IBM-Capstone-Project-The-Battle-of-neighbourhoods

Coronavirus Covid-19 - A Study

Introduction

Background and Problem Statement

2019 Novel Coronavirus (2019-nCoV) or Covid-19 is a virus (more specifically, a coronavirus) identified as the cause of an outbreak of respiratory illness first detected in Wuhan, China. Early on, many of the patients in the outbreak in Wuhan, China reportedly had some link to a large seafood and animal market, suggesting animal-to-person spread. However, a growing number of patients reportedly have not had exposure to animal markets, indicating person-to-person spread is occurring.

Individual risk for the disease is dependent on exposure. Covid-19 has now been detected in almost 150 locations internationally, including in the United States.There have been close to 1,70,000 people sickened by COVID-19 and more than 7,000 people have died from the disease—a death toll that has far surpassed that of the severe acute respiratory syndrome (SARS) epidemic that occurred in 2002 and 2003. Officials everywhere have implemented measures to contain the virus, including travel restrictions and quarantines. Based on the circumstances, WHO (World Health Organization) has declared COVID-19 a pandemic (an epidemic that spreads throughout the world).

This study will highlight the different regions affected globally, the number of people affected, deaths, people who have recovered and the common symptoms.The different visualisations will help us understand the spread of the disease, the estimated mortality rate and other statistics. This study aims to create awareness among the general public, which is of extreme importance in this situation and provides several useful insights into the data. This can also be used by researchers, students who are interested in the subject.

Data

The data used for this study has been taken from the links mentioned below. Furthermore, we will access data through FourSquare API interface and arrange them as a dataframe for visualization. One of the datasets consists of data from 22nd Jan, 2020 while the other consists of daily case reports of the number of confirmed, dead and recovered cases. The usage of data is explained in detail in the implementation section.

Links :- https://github.com/CSSEGISandData/COVID-19
https://www.kaggle.com/sudalairajkumar/novel-corona-virus-2019-dataset

Methodology

This section will describe the main components of our Study. The steps are as follows:-

1. Collect covid-19 related data from https://github.com/CSSEGISandData/COVID-19 and https://www.kaggle.com/sudalairajkumar/novel-corona-virus-2019-dataset
2. Explore, understand the data, data preprocessing
3. Using FourSquare API we will find all location related information
4. Visualize and Analyse using folium library(python) and plotly

Let's get started !

We'll import all the necessary libraries and collect the data used for analysis.

In [1]:

```
!pip install plotly
!pip install folium
!pip install geopy
from geopy.geocoders import Nominatim # convert an address into latitude and longitude values
import folium # plotting library
!python -m pip install --upgrade pip
import requests # library to handle requests
import pandas as pd # library for data analsysis
import numpy as np # library to handle data in a vectorized manner
import plotly as py #library for data visualization
import plotly.express as px
import plotly.graph_objs as go
```

```
#Data as on 15-03-2020
df1 = pd.read_csv("03-15-2020.csv")
df1.head()
```

```
Requirement already satisfied: plotly in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (4.5.4)
Requirement already satisfied: retrying>=1.3.3 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from plotly) (1.3.3)
Requirement already satisfied: six in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from plotly) (1.11.0)
Requirement already satisfied: folium in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (0.10.1)
Requirement already satisfied: jinja2>=2.9 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from folium) (2.10.1)
Requirement already satisfied: numpy in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from folium) (1.14.4)
Requirement already satisfied: requests in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from folium) (2.22.0)
Requirement already satisfied: branca>=0.3.0 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from folium) (0.3.1)
Requirement already satisfied: MarkupSafe>=0.23 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from jinja2>=2.9-
>folium) (1.1.1)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from requests->folium) (
2019.11.28)
Requirement already satisfied: idna<2.9,>=2.5 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from requests->folium) (
2.8)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from requests->folium) (
3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from requests->folium) (
1.25.7)
Requirement already satisfied: six in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from branca>=0.3.0-
>folium) (1.11.0)
Requirement already satisfied: geopy in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (1.20.0)
Requirement already satisfied: geographiclib<2,>=1.49 in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (from geopy) (1.50)
Requirement already up-to-date: pip in
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages (20.0.2)
```

Out[1]:

	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered	Latitude	Longitude
0	Hubei	China	2020-03-15T18:20:18	67794	3085	54288	30.9756	112.2707
1	NaN	Italy	2020-03-14T20:13:16	24747	1809	2335	41.8719	12.5674
2	NaN	Iran	2020-03-15T18:20:18	13938	724	4590	32.4279	53.6880
3	NaN	Korea, South	2020-03-15T18:20:18	8162	75	510	35.9078	127.7669
4	NaN	Spain	2020-03-15T18:20:18	7798	289	517	40.4637	-3.7492

In [2]:

```
df2 = pd.read_csv('covid_19_data.csv')
df2.rename(columns={'ObservationDate':'Date', 'Country/Region':'Country'}, inplace=True)
```

Earliest cases - Starting from 22nd January,2020

In [3]:

```
df2.head()
```

Out[3]:

	SNo	Date	Province/State	Country	Last Update	Confirmed	Deaths	Recovered

	SNo	Date	Province/State	Country	Last Update	Confirmed	Deaths	Recovered
0	1	01/22/2020	Anhui	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
1	2	01/22/2020	Beijing	Mainland China	1/22/2020 17:00	14.0	0.0	0.0
2	3	01/22/2020	Chongqing	Mainland China	1/22/2020 17:00	6.0	0.0	0.0
3	4	01/22/2020	Fujian	Mainland China	1/22/2020 17:00	1.0	0.0	0.0
4	5	01/22/2020	Gansu	Mainland China	1/22/2020 17:00	0.0	0.0	0.0

As we know, the earliest cases were mainly in China whereas the latest cases are spread around the globe.

Latest Cases - 15th March,2020

In [4]:

```
df2.tail()
```

Out[4]:

	SNo	Date	Province/State	Country	Last Update	Confirmed	Deaths	Recovered
5885	5886	03/15/2020	Gibraltar	UK	2020-03-14T16:33:03	1.0	0.0	1.0
5886	5887	03/15/2020	NaN	Uzbekistan	2020-03-15T18:20:19	1.0	0.0	0.0
5887	5888	03/15/2020	Diamond Princess cruise ship	Australia	2020-03-14T02:33:04	0.0	0.0	0.0
5888	5889	03/15/2020	West Virginia	US	2020-03-10T02:33:04	0.0	0.0	0.0
5889	5890	03/15/2020	NaN	occupied Palestinian territory	2020-03-11T20:53:02	0.0	0.0	0.0

In [5]:

```
#Grouping the data countrywise
df_countries = df2.groupby(['Country', 'Date']).max().reset_index().sort_values('Date', ascending=F
alse)
df_countries = df_countries.drop_duplicates(subset = ['Country'])
df_countries = df_countries[df_countries['Confirmed']>0]

df_countries
```

Out[5]:

	Country	Date	SNo	Province/State	Last Update	Confirmed	Deaths	Recovered
397	Cameroon	03/15/2020	5837	NaN	2020-03-13T22:22:02	2.0	0.0	0.0
1721	North Macedonia	03/15/2020	5784	NaN	2020-03-13T23:33:05	14.0	0.0	1.0
2434	Taiwan	03/15/2020	5724	NaN	2020-03-15T06:53:03	59.0	1.0	20.0
1740	Norway	03/15/2020	5644	NaN	2020-03-15T18:20:18	1221.0	3.0	1.0
447	Canada	03/15/2020	5855	Saskatchewan	2020-03-15T02:13:21	104.0	1.0	4.0
450	Cayman Islands	03/15/2020	5856	NaN	2020-03-13T12:33:03	1.0	0.0	0.0
451	Central African Republic	03/15/2020	5857	NaN	2020-03-15T18:20:19	1.0	0.0	0.0
465	Chile	03/15/2020	5722	NaN	2020-03-14T20:33:03	74.0	0.0	0.0
1454	Mainland China	03/15/2020	5858	Zhejiang	2020-03-15T18:20:18	67794.0	3085.0	54288.0
476	Colombia	03/15/2020	5746	NaN	2020-03-15T18:20:18	34.0	0.0	0.0
477	Congo (Brazzaville)	03/15/2020	5859	NaN	2020-03-15T18:20:19	1.0	0.0	0.0
482	Congo (Kinshasa)	03/15/2020	5840	NaN	2020-03-13T22:22:02	2.0	0.0	0.0
492	Costa Rica	03/15/2020	5759	NaN	2020-03-15T18:20:18	27.0	0.0	0.0
1761	Oman	03/15/2020	5770	NaN	2020-03-15T18:20:18	22.0	0.0	9.0
512	Croatia	03/15/2020	5731	NaN	2020-03-15T18:20:18	49.0	0.0	1.0
516	Cuba	03/15/2020	5823	NaN	2020-03-13T23:53:02	4.0	0.0	0.0

	Country	Date	SNo	Province/State	Last Update	Confirmed	Deaths	Recovered
518	Croatia	03/15/2020	5801	NaN	2020-03-14T10:15:18	1.0	0.0	0.0
525	Cyprus	03/15/2020	5760	NaN	2020-03-14T22:53:02	26.0	0.0	0.0
540	Czech Republic	03/15/2020	5674	NaN	2020-03-15T18:20:18	253.0	0.0	0.0
2380	Switzerland	03/15/2020	5640	NaN	2020-03-15T18:20:18	2200.0	14.0	4.0
1799	Others	03/15/2020	5657	Diamond Princess cruise ship	2020-03-13T14:13:25	696.0	7.0	325.0
558	Denmark	03/15/2020	5791	Faroe Islands	2020-03-15T18:20:19	864.0	2.0	1.0
1818	Pakistan	03/15/2020	5728	NaN	2020-03-15T18:20:18	53.0	0.0	2.0
573	Dominican Republic	03/15/2020	5792	NaN	2020-03-14T20:53:02	11.0	0.0	0.0
1400	Macau	03/15/2020	5795	Macau	2020-03-11T18:52:03	10.0	0.0	10.0
588	Ecuador	03/15/2020	5755	NaN	2020-03-14T22:53:02	28.0	2.0	0.0
1829	Panama	03/15/2020	5739	NaN	2020-03-15T18:20:18	43.0	1.0	0.0
619	Egypt	03/15/2020	5702	NaN	2020-03-15T18:20:18	110.0	2.0	21.0
620	Equatorial Guinea	03/15/2020	5862	NaN	2020-03-15T06:41:54	1.0	0.0	0.0
2360	Sweden	03/15/2020	5647	NaN	2020-03-15T18:20:18	1022.0	3.0	1.0
...	...	...	...	...	...	...	...	...
1961	Romania	03/15/2020	5693	NaN	2020-03-15T18:20:18	131.0	0.0	9.0
22	Afghanistan	03/15/2020	5780	NaN	2020-03-15T18:20:18	16.0	0.0	0.0
1667	Netherlands	03/15/2020	5646	NaN	2020-03-15T18:20:18	1135.0	20.0	2.0
79	Argentina	03/15/2020	5736	NaN	2020-03-15T18:20:18	45.0	2.0	1.0
49	Algeria	03/15/2020	5732	NaN	2020-03-15T18:20:18	48.0	4.0	12.0
94	Armenia	03/15/2020	5764	NaN	2020-03-15T18:20:18	26.0	0.0	0.0
1684	New Zealand	03/15/2020	5801	NaN	2020-03-15T18:20:18	8.0	0.0	0.0
66	Antigua and Barbuda	03/15/2020	5850	NaN	2020-03-15T18:20:19	1.0	0.0	0.0
1649	Nepal	03/15/2020	5876	NaN	2020-03-13T22:22:03	1.0	0.0	1.0
2676	United Arab Emirates	03/15/2020	5712	NaN	2020-03-15T18:20:18	98.0	0.0	23.0
2679	Uzbekistan	03/15/2020	5887	NaN	2020-03-15T18:20:19	1.0	0.0	0.0
29	Albania	03/15/2020	5741	NaN	2020-03-15T18:20:18	42.0	1.0	0.0
2738	Vietnam	03/15/2020	5726	NaN	2020-03-15T18:20:18	56.0	0.0	16.0
2678	Uruguay	03/15/2020	5827	NaN	2020-03-14T16:33:03	4.0	0.0	0.0
2685	Venezuela	03/15/2020	5803	NaN	2020-03-15T18:20:18	10.0	0.0	0.0
63	Andorra	03/15/2020	5849	NaN	2020-03-13T22:22:02	1.0	0.0	1.0
97	Aruba	03/15/2020	5835	NaN	2020-03-13T23:53:03	2.0	0.0	0.0
1701	Nigeria	03/15/2020	5846	NaN	2020-03-13T22:22:02	2.0	0.0	0.0
757	French Guiana	03/14/2020	5562	NaN	2020-03-07T03:23:05	5.0	0.0	0.0
835	Gibraltar	03/10/2020	4687	NaN	2020-03-10T08:33:03	1.0	0.0	1.0
2015	Saint Barthelemy	03/10/2020	4698	NaN	2020-03-09T10:43:06	1.0	0.0	0.0
1	('St. Martin',)	03/10/2020	4675	NaN	2020-03-10T05:33:02	2.0	0.0	0.0
650	Faroe Islands	03/10/2020	4672	NaN	2020-03-08T05:13:07	2.0	0.0	0.0
452	Channel Islands	03/10/2020	4686	NaN	2020-03-10T05:33:02	1.0	0.0	0.0
2310	St. Martin	03/09/2020	4412	NaN	2020-03-09T10:43:06	2.0	0.0	0.0
1823	Palestine	03/09/2020	4322	NaN	2020-03-07T02:23:06	22.0	0.0	0.0
2683	Vatican City	03/09/2020	4507	NaN	2020-03-06T15:43:02	1.0	0.0	0.0
1937	Republic of Ireland	03/08/2020	4067	NaN	2020-03-08T21:03:03	21.0	0.0	0.0
1702	North Ireland	02/28/2020	2685	NaN	2020-02-28T05:43:02	1.0	0.0	0.0
0	Azerbaijan	02/28/2020	2664	NaN	2020-02-28T15:03:26	1.0	0.0	0.0

160 rows × 8 columns

In [6]:

```
fig = go.Figure(data=go.Choropleth(
    locations = df_countries['Country'],
```

```
    locationmode = 'country names',
    z = df_countries['Confirmed'],
    colorscale = 'Reds',
    marker_line_color = 'black',
    marker_line_width = 0.5,
))

fig.update_layout(
    title_text = 'Confirmed Cases as on March 15, 2020',
    title_x = 0.5,
    geo=dict(
        showframe = False,
        showcoastlines = False,
        projection_type = 'equirectangular'
    )
)
```

As we can see, the highest number of confirmed cases (as on March 15, 2020), has been recorded in China, followed by Italy and then Iran. US, Brazil, Australia and other countries also have a considerable number of confirmed cases.

In [7]:

```
df_countrydate = df2[df2['Confirmed']>0]
df_countrydate = df_countrydate.groupby(['Date','Country']).sum().reset_index()
df_countrydate
```

Out[7]:

	Date	Country	SNo	Confirmed	Deaths	Recovered
0	01/22/2020	Japan	36	2.0	0.0	0.0
1	01/22/2020	Macau	21	1.0	0.0	0.0
2	01/22/2020	Mainland China	373	547.0	17.0	28.0
3	01/22/2020	South Korea	38	1.0	0.0	0.0
4	01/22/2020	Taiwan	29	1.0	0.0	0.0
5	01/22/2020	Thailand	37	2.0	0.0	0.0
6	01/22/2020	US	32	1.0	0.0	0.0
7	01/23/2020	Hong Kong	51	2.0	0.0	0.0

	Date	Country	SNo	Confirmed	Deaths	Recovered
8	01/23/2020	Japan	74	1.0	0.0	0.0
9	01/23/2020	Macau	59	2.0	0.0	0.0
10	01/23/2020	Mainland China	1529	639.0	18.0	30.0
11	01/23/2020	Singapore	77	1.0	0.0	0.0
12	01/23/2020	South Korea	76	1.0	0.0	0.0
13	01/23/2020	Taiwan	67	1.0	0.0	0.0
14	01/23/2020	Thailand	75	3.0	0.0	0.0
15	01/23/2020	US	70	1.0	0.0	0.0
16	01/23/2020	Vietnam	80	2.0	0.0	0.0
17	01/24/2020	France	125	2.0	0.0	0.0
18	01/24/2020	Hong Kong	110	2.0	0.0	0.0
19	01/24/2020	Japan	120	2.0	0.0	0.0
20	01/24/2020	Macau	111	2.0	0.0	0.0
21	01/24/2020	Mainland China	2887	916.0	26.0	36.0
22	01/24/2020	Singapore	123	3.0	0.0	0.0
23	01/24/2020	South Korea	122	2.0	0.0	0.0
24	01/24/2020	Taiwan	108	3.0	0.0	0.0
25	01/24/2020	Thailand	121	5.0	0.0	0.0
26	01/24/2020	US	237	2.0	0.0	0.0
27	01/24/2020	Vietnam	124	2.0	0.0	0.0
28	01/25/2020	Australia	167	4.0	0.0	0.0
29	01/25/2020	France	166	3.0	0.0	0.0
...	...	...	...	...	...	...
2702	03/15/2020	San Marino	5710	101.0	5.0	4.0
2703	03/15/2020	Saudi Arabia	5706	103.0	0.0	1.0
2704	03/15/2020	Senegal	5796	24.0	0.0	1.0
2705	03/15/2020	Serbia	5734	48.0	0.0	0.0
2706	03/15/2020	Seychelles	5847	2.0	0.0	0.0
2707	03/15/2020	Singapore	5676	226.0	0.0	105.0
2708	03/15/2020	Slovakia	5727	54.0	0.0	0.0
2709	03/15/2020	Slovenia	5677	219.0	1.0	0.0
2710	03/15/2020	South Africa	5729	51.0	0.0	0.0
2711	03/15/2020	South Korea	5636	8162.0	75.0	510.0
2712	03/15/2020	Spain	5637	7798.0	289.0	517.0
2713	03/15/2020	Sri Lanka	5793	18.0	0.0	1.0
2714	03/15/2020	Sudan	5880	1.0	1.0	0.0
2715	03/15/2020	Suriname	5881	1.0	0.0	0.0
2716	03/15/2020	Sweden	5647	1022.0	3.0	1.0
2717	03/15/2020	Switzerland	5640	2200.0	14.0	4.0
2718	03/15/2020	Taiwan	5724	59.0	1.0	20.0
2719	03/15/2020	Thailand	5699	114.0	1.0	35.0
2720	03/15/2020	Togo	5882	1.0	0.0	0.0
2721	03/15/2020	Trinidad and Tobago	5848	2.0	0.0	0.0
2722	03/15/2020	Tunisia	5775	18.0	0.0	0.0
2723	03/15/2020	Turkey	5812	6.0	0.0	0.0
2724	03/15/2020	UK	17365	1144.0	21.0	19.0
2725	03/15/2020	US	317386	3499.0	63.0	12.0
2726	03/15/2020	Ukraine	5833	3.0	1.0	0.0
2727	03/15/2020	United Arab Emirates	5712	98.0	0.0	23.0
2728	03/15/2020	Uruguay	5827	4.0	0.0	0.0

	Date	Country	SNo	Confirmed	Deaths	Recovered
2729	03/15/2020	Uzbekistan	5687	1.0	0.0	0.0
2730	03/15/2020	Venezuela	5803	10.0	0.0	0.0
2731	03/15/2020	Vietnam	5726	56.0	0.0	16.0

2732 rows × 6 columns

In [8]:

```
fig = px.choropleth(df_countrydate,
                    locations="Country",
                    locationmode = "country names",
                    color="Confirmed",
                    hover_name="Country",
                    animation_frame="Date"
                   )

fig.update_layout(
    title_text = 'Spread of Coronavirus',
    title_x = 0.5,
    geo=dict(
        showframe = False,
        showcoastlines = False,
    ))

fig.show()
```

This is an animated choropleth which shows how coronavirus has been been spreading, starting from 22nd January to 15th March, 2020. We can see that the spread began in China (as on 22nd Jan) and US, South Korea, Japan having one or two cases each and then gradually it spread to different parts of the world.

In [9]:

```
fig = px.pie(df_countries, values = 'Confirmed',names='Country', height=600)
fig.update_traces(textposition='inside', textinfo='percent+label')

fig.update_layout(
    title_x = 0.5,
    geo=dict(
        showframe = False,
        showcoastlines = False,
    ))
```

```
fig.show()
```

It's clear that the majority of confirmed cases remain in China. However, there has been a gargantuan increase in number of cases outside China, sum of which is more than that in China.

In [10]:

```
confirmed = df2.groupby('Date').sum()['Confirmed'].reset_index()
deaths = df2.groupby('Date').sum()['Deaths'].reset_index()
recovered = df2.groupby('Date').sum()['Recovered'].reset_index()
```

In [11]:

```
fig = go.Figure()
fig.add_trace(go.Bar(x=confirmed['Date'],
                y=confirmed['Confirmed'],
                name='Confirmed',
                marker_color='blue'
                ))
fig.add_trace(go.Bar(x=deaths['Date'],
                y=deaths['Deaths'],
                name='Deaths',
                marker_color='Red'
                ))
fig.add_trace(go.Bar(x=recovered['Date'],
                y=recovered['Recovered'],
                name='Recovered',
                marker_color='Green'
                ))

fig.update_layout(
    title='Worldwide CoronaVirus Cases - Confirmed, Deaths, Recovered (Bar Chart)',
    xaxis_tickfont_size=14,
    yaxis=dict(
        title='Number of Cases',
        titlefont_size=16,
        tickfont_size=14
```

```
    ),
    legend=dict(
        x=0,
        y=1.0,
        bgcolor='rgba(233, 233, 233, 0)',
        bordercolor='rgba(255, 255, 255, 0)'
    ),
    barmode='group',
    bargap=0.15, # gap between bars of adjacent location coordinates.
    bargroupgap=0.1 # gap between bars of the same location coordinate.
)
fig.show()
```

The bar chart shows confirmed, deaths, and recovered cases worldwide. We can see a spike in the number of confirmed cases on 13th February due to a new method of reclassifying confirmed cases, which is when Wuhan updated their curfew to a full lockdown. It started to plateau, but because of the release of the members on the Diamond Cruise Ship, we can see an increasing number of countries with cases towards the end of February. Since then, the numbers have inceased exponentially, especially outide China.

Now let's study these graphs individually.

In [12]:

```
bar_data = df2.groupby(['Country', 'Date'])['Confirmed', 'Deaths', 'Recovered'].sum().reset_index()
.sort_values('Date', ascending=True)

fig = px.bar(bar_data, x="Date", y="Confirmed", color='Country', text = 'Confirmed', orientation='v
', height=600,
             title='Confirmed')
fig.show()
```

There is no doubt that China has the largest number of cases. However, we can observe that the in case of Mainland China, the curve has flattened in late February and March, to a great extent. The increase is no longer exponential, which is a positive sign and also shows China's competency in dealing with this pandemic. On the other hand, there has been an exponential increase in confirmed cases outside China, especially in Italy.

In [13]:

```
fig = px.bar(bar_data, x="Date", y="Deaths", color='Country', text = 'Deaths', orientation='v', hei
ght=600,
             title='Deaths')
fig.show()
```

Similarly, death cases have also seen a gradual increase. The curve has flattened in China but is still exponential outside China,especially in Italy and Iran.

In [14]:

```
fig = px.bar(bar_data, x="Date", y="Recovered", color='Country', text = 'Recovered', orientation='v
', height=600,
             title='Recovered')
fig.show()
```

The bar chart is an indicator of the exponential increase in the number of recovered cases. The aim is to not let this graph flatten with time, till all the cases have recovered.

In [15]:

```
line_data = df2.groupby('Date').sum().reset_index()

line_data = line_data.melt(id_vars='Date',
                 value_vars=['Confirmed',
                             'Recovered',
                             'Deaths'],
                 var_name='Ratio',
                 value_name='Value')

fig = px.line(line_data, x="Date", y="Value", color='Ratio',
              title='Confirmed cases, Recovered cases, and Deaths Over Time')
fig.show()
```

This is a consolidated representation of the bar graphs shown above. Ideally, we would like to see the red line and blue line converge and pass each other.

For latest Date

In [16]:

```
df1.rename(columns={'Country/Region':'Country'}, inplace=True)
confirmed = df1.groupby(['Last Update', 'Country']).sum()[['Confirmed']].reset_index()
deaths = df1.groupby(['Last Update', 'Country']).sum()[['Deaths']].reset_index()
recovered = df1.groupby(['Last Update', 'Country']).sum()[['Recovered']].reset_index()
```

In [17]:

```
all_countries = confirmed['Country'].unique()
print("Number of countries/regions with cases: " + str(len(all_countries)))
print("Countries/Regions with cases: ")
for i in all_countries:
    print("    " + str(i))
```

```
Number of countries/regions with cases: 147
Countries/Regions with cases:
    US
    China
    occupied Palestinian territory
    Azerbaijan
    Cayman Islands
    Guadeloupe
    Cruise Ship
    Belarus
    Andorra
    Bhutan
    Burkina Faso
    Cameroon
    Congo (Kinshasa)
    Cote d'Ivoire
    Holy See
    Jamaica
    Mongolia
    Nigeria
    Togo
    Ukraine
    Guernsey
    Kenya
    Nepal
    North Macedonia
    Senegal
    Ethiopia
    Paraguay
    Cuba
    Aruba
    Canada
    Sudan
    Guatemala
    Australia
```

Bahrain
Georgia
Bolivia
Cambodia
Gabon
Martinique
United Kingdom
Trinidad and Tobago
Curacao
Saint Lucia
France
Jersey
Namibia
Saint Vincent and the Grenadines
Suriname
Uruguay
Ireland
Italy
Chile
Saudi Arabia
Dominican Republic
Liechtenstein
Moldova
Tunisia
Iraq
Japan
Qatar
Cyprus
Ecuador
Serbia
Monaco
Equatorial Guinea
Eswatini
Guinea
Rwanda
Taiwan*
Seychelles
Afghanistan
Albania
Algeria
Argentina
Armenia
Austria
Bangladesh
Belgium
Bosnia and Herzegovina
Brazil
Brunei
Bulgaria
Colombia
Costa Rica
Croatia
Czechia
Egypt
Estonia
Finland
Germany
Ghana
Greece
Hungary
Iceland
India
Indonesia
Iran
Israel
Jordan
Kazakhstan
Korea, South
Kuwait
Latvia
Lebanon
Lithuania
Luxembourg
Malaysia
Maldives
Malta
Mexico

```
Morocco
Netherlands
New Zealand
Norway
Oman
Pakistan
Panama
Peru
Philippines
Poland
Portugal
Romania
Russia
San Marino
Singapore
Slovakia
Slovenia
South Africa
Spain
Sri Lanka
Sweden
Switzerland
Thailand
Turkey
United Arab Emirates
Venezuela
Vietnam
Antigua and Barbuda
Central African Republic
Congo (Brazzaville)
Denmark
Guyana
Honduras
Kosovo
Mauritania
Reunion
Uzbekistan
```

In [18]:

```
line_data = df1.groupby('Country').sum().reset_index()

line_data = line_data.melt(id_vars='Country',
                 value_vars=['Confirmed',
                             'Recovered',
                             'Deaths'],
                 var_name='Ratio',
                 value_name='Value')

fig = px.line(line_data, x="Country", y="Value", color='Ratio',
              title='Confirmed cases, Recovered cases, and Deaths Over Time')
fig.show()
```

The line chart shows the number of confirmed, recovered and death cases across the globe, as on 15th March,2020. The mortality rate is around 4% in China whereas it is around 2.5% in Iran. This is less than that of SARS (10%) and MERS. Most of the casualties are senior citizens with underlying health problems.

Now, we'll focus on the the cases in China, where it was first detected.

In [19]:

```
df3 = df1[df1['Country']=='China']
df3
```

Out[19]:

	Province/State	Country	Last Update	Confirmed	Deaths	Recovered	Latitude	Longitude
0	Hubei	China	2020-03-15T18:20:18	67794	3085	54288	30.9756	112.2707
8	Guangdong	China	2020-03-15T18:20:18	1360	8	1304	23.3417	113.4244
9	Henan	China	2020-03-14T09:53:08	1273	22	1250	33.8820	113.6140
10	Zhejiang	China	2020-03-15T01:33:02	1231	1	1211	29.1832	120.0934
15	Hunan	China	2020-03-14T08:33:03	1018	4	1014	27.6104	111.7088
16	Anhui	China	2020-03-11T02:18:14	990	6	984	31.8257	117.2264
17	Jiangxi	China	2020-03-12T02:13:04	935	1	934	27.6140	115.7221
22	Shandong	China	2020-03-14T04:33:02	760	7	741	36.3427	118.1498
26	Jiangsu	China	2020-03-15T01:53:02	631	0	631	32.9711	119.4550
27	Chongqing	China	2020-03-15T03:53:04	576	6	570	30.0572	107.8740
28	Sichuan	China	2020-03-15T18:20:18	539	3	516	30.6171	102.7103
29	Heilongjiang	China	2020-03-15T00:33:02	482	13	453	47.8620	127.7615
30	Beijing	China	2020-03-15T01:13:10	442	8	353	40.1824	116.4142
33	Shanghai	China	2020-03-14T23:53:02	353	3	324	31.2020	121.4491
35	Hebei	China	2020-03-13T11:09:03	318	6	310	38.0428	114.5149
36	Fujian	China	2020-03-11T02:18:14	296	1	295	26.0789	117.9874
37	Guangxi	China	2020-03-13T11:09:03	252	2	243	23.8298	108.7881
38	Shaanxi	China	2020-03-12T01:33:02	245	2	232	35.1917	108.8701
47	Yunnan	China	2020-03-14T23:33:02	174	2	172	24.9740	101.4870
48	Hainan	China	2020-03-12T03:53:02	168	6	160	19.1959	109.7453
51	Guizhou	China	2020-03-14T00:53:03	146	2	143	26.8154	106.8748
52	Hong Kong	China	2020-03-15T18:20:18	145	4	81	22.3000	114.2000
55	Tianjin	China	2020-03-15T18:20:18	136	3	133	39.3054	117.3230
58	Gansu	China	2020-03-15T18:20:18	133	2	91	36.0611	103.8343
59	Shanxi	China	2020-03-13T11:09:03	133	0	133	37.5777	112.2922
62	Liaoning	China	2020-03-15T04:13:27	125	1	114	41.2956	122.6085
76	Jilin	China	2020-03-13T11:09:03	93	1	91	43.6661	126.1923
80	Xinjiang	China	2020-03-11T02:18:14	76	3	73	41.1129	85.2401
81	Inner Mongolia	China	2020-03-11T03:53:03	75	1	71	44.0935	113.9448
82	Ningxia	China	2020-03-14T09:53:08	75	0	73	37.2692	106.1655
141	Qinghai	China	2020-03-11T02:18:14	18	0	18	35.7452	95.9956
162	Macau	China	2020-03-11T18:52:03	10	0	10	22.1667	113.5500
225	Tibet	China	2020-03-11T02:18:14	1	0	1	31.6927	88.0924

In [20]:

```
line_data = df3.groupby('Province/State').sum().reset_index()

line_data = line_data.melt(id_vars='Province/State',
                 value_vars=['Confirmed',
                             'Recovered',
                             'Deaths'],
                 var_name='Ratio',
                 value_name='Value')

fig = px.line(line_data, x="Province/State", y="Value", color='Ratio',
              title='Confirmed cases, Recovered cases, and Deaths Over Time')
fig.show()
```

As we know, Wuhan, capital of Hubei has been the epicenter of the disease with almost 67,794 confirmed cases as on 15th March, with around 80% cases of recovery.

In [21]:

```
address = 'China'

geolocator = Nominatim()
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinates of the City of China are {}, {}.'.format(latitude, longitude))
```

```
c:\users\sheona\appdata\local\programs\python\python36\lib\site-packages\ipykernel_launcher.py:3:
DeprecationWarning:

Using Nominatim with the default "geopy/1.20.0" `user_agent` is strongly discouraged, as it
violates Nominatim's ToS https://operations.osmfoundation.org/policies/nominatim/ and may possibly
cause 403 and 429 HTTP errors. Please specify a custom `user_agent` with
`Nominatim(user_agent="my-application")` or by overriding the default `user_agent`:
`geopy.geocoders.options.default_user_agent = "my-application"`. In geopy 2.0 this will become an
exception.


The geograpical coordinates of the City of China are 35.000074, 104.999927.
```

In [22]:

```
# create map of world using latitude and longitude values
map_world = folium.Map(location=[latitude, longitude], zoom_start=2)

# add markers to map
for lat, lng, province in zip(df1['Latitude'], df1['Longitude'], df1['Province/State']):
    label = '{}'.format(province)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=3,
        popup=label,
        color='green',
        fill=True,
        fill_color='#3199cc',
        fill_opacity=0.3,
        parse_html=False).add_to(map_world)

map_world
```

Out[22]:

In [23]:

```
# create map of China using latitude and longitude values
map_china = folium.Map(location=[latitude, longitude], zoom_start=2)

# add markers to map
for lat, lng, province in zip(df3['Latitude'], df3['Longitude'], df3['Province/State']):
    label = '{}'.format(province)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=3,
        popup=label,
        color='green',
        fill=True,
        fill_color='#3199cc',
        fill_opacity=0.3,
        parse_html=False).add_to(map_china)

map_china
```

Out[23]:

The maps show the areas affected worldwide and then focusing on China, respectively.

We have a list of all the provinces in China where people have been affected by Covid-19.

Using Foursquare API, we will search for hospitals within 5 km of these areas, so as to see the distribution of hospitals in these provinces. This distribution plays an important role in determining accessibility to healthcare facilities in case of a health disaster like this.

In [24]:

```
# tranforming json file into a pandas dataframe library
from pandas.io.json import json_normalize

VERSION = '20180604'
LIMIT = 100
```

In [29]:

```
#These details have been removed
CLIENT_ID = "" # your Foursquare ID
CLIENT_SECRET = "" # your Foursquare Secret
```

In [26]:

```
df4 = pd.DataFrame()

for i in df3['Province/State']:
    address = i
    geolocator = Nominatim(user_agent="foursquare_agent")
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    #print(latitude, longitude)

    search_query = 'Hospitals'
    radius = 5000
    #print(search_query + ' .... OK!')
    url = 'https://api.foursquare.com/v2/venues/search?client_id={}&client_secret={}&ll={},{}&v={}
&query={}&radius={}&limit={}'.format(CLIENT_ID, CLIENT_SECRET, latitude, longitude, VERSION,
search_query, radius, LIMIT)
    results = requests.get(url).json()
    # assign relevant part of JSON to venues
```

```
    venues = results['response']['venues']

    # tranform venues into a dataframe
    df4 = df4.append(json_normalize(venues))

df4
```

```
c:\users\sheona\appdata\local\programs\python\python36\lib\site-
packages\pandas\core\frame.py:6201: FutureWarning:

Sorting because non-concatenation axis is not aligned. A future version
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=True'.

To retain the current behavior and silence the warning, pass sort=False
```

Out[26]:

	categories	hasPerk	id	location.address	location.cc	location.city	location.country	locati
0	[{'id': '4bf58dd8d48988d12b951735', 'name': 'B...	False	4c874c2a56e037041528b1a3	Fuxingmen	CN	西城区	中国	
0	[{'id': '4bf58dd8d48988d196941735', 'name': 'H...	False	4cbf146619ceb1f7724f21cd	山东中路145号	CN	Huangpu	中国	
1	[{'id': '4bf58dd8d48988d196941735', 'name': 'H...	False	4d06c849a2685481a099c3bd	东方路1630号 (近浦建路)	CN	上海市	中国	
0	[{'id': '52e81612bcbc57f1066b7a34', 'name': 'C...	False	5618ce90498e0c5c480c0638	129-133, G/F, Hung Shek House, Ping Shek Estate	HK	Choi Hung, Kowloon	香港	
1	[{'id': '4bf58dd8d48988d142941735', 'name': 'A...	False	4d2125b6f7a9a1437fbd339f	2/F, Block K, Queen Mary Hospital	HK	薄扶林	香港	
2	[{'id': '50328a8e91d4c4b30a586d6c', 'name': 'N...	False	4f3b1a01e4b0cdda83d5027f	12 Po Yan St	HK	Sheung Wan	香港	
3	[{'id': '4bf58dd8d48988d196941735', 'name': 'H...	False	4b484841f964a5200b4b26e3	102 Pok Fu Lam Rd	HK	薄扶林	香港	
4	[{'id': '50328a8e91d4c4b30a586d6c', 'name': 'N...	False	4cc2a4d901fb236a065ea0ba	Room 101-116, G/F, Tung Ping House, Lei Tung E...	HK	Ap Lei Chau	香港	
5	[{'id': '52e81612bcbc57f1066b7a34', 'name': 'C...	False	531a74e3498e5d9cb15edeef	Shop 2 & 9, June Garden, 28 Tung Chau St	HK	大角咀	香港	

In [27]:

```
# keep only columns that include venue name, and anything that is associated with location
filtered_columns = ['name', 'categories'] + [col for col in df4.columns if col.startswith('location
.')] + ['id']
dataframe_filtered = df4.loc[:, filtered_columns]

# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']

# filter the category for each row
dataframe_filtered['categories'] = dataframe_filtered.apply(get_category_type, axis=1)

# clean column names by keeping only last term
dataframe_filtered.columns = [column.split('.')[-1] for column in dataframe_filtered.columns]

dataframe_filtered
```

Out[27]:

	name	categories	address	cc	city	country	crossStreet	distance	formattedAddress	labeledLatLngs	lat
0	Beijing Children's Hospital Station 北京儿童医院站	Bus Line	Fuxingmen	CN	西城区	中国	NaN	3473	[Fuxingmen, 西城区, 北京市, 中国]	[{'label': 'display', 'lat': 39.91267170249076...	39.912672
0	上海交大医学院附属上海仁济医院 Renji Hospital SJTU School of ...	Hospital	山东中路145号	CN	Huangpu	中国	仁济西院	1094	[山东中路145号 (仁济西院), Huangpu, 上海市, 200001, 中国]	[{'label': 'display', 'lat': 31.235408671631 14...	31.235409
1	Renji Hospital SJTU School of Medicine (上海交大医学...	Hospital	东方路1630号 (近浦建路)	CN	上海市	中国	仁济东院	5344	[东方路1630号 (近浦建路) (仁济东院), 上海市, 上海市, 中国]	[{'label': 'display', 'lat': 31.208544482796 53...	31.208544
0	Tung Wah Group of Hospitals Pong Wing Shiu Nei...	Community Center	129-133, G/F, Hung Shek House, Ping Shek Estate	HK	Choi Hung, Kowloon	香港	NaN	4414	[129-133, G/F, Hung Shek House, Ping Shek Esta...	[{'label': 'display', 'lat': 22.318407, 'lng':...	22.318407
1	Hospital Staff Restaurant 醫院職員餐廳	Asian Restaurant	2/F, Block K, Queen Mary Hospital	HK	薄扶林	香港	NaN	3424	[2/F, Block K, Queen Mary Hospital, 香港]	[{'label': 'display', 'lat': 22.270907037227 34...	22.270907
2	The Tung Wah Group of Hospitals (東華三院)	Non-Profit	12 Po Yan St	HK	Sheung Wan	香港	NaN	1664	[12 Po Yan St, 香港]	[{'label': 'display', 'lat': 22.286792040800 26...	22.286792
3	Queen Mary Hospital (瑪麗醫院)	Hospital	102 Pok Fu Lam Rd	HK	薄扶林	香港	NaN	3432	[102 Pok Fu Lam Rd, 香港]	[{'label': 'display', 'lat': 22.269891, 'lng':...	22.269891
4	TWGHs Jockey Club Lei Tung Integrated Services...	Non-Profit	Room 101-116, G/F, Tung Ping House, Lei Tung E...	HK	Ap Lei Chau	香港	NaN	4227	[Room 101-116, G/F, Tung Ping House, Lei Tung ...	[{'label': 'display', 'lat': 22.241666666666 66...	22.241667

	name	categories	address	cc	city	country	crossStreet	distance	formattedAddress	labeledLatLngs	lat
5	TWGHs Jockey Club Tai Kok Tsui Integrated Serv...	Community Center	Shop 2 & 9, June Garden, 28 Tung Chau St	HK	大角咀	香港	NaN	4942	[Shop 2 & 9, June Garden, 28 Tung Chau St, 香港]	[{'label': 'display', 'lat': 22.323712468000092...	22.323712

In [28]:

```
hospitals_map = folium.Map(location=[35.000074,104.999927], zoom_start=3) # generate map centred in China

# add the hospitals as blue circle markers
for lat, lng, label in zip(dataframe_filtered.lat, dataframe_filtered.lng, dataframe_filtered.categories):
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        color='blue',
        popup=label,
        fill = True,
        fill_color='blue',
        fill_opacity=0.6
    ).add_to(hospitals_map)

# display map
hospitals_map
```

Out[28]:

So on plotting the dataframe with hospital related information, we can see only 3 hospitals on the map. Foursquare API had returned some more venues which were filtered out based on the category.

Results and Discussion

Based on visualization of our data, we can say that Covid-19, which was first detected in China has now become a pandemic because of the way it has spread across the globe. Numbers have somewhat stabilized in China. However, the cases are growing at an alarming rate outside China, especially in Italy and Iran.Italy has the most recorded cases outside China. Using Foursquare API and Folium library, we have visualized the distribution of hospitals in different Covid-hit provinces of China. Such visualizations can be used to compare the health facilities of different countries to see how well prepared they are, in case of a pandemic like this. The mortality rate of Covid-19 has been less than SARS, MERS so far and close to that of Spanish Flu. However, the large number of confirmed cases proves that it spreads very easily.

Conclusion

With no vaccine or treatment that can prevent it yet, containing the spread of Covid-19 is vital. As a final note, all of the above analysis is dependent on the adequacy and accuracy of the data collected from various sources and Foursquare API. A more comprehensive analysis and future work would need incorporation of data from other external databases.