Name: DHARMANA GNANA SAI

Reg No: 21BCE7400

Campus: VIT-AP

Email: gnanasai.21bce7400@vitapstudent.ac.in

Branch: CSE AI and ML

Data Preprocessing.

Import the Libraries.

Importing the dataset.

Checking for Null Values.

Data Visualization.

Outlier Detection

Splitting Dependent and Independent variables

Encoding

Feature Scaling.

Splitting Data into Train and Test.

Import the Libraries

In [3]:

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

Importing the dataset

In [61]:

```
data = pd.read_csv('Titanic-Dataset.csv')
data
```

Out[61]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450
...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376

891 rows × 12 columns

In [21]:

```
data.head()
```

Out[21]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0

```
In [22]: data.describe()
```

Out[22]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329

```
In [23]: data.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 [24]:

```
data.corr()
```

```
C:\Users\chatu\AppData\Local\Temp\ipykernel_13368\2627137660.py:1: FutureWarning:
The default value of numeric_only in DataFrame.corr is deprecated. In a future ve
rsion, it will default to False. Select only valid columns or specify the value o
f numeric_only to silence this warning.
  data.corr()
```

Out[24]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fa
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.0126
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.2573
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.5495
Age	0.036847	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.0960
SibSp	-0.057527	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.1596
Parch	-0.001652	0.081629	0.018443	-0.189119	0.414838	1.000000	0.2162
Fare	0.012658	0.257307	-0.549500	0.096067	0.159651	0.216225	1.0000

In [25]:

```
data.corr().Age.sort_values(ascending=False)
```

```
C:\Users\chatu\AppData\Local\Temp\ipykernel_13368\1767978217.py:1: FutureWarning:
The default value of numeric_only in DataFrame.corr is deprecated. In a future ve
rsion, it will default to False. Select only valid columns or specify the value o
f numeric_only to silence this warning.
  data.corr().Age.sort_values(ascending=False)
```

Out[25]:

```
Age            1.000000
Fare           0.096067
PassengerId    0.036847
Survived      -0.077221
Parch         -0.189119
SibSp         -0.308247
Pclass        -0.369226
Name: Age, dtype: float64
```

Checking for Null Values

In [26]:
```
data.isnull().any()
```

Out[26]:
```
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 [27]:
```
data.isnull().sum()
```

Out[27]:
```
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 [28]:
```
data.Cabin.value_counts()
```

Out[28]:
```
B96 B98        4
G6             4
C23 C25 C27    4
C22 C26        3
F33            3
              ..
E34            1
C7             1
C54            1
E36            1
C148           1
Name: Cabin, Length: 147, dtype: int64
```

In [40]:
```
data.Ticket.nunique()
```

Out[40]:
```
681
```

In [42]:
```
data.Embarked.unique()
```

Out[42]:
```
array(['S', 'C', 'Q', nan], dtype=object)
```

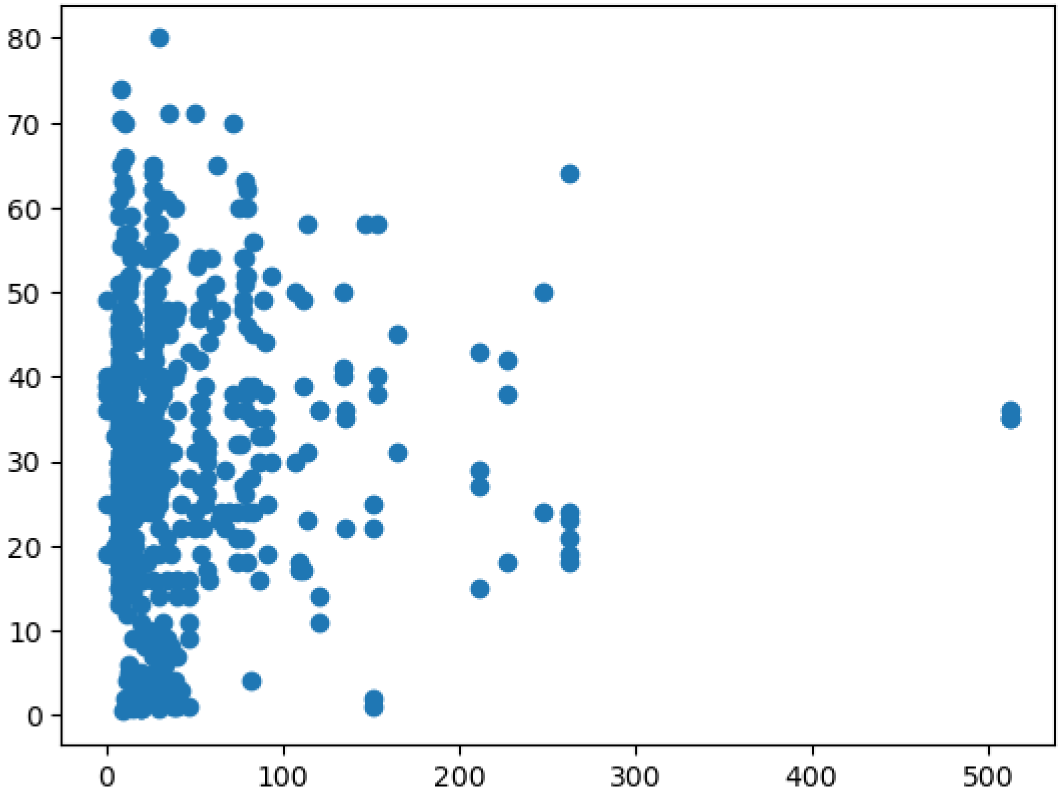
Data Visualization

In [46]:

```
plt.scatter(data["Fare"],data["Age"])
```

Out[46]:

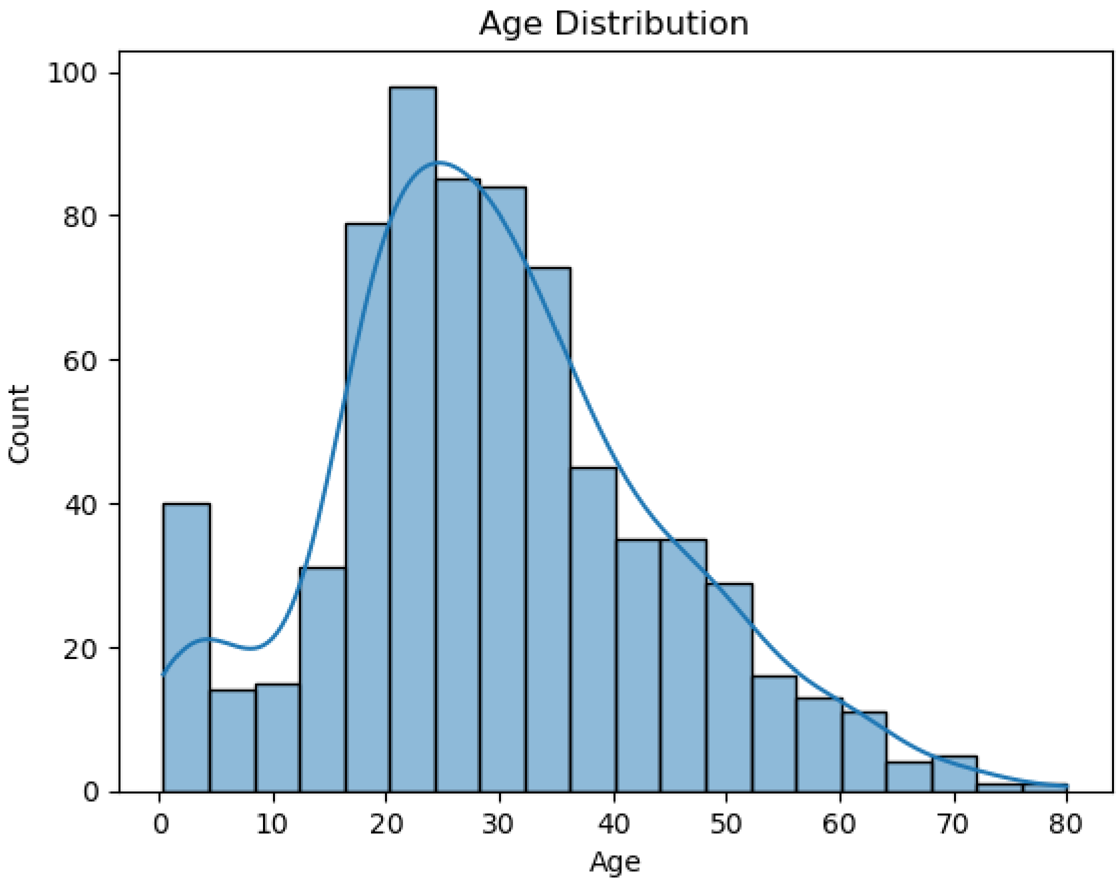
```
<matplotlib.collections.PathCollection at 0x28b11605350>
```



Inference: There are a few outliers where passengers paid significantly higher fares relative to their age, indicating potential variability in ticket pricing or unique circumstances for certain individuals.

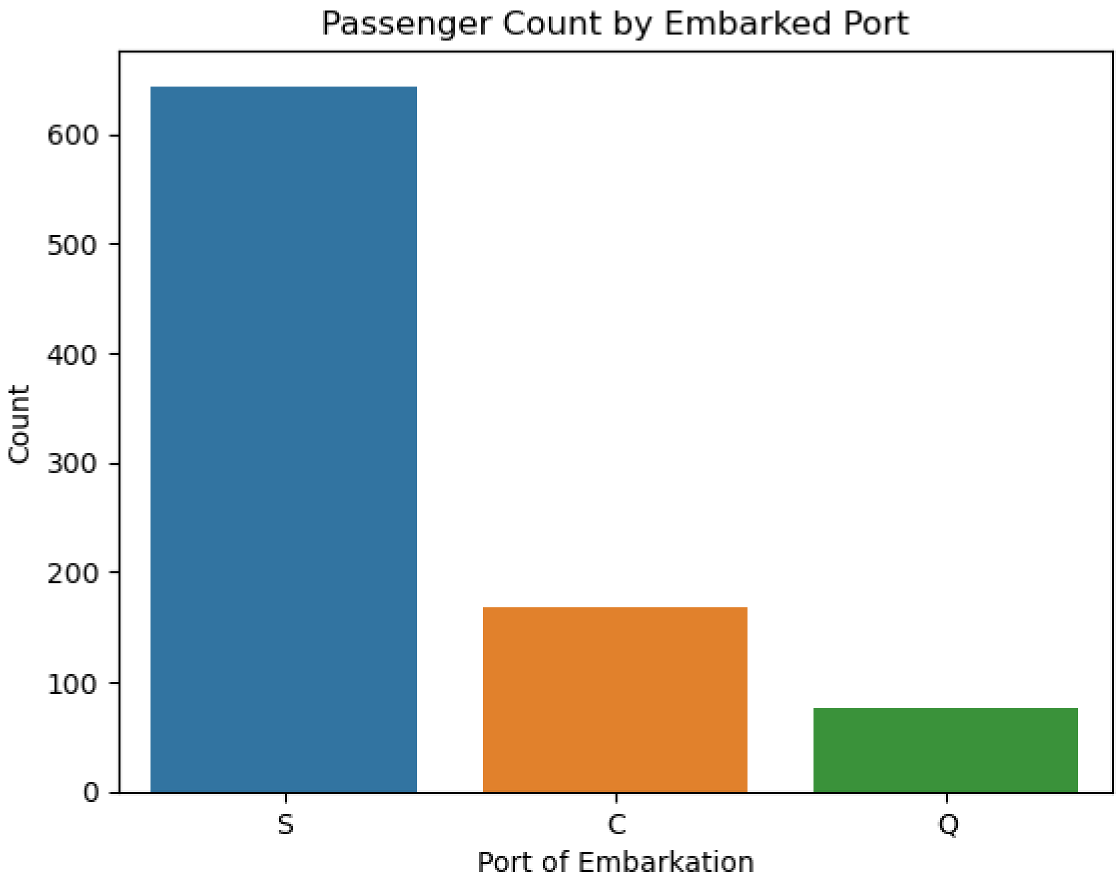
In [47]:

```
# Example: Histogram of age distribution
sns.histplot(data['Age'], bins=20, kde=True)
plt.xlabel('Age')
plt.ylabel('Count')
plt.title('Age Distribution')
plt.show()
```



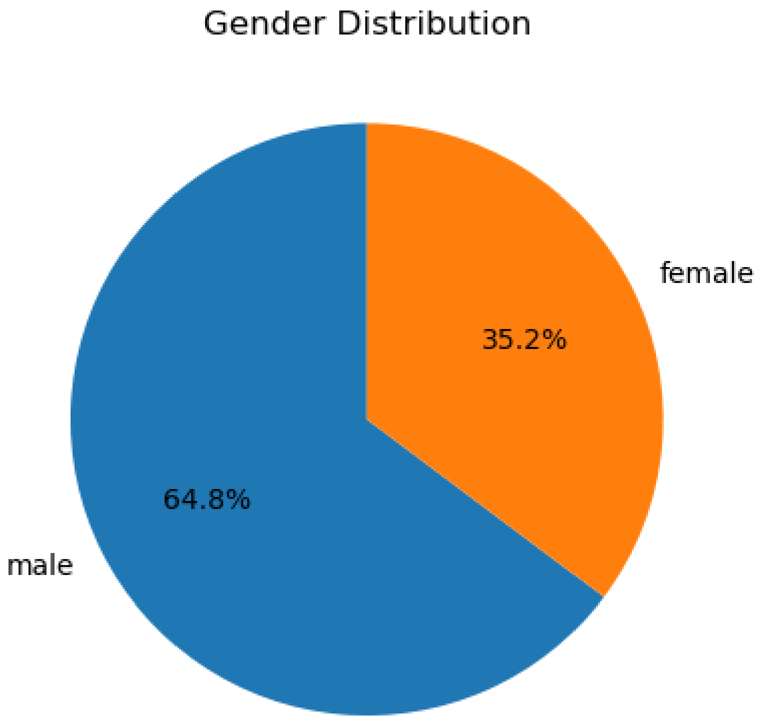
In [48]:
```
sns.countplot(data=data, x='Embarked')
plt.xlabel('Port of Embarkation')
plt.ylabel('Count')
plt.title('Passenger Count by Embarked Port')
```

Out[48]:
```
Text(0.5, 1.0, 'Passenger Count by Embarked Port')
```



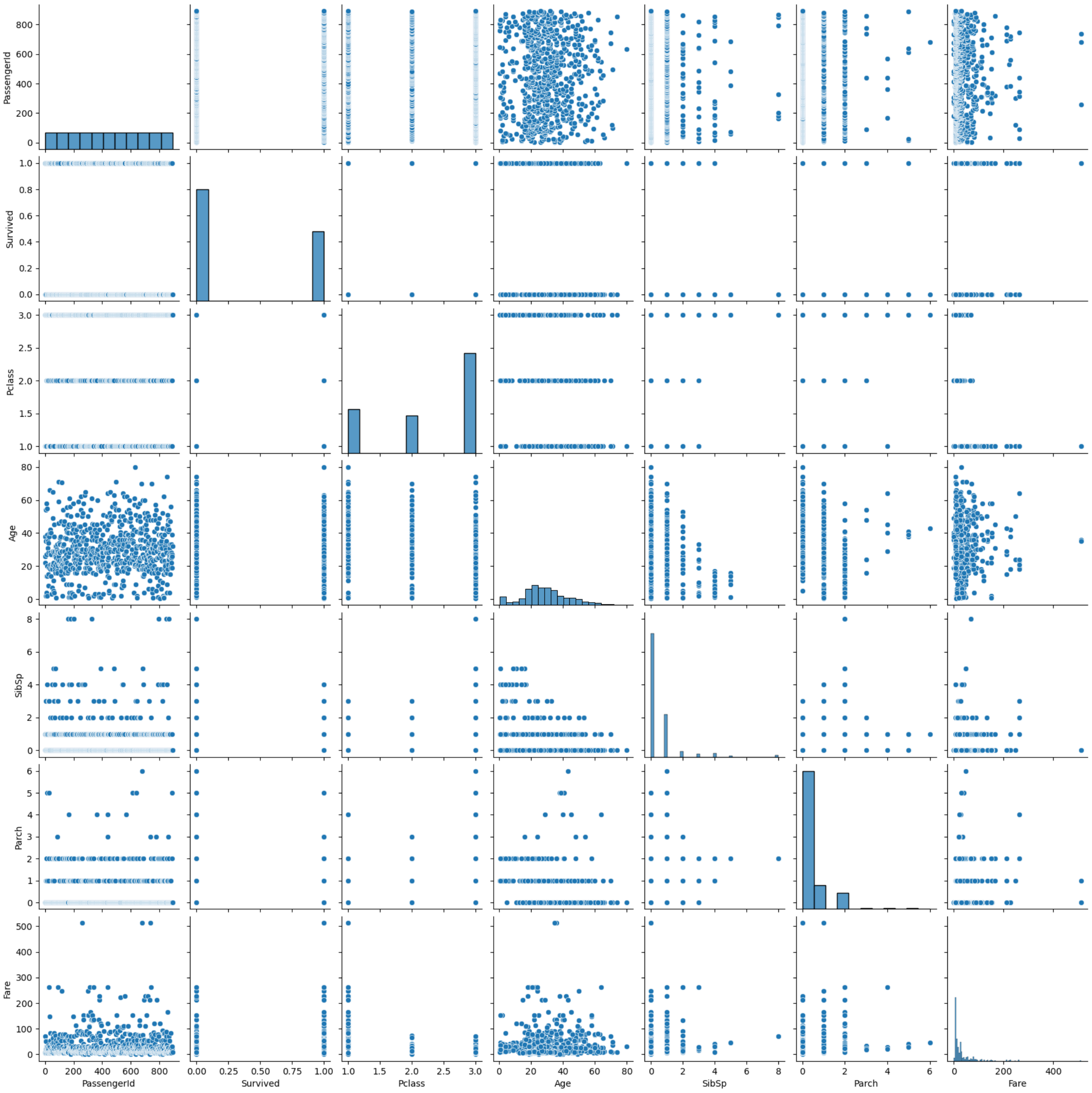
```
In [49]: gender_counts = data['Sex'].value_counts()
         plt.pie(gender_counts, labels=gender_counts.index, autopct='%1.1f%%', startangle
         plt.title('Gender Distribution')
```

```
Out[49]: Text(0.5, 1.0, 'Gender Distribution')
```



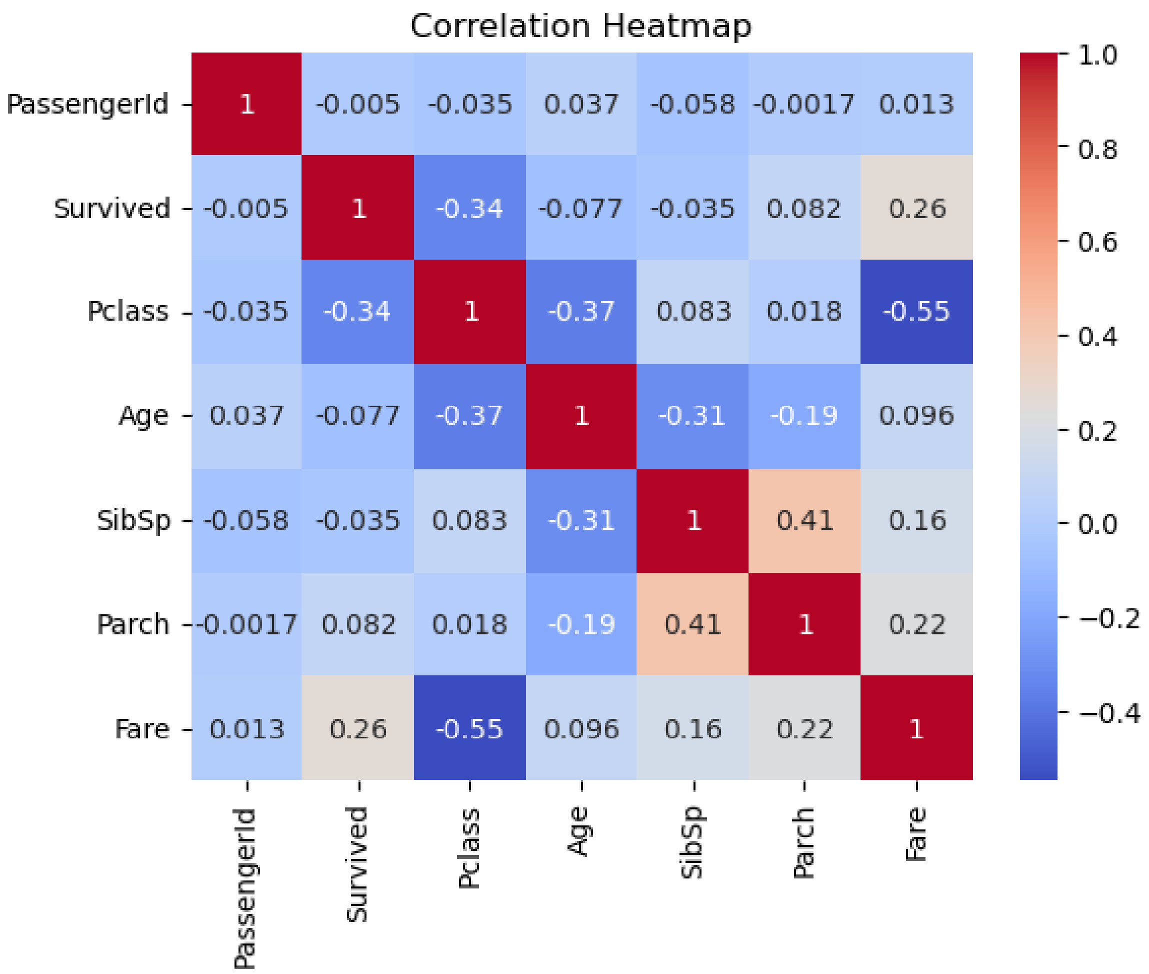
```
In [52]: sns.pairplot(data)
```

```
Out[52]: <seaborn.axisgrid.PairGrid at 0x28b1449be90>
```



```
In [50]: correlation_matrix = data.corr()
         sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
         plt.title('Correlation Heatmap')
         plt.show()
```

```
C:\Users\chatu\AppData\Local\Temp\ipykernel_13368\3963569686.py:1: FutureWarning:
The default value of numeric_only in DataFrame.corr is deprecated. In a future ve
rsion, it will default to False. Select only valid columns or specify the value o
f numeric_only to silence this warning.
  correlation_matrix = data.corr()
```



Outlier Detection

In [63]:
```
col='Fare'
Q1 = data[col].quantile(0.25)
Q3 = data[col].quantile(0.75)
IQR = Q3 - Q1
IQR
```

Out[63]:
```
23.0896
```

In [64]:
```
# Determine outlier boundaries
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Identify outliers
outliers = data[(data[col] < lower_bound) | (data[col] > upper_bound)]
outliers
```

Out[64]:

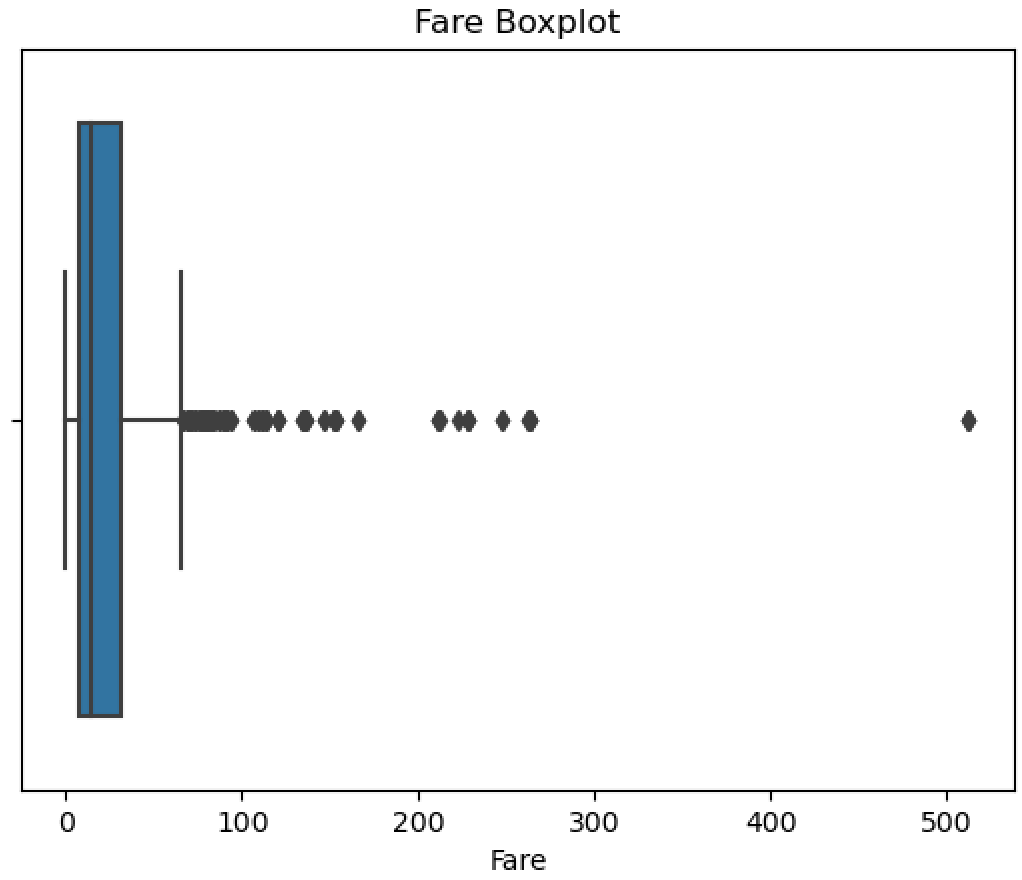
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	7
27	28	0	1	Fortune, Mr. Charles Alexander	male	19.0	3	2	19950	26
31	32	1	1	Spencer, Mrs. William Augustus (Marie Eugenie)	female	NaN	1	0	PC 17569	14
34	35	0	1	Meyer, Mr. Edgar Joseph	male	28.0	1	0	PC 17604	8
52	53	1	1	Harper, Mrs. Henry Sleeper (Myna Haxtun)	female	49.0	1	0	PC 17572	7
...	...	...	...	...	...	...	...	...	...	
846	847	0	3	Sage, Mr. Douglas Bullen	male	NaN	8	2	CA. 2343	6
849	850	1	1	Goldenberg, Mrs. Samuel L (Edwiga Grabowska)	female	NaN	1	0	17453	8
856	857	1	1	Wick, Mrs. George Dennick (Mary Hitchcock)	female	45.0	1	1	36928	16
863	864	0	3	Sage, Miss. Dorothy Edith "Dolly"	female	NaN	8	2	CA. 2343	6
879	880	1	1	Potter, Mrs. Thomas Jr (Lily Alexenia Wilson)	female	56.0	0	1	11767	8

116 rows × 12 columns

In [12]:

```
sns.boxplot(x=data['Fare'])
plt.xlabel('Fare')
plt.title('Fare Boxplot')
plt.show()

# Handle outliers (example: capping extreme fare values)
data['Fare'] = np.where(data['Fare'] > data['Fare'].quantile(0.95), data['Fare']
```

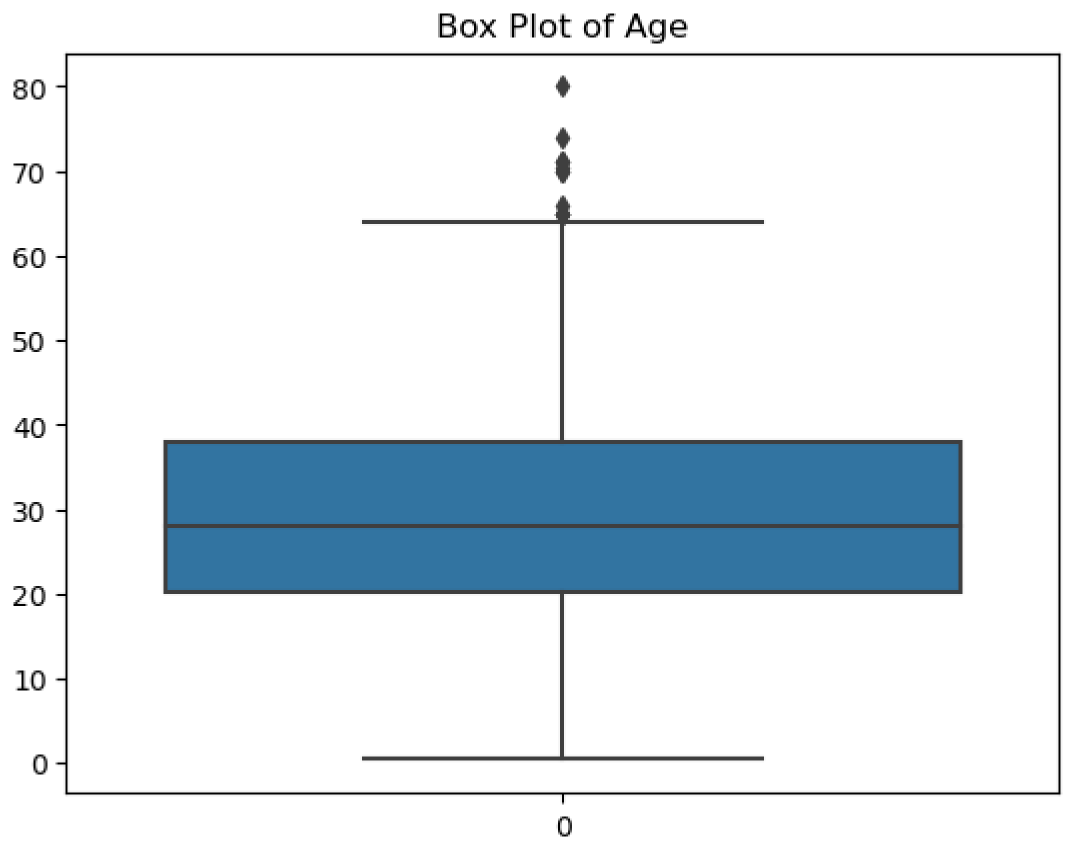


In [67]:

```
sns.boxplot(data["Age"])
plt.title('Box Plot of Age')
```

Out[67]:

```
Text(0.5, 1.0, 'Box Plot of Age')
```



Splitting Dependent and independent Variables

```
In [73]: X=data.drop(columns=["Survived"],axis=1)
         X.head()
```

Out[73]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	1	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C8
2	3	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C12
4	5	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN

In [74]:

```
X.shape
```

Out[74]:

```
(891, 11)
```

In [75]:

```
type(X)
```

Out[75]:

```
pandas.core.frame.DataFrame
```

In [76]:

```
y=data["Survived"]
y.head()
```

Out[76]:

```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

Perform Encoding

In [78]:

```
X.head()
```

Out[78]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	1	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	Na
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C8
2	3	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	Na
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C12
4	5	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	Na

In [79]:

```python
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

In [89]:

```python
X["Embarked"]=le.fit_transform(X["Embarked"])
```

In [90]:

```python
X.head()
```

Out[90]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Em
0	1	3	Braund, Mr. Owen Harris	1	28	1	0	523	7.2500	NaN	
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	51	1	0	596	71.2833	C85	
2	3	3	Heikkinen, Miss. Laina	0	34	0	0	669	7.9250	NaN	
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	47	1	0	49	53.1000	C123	
4	5	3	Allen, Mr. William Henry	1	47	0	0	472	8.0500	NaN	

In [91]:

```
print(le.classes_)
```

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

In [92]:

```
mapping=dict(zip(le.classes_,range(len(le.classes_))))
mapping
```

Out[92]:

```
{'C': 0, 'Q': 1, 'S': 2, nan: 3}
```

Feature Scaling

In [101…

```
from sklearn.preprocessing import MinMaxScaler
cols = ['Age', 'Fare']
# Initialize the MinMaxScaler
scaler = MinMaxScaler()
X = data[cols]
# Fit the scaler to the data and transform the selected columns
Xscale = scaler.fit_transform(X)
```

In [103…

```
Xscale=pd.DataFrame(scaler.fit_transform(X),columns=cols)
```

In [104…

```
Xscale.head()
```

Out[104…

	Age	Fare
0	0.271174	0.014151
1	0.472229	0.139136
2	0.321438	0.015469
3	0.434531	0.103644
4	0.434531	0.015713

Splitting Data into Train and Test

In [107…

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(Xscale, y, test_size=0.2, ra
print(X_train.shape,X_test.shape,y_train.shape,y_test.shape)
```

```
(712, 2) (179, 2) (712,) (179,)
```