NAME: P.NAGA BHAVANI

SLOT: MORNING SLOT(10-12)

CAMPUS: VIT AP

REGISTRATION NO.: 21BCE9043

ASSIGNMENT NO.: 3

Assignment 15 sep

Perform Data preprocessing on Titanic dataset 1.Data Collection. Please download the dataset from https://www.kaggle.com/datasets/yasserh/titanic-datase

In [ ]:
```
2.Data Preprocessing
    o   Import the Libraries.
    o   Importing the dataset.
    o   Checking for Null Values.
    o   Data Visualization.
    o   Outlier Detection
    o   Splitting Dependent and Independent variables
    o   Perform Encoding
    o   Feature Scaling.
    o   Splitting Data into Train and Test
```

1. IMPORT THE LIBRARIES

In [62]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

2. IMPORT THE DATASET

In [63]:
```
df=pd.read_csv("Titanic-Dataset.csv")
```

In [64]:
```
df
```

Loading [MathJax]/extensions/Safe.js

Out[64]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	T
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STC 31
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	3
...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	2
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	1
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	1
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	3

891 rows × 12 columns

In [65]:
```
df.head()
```

Loading [MathJax]/extensions/Safe.js

Out[65]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tic
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	21
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/ 3101
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373

In [66]:
```
df.tail()
```

Out[66]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tic
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W 6
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370

In [67]:
```
df.shape
```

Out[67]:
```
(891, 12)
```

In [68]:
```
df.info()
```

Loading [MathJax]/extensions/Safe.js

```
<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
```

In [69]:
```
df.describe()
```

Out[69]:

	PassengerId	Survived	Pclass	Age	SibSp	Parc
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.00000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.38159
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.80605
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.00000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.00000
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.00000
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.00000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.00000

In [70]:
```
corr=df.corr()
corr
```

```
C:\Users\NAGA BHAVANI\AppData\Local\Temp\ipykernel_1968\3182140910.py:1: Fut
ureWarning: The default value of numeric_only in DataFrame.corr is deprecate
d. 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=df.corr()
```

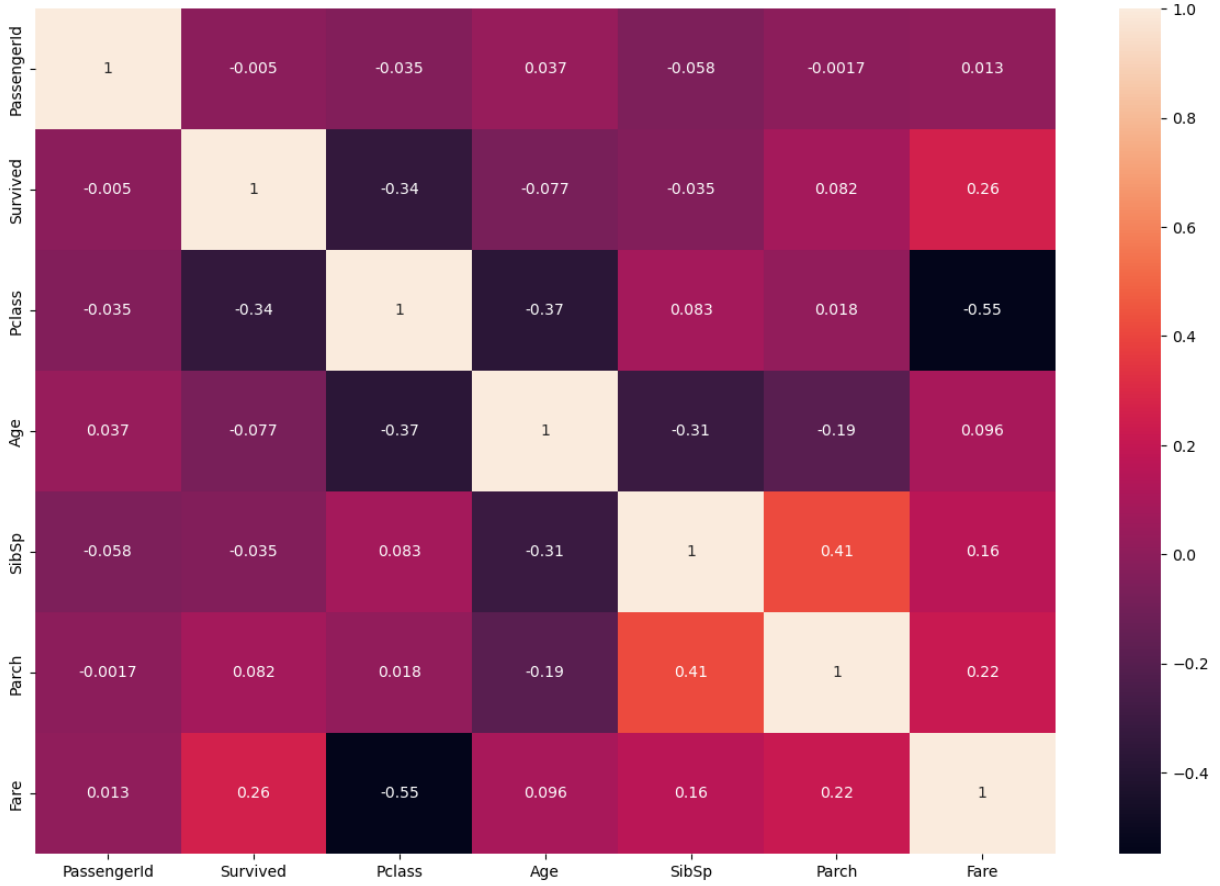
Loading [MathJax]/extensions/Safe.js

Out[70]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443
Age	0.036847	-0.077221	-0.369226	1.000000	-0.308247	-0.189119
SibSp	-0.057527	-0.035322	0.083081	-0.308247	1.000000	0.414838
Parch	-0.001652	0.081629	0.018443	-0.189119	0.414838	1.000000
Fare	0.012658	0.257307	-0.549500	0.096067	0.159651	0.216225

In [71]:
```
plt.subplots(figsize=(15,10))
sns.heatmap(corr,annot=True)
```

Out[71]: <Axes: >



In [72]:
```
df.Survived.value_counts()
```

Out[72]:
```
0    549
1    342
Name: Survived, dtype: int64
```

In [73]:
```
df.Sex.value_counts()
```

Loading [MathJax]/extensions/Safe.js

Out[73]:
```
male      577
female    314
Name: Sex, dtype: int64
```

In [74]:
```
df.Embarked.value_counts()
```

Out[74]:
```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

3. CHECK FOR NULL VALUES

In [75]:
```
df.isnull().any()
```

Out[75]:
```
PassengerId    False
Survived       False
Pclass         False
Name           False
Sex            False
Age             True
SibSp          False
Parch          False
Ticket         False
Fare           False
Cabin           True
Embarked        True
dtype: bool
```

In [76]:
```
df.isnull().sum()
```

Out[76]:
```
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
```

In [77]:
```
mean_age = df['Age'].mean()
df['Age'].fillna(mean_age, inplace=True)
```

Fill null values in the 'Embarked' column with the most common value

In [78]:
```
most_common_embarked = df['Embarked'].mode()[0]
df['Embarked'].fillna(most_common_embarked, inplace=True)
```

Loading [MathJax]/extensions/Safe.js

In [79]:
```
df.drop(['Cabin'],axis=1, inplace=True)
```

In [80]:
```
df.drop(['Ticket'],axis=1, inplace=True)
```

In [81]:
```
df.drop(['Name'],axis=1,inplace=True)
```

In [82]:
```
print(df.isnull().sum())
```

```
PassengerId    0
Survived       0
Pclass         0
Sex            0
Age            0
SibSp          0
Parch          0
Fare           0
Embarked       0
dtype: int64
```

4. DATA VISUALIZATION

In [83]:
```
df_cleaned.corr()
```

```
C:\Users\NAGA BHAVANI\AppData\Local\Temp\ipykernel_1968\1367570080.py:1: Fut
ureWarning: The default value of numeric_only in DataFrame.corr is deprecate
d. In a future version, it will default to False. Select only valid columns
or specify the value of numeric_only to silence this warning.
  df_cleaned.corr()
```

Out[83]:

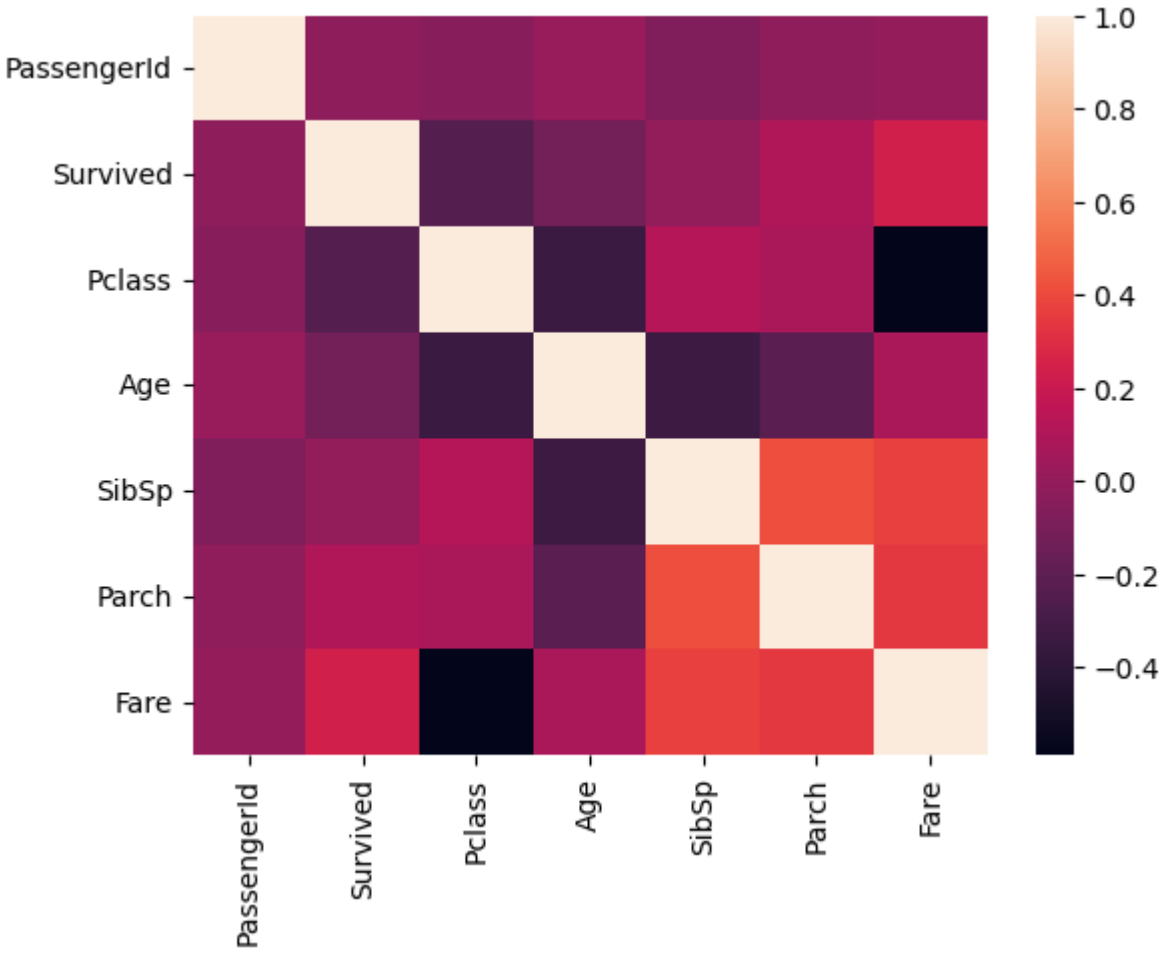
	PassengerId	Survived	Pclass	Age	SibSp	Parch
PassengerId	1.000000	-0.024438	-0.045179	0.015393	-0.080426	-0.015117
Survived	-0.024438	1.000000	-0.238532	-0.121283	-0.003156	0.098575
Pclass	-0.045179	-0.238532	1.000000	-0.342826	0.113943	0.083451
Age	0.015393	-0.121283	-0.342826	1.000000	-0.340947	-0.212343
SibSp	-0.080426	-0.003156	0.113943	-0.340947	1.000000	0.410182
Parch	-0.015117	0.098575	0.083451	-0.212343	0.410182	1.000000
Fare	0.002942	0.234422	-0.589776	0.078863	0.370388	0.336844

In [84]:
```
sns.heatmap(df_cleaned.corr())
```

```
C:\Users\NAGA BHAVANI\AppData\Local\Temp\ipykernel_1968\3970833705.py:1: Fut
ureWarning: The default value of numeric_only in DataFrame.corr is deprecate
d. In a future version, it will default to False. Select only valid columns
or specify the value of numeric_only to silence this warning.
  sns.heatmap(df_cleaned.corr())
```

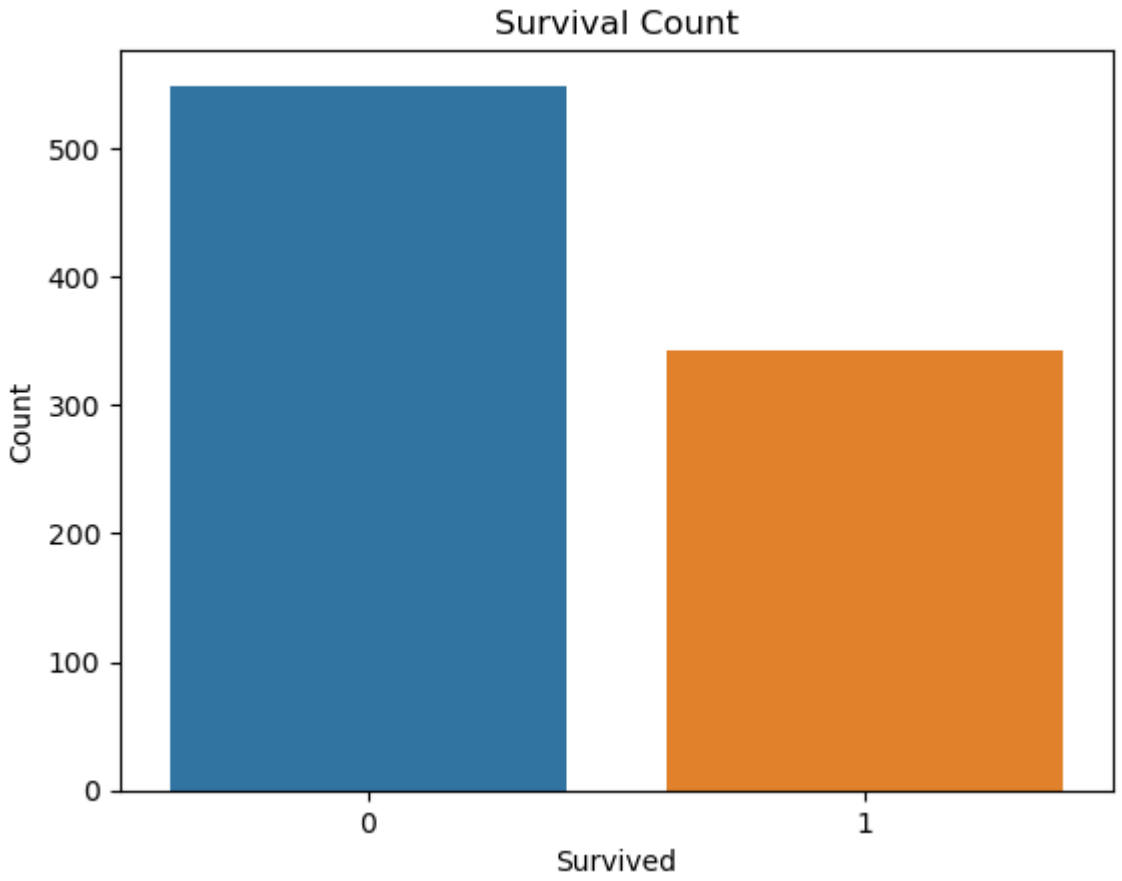
Out[84]: <Axes: >

Loading [MathJax]/extensions/Safe.js



In [85]:
```
# Visualize the distribution of the 'Survived' column (0 = Not Survived, 1 =
sns.countplot(data=df, x='Survived')
plt.title('Survival Count')
plt.xlabel('Survived')
plt.ylabel('Count')
plt.show()
```

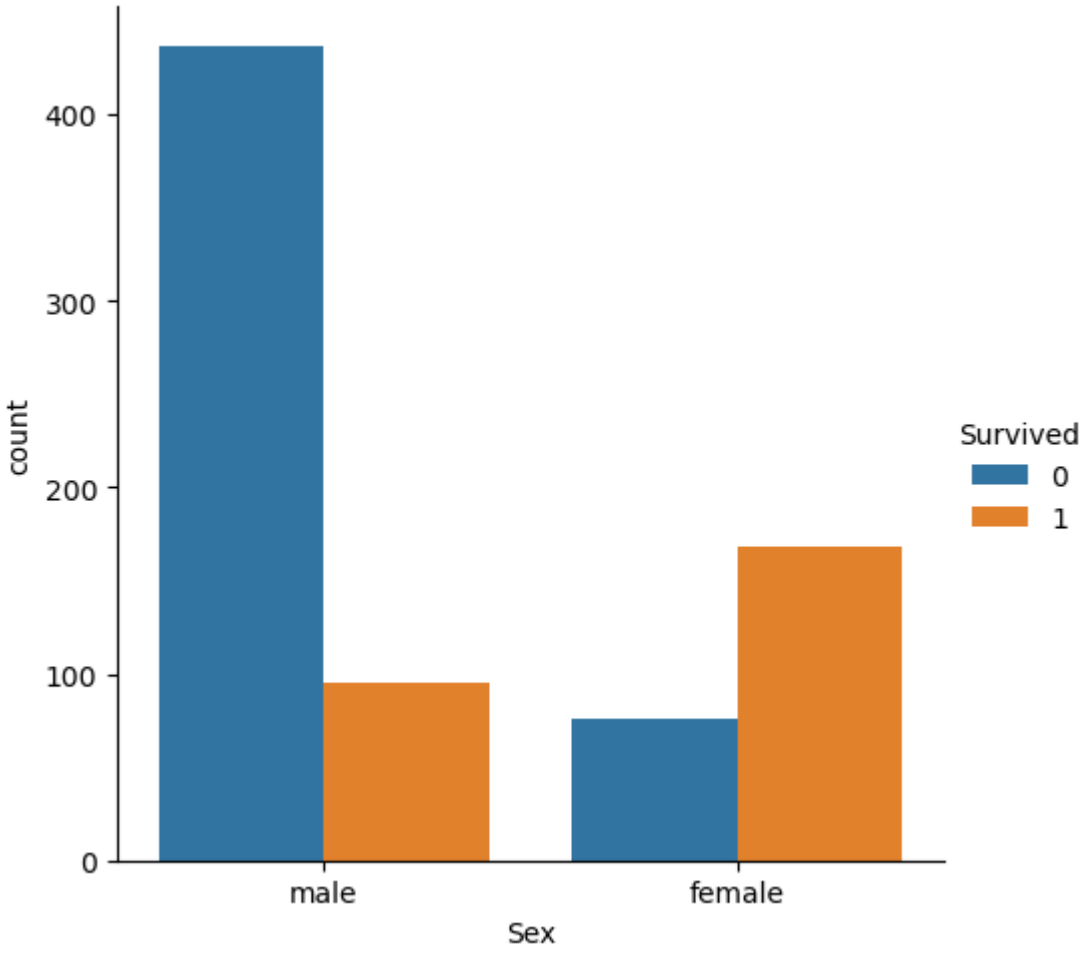
Loading [MathJax]/extensions/Safe.js



In [86]:
```
sns.catplot(x ="Sex", hue ="Survived",kind ="count", data = df_cleaned)
```

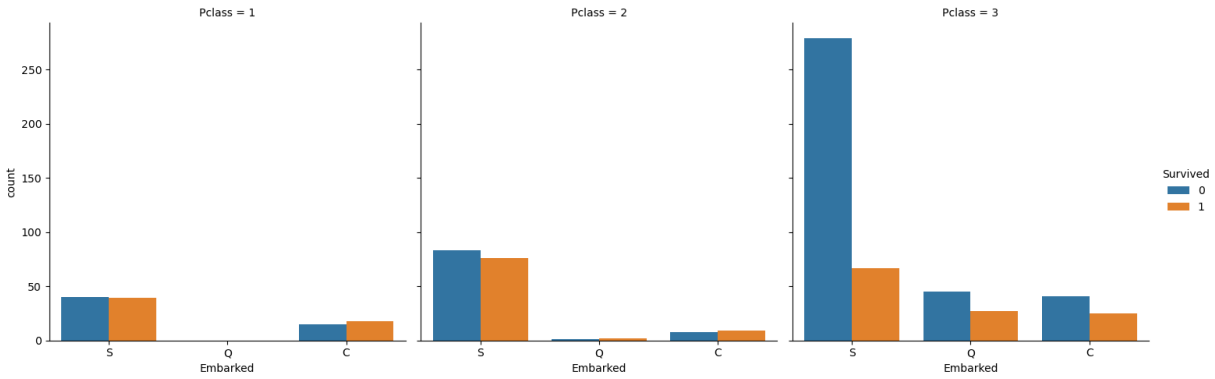
Out[86]: <seaborn.axisgrid.FacetGrid at 0x213c0809c10>

Loading [MathJax]/extensions/Safe.js



In [87]:
```
# Countplot
sns.catplot(x ='Embarked', hue ='Survived',
kind ='count', col ='Pclass', data = df_cleaned)
```

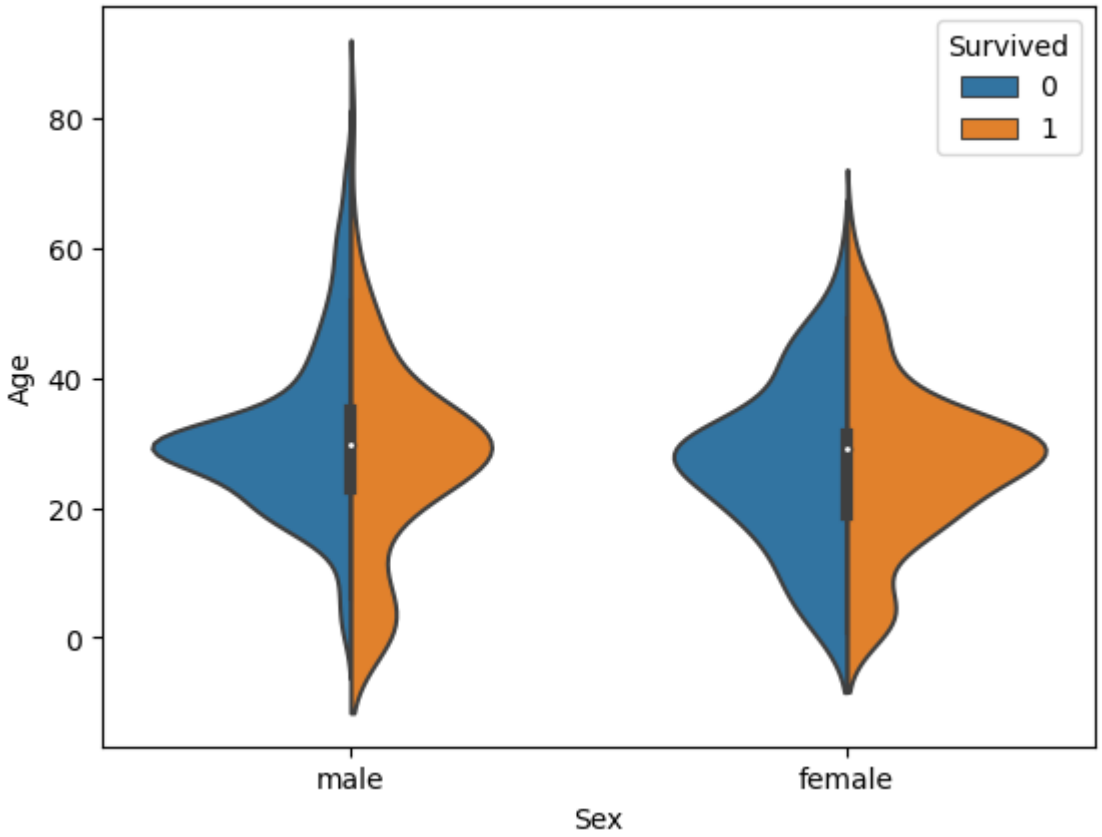
Out[87]: <seaborn.axisgrid.FacetGrid at 0x213c08a2ed0>



In [88]:
```
sns.violinplot(x ="Sex", y ="Age", hue ="Survived",data = df_cleaned, split
```

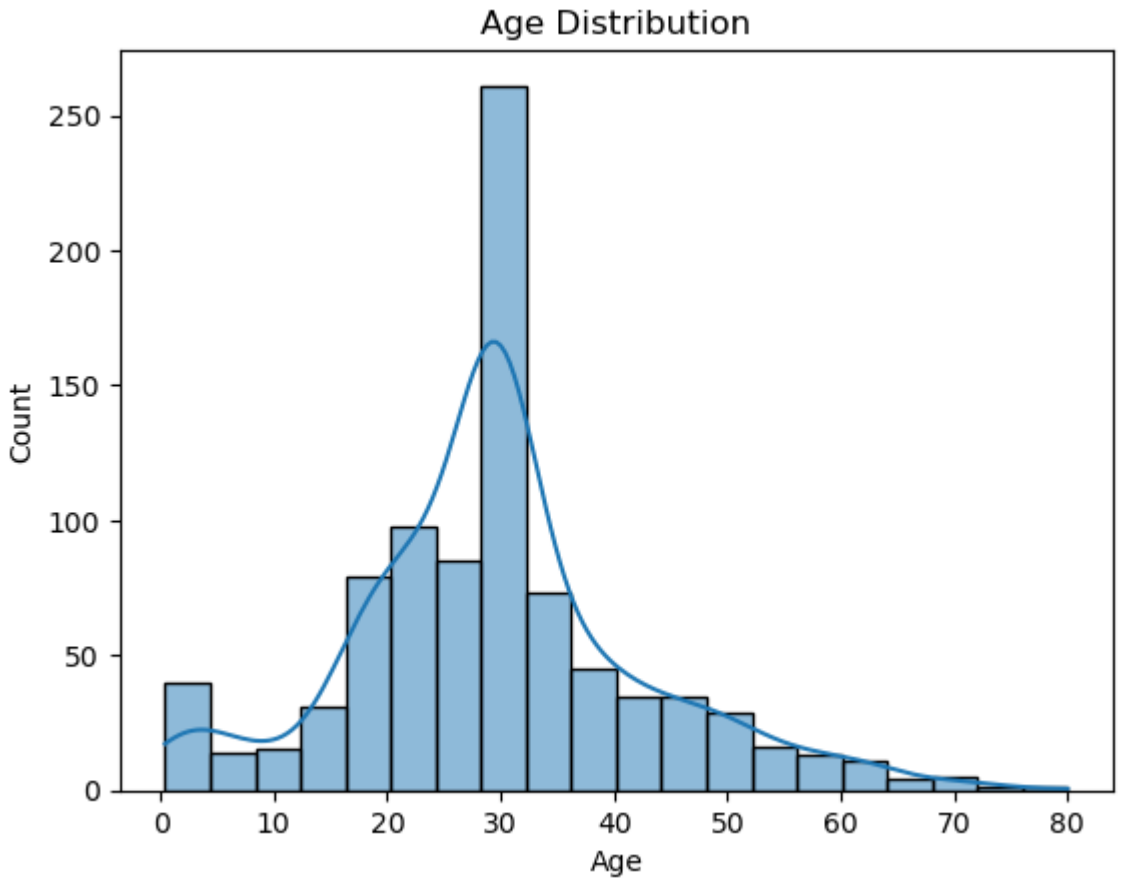
Out[88]: <Axes: xlabel='Sex', ylabel='Age'>

Loading [MathJax]/extensions/Safe.js



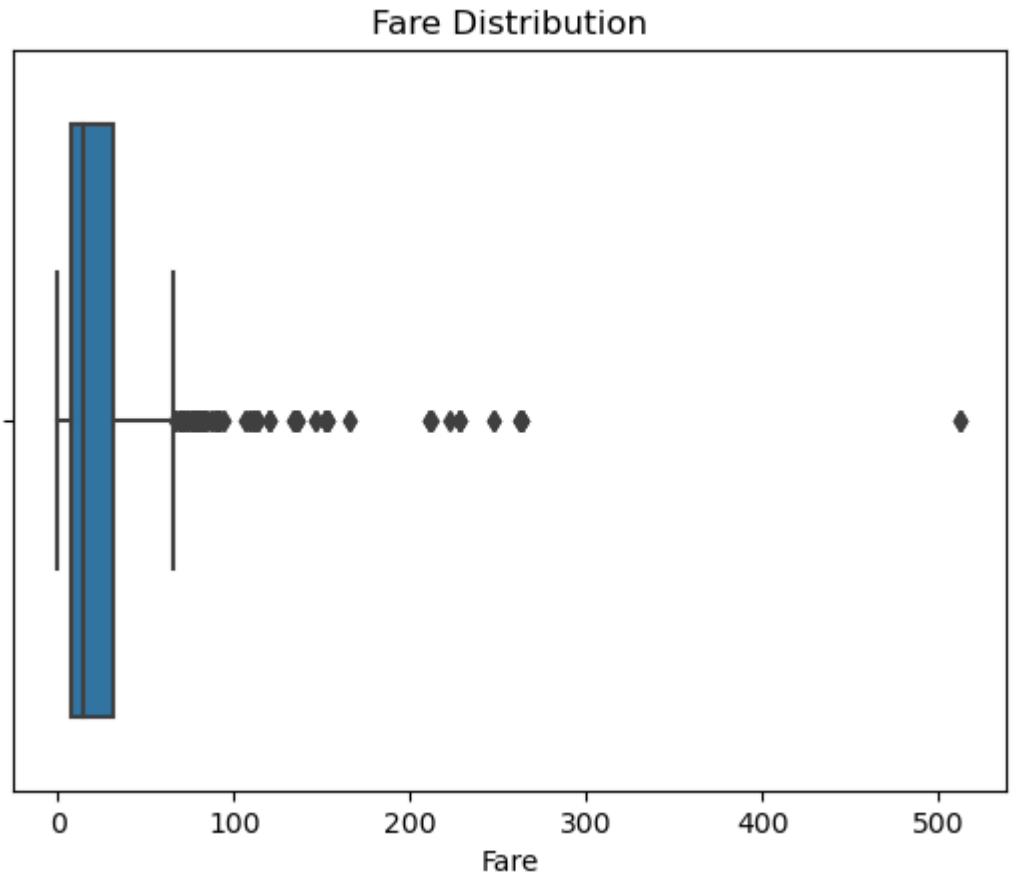
In [89]:
```
#Visualize the distribution of the 'Age' column
sns.histplot(data=df, x='Age', bins=20, kde=True)
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```

Loading [MathJax]/extensions/Safe.js



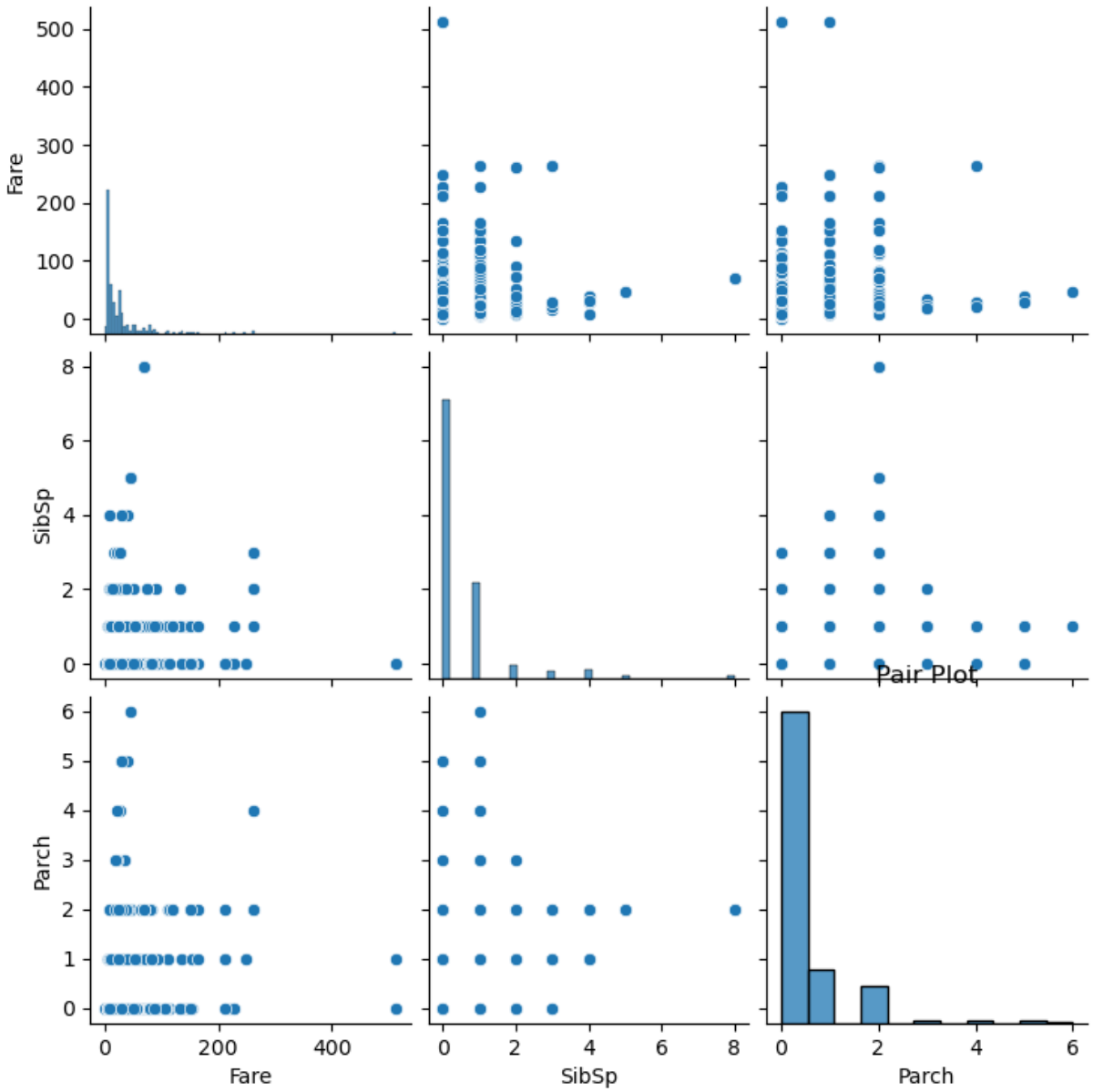
In [90]:
```
#Visualize the distribution of the 'Fare' column and detect outliers we will
sns.boxplot(data=df, x='Fare')
plt.title('Fare Distribution')
plt.xlabel('Fare')
plt.show()
```

Loading [MathJax]/extensions/Safe.js



In [91]:
```
#Pair plot for selected numerical columns
sns.pairplot(data=df[['Fare', 'SibSp', 'Parch']])
plt.title('Pair Plot')
plt.show()
```

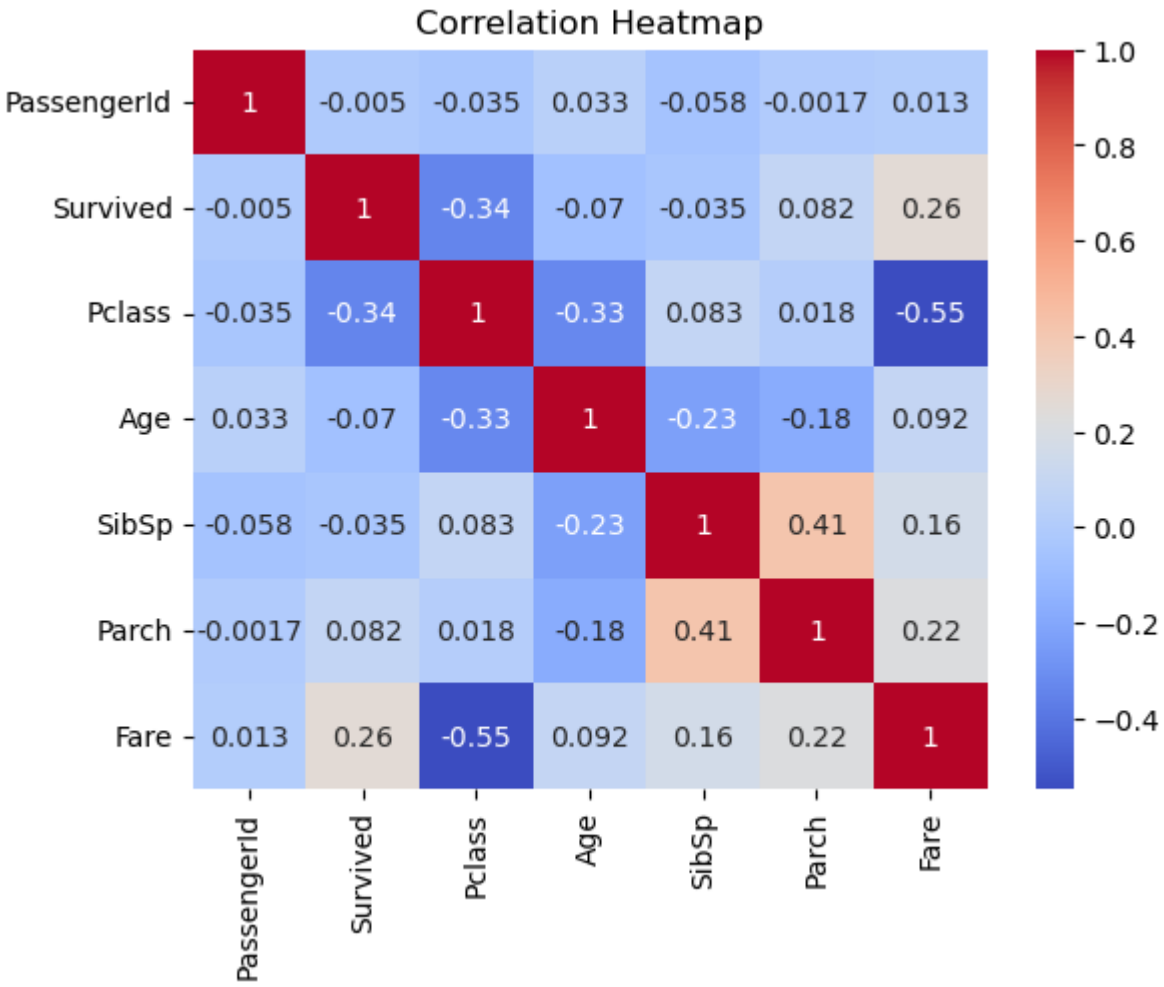
Loading [MathJax]/extensions/Safe.js



In [92]:
```
corr_matrix = df.corr()
sns.heatmap(corr_matrix, annot=True,cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

```
C:\Users\NAGA BHAVANI\AppData\Local\Temp\ipykernel_1968\554220597.py:1: Futu
reWarning: The default value of numeric_only in DataFrame.corr is deprecate
d. 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 = df.corr()
```

Loading [MathJax]/extensions/Safe.js



5. DETECT AND HANDLE OUTLIERS

In [93]:
```
z_scores = np.abs(stats.zscore(df['Age']))
max_threshold=3
outliers = df['Age'][z_scores > max_threshold]

# Print and visualize the outliers
print("Outliers detected using Z-Score:")
print(outliers)
```

```
Outliers detected using Z-Score:
96     71.0
116    70.5
493    71.0
630    80.0
672    70.0
745    70.0
851    74.0
Name: Age, dtype: float64
```

In [94]:
```
z_scores = np.abs(stats.zscore(df['Fare']))
max_threshold=3
outliers = df['Fare'][z_scores > max_threshold]
```

Loading [MathJax]/extensions/Safe.js

```
# Print and visualize the outliers
print("Outliers detected using Z-Score:")
print(outliers)
```

```
Outliers detected using Z-Score:
27     263.0000
88     263.0000
118    247.5208
258    512.3292
299    247.5208
311    262.3750
341    263.0000
377    211.5000
380    227.5250
438    263.0000
527    221.7792
557    227.5250
679    512.3292
689    211.3375
700    227.5250
716    227.5250
730    211.3375
737    512.3292
742    262.3750
779    211.3375
Name: Fare, dtype: float64
```

In [95]:
```
column_name = 'Fare'

# Calculate the first quartile (Q1) and third quartile (Q3)
Q1 = df[column_name].quantile(0.25)
Q3 = df[column_name].quantile(0.75)

# Calculate the IQR
IQR = Q3 - Q1

# Define the lower and upper bounds for outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Filter rows with values outside the IQR bounds
df_cleaned = df[(df[column_name] > lower_bound) & (df[column_name] <upper_bo

# Display the original and cleaned DataFrame sizes
print(f"Original DataFrame size: {df.shape}")
print(f"Cleaned DataFrame size: {df_cleaned.shape}")
df_cleaned
```

```
Original DataFrame size: (891, 9)
Cleaned DataFrame size: (775, 9)
```

Loading [MathJax]/extensions/Safe.js

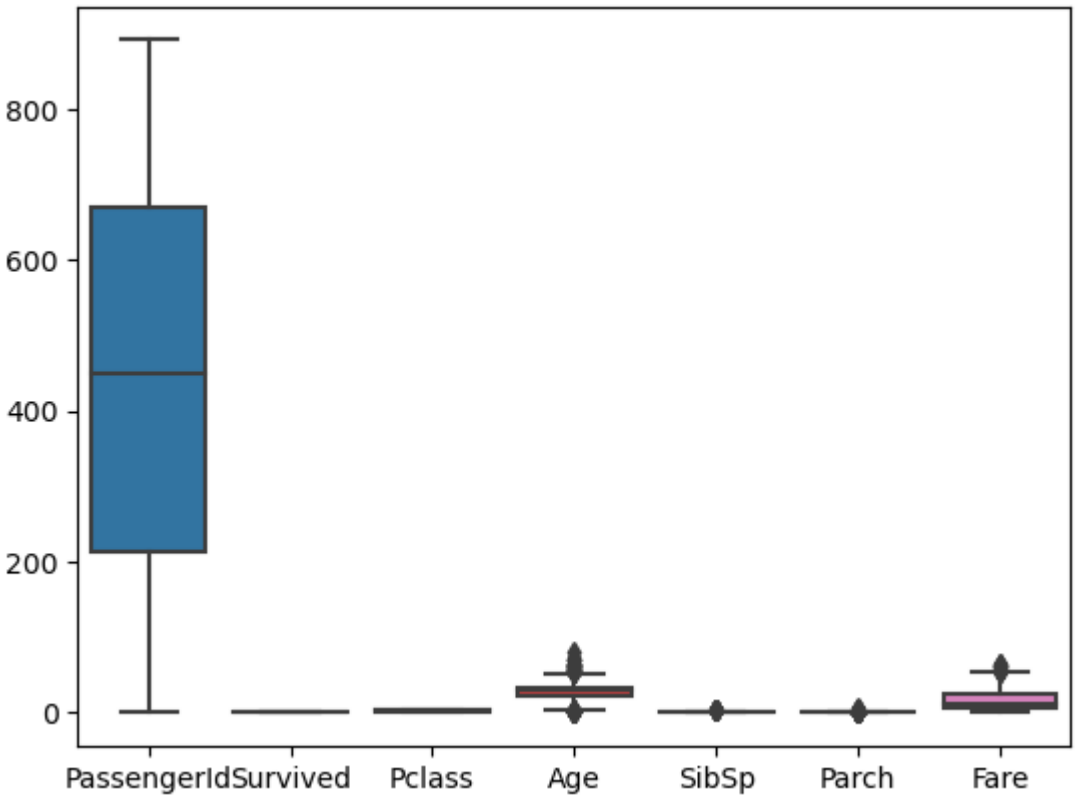
Out[95]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare
0	1	0	3	male	22.000000	1	0	7.2500
2	3	1	3	female	26.000000	0	0	7.9250
3	4	1	1	female	35.000000	1	0	53.1000
4	5	0	3	male	35.000000	0	0	8.0500
5	6	0	3	male	29.699118	0	0	8.4583
...	...	...	...	...	...	...	...	...
886	887	0	2	male	27.000000	0	0	13.0000
887	888	1	1	female	19.000000	0	0	30.0000
888	889	0	3	female	29.699118	1	2	23.4500
889	890	1	1	male	26.000000	0	0	30.0000
890	891	0	3	male	32.000000	0	0	7.7500

775 rows × 9 columns

In [96]:
```
sns.boxplot(df_cleaned)
```

Out[96]: <Axes: >



6. SPLIT DEPENDENT AND INDEPENDENT DATA

Loading [MathJax]/extensions/Safe.js

In [97]:
```
df=df_cleaned
```

In [98]:
```
x=df.drop('Survived', axis=1)
y=df['Survived']
```

In [99]:
```
x.head()
```

Out[99]:

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	male	22.000000	1	0	7.2500	S
2	3	3	female	26.000000	0	0	7.9250	S
3	4	1	female	35.000000	1	0	53.1000	S
4	5	3	male	35.000000	0	0	8.0500	S
5	6	3	male	29.699118	0	0	8.4583	Q

In [100…
```
y.head()
```

Out[100…
```
0    0
2    1
3    1
4    0
5    0
Name: Survived, dtype: int64
```

7. PERFORM ENCODING

In [101…
```
en = LabelEncoder()
x['Sex'] = en.fit_transform(x['Sex'])
```

In [102…
```
x.head()
```

Out[102…

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	3	1	22.000000	1	0	7.2500	S
2	3	3	0	26.000000	0	0	7.9250	S
3	4	1	0	35.000000	1	0	53.1000	S
4	5	3	1	35.000000	0	0	8.0500	S
5	6	3	1	29.699118	0	0	8.4583	Q

In [103…
```
x = pd.get_dummies(x,columns=['Embarked'])
```

In [104…
```
x.head()
```

Loading [MathJax]/extensions/Safe.js

Out[104…

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_C	E
0	1	3	1	22.000000	1	0	7.2500	0	
2	3	3	0	26.000000	0	0	7.9250	0	
3	4	1	0	35.000000	1	0	53.1000	0	
4	5	3	1	35.000000	0	0	8.0500	0	
5	6	3	1	29.699118	0	0	8.4583	0	

8. FEATURE SCALING

In [105…
```
scale = StandardScaler()
x[['Age', 'Fare']] = scale.fit_transform(x[['Age', 'Fare']])
```

In [106…
```
x.head()
```

Out[106…

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_C
0	1	3	1	-0.556219	1	0	-0.779117	0
2	3	3	0	-0.243027	0	0	-0.729373	0
3	4	1	0	0.461654	1	0	2.599828	0
4	5	3	1	0.461654	0	0	-0.720161	0
5	6	3	1	0.046606	0	0	-0.690071	0

9. SPLITTING THE DATA INTO TRAIN AND TEST

In [107…
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, ran
```

In [108…
```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(620, 10)
(155, 10)
(620,)
(155,)
```

In [109…
```
x_train
```

Loading [MathJax]/extensions/Safe.js

Out[109…

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_
147	148	3	0	-1.574091	2	2	1.219877	
868	869	3	1	0.046606	0	0	-0.613302	
60	61	3	1	-0.556219	0	0	-0.780650	
468	469	3	1	0.046606	0	0	-0.744112	
777	778	3	0	-1.887282	0	0	-0.394057	
...	...	...	...	...	...	...	...	.
79	80	3	0	0.070164	0	0	-0.394057	
116	117	3	1	3.241228	0	0	-0.742269	
308	309	2	1	0.070164	1	0	0.455285	
502	503	3	0	0.046606	0	0	-0.751172	
112	113	3	1	-0.556219	0	0	-0.720161	

620 rows × 10 columns

In [110…
```
y_train
```

Out[110…
```
147    0
868    0
60     0
468    0
777    1
      ..
79     1
116    0
308    0
502    0
112    0
Name: Survived, Length: 620, dtype: int64
```

In [111…
```
x_test
```

Loading [MathJax]/extensions/Safe.js

Out[111…

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_
493	494	1	1	3.280377	0	0	2.334834	
821	822	3	1	-0.164729	0	0	-0.675022	
381	382	3	0	-2.200474	0	2	-0.153316	
881	882	3	1	0.305058	0	0	-0.731524	
420	421	3	1	0.046606	0	0	-0.731524	
...	...	...	...	...	...	...	...	.
164	165	3	1	-2.200474	4	1	1.611385	
384	385	3	1	0.046606	0	0	-0.731524	
535	536	2	0	-1.730687	0	2	0.621100	
339	340	1	1	1.244632	0	0	1.302785	
657	658	3	0	0.226760	1	1	-0.171128	

155 rows × 10 columns

In [112…
```
y_test
```

Out[112…
```
493    0
821    1
381    1
881    0
420    0
      ..
164    0
384    0
535    1
339    0
657    0
Name: Survived, Length: 155, dtype: int64
```

In [ ]:

In [ ]:

Loading [MathJax]/extensions/Safe.js