

# **PROJECT REPORT**

**ON**

**Smart Agriculture System Based on  
Internet of Things**

**INTERNSHIP OFFERED BY: SMART BRIDGE**

**PROJECT MENTOR: Mr. Durga Prasad**

**Submitted By:**

**N.V.S. ADITYA**

**B.TECH, CSE, GITAM**

# **1. INTRODUCTION:**

## **1.1. OVERVIEW:**

In Agriculture, yield depends on many factors such as seeds quality, soil type, moisture, temperature and other climatic factors. As a result, production of food-grains fluctuates year after year if any of factors make an impact. A year of abundant output of cereals is often followed by a year of acute shortage especially in India. Due to this problem, obtained total yeild was not meeting to food requirements of people and as a result leaving many people to starvation. This has been for many years due to Traditional Agriculture was followed. In the recent years Government started many initiatives like setting soil testing labs, good quality fertilizers and seeds and modern equipment like tractors etc. and most importantly Modern Agriculture had taken shape. But still many farmers do not have any information on climatic and plant conditions before hand so that requires action can be taken care.

## **1.2. PURPOSE:**

Through many scientific research, it is found that knowing beforehand the climatic conditions by farmers with an easy UI (User Interface) so that they can monitor closely and perform required actions.

Therefore the purpose of this project is to make a Smart Agriculture System based on Internet of Things where the

dashboard can give all the agricultural conditions of crops and weather conditions, also the water pump can be toggled on/off through the same dashboard, instead of doing it manually. Also the all the climatic and crop condition information is recorded for future reference and analysis.

## **2. LITERATURE SURVEY:**

### **2.1. EXISTING PROBLEM:**

The Traditional agriculture methods is still used by many farmers and though a small percentage of farmers converted into modern agriculture, majority of yield is not produced due to no easy to use system to closely monitor the crop conditions like moisture, temperature etc. Also another major issue is the unpredictable weather conditions and the farmers wholly depend on Television broadcast which does not give real time updates.

### **2.2. PROPOSED SOLUTION:**

To overcome the above mentioned we proposed to build user friendly dashboard where the farmer can get all the crop conditions in real time and that too remotely and also can track the climatic changes and conditions at his/her location.

To track the crop conditions, we can choose from

variety of IoT devices and micro controllers which monitor the condition and shows the gathered information in the dashboard. Also the motor pump can be controlled remotely.

For the weather changes to be shown in the dashboard we can open weather API for accurate and real time information.

### 3. THEORITICAL ANALYSIS:

#### 3.1. BLOCK DIAGRAM

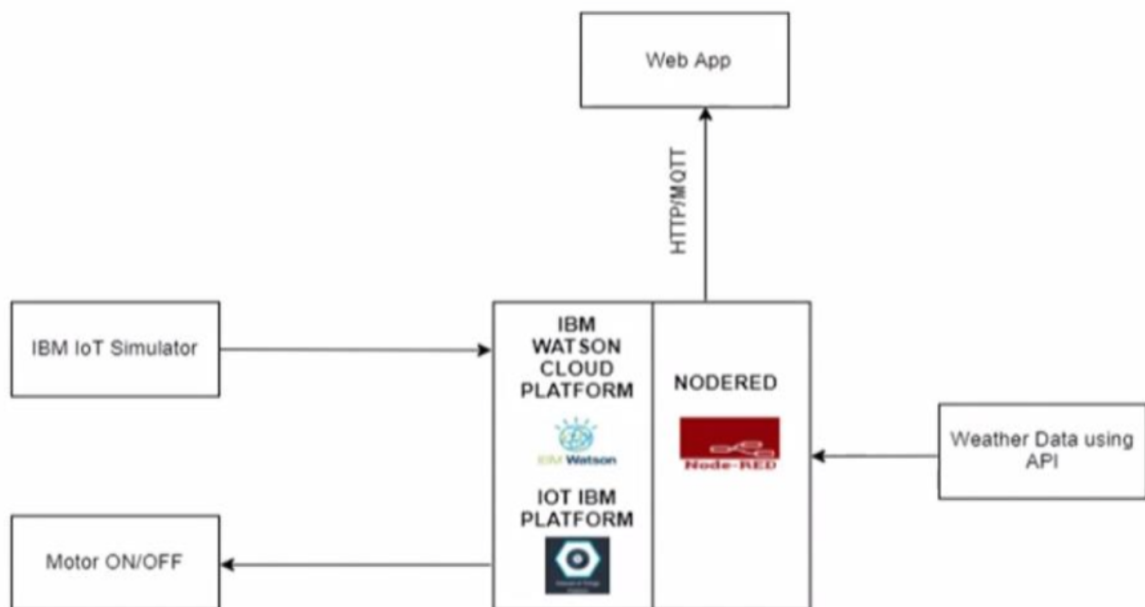


Figure: Block Diagram of The Smart Agriculture

## **3.2. HARDWARE/ SOFTWARE DESIGNING:**

### **3.2.1. HARDWARE:**

Hardware Requirements are IoT based microcontrollers and sensors.

### **3.2.2. SOFTWARE:**

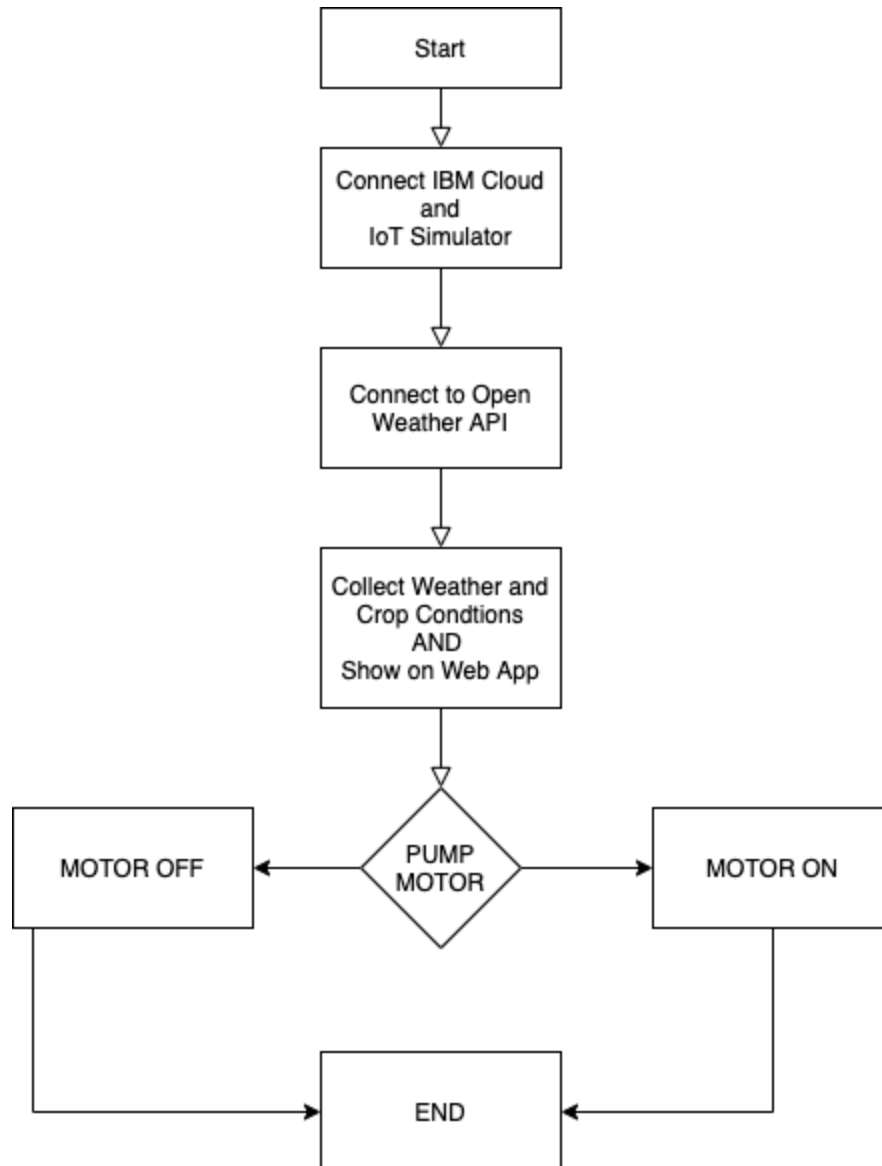
- 1) Watson IoT Platform
- 2) Node-red
- 3) IBM Cloud
- 4) Weather Info API
- 5) Web App

## **4. EXPERIMENTAL INVESTIGATION:**

For Getting information we have used IBM Cloud and Watson IOT Platform to get data from the simulator and Data from openweather API and sent to NODE-RED Platform

Further the Web App showed all the collected information and Motor can be controlled using the buttons in the control center.

## 5. FLOW CHART:



## 6. RESULT:

Data from Weather API and Simulator Data were been shown in UI and can be monitored in real time.

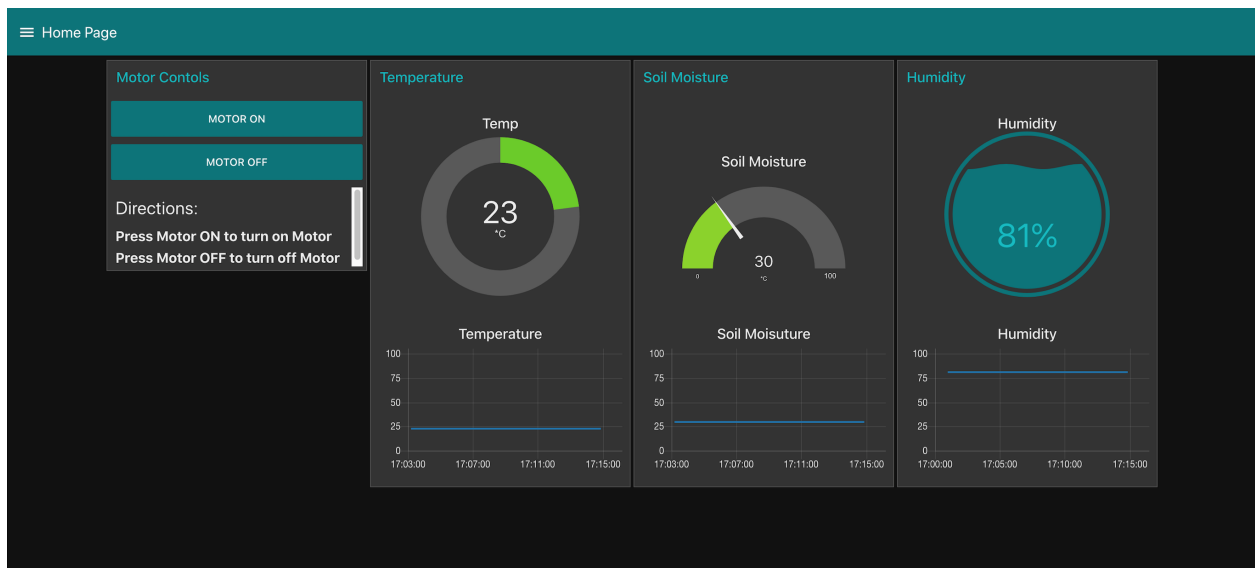


Figure: Showing the Crop Conditions Data in Realtime

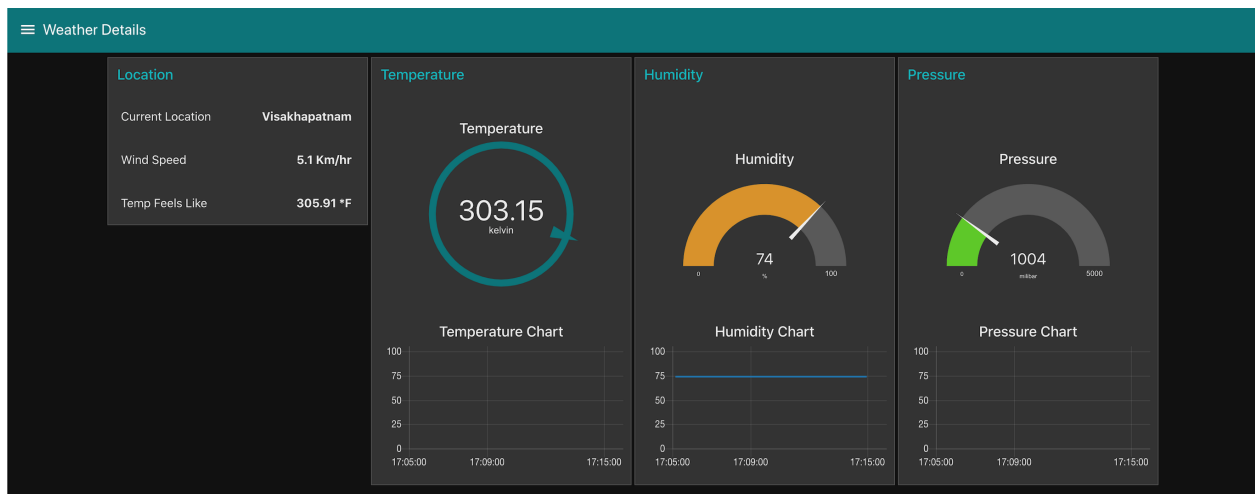
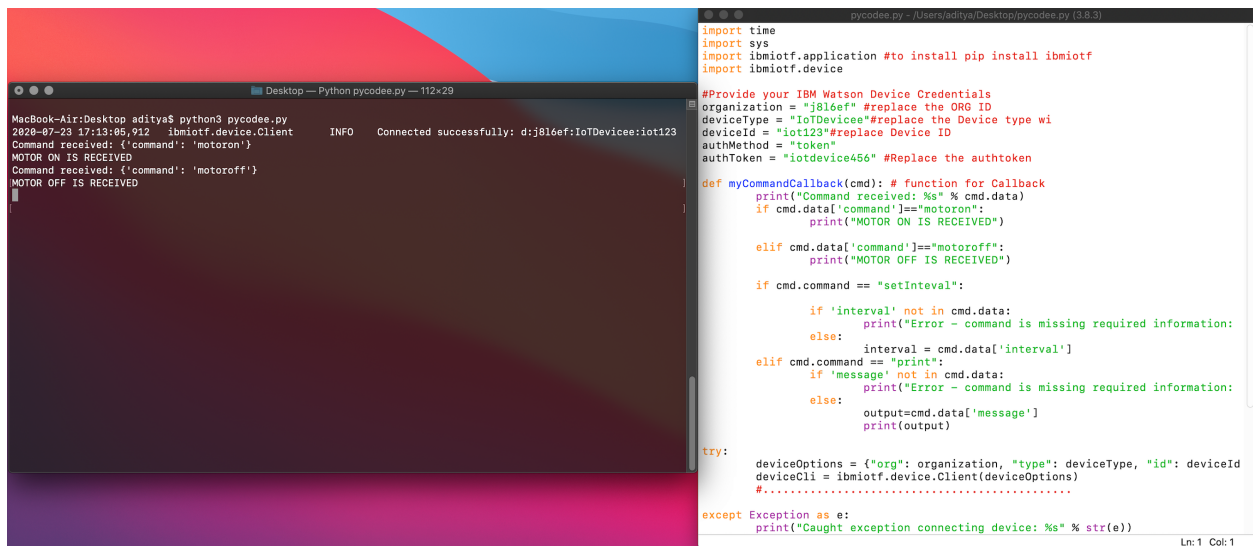


Figure: Showing the Current Location Statistics in Realtime

## Motor Control Through UI and Connected through Python Script:



The image shows a Python script named `pycodee.py` and its terminal output. The script uses the `ibmiotf` library to connect to an IBM Watson IoT device and control a motor. The terminal output shows the script running successfully and receiving commands to turn the motor on and off.

```
import time
import sys
import ibmiotf.application #to install pip install ibmiotf
import ibmiotf.device

#Provide your IBM Watson Device Credentials
organization = "j8l6ef" #replace the ORG ID
deviceType = "IoTDevice" #replace the Device type wi
deviceId = "iot123" #replace Device ID
authMethod = "token"
authToken = "iotdevice456" #Replace the authToken

def myCommandCallback(cmd): # function for Callback
    print("Command received: %s" % cmd.data)
    if cmd.data['command']=="motoron":
        print("MOTOR ON IS RECEIVED")

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

    if cmd.command == "setInterval":
        if 'interval' not in cmd.data:
            print("Error - command is missing required information: ")
        else:
            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}
    deviceCli = ibmiotf.device.Client(deviceOptions)
    #.....
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
```

```
MacBook-Air:Desktop aditya$ python3 pycodee.py
2020-07-23 17:13:05,912 ibmiotf.device.Client INFO Connected successfully: d:j8l6ef:IoTDevice:iot123
Command received: {'command': 'motoron'}
MOTOR ON IS RECEIVED
Command received: {'command': 'motoroff'}
MOTOR OFF IS RECEIVED
```

## 7. ADVANTAGES AND DISADVANTAGES

### Advantages:

- Crop Condition Details
- Weather Forecast
- Remote Monitoring
- Easy To Use UI
- Data Collection
- Analysis of Data
- Remote Motor Control

### Disadvantages:

- Privacy Issue
- Internet Connectivity



## **8. APPLICATIONS:**

By the implementation of IoT based Agriculture System, It not only improves the efficiency of yield produced but also closely monitor and get accurate information about the crop and weather before hand.

It also enables the farmer to take correct decision based on the data collected and analysed. This information/data can also be used by Agriculture Department can give suggestions on better techniques and new methods that the farmers can implement in the future.

This also enables future proofing, so that if any upgradation or modification can be done to the micro controller to add other new automated actions easily.

## **9. CONCLUSION:**

Smart Agriculture System Based On Internet Of Things can deliver the farmer all the required information like temperature, humidity, soil moisture of the crop in realtime and also the weather forecast at fingertips. Also instead of using manual based Motor control, the farmer can do this remotely anywhere as long as he's connected to network.

To make this possible we have used IBM Cloud Platform, Watson IoT Platform, Openweather API and Node-red to gather and show the information on Web Application. By using a Python

Script we were able to subscribe to IBM platform to send and receive commands to motor for controlling it.

Using this Smart Agriculture System the farmer can not only monitor all the required data in realtime but also can make smart decisions for better yield based on the data collected. In this way he can produce yield effectively and also earn profitably more based on accurate data received.

## **10. FUTURE SCOPE:**

Future scope of this smart agriculture system will be to add more sensors to the existing micro controller, to add increase the current functionality or to do more automated tasks like automatic watering system, adding pest control information and geotagging the farm etc. This information can be shared on consent to Government authorities or Private companies for more suggestions of better techniques remotely. As the data stored can be used for reference and analysis which can be very helpful in future.

## **11. BIBLIOGRAPHY:**

1. [https://smartinternz.com/assets/docs/Smart%20Home%20Automation%20using%20IBM%20cloud%20Service%20\(1\).pdf](https://smartinternz.com/assets/docs/Smart%20Home%20Automation%20using%20IBM%20cloud%20Service%20(1).pdf)
2. <https://smartinternz.com/assets/docs/Smart%20Home%20Automation%20using%20IBM%20cloud%20Service>

- s%20(1).pdf
3. <https://openweathermap.org/>
  4. <https://smartinternz.com/assets/docs/Sending%20Http%20request%20to%20Open%20weather%20map%20web%20site%20to%20get%20the%20weather%20forecast.pdf>
  5. <https://www.youtube.com/watch?v=cicTw4SEdxk>
  6. [https://smartinternz.com/assets/docs/Smart%20Home%20Automation%20using%20IBM%20cloud%20Service%20\(1\).pdf](https://smartinternz.com/assets/docs/Smart%20Home%20Automation%20using%20IBM%20cloud%20Service%20(1).pdf)
  7. <https://github.com/rachuriharish23/ibmsubscribe>
  8. [https://en.wikipedia.org/wiki/Agricultural\\_Research\\_Service\\_\(ICAR,\\_India\)](https://en.wikipedia.org/wiki/Agricultural_Research_Service_(ICAR,_India))

## **12. APPENDIX:**

### **A. SOURCE CODE:**

#### **1) Node-red Flow**

[https://github.com/SmartPracticeschool/IISPS-INT-3287-Smart-Agriculture-system-based-on-IoT/blob/master/All\\_Flows.json](https://github.com/SmartPracticeschool/IISPS-INT-3287-Smart-Agriculture-system-based-on-IoT/blob/master/All_Flows.json)

#### **2) Python Script**

```
1 import time
2 import sys
3 import ibmiotf.application #to install
  pip install ibmiotf
4 import ibmiotf.device
5
6 #Provide your IBM Watson Device
  Credentials
7 organization = "j8l6ef" #replace the ORG
  ID
8 deviceType = "IoTDevicee"#replace the
  Device type wi
9 deviceId = "iot123"#replace Device ID
10 authMethod = "token"
11 authToken = "iotdevice456" #Replace the
  authtoken
12
13 def myCommandCallback(cmd): # function
  for Callback
14     print("Command received: %s" %
    cmd.data)
15     if
    cmd.data['command']=="motoron":
16         print("MOTOR ON IS
```

```
    RECEIVED")
17         elif
    cmd.data['command']=="motoroff":
18             print("MOTOR OFF IS
    RECEIVED")
19         if cmd.command == "setInteval":
20             if 'interval' not in
    cmd.data:
21                 print("Error -
    command is missing required information:
    'interval'")
22             else:
23                 interval =
    cmd.data['interval']
24         elif cmd.command == "print":
25             if 'message' not in
    cmd.data:
26                 print("Error -
    command is missing required information:
    'message'")
27             else:
28
    output=cmd.data['message']
29                 print(output)
```

```
30
31 try:
32     deviceOptions = {"org": organization,
33                       "type": deviceType, "id": deviceId,
34                       "auth-method": authMethod, "auth-token":
35                       authToken}
36     deviceCli =
37         ibmiotf.device.Client(deviceOptions)
38
39     # .....
40     .....
41
42 except Exception as e:
43     print("Caught exception connecting
44           device: %s" % str(e))
45     sys.exit()
46
47 # Connect and send a datapoint "hello"
48 # with value "world" into the cloud as an
49 # event of type "greeting" 10 times
50 deviceCli.connect()
51
52 while True:
53     deviceCli.commandCallback =
```

```
myCommandCallback
```

```
45
```

```
46 # Disconnect the device and application  
    from the cloud
```

```
47 deviceCli.disconnect()
```