APP inventor to design the APP



Customize the App interface to Display the Values



8.TESTING:

Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	11	4	2	2	19
Duplicate	1	1	2	0	4
External	2	3	0	1	6
Fixed	10	2	3	20	35
Not Reproduced	0	0	2	0	2
Skipped	0	0	2	1	3
Won't Fix	0	5	2	1	8
Totals	24	15	13	25	77

Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	1	4
Client Application	47	0	2	45
Security	3	0	0	3
Outsource Shipping	2	0	0	2
Exception Reporting	11	0	2	9
Final Report Output	5	0	0	5
Version Control	3	0	1	2

9.RESULT:

We have successfully built an IOT Based Smart Crop Protection System for Agriculture andintegrated all the services using Node-RED.

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 automatedand 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 Crop Protection requires internet connectivity continuously, but rural parts cannot fulfill this requirement.

1. Any faults in the sensors can cause great loss in the agriculture, due to wrong record and the actions of automated processes.
2. IoT devices need much money to implement.

11 CONCLUSION:

IoT based smart Crop Monitoring System for Agriculture for Live Monitoring of Temperature and Soil Moisture and to control motor and light remotely has been proposed using Node Red and IBM CloudPlatform. 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 project 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. Therefore, the project proposes a thought of consolidating the most recent innovation into the agrarian field to turn the customary techniques for water system to current strategies in this way making simple profitable and temperate trimming.

12.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 resource like electricity and water IoT can be implemented in most of the places.

13.APPENDIX:

```
import random
        import ibmiotf.device          from
  time import sleep
                    import sys
                    #IBM Watson Device Credentials.
        organization = "op701j"          deviceType
  = "Lokesh"          deviceId = "Lokesh89"
  authMethod = "token"          authToken =
```

```
"1223334444"            def
myCommandCallback(cmd):
                        print("Command received: %s" % cmd.data['command'])
  status=cmd.data['command']                    if
status=="sprinkler_on":      print ("sprinkler is
ON")    else :
                           print ("sprinkler is OFF")
                        #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()
                        #Connecting to IBM watson.
        deviceCli.connect()            while
True:
                        #Getting values from sensors.
                         temp_sensor = round( random.uniform(0,80),2)
                         PH_sensor = round(random.uniform(1,14),3)

                         camera = ["Detected","Not Detected","Not Detected","Not Detected","Not
        Detected","Not Detected",]     camera_reading =
random.choice(camera)
                         flame = ["Detected","Not Detected","Not Detected","Not Detected","Not
        Detected","Not Detected",]     flame_reading =
random.choice(flame)          moist_level =
round(random.uniform(0,100),2)
                         water_level = round(random.uniform(0,30),2)



                        #storing the sensor data to send in json format to cloud.
```

```
        temp_data = { 'Temperature' : temp_sensor }          PH_data
= { 'PH Level' : PH_sensor }        camera_data = { 'Animal attack' :
camera_reading}         flame_data = { 'Flame' : flame_reading }
moist_data = { 'Moisture Level' : moist_level}          water_data =
{ 'Water Level' : water_level}


                # publishing Sensor data to IBM Watson for every 5-10 seconds.  success =
                deviceCli.publishEvent("Temperature sensor", "json", temp_data,
                qos=0)
        sleep(1)          if
success:
                    print (" .............................publish ok..... ")
            print ("Published Temperature = %s C" % temp_sensor, "to IBM Watson")


            success = deviceCli.publishEvent("PH sensor", "json", PH_data, qos=0)
        sleep(1)          if
success:
                 print ("Published PH Level = %s" % PH_sensor, "to IBM Watson")


           success = deviceCli.publishEvent("camera", "json", camera_data, qos=0)
        sleep(1)          if
success:
                   print ("Published Animal attack %s " % camera_reading, "to IBM
                Watson")
                 success = deviceCli.publishEvent("Flame sensor", "json", flame_data,
                qos=0)  sleep(1)  if success:
                   print ("Published Flame %s " % flame_reading, "to IBM Watson")


                 success = deviceCli.publishEvent("Moisture sensor", "json", moist_data,
                qos=0)
        sleep(1)          if
success:
                    print ("Published Moisture Level = %s " % moist_level, "to IBM
                Watson")
```

```
                success = deviceCli.publishEvent("Water sensor", "json", water_data,
                qos=0)
        sleep(1)          if
success:
            print ("Published Water Level = %s cm" % water_level, "to IBM Watson")
                print ("")
                #Automation to control sprinklers by present temperature an to send alert
                message to IBM Watson.

                                                                    if (temp_sensor > 35):
                  print("sprinkler-1 is ON")
                success = deviceCli.publishEvent("Alert1", "json",{ 'alert1' :
              "Temperature(%s) is high, sprinkerlers are turned ON" %temp_sensor }
       , qos=0)        sleep(1)
if success:
                  print( 'Published alert1 : ', "Temperature(%s) is high, sprinkerlers are
                turned ON" %temp_sensor,"to IBM Watson")
        print("")          else:
        print("sprinkler-1 is OFF")     print("")


                #To send alert message if farmer uses the unsafe fertilizer to crops.

                                                    if (PH_sensor > 7.5 or PH_sensor < 5.5):
                  success = deviceCli.publishEvent("Alert2", "json",{ 'alert2' : "Fertilizer PH
                level(%s) is not safe,use other fertilizer" %PH_sensor } ,
                qos=0)
                sleep(1)  if
                success:
                print('Publi
                shed alert2
                : ' ,
                "Fertilizer
                PH
                level(%s)
                is not
```

```
safe,use
other
fertilizer" %PH_sensor,"to IBM Watson")
 print("")


 #To send alert message to farmer that animal attack on crops.

                                                    if (camera_reading == "Detected"):
   success = deviceCli.publishEvent("Alert3", "json", { 'alert3' : "Animal
attack on crops detected" }, qos=0)
 sleep(1)          if
success:
   print('Published alert3 : ' , "Animal attack on crops detected","to IBM
 Watson","to IBM Watson")     print("")
 #To send alert message if flame detected on crop land and turn ON the
 splinkers to take immediate action.

                                                     if (flame_reading == "Detected"):
   print("sprinkler-2 is ON")
 success = deviceCli.publishEvent("Alert4", "json", { 'alert4' : "Flame is
 detected crops are in danger,sprinklers turned ON" }, qos=0)
 sleep(1)          if
success:
   print( 'Published alert4 : ' , "Flame is detected crops are in
 danger,sprinklers turned ON","to IBM Watson")


 #To send alert message if Moisture level is LOW and to Turn ON Motor-1
for irrigation.   if (moist_level < 20):      print("Motor-1 is ON")
 success = deviceCli.publishEvent("Alert5", "json", { 'alert5' : "Moisture
 level(%s) is low, Irrigation started" %moist_level }, qos=0)          sleep(1)
if success:
   print('Published alert5 : ' , "Moisture level(%s) is low, Irrigation started"
 %moist_level,"to IBM Watson" )
 print("")
 #To send alert message if Water level is HIGH and to Turn ON Motor-2 to
 take water out.  if (water_level > 20):     print("Motor-2 is ON")
```

```
                success = deviceCli.publishEvent("Alert6", "json", { 'alert6' : "Water
                level(%s) is high, so motor is ON to take water out "
                %water_level }, qos=0)
        sleep(1)            if
success:
                    print('Published alert6 : ' , "water level(%s) is high, so motor is ON to take
      water out " %water_level,"to IBM Watson" )              print("")
 #command recived by farmer  deviceCli.commandCallback =
myCommandCallback
                # Disconnect the device and application from the cloud deviceCli.disconnect()
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-5381-1658761312