In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [3]:

```
train=pd.read_csv(r'C:\Users\Pranav\Desktop\Prachi\titanic-dataset\titanic\train.csv')
```

In [4]:

```
train.head(5)
```

Out[4]:

	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

In [5]:

```
test=pd.read_csv(r'C:\Users\Pranav\Desktop\Prachi\titanic-dataset\titanic\test.csv')
```

In [6]:

```
test.head(5)
```

Out[6]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [7]:

```
train.info()
#to check where are null values
```

```
<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 [8]:

```
row=train.shape[0]
r_null=train.isnull()
r_null #to check where are null values
```

Out[8]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	False	False	False	False	False	False	False	False	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True	False
887	False	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True	False
889	False	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True	False

891 rows × 12 columns

In [9]:

```
r_null=train.isnull().sum()
r_null
```

Out[9]:

```
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 [10]:

```
r_null/row*100  #converting data to percentage of null values
```

Out[10]:

```
PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            19.865320
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.000000
Cabin          77.104377
Embarked        0.224467
dtype: float64
```

In [11]:

```
train=train.drop(columns=['Cabin'],axis=1) #to drop null values columns #axis=1 is for columns 0 is for rows
```

In [12]:

```
train.head()
```

Out[12]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
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	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S

In [13]:

```
r_null/row*100
```

Out[13]:

```
PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            19.865320
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.000000
Cabin          77.104377
Embarked        0.224467
dtype: float64
```

In [14]:

```
train.isnull().sum()
```

Out[14]:

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Embarked         2
dtype: int64
```

In [15]:

```
train['Age']=train['Age'].fillna(train['Age'].mean())  #filling the null values by mean value of age column
```

In [16]:

```
train.isnull().sum()
```

Out[16]:

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

In [17]:

```
#train['fare']=train['fare'].fillna(train['fare'].ffill) #by using forward fill we are filling null value by upper values
```

In [18]:

```
train['Embarked']=train['Embarked'].fillna(train['Embarked'].mode()[0])# using mode function to fill categorical values
```

In [19]:

```
train.isnull().sum() # dat
```

Out[19]:

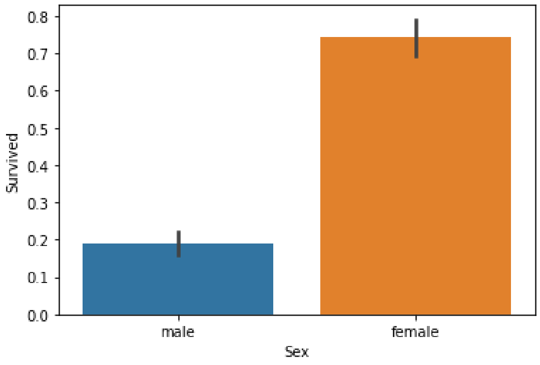
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [20]:

```
#EDA
sns.barplot(y=train['Survived'],x=train['Sex'])
```

Out[20]:

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



In [21]:

```
train
```

Out[21]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.000000	1	0	A/5 21171	7.2500	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.000000	1	0	PC 17599	71.2833	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.000000	0	0	STON/O2. 3101282	7.9250	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.000000	1	0	113803	53.1000	S
4	5	0	3	Allen, Mr. William Henry	male	35.000000	0	0	373450	8.0500	S
...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.000000	0	0	211536	13.0000	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.000000	0	0	112053	30.0000	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	29.699118	1	2	W./C. 6607	23.4500	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.000000	0	0	111369	30.0000	C
890	891	0	3	Dooley, Mr. Patrick	male	32.000000	0	0	370376	7.7500	Q

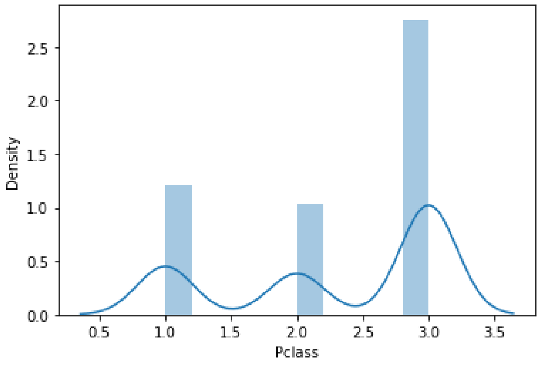
891 rows × 11 columns

In [22]:

```
sns.distplot(train['Pclass'])
```

Out[22]:

```
<AxesSubplot:xlabel='Pclass', ylabel='Density'>
```

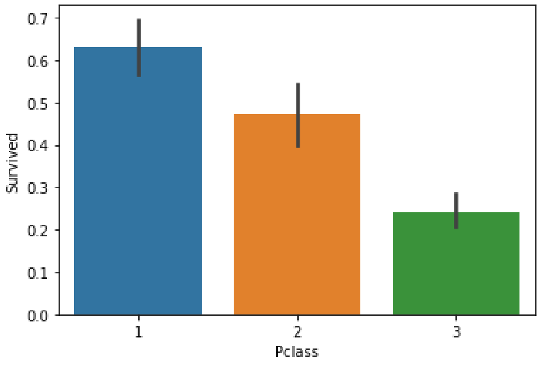


In [23]:

```
sns.barplot(y=train['Survived'],x=train['Pclass'], label='Sex')
```

Out[23]:

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

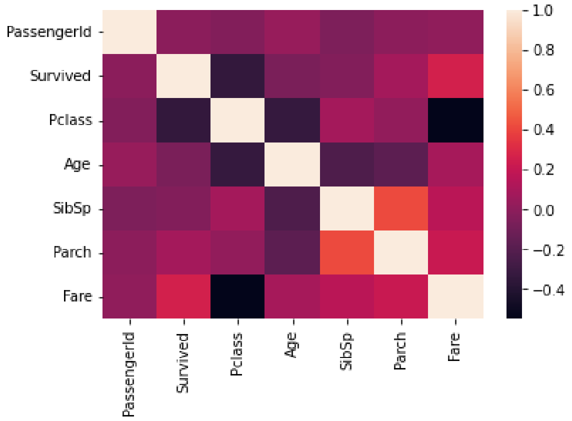


In [24]:

```
sns.heatmap(train.corr(),cbar=True)
```

Out[24]:

```
<AxesSubplot:>
```

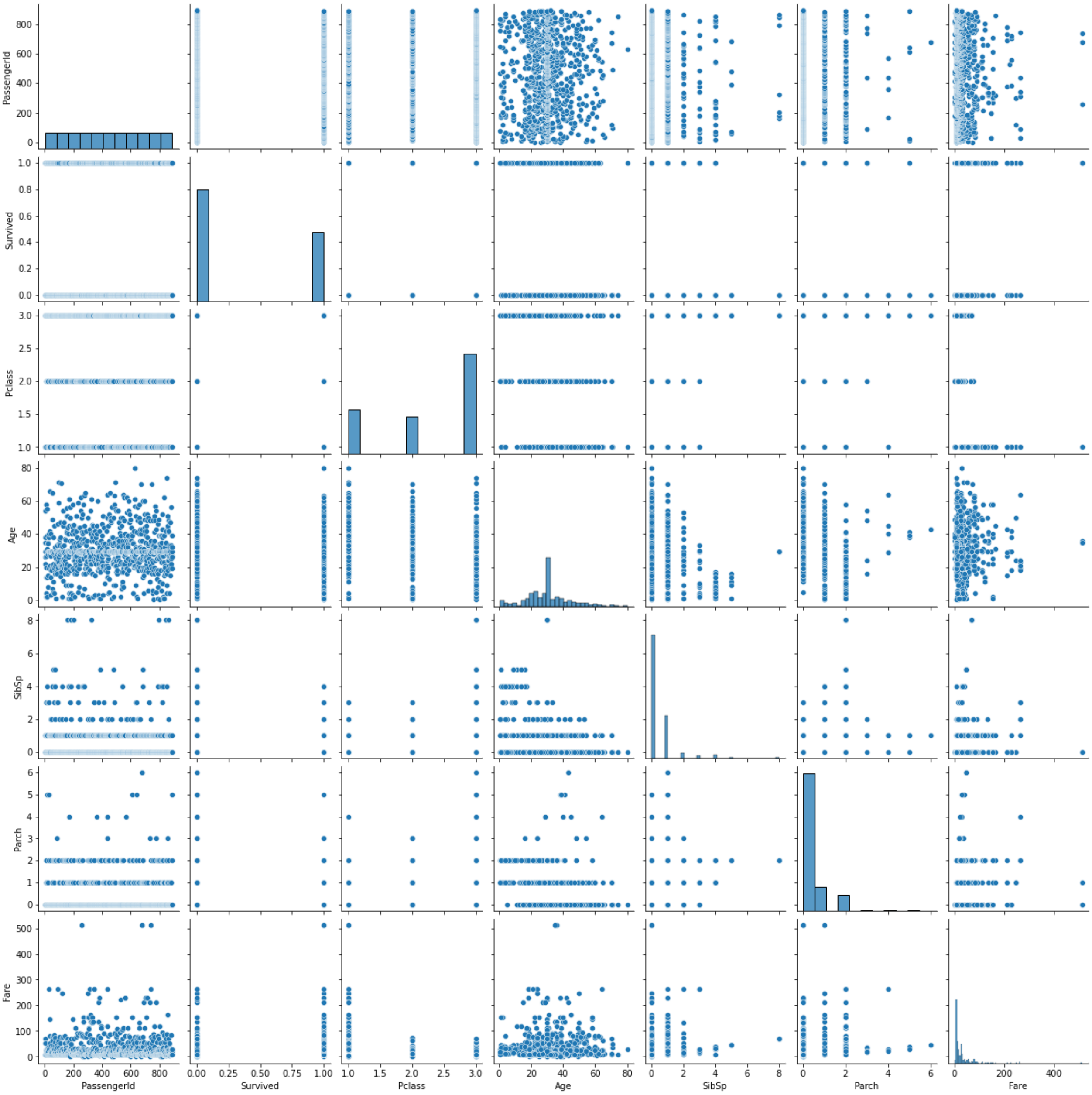


In [25]:

```
sns.pairplot(train)
```

Out[25]:

```
<seaborn.axisgrid.PairGrid at 0x22cef22ed30>
```



In [26]:

```
train.to_csv('Cleandataset-Train')
```

In [27]:

```
test=pd.read_csv(r'C:\Users\Pranav\Desktop\Prachi\titanic-dataset\titanic\test.csv')
```

In [28]:

```
test.head()
```

Out[28]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [29]:

```
test.info()
#to check where are null values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  418 non-null    int64  
 1   Survived     418 non-null    int64  
 2   Pclass       418 non-null    int64  
 3   Name         418 non-null    object 
 4   Sex          418 non-null    object 
 5   Age          332 non-null    float64
 6   SibSp        418 non-null    int64  
 7   Parch        418 non-null    int64  
 8   Ticket       418 non-null    object 
 9   Fare         417 non-null    float64
 10  Cabin        91 non-null     object 
 11  Embarked     418 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 39.3+ KB
```

In [30]:

```
row=test.shape[0]
r_null=test.isnull().sum()
r_null #to check where are null values
```

Out[30]:

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [31]:

```
r_null/row*100  #converting data to percentage of null values
```

Out[31]:

```
PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            20.574163
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.239234
Cabin          78.229665
Embarked        0.000000
dtype: float64
```

In [32]:

```
test=test.drop(columns=['Cabin'],axis=1) #to drop null values columns #axis=1 is for columns 0 is for rows
```

In [33]:

```
r_null/row*100
```

Out[33]:

```
PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            20.574163
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.239234
Cabin          78.229665
Embarked        0.000000
dtype: float64
```

In [34]:

```
test['Age']=test['Age'].fillna(test['Age'].mean())  #filling the null values by mean value of age column
```

In [35]:

```
#test['embarked']=test['embarked'].fillna(test['embarked'].ffill)
test
r_null/row*100
```

Out[35]:

```
PassengerId     0.000000
Survived        0.000000
Pclass          0.000000
Name            0.000000
Sex             0.000000
Age            20.574163
SibSp           0.000000
Parch           0.000000
Ticket          0.000000
Fare            0.239234
Cabin          78.229665
Embarked        0.000000
dtype: float64
```

In [36]:

```
test['Fare']=train['Fare'].fillna(train['Fare'].mean()) #by using forward fill we are filling null value by upper values
```

In [37]:

```
test
```

Out[37]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	892	0	3	Kelly, Mr. James	male	34.50000	0	0	330911	7.2500	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.00000	1	0	363272	71.2833	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.00000	0	0	240276	7.9250	Q
3	895	0	3	Wirz, Mr. Albert	male	27.00000	0	0	315154	53.1000	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.00000	1	1	3101298	8.0500	S
...	...	...	...	...	...	...	...	...	...	...	...
413	1305	0	3	Spector, Mr. Woolf	male	30.27259	0	0	A.5. 3236	0.0000	S
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.00000	0	0	PC 17758	7.9250	C
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	38.50000	0	0	SOTON/O.Q. 3101262	8.0500	S
416	1308	0	3	Ware, Mr. Frederick	male	30.27259	0	0	359309	32.5000	S
417	1309	0	3	Peter, Master. Michael J	male	30.27259	1	1	2668	13.0000	C

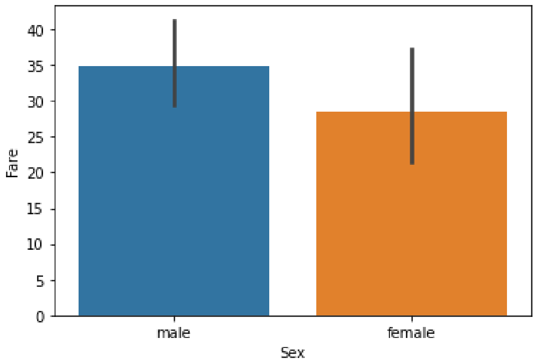
418 rows × 11 columns

In [38]:

```
#EDA
sns.barplot(y=test['Fare'],x=test['Sex'])
```

Out[38]:

```
<AxesSubplot:xlabel='Sex', ylabel='Fare'>
```

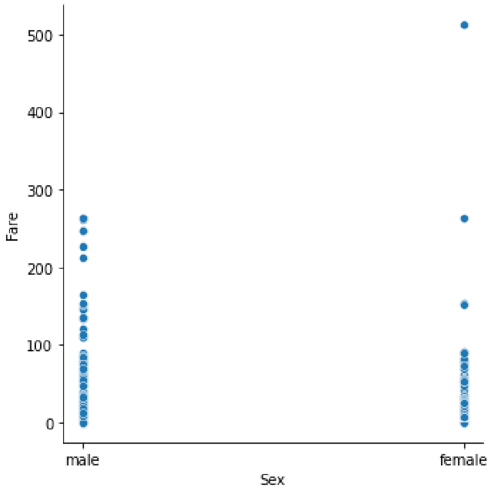


In [39]:

```
sns.relplot(
    data=test,
    x="Sex", y="Fare")
```

Out[39]:

```
<seaborn.axisgrid.FacetGrid at 0x22cf249bbb0>
```

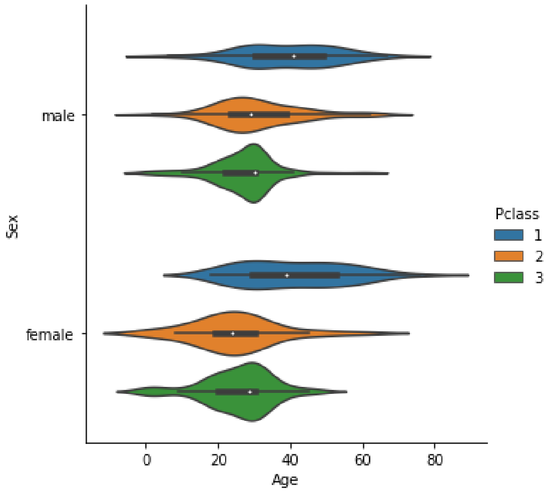


In [40]:

```
sns.catplot(data=test, kind="violin", x="Age", y="Sex",hue="Pclass")
```

Out[40]:

```
<seaborn.axisgrid.FacetGrid at 0x22cf2b473a0>
```

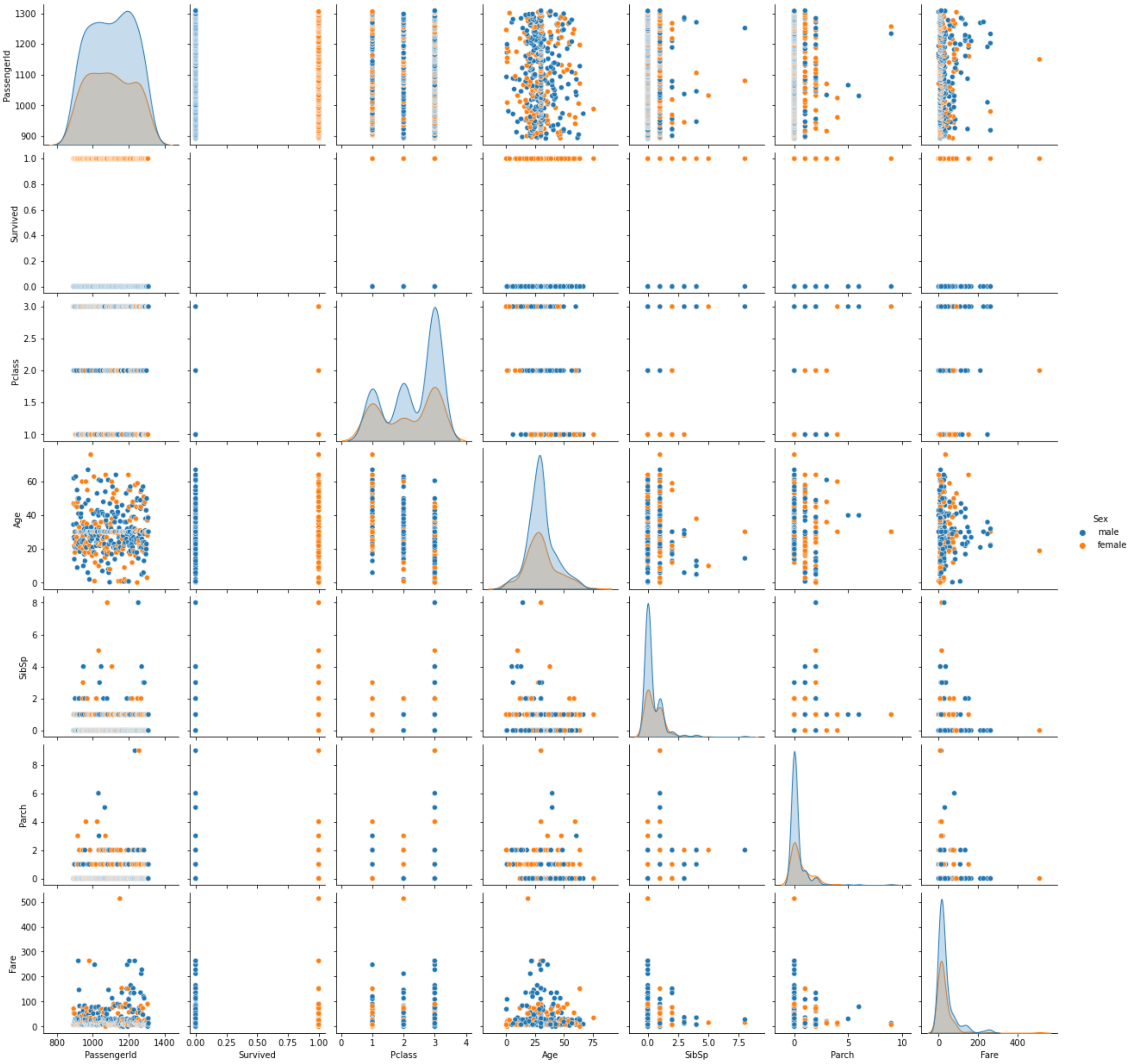


In [41]:

```
sns.pairplot(data=test, hue="Sex")
```

Out[41]:

```
<seaborn.axisgrid.PairGrid at 0x22cf2c6a3d0>
```



In [52]:

```
#try for 3d graph
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.colors import ListedColormap

fig = plt.figure(figsize=(6,6))
ax = Axes3D(fig, auto_add_to_figure=False)
fig.add_axes(ax)

cmap = ListedColormap(sns.color_palette("husl", 256).as_hex())

ax.scatter(x='Survived',y='Sex',z='Age', data=train, marker='o', cmap=cmap, alpha=1)
plt.legend(*sc.legend_elements(), bbox_to_anchor=(1.05, 1), loc=2)
```

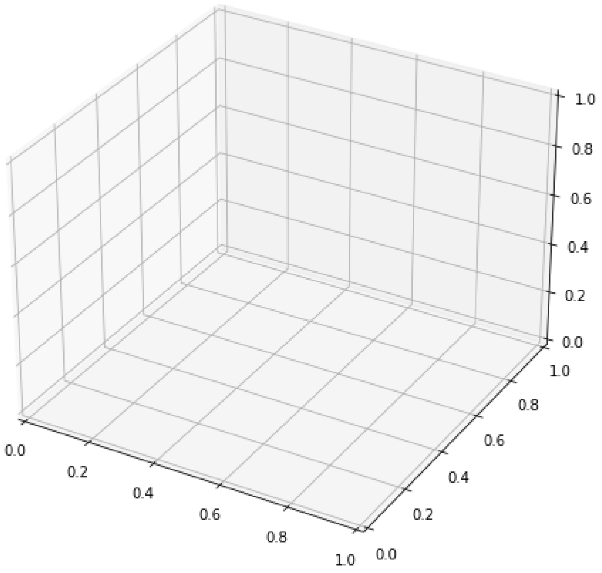
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Input In [52], in <cell line: 11>()
      7 fig.add_axes(ax)
      9 cmap = ListedColormap(sns.color_palette("husl", 256).as_hex())
---> 11 ax.scatter(x='Survived',y='Sex',z='Age', data=train, marker='o', cmap=cmap, alpha=1)
     12 plt.legend(*sc.legend_elements(), bbox_to_anchor=(1.05, 1), loc=2)

File ~\anaconda3\lib\site-packages\matplotlib\__init__.py:1414, in _preprocess_data.<locals>.inner(ax, data, *args,
**kwargs)
   1411 if data is None:
   1412     return func(ax, *map(sanitize_sequence, args), **kwargs)
-> 1414 bound = new_sig.bind(ax, *args, **kwargs)
   1415 auto_label = (bound.arguments.get(label_namer)
   1416               or bound.kwargs.get(label_namer))
   1418 for k, v in bound.arguments.items():

File ~\anaconda3\lib\inspect.py:3045, in Signature.bind(self, *args, **kwargs)
   3040 def bind(self, /, *args, **kwargs):
   3041     """Get a BoundArguments object, that maps the passed `args`
   3042     and `kwargs` to the function's signature.  Raises `TypeError`
   3043     if the passed arguments can not be bound.
   3044     """
-> 3045     return self._bind(args, kwargs)

File ~\anaconda3\lib\inspect.py:2960, in Signature._bind(self, args, kwargs, partial)
   2958                 msg = 'missing a required argument: {arg!r}'
   2959                 msg = msg.format(arg=param.name)
-> 2960                 raise TypeError(msg) from None
   2961 else:
   2962     # We have a positional argument to process
   2963     try:

TypeError: missing a required argument: 'xs'
```



```
In [ ]:
```