Introduction

Visualizing the number of COVID-19 cases - ***Confirmed, Deaths, Recovered*** - in Singapore over time.

Table of Contents

In [1]:

```
import json
import requests
import pandas as pd
from datetime import datetime as dt

# visualization
import plotly.express as px
import matplotlib.pyplot as plt
```

In [2]:

```
import warnings
with warnings.catch_warnings():
    warnings.simplefilter("ignore")
```

1. API Call

In [3]:

```
API_URL =
"https://api.covid19api.com/total/dayone/country/singapore"
```

The API returns all the cases by case type for Singapore from the first recorded case.

In [4]:

```
response = requests.get(API_URL)
response_dict = json.loads(response.text)
```

In [5]:

```
covid19_cases_df = pd.DataFrame(response_dict)
```

In [6]:

```
covid19_cases_df.tail(10)
```

Out[6]:

	Country	CountryCode	Province	City	CityCode	Lat	Lon	Confirmed	Deaths	Recove
663	Singapore					0	0	241341	612	
664	Singapore					0	0	244815	619	

	Country	CountryCode	Province	City	CityCode	Lat	Lon	Confirmed	Deaths	Recove
665	Singapore					0	0	244815	619	
666	Singapore					0	0	248587	641	
667	Singapore					0	0	250518	654	
668	Singapore					0	0	252188	662	
669	Singapore					0	0	253649	667	
670	Singapore					0	0	255431	672	
671	Singapore					0	0	257510	678	
672	Singapore					0	0	258785	681	

2. Data Formatting

Keeping the required columns

In [7]:
```
covid19_cases_df = covid19_cases_df[['Date', 'Confirmed',
'Deaths', 'Recovered']]
```

Formatting timestamp to keep only the date

In [8]:
```
covid19_cases_df['Date'] =
pd.to_datetime(covid19_cases_df['Date'], errors='coerce')
covid19_cases_df['date'] = covid19_cases_df.Date.dt.strftime('%Y-
%m-%d')
```

In [9]:
```
# Dropping the exsisting Date column with timestamp since it is
not required anymore
covid19_cases_df.drop('Date', axis=1, inplace=True)
```

In [10]:
```
covid19_cases_df
```

Out[10]:

	Confirmed	Deaths	Recovered	date
0	1	0	0	2020-01-23
1	3	0	0	2020-01-24
2	3	0	0	2020-01-25
3	4	0	0	2020-01-26
4	5	0	0	2020-01-27
...	...	...	...	...

	Confirmed	Deaths	Recovered	date
668	252188	662	0	2021-11-21
669	253649	667	0	2021-11-22
670	255431	672	0	2021-11-23
671	257510	678	0	2021-11-24
672	258785	681	0	2021-11-25

673 rows × 4 columns

Merging the cases types - **Confirmed, Deaths, Recovered** - into a single column for visualisation

In [11]:
```
covid19_cases_df = pd.melt(covid19_cases_df, id_vars=['date'],
                value_vars=['Confirmed', 'Deaths', 'Recovered'],
                value_name="cases", var_name="case_type")
covid19_cases_df
```

Out[11]:

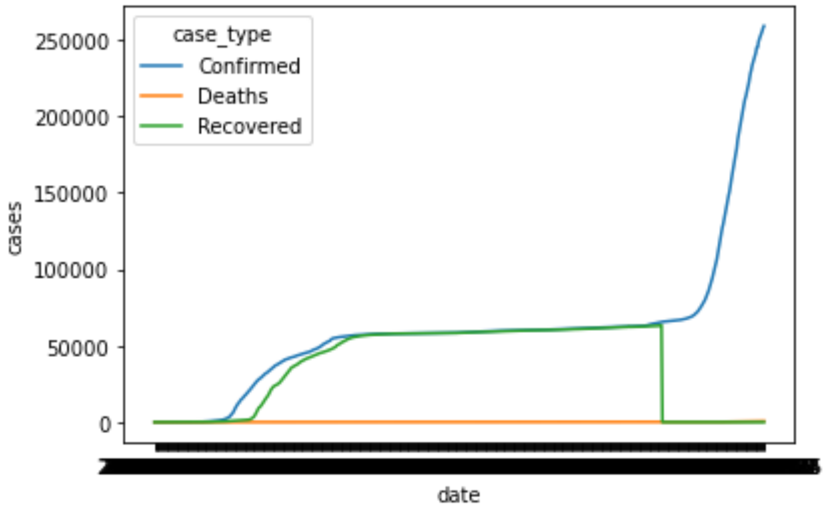
	date	case_type	cases
0	2020-01-23	Confirmed	1
1	2020-01-24	Confirmed	3
2	2020-01-25	Confirmed	3
3	2020-01-26	Confirmed	4
4	2020-01-27	Confirmed	5
...	...	...	...
2014	2021-11-21	Recovered	0
2015	2021-11-22	Recovered	0
2016	2021-11-23	Recovered	0
2017	2021-11-24	Recovered	0
2018	2021-11-25	Recovered	0

2019 rows × 3 columns

3. Visualizations

- Using Seaborn

In [12]:
```
import seaborn as sns
sns.lineplot(data=covid19_cases_df, x="date", y="cases",
hue="case_type")
plt.show()
```



- Using Plotly for the dashboard (better visualizations)

In [14]:
```
fig = px.line(covid19_cases_df, x="date", y="cases",
color='case_type', template='plotly_white')
fig.update_layout(
    title='Covid-19 cases in Singapore overtime',
    font=dict(
        family='Verdana, monospace',
        size=12
    ),
)
fig.show()
```