

INTERNSHIP

SMART KITCHEN USING IBM CLOUD

By

DIVYA CHOUDHARY 17BEC1188

NISHANTH D 17BEC1148

THANUJA V 17BEC1078

A project report submitted to



CONTENTS

S.NO	CONTENTS	Pg.NO
1	INTRODUCTION	3
	1.1 Overview	3
	1.2 Purpose	3
2	LITERATURE SURVEY	4
	2.1 Existing problem	4
	2.2 Proposed solution	4
3	THEORITICAL ANALYSIS	5
	3.1 Block diagram	5
	3.2 Hardware / Software designing	5
4	EXPERIMENTAL INVESTIGATIONS	13
5	FLOWCHART	26
6	RESULT	29
7	ADVANTAGES & DISADVANTAGES	30
8	APPLICATIONS	31
9	CONCLUSION	32
10	FUTURE SCOPE	33
11	BIBILOGRAPHY	34
	APPENDIX	35
	A. Source code	35

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW:

This paper aims to highlight the various aspects of IoT and its role in smart kitchen. The different technologies such as Node-red, WSN, Cloud Computing, Networking Technology and IBM cloud that support the IoT, and their applications in various fields i.e Smart home, Smart City, Smart Grid, Smart Health and Smart Farming have been covered. In addition to this a special coverage has been made with regard to Smart Kitchen. The description of various appliances and their application in the smart kitchen has been enumerated. In recent days kitchen based accident has been increased in both commercial kitchens and domestic kitchens. These accidents can be avoided using IOT technologies like monitoring the entire kitchen from remote areas. In order to implement this research both hardware and software will be utilized. From the hardware side ultrasonic (HC-SR04) MQ-6, load cell, fan, ESP8266 microcontroller WiFi module, NRF module has been used. From the software side integrated cloud application like IBM cloud, IoT platform, Node-RED, Cloudant db and mobile application has been used. All these sensors will be integrated with Arduino Nano ATMEGA328 processor board for cloud data transfer. In addition with log reports can be generated using the same methods.

Keywords :— IOT, Cloud Application, Mobile application, Integration, sensor, data transfer

1.2 PURPOSE:

LPG is flammable gas, it has the potential to cause fire accidents heavily. In order to avoid fire accident, many researchers are applying their effort to design new prototype model. IoT based gas leakage detection and monitor system uses different sensors and gas knob to avoid the fire accident in the kitchen. It will acquire input from the sensor and control the gas knob automatically either from High ignition to Medium or from Medium to Low. If gas leakage is detected then it will be intimated to the end user by using push button option in the cloud environment.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING WORK:

There are different LPG detecting technology is used in existing method. The Liquefied Petroleum Gas is detected by semiconductor devices. The Percentage of accidents occurs due to Leakage of LPG is increasing rapidly every year. The LPG cylinder is manual control object which using valve regulator. In the existing system there are two methods one is when leakage detected the alarm will sound and the owner have to modify manually. Second method is when fire accident occurred heavily due to leakage of gas the system will send message to the fire stations using GSM module. In first method the action is made manually and the next method action made automatically. In further developed gas detecting methods the system made so simple whenever the gas leakage is detecting the cylinder valve will be closed. in this system the Internet of things used effectively. The Alert message sent to the user by using mobile application via Internet of Things.

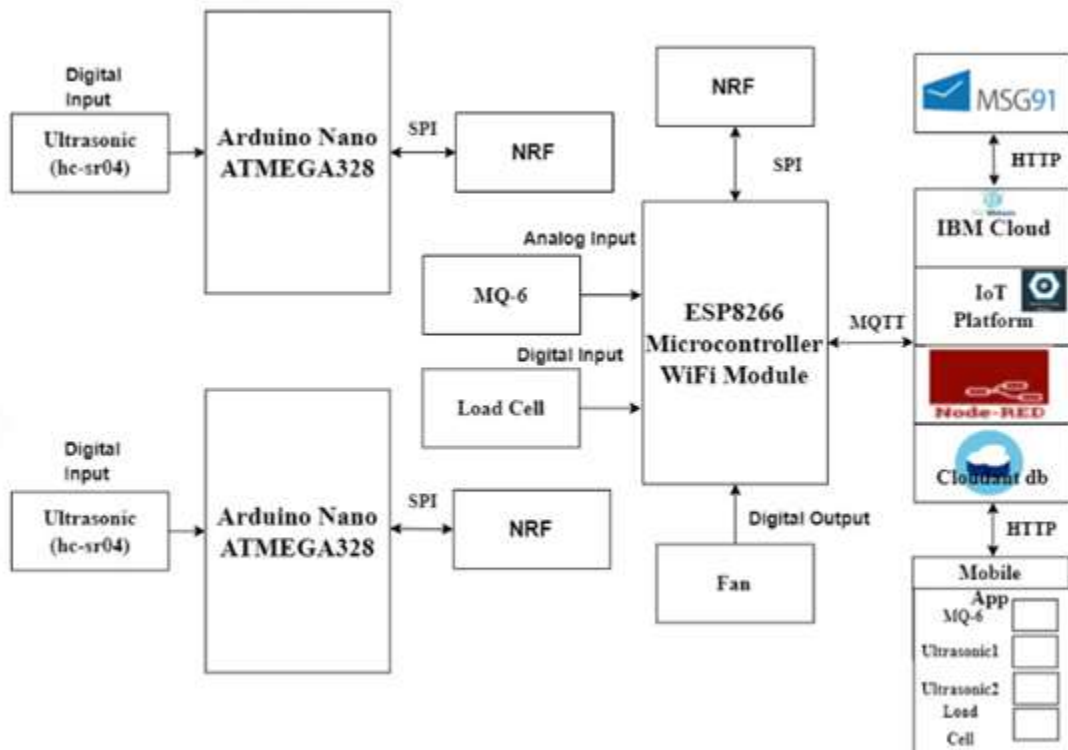
2.2 PROPOSED SOLUTION :

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. This is useful for many of the real-world applications and services, and one would for example apply it to build a smart residence; windows can be closed automatically when the air conditioner is turned on, or can be opened for oxygen when the gas oven is turned on. The idea of IoT is especially valuable for persons with disabilities, as IoT technologies can support human activities at larger scale like building or society, as the devices can mutually cooperate to act as a total system. The next one is 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. In this system we present a wireless detector system. Based on a small scale and low cost device for achieving processing, storing, sensing and communicating. All sensors are connected with micro-controller in specified manner. If Gas Leakage, Sound, Moisture is sensed means the Stove valve has to stop by using Motor action. All sensor data's are continuously updated in the Cloud by using Wi-Fi module in controller for user. The alert signal is sent to user through the Mobile by using IoT application.

CHAPTER 3

THEORATICAL ANALYSIS

3.1 BLOCK DIAGRAM:



3.2 HARDWARE DESIGN AND SOFTWARE DESIGN:

HARDWARE DESIGN:

ULTRASONIC (hc-sr04):

Ultrasonic (US) sensor is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that

$$\text{Distance} = \text{Speed} \times \text{Time}$$

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module



Fig 3.1 ULTRASONIC (hc-sr04):

ARDUINO NANO ATMEGA 328:

Arduino Nano is a small, compatible, flexible and breadboard friendly Microcontroller board, developed by Arduino.cc in Italy, based on ATmega328p (Arduino Nano V3.x) / Atmega168 (Arduino Nano V3.x).It comes with exactly the same functionality as in Arduino UNO but quite in small size.Functions like pinMode() and digitalWrite() are used to control the operations of digital pins while analogRead() is used to control analog pins.The analog pins come with a total resolution of 10bits which measure the value from zero to 5V.

Arduino Nano comes with a crystal oscillator of frequency 16 MHz. It is used to produce a clock of precise frequency using constant voltage.There is one limitation using Arduino Nano i.e. it doesn't come with DC power jack, means you can not supply external power source through a battery.This board doesn't use standard USB for connection with a computer, instead, it comes with Mini USB support.



Fig 3.2 ATmega328p

It is programmed using Arduino IDE which is an Integrated Development Environment that runs both offline and online.

Microcontroller	Atmel ATmega328
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (of which 2KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz
Dimensions	0.70" x 1.70"

Fig 3.3 features of Atmega328

MQ-6:

Using a MQ sensor to detect a gas is very easy. You can either use the digital pin or the analog pin to accomplish this. Simply power the module with 5V and you should notice the power LED on the module to glow and when no gas is detected the output LED will remain turned off meaning the digital output pin will be 0V. These sensors have to be kept on for pre-heating time (mentioned in features above) before you can actually work with it. Now, introduce the sensor to the gas you want to detect and you should see the output LED to go high along with the digital pin, if not use the potentiometer until the output gets high. Now every time your sensor gets introduced to this gas at this particular concentration the digital pin will go high (5V) else will remain low (0V).

You can also use the analog pin to achieve the same thing. Read the analog values (0-5V) using a microcontroller, this value will be directly proportional to the concentration of the gas to which the sensor detects. You can experiment with these values and check how the sensor reacts to different concentrations of gas and develop your program accordingly.



fig 3.4 Mq-6 gas sensor

ESP8266 MICROCONTROLLER WIFI MODULE:

The ESP8266 is a very user friendly and low cost device to provide internet connectivity to your projects. The module can work both as a Access point (can create hotspot) and as a station (can connect to Wi-Fi), hence it can easily fetch data and upload it to the internet making Internet of Things as easy as possible. It can also fetch data from internet using API's hence your project could access any information that is available in the internet, thus making it smarter. Another exciting feature of this module is that it can be programmed using the Arduino IDE which makes it a lot more user friendly. However this version of the module has only 2 GPIO pins (you can hack it to use upto 4) so you have to use it along with another microcontroller like [Arduino](#), else you can look onto the more standalone ESP-12 or ESP-32 versions. So if you are looking for a module to get started with IOT or to provide internet connectivity to your project then this module is the right choice for you.

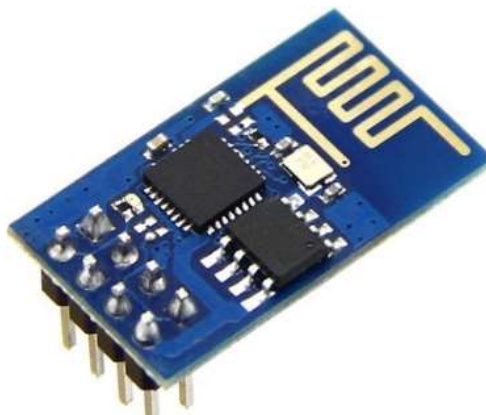


fig 3.5 ESP8266

SOFTWARE DESIGN:

IBM CLOUD:

Cloud Data Management Normally whenever the code is executing it will the data in terminal panel only. For collecting all data we have store some and it can able to whenever we want. Wireless storage is more efficient. We can access wherever in the world with internet it can be viewed by owner the server only.

The hub of the IBM IoT approach: set up and manage your connected devices so your apps can access live and historical data. Use the secure APIs of the IBM Cloud to connect your apps with data from your devices. Create applications within IBM Cloud, another cloud or your own servers to interpret data. A secure IoT platform with the power of Watson's cognitive engine. The Watson IoT Platform is a fully managed.

IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices. The IBM Watson IoT Platform is a hub for connecting devices, gateways, and applications for IoT solutions. It supports REST and MQTT protocols for applications, devices, gateways, event processing, and administrative tasks. The IBM Watson IoT Platform is available on the IBM Cloud platform (formerly IBM Bluemix), a cloud platform based on Cloud Foundry and Kubernetes.

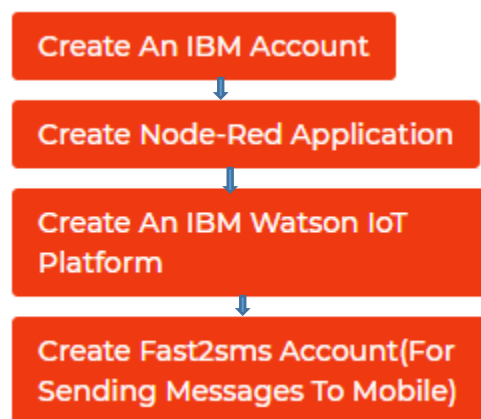


fig 3.6 flow chart

- **Dashboard:**

This is the first thing that you will see when you access the IBM Watson IoT Platform. This dashboard can be a combination a number of boards and cards, offering several visualization options for your IoT solution.

- **Devices, gateways, and applications:**

Another feature available in the platform is device management control. This feature makes it possible to create and remove devices, gateways, applications,

and device types. It also makes it possible to check and trigger actions to the device, such as a firmware upgrade request or reset. we can also create API keys so that your applications can connect to the IoT organization and interact with the other components of the solution.

- **Cloudant db:**

IBM Cloudant is a fully managed JSON document database that offers independent serverless scaling of provisioned throughput capacity and storage. Cloudant is compatible with Apache CouchDB and accessible through a simple to use HTTPS API for web, mobile, and IoT applications. Cloudant is a non-relational, distributed database service of the same name. Cloudant is based on the Apache-backed CouchDB project and the open source BigCouch project.

FAST2SMS:

Fast2SMS is a one of a kind Bulk SMS Service Provider in India having plethora of features designed to enhance your marketing experience and build an instant connectivity with your target audience. Taking care of the needs of our users, recently we have introduced new features so that you can benefit immensely from our service and make the best use of it.

Features in Fast2SMS:

1. AI Algorithm – “Failed SMS” Retry via different operator
2. SMS preview before sending
3. Save Message for future use
4. Tutorial Videos in Hindi and English
5. Dedicated chat support team and phone number

CONFIGURATION OF WEB APP:

CREATE A NODE-RED FLOW TO GET DATA FROM THE DEVICE:

Node-RED:

NODE-red is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

BROWSER-BASED FLOW EDITING:

Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette. Flows can be then deployed to the runtime in a single-click. JavaScript functions can be created within the editor using a rich text editor. A built-in library allows you to save useful functions, templates or flows for re-use.

Name	Description
Node-RED	A visual tool for wiring the Internet of Things
Node-RED Dashboard	A dashboard UI for Node-RED
Node generator	Command line tool to generate Node-RED node modules from several various sources, including Open API document and function node's source.
Node-RED Command Line Tool	Command-line tool allows you to remotely administer a Node-RED instance.

fig 3.7 node-red dashboard

MIT APP:

It uses a graphical user interface (GUI) very similar to the programming languages Scratch (programming language) and the StarLogo, which allows users to drag and drop visual objects to create an application that can run on android devices, while a App-Inventor Companion (The program that allows the app to run and debug on) that works on iOS running devices are still under development. In creating App Inventor, Google drew upon significant prior research in educational computing, and work done within Google on online development environment.

App Inventor and the other projects are based on and informed by constructionist learning theories, which emphasize that programming can be a vehicle for engaging powerful ideas through active learning.

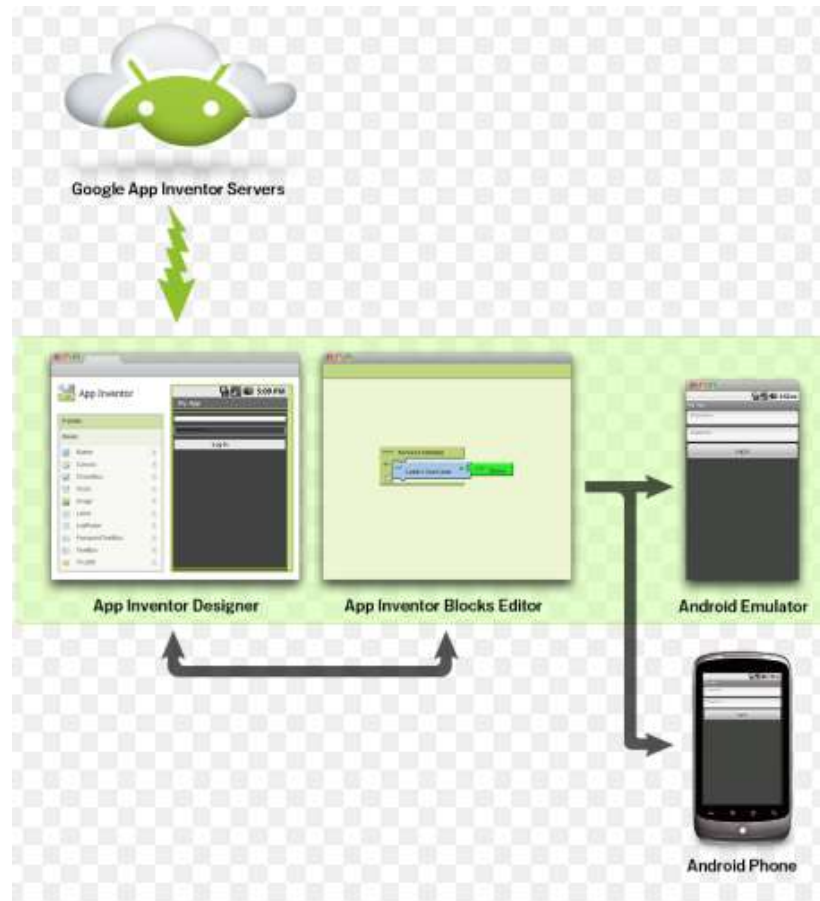


Fig 3.8 MIT app inverter

CHAPTER 4

EXPERIMENTAL INVESTIGATIONS

IMAGES:

OUTPUT IN PYTHON:

1) When cylinder and jar are full and their status are high.

```
Published Cylinder_weight = 14.9 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1486 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.8 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1472 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.7 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1458 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.6 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1444 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.5 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1430 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.4 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1416 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.3 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1402 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.2 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1388 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.1 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1374 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 14.0 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1360 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 13.9 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1346 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 13.8 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1332 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 13.7 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1318 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 13.6 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1304 Jar_Status=HIGH FAN_STATUS =OFF
```

2) Jar status drops to medium.

```
Published Cylinder_weight = 11.8 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1052 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 11.7 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1038 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 11.6 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1024 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 11.5 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 1010 Jar_Status=HIGH FAN_STATUS =OFF
Published Cylinder_weight = 11.4 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 996 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 11.3 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 982 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 11.2 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 968 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 11.1 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 954 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 11.0 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 940 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.9 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 926 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.8 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 912 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.7 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 898 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.6 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 884 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.5 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 870 Jar_Status=MEDIUM FAN_STATUS =OFF
```

3) When gas leakage is detected and fan is on and cylinder goes to medium from low.

```
Published Cylinder_weight = 10.6 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 884 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.5 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 870 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.4 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 856 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.3 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 842 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.2 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 828 Jar_Status=MEDIUM FAN_STATUS =OFF
Published Cylinder_weight = 10.1 Cylinder_Status=HIGH Gas leakage= OFF jar_weight = 814 Jar_Status=MEDIUM FAN_STATUS =OFF
Gas leakage detected! Check immediately!
Published Cylinder_weight = 10.0 Cylinder_Status=HIGH Gas leakage= ON jar_weight = 800 Jar_Status=MEDIUM FAN_STATUS =ON
Published Cylinder_weight = 9.9 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 786 Jar_Status=MEDIUM FAN_STATUS =ON
```

```
2020-06-28 11:41:08,896 ibmiotf.device.Client ERROR Unexpected disconnect from the IBM Watson IoT Platform: 1
```

```
Published Cylinder_weight = 9.8 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 772 Jar_Status=MEDIUM FAN_STATUS =ON
Published Cylinder_weight = 9.7 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 758 Jar_Status=MEDIUM FAN_STATUS =ON
Published Cylinder_weight = 9.6 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 744 Jar_Status=MEDIUM FAN_STATUS =ON
Published Cylinder_weight = 9.5 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 730 Jar_Status=MEDIUM FAN_STATUS =ON
Published Cylinder_weight = 9.4 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 716 Jar_Status=MEDIUM FAN_STATUS =ON
```

4) When Cylinder and jar both volumes drop to low from medium

```
Published Cylinder_weight = 5.3 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 142 Jar_Status=LOW FAN_STATUS =ON
Published Cylinder_weight = 5.2 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 128 Jar_Status=LOW FAN_STATUS =ON
Published Cylinder_weight = 5.1 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 114 Jar_Status=LOW FAN_STATUS =ON
```

```
2020-06-28 11:41:13,888 ibmiotf.device.Client INFO Connected successfully: d:maql17:RaspberryPi:123456
```

```
2020-06-28 11:41:13,888 ibmiotf.device.Client ERROR Unexpected disconnect from the IBM Watson IoT Platform: 1
```

```
Published Cylinder_weight = 5.0 Cylinder_Status=MEDIUM Gas leakage= ON jar_weight = 100 Jar_Status=LOW FAN_STATUS =ON
Published Cylinder_weight = 4.9 Cylinder_Status=LOW Gas leakage= ON jar_weight = 86 Jar_Status=LOW FAN_STATUS =ON
Published Cylinder_weight = 4.8 Cylinder_Status=LOW Gas leakage= ON jar_weight = 72 Jar_Status=LOW FAN_STATUS =ON
Published Cylinder_weight = 4.7 Cylinder_Status=LOW Gas leakage= ON jar_weight = 58 Jar_Status=LOW FAN_STATUS =ON
Published Cylinder_weight = 4.6 Cylinder_Status=LOW Gas leakage= ON jar_weight = 44 Jar_Status=LOW FAN_STATUS =ON
```

5) Jar gets emptied.

```
Published Cylinder_weight = 4.4 Cylinder_Status=LOW Gas leakage= ON jar_weight = 16 Jar_Status=LOW FAN_STATUS =ON
Published Cylinder_weight = 4.3 Cylinder_Status=LOW Gas leakage= ON jar_weight = 2 Jar_Status=LOW FAN_STATUS =ON
The JAR is empty!
Published Cylinder_weight = 4.2 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
```


6) Cylinder gets emptied too.



```
Published Cylinder_weight = 0.6 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
Published Cylinder_weight = 0.5 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
Published Cylinder_weight = 0.4 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
Published Cylinder_weight = 0.3 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
Published Cylinder_weight = 0.2 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
Published Cylinder_weight = 0.1 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
Published Cylinder_weight = 0.0 Cylinder_Status=LOW Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
```

```
2020-06-28 11:57:43,602 ibmiotf.device.Client INFO Connected successfully: d:maql17:RaspberryPi:123456
```

The cylinder is empty

```
Published Cylinder_weight = 0 Cylinder_Status=EMPTY Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
Published Cylinder_weight = 0 Cylinder_Status=EMPTY Gas leakage= ON jar_weight = 0 Jar_Status=EMPTY FAN_STATUS =ON
```

2)Values are being updated to IBM platform.

	123456	 Connected	RaspberryPi	Device	Jun 25, 2020
---	--------	---	-------------	--------	--------------

Identity

Device Information

Recent Events

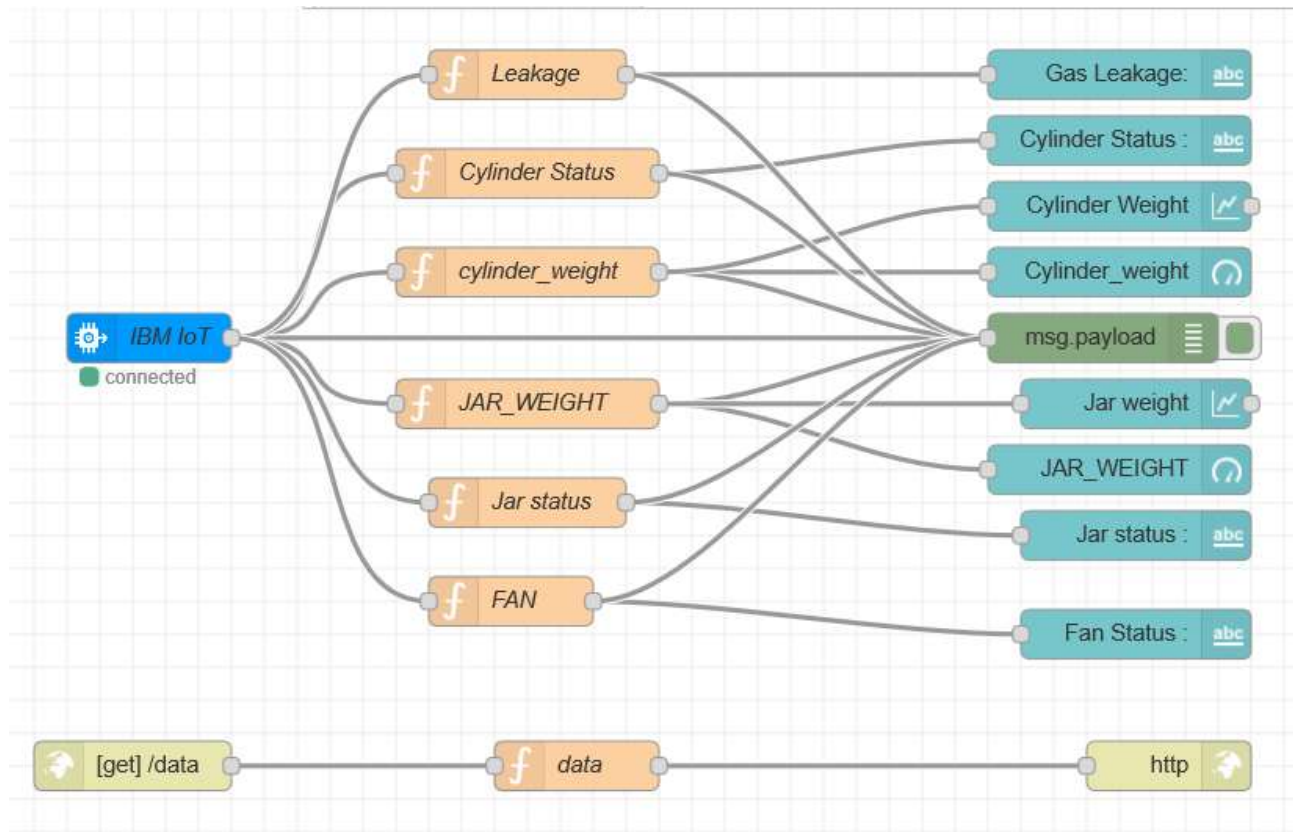
State

Logs

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":0,"status":"EMPTY","leak":"O...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":0,"status":"EMPTY","leak":"O...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":0,"status":"EMPTY","leak":"O...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":0,"status":"EMPTY","leak":"O...	json	a few seconds ago
Smart Kitchen	{"cylinder_weight":0,"status":"EMPTY","leak":"O...	json	a few seconds ago

1) Node RED SETUP:



2) WEB APP DASHBOARD

4.1) When the Jar and cylinder are full



4.2) Jar goes to medium



4.3) Cylinder too goes to medium



4.4)When both are low



4.5)When everything is empty



3) Node RED /data url from which we send data to mobile application



```
{"Cylinder Weight":14.9,"Cylinder status":"HIGH","Leakage":"OFF","Jar weight":1486,"Jar status":"HIGH","Fan status":"OFF"}
```

4) Mobile App UI values:

6.1) When cylinder and jar contents are high and there is no gas leakage.



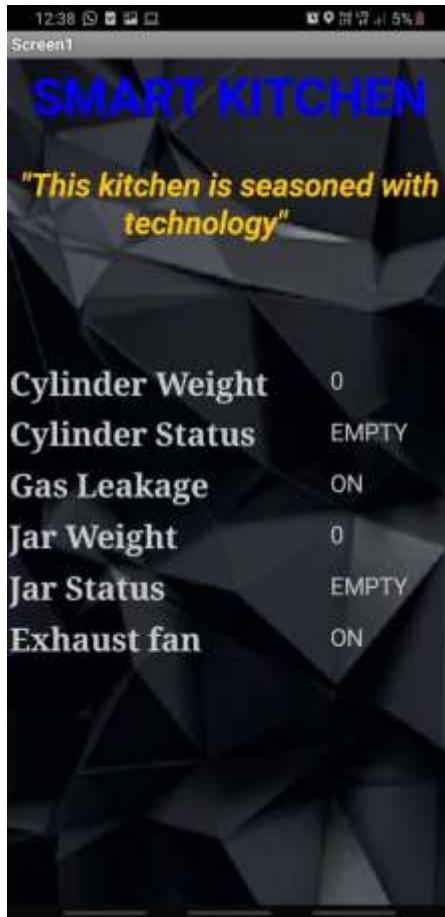
6.2) When the volume of cylinder and jar drops to medium and there is gas leakage and the exhaust fan is turned on.



6.3) When the volume of cylinder and jar are low.

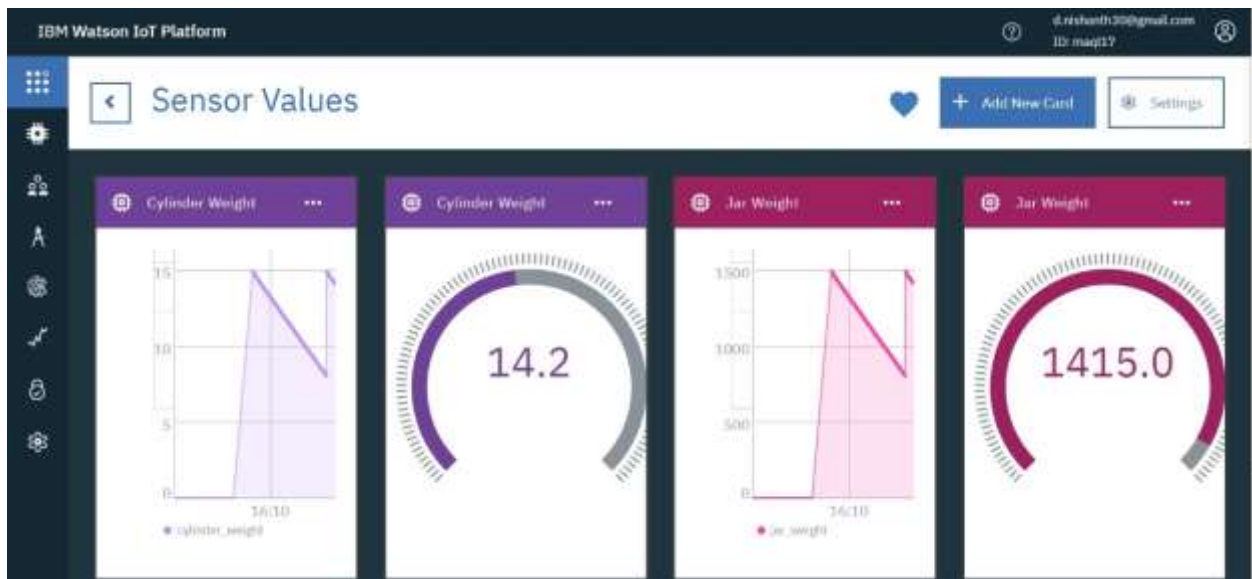


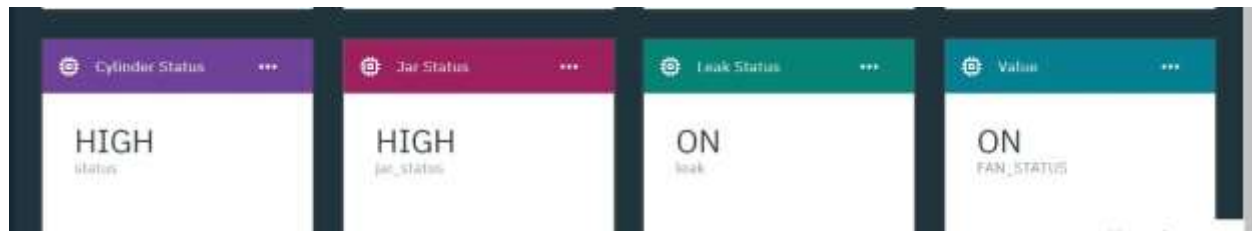
6.4) When the containers are empty.



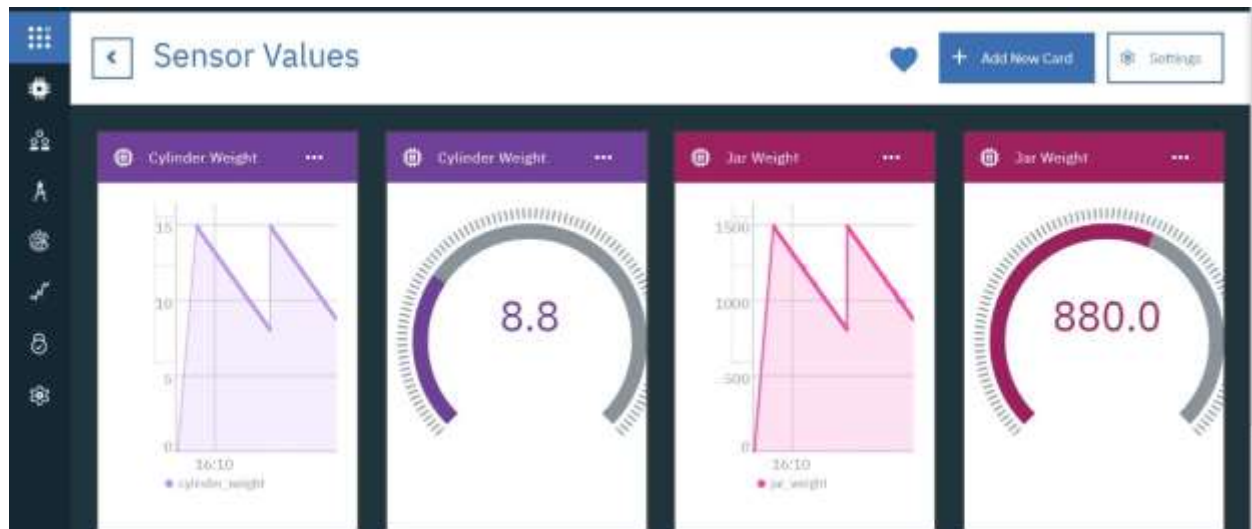
7) IBM IOT PLATFORM BOARD OUTPUTS: USER INTERFACE

7.1) When both the jar and cylinder status is high:

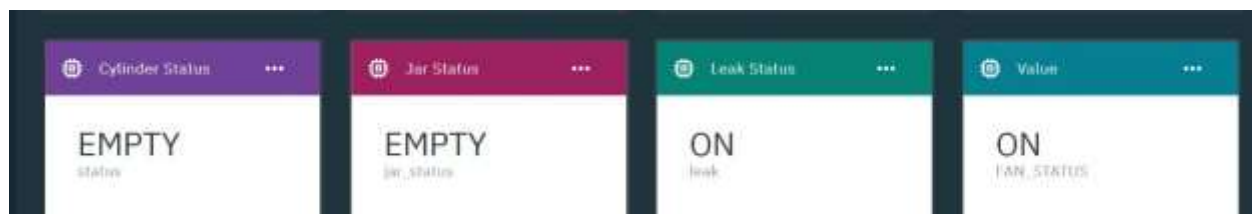
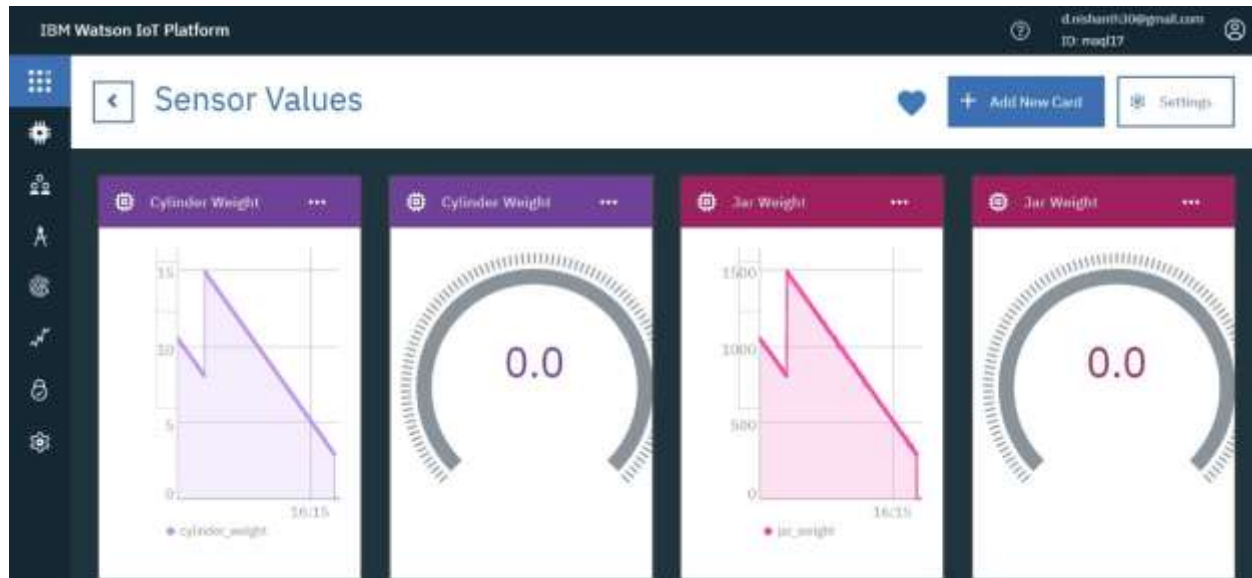




7.2) When the jar status and cylinder status is medium:



7.3) When the jar and cylinder status is empty:

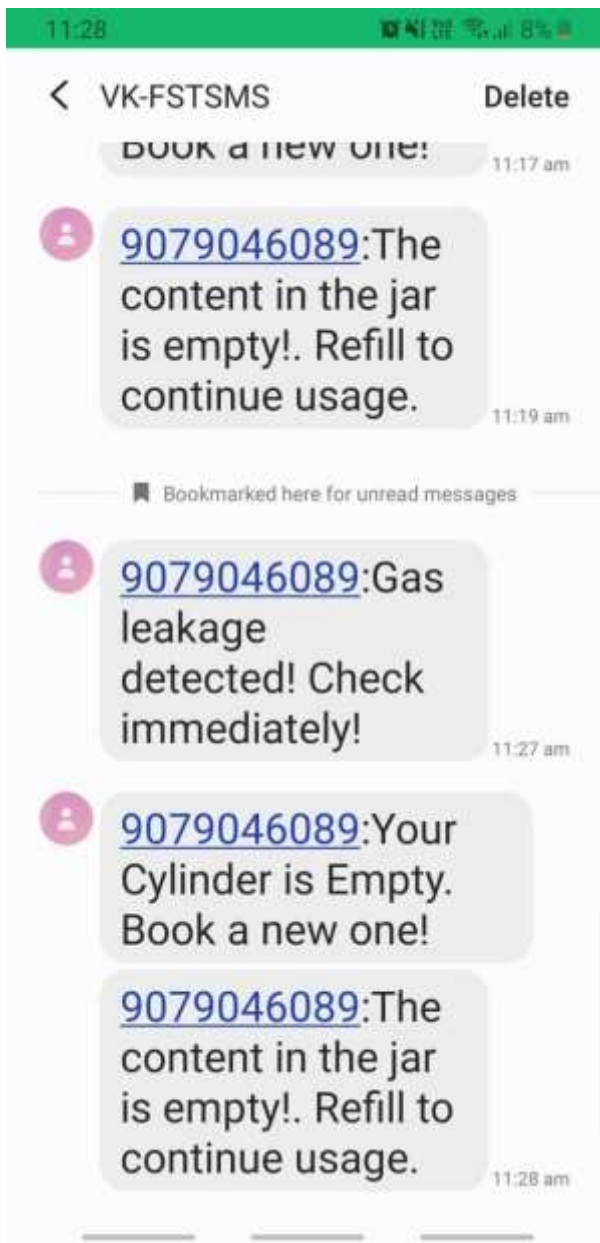


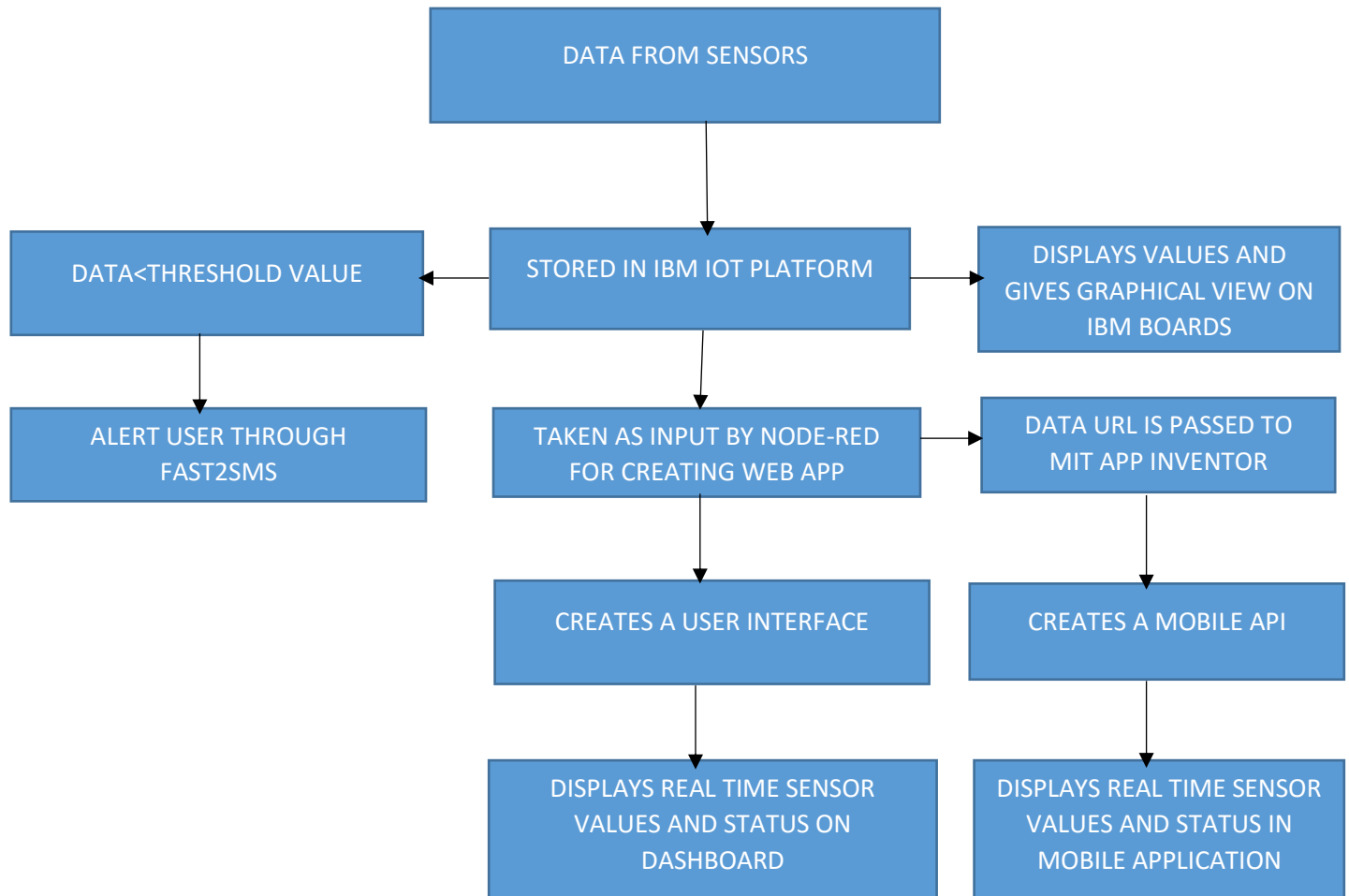
8) Fast2Sms Output to alert user:

8.1) Output of message alert when jar gets empty.

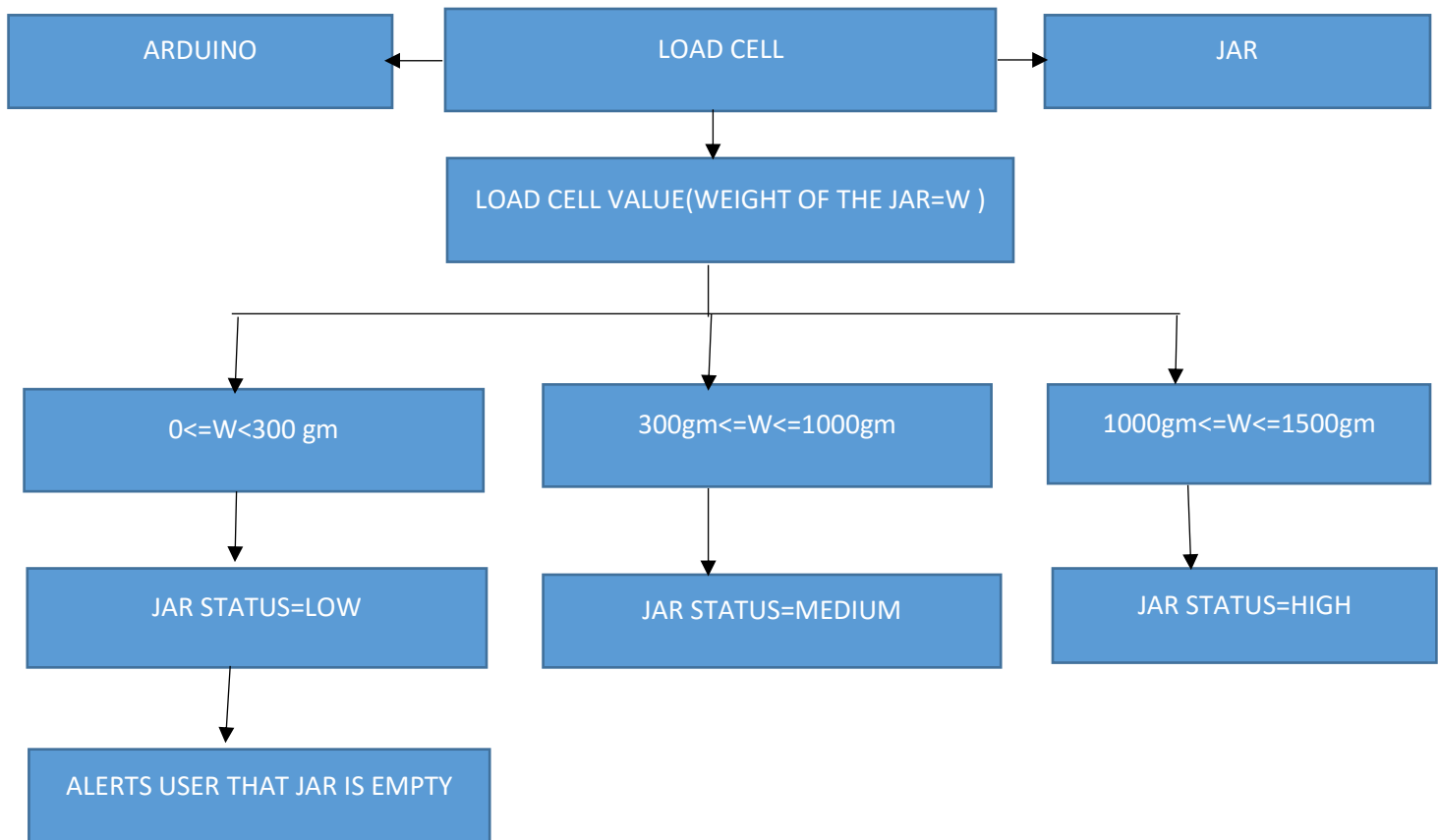
8.2) Output of message alert when there is gas leakage.

8.3) Output of message alert when cylinder gets empty.

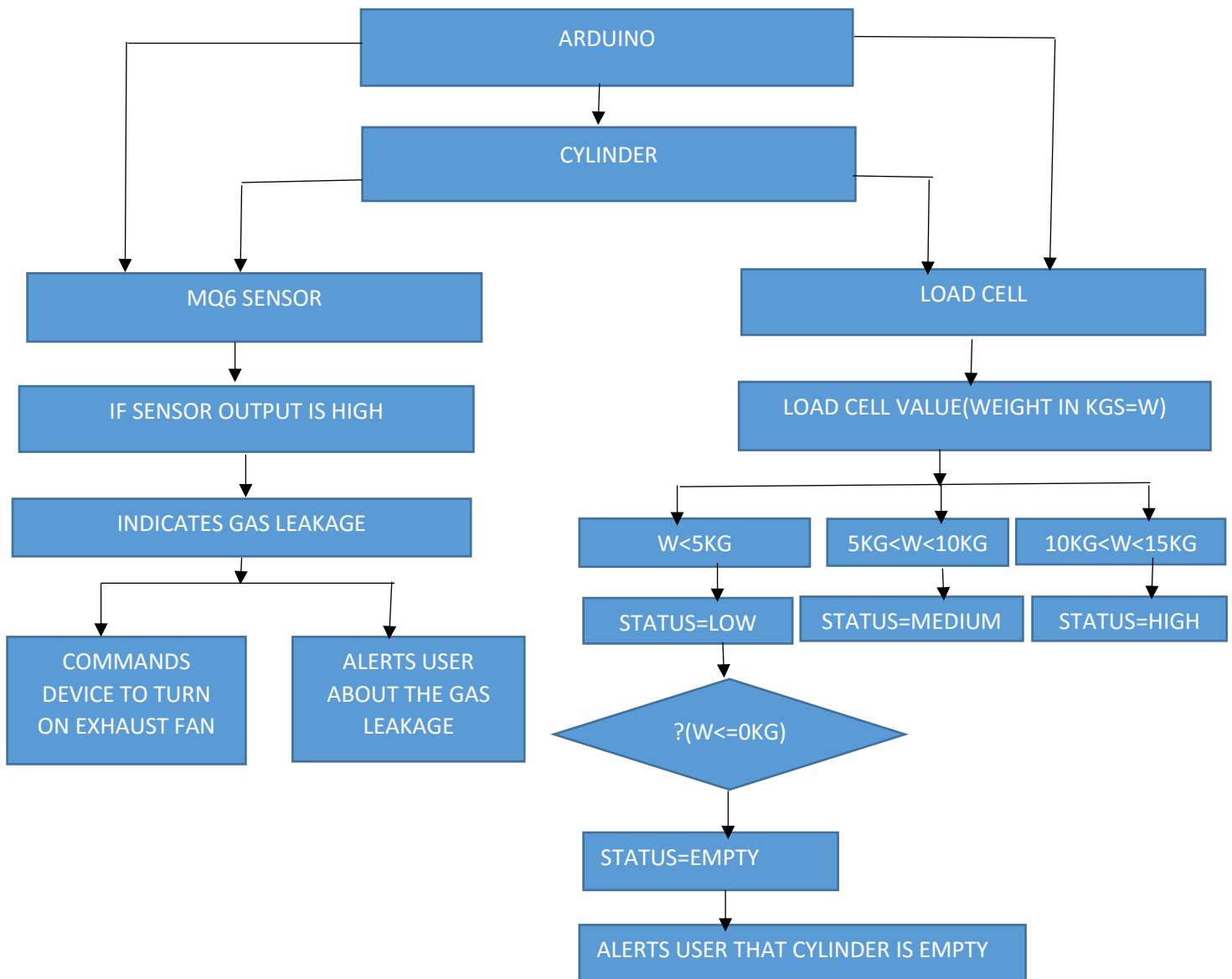


CHAPTER 5**OVERALL FLOWCHART**

5) WORK FLOW: JAR



6) WORK FLOW: CYLINDER:



CHAPTER 6

RESULT

Smart Kitchen integrates IOT with IBM and sends alert notification to user through Fast2sms whenever:

- There is a gas leakage
- Cylinder gets empty
- Jar gets empty

It also sends command to turn on the exhaust fan in case of any leakage.

CHAPTER 7

ADVANTAGES & DISADVANTAGES

Advantages:

Our system will detect the leakage of the gas, in case there is any leakage it will send a SMS to the owner and it will turn off power and activate an alarm. The system will continuously monitor the weight of the gas. There will be automatic booking of the gas done (by setting a threshold value for the weight sensor). We will even measure the weight and the Temperature around the gas cylinder.

Disadvantages:

1. Smart appliances cost more. In addition to higher purchase prices, they often require more repairs than mechanical versions of the same machines. those repairs can be 50 to 100 percent more expensive.
2. They may pose data and privacy risks. Smart kitchen appliances may not utilize reliable internet security protocols, giving hackers a pathway to access other connected devices in your home. Also, the more data that these devices are collecting about you, your habits, and your home, the more that data could be vulnerable.
3. Firmware issues. Manufacturers may not provide timely firmware updates, which means an appliance may no longer integrate with other devices, like a smart kitchen hub and voice-activated controllers. It's also easier for hackers to access devices that aren't kept up-to-date and secure.
4. No connection = dumb appliances. If your smart appliances can't connect to the internet, they are no longer "smart." Before buying, be sure to check reviews for individual products as well as the manufacturer's customer service ratings

CHAPTER 8

APPLICATIONS

Applications:

1. Monitoring the all sensors and its value for safty detection of gas leakage, temperature of room,and daily usage of system to the user.
2. Exhaust fan switched on in case of abnormal reading
3. Stores the data related to the system like daily data monitoring Intelligent System for Domestic Gas Appliances using IOT. In our day-to-day life there is serious threat about leakage which leads to suffocation when inhaled, when ignited leads to explosion and causes a number of deaths. This project is about designing a LPG leakage monitoring system which is proposed for home safety. This system detects the leakage of the LPG and alerts the consumer about the leak by SMS and as an emergency measure the system will turnoff the power supply, while activating the alarm.

CHAPTER 9

CONCLUSION

Thus the concept of IoT based Smart Kitchen and Avoiding Fire Accidents due Leakage of LPG Gas is applied and Verified experimentally. The Output data of this system is continuously transferred to the User in IoT cloud data Transfer Process. 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, temperature 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. Temperature sensors are used to detect the current room temperature. 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, also stores the room temperature and information about the users. Threshold values are set into the room, when it crosses that values it will send a notification to the user, about the leakage of a gas cylinder and leakage of a gas. Server can communicate with the user through android device. Through email and SMS server can sends a notification to the user which will display on the android devices. It can prevent the accident and hazards. The only way to access the information is if the user is far from the home. It is a cost effective and time-consuming solution.

CHAPTER 10
FUTURE SCOPE:

One of the modifications is to provide the system with a dual power supply i.e. include a battery power supply source in addition to the utility power supply. Design the sensors that can be used for more kitchen parameters. Apply various techniques to make the system more secure. Also we can increase some sensors.

CHAPTER 11

BIBILOGRAPHY

- [1] Mohd Zaki Ghazali, Noorhayati Mohamed Noor and Sulastris Putit "Development of Microcontroller Based Mobile Gas Monitoring Sensing Robot" in International symposium on Robotics and Intelligent Sensors 2012 (IRIS 2012)- vol.3 no.9 pp. 1190- 1196,2012
- [2] Kumar Keshamoni and Sabbani Hemanth "Smart Gas Level Monitoring, Booking & Gas Leakage Detector over IoT" in 2017 IEEE 7th International Advance Computing Conference, vol.7 no.9 pp. 330-332, 2017.
- [3] Asmita Varma, Prabhakar S and Kayalvizhi Jayavel "Gas Leakage Detection and Smart Alerting and Prediction Using IoT" in 2017 Second International Conference on Computing and Communications Technologies, - vol.6 no.3 pp.327-333, 2017.
- [4] Sushil Kumar Paridda, Ankit Pratik, Sharad Kumar Pani and Rati Ranjan Sabat " Innovative Design and Simulation of gas Level Detection System in Liquefied Petroleum Gas Cylinder For Indian Household Application" in International Journal of Industrial Electronics and Electrical Engineering, vol.9, pp. 87-89, 2014.
- [5] Selvapriya, Sathya Prabha, Abdulrahim and Aarthi "LPG Leakage Monitoring and Multilevel Alerting System" in International Journal of Engineering Sciences & Research Technology, vol.6 pp. 3287-3290, June 2013.
- [6] Apeh S. T, Eramah K. B and Iruansi. U "Design and Development of Kitchen Gas Leakage Detection and Automatic Gas Shut off System" in Journal of Emerging Trends and Applied Sciences (JETEAS), vol.6 pp-222.228 September 2014.
- [7] Kenneth V. Balmes, James Matthew T. Chua, Mary Anne O. De Jesus, and Karen Cristine O. Tan, Argel A. Bandala "Utilization of Sensor Network for Combustible Gas Detection and Monitoring Implemented in House Hold" in 2015 IEEE pp. 978-980, 2015.

APPENDIX:**1)PYTHON SOURCE CODE:**

CODE:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import requests

url = "https://www.fast2sms.com/dev/bulk"

#My IoT platform device's credentials to connect the device and the platform
organization="maql17"
deviceType= "RaspberryPi"
deviceId= "123456"
authMethod= "token"
authToken="12345678"

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

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 connection device: %s" % str(e))
    sys.exit()

deviceCli.connect()
```

Since we are not using actual hardware to demonstrate. We assume a scenario which contains all the possible outputs.

We assume weight of the jar to be 1500 grams.

We know the weight of a full cylinder is 30 Kgs. But we must calibrate our device such that it only takes the weight of

#the gas inside the cylinder and let us know if its empty. So we take the weight of the cylinder to be 15Kgs(gas weight).

```
jar_weight=1500
```

```
cylinder_weight=15
```

```
Fan= "OFF"
```

```
leak="OFF"
```

```
i=0
```

Variables cyl_empty and jar_empty are used as flags. This is to make sure that when we send the alert sms, the alert is sent

only once until reset. So this prevents resending messages everytime the sensor detects empty in the loop.

```
cyl_empty=0;
```

```
jar_empty=0
```

```
while True:
```

#We assume the cylinder to be used 0.01 times in every loop. So we can demonstrate the stages of the cylinder and its status

```
cylinder_weight=cylinder_weight-0.1;
```

One tablespoon is 14 grams. Hence we reduce 14 grams everytime to simulate taking sugar out of a jar with a tablespoon

```
jar_weight=jar_weight-14;
```

Depending on the weight of the cylinder in Kgs. The current status of gas is sent ot the user. The condition is as below.

```

if(cylinder_weight <0 and cylinder_weight>=5):
    status="LOW"
elif(cylinder_weight <5 and cylinder_weight>=10):
    status="MEDIUM"
elif(cylinder_weight >10 and cylinder_weight<=15):
    status="HIGH"
else:
    cylinder_weight=0
    status="EMPTY"

```

#If the cylinder goes empty, We alert the user by sending an SMS that the cylinder has gone empty using FAST2SMS.

```

if(cyl_empty==0):
    payload = "sender_id=FSTSMS&message=Your Cylinder is Empty. Book a new
one!&language=english&route=p&numbers=9003191279"

    headers = {'authorization':
"64a9QD2NfZVlpxqGwcOtKrBMnFWAzdkmXCjTYoR5yeS107Llb39NAEYamlfeipo4Wvht8qn0Q
LO2cTzR",
               'Content-Type': "application/x-www-form-urlencoded", 'Cache-Control': "no-
cache", }

    response = requests.request("POST", url, data=payload, headers=headers)
    print(response.text)
    print("The cylinder is empty")
    # cyl_empty flag is changed to 1 to prevent repeating the sms until reset.
    cyl_empty=1;

```

Depending on the weight of the Jar in grams. The current status of gas is sent ot the user. The condition is as below.

```

if(jar_weight <300 and jar_weight>=0):
    jar_status="LOW"

```

```

elif(jar_weight >=300 and jar_weight<1000):
    jar_status="MEDIUM"
elif(jar_weight >=1000 and jar_weight<=1500):
    jar_status="HIGH"
else:
    jar_weight=0
    jar_status="EMPTY"
    if(jar_empty==0):
        print("The JAR is empty!")

        payload = "sender_id=FSTSMS&message=The content in the jar is empty!. Refill to
continue usage.&language=english&route=p&numbers=9003191279"

        headers = {'authorization':
"64a9QD2NfZVlpxqGwcOtKrBMnFWAzdkmXCjTYoR5yeS107Llb39NAEYamlfeipo4Wvht8qn0Q
LO2cTzR",

                    'Content-Type': "application/x-www-form-urlencoded",'Cache-Control': "no-
cache", }

        response = requests.request("POST", url, data=payload, headers=headers)
        print(response.text)

        # jar_empty flag is changed to 1 to prevent repeating the sms until reset.
        jar_empty=1;

#Since we are not using hardware to detect the gas leak, we simulate to show this condition
is also taken care.

#We assume that after 50 loops, the gas starts leaking and we alert the user by sending an
sms.
i=i+1
if(i==50):
    print("Gas leakage detected! Check immediately!")

    payload = "sender_id=FSTSMS&message=Gas leakage detected! Check
immediately!&language=english&route=p&numbers=9003191279"

    headers = {'authorization':
"64a9QD2NfZVlpxqGwcOtKrBMnFWAzdkmXCjTYoR5yeS107Llb39NAEYamlfeipo4Wvht8qn0Q
LO2cTzR",

```

```

        'Content-Type': "application/x-www-form-urlencoded",'Cache-Control': "no-cache",
    }

    response = requests.request("POST", url, data=payload, headers=headers)
    print(response.text)

    #If there is gas leak, we turn on the exhaust fan.
    Fan="ON"
    leak="ON"

    #We store all our data collected in JSON format in as "data".

    data= {'cylinder_weight' : round(cylinder_weight,2), 'status': status,'leak':leak, 'jar_weight':
jar_weight, 'jar_status':jar_status,'FAN_STATUS':Fan}

    #function to show if the data is published in our iot platform.
    def myOnPublishCallback():
        print(" Published Cylinder_weight = %s " %round(cylinder_weight,2) , "Cylinder_Status=%s
" %status,"Gas leakage= %s"%leak,"jar_weight = %s " %jar_weight , "Jar_Status=%s "
%jar_status,"FAN_STATUS =%s"%Fan,"\n")

    #publishEvent function published the json data to the IBM platform
    success = deviceCli.publishEvent("Smart Kitchen","json", data ,qos=0,
on_publish=myOnPublishCallback)

    #if we couldnt connect to the IBM platform, not connected to IoT is printed.
    if not success:
        print("Not connected to IoT")
        time.sleep(0.1)

    deviceCli.commandCallback= myCommandCallback
    deviceCli.disconnect()

```

2)NODE-RED FLOW CODE:

```
[{"id":"600afed6.9bbbd","type":"tab","label":"Flow
1","disabled":false,"info":"","id":"3756272f.d133f8","type":"debug","z":"600afed6.9bbbd","name"
":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType
":"msg","x":710,"y":200,"wires":[]},{id":"f2f837cd.869568","type":"ibmiot
in","z":"600afed6.9bbbd","authentication":"apiKey","apiKey":"857e7ad0.9e65f8","inputType":"evt
","logicalInterface":"","ruleId":"","deviceId":"123456","applicationId":"","deviceType":"RaspberryPi
","eventType":"+","commandType":"","format":"json","name":"IBM
IoT","service":"registered","allDevices":"","allApplications":"","allDeviceTypes":"","allLogicalInterf
aces":"","allEvents":true,"allCommands":"","allFormats":"","qos":0,"x":130,"y":200,"wires":[["18a9
b9b9.440e46","a51801d9.96dda","3756272f.d133f8","dfb6b71c.e13998","76b5423c.33d19c","8
d3a73c6.77782","d25b01bf.25f76"]],{"id":"8cef6b9.fe32098","type":"ui_gauge","z":"600afed6.9b
bbd","name":"","group":"45e19a37.9354e4","order":1,"width":0,"height":0,"gtype":"gauge","title":"C
ylinder_weight","label":"units","format":"{{value}}","min":0,"max":15,"colors":["#b31800","#e6e6
00","#57ca3a"],"seg1":"","seg2":"","x":720,"y":160,"wires":[]},{id":"789690c0.15675","type":"ui_g
auge","z":"600afed6.9bbbd","name":"","group":"805fca1b.089668","order":1,"width":0,"height":0,"
gtype":"gauge","title":"JAR_WEIGHT","label":"units","format":"{{value}}","min":0,"max":1500,"col
ors":["#b30f00","#e6e600","#63ca3a"],"seg1":"","seg2":"","x":720,"y":280,"wires":[]},{id":"18a9b9
b9.440e46","type":"function","z":"600afed6.9bbbd","name":"cylinder_weight","func":"global.set('
Cylinder_weight',msg.payload.cylinder_weight)\nmsg.payload=msg.payload.cylinder_weight\nre
turn
msg;","outputs":1,"noerr":0,"x":360,"y":160,"wires":[["8cef6b9.fe32098","3756272f.d133f8","e166
188b.a68288"]],{"id":"a51801d9.96dda","type":"function","z":"600afed6.9bbbd","name":"JAR_W
EIGHT","func":"global.set('jar_weight',msg.payload.jar_weight)\nmsg.payload=msg.payload.jar_
weight\nreturn
msg;","outputs":1,"noerr":0,"x":360,"y":240,"wires":[["789690c0.15675","3756272f.d133f8","b8b2
85b1.29af38"]],{"id":"be93b0ab.775fc","type":"ui_text","z":"600afed6.9bbbd","group":"45e19a37.
9354e4","order":2,"width":0,"height":0,"name":"","label":"Cylinder Status :
","format":"{{msg.payload}}","layout":"row-
center","x":720,"y":80,"wires":[]},{id":"f6e0f8ae.db6628","type":"ui_text","z":"600afed6.9bbbd","g
roup":"805fca1b.089668","order":2,"width":0,"height":0,"name":"","label":"Jar status
:","format":"{{msg.payload}}","layout":"row-
center","x":730,"y":320,"wires":[]},{id":"dfb6b71c.e13998","type":"function","z":"600afed6.9bbbd"
,"name":"Cylinder
Status","func":"global.set('cylinder_Status',msg.payload.status)\nmsg.payload=msg.payload.stat
us\nreturn
msg;","outputs":1,"noerr":0,"x":360,"y":100,"wires":[["be93b0ab.775fc","3756272f.d133f8"]],{"id":
"76b5423c.33d19c","type":"function","z":"600afed6.9bbbd","name":"Jar
status","func":"global.set('jar_status',msg.payload.jar_status)\nmsg.payload=msg.payload.jar_st
atus\nreturn
msg;","outputs":1,"noerr":0,"x":360,"y":300,"wires":[["f6e0f8ae.db6628","3756272f.d133f8"]],{"id"
:"8d3a73c6.77782","type":"function","z":"600afed6.9bbbd","name":"Leakage","func":"global.set('l
eak',msg.payload.leak)\nmsg.payload=msg.payload.leak\nreturn
msg;\n","outputs":1,"noerr":0,"x":360,"y":40,"wires":[["44ff9851.945748","3756272f.d133f8"]],{"id
":"44ff9851.945748","type":"ui_text","z":"600afed6.9bbbd","group":"45e19a37.9354e4","order":3,
"width":0,"height":0,"name":"","label":"Gas Leakage: ","format":"{{msg.payload}}","layout":"row-
```



```

center","x":720,"y":40,"wires":[]},{id:"b86a1410.c92898","type":"http
in","z":"600afed6.9bbbd","name":"","url":"/data","method":"get","upload":false,"swaggerDoc":"","x
":120,"y":460,"wires":["77d65505.0fb62c"]},{id:"94230dc6.0eb92","type":"http
response","z":"600afed6.9bbbd","name":"","statusCode":"","headers":{"x":750,"y":460,"wires":[]
},{id:"77d65505.0fb62c","type":"function","z":"600afed6.9bbbd","name":"data","func":"msg.payload
oad={\"Cylinder Weight\":global.get(\"Cylinder_weight\"),\\n'Cylinder
status':global.get(\"cylinder_Status\"),\\n'Leakage':global.get(\"leak\"),\\n'Jar
weight':global.get(\"jar_weight\"),\\n'Jar status':global.get(\"jar_status\"),\\n'Fan
status':global.get(\"FAN\")}\\nreturn
msg;\",\"outputs\":1,\"noerr\":0,\"x\":390,\"y\":460,\"wires":["94230dc6.0eb92"]},{id:"d25b01bf.25f76","
type":"function","z":"600afed6.9bbbd","name":"FAN","func":"global.set('FAN',msg.payload.FAN_
STATUS)\\nmsg.payload=msg.payload.FAN_STATUS\\nreturn
msg;\",\"outputs\":1,\"noerr\":0,\"x\":350,\"y\":360,\"wires":["9f3e7047.b0482","3756272f.d133f8"]},{id:"
9f3e7047.b0482","type":"ui_text","z":"600afed6.9bbbd","group":"805fca1b.089668","order":3,"wi
dth":0,"height":0,"name":"","label":"Fan Status : ","format":"{{msg.payload}}","layout":"row-
center","x":730,"y":380,"wires":[]},{id:"e166188b.a68288","type":"ui_chart","z":"600afed6.9bbbd
","name":"","group":"45e19a37.9354e4","order":4,"width":0,"height":0,"label":"Cylinder
Weight","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nodata":"
","dot":false,"ymin":0,"ymax":15,"removeOlder":1,"removeOlderPoints":"","removeOlderUnit":
"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#
2ca02c","#98df8a","#ba1212","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x
":720,"y":120,"wires":[]},{id:"b8b285b1.29af38","type":"ui_chart","z":"600afed6.9bbbd","name"
":"","group":"805fca1b.089668","order":4,"width":0,"height":0,"label":"Jar
weight","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nodata":"
","dot":false,"ymin":0,"ymax":1600,"removeOlder":1,"removeOlderPoints":"","removeOlderUnit
":"3600","cutout":0,"useOneColor":false,"useUTC":false,"colors":["#18496d","#aec7e8","#ff7f0e",
"#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,
"x":730,"y":240,"wires":[]},{id:"857e7ad0.9e65f8","type":"ibmiot","z":"","name":"","keepalive":"6
0","serverName":"","cleansession":true,"appId":"","shared":false},{id:"45e19a37.9354e4","type"
:"ui_group","z":"","name":"cylinder","tab":"cec970f7.87c2","order":3,"disp":true,"width":"6","collap
se":false},{id:"805fca1b.089668","type":"ui_group","z":"","name":"Jar","tab":"cec970f7.87c2","or
der":2,"disp":true,"width":"6","collapse":false},{id:"cec970f7.87c2","type":"ui_tab","z":"","name":"
SMART KITCHEN","icon":"dashboard","disabled":false,"hidden":false}]

```

NODE-RED WORKSPACE LINK:

<https://node-red-suecq.eu-gb.mybluemix.net/red/#flow/600afed6.9bbbd>

NODE-RED DASHBOARD LINK

https://node-red-suecq.eu-gb.mybluemix.net/ui/#/0?socketid=gg-t4MvB8v_eurVmAAAB