IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

TEAM ID : PNT2022TMID35045

TEAM MEMBERS :

AARTHIKA.R

NAGATHANSI.P

KEERTHANA.M

JOTHIKA.K

INTRODUCTION :

Crops in farms are many times ravaged by local animals like buffaloes ,cows,goat,dog,birds etc.This leads to huge losses for the farmers.Due to over population, it occurs a deforestation this results in shortage of food,water and shelter in forest area.So,animals interference in residential areas in increaseing day by day which affect human life and property causes human,animal conflict but as per natures rule every living creature on this earth has important role in ecosystem.Elephant and other animals coming into contact with humans,impact negatively in various means such as by depredation of crops,damaging grain stores,water supplies, houeses and other assets,injuring and death of humans

So here we proposed automatic crop protection system from animals.This is a microcontroller-based system using PIC family microcontroller.These systems use a motion sensor to detect wild animals approaching near the field.

LITERATURE SURVEY:

EXISTING PROBLEM:

The existing system mainly provide the surveillance functionality . Also these system dont provide protection from wild animals,especially in such an application areas. They also need to take action based on the type of animal that tries to enter the area, as different methods are adapted to prevent different animals from entering restricted areas. The other commonly used method by farmers inorder to prevent vandilization by animals include building physical barriers ,use of electric fences and manual surveillance and various such exhaustive and dangerous methods.

REFERENCES:

- Dr.M.Chandra, Mohan Reddy, Keerthi Raju Kamakshi kodi, Babitha Anapali Mounika pulla,"SMART CROP PROTECTION SYSTEM FROM LIVING OBJECTS AND FIRE USING ARDUINO ",Science,tecnology and development,volume9 issue9,page no 261-265,september2020.

- G.Naveen Balaji, V.Nandhini, S.Mithra,N.Priya,R.Naveena,"IOT BASED SMART CROP MONITORING IN FARM LAND ", Imperial Journal of inter disciplinary research,Volume4,Issue1,nov2018.

- P.Rekha, T.Saranya,P.Preethi,L.Saraswathy,G.Sobana,"SMART AGRO USING ARDUINO AND GSM",International Journal of emerging technologies in engineering research Volume5,Issue3,mar2017.

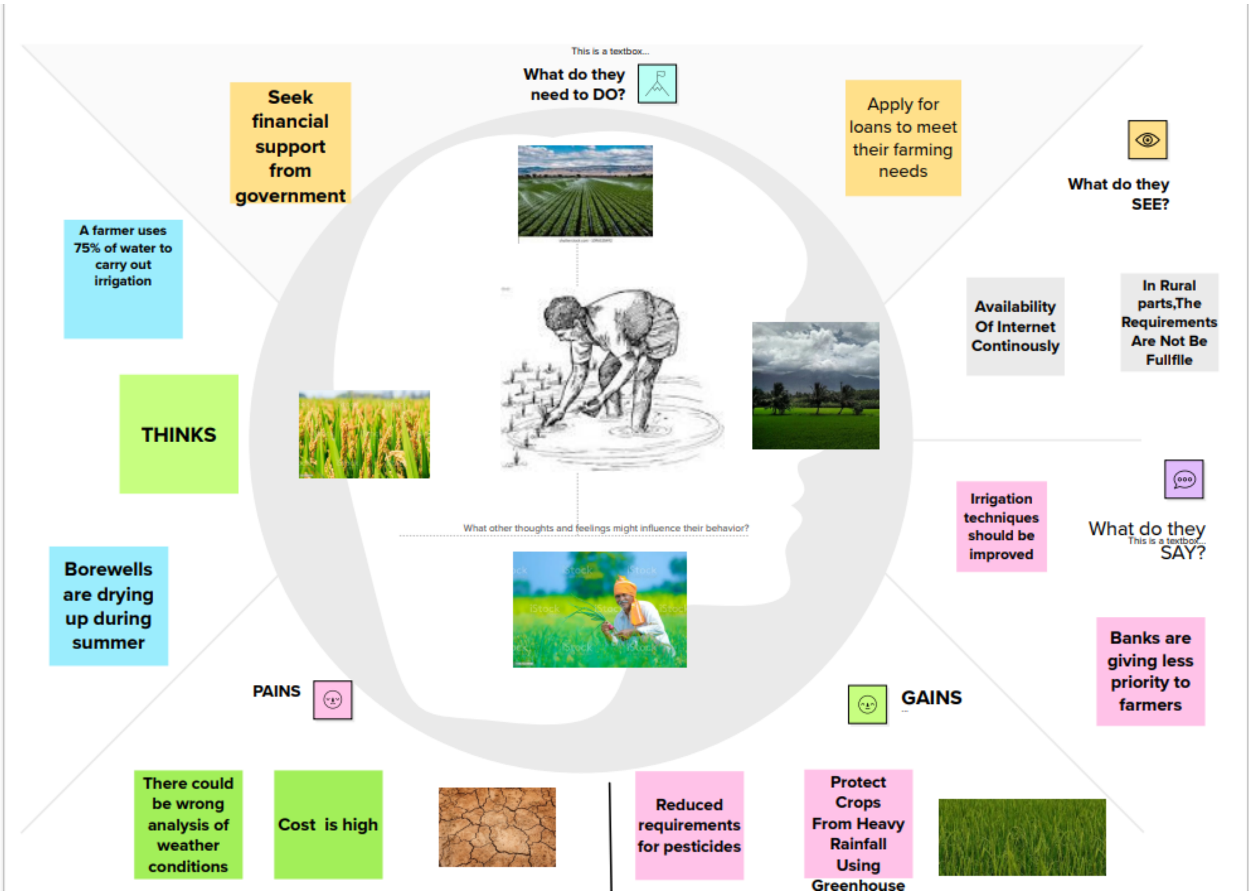
PROBLEM STATEMENT DEFINITION:

- A System is formed to predict the form land with the help of sensors,humidity,soil moisture etc.
- It increase production, Remote Monitoring,lower operation cost,water conservation.
- Poor Internet connectivity in terms,lack of infrastructure.
- It Improves the entire agriculture system by monitoring the field in real -time.

⭐ **Main aim of our project is to protect the crops from damage caused by animals as well as divert the animals without any harm.**

IDEATION AND PROPOSED SOLUTION:

Empathy Map Canvas:



This is a textbox...

What do they need to DO?

Seek financial support from government

Apply for loans to meet their farming needs

What do they SEE?

A farmer uses 75% of water to carry out irrigation

THINKS

Availability Of Internet Continously

In Rural parts,The Requirements Are Not Be Fullfile

What other thoughts and feelings might influence their behavior?

Irrigation techniques should be improved

What do they SAY?

This is a textbox

Borewells are drying up during summer

Banks are giving less priority to farmers

PAINS

GAINS

There could be wrong analysis of weather conditions

Cost is high

Reduced requirements for pesticides

Protect Crops From Heavy Rainfall Using Greenhouse

IDEATION AND BRAINSTORMING :

idea 1 :

Crops in the farms are many times devastated by the wild as well as domestic animals and low productivity of crops is one of the reasons for this. It is not possible to stay 24 hours in the farm to sentinel the crops. So to surmount this issue an automated perspicacious crop aegis system is proposed utilizing Internet of Things (IOT). 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. As soon as any kineticism is detected the system will engender an alarm to be taken and the lights will glow up implemented at every corner of the farm. This will not harm any animal and the crops will stay forfended

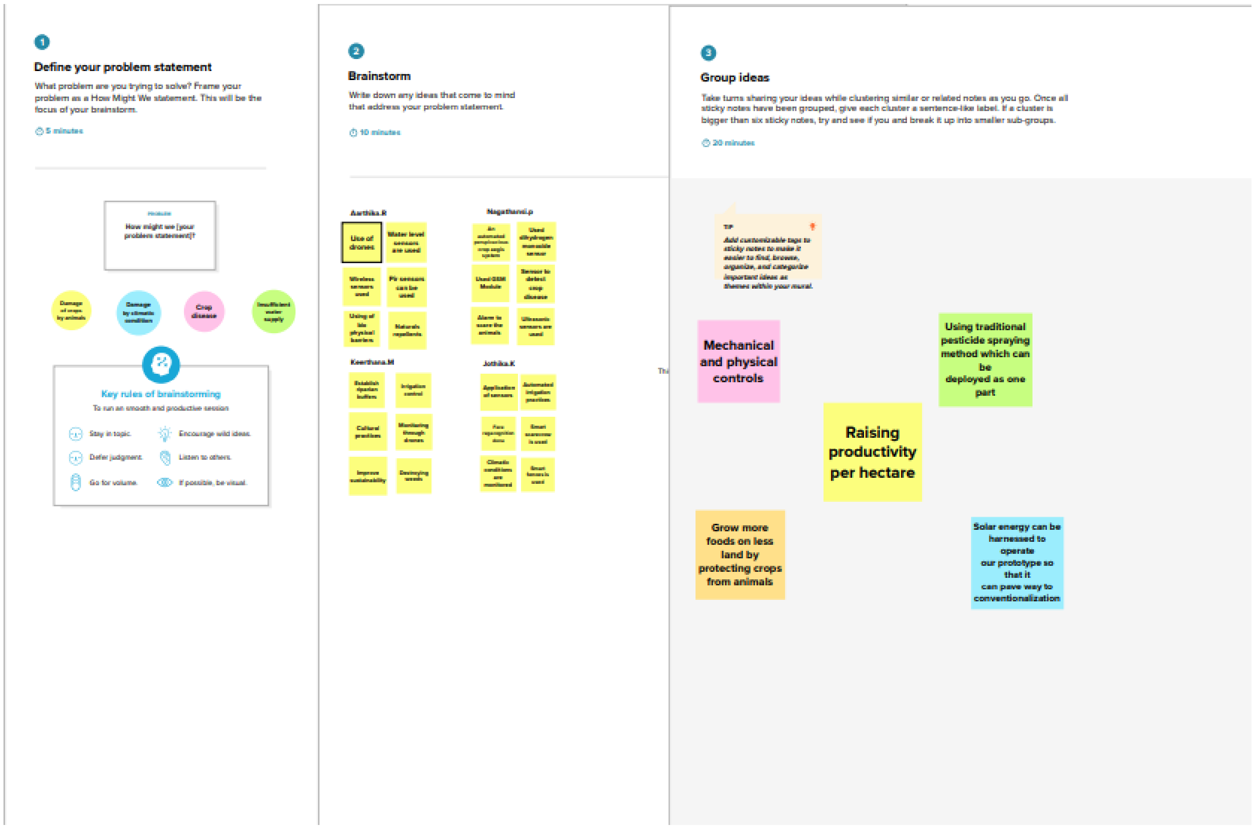
Idea 2:

The Smart protection system defines that this project help to farmer for the protection of a farm. We have designed this project for the only secure from animals but this project have the provision to secure from the human begins also. This can be achieved by the help of IOT device. The SCPS work on the battery so that this project can be easily portable and also we are added 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.

Idea 3:

A centralizing method in the area of IIoT (Industrial Internet of Things) contrived for understanding agriculture which is preceding the arrangements low-power devices . This project yields a monitoring procedure for farm safety against animal attacks and climate change conditions. IIoT advances are frequently used in smart farming to emphasize the standard of agriculture. It contains types of sensors, controllers. On behalf of WSN, the ARM Cortex-A board which consumes 3W is the foremost essence of the procedure . Different sensors like DHT 11 Humidity & Temperature Sensor, PIR Sensor, LDR sensor, HC-SR04 Ultrasonic Sensor, and camera are mounted on the ARM Cortex-A board. The PIR goes high on noticing the movement within the scope, the camera starts to record, and the data will be reserved onboard and in the IoT cloud, instantaneously information will be generated automatically towards the recorded

quantity using a SIM900A unit to notify about the interference with the information of the weather conditions attained by DHt11. If a variance happens, 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.



1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we [your problem statement]?

Damage of crops by animals

Damage by climatic condition

Crop disease

Insufficient water supply

Key rules of brainstorming

To run an smooth and productive session

Stay in topic.

Encourage wild ideas.

Defer judgment.

Listen to others.

Go for volume.

If possible, be visual.

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

Aarthika.R

Use of drones

Water level sensors are used

Wireless sensors used

Pir sensors can be used

Using of bio physical barriers

Naturals repellents

Nagathervi.p

An automated precision crop angle system

Used dihydrogen monoxide sensor

Used GSM Module

Sensor to detect crop disease

Alarm to scare the animals

Ultrasonic sensors are used

Keerthana.M

Establish riparian buffers

Irrigation control

Cultural practices

Monitoring through drones

Improve sustainability

Destroying weeds

Jothika.K

Application of sensors

Automated irrigation practices

Face recognition done

Smart scarecrow is used

Climatic conditions are monitored

Smart fences is used

3

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

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

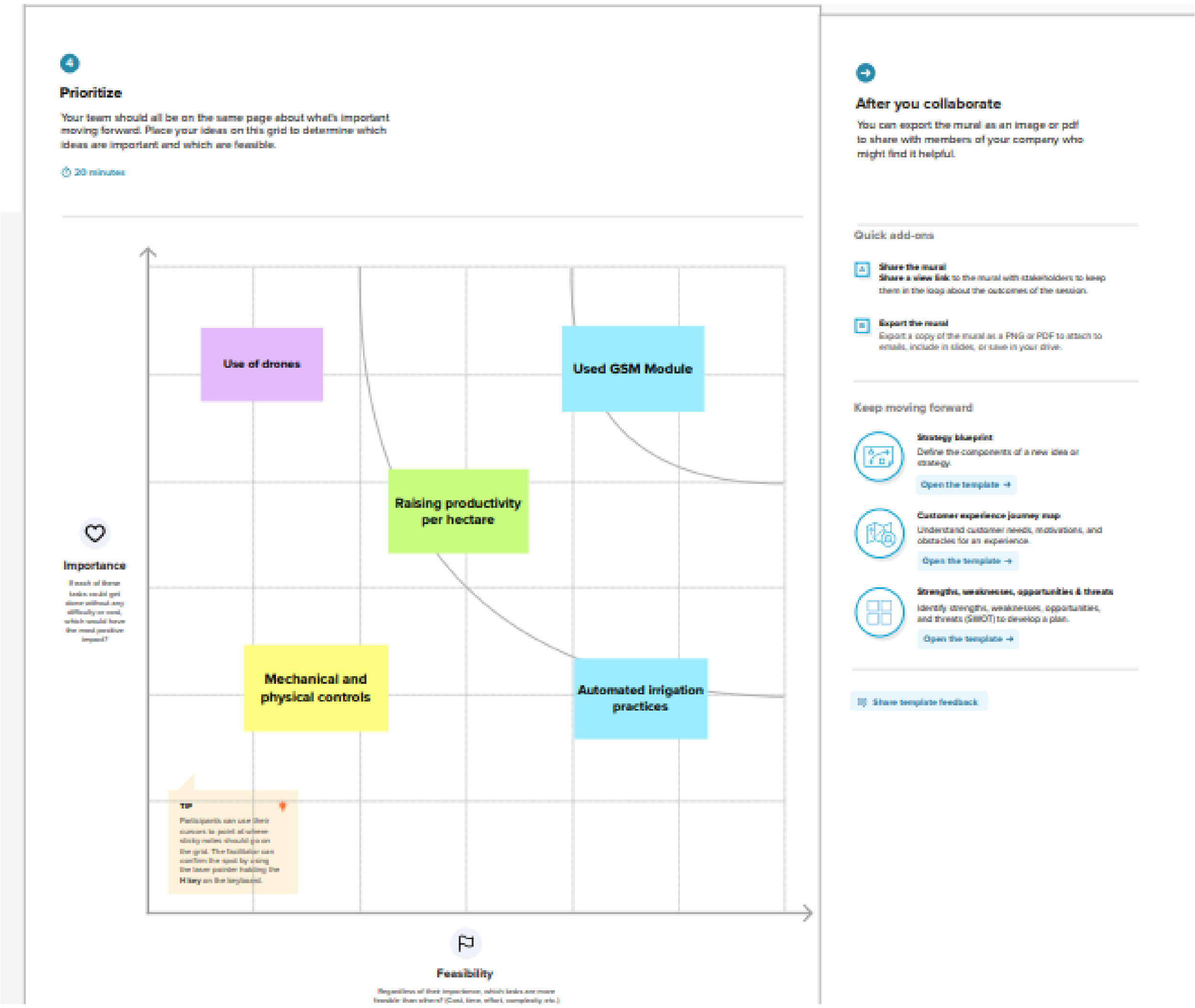
Mechanical and physical controls

Using traditional pesticide spraying method which can be deployed as one part

Raising productivity per hectare

Grow more foods on less land by protecting crops from animals

Solar energy can be harnessed to operate our prototype so that it can pave way to conventionalization



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

Use of drones

Used GSM Module

Raising productivity per hectare

Mechanical and physical controls

Automated irrigation practices

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

Share the mural
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

Export the mural
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

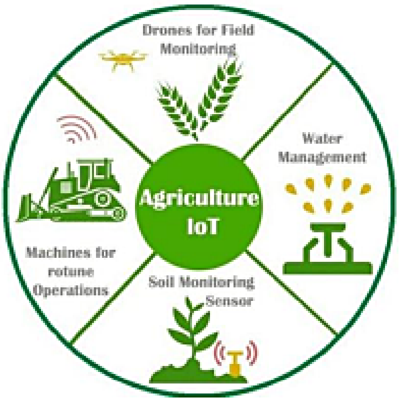
Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

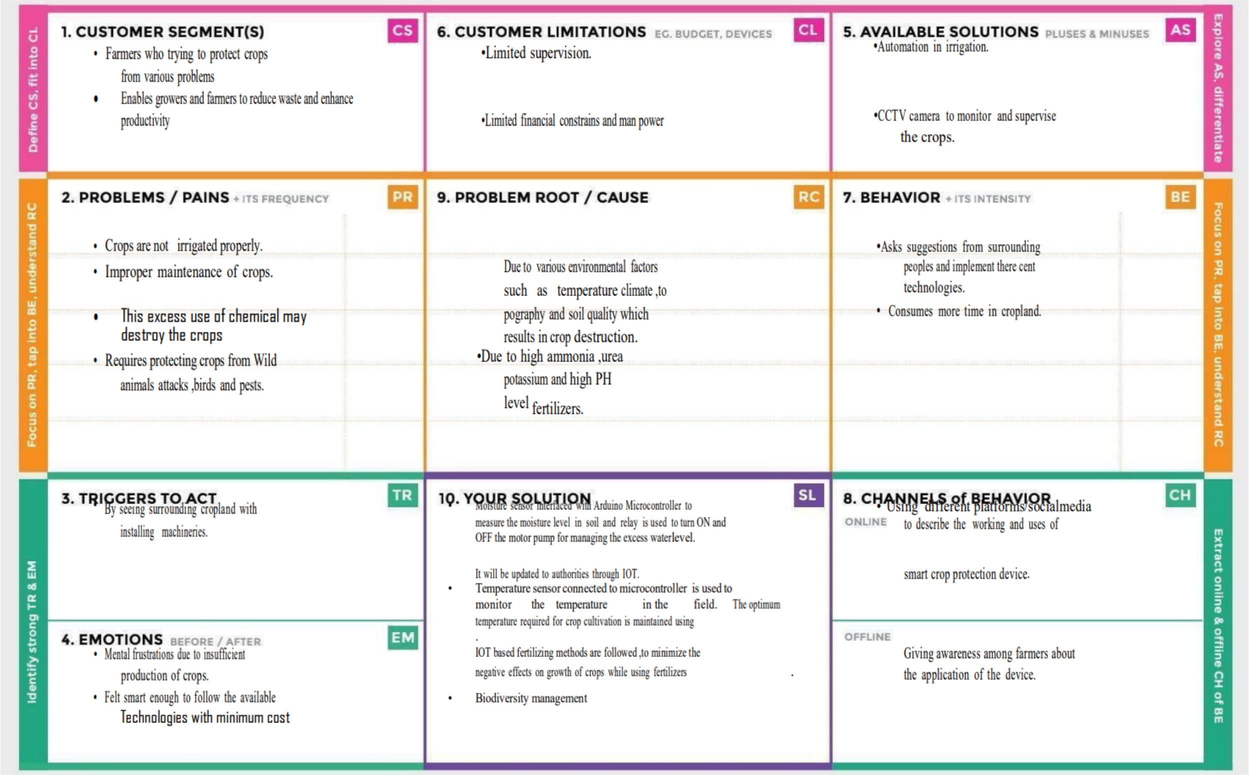
Share template feedback

PROPOSED SOLUTION :

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Develop smart & affordable solution to protect crops from wild animals
2.	Idea / Solution description	With the help of remote sensing technologies develop crop protection solution from wild animal attacks. Provide alerts on any crop damage in case animals destroy crops.
3.	Novelty / Uniqueness	Role of SENSORS:IOT smart agriculture products are designed to help monitor cropped fields using sensors and by automating irrigation systems. As a result, farmers and associated brands and easily monitor the field conditions from anywhere without any hassle.
4.	Social Impact / Customer Satisfaction	Conservation of water, optimization of energy resources, Better crop yield, Saves lot of time, Increased quality of production, Real time data and production insight, Remote monitoring.
5.	Business Model (Revenue Model)	 Drones for Field Monitoring Water Management Agriculture IoT Machines for rotune Operations Soil Monitoring Sensor
6.	Scalability of the Solution	By the time we have 9 billion people on the planet, 70% of them will live in urban areas. IOT-based greenhouses and hydroponic systems enable short food supply chains and should be able to feed the people. Smart closed-cycle agricultural systems allow growing food everywhere—in supermarkets, on skyscrapers' walls and rooftops in shipping containers, and ,of course, in the comfort of everyone's home.

PROBLEM SOLUTION FIT :



Define CS, fit into CL

1. CUSTOMER SEGMENT(S) CS

- Farmers who trying to protect crops from various problems
- Enables growers and farmers to reduce waste and enhance productivity

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL

•Limited supervision.

•Limited financial constrains and man power

5. AVAILABLE SOLUTIONS PLUSES & MINUSES AS

•Automation in irrigation.

•CCTV camera to monitor and supervise the crops.

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2. PROBLEMS / PAINS + ITS FREQUENCY PR

- Crops are not irrigated properly.
- Improper maintenance of crops.
- This excess use of chemical may destroy the crops
- Requires protecting crops from Wild animals attacks ,birds and pests.

9. PROBLEM ROOT / CAUSE RC

Due to various environmental factors such as temperature climate ,topography and soil quality which results in crop destruction.

•Due to high ammonia ,urea potassium and high PH level fertilizers.

7. BEHAVIOR + ITS INTENSITY BE

•Asks suggestions from surrounding peoples and implement there cent technologies.

- Consumes more time in cropland.

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TO ACT TR

By seeing surrounding cropland with installing machineries.

4. EMOTIONS BEFORE / AFTER EM

- Mental frustrations due to insufficient production of crops.
- Felt smart enough to follow the available Technologies with minimum cost

10. YOUR SOLUTION SL

- Moisture sensor interfaced with Arduino Microcontroller to measure the moisture level in soil and relay is used to turn ON and OFF the motor pump for managing the excess waterlevel.

It will be updated to authorities through IOT.

- 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 .
- Biodiversity management

8. CHANNELS of BEHAVIOR CH

ONLINE

- Using different platforms/socialmedia to describe the working and uses of smart crop protection device.

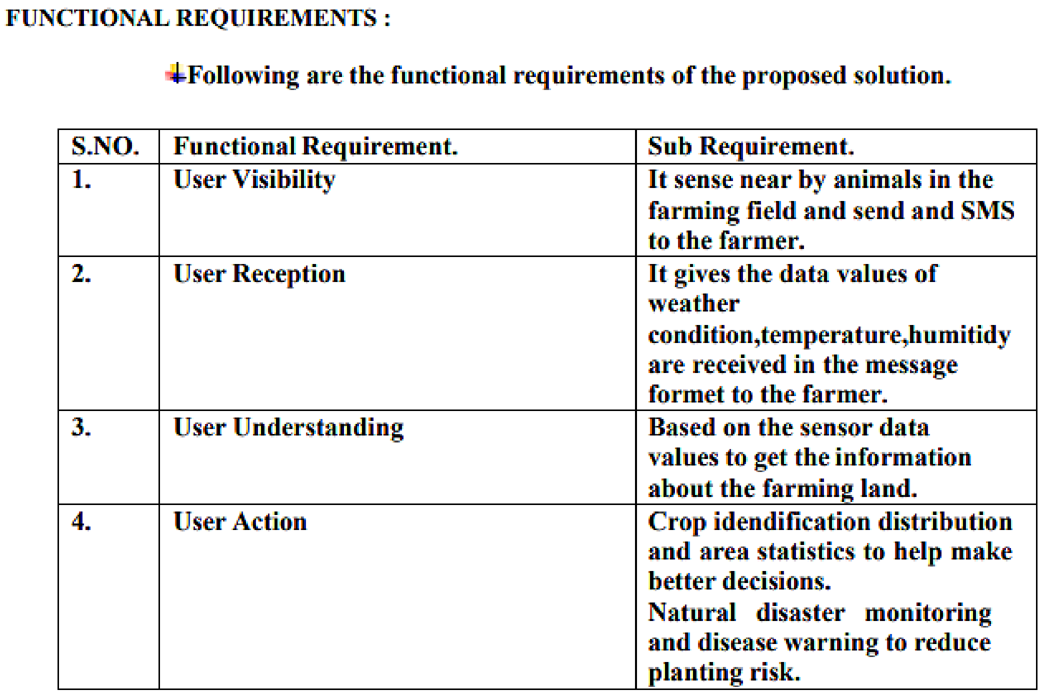
OFFLINE

Giving awareness among farmers about the application of the device.

Extract online & offline CH of BE

REQUIREMENT ANALYSIS :

FUNCTIONAL REQUIREMENTS:

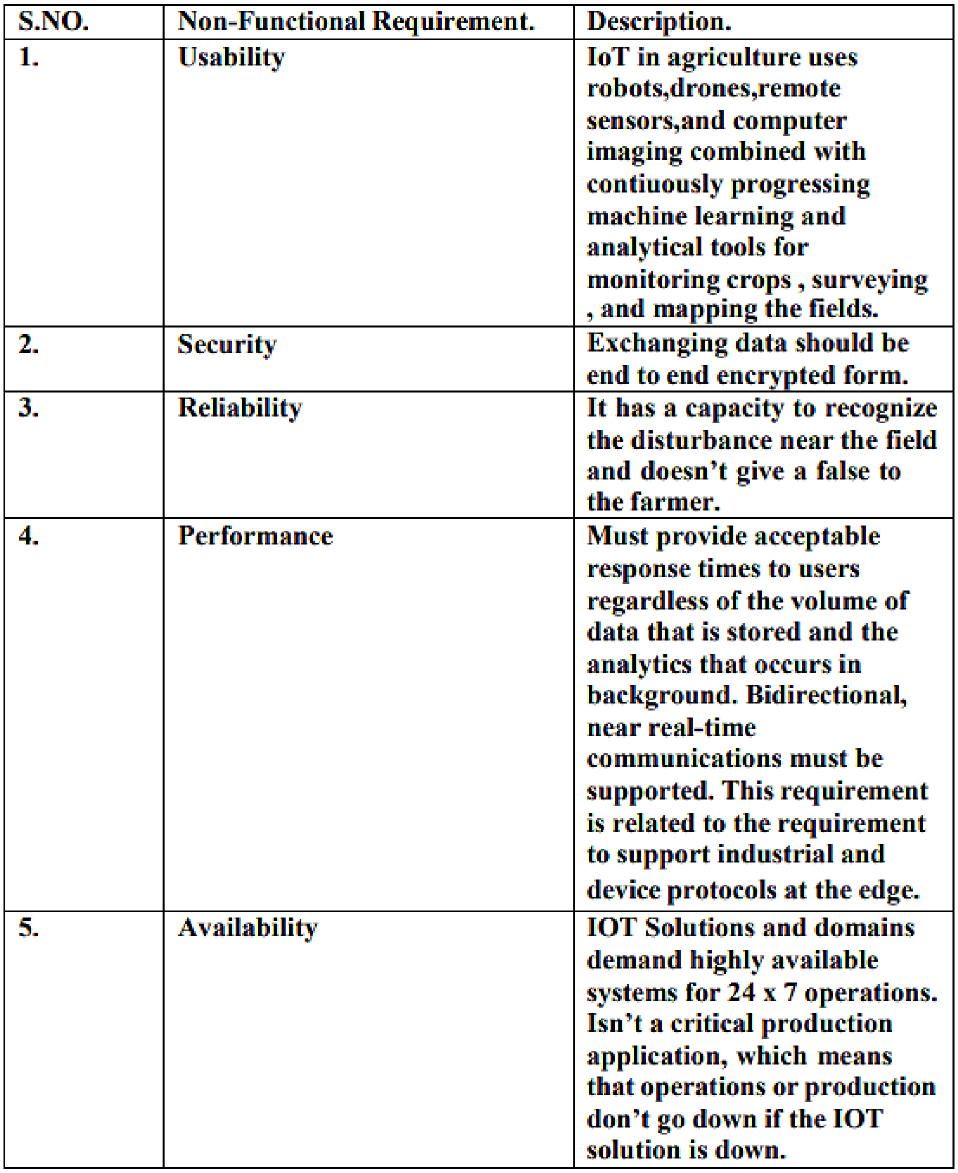


FUNCTIONAL REQUIREMENTS :

- Following are the functional requirements of the proposed solution.

S.NO.	Functional Requirement.	Sub Requirement.
1.	User Visibility	It sense near by animals in the farming field and send and SMS to the farmer.
2.	User Reception	It gives the data values of weather condition,temperature,humitidy are received in the message formet to the farmer.
3.	User Understanding	Based on the sensor data values to get the information about the farming land.
4.	User Action	Crop idendification distribution and area statistics to help make better decisions. Natural disaster monitoring and disease warning to reduce planting risk.

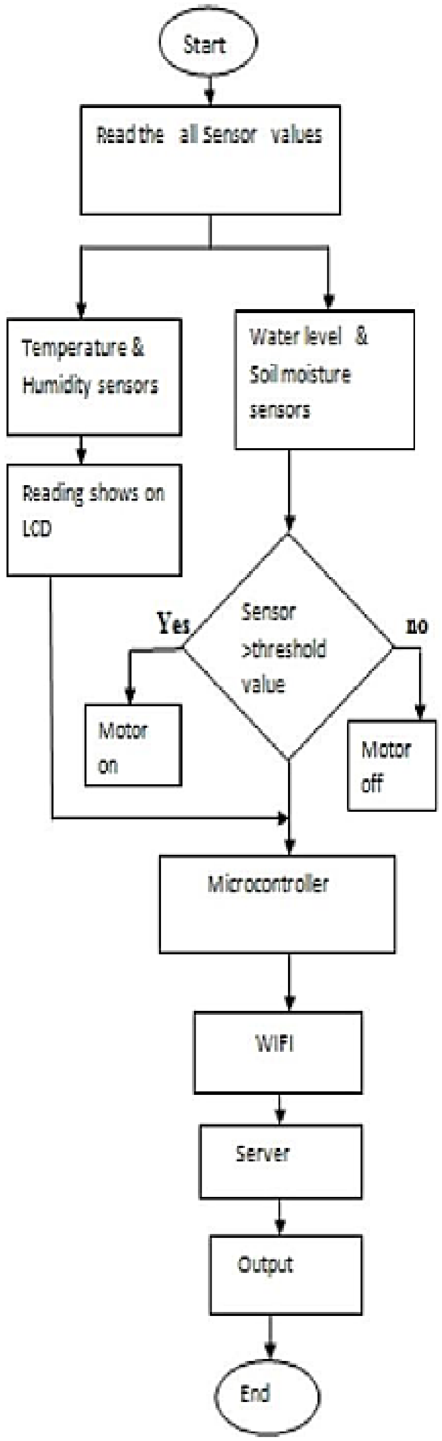
NON FUNCTIONAL REQUIREMENTS :



S.NO.	Non-Functional Requirement.	Description.
1.	Usability	IoT in agriculture uses robots,drones,remote sensors,and computer imaging combined with contiuously progressing machine learning and analytical tools for monitoring crops , surveying , and mapping the fields.
2.	Security	Exchanging data should be end to end encrypted form.
3.	Reliability	It has a capacity to recognize the disturbance near the field and doesn't give a false to the farmer.
4.	Performance	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.
5.	Availability	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.

PROJECT DESIGN :

DATA FLOW :



SOLUTION AND TECHNICAL ARCHITECTURE :

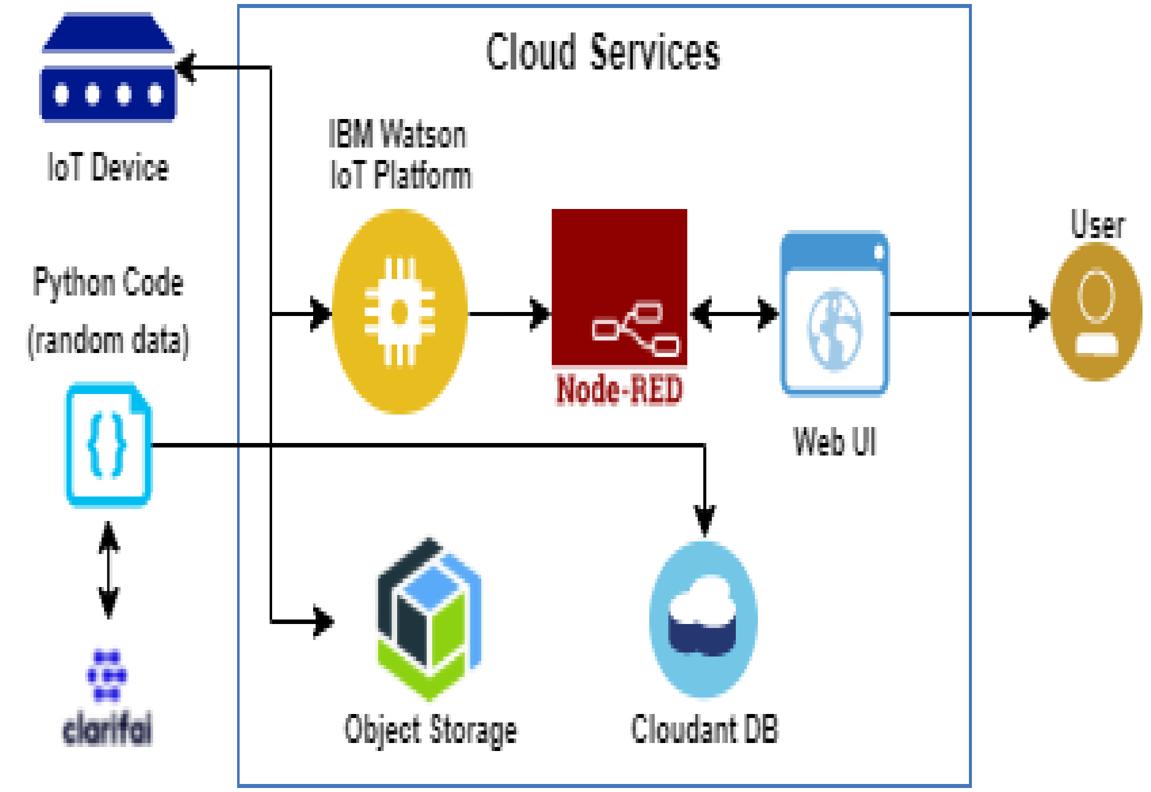
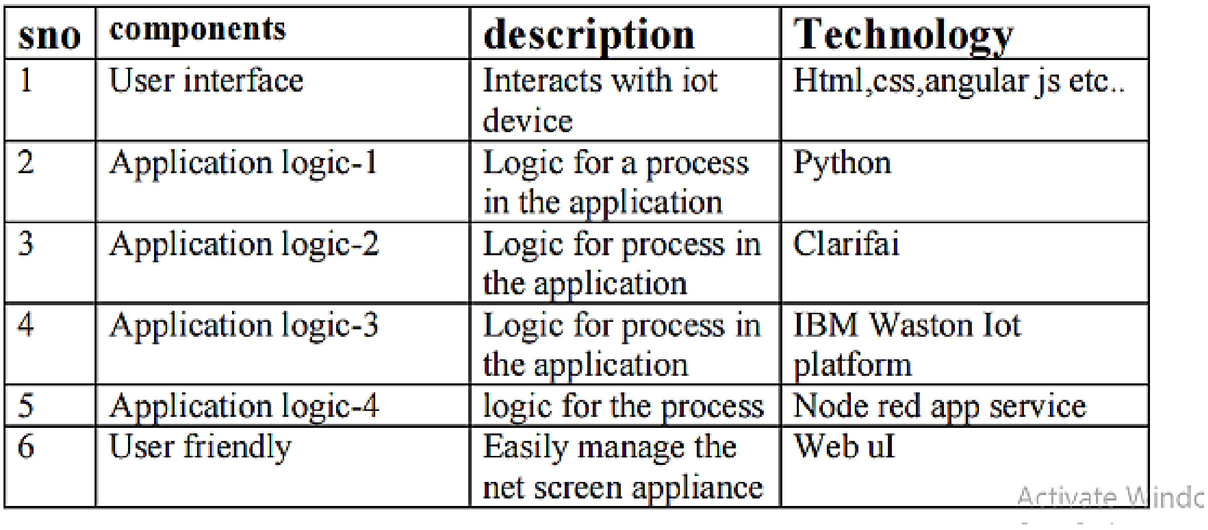
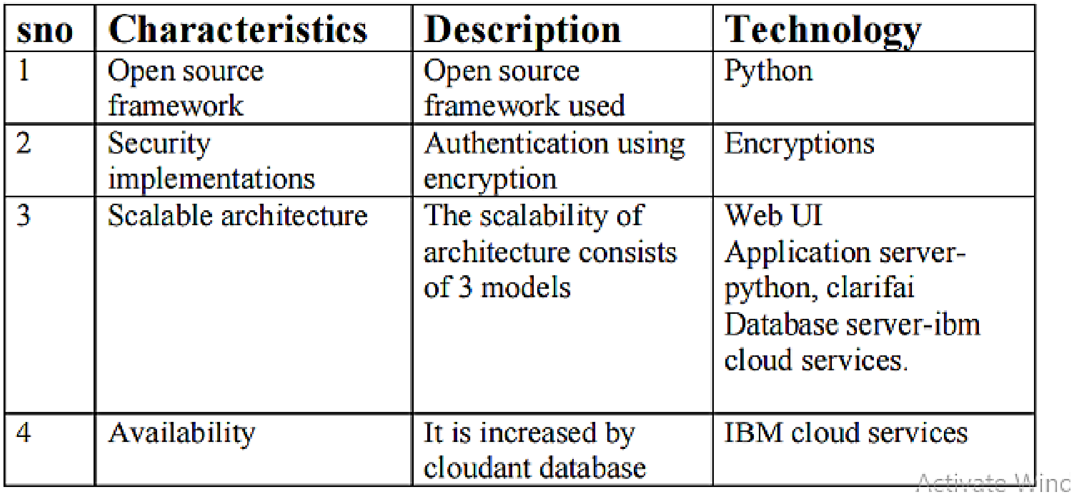


TABLE 1:



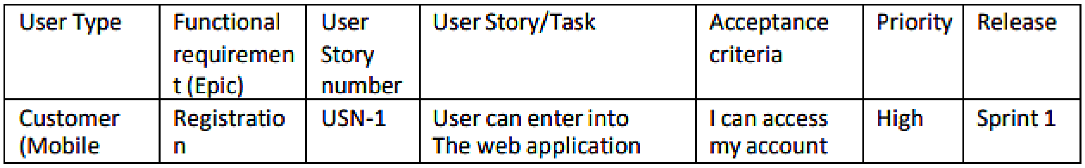
sno	components	description	Technology
1	User interface	Interacts with iot device	Html,css,angular js etc..
2	Application logic-1	Logic for a process in the application	Python
3	Application logic-2	Logic for process in the application	Clarifai
4	Application logic-3	Logic for process in the application	IBM Waston Iot platform
5	Application logic-4	logic for the process	Node red app service
6	User friendly	Easily manage the net screen appliance	Web uI

TABLE 2: APPLICATION AND CHARCTERISTICS

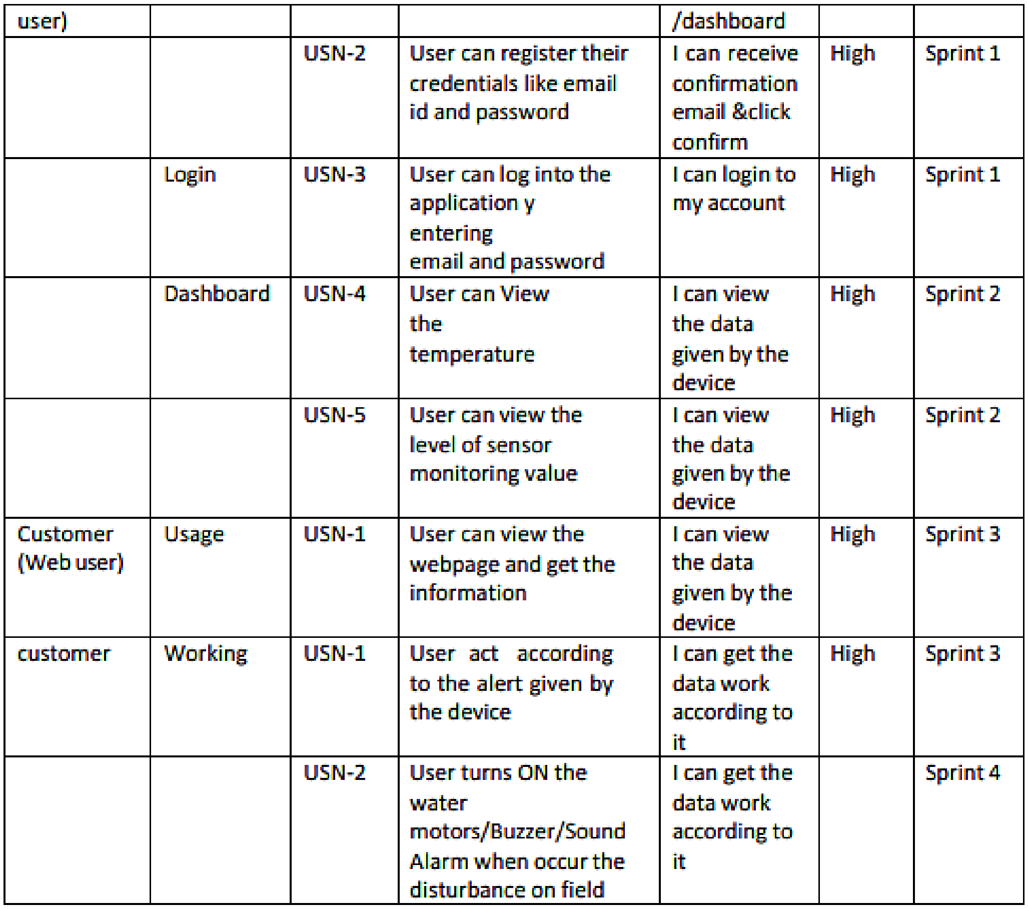


sno	Characteristics	Description	Technology
1	Open source framework	Open source framework used	Python
2	Security implementations	Authentication using encryption	Encryptions
3	Scalable architecture	The scalability of architecture consists of 3 models	Web UI Application server-python, clarifai Database server-ibm cloud services.
4	Availability	It is increased by cloudant database	IBM cloud services

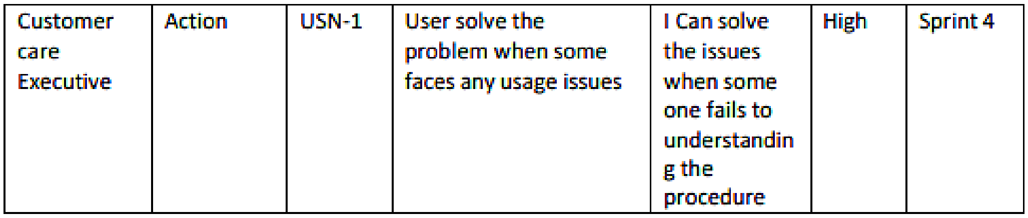
USER STORIES :



User Type	Functional requirement (Epic)	User Story number	User Story/Task	Acceptance criteria	Priority	Release
Customer (Mobile	Registration	USN-1	User can enter into The web application	I can access my account	High	Sprint 1

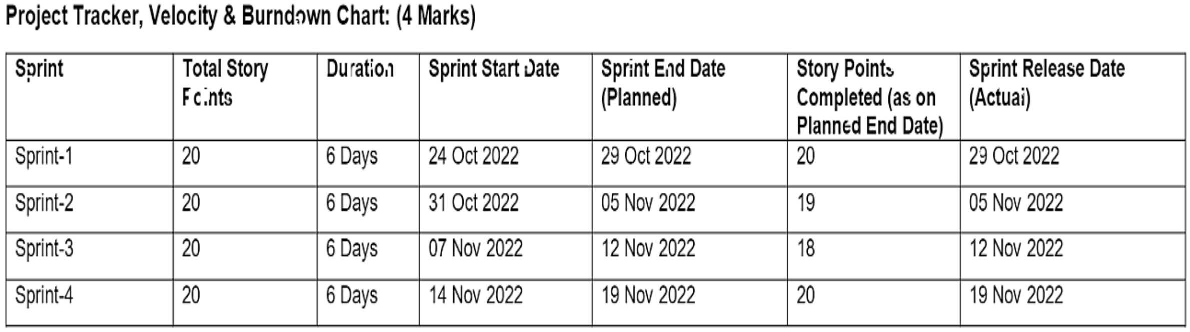


user)				/dashboard		
		USN-2	User can register their credentials like email id and password	I can receive confirmation email &click confirm	High	Sprint 1
	Login	USN-3	User can log into the application y entering email and password	I can login to my account	High	Sprint 1
	Dashboard	USN-4	User can View the temperature	I can view the data given by the device	High	Sprint 2
		USN-5	User can view 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 webpage and get the information	I can view the data given by the device	High	Sprint 3
customer	Working	USN-1	User act 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/Sound Alarm when occur the disturbance on field	I can get the data work according to it		Sprint 4



Customer care Executive	Action	USN-1	User solve the problem when some faces any usage issues	I Can solve the issues when some one fails to understanding the procedure	High	Sprint 4

SPRINT PLANNING AND ESTIMATION :

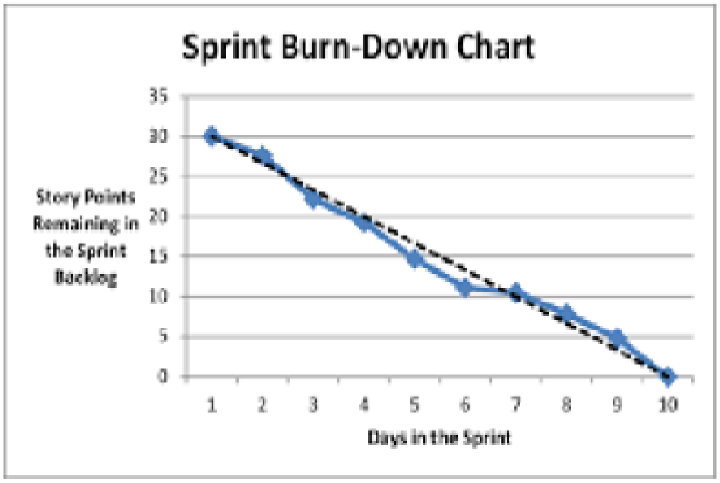


Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	19	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	18	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

VELOCITY :

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



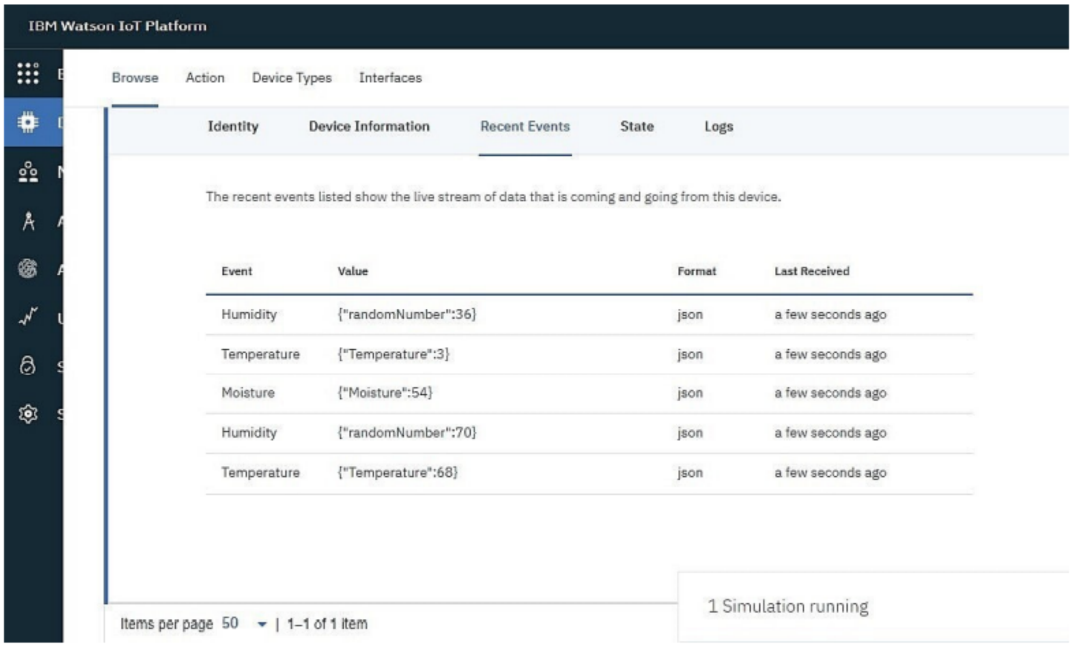
CODING AND SOLUTIONING :

FEATURER 1:

```
import random
 import ibmiotf.application
 import ibmiotf.device
 from time import sleep
import sys
 #IBM Watson Device Credentials.
 organization = "op701j"
 deviceType = "moon"
 deviceId = "moon35"
 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()
```



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

Event	Value	Format	Last Received
Humidity	{"randomNumber":36}	json	a few seconds ago
Temperature	{"Temperature":3}	json	a few seconds ago
Moisture	{"Moisture":54}	json	a few seconds ago
Humidity	{"randomNumber":70}	json	a few seconds ago
Temperature	{"Temperature":68}	json	a few seconds 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 :

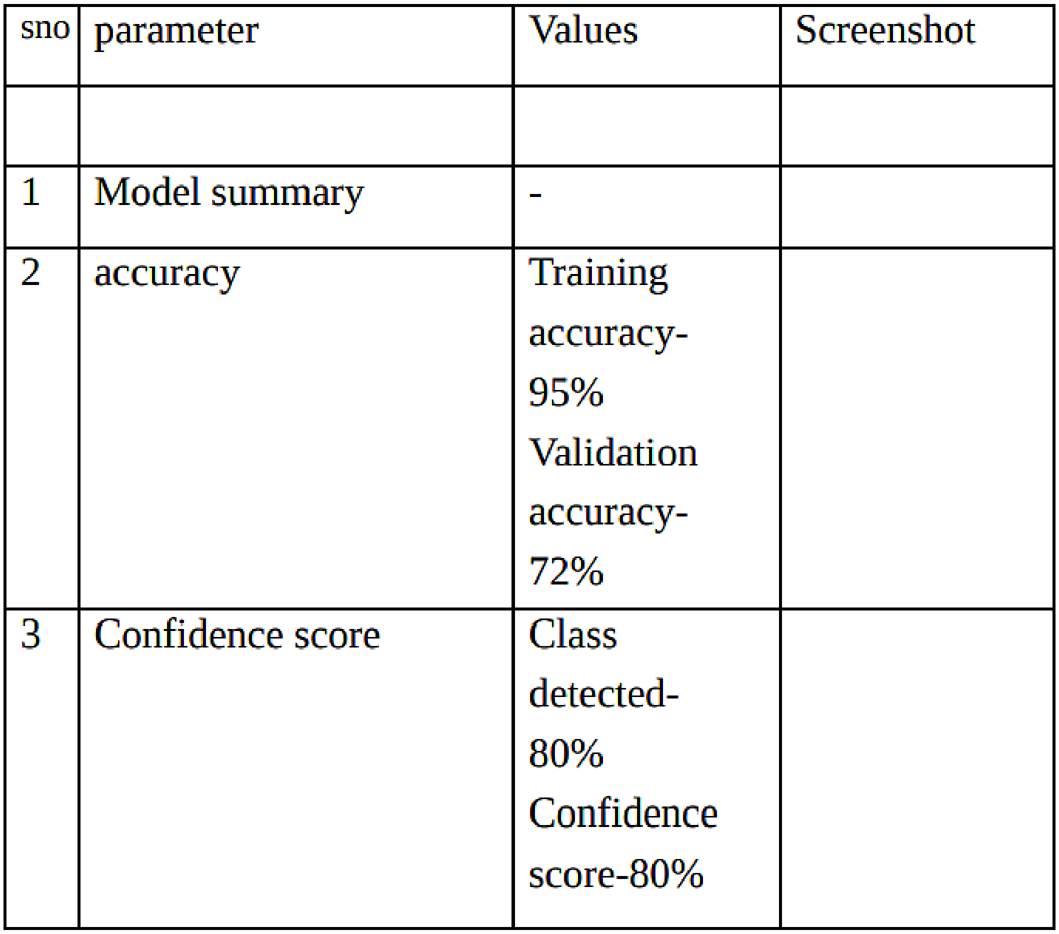
- RatedVoltage : 6V DC
- Operating Voltage : 4 to 8V DC
- Rated Current*: ≤30mA
- SoundOutput at 10cm* : ≥85dB
- Resonant Frequency : 2300 ±300Hz
- Tone: Continuous A buzzer is a loud noise maker.

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.

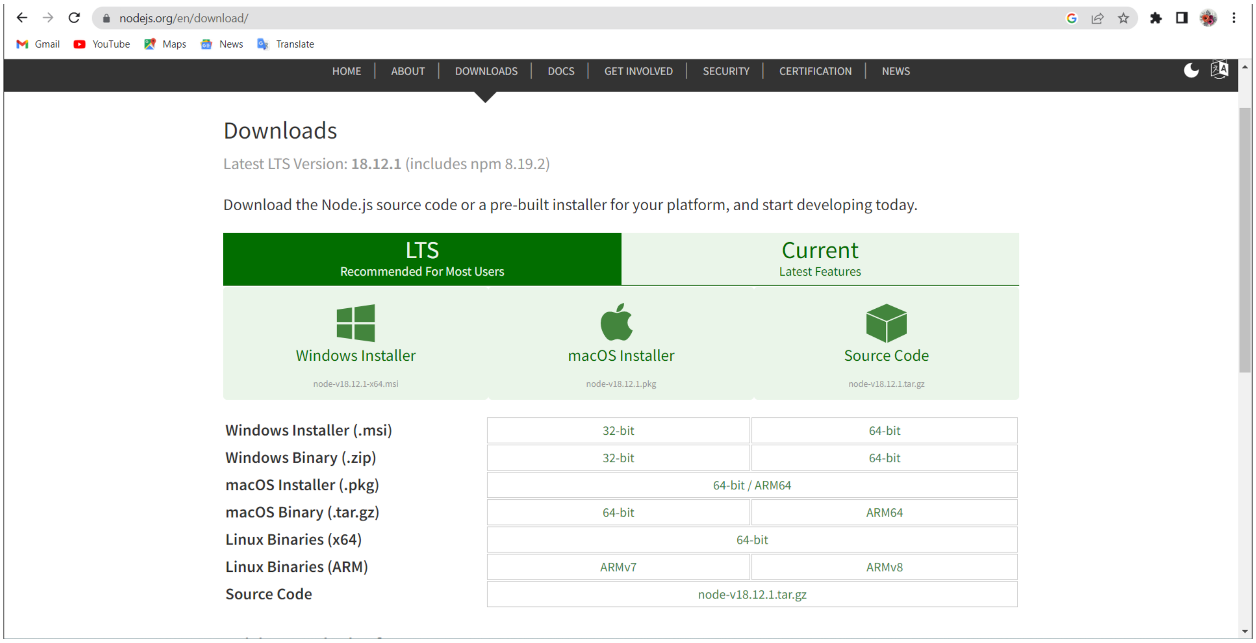
TESTING :

TEST CASES :



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 :



Downloads

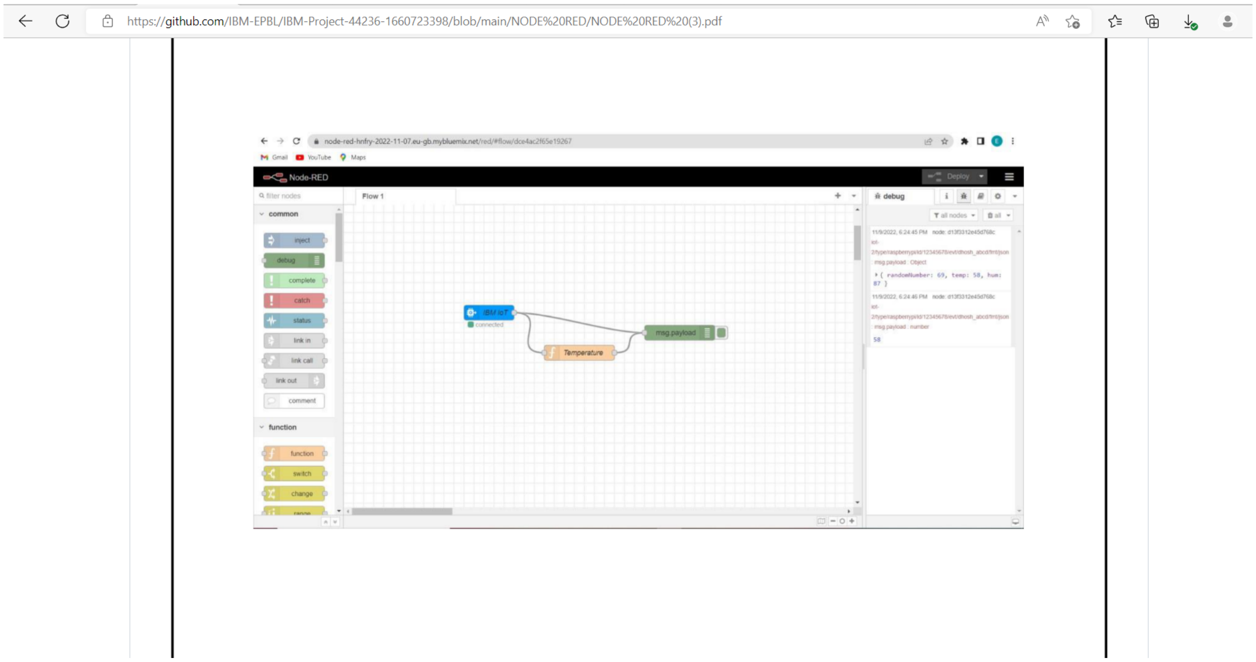
Latest LTS Version: **18.12.1** (includes npm 8.19.2)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

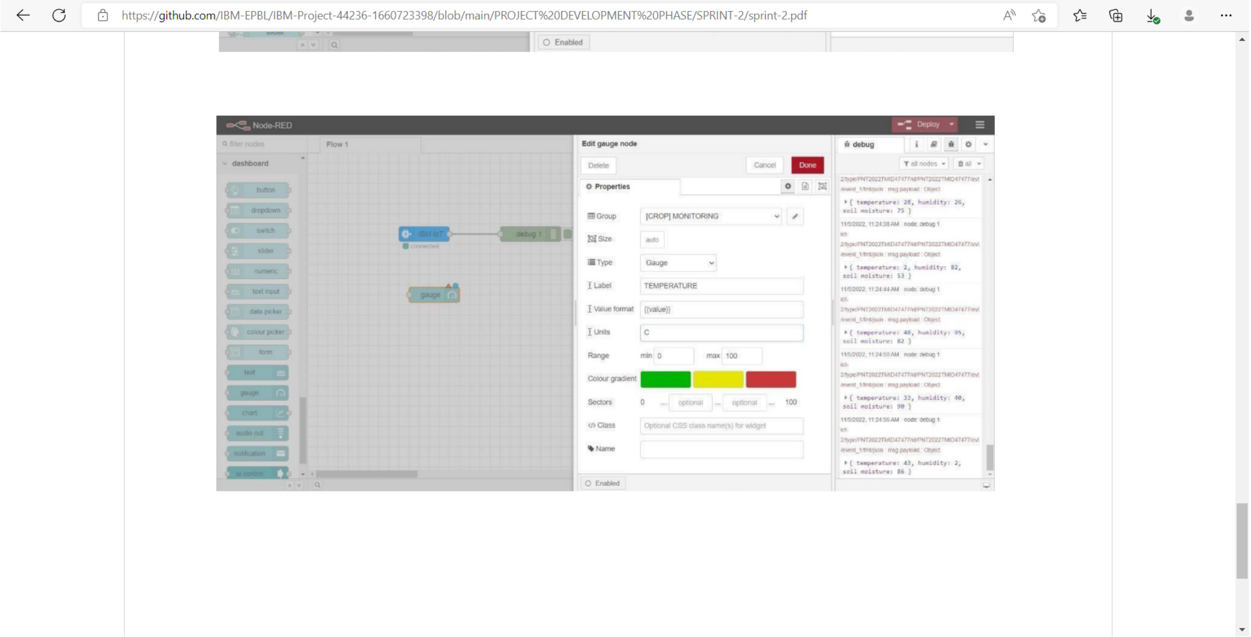
LTS Recommended For Most Users	Current Latest Features

Windows Installer	macOS Installer	Source Code
node-v18.12.1-x64.msi	node-v18.12.1.pkg	node-v18.12.1.tar.gz

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v18.12.1.tar.gz	

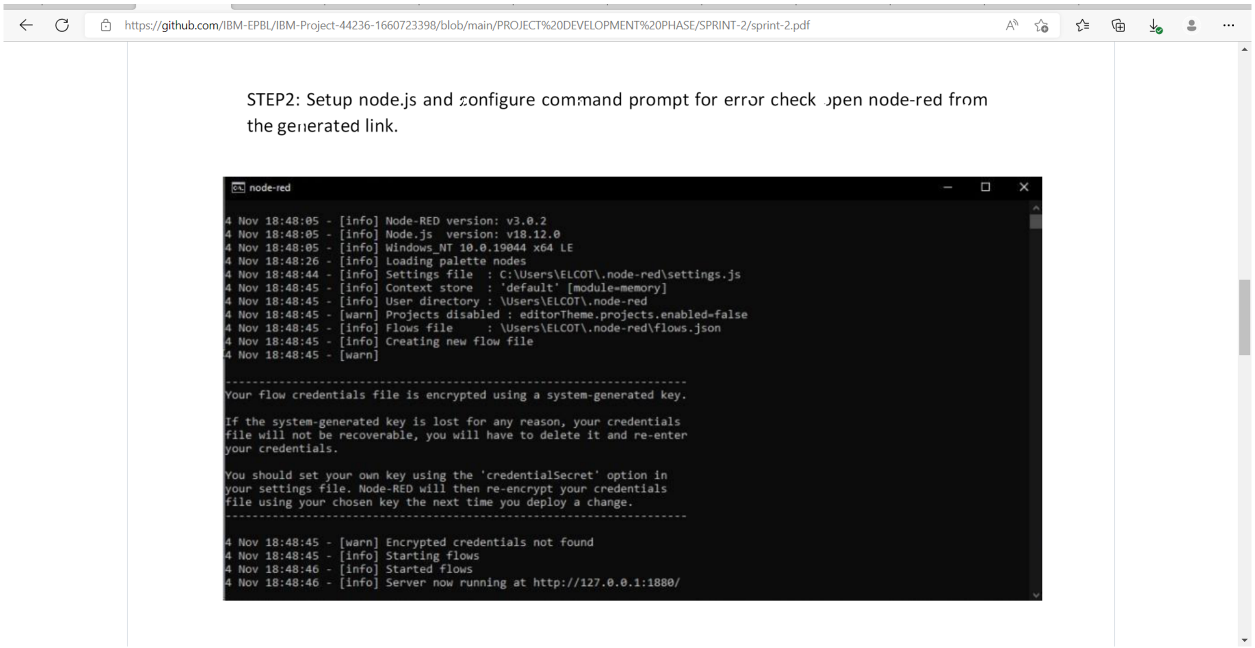


```
11/9/2022, 6:24:45 PM   node: d13f3312e45d768c
iot-2/type/raspberrypi/id/12345678/evt/dhosh_abcd/fmt/json
: msg.payload : Object
▸{ randomNumber: 69, temp: 58, hum: 87 }
11/9/2022, 6:24:46 PM   node: d13f3312e45d768c
iot-2/type/raspberrypi/id/12345678/evt/dhosh_abcd/fmt/json
: msg.payload : number
58
```



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

```
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 28, humidity: 26,
soil moisture: 75 }
11/5/2022, 11:24:38 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 2, humidity: 82,
soil moisture: 53 }
11/5/2022, 11:24:44 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 48, humidity: 95,
soil moisture: 82 }
11/5/2022, 11:24:50 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 33, humidity: 40,
soil moisture: 90 }
11/5/2022, 11:24:56 AM  node: debug 1
iot-
2/type/PNT2022TMID47477/id/PNT2022TMID47477/evt
/event_1/fmt/json : msg.payload : Object
▸{ temperature: 43, humidity: 2,
soil moisture: 86 }
```


https://github.com/IBM-EPBL/IBM-Project-44236-1660723398/blob/main/PROJECT%20DEVELOPMENT%20PHASE/SPRINT-2/sprint-2.pdf
node-red

STEP2: Setup node.js and configure command prompt for error check open node-red from the generated link.

```
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/
```

RESULT :

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 :

- Increase in productivity
- Safe
- Reduced water consumption
- Labour savings

DISADVANTAGES :

- High cost
- Destructive
- Lack of accuracy in sandy soil

CONCLUSION :

Agriculture irrigation control stays unique of the determined significant

interests in agriculture . The simulation result describes the aqua utilization according to the field parameters in the cultivation field. Guideline of horticultural water system stays restrictive to the set up significant interests of farming . The re-enactment result clarifies the utilization of water as indicated by field boundaries in the field of horticulture . Equipment usage and water system control over Android telephones. In the field of IoT, we proposed an integrative way to deal with brilliant horticulture at modern level, zeroed in on low-power crusades and arising causes . This field of this effort remains towards withdraw to monitor the system for crop security conflicting to subconscious occurrences and meteorological conditions.

FUTURE SCOPE:

With a 16% contribution to the gross domestic product (GDP), agriculture still provides livelihood support to about two-thirds of country's population.

** The sector provides employment to 58% of country's work force and is the single largest private sector occupation.**

** Agriculture accounts for about 15% of the total export earnings and provides raw material to a large number of Industries (textiles, silk, sugar, rice, flour mills, milk products).**

** Rural areas are the biggest markets for low-priced and middle-priced consumer goods, including consumer durables and rural domestic savings are an important source of resource mobilization.**

** The agriculture sector acts as a wall in maintaining food security and in the process, national security as well.**

** The allied sectors like horticulture, animal husbandry, dairy and fisheries, have an important role in improving the overall economic conditions and health and nutrition of the rural masses.**

** To maintain the ecological balance, there is need for sustainable and balanced development of agriculture and allied sectors.**

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()
```

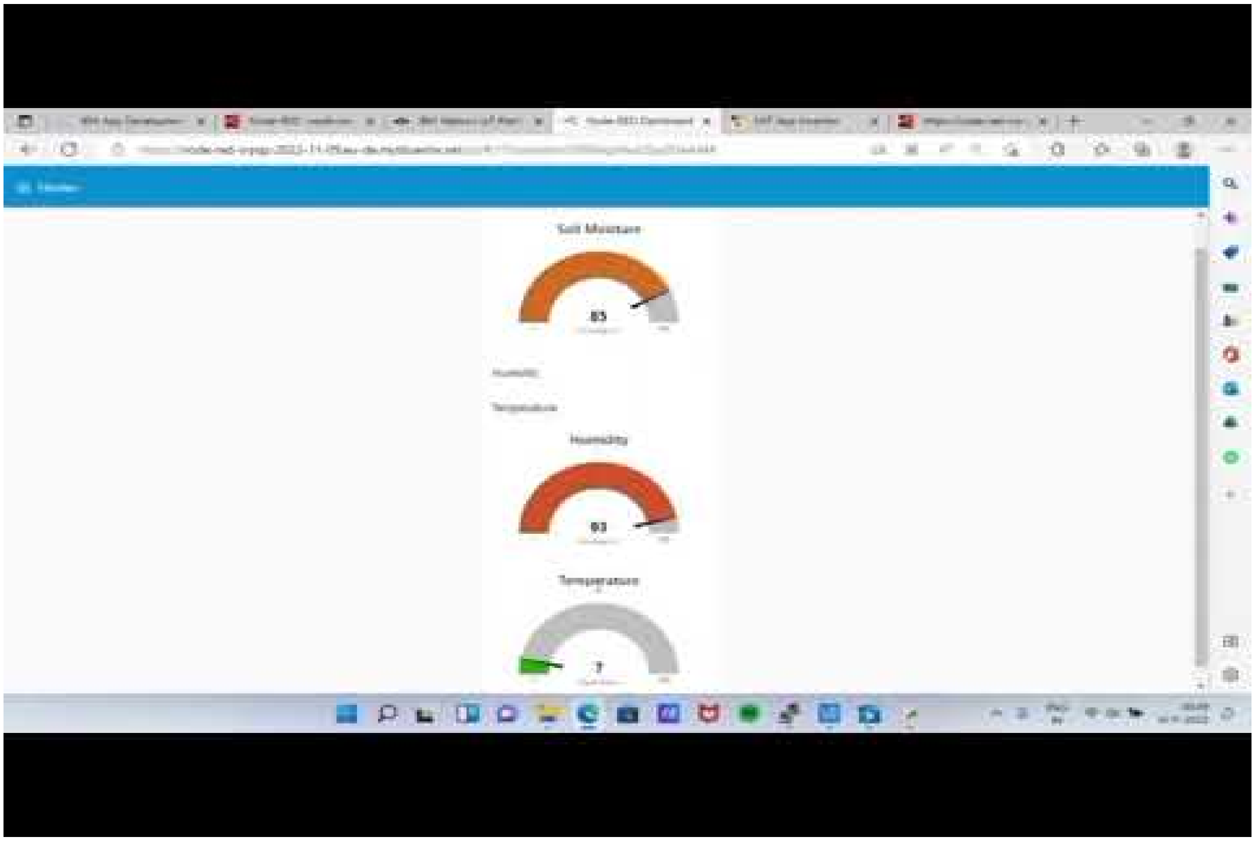
NODE-RED FLOW:

```
               [
 {
 "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":"",
 "className":"",
 "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":"",
 "className":"",
 "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":"",
"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":"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
}
```

GITHUB &PROJECT DEMO LINK:

LINK:

https://youtu.be/G4xxylgLeVU


Home
Soil Moisture
85
Humidity
Temperature
Humidity
83
Temperature
7