# COVID-19 IN VIETNAM VISUALIZATION

**Trần Thị Mai Hương, Trần Thị Huệ, Bùi Tuấn Anh, Nguyễn Thế An**
GVHD: Bùi Quốc Khánh
Faculty Information of Technology, Hanoi University

***Abstract:* *This paper presents a contemporary visualization for COVID-19 disease status in Vietnam. By collecting records from February to June 2020 by ourselves, using python libraries to analyze data and draw, the research was measured to construct charts and maps about provinces in Viet Nam and particularly in Hanoi. We came to the following conclusion: At the moment of the research, there are just 4 provinces in Vietnam have patients, Thai Binh has the highest number of active infected people, the number of infected are decreasing day by day and prime-working age (25-54) is the most vulnerable age cluster in Vietnam. The work presented here has profound implications for future studies of disease statistics visualization and may one day help solve the problem of COVID disaster.***

***Keywords:*** COVID-19, Visualization, Vietnam, Matplotlib.

## I. INTRODUCTION

The first half of 2020 which is a 'terrible' with the worldwide pandemic of corona virus disease (COVID-19). The COVID-19 pandemic caused by the SARS-CoV-2 virus was first confirmed in Vietnam on January 23, 2020. [1] As of 06:00 on June 17, Vietnam recorded 335 cases, of which 325 patients were discharged from hospital and no deaths due to infection were recorded.

Phase 1, first 16 cases of Covid-19: On January 23, 2020 (i.e, the 29 of the Year of the Rat), Vietnam officially entered the war against Covid-19. Cho Ray Hospital (HCMC) confirmed the first two Covid-19 patients in Vietnam, who are father and son of Wuhan, China [2]. On February 12, 2020, Vinh Phuc Province decided to isolate the entire Son Loi commune [3].

Phase 2, Cases of invasion from abroad: On March 6, 2020, Hanoi announced the first case and was the 17th patient in Vietnam. On March 10, 2020: appeared "super infectious" patient in Binh Thuan - the 34th patient in Vietnam. The patient returned from the United States at Tan

Son Nhat airport and then took a private vehicle to Phan Thiet, infecting 11 others. On March 17, 2020, Vietnam temporarily suspended visas for foreigners on entry. On March 21, Vietnam temporarily stopped the entry of foreign visitors[4].

Phase 3, Risk of community spread: On March 20, 2020, the Ministry of Health announced two nurses from the Center for Tropical Diseases, Bach Mai Hospital infected with Covid-19, respectively 86 and 87 patients in Vietnam. On March 28, 2020, Bach Mai Hospital was blockaded. On March 31, 2020, the Prime Minister issued Directive No. 16 on the social isolation of the whole country within 15 days to prevent COVID-19, from 0h on April 1. On April 15, social isolation was extended with Hanoi, Ho Chi Minh City, and some provinces at high risk.

Phase 4, Combating long-term epidemic together with socio-economic development: From April 23, 2020, the country stopped social isolation but continues to ensure the prevention of epidemic rules. On April 25, 2020, the Prime Minister issued Directive 19 to continue measures to prevent and combat COVID-19 in the new situation [5].

The COVID-19 pandemic has placed figures, data and charts at the forefront of millions of people's everyday lives. Either monitoring public health obligations for diseases or simply taking a thorough interest in one of the biggest stories for more than a decade, there's a massive appetite for accurate data. Data are available that show the corona virus spread per capita, or per million people. But that could generate some potentially inaccurate results. A very small country with just a few infections would rate much higher than a bigger country that might struggle to contain the outbreak. A country's total population is not constrained by how quickly the virus spreads. One of which is certainly the number of tests the country is working out. While we have factored in the population, the number of confirmed cases still associated heavily with the sum of the research. Ideally, we want as much information as possible on COVID-19 tests in each province. There's no official database yet on this, though.

In this report, we used a more considerate approach for data collection to look at the current status of Corona virus in specific provinces. We note a few possible curves along the way and consider the intensity of COVID-19 in some provinces. However, the information also showed that we should be vigilant about any significant adjustments in terms of spread and data as well.

## II. METHODOLOGY

### *A. Dataset*

At the time of writing this report, one of the most referencing data sources is the Ministry of Health (Vietnam). The raw data is gathered manually day by day with the timeline on the source page. The dataset has the confirmed date, case, age, location, etc. which can be used remarkably for visualizing the COVID-19 information. In the overall country, Hanoi has the most confirmed case so that this city can be used categorically for visualizing each province has the confirmed case of the Vietnam map. The data has 328 cases and is clean, which is fitting for further research purposes.

### *B. Technology*

#### *1. Technology*

Our project is supported by many Google tools.

**Google Sheets** works as our database. Choosing an online database because it helps members in the team can add and update new records anytime and anywhere. Google Sheets is our choice because this company provides APIs for sheets. Time for making APIs will be saved and with API, we can interact with the database by code easily. Moreover, Google Sheets records all changes history, therefore if the data is lost accidentally, we can revert them.

**Google Colab** is a trustful IDE and Version Control Tool in our project. This Google Service provides a development environment for us to execute Python codes, indeed libraries can be installed in just one line of code. It is easier for the team leader to manage and control work as the code is saved automatically at the same time it is written.

Moreover, we use **Sublime Text** to show animation.

#### *2. Algorithms & Libraries*

**Request:** The requests module allows sending HTTP requests using Python, in our case we use it to interact with our Google Sheet.

**Pandas:** Pandas is a solid library written for Python to manipulate and analyze data. In our project, it is used for reading data from our Google Sheet and analyzing data before visualizing.

**Numpy:** NumPy is a library for the Python programming language, supports working with arrays.

**Matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. This library's functions allow us to turn data into graphs and charts.

**Geopy:** In order to draw patients' maps, longitude and latitude of patients' address is needed. We tried with Google Maps API. However, having an international banking account is a must to be allowed using Maps API. So that, finally we use geopy. geopy makes it easy for Python developers to locate the coordinates of addresses, cities, countries, and landmarks across the. Three most strong location services: Google Maps, Bing Maps, and Nominatim, has their own class in geopy.geocoders abstracting the service's API. We used geopy to get coordinates of cities and particularly districts in Hanoi where exist COVID patients.

**Folium:** folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library. Developers manipulate data in Python, then visualize it on a Leaflet map via folium. By providing coordinates for folium function, patients' maps are drawn.

*3. How to initialize*

Our team discussed to produce the best suitable work flow:

Build dataset **-->**Process data **-->**Visualize **-->** Assess the product

***Build dataset****:* We find data about COVID-19 patients on government's websites. With missing values, for example districts of Hanoi's patients, we read timelines on these websites and other trustful sources on the Internet to fill in.

***Process data:***We use Request library and Pandas DataFrame to read the data set. Based on the requirements of each part, we have a different way to select and manipulate appropriate data which can be presented more carefully in part 3 (implementation).

***Visualize****:* Data is turned to graphs, maps and charts thanks to visualization libraries.

***Assess the product:*** Graph, map and animation are analyzed carefully.

*4. Evaluation Criteria*

To determine whether visualization is good enough or not, we take turns to calculate various metrics such as accuracy, readability. Those metrics used as the evaluation criteria of our visualization product.

***Accuracy****:* Statistics on the map, graph and charts matches with the dataset.

***Readability****:* Focal attributes icons must have significant hue and acceptable size. Space in the visualization product must be appropriate.

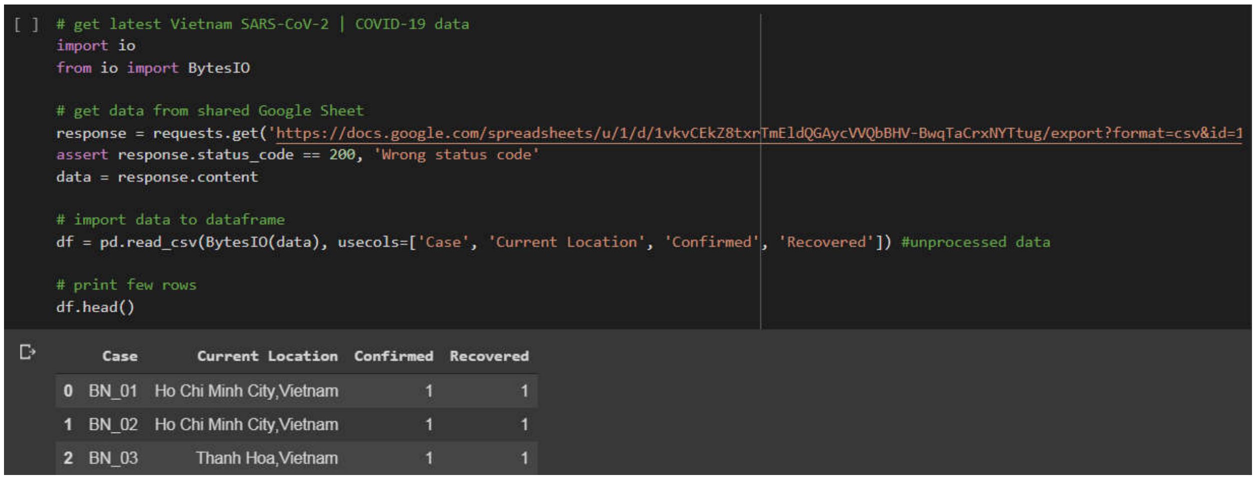
If all the metrics above are rated with a high score, the product is good. By contrary, the team must hold a meeting to find methods to level up product quality.

## III. IMPLEMENTATION

***Group's role***: There are 4 members in our group: Tran Thi Mai Huong, Tran Thi Hue, Bui Tuan Anh, Nguyen The An. All the team's members participate in building the data set. Huong carried out drawing national and Hanoi's patients maps, Hue visualizes Top 5 highest number of patients cities, districts in Hanoi bar chart and animation. Tuan Anh codes the age clusters pie chart and An takes responsibility for new cases per day bar chart. After we have finished coding, we write a report together to explain the project's purpose, work and analyze the result.

*A. National and Hanoi Map*

After installing libraries and import necessary packages, we get data from our shared Google Sheet by using Requests and Pandas. Based on each parts, columns of interests are chosen.

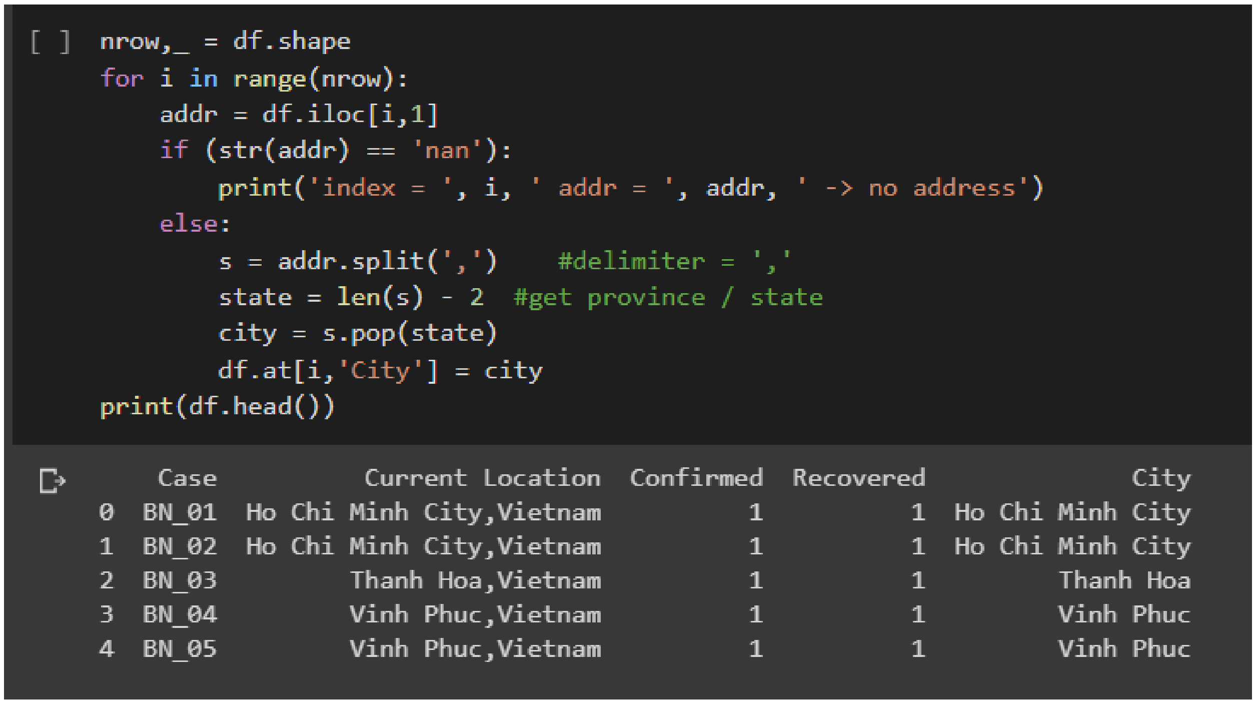


```
[ ] # get latest Vietnam SARS-CoV-2 | COVID-19 data
    import io
    from io import BytesIO

    # get data from shared Google Sheet
    response = requests.get('https://docs.google.com/spreadsheets/u/1/d/1vkvCEkZ8txrTmEldQGAycVVQbBHV-BwqTaCrxNYTtug/export?format=csv&id=1
    assert response.status_code == 200, 'Wrong status code'
    data = response.content

    # import data to dataframe
    df = pd.read_csv(BytesIO(data), usecols=['Case', 'Current Location', 'Confirmed', 'Recovered']) #unprocessed data

    # print few rows
    df.head()
```

```
      Case   Current Location          Confirmed  Recovered
   0  BN_01  Ho Chi Minh City,Vietnam          1          1
   1  BN_02  Ho Chi Minh City,Vietnam          1          1
   2  BN_03         Thanh Hoa,Vietnam          1          1
```

*1. National map*

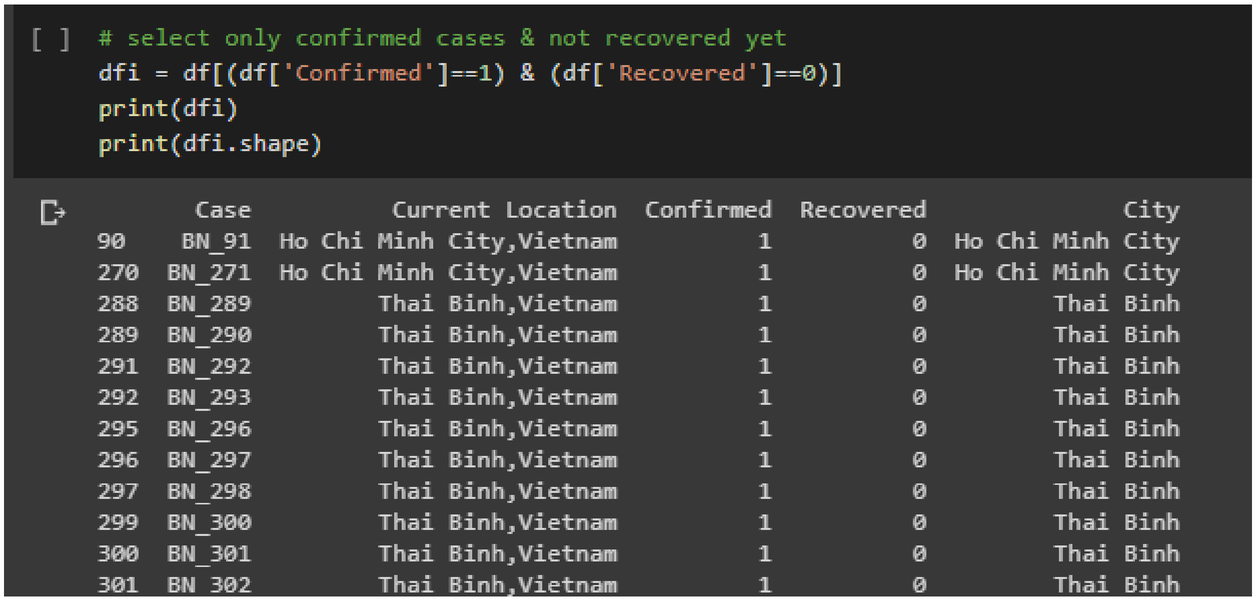
Then we must have names of cities where infected people live. We use these lines of code to get cities' names from records.



```
[ ] nrow,_ = df.shape
    for i in range(nrow):
        addr = df.iloc[i,1]
        if (str(addr) == 'nan'):
            print('index = ', i, ' addr = ', addr, ' -> no address')
        else:
            s = addr.split(',')    #delimiter = ','
            state = len(s) - 2  #get province / state
            city = s.pop(state)
            df.at[i,'City'] = city
    print(df.head())
```

```
     Case          Current Location  Confirmed  Recovered              City
0   BN_01  Ho Chi Minh City,Vietnam          1          1  Ho Chi Minh City
1   BN_02  Ho Chi Minh City,Vietnam          1          1  Ho Chi Minh City
2   BN_03         Thanh Hoa,Vietnam          1          1         Thanh Hoa
3   BN_04         Vinh Phuc,Vietnam          1          1         Vinh Phuc
4   BN_05         Vinh Phuc,Vietnam          1          1         Vinh Phuc
```

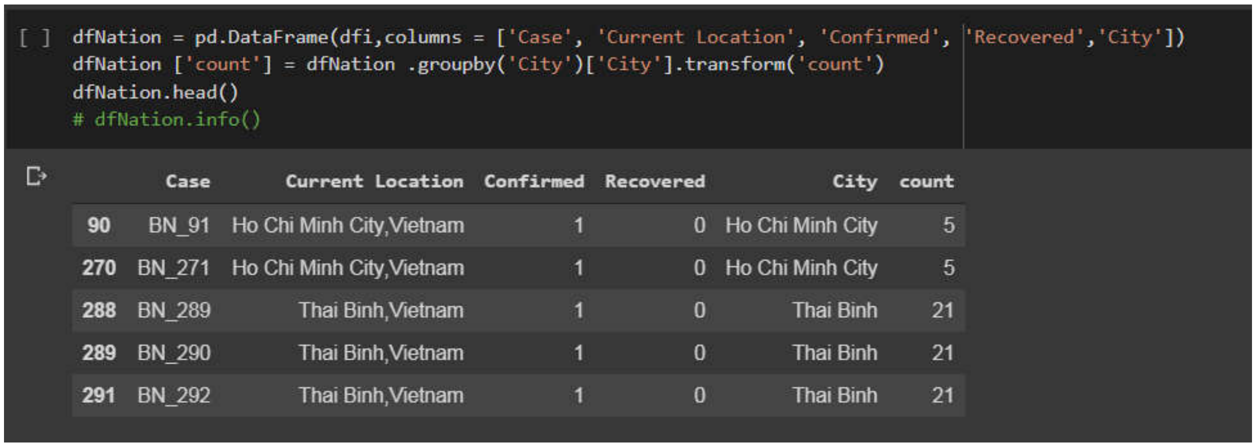
The map just shows not yet recovered patients' location, so we need to create a new dataframe called dfi, which just have not recovered cases. 30 records found.



```
[ ] # select only confirmed cases & not recovered yet
    dfi = df[(df['Confirmed']==1) & (df['Recovered']==0)]
    print(dfi)
    print(dfi.shape)
```

```
       Case          Current Location  Confirmed  Recovered              City
90    BN_91  Ho Chi Minh City,Vietnam          1          0  Ho Chi Minh City
270  BN_271  Ho Chi Minh City,Vietnam          1          0  Ho Chi Minh City
288  BN_289         Thai Binh,Vietnam          1          0         Thai Binh
289  BN_290         Thai Binh,Vietnam          1          0         Thai Binh
291  BN_292         Thai Binh,Vietnam          1          0         Thai Binh
292  BN_293         Thai Binh,Vietnam          1          0         Thai Binh
295  BN_296         Thai Binh,Vietnam          1          0         Thai Binh
296  BN_297         Thai Binh,Vietnam          1          0         Thai Binh
297  BN_298         Thai Binh,Vietnam          1          0         Thai Binh
299  BN_300         Thai Binh,Vietnam          1          0         Thai Binh
300  BN_301         Thai Binh,Vietnam          1          0         Thai Binh
301  BN_302         Thai Binh,Vietnam          1          0         Thai Binh
```

Then, in order to know the number of non-recovered people each cities have:

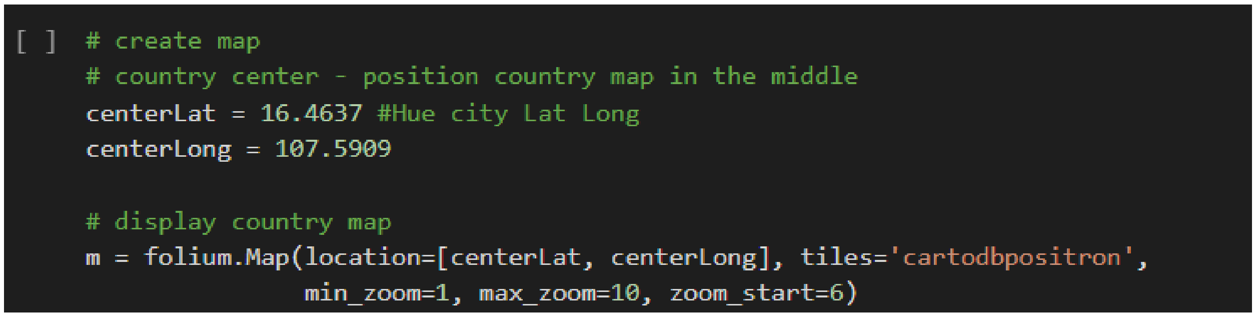


```
[ ] dfNation = pd.DataFrame(dfi,columns = ['Case', 'Current Location', 'Confirmed', 'Recovered','City'])
    dfNation ['count'] = dfNation .groupby('City')['City'].transform('count')
    dfNation.head()
    # dfNation.info()
```

|  | Case | Current Location | Confirmed | Recovered | City | count |
|---|---|---|---|---|---|---|
| 90 | BN_91 | Ho Chi Minh City,Vietnam | 1 | 0 | Ho Chi Minh City | 5 |
| 270 | BN_271 | Ho Chi Minh City,Vietnam | 1 | 0 | Ho Chi Minh City | 5 |
| 288 | BN_289 | Thai Binh,Vietnam | 1 | 0 | Thai Binh | 21 |
| 289 | BN_290 | Thai Binh,Vietnam | 1 | 0 | Thai Binh | 21 |
| 291 | BN_292 | Thai Binh,Vietnam | 1 | 0 | Thai Binh | 21 |

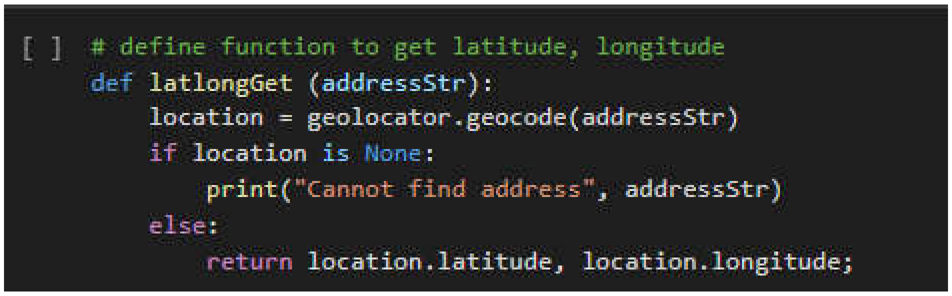
The column 'count' shows statistics of active patients in cities.

Now we have all the data already. Choose the center of the map. As we draw the Vietnam map, the center location provided for folium will be Hue's coordinate which is in the middle of Vietnam. We search for Hue's location on the Internet and produce a map of Vietnam.



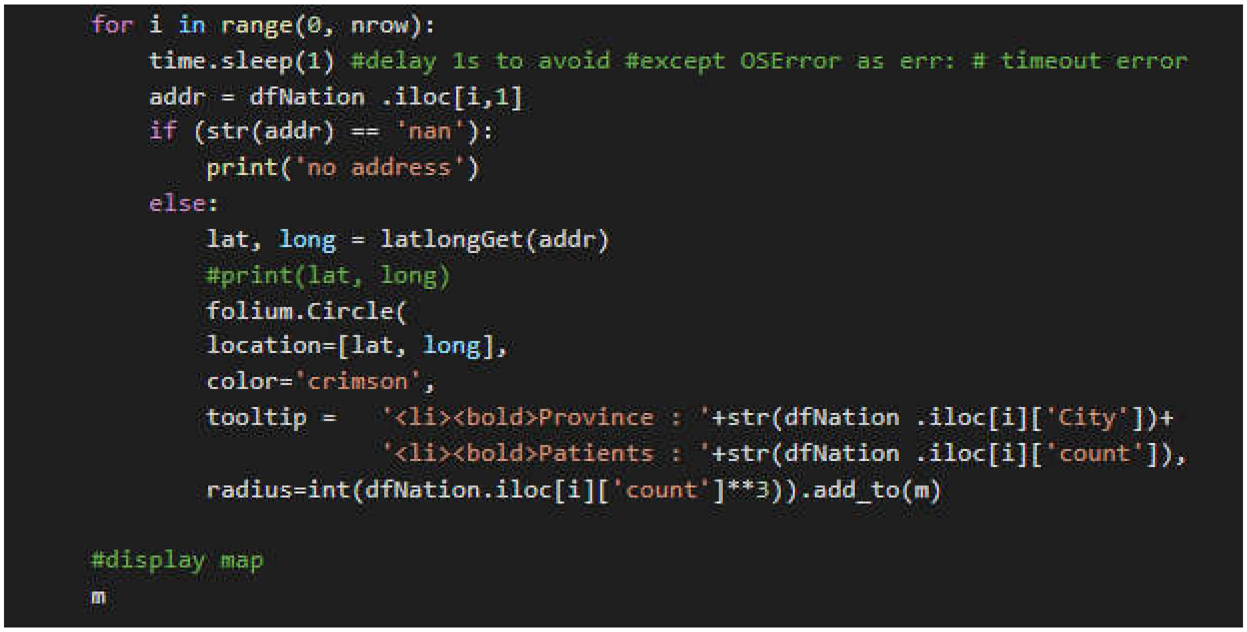
```
[ ] # create map
    # country center - position country map in the middle
    centerLat = 16.4637 #Hue city Lat Long
    centerLong = 107.5909

    # display country map
    m = folium.Map(location=[centerLat, centerLong], tiles='cartodbpositron',
                   min_zoom=1, max_zoom=10, zoom_start=6)
```

We define function latlongGet to get latitude, longitude from location's name.



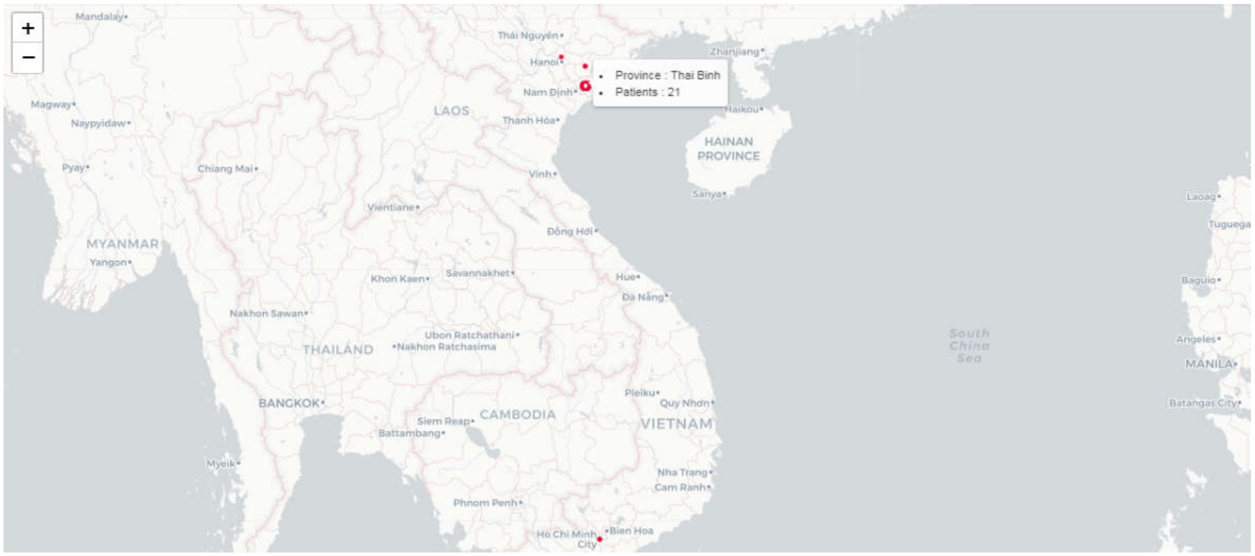
```
[ ] # define function to get latitude, longitude
    def latlongGet (addressStr):
        location = geolocator.geocode(addressStr)
        if location is None:
            print("Cannot find address", addressStr)
        else:
            return location.latitude, location.longitude;
```

Get coordination of all rows in dfNation and show them as circles by folium.Circle.



```
for i in range(0, nrow):
    time.sleep(1) #delay 1s to avoid #except OSError as err: # timeout error
    addr = dfNation .iloc[i,1]
    if (str(addr) == 'nan'):
        print('no address')
    else:
        lat, long = latlongGet(addr)
        #print(lat, long)
        folium.Circle(
        location=[lat, long],
        color='crimson',
        tooltip =   '<li><bold>Province : '+str(dfNation .iloc[i]['City'])+
                    '<li><bold>Patients : '+str(dfNation .iloc[i]['count']),
        radius=int(dfNation.iloc[i]['count']**3)).add_to(m)

#display map
m
```

Our result:



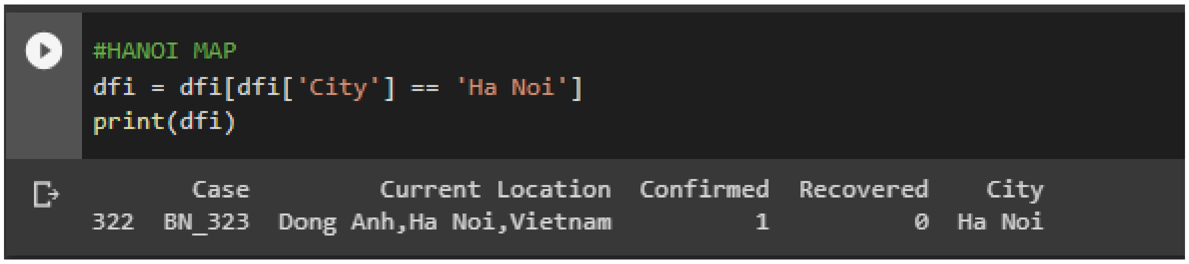
On the map, some crimson circles will be added on the coordinate of infected cities. When the mouse hovers on the circle, names of the cities and numbers of patients will appear. The size of the circle based on the patient volume of the city. The map is also equipped with zoom options.

In general, Vietnam at the time this report is written just has 4 infected cities: Ha Noi. Thai Binh, Hai Duong, TP Ho Chi Minh. Thai Binh reached the peak with 21 citizens remaining infected, 4 times higher than Ho Chi Minh with 5 patients. Ha Noi is the fewest, only 1 compared to 3 in Hai Duong. Almost all cities in the Northern with hot and humid climate characteristics. Half of all are big cities. From the map, it is clear that Thai Binh has the highest number of active patients in Viet Nam.

## 2. *Hanoi map*

With the Hanoi map, steps are a few different.

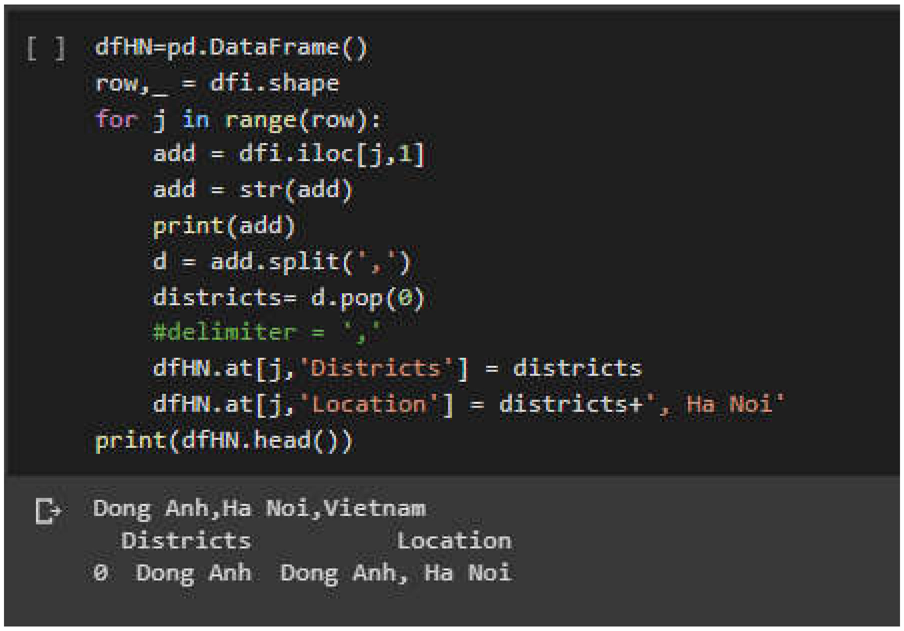
First, the dataframe dfi is modified, just records of Ha Noi patients are kept.



```
#HANOI MAP
dfi = dfi[dfi['City'] == 'Ha Noi']
print(dfi)
```

```
        Case        Current Location  Confirmed  Recovered    City
322  BN_323  Dong Anh,Ha Noi,Vietnam          1          0  Ha Noi
```

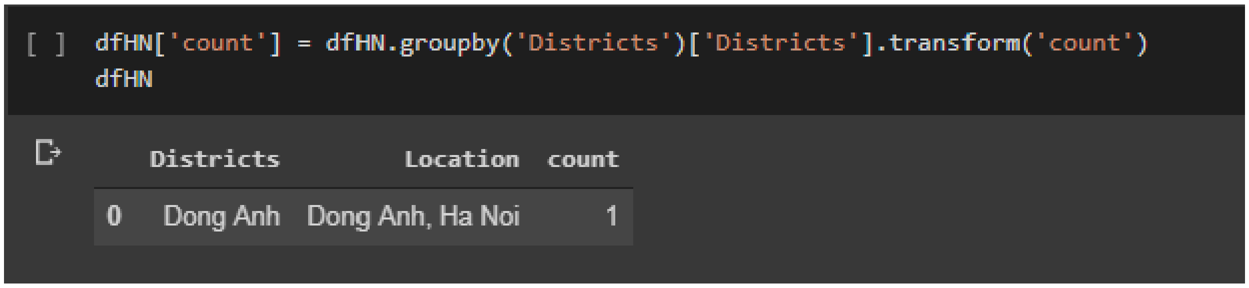
A new dataframe dfHN is created with 2 columns: Districts and Location. ‘Districts’ column is for display districts name in the map circles and ‘Location’ column ‘s record is a parameter for the calmap function to receive coordinates. ‘Location’ column is created instead of using the ‘Districts’ column since calmap is unable to find coordinates of a district without the city’s name.



```
[ ] dfHN=pd.DataFrame()
    row,_ = dfi.shape
    for j in range(row):
        add = dfi.iloc[j,1]
        add = str(add)
        print(add)
        d = add.split(',')
        districts= d.pop(0)
        #delimiter = ','
        dfHN.at[j,'Districts'] = districts
        dfHN.at[j,'Location'] = districts+', Ha Noi'
    print(dfHN.head())
```

```
Dong Anh,Ha Noi,Vietnam
  Districts          Location
0  Dong Anh  Dong Anh, Ha Noi
```

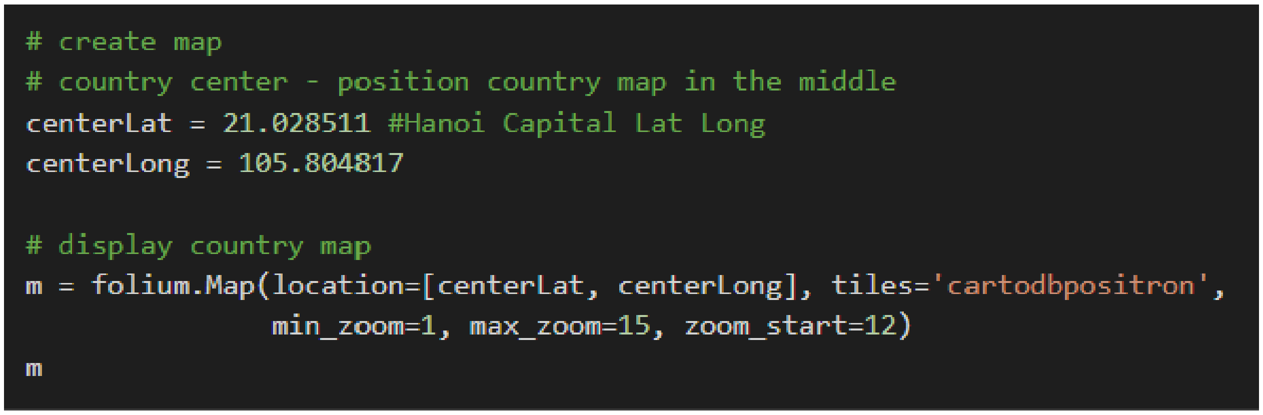
Thirdly, we calculate times each district appears in the data frame dfHN.



```
[ ]  dfHN['count'] = dfHN.groupby('Districts')['Districts'].transform('count')
     dfHN
```

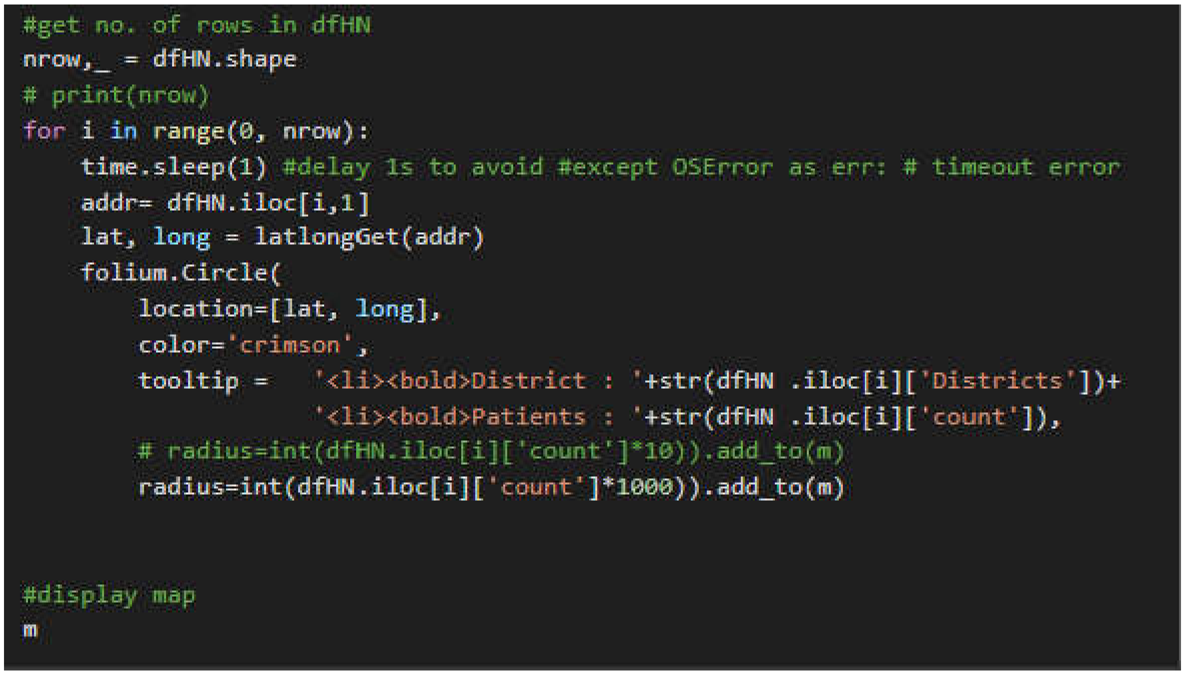
```
       Districts        Location   count
   0   Dong Anh   Dong Anh, Ha Noi     1
```

Center location provided for folium is the center location of Hanoi.



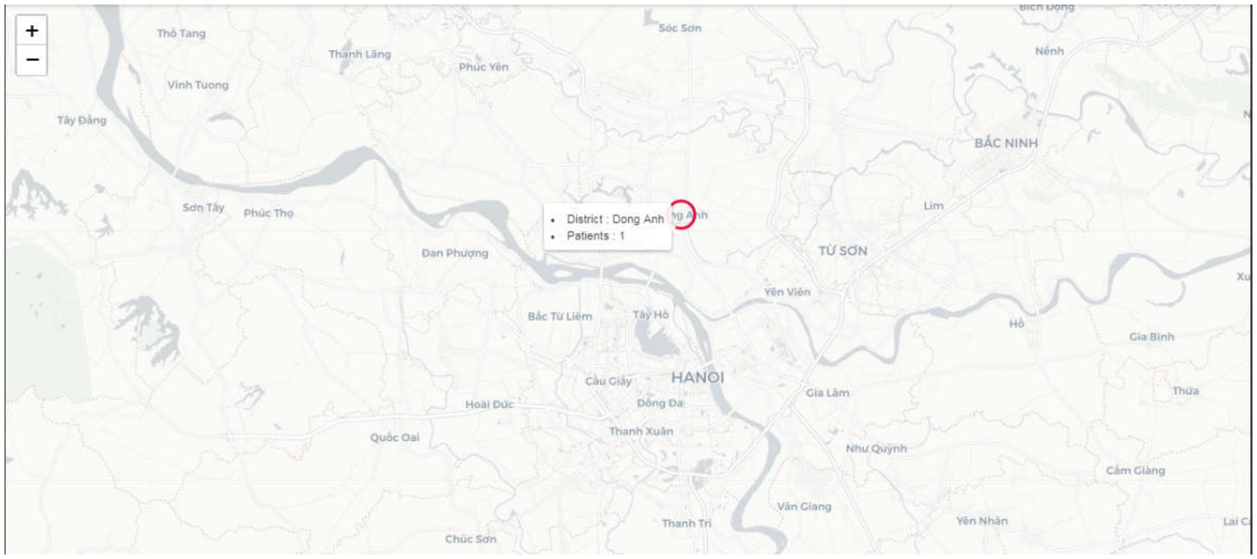
```
# create map
# country center - position country map in the middle
centerLat = 21.028511 #Hanoi Capital Lat Long
centerLong = 105.804817

# display country map
m = folium.Map(location=[centerLat, centerLong], tiles='cartodbpositron',
               min_zoom=1, max_zoom=15, zoom_start=12)
m
```

Finally, map circles are drawn.



```
#get no. of rows in dfHN
nrow,_ = dfHN.shape
# print(nrow)
for i in range(0, nrow):
    time.sleep(1) #delay 1s to avoid #except OSError as err: # timeout error
    addr= dfHN.iloc[i,1]
    lat, long = latlongGet(addr)
    folium.Circle(
        location=[lat, long],
        color='crimson',
        tooltip =   '<li><bold>District : '+str(dfHN .iloc[i]['Districts'])+
                    '<li><bold>Patients : '+str(dfHN .iloc[i]['count']),
        # radius=int(dfHN.iloc[i]['count']*10)).add_to(m)
        radius=int(dfHN.iloc[i]['count']*1000)).add_to(m)


#display map
m
```

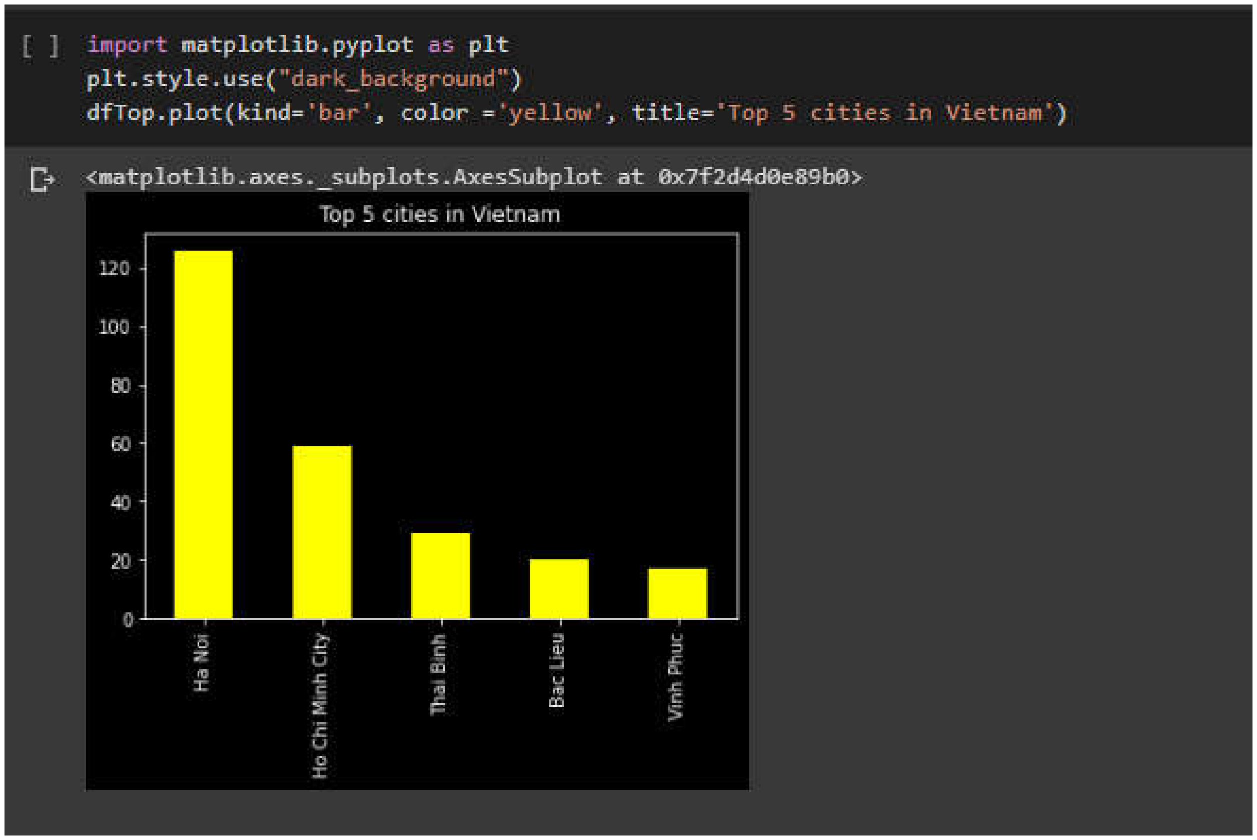
The product:



From the map, we can see that just Dong Anh still has a patient. In other districts, all cases are recovered.

*B. Top 5 highest number of patients cities and districts in Hanoi bar chart*

The first bar chart shows information about the top five highest number of patient cities in Vietnam, which is Ha Noi, Ho Chi Minh, Thai Binh, Bac Lieu, Vinh Phuc.

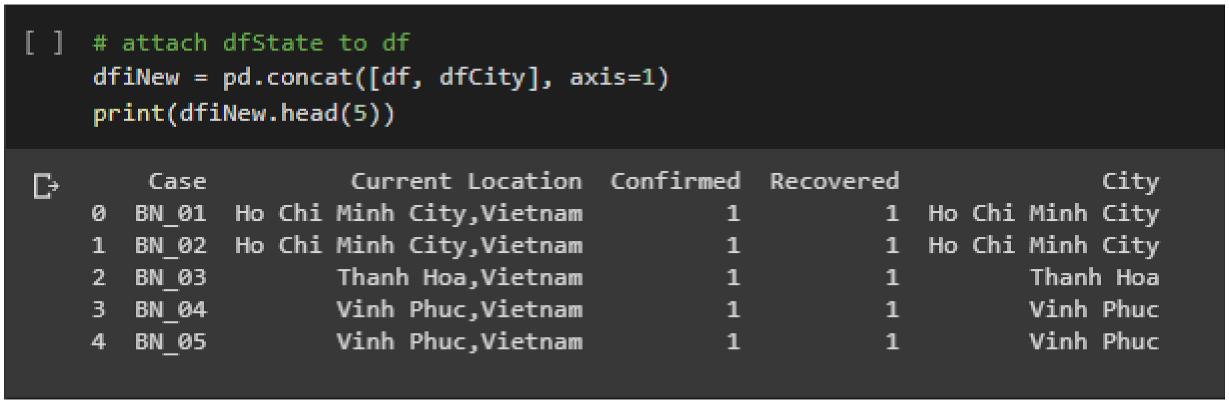


```
[ ]  import matplotlib.pyplot as plt
     plt.style.use("dark_background")
     dfTop.plot(kind='bar', color ='yellow', title='Top 5 cities in Vietnam')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2d4d0e89b0>
```

It is clear that the largest number of people confirmed in the country went to Hanoi with 126 diagnosed coronavirus cases. The figure is more than double that the confirmed cases own Ho Chi Minh (approximately 60 cases). In the following three cities, there is a significant decrease in the number of people testing positive, which is 29, 20 and 17 cases in Thai Binh, Bac Lieu and Vinh phuc.

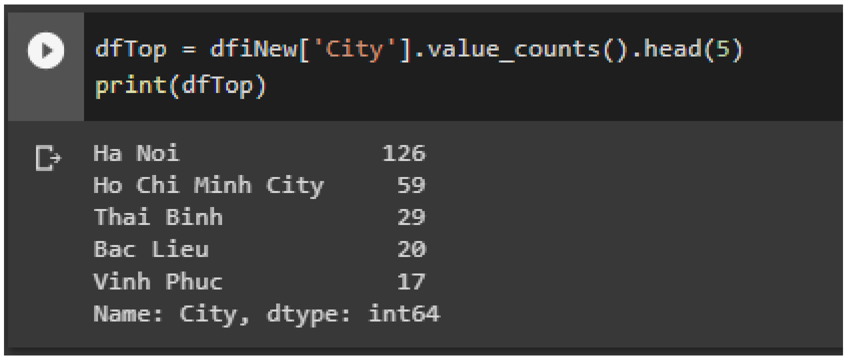
To draw this bar chart, we get data from our shared Google Sheet by using Requests and Pandas and need data processing to add a city column, alike Map.



```
[ ] # attach dfState to df
    dfiNew = pd.concat([df, dfCity], axis=1)
    print(dfiNew.head(5))
```

```
     Case          Current Location  Confirmed  Recovered              City
0   BN_01  Ho Chi Minh City,Vietnam          1          1  Ho Chi Minh City
1   BN_02  Ho Chi Minh City,Vietnam          1          1  Ho Chi Minh City
2   BN_03         Thanh Hoa,Vietnam          1          1         Thanh Hoa
3   BN_04         Vinh Phuc,Vietnam          1          1         Vinh Phuc
4   BN_05         Vinh Phuc,Vietnam          1          1         Vinh Phuc
```

The next step filters top 5 cities with the highest confirmed patients and the number of people positive testing.

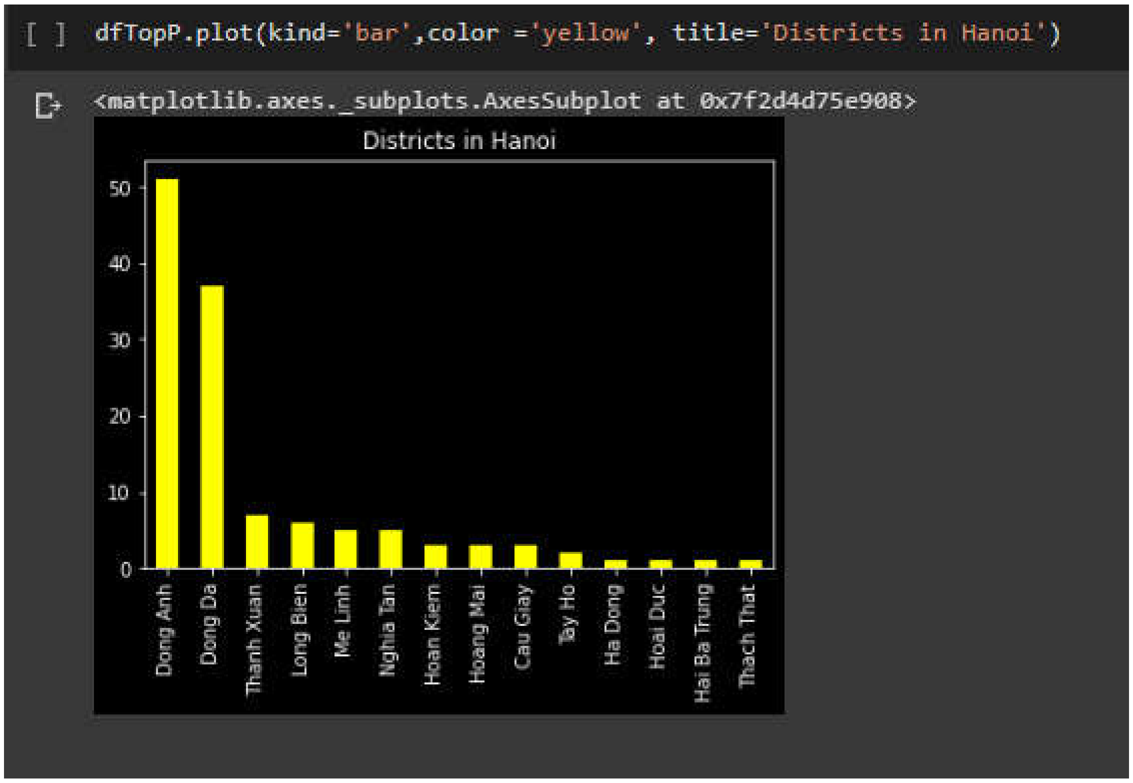


```
dfTop = dfiNew['City'].value_counts().head(5)
print(dfTop)
```

```
Ha Noi              126
Ho Chi Minh City     59
Thai Binh            29
Bac Lieu             20
Vinh Phuc            17
Name: City, dtype: int64
```

Finally, we use **Matplotlib** to draw a bar chart about the top five highest number of patient cities in Vietnam.

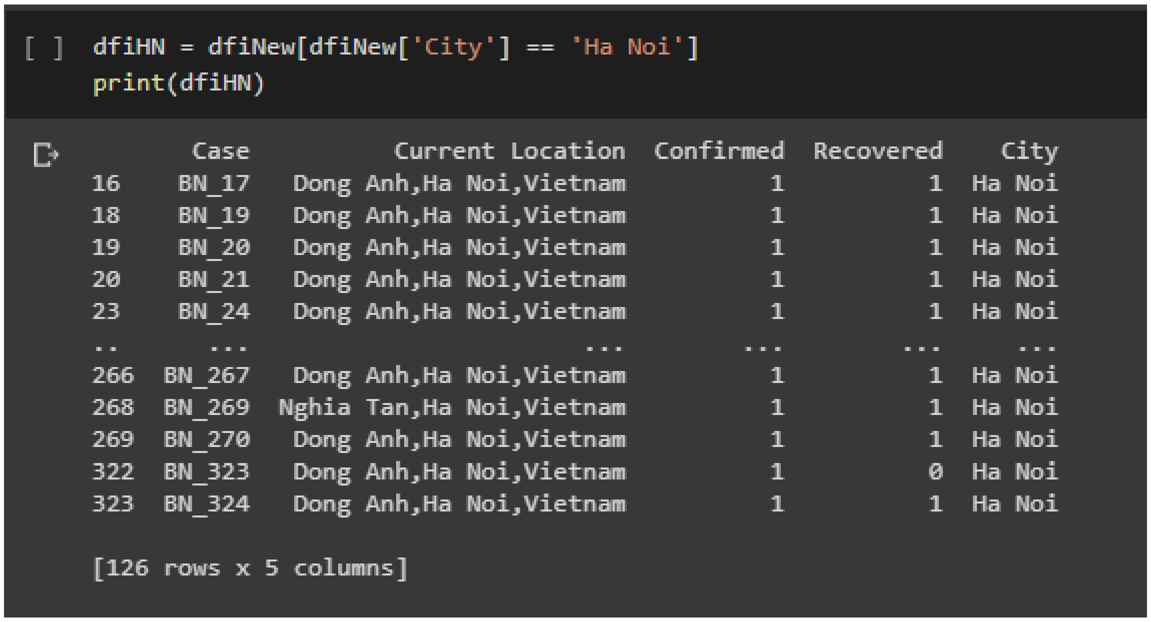
The second bar chart shows information about the number of patients in Hanoi city.



```
[ ]  dfTopP.plot(kind='bar',color ='yellow', title='Districts in Hanoi')

     <matplotlib.axes._subplots.AxesSubplot at 0x7f2d4d75e908>
```

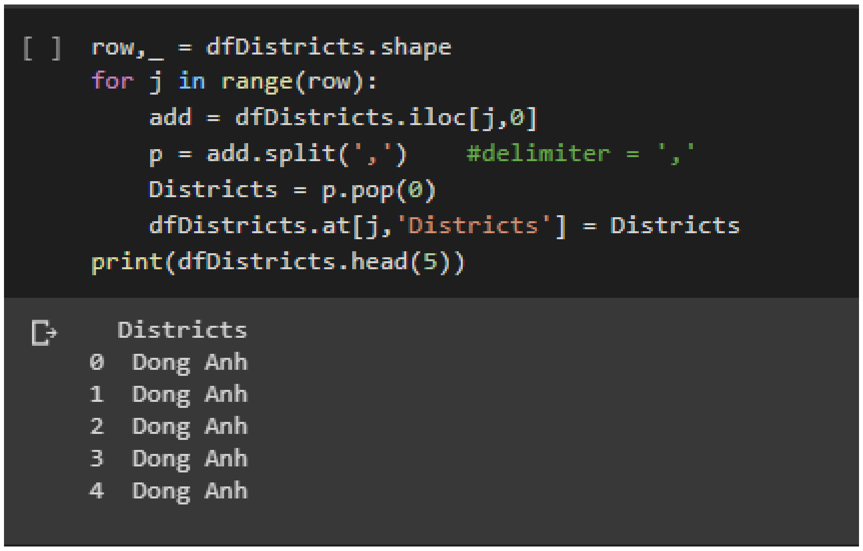
Out of all provinces in Hanoi, patient confirmation in Dong Anh and Dong Da is noticeably higher than others, at 51 cases in Dong Anh , and Dong Da at nearly 40 cases. The figure of people testing positive is the lowest in Hoai Duc, Thach That, Hai Ba Trung and Ha Dong in just one case.

Based on the before data processing, we filter only 'Ha Noi' rows in the city column.



```
[ ]  dfiHN = dfiNew[dfiNew['City'] == 'Ha Noi']
     print(dfiHN)

            Case          Current Location  Confirmed  Recovered    City
     16    BN_17    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi
     18    BN_19    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi
     19    BN_20    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi
     20    BN_21    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi
     23    BN_24    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi
     ..      ...                        ...        ...        ...     ...
     266  BN_267    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi
     268  BN_269  Nghia Tan,Ha Noi,Vietnam          1          1  Ha Noi
     269  BN_270    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi
     322  BN_323    Dong Anh,Ha Noi,Vietnam          1          0  Ha Noi
     323  BN_324    Dong Anh,Ha Noi,Vietnam          1          1  Ha Noi

     [126 rows x 5 columns]
```

Then, there is the separation of districts values in Hanoi from the current location column and save it in a new table.



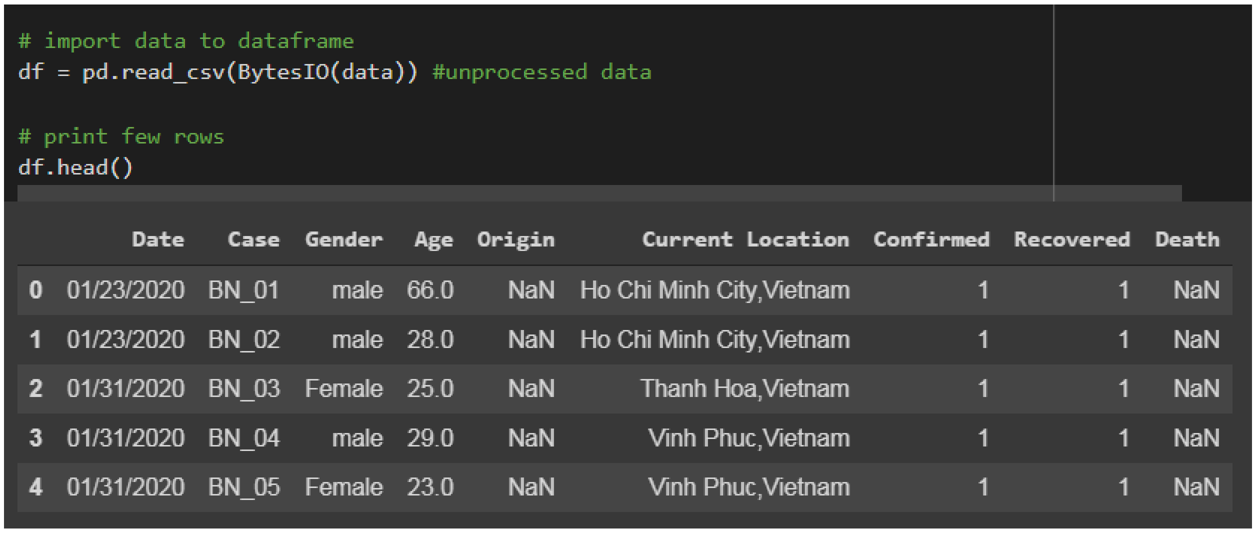
```
[ ] row,_ = dfDistricts.shape
    for j in range(row):
        add = dfDistricts.iloc[j,0]
        p = add.split(',')    #delimiter = ','
        Districts = p.pop(0)
        dfDistricts.at[j,'Districts'] = Districts
    print(dfDistricts.head(5))
```

```
  Districts
0  Dong Anh
1  Dong Anh
2  Dong Anh
3  Dong Anh
4  Dong Anh
```

The final step counts the number of confirmed cases of each province and draws the bar chart by matplotlib.

*C. New cases per day bar chart*

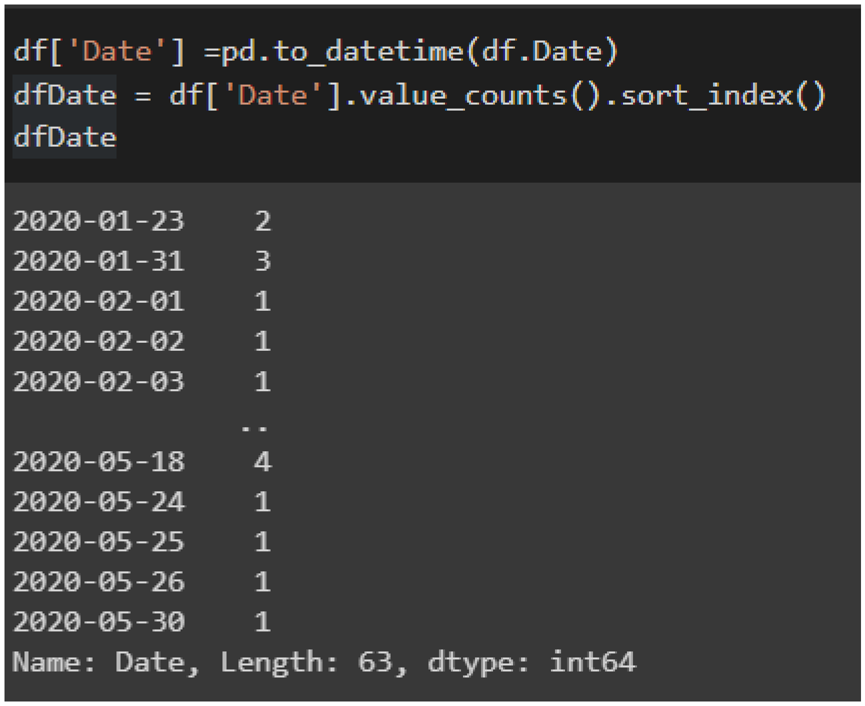
First, we need to import some libraries, install packages and import data to data frame.



```
# import data to dataframe
df = pd.read_csv(BytesIO(data)) #unprocessed data

# print few rows
df.head()
```

| | Date | Case | Gender | Age | Origin | Current Location | Confirmed | Recovered | Death |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 01/23/2020 | BN_01 | male | 66.0 | NaN | Ho Chi Minh City,Vietnam | 1 | 1 | NaN |
| 1 | 01/23/2020 | BN_02 | male | 28.0 | NaN | Ho Chi Minh City,Vietnam | 1 | 1 | NaN |
| 2 | 01/31/2020 | BN_03 | Female | 25.0 | NaN | Thanh Hoa,Vietnam | 1 | 1 | NaN |
| 3 | 01/31/2020 | BN_04 | male | 29.0 | NaN | Vinh Phuc,Vietnam | 1 | 1 | NaN |
| 4 | 01/31/2020 | BN_05 | Female | 23.0 | NaN | Vinh Phuc,Vietnam | 1 | 1 | NaN |

We sort data by date and count number case per day before visualizing:



```
df['Date'] =pd.to_datetime(df.Date)
dfDate = df['Date'].value_counts().sort_index()
dfDate
```

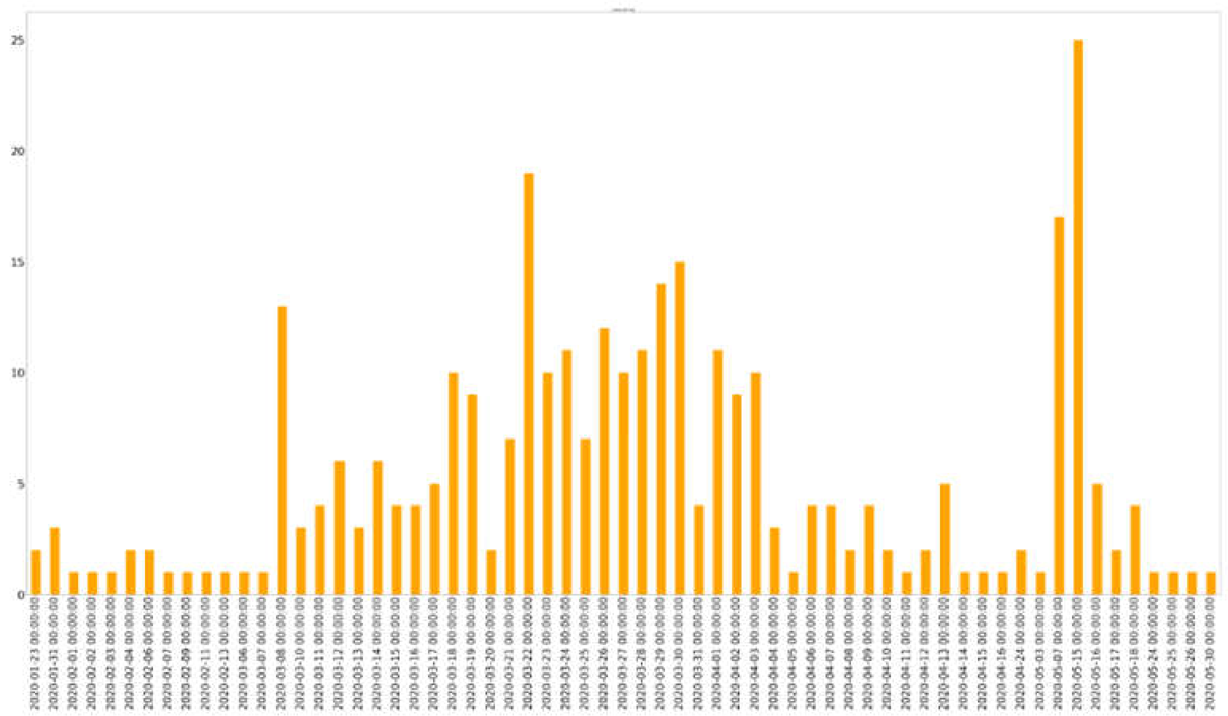
```
2020-01-23    2
2020-01-31    3
2020-02-01    1
2020-02-02    1
2020-02-03    1
             ..
2020-05-18    4
2020-05-24    1
2020-05-25    1
2020-05-26    1
2020-05-30    1
Name: Date, Length: 63, dtype: int64
```

We use the function to_datetime of pandas to convert values of the 'Date' column to the format Year-Month-Day. This step is compulsory for sorting dates.

After that we count times each date appear in dfDate by the function value_counts(), the result is the number of patients on that day. Sort_index() function will sort date ascendingly.

So, we got data sorted and the number of infected each day in an array. But It's hard to be seen and compared in table or row view.

Then, using sorted data to create bar chart:



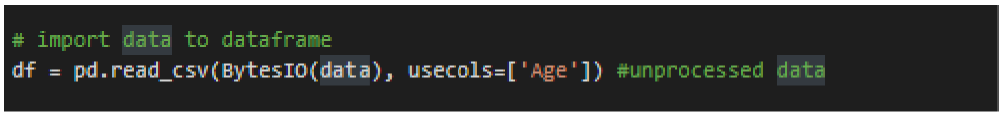
Clearly seen that although the highest people infected in the fifth of May but most cases are in range from middle to end of March. There were fluctuations hardly from Jan to middle of March. Then, numbers rise lightly day by day until the end of March. It falls down equally when disease starts.

On 22 March, the number was rocket changed but still lower than the peak number in the middle of May. 07 May is 3rd highest number but still much higher than nearby day. 30/03 is 4th but it's in a rising chain when disease starts so it isn't a surprise.

Overall after the first wave, number cases per day tend to decrease and remain number monthly, although sometimes they rise suddenly.

*D. Age clusters pie chart*

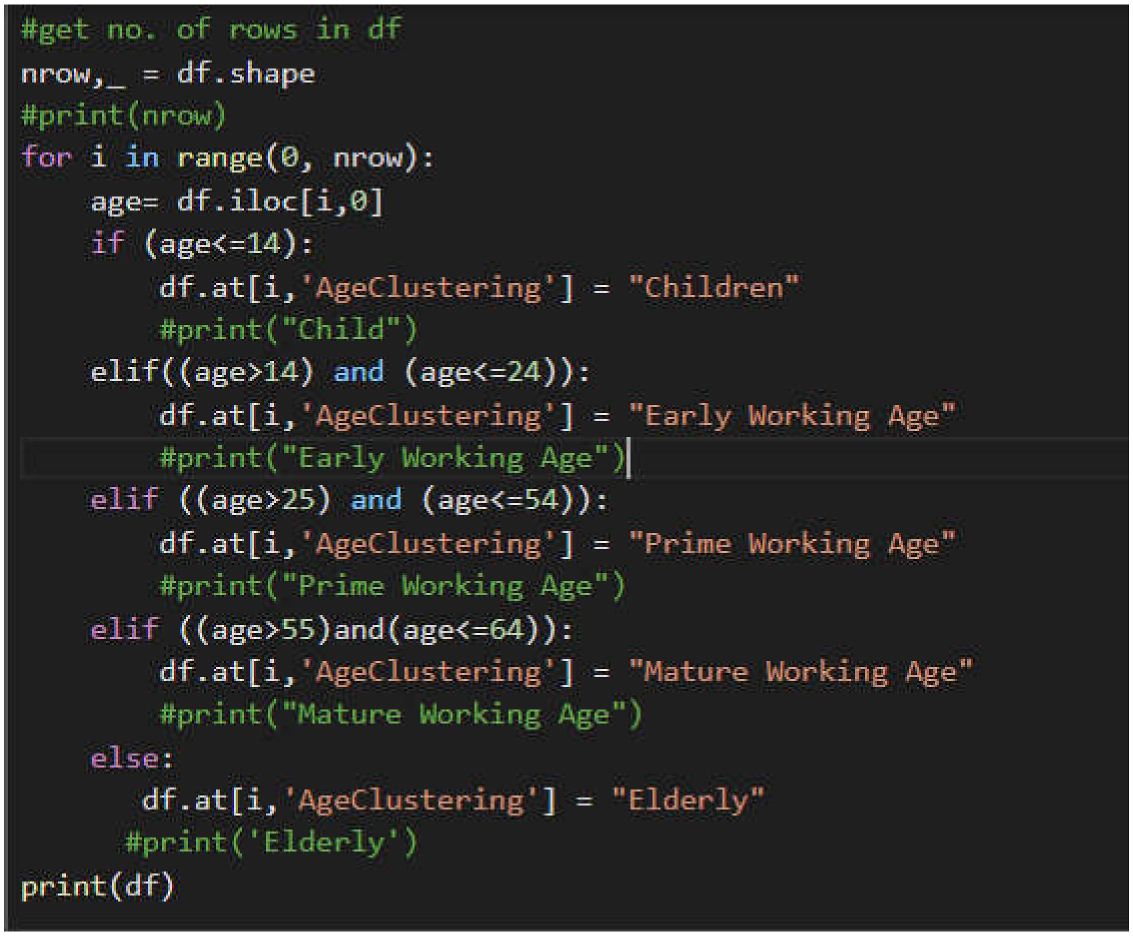
- The first is to get the data from the datasets using API from Google Sheets, we need to import Main dataset which we can load in using pandas read_csv method. We will visualize the age clusters so there's only one column we use is 'Age'.



```
# import data to dataframe
df = pd.read_csv(BytesIO(data), usecols=['Age']) #unprocessed data
```

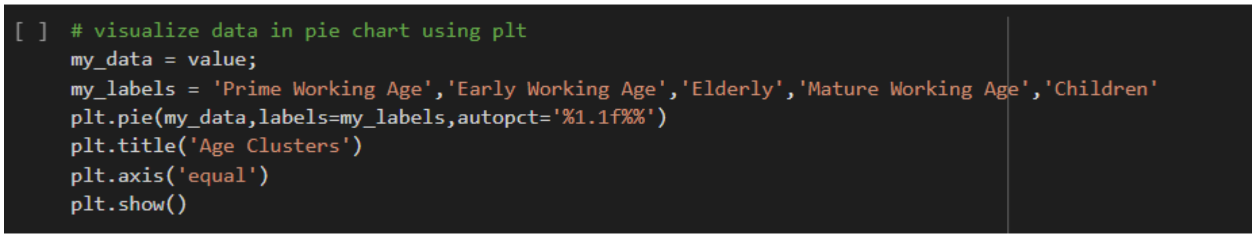
We have divided the age into five clusters which have:

- The age group less than or equal 14 is Children.
- The age group more than 14 and less than or equal 24 is the Early Working Age.
- The age group more than 25 and less than or equal 54 is Prime Working Age.
- The age group more than 54 and less than or equal 64 is the Mature Working Age.
- The last is the Elderly.



```
#get no. of rows in df
nrow,_ = df.shape
#print(nrow)
for i in range(0, nrow):
    age= df.iloc[i,0]
    if (age<=14):
        df.at[i,'AgeClustering'] = "Children"
        #print("Child")
    elif((age>14) and (age<=24)):
        df.at[i,'AgeClustering'] = "Early Working Age"
        #print("Early Working Age")
    elif ((age>25) and (age<=54)):
        df.at[i,'AgeClustering'] = "Prime Working Age"
        #print("Prime Working Age")
    elif ((age>55)and(age<=64)):
        df.at[i,'AgeClustering'] = "Mature Working Age"
        #print("Mature Working Age")
    else:
       df.at[i,'AgeClustering'] = "Elderly"
      #print('Elderly')
print(df)
```

- The result will be printed to show probably the number of confirmed cases in each age cluster. Use value_counts() to the total value of each age group then change that value into an array to count the cluster percentage. And finally, we display the pie chart with matplotlib.

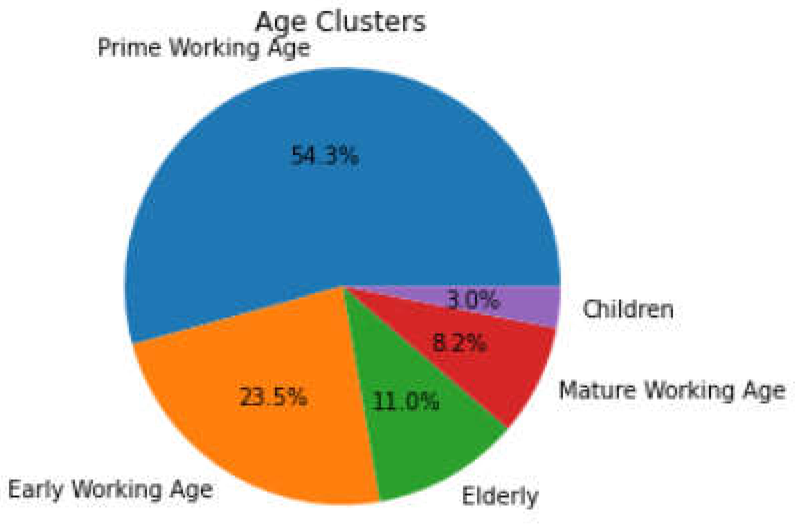


```
[ ] # visualize data in pie chart using plt
    my_data = value;
    my_labels = 'Prime Working Age','Early Working Age','Elderly','Mature Working Age','Children'
    plt.pie(my_data,labels=my_labels,autopct='%1.1f%%')
    plt.title('Age Clusters')
    plt.axis('equal')
    plt.show()
```

The pie chart illustrates the proportion of five groups of age that is the confirmed case in Vietnam.

Overall, in our country, the most significant age clusters were Prime Working Age, which accounts for more than half the age group, while Elderly, Mature Working Age, Children display the least confirmed cases. Lastly, the Early Working Age, which represents 23.5% separately.

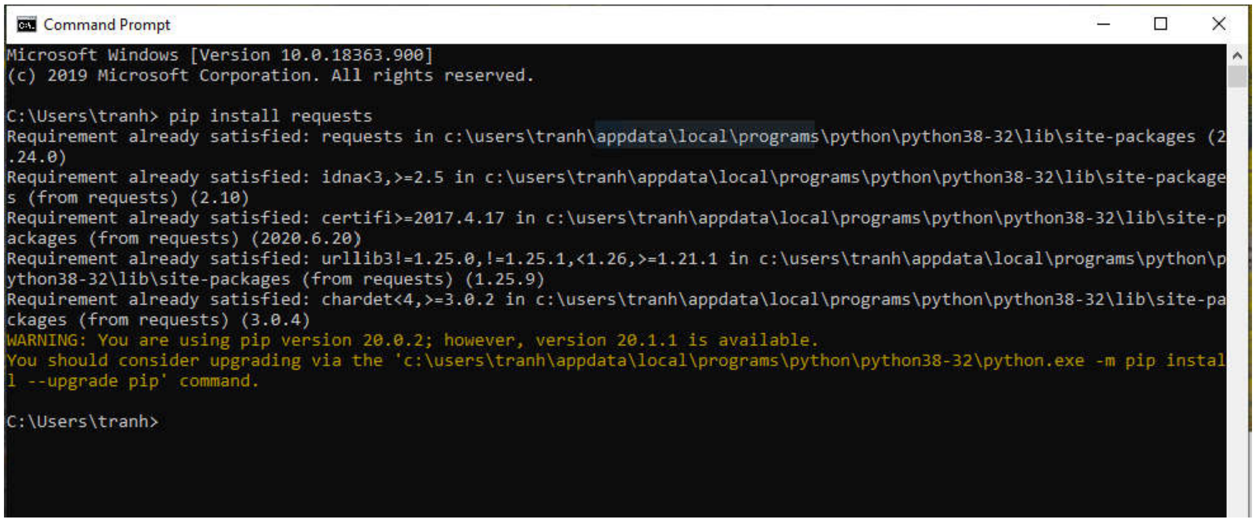
Eventually, people in the age of 25 to 54 have the most confirmed cases. On the other hand, the number of children is just a few people.



*E. Animation: line chart about cases over the time and news cases.*

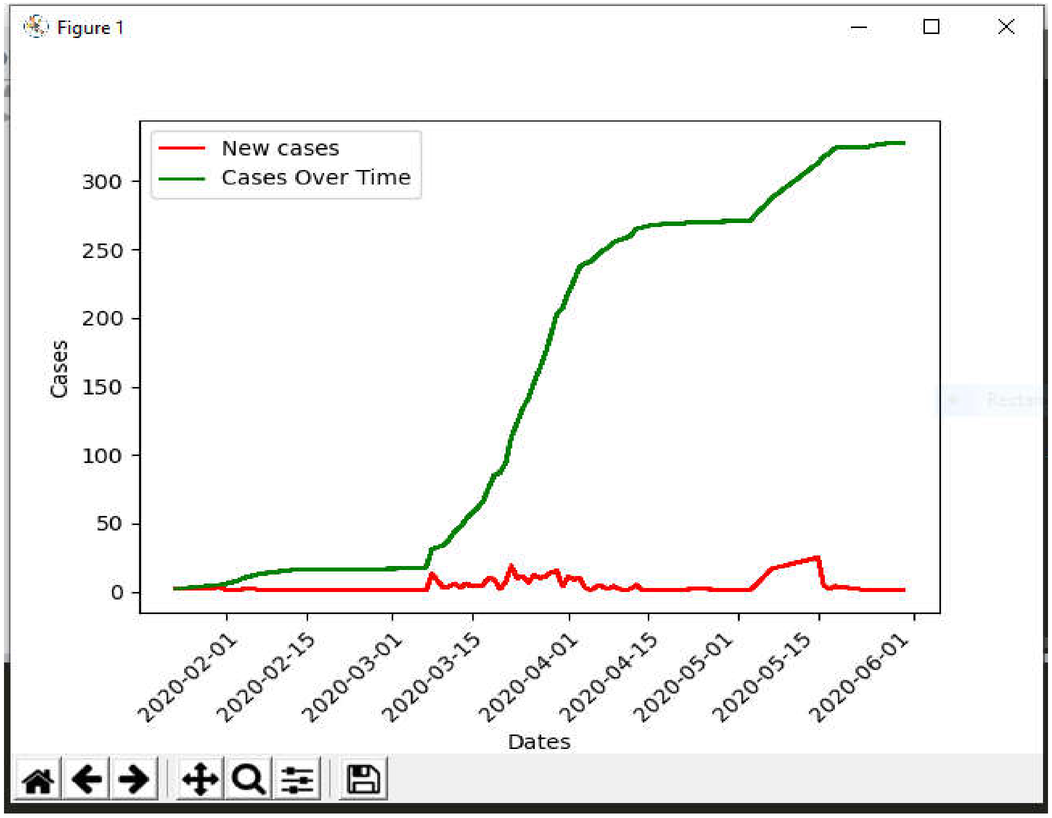
We must use Sublime text to show animation because google colab does not support. We will introduce how to install environments for PCs and annelize our code.

Firstly, you need an open Command prompt in your PC and install numpy, matplotlib and requests libraries with command : 'pip install <*name library*>'.



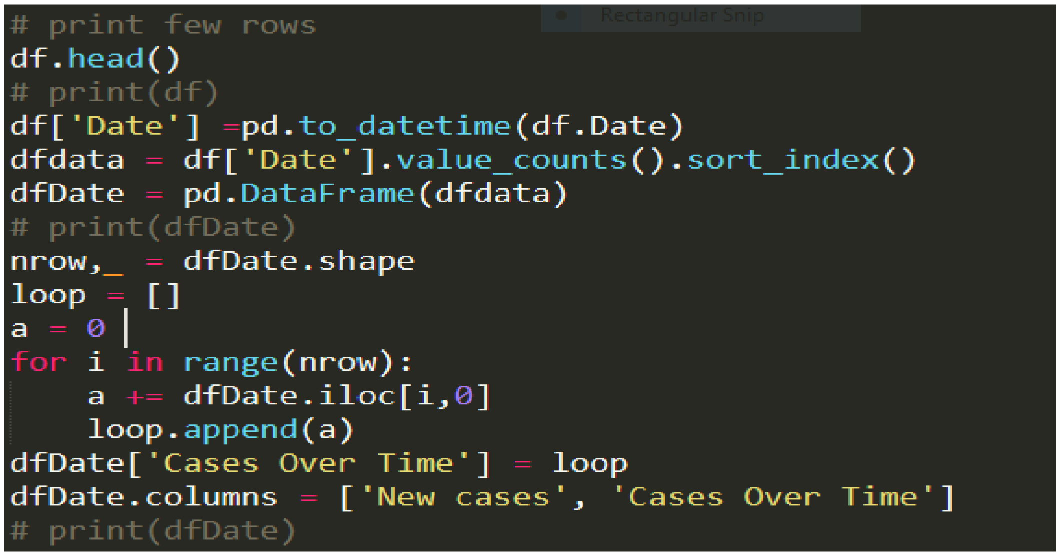
```
Microsoft Windows [Version 10.0.18363.900]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\tranh> pip install requests
Requirement already satisfied: requests in c:\users\tranh\appdata\local\programs\python\python38-32\lib\site-packages (2
.24.0)
Requirement already satisfied: idna<3,>=2.5 in c:\users\tranh\appdata\local\programs\python\python38-32\lib\site-package
s (from requests) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\tranh\appdata\local\programs\python\python38-32\lib\site-p
ackages (from requests) (2020.6.20)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\users\tranh\appdata\local\programs\python\p
ython38-32\lib\site-packages (from requests) (1.25.9)
Requirement already satisfied: chardet<4,>=3.0.2 in c:\users\tranh\appdata\local\programs\python\python38-32\lib\site-pa
ckages (from requests) (3.0.4)
WARNING: You are using pip version 20.0.2; however, version 20.1.1 is available.
You should consider upgrading via the 'c:\users\tranh\appdata\local\programs\python\python38-32\python.exe -m pip instal
l --upgrade pip' command.

C:\Users\tranh>
```

Then, you open file ‘DateAnimation.py’ by sublime text and run (Ctrl + B).



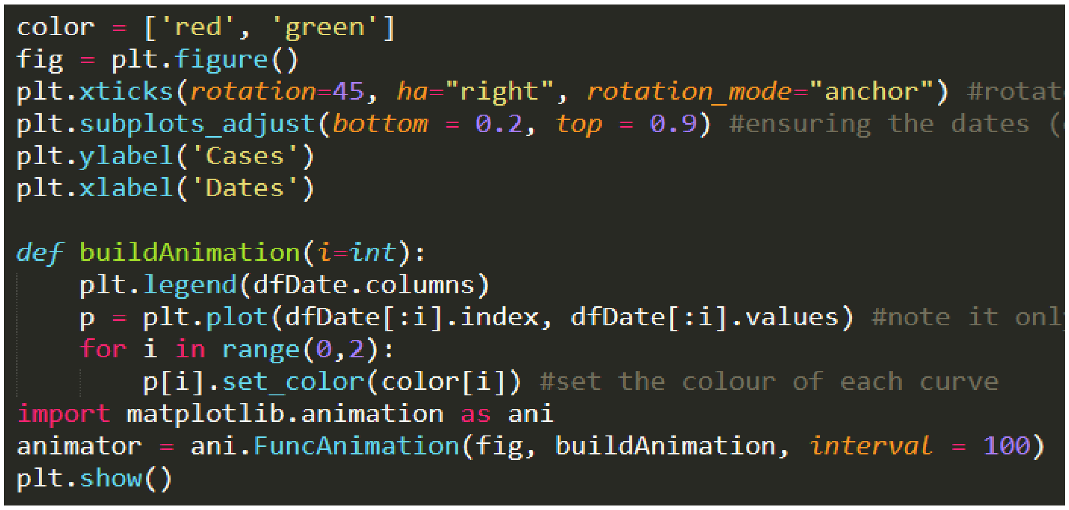
The line chart showed two functions with green and red lines. It is clear that the number of positive testing cases increased significantly from 03.2020 to 04.2020 approach to approximately 250 cases. In the following time, the figure was controlled and no more new cases. On 30.06.2020, positive testing cases were 329 cases in Vietnam and no more new cases.

To show animation, we get data from our shared Google Sheet by using Requests and Pandas and need data processing to add ‘new cases’ and ‘case over time ‘ columns.



```
# print few rows
df.head()
# print(df)
df['Date'] =pd.to_datetime(df.Date)
dfdata = df['Date'].value_counts().sort_index()
dfDate = pd.DataFrame(dfdata)
# print(dfDate)
nrow,_ = dfDate.shape
loop = []
a = 0 |
for i in range(nrow):
    a += dfDate.iloc[i,0]
    loop.append(a)
dfDate['Cases Over Time'] = loop
dfDate.columns = ['New cases', 'Cases Over Time']
# print(dfDate)
```

Finally, we set up a chart and animation.



```
color = ['red', 'green']
fig = plt.figure()
plt.xticks(rotation=45, ha="right", rotation_mode="anchor") #rotat
plt.subplots_adjust(bottom = 0.2, top = 0.9) #ensuring the dates (
plt.ylabel('Cases')
plt.xlabel('Dates')

def buildAnimation(i=int):
    plt.legend(dfDate.columns)
    p = plt.plot(dfDate[:i].index, dfDate[:i].values) #note it onl
    for i in range(0,2):
        p[i].set_color(color[i]) #set the colour of each curve
import matplotlib.animation as ani
animator = ani.FuncAnimation(fig, buildAnimation, interval = 100)
plt.show()
```

## IV. CONCLUSION AND FUTURE WORK

In this report, we addressed a contemporary visualization for COVID-19 disease status in Vietnam using map representation and chart. One of the important contributions of our work is to express the dataset to visualise representation  by pie chart, bar chart and map about the detail situation of Covid-19 disease based on python. We presented data sets, algorithms & libraries and implemented them in the below parts. It is important that the main focus of our report is Implement to explain our code. Finally, we believe that our project contributed a vital part in the awareness of Vietnamese about the detailed disease status from February to June 2020 in Vietnam.

From websites about Covid-19 disease, our team will update our dataset and provide more and more visualization in different ways.

## V.REFERENCES

[1][2]Phuong, L. (2020). *VnExpress: Coronavirus in Vietnam: Two people from Wuhan pneumonia were isolated at Cho Ray Hospital.* Retrieved from VnExpress: https://vnexpress.net/dich-viem-phoi-corona/hai-nguoi-viem-phoi-vu-han-cach-ly-tai-benh-vien-cho-ray-4046299.html

[3]Vinmec. (2020). *Vinmec: Corona News on 13/2: Isolate Vinh Phuc epidemic area, more schools are absent.* Retrieved from Vinmec: https://www.vinmec.com/vi/tin-tuc/thong-tin-suc-khoe/dich-2019-ncov/thong-tin-suc-khoe/ban-tin-corona-132-co-lap-vung-dich-vinh-phuc-them-nhieu-truong-nghi-hoc/

[4]Viet, T. (2020). *World & VN.* Retrieved from TGVN: https://baoquocte.vn/viet-nam-ghi-nhan-them-5-ca-mac-covid-19-tai-ha-noi-quang-ninh-va-tp-ho-chi-minh-111704.html

[5]MOH. (2020). *Ministry of Health: Timeline.* Retrieved from MOH: https://ncov.moh.gov.vn/web/guest/dong-thoi-gian?1