Documentation
On
Smart plant Communicator
using IBM Cloud

Contents

1. Introduction
    a. Overview
    b. Purpose
2. Literature Survey
    a. Existing problem
    b. Proposed solution
3. Theoretical Analysis
    a. Block diagram
    b. Hardware / Software designing
4. Experimental Investigations
5. Flowchart
6. Result
7. Advantages & Disadvantages
8. Applications
9. Conclusion
10. Future Scope
11. Bibliography
12. Appendix
    a. Source code
    b. UI output Screenshot.

1.Introduction :

A) Overview: The main aim of the project is to build a smart plant monitoring system. It is used to check the three main factors of the plant which are

- Temperature
- Humidity
- Soil moisture

The soil moisture indicates weather the motor to ON or OFF. And in this way by the code given we will randomly get the 3 main factors and can work accordingly.

B) Purpose : This is suitable to check the soil conditions as well as which may help to stat a better growth of a plant. It mainly shows the amount the water required by the plant and the excess water. And the temperature that indicates the plant growth. We will be able to know the plant conditions based on the three factors .

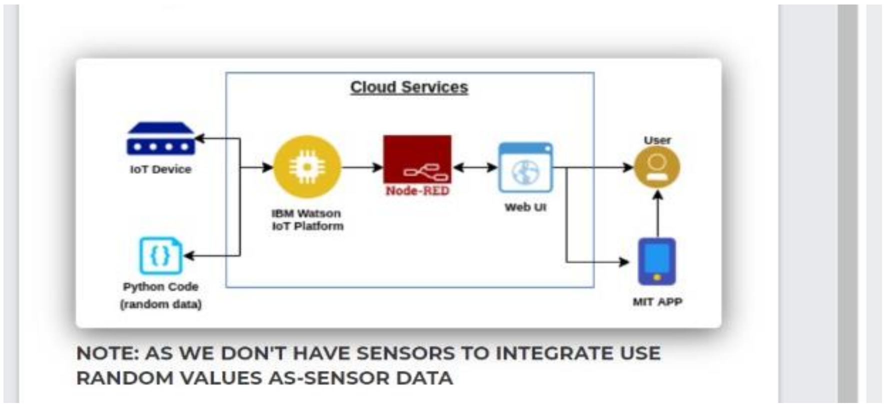
2.Literature survey :

A) Existing problem : As we see lot of plant die due to lack of water or because no care is taken. In the fields the crop doesn't come out good because of no water availability or when proper care is not taken . As we know the water tables and the under ground water are drying up and the crops or plants need sufficient water to grow healthly we propose this project to know the soil moisture , humidity ,and the plant requirements for the good plant heath .

B) Proposed solution : Collecting all the data from various sensors like temp, humidity ,moisture and other environmental factors and will do analysis on the same. During the analysis we get better results by comparing the values , and also can know the plant health and other things.

3.Theoretical analysis

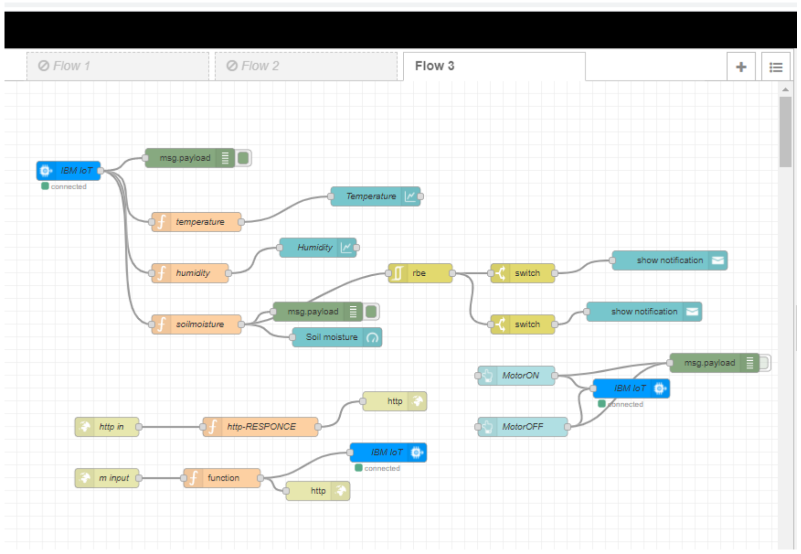
A) **Block diagram**:



NOTE: AS WE DON'T HAVE SENSORS TO INTEGRATE USE RANDOM VALUES AS-SENSOR DATA

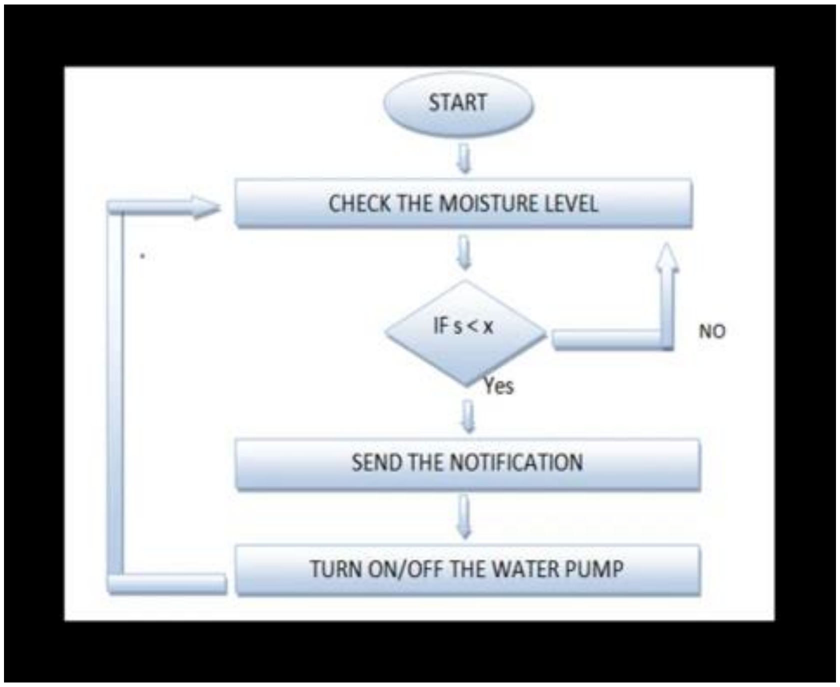
B) Hardware/ software design :

[redacted]

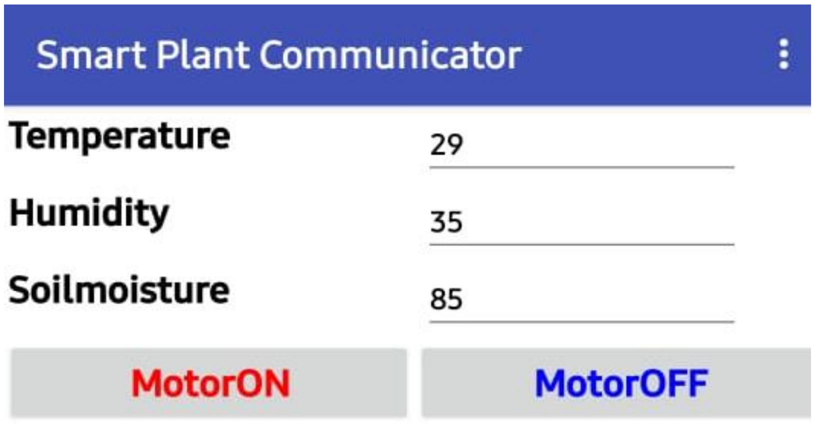


4.Experimental investigations : In this smart plant communicator we need to greet the master first and then check the conditions of the plant. The various factors like water level, humidity, temperature and the plant condition will be known and under vision while going through the experiment .

5.Flow chart :



6.Result :


Smart Plant Communicator

Temperature	29
Humidity	35
Soilmoisture	85

7.Advantages and disadvantages :

Advantages:

1. Helps to know plan
2. Easy to identify the defects
3. Good plant growth

4. The plant is under the vision.

Disadvantages:

1. Implementation is expensive
2. More equipment may be required for large area
3. Climatic changes may harm the plant /crop .

8.Applications :

1. Used in farming / agriculture.
2. Used to industries
3. Helpful at plant nursery
4. Used at home for monitoring plants.

9.Conclusion : by this project we conclude that ,the plant or crop is under the human guidance and will have better performance compared to the earlier . The three main factors which are mostly required and which play the crucial role in plant growth can be know . and the water levels needed for the plant ,temperature that suits for the plant all other factors . so finally we will know the plant health .

10.Future scope : the performance of the plant can be increased by taking care .and the yield and the plant health can be better .it can be used in large area if needed and worked properly.

11.Appendix :

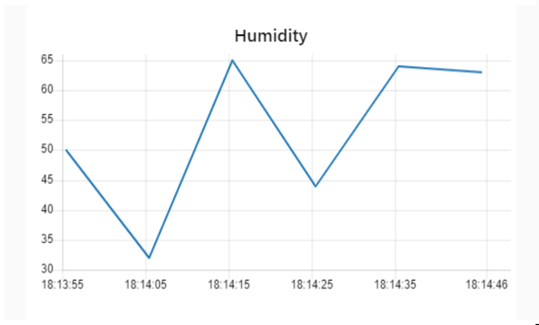
A) Source code :

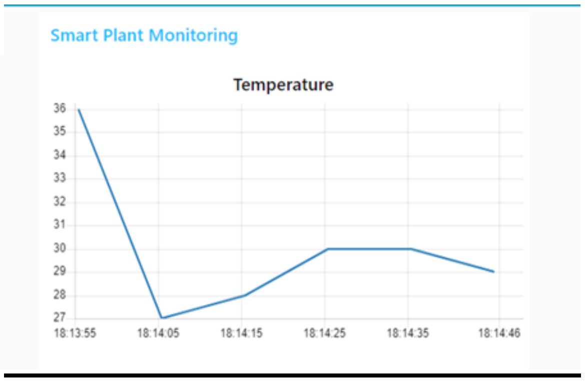
```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
import json
from ibm_watson import TextToSpeechV1
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
import playsound

authenticator = IAMAuthenticator('UH_AiWIbt3uflaEMJGLbpJk6JIL3yJZ7_LHpXQpSaRNB')
text_to_speech = TextToSpeechV1(
    authenticator=authenticator
)
```

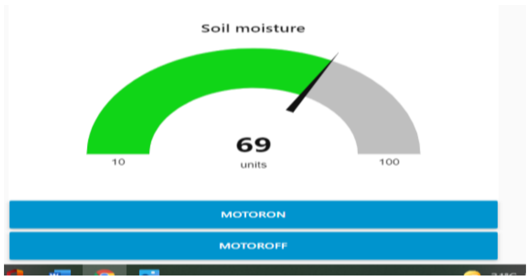
```
text_to_speech.set_service_url('https://api.eu-gb.text-to-
speech.watson.cloud.ibm.com/instances/d8ab37c5-f3ff-4da8-9c91-fdc0430f7458')

with open('new.mp3', 'wb') as audio_file:
    audio_file.write(
        text_to_speech.synthesize(
            'Hello this your plant warm greetings to you',
            voice='en-US_AllisonV3Voice',
            accept='audio/mp3'
            ).get_result().content)

playsound.playsound('new.mp3')



#Provide your IBM Watson Device Credentials
organization = "roduhu"
deviceType = "iotdevice"
deviceId = "1001"
authMethod = "token"
authToken = "1234567890"


# Initialize the device client.
T=0
H=0
S=0
P=0
def myCommandCallback(cmd):
        print("Command received: %s" % cmd.data['command'])


        if cmd.data['command']=='motor on':
                print("MOTOR ON IS RECEIVED")


        elif cmd.data['command']=='motor off':
                print("MOTOR OFF IS RECEIVED")

        if cmd.command == "setInterval":
                if 'interval' not in cmd.data:
                        print("Error - command is missing required information: 'interval'")
                else:
                        interval = cmd.data['interval']
        elif cmd.command == "print":
                if 'message' not in cmd.data:
                        print("Error - command is missing required information: 'message'")
                else:
```

```
                print(cmd.data['message'])

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:
    T=random.randint(27,40)
    H=random.randint(32,65)
    S=random.randint(20,95)
    #Send Temperature & Humidity to IBM Watson
    data = {"d":{ 'temperature' : T, 'humidity': H,'soilmoisture': S }}
    #print data
    def myOnPublishCallback():

       print ("Published Temperature = %s C" % T, "Humidity = %s %%" % H,"Soilmoisture = %s %%" %
S,"to IBM Watson")
       if S<=50:
        print("MotorON")
       else:
        print("MotorOFF")

    success = deviceCli.publishEvent("Data", "json", data, qos=0, on_publish=myOnPublishCallback)
    if not success:
       print("Not connected to IoTF")
    time.sleep(10)

    deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

B) Ui interface :







Team members :

G. Ganesh (18481A0468)

G. Francis (18481A0469)

G. Venkata pavan (18481A0470)

G. Badrinath (18481A0471)

G. Seershveda (18481A0473)