Exploratory Data Analysis (EDA)

In this section, we conduct Exploratory Data Analysis (EDA) on the dataset to discover patterns, trends, and relationships.
Using Python libraries like Pandas, Matplotlib, and Seaborn, we visualize and statistically explore the data to extract valuable insights.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style('whitegrid')
plt.rcParams['figure.figsize'] = (10,6)
```

Step 1: Importing Libraries

We start by importing the essential Python libraries for data manipulation, analysis, and visualization:

- **pandas** for data handling and operations.
- **numpy** for numerical computations.
- **matplotlib.pyplot** and **seaborn** for creating attractive data visualizations.

We also set the default style for our plots to 'whitegrid' for better readability and adjust the figure size for consistency across visualizations.

```
# Read the CSV
df = pd.read_csv('/content/train (1).csv')
```

Step 2: Loading the Dataset

We proceed by loading the dataset into a pandas DataFrame to begin our analysis.
The dataset is read from a CSV file using the `read_csv()` function

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarke
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	
4	5	0	3	Allen, Mr. William	male	35.0	0	0	373450	8.0500	NaN	

Next steps: Generate code with df View recommended plots New interactive sheet

Step 3: Viewing the First Few Rows

To get an initial understanding of the dataset, we use the `head()` function to display the first few rows.

- This allows us to observe the structure of the data, column names, and data types.
- We can also spot any immediate issues such as missing values or inconsistent data entries.

From the preview, we notice important features like **PassengerId**, **Survived**, **Pclass**, **Name**, **Sex**, **Age**, **Fare**, and others that will be crucial for our exploratory data analysis.

Double-click (or enter) to edit

```
# Shape of data
print(f"Shape of dataset: {df.shape}")

# Data types and missing values
df.info()

# Summary statistics
df.describe()

# Unique values in each column
df.nunique()

# Checking missing values
df.isnull().sum()
```

```
Shape of dataset: (891, 12)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

	0
PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	177
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	687
Embarked	2

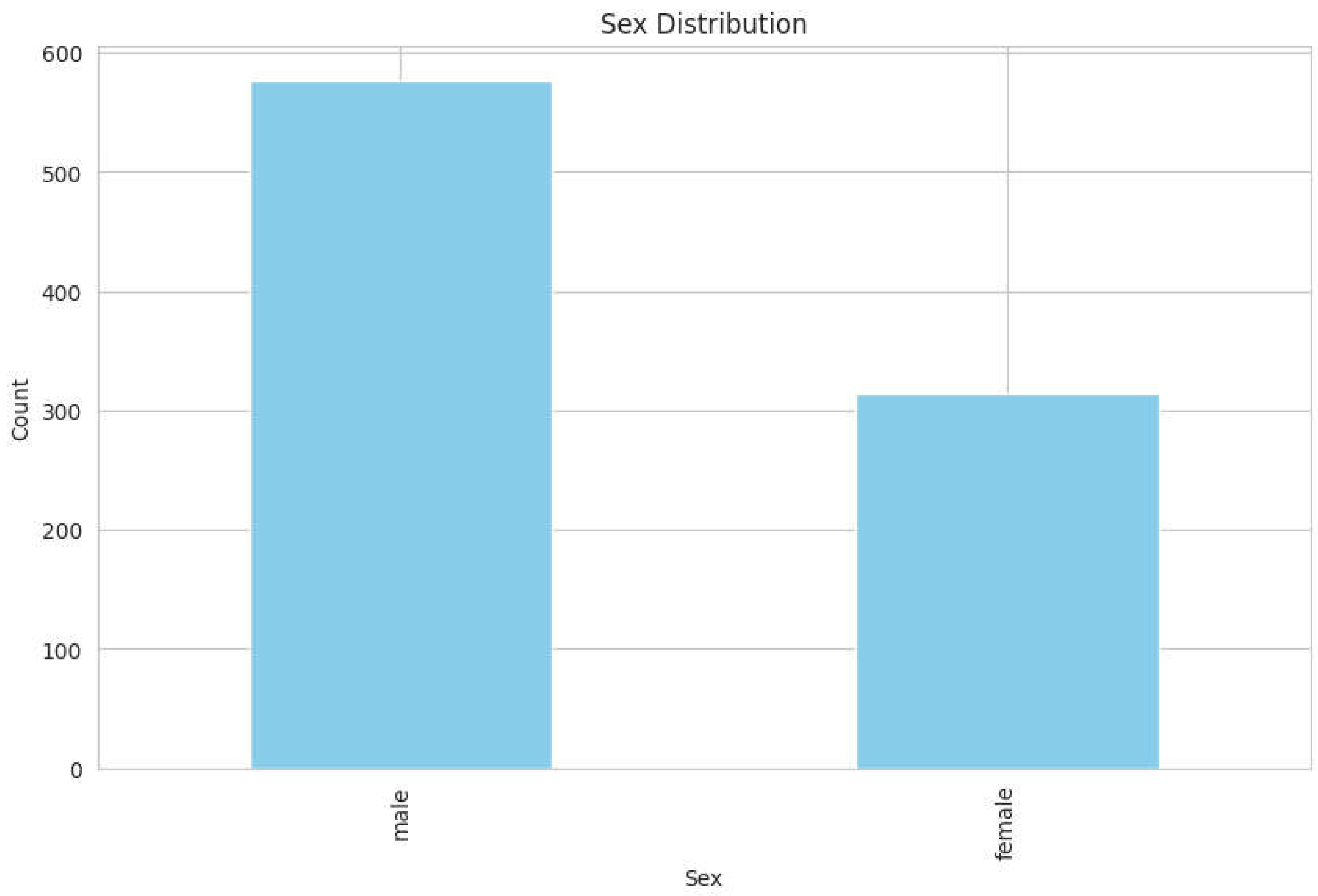
Step 4: Basic Information about the Dataset

In this step, we explore some fundamental characteristics of the dataset:

- **Shape of the dataset**: We have 891 rows and 12 columns.
- **Data types and missing values**: Using the `info()` method, we observe the data types and identify any columns with missing values.
- **Summary statistics**: The `describe()` function provides key statistical measures such as mean, median, min, and max for numerical columns.
- **Unique values**: The `nunique()` function helps us understand the distinct number of entries in each column.
- **Missing values**: Checking missing values is important to handle data quality issues during analysis.

This initial exploration helps us plan further steps for cleaning and analyzing the data effectively.

```
# Sex Distribution
df['Sex'].value_counts().plot(kind='bar', color='skyblue')
plt.title('Sex Distribution')
plt.xlabel('Sex')
plt.ylabel('Count')
plt.show()
```



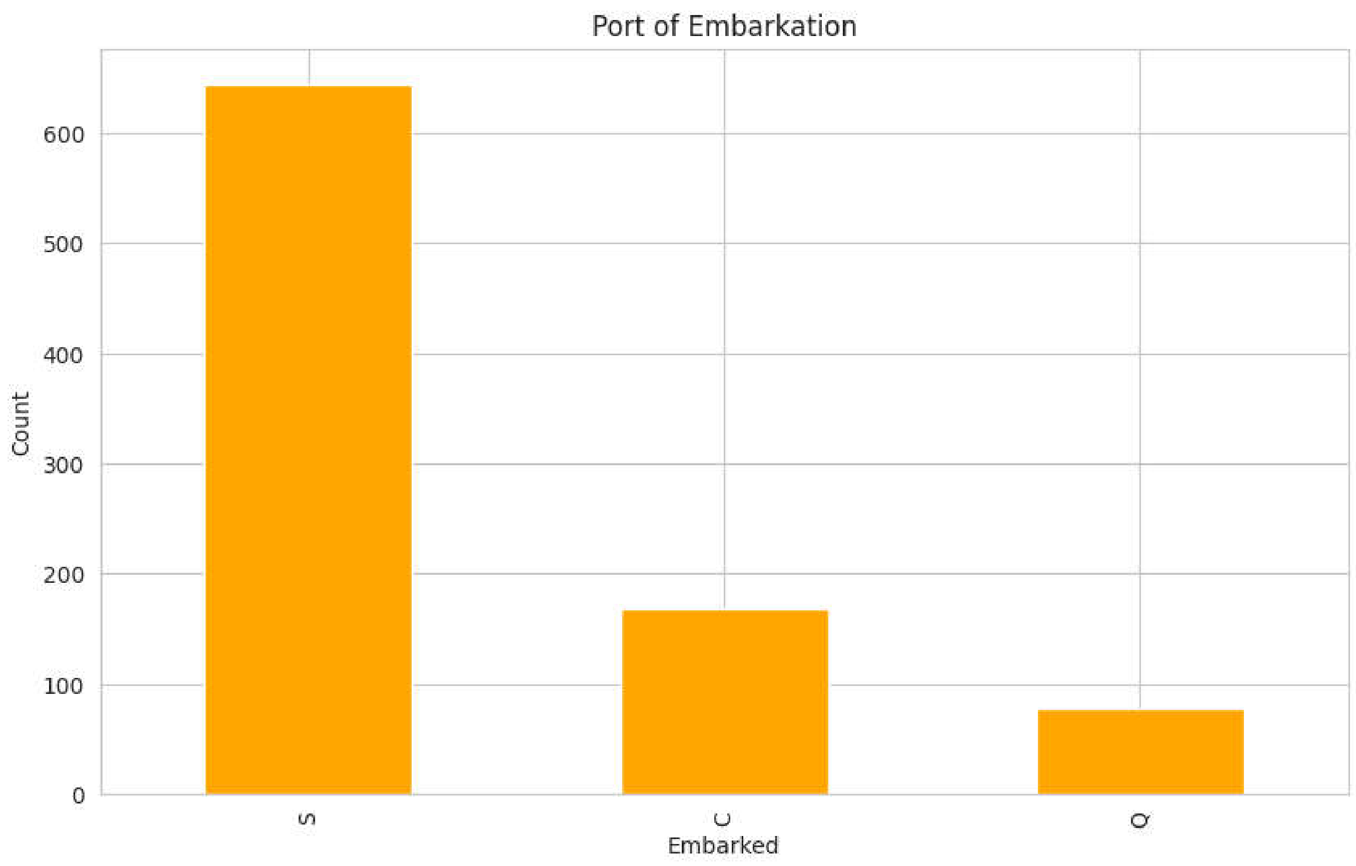
Step 5: Univariate Analysis - Sex Distribution

We start the univariate analysis by examining the distribution of passengers based on their sex.

- The bar plot shows the count of male and female passengers in the dataset.
- It is evident that there are significantly more male passengers compared to female passengers.
- Understanding the distribution of categorical features like Sex is important for further survival analysis.

This insight will help in analyzing how gender may have influenced survival chances on the Titanic.

```
# Embarked
df['Embarked'].value_counts().plot(kind='bar', color='orange')
plt.title('Port of Embarkation')
plt.xlabel('Embarked')
plt.ylabel('Count')
plt.show()
```



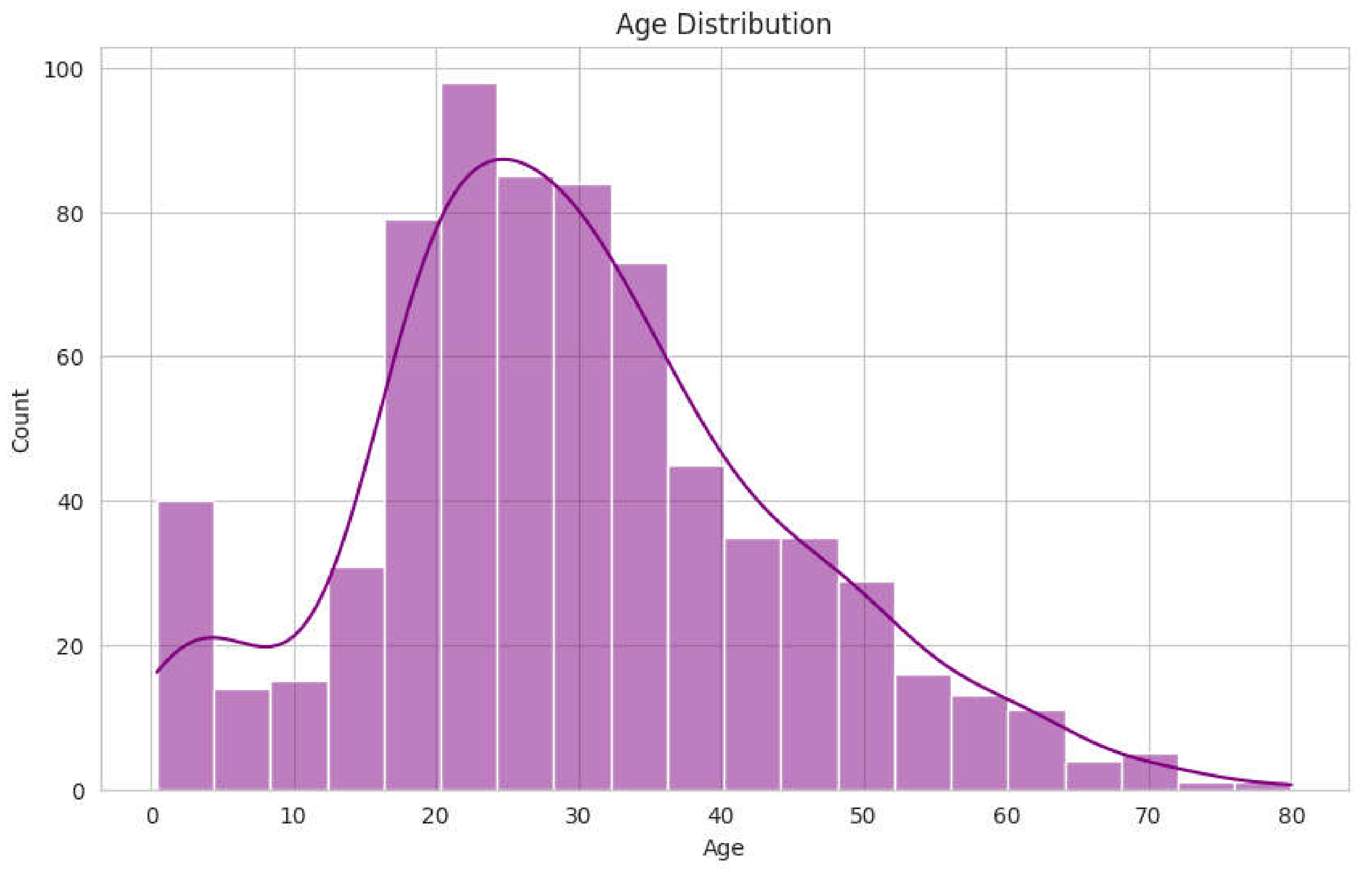
Step 6: Univariate Analysis - Port of Embarkation

Next, we analyze the distribution of passengers based on their port of embarkation.

- The bar plot shows the number of passengers who boarded from each port.
- Most passengers embarked from one particular port (likely 'S' - Southampton).
- Analyzing the embarkation points can provide insights into the demographics and journey patterns of the passengers.

This information may also help in understanding survival patterns based on embarkation points.

```
# Age Distribution
sns.histplot(df['Age'].dropna(), kde=True, color='purple')
plt.title('Age Distribution')
plt.show()
```



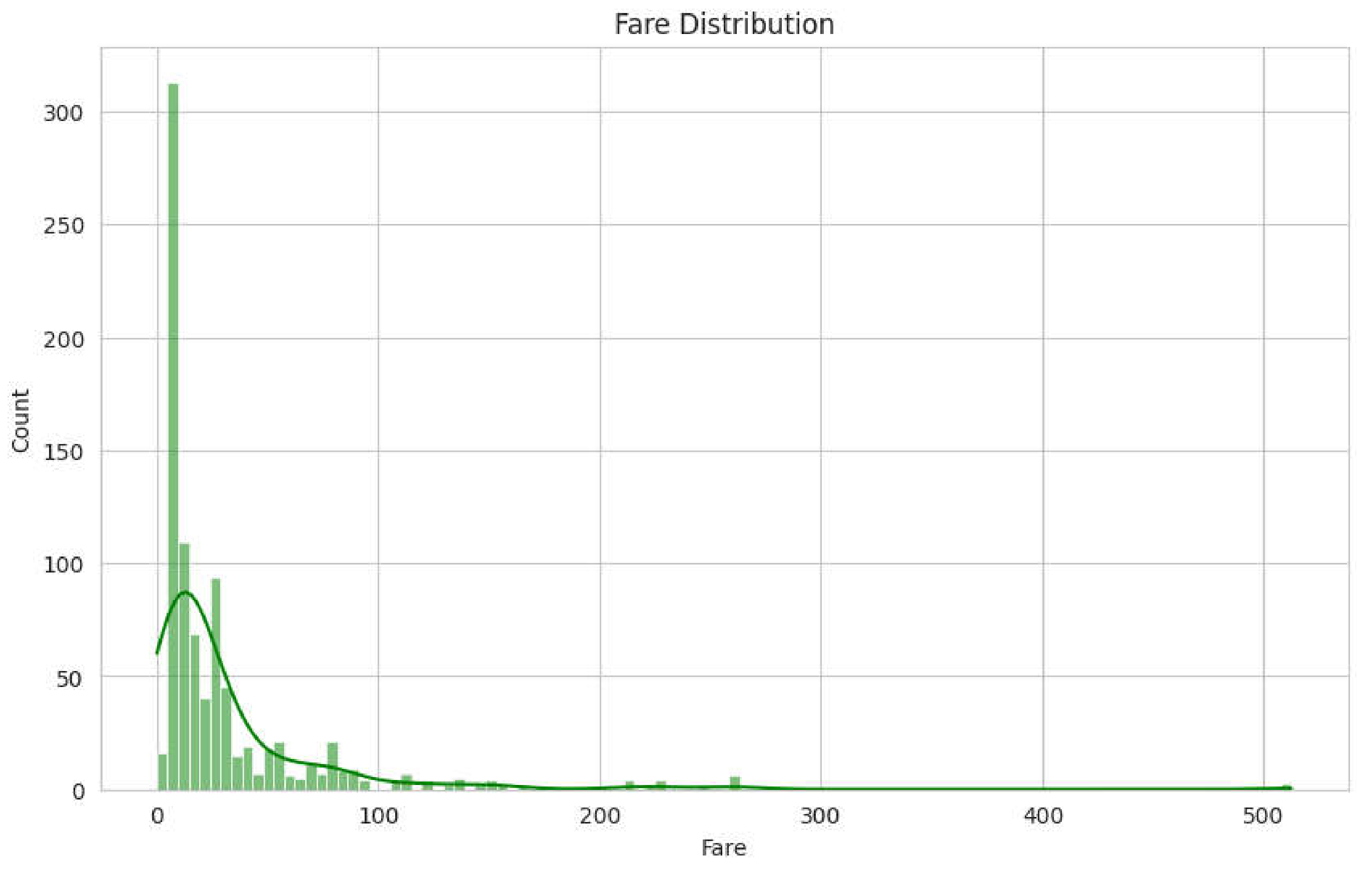
Step 7: Univariate Analysis - Age Distribution

We now examine the distribution of passengers' ages.

- The histogram shows the age distribution, with a KDE (Kernel Density Estimate) curve to understand the spread more clearly.
- Most passengers were between 20 and 40 years old.
- The distribution is slightly right-skewed, meaning there were more younger passengers than older ones.

Analyzing age distribution helps us understand the demographics on board and may reveal patterns related to survival rates.

```
# Fare Distribution
sns.histplot(df['Fare'], kde=True, color='green')
plt.title('Fare Distribution')
plt.show()
```



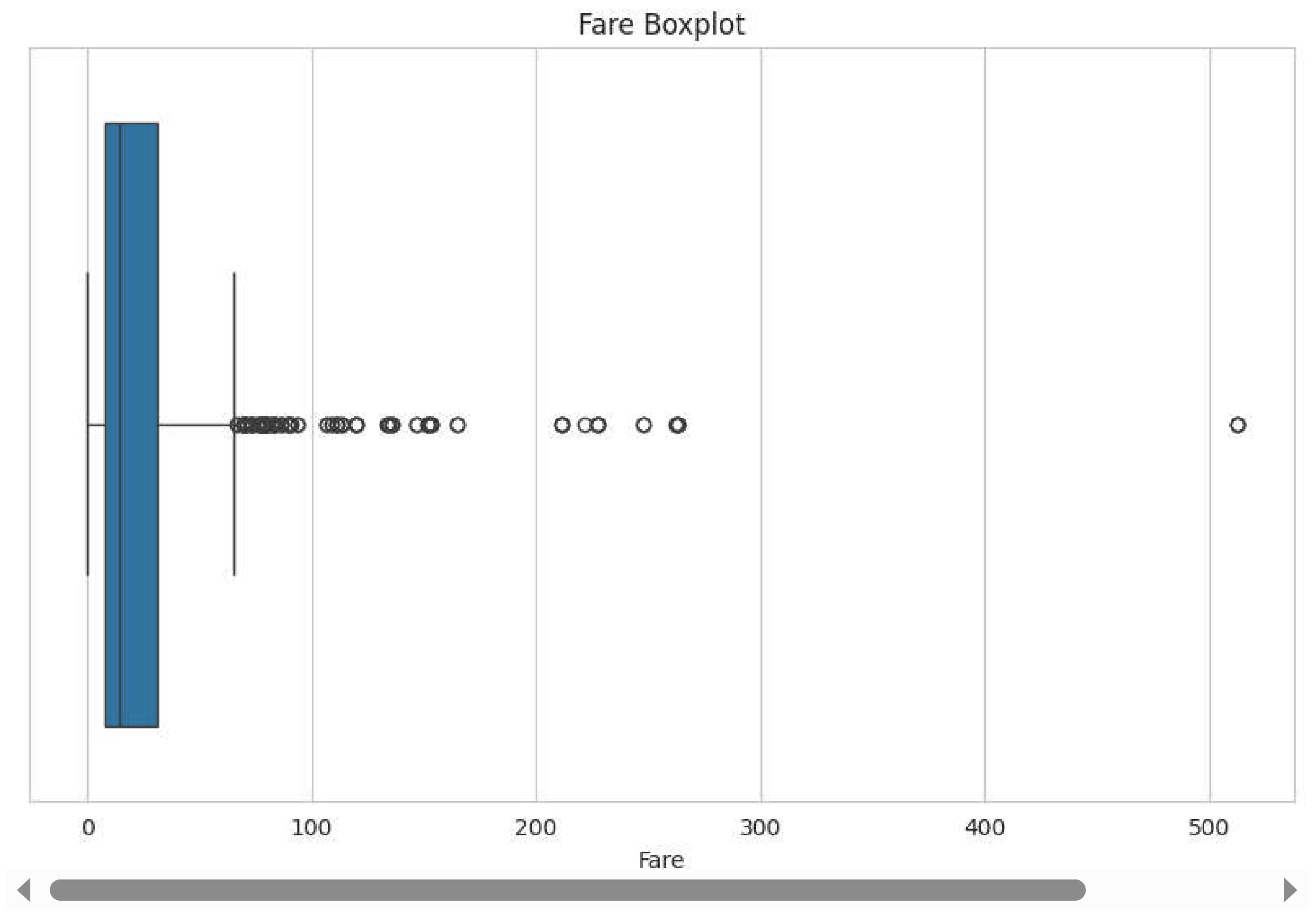
Step 8: Univariate Analysis - Fare Distribution

We now analyze the distribution of fares paid by the passengers.

- The histogram shows that most passengers paid lower fares, with a long tail towards higher fare amounts.
- The distribution is highly right-skewed, indicating that while most passengers paid relatively low fares, a few paid very high fares.
- The presence of outliers suggests that some passengers traveled in much more expensive classes or with extra services.

Understanding fare distribution is important because fare could be related to passenger class and survival probability.

```
# Boxplot for Fare
sns.boxplot(x=df['Fare'])
plt.title('Fare Boxplot')
plt.show()
```



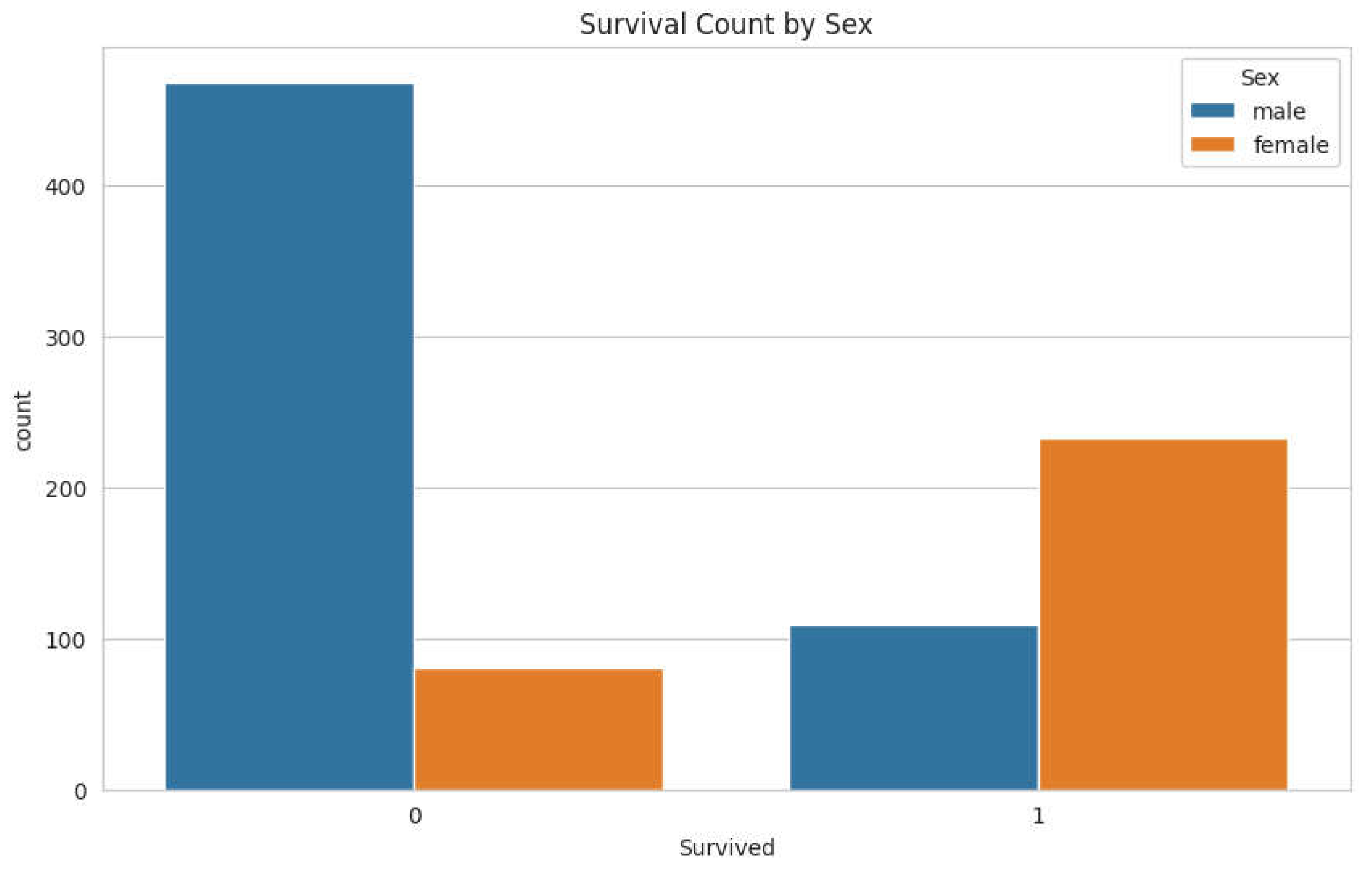
Step 9: Univariate Analysis - Fare Boxplot

We now visualize the fare distribution using a boxplot.

- The boxplot shows that most passengers paid lower fares, with a few passengers paying significantly higher amounts.
- The distribution has several outliers, indicating a wide variation in fare prices.
- The presence of extreme fare values suggests that some passengers traveled in more luxurious classes or purchased additional services.

Understanding the fare distribution through a boxplot helps in detecting anomalies and understanding the spread of ticket prices among different passengers.

```
# Survival based on Sex
sns.countplot(x='Survived', hue='Sex', data=df)
plt.title('Survival Count by Sex')
plt.show()
```



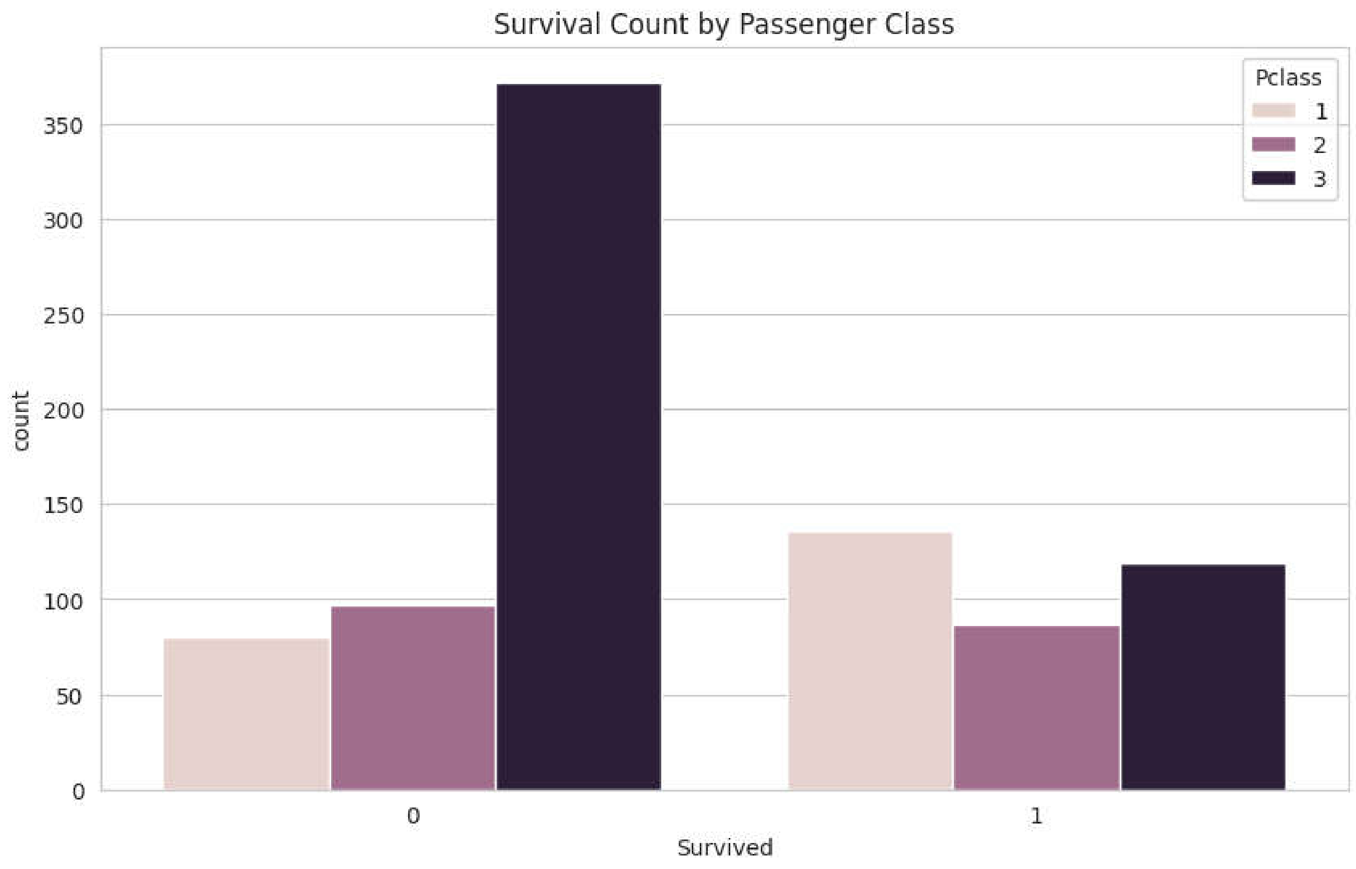
Step 10: Bivariate Analysis - Survival Based on Sex

We now analyze the survival rates based on the sex of the passengers.

- The countplot shows that a higher proportion of females survived compared to males.
- A majority of the male passengers did not survive, while a significant number of female passengers survived.
- This suggests that gender played an important role in survival chances, possibly due to the "women and children first" evacuation protocol.

Understanding survival patterns by gender provides important insights into social behavior during the disaster.

```
# Survival based on Pclass
sns.countplot(x='Survived', hue='Pclass', data=df)
plt.title('Survival Count by Passenger Class')
plt.show()
```



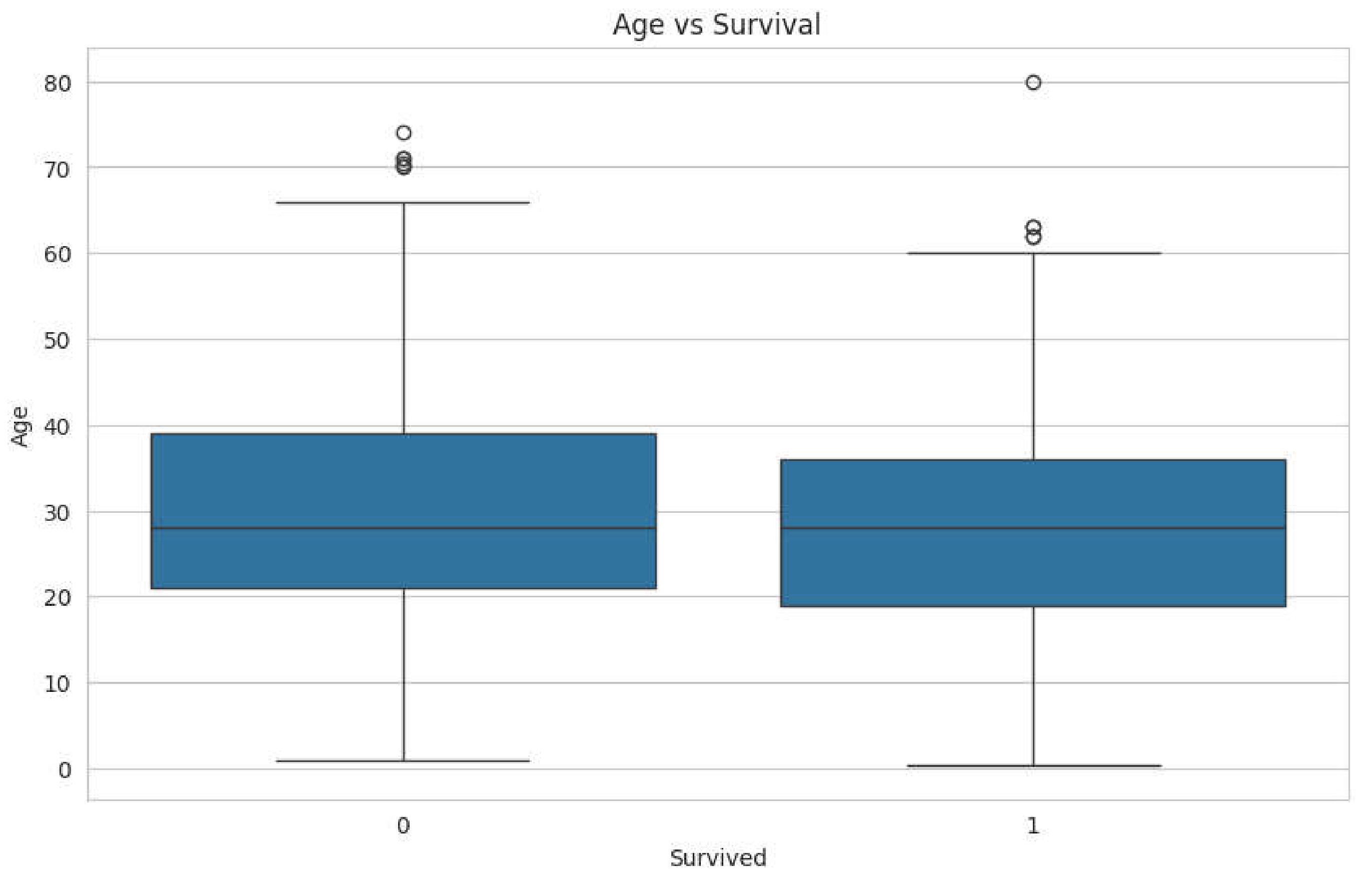
Step 11: Bivariate Analysis - Survival Based on Passenger Class

We now analyze how survival rates varied across different passenger classes.

- The countplot shows that passengers from higher classes (Class 1) had a better survival rate compared to lower classes.
- Passengers in Class 3 had the highest number of non-survivors.
- This indicates that socioeconomic status (as reflected by ticket class) played a significant role in survival probability.

Analyzing survival based on passenger class highlights the impact of wealth and privilege during emergency situations.

```
# Age vs Survival
sns.boxplot(x='Survived', y='Age', data=df)
plt.title('Age vs Survival')
plt.show()
```



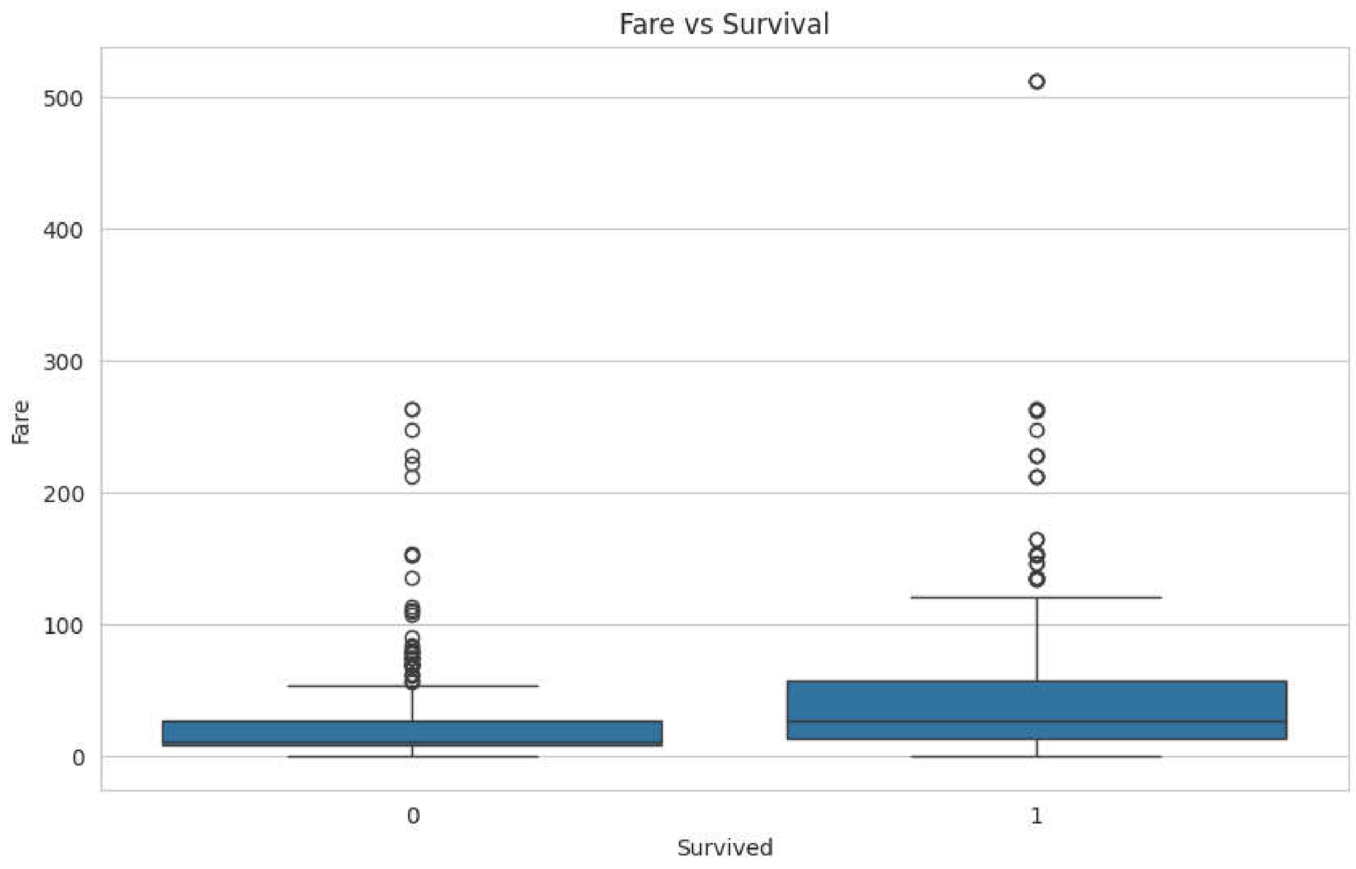
Step 12: Bivariate Analysis - Age vs Survival

We now analyze the age distribution among passengers who survived and those who did not.

- The boxplot shows that the median age of survivors and non-survivors is quite similar, slightly lower for survivors.
- Younger passengers (especially children) appear to have had slightly better survival rates, although survival was not strictly dependent on age.
- There are several outliers among older passengers, some of whom survived despite higher age.

Analyzing survival based on age helps us understand if younger passengers had any advantage during the evacuation.

```
# Fare vs Survival
sns.boxplot(x='Survived', y='Fare', data=df)
plt.title('Fare vs Survival')
plt.show()
```



Step 13: Bivariate Analysis - Fare vs Survival

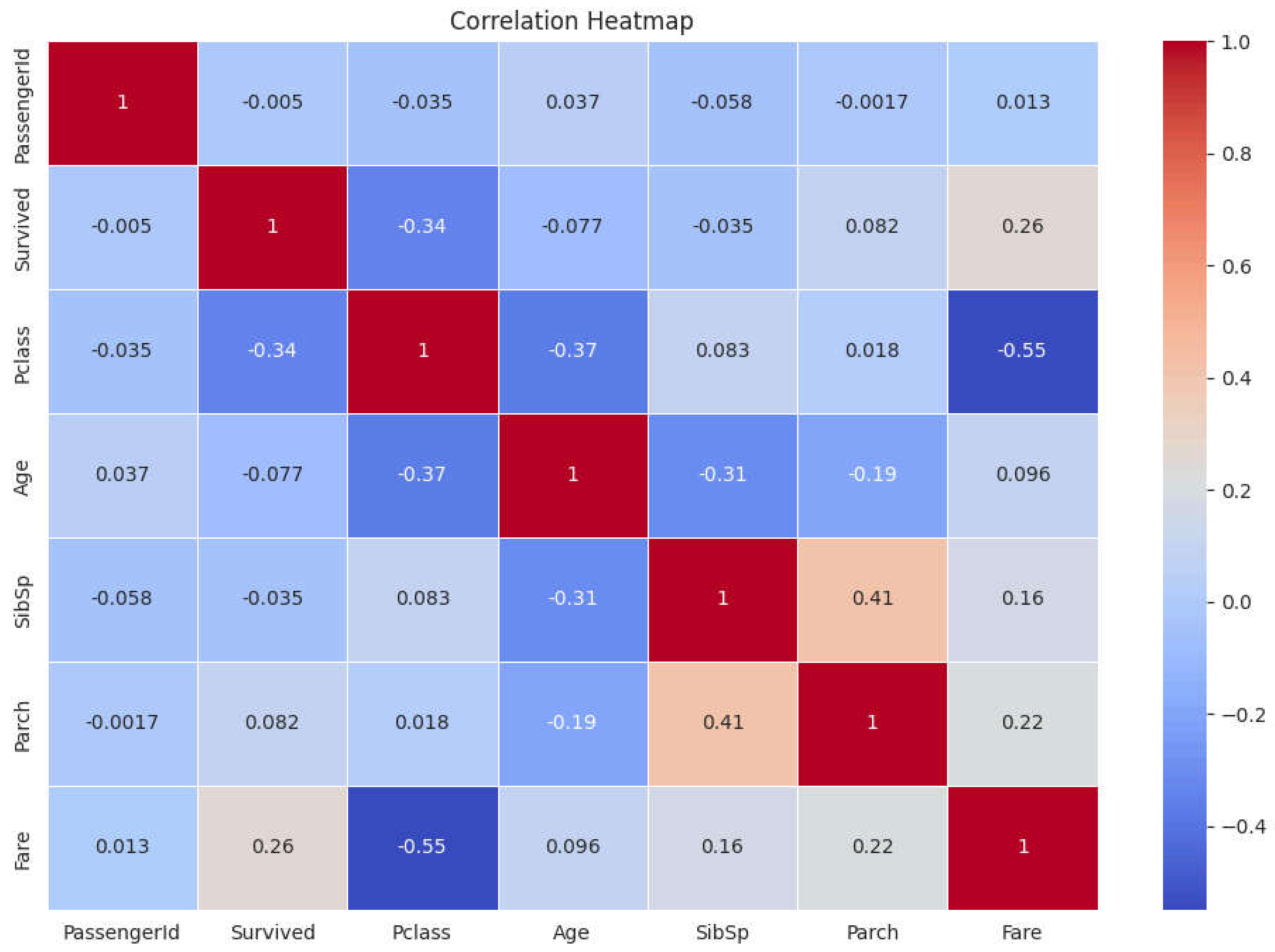
We now analyze how fare amounts relate to survival probability.

- The boxplot shows that passengers who paid higher fares had a higher chance of survival compared to those who paid lower fares.
- The median fare for survivors is noticeably higher than that for non-survivors.
- There are many outliers, indicating that some passengers with very high fares had varying survival outcomes.

This analysis suggests that fare price, which is linked to passenger class and access to resources, impacted survival likelihood.

```
numeric_df = df.select_dtypes(include=['number'])

# Now plot heatmap
plt.figure(figsize=(12,8))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```



	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1	-0.005	-0.035	0.037	-0.058	-0.0017	0.013
Survived	-0.005	1	-0.34	-0.077	-0.035	0.082	0.26
Pclass	-0.035	-0.34	1	-0.37	0.083	0.018	-0.55
Age	0.037	-0.077	-0.37	1	-0.31	-0.19	0.096
SibSp	-0.058	-0.035	0.083	-0.31	1	0.41	0.16
Parch	-0.0017	0.082	0.018	-0.19	0.41	1	0.22
Fare	0.013	0.26	-0.55	0.096	0.16	0.22	1

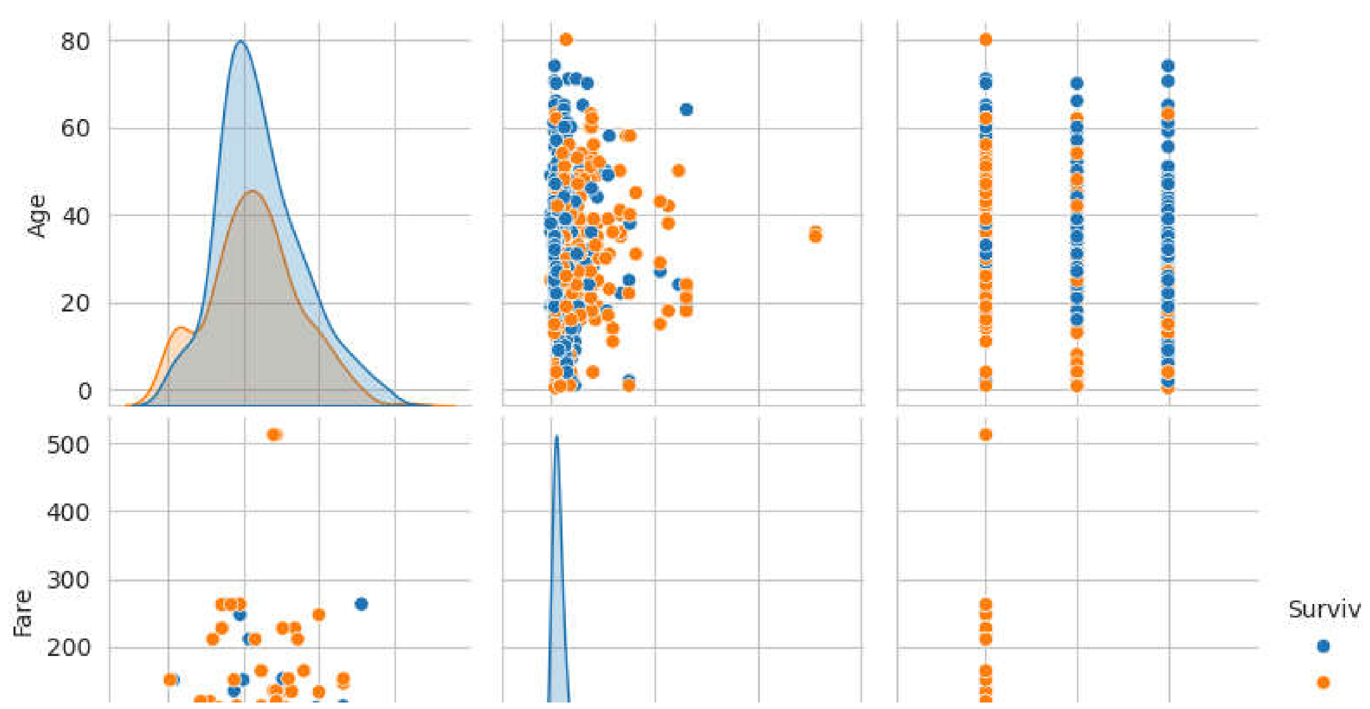
Step 14: Multivariate Analysis - Correlation Heatmap

We now analyze the correlation between numerical features.

- The heatmap shows that 'Pclass' and 'Fare' are negatively correlated (-0.55), meaning higher class passengers generally paid higher fares.
- 'Survived' has a moderate positive correlation with 'Fare' (0.26), suggesting that passengers who paid higher fares had better survival rates.
- Other features such as 'Age', 'SibSp', and 'Parch' show weak correlations with 'Survived'.
- No strong multicollinearity (very high correlation) is observed between the independent variables.

Correlation analysis helps in understanding relationships between variables and identifying important predictors for survival.

```
sns.pairplot(df[['Survived', 'Age', 'Fare', 'Pclass']], hue='Survived')
plt.show()
```


Age
80
60
40
20
0
Fare
500
400
300
200
Surviv
●
●