```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import plotly.figure_factory as ff
df = pd.read_csv('/bin/Titanic-Dataset.csv')
print("🔹 Summary Statistics:")
print(df.describe(include='all'))
```

```
🔹 Summary Statistics:
        PassengerId    Survived      Pclass                Name   Sex  \
count    891.000000  891.000000  891.000000                 891   891
unique          NaN         NaN         NaN                 891     2
top             NaN         NaN         NaN  Dooley, Mr. Patrick  male
freq            NaN         NaN         NaN                   1   577
mean     446.000000    0.383838    2.308642                 NaN   NaN
std      257.353842    0.486592    0.836071                 NaN   NaN
min        1.000000    0.000000    1.000000                 NaN   NaN
25%      223.500000    0.000000    2.000000                 NaN   NaN
50%      446.000000    0.000000    3.000000                 NaN   NaN
75%      668.500000    1.000000    3.000000                 NaN   NaN
max      891.000000    1.000000    3.000000                 NaN   NaN

                Age       SibSp       Parch  Ticket        Fare Cabin Embarked
count   714.000000  891.000000  891.000000     891  891.000000   204      889
unique         NaN         NaN         NaN     681         NaN   147        3
top            NaN         NaN         NaN  347082         NaN    G6        S
freq           NaN         NaN         NaN       7         NaN     4      644
mean     29.699118    0.523008    0.381594     NaN   32.204208   NaN      NaN
std      14.526497    1.102743    0.806057     NaN   49.693429   NaN      NaN
min       0.420000    0.000000    0.000000     NaN    0.000000   NaN      NaN
25%      20.125000    0.000000    0.000000     NaN    7.910400   NaN      NaN
50%      28.000000    0.000000    0.000000     NaN   14.454200   NaN      NaN
75%      38.000000    1.000000    0.000000     NaN   31.000000   NaN      NaN
max      80.000000    8.000000    6.000000     NaN  512.329200   NaN      NaN
```

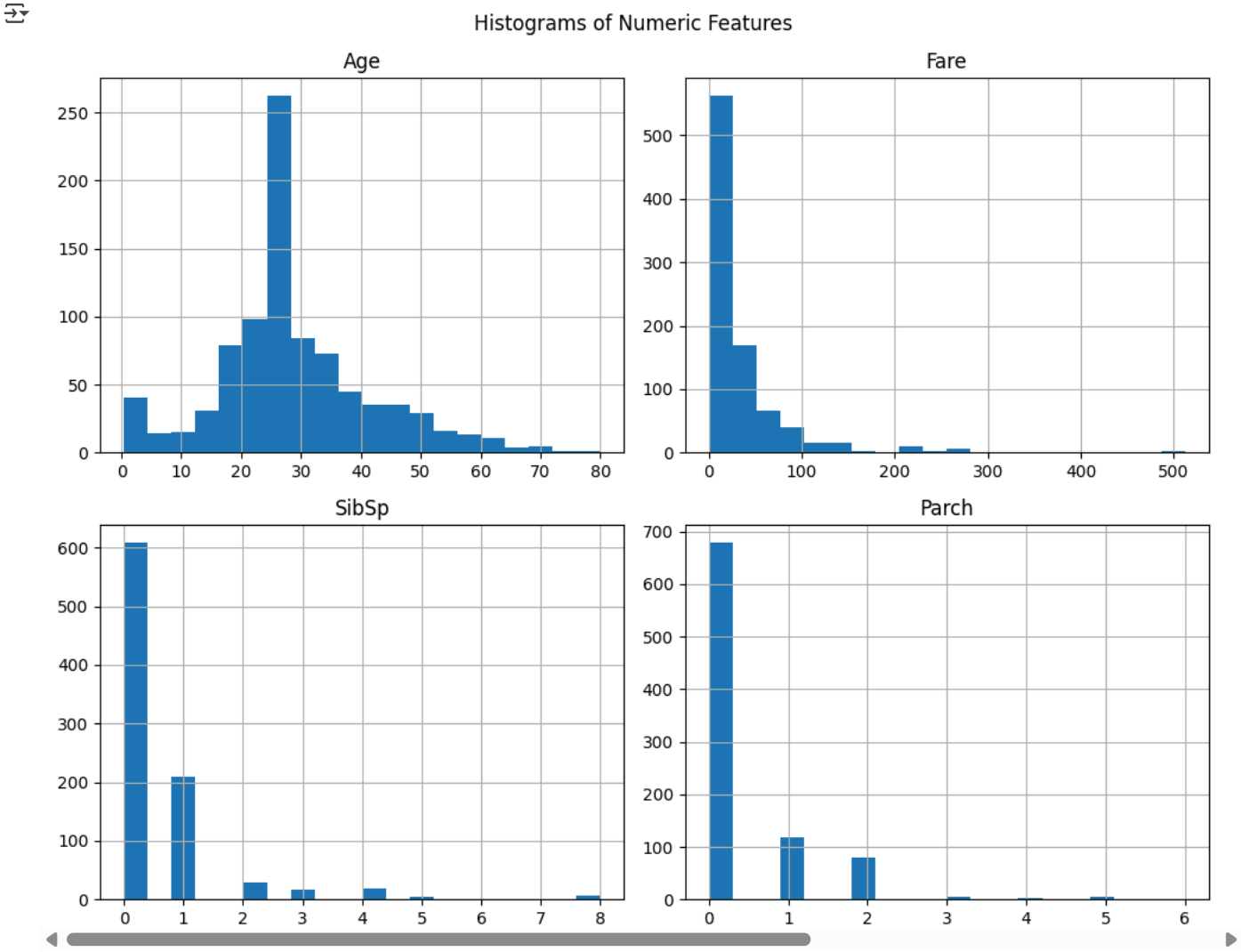
```
print("\n🔹 Missing Values:")
print(df.isnull().sum())
```

```
🔹 Missing Values:
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

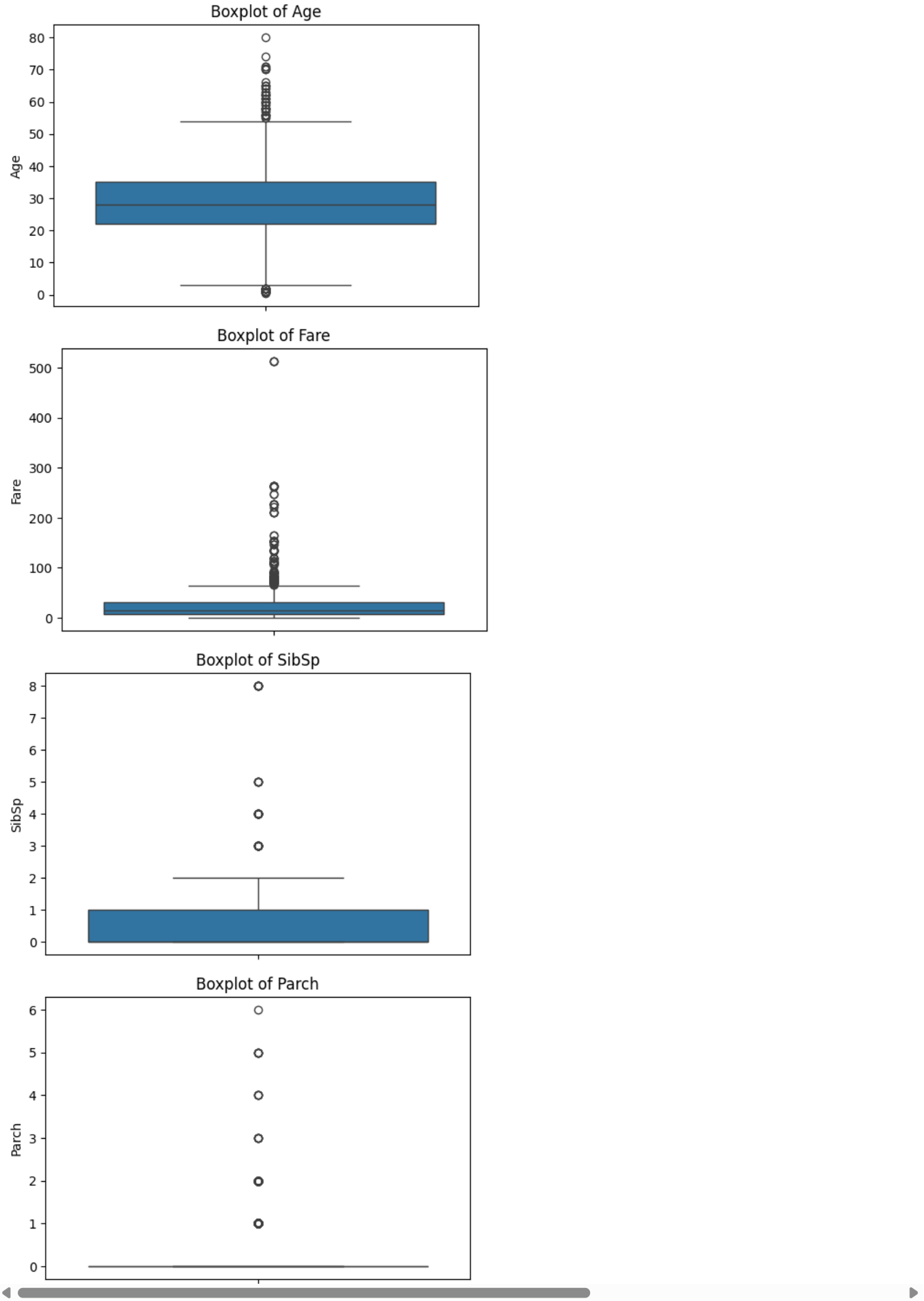
```
df['Age'].fillna(df['Age'].median(), inplace=True)
df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)
df['Fare'].fillna(df['Fare'].median(), inplace=True)
```

```
<ipython-input-5-d55561829552>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  df['Age'].fillna(df['Age'].median(), inplace=True)
<ipython-input-5-d55561829552>:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)
<ipython-input-5-d55561829552>:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assi
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col

  df['Fare'].fillna(df['Fare'].median(), inplace=True)
```

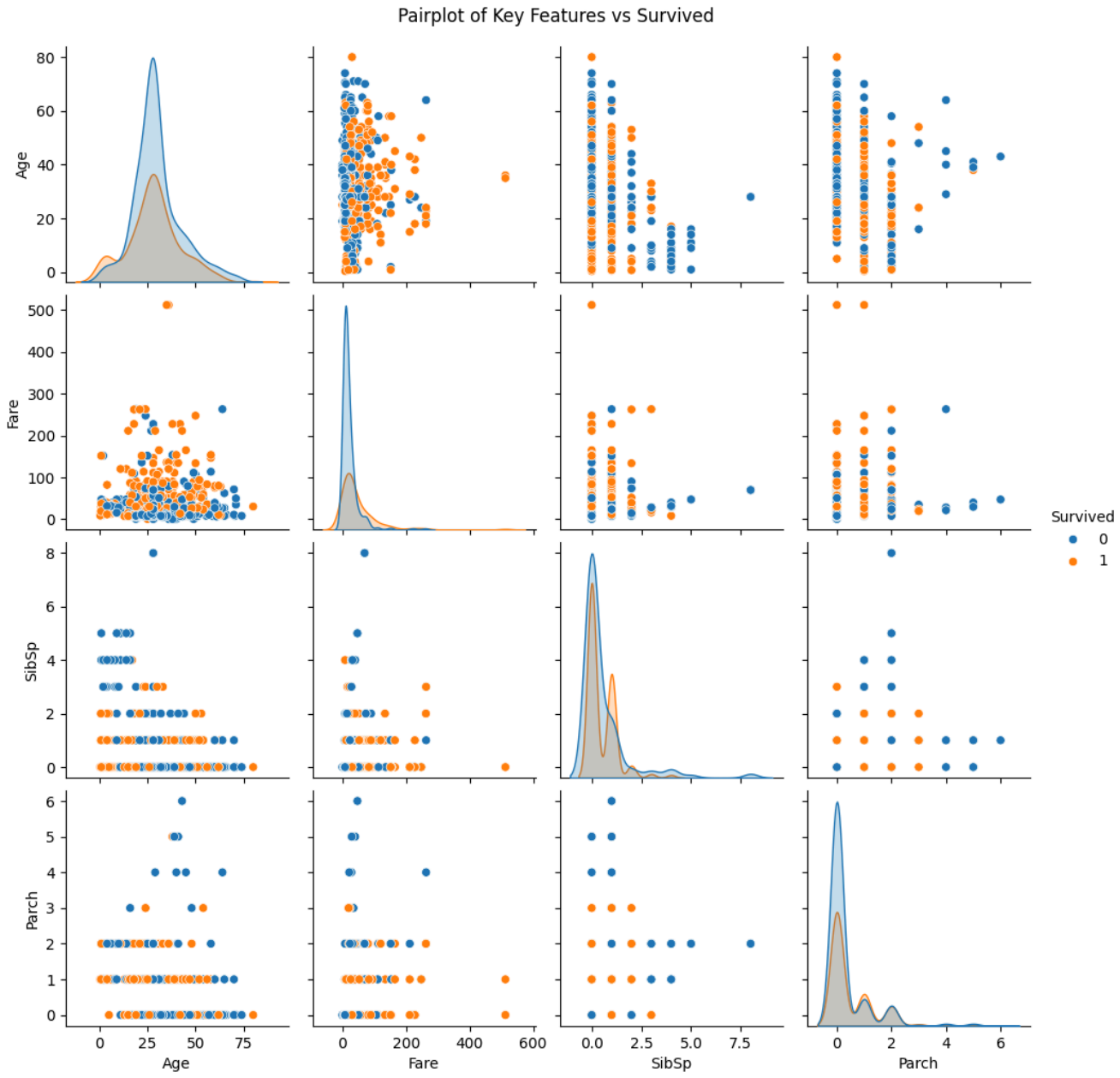
```
numeric_cols = ['Age', 'Fare', 'SibSp', 'Parch']
df[numeric_cols].hist(bins=20, figsize=(10, 8), layout=(2, 2))
plt.suptitle("Histograms of Numeric Features")
plt.tight_layout()
plt.show()
```



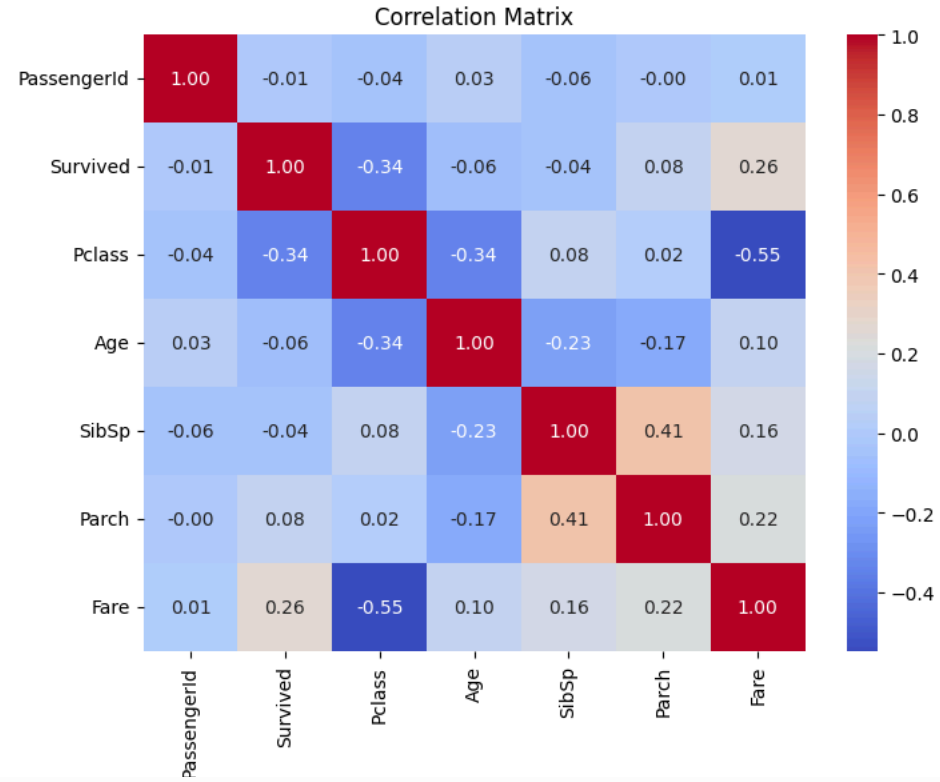

```
for col in numeric_cols:
    plt.figure(figsize=(6, 4))
    sns.boxplot(data=df, y=col)
    plt.title(f"Boxplot of {col}")
    plt.show()
```




```
sns.pairplot(df[['Survived', 'Age', 'Fare', 'SibSp', 'Parch']], hue='Survived')
plt.suptitle("Pairplot of Key Features vs Survived", y=1.02)
plt.show()
```



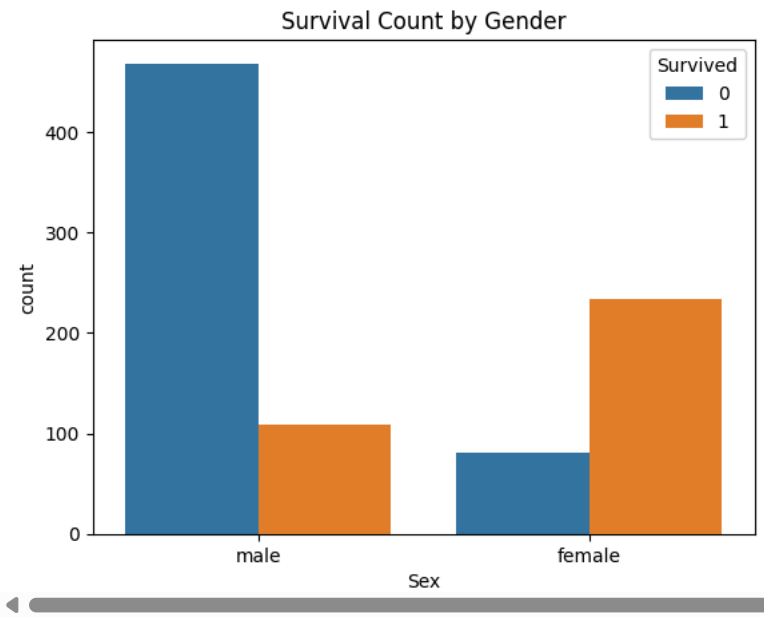

```
plt.figure(figsize=(8, 6))
numeric_df = df.select_dtypes(include=[np.number])
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Matrix")
plt.show()
```



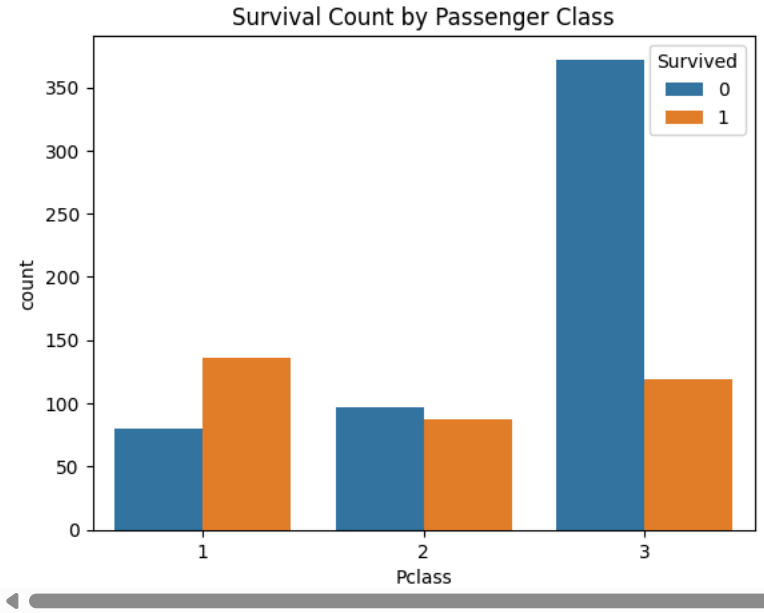

```
print("\n 🔹 Skewness of Numeric Features:")
print(df[numeric_cols].skew())
```

```
 🔹 Skewness of Numeric Features:
Age      0.510245
Fare     4.787317
SibSp    3.695352
Parch    2.749117
dtype: float64
```

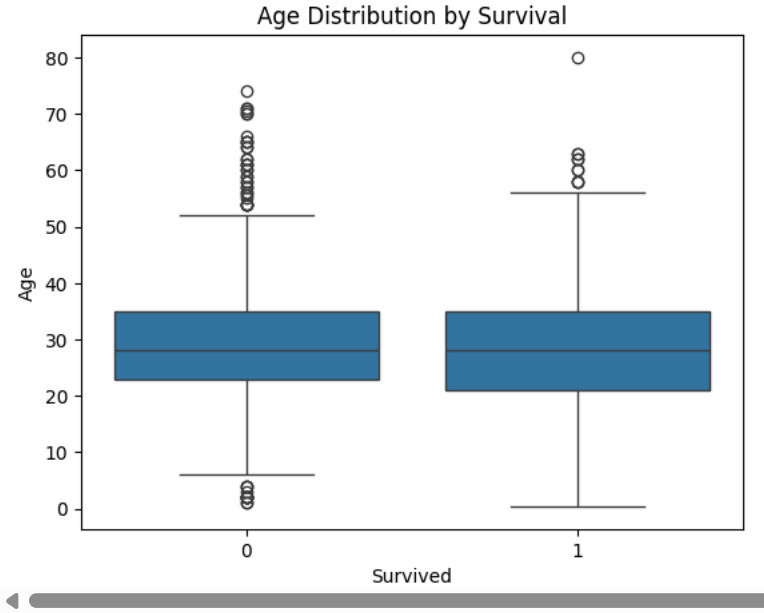
```
sns.countplot(data=df, x='Sex', hue='Survived')
plt.title("Survival Count by Gender")
plt.show()
```




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




```
sns.boxplot(x='Survived', y='Age', data=df)
plt.title("Age Distribution by Survival")
plt.show()
```




```
print("\n 🔹 Observations & Inferences:")
print("""
1. Females had a much higher survival rate than males.
2. Passengers in Pclass 1 had a higher survival probability.
3. Boxplots show outliers in Age and Fare.
4. Correlation between Fare and Survived is mildly positive.
5. Pairplot reveals non-linear and weak relationships between features.
6. Age and Fare distributions are right-skewed.
""")
```

```
 🔹 Observations & Inferences:

1. Females had a much higher survival rate than males.
2. Passengers in Pclass 1 had a higher survival probability.
3. Boxplots show outliers in Age and Fare.
4. Correlation between Fare and Survived is mildly positive.
5. Pairplot reveals non-linear and weak relationships between features.
6. Age and Fare distributions are right-skewed.
```

```
corr_matrix = numeric_df.corr().round(2)
fig = ff.create_annotated_heatmap(
    z=corr_matrix.values,
    x=list(corr_matrix.columns),
    y=list(corr_matrix.index),
    annotation_text=corr_matrix.values,
```

```
    colorscale='Viridis'
)
fig.update_layout(title_text='Interactive Correlation Matrix')
fig.show()
```


```
df['LogFare'] = np.log1p(df['Fare'])
sns.histplot(df['LogFare'], kde=True)
```