



PROJECT REPORT

Instructor: Dr. Tran Thanh Tung

COVID-19 EPIDEMIC IN VIETNAM
Complicated outbreak

Author: Võ Văn Việt,
Đặng Quang Hưng,
Khổng Minh Đức, Cao
Ngọc Bảo Long.

<https://github.com/vvviet2908/DS-DV>

Data Science and Data Visualization

APPENDIX

Overview - Background and Motivation:	2
Objectives: the primary question to answer	3
Data Collecting:	3
3.1 Center for Systems Science and Engineering - Johns Hopkins University:	3
3.2 Tableau COVID-19 Data Resource Hub:	5
3.3 Ministry of Health of Vietnam:	5
3.4 Vietnamese Geo-data:	7
Data Processing:	7
4.1 The data from CSSE JHU and Tableau Data Resource Hub:	8
4.2 The data from the Ministry of Health of Vietnam:	10
Visualization Design:	10
Our D3.js Web-Based Interactive Visualization:	11
6.1 About D3.js:	11
6.2 D3.js and our project:	12
6.2.1 Confirmed Choropleth Map:	12
6.2.2 Age Histogram:	17
6.2.3 Patient's Status Pie Chart:	18
6.2.4 Patient's Nationality Pie Chart:	20
6.4.5 Time-series Line Chart:	22
Result- The Dashboard:	27
Development directions in the future:	28
References:	28
Conclusion:	29

COVID-19 EPIDEMIC IN VIETNAM

1. Overview - Background and Motivation:

On December 31, 2019, the World Health Organization (WHO) was informed of an outbreak of “pneumonia of unknown cause” detected in Wuhan City, Hubei Province, China – the seventh-largest city in China with 11 million residents. As of January 23, there are over 800 cases of 2019-nCoV confirmed globally, including cases in at least 20 regions in China and nine countries/territories. The first reported infected individuals, some of whom showed symptoms as early as December 8, were discovered to be among stallholders from the Wuhan South China Seafood Market. Subsequently, the wet market was closed on Jan 1. The virus causing the outbreak was quickly determined to be a novel coronavirus. On January 10, gene sequencing further determined it to be the new Wuhan coronavirus, namely 2019-nCoV, a betacoronavirus, related to the Middle Eastern Respiratory Syndrome virus (MERS-CoV) and the Severe Acute Respiratory Syndrome virus (SARS - CoV). However, the mortality and transmissibility of 2019-nCoV are still unknown, and likely to vary from those of the prior referenced coronaviruses. As of 14 May, more than 4.4 million cases of COVID-19 have been reported in more than 210 countries and territories, resulting in nearly 300,000 deaths, but almost 1,660,000 recoveries.

In the first wave of the epidemic, Central China, Korea, Italy, Iran have been the countries that have the most number of cases and death. In the second wave, the US, Latin America, Russia,... have become the biggest infected countries. In the US, there are nearly 2 million confirmed cases.

In Vietnam, as of today, the country had, according to official statistics, suffered no deaths from the virus and had limited total infections to just 332, despite being next door to China and a popular holiday destination during the spring festival, when the coronavirus first hit the Chinese city of Wuhan. This has led many observers to suggest that the country's pandemic control strategy could be a model for others to copy, especially for developing countries. But that is unlikely to succeed because few other countries have, or want to have, the structures of control that Vietnam possesses.

It is no longer infected cases in the community, almost confirmed cases are from overseas citizens who have come home and are in quarantine. In other words, we can say that Vietnam has defeated the pandemic. It is not only the deserts of Vietnam's government, but also the deserts of all Vietnamese citizens. Although the outbreak is over, we should not be caught off guard about it, because it is not over in the world.

The interactive visualization, which illustrates the location and number of confirmed, active COVID-19 cases, deaths, and recoveries for all affected countries. It was developed to provide researchers, public health authorities, and the general public with a user-friendly tool to track the outbreak as it unfolds. Helping people understand the disease situation clearly, updating the latest news about this virus, and tracking the rise of the virus over time to see how effective anti-epidemic Vietnamese is. It can gain many valuable pieces of information to the community.

2. Objectives: the primary question to answer

HOW DOES COVID-19 AFFECT VIETNAM?

HOW IS COVID-19 CONTROLLED IN VIETNAM?

3. Data Collecting:

As the COVID-19 situation has unfolded across the globe, data collection and reporting has proven both difficult and highly variable. But it's also shown just how important rigorous data collection and reporting really is.

3.1 Center for Systems Science and Engineering - Johns Hopkins University:

In this Project, we will mainly use the dataset from Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE - USA) via their Github (<https://github.com/CSSEGISandData/COVID-19>). This GitHub repository and its contents herein, including all data, mapping, and analysis, copyright 2020 Johns Hopkins University, all rights reserved, is provided to the public strictly for educational and academic research purposes. They provide some .csv files which are the daily reports data and the time series data. The daily report data has been updated twice a day about the latest confirmed, recovered cases, deaths separated by countries/regions and provinces. The time series contains 3 particular datasets, that are confirmed, deaths, recovered cases. The data has been generated from 22/1/2020 and updated day by day.

This is a reputable dataset that JHU has synthesized from many official sources, which has been reported by the government office of many countries.

Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29/20
Anhui	Mainland China	31.8257	117.2264	1	9	15	39	60	70	106	152
Beijing	Mainland China	40.1824	116.4142	14	22	36	41	68	80	91	111
Chongqing	Mainland China	30.0572	107.874	6	9	27	57	75	110	132	147
Fujian	Mainland China	26.0789	117.9874	1	5	10	18	35	59	80	84
Gansu	Mainland China	36.0611	103.8343	0	2	2	4	7	14	19	24
Guangdong	Mainland China	23.3417	113.4244	26	32	53	78	111	151	207	277
Guangxi	Mainland China	23.8298	108.7881	2	5	23	23	36	46	51	58
Guizhou	Mainland China	26.8154	106.8748	1	3	3	4	5	7	9	9
Hainan	Mainland China	19.1959	109.7453	4	5	8	19	22	33	40	43
Hebei	Mainland China	38.0428	114.5149	1	1	2	8	13	18	33	48
Heilongjiang	Mainland China	47.862	127.7615	0	2	4	9	15	21	33	38
Henan	Mainland China	33.88202	113.614	5	5	9	32	83	128	168	206
Hubei	Mainland China	30.9756	112.2707	444	444	549	761	1058	1423	3554	3554
Hunan	Mainland China	27.6104	111.7088	4	9	24	43	69	100	143	221

Data Sources:

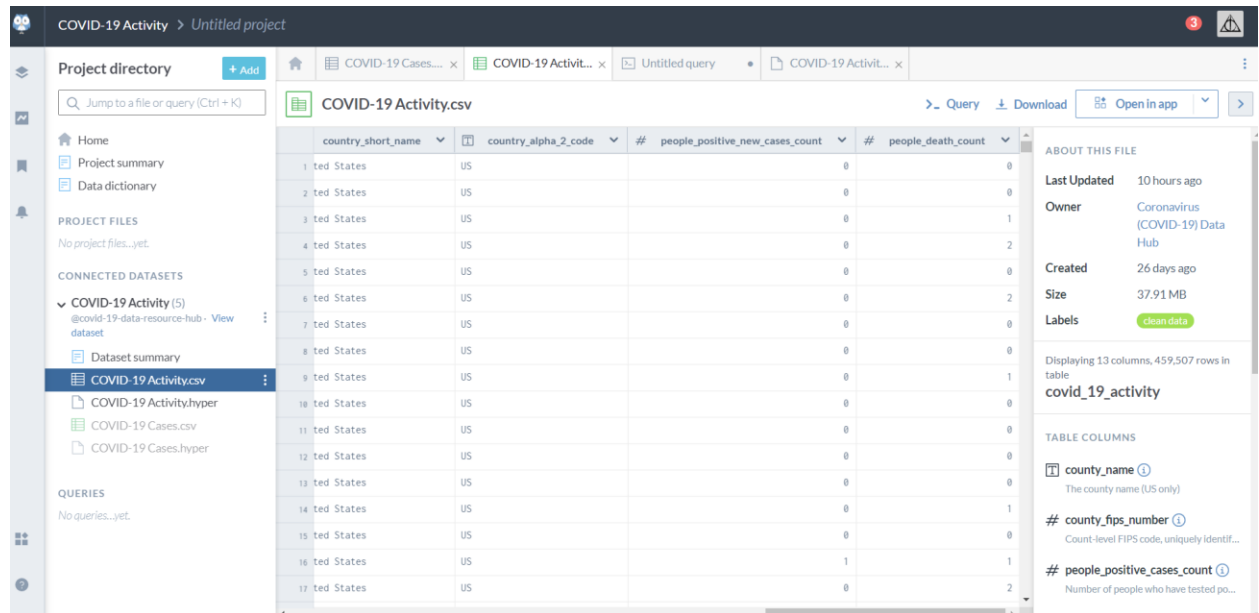
- World Health Organization (WHO): <https://www.who.int/>
- DXY.cn. Pneumonia. 2020. <http://3g.dxy.cn/newh5/view/pneumonia>.
- BNO News: <https://bnonews.com/index.php/2020/02/the-latest-coronavirus-cases/>
- National Health Commission of the People's Republic of China (NHC): http://www.nhc.gov.cn/xcs/yqtb/list_gzbd.shtml
- China CDC (CCDC): <http://weekly.chinacdc.cn/news/TrackingtheEpidemic.htm>
- Hong Kong Department of Health: <https://www.chp.gov.hk/en/features/102465.html>
- Macau Government: <https://www.ssm.gov.mo/portal/>
- Taiwan CDC: <https://sites.google.com/cdc.gov.tw/2019ncov/taiwan?authuser=0>
- US CDC: <https://www.cdc.gov/coronavirus/2019-ncov/index.html>
- Government of Canada: <https://www.canada.ca/en/public-health/services/diseases/coronavirus.html>
- Australia Government Department of Health: <https://www.health.gov.au/news/coronavirus-update-at-a-glance>
- European Centre for Disease Prevention and Control (ECDC): <https://www.ecdc.europa.eu/en/geographical-distribution-2019-ncov-cases>
- Ministry of Health Singapore (MOH): <https://www.moh.gov.sg/covid-19>
- Italy Ministry of Health: <http://www.salute.gov.it/nuovocoronavirus>

3.2 Tableau COVID-19 Data Resource Hub:

In this dataset, they provided the data of new cases every day that we can sketch a time series to track the rise of new cases.

COVID-19 case data can be directly downloaded or accessed through a Web Data Connector from data.world, a platform for data that enables users to post, search, and collaborate on data sets on a large and meaningful scale.

<https://data.world/covid-19-data-resource-hub/>



The screenshot displays the Tableau interface for a project named "COVID-19 Activity". The main view shows a table with the following columns: country_short_name, country_alpha_2_code, # people_positive_new_cases_count, and # people_death_count. The table contains 17 rows of data, all for the United States (US). The right-hand panel provides details about the file, including its last updated time (10 hours ago), owner (Coronavirus (COVID-19) Data Hub), created time (26 days ago), size (37.91 MB), and a "clean data" label. It also lists the table columns: county_name, county_fips_number, and people_positive_cases_count.

	country_short_name	country_alpha_2_code	# people_positive_new_cases_count	# people_death_count
1	ted States	US	0	0
2	ted States	US	0	0
3	ted States	US	0	1
4	ted States	US	0	2
5	ted States	US	0	0
6	ted States	US	0	2
7	ted States	US	0	0
8	ted States	US	0	0
9	ted States	US	0	1
10	ted States	US	0	0
11	ted States	US	0	0
12	ted States	US	0	0
13	ted States	US	0	0
14	ted States	US	0	1
15	ted States	US	0	0
16	ted States	US	1	1
17	ted States	US	0	2

3.3 Ministry of Health of Vietnam:

The Ministry of Health has published a dashboard about Covid-19 in Vietnam that they have built by Google Data Studio. The dashboard has many detailed

information about the patient such as their gender, city, nationality, age.



But the problem is that they did not publicized the data, so our group have to crawl the data from their website by Python (.ipynb file in Github repository):

```

In [4]: from selenium import webdriver
        from selenium.webdriver.common.keys import Keys
        from bs4 import BeautifulSoup
        import re
        import pandas as pd
        import os

        from selenium.webdriver.chrome.options import Options
        opts = Options()
        opts.add_argument("user-agent=vietyv")

In [13]: driver = webdriver.Chrome("C:\Program Files (x86)\Google\Chrome\Application\chromedriver.exe",options=opts)

In [14]: url = "https://ncov.moh.gov.vn/"
        driver.get(url)
        content = driver.page_source
        soup = BeautifulSoup(content)

In [15]: table=soup.find_all(id='sailorTable',attrs={'class':"table table-striped table-covid19"})

In [16]: tinh_thanh_pho=[]
        so_ca_nhiem=[]
        dang_dieu_tri=[]
        khoi=[]
        tu_vong=[]
        data_1=table[0].find('tbody') #this is table 1

        #Navigate the data from table 1 and fit them to the lists above
        for row in data_1.find_all('tr'):
            tinh_thanh_pho.append(row.find_all('td')[0].string)
            so_ca_nhiem.append(row.find_all('td')[1].string)
            dang_dieu_tri.append(row.find_all('td')[2].string)
            khoi.append(row.find_all('td')[3].string)
            tu_vong.append(row.find_all('td')[4].string)

In [17]: #Create a pd dataframe for table 1
        df_1=pd.DataFrame({'Province':tinh_thanh_pho,'Confirm':so_ca_nhiem,'Active':dang_dieu_tri,'Recovered':khoi,'Death':tu_vong})

In [18]: #Extract the data to a csv file

```

After run the code, we have 2 extracted datasets - covid19_province.csv and covid19_patient.csv (on Github repository)

3.4 Vietnamese Geo-data:

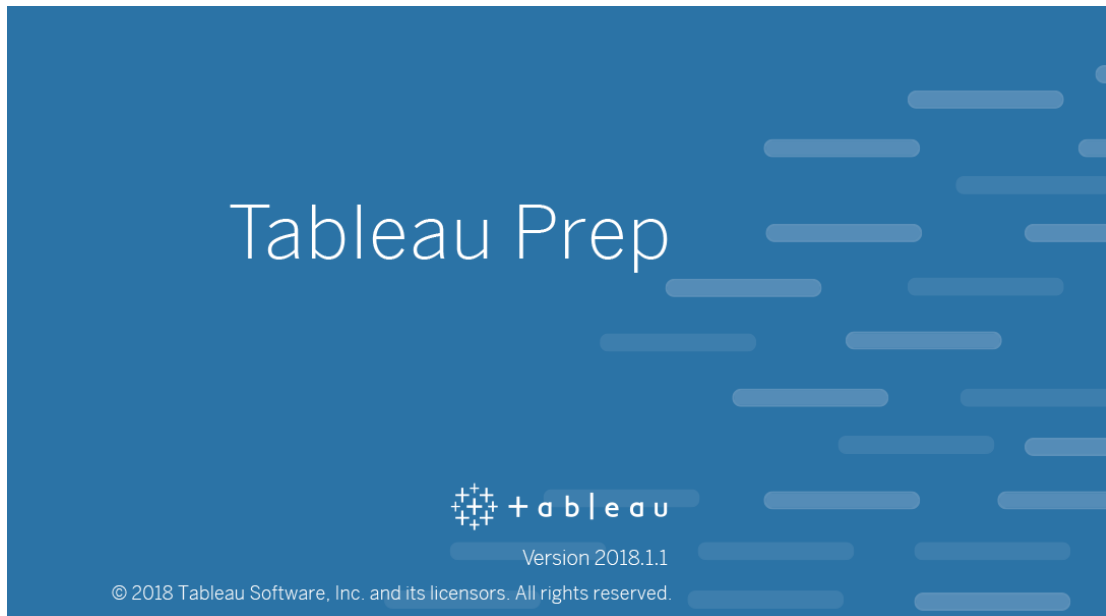
To construct the map of Covid-19, we need a GeoJSON file that has been designed for represent geographical features. In this case, the json file is going to be used to construct the map of Vietnam.

Center for Applied GIS of Ho Chi Minh City (HCMGIS) has provided Vietnam's GeoJSON file (<http://opendata.hcmgis.vn/>; <https://portal.hcmgis.vn/>). The file also has been provided by Dr. Tran Tung Thanh in lab 7 of the course Data Science and Data Visualization.

4. Data Processing:

Some of the collected data is complicated and hard for us to process and visualize it. So, we need the preprocessing step to clean, reduce non-use attribute, transformation data.

In this step, we mainly use the tool Tableau Prep Builder from Tableau Software to prepare data.

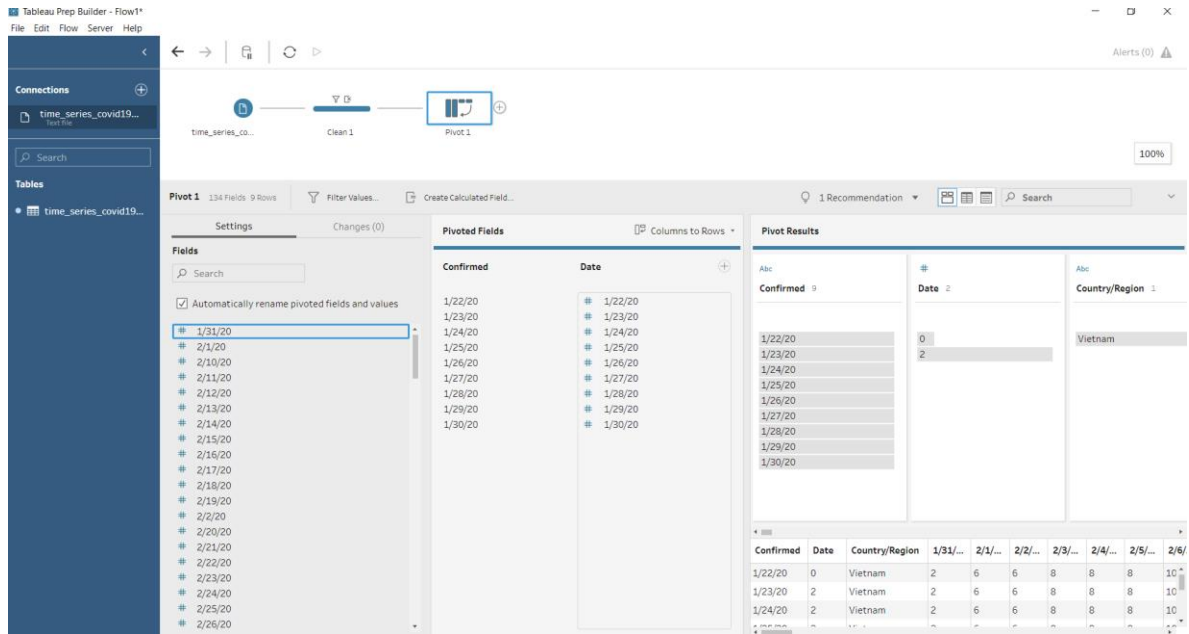


4.1 The data from CSSE JHU and Tableau Data Resource Hub:

With the raw data from JHU

(https://raw.githubusercontent.com/CSSEGISandData/COVID-19/8ff4ed75c9192f9f04766054f4b1e54add9b7cea/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv and https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_recovered_global.csv and <https://data.world/covid-19-data-resource-hub/covid-19-case-counts/workspace/file?filename=COVID-19+Cases.csv>). There are data from many countries, and confirmed, recovered cases of each day are recorded on each column. We need to pivot the columns into rows, and filter out the data that is not necessary for our Data Visualization, keep only Vietnam's data, then join their three together.

Filter and pivot: That two steps are easily done by Tableau Prep Builder



After join step, we have the final data covid19_vietnam.csv (Github repository):

1	Date	Type	Case
2	1/22/2020	Confirmed	0
3	1/22/2020	New Cases	0
4	1/22/2020	Recovered	0
5	1/23/2020	Confirmed	2
6	1/23/2020	New Cases	0
7	1/23/2020	Recovered	0
8	1/24/2020	Confirmed	2
9	1/24/2020	New Cases	2
10	1/24/2020	Recovered	0
11	1/25/2020	Confirmed	2
12	1/25/2020	New Cases	0
13	1/25/2020	Recovered	0
14	1/26/2020	Confirmed	2
15	1/26/2020	New Cases	0
16	1/26/2020	Recovered	0
17	1/27/2020	Confirmed	2
18	1/27/2020	New Cases	0
19	1/27/2020	Recovered	0
20	1/28/2020	Confirmed	2
21	1/28/2020	New Cases	0
22	1/28/2020	Recovered	0
23	1/29/2020	Confirmed	2
24	1/29/2020	New Cases	0

4.2 The data from the Ministry of Health of Vietnam:

To have the connection related to the GeoJSON file, from the raw file that we crawl from the website, we have to fill a new column which is 'ma' represents the ID of each province related to the json file.

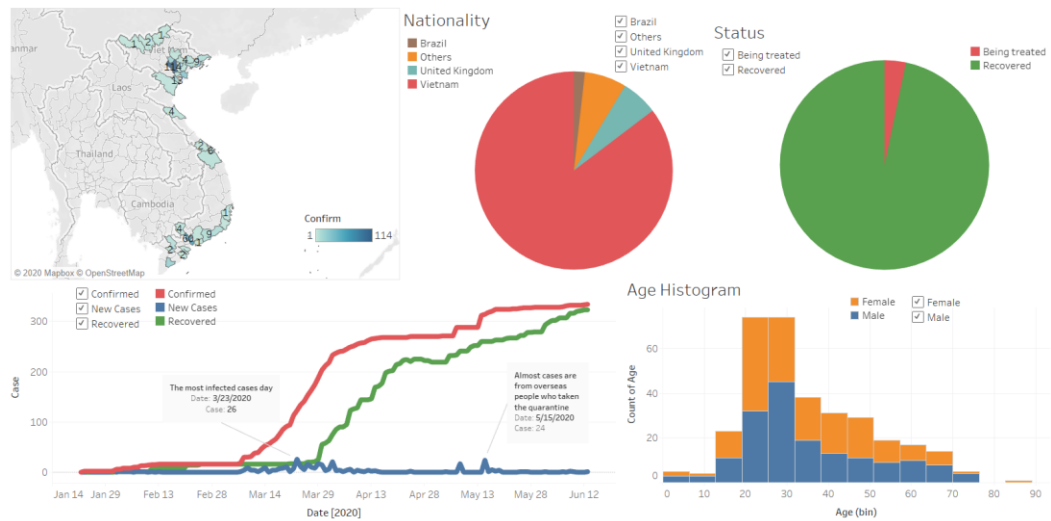
Q Search this file...

	ma	Province	Confirm	Active	Recovered	Death
1	1	Hà Nội	114	1	113	0
2	79	Hồ Chí Minh	60	6	54	0
3	34	Thái Bình	30	14	16	0
4	95	Bạc Liêu	21	0	21	0
5	26	Vĩnh Phúc	19	0	19	0
6	37	Ninh Bình	13	0	13	0
7	60	Bình Thuận	9	0	9	0
8	22	Quảng Ninh	8	0	8	0
9	48	Đà Nẵng	6	0	6	0
10	87	Đồng Tháp	5	0	5	0
11	30	Hải Dương	4	1	3	0
12	35	Hà Nam	4	0	4	0
13	24	Bắc Giang	4	0	4	0
14	42	Hà Tĩnh	4	0	4	0
15	72	Tây Ninh	4	0	4	0
16	49	Quảng Nam	3	0	3	0
17	38	Thanh Hóa	3	0	3	0
18	46	Thừa Thiên Huế	2	0	2	0
19						

5. Visualization Design:

Refer to some great visualizations that are published on the Internet, such as the dashboard from CSSE, The Ministry of Health of Vietnam, Tableau Public,... we have designed a prototype visualization by using Tableau - a powerful BI tool.

COVID-19 EPIDEMIC in VIETNAM



Timeseries Map Age Dashboard Status Nationality

The design has 5 main charts, a map to represent the confirmed case base on each province; a time-series line chart to illustrate the number of confirmed, recovered and new cases by time from 22/01/2020 to 13/06/2020; a histogram that shows confirmed figures by age with bins equal to 10, beside it also shows the sexuality of each cases by stacked bars; and two pie charts that give information about patient's status, and nationality.

The design has many interactive features, such as mouse on, mouse out, zooming, filtering, selecting, brushing... By mouse on and mouse out event, the charts can show the tooltip. In addition, each chart has the filter to see the details of each condition, the map also has the zoom in and zoom out feature.

The prototype also has some details such as legends, annotation, title.

6. Our D3.js Web-Based Interactive Visualization:

6.1 About D3.js:



Data-Driven Documents

D3.js stands for Data-Driven Documents is a front-end visualization library in javascript for creating interactive and dynamic web-based data visualizations, it also has an active community behind it for which it is very famous. It uses HTML, CSS a, and SVG to bring data to life and mainly it is for manipulating DOM objects, focusing on objects and concepts, not just the pre-built charts and graphs. It is mostly compatible with popular web browsers, like Chrome or Firefox.

It can even create different shapes arcs, lines, arcs, rectangles, and points. The essential feature of d3.js that it provides the beautiful fully customized visualizations. It is a suite of small modules for data visualization and analysis. These modules work well together, but we should pick and choose the parts that we only need. D3's most complex concept is selections and if we are using a virtual DOM framework like React (and don't need transitions), so don't need selections at all.

6.2 D3.js and our project:

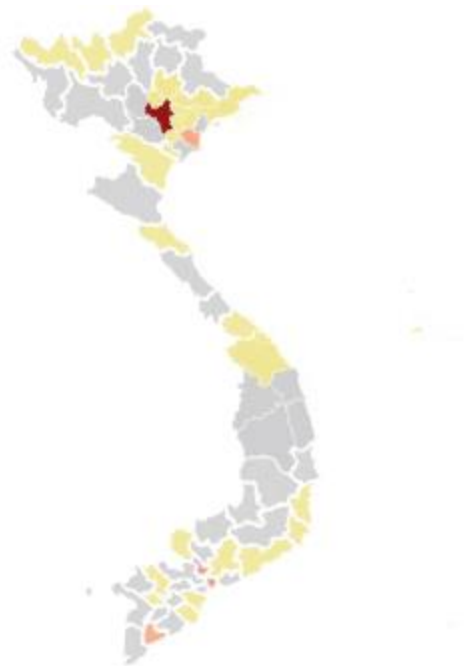
6.2.1 Confirmed Choropleth Map:

Description: Represents the confirmed case based on province.

Library: D3.js, Susie Lu's D3 SVG legend (<https://d3-legend.susielu.com/>)

Data: covid19_province.csv and the GeoJSON file.

Features: Zooming, Dragging , Showing Tooltip



Code:

The code to connect .csv and GeoJSON file by the key “ma” and “Ma” that represent the province ID:

```

80
81 d3.csv("https://raw.githubusercontent.com/dangquanghung/covid/master/covid19_province.csv", function(data) {
82     //Set input domain for color scale
83     color.domain([
84         d3.min(data, function(d) { return d.Confirm; }), 121
85     ]);
86
87
88
89 d3.json("https://raw.githubusercontent.com/TungTh/tungth.github.io/master/data/vn-provinces.json", function(json) {
90
91     for (var i = 0; i < data.length; i++) {
92         //Grab state name
93         var dataState = data[i].ma;
94         //Grab data value, and convert from string to float
95         var dataValue = parseFloat(data[i].Confirm);
96
97         //Find the corresponding state inside the GeoJSON
98         for (var j = 0; j < json.features.length; j++) {
99             var jsonState = json.features[j].properties.Ma;
100             if (parseFloat(dataState) == parseFloat(jsonState) ) {
101                 //Copy the data value into the JSON
102                 json.features[j].properties.Confirm = dataValue;
103
104                 //Stop looking through the JSON
105                 break;
106             }
107         }
108     }
109 }

```

Create the legend:

```
138 //create legend
139
140 svg.append("g")
141   .attr("class", "legend")
142   .attr("transform", "translate(20,20)")
143   .attr("width", width)
144   .attr("height",height);
145
146 var legend = d3.legendColor()
147   .shapeWidth(30)
148   .cells(5)
149   .orient('vertical')
150   .scale(color);
151
152 svg.select(".legend")
153   .call(legend);
154
```

Create zoom feature:

```

159 var zooming = function(d) {
160     //Log out d3.event, so you can see all the goodies inside
161     console.log(d3.event.transform);
162     //Get the current (pre-dragging) translation offset
163
164     var offset = [d3.event.transform.x, d3.event.transform.y];
165
166     //Calculate new scale
167     var newScale = d3.event.transform.k * 2000;
168
169     //Update projection with new offset and scale
170     projection.translate(offset)
171     .scale(newScale);
172
173     svg.selectAll("path")
174         .attr("d", path);
175     svg.selectAll("circle")
176         .attr("cx", function(d) {
177             return projection([d.lon, d.lat])[0];
178         })
179         .attr("cy", function(d) {
180             return projection([d.lon, d.lat])[1];
181         });
182 }
183
184
185 var zoom = d3.zoom()
186 .on("zoom", zooming);
187

```

Tooltip calling:


```

206 var mouseover = function(d) {
207   d3.select(this).transition()
208     .duration('200')
209     .style("opacity", 1)
210     .style("stroke","white")
211     .style("stroke-width",3)
212
213   tooltip1.transition()
214     .duration(100)
215     .style("opacity", 1)
216
217   tooltip1.html("Province: " + d.properties.Ten + "<br>" + "Confirmed cases: " + d.properties.Confirm)
218     .style("left", (d3.event.pageX + 10) + "px")
219     .style("top", (d3.event.pageY - 15) + "px")
220
221   };
222
223 var mouseout = function(d) {
224   d3.select(this).transition()
225     .duration('200')
226     .style("opacity", 0.9)
227     .style("stroke","white")
228     .style("stroke-width",0.3);
229
230
231
232   tooltip1.transition()
233     .duration('100')
234     .style("opacity", 0)
235     .style("stroke","white")
236     .style("stroke-width",0.3);
237

```

Sketch the choropleth:

```

241   svg.selectAll("path")
242     .data(json.features)
243     .enter()
244     .append("path")
245     .attr("d", path)
246     .style("fill", function(d) {
247       //Get data value
248       var value = d.properties.Confirm;
249       if (value) {
250         //If value exists...
251         return color(value);
252       } else {
253         //If value is undefined...
254         return "#ccc";
255       }
256     })
257     .style('stroke', 'white')
258     .style('stroke-width', 1.5)
259     .style("opacity",0.8)
260     .on('mouseover', mouseover)
261     .on("mouseout", mouseout);
262 })

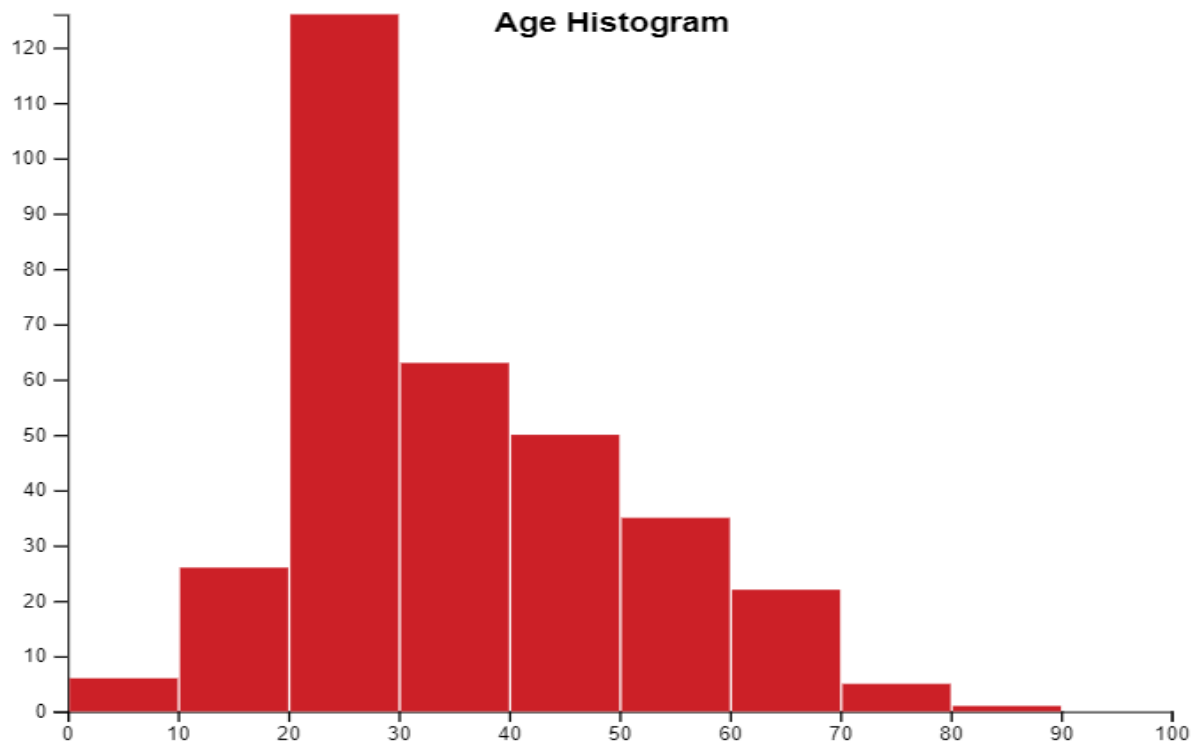
```

6.2.2 Age Histogram:

Description: shows confirmed figures by age with bins, from 0 to 100.

Library: D3.js

Data: covid19_patient.csv



Code:

Create Scales:

```
28     var x = d3.scaleLinear()
29     .domain([0, 100])    // can use this instead of 1000 to have
30     .range([0, width]);
31     svg.append("g")
32     .attr("transform", "translate(0," + height + ")")
33     .call(d3.axisBottom(x));
34
35     // Y axis: initialization
36     var y = d3.scaleLinear()
37     .range([height, 0]);
```

Set parameters for the histogram and get the bins:

```

40 // A function that builds the graph for a specific value of bin
41 function update(nBin) {
42
43     // set the parameters for the histogram
44     var histogram = d3.histogram()
45         .value(function(d) { return d.Age; }) // I need to give the vector of value
46         .domain(x.domain()) // then the domain of the graphic
47         .thresholds(x.ticks(nBin)); // then the numbers of bins
48
49     // And apply this function to data to get the bins
50     var bins = histogram(data);
51
52     // Y axis: update now that we know the domain
53     y.domain([0, d3.max(bins, function(d) { return d.length; })]); // d3.hist has to be called before the Y axis obviously
54     yAxis
55         .transition()
56         .duration(100)
57         .call(d3.axisLeft(y));
58

```

Append rect:

```

60 var u = svg.selectAll("rect")
61     .data(bins)
62
63 // Manage the existing bars and eventually the new ones:
64 u
65     .enter()
66     .append("rect") // Add a new rect for each new elements
67     .merge(u) // get the already existing elements as well
68     .transition() // and apply changes to all of them
69     .duration(100)
70     .attr("x", 1)
71     .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
72     .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
73     .attr("height", function(d) { return height - y(d.length); })
74     .attr("fill", "#CC2027");

```

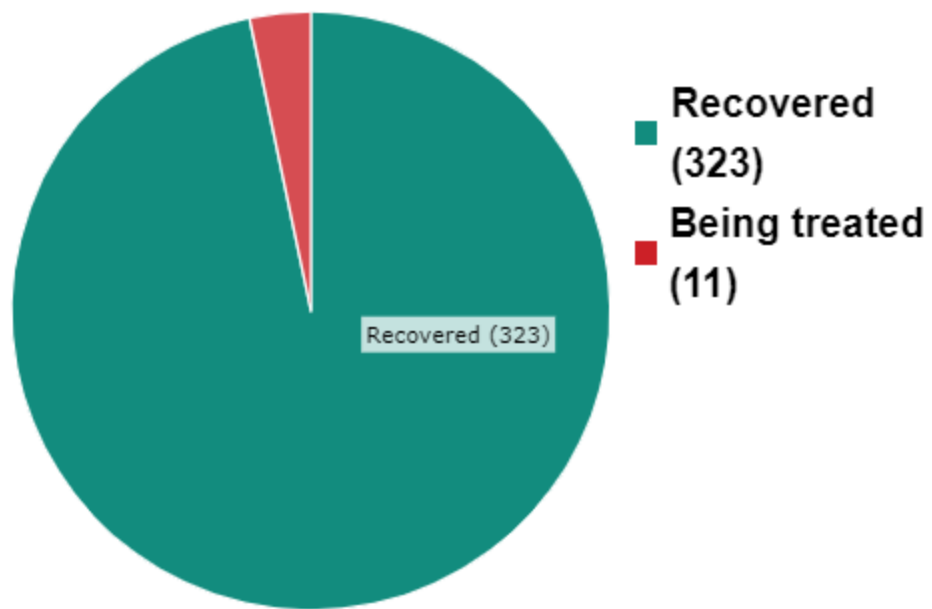
6.2.3 Patient's Status Pie Chart:

Description: proportion of status of patient who are recovered or being treated.

Library: D3.js

Data: covid19_status.csv

Feature: tooltip



Code:

Append the pie

```

30  var arc = d3.arc()
31  .innerRadius(0)
32  .outerRadius(radius);
33
34  var pie = d3.pie()
35  .value(function(d) { return d.Value; })
36  .sort(null);
37
38  var path = g.selectAll('path1')
39  .data(pie(data))
40  .enter()
41  .append("g")
42  .append('path')
43  .attr('d', arc)
44  .attr('fill', (d,i) => color(i))
45  .style('opacity', opacity)
46  .style('stroke', 'white')
47  .on("mouseover", function(d) {
48      d3.selectAll('path')
49      .style("opacity", otherOpacityOnHover);
50      d3.select(this)
51      .style("opacity", opacityHover);
52
53      let g = d3.select("svg")
54      .style("cursor", "pointer")

```

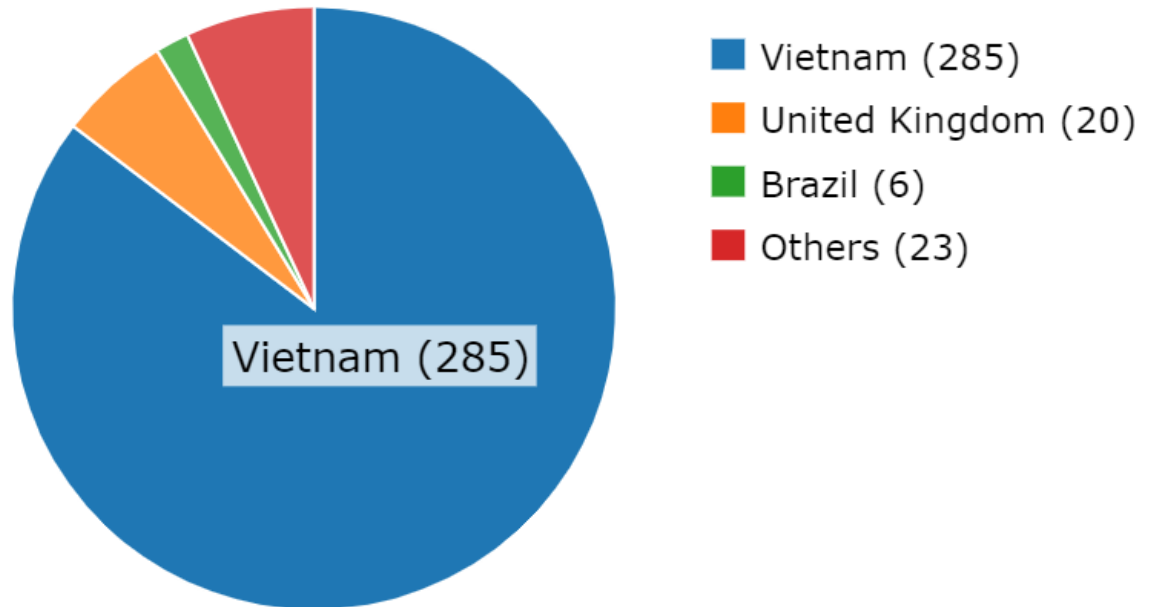
6.2.4 Patient's Nationality Pie Chart:

Description: proportion of nationality of patients.

Library: D3.js

Data: covid19_nationality.csv

Feature: tooltip



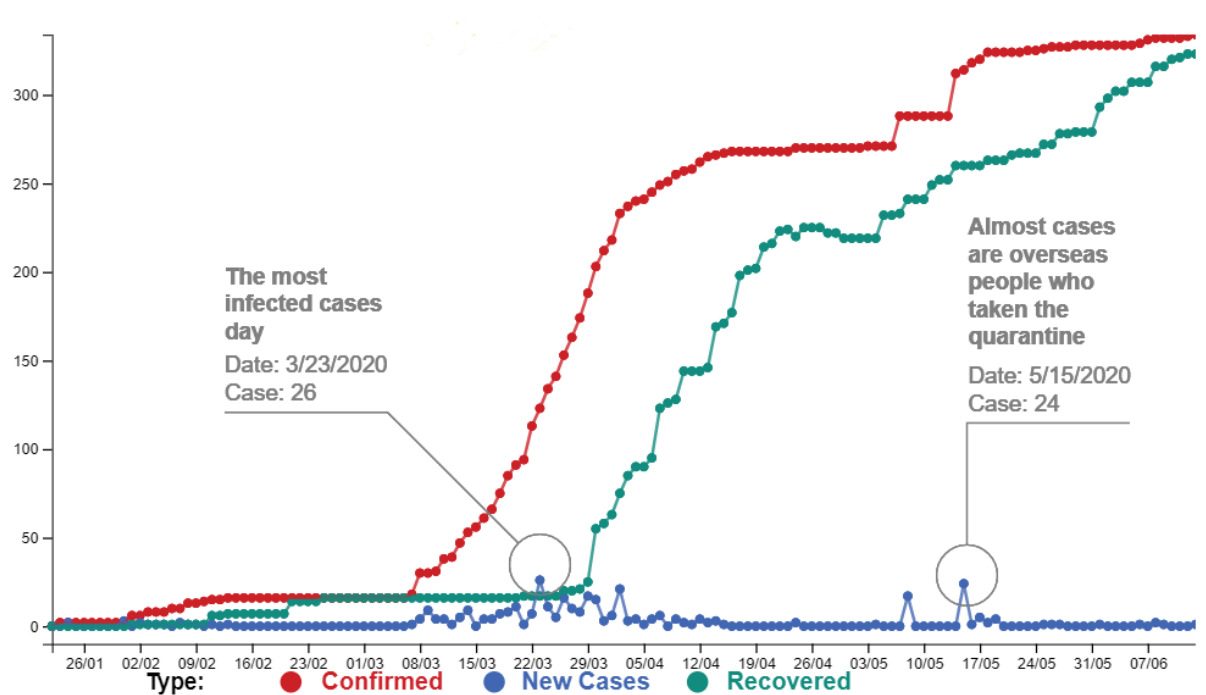
Code:

Append the pie:

```
25 var g = svg.append('g')
26 .attr('transform', 'translate(' + (width/2) + ',' + (height/2) + ')');
27
28 var arc = d3.arc()
29 .innerRadius(0)
30 .outerRadius(radius);
31
32 var pie = d3.pie()
33 .value(function(d) { return d.Number; })
34 .sort(null);
35
36 var path2 = g.selectAll('path2')
37 .data(pie(data))
38 .enter()
39 .append("g")
40 .append('path')
41 .attr('d', arc)
42 .attr('fill', (d,i) => color(i))
43 .style('opacity', opacity)
44 .style('stroke', 'white')
45 .on("mouseover", function(d) {
46     d3.selectAll('path2')
47     .style("opacity", otherOpacityOnHover);
48     d3.select(this)
49     .style("opacity", opacityHover);
50
51     let g = d3.select("svg")
52     .style("cursor", "pointer")
53     .append("g")
54     .attr("class", "tooltip")
55     .style("opacity", 0);
56
```

6.4.5 Time-series Line Chart:

Description: illustrate number of confirmed, recovered, new cases by time from 22/01/2020 to 12/06/2020.



Data: covid19_vietnam.csv

Library: D3.js, Susie Lu's D3 SVG Annotation (<https://d3-annotation.susielu.com/>)

Code:

Formatting time and RowConverter:

```

8      //For converting Dates to strings
9      var formatTime = d3.timeFormat("%d/%m");
10
11     var rowConverter = function(d) {
12         return {
13             type: d.Type,
14             date: new Date(d.Date),
15             //Convert from string to float
16             cases: parseFloat(d.Case)
17         };
18     }
19

```

Create Scale:


```

36 xScale = d3.scaleTime()
37   .domain([
38     d3.min(data, function(d) { return d.date; }),
39     d3.max(data, function(d) { return d.date; })
40   ])
41   .range([padding, w]);
42
43 yScale = d3.scaleLinear()
44   .domain([
45     d3.min(data, function(d) { if (d.cases >= 0) return d.cases; }) - 10,
46     d3.max(data, function(d) { return d.cases; })
47   ])
48   .range([h - padding, 0]);
49
50 colorScale = d3.scaleOrdinal()
51   .domain(["Confirmed", "Recovered", "New Cases"])
52   .range(["#CC2027", "#128C7E", "#4267b2"]);
53

```

Define line generators:

```

64
65 //Define line generators
66 line = d3.line()
67   .x(function(d) { return xScale(d.date); })
68   .y(function(d) { return yScale(+d.cases); });
69
70

```

Create Tooltip:

```

78 var mouseover = function(d) {
79
80   var xPosition = parseFloat(d3.select(this).attr("cx")) + padding;
81   var yPosition = parseFloat(d3.select(this).attr("cy")) / 2 + h;
82
83   //Update the tooltip position and value
84   d3.select("#tooltip")
85     .style("left", xPosition + "px")
86     .style("top", yPosition + "px")
87     .select("#type")
88     .text("Date: " + formatTime(d.date) + " - type: " + d.type + " - Case: " + d.cases);
89
90   //Show the tooltip
91   d3.select("#tooltip").classed("hidden", false);
92 }
93
94
95 var mouseout = function(d) {
96   d3.select("#tooltip").classed("hidden", true);
97 }
98

```

Draw the lines and the circles along:

```

132     let lines = linechart.append('g')
133         .attr('class', 'lines');
134
135
136     legendSpace = w/dataset.length; // spacing for legend
137
138     // Loop through each symbol / key
139     dataset.forEach(function(d,i) {
140
141         linechart.append("path")
142             .attr("class", "line")
143             .style("stroke", function() {
144                 return d.colorScale = colorScale(d.key); })
145             .attr("id", 'tag'+d.key.replace(/\s+/g, '')) // assign ID
146             .attr("d", line(d.values));
147
148         //Create the dots on the lines
149         lines.selectAll("circle-group")
150             .data(dataset).enter()
151             .append("g")
152             .style("fill", (d, i) => colorScale(d.key))
153             .selectAll("circle")
154             .data(d => d.values)
155             .enter()
156             .append("g")
157             .attr("class", "circle")
158             .append("circle")
159             .attr("cx", d => xScale(d.date))
160             .attr("cy", d => yScale(d.cases))
161             .attr("r", 3)
162             .on("mouseover", mouseover )
163             .on("mouseout", mouseout );

```

Create the legend:

```

165     //Add the legend
166     linechart.append("circle")
167         .attr("cx", (legendSpace)+i*legendSpace/2-60)
168         .attr("cy", h - 25) // 100 is where the first dot appears. 25 is the distance between dots
169         .attr("r", 7)
170         .style("fill",function() { // dynamic colours
171             return d.colorScale = colorScale(d.key);});
172
173
174     // Add the Legend
175     linechart.append("text")
176         .attr("x", (legendSpace)+i*legendSpace/2) // spacing
177         .attr("y", h - 20) // *****
178         .attr("class", "legend")
179         .attr("font-family", "sans-serif")
180         .attr("font-size", "15px") // style the legend
181         .style("fill", function() { // dynamic colours
182             return d.colorScale = colorScale(d.key); })
183         .text(d.key);
184

```

Create the annotation:

```

187  const annotations1 = [
188    {
189      note: {
190        label: "Date: 3/23/2020",
191        bgPadding: 20,
192        title: "The most infected cases day"
193      },
194      x :380,
195      y:350,
196      dy: -100,
197      dx: -100
198    }
199  ]
200  const makeAnnotations = d3.annotation()
201                          // .editMode(true)
202                          .type(type)
203                          .annotations(annotations1)
204
205
206      d3.select("svg")
207      .append("g")
208      .attr("class", "annotation-group")
209      .call(makeAnnotations)

```

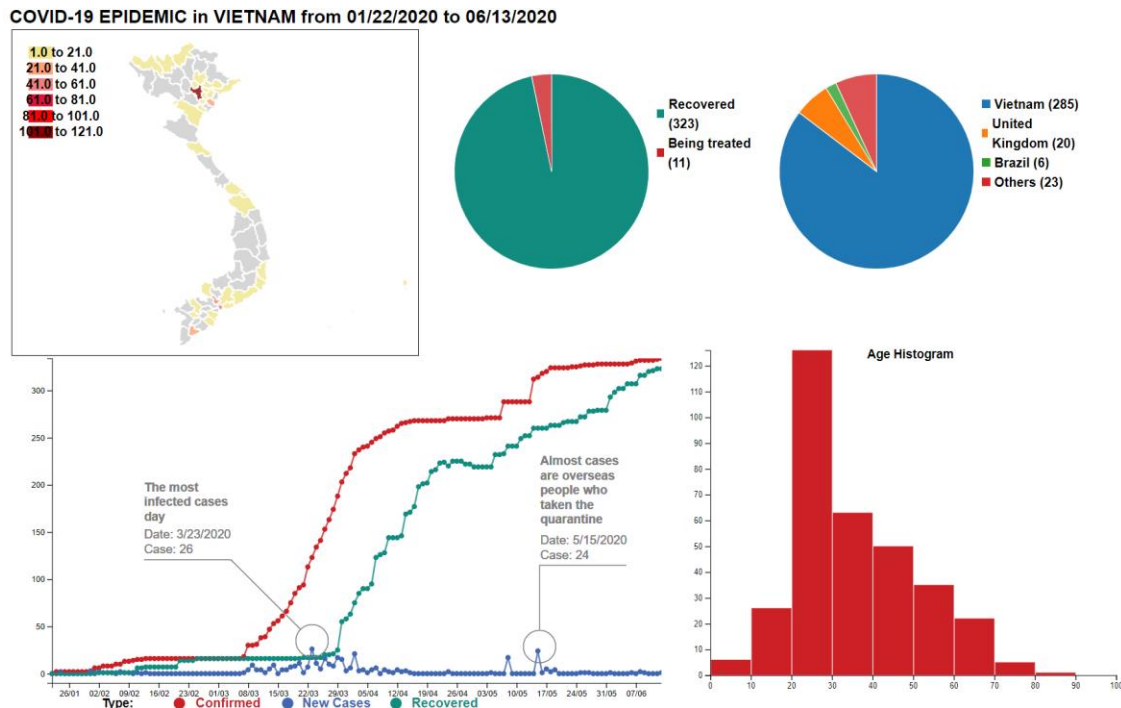
```

211  const annotations2 = [
212    {
213      note: {
214        label: "Date: 5/15/2020 ",
215        bgPadding: 20,
216        title: "Almost cases are overseas people who taken the quarantine "
217      },
218      x :680,
219      y:350,
220      dy: -100,
221      dx: 0
222    }
223  ]
224  const Anno1 = d3.annotation()
225                // .editMode(true)
226                .type(type)
227                .annotations(annotations2)
228
229
230      d3.select("svg")
231      .append("g")
232      .attr("class", "annotation-group")
233      .call(Anno1)

```

7. Result- The Dashboard:

From these five charts, we have completed the final dashboard that gains many insights to the audience, gives information clearly, exactly about the epidemic in Vietnam.



Insight:

Corona's outbreak broke out in late December 2019 in China. But it was not until January 2020 that Vietnam began to record the first cases. In general, the prevention method in Vietnam is very good. According to the international report, Vietnam is one of the leading countries on Coronavirus prevention.

First, according to international reports, cases usually range from 50-60 years old and suffer from previous diseases (hepatitis, lung) but according to the chart of the most affected age in Vietnam (more than 120 people) is 20-30 years old. Next is 30-40 years old. And the 80-90 age range is less than 10 cases. Second, according to the time series, most cases are from abroad (according to the "No one is left behind" policy) and the most infected are international students who are returning from abroad and have aged within 20-40.

As I mentioned above, Vietnam is one of the leading countries to prevent the disease. As we can see, more than 323 cases were cured out of a total of 334 cases. Along with the efforts of Vietnamese doctors, Vietnamese people have also complied with the provisions of the Government to unite against the disease. Of the more than 323 cases recovered, we can see case number 19 (uncle of case number 17), and the case of the British pilot miraculously recovering. All are based on the efforts of the Vietnamese doctors.

While developed countries like the UK and the US, they are losing control because of the Corona epidemic (the US has more than 2,000,000 cases). And Vietnam has only over 300 cases while 90% are cured. This makes the world have a different perspective on Vietnamese medicine in particular and the Vietnamese state in general. It also helps to strengthen our position on the world map. It can also lead to other positive aspects such as the economy will develop, trade will promote (Some international brands will choose Vietnam to be a supplier), etc.

From the pie chart of patients' nationality, we can see about 85% of confirmed cases are Vietnamese, and 15% are foreigners. All patients have paid a very low cost to treat the virus, but in other countries, it may take \$10,000 for a treatment.

Looking at the chart time series, we can see more clearly how to effectively prevent disease in Vietnam. And fortunately as well as show our preparation is that Vietnam is not infected in a rush, new cases are detected very little. When at most one day nearly 20 cases (quarantined immediately after arriving at the airport). And within 2 weeks there were no cases (4/16 - 7/5). We can see the two lines Confirmed and Recovered, once again it shows the advance of Vietnamese doctors when both lines are translating together. There is no case of an increase or decrease. (Ex: confirmed increase but recovered reduced). From linechart newcases, we can see that the covid 19 epidemic in Vietnam has experienced two outbreaks, the first one (March 7 - April 15) at this time the epidemic is strongly outbreaks in China, South Korea and Europe. Vietnam also fought the outbreak and an ongoing struggle, the Party, the Government, the medical team, the doctors and the entire people, Vietnam won the epidemic. In the second stage (May 7 to May 16), all citizens coming back from abroad were concentratedly quarantined and all Covid positive cases Covid-19 were from isolation areas, there were no cases of internal infection. community. After 2 epidemics, it can be said that Vietnam has officially won the epidemic.

According to the choropleth, it is about $\frac{1}{3}$ province had the positive case with COVID-19, Hanoi and Ho Chi Minh City had the most confirmed cases because Noi Bai and Tan Son Nhat international airport receive oversea people, and almost higher hospitals are in that city.

8. Development directions in the future:

Actually, because of lack of knowledge and time, our project has less features than the prototype. In the future, more features like synchronized filters, brushing, transition in each change, a stacked histogram which can show the proportion about female and male,... will be added to the interactive visualization.

9. References:

Data:

<https://data.world/covid-19-data-resource-hub/>

https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data

<https://ncov.moh.gov.vn/>

<http://opendata.hcmgis.vn/>

<https://portal.hcmgis.vn/>

Dashboard:

<https://ncov.vncdc.gov.vn/>

<https://covid19.who.int/>

<https://coronavirus.jhu.edu/map.html>

<https://public.tableau.com/profile/covid.19.data.resource.hub#!/>

News:

<https://www.bbc.com/news/world-asia-52628283>

<https://www.aa.com.tr/en/asia-pacific/vietnam-a-success-story-in-fight-against-covid-19/1866670>

<https://ncov.moh.gov.vn/-/tai-sao-viet-nam-chien-thang-dich-covid-19->

<https://www.nytimes.com/news-event/coronavirus>

<https://www.who.int/emergencies/diseases/novel-coronavirus-2019/events-as-they-happen>

10. Conclusion:

In this project of the course “Data Science and Data Visualization”, we have designed and implemented a web-based interactive visualization about COVID-19 Epidemic in Vietnam. The visualization provided enough insights and information to primarily answer the questions about COVID-19: HOW DOES COVID-19 AFFECT VIETNAM? and HOW IS COVID-19 CONTROLLED IN VIETNAM?

