Delivery of Sprint-1

Project	Smart Farmer-IoT Enabled Smart Farmer Application
Team ID	PNT2022TMID25834

1. Introduction

The project aims at making use of evolving technology that is smart agriculture using automation. Monitoring environmental conditions is the major factor to improve yield of the efficient crops. As the world is trending into new technologies and implementations it is a necessary goal to trend up in agriculture also. Much research is done in the field of agriculture. In this regard, precision agriculture, automated irrigation scheduling, optimization of plant growth, farm land monitoring, and farming production process management in crops, are among a few key applications.

2. Problem Statement

Most projects signify the use of wireless sensor networks to collect data from different sensors deployed at various nodes and send it through the wireless protocol. The collected data provide the information about the various environmental factors. Monitoring the environmental factors is not the complete solution to increase the yield of crops. There are a number of other factors that decrease productivity to a greater extent. Hence automation must be implemented in agriculture to overcome these problems.

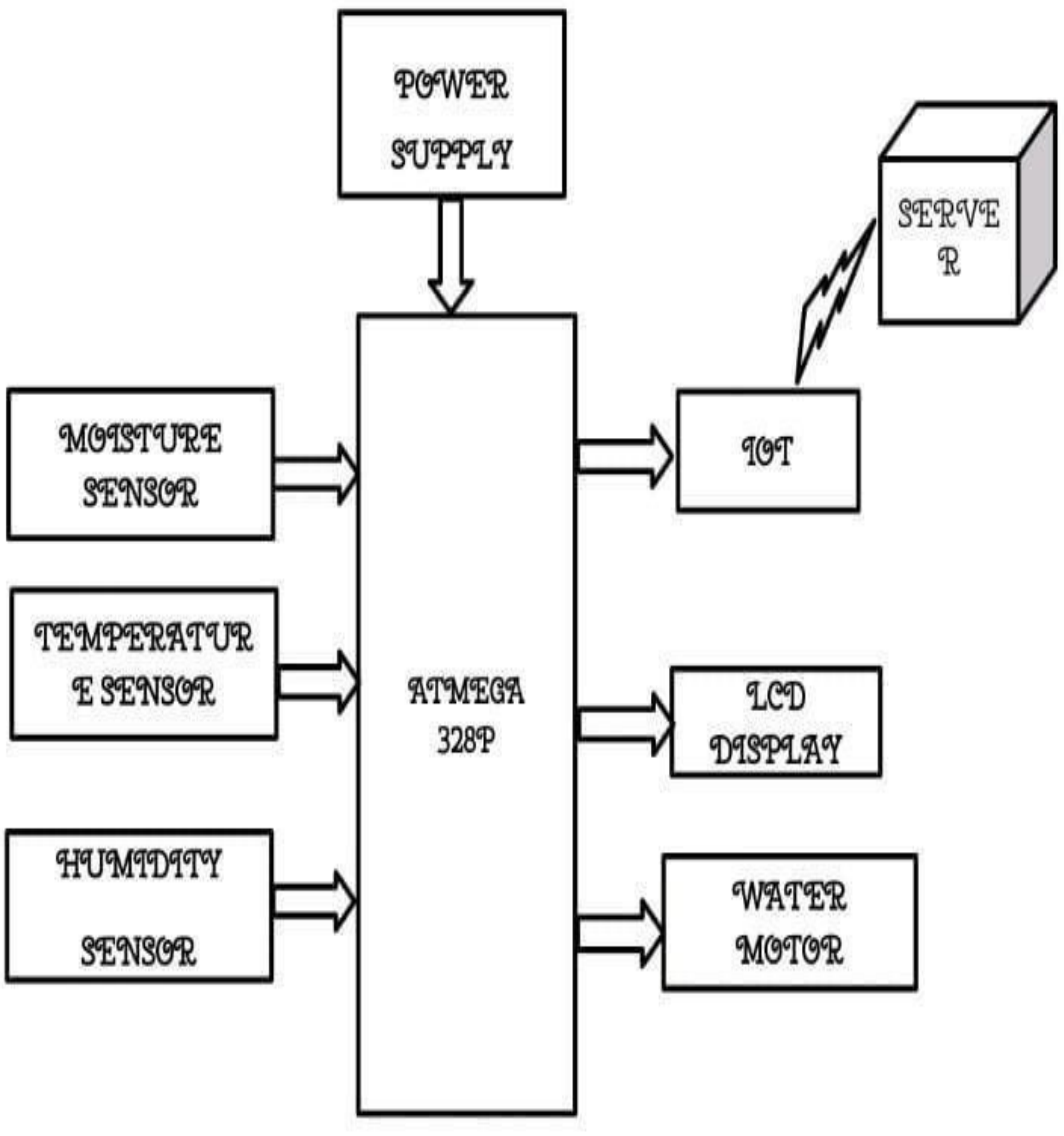
3. Proposed Solution

Proposes the automatic irrigation system using Arduino for smart crop field productivity. This system consists of sensor like moisture sensor, temperature sensor and humidity sensor. Moisture sensor used for detecting the moisture content in soil. The goal of the implementation is to exhibit the microcontroller's clever and intelligent capabilities by allowing decisions on watering the plants to be made based on constant monitoring of the field's environmental conditions.

4. Theoretical Analysis

4.1 Block Diagram

In order to implement the solution, the following approach as shown in the block diagram is used

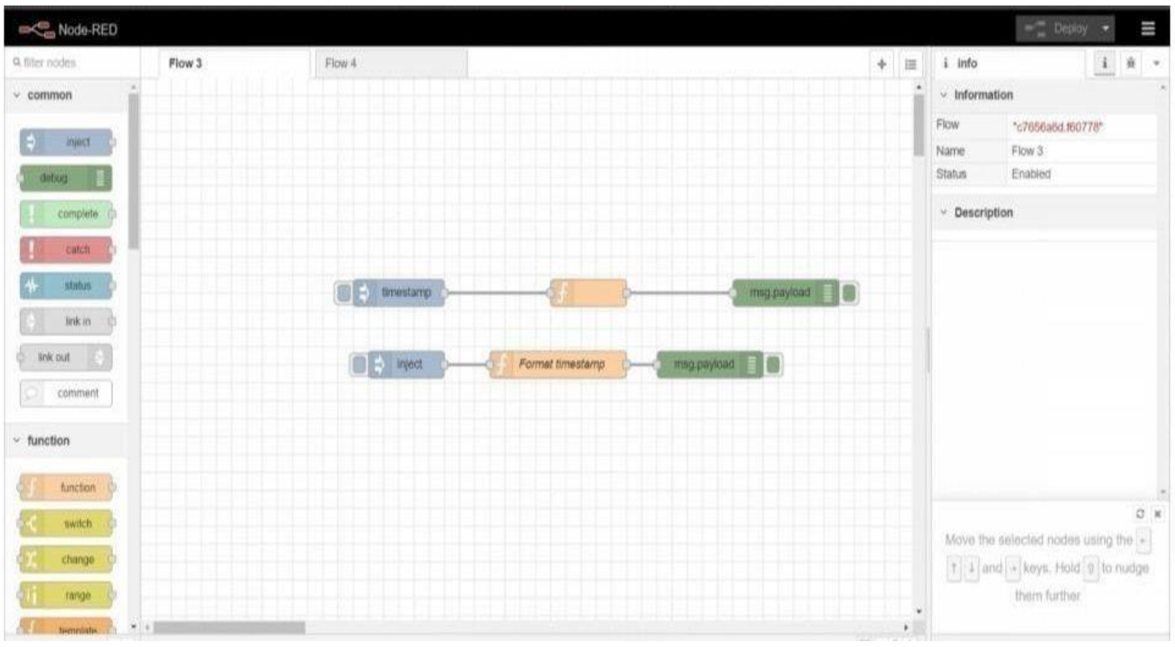


4.2 Required Software Installation

4.2.A Node-Red

Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.

Installation:



- First install npm/node.js
- Open cmd prompt
- Type => npm install node-red

To run the application :

- Open cmd prompt
- Type=>node-red
- Then open http://localhost:1880/ in browser

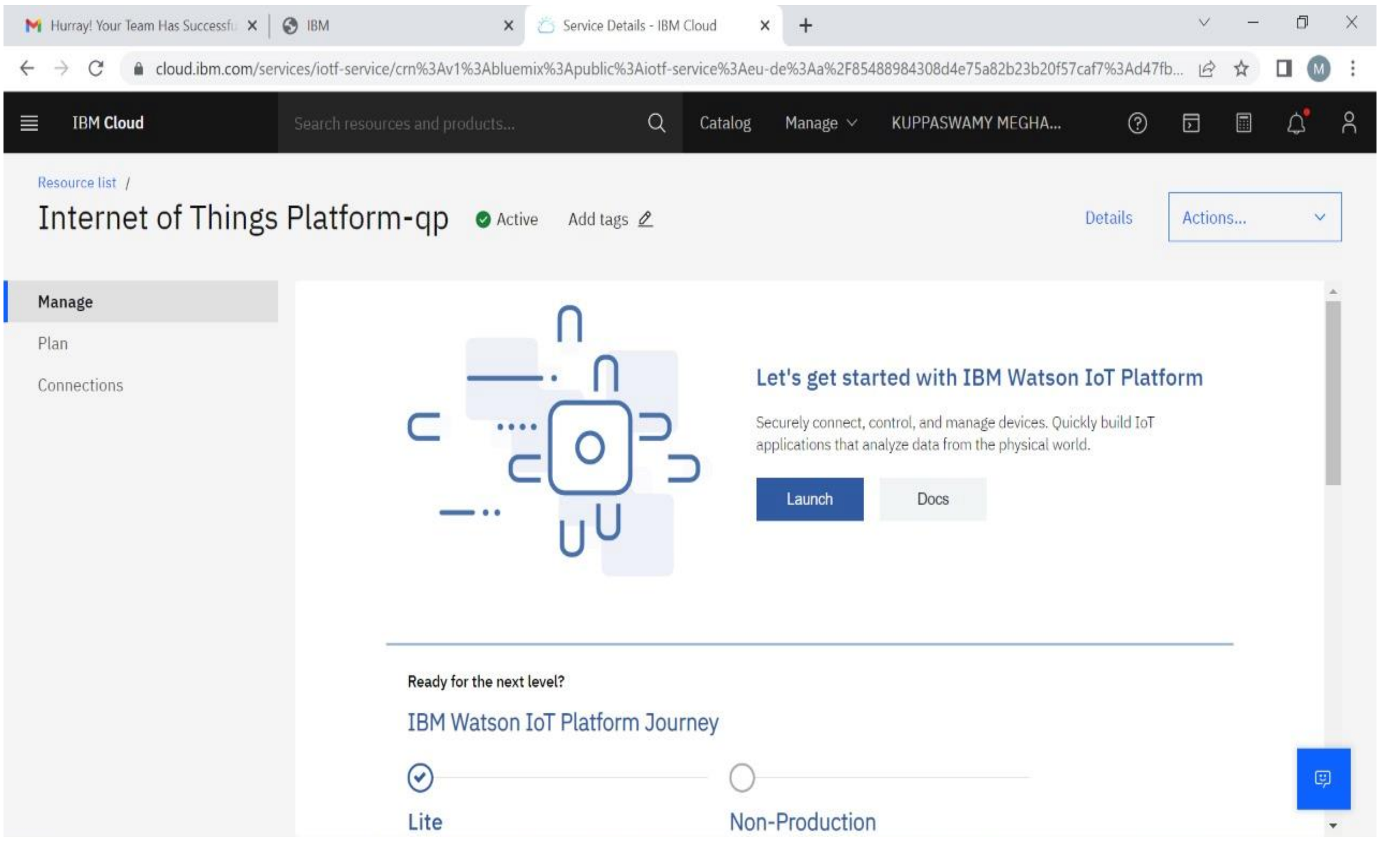
Installation of IBM IoT and Dashboard nodes for Node-Red

In order to connect to IBM Watson IoT platform and create the Web App UI these nodes are required 1. IBM IoT node

2. Dashboard node

4.2.B IBM Watson IoT Platform

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization and data storage. IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices.


Hurray! Your Team Has Successfu
IBM
Service Details - IBM Cloud
cloud.ibm.com/services/iotf-service/crn%3Av1%3Abluemix%3Apublic%3Aiotf-service%3Aeu-de%3Aa%2F85488984308d4e75a82b23b20f57caf7%3Ad47fb...
IBM Cloud
Search resources and products...
Catalog
Manage
KUPPASWAMY MEGHA...
Resource list /
Internet of Things Platform-qp
Active
Add tags
Details
Actions...
Manage
Plan
Connections
Launch
Docs
Lite
Non-Production

Let's get started with IBM Watson IoT Platform

Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.

Ready for the next level?

IBM Watson IoT Platform Journey

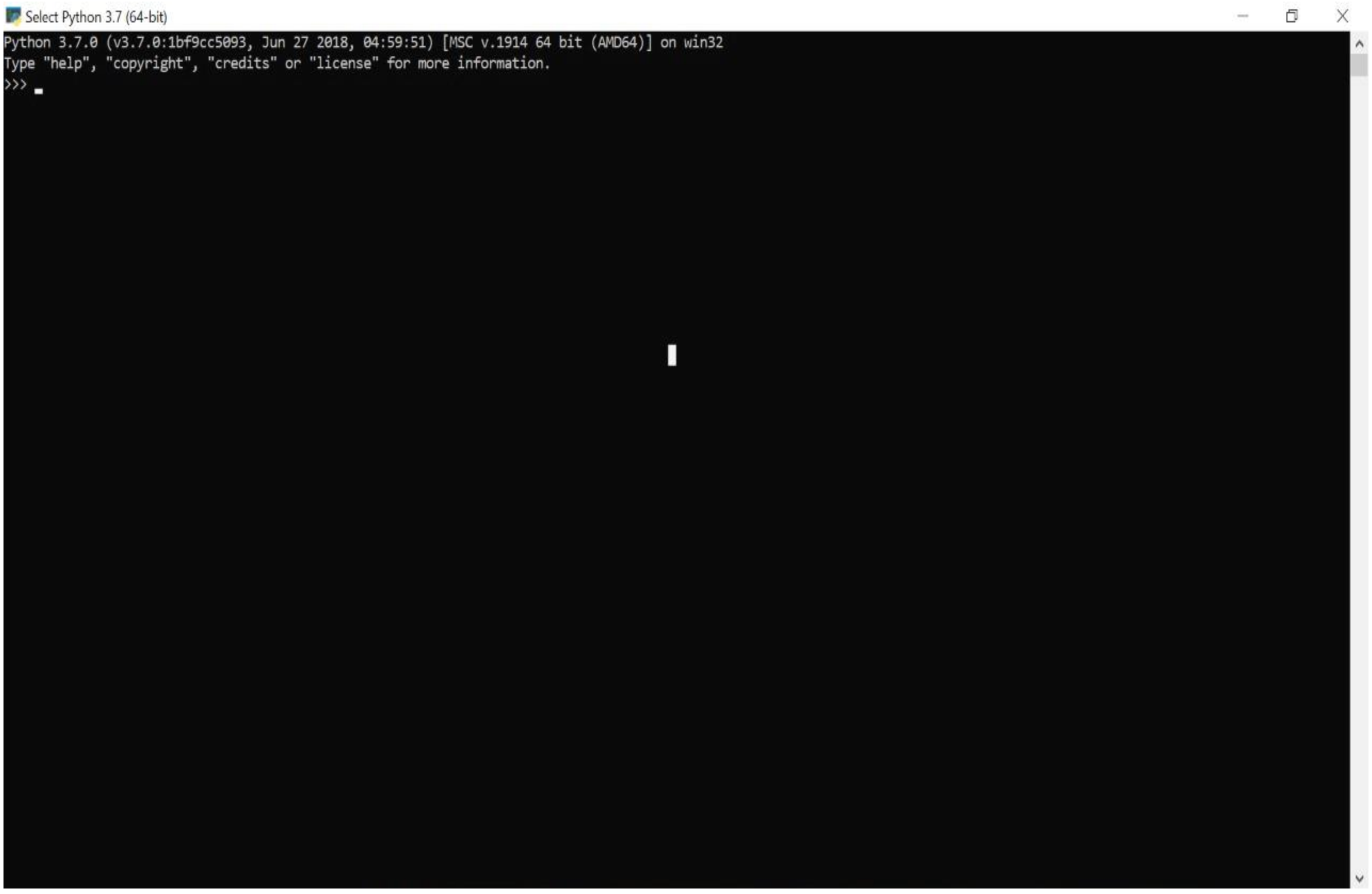
Steps to configure:

- Create an account in IBM cloud using your email ID
- Create IBM Watson Platform in services in your IBM cloud account
- Launch the IBM Watson IoT Platform
- Create a new device
- Give credentials like device type, device ID, Auth. Token
- Create API key and store API key and token elsewhere.

4.2.C Python IDE

Install Python3 compiler

Install any python IDE to execute python scripts, in my case I used Spyder to execute the code.



```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Code:

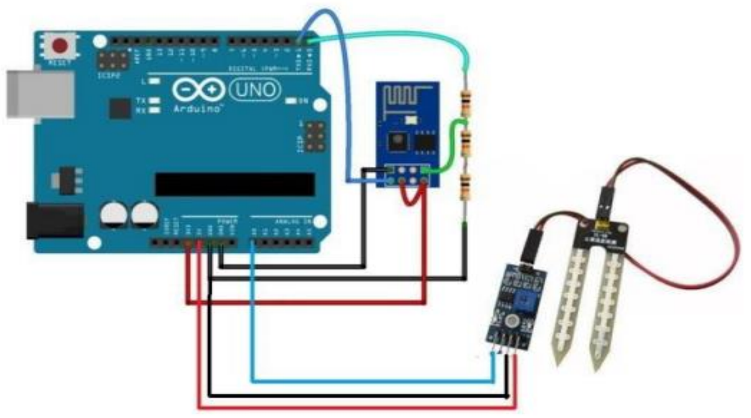
```
import time import sys
import ibmiotf.application
importibmiotf.device
import random


#Provide your IBM Watson Device Credentials
organization = "157uf3" deviceType = "abcd"
deviceId = "7654321" authMethod = "token"
authToken = "87654321"


# Initialize GPIO

def myCommandCallback(cmd): print("Command
received: %s" % cmd.data['command'])
status=cmd.data['command'] if status=="motoron":
print ("motor is on")
elif status == "motoroff":
print ("motor is off") else :
print ("please send proper command")

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
 event of type "greeting" 10 times deviceCli.connect()

 while True:
 #Get Sensor Data from DHT11

 temp=random.randint(90,110)
 Humid=random.randint(60,100)Mois=random.randint(20,120)
 data = { 'temp' : temp, 'Humid': Humid, 'Mois' :Mois}
 #print data def
 myOnPublishCallback():
 print ("Published Temperature = %s C" % temp, "Humidity = %s
 %%" % Humid, "Moisture =%sdeg c" %Mois, "to IBMWatson")

 success=deviceCli.publishEvent("IoTSensor","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()
```

Arduino code for C :

```
#include<LiquidCrystal.h>
LiquidCrystal lcd (13,12,11,10,9,8);
#include <Adafruit_Sensor.h>
#include "DHT.h"
#define DHTPIN A5 // what pin we're connected to
#define DHTTYPE DHT11 // DHT 11
int t;
DHT dht (DHTPIN, DHTTYPE);
int m;
int h;
int a,b;
void setup ()
{
Serial.begin(9600);
// put your setup code here, to run once:
lcd.begin(16,2);
dht.begin();
pinMode (7, INPUT); // Moisture
 pinMode (4, OUTPUT); // motor

 pinMode(A5, INPUT); //temperature

 }
 void loop ()
 {
 dht11();
 if((a==1)||(b==1)) /////motor and buzzer contion
{
 digitalWrite(4, HIGH);
 }
 else if((a==0)||(b == 0))
 {
 digitalWrite(4, LOW);
 }
 int m=analogRead(A0);
 if(m>500)
{
lcd.setCursor(0,1);
lcd.print("Moist_H");
b=1;
}
else{
lcd.setCursor(0,1);
lcd.print("Moist_L");
b=0;
}
Serial.println("T"); //temp
Serial.println(t);
delay (200);
Serial.println("A"); //temp status

Serial.println(a);
delay (200);
Serial.println("H”) ;//Huminity status
Serial.println(h);
delay (200);
Serial.println("M");
Serial.println(m);
delay (200);
Serial.println("B"); //moisture status
Serial.println(b);
delay (200);
}
void dht11()
{
 h = dht.readHumidity();
 t = dht.readTemperature();
 float f = dht.readTemperature(true);
 if (isnan(h) || isnan(t) || isnan(f))
{
 //Serial.println("Failed to read from DHT sensor!");
 return;
```

```
}
 lcd.setCursor(0,0);
 lcd.print("T:");
 lcd.setCursor(2,0);
 lcd.print(t);
 lcd.print(" ");
 lcd.setCursor(8,0);
 lcd.print("H:");
 lcd.setCursor(10,0);
 lcd.print(h);
 lcd.print(" ");
 delay (1000);
 if (t > 38)
{
lcd.setCursor(5,0);
lcd.print("T-H");
//digitalWrite(A2, HIGH);
a=1;
}
else
{
lcd.setCursor(5,0); lcd.print("T-L");
//digitalWrite(A2, LOW);
a=0;
}
if (h > 60)
{
lcd.setCursor(13,0);
lcd.print("H-H");
}
else
{
lcd.setCursor(13,0);
lcd.print("H-L");
}
}
```

Circuit Diagram



4.3 IoT Simulator

In our project in the place of sensors we are going to use IoT sensor simulator which give random readings to the connected cloud.

The link to simulator: https://watson-iot-sensor-simulator.mybluemix.net/

We need to give the credentials of the created device in IBM Watson IoT Platform to connect cloud to simulator.

4.4 OpenWeather API

OpenWeatherMap is an online service that provides weather data. It provides current weather data, forecasts and historical data to more than 2 million customer.

Website link: https://openweathermap.org/guide

Steps to configure:

o Create account in OpenWeather o Find the name of your city by searching o Create API key to your account

o Replace "city name" and "your api key" with your city and API key in below red text

api.openweathermap.org/data/2.5/weather?q={city name}&appid={your api key}