

# **SMART AGRICULTURE SYSTEM BASED ON IoT**

## **1.INTRODUCTION**

### **OVERVIEW:**

India is an Agri-based economy at the core. More than 70% of the population depend on agriculture. Due to lack of adaptation of technology, the farmers are not gaining as much as profit as they can. This project involves building a smart Internet of Things based agriculture system to monitor the weather conditions and soil conditions and help the farmer to gain better yield. This will be accomplished by using the IBM Watson IoT platform and Openweather API. We use Python language to interact with the system. The highlighting features of this project include smart irrigation with smart control based on real time field data. Secondly temperature maintenance, humidity maintenance and other environmental parameters. And finally the recommendation to farmer for smart agriculture

### **PURPOSE:**

The aim of the project is to make farming affordable and profitable to all by making use of the technological developments like cloud, smartphones and IoT. By measuring and giving the farmer information about the soil and weather, a farmer can know the crop condition without going to the field.

## **2.LITERATURE SURVEY**

### **EXISTING PROBLEM:**

Indian agriculture is plagued by several problems; some of them are natural and some others are manmade. Of those, being unable to predict the climate and plant requirements is the major problem. Also some of the other problems include soil erosion, lack of mechanisation, irrigation. Soil quality testing is not done effectively in India. Moreover, not being able to adapt to technological trends has become fatal to agriculture.

### ***PROPOSED SOLUTION:***

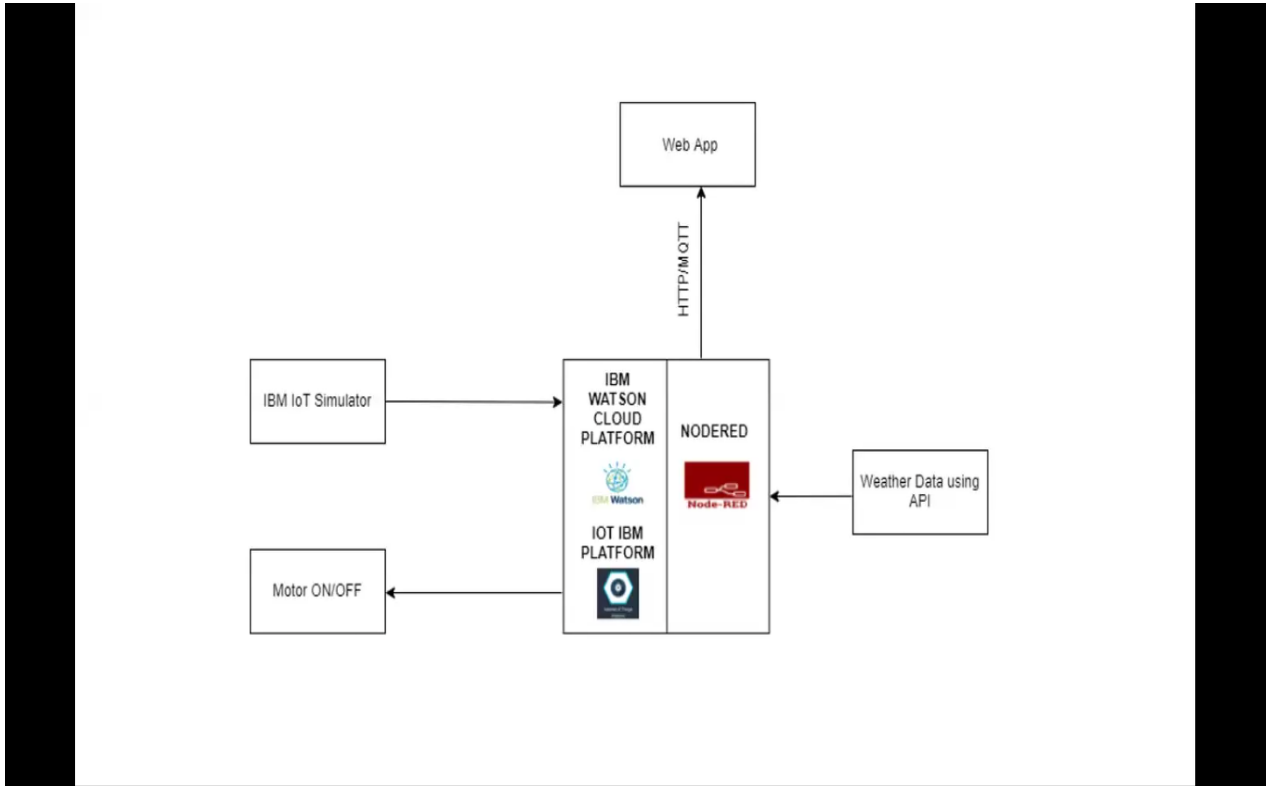
There are many new farming techniques like precision farming, GSM based farming and smart irrigation control. Here, we propose a solution using cloud and IoT to monitor the soil and weather conditions. We use temperature, humidity and moisture sensors to gain the required information and pass them to the cloud platform. Also, we create a web-based interface to interact with the farmer and control the irrigation system. Based on the information from the sensors, the farmer can take a decision on how to control the irrigation system.

### ***3. THEORETICAL ANALYSIS***

#### ***PROJECT SCOPE:***

We create a device in the IBM Watson IoT platform and enable simulation. The simulation is done in the Watson IoT sensor simulator. The sensors take readings every minute and upload to the cloud. Node-RED is used to wire together the hardware, online services and APIs. To simulate weather information, we create an account in OpenWeather.org and provide through the sensors. Later, these are used through a web interface to control the motor.

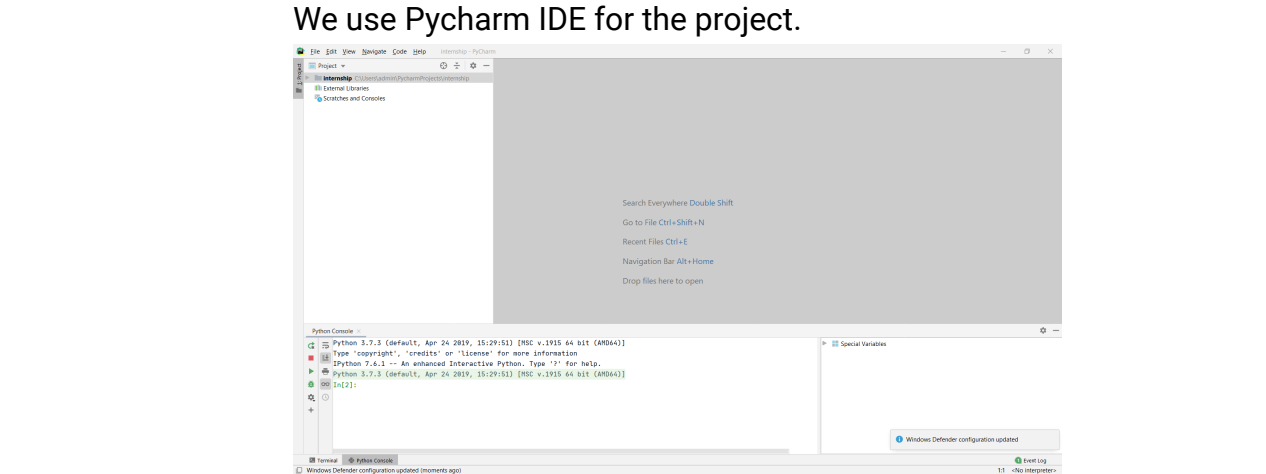
#### ***BLOCK DIAGRAM:***



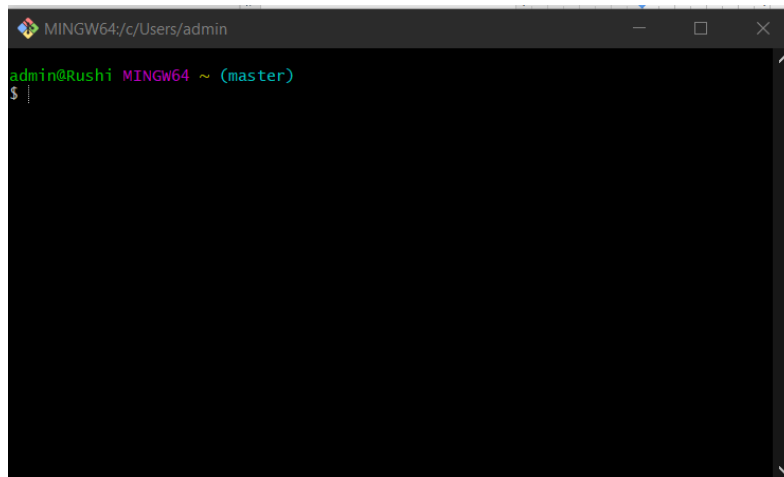
1. Install the required tools and create the required accounts.

### SETTING UP IDE:

We use Pycharm IDE for the project.



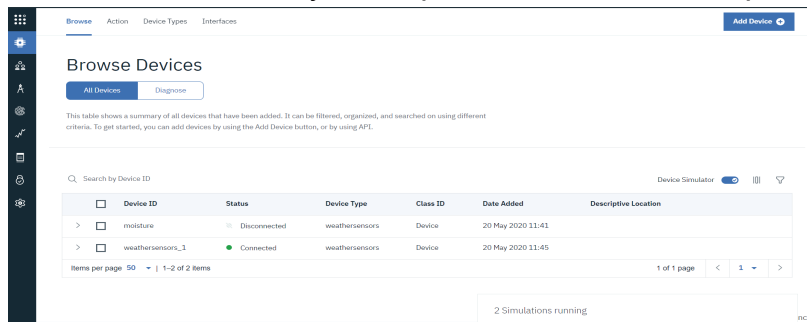
## INSTALL GIT:



```
MINGW64/c/Users/admin
admin@Rushi MINGW64 ~ (master)
$
```

## 2.CREATE A DEVICE IN THE IBM WATSON IOT PLATFORM:

We need to create two device in the platform . One acts as a processor for the sensor information and the other is an instance of a motor. Also create an API key-token pair and save in a safe place.



Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
> moisture	Disconnected	weathersensors	Device	20 May 2020 11:41	
> weathersensors_1	Connected	weathersensors	Device	20 May 2020 11:45	

1 of 1 page

2 Simulations running

## 3.INSTALL NODE-RED LOCALLY:

Node-red is a Node.js based implementation and can be installed using chocolatey or yarn.

```
node-red
21 May 10:32:51 - [info] Node-RED version: v1.0.6
21 May 10:32:51 - [info] Node.js version: v14.2.0
21 May 10:32:51 - [info] Windows_NT 10.0.17763 x64 LE
21 May 10:32:54 - [info] Loading palette nodes
21 May 10:32:55 - [info] Settings file : Users\admin\node-red\settings.js
21 May 10:32:55 - [info] Context store : 'default' [module=memory]
21 May 10:32:55 - [info] User directory : Users\admin\node-red
21 May 10:32:55 - [warn] Projects disabled : editorTheme.projects.enabled=false
21 May 10:32:55 - [info] Flows file : Users\admin\node-red\flows_Rushi.json
21 May 10:32:55 - [info] Creating new flow file
21 May 10:32:55 - [warn]

-----
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
-----
21 May 10:32:55 - [info] Starting flows
21 May 10:32:55 - [info] Started flows
```

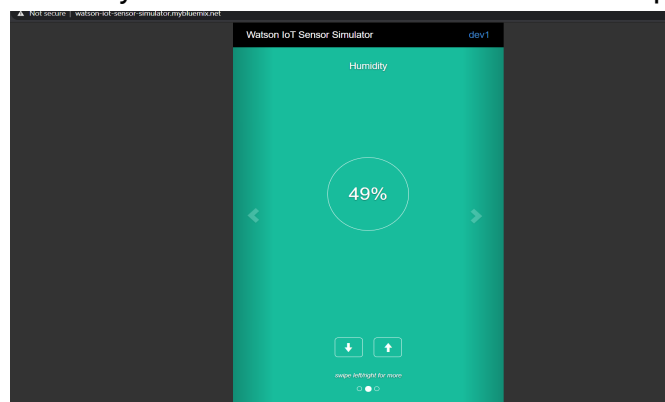
## 4.CONNECT THE IOT DEVICE TO A SIMULATOR

The iot sensor simulator requires the following information.

- 1.Device organization
- 2.Device Type
- 3.Device Id
- 4.Device Token

Once given all the required instructions, the sensor is connected to the device we created.

This can be verified by the device name in blue colour on top right.



## 5.VISUALISING THE DATA:

- Once the device is connected, we can create boards in the IBM IoT platform.

Boards-->Create new board--> Add Cards

Different types of visualizations like line plot,gauge can be done in the IBM platform.

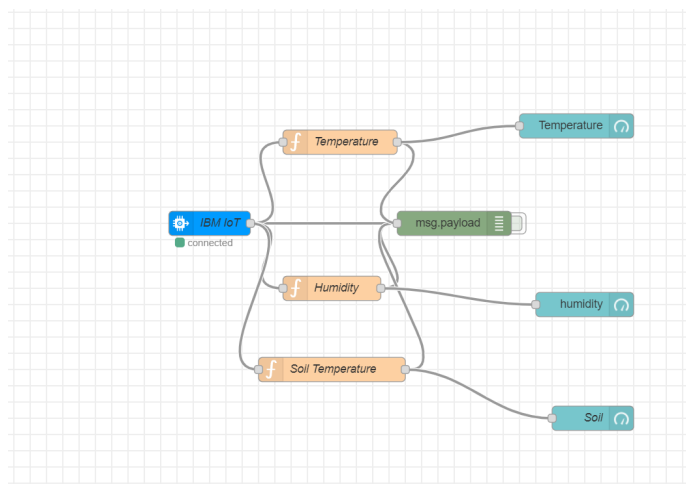
## 6.CONNECTING TO OPENWEATHER API:

- Create an account in openweather.org
- Go to API marketplace and register a API key
- Select "By city Name" in API options

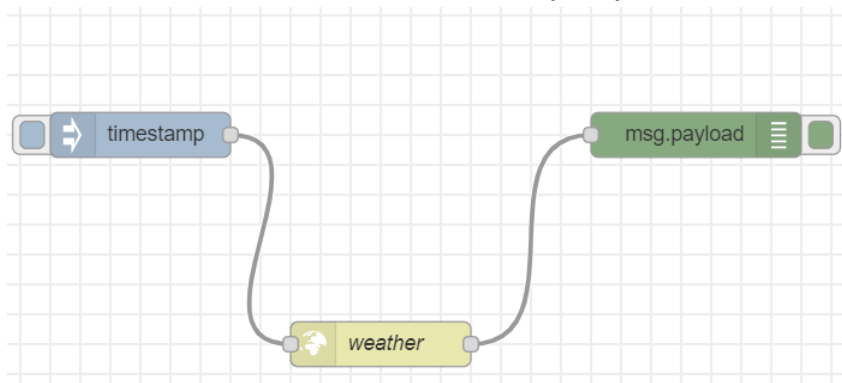
- Copy and save the API call to be used later

## 7.CREATING THE UI USING NODE-RED:

- We need to create flows for our purposes.
- We need to install the ibm iot node package and ibm watson package. To install a new node-set,  
**manage palette-->install-->search for required nodes**
- Firstly, to take the input data from the sensor we make the following flow .
- This also creates a UI for the user to interact with the data and the devices.



- To view the data from API,we use a http request node and inject node.

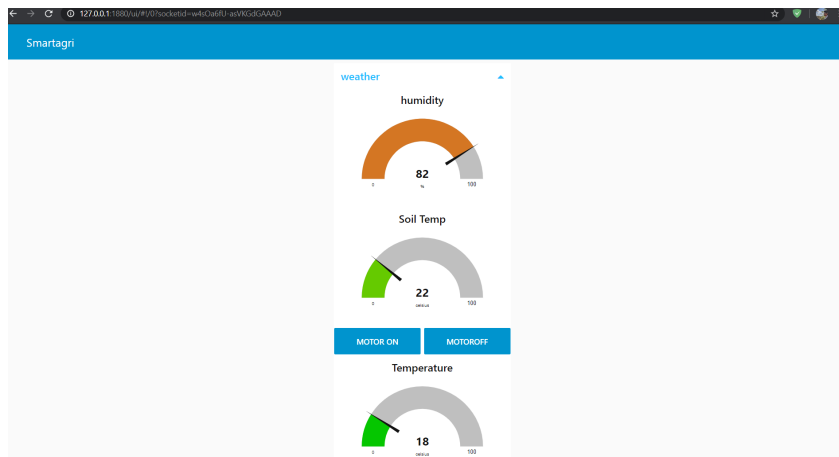


- To give the instructions to the device,in our case motor we create another flow using the ibm out node and Motor ON and OFF buttons.

## 8.RUNNING THE PYTHON CODE AND TURNING MOTOR ON OR OFF:

We use the python programming language to interact with the devices and the cloud. On clicking the button in the UI,the motor can be made ON or OFF.

## RESULTS:



```
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
Command received: { 'command': 'motoron' }
MOTOR ON
{ 'Status': 'Sensor is ON' } to IBM Watson
CALL MADE
{ 'Status': 'Sensor is ON' } to IBM Watson
Command received: { 'command': 'motoroff' }
MOTOR OFF
{ 'Status': 'Sensor is ON' } to IBM Watson
CALL MADE
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
{ 'Status': 'Sensor is ON' } to IBM Watson
```

## FUTURE SCOPE:

The project can be further extended to enabling the usage of AI in the agriculture ecosystem. It helps a lot in implementation of Precision farming. We can suggest crops based on the climatic conditions of the data. Based on the water level, we can alert the farmer or automatically turn the motor off.