JavaScript Visualization: Group 5

## **Group Members:** Rajeshwari Radhakrishna; Kelly Blake; Aisha Henderson

## **Topic:**

The goal of this project is to create an interactive visualization that will allow end-users to view and explore COVID-19 data by country. We will be leveraging available datasets from reliable sources to understand the differences and global impact of COVID-19 over time, looking at areas such as total cases, total deaths, and new/active cases.

## **Dataset(s):**

1. Covid Cases by Country -2023
   * **Source:** Kaggle
   * **About Dataset:** Number of reported cases and deaths from the SARS-CoV-2 Virus for each country.Data (to-date) as of 8/6/2023.
   * **Link:** <https://www.kaggle.com/datasets/joebeachcapital/covid-cases-by-country-2023>
2. Daily Cases and Deaths by Date (1) // Vaccination Data (2)
   * **Source**: World Health Organization (WHO)
   * **Dataset 1:** New and total confirmed cases and deaths for COVID-19, per country. Updated daily.
   * **Dataset 2:** Total Vaccinations and Boosters per country. Updated weekly.
   * **Links:**
     1. <https://covid19.who.int/WHO-COVID-19-global-data.csv>
     2. <https://covid19.who.int/who-data/vaccination-data.csv>

## **Tools/Visualization *Ideas*:**

JavaScript Libraries

* Leaflet
* Apex
* D3

Visualizations:

* Choropleth Map – global visualization color-coded.
* Marker Clusters – global data, display of clusters showing areas with significant case or death totals.
* Pie Chart – comparison for total vaccinations across countries?
* Line Chart – data over time – new cases?
* Apex Chart – data by country

## **Visualization Inspiration:**

|  |  |
| --- | --- |
| A map of the world  Description automatically generated |  |
| A graph with colorful rectangular bars  Description automatically generated with medium confidence |  |

## **Dashboard Sketch (Ideas):**

A screenshot of a computer

Description automatically generated

A screenshot of a graph

Description automatically generated

let link = "https://disease.sh/v3/covid-19/countries"

// Perform a GET request to the query URL/

d3.json(link).then(function (data) {

// Send the data features object to createFeatures function.

createFeatures(data.features);

console.log(data);

});

function createFeatures(covidData) {

// Define a function that we want to run once for each feature in the features array

// Give each feature a popup that describes the place and number of cases

function onEachFeature(feature, layer) {

layer.bindPopup(`<h3>${feature.properties.country}</h3><hr><p>${new Cases(feature.properties.cases)}</p>`);

}

// Create a function to increase marker size based on case numbers

function markerSize(cases) {

return cases \* 2;

}

// Create a GeoJSON layer containing the features array on the covidData object

let covid\_info = L.geoJSON(covidData, {

onEachFeature: onEachFeature,

pointToLayer: function (feature, latlong) {

return L.circleMarker(latlong, {

radius: markerSize(feature.properties.cases),

fillColor: "blue",

color: "white",

weight: 1,

opacity: 1,

fillOpacity: 0.9

});

}

});

// Send our covid data to the createMap function

createMap(covid\_info);

}

function createMap(covid\_info) {

// Create the base layers

let street = L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {

attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'

})

let topo = L.tileLayer('https://{s}.tile.opentopomap.org/{z}/{x}/{y}.png', {

attribution: 'Map data: &copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors, <a href="http://viewfinderpanoramas.org">SRTM</a> | Map style: &copy; <a href="https://opentopomap.org">OpenTopoMap</a> (<a href="https://creativecommons.org/licenses/by-sa/3.0/">CC-BY-SA</a>)'

});

// Create a basemaps object

let basemaps = {

"Street Map": street,

"Topographic Map" : topo

};

// Create an overlayMaps object to hold the cases layer

let overlayMaps = {

Covid\_info: covid\_info

};

// Create the map object to display streetmap and earthquakes layers on load

let map = L.map("map", {

center: [37.0902, -95.7129],

zoom: 5,

layers: [street, covid\_info]

});

// Create a layer control and pass baseMaps and overlayMaps. Add the layer control to the map.

L.control.layers(basemaps, overlayMaps, {

collapsed: false

}).addTo(map);

}