IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

TEAM ID:PNT2022TMID28957

TEAM MEMBERS

NIVETHA.S

SARANYA.R

SAKTHI PRIYA .K

RITHIKA .S

1.INTRODUCTION

1.1 Project Overview

The device will detect the animals and birds using the Clarifai service.If any animal or bird is detected the image will be captured and stored in the IBM Cloud object storage.It also generates an alarm and avoid animals from destroying the crop.The image URL will be stored in the IBM Cloudant DB service.The device will also monitor the soil moisture levels, temperature, and humidity values and send them to the IBM IoT Platform.The image will be retrieved from Object storage and displayed in the web application.A web application is developed to visualize the soil moisture, temperature, and humidity values.Users can also control the motors through web applications.To accomplish this, we have to complete all the activities and tasks

1.2 PURPOSE

The main purpose of 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

1. Smart Crop protection system from living objects and fire using Arduino [1]

This paper motive to designing and executing the superior improvement in embedded Device for Crops in farms are over and over ravaged with the aid of nearby animals like Buffaloes, cows, goats, birds, and fireplace etc. This results in huge losses for the farmers. It is now not feasible for farmers to barricade complete fields or precede field 24 hours and Protect it. Therefore here we present computerized crop safety system from animals and Fire. This is a Arduino Uno primarily based device the use of microcontroller. This Technique makes use of a motion sensor to discover wild animals drawing near the sphere And smoke sensor to discover the hearth. In such a case the sensor alerts the microcontroller to require action. The microcontroller now sounds an alarm to woo the animals away from The sector further as sends SMS to the farmer and makes call, in order that farmer may Fathom the difficulty and come to the spot just in case the animals don't recede by the alarm. If there's a smoke, it immediately turns ON the motor. This provide us entire safety of plants From animals and from fireplace for this reason protecting the farmer's loss[1]

2. Review on IoT in Agricultural Crop Protection and Power Generation[2]

Agriculture is that the science and artwork of cultivating plants. Agriculture performs most Important position inside the economic development of our us of a and this can be the first Occupation from a few years. So as to extend the productivity of the crops and to attenuate The expenses of agricultural practices we adopt smart agriculture techniques using IOT. The Sensors are placed at different locations within the farm, by which the parameters is Controlled using remote or through internet services and by interfacing the sensors Operations are performed with microcontrollers. India is that the second most populated Country. Power generation and supply is typically an unlimited problem. This paper mainly Addresses power generation and rainwater harvesting as an influence generation method using energy together with crop protection.

3. IOT based smart crop monitoring in farm land

As new technologies has been introduced and utilized in modern world, there is a need to Bring advancement in the sector of agriculture also. Various Researches have been Undergone to enhance crop cultivation and are widely used. So as to enhance the crop Productivity efficiently, it is necessary to monitor the environmental conditions in and Around the field. The parameters that has to be exact monitored to enhance the yield are Soil characteristics, weather conditions, moisture, temperature, etc., Internet of Things (IOT) is being utilized in a number of real time applications. The introduction of Internet Of thing (IOT) along with the sensor network in framrefurbishes the traditional way of Farming. Online crop monitoring the use of IOT helps the farmers to stay related to his Subject from somewhere and anytime. Various sensors are used to screen and collect Records about the area conditions. Collectively the about the farm circumstance is disbursed To the farmer thru GSM technology.[3]

4. Development of IOT based Smart Security and Monitoring Devices for Agriculture

Agriculture area being the backbone of the Indian economy deserves security. Security no Longer in phrases of sources solely however additionally agricultural products wishes Protection and safety at very preliminary stage, like protection from attacks of rodents or Insects, in fields or grain stores. Such challenges should even be taken into consideration. Security systems which are getting used now a days don't seem to be smart enough to Produce real time notification after sensing the matter.the mixture of typical methodology With present day technologies as Internet of Things and Wireless Sensor Networks can Cause agricultural modernization. Keeping this scenario in our mind we've got designed Tested and analyzed an 'Internet of Things' based device which is capable of analyzing the Sensed information then transmitting it to the user. This gadget will be controlled and Monitored from far off region and it is carried out in agricultural fields, grain shops and Bloodless stores for protection purpose. This paper is oriented to intensify the methods to Unravel such problems like identification of rodents, threats to crops and turning in actualtime notification supported records evaluation and processing besides human intervention.During this device, referred to sensors and digital units are built-in using Python scripts. Supported attempted take a look at cases, we had been capable to obtain success in 84.8% check cases. [4]

2.1 REFERENCES

[1] Dr.M. Chandra ,Mohan Reddy, KeerthiRajuKamakshiKodi, BabithaAnapalliMounikaPulla, “SMART CROP PROTECTION SYSTEM FROM LIVING OBJECTS AND FIRE USING ARDUINO”, Science, Technology and Development, Volume IX Issue IX

[2] Anjana ,Sowmya , Charan Kumar , Monisha , Sahana, “ Review on IoT in Agricultural Crop Protection and Power Generation”, International Research Journal of Engineering and Technology (IRJET) , Volume 06, Issue 11 ,Nov 2019.

[3] G. NaveenBalaji, V. Nandhini, S. Mithra, N. Priya , R. Naveena, “IOT based smart crop monitoring in farm land ”,Imperial Journal of Interdisciplinary Research (IJIR), Volume 04, Issue 01 , Nov 2018.

[4] P.Rekha, T.Saranya, P.Preethi, L.Saraswathi, G.Shobana, “Smart AGRO Using ARDUINO and GSM”, International Journal of Emerging Technologies in Engineering Research (IJETER) Volume 5, Issue 3, March 2017.

2.2 PROBLEM STATEMENT

● The significant problem which raises the requirement of this project was that the traditional agriculture method consumes time, manual labor work and is also not cost efficient.

● The detected signal from the soil moisture sensor is processed by a conditional comparator circuit corresponding to different levels of actual soil moisture content. A logic circuit follows the conditional circuit with its output signals used to Activate a system of relays that control the power circuit of the motors used for Water pumping.

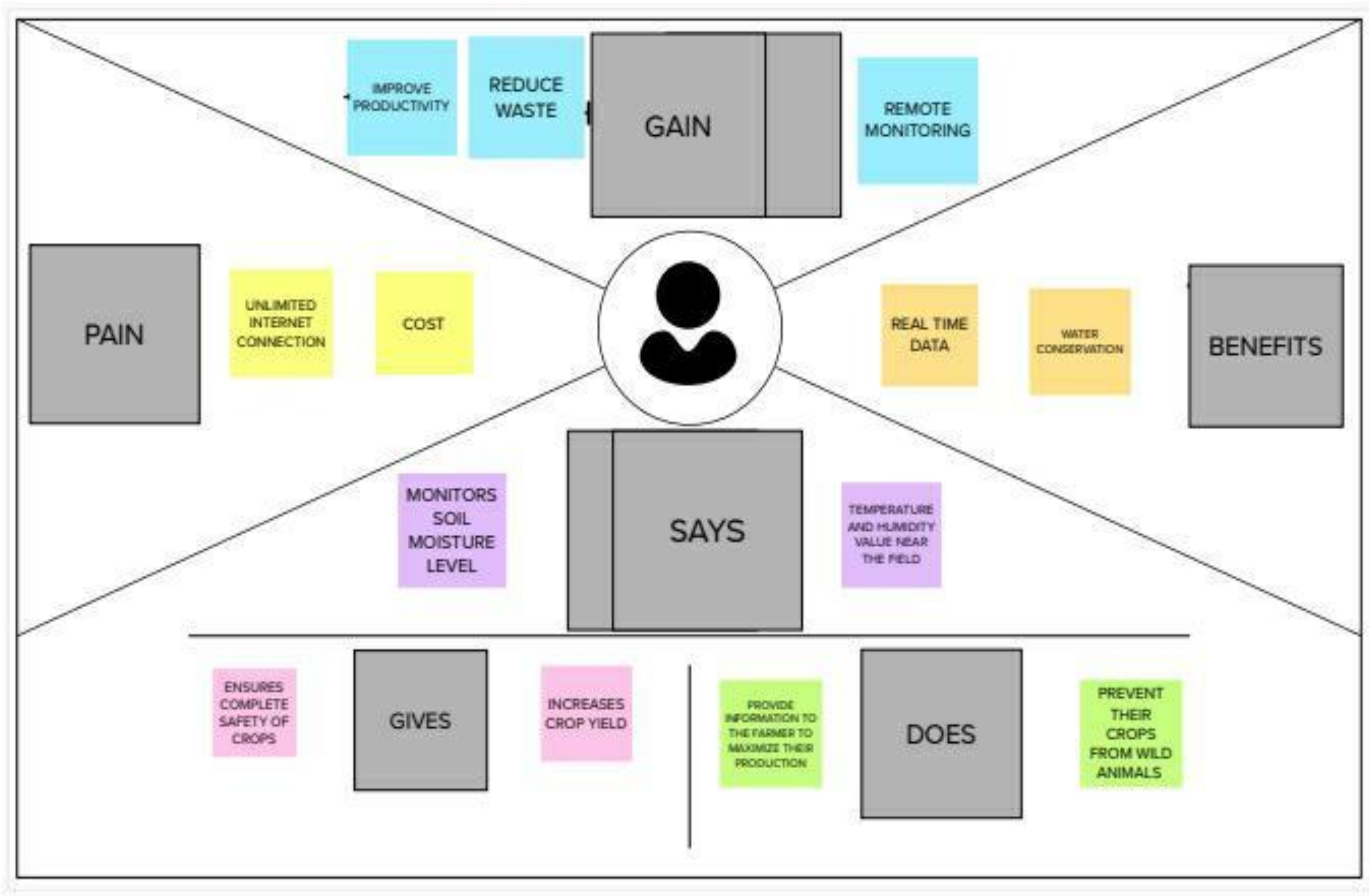
● IOT is developing rapidly and widely applied in all wireless environments. In This paper, sensor technology and wireless networks integration of IOT technology Has been studied and reviewed based on the actual situation of agricultural system. A combined approach with internet and wireless communications, Remote Monitoring System (RMS) is proposed.

● The system consists of esp8266 (nodeMCU), soil moisture sensor, dihydrogen Monoxide sensor, GPRS and GSM module, servo motor, dihydrogen monoxide Pump, etc. to obtain the required output.

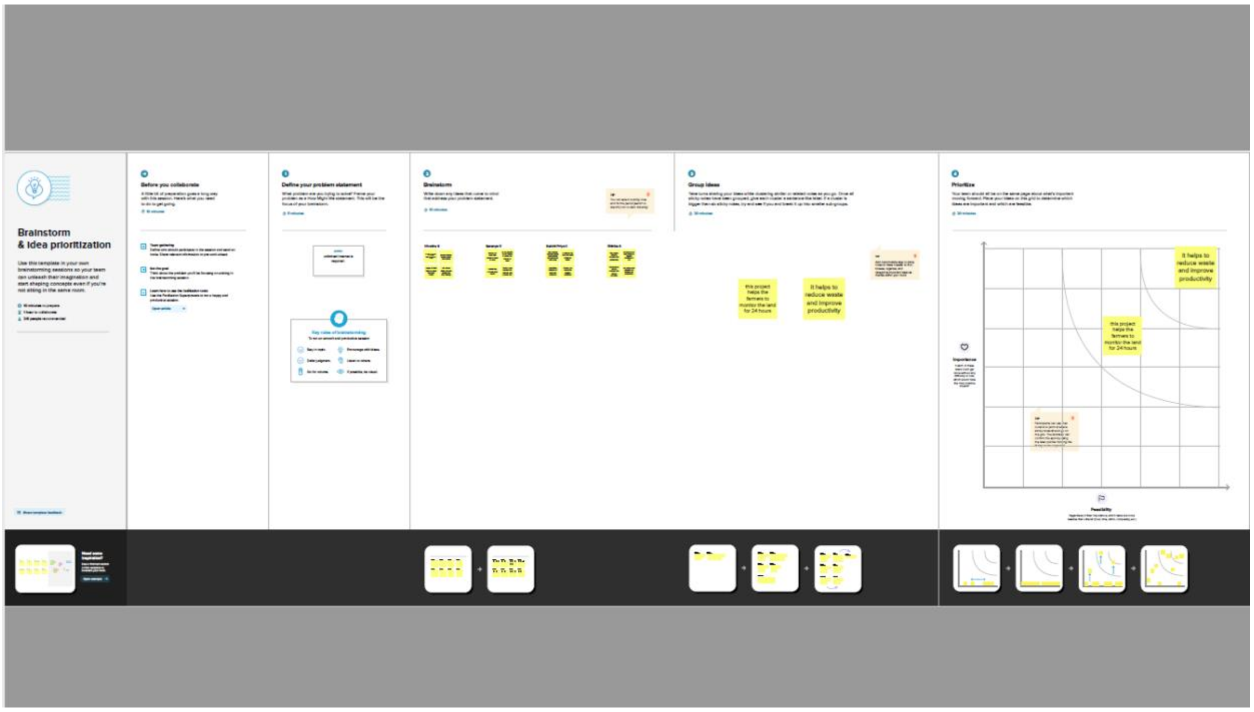
● The system was powered by photovoltaic panels and had a duplex Communication link based on a cellular-Internet interface that allowed for data Inspection and irrigation scheduling to be programmed through a web page.

3 IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP


IMPROVE PRODUCTIVITY
REDUCE WASTE
GAIN
REMOTE MONITORING
PAIN
UNLIMITED INTERNET CONNECTION
COST
REAL TIME DATA
WATER CONSERVATION
BENEFITS
MONITORS SOIL MOISTURE LEVEL
SAYS
TEMPERATURE AND HUMIDITY VALUE NEAR THE FIELD
ENSURES COMPLETE SAFETY OF CROPS
GIVES
INCREASES CROP YIELD
PROVIDE INFORMATION TO THE FARMER TO MAXIMIZE THEIR PRODUCTION
DOES
PREVENT THEIR CROPS FROM WILD ANIMALS

3.2 IDEATION AND BRAINSTORMING

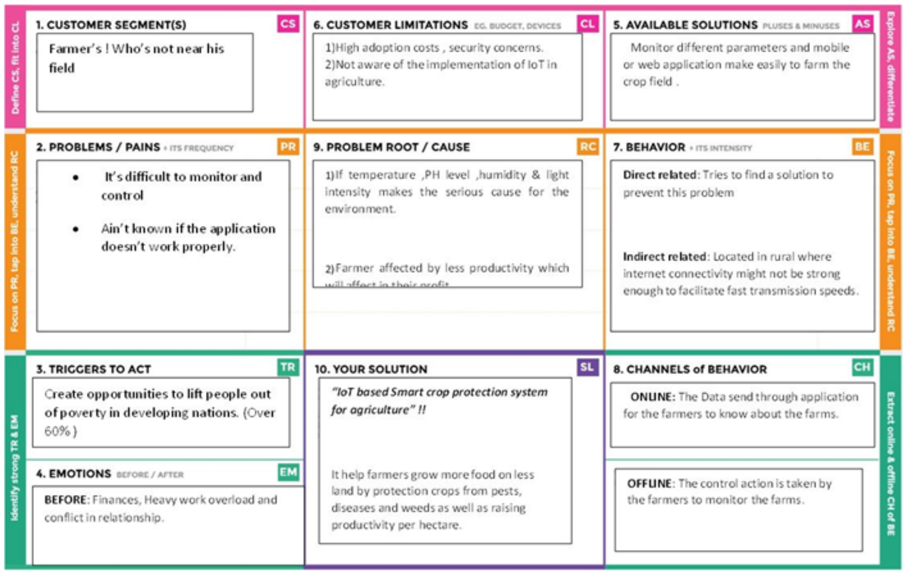

Brainstorm
& idea prioritization
Before you collaborate
Define your problem statement
Brainstorm
Group ideas
Prioritize
this project helps the farmers to monitor the land for 24 hours
It helps to reduce waste and improve productivity
Importance
Feasibility

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 to stay 24 hours in the farm to sentinel the crops.With the help of the IoT devices, you can know the real-time status of the crops by capturing the data from sensors
2	Idea / Solution description	Smart Farming has enabled farmers to reduce waste and enhance productivity with the help of sensors (light, humidity, temperature, soil moisture, etc.) .Further with the help of these

		sensors, farmers can monitor the field conditions from anywhere.
3	Novelty / Uniqueness	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.the announcement of the threshold rate will be sent to the cell number or to the website.The result will be generated on a catalog of the mobile of the person to take the necessary action.
4	Social Impact / Customer Satisfaction	Social Impact / Customer Satisfaction Improve the productivity,Save lives forFarmers/help to farmer for to protect his farm Increased production: the optimisation of all the processes related to agriculture and livestock-rearing increases production rates.
5	Business Model	Community based solution by FAO’s solution through contract farming.
6	Scalability of the solution	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 panal,solar charges converter. This whole project is work on 12v dc supply from battery. We used solar panel to charge the battery

3.4 PROBLEM SOLUTION FIT



Define CS, fit into CL

1. CUSTOMER SEGMENT(S) CS

Farmer's ! Who's not near his field

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL

1)High adoption costs , security concerns.
2)Not aware of the implementation of IoT in agriculture.

5. AVAILABLE SOLUTIONS PLUSES & MINUSES AS

Monitor different parameters and mobile or web application make easily to farm the crop field .

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2. PROBLEMS / PAINS + ITS FREQUENCY PR

- It's difficult to monitor and control
- Ain't known if the application doesn't work properly.

9. PROBLEM ROOT / CAUSE RC

1)If temperature ,PH level ,humidity & light intensity makes the serious cause for the environment.

2)Farmer affected by less productivity which will affect in their profit

7. BEHAVIOR + ITS INTENSITY BE

Direct related: Tries to find a solution to prevent this problem

Indirect related: Located in rural where internet connectivity might not be strong enough to facilitate fast transmission speeds.

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TO ACT TR

Create opportunities to lift people out of poverty in developing nations. (Over 60%)

4. EMOTIONS BEFORE / AFTER EM

BEFORE: Finances, Heavy work overload and conflict in relationship.

10. YOUR SOLUTION SL

"IoT based Smart crop protection system for agriculture" !!

It help farmers grow more food on less land by protection crops from pests, diseases and weeds as well as raising productivity per hectare.

8. CHANNELS of BEHAVIOR CH

ONLINE: The Data send through application for the farmers to know about the farms.

OFFLINE: The control action is taken by the farmers to monitor the farms.

Extract online & offline CH of BE

4 REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR NO.	Functional Requirement	Sub Requirement
FR-1	User registration	Install the app. Signing up with Gmail or phone number Creating a profile. Understand the guidelines
FR-2	User Confirmation	Email or phone number verification required via OTP
FR-3	Accessing datasets	Data's are obtained from iot device and send to cloud
FR-4	Mobile application	Farmer is provided a mobile app using which he can monitor the temperature, humidity and soil moisture parameters along with weather forecasting details.

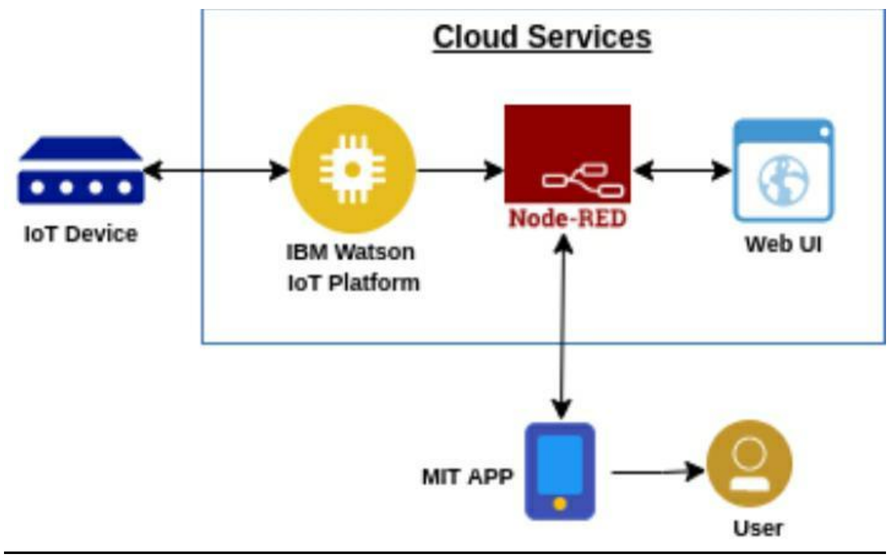
4.2 NON-FUNCTIONAL REQUIREMENTS

FR NO.	Non-functional Requirements	Description

NFR-1	Usability	An intelligent crop protection system helps the farmers in protecting the crop from the animals and birds which destroy the crop.
NFR-2	Security	It was created to protect the crops from animals
NFR-3	Reliability	increase efficiency, improve quality, and lower costs. And farmers can protect their land.
NFR-4	Performance	When animals or birds enter the land the sensor detects and alert the farmer via message .
NFR-5	Availability	We can defend the crops against wild animals by creating and implementing resilient hardware and software.
NFR-6	Scalability	This system’s integration of computer vision algorithms with IBM cloudant services makes it more efficient to retrieve photos at scale,enhancing scalability.

5.PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

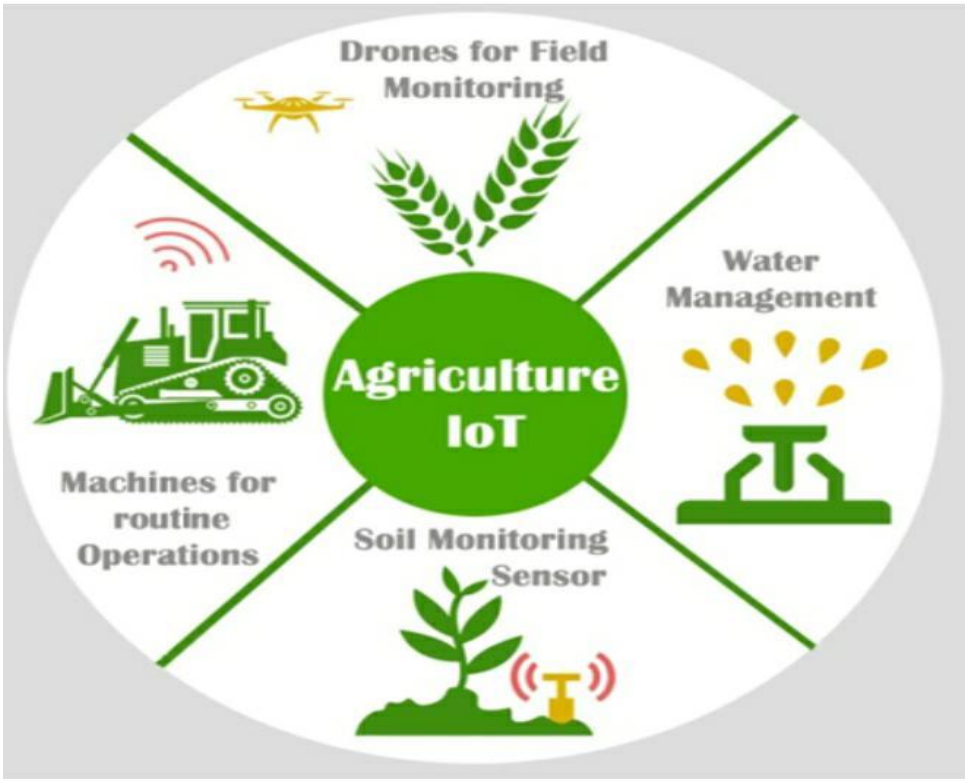


User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Developer	System development	USN-1	Collect dataset	I can collect dataset	high	Sprint 1
		USN-2	Collecting data from sensors	I can collect data from sensors	high	Sprint 1
		USN-3	Implementing arduino UNO from data collection		high	Sprint 2
		USN-4	Message alert to farmers	I can receive message	high	Sprint 3
		USN-5	Farmers identify the problem and resolve it by using mobile application	I can identify the problem and I try to resolve it	medium	Sprint 3
Customer (Web user)	Adoption	USN-1	Adopting new technology for boosting production	I can adopt new technology	low	Sprint 1
	Detection	USN-2	Detect the ratio of defected crops on land	I can detect the defected crops	high	Sprint 2

5.2SOLUTION ARCHITECTURE

- Different soil parameters (temperature, humidity, Light intensity, pH level) are sensed using different sensors and the obtained value is stored in IBM cloud.
- Arduino uno is used as a processing unit which processes the data obtained from sensors and weather data from weather API.
- Node red is used as a programming tool to wire the Hardware, software and APIs. The MQTT protocol is followed for communication.

- All the collected data are provided to the user through a Mobile application which was developed. Depending upon the Sensor values, Mobile Motor Pump controller waters the crop.



5.3 TECHNOLOGY ARCHITECTURES

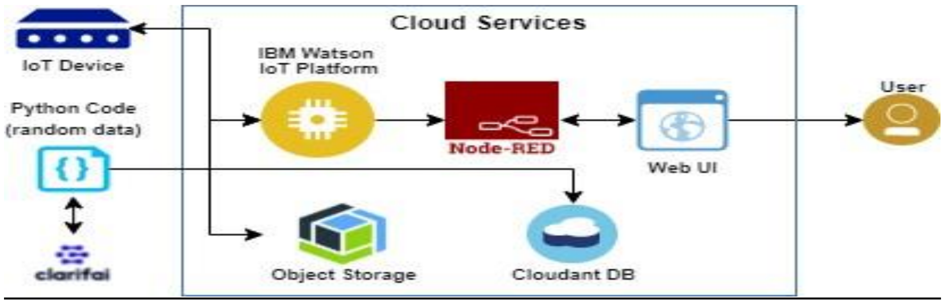


Table 1: Components and technologies

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson/node red
4.	Application Logic-3	Logic for a process in the application	IBM Watson/node red
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM Cloudant.
7.	Temperature sensor	Monitor the temperature	TMP36
8.	Humidity sensor	Monitor the humidity	DHT11
9.	Soil moisture sensor	Measure the amount of water in the soil	Soil maoisture sensor
10.	Weather monitoring	Monitor the weather	Temperature sensor

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Clarifai,Node- red	Software
2.	Security Implementations	Senisitive and private data must be protected from their protection untill the decision-making and storage stages.	Encryption process
3.	Scalable Architecture	Scalability is a major concern for IOT platform it has been shown that different architectural choices of IOT platform affect system capability and that automatic real time decision making is feasible in an environment composed of dozens of thousand.	Software
4.	Availability	Automatic adjustment of farming equipment made possible by linking information like crops/weather and temperature,humidity etc.	Software

5.	Performance	The ideas of implementing integerated sensors with sensing soil and envirenmental or ambient parameters in framing will be more efficient for overall monitoring .	Software

6.PROJECT PLANNING AND SCHEDULING

6.1 SPRINT DELIVERY PLAN

Sprint	Functional Requirement (Epic)	User story number	User story/task	Story points	Priority	Team members
Sprint-1		US-1	Create IBM Wastin IOT platform	6	High	Nivetha. S Saranya. Sakthi Priya k Rithika S
Sprint-1		US-2	Create a device and configure the IBM IOT platform	4	Medium	Nivetha. S Saranya. R Sakthi Priya k Rithika S
Sprint-2		US-3	IBM Watson IoT platform acts as the mediator to connect the web application to IoT	5	Medium	Nivetha. S Saranya. R Sakthi Priya k Rithika S

			devices, so create the IBM Watson IoT platform.			
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	Hidh	Nivetha. S Saranya. R Sakthi Priya k Rithika 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	Nivetha. S Saranya. R Sakthi Priya k Rithika S
Sprint-3		US-2	Create a node -Red service.	10	High	Nivetha. S Saranya. R Sakthi Priya k Rithika S
Sprint-3		US-1	Create a	5	Medium	Nivetha. S Saranya.

			database in cloudant DB to store location data			R Sakthi Priya k Rithika S
Sprint-4		US-2	Develop a python script to publish random sensor data such as temperature, moisture, soil and humidity to the IBM IoT platform	7	High	Nivetha. S Saranya. R Sakthi Priya k Rithika S
Sprint -4		US-3	Publish data to the IBM Cloud	8	High	Nivetha. S Saranya. R Sakthi Priya k Rithika S
Sprint -4		US-4	Create web UI in Node Red	10	High	Nivetha. S Saranya. R Sakthi Priya k Rithika S
Sprint -4		US-1	Display the image in the Node-RED web UI and also display the temperature, humidity, and soil moisture levels. Integrate the	10	High	Nivetha. S Saranya. R Sakthi Priya k Rithika S

			buttons in the UI to control the Motors.			

Project Tracker:

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	05Nov 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

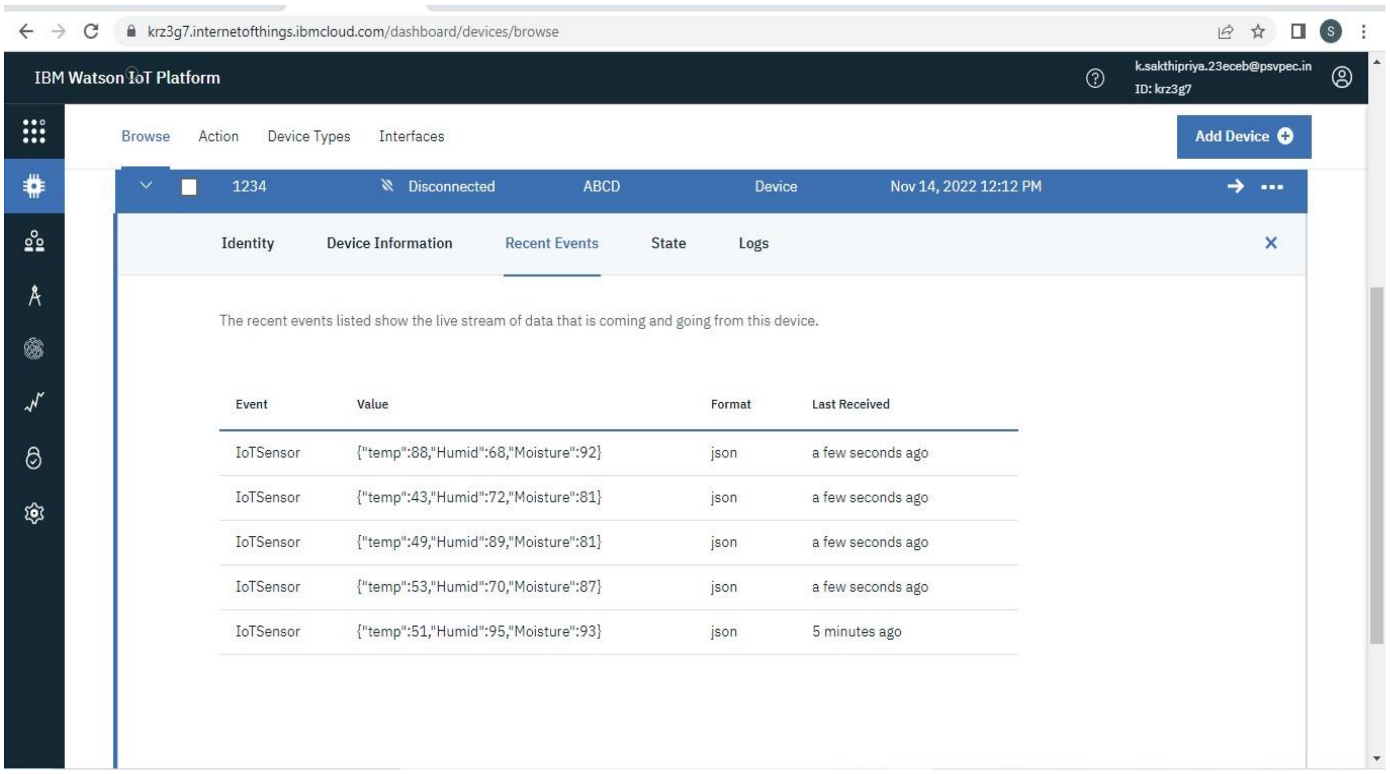
7. CODING AND SOLUTIONING

7.1 FEATURE 1

```
import random

import ibmiotf.application

import ibmiotf.device

from time import sleep

import sys

#IBM Watson Device Credentials.

organization = "krz3g7"

deviceType = "ABCD"
```

```
deviceId = "1234"

authMethod = "token"

authToken = "12345678"

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('Published 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()
```


krz3g7.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform — k.sakthipriya.23eceb@psvpec.in ID: krz3g7

Browse | Action | Device Types | Interfaces | Add Device

1234 | Disconnected | ABCD | Device | Nov 14, 2022 12:12 PM

Identity | Device Information | Recent Events | State | Logs

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

Event	Value	Format	Last Received
IoTSensor	{"temp":88,"Humid":68,"Moisture":92}	json	a few seconds ago
IoTSensor	{"temp":43,"Humid":72,"Moisture":81}	json	a few seconds ago
IoTSensor	{"temp":49,"Humid":89,"Moisture":81}	json	a few seconds ago
IoTSensor	{"temp":53,"Humid":70,"Moisture":87}	json	a few seconds ago
IoTSensor	{"temp":51,"Humid":95,"Moisture":93}	json	5 minutes ago

FEATURES

Output: Digital pulse high (3V) when triggered (motion detected) digital low when idle (no motion detected). Pulse lengths are determined by resistors and capacitors on the PCB and differ from sensor to sensor. Power supply: 5V-12V input voltage for most modules (they have a 3.3V regulator),but 5V is ideal in case the regulator has different specs.

BUZZER

Specifications

- RatedVoltage : 6V DC
- Operating Voltage : 4 to 8V
- Rated Current*: ≤30mA
- SoundOutput at 10cm* : ≥85dB
- Resonant Frequency : 2300 ±300Hz
- Tone: Continuous A buzzer is a loud noise maker.

Most modern ones are civil defense or air- raid sirens, tornado sirens, or the sirens on emergency service vehiclessuch as ambulances, police cars and fire trucks. There are two general types, pneumatic and electronic.

7.2 FEATURE 2

- Goodsensitivity to Combustible gas in wide range .
- Highsensitivity to LPG, Propane and Hydrogen .
- Longlife and low cost.
- Simpledrive circuit.

8. TESTING

8.1 TEST CASES

S.NO	Parameter	Values
1	Model summary	-
2	Accuracy	Training accuracy-95% Validationaccuracy-72%
3	Confidence score	Class detected-80% Confidencescore-80%

8.2 USER ACCEPTANCE TESTING

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the IoT Based smart crop protection system for agriculture project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

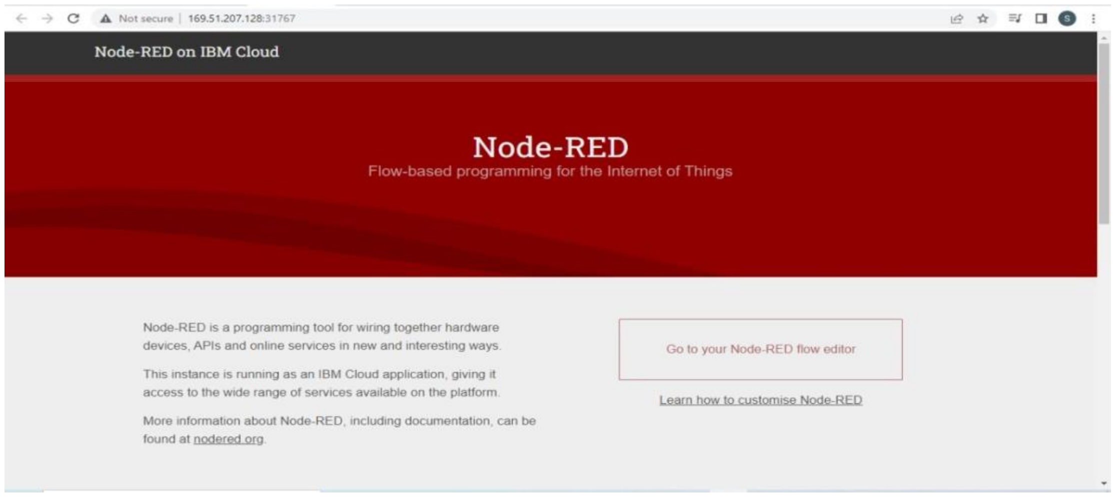
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	11	4	2	2	19
Duplicate	2	1	1	0	4
External	2	3	0	1	6
Fixed	11	2	2	20	35
Not	0	1	1	0	2

Reproduced					
Skipped	0	2	0	1	3
Won't Fix	0	5	2	1	8
Totals	26	18	8	25	77

1. 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	6	0	2	4
Client Application	45	0	3	42
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	10	0	1	9
Final Report Output	4	0	0	4
Version Control	3	0	1	2


Not secure | 169.51.207.128:31767
Node-RED on IBM Cloud
Go to your Node-RED flow editor
Learn how to customise Node-RED

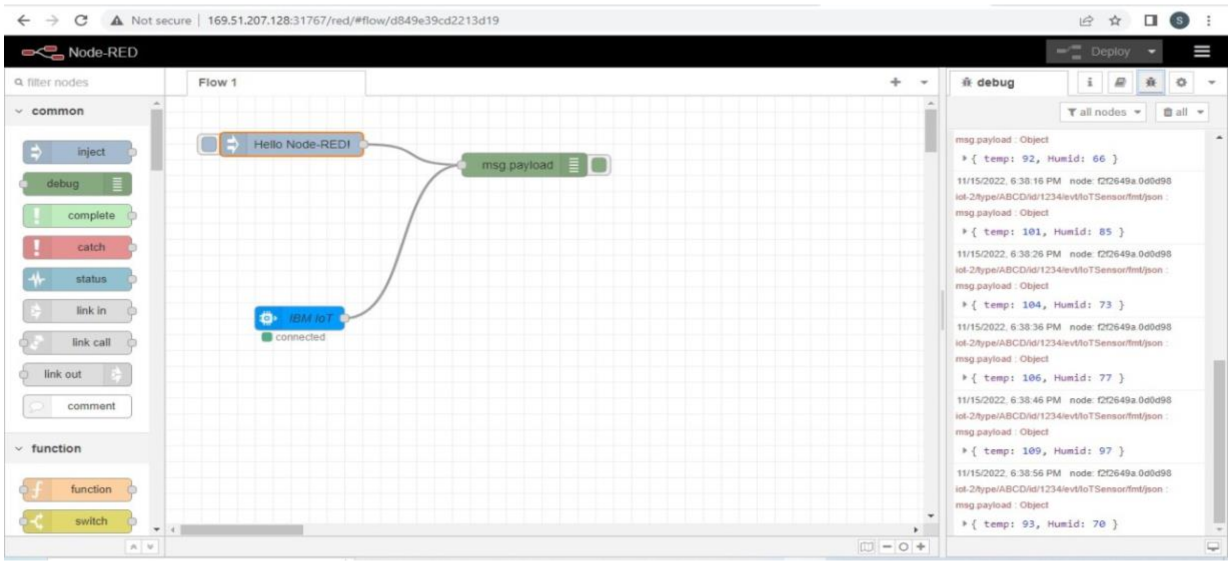
Node-RED

Flow-based programming for the Internet of Things

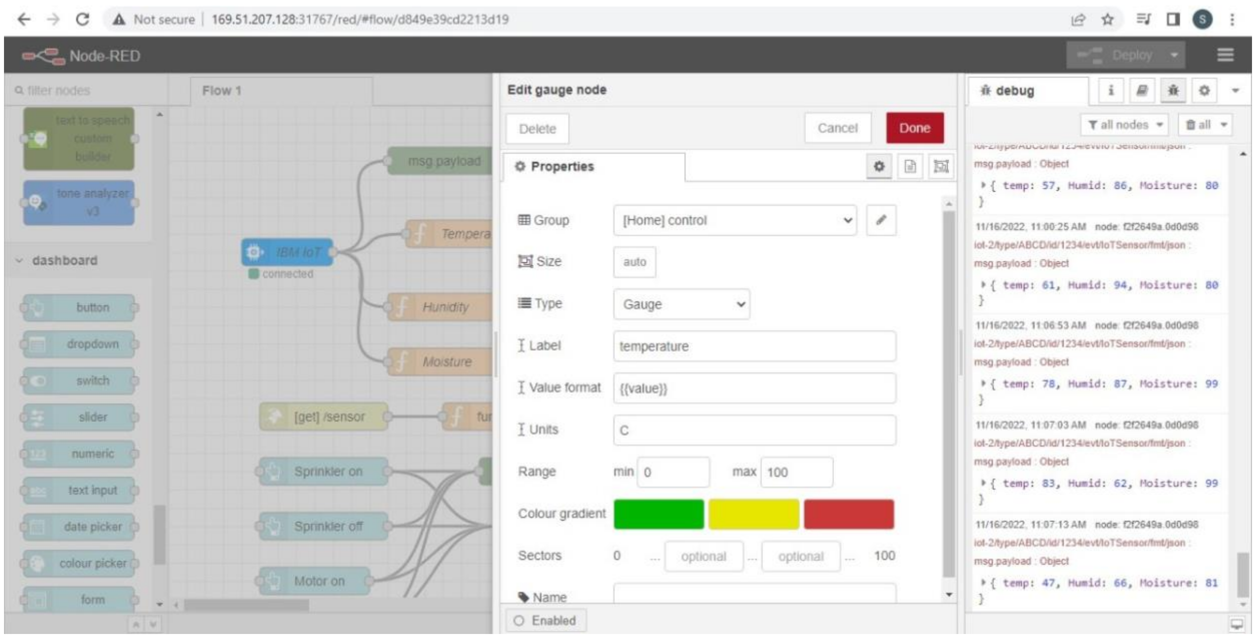
Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

This instance is running as an IBM Cloud application, giving it access to the wide range of services available on the platform.

More information about Node-RED, including documentation, can be found at nodered.org.



```
msg.payload : Object
▸{ temp: 92, Humid: 66 }
11/15/2022, 6:38:16 PM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 101, Humid: 85 }
11/15/2022, 6:38:26 PM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 104, Humid: 73 }
11/15/2022, 6:38:36 PM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 106, Humid: 77 }
11/15/2022, 6:38:46 PM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 109, Humid: 97 }
11/15/2022, 6:38:56 PM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 93, Humid: 70 }
```



Group	[Home] control
Size	auto
Type	Gauge
Label	temperature
Value format	{{value}}
Units	C
Range	min 0 max 100
Colour gradient	🟩 🟨 🟥
Sectors	0 ... optional ... optional ... 100
Name	

```
msg.payload : Object
▸{ temp: 57, Humid: 86, Moisture: 80
}
11/16/2022, 11:00:25 AM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 61, Humid: 94, Moisture: 80
}
11/16/2022, 11:06:53 AM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 78, Humid: 87, Moisture: 99
}
11/16/2022, 11:07:03 AM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 83, Humid: 62, Moisture: 99
}
11/16/2022, 11:07:13 AM  node: f2f2649a.0d0d98
iot-2/type/ABCD/id/1234/evt/IoTSensor/fmt/json :
msg.payload : Object
▸{ temp: 47, Humid: 66, Moisture: 81
}
```

9. RESULTS

The problem of crop vandalization by wild animals and fire has become a major social problem in current time.It requires urgent attention as no effective solution exists till date for this problem. Thus this project carries a great social relevance as it aims to address this problem. This project willhelp 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 achievingbetter crop yields thus leading to theireconomic wellbeing.

10. ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- Intelligent data collection.
- Waste reduction.
- Process automation.
- Animal monitoring.
- Competitive advantage.

DISADVANTAGES

- Smart Agriculture Cost
- Possibility for wrong Analysis of Weather Conditions
- High Cost
- Increased use of chemicals
- Uneven water distribution

11. CONCLUSION

A IoT Web Application is built for smart agricultural system using Watson IoT Platform, Watsonsimulator, IBM cloud and Node red.

12. FEATURE SCOPE

In the future, there will be very large scope, this project can be made based on Image processing in which wild animaland fire can be detected by cameras and if it comes towards farmthen system will be directly activated through wireless networks. Wild animals can also be detected by using wireless networks such as laser wireless sensors and by sensing this laser or sensor's security system will beactivated.

13. APPENDIX

SOURCE CODE

```
import time importsys import ibmiotf.application # to

installpip

install ibmiotf importibmiotf.device

# Provide your IBM Watson Device Credentials organization = "krz3g7" #
```

```
replace the ORG ID deviceType = "ABCD" #replace the Device
type deviceId = "1234" # replace Device ID authMethod = "token"
authToken = "12345678" # Replace the authtoken

def myCommandCallback(cmd): # function for Callbackif

cm.data['command'] == 'motoron':
print("MOTOR ON IS RECEIVED")
elif cmd.data['command'] == 'motoroff':print("MOTOR OFF IS RECEIVED")
if cmd.command == "setInterval":
else:
if 'interval' not in cmd.data:
print("Error - command is missing requiredinformation: 'interval'")
interval = cmd.data['interval']
elif cmd.command == "print":
if 'message' not in cmd.data:
print("Error - commandis missing requiredinformation: 'message'")
else:output = cmd.data[‘message’]
print(output)
try:
deviceOptions = {"org": organization,"type": deviceType, "id": deviceId, authmethod":
authMethod,
    "auth-token": authToken} deviceCli
= ibmiotf.device.Client(deviceOptions)#
............................................
exceptException 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 oftype
```

```
"greeting"

10 times

deviceCli.connect()

while True:

deviceCli.commandCallback = myCommandCallback

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

SENSOR.PY

```
import time import

sysimport

ibmiotf.application

importibmiotf.device

import random

# Provide your IBM Watson Device Credentials organization = "krz3g7" #

replace the ORG ID deviceType = "ABCD" #replace the Device

type deviceId = "1234" # replace Device ID authMethod = "token"

authToken = “12345678” # 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)

#.............................................

exceptException 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 oftype
"greeting"
```

```
10 times

deviceCli.connect()

while True:

temp=random.randint(0,1

00)

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 IoTF")time.sleep(1)

deviceCli.commandCallback = myCommandCallback


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

Node-RED FLOW :

```
[

{

"id":"625574ead9839b34",

"type":"ibmiotout", "z":"630c8601c5ac3295",


"authentication":"apiKey",

"apiKey":"ef745d48e395ccc0",
“outputType”:”cmd”,

“deviceId”:1234”,

“deviceType”:”ABCD”,

“eventCommandType”:”data”,

“format”:”json”,

“data”:”data”,

“qos”:0,

“name”:”IBM IoT”,

“service”:”registere

D”,”x”:680,

“y”:220,

“wires”:[]

},

{

“id”:”4cff18c3274cccc4”,”type”:”ui_button”,

“z”:”630c8601c5ac3295”,

“name”:””,

“group”:”716e956.00eed6c”,

“order”:2,

“width”:”0”,

“height”:”0”,

“passthru”:false,

“label”:”MotorON”,

“tooltip”:””,

“color”:””,


“bgcolor”:””,

“className”:””,
```

“icon”:””,

“payload”:”{\”command\”:\”motoron\”}”,

“payloadType”:”str”,

“topic”:”motoron”,

“topicType”:”sTr”,”

X”:360,

“y”:160, “wires”:[[“625574ead9839b34”]]},

{

“id”:”659589baceb4e0b0”,

“type”:”ui_button”, “z”:”630c8601c5ac3295”,

“name”:””,

“group”:”716e956.00eed6c”,

“order”:3,

“width”:”0”,

“height”:”0”,

“passthru”:true,

“label”:”MotorOFF”,

“tooltip”:””,

“color”:””,

“bgcolor”:””,

“className”:””,

“icon”:””,

“payload”:”{\”command\”:\”motoroff\”}”,

“payloadType”:”str”,

“topic”:”motoroff”,

“topicType”:”sTr”,”

X”:350,

“y”:220, “wires”:[[“625574ead9839b34”]]},

{“id”:”ef745d48e395ccc0”,”

Type”:”ibmiot”,

“name”:”weather_monitor”,”keepalive”:”60”,

“serverName”:””,

“cleansession”:true,

“appId”:””,

“shared”:false},

{“id”:”716e956.00eed6c”,

“type”:”ui_group”,

“name”:”Form”,

“tab”:”7e62365e.b7e6b8

“,”order”:1,

“disp”:true,

“width”:”6”,

“collapse”:falSe},

{“id”:”7e62365e.b7e6b8”,

“type”:”ui_tab”,

“name”:”contorl”,

“icon”:”dashboard

“,”order”:1,

“disabled”:false,

“hidden”:false}

]

[

{

“id”:”b42b5519fee73ee2”, “type”:”ibmiotin”, “z”:”03acb6ae05a0c712”,

“authentication”:”apiKey”,

“apiKey”:”a-krz3g7-jwi612ftm”,

“inputType”:”evt”,

“logicalInterface”:””,

“ruleId”:””,

“deviceId”:”1234”,

“applicationId”:””,

“deviceType”:”ABCD”,

“eventType”:”+”,

“commandType”:””,

“format”:”json”,

“name”:”IBMIoT”,

“service”:”registered”,

“allDevices”:””,

“allApplications”:””,

“allDeviceTypes”:””,

“allLogicalInterfaces”:””,

“allEvents”:true,

“allCommands”:””,

“allFormats“:””,

“qos”:0,

“x”:270,

“y”:180,

“wires”:[[“50b13e02170d73fc”,”d7da6c2f5302ffaf”,”a949797028158f3f”,”a71f164bc3
78bcf1”]]

},

{

“id”:”50b13e02170d73fc“,

“type”:”function”,

“z”:”03acb6ae05a0c712

“,”

Name”:”Soil Moisture”,

“func”:”msg.payload = msg.payload.soil;\nglobal.set(‘s’,msg.payload);\nreturn msg;”,“outputs”:1,“noerr”:0,

“initialize“:””,

“finalize”:””,

“libs”:[],

“x”:490,

“y”:120,

“wires”:[[“a949797028158f3f”,”ba98e701f55f04fe”]]

}

“id”:”d7da6c2f5302ffaf”,”

Type”:”function”,

“z”:”03acb6ae05a0c712”,

“name”:”Humidity”,

“func”:”msg.payload = msg.payload.pulse;\nglobal.set(‘p’,msg.payload)\nreturn msg;”,

“outputs”:1,“noerr”:0,

“initialize

“:””,

“finalize”:””,

“li

Bs“:[],“x“:480,

“y”:260, “wires”:[[“a949797028158f3f”,”70a5b076eeb80b70”]]

},
{
“id”:”a949797028158f3f
“,
“type”:”debug”,
“z”:”03acb6ae05a0c712“,”
Name”:”IBMo/p”,
“active”:true,
“tosidebar”:true,
“console”:false,
“tostatus”:false,
“complete”:”payload”,
“targetType”:”msg”,
“statusVal”:””,
“statusType”:”auto”,
“x”:780,
“y”:180,
“wires”:[]
};
{
“id”:”70a5b076eeb80b70”,
“type”:”ui_gauge”,
“z”:”03acb6ae05a0c712”,
“name”:””,
“group”:”f4cb8513b95c98a4”,

“order”:6,
“width”:”0”,

“height”:”0”,

“gtype”:”gage”,

“title”:”Humidity”,

“label”:”Percentage(%)”,

“format”:”{{value}}

“,”min”:0,

“max”:”100”,

“colors”:[“#00b500”,”#e6e600”,”#ca3838”],”seg1”:””,“seg2”:””,

“className

“:””,”x”:860,

“y”:260,

“wires”:[]

},

{

“id”:”a71f164bc378bcf1”,”type”:”function”,

“z”:”03acb6ae05a0c712”,

“name”:”Temperature”,

“func”:”msg.payload=msg.payload.temp;\nglobal.set(‘t’,msg.payload);\nreturn msg;”,”outputs”:1,

“noerr”:0,

“initialize

“:””,

“finalize”:””,“liB

“:[

],

“:490,“y”:360,

“wires”:[[“8e8b63b110c5ec2d”,”a949797028158f3f”]]

},

{

“id”:”8e8b63b110c5ec2d”,

“type”:”ui_gauge”,

“z”:”03acb6ae05a0c712”,

“name”:””,

“group”:”f4cb8513b95c98a4”,

“order”:11,

“width”:”0”,

“height”:”0”,

“gtype”:”gage”,

“title”:”Temperature”,

“label”:”DegreeCelcius”,

“format”:”{{value}}”,

“min”:0,

“max”:”100”,

“colors”:[“#00b500”,”#e6e600”,”#ca3838”],”

Seg1”:””,

“seg2”:””,

“className

“:””,

“x”:790,

“y”:360,

“wires”:[]

},

{

“id”:”ba98e701f55f04fe”,

“type”:”ui_gauge”,

“z”:”03acb6ae05a0c712”,

“name”:””,

“group”:”f4cb8513b95c98a4”,

“order”:1

“width”:”0”,

“height”:”0”,

“gtype”:”gage”,

“title”:”Soil Moisture”,

“label”:”Percentage(%)”,

“format”:”{{value}}

“,”min”:0,

“max”:”100”,

“colors”:[“#00b500”,”#e6e600”,”#ca3838”],”

Seg1”:””,

“seg2”:””,

“className

“:””,

“x”:790,

“y”:120,

“wires”:[]

},

{

“id”:”a259673baf5f0f98

“,”

Type”:”httpin”,

“z”:”03acb6ae05a0c712

“,”name”:””,

```
“url”:”/sensor”,
“method”:”geT”,
“upload”:falsE,
“swaggerDoc”
:””,”x”:370,
“y”:500,
“wires”:[[“18a8cdbf7943d27a”]]
},
{
“id”:”18a8cdbf7943d27a”,”type”:”function”,
“z”:”03acb6ae05a0c712”
“name”:”httpfunction”,
“func”:”msg.payload{\”pulse\”:global.get(‘p’),\”temp\”:global.get(‘t’),\”soil\”:global.get( ‘s’)};\
Msg;”,
“outputs”:1,
“noerr”:0,
“initialize”:””,
“finalize”:””,
“liBs
“:[
],“x“:630,
“y”:500, “wires”:[[“5c7996d53a445412”]]
},
{
“id”:”5c7996d53a445412“,
“type”:”httpresponse”,

“z”:”03acb6ae05a0c712
```

“,”name”:””,

“statusCode”:””,

“header

S”:{},

“x”:870,

“y”:500,

“wires”:[]

},

{

“id”:”ef745d48e395ccc0”,

“type”:”ibmiot”,

“name”:”weather_monitor”,

“keepalive”:”60”,

“serverName”:””,

“cleansession”:true,

“appId”:””,

“shared”:false},

{

“id”:”f4cb8513b95c98a4”,”

Type”:”ui_group”,

“name”:”monitor”,

“tab”:”1f4cb829.2fdee8“,

“Order”:2,

“disp”:

True,“width“:”6”,

“collapse”:fAlse,

“className

“:””

},

{

“id”:”1f4cb829.2fdee8”,

“type”:”ui_tab”,

“name”:”Home”,

“icon”:”dashboard

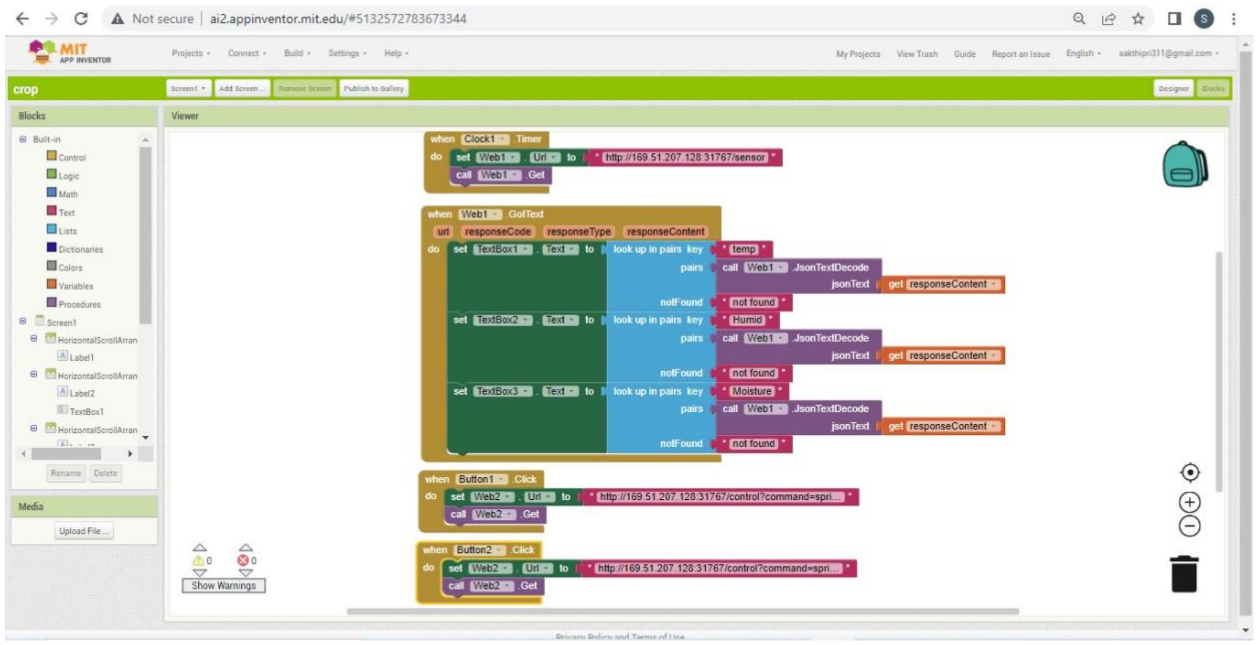
“,”

Order”:3,

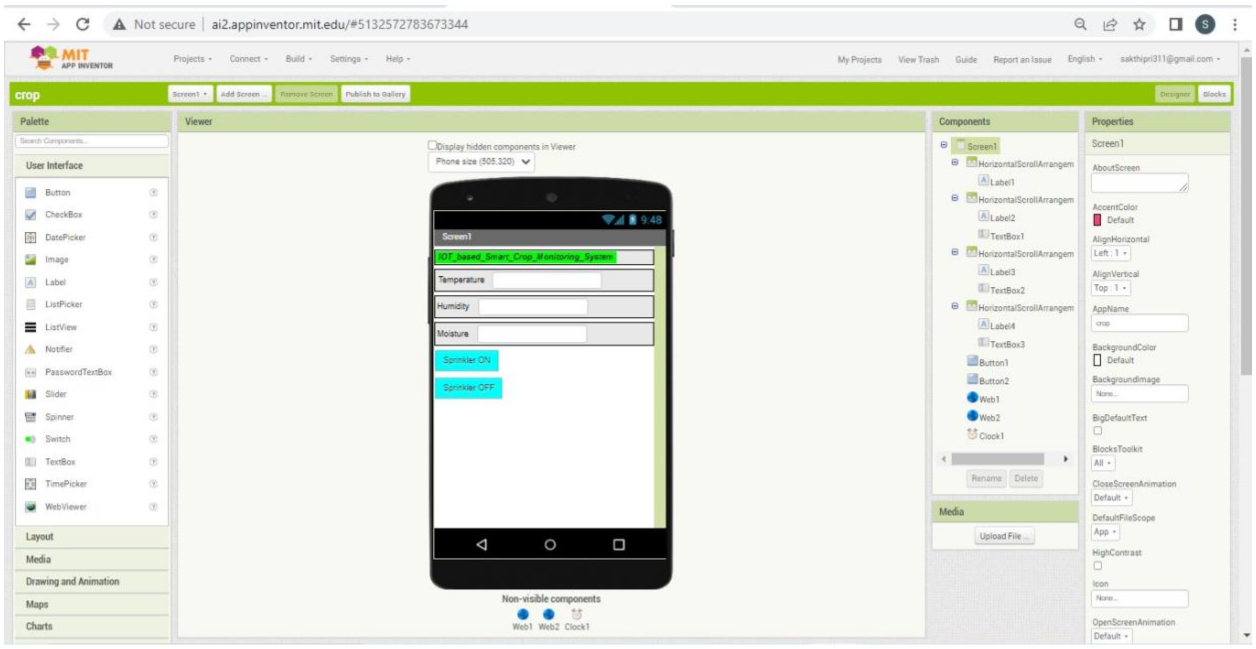
“disabled”:false,

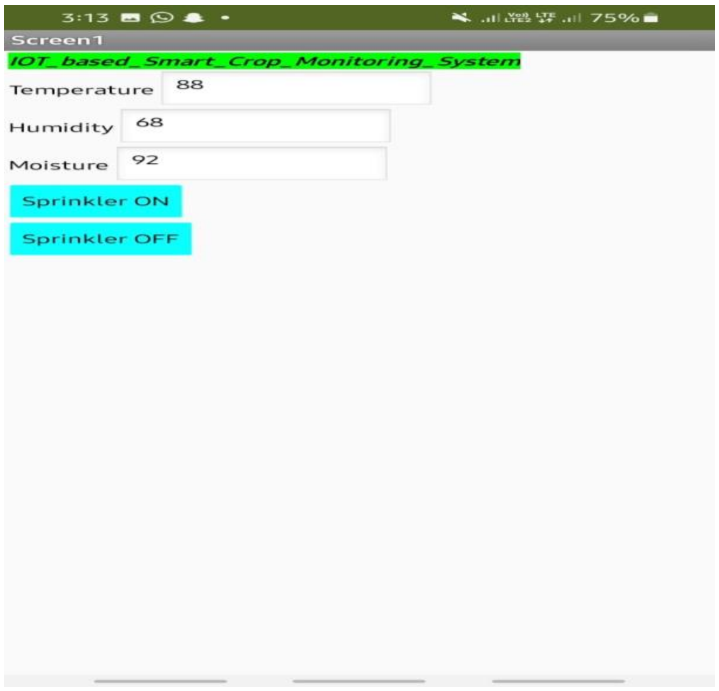
“hidden”:false }

GITHUB AND PROJECT DEMO LINK



```
when Clock1 .Timer
do  set Web1 . Url to "http://169.51.207.128:31767/sensor"
    call Web1 .Get

when Web1 .GotText
 url  responseCode  responseType  responseContent
do  set TextBox1 . Text to  look up in pairs  key  "temp"
                                              pairs  call Web1 .JsonTextDecode
                                                                 jsonText  get responseContent
                                           notFound  "not found"
    set TextBox2 . Text to  look up in pairs  key  "Humid"
                                              pairs  call Web1 .JsonTextDecode
                                                                 jsonText  get responseContent
                                           notFound  "not found"
    set TextBox3 . Text to  look up in pairs  key  "Moisture"
                                              pairs  call Web1 .JsonTextDecode
                                                                 jsonText  get responseContent
                                           notFound  "not found"

when Button1 .Click
do  set Web2 . Url to "http://169.51.207.128:31767/control?command=spri..."
    call Web2 .Get

when Button2 .Click
do  set Web2 . Url to "http://169.51.207.128:31767/control?command=spri..."
    call Web2 .Get
```





Temperature 88

Humidity 68

Moisture 92