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

df = pd.read_csv("train.csv")
df.head()
```

Out[1]:

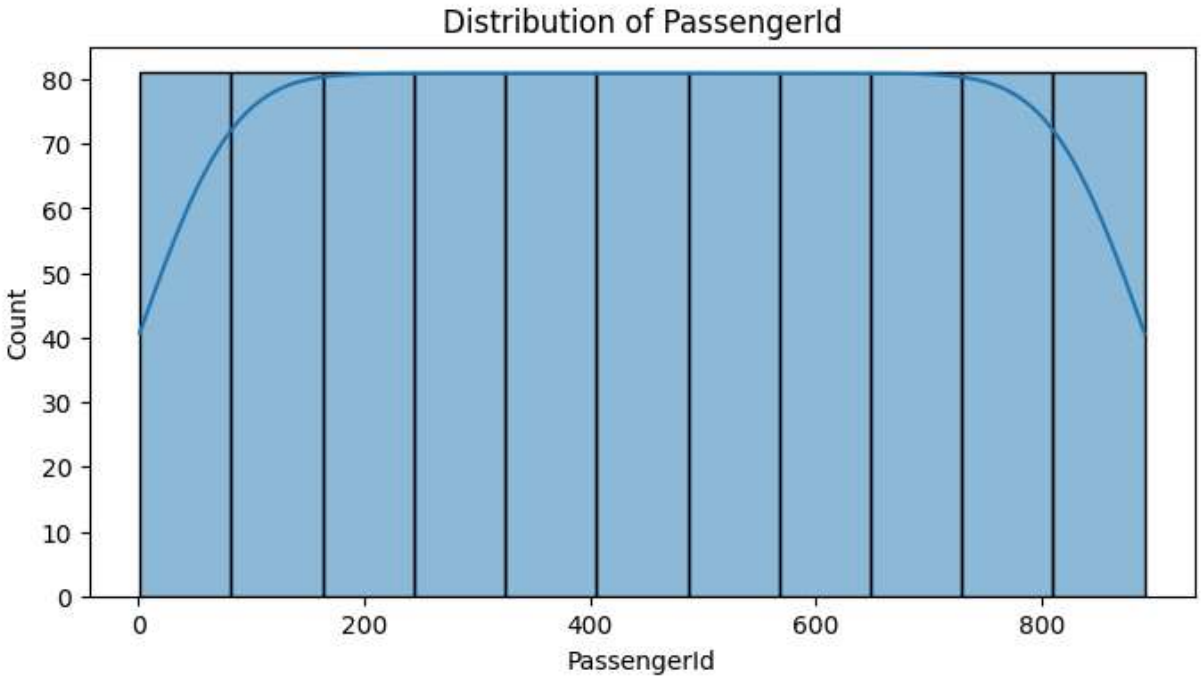
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	4	1	1	Futrelle,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

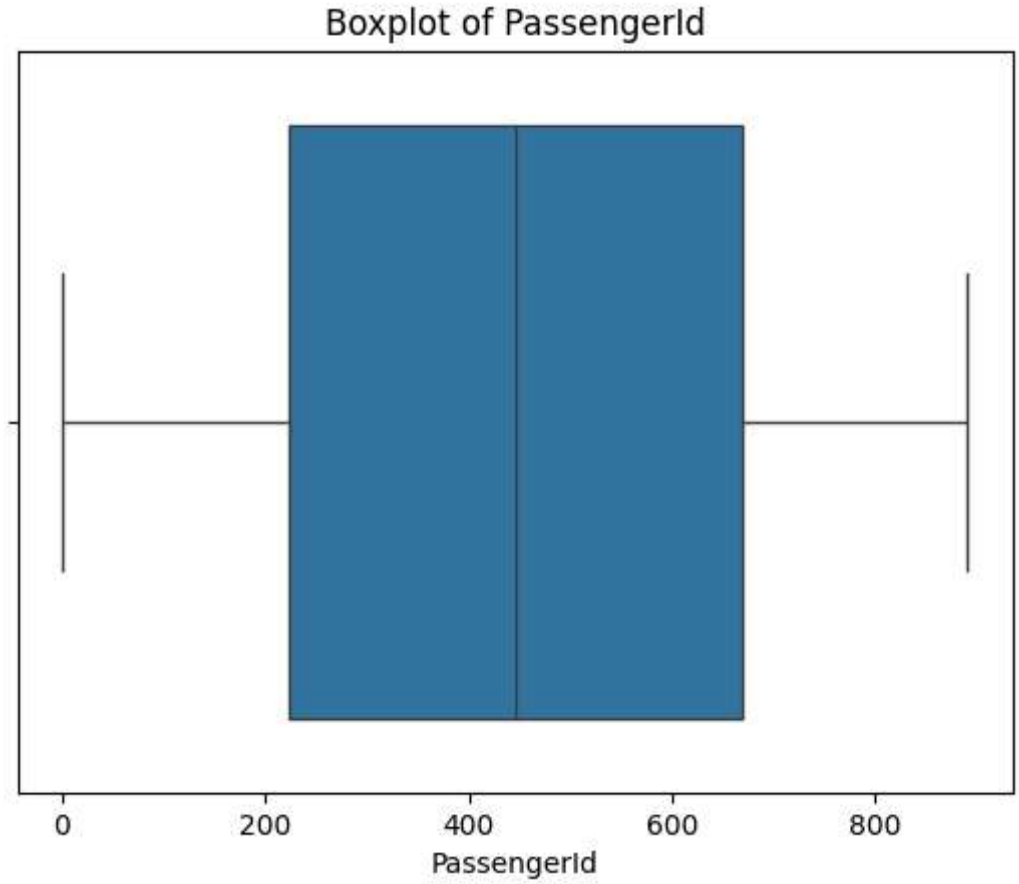
In [2]:
```
# Dataset info
df.info()

# Statistical summary
df.describe()

# Missing values
df.isnull().sum()

# Unique values in each column
df.nunique()
```

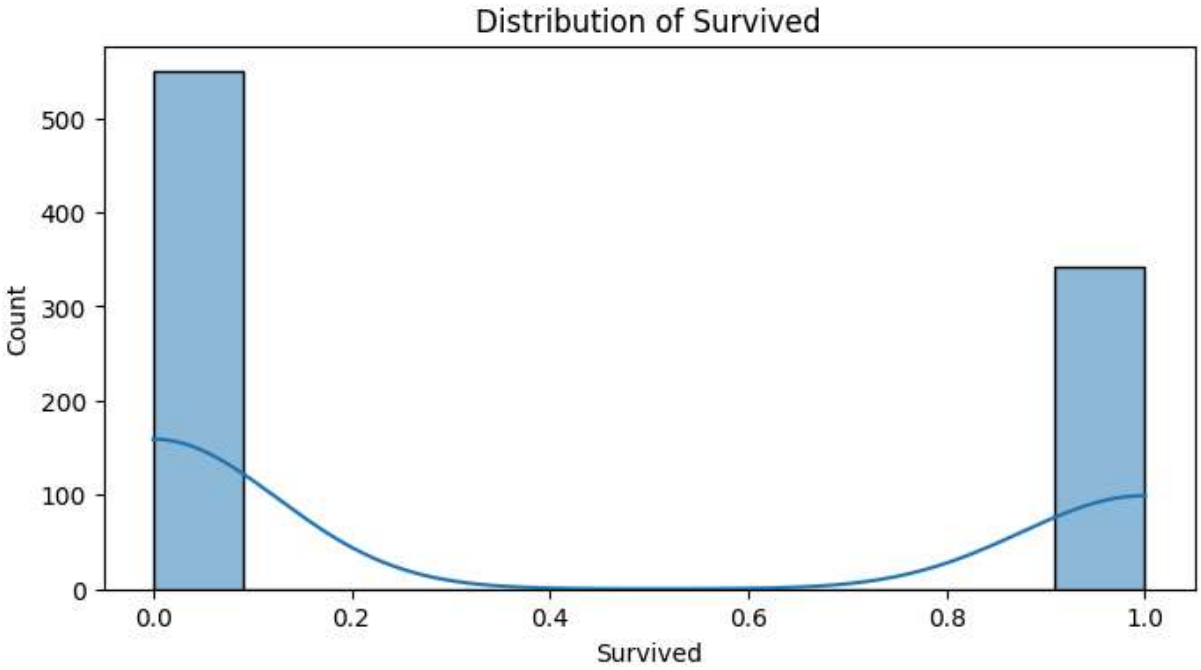
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    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
```

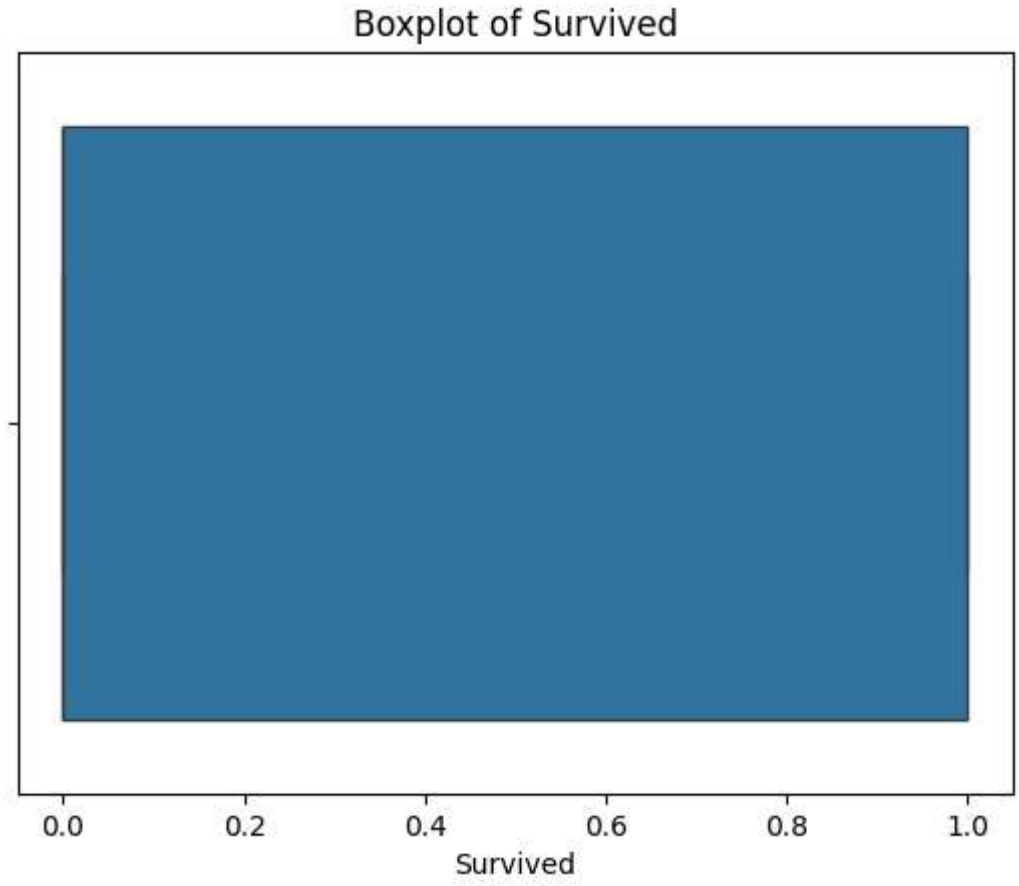
Out[2]:
```
PassengerId    891
Survived         2
Pclass           3
Name           891
Sex              2
Age             88
SibSp            7
Parch            7
Ticket         681
Fare           248
Cabin          147
Embarked         3
dtype: int64
```

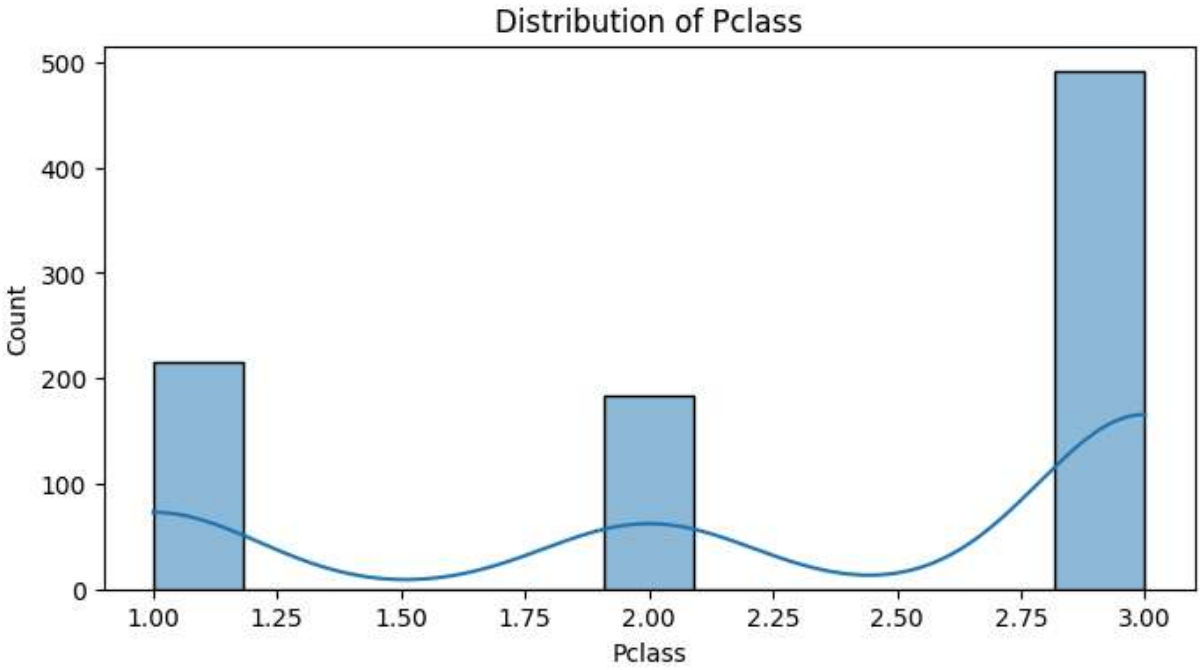
In [ ]:
```
# num_cols = df.select_dtypes(include=['int64','float64']).columns

# for col in num_cols:
#     plt.figure(figsize=(8,4))
#     sns.histplot(df[col], kde=True)
#     plt.title(f'Distribution of {col}')
#     plt.show()

#     sns.boxplot(x=df[col])
#     plt.title(f'Boxplot of {col}')
#     plt.show()
```

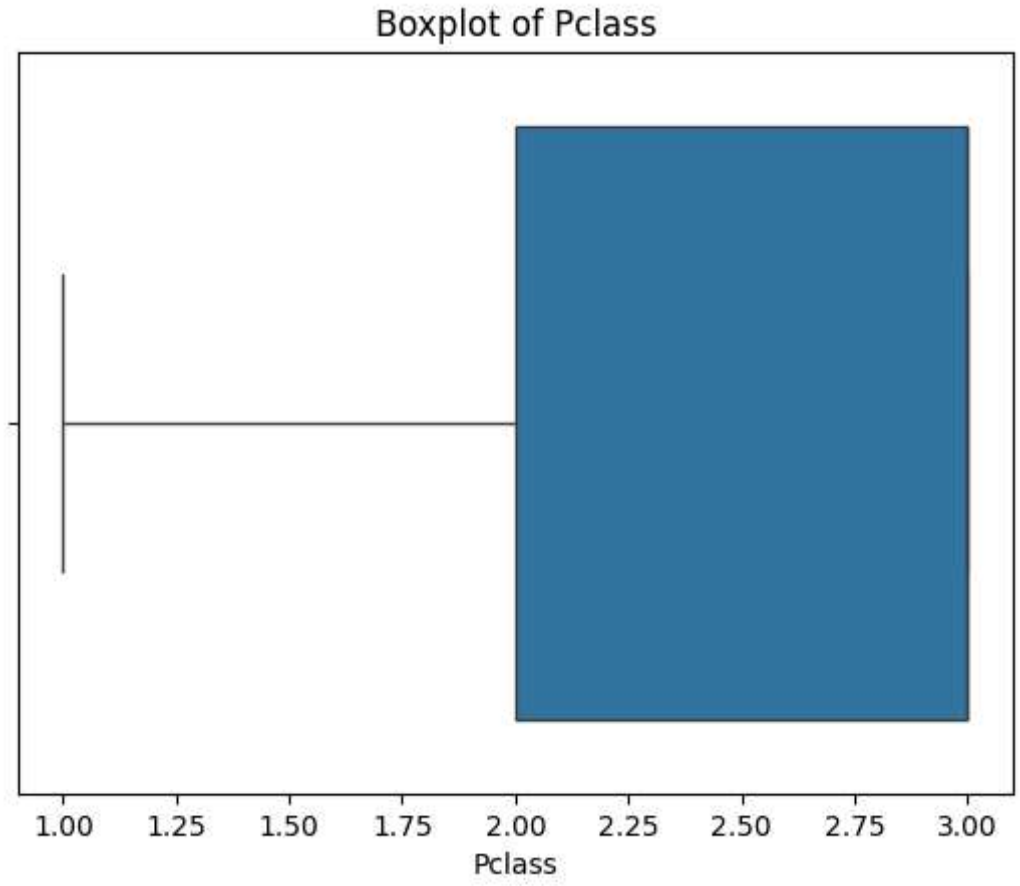

Distribution of PassengerId
Count
PassengerId
0
10
20
30
40
50
60
70
80
0
200
400
600
800

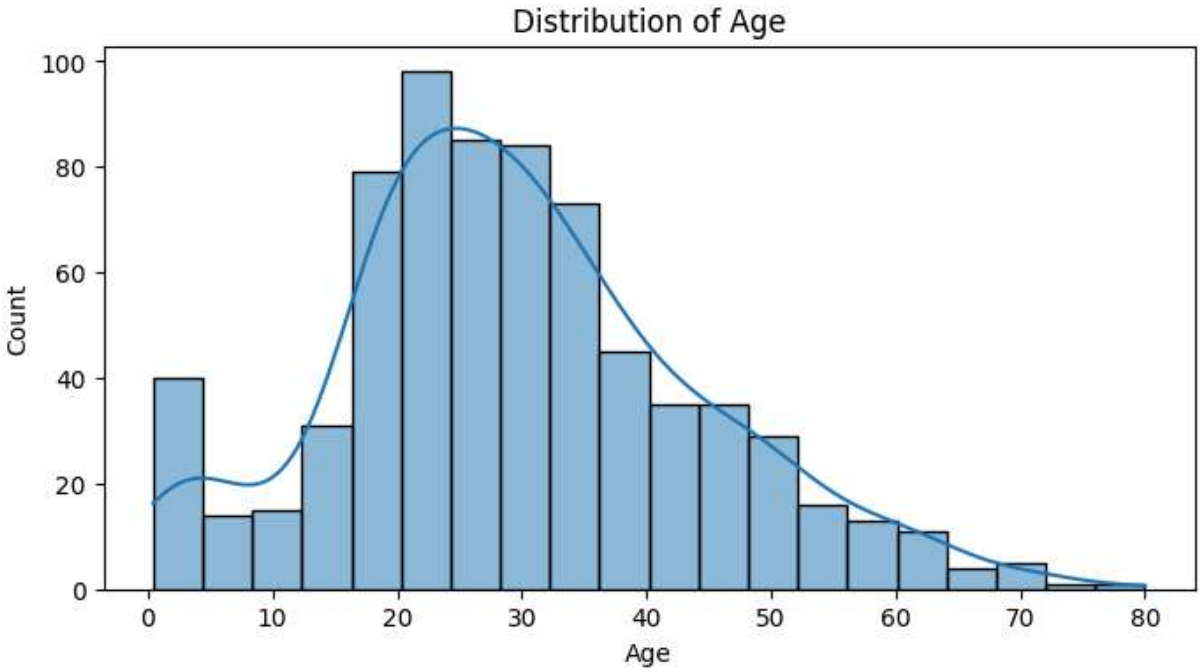

Boxplot of PassengerId
PassengerId
0
200
400
600
800

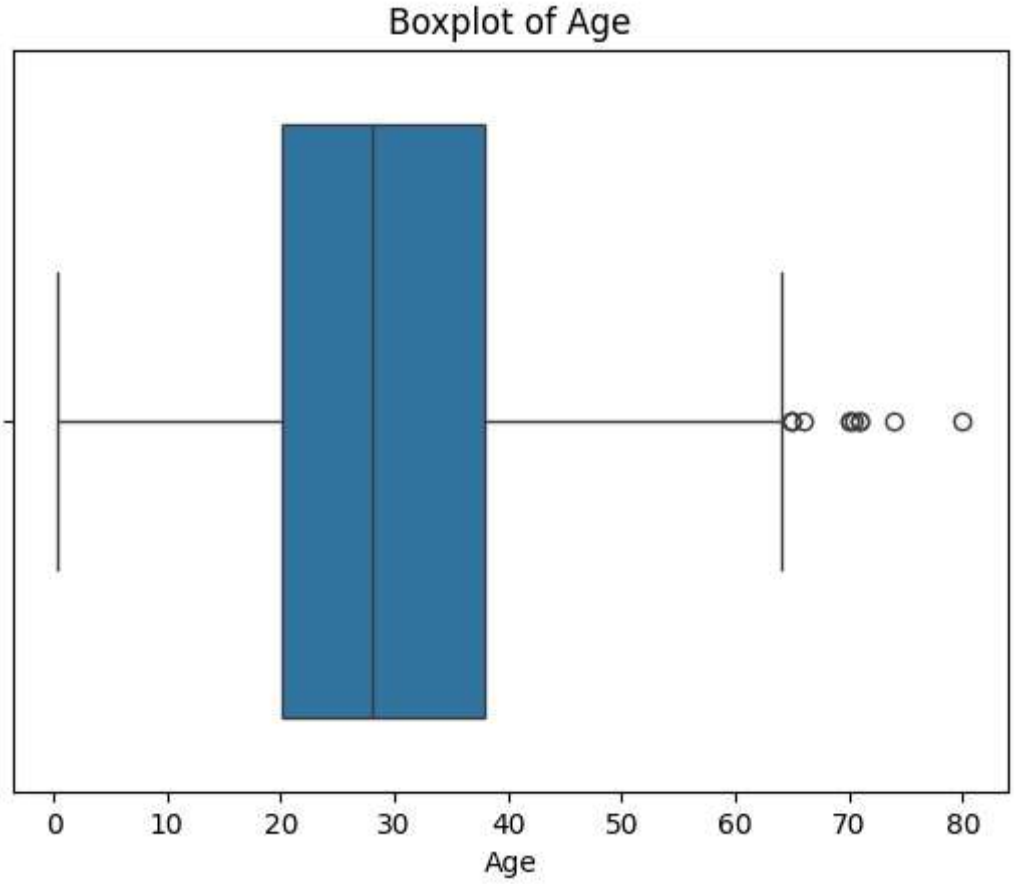

Distribution of Survived
Count
500
400
300
200
100
0
0.0
0.2
0.4
0.6
0.8
1.0
Survived

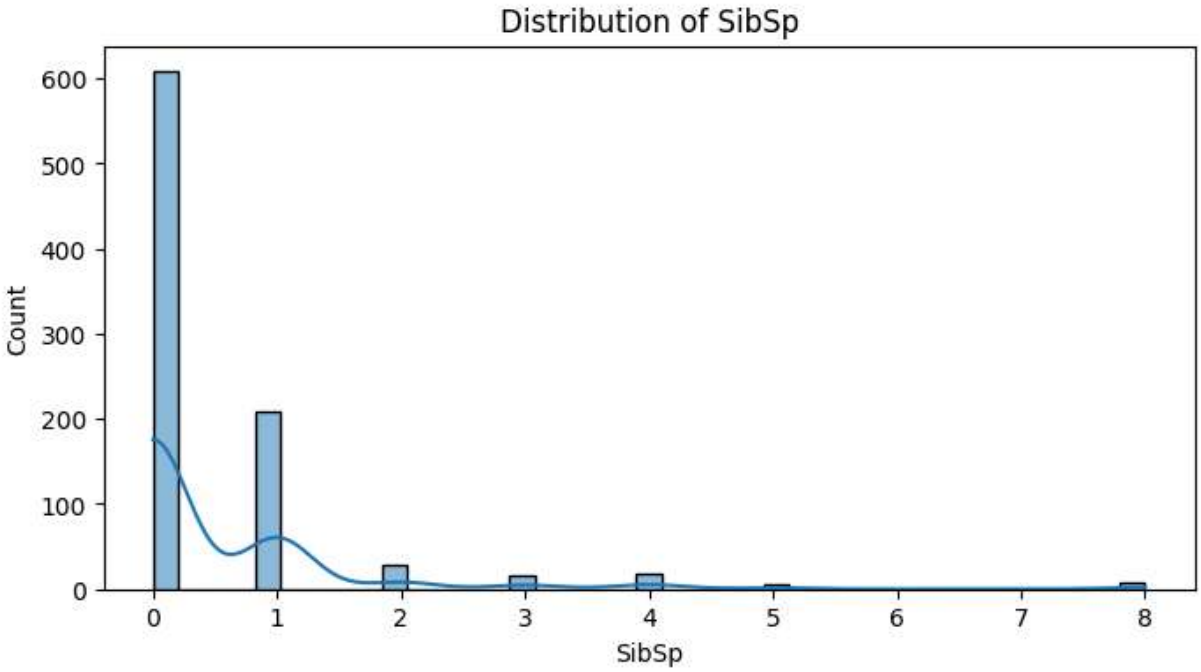

Boxplot of Survived
0.0
0.2
0.4
0.6
0.8
1.0
Survived

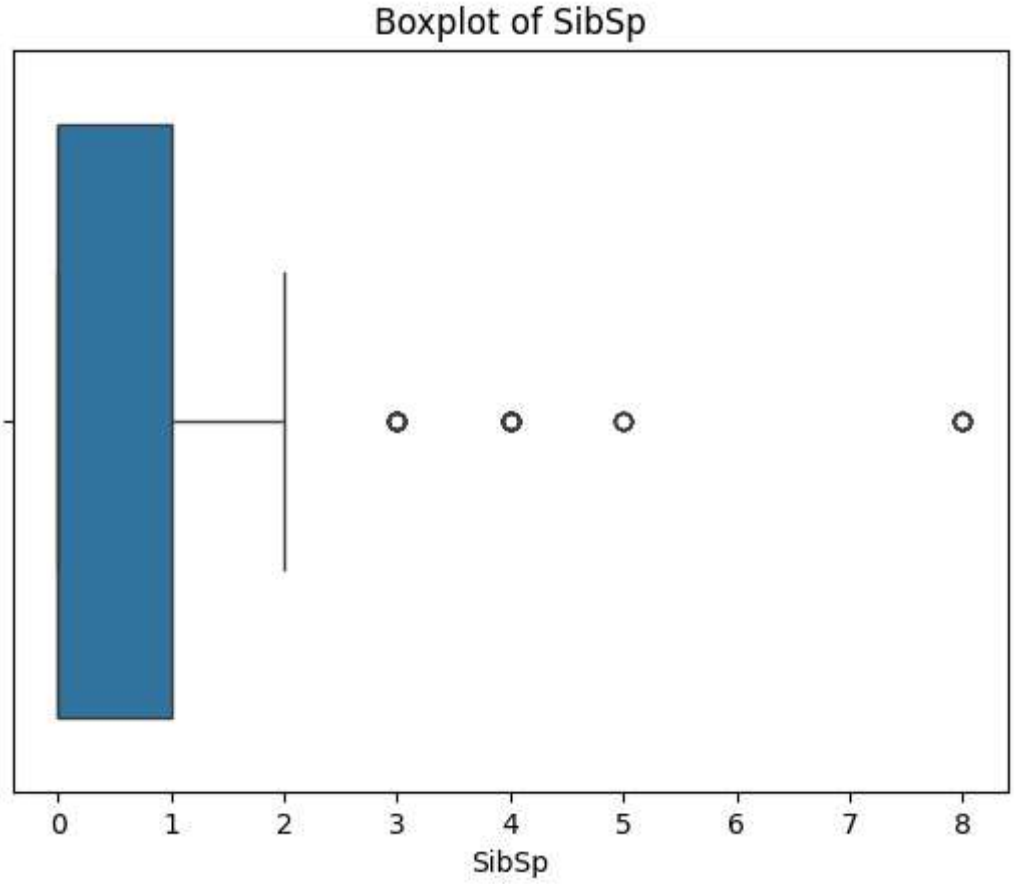

Distribution of Pclass
Count
Pclass
0
100
200
300
400
500
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00

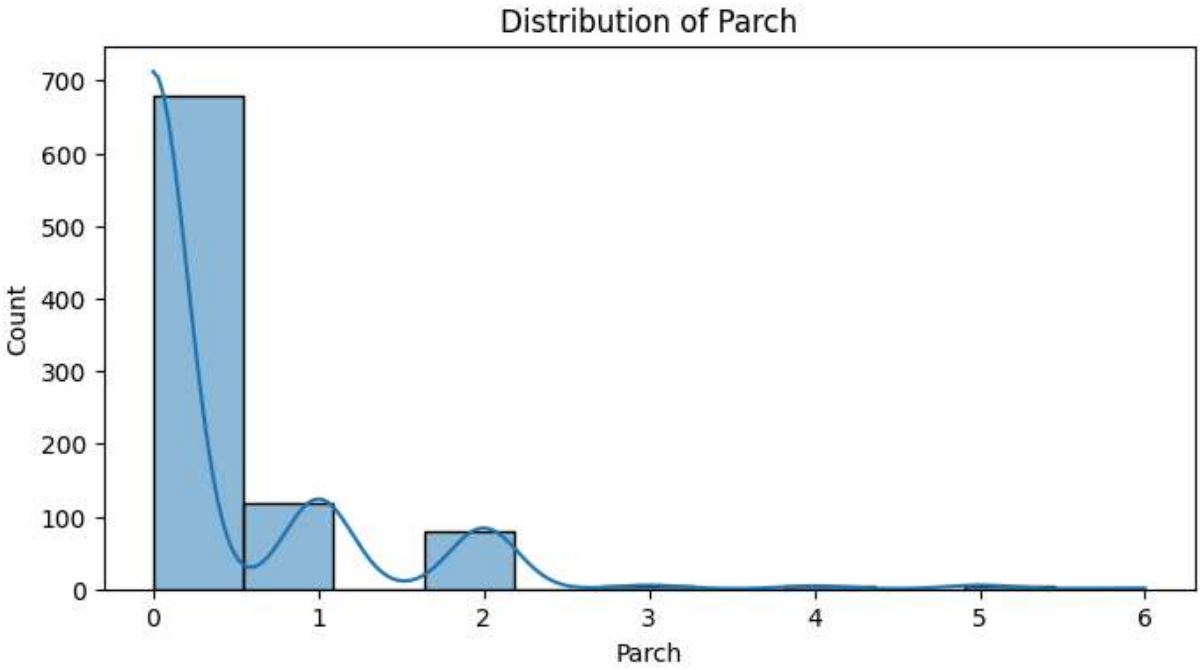

Boxplot of Pclass
Pclass
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00

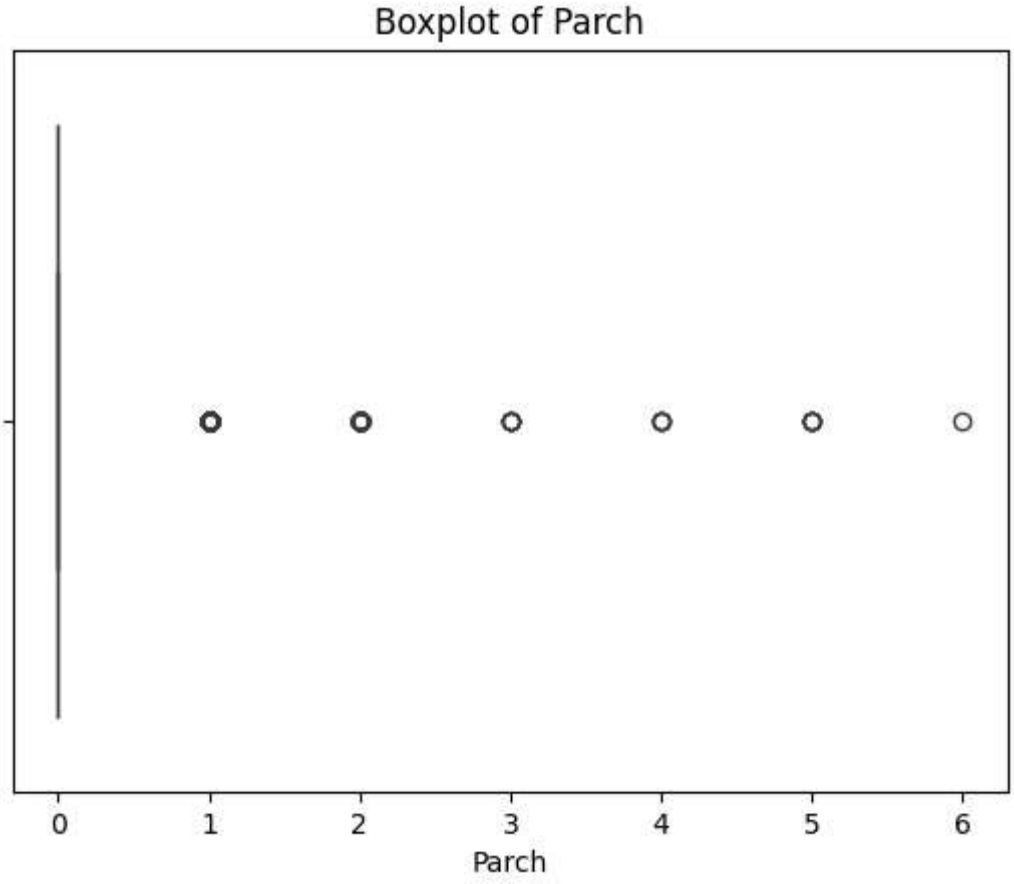

Distribution of Age
Count
Age
0
20
40
60
80
100
0
10
20
30
40
50
60
70
80

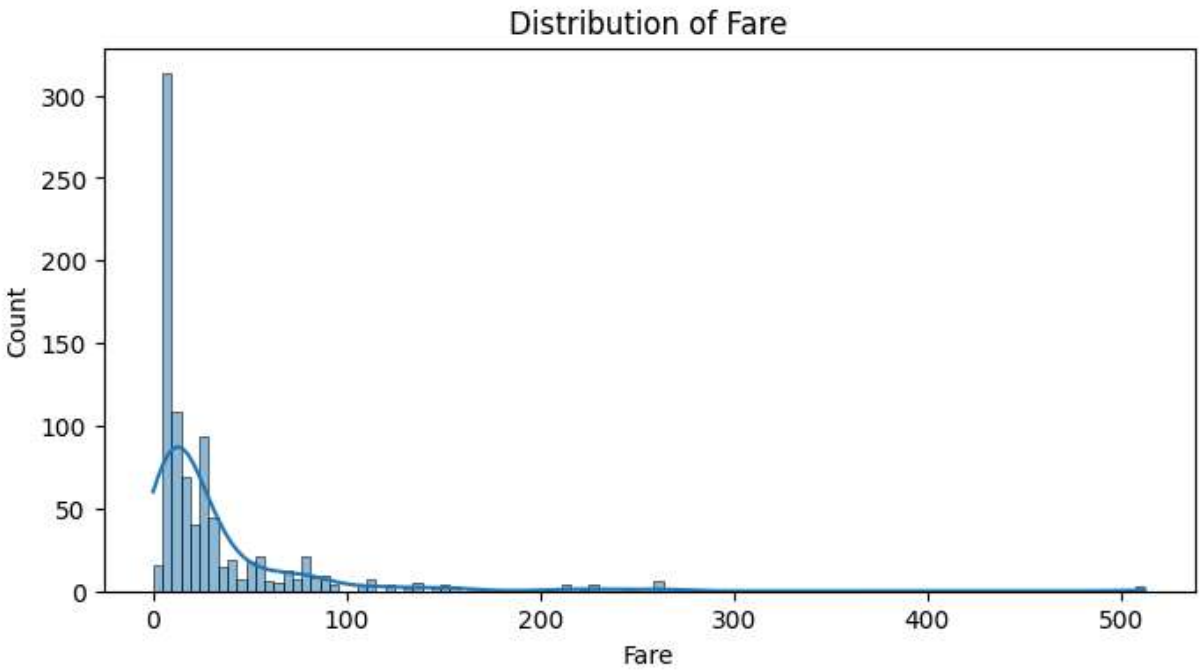

Boxplot of Age
Age
0
10
20
30
40
50
60
70
80

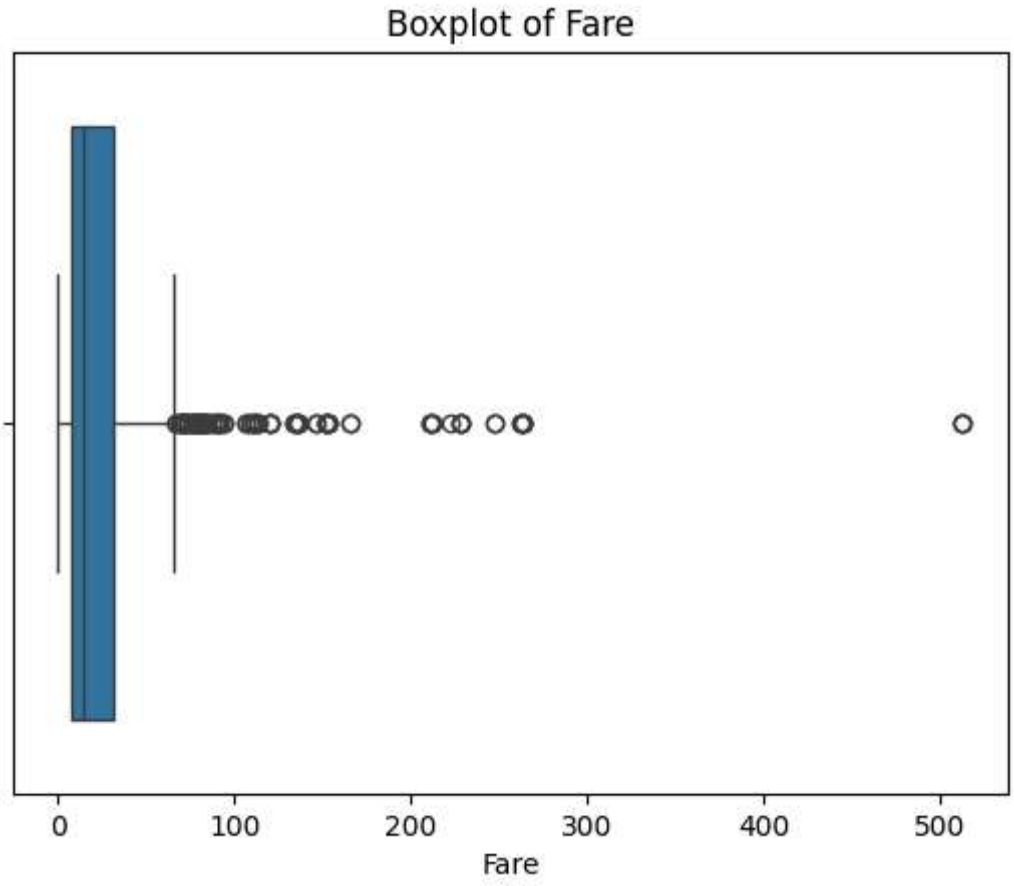

Distribution of SibSp
Count
SibSp
0
100
200
300
400
500
600
0
1
2
3
4
5
6
7
8


Boxplot of SibSp
SibSp
0
1
2
3
4
5
6
7
8

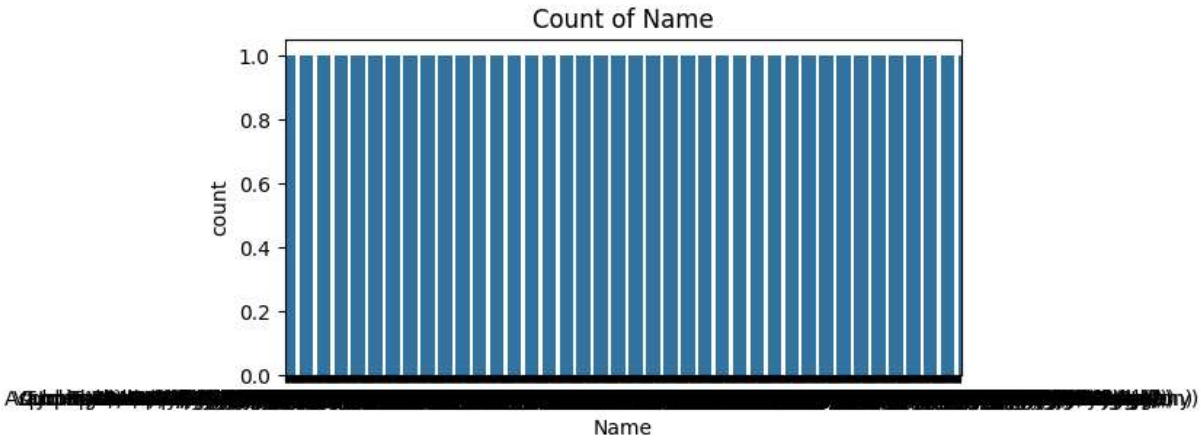

Distribution of Parch
Count
Parch
0
100
200
300
400
500
600
700
0
1
2
3
4
5
6

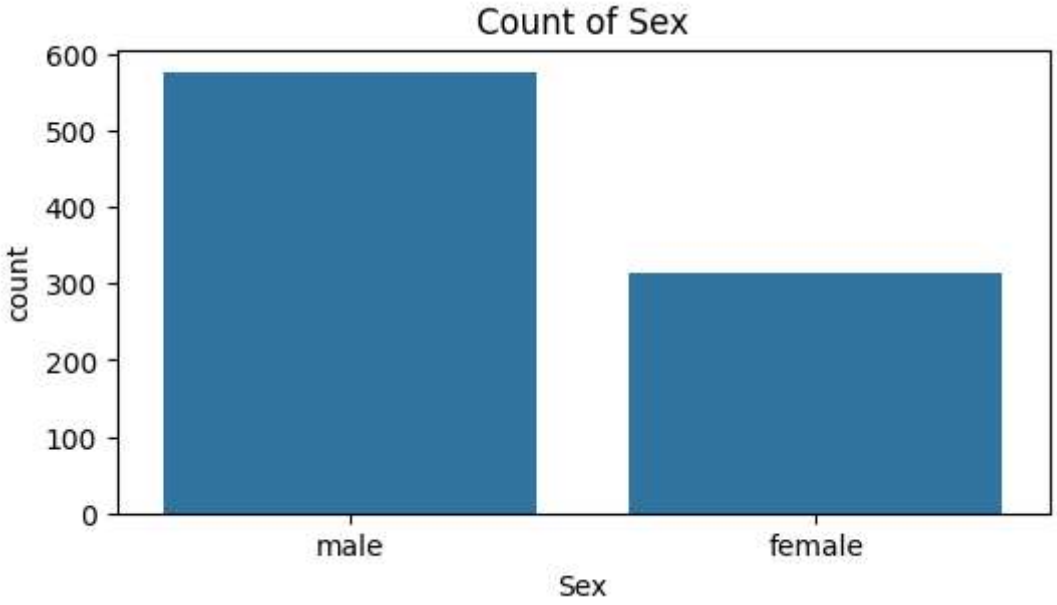

Boxplot of Parch
Parch
0
1
2
3
4
5
6

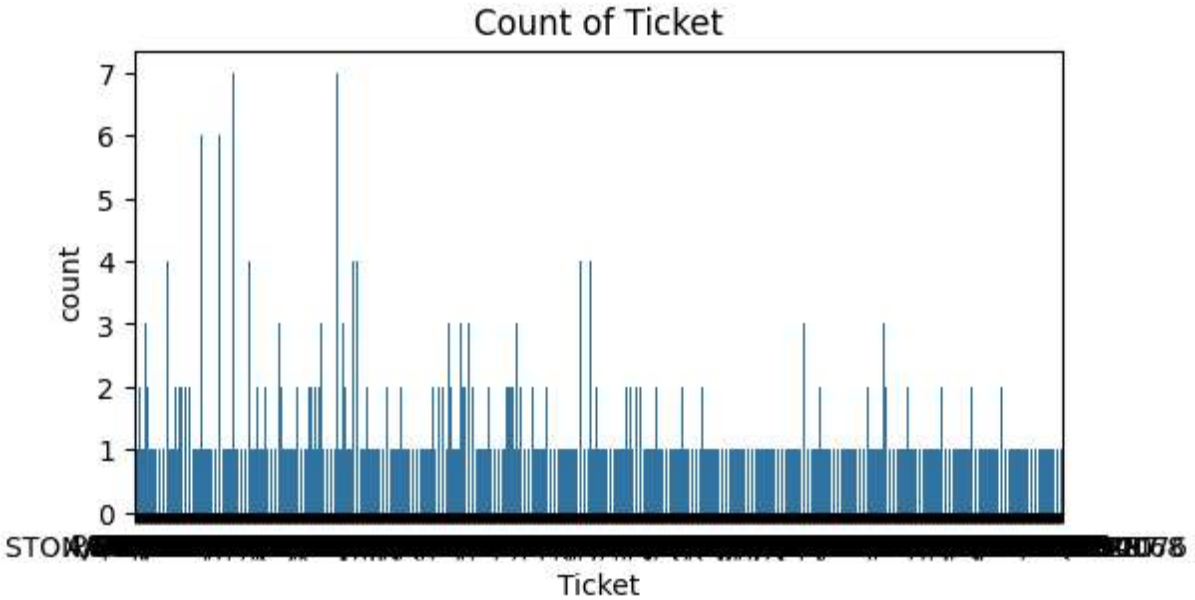


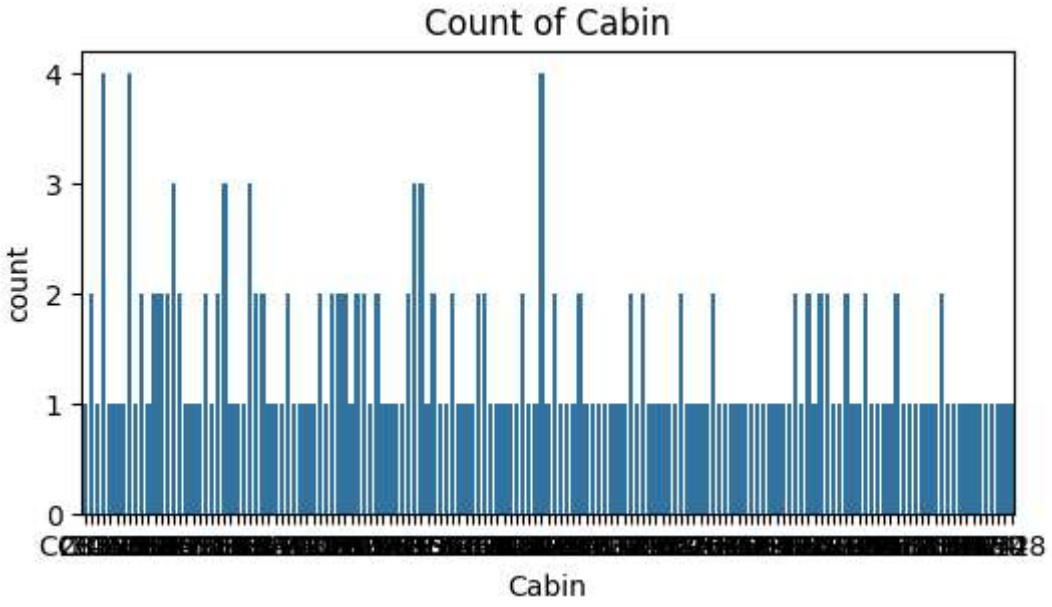


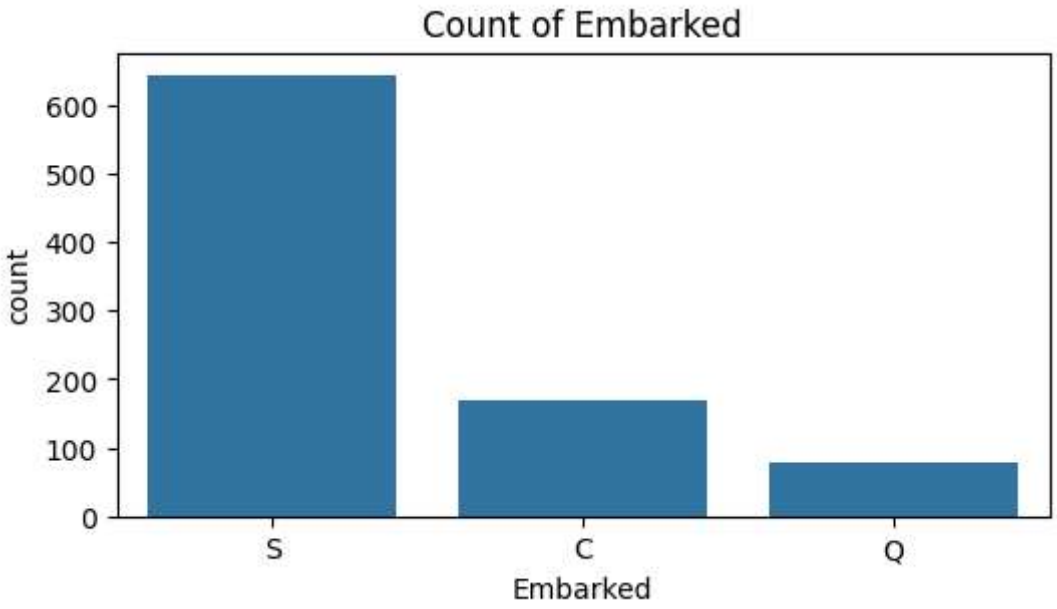
In [4]:
```
cat_cols = df.select_dtypes(include=['object']).columns

for col in cat_cols:
    plt.figure(figsize=(6,3))
    sns.countplot(data=df, x=col)
    plt.title(f'Count of {col}')
    plt.show()
```


Count of Name
1.0
0.8
0.6
0.4
0.2
0.0
count
Name

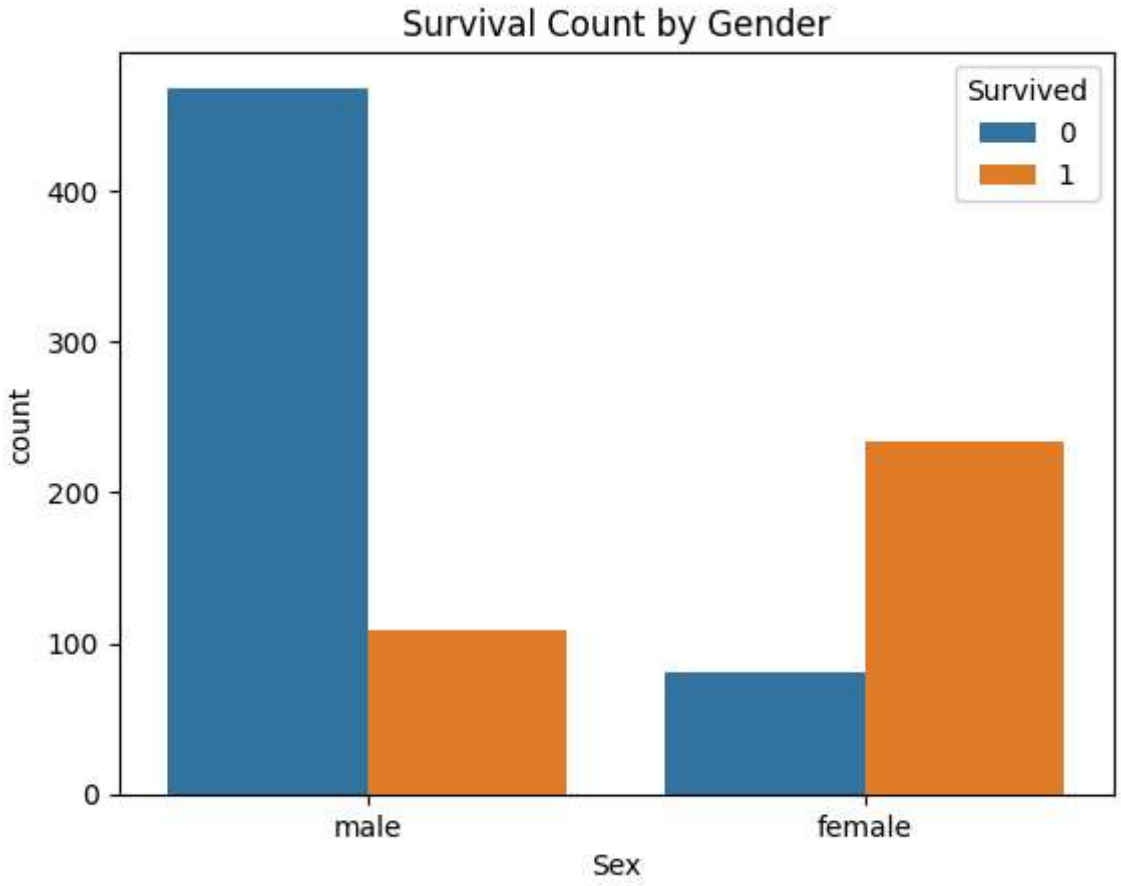

Count of Sex
600
500
400
300
200
100
0
count
male
female
Sex

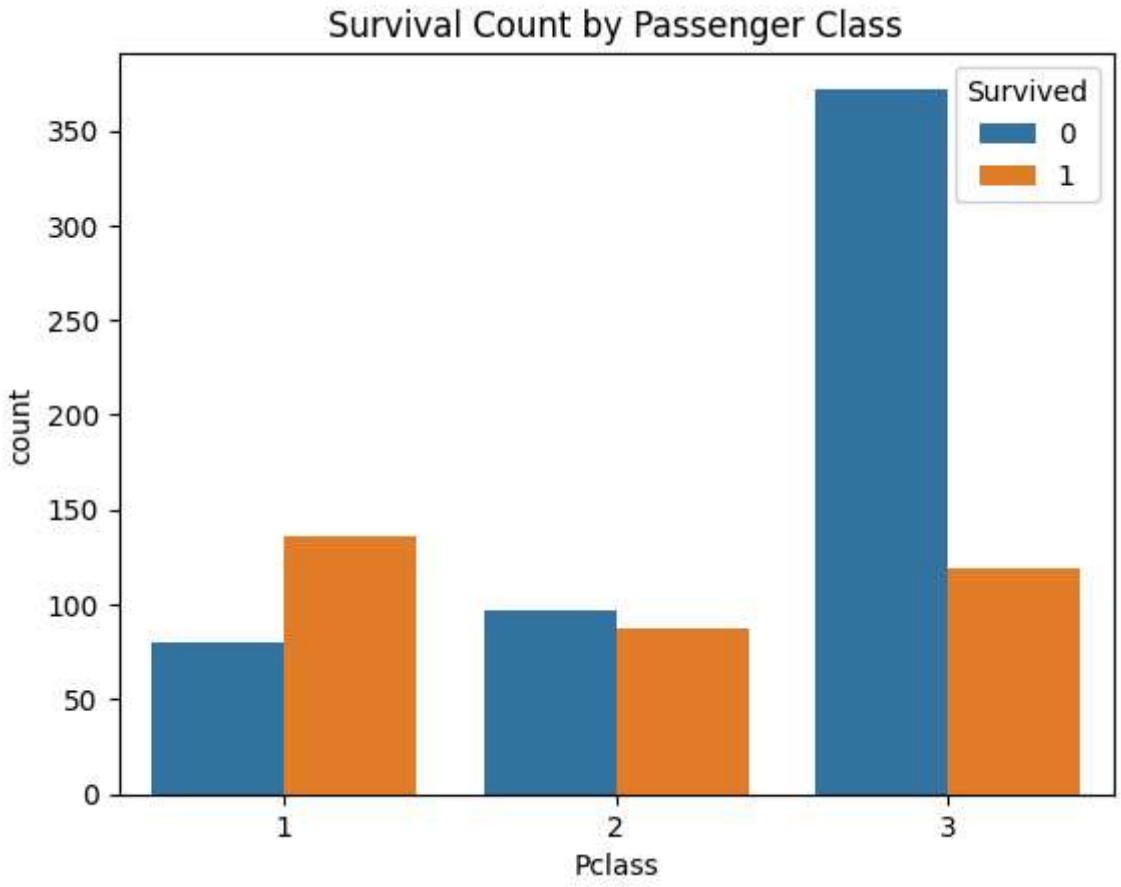

Count of Ticket
7
6
5
4
3
2
1
0
count
Ticket

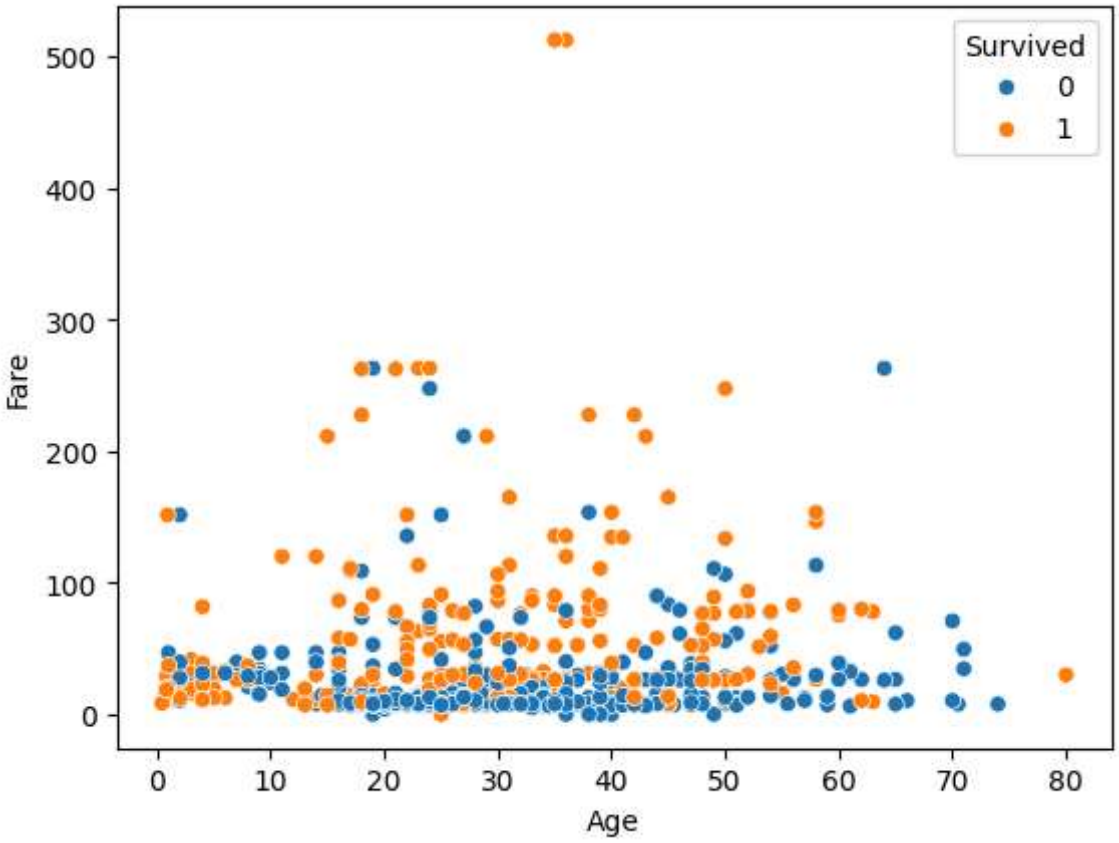




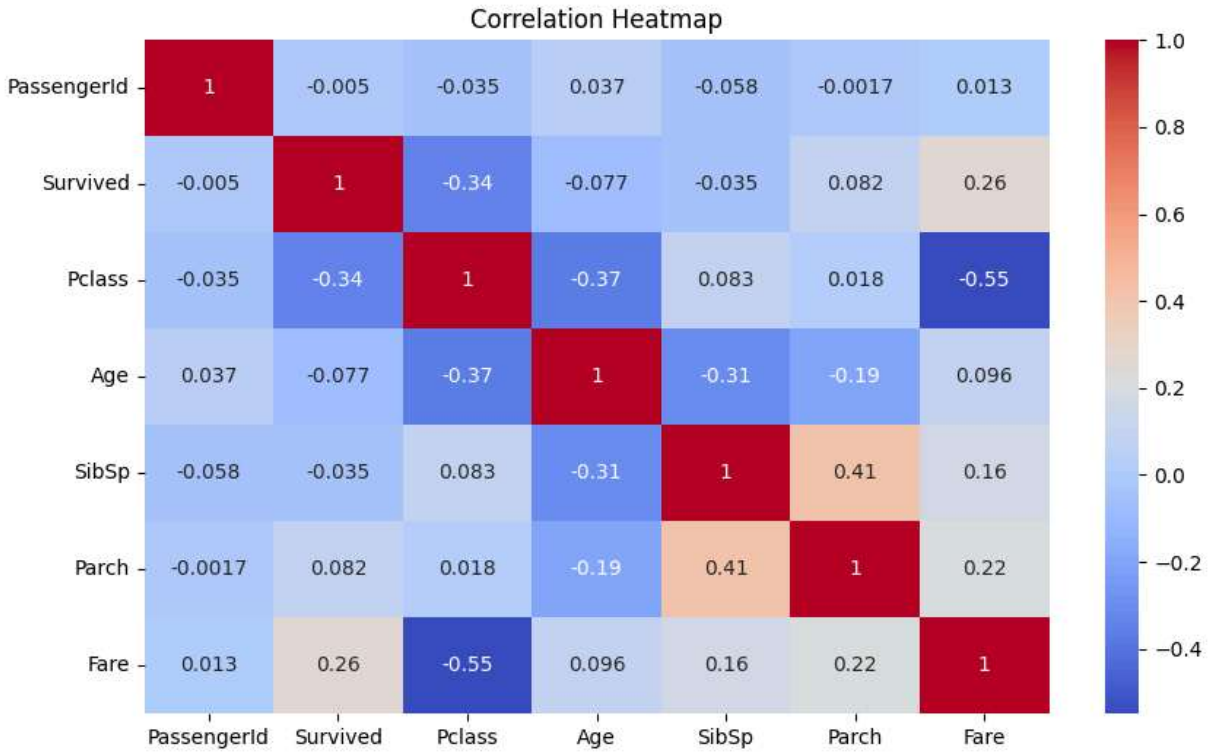
In [5]:
```
# Survival rate by gender
sns.countplot(data=df, x='Sex', hue='Survived')
plt.title('Survival Count by Gender')
plt.show()

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

# Age vs Fare scatter plot
sns.scatterplot(data=df, x='Age', y='Fare', hue='Survived')
plt.show()
```


Survival Count by Gender
Survived
0
1
400
300
200
100
0
count
male
female
Sex


Survival Count by Passenger Class
Survived
0
1
350
300
250
200
150
100
50
0
count
1
2
3
Pclass



In [9]:
```
# Select only numeric columns for correlation
numeric_df = df.select_dtypes(include=['number'])

# Plot heatmap
plt.figure(figsize=(10,6))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



In [10]:
```
# Fill missing Age with median
df['Age'].fillna(df['Age'].median(), inplace=True)

# Fill Embarked with mode
df['Embarked'].fillna(df['Embarked'].mode()[0], inplace=True)
```

```
C:\Users\LENOVO\AppData\Local\Temp\ipykernel_8096\1481385193.py:2: FutureWarning: A
value is trying to be set on a copy of a DataFrame or Series through chained assignm
ent using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

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


  df['Age'].fillna(df['Age'].median(), inplace=True)
C:\Users\LENOVO\AppData\Local\Temp\ipykernel_8096\1481385193.py:5: FutureWarning: A
value is trying to be set on a copy of a DataFrame or Series through chained assignm
ent using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

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


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