

SCHOOL OF DIGITAL MEDIA AND INFOCOMM TECHNOLOGY (DMIT)

IOT CA2 Step-by-step Tutorial

DIPLOMA IN BUSINESS INFORMATION TECHNOLOGY
DIPLOMA IN INFORMATION TECHNOLOGY
DIPLOMA IN INFOCOMM SECURITY MANAGEMENT

ST0324 Internet of Things (IOT)

Date of Submission: 21st August 2017

Prepared for: Ms Dora Chua

Class: DISM/FT/3A/34

Submitted by: Lim Xin Li

Student ID	Name
1529546	Lim Xin Li
1444968	Radin Ayuwandira Binte Radin Amirmuminin
1551354	T. Puvarneswaren Raja

Table of Contents

Section 1 Overview of project.....	3
A. Where we have uploaded our tutorial	3
B. Why have we chosen to upload to this site	3
C. What have we uploaded	3
D. What is the application about?	3
E. Summary of the steps that will be described	4
F. How does the final RPI set-up looks like?	4
G. How does the web application look like?	5
Section 2 Hardware requirements	7
A. Hardware Checklist.....	7
Motion Sensor	7
Light Emitting Diode (LED)	7
Raspberry Pi camera (piCam)	7
330 Ω Resistors	7
Male to male jumper wires	7
Female to male jumper wires.....	8
B. Fritzing Diagram.....	8
C. Software Required	8
Section 3 Create IoT Bluemix app	9
A. Create a Cloud Foundry App	9
B. Set up Bluemix IoT Service	11
C. Create a toolchain.....	14
Section 4 Node-RED Bluemix	17
A. Open Node-RED in IBM.....	17
B. Create flow in IBM Bluemix Node-RED	18
Section 5 Send sensor data to Bluemix	20
A. Install required node in Raspberry Pi	20
Create flow in RPi Node-RED	20
B. Detect motion and control lights	20
C. Send data to Watson IoT.....	22
D. Video component	25
Section 6 Receive command from Bluemix	27
A. Receive commands from web application.....	27

Section 7 Expected Outcome	29
A. Deploy apps	29
Section 8 Tasklist	31
Individual tasks.....	31

Section 1

Overview of project

A. Where we have uploaded our tutorial

We have uploaded our tutorial under the public domain and it can be found in the link below:
<https://github.com/Penelope1rose/RPXi-IDS>

B. Why have we chosen to upload to this site

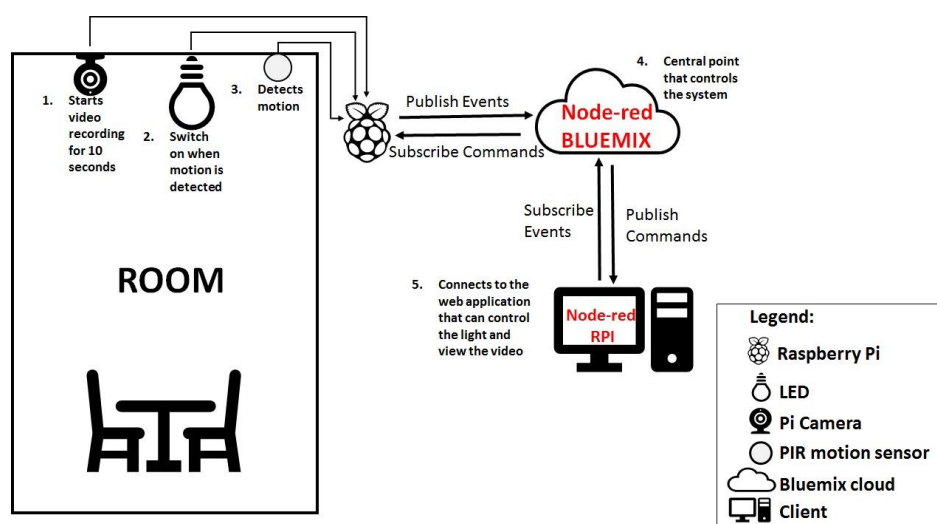
We have been using Github for our module, Enterprise Application Development and other modules to share our source codes among the group members, thus we are most familiar with Github interface.

C. What have we uploaded

We have uploaded files that contain the flows for node-red on both the raspberry pi and ibm bluemix. Not to mention, the web application and a step-by-step tutorial on how to build our system is available to achieve the same results.

D. What is the application about?

Our application is an Physical Intrusion Detection System (PIDS) that detects and responds to abnormal behaviour out of operating hours within an organization.



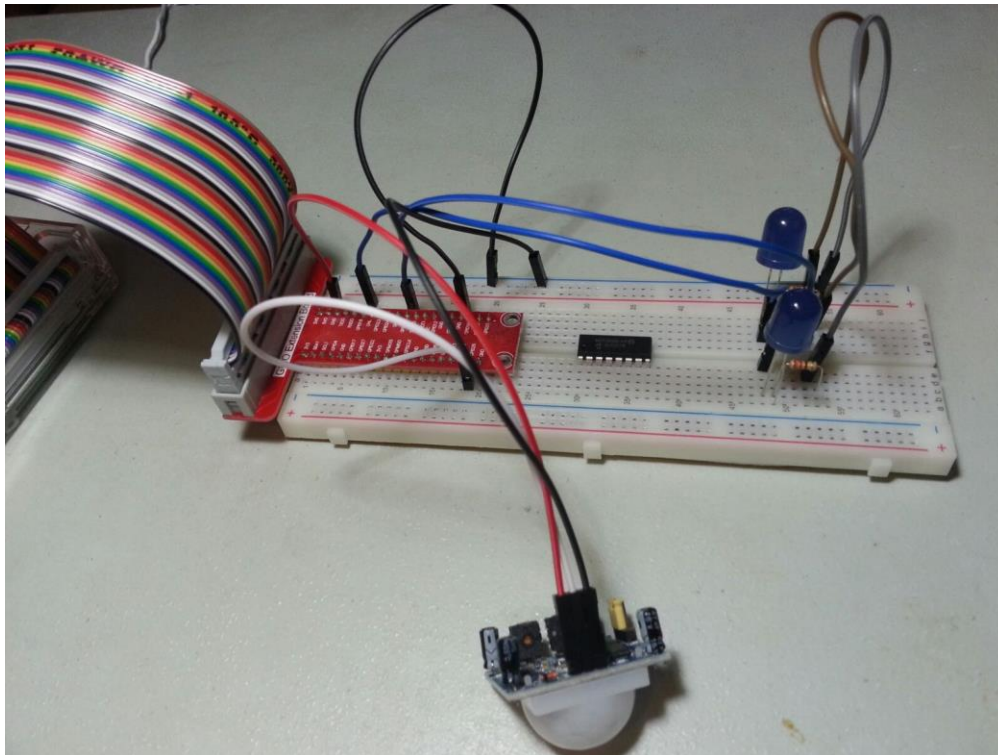
Outside of working hours, PIDS will be activated. It can detect when someone enters a room and automatically switches on the lights and takes a video of the room for 10 seconds while sending

over the status of the room to a web application. If there is an intruder, the security personnel will be alerted. Else, the security personnel can switch off the lights from the web application as well.

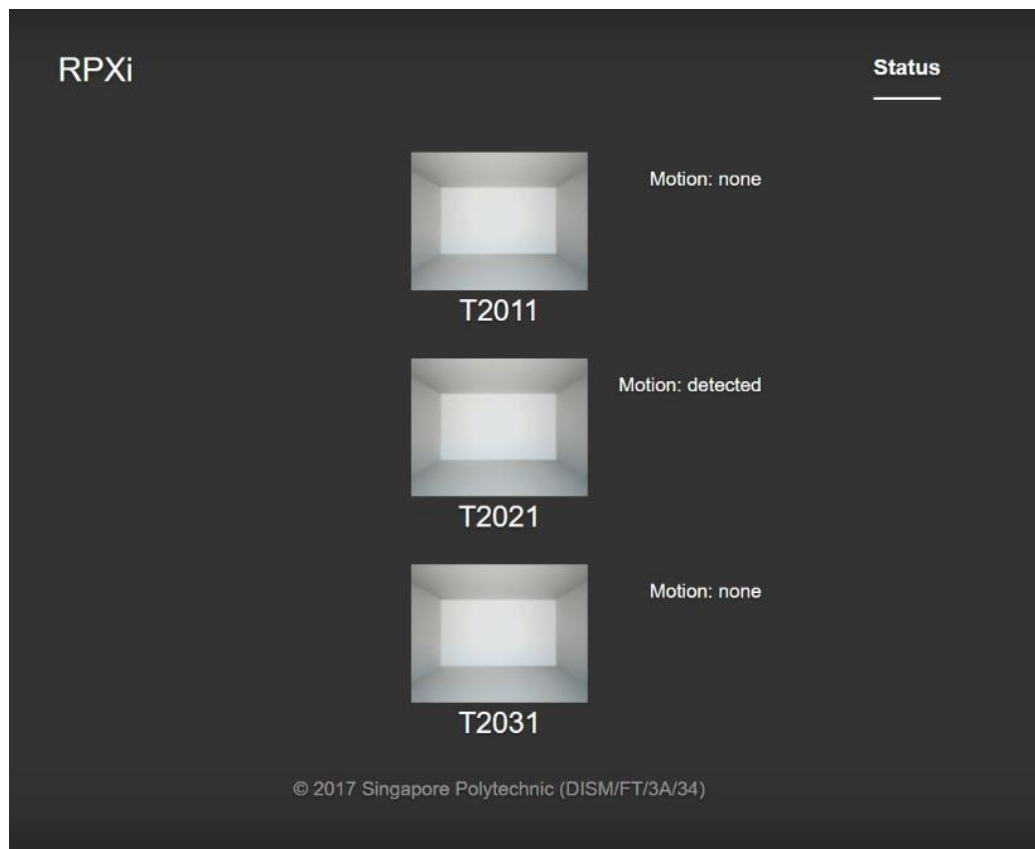
E. Summary of the steps that will be described

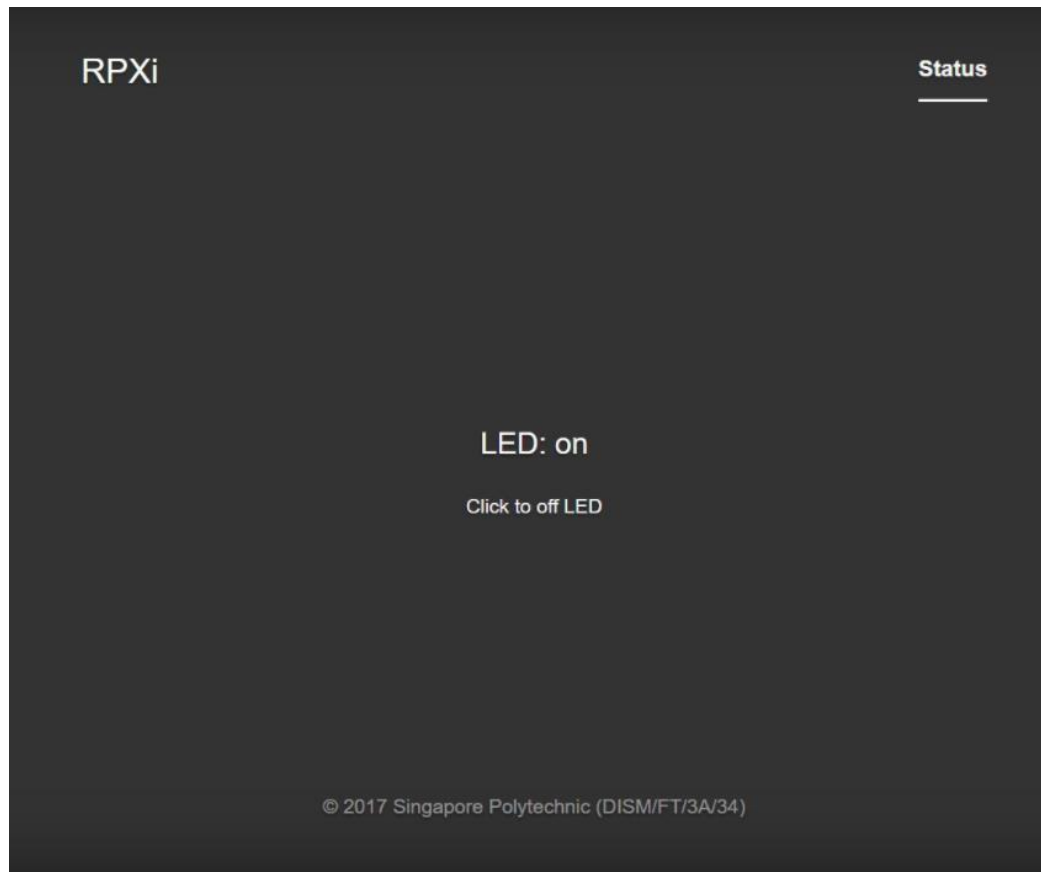
	Section	Description
1)	Overview of project	Get an overview of what the application is about do and where our sources can be found.
Sections 2 to 7 provides the step-by-step instructions to set up the application		
2)	Hardware and software requirements	Provides overview of hardware required, the hardware set-up and the software required
3)	Create IoT Bluemix app	Provides steps to create a cloud foundry app, gateway device type, device type and toolchain in Bluemix console
4)	Node-RED Bluemix	Writing and configuring the necessary nodes
5)	Send sensor data to Bluemix	Use Raspberry Pi Node-RED to code motion detection and switch on light, take video and send sensor data to cloud
6)	Receive command from Bluemix	Nodes will be receiving command from a web application to control light of each individual room
7)	Expected Outcome	Putting everything together and see the expected results
Section 8 provides the listing of tasks		
8)	Tasklist	Provides an overview of what task each individual partake in this project.

F. How does the final RPI set-up looks like?



G. How does the web application look like?





Section 2

Hardware and software requirements

A. Hardware Checklist

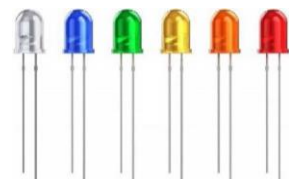
Motion Sensor

You will need one motion sensor. This motion sensor will be used to detect any motion within the premises, indicating any unauthorised persons.



Light Emitting Diode (LED)

You will need two LEDs with any colour of your choice. These LEDs will be used to light up the premise when the motion sensor detects movement.



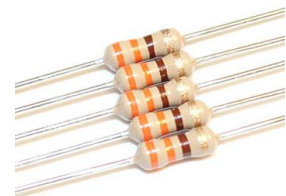
Raspberry Pi camera (piCam)

Make sure you have a piCam connected to the Raspberry Pi. This piCam will start to take a video when the motion sensor detects movement and will stop when demanded by the personnel.



330 Ω Resistors

You will need two 330 Ω resistors for the two LEDs. These 330 Ω resistors will be used to limit current flow so that the LED and RPi will not be damaged.



Male to male jumper wires

You will need 5 male to male jumper wires. The male to male jumper wires will be used to connect the LED and resistors to the breadboard.

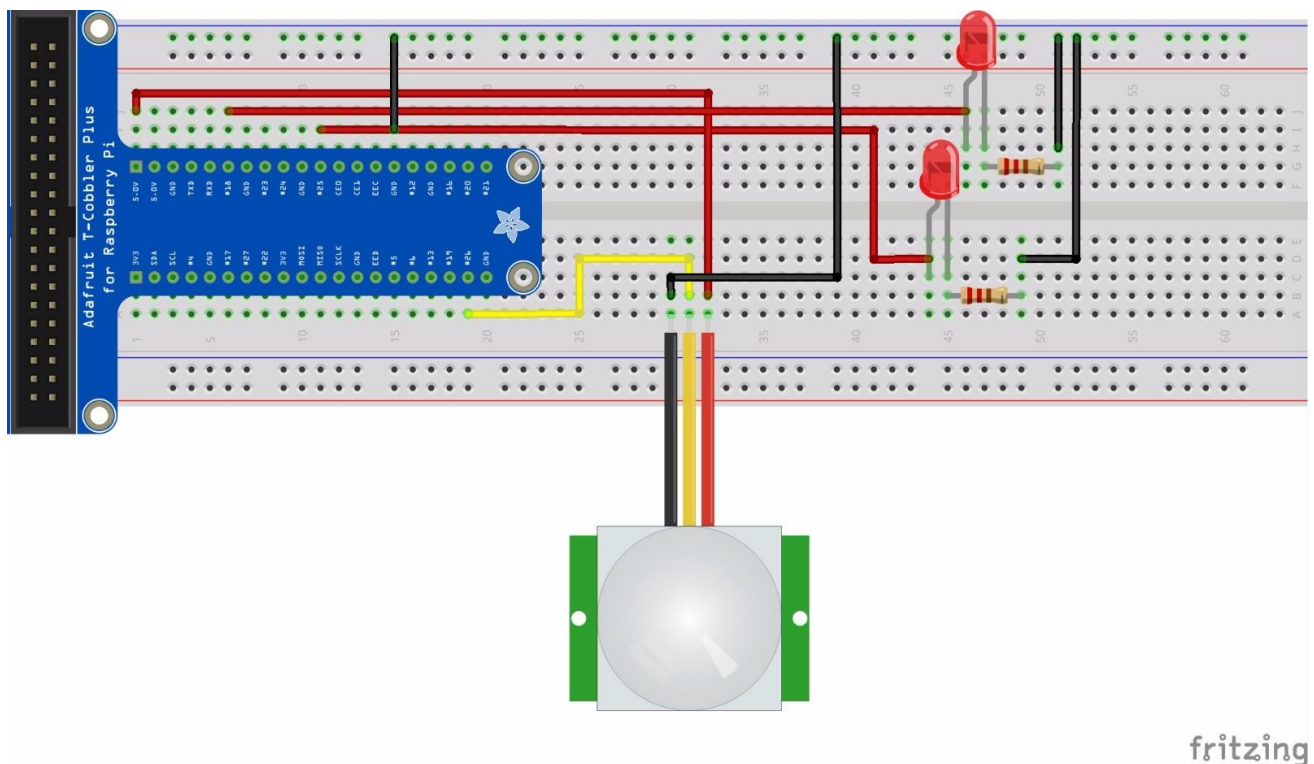


Female to male jumper wires

You will need 3 female to male jumper wires. The female to male jumper wires will be used to connect the motion sensor to the breadboard.



B. Fritzing Diagram



C. Software Required

1. A Node-RED program running on a Raspberry Pi.
 - This program reads the motion sensor value and automatically turns the LED lights on and captures a video of the room every 10 seconds while sending a message to the cloud that they detect motion and the status of the LED. It will also receive commands from the cloud to control the lights.
2. A Node-RED program running on the IBM Bluemix cloud.
 - This program reads the sensor values that were previously sent by the Raspberry Pi. It can send a command to switch off the LED lights and will receive status such as the LED status and the motion sensor status.

Section 3

Create IoT Bluemix app

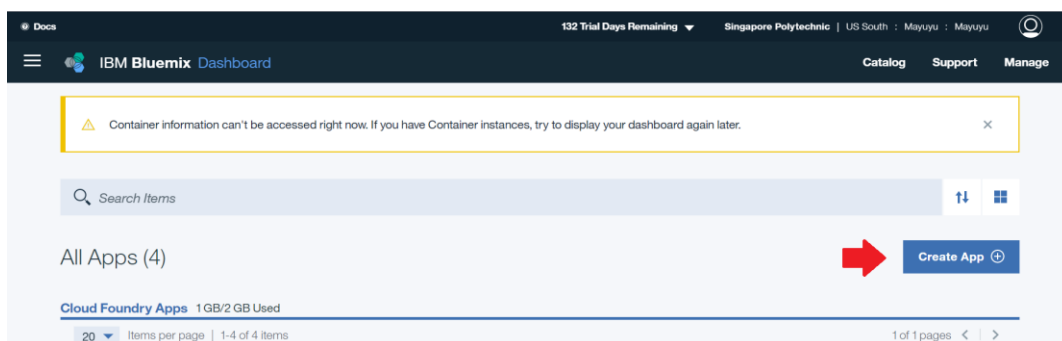
In this section, you will learn how to create a Bluemix App in IBM Bluemix and the necessary services and toolchain. We assume that you would already have created an IBM Bluemix account beforehand. If not, please do follow the tutorial below to create one.

<https://developer.ibm.com/courses/labs/create-bluemix-account-dwc010/>

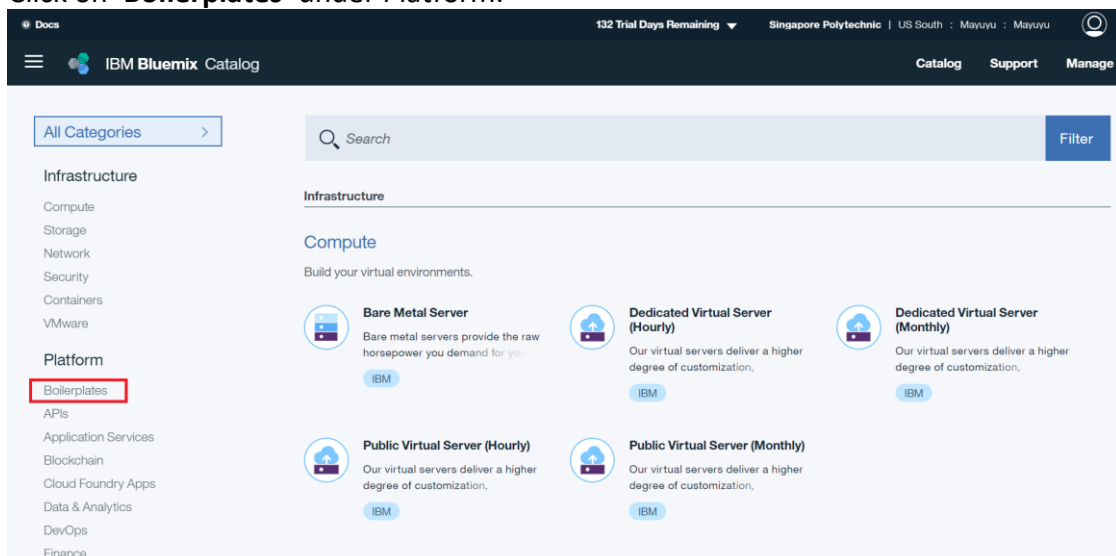
A. Create a Cloud Foundry App

Task

- Log in to your Bluemix account and you should be redirected to the dashboard page.
- Click the **'Create App'** button on the dashboard to create a cloud foundry app.



- Click on **'Boilerplates'** under Platform.



- Choose the **'Internet of Things Platform starter'**.

Get started with a new app, now.

ASP.NET Core Cloudant Starter
Use the Cloudant NoSQL DB Service in an ASP.NET Core

IBM

Internet of Things Platform Starter
Get started with IBM Watson IoT platform using the Node-RED

Lite IBM

IoT for Electronics Starter
IoT for Electronics is an integrated end-to-end solution (made of

IBM

Java Cloudant Web Starter
Use the Cloudant NoSQL DB service with the 'Liberty for

IBM

Java Workload Scheduler Web Starter
This application demonstrates how to use the Workload Scheduler

IBM

LoopBack Starter
This is a sample StrongLoop LoopBack Node.js application,

IBM

MobileFirst Services Starter
Start building your next mobile app with mobile services on Bluemix.

IBM

Node.js Cloudant DB Web Starter
Use the Cloudant NoSQL DB service with the 'SDK for

IBM

Personality Insights Java Web Starter
A simple Java app that uses the Personality Insights service to

IBM

- e) Enter the following information in the fields, leaving the rest of the fields at default, then click the Create button.

← View all

Create a Cloud Foundry App

Internet of Things Platform Starter

Get started with IBM Watson IoT platform using the Node-RED Node.js sample application. With the Starter, you can quickly simulate an Internet of Things device, create cards, generate data, and begin analyzing and displaying data in the Watson IoT Platform dashboard.

Lite IBM

App name:

RPXi-CA2-IDS

Host name:

RPXi-CA2-IDS

Domain:

mybluemix.net

Select region to deploy in:

US South

Choose an organization:

Mayuyu

Choose a space:

Mayuyu

- f) Bluemix will start provisioning your app with a status message “Starting...” This process may take up to 10 minutes, so wait patiently.

- g) After 10 minutes, the status should change to “Running”.

Getting started

Overview

Runtime

Connections

Logs

Monitoring

API Management

Cloud Foundry apps / RPXi-CA2-IDS

RPXi-CA2-IDS Running [Visit App URL](#)

Routes

Runtime

BUILDPACK
SDK for Node.js™

INSTANCES
All instances are running
Health is 100%

MB MEMORY PER INSTANCE

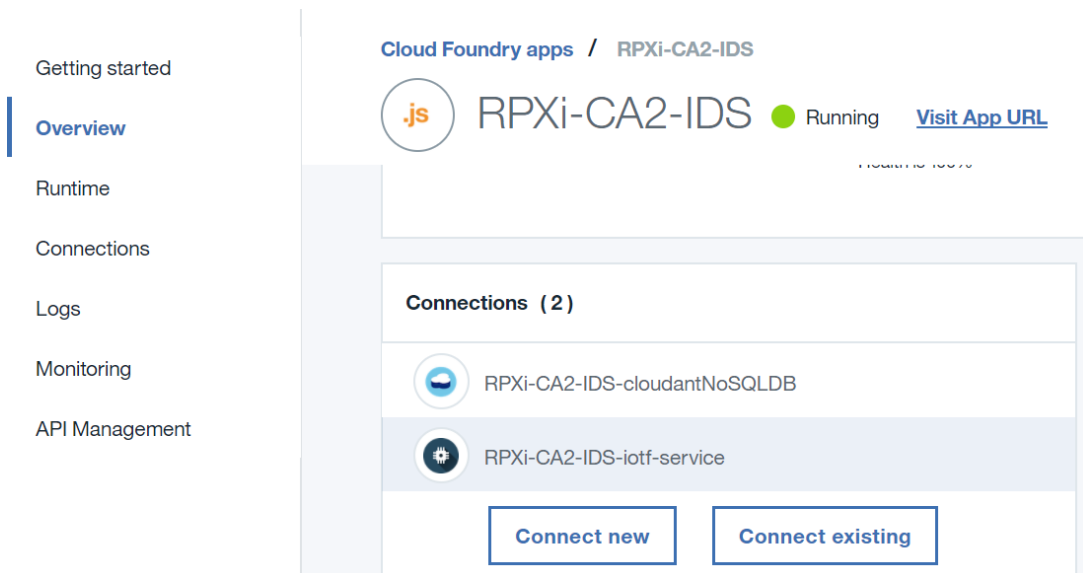
TOTAL MB ALLOCATION
1.25 GB still available

B. Set up Bluemix IoT Service

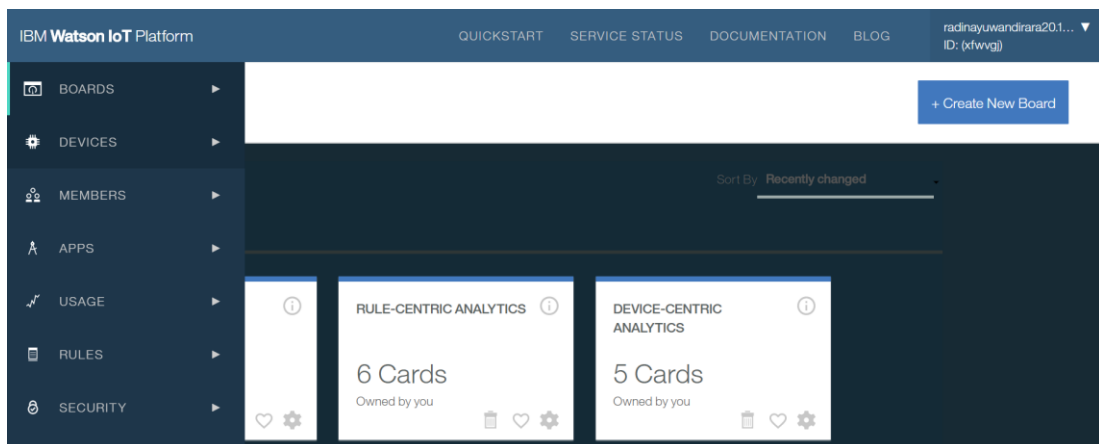
We will now create a gateway device type and a device type. This is to allow the Bluemix application to be able to communicate with the Raspberry Pi.

Task

- a) Under connections, click the iotf-service and click Launch.



- b) Go to devices and click 'Add Device'.



Devices

Browse | Diagnose | Action | Device Types

Refresh + Add Device

Device ID	Device Type	Class ID	Date Added
-----------	-------------	----------	------------

This table shows a summary of all added devices. It can be filtered, organized, and searched on multiple device criteria. You can get started by adding devices using the Add Device button at the bottom of the page, or by using our API.

- c) Click on 'Create device type'.

Add Device

Choose Device Type

Choose Device Type

Or

Create device type

- d) Click on 'Create gateway type'.

Create Device Type

Create Type

Create device type

Create gateway type

- e) Enter the name as shown below and click 'Next'.

✓ Create Type

General Information

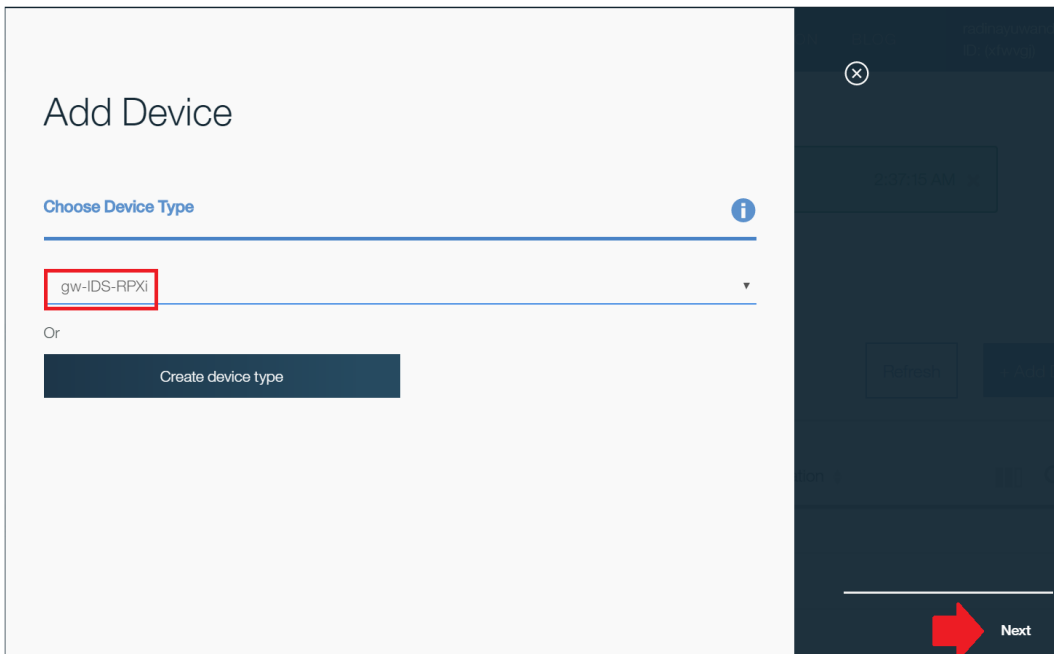
Name gw-IDS-RPXi

The device type name is used to identify the device type uniquely, using a restricted set of characters to make it suitable for API use.

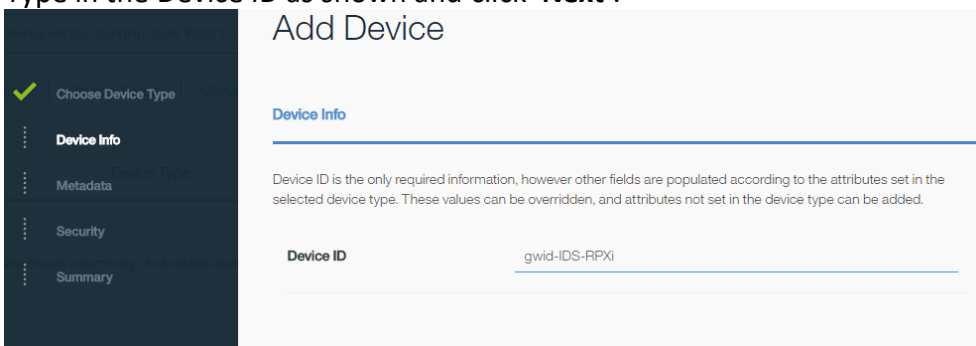
- f) On the next few pages, do not fill in any information but just click 'Next'. Finally, click 'Create'.



- g) After creating the device, do not exit. Choose the device type as the device you have just created and click 'Next'.



- h) Type in the Device ID as shown and click 'Next'.



- i) Click 'Next' at the Metadata screen.

- j) Type in the following as your Authentication Token

Add Device	
<div> <div>✓ Choose Device Type</div> <div>✓ Device Info</div> <div>✓ Metadata</div> <div>..... Security</div> <div>..... Summary</div> </div>	<p>Security</p> <p>You have two options:</p> <p>Auto-generated authentication token</p> <p>Allow the service to generate an authentication token for you. The token will be 18 characters long and will contain a mix of alphanumeric characters and symbols. The token will be returned to you at the end of the registration process.</p> <p>Self-provided authentication token</p> <p>Provide your own authentication token for this device. The token must be between 8 and 36 characters long, and should contain a mix of lower and upper case letters, numbers, and symbols (hyphen, underscore, exclamation-point, ampersand, at sign, question mark, period, right and left parentheses are permitted). The token should be free of repetition, dictionary words, user names, and other predefined sequences.</p> <p>Provide a token (optional) <input type="text" value="AUTHTOKEN-IDS-RPX "/></p>
<p>k) On the summary page, click 'Add'.</p>	
<p>l) When your device has been successfully created, you will see the "Your Device Credentials" page as shown.</p>	
<p>m) Note the Organization ID, DeviceType, DeviceID and Authentication Token values.</p>	
<p>n) Copy and Paste the text on this screen and save it to a text file on your laptop. You will need this later.</p>	
<p>o) Click the "X" button on the top-right hand corner to close the window.</p>	

C. Create a toolchain

We will now create a toolchain that serves as a place to reference the image, css and js in the html nodes in IBM Node-Red.

Task

- Go back to your dashboard and click into your cloud foundry app.
- Scroll down all the way and click '**Enable**' under '**Continuous delivery**'.

Continuous delivery

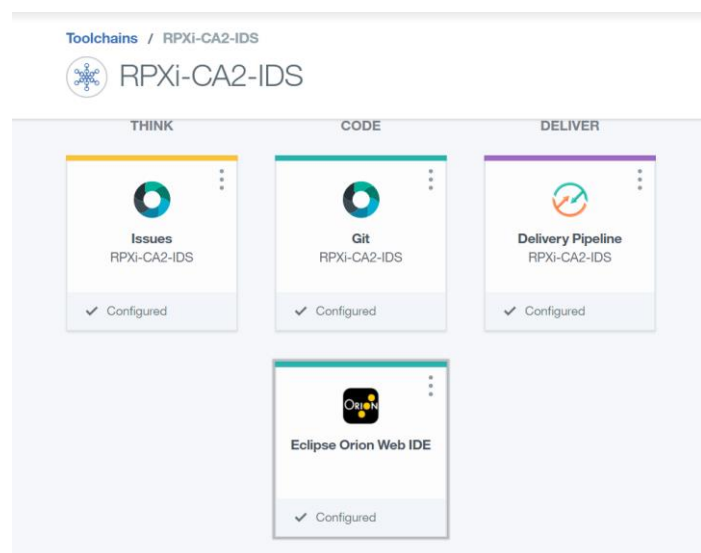
Continuous delivery is not enabled for this app.

Enable continuous delivery to automate builds, tests, and deployments through the Delivery Pipeline, GitHub, and more.

Enable

c) Leave everything as default and click **'Create'**.

d) Afterwards, click into **'Eclipse Orion Web IDE'**.

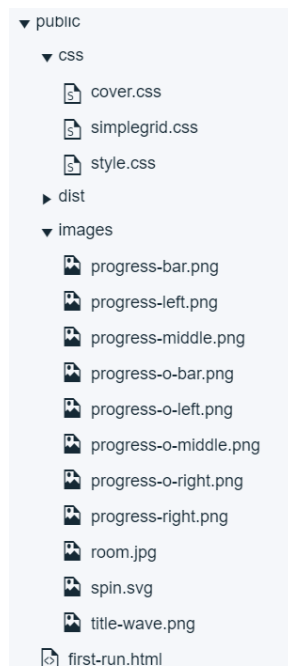


e) Remove **'index.html'** file under **'Public'**.

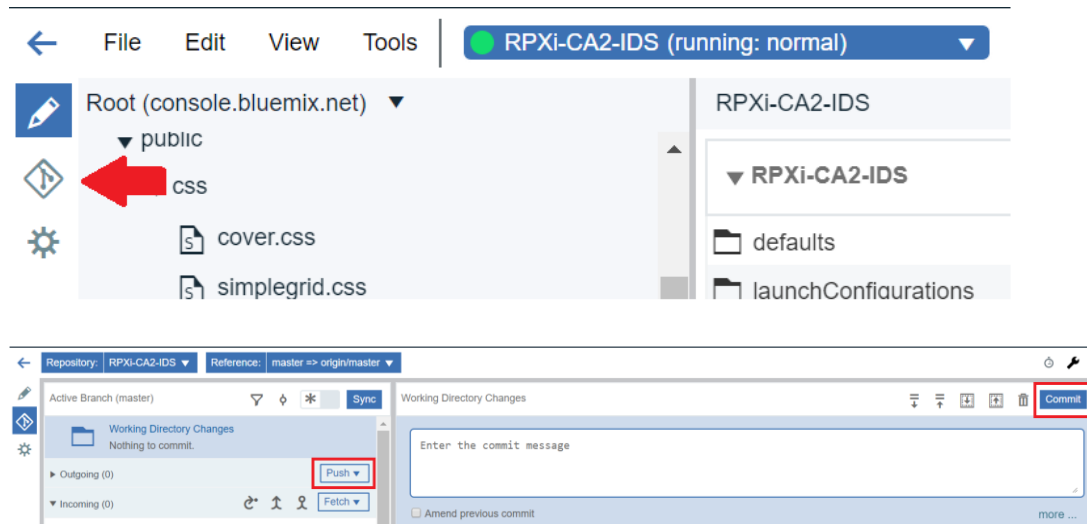
Add in the **'dist'** folder under **'Public'**.

Add **'cover.css'** in the **'css'** folder.

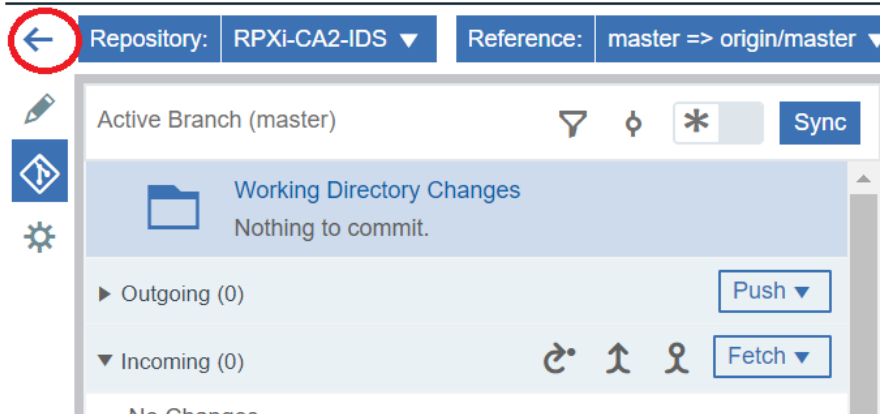
Add **'room.jpg'** in the **'images'** folder.



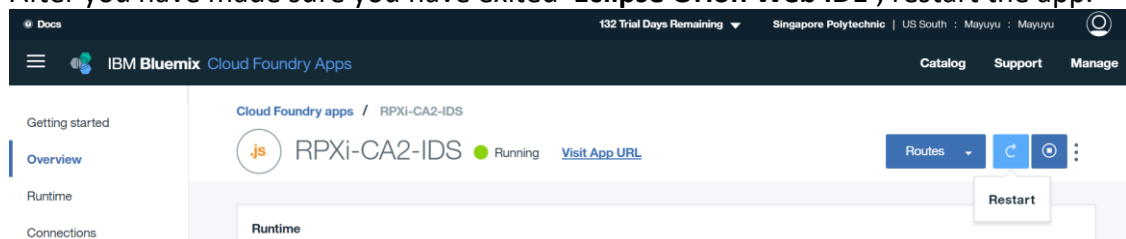
f) Afterwards, go to **'Git'** to commit the changes and push.



g) Afterwards, click the back arrow to go back.



h) After you have made sure you have exited 'Eclipse Orion Web IDE', restart the app.



i) Wait for the app to be back in the 'Running' state.

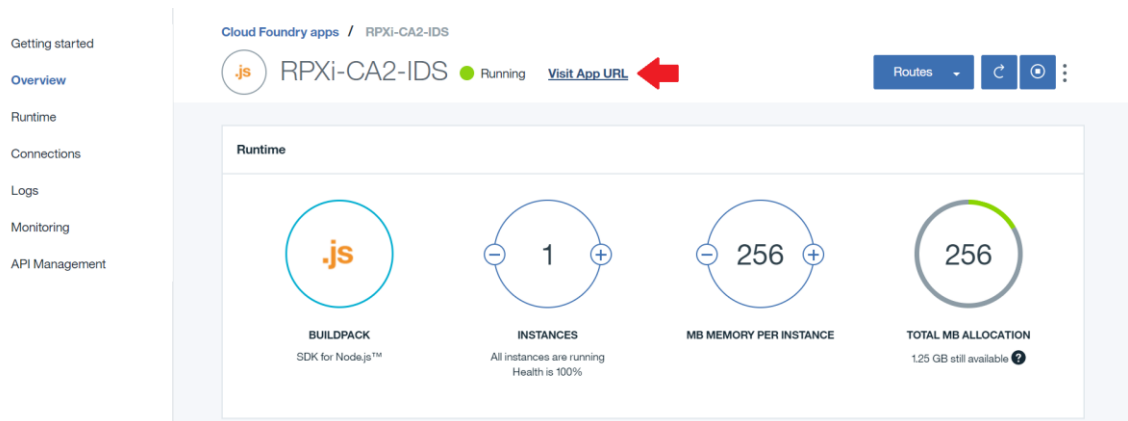
Section 4

Node-RED Bluemix

A. Open Node-RED in IBM

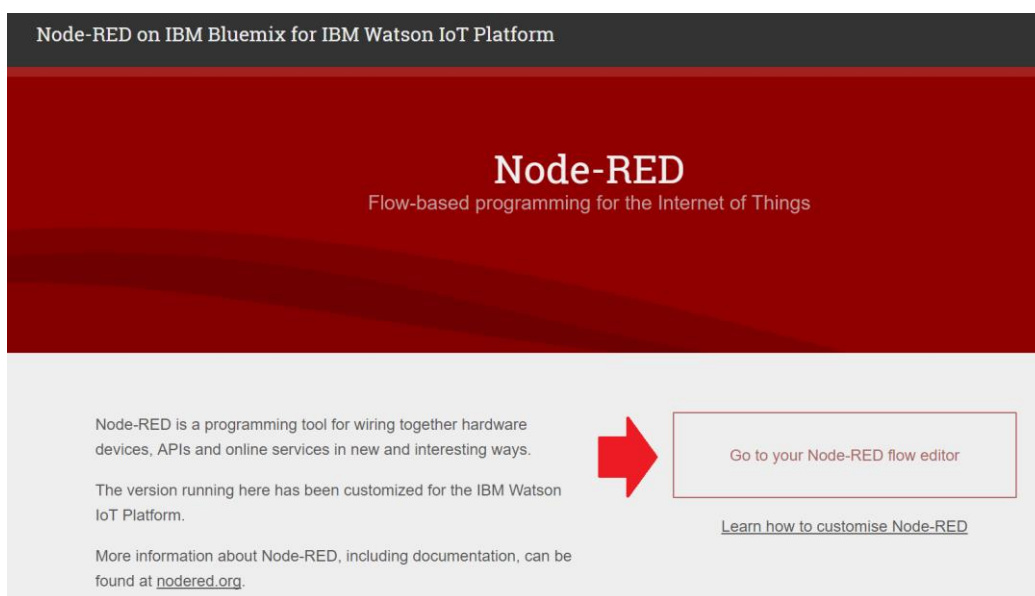
Task

- j) Continuing from the previous step, proceed to the App's URL by clicking on 'Visit App URL'.



- k) You will be directed to Node-RED's page where you will be asked to setup a username and password to secure your Node-RED.

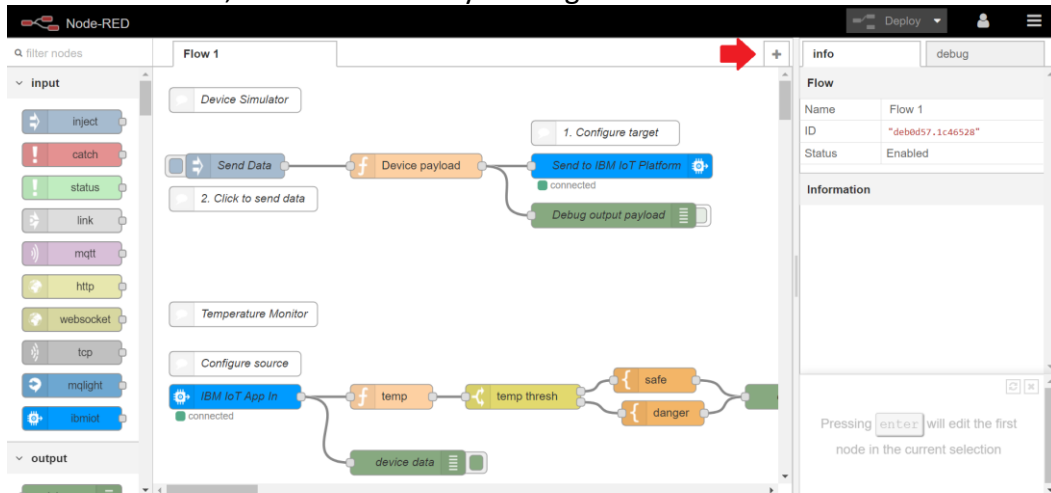
- l) Enter the username and password of your choice. Then, proceed to your Node-RED flow editor.



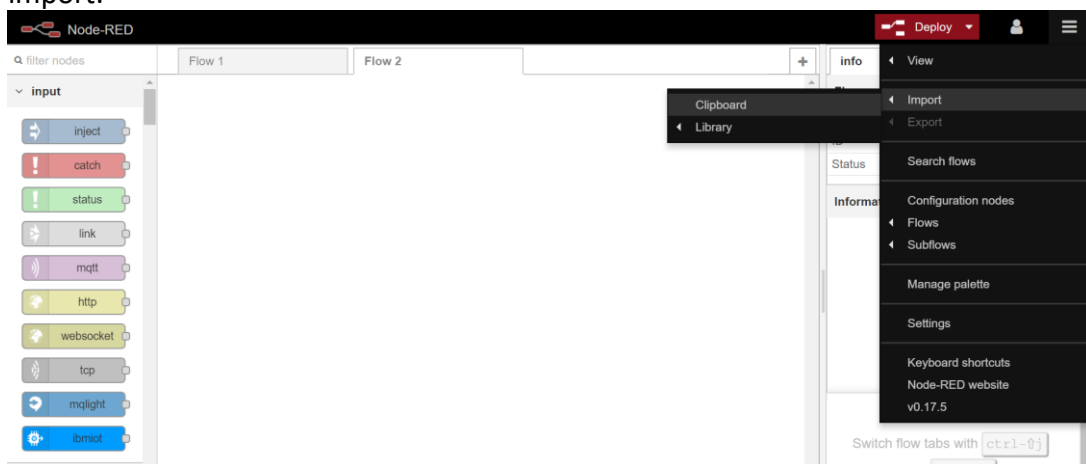
B. Create flow in IBM Bluemix Node-RED

Task

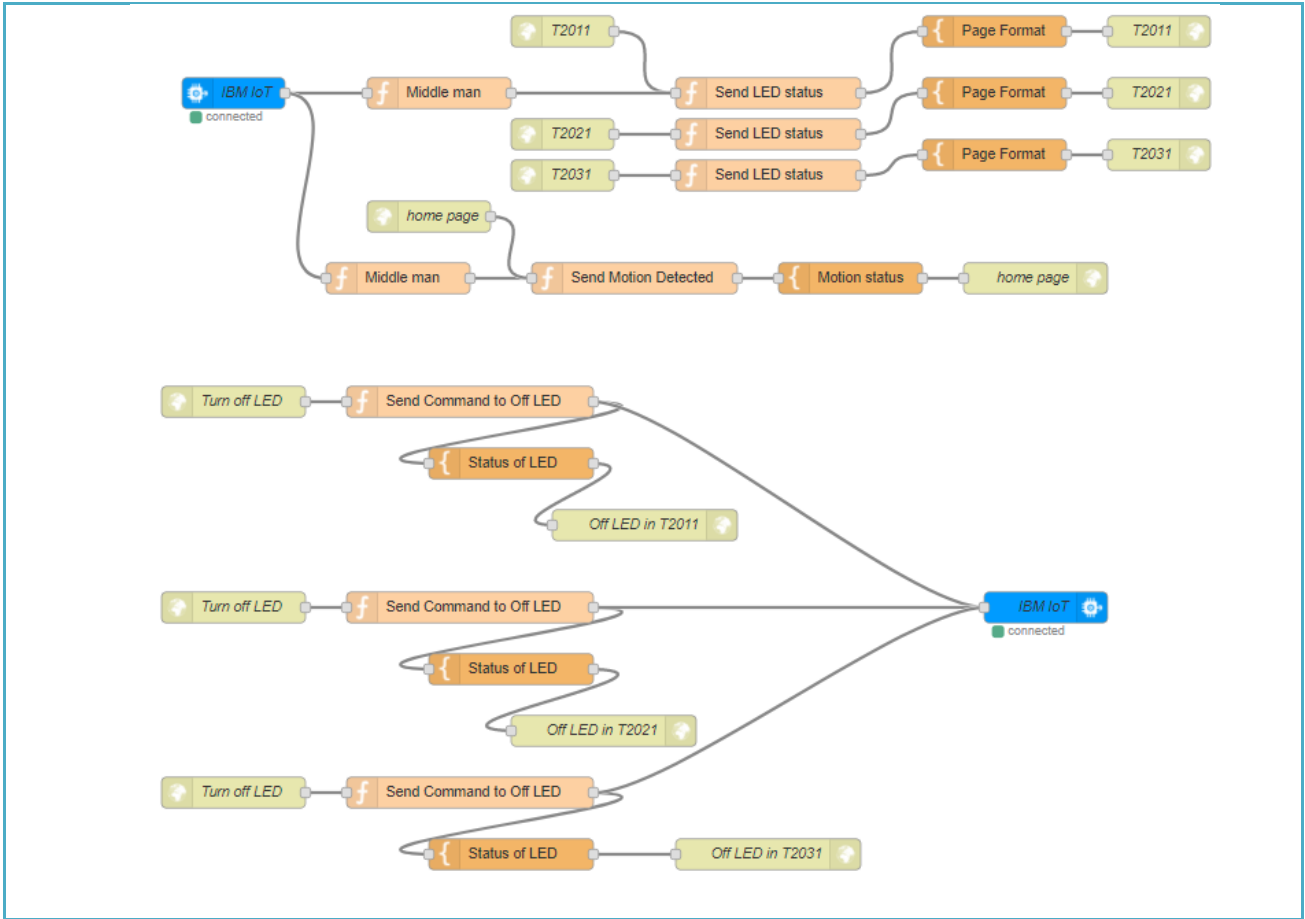
- a) Inside Node-RED, add a new flow by clicking on the '+' button.



- b) Import flows using clipboard. Paste the source code found in *ibm_flow.doc* and click import.



- c) The flow should look like this.



Section 5

Send sensor data to Bluemix


In this section, you will learn how to configure how to get your Raspberry Pi to detect motion and control lights and send these data to your IBM Bluemix using Watson IoT. On the side, it will also take a video for 10 seconds when motion is detected.

A. Install required node in Raspberry Pi

Task	
a)	Open a Terminal window and install the Node-RED node on your Raspberry Pi. <code>sudo npm i -g node-red-contrib-ibm-watson-iot</code>
b)	Once successful, reboot your machine. <code>sudo reboot now</code>
c)	Start Node-RED in RPi. <code>node-red start</code>

Create flow in RPi Node-RED

We will be creating all of the nodes and configurations necessary in the RPi.

Task	
a)	Open a browser and browse to the Node-RED webpage. The URL would be the IP address of your Raspberry Pi, followed by a semi colon and the 1880 port e.g. http://192.168.0.111:1880
b)	Add a new flow so that you can work on a brand new workspace. 

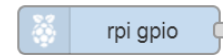
B. Detect motion and control lights

We will first create the motion sensor to switch the lights on.

Task

- a) Create a new flow and drag an **rpi gpio in** node under the Raspberry_Pi section to the workspace.

▼ Raspberry_Pi



- b) Double click the node and configure the node as shown. Name it 'PIR'.

Edit rpi-gpio in node

Delete Cancel Done

● GPIO Pin 37 - GPIO26 ▼ Pi 3 Model B

↑ Resistor? pullup ▼ Debounce mS

☐ Read initial state of pin on deploy/restart?

📌 Name PIR

Pins in Use: 12,22,37

Tip: Only Digital Input is supported - input must be 0 or 1.

- c) Drag the **trigger** node under function section and configure it as follows. Name it 'Reset light'

▼ function

function template delay **trigger**

Edit trigger node

Delete Cancel Done

Send 0 1

then wait to be reset ▼

Reset the trigger if: • msg.reset is set • msg.payload equals 0

📌 Name Reset light

- d) Drag out 2 **rpi gpio out** node under the Raspberry_Pi section and. Name it 'LED1' and 'LED2' respectively.

▼ Raspberry_Pi



- e) Configure it as follows accordingly.

Edit rpi-gpio out node

Delete Cancel Done

GPIO Pin 12 - GPIO18 Pi 3 Model B

Type Digital output

☒ Initialise pin state?

initial level of pin - low (0)

Name LED1

Edit rpi-gpio out node

Delete Cancel Done

GPIO Pin 22 - GPIO25 Pi 3 Model B

Type Digital output

☒ Initialise pin state?

initial level of pin - low (0)

Name LED2

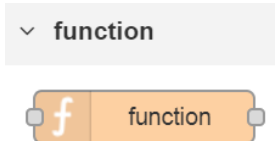
f) Then, connect the nodes together. Follow the image below.

C. Send data to Watson IoT

We will now make it so that it sends the data to the cloud.

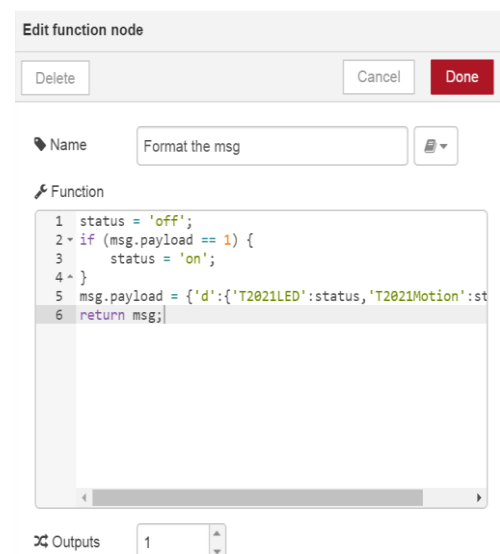
Task

- a) Drag out a **function** node and name it 'Format the msg'

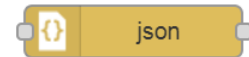


- b) Fill in the blank with the following details.

```
status = 'off';
if (msg.payload == 1) {
  status = 'on';
}
msg.payload =
{'d':{'T2021LED':status,'T2021Motion':status}};
return msg;
```



- c) Drag a **json** node under function section. Double click it and name it '**Status**'.



- d) Drag 2 nodes out from under function section, the **debug** node, which will be automatically renamed as '**msg.payload**' and **Watson IoT** node as '**IBM IoT Device**'.



- e) For the debug node, select msg as output and configure it as follows.

Edit debug node

Delete

Cancel

Done

Output

▼ msg.payload

to

debug tab ▼

Name

Name

- f) For the Watson IoT node, configure it as follows.

Choose '**Connect as Gateway**' and select '**Registered**'

Name it as '**Gateway**'

Next, click on the 'Pencil Icon' to add the device credentials created earlier.

- g) Fill in the fields you have noted earlier on and click on '**Update**' button.

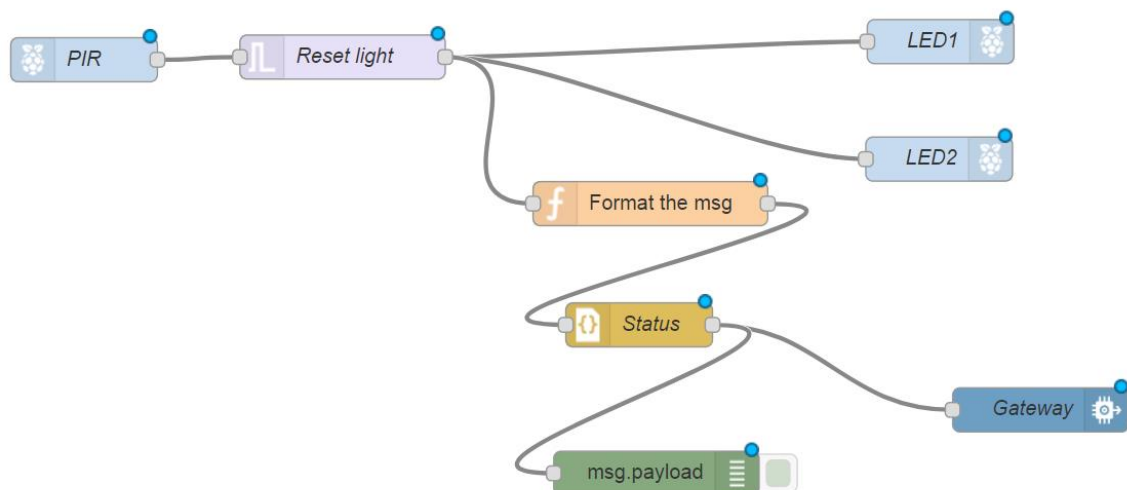
You can name it '**IDS-RPXi**'

DeviceType should be gw-**<devicetype>**

DeviceID should be gwid-**<deviceid>**

Auth Token should be **<token>**

- h) Connect the nodes together.



D. Video component

We will now make it so that it takes a video every 10 seconds.

Task

- a) Drag out a **function** node and name it 'Change name of video'.

```
msg.payload = '/home/pi/Videos/recording_' +  
Date.now() + '.h264 -t 10000';  
return msg;
```

Dialog: Edit function node

Buttons: Delete, Cancel, Done

Name: Change name of video

Function:

```
1 msg.payload = '/home/pi/Videos/recording_' + Date.now() + '.h264 -t 10000';  
2 return msg;  
3
```

Outputs: 1

- b) Drag a **exec** node from advanced section.

Name it 'Video'.

advanced

Nodes: watch, feedparse, mcp3008, **exec**

- c) Configure the node as follows.

Dialog: Edit exec node

Buttons: Delete, Cancel, Done

Command: raspivid -o

Append: ☒ msg.payload

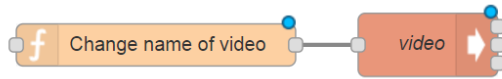
extra input parameters

☐ Use spawn() instead of exec()?

Timeout: optional seconds

Name: video

d) Connect the two together.



Section 6

Receive command from Bluemix

A. Receive commands from web application

We will now make it so that it not only does the above things, it also receive commands through the web application.

Task

- a) Drag an **Watson IoT input** node under the input section to the workspace.



- b) Configure the node such that it matches the image.

The one highlighted represents the which room's lights that we wish to switch it off.

Name it '**Stop**' and click on the 'Pencil Icon'.

Edit Watson IoT node

Delete

Cancel

Done

Connect as Gateway

Credentials IDS-RPXi

Subscribe to ☒ Gateway commands ☐ Device commandsCommand

QoS 0

Name Stop

- c) Fill in the fields appropriately with the same credentials we set earlier.

Watson IoT > Edit wiotp-credentials node

Delete

Cancel

Update

Organization xfwvgj

Server-Name xfwvgj.messaging.internetofthings.ibmcloud.com

Device Type gw-IDS-RPXi

Device ID gwid-IDS-RPXi

Auth Token

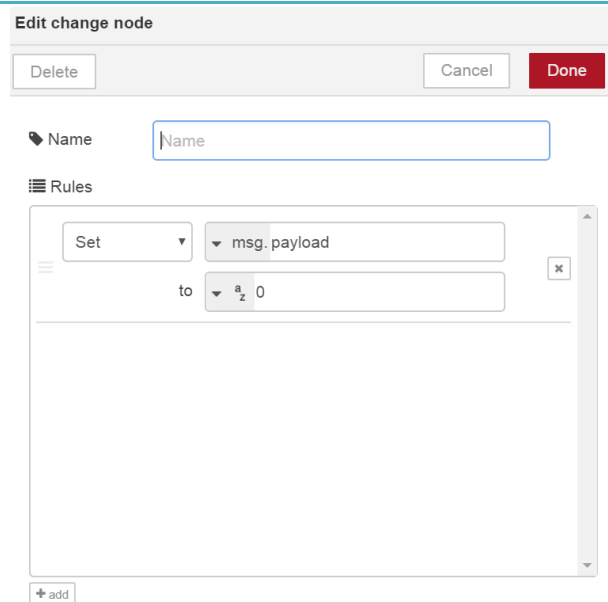
Keep Alive 60 Seconds ☐ Use Clean Session☐ Enable secure (SSL/TLS) connection

Name IDS-RPXi

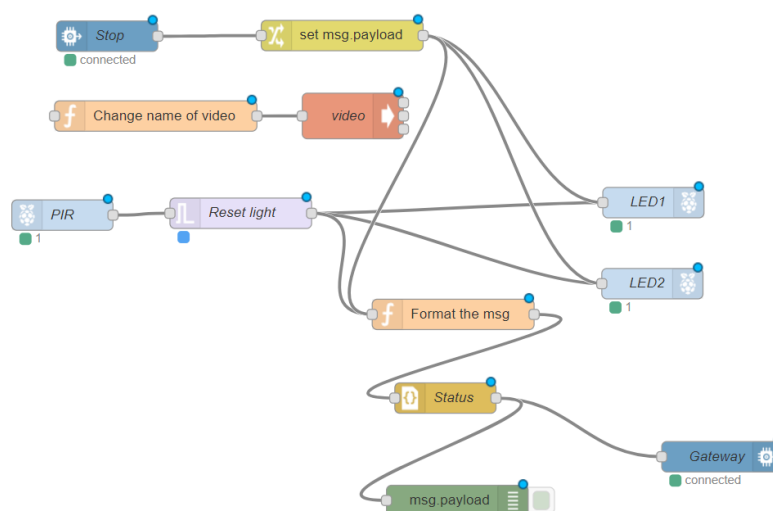
d) Drag out a **change** node from under function section.



e) Just fill in the 'payload' the output.



f) Finally, the complete connection would look like this.

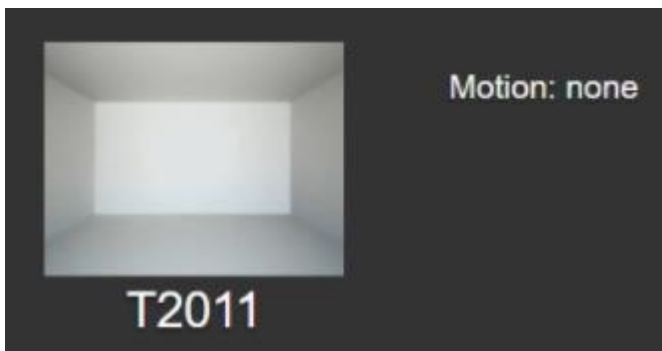
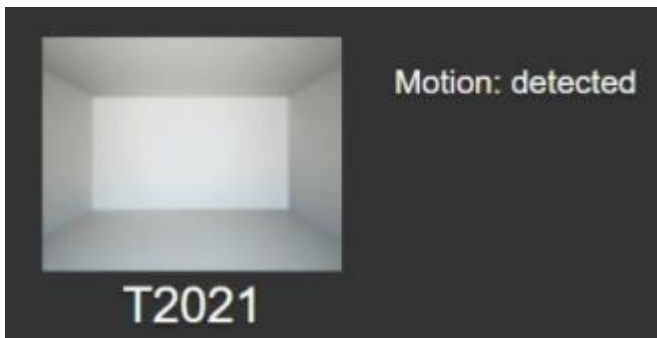


g) You can also import flows using clipboard. Paste the source code found in *rpi_flow.doc* and click import.

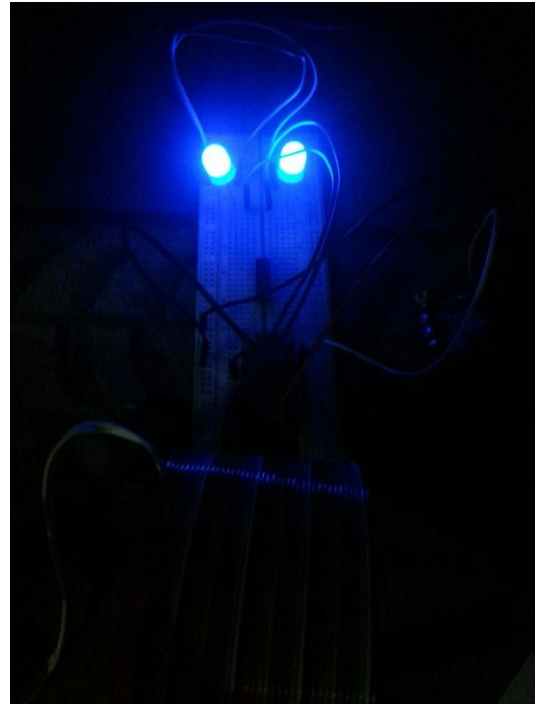
Section 7

Expected Outcome

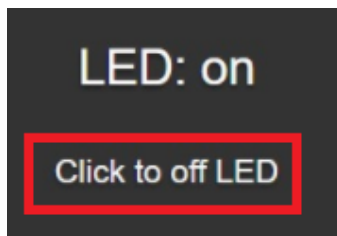
A. Deploy apps

Task
a) Deploy the RPI node-red and IBM node-red.
b) Click 'Visit App URL'.
c) You should see the web application showing status of the rooms.
d) If there is no motion detected, it will display motion as none.  A screenshot of a web application interface. It features a dark background. On the left, there is a square video feed showing a bright, empty room. Below the video feed, the text 'T2011' is displayed in white. To the right of the video feed, the text 'Motion: none' is displayed in white.
e) Else, if there is motion detected, it will display motion as detected.  A screenshot of the same web application interface. The video feed shows a bright, empty room. Below the video feed, the text 'T2021' is displayed in white. To the right of the video feed, the text 'Motion: detected' is displayed in white.
f) Picam will be started and will record for 10 seconds and saved in the file path /home/pi/Videos.

- g) On the breadboard, you should also see that the LED is turned on when the system detects motion.



- h) After ensuring the reason for the motion, you can click into the room where motion is being detected and click the link to turn off LED.



- i) Click the link to go back to the home page.

LED has been turned off
[Go back](#)

Section 8

Tasklist

Individual tasks

Lim Xin Li:

- Documentation
- IBM node-red flow
- Connect RPi node-red with IBM node-red via MQTT
- Web application outlook
- View motion sensor status
- Off LED from web application
- Convert video name
- Testing

Radin Ayuwandira Binte Radin Amirmuminin:

- Fritzing diagram
- Hardware Set-up
- Documentation
- Attempt to extract data from cloud
- Motion sensor detected, send data to cloud
- Testing

T. Puvarneswaren Raja:

- Picam take video
- Motion sensor detected, turn on LED
- Diagram
- Testing

-- End of CA2 Step-by-step tutorial --