```
'''Crime Trend Predictions Using ARIMA and VAR Models
    ------------------------------------------------
Overview
In this analysis, we employ ARIMA and VAR models to forecast crime trends across India.
ARIMA focuses on predicting future crime rates based on historical data for individual states or districts,
while VAR captures the interdependencies among different crime categories, offering both state-level and national :
Together, these models provide a robust framework for understanding and projecting crime trends, aiding in data-dr:

Predicting Future Crimes Using ARIMA (2014-2023)
------------------------------------------------
Overview
In this section, we employ the ARIMA (AutoRegressive Integrated Moving Average) model to forecast crime trends for

ARIMA Components
----------------
AutoRegression (AR): Models the relationship between an observation and its lagged values.
Integrated (I): Applies differencing to make the time series stationary.
Moving Average (MA): Models the relationship between an observation and the residual errors from a lagged moving a
'''

'''
Preprocessing: Stationarity Check and Differencing
To ensure accurate forecasts, we performed the following preprocessing steps:

1.Stationarity Check:
Conducted the Augmented Dickey-Fuller (ADF) test to assess stationarity.
Observed that raw crime data exhibited non-stationary behavior, with trends and seasonality affecting the series.

2.Differencing:
Applied differencing techniques to stabilize the mean and remove trends, transforming the data into a stationary t:
Ensured that the differenced series passed the stationarity test with consistent mean and variance.'''

import pandas as pd
from sklearn.preprocessing import LabelEncoder

data = pd.read_csv('crimes_cleaned.csv')

# Create backup columns to store the original 'STATE/UT' and 'DISTRICT' values
data['STATE/UT_original'] = data['STATE/UT']
data['DISTRICT_original'] = data['DISTRICT']



# Initialize the label encoder to convert categorical data into numeric labels
label_encoder = LabelEncoder()

# Encode 'STATE/UT' and 'DISTRICT' columns with numeric labels
data['STATE/UT'] = label_encoder.fit_transform(data['STATE/UT'])
data['DISTRICT'] = label_encoder.fit_transform(data['DISTRICT'])

# Configure pandas to display all rows (avoids truncation of output)
pd.set_option('display.max_rows', None)

# Print the comparison of original and encoded values for 'STATE/UT' and 'DISTRICT'
print("Original vs. Encoded Values for STATE/UT and DISTRICT:")
print(data[['STATE/UT_original', 'STATE/UT', 'DISTRICT_original', 'DISTRICT']])

# Reset the display options to default (useful after the operation)
pd.reset_option('display.max_rows')
```

```
Original vs. Encoded Values for STATE/UT and DISTRICT:
        STATE/UT_original  STATE/UT     DISTRICT_original  DISTRICT
0          ANDHRA PRADESH         2              ADILABAD         3
1          ANDHRA PRADESH         2             ANANTAPUR        31
2          ANDHRA PRADESH         2              CHITTOOR       157
3          ANDHRA PRADESH         2              CUDDAPAH       175
4          ANDHRA PRADESH         2             CYBERABAD       177
5          ANDHRA PRADESH         2         EAST GODAVARI       225
```

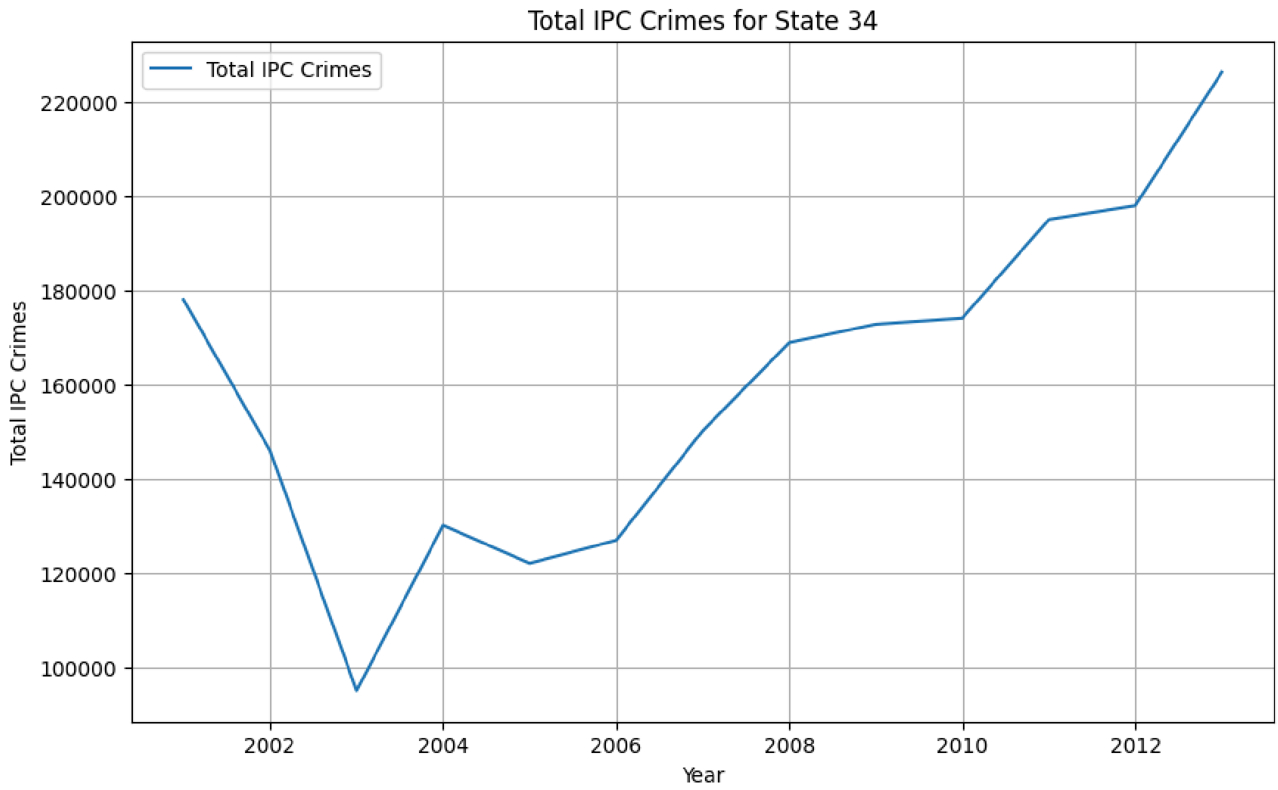
```
6        ANDHRA PRADESH          2        GUNTAKAL RLY.       284
7        ANDHRA PRADESH          2              GUNTUR        285
8        ANDHRA PRADESH          2         GUNTUR URBAN       286
9        ANDHRA PRADESH          2       HYDERABAD CITY       313
10       ANDHRA PRADESH          2          KARIMNAGAR        398
11       ANDHRA PRADESH          2             KHAMMAM        412
12       ANDHRA PRADESH          2             KRISHNA        452
13       ANDHRA PRADESH          2             KURNOOL        457
14       ANDHRA PRADESH          2        MAHABOOBNAGAR       484
15       ANDHRA PRADESH          2               MEDAK        504
16       ANDHRA PRADESH          2            NALGONDA        541
17       ANDHRA PRADESH          2             NELLORE        557
18       ANDHRA PRADESH          2           NIZAMABAD        561
19       ANDHRA PRADESH          2           PRAKASHAM        602
20       ANDHRA PRADESH          2         RAJAHMUNDRY        626
21       ANDHRA PRADESH          2         RANGA REDDY        640
22       ANDHRA PRADESH          2    SECUNDERABAD RLY.       678
23       ANDHRA PRADESH          2          SRIKAKULAM        730
24       ANDHRA PRADESH          2      TIRUPATHI URBAN       768
25       ANDHRA PRADESH          2      VIJAYAWADA CITY       803
26       ANDHRA PRADESH          2      VIJAYAWADA RLY.       804
27       ANDHRA PRADESH          2        VISAKHA RURAL       808
28       ANDHRA PRADESH          2        VISAKHAPATNAM       809
29       ANDHRA PRADESH          2         VIZIANAGARAM       810
30       ANDHRA PRADESH          2            WARANGAL        814
31       ANDHRA PRADESH          2       WARANGAL URBAN       815
32       ANDHRA PRADESH          2        WEST GODAVARI       820
33    ARUNACHAL PRADESH          3               ANJAW         35
34    ARUNACHAL PRADESH          3           CHANGLANG        145
35    ARUNACHAL PRADESH          3        DIBANG VALLEY       213
36    ARUNACHAL PRADESH          3            K/KUMEY         375
37    ARUNACHAL PRADESH          3          KAMENG EAST       379
38    ARUNACHAL PRADESH          3          KAMENG WEST       380
39    ARUNACHAL PRADESH          3               LOHIT        473
40    ARUNACHAL PRADESH          3            LONGDING        474
41    ARUNACHAL PRADESH          3          PAPUM PARE        583
42    ARUNACHAL PRADESH          3               RURAL        653
43    ARUNACHAL PRADESH          3          SIANG EAST        694
44    ARUNACHAL PRADESH          3         SIANG UPPER        695
45    ARUNACHAL PRADESH          3          SIANG WEST        696
46    ARUNACHAL PRADESH          3      SUBANSIRI LOWER       736
47    ARUNACHAL PRADESH          3      SUBANSIRI UPPER       737
48    ARUNACHAL PRADESH          3              TAWANG        749
49    ARUNACHAL PRADESH          3               TIRAP        767
50    ARUNACHAL PRADESH          3   UPPER DIBANG VALLEY      791
51                ASSAM          4               BAKSA         55
52                ASSAM          4             BARPETA         76
53                ASSAM          4                BIEO        106
54                ASSAM          4          BONGAIGAON        117
```

```
'''Customizing Predictions for a Specific State
Select a code from the list of states provided and edit it in the code.
The ARIMA model will generate predictions for the chosen state, displaying the forecasted crime trends for the yea
This customization allows you to focus on the state of your interest, providing localized insights into crime tren
```

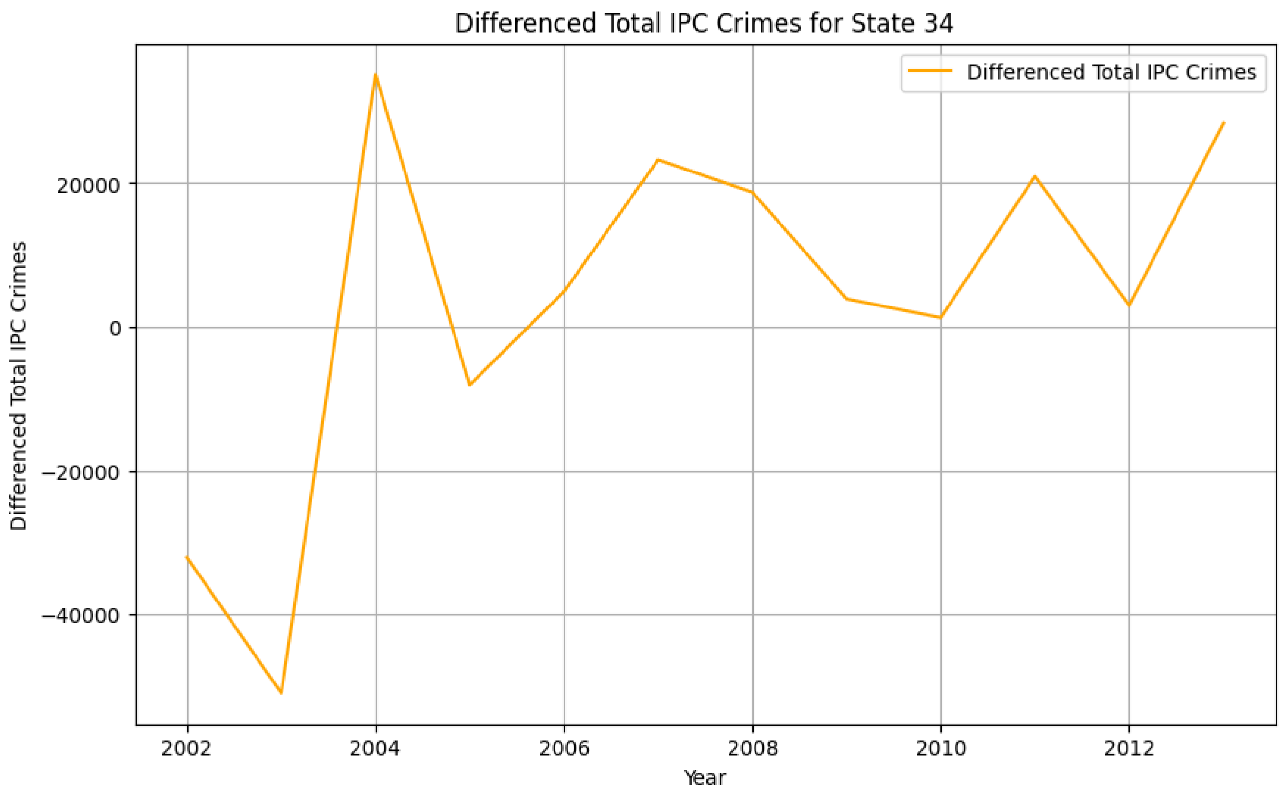
```
import statsmodels.api as sm
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
import pandas as pd

# Select the state and prepare the data
state_selected = 34  # Plug in the STATE code from the above list to get the predictions

arima_data = data[data['STATE/UT'] == state_selected].groupby('YEAR')['TOTAL IPC CRIMES'].sum()

# Plot the original data
plt.figure(figsize=(10, 6))
plt.plot(arima_data, label='Total IPC Crimes')
plt.title(f"Total IPC Crimes for State {state_selected}")
plt.xlabel('Year')
plt.ylabel('Total IPC Crimes')
plt.grid(True)
```

```
plt.legend()
plt.show()

# Perform Augmented Dickey-Fuller (ADF) test to check for stationarity
adf_test = adfuller(arima_data)
print(f"ADF Statistic: {adf_test[0]}")
print(f"p-value: {adf_test[1]}")

# Check if the data is stationary
if adf_test[1] > 0.05:
    print("Data is non-stationary. Differencing will be applied.")

    # First differencing
    arima_data_diff = arima_data.diff().dropna()  # First differencing

    # Check stationarity again after differencing
    adf_test_diff = adfuller(arima_data_diff)
    print(f"ADF Statistic (Differenced): {adf_test_diff[0]}")
    print(f"p-value (Differenced): {adf_test_diff[1]}")

    if adf_test_diff[1] > 0.05:
        print("Still non-stationary after differencing. Further differencing required.")
        # Second differencing if still non-stationary
        arima_data_diff = arima_data_diff.diff().dropna()
else:
    print("Data is already stationary.")

# Plot the differenced data
plt.figure(figsize=(10, 6))
plt.plot(arima_data_diff, label='Differenced Total IPC Crimes', color='orange')
plt.title(f"Differenced Total IPC Crimes for State {state_selected}")
plt.xlabel('Year')
plt.ylabel('Differenced Total IPC Crimes')
plt.grid(True)
plt.legend()
plt.show()



model = sm.tsa.ARIMA(arima_data_diff, order=(1, 0, 1))  # Adjusted for differenced data
model_fit = model.fit()

# Print the summary of the ARIMA model
print(model_fit.summary())

# Make predictions
start_year = arima_data.index[-1] + 1  # The year after the last data point
arima_pred = model_fit.forecast(steps=10)  # Forecasting for the next 10 years
predicted_years = list(range(start_year, start_year + len(arima_pred)))

# Create a DataFrame to store the predictions
prediction_df = pd.DataFrame({
    'Year': predicted_years,
    'Prediction': arima_pred
})

# Print predictions
print(f"ARIMA Predictions for State {state_selected}:")
print(prediction_df)

# Plotting the predictions
plt.figure(figsize=(10, 6))
plt.plot(prediction_df['Year'], prediction_df['Prediction'], marker='o', color='b', linestyle='-', label='ARIMA Pr
plt.title(f"ARIMA Predictions for State {state_selected}")
plt.xlabel('Year')
plt.ylabel('Predicted Total IPC Crimes')
plt.grid(True)
plt.legend()
plt.show()
```

```
# Ensure the correct index length when saving to CSV
# The differenced data will have one fewer entry, so adjust for that
corrected_data = pd.DataFrame({
    'Year': arima_data.index[1:],  # Adjust to match the length of differenced data
    'Differenced Total IPC Crimes': arima_data_diff
})

# Save the differenced data and predictions to a CSV file
corrected_data.to_csv('corrected_data.csv', index=False)

# Save predictions to CSV
prediction_df.to_csv('arima_predictions.csv', index=False)
```

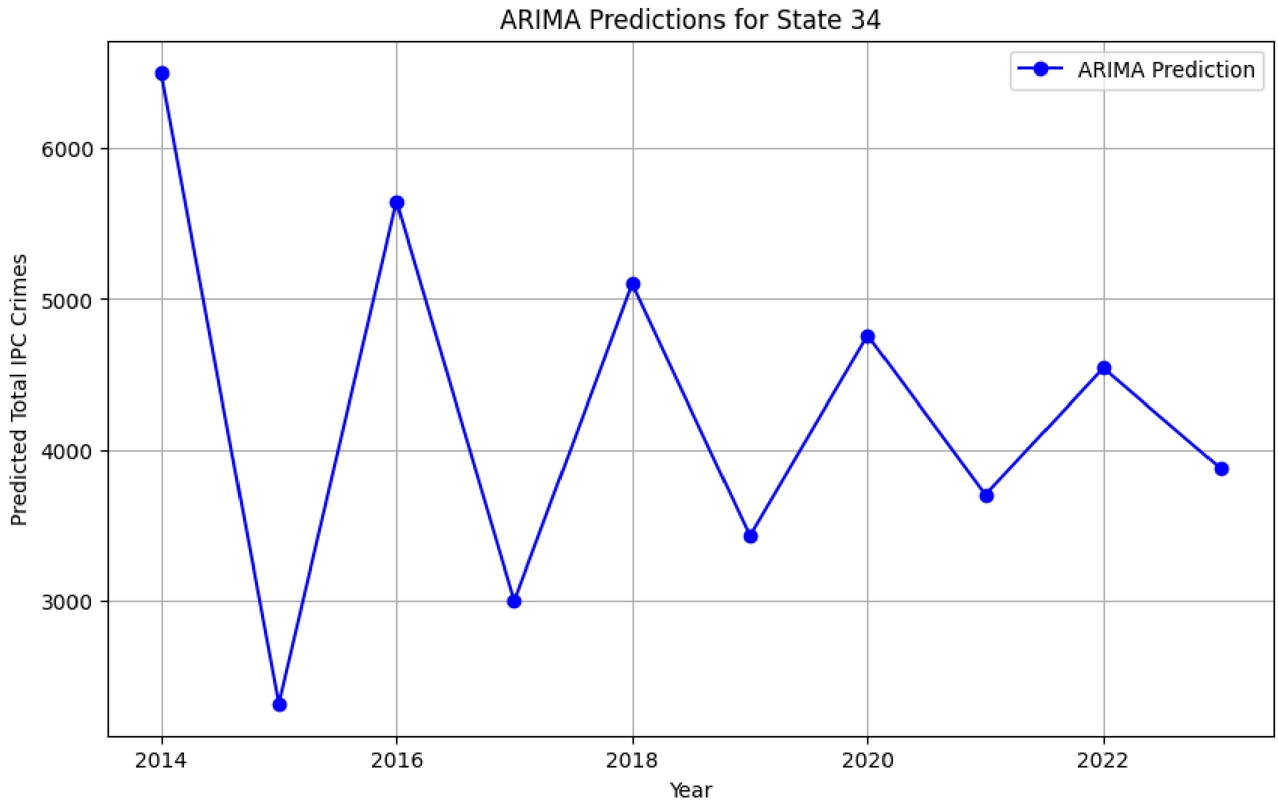



```
ADF Statistic: -0.46121969756634557
p-value: 0.8993646419408896
Data is non-stationary. Differencing will be applied.
ADF Statistic (Differenced): -4.536706706766679
p-value (Differenced): 0.00016869394190484366
```




```
/usr/local/lib/python3.11/dist-packages/statsmodels/tsa/base/tsa_model.py:473: ValueWarning: An unsupporte
  self._init_dates(dates, freq)
/usr/local/lib/python3.11/dist-packages/statsmodels/tsa/base/tsa_model.py:473: ValueWarning: An unsupporte
  self._init_dates(dates, freq)
/usr/local/lib/python3.11/dist-packages/statsmodels/tsa/base/tsa_model.py:473: ValueWarning: An unsupporte
  self._init_dates(dates, freq)
/usr/local/lib/python3.11/dist-packages/statsmodels/tsa/base/tsa_model.py:837: ValueWarning: No supported
  return get_prediction_index(
/usr/local/lib/python3.11/dist-packages/statsmodels/tsa/base/tsa_model.py:837: FutureWarning: No supportec
  return get_prediction_index(
                               SARIMAX Results
==============================================================================
```

```
Dep. Variable:        TOTAL IPC CRIMES   No. Observations:                   12
Model:                  ARIMA(1, 0, 1)   Log Likelihood                -137.757
Date:                 Wed, 09 Apr 2025   AIC                            283.514
Time:                         13:14:49   BIC                            285.453
Sample:                              0   HQIC                           282.796
                                  - 12
Covariance Type:                   opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const       4169.9176   9264.304      0.450      0.653    -1.4e+04    2.23e+04
ar.L1         -0.7949      0.434     -1.830      0.067      -1.646       0.057
ma.L1          1.0000      0.614      1.629      0.103      -0.203       2.203
sigma2      5.098e+08   5.52e-09   9.23e+16      0.000     5.1e+08     5.1e+08
===================================================================================
Ljung-Box (L1) (Q):                   0.06   Jarque-Bera (JB):                 1.28
Prob(Q):                              0.81   Prob(JB):                         0.53
Heteroskedasticity (H):               0.13   Skew:                            -0.79
Prob(H) (two-sided):                  0.07   Kurtosis:                         2.79
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
[2] Covariance matrix is singular or near-singular, with condition number    inf. Standard errors may be u
ARIMA Predictions for State 34:
    Year   Prediction
12  2014  6501.266155
13  2015  2316.732260
14  2016  5643.011597
15  2017  2998.957443
16  2018  5100.711884
17  2019  3430.030622
18  2020  4758.052500
19  2021  3702.410022
20  2022  4541.538548
21  2023  3874.516618
```



```
'''Crime Trend Estimation Using VAR Model
In this section, we implement the VAR (Vector AutoRegression) model for estimating crime trends.
 The VAR model is a powerful statistical tool that captures the interdependencies among multiple time series.
 It allows us to analyze how different types of crimes influence one another over time.

You can run this code directly and then type the name of the state in capital letters with
the correct spelling to generate predictions. This provides a user-friendly way to
explore crime trends for your state of interest, offering deeper insights into the dynamics of crime patterns.'''


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.api import VAR


file_path = 'crimes_cleaned.csv'  # Path to the cleaned dataset
data = pd.read_csv(file_path)

# Input: Ask the user to select a state for prediction
selected_state = input("Enter the state for which you want to make predictions: ").strip()

# Filter the dataset for the chosen state
state_data = data[data['STATE/UT'] == selected_state]

# Check if data exists for the selected state
if state_data.empty:
    print(f"No data found for the state '{selected_state}'. Please check the input.")
else:
    # Group data by year and aggregate numerical values
    data_aggregated = state_data.groupby('YEAR').sum(numeric_only=True)

    # Define the columns to use for crime trend prediction
    features = [
        'MURDER', 'ATTEMPT TO MURDER', 'CULPABLE HOMICIDE NOT AMOUNTING TO MURDER',
        'RAPE', 'CUSTODIAL RAPE', 'OTHER RAPE', 'KIDNAPPING & ABDUCTION',
        'KIDNAPPING AND ABDUCTION OF WOMEN AND GIRLS', 'KIDNAPPING AND ABDUCTION OF OTHERS',
        'DACOITY', 'PREPARATION AND ASSEMBLY FOR DACOITY', 'ROBBERY', 'BURGLARY',
        'THEFT', 'AUTO THEFT', 'OTHER THEFT', 'RIOTS', 'CRIMINAL BREACH OF TRUST',
        'CHEATING', 'COUNTERFIETING', 'ARSON', 'HURT/GREVIOUS HURT', 'DOWRY DEATHS',
        'ASSAULT ON WOMEN WITH INTENT TO OUTRAGE HER MODESTY', 'INSULT TO MODESTY OF WOMEN',
        'CRUELTY BY HUSBAND OR HIS RELATIVES', 'IMPORTATION OF GIRLS FROM FOREIGN COUNTRIES',
        'CAUSING DEATH BY NEGLIGENCE', 'OTHER IPC CRIMES'
    ]
     # Extract training data for the years 2001 to 2013
    train_data = data_aggregated.loc[2001:2013, features]

    # Ensure sufficient training data is available
    if train_data.empty:
        print("Insufficient data for training. Please ensure data from 2001 to 2013 is available.")
    else:
        # Fit a Vector Autoregression (VAR) model
        model = VAR(train_data)
        results = model.fit(maxlags=1)  # Use 1 lag, adjust based on dataset characteristics

        # Generate predictions for the years 2014 to 2023
        forecast = results.forecast(train_data.values, steps=10)
        forecast_df = pd.DataFrame(forecast, index=range(2014, 2024), columns=features)

        # Display the forecasted crime data
        print(f"\nPredicted Crime Data for {selected_state} (2014-2023):")
        print(forecast_df)

        # Combine actual data (up to 2013) with forecasted data (2014 onwards) for visualization
        actual_and_forecast = pd.concat([data_aggregated[features], forecast_df])

        # Visualize the trends using a line plot
        plt.figure(figsize=(15, 10))
```

```
for column in features:
    plt.plot(actual_and_forecast.index, actual_and_forecast[column], label=column)
    # Highlight the prediction period
    plt.axvline(x=2013.5, color='red', linestyle='--', label='Prediction Start' if column == features[0] e

# Add titles, labels, and legend to the plot
plt.title(f'Crime Predictions for {selected_state} (2014-2023)')
plt.xlabel('Year')
plt.ylabel('Crime Counts')
plt.legend(loc='upper left')
plt.grid(True)
# Show the plot
plt.show()
```

```
Enter the state for which you want to make predictions: LAKSHADWEEP
/usr/local/lib/python3.11/dist-packages/statsmodels/tsa/base/tsa_model.py:473: ValueWarning: An unsupported i
  self._init_dates(dates, freq)

Predicted Crime Data for LAKSHADWEEP (2014-2023):
        MURDER  ATTEMPT TO MURDER  CULPABLE HOMICIDE NOT AMOUNTING TO MURDER  \
2014  0.080511           0.309578                                        0.0
2015  0.119399           0.777278                                        0.0
2016  0.728759           0.636584                                        0.0
2017 -0.323096           2.051165                                        0.0
2018  0.818756          -0.466780                                        0.0
2019 -0.513939           1.750266                                        0.0
2020  0.713475          -1.369125                                        0.0
2021 -0.659465           2.062718                                        0.0
2022  0.874415          -1.219250                                        0.0
2023 -0.860888           2.569705                                        0.0

          RAPE  CUSTODIAL RAPE  OTHER RAPE  KIDNAPPING & ABDUCTION  \
2014  0.487917             0.0    0.487917                0.202480
2015  1.279126             0.0    1.279126               -0.240905
2016 -0.389395             0.0   -0.389395                0.647066
2017  1.101382             0.0    1.101382               -0.505299
2018  0.249860             0.0    0.249860                0.986927
2019  2.064489             0.0    2.064489               -0.557460
2020 -0.676503             0.0   -0.676503                0.848189
2021  2.143192             0.0    2.143192               -0.914019
2022 -1.980541             0.0   -1.980541                1.080466
2023  3.627010             0.0    3.627010               -1.286579

      KIDNAPPING AND ABDUCTION OF WOMEN AND GIRLS  \
2014                                     0.202480
2015                                    -0.240905
2016                                     0.647066
2017                                    -0.505299
2018                                     0.986927
2019                                    -0.557460
2020                                     0.848189
2021                                    -0.914019
2022                                     1.080466
2023                                    -1.286579

      KIDNAPPING AND ABDUCTION OF OTHERS   DACOITY  ...  COUNTERFIETING  \
2014                                 0.0  0.962700  ...        0.337681
2015                                 0.0  0.415477  ...        0.167883
2016                                 0.0 -0.047452  ...        0.085120
2017                                 0.0  1.162626  ...       -0.088498
2018                                 0.0 -0.883264  ...       -0.042345
2019                                 0.0  2.276410  ...       -0.001885
2020                                 0.0 -0.618258  ...        0.134008
2021                                 0.0  1.581697  ...       -0.006301
2022                                 0.0 -1.158576  ...        0.205931
2023                                 0.0  1.488723  ...       -0.124129

         ARSON  HURT/GREVIOUS HURT  DOWRY DEATHS  \
2014  3.190087            9.822363           0.0
2015 -1.819449            2.374982           0.0
2016  4.961729            9.069335           0.0
2017 -1.167720            1.440960           0.0
2018  5.551813            7.831305           0.0
2019 -1.461600            6.809587           0.0
2020  6.104160            7.593753           0.0
2021 -3.072610            1.867579           0.0
2022  8.921992            7.856775           0.0
2023 -6.896469           -1.822248           0.0

      ASSAULT ON WOMEN WITH INTENT TO OUTRAGE HER MODESTY  \
2014                                           0.192877
2015                                           0.910992
2016                                          -0.884186
2017                                           1.604012
2018                                          -0.414433
2019                                           1.403680
2020                                          -0.759487
2021                                           1.400667
2022                                          -1.032424
```