REPORT ON SMART KITCHEN USING IBM CLOUD

By:- VARSHA RANI PANDA
JAIKIRAN R

INDEX

INTRODUCTION

1.1 OVERVIEW

Kitchen is the unique place, or we can say the heart of the home or hotel industries. It is the place where one of the basic needs i.e. food is prepared. It is the common center of social activities of all the family members who share their feelings or emotions. It is equipped with all basic amenities.

With a variety of activities conducted in the kitchen, will certainly make the kitchen temperature conditions quickly changed. In addition to temperature, the use of gas stoves has a high fire risk. Based on that, the use of IoT in the kitchen is necessary to keep the kitchen air condition always be comfortable and to reduce risk from the use of gas stoves.

It is the smart kitchen system based IoT IBM technology bringing change to each and every common man's life by making everything smart and intelligent. The aim of this project is to create a smart Iot based kitchen giving people the live data about the cylinder weight, jars' quantity, gas leakage for efficient monitoring of their kitchen and reducing the life threat from leakage or fire and last minute hurry in refilling the cylinder or jars.

We will use the following services and apps:-

- IBM IoT Platform
- IBM Node-red
- MIT App
- Fast2SMS

1.2 PURPOSE

Safety factor is the main aspect that must be taken into account during the activity in the kitchen. The existence of gas leakage, uncontrolled fire and excessive temperatures must be quickly identified and addressed.

Its very often to get the news of fire in a building due to gas leakage. This is a serious problem that needs a solution. A lot of people loose their life due to this uncertain outcome.

Other than this , the project will also manage the essentials jars in our kitchen.
In today's busy and hectic world, people don't remember these small issues like sugar or salt (or any type of pulses or spices) is going to be finish. 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.

The purpose of this project is to make prototype of kitchen security system using Internet Of Things.

LITERATURE SURVEY

2.1 EXISTING PROBLEMS

Safety is the major issue in kitchen. Many houses get burnt due to the fire caught in leakage of gas. We often get the news of these accidents. Therefore, people can opt for smart kitchen facility and can make their houses and themselves safe from such drastic situation.

Shortage or sudden emptiness of food jars can be handled from the Smart Kitchen.
In big restaurants and hotels, management of these are really hectic and crucial for their business. So, this can be handled by our Smart Kitchen solution.

2.2 PROPOSED SOLUTION

To overcome the problem of traditional kitchen, we proposed a solution by building a smart kitchen that works with IoT (Internet of Things) and cloud computing.

1) **IoT**: The Internet of Things (IoT) is a system of interrelated computing devices, mechanical and digital machines provided with unique keys and ids and the ability to transfer data without requiring human-to-human interaction. In this project we integrate this technology with our kitchen to make our kitchen advanced and smart by solving the problems. Although much of the work has been done until today to realize the Internet of Things (IoT) into practice, most of the work focuses on resource-constrained nodes, rather than linking the existing to the IoT network. The Internet of things (IOTs) is a network of physical objects or things embedded with electronic, software, sensors and connectivity to enable objects to exchange data with manufacturer, operators and connected devices.

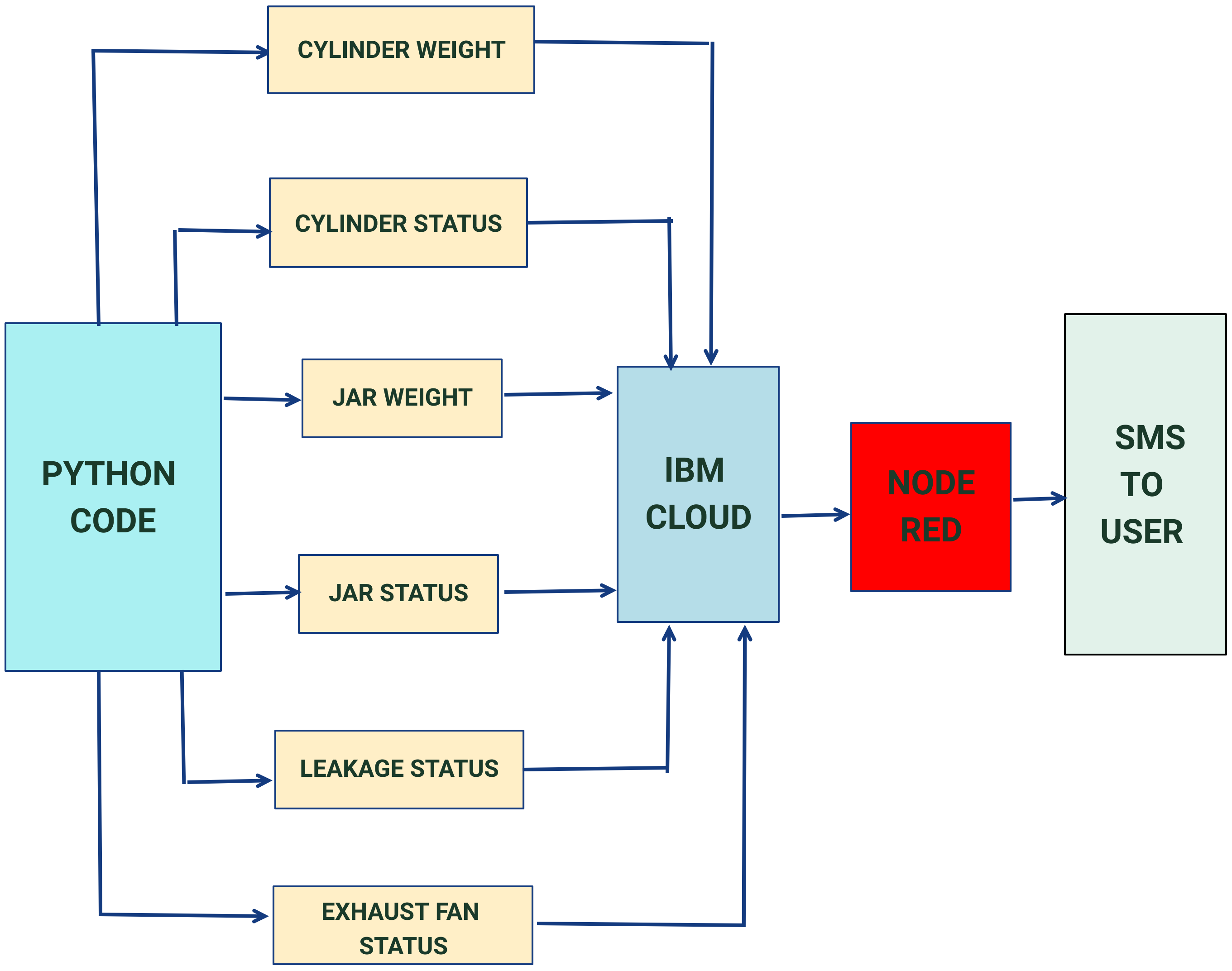
It can be described as connecting everyday objects like smart phones, sensors and actuators to the Internet where the devices are intelligently linked together enabling new forms of communication between things and people, and between things themselves. IOT allows objects to be sensed and controlled remotely across existing network infrastructure. It also provides efficiency , accuracy, comfort and economic benefit.

Now anyone, from anytime and anywhere can have connectivity for anything and it is expected that these connections will extend and create an entirely advanced dynamic network of IOTs. Our work here tries to enhance Internet oriented approach with semantic oriented method, both of which are required to build practical, complex IOT applications, which are expected on rich embedded devices

2) **Cloud computing**: Cloud computing is the on-demand availability of computer system resources, especially data storage(cloud storage) and computing power, without direct active management by the user. With the help of cloud computing technology, we can store the data collected by the sensors and also weather conditions in that particular area and use it for future analysis. These data are being used to provide predictive insights in farming operations, drive real-time operational decisions, and redesign business processes.

THEORETICAL ANALYSIS

3.1 BLOCK DIAGRAM

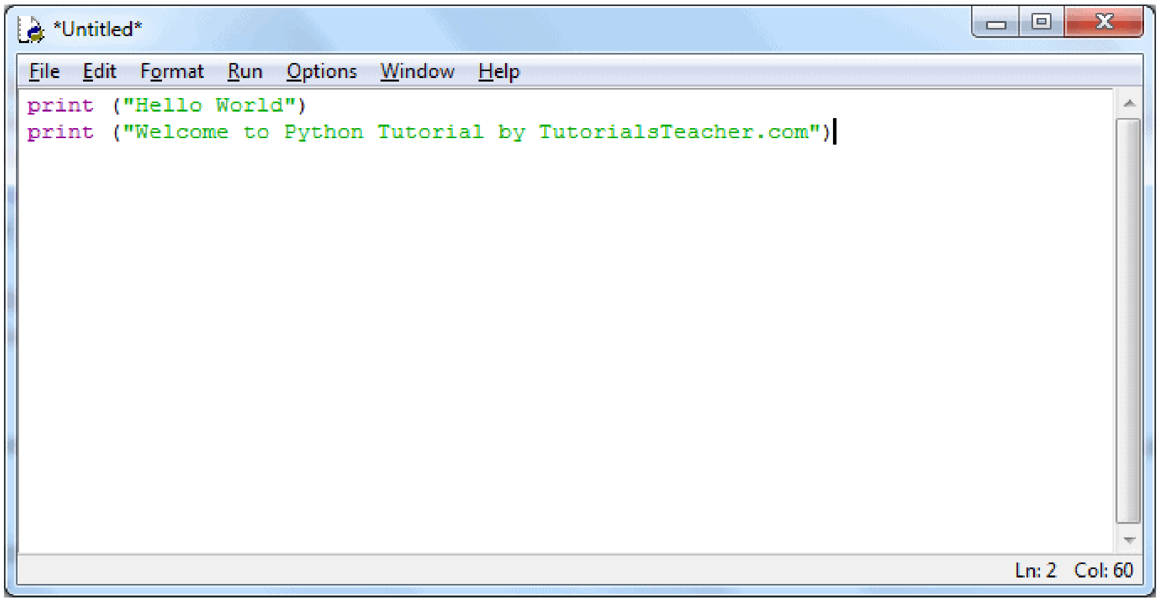


3.2 SOFTWARE SOLUTIONS

PYTHON IDLE

IDLE (Integrated Development and Learning Environment) is an integrated development environment (IDE) for Python. The Python installer for Windows contains the IDLE module by default.
IDLE can be used to execute a single statement just like Python Shell and also to create, modify and execute Python scripts. IDLE provides a fully-featured text editor to create Python scripts that includes features like syntax highlighting, auto completion and smart indent. It also has a debugger with stepping and breakpoints features.

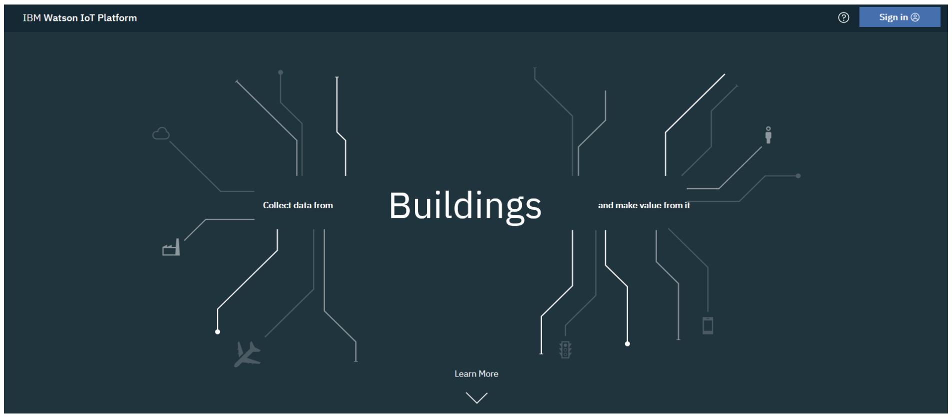
In this project, python is used to write a code which interacts with IBM Cloud platform and sends the data there. Its also alerts the user by using Fast2SMS.
The python code is the one which controls other aspects in the project.



```
print ("Hello World")
print ("Welcome to Python Tutorial by TutorialsTeacher.com")
```

IBM Watson IoT Platfrom

IBM Watson IoT Platform is a foundational cloud offering that can connect and control IoT sensors, appliances, homes, and industries. Built on IBM Cloud, Watson IoT Platform provides an extensive set of built-in and add-on tools. Use these tools to process IoT data with real-time and historical analytics, extract key performance indicators (KPIs) from your data, add “smarts” in the cloud for non-smart products, and securely connect your own apps and existing tools to the Watson IoT Platform infrastructure.



The following key features are included with the Watson IoT Platform solution:

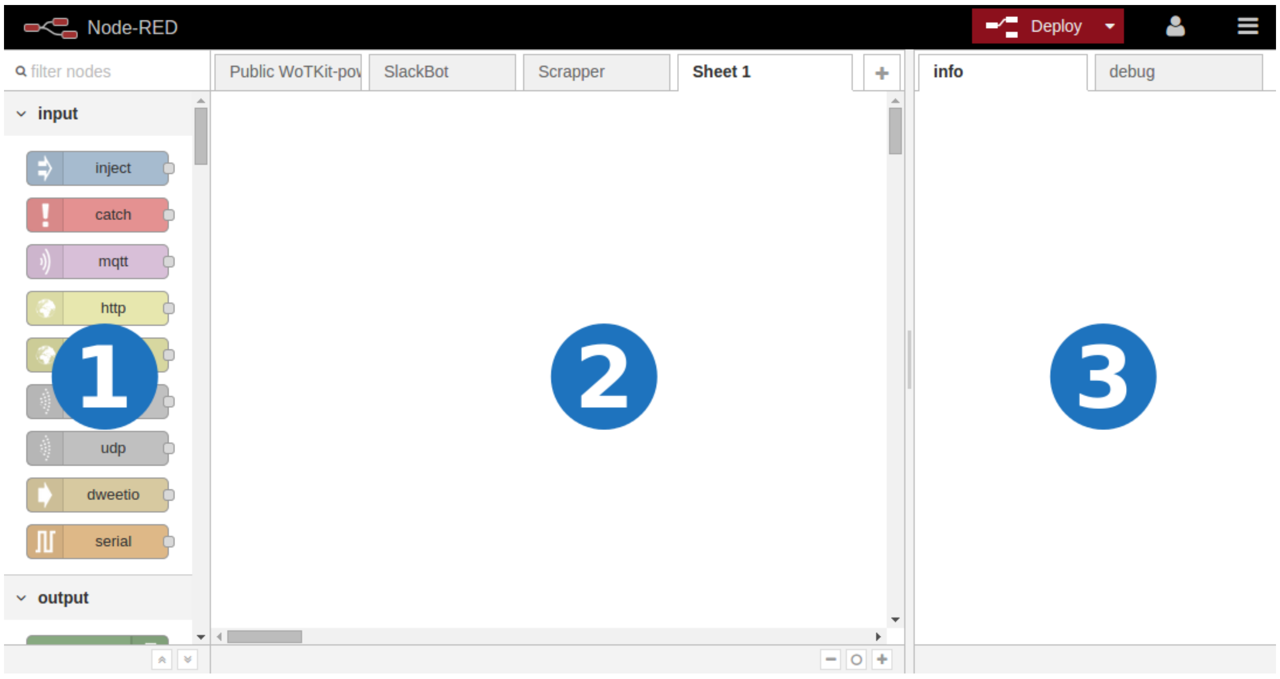
- Connect
  It connects IoT data in the form of MQTT events from any source by using Platform Service to register, connect, and control your IoT devices your IoT devices.
- Capture
  It captures your IoT data for the short-term in the Cloudant NoSQL DB solution, which gives you direct access to the last 30 days’ worth of data for your devices.
- Monitor
  You can get an overview of your assets and entities with built-in monitoring dashboards.
- Manage

The Watson IoT Platform SaaS solution is managed as a unit, with full status and connection through the Watson IoT Platform dashboard.

- Monitor
  You can use the Watson IoT Platform dashboard to monitor your data consumption and usage for each included service in your account to correctly size your solution needs.

NODE - RED

Node-RED is a powerful tool for building Internet of Things (IoT) applications with a focus on simplifying the 'wiring together' of code blocks to carry out tasks. It uses a visual programming approach that allows developers to connect predefined code blocks, known as 'nodes', together to perform a task. The connected nodes, usually a combination of input nodes, processing nodes and output nodes, when wired together, make up a 'flows'.
Although Node-RED was originally designed to work with the Internet of Things, i.e. devices that interact and control the real world, as it has evolved, it has become useful for a range of applications.

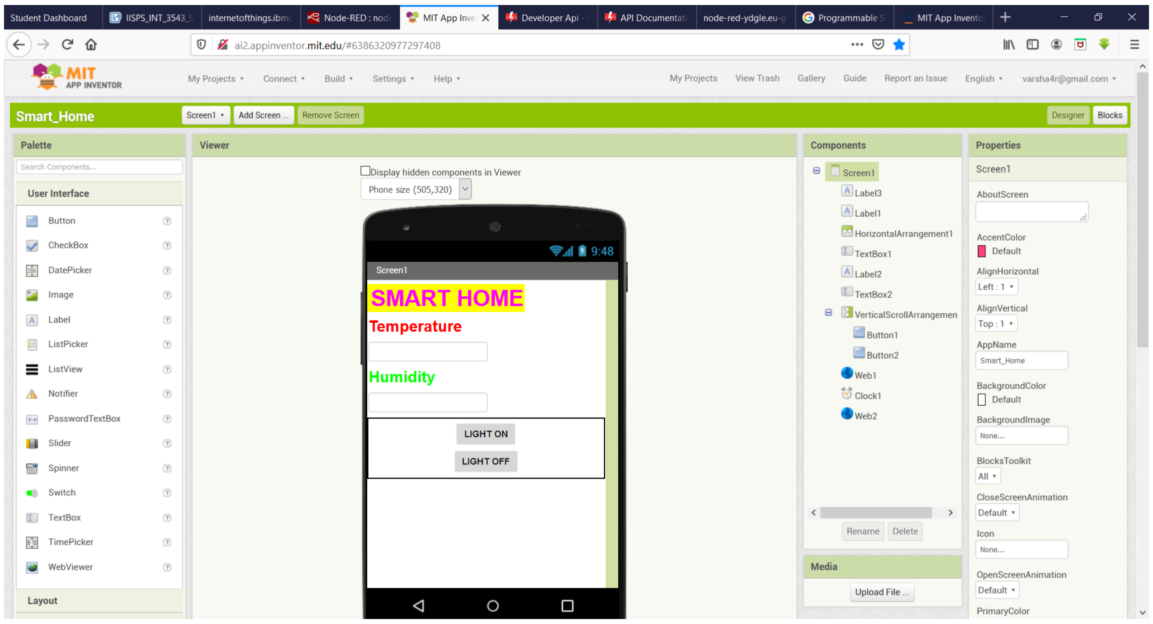


Node_RED has 3 main components:

1. Node Panel
2. Sheets Panel
3. Info and Debug Panel.

<u>MIT-APP</u>

MIT App Inventor is an online platform designed to teach computational thinking concepts through development of mobile applications.
The MIT App Inventor user interface includes two main editors: the design editor and the blocks editor. The design editor, or designer, is a drag and drop interface to lay out the elements of the application's user interface (UI). The blocks editor is an environment in which app inventors can visually lay out the logic of their apps using color-coded blocks that snap together like puzzle pieces to describe the program. To aid in development and testing, App Inventor provides a mobile app called the App Inventor Companion that developers can use to test and adjust the behavior of their apps in real time. In this way, anyone can quickly build a mobile app and immediately begin to iterate and test.



<u>Fig: Designer editor</u>

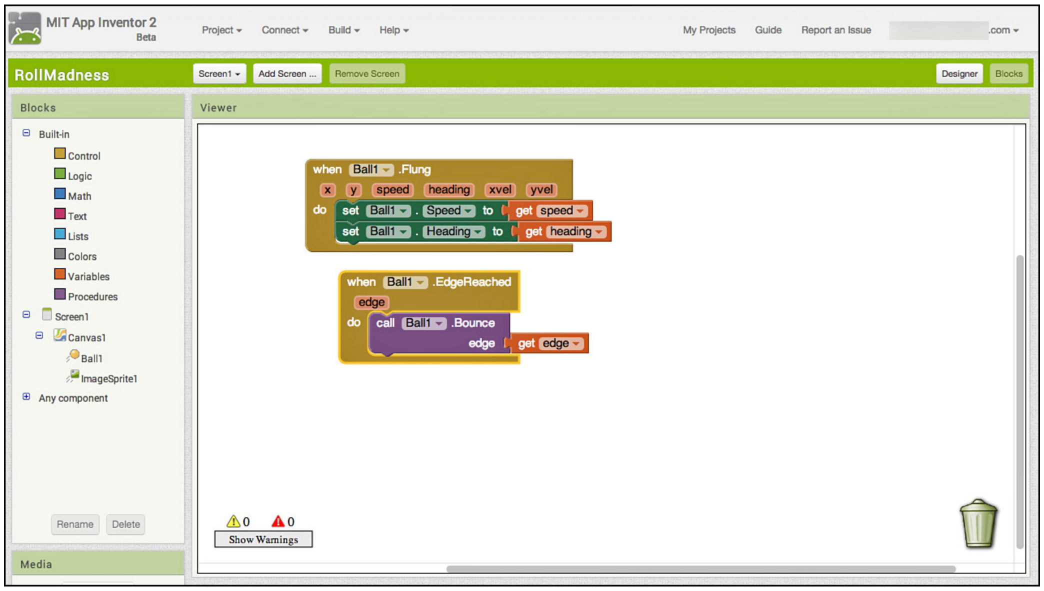
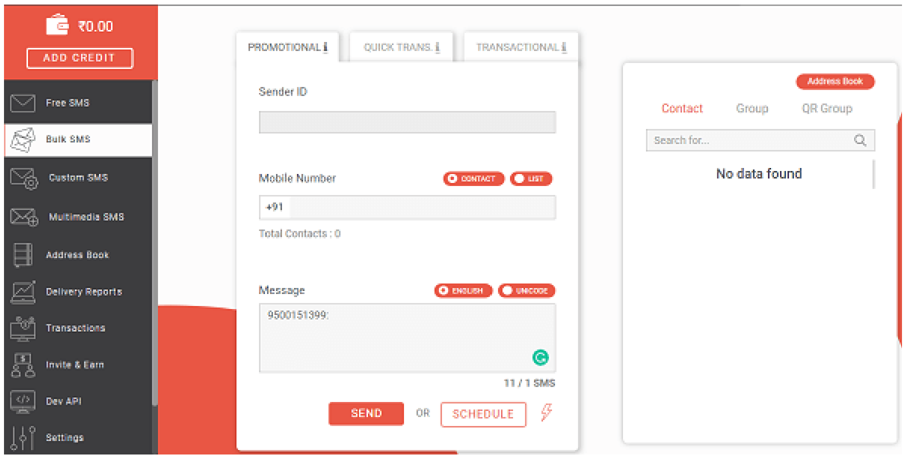


Fig: Block Editor

Fast2SMS

Fast2SMS is a popular bulk SMS service provider in India. It was started in 21st July 2011.
It can send messages without approval and that too for both DND and non DND numbers. Users can use their own sender ID and can import bulk contacts from list and create QR. Also SMS can be sent 24*7 without any time restriction.
In the project, this is used to give important alerts to the user.

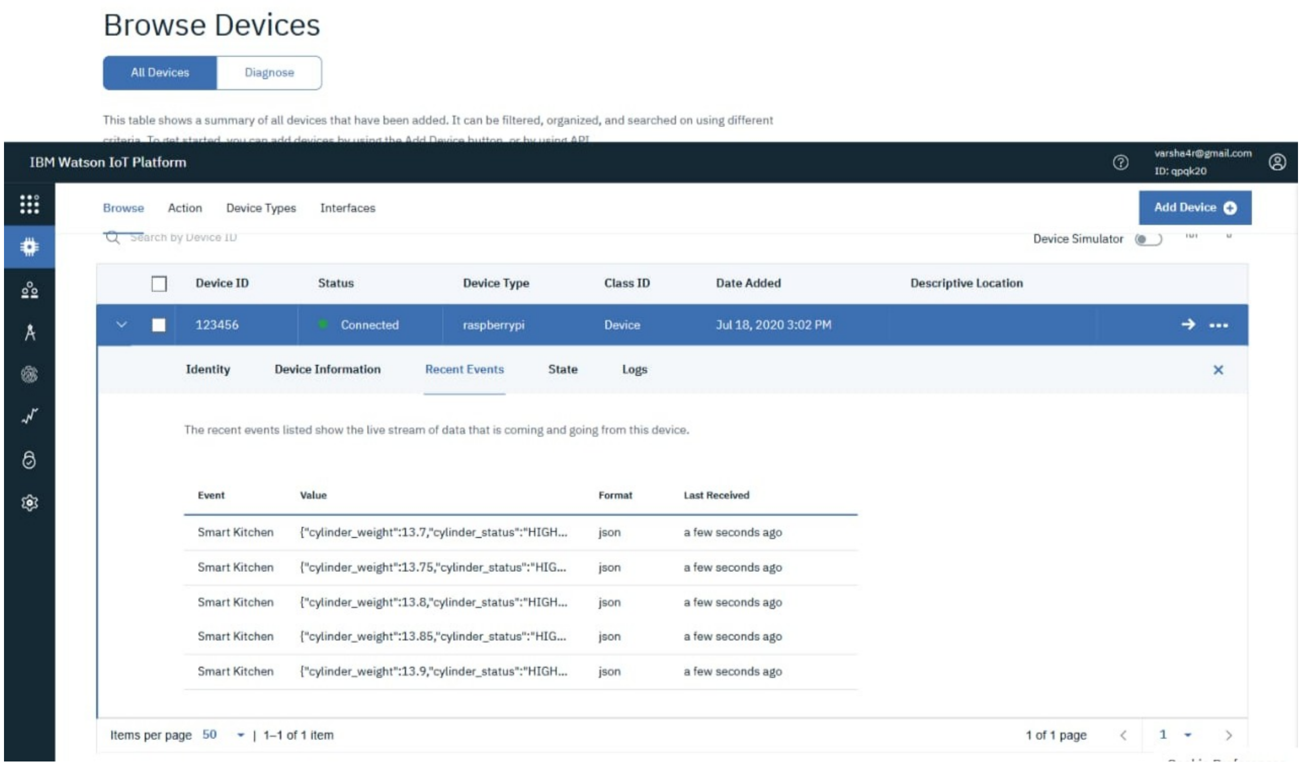


EXPERIMENTAL INVESTIGATIONS

The project controls the gas leakage by switching ON the exhaust fan. It also gives the status of the cylinder weight and jar weight. It gives message to the user whenever the cylinder or jar is empty and when there is leakage of gas.

STEP 1: Firstly create an IBM Watson IoT Platfrom and save the credentials because they will be required in future for the project.

STEP 2: We have to give the credentials (device id,device type, authentication token, organization id etc.) in the python code to connect it to IBM IOT Platform. When the code is running then it starts publishing the data to the cloud. We can view the simulated data in cloud platform by clicking device type and then clicking recent activity.



This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123456	Connected	raspberrypi	Device	Jul 18, 2020 3:02 PM	

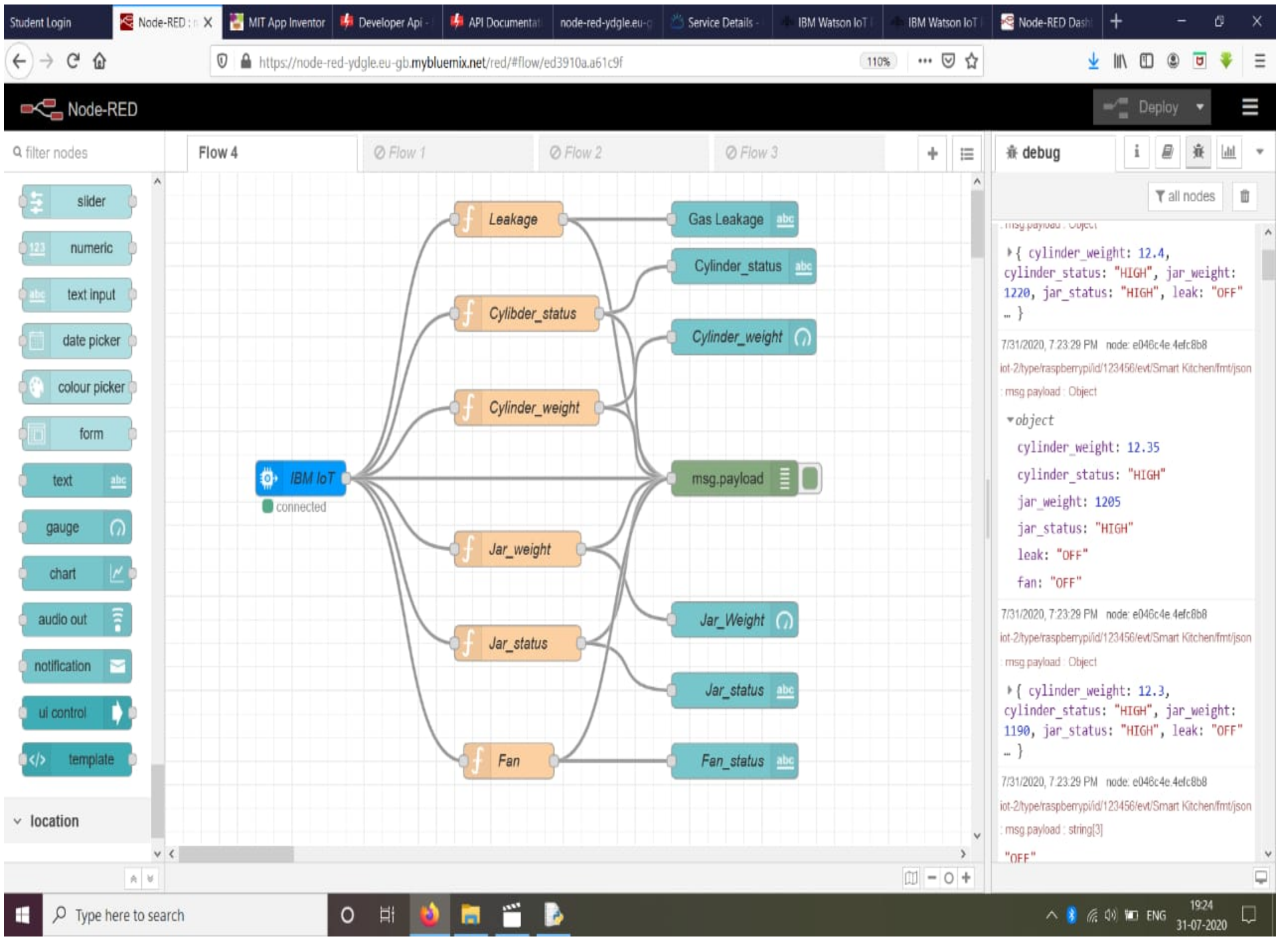
The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Smart Kitchen	{"cylinder_weight":13.7,"cylinder_status":"HIGH...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.75,"cylinder_status":"HIG...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.8,"cylinder_status":"HIGH...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.85,"cylinder_status":"HIG...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.9,"cylinder_status":"HIGH...	json	a few seconds ago

STEP 3: Configure the Node red and install the required nodes to retrieve the data

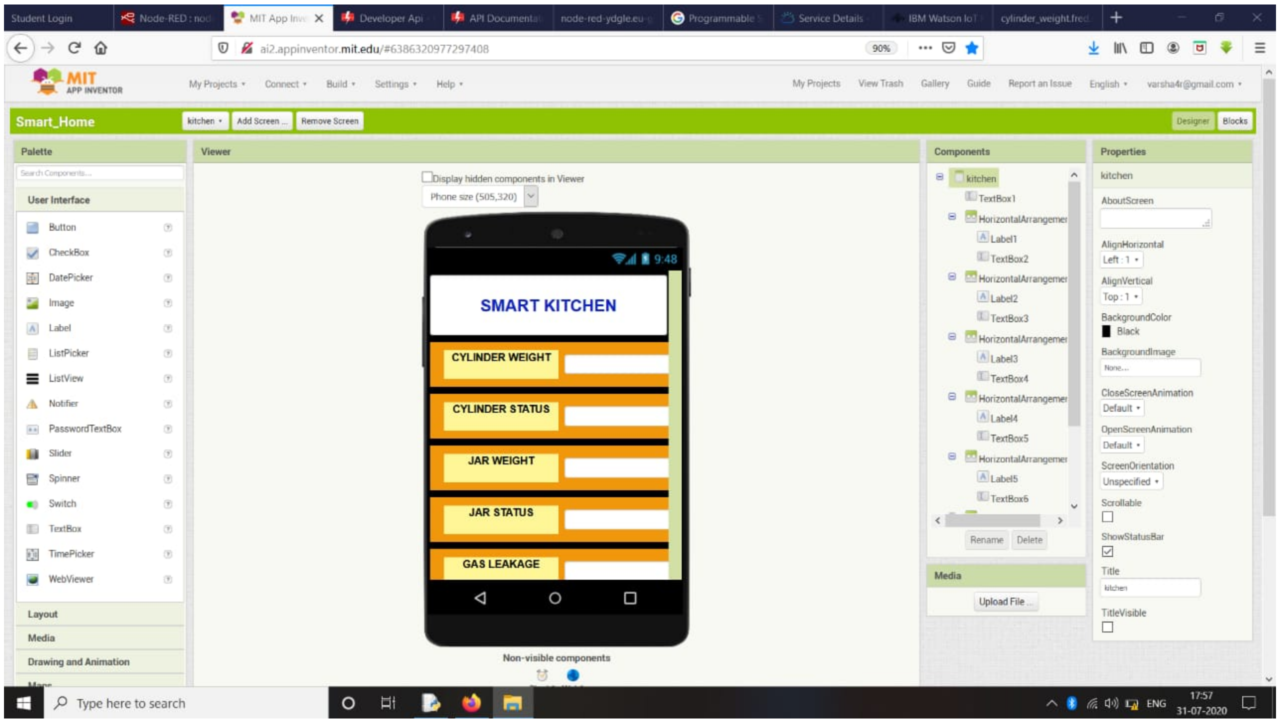
from IBM IoT Platform and send commands to device.

STEP 4: Drag and drop the required nodes from node panel to sheet panel to get the cylinder weight and status and jar weight and status and status of leakage and exhaust fan. Use http request and response to send the status to the MIT App.

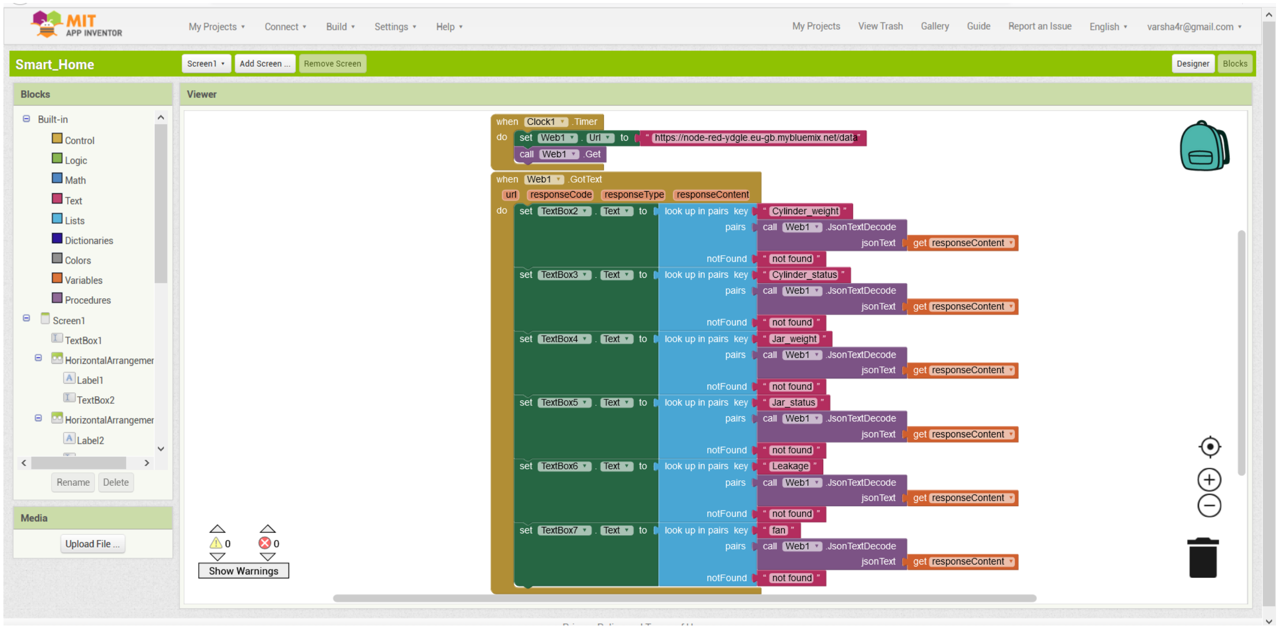


```
: msg.payload : Object
▸{ cylinder_weight: 12.4,
cylinder_status: "HIGH", jar_weight:
1220, jar_status: "HIGH", leak: "OFF"
… }
7/31/2020, 7:23:29 PM   node: e046c4e.4efc8b8
iot-2/type/raspberrypi/id/123456/evt/Smart Kitchen/fmt/json
: msg.payload : Object
▾object
  cylinder_weight: 12.35
  cylinder_status: "HIGH"
  jar_weight: 1205
  jar_status: "HIGH"
  leak: "OFF"
  fan: "OFF"
7/31/2020, 7:23:29 PM   node: e046c4e.4efc8b8
iot-2/type/raspberrypi/id/123456/evt/Smart Kitchen/fmt/json
: msg.payload : Object
▸{ cylinder_weight: 12.3,
cylinder_status: "HIGH", jar_weight:
1190, jar_status: "HIGH", leak: "OFF"
… }
7/31/2020, 7:23:29 PM   node: e046c4e.4efc8b8
iot-2/type/raspberrypi/id/123456/evt/Smart Kitchen/fmt/json
: msg.payload : string[3]
"OFF"
```

STEP 5: For mobile application use MIT App Inventor and create an application for viewing details of Cylinder weight and status and also Jar weight and status and if any gas leakage, these are shown through it

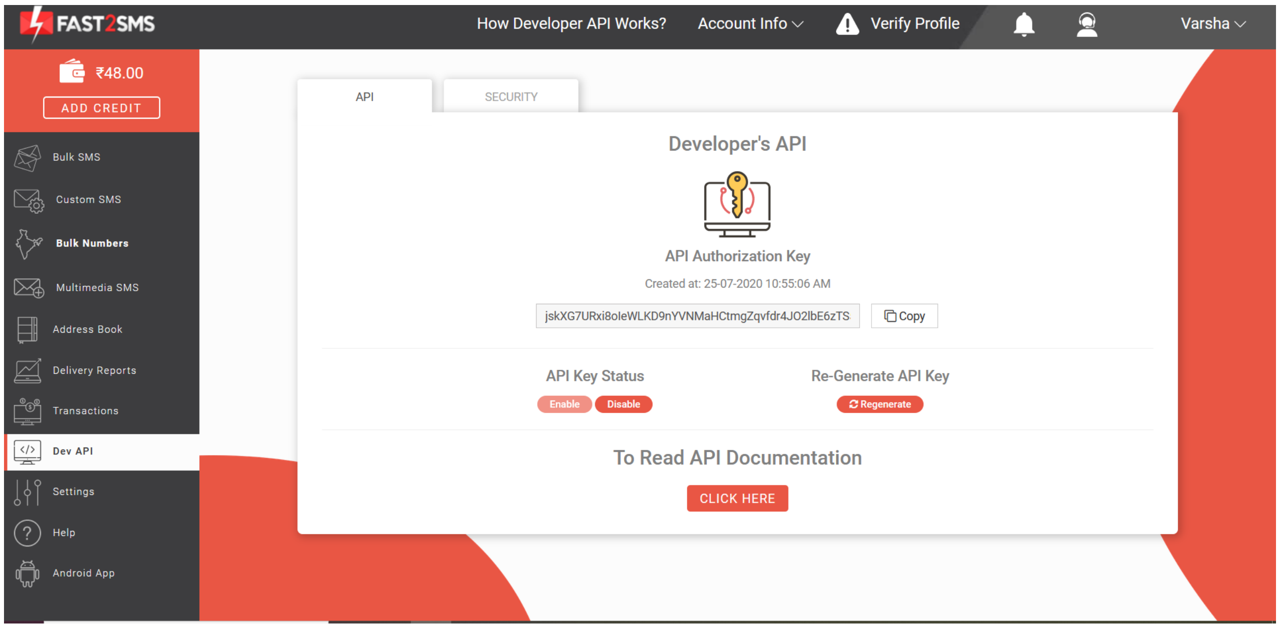


STEP 6: In the blocks editor of MIT app inventor, following blocks are required to get data from the cloud and then send it to the app in our mobile phones.



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

when Web1 .GotText
 url  responseCode  responseType  responseContent
do  set TextBox2 . Text to  look up in pairs  key " Cylinder_weight "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox3 . Text to  look up in pairs  key " Cylinder_status "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox4 . Text to  look up in pairs  key " Jar_weight "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox5 . Text to  look up in pairs  key " Jar_status "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox6 . Text to  look up in pairs  key " Leakage "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
    set TextBox7 . Text to  look up in pairs  key " fan "
                                              pairs  call Web1 .JsonTextDecode
                                                                    jsonText  get responseContent
                                           notFound " not found "
```

STEP 7: Now open an account in Fast2SMS and use the API key in the python code to send alert to the user.



Developer's API

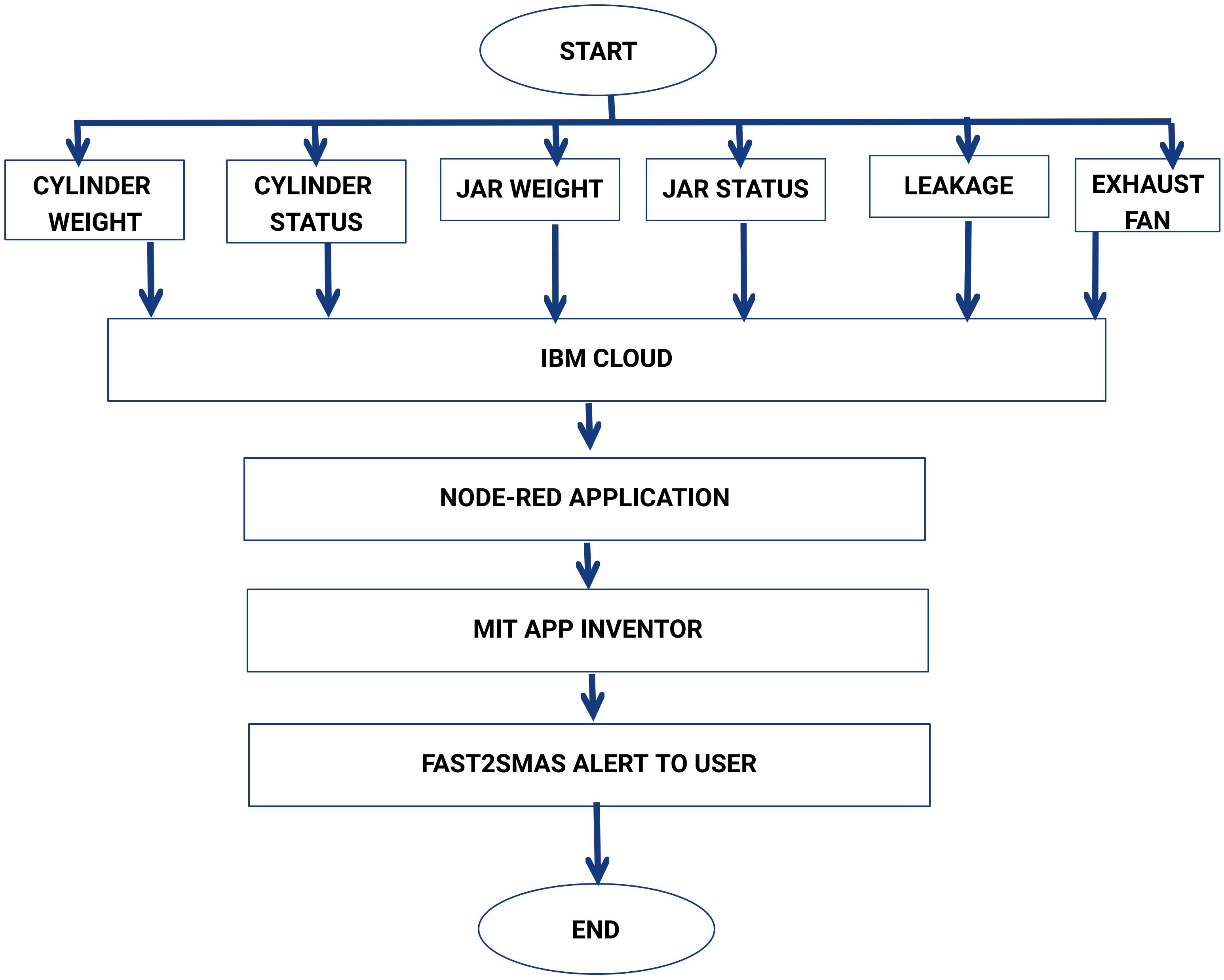
API Authorization Key

Created at: 25-07-2020 10:55:06 AM

jskXG7URxi8oleWLKD9nYVNMaHCtmgZqvfdr4JO2lbE6zTS

To Read API Documentation

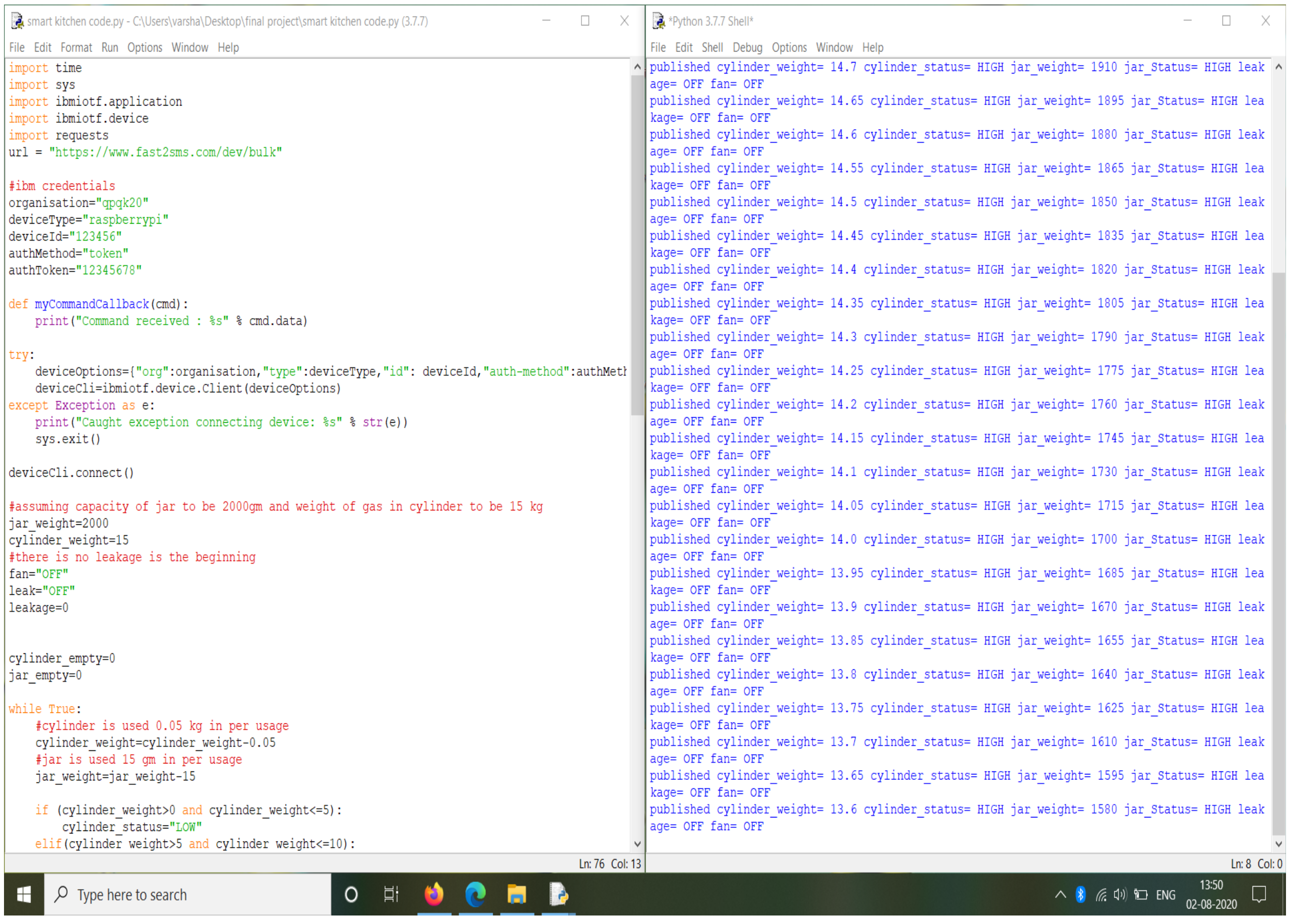
FLOWCHART


START
CYLINDER WEIGHT
CYLINDER STATUS
JAR WEIGHT
JAR STATUS
LEAKAGE
EXHAUST FAN
IBM CLOUD
NODE-RED APPLICATION
MIT APP INVENTOR
FAST2SMAS ALERT TO USER
END

RESULTS

1. When we connect with IoT Platform, we get values like cylinder and jar weight and also status of leakage and whether exhaust fan is ON or not which can be viewed and analyzed in cloud.

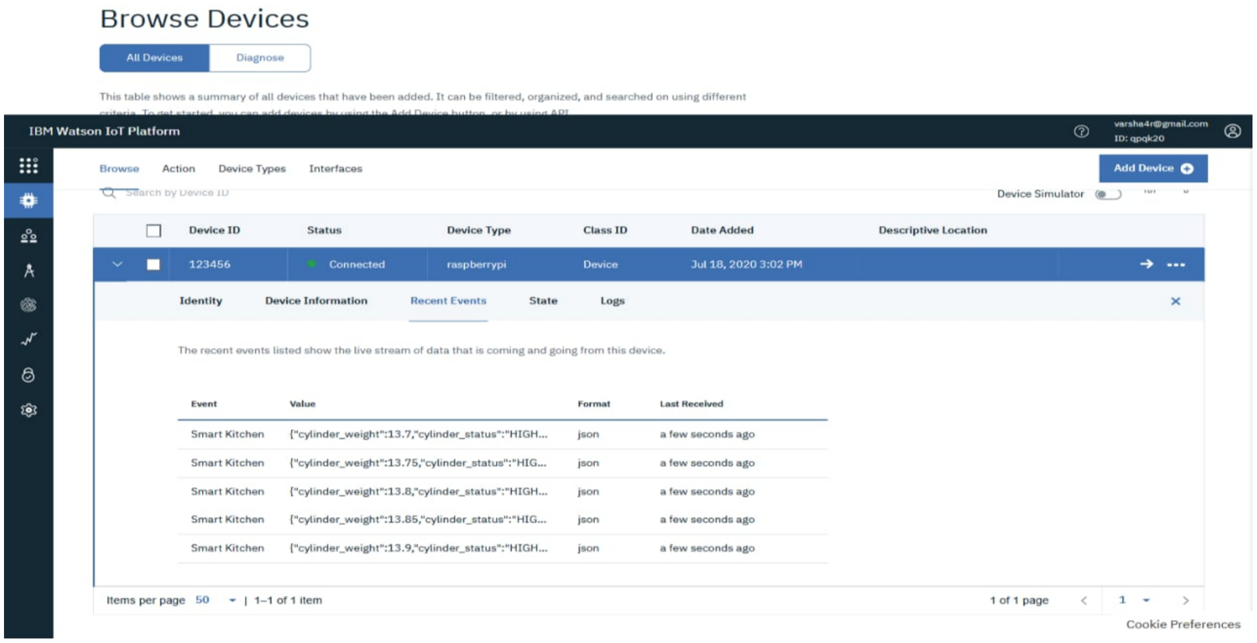
2. Python code to publish and subscribe to IBM IoT Platform for sending sensor values and receiving the commands. The details can be viewed in cloud and web page user interface.


smart kitchen code.py - C:\Users\varsha\Desktop\final project\smart kitchen code.py (3.7.7)
File Edit Format Run Options Window Help

```python
import time
import sys
import ibmiotf.application
import ibmiotf.device
import requests
url = "https://www.fast2sms.com/dev/bulk"

#ibm credentials
organisation="qpqk20"
deviceType="raspberrypi"
deviceId="123456"
authMethod="token"
authToken="12345678"

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

try:
    deviceOptions={"org":organisation,"type":deviceType,"id": deviceId,"auth-method":authMeth
    deviceCli=ibmiotf.device.Client(deviceOptions)
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()

deviceCli.connect()

#assuming capacity of jar to be 2000gm and weight of gas in cylinder to be 15 kg
jar_weight=2000
cylinder_weight=15
#there is no leakage is the beginning
fan="OFF"
leak="OFF"
leakage=0


cylinder_empty=0
jar_empty=0

while True:
    #cylinder is used 0.05 kg in per usage
    cylinder_weight=cylinder_weight-0.05
    #jar is used 15 gm in per usage
    jar_weight=jar_weight-15

    if (cylinder_weight>0 and cylinder_weight<=5):
        cylinder_status="LOW"
    elif(cylinder_weight>5 and cylinder_weight<=10):
```

```
published cylinder_weight= 14.7 cylinder_status= HIGH jar_weight= 1910 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 14.65 cylinder_status= HIGH jar_weight= 1895 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 14.6 cylinder_status= HIGH jar_weight= 1880 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 14.55 cylinder_status= HIGH jar_weight= 1865 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 14.5 cylinder_status= HIGH jar_weight= 1850 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 14.45 cylinder_status= HIGH jar_weight= 1835 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 14.4 cylinder_status= HIGH jar_weight= 1820 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 14.35 cylinder_status= HIGH jar_weight= 1805 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 14.3 cylinder_status= HIGH jar_weight= 1790 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 14.25 cylinder_status= HIGH jar_weight= 1775 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 14.2 cylinder_status= HIGH jar_weight= 1760 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 14.15 cylinder_status= HIGH jar_weight= 1745 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 14.1 cylinder_status= HIGH jar_weight= 1730 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 14.05 cylinder_status= HIGH jar_weight= 1715 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 14.0 cylinder_status= HIGH jar_weight= 1700 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 13.95 cylinder_status= HIGH jar_weight= 1685 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 13.9 cylinder_status= HIGH jar_weight= 1670 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 13.85 cylinder_status= HIGH jar_weight= 1655 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 13.8 cylinder_status= HIGH jar_weight= 1640 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 13.75 cylinder_status= HIGH jar_weight= 1625 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 13.7 cylinder_status= HIGH jar_weight= 1610 jar_Status= HIGH leak
age= OFF fan= OFF
published cylinder_weight= 13.65 cylinder_status= HIGH jar_weight= 1595 jar_Status= HIGH lea
kage= OFF fan= OFF
published cylinder_weight= 13.6 cylinder_status= HIGH jar_weight= 1580 jar_Status= HIGH leak
age= OFF fan= OFF
```

3. These data are sent to the IBM Cloud because of the credentials used in the python code. This can be viewed in the recent events.



Browse Devices

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different

	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
	123456	Connected	raspberrypi	Device	Jul 18, 2020 3:02 PM	

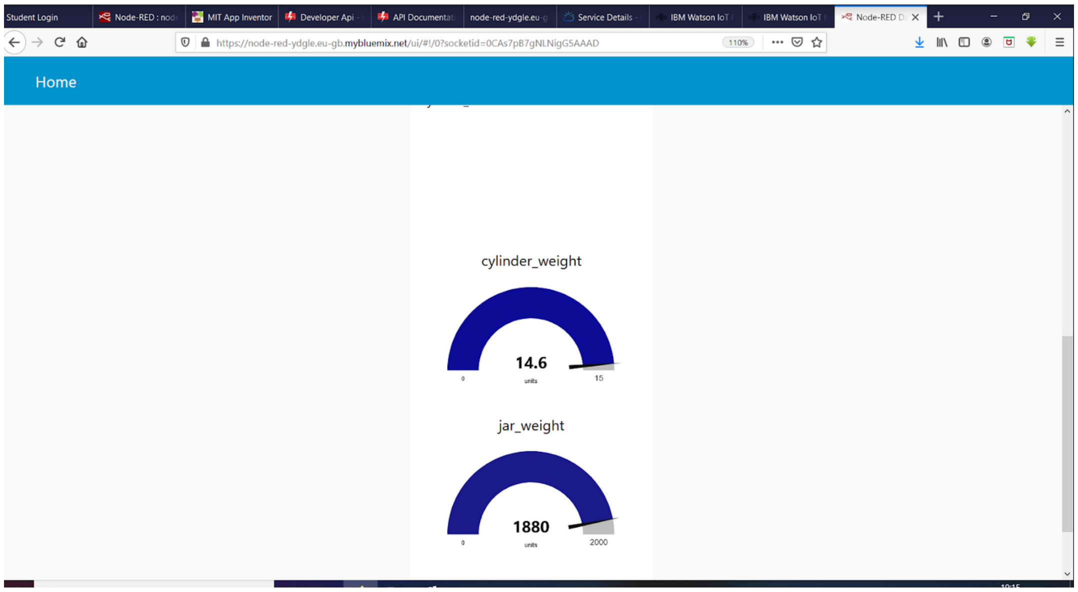
The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Smart Kitchen	{"cylinder_weight":13.7,"cylinder_status":"HIGH...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.75,"cylinder_status":"HIG...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.8,"cylinder_status":"HIGH...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.85,"cylinder_status":"HIG...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":13.9,"cylinder_status":"HIGH...	json	a few seconds ago

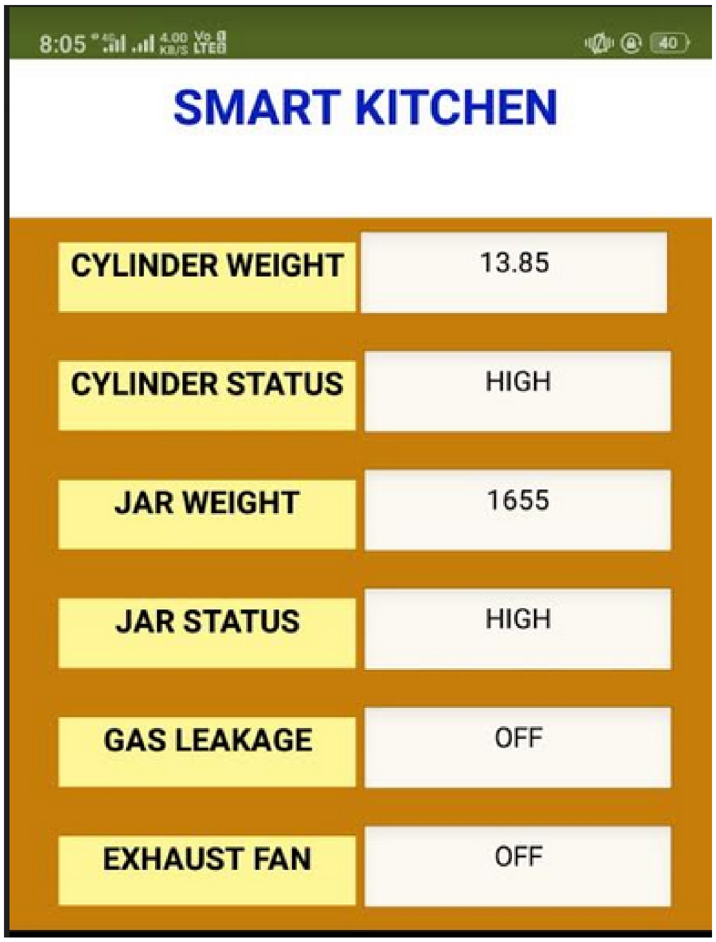
Items per page 50 | 1–1 of 1 item — 1 of 1 page

4. Now the data are also getting published in the Node-Red. We can view this in the debug section.

5. The data can also be viewed in node-red UI.



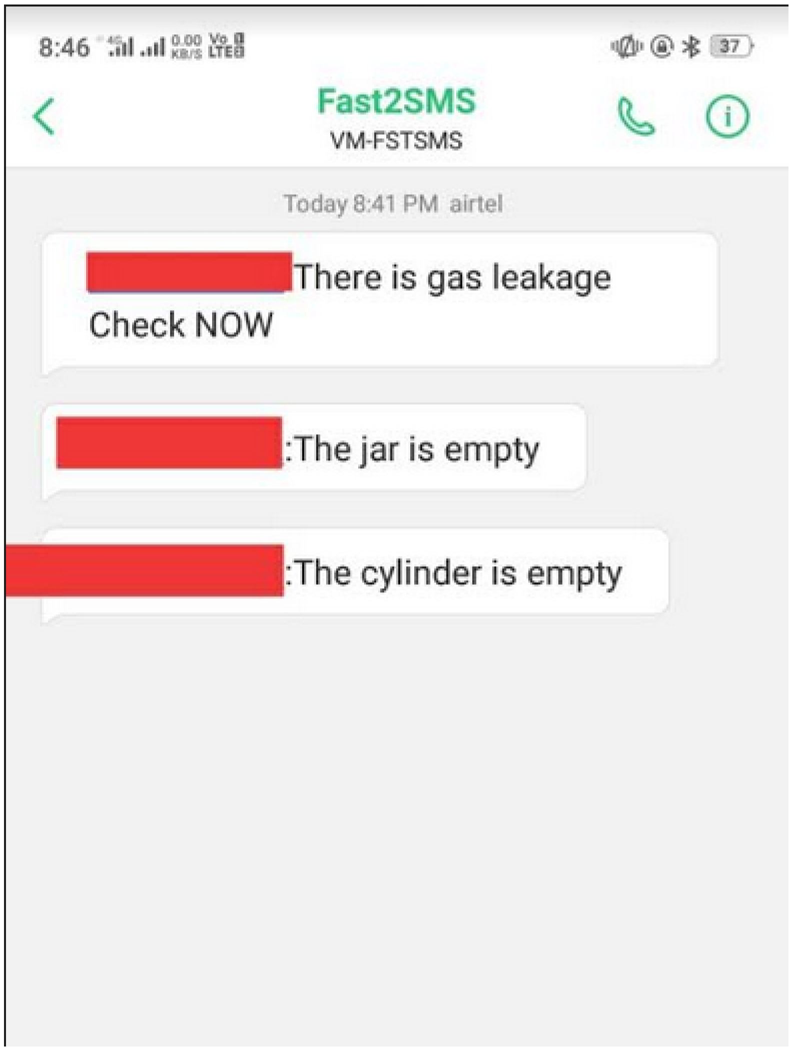
6. From the node red the data is sent to the MIT app in the user's phone.



SMART KITCHEN

CYLINDER WEIGHT	13.85
CYLINDER STATUS	HIGH
JAR WEIGHT	1655
JAR STATUS	HIGH
GAS LEAKAGE	OFF
EXHAUST FAN	OFF

7. SMS alert in the sent to the user.



8:46

Fast2SMS
VM-FSTSMS

Today 8:41 PM airtel

[redacted]There is gas leakage Check NOW

[redacted]:The jar is empty

[redacted]:The cylinder is empty

ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- Smart kitchen ensures safety from the leakage.
- People can visualize cylinder's quantity, jar quantity, sunlight in real time and remotely
- They can also get important information when the cylinder or jar is empty.
- Smart kitchen can also have such a feature that whenever jar is empty it can automatically call the nearest grocery and order that itself.
- It can also automatically book the cylinder when cylinder's quantity is low.
- one of the interesting feature is that if there is gas leakage then the exhaust fan is automatically turned ON so that gas can get out of the kitchen.
- Smart kitchen is using Fast2SMS in which the SMS can be sent to more than one members of the house.

DISADVANTAGES

- One huge disadvantage of Smart Kitchen is that it requires an unlimited or continuous internet connection to be successful. This means that in rural communities, especially in the developing countries it is not accessible by all the people.
- Also this is expensive for some people.

APPLICATIONS

The application allows users to program the various appliances. It is also able to check the status of the appliance, as well as controlling the power consumption of the house and its cost

1)GROCERY ORDERING

2)SMARTER FRIDGE CAM

3)NUTRIBULLET BALANCE

4)DROP SCALE

5)EMBER TEMPERATURE CONTROL SMART MUG

CONCLUSION

Our Smart Kitchen using IoT system has been designed, constructed and tested. The result obtained from the tests carried out shows that the system is capable of sending SMS alerts whenever there is gas concentration at the inputs of the gas sensors. Hence this system can be used in homes and public buildings such as hotels and restaurants. Smart kitchen provides you all the automation features that include safety features over gas leakage detection system. For this we are using gas sensors, weight sensors. Gas sensors are used to detect the leakage of a gas in the system, weight sensors are used to detect the weight of the gas cylinder.If there is any leakage in the gas the gas sensors will detect and the exhaust fan will turn ON. and if the cylinder is empty, it will send the alert message to the owner. Same thing will repeat with the jars. Server stores information and related data are stored in it; it also stores the information about the hardware, sensors, and also maintains the logs and status of system and also the information

FUTURE SCOPE

Future scope for this project is with the concept of IoT, given below are some of the main future scope of IoT in field of irrigation:

1)Automation and Monitoring Smart Kitchen Based on Internet of Things (IoT):

The kitchen is one of the important places in a house. Safety factor is the main aspect that must be taken into account during the activity in the kitchen. The existence of gas leakage, uncontrolled fire and excessive temperatures must be quickly identified and addressed. Ts.

2)**A Smart Kitchen for Ambient Assisted Living**

The kitchen environment is one of the scenarios in the home where users can benefit from Ambient Assisted Living (AAL) applications. Moreover, it is the place where old people suffer from most domestic injuries. It is based on a modular architecture which integrates a wide variety of home technology (household appliances, sensors, user interfaces, *etc*.) and associated communication standards needs. The system has been evaluated by a large number of real users and carers. Results show a large potential of system functionalities combined with good usability and physical, sensory and cognitive accessibility

3)**Enabling nutrition-aware cooking in a smart kitchen**

We present a smart kitchen that can enhance the traditional meal preparation and cookin processes by raising awareness of the nutritional facts in food ingredients that go into ameal. The goal is to promote healthy cooking.

BIBLIOGRAPHY

https://www.researchgate.net/publication/326349806_Automation_and_Monitoring_Smart_Kitchen_Based_on_Internet_of_Things_IoT

https://www.researchgate.net/publication/326420019_Internet_of_Things_based_system_for_Smart_Kitchen/link/5b527d9c0f7e9b240ff51091/download

https://medium.com/@ritidass29/6-ways-to-reshape-your-kitchen-using-iot-b94727bdb2a3

https://nodered.org/about/

APPENDIX

SOURCE CODE

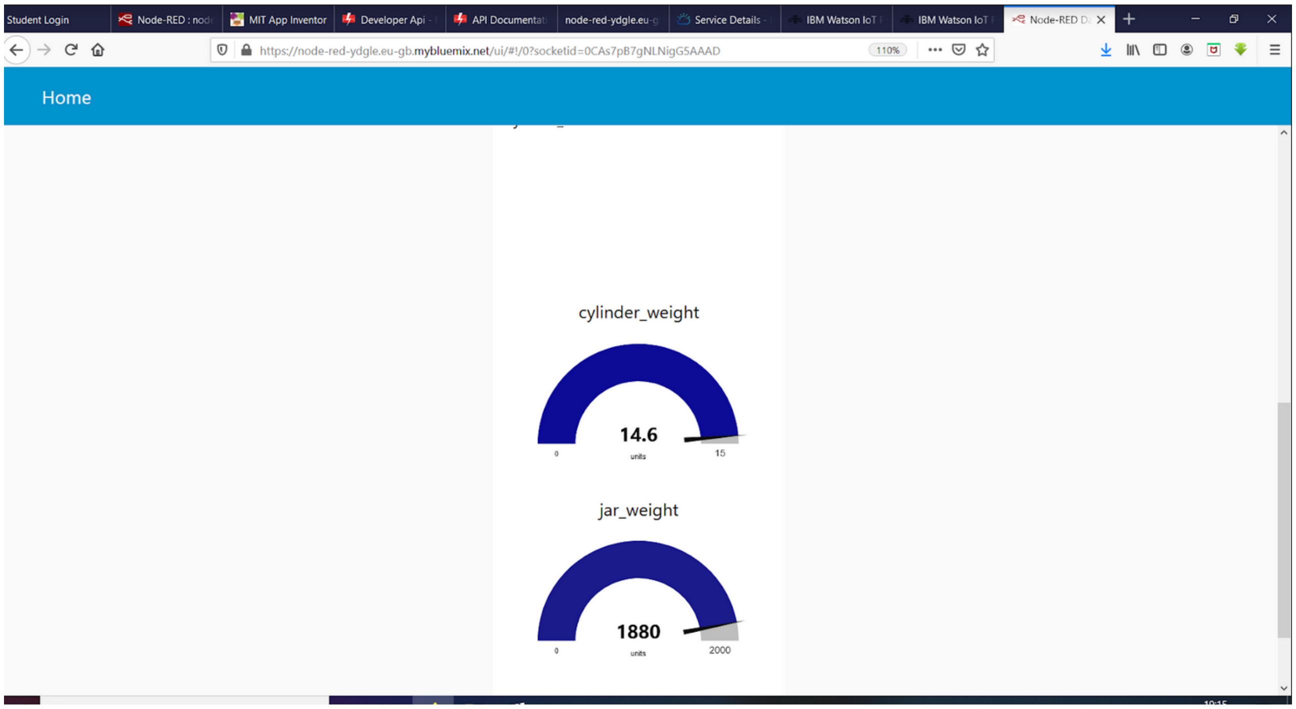
```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import requests
url = "https://www.fast2sms.com/dev/bulk"

#ibm credentials
organisation="qpqk20"
deviceType="raspberrypi"
deviceId="123456"
authMethod="token"
authToken="12345678"

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

try:
   deviceOptions={"org":organisation,"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()
deviceCli.connect()

#assuming capacity of jar to be 2000gm and weight of gas in cylinder to be 15 kg
jar_weight=2000
```

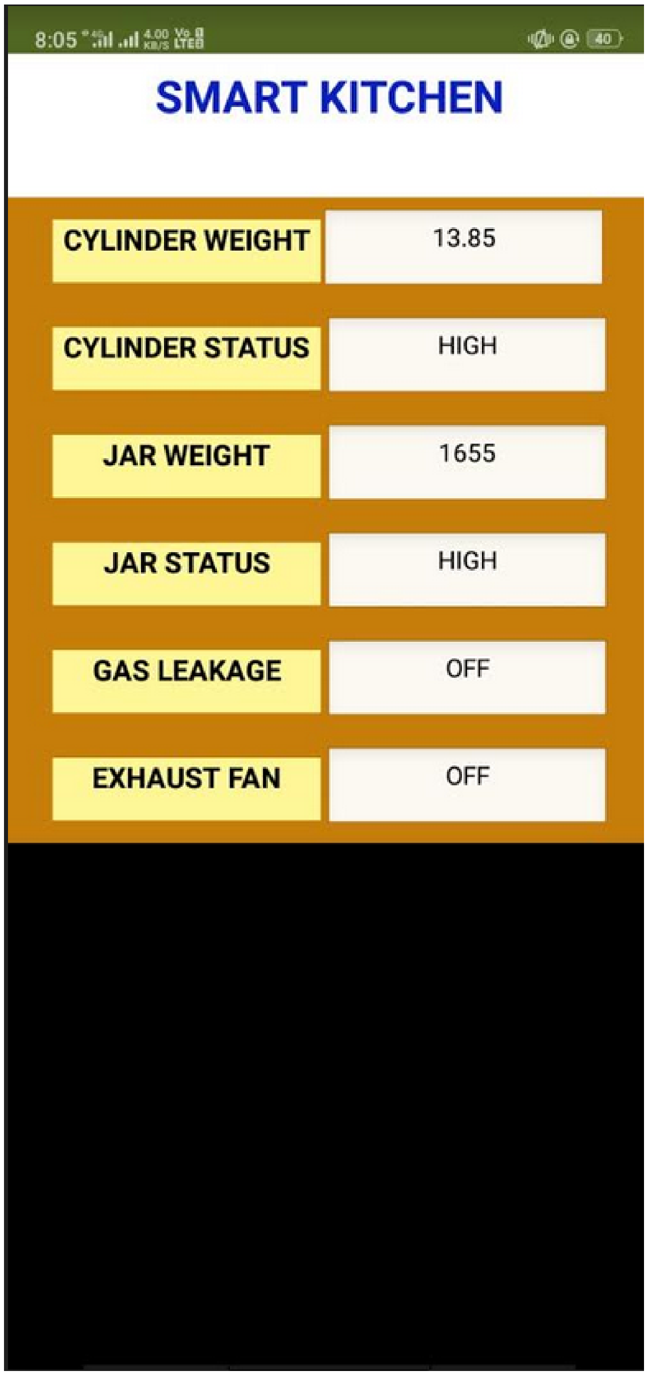
```
cylinder_weight=15
#there is no leakage is the beginning
fan="OFF"
leak="OFF"
leakage=0


cylinder_empty=0
jar_empty=0

while True:
   #cylinder is used 0.05 kg in per usage
   cylinder_weight=cylinder_weight-0.05
   #jar is used 15 gm in per usage
   jar_weight=jar_weight-15

   if (cylinder_weight>0 and cylinder_weight<=5):
      cylinder_status="LOW"
   elif(cylinder_weight>5 and cylinder_weight<=10):
      cylinder_status="MODERATE"
   elif (cylinder_weight>10 and cylinder_weight<=15):
      cylinder_status="HIGH"
   else:
      cylinder_weight=0
      cylinder_status="EMPTY!!"

      if (cylinder_empty==0):
         #SMS is send to user

r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=jskXG7URxi8o
IeWLKD9nYVNMaHCtmgZqvfdr4JO2lbE6zTS5pAFv93mURQGb2tWV8gkTdAl4osL
pOXIC&sender_id=FSTSMS&message=The%20cylinder%20is%20empty&language
=english&route=p&numbers=8130383672')
         print(r.status_code)
```

```
            print("the cylinder is empty")
            cylinder_empty=1


    if(jar_weight<300 and jar_weight>=0):
        jar_status="LOW"
    elif (jar_weight>=300 and jar_weight<1100):
        jar_status="MODERATE"
    elif (jar_weight>=1100 and jar_weight<=2000):
        jar_status="HIGH"
    else:
        jar_weight=0
        jar_status="EMPTY!!"
        if (jar_empty==0):
            print("The jar is empty!!")
            #SMS is send to user

r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=jskXG7URxi8o
IeWLKD9nYVNMaHCtmgZqvfdr4JO2lbE6zTS5pAFv93mURQGb2tWV8gkTdAl4osL
pOXIC&sender_id=FSTSMS&message=The%20jar%20is%20empty&language=engli
sh&route=p&numbers=8130383672')
            print(r.status_code)
            jar_empty=1

    #leakage is increasing
    leakage=leakage+1
    if(leakage==100):
        print(" ALERT!!! GAS LEAKAGE in your kitchen. ")
        #SMS is send to user

r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=jskXG7URxi8o
IeWLKD9nYVNMaHCtmgZqvfdr4JO2lbE6zTS5pAFv93mURQGb2tWV8gkTdAl4osL
pOXIC&sender_id=FSTSMS&message=There%20is%20gas%20leakage%20Check%
20NOW&language=english&route=p&numbers=8130383672')
```

```
        print(r.status_code)
        fan="ON"
        leak="ON"

    data={ 'cylinder_weight' :round(cylinder_weight,2), 'cylinder_status':
cylinder_status, 'jar_weight':jar_weight, 'jar_status':jar_status, 'leak': leak, 'fan':fan}

    def myOnPublishCallback():
        print("published cylinder_weight= %s" %round(cylinder_weight,2),
"cylinder_status= %s" %cylinder_status , "jar_weight= %s" %jar_weight,"jar_Status=
%s" %jar_status,"leakage= %s" %leak,"fan= %s" %fan)
    success =deviceCli.publishEvent("Smart
Kitchen","json",data,qos=0,on_publish=myOnPublishCallback)

    if not success:
        print("not coonected to IoTF")
        time.sleep(0.5)
        device.ClicommandCallback=myCommandCallback
    time.sleep(0.2)
deviceCli.disconnect()
```

UI OUTPUT


Student Login
Node-RED : node
MIT App Inventor
Developer Api
API Documentat
node-red-ydgle.eu-g
Service Details
IBM Watson IoT
IBM Watson IoT
Node-RED D
https://node-red-ydgle.eu-gb.mybluemix.net/ui/#!/0?socketid=0CAs7pB7gNLNigG5AAAD
110%
Home
cylinder_weight
14.6
units
0
15
jar_weight
1880
units
0
2000



8:05

SMART KITCHEN

CYLINDER WEIGHT	13.85
CYLINDER STATUS	HIGH
JAR WEIGHT	1655
JAR STATUS	HIGH
GAS LEAKAGE	OFF
EXHAUST FAN	OFF