NAME: AVNISH CHAUHAN

INTERNSHIP ID: UMIP24311

PROJECT 2: WORLD POPULATION ANALYSIS

DATA ANALYTICS ONE MONTH INTERNSHIP

BATCH (05 Oct 2024 to 05 Nov 2024)

OBJECTIVE

The goal of this project is to analyze global population trends using historical data and predict future population growth. This involves using machine learning techniques to explore demographic data, identify key factors influencing population changes, and build predictive models.

ABOUT DATASET

Context

The current US Census Bureau world population estimate in June 2019 shows that the current global population is7,577,130,400 people on earth, which far exceeds the world population of 7.2 billion in 2015. Our own estimate based on UN data shows the world's population surpassing 7.7 billion. China is the most populous country in the world with a population exceeding 1.4 billion. It is one of just two countries with a population of more than 1 billion, with India being the second. As of 2018, India has a population of over 1.355 billion people, and its population growth is expected to continue through at least 2050. By the year 2030, the country of India is expected to become the most populous country in the world. This is because India’s population will grow, while China is projected to see a loss in population. The following 11 countries that are the most populous in the world each have populations exceeding 100 million. These include the United States, Indonesia, Brazil, Pakistan, Nigeria, Bangladesh, Russia, Mexico, Japan, Ethiopia, and the Philippines. Of these nations, all are expected to continue to grow except Russia and Japan, which will see their populations drop by 2030 before falling again significantly by 2050. Many other nations have populations of at least one million, while there are also countries that have just thousands. The smallest population in the world can be found in Vatican City, where only 801 people reside. In 2018, the world’s population growth rate was 1.12%. Every five years since the 1970s, the population growth rate has continued to fall. The world’s population is expected to continue to grow larger but at a much slower pace. By 2030, the population will exceed 8 billion. In 2040, this number will grow to more than 9 billion. In 2055, the number will rise to over 10 billion, and another billion people won’t be added until near the end of the century. The current annual population growth estimates from the United Nations are in the millions -

estimating that over 80 million new lives are added each year. This population growth will be significantly impacted by nine specific countries which are situated to contribute to the population growing more quickly than other nations. These nations include the Democratic Republic of the Congo, Ethiopia, India, Indonesia, Nigeria, Pakistan, Uganda, the United Republic of Tanzania, and the United States of America. Particularly of interest, India is on track to overtake China's position as the most populous country by 2030. Additionally, multiple nations within Africa are expected to double their populations before fertility rates begin to slow entirely.

Content

In this Dataset, we have Historical Population data for every Country/Territory in the world by different parameters like Area Size of the Country/Territory, Name of the Continent, Name of the Capital, Density, Population Growth Rate, Ranking based on Population, World Population Percentage, etc.

IMPORT LIBRARIES

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.subplots as sp
import plotly.graph_objects as go
```

In [2]:

```
from plotly.subplots import make_subplots
import warnings
# Suppress FutureWarning messages
warnings.simplefilter(action='ignore', category=FutureWarning)
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
init_notebook_mode(connected=True)
# Graph
```

IMPORT DATA

In [3]:
```
df = pd.read_csv('C:/Users/Admin/Downloads/world_population.csv')
df.head()
```

Out[3]:

	Rank	CCA3	Country/Territory	Capital	Continent	2022 Population	2020 Population	2015 Population	2010 Population	2000 Population	1990 Population	198 Populatio
0	36	AFG	Afghanistan	Kabul	Asia	41128771	38972230	33753499	28189672	19542982	10694796	1248663
1	138	ALB	Albania	Tirana	Europe	2842321	2866849	2882481	2913399	3182021	3295066	294165
2	34	DZA	Algeria	Algiers	Africa	44903225	43451666	39543154	35856344	30774621	25518074	1873937
3	213	ASM	American Samoa	Pago Pago	Oceania	44273	46189	51368	54849	58230	47818	3288
4	203	AND	Andorra	Andorra la Vella	Europe	79824	77700	71746	71519	66097	53569	3561

```
In [4]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 234 entries, 0 to 233
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   Rank                         234 non-null    int64  
 1   CCA3                         234 non-null    object 
 2   Country/Territory            234 non-null    object 
 3   Capital                      234 non-null    object 
 4   Continent                    234 non-null    object 
 5   2022 Population              234 non-null    int64  
 6   2020 Population              234 non-null    int64  
 7   2015 Population              234 non-null    int64  
 8   2010 Population              234 non-null    int64  
 9   2000 Population              234 non-null    int64  
 10  1990 Population              234 non-null    int64  
 11  1980 Population              234 non-null    int64  
 12  1970 Population              234 non-null    int64  
 13  Area (km²)                   234 non-null    int64  
 14  Density (per km²)            234 non-null    float64
 15  Growth Rate                  234 non-null    float64
 16  World Population Percentage  234 non-null    float64
dtypes: float64(3), int64(10), object(4)
memory usage: 31.2+ KB
```

```
In [5]: df.isna().sum()

Out[5]: Rank                           0
        CCA3                           0
        Country/Territory              0
        Capital                        0
        Continent                      0
        2022 Population                0
        2020 Population                0
        2015 Population                0
        2010 Population                0
        2000 Population                0
        1990 Population                0
        1980 Population                0
        1970 Population                0
        Area (km²)                     0
        Density (per km²)              0
        Growth Rate                    0
        World Population Percentage    0
        dtype: int64
```

```
In [6]: print(f"Amount of duplicates: {df.duplicated().sum()}")

        Amount of duplicates: 0
```

```
In [7]: df.columns

Out[7]: Index(['Rank', 'CCA3', 'Country/Territory', 'Capital', 'Continent',
               '2022 Population', '2020 Population', '2015 Population',
               '2010 Population', '2000 Population', '1990 Population',
               '1980 Population', '1970 Population', 'Area (km²)', 'Density (per km²)',
               'Growth Rate', 'World Population Percentage'],
              dtype='object')
```

```
In [8]: df.drop(['CCA3', 'Capital'], axis=1, inplace=True)
```

In [9]:
```
df.head()
```

Out[9]:

	Rank	Country/Territory	Continent	2022 Population	2020 Population	2015 Population	2010 Population	2000 Population	1990 Population	1980 Population	1970 Population
0	36	Afghanistan	Asia	41128771	38972230	33753499	28189672	19542982	10694796	12486631	10752971
1	138	Albania	Europe	2842321	2866849	2882481	2913399	3182021	3295066	2941651	2324731
2	34	Algeria	Africa	44903225	43451666	39543154	35856344	30774621	25518074	18739378	13795915
3	213	American Samoa	Oceania	44273	46189	51368	54849	58230	47818	32886	27075
4	203	Andorra	Europe	79824	77700	71746	71519	66097	53569	35611	19860

In [10]:
```
df.tail()
```

Out[10]:

	Rank	Country/Territory	Continent	2022 Population	2020 Population	2015 Population	2010 Population	2000 Population	1990 Population	1980 Population	1970 Population
229	226	Wallis and Futuna	Oceania	11572	11655	12182	13142	14723	13454	11315	9377
230	172	Western Sahara	Africa	575986	556048	491824	413296	270375	178529	116775	76371
231	46	Yemen	Asia	33696614	32284046	28516545	24743946	18628700	13375121	9204938	6843607
232	63	Zambia	Africa	20017675	18927715	16248230	13792086	9891136	7686401	5720438	4281671
233	74	Zimbabwe	Africa	16320537	15669666	14154937	12839771	11834676	10113893	7049926	5202918

Visualizations

In [11]:
```
custom_palette = ['#0b3d91', '#e0f7fa', '#228b22', '#1e90ff', '#8B4513', '#D2691E',
'#DAA520', '#556B2F']
```

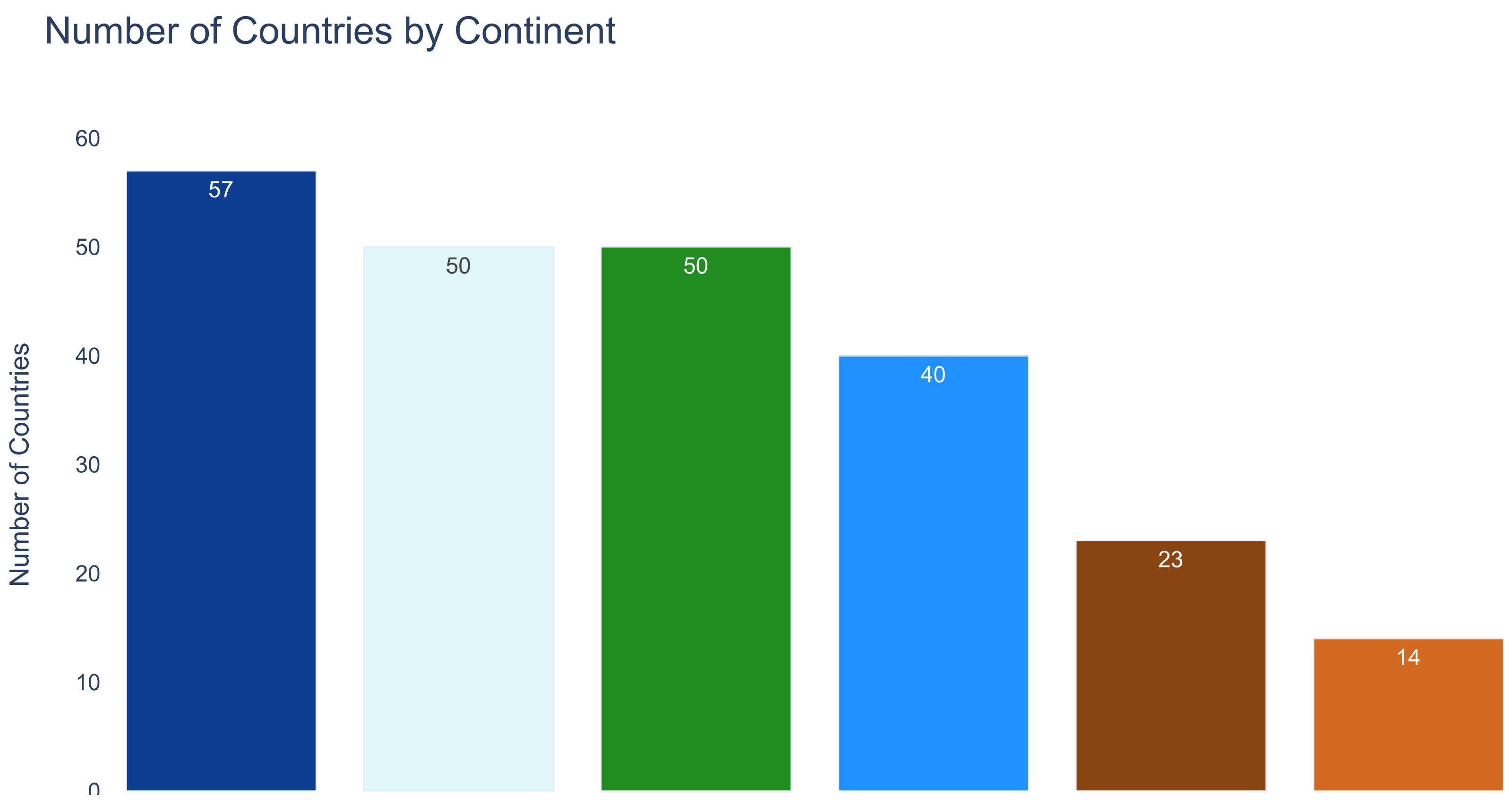
In [12]:
```
countries_by_continent = df['Continent'].value_counts().reset_index()
```

In [13]:

```
# Create the bar chart
fig = px.bar(
countries_by_continent,
x='Continent',
y='count',
color='Continent',
text='count',
title='Number of Countries by Continent',
color_discrete_sequence=custom_palette
)
```

In [14]:

```
# Customize the layout
fig.update_layout(
xaxis_title='Continents',
yaxis_title='Number of Countries',
plot_bgcolor='rgba(0,0,0,0)', # Set the background color to transparent
font_family='Arial', # Set font family
title_font_size=20) # Set title font size
fig.show()
```

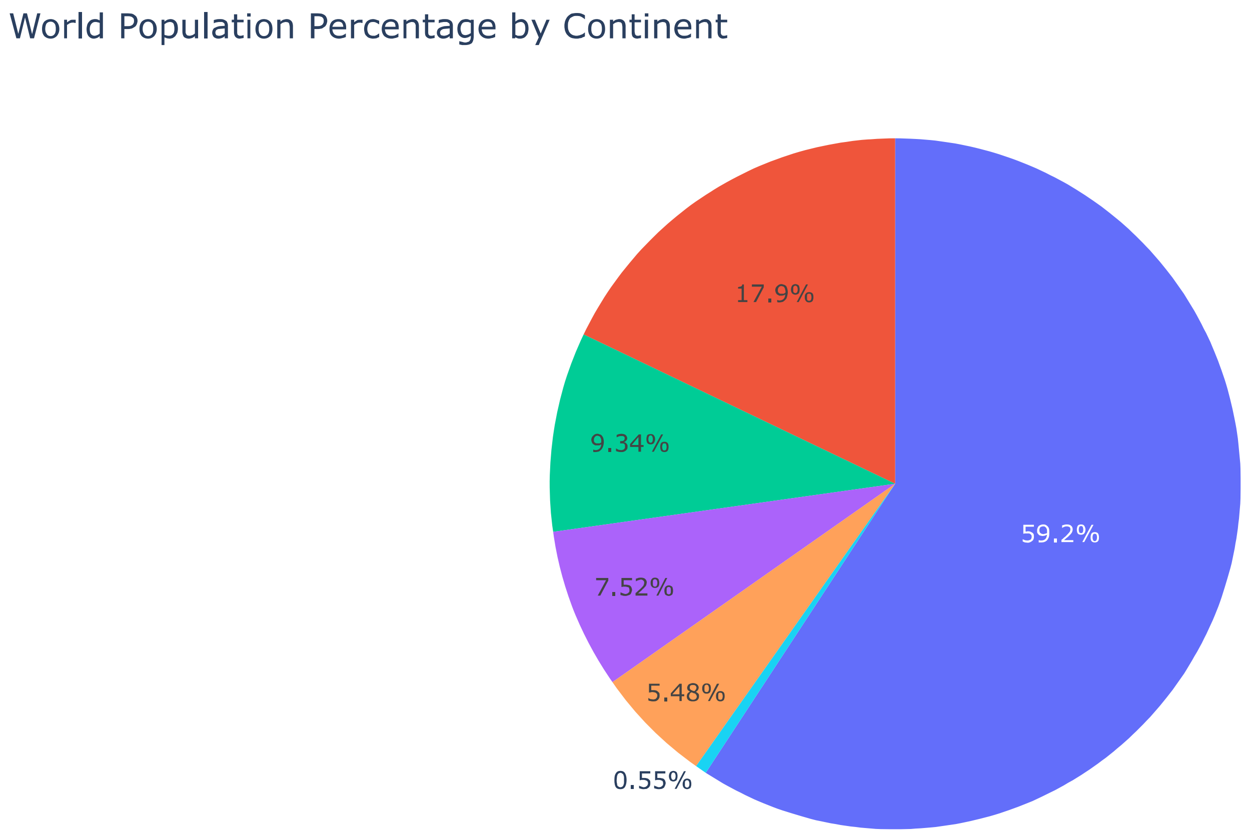


In [15]:

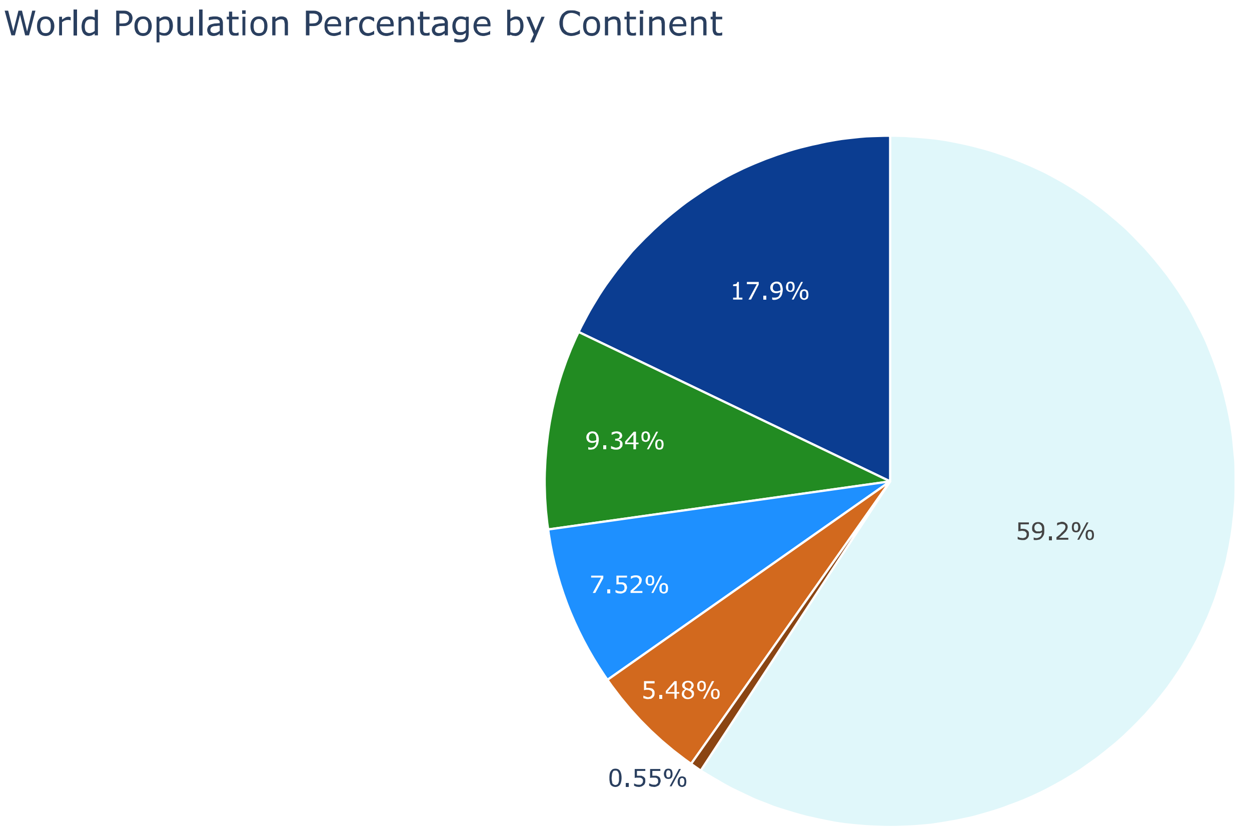
```
continent_population_percentage = df.groupby('Continent')['World Population Percentage'].sum().reset_index()
```

In [16]:

```
# Create the pie chart
fig = go.Figure(data=[go.Pie(labels=continent_population_percentage['Continent'],values=continent_population_p

# Update layout
fig.update_layout(
title='World Population Percentage by Continent',
template='plotly',
paper_bgcolor='rgba(255,255,255,0)', # Set the paper background color to transparent
plot_bgcolor='rgba(255,255,255,0)' # Set the plot background color to transparent
)
```


World Population Percentage by Continent
17.9%
9.34%
7.52%
5.48%
0.55%
59.2%

In [17]:
```
# Update pie colors
fig.update_traces(marker=dict(colors=custom_palette, line=dict(color='#FFFFFF',width=1)))
# Show the plot
fig.show()
```

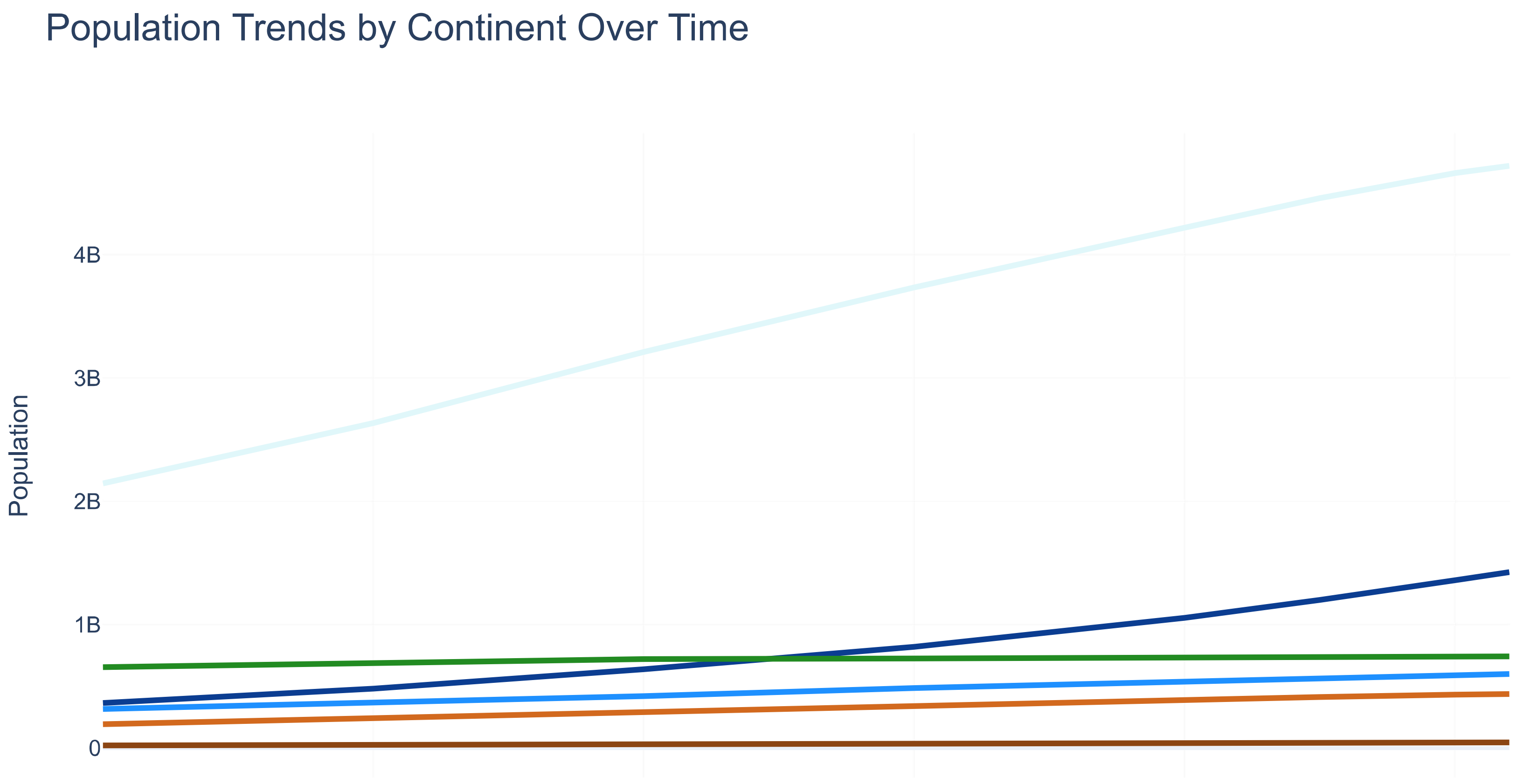

World Population Percentage by Continent

In [18]:

```
# Melt the DataFrame to have a long format
df_melted = df.melt(
    id_vars=['Continent'],
    value_vars=[
        '2022 Population', '2020 Population', '2015 Population',
        '2010 Population', '2000 Population', '1990 Population',
        '1980 Population', '1970 Population'
    ],
    var_name='Year',
    value_name='Population'
)

# Convert 'Year' to a more suitable format by extracting the year as an integer
df_melted['Year'] = df_melted['Year'].str.split().str[0].astype(int)

# Aggregate population by continent and year
population_by_continent = df_melted.groupby(['Continent', 'Year']).sum().reset_index()
```

In [19]:

```python
fig = px.line(population_by_continent, x='Year', y='Population', color='Continent',

title='Population Trends by Continent Over Time',
labels={'Population': 'Population', 'Year': 'Year'},
color_discrete_sequence=custom_palette)

fig.update_layout(template='plotly_white',
xaxis_title='Year',
yaxis_title='Population',
font_family='Arial',
title_font_size=20,
)

fig.update_traces(line=dict(width=3))

fig.show()
```

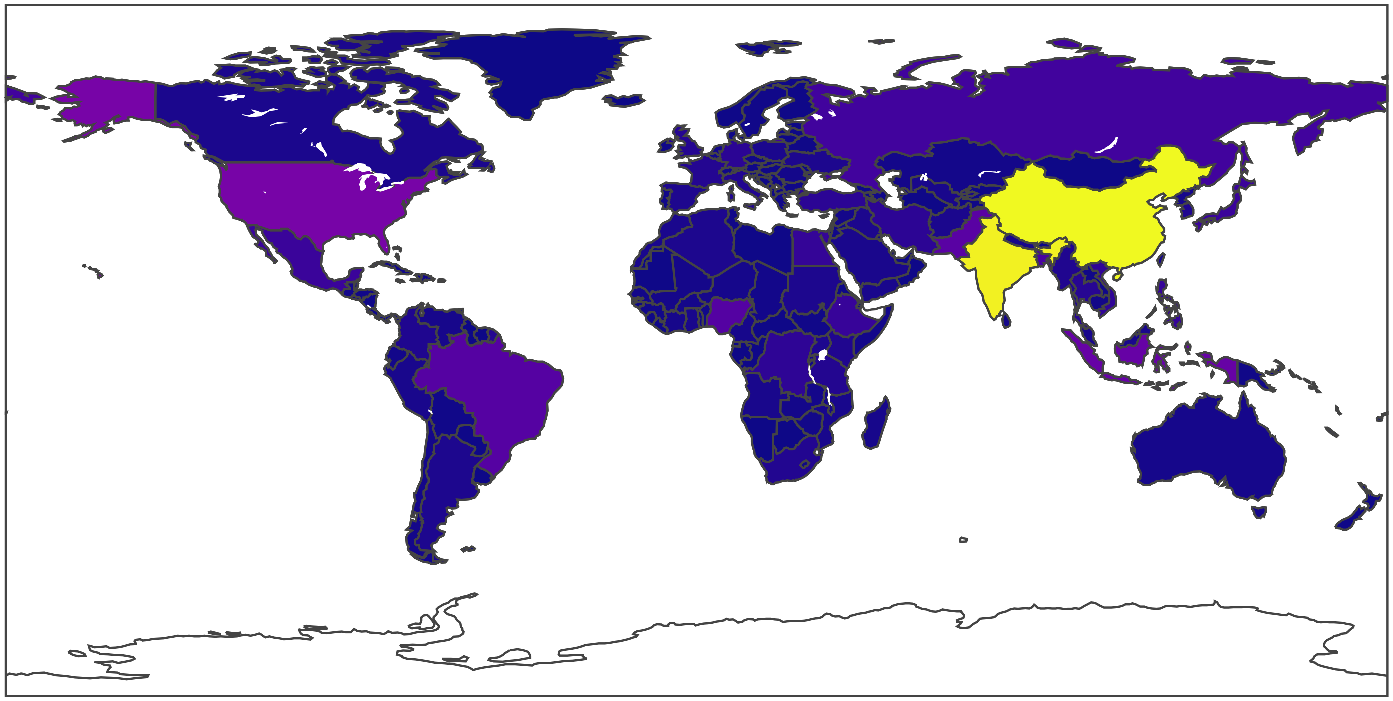


World Population Comparison: 1970 to 2020

In [20]:

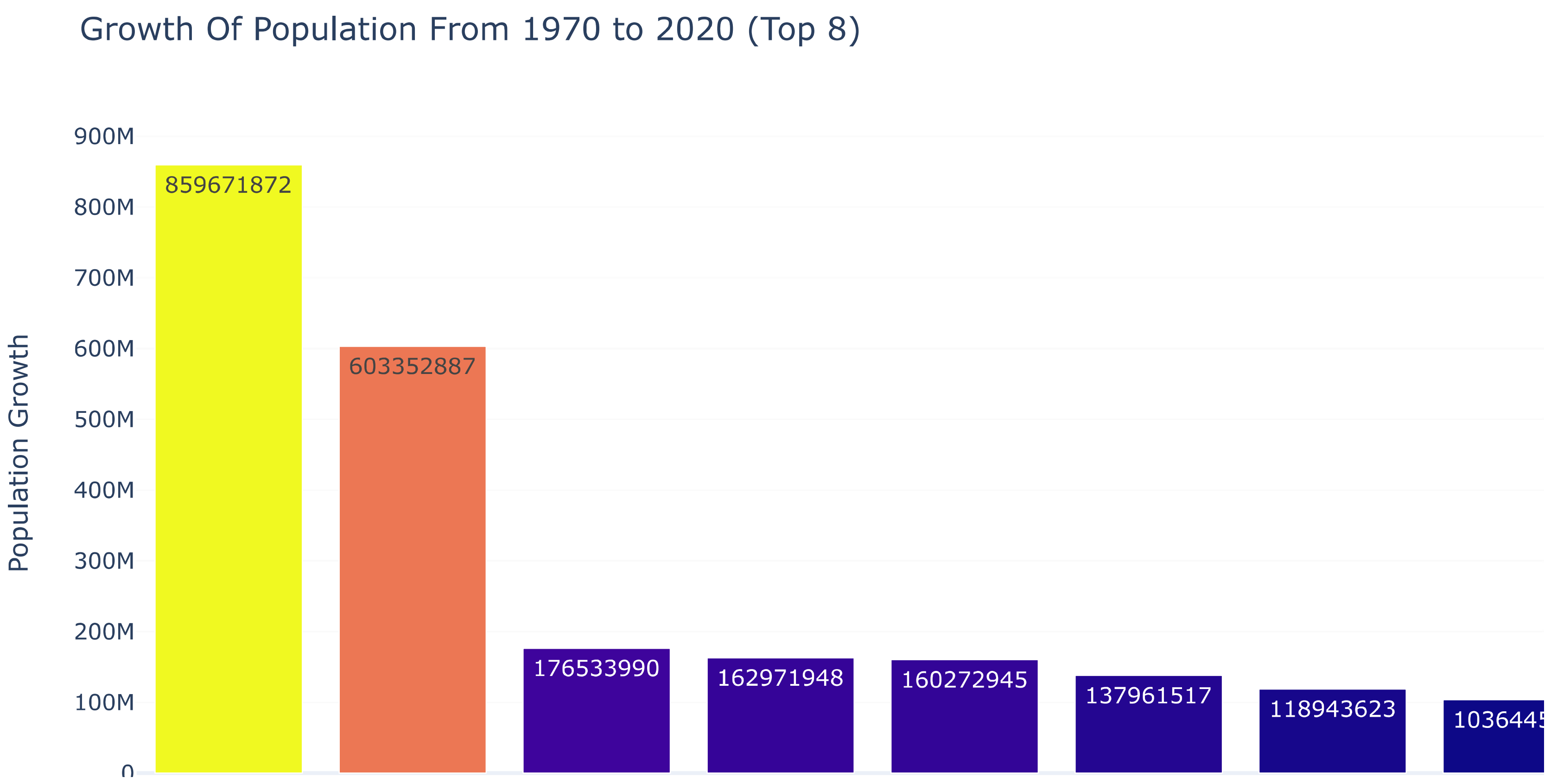
```
features=['1970 Population' ,'2020 Population']
for feature in features:
    fig = px.choropleth(df,

locations='Country/Territory',
locationmode='country names',
color=feature,
hover_name='Country/Territory',
template='plotly_white',
title = feature)

fig.show()
```

2020 Population

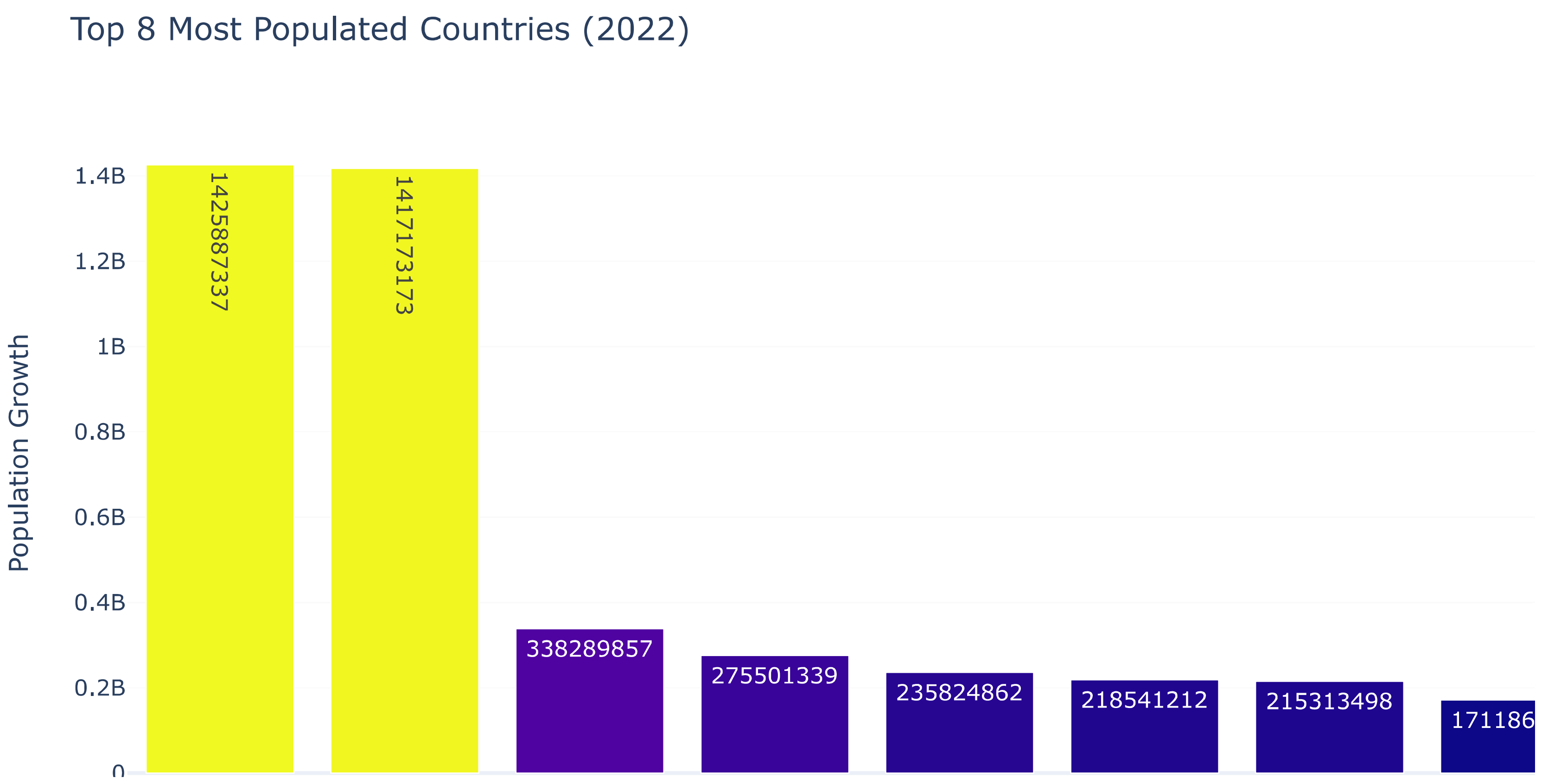


In [21]:

```
growth = (df.groupby(by='Country/Territory')['2022 Population'].sum()-df.groupby(by='Country/Territory')['1970

fig=px.bar(x=growth.index,
y=growth.values,
text=growth.values,
color=growth.values,
title='Growth Of Population From 1970 to 2020 (Top 8)',
template='plotly_white')
fig.update_layout(xaxis_title='Country',

yaxis_title='Population Growth')

fig.show()
```


Growth Of Population From 1970 to 2020 (Top 8)
Population Growth
900M
800M
700M
600M
500M
400M
300M
200M
100M
0
859671872
603352887
176533990
162971948
160272945
137961517
118943623
1036445

In [22]:
```
top_8_populated_countries_1970 = df.groupby('Country/Territory')['1970 Population'].sum().sort_values(ascendir
top_8_populated_countries_2022 = df.groupby('Country/Territory')['2022 Population'].sum().sort_values(ascendir

features = {'top_8_populated_countries_1970': top_8_populated_countries_1970, 'top_8_populated_countries_2022'

for feature_name, feature_data in features.items():
    year = feature_name.split('_')[-1] # Extract the year from the feature name
fig = px.bar(x=feature_data.index,
y=feature_data.values,
text=feature_data.values,
color=feature_data.values,
title=f'Top 8 Most Populated Countries ({year})',
template='plotly_white')
fig.update_layout(xaxis_title='Country',

yaxis_title='Population Growth')

fig.show()
```


Top 8 Most Populated Countries (2022)

World Population Growth Rates: The Fastest Growing Countries

```
In [23]: sorted_df_growth = df.sort_values(by='Growth Rate', ascending=False)
         top_fastest = sorted_df_growth.head(6)
         top_slowest = sorted_df_growth.tail(6)
```

In [24]:

```
def plot_population_trends(countries):         # Calculate the number of rows needed
    n_cols = 2
    n_rows = (len(countries) + n_cols - 1) // n_cols

# Create subplots
    fig = sp.make_subplots(rows=n_rows, cols=n_cols, subplot_titles=countries,
    horizontal_spacing=0.1, vertical_spacing=0.1)

    for i, country in enumerate(countries, start=1):      # Filter data for the selected country
        country_df = df[df['Country/Territory'] == country]

# Melt the DataFrame to have a long format
        country_melted = country_df.melt(id_vars=['Country/Territory'],

        value_vars=['2022 Population', '2020 Population', '2015 Population', '2010 Population', '2000 Populati

        var_name='Year',
        value_name='Population')

# Convert 'Year' to a more suitable format
        country_melted['Year'] = country_melted['Year'].str.split().str[0].astype(int)
# Create a line plot for each country
        line_fig = px.line(country_melted, x='Year', y='Population', color='Country/Territory', labels={'Popul
# Update the line plot to fit the subplot
        row = (i - 1) // n_cols + 1
        col = (i - 1) % n_cols + 1
        for trace in line_fig.data:
            fig.add_trace(trace, row=row, col=col)
# Update the layout of the subplots
    fig.update_layout(
    title='Population Trends of Selected Countries Over Time',
    template='plotly_white',
    font_family='Arial',
    title_font_size=20,
    showlegend=False,
    height=600*n_rows, # Adjust height for bigger plots
    )
    fig.update_traces(line=dict(width=3))
    fig.update_xaxes(title_text='Year')
    fig.update_yaxes(title_text='Population')
    fig.show()
```

In [ ]:

In [25]:

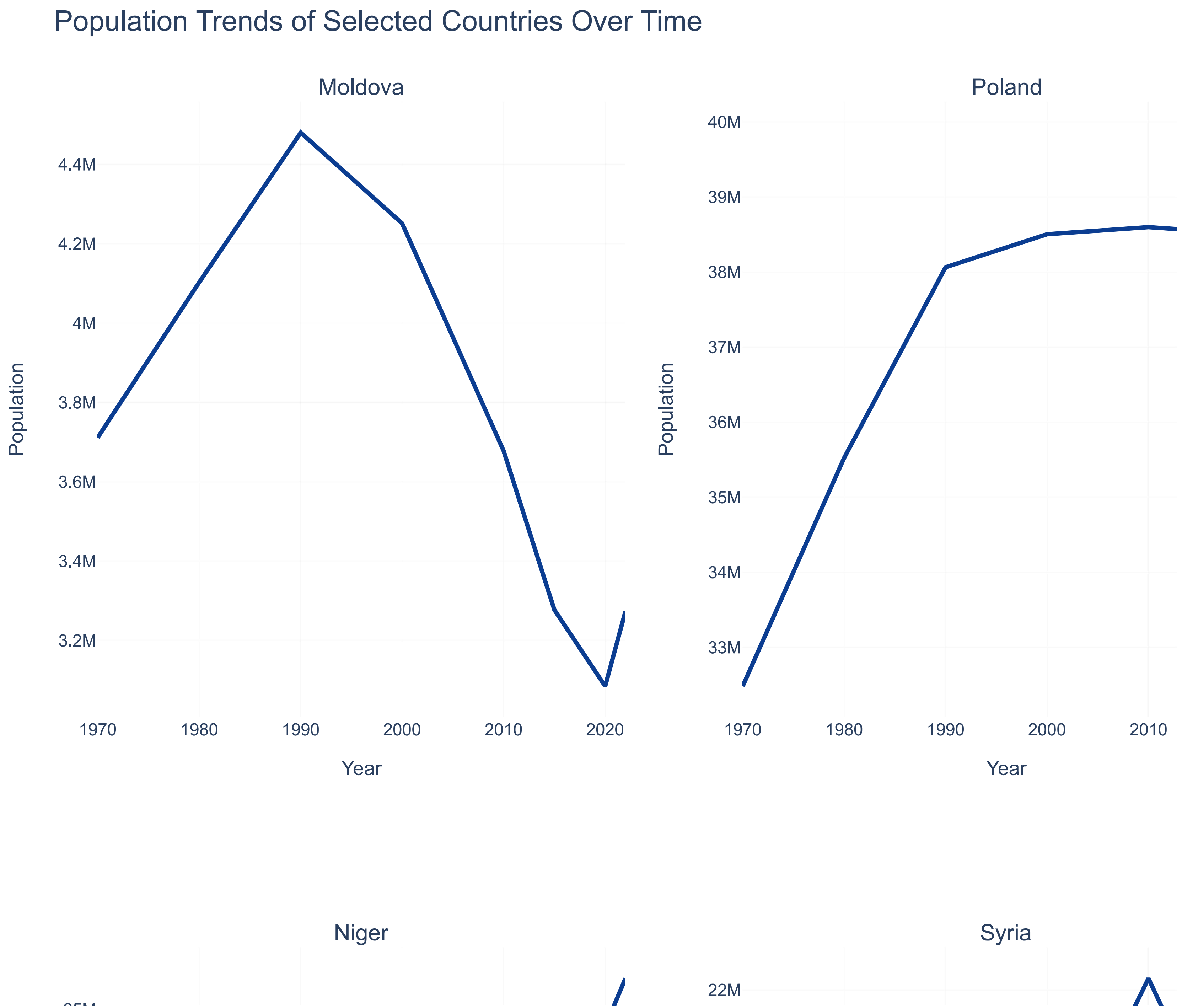
```
fastest = top_fastest[['Country/Territory', 'Growth Rate']].sort_values(by='Growth Rate', ascending=False).res
fastest
```

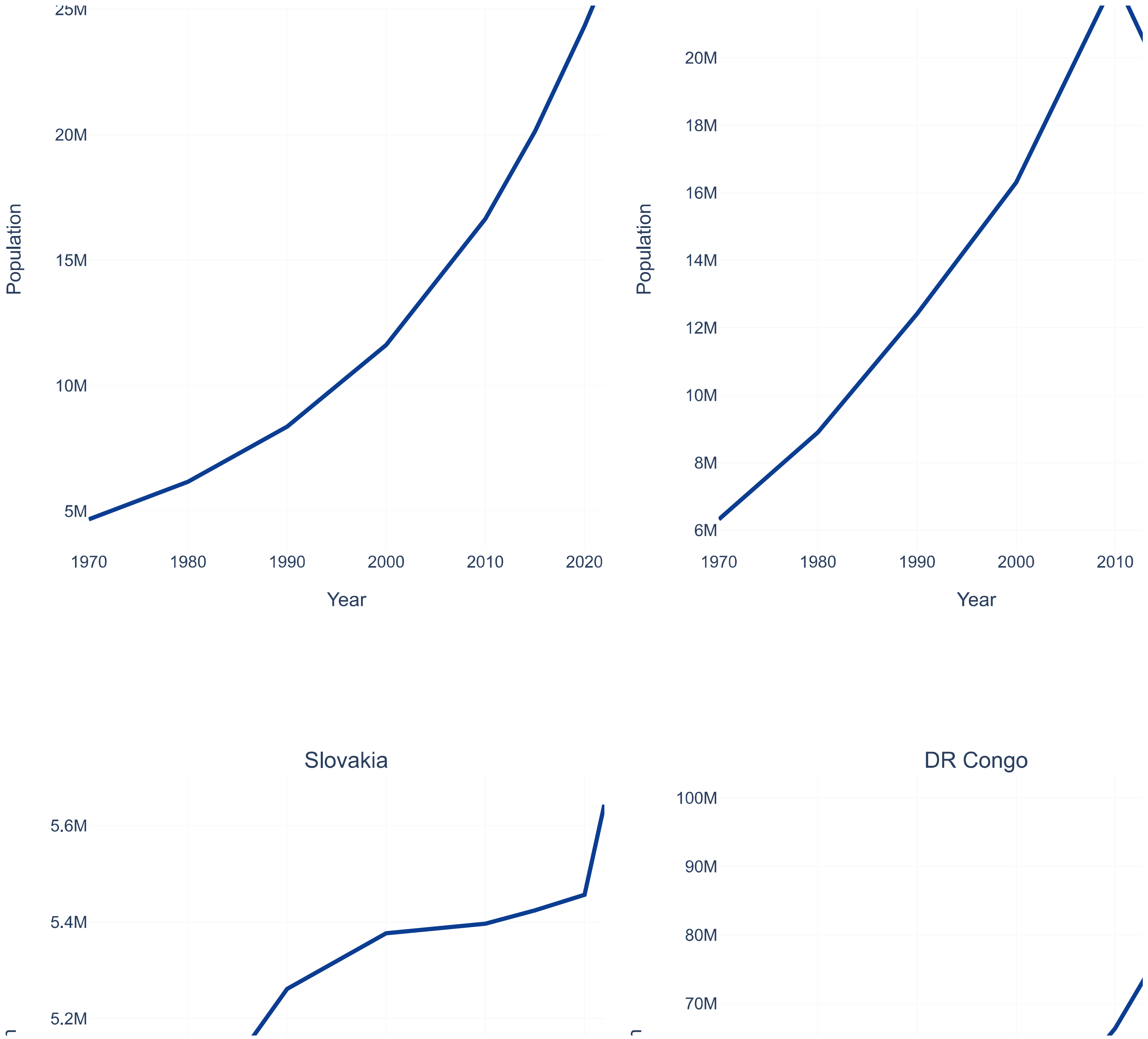
Out[25]:

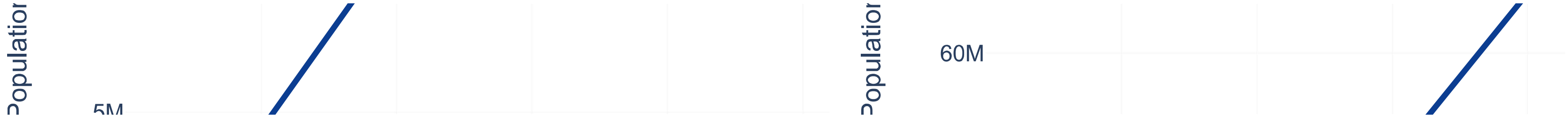
	Country/Territory	Growth Rate
0	Moldova	1.0691
1	Poland	1.0404
2	Niger	1.0378
3	Syria	1.0376
4	Slovakia	1.0359
5	DR Congo	1.0325

In [26]:

```
plot_population_trends(['Moldova', 'Poland', 'Niger', 'Syria', 'Slovakia', 'DR Congo'])
```


Population Trends of Selected Countries Over Time
Moldova
Poland
Population
4.4M
4.2M
4M
3.8M
3.6M
3.4M
3.2M
1970
1980
1990
2000
2010
2020
Year
40M
39M
38M
37M
36M
35M
34M
33M
1970
1980
1990
2000
2010
Year
Niger
Syria
22M


25M
20M
15M
10M
5M
Population
1970
1980
1990
2000
2010
2020
Year
20M
18M
16M
14M
12M
10M
8M
6M
Population
1970
1980
1990
2000
2010
Year
Slovakia
5.6M
5.4M
5.2M
DR Congo
100M
90M
80M
70M



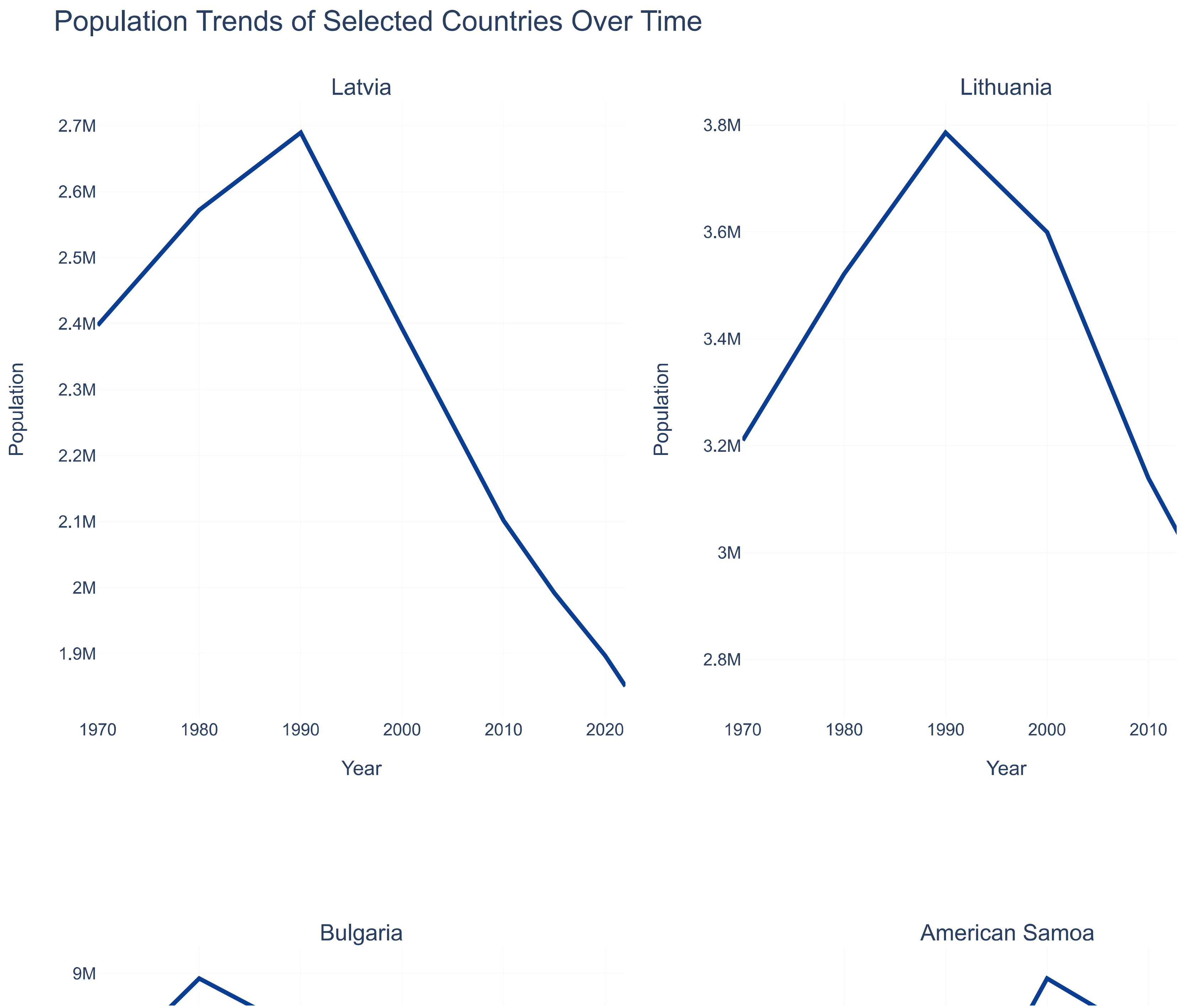
World Population Growth Rates: The Slowest Growing Countries

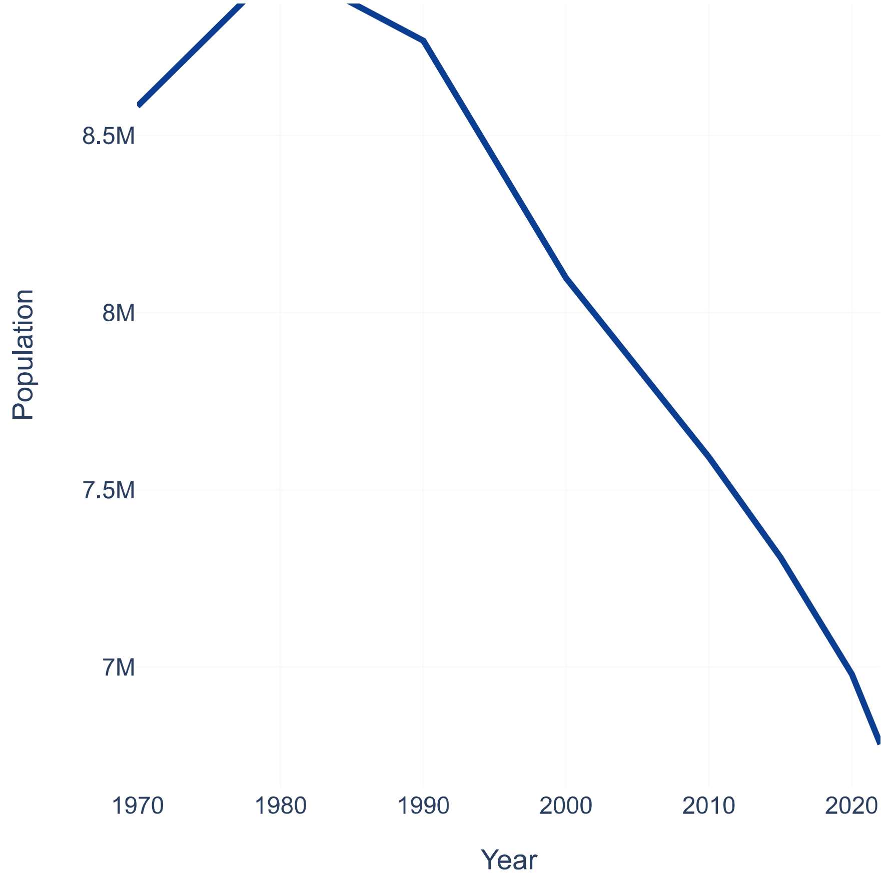
In [27]:
```
slowest = top_slowest[['Country/Territory', 'Growth Rate']].sort_values(by='Growth Rate', ascending=False).res
slowest
```

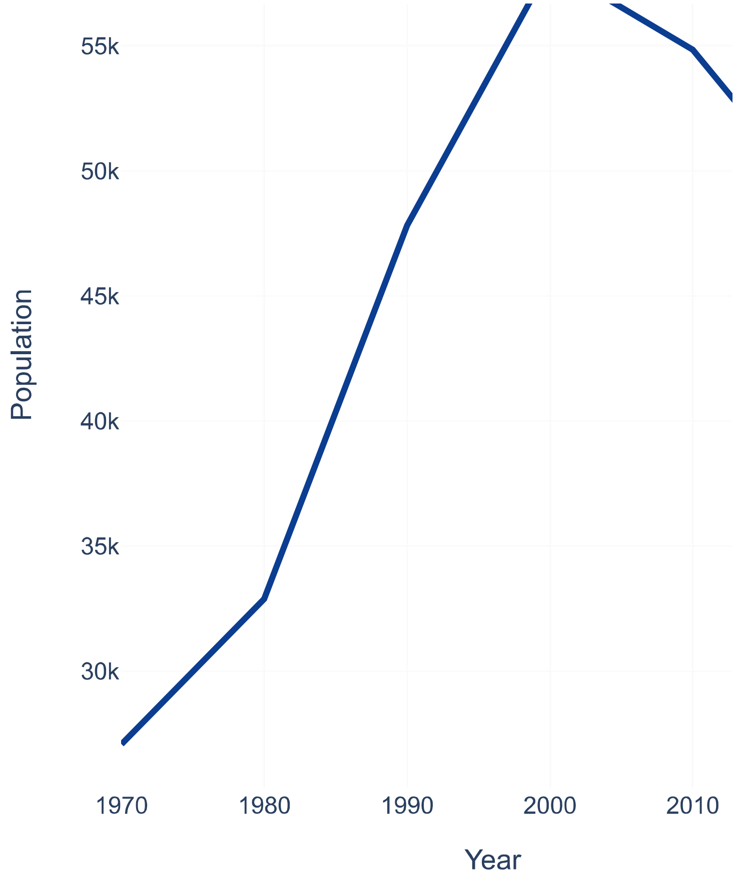
Out[27]:

	Country/Territory	Growth Rate
0	Latvia	0.9876
1	Lithuania	0.9869
2	Bulgaria	0.9849
3	American Samoa	0.9831
4	Lebanon	0.9816
5	Ukraine	0.9120

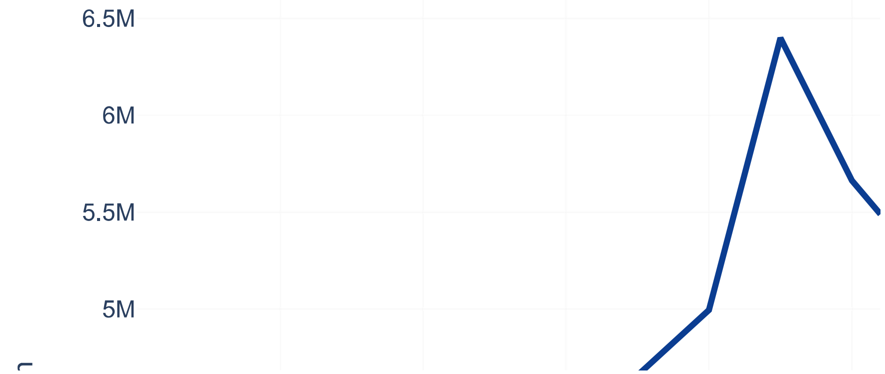
```
In [28]: plot_population_trends(['Latvia', 'Lithuania', 'Bulgaria', 'American Samoa', 'Lebanon', 'Ukraine'])
```


Population Trends of Selected Countries Over Time
Latvia
2.7M
2.6M
2.5M
2.4M
2.3M
2.2M
2.1M
2M
1.9M
Population
1970
1980
1990
2000
2010
2020
Year
Lithuania
3.8M
3.6M
3.4M
3.2M
3M
2.8M
Population
1970
1980
1990
2000
2010
Year
Bulgaria
9M
American Samoa

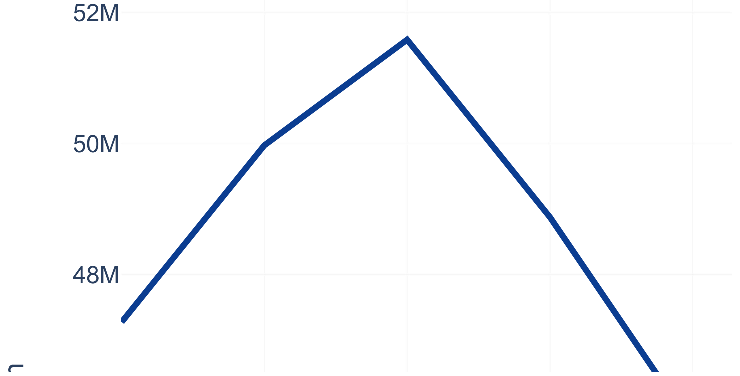


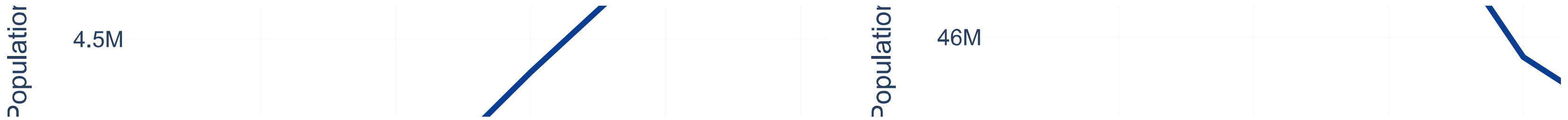


Lebanon



Ukraine





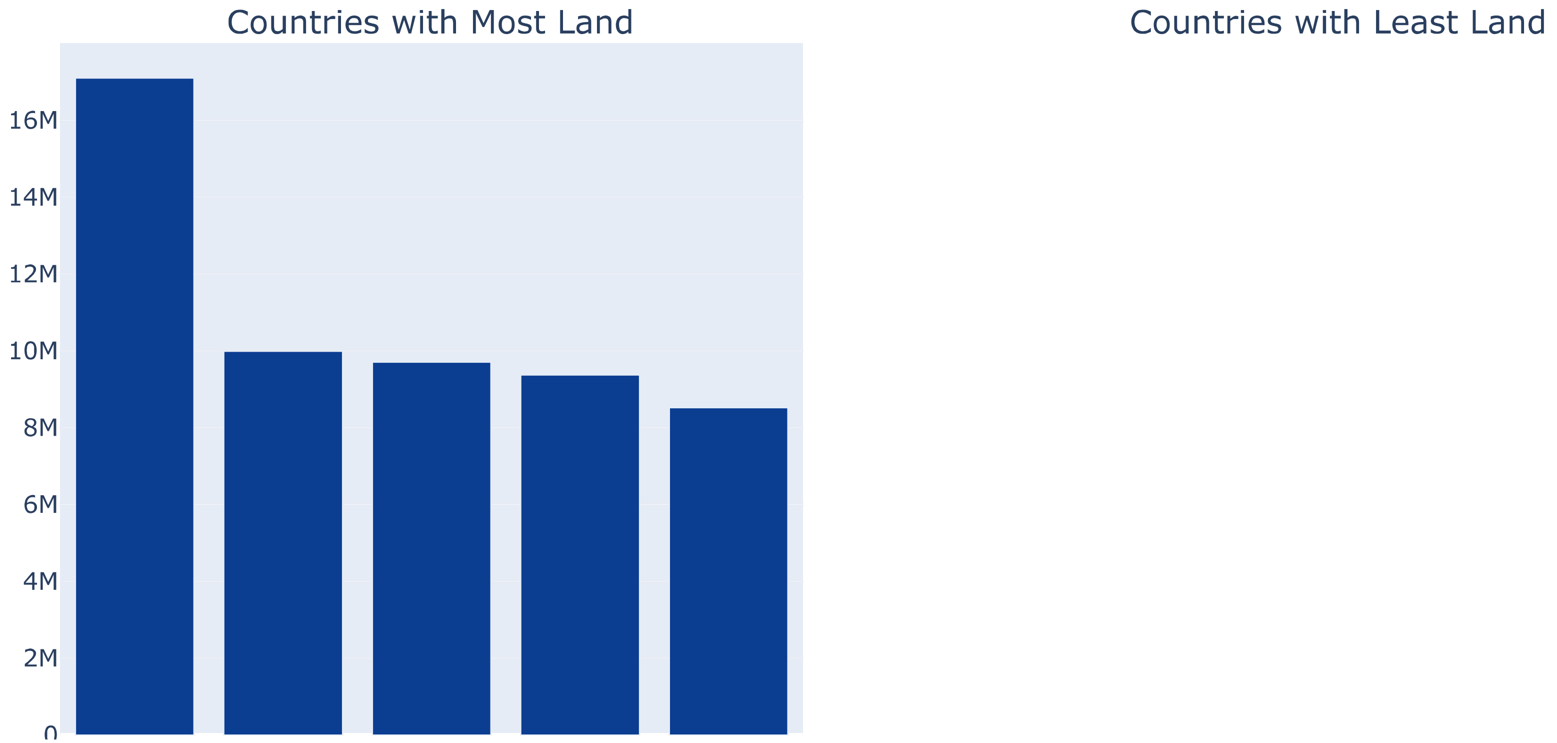
Land Area by Country

In [29]:

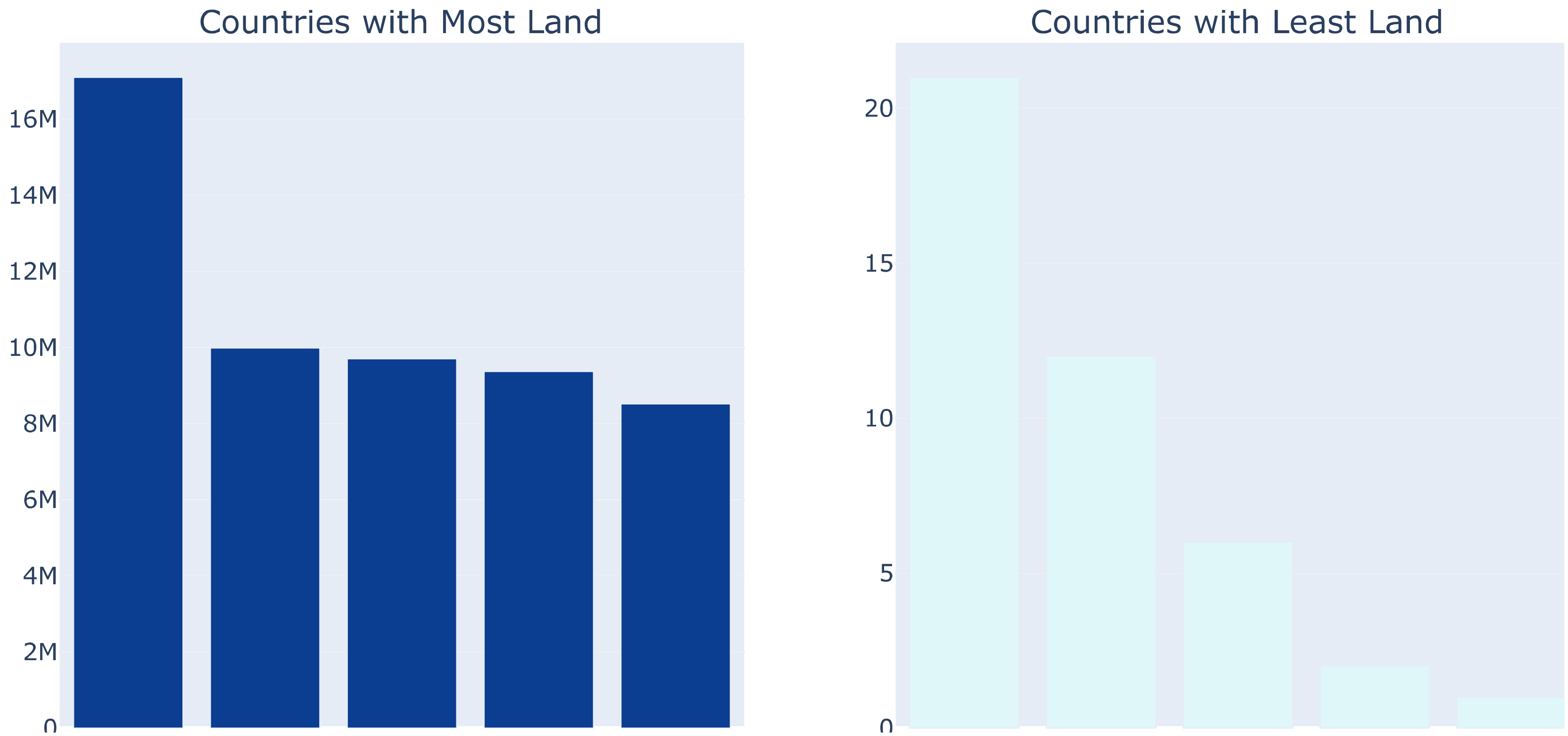
```
land_by_country = df.groupby('Country/Territory')['Area (km²)'].sum().sort_values(ascending=False)
most_land = land_by_country.head(5)
least_land = land_by_country.tail(5)
```

In [30]:

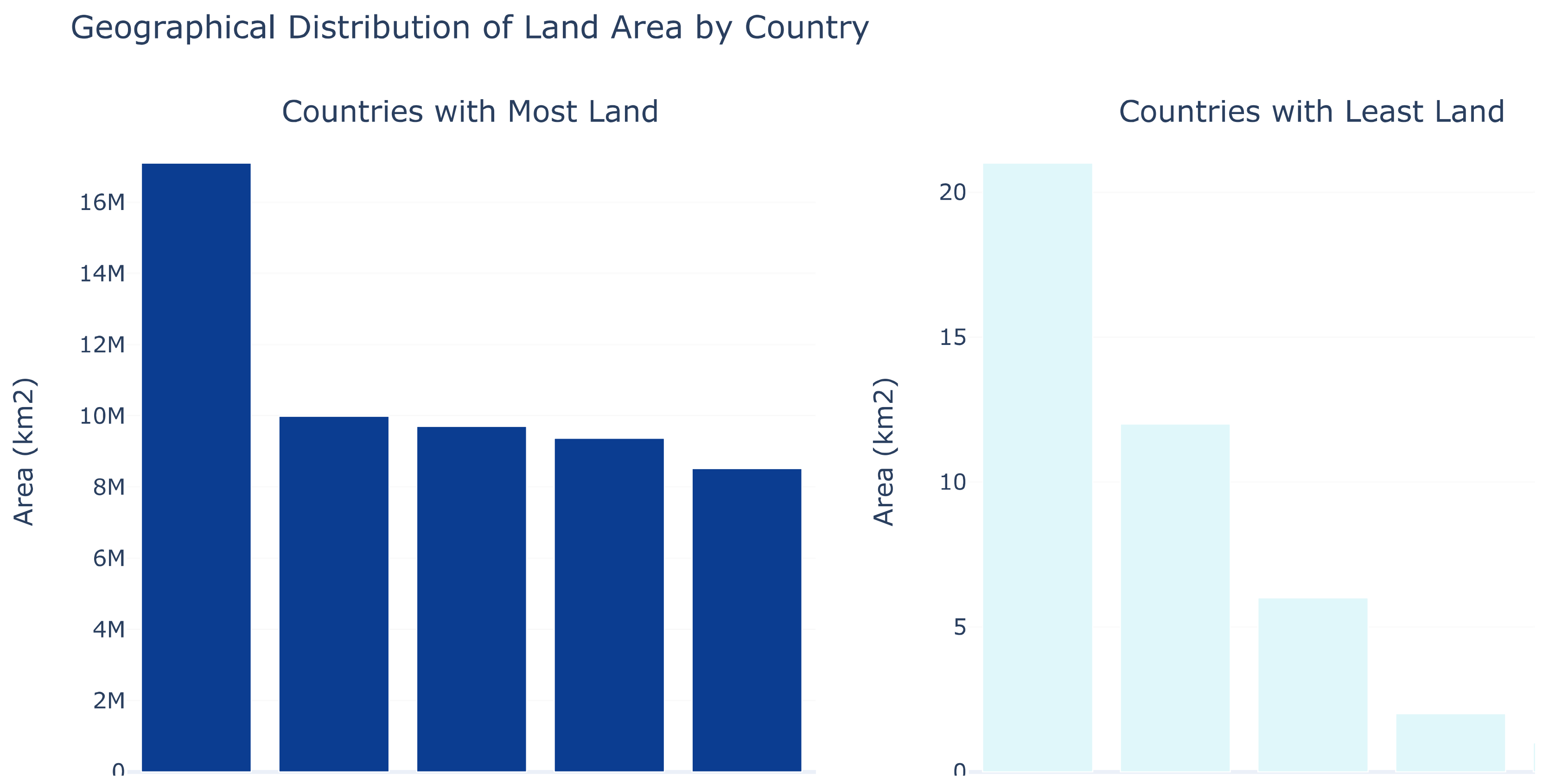
```
# Create subplots
fig = sp.make_subplots(rows=1, cols=2, subplot_titles=("Countries with Most Land", "Countries with Least Land"
In [31]: # Plot countries with the most land
         fig.add_trace(go.Bar(x=most_land.index, y=most_land.values, name='Most Land',  marker_color=custom_palette[0])
```



In [32]:
```
# Plot countries with the least land
fig.add_trace(go.Bar(x=least_land.index, y=least_land.values, name='Least Land', marker_color=custom_palette[1
```



In [33]:
```
fig.update_layout(
title_text="Geographical Distribution of Land Area by Country",
showlegend=False,
template='plotly_white'
)

fig.update_yaxes(title_text="Area (km2)", row=1, col=1)
fig.update_yaxes(title_text="Area (km2)", row=1, col=2)

fig.show()
```



Geographical Distribution of Land Area by Country

Countries with Most Land

Countries with Least Land

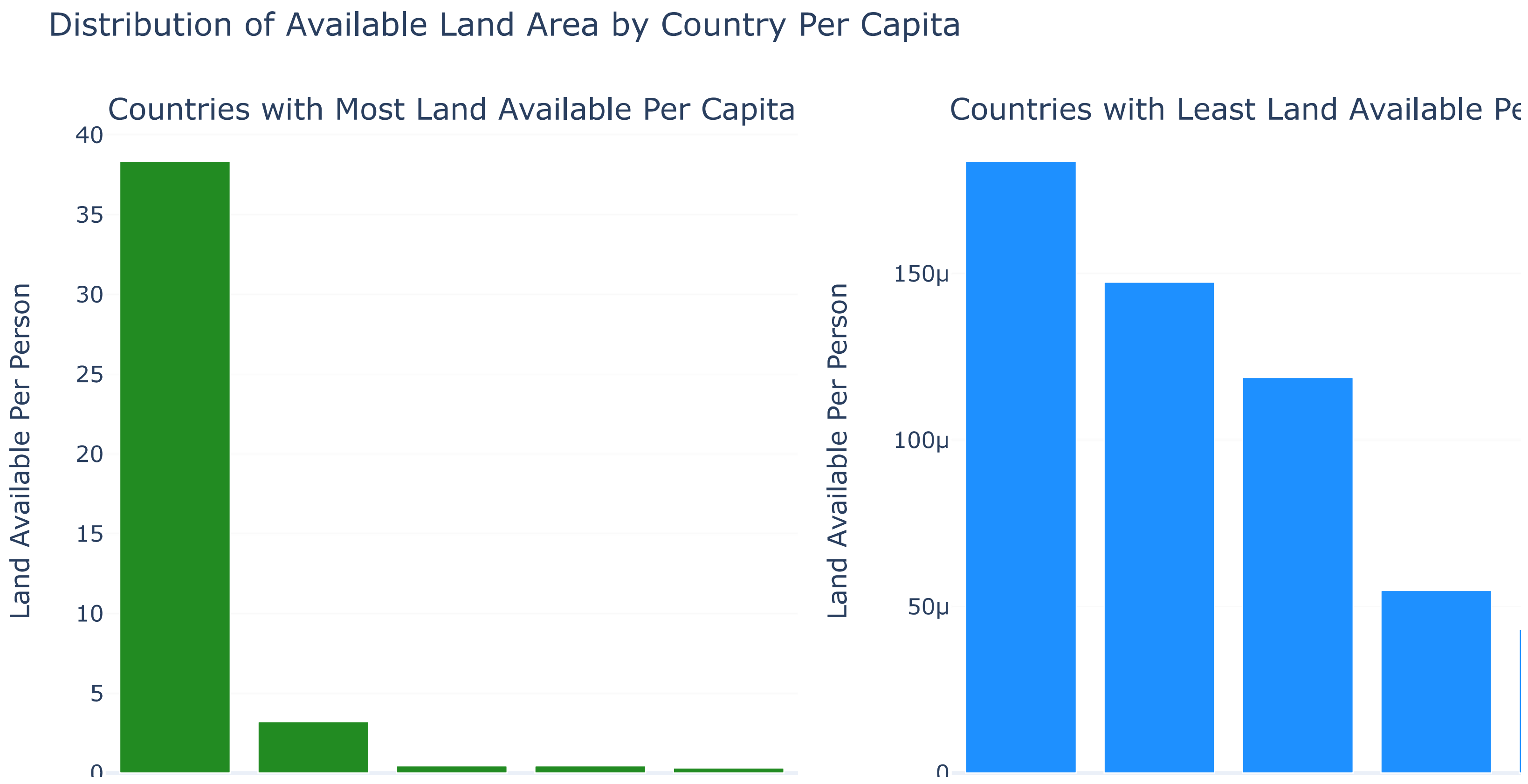
Land Area Per Person by Country

In [34]:

```
df['Area per Person']=df['Area (km²)'] / df['2022 Population']
country_area_per_person = df.groupby('Country/Territory')['Area per Person'].sum()
most_land_available = country_area_per_person.sort_values(ascending=False).head(5)
least_land_available = country_area_per_person.sort_values(ascending=False).tail(5)
```

In [35]:

```
# Create subplots
fig = sp.make_subplots(rows=1, cols=2, subplot_titles=("Countries with Most Land Available Per Capita", "Count

# Plot countries with the most land
fig.add_trace(go.Bar(x=most_land_available.index, y=most_land_available.values,
name='Most Land', marker_color=custom_palette[2]), row=1, col=1)

# Plot countries with the least land
fig.add_trace(go.Bar(x=least_land_available.index, y=least_land_available.values,
name='Least Land', marker_color=custom_palette[3]), row=1, col=2)

fig.update_layout(
title_text="Distribution of Available Land Area by Country Per Capita",
showlegend=False,
template='plotly_white'
)

fig.update_yaxes(title_text="Land Available Per Person", row=1, col=1)
fig.update_yaxes(title_text="Land Available Per Person", row=1, col=2)

fig.show()
```



Build Predective Model, Model Evaluation and Model Visualizations

In [38]:

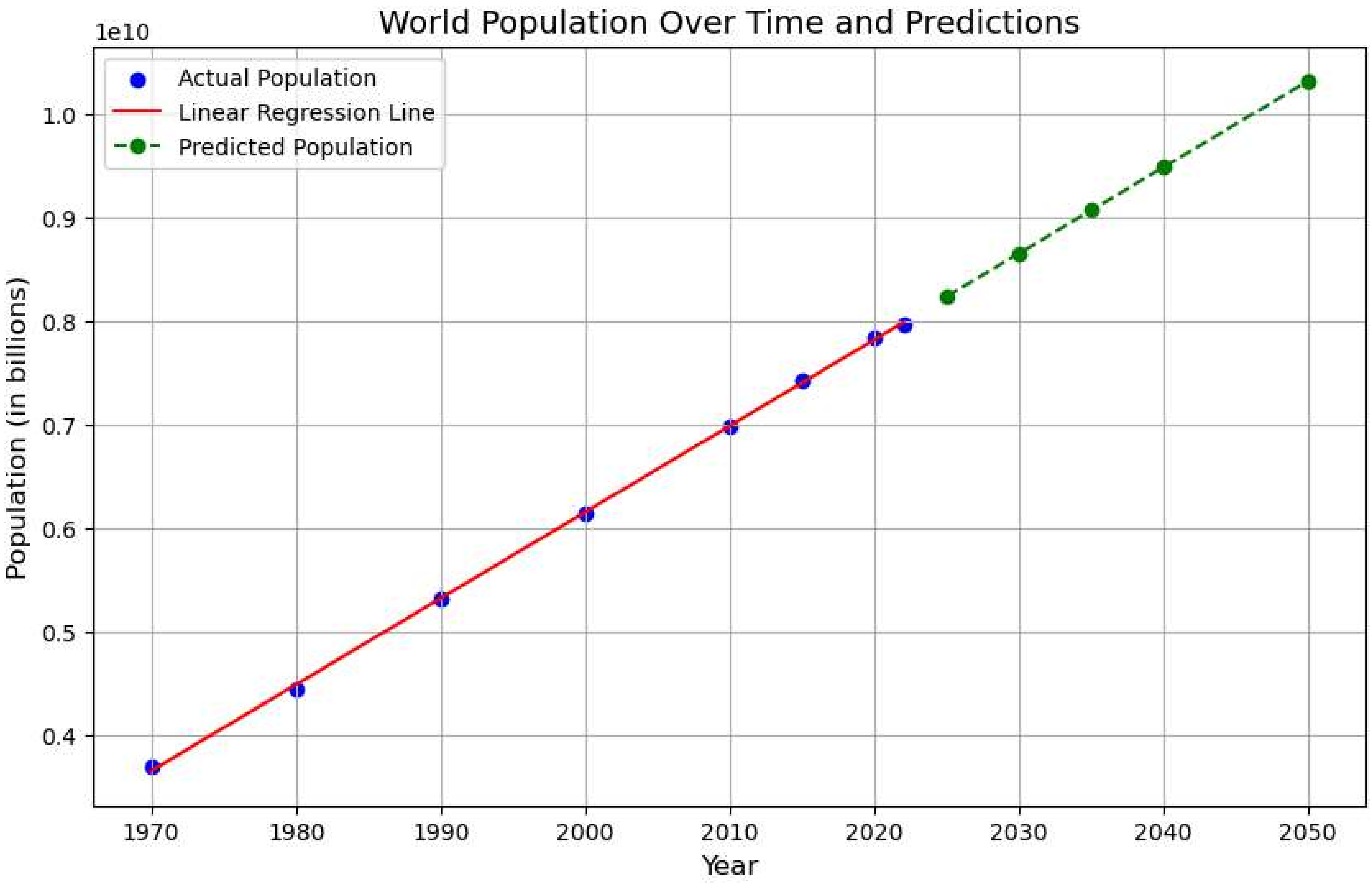
```python
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score

# Load the dataset
data = pd.read_csv('C:/Users/Admin/Downloads/world_population.csv')  # Adjust the file name

# Aggregate population by year for the entire world
world_population = {
    'Year': [1970, 1980, 1990, 2000, 2010, 2015, 2020, 2022],
    'Population': [
        data['1970 Population'].sum(),
        data['1980 Population'].sum(),
        data['1990 Population'].sum(),
        data['2000 Population'].sum(),
        data['2010 Population'].sum(),
        data['2015 Population'].sum(),
        data['2020 Population'].sum(),
        data['2022 Population'].sum(),
    ]
}

# Convert to a DataFrame
world_population_df = pd.DataFrame(world_population)

# Prepare the data for linear regression
X = world_population_df['Year'].values.reshape(-1, 1)  # Year as the independent variable
y = world_population_df['Population'].values  # World population as the dependent variable

# Create and train the model
model = LinearRegression()
model.fit(X, y)

# Predict future population
future_years = np.array([2025, 2030, 2035, 2040, 2050]).reshape(-1, 1)  # Years to predict
predictions = model.predict(future_years)

# Evaluate the model
y_pred = model.predict(X)
r2 = r2_score(y, y_pred)
mae = mean_absolute_error(y, y_pred)
```

```
# Output the results
print(f"Predictions for future years: {dict(zip(future_years.flatten(), predictions))}")
print(f"R-squared: {r2}")
print(f"Mean Absolute Error: {mae}")
```

```
Predictions for future years: {2025: 8238706813.63797, 2030: 8655562039.158783, 2035: 9072417264.679596, 204
0: 9489272490.200409, 2050: 10322982941.242035}
R-squared: 0.9997403143109639
Mean Absolute Error: 19512435.344406128
```

In [39]:

```
plt.figure(figsize=(10, 6))

# Plot historical population data
plt.scatter(world_population_df['Year'], world_population_df['Population'], color='blue', label='Actual Popula

# Plot the linear regression line
plt.plot(world_population_df['Year'], model.predict(X), color='red', linestyle='-', label='Linear Regression L

# Plot future predictions
plt.plot(future_years, predictions, color='green', marker='o', linestyle='--', label='Predicted Population')

# Labels and title
plt.title('World Population Over Time and Predictions', fontsize=14)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Population (in billions)', fontsize=12)
plt.legend()

# Show the plot
plt.grid(True)
plt.show()
```


World Population Over Time and Predictions
1e10
Actual Population
Linear Regression Line
Predicted Population
1.0
0.9
0.8
0.7
0.6
0.5
0.4
Population (in billions)
1970
1980
1990
2000
2010
2020
2030
2040
2050
Year

In [ ]:

In [ ]:

In [ ]:

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

CONCLUSION

This project successfully demonstrated the use of machine learning techniques to explore demographic data, identify key factors influencing population changes, and build predictive models. By employing a simple linear regression model, we were able to achieve meaningful insights and evaluate the model's performance with metrics such as R-squared and Mean Absolute Error. The outcomes confirm that even straightforward models can capture general population trends effectively.