In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import folium
from folium.plugins import MarkerCluster
from statsmodels.tsa.arima.model import ARIMA
from pandas.plotting import autocorrelation_plot
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from folium.plugins import FastMarkerCluster
```

Data Acquisition

Check this for additional info of the dataset and carry on with imputations : https://data.lacity.org/Public-Safety/Crime-Data-from-2020-to-Present/2nrs-mtv8/about_data

In [6]:
```
data=pd.read_csv("Crime_Data_from_2020_to_Present.csv")
dataframe=pd.DataFrame(data)
print(dataframe)
```

	DR_NO	Date Rptd	DATE OCC	TIME OCC
0	190326475	03/01/2020 12:00:00 AM	03/01/2020 12:00:00 AM	2130
1	200106753	02/09/2020 12:00:00 AM	02/08/2020 12:00:00 AM	1800
2	200320258	11/11/2020 12:00:00 AM	11/04/2020 12:00:00 AM	1700
3	200907217	05/10/2023 12:00:00 AM	03/10/2020 12:00:00 AM	2037
4	220614831	08/18/2022 12:00:00 AM	08/17/2020 12:00:00 AM	1200
...	...	...	...	...
982633	242011172	08/20/2024 12:00:00 AM	08/17/2024 12:00:00 AM	2300
982634	240710284	07/24/2024 12:00:00 AM	07/23/2024 12:00:00 AM	1400
982635	240104953	01/15/2024 12:00:00 AM	01/15/2024 12:00:00 AM	100
982636	240309674	04/24/2024 12:00:00 AM	04/24/2024 12:00:00 AM	1500
982637	240910892	08/13/2024 12:00:00 AM	08/12/2024 12:00:00 AM	2300

	AREA	AREA NAME	Rpt Dist No	Part 1-2	Crm Cd
0	7	Wilshire	784	1	510
1	1	Central	182	1	330
2	3	Southwest	356	1	480
3	9	Van Nuys	964	1	343
4	6	Hollywood	666	2	354
...	...	...	...	...	...
982633	20	Olympic	2033	1	341
982634	7	Wilshire	788	1	510
982635	1	Central	101	2	745
982636	3	Southwest	358	1	230
982637	9	Van Nuys	914	1	510

	Crm Cd Desc	...	Status
0	VEHICLE - STOLEN	...	AA
1	BURGLARY FROM VEHICLE	...	IC
2	BIKE - STOLEN	...	IC
3	SHOPLIFTING-GRAND THEFT ($950.01 & OVER)	...	IC
4	THEFT OF IDENTITY	...	IC
...	...	...	...
982633	THEFT-GRAND ($950.01 & OVER)EXCPT,GUNS,FOWL,LI...	...	IC
982634	VEHICLE - STOLEN	...	IC
982635	VANDALISM - MISDEAMEANOR ($399 OR UNDER)	...	IC
982636	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT	...	IC
982637	VEHICLE - STOLEN	...	IC

	Status Desc	Crm Cd 1	Crm Cd 2	Crm Cd 3	Crm Cd 4
0	Adult Arrest	510.0	998.0	NaN	NaN
1	Invest Cont	330.0	998.0	NaN	NaN
2	Invest Cont	480.0	NaN	NaN	NaN
3	Invest Cont	343.0	NaN	NaN	NaN
4	Invest Cont	354.0	NaN	NaN	NaN
...	...	...	...	...	...
982633	Invest Cont	341.0	NaN	NaN	NaN
982634	Invest Cont	510.0	NaN	NaN	NaN
982635	Invest Cont	745.0	NaN	NaN	NaN
982636	Invest Cont	230.0	NaN	NaN	NaN
982637	Invest Cont	510.0	NaN	NaN	NaN

	LOCATION
0	1900 S LONGWOOD AV
1	1000 S FLOWER ST

```
2        1400 W  37TH                        ST
3       14000    RIVERSIDE                   DR
4                              1900    TRANSIENT
...                                         ...
982633   3700    WILSHIRE                    BL
982634   4000 W  23RD                        ST
982635   1300 W  SUNSET                      BL
982636           FLOWER                      ST
982637   6900    VESPER                      AV

                          Cross Street      LAT       LON
0                                  NaN  34.0375 -118.3506
1                                  NaN  34.0444 -118.2628
2                                  NaN  34.0210 -118.3002
3                                  NaN  34.1576 -118.4387
4                                  NaN  34.0944 -118.3277
...                                ...      ...       ...
982633                             NaN  34.0617 -118.3066
982634                             NaN  34.0362 -118.3284
982635                             NaN  34.0685 -118.2460
982636  JEFFERSON                   BL  34.0215 -118.2868
982637                             NaN  34.1961 -118.4510

[982638 rows x 28 columns]
```

Data Inspection

In [4]:
```
first_data=dataframe.head()
print(first_data)
```

```
       DR_NO               Date Rptd                DATE OCC  TIME OCC  AREA
\
0  190326475  03/01/2020 12:00:00 AM  03/01/2020 12:00:00 AM      2130     7
1  200106753  02/09/2020 12:00:00 AM  02/08/2020 12:00:00 AM      1800     1
2  200320258  11/11/2020 12:00:00 AM  11/04/2020 12:00:00 AM      1700     3
3  200907217  05/10/2023 12:00:00 AM  03/10/2020 12:00:00 AM      2037     9
4  220614831  08/18/2022 12:00:00 AM  08/17/2020 12:00:00 AM      1200     6

   AREA NAME  Rpt Dist No  Part 1-2  Crm Cd  \
0   Wilshire          784         1     510
1    Central          182         1     330
2  Southwest          356         1     480
3   Van Nuys          964         1     343
4  Hollywood          666         2     354

                              Crm Cd Desc  ... Status   Status Desc  \
0                         VEHICLE - STOLEN  ...     AA  Adult Arrest
1                      BURGLARY FROM VEHICLE  ...     IC   Invest Cont
2                             BIKE - STOLEN  ...     IC   Invest Cont
3  SHOPLIFTING-GRAND THEFT ($950.01 & OVER)  ...     IC   Invest Cont
4                          THEFT OF IDENTITY  ...     IC   Invest Cont

  Crm Cd 1 Crm Cd 2  Crm Cd 3 Crm Cd 4  \
0    510.0    998.0       NaN      NaN
1    330.0    998.0       NaN      NaN
2    480.0      NaN       NaN      NaN
3    343.0      NaN       NaN      NaN
4    354.0      NaN       NaN      NaN

                                  LOCATION Cross Street      LAT       LON
0   1900 S  LONGWOOD                   AV          NaN  34.0375 -118.3506
1   1000 S  FLOWER                    ST          NaN  34.0444 -118.2628
2   1400 W  37TH                      ST          NaN  34.0210 -118.3002
3  14000    RIVERSIDE                 DR          NaN  34.1576 -118.4387
4                          1900    TRANSIENT         NaN  34.0944 -118.3277

[5 rows x 28 columns]
```

In [6]:
```
df_selected_range = dataframe.iloc[:, 0:29]
print(df_selected_range.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
Status Desc        object
Crm Cd 1          float64
Crm Cd 2          float64
Crm Cd 3          float64
Crm Cd 4          float64
LOCATION           object
Cross Street       object
LAT               float64
LON               float64
dtype: object
```

In [7]:
```
for columns in df_selected_range:
    print(columns)
```

```
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
Crm Cd 2
Crm Cd 3
Crm Cd 4
LOCATION
Cross Street
LAT
LON
```

Data Cleaning

In [8]:
```
missing_data=dataframe.isnull().sum()
print(missing_data)
```

```
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          145262
Vict Age              0
Vict Sex         138445
Vict Descent     138456
Premis Cd            14
Premis Desc         585
Weapon Used Cd   656471
Weapon Desc      656471
Status                1
Status Desc           0
Crm Cd 1             11
Crm Cd 2         913763
Crm Cd 3         980327
Crm Cd 4         982574
LOCATION              0
Cross Street     830789
LAT                   0
LON                   0
dtype: int64
```

In [14]:
```
drop_all_missing_data=dataframe.dropna(how="all")
```

In [11]:
```
dataframe['Vict Age']=dataframe['Vict Age'].replace([0], np.nan)
print(dataframe['Vict Age'].isnull().sum())
```

```
259601
```

In [12]:
```
column_to_check = 'Vict Sex'
missing_values = dataframe[column_to_check][dataframe[column_to_check].isnul
if not missing_values.empty:
    print(f"\nMissing values in column '{column_to_check}':")
    print(missing_values)
else:
    print(f"\nNo missing values in column '{column_to_check}'.")
```

```
Missing values in column 'Vict Sex':
13        NaN
23        NaN
26        NaN
27        NaN
33        NaN
          ...
982618    NaN
982621    NaN
982628    NaN
982634    NaN
982637    NaN
Name: Vict Sex, Length: 138445, dtype: object
```

In [13]:
```
duplicates=dataframe.duplicated()
print(duplicates)
```

```
0         False
1         False
2         False
3         False
4         False
          ...
982633    False
982634    False
982635    False
982636    False
982637    False
Length: 982638, dtype: bool
```

In [22]:
```
dataframe['DR_NO'] = dataframe['DR_NO'].astype(int)
dataframe['DATE OCC'] = pd.to_datetime(dataframe['DATE OCC'])
```

In [23]:
```
df_selected_range = dataframe.iloc[:, 0:29]
print(df_selected_range.dtypes)
```

```
DR_NO                      int64
Date Rptd                 object
DATE OCC          datetime64[ns]
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                 float64
Vict Sex                  object
Vict Descent              object
Premis Cd                float64
Premis Desc               object
Weapon Used Cd           float64
Weapon Desc               object
Status                    object
Status Desc               object
Crm Cd 1                 float64
Crm Cd 2                 float64
Crm Cd 3                 float64
Crm Cd 4                 float64
LOCATION                  object
Cross Street              object
LAT                      float64
LON                      float64
dtype: object
```

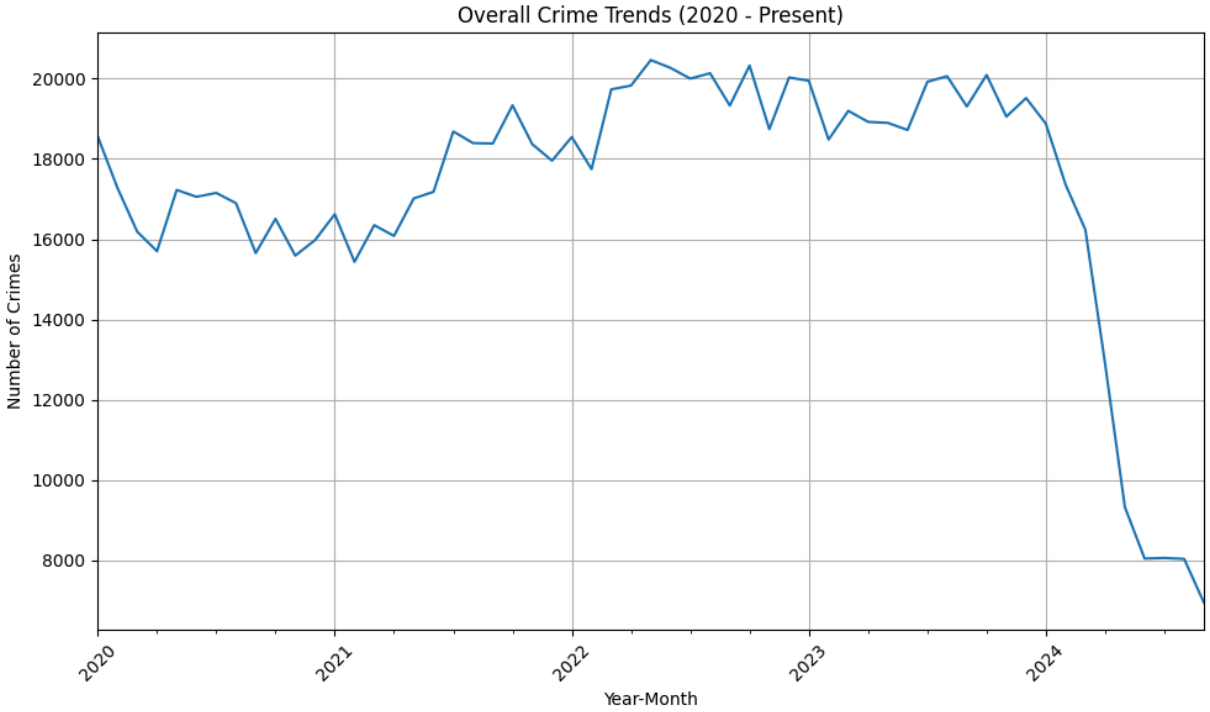
In [17]:
```
dataframe.to_csv("cleaned_csv.csv")
```

Exploratory Data Analysis (EDA):

Visualize overall crime trends from 2020 to the present year.

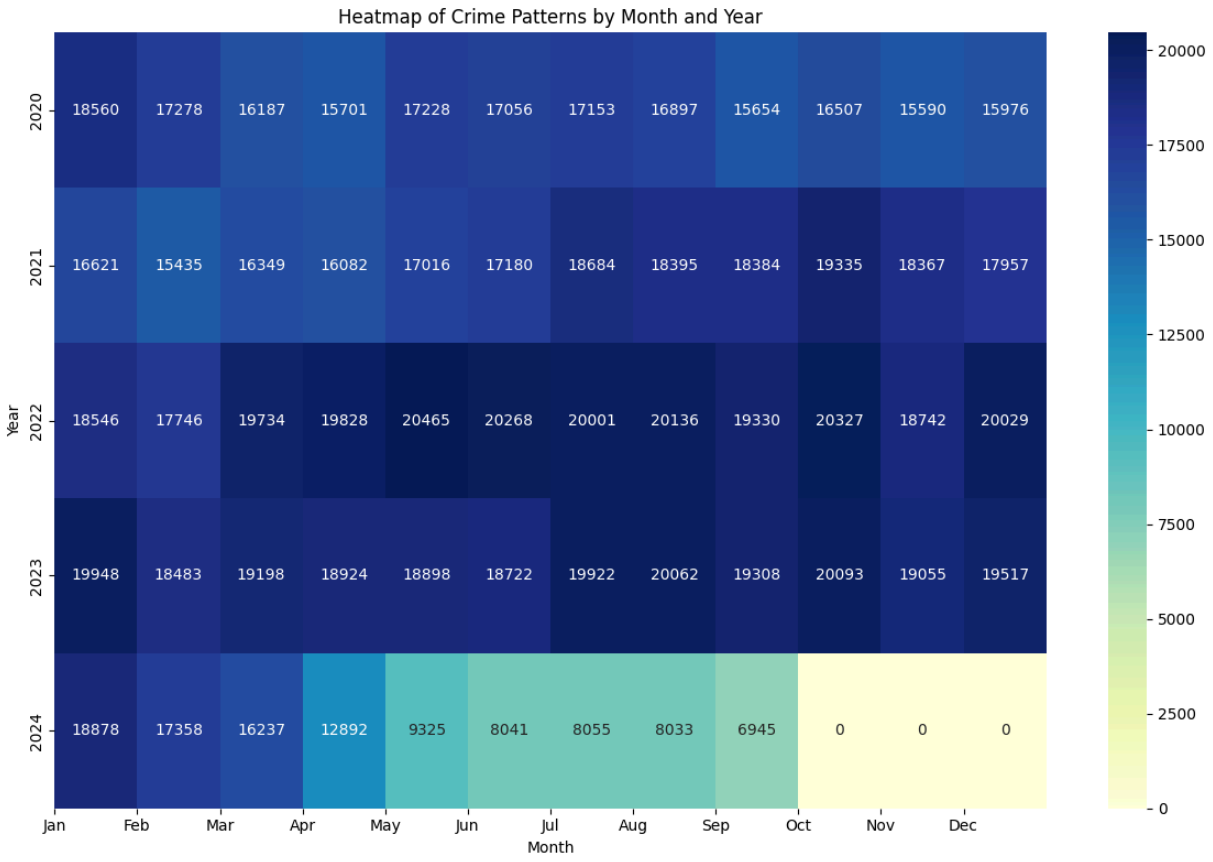
In [25]:
```
df_filtered = dataframe[dataframe['DATE OCC'].dt.year >= 2020]
df_filtered['YearMonth'] = df_filtered['DATE OCC'].dt.to_period('M')
crime_trend = df_filtered.groupby('YearMonth').size()
```

In [27]:
```
plt.figure(figsize=(10, 6))
crime_trend.plot(kind='line')
plt.title('Overall Crime Trends (2020 - Present)')
plt.xlabel('Year-Month')
plt.ylabel('Number of Crimes')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Analyze and visualize seasonal patterns in crime data.

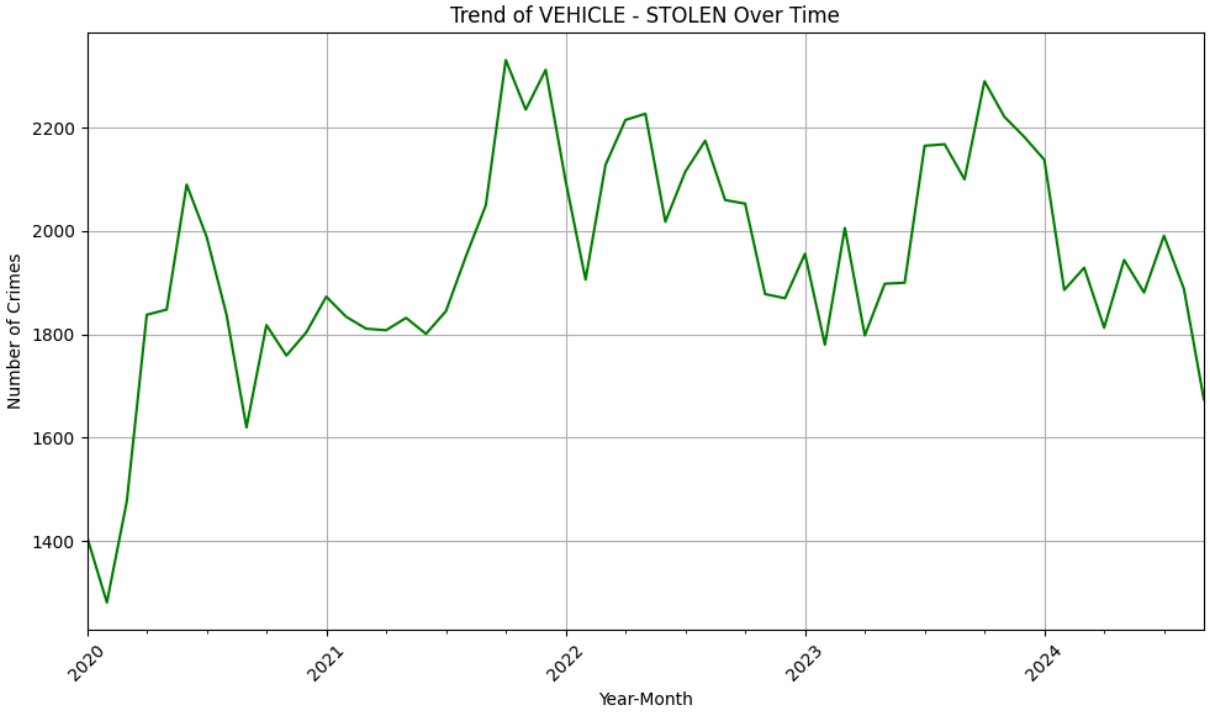
In [30]:
```
# Displays crime frequency across both months and years. Darker cells indica
# Extract year and month
dataframe['Year'] = dataframe['DATE OCC'].dt.year
dataframe['Month'] = dataframe['DATE OCC'].dt.month

# Create a pivot table for heatmap
crime_pivot = dataframe.pivot_table(index='Year', columns='Month', aggfunc='

# Plotting the heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(crime_pivot, cmap='YlGnBu', annot=True, fmt="d")
plt.title('Heatmap of Crime Patterns by Month and Year')
plt.xlabel('Month')
plt.ylabel('Year')
plt.xticks(ticks=range(12), labels=['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun'
plt.tight_layout()
plt.show()
```



Identify the most common type of crime and its trends over time.

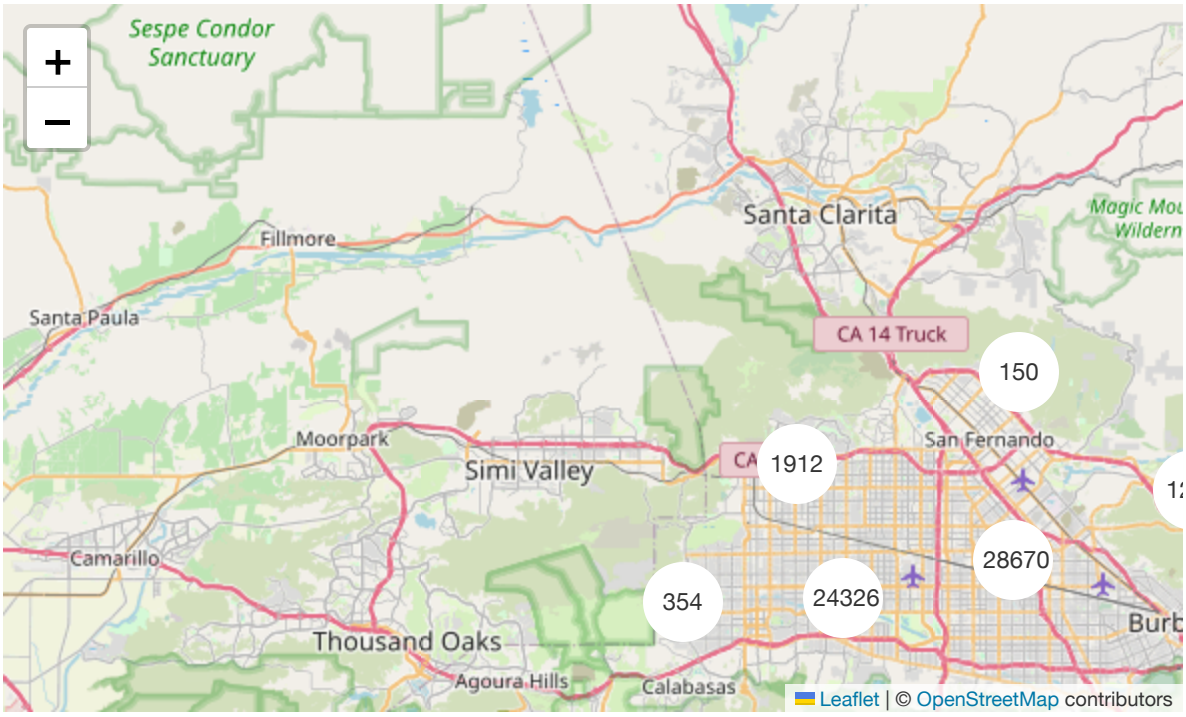
In [36]:
```
crime_counts = dataframe['Crm Cd Desc'].value_counts()
most_common_crime = crime_counts.idxmax()
print("The most common crime is:", most_common_crime)

# Filter the dataset for the most common crime type
df_most_common = dataframe[dataframe['Crm Cd Desc'] == most_common_crime].co

# Convert 'DATE OCC' to datetime and extract year/month
df_most_common.loc[:, 'DATE OCC'] = pd.to_datetime(df_most_common['DATE OCC'
df_most_common['YearMonth'] = df_most_common['DATE OCC'].dt.to_period('M')

# Group by year and month to count the occurrences of the most common crime
crime_trend = df_most_common.groupby('YearMonth').size()

# Plotting the trend over time
plt.figure(figsize=(10, 6))
crime_trend.plot(kind='line', color='green')
plt.title(f'Trend of {most_common_crime} Over Time')
plt.xlabel('Year-Month')
plt.ylabel('Number of Crimes')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
The most common crime is: VEHICLE - STOLEN
```



Investigate if there are any notable differences in crime rates between regions or cities.

In [34]:
```
dataframe['LAT'] = pd.to_numeric(dataframe['LAT'], errors='coerce')
dataframe['LON'] = pd.to_numeric(dataframe['LON'], errors='coerce')

# Drop rows with NaN values in LAT and LON after conversion
dataframe = dataframe.dropna(subset=['LAT', 'LON'])

# Sample the data if it's large
sample_size = 0.1
dataframe_sampled = dataframe.sample(frac=sample_size, random_state=1)

# Create the map centered around the mean latitude and longitude
crime_map = folium.Map(location=[dataframe_sampled['LAT'].mean(), dataframe_

# Prepare data for the cluster
locations = dataframe_sampled[['LAT', 'LON']].values.tolist()  # Get only la

# Create FastMarkerCluster and add data
marker_cluster = FastMarkerCluster(data=locations).add_to(crime_map)

for index, row in dataframe_sampled.iterrows():
    folium.Marker([row['LAT'], row['LON']], popup=row['Crm Cd Desc']).add_to

display(crime_map)
```

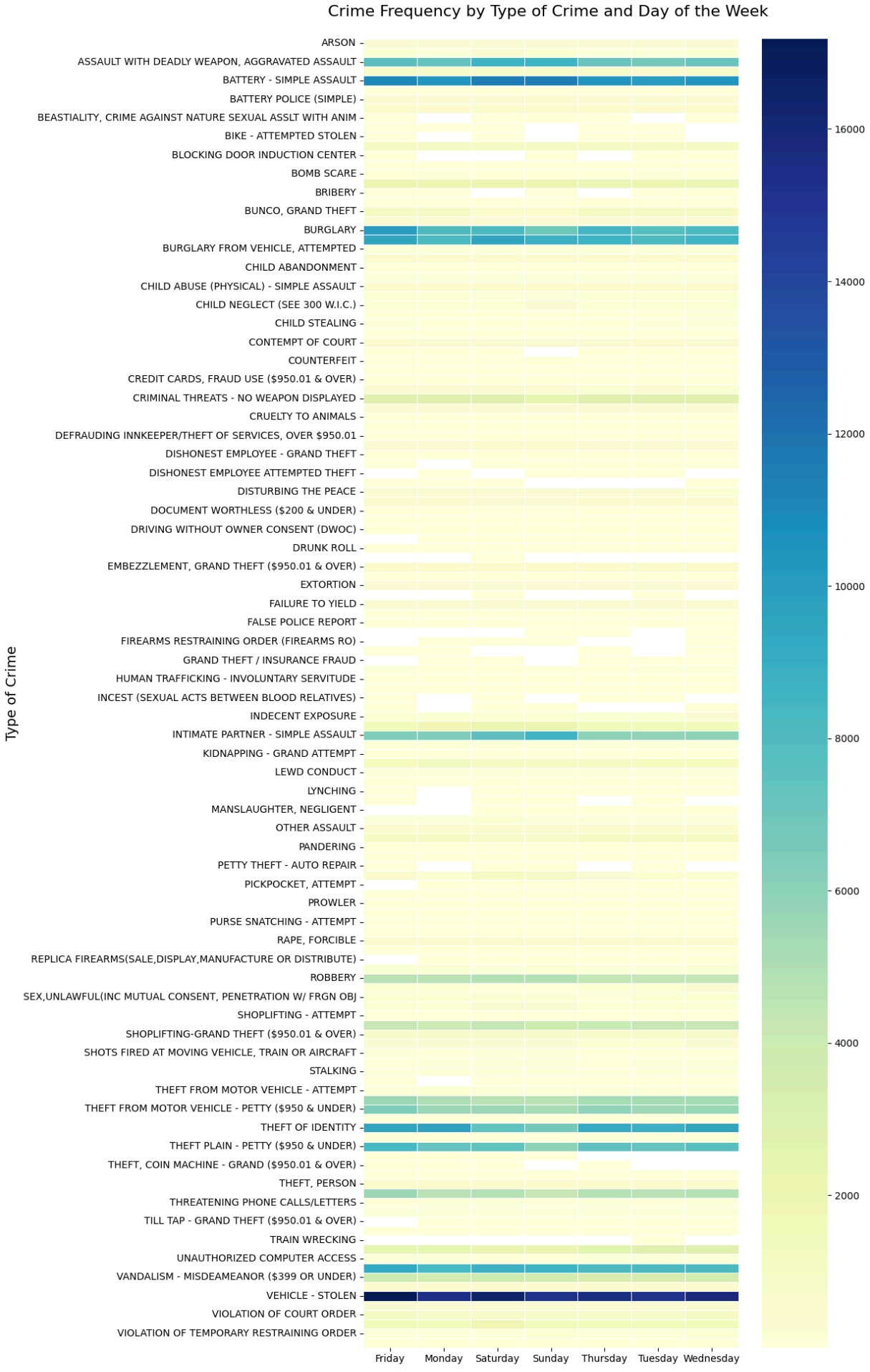


Explore correlations between economic factors (if available) and crime rates.

In [5]:
```
# No column named INCOME in the dataset
```

Analyze the relationship between the day of the week and the frequency of certain types of crimes.

In [10]:
```
dataframe['DATE OCC'] = pd.to_datetime(dataframe['DATE OCC'], errors='coerce
dataframe['Day of Week'] = dataframe['DATE OCC'].dt.dayofweek

days = {0: 'Monday', 1: 'Tuesday', 2: 'Wednesday', 3: 'Thursday', 4: 'Friday
dataframe['Day of Week'] = dataframe['Day of Week'].map(days)

# Group by day of the week and crime type
crime_by_day = dataframe.groupby(['Day of Week', 'Crm Cd Desc'])['DR_NO'].co

# Pivot the table to make 'Crm Cd Desc' the index and 'Day of Week' the colu
crime_by_day_pivot = crime_by_day.pivot(index='Crm Cd Desc', columns='Day of

# Plot the heatmap
plt.figure(figsize=(12, 20))  # Adjust figsize for vertical display
sns.heatmap(crime_by_day_pivot, cmap='YlGnBu', annot=False, linewidths=.5)
plt.title('Crime Frequency by Type of Crime and Day of the Week \n', fontsiz
plt.xlabel('\n Day of the Week', fontsize=14)
plt.ylabel('Type of Crime', fontsize=14)

plt.tight_layout()
plt.show()
```


Crime Frequency by Type of Crime and Day of the Week
Type of Crime
ARSON
ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT
BATTERY - SIMPLE ASSAULT
BATTERY POLICE (SIMPLE)
BEASTIALITY, CRIME AGAINST NATURE SEXUAL ASSLT WITH ANIM
BIKE - ATTEMPTED STOLEN
BLOCKING DOOR INDUCTION CENTER
BOMB SCARE
BRIBERY
BUNCO, GRAND THEFT
BURGLARY
BURGLARY FROM VEHICLE, ATTEMPTED
CHILD ABANDONMENT
CHILD ABUSE (PHYSICAL) - SIMPLE ASSAULT
CHILD NEGLECT (SEE 300 W.I.C.)
CHILD STEALING
CONTEMPT OF COURT
COUNTERFEIT
CREDIT CARDS, FRAUD USE ($950.01 & OVER)
CRIMINAL THREATS - NO WEAPON DISPLAYED
CRUELTY TO ANIMALS
DEFRAUDING INNKEEPER/THEFT OF SERVICES, OVER $950.01
DISHONEST EMPLOYEE - GRAND THEFT
DISHONEST EMPLOYEE ATTEMPTED THEFT
DISTURBING THE PEACE
DOCUMENT WORTHLESS ($200 & UNDER)
DRIVING WITHOUT OWNER CONSENT (DWOC)
DRUNK ROLL
EMBEZZLEMENT, GRAND THEFT ($950.01 & OVER)
EXTORTION
FAILURE TO YIELD
FALSE POLICE REPORT
FIREARMS RESTRAINING ORDER (FIREARMS RO)
GRAND THEFT / INSURANCE FRAUD
HUMAN TRAFFICKING - INVOLUNTARY SERVITUDE
INCEST (SEXUAL ACTS BETWEEN BLOOD RELATIVES)
INDECENT EXPOSURE
INTIMATE PARTNER - SIMPLE ASSAULT
KIDNAPPING - GRAND ATTEMPT
LEWD CONDUCT
LYNCHING
MANSLAUGHTER, NEGLIGENT
OTHER ASSAULT
PANDERING
PETTY THEFT - AUTO REPAIR
PICKPOCKET, ATTEMPT
PROWLER
PURSE SNATCHING - ATTEMPT
RAPE, FORCIBLE
REPLICA FIREARMS(SALE,DISPLAY,MANUFACTURE OR DISTRIBUTE)
ROBBERY
SEX,UNLAWFUL(INC MUTUAL CONSENT, PENETRATION W/ FRGN OBJ
SHOPLIFTING - ATTEMPT
SHOPLIFTING-GRAND THEFT ($950.01 & OVER)
SHOTS FIRED AT MOVING VEHICLE, TRAIN OR AIRCRAFT
STALKING
THEFT FROM MOTOR VEHICLE - ATTEMPT
THEFT FROM MOTOR VEHICLE - PETTY ($950 & UNDER)
THEFT OF IDENTITY
THEFT PLAIN - PETTY ($950 & UNDER)
THEFT, COIN MACHINE - GRAND ($950.01 & OVER)
THEFT, PERSON
THREATENING PHONE CALLS/LETTERS
TILL TAP - GRAND THEFT ($950.01 & OVER)
TRAIN WRECKING
UNAUTHORIZED COMPUTER ACCESS
VANDALISM - MISDEAMEANOR ($399 OR UNDER)
VEHICLE - STOLEN
VIOLATION OF COURT ORDER
VIOLATION OF TEMPORARY RESTRAINING ORDER
Friday
Monday
Saturday
Sunday
Thursday
Tuesday
Wednesday
16000
14000
12000
10000
8000
6000
4000
2000

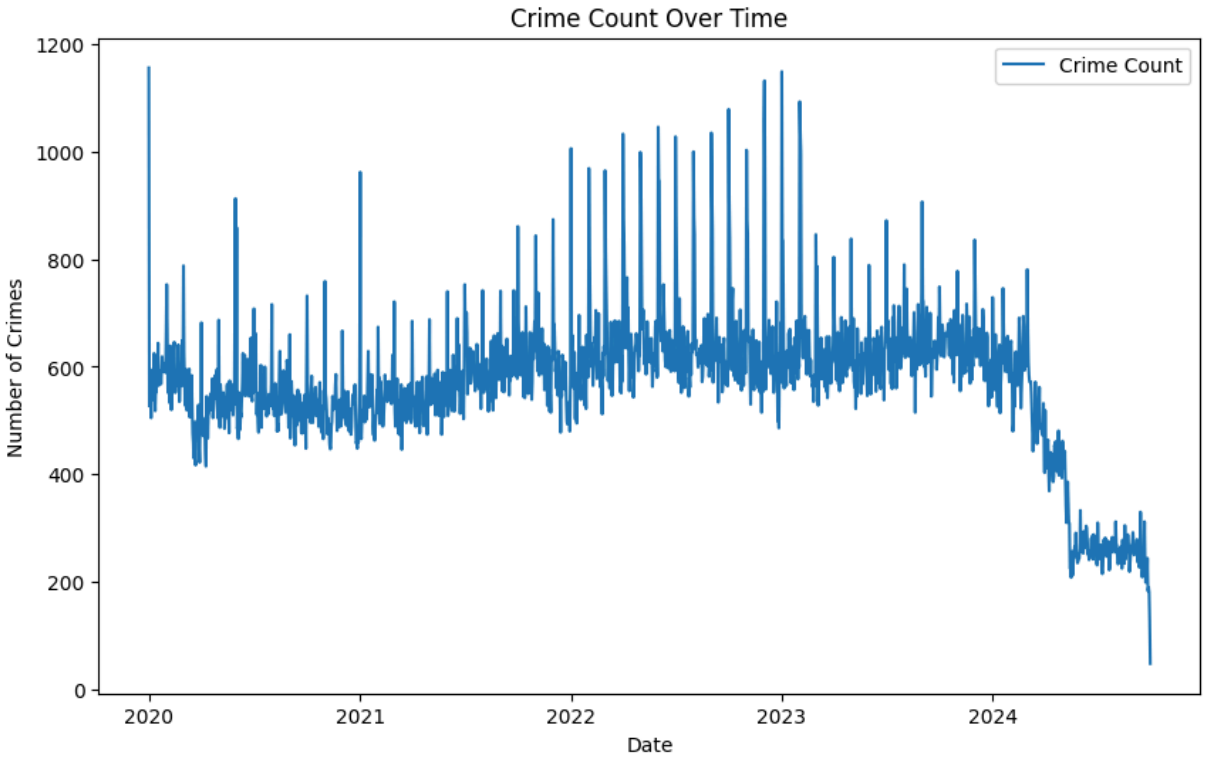
Day of the Week

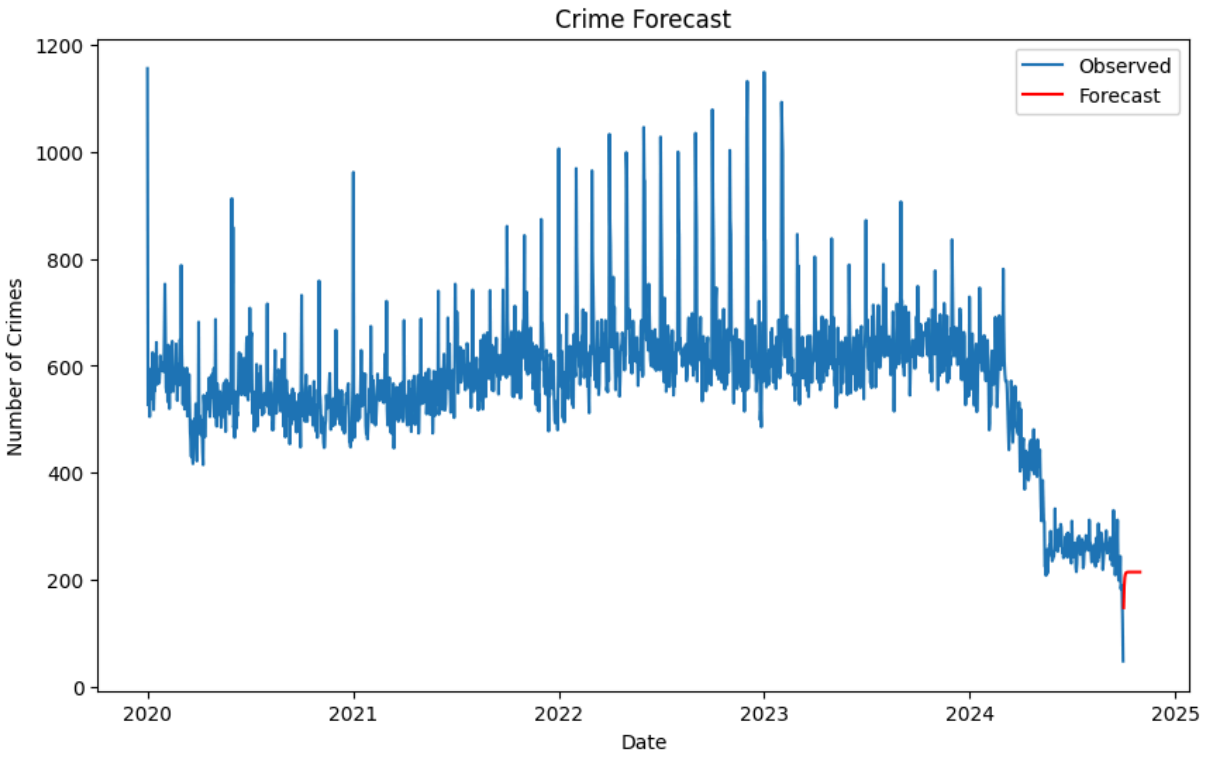
Investigate any impact of significant events or policy changes on crime rates.

In [1]:
```
# There are no significant events or policy changes mentioned in the dataset
```

Use predictive modeling techniques (e.g., time series forecasting) to predict future crime trends.

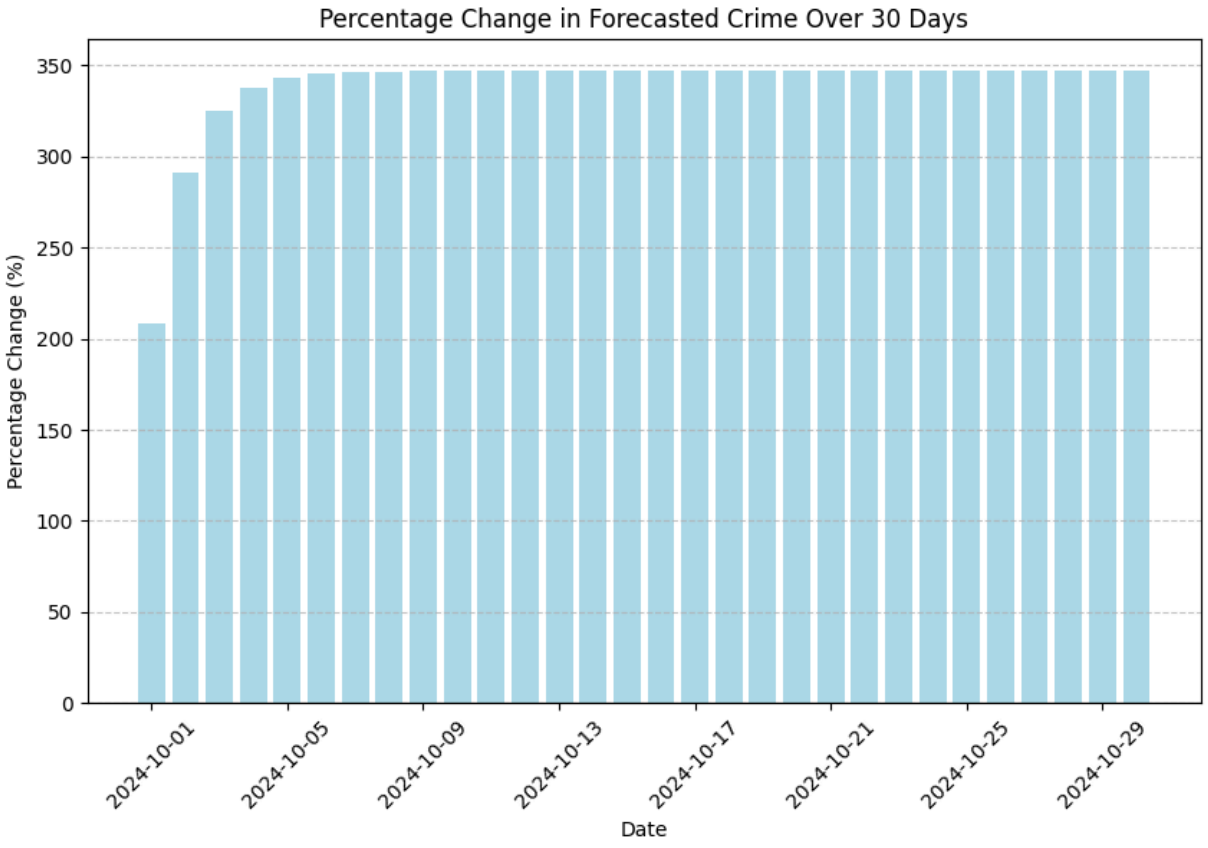
In [20]:
```
df = pd.read_csv("Crime_Data_from_2020_to_Present.csv", parse_dates=['DATE O

# Aggregate data by day (or by month if you need to) and take the crime coun
crime_series = df.resample('D').size()

# Visualize the time series data
plt.figure(figsize=(10,6))
plt.plot(crime_series, label='Crime Count')
plt.title('Crime Count Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.legend()
plt.show()

# Check for stationarity and differencing
crime_series_diff = crime_series.diff().dropna()

model = ARIMA(crime_series, order=(1, 1, 1))

# Fit the model
model_fit = model.fit()

# Forecasting future crime trends
forecast = model_fit.forecast(steps=30)

# Plot forecasted values
plt.figure(figsize=(10,6))
plt.plot(crime_series, label='Observed')
plt.plot(forecast, label='Forecast', color='red')
plt.title('Crime Forecast')
plt.xlabel('Date')
plt.ylabel('Number of Crimes')
plt.legend()
plt.show()
```





In [37]:
```
df = pd.read_csv("Crime_Data_from_2020_to_Present.csv", parse_dates=['DATE O

# Aggregate data by day and count crimes
crime_series = df.resample('D').size()

# Fit the ARIMA model
model = ARIMA(crime_series, order=(1, 1, 1))
model_fit = model.fit()
```

```
# Forecast future crime trends for the next 30 days
forecast_steps = 30
forecast = model_fit.forecast(steps=forecast_steps)

# Calculate percentage change relative to the last observed value
last_observed_value = crime_series.iloc[-1]
forecast_percent_change = ((forecast - last_observed_value) / last_observed_

# Prepare data for bar chart
forecast_dates = pd.date_range(start=crime_series.index[-1] + pd.Timedelta(d
forecast_percent_change_df = pd.DataFrame({'Date': forecast_dates, 'Percent

# Plot the bar chart
plt.figure(figsize=(10, 6))
plt.bar(forecast_percent_change_df['Date'], forecast_percent_change_df['Perc
plt.axhline(0, color='black', linewidth=0.8)
plt.title('Percentage Change in Forecasted Crime Over 30 Days')
plt.xlabel('Date')
plt.ylabel('Percentage Change (%)')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



Out[37]:
```
"plt.figure(figsize=(10, 6))\nplt.plot(crime_series, label='Observed', colo
r='lightblue')\nplt.plot(forecast_dates, forecast, label='Forecast', color
='red')\nplt.title('Crime Forecast with Observed Data')\nplt.xlabel('Date')
\nplt.ylabel('Number of Crimes')\nplt.legend()\nplt.show()"
```