Automatic pet feeder using iot

Team Name : ECE_C06

1) M.Pavan rohith gandhi(18481A04D9)

2) M.Basha(18481A04D5)

3) M.Likhitha parameswari (18481A04D8)

4) M.Amulya (18481A04D6)

5)M.Jahnavi(18481A04D7)

<u>TABLE OF CONTENTS:</u>

1 INTRODUCTION:

1.1 Overview:

- Household pets need special treatment and care. Owners need to ensure food, drinks, and medication are served as at when due.
- Lack of adequate attention to pets' needs might have great consequential effects, such as starvation, ill health, among others. Due to concurrent tasks demanding owners attention, couple with busy life style, management of these pets may not be as simple as expected.
- Hence, the need to mi-grate from manual to technology-based management of pets daily needs. An Internet of Things (IoT) based automatic feed-er system comes handy to assist in the management of pets needs.
- The latter technology will enable pet owners to remotely manage critical needs that are automatable while engaged in other time and attention demanding tasks.

1.2 PURPOSE:

- Automatic Dog Feeder:- An automatic dog feeder allows dogs be fed every day at

specified times.

- The feeder can feed a dog up to five times a day this can be programmed in by the owner

of the dog feeder.

- The automatic feeder will be able to feed dry food and will also supply constant water for the dog

2. Literature Survey:

2.1 Existing problem:

Generally household pets need special treatment and care. Owners have to feed that pet on time.

But sometimes if owners are not able to feed their pets.

With the help of this automatic pet feeder, we can overcome this issue. There is no need of owners to feed their pets. At anytime this automatic pet feeder is used.

2.2 Proposed Solution:

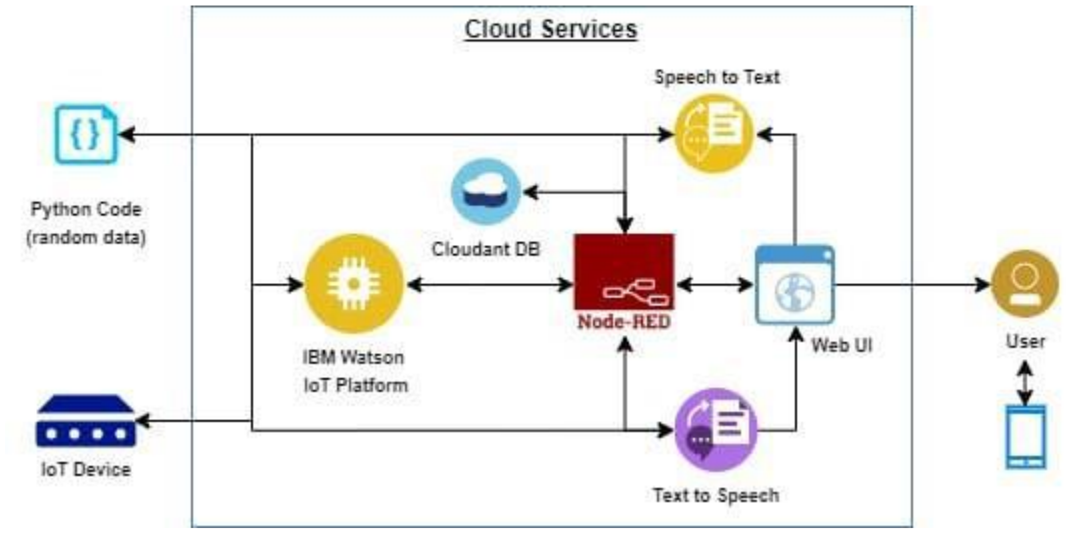
☆ IoT-based feeder system can be designed in a way that it dispenses precise amount of food or other provisions at specific time intervals, reduce the amount of time owners spend on feeding an monitoring of household pets.

☆ Asides the benefits automatic pet feeders give its users, it can also regulate the amount of food given to pets. Since it can be programmed to dispense specific amount of food, thereby ensuring pets are not malnourished or overfed which may lead to obesity especially when the pets are still young.

3 THEORETICAL ANALYSiS:

3.1 BLOCK DIAGRAM:

Fig (1): block diagram of automatic pet feeder



3.2 Hardware/Software Designing:

➢ **Software Design:**

➢ Installation of Node Red Installation of python idle

➢ Installation of Text to Speech (TTS) along with their App URL and their respective API Keys.

➢ Installation of cloud ant database service.

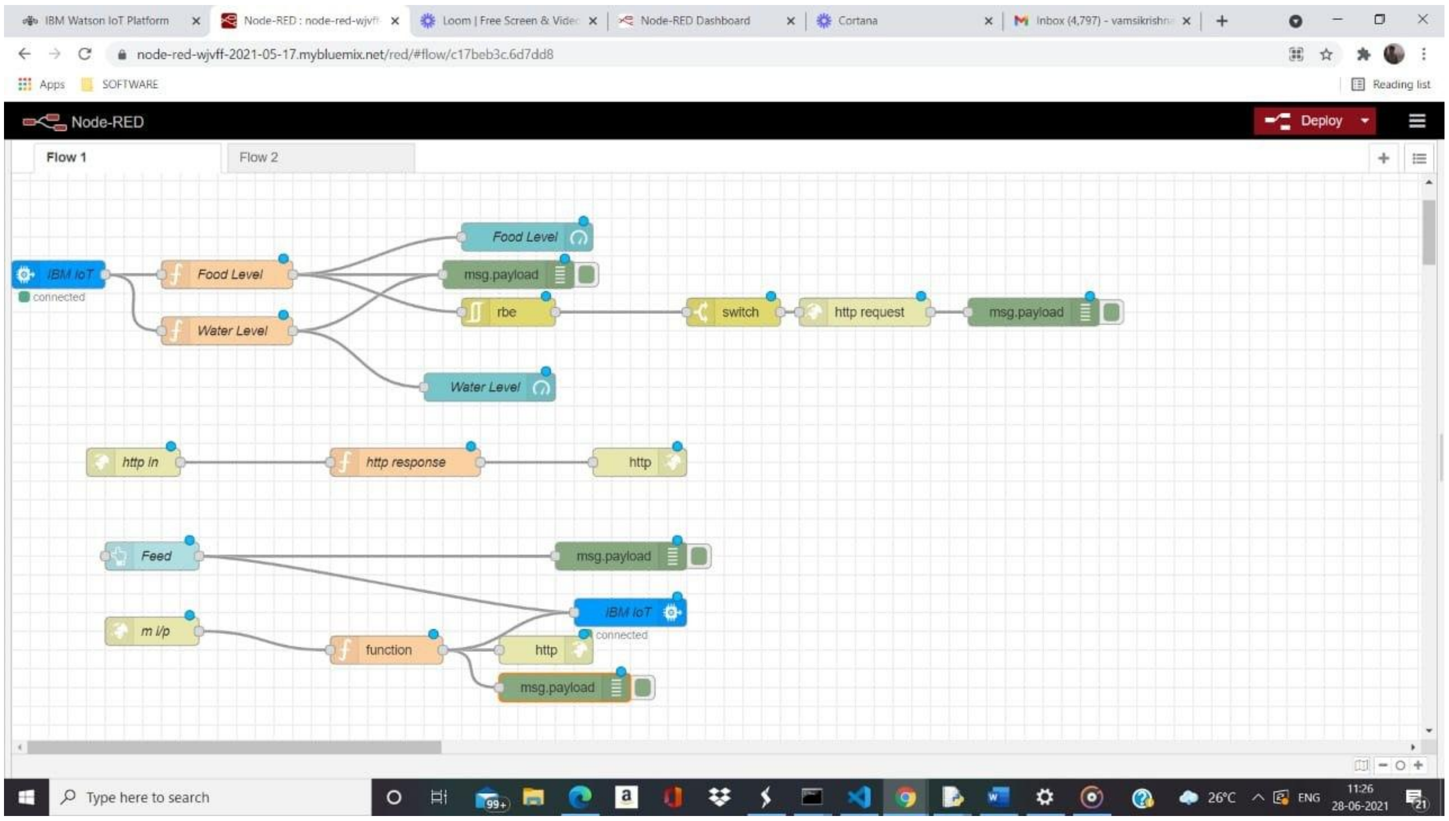
4 EXPERIMENTAL INVESTIGATIONS :

The term “IoT” stands for the Internet of Things and it can be defined as the interconnection between the individually identifiable embedded computing apparatus in the accessible internet infrastructure. ‘IoT’ connects various devices and transportations with the help of internet as well as electronic sensors. Automatic pet feeder is an IoT based device which is capable of automating the pet feeder process by analysing the level of food and the level of water .Also the data of sensors will be displayed on the developed Mobile application. In order to develop the iot based automatic pet feeder we have to use node red, mit app inventer platforms .First of all we have to write the code in python code idle according to the requirements of the project .

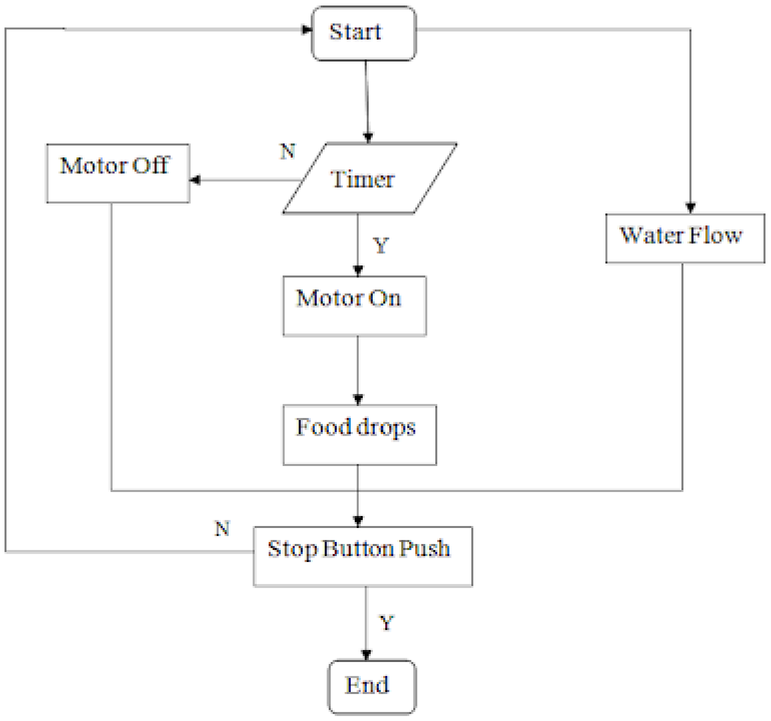
We have compile and run that code using python code idle. After the execution of the code in python code idle.By using mit app inventer we have develop a mobile application Which is used to give the information about the level of water and quantity of the food.

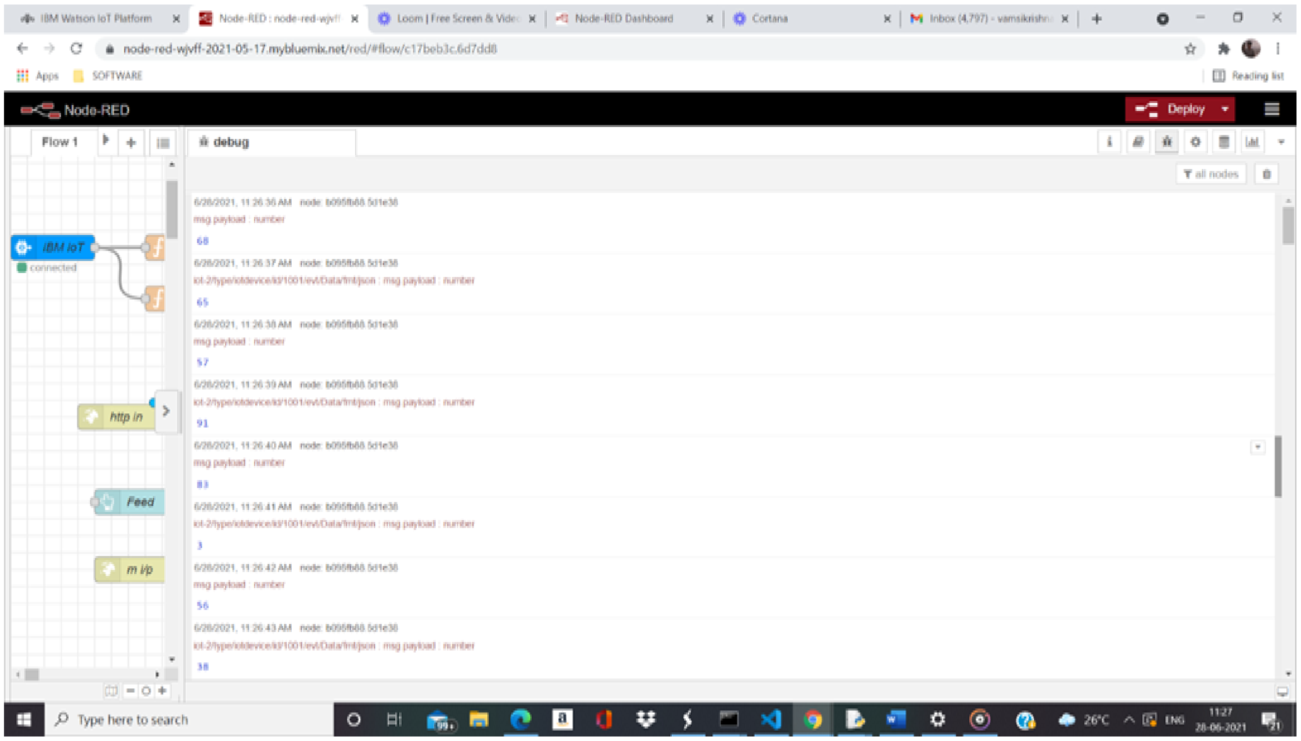
By using this mobile application we get a notification to our mobile about the level of water and quantity of food. In node red we have give the necessary connections according to the requirements of the project.

fig(1):node red



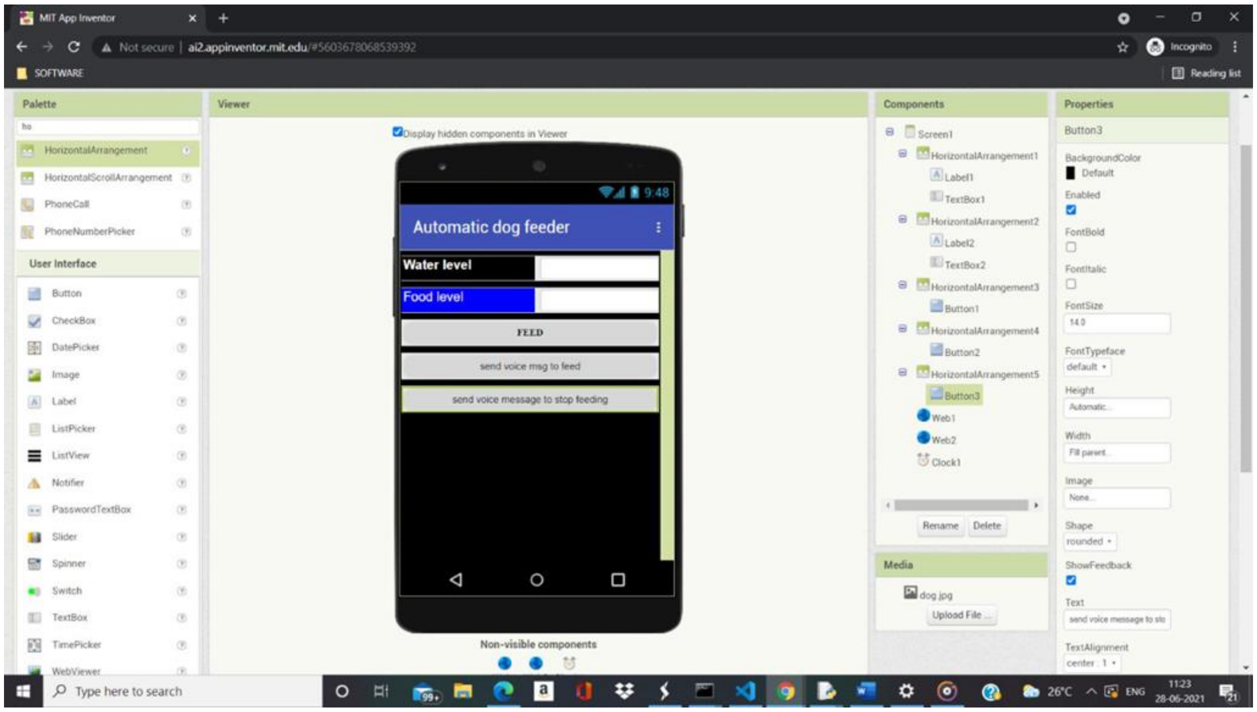
5.FLOW CHART:





```
6/28/2021, 11:26:36 AM   node: b095fb88.5d1e38
msg.payload : number
68
6/28/2021, 11:26:37 AM   node: b095fb88.5d1e38
iot-2/type/iotdevice/id/1001/evt/Data/fmt/json : msg.payload : number
65
6/28/2021, 11:26:38 AM   node: b095fb88.5d1e38
msg.payload : number
57
6/28/2021, 11:26:39 AM   node: b095fb88.5d1e38
iot-2/type/iotdevice/id/1001/evt/Data/fmt/json : msg.payload : number
91
6/28/2021, 11:26:40 AM   node: b095fb88.5d1e38
msg.payload : number
83
6/28/2021, 11:26:41 AM   node: b095fb88.5d1e38
iot-2/type/iotdevice/id/1001/evt/Data/fmt/json : msg.payload : number
3
6/28/2021, 11:26:42 AM   node: b095fb88.5d1e38
msg.payload : number
56
6/28/2021, 11:26:43 AM   node: b095fb88.5d1e38
iot-2/type/iotdevice/id/1001/evt/Data/fmt/json : msg.payload : number
38
```

fig(2):output of node red



fig(3):mobile application

6. Advantages:

▪ Keeps pets hydrated and healthy.

▪Feeders can schedule multiple meals in advance.

▪Instead of the pet having huge amounts of food at long intervals, they can get many small amounts.

▪ You get to save on precious time. 5. The pet always gets fresh food.

▪ Feeders can plan their schedule without much worry that their pet will go hungry.

▪ An automatic feeder will help to minimise overfeeding by measuring out the portions.

7.APPLICATIONS:

➢ Automatic pet feeders help to provide proper weight management by giving your pet the portioned feedings they need.

➢ If you have a pet who gulps his food or eats too fats, a feeder can help him slow down .

➢ The slow feed option with the new simply feed pet feeder dispenses each meal over a 15-minute of period.

8. CONCLUSION:

This design of an loT based automatic pet feeder system was done in consideration of some factors such as: economic application, user convenience, availability of components and research materials, efficiency, compatibility, portability and durability. As earlier stated, this work aims to enhance the management of pets, giving their owners greater flexibility in the provision of essential care and nutritional and medical needs, despite their multiple time and attention demanding tasks and busy schedules. The prototype and subsequent evaluation, however, indicates that the research goal is feasible and achievable. Thus, current work extends previous efforts in the management of household pets.

9. Future scope :

According to Verified Market Research, Global Automatic and Smart Pet Feeder Market is growing at a faster pace with substantial growth rates over the last few years and is estimated that the market will grow significantly in the forecasted period i.e. 2019 to 2026.

10. Bibliography :

http://repository.psa.edu.my › ...PDF

SMART PET FEEDER NAMA NO. PENDAFTARAN MUHAMMAD ...

https://courses.engr.illinois.edu › ...PDF

Web results

Automatic Pet Feeder Project

https://digitalcommons.calpoly.edu › ...PDF

Design and Build of an Automated Animal ... - DigitalCommons@CalPoly

https://www.researchgate.net › 3141...

(PDF) Smart dog feeder design using wireless communication, MQTT and ...

https://www.researchgate.net › 3402...

(PDF) IOT based Pet Feeder - ResearchGate

11. APPENDIX:

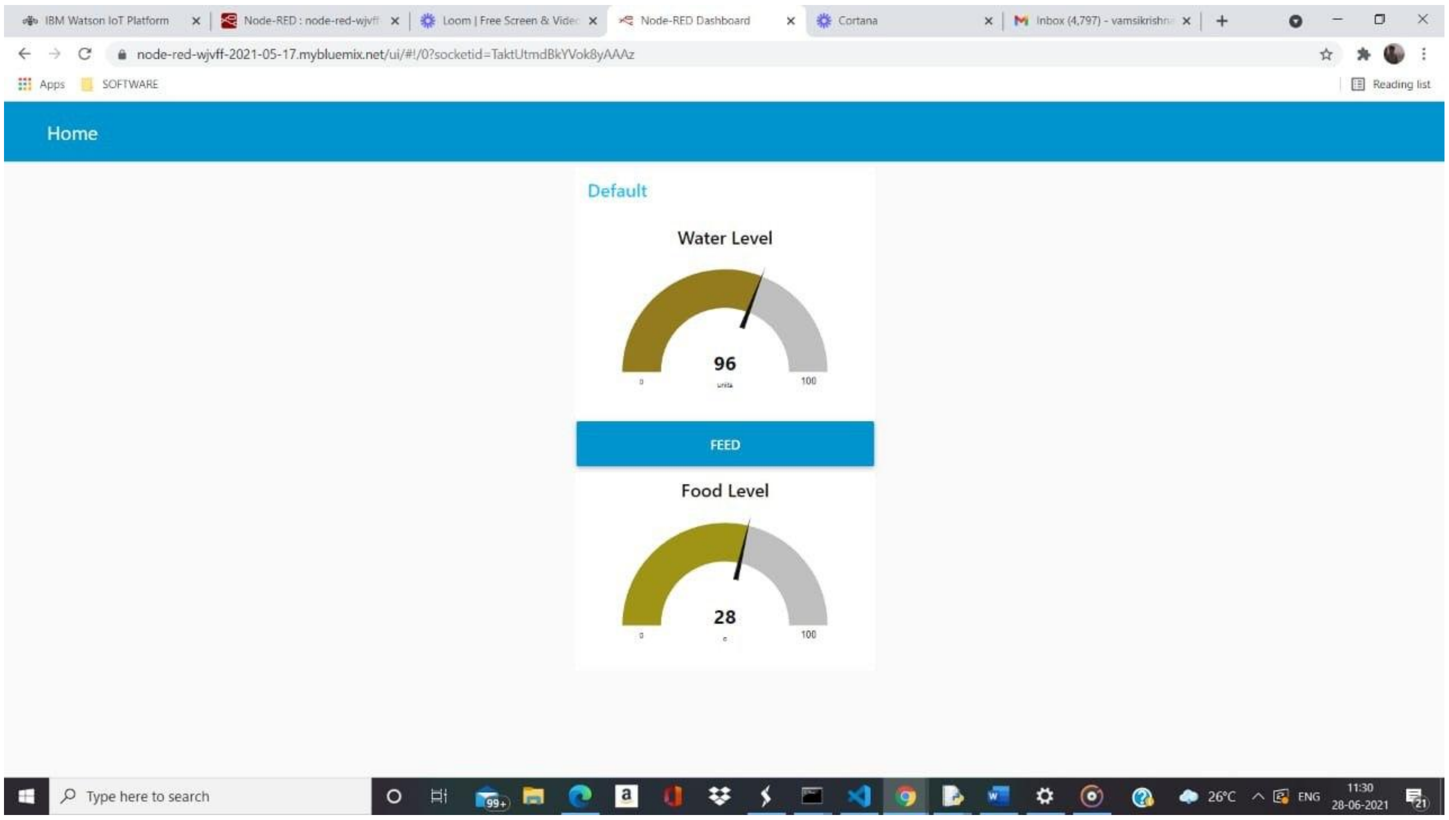
11.1 SOURCE CODE:

```
import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

import json

from gtts import gTTS

import os


#Provide your IBM Watson Device Credentials

organization = "1zqjlv"

deviceType = "iotdevice"

deviceId = "1001"

authMethod = "token"

authToken = "1234567890"


# Initialize the device client.
```

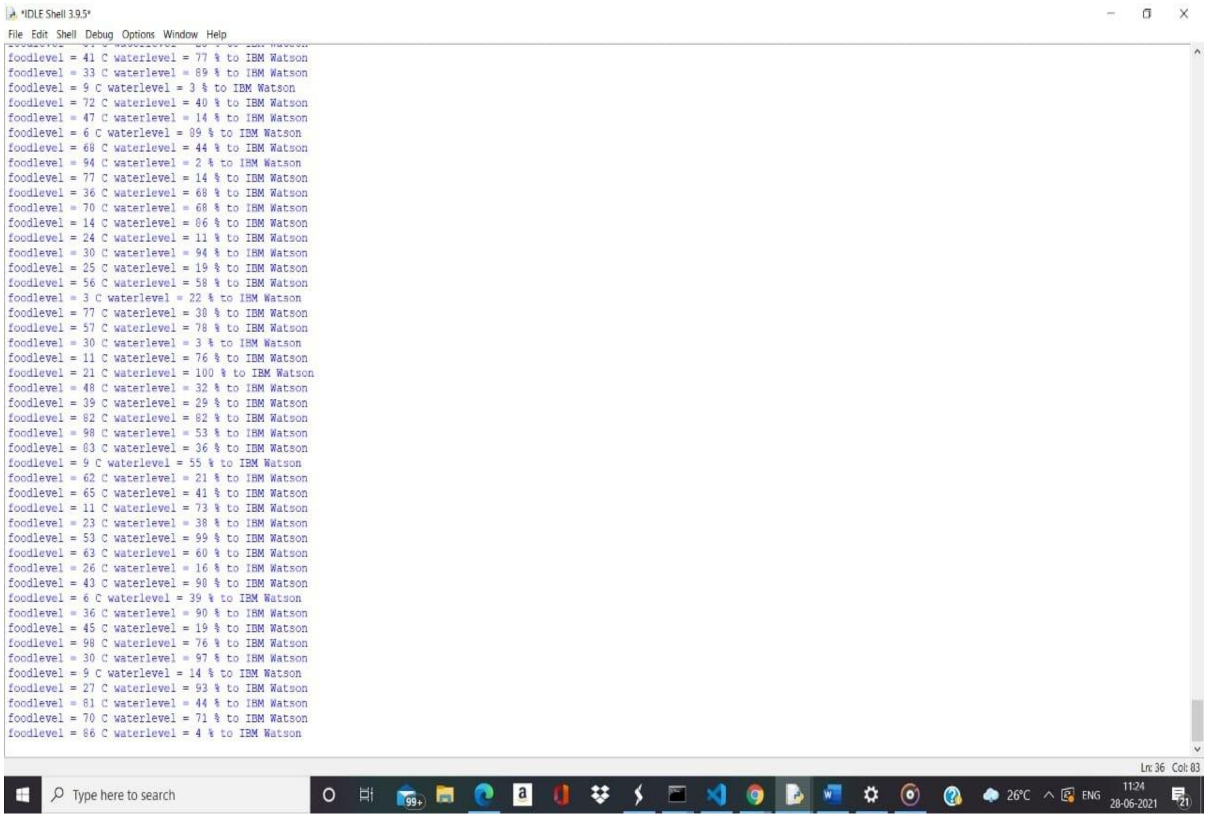
```
T=0

H=0

S=0


 def myCommandCallback(cmd):

      print("Command received: %s" % cmd.data['command'])

      if cmd.data['command']=='feed':

          print("FEED")

      if cmd.data['command']=='feedon':


           text="feeding device is activated"

           language='en'


          output=gTTS(text=text, lang=language,slow=False)

          output.save("feedon.mp3")


          os.system("start feedon.mp3")

       if cmd.data['command']=='feedoff':


            text="feeding device is diactivated"

            language='en'

            output=gTTS(text=text, lang=language,slow=False)

            output.save("feedoff.mp3")

            os.system("start feedoff.mp3")

       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:

                print(cmd.data['message'])



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:

    T=random.randint(0,100)

    H=random.randint(0,100)

    #Send Temperature & Humidity to IBM Watson

    data = {"d":{ 'foodlevel' : T, 'waterlevel': H, }}

    #print data

    def myOnPublishCallback():

        print ("foodlevel = %s C" % T, "waterlevel = %s %%" % H,"to IBM Watson")

        success = deviceCli.publishEvent("Data", "json", data, qos=0, on_publish=myOnPublishCallback)

        if not success:

            print("Not connected to IoTF")

       time.sleep(1)
```

```
        deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()
```

11.2 UI OUTPUT:



fig(1):node red ui



```
foodlevel = 41 C waterlevel = 77 % to IBM Watson
foodlevel = 33 C waterlevel = 89 % to IBM Watson
foodlevel = 9 C waterlevel = 3 % to IBM Watson
foodlevel = 72 C waterlevel = 40 % to IBM Watson
foodlevel = 47 C waterlevel = 14 % to IBM Watson
foodlevel = 6 C waterlevel = 89 % to IBM Watson
foodlevel = 68 C waterlevel = 44 % to IBM Watson
foodlevel = 94 C waterlevel = 2 % to IBM Watson
foodlevel = 77 C waterlevel = 14 % to IBM Watson
foodlevel = 36 C waterlevel = 68 % to IBM Watson
foodlevel = 70 C waterlevel = 68 % to IBM Watson
foodlevel = 14 C waterlevel = 86 % to IBM Watson
foodlevel = 24 C waterlevel = 11 % to IBM Watson
foodlevel = 30 C waterlevel = 94 % to IBM Watson
foodlevel = 25 C waterlevel = 19 % to IBM Watson
foodlevel = 56 C waterlevel = 58 % to IBM Watson
foodlevel = 3 C waterlevel = 22 % to IBM Watson
foodlevel = 77 C waterlevel = 38 % to IBM Watson
foodlevel = 57 C waterlevel = 78 % to IBM Watson
foodlevel = 30 C waterlevel = 3 % to IBM Watson
foodlevel = 11 C waterlevel = 76 % to IBM Watson
foodlevel = 21 C waterlevel = 100 % to IBM Watson
foodlevel = 48 C waterlevel = 32 % to IBM Watson
foodlevel = 39 C waterlevel = 29 % to IBM Watson
foodlevel = 82 C waterlevel = 82 % to IBM Watson
foodlevel = 98 C waterlevel = 53 % to IBM Watson
foodlevel = 83 C waterlevel = 36 % to IBM Watson
foodlevel = 9 C waterlevel = 55 % to IBM Watson
foodlevel = 62 C waterlevel = 21 % to IBM Watson
foodlevel = 65 C waterlevel = 41 % to IBM Watson
foodlevel = 11 C waterlevel = 73 % to IBM Watson
foodlevel = 23 C waterlevel = 38 % to IBM Watson
foodlevel = 53 C waterlevel = 99 % to IBM Watson
foodlevel = 63 C waterlevel = 60 % to IBM Watson
foodlevel = 26 C waterlevel = 16 % to IBM Watson
foodlevel = 43 C waterlevel = 98 % to IBM Watson
foodlevel = 6 C waterlevel = 39 % to IBM Watson
foodlevel = 36 C waterlevel = 90 % to IBM Watson
foodlevel = 45 C waterlevel = 19 % to IBM Watson
foodlevel = 98 C waterlevel = 76 % to IBM Watson
foodlevel = 30 C waterlevel = 97 % to IBM Watson
foodlevel = 9 C waterlevel = 14 % to IBM Watson
foodlevel = 27 C waterlevel = 93 % to IBM Watson
foodlevel = 81 C waterlevel = 44 % to IBM Watson
foodlevel = 70 C waterlevel = 71 % to IBM Watson
foodlevel = 86 C waterlevel = 4 % to IBM Watson
```

fig(2):python code result