In [2]:
```
import pandas as pd
crime_data = pd.read_csv('Crime_Data_from_2020_to_Present.csv')
```

2. Data Inspection

In [3]:
```
print(crime_data.head())
```

```
        DR_NO                Date Rptd                DATE OCC  TIME OCC  AR
EA  \
0   10304468  01/08/2020 12:00:00 AM  01/08/2020 12:00:00 AM      2230
3
1   190101086  01/02/2020 12:00:00 AM  01/01/2020 12:00:00 AM       330
1
2   200110444  04/14/2020 12:00:00 AM  02/13/2020 12:00:00 AM      1200
1
3   191501505  01/01/2020 12:00:00 AM  01/01/2020 12:00:00 AM      1730
15
4   191921269  01/01/2020 12:00:00 AM  01/01/2020 12:00:00 AM       415
19

     AREA NAME  Rpt Dist No  Part 1-2  Crm Cd  \
0    Southwest          377         2     624
1      Central          163         2     624
```

```

```

```
Status Desc        object
```

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO                  0
Date Rptd              0
DATE OCC               0
TIME OCC               0
AREA                   0
```

3. Data Cleaning

```
2   Central          155         2     845
3   N Hollywood         1543         2     745
4   Mission         1998         2     740
```

In [4]:

```
print(crime_data.dtypes)
```

```
DR_NO              int64
Date Rptd         object
DATE OCC          object
TIME OCC           int64
AREA               int64
AREA NAME         object
Rpt Dist No        int64
Part 1-2           int64
Crm Cd             int64
Crm Cd Desc       object
Mocodes           object
Vict Age           int64
Vict Sex          object
Vict Descent      object
Premis Cd        float64
Premis Desc       object
Weapon Used Cd   float64
Weapon Desc       object
Status            object
```

In [5]:

```
AREA NAME             0
Rpt Dist No           0
Part 1-2              0
Crm Cd                0
Crm Cd Desc           0
Mocodes          112762
Vict Age              0
Vict Sex         107192
Vict Descent     107200
Premis Cd             9
```

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO                0
Date Rptd            0
DATE OCC             0
TIME OCC             0
AREA                 0
Premis Desc        480
Weapon Used Cd  531448
Weapon Desc     531448
Status               0
Status Desc          0
Crm Cd 1            10
Crm Cd 2        755765
Crm Cd 3        813869
Crm Cd 4        815823
LOCATION             0
Cross Street    685361
LAT                  0
LON                  0
dtype: int64
```

In [6]:

```
print(crime_data['Weapon Desc'])
```

```
0                                                STRONG-ARM (HANDS, FIST,
                                                 FEET OR BODILY FORCE)
1                                                UNKNOWN WEAPON/OTHER
                                                 WEAPON
2                                                NaN
3                                                NaN
4                                                NaN
                                                 ...
                                                 815877
                                                 NaN
815878                                           STRONG-ARM (HANDS, FIST,
                                                 FEET OR BODILY FORCE)
815879                                           UNKNOWN WEAPON/OTHER
                                                 WEAPON
815880                                           NaN
815881                                           NaN
Name: Weapon Desc, Length: 815882, dtype: object
```

In [7]:

```
crime_data.drop(['Crm Cd 3', 'Crm Cd 4'], axis=1, inplace=True)
```

In [8]:

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO                 0
Date Rptd             0
DATE OCC              0
TIME OCC              0
AREA                  0
AREA NAME             0
Rpt Dist No           0
Part 1-2              0
Crm Cd                0
Crm Cd Desc           0
Mocodes          112762
Vict Age              0
Vict Sex         107192
Vict Descent     107200
Premis Cd             9
Premis Desc         480
Weapon Used Cd   531448
Weapon Desc      531448
Status                0
Status Desc           0
Crm Cd 1             10
Crm Cd 2         755765
LOCATION              0
Cross Street     685361
LAT                   0
LON                   0
dtype: int64
```

In [9]:
```
crime_data.drop(['Crm Cd 2'], axis=1, inplace=True)
```

In

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO                 0
Date Rptd             0
DATE OCC              0
TIME OCC              0
AREA                  0
```

[10]:

```
AREA NAME             0
Rpt Dist No           0
Part 1-2              0
Crm Cd                0
Crm Cd Desc           0
Mocodes          112762
Vict Age              0
Vict Sex         107192
Vict Descent     107200
Premis Cd             9
Premis Desc         480
Weapon Used Cd   531448
Weapon Desc      531448
Status                0
Status Desc           0
Crm Cd 1             10
LOCATION              0
Cross Street     685361
LAT                   0
LON                   0
dtype: int64
```

In

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO                0
Date Rptd            0
DATE OCC             0
TIME OCC             0
AREA                 0
```

In [11]:

```
total_rows = len(crime_data)
print("Total number of rows:", total_rows)
```

```
Total number of rows: 815882
```

In [12]:

```
crime_data['Vict Sex'].fillna('Unknown', inplace=True)
crime_data['Vict Descent'].fillna('Unknown', inplace=True)
crime_data['Cross Street'].fillna('Unknown', inplace=True)
```

[13]:

```
AREA NAME             0
Rpt Dist No           0
Part 1-2              0
Crm Cd                0
Crm Cd Desc           0
Mocodes          112762
Vict Age              0
Vict Sex              0
Vict Descent          0
Premis Cd             9
Premis Desc         480
Weapon Used Cd    531448
```

In

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO                 0
Date Rptd             0
DATE OCC              0
TIME OCC              0
AREA                  0
Weapon Desc      531448
Status                0
Status Desc           0
Crm Cd 1             10
LOCATION              0
Cross Street          0
LAT                   0
LON                   0
dtype: int64
```

In [14]:

```
crime_data['Weapon Desc'].fillna('Unknown', inplace=True)
crime_data['Weapon Used Cd'].fillna('Unknown', inplace=True)
```

[15]:

```
AREA NAME             0
Rpt Dist No           0
Part 1-2              0
Crm Cd                0
Crm Cd Desc           0
Mocodes          112762
Vict Age              0
Vict Sex              0
Vict Descent          0
```

In

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO                0
Date Rptd            0
DATE OCC             0
TIME OCC             0
AREA                 0
Premis Cd            9
Premis Desc        480
Weapon Used Cd       0
Weapon Desc          0
Status               0
Status Desc          0
Crm Cd 1            10
LOCATION             0
Cross Street         0
LAT                  0
LON                  0
dtype: int64
```

In [16]:

```
crime_data.dropna(subset=['Mocodes'], inplace=True)
```

In [17]:
```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
DR_NO              0
Date Rptd          0
DATE OCC           0
TIME OCC           0
AREA               0
AREA NAME          0
Rpt Dist No        0
Part 1-2           0
Crm Cd             0
Crm Cd Desc        0
Mocodes            0
Vict Age           0
Vict Sex           0
Vict Descent       0
Premis Cd          0
Premis Desc      469
Weapon Used Cd     0
Weapon Desc        0
Status             0
Status Desc        0
Crm Cd 1          10
LOCATION           0
Cross Street       0
LAT                0
LON                0
dtype: int64
```

In [18]:
```
crime_data.dropna(subset=['Premis Desc'], inplace=True)
crime_data['Crm Cd 1'].fillna(crime_data['Crm Cd 1'].mean(), inplace=True)
```

[19]:

```
DR_NO          0
Date Rptd      0
DATE OCC       0
TIME OCC       0
```

In

```
missing_values = crime_data.isnull().sum()
print(missing_values)
```

```
AREA              0
AREA NAME         0
Rpt Dist No       0
Part 1-2          0
Crm Cd            0
Crm Cd Desc       0
Mocodes           0
Vict Age          0
Vict Sex          0
Vict Descent      0
Premis Cd         0
Premis Desc       0
Weapon Used Cd    0
Weapon Desc       0
Status            0
Status Desc       0
Crm Cd 1          0
LOCATION          0
Cross Street      0
LAT               0
LON               0
dtype: int64
```

4. Exploratory Data Analysis (EDA)

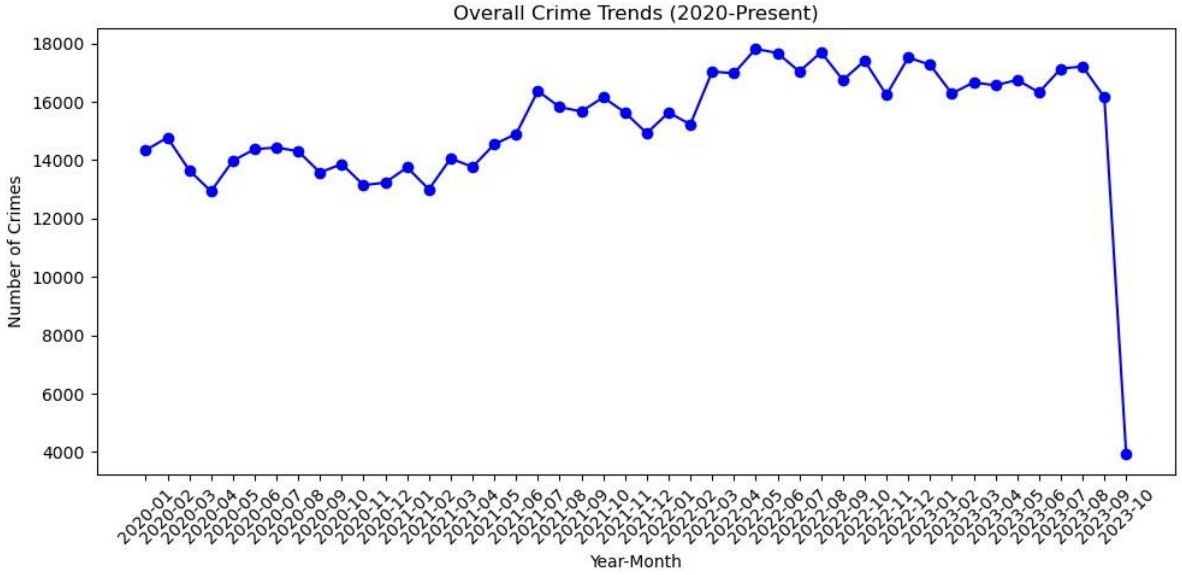
[20]:

```
#for overall crime trends (2020-Present)
import pandas as pd
import matplotlib.pyplot as plt

crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])
crime_data['Year'] = crime_data['Date Rptd'].dt.year
crime_data['Month'] = crime_data['Date Rptd'].dt.month
crime_data_filtered = crime_data[crime_data['Year'] >= 2020]
crime_counts = crime_data_filtered.groupby(['Year', 'Month']).size().reset

plt.figure(figsize=(10, 5))
plt.plot(crime_counts['Year'].astype(str) + '-' + crime_counts['Month'].as
plt.xlabel('Year-Month')
```

In

```
plt.ylabel('Number of Crimes') plt.title('Overall Crime Trends (2020-Present)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
C:\Users\prabh\AppData\Local\Temp\ipykernel_16380\2007973645.py:5: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.  crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])
```



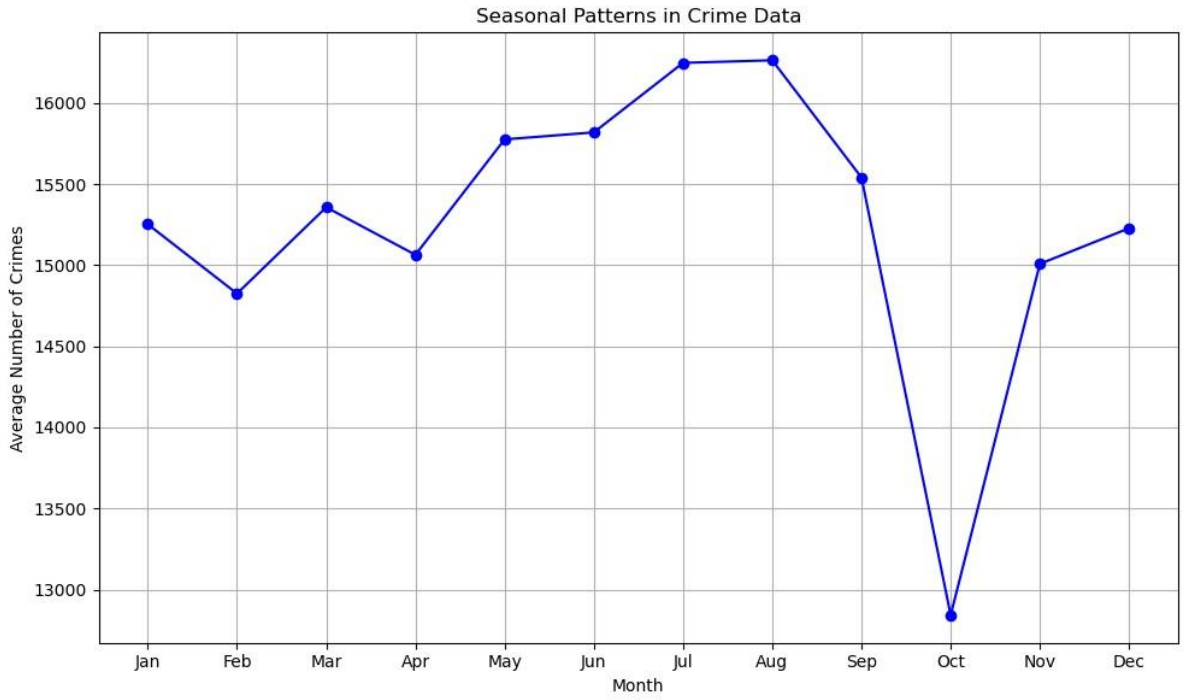
In [21]:

```
import pandas as pd

crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])
crime_data['Month'] = crime_data['Date Rptd'].dt.month
crime_data['Year'] = crime_data['Date Rptd'].dt.year

monthly_crime_counts = crime_data.groupby(['Year', 'Month']).size().reset_
average_monthly_crime_counts = monthly_crime_counts.groupby('Month')['Coun
print(average_monthly_crime_counts)
```

```
Month
1     15252.250000
2     14824.750000
3     15357.000000
4     15064.000000
5     15775.500000
6     15818.750000
7     16247.250000
8     16263.250000
9     15540.500000
10    12842.750000
11    15008.000000
12    15227.666667
Name: Count, dtype: float64
```

In [22]:

```
#for seasonal patterns in crime data
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.plot(average_monthly_crime_counts.index, average_monthly_crime_counts.
plt.xlabel('Month')
plt.ylabel('Average Number of Crimes')
plt.title('Seasonal Patterns in Crime Data')
plt.xticks(range(1, 13), ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul',
plt.grid(True)
plt.tight_layout()
plt.show()
```



Identifying the Most Common Type of Crime

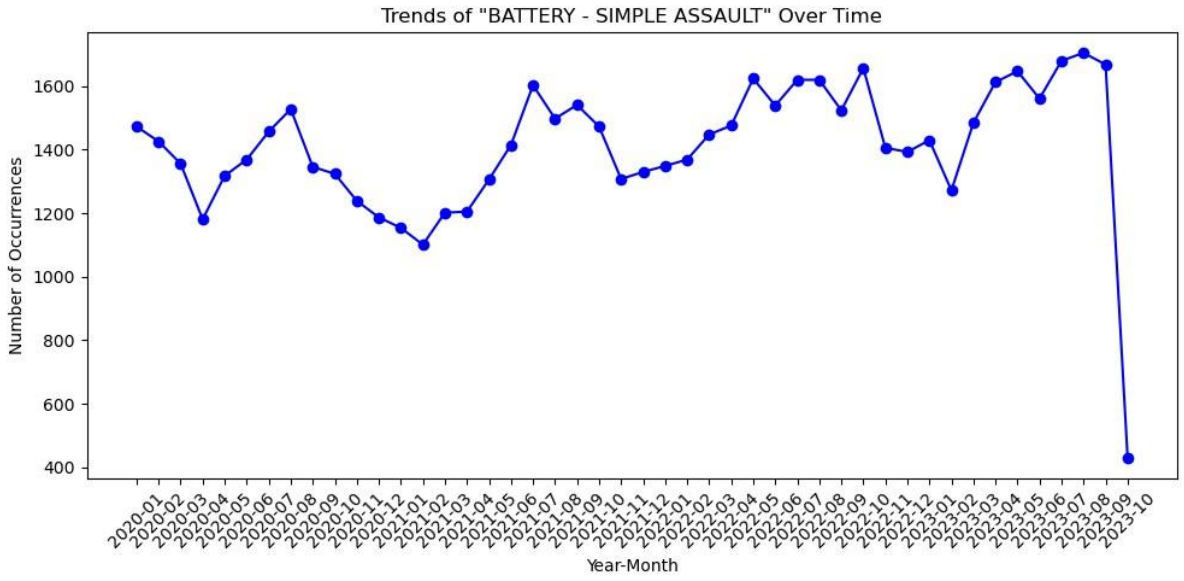
In [23]:

```
crime_counts = crime_data['Crm Cd Desc'].value_counts()
most_common_crime = crime_counts.index[0]
most_common_crime_count = crime_counts.iloc[0]
print(f"The most common type of crime is '{most_common_crime}' with {most_
```

```
The most common type of crime is 'BATTERY - SIMPLE ASSAULT' with 64856 occurr
ences.
```

Analyzing Trends of the Most Common Crime Over Time:

In [24]:
```
most_common_crime_data = crime_data[crime_data['Crm Cd Desc'] == most_comm

most_common_crime_monthly_counts = most_common_crime_data.groupby(['Year',

plt.figure(figsize=(10, 5))
plt.plot(most_common_crime_monthly_counts['Year'].astype(str) + '-' + most
plt.xlabel('Year-Month')
plt.ylabel('Number of Occurrences')
plt.title(f'Trends of "{most_common_crime}" Over Time')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Grouping Data by regions

In [25]:
```
crime_by_area = crime_data.groupby('AREA NAME').size().reset_index(name='C
```

Comparing Crime Rates:

[26]:
```
crime_by_area_sorted = crime_by_area.sort_values(by='Crime Count', ascendi
print(crime_by_area_sorted)
```

```
      AREA NAME  Crime Count
1       Central        50585
0    77th Street        43336
12      Pacific        40591
15    Southwest        39404
6     Hollywood        39404
11      Olympic        35602
14    Southeast        34924
```

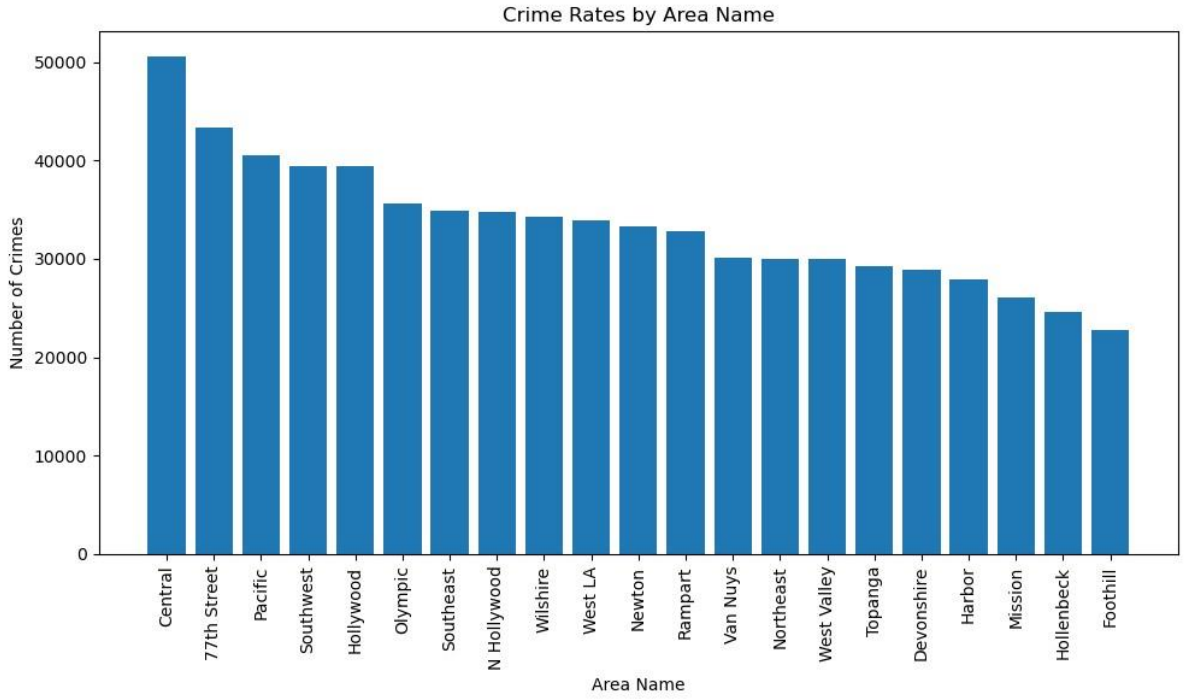
In

```
8   N Hollywood        34788
20     Wilshire        34305
18      West LA        33928
9         Newton       33319
13      Rampart        32865
17     Van Nuys        30086
10    Northeast        30007
19  West Valley        29963
16      Topanga        29306
2    Devonshire        28917
4         Harbor       27875
7       Mission        26094
5    Hollenbeck        24592
3      Foothill        22760
```

Visualizing the Differences

In [27]:

```
plt.figure(figsize=(10, 6))
plt.bar(crime_by_area_sorted['AREA NAME'], crime_by_area_sorted['Crime Cou
plt.xlabel('Area Name')
plt.ylabel('Number of Crimes')
plt.title('Crime Rates by Area Name')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



Explore correlations between economic factors (if available) and crime rates.

In [28]:

```
file_path = '2020data_to_2023data.csv'
dataset = pd.read_csv(file_path)
dataset.head(10)
```

Out[28]:

	Year	Month	Labor Force	Employment	Unemployment	Unemployment rate
0	2020	January	6887404	6580637	306767	4.5
1	2020	February	6929744	6626136	303608	4.4
2	2020	March	6624544	6221296	403248	6.1
3	2020	April	6316165	5281552	1034613	16.4
4	2020	May	6201220	5081930	1119290	18.0
5	2020	June	6488757	5405845	1082912	16.7
6	2020	July	6615813	5481710	1134103	17.1

In

7	2020	August	6588811	5573127	101568415.4
8	2020	September	63747405	6807516	93989 10.9
9	2020	October	64243025	7918906	32412 9.8

[29]:
```
economic_by_year = dataset.groupby('Year').size().reset_index(name='Econom
economic_by_year
```

Out[29]:

	Year	Economic Count
0	2020	12
1	2021	12
2	2022	12
3	2023	9

In [30]:
```
crime_by_year = crime_data.groupby('Year').size().reset_index(name='Crime
crime_by_year
```

Out[30]:

	Year	Crime Count
0	2020	166655
1	2021	178626
2	2022	203045
3	2023	154325

In [31]:
```
correlation = economic_by_year['Economic Count'].corr(crime_by_year['Crime

print(f"Correlation between Unemployment rate and crime count: {correlatio
```

```
Correlation between Unemployment rate and crime count: 0.6846708405377957
```

Extracting the Day of the Week

In [32]:
```
crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])
crime_data['Day of Week'] = crime_data['Date Rptd'].dt.dayofweek
```

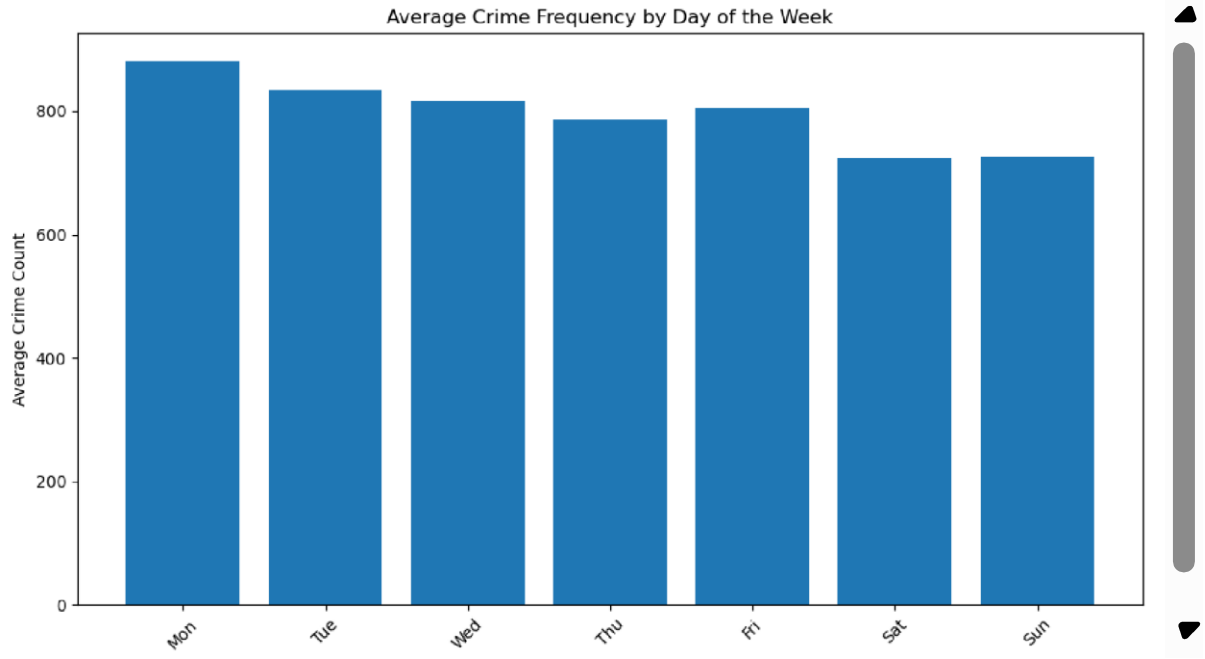
Grouping the Data and Analyzing

In [33]:
```
crime_type_counts = crime_data.groupby(['Day of Week', 'Crm Cd Desc']).siz
```

Visualizing the relationship between the day of the week and the frequency of certain types of crimes

In [34]:

```
import numpy as np

avg_crime_counts = crime_data.groupby('Day of Week')['Crm Cd Desc'].value_
plt.figure(figsize=(10, 6))
x = np.arange(len(avg_crime_counts))
plt.bar(x, avg_crime_counts, tick_label=['Mon', 'Tue', 'Wed', 'Thu', 'Fri'
plt.xlabel('Day of the Week')
plt.ylabel('Average Crime Count')
plt.title('Average Crime Frequency by Day of the Week')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Investigate any impact of major events or policy changes on crime rates

The George Floyd protests in Los Angeles began on May 27, 2020 and ended around June 13, 2020. We have decided to perform a t-test to find if there have been any significant changes in crime rates before and after these events lasted. First we will consider a time period of two months before and after for this.

[35]:

```
import pandas as pd

crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])

start_date_before = pd.to_datetime('2020-03-27')
end_date_before = pd.to_datetime('2020-05-27')
start_date_after = pd.to_datetime('2020-06-13')
end_date_after = pd.to_datetime('2020-08-13')

```

In

```
data_before = crime_data[(crime_data['Date Rptd'] >= start_date_before) &
data_after = crime_data[(crime_data['Date Rptd'] >= start_date_after) & (c

data_before.loc[:, 'Crime Rate'] = data_before.groupby(data_before['Date R
data_after.loc[:, 'Crime Rate'] = data_after.groupby(data_after['Date Rptd
```

In [36]:

```
from scipy import stats

t_stat, p_value = stats.ttest_ind(data_before['Crime Rate'], data_after['C

print("T-statistic:", t_stat)
print("P-value:", p_value)

alpha = 0.05
if p_value < alpha:
print("As the p-value is very less than the alpha value of 0.05,we can
else:
  print("As the p-value is higher than the alpha value of 0.05,we can co
```

```
T-statistic: 31.363996969078233
P-value: 4.63352944160365e-214
As the p-value is very less than the alpha value of 0.05,we can conclude that
there is a significant difference in crime rates between the two time period
s.
```

We can also visualize the change in crime rates before and after a certain event using matplotlib

Let us consider another event when the curfew imposed by the LAPD was lifted on June 1, 2020

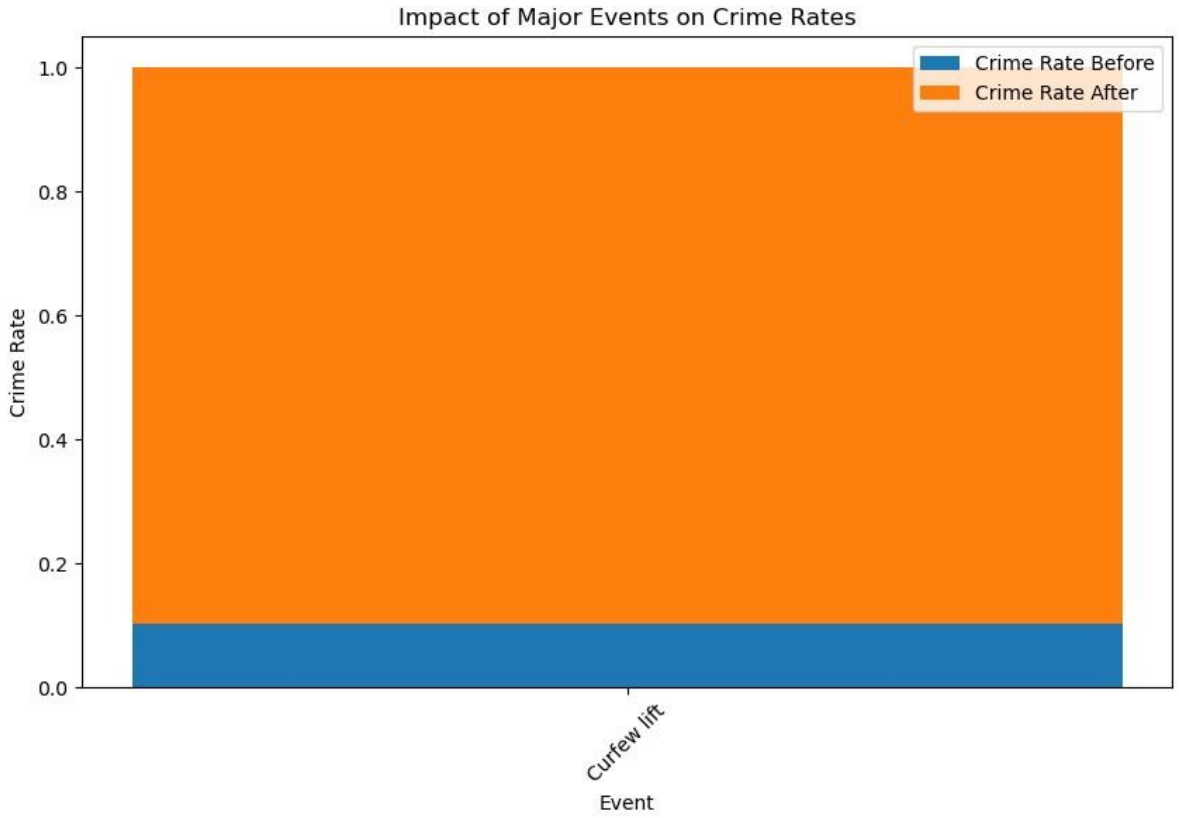
[37]:

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('Crime_Data_from_2020_to_Present.csv')
events = {
  'Curfew lift': '2020-06-01'
    }
    date_format = '%Y-%m-%d'
    data['DATE OCC'] = pd.to_datetime(data['DATE OCC'])
    crime_rates_before = []
    crime_rates_after = []
    for event, event_date in events.items():
    event_date = pd.to_datetime(event_date)
    before_event = data[data['DATE OCC'] < event_date]
    after_event = data[data['DATE OCC'] >= event_date]

```

In

```
    crime_rate_before = len(before_event) / len(data)
    crime_rate_after = len(after_event) / len(data)

    crime_rates_before.append(crime_rate_before)
    crime_rates_after.append(crime_rate_after)

plt.figure(figsize=(10, 6))
plt.bar(events.keys(), crime_rates_before, label='Crime Rate Before')
plt.bar(events.keys(), crime_rates_after, label='Crime Rate After', bottom
plt.xlabel('Event')
plt.ylabel('Crime Rate')
plt.title('Impact of Major Events on Crime Rates')
plt.legend()
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\prabh\AppData\Local\Temp\ipykernel_16380\1956949937.py:9: UserWarnin
g: Could not infer format, so each element will be parsed individually, falli
ng back to `dateutil`. To ensure parsing is consistent and as-expected, pleas
e specify a format.  data['DATE OCC'] = pd.to_datetime(data['DATE OCC'])
```



We can see from the plot that there has been a significant increase in the crime rate after the curfew was lifted.

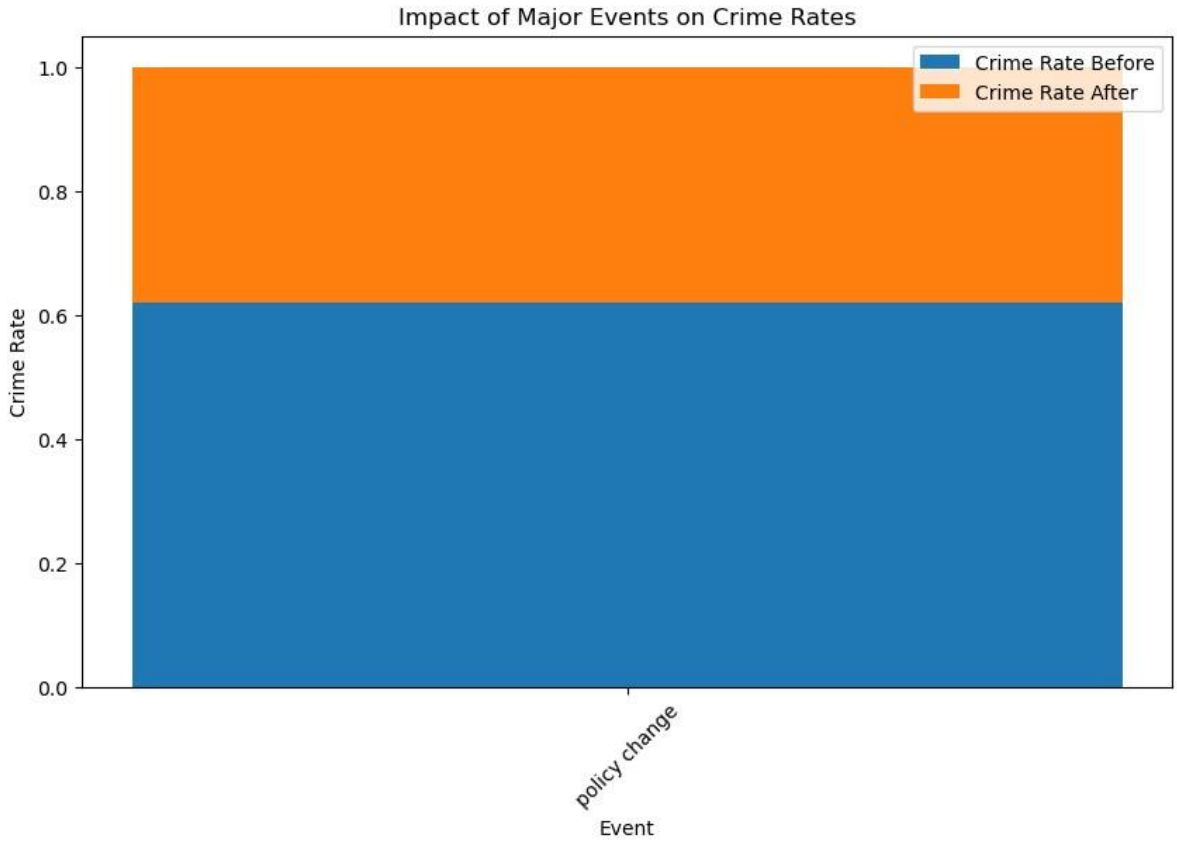
A case of policy changes that could effect the crime rate has happened on June 2, 2022, when Los Angeles County District Attorney George Gascón announced that he will not prosecute certain misdemeanors, including drug possession and prostitution. This decision was made in an effort to reduce the number of people incarcerated in Los Angeles County jails and to focus on prosecuting more serious crimes. We will try to plot the crimes before and after this decision.

[38]:
```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('Crime_Data_from_2020_to_Present.csv')
events = {
  'policy change': '2022-06-02'
   }
   date_format = '%Y-%m-%d'
   data['DATE OCC'] = pd.to_datetime(data['DATE OCC'])
   crime_rates_before = []
   crime_rates_after = []
   for event, event_date in events.items():
   event_date = pd.to_datetime(event_date)
```

In

```
    before_event = data[data['DATE OCC'] < event_date]
    after_event = data[data['DATE OCC'] >= event_date]

    crime_rate_before = len(before_event) / len(data)
    crime_rate_after = len(after_event) / len(data)

    crime_rates_before.append(crime_rate_before)
    crime_rates_after.append(crime_rate_after)

plt.figure(figsize=(10, 6))
plt.bar(events.keys(), crime_rates_before, label='Crime Rate Before')
plt.bar(events.keys(), crime_rates_after, label='Crime Rate After', bottom
plt.xlabel('Event')
plt.ylabel('Crime Rate')
plt.title('Impact of Major Events on Crime Rates')
plt.legend()
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\prabh\AppData\Local\Temp\ipykernel_16380\3230390676.py:9: UserWarnin
g: Could not infer format, so each element will be parsed individually, falli
ng back to `dateutil`. To ensure parsing is consistent and as-expected, pleas
e specify a format.  data['DATE OCC'] = pd.to_datetime(data['DATE OCC'])
```



We can see from the plot that the crime rate took a surprising turn resulting in decrease in the crime rate after the DA's decision. We can understand that this decision has not made an impact on the crime rate.

Questions:

1) Overall Crime Trends

Extracting year from date

In [39]:
```
crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])
crime_data['Year'] = crime_data['Date Rptd'].dt.year
```

Calculating total crimes per year

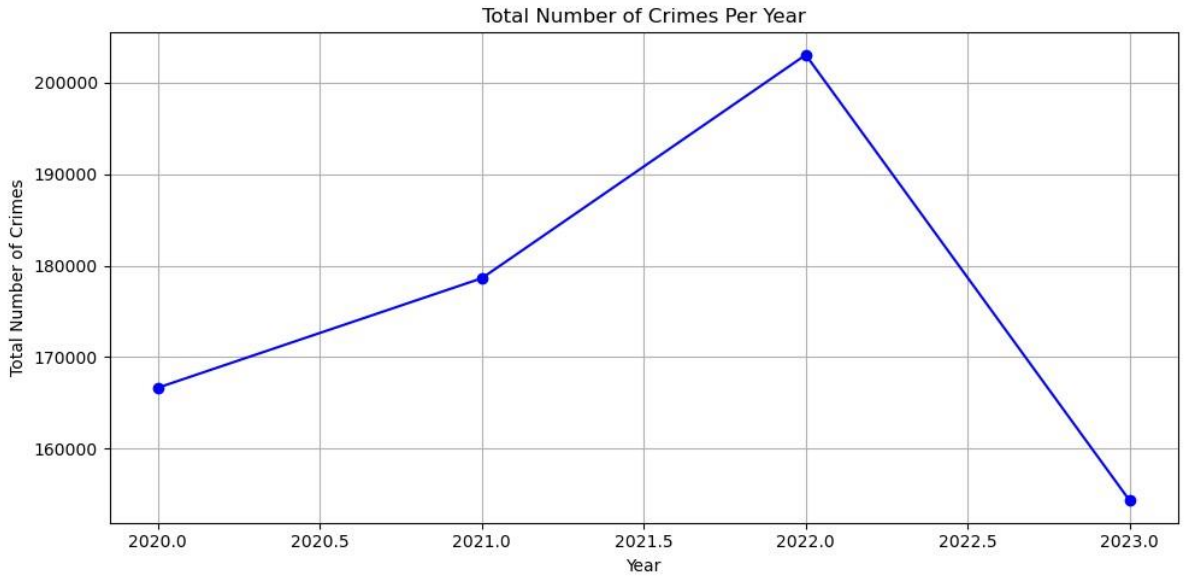
In [40]:
```
crimes_per_year = crime_data.groupby('Year').size()
```

In

Plotting the total number of crimes per year

[41]:

```
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 5))
plt.plot(crimes_per_year.index, crimes_per_year.values, marker='o', linest
plt.xlabel('Year')
plt.ylabel('Total Number of Crimes')
plt.title('Total Number of Crimes Per Year')
plt.grid(True)
plt.tight_layout()
plt.show()
```



2) Seasonal Patterns

Extracting month and year from date

In [42]:

```
crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])
crime_data['Month'] = crime_data['Date Rptd'].dt.month
crime_data['Year'] = crime_data['Date Rptd'].dt.year
```

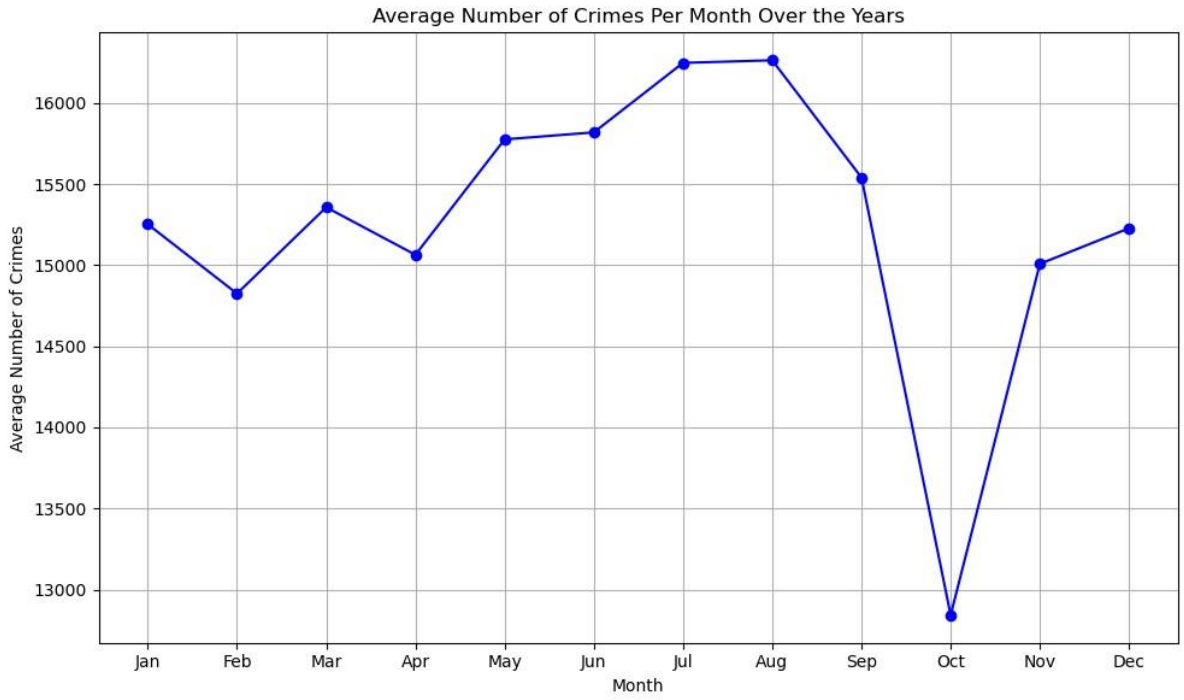
Calculating average crimes per month

In [43]:
```
1 average_crimes_per_month = crime_data.groupby(['Year', 'Month']).size().gr
```

Plotting the average number of crimes per month

In [44]:

```
import calendar
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.plot(range(1, 13), average_crimes_per_month.values, marker='o', linest
plt.xlabel('Month')
plt.ylabel('Average Number of Crimes')
plt.title('Average Number of Crimes Per Month Over the Years')
plt.xticks(range(1, 13), [calendar.month_abbr[i] for i in range(1, 13)])
plt.grid(True)
plt.tight_layout()
plt.show()
```



3. Most common crime type

In [45]:

```
crime_type_counts = crime_data['Crm Cd Desc'].value_counts()
most_common_crime_type = crime_type_counts.index[0]
most_common_crime_type_count = crime_type_counts.iloc[0]
print(f"The most common crime type is '{most_common_crime_type}' with {mos
```

```
The most common crime type is 'BATTERY - SIMPLE ASSAULT' with 64856 occurrenc
es.
```

4. Regional Differences:

Grouping data by areas

In [46]:
```
crime_by_area = crime_data.groupby('AREA NAME').size().reset_index(name='C
crime_by_area
```

Out[46]:

	AREA NAME	Crime Count
0	77th Street	43336
1	Central	50585
2	Devonshire	28917
3	Foothill	22760
4	Harbor	27875
5	Hollenbeck	24592
6	Hollywood	39404
7	Mission	26094
8	N Hollywood	34788
9	Newton	33319
10	Northeast	30007
11	Olympic	35602
12	Pacific	40591
13	Rampart	32865
14	Southeast	34924
15	Southwest	39404
16	Topanga	29306
17	Van Nuys	30086
18	West LA	33928
19	West Valley	29963
20	Wilshire	34305

Descriptive Statistics

In [47]:
```
print(crime_by_area['Crime Count'].describe())
```

```
count       21.000000
mean     33459.571429
std       6611.084227
min      22760.000000
25%      29306.000000
50%      33319.000000 75%
35602.000000 max
```

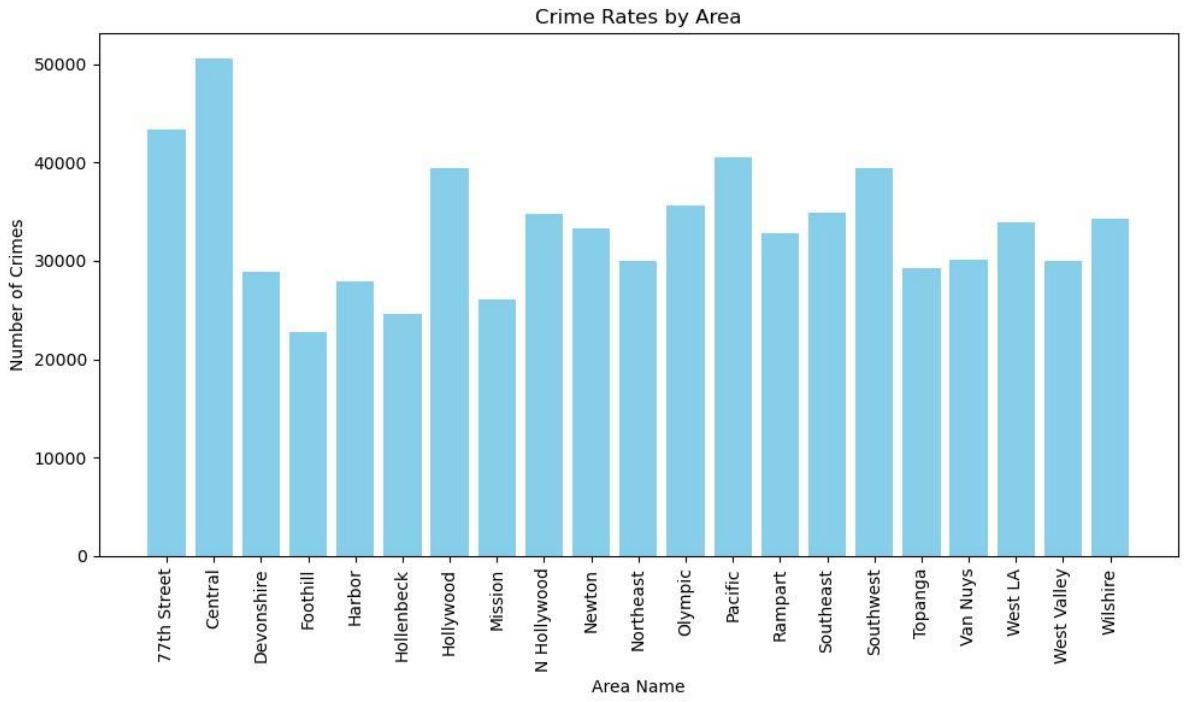
In

```
50585.000000 Name: Crime Count,
dtype: float64
```

Box Plot Visualization

[48]:

```
plt.figure(figsize=(10, 6))
plt.bar(crime_by_area['AREA NAME'], crime_by_area['Crime Count'], color='s
plt.xlabel('Area Name')
plt.ylabel('Number of Crimes')
plt.title('Crime Rates by Area')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



5. Correlation with Economic Factors

In [49]:

```
file_path = '2020data_to_2023data.csv'
dataset = pd.read_csv(file_path)
dataset.head(10)
```

Out[49]:

	Year	Month	Labor Force	Employment	Unemployment	Unemployment rate
0	2020	January	68874046	58063730	6767	4.5
1	2020	February	69297446	62613630	3608	4.4
2	2020	March	66245446	22129640	3248	6.1

In

3	2020	April	631616552815521034613	16.4
4	2020	May	620122050819301119290	18.0
5	2020	June	648875754058451082912	16.7
6	2020	July	661581354817101134103	17.1
7	2020	August	6588811 5573127101568	15.4
8	2020	September	637474056807516939 89	10.9
9	2020	October	642430257918906324 12	9.8

[50]:
```
economic_by_year = dataset.groupby('Year').size().reset_index(name='Econom
economic_by_year
```

Out[50]:

	Year	Economic Count
0	2020	12
1	2021	12
2	2022	12
3	2023	9

In [51]:
```
print(crime_data.columns)
```

```
Index(['DR_NO', 'Date Rptd', 'DATE OCC', 'TIME OCC', 'AREA', 'AREA NAME',
       'Rpt Dist No', 'Part 1-2', 'Crm Cd', 'Crm Cd Desc', 'Mocodes',
       'Vict Age', 'Vict Sex', 'Vict Descent', 'Premis Cd', 'Premis Desc',
       'Weapon Used Cd', 'Weapon Desc', 'Status', 'Status Desc', 'Crm Cd 1',
 'LOCATION', 'Cross Street', 'LAT', 'LON', 'Year', 'Month',
       'Day of Week'],
 dtype='object')
```

In [52]:
```
crime_by_year = crime_data.groupby('Year').size().reset_index(name='Crime 2
crime_by_year
```

Out[52]:

	Year	Crime Count
0	2020	166655
1	2021	178626
2	2022	203045
3	2023	154325

In [53]:
```
correlation = economic_by_year['Economic Count'].corr(crime_by_year['Crime
print(f"Correlation between Unemployment rate and crime count: {correlatio
```

In

```
Correlation between Unemployment rate and crime count: 0.6846708405377957
```

6. Day of the Week Analysis

Extracting day of the week

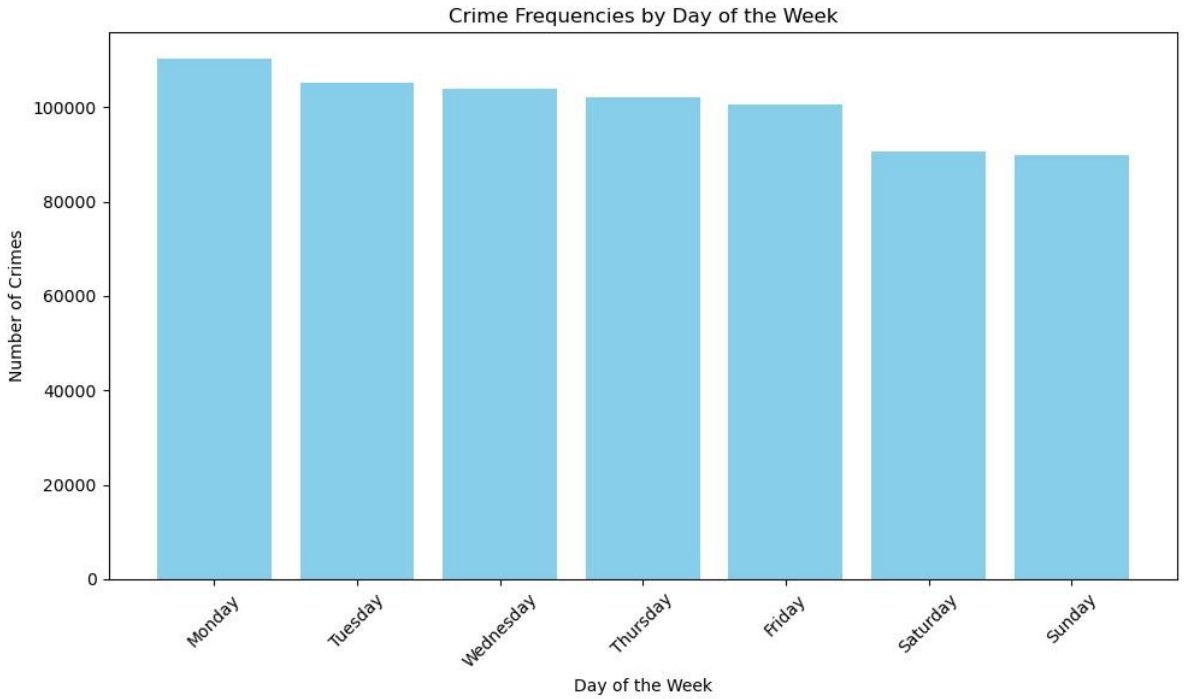
In [54]:
```
crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])
crime_data['Day of Week'] = crime_data['Date Rptd'].dt.dayofweek
```

Grouping data by day of the week

In [55]:
```
crime_by_day = crime_data.groupby('Day of Week').size().reset_index(name='
```

Plotting crime frequencies for each day

[56]:
```
import calendar
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.bar(crime_by_day['Day of Week'], crime_by_day['Crime Count'], color='s
plt.xlabel('Day of the Week')
plt.ylabel('Number of Crimes')
plt.title('Crime Frequencies by Day of the Week')
plt.xticks(range(7), [calendar.day_name[i] for i in range(7)], rotation=45
plt.tight_layout()
plt.show()
```



In

7. Impact of Major Events

The George Floyd protests in Los Angeles began on May 27, 2020 and ended around June 13, 2020. We have decided to perform a t-test to find if there have been any significant changes in crime rates before and after these events lasted. First we will consider a time period of two months before and after for this.

[57]:
```
import pandas as pd

crime_data['Date Rptd'] = pd.to_datetime(crime_data['Date Rptd'])

 start_date_before = pd.to_datetime('2020-03-27')
 end_date_before = pd.to_datetime('2020-05-27')
 start_date_after = pd.to_datetime('2020-06-13')
 end_date_after = pd.to_datetime('2020-08-13')

data_before = crime_data[(crime_data['Date Rptd'] >= start_date_before) &
data_after = crime_data[(crime_data['Date Rptd'] >= start_date_after) & (c

data_before.loc[:, 'Crime Rate'] = data_before.groupby(data_before['Date R
data_after.loc[:, 'Crime Rate'] = data_after.groupby(data_after['Date Rptd
```

In [58]:
```
from scipy import stats

t_stat, p_value = stats.ttest_ind(data_before['Crime Rate'], data_after['C
print("T-statistic:", t_stat)
print("P-value:", p_value)
alpha = 0.05
 if p_value < alpha:
 print("As the p-value is very less than the alpha value of 0.05,we can 9
 else:
    print("As the p-value is higher than the alpha value of 0.05,we can co
```

```
T-statistic: 31.363996969078233
P-value: 4.63352944160365e-214
As the p-value is very less than the alpha value of 0.05,we can conclude that
there is a significant difference in crime rates between the two time period
s.
```

We can also visualize the change in crime rates before and after a certain event using matplotlib

Let us consider another event when the curfew imposed by the LAPD was lifted on June 1, 2020

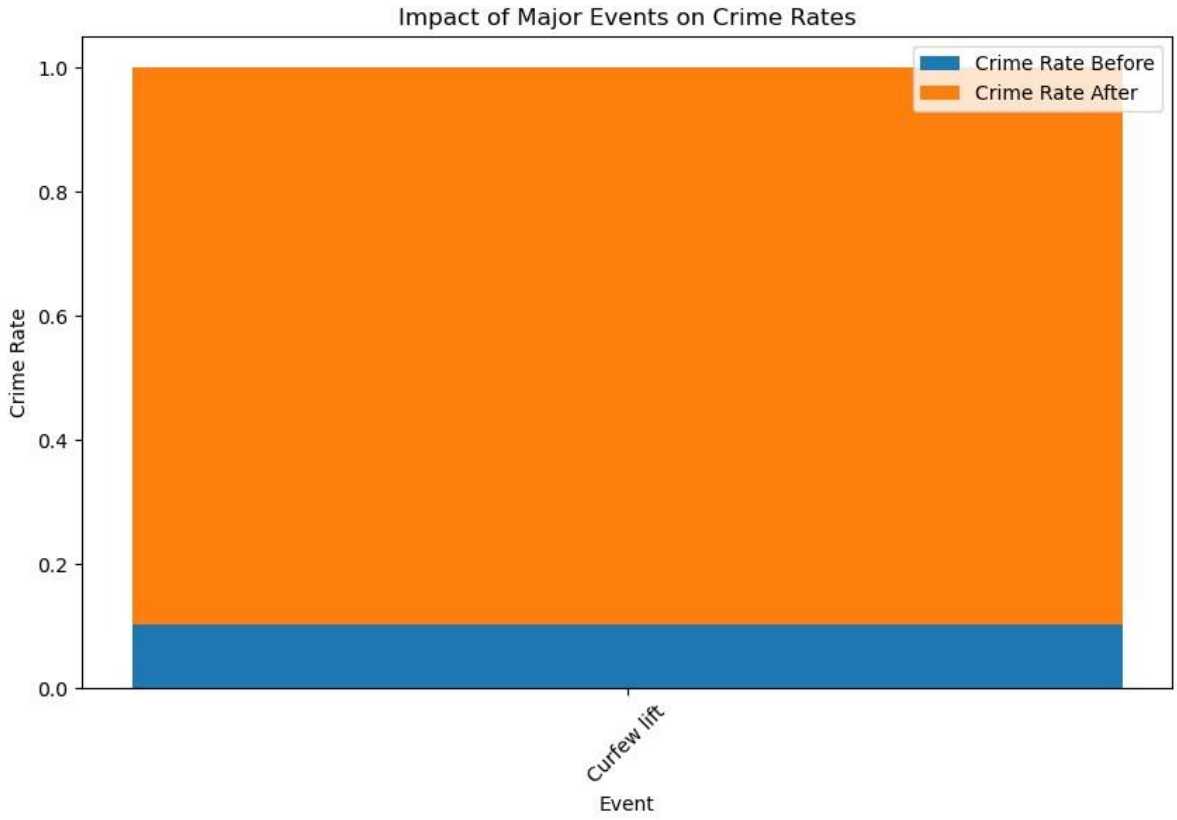
[59]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In

```

data = pd.read_csv('Crime_Data_from_2020_to_Present.csv')
events = {
  'Curfew lift': '2020-06-01'
    }
    date_format = '%Y-%m-%d'
    data['DATE OCC'] = pd.to_datetime(data['DATE OCC'])
    crime_rates_before = []
    crime_rates_after = []
    for event, event_date in events.items():
    event_date = pd.to_datetime(event_date)
    before_event = data[data['DATE OCC'] < event_date]
    after_event = data[data['DATE OCC'] >= event_date]

    crime_rate_before = len(before_event) / len(data)
    crime_rate_after = len(after_event) / len(data)

    crime_rates_before.append(crime_rate_before)
    crime_rates_after.append(crime_rate_after)

plt.figure(figsize=(10, 6))
plt.bar(events.keys(), crime_rates_before, label='Crime Rate Before')
plt.bar(events.keys(), crime_rates_after, label='Crime Rate After', bottom
plt.xlabel('Event')
plt.ylabel('Crime Rate')
plt.title('Impact of Major Events on Crime Rates')
plt.legend()
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\prabh\AppData\Local\Temp\ipykernel_16380\1956949937.py:9: UserWarnin
g: Could not infer format, so each element will be parsed individually, falli
ng back to `dateutil`. To ensure parsing is consistent and as-expected, pleas
e specify a format.  data['DATE OCC'] = pd.to_datetime(data['DATE OCC'])
```

In



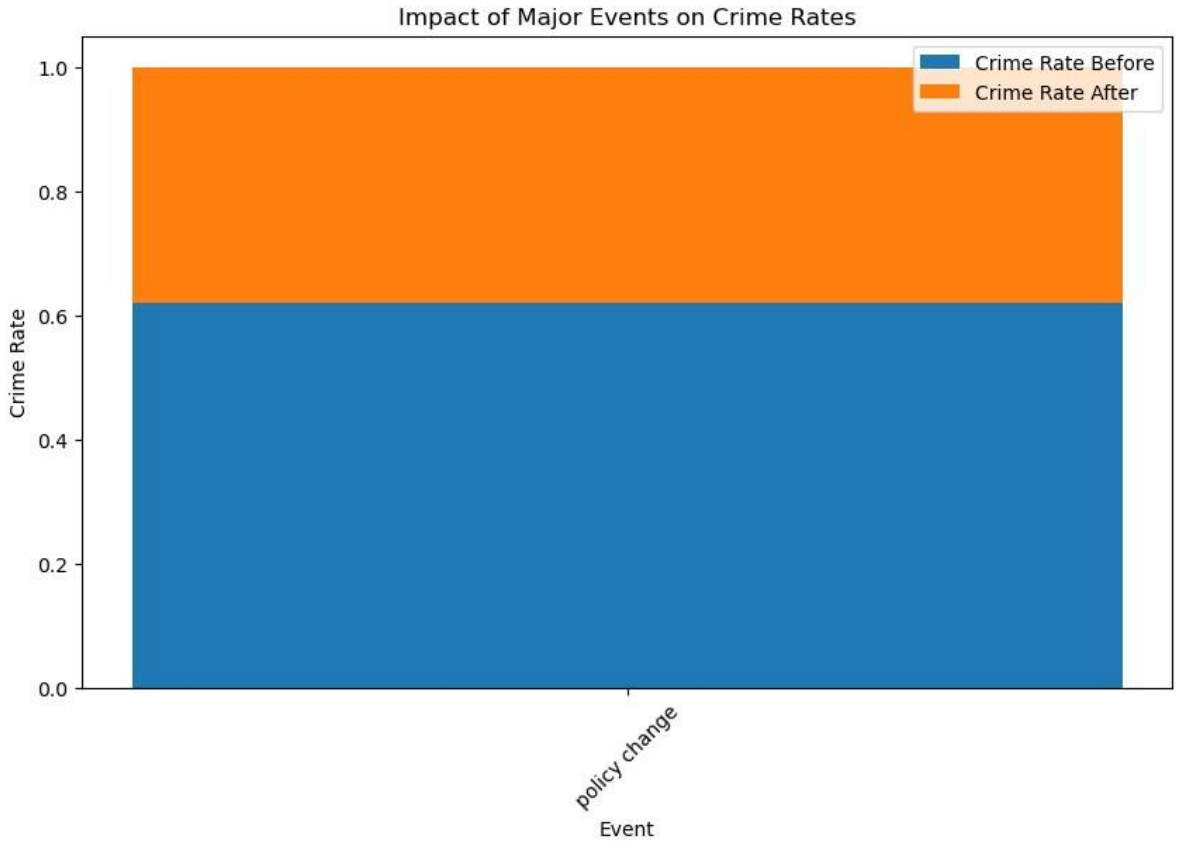
We can see from the plot that there has been a significant increase in the crime rate after the curfew was lifted.

A case of policy changes that could effect the crime rate has happened on June 2, 2022, when Los Angeles County District Attorney George Gascón announced that he will not prosecute certain misdemeanors, including drug possession and prostitution. This decision was made in an effort to reduce the number of people incarcerated in Los Angeles County jails and to focus on prosecuting more serious crimes. We will try to plot the crimes before and after this decision. [60]: 1 import pandas as pd

```
import matplotlib.pyplot as plt

data = pd.read_csv('Crime_Data_from_2020_to_Present.csv')
events = {
   'policy change': '2022-06-02'
    }
    date_format = '%Y-%m-%d'
    data['DATE OCC'] = pd.to_datetime(data['DATE OCC'])
    crime_rates_before = []
    crime_rates_after = []
    for event, event_date in events.items():
    event_date = pd.to_datetime(event_date)
    before_event = data[data['DATE OCC'] < event_date]
    after_event = data[data['DATE OCC'] >= event_date]

    crime_rate_before = len(before_event) / len(data)
    crime_rate_after = len(after_event) / len(data)

    crime_rates_before.append(crime_rate_before)
    crime_rates_after.append(crime_rate_after)

plt.figure(figsize=(10, 6))
plt.bar(events.keys(), crime_rates_before, label='Crime Rate Before')
plt.bar(events.keys(), crime_rates_after, label='Crime Rate After', bottom
plt.xlabel('Event')
plt.ylabel('Crime Rate')
plt.title('Impact of Major Events on Crime Rates')
plt.legend()
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\prabh\AppData\Local\Temp\ipykernel_16380\3230390676.py:9: UserWarnin
g: Could not infer format, so each element will be parsed individually,
falli ng back to `dateutil`. To ensure parsing is consistent and as-
expected, pleas e specify a format.  data['DATE OCC'] =
pd.to_datetime(data['DATE OCC'])
```

In



We can see from the plot that the crime rate took a surprising turn resulting in decrease in the crime rate after the DA's decision. We can understand that this decision has not made an impact on the crime rate.

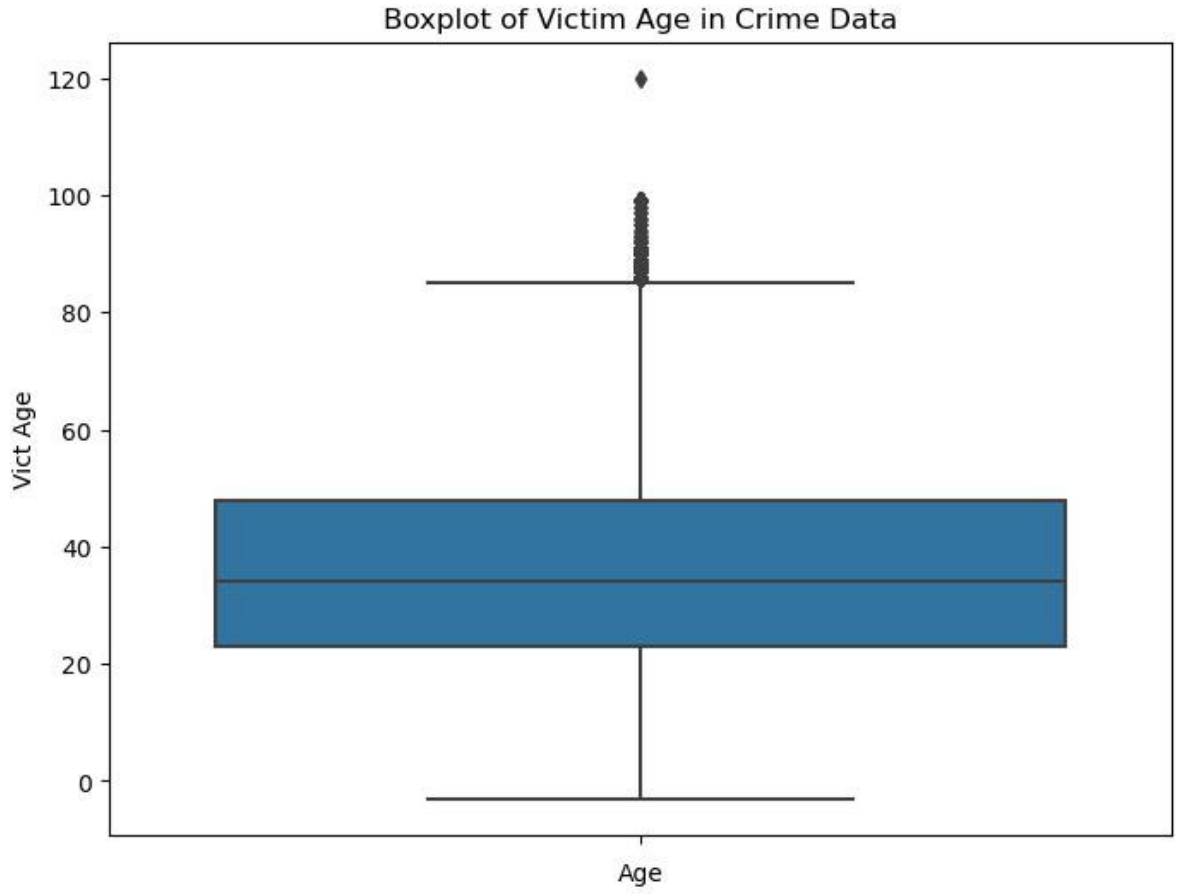
8. Outliers and Anomalies

Box plots are a graphical depiction of numerical data through their quantiles. It is a very simple but effective way to visualize outliers. Think about the lower and upper whiskers as the

boundaries of the data distribution. Any data points that show above or below the whiskers, can be considered outliers or anomalous.

In [61]:

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
sns.boxplot(y=crime_data['Vict Age'].dropna(), orient='v')
plt.title('Boxplot of Victim Age in Crime Data')
plt.xlabel('Age')
plt.show()
```



9. Demographic Factors:

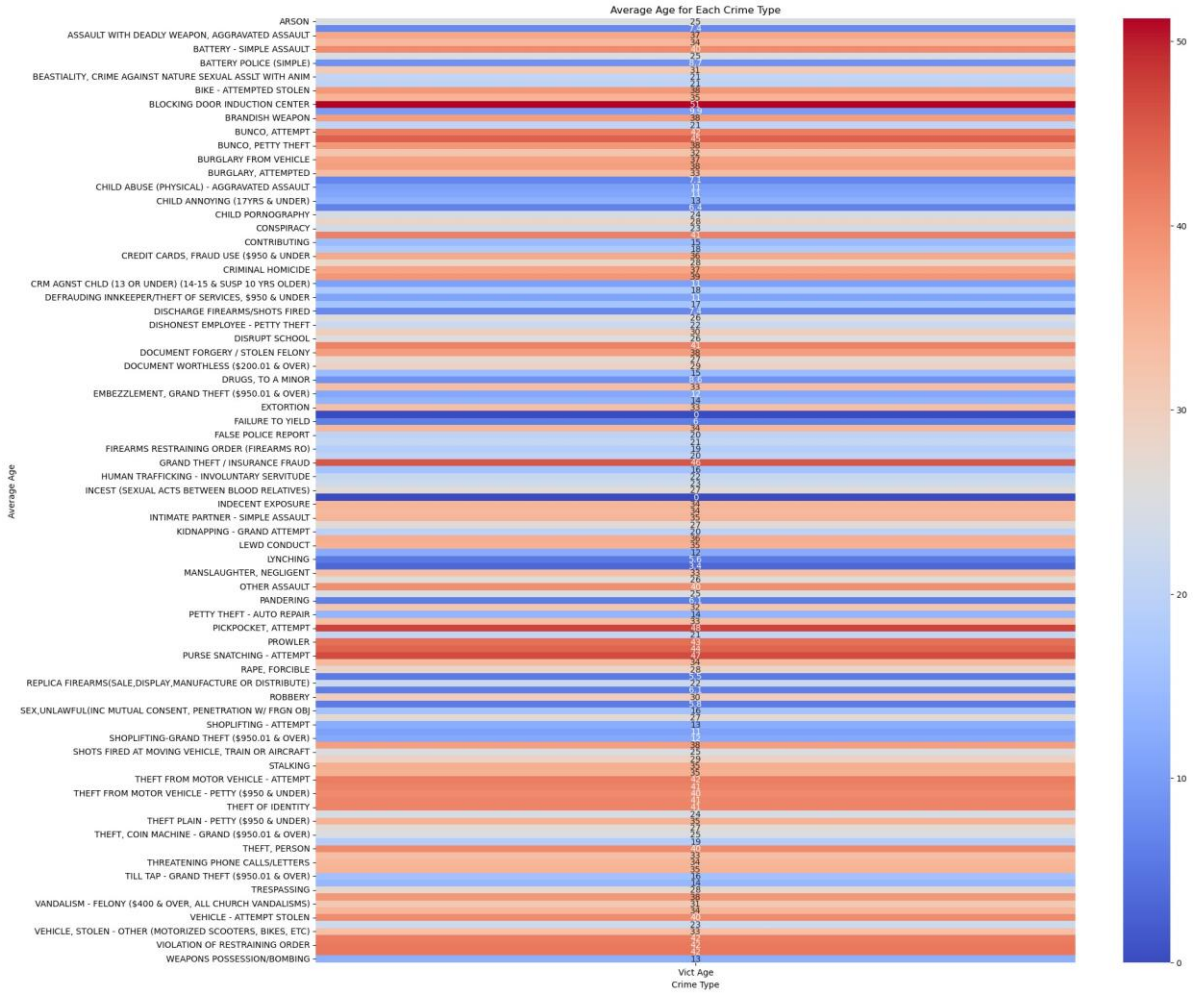
In [62]:
```
import seaborn as sns

print(crime_data['Vict Age'].describe())
print(crime_data['Vict Sex'].value_counts())
```

```
count    702651.000000
mean         34.345109
std          19.761423
min          -3.000000
25%          23.000000
50%          34.000000
75%          48.000000
max         120.000000
Name: Vict Age, dtype: float64
Vict Sex
M          334257
F          297846
X           70159
Unknown       298
H              90
-               1
Name: count, dtype: int64
```

Analyzing Age and Crime Type Correlations

In [63]:

```
import seaborn as sns

age_crime_pivot = crime_data.pivot_table(index='Crm Cd Desc', values='Vict

plt.figure(figsize=(20, 20))
sns.heatmap(age_crime_pivot, annot=True, cmap='coolwarm', cbar=True)
plt.xlabel('Crime Type')
plt.ylabel('Average Age')
plt.title('Average Age for Each Crime Type')
plt.show()
```

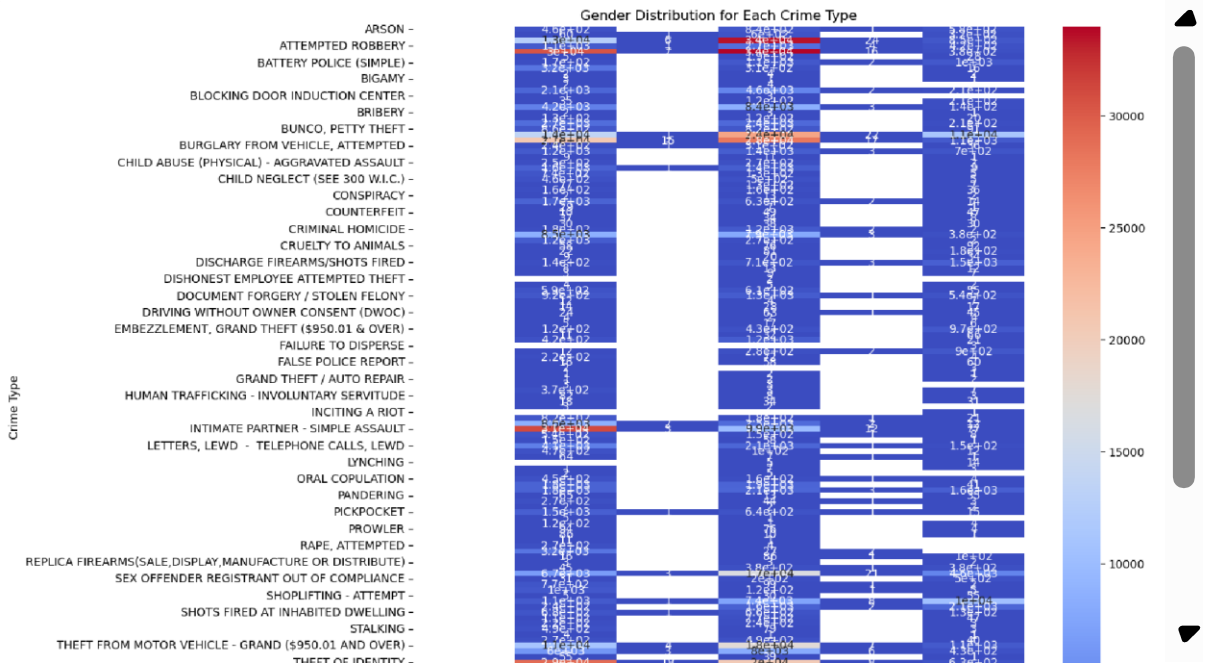


In

Example: Analyzing Gender and Crime Type Correlations

[64]:

```
gender_crime_pivot = crime_data.pivot_table(index='Crm Cd Desc', columns='

plt.figure(figsize=(12, 12))
sns.heatmap(gender_crime_pivot, annot=True, cmap='coolwarm', cbar=True)
plt.xlabel('Gender')
plt.ylabel('Crime Type')
plt.title('Gender Distribution for Each Crime Type')
plt.show()
```



10. Predicting Future Trends:

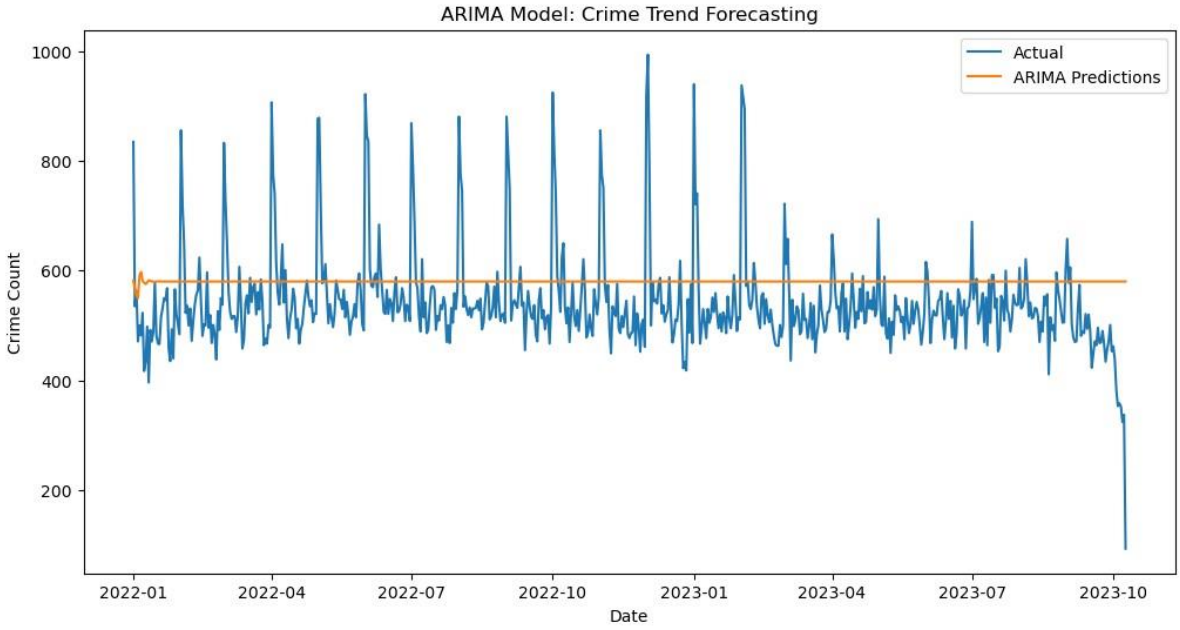
In [65]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
from math import sqrt

date_format = '%Y-%m-%d'

crime_data['DATE OCC'] = pd.to_datetime(crime_data['DATE OCC'])

crime_data.set_index('DATE OCC', inplace=True)

crime_count = crime_data.resample('D').size()

train = crime_count[:'2022-01-01']
test = crime_count['2022-01-01':]

model = ARIMA(train, order=(5,1,0))
model_fit = model.fit()

predictions = model_fit.forecast(steps=len(test))

rmse = sqrt(mean_squared_error(test, predictions))
print('RMSE:', rmse)

plt.figure(figsize=(12, 6))
plt.plot(test.index, test.values, label='Actual')
plt.plot(test.index, predictions, label='ARIMA Predictions')
plt.legend()
plt.xlabel('Date')
plt.ylabel('Crime Count')
plt.title('ARIMA Model: Crime Trend Forecasting')
plt.show()
```

```
C:\Users\prabh\AppData\Local\Temp\ipykernel_16380\3753247279.py:10:
UserWarni ng: Could not infer format, so each element will be parsed
individually, fall ing back to `dateutil`. To ensure parsing is consistent
and as-expected, plea se specify a format.
  crime_data['DATE OCC'] = pd.to_datetime(crime_data['DATE OCC'])
```

In

```
RMSE: 96.32001126993002
```


ARIMA Model: Crime Trend Forecasting
Actual
ARIMA Predictions
Crime Count
1000
800
600
400
200
2022-01
2022-04
2022-07
2022-10
2023-01
2023-04
2023-07
2023-10
Date