# IoT Based Garage Door Opener: HTML Coding for Web Page

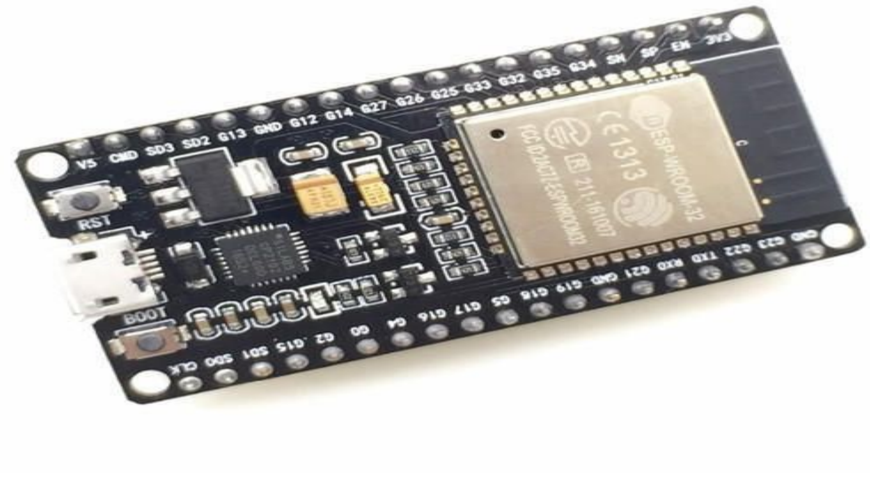
Dawinder Kaur
(C0765505)
Group 4

# Content:

- Introduction of ESP32 Wifi Module
- Features of ESP32
- Pinout of ESP32
- Introduction to HTML Coding
- Versions of HTML
- Example of HTML Document
- What is HTML element
- HTML Page Structure
- Introduction of CSS3
- Features of CSS3
- CSS3 Example
- Advantages of CSS3
- References

# Introduction of ESP32 Wifi Module

- ESP32 is a series of low-cost, low-power system on a chip microcontrollers with integrated Wi-Fi and dual-mode Bluetooth.
- ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process.
- It is a successor to the ESP8266 microcontroller.


V5 GND SD3 SD2 G13 GND G12 G14 G27 G26 G25 G33 G32 G35 G34 SN SP EN 3V3
RST
BOOT
CLK SD0 SD1 G15 G2 G0 G4 G16 G17 G5 G18 G19 GND G21 RXD TXD G22 G23 GND
ESP-WROOM-32
CE 1313

# Features of  ESP32 Wifi module

**Processors**

- CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, operating at 160 or 240 MHz and performing at up to 600 DMIPS
- Ultra low power (ULP) co-processor

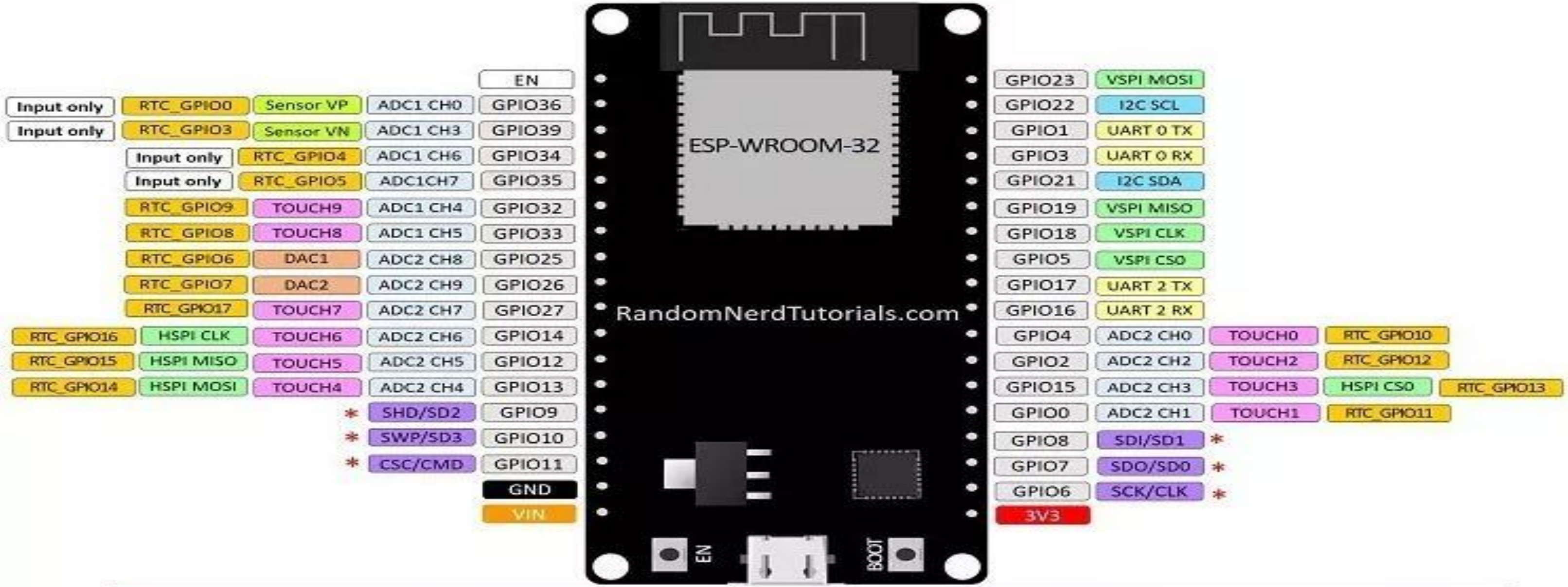
**Peripheral interfaces**

- 12-bit SAR ADC up to 18 channels
- 2 × 8-bit DACs
- 10 × touch sensors
- 4 × SPI
- 2 × $I^2S$ and 2 x $I^2C$ interfaces
- 3 × UART
- MMC/eMMC host controller
- SDIO/SPI slave controller

# Features of  ESP32 Wifi module

- Ethernet MAC interface with dedicated DMA and IEEE 1588 Precision Time Protocol support
- CAN bus 2.0
- Infrared remote controller (TX/RX, up to 8 channels)
- Motor PWM
- LED PWM (up to 16 channels)
- Hall effect sensor
- Ultra low power analog preamplifier

# Pin Diagram of  ESP32 Wifi module


ESP-WROOM-32
RandomNerdTutorials.com
EN
Input only RTC_GPIO0 Sensor VP ADC1 CH0 GPIO36
Input only RTC_GPIO3 Sensor VN ADC1 CH3 GPIO39
Input only RTC_GPIO4 ADC1 CH6 GPIO34
Input only RTC_GPIO5 ADC1CH7 GPIO35
RTC_GPIO9 TOUCH9 ADC1 CH4 GPIO32
RTC_GPIO8 TOUCH8 ADC1 CH5 GPIO33
RTC_GPIO6 DAC1 ADC2 CH8 GPIO25
RTC_GPIO7 DAC2 ADC2 CH9 GPIO26
RTC_GPIO17 TOUCH7 ADC2 CH7 GPIO27
RTC_GPIO16 HSPI CLK TOUCH6 ADC2 CH6 GPIO14
RTC_GPIO15 HSPI MISO TOUCH5 ADC2 CH5 GPIO12
RTC_GPIO14 HSPI MOSI TOUCH4 ADC2 CH4 GPIO13
* SHD/SD2 GPIO9
* SWP/SD3 GPIO10
* CSC/CMD GPIO11
GND
VIN
GPIO23 VSPI MOSI
GPIO22 I2C SCL
GPIO1 UART 0 TX
GPIO3 UART 0 RX
GPIO21 I2C SDA
GPIO19 VSPI MISO
GPIO18 VSPI CLK
GPIO5 VSPI CS0
GPIO17 UART 2 TX
GPIO16 UART 2 RX
GPIO4 ADC2 CH0 TOUCH0 RTC_GPIO10
GPIO2 ADC2 CH2 TOUCH2 RTC_GPIO12
GPIO15 ADC2 CH3 TOUCH3 HSPI CS0 RTC_GPIO13
GPIO0 ADC2 CH1 TOUCH1 RTC_GPIO11
GPIO8 SDI/SD1 *
GPIO7 SDO/SD0 *
GPIO6 SCK/CLK *
3V3
EN
BOOT

* Pins SCK/CLK, SDO/SD0, SDI/SD1, SHD/SD2, SWP/SD3 and SCS/CMD, namely, GPIO6 to GPIO11 are connected to the integrated SPI flash integrated on ESP-WROOM-32 and are not recommended for other uses.

# Introduction to HTML Coding

HTML is the standard markup language for creating Web pages.

- HTML stands for Hyper Text Markup Language.
- It is a language that we use to create Web pages.
- It describes the structure of Web pages.
- It consists of a series of elements.
- HTML elements tell the browser how to display the content.
- HTML elements label pieces of content such as “this is a heading”, “this is a paragraph”, “this is a link”, etc.

# Introduction to HTML Coding

HTML Version:

- HTML 1.0
- HTML 2
- HTML 3.2
- HTML 4.01
- HTML 5

# Introduction to HTML Coding

HTML 5 is the latest version of HTML language. It was started to use in 2014 and it came up with lots of HTML tags support. It provided support for new form elements like input elements of different types, geolocations, support tags, etc.

The new tags that has been added are Email, password, audio tag, semantic tags and section tag.

# Introduction to HTML Coding

```
<!DOCTYPE html>           //this declares that this is an HTML5 document
<html>                    //this is the root element of HTML page
<head>                    //this contains meta information about HTML page
<title>Page Title</title>     //this element is shown in the browser’s title
</head>
<body>                //element defines the document’s body and is the
                      container for all visible contents like headings,
                      paragraphs, images, hyperlinks, etc

<h1>My First Heading</h1>     //defines a large heading

<p>My first paragraph.</p>    //defines a paragraph

</body>
</html>
```

# Introduction to HTML Coding

### HTML Element:

An HTML is defined by a start tag, some content, and an end tag:

```
<tagname>Content goes here...</tagname>
```

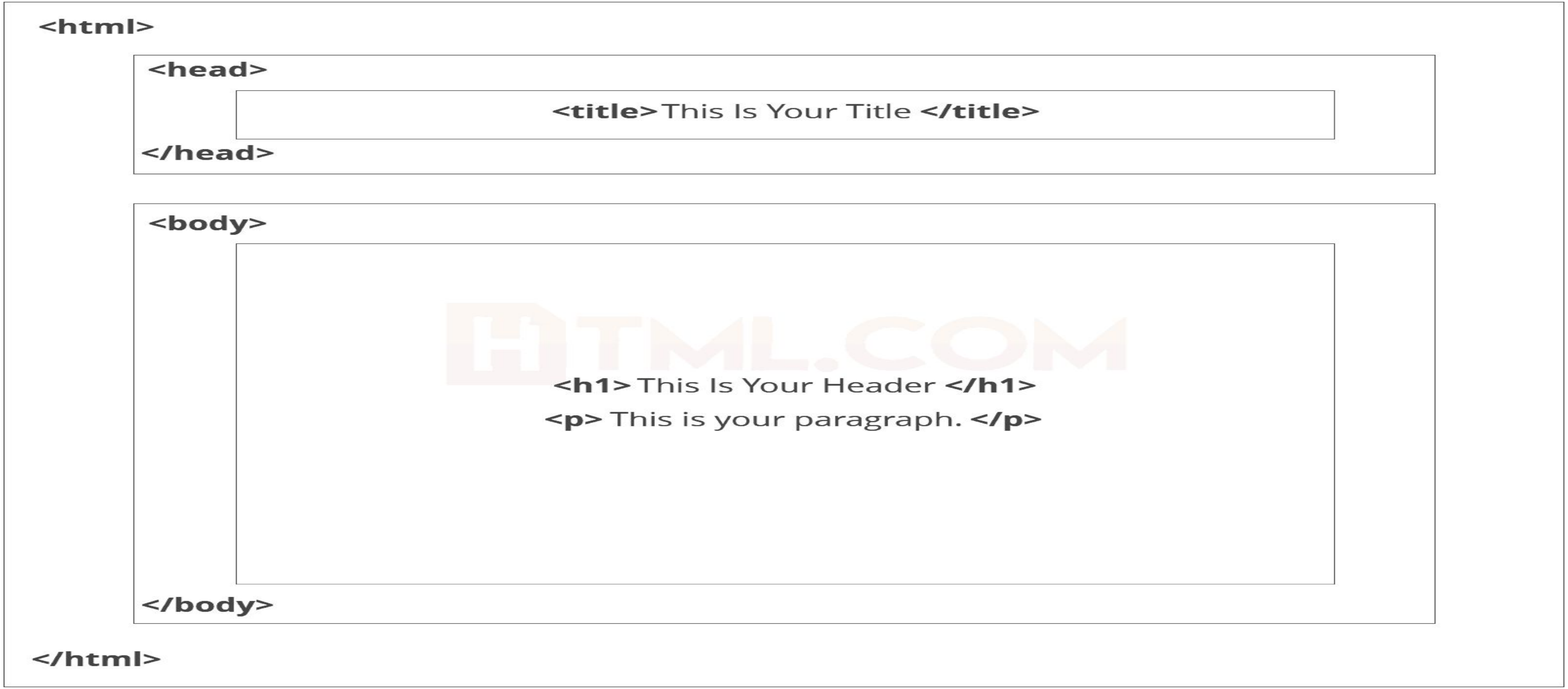
The HTML element is everything from the start tag to the the end tag:

```
 <h1>My First Heading</h1>
 <p>My first paragraph.</p>
```

# Introduction to HTML Coding

| Start Tag | Element Content | End Tag |
|---|---|---|
| <h1> | My First Heading | </h1> |
| <p> | My first paragraph | </p> |
| <br> | none | none |

# Introduction to HTML Coding



```
<html>
<head>
<title>This Is Your Title </title>
</head>
<body>
<h1>This Is Your Header </h1>
<p> This is your paragraph. </p>
</body>
</html>
```

# Introduction CSS3(Cascading Style Sheets):

Cascading Style Sheets (CSS) is a language that is used to illustrate the look, style, and format of a document written in any markup language.

In simple words, it is used to style and organize the layout of Web pages.

CSS3 is the latest version of an earlier CSS version, CSS2.

A significant change in CSS3 in comparison to CSS2 is the introduction of modules.

The benefit of this functionality is that it allows the specification to be finalized and accept faster, as segments are finalized and accepted in portions.

# Introduction CSS3(Cascading Style Sheets):

Some of the key modules of CSS3 are:

- Box model
- Image values and replaced content
- Text effects
- Selectors
- Backgrounds and borders
- Animations
- User interface (UI)
- Multiple column layouts
- 2D/3D transformations.

# Introduction CSS3(Cascading Style Sheets):

```
body {
  background-color: lightblue;
}

h1 {                         //<h1> This is a heading
  color: white;
  text-align: center;
}

p {                          //<p> This is a paragraph
  font-family: verdana;
  font-size: 20px;
}
```

# Introduction CSS3(Cascading Style Sheets):

Advantages of CSS3

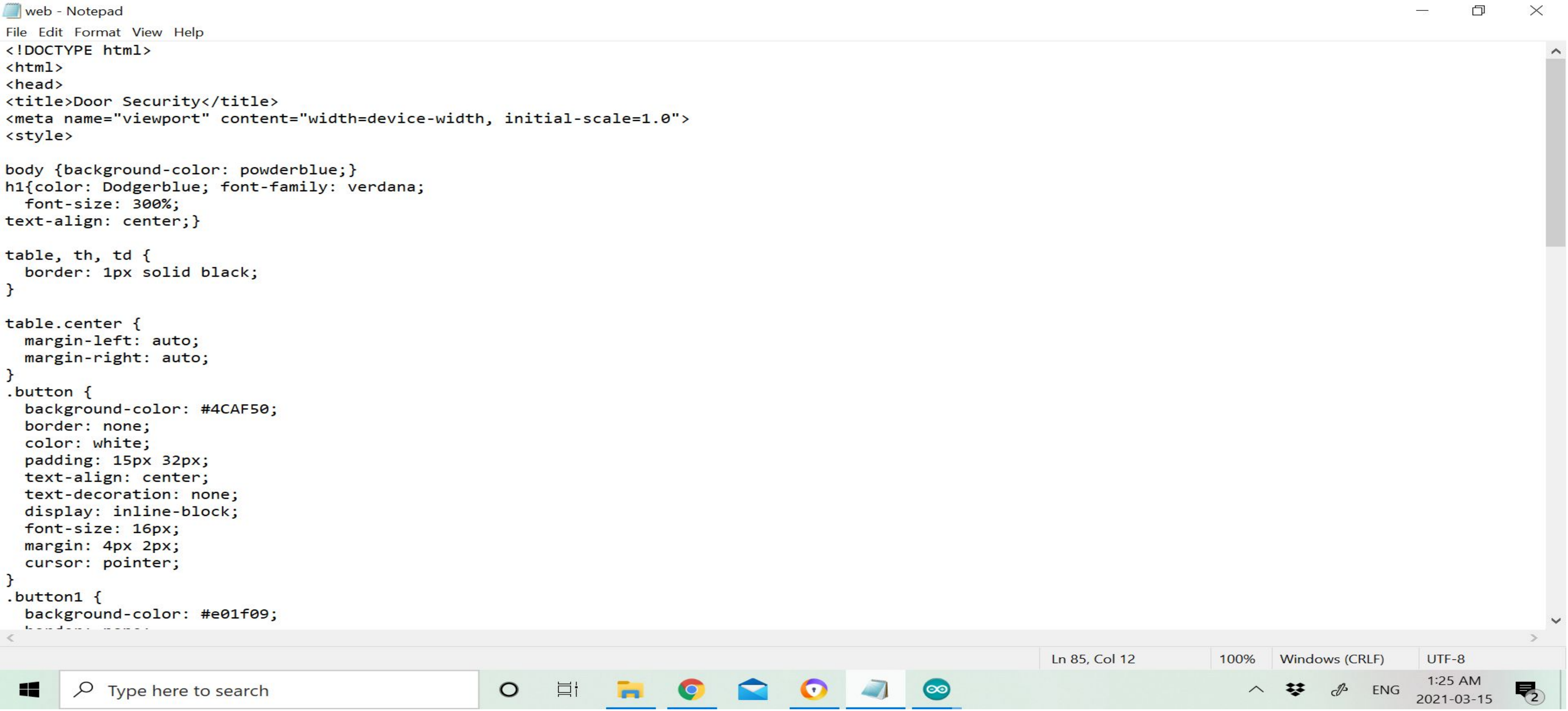
- CSS3 provides a consistent and precise positioning of navigable elements.
- It is easy to customize a web page as it can be done by merely altering a modular file.
- Graphics are easier in CSS3, thus making it easy to make the site appealing.
- It permits online videos to be seen without using third-party plug-ins.
- CSS3 is economical, time-saving, and most browsers support it.

# HTML Coding on Notepad:

HTML Programming on Notepad:

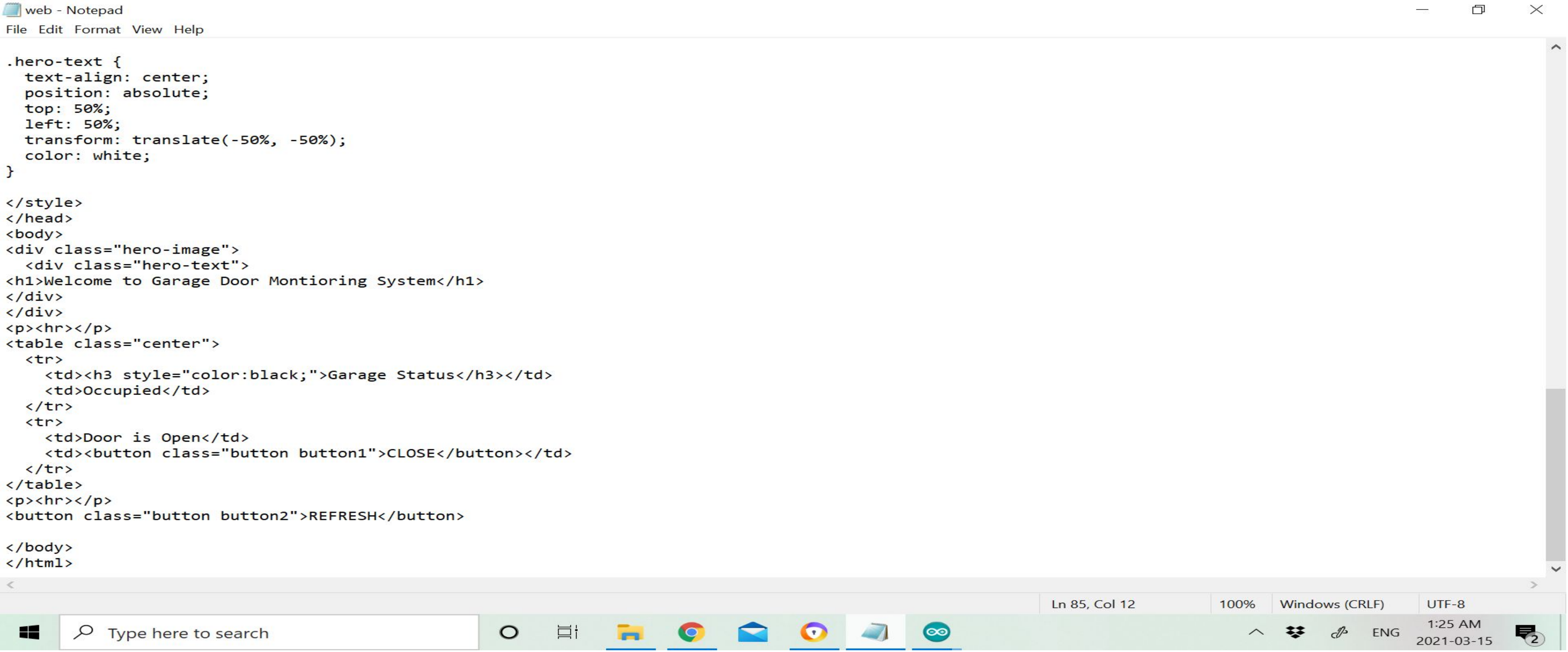
1. Open Notepad
2. Write HTML/CSS3 code
3. Save the file named as web.html (with .html extension so that it can run on browser).
4. Go to the location where u saved the file.
5. Click on file and choose open with chrome browser
6. Web page will appear on web browser.

# HTML Coding on Notepad:



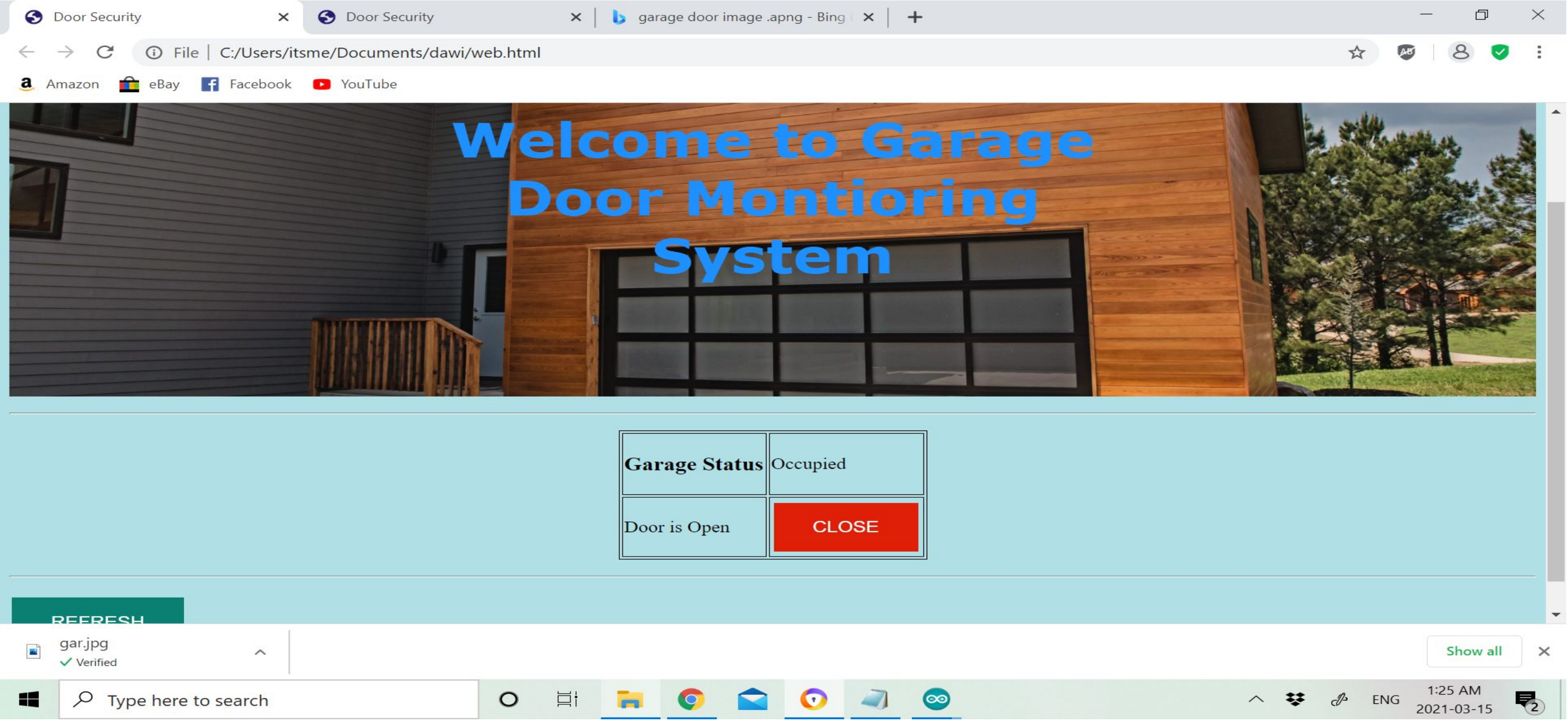
```
<!DOCTYPE html>
<html>
<head>
<title>Door Security</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>

body {background-color: powderblue;}
h1{color: Dodgerblue; font-family: verdana;
  font-size: 300%;
text-align: center;}

table, th, td {
  border: 1px solid black;
}

table.center {
  margin-left: auto;
  margin-right: auto;
}
.button {
  background-color: #4CAF50;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}
.button1 {
  background-color: #e01f09;
  border: none;
```

# HTML Coding on Notepad:



```
.hero-text {
  text-align: center;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  color: white;
}

</style>
</head>
<body>
<div class="hero-image">
  <div class="hero-text">
<h1>Welcome to Garage Door Montioring System</h1>
</div>
</div>
<p><hr></p>
<table class="center">
  <tr>
    <td><h3 style="color:black;">Garage Status</h3></td>
    <td>Occupied</td>
  </tr>
  <tr>
    <td>Door is Open</td>
    <td><button class="button button1">CLOSE</button></td>
  </tr>
</table>
<p><hr></p>
<button class="button button2">REFRESH</button>

</body>
</html>
```

# HTML Coding on Notepad: Output



Welcome to Garage Door Montioring System

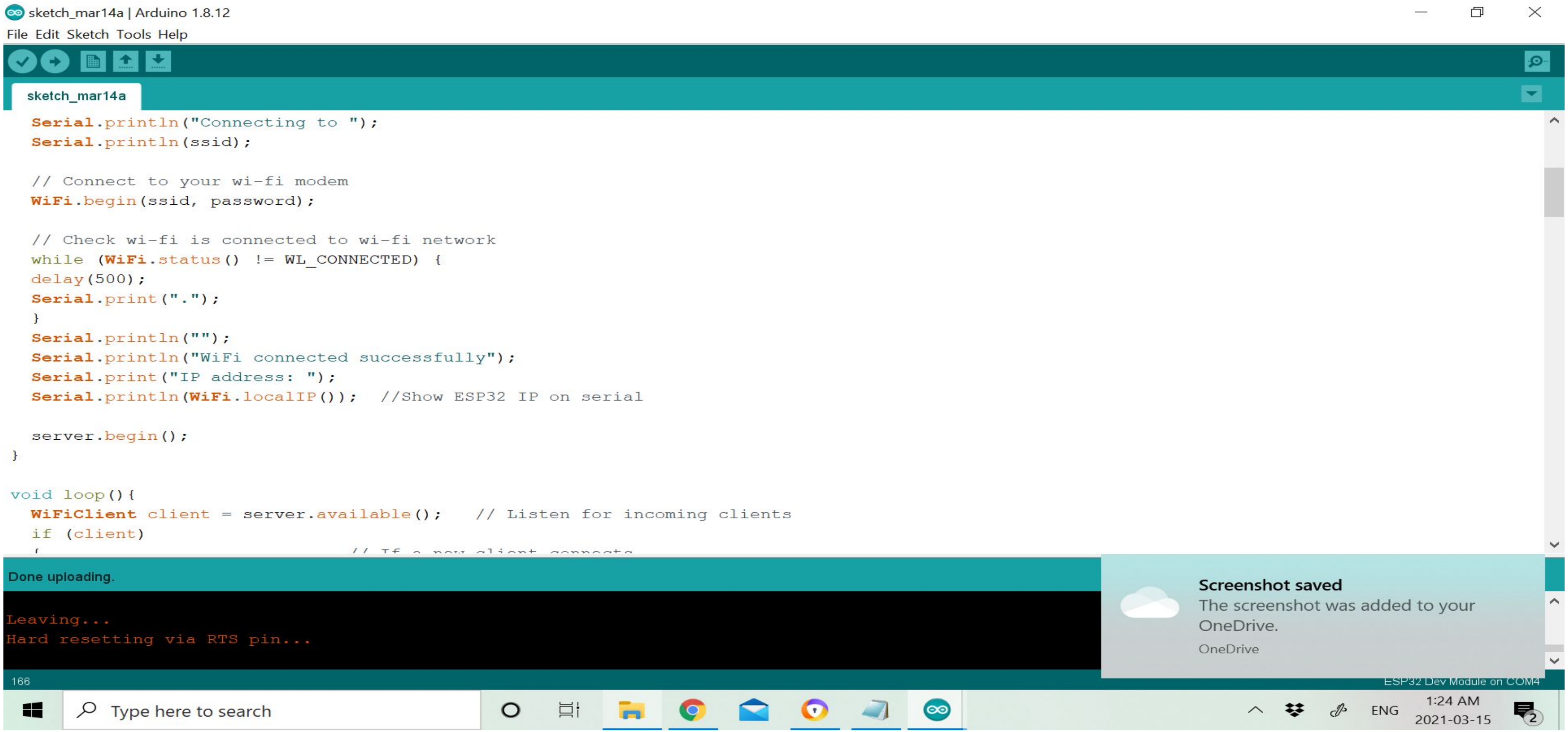
| | |
|---|---|
| **Garage Status** | Occupied |
| Door is Open | CLOSE |

# HTML coding on ESP32:

HTML coding on ESP32:

- Open arduino ide
- Write arduino ide code for esp32 wifi connectivity and in void loop{ } area add the html code using command “client.print {html code here}”
- Save the file
- Compile it and execute it by connecting ESP32 with system.
- Output will appear in serial monitor

# HTML coding on ESP32:



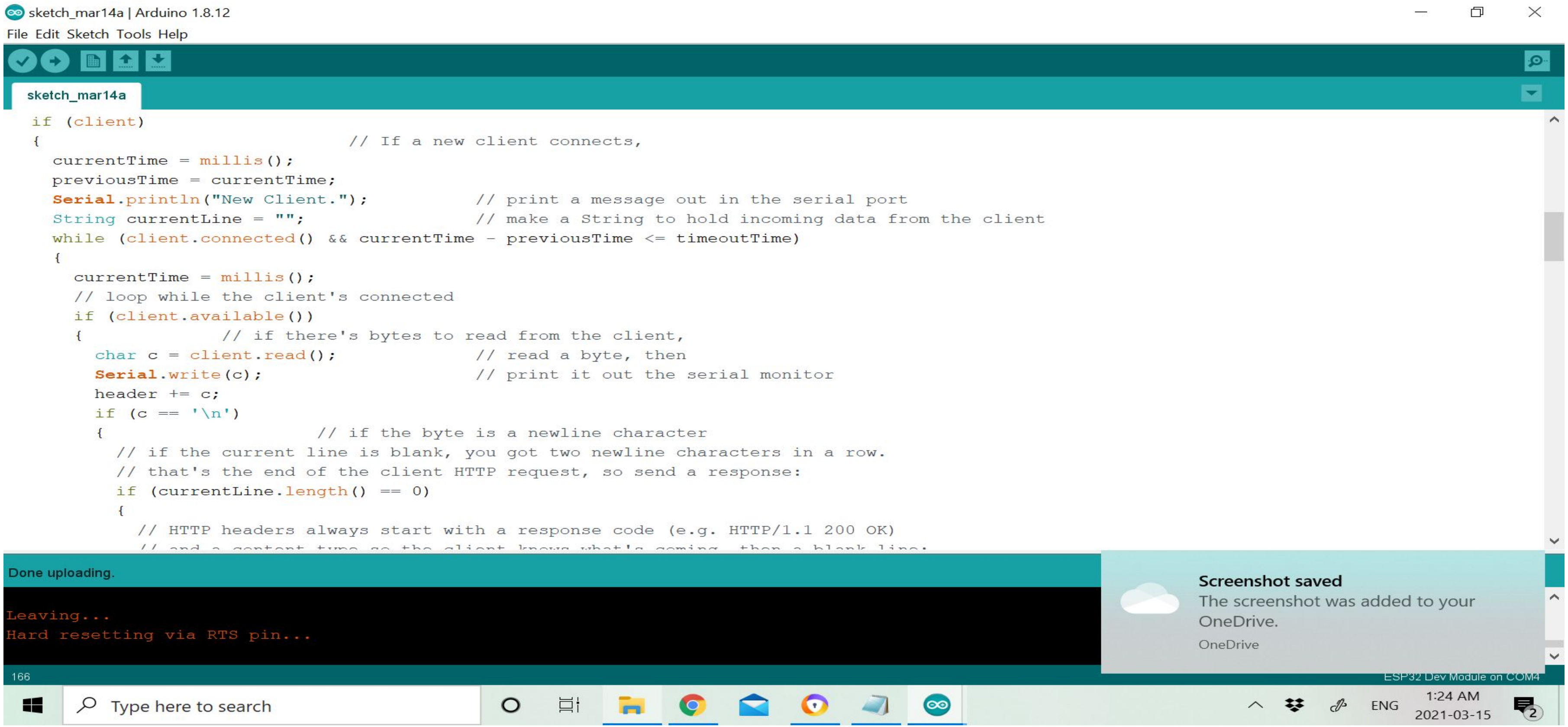
```
  Serial.println("Connecting to ");
  Serial.println(ssid);

  // Connect to your wi-fi modem
  WiFi.begin(ssid, password);

  // Check wi-fi is connected to wi-fi network
  while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected successfully");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());  //Show ESP32 IP on serial

  server.begin();
}

void loop(){
  WiFiClient client = server.available();   // Listen for incoming clients
  if (client)
  {                          // If a new client connects
```

Done uploading.

```
Leaving...
Hard resetting via RTS pin...
```

Screenshot saved
The screenshot was added to your OneDrive.
OneDrive

# HTML coding on ESP32:



```
if (client)
{                             // If a new client connects,
  currentTime = millis();
  previousTime = currentTime;
  Serial.println("New Client.");          // print a message out in the serial port
  String currentLine = "";                // make a String to hold incoming data from the client
  while (client.connected() && currentTime - previousTime <= timeoutTime)
  {
    currentTime = millis();
    // loop while the client's connected
    if (client.available())
    {             // if there's bytes to read from the client,
      char c = client.read();             // read a byte, then
      Serial.write(c);                    // print it out the serial monitor
      header += c;
      if (c == '\n')
      {                    // if the byte is a newline character
        // if the current line is blank, you got two newline characters in a row.
        // that's the end of the client HTTP request, so send a response:
        if (currentLine.length() == 0)
        {
          // HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK)
          // and a content-type so the client knows what's coming, then a blank line:
```

Done uploading.

```
Leaving...
Hard resetting via RTS pin...
```

166

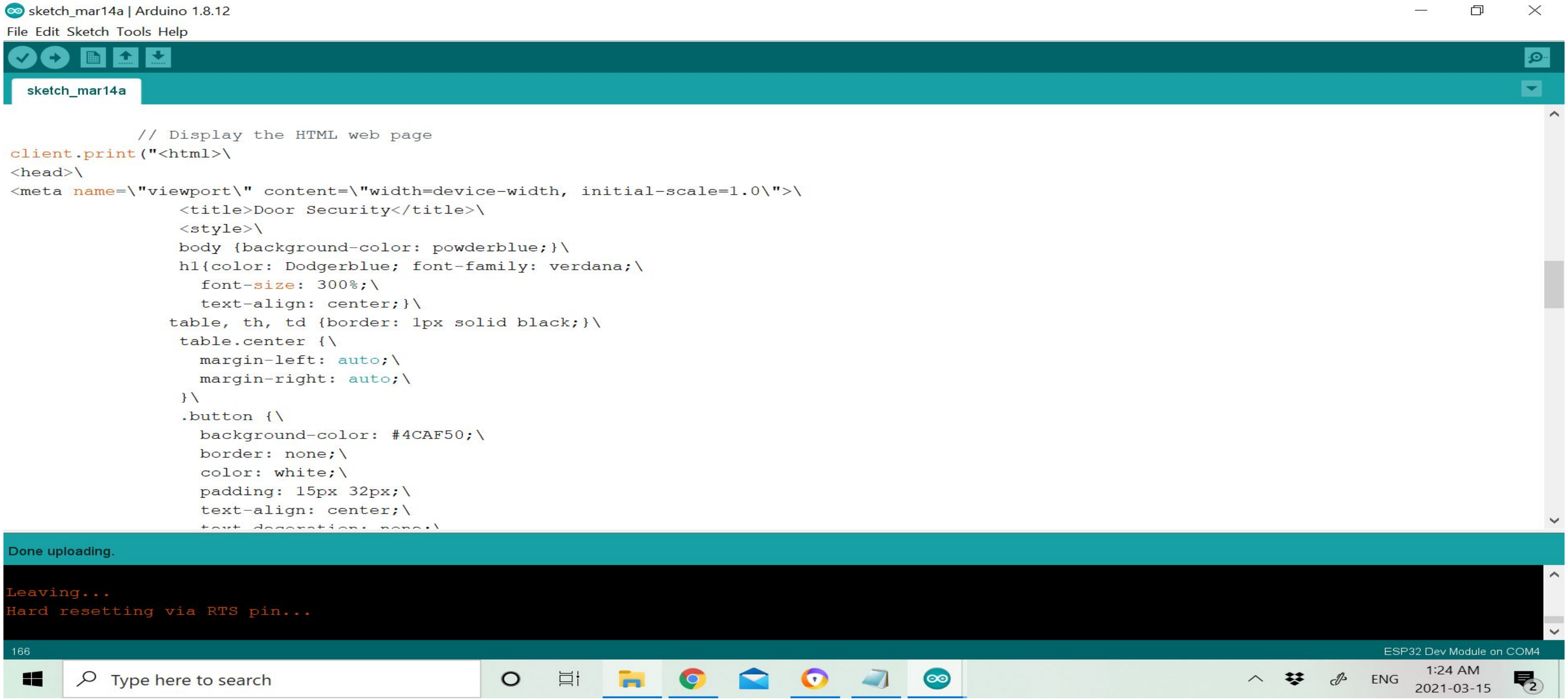
ESP32 Dev Module on COM4

Screenshot saved

The screenshot was added to your OneDrive.

OneDrive

# HTML coding on ESP32:

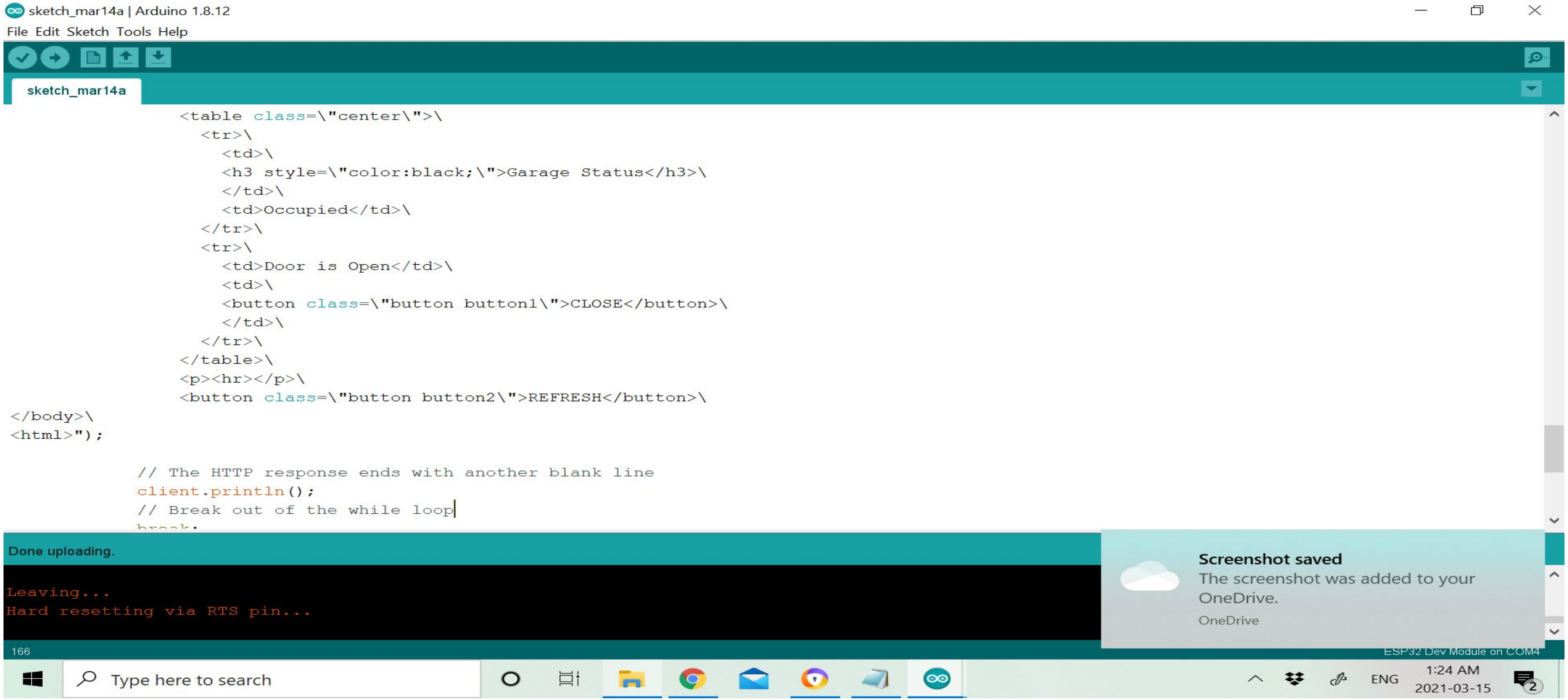


```
            // Display the HTML web page
client.print("<html>\
<head>\
<meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0\">\
                <title>Door Security</title>\
                <style>\
                body {background-color: powderblue;}\
                h1{color: Dodgerblue; font-family: verdana;\
                  font-size: 300%;\
                  text-align: center;}\
               table, th, td {border: 1px solid black;}\
                table.center {\
                  margin-left: auto;\
                  margin-right: auto;\
                }\
                .button {\
                  background-color: #4CAF50;\
                  border: none;\
                  color: white;\
                  padding: 15px 32px;\
                  text-align: center;\
                  text-decoration: none;\
```

Done uploading.

```
Leaving...
Hard resetting via RTS pin...
```

# HTML coding on ESP32:



```
                <table class=\"center\">\
                  <tr>\
                    <td>\
                    <h3 style=\"color:black;\">Garage Status</h3>\
                    </td>\
                    <td>Occupied</td>\
                  </tr>\
                  <tr>\
                    <td>Door is Open</td>\
                    <td>\
                    <button class=\"button button1\">CLOSE</button>\
                    </td>\
                  </tr>\
                </table>\
                <p><hr></p>\
                <button class=\"button button2\">REFRESH</button>\
</body>\
<html>");

            // The HTTP response ends with another blank line
            client.println();
            // Break out of the while loop
            break;
```

Done uploading.

```
Leaving...
Hard resetting via RTS pin...
```

166

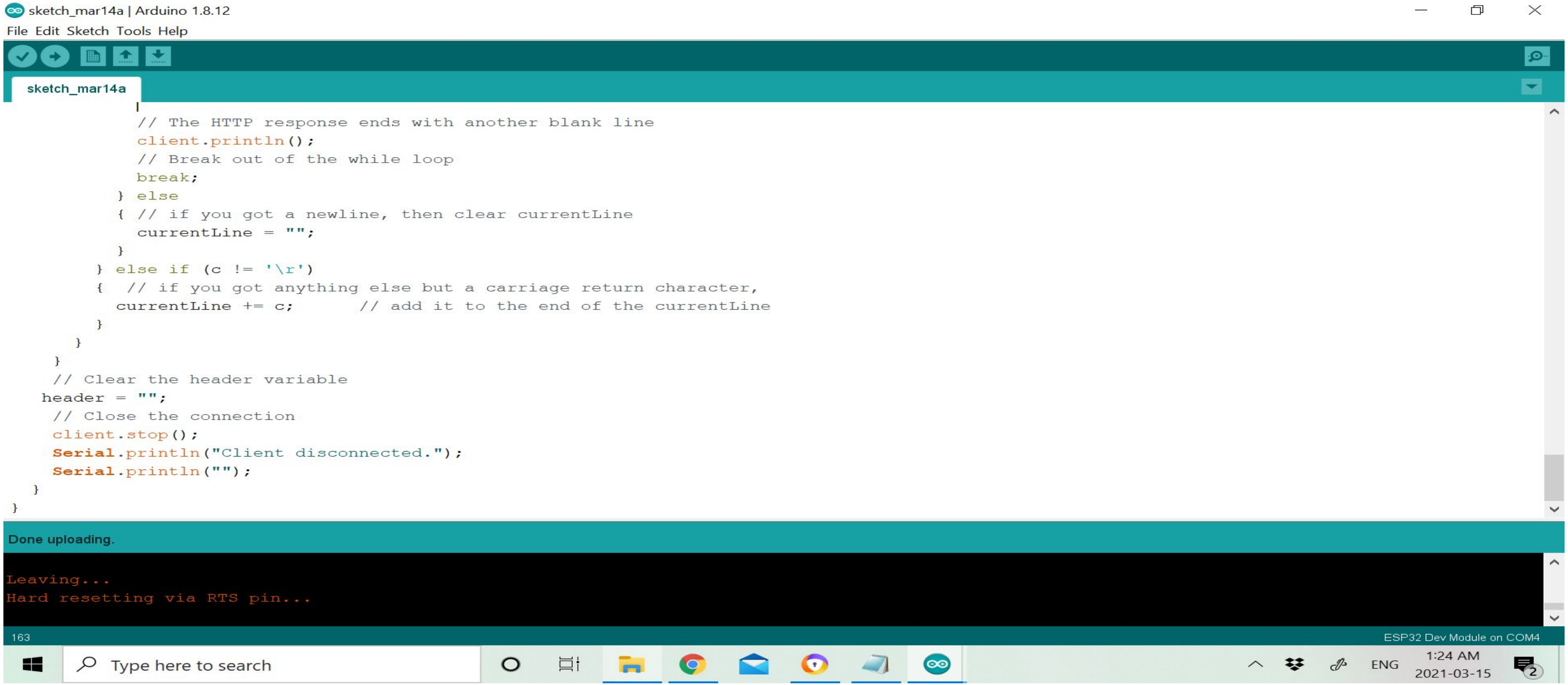
ESP32 Dev Module on COM4

Screenshot saved

The screenshot was added to your OneDrive.

OneDrive

# HTML coding on ESP32:



```
            // The HTTP response ends with another blank line
            client.println();
            // Break out of the while loop
            break;
          } else
          { // if you got a newline, then clear currentLine
            currentLine = "";
          }
        } else if (c != '\r')
        {  // if you got anything else but a carriage return character,
          currentLine += c;      // add it to the end of the currentLine
        }
      }
    }
    // Clear the header variable
   header = "";
    // Close the connection
    client.stop();
    Serial.println("Client disconnected.");
    Serial.println("");
  }
}
```

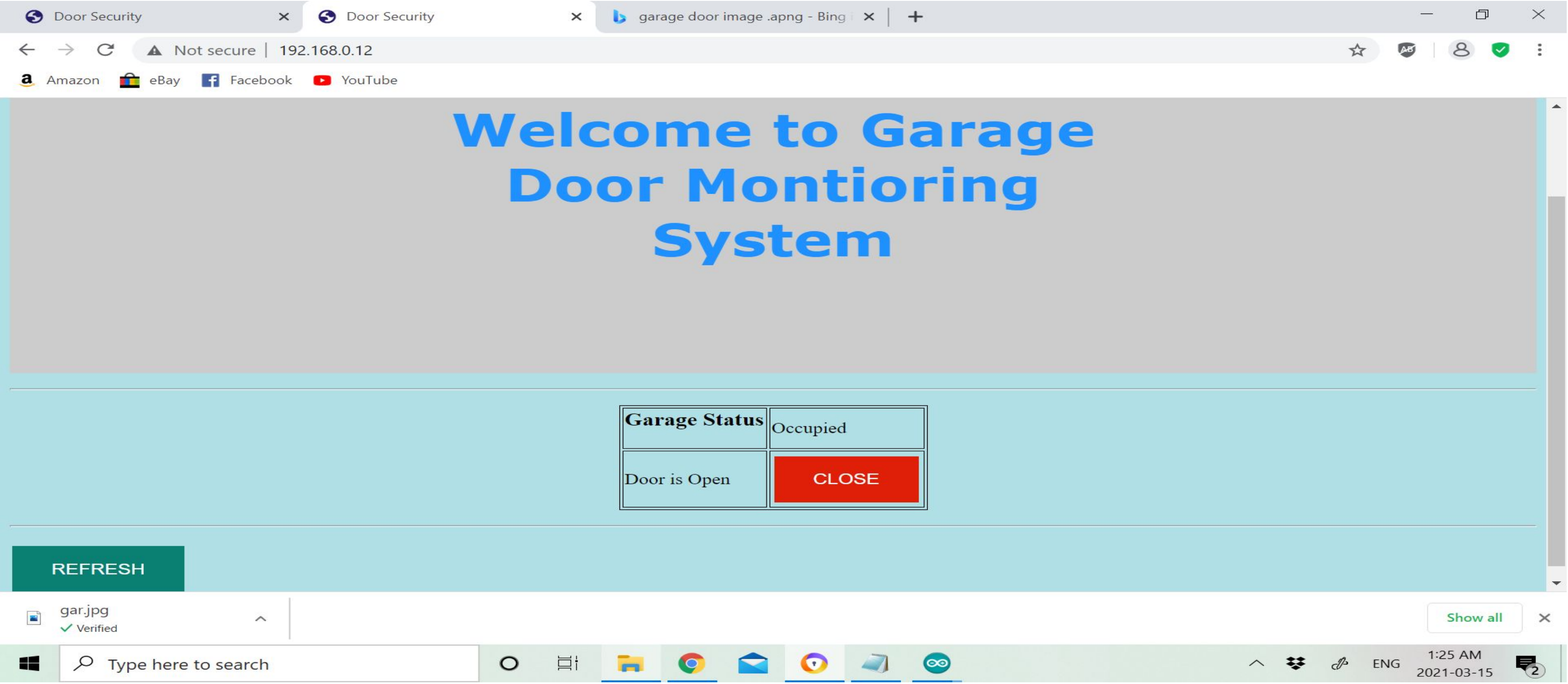
Done uploading.

```
Leaving...
Hard resetting via RTS pin...
```

163

ESP32 Dev Module on COM4

# HTML coding on ESP32: Output



Welcome to Garage Door Montioring System

| Garage Status | Occupied |
|---|---|
| Door is Open | CLOSE |

REFRESH

# References:

- Features of ESP32 retrieved from Esp32.net; features and specifications URL: http://esp32.net/
- Image of block diagram retrieved from Esp32.net; features and specifications URL: http://esp32.net/
- CSS3 retrieved from eduba.co;what is CSS3? URL: https://www.educba.com/what-is-css3/
- HTML retrieved from w3schools.com; URL: https://www.w3schools.com/html/default.asp
- ESP32 coding retrieved from randomnerdtutorial.com; URL: https://randomnerdtutorials.com/esp32-web-server-arduino-ide/