|  |  |
| --- | --- |
| **TEAM ID** | PNT2022TMID26051 |
| **PROJECT NAME** | IoT Based Smart crop protection system for Agriculture |
| **TEAM MEMBERS** | G Poojitha Naidu  K pravalika  S Bhavana  G Shamitha Reddy |

**INTRODUCTION**

**PROJECT OVERVIEW:**

Many times, local animals like buffaloes, cows, goats, birds, etc. attack crops in farms, causing enormous losses for the farmers. Farmers are unable to block entire fields or remain on the field all day to guard it. Therefore, we suggest an automated crop security system against animals. This system is microcontroller-based and uses microcontrollers from the PIC family. The microcontroller now sounds an alarm to entice the animal out of the field and sends a text message to the farmer alerting him to the problem so he may respond by travelling to the scene if the animal does not flee after hearing the alarm. This completely protects crops from animals, preventing farmers' losses.

**PURPOSE:**

The primary goal of the project is to create a farm intruder alarm system to prevent losses from animal and fire damage. These intrusion alerts shield the crop from harm, which inadvertently boosts agricultural productivity. Both humans and animals won't be harmed or injured by the developed technology. The goal of the project is to develop an embedded security system with intelligence for defending Farms

**LITERATURE SURVEY**

**EXISTING PROBLEM:**

The functionality of the current technology is mostly focused on surveillance. Additionally, especially in such an application region, these systems do not offer protection from wild animals. They must also take action based on the kind of animal that tries to enter the area, as various strategies are used to keep certain species out of restricted regions. Another frequently employed technique by farmers to stop animals from damaging their crops is the construction of physical barriers, the use of electric fences, manual surveillance, and other such time-consuming and hazardous techniques.

REFERENCES:

<https://www.researchgate.net/publication/349940582_Implementation_of_IIoT_based_smart_crop_protection_and_irrigation_system>

<https://www.irjet.net/archives/V8/i2/IRJET-V8I2317.pdf>

<https://ijirt.org/master/publishedpaper/IJIRT151020_PAPER.pdf>

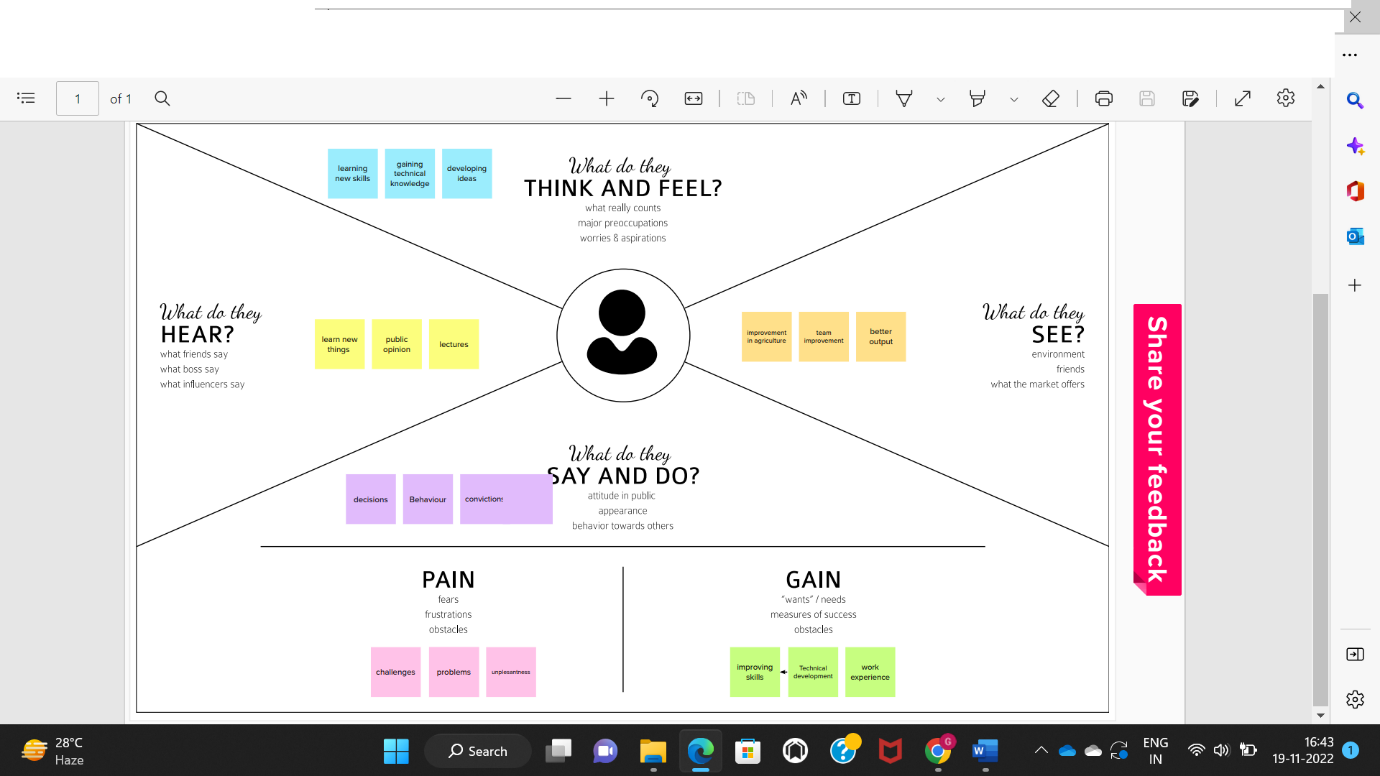
PROBLEM STATEMENT DEFINITION:

Many nations around the world rely on agriculture for their economies.

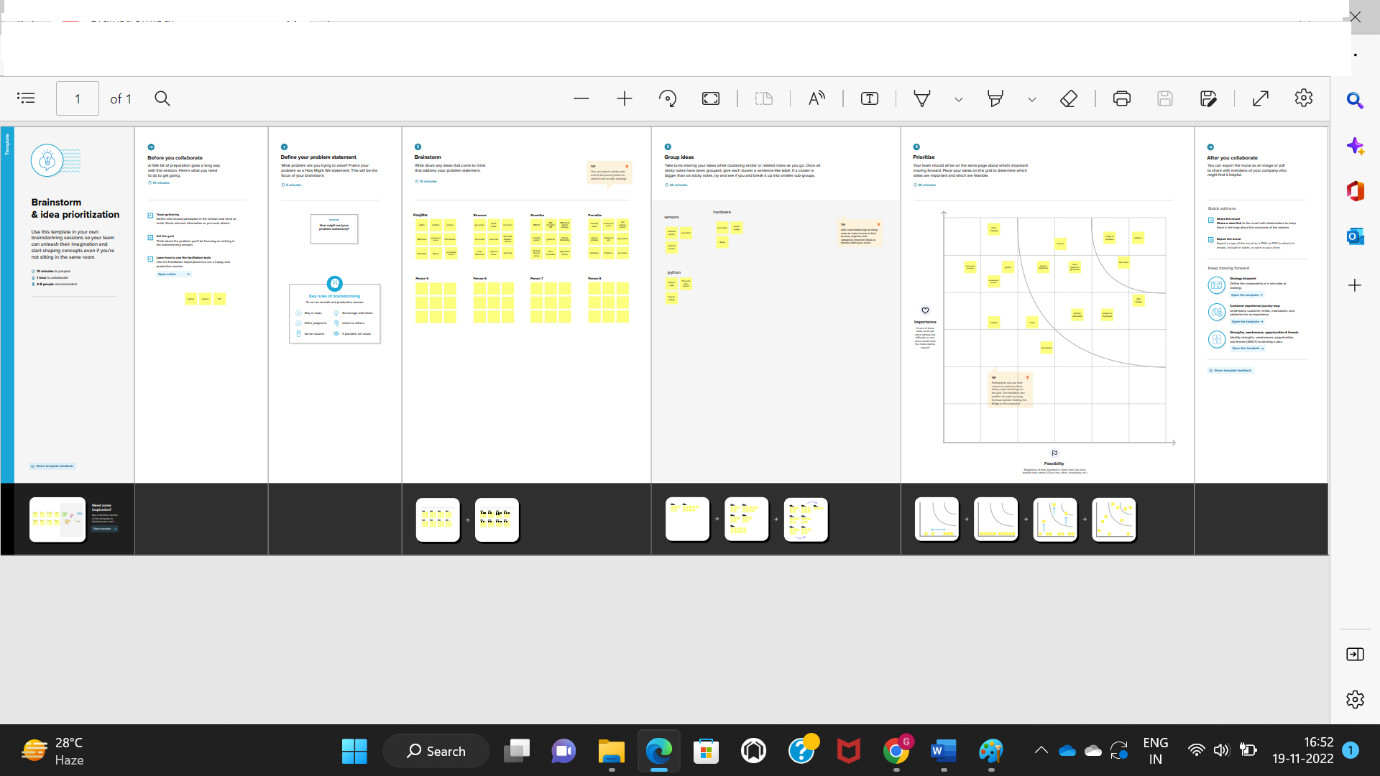
Agriculture remains the foundation of the economy despite its growth. Numerous indigenous creatures, including buffalo, cows, goats, birds, and fire, among others, often destroy crops in various forms. For the farmers, this causes a big loss. Farmers are unable to blockade entire fields or stay on site to guard them round-the-clock. In addition to providing people with the food they need, agriculture also generates a variety of industrial raw materials. There will be a substantial loss of crops nonetheless due to animal interference and fire in agricultural fields. The destruction of crops will be complete.

IDEATION AND PROPSED SOLUTION

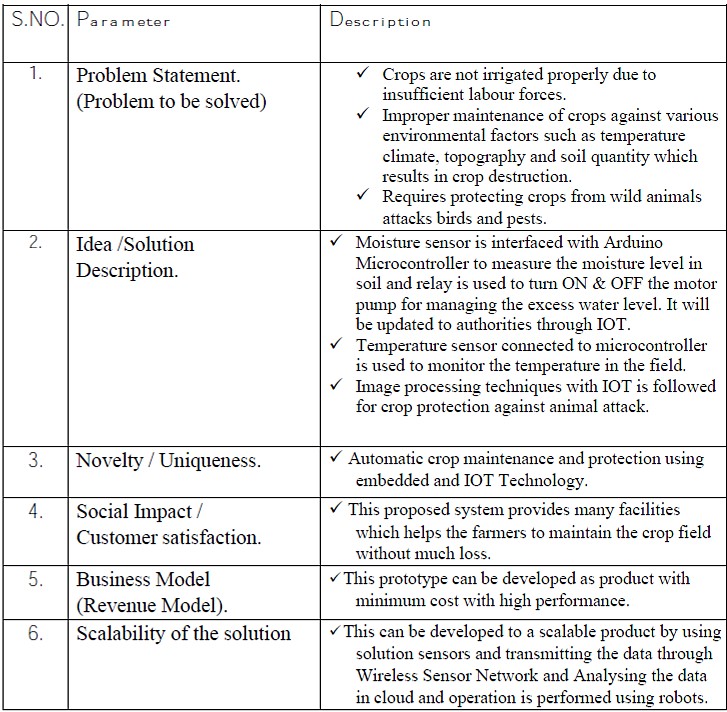
EMPATHY MAP CANVAS:



IDEATION & BRAINSTORMING:



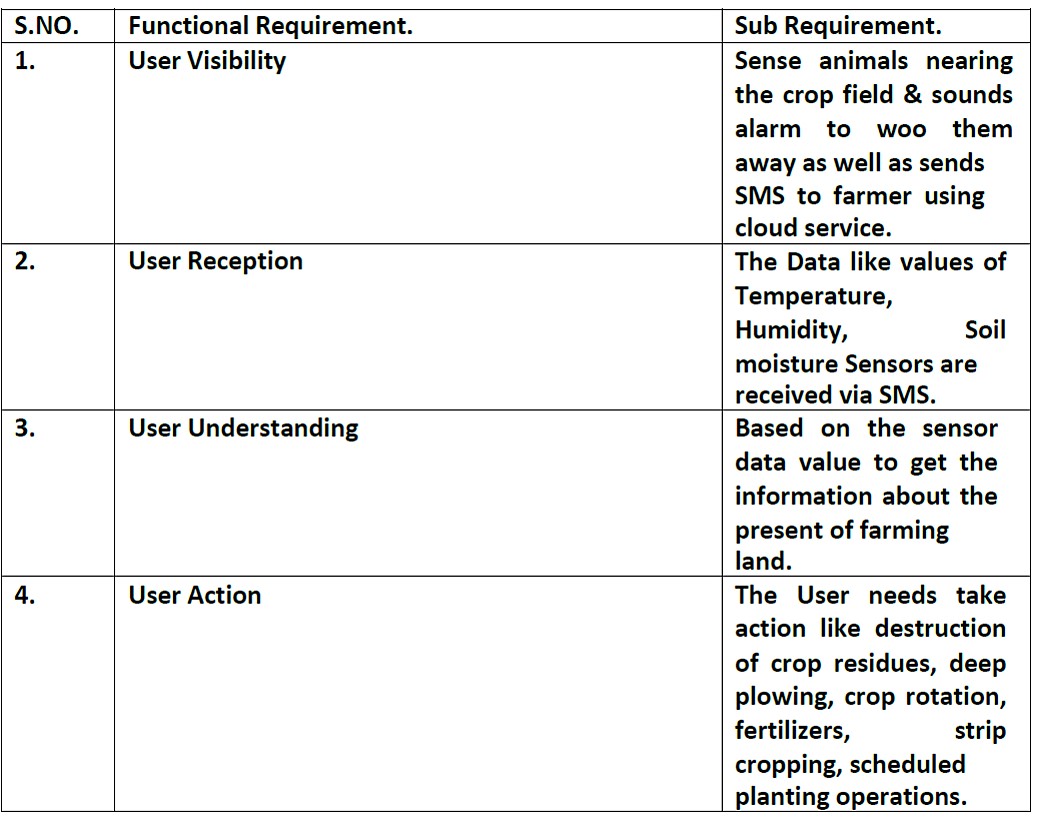
PROPSED SOLUTION:



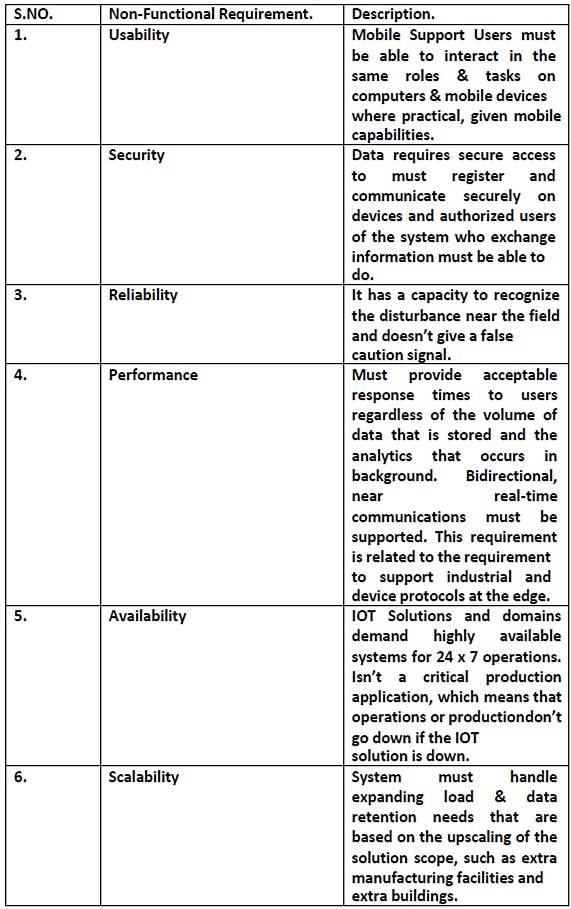
PROBLEM SOLUTIONFIT:



FUNCTIONAL REQUIREMENTS:

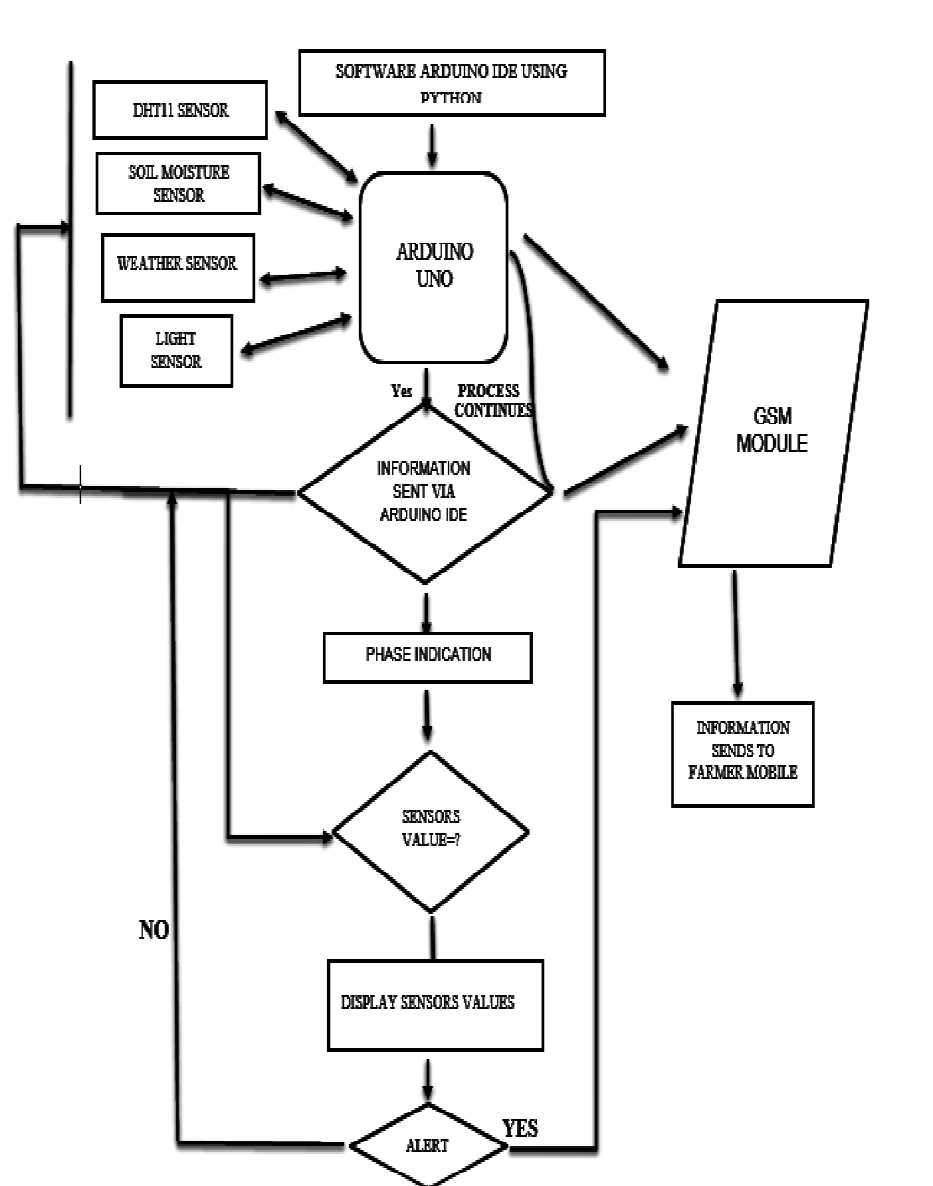


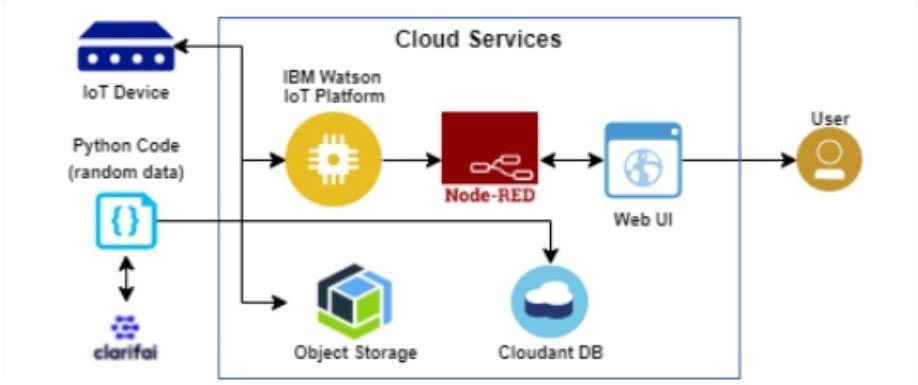
NON FUNCTIONAL REQURIEMENTS:



PROJECT DESIGN:

DATA FLOW DIAGRAM



SOLUTION AND TECHNICAL ARCHITECTURE:

|  |  |  |  |
| --- | --- | --- | --- |
| S.NO | Component | Description | Technology |
| 1 | User Interface | How user  interacts with application  e.g., Mobile application | HTML, CSS,  JavaScript /  Angular JS  /Node red |
| 2 | Application Logic-1 | Logic for a process in the application | Java / Python |
| 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 | IBM Watson  Assistant |
| 5 | Database | Data Type, Configuratio ns etc. | MySQL,  NoSQL, etc |
| 6 | Cloud Database | Database  Service on Cloud | IBM DB2 |
| 7 | File storage | File storage requirements | IBM Block  Storage or  Other  Storage  Service or  Local  Filesystem |
| 8 | External API-1 | Purpose of External API used in the application | IBM  Weather API, etc. |
| 9 | IOT Model | Purpose of IoT Model is for integrating the sensors with a user interface. | IBM IoT Platform |
| 10 | Infrastructure(server/cloud) | Application Deployment on Local System /  Cloud Local  Server Configuratio n: Cloud Server Configuratio n : | Local, Cloud  Foundry, Kubernetes,  etc |

USER STORIES:

|  |  |  |
| --- | --- | --- |
| **Sprint** | **User Story**  **Number** | **User Story / Task** |
| Sprint-1 | US-1 | Create the IBM Cloud services which are being used in this project. |
| Sprint-1 | US-2 | Configure the IBM Cloud services which are being used in completing this project. |
| 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. |
| 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. |
| 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. |
| Sprint-3 | US-2 | Create a Node-RED service. |
| 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 |
| Sprint-3 | US-2 | After developing python code, commands are received just print the statements which represent the control of the devices. |
| Sprint-4 | US-3 | Publish Data to The IBM Cloud |
| Sprint-4 | US-1 | Create Web UI in Node- Red |
| 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 |

**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 | 03 Nov 2022 | 09 Nov 2022 | 20 | 09 Nov 2022 |
| Sprint-2 | 20 | 6 Days | 04 Nov 2022 | 11 Nov 2022 | 20 | 11 Nov 2022 |
| Sprint-3 | 20 | 6 Days | 06 Nov 2022 | 13 Nov 2022 | 20 | 13 Nov 2022 |
| Sprint-4 | 20 | 6 Days | 09 Nov 2022 | 15 Nov 2022 | 20 | 15 Nov 2022 |

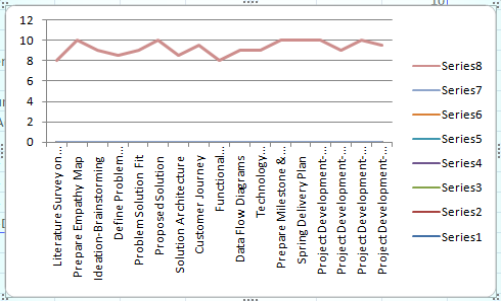
**Velocity:**

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)



**Burndown Chart:**

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile[software development](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as [Scrum](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/). However, burn down charts can be applied to any project containing measurable progress over time.



CODING AND SOLUTION

FEATURE1:

float temp;

const int buzzer = 8;

void setup()

{

pinMode(A0, INPUT);

pinMode(11, OUTPUT);

pinMode(buzzer, OUTPUT);

Serial.begin(9600);

pinMode(3, INPUT);

pinMode(13, OUTPUT);

}

void loop()

{

temp = analogRead(A0);

temp =((temp\*5)/1024);

temp = (temp-0.5)\*100;

Serial.print("Temperature = ");

Serial.println(temp);

if (temp > 40)

{

digitalWrite(11, HIGH);

tone(buzzer, 1000);

delay(1000);

}

if (digitalRead(3) == HIGH) {

digitalWrite(13, HIGH);

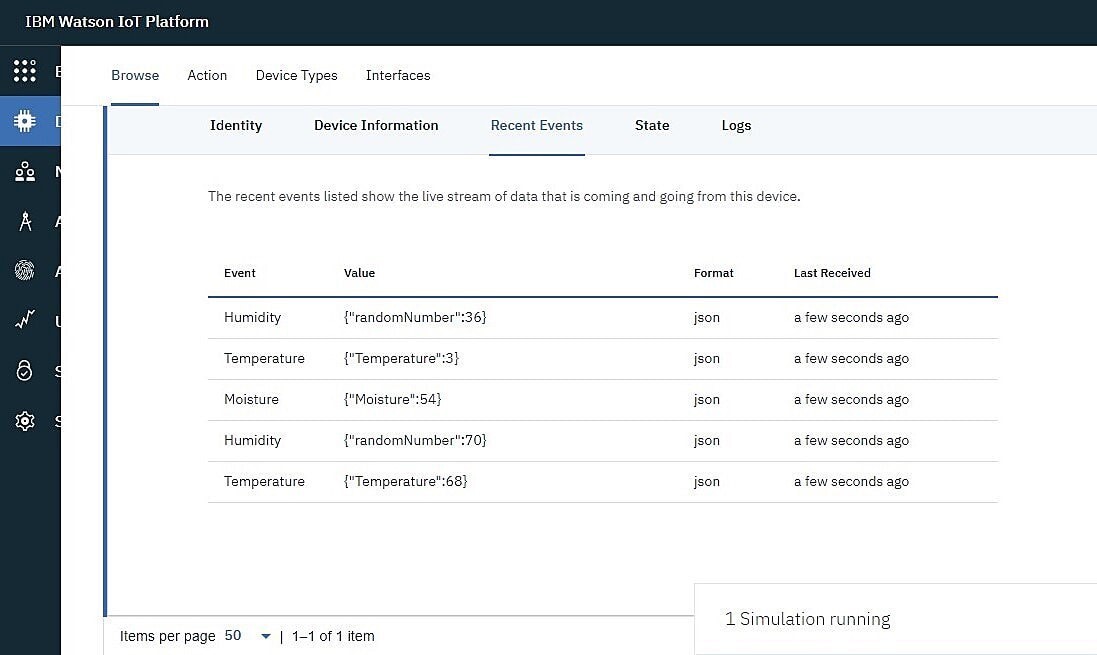
} else {

digitalWrite(13, LOW);

}

delay(10); // Delay a little bit to improve simulation performance

}



FEATURES:

Output: Digital pulse high (3V) when triggered (mo on detected) digital low when idle (no mo on 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:**

* RatedVoltage : 6V DC
* Opera ng Voltage : 4 to 8V DC
* Rated Current\*: ≤30mA
* SoundOutput at 10cm\* : ≥85dB
* Resonant Frequency : 2300 ±300Hz
* Tone: Con nuous 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, pneuma c and electronic.

**FEATURE 2:**

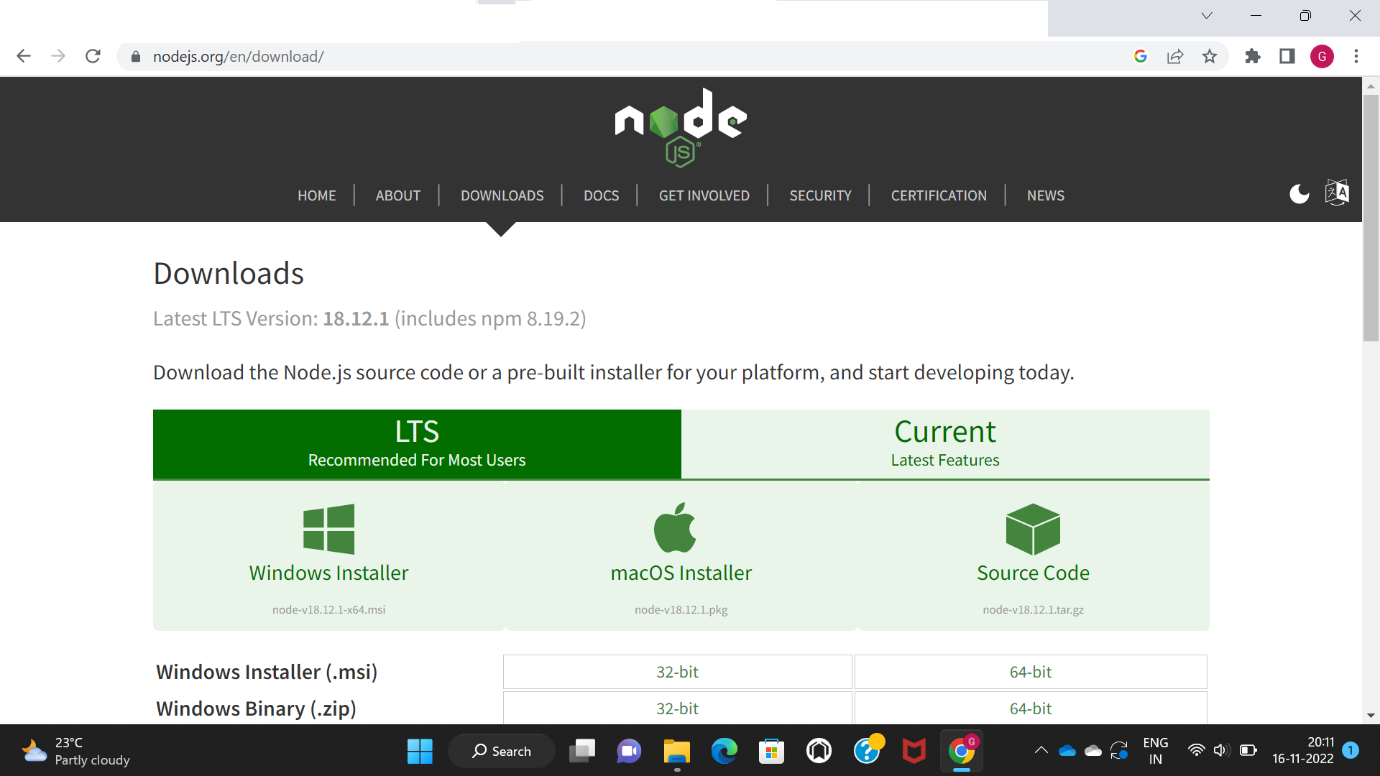
* 1. Goodsensi vity to Combus ble gas in wide range .
  2. Highsensi vity to LPG, Propane and Hydrogen .
  3. Longlife and low cost.
  4. Simpledrive circuit.

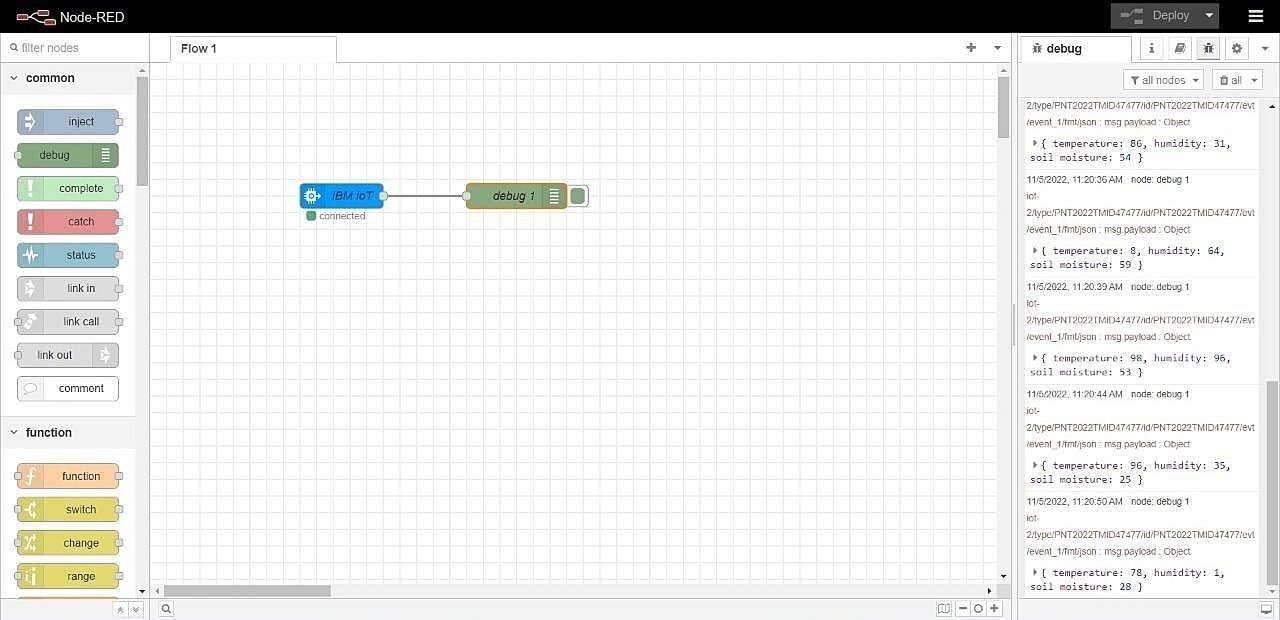
**TESTING**

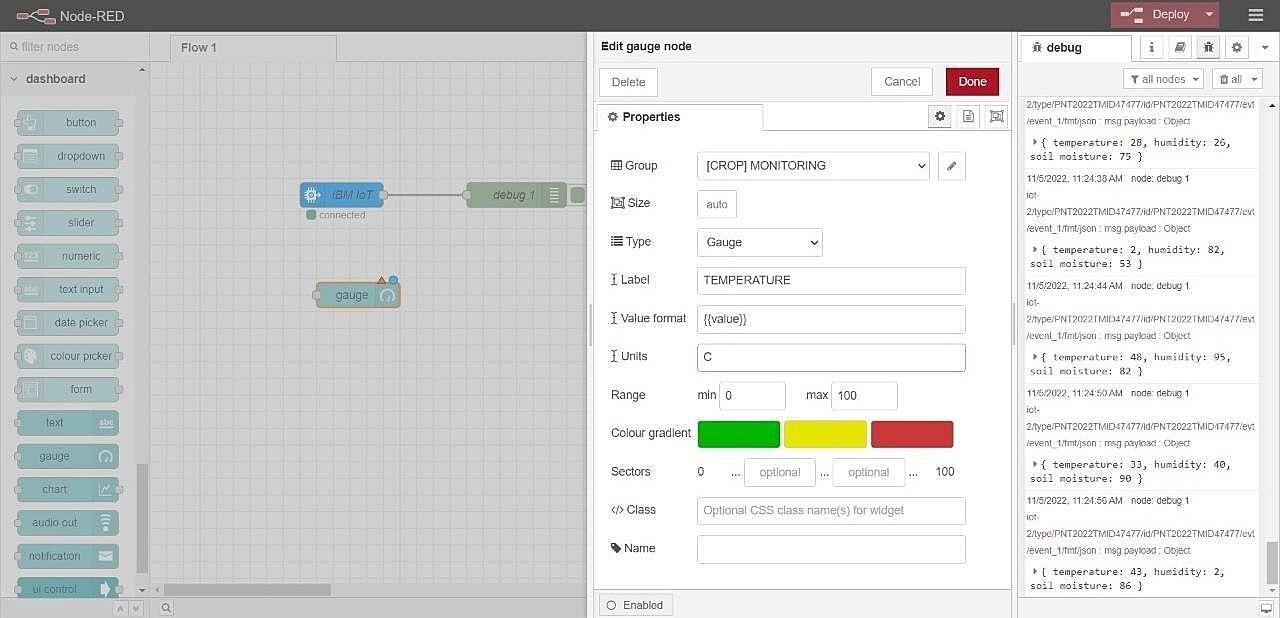
TEST CASES:

|  |  |  |  |
| --- | --- | --- | --- |
| 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:**

****





RESULTS

In the present, agricultural vandalism caused by wild animals and fire has grown to be a significant social issue.

Given that there is currently no working solution to this issue, it needs urgent attention. As a result, this project has significant social significance because it seeks to solve this issue. This project will assist farmers in safeguarding their orchards and fields, save them from suffering major financial losses, and spare them from making futile efforts to safeguard their fields. They will also benefit from higher food yields, which will improve their economic well-being.

**ADVANTAGES:**

controlled access to food. If you are cultivating the crops and breeding them to be more resilient, you have a better chance of avoiding droughts or floods. It enables farmers to increase yields while utilising the least amount of water and fertiliser.

**DISADVANTAGES:**

The biggest drawback is the potential processing time for the data. You need to drastically alter the planet's environment if you want to continue feeding people as the population increases.

**CONCLUSION**

A Watson IoT platform, Watson simulator, IBM cloud, and Node-RED were used to create an IoT web application for a smart agricultural system.

**FUTURE SCOPE**

Future versions of this technology might have a very broad scope, based on image processing, where cameras could identify wild animals and fire and, if they approached a farm, immediately activate the system using wireless networks. Wild animals can also be found via wireless networks, such as laser wireless sensors, and the security system will be activated when it detects them.

**APPENDIX**

**SOURCE CODE:**

import wiotp.sdk.device import time import random myConfig={

"identity": (

"orgId": "gagtey",

"typeId": "GPS",

"deviceId":"12345"},

"auth": {

"token": "12345678"

}}

def myCommandCallback (cmd):

print ("Message received from IBM IoT Platform: %s" % cmd.data['command']) m-cmd.data['command'] client= wiotp.sdk.device.DeviceClient (config=myConfig, logHandlers=None) client.connect() def pub (data):

client.publishEvent (eventId="status", msgFormat="json", data=myData, qos=0, print("Published data Successfully: %s", myData) while True:

myData={'name': 'Train1', 'lat': 17.6387448, 'lon':

78.4754336) pub (myData) time.sleep (3)

#myData('name': 'Train2', 'lat': 17.6387448, 'lon':

78.4754336)

#pub (myData) #time.sleep (3) myData={'name': 'Train1', 'lat': 17.6341908, 'lon':

78.4744722) pub (myData) time.sleep(3)

myData={'name': 'Trainl', 'lat': 17.6340889, lon': 78.4745052) pub (myData) time.sleep(3)

myData={'name': 'Trainl', 'lat': 17.6248626, 'lon': 78.4720259) pub (myData) time.sleep (3)

myData={'name': 'Trainl', 'lat': 17.6188577, 'lon': 78.4698726) pub (myData) time.sleep (3) myData={'name': 'Train1', 'lat': 17.6132382, 'lon':

78.4707318) pub (myData) time.sleep (3)

client.commandCallback = myCommandCallback client.disconnect()

**QR SCANNER CODE:** Import cv2 import numpy as np import time

Import pyzbar.pyzbar as pyzbar

from ibmcloudant.cloudant\_v1 import CloudantV1 from ibmcloudant import CouchDbSessionAuthenticator from ibm\_cloud\_ sdk\_core.authenticators import BasicAuthenticator

authenticator= BasicAuthenticator ('apikey-v2-

16u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz',

'b0ab119f45d3e6255eabb978

service Cloudant V1 (authenticator-authenticator) service.set\_service\_url('https://apikey-v216u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz:b0ab119 f45d3e6255eabb978e7e2f0 cap= cv2.VideoCapture (0) font cv2.FONT HERSHEY PLAIN while True:

frame cap.read()

decodedobjects pyzbar.decode (frame) for obj in decodedObjects: #print ("Data", obj.data) a-obj.data.decode('UTF-8')

cv2.putText (frame, "Ticket", (50, 50), font, 2,

(255, 0, 0), 3) #print (a)

try: response = service.get\_document ( db='booking, doc\_id = a

).get\_result()

print (response) time.sleep(5) except Exception as e:

print ("Not a Valid Ticket") time.sleep (5) cv2.imshow("Frame", frame) if cv2.waitKey(1) & 0xFF==ord('q'):

break cap.release() cv2.destroyAllWindows () client.disconnect()

GitHub & Project Demo Link:

<https://drive.google.com/file/d/1zSeC85RFPGMFsLz7YA9DQI9iB9DvDLJ6/view?usp=sharing>