GREEN HOUSE MONITERING SYSTEM

1. INTRODUCTION

1.1 OVERVIEW

Plant needs suitable climatic conditions to grow and yield a good crop. Green house monitoring system allows the farmer to know the temperature,humidity and soil moisture content.

Sprinklers or motor is Self regulated to switch ON/OFF depending upon the soil moisture content.

User can know the temperature, humidity and soil moisture values in Web page or Mobile App.

User has an option of selecting particular crop from the mobile app so that the optimal conditions at the device will be configured according to the crop.

If any parameters reach beyond the threshold values user is notified with messages.

1.2 PURPOSE

The **purpose of a greenhouse** is to shield crops from excess cold or heat , humidity and soil moisture.A **greenhouse** makes it possible to grow certain types of crops year round, and fruits, tobacco plants, vegetables, and flowers are what a **greenhouse** most commonly grows.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

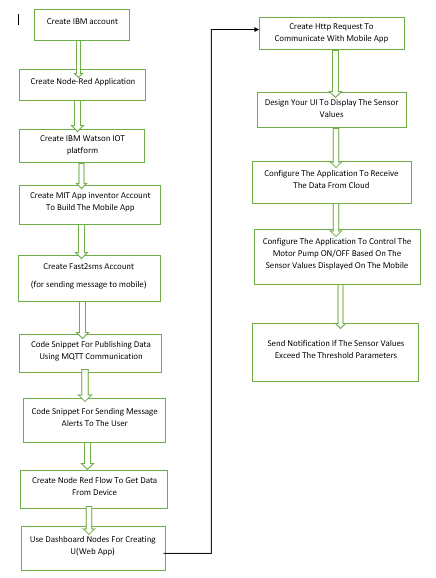
Certain crops requires a particular temperature, humidity and soil moisture conditions such that if those particular conditions are not provided to crops,they won't give much yeild.so to control the environment conditions around the crop,we require a green house monitoring system.

2.2 PROPOSED SOLUTION

To monitor the crops,we are proposing a green house monitoring system in which we can see the temperature, humidity, soil moisture around the crop using a mobile app.user can on and off the motor using buttons.user can also get a alert message whenever the conditions around the crop exceeds it threshold value.

3 .THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM

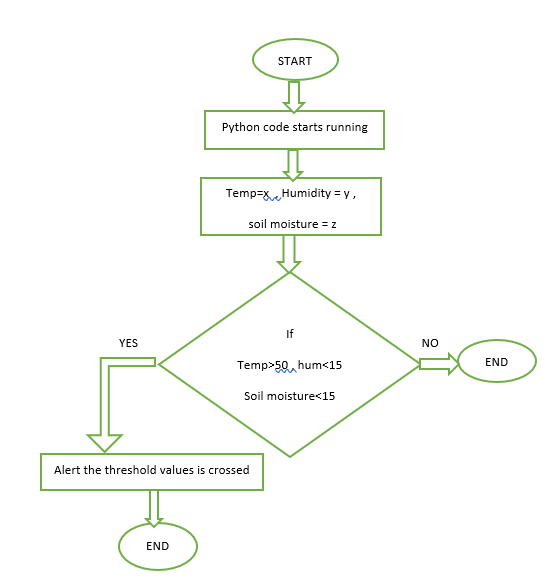


3.2 HARDWARE/SOFTWARE DESIGNING

Python, IBM Cloud, IBM Nodered application, MIT app inventor, Fast2sms.

4. EXPERIMENTAL INVESTIGATION

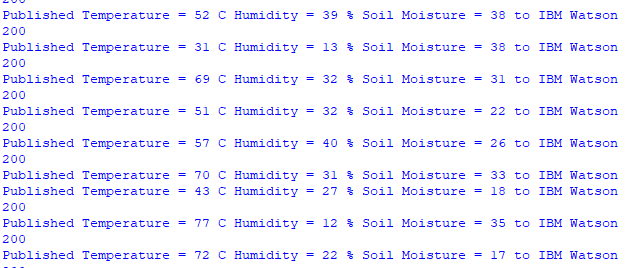
From the obtained results we have observed that,when the soil moisture is below the threshold value(15) or the tempature is above the threshold value(50) or the humidity is below the threshold value(15) then the alert message is received to the mobile by using fast2sms and it display "alert the threshold value is crossed".It also displays temparature,humidity,soil moisture values in the MIT app and user can control the motor action using motor on or motor off buttons that are present in the app.

5. FLOWCHART  


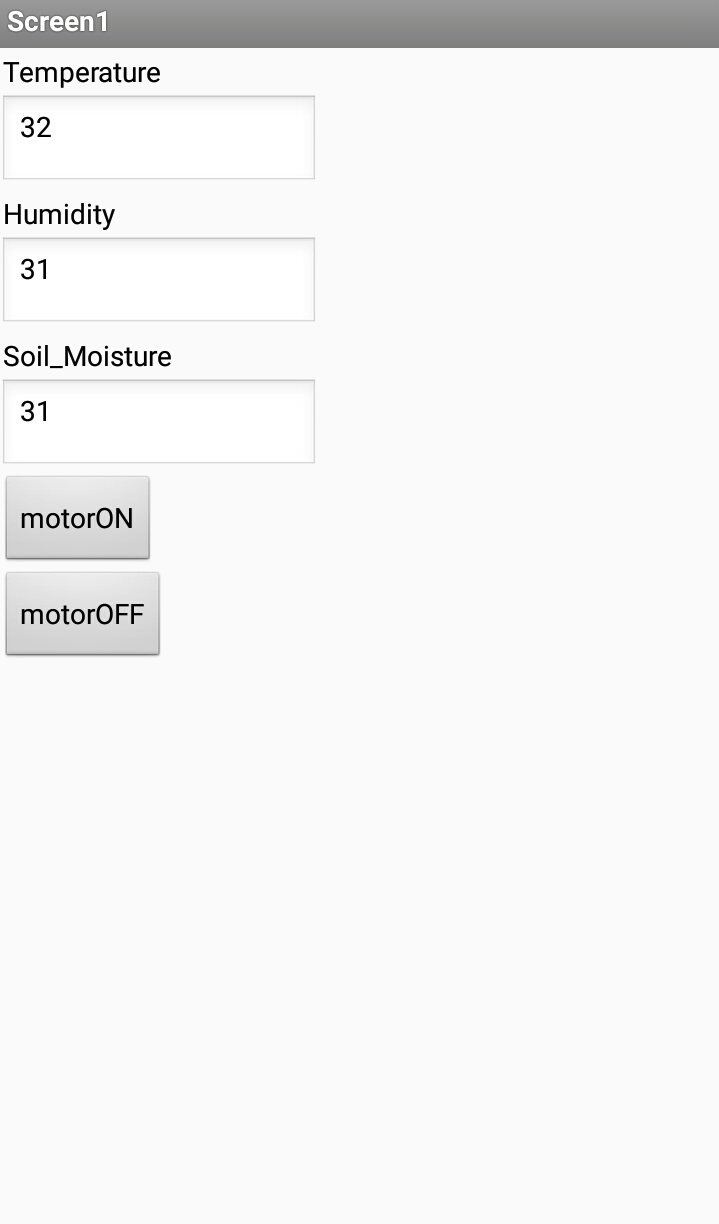
6. RESULT

We have analysed the temperature,humidity and soil moisture levels and found that they accurately fit for the crop growth . The usage of Nodered and MIT App inventor enables user to control the motor on/off depending upon the moisture levels. This model has highest accuracy and is a good choice for this problem. This will make our model more robust.

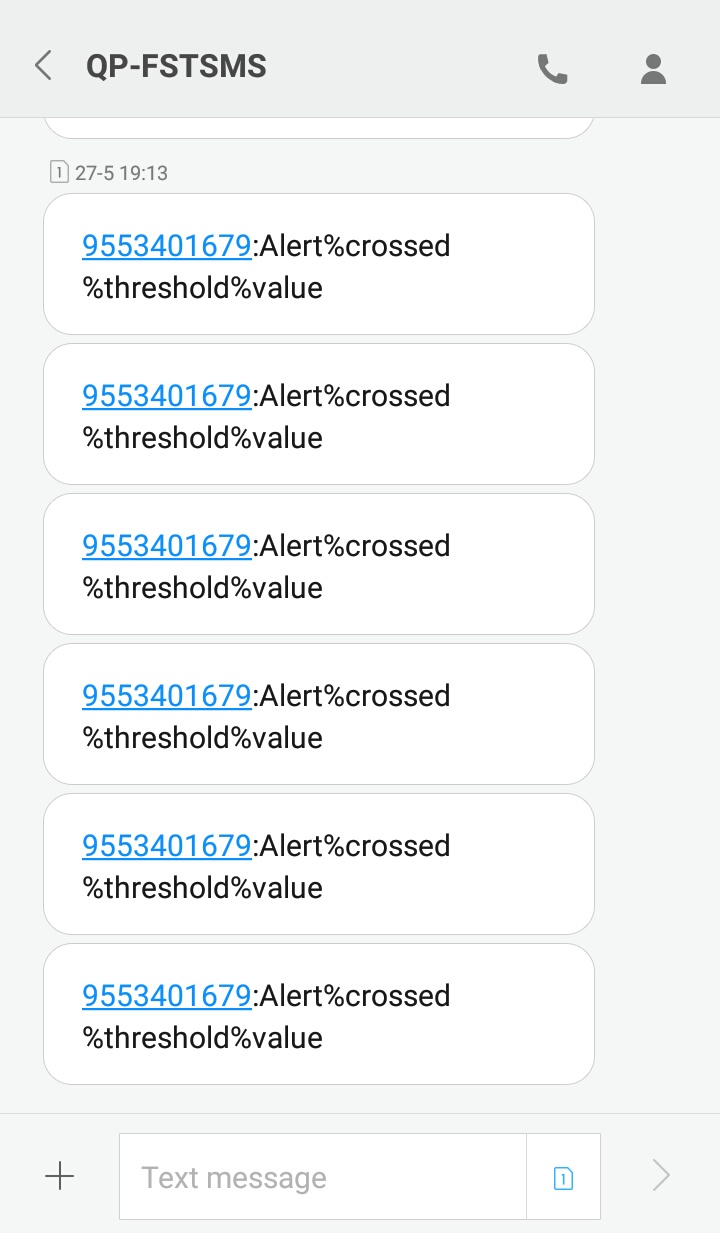
RESULTANT OUTPUT OF PYTHON CODE



RESULTANAT OUTPUT OF MIT APP INVENTOR



RESULTANT OUTPUT OF FAST2SMS



7) ADVANTAGES AND DISADVANTAGES

Advantages:

 This type of green house monitering system saves time,cost effectiveness and productivity of crops can be improved.

 Inexpensive and easy to install.

Disadvantages :

 There can be Hardware failure of components like sensors after using themfor a time long.So they may have to replaced.

APPLICATIONS

 Used in Smart Home Automation

 Used in Smart irrigation

 Used in the case of Physically challenged people like blink stick and ear

instruments.

 Used in Driverless cars,Smart Attendance systems and many more.

8) CONCLUSION

In this study, green house monitering system has enabled the high crop yielding by installing sensors in the fields through continuous monitoring of Weather including temperature,humidity and soil moisture levels made this project to be a successful one in smart weather forcast for green house monitering systems.

Future Scope

 Areas of drought condition, IoT can prove to be a great value as it manages

the limited water supply smartly with least wastage of water resource.

 Smart irrigation enables in pest control,crop productivity,reduces wastage of

water.

9) BIBLIOGRAPHY

Books

 Attar, S., &amp; Sudhakar, K. N. Real-Time Monitoring Of Agricultural Activities Using Wireless Sensor Network.

 Awasthi, A., &amp; Reddy, S. R. N. (2013). Monitoring for Precision Agriculture using Wireless Sensor Network-A review. GJCST-E: Network, Web &amp;

Security, 13(7).

DATA REPOSITERIES

Github repositary

ALGORITHMS

thesmartbridgeteachable.com

10. APPENDIX

SOURCE CODE

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

import requests

url = "https://www.fast2sms.com/dev/bulk"

#Provide your IBM Watson Device Credentials

organization = "pwyh4c"

deviceType = "greenhouse"

deviceId = "572604"

authMethod = "token"

authToken = "12345678"

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data)#Commands

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:

hum=random.randint(10, 40)

#print(hum)

temp =random.randint(30, 80)

sm=random.randint(10, 40)

#Send Temperature & Humidity & Soil Moisture to IBM Watson

data = { 'Temperature' : temp, 'Humidity': hum,'Soil\_Moisture':sm }

if (hum<15 or temp>50 or sm<15):

r=requests.get("https://www.fast2sms.com/dev/bulk?authorization=YpMetB4F7XQnE0iDghZRkHJvCTofAOs5Gc1PI8uqlSdU6Nmar2neW1wki4dZBAOPUv37NXrKGf205Jaq&sender\_id=FSTSMS&message=Alert%crossed%threshold%value&language=english&route=p&numbers=9553401679")

print(r.status\_code)

#print (data)

def myOnPublishCallback():

print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % hum,"Soil Moisture = %s"%sm, "to IBM Watson")

success = deviceCli.publishEvent("Weather", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF")

time.sleep(2)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()