# Training Module – Bootstrap + Chart.js

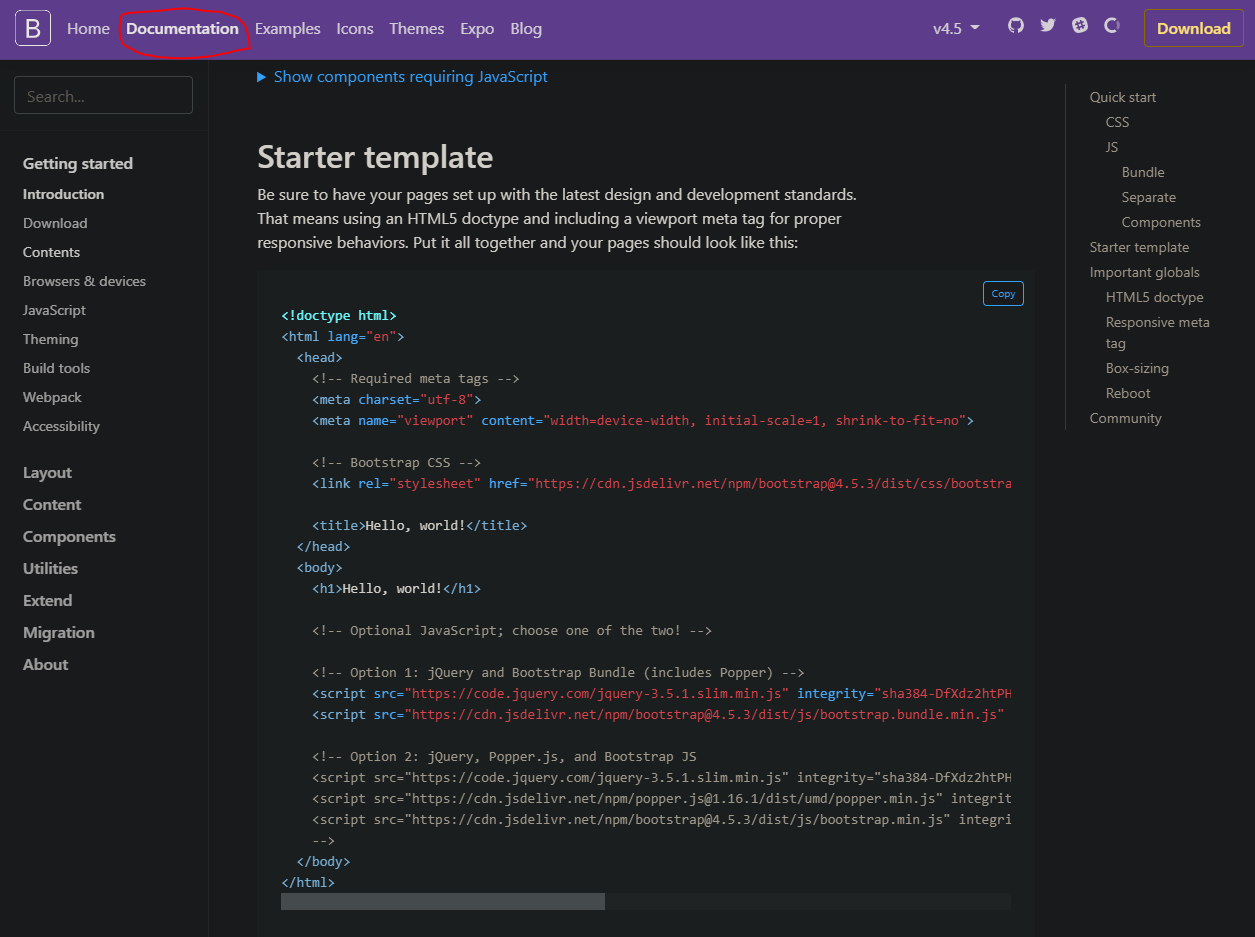
## What you will need:

* A computer
* Code editing software
  + [Visual Studio Code](https://code.visualstudio.com/download)
* An internet browser (Preferably chrome)
* Access to the internet for [bootstrap](https://getbootstrap.com/docs/4.5/getting-started/introduction/) starter code
* Trainer package from Github
* An open mind

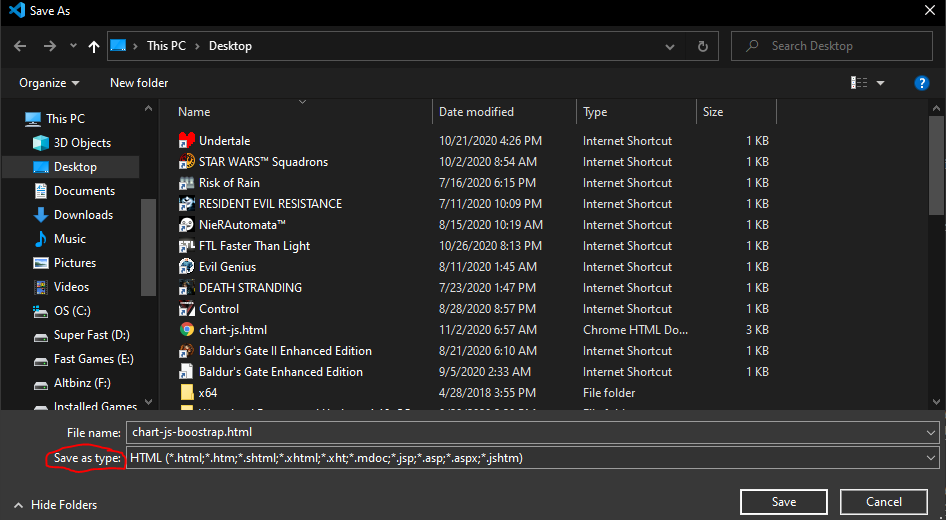
## Instructions:

GETTING STARTED

1. Open up Practice.HTML file found in the Trainer package with your code editing software.
2. Then get the starter template from the [bootstrap](https://getbootstrap.com/docs/4.5/getting-started/introduction/) website under the documentation tap at the top of the page.



1. Copy and past starter template into code editing software
2. Then make sure you save the file to any location on your computer. Name the file anything you like but make sure that the “save as type” stays as HTML.

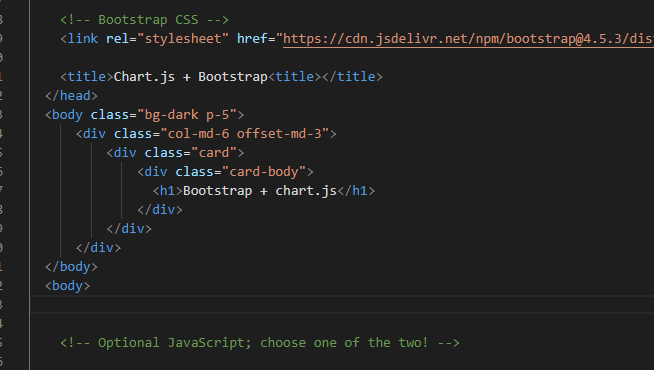


1. When you open the file, it should now display in your internet browser showing the text “Hello, World!”



CREATE A LINE CHART USING CHART.JS

1. Now change the h1 header and change the title to whatever title you want. We now are going to create a card to put the canvas on for the graph. You should add the card into the body as shown below. Also a good idea to add some changes to the CSS such as the padding.



Save and refresh your page. It should now show the card with padding on the webpage

(Note: this isn’t required but does help make the page more complete)

1. Now create a canvas tag inside the card-body class.

**Note**: Chart JS works with the canvas tag. The canvas tag is a drawing tag which gives you options to draw by using JavaScript commands. Without this tag you will not be able to draw any charts.

|  |
| --- |
| <canvas id=”myChart”></canvas> |

1. Now add the Chart.js library inside the body under the last div line.

|  |
| --- |
| <script src="https://cdn.jsdelivr.net/npm/chart.js@2.8.0"></script> |

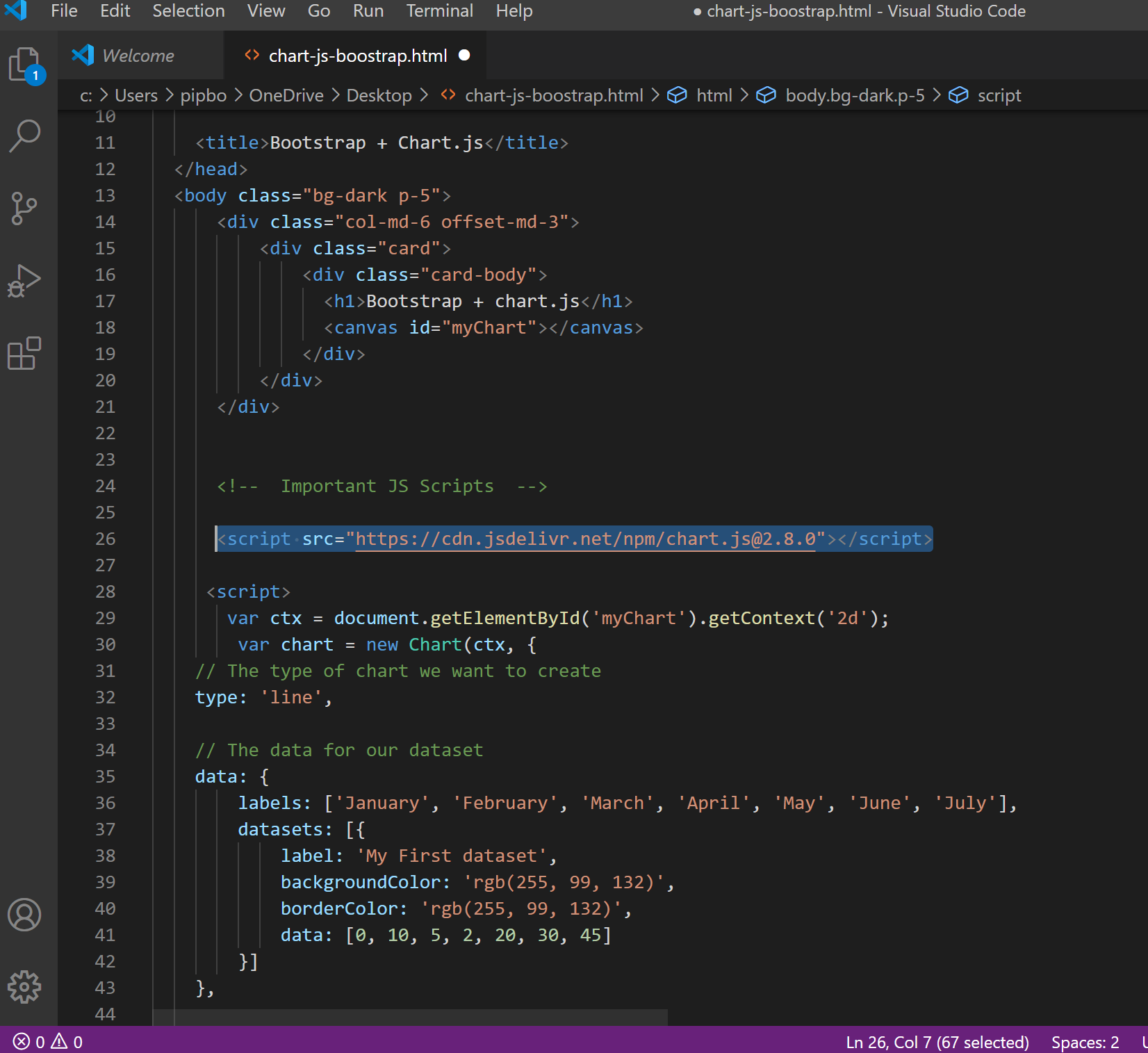
**Note**: The commands related to drawing in the canvas are in the Chart JS library. Basically, Chart JS did the heavy lifting for you by having a preprogrammed drawing the moment you enter your chart details.

1. Now add the chart commands script (Don’t forget to add this to a new <script> tag)

|  |
| --- |
| var ctx = document.getElementById('myChart').getContext('2d');  var chart = new Chart(ctx, {  // The type of chart we want to create  type: 'line',  // The data for our dataset  data: {  labels: ['January', 'February', 'March', 'April', 'May', 'June', 'July'],  datasets: [{  label: 'My First dataset',  backgroundColor: 'rgb(255, 99, 132)',  borderColor: 'rgb(255, 99, 132)',  data: [0, 10, 5, 2, 20, 30, 45]  }]  },  // Configuration options go here  options: {}  }); |

**Note**: This step is the chart commands which essentially is the drawing of your particular chart. This must always be placed underneath the Chart JS library or else it will not work

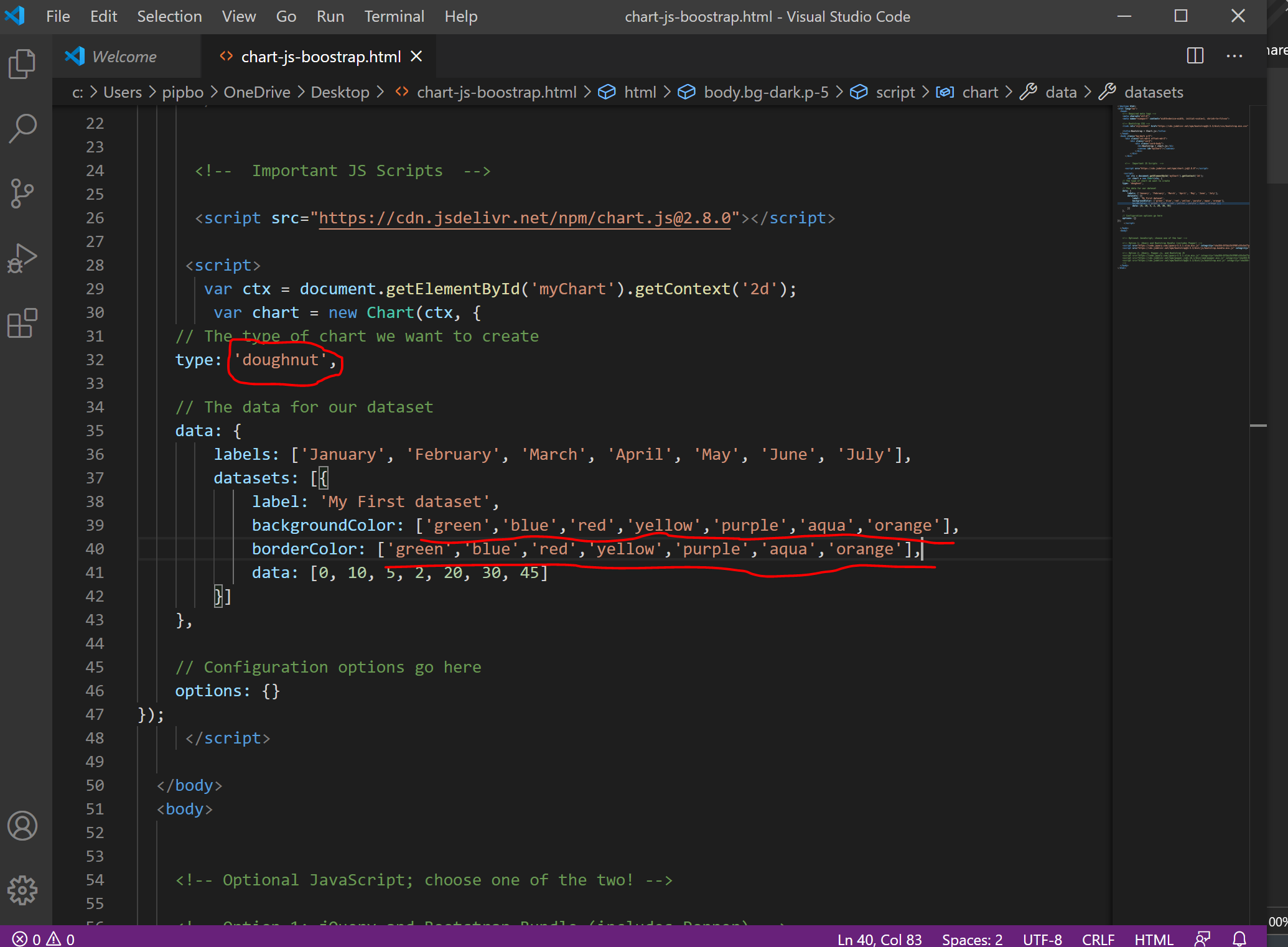
Your code should now look like this:



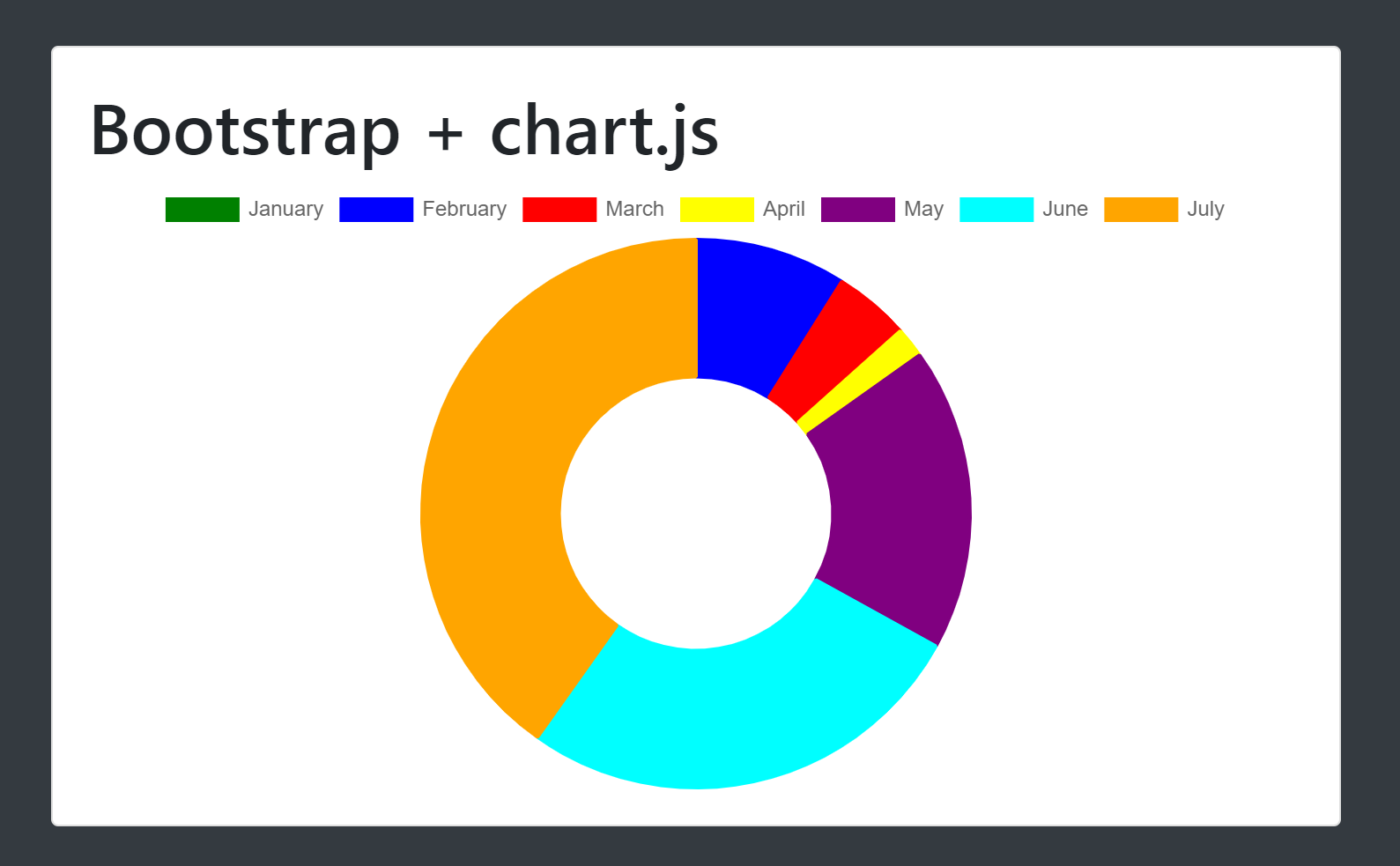
Save and refresh the webpage and you should now see the line chart that we just created.

CREATING A GUAGE CHART

1. Change the type of chart into doughnut and also to help differentiate the different datasets on the chart, change the backgroundColor and borderColor into JSON. (don’t forget to put in the [ ] so it is read as JSON) and add your own colors for each dataset.



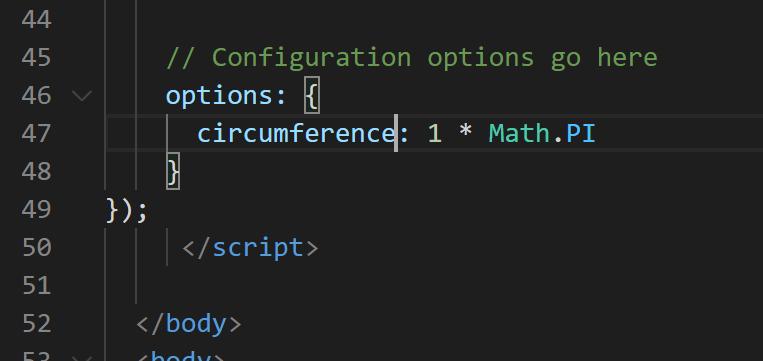
Save and refresh the webpage. It should know look like this:



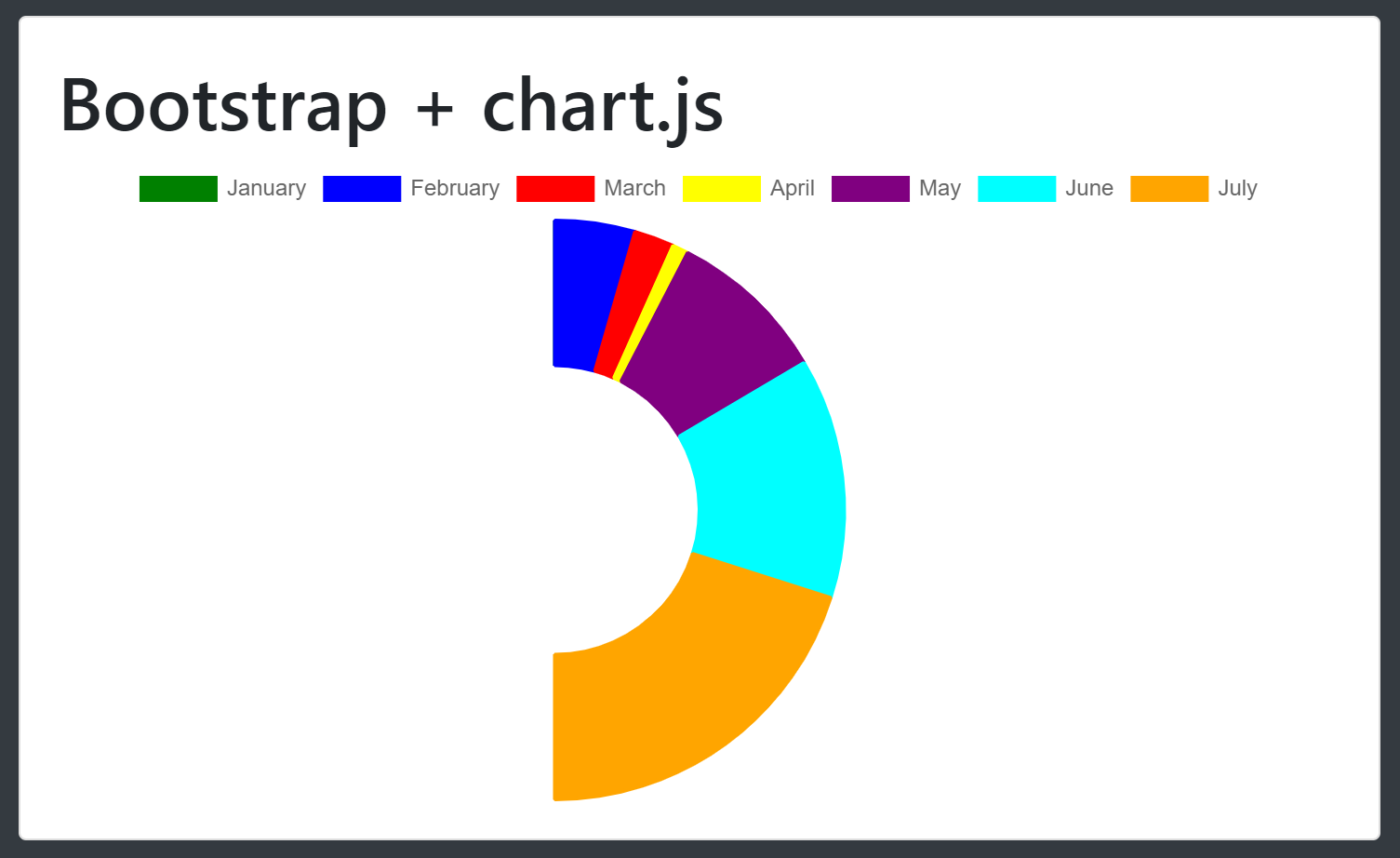
1. Now we are going to turn the donut chart into a gauge chart by adding some chart.js commands. To do this we need to add special options into the code.

To do this we need to change the circumference of the circle. We will need to add the formula into the “options” at the bottom of the chart.js of the code. (Note. This is case sensitive)

|  |
| --- |
| circumference: 1 \* Math.PI |



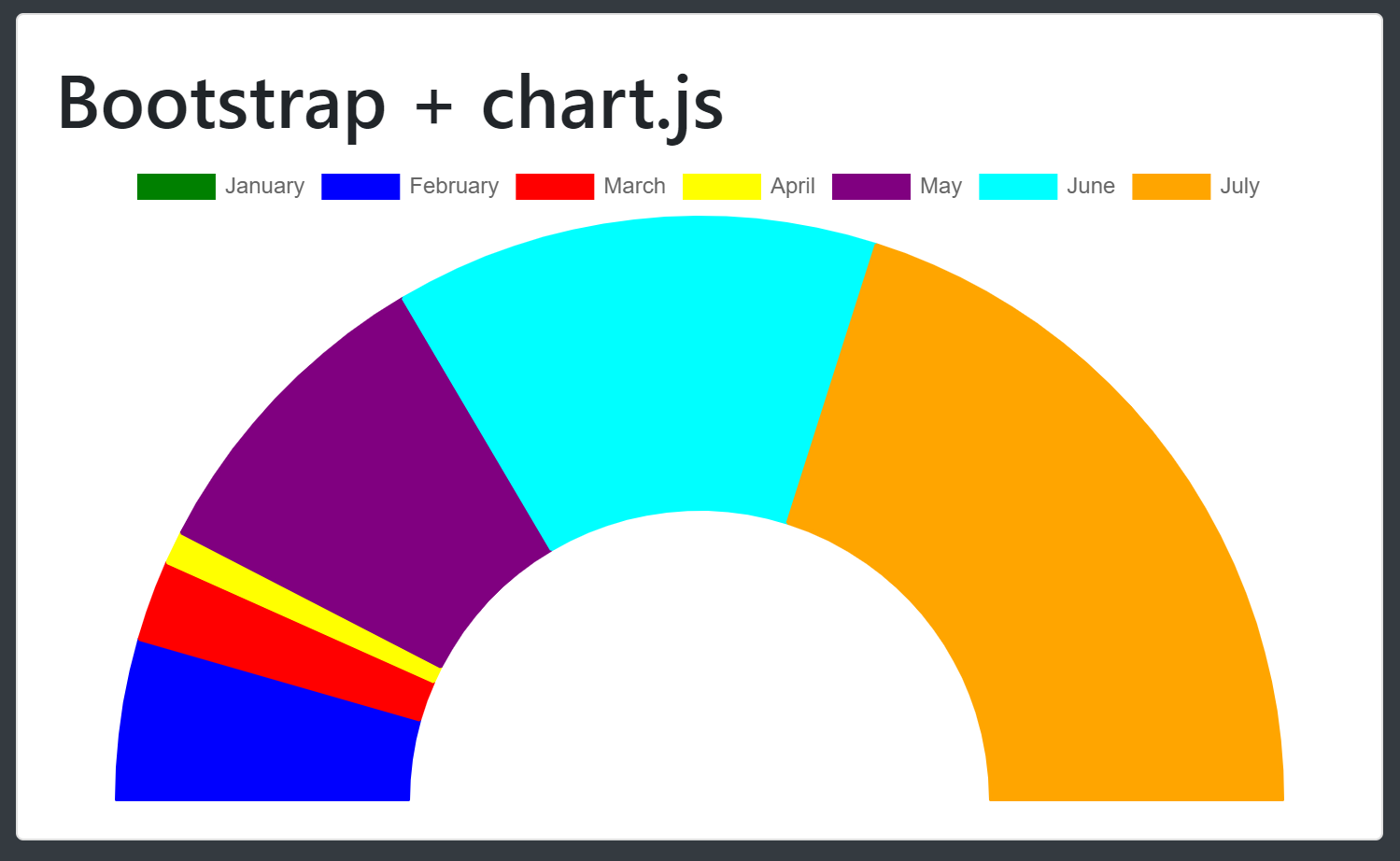
Save and refresh and now the doughnut should be half a circle:



Note: As you notice though the circle needs to be rotated so it can look like a gauge. We will need to add that to the options as well.

|  |
| --- |
| rotation: 1 \* Math.PI |

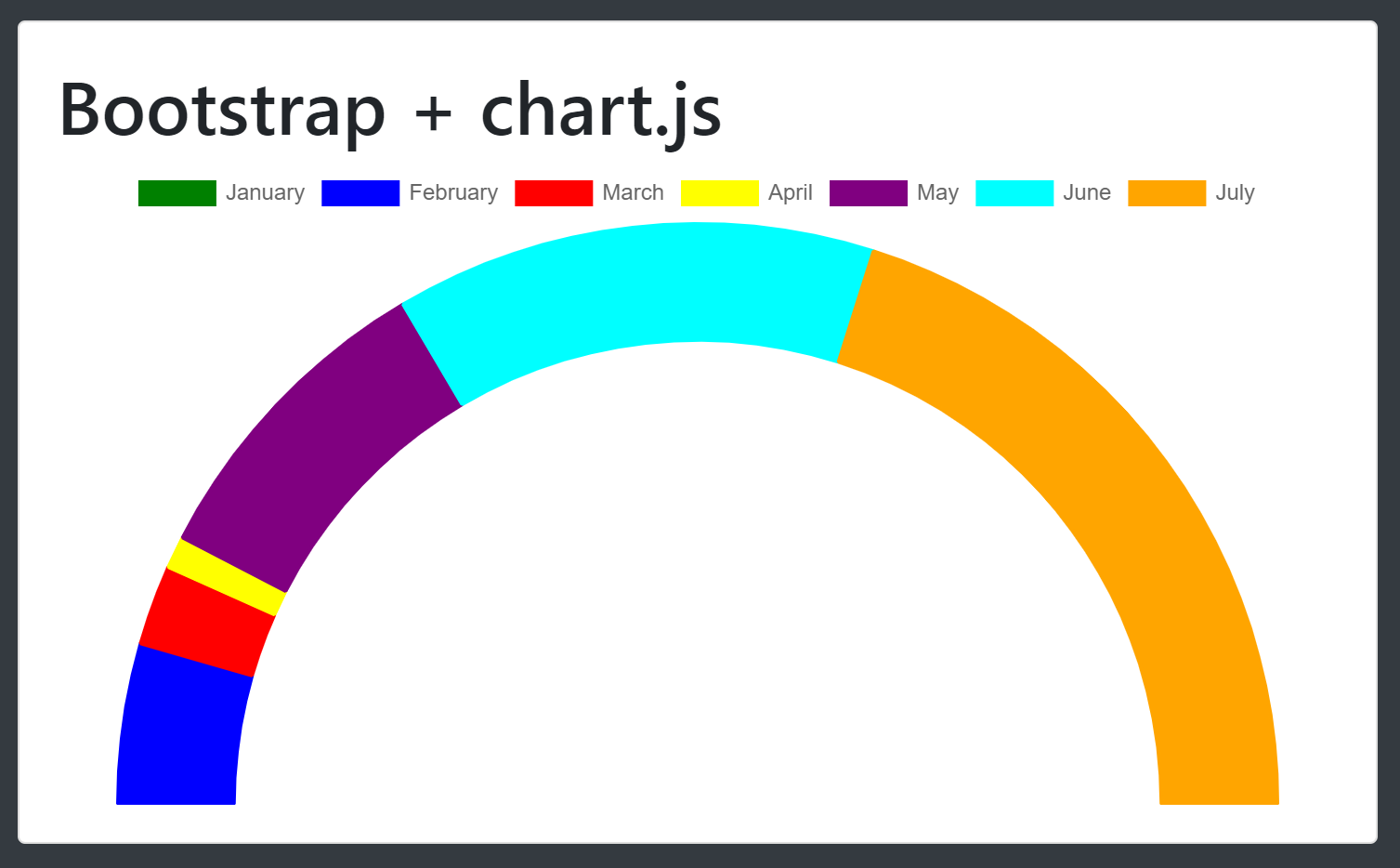
Save and refresh:



1. Lets now make some changes to the gauge chart to look more appealing. Add the following code to the options as well

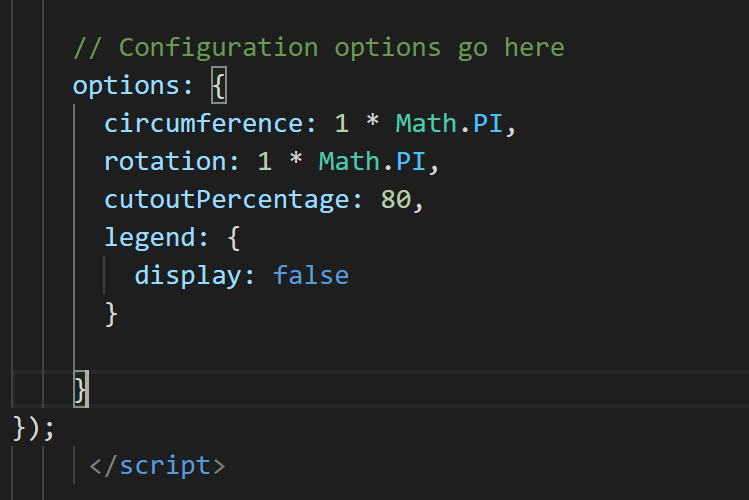
|  |
| --- |
| cutoutPercentage: 80 |

Save and refresh:

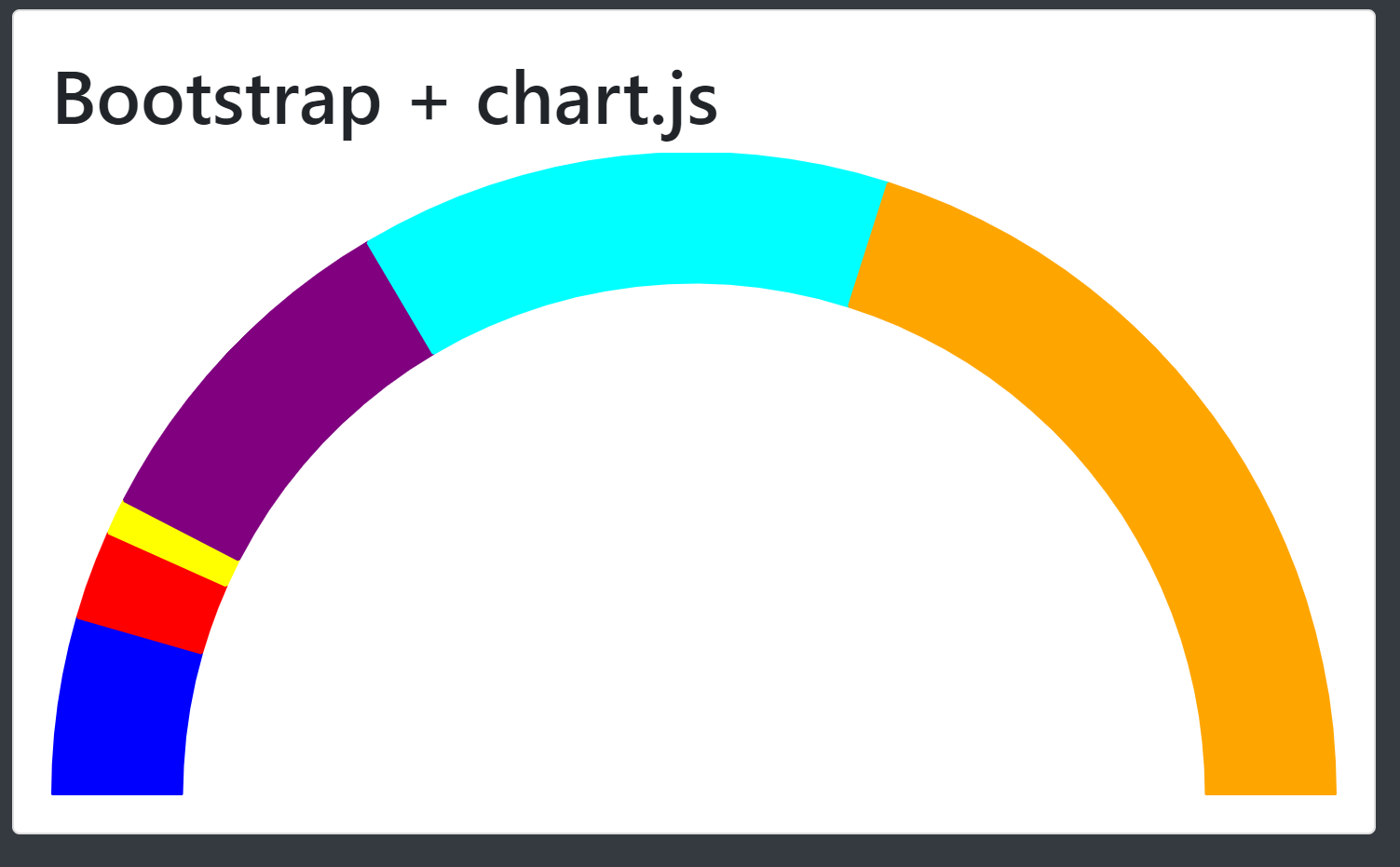


1. Now let’s get rid of the month names as we currently don’t want them to be displayed. To do this you can disable them in the same options with this line:

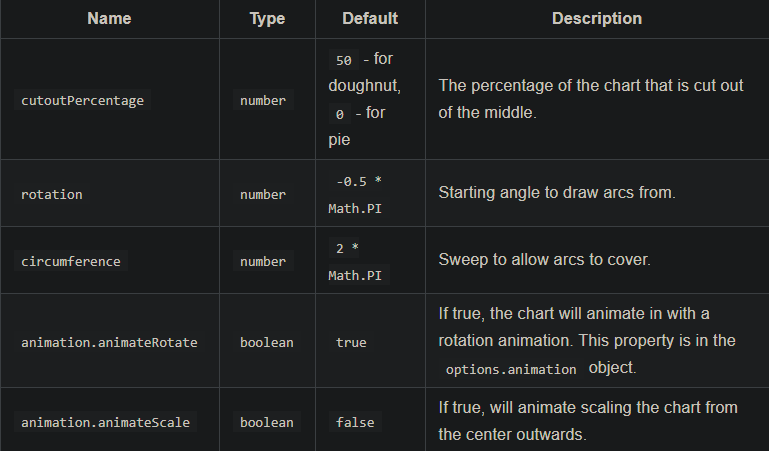
|  |
| --- |
| legend: {  display: false  } |



Save and Refresh:

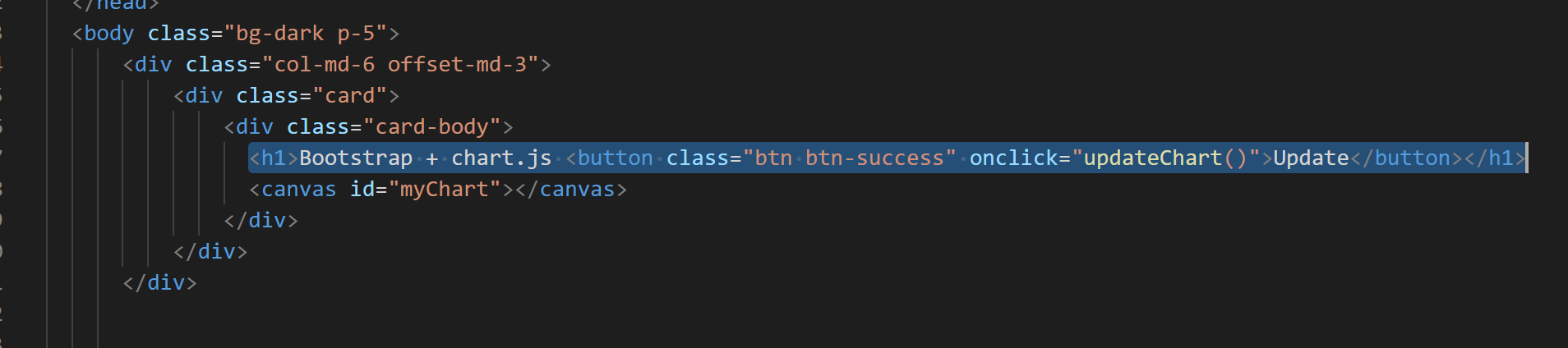


Here is a list of Config options for the Doughnut and Pie chart:



CREATING UPDATE BUTTON(Optional)

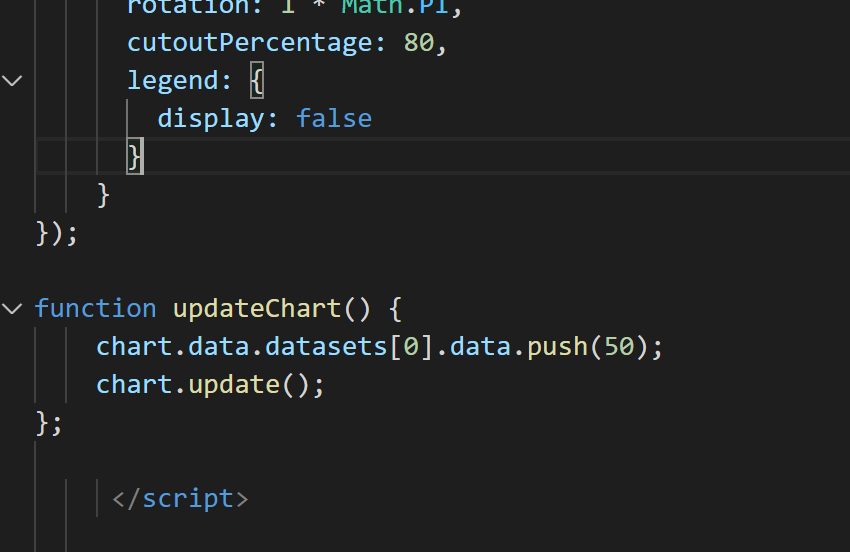
1. Create a button inside the body.



Save and Refresh:

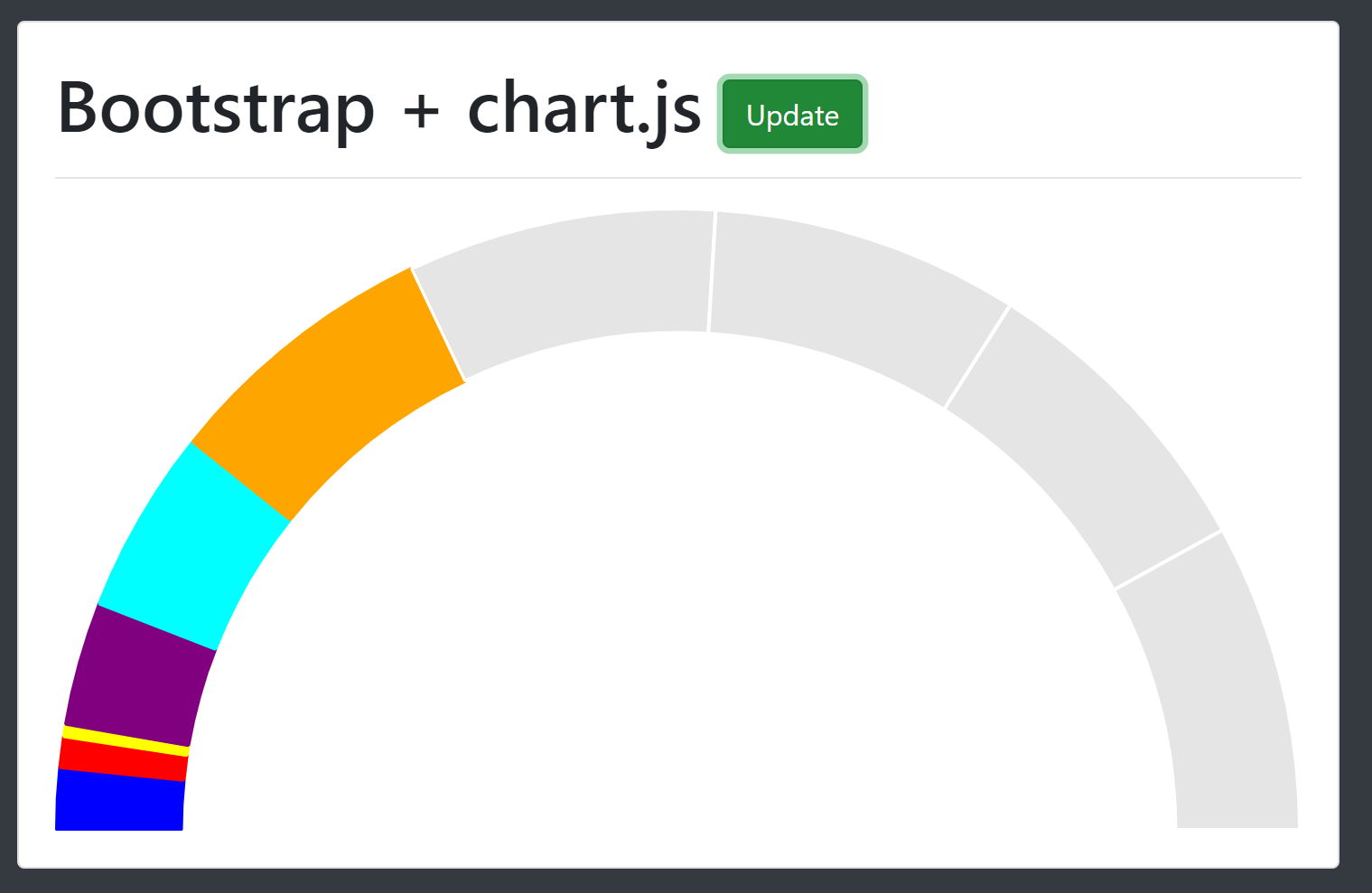
Note: You should now see an update button inside the body of the card. You can press on it but it will not do anything as we still need to add the functionality of what the button does.

1. Under options but still inside the script of the program add the updateChart() function.



Save and Refresh:

Now when you press the update button on the site it will add the value that you put into the function to the chart.



Note: as you noticed it does not add the color, you can add this to the current function you just made by adding this line to it:

|  |
| --- |
| chart.data.datasets[0].backgroundColor.push(“blue”); |

