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

```
df = pd.read_csv("/content/titanic.csv")
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Next steps: Generate code with df View recommended plots

```
df.columns
```

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

```
df.drop(['PassengerId','Name','Ticket'],axis=1,inplace=True) # Unwanted Columns.
```

```
df.shape
```

```
(891, 9)
```

```
df.duplicated().sum() # Total Duplicates
```

```
107
```

```
df.drop_duplicates(inplace=True) # Droping Duplicates
```

```
df.duplicated().sum() # No More Duplicates
```

```
0
```

```
df.isnull().sum() #null values in every columns
```

```
Survived      0
Pclass        0
Sex           0
Age         106
SibSp         0
Parch         0
Fare          0
Cabin       581
Embarked      2
dtype: int64
```

```
df["Age"] = df["Age"].fillna(df["Age"].mean())
df["Cabin"].fillna(df["Cabin"].mode()[0], inplace=True)    # Handling Missing Values using mean , median , mode
```

```
df.dropna(inplace=True)
```

```
df.isnull().sum() # All Missing Values are handled
```

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

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 782 entries, 0 to 890
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  782 non-null    int64  
 1   Pclass    782 non-null    int64  
 2   Sex       782 non-null    object 
 3   Age       782 non-null    float64
 4   SibSp     782 non-null    int64  
 5   Parch     782 non-null    int64  
 6   Fare      782 non-null    float64
 7   Cabin     782 non-null    object 
 8   Embarked  782 non-null    object 
dtypes: float64(2), int64(4), object(3)
memory usage: 61.1+ KB
```

```
df['Embarked'].unique()
```

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

```
df["Sex"].unique()
```

```
array(['male', 'female'], dtype=object)
```

```
df["Sex"] = pd.get_dummies(df["Sex"],drop_first=True).astype(int)
df.head() # Male -> 1 and Female -> 2
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	0	3	1	22.0	1	0	7.2500	B96 B98	S
1	1	1	0	38.0	1	0	71.2833	C85	C
2	1	3	0	26.0	0	0	7.9250	B96 B98	S
3	1	1	0	35.0	1	0	53.1000	C123	S
4	0	3	1	35.0	0	0	8.0500	B96 B98	S

Next steps: Generate code with df View recommended plots

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df["Embarked"] = le.fit_transform(df['Embarked'])
df.head() # Encoded the Embarked column
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	0	3	1	22.0	1	0	7.2500	B96 B98	2
1	1	1	0	38.0	1	0	71.2833	C85	0
2	1	3	0	26.0	0	0	7.9250	B96 B98	2
3	1	1	0	35.0	1	0	53.1000	C123	2
4	0	3	1	35.0	0	0	8.0500	B96 B98	2

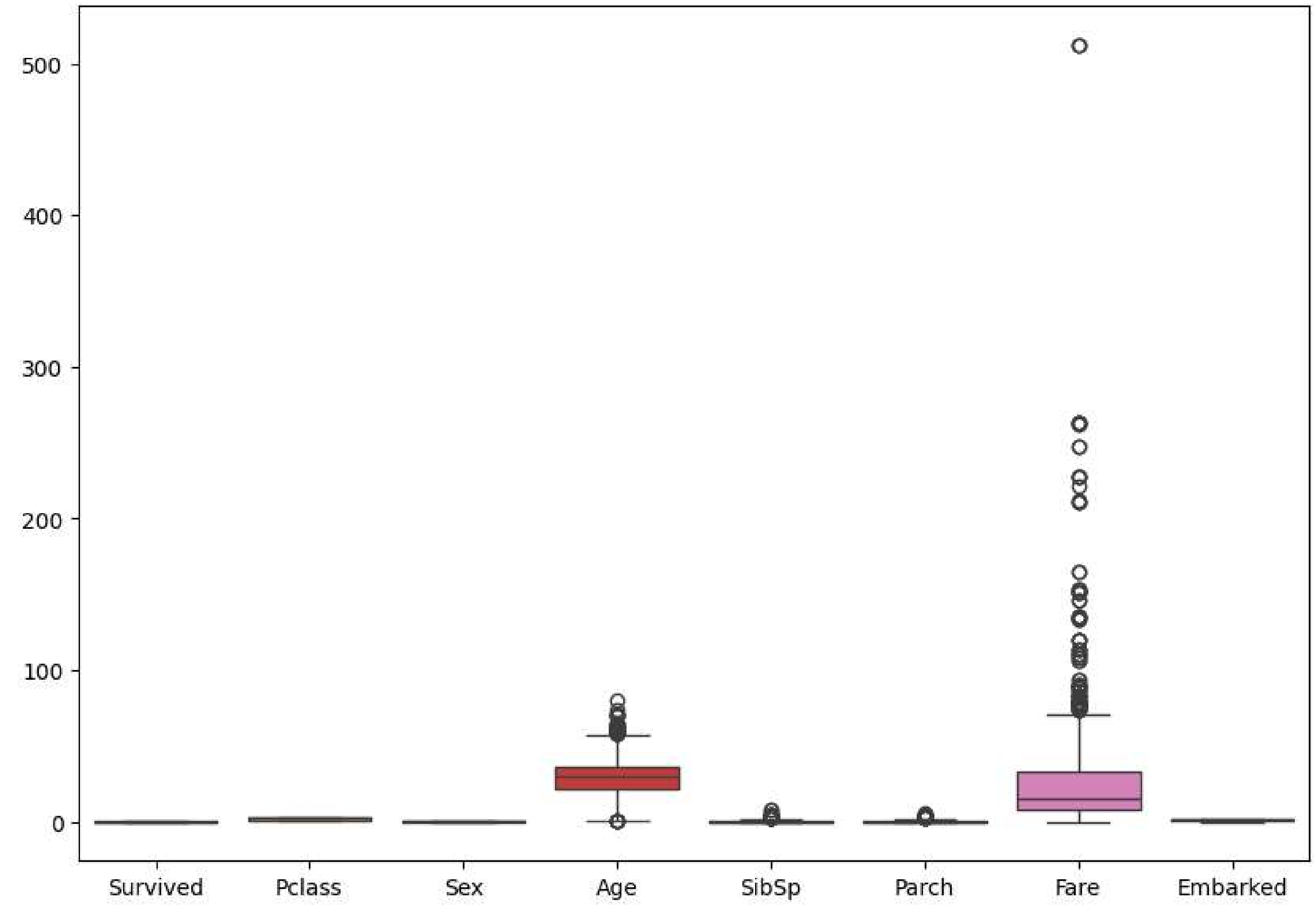
Next steps: Generate code with df View recommended plots

```
df.describe()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
count	782.000000	782.000000	782.000000	782.000000	782.000000	782.000000	782.000000	782.000000
mean	0.410486	2.246803	0.627877	29.817866	0.524297	0.416880	34.595913	1.528133
std	0.492237	0.853828	0.483680	13.689935	0.987138	0.837728	52.176458	0.804024
min	0.000000	1.000000	0.000000	0.420000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	0.000000	22.000000	0.000000	0.000000	8.050000	1.000000
50%	0.000000	3.000000	1.000000	29.869351	0.000000	0.000000	15.875000	2.000000
75%	1.000000	3.000000	1.000000	36.000000	1.000000	1.000000	33.375000	2.000000
max	1.000000	3.000000	1.000000	80.000000	8.000000	6.000000	512.329200	2.000000

```
plt.figure(figsize=(10,7))
sns.boxplot(df)
```

```
<Axes: >
```



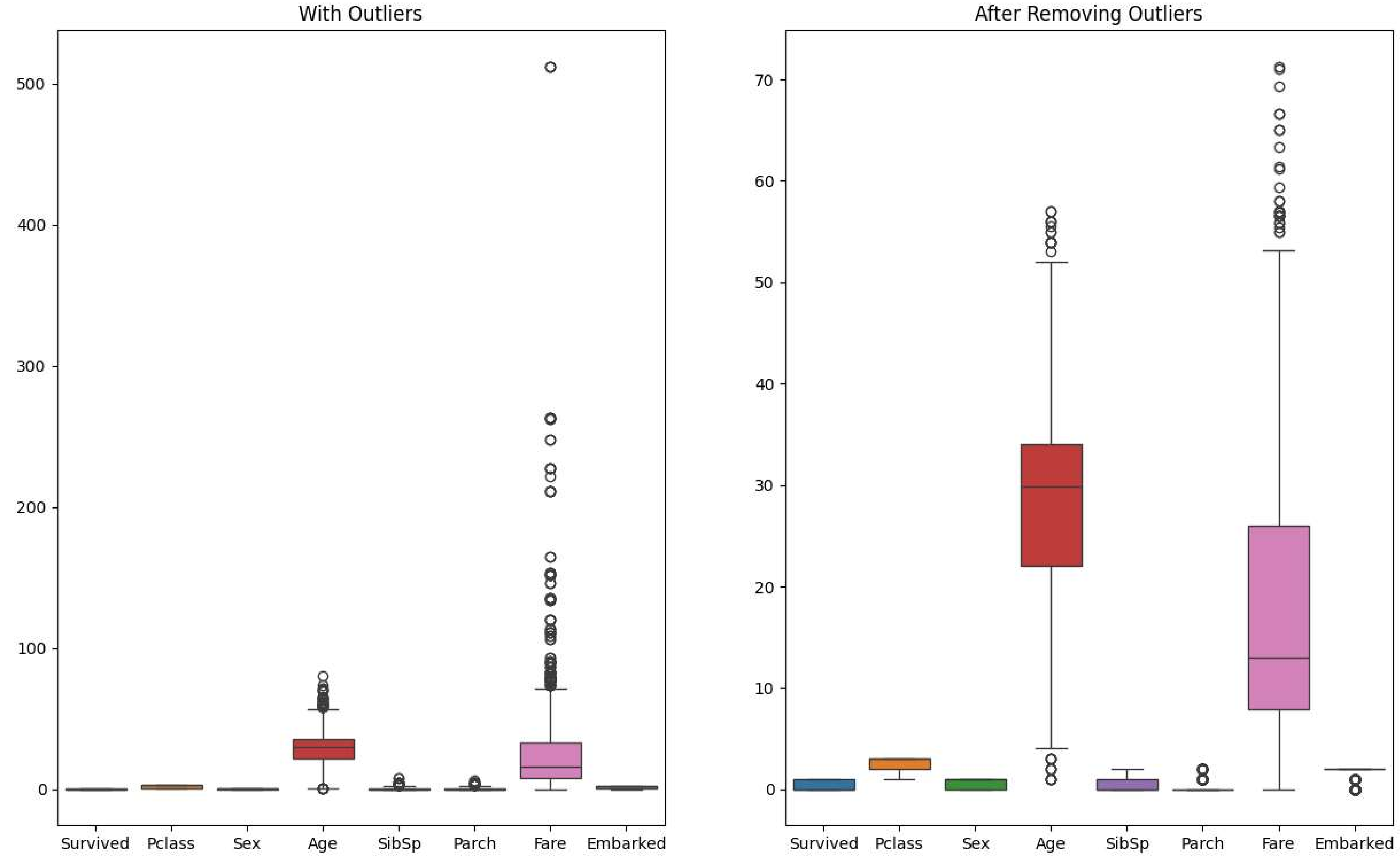
```
numericals = df[["Survived","Pclass","Sex","Age","SibSp","Parch","Fare","Embarked"]]
q1 = numericals.quantile(0.25)
q3 = numericals.quantile(0.75)
IQR = q3-q1
lower = q1 - 1.5*(IQR)
higher = q3 + 1.5*(IQR)
cleaned_data = df[~((numericals < lower) | (numericals > higher)).any(axis=1)]
cleaned_data.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Cabin	Embarked
0	0	3	1	22.0	1	0	7.2500	B96 B98	2
1	1	1	0	38.0	1	0	71.2833	C85	0
2	1	3	0	26.0	0	0	7.9250	B96 B98	2
3	1	1	0	35.0	1	0	53.1000	C123	2
4	0	3	1	35.0	0	0	8.0500	B96 B98	2

Next steps: Generate code with `cleaned_data` View recommended plots

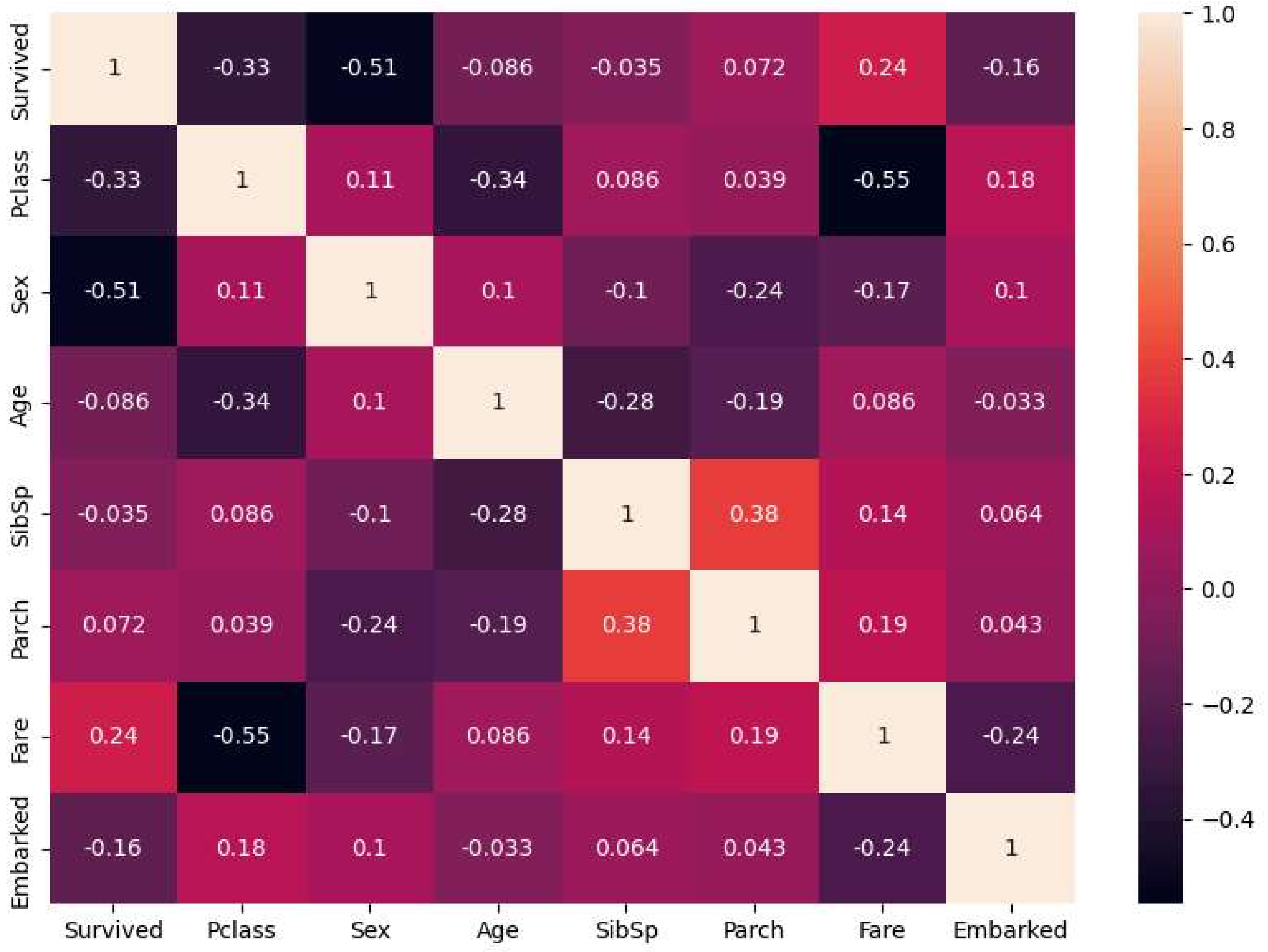
```
import warnings
warnings.filterwarnings('ignore')
plt.figure(figsize=(10,7))
plt.subplots(figsize=(15,9))
plt.subplot(1,2,1)
sns.boxplot(df)
plt.title("With Outliers")
plt.subplot(1,2,2)
sns.boxplot(cleaned_data)
plt.title("After Removing Outliers")

print("Shape of Original Dataframe with outliers : ",df.shape)
print("Shape after removing Outliers from dataframe: ",cleaned_data.shape)
```

```
Shape of Original Dataframe with outliers :  (782, 9)
Shape after removing Outliers from dataframe:  (602, 9)
<Figure size 1000x700 with 0 Axes>
```



```
plt.figure(figsize=(10,7))
sns.heatmap(numericals.corr(),annot=True)
```

```
<Axes: >
```

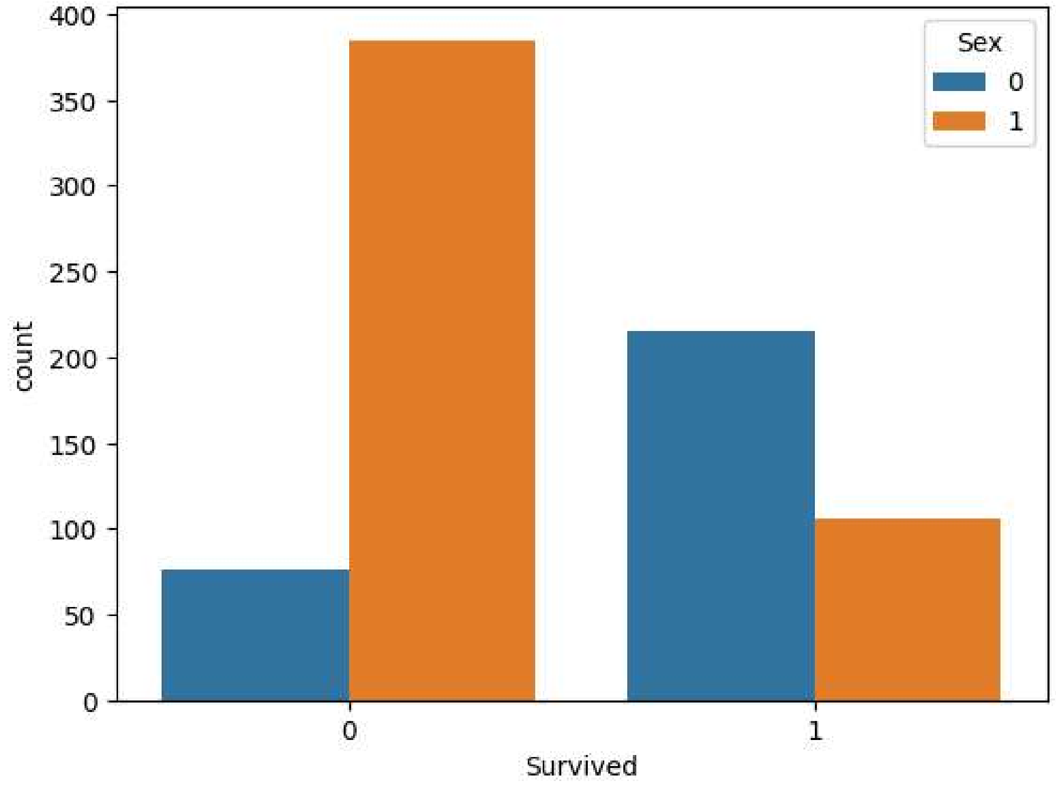


```
df.columns
```

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

```
sns.countplot(x=df["Survived"],hue=df["Sex"]) # Sex : 0-> Female , 1 -> Male.
```

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



Sex : 0-> Female , 1 -> Male.

Survived : 0-> Unsurvived , 1 -> Survived.

Here , Males Survival Rate is very Less as Compared to Females

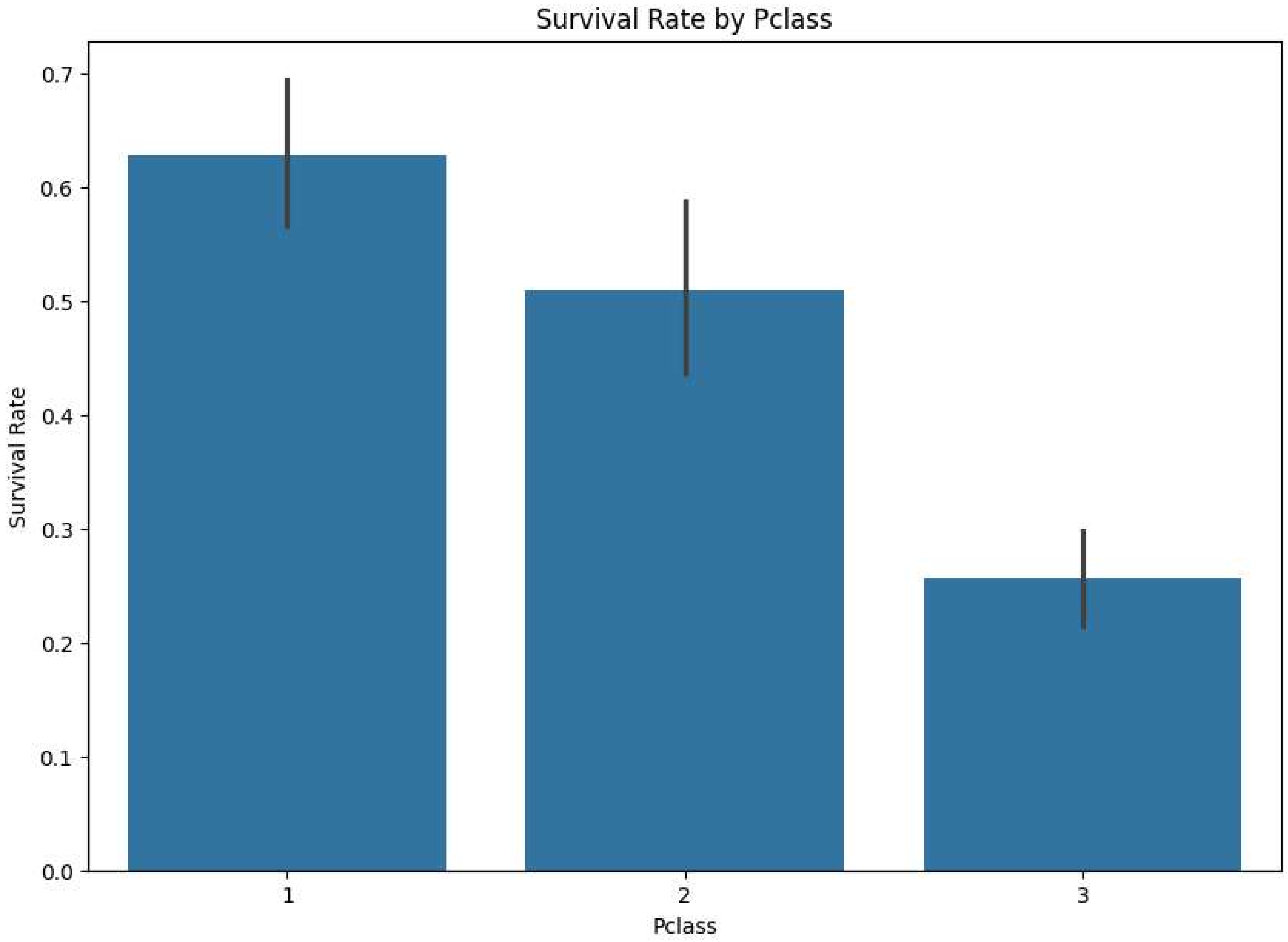
```
df1 = numericals
df1.head()
```

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

Next steps: Generate code with df1 View recommended plots

```
col = ['Age','Fare']
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df1[col] = scaler.fit_transform(df1[col]) # Feature Scaling -> Standarization
```

```
plt.figure(figsize=(10, 7))
sns.barplot(data = df1 , x='Pclass', y='Survived')
plt.title('Survival Rate by Pclass')
plt.xlabel('Pclass')
plt.ylabel('Survival Rate')
plt.show()
```



Observations:

In PClass -> Class-1 Survival Rate is High..