Simple Traffic Light Project

September 14, 2020

By Timothy Gow

Online Link:

This Project can be found on my portfolio page:

http://timothyjgow.com/

Objective

The purpose of this project is to obtain an understanding of how to use the Raspberry Pi GPIO pins, creating a basic web server, and controlling the on/off states of LEDs

By the end of this lab we should be able to turn on/off the green, yellow, and red LEDs. We should be able to have cycle through them and stop the cycle.

Materials

- Raspberry Pi 3 or higher
- Red LED
- Yellow LED
- Green LED
- 3x 220 ohm resistors
- 4x male to female jumper wires
- Breadboard

References

The following were used as resources for this project:

- https://projects.raspberrypi.org/en/projects/traffic-lights-python
  This site gives you a good understanding of how to use python to program the LEDs to turn on and off.
- https://thepihut.com/blogs/raspberry-pi-tutorials/gpio-and-python-39-blinking-led

I used the circuit diagram image here as the basis of how to wire the LEDs

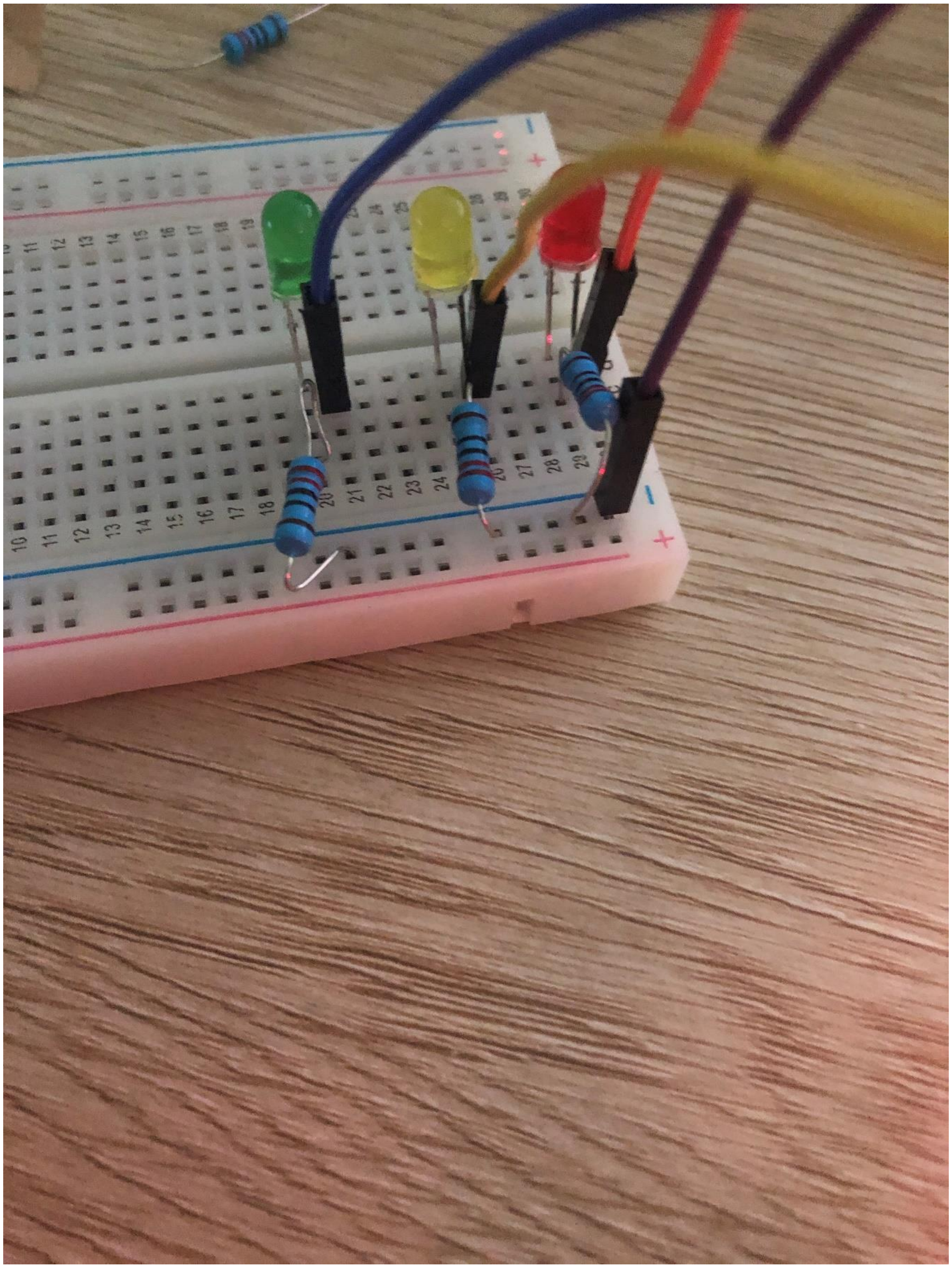
- https://www.raspberrypi-spy.co.uk/2012/06/simple-guide-to-the-rpi-gpio-header-and-pins/
  This was my guide to understanding the locations and functions of each pin.

Procedures:

It is assumed that you will already have Raspbian installed on your Pi. I also assume you have all other necessary items to be able to program your Pi.

1. Connect LEDs, resistors, wires and Pi. Each LED will have a resistor to avoid burning out the LED.

   This is how it will look on the breadboard side.

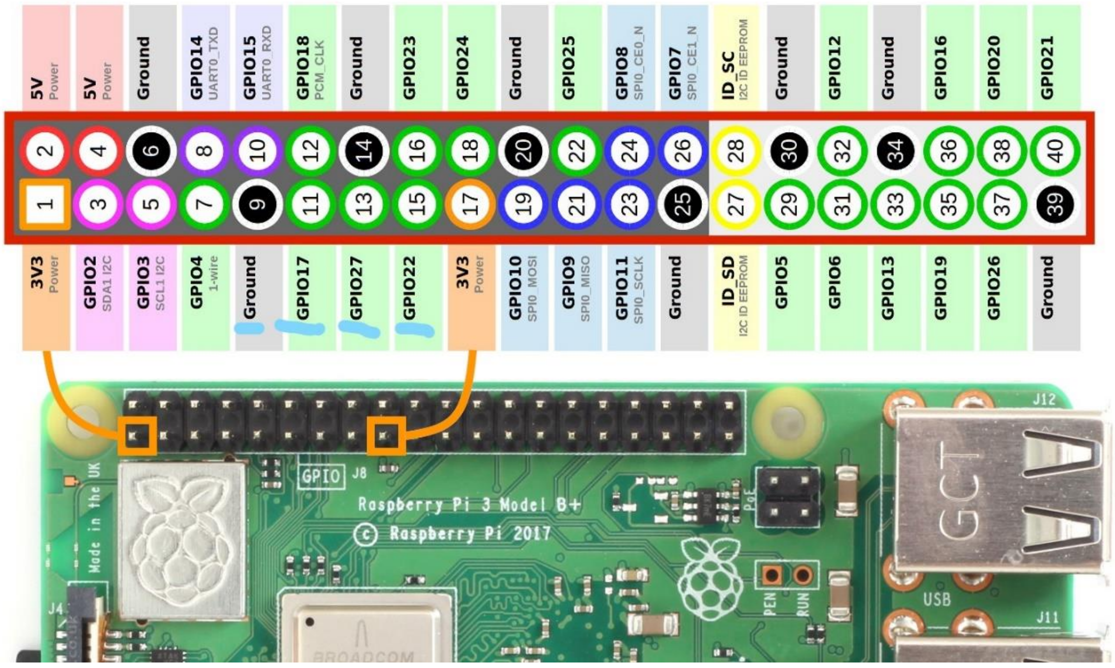


Green LED = pin 17

Yellow LED = GPIO pin 27

Red LED = GPIO pin 22

Ground should be connected to the ground right above GPIO pin 17



2. The next step is to write out the code for this to work.

We will be using Flask as our web server. Enter the following command in the terminal if you do not have flask installed on your machine.

```
$ pip install Flask
```

The source code can be found here:

https://github.com/Choyero/Traffic-Light-Raspberry-Pi.git

In the terminal navigate to an available folder where you want the project and clone the repository.

Once the code is set up you can go ahead and run the flask server.

In the terminal run

```
$ python contrellerLED.py
```

That will start the web server and it will be waiting for your input.

Finally, you can navigate to the port that the server started on. Typically, will be localhost:5000

Once there you can click on any of the buttons on the page to interact with the lights.

Thought Questions:

1. What language did you choose for implementing this project? Why?

   I used Python for my project. The reason being is because it is simple and requires very little to actually know about coding to get up and running.

2. What is the purpose of the resistor in this simple circuit? What would happen if you omitted it?

   The resistors serve as a way to lighten the load that the LEDs have. Without the resistors the LEDs would burn out.

3. What are practical applications of this device? What enhancements or modifications would you make?

   So this device at this scale is not very useful. But if we build this up to control lighting in our house or timed lights than it would be more useful. Some things we could enhance would be making lights turn on based on a event. For example maybe it is night and we need lights. Program the lights to turn on at sunset.

4. Please estimate the total time you spent on this lab and report.

   This whole process took maybe 6 hours to complete.

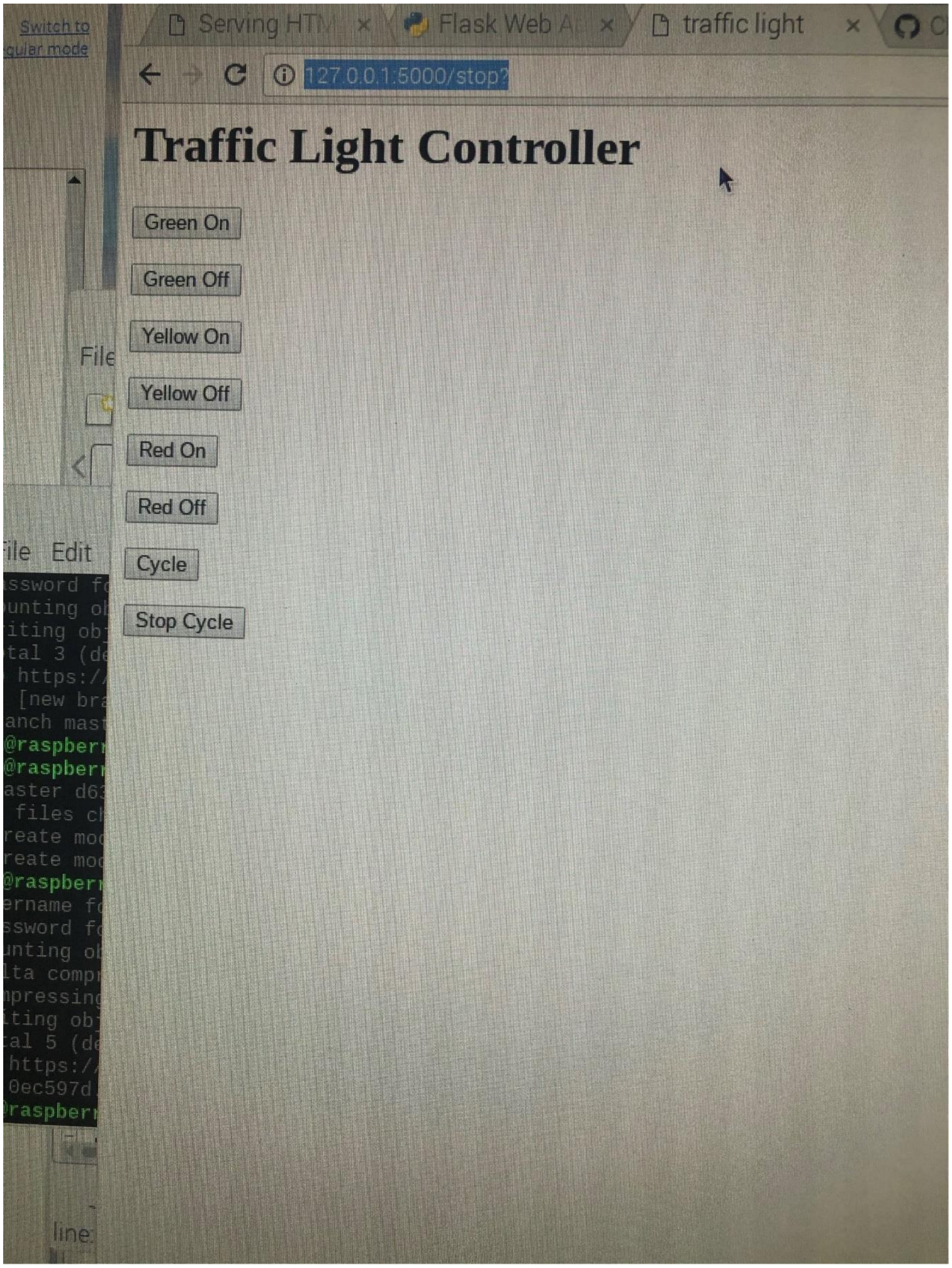
Certification of Work

I certify that the solution presented in this lab represents my own work. In the case where I have borrowed code or ideas from another person, I have provided a link to the author’s work in the references, and included a citation in the comments of my code.

Timothy Gow

Appendix

Web Interface

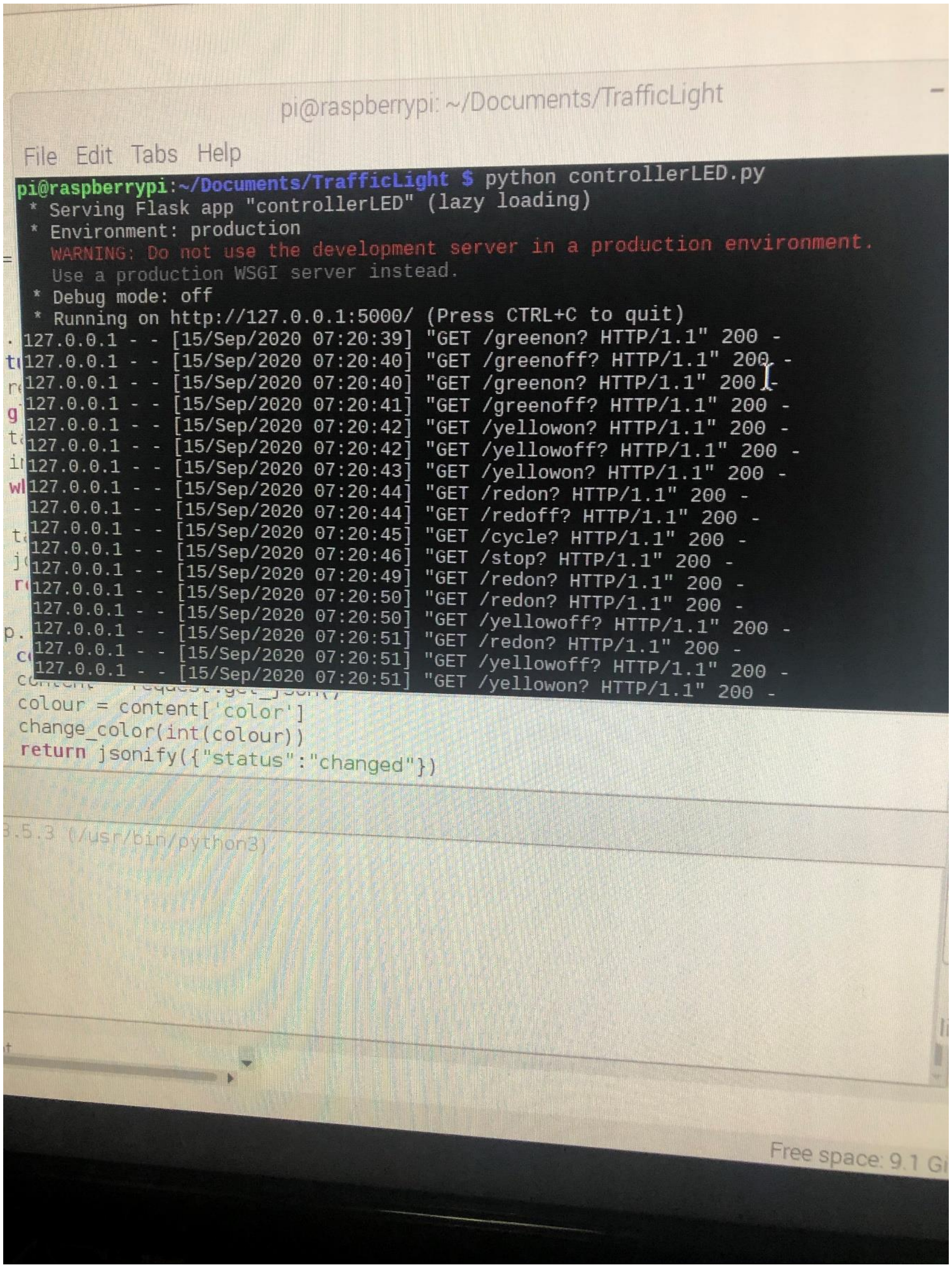


127.0.0.1:5000/stop?

Traffic Light Controller

```
ssword fo
ounting ob
riting obj
otal 3 (de
 https://
 [new bra
anch mast
@raspberr
@raspberr
aster d63
 files ch
reate mod
reate mod
@raspberr
ername fo
ssword fo
unting ob
lta compr
mpressing
iting obj
tal 5 (de
 https://
 0ec597d
@raspberr
```

Server Response



```
pi@raspberrypi: ~/Documents/TrafficLight
File  Edit  Tabs  Help
pi@raspberrypi:~/Documents/TrafficLight $ python controllerLED.py
 * Serving Flask app "controllerLED" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [15/Sep/2020 07:20:39] "GET /greenon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:40] "GET /greenoff? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:40] "GET /greenon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:41] "GET /greenoff? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:42] "GET /yellowon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:42] "GET /yellowoff? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:43] "GET /yellowon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:44] "GET /redon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:44] "GET /redoff? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:45] "GET /cycle? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:46] "GET /stop? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:49] "GET /redon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:50] "GET /redon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:50] "GET /yellowoff? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:51] "GET /redon? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:51] "GET /yellowoff? HTTP/1.1" 200 -
127.0.0.1 - - [15/Sep/2020 07:20:51] "GET /yellowon? HTTP/1.1" 200 -
```

```
    colour = content['color']
    change_color(int(colour))
    return jsonify({"status":"changed"})
```

```
3.5.3 (/usr/bin/python3)
```

Source Code

https://github.com/Choyero/Traffic-Light-Raspberry-Pi.git