PiHue Project

Yan Yu

Project Github Website: https://github.com/bigbosstony/bigbosstony.github.io

Mar 28th, 2017

Declaration of Sole Authorship

I confirm that that this work submitted for assessment is my own and is expressed in my own words. Any uses that was used in this documents that came from any other authors were acknowledged where their works were used. A list of references are included in this document.

Yan Yu

Date: February 7, 2017

Previous Project Description

During the fifth semester me and my group members have been working on a project called Smart Parking Lot. The full project would has the functions to taking pictures of vehicles' licenses plates and then open the gate or close the gate let the vehicle enter or exit automatically. However, due to the cancellation of the project in this semester, I started on my own project by myself. You can check out my previous project on this website https://bigbosstony.github.io.

Proposal

Jan 30, 2017

Proposal for the PiHue Project

Prepared by Yan Yu

Computer Engineering Technology Student

Executive Summary

As a student in the Computer Engineering Technology program, I will be integrating the knowledge and skills I have learned from our program into this Internet of Things themed capstone project. This proposal requests the approval to build the hardware portion that will working with the software running on raspberry pi system. This project mainly focus on the connection between raspberry pi and Philips Hue. I will be collaborating with the following company/department Humber Department of Public Safety. In the winter semester I have no plan to form a group with the following students, who are also building similar hardware this term and working on the mobile application with me. The hardware will be completed in CENG 317 Hardware Production Techniques independently and the application will be completed in CENG 319 Software Project. These will be integrated together in the subsequent term in CENG 355 Computer Systems Project as a member of a 2 or 3 student group.

Background

By using this project, you can automatically turn on your lamp when you get home.

I have searched for prior art via Humber's IEEE subscription selecting "My Subscribed Content" and have found and read which provides insight into similar efforts.

The first article I found discusses how to control devices through wireless transmission(Yin & Keoh, 2016).

The second journal talks about design of smart home control system by Internet of Things based on ZigBee(Yi, Zhou, & Liu, 2016).

The last one is a breakdown of Amazon echo digital personal assistant(Dempsey, 2015).

In the Computer Engineering Technology program we have learned about the following topics from the respective relevant courses:

• Java Docs from CENG 212 Programming Techniques In Java,

• Construction of circuits from CENG 215 Digital And Interfacing Systems,

• Rapid application development and Gantt charts from CENG 216 Intro to Software Engineering,

• Micro computing from CENG 252 Embedded Systems,

• SQL from CENG 254 Database With Java,

• Web access of databases from CENG 256 Internet Scripting; and,

• Wireless protocols such as 802.11 from TECH152 Telecom Networks.

This knowledge and skill set will enable me to build the subsystems and integrate them together as my capstone project.

Methodology

This proposal is assigned in the first week of class and is due at the beginning of class in the second week of the fall semester. My coursework will focus on the first two of the 3 phases of this project:

Phase 1 Hardware build.

Phase 2 System integration.

Phase 3 Demonstration to future employers.

Phase 1 Hardware build

The hardware build will be completed in the fall term. It will fit within the CENG Project maximum dimensions of 12 13/16" x 6" x 2 7/8" (32.5cm x 15.25cm x 7.25cm) which represents the space below the tray in the parts kit. The highest AC voltage that will be used is 16Vrms from a wall adaptor from which +/- 15V or as high as 45 VDC can be obtained. Maximum power consumption will be 20 Watts. *Phase 2 System integration*

The system integration will be completed in the fall term.

Phase 3 Demonstration to future employers

This project will showcase the knowledge and skills that I have learned to potential employers.

The tables below provide rough effort and non-labour estimates respectively for each phase. A Gantt chart will be added by week 3 to provide more project schedule details and a more complete budget will be added by week 4. It is important to start tasks as soon as possible to be able to meet deadlines.

Labour Estimates	**Hrs**	**Notes**
Phase 1		
Writing proposal.	9	Tech identification quiz.
Creating project schedule. Initial project team meeting.	9	Proposal due.
Creating budget. Status Meeting.	9	Project Schedule due.
Acquiring components and writing progress report.	9	Budget due.
Mechanical assembly and writing progress report. Status Meeting.	9	Progress Report due (components acquired milestone).
PCB fabrication.	9	Progress Report due (Mechanical Assembly milestone).
Interface wiring, Placard design, Status Meeting.	9	PCB Due (power up milestone).
Preparing for demonstration.	9	Placard due.
Writing progress report and demonstrating project.	9	Progress Report due (Demonstrations at Open House Saturday, November 12th, 2016 from 10 a.m. - 2 p.m.).
Editing build video.	9	Peer grading of demonstrations due.
Incorporation of feedback from demonstration and writing progress report. Status Meeting.	9	30 second build video due.
Practice presentations	9	Progress Report due.

1st round of Presentations, Collaborators present.	9	Presentation PowerPoint file due.
2nd round of Presentations	9	Build instructions up due.
Project videos, Status Meeting.	9	30 second script due.
Phase 1 Total	**135**	
Phase 2		
Meet with collaborators	9	Status Meeting
Initial integration.	9	Progress Report
Meet with collaborators	9	Status Meeting
Testing.	9	Progress Report
Meet with collaborators	9	Status Meeting
Meet with collaborators	9	Status Meeting
Incorporation of feedback.	9	Progress Report
Meet with collaborators	9	Status Meeting
Testing.	9	Progress Report
Meet with collaborators	9	Status Meeting
Prepare for demonstration.	9	Progress Report
Complete presentation.	9	Demonstration at Open House Saturday, April 8th, 2017 10 a.m. to 2 p.m.
Complete final report. 1st round of Presentations.	9	Presentation PowerPoint file due.
Write video script. 2nd round of Presentations, delivery of project.	9	Final written report including final budget and record of expenditures, covering both this semester and the previous semester.
Project videos.	9	Video script due
Phase 2 Total	**135**	
Phase 3		
Interviews	TBD	
Phase 3 Total	**TBD**	

Material Estimates	**Cost**	**Notes**
Phase 1		
RaspBerry Pi 3 Starter Kit	$89.99	https://www.amazon.com/Vilros-Raspberry-Ultimate-Starter-Kit-Clear/dp/B01CYWE20U
Pi Camera Module with Case	$38.79	https://www.amazon.com/Raspberry-Pi-Camera-Module-Megapixel/dp/B01ER2SKFS/ref=sr_1_2?s=pc&ie=UTF8&q 2&keywords=raspberry+pi+camera
Piezo Buzzer Element (Vibration Sensor)	$5.19	Canada Robotix
LED	$0.50	Canada Robotix
USB GPS Dongle.	$50	Amazon
Phase 1 Total	**>$200.00**	
Phase 2		
Materials to improve functionality, fit, and finish of project.		
Phase 2 Total	**TBD**	
Phase 3		
Off campus colocation	<$100.00	An example: [4].
Shipping	*TBD*	
Tax	*TBD*	
Duty	*TBD*	
Phase 3 Total	**TBD**	

Concluding remarks

This proposal presents a plan for providing a smart home accessories. This is an opportunity to integrate the knowledge and skills developed in our program to create a collaborative IoT capstone project demonstrating my ability to learn how to support projects. I request approval of this project.

Abstract

Philips Hue starter kit is a great product. I want to discover the possibilities of connecting Hue bridge to raspberry pi.

Table of Contents

1.0 Introduction

1.1 Purpose

The purpose of this documentation is to give a description of the “PiHue” in both sides on hardware requirements and software requirements. It will also contain a explaination of the application.

1.2 Scope

The main use of PiHue is to help you control your lights in your home. Philips has developed a wireless lighting system called Philips Hue. It allows you wirelessly control your lights of your home by using application or voice assistance with Amazon Alexa, Apple HomeKit and Google Home. The idea of this project is hacking the bridge with available resources and setup connection between raspberry pi and the bridge. Therefor you can control the lights wirelessly on your raspberry pi with a dedicated API.

In beta phase, after set up the system, you can easily control your Hue through raspberry pi.

In final phase of PiHue, raspberry pi will have the full functionality to control Philips Hue lighting systems wirelessly through raspberry pi.

1.3 Definitions, acronyms, and abbreviations

The Bridge:

The bridge is the heart of your Philips Hue system that connects your smart device to your Philips Hue lights. You can add up to 50 Philps Hue lights and accessories to one bridge. Linked to Wi-Fi via your router, it also connects your system to the wider world via the internet for out-of-home control and other smart features. The bridge is included in all Philips Hue starter kits, or you can buy it separatly and simply build your own Philips Hue system.

Smart Philips Hue LED bulbs:

The smart and energy-efficient LED lights bring Philips Hue to life. They deliver bright and beautiful light for your daily activities, special moments, and immersive experiences. They’re practical too. They

dim. They flash. They pulse. They do pretty much anything you want. And they're available in different shapes, sizes and models to suit your home.

Raspberry pi:

Small single-board computers.

Ethernet cable:

IEE802.3

IFTTT:

IFTTT is a free web-based service that people use to create chains of simple conditional statements, called applets. An applet is triggered by changes that occur within other web services such as Gmail, Facebook, Instagram, or Pinterest. IFTTT is an initialism for *If This Then That*.

ngrok:

ngrok secure introspect able tunnels to localhost webhook development tool and debugging tool.

Node.js:

Node.js is an open-source, cross-platform JavaScript runtime environment for developing a diverse variety of server tools and applications.

npm:

npm is the default package manager for the JavaScript runtime environment Node.js.

pip:

pip is a package management system used to install and manage software package written in Python.

Python:

Python is a widely used high-level programming language for general-purpose programming.

Brew:

Brew program, a build system used within and internal to Red Hat to build their Linux distribution packages.

JSON:

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language.

Apache:

The Apache HTTP Server, colloquially called Apache, is the world's most used web server software.

2.0 System Requirements Specifications

2.1 Product Description

2.1.1 Goal

This project will combine Raspberry Pi, Philips Hue and the bridge, connect all the device by internet, more specifically all the devices will connect in a local network area. Control lights using raspberry pi.

2.1.2 Targeted Users

Tech-fans, with modern technology to build smart home accessories.

2.1.3 Overview Of Product

PiHue includes a Raspberry 2 Model B, Philips Hue bulbs, the bridge, a router or a ethernet switch, power adapter and ethernet to USB adapter(in case your laptop do not have a ethernet port). Also, website service will be needed as IFTTT, you will need to create a account on this website. This website is used to trigger event or make web requests. A json server is necessary for handling the data transmission and data update. Local host or using other available server is your choice. In my case, I host a local server. If you are using local host, you will need to make it public accessible by a secure line.

2.2 System Perspective

2.2.1 Product Perspective

This product is open source, with the hopes that users will modify and distribute their own customized version. It will be written by Python on the Raspberry Pi side. It will using some libraries adopted from internet.

2.2.2 Product Functionality

Tha main functionality of my product is to provide user to control lights in your home. Connect the bridge and raspberry pi in the same local network. For testing and demonstration purpose, raspberry pi and the bridge are connected to a ethernet switch, and the ethernet switch is connect to laptop witch has setup the internet sharing function. This project is designed to work with the internet. Users will take input from keyboard to turn On/Off the lights, also adjust the brightness. On the mobile side, with a IFTTT account, users will be able to control light on their cellphone. The software on raspberry pi will have the function to display the current light status on the screen in a fullscreen mode. Users will get SMS message when the lights on or off.

2.2.3 Requirements

Internet connection is needed to send command and script to bridge to turn on or off the lights. A router will be needed to build a secure line for the bridge. A ethernet switch will be helpful to test it in a common public places which normally not easy to connect bridge and raspberry pi in the same network area, for example, college and library.

2.3 Overall Description

2.3.1 System Interface

System Interface for my project includes raspberry pi and a the bridge, a router or a ethernet switch used to create the local network.

2.3.2 Hardware

The main process of this project is to understand communication of Hue bridge. And learn how to set up a connection between raspberry pi and the bridge in a public place.

2.3.3 Software

User will use raspberry pi to integrate with the bridge. Hue API for raspberry pi will be introduced by the following section. Python is the programming language used for this project.

2.4 Build Instruction

2.4.1 Introduction

Understanding how Philips Hue works, and hack Hue bridge with raspberry pi then set up connection between raspberry pi and the bridge. I have searched through the internet for a proper installation on raspberry pi and found this build instructions adopted from Randy Reed on hackster.

2.4.1.1 Bridge

The bridge used to enable your smart bulbs to communicate with other devices via the internet. The main API offered by the bridge can be accessed when your app and bridge are on the the local network. Also I noticed the bridge's ethernet cable wiring is different than the standard TIA/EIA 568B Wiring.

2.4.1.2 Bulbs and ZigBee

These smart bulbs contain 3 types of LED specifically chosen to produce a range of colours and intensities. The bulbs I use is Philips Hue White A19 Bulbs. They are connected to the bridge via an open standards protocol called ZigBee Light Link. ZigBee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios, such as for home automation, medical device data collection, and other low-power low-bandwidth needs. You can find more information on the developers' website http://www.zigbee.org.

2.4.2 Preparation

Start up with Philips Hue API, you can check out the API from Philips https://www.developers.meethue.com. Also a hue library for raspberry pi is needed, I found two libraries on GitHub, you can choose one of them or both: phue and pyhue.

2.4.2.1 Hardware Wiring and Network Setup

Connect raspberry pi and the bridge in the same network area location is very easy to do in your house network, just connect raspberry pi wireless or wired to the router, and connect the bridge to the router wired, and you are ready to go.

But in a public area, it's could be difficult to do. In Humber College, I tried to connect raspberry pi to the college WiFi, however, raspberry pi does not recognized by our network, and can not be configured. The reason is Humber uses a PEAP(Protected Extensible Authentication Protocol) network protocol, this protocol is a high level protected communication channel. It requires a server-side PKI certificate to create a secure TLS tunnel to protect user authentication and use server-side public key certificates to authenticate the server.

To solve this question, I set up a router at school, by creating your own network and set up your protocol, it it possible to connect bridge and raspberry pi in the same network. I used a AirPort Express, you can easily set up and administrate this router from your Mac or iOS devices.

Finally, using AirPort Express is one way to set up your network, but I discovered a new way and the best way to handle this situation. Taking your raspberry pi outside to test your project can be messed up, you have to connect power cable, keyboard, mouse and a display. Then you have to set up your router and bridge, this process is possible but not convenience. I found a short cut to do this. You will set up the connection to by the following diagram.

~.....MacBook (Ethernet port or Adaptor).....~

~.....| Ethernet cable |.....~

~.....Ethernet switch <=> power adaptor.....~

~.....| Ethernet cable |.....~

~...../////.....\\\\.....~

~.....The bridge <=> power adapter <=> raspberry pi.....~

Next, you need to configure your network setting on your MacBook.

Go to your System Preferences, then select Sharing, on the Internet Sharing section, turn on Ether-

net Port or Ethernet adapter then turn on Internet Sharing. Now open VNC Viewer(download from https://www.realvnc.com), (your raspberry pi should be configured to turn on VNC server), enter the server address normally 192.168.2.2, and you are connected to raspberry pi and can use your MacBook's keyboard, trackpad and display. The bridge's IP address would be 192.168.2.3.

2.4.3 System Diagram

Input on raspberry pi side

#.....INPUT.....#

#.....Raspberry Pi.....#

#.....Action.....>.....#

#.....Philips Hue ApI.....Trigger.....#

#.....Hue Bridge.....IFTTT.....#

#.....Lights.....Message.....#

Input on smartphone

#.....Press Button.....#

#.....IFTTT.web.request.....#

#.....PUT.data.to.server.....#

#.....check.data.on.server.....#

#.....IF.updated.....#

#.....Action.to.Hue.API.....#

#.....Lights.....#

2.4.4 Cost of Material

Required Items	Quantity	Cost
Philips Hue White A19 Starter Kit	1	$69.99

TP-Link Desktop Switch	1	$14.99
Ethernet Cable	3	$4.99
Raspberry pi 2 Model B	1	$49.99
PiTFT Touchscreen 3.5”	1	$59.09
USB-C to Gigabit Ethernet Adapter	1	$29.00
Total	8	$228.05

2.4.5 Time Commitment

Todo	**Time Required**
Hue library and other software installation	(1 hr) I take this amount of time to find out the proper library for the system.
Hue bridge connection	(1 hr) I using meethue.com to find out the unique ID for the bridge and testing the command with the bridge.
Server setup	(1 hr) Following this instruction you can finish that in 30 min.
Sample code testing	(2 hr) Error fixing.
Server testing	(1 hr) Connection and database.
Total	**5 hr**

2.4.6 Hardware Installation

2.4.6.1 PiTFT Plus Touchscreen Installation

For easy way to control the bridge using raspberry pi, you can install a touchscreen. I used PiTFT Plus 480x320 3.5” TFT+Touchscreen from adafruit. The installation process is followed by adafruit. I choose easy install option which you can download the fully prepared image from the website above. After that you will start up a fresh sd card, use SDFormater to formate your sd card. Then I used ApplePi Baker(you can find other useful tools on http://elinux.org/RPi_Easy_SD_Card_Setup) to burn the image to the sd

card. After that plug in your sd card to the raspberry pi and you are ready to start using the touchscreen.

[Update] I noticed if you update your raspberry pi to the latest version, it will overwrite the configuration file of the touchscreen, and the raspberry pi will stuck at the booting section leave a black screen. I suggest you to follow section 2.4.2.1 for the setup.

2.4.7 Software Installation

2.4.7.1 System Update

forst, update you system's package list by:

```
sudo apt-get update
```

Next, upgrade all your installed packages to their latest versions with the command:

```
sudo apt-get dist-upgrade
```

2.4.7.2 Library Installation

Two kinds of libraries for pi I found is pyhue and phue.

phue: `sudo easy_install phue` or `pip install phue`

requests: `sudo apt-get install python-pip`

```
sudo pip install requests
```

2.4.7.3 Other Requirement

Node.js is a server-side platform built on Google Chrome's JavaScript Engine. You might need this for your server environment.

Download the package directly from the website.

Or install it form terminal.

```
$ cd /tmp
$ wget http://nodejs.org/dist/v6.3.1/node-v6.3.1-linux-x64.tar.gz
```

```
$ tar xvfz node-v6.3.1-linux-x64.tar.gz

$ mkdir -p /usr/local/nodejs

$ mv node-v6.3.1-linux-x64/* /usr/local/nodejs
```

2.4.7.4 The bridge Setup

Using www.meethue.com/api/nupnp to discover the IP address and ID of the bridge.

Go to http://ip_address_you_found/debug/clip.html.

Address `http://<bridge ip address>/api`

URL `/api`

Body `{"devicetype":"my_hue_app#pi tony"}`

Press the button on the bridge and then press **POST** button to generate the unique ID.

Address `http://<bridge ip address>/api/unique_id/lights`

Body

GET

Address `http://<bridge ip address>/api/unique_id/lights/1` \

Body

GET

Address `http://<bridge ip address>/api/unique_id/lights/1/state`

Body `{"on":false}`

PUT

2.4.7.5 IFTTT Setup

Create a IFTTT account on https://ifttt.com.

Start using Maker Channel to extends the possibility of your project.

2.4.7.5.1 Receive a web request

After create your IFTTT account, search for Maker Webhooks, in Maker Webhooks, click Settings, now you will see a url for your maker channel, go to the url in a new webpage which will be used later. Now go back to IFTTT homepage and click My Applets, then New Applet, to load new applet creation page. Select `this` to to choose a service, then search maker to receive a web requests, enter a event name you will be used and hit Create trigger. Following on that, select `that` keyword and search SMS service, you will be asked to enter your phone number and then edit the message you want to sent, hit on Create action button. Your applet is now created. Place your event name in the url you have created before and test it with `curl` from a command line.

```
curl -X POST https://maker.ifttt.com/trigger/{event}/with/key/your_key
```

2.4.7.6 Server Setup

I set up my local JSON Server by using json-server from Github. Following the document on the web-page, you can set up your local server. Of course this server is a very lite version json server, but it is enough for my project. If you want to do more on the database, you can also use Firebase from Google or AWS from Amazon. For the installation part, you will need `node` to install json-server. Make sure you have `Homebrew` and `Xcode` installed on your Mac.

2.4.7.6.1 JSON Server

Install Xcode:

Xcode can be download directly from the App Store.

Install `Homebrew` on macOS, paste the line below at a terminal prompt:

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master
```

Install `node` and `npm`, open terminal and type:

```
brew install node
```

Install `json-server`, in terminal:

```
npm install -g json-server
```

After install the `json-server` create a `db.json` file with some data in it, then start JSON Server with:

```
json-server --watch db.json
```

Then go to http://localhost:3000 you will see the data you have created in the db.json file. Functions like GET, POST, PUT, DELETE, HEAD, PATCH are all included.

2.4.7.6.2 Access Local Host From Anywhere

if your localhost does not start or you can not see the localhost webpage, you probably need to start your apache server. In macOS, apache server is included by default. Start it with:

```
apachectl start
```

Now you have your server online, but you can't visit it outside of your local network. After doing some research, I find there are actually plenty of ways to make your local host go public by using a secure tunnel.

`ngrok` is an application for Windows, macOS and Linux that creates a tunnel but also allows you to inspect all traffic that goes through the tunnel. Download ngrok from it's website, unzip it by `unzip /path/to/ngrok.zip`, navigate to the folder it located and run it with `./ngrok help`.

in my case, as I use `localhost:3000`, I would set this port publicly.

```
./ngrok http 3000
```

A successful message would shows up, and you will see it online and two forwarding addresses created by ngrok. Open the url from any device with internet connection and you will accessing your localhost. If you go to http://localhost:4040 on your computer, a traffic history will be provided.

2.4.7.6.3 Test Server from terminal

Once you started your server, test it form terminal,

Return a detailed list of your database:

```
curl -X GET "http://localhost:3000/db"
```

To insert a new section for your posts section, you will need to send data using a POST request:

```
curl -X POST -H "Content-Type: application/json" -d '{your_new_data}' "http://localhost:
```

Get your new data by:

```
curl -X GET "http://localhost:3000/posts"
```

To update your existing data, you will use PUT requests:

```
curl -X PUT -H 'Content-Type: application/json' -d '[your_JSON_data]' "http://localhost:
```

2.4.7.6.4 Test Server using python

I found a Quickstart for python-requests from this website: http://docs.python-requests.org/en/master/. Make sure your requests is installed and up to data.

Open a new terminal window then type: `python`

```
>>> import requests
>>> import json
```

Now, get your local host:

```
>>> r = requests.get('http://localhost:3000')
```

Get your response code:

```
>>> r.status_code
200
```

Get the header:

```
>>> r.headers['content-type']
'application/json; charset=utf8'
```

Get the text:

```
>>> r.text
u'{...}
```

Get JSON data:

```
>>> r.json()
{u'...}
```

POST to your database:

```
r = requests.post('http://localhost:3000/posts', data = {'key':'value'})
```

Other functions will be similar:

```
r = requests.put('http://localhost:3000/posts/1', data = {'key':'value'})
r = requests.delete('http://localhost:3000/posts/2')
```

2.4.8 Program Testing

2.4.8.1 JSON data comparison

When you updated your database, you have to check the value changed or not:

```
import json
import requests

r = requests.get("http://localhost:3000/posts/1")
x = json.loads("""{"id": 1, "on": "true"}""")
y = r.json()
x == y
# result: True
```

2.4.8.2 Display and the Bridge initialization

I use `pygame` to display the lights status on the screen:

```
import pygame, sys
from pygame.locals import *


SCREEN_WIDTH = 1280
SCREEN_HEIGHT = 1024


pygame.init()
pygame.display.init()
pygame.font.init()
pygame.font.get_fonts()
size = (SCREEN_WIDTH, SCREEN_HEIGHT)
windowSurface = pygame.display.set_mode(size, pygame.FULLSCREEN)
```

You also need to refresh the screen by put the code at the end of your loop:

```
pygame.display.update()
```

Set up the bridge by `phue` library:

```
from phue import Bridge


b = Bridge('your_bridge_ip_address')
b.connect()
b.get_api()
```

Initialize on and off command for your lights:

```
MAX = 254
on_command = {'on': True, 'bri': MAX}
off_command = {'on': False}
```

I use the `pygame` function to let user take single key press to control the lights:

```
for event in pygame.event.get():
    if event.type == QUIT:
        pygame.quit()
        sys.exit()
    if event.type == KEYDOWN:
        if event.key == pygame.K_LEFT:
        b.set_light(2, on_command)
```

In the Hue API, transition times are specified in deciseconds(tenths of a second), this is not what we need if we want blink the lights. Setting transition times:

```
# Set brightness of lamp 1 to max , rapidly
b.set_light(1, 'bri', 254, transitiontime=1)
```

Now combine the IFTTT with your python code to turn on the light 1 when you press the right arrow key:

```
if event.key == pygame.K_LEFT:
    b.set_light(1, on_command)
    r = requests.post('https://maker.ifttt.com/trigger/{event}/with/key/your_key')
```

2.4.8.3 Using IFTTT to make a web requests

In the previous section you have set up the IFTTT to receive a web request, now you will do a little more work to make a web request to your server. Go to My Applets and select New Applet, then in `this` section select Button widget (or any other option you like). Follow the site and select `that` with Maker Webhooks, in make a web request section, fill the URL with your JSON-Server http link that have broadcast by `ngork`, in method field select `PUT`, content type you will select application/json, at the body section you put `{"id": 1, "on":"true"}` in the field and then save. Now in your python code you can use `request` to listen on your server. Ever time you press the button on your phone, it will blink the lights.

2.4.8.4 Other Issues

Initially I was trying to connect GPS from my smartphone to raspberry pi, and set a range, when I enter or exit this area, it will trigger the Hue. However, during my test, the GPS signal is weak in the building, and the data transfer got laggy. It's also not easy for testing purpose.

2.4.8.5 Final Project

You can find the full code from my Github: https://github.com/bigbosstony/bigbosstony.github.io/

3.0 Conclusion

As the smartphone has been the most important device we use everyday, and network based assistant is already taking place of so many area, like Google Now, Siri and Amazon echo. These smart devices are entering your house. With Philips Hue, it is possible to control lights in your home by your cellphone or just your voice. This project is focus on hacking the Hue bridge using raspberry pi through a local network.

Learning how the bridge is talking with the Philips Hue Bulbs is the key for this program. I simplified on the programming side, set up the connection between raspberry pi and the bridge by using python. There are multiple ways to do so. I first use IFTTT which is the easiest way to get connection with the bridge. The whole process is done by ZigBee protocol, and I am coding on the raspberry pi by python and existed hue library provided by GitHub. The python library foe Philips Hue is compatible with Philips Hue API 1.0.

The project gives you the freedom to use your lights and discover the possibility to extend ways to use Philips Hue. With A19 white Bulbs, you only find a little fraction of the Hue, but understanding the theory behind it, you can set up more ambitions project with colour ambiance kit A19 and others.

4.0 References

References (Generated in pdf)

Dempsey, P. (2015). The teardown amazon echo digital personal assistant [teardown consumer electronics]. *Engineering Technology*, *10*(2), 88–89. https://doi.org/10.1049/et.2015.0231

Yi, X. J., Zhou, M., & Liu, J. (2016). Design of smart home control system by internet of things based on zigbee. In *2016 ieee 11th conference on industrial electronics and applications (iciea)* (pp. 128–133). https://doi.org/10.1109/ICIEA.2016.7603564

Yin, X., & Keoh, S. L. (2016). Personalized ambience: An integration of learning model and intelligent lighting control. In *2016 ieee 3rd world forum on internet of things (wf-iot)* (pp. 666–671). https://doi.org/10.1109/WF-IoT.2016.7845398