INTERNSHIP REPORT

INTRODUCTION:

1.1 - Overview :
The projects was executed in the following manner -

1. IBM cloud was used to obtain device and API keys for the configuration of nodes.
2. A network containing nodes was made using Node-red.
3. IoT simulator was fed with the device data obtained from IBM cloud.
4. API were generated using the openweather.org.
5. A Web UI was desinged using the node-red.
6. At last, the python code was run in shell to take the commands from Wed UI.

1.2 - Purpose :

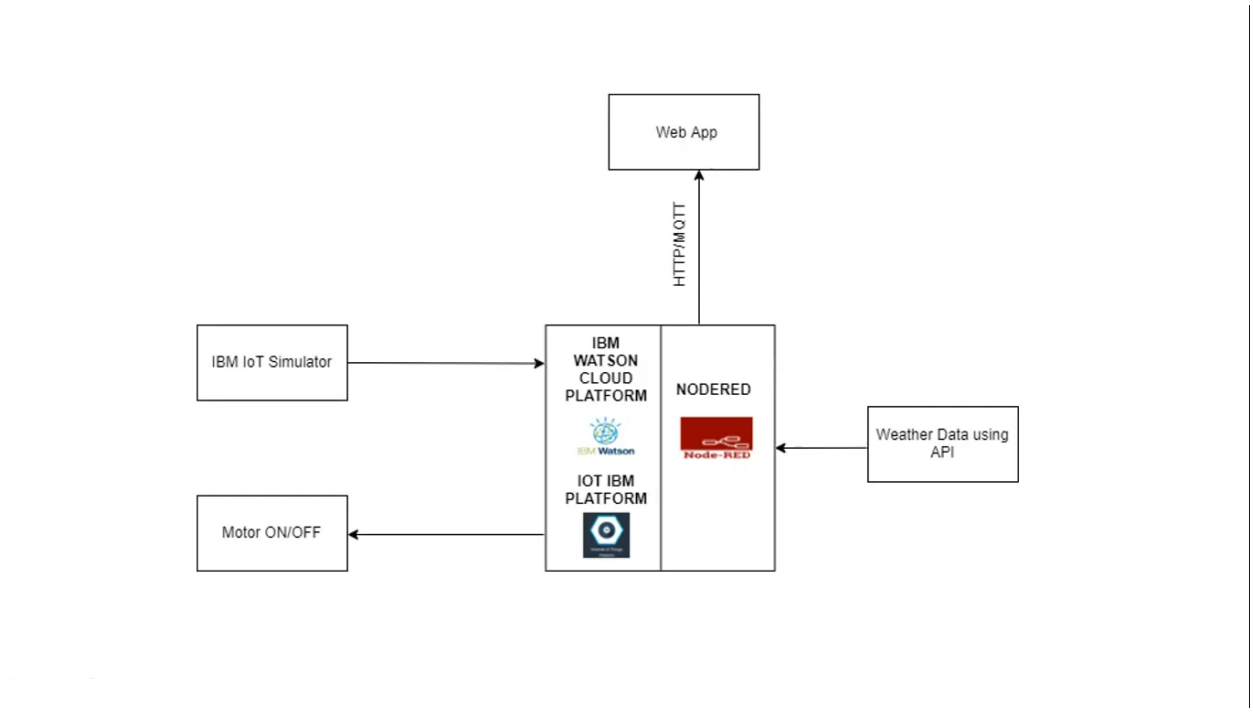
The ease of farming and tracking the environmental changes according to the weather was purpose behind the making of project.

LITERATURE SURVEY:

2.1 - Existing problem : Without the technology of smart agriculture, the farmers had to suffer a lot. In different environment conditions , they have to check the progress of crop physically(at the site).
2.2 - Proposed Solution : By implementing the IoT simulator and connecting the devices to the weather API , farmers can gain the info about the climatic conditions of the filed without physically present there.

THEORITICAL ANALYSIS :

3.1 - Block Diagram : The block diagram of the system is shown below -



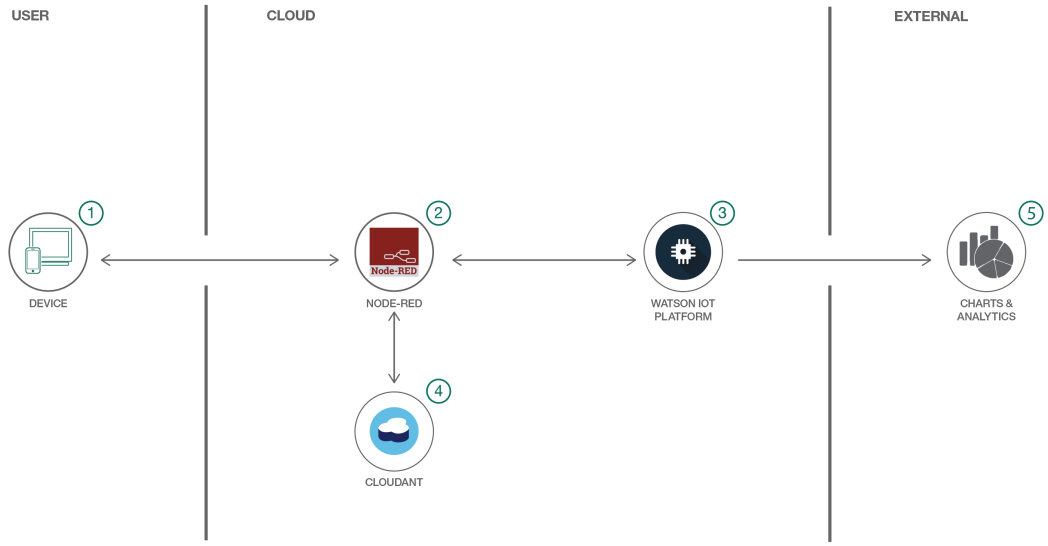
3.2 - Hardware/Software desgining : There was no hardware component involved during the whole project. All the work from designing to simulation was done on softwares. The IBM Cloud(Watson IoT) was used to create device and retrieve the API keys for the input node. The designing part was done by Node-red platform. All the nodes were configured in the node-red. The data was taken from the IBM watson IoT simulator. The API keys for real time weather were generated by openweatherapi.org site. The observation was took from the Web UI built using the Node-red. And finally , the code was run on python shell and commands were ran on it.

EXPERIMENTAL INVESTIGATIONS :

This project's primary research goal is to identify and summarise the IBM cloud and Node-red configuration with respect to IoT technology through analysing data from multiple sources.

FLOWCHART :

INTERNSHIP REPORT



RESULT :

By using the cloud service and API keys , a model is designed to monitor the farms and fields and making it a smart agriculture system using IoT.

ADVANTAGES AND DISADVANTAGES :

The advantages are as follows :

1. There is no need of physical presence of an individual in the farm to check the crops.
2. It can be monitored from anywhere via a device.
3. Time and energy can be saved.

The disadvantages are as follows :

1. As more number of IoT devices are present so there is possibility of interfernce or hacking.
2. Malfunctioning can occur.

APPLICATIONS :

1. Real time weather monitoring
2. Smart farming technique

CONCLUSION :

Overall , the project is made using the IBM Watson IoT simulator and Node-red for Smart Agriculture for the betterment of mankind.

FUTURE SCOPE :

The IoT has a great future ahead . As the years are increasing, more number of IoT devices are being connected worldwide. The traditional farming will be replaced by the Smart Agriculture using the IoT technology.

BIBILOGRAPHY :

1. Documentation - raspberrypi.org
2. Documentation - cloud.google.com
3. Getting started with the Internet of Things by Cuno Pfister

APPENDIX :

Source Code - :

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

#Provide your IBM Watson Device Credentials
organization = "nqqcvt" #replace the ORG ID
deviceType = "nodemcu"#replace the Device type
deviceId = "1999"#replace Device ID
authMethod = "token"
authToken = "19991999" #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: '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:
                        output=cmd.data['message']
                        print(output)

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:
        deviceCli.commandCallback = myCommandCallback

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