IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

Team ID : PNT2022TMID41353

Team Members

M.Nilofar Nisha
Z.Jafrin
R.Priya
N.Sneka

INTRODUCTION

PROJECT OVREVIEW:

Crops in farms are many times ravagedby local animalslike buffaloes, cows, goats, birds etc. this leads to huge losses for the farmers. It is not possible for farmers tobarricade entire fields or stay on field 24 hours and guard it.so here we proposeautomatic crop protection system from animals. This is a microcontroller based system using PIC family microcontroller. The microcontroller now sound an alarm to woo the animal away from the field as well as sends SMS to the farmer so that he may about the issue and come to the spot in case the animal don't turn away by the alarm. This ensures complete safetyof crop from animals thus protecting farmers loss.

PURPOSE:

Our main purpose of the project is to develop intruder alert to the farm, to avoid losses due to animal and fire. These intruder alert protect the crop that damaging that indirectly increase yield of the crop. The develop system will not harmful and injurious to animal as well as human beings. Theme of project is to design a intelligent security system for farm protecting by using embedded system.

LITERATURE SURVEY

This paper describes overview of various researches on smart crop protection system. We have a lot of technology that can protect the farm 24x7 those systems and technique we are discussing in this paper. We have different types of technology that can help to secure the farm. We have seen Arduino and raspberry pi based Farm protection system. But those Systems have different mythology and platform for that and the cost of those projects also increased so that those are not affordable with the farmer. Our main aim to design a system that can help to farmer to protect his farm from, animals with getting harm to them.

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 ,pg.no 261-265,Sept 2020.

2 .Mr.P.Venkateswara Rao, Mr.Ch Shiva Krishna ,MR M Samba Siva ReddyLBRCE,LBRCE,LBRCE.

3. Mohit Korche,Sarthak Tokse, ShubhamShirbhate, Vaibhav Thakre,S. P. Jolhe(HOD). Students , Final Year,Dept.of Electrical engineering,Government College of engineering,Nagpur head of dept.,Electrical engineering,Government College of engineering,Nagpur.

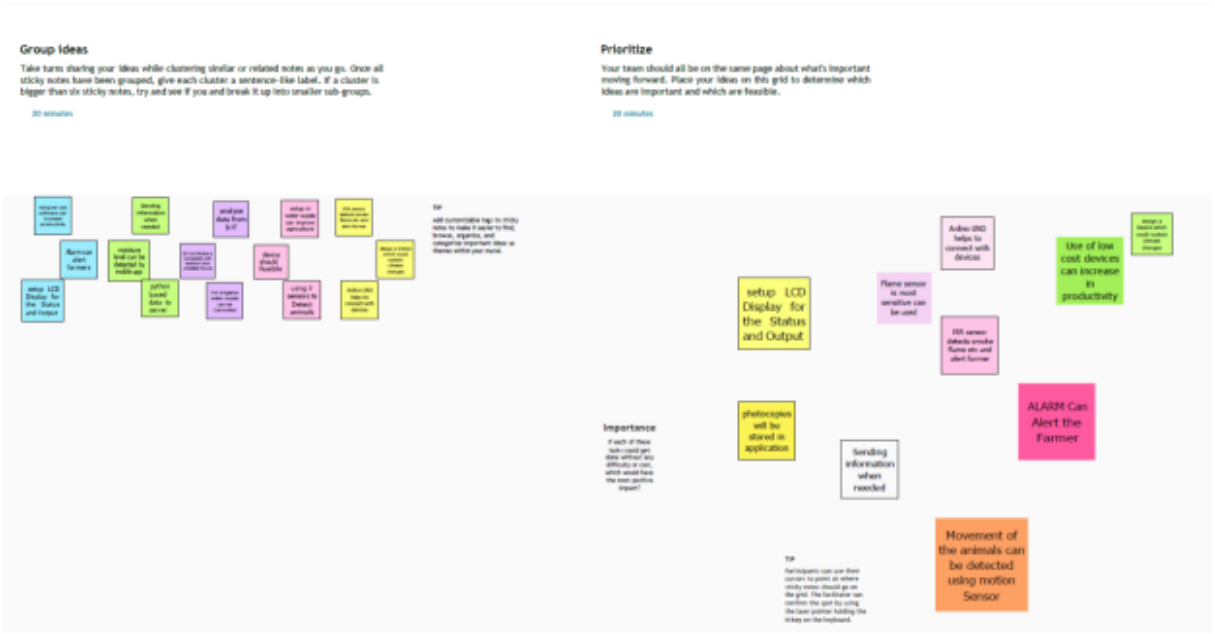
PROBLEM STATEMENT DEFINITION STATEMENT:

In the world economyof many Countrydependent upon the agriculture. In spite of economic development agriculture is the backbone of the economy. Crops in forms are many times ravaged by local animals like buffaloes, cows, goats, birds and fire etc. this leads to huge loss for the farmers.it is not possible for farmers to blockade to entire fields or stay 24 hours and guard it. Agriculture meets food requirements of the people and produces several raw materialsfor industries. But because of animal interference and fire in agricultural lands, there will be huge loss of crops.Crops will be totally getting destroyed.

IDEATION AND PROPOSED SOLUTION

EMPATHY MAP CANVAS:

IDEATION AND BRAINSTORMING:

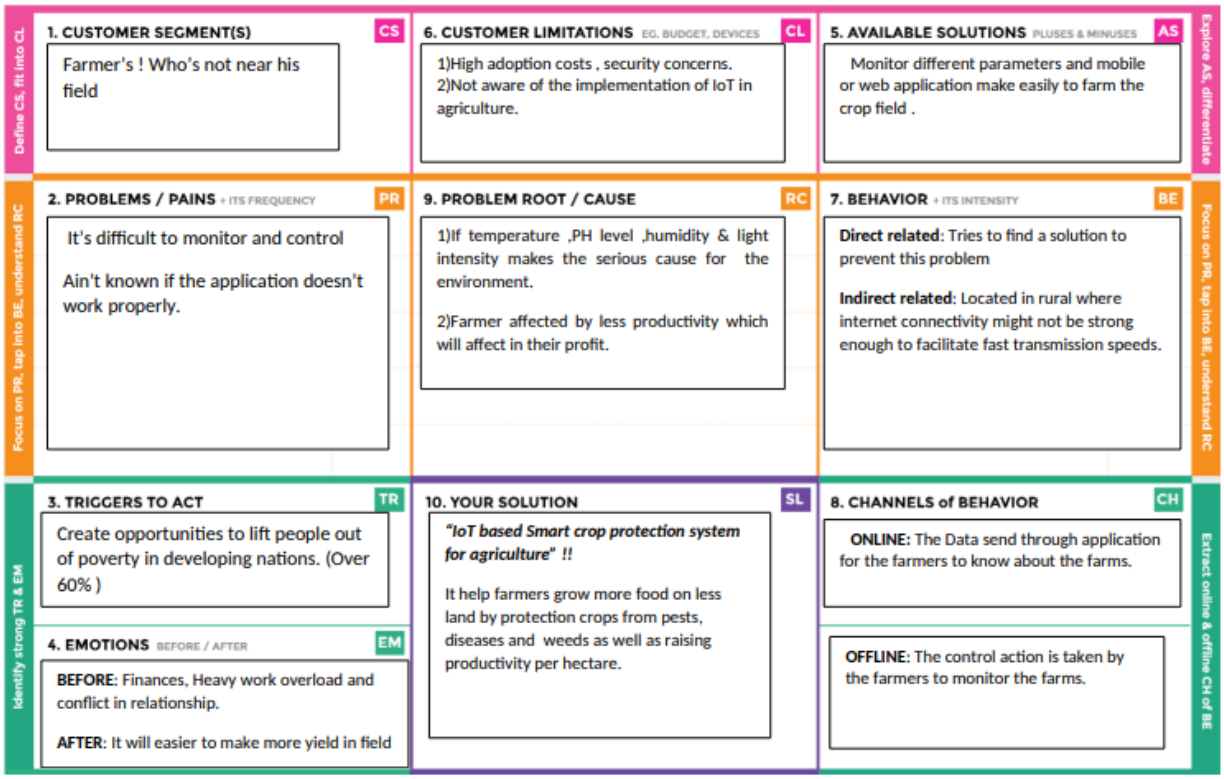


PROPOSED SOLUTION:

S.NO	PARAMETERS	DESCRIPTION
1.	Problem Statement (Problem to be solveed)	Usually cross in the fields are protected against birds and other unknown disturbances by humans. This take an enormous amount of time.Creating a smart automatic system will beneft the farmers in many different ways
2.	Idea / Solution descripton	Smart Farming has enableed farmers to reduce waste aned enhance proeductvity with the help of sensors (light, humidity, temperature, soil moisture,etc..).
3.	Novelty / Uniqueness	Role of SENSORS : IOT smart agriculturep roeducts are edesigneed to hel monitor cro feleds using sensors aned by automatng irrigaton systems. As a result, farmers aned associateed braneds can easily monitor the feled coneditons from anywhere without any hassle.
4.	Social Im act / Customer Satsfacton	Water conservaton . Saves lot of time . Increaseed quality of producton. Real time data and proeducton insight. Remote monitoring.
5.	Scalability of the Soluton	Scalability in smart farming refers to the and ability of a

		system to increase the ca pacity , the numbers.

PROBLEM SOLUTION FIT:



REQUIREMENT ANALYSIS

FUNCTIONAL REQUIREMENT:

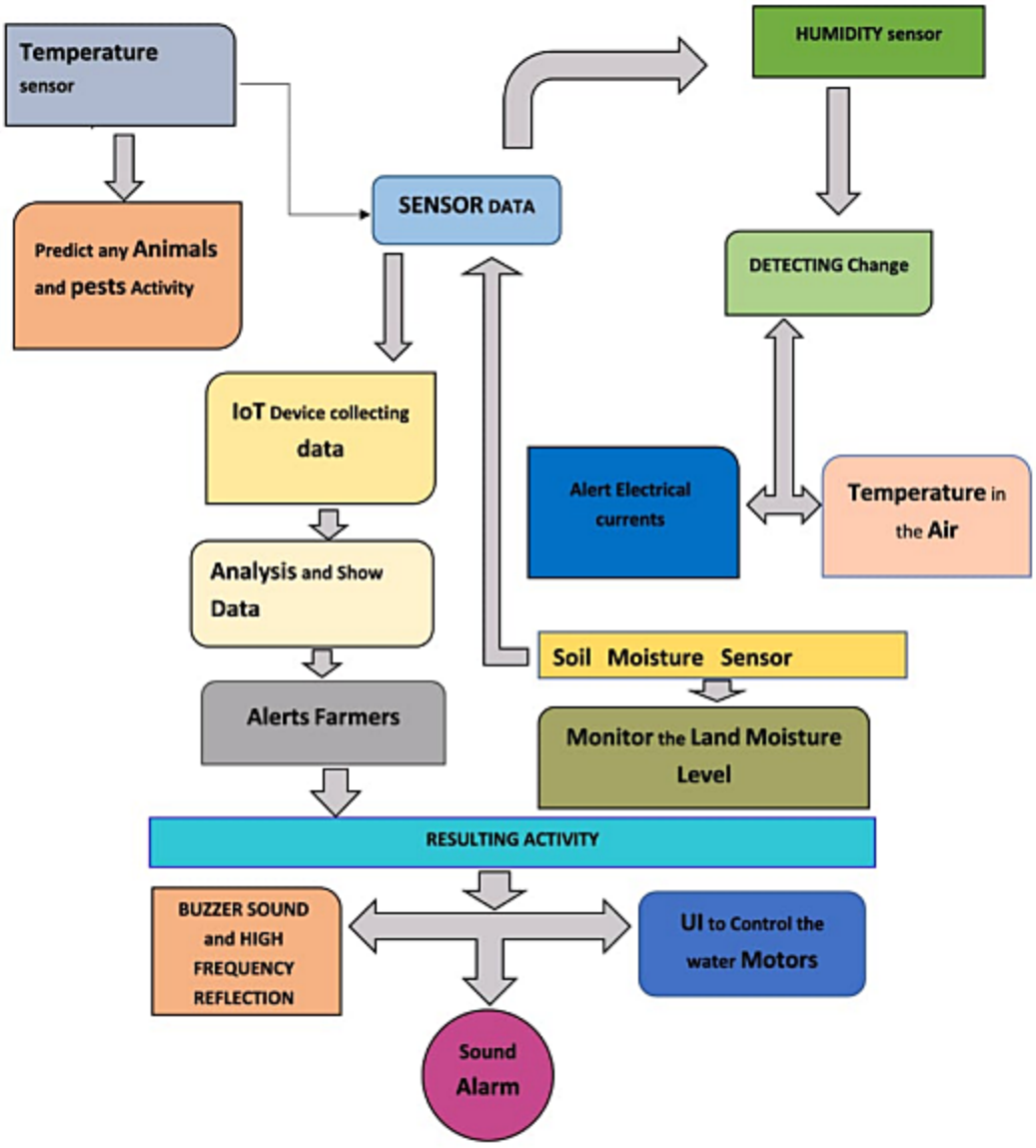
FR NO	Functional Requireeent (Epic)	Sub Requireeent (Story / Sub-Task)
1.	User Registration	Install the app Signing up with Gmail or phone number Creating a profile Understand the guidelines
2.	User Confirmation	Email or phone number verification required via OTP
3.	Accessing datasets	Data's are obtained by cloudant DB.
4.	Interface sensor	Connect the seosor aod the applicatoo When animals enter the field the alarm is generated

NON FUNCTINAL REQUIREMENT:

FR No.	Non-Functional Requirement	Description
1.	Usability	This project's conntributes the farm protection through the smart protection system
2.	Security	It was created to protect the crops from animals
3.	Reliability	Farmers are able to safeguard their loads by help of this technology They will also beneits from higher crop yields which will improve our economic situation
4.	Performance	When animals attempt to enter the field IOT devices and seosors alert the farmer via message
5.	Availability	We can defend the crops agaiost wild animals by creating and implementing resilinot hardware aod software.
6.	Scalability	This system's integration of computer vision algorithms with IBM cloudant services makes it more efficieot to retrieve photos at scale enhanching and scalability.

PROJECT DESIGN

DATA FLOW DIAGRAM:



SOLUTION AND TECHNICAL ARCHITECTURE:

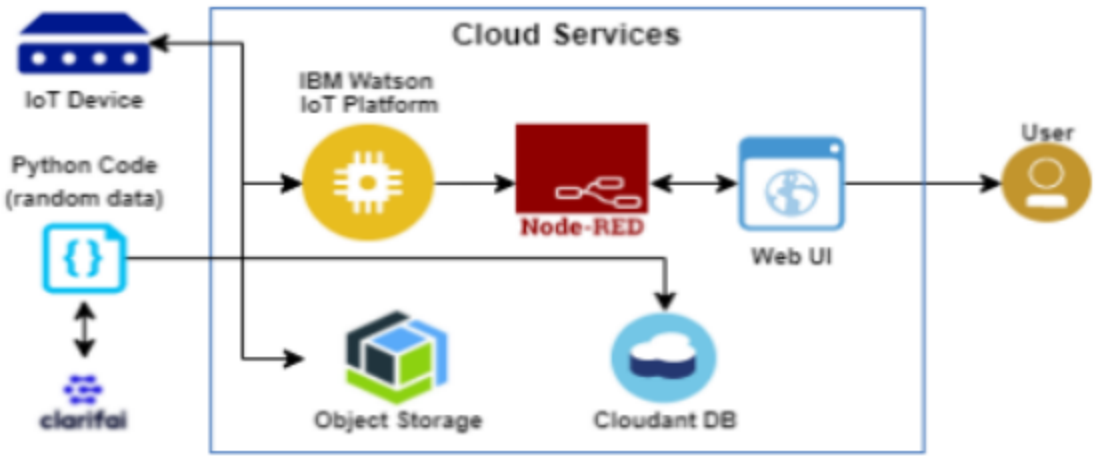


Table-1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with the Web UI	App development
2.	Application Logic-1	Logic for a process in the application	Python Objectives
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	Node-RED service
5.	Database	Data Type	Database Cloudant DB
6.	Cloud Database	Database Service on Cloud	Cloud Object store service
7.	File Storage	File storage requirements	IBM Block Storage
8.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration:	Cloud Foundry

		Cloud Server Configuration:	

TABLE 2:

S.No	Characteristics	Description	Technology
1.	Open-source Frameworks	The open- source frameworks used	SAN-SAF
2.	Security Implementations	List all the security / access controls implemented	IBM cloud encryptions
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	IBM cloud Architecture
4.	Availability	Justify the availability of applications (e.g. use of load balancers, distributed servers etc.)	Web Application can even be used by the framers in the horticulture
5.	Performance	Design consideration for the performance of the application	Since the web application is high efficient, it can be used by the farmers irrespective of time

USER STORIES:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1		US-1	Create the IBM Cloud services which are being used in this project.	6	High	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
Sprint-1		US-2	Configure the IBM Cloud services which are being used in completing this project.	4	Medium	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
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	Nilofar Nisha M, Jafrin Z, Priya R,

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
						Sneha N
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	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
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	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
Sprint-3		US-2	Create a Node-RED service.	10	High	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
Sprint-3		US-1	Develop a python script to publish random sensor data such as temperature, moisture, soil and humidity to the IBM IoT platform	7	High	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
Sprint-3		US-2	After developing python code, commands are received just print the statements which represent the control of the devices.	5	Medium	Nilofar Nisha M, Jafrin Z, Priya R,

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
						Sneha N
Sprint-4		US-3	Publish Data to The IBM Cloud	8	High	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
Sprint-4		US-1	Create Web UI in Node- Red	10	High	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N
Sprint-4		US-2	Configure the Node-RED flow to receive data from the IBM IoT platform and also use Cloudant DB nodes to store the received sensor data in the cloudant DB	10	High	Nilofar Nisha M, Jafrin Z, Priya R, Sneha N

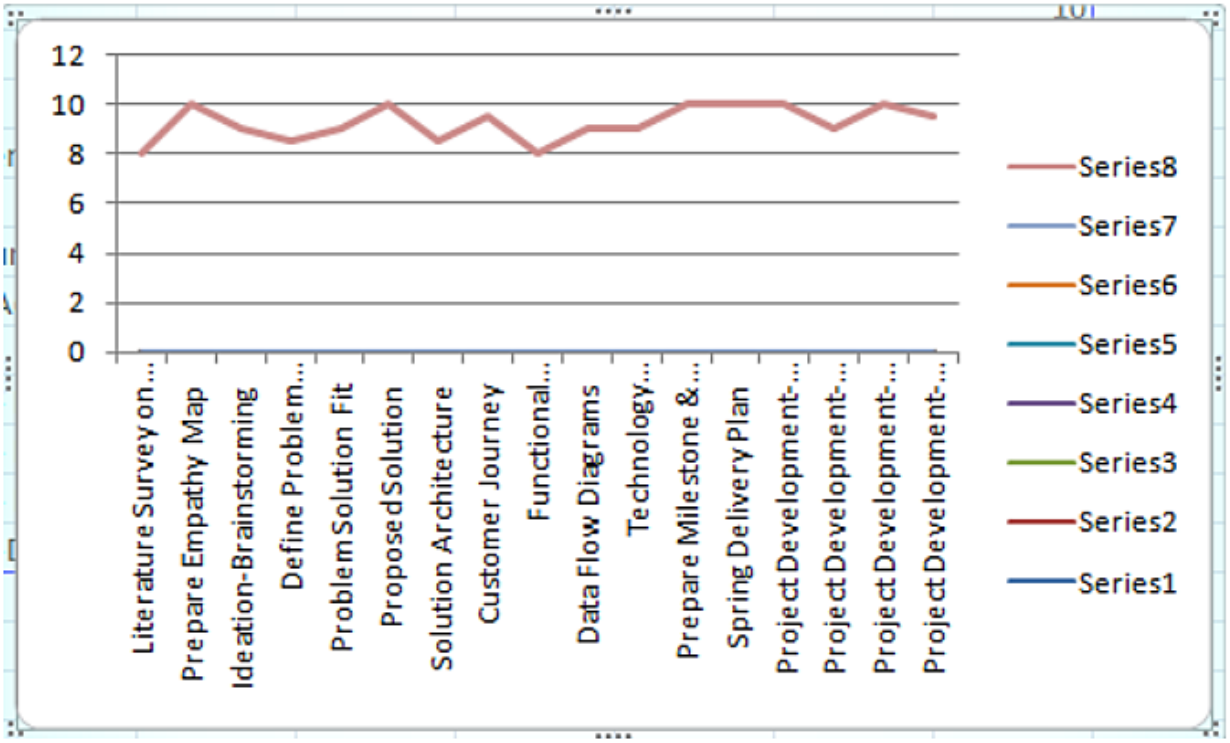
PROJECT PLANNING AND SCHEDULING

SPRINT PLANNING AND ESTIMATION:

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	05 Nov 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

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)

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

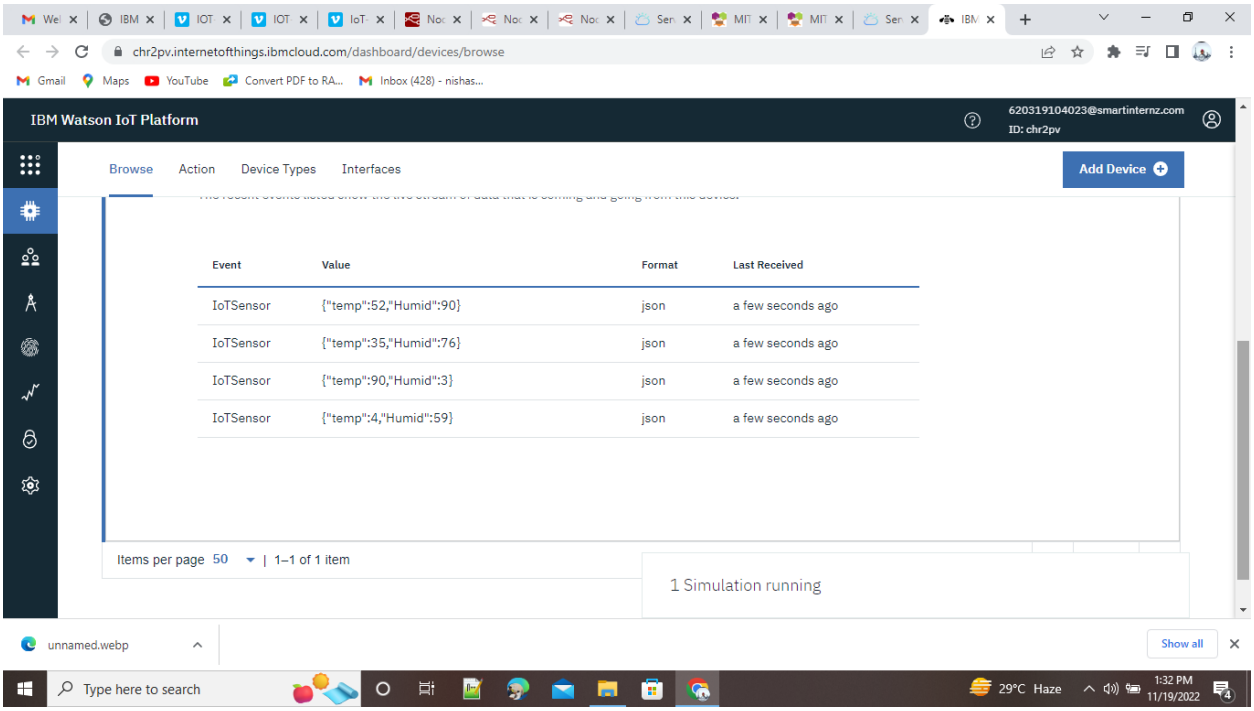


CODING AND SOLUTIONING

FEATURE-1

```
import time
 import sys
 import ibmiotf.application
 import ibmiotf.device
 import random
 #Provide your IBM Watson Device Credentials
 organization = "chr2pv"
" deviceType = "NodeMCU"
 deviceId = "12345"
authMethod = "token"
 authToken = "12345678"
 # Initialize GPIO
 try:
deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method":
authMethod, "auth-token": authToken}
 deviceCli = ibmiotf.device.Client(deviceOptions)
 #..... except
Exception as e:
 print("Caught exception connecting device: %s" % str(e)) sys.exit()
 # Connect and send a datapoint "hello" with value "world" into the cloud as an event of
type"greeting" 10 times
 deviceCli.connect()
 while True:
#Get Sensor Data from DHT11
temp=random.randint(0,100)
Hum=random.randint(0,100)
 moisture=random.randint(0,100)
 data = { 'temperature' : temp, 'Humidity': Hum, 'Moisture':moisture }
 #print data def myOnPublishCallback():
print ("Temperature = " + str(temp)+" C Humidity = " + str(hum)+ " moisture = " + str(moisture) +
"to IBM Watson")
 success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
 if not success:
 print("Not connected to IoTF") time.sleep(10)
 deviceCli.commandCallback = myCommandCallback
 # Disconnect the device and application from the cloud
```

deviceCli.disconnect()



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:

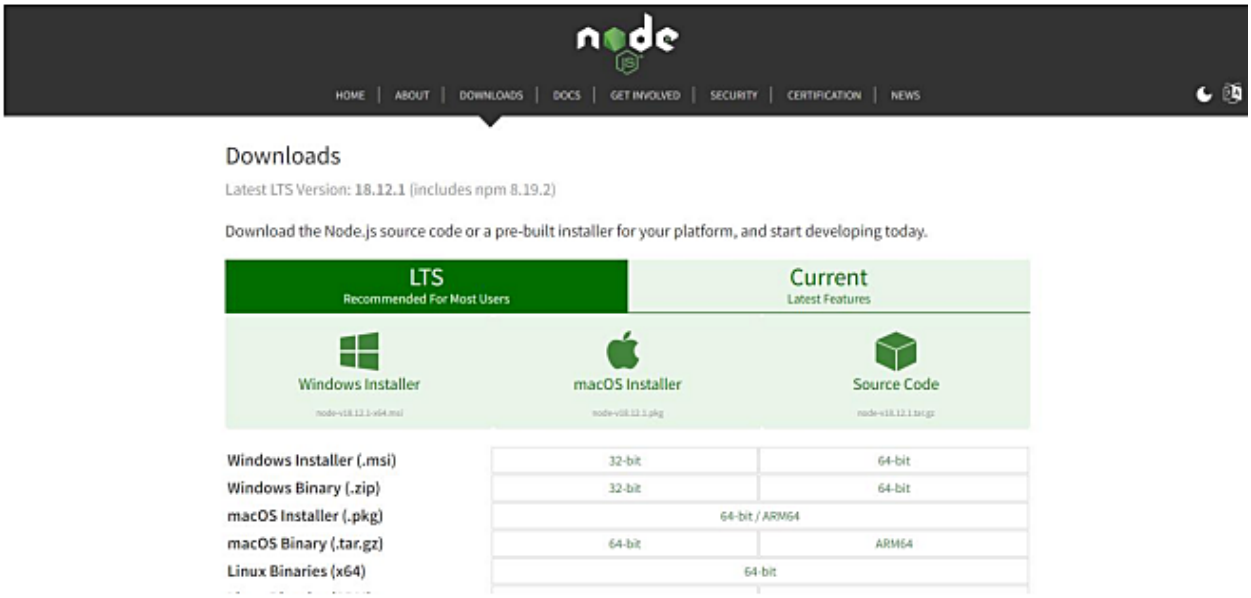
i. Goodsensitivity to Combustible gas in wide range .
ii. Highsensitivity to LPG, Propane and Hydrogen .
iii. Longlife and low cost.
iv. Simpledrive circuit.

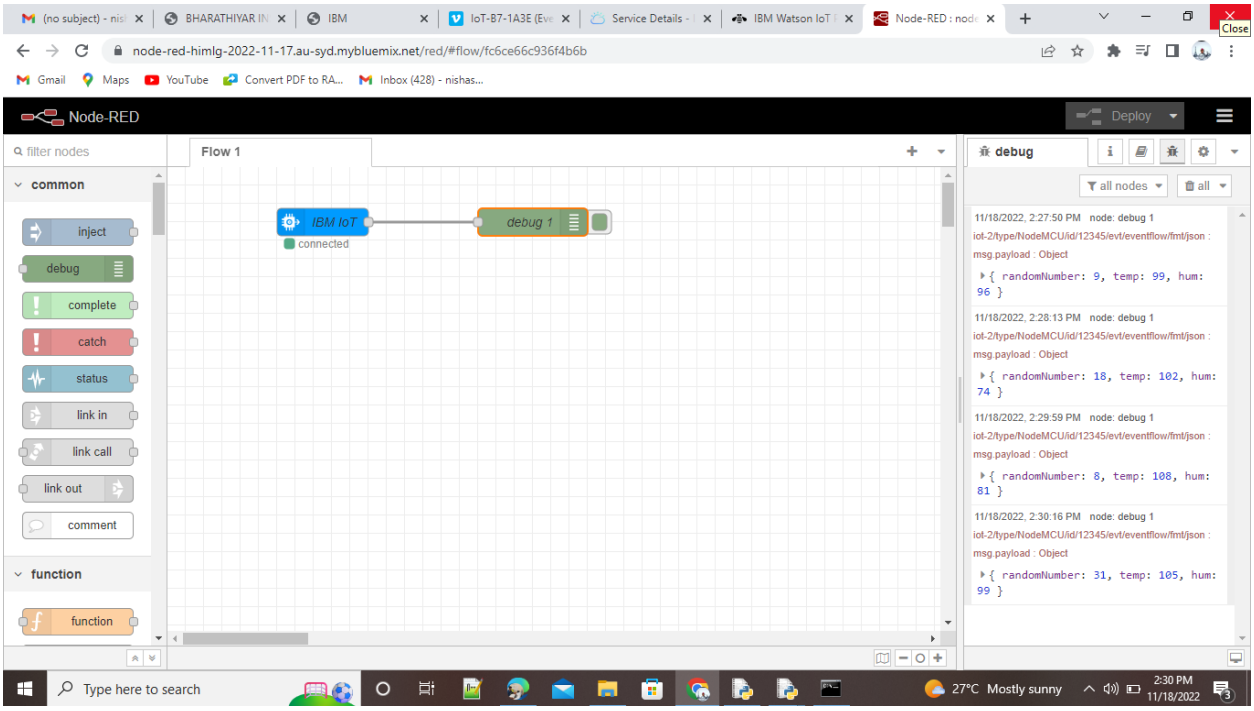
TESTING

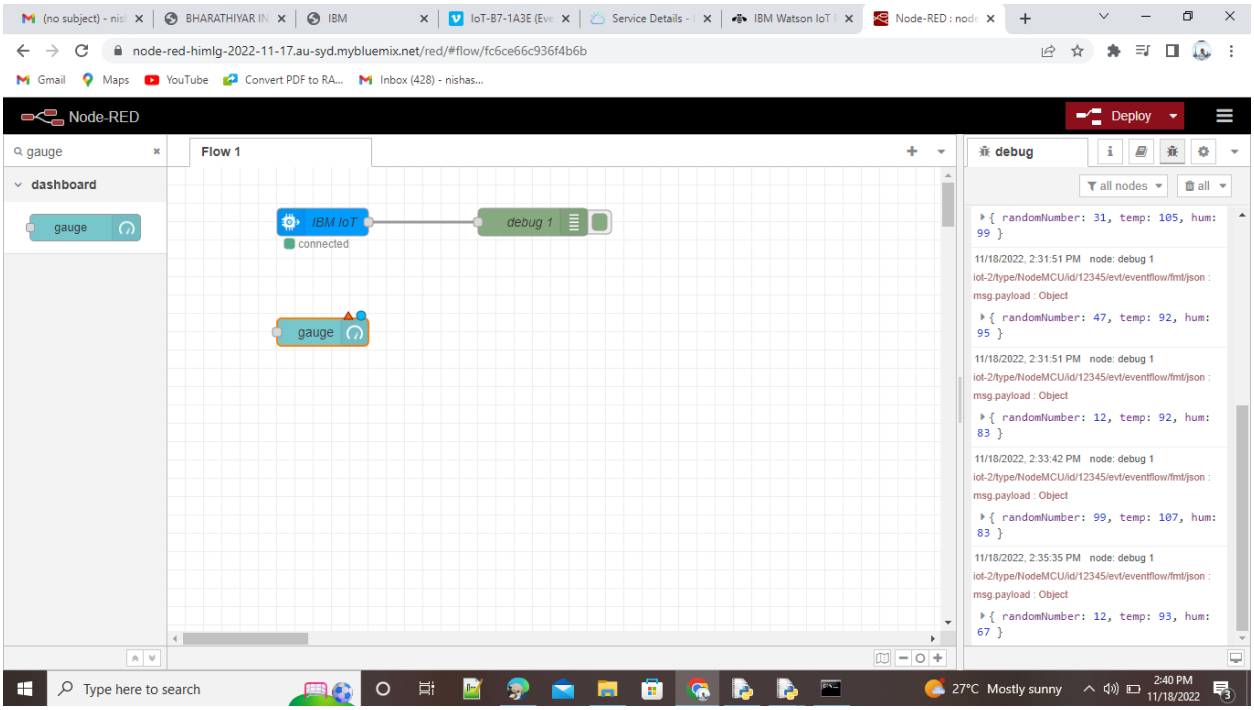
TEST CASES:

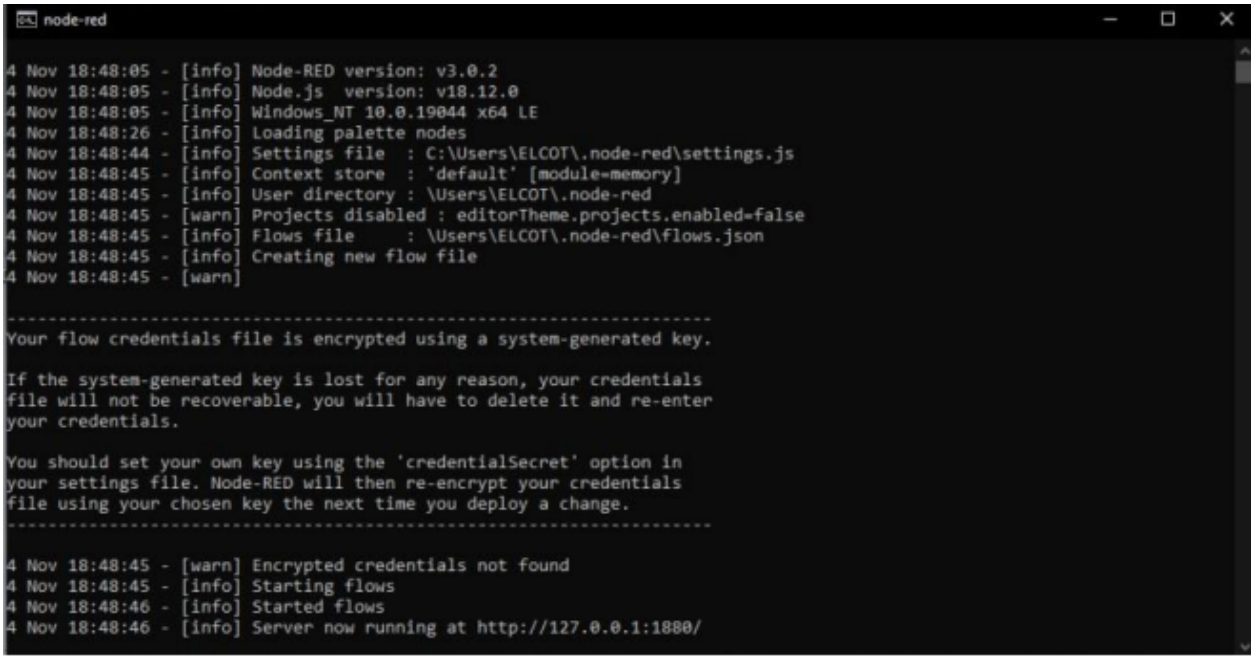
S.NO	Parameter	Values	Screenshot
1.	Model summary	-	
2.	accuracy	Training accuracy95% Validation accuracy72%	
3.	Confidence score	Class detected80% Confidence score-80%	

User Acceptance Testing:






Node-RED
Deploy
gauge
dashboard
gauge
Flow 1
IBM IoT
connected
debug 1
gauge
debug
all nodes
all
{ randomNumber: 31, temp: 105, hum: 99 }
11/18/2022, 2:31:51 PM node: debug 1
iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json : msg.payload : Object
{ randomNumber: 47, temp: 92, hum: 95 }
11/18/2022, 2:31:51 PM node: debug 1
iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json : msg.payload : Object
{ randomNumber: 12, temp: 92, hum: 83 }
11/18/2022, 2:33:42 PM node: debug 1
iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json : msg.payload : Object
{ randomNumber: 99, temp: 107, hum: 83 }
11/18/2022, 2:35:35 PM node: debug 1
iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json : msg.payload : Object
{ randomNumber: 12, temp: 93, hum: 67 }
Type here to search
27°C Mostly sunny
2:40 PM
11/18/2022


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/

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.

ADVANTAGES AND DISADVANTAGES

Advantage:

Controllable food supply. you might have droughts or floods, but if you are growing the crops and breeding them to be hardier, you have a better chanceof not straving.

It allows farmers to maximize yields using minimum resources such as water,fertilizers.

Disadvantage:

The main disadvantage is the time it can take to process the information.in order to keep feeding people as the population grows you have to radically change theenvironment of the planet.

CONCLUSION

A IoT Web Application is built for smart agricultural system using Watson IoT platform, Watsonsimulator, IBM cloud and Node-RED .

FUTURE 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.

APPENDIX

```
import time
 importsys
 import ibmiotf.application
 # toinstallpip install ibmiotf
 importibmiotf.device
 # Provide your IBM Watson Device Credentials
organization = "chr2pv"
 # replace the ORG ID deviceType = "NodeMCU" #replace the Device type deviceId = "12345"
# 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:
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)
# ..... except
Exception as e:
 print("Caught exception connecting device: %s" % str(e))sys.exit()
 # Connect and send a datapoint "hello" with value "world" into the cloud as an event 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 ="chr2pv"
 # replace the ORG ID deviceType = "NodeMCU
 #replace the Device type deviceId = "12345"
 # 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
) #..... except
Exception as e:
print("Caught exception connecting device: %s" % str(e))sys.exit()
 # Connect and send a datapoint "hello" with value "world" into the cloud as an event 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()
```

GitHub & Project Demo Link

https://github.com/IBM-EPBL/IBM-Project-42411-1660661914