Elevate Labs Internship

TASK-05

Perform data cleaning and exploratory data analysis (EDA) on the Titanic dataset from Kaggle.

Overview of the datasets:

Training Data (train.csv):

Contains information on 891 passengers with the following columns:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked

Survived is the target variable indicating survival (1) or not (0).

Test Data (test.csv):

Contains information on 418 passengers, but without the Survived column. This is used for predictions.

Gender Submission (gender_submission.csv):

Provides a sample submission format with PassengerId and Survived.

Load the datasets

In [1]:
```
import pandas as pd

train_data = pd.read_csv('D:/Internship/Prasunet/Task 2/train.csv')
test_data = pd.read_csv('D:/Internship/Prasunet/Task 2/test.csv')
gender_submission = pd.read_csv('D:/Internship/Prasunet/Task 2/gender_submission.csv')
```

Display the first few rows of each dataset

In [55]:
```
train_data.head()
```

Train Data

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

In [34]:
```
test_data.head()
```

Out[34]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [33]:
```
gender_submission.head()
```

Out[33]:

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1

Summary statistics

In [35]:
```
train_data.info()
```

```
<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          891 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        891 non-null    object 
 11  Embarked     891 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [36]:
```
train_data.describe(include='all')
```

Out[36]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	E
count	891.000000	891.000000	891.000000	891	891	891.000000	891.000000	891.000000	891	891.000000	891	
unique	NaN	NaN	NaN	891	2	NaN	NaN	NaN	681	NaN	148	
top	NaN	NaN	NaN	Braund, Mr. Owen Harris	male	NaN	NaN	NaN	347082	NaN	Missing	
freq	NaN	NaN	NaN	1	577	NaN	NaN	NaN	7	NaN	687	
mean	446.000000	0.383838	2.308642	NaN	NaN	29.361582	0.523008	0.381594	NaN	32.204208	NaN	
std	257.353842	0.486592	0.836071	NaN	NaN	13.019697	1.102743	0.806057	NaN	49.693429	NaN	
min	1.000000	0.000000	1.000000	NaN	NaN	0.420000	0.000000	0.000000	NaN	0.000000	NaN	
25%	223.500000	0.000000	2.000000	NaN	NaN	22.000000	0.000000	0.000000	NaN	7.910400	NaN	
50%	446.000000	0.000000	3.000000	NaN	NaN	28.000000	0.000000	0.000000	NaN	14.454200	NaN	
75%	668.500000	1.000000	3.000000	NaN	NaN	35.000000	1.000000	0.000000	NaN	31.000000	NaN	
max	891.000000	1.000000	3.000000	NaN	NaN	80.000000	8.000000	6.000000	NaN	512.329200	NaN	

Missing values for the training data

In [37]:
```
train_data.isnull().sum()
```

Out[37]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
```

Data Cleaning

In [4]:
```python
# Impute missing values for 'Age' with median
train_data['Age'].fillna(train_data['Age'].median(), inplace=True)

# Fill missing 'Cabin' values with 'Missing'
train_data['Cabin'].fillna('Missing', inplace=True)

# Impute missing 'Embarked' values with the most frequent value ('S')
train_data['Embarked'].fillna(train_data['Embarked'].mode()[0], inplace=True)

# Verify if missing values are handled
train_data_missing_after = train_data.isnull().sum()
train_data_missing_after
```

Out[4]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
Embarked       0
dtype: int64
```

All missing values have been handled in the training dataset.

• Age: Imputed with the median.

• Cabin: Missing values replaced with Missing.

• Embarked: Imputed with the most frequent value (S).

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) focusing on univariate analysis, bivariate analysis, and multivariate analysis, along with visualizations. This process helps in understanding the data and discovering patterns, relationships, and anomalies.

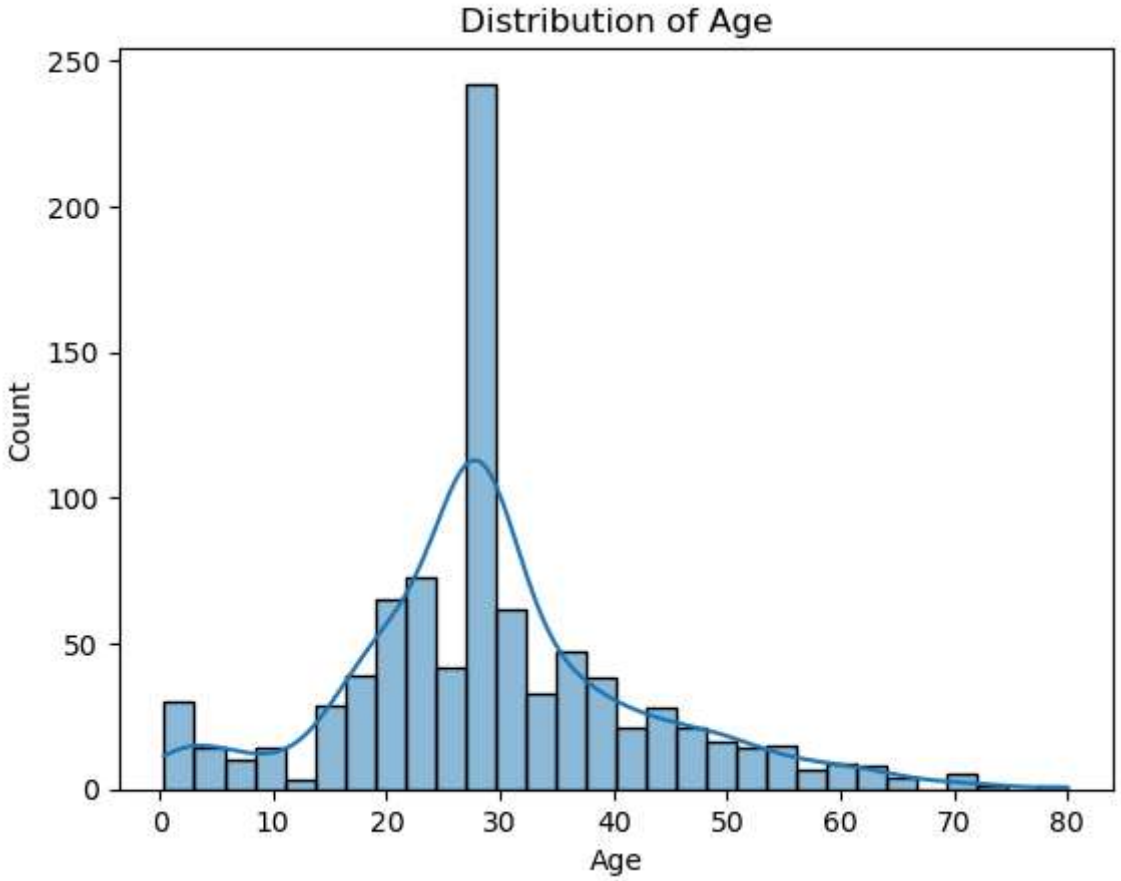
1. Univariate Analysis

Univariate analysis focuses on a single variable to understand its distribution, central tendency, and spread. It's the simplest form of analysis since it deals with one variable at a time.

In [5]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

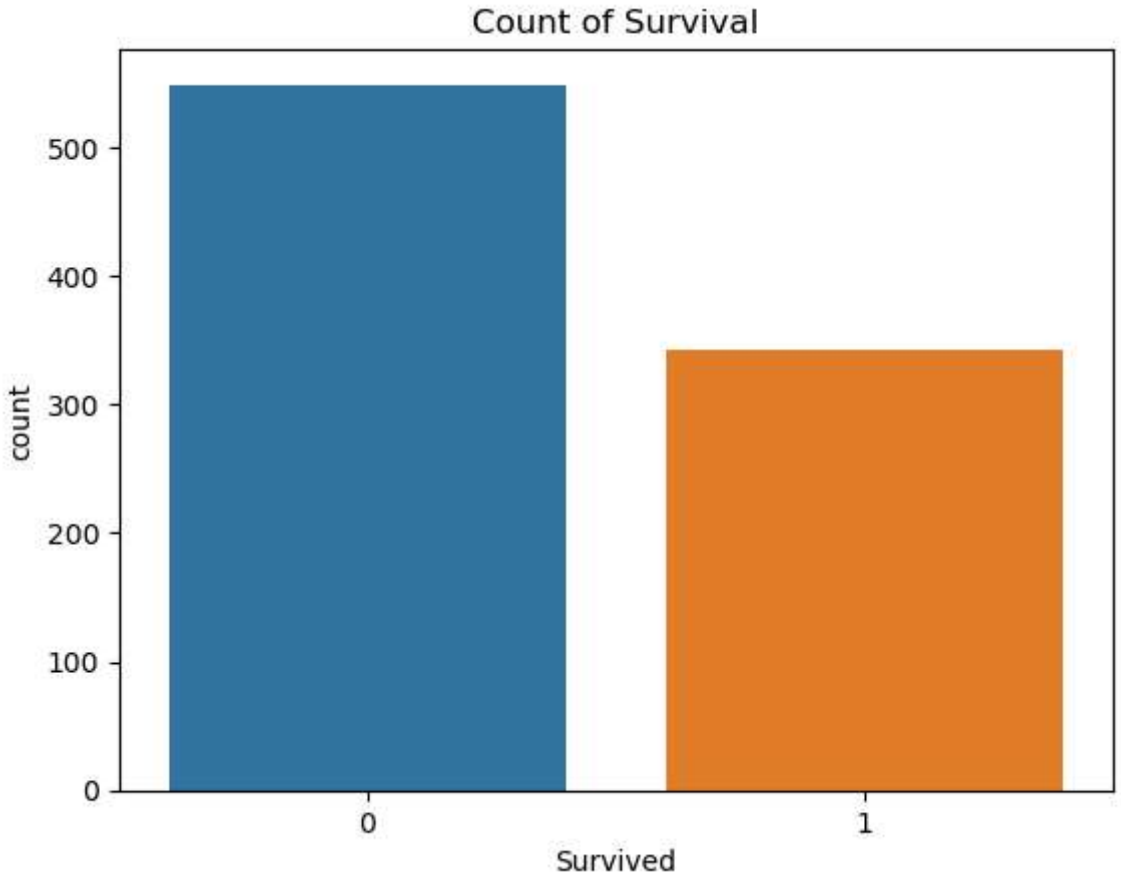
Age Distribution:

In [48]:
```
sns.histplot(train_data['Age'], kde=True)
plt.title('Distribution of Age')
plt.show()
```



Survival Counts:

In [49]:
```
sns.countplot(x='Survived', data=train_data)
plt.title('Count of Survival')
plt.show()
```

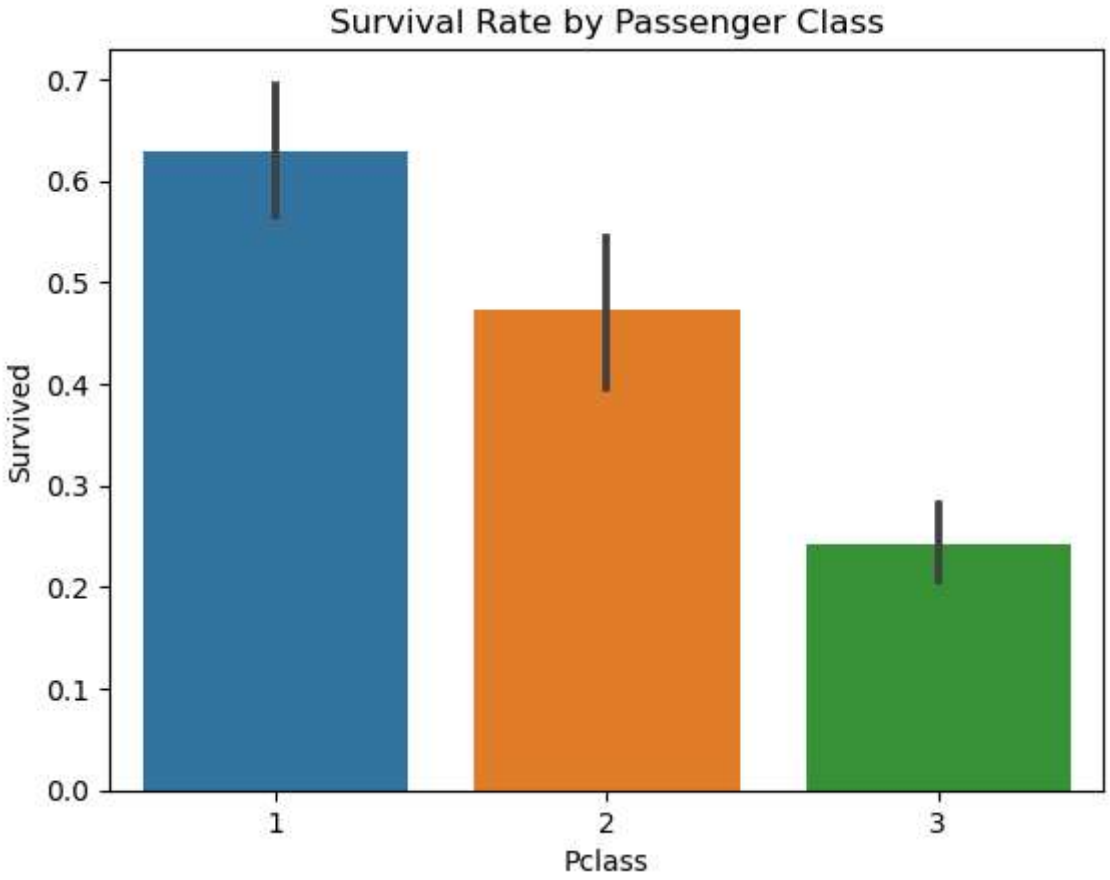


2. Bivariate Analysis

Bivariate analysis examines the relationship between two variables. This helps in understanding how one variable changes with respect to another.

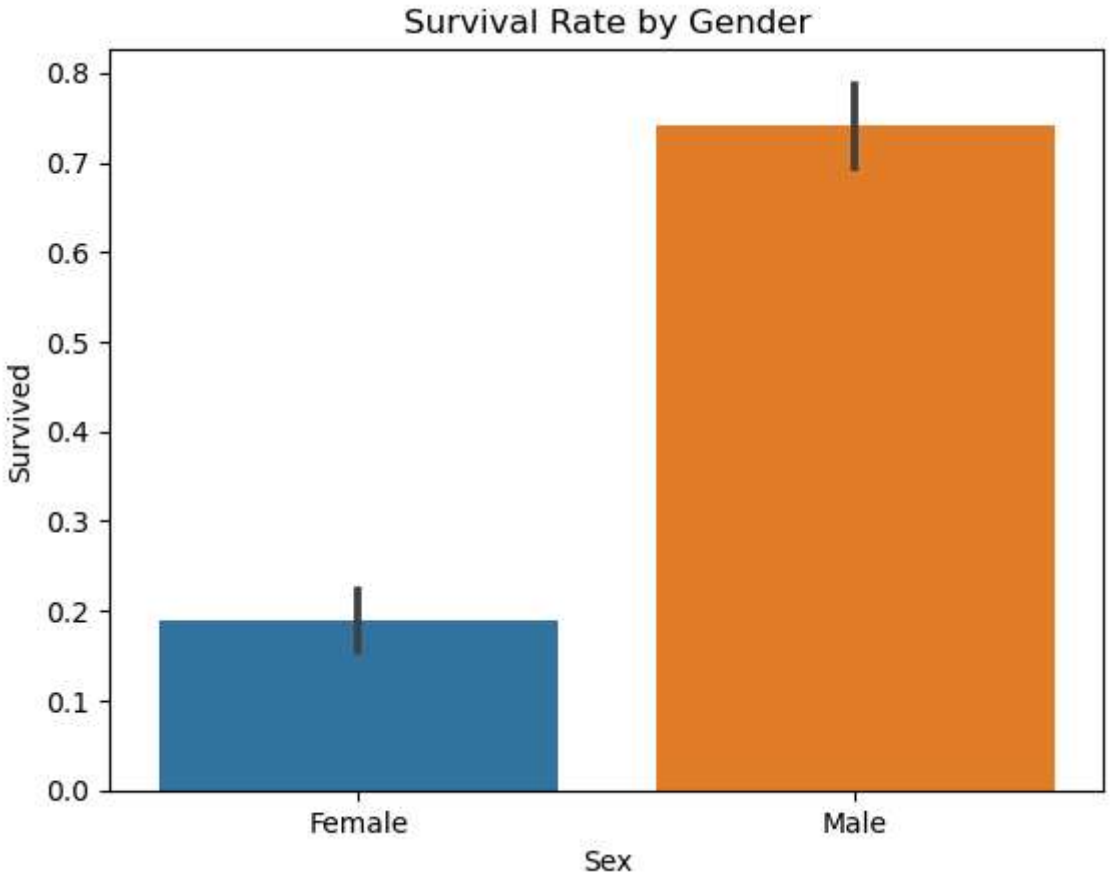
Survival by Passenger Class:

In [50]:
```
sns.barplot(x='Pclass', y='Survived', data=data)
plt.title('Survival Rate by Passenger Class')
plt.show()
```



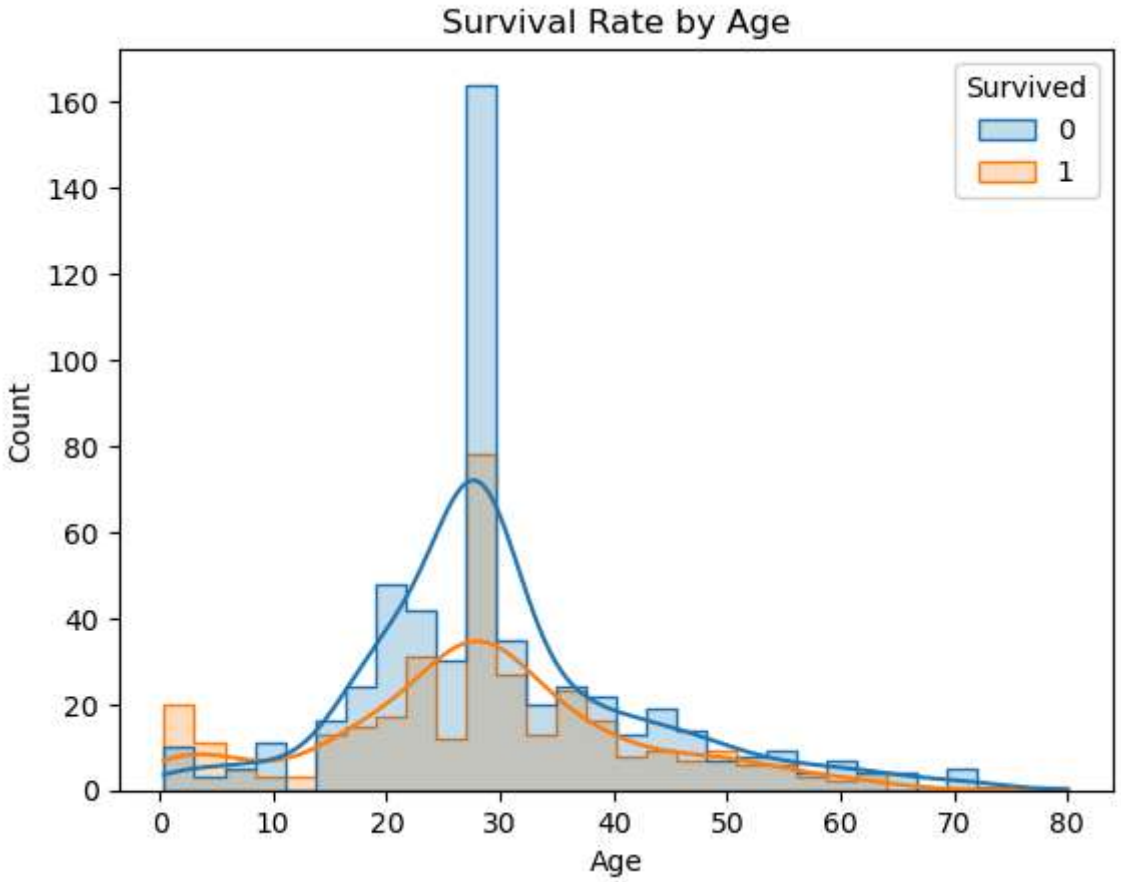
Survival by Gender

In [30]:
```
sns.barplot(x='Sex', y='Survived', data=train_data)
plt.title('Survival Rate by Gender')
plt.xticks([0, 1], ['Female', 'Male'])
plt.show()
```



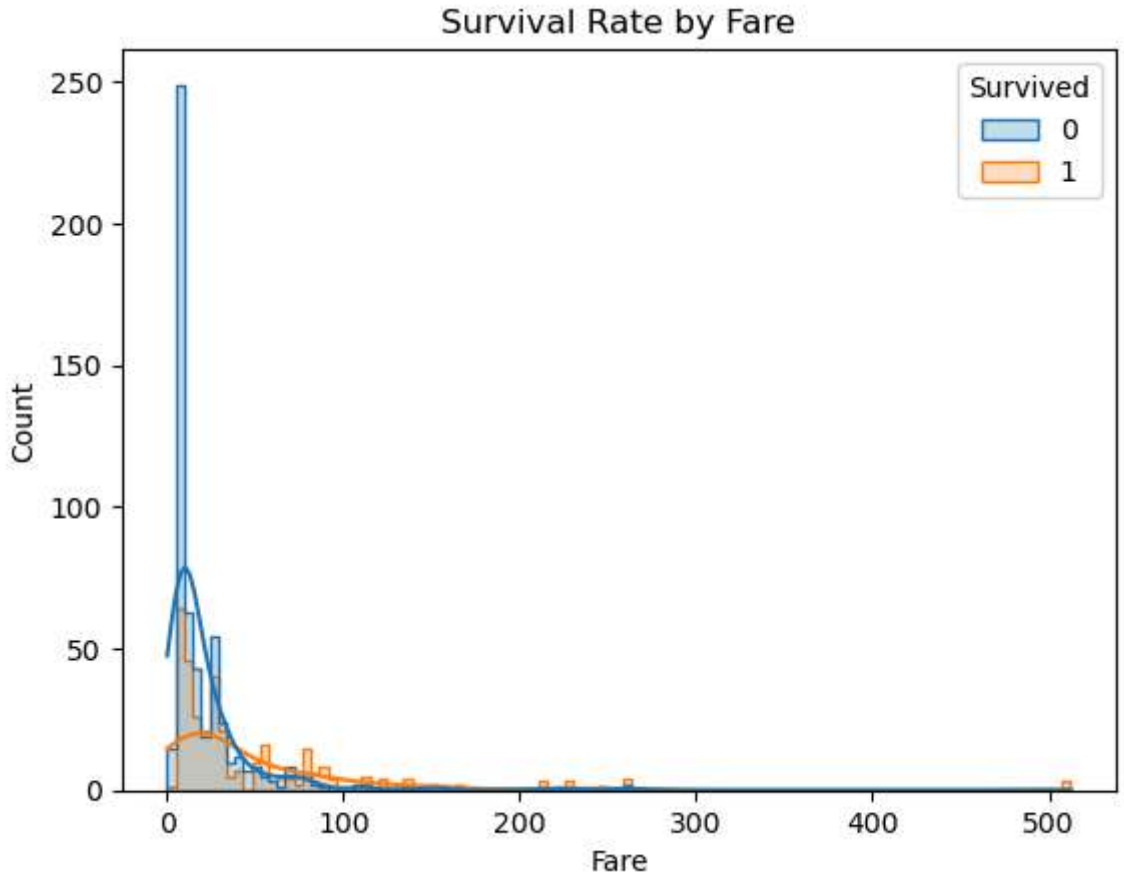
Survival by Age

In [24]:
```
sns.histplot(data=train_data, x='Age', hue='Survived', kde=True, element='step')
plt.title('Survival Rate by Age')
plt.show()
```



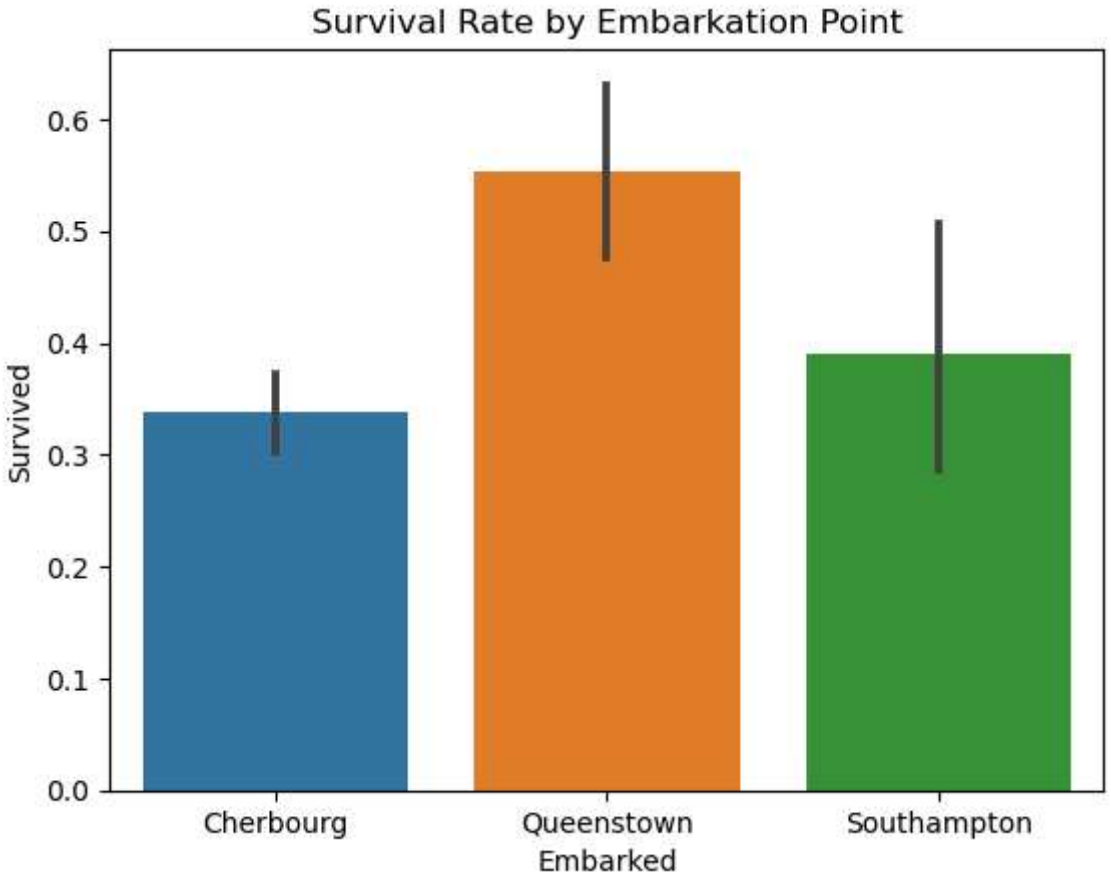
Survival by Fare

In [25]:
```
sns.histplot(data=train_data, x='Fare', hue='Survived', kde=True, element='step')
plt.title('Survival Rate by Fare')
plt.show()
```



Survival by Embarkation Point

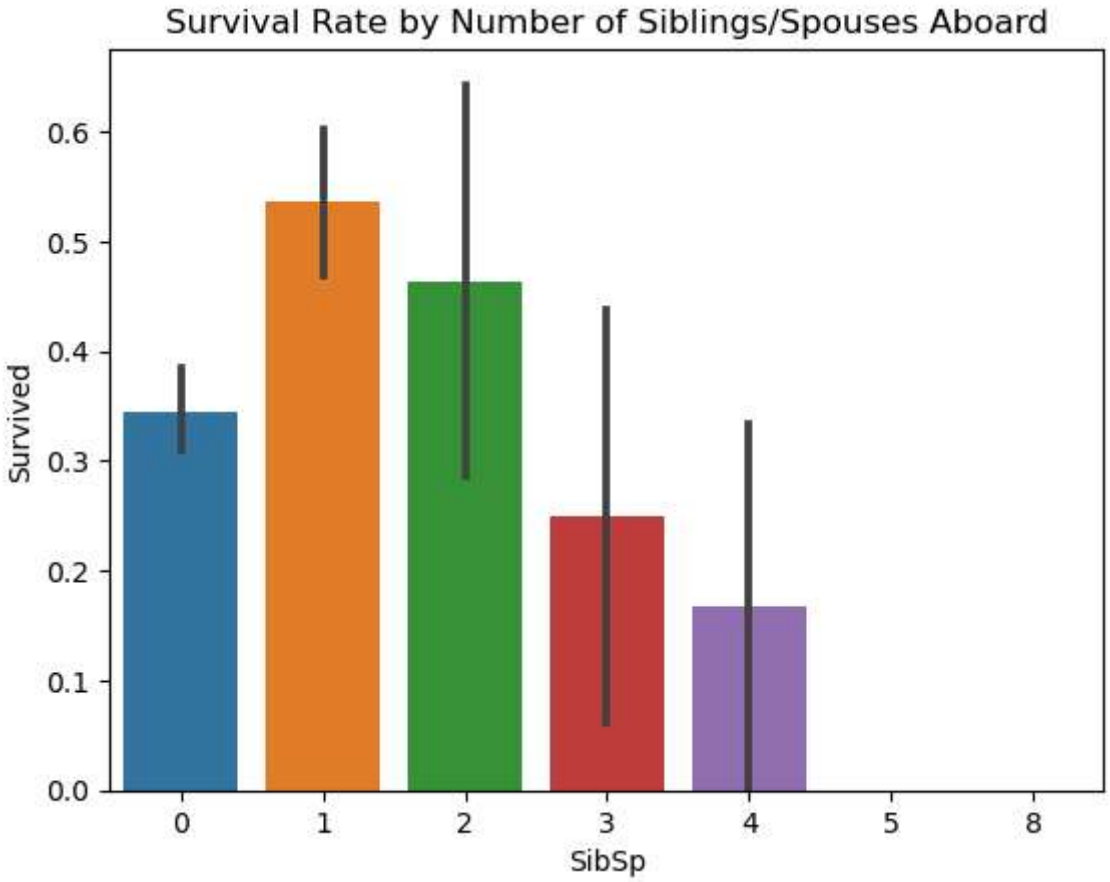
In [26]:
```
sns.barplot(x='Embarked', y='Survived', data=train_data)
plt.title('Survival Rate by Embarkation Point')
plt.xticks([0, 1, 2], ['Cherbourg', 'Queenstown', 'Southampton'])
plt.show()
```

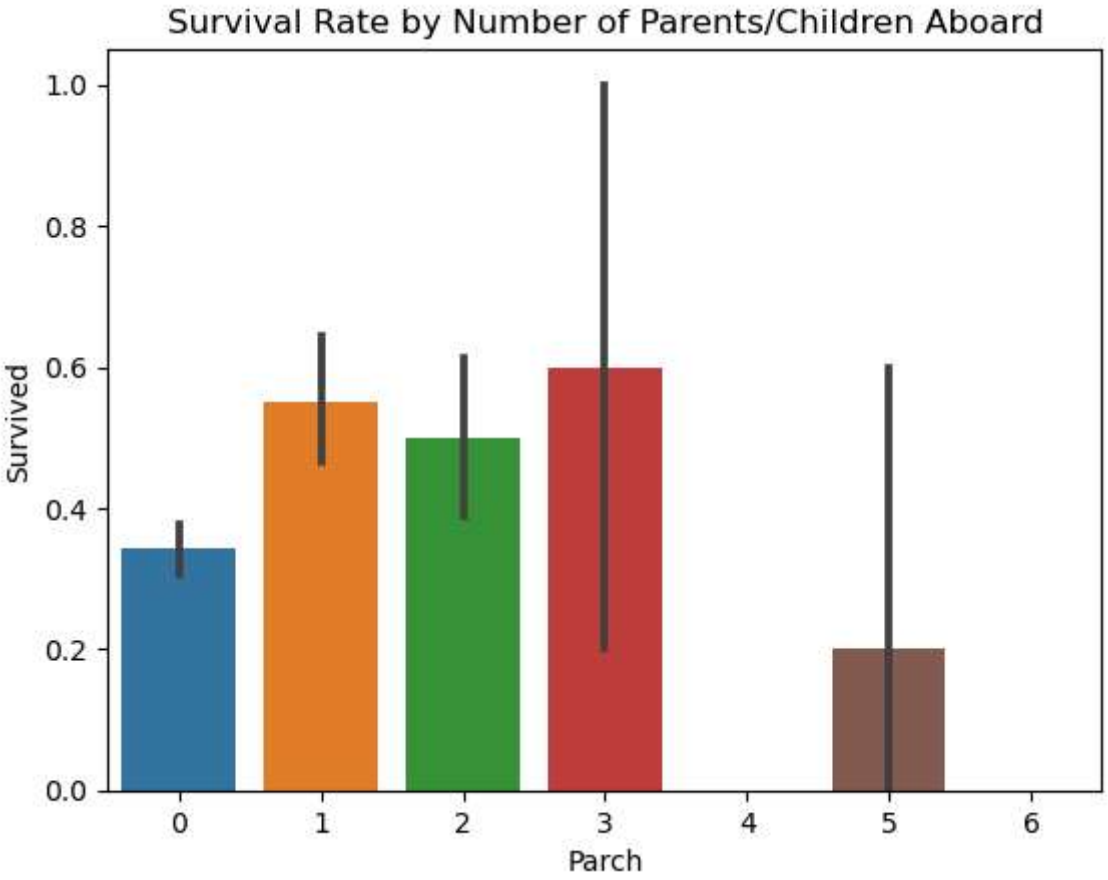


Survival by Family Size

In [27]:
```
sns.barplot(x='SibSp', y='Survived', data=train_data)
plt.title('Survival Rate by Number of Siblings/Spouses Aboard')
plt.show()

sns.barplot(x='Parch', y='Survived', data=train_data)
plt.title('Survival Rate by Number of Parents/Children Aboard')
plt.show()
```


Survival Rate by Number of Siblings/Spouses Aboard
Survived
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0
1
2
3
4
5
8
SibSp



3. Multivariate Analysis

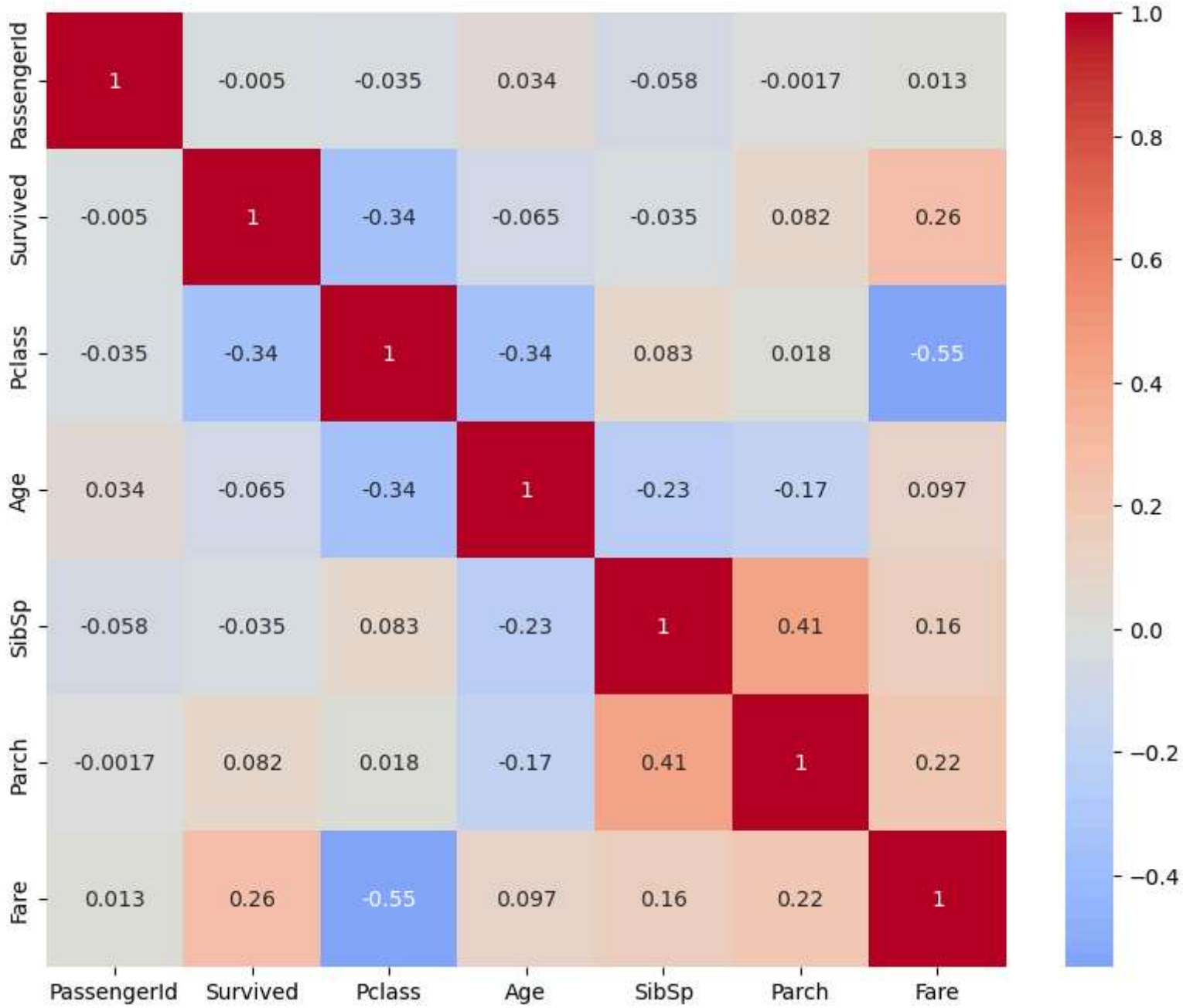
Multivariate analysis involves examining more than two variables simultaneously. This can reveal complex relationships and interactions that are not evident in bivariate analysis.

Correlation Heatmap:

In [47]:
```
plt.figure(figsize=(10, 8))
corr_matrix = train_data.corr()
```

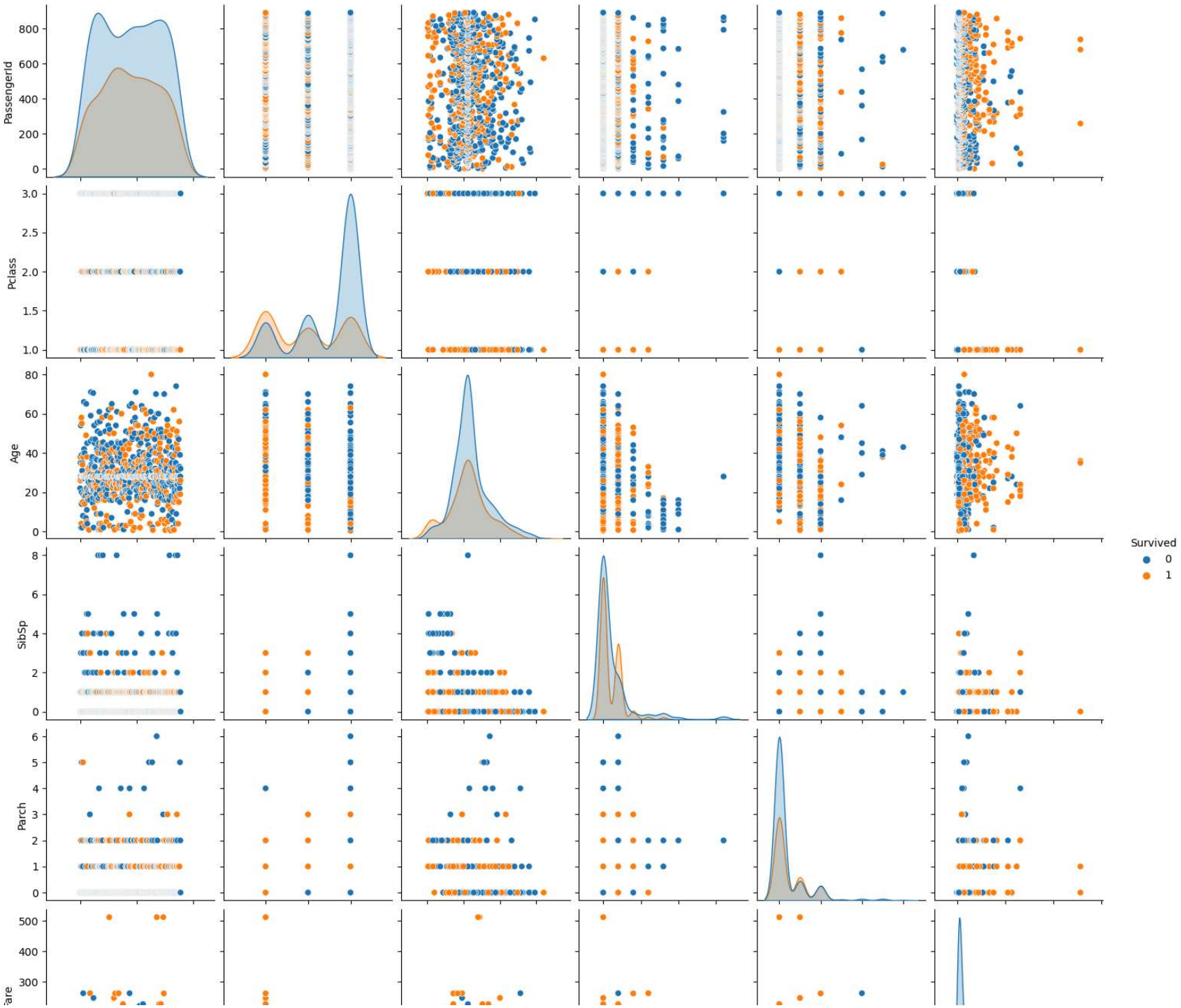
```
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', center=0)
plt.show()
```

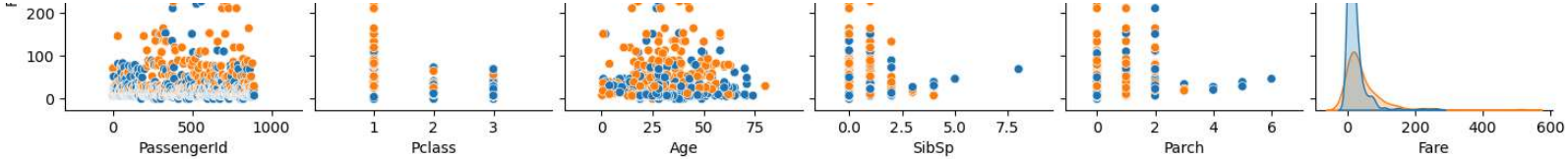
```
C:\Users\siyam\AppData\Local\Temp\ipykernel_8776\3941700078.py:2: FutureWarning: The default value of numeric_only in
DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the
value of numeric_only to silence this warning.
  corr_matrix = train_data.corr()
```


PassengerId
Survived
Pclass
Age
SibSp
Parch
Fare
1 -0.005 -0.035 0.034 -0.058 -0.0017 0.013
-0.005 1 -0.34 -0.065 -0.035 0.082 0.26
-0.035 -0.34 1 -0.34 0.083 0.018 -0.55
0.034 -0.065 -0.34 1 -0.23 -0.17 0.097
-0.058 -0.035 0.083 -0.23 1 0.41 0.16
-0.0017 0.082 0.018 -0.17 0.41 1 0.22
0.013 0.26 -0.55 0.097 0.16 0.22 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

Pair Plot:

In [17]:
```
sns.pairplot(train_data, hue='Survived')
plt.show()
```


PassengerId
Pclass
Age
SibSp
Parch
Fare
Survived
0
1



Insights:

• Passengers in higher classes (Pclass=1) had higher survival rates.

• Females had a significantly higher survival rate compared to males.

• Children and elderly passengers had higher survival rates.

• Passengers who paid higher fares tended to have higher survival rates.

• Embarkation point 'C' (Cherbourg) showed a higher survival rate.

• Small to medium-sized families (SibSp and Parch) had better survival chances.

• The correlation heatmap and pair plot suggest some relationships between variables, such as positive correlations between fare and survival, and potentially negative correlations between survival and certain demographic factors like age or class.

In [ ]: