

**GOVERNMENT COLLEGE OF ENGINEERING  
CHETTIKARAI, DHARMAPURI**



**IOT BASED SMART CROP PROTECTION SYSTEM FOR  
AGRICULTURE**

**IBM NALAIYATHIRAN**

**Document Title**

<b>TITLE</b>	IoT Based Smart Crop Protection System for Agriculture
<b>DOMAIN NAME</b>	INTERNET OF THINGS
<b>TEAM ID</b>	PNT2022TMID41289
<b>TEAM LEADER NAME</b>	MOUNISHA C N
<b>TEAM MEMBER NAME</b>	DEEPIKA R KAVIPRIYA K KALPANADEV I R
<b>MENTOR NAME</b>	Dr. THIYAGARAJAN R

# CONTENT

## 1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

## 2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

## 3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

## 4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

## 5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

## 6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

## 7. CODING & SOLUTIONING

7.1 Feature 1

7.2 Feature 2

7.3 Feature 3

## 8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

## 9. RESULTS

## 10. ADVANTAGES & DISADVANTAGES

## 11. CONCLUSION

## 12. FUTURE SCOPE

## 13. APPENDIX

Source Code

GitHub & Project Demo Link

## **1. INTRODUCTION**

### **1.1 PROJECT OVERVIEW**

An intelligent crop protection system helps the farmers in protecting the crop from animals and birds which destroy the crop. This system also helps farmers to monitor the soil moisture levels in the field and also the temperature and humidity values near the field. The motors and sprinklers in the field can be controlled using the mobile application.

### **1.2 PURPOSE**

The purpose of the project is to increase the productivity by protecting the crops from animal and bird attack. This system also include remote monitoring and control of pump to avoid the farmer to visit the farm in nighttime. It also helps to monitor the real-time measures of soil moisture, humidity and temperature.

## **2. LITERATURE SURVEY**

### **2.1 EXISTING PROBLEM**

#### **PAPER 1: IOT BASED CROP PROTECTION SYSTEM AGAINST BIRDS AND ANIMAL ATTACKS**

The main aim of our project is to protect the crops from damage caused by animal as well as divert the animal without any harm. 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. Animal detection system is designed to detect the presence of animal and offer a warning. In this project we used PIR and ultrasonic sensors to detect the movement of the animal and send signal to the controller. It diverts the animal by producing sound and signal further, this signal is transmitted to GSM and which gives an alert to farmers and forest department immediately.

#### **PAPER 2: IOT SOLUTIONS FOR CROP PROTECTION AGAINST WILD ANIMAL ATTACKS**

Technology plays a central role in our everyday life. There has been a surge in the demand of Internet of Things (IoT) in many sectors, which has drawn significant research attention from both the academia and the industry. In the agriculture sector alone, the deployment of IoT has led to smart farming, precision agriculture, just to mention a few. This paper presents the development of Internet of Things application for crop protection to prevent animal intrusions in the crop field. A repelling and a monitoring system is provided to prevent potential damages in Agriculture, both from wild animal attacks and weather conditions.

### PAPER 3: SMART CROP PROTECTION SYSTEM FROM WILD ANIMALS USING IOT

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

### PAPER 4: IOT BASED MONITORING SYSTEM IN SMART AGRICULTURE

Internet of Things (IoT) plays a crucial role in smart agriculture. Smart farming is an emerging concept, because IoT sensors capable of providing information about their agriculture fields. The paper aims making use of evolving technology i.e. IoT and smart agriculture using automation. Monitoring environmental factors is the major factor to improve the yield of the efficient crops. The feature of this paper includes monitoring temperature and humidity in agricultural field through sensors using CC3200 single chip. Camera is interfaced with CC3200 to capture images and send that pictures through MMS to farmers mobile using Wi-Fi.

### PAPER 5: IOT BASED SMART AGRICULTURE

Agriculture plays vital role in the development of agricultural country. In India about 70% of population depends upon farming and one third of the nation's capital comes from farming. Issues concerning agriculture have been always hindering the development of the country. The only solution to this problem is smart agriculture by modernizing the current traditional methods of agriculture. Hence the project aims at making agriculture smart using automation and IoT technologies. The highlighting features of this project includes smart GPS based remote controlled robot to perform tasks like weeding, spraying, moisture sensing, bird and animal scaring, keeping vigilance, etc. Secondly it includes smart irrigation with smart control and intelligent decision making based on accurate real time field data. Thirdly, smart warehouse management which includes temperature maintenance, humidity maintenance and theft detection in the warehouse. Controlling of all these operations will be through any remote smart device or computer connected to Internet and the operations will be performed by interfacing sensors, Wi-Fi or ZigBee modules, camera and actuators with micro-controller and raspberry pi.

## 2.2 REFERENCE

- PAPER 1: P Navaneetha, R Ramiya Devi, S Vennila, P Manikandan, Dr S Saravanan  
International Journal of Innovative Research In Technology 6 (11), 2020.
- PAPER 2: Stefano Giordano, Ilias Seitanidis, Mike Ojo, Davide Adami, Fabio Vignoli  
2018 IEEE international conference on Environmental Engineering (EE), 1-5, 2018.
- PAPER 3: Priyanka Deotale, Prasad Lokulwar  
2021 International Conference on Computational Intelligence and Computing Applications (ICCICA) 1-4,2021.
- PAPER 4: SR Prathibha, Anupama Hongal, MP Jyothi  
2017 international conference on recent advances in electronics and communication technology (ICRAECT), 81-84, 2017
- PAPER 5: Nikesh Gondchawar, RS Kawitkar  
International Journal of advanced research in Computer and Communication Engineering 5 (6), 838-842, 2016

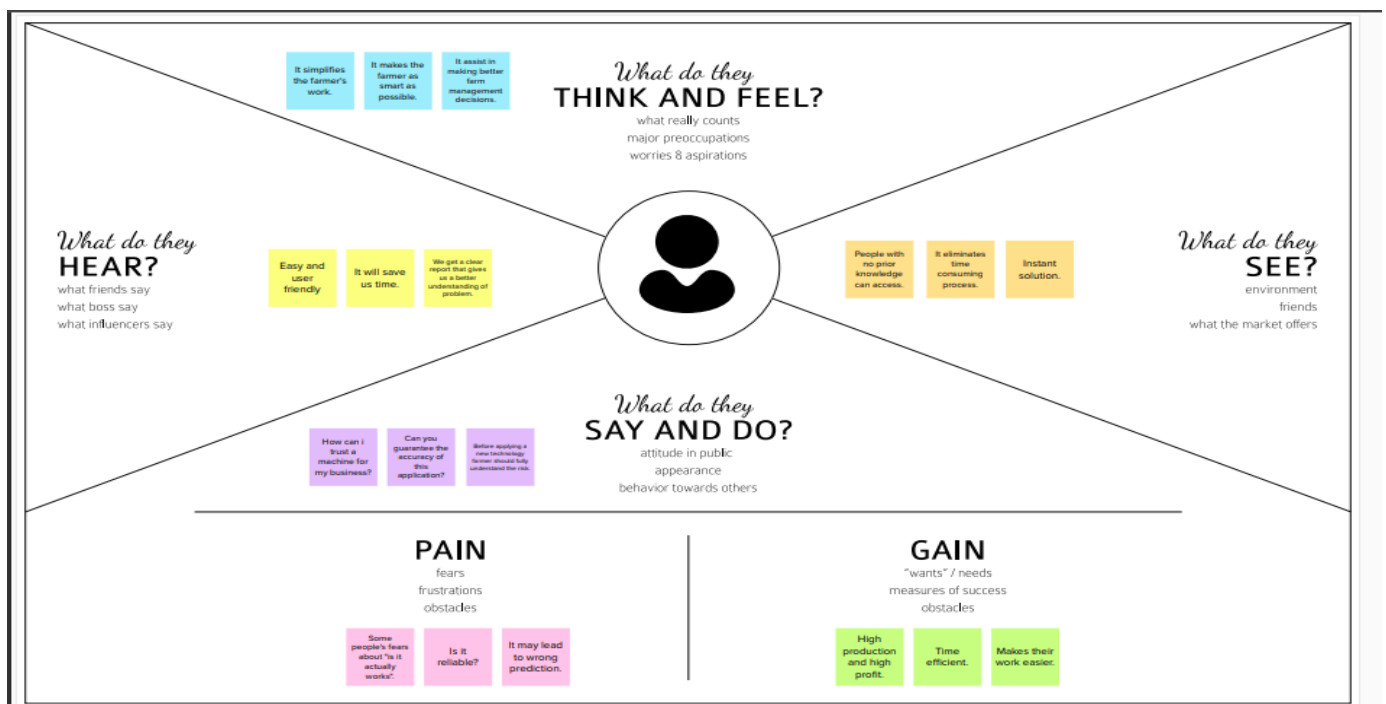
## 2.3 PROBLEM STATEMENT DEFINITION

Crops in the farms are many times devastated by the wild as well as domestic animals and it also leads to low productivity of crops. It is also difficult for farmers to visit the fields at night time to control motors and pump.

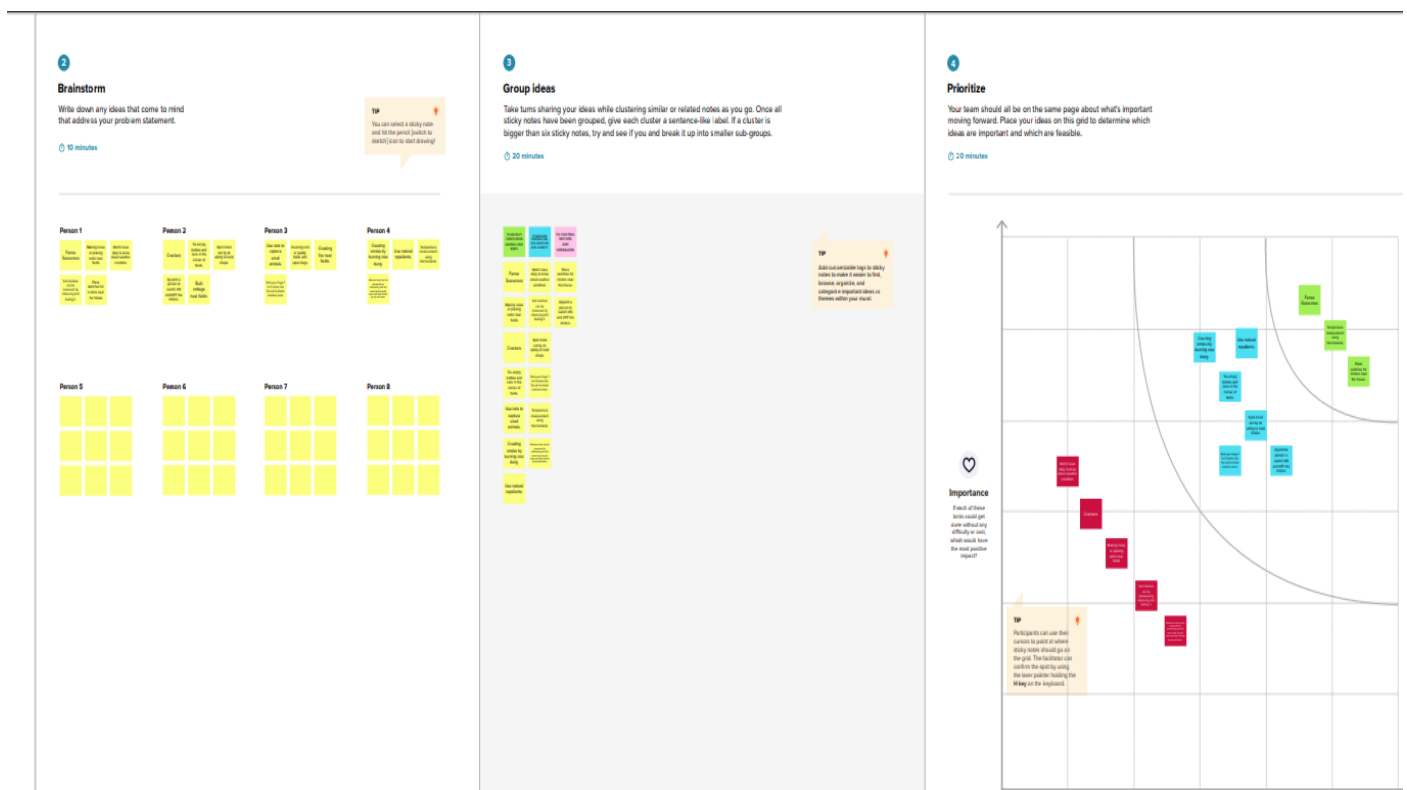
Question	Description
Who does the problem affect?	It affects the farmers and the public.
What are the boundaries of the problem?	The boundaries are agricultural fields.
What is the issue?	It causes low productivity & financial loss. If it is fixed, there will be no shortage for food and it also save lives of farmer.
When does the issue occur?	It occurs when there is no one in the field.
Where does the issue occur?	It occurs in the agricultural fields.
Why it is important that we fix the problem?	To improve productivity thereby to reduce poverty.

### 3. IDEATION & PROPOSED SOLUTION

#### 3.1 EMPATHY MAP CANVAS



#### 3.2 IDEATION & BRAINSTORMING



### 3.3 PROPOSED SOLUTION

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	Crop devastation due to animals and birds should be controlled. Difficulty of farmers in visiting fields at nighttime to control motors should be solved.
2.	Idea / Solution description	An IoT based intelligent system to protect crops from animals by sensing them. It also helps to measure and monitor the temperature, soil moisture and humidity level in the soil. It also enables remote monitoring and control of motors using mobile application.
3.	Novelty / Uniqueness	Protecting crops from animals and birds. Remote monitoring and control of motors and sprinklers.
4.	Social Impact / Customer Satisfaction	Improve the productivity, Save lives of farmers.
5.	Business Model (Revenue Model)	Community based solution by FAO's Solution through contract farming
6.	Scalability of the Solution	It can be practically established in all kinds of fields.

### 3.4 PROBLEM SOLUTION FIT

Define CS, fit into CC Focus on J&P, understand RC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> Farmers are my customers.	<b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span> Budget Lack of awareness	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> Electric Fence Scarecrow.	Explore AS, differentiate
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span> Devastation of crops by animals and birds.	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> Lack of proper maintenance.	<b>7. BEHAVIOUR</b> <span>BE</span> Built a fence around the fields to prevent animals from entering it	
Identify strong TR & EM	<b>3. TRIGGERS</b> <span>TR</span> Seeing their neighbour installing and by hearing their feedback.	<b>10. YOUR SOLUTION</b> <span>SL</span> I would install sensors to sense animals and prevent crops before they destroy it. I would buy an system to remote monitor and control the motors using mobile application.	<b>8.CHANNELS of BEHAVIOUR</b> <span>CH</span> <b>1.ONLINE</b> Search for any product to overcome their problems. <b>2. OFFLINE</b> Install fence Appoint person to guard the fields	
	<b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span> Lost Insecure Depressed			



## 4. REQUIREMENT ANALYSIS

### 4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Requirements	Crop Protection Monitors Soil Moisture, Temperature and Humidity Automatic Sprinkler System
FR-4	Payment Options	Cash on Delivery Net Banking/UPI Credit/Debit/ATM card
FR-5	Product Delivery and Installation	Door Step delivery Marketing, Product Camp Free Installation and One-year Warranty

### 4.2 NON-FUNCTIONAL REQUIREMENT

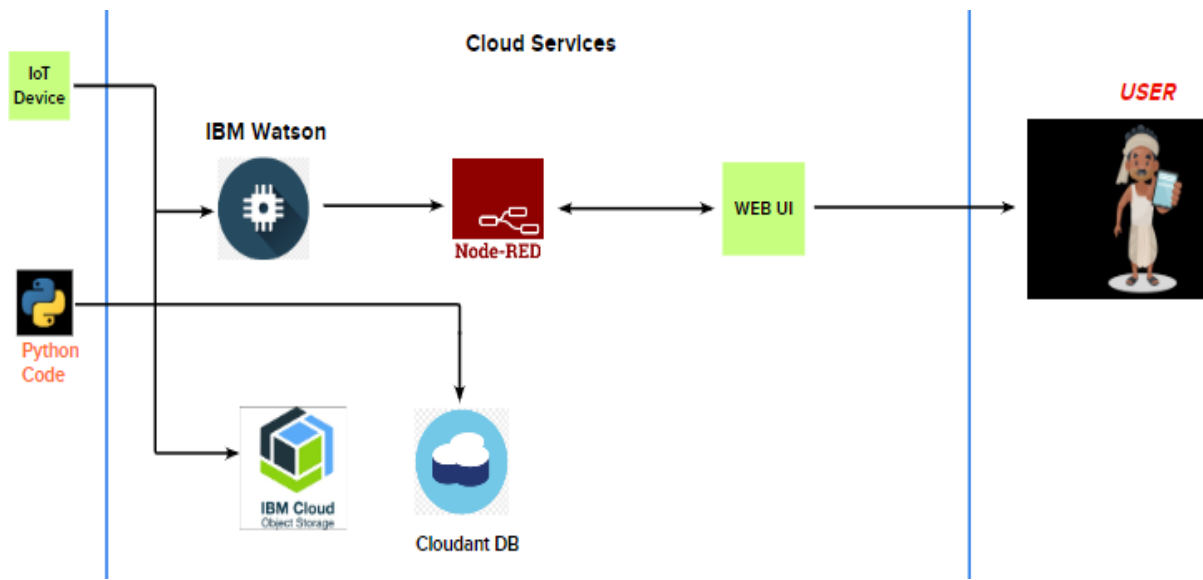
NFR-1	<b>Usability</b>	Easier to use Even an illiterate farmer has to use the product without any difficulties
NFR-2	<b>Security</b>	The system can be accessed by authorized the users need
NFR-3	<b>Reliability</b>	Monitoring the temperature parameter and providing clear solution Immediate alert is provided in case of any system failure
NFR-4	<b>Performance</b>	The performance should be very effective and efficiency
NFR-5	<b>Availability</b>	All the features will be available when the user requires
NFR-6	<b>Scalability</b>	The product has to cover all the space of the land The system should have upgradable feature

## 5. PROJECT DESIGN

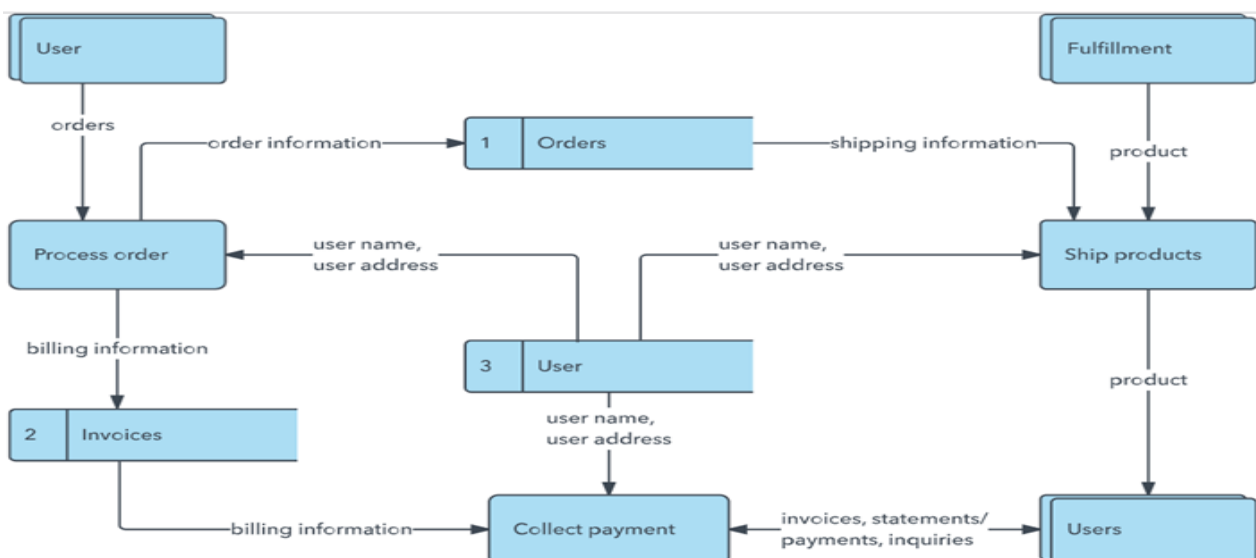
### 5.1 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

FLOW :

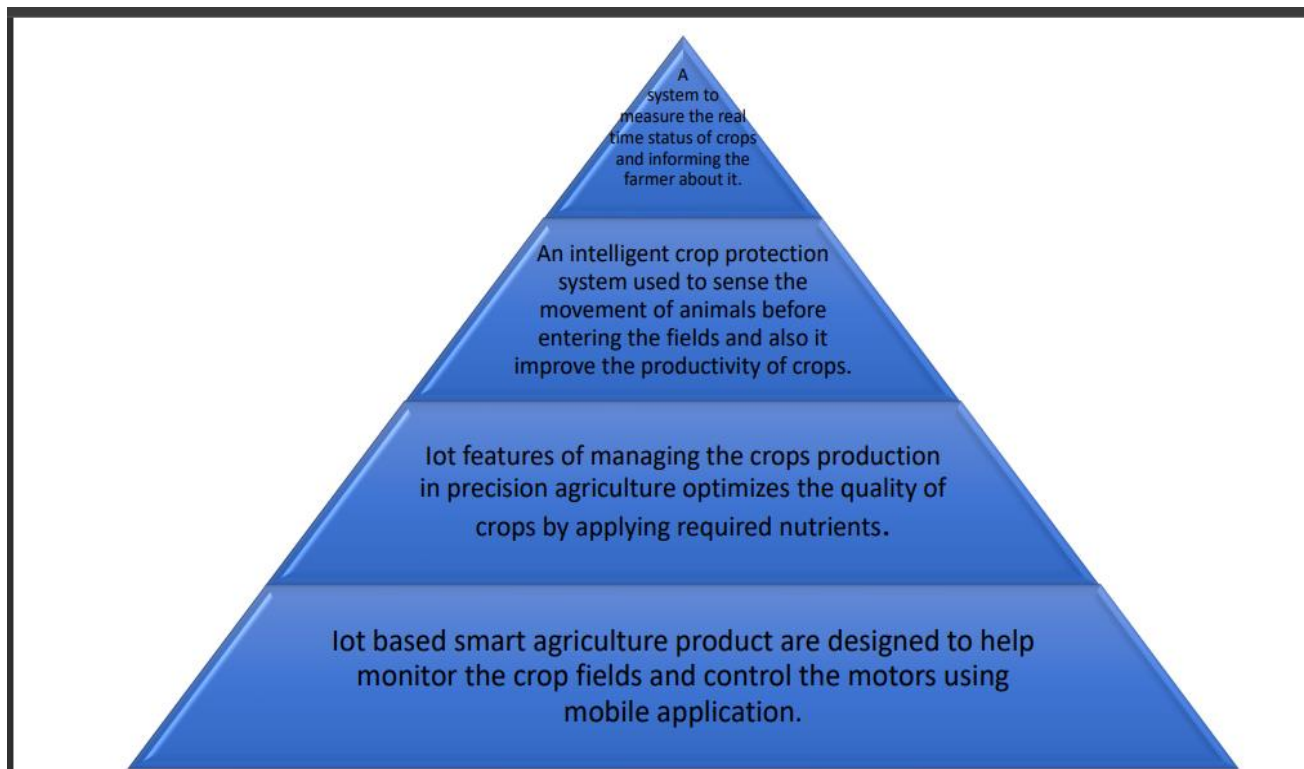


Example: DFD Level 0 (Industry Standard)

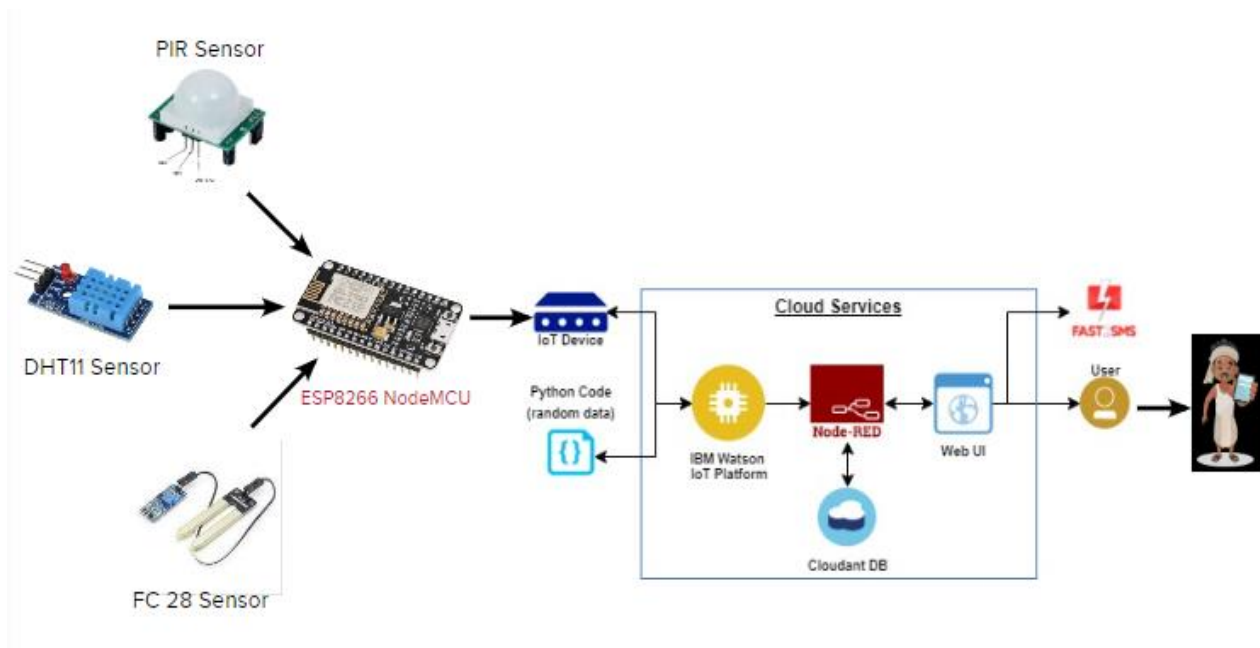


## 5.2 SOLUTION & TECHNICAL ARCHITECTURE

### Solution Architecture:



### Technical Architecture:



## 5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account/dashboard	High	Sprint-1
	Mail Confirmation	USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Facebook Access	USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
	Register	USN-4	As a user, I can register for the application through Gmail	I can register for the application	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can log into the required application	High	Sprint-1
Customer (Web user)	Same as Mobile user	Same as Mobile user	Same as Mobile user	Same as Mobile user	High	Sprint-1
Customer Care Executive	Farmer Welfare Department	USN-1	As a user, I managing a team of representatives offering customer support	I can communicate to them in proper manner	High	Sprint-1
	Agriculture Extension Department	USN-2	As a user, I provide technical aid to farmers on any agriculture issues	I can implementation of agriculture extension activities	High	Sprint-1
Administrator	Farm Administrator	USN-1	As a user, I provide administrative support for farmers	I informed about the financial and physical performance	High	Sprint-1
		USN-2	As a user, I live and work on the farm or an estate	I take responsibility	Medium	Sprint-1

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Mounisha
Sprint-1	confirmation	USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Deepika
Sprint-2	confirmation via Facebook	USN-3	As a user, I can register for the application through Facebook	2	Low	Kalpanadevi
Sprint-1	confirmation via Email	USN-4	As a user, I can register for the application through Gmail	2	Medium	Kavipriya
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Kavipriya

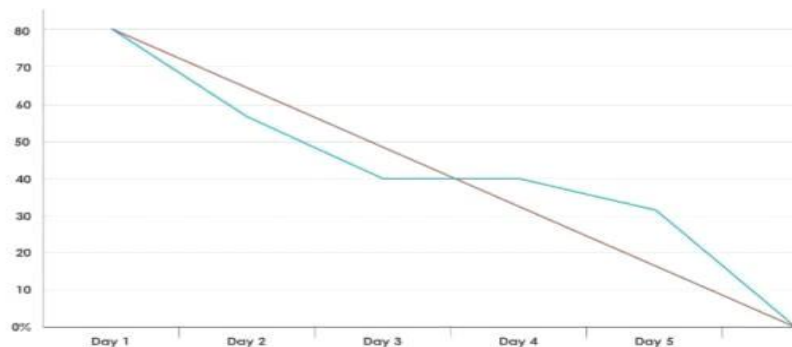
## 6.2 SPRINT DELIEVERY SCHEDULE

S.NO	ACTIVITY TITLE	DESCRIPTION	DURATION
1	Understanding the project	Assign the team members after that create repository in the GitHub and then assign task to each member and guide them how to access the GitHub while submitting the assignments	1 week
2	Starng The Project	Team Members to Assign All the Tasks Based on Sprints and Work on It Accordingly.	1 week
3	Completing Every Task	Team Leader should ensure that whether every team member have completed the assigned task or not	1 week
4	Stand Up Meetings	Team Lead Must Have a Stand-Up Meeting with The Team and Work on The Updates and Requirement Session	1 week
5	Deadline	Ensure that team members are completing every task within the deadline	1 week

VELOCITY:

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

**Burndown Chart:**



## 7. CODING & SOLUTION SOLUTIONING

### 7.1 FEATURE 1

In this code all the services (cloud object storage, cloudant database, clarifai and node-red) used for the project are connected to the IBM Watson cloud using their api keys.

```
Crop_protection.py - C:\Users\mouni\OneDrive\Desktop\python_code\Crop_protection.py (3.7.4)
File Edit Format Run Options Window Help

import cv2
import numpy as np
import wiotp.sdk.device
import playsound
import random
import time
import datetime
import ibm_boto3
from ibm_botocore.client import Config, ClientError
#cloudantDB
from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel
from clarifai_grpc.grpc.api import service_pb2_grpc
stub = service_pb2_grpc.V2Stub(ClarifaiChannel.get_grpc_channel())
from clarifai_grpc.grpc.api import service_pb2, resources_pb2
from clarifai_grpc.grpc.api.status import status_code_pb2
#This is how you authenticate.
metadata = (('authorization', 'Key 84a796d4dedb4dccc8aa71810ab16a3fd')),
COS_ENDPOINT= "https://samplestoragebucket123.s3.jp-tok.cloud-object-storage.appdomain.cloud" #current list available at https://control.cloud-object-storage.cloud.ibm
COS_API_KEY_ID = "2BvtmkJHb5xtAIatYUEWhHnKG-NnbtVOgCfmyyaUSy4"
COS_AUTH_ENDPOINT="https://iam.cloud.ibm.com/identity/token"
COS_RESOURCE_CRN="crn:vl:bluemix:public:cloud-object-storage:global:a/a6a97499ed5743b7b1f42a2f32c439e7:9f6410d4-28bd-4964-a37d-06f665016fae::"
clientdb = Cloudant("apikey-v2-jruoln188tlw585vgvj72y3qex0p75efgz2vd27kzex", "a4c83e0fb38a78ed85faef63fed009", url="https://apikey-v2-jruoln188tlw585vgvj72y3qex0p75efgz
clientdb.connect()
# Create resource
cos = ibm_boto3.resource("s3",
    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_RESOURCE_CRN,
    ibm_auth_endpoint=COS_AUTH_ENDPOINT,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)
```

### 7.2 FEATURE 2

This code process the video file frame by frame and send it to the clarifai service to check whether it is an animal or not, if the response (label) from the clarifai service is same as the animal then “Animal detect = True” and an alert audio will play.


```
Crop_protection.py - C:\Users\mouni\OneDrive\Desktop\python_code\Crop_protection.py (3.7.4)
File Edit Format Run Options Window Help

my_database = clientdb.create_database(database_name)
if my_database.exists():
    print(f"({database_name})' successfully created.")
cap=cv2.VideoCapture('monkey.mp4')
if (cap.isOpened() == True):
    print('File opened')
else:
    print('File not found')

while (cap.isOpened()):
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    ims = cv2.resize(frame, (960, 540))
    cv2.imwrite('ex.jpg',ims)
    with open("ex.jpg", "rb") as f:
        file_bytes = f.read()
    # this is the model ID of a publicly available General model. You may use any other public or custom model ID.
    request = service_pb2.PostModelOutputsRequest(
        model_id='aaa03c23b3724a16a56b629203edc62c',
        inputs=[resources_pb2.Input(data=resources_pb2.Data(image=resources_pb2.Image(base64=file_bytes))
    ))
    response = stub.PostModelOutputs(request, metadata=metadata)
    if response.status_code != status_code_pb2.SUCCESS:
        raise Exception("Request failed, status code: " + str(response.status_code))
    detect= False
    for concept in response.outputs[0].data.concepts:
        #print("%12s. %2f" % (concept.name, concept.value))
        if (concept.value>0.98):
            #print(concept.name)
            if (concept.name=="animal"):
                print("Alert!Alert!Animal Detected")
                playsound.playsound('alert.mp3')
                picname=datetime.datetime.now().strftime("%Y-%m-%d-%H-%M")
                cv2.imwrite(picname+'.jpg',frame)
                multi_part_upload('samplestoragebucket123', picname+'.jpg', picname+'.jpg')
                json_document={"link":COS_ENDPOINT+'/'+'samplestoragebucket123'+'/'+'picname+'.jpg'}
                new_document = my_database.create_document(json_document)
                if new_document.exists():
                    print(f"Document successfully created.")
                time.sleep(5)
                detect=True
```

### 7.3 FEATURE 3

In this code the light ON and light OFF , motor ON and motor OFF commands are generated. Random values of moisture and humidity are also generated using this code.

 Crop\_protection.py - C:\Users\mouni\OneDrive\Desktop\python\_code\Crop\_protection.py (3.7.4)

File Edit Format Run Options Window Help

```
def myCommandCallback(cmd):
    print("Command received: %s" %cmd.data)
    command=cmd.data['command']
    print(command)
    if(command=='lighton'):
        print('lighton')
    elif(command=='lightoff'):
        print('lightoff')
    elif(command=='motoron'):
        print('motoron')
    elif(command=='motoroff'):
        print('motoroff')

moisture=random.randint(0,100)
humidity=random.randint(0,100)
myData={'Animal': detect,'moisture':moisture,'humidity':humidity}
print(myData)
if(humidity!=None):
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
    print("Publish Ok..")
client.commandCallback = myCommandCallback
cv2.imshow('frame',ims)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
client.disconnect()
cap.release()
cv2.destroyAllWindows()
```

## 8. TESTING

### 8.1 TEST CASES

SL.NO	INPUT	OUTPUT	RESULT
01.	Humidity:67 Soil moisture:32	Animal Detect: TRUE Motor ON	Passed
02.	Humidity:47 Soil moisture:50	Animal Detect: TRUE Motor OFF	Passed
03.	Humidity:87 Soil moisture:28	Animal Detect: FALSE Motor ON	Passed
04.	Humidity:64 Soil moisture:44	Animal Detect: TRUE Motor OFF	Passed
05.	Humidity:57 Soil moisture:62	Animal Detect: TRUE Motor ON	Passed
06.	Humidity:39 Soil moisture:52	Animal Detect: FALSE Motor OFF	Passed
07.	Humidity:11 Soil moisture:64	Animal Detect: TRUE Motor ON	Passed
08.	Humidity:20 Soil moisture:82	Animal Detect: FALSE Motor ON	Passed
09.	Humidity:34 Soil moisture:38	Animal Detect: TRUE Motor OFF	Passed
10.	Humidity:53 Soil moisture:66	Animal Detect: TRUE Motor ON	Passed
11.	Humidity:47 Soil moisture:33	Animal Detect: FALSE Motor OFF	Passed
12.	Humidity:55 Soil moisture:35	Animal Detect: TRUE Motor ON	Passed
13.	Humidity:63 Soil moisture:48	Animal Detect: TRUE Motor OFF	Passed
14.	Humidity:67 Soil moisture:32	Animal Detect: FALSE Motor ON	Passed



## 8.2 USER ACCEPTANCE

### 1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

### 2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	8	3	2	2	15
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	1	1
Won't Fix	0	4	2	1	7
Totals	22	12	12	25	71

### 3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	49	0	0	49
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

## 9. RESULT

### CODE:

```
Crop_protection.py - C:\Users\mouni\OneDrive\Desktop\python_code\Crop_protection.py (3.7.4)
File Edit Format Run Options Window Help

myConfig = {
    "identity": {
        "orgId": "gm2lj6",
        "typeId": "Smartcrop",
        "deviceId": "12345"
    },
    "auth": {
        "token": "12345678"
    }
}

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

database_name= "moderedtoyyw20221019"
my_database = clientdb.create_database(database_name)
if my_database.exists():
    print(f'{database_name} successfully created.')
cap=cv2.VideoCapture("garden.mp4")
if (cap.isOpened()==True):
    print('File opened')
else:
    print('File not found')

while (cap.isOpened()):
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    ims = cv2.resize(frame, (960, 540))
    cv2.imwrite('ex.jpg',ims)
    with open("ex.jpg", "rb") as f:
        file_bytes = f.read()
    # this is the model ID of a publicly available General model. You may use any other public or custom model ID.
    request = service_pb2.PostModelOutputsRequest(
        model_id='aaa03c23b3724a16a56b629203edc62c',
        inputs=[resources_pb2.Input (data=resources_pb2.Data (image=resources_pb2.Image (base64=file_bytes))
    ))
    response = stub.PostModelOutputs(request, metadata=metadata)
    if response.status_code != status_code_pb2.SUCCESS:
        raise Exception("Request failed, status code: " + str(response.status_code))
    detect= False
    for concept in response.outputs[0].data.concepts:
        #print("%12s. %.2f" % (concept.name, concept.value))

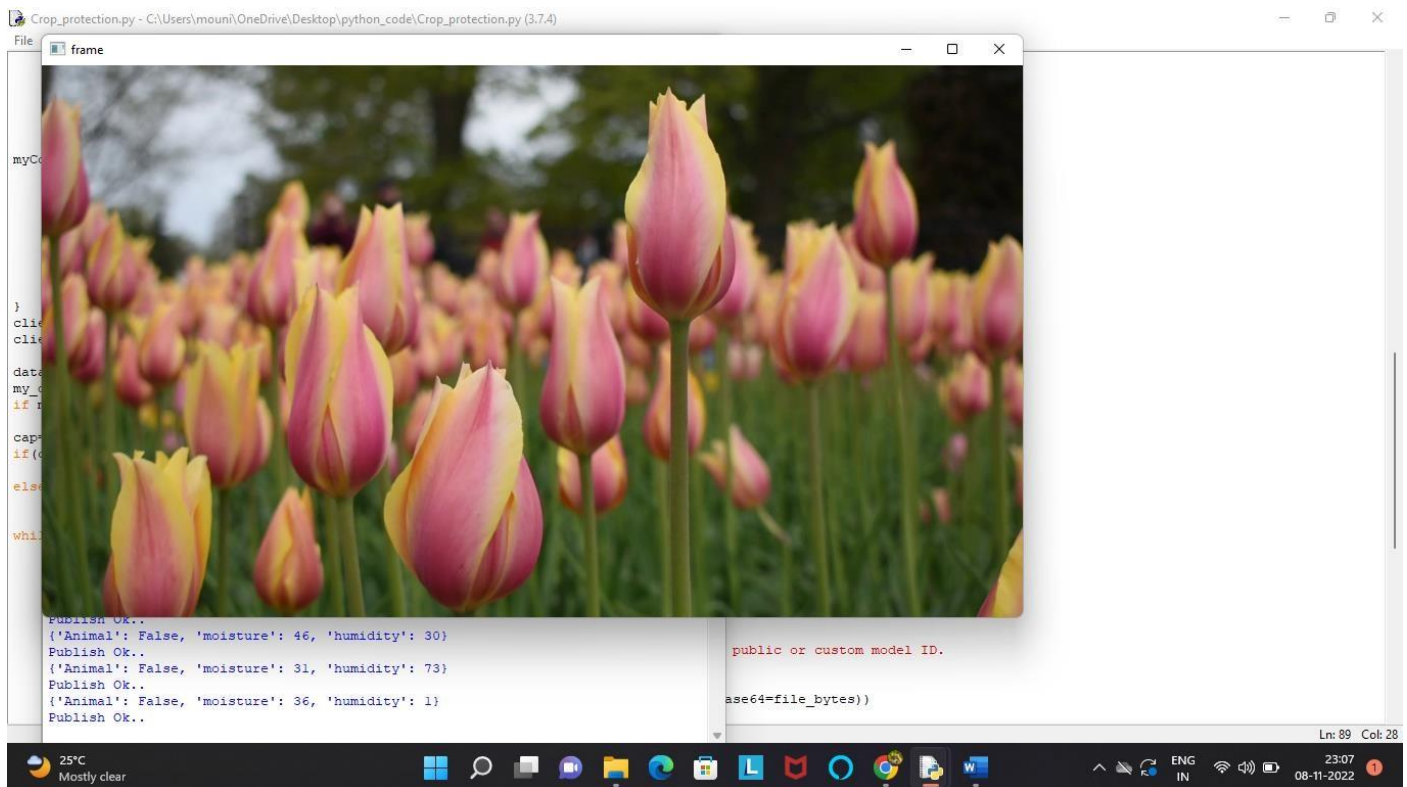
Ln: 58 Col: 71
```

### OUTPUT:

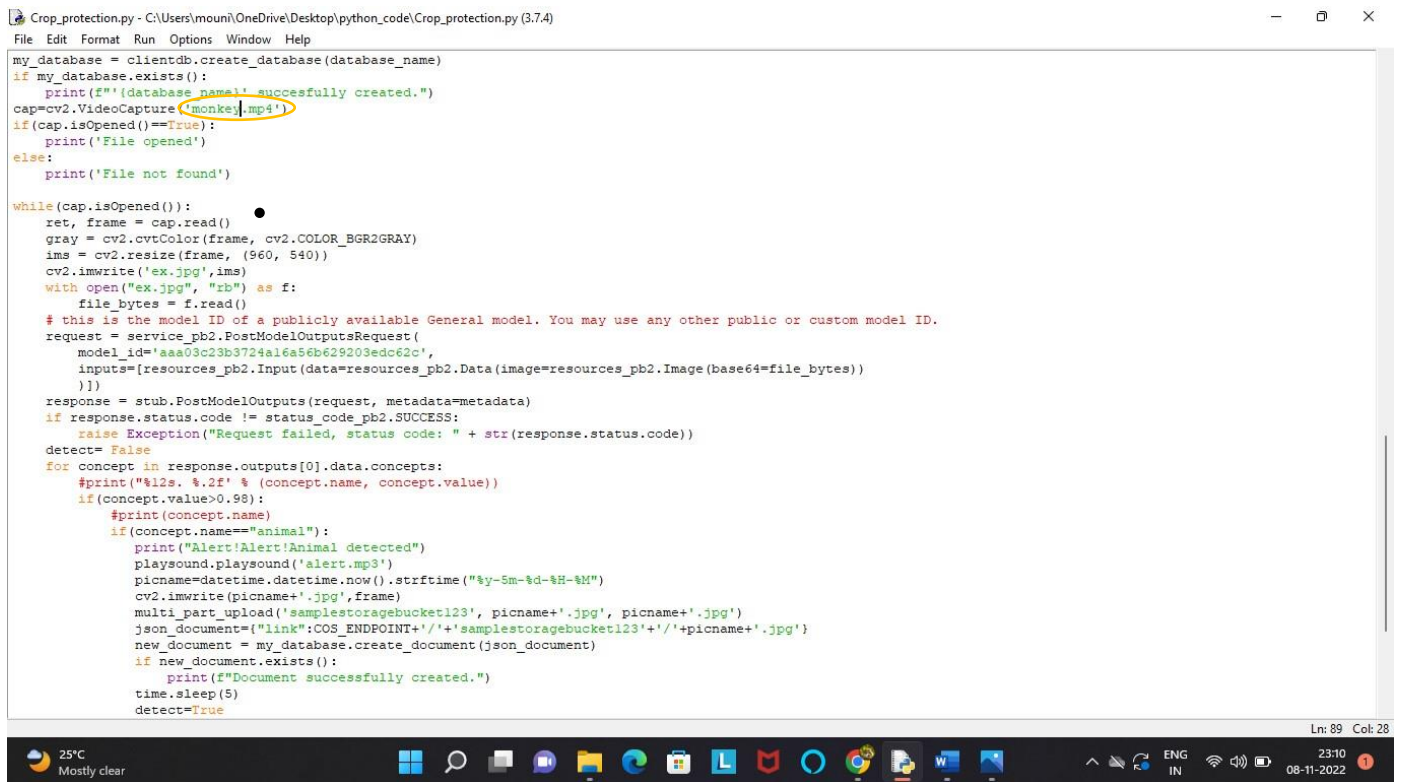
```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help

== RESTART: C:\Users\mouni\OneDrive\Desktop\python_code\Crop_protection.py ==
2022-11-08 21:38:04,024 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:gm2lj6:Smartcrop:12345
'moderedtoyyw20221019' succesfully created.
File opened
{'Animal': False, 'moisture': 23, 'humidity': 66}
Publish Ok..
{'Animal': False, 'moisture': 98, 'humidity': 17}
Publish Ok..
{'Animal': False, 'moisture': 6, 'humidity': 71}
Publish Ok..
{'Animal': False, 'moisture': 19, 'humidity': 37}
Publish Ok..
{'Animal': False, 'moisture': 66, 'humidity': 84}
Publish Ok..
{'Animal': False, 'moisture': 80, 'humidity': 15}
Publish Ok..
{'Animal': False, 'moisture': 21, 'humidity': 70}
Publish Ok..
{'Animal': False, 'moisture': 1, 'humidity': 38}
Publish Ok..
{'Animal': False, 'moisture': 56, 'humidity': 9}
Publish Ok..
{'Animal': False, 'moisture': 28, 'humidity': 82}
Publish Ok..
{'Animal': False, 'moisture': 100, 'humidity': 59}
Publish Ok..
{'Animal': False, 'moisture': 78, 'humidity': 79}
Publish Ok..
{'Animal': False, 'moisture': 91, 'humidity': 4}
Publish Ok..
{'Animal': False, 'moisture': 23, 'humidity': 67}
Publish Ok..
{'Animal': False, 'moisture': 72, 'humidity': 74}
Publish Ok..
{'Animal': False, 'moisture': 72, 'humidity': 4}
Publish Ok..
{'Animal': False, 'moisture': 71, 'humidity': 93}
Publish Ok..
{'Animal': False, 'moisture': 93, 'humidity': 100}
Publish Ok..

Ln: 5 Col: 0
```



## CODE:



## OUTPUT:

```
Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\mouni\OneDrive\Desktop\python_code\Crop_protection.py ==
2022-11-08 21:38:54,510 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:gm2lj6:Smartcrop:12345
'moderedtoyw20221019' succesfully created.
File opened
Alert!Alert!Animal detected

Error 263 for command:
close alert.mp3
The specified device is not open or is not recognized by MCI.
Failed to close the file: alert.mp3
Starting file transfer for 22-5m-08-21-39.jpg to bucket: samplestoragebucket123

CLIENT ERROR: An error occurred (AccessDenied) when calling the PutObject operation: Access Denied

Document successfully created.
{'Animal': True, 'moisture': 43, 'humidity': 87}
Publish Ok..
Alert!Alert!Animal detected

Error 265 for command:
open alert.mp3
The device name is already being used as an alias by this application. Use a unique alias.

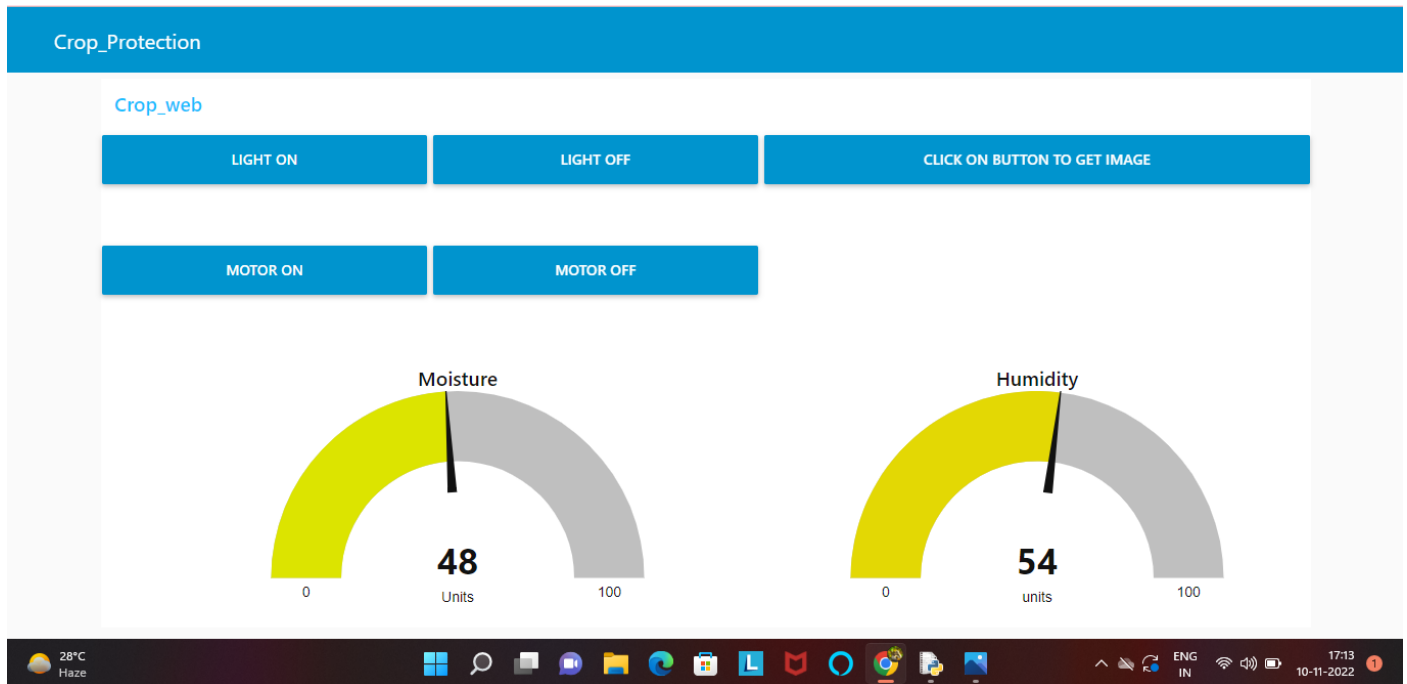
Error 263 for command:
close alert.mp3
The specified device is not open or is not recognized by MCI.
Failed to close the file: alert.mp3
Traceback (most recent call last):
  File "C:\Users\mouni\OneDrive\Desktop\python_code\Crop_protection.py", line 117, in <module>
    playsound.playsound('alert.mp3')
  File "C:\Users\mouni\AppData\Local\Programs\Python\Python37\lib\site-packages\playsound.py", line 72, in _playsoundWin
    winCommand(u'open {}'.format(sound))
  File "C:\Users\mouni\AppData\Local\Programs\Python\Python37\lib\site-packages\playsound.py", line 64, in winCommand
    raise PlaysoundException(exceptionMessage)
playsound.PlaysoundException:
Error 265 for command:
open alert.mp3
The device name is already being used as an alias by this application. Use a unique alias.
>>>
```

Ln: 42 Col: 4

```
Crop_protection.py - C:\Users\mouni\OneDrive\Desktop\python_code\Crop_protection.py (3.7.4)
File frame
myCo
}
cli
cli
dat
my_c
if r
cap
if (c
else
whi
File "C:\Users\mouni\AppData\Local\Programs\Python\Python37\lib\site-packages\
playsound.py", line 64, in winCommand
    raise PlaysoundException(exceptionMessage)
playsound.PlaysoundException:
Error 265 for command:
open alert.mp3
The device name is already being used as an alias by this application. Use
a unique alias.
blic or custom model ID.
34=file_bytes))
```

Ln: 89 Col: 28

## WEB UI:



## 10. ADVANTAGES & DISADVANTAGES

### ADVANTAGES:

- ❖ The advantages of the system are that it is effective at deterring animals and birds from crops, and is also environmentally friendly.
- ❖ It saves farmer's time by making easy remote monitor and control of motors and sprinklers in the field.
- ❖ It also gives the real-time measures of humidity and soil moisture level in the field.
- ❖ The system can be used to target a wide variety of animals, including cows, rats, mice, and birds.

### DISADVANTAGES:

- ❖ The system requires regular maintenance, and it may be difficult to keep up with the maintenance schedule.
- ❖ The system may not be effective against all animals and farmers may still need to visit fields to control some animals.
- ❖ The smart agriculture needs availability of internet continuously. Rural part of most of the developing countries do not fulfil this requirement. Moreover, internet connection is slower.



## **11. CONCLUSION**

The problem of crop vandalization by wild animals has become a major social problem in the current time. It requires urgent attention and an effective solution. Thus, this project carries a great social relevance as it aims to address this problem. The main objective of this project is to develop a smart crop protection system for animals and birds. The system should be able to automatically detect the presence of animals and birds in the vicinity of the crops and take necessary measures to protect the crops. The system should also be able to send alerts to the farmers in case of any potential threat to the crops.

## **12. FUTURE SCOPE**

This project is used to identify animals or illegal trespassers entering into agricultural fields. At present it is done manually it consumes more time and also in the coming future, we review the application of the animal detection technology in the agricultural field and it can promote for detecting trespassers and animals entering into agricultural fields with good accuracy. In the field of agriculture there are more ways to develop or convert this project in many ways. Thus, this project as an efficient scope in coming future where manual predicting can be converted to computerized production in a cheap way.

## **13. APPENDIX**

The project aims to develop a smart crop protection system for animals and birds that can be controlled remotely. The system will be equipped with sensors that will detect the presence of animals and birds and trigger an alarm. The alarm will be sent to a control panel that will be located in the farmer's house. The control panel will be connected to a computer or smartphone, which will be used to control the system. The system will also be equipped with a camera that will be used to take pictures or videos of the animals and birds. The pictures and videos will be stored in a database that will be used to identify the animals and birds.

### **SOURCE CODE:**

```
import cv2
import numpy as np
import wiotp.sdk.device
import playsound
import random
import time
import datetime
import ibm_boto3
from ibm_botocore.client import Config, ClientError
```

```

#cloudantDB
from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel
from clarifai_grpc.grpc.api import service_pb2_grpc
stub = service_pb2_grpc.V2Stub(ClarifaiChannel.get_grpc_channel())
from clarifai_grpc.grpc.api import service_pb2, resources_pb2
from clarifai_grpc.grpc.api.status import status_code_pb2
#This is how you authenticate.
metadata = (('authorization', 'Key 84a796d4dedb4dcc8aa71810ab16a3fd')),
COS_ENDPOINT="https://samplestoragebucket123.s3.jp-tok.cloud-object-
storage.appdomain.cloud"
COS_API_KEY_ID="2BvtnkJHb5xtAlatYUEEWhHnKG-NnbtVOgCfmyyaUSy4"
COS_AUTH_ENDPOINT="https://iam.cloud.ibm.com/identity/token"
COS_RESOURCE_CRN="crn:v1:bluemix:public:cloud-object-
storage:global:a/a6a97499ed5743b7b1f42a2f32c439e7:9f6410d4-28bd-4964-a37d-
06f665016fae::"
Clientdb=Cloudant("apikey-v2-
jruolni88tlw585vgvj72y3qsx0p75efzg2vd27kzex","a4c83e0fb38a78ed85faeefa63fed009",ur
l="https://apikey-v2-
jruolni88tlw585vgvj72y3qsx0p75efzg2vd27kzex:a4c83e0fb38a78ed85faeefa63fed009@1b5
ee1fe-1ffb-443d-a32a-599d7b39cb69-bluemix.cloudantnosqldb.appdomain.cloud")
clientdb.connect()
# Create resource
cos = ibm_boto3.resource("s3",
    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_RESOURCE_CRN,
    ibm_auth_endpoint=COS_AUTH_ENDPOINT,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)
def multi_part_upload(bucket_name, item_name, file_path):
    try:
        print("Starting file transfer for {0} to bucket: {1}\n".format(item_name, bucket_name))
        #set 5 MB chunks
        part_size = 1024* 1024 *5
        #set threshold to 15 MB
        file_threshold = 1024 * 1024 *15
        #set the transfer threshold and chunk size

```

```

transfer_config = ibm_boto3.s3.transfer.TransferConfig(
    multipart_threshold=file_threshold,
    multipart_chunksize=part_size
)
#the upload_fileobj method will automatically execute a multi-part upload
#in 5MB chunks for all files over 15MB
with open(file_path, "rb") as file_data:
    cos.Object(bucket_name, item_name).upload_fileobj(
        Fileobj=file_data,
        Config=transfer_config
    )
    print("Transfer for {0} Complete!\n".format(item_name))
except ClientError as be:
    print("CLIENT ERROR: {0}\n".format(be))
except Exception as e:
    print("Unable to complete multi-part upload: {0}".format(e))

def myCommandCallback(cmd):
    print("Command received: %s" %cmd.data)
    command=cmd.data['command']
    print(command)
    if(command=='lighton'):
        print('lighton')
    elif(command=='lightoff'):
        print('lightoff')
    elif(command=='motoron'):
        print('motoron')
    elif(command=='motoroff'):
        print('motoroff')
myConfig = {
    "identity": {
        "orgId": "gm2lj6",
        "typeId": "Smartcrop",
        "deviceId": "12345"
    },
    "auth": {
        "token":"12345678"
    }
}
client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)

```



```

client.connect()

database_name= "noderedtoyyw20221019"
my_database = clientdb.create_database(database_name)
if my_database.exists():
    print(f'''{database_name}' succesfully created.")
cap=cv2.VideoCapture('monkey.mp4')
if(cap.isOpened()==True):
    print('File opened')
else:
    print('File not found')

while(cap.isOpened()):
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    ims = cv2.resize(frame, (960, 540))
    cv2.imwrite('ex.jpg',ims)
    with open("ex.jpg", "rb") as f:
        file_bytes = f.read()
    # this is the model ID of a publicly available General model. You may use any other public
    or custom model ID.
    request = service_pb2.PostModelOutputsRequest(
        model_id='aaa03c23b3724a16a56b629203edc62c',
inputs=[resources_pb2.Input(data=resources_pb2.Data(image=resources_pb2.Image(base6
4=file_bytes))
    ))
    response = stub.PostModelOutputs(request, metadata=metadata)
    if response.status.code != status_code_pb2.SUCCESS:
        raise Exception("Request failed, status code: " + str(response.status.code))
    detect= False
    for concept in response.outputs[0].data.concepts:
        #print("%12s. %.2f" % (concept.name, concept.value))
        if(concept.value>0.98):
            #print(concept.name)
            if(concept.name=="animal"):
                print("Alert!Alert!Animal Detected")
                playsound.playsound('alert.mp3')
                picname=datetime.datetime.now().strftime("%y-5m-%d-%H-%M")
                cv2.imwrite(picname+'.jpg',frame)
                multi_part_upload('samplestoragebucket123', picname+'.jpg', picname+'.jpg')

```

```

json_document={"link":COS_ENDPOINT+'/'+'samplestoragebucket123+'/'+'picname+'.jpg'}
new_document = my_database.create_document(json_document)
if new_document.exists():
    print(f"Document successfully created.")
    time.sleep(5)
    detect=True
moisture=random.randint(0,100)
humidity=random.randint(0,100)
myData={'Animal': detect,'moisture':moisture,'humidity':humidity}
print(myData)
if(humidity!=None):
    client.publishEvent(eventId="status",    msgFormat="json",    data=myData,    qos=0,
onPublish=None)
    print("Publish Ok..")
    client.commandCallback = myCommandCallback
    cv2.imshow('frame',ims)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
client.disconnect()
cap.release()
cv2.destroyAllWindows()

```

### **GITHUB & PROJECT DEMO LINK:**

Github link: <https://github.com/IBM-EPBL/IBM-Project-26534-1660029134>

Project Demo link: <https://youtu.be/gaC9b8hoxe4>