# IMPLEMENTATION OF IoT USING MQTT (HOME AUTOMATION)

**Project Report**

**by**

**EE281**

**GROUP-10**

**AASHAV PANCHAL-011463361**
**MOHAMMED ABDUL MUJEEB ANSARI-011168144**
**SNEHAL CHAVAN-011452571**
**SWAPNASHEEL SONKAMBLE-011452610**

**Chao-Li Tarng, Project Advisor**
**12/2016**

**ABSTRACT**

# IMPLEMENTATION OF IoT USING MQTT (HOME AUTOMATION)

By
AASHAV PANCHAL - 011463361
MOHAMMED ABDUL MUJEEB ANSARI - 011168144
SNEHAL CHAVAN - 011452571
SWAPNASHEEL SONKAMBLE - 011452610

With the proliferation of embedded technologies and the pervasiveness of the Internet, it is only natural to connect the two – giving rise to The Internet of things (IoT). IoT is defined as the “network of interconnected things” in which the term “things” could be anything such as sensors, systems or devices whose generated data is needed to be analyzed and acted upon. To help manage data at a low cost and more efficiently, we make use of the MQTT protocol.

MQTT protocol is so lightweight that the smallest measuring and monitoring devices can support it, and transmit data to remote networks. It is also open source, which makes it easy to adapt to a variety of messaging and communication needs.

In this project, we implemented IoT with MQTT. We designed a home automation system, which refers to the automatic and electronic control of household features, activity, and appliances.

# Table of Contents

# Chapter 1

## Introduction

### 1.1 Project goals and objectives

The project is based on implementation of IoT using MQTT protocol for which the Implementation is based Home Automation. An IoT device is just a normal thing which is driven by some computational power and a network connection. The objective of our project is focused on providing computational powers to normal home devices by using suitable microcontroller and actuators along with the network connection which would be fulfilled by the MQTT protocol and thus to implement a Home Automation system which would fulfill our goal of implementation of IoT using MQTT protocol.

This project proposes the design, of a hardware and software based on the concept of Internet of Things (IoT) which could intelligently control the residential loads.

### 1.2 Problems and motivation

IoT has become an immensely popular field during the last decade because of the emerging technologies such as cloud platforms, dropped sensor prices & protocols like MQTT, which makes it possible to capture store and analyze a large amount of data to provide easy solutions to real life problems. IoT aims at making life easier by taking smart actions without requiring human effort.

Thus, it motivated us to get a deep insight of such an emerging technology, which aims at providing simple, and real time solutions.

By implementing the IoT using MQTT protocol, we plan to convert the normal residential devices to smart devices.

## 1.3 Project application and impact

The result of our project includes a user interface through which a user can control the residential devices connected at home from anywhere around the world anytime and could keep a track of all the activities of the connected devices.
This would make it easier to control the devices remotely and keep a track of its activities.

## 1.4 Project results and deliverables

Please refer Chapter 7 for project results.

Project deliverables are available at the following link:

**https://github.com/Swapnasheel/MQTT**

# Chapter 2
# Background and Related Work

## 2.1 Background and used technologies

### 2.1.1 Internet of Things

Just as most of the technological innovations created by the human being through the ages, also the emerging IoT technology is driven by applications that are mainly aimed at improving people's quality of life while saving operational costs. "Internet of Things" is the interconnection of physical devices which are embedded with sensors, actuators which enable these objects to collect and exchange data. Applications related to IoT can be found in several and different domains: energy, health, transportation, environment, etc. Thousands of applications can be identified in each domain and new ones appear every day, requiring a strong interconnection among things. One of the protocol that facilitates the transfer of information between the "Things" is MQTT protocol, which is discussed in the next section.

### 2.1.2 MQTT

Message Queuing telemetry transport protocol is an extremely simple and lightweight messaging protocol. Its publish/subscribe architecture is designed to be open and easy to implement, with up to thousands of remote clients capable of being supported by a single server. These characteristics make MQTT ideal for use in constrained environments where network bandwidth is low or where there is high latency and with remote devices that might have limited processing capabilities and memory.

It is a protocol, which is used for the transport of telemetry data. Telemetry technology allows things to be measured or monitored from a distance. In addition, today, improvements in telemetry technology make it possible to interconnect measuring and monitoring devices at different locations and to reduce the cost of building applications that can run on these smart devices make them even more useful.

## Concepts of MQTT

The MQTT protocol is built upon several basic concepts, all aimed at assuring message delivery while keeping the messages themselves as lightweight as possible.

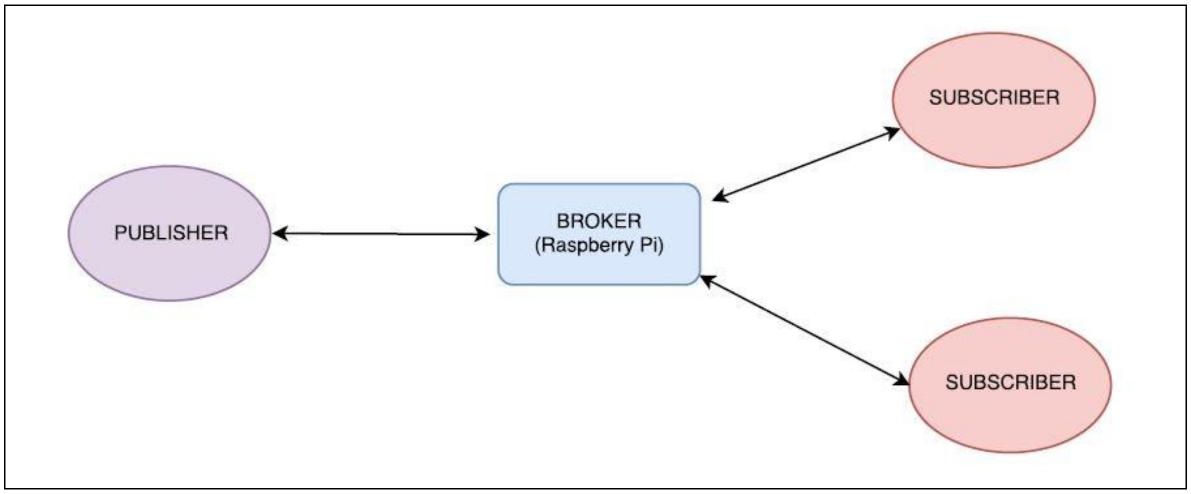


**Figure 1: Basic Functional diagram of MQTT**

## Publish/subscribe

The MQTT protocol is based on the principle of publishing messages and subscribing to topics, which is typically referred to as a *publish/subscribe* model. Clients can subscribe to topics that pertain to them and thereby receive whatever messages are published to those topics. Alternatively, clients can publish messages to topics, thus making them available to all subscribers to those topics.

## Topics and Subscriptions

Messages in MQTT are published to *topics*, which can be thought of as *subject areas*. Clients, in turn, sign up to receive particular messages by subscribing to a topic.

*Subscriptions* can be explicit, which limits the messages that are received to the specific topic at hand or can use wildcard designators, such as a number sign (#) to receive messages for a variety of related topics.

Let us take an example to understand this

MYHOME/GROUNDFLOOR/LIVINGROOM/TEMPERATURE
MYHOME/GROUNDFLOOR/BEDROOM/TEMPERATURE
MYHOME/GROUNDFLOOR/LIVINGROOM/BRIGHTNESS
MYHOME/GROUNDFLOOR/BEDROOM/BRIGHTNESS

A subscription to the topic MYHOME/GROUNDFLOOR/LIVINGROOM/TEMPERATURE returns the temperature of the livingroom.

We can use wildcards to subscribe to bulk data. Wild cards are of two types
SINGLE LEVEL
MULTI LEVEL

SINGLE LEVEL WILDCARD TOPIC LIKE MYHOME/GROUNDFLOOR/+/TEMPERATURE returns the temperature of both the bedroom.

Multi level wildcard topic like MYHOME/GROUNDFLOOR/# returns the temperature and brightness of all the rooms in my house.

**Quality of service levels**

MQTT defines three *quality of service* (QoS) levels for message delivery, with each level designating a higher level of effort by the server to ensure that the message gets delivered. Higher QoS levels ensure more reliable message delivery but might consume more network bandwidth or subject the message to delays due to issues such as latency.

**<u>Quality of service 0</u> (at most once)**

The minimal level is zero and it guarantees a best effort delivery. A message won’t be acknowledged by the receiver or stored and redelivered by the sender. This is often called “fire and forget”.

**<u>Quality of service 1</u> (at least once)**

When using QoS level 1, it is guaranteed that a message will be delivered at least once to the receiver. But the message can also be delivered more than once.

**<u>Quality of service 2</u> (exactly once)**

The highest QoS is 2, it guarantees that each message is received only once by the counterpart. It is the safest and the slowest quality of service level.

## 2.2 State-of-the-art

Let us take a look at some of the devices based on “Internet of Things” which are available in the market.

**<u>Philips hue</u>**: It is an intelligent bulb which helps us control the light, brightness and colors remotely from any corner of the world when connected to the internet.

**<u>Google Home</u>**: It is a complex of three devices, all in one. The speaker, the AI personal assistant, and the smart controller for all your smart devices in your home. The new gadget comes as a small-sized speaker and microphone set, designed to sit in your room and respond to your commands. Amazon too has its version of home assistant known as Amazon Alexa.

## 2.3 Literature survey

The number of smart things is growing exponentially. By 2020, tens of billions of things will be deployed worldwide, collecting a wealth of diverse data. Traditional computing models collect in-field data and then transmit it to a central data center where analytics are applied to it, but this is no longer a sustainable model. New approaches and new technologies are required to transform enormous amounts of collected data into meaningful information. Technology also will enable the interconnection around things in the IoT ecosystem but further research is required in the development, convergence and interoperability of the different IoT elements.

In this section, discussed are various implementations of Internet of Things done by different authors**.**

### Technical papers

- In the paper **"A Survey of Technologies for the Internet of Things"** presented by Vangelis Gazis and company. They provided a picture of the main technological components needed to enable the interconnection among things in order to realize IoT concepts and applications
- In the paper "**Design and Implementation of a WiFi Based Home Automation System"** presented by Ahmed ElShafee and Karim Alaa Hamed , they described that the home automation system that uses WiFi technology contains three main components; web server, which presents system core that controls, and monitors users' home and hardware interface module(Arduino PCB (ready-made), Wi-Fi shield PCB, 3 input alarms PCB, and 3 output actuators PCB.), which provides appropriate interface to sensors and actuator of home automation system. They studied that the system is better from the scalability and flexibility point of view than the commercially available home automation systems. The User may use the same technology to login to the server web based application. If server is connected to the internet, so remote users can access server web based application through the internet using compatible web browser.
- In the paper **"Home Automation Using Internet of Things"** by Vinay Sagar KN and Kusuma SM. They presented a Home Automation system(HAS) using Intel Galileo that employs the integration of cloud networking, wireless communication, to provide the user with remote control of various lights, fans, and appliances within their home and storing the data in the cloud. The system will automatically change based on sensors' data. They designed the system to be low cost and expandable allowing a variety of devices to be controlled.

# Chapter 3

## Project Requirements

### 3.1 Technology and resource requirements

As discussed earlier, any IoT device needs two main parts i.e.: computational power and a network connection.

The following hardware components are required for setting up a home automation system:

- Relay Board
- Particle Photon(microcontroller)
- Raspberry Pi

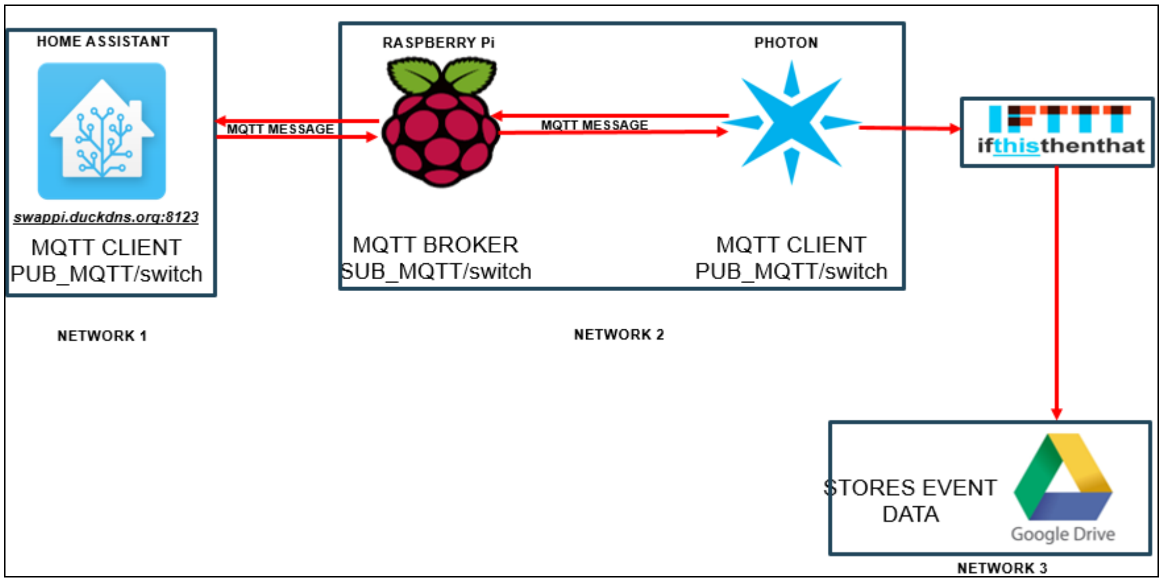
Along with this hardware the following things are needed to build the home automations system:

- Mosquito broker on Raspberry Pi
- Home automation on Raspberry Pi
- Putty
- Samba on Raspberry Pi
- DuckDNS on Raspberry Pi
- Account on build.particle.io for IDE
- Remot3.it

# Chapter 4
# System Design

## 4.1 Architecture design

The project implementation is shown below in the form of block diagrams



**Home Assistant**: It is our Graphical User Interface which is on a remote network. It acts like a client.

**Raspberry Pi**: It is our MQTT broker which mediates between the clients.

**Photon**: It is a microcontroller which is connected to a relay. It acts like a client. The photon and raspberry pi are on the same network.

**IFTTT**: If This Then That, is a functionality which triggers a chain of events when a particular event occurs. In our case when here is exchange of data between the clients, the activity log in google drive is updated.

The home assistant and photon are published/subscribed to the topic MQTT/switch

# Chapter 5
# Project Plan and Schedule

## 5.1 Project Team

Our team comprised of 4 members:

Aashav Panchal-011463361
Mohammed Abdul Mujeeb Ansari-011168144
Snehal Chavan-011452571
Swapnasheel Sonkamble-011452610

The team was divided into 2 groups and the work was distributed accordingly:

**TEAM 1** (Aashav & Swapnasheel):
The work for team 1 was based on Raspberry Pi as the broker, which included the following task's:

- Setting up Pi as MQTT Broker
- Configuration of Home Assistance
- Port forwarding & DNS

**TEAM 2** (Snehal & Mujeeb):
The work for team 2 was based on Photon and Relay which included the following tasks:

- Setting up photon as client
- Configuration of relay with photon
- Hardware setup

## 5.2 Project Tasks and Schedule

| SR NO. | WORK | WEEK | TEAM |
|---|---|---|---|
| 1. | Overview & Discussion of topic | 1 | 1 & 2 |
| 2. | Finalizing Components required. | 1 | 1 & 2 |
| 3. | Preparing Abstract | 2 | 1 & 2 |
| 4. | Installing OS in the Rasberry PI. | 2 | 1 |
| 5. | Configuring the Rasberry Pi as broker. | 3 | 1 |
| 6. | Configuring Photon as client. | 3-4 | 2 |
| 7. | Interfacing Photon with Relay. | 5 | 2 |
| 8. | Communicating in Local Network. | 6 | 1 & 2 |
| 9. | Port forwarding. | 7 | 1 |
| 10. | Setting Duck-dns. | 7 | 1 |
| 11. | Home assistant | 8 | 1 |
| 12. | Communication in different network | 8 | 1 & 2 |
| 13. | IFTTT | 8 | 2 |

# Chapter 6
# Tools and Standards

## 6.1. Tools Used

**Hardware tools:**

**Rasberry pi-3.**

- Raspberry Pi is used as a Broker, which is a central server that keeps track of who wants to hear what and sends messages to clients accordingly.

**Particle Photon.**

- Photon is used as client, which is connected to Broker (Rasberry-Pi) .Client, is subscribed to a particular topic. Depending upon subscribed topic the actions are performed by the Client.

**Relays.**

- Relays are connected to the Photon which could be used to control any device connected to it through the home assistant via raspberry pi.

**Software Tools:**

**Home Assistant.**

- Home Assistant is an open-source home automation platform running on Python 3.
- Home Assistant will track the state of all devices (LED's) and controls all your devices from a single, mobile friendly, interface.
- Home Assistant allows controlling all devices without storing any of your data in the cloud.

**Duck-Dns**

- Duck-dns is free service, which points a DNS (subdomains of duck-dns.org) to an IP of your Choice.
- DDNS is handy way for you to refer to server/router with an easily rememberable name, where servers IP address is likely to change.

**Putty**

- Putty is open source tool, which used for Telnet and SSH.

**WINSCP /SAMBA**

- WINSCP/SAMBA is used for secure file transfer between a Broker and a remote Computer.

## 6.2. Standards

**RASBERRY PI:**

**Mosquitto MQTT broker**
MQTT broker is the heart of any publish/subscribe protocol. An MQTT broker can handle up to thousands of concurrently connected MQTT clients.
Types of MQTT brokers:-

1) Cloud based/ Public Brokers
    - HiveMQ
    - Adafruit.io
    - IBM Bluemix
    - Iot.eclipse.org / dev.rabbitmq.com / www.cloudmqtt.com

2) Independent brokers
    - Personal home brokers, e.g Raspberry Pi

**Configuring MQTT on the Raspberry Pi 3**
MQTT as discussed above stands for Message Queue Telemetry Transport, which is a light weight message queue protocol designed for small data packets sent across high latency and low bandwidth links. MQTT is a simple protocol and a perfect for the Internet of Things.
MQTT Broker that we used in our project is “Mosquitto”, which is an open source broker developed by IBM.We configured Mosquitto broker in Raspberry Pi 3. It is available via ‘apt’, thus installing it is quite easy. We are using Raspberry Pi 3 running the latest version of Raspbian Jessie.

**Installing Mosquitto Components:-**
We need to run few apt commands. Make sure you are running the latest version of Raspbian Jessie and software is up to date.

- **wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key**
- **sudo apt-key add mosquitto-repo.gpg.key**

Then make the repository available to apt:

- **cd /etc/apt/sources.list.d/**

  **Installing Jessie repository**
- **sudo wget http://repo.mosquitto.org/debian/mosquitto-jessie.list**

To install the mqtt mosquitto for the raspberry pi follow the below steps use sudo before the command if not using root.
The below command is used to update the source list

- **sudo apt-get update**
- **apt-get install mosquitto**

The above command is to install mqtt mosquitto broker.

- **sudo apt-get install mosquitto mosquitto-clients**

The above command is to install mqtt mosquito client

**Communicating with the Broker:-**

1. **CUI (Command line interface):-**

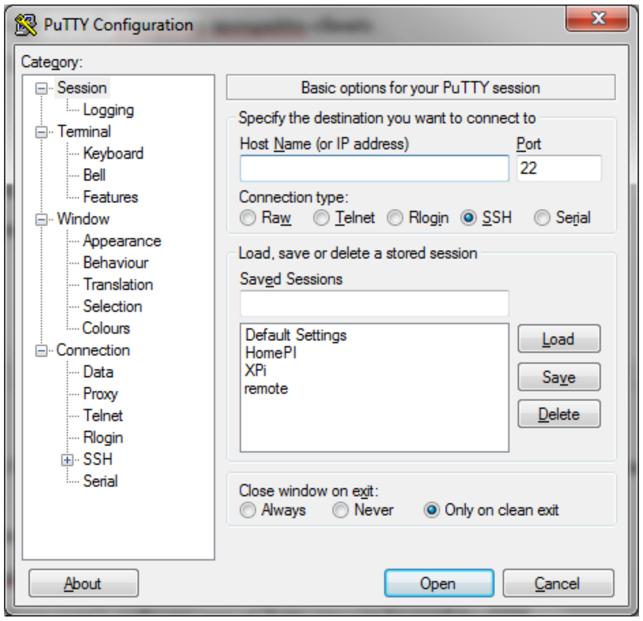
   This can be achieved by SSH (Secure Shell) and can be done using Terminal/ Bash in Linux or Mac devices.

   SSH command is as follows:-

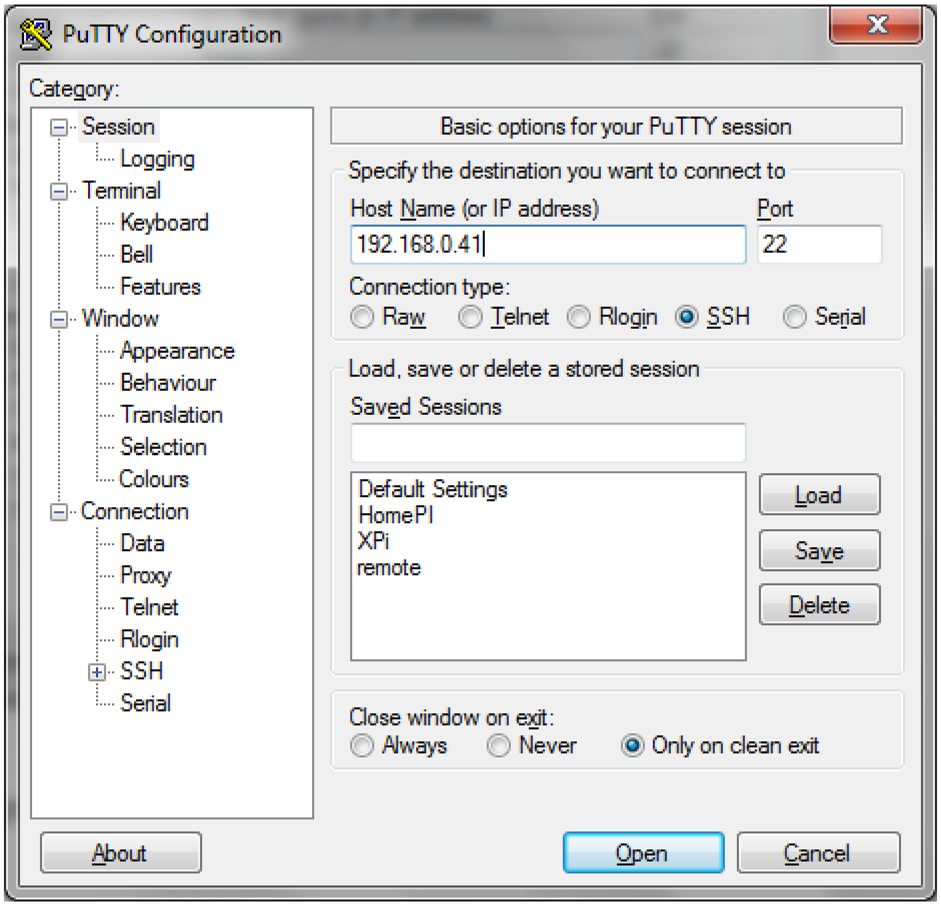
   **ssh –l {hostname} {host IP address**

   For Windows user's, software named Putty.exe can be used for SSH.

   Image putty is as follows:



Broker IP address must be entered in the Host name (or IP address) section.



Specify the destination you want to connect to

Host Name (or IP address): 192.168.0.41

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions

Default Settings
HomePI
XPi
remote

Close window on exit: Always Never [✓ Only on clean exit]

**Testing your MQTT Broker:-**

**MQTT Server and Client Communication:-**

To Subcribe :- mosquito_sub –t “test/topic”

To Publish :- mosquitto_pub –t “test/topic” –m “Hello”

**GUI (Graphical User Interface):-**

Graphical user interface allows users to interact with electronic devices through icons and visual indicators such as setting date and time without text-based user interface. For first time users, setting up a WiFi connection is easy using a GUI instead of a CUI. Once the WiFi is connected to the system, automatic connection can be established as soon as the device is powered on.

GUI interface also gives a much better flexibility in searching files and navigating through the Operating Systems for a user who has not used CUI very often.

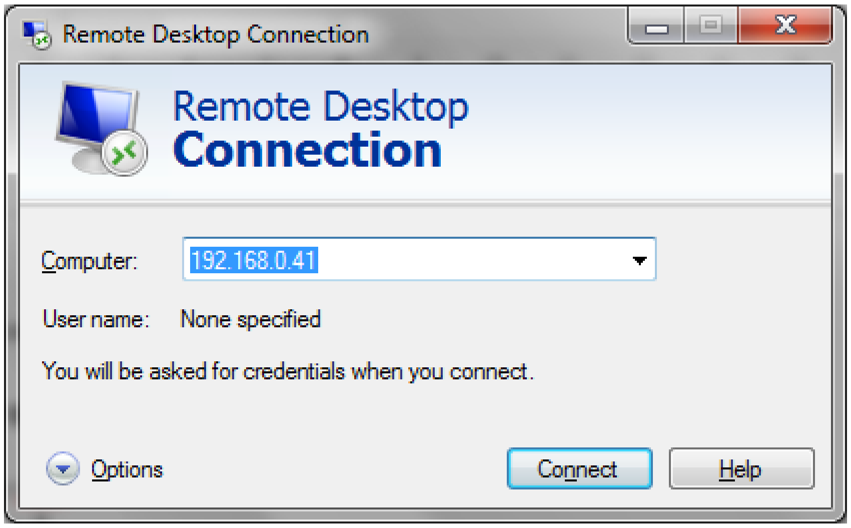
**Configuring GUI in the Broker:-**

In order to achieve Graphical user interface in the broker, we implemented two methods as described below.

1) Remote Desktop Connection.
2) VNC Server and Client.

**Remote Desktop Connection:-**

Windows is equipped with an application of remote desktop login. To access remote desktop from windows, one might search it in the start menu and locate it easily.


Remote Desktop Connection

Remote Desktop Connection

Computer: 192.168.0.41

User name: None specified

You will be asked for credentials when you connect.

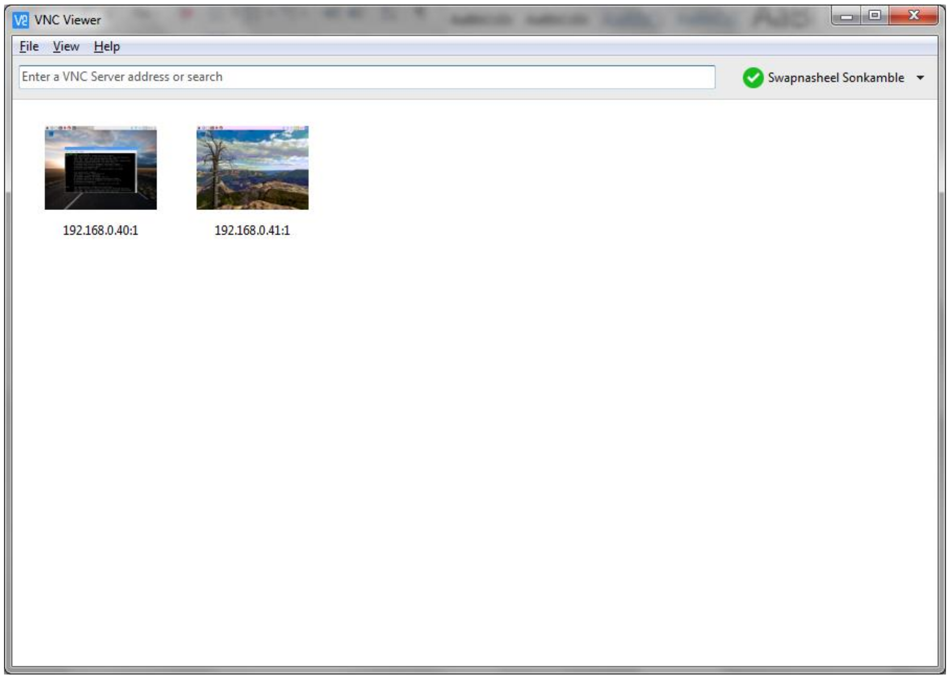
In order to access the broker using remote desktop connection service, we need to install “xrdp” (An open source remote desktop protocol (rdp) server).

Following commands will help use install required package:-

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install xrdp
```

**Tight VNC:-**

TightVNC is a free remote control software package. With TightVNC, you can see the desktop of a remote machine and control it with your local mouse and keyboard, just like you would do it sitting in the front of that computer.
In case of ‘xrdp’, a new session is initialized, thus making it different from the primary session. Whereas, VNC eliminates creation of new session but instead runs in the main/ primary session only.
A VNC client must be installed on the client machine via which you can log into the VNC server; in our case, the broker.



**sudo apt-get update**
**sudo apt-get upgrade**
**sudo apt-get tightvncserver -** get VNC server in your Pi (which will act as a server)
setting up vnc server in Pi
**vncserver :1**
set up a password
eg. password
Make vnc run at startup
**sudo nano /etc/rc.local**
before exit 0 ...,enter :-
**su - pi -c '/user/bin/vncserver :1'**
Save and exit (^X and Y)

**Accessing Broker from anywhere:-**

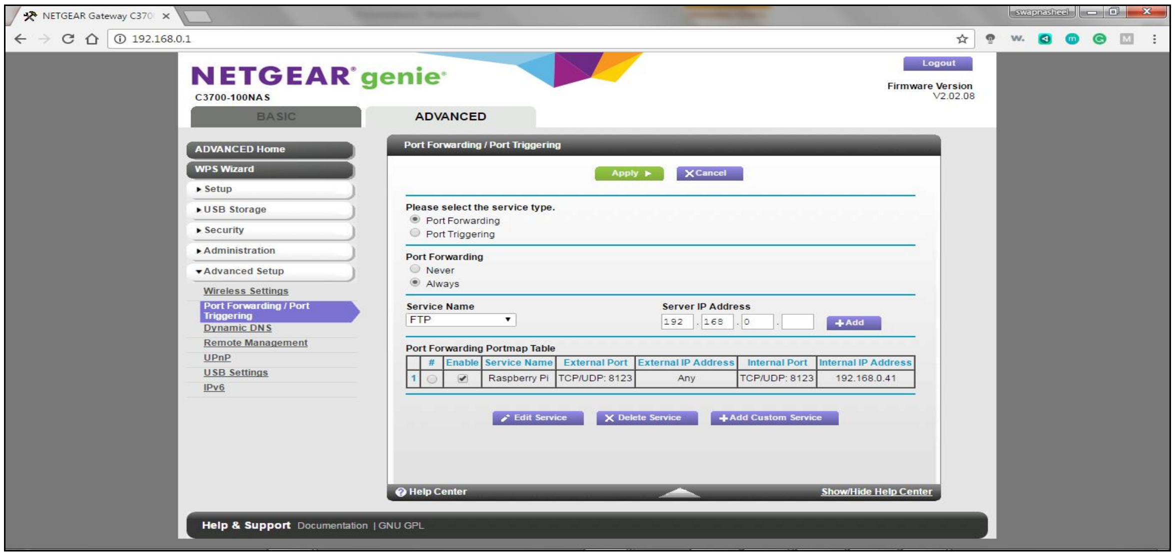
You can access the MQTT broker and client setup from any device on local network or same LAN subnet. What if we wish to access our broker and update changes form a different network than a home network, say Work network?
To achieve world-wide broadband access, we setup Port forwarding in home router.

**Port Forwarding:-**

In computer networking, **port forwarding** or **port** mapping is an application of network address translation (NAT) that redirects a communication request from one address and **port** number combination to another while the packets are traversing a network gateway, such as a router or firewall.

In a home network, there are usually more than one devices connected to the local network. Using the “Host Name” information we can filter out the Raspberry Pi and trace its IP address assigned by the DNS server. Therefore we can setup all the communication requests coming on one port forwarded to the brokers IP address.

**Setup:-**



Port Forwarding / Port Triggering

Please select the service type.
[✓ Port Forwarding]
Port Triggering

Port Forwarding
Never
[✓ Always]

Service Name: FTP
Server IP Address: 192 . 168 . 0 .

Port Forwarding Portmap Table

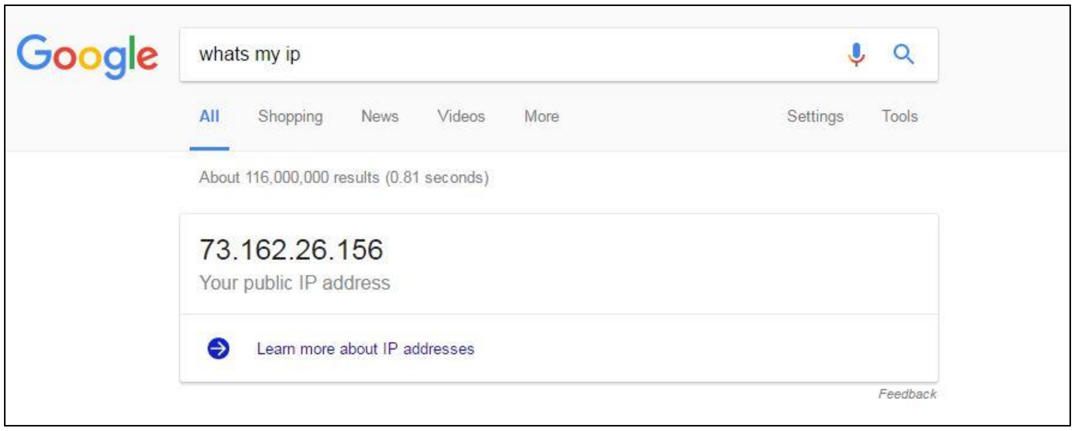
| | # | Enable | Service Name | External Port | External IP Address | Internal Port | Internal IP Address |
|---|---|---|---|---|---|---|---|
| 1 | ○ | ✓ | Raspberry Pi | TCP/UDP: 8123 | Any | TCP/UDP: 8123 | 192.168.0.41 |

Port **8123** is selected in the router.

Any communication information coming to that port is directed to IP address **192.168.0.41** (which is the IP address of the broker)

Start port and End port = 8123

Accessing the broker from anywhere becomes easy. Now we need to know our networks IP address, this can be found out by searching for our IP address on www.google.com as shown below.



whats my ip

About 116,000,000 results (0.81 seconds)

73.162.26.156
Your public IP address

http://73.162.26.156 :8126. Will take you to the Broker.

**Domain name services (DDNS):-**

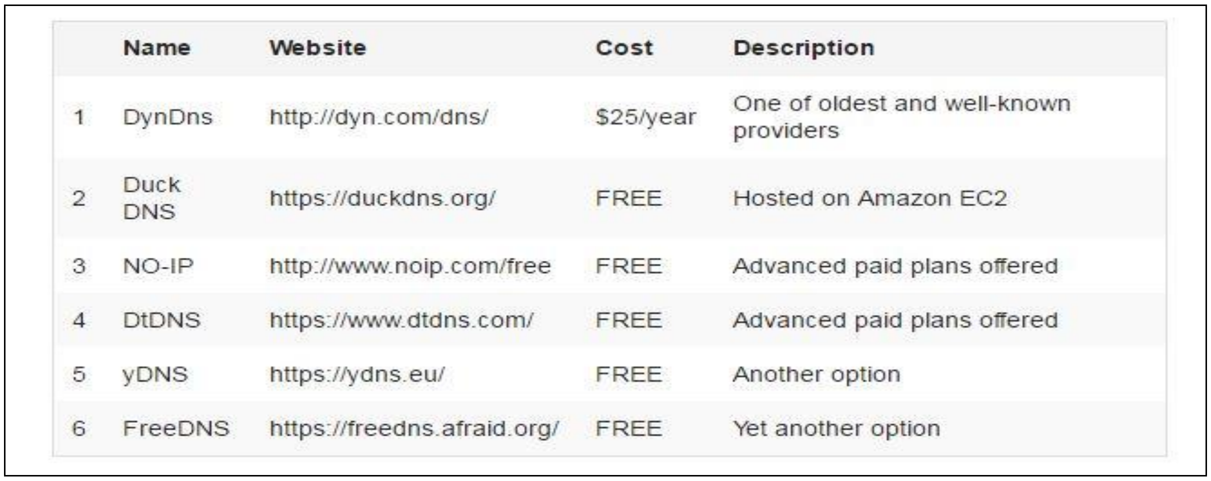
The Internet's system for converting alphabetic names into numeric IP addresses. For example, when a Web address (URL) is typed into a browser, DNS servers return the IP address of the Web server associated with that name.

Problems with ISP IP address:-

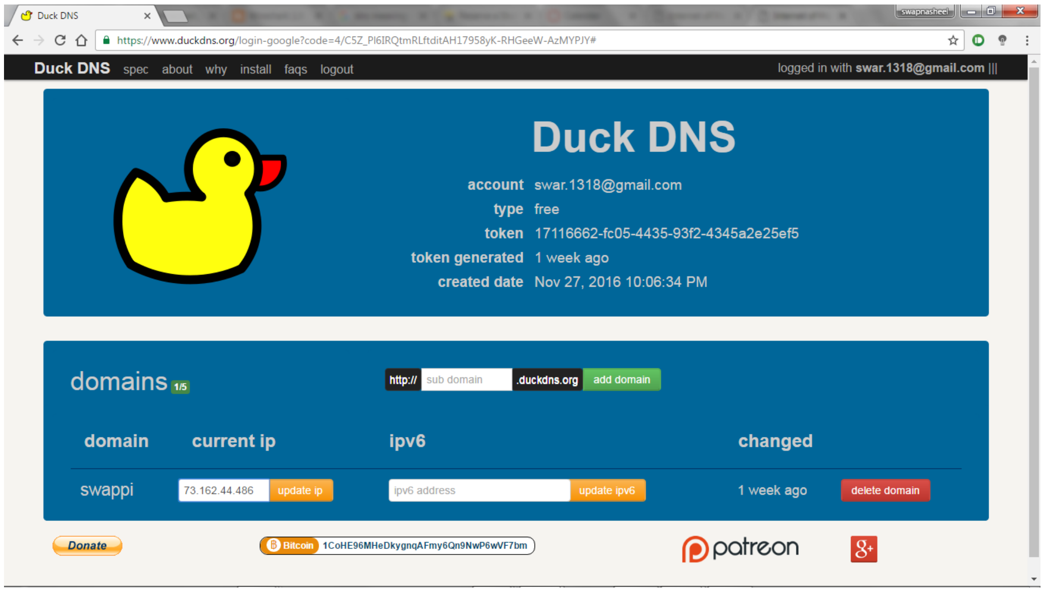
ISP may change the IP address at any time, and if you are away from the home network, it will be difficult to determine the updated address. So you become cut off from your home network.

To resolve this issue, a DDNS (Dynamic domain name server) account is needed. A DNS simple resolves a domain host name to an IP address. Using a dynamic DNS updates the IP address that the domain name points to whenever it changes automatically.

**DDNS Providers:-**



| | Name | Website | Cost | Description |
|---|---|---|---|---|
| 1 | DynDns | http://dyn.com/dns/ | $25/year | One of oldest and well-known providers |
| 2 | Duck DNS | https://duckdns.org/ | FREE | Hosted on Amazon EC2 |
| 3 | NO-IP | http://www.noip.com/free | FREE | Advanced paid plans offered |
| 4 | DtDNS | https://www.dtdns.com/ | FREE | Advanced paid plans offered |
| 5 | yDNS | https://ydns.eu/ | FREE | Another option |
| 6 | FreeDNS | https://freedns.afraid.org/ | FREE | Yet another option |


Duck DNS spec about why install faqs logout logged in with swar.1318@gmail.com

Duck DNS

| | |
|---|---|
| account | swar.1318@gmail.com |
| type | free |
| token | 17116662-fc05-4435-93f2-4345a2e25ef5 |
| token generated | 1 week ago |
| created date | Nov 27, 2016 10:06:34 PM |

domains 1/5 http:// sub domain .duckdns.org add domain

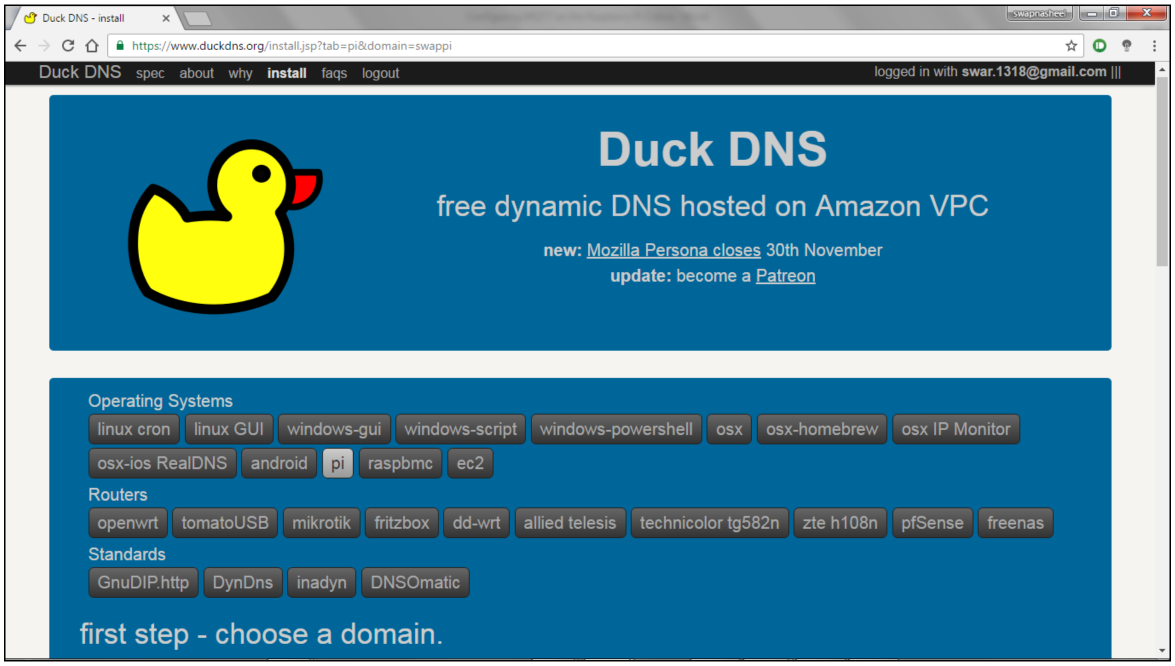
| domain | current ip | ipv6 | changed |
|---|---|---|---|
| swappi | 73.162.44.486 update ip | ipv6 address update ipv6 | 1 week ago delete domain |

Donate Bitcoin 1CoHE96MHeDkygnqAFmy6Qn9NwP6wVF7bm patreon

**Installing DNS in Raspberry PI:-**

Following steps were taken in setting up DNS in the broker:-

Select **PI** from the '**install**' section, as shown below:


Duck DNS spec about why install faqs logout logged in with swar.1318@gmail.com

Duck DNS

free dynamic DNS hosted on Amazon VPC

**new:** Mozilla Persona closes 30th November

**update:** become a Patreon

Operating Systems

linux cron | linux GUI | windows-gui | windows-script | windows-powershell | osx | osx-homebrew | osx IP Monitor | osx-ios RealDNS | android | pi | raspbmc | ec2

Routers

openwrt | tomatoUSB | mikrotik | fritzbox | dd-wrt | allied telesis | technicolor tg582n | zte h108n | pfSense | freenas

Standards

GnuDIP.http | DynDns | inadyn | DNSOmatic

first step - choose a domain.

**Steps:-**

1. Choose domain you wish to set DNS for
2. Login to your raspberry Pi using Putty / Terminal
3. mkdir duckdns
4. cd duckdns
5. vi duck.sh
6. paste by pressing “i” and then “right click” echo url="https://www.duckdns.org/update?domains=example2&token=abcdefg123123123123&ip=" | curl -k -o ~/duckdns/duck.log -K -
7. press -> ESC then :wq! then ENTER
8. chmod 700 duck.sh
9. crontab –e
10. select ‘nano’ and paste this line at the bottom of the script */5 * * * * ~/duckdns/duck.sh >/dev/null 2>&1
11. save and exit
12. ./duck.sh
13. cat duck.log, should return OK
14. sudo service cron start
15. cd

## PARTICLE PHOTON:

Particle's Internet of Things hardware development kit, the Photon, provides everything you need to build a connected product. Photon is a microcontroller specially designed for IoT as it provides computational power through its processor as well as network connection by the Wi-Fi module. The Wi-Fi module allows Photon to communicate with the internet. It connects your device to the internet in the same way that your smartphone might connect to a wifi network.

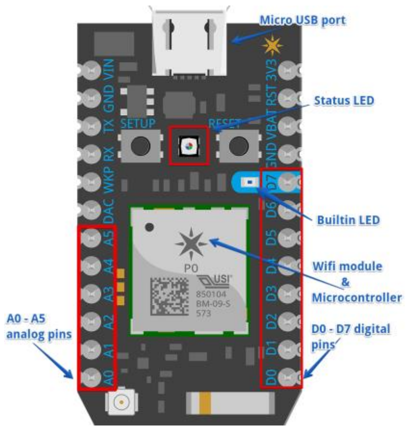

Micro USB port
Status LED
Builtin LED
Wifi module & Microcontroller
D0 - D7 digital pins
A0 - A5 analog pins
SETUP
RESET
VIN GND TX RX WKP DAC A5 A4 A3 A2 A1 A0
3V3 RST VBAT GND D7 D6 D5 D4 D3 D2 D1 D0
P0
USI
850104 BM-09-S 573

**Fig. Spark Photon**

**Features:**

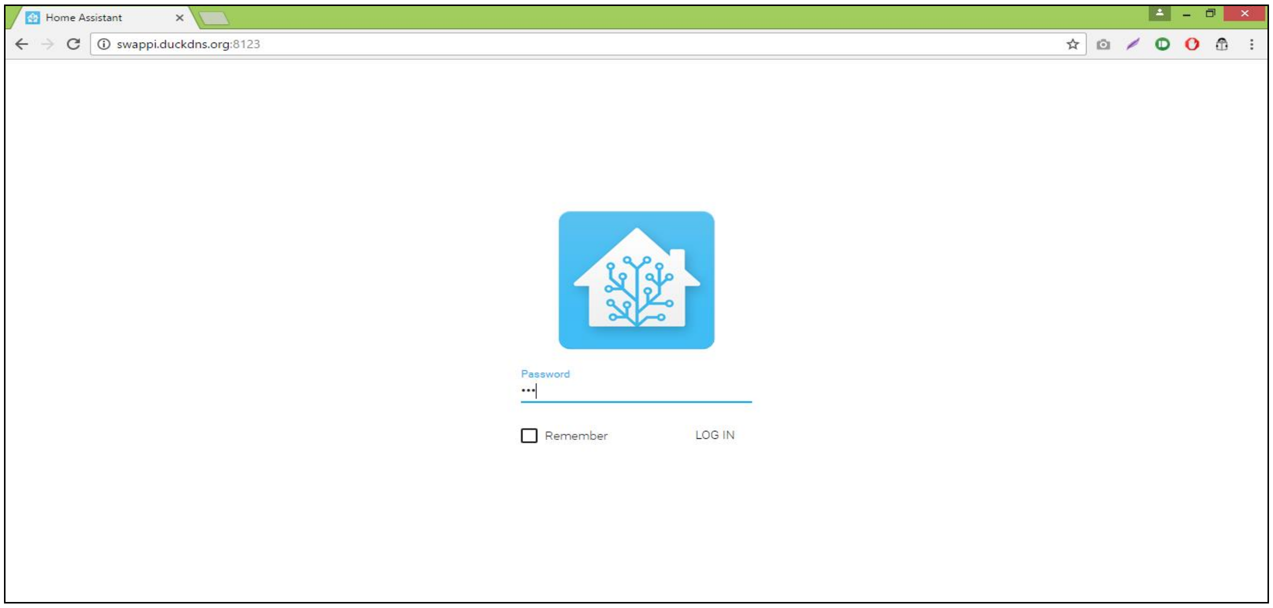
- PARTICLE PØ WI-FI MODULE
    - BROADCOM BCM43362 WI-FI CHIP.
    - 802.11B/G/N WI-FI.
    - STM32F205 120MHZ ARM CORTEX M3.
    - 1MB FLASH, 128KB RAM.
- ON-BOARD RGB STATUS LED (EXT. DRIVE PROVIDED)
- 18 MIXED-SIGNAL GPIO AND ADVANCED PERIPHERALS
- REAL-TIME OPERATING SYSTEM (FREERTOS)
- SOFT AP SETUP
- FCC, CE AND IC CERTIFIED

## HOME ASSISTANT (HA):

Home assistant is an open-source home automation platform running on Python 3. We can control, monitor, access all connected devise form a single, mobile-friendly interface.

Home assistant allows you to keep your data private without saving anything on the cloud.

Password protected HA looks something like this:-

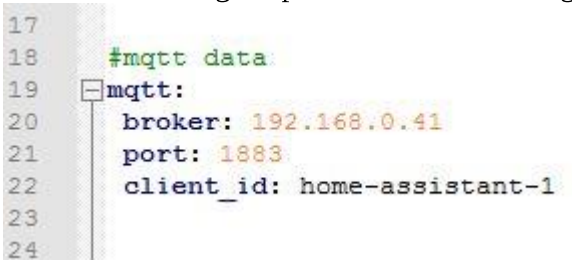


### Configuring Home Assistant:-

Home assistant is your personal User interface which you can configure according to your needs. It has tiles format which can be updated, changed or even automated by editing the "configuration.yaml" file.

- Creating MQTT service in the Home Assistant:

  Following snapshot shows the configuration in the .yaml file to add MQTT service.

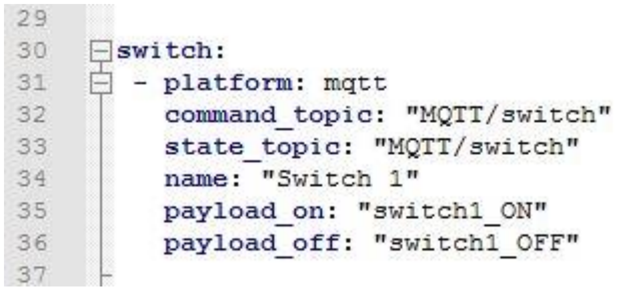


```
#mqtt data
mqtt:
  broker: 192.168.0.41
  port: 1883
  client_id: home-assistant-1
```

Mqtt – defines the service type
Broker – defines the broker address

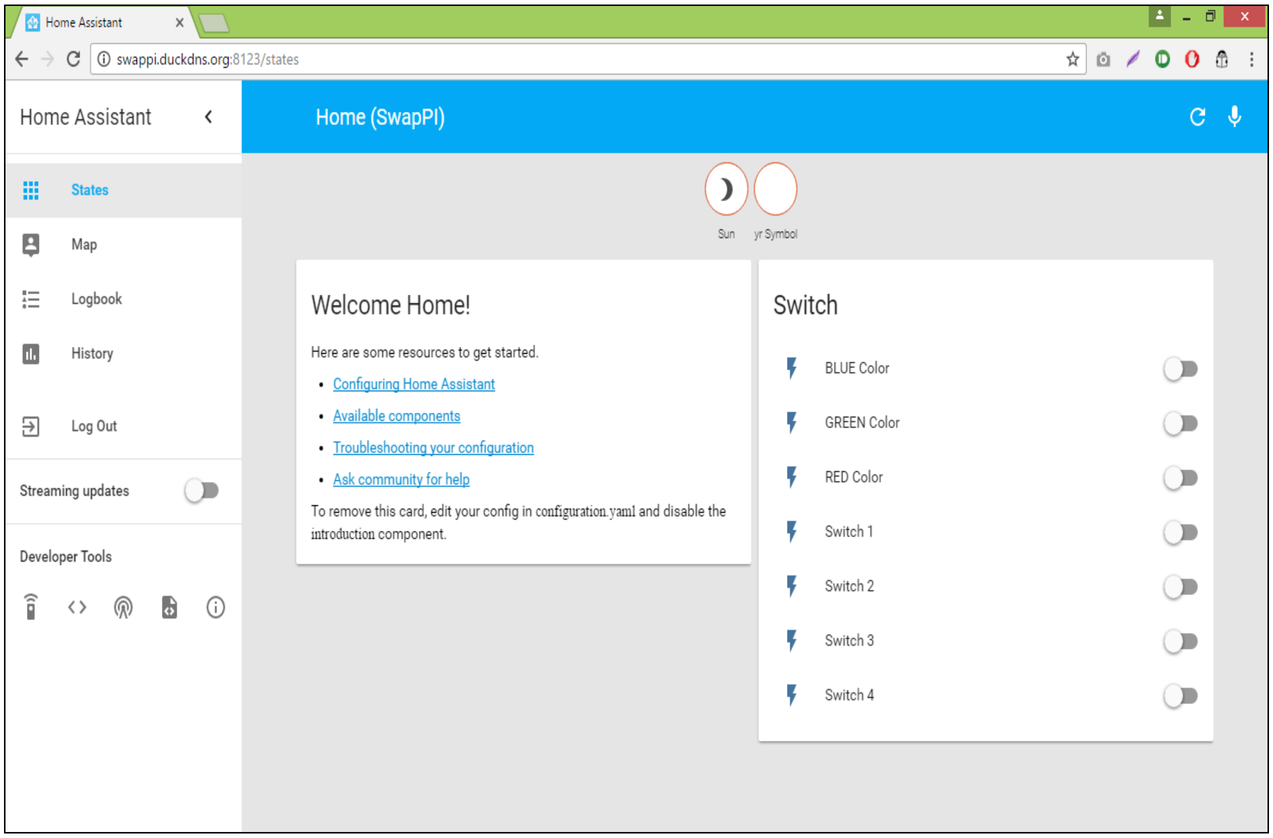
Port – defines the MQTT port number

- Creating Switch in Home assistant:-



```
switch:
  - platform: mqtt
    command_topic: "MQTT/switch"
    state_topic: "MQTT/switch"
    name: "Switch 1"
    payload_on: "switch1_ON"
    payload_off: "switch1_OFF"
```

Switch – Service
Platform – Defines the type of protocol
Command_topic - Defines the Published topic
State-topic – Defines the Subscribed topic
Payload – Data to be sent/ received



Welcome Home!

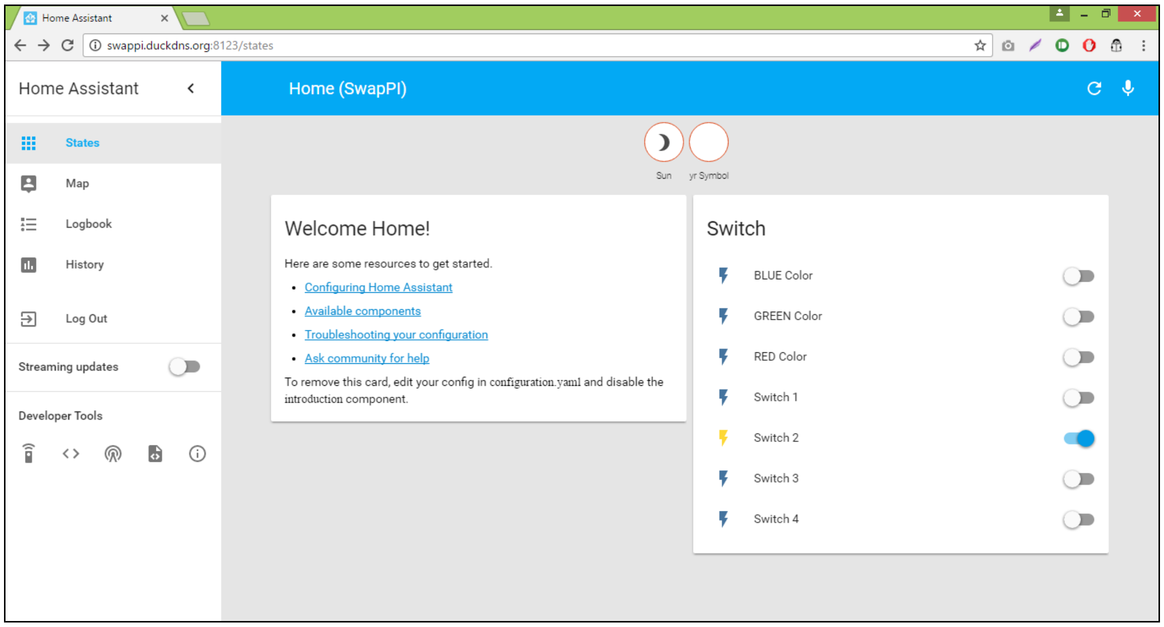
Here are some resources to get started.

- Configuring Home Assistant
- Available components
- Troubleshooting your configuration
- Ask community for help

To remove this card, edit your config in configuration.yaml and disable the introduction component.

Switch

BLUE Color
GREEN Color
RED Color
Switch 1
Switch 2
Switch 3
Switch 4



Streaming updates: off

Welcome Home!

Here are some resources to get started.

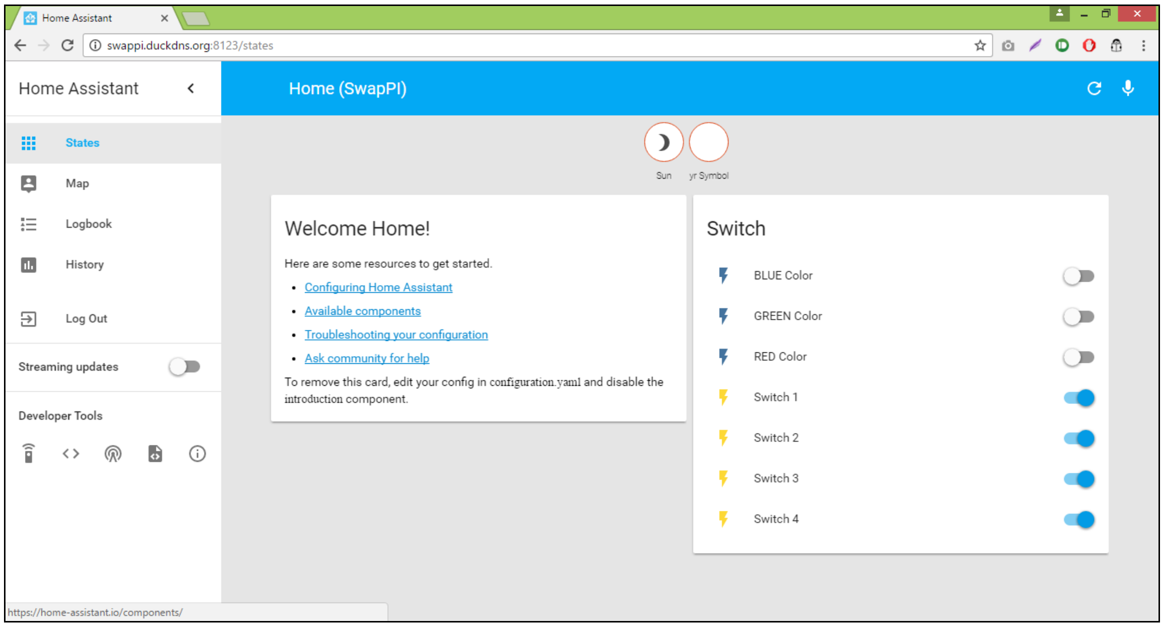
- Configuring Home Assistant
- Available components
- Troubleshooting your configuration
- Ask community for help

To remove this card, edit your config in configuration.yaml and disable the introduction component.

Switch

| | |
|---|---|
| BLUE Color | off |
| GREEN Color | off |
| RED Color | off |
| Switch 1 | off |
| Switch 2 | on |
| Switch 3 | off |
| Switch 4 | off |

**Switch1, Switch2 , Switch 3 and Switch 4 turned ON.**



Streaming updates: off

Welcome Home!

Here are some resources to get started.

- Configuring Home Assistant
- Available components
- Troubleshooting your configuration
- Ask community for help

To remove this card, edit your config in configuration.yaml and disable the introduction component.

Switch

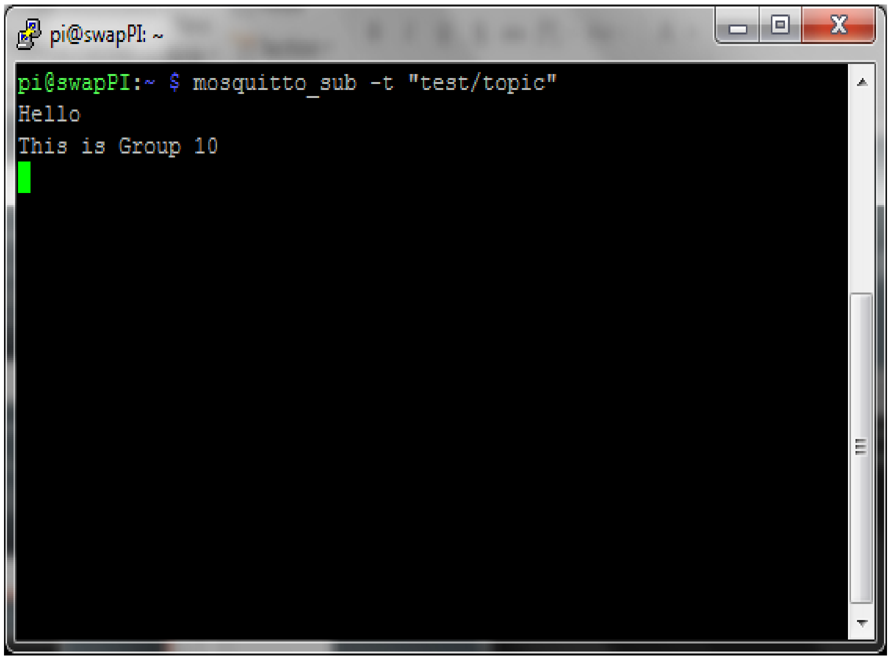
| | |
|---|---|
| BLUE Color | off |
| GREEN Color | off |
| RED Color | off |
| Switch 1 | on |
| Switch 2 | on |
| Switch 3 | on |
| Switch 4 | on |

# Chapter 7

## <u>Project progress and Status</u>

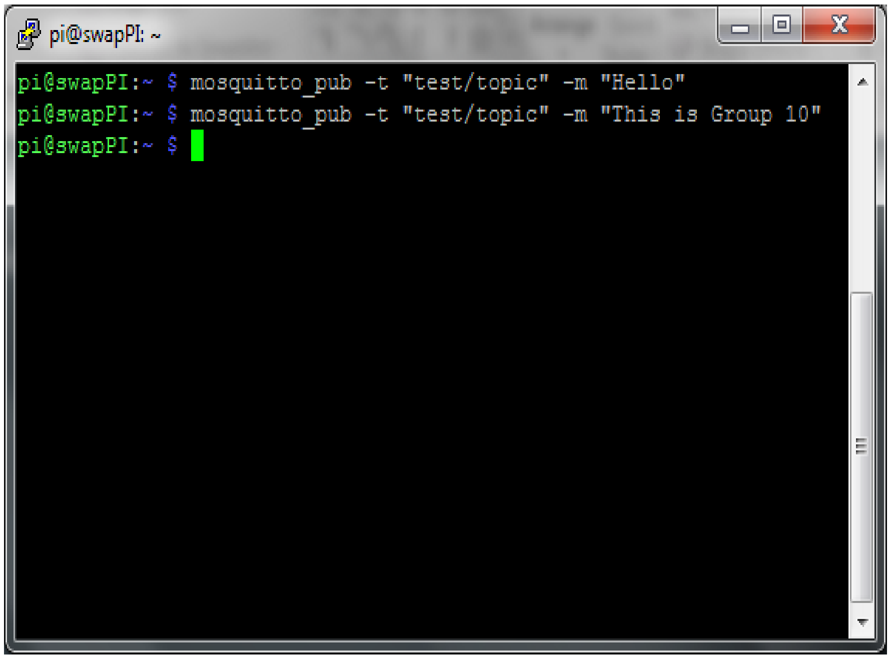
1. Installing OS in the Rasberry PI.

2. Configuring the Rasberry Pi as broker.



pi@swapPI: ~

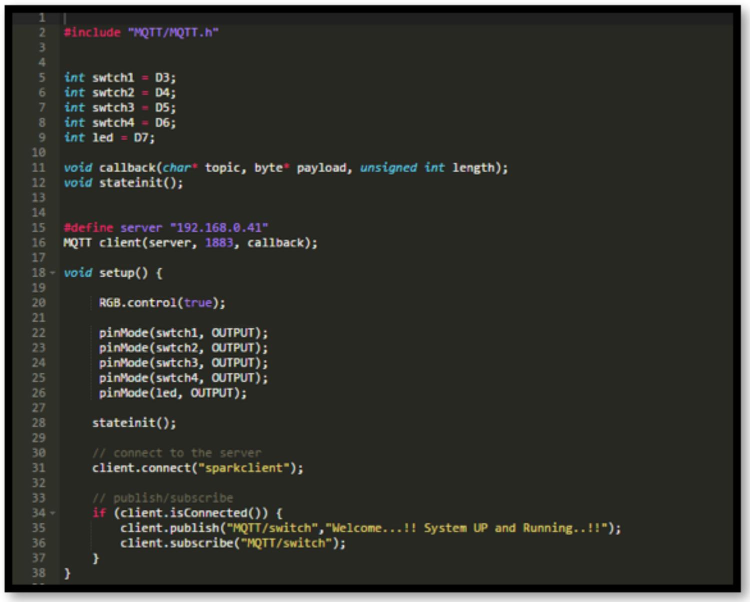
```
pi@swapPI:~ $ mosquitto_sub -t "test/topic"
Hello
This is Group 10
```



pi@swapPI: ~

```
pi@swapPI:~ $ mosquitto_pub -t "test/topic" -m "Hello"
pi@swapPI:~ $ mosquitto_pub -t "test/topic" -m "This is Group 10"
pi@swapPI:~ $
```

**3.** Configuring Photon as client.

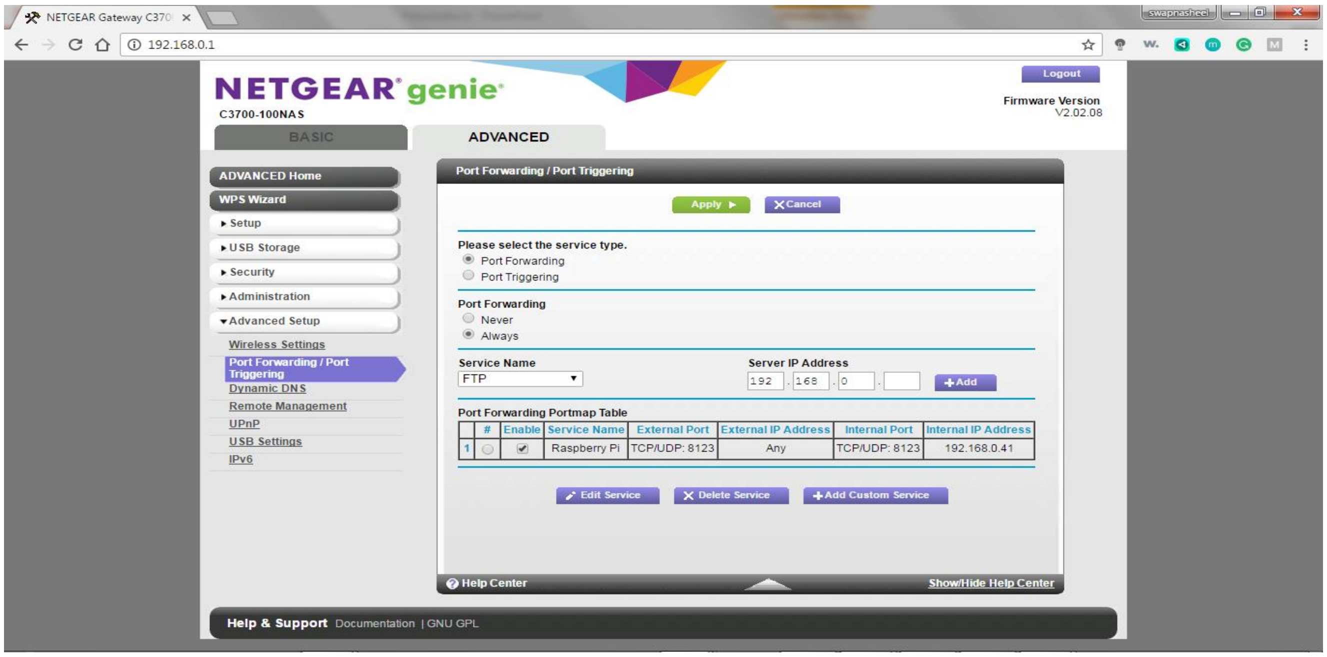


```
#include "MQTT/MQTT.h"


int swtch1 = D3;
int swtch2 = D4;
int swtch3 = D5;
int swtch4 = D6;
int led = D7;

void callback(char* topic, byte* payload, unsigned int length);
void stateinit();


#define server "192.168.0.41"
MQTT client(server, 1883, callback);

void setup() {

     RGB.control(true);

     pinMode(swtch1, OUTPUT);
     pinMode(swtch2, OUTPUT);
     pinMode(swtch3, OUTPUT);
     pinMode(swtch4, OUTPUT);
     pinMode(led, OUTPUT);

    stateinit();

    // connect to the server
    client.connect("sparkclient");

    // publish/subscribe
    if (client.isConnected()) {
        client.publish("MQTT/switch","Welcome...!! System UP and Running..!!");
        client.subscribe("MQTT/switch");
    }
}
```

**4.** Interfacing Photon with Relay.

**5.** Communicating in Local Network.

**6.** Portforwarding.



Firmware Version V2.02.08

Port Forwarding / Port Triggering

Please select the service type.
[✓ Port Forwarding]
Port Triggering

Port Forwarding
Never
[✓ Always]

Service Name: FTP

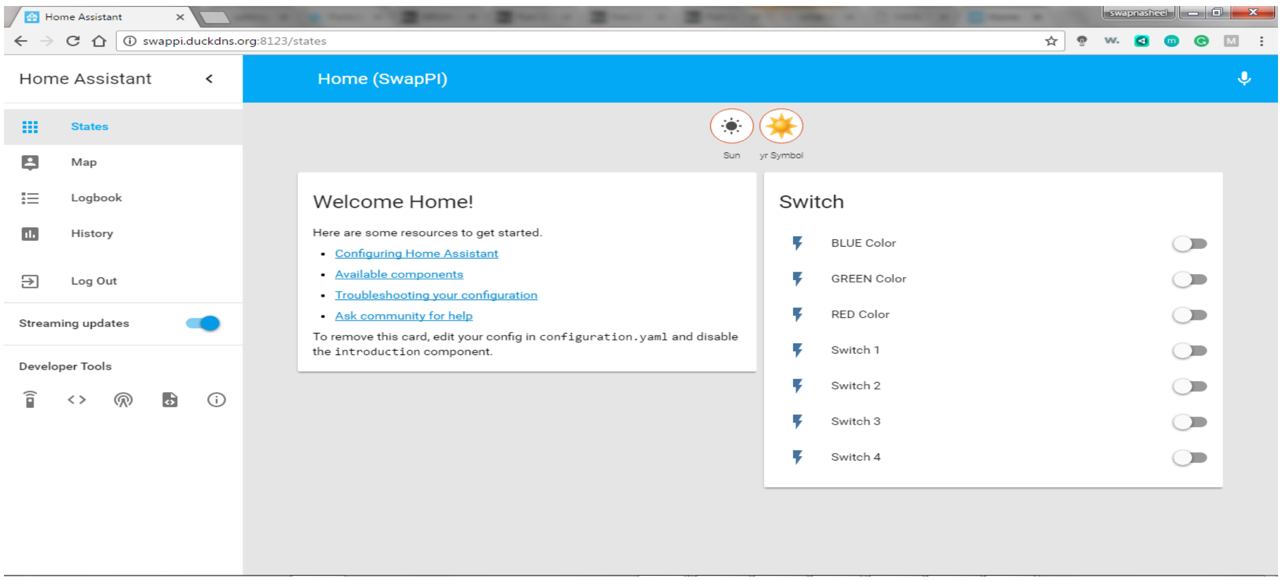
Server IP Address: 192 . 168 . 0 .

Port Forwarding Portmap Table

| | # | Enable | Service Name | External Port | External IP Address | Internal Port | Internal IP Address |
|---|---|---|---|---|---|---|---|
| 1 | ○ | ✓ | Raspberry Pi | TCP/UDP: 8123 | Any | TCP/UDP: 8123 | 192.168.0.41 |

**7.** Setting Duck-dns

**8.** Home assistant



Welcome Home!

Here are some resources to get started.

- Configuring Home Assistant
- Available components
- Troubleshooting your configuration
- Ask community for help

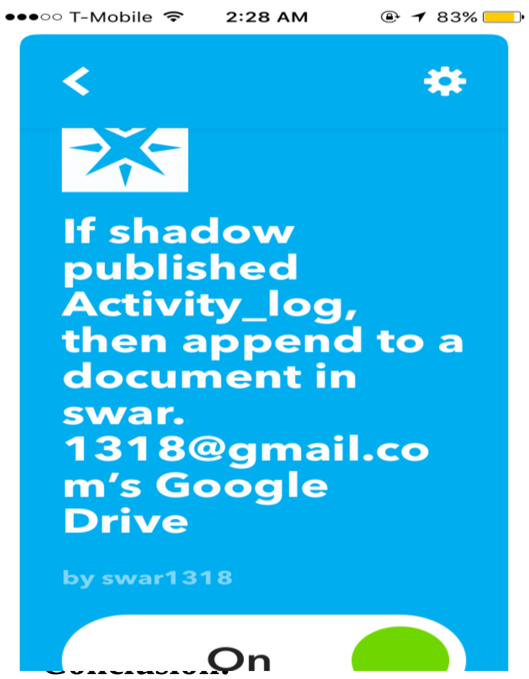
To remove this card, edit your config in `configuration.yaml` and disable the `introduction` component.

Switch

- BLUE Color
- GREEN Color
- RED Color
- Switch 1
- Switch 2
- Switch 3
- Switch 4

**9.** Communcation in different network

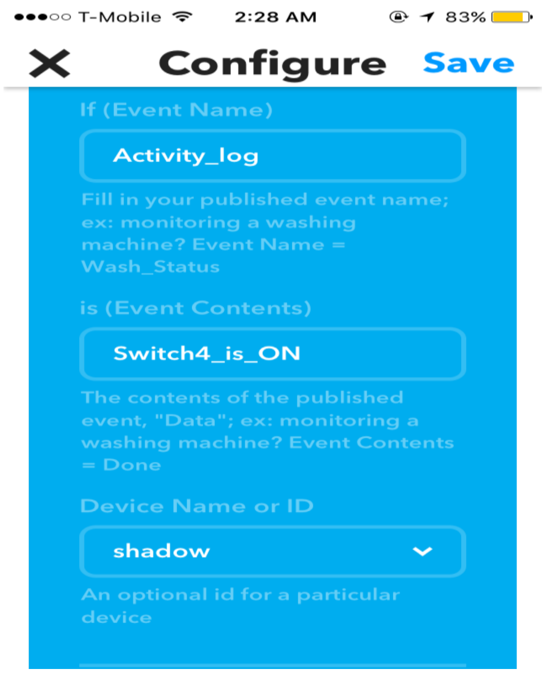
**10.** Configuring IFTTT



If shadow published Activity_log, then append to a document in swar.1318@gmail.com's Google Drive

by swar1318

On



If (Event Name)

Activity_log

Fill in your published event name; ex: monitoring a washing machine? Event Name = Wash_Status

is (Event Contents)

Switch4_is_ON

The contents of the published event, "Data"; ex: monitoring a washing machine? Event Contents = Done

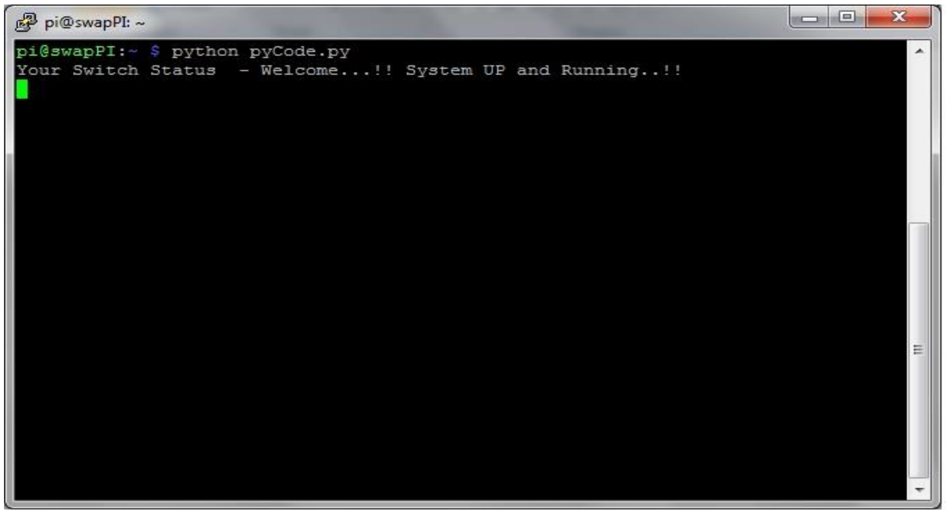
Device Name or ID

shadow

An optional id for a particular device

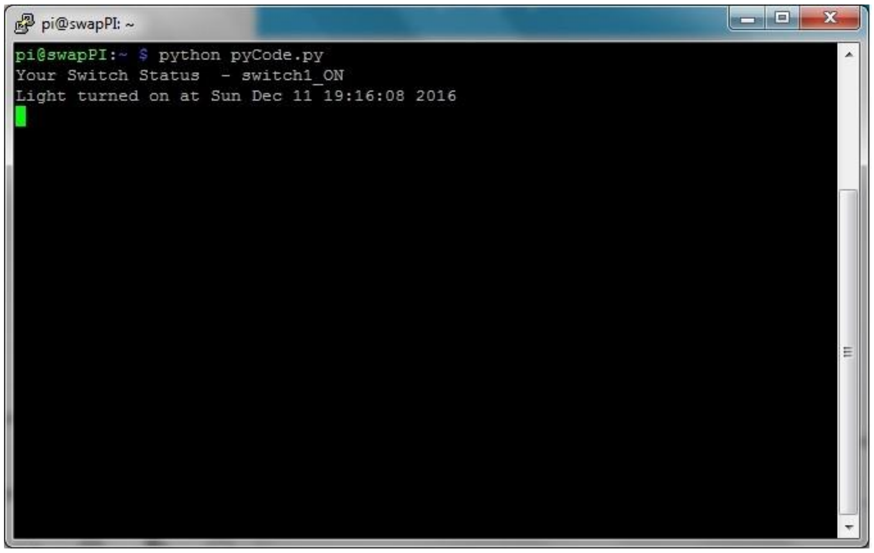
Conclusion:

The client is first connected to the broker and the connection is initiated from the client side, which is displayed as below:



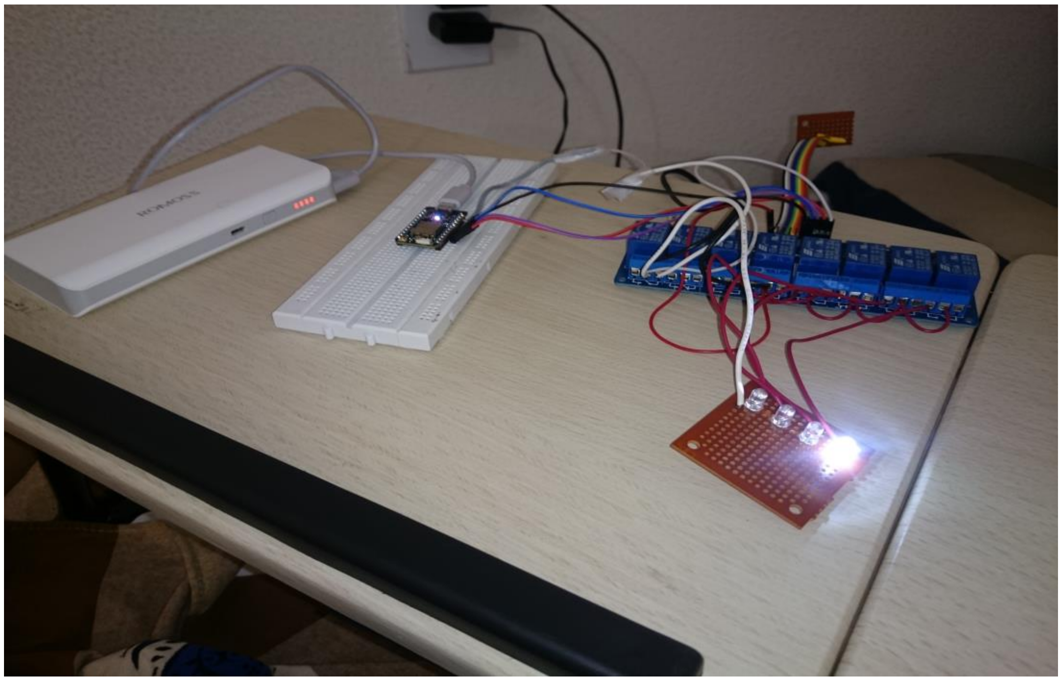
```
pi@swapPI:~ $ python pyCode.py
Your Switch Status  - Welcome...!! System UP and Running..!!
```

When the switch 1 is toggled from the home assistant page the message is sent from home assistant to the MQTT broker which is subscribed on the same topic as shown below:

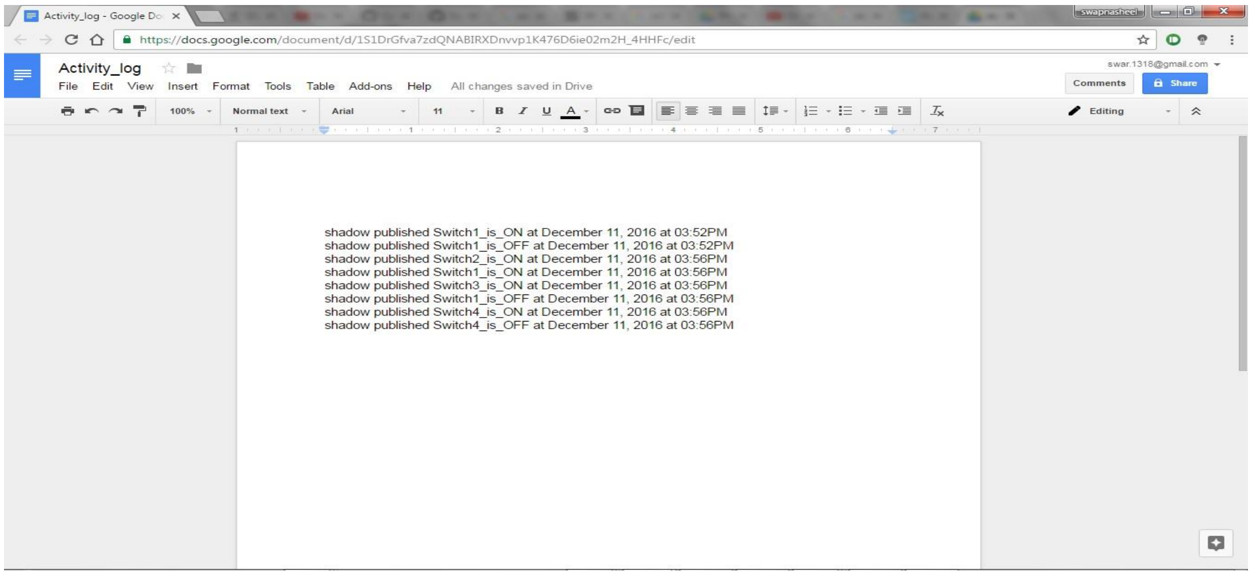


```
pi@swapPI:~ $ python pyCode.py
Your Switch Status  - switch1_ON
Light turned on at Sun Dec 11 19:16:08 2016
```

The same message is sent to the other client which is photon connected to the relay and the led connected to switch 1 is turned ON :

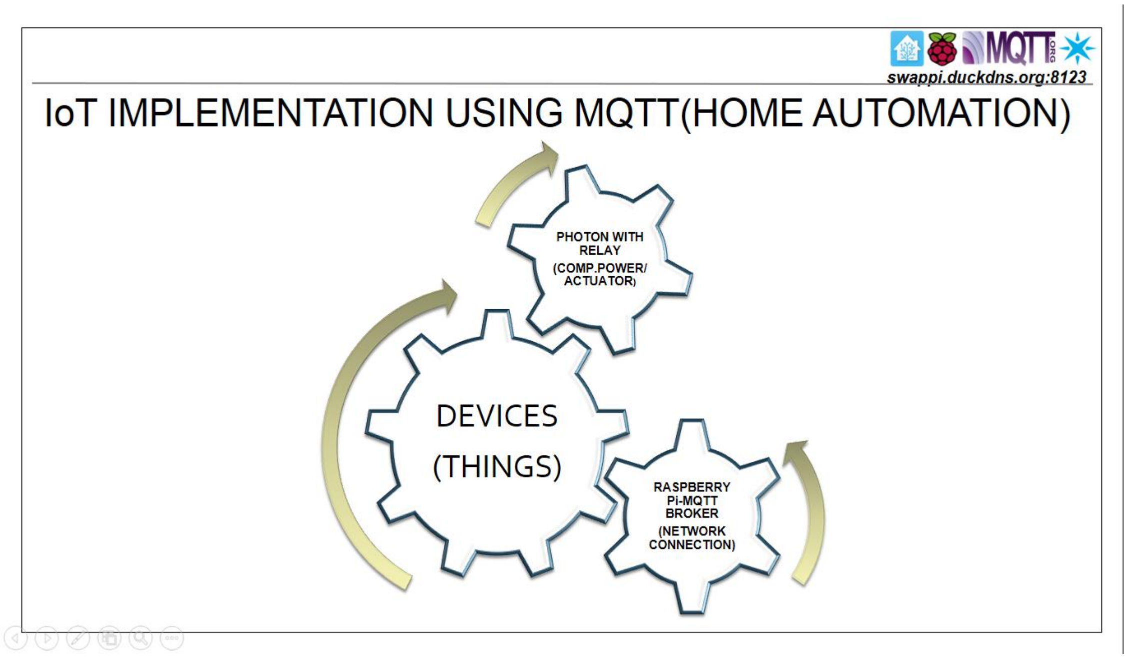


The data of tuning ON the led is then published on the google drive through IFTTT


Activity_log - Google Do
https://docs.google.com/document/d/1S1DrGfva7zdQNABIRXDnvvp1K476D6ie02m2H_4HHFc/edit
Activity_log
File Edit View Insert Format Tools Table Add-ons Help All changes saved in Drive
swar.1318@gmail.com
Comments
Share
100%
Normal text
Arial
11
Editing

```
shadow published Switch1_is_ON at December 11, 2016 at 03:52PM
shadow published Switch1_is_OFF at December 11, 2016 at 03:52PM
shadow published Switch2_is_ON at December 11, 2016 at 03:56PM
shadow published Switch1_is_ON at December 11, 2016 at 03:56PM
shadow published Switch3_is_ON at December 11, 2016 at 03:56PM
shadow published Switch1_is_OFF at December 11, 2016 at 03:56PM
shadow published Switch4_is_ON at December 11, 2016 at 03:56PM
shadow published Switch4_is_OFF at December 11, 2016 at 03:56PM
```

**The same thing is repeated for every toggle on Home Assistant.**


swappi.duckdns.org:8123

## IoT IMPLEMENTATION USING MQTT(HOME AUTOMATION)

PHOTON WITH RELAY (COMP.POWER/ ACTUATOR)
DEVICES (THINGS)
RASPBERRY Pi-MQTT BROKER (NETWORK CONNECTION)

**Thus we have taken home devices as a "Thing" which is provided with "computational power" using spark photon with relay and "network connection" by Raspberry Pi as an MQTT Broker to convert it to an IoT.**

# References

1. **“A Survey of Technologies for the Internet of Things”** presented by Vangelis Gazis and company.
2. “**Design and Implementation of a WiFi Based Home Automation System”** presented by Ahmed ElShafee and Karim Alaa Hamed
3. **“Home Automation Using Internet of Things”** by Vinay Sagar KN and Kusuma SM.
4. **“Building smarter planet solution with MQTT and IBM websphere MQ telemetry”** by IBM.
5. **Home Assistant**
6. https://www.youtube.com/channel/UCLecVrux63S6aYiErxdiy4w/videos
7. http://www.bruhautomation.com/