• 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-    Encoding

o    Feature Scaling.

o    Splitting Data into Train and Test.
```

1.Import the Libraries.

In [1]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2.Importing the dataset.

In [2]:

```python
df=pd.read_csv("Titanic-Dataset.csv")
```

In [3]:
```
df
```

Out[3]:

	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
...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500

891 rows × 12 columns

In [4]:
```
df.head()
```

Out[4]:

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

In [5]:
```
df.tail()
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN

In [6]:
```
df.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          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 [7]:
```
df.describe()
```

Out[7]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [8]:
```
df.shape
```

Out[8]:
```
(891, 12)
```

In [9]:

```
df.corr()
```

Out[9]:

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

In [10]:

```
df.corr().Fare.sort_values(ascending=False)
```

Out[10]:

```
Fare           1.000000
Survived       0.257307
Parch          0.216225
SibSp          0.159651
Age            0.096067
PassengerId    0.012658
Pclass        -0.549500
Name: Fare, dtype: float64
```

In [11]:

```
df.Survived.value_counts()
```

Out[11]:

```
0    549
1    342
Name: Survived, dtype: int64
```

In [12]:

```
df.Sex.value_counts()
```

Out[12]:

```
male      577
female    314
Name: Sex, dtype: int64
```

In [13]:

```
df.Embarked.value_counts()
```

Out[13]:

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

3.Checking for Null Values.

In [14]:

```
df.isnull().any()
```

Out[14]:

```
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 [15]:

```
df.isnull().sum()
```

Out[15]:

```
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 [16]:

```
df["Age"].mean()
```

Out[16]:

```
29.69911764705882
```

In [17]:

```
df['Age'].fillna(df['Age'].mean(),inplace=True)
```

In [18]:

```
df.isnull().sum()
```

Out[18]:

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

```
In [19]: df["Embarked"].mode()
```

```
Out[19]: 0    S
         Name: Embarked, dtype: object
```

```
In [20]: df['Embarked'].fillna(df['Embarked'].mode()[0],inplace=True)
```

```
In [21]: df.isnull().sum()
```

```
Out[21]: PassengerId      0
         Survived         0
         Pclass           0
         Name             0
         Sex              0
         Age              0
         SibSp            0
         Parch            0
         Ticket           0
         Fare             0
         Cabin          687
         Embarked         0
         dtype: int64
```

```
In [22]: df.drop(["Cabin"],axis=1,inplace=True)
```

```
In [23]: df.drop(["Ticket"],axis=1,inplace=True)
```

```
In [24]: df.drop(["Name"],axis=1,inplace=True)
```

```
In [25]: df.isnull().sum()
```

```
Out[25]: PassengerId    0
         Survived       0
         Pclass         0
         Sex            0
         Age            0
         SibSp          0
         Parch          0
         Fare           0
         Embarked       0
         dtype: int64
```

```
In [26]: df.Embarked.nunique()
```

```
Out[26]: 3
```

```
In [27]: df.Embarked.unique()
```

```
Out[27]: array(['S', 'C', 'Q'], dtype=object)
```

In [28]:

```
df.Embarked.value_counts()
```

Out[28]:

```
S    646
C    168
Q     77
Name: Embarked, dtype: int64
```

In [29]:

```
df.head()
```

Out[29]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	male	22.0	1	0	7.2500	S
1	2	1	1	female	38.0	1	0	71.2833	C
2	3	1	3	female	26.0	0	0	7.9250	S
3	4	1	1	female	35.0	1	0	53.1000	S
4	5	0	3	male	35.0	0	0	8.0500	S

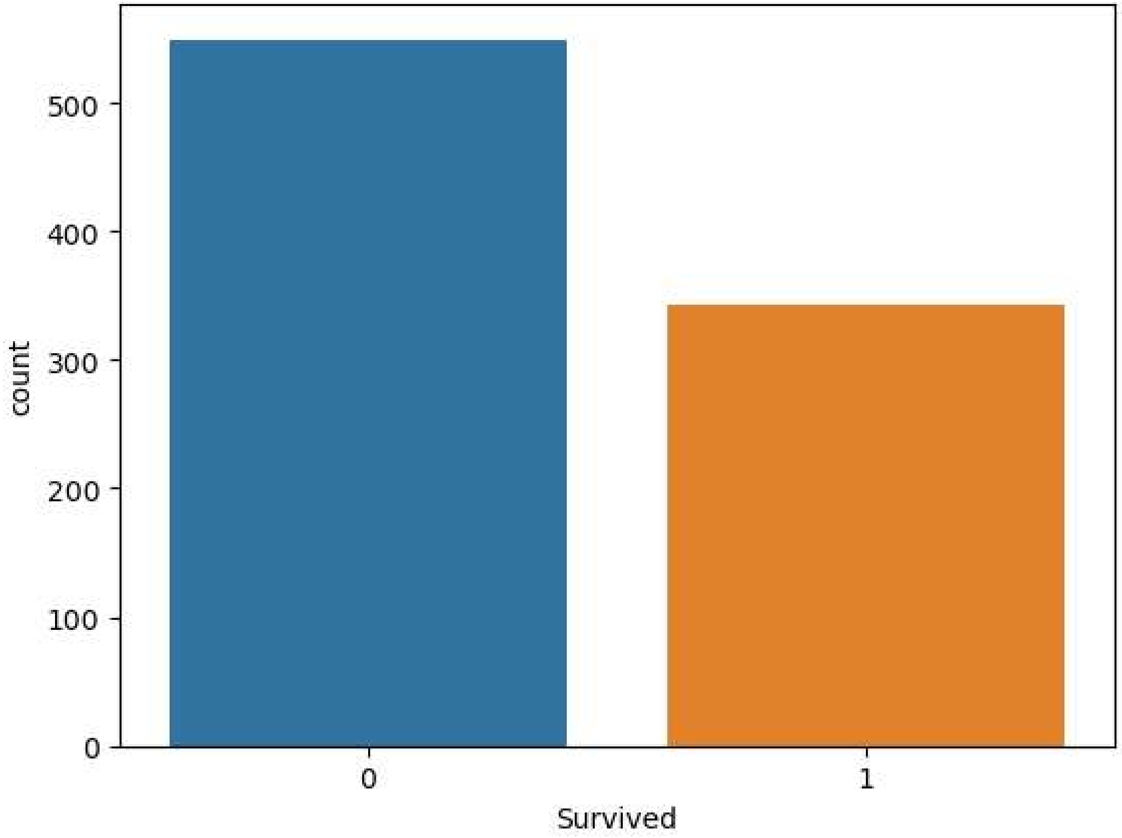
4.Data Visualization.

In [30]:

```
sns.countplot(x="Survived",data=df)
```

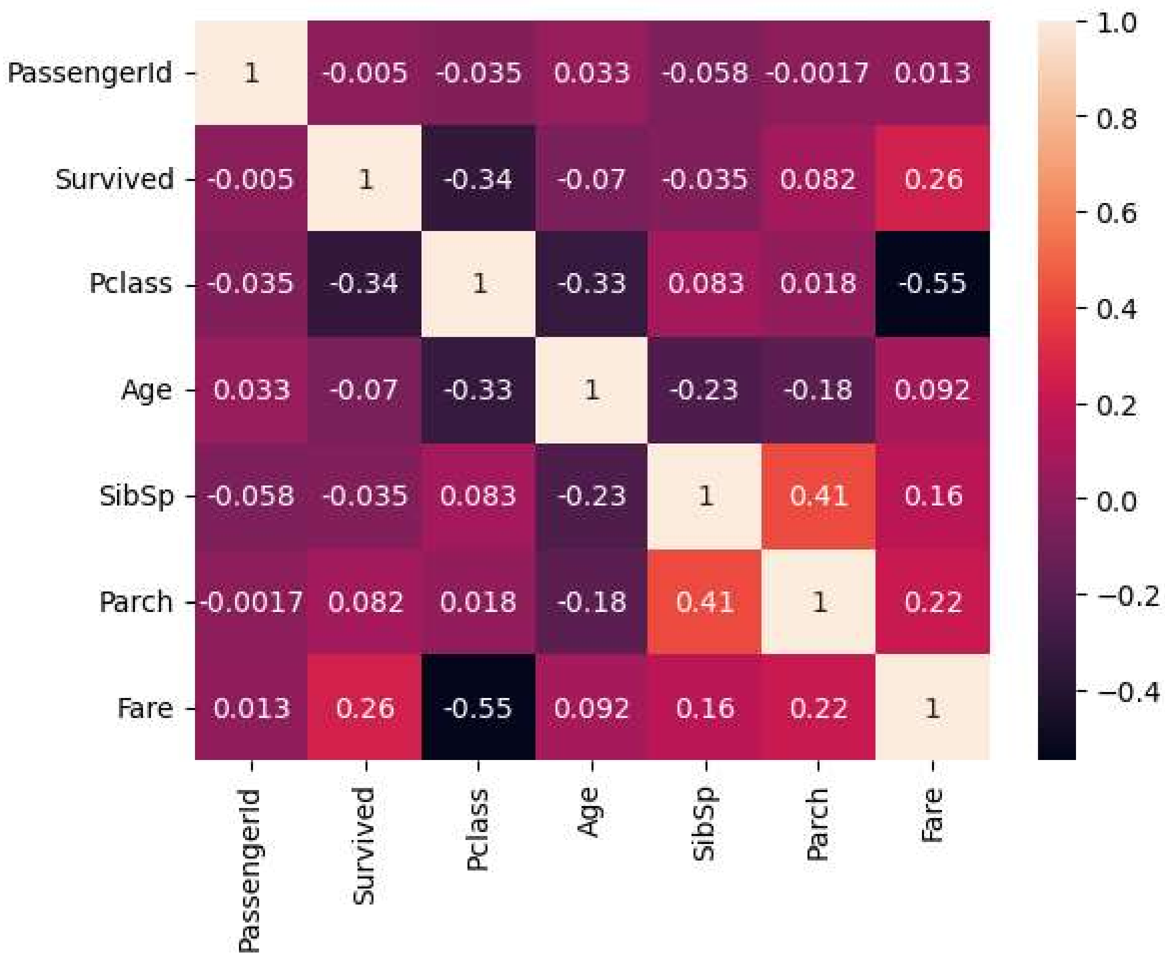
Out[30]:

```
<AxesSubplot:xlabel='Survived', ylabel='count'>
```



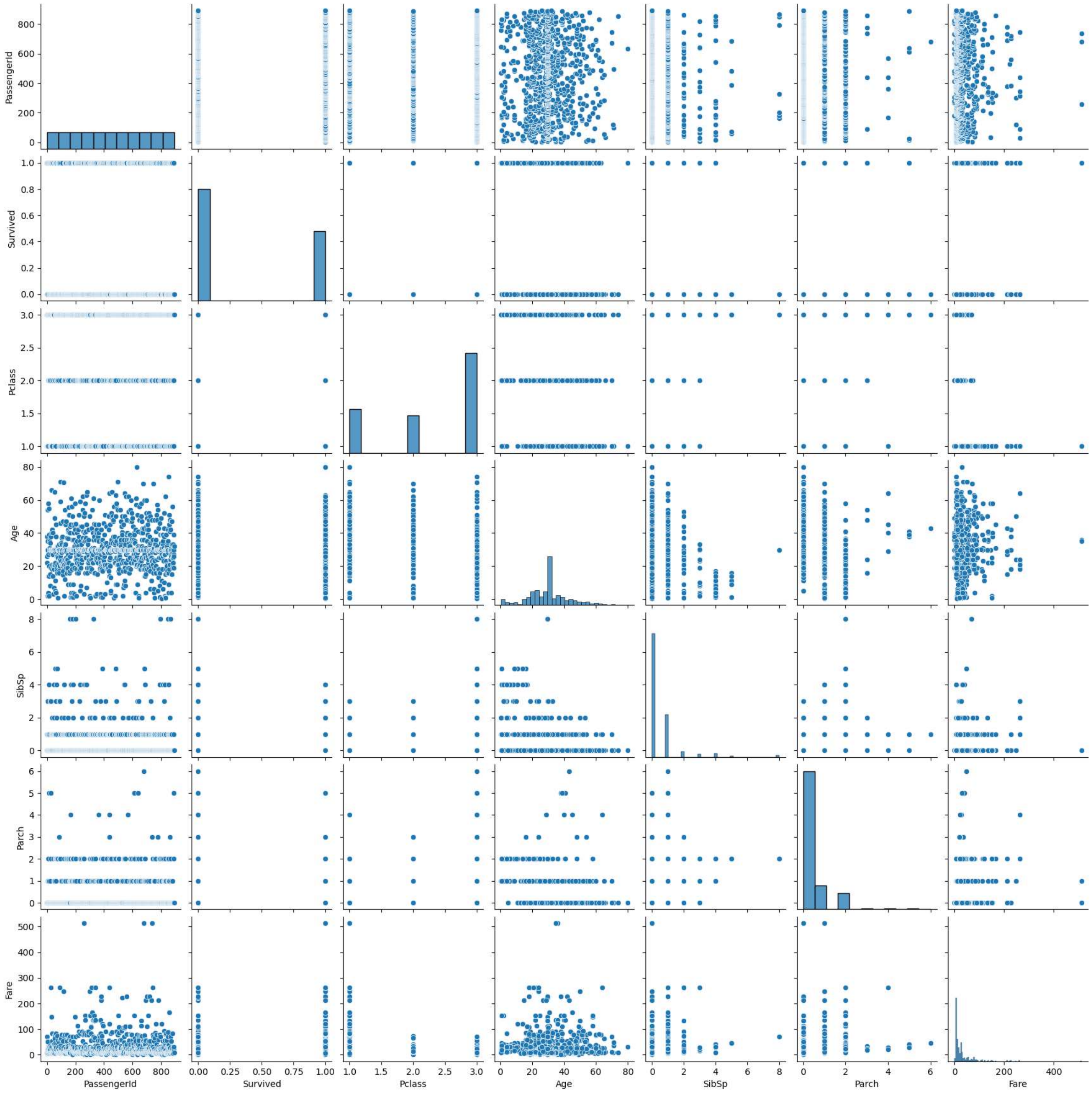
```
In [31]: sns.heatmap(df.corr(),annot=True)
```

```
Out[31]: <AxesSubplot:>
```



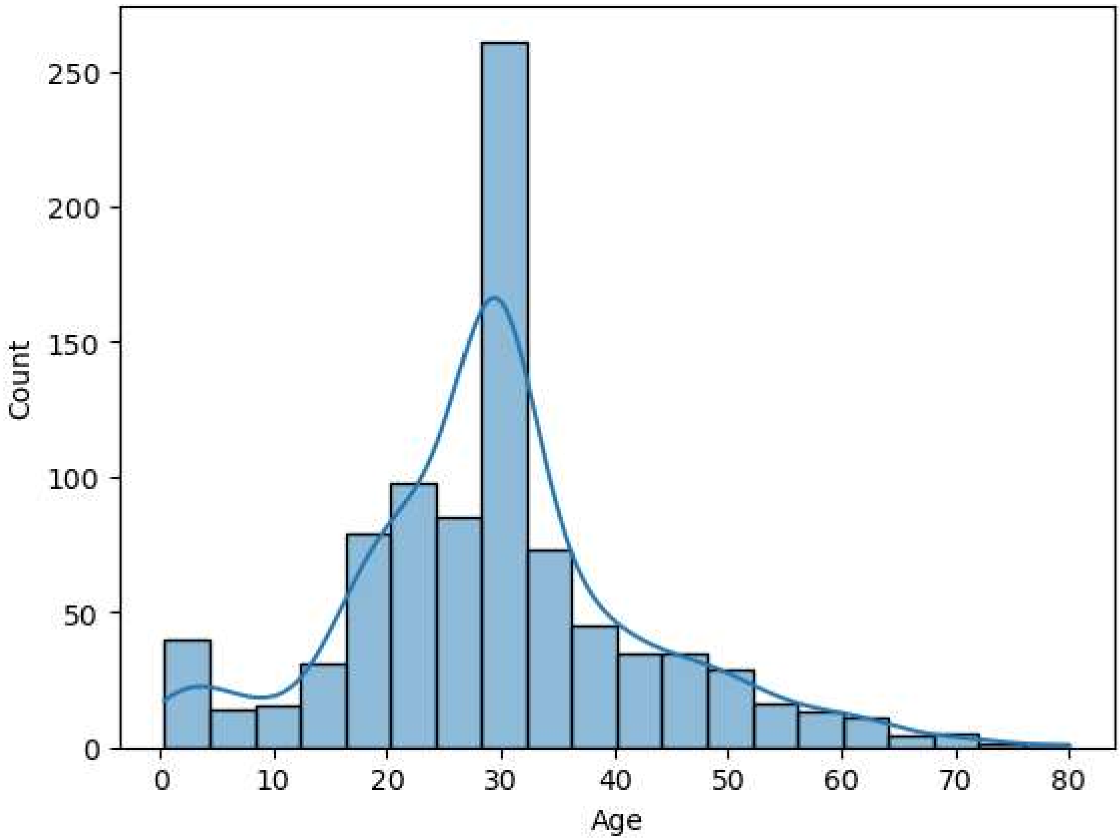
In [32]:
```
sns.pairplot(df)
```

Out[32]:
```
<seaborn.axisgrid.PairGrid at 0x1d0398ef220>
```



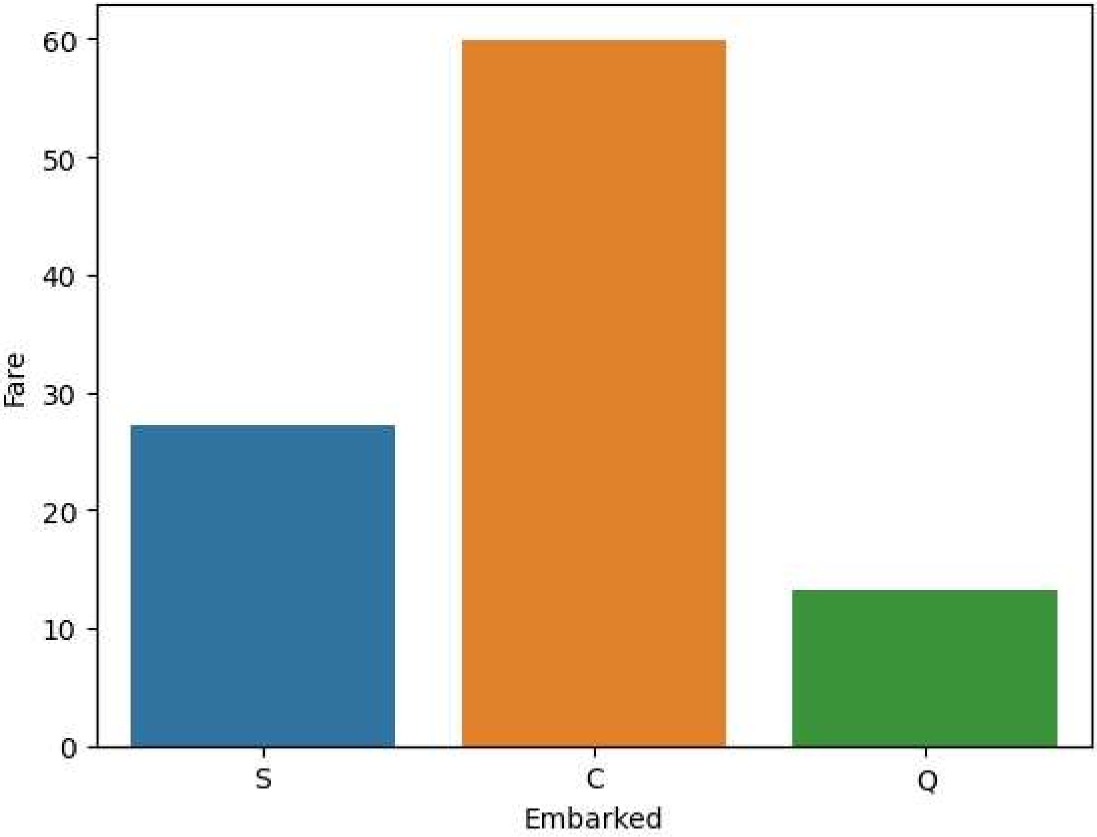
```
In [34]: sns.histplot(data=df,x="Age",bins=20,kde=True)
```

```
Out[34]: <AxesSubplot:xlabel='Age', ylabel='Count'>
```



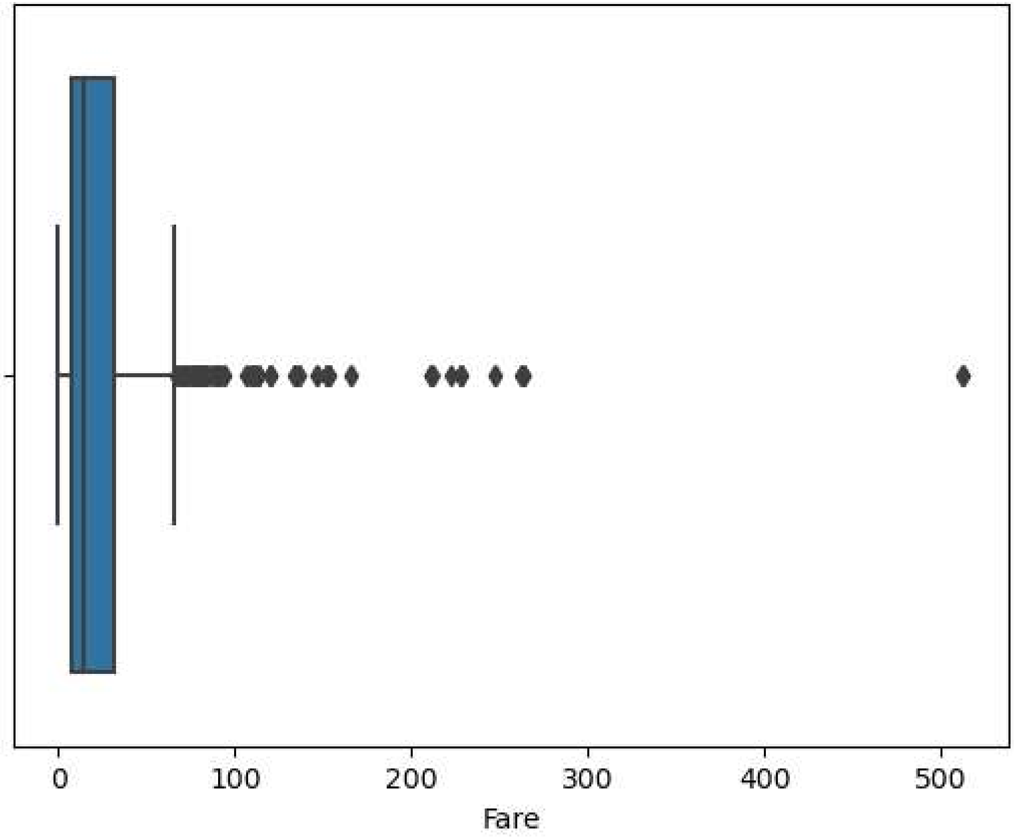
In [35]:
```
sns.barplot(x=df["Embarked"],y=df["Fare"],ci=None)
```

Out[35]:
```
<AxesSubplot:xlabel='Embarked', ylabel='Fare'>
```



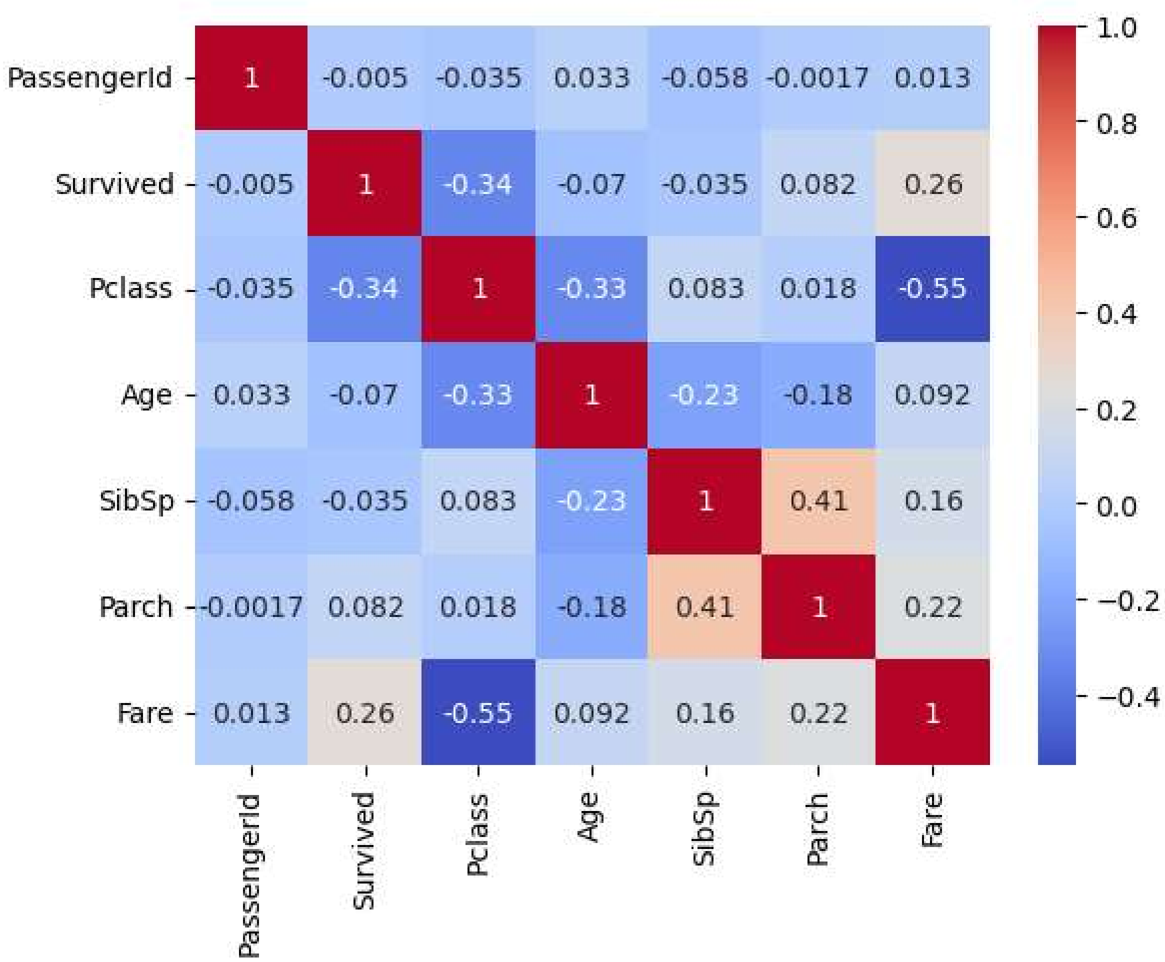
In [36]:
```
sns.boxplot(x="Fare",data=df)
```

Out[36]:
```
<AxesSubplot:xlabel='Fare'>
```



In [37]:
```
sns.heatmap(df.corr(),annot=True,cmap='coolwarm')
```

Out[37]:
```
<AxesSubplot:>
```



5.Outlier Detection

In [38]:
```
df.head()
```

Out[38]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	male	22.0	1	0	7.2500	S
1	2	1	1	female	38.0	1	0	71.2833	C
2	3	1	3	female	26.0	0	0	7.9250	S
3	4	1	1	female	35.0	1	0	53.1000	S
4	5	0	3	male	35.0	0	0	8.0500	S

In [45]:
```
from scipy import stats
z_scores=np.abs(stats.zscore(df["Age"]))
```

In [46]:
```
outliers=df["Age"][z_scores>3]
```

In [47]:
```
outliers
```

Out[47]:
```
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 [49]:
```
z_score=np.abs(stats.zscore(df["Fare"]))
outlier=df["Fare"][z_score>3]
```

In [50]:
```
outlier
```

Out[50]:
```
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 [51]:

```
Q1 = df["Fare"].quantile(0.25)
Q3 = df["Fare"].quantile(0.75)

IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

df_cleaned = df[(df["Fare"] > lower_bound) & (df["Fare"] <upper_bound)]

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)
```

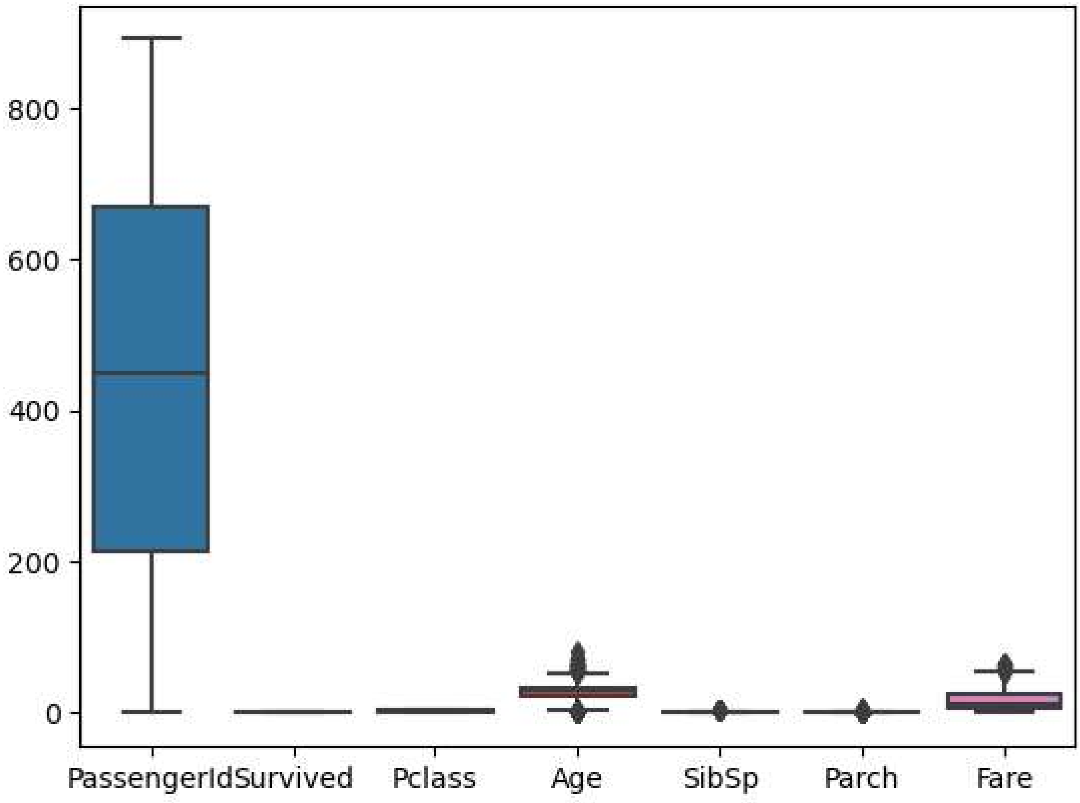
Out[51]:

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

775 rows × 9 columns

In [109]:
```
sns.boxplot(data=df_cleaned)
```

Out[109]:
```
<AxesSubplot:>
```



6.Splitting Dependent and Independent variables

In [58]:
```
df.head()
```

Out[58]:

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

In [86]:
```
#Independent variable should be a 2D array
x=df.drop(columns=["Survived"],axis=1)
```

In [87]:
```
x.head()
```

Out[87]:

	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 [88]:
```
type(x)
```

Out[88]:
```
pandas.core.frame.DataFrame
```

In [89]:
```
x.shape
```

Out[89]:
```
(775, 8)
```

In [90]:
```
y=df["Survived"]
```

In [91]:
```
y.head()
```

Out[91]:
```
0    0
2    1
3    1
4    0
5    0
Name: Survived, dtype: int64
```

In [92]:
```
type(y)
```

Out[92]:
```
pandas.core.series.Series
```

In [93]:
```
y.shape
```

Out[93]:
```
(775,)
```

7.Encoding

In [94]:
```
x.head()
```

Out[94]:

	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 [95]:
```
from sklearn.preprocessing import LabelEncoder
```

In [96]:
```
le=LabelEncoder()
```

In [97]:
```
x["Embarked"]=le.fit_transform(x["Embarked"])
```

In [98]:
```
x.head()
```

Out[98]:

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

In [99]:
```
print(le.classes_)
```

```
['C' 'Q' 'S']
```

In [100]:
```
x["Sex"]=le.fit_transform(x["Sex"])
```

In [101]:
```
x.head()
```

Out[101]:

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

8.Feature Scaling.

In [102]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [103]:
```
x[['Age', 'Fare']] = sc.fit_transform(x[['Age', 'Fare']])
```

In [104]:
```
x.head()
```

Out[104]:

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

9.Splitting Data into Train and Test.

In [105]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size =0.2,random_sta
```

In [106]:
```
print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(620, 8) (155, 8) (620,) (155,)
```