A Project Report

on

Smart Kitchen Using IBM Cloud

By

B. Pravalika Reddy

Sathwika Pothkanuri

Andem Sai Prakash Reddy

as an intern at

smartinternz.com@rsip2020

on

Internet of Things

INTRODUCTION:-

Overview:-

The objective of this report is to propose IOT based Smart Kitchen using IBM Cloud.This application relies on sensing the weight of a kitchen storage container to track food consumption. This Web App and Mobile App can provide valuable insights around consumption patterns and help chefs predict and replenish their inventory just in time and also this project is about designing a GAS cylinder leakage monitoring system which is proposed for home safety. This Web App detects the leakage of the gas cylinder and alerts the consumer about the leak by SMS and as an emergency measure the system will turn off the power supply, while activating the alarm.

PURPOSE:-

The scenario of sensing the kitchen storage containers and jars to track the food consumption and a system which will sense if the gas in the cylinder is about to get over and to detect the leakage from the cylinders and to immediately switch ON the exhaust fans in the kitchen.To cope up with this a modern technique called Internet of Things is implemented, where the kitchen can be in a smart way.

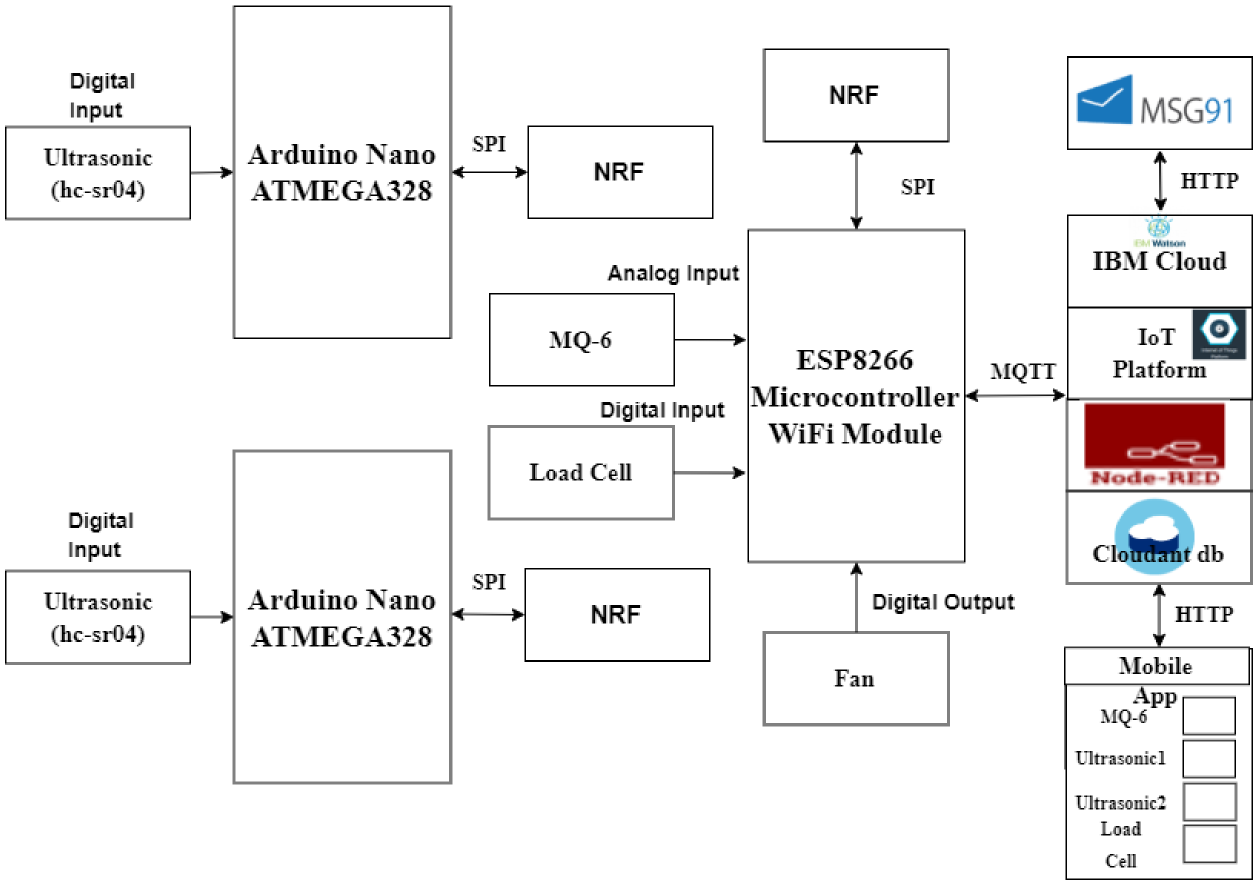
SCOPE:-

- When things like household appliances are connected to a network, they can work together in cooperation to provide the ideal service as a whole, not as a collection of independently working devices.
- We can replace all the regular storage jars with the smart jars, which sends an alert when the jar gets empty or the measured sensor value is below the threshold.
- These jars communicate with the controller through Nrf communication.
- The cylinder is attached with a leakage sensor that detects the leakage from the cylinder and sends a notification if any leakage is detected.
- If any leakage is detected the exhaust fans are automatically switched ON.
- Cylinder weight is also measured and sends an alert when it is empty, based on the empty cylinder weight

LITERATURE SURVEY:-

In this project, the technological development provides and increase human beings safety and comfort directly and indirectly. For this purpose developing technologies directly affects the life standards using smart home systems design.
The embedded sensor measures the weight or the level of the items which in updated to the database whenever grocery items are placed or taken out for cooking. When the items reach the predefined threshold level, the system generates the automated shopping list.
The system detects kitchen parameters each as room temperature, fire detection, motion detection has been developed. The system can detect the status of kitchen and send alert messages via network automatically. If the conditions get abnormal, the concerned authority can control the system though this mobile phone by sending proper decision.

THEORITICAL ANALYSIS:-
The Block Diagram for the IoT based Smart Kitchen can be seen below:



Project deliverables:

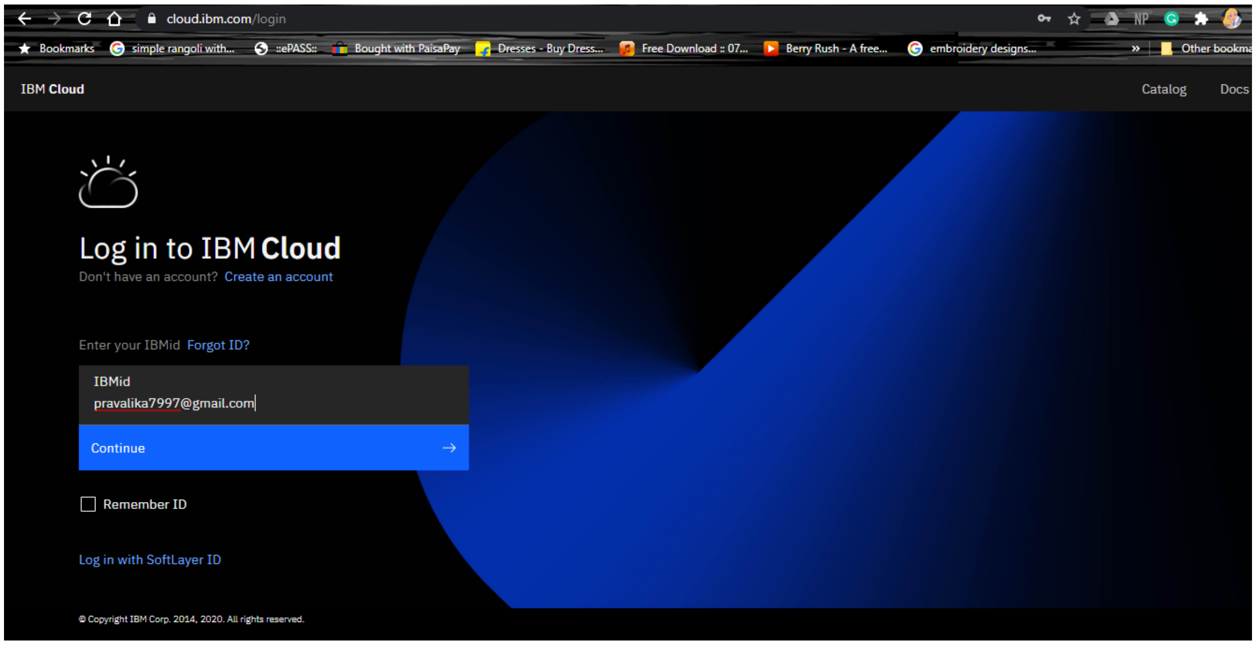
1.Create an IBM account, create node red application, create a IBM Watson Iot platform.
2.Create a fast2sms account (for sending alert messages).
3.Code snippet for sending sensor data to the Watson platform and for sending alert messages to the user.
4.Create the node red flow to get data from the device and request to communicate with the mobile app.
5.Create a mobile app using MIT APP INVENTOR and configure it to get data from the cloud.

Designing Procedure:-

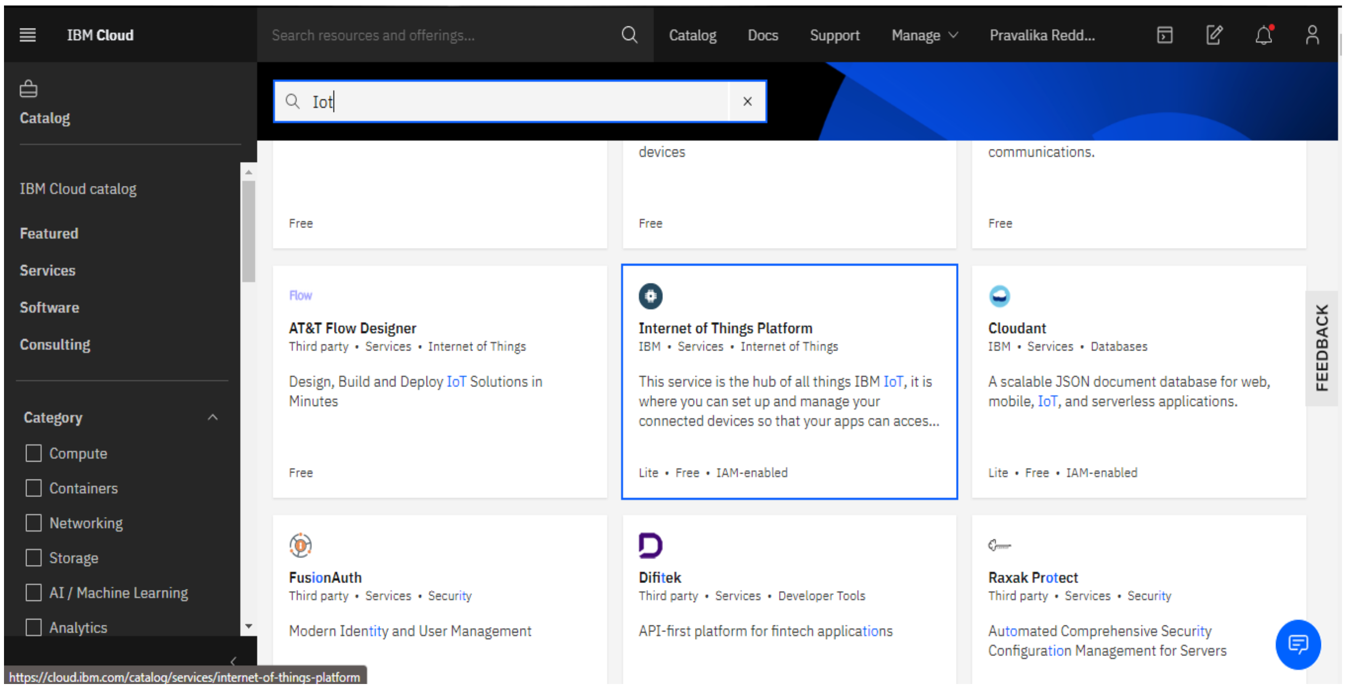
- Sign-up for IBM Academic Initiative Account through the link

https://my15.digitalexperience.ibm.com/b73a5759-c6a6-4033-ab6b-d9d4f9a6d65b/dxsites/151914d1-03d2-48fe-97d9-d21166848e65/academic/home

1) Sign-in to your IBM cloud account from the link https://cloud.ibm.com/login.There, go to Catalog and search for IoT in the search bar. Then select Internet of Things platform and subscribe for the desired plan and click create. Now, in the menu, go to Resource List → Services →Internet of Things Platform and then click Launch, as shown below:



Log in to IBM Cloud
Don't have an account? Create an account
Enter your IBMid Forgot ID?
IBMid
pravalika7997@gmail.com
Continue
Remember ID
Log in with SoftLayer ID
© Copyright IBM Corp. 2014, 2020. All rights reserved.



devices

communications.

Free

Free

Free

AT&T Flow Designer
Third party • Services • Internet of Things
Design, Build and Deploy IoT Solutions in Minutes
Free

Internet of Things Platform
IBM • Services • Internet of Things
This service is the hub of all things IBM IoT, it is where you can set up and manage your connected devices so that your apps can acces...
Lite • Free • IAM-enabled

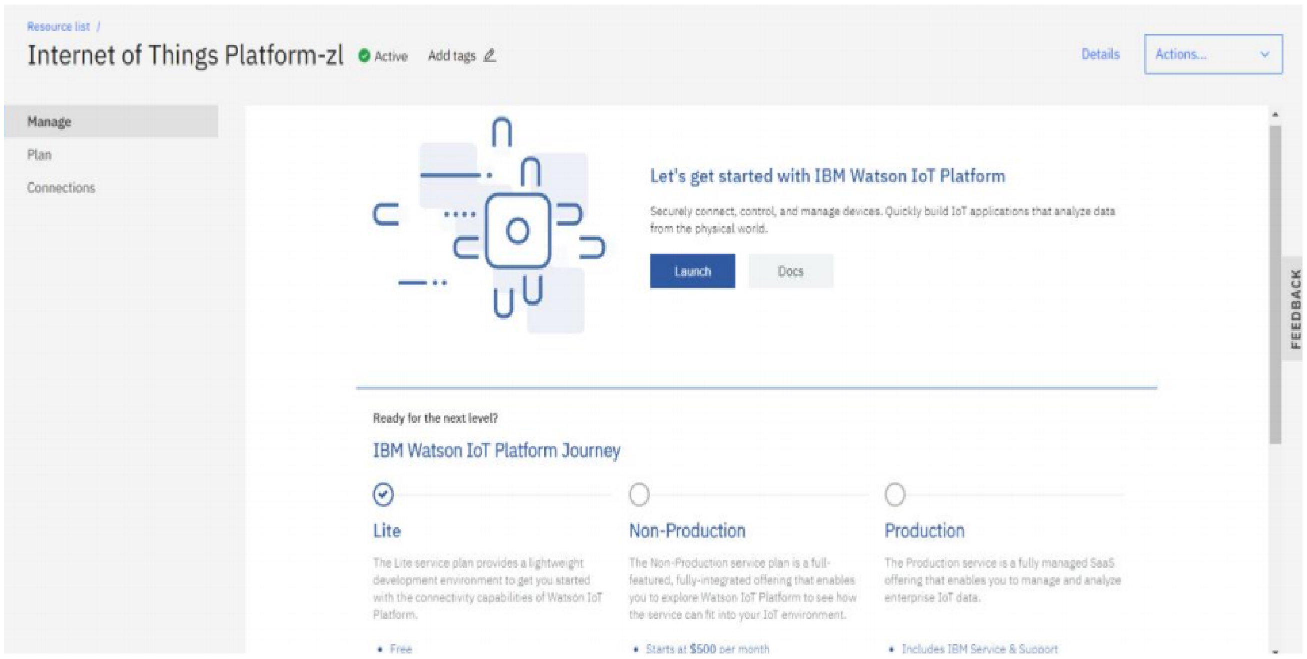
Cloudant
IBM • Services • Databases
A scalable JSON document database for web, mobile, IoT, and serverless applications.
Lite • Free • IAM-enabled

FusionAuth
Third party • Services • Security
Modern Identity and User Management

Difitek
Third party • Services • Developer Tools
API-first platform for fintech applications

Raxak Protect
Third party • Services • Security
Automated Comprehensive Security Configuration Management for Servers

2) In the next step, create the service and launch the service on the dashboard.After you launch the watson Iot platform,Add a device to it.Here shows the launch below:



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.

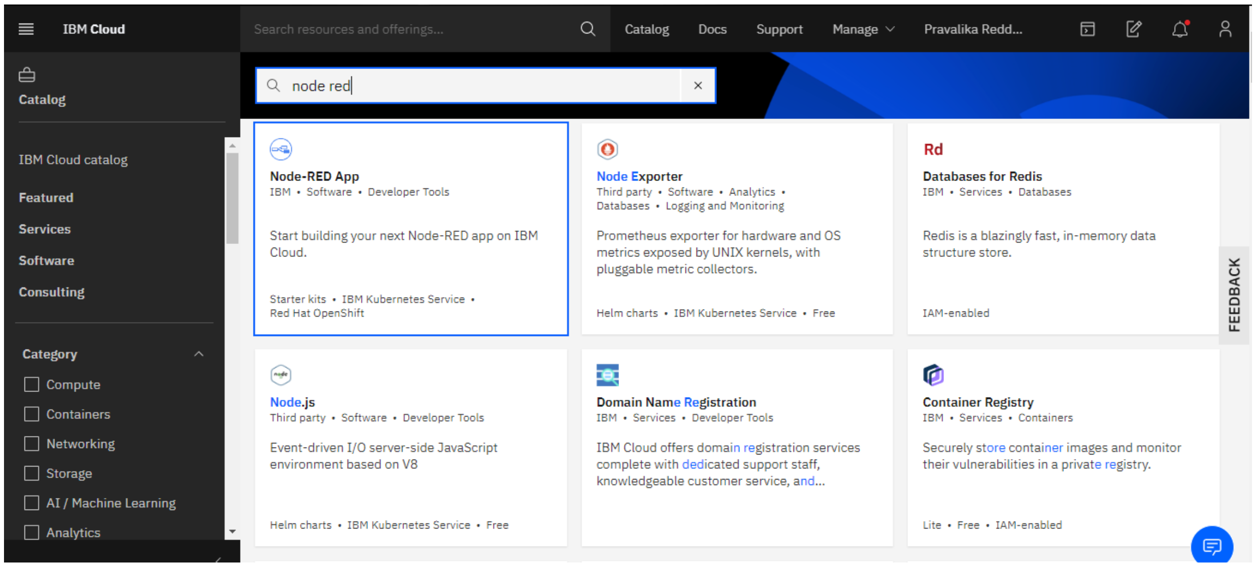
Ready for the next level?
IBM Watson IoT Platform Journey

Lite
The Lite service plan provides a lightweight development environment to get you started with the connectivity capabilities of Watson IoT Platform.
• Free

Non-Production
The Non-Production service plan is a full-featured, fully-integrated offering that enables you to explore Watson IoT Platform to see how the service can fit into your IoT environment.
• Starts at $500 per month

Production
The Production service is a fully managed SaaS offering that enables you to manage and analyze enterprise IoT data.
• Includes IBM Service & Support

3) Return back to the main page and search node red in the catalog, then select the node red app and create a node red application.



node red

Node-RED App
IBM • Software • Developer Tools
Start building your next Node-RED app on IBM Cloud.
Starter kits • IBM Kubernetes Service • Red Hat OpenShift

Node Exporter
Third party • Software • Analytics • Databases • Logging and Monitoring
Prometheus exporter for hardware and OS metrics exposed by UNIX kernels, with pluggable metric collectors.
Helm charts • IBM Kubernetes Service • Free

Databases for Redis
IBM • Services • Databases
Redis is a blazingly fast, in-memory data structure store.
IAM-enabled

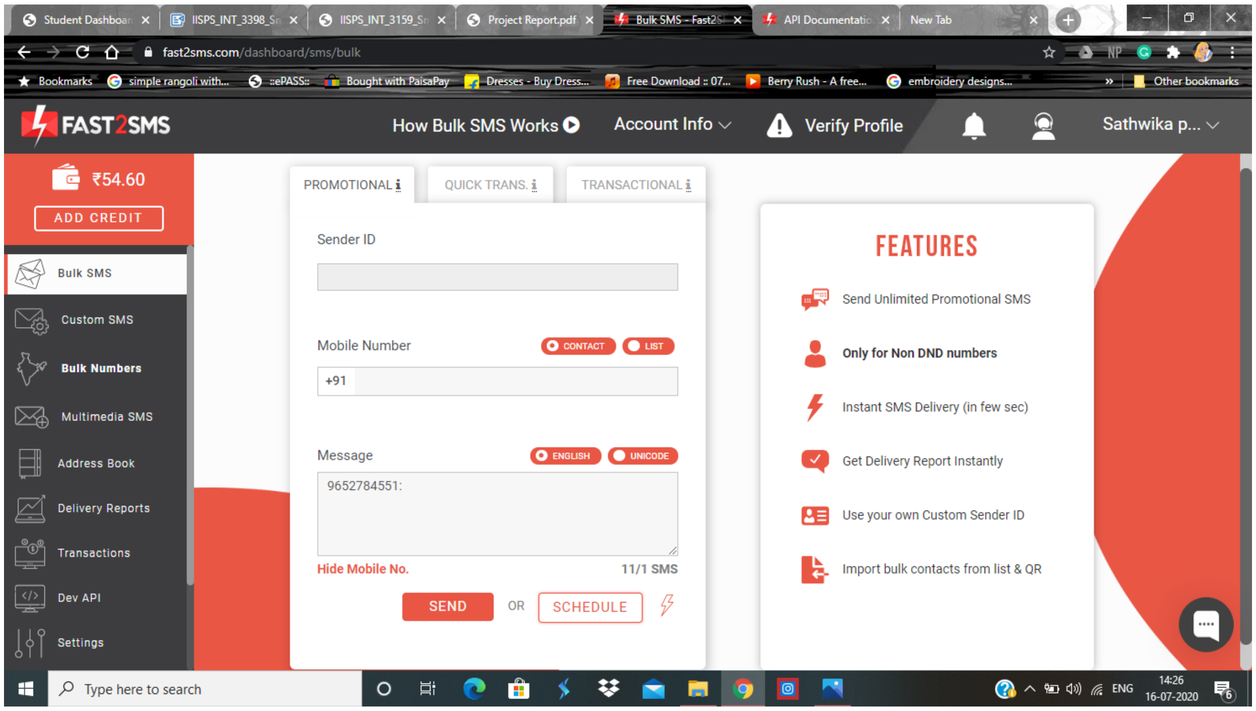
Node.js
Third party • Software • Developer Tools
Event-driven I/O server-side JavaScript environment based on V8
Helm charts • IBM Kubernetes Service • Free

Domain Name Registration
IBM • Services • Developer Tools
IBM Cloud offers domain registration services complete with dedicated support staff, knowledgeable customer service, and...

Container Registry
IBM • Services • Containers
Securely store container images and monitor their vulnerabilities in a private registry.
Lite • Free • IAM-enabled

FAST2SMS ACCOUNT FOR SENDING ALERT MESSAGES

1) Now create fast2sms account to send alert messages to the user.
Create an account in the fast2sms app by logging in to that website.



Sender ID

Mobile Number
+91

Message
9652784551:

11/1 SMS

FEATURES
Send Unlimited Promotional SMS
Only for Non DND numbers
Instant SMS Delivery (in few sec)
Get Delivery Report Instantly
Use your own Custom Sender ID
Import bulk contacts from list & QR

2) Code snippet for sending sensor data to the Watson Iot platform and for sending alert messages to the user.

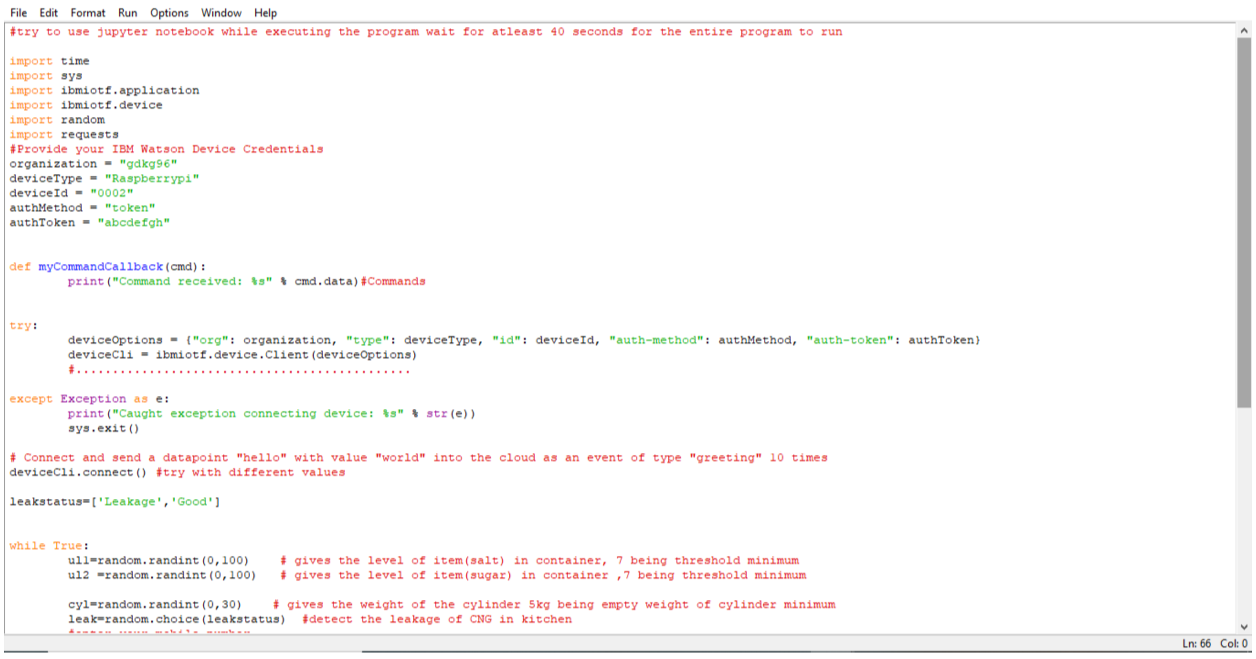
Note:- we won't have any sensors to send the data to the cloud so we send sensor data with python code.

The following is the code used for this task

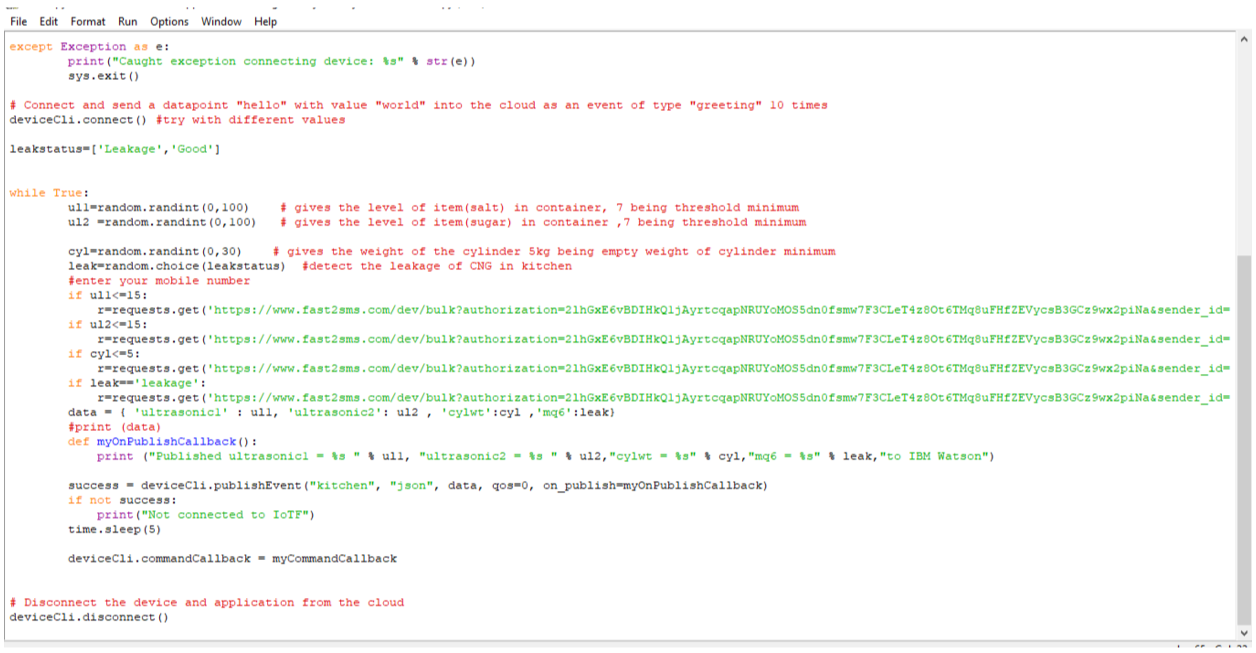
In the below code enter the credentials of the device that you created in the Watson Iot platform

PYTHON CODE

NOTE:- DONT PLACE MESSAGE CODE INSIDE THE LOOP IF YOU DO THAT THE FLOW OF MESSAGES WONT STOP

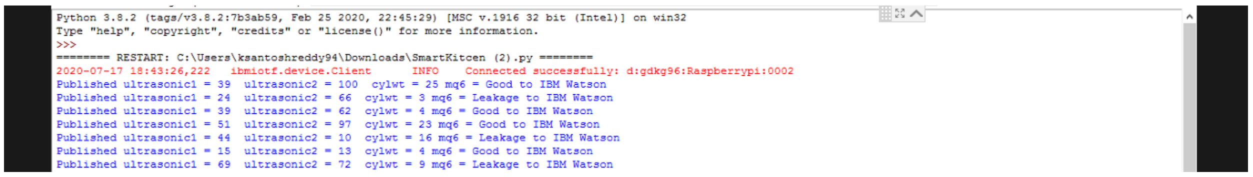


```python
#try to use jupyter notebook while executing the program wait for atleast 40 seconds for the entire program to run

import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
import requests
#Provide your IBM Watson Device Credentials
organization = "gdkg96"
deviceType = "Raspberrypi"
deviceId = "0002"
authMethod = "token"
authToken = "abcdefgh"


def myCommandCallback(cmd):
        print("Command received: %s" % cmd.data)#Commands


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() #try with different values

leakstatus=['Leakage','Good']


while True:
        ul1=random.randint(0,100)    # gives the level of item(salt) in container, 7 being threshold minimum
        ul2 =random.randint(0,100)   # gives the level of item(sugar) in container ,7 being threshold minimum

        cyl=random.randint(0,30)    # gives the weight of the cylinder 5kg being empty weight of cylinder minimum
        leak=random.choice(leakstatus)  #detect the leakage of CNG in kitchen
```



```
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() #try with different values

leakstatus=['Leakage','Good']


while True:
        ul1=random.randint(0,100)    # gives the level of item(salt) in container, 7 being threshold minimum
        ul2 =random.randint(0,100)   # gives the level of item(sugar) in container ,7 being threshold minimum

        cyl=random.randint(0,30)    # gives the weight of the cylinder 5kg being empty weight of cylinder minimum
        leak=random.choice(leakstatus)  #detect the leakage of CNG in kitchen
        #enter your mobile number
        if ul1<=15:
            r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=21hGxE6vBDIHkQ1jAyrtcqapNRUYoMOS5dn0fsmw7F3CLeT4z8Ot6TMq8uFHfZEVycsB3GCz9wx2piNa&sender_id=
        if ul2<=15:
            r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=21hGxE6vBDIHkQ1jAyrtcqapNRUYoMOS5dn0fsmw7F3CLeT4z8Ot6TMq8uFHfZEVycsB3GCz9wx2piNa&sender_id=
        if cyl<=5:
            r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=21hGxE6vBDIHkQ1jAyrtcqapNRUYoMOS5dn0fsmw7F3CLeT4z8Ot6TMq8uFHfZEVycsB3GCz9wx2piNa&sender_id=
        if leak=='leakage':
            r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=21hGxE6vBDIHkQ1jAyrtcqapNRUYoMOS5dn0fsmw7F3CLeT4z8Ot6TMq8uFHfZEVycsB3GCz9wx2piNa&sender_id=
        data = { 'ultrasonic1' : ul1, 'ultrasonic2': ul2 , 'cylwt':cyl ,'mq6':leak}
        #print (data)
        def myOnPublishCallback():
            print ("Published ultrasonic1 = %s " % ul1, "ultrasonic2 = %s " % ul2,"cylwt = %s" % cyl,"mq6 = %s" % leak,"to IBM Watson")

        success = deviceCli.publishEvent("kitchen", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(5)

        deviceCli.commandCallback = myCommandCallback


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

So the output for this code is shown below:



```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======== RESTART: C:\Users\ksantoshreddy94\Downloads\SmartKitcen (2).py ========
2020-07-17 18:43:26,222   ibmiotf.device.Client      INFO    Connected successfully: d:gdkg96:Raspberrypi:0002
Published ultrasonic1 = 39  ultrasonic2 = 100  cylwt = 25 mq6 = Good to IBM Watson
Published ultrasonic1 = 24  ultrasonic2 = 66  cylwt = 3 mq6 = Leakage to IBM Watson
Published ultrasonic1 = 39  ultrasonic2 = 62  cylwt = 4 mq6 = Good to IBM Watson
Published ultrasonic1 = 51  ultrasonic2 = 97  cylwt = 23 mq6 = Good to IBM Watson
Published ultrasonic1 = 44  ultrasonic2 = 10  cylwt = 16 mq6 = Leakage to IBM Watson
Published ultrasonic1 = 15  ultrasonic2 = 13  cylwt = 4 mq6 = Good to IBM Watson
Published ultrasonic1 = 69  ultrasonic2 = 72  cylwt = 9 mq6 = Leakage to IBM Watson
```

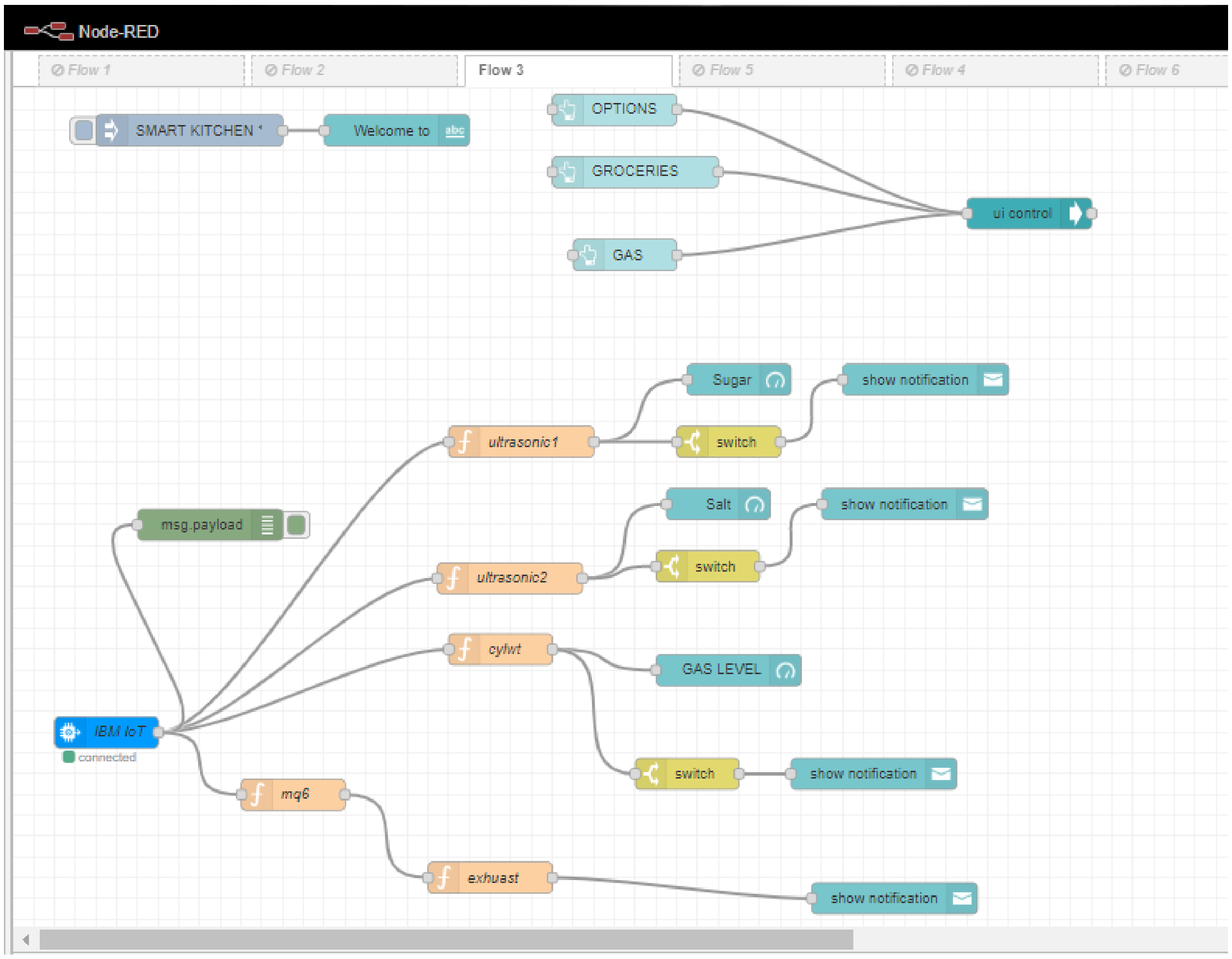
3)Now, once the sensor data is received by the cloud, we use a special tool called NodeRed, a low-code programming tool for event-driven applications, to build a Web-App.

To install Node-Red on windows, go to [https://nodered.org/docs/gettingstarted/windows.](https://nodered.org/docs/gettingstarted/windows)
(For further details on how to use Node-Red, visit https://nodered.org/docs/userguide/)

Now, to **Create** the node red flow to get data from the device and http request to communicate with the mobile app.

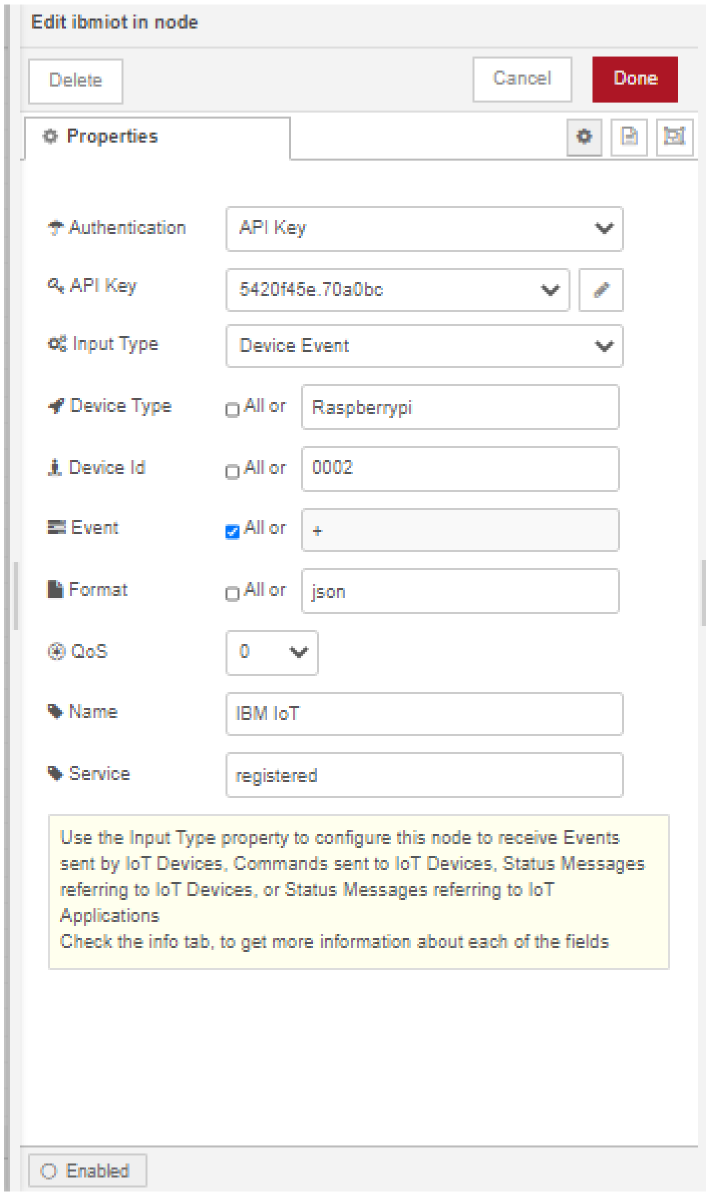
Now, to build the web app for the Smart kitchen using Node-Red, a minimum of two flows are required:

FLOW 1:
To get data from the device



To connect **ibmiot** device node ,double click on the node and enter the device credentials of the device that you have in your Watson Iot platform.

In the above flow, the node properties of 'http request' and 'function' nodes are as follows:



Authentication: API Key

API Key: 5420f45e.70a0bc

Input Type: Device Event

Device Type: ☐ All or Raspberrypi

Device Id: ☐ All or 0002

Event: ✓ All or +

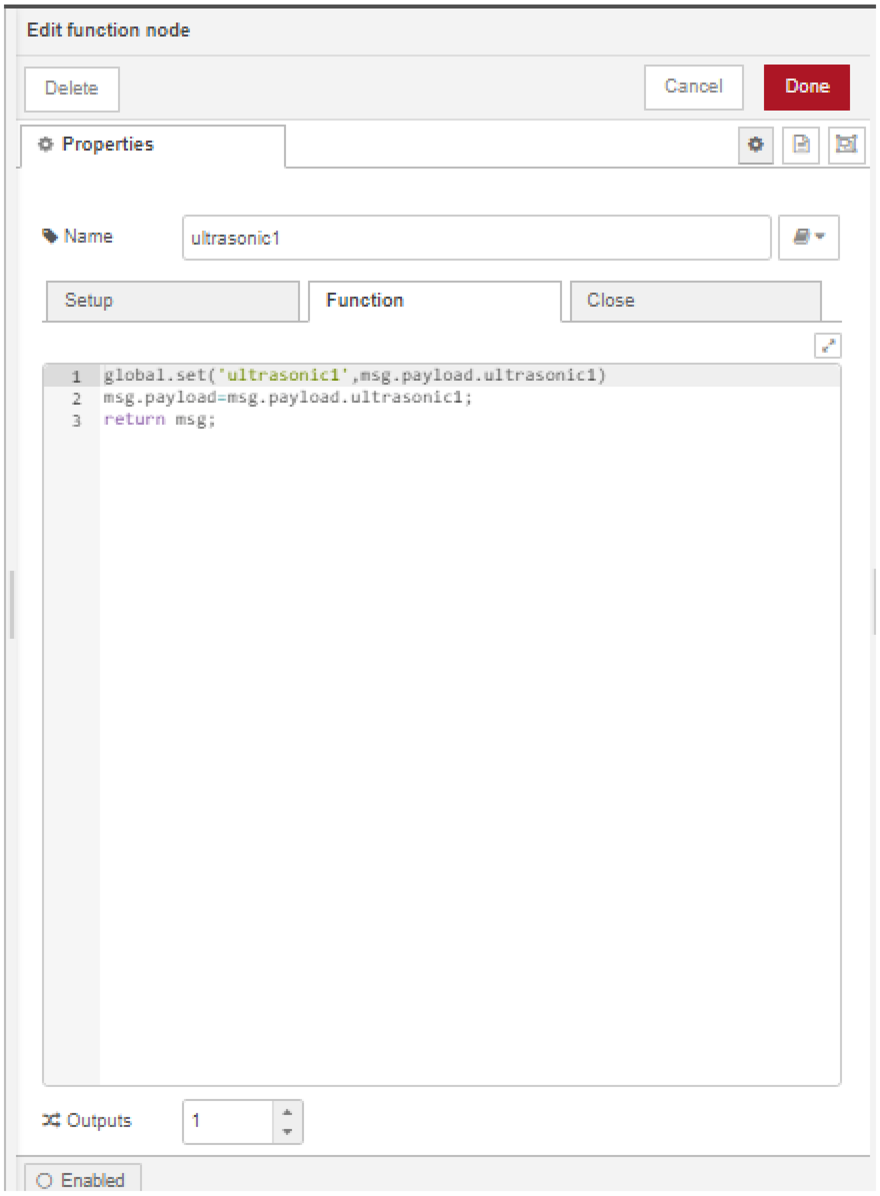
Format: ☐ All or json

QoS: 0

Name: IBM IoT

Service: registered

Use the Input Type property to configure this node to receive Events sent by IoT Devices, Commands sent to IoT Devices, Status Messages referring to IoT Devices, or Status Messages referring to IoT Applications
Check the info tab, to get more information about each of the fields



Name: ultrasonic1

```
global.set('ultrasonic1',msg.payload.ultrasonic1)
msg.payload=msg.payload.ultrasonic1;
return msg;
```

Outputs: 1

code for all the 4 function nodes:

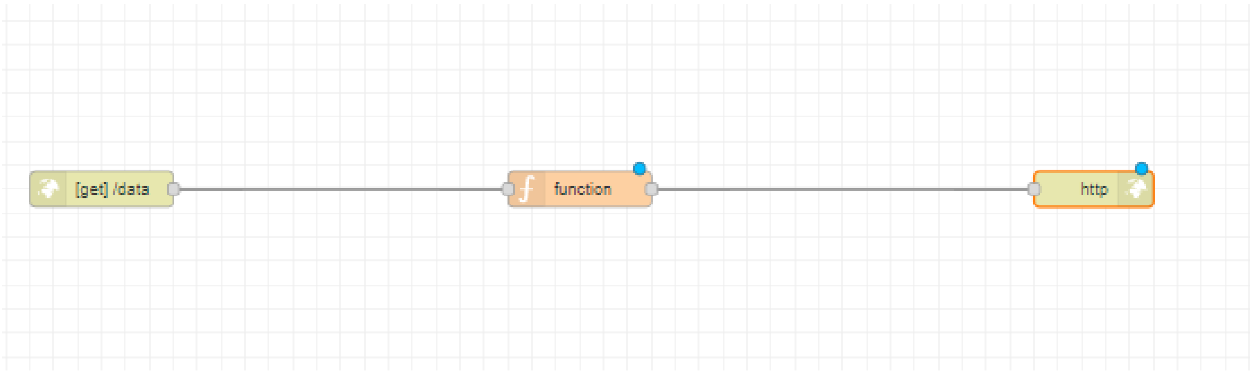
1) **global.set('ultrasonic1',msg.payload.ultrasonic1)**
msg.payload=msg.payload.ultrasonic1;
return msg;

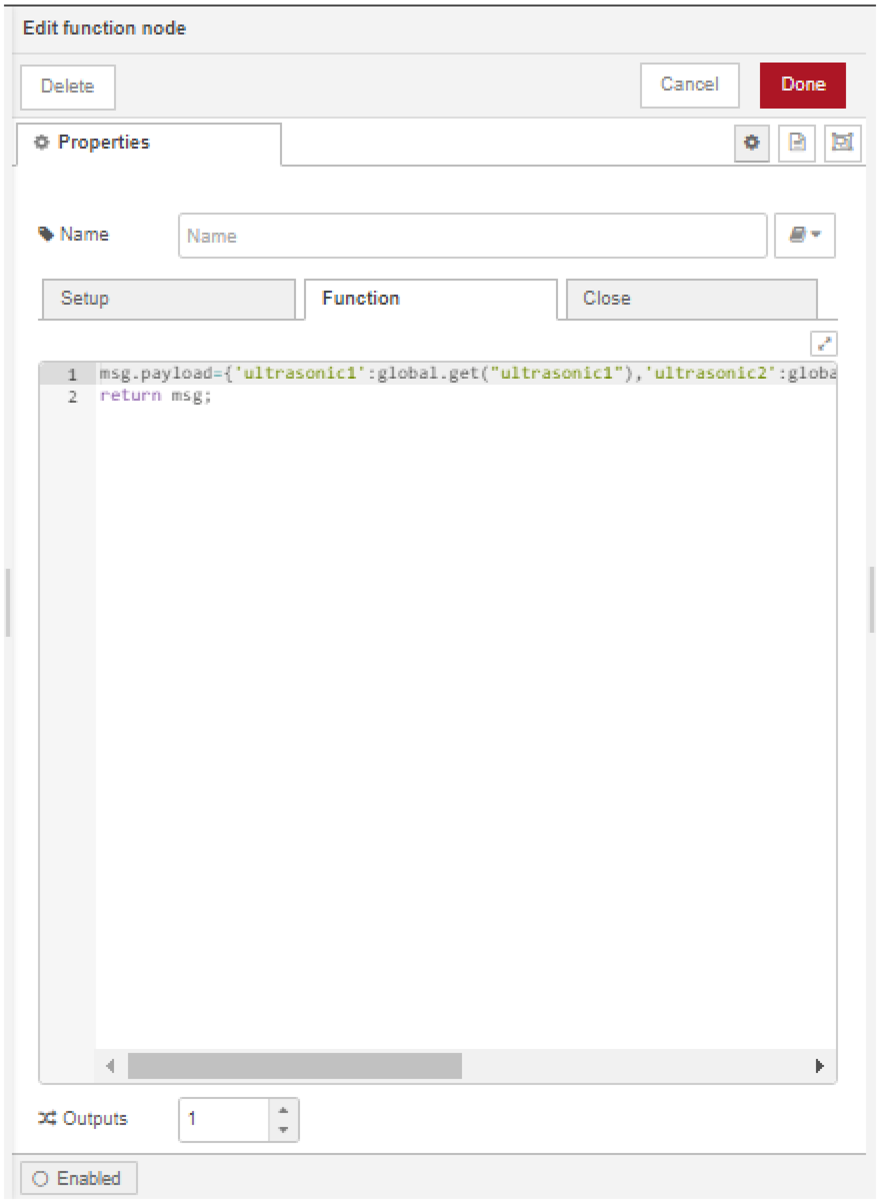
2) **global.set('ultrasonic2',msg.payload.ultrasonic2)**
msg.payload=msg.payload.ultrasonic2;
return msg;

```
3)  global.set('cylwt',msg.payload.cylwt)
    msg.payload=msg.payload.cylwt;
    return msg;

4)  global.set('mq6',msg.payload.mq6)
    msg.payload=msg.payload.mq6;
    return msg;
```

Flow2:-

To create http request to communicate with mobile app





```
msg.payload={'ultrasonic1':global.get("ultrasonic1"),'ultrasonic2':globa
return msg;
```

Name: Name

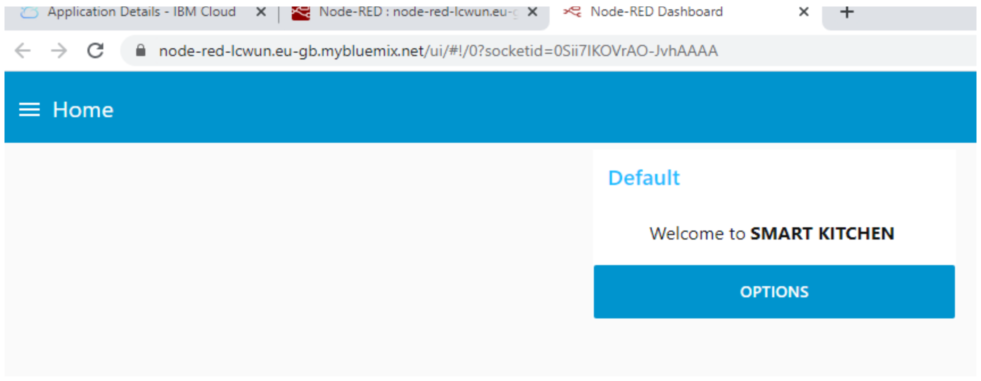
Outputs: 1

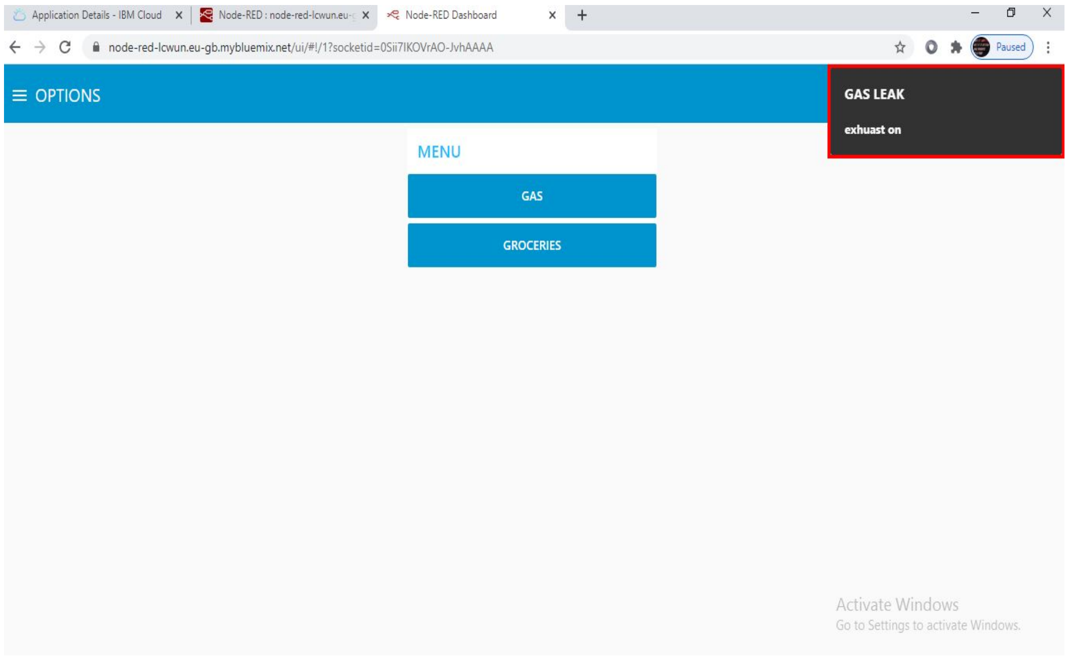
code for the function node:

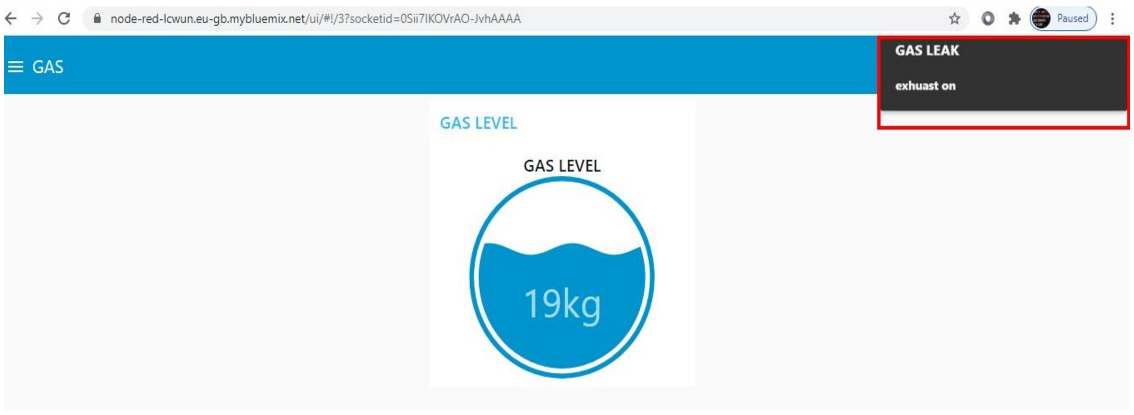
1)
sg.payload={'ultrasonic1':global.get("ultrasonic1"),'ultrasonic2':global.get("ultrasonic2"),'cylwt':global.get("cylwt"),'mq6':global.get("mq6")}
return msg;

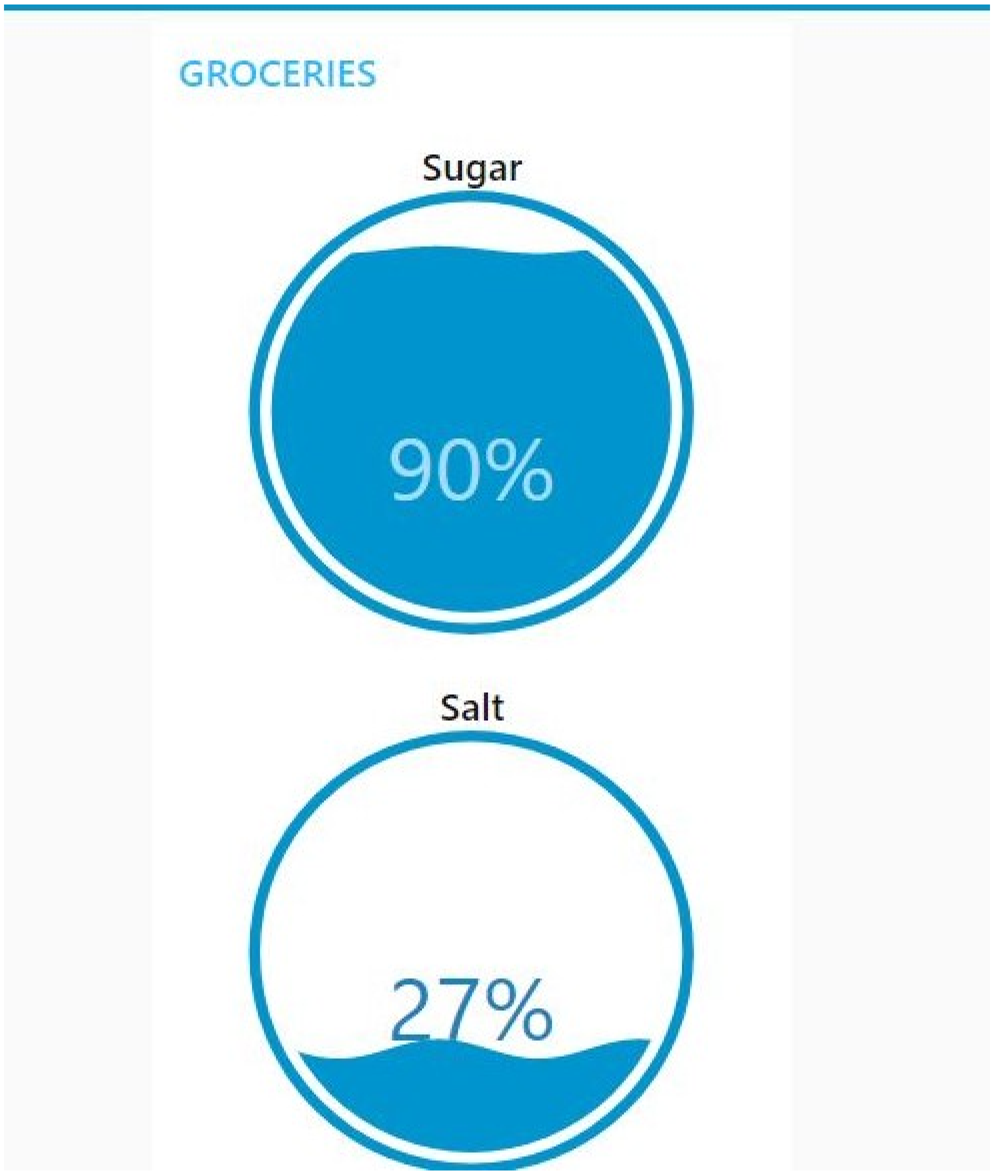
The data has been sent to the server

RESULT:-
Following the above designing procedure results in a Web Application in a smart way.
The Web Application generated by the above designing procedure is as follows:


Application Details - IBM Cloud
Node-RED : node-red-lcwun.eu-
Node-RED Dashboard
node-red-lcwun.eu-gb.mybluemix.net/ui/#!/0?socketid=0Sii7IKOVrAO-JvhAAAA
Home
Default
Welcome to SMART KITCHEN
OPTIONS

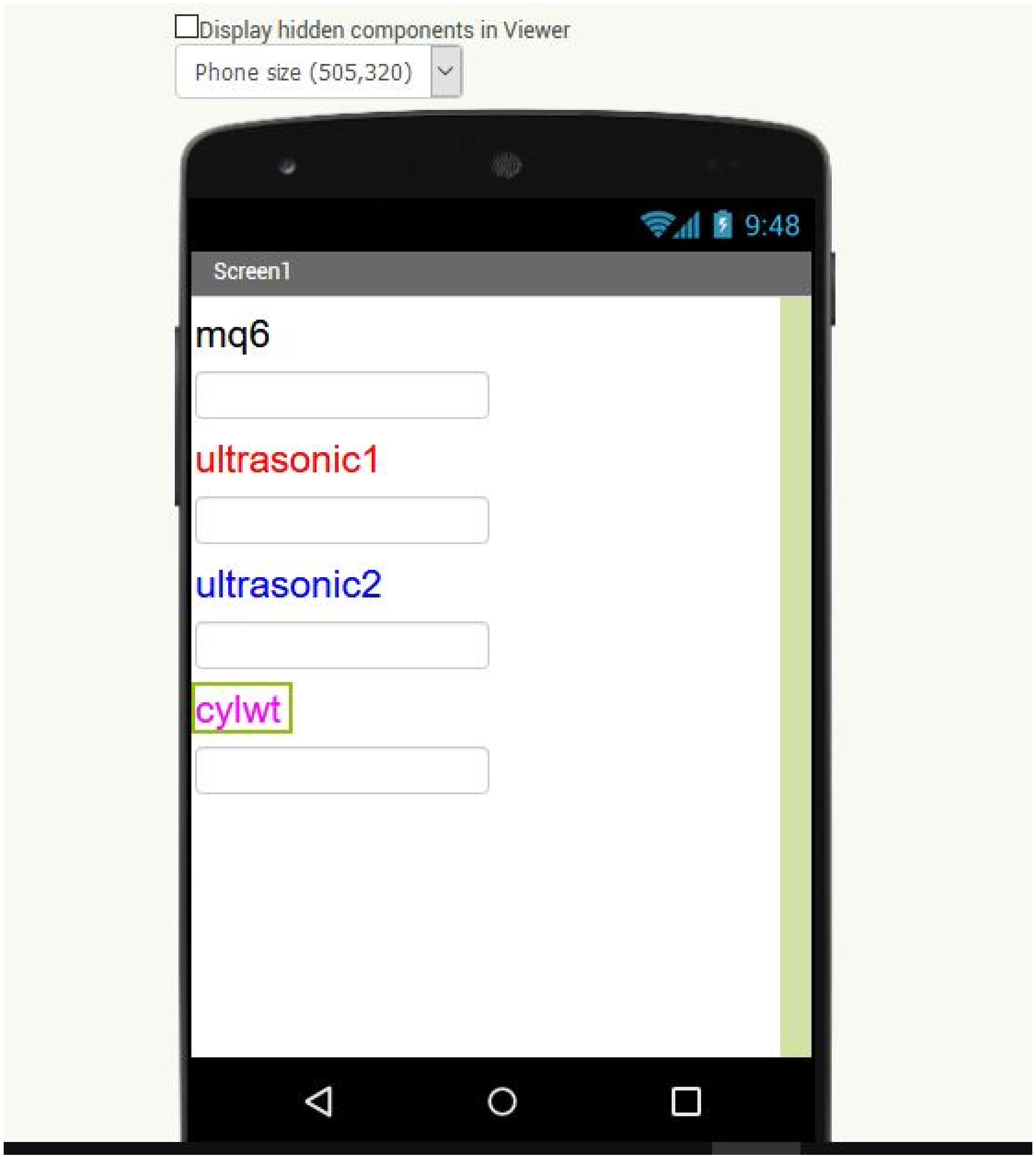

Application Details - IBM Cloud
Node-RED : node-red-lcwun.eu-
Node-RED Dashboard
node-red-lcwun.eu-gb.mybluemix.net/ui/#!/1?socketid=0Sii7IKOVrAO-JvhAAAA
Paused
OPTIONS
GAS LEAK
exhuast on
MENU
GAS
GROCERIES
Activate Windows
Go to Settings to activate Windows.


node-red-lcwun.eu-gb.mybluemix.net/ui/#!/3?socketid=0Sii7IKOVrAO-JvhAAAA
Paused
GAS
GAS LEAK
exhuast on
GAS LEVEL
GAS LEVEL
19kg

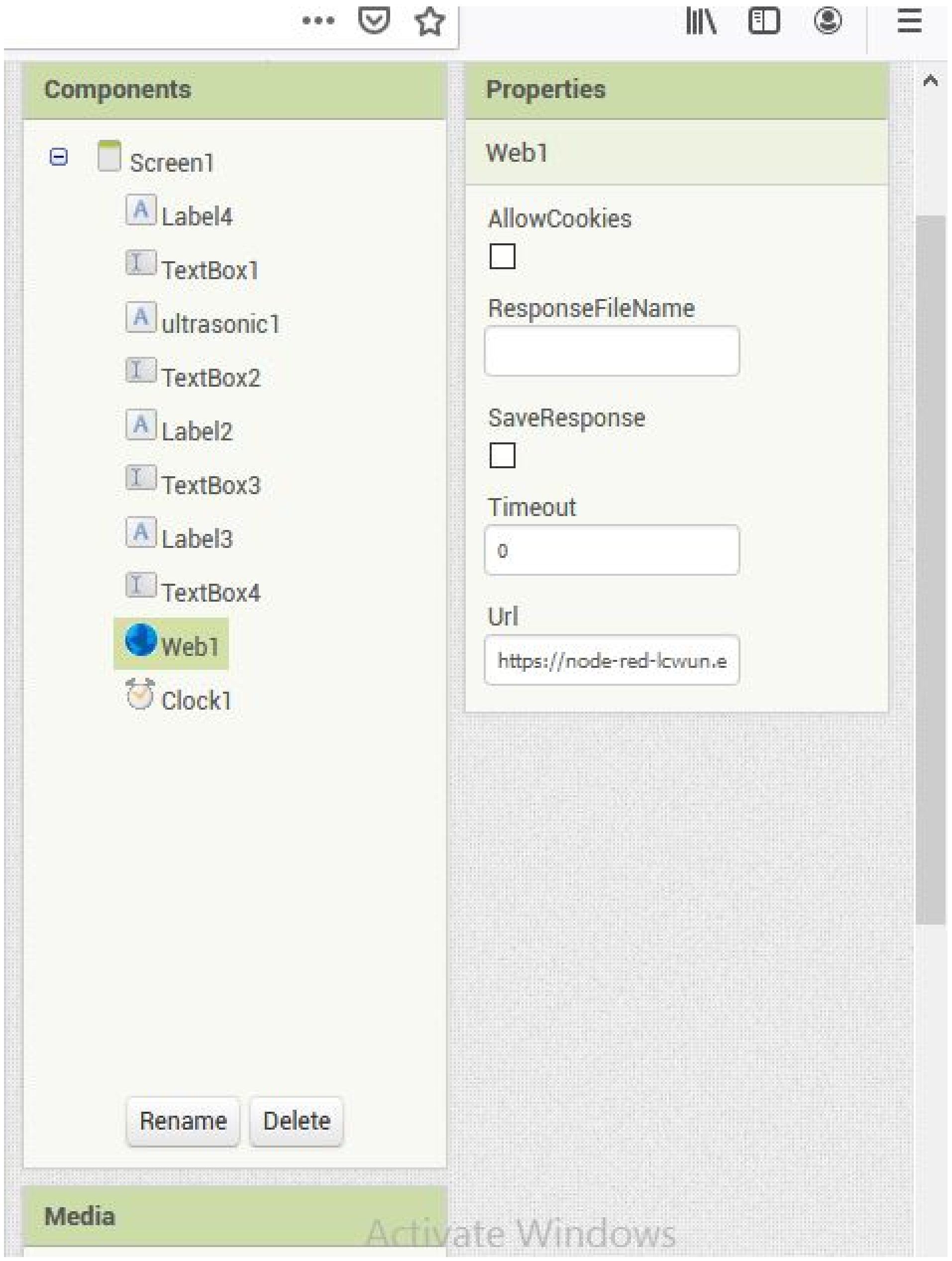


note:- cylwt is the weight of cylinder and ultrasonic1 gives the quantity of sugar and ultrasonic2 gives salt quantity kept in jars.

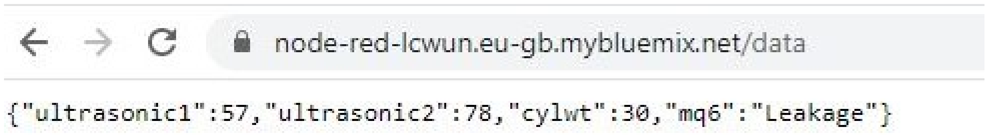
4) Create a mobile app using MIT APP INVENTOR and configure it to get data from the cloud.Now open your browser and search 'mit app inventor' and open the website.
Click on 'create apps' on the dashboard by logging to your google account.Give a name to your project leaving no space in between.Configure the ui of your app and it should have 4 labels and their respective block.



Now drop the web from connectivity on to the board and enter the url in the web

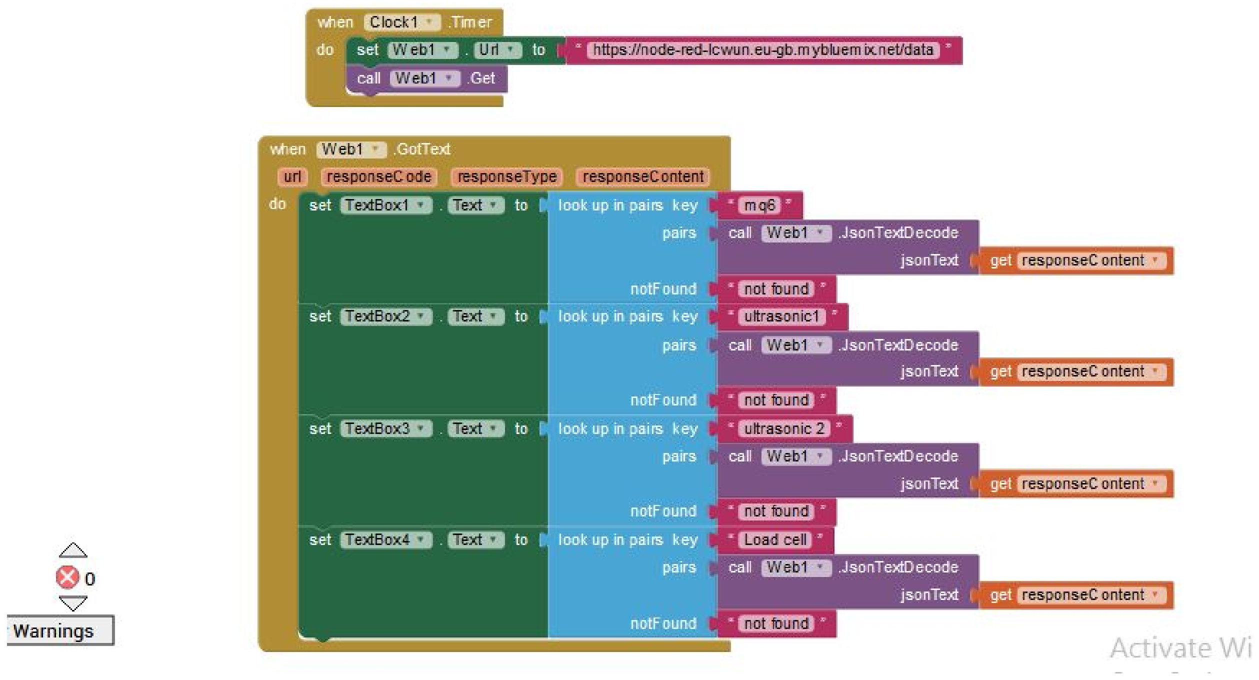


Note:- Enter the url so that the app receives data from the url that you entered in the web from IBM devices.



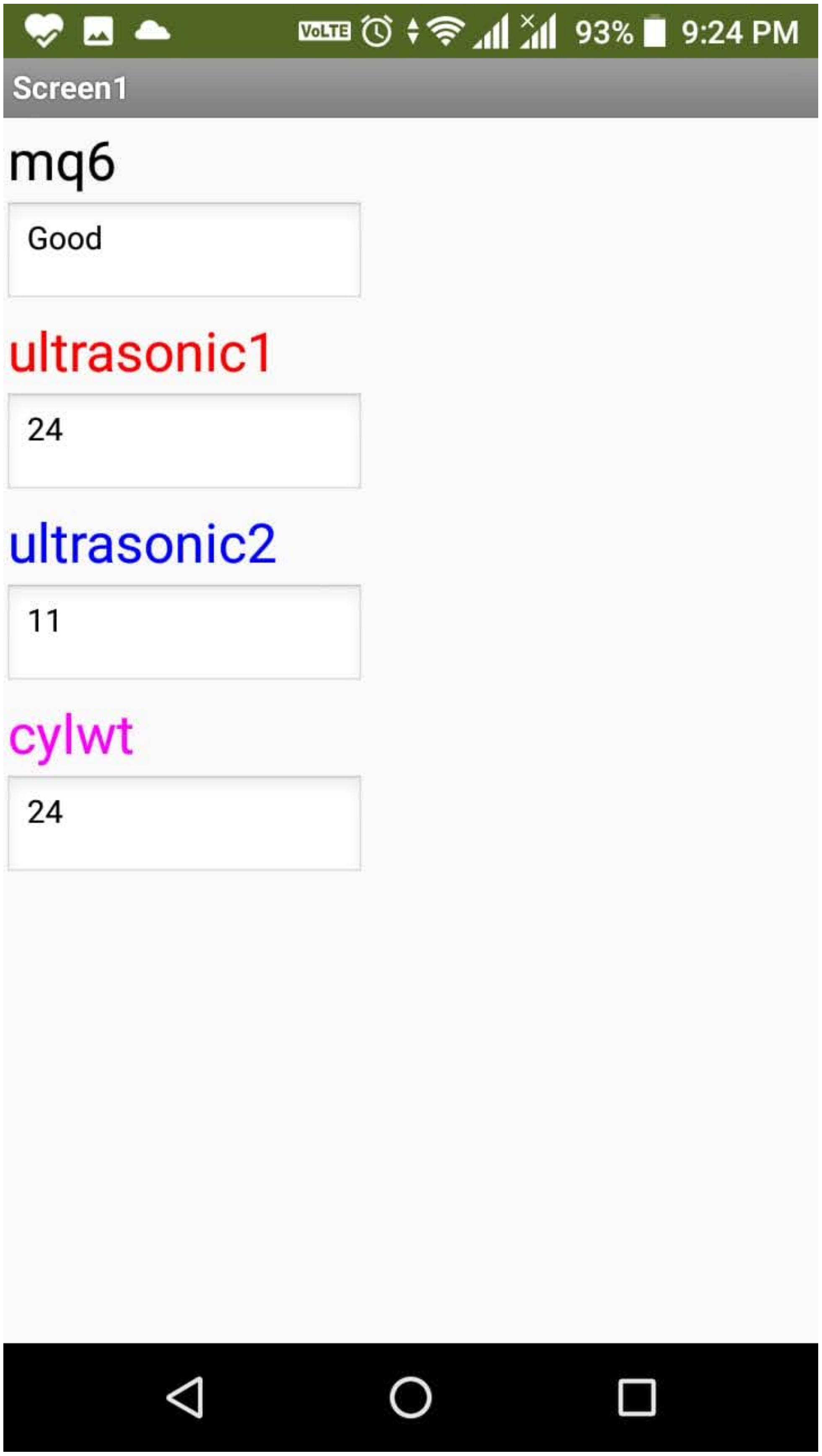
```
{"ultrasonic1":57,"ultrasonic2":78,"cylwt":30,"mq6":"Leakage"}
```

Click on the blocks on top right corner of the screen and start arranging the blocks to create backend of the app. Set the blocks in this manner for the text boxes.



```
when Clock1 .Timer
do  set Web1 . Url to " https://node-red-lcwun.eu-gb.mybluemix.net/data "
    call Web1 .Get

when Web1 .GotText
 url  responseCode  responseType  responseContent
do  set TextBox1 . Text to  look up in pairs  key " mq6 "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox2 . Text to  look up in pairs  key " ultrasonic1 "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox3 . Text to  look up in pairs  key " ultrasonic 2 "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox4 . Text to  look up in pairs  key " Load cell "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
```

These blocks are to decode data that is in the form of json and display them in there respective text boxes. Click on build option on the top of dashboard and download the apk file in your mobile.



Screen1

mq6

Good

ultrasonic1

24

ultrasonic2

11

cylwt

24

CONCLUSION:-

This project presents the design and implementation of an interactive kitchen monitoring system with the control, communication and web-enabled measurement and control systems. The web based monitor and automatic control of equipment is forming a trend in automation field.The design is completely wireless and integrated with the software to form a low cost, robust and easily operable system.

FUTURE SCOPE:-

Would be focused more on increasing sensors on this system to fetch more data especially with regard to identify the leakage of gas before the destroying of everything so that all can stay safe.