Task-5 Titanic Dataset - Exploratory Data Analysis (EDA)

📘 Project Overview

This project explores the Titanic dataset, a classic dataset used to teach data science and machine learning. The goal is to perform an in-depth exploratory data analysis (EDA) to understand the key factors that influenced survival during the Titanic disaster.

Using Python and its data analysis libraries, this notebook walks through the process of data cleaning, visualization, and interpretation. The insights gathered here can serve as a foundation for building predictive models.

🧰 Tools and Libraries Used

- Python 3.x
- Pandas
- NumPy
- Matplotlib
- Seaborn
- Jupyter Notebook

📊 EDA Objectives

- Inspect the structure of the dataset
- Identify and handle missing data
- Explore feature distributions
- Analyze correlations and survival impact
- Visualize relationships between variables
- Derive key insights and storytelling from data

📁 Repository Structure

titanic-eda-afnan/ ├── titanic_eda.ipynb # Main notebook with EDA ├── titanic.csv # Dataset file └── README.md # Project overview and details

🔍 Key Findings

- **Gender:** Females had a significantly higher survival rate than males.
- **Class:** Passengers from higher classes (especially 1st class) were more likely to survive.
- **Age:** Children (especially under 10) had better survival odds.
- **Fare & Embarkation:** Higher fares and boarding from Cherbourg (C) were associated with better survival chances.

📈 Sample Visualizations

- Survival distribution by sex and class (stacked bar charts)
- Age distribution across survivors and non-survivors (histograms)
- Heatmaps to show correlations among numeric features
- Box plots to show fare variation by class and survival status

📌 Conclusion

The Titanic dataset provides a great starting point to practice data analysis, cleaning, and visualization. Through EDA, we uncover hidden patterns and relationships that are crucial for building machine learning models or for data storytelling.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
df = pd.read_csv('tested.csv')

print("--- Initial Data Exploration ---")
print("\nFirst 5 rows of the dataset:")
print(df.head())

print("\n-----------------------------------")
print("Dataset Information:")
df.info()

print("\n-----------------------------------")
print("Missing Values per Column:")
print(df.isnull().sum())

print("\n-----------------------------------")
print("Summary Statistics for Numerical Columns:")
print(df.describe())
```

```
--- Initial Data Exploration ---

First 5 rows of the dataset:
   PassengerId  Survived  Pclass  \
0          892         0       3
1          893         1       3
2          894         0       2
3          895         0       3
4          896         1       3

                                          Name     Sex   Age  SibSp  Parch  \
0                              Kelly, Mr. James    male  34.5      0      0
1              Wilkes, Mrs. James (Ellen Needs)  female  47.0      1      0
2                     Myles, Mr. Thomas Francis    male  62.0      0      0
3                              Wirz, Mr. Albert    male  27.0      0      0
4  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  female  22.0      1      1

    Ticket     Fare Cabin Embarked
0   330911   7.8292   NaN        Q
1   363272   7.0000   NaN        S
2   240276   9.6875   NaN        Q
3   315154   8.6625   NaN        S
4  3101298  12.2875   NaN        S

---------------------------------
Dataset Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Survived     418 non-null    int64
 2   Pclass       418 non-null    int64
 3   Name         418 non-null    object
 4   Sex          418 non-null    object
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64
 7   Parch        418 non-null    int64
 8   Ticket       418 non-null    object
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object
 11  Embarked     418 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB

---------------------------------
Missing Values per Column:
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
```

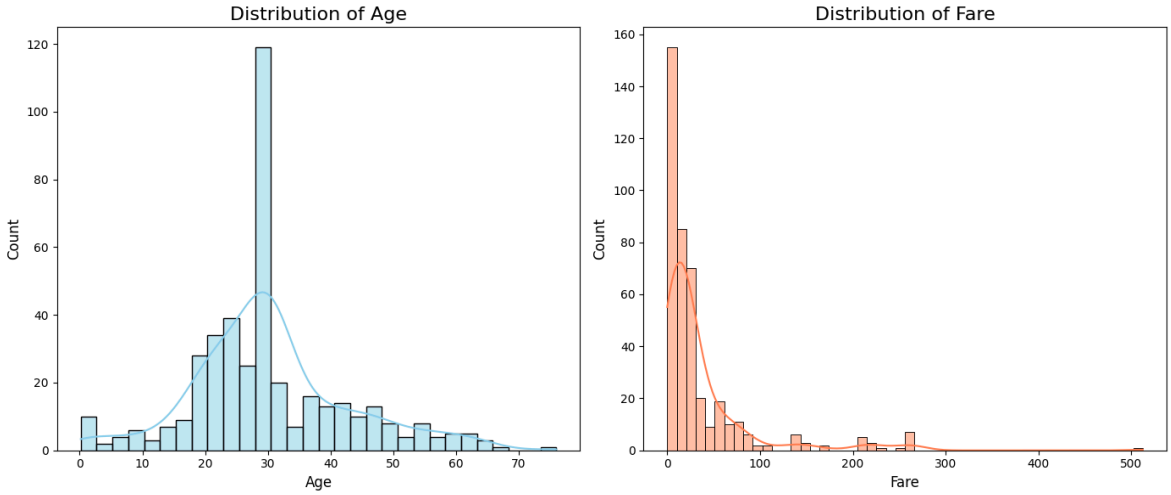
```
dtype: int64

---------------------------------
Summary Statistics for Numerical Columns:
       PassengerId    Survived      Pclass         Age       SibSp  \
count   418.000000  418.000000  418.000000  332.000000  418.000000
mean   1100.500000    0.363636    2.265550   30.272590    0.447368
std     120.810458    0.481622    0.841838   14.181209    0.896760
min     892.000000    0.000000    1.000000    0.170000    0.000000
25%     996.250000    0.000000    1.000000   21.000000    0.000000
50%    1100.500000    0.000000    3.000000   27.000000    0.000000
75%    1204.750000    1.000000    3.000000   39.000000    1.000000
max    1309.000000    1.000000    3.000000   76.000000    8.000000

            Parch        Fare
count  418.000000  417.000000
mean     0.392344   35.627188
std      0.981429   55.907576
min      0.000000    0.000000
25%      0.000000    7.895800
50%      0.000000   14.454200
75%      0.000000   31.500000
max      9.000000  512.329200
```

In [4]:

```python
# Fill missing 'Age' values with the mean of the column
df['Age'].fillna(df['Age'].mean(), inplace=True)

# Fill the single missing 'Fare' value with the median of the column
df['Fare'].fillna(df['Fare'].median(), inplace=True)

# For correlation analysis, we need to convert the 'Sex' column into a numerical
# 'male' will be 0 and 'female' will be 1.
df['Sex_numeric'] = df['Sex'].map({'male': 0, 'female': 1})
print("\n--- Univariate Analysis ---")
print("\nValue counts for key categorical columns:")
for column in ['Survived', 'Pclass', 'Sex', 'SibSp', 'Parch', 'Embarked']:
    print(f"\n--- {column} ---")
    print(df[column].value_counts())

# Plot histograms for 'Age' and 'Fare' to visualize their distributions.
plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
sns.histplot(df['Age'], kde=True, bins=30, color='skyblue')
plt.title('Distribution of Age', fontsize=16)
plt.xlabel('Age', fontsize=12)
plt.ylabel('Count', fontsize=12)

plt.subplot(1, 2, 2)
sns.histplot(df['Fare'], kde=True, bins=50, color='coral')
plt.title('Distribution of Fare', fontsize=16)
plt.xlabel('Fare', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.tight_layout()
plt.show()
```

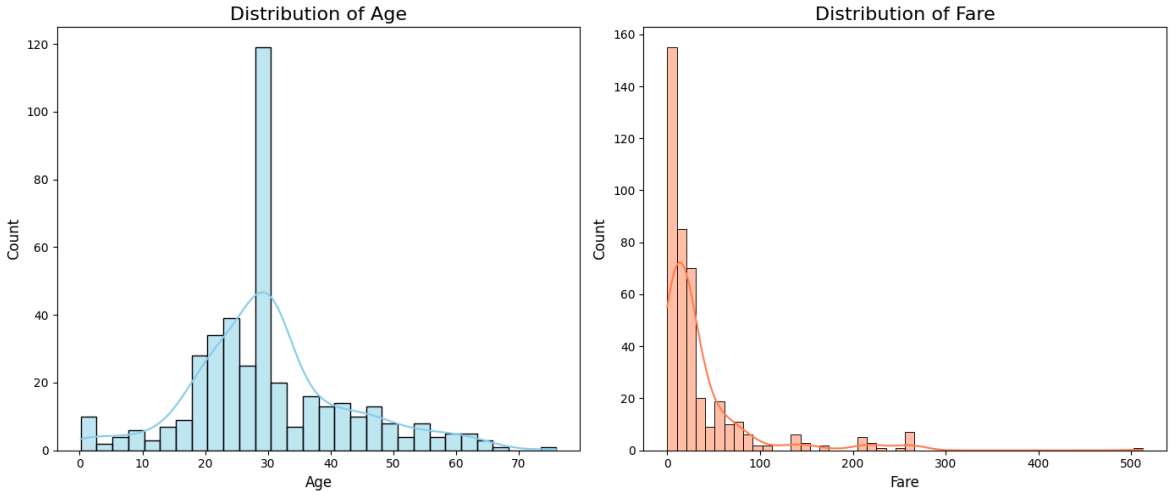
```
C:\Users\kkjeg\AppData\Local\Temp\ipykernel_27196\980813540.py:2: FutureWarning:
A value is trying to be set on a copy of a DataFrame or Series through chained as
signment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work becau
se the intermediate object on which we are setting values always behaves as a cop
y.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.meth
od({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to pe
rform the operation inplace on the original object.


  df['Age'].fillna(df['Age'].mean(), inplace=True)
C:\Users\kkjeg\AppData\Local\Temp\ipykernel_27196\980813540.py:5: FutureWarning:
A value is trying to be set on a copy of a DataFrame or Series through chained as
signment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work becau
se the intermediate object on which we are setting values always behaves as a cop
y.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.meth
od({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to pe
rform the operation inplace on the original object.


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

```
--- Univariate Analysis ---

Value counts for key categorical columns:

--- Survived ---
Survived
0    266
1    152
Name: count, dtype: int64

--- Pclass ---
Pclass
3    218
1    107
2     93
Name: count, dtype: int64

--- Sex ---
Sex
male      266
female    152
Name: count, dtype: int64

--- SibSp ---
SibSp
0    283
1    110
2     14
3      4
4      4
8      2
5      1
Name: count, dtype: int64

--- Parch ---
Parch
0    324
1     52
2     33
3      3
4      2
9      2
6      1
5      1
Name: count, dtype: int64

--- Embarked ---
Embarked
S    270
C    102
Q     46
Name: count, dtype: int64
```



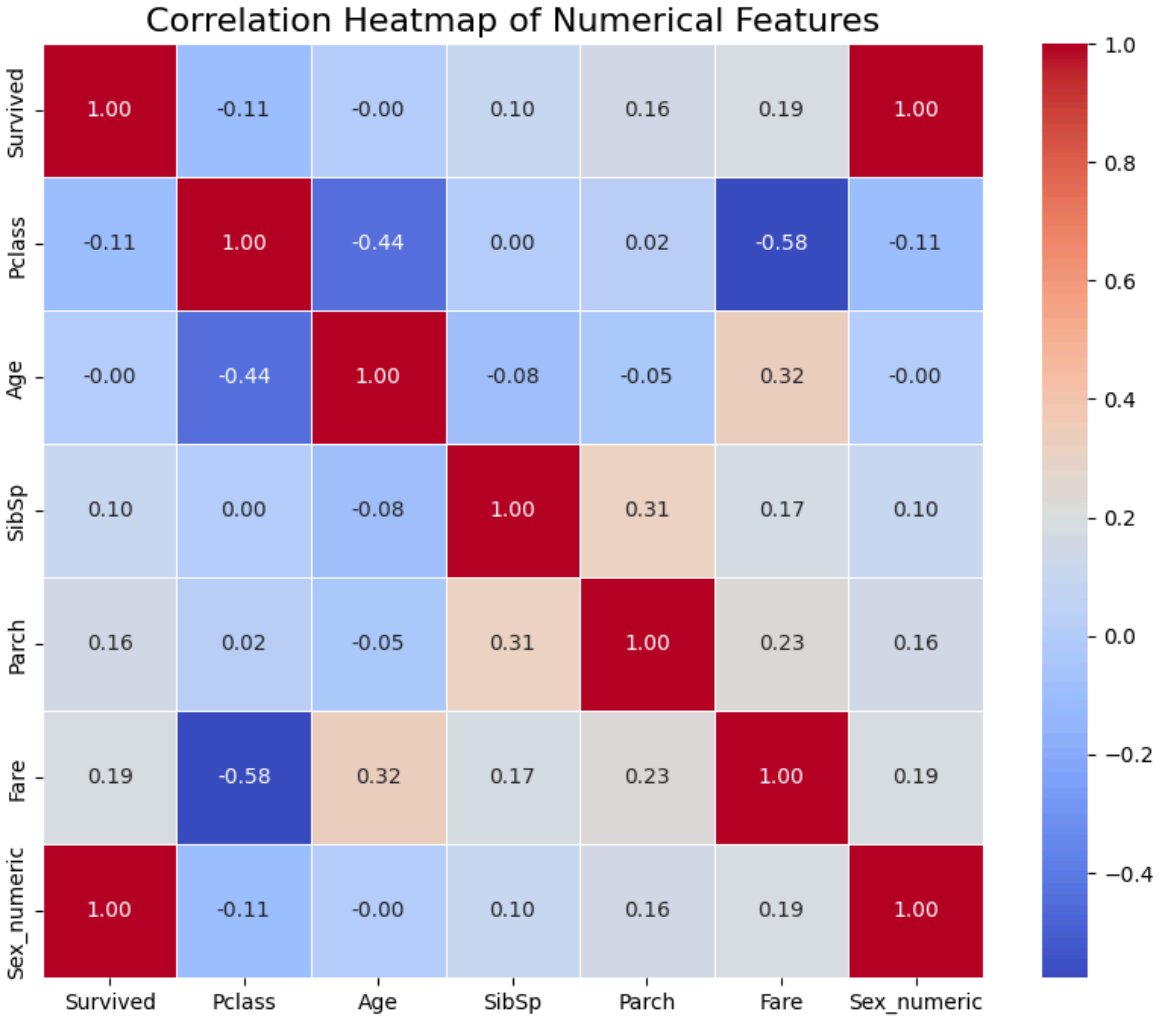
In [5]:
```
print("\n--- Univariate Analysis ---")
print("\nValue counts for key categorical columns:")
for column in ['Survived', 'Pclass', 'Sex', 'SibSp', 'Parch', 'Embarked']:
    print(f"\n--- {column} ---")
    print(df[column].value_counts())

# Plot histograms for 'Age' and 'Fare' to visualize their distributions.
plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
sns.histplot(df['Age'], kde=True, bins=30, color='skyblue')
plt.title('Distribution of Age', fontsize=16)
plt.xlabel('Age', fontsize=12)
plt.ylabel('Count', fontsize=12)

plt.subplot(1, 2, 2)
sns.histplot(df['Fare'], kde=True, bins=50, color='coral')
plt.title('Distribution of Fare', fontsize=16)
plt.xlabel('Fare', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.tight_layout()
plt.show()
```

```
--- Univariate Analysis ---

Value counts for key categorical columns:

--- Survived ---
Survived
0    266
1    152
Name: count, dtype: int64

--- Pclass ---
Pclass
3    218
1    107
2     93
Name: count, dtype: int64

--- Sex ---
Sex
male      266
female    152
Name: count, dtype: int64

--- SibSp ---
SibSp
0    283
1    110
2     14
3      4
4      4
8      2
5      1
Name: count, dtype: int64

--- Parch ---
Parch
0    324
1     52
2     33
3      3
4      2
9      2
6      1
5      1
Name: count, dtype: int64

--- Embarked ---
Embarked
S    270
C    102
Q     46
Name: count, dtype: int64
```



In [8]:
```
# Cell 5: Correlation Heatmap
# ------------------------------------------------------------------------------
print("\n--- Bivariate Analysis: Correlation Heatmap ---")
plt.figure(figsize=(10, 8))
numerical_cols = ['Survived', 'Pclass', 'Age', 'SibSp', 'Parch', 'Fare', 'Sex_nu
corr_matrix = df[numerical_cols].corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
plt.title('Correlation Heatmap of Numerical Features', fontsize=16)
plt.show()
```

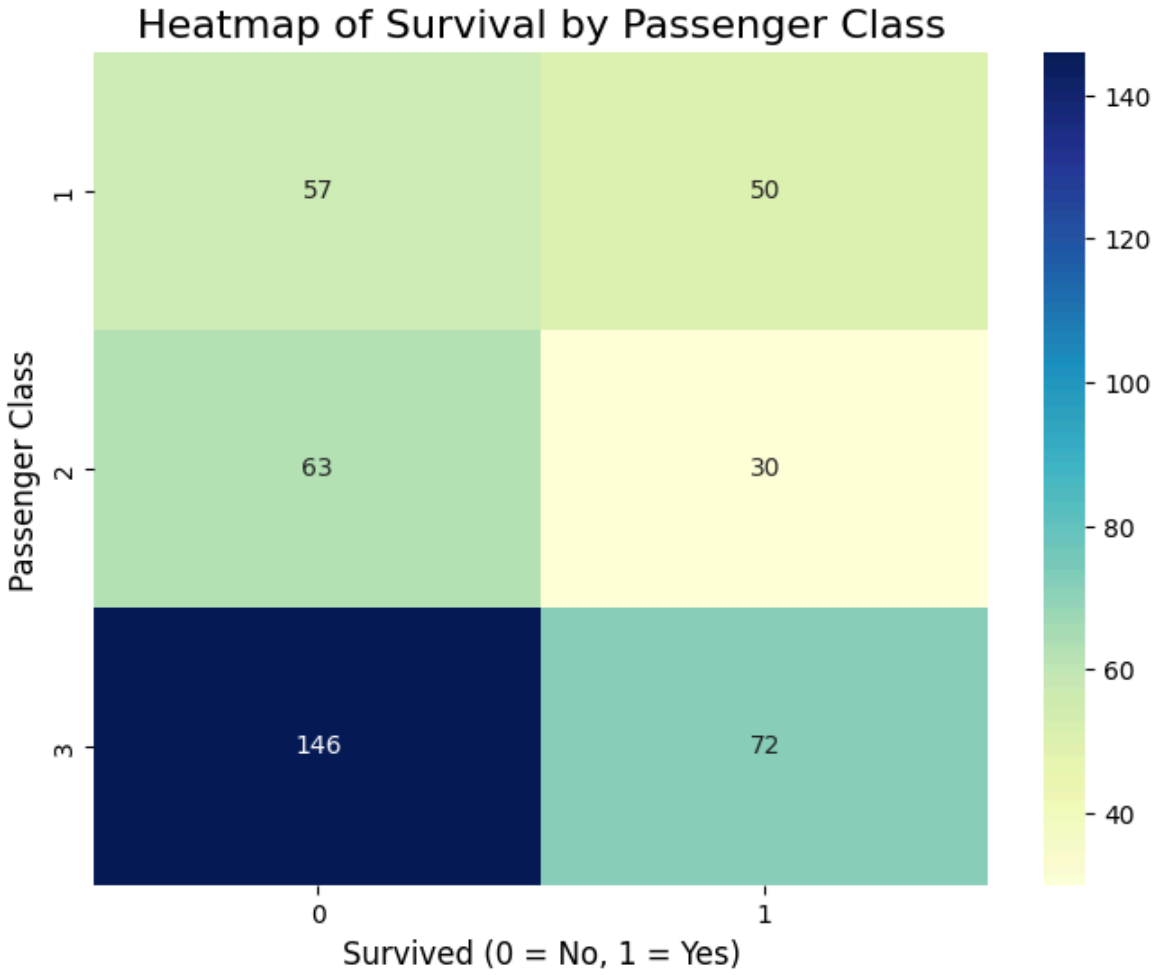
```
--- Bivariate Analysis: Correlation Heatmap ---
```



In [9]:
```
# Cell 6: Survival by Passenger Class Heatmap
# ------------------------------------------------------------------------------
```

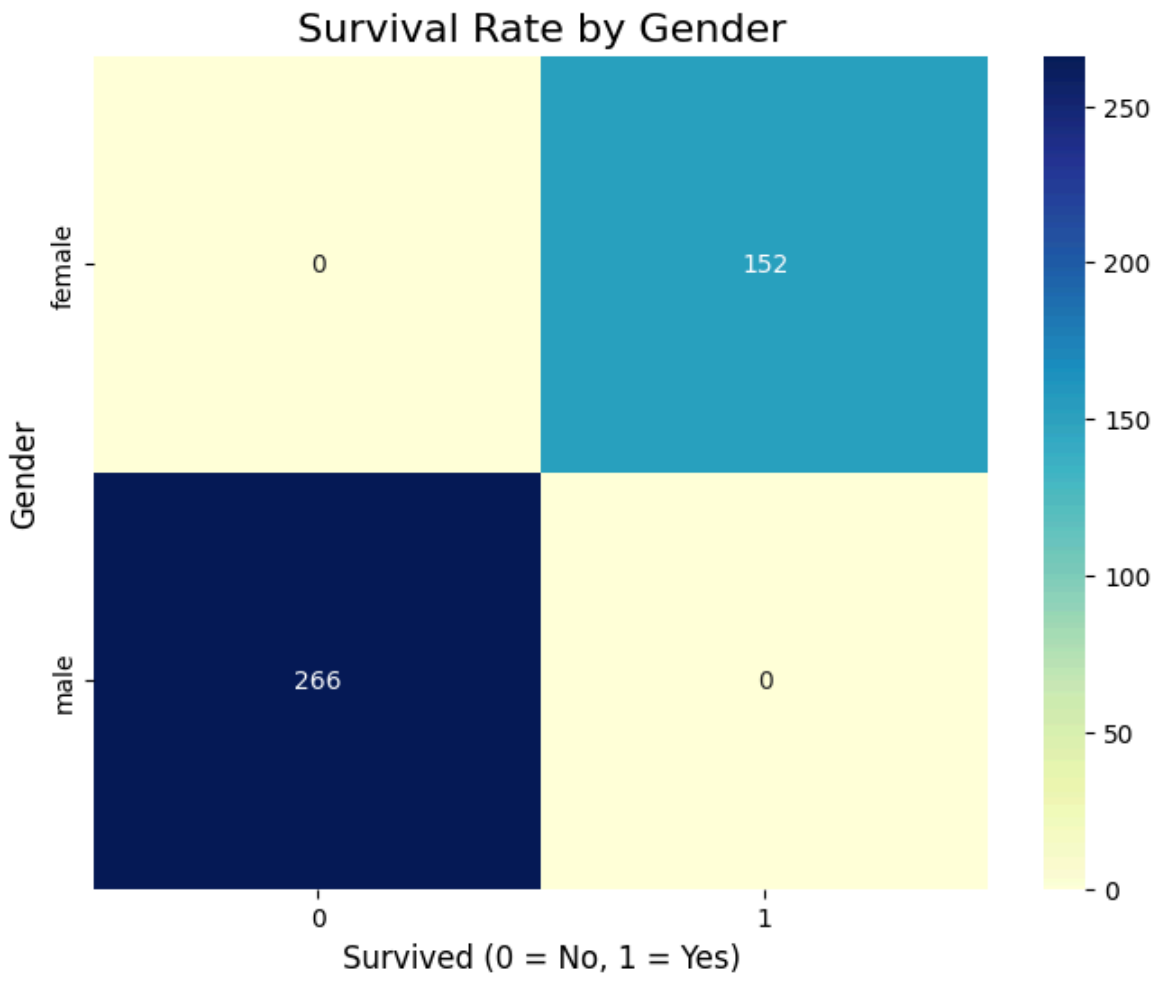
```
print("\n--- Bivariate Analysis: Survival by Passenger Class ---")
crosstab_pclass_survived = pd.crosstab(df['Pclass'], df['Survived'])
plt.figure(figsize=(8, 6))
sns.heatmap(crosstab_pclass_survived, annot=True, fmt='d', cmap='YlGnBu')
plt.title('Heatmap of Survival by Passenger Class', fontsize=16)
plt.xlabel('Survived (0 = No, 1 = Yes)', fontsize=12)
plt.ylabel('Passenger Class', fontsize=12)
plt.show()
```

```
--- Bivariate Analysis: Survival by Passenger Class ---
```



In [10]:
```
# Cell 7: Survival by Gender Heatmap
# ------------------------------------------------------------------------------
print("\n--- Bivariate Analysis: Survival by Gender ---")
crosstab_sex_survived = pd.crosstab(df['Sex'], df['Survived'])
plt.figure(figsize=(8, 6))
sns.heatmap(crosstab_sex_survived, annot=True, fmt='d', cmap='YlGnBu')
plt.title('Survival Rate by Gender', fontsize=16)
plt.xlabel('Survived (0 = No, 1 = Yes)', fontsize=12)
plt.ylabel('Gender', fontsize=12)
plt.show()
```

```
--- Bivariate Analysis: Survival by Gender ---
```



In [11]:
```
# Cell 8: Final Summary
# ------------------------------------------------------------------------------
# The final summary of the EDA, presented as a concluding text cell.
print("\n--- End of Notebook ---")
print("\nFinal Summary of Key Findings:")
print("1. Correlation Heatmap: The strongest positive correlation with survival
print("2. Pclass vs. Survival Heatmap: This visualization clearly shows that a s
print("3. Gender vs. Survival Heatmap: This final heatmap visually confirms the
print("\nThis EDA provides a solid foundation for building a predictive model fo
```

```
--- End of Notebook ---

Final Summary of Key Findings:
1. Correlation Heatmap: The strongest positive correlation with survival is with
gender ('Sex_numeric'), indicating females had a much higher survival rate. Pclas
s has a strong negative correlation, meaning passengers in higher classes had a b
etter chance of survival.
2. Pclass vs. Survival Heatmap: This visualization clearly shows that a significa
ntly larger number of passengers from the lower class (Pclass 3) did not survive,
compared to the higher classes.
3. Gender vs. Survival Heatmap: This final heatmap visually confirms the most sig
nificant finding from the correlation analysis: female passengers had a dramatica
lly higher survival count than male passengers.

This EDA provides a solid foundation for building a predictive model for Titanic
survival.
```