LAPORAN

MATA KULIAH ANALISIS DATA EKSPLORATIF (A)

"ANALISIS KORELASI, TIME SERIES ANALYSIS, HYPOTESIS TESTING, DAN REGRESI"

DISUSUN OLEH:

Reza Putri Angga (22083010006)

DOSEN PENGAMPU:

Tresna Maulana Fahrudin S.ST., M.T. (NIP. 199305012022031007)

PROGRAM STUDI SAINS DATA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN" JAWA TIMUR

2023

STUDI KASUS DAN PEMBAHASAN

- **Melakukan Proses Pembacaan *(Load)* Dataset Covid-19 Yang Akan Di Analisis Lebih Lanjut**

Melakukan Proses Pembacaan (Load) Dataset Covid-19 Yang Akan Di Analisis Lebih Lanjut

In [1]:
```
import pandas as pd

DataCovid = pd.read_csv("covid_19_indonesia_time_series_all.csv")
DataCovid
```

Out[1]:

	Date	Location ISO Code	Location	New Cases	New Deaths	New Recovered	New Active Cases	Total Cases	Total Deaths	Total Recovered	...	Latitude	New Cases per Million	Total Cases per Million	New Deaths per Million	Total Deaths per Million
0	3/1/2020	ID-JK	DKI Jakarta	2	0	0	2	39	20	75	...	-6.204699	0.18	3.60	0.0	1.84
1	3/2/2020	ID-JK	DKI Jakarta	2	0	0	2	41	20	75	...	-6.204699	0.18	3.78	0.0	1.84
2	3/2/2020	IDN	Indonesia	2	0	0	2	2	0	0	...	-0.789275	0.01	0.01	0.0	0.00
3	3/2/2020	ID-RI	Riau	1	0	0	1	1	0	1	...	0.511648	0.16	0.16	0.0	0.00
4	3/3/2020	ID-JK	DKI Jakarta	2	0	0	2	43	20	75	...	-6.204699	0.18	3.96	0.0	1.84
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
31817	9/15/2022	ID-SA	Sulawesi Utara	37	0	0	37	52770	1213	50997	...	1.259638	14.01	19974.38	0.0	459.14
31818	9/15/2022	ID-SB	Sumatera Barat	13	0	3	10	104640	2371	102066	...	-0.850253	2.36	18959.11	0.0	429.59
31819	9/15/2022	ID-SS	Sumatera Selatan	16	0	1	15	82198	3376	78510	...	-3.216212	1.95	10002.74	0.0	410.83
31820	9/15/2022	ID-SU	Sumatera Utara	50	0	5	45	158866	3288	154924	...	2.191894	3.36	10680.15	0.0	221.04
31821	9/16/2022	IDN	Indonesia	2358	27	2997	-666	6405044	157876	6218708	...	-0.789275	8.89	24153.07	0.1	595.34

31822 rows × 38 columns

Pada kode script di atas, di lakukan proses *load* atau pembacaan dataset. Dengan menggunakan *library* pandas yang di permisalkan sebagai pd. Terdapat variabel DataCovid yang di pergunakan untuk menyimpan *dataframe* dari dataset "covid_19_indonesia_time_series_all.csv". Maka, akan di tampilkan *dataframe* dari dataset Covid-19 di Indonesia dengan total 31822 baris dan 38 kolom.

Dari dataset kasus Covid-19 di Indonesia di rentang tahun 2020-2022 di atas, akan di lakukan implementasi teknik-teknik analisis korelasi, *time series analysis*, *hypothesis testing*, dan regresi untuk menyelesaikan berbagai permasalahan data Covid-19 di Indonesia. Dengan menggunakan *field (record data)* di kolom *"Date"* dan *"New Cases"*. Selanjutnya, akan di bahas implementasi dari poin–poin yang telah di sebutkan sebelumnya, sebagai berikut.

A. Time Series Analysis

1. Visualisasi Data Selama Rentang Beberapa Tahun

A. Time Series Analysis

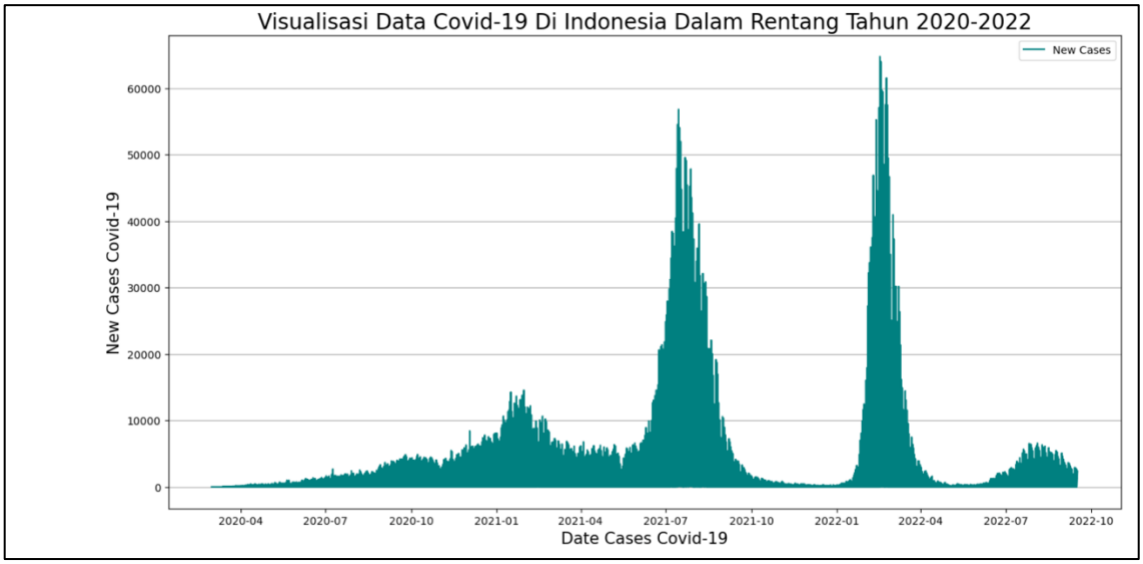
1. Visualisasi Data Selama Rentang Beberapa Tahun

In [4]:
```
import matplotlib.pyplot as plt

#mengkonversi tipe data pada kolom "date" menjadi format datetime
DataCovid["Date"] = pd.to_datetime(DataCovid["Date"])

#visualisasi data covid-19 dalam rentang tahun
plt.figure(figsize = (16, 8))
plt.plot(DataCovid["Date"], DataCovid["New Cases"], color = "teal", label = "New Cases")
plt.grid(True, which = "major", axis = "y", linestyle = "-", color = "grey", alpha = 0.7)
plt.title("Visualisasi Data Covid-19 Di Indonesia Dalam Rentang Tahun 2020-2022", fontsize = 20)
plt.xlabel("Date Cases Covid-19", fontsize = 15)
plt.ylabel("New Cases Covid-19", fontsize = 15)
plt.legend()

plt.show()
```



Pada kode script di atas, di lakukan proses visualisasi data dari kolom *"date"* dan *"new cases"* Covid-19 di Indonesia dalam rentang tahun 2020-2022 dengan menggunakan *library* matplotlib yang di permisalkan sebagai plt. Di lakukan proses pengubahan tipe data dalam kolom *"date"* menjadi datetime dengan menggunakan *library* pandas yang di permisalkan sebagai pd. Dengan kolom *"date"* sebagai sumbu x dan kolom *"new cases"* sebagai sumbu y.

Berdasarkan visualisasi yang di hasilkan, dapat di perlihatkan bahwa jumlah kasus harian Covid-19 di Indonesia mengalami peningkatan signifikan di awal tahun 2021 tepatnya di bulan Juli sebanyak lebih dari 50000, setelahnya jumlah kasus harian Covid-19 menurun secara bertahap. Kemudian di tahun 2022 tepatnya di bulan April 2022 juga mengalami peningkatan signifikan sebanyak lebih dari 60000, setelahnya jumlah kasus harian Covid-19 di Indonesia mengalami penurunan.

2. Pengelompokan (Grouping) Data Time Series Berdasarkan Bulan

2.1 Grouping Data Berdasarkan Bulan Di Semua Tahun

2. Pengelompokan (Grouping) Data Time Series Berdasarkan Bulan

2.1 Grouping Data Berdasarkan Bulan Di Semua Tahun

In [44]:
```
#telah di lakukan pengubahan tipe data kolom "date" menjadi format datetime di langkah sebelumnya
#melakukan pembuatan kolom "month/bulan" dari ekstrasi data di kolom "date"
DataCovid["Month"] = DataCovid["Date"].dt.month

#melakukan pengelompokkan data
GroupingData = DataCovid.groupby("Month")["New Cases"].sum().reset_index()

#melakukan pengurutan data berdasarkan bulan
GroupingSort = GroupingData.sort_index()

#melakukan penampilkan grouping data
print("Di Tampilkan Grouping Data Berdasarkan Bulan Di Semua Tahun : ")
GroupingSort[["Month", "New Cases"]]
```

```
Di Tampilkan Grouping Data Berdasarkan Bulan Di Semua Tahun :
```

Out[44]:

	Month	New Cases
0	1	847513
1	2	2934624
2	3	1251927
3	4	398500
4	5	356933
5	6	843214
6	7	2799986
7	8	1797902
8	9	560379
9	10	305804
10	11	282650
11	12	422921

Pada kode script di atas, di lakukan proses *grouping* data berdasarkan bulan di semua tahun. Terdapat proses pembuatan kolom *"Month"* dengan data yang di hasilkan di peroleh dari ekstrasi kolom *"Date"* menggunakan metode dt.month. Lalu, di lakukan proses *grouping* data berdasarkan nilai unik dalam kolom *"Month"* dan di ambil nilai dari kolom *"New Cases"*. Selanjutnya, di lakukan penjumlahan untuk setiap kasus di dalam bulan tertentu.

Maka, akan di tampilkan *grouping* data dari bulan 1-12 atau Januari sampai dengan Desember dengan total kasus pada bulan tersebut, seperti pada bulan 1 atau Januari terdapat total *new cases* harian Covid-19 sebesar 847513.

2.2 Grouping Data Berdasarkan Bulan Di Masing–Masing Tahun

2.2 Grouping Data Pada Setiap Bulan Di Masing-Masing Tahun

In [51]:
```
#telah di lakukan pengubahan tipe data kolom "date" menjadi format datetime di langkah sebelumnya
#melakukan pembuatan kolom "month/bulan" dari ekstrasi data di kolom "date"
DataCovid["Month"] = DataCovid["Date"].dt.strftime("%m-%Y")

#melakukan pengelompokkan data
GroupingData = DataCovid.groupby("Month")["New Cases"].sum().reset_index()

#melakukan ekstrasi data dari "month" dan "year"
GroupingData["Year"] = GroupingData["Month"].str[-4:].astype(int)
GroupingData["MonthNum"] = GroupingData["Month"].str[:2].astype(int)

#melakukan pengurutan data berdasarkan bulan pada tahun tertentu
GroupingSort = GroupingData.sort_values(by = ["Year", "MonthNum"]).drop(["Year", "MonthNum"],
                                                                        axis = 1)
GroupingSort.reset_index(drop = True, inplace = True)

#melakukan penampilkan grouping data
print("Di Tampilkan Grouping Data Pada Setiap Bulan Di Masing-Masing Tahun : ")
GroupingSort[["Month", "New Cases"]]
```

```
Di Tampilkan Grouping Data Pada Setiap Bulan Di Masing-Masing Tahun :
```

Out[51]:

	Month	New Cases
0	03-2020	2548
1	04-2020	17248
2	05-2020	33546
3	06-2020	60649
4	07-2020	105594
5	08-2020	133379
6	09-2020	224976
7	10-2020	247294
8	11-2020	258517
9	12-2020	410295
10	01-2021	666208
11	02-2021	512461
12	03-2021	352648
13	04-2021	313292
14	05-2021	307028
15	06-2021	715593
16	07-2021	2457118
17	08-2021	1361101
18	09-2021	250605
19	10-2021	58510
20	11-2021	24133
21	12-2021	12626
22	01-2022	181305
23	02-2022	2422163
24	03-2022	896731
25	04-2022	67960
26	05-2022	16359
27	06-2022	66972
28	07-2022	237274
29	08-2022	303422
30	09-2022	84798

Pada kode script di atas, di lakukan proses *grouping* data berdasarkan bulan di masing-masing tahun. Terdapat proses pembuatan kolom *"Month"* dengan data yang di hasilkan di peroleh dari ekstrasi kolom *"Date"* menggunakan metode dt.month. Lalu, di lakukan proses *grouping* data berdasarkan nilai unik dalam kolom *"Month"* dan di ambil nilai dari kolom *"New Cases".* Selanjutnya, di lakukan penjumlahan untuk setiap kasus di dalam bulan tertentu dan di lakukan proses ekstrasi tahun dan bulan numerik untuk di lakukan pengurutan.

Maka, akan di tampilkan hasil dari *grouping* data dari setiap bulan 1-12 atau Januari sampai Desember pada masing-masing tahun di rentang tahun 2020-2022, seperti pada bulan 03 atau Maret 2020 terdapat banyak total *new cases* Covid-19 di Indonesia sebesar 2548.

3. Pengelompokan (Grouping) Data Time Series Berdasarkan Hari

3.1 Grouping Data Berdasarkan Hari Di Semua Tahun

3. Pengelompokan (Grouping) Data Time Series Berdasarkan Hari

3.1 Grouping Data Berdasarkan Hari Di Semua Tahun

In [57]:
```
#telah di lakukan pengubahan tipe data kolom "date" menjadi format datetime di langkah sebelumnya
#melakukan pembuatan kolom "day/hari" dari ekstrasi data di kolom "date"
DataCovid["Day"] = DataCovid["Date"].dt.day
DataCovid["Day Name"] = DataCovid["Date"].dt.day_name()

#melakukan pengelompokkan data
#melakukan pengelompokkan data dari kolom "day name" dengan jumlah "new cases" disemua tahun
GroupingData = DataCovid.groupby("Day Name")["New Cases"].sum().reset_index()

#melakukan inisiasi "day name/nama hari"
NamaHari = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]

#melakukan pengelompokkan data sesuai inisiasi "day name/nama hari"
GroupingData["Day Name"] = pd.Categorical(GroupingData["Day Name"], categories = NamaHari, ordered = True)

#melakukan pengurutan data berdasarkan hari
GroupingData.sort_values("Day Name", inplace = True)
GroupingData.reset_index(drop = True, inplace = True)

#melakukan penampilkan grouping data
print("Di Tampilkan Grouping Data Berdasarkan Hari Di Semua Tahun : ")
GroupingData[["Day Name", "New Cases"]]
```

```
Di Tampilkan Grouping Data Berdasarkan Hari Di Semua Tahun :
```

Out[57]:

	Day Name	New Cases
0	Monday	1448154
1	Tuesday	1844139
2	Wednesday	1975579
3	Thursday	2015644
4	Friday	1953851
5	Saturday	1896767
6	Sunday	1668219

Pada kode script di atas, di lakukan proses *grouping* data berdasarkan hari di semua tahun. Terdapat proses pembuatan kolom *"Day"* dan *"Day Name"* dengan data yang di hasilkan di peroleh dari ekstrasi kolom *"Date"* dengan menggunakan metode dt.day dan dt.day_name(). Lalu, di lakukan proses *grouping* data berdasarkan "*Day Name"* yang telah di inisiasi dalam variabel NamaHari dan di ambil nilai *"New Cases"*. Selanjutnya, di lakukan penjumlahan untuk setiap kasus di dalam hari tertentu.

Maka, akan di tampilkan hasil dari *grouping* data dari hari *Monday-Sunday* atau Senin sampai dengan Minggu dengan total kasus pada hari tersebut, seperti pada hari *Monday* terapat total *new cases* harian Covid-19 sebesar 1448154.

3.2 Grouping Data Berdasarkan Hari Di Masing–Masing Tahun

3.2 Grouping Data Pada Setiap Hari Di Masing-Masing Tahun

In [61]:
```
#telah di lakukan pengubahan tipe data kolom "date" menjadi format datetime di langkah sebelumnya
#melakukan pengambilan data untuk "year/tahun" dari kolom "date"
DataCovid["Year"] = DataCovid["Date"].dt.year
#melakukan pembuatan kolom "day/hari" dari ekstrasi data di kolom "date"
DataCovid["Day"] = DataCovid["Date"].dt.day
DataCovid["Day Name"] = DataCovid["Date"].dt.day_name()

#melakukan pengelompokkan data
#melakukan pengelompokkan data dari kolom "day name" dengan jumlah "new cases" disemua tahun
GroupingData = DataCovid.groupby(["Year", "Day Name"])["New Cases"].sum().reset_index()

#melakukan inisiasi "day name/nama hari"
NamaHari = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]

#melakukan pengelompkan data berdasarkan masing-masing "new cases" di setiap "day name", di masing-masing tahun
GroupingData["Day Name"] = pd.Categorical(GroupingData["Day Name"], categories = NamaHari, ordered = True)

#melakukan pengurutan data berdasarkan hari
GroupingData.sort_values(["Year", "Day Name"], inplace = True)
GroupingData.reset_index(drop = True, inplace = True)

#melakukan penampilkan grouping data
print("Di Tampilkan Grouping Data Berdasarkan Hari Di Masing-Masing Tahun : ")
GroupingData[["Year", "Day Name", "New Cases"]]
```

```
Di Tampilkan Grouping Data Berdasarkan Hari Di Masing-Masing Tahun :
```

Out[61]:

	Year	Day Name	New Cases
0	2020	Monday	183926
1	2020	Tuesday	202756
2	2020	Wednesday	224594
3	2020	Thursday	240008
4	2020	Friday	217853
5	2020	Saturday	218958
6	2020	Sunday	205951
7	2021	Monday	833710
8	2021	Tuesday	1004581
9	2021	Wednesday	1039297
10	2021	Thursday	1094216
11	2021	Friday	1111176
12	2021	Saturday	1030123
13	2021	Sunday	918220
14	2022	Monday	430518
15	2022	Tuesday	636802
16	2022	Wednesday	711688
17	2022	Thursday	681420
18	2022	Friday	624822
19	2022	Saturday	647686
20	2022	Sunday	544048

Pada kode script di atas, di lakukan proses *grouping* data berdasarkan hari di masing-masing tahun. Terdapat proses pembuatan kolom *"Year", "Day",* dan *"Day Name"* dengan data yang di hasilkan di peroleh dari ekstrasi kolom *"Date"* dengan menggunakan metode dt.year, dt.day, dan dt.day_name(). Lalu, di lakukan proses *grouping* data berdasarkan "Year", "*Day Name"* yang telah di inisiasi dalam variabel NamaHari dan di ambil nilai *"New Cases"*. Selanjutnya, di lakukan penjumlahan untuk setiap kasus di dalam hari tertentu. Dan di lakukan proses pengurutan. Maka, akan di tampilkan hasil dari *grouping* data dari setiap hari *Monday-Sunday* atau Senin sampai dengan Minggu pada masing-masing tahun di

rentang tahun 2020-2022, seperti pada hari *Monday* atau Senin tahun 2020 terdapat total *new cases* harian Covid-19 sebesar 183926.

3.3 Grouping Data Pada Setiap Hari Di Semua Tahun

3.3 Grouping Data Pada Setiap Hari Di Semua Tahun

In [63]:
```
#telah di lakukan pengubahan tipe data kolom "date" menjadi format datetime di langkah sebelumnya
#melakukan pengambilan "day/hari" dari kolom "date"
DataCovid["Day"] = DataCovid["Date"].dt.day

#melakukan pengelompokan data
GroupingData = DataCovid.groupby("Day")["New Cases"].sum().reset_index()

#melakukan penampilan grouping data
print("Di Tampilkan Grouping Data Pada Setiap Hari Di Tahun 2020-2022 : ")
GroupingData
```

```
Di Tampilkan Grouping Data Pada Setiap Hari Di Tahun 2020-2022 :
```

Out[63]:

	Day	New Cases
0	1	365185
1	2	381198
2	3	425885
3	4	415384
4	5	429385
5	6	436006
6	7	407955
7	8	441681
8	9	439607
9	10	431179
10	11	410299
11	12	442050
12	13	442880
13	14	422623
14	15	464555
15	16	479136
16	17	478587
17	18	432630
18	19	433600
19	20	416860
20	21	369676
21	22	449586
22	23	452008
23	24	455120
24	25	424020
25	26	405397
26	27	401607
27	28	384772
28	29	323060
29	30	317223
30	31	223199

Pada kode script di atas, di lakukan proses *grouping* data pada setiap hari di tahun 2020-2022 di mana jika di kode sebelumnya menggunakan nama-nama hari, namun jika di kode ini menggunakan hari 1-31. Di lakukan pembuatan kolom *"Day"* dengan data yang di hasilkan di peroleh dari ekstrasi kolom *"Date"* menggunakan metode dt.day. Lalu, di lakukan proses *grouping* data dari kolom *"Day"* dan di ambil nilai total dari kolom *"New Cases"*. Maka, akan di tampilkan *grouping* data berdasarkan hari 1-31, seperti pada hari ke-1 terdapat total *new cases* harian sebesar 365185.

4. Visualisasi Bersama Rata-Rata Mingguan Kasus Covid-19 Dan Harian

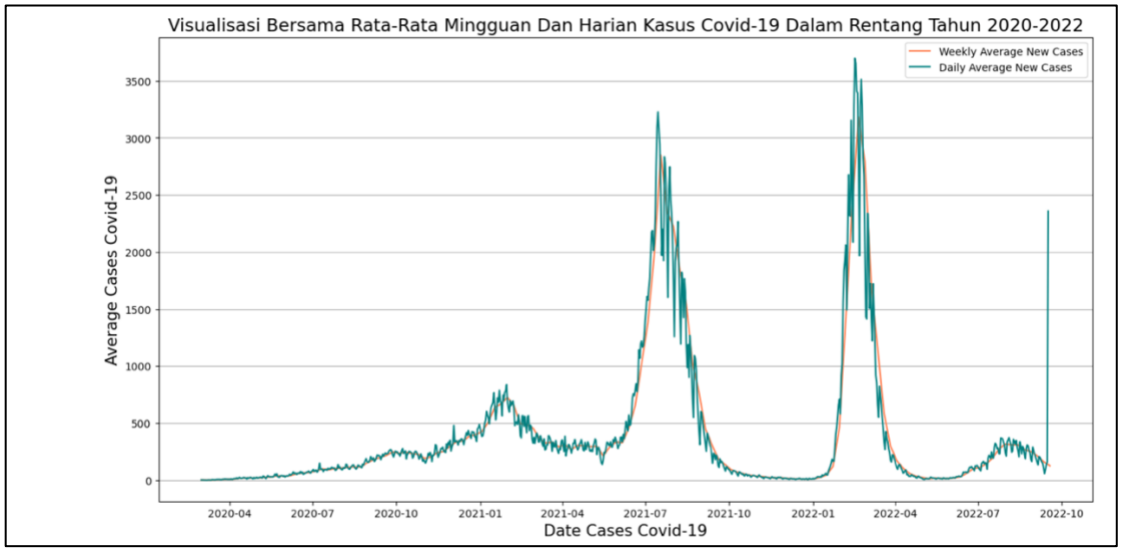
4. Visualisasi Bersama Rata-Rata Mingguan Kasus Covid-19 Dan Harian

In [65]:
```
import matplotlib.pyplot as plt

#menghitung rata-rata mingguan dan harian kasus COVID-19
WeeklyAverageCases = DataCovid.resample("W-Sun", on = "Date")["New Cases"].mean()
DailyAverageCases = DataCovid.resample("D", on = "Date")["New Cases"].mean()

#visualisasi rata-rata mingguan dan harian kasus COVID-19
plt.figure(figsize=(16, 8))
plt.plot(WeeklyAverageCases.index, WeeklyAverageCases.values, label = "Weekly Average New Cases",
         color = "coral")
plt.plot(DailyAverageCases.index, DailyAverageCases.values, label = "Daily Average New Cases", color = "teal")
plt.grid(True, which = "major", axis = "y", linestyle = "-", color = "grey", alpha = 0.7)
plt.title("Visualisasi Bersama Rata-Rata Mingguan Dan Harian Kasus Covid-19 Dalam Rentang Tahun 2020-2022",
          fontsize = 17)
plt.xlabel("Date Cases Covid-19", fontsize = 15)
plt.ylabel("Average Cases Covid-19", fontsize = 15)
plt.legend()

plt.show()
```



Pada kode script di atas, di lakukan proses visualisasi bersama antara rata-rata mingguan dan harian dari kasus Covid-19 di Indonesia menggunakan library matplotlib yang di permisalkan sebagai plt. Lalu, di lakukan proses perhitungan rata-rata mingguan *"WeeklyAverageCases"* dengan ekstrasi informasi dari kolom *"Date"* dengan frekuensi mingguan menggunakan metode W-Sun dan perhitungan rata-rata harian *"DailyAverageCases"* dengan ekstrasi dari kolom *"Date"* dengan frekuensi harian menggunakan metode D. Lalu di lakukan proses visualisasi bersama dengan kolom *"Date"* sebagai sumbu x dengan sumbu y terdapat *"WeeklyAverageCases"* yang di representasikan dengan warna coral dan "DailyAverageCases" yang di representasikan dengan warna teal.

Berdasarkan visualisasi yang di hasilkan, dapat di perlihatkan plot dalam bentuk garis waktu dengan mewakili rata-rata kasus harian dan mingguan pada periode tertentu. Dapat di perlihatkan bahwa rata-rata mingguan kasus Covid-19 mengalami peningkatan signifikan pada awal tahun 2021 lalu mengalami penurunan secara bertahap, dan mengalami peningkatan signifikan kembali pada awal tahun 2022 dan mengalami penurunan secara bertahap kembali. Hal tersebut juga terjadi pada rata-rata harian kasus Covid-19.

B. Hypotesis Testing

1. Gunakan T-Test Dan H0 = Rata-Rata Dari Kasus Harian Covid-19, H1 = Lebih Dari (>) Rata-Rata Dari Kasus Harian Covid-19

B. Hypotesis Testing

1. Gunakan T-Test Dan H0 = Rata-Rata Dari Kasus Harian Covid-19, H1 = Lebih Dari (>) Rata-Rata Dari Kasus Harian Covid-19

In [71]:
```
import numpy as np
from scipy.stats import norm

#melakukan perhitungan rata-rata "new cases"
AverageNewCases = DataCovid["New Cases"].mean()
print("Di Peroleh Nilai Rata-Rata Dari Kasus Harian Covid-19 Sebesar:", AverageNewCases)

#melakukan uji t dengan resampling
#melakukan perhitungan z-score dan p-value
MeanCases = DataCovid["New Cases"].mean()
StdCases = DataCovid["New Cases"].std()
Sample = len(DataCovid["New Cases"])

#distribusi normal sampel
NormalDist = np.random.normal(loc = MeanCases, scale = StdCases, size = (10000, Sample))

#perhitungan mean sample
MeanSample = NormalDist.mean(axis=1)

#melakukan perhitungan z-score
z_score = (MeanCases - MeanSample.mean()) / (MeanSample.std() / np.sqrt(Sample))

#melakukan perhitungan p-value
p_value = 2 * norm.cdf(-np.abs(z_score))

#menampilkan hasil
print("Di Peroleh Nilai Z-Score Sebesar:", z_score)
print("Di Peroleh Nilai P-Value Sebesar:", p_value)

#melakukan inisiasi alpha
alpha = 0.05

#melakukan inisiasi kondisi
if p_value < 0.05:
    print("H0 Di Tolak, T-Test Lebih Dari Rata-Rata Kasus Harian Covid-19")
else:
    print("H0 Gagal Di Tolak, T-Test Tidak Lebih Dari Rata-Rata Kasus Harian Covid-19")
```

```
Di Peroleh Nilai Rata-Rata Dari Kasus Harian Covid-19 Sebesar: 402.311388347684
Di Peroleh Nilai Z-Score Sebesar: 1.309481201840198
Di Peroleh Nilai P-Value Sebesar: 0.19037140129190233
H0 Gagal Di Tolak, T-Test Tidak Lebih Dari Rata-Rata Kasus Harian Covid-19
```

Pada kode script di atas, di lakukan proses perhitungan *hypothesis testing* secara manual menggunakan uji-t dengan bantuan *library* numpy yang di permisalkan sebagai np dan library scipy.stats dengan fungsi norm. Dengan kondisi H0 adalah sama dengan atau kurang dari rata-rata harian kasus Covid-19 dan H1 adalah lebih dari rata-rata harian kasus Covid-19. Di lakukan proses perhitungan rata-rata *"new cases"* dan di peroleh nilai sebesar 402.311. Kemudian di lakukan proses perhitungan *z-score* dari rata-rata dari kasus harian Covid-19 di Indonesia dan *p-value dari z-score.* Di peroleh nilai *z-score* sebesar 1.309 dan *p-value* sebesar 0,190.

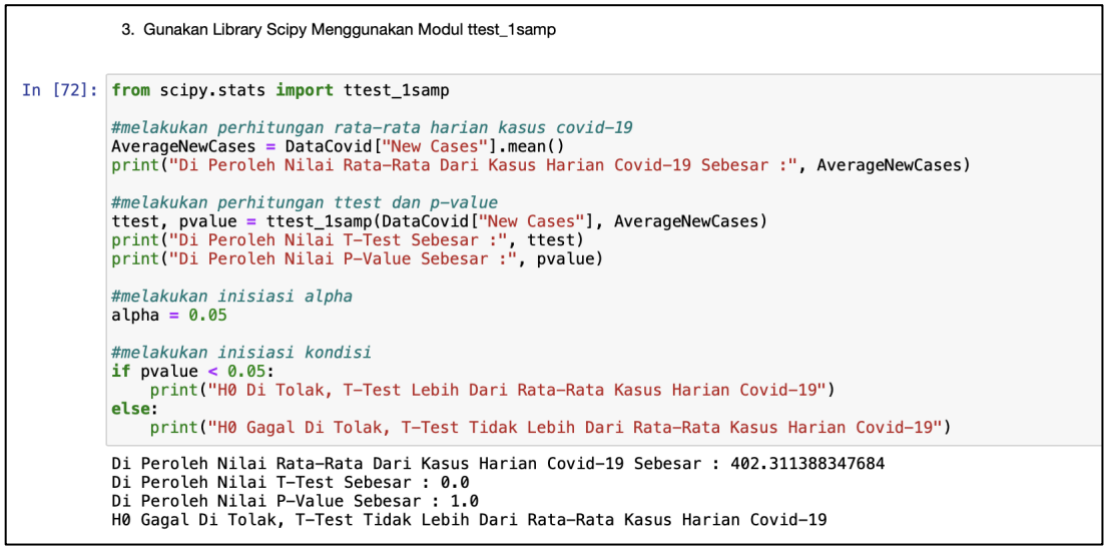
Di lakukan inisiasi alpha dan di buat dua kondisi jika, *p-value<alpha* maka H0 di tolal, dan jika *p-value>alpha* maka H0 gagal di tolak atau H0 di terima. Dan di tampilkan interpretasi bahwa H0 gagal di tolak atau H0 di terima di karenakan nilai *p-value* dari uji-t yang di lakukan tidak lebih dari rata-rata kasus harian Covid-19.

2. Simpulkan Apakah H0 DI Terima Atau Di Tolak

Berdasarkan proses perhitungan uji-t baik secara *stratch* di nomer 1 dan menggunakan modul ttest_1samp di nomer 3 dapat di perlihatkan bahwa nilai *p-value<alpha* maka dapat di interpretasikan bahwa H0 gagal di tolak atau H0 di terima.

Dengan kata lain, tidak terdapat bukti yang cukup mendukung bahwa rata-rata kasus harian Covid-19 di Indonesia lebih besar dari rata-rata populasi.

3. Gunakan Library Scipy Menggunakan Modul ttest_1samp



Pada kode script di atas, di lakukan proses uji-t dengan menggunakan modul scipy.stats dengan fungsi ttest_1samp. Di lakukan proses perhitungan nilai rata-rata dari kasus harian Covid-19 dan di peroleh nilai sebesar 402,311. Kemudian, dengan menggunakan fungsi ttest_1samp di lakukan proses perhitungaan *t-test* dan *p-value* di peroleh nilai *t-test* sebesar 0,0 dan nilai *p-value* sebesar 1,0.

Di lakukan inisiasi alpha dan di buat dua kondisi jika, *p-value<alpha* maka H0 di tolal, dan jika *p-value>alpha* maka H0 gagal di tolak atau H0 di terima. Dan di tampilkan interpretasi bahwa H0 gagal di tolak atau H0 di terima di karenakan nilai *p-value* dari uji-t yang di lakukan tidak lebih dari rata-rata kasus harian Covid-19.

C. Analisis Korelasi

1. Gunakan Pearson Untuk Mendapatkan Nilai Koefisien Korelasi, Implementasikan Ke Dalam Python Secara Stratch

C. Analisis Korelasi

1. Gunakan Pearson Untuk Mendapatkan Nilai Koefisien Korelasi, Implementasikan Ke Dalam Python Secara Stratch

In [142]:
```
import math

#mengubah tipe data DateCases menjadi numerik dan menghitung jumlahnya sejak awal periode
DateCases = DataCovid["Date"]
DataCovid["DateCasesNumeric"] = (DateCases - DateCases.min()).dt.days

#inisiasi variabel x dan y
x = DataCovid["DateCasesNumeric"]
y = DataCovid["New Cases"]

def KorelasiPearson(x, y):
    """
    variable x -> date cases
    variable y -> new cases

    """

    #array
    x2 = [xi ** 2 for xi in x]
    y2 = [yi ** 2 for yi in y]
    xy = [xi * yi for xi, yi in zip(x,y)]
    n = len(x)

    #hasil perhitungan looping for
    total_x = sum(x)
    total_y = sum(y)
    total_x2 = sum(x2)
    total_y2 = sum(y2)
    total_xy = sum(xy)

    #perhitungan koefisien korelasi
    r = (n * total_xy - total_x * total_y) / (math.sqrt((n * total_x2 - total_x ** 2) * (n * total_y2 - total_y
                                                                                         ** 2)))

    #penampilan nilai koefisien korelasi pearson
    print("Di Peroleh Nilai Koefisien Korelasi Pearson :", r)

KorelasiPearson(x, y)
```

```
Di Peroleh Nilai Koefisien Korelasi Pearson : 0.03918944827301286
```

Pada kode script di atas, di lakukan proses perhitungan korelasi *pearson* dengan menggunakan *library* math. Setelah sebelumnya di lakukan pengubahan tipe data di kolom *"Date"* menjadi datetime, selanjutnya di kode ini di lakukan proses pembuatan kolom dengan nama *"DateCaseNumeric"* dengan melakukan konversi data dari kolom *"Date"* dan di lakukan perhitungan jumlah hari sejak awal periode kasus. Di lakukan inisiasi variabel x adalah *"DateCasesNumeric"* dan variabel y adalah *"New Cases"*. Di buat *define function* KorelasiPearson, dengan langkah-langkah perhitungan kuadrat dari data x dan y yang di simpan dalam *array* x2 dan y2, di lakukan perhitungan xy yang di simpan dalam *array* xy, dan di lakukan perhitungan jumlah panjang data yang di simpan dalam variabel n.

Selanjutnya, di lakukan total dari perhitungan dalam masing-masing array tersebut yang menggunakan proses *looping for*, dengan nilai total_x yang merupakan jumlah dari variabel x, total_y yang merupakan jumlah dari variabel y, total_x2 yang merupakan jumlah dari array x2, total_y2 yang merupakan jumlah dari array y2, dan total_xy yang merupakan jumlah dari variabel x*y. Di lakukan proses perhitungan koefisien korelasi *pearson* yang di simpan dalam variabel r dengan rumus perhitungan r = (n * total_xy - total_x * total_y) / (math.sqrt((n * total_x2 - total_x ** 2) * (n * total_y2 - total_y ** 2))). Maka, akan di tampilakn nilai koefisien korelasi *pearson* sebesar 0,03918944827301286 atau dapat di bulatkan menjadi 0,0392.

D. Regresi Dan Evaluasi

1. **Cari Model Regresi Yang Tepat Untuk Menyelesaikan Data Harian Covid-19 Di Indonesia**

D. Regresi Dan Evaluasi

1. Cari Model Regresi Yang Tepat Untuk Menyelesaikan Data Harian Covid-19 Di Indonesia

In [168]:
```
DateCases = DataCovid["Date"]
DataCovid["DateCasesNumeric"] = (DateCases - DateCases.min()).dt.days

#inisiasi variabel x dan y
x = DataCovid["DateCasesNumeric"]
y = DataCovid["New Cases"]

def Regression(x, y):

    #array dan perhitungannya
    x2 = [xi ** 2 for xi in x]
    y2 = [yi ** 2 for yi in y]
    xy = [xi * yi for xi, yi in zip(x, y)]

    #total perhitungan dari array
    total_x = sum(x)
    total_y = sum(y)
    total_x2 = sum(x2)
    total_y2 = sum(y2)
    total_xy = sum(xy)

    #perhitungan nilai a dan b
    a = ((total_y * total_x2) - (total_x * total_xy)) / ((len(x) * total_x2) - (total_x ** 2))
    b = ((len(x) * total_xy) - (total_x * total_y)) / ((len(x) * total_x2) - (total_x ** 2))

    #penampilan nilai a dan b
    print("Di Peroleh Nilai A :", a)
    print("Di Peroleh Nilai B :", b)

    #perhitungan y_predict
    y_predict = [a + b * xi for xi in x]

    #penampilan nilai y_predict
    print("Di Peroleh Nilai y_predict :", y_predict)

Regression(x, y)
```

```
Di Peroleh Nilai A : 238.19841608750625
Di Peroleh Nilai B : 0.346365702603417
Di Peroleh Nilai y_predict : [238.19841608750625, 238.54478179010965, 238.54478179010965, 238.54478179010965, 238.
8911474927131, 238.8911474927131, 238.8911474927131, 238.8911474927131, 239.2375131953165, 239.2375131953165, 239.
2375131953165, 239.2375131953165, 239.5838788979199, 239.5838788979199, 239.5838788979199, 239.5838788979199, 239.
93024460052334, 239.93024460052334, 239.93024460052334, 239.93024460052334, 239.93024460052334, 240.2766103031267
4, 240.27661030312674, 240.27661030312674, 240.27661030312674, 240.27661030312674, 240.62297600573015, 240.6229760
0573015, 240.62297600573015, 240.62297600573015, 240.62297600573015, 240.62297600573015, 240.9693417083336, 240.96
93417083336, 240.9693417083336, 240.9693417083336, 240.9693417083336, 240.9693417083336, 241.315707410937, 241.315
```

Pada kode script di atas, di lakukan proses perhitungan regresi menggunakan model regresi linier untuk menyelesaikan kasus data harian Covid-19 di Indonesia. Setelah sebelumnya di lakukan pengubahan tipe data di kolom *"Date"* menjadi datetime, selanjutnya di kode ini di lakukan proses pembuatan kolom dengan nama *"DateCaseNumeric"* dengan melakukan konversi data dari kolom *"Date"* dan di lakukan perhitungan jumlah hari sejak awal periode kasus Di lakukan inisiasi variabel x adalah *"DateCasesNumeric"* dan variabel y adalah *"New Cases"*. Di buat *define function* Regression dengan parameter x dan y.

Dengan langkah-langkah perhitungan kuadrat dari data x dan y yang di simpan dalam *array* x2 dan y2, di lakukan perhitungan xy yang di simpan dalam *array* xy, dan di lakukan perhitungan jumlah panjang data yang di simpan dalam variabel n. Selanjutnya, di lakukan total dari perhitungan dalam masing-masing array tersebut yang menggunakan proses *looping for*, dengan nilai total_x yang merupakan jumlah dari variabel x, total_y yang merupakan jumlah dari variabel y, total_x2 yang

merupakan jumlah dari array x2, total_y2 yang merupakan jumlah dari array y2, dan total_xy yang merupakan jumlah dari variabel x*y.

Kemudian, di lakukan proses perhitungan nilai a dan b, dengan rumus nilai a menggunakan a = ((total_y * total_x2) - (total_x * total_xy)) / ((len(x) * total_x2) - (total_x ** 2)) dan nilai b menggunakan b = ((len(x) * total_xy) - (total_x * total_y)) / ((len(x) * total_x2) - (total_x ** 2)). Di peroleh nilai a sebesar 238,198 dan nilai b sebesar 0,346. Setelah mendapatkan nilai a dan b, selanjutnya di lakukan proses perhitungan y_predict dengan rumus y_predict = [a + b * xi for xi in x]. Maka, akan di tampilkan nilai y_predict dengan nilai a sebesar 238,198 di tambah nilai b sebesar 0,346 di kali dengan nilai x sebesar 0 (di peroleh dari data indeks ke-0 dari kolom *date*) sehingga, di peroleh nilai y_predict sebesar 238,198.

2. Evaluasi Model Menggunakan MAPE

2. Evaluasi Model Regresi Menggunakan MAPE

In [175]:
```
DateCases = DataCovid["Date"]
DataCovid["DateCasesNumeric"] = (DateCases - DateCases.min()).dt.days

#inisiasi variabel x dan y
x = DataCovid["DateCasesNumeric"]
y = DataCovid["New Cases"]

def Regression(x, y):

    #array dan perhitungannya
    x2 = [xi ** 2 for xi in x]
    y2 = [yi ** 2 for yi in y]
    xy = [xi * yi for xi, yi in zip(x, y)]

    #total perhitungan dari array
    total_x = sum(x)
    total_y = sum(y)
    total_x2 = sum(x2)
    total_y2 = sum(y2)
    total_xy = sum(xy)

    #perhitungan nilai a dan b
    a = ((total_y * total_x2) - (total_x * total_xy)) / ((len(x) * total_x2) - (total_x ** 2))
    b = ((len(x) * total_xy) - (total_x * total_y)) / ((len(x) * total_x2) - (total_x ** 2))

    #perhitungan y_predict
    y_predict = [a + b * xi for xi in x]

    #perhitungan MAPE
    mape = []
    for i in range(len(x)):
        mape.append((abs(y[i] - y_predict[i]) / y_predict[i]) * 100)
    value_mape = sum(mape) / len(x)
    print("Di Peroleh Nilai MAPE Sebesar : {:.5f}%".format(value_mape))

Regression(x, y)
```

```
Di Peroleh Nilai MAPE Sebesar : 147.50375%
```

Pada kode script di atas, di lakukan proses perhitungan MAPE *(Mean Absolute Percentage Error).* Setelah sebelumnya di lakukan pengubahan tipe data di kolom *"Date"* menjadi datetime, selanjutnya di kode ini di lakukan proses pembuatan kolom dengan nama *"DateCaseNumeric"* dengan melakukan konversi data dari kolom *"Date"* dan di lakukan perhitungan jumlah hari sejak awal periode kasus Di lakukan inisiasi variabel x adalah *"DateCasesNumeric"* dan variabel y adalah *"New Cases".* Di buat *define function* Regression dengan parameter x dan y.

Setelah melakukan proses model regresi di nomer sebelumnya, dan telah mendapatkan nilai a, b, dan y_predict langkah selanjutnya adalah melakukan proses evaluasi model menggunakan MAPE dengan langkah pembuatan array dan melakukan

perhitungan *looping for* untuk setiap titik data dengan rumus y |(y – y_predict) / y_predict| * 100 dan di lanjutkan dengan perhitungan rata-rata MAPE yang di tuliskan dalam variabel value_mape. Maka, akan di tampilkan nilai hasil MAPE sebesar 147,50375%.

3. **Implementasikan Model Regresi Dan Evaluasi Menggunakan Python Secara Stratch**

3. Implementasikan Model Regresi Dan Evaluasi Menggunakan Python Secara Stratch

In [178]:
```
DateCases = DataCovid["Date"]
DataCovid["DateCasesNumeric"] = (DateCases - DateCases.min()).dt.days

#inisiasi variabel x dan y
x = DataCovid["DateCasesNumeric"]
y = DataCovid["New Cases"]

def Regression(x, y):

    #array dan perhitungannya
    x2 = [xi ** 2 for xi in x]
    y2 = [yi ** 2 for yi in y]
    xy = [xi * yi for xi, yi in zip(x, y)]

    #total perhitungan dari array
    total_x = sum(x)
    total_y = sum(y)
    total_x2 = sum(x2)
    total_y2 = sum(y2)
    total_xy = sum(xy)

    #perhitungan nilai a dan b
    a = ((total_y * total_x2) - (total_x * total_xy)) / ((len(x) * total_x2) - (total_x ** 2))
    b = ((len(x) * total_xy) - (total_x * total_y)) / ((len(x) * total_x2) - (total_x ** 2))

    #penampilan nilai a dan b
    print("Di Peroleh Nilai A :", a)
    print("Di Peroleh Nilai B :", b)

    #perhitungan y_predict
    y_predict = [a + b * xi for xi in x]

    #penampilan nilai y_predict
    print("Di Peroleh Nilai y_predict:", y_predict)

    #perhitungan MAPE
    mape = []
    for i in range(len(x)):
        mape.append((abs(y[i] - y_predict[i]) / y_predict[i]) * 100)
    value_mape = sum(mape) / len(x)
    print("Di Peroleh Nilai MAPE Sebesar : {:.5f}%".format(value_mape))

Regression(x, y)
```

```
Di Peroleh Nilai A : 238.19841608750625
Di Peroleh Nilai B : 0.346365702603417
Di Peroleh Nilai y_predict: [238.19841608750625, 238.54478179010965, 238.54478179010965, 238.54478179010965,
238.8911474927131, 238.8911474927131, 238.8911474927131, 238.8911474927131, 239.2375131953165, 239.2375131953
165, 239.2375131953165, 239.2375131953165, 239.5838788979199, 239.5838788979199, 239.5838788979199, 239.58387
88979199, 239.93024460052334, 239.93024460052334, 239.93024460052334, 239.93024460052334, 239.93024460052334,
240.27661030312674, 240.27661030312674, 240.27661030312674, 240.27661030312674, 240.27661030312674, 240.62297
600573015, 240.62297600573015, 240.62297600573015, 240.62297600573015, 240.62297600573015, 240.6229760057301
5, 240.9693417083336, 240.9693417083336, 240.9693417083336, 240.9693417083336, 240.9693417083336, 240.9693417
083336, 241.315707410937, 241.315707410937, 241.315707410937, 241.315707410937, 241.315707410937, 241.3157074
```

```
24008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.
2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737,
559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008
737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.2794224008737, 559.27942
24008737, 559.2794224008737, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.
6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771,
559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034
771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.62578
81034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.
6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771, 559.6257881034771,
559.9721538060805]
Di Peroleh Nilai MAPE Sebesar : 147.50375%
```

Pada kode script di atas di lakukan proses pencarian model regresi menggunkan *simple linier regression* hal ini di karenakan hanya terdapat satu variabel x dan y yakni, *"Date"* sebagai variabel x dan *"New Cases"* sebagai variabel y. Dalam proses pembuatan model regresi, di lakukan pencarian nilai a, b, dan y_predict. Di peroleh

nilai a sebesar 238,198, nilai b sebesar 0,346, dan nilai y_predict untuk indeks data ke-0 sebesar 238,198. Setelah melakukan proses perhitungan model regresi langkah selanjutnya di lakukan proses evaluasi model regresi dengan menggunakan MAPE *(Mean Absolute Percentage Error)* dan di peroleh nilai MAPE sebesar 147,50375% . Nilai ini menunjukkan angka yang cukup tinggi dan menunjukkan bahwa model regresi yang telah di tuliskan memiliki tingkat kesalahan yang cukup tinggi.