# Lab 2 - Wifi-Controlled LED Stoplight (Arduino)

## Online Links

This lab is available on my personal website at: http://AaronNelson95.com/IT441Lab2.php
The code is also available on my GitHub repository at https://github.com/AaronNelson95/IT441

## Objective

The purpose of this lab is to learn more about the basics of GPIO pins by creating a wifi-controlled stoplight. This stoplight is connected to the GPIO pins of the Wemos D1 mini microcontroller. The Wemos board will broadcast a webpage where a user can choose the color they would like the stoplight to be, and the light will reflect that option. An auto option will also automatically rotate between the different light colors. This lab will help one learn:

- How to install and configure the Arduino IDE
- How to code and develop a program in Arduino using the setup and loop functions
- How to connect an Arduino board to wifi and broadcast a simple web server from it
- How basic state diagrams and flow charts help in developing and understanding a project
- How to work with the millis() function to control time-based events

## Materials

To complete this lab, you will need:

- A Wemos D1 mini microcontroller
- A Micro USB power cord
- A breadboard
- A traffic light LED display module
- 4 Male-Male Jumper Wires
- A computer with the Arduino IDE

*The LED traffic light*

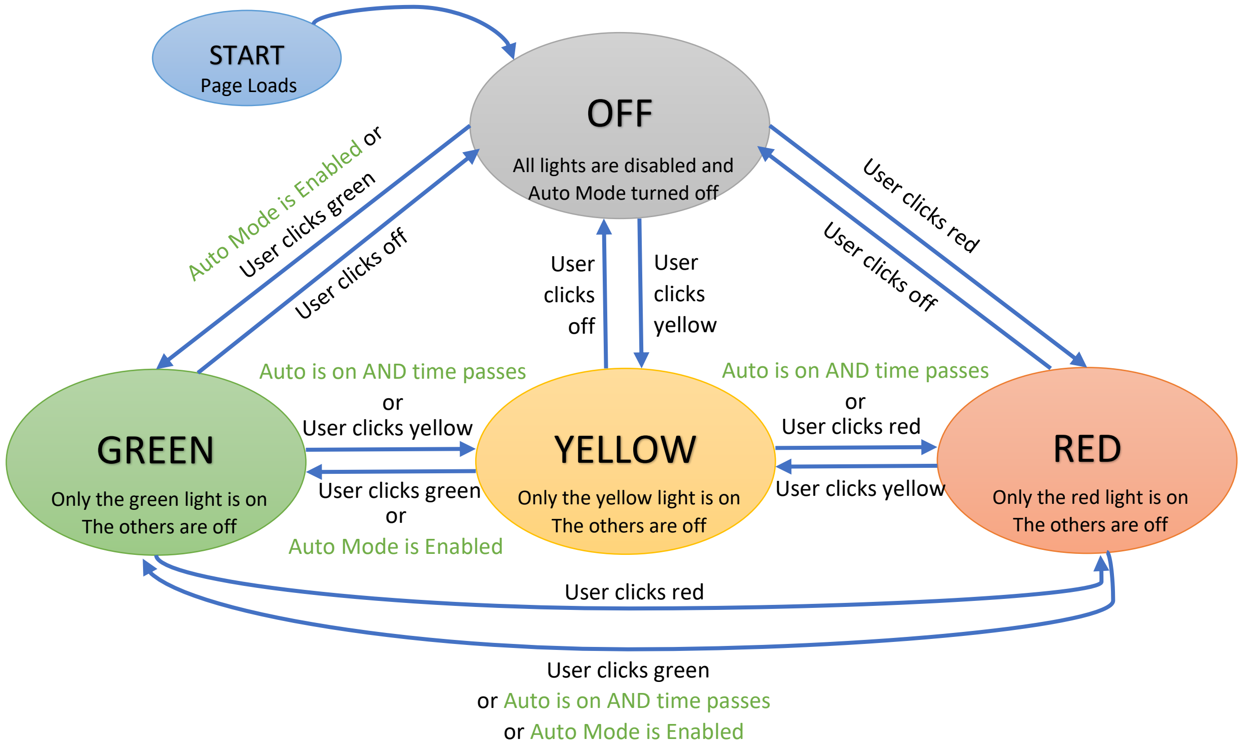
## References

The following links may be helpful throughout this lab:

- https://www.w3schools.com/ - A great reference for coding HTML, CSS, and JavaScript. This is handy for designing the website the client will view.
- http://www.esp8266learning.com/wemos-webserver-example.php - A basic tutorial for connecting the microcontroller to the internet and broadcasting a simple server. This is also available in the IDE through File -> Examples -> ESP8266WiFi -> WiFiManualWebServer.
- https://randomnerdtutorials.com/esp8266-web-server-with-arduino-ide/ - Another useful example tutorial for setting up a web server with Arduino
- https://www.teachmemicro.com/getting-started-wemos-d1-mini/ - The exact steps to take to initially setup Arduino IDE to work with the Wemos D1 mini, including the link where the esp8266 boards are located at for the boards manager.
- https://www.arduino.cc/en/main/software - Installing the Arduino IDE (I did it from the Windows 10 app store)

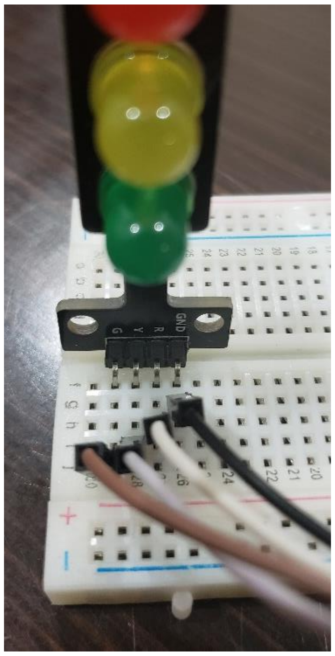
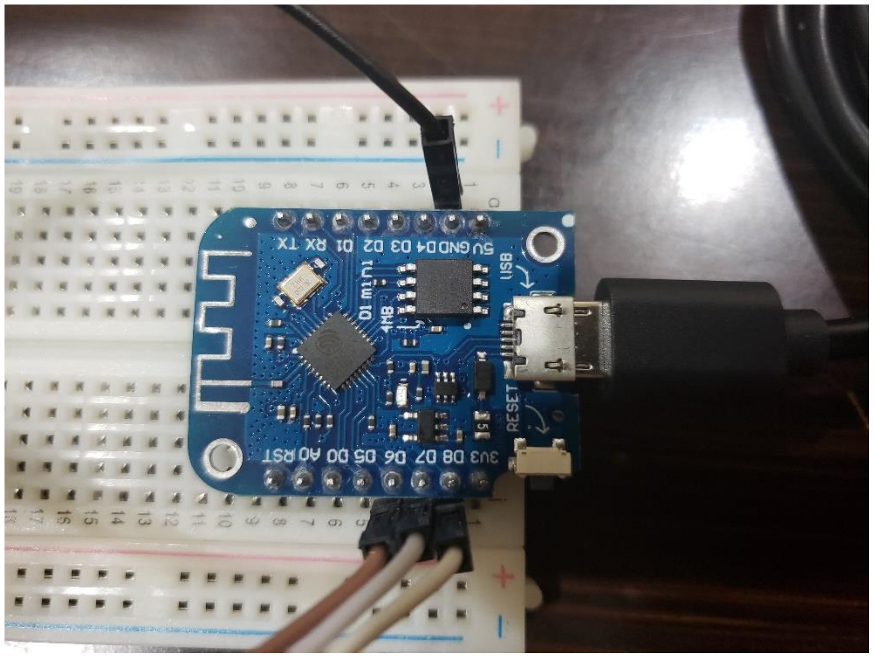
## Procedures

1. Install the Arduino IDE either from the Windows Store or from the Arduino website.
2. Setup the IDE to work with the Wemos D1 mini model
    a. Click **File** -> **Preferences**. Under **Additional Boards Manager**, input https://arduino.esp8266.com/stable/package_esp8266com_index.json. Click OK to exit out.
    b. Click **Tools** -> **Board:** -> **Boards Manager**. Here, search for ESP8266 and one result should appear. Click the **Install** button.
    c. Now select this board to work with. Go to **Tools** -> **Board:** -> **"LOLIN(WEMOS) D1 R2 & mini"**.
    d. Plug your Wemos board into the computer. Windows should automatically install necessary drivers for the microcontroller.
    e. Select the port the board is plugged into by going to **Tools** -> **Port:**. Next select the COM port your chip is in. If multiple ones are recognized, consult the Windows Device Manager to find the correct port.
3. Design documents to help you understand the logical flow of the system.
    a. Develop a state diagram.
        i. There should be 4 light states: when no lights are on, when the green light is on, when the yellow light is on, and when the red light is on. There is also an "auto" mode that, when enabled, will rotate between the green, yellow, and red lights after time passes.
        ii. When the board is turned on, the device should start and go to the off state. Here, all of the GPIO pins for the three lights should be set off.
        iii. When a user selects one of the color options, go to that state. Also when they select one of these, the timers for the auto state should be canceled, if they are running.
        iv. When auto mode is enabled, move to the green state. Here, a timer is set (which will be done by incorporating "millis()"). If auto mode is still on (a different state has not been manually selected), and so much time passes, then move to the yellow state. Begin another timer and when another period passes, move to the red state and once again begin another timer. When that time passes, move back to the green state and repeat the cycle.


START
Page Loads
OFF
All lights are disabled and Auto Mode turned off
Auto Mode is Enabled or User clicks green
User clicks off
User clicks off
User clicks yellow
User clicks red
User clicks off
GREEN
Only the green light is on
The others are off
Auto is on AND time passes or User clicks yellow
User clicks green or Auto Mode is Enabled
YELLOW
Only the yellow light is on
The others are off
Auto is on AND time passes or User clicks red
User clicks yellow
RED
Only the red light is on
The others are off
User clicks red
User clicks green or Auto is on AND time passes or Auto Mode is Enabled

   b. Develop a system flowchart
      i. This should show an idea of how to program the microcontroller. Arduino has two main functions: the startup() one, which runs once as soon as the board gains power, and the loop() function which continually runs after the startup is finished. Try to consider how you can implement the state diagram using these two functions. This diagram will be modified as the code progresses.
      ii. An example of a finished system flowchart for this lab is available in the appendix.

4. Connect the Wemos board to the stoplight by using a breadboard and jumper wires. In this project, the green light will be connected to pin 12 (D6), the yellow light is on pin 13 (D7), the red light is on pin 15 (D8), and the stoplights ground is on the board's GND. A circuit diagram of this setup is available in the appendix.

5. Begin programming the Arduino code. First, learn how to work with the setup() and loop() functions by building a simple program that will cycle a light on and off. In the setup, use the command "pinMode(15, OUTPUT);" to set the red light to the output mode. In the loop, you can use "digitalWrite(15, HIGH); delay(1000); digitalWrite(15, LOW); delay(1000);" to turn the light on and off continuously.
   a. Save the sketch to the board by selecting **Sketch** -> **Upload**. The code will be compiled and written to the board. If successful the light will flash on and off.
6. Implement the system flowchart and state diagram designed earlier. For help and practice with implementing a webserver (the page that a user will select an option from), check the available examples such as the one at **File** -> **Examples** -> **ESP8266WiFi** -> **WiFiManualWebServer**. Try to design a simple page before you begin implementing all of the code at once.
   a. Take the project steps at a time, such as by developing the webpage, managing the off and color light functionality, and finally work with the auto mode.
   b. For auto mode, you will want to utilize "millis()", which keeps track of the time that passes since the board was turned on. You can compare the current value of millis() to one that was saved and recorded when, say, auto mode is turned on, to see how much time passes.
   c. The serial monitor (available at **Tools** -> **Serial Monitor**) can be helpful with debugging. In the setup, type "Serial.begin(9600);" to set the monitor to read inputs at the 9600 speed (the speed is set in the bottom right of the monitor). Use "Serial.println("text");" to write "text" to the monitor. This can be helpful with identifying problems and the places the code stops working.

## Thought Questions

1. What are some key differences between developing this lab on a Raspberry Pi, and developing on Arduino?

   There are a couple of major differences between developing this lab on a Pi and on an Arduino. First of all, when developing on the Raspberry Pi, we had more freedom to choose which language we would like to develop with. Essentially, we created a web *site* hosted with Apache, and then could run scripts that essentially just ran terminal commands to turn GPIO pins on and off. These scripts could be written in different files and be called at any time. Developing on Arduino is quite a bit different as you do not just write functions that run, for example, when a button is clicked. Instead, Arduino contains one single script where essentially everything is either initialized in the setup, or it is *continuously* cycling in the loop portion of the code. Trying to think of how to code in a loop takes a bit of a learning curve. You have to realize that different functions will be trying to run all the time, so there would be a heavy use of "if" statements, to prevent undesired portions of code running every cycle. In addition, you also have to be careful of blocking the loop from progressing, such as by using never ending while statements or the "delay" command. Instead, you need to use techniques that utilize millis() so the loop can continue working. Also, the web *page* the client sees is also stored directly in the loop and may be repeatedly reloaded any time an action is taken. This page can handle user requests like an API does.

2. What are the strengths and trade-offs of each of these platforms?

   The Arduino boards are relatively cheap and are much smaller than the Raspberry Pi. They can be easily stored in a small place and they don't require a massive amount of processing power. There are less slowdowns as it is continually running the same script, unlike a Pi, where other processes may also be running fighting for CPU. The Arduino boards mainly use the Arduino code so there isn't as much freedom in development languages. However, there are lots of libraries to pull from that can work with the handy shields, and many of these libraries have very helpful built-in examples.

   A Raspberry Pi can handle more resource-hungry requests. It has a CPU, RAM, and storage like a regular computer does. Here, there is a filesystem and GUI that information can be stored into. Whereas the Arduino is built to run a single purpose from a script, a Pi can have multiple scripts running and saved to be accessed at any time. Once a new script is written to the Arduino board, the old one is lost completely form its memory. The Pi also has more GPIO pins, USB ports, and other I/O options. A Pi can also handle entire web servers easier as well as providing photos and other multimedia features directly on it. However, a Pi is much more expensive and takes more resources to run.

3. How familiar were you with the Arduino platform prior to this lab?

   Before I started this lab, I had literally never heard the word "Arduino" before. Even when the teacher mentioned it, I was honestly wondering if it was a real thing or if he had a word mixed up. Aside from a little bit in an IT class at school, I had never worked with microcontrollers before. This lab was a little bit of a learning curve for me with trying to wrap my mind around how I can program things within a continually cycling loop, but it was a fun challenge and I learned a lot from the experience. I now feel comfortable with programing with Arduino after this lab, and I am excited to work on more projects with it.

4. What was the biggest challenge you overcame in this lab?

   I actually had two major issues with this lab that took a lot of time to solve. For one, I had a challenge getting millis() to work properly. I tried it in multiple ways and with different techniques (adding time to it beforehand for a comparison, subtracting the current from the past and seeing how it compares to a period of time, etc.), but it would just not work! Somehow, somewhere, the code wasn't cycling the way it should have, even though everything logically seemed that it should work. I actually scrapped the project twice and started over from scratch to see if I could rethink it over. The third time, I finally got the auto mode to work correctly!

The main differences I used was taking out every place I used an if/else statement and separating them *all* into separate if statements. By doing this, script cycled correctly and auto mode worked.

I also had a very strange issue with the wifi. My board, although it was working totally fine before, just completely stopped connecting to any wifi source. The serial monitor also began throwing out lots (like a ton) of cryptic numbers. I troubleshooted and found this was happening when it was trying to establish the wifi connection. I searched for a while on the internet to try to find a solution, but it didn't help much. Eventually, what did help, was when I erased the entire board by setting **Tools** -> **Erase Flash** -> **All Flash Contents**. I reuploaded my project, and turned this setting back to only erasing the sketch, and it has worked fine ever since.

5. Estimate the time you spent on this lab and report.

   This was my first time working with Arduino so it took a little bit of time getting used to the new environment. Unfortunately, I had the two major issues listed before, and that took up the vast majority of my project time (like hours with the wifi issue, and of course lots of time being stumped on why millis() wasn't working like it should). It took me around 12 hours on the project. This report took me about 4.5 hours. Now that I am getting into the swing of things and have basic templates, I am able to get my work done much faster and efficiently.

## Certification of Work

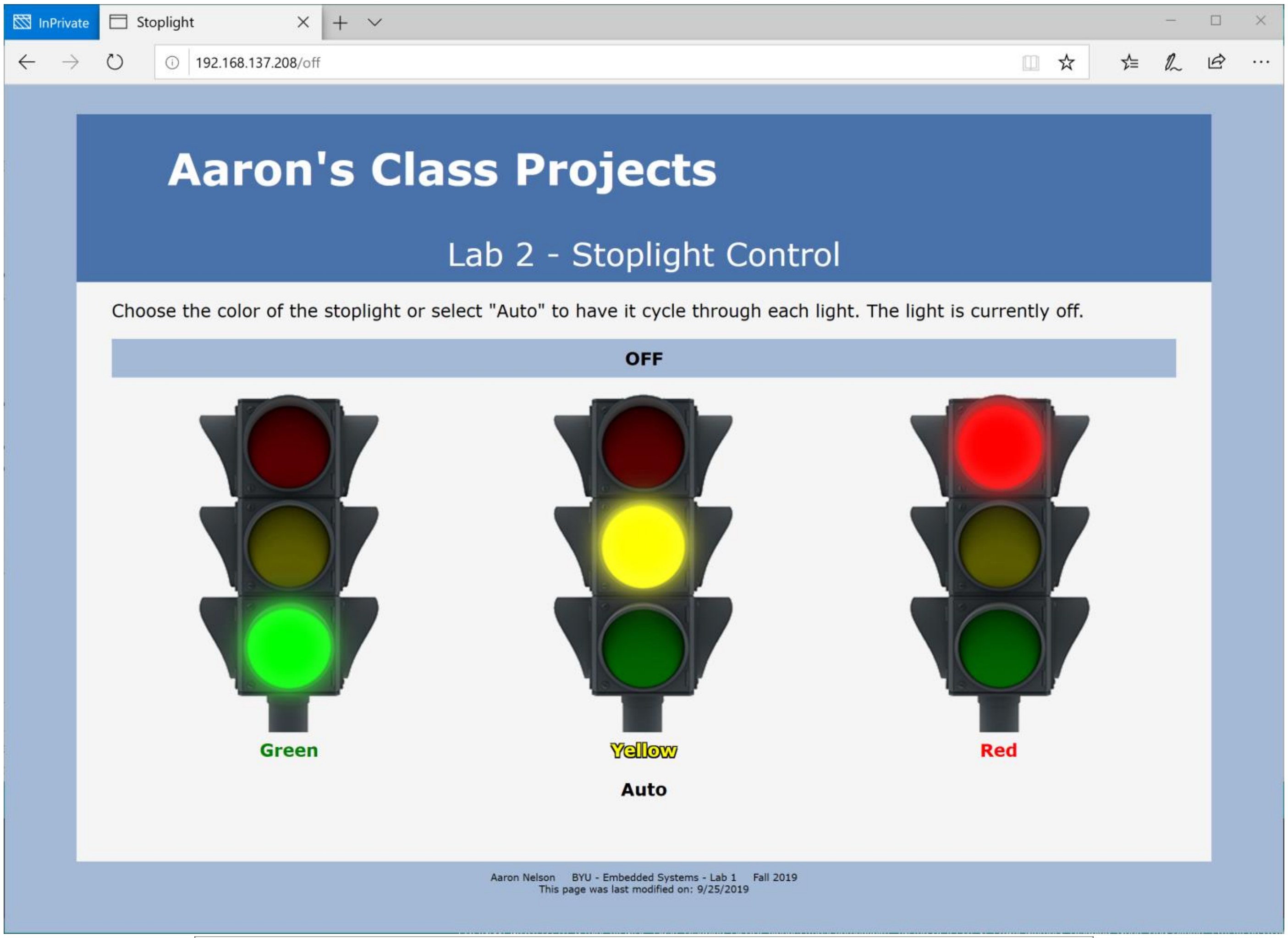
I certify that this lab represents my own work. I used various web resources to help get me started with Arduino, and a simple web server example, but I did not copy anything that was not basic knowledge unless I specified it inside of my code comments. I did use three images on my page that were obtained from the internet, and that web resource is linked to in my code comments. Also, I used my same webpage layout from lab 1 for this lab, which was also my original work.

– Aaron Nelson

## Appendix

### Images of Final Product

As can be seen in the images below, when a light is selected on the webpage, the respective light will turn on. Auto mode will also rotate around between the green, yellow, and red lights. The Serial Monitor can be used for debugging and informing the developer of certain information, such as the board's IP address. Here, it also shows the board's current light that was selected.



InPrivate | Stoplight | 192.168.137.208/off

# Aaron's Class Projects

## Lab 2 - Stoplight Control

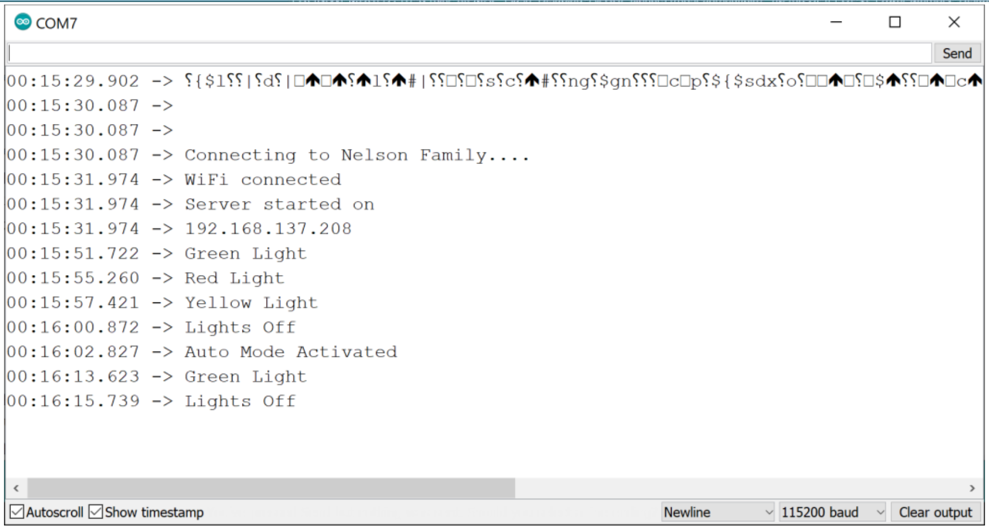
Choose the color of the stoplight or select "Auto" to have it cycle through each light. The light is currently off.

**OFF**

**Green** **Yellow** **Red**

**Auto**

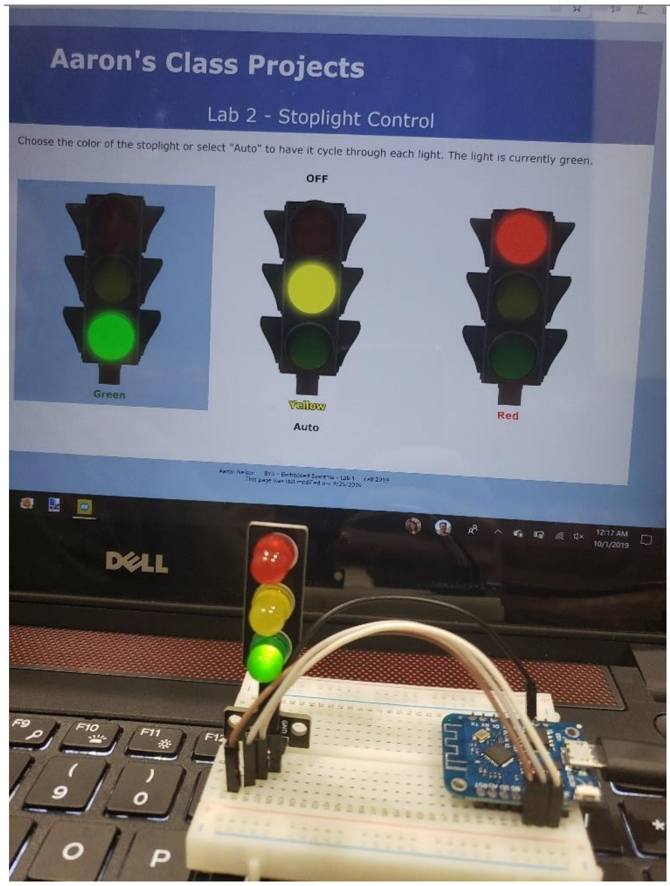
Aaron Nelson BYU - Embedded Systems - Lab 1 Fall 2019
This page was last modified on: 9/25/2019



COM7

```
00:15:29.902 -> ʕ{$1ʕʕ|ʕdʕ|□🡅□🡅ʕ🡅1ʕ🡅#|ʕʕ□ʕ□ʕsʕcʕ🡅#ʕʕngʕ$gnʕʕʕ□c□pʕ${$sdxʕoʕ□□🡅□ʕ□$🡅ʕʕ□🡅□c🡅
00:15:30.087 ->
00:15:30.087 ->
00:15:30.087 -> Connecting to Nelson Family....
00:15:31.974 -> WiFi connected
00:15:31.974 -> Server started on
00:15:31.974 -> 192.168.137.208
00:15:51.722 -> Green Light
00:15:55.260 -> Red Light
00:15:57.421 -> Yellow Light
00:16:00.872 -> Lights Off
00:16:02.827 -> Auto Mode Activated
00:16:13.623 -> Green Light
00:16:15.739 -> Lights Off
```

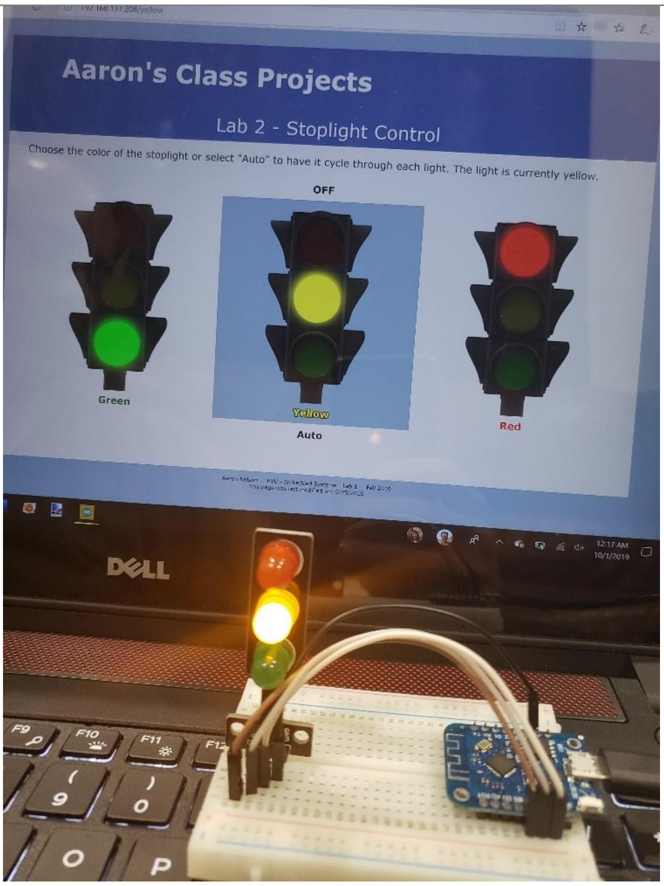
☑ Autoscroll ☑ Show timestamp | Newline | 115200 baud | Clear output



Aaron's Class Projects

Lab 2 - Stoplight Control

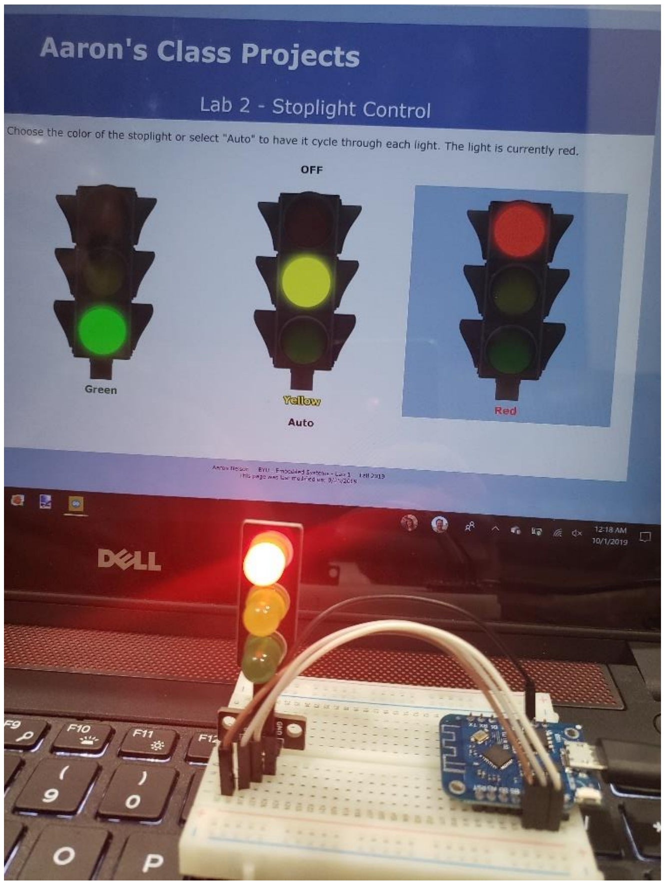
Choose the color of the stoplight or select "Auto" to have it cycle through each light. The light is currently green.



Aaron's Class Projects

Lab 2 - Stoplight Control

Choose the color of the stoplight or select "Auto" to have it cycle through each light. The light is currently yellow.

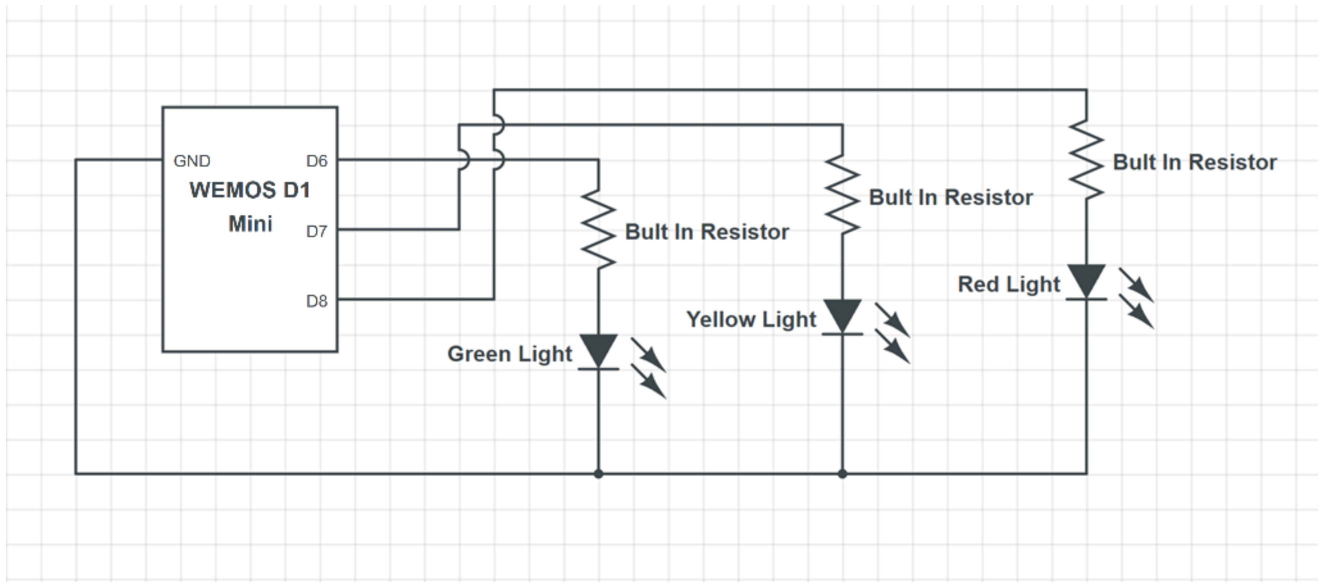


Aaron's Class Projects

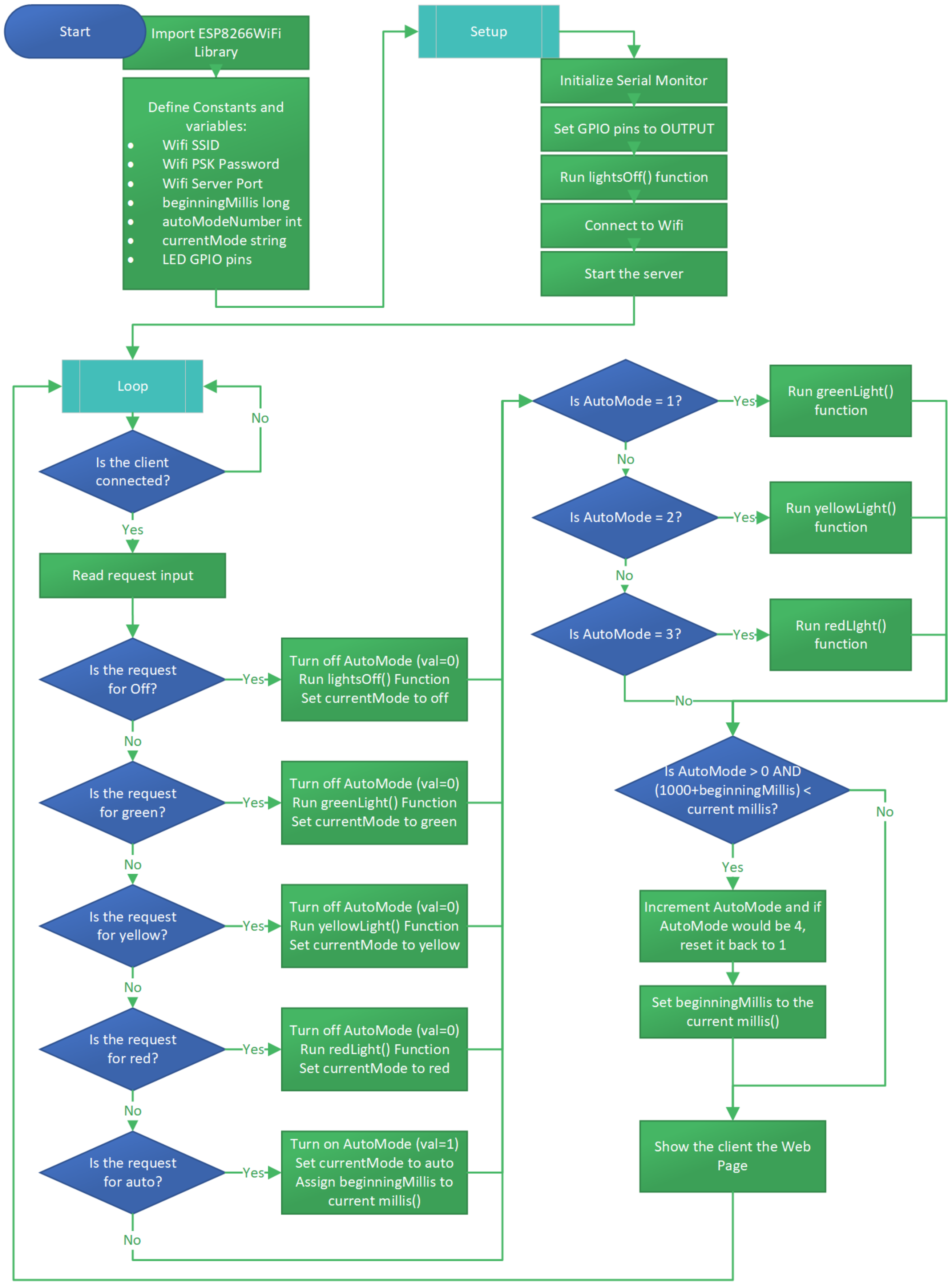
Lab 2 - Stoplight Control

Choose the color of the stoplight or select "Auto" to have it cycle through each light. The light is currently red.

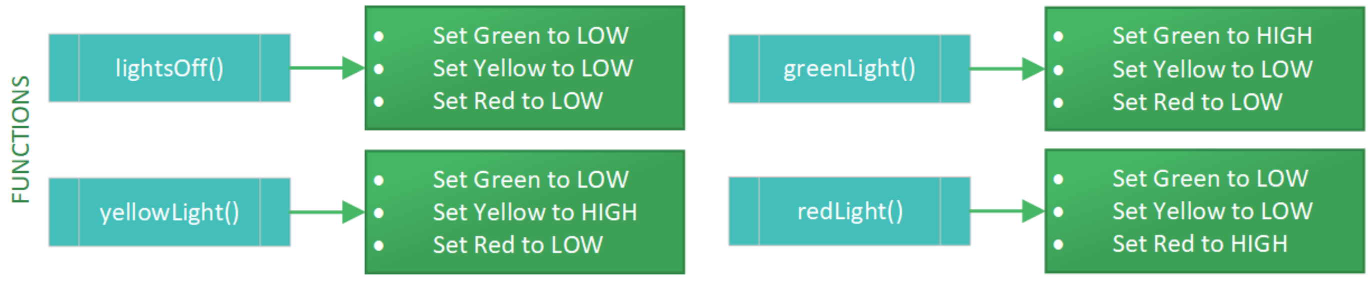
## Circuit Diagram



## System Flowchart


Start
Import ESP8266WiFi Library
Define Constants and variables:
Wifi SSID
Wifi PSK Password
Wifi Server Port
beginningMillis long
autoModeNumber int
currentMode string
LED GPIO pins
Setup
Initialize Serial Monitor
Set GPIO pins to OUTPUT
Run lightsOff() function
Connect to Wifi
Start the server
Loop
Is the client connected?
No
Yes
Read request input
Is the request for Off?
Yes
Turn off AutoMode (val=0) Run lightsOff() Function Set currentMode to off
No
Is the request for green?
Yes
Turn off AutoMode (val=0) Run greenLight() Function Set currentMode to green
No
Is the request for yellow?
Yes
Turn off AutoMode (val=0) Run yellowLight() Function Set currentMode to yellow
No
Is the request for red?
Yes
Turn off AutoMode (val=0) Run redLight() Function Set currentMode to red
No
Is the request for auto?
Yes
Turn on AutoMode (val=1) Set currentMode to auto Assign beginningMillis to current millis()
No
Is AutoMode = 1?
Yes
Run greenLight() function
No
Is AutoMode = 2?
Yes
Run yellowLight() function
No
Is AutoMode = 3?
Yes
Run redLIght() function
No
Is AutoMode > 0 AND (1000+beginningMillis) < current millis?
No
Yes
Increment AutoMode and if AutoMode would be 4, reset it back to 1
Set beginningMillis to the current millis()
Show the client the Web Page

## System Flowchart Extra Functions



## Code (available at https://github.com/AaronNelson95/IT441)

### Arduino Code

#### Lab_2_Wifi_Stoplight.ino

```
/*
 * Aaron Nelson
 * Lab 2 Stoplight Control
 * 9-25-19
 */
#include <ESP8266WiFi.h>            // This contains the libraries that allows the board
                                          to connect to wifi

/* The basic server setup and framework was obtained from the Arduino Examples-
    File -> Examples -> ESP8266WiFI -> WiFiManualWebServer */
#ifndef STASSID
#define STASSID "YOUR SSID HERE"      // Specify the name of your wifi
#define STAPSK  "YOUR PASSWORD HERE"         // Specify the password for that wifi
#endif
const char* ssid = STASSID;
const char* password = STAPSK;

WiFiServer server(80);              // This sets up the server to listen on port 80
                                          (http)

unsigned long beginningMillis;      // A variable to hold our current time (used for
                                          auto mode to keep track of how much time has
                                          passed)
int autoModeNumber;                 // A variable to keep track of what color the light
                                          is in during auto mode

int green = 12;                     // Pin D6 is used for green. If using a different
                                          pin, specify that here
int yellow = 13;                    // Pin D7 is used for yellow. If using a different
                                          pin, specify that here
int red = 15;                       // Pin D8 is used for red. If using a different pin,
                                          specify that here
String currentMode = "off";         // CurrentMode is used to inform the user with
                                          visible text what light is on. It also
                                          highlights the table cell containing the mode




void setup() {                      // This runs when the board is first turned on
  Serial.begin(115200);             // This allows serial information when connected to
                                          a computer (in here, this just tells what
                                          light was requested to turn on)
  pinMode(green, OUTPUT);           // Prepares the pin connected to the green light for
                                          output
  pinMode(yellow, OUTPUT);          // Prepares the pin connected to the yellow light
                                          for output
  pinMode(red, OUTPUT);             // Prepares the pin connected to the red light for
                                          output
  autoModeNumber = 0;               // Sets the auto mode number to 0 (meaning it is off
                                          and not supposed to cycle)
  lightsOff();                      // Runs the function (at the bottom of this script)
                                          that turns all pin lights to LOW/OFF
  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");   // This is shown on the serial if connected to a
                                        computer
  Serial.print(ssid);               // It displays the wifi it is trying to connect to

  WiFi.mode(WIFI_STA);              // It sets the wifis mode to "Station" (rather than
                                        "AP" which is the Access Point mode)
  WiFi.begin(ssid, password);       // Attempts to connect to wifi using provided
                                        credentials

  while (WiFi.status() != WL_CONNECTED) {     // While the wifi is not connected yet:
    delay(500);                     // every half second,
    Serial.print(".");              // it shows a dot on the Serial Monitor to show it
                                        is still trying to connect
  }
  Serial.println();
  Serial.println("WiFi connected"); // When it does connect, show that there was success

  // Start the server
  server.begin();
  Serial.println("Server started on ");
  Serial.println(WiFi.localIP());   // Show the IP address the board has now that it is
                                        connected and broadcasting

}




void loop() {                       // This constantly cycles through, accepting
                                        clients, showing them a webpage, and
                                        responding to their inputs
  // Check if a client is connected
  WiFiClient client = server.available();     // A client object is created for
                                                  connections
  if (client) {                     // This waits until a client sends some data (such
                                        as by clicking a light option)
    while (!client.available()) {   // The client hasn't sent info so wait for it to do
                                        so
      delay(10);
    }
  }
  client.setTimeout(100);           // This sets the maximum ms to wait for data to come
                                        through. Each page the client will go has a
                                        short url, so a small amount is fine

  String req = client.readStringUntil('\r');  // Reads the first line of the request
                                                  (the URL string) made by the client
  if (req.indexOf("/off") != -1) {  // If the request is for the "/off" page:
    autoModeNumber = 0;             // Turn off auto mode
    lightsOff();                    // This function turns off all of the lights
    currentMode = "off";            // Show that the current state is off to the client
  }
  if (req.indexOf("/green") != -1) {          // If the request is for the "/green"
                                                  page:
    autoModeNumber = 0;             // Turn off auto mode
    Serial.println("Green Light");  // Print to the computer Serial Monitor that the
                                          light is green
    greenLight();                   // This function turns on only the green light
    currentMode = "green";          // Show that the current state is green to the
                                          client
  }
  if (req.indexOf("/yellow") != -1) {          // If the request is for the "/yellow"
                                                        page:
    autoModeNumber = 0;             // Turn off auto mode
    Serial.println("Yellow Light"); // Print to the computer Serial Monitor that the
                                          light is yellow
    yellowLight();                  // This function turns on only the yellow light
    currentMode = "yellow";         // Show that the current state is yellow to the
                                          client
  }
  if (req.indexOf("/red") != -1) {  // If the request is for the "/red" page:
    autoModeNumber = 0;             // Turn off auto mode
    Serial.println("Red Light");    // Print to the computer Serial Monitor that the
                                          light is red
    redLight();                     // This function turns on only the red light
    currentMode = "red";            // Show that the current state is red to the client
  }
  if (req.indexOf("/auto") != -1) {            // If the request is for the "/auto" page:
    autoModeNumber = autoModeNumber + 1;       // Increment the auto number. If it was
                                                        off (0) turn it on to 1 (green)
    if (autoModeNumber == 4) {      // If the light has cycled through and was in the
                                          red state:
      autoModeNumber = 1;           // Set it back to the green state
    }
    Serial.println("Auto Mode Activated");     // Print to the computer Serial Monitor
                                                  that the lights are now in auto mode
    currentMode = "in auto mode";   // Show that the current state is in auto mode to
                                          the client
    beginningMillis = millis();     // Take the current time that the processor has been
                                          on and set that as the beginning amount (which
                                          will be compared to see how much time passes)
  }




  if (autoModeNumber == 1) {        // The first cycle of the auto mode is to show the
                                          green light
    greenLight();                   // Call the green function
  }
  if (autoModeNumber == 2) {        // The second cycle of the auto mode is to show the
                                          yellow light
    yellowLight();                  // Call the yellow function
  }
  if (autoModeNumber == 3) {        // The third cycle of the auto mode is to show the
                                          red light
    redLight();                     // Call the red function
  }

  if (autoModeNumber > 0) {         // Auto mode is currently on so check the following:
    if ((1000 + beginningMillis) < millis()) {  // If there has been 1 second passed
                                                  between now and the last light change:
      autoModeNumber = autoModeNumber + 1;      // Increment the color mode that auto
                                                        mode is currently on
      if (autoModeNumber == 4) {     // If we were just in the red mode,
        autoModeNumber = 1;          // Set the next mode back to green
      }
      beginningMillis = millis();    // This "resets" our timer to be the new starting
                                        point for the next period
      delay(5);                      // A short delay so the script processes the changes
    }
  }






  /* This is the webpage that is hosted on port 80 (HTTP) and is shown to the client */
  /* Thanks to Jonathan Black for writing and providing a script that converts HTML to
      the Arduino Client.Print format */
  client.println("<!doctype html>");
  client.println("<html>");
  client.println("  <head>");
  client.println("    <!--Aaron Nelson");
  client.println("    Last Updated: 9-25-19 -->");
  client.println("    <title>Stoplight</title>");
  // Styles are embedded and specified within the HTML page
  client.println("    <style>");
  client.println("      /* UNIVERSAL STYLES */");
  client.println("      /* Overall Page Structure */");
  client.println("      body {");
  client.println("        background-color: #a4b9d5; ");
  client.println("        color:#000000; ");
  client.println("        font-family: Verdana, Arial, serif; ");
  client.println("      }");
  client.println("      #wrapper {");
  client.println("        background-color: #F5F5F5; ");
  client.println("        width: 90%; ");
  client.println("        margin-top: 25px;");
  client.println("        margin-left: auto; ");
  client.println("        margin-right: auto; ");
  client.println("        min-width: 600px; ");
  client.println("        max-width: 1480px; ");
  client.println("      }");
  client.println("");
  client.println("");
  client.println("      /* Header Appearance (Site Title) */");
  client.println("      header {");
  client.println("        text-align: left;");
  client.println("        background-color: #4972aa;");
  client.println("        height: 100px;");
  client.println("      }");
  client.println("      header h1 {");
  client.println("        padding-left: 8%;");
  client.println("        padding-top: 0.6em;");
  client.println("        color: white;");
  client.println("        font-size: 2.5em;");
  client.println("      }");
  client.println("");
  client.println("");
  client.println("      #pageTitle {");
  client.println("        font-size: 1.8em;");
  client.println("        font-style: bold;");
  client.println("          background-color: #4972aa;");
  client.println("          color: white;");
  client.println("          padding: 0.3em;");
  client.println("          text-align: center;");
  client.println("        }");
  client.println("        #main {");
  client.println("          padding-left: 2em;");
  client.println("          padding-right: 2em;");
  client.println("          display: block;");
  client.println("        }");
  client.println("");
  client.println("");
  client.println("        /* Link Colors */");
  client.println("        a:link {color: #000000; text-decoration: none; }");
  client.println("        a:visited {color: #000000; text-decoration: none; }");
  client.println("        a:active {color: #000000; text-decoration: none; }");
  client.println("");
  client.println("");
  client.println("        /* Footer Format */");
  client.println("        footer {");
  client.println("          text-align:center;");
  client.println("          background-color: #a4b9d5;");
  client.println("          font-size: .55em;");
  client.println("          padding: 1em; ");
  client.println("        }");
  client.println("");
  client.println("");
  client.println("");
  client.println("");
  client.println("        /* SPECIFIC WEBPAGE STYELS */");
  client.println("        /* LAB 2 */");
  client.println("        /* Spotlight Table */");
  client.println("        .SpotlightColors {");
  client.println("          width: 100%; ");
  client.println("          border: 0; ");
  client.println("          border-collapse: collapse; ");
  client.println("          margin-bottom: 3em; ");
  client.println("        }");
  client.println("        .SpotlightColors td, th {");
  client.println("          border: 0; ");
  client.println("          padding: 5px; ");
  client.println("          text-align: center; ");
  client.println("        }");
  client.println("        h4 {");
  client.println("          margin-top: 3px;");
  client.println("          margin-bottom: 3px;");
  client.println("        }");
  client.println("        /* Spotlight Table Text */");
  client.println("        .green {");
  client.println("          color: green;");
  client.println("        }");
  client.println("        .yellow {");
  client.println("          color: yellow;");
  client.println("          /* Yellow text is not readable so it needs a black
                                                              outline*/");
  client.println("          text-shadow:");
  client.println("            -1px -1px 0 #000,");
  client.println("            1px -1px 0 #000,");
  client.println("            -1px 1px 0 #000,");
```

```
client.println("          1px 1px 0 #000;");
client.println("      }");
client.println("      .red {");
client.println("        color: red;");
client.println("      }");
client.println("      .auto, .off {");
client.println("        text-align: center;");
client.println("      }");
client.println("      #green:hover, #yellow:hover, #red:hover, #auto:hover, #off:hover
                                                                                  {");
client.println("        /* Change the color of what is being selected */");
client.println("        background-color: #ddd;");
client.println("      }");
client.println("      .running {");
client.println("        background-color: #a4b9d5;");
client.println("      }");
client.println("      .autoOn {");
client.println("        background-color: #a4b9d5;");
client.println("      }");
client.println("      .colorPick {");
client.println("        width: 50px;");
client.println("        text-align: right;");
client.println("      }");
client.println("          ");
client.println("    ");
client.println("    </style>    ");
// End of the styles and beginning of the HTML page
client.println("  </head>");
client.println("  <body>");
client.println("    <div id=\"wrapper\">");
client.println("      <header>");
client.println("        <h1>Aaron's Class Projects</h1>");
client.println("      </header>");
client.println("      <div id=\"pageTitle\">Lab 2 - Stoplight Control</div>");
client.println("    ");
client.println("      <div id=\"main\"> ");
client.println("        <p>Choose the color of the stoplight or select \"Auto\" to
                      have it cycle through each light. The light is currently ");
// This will show in plain text what mode the light is set to show
client.print(currentMode);
client.println(".</p>");
client.println("        <table class=SpotlightColors>");
client.println("          <tr>");
client.print("            <td id=\"off\" colspan=3 ");
// If the current mode is off, the table cells are highlighted with CSS by giving it a
          special class
if (currentMode == "off"){
  client.print("class=\"running\" ");
}

client.print("><a href='http://");
// This hyperlink will include the current IP address the server is on
client.print(WiFi.localIP());
client.print("/off'>");

client.println("<h4 class=\"off\">OFF</h4></a></td>");
client.println("          </tr>");
client.println("          <tr>");
```

```
  client.println("            <!-- Stoplight images from 
https://www.freemanhealth.com/blog/the-behavior-traffic-light-colors-of-the-light -->");
  client.print("            <td id=\"green\" ");
  // If the current mode is green, the table cells are highlighted with CSS by giving it 
          a special class
  if (currentMode == "green"){
    client.print("class=\"running\" ");
  }

  client.print("><a href='http://");
  // This hyperlink will include the current IP address the server is on
  client.print(WiFi.localIP());
  client.print("/green'>");

  client.println("<img 
src=\"https://raw.githubusercontent.com/AaronNelson95/IT441/master/Lab%201/Images/GreenL
ight.png\"><br><h4 class=\"green\">Green</h4></a></td>");
  client.println("");
  client.print("            <td id=\"yellow\" ");
  // If the current mode is yellow, the table cells are highlighted with CSS by giving 
          it a special class
  if (currentMode == "yellow"){
    client.print("class=\"running\" ");
  }

  client.print("><a href='http://");
  // This hyperlink will include the current IP address the server is on
  client.print(WiFi.localIP());
  client.print("/yellow'>");

  client.println("<img 
src=\"https://raw.githubusercontent.com/AaronNelson95/IT441/master/Lab%201/Images/Yellow
Light.png\"><br><h4 class=\"yellow\">Yellow</h4></a></td>");
  client.println("");
  client.print("            <td id=\"red\" ");
  // If the current mode is red, the table cells are highlighted with CSS by giving it a 
          special class
  if (currentMode == "red"){
    client.print("class=\"running\" ");
  }
  client.print("><a href='http://");
  // This hyperlink will include the current IP address the server is on
  client.print(WiFi.localIP());
  client.print("/red'>");

  client.println("<img 
src=\"https://raw.githubusercontent.com/AaronNelson95/IT441/master/Lab%201/Images/RedLig
ht.png\"><br><h4 class=\"red\">Red</h4></a></td>");
  client.println("          </tr>");
  client.println("          <tr>");
  client.print("            <td id=\"auto\" colspan=3 ");
  // If the current mode is on auto, the table cells are highlighted with CSS by giving 
          it a special class
  if (currentMode == "in auto mode"){
    client.print("class=\"running\" ");
  }

  client.print("><a href='http://");
  // This hyperlink will include the current IP address the server is on
  client.print(WiFi.localIP());
  client.print("/auto'>");

  client.println("<h4 class=\"auto\">Auto</h4></a></td>");
  client.println("          </tr>");
  client.println("        </table>");
  client.println("        ");
  client.println("        ");
  client.println("        ");
  client.println("      ");
  client.println("      </div>");
  client.println("    ");
  client.println("      <footer>Aaron Nelson &emsp; BYU - Embedded Systems - Lab 1
            &emsp; Fall 2019");
  client.println("      <br>This page was last modified on: 9/25/2019");
  client.println("      </footer>");
  client.println("    </div>");
  client.println("  </body>");
  client.println("</html>");
}





/* These are the functions that are called when a client sends a request.
    They only turn on the light they are supposed to according to the pin number
    specified at the beginning of the script */
void lightsOff() {
  digitalWrite(green, LOW);
  digitalWrite(yellow, LOW);
  digitalWrite(red, LOW);
  Serial.println("Lights Off");
}

void greenLight() {
  digitalWrite(green, HIGH);
  digitalWrite(yellow, LOW);
  digitalWrite(red, LOW);
}

void yellowLight() {
  digitalWrite(green, LOW);
  digitalWrite(yellow, HIGH);
  digitalWrite(red, LOW);
}

void redLight() {
  digitalWrite(green, LOW);
  digitalWrite(yellow, LOW);
  digitalWrite(red, HIGH);
}
```