

IOT BASED SMART CROP PROTECTION SYSTEM FOR AGRICULTURE

Team ID : PNT2022TMID41353

Team Members

M.Nilofar Nisha

Z.Jafrin

R.Priya

N.Sneka

INTRODUCTION

PROJECT OVERVIEW:

Crops in farms are many times ravaged by local animals like buffaloes, cows, goats, birds etc. this leads to huge losses for the farmers. It is not possible for farmers to barricade entire fields or stay on field 24 hours and guard it. so here we propose automatic 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 safety of 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 economy of many countries dependent upon the agriculture. In spite of economic development agriculture is the backbone of the economy. Crops in farms 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 entire fields or stay 24 hours and guard it. Agriculture meets food requirements of the people and produces several raw materials for 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:

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-8 people recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

60 estimators

- A. **Team gathering**
Define who should participate in the session and send out the Share-Important-Information-as-you-work ahead.
 - B. **Get the goal**
Think about the problem you'll be focusing on solving in the team-solving session.
 - C. **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.
- Open article*

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.

4 minutes

Brainstorm

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

10 minutes



Group Ideas

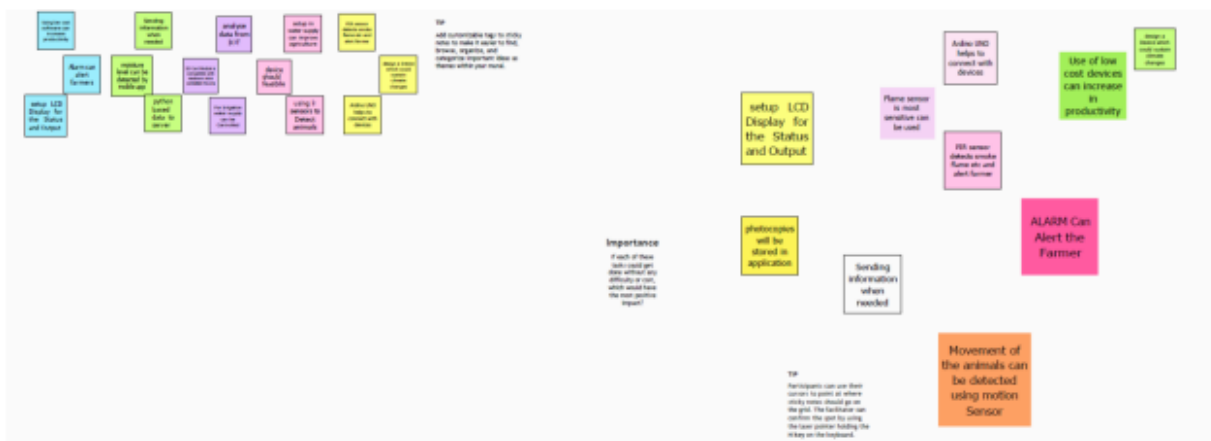
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes

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.

10 minutes



PROPOSED SOLUTION:

S.NO	PARAMETERS	DESCRIPTION
1.	Problem Statement (Problem to be solved)	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 benefit the farmers in many different ways
2.	Idea / Solution description	Smart Farming has enabled farmers to reduce waste and enhance productivity with the help of sensors (light, humidity, temperature, soil moisture, etc.).
3.	Novelty / Uniqueness	Role of SENSORS : IOT smart agriculture products are designed to help monitor crops and fields using sensors and by automating irrigation systems. As a result, farmers and associated brands can easily monitor the field conditions from anywhere without any hassle.
4.	Social Impact / Customer Satisfaction	Water conservation . Saves lot of time . Increased quality of production. Real time data and production insight. Remote monitoring.
5.	Scalability of the Solution	Scalability in smart farming refers to the and ability of a

		system to increase the capacity , the numbers.
--	--	--

PROBLEM SOLUTION FIT:

Define CS, fit into CL	1. CUSTOMER SEGMENT(S) CS Farmer's ! Who's not near his field	6. CUSTOMER LIMITATIONS <small>EG. BUDGET, DEVICES</small> CL 1)High adoption costs , security concerns. 2)Not aware of the implementation of IoT in agriculture.	5. AVAILABLE SOLUTIONS <small>PLUSES & MINUSES</small> AS Monitor different parameters and mobile or web application make easily to farm the crop field .	Explore AS, differentiate
	2. PROBLEMS / PAINS <small>+ ITS FREQUENCY</small> PR It's difficult to monitor and control Ain't known if the application doesn't work properly.	9. PROBLEM ROOT / CAUSE RC 1)If temperature ,PH level ,humidity & light intensity makes the serious cause for the environment. 2)Farmer affected by less productivity which will affect in their profit.	7. BEHAVIOR <small>+ ITS INTENSITY</small> BE Direct related: Tries to find a solution to prevent this problem Indirect related: Located in rural where internet connectivity might not be strong enough to facilitate fast transmission speeds.	
Identify strong TR & EM	3. TRIGGERS TO ACT TR Create opportunities to lift people out of poverty in developing nations. (Over 60%)	10. YOUR SOLUTION SL <i>"IoT based Smart crop protection system for agriculture" !!</i> It help farmers grow more food on less land by protection crops from pests, diseases and weeds as well as raising productivity per hectare.	8. CHANNELS of BEHAVIOR CH ONLINE: The Data send through application for the farmers to know about the farms. OFFLINE: The control action is taken by the farmers to monitor the farms.	Extract online & offline CH of BE
	4. EMOTIONS <small>BEFORE / AFTER</small> EM BEFORE: Finances, Heavy work overload and conflict in relationship. AFTER: It will easier to make more yield in field			

REQUIREMENT ANALYSIS

FUNCTIONAL REQUIREMENT:

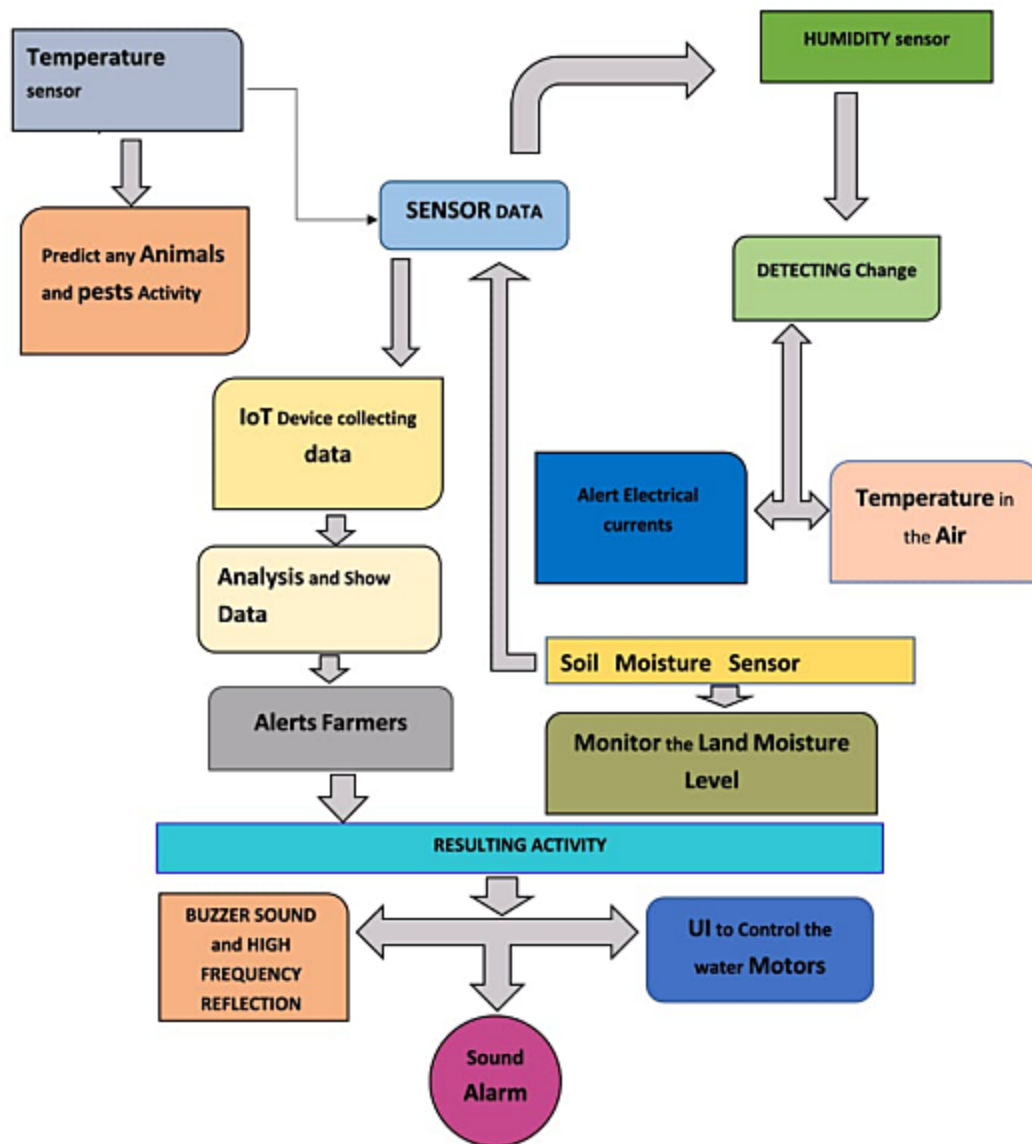
FR NO	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
1.	User Registration	Install the app Signing up with Gmail or phone number Creating a profile Understand the guidelines
2.	User Confirmation	Email or phone number verification required via OTP
3.	Accessing datasets	Data's are obtained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

NON FUNCTIONAL REQUIREMENT:

FR No.	Non-Functional Requirement	Description
1.	Usability	This project's contributes 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 benefits 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 agaioست 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 effcieot to retrieve photos at scale enhanching and scalability.

PROJECT DESIGN

DATA FLOW DIAGRAM:



SOLUTION AND TECHNICAL ARCHITECTURE:

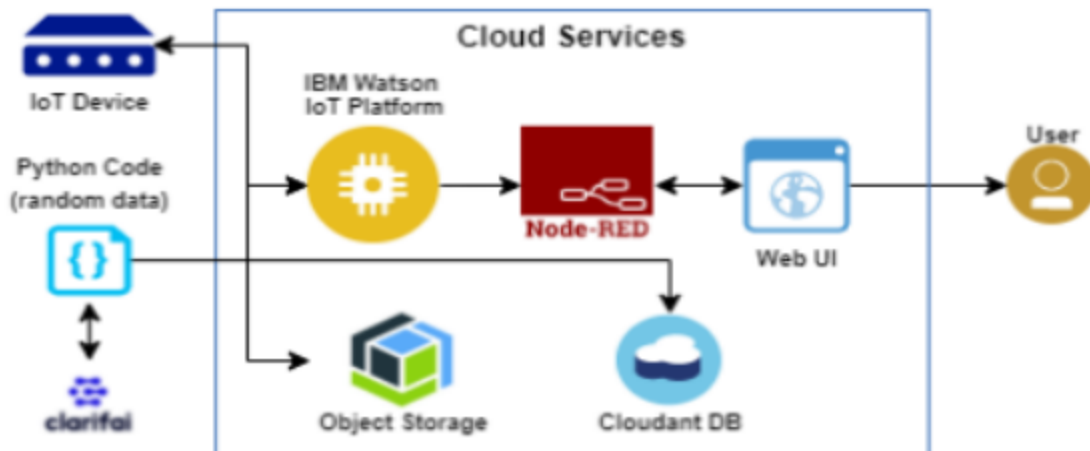


Table-1: Components & Technologies:

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

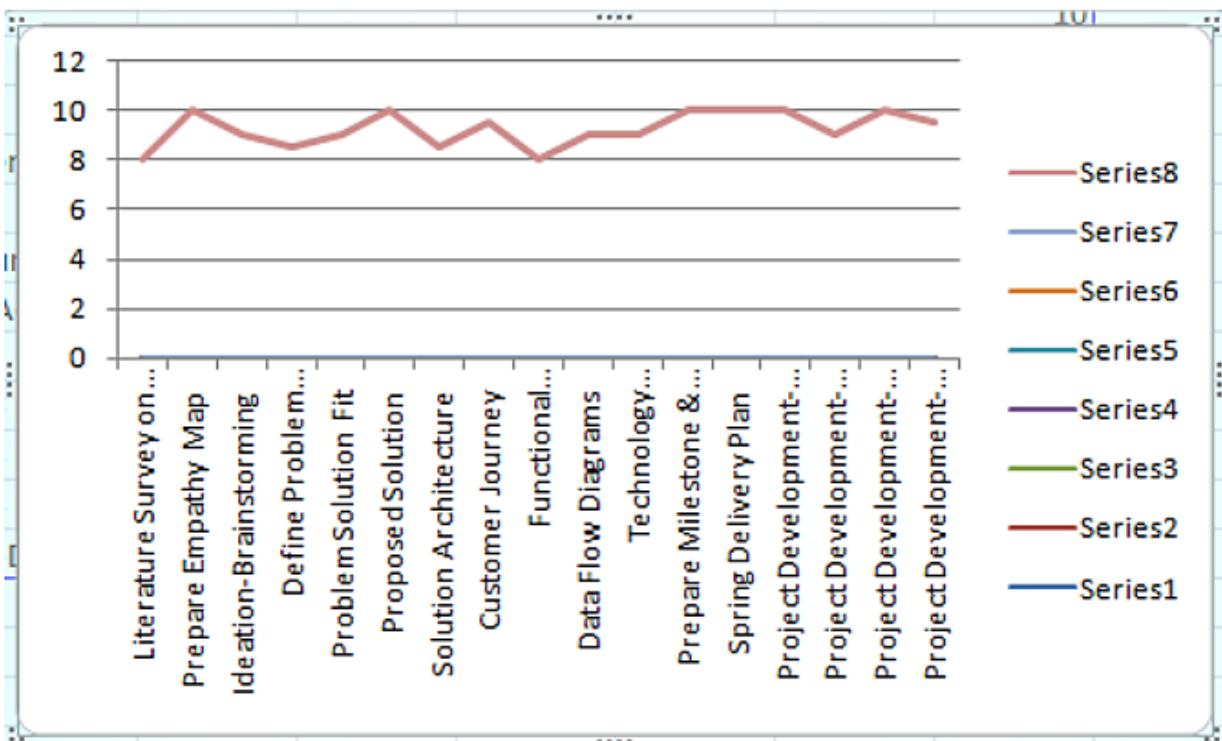
SPRINT PLANNING AND ESTIMATION:

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)

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



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 IoT") time.sleep(10)
deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
```

deviceCli.disconnect()

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A table displays sensor data with columns for Event, Value, Format, and Last Received. The table contains four rows of IoT sensor data. Below the table, it indicates 'Items per page 50' and '1-1 of 1 item'. A status bar at the bottom shows '1 Simulation running'. The browser's address bar shows the URL 'chr2pv.internetofthings.ibmcloud.com/dashboard/devices/browse'.

Event	Value	Format	Last Received
IoTSensor	{"temp":52,"Humid":90}	json	a few seconds ago
IoTSensor	{"temp":35,"Humid":76}	json	a few seconds ago
IoTSensor	{"temp":90,"Humid":3}	json	a few seconds ago
IoTSensor	{"temp":4,"Humid":59}	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.

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

FEATURE-2:


- i. Good sensitivity to Combustible gas in wide range .
- ii. High sensitivity to LPG, Propane and Hydrogen .
- iii. Long life and low cost.
- iv. Simple drive 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:



HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS


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
node-v18.12.1-x64.msi


macOS Installer
node-v18.12.1.pkg


Source Code
node-v18.12.1.tar.gz

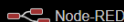
Windows Installer (.msi)
Windows Binary (.zip)
macOS Installer (.pkg)
macOS Binary (.tar.gz)
Linux Binaries (x64)

32-bit	64-bit
32-bit	64-bit
64-bit / ARM64	
64-bit	ARM64
64-bit	

(no subject) - nish... | BHARATHIYAR IN... | IBM... | IoT-B7-1A3E (Eve... | Service Details - | IBM Watson IoT... | Node-RED : nodi... | + | - | Close

node-red-himlg-2022-11-17.au-syd.mybluemix.net/red/#flow/fc6ce66c936f4b6b

Gmail | Maps | YouTube | Convert PDF to RA... | Inbox (428) - nishas...

 Node-RED

Deploy

filter nodes

Flow 1

common

inject

debug

complete

catch

status

link in

link call

link out

comment

function

function

IBM IoT

connected

debug 1

debug

all nodes

all

11/18/2022, 2:27:50 PM node: debug 1

iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json :

msg.payload : Object

> { randomNumber: 9, temp: 99, hum: 96 }

11/18/2022, 2:28:13 PM node: debug 1

iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json :

msg.payload : Object

> { randomNumber: 18, temp: 102, hum: 74 }

11/18/2022, 2:29:59 PM node: debug 1

iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json :

msg.payload : Object

> { randomNumber: 8, temp: 108, hum: 81 }

11/18/2022, 2:30:16 PM node: debug 1

iot-2/type/NodeMCU/id/12345/evt/eventflow/fmt/json :

msg.payload : Object

> { randomNumber: 31, temp: 105, hum: 99 }

Type here to search | 27°C Mostly sunny | 2:30 PM 11/18/2022

node-red-himlg-2022-11-17.au-syd.mybluemix.net/red/#flow/fc6ce66c936f4b6b

Node-RED

gauge

Flow 1

debug

gauge

IBM IoT

debug 1

gauge

randomNumber: 31, temp: 105, hum: 99

11/18/2022, 2:31:51 PM node: debug 1

iot-2/type/NodeMCUId/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/NodeMCUId/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/NodeMCUId/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/NodeMCUId/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 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. This will also help them in achieving better crop yields thus leading to their economic 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 chance of not starving.

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 the environment 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 animal and fire can be detected by cameras and if it comes towards farm then 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 be activated.

APPENDIX

```
import time
import sys
import ibmiotf.application
# toinstallpip install ibmiotf
import ibmiotf.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 Callback if cmd.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 required information: 'interval'")
interval = cmd.data['interval']
elif cmd.command == "print":
if 'message' not in cmd.data:
print("Error - command is missing required information: ")
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 of type
"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 sys
import ibmiotf.application
import ibmiotf.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 auth token
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 of type
    "greeting" 10 times
    deviceCli.connect()
    while True:
        temp=random.randint(0,100)
        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("IoT Sensor", "json", data, qos=0,
            on_publish=myOnPublishCallback)
            if not success:
                print("Not connected to IoT")
                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>