SETHU INSTITUTE OF TECHNOLOGY, KARIAPATTI

An Autonomous Institution

B.E. ELECTRONICS AND COMMUNICATION ENGINEERING

IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

INNOVATIVE PROJECT REPORT

Submitted by

B.MADHUMITHA	92172019104083
E.LAVANYAA	92172019104082
S.KAMALI	92172019104071
V.S.KRISHNA PRIYA	92172019104080

TEAM ID: PNT2022TMID17253

for the course of

PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP
(Naalaiya Thiran Program)

INDEX

1.INTRODUCTION

1.1 Project Overview

The term "Internet of Things" refers to the connection of objects, equipment, vehicles, and other electronic devices to a network for the purpose of data exchange (IoT). The Internet of Things (IoT) is increasingly being utilised to connect objects and collect data. As a result, the Internet of Things' use in agriculture is crucial. The idea behind the project is to create a smart agriculture system that is connected to the internet of things.

Moisture sensor is interfaced with Arduino Microcontroller to measure the moisture level in soil and relay is used to turn ON & OFF the motor pump for managing the excess water level. It will be updated to authorities through IOT. Temperature sensor connected to microcontroller is used to monitor the temperature in the field. Image processing techniques with IOT is followed for crop protection against animal attack.

Automatic crop maintenance and protection using embedded and IOT Technology. This proposed system provides many facilities which helps the farmers to maintain the crop field without much loss.This prototype can be developed as product with minimum cost with high protection.This can be developed to a scalable product by using solution sensors and transmitting the data through Wireless Sensor Network and Analysing the data in cloud and operation is performed using robots. As a result, it's critical to water the plants wisely in order to maximise yield per unit space and so achieve good output.

1.2 Purpose

Agriculture is essential to India's economy and people's survival. Agriculture is a substantial source of revenue for Indians and has a huge impact on the Indian economy. Crop development is essential for enhanced yield and higher-quality delivery.Smart agriculture, which comprises modernising present agricultural systems, is the only answer to this challenge.

The purpose of this project is to create an embedded-based soil monitoring and irrigation system that will reduce manual field monitoring and provide information via a mobile app. The suggested strategy attempts to use automation and Internet of Things technologies to make agriculture smarter. Crop growth monitoring and selection, irrigation decision assistance, and other uses are possible thanks to the Internet of Things (IoT). The method is intended to help farmers increase their agricultural output. Development of an effective loT-based smart irrigation system is also a crucial demand for farmers in the field of agriculture. The main purpose of this study is to develop a smart agricultural system that utilises cutting-edge technologies such as Arduino, Internet of Things, and wireless sensor networks. Through automation, the research tries to take use of emerging technologies such as the Internet of Things (IoT) and smart agriculture. The capacity to monitor environmental factors is a critical component in increasing crop efficiency. The purpose of this study is to develop a system that can monitor temperature, humidity, wetness, and even the movement of animals that might damage crops in agricultural areas using sensors, and then send an SMS notification as well as a notification on the app developed for the same to the farmer's smartphone via Wi-Fi/3G/4G if there is a discrepancy.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

- It is critical to water the plants wisely in order to maximise yield per unit space and so achieve good output. The capacity to monitor environmental factors is a critical component in increasing crop efficiency.
- Traditional irrigation methods, such overhead sprinklers and flood irrigation, are inefficient.
- Due to in sufficient labour forces. Asks suggestions from surrounding Improper maintenance of crops. Lack of knowledge among farmers in usage of fertilizers and hence crops are affected.
- Due to various environmental factors such as temperature climate, topography and soil quality which results in crop destruction.
- Consumes more time in cropland.
- Searching for an alternative solution for protecting crops from wild animals attacks, birds and pests.

2.2 REFERENCES

[1] Krunal Mahajan1, Riya Parate2, Ekta Zade3, Shubham Khante4, Shishir Bagal5," REVIEW PAPER ON SMART CROP PROTECTION SYSTEM", International Research Journal of Engineering and Technology (IRJET), Volume: 08, issue 02 Feb 2021.

[2]https://www.electronicshub.org/arduino-flamesensor-interface/#:~:text=Flame%20Sensor%20 has%20three%20pins,fire%2C%20a%20Buzzer%20is%20used.

[3] Dr.M. Chandra, Mohan Reddy, Keerthi Raju KamakshiKodi, BabithaAnapalliMounikaPulla, "SMART CROP PROTECTION SYSTEM FROM LIVING OBJECTS AND FIRE USING ARDUINO", Science, Technology and Development, Volume IX Issue IX, pg.no 261- 265, Sept 2020.

[4] 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.

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

[6] 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.

[7] Tanmay Baranwal" Development of IOT based Smart Security and Monitoring Devices for Agriculture", Department of Computer Science Lovely Professional University Phagwara, Punjab, IEEE-2016.

[8] M. Sathishkumar1, S.Rajini "Smart Surveillance System Using PIR Sensor Network and GSM" International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume4 Issue 1, January 2015.

2.3 PROBLEM STATEMENT DEFINITION

In traditional irrigation systems we require an operator or farmer to put water on crops but he does not come to know which crop require how much amount of water to get proper amount of yields, They waste a lot of water and may even make people sick by causing fungus growth in the soil due to too much moisture.Due to the scarcity of water, an automated irrigation system is essential for water conservation and, as a result, agricultural profitability. Irrigation consumes around 85% of the world's total accessible water resources. This need is projected to increase in the coming years as the population grows.

Cope with climate change, soil erosion and biodiversity loss. Satisfy consumers' changing tastes and expectations. Meet rising demand for more food of higher quality. Invest in farm productivity. Farmers must meet the changing needs of our planet and the expectations of regulators, consumers, and food processors and retailers.

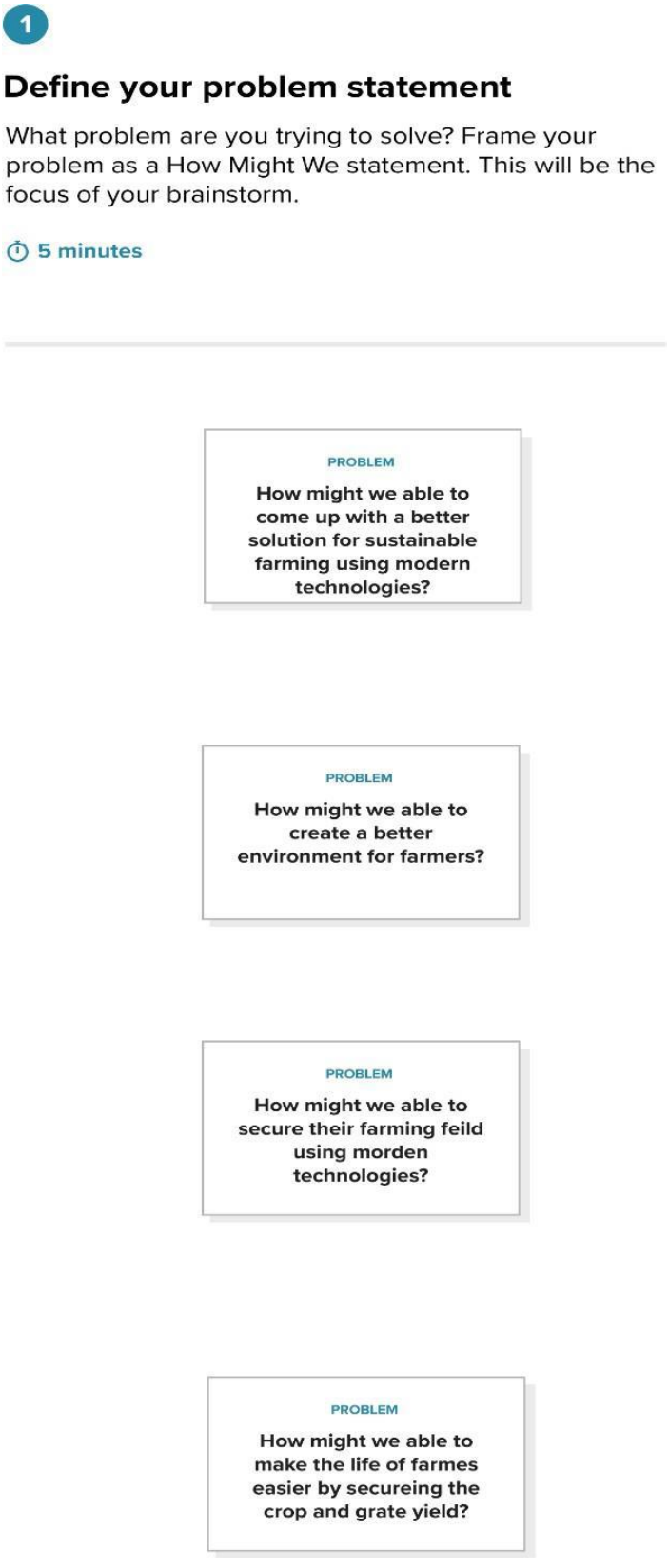
There are increasing pressures from climate change, soil erosion and biodiversity loss and from consumers' changing tastes in food and concerns about how it is produced. And the natural world that farming works with – plants, pests and diseases – continue to pose their own challenges.

3.IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

Step-1: Team Gathering, Collaboration and select the Problem Statement



Step-2: Brainstorm, Idea listing and Grouping

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

@ 20 minutes

Step-3: Idea Prioritization

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

3.3 Proposed Solution

The idea behind the project is to create a smart agriculture system that is connected to the internet of things. Moisture sensor is interfaced with Arduino Microcontroller to measure the moisture level in soil and relay is used to turn ON & OFF the motor pump for managing the excess water level. It will be updated to authorities through IOT.

Temperature sensor connected to microcontroller is used to monitor the temperature in the field. Image processing techniques with IOT is followed for crop protection against animal attack.

Automatic crop maintenance and protection using embedded and IOT Technology. This proposed system provides many facilities which helps the farmers to maintain the crop field without much loss. This prototype can be developed as product with minimum cost with high protection. This can be developed to a scalable product by using solution sensors and transmitting the data through Wireless Sensor Network and Analysing the data in cloud and operation is performed using robots. As a result, it's critical to water the plants wisely in order to maximise yield per unit space and so achieve good output.

3.4 Problem Solution Fit

By seeing surrounding cropland with installing machineries. Moisture sensor interfaced with Arduino Microcontroller to measure the moisture level in soil and relay isused to turn ON and OFF the motor pump for managing the excess water level. It will be updated to authorities through IOT.

Using different platforms/social media to describe the working and uses of smart crop protection device. Hearing about innovative technologies and effective solutions. Temperature sensor connected to microcontroller is used to monitor the temperature in the field. The optimum temperature required for crop cultivation is maintained using IOT based fertilizing methods are followed, to minimize the negative effects on growth of crops while using fertilizers.

Mental frustrations due to insufficient production of crops. Felt smart enough to follow the available technologies with minimum cost. Giving awareness among farmers about the application of the device. Image processing techniques with IOT is followed for crop. protection against animal attacks.

4.REQUIREMENT ANALYSIS

4.1 Functional requirement

- Sense animals nearing the crop field & sounds alarm to woo them away as well as sends SMS to farmer using cloud service.
- The Data like values of Temperature, Humidity, Soil moisture Sensors are received via SMS.
- Based on the sensor data value to get the information about the present of farming land.
- The User needs take action like destruction of crop residues, deep plowing, crop rotation, fertilizers, strip cropping, scheduled planting operations.

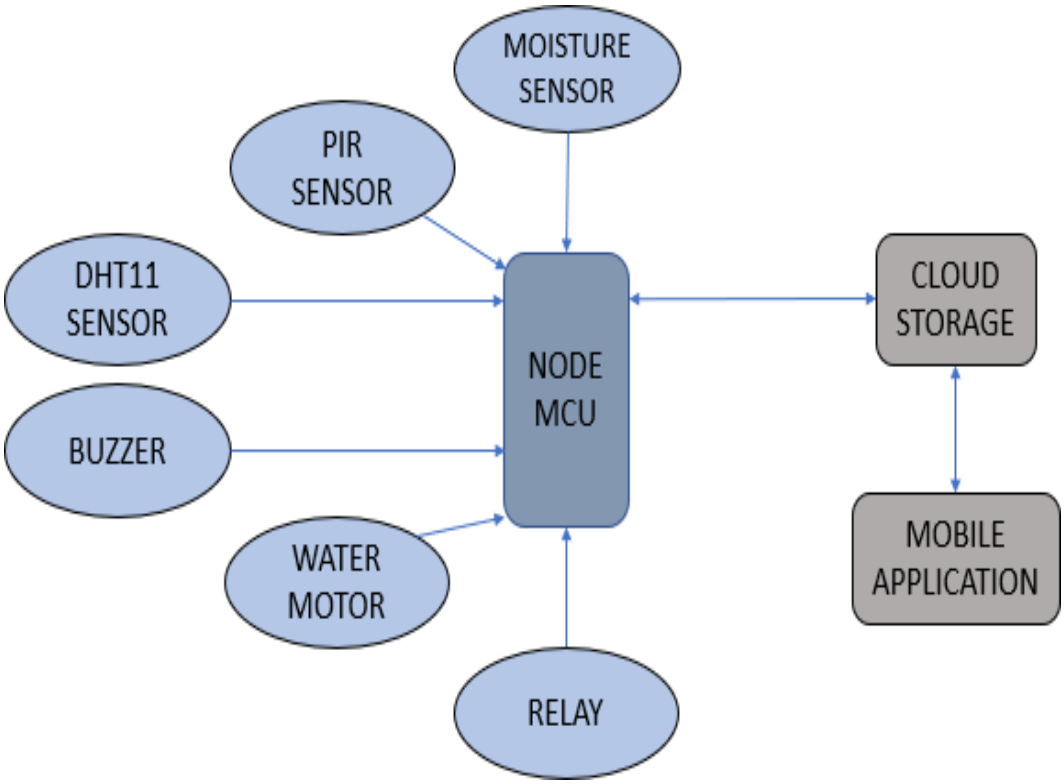
4.2 Non functional requirement

- Mobile Support Users must be able to interact in the same roles & tasks on computers & mobile devices where practical, given mobile capabilities.
- Data requires secure access to must register and communicate securely on devices and authorized users of the system who exchange information must be able to do.
- It has a capacity to recognize the disturbance near the field and doesn't give false caution signal.

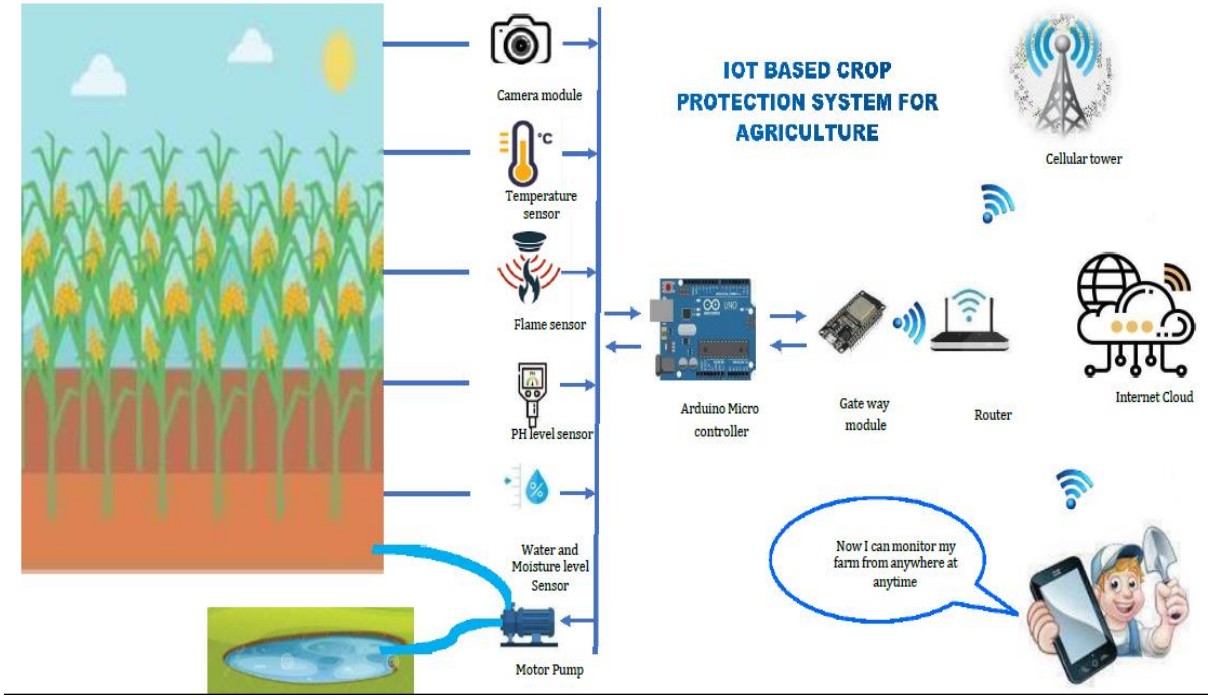
- Must provide acceptable response times to users regardless of the volume of data that is stored and the analytics that occurs in background. Bidirectional, near real-time communications must be supported. This requirement is related to the requirement to support industrial and device protocols at the edge.
- IOT Solutions and domains demand highly available systems for 24 x 7 operations. Isn't a critical production application, which means that operations or production don't go down if the IOT solution is down.
- System must handle expanding load & data retention needs that are based on the upscaling of the solution scope, such as extra manufacturing facilities and extra buildings.

5.PROJECT DESIGN

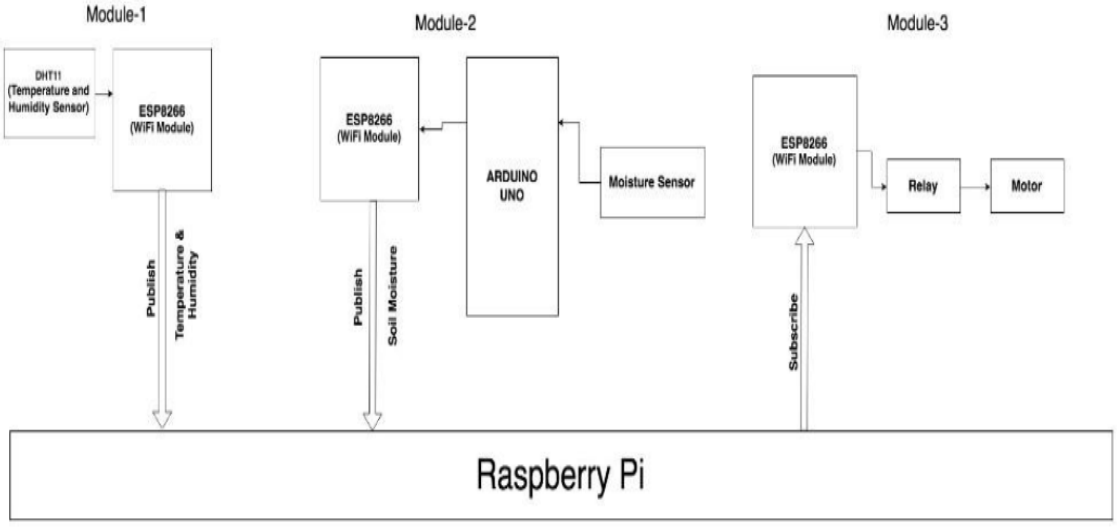
5.1 Data Flow Diagrams



5.2 Solution Architecture



Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story number	User Story/Task	Acceptance Criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	User can enter into the webapplication	I can access my account / dashboard	High	Sprint 1
		USN-2	User can register their Credentials like e-mail id and password	I can receive Confirmation email & click confirm	High	Sprint 1
	Login	USN-3	User can log into the application by entering email &password	I can log into my account	High	Sprint1
	Dashboard	USN-4	User can view thetemperature	I can view the data given by the device	High	Sprint 2
		USN-5	User can view the level 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 the information	I can view the data givenby the device	High	Sprint 3
Customer	Working	USN-1	User acts according to The alert given by the device	I can get the data work according to it	High	Sprint 3
		USN-2	User turns ON the Water motors/ Buzzer/ Relay when the disturbance occur on the field	I can get the data work according to it	High	Sprint 4
Customer Care Executive	Action	USN-1	User solves the problem when some faces anyusage issues	I can solve the issues when someone fails to understand the procedure	High	Sprint 4
Administration	Administration	USN-1	User stores every information	I can store the gained information	High	Sprint 4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

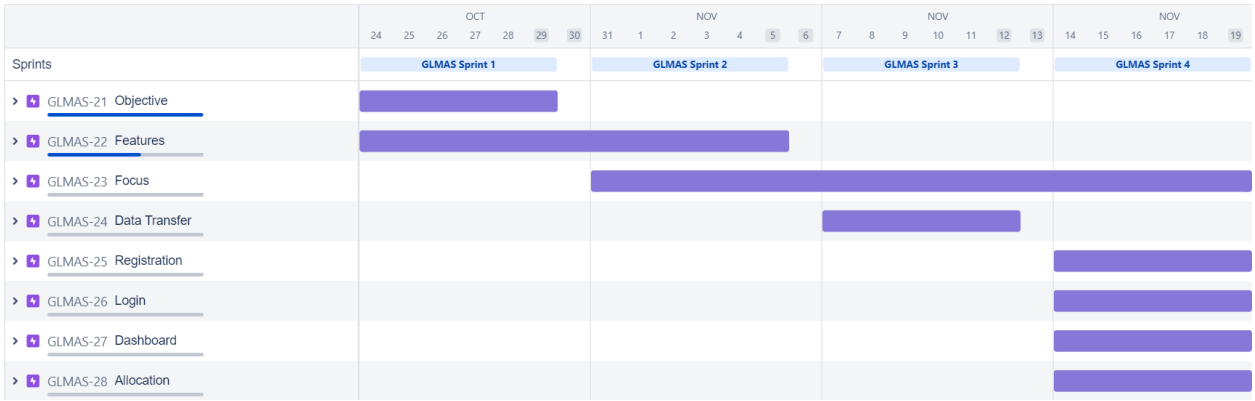
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	User can enter into the webapplication	2	LOW	Madhumitha
Sprint-1	Registration	USN-2	User can register their Credentials like e-mail id and password	4	MEDIUM	Lavanyaa
Sprint-1	Login	USN-3	User can log into the application by entering email &password	4	MEDIUM	Kamali, Krishnapriya
Sprint-2	Dashboard	USN-4	User can view thetemperature	5	HIGH	Lavanyaa,Kamali
Sprint-2	Dashboard	USN-5	User can view the level of sensor monitoring value	5	HIGH	Madhumitha, Krishnapriya
Sprint-3	Usage	USN-6	User can view the web page and get the information	5	HIGH	Lavanyaa, Krishnapriya

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Working	USN-7	User acts according to The alert given by the device	5	HIGH	Madhumitha, Kamali
Sprint-4	Working	USN-8	User turns ON the Water motors/ Buzzer/ Relay when the disturbance occur on the field	4	HIGH	Lavanyaa, Madhumitha
Sprint-4	Action	USN-9	User solves the problem when some faces Any usage issues	4	HIGH	Kamali
Sprint-4	Administration	USN-10	User stores every information	2	MEDIUM	Krishnapriya

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	10	29 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	05 Nov 2022	10	05 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	12 Nov 2022	10	12 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022	10	19 Nov 2022

6.3 Reports from JIRA



7.CODING AND SOLUTIONING

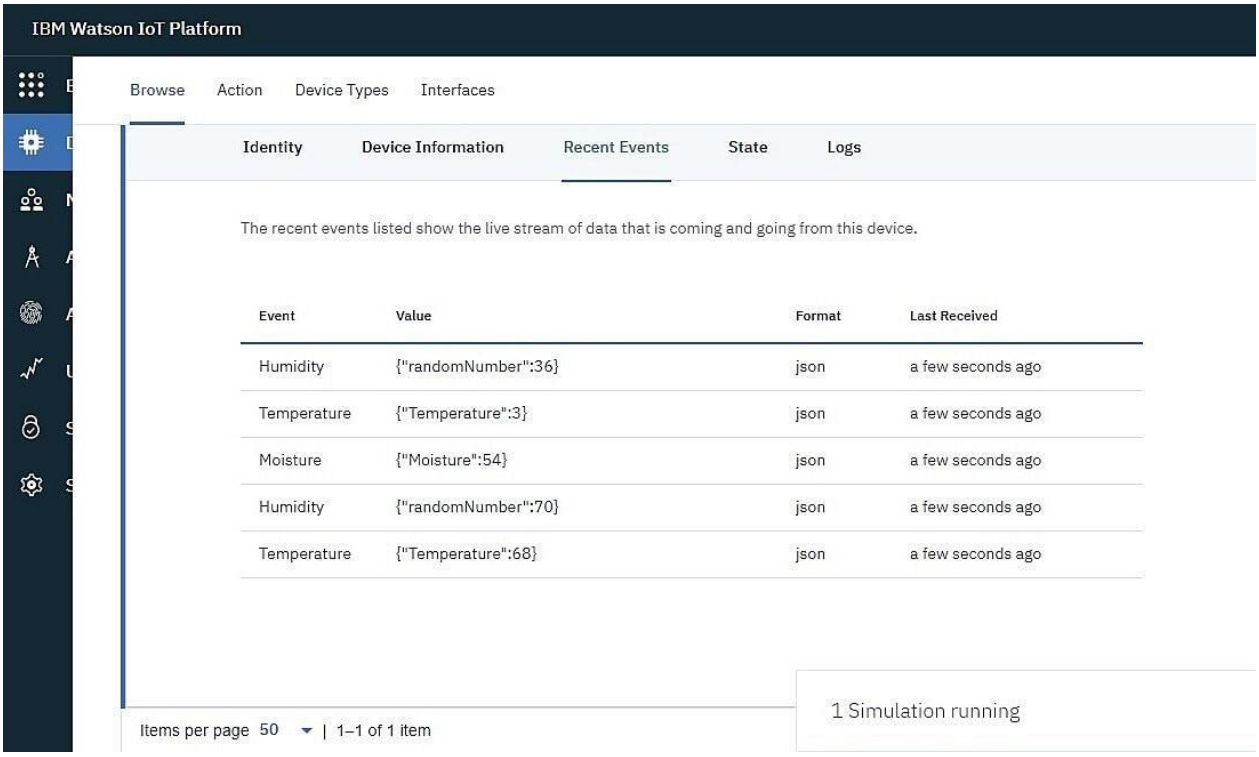
FEATURE-1

```
import random
import ibmiotf.application
import ibmiotf.device
from time import sleep
import sys
#IBM Watson Device Credentials.
organization = "op701j"
deviceType = "Dheepak"
deviceId = "Dheepak50"
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('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()
```



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 DC

- 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.

FEATURE-2:

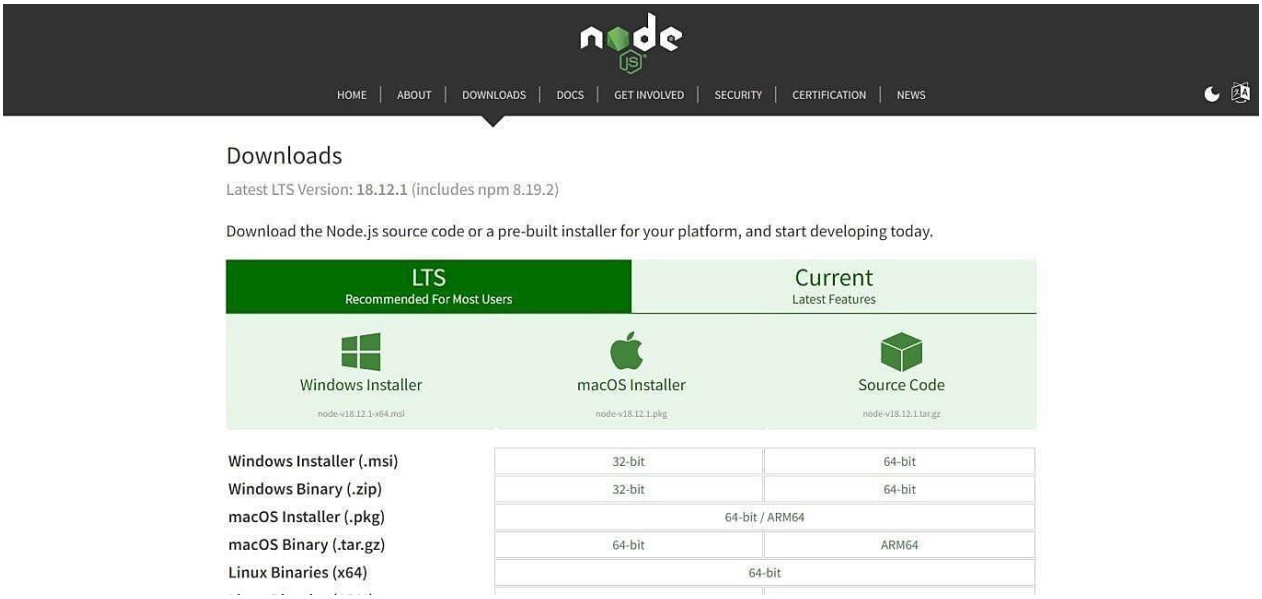
i. Goodsensitivity to Combustible gas in wide range .
ii. Highsensitivity to LPG, Propane and Hydrogen .
iii. Longlife and low cost.
iv. Simpledrive circuit.

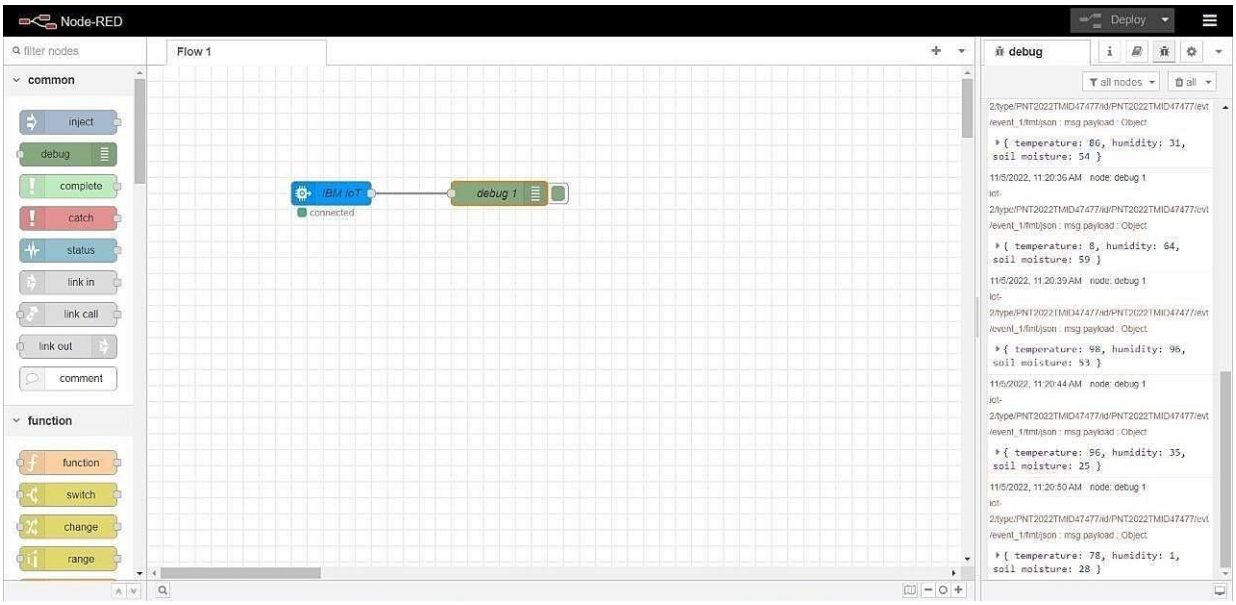
8.TESTING

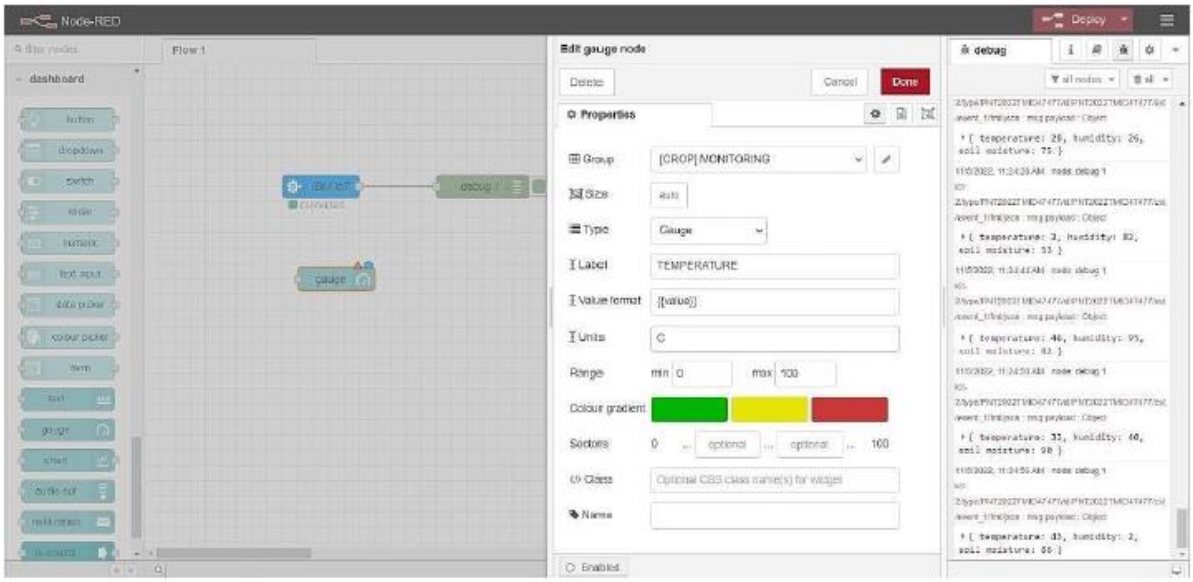
TEST CASES:

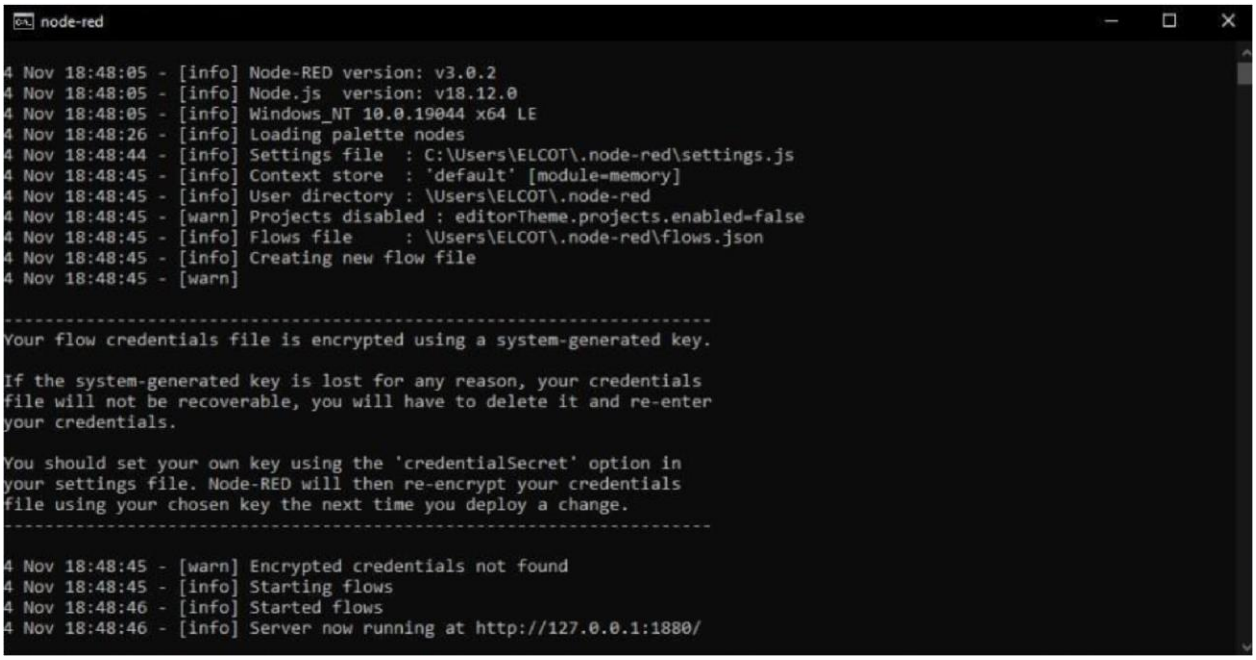
sno	parameter	Values	Screenshot
1	Model summary	-	
2	accuracy	Training accuracy-95% Validation accuracy-72%	
3	Confidence score	Class detected-80% Confidence score-80%	

User Acceptance Testing:






Node-RED
Deploy
Flow 1
dashboard
Edit gauge node
Delete
Cancel
Done
Properties
Group
[CROP]MONITORING
Size
auto
Type
Gauge
Label
TEMPERATURE
Value format
{{value}}
Units
C
Range
min 0
max 100
Colour gradient
Sectors
0
optional
optional
100
Class
Optional CSS class name(s) for widget
Name
Enabled
debug
{ temperature: 28, humidity: 26, soil moisture: 75 }
{ temperature: 3, humidity: 82, soil moisture: 53 }
{ temperature: 46, humidity: 95, soil moisture: 82 }
{ temperature: 33, humidity: 60, soil moisture: 98 }
{ temperature: 43, humidity: 2, soil moisture: 86 }


node-red
4 Nov 18:48:05 - [info] Node-RED version: v3.0.2
4 Nov 18:48:05 - [info] Node.js version: v18.12.0
4 Nov 18:48:05 - [info] Windows_NT 10.0.19044 x64 LE
4 Nov 18:48:26 - [info] Loading palette nodes
4 Nov 18:48:44 - [info] Settings file : C:\Users\ELCOT\.node-red\settings.js
4 Nov 18:48:45 - [info] Context store : 'default' [module=memory]
4 Nov 18:48:45 - [info] User directory : \Users\ELCOT\.node-red
4 Nov 18:48:45 - [warn] Projects disabled : editorTheme.projects.enabled=false
4 Nov 18:48:45 - [info] Flows file : \Users\ELCOT\.node-red\flows.json
4 Nov 18:48:45 - [info] Creating new flow file
4 Nov 18:48:45 - [warn]
Your flow credentials file is encrypted using a system-generated key.
If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.
You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
4 Nov 18:48:45 - [warn] Encrypted credentials not found
4 Nov 18:48:45 - [info] Starting flows
4 Nov 18:48:46 - [info] Started flows
4 Nov 18:48:46 - [info] Server now running at http://127.0.0.1:1880/

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

1. Improved data collection driving farming efficiency

IoT-enabled agricultural solutions allow farmers to monitor the conditions and their products in real-time carefully and collect crucial data about the weather, land conditions, crop conditions, livestock, etc. The quick collection of data allows farmers to get insights fast and predict issues even before they happen. So, farmers can make informed decisions and find ways to avoid the problems.

Farming efficiency rises with the introduction of IoT solutions that help automate the farming process. For example, demand-based fertilizing, irrigation, and robot harvesting.

2. Resource optimization

IoT in agriculture focuses on optimizing the use of land, energy, and water. Thanks to precision farming relying on IoT, it is possible to quickly collect real-time data for varied sensors in the field. Farmers use the data to make accurate decisions and accurately allocate enough resources for farming efficiency.

3. End-to-end production control

The IoT-enabled agriculture solutions enhance the agility of the farming processes. The prediction and real-time monitoring systems make farmers control the entire crop production process without hassle. Farmers can quickly respond to changes in weather conditions, air quality, and humidity. Therefore, the production of crops is under the farmers' control allowing damage prevention.

4. Reduced wastage and cost management

IoT solutions in farming help to mitigate the risks of agriculture. By quickly detecting anomalies and inconsistencies in crop production, farmers can reduce waste and control costs while increasing production.

5. Cleaner process reducing the carbon footprint

Smart farming using IoT is genuinely the solution to decrease the use of fertilizers and pesticides. Precision farming helps farmers save energy and water and make farming a greener activity. The approach ensures a more organic and cleaner final product compared to conventional agricultural methods. And, it reduces the overall carbon footprint.

6. Process automation

For the longest time, farming and manual labor were synonymous. Thanks to IoT and smart farming, the dependency on manual labor has reduced significantly. The processes like pest control, fertilizing, and irrigation are increasingly becoming automated, and farmers can control them remotely. The use of smart IoT sensors can maintain these processes, increasing crop production.

7. Accentuated product quality

By utilizing IoT solutions, smart farming is able to meet the growing demand for crops while providing the highest quality standards. With more and more solutions being designed and introduced, nations can decrease the necessity to import and increase the chances of exporting their products. As decisions are taken beforehand, the product quality remains pristine.

DISADVANTAGES

- The smart agriculture needs availability of internet continuously. Rural part of most of the developing countries do not fulfil this requirement. Moreover internet connection isslower.

  The smart farming based equipments require farmers to understand and learn the use of technology. This is major challange in adopting smart agriculture farming at large scale across the countries.

- **Challenges in Using Smart Technologies in Farming -** The use of technology in agriculture and agriculture makes smart agriculture a successful and much-needed initiative, of course with the increasing food supply demand. There is opportunity for intelligent farming, especially to understand and operate the equipment, to require certain skills. With technologies such as robots and computer-based intelligence for the operation of machines, it is highly unlikely that a typical farmer will acquire or even grow such skills. Such high-end innovations are not used by farmers. Computer language or artificial intelligence are not known.
- **The Smart Agriculture Cost -** While the use of intelligent technology is impressive in agriculture, it entails many costs. As mentioned earlier, it takes a great deal of money to convert these equipment if the machines are altered according to the farmer 's point. At the other side, it will cost a lot of money to do the operation. As there are no higher profit levels for the agricultural industry, enormous investments are unlikely in this field. Even after altering the machinery, farmers may continue to operate the machinery improperly, causing harm or reparation. Because these devices are expensive now, it would cost a great deal of money to reparate or upgrade it.
- **Possibility for wrong Analysis of Weather Conditions -** The bulk of the cycle is weather-specific in the case of agriculture. It is a natural phenomenon that can become unpredictable given the modified technologies. Climate conditions such as rain, sunshine, drought etc. are not influenced or regulated by any power. The value of natural occurrences can not be reversed while the smart systems are in

place. In intelligent agriculture, there is a issue in which computers may have a detrimental effect on the environment. Because technology requires several computers, often the data will go wrong.

11.CONCLUTION

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 will 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. Specially IOT based system who can monitor the farm online. This will also help them in achieving better crop yields thus leading to their economic wellbeing.

12. FUTURE SCOPE

- IoT helps us meet our food needs by reducing environmental hazards, such as extreme weather and climatic transitions.
- The harvesters and tractors were both mechanical inventions that work in agriculture since the 20th century. The agriculture industry is heavily dependent on innovative ideas because of the increasing demand for food.
- The Industrial IoT has aided increased agricultural productivity with a lower cost, so, over the next few years, smart systems based on IoT will be more common in agricultural operations.
- A recent estimate shows that the agricultural industry will experience a compound annual growth rate (CAGR) of 20% due to IoT system installations.
- In addition, the number of linked agricultural devices will increase from 13 million in 2014 to 225 million by 2024.
- The future of IoT in agriculture allows predictive analytics to help you make better harvesting decisions. Pattern forecasting can be used by farmers to predict weather patterns and crop harvesting. IoT has helped farmers maintain the quality of their crop production.

APPENDIX

SOURCE CODE

```
import time importsys import ibmiotf.application  # toinstallpip
install ibmiotf importibmiotf.device


# Provide your IBM Watson Device Credentials organization = "8gyz7t"  #
replace the ORG ID deviceType = "weather_monitor"  #replace the Device
type deviceId = "b827ebd607b5" # replace Device ID authMethod = "token"
authToken = "LWVpQPaVQ166HWN48f"  # 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 = "8gyz7t"  #
    replace the ORG ID deviceType = "weather_monitor"  #replace the Device
    type deviceId = "b827ebd607b5" # replace Device ID authMethod = "token"
    authToken = "LWVpQPaVQ166HWN48f"  # 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":"b827ebd607b5",
"deviceType":"weather_monitor",
"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":"",
"classNam e":"",
"icon":"",
"payload":"{\"command\":\"motoron\"}",
"payloadType":"str",
"topic":"motoron",
"topicType":"s
tr","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":"MotorOF
F",
"tooltip":"",
"color":"",
"bgcolor":"",
"classNam e":"",
"icon":"",
"payload":"{\"command\":\"motoroff\"}",
"payloadType":"str",
"topic":"motoroff",
"topicType":"s
tr","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":fal
se},
{"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":"ef745d48e395ccc0",

"inputType":"evt",
"logicalInterface":"",
"ruleId":"",
"deviceId":"b827ebd607b5",
"applicationId":"",
"deviceType":"weather_monitor",

```
"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
":
48
0,
"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":86
0,
"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":"",
"li
bs
":[
],
```

"x
":
49
0,
"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":"",
"classNam
e":"",
"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":"",
"classNam
e
":"",
"x":790,
"y":120,
"wires":[]
},
{
"id":"a259673baf5f0f98
","type":"httpin",
"z":"03acb6ae05a0c712
","name":"",
"url":"/sensor",
"method":"ge
t",
"upload":fals
e,
"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')};\nreturn
msg;",

```
"outputs":1,
"noerr":0,

"initialize":"",
"finalize":"",
"li
bs
":[
],
"x
":
63
0,
"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":f
alse,
"className
":""
},
{
"id":"1f4cb829.2fdee8",
"type":"ui_tab",
"name":"Home",
"icon":"dashboard
","order":3,
"disabled":false,
"hidden":false }