project on

IOT based smart crop protection system for agriculture

powered by IBM india

submitted by

HARI KRISHNAN V

KANAGARAJ K

KORKAIMARAN M

NARESH KUMAR V

SURYA RAO M

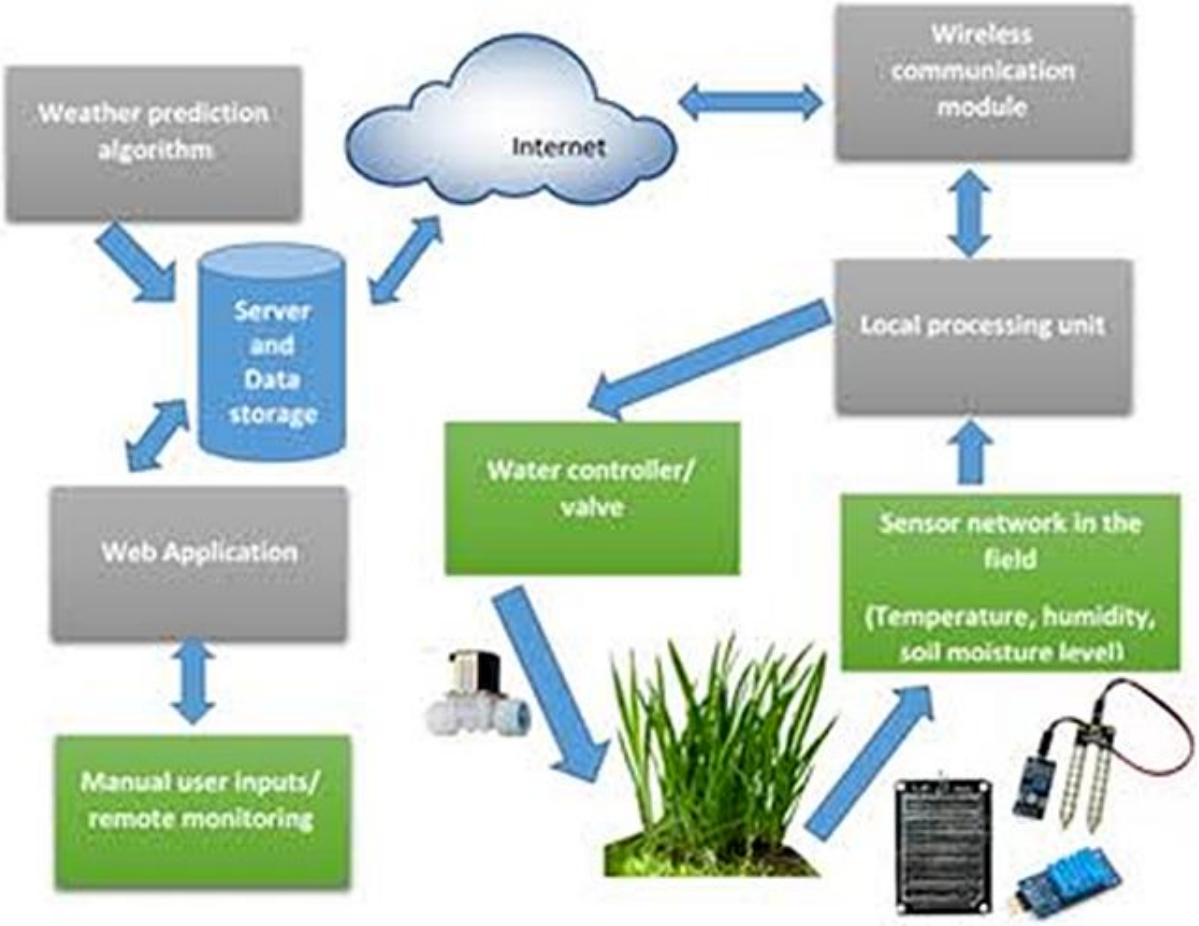
PROJECT ID PNT2022TMID07140

Introduction

Automation of farm activities can transform agricultural domain from being manual and static to intelligent and dynamic leading to higher production with lesser human supervision. This paper proposes an automated irrigation system which monitors and maintains the desired soil moisture content via automatic watering. Microcontroller ATMEGA328P on arduino uno platform is used to implement the control unit. The setup uses soil moisture sensors which measure the exact moisture level in soil. This value enables the system to use appropriate quantity of water which avoids over/under irrigation. IOT is used to keep the farmers updated about the status of sprinklers. Information from the sensors is regularly updated on a webpage using GSM-GPRS SIM900A modem through which a farmer can check whether the water sprinklers are ON/OFF at any given time. Also, the sensor readings are transmitted to a Thing speak channel to generate graphs for analysis.

literature Survey

Identification, Selection and Analysis of the Literature



References

[1] Csótó, Magyar, “Information flow in agriculture – through new channels for improved effectiveness”, Journal of Agricultural Informatics 1 (2), 25–34, 2010

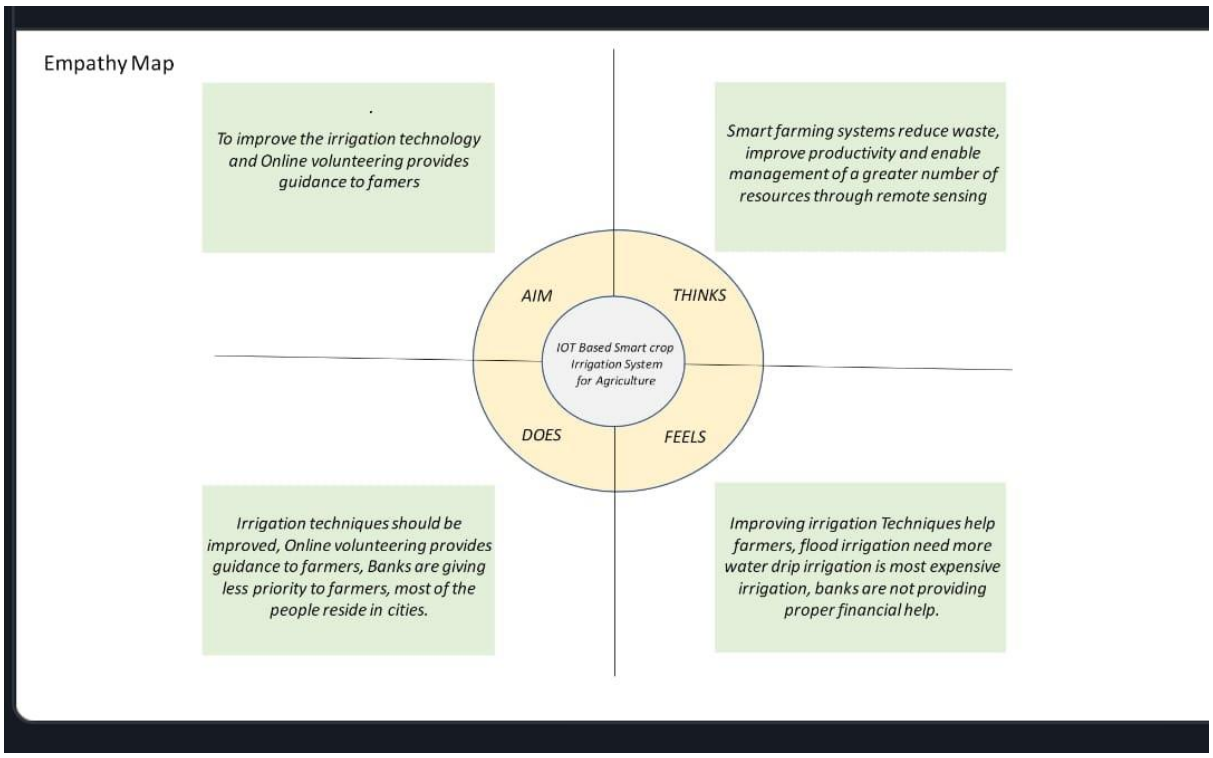
[2] R.Suresh, S.Gopinath, K.Govindaraju, T.Devika, N.SuthanthiraVanitha, “GSM based Automated IrrigationControl using Raingun Irrigation System”,

InternationalJournal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 2, February 2014

[3] Sumeet. S. Bedekar, Monoj. A. Mechkul, and Sonali. R. Deshpande “IoT based Automated Irrigation System”, IJSRD - International Journal for Scientific Research& Development| Vol. 3, Issue 04, 2015 | ISSN (online): 2321- 0613

Ideation and proposed solution

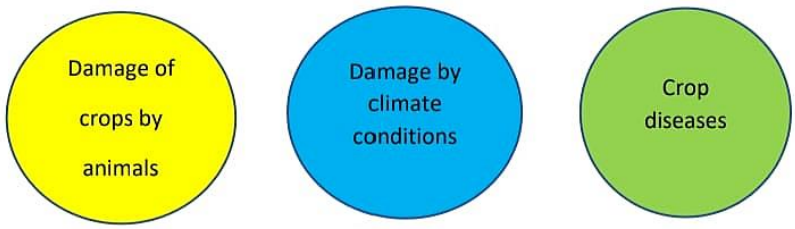
EMPATHY MAP



BRAINSTORAM

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. Team gathering Define who should participate in the session and send an invite. Share relevant information or prework ahead. Set the goal Think about the problem you'll be focusing on solving in the brainstorming session. Learn how to use the facilitation tools Use the Facilitation Superpowers to run a happy and productive session. Define your problem statement What problem are you trying to solve? Frame your problem as How Might We state this will be the focus of your brain storm.

Problems faced:



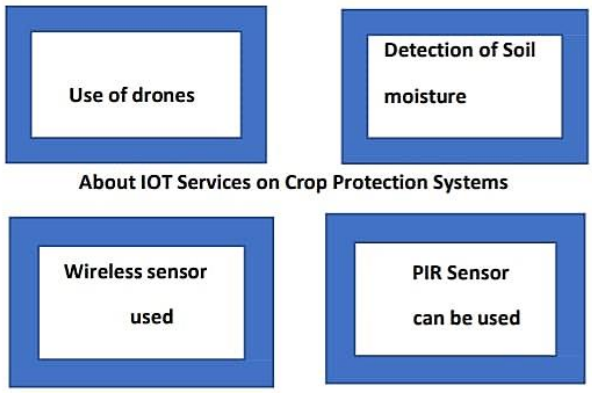
Key rules of brainstorming

To run an smooth and productive session

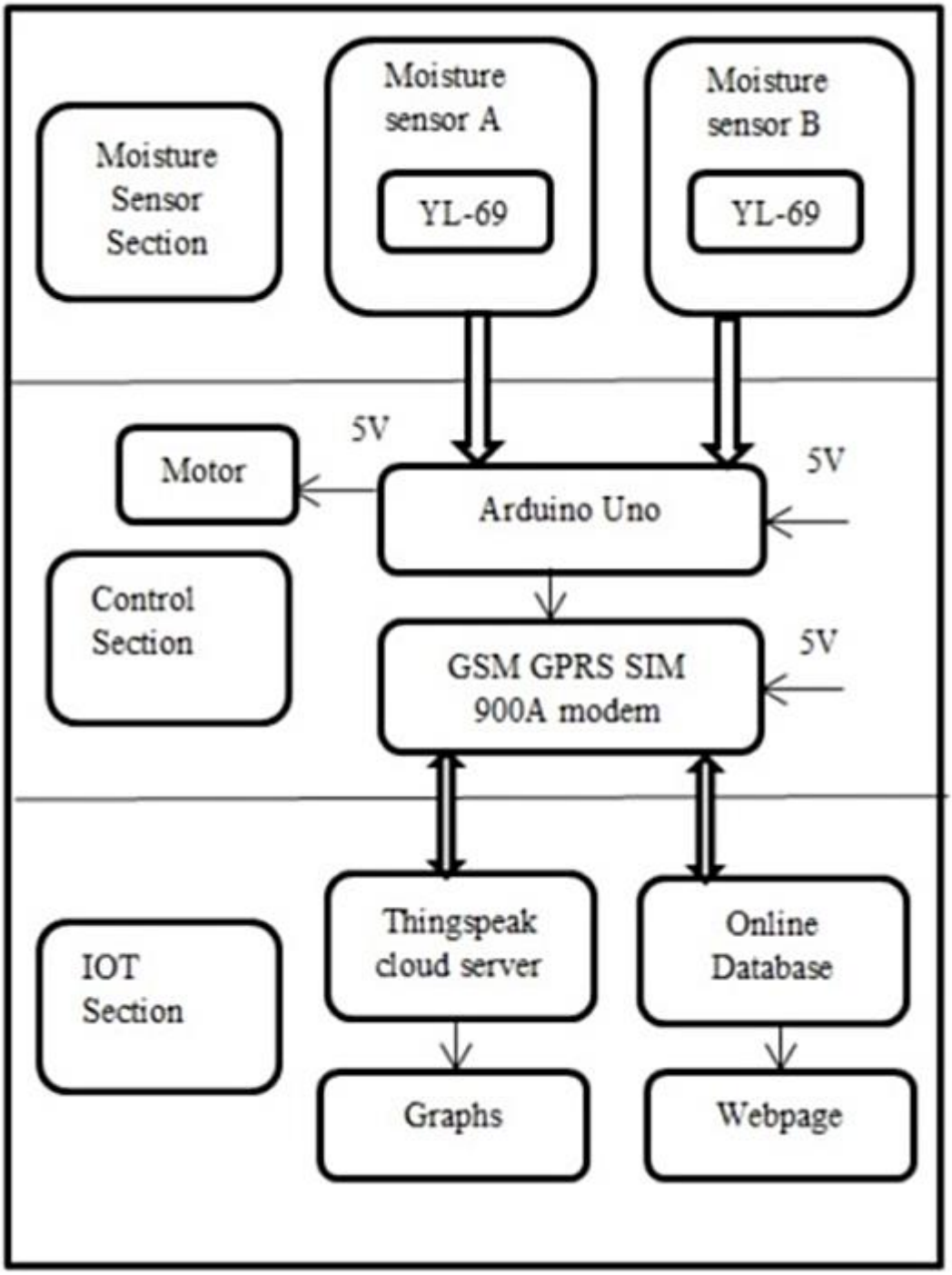
- Stay in topic.
- Defer judgment
- Encourage wild ideas.
- Listen to others.
- Go for volume.
- If possible, be visual.

Brainstorm

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



Requirements Analysis



Project Planning and Scheduling

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points(40)	Priority (Low to High)	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the required dataset by entering my email, password, and confirmingmy password.	3	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-1		USN-2	As a user, I will receive confirmation email andthe SMS once I have registered for the application	2	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-2	Cloud services	USN-3	As a user, I can register for the applicationthrough Facebook or any socialmedia	1	Low	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-4		USN-4	As a user, I can register for the applicationthrough Gmail/web service	2	Medium	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-3	Login	USN-5	As a user, I can log into the application networkby entering email & password	4	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-2	Pre processing	USN-6	As a farmer, the user must be able to find the system easy to access so pre-processes and othertask mustbe perfect.	3	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-1	Collecting Dataset	USN-7	To collect various sources of animal threats andkeep developing a dataset.	3	Medium	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-4	Integrating	USN-8	To integrate the available dataset and keepimproving the accuracy of finding animals	2	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao

Sprint-3		USN-9	To find and use appropriate compiler to run andtest the data so that we can implement our program	1	Low	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-2		USN-10	Request Saveetha Engineering College to deploythe project in our campusand test	1	Low	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-1	Training	USN-11	As programmer, we need to train our data perfectly so that the program runssmoothly	3	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-3		USN-12	Train thedata using outavailable services and IBM dataset from server and improve that	2	Medium	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-4	Coding	USN-13	To modify the code according to our programand improve the efficiency of that code	4	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-2		USN-13	To improve performance	1	Low	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-2	Record	USN-5	To record the data and plot the graph to show thecharacteristics officially	4	High	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao
Sprint-1	Planning	USN-4	Plan the programming language and feasibility	3	Medium	Kanagaraj Naresh kumar Korkaimaran HariKrishnan Surya rao

Coding and Solutioning
python coding

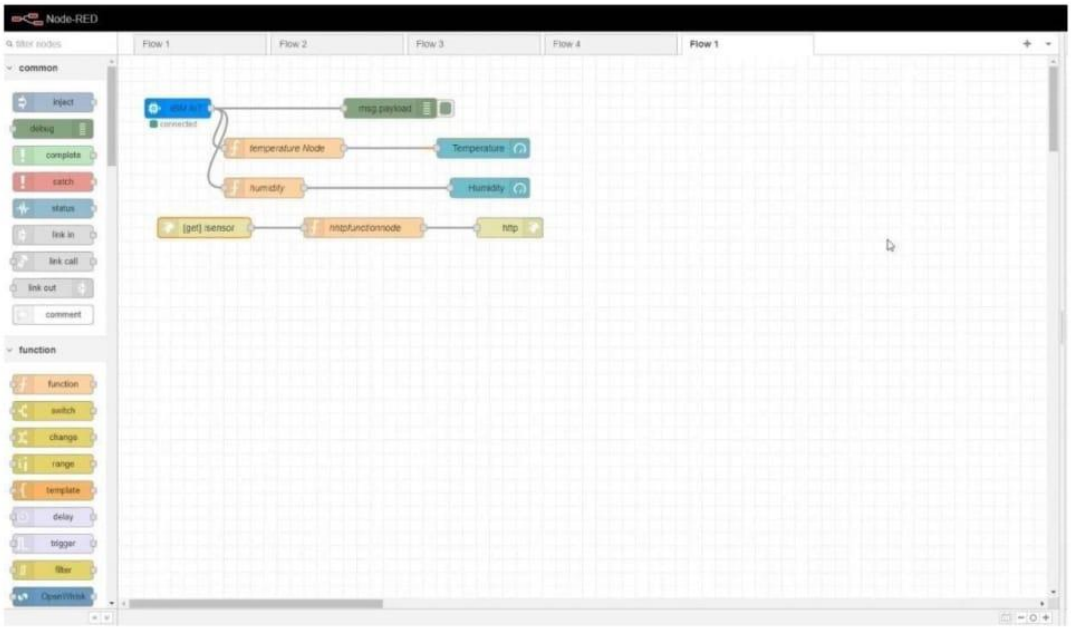
```
import cv2
import numpy as np
import wiot.sdk.device
import playsound
import random
import time
import datetime
import ibm_boto3
from ibm_botocore.client import Config, ClientError

#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, resource_pb2
from clarifai_grpc.grpc.api.status import status_code_pb2


#This is how you authenticate
metadata = (('authorization', 'key 0620e202302b4508b90eab7efe7475e4'),)
COS_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud"
COS_API_KEY_ID = "g5d4qO8EIgv4TWUCJj4hfEzgalqEjrDbE82AJDWlAOHo"


COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"
COS_RESOURCE_CRN = "crn:v1:bluemix:public:cloud-object-
storage:global:a/c2fa2836eaf3434bbc8b5b58fefff3f0:62e450fd-4c82-4153-ba41-
ccb53adb8111::"
clientdb = cloudant("apikey-W2njldnwtjO16V53LAVUCqPwc2aHTLmlj1xXvtdGKJBn",
"88cc5f47c1a28afbfb8ad16161583f5a", url="https://d6c89f97-cf91-48b7-b14b-
c99b2fe27c2f-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
```

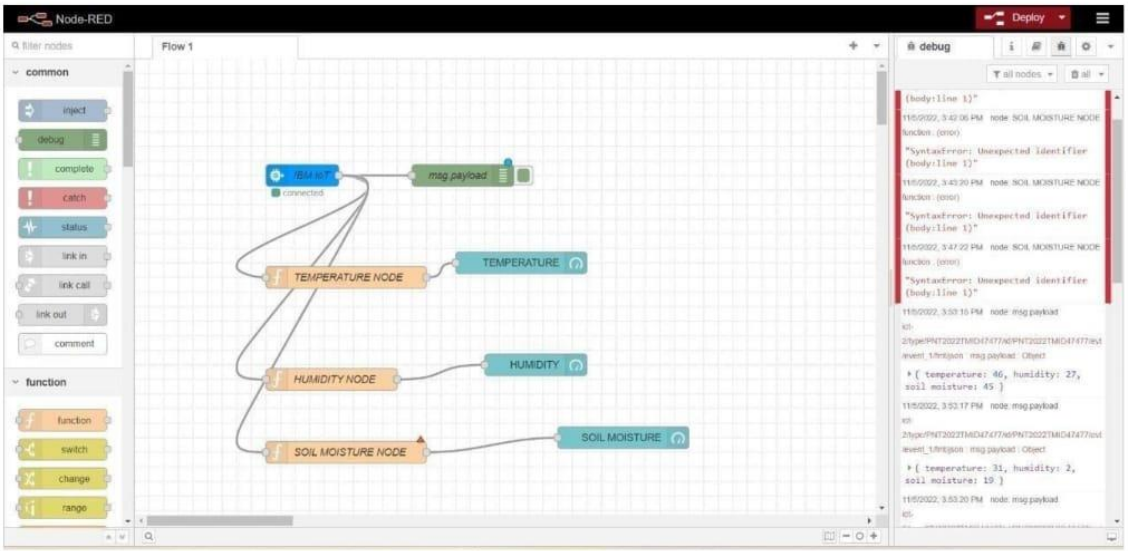
```
                    )
def = multi_part_upload(bucket_name, item_name, file_path):
    try:
        print("Starting file transfer for {0} to bucket: {1}\n".format(item_name,
bucket_name))
        #set 5 MB chunks
        part_size = 1024 * 1024 * 5
        #set threadhold to 15 MB
        file_threshold = 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 5 MB chunks size
        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(commamd=="lighton"):
        print('lighton')
    elif(command=="lightoff"):
        print('lightoff')
    elif(command=="motoron"):
        print('motoron')
    elif(command=="motoroff"):
        print('motoroff')
myConfig = {
    "identity": {
        "orgId": "chytun",
        "typeId": "NodeMCU",
        "deviceId": "12345"
        },
```

```
    "auth": {
       "token": "12345678"
       }
    }
client = wiot.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

database_name = "sample"
my_database = clientdb.create_database(database_name)
if my_dtabase.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 = cv3.cvtColor(frame, cv2.COLOR_BGR@GRAY)
    imS= cv2.resize(frame, (960,540))
    cv2.inwrite('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='e9359dbe6ee44dbc8842ebe97247b201',




inputs=[resources_pb2.Input(data=resources_pb2.Data(image=resources_pb2.Imag
e(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: %.f' % (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.inwrite(picname+'.jpg',frame)
```

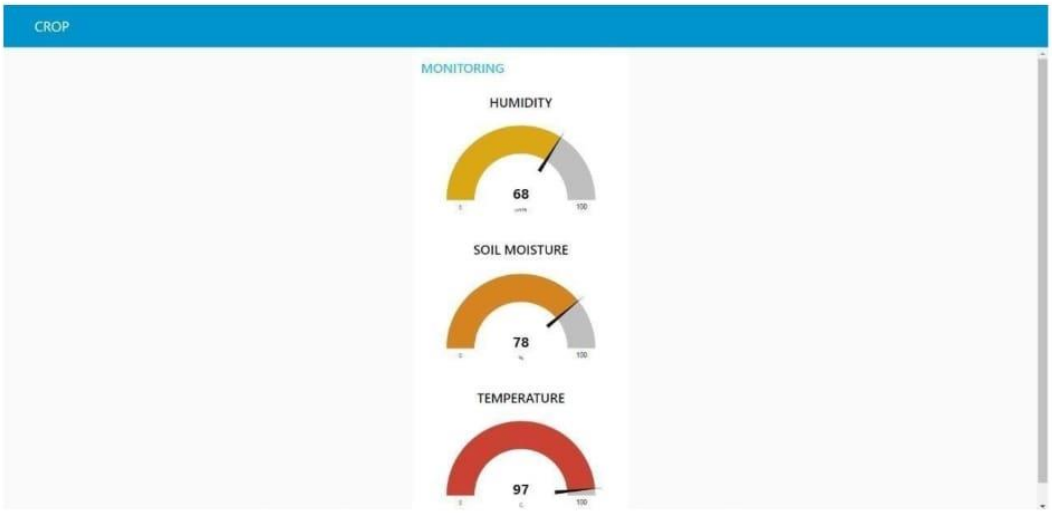
```
            multi_part_upload('Dhakshesh', picname+'.jpg', picname+'.jpg')
            json_document={"link":COS_ENDPOINT+'/'+'Dhakshesh'+'/'+picname+'.jpg'}
            new_document = my_database.create_document(json_document)
            if new_document.exists():
               print(f"Document successfully created.")
            time.sleep(5)
            detect=True
    moist=random.randint(0,100)
    humidity=random.randint(0,100)
    myData={'Animal':detect,'moisture':moist,'humidity':humidity}
    print(myData)
    if(humidity!=None):
        client.publishEvent(eventId="status",msgFormat="json", daya=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()
```

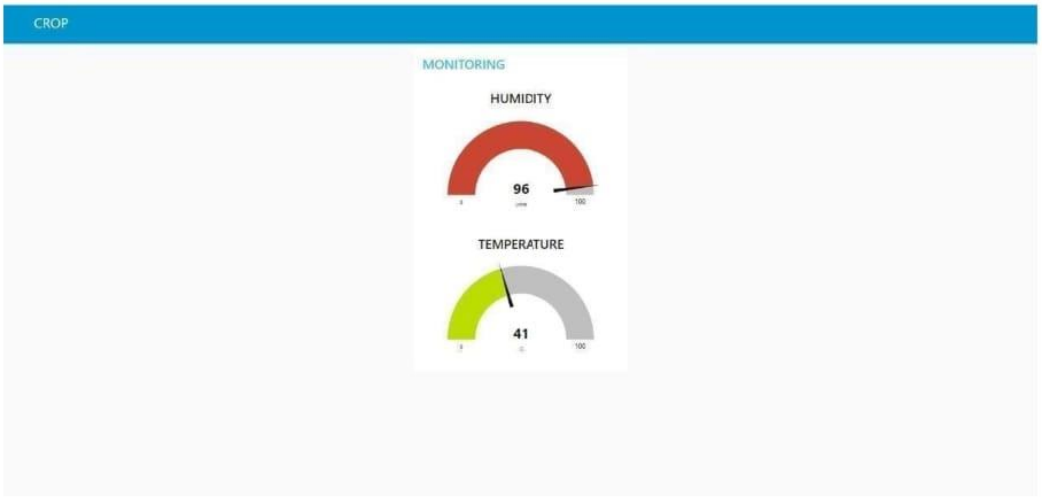
Testing


Node-RED
Flow 1
Flow 2
Flow 3
Flow 4
Flow 1
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
template
delay
trigger
filter
msg.payload
temperature Node
Temperature
humidity
Humidity
[get] tsensor
http


Node-RED
Deploy
Flow 1
debug
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
msg.payload
TEMPERATURE NODE
TEMPERATURE
HUMIDITY NODE
HUMIDITY
SOIL MOISTURE NODE
SOIL MOISTURE

Result


CROP
MONITORING
HUMIDITY
68
SOIL MOISTURE
78
TEMPERATURE
97


CROP
MONITORING
HUMIDITY
96
TEMPERATURE
41

Advantages and Disadvantages

Smart irrigation systems offer a variety of advantages over traditional irrigation systems. Smart irrigation systems can optimize water levels based on things such as soil moisture and weather predictions. This is done with wireless moisture sensors that communicate with the smart irrigation controls and help inform the system whether or not the landscape is in need of water.

Additionally, the smart irrigation controlled receives local weather data that can help it determine when a landscape should be watered. If you have ever returned home during a storm only to see your sprinklers spraying water you know how beneficial this is. Rather than wasting water resources and your valuable money on watering your landscape you can take advantage of the nature moisture from the storm and save that water for another day when it is more needed. The advantages of these smart irrigation systems are wide reaching.

The smart irrigation system will help you have better control of your landscape and irrigation needs as well as peace of mind that the smart system can make decisions independently if you are away. You will save a significant amount of money on your water bills because through intelligent control and automation, your smart irrigation system will optimize resources so that everything gets what it needs without needless waste. Additionally, we have all seen many places in the country that have experienced droughts and we know that our water resources are precious.

Conclusion

A system to monitor moisture levels in the soil was designed and the project provided an opportunity to study the existing systems, along with their features and drawbacks. The proposed system can be used to switch on/off the water sprinkler according to soil moisture levels thereby automating the process of irrigation which is one of the most time consuming activities in farming. Agriculture is one of the most water-consuming activities. The system uses information from soil moisture sensors to irrigate soil which helps to prevent over irrigation or under irrigation of soil thereby avoiding crop damage.

Future Scope

The farm owner can monitor the process online through a website. Through this project it can be concluded that there can be considerable development in farming with the use of IOT and automation. Thus, the system is a potential solution to the problems faced in the existing manual and cumbersome process of irrigation by enabling efficient utilization of water resources.