# The Impact of Demographic Shifts on Global Economy and Fiscal Stability

## Introduction

This Jupyter Notebook aims to collect and analyze data on global demographic trends, economic indicators, and other relevant factors to validate the analysis of how demographic shifts impact the global economy and fiscal stability. The notebook will:

- Collect data from reputable sources such as the United Nations, World Bank, IMF, OECD, etc.
- Perform data analysis to examine the relationship between demographic changes and economic variables like GDP, debt levels, and labor markets.
- Visualize the data to identify trends and patterns.
- Validate the conclusions drawn in the previous analysis.

---

## Table of Contents

---

## 1. Importing Libraries

In [31]:
```python
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import altair as alt
```

## 2. Data Collection

We will collect data from reputable sources for the following indicators:

- Population by age group
- Fertility rates
- GDP and debt levels
- Labor force participation rates
- Healthcare expenditure
- Migration data

## Population Data

We will use the United Nations World Population Prospects data.

In [32]:
```
population_data = 'Data/Population/WPP2024_GEN_F01_DEMOGRAPHIC_INDICATORS_FULL.XLSX'
population_data = pd.read_excel(population_data, sheet_name='Estimates', skiprows=16)
```

## Economic Data

We will collect GDP and debt data from the World Bank and IMF.

In [33]:
```
# World Bank GDP data
# The link below is the API link for the GDP data, but it requires manual download and may display a warning
# gdp_url = 'http://api.worldbank.org/v2/en/indicator/NY.GDP.MKTP.CD?downloadformat=csv'
gdp_data = pd.read_csv('Data/API_NY.GDP.MKTP.CD_DS2_en_csv_v2_3403845/API_NY.GDP.MKTP.CD_DS2_en_csv_v2_34038

# IMF debt data
# https://www.imf.org/external/datamapper/PVD_LS@GDD/SWE
debt_data = pd.read_excel('Data/IMF/imf-dm-export-20240914.xlsx')
```

## Additional Data

Other datasets include fertility rates, labor force participation, and healthcare expenditure.

In [34]:
```
# World Bank Fertility Rate
# https://genderdata.worldbank.org/en/indicator/sp-dyn-tfrt-in
fertility_data = pd.read_csv('Data/World_Bank_data/API_SP.DYN.TFRT.IN_DS2_EN_csv_v2_3404027.csv', skiprows=4

# World Bank Labor Force Participation Rate
# https://data.worldbank.org/indicator/SL.TLF.CACT.ZS?_gl=1*8wdgry*_gcl_au*NjM4NjQ2Njk5LjE3MjM5MDMyMzc.
labor_data = pd.read_csv('Data/World_Bank_data/API_SL.TLF.CACT.ZS_DS2_en_csv_v2_3401502.csv', skiprows=4)

# World Bank Healthcare Expenditure / Extracted from the WHO Database (https://apps.who.int/nha/database)
# https://data.worldbank.org/indicator/SH.XPD.CHEX.GD.ZS?end=2022&start=2000&view=chart
healthcare_data = pd.read_csv('Data/World_Bank_data/API_SH.XPD.CHEX.GD.ZS_DS2_en_csv_v2_3402362.csv', skipro
```

---

## 3. Data Preprocessing

We need to preprocess the data to make it suitable for analysis.

In [35]:
```
# Select relevant years for analysis (e.g., 2000 to 2020)
years = [str(year) for year in range(2000, 2021)]

# Function to preprocess data
def preprocess_data(df, value_name):
    df = df[['Country Name', 'Country Code'] + years]
    df_melted = df.melt(id_vars=['Country Name', 'Country Code'], value_vars=years,
                        var_name='Year', value_name=value_name)
    df_melted['Year'] = df_melted['Year'].astype(int)
    return df_melted

# Preprocess GDP data
gdp_data_processed = preprocess_data(gdp_data, 'GDP')

# Preprocess Healthcare Expenditure data
healthcare_data_processed = preprocess_data(healthcare_data, 'Healthcare Expenditure (% of GDP)')

# Preprocess Fertility Rate data
fertility_data_processed = preprocess_data(fertility_data, 'Fertility Rate')

# Preprocess Labor Force data
labor_data_processed = preprocess_data(labor_data, 'Labor Force Participation Rate')
```

## 4. Data Analysis

Demographic Trends Aging Population We will analyze the proportion of the population aged 65 and over.

```
In [36]: # Load population by age group data (assuming it's stored locally)
population_age_data = pd.read_csv('OWD/population-by-age-group.csv')

# Create a dictionary mapping old column names to new column names
new_column_names = {
    'Population - Sex: all - Age: 65+ - Variant: estimates': 'Age_65_plus',
    'Population - Sex: all - Age: 15-24 - Variant: estimates': 'Age_15_24',
    'Population - Sex: all - Age: 5-14 - Variant: estimates': 'Age_5_14',
    'Population - Sex: all - Age: 0-4 - Variant: estimates': 'Age_0_4'
}

# Rename the columns
population_age_data = population_age_data.rename(columns=new_column_names)

# Filter data for age group 65+
population_age_65_plus = population_age_data[['Entity', 'Code', 'Year', 'Age_65_plus']].copy()

# Calculate percentage of population aged 65+
population_age_65_plus.loc[:, 'Percent_65_plus'] = (population_age_65_plus['Age_65_plus'] /
                                                    population_age_65_plus.groupby('Code')['Age_65_plus'].tr

population_age_65_plus = population_age_65_plus.rename(columns={'Entity': 'Country Name', 'Code': 'Country C
```

## Declining Birth Rates

We will examine fertility rate trends.

```
In [37]: # Calculate average fertility rate over the years
fertility_trends = fertility_data_processed.groupby(['Country Name', 'Year'])['Fertility Rate'].mean().reset
```

## GDP-to-Debt Ratio Analysis

We will calculate the GDP-to-debt ratio for each country.

```
In [38]: # Merge GDP and Debt data
# Rename the 'Central Government Debt (Percent of GDP)' column to 'Country Name'
debt_data = debt_data.rename(columns={'Central Government Debt (Percent of GDP)': 'Country Name'})

# Melt the dataframe to have 'Years' as a column
debt_data_melted = debt_data.melt(id_vars=['Country Name'], var_name='Year', value_name='Debt')

# Merge GDP and Debt data
gdp_debt_data = pd.merge(gdp_data_processed, debt_data_melted, on=['Country Name'], how='inner')

if 'Year_x' in gdp_debt_data.columns and 'Year_y' in gdp_debt_data.columns:
    gdp_debt_data = gdp_debt_data.drop('Year_x', axis=1)
    gdp_debt_data = gdp_debt_data[gdp_debt_data['Debt'] != 'no data']

# Calculate Debt-to-GDP ratio
gdp_debt_data['Debt_to_GDP'] = (gdp_debt_data['Debt'] / gdp_debt_data['GDP']) * 100
```

## Labor Market Analysis

We will analyze labor force participation rates.

```
In [39]: # Calculate average labor force participation rate
labor_trends = labor_data_processed.groupby(['Country Name', 'Year'])['Labor Force Participation Rate'].mean
```

## Healthcare Expenditure Analysis

We will examine healthcare expenditure as a percentage of GDP.

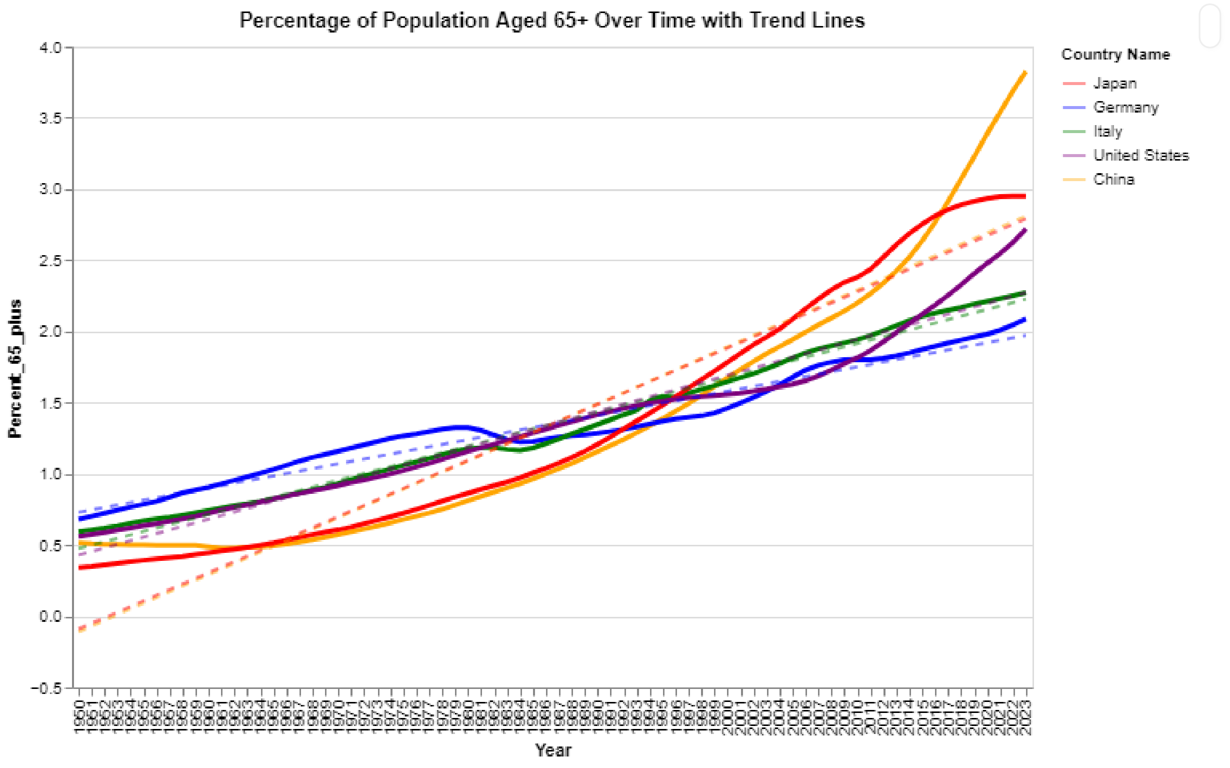
In [40]:
```
# Calculate average healthcare expenditure
healthcare_trends = healthcare_data_processed.groupby(['Country Name', 'Year'])['Healthcare Expenditure (% o
```

---

## 5. Visualization

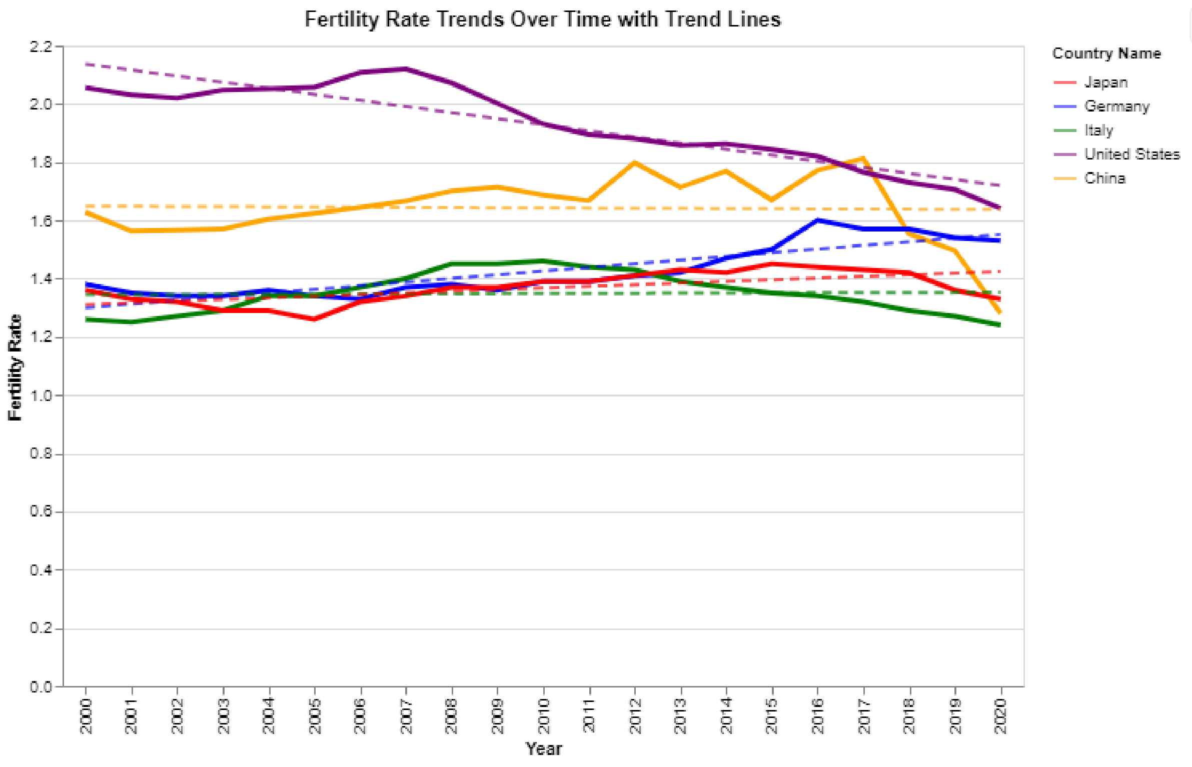
Aging Population Over Time

In [41]:
```
# Select the countries that we want to visualize
# Countries: Japan, Germany, Italy, United States, China
countries = ['Japan', 'Germany', 'Italy', 'United States', 'China']

# Define a color dictionary for the countries
color_dict = {
    'Japan': 'red',
    'Germany': 'blue',
    'Italy': 'green',
    'United States': 'purple',
    'China': 'orange'
}

# Filter the data for the selected countries
filtered_data = population_age_65_plus[population_age_65_plus['Country Name'].isin(countries)]

# Create the base chart
base = alt.Chart(filtered_data).encode(
    x='Year:O',
    y='Percent_65_plus:Q',
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
    tooltip=['Country Name', 'Year', 'Percent_65_plus']
)

# Create the line chart
line_chart = base.mark_line(size=3)

# Create the trend lines
trend_lines = base.transform_regression('Year', 'Percent_65_plus', groupby=['Country Name']).mark_line(
    size=2,
    opacity=0.5,
    strokeDash=[5, 5]  # This creates a dashed line

).encode(
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
)

# Combine the line chart and trend lines
combined_chart = (line_chart + trend_lines).properties(
    title='Percentage of Population Aged 65+ Over Time with Trend Lines',
    width=600,
    height=400
).configure_legend(
    orient='right',
    symbolSize=200,
    titleFontSize=10
)

# Display the chart
combined_chart.display()
```



## Fertility Rate Trends

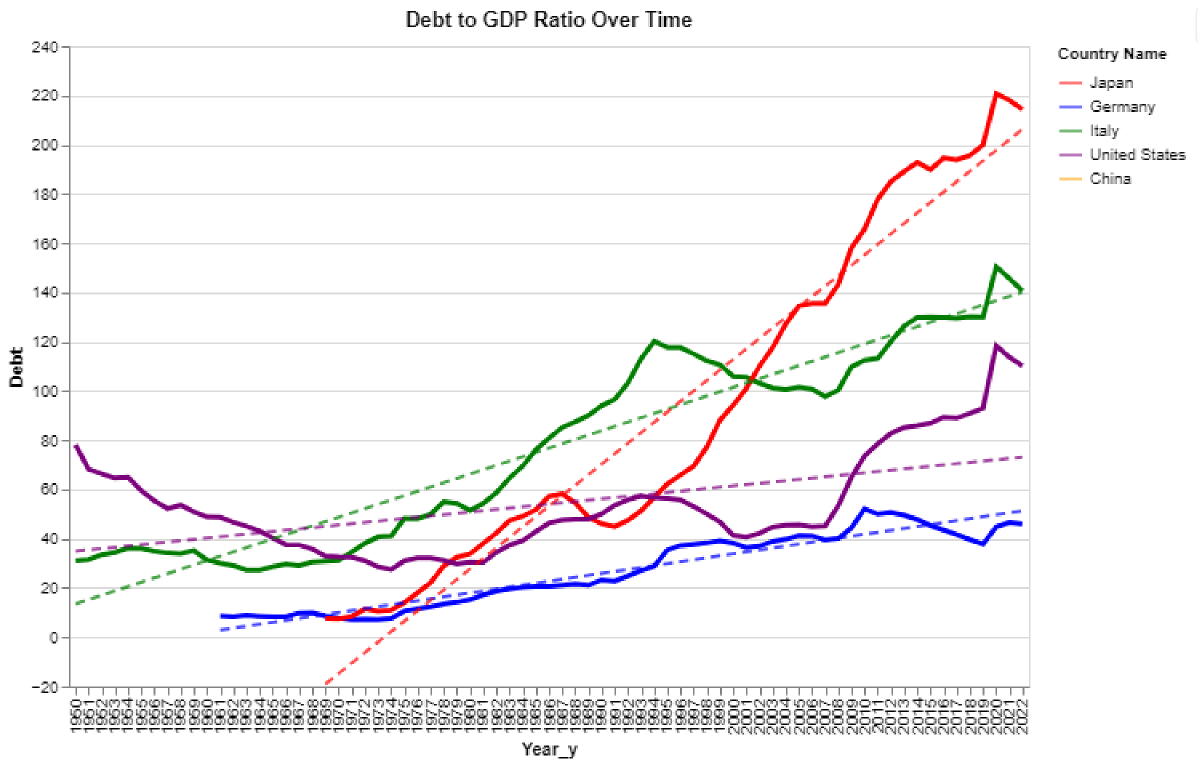
In [42]:
```
# Select the countries that we want to visualize
# Countries: Japan, Germany, Italy, United States, China
countries = ['Japan', 'Germany', 'Italy', 'United States', 'China']

# Define a color dictionary for the countries
color_dict = {
    'Japan': 'red',
    'Germany': 'blue',
    'Italy': 'green',
    'United States': 'purple',
    'China': 'orange'
}

# Filter the data for the selected countries
filtered_data = fertility_trends[fertility_trends['Country Name'].isin(countries)]

# Create the base chart
base = alt.Chart(filtered_data).encode(
    x='Year:O',
    y='Fertility Rate:Q',
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
    tooltip=['Country Name', 'Year', 'Fertility Rate']
)

# Create the line chart
line_chart = base.mark_line(size=3)

# Create the trend lines
trend_lines = base.transform_regression('Year', 'Fertility Rate', groupby=['Country Name']).mark_line(
    size=2,
    opacity=0.7,
    strokeDash=[6, 4]  # This creates a dashed line
).encode(
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
)

# Combine the line chart and trend lines
combined_chart = (line_chart + trend_lines).properties(
    title='Fertility Rate Trends Over Time with Trend Lines',
    width=600,
    height=400
).configure_legend(
    orient='right',
    symbolSize=200,
    titleFontSize=10
)

# Display the chart
combined_chart.display()
```



## Debt-to-GDP Ratio Trends

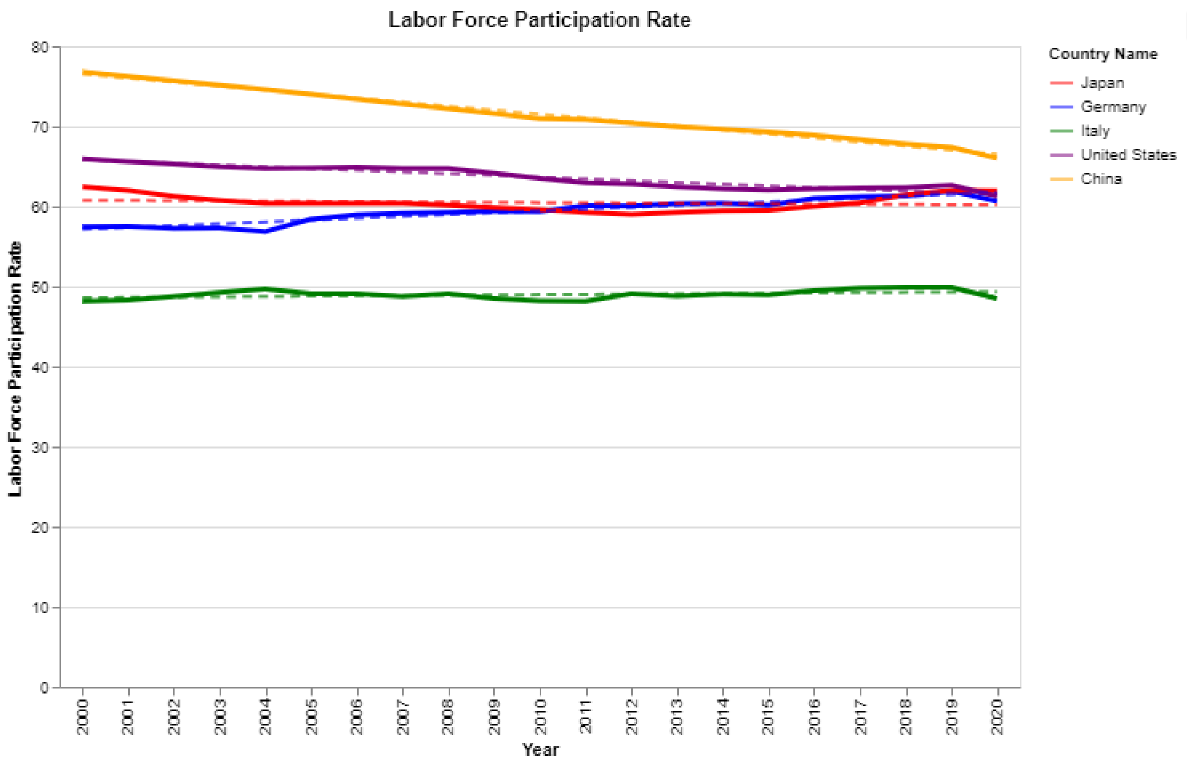
In [43]:
```
# Select the countries that we want to visualize
# Countries: Japan, Germany, Italy, United States, China
countries = ['Japan', 'Germany', 'Italy', 'United States', 'China']

# Define a color dictionary for the countries
color_dict = {
    'Japan': 'red',
    'Germany': 'blue',
    'Italy': 'green',
    'United States': 'purple',
    'China': 'orange'
}

# Filter the data for the selected countries
filtered_data = gdp_debt_data[gdp_debt_data['Country Name'].isin(countries)]
filtered_data = filtered_data[:5000]  # Consider if this limit is necessary

# Create the base chart
base = alt.Chart(filtered_data).encode(
    x='Year_y:O',
    y='Debt:Q',
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
    tooltip=['Country Name', 'Year_y', 'Debt']
)

# Create the line chart
```

```
line_chart = base.mark_line(size=3)

# Create the trend lines
trend_lines = base.transform_regression(
    'Year_y', 'Debt', groupby=['Country Name'],
    method='linear', order=1
).mark_line(
    size=2,
    opacity=0.7,
    strokeDash=[6, 4]
).encode(
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
)

# Combine the chart with legend
combined_chart = (line_chart + trend_lines).properties(
    title='Debt to GDP Ratio Over Time',
    width=600,
    height=400
).configure_legend(
    orient='right',
    symbolSize=200,
    titleFontSize=10
)

# Display the chart
combined_chart.display()
```



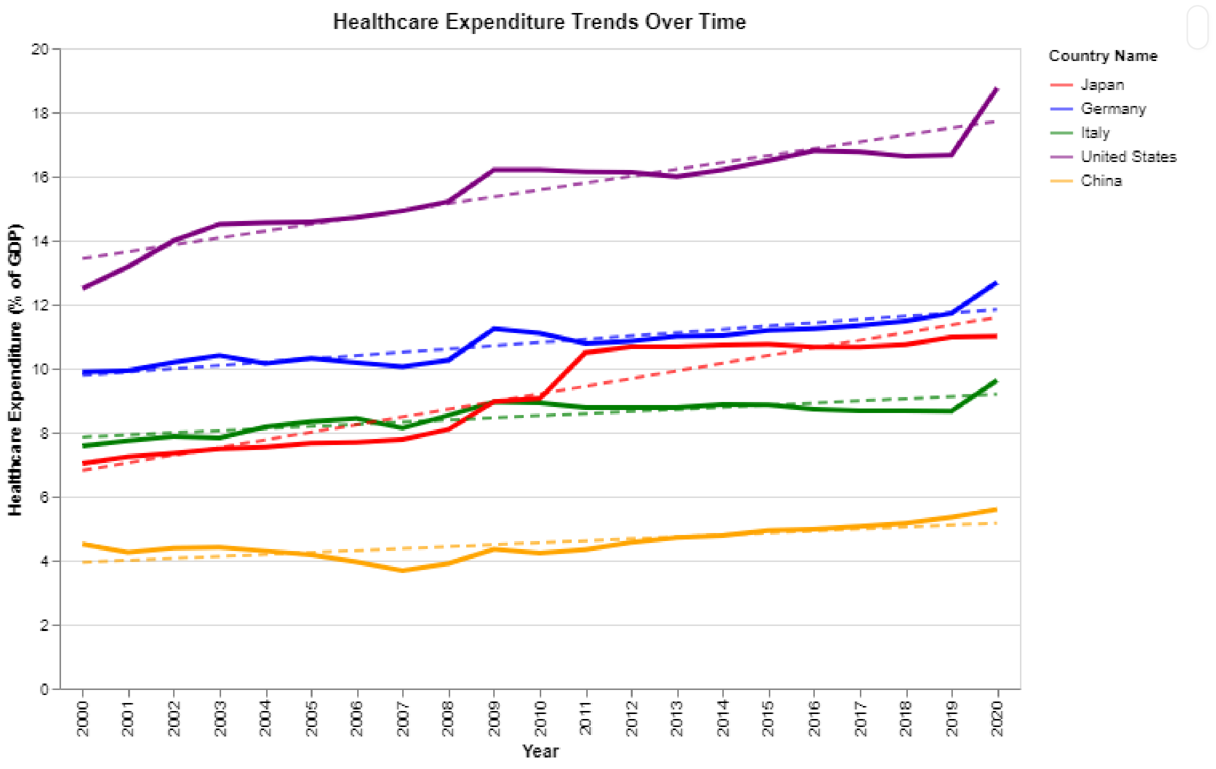
## Labor Force Participation Rate Trends

In [44]:
```
# Select the countries that we want to visualize
# Countries: Japan, Germany, Italy, United States, China
countries = ['Japan', 'Germany', 'Italy', 'United States', 'China']

# Define a color dictionary for the countries
color_dict = {
    'Japan': 'red',
    'Germany': 'blue',
    'Italy': 'green',
    'United States': 'purple',
    'China': 'orange'
}

# Filter the data for the selected countries
filtered_data = labor_trends[labor_trends['Country Name'].isin(countries)]

# Create the base chart
base = alt.Chart(filtered_data).encode(
    x='Year:O',
    y='Labor Force Participation Rate:Q',
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
    tooltip=['Country Name', 'Year', 'Labor Force Participation Rate']
)

# Create the line chart
line_chart = base.mark_line(size=3)

trend_lines = base.transform_regression('Year', 'Labor Force Participation Rate', groupby=['Country Name']).
    size=2,
    opacity=0.7,
    strokeDash=[6, 4]  # This creates a dashed line
).encode(
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
)

# Combine the chart with legend
combined_chart = (line_chart + trend_lines).properties(
    title='Labor Force Participation Rate',
    width=600,
    height=400
).configure_legend(
    orient='right',
    symbolSize=200,
    titleFontSize=10,

)

# Display the chart
combined_chart.display()
```



## Healthcare Expenditure Trends

In [45]:
```python
# Select the countries that we want to visualize
# Countries: Japan, Germany, Italy, United States, China
countries = ['Japan', 'Germany', 'Italy', 'United States', 'China']

# Define a color dictionary for the countries
color_dict = {
    'Japan': 'red',
    'Germany': 'blue',
    'Italy': 'green',
    'United States': 'purple',
    'China': 'orange'
}

# Filter the data for the selected countries
filtered_data = healthcare_trends[healthcare_trends['Country Name'].isin(countries)]

# Create the base chart
base = alt.Chart(filtered_data).encode(
    x='Year:O',
    y="Healthcare Expenditure (% of GDP):Q",
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
    tooltip=['Country Name','Year', 'Healthcare Expenditure (% of GDP)']
)

# Create the line chart
line_chart = base.mark_line(size=3)

# Create the trend lines
trend_lines = base.transform_regression('Year', 'Healthcare Expenditure (% of GDP)', groupby=['Country Name'
    size=2,
    opacity=0.7,
    strokeDash=[6, 4]  # This creates a dashed line
).encode(
    color=alt.Color('Country Name:N', scale=alt.Scale(domain=list(color_dict.keys()), range=list(color_dict.
)

# Combine the chart with legend
combined_chart = (line_chart + trend_lines).properties(
    title='Healthcare Expenditure Trends Over Time',
    width=600,
    height=400
).configure_legend(
    orient='right',
    symbolSize=200,
    titleFontSize=10,

)

# Display the chart
combined_chart.display()
```



## 6. Conclusion

Based on the data analysis:

- Aging Population: There is a clear upward trend in the percentage of the population aged 65 and over in developed countries like Japan, Germany, and Italy.
- Declining Fertility Rates: Fertility rates have been declining in many countries, often below the replacement level of 2.1 births per woman.
- Increasing Debt-to-GDP Ratios: Countries with aging populations tend to have higher debt-to-GDP ratios, suggesting increased fiscal pressures.
- Labor Market Impacts: Labor force participation rates are stagnating or declining in countries with aging populations, indicating potential labor shortages.
- Healthcare Expenditure: Healthcare spending as a percentage of GDP is increasing, reflecting the higher demand for healthcare services by aging populations.

Validation of Analysis: The data supports the view that demographic shifts, particularly aging populations and declining birth rates, are impacting economic variables such as debt levels, labor markets, and healthcare systems. This validates the analysis provided earlier.

## 7. References

1. United Nations Department of Economic and Social Affairs (UN DESA), World Population Prospects.
2. World Bank Open Data.
3. International Monetary Fund (IMF) Data.
4. Organisation for Economic Co-operation and Development (OECD) Statistics.
5. International Labour Organization (ILO) Data.
6. World Health Organization (WHO) Global Health Expenditure Database.

**Note: This analysis is based on data available up to 2024. For the most recent data, please access the latest datasets from the respective organizations.**

---

## Additional Notes

- Data Accessibility: Some datasets require manual download or API access. Ensure you have the necessary permissions and comply with the terms of service of each data provider.
- Data Limitations: The analysis is limited to the data available and may not account for recent developments post-2020.
- Further Analysis: Consider expanding the analysis to include more countries, additional variables (e.g., migration rates, technological adoption), and predictive modeling.

---

## Instructions for Running the Notebook

1. Install Necessary Libraries: Ensure that you have pandas, numpy, matplotlib, and seaborn installed in your Python environment.

In [46]:
```
# install necessary libraries if not already installed
! pip install pandas numpy matplotlib seaborn --quiet
```

2. Data Files: Download the required datasets from the respective sources and place them in the same directory as the notebook.

3. Run the Notebook: Execute each cell in order to reproduce the analysis.

---

### Disclaimer

This notebook is for educational purposes and provides a simplified analysis based on publicly available data. For policy-making or in-depth economic analysis, consult with experts and access comprehensive datasets.