Assignment 1 By Brian .T. Mutsetsa (N02016615R)

Interest: What were the most dangerous states in the US between 2018 & 2022?

Step 1: **SET UP AMERICAN BOUNDARY TO KNOW THE SCOPE OF THE ANALYSIS**

I used the following to achieve obtaining the data:

- *Googlemaps Places API*: To acquire the location data of the United states.
- *Pandas and other Python Libraries*: To normalise the data collected from Googlemaps, and present it in a way conducive to the user.
- *Folium & Geopandas*: To make a sketch of the map of the United STates and shos the states and boundaries between them.

In [27]:
```
# Step 1: Setting up Google Maps API Account
# Follow the instructions provided by Google to set up the API keys.

# Import necessary libraries
import requests

# Define API key and endpoint
api_key = "AIzaSyBhGfz9n4wcxnU2ySX648PGs9POiTjo6AU"  # Replace with your API key
endpoint = "https://maps.googleapis.com/maps/api/place/textsearch/json"

# Make GET request to search for the United States
params = {
    "query": "United States",
    "key": api_key
}
response = requests.get(endpoint, params=params)
cities_data = response.json()
print(cities_data)
```

```
{'html_attributions': [], 'results': [{'formatted_address': 'United States', 'geo
metry': {'location': {'lat': 37.09024, 'lng': -95.712891}, 'viewport': {'northeas
t': {'lat': 72.7087158, 'lng': -66.3193754}, 'southwest': {'lat': 15.7760139, 'ln
g': -173.2992296}}}, 'icon': 'https://maps.gstatic.com/mapfiles/place_api/icons/v
1/png_71/geocode-71.png', 'icon_background_color': '#7B9EB0', 'icon_mask_base_ur
i': 'https://maps.gstatic.com/mapfiles/place_api/icons/v2/generic_pinlet', 'nam
e': 'United States', 'photos': [{'height': 1536, 'html_attributions': ['<a href
="https://maps.google.com/maps/contrib/103996003334839676331">neodly</a>'], 'phot
o_reference': 'ATplDJayZxBGDy3d4ImskHv5ByBKXRuvu-IHlE7QNB6A8SFarhipiK6rckeLtjYD3w
VNq_-kQfSp7t9JXOlhD5hrA71qyxP9y2Qks921QExrLXhYcbK5wsN1BWNQ2UQrla3TqRYa4qpOOtwLA_p
brmROACMJYOpM6YcCGmUe5u7-qCHB_d6W', 'width': 2048}], 'place_id': 'ChIJCzYy5IS16lQ
RQrfeQ5K5Oxw', 'reference': 'ChIJCzYy5IS16lQRQrfeQ5K5Oxw', 'types': ['country',
'political']}], 'status': 'OK'}
```

Data comes in as a Json file which isn't readable, so we normalise it, sperating it via the 'json_normalize()' function

In [28]:
```
# Step 3: Data Normalization
import pandas as pd

# Convert JSON data to DataFrame
cities_df = pd.json_normalize(cities_data['results'])

# Save the DataFrame as a CSV named "us_crimes_fact_table.csv"
cities_df.to_csv('us_crime_fact_table.csv', index=False)

# Display the first few rows of the DataFrame
cities_df.head()
```

Out[28]:

	formatted_address	icon	icon_background_colo
0	United States	https://maps.gstatic.com/mapfiles/place_api/ic...	#7B9EB

Once we've normalized the location data, we can begin to get the necessary coordinates to plot a *map of the United States* and show it's boundaries.

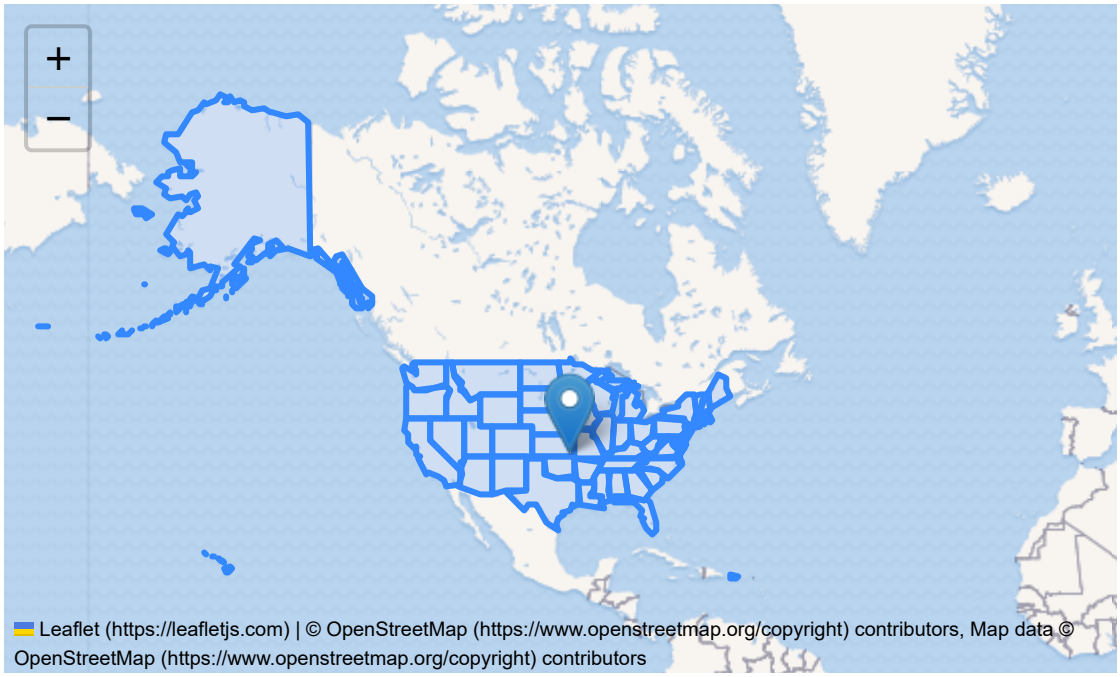
Please note: To show the boundaries between states, I used the us-states.json which I got from https://github.com/PublicaMundi/MappingAPI/blob/master/data/geojson/us-states.json

In [29]:
```
# Step 4: Data Visualization with Geopandas and Folium
import geopandas as gpd
import csv
import matplotlib.pyplot as plt
import folium
import chardet

# Save city data to a CSV file
with open('us_states.csv', 'w', newline='') as csvfile:
    fieldnames = ['name', 'latitude', 'longitude']
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    for state in cities_data['results']:
        writer.writerow({'name': state['name'], 'latitude': state['geometry']['l

# Determine the encoding of the file
with open('us_states.csv', 'rb') as f:
    result = chardet.detect(f.read())

print(result['encoding'])

# Read city data from CSV file with specified encoding
us_states_df = pd.read_csv('us_states.csv', encoding=result['encoding'])

# Create the map centered on the USA
usa_map = folium.Map(location=[37.0902, -95.7129], zoom_start=4)
```

```
# Add OpenStreetMap (Wikimedia) tile layer with English labels and attribution
folium.TileLayer('https://maps.wikimedia.org/osm-intl/{z}/{x}/{y}{r}.png', attr=

# Load the boundary GeoJSON layer for USA states (if available)
# Replace 'us-states.json' with the file path to the GeoJSON file containing USA
usa_boundary = 'us-states.json'
folium.GeoJson(usa_boundary, name='USA States').add_to(usa_map)

# Add markers for state locations
for idx, state in us_states_df.iterrows():
    folium.Marker([state['latitude'], state['longitude']], popup=state['name']).

# Display the map
usa_map
```

ascii

Out[29]:



Step 2: **USING ANCILLIARY DATA**

The following steps were used to obtain and use the ancilliary data:

1) *Collect Crime Rate Data from any reputable USA government website*: In this case I got it from the ***Federal Bureau of Investigation's Crime Data Explorer***.

- The dataset is called **estimated_crimes_1979_2022.csv**
- This dataset contains estimated data at the state and national level and was derived from the Summary Reporting System (SRS).
- These data reflect the estimates the FBI has traditionally included in its annual publications.
- This estimated data is from 1979 to 2022.
- Link: https://cde.ucr.cjis.gov/LATEST/webapp/#/pages/downloads

2) *Clean the dataset*:

- The dataset has the number of crimes committed by category
- Some columns like 'rape_legacy' and 'rape_revised' were were empty or filled sometimes, so I kept them in and the Nan values would be added as zeros
- the 'caveats' column didnt have any information necessary to the analysis, as all it held was comments, and not digits.

In [30]:
```
# Read the data from the CSV file
crime_data = pd.read_csv('estimated_crimes_1979_2022.csv')

# Clean the data by removing the "rape_revised" and "caveats" column
crime_data = crime_data.drop(columns=['caveats'])

# Create a new DataFrame with the required columns
required_columns = ['year', 'state_abbr', 'state_name', 'population', 'violent_c
                    'robbery', 'aggravated_assault', 'property_crime', 'burglary
crime_data = crime_data[required_columns]

crime_data_2018 = crime_data[(crime_data['year'] == 2018)]
crime_data_2019 = crime_data[(crime_data['year'] == 2019)]
crime_data_2020 = crime_data[(crime_data['year'] == 2020)]
crime_data_2021 = crime_data[(crime_data['year'] == 2021)]
crime_data_2022 = crime_data[(crime_data['year'] == 2022)]
```

3) *Seperate data year by year between the 2018 and 2022 and further clean it*:

- Some of the years came with an additional column showing the total crimes by category of the United states, the the individual steps followed.
- Upon analysis, I found that some of the additions were incorrect, and thus unreliable so I removed the totals rows and calculated for myself for each crime category.
- To make this process easier, I seperated the dataset by year, removed the totals rown and did the calculations on my own.
- This also prevented unecessary skew for the bar chart visualizations used.

4) *Simplifying the datase and obtaining information necessary to rank the states*:

- After seperating each dataset and cleaning it further, I summed the individual crime rates and produced the **'total crime rate'**.
- This would be shown by state, to show which states were the most dangerous within their respective years.
- This also made the datase simpler in terms of comprehension and ease of use

2018

In [31]:
```
print("\n ------------------------------------2018-----------------------------
crime_data_2018.head()
```

```
------------------------------------2018-----------------------------------
```

Out[31]:

	year	state_abbr	state_name	population	violent_crime	homicide	rape_legacy
2027	2018	AK	Alaska	735139	6555	47	NaN
2028	2018	AL	Alabama	4887681	25567	383	NaN
2029	2018	AR	Arkansas	3009733	16904	222	NaN
2030	2018	AZ	Arizona	7158024	34053	383	NaN
2031	2018	CA	California	39461588	176604	1739	NaN

In [32]:

```
print("\n 2018 Simplified \n")
# Select the columns to sum
columns_to_sum = crime_data_2018.iloc[:, 4:14]

# Calculate the total across each row
total_crime_rate = columns_to_sum.sum(axis=1)

# Create a copy of the slice
crime_data_2018_sim = crime_data_2018.copy()

# Add the total crime rate column to the new DataFrame
crime_data_2018_sim['total_crime_rate'] = total_crime_rate

required_columns = ['year', 'state_abbr', 'state_name', 'total_crime_rate']

crime_data_2018_sim = crime_data_2018_sim[required_columns]

crime_data_2018_sim.head()
```

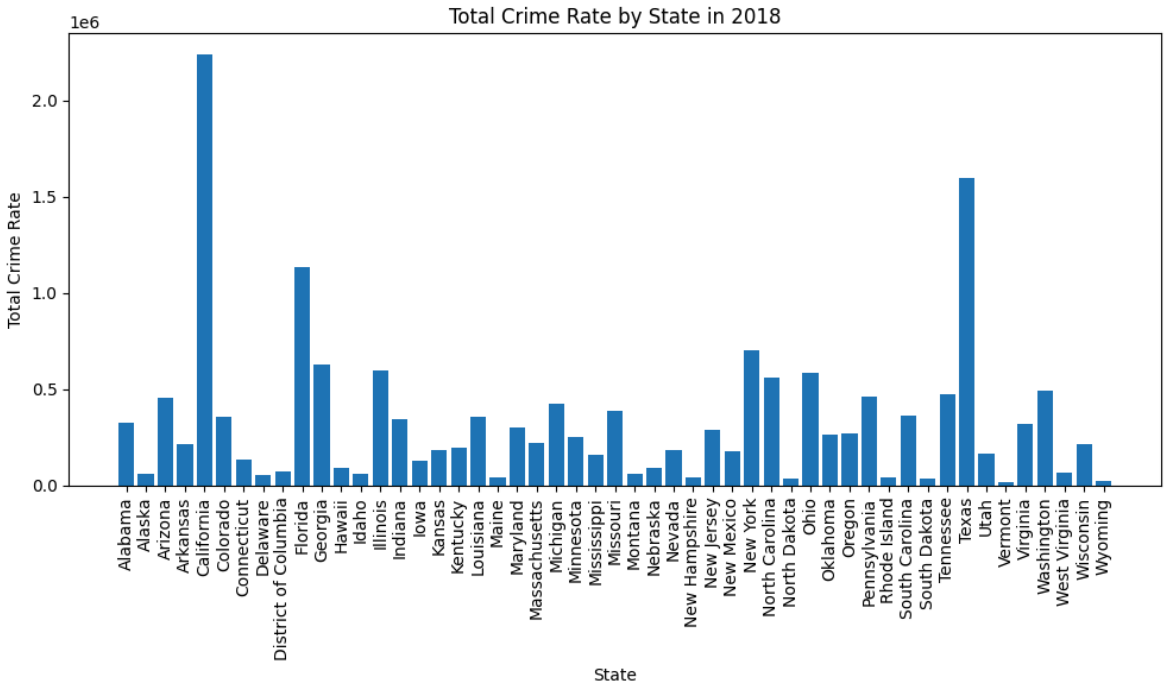
```
2018 Simplified
```

Out[32]:

	year	state_abbr	state_name	total_crime_rate
2027	2018	AK	Alaska	61800.0
2028	2018	AL	Alabama	328538.0
2029	2018	AR	Arkansas	212162.0
2030	2018	AZ	Arizona	453566.0
2031	2018	CA	California	2236496.0

In [33]:

```
# Sort the data by state name
crime_data_2018_sim_sorted = crime_data_2018_sim.sort_values(by='state_name', as

# Plot the bar chart
plt.figure(figsize=(10, 6))
plt.bar(crime_data_2018_sim_sorted['state_name'], crime_data_2018_sim_sorted['to
plt.xlabel('State')
plt.ylabel('Total Crime Rate')
plt.title('Total Crime Rate by State in 2018')
plt.xticks(rotation=90)  # Rotate state names for better readability
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```



2019

In [34]:
```
print("\n ------------------------------------2019-----------------------------
crime_data_2019.head()
```

```
------------------------------------2019-----------------------------------
```

Out[34]:

	year	state_abbr	state_name	population	violent_crime	homicide	rape_legacy
2078	2019	AK	Alaska	733603	6346	69	NaN
2079	2019	AL	Alabama	4907965	24769	390	NaN
2080	2019	AR	Arkansas	3020985	17547	237	NaN
2081	2019	AZ	Arizona	7291843	32603	397	NaN
2082	2019	CA	California	39437610	174341	1690	NaN

In [35]:
```
print("\n 2019 Simplified \n")
# Select the columns to sum
columns_to_sum = crime_data_2019.iloc[:, 4:14]

# Calculate the total across each row
total_crime_rate = columns_to_sum.sum(axis=1)

# Create a copy of the slice
crime_data_2019_sim = crime_data_2019.copy()

# Add the total crime rate column to the new DataFrame
crime_data_2019_sim['total_crime_rate'] = total_crime_rate

required_columns = ['year', 'state_abbr', 'state_name', 'total_crime_rate']

crime_data_2019_sim = crime_data_2019_sim[required_columns]
```

```
crime_data_2019_sim.head()
```

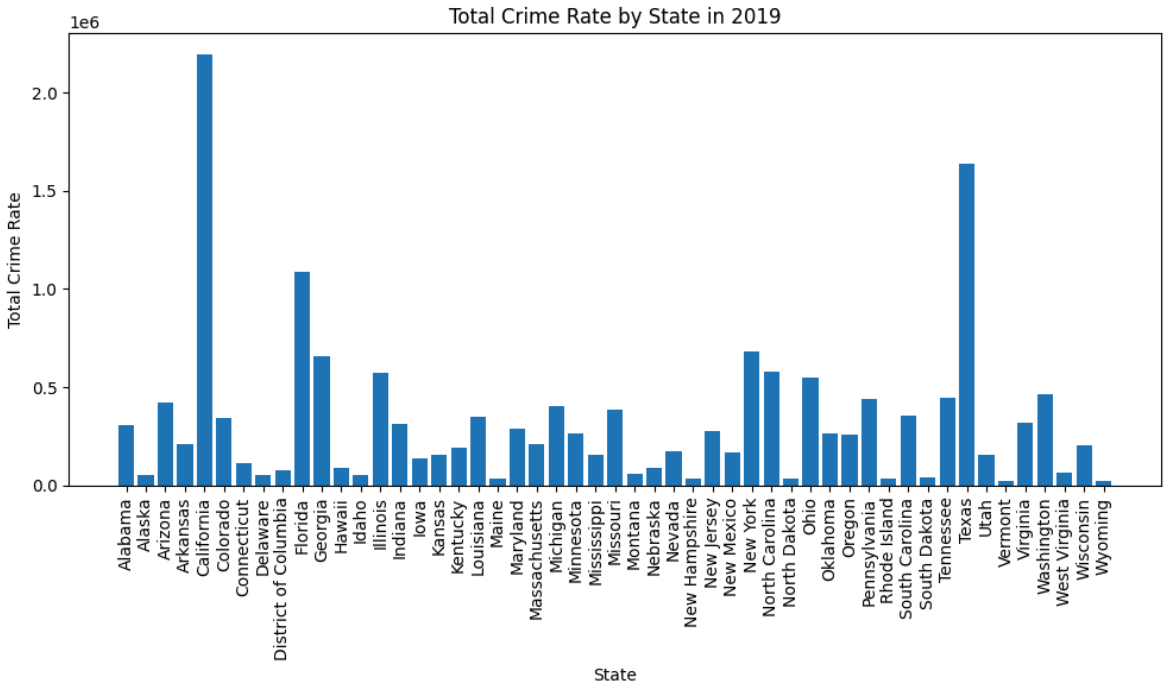
2019 Simplified

Out[35]:

	year	state_abbr	state_name	total_crime_rate
2078	2019	AK	Alaska	55278.0
2079	2019	AL	Alabama	306990.0
2080	2019	AR	Arkansas	207152.0
2081	2019	AZ	Arizona	420462.0
2082	2019	CA	California	2191036.0

In [36]:

```
# Sort the data by state name
crime_data_2019_sim_sorted = crime_data_2019_sim.sort_values(by='state_name', as

# Plot the bar chart
plt.figure(figsize=(10, 6))
plt.bar(crime_data_2019_sim_sorted['state_name'], crime_data_2019_sim_sorted['to
plt.xlabel('State')
plt.ylabel('Total Crime Rate')
plt.title('Total Crime Rate by State in 2019')
plt.xticks(rotation=90)  # Rotate state names for better readability
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```



2020

In [37]:

```
print("\n ------------------------------------2020-----------------------------
crime_data_2020.head()
```

------------------------------------2020-----------------------------------

Out[37]:

	year	state_abbr	state_name	population	violent_crime	homicide	rape_legacy
2129	2020	AK	Alaska	731158	6126	49	NaN
2130	2020	AL	Alabama	4921532	22322	471	NaN
2131	2020	AR	Arkansas	3030522	20363	321	NaN
2132	2020	AZ	Arizona	7421401	35980	513	NaN
2133	2020	CA	California	39368078	174026	2203	NaN

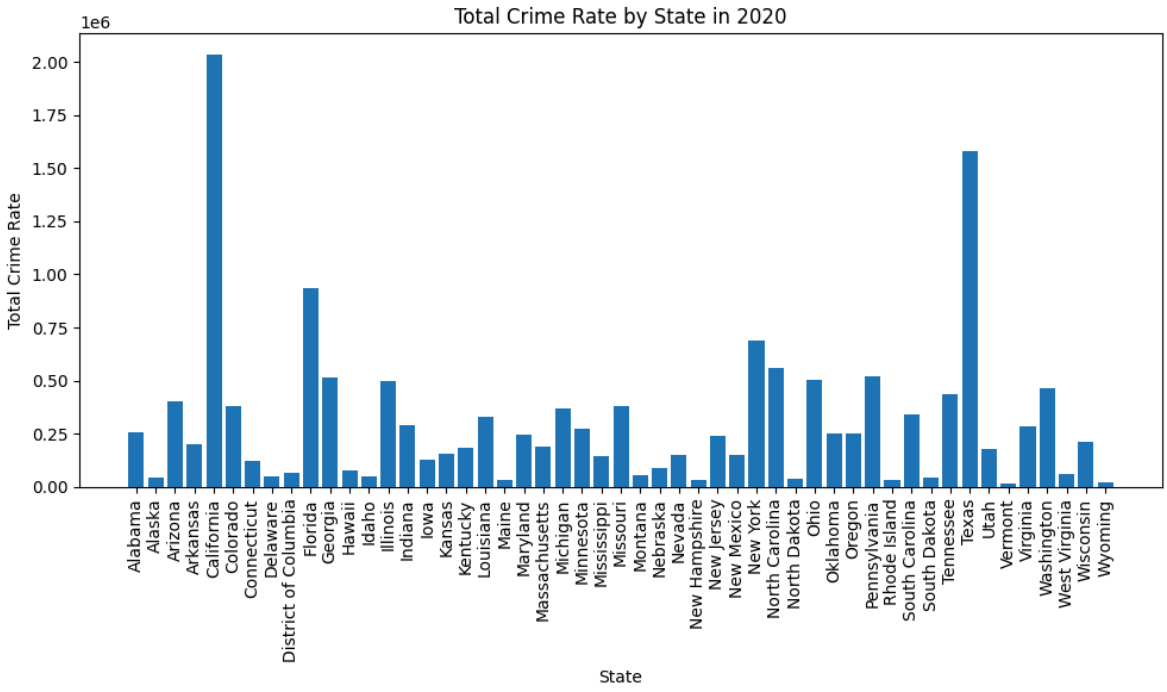
In [38]:
```
print("\n 2020 Simplified \n")
# Select the columns to sum
columns_to_sum = crime_data_2020.iloc[:, 4:14]

# Calculate the total across each row
total_crime_rate = columns_to_sum.sum(axis=1)

# Create a copy of the slice
crime_data_2020_sim = crime_data_2020.copy()

# Add the total crime rate column to the new DataFrame
crime_data_2020_sim['total_crime_rate'] = total_crime_rate

required_columns = ['year', 'state_abbr', 'state_name', 'total_crime_rate']

crime_data_2020_sim = crime_data_2020_sim[required_columns]

crime_data_2020_sim.head()
```

```
2020 Simplified
```

Out[38]:

	year	state_abbr	state_name	total_crime_rate
2129	2020	AK	Alaska	45308.0
2130	2020	AL	Alabama	254966.0
2131	2020	AR	Arkansas	199126.0
2132	2020	AZ	Arizona	402606.0
2133	2020	CA	California	2032160.0

In [39]:
```
# Sort the data by state name
crime_data_2020_sim_sorted = crime_data_2020_sim.sort_values(by='state_name', as

# Plot the bar chart
plt.figure(figsize=(10, 6))
plt.bar(crime_data_2020_sim_sorted['state_name'], crime_data_2020_sim_sorted['to
plt.xlabel('State')
plt.ylabel('Total Crime Rate')
plt.title('Total Crime Rate by State in 2020')
plt.xticks(rotation=90)  # Rotate state names for better readability
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```



2021

In [40]:
```
print("\n ------------------------------------2021-----------------------------
# Remove the first row from the DataFrame
crime_data_2021 = crime_data_2021.iloc[1:]
crime_data_2021.head()
```

```
------------------------------------2021-----------------------------------
```

Out[40]:

	year	state_abbr	state_name	population	violent_crime	homicide	rape_legacy
2181	2021	AK	Alaska	734182	5573	45	NaN
2182	2021	AL	Alabama	5049846	17590	476	NaN
2183	2021	AR	Arkansas	3028122	21271	334	NaN
2184	2021	AZ	Arizona	7264877	30922	485	NaN
2185	2021	CA	California	39142991	188343	2346	NaN

In [41]:
```
print("\n 2021 Simplified \n")
# Select the columns to sum
columns_to_sum = crime_data_2021.iloc[:, 4:14]

# Calculate the total across each row
total_crime_rate = columns_to_sum.sum(axis=1)

# Create a copy of the slice
crime_data_2021_sim = crime_data_2021.copy()

# Add the total crime rate column to the new DataFrame
crime_data_2021_sim['total_crime_rate'] = total_crime_rate

required_columns = ['year', 'state_abbr', 'state_name', 'total_crime_rate']
```

```
crime_data_2021_sim = crime_data_2021_sim[required_columns]

crime_data_2021_sim.head()
```

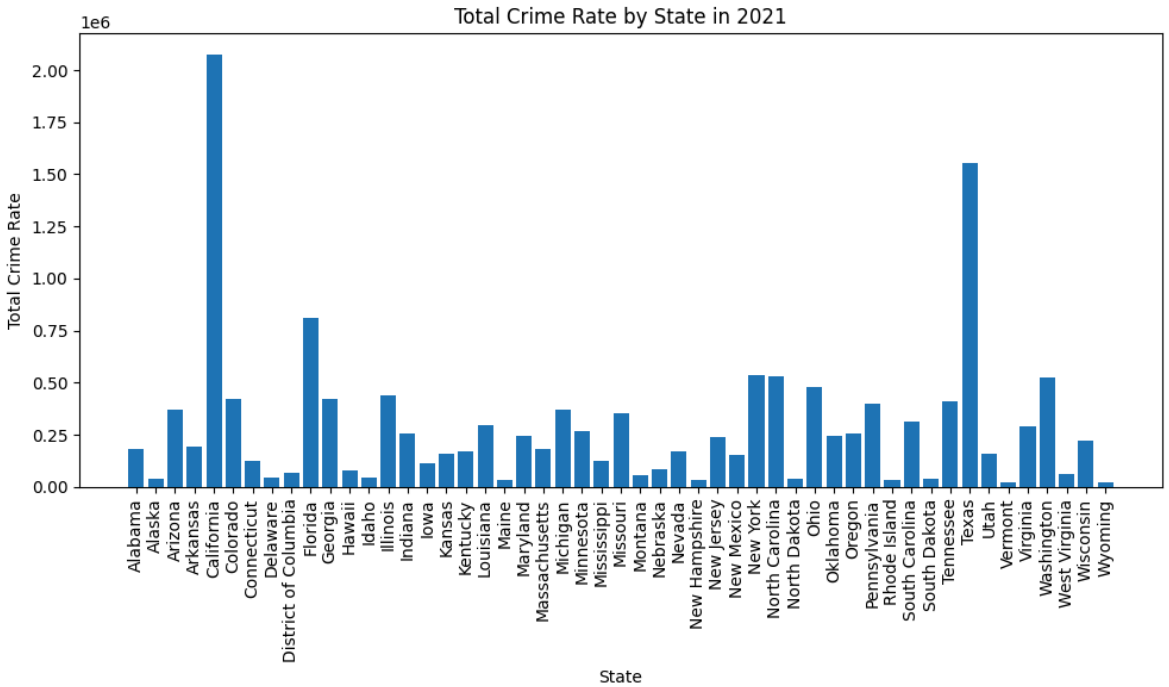
2021 Simplified

Out[41]:

	year	state_abbr	state_name	total_crime_rate
2181	2021	AK	Alaska	38058.0
2182	2021	AL	Alabama	183722.0
2183	2021	AR	Arkansas	195702.0
2184	2021	AZ	Arizona	369126.0
2185	2021	CA	California	2071820.0

In [42]:

```
# Sort the data by state name
crime_data_2021_sim_sorted = crime_data_2021_sim.sort_values(by='state_name', as

# Plot the bar chart
plt.figure(figsize=(10, 6))
plt.bar(crime_data_2021_sim_sorted['state_name'], crime_data_2021_sim_sorted['to
plt.xlabel('State')
plt.ylabel('Total Crime Rate')
plt.title('Total Crime Rate by State in 2021')
plt.xticks(rotation=90)  # Rotate state names for better readability
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```



2022

In [43]:

```
print("\n ------------------------------------2022-----------------------------
# Remove the first row from the DataFrame
crime_data_2022 = crime_data_2022.iloc[1:]
crime_data_2022.head()
```

```
-------------------------------------2022----------------------------------
```

Out[43]:

	year	state_abbr	state_name	population	violent_crime	homicide	rape_legacy
2233	2022	AK	Alaska	733583	5567	70	NaN
2234	2022	AL	Alabama	5074296	20759	552	NaN
2235	2022	AR	Arkansas	3045637	19654	312	NaN
2236	2022	AZ	Arizona	7359197	31754	500	NaN
2237	2022	CA	California	39029342	194935	2231	NaN

In [44]:

```
print("\n 2022 Simplified \n")
# Select the columns to sum
columns_to_sum = crime_data_2022.iloc[:, 4:14]

# Calculate the total across each row
total_crime_rate = columns_to_sum.sum(axis=1)

# Create a copy of the slice
crime_data_2022_sim = crime_data_2022.copy()

# Add the total crime rate column to the new DataFrame
crime_data_2022_sim['total_crime_rate'] = total_crime_rate

required_columns = ['year', 'state_abbr', 'state_name', 'total_crime_rate']

crime_data_2022_sim = crime_data_2022_sim[required_columns]

crime_data_2022_sim.head()
```

```
2022 Simplified
```

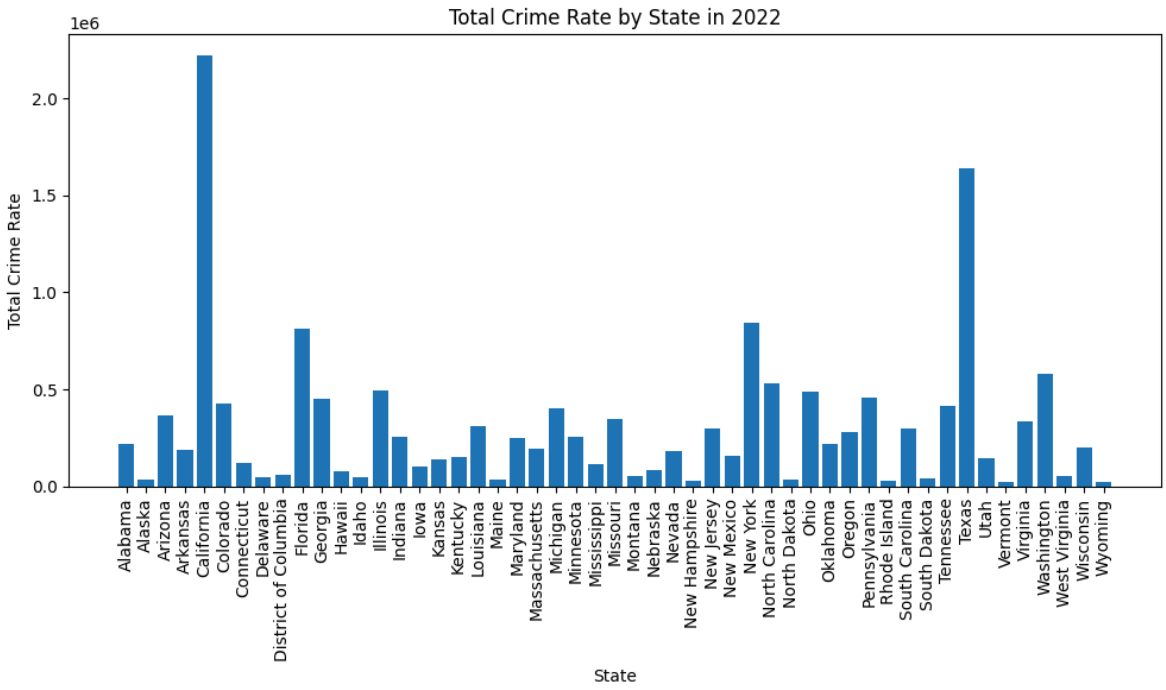
Out[44]:

	year	state_abbr	state_name	total_crime_rate
2233	2022	AK	Alaska	37382.0
2234	2022	AL	Alabama	217998.0
2235	2022	AR	Arkansas	188636.0
2236	2022	AZ	Arizona	366350.0
2237	2022	CA	California	2218904.0

In [45]:

```
# Sort the data by state name
crime_data_2022_sim_sorted = crime_data_2022_sim.sort_values(by='state_name', as

# Plot the bar chart
plt.figure(figsize=(10, 6))
plt.bar(crime_data_2022_sim_sorted['state_name'], crime_data_2022_sim_sorted['to
plt.xlabel('State')
plt.ylabel('Total Crime Rate')
plt.title('Total Crime Rate by State in 2022')
plt.xticks(rotation=90)  # Rotate state names for better readability
```

```
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```



5) *Combining the data*:

- Once I got the total crime rate for each year for each state, I added up the total crime rates for all 5 years by state to get my absolute total
- This data was then saved on another dataframe for storage, view and use.
- Please note: **This is not the final dataframe.**

In [46]:
```
# Concatenate the dataframes vertically
all_crime_data = pd.concat([crime_data_2018_sim, crime_data_2019_sim, crime_data

# Group by state and sum the total crime rates
total_crime_per_state = all_crime_data.groupby('state_name')['total_crime_rate']

# Display the resulting dataframe
total_crime_per_state
```

Out[46]:

	state_name	total_crime_rate
0	Alabama	1292214.0
1	Alaska	237826.0
2	Arizona	2012110.0
3	Arkansas	1002778.0
4	California	10750416.0
5	Colorado	1924698.0
6	Connecticut	614744.0
7	Delaware	246370.0
8	District of Columbia	342766.0
9	Florida	4778104.0
10	Georgia	2671612.0
11	Hawaii	411888.0
12	Idaho	253556.0
13	Illinois	2609854.0
14	Indiana	1455176.0
15	Iowa	611270.0
16	Kansas	789346.0
17	Kentucky	891640.0
18	Louisiana	1638364.0
19	Maine	181096.0
20	Maryland	1329606.0
21	Massachusetts	994384.0
22	Michigan	1967400.0
23	Minnesota	1309662.0
24	Mississippi	698180.0
25	Missouri	1856476.0
26	Montana	282208.0
27	Nebraska	437912.0
28	Nevada	853744.0
29	New Hampshire	174752.0
30	New Jersey	1341530.0
31	New Mexico	808540.0
32	New York	3451480.0

	state_name	total_crime_rate
33	North Carolina	2761456.0
34	North Dakota	181360.0
35	Ohio	2607372.0
36	Oklahoma	1245714.0
37	Oregon	1315390.0
38	Pennsylvania	2271938.0
39	Rhode Island	171684.0
40	South Carolina	1667664.0
41	South Dakota	195228.0
42	Tennessee	2172946.0
43	Texas	8010852.0
44	Utah	800914.0
45	Vermont	101502.0
46	Virginia	1549528.0
47	Washington	2523914.0
48	West Virginia	307758.0
49	Wisconsin	1051750.0
50	Wyoming	108772.0

Step 3: **CREATING THE FINAL DATASET**

The following steps were used to create the final dataset:

1) *Obtain State Coordinates*:

- I used the googlemaps library to obtain the coordinates for each state so that plotting can be possible
- I obtained them in dictionary format so I took each value out of the state key and added them to the data frame by latitude and longitude, each as a seperate column to the corresponding state

2) *Ranking the States*:

- After getting the state name, latitude, longitude, and total crime rate from 2018 to 2022, I ranked them in order of highest to the Lowest, with the highest being at rank 1, and the lowest bwing at rank 51.
- I also added their ranking as a column, named danger ranking, thus completing my final dataset.

In [47]:
```
import googlemaps
from datetime import datetime

# Initialize the Google Maps API client
gmaps = googlemaps.Client(key=api_key)  # Replace 'YOUR_API_KEY' with your actua

# Define a function to get the latitude and longitude of a location
def get_coordinates(state_name):
    # Use the Geocoding API to get the latitude and longitude of the state
    geocode_result = gmaps.geocode(state_name)

    # Check if any results were returned
    if geocode_result:
        # Extract latitude and longitude from the result
        latitude = geocode_result[0]['geometry']['location']['lat']
        longitude = geocode_result[0]['geometry']['location']['lng']
        return latitude, longitude
    else:
        print(f"No results found for {state_name}.")
        return None, None

# List of state names
state_names = total_crime_per_state["state_name"]

# Dictionary to store state coordinates
state_coordinates = {}

# Get coordinates for each state
for state_name in state_names:
    latitude, longitude = get_coordinates(state_name)
    if latitude is not None and longitude is not None:
        state_coordinates[state_name] = (latitude, longitude)

# Print the resulting dictionary
print(state_coordinates)
```

```
{'Alabama': (32.3182314, -86.902298), 'Alaska': (63.588753, -154.4930619), 'Arizo
na': (34.0489281, -111.0937311), 'Arkansas': (35.20105, -91.8318334), 'Californi
a': (36.778261, -119.4179324), 'Colorado': (39.5500507, -105.7820674), 'Connectic
ut': (41.6032207, -73.087749), 'Delaware': (38.9108325, -75.52766989999999), 'Dis
trict of Columbia': (38.9071923, -77.0368707), 'Florida': (27.6648274, -81.515753
5), 'Georgia': (42.315407, 43.35689199999999), 'Hawaii': (19.8986819, -155.665856
8), 'Idaho': (44.0682019, -114.7420408), 'Illinois': (40.6331249, -89.3985283),
'Indiana': (40.5512165, -85.60236429999999), 'Iowa': (41.8780025, -93.097702), 'K
ansas': (39.011902, -98.4842465), 'Kentucky': (37.8393332, -84.2700179), 'Louisia
na': (30.5190775, -91.5208624), 'Maine': (45.253783, -69.4454689), 'Maryland': (3
9.0457549, -76.64127119999999), 'Massachusetts': (42.4072107, -71.3824374), 'Mich
igan': (44.3148443, -85.60236429999999), 'Minnesota': (46.729553, -94.6858998),
'Mississippi': (32.3546679, -89.3985283), 'Missouri': (37.9642529, -91.8318334),
'Montana': (46.8796822, -110.3625658), 'Nebraska': (41.4925374, -99.9018131), 'Ne
vada': (38.8026097, -116.419389), 'New Hampshire': (43.1938516, -71.5723953), 'Ne
w Jersey': (40.0583238, -74.4056612), 'New Mexico': (34.9727305, -105.0323635),
'New York': (40.7127753, -74.0059728), 'North Carolina': (35.7595731, -79.0192996
9999999), 'North Dakota': (47.5514926, -101.0020119), 'Ohio': (40.4172871, -82.90
712300000001), 'Oklahoma': (35.0077519, -97.092877), 'Oregon': (43.8041334, -120.
5542012), 'Pennsylvania': (41.2033216, -77.1945247), 'Rhode Island': (41.5800945,
-71.4774291), 'South Carolina': (33.836081, -81.1637245), 'South Dakota': (43.969
5148, -99.9018131), 'Tennessee': (35.5174913, -86.5804473), 'Texas': (31.9685988,
-99.9018131), 'Utah': (39.3209801, -111.0937311), 'Vermont': (44.5588028, -72.577
84149999999), 'Virginia': (37.4315734, -78.6568942), 'Washington': (38.9071923, -
77.0368707), 'West Virginia': (38.5976262, -80.4549026), 'Wisconsin': (43.784439
7, -88.7878678), 'Wyoming': (43.0759678, -107.2902839)}
```

In [48]:

```python
# Convert state_coordinates dictionary to DataFrame
coordinates_df = pd.DataFrame(state_coordinates.values(), index=state_coordinate

# Merge coordinates with the final DataFrame
final_df = pd.merge(total_crime_per_state, coordinates_df, left_on='state_name',

# Sort the DataFrame by total_crime_rate in descending order
final_df = final_df.sort_values(by='total_crime_rate', ascending=False)

# Reset the index to renumber the rows
final_df.reset_index(drop=True, inplace=True)

# Add a new column for danger ranking
final_df['danger_ranking'] = final_df.index + 1

# Reorder the columns
final_df = final_df[['danger_ranking', 'state_name', 'state_latitude', 'state_lo

# Display the updated final DataFrame
final_df
```

Out[48]:

	danger_ranking	state_name	state_latitude	state_longitude	total_crime_rate
0	1	California	36.778261	-119.417932	10750416.0
1	2	Texas	31.968599	-99.901813	8010852.0
2	3	Florida	27.664827	-81.515754	4778104.0
3	4	New York	40.712775	-74.005973	3451480.0
4	5	North Carolina	35.759573	-79.019300	2761456.0
5	6	Georgia	42.315407	43.356892	2671612.0
6	7	Illinois	40.633125	-89.398528	2609854.0
7	8	Ohio	40.417287	-82.907123	2607372.0
8	9	Washington	38.907192	-77.036871	2523914.0
9	10	Pennsylvania	41.203322	-77.194525	2271938.0
10	11	Tennessee	35.517491	-86.580447	2172946.0
11	12	Arizona	34.048928	-111.093731	2012110.0
12	13	Michigan	44.314844	-85.602364	1967400.0
13	14	Colorado	39.550051	-105.782067	1924698.0
14	15	Missouri	37.964253	-91.831833	1856476.0
15	16	South Carolina	33.836081	-81.163725	1667664.0
16	17	Louisiana	30.519078	-91.520862	1638364.0
17	18	Virginia	37.431573	-78.656894	1549528.0
18	19	Indiana	40.551217	-85.602364	1455176.0
19	20	New Jersey	40.058324	-74.405661	1341530.0
20	21	Maryland	39.045755	-76.641271	1329606.0
21	22	Oregon	43.804133	-120.554201	1315390.0
22	23	Minnesota	46.729553	-94.685900	1309662.0
23	24	Alabama	32.318231	-86.902298	1292214.0
24	25	Oklahoma	35.007752	-97.092877	1245714.0
25	26	Wisconsin	43.784440	-88.787868	1051750.0
26	27	Arkansas	35.201050	-91.831833	1002778.0
27	28	Massachusetts	42.407211	-71.382437	994384.0
28	29	Kentucky	37.839333	-84.270018	891640.0
29	30	Nevada	38.802610	-116.419389	853744.0
30	31	New Mexico	34.972730	-105.032364	808540.0
31	32	Utah	39.320980	-111.093731	800914.0
32	33	Kansas	39.011902	-98.484246	789346.0

	danger_ranking	state_name	state_latitude	state_longitude	total_crime_rate
33	34	Mississippi	32.354668	-89.398528	698180.0
34	35	Connecticut	41.603221	-73.087749	614744.0
35	36	Iowa	41.878003	-93.097702	611270.0
36	37	Nebraska	41.492537	-99.901813	437912.0
37	38	Hawaii	19.898682	-155.665857	411888.0
38	39	District of Columbia	38.907192	-77.036871	342766.0
39	40	West Virginia	38.597626	-80.454903	307758.0
40	41	Montana	46.879682	-110.362566	282208.0
41	42	Idaho	44.068202	-114.742041	253556.0
42	43	Delaware	38.910832	-75.527670	246370.0
43	44	Alaska	63.588753	-154.493062	237826.0
44	45	South Dakota	43.969515	-99.901813	195228.0
45	46	North Dakota	47.551493	-101.002012	181360.0
46	47	Maine	45.253783	-69.445469	181096.0
47	48	New Hampshire	43.193852	-71.572395	174752.0
48	49	Rhode Island	41.580095	-71.477429	171684.0
49	50	Wyoming	43.075968	-107.290284	108772.0
50	51	Vermont	44.558803	-72.577841	101502.0

Step 4: **PLOTTING & FINAL ANSWER**

I plotted the following graphs:

1) *Bar Chart*:

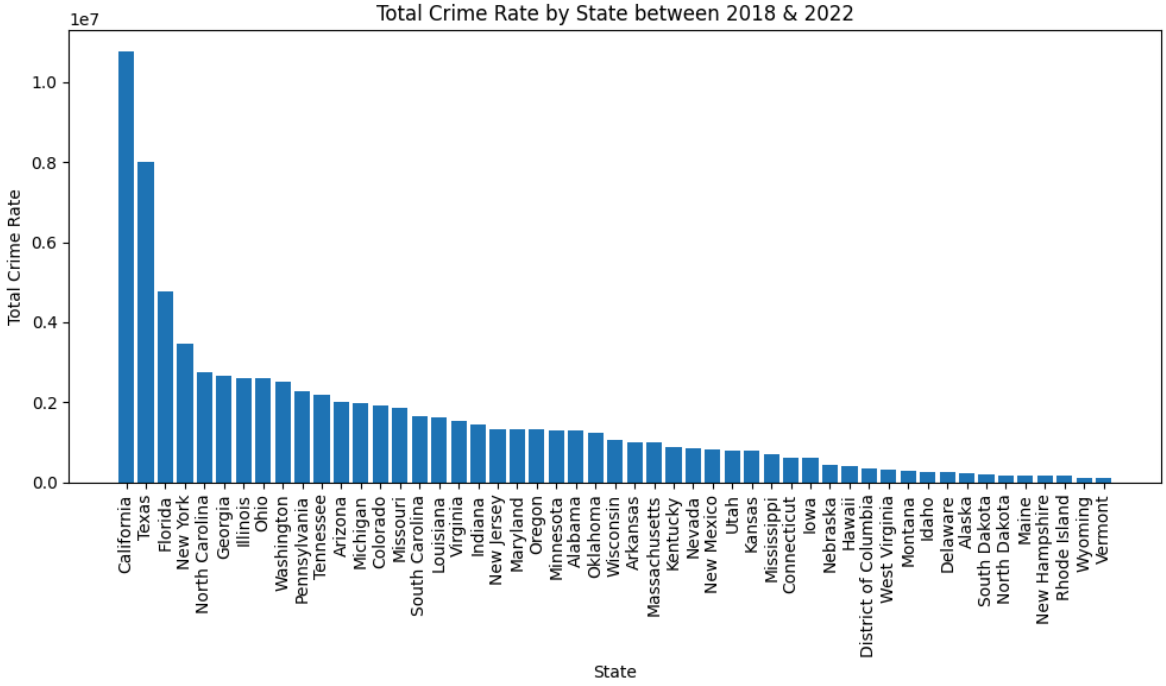
- I used the final data frame, after exporting it as a csv file, to plot out another bar chart to show the ranking of the states by total crime rate showing how they each state ranked compared to the other states.

2) *Final Choropleth Map*:

- I used the data within the final data frame to plot a map of America, and show how unsafe certain states are using circles.
- The bigger and more red the circle is, the more dangerous it becomes.
- The smaller and more blue the circle is, the more safer it becomes.

In [49]:
```
# Export final_df to a CSV file
final_df.to_csv('Final Crime Rate 2018_2022.csv', index=False)
```

```
# Sort the data by state name
final_df_sorted = final_df.sort_values(by='danger_ranking', ascending=True)

# Plot the bar chart
plt.figure(figsize=(10, 6))
plt.bar(final_df_sorted['state_name'], final_df_sorted['total_crime_rate'])
plt.xlabel('State')
plt.ylabel('Total Crime Rate')
plt.title('Total Crime Rate by State between 2018 & 2022')
plt.xticks(rotation=90)  # Rotate state names for better readability
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```



In [55]:

```
import plotly.express as px

# Create scatter mapbox plot with colors based on ancillary data
fig = px.scatter_mapbox(final_df,
                        lat='state_latitude',
                        lon='state_longitude',
                        hover_name='state_name',
                        hover_data=['state_name', 'total_crime_rate'],
                        zoom=1.8,
                        size='total_crime_rate',  # Size of the scatter points b
                        size_max=30,  # Maximum size of the scatter points
                        opacity=0.7,  # Adjust the opacity for better visualizat
                        color='total_crime_rate',  # Color intensity based on th
                        color_continuous_scale='RdYlBu_r',  # Use a diverging co
                        )

# Update layout to make the map larger
fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r": 0, "t": 0, "l": 0, "b": 0}, width=900, height=600

# Show plot
fig.show()
```

From both these craphs we can confirm:

3) *The top 3 most dangerous states in the United States of America betweeen the years 2018 and 2022 are*:

- California,
- Texas, and
- Florida

4) *The top 3 most safest states in the United States of America betweeen the years 2018 and 2022 are*:

- Rhode Island,
- Wyoming, and
- Vermont

In [ ]: