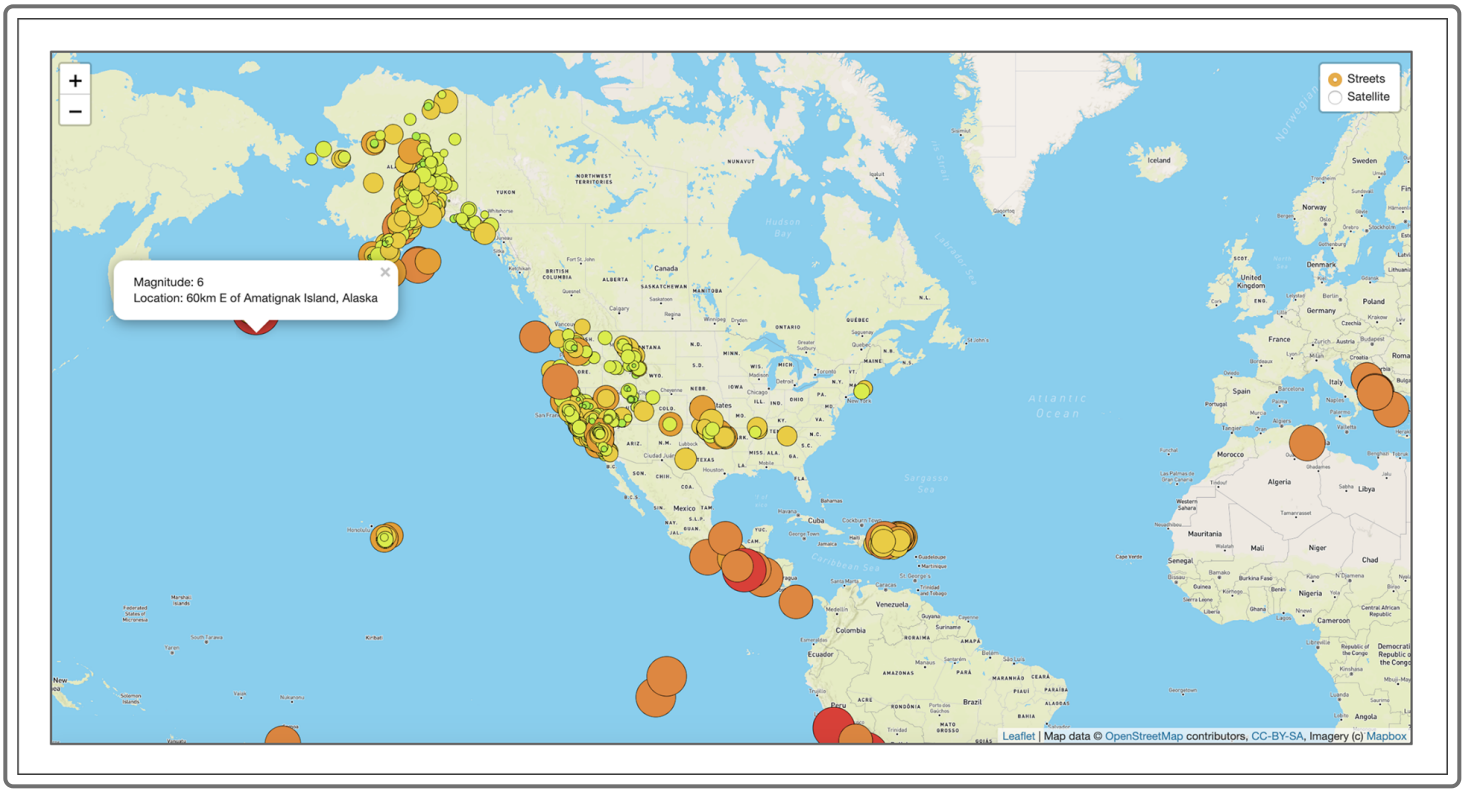
13.6.3

## Add Color and a Popup for Each Earthquake

**Sadhana** thinks that the size of the earthquake data based on magnitude looks great, but it's hard to tell the difference between earthquakes within the same area. As you toss around ideas to make this data more accessible to the viewer, you come up with the idea to color-code the earthquakes based on magnitude. You aren't quite sure how to do this yet, but you know that it should be possible based on your experience with JavaScript thus far. Basil loves the idea, so you get back to coding to figure out how to make it happen. And while you're working on changing the color code for magnitude, Basil and Sadhana suggest that you add the magnitude and location as a popup for each earthquake.

After we're done adding a color range for the magnitude and a popup for each earthquake, our map should look like the following:



Before we write the code to create this map, make a copy of the `logicStep2.js` file and name it `logicStep3.js`. Now let's edit the file.

First, we'll create a fill-color range for the magnitude. In the `styleInfo()` function, our `fillColor` was set with `fillColor: "#ffae42"`. We'll replace the hexadecimal color code with the function `getColor()`. Inside the parentheses, we'll add the dot notation code to get the magnitude as we did for the `getRadius()` function, since we'll change the color of each earthquake marker based on the magnitude.

Add the `getColor(feature.properties.mag)` function for the `fillColor` so that our `styleInfo()` function looks like the following:

```
// This function returns the style data for each of the earthquakes we plot on
// the map. We pass the magnitude of the earthquake into two separate functions
// to calculate the color and radius.
function styleInfo(feature) {
  return {
    opacity: 1,
    fillOpacity: 1,
    fillColor: getColor(feature.properties.mag),
    color: "#000000",
    radius: getRadius(feature.properties.mag),
    stroke: true,
    weight: 0.5
  };
}
```

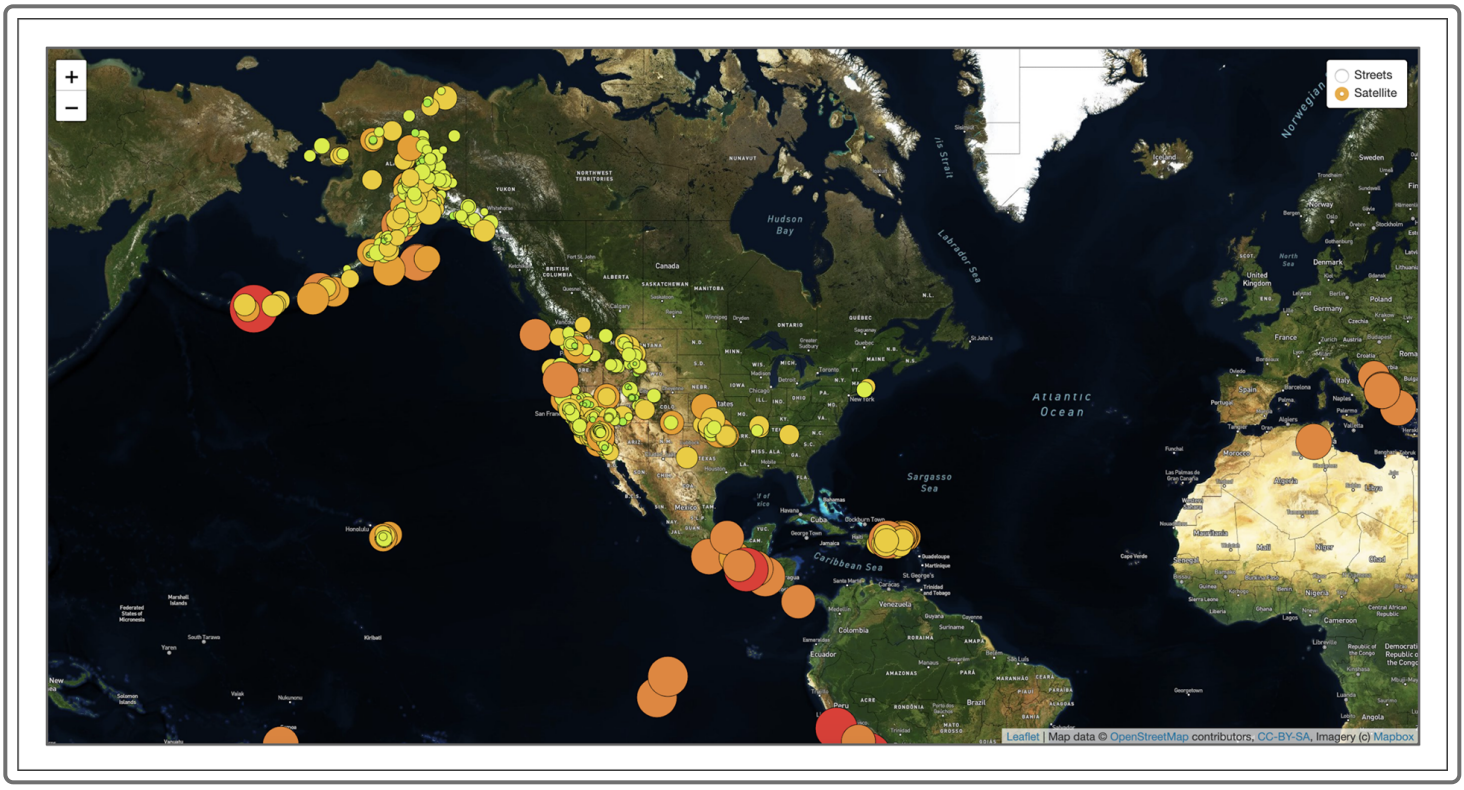
Now we need to write code for the `getColor()` function to change the marker's color based on the magnitude. For example, if the magnitude is greater than 5, it will be a certain color, if the magnitude is greater than 4, it will be a different color, and so on.

Retake

For the `getColor()` function, we'll write a conditional expression with logical operators for the magnitudes. Add the following `getColor()` function below the `styleInfo()` function and above the `getRadius()` function. Sadhana suggests using the following colors for the magnitudes since they'll be visible on the Satellite map:

```
// This function determines the color of the circle based on the magnitude of the earthquake.
function getColor(magnitude) {
  if (magnitude > 5) {
    return "#ea2c2c";
  }
  if (magnitude > 4) {
    return "#ea822c";
  }
  if (magnitude > 3) {
    return "#ee9c00";
  }
  if (magnitude > 2) {
    return "#eecc00";
  }
  if (magnitude > 1) {
    return "#d4ee00";
  }
  return "#98ee00";
}
```

Let's save our `logicStep3.js` file and open the `index.html` file in the browser to confirm our code is working. When we select the dark map, our map should look similar to the following map:



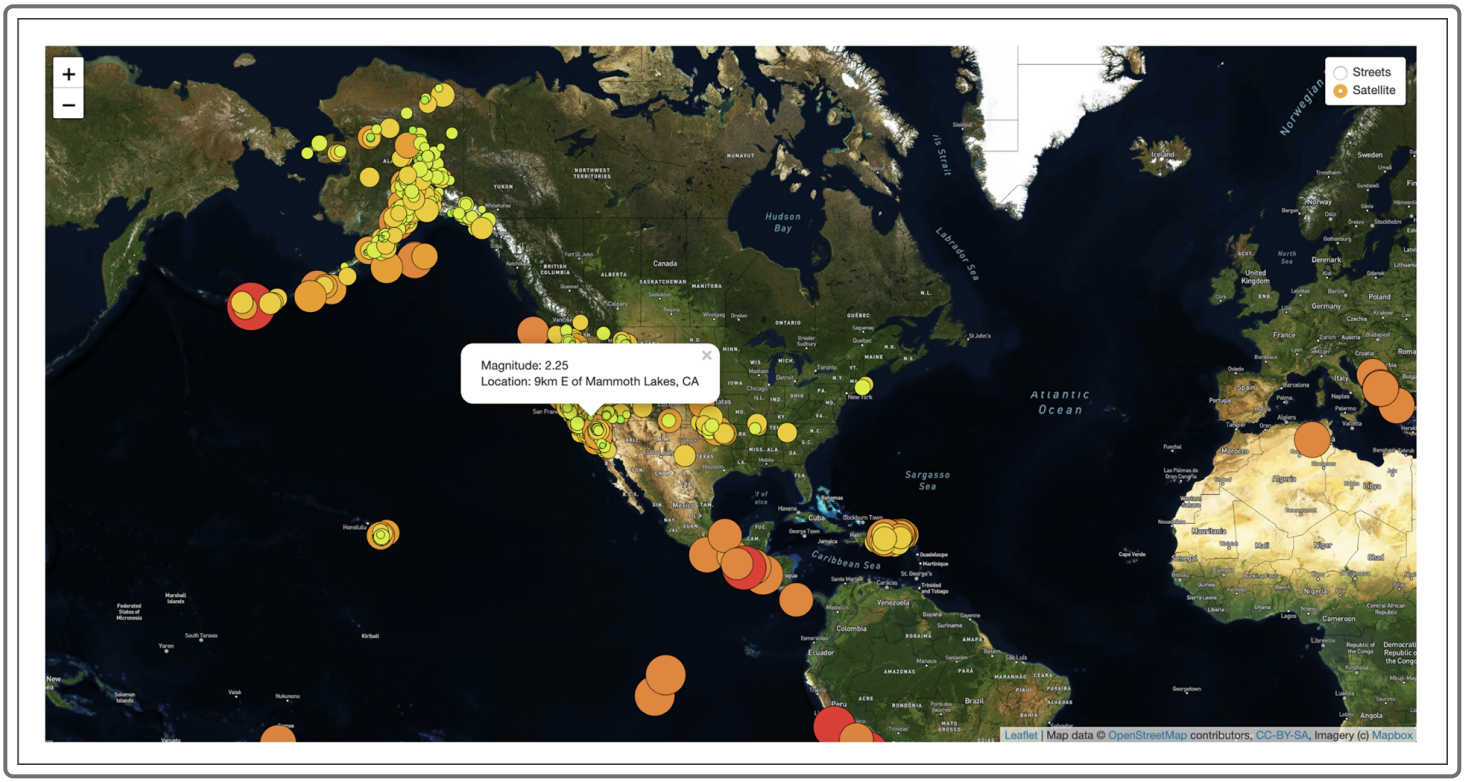
Now we need to edit the GeoJSON layer code to add the popup for the magnitude and location.

Retake

In the `geoJson` layer, we'll add the `onEachFeature` function to add a popup for each circle marker. Edit the `L.geoJSON()` layer code to include the `onEachFeature` function with the `bindPopup()` method:

```
// Creating a GeoJSON layer with the retrieved data.
L.geoJSON(data, {
    // We turn each feature into a circleMarker on the map.
    pointToLayer: function(feature, latlng) {
        console.log(data);
        return L.circleMarker(latlng);
      },
    // We set the style for each circleMarker using our styleInfo function.
  style: styleInfo,
    // We create a popup for each circleMarker to display the magnitude and
    //  location of the earthquake after the marker has been created and styled.
    onEachFeature: function(feature, layer) {
    layer.bindPopup("Magnitude: " + feature.properties.mag + "<br>Location: " + feature.properties.place);
  }
}).addTo(map);
```

When you save the `logicStep3.js` file and open your `index.html` file in your browser, the Satellite map option will look like the following:




Great job on adding color and a popup marker to each earthquake!

### ADD/COMMIT/PUSH

Add, commit, and push your changes to your Earthquakes_past7days branch.

Next, Sadhana will show you how to add the earthquake data as an overlay to the tile layer so that the data can be turned on and off by the viewer.