In [3]:
```
#import libraries for titanic project

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

In [4]:
```
# importing dataset

df=pd.read_csv(r"C:\Users\ARUTHRA D\Downloads\archive (1)\Titanic-Dataset.csv")
print(df)
```

```
     PassengerId  Survived  Pclass  \
0              1         0       3
1              2         1       1
2              3         1       3
3              4         1       1
4              5         0       3
..           ...       ...     ...
886          887         0       2
887          888         1       1
888          889         0       3
889          890         1       1
890          891         0       3

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

     Parch            Ticket     Fare Cabin Embarked
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     ...               ...      ...   ...      ...
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

In [5]:
```
df.shape
```

Out[5]:
```
(891, 12)
```

In [6]:

```
df.describe()
```

Out[6]:

	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 [7]:

```
# Check the columns

print(df.columns)
```

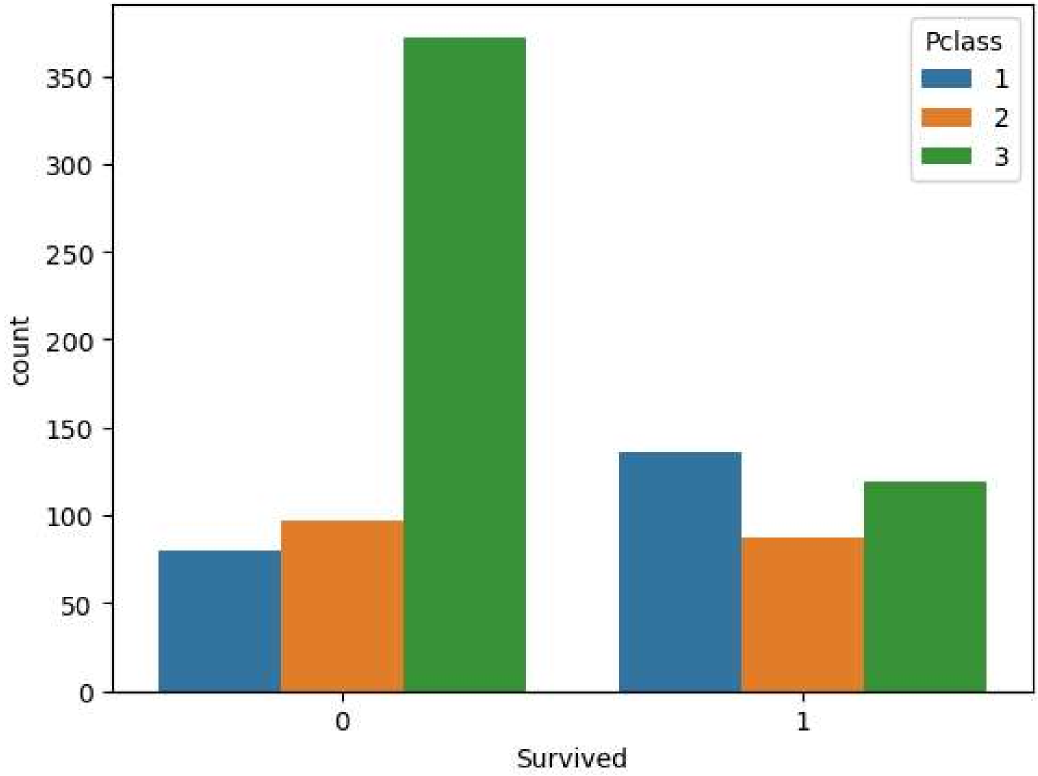
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [8]:

```
# If the column is named 'Survived'

print(df['Survived'].value_counts())
```

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

In [21]:
```
# Plotting with seaborn (sns)

import seaborn as sns
import matplotlib.pyplot as plt

sns.countplot(x=df['Survived'], hue=df['Pclass'])
plt.show()
```



In [1]:
```
# data cleaning
```

In [9]:
```
f.drop('PassengerId',inplace=True,axis=1)
df['Cabin'].count()/df.shape[0]
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[9], line 1
----> 1 f.drop('PassengerId',inplace=True,axis=1)
      2 df['Cabin'].count()/df.shape[0]

NameError: name 'f' is not defined
```

In [10]:
```
df.drop('PassengerId',inplace=True,axis=1)
df['Cabin'].count()/df.shape[0]
```

Out[10]:
```
0.22895622895622897
```

In [11]:

```
df.drop('Cabin',inplace=True,axis=1)#because there is more than 70% empty values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  891 non-null    int64  
 1   Pclass    891 non-null    int64  
 2   Name      891 non-null    object 
 3   Sex       891 non-null    object 
 4   Age       714 non-null    float64
 5   SibSp     891 non-null    int64  
 6   Parch     891 non-null    int64  
 7   Ticket    891 non-null    object 
 8   Fare      891 non-null    float64
 9   Embarked  889 non-null    object 
dtypes: float64(2), int64(4), object(4)
memory usage: 69.7+ KB
```

In [12]:

```
df['Age'].count()/df.shape[0]
0.8013468013468014
df.dropna(subset=['Age','Embarked'],inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 712 entries, 0 to 890
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  712 non-null    int64  
 1   Pclass    712 non-null    int64  
 2   Name      712 non-null    object 
 3   Sex       712 non-null    object 
 4   Age       712 non-null    float64
 5   SibSp     712 non-null    int64  
 6   Parch     712 non-null    int64  
 7   Ticket    712 non-null    object 
 8   Fare      712 non-null    float64
 9   Embarked  712 non-null    object 
dtypes: float64(2), int64(4), object(4)
memory usage: 61.2+ KB
```

In [13]:
```
df['Title']=df['Name'].str.extract(r'([A-Za-z]+\.)',expand=False)
df.drop('Name',inplace=True,axis=1)
pd.set_option('display.max_columns',None)
pd.set_option('display.max_rows',None)
df
```

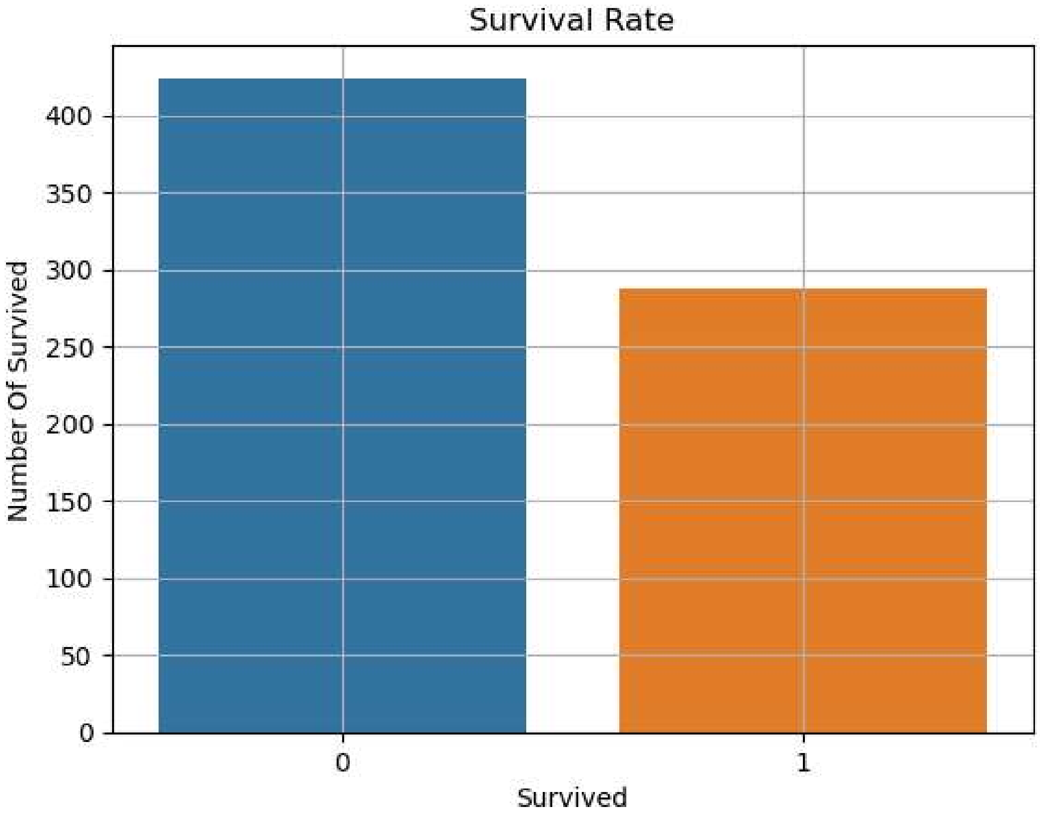
333	0	3	male	16.00	2	0	345764	18.0000	S	Mr.
336	0	1	male	29.00	1	0	113776	66.6000	S	Mr.
337	1	1	female	41.00	0	0	16966	134.5000	C	Miss.
338	1	3	male	45.00	0	0	7598	8.0500	S	Mr.
339	0	1	male	45.00	0	0	113784	35.5000	S	Mr.
340	1	2	male	2.00	1	1	230080	26.0000	S	Master.
341	1	1	female	24.00	3	2	19950	263.0000	S	Miss.
342	0	2	male	28.00	0	0	248740	13.0000	S	Mr.
343	0	2	male	25.00	0	0	244361	13.0000	S	Mr.
344	0	2	male	36.00	0	0	229236	13.0000	S	Mr.
345	1	2	female	24.00	0	0	248733	13.0000	S	Miss.
346	1	2	female	40.00	0	0	31418	13.0000	S	Miss.
348	1	3	male	3.00	1	1	C.A. 37671	15.9000	S	Master.

In [14]:
```
df['Survived'].value_counts()
```

Out[14]:
```
Survived
0    424
1    288
Name: count, dtype: int64
```

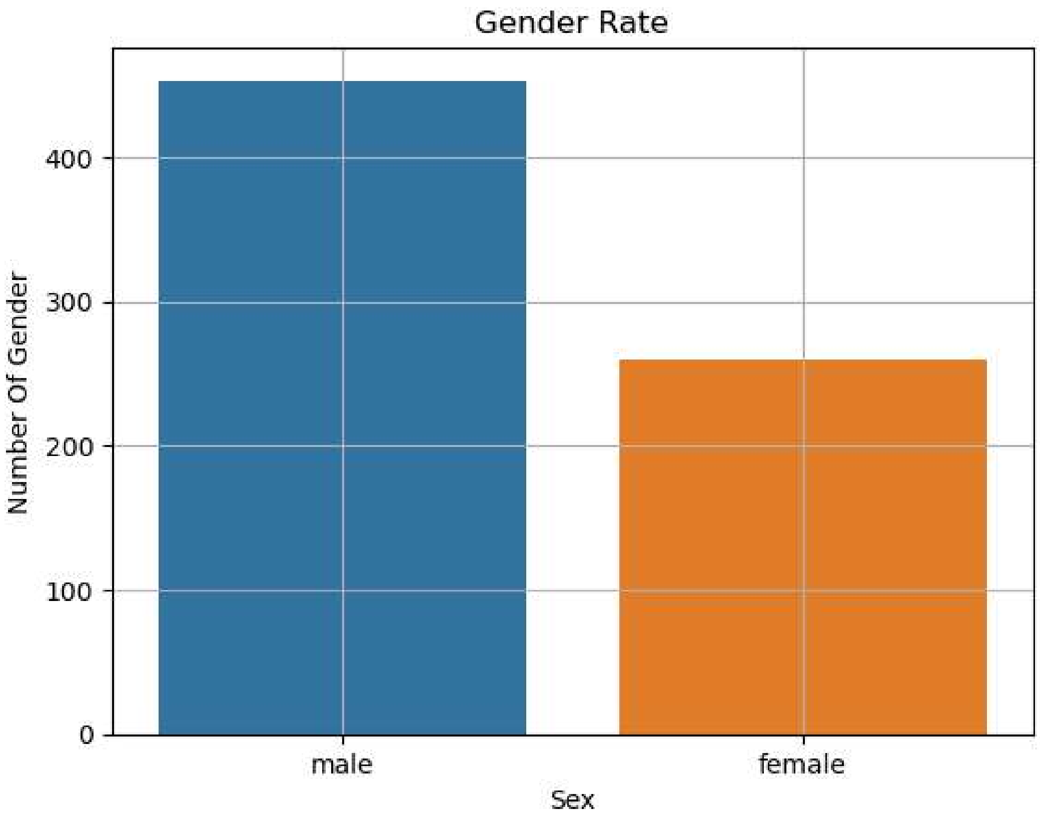
In [15]:

```
sns.countplot(data=df,x='Survived')
plt.ylabel('Number Of Survived')
plt.title('Survival Rate')
plt.grid()
plt.show()
```



In [16]:

```python
sns.countplot(x='Sex',data=df)
plt.ylabel('Number Of Gender')
plt.title('Gender Rate')
plt.grid()
plt.show()
```



In [17]:

```python
df['Sex'].value_counts()
```

Out[17]:

```
Sex
male      453
female    259
Name: count, dtype: int64
```

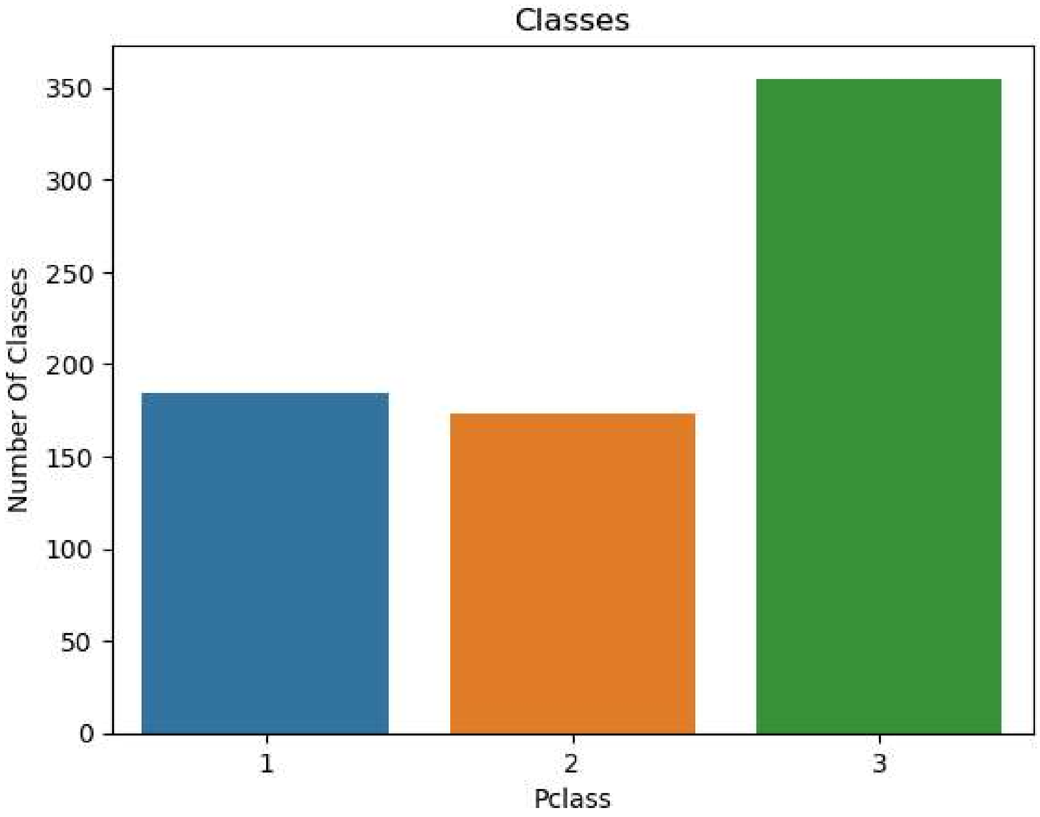
In [18]:

```python
df['Pclass'].value_counts()
```

Out[18]:

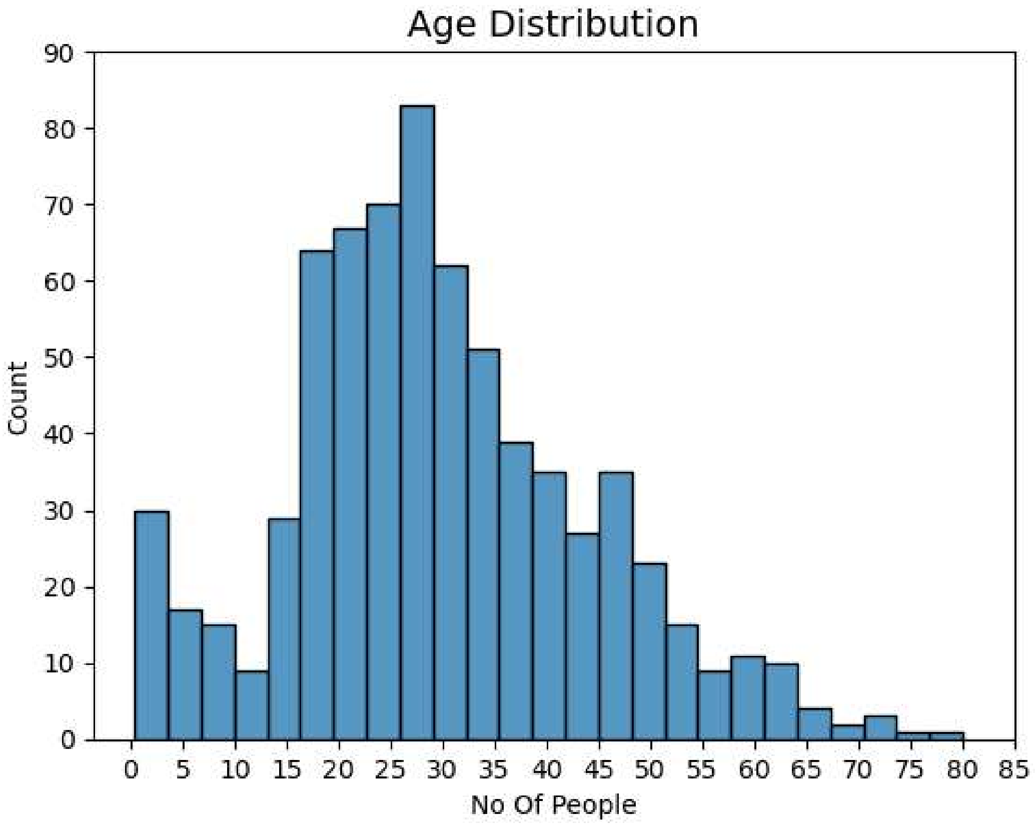
```
Pclass
3    355
1    184
2    173
Name: count, dtype: int64
```

In [19]:
```
sns.countplot(x='Pclass',data=df)
plt.ylabel('Number Of Classes')
plt.title('Classes')
plt.show()
```



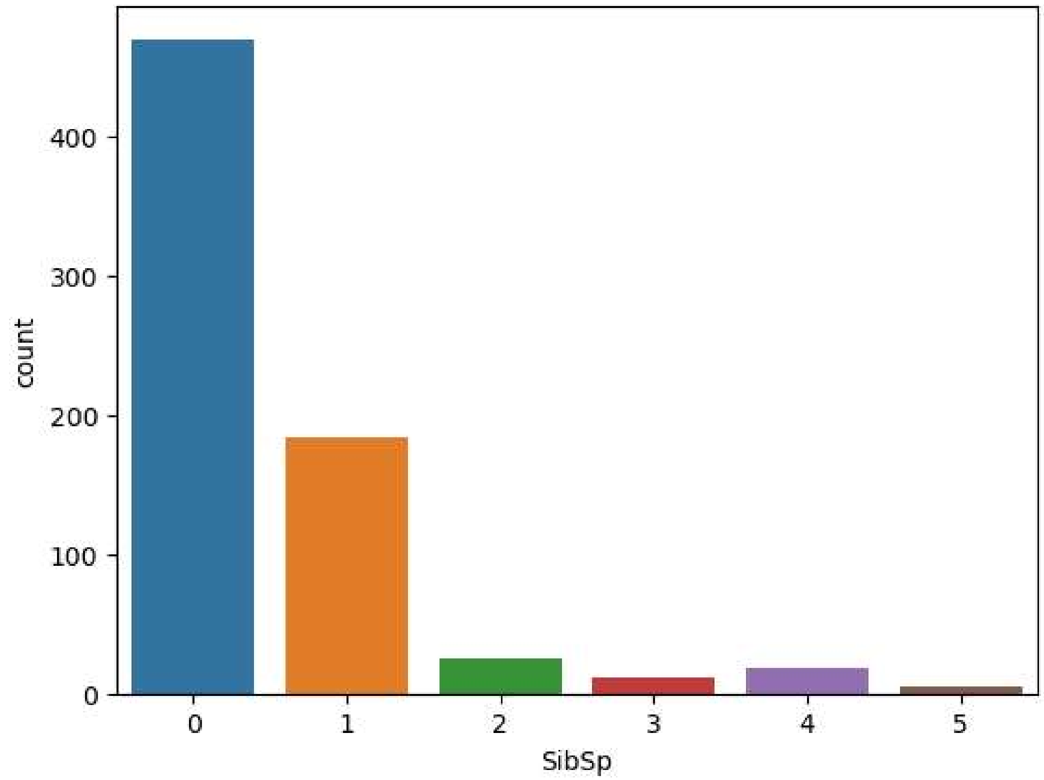
In [20]:

```
sns.histplot(df['Age'],bins=25)
plt.yticks(np.arange(0,100,10))
plt.xticks(np.arange(0,90,5))
plt.title('Age Distribution',fontsize=14)
plt.xlabel('No Of People')
plt.show()
```



In [21]:

```
sns.countplot(x='SibSp',data=df)
plt.show()
```

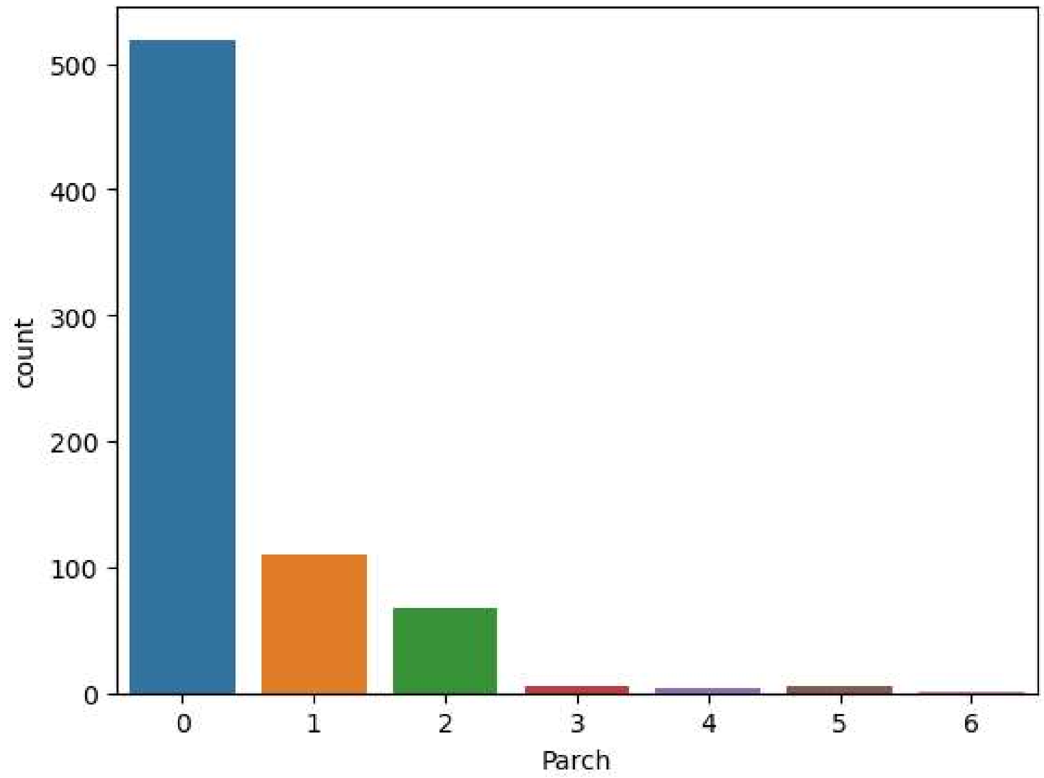


In [22]:

```
sns.countplot(x='Parch',data=df)
```

Out[22]:

```
<Axes: xlabel='Parch', ylabel='count'>
```



In [23]:

```
print('Siblings:\n',df['SibSp'].value_counts())
print('Parents : \n',df['Parch'].value_counts())
```

```
Siblings:
 SibSp
0    469
1    183
2     25
4     18
3     12
5      5
Name: count, dtype: int64
Parents :
 Parch
0    519
1    110
2     68
5      5
3      5
4      4
6      1
Name: count, dtype: int64
```

In [24]:

```
df.min()
```

Out[24]:

```
Survived         0
Pclass           1
Sex         female
Age           0.42
SibSp            0
Parch            0
Ticket      110152
Fare           0.0
Embarked         C
Title        Capt.
dtype: object
```

In [26]:

```
df.max()
```

Out[26]:

```
Survived            1
Pclass              3
Sex              male
Age              80.0
SibSp               5
Parch               6
Ticket      WE/P 5735
Fare         512.3292
Embarked            S
Title            Sir.
dtype: object
```

In [27]:

```
df.mode()
```

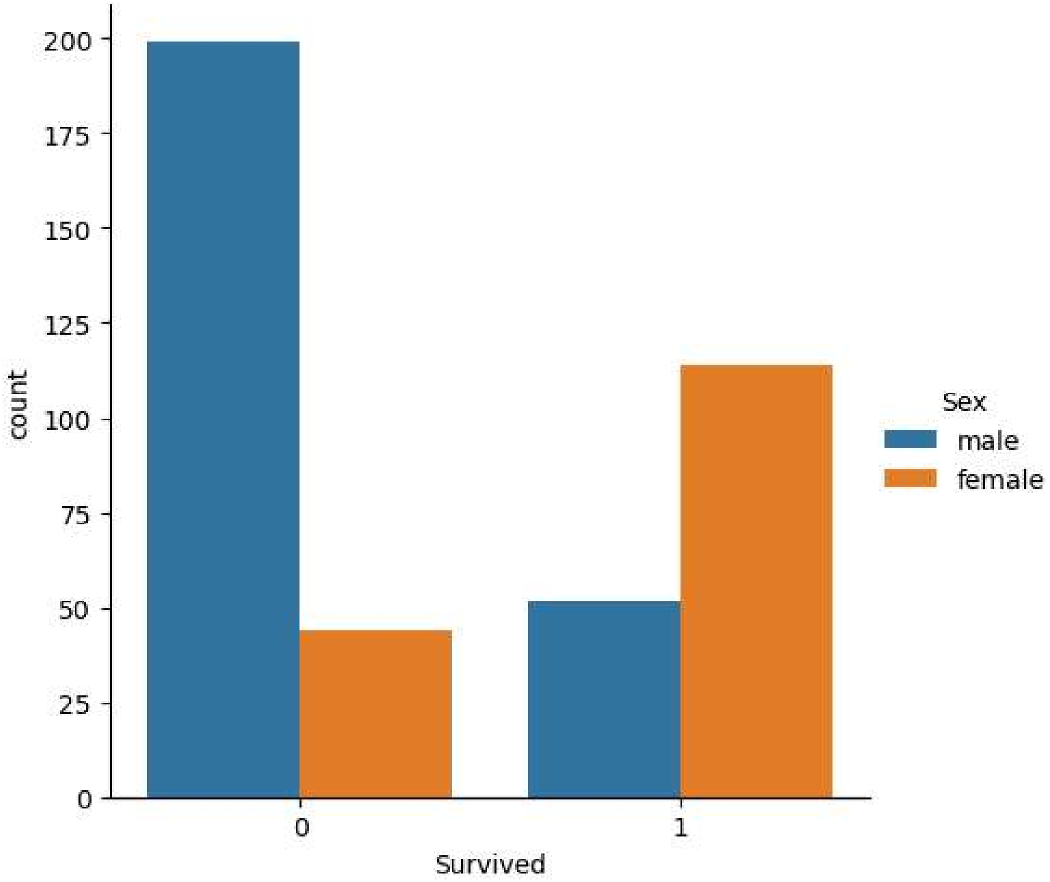
Out[27]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Title
0	0	3	male	24.0	0	0	347082	13.0	S	Mr.

In [28]:

```
ind=df[df['Age']>30].index
df_1=df.drop(ind,axis=0)
sns.catplot(x='Survived',hue='Sex',kind='count',data=df_1)
plt.show()
```

```
C:\Users\ARUTHRA D\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: Th
e figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [29]:

```
df.groupby('Survived')['Sex'].value_counts()
```

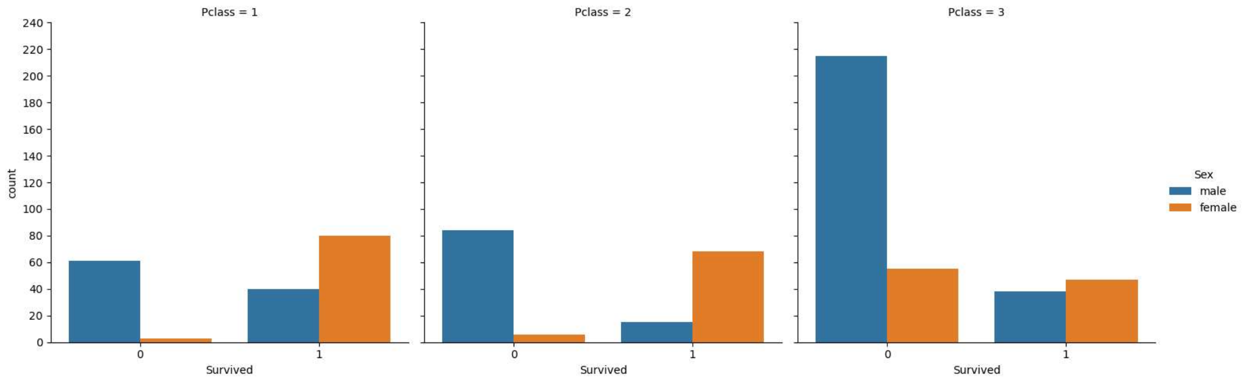
Out[29]:

```
Survived  Sex   
0         male      360
          female     64
1         female    195
          male       93
Name: count, dtype: int64
```

In [30]:

```
sns.catplot(x='Survived',hue='Sex',col='Pclass',data=df,kind='count')
plt.yticks(np.arange(0,250,20))
plt.show()
```

```
C:\Users\ARUTHRA D\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: Th
e figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [31]:

```
df.groupby(['Survived','Pclass'])['Sex'].value_counts()
```

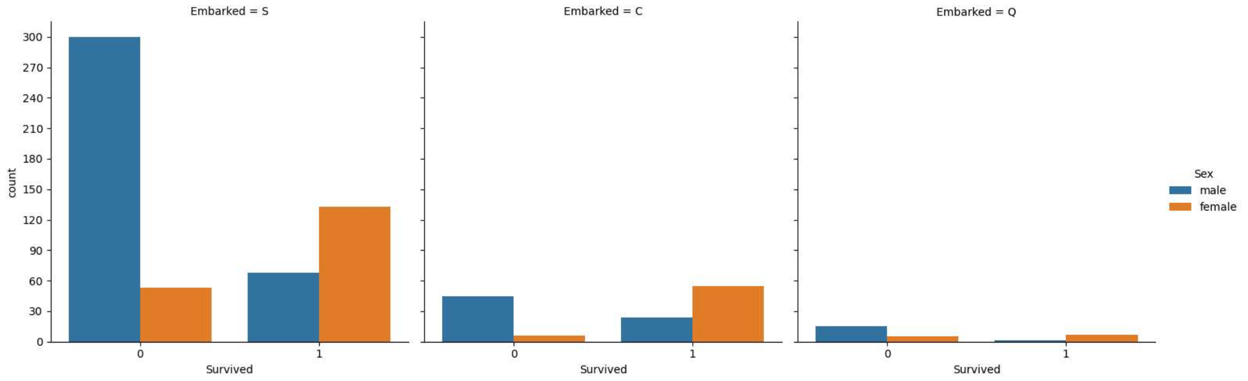
Out[31]:

```
Survived  Pclass  Sex   
0         1       male       61
                  female      3
          2       male       84
                  female      6
          3       male      215
                  female     55
1         1       female     80
                  male       40
          2       female     68
                  male       15
          3       female     47
                  male       38
Name: count, dtype: int64
```

In [32]:

```
sns.catplot(x='Survived',hue='Sex',col='Embarked',data=df,kind='count')
plt.yticks(np.arange(0,330,30))
plt.show()
```

```
C:\Users\ARUTHRA D\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: Th
e figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [33]:

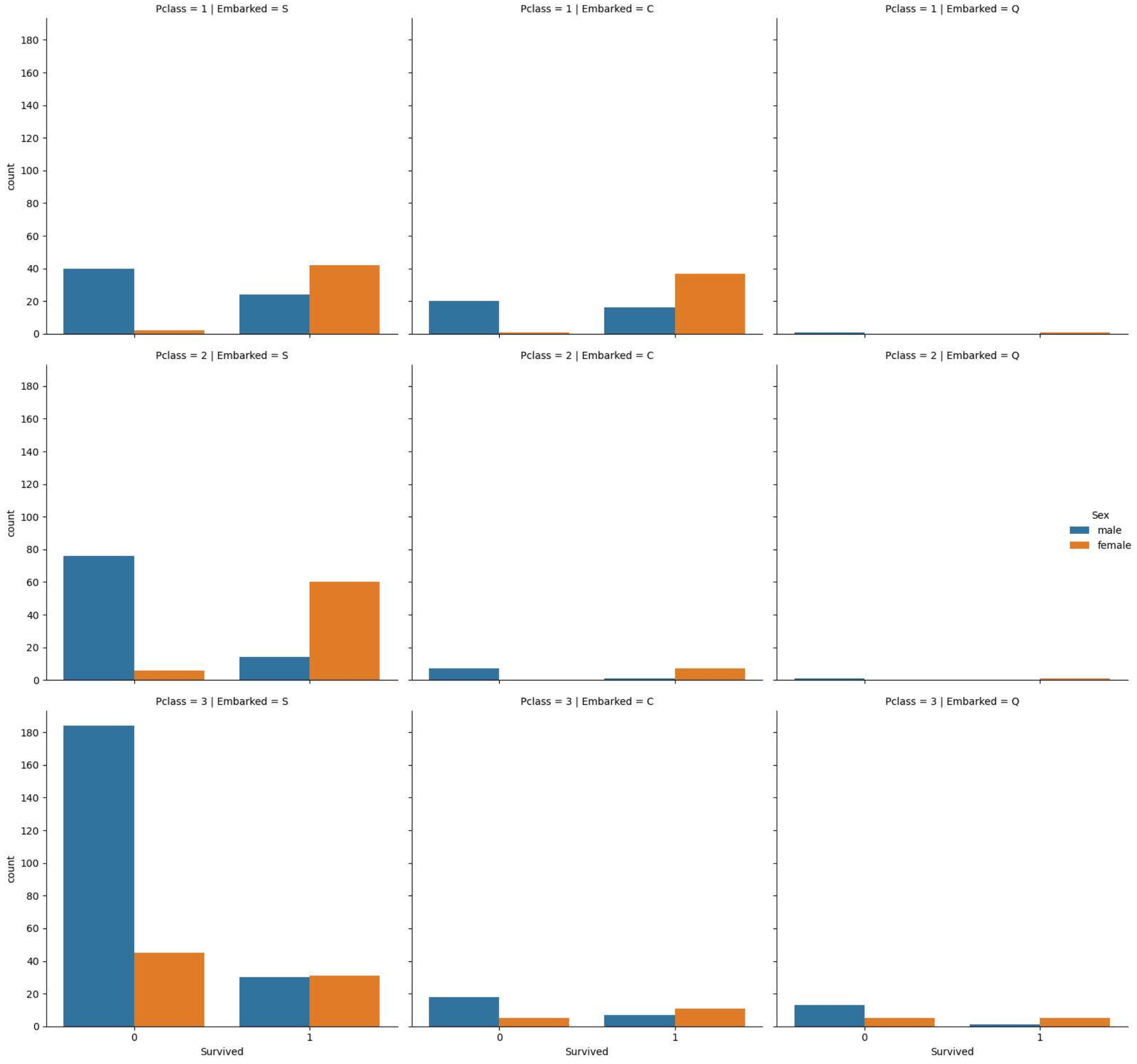
```
df.groupby(['Survived','Embarked'])['Sex'].value_counts()
```

Out[33]:

```
Survived  Embarked  Sex   
0         C         male       45
                    female      6
          Q         male       15
                    female      5
          S         male      300
                    female     53
1         C         female     55
                    male       24
          Q         female      7
                    male        1
          S         female    133
                    male       68
Name: count, dtype: int64
```

```
In [34]: sns.catplot(x='Survived',hue='Sex',col='Embarked',row='Pclass',data=df,kind='count')
         plt.yticks(np.arange(0,190,20))
         plt.tight_layout()
         plt.show()
```

```
C:\Users\ARUTHRA D\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: Th
e figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\Users\ARUTHRA D\AppData\Local\Temp\ipykernel_15688\1850565211.py:3: UserWarning: The
figure layout has changed to tight
  plt.tight_layout()
```

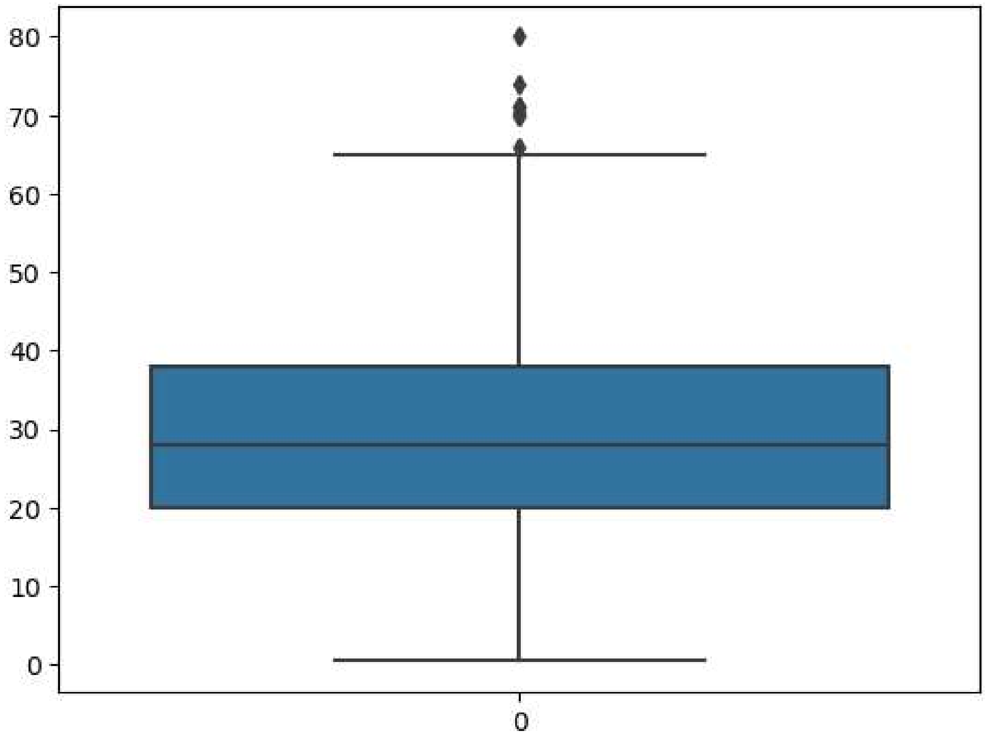


```
In [35]: df.groupby(['Survived','Embarked','Pclass'])['Sex'].value_counts()
```

```
Out[35]: Survived  Embarked  Pclass  Sex   
         0         C         1       male       20
                                     female      1
                             2       male        7
                             3       male       18
                                     female      5
                   Q         1       male        1
                             2       male        1
                             3       male       13
                                     female      5
                   S         1       male       40
                                     female      2
                             2       male       76
                                     female      6
                             3       male      184
                                     female     45
         1         C         1       female     37
                                     male       16
                             2       female      7
                                     male        1
                             3       female     11
                                     male        7
                   Q         1       female      1
                             2       female      1
                             3       female      5
                                     male        1
                   S         1       female     42
                                     male       24
                             2       female     60
                                     male       14
                             3       female     31
                                     male       30
         Name: count, dtype: int64
```

In [37]:

```
sns.boxplot(df['Age'])
plt.show()
```



In [38]:

```
def boundaries(data,col,dis):
    Q1=data[col].quantile(0.25)
    Q3=data[col].quantile(0.75)
    IQR=Q3-Q1
    low=Q1-(IQR*dis)
    upper=Q3+(IQR*dis)
    return low,upper
lower,upper=boundaries(df,'Age',1.5)
print('Lower Range : ',lower,' Upper Range : ',upper)
```

```
Lower Range :  -7.0  Upper Range :  65.0
```

In [39]:

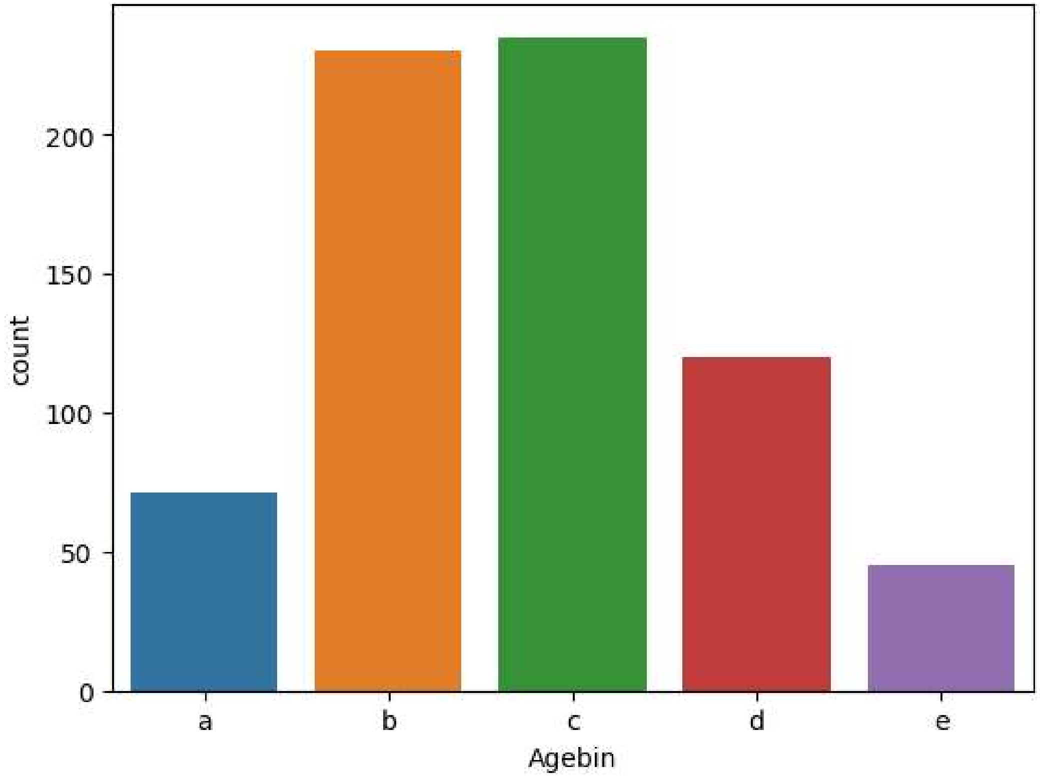
```
not_out=(df['Age']<upper)&(df['Age']>lower)
df['Age'][~not_out].count()
```

Out[39]:

```
11
```

In [40]:
```
df=df[not_out]
df['Agebin']=pd.cut(df['Age'],5,labels=['a','b','c','d','e'],include_lowest=True)
sns.countplot(x='Agebin',data=df)
plt.show()
```

```
C:\Users\ARUTHRA D\AppData\Local\Temp\ipykernel_15688\1455345314.py:2: SettingWithCopyW
arning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-d
ocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df['Agebin']=pd.cut(df['Age'],5,labels=['a','b','c','d','e'],include_lowest=True)
```



In [48]:
```
# Step 1: Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [45]:
```
# Step 2: Load the dataset
file_path = r"C:\Users\ARUTHRA D\Downloads\archive (1)\Titanic-Dataset.csv"
titanic_data = pd.read_csv(file_path)
```

In [46]:

```
# Step 3: Data Preprocessing
# Handling missing values
titanic_data['Age'].fillna(titanic_data['Age'].median(), inplace=True)
titanic_data['Embarked'].fillna(titanic_data['Embarked'].mode()[0], inplace=True)
titanic_data.drop(columns=['Cabin'], inplace=True)
```

In [49]:

```
# Encoding categorical variables
label_encoder = LabelEncoder()
titanic_data['Sex'] = label_encoder.fit_transform(titanic_data['Sex'])
titanic_data['Embarked'] = label_encoder.fit_transform(titanic_data['Embarked'])
```

In [50]:

```
# Selecting features and target variable
features = ['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked']
X = titanic_data[features]
y = titanic_data['Survived']
```

In [51]:

```
# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42

# Scaling the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [52]:

```
# Step 4: Model Building
model = LogisticRegression()
model.fit(X_train, y_train)

# Making predictions
y_pred = model.predict(X_test)

# Step 5: Model Evaluation
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
```

In [53]:

```
print(f"Accuracy: {accuracy}")
print("Confusion Matrix:")
print(conf_matrix)
print("Classification Report:")
print(class_report)
```

```
Accuracy: 0.8044692737430168
Confusion Matrix:
[[90 15]
 [20 54]]
Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.86      0.84       105
           1       0.78      0.73      0.76        74

    accuracy                           0.80       179
   macro avg       0.80      0.79      0.80       179
weighted avg       0.80      0.80      0.80       179
```

In [54]:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
```

In [56]:

```
# importing dataset

df=pd.read_csv(r"C:\Users\ARUTHRA D\Downloads\archive (1)\Titanic-Dataset.csv")
print(df)
```

```
143          144         0       3
144          145         0       2
145          146         0       2
146          147         1       3
147          148         0       3
148          149         0       2
149          150         0       2
150          151         0       2
151          152         1       1
152          153         0       3
153          154         0       3
154          155         0       3
155          156         0       1
156          157         1       3
157          158         0       3
158          159         0       3
159          160         0       3
160          161         0       3
161          162         1       2
162          163         0       3
```

In [57]:

```
# Load the dataset
titanic_data = pd.read_csv(r"C:\Users\ARUTHRA D\Downloads\archive (1)\Titanic-Dataset.cs

# Display the first few rows of the dataset
print(titanic_data.head())
```

```
   PassengerId  Survived  Pclass  \
0            1         0       3   
1            2         1       1   
2            3         1       3   
3            4         1       1   
4            5         0       3   

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1   
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1   
2                             Heikkinen, Miss. Laina  female  26.0      0   
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1   
4                           Allen, Mr. William Henry    male  35.0      0   

   Parch            Ticket     Fare Cabin Embarked  
0      0         A/5 21171   7.2500   NaN        S  
1      0          PC 17599  71.2833   C85        C  
2      0  STON/O2. 3101282   7.9250   NaN        S  
3      0            113803  53.1000  C123        S  
4      0            373450   8.0500   NaN        S  
```

In [58]:

```
# Drop columns that are not needed for this analysis
titanic_data = titanic_data.drop(columns=['Name', 'Ticket', 'Cabin'])

# Encode categorical variables
label_encoder = LabelEncoder()
titanic_data['Sex'] = label_encoder.fit_transform(titanic_data['Sex'])
titanic_data['Embarked'] = titanic_data['Embarked'].fillna('S')
titanic_data['Embarked'] = label_encoder.fit_transform(titanic_data['Embarked'])

# Handle missing values
imputer = SimpleImputer(strategy='mean')
titanic_data['Age'] = imputer.fit_transform(titanic_data[['Age']])

# Display the preprocessed data
print(titanic_data.head())
```

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

In [59]:

```
# Define features and target variable
X = titanic_data.drop(columns=['Survived'])
y = titanic_data['Survived']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42
```

In [60]:

```
# Initialize and train the Random Forest classifier
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)
```

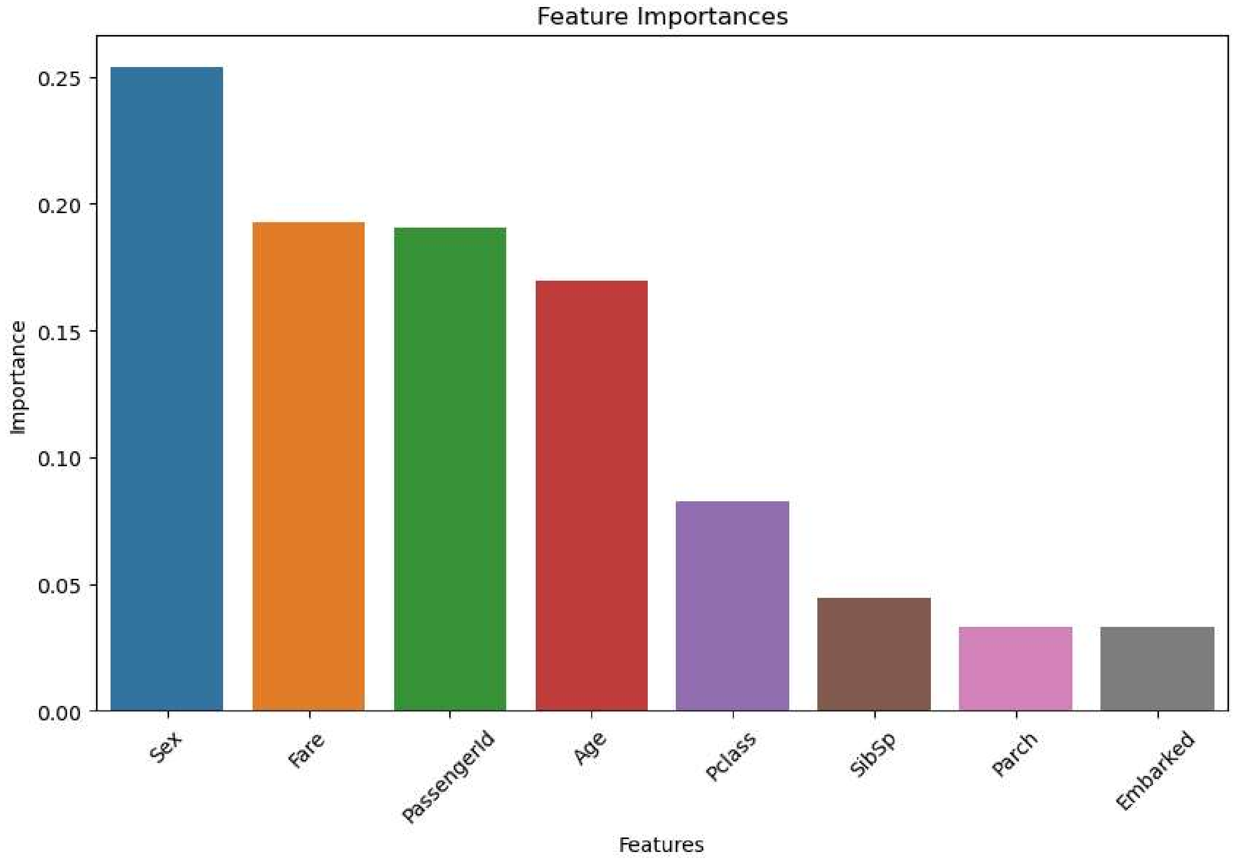
In [61]:

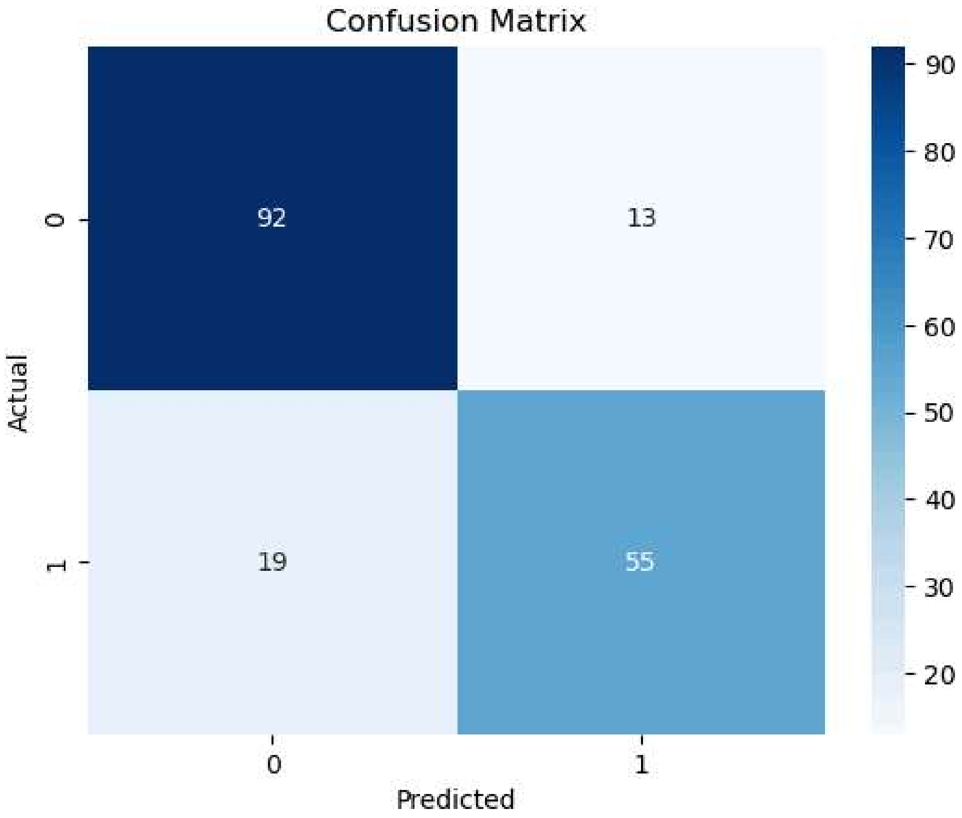
```
# Print classification report and confusion matrix
print("Classification Report:")
print(classification_report(y_test, y_pred))

print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.88      0.85       105
           1       0.81      0.74      0.77        74

    accuracy                           0.82       179
   macro avg       0.82      0.81      0.81       179
weighted avg       0.82      0.82      0.82       179

Confusion Matrix:
[[92 13]
 [19 55]]
```

In [62]:

```
# Plot feature importances
feature_importances = pd.DataFrame(model.feature_importances_, index=X.columns, columns=

plt.figure(figsize=(10, 6))
sns.barplot(x=feature_importances.index, y=feature_importances['importance'])
plt.title('Feature Importances')
plt.xlabel('Features')
plt.ylabel('Importance')
plt.xticks(rotation=45)
plt.show()

# Plot confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```




Confusion Matrix
Actual
0
1
92
13
19
55
0
1
Predicted
90
80
70
60
50
40
30
20

In [ ]: