In [1]:
```
import sys
print (sys.executable)
```

```
C:\Users\abhis\AppData\Local\Programs\Python\Python313\python.exe
```

In [2]:
```
!"C:\\Users\\abhis\\AppData\\Local\\Programs\\Python\\Python313\\python.exe" -m pip install matplotlib seaborn
```

```
Requirement already satisfied: matplotlib in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (3.10.7)
Requirement already satisfied: seaborn in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (0.13.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotli
b) (1.3.3)
Requirement already satisfied: cycler>=0.10 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotlib)
(0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotli
b) (4.60.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotli
b) (1.4.9)
Requirement already satisfied: numpy>=1.23 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotlib) (2.
2.5)
Requirement already satisfied: packaging>=20.0 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotlib)
(25.0)
Requirement already satisfied: pillow>=8 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotlib) (11.
2.1)
Requirement already satisfied: pyparsing>=3 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplotlib)
(3.2.5)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from matplo
tlib) (2.9.0.post0)
Requirement already satisfied: pandas>=1.2 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from seaborn) (2.3.
3)
Requirement already satisfied: pytz>=2020.1 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from pandas>=1.2->s
eaborn) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from pandas>=1.2-
>seaborn) (2025.2)
Requirement already satisfied: six>=1.5 in c:\users\abhis\appdata\local\programs\python\python313\lib\site-packages (from python-dateutil>=
2.7->matplotlib) (1.17.0)
[notice] A new release of pip is available: 24.3.1 -> 25.3
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [3]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")
```

In [4]:
```
import pandas as pd
df = pd.read_csv("train.csv")
```

```
df.head()
```

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

Out[6]:

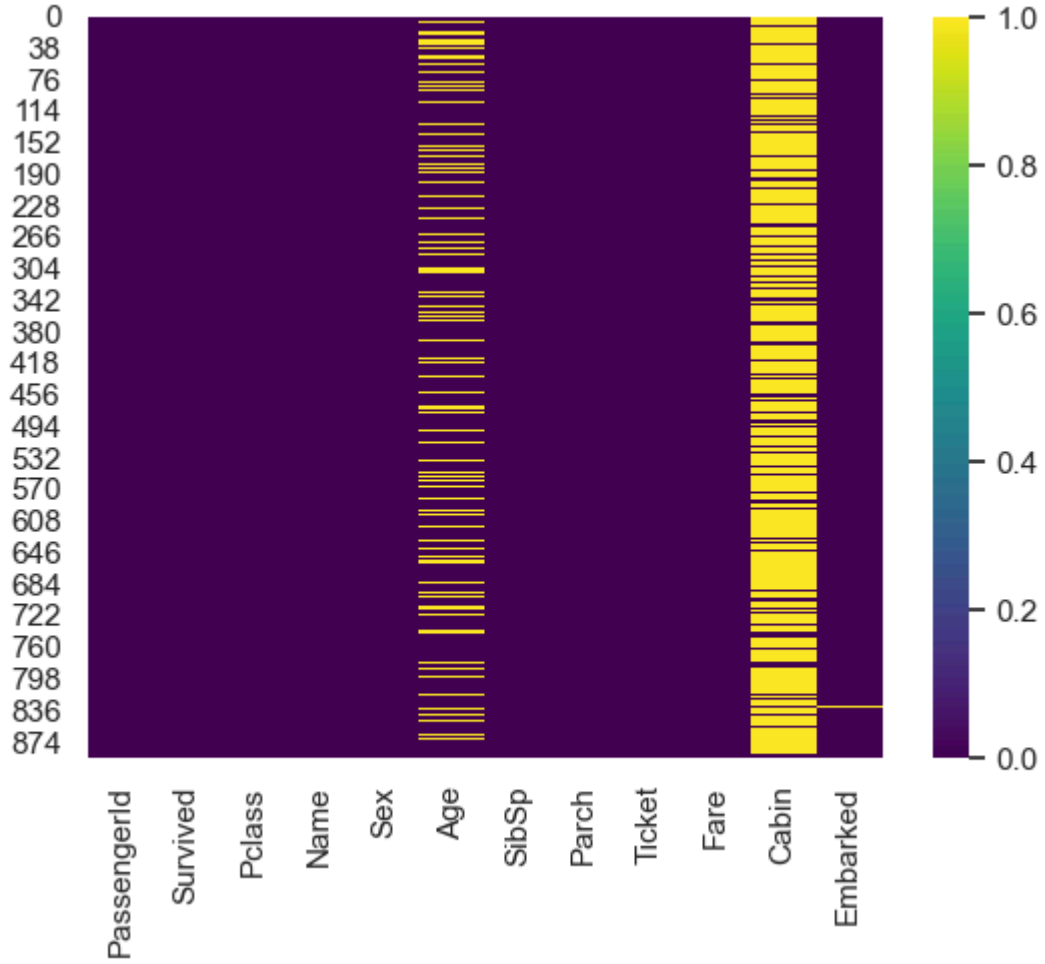
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	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]:
```
df.isnull().sum()
```

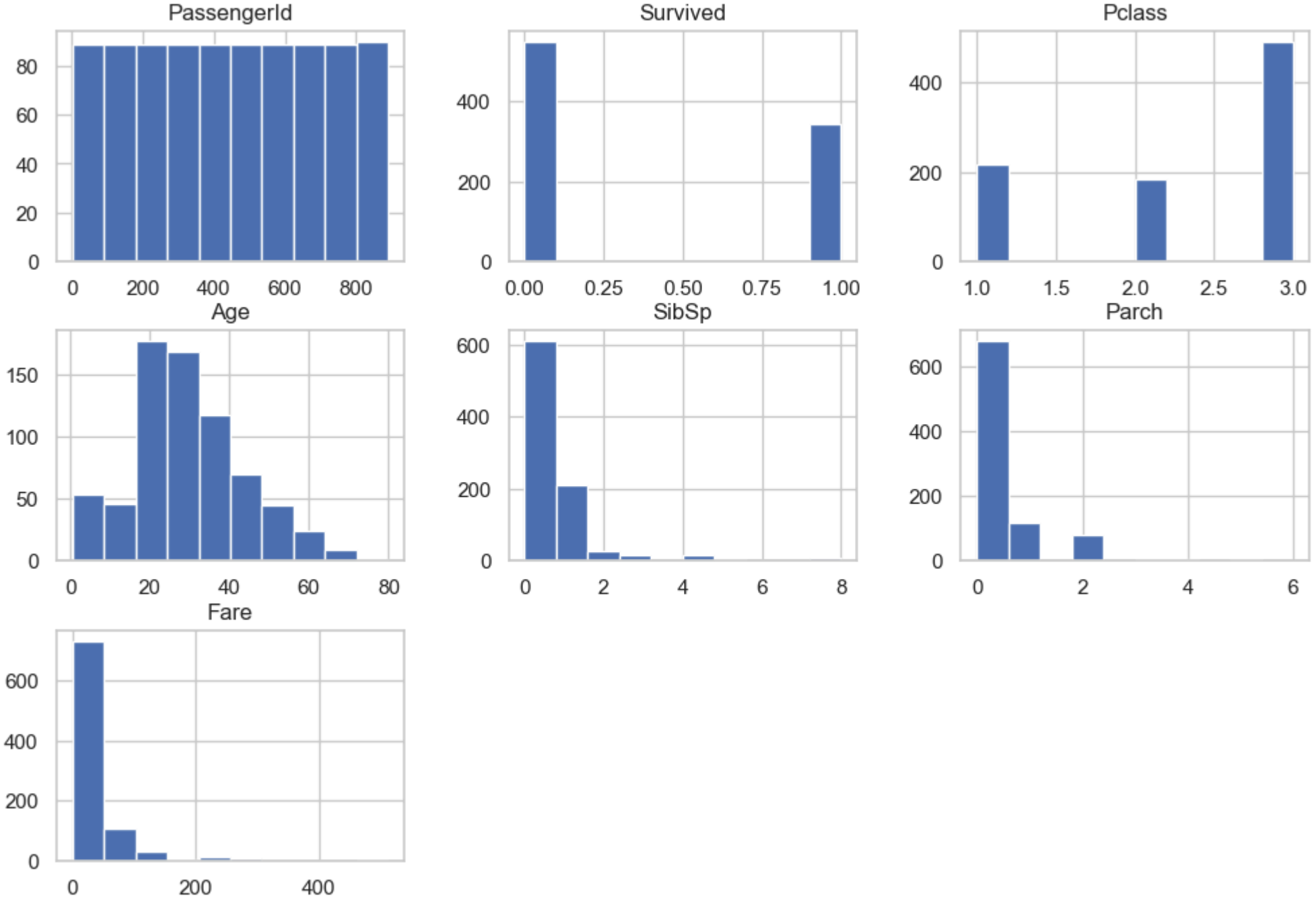
Out[7]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [8]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.heatmap(df.isnull(),cmap="viridis")
```

Out[8]:
```
<Axes: >
```



In [9]:
```
df.hist(figsize=(12,8))
plt.show()
```

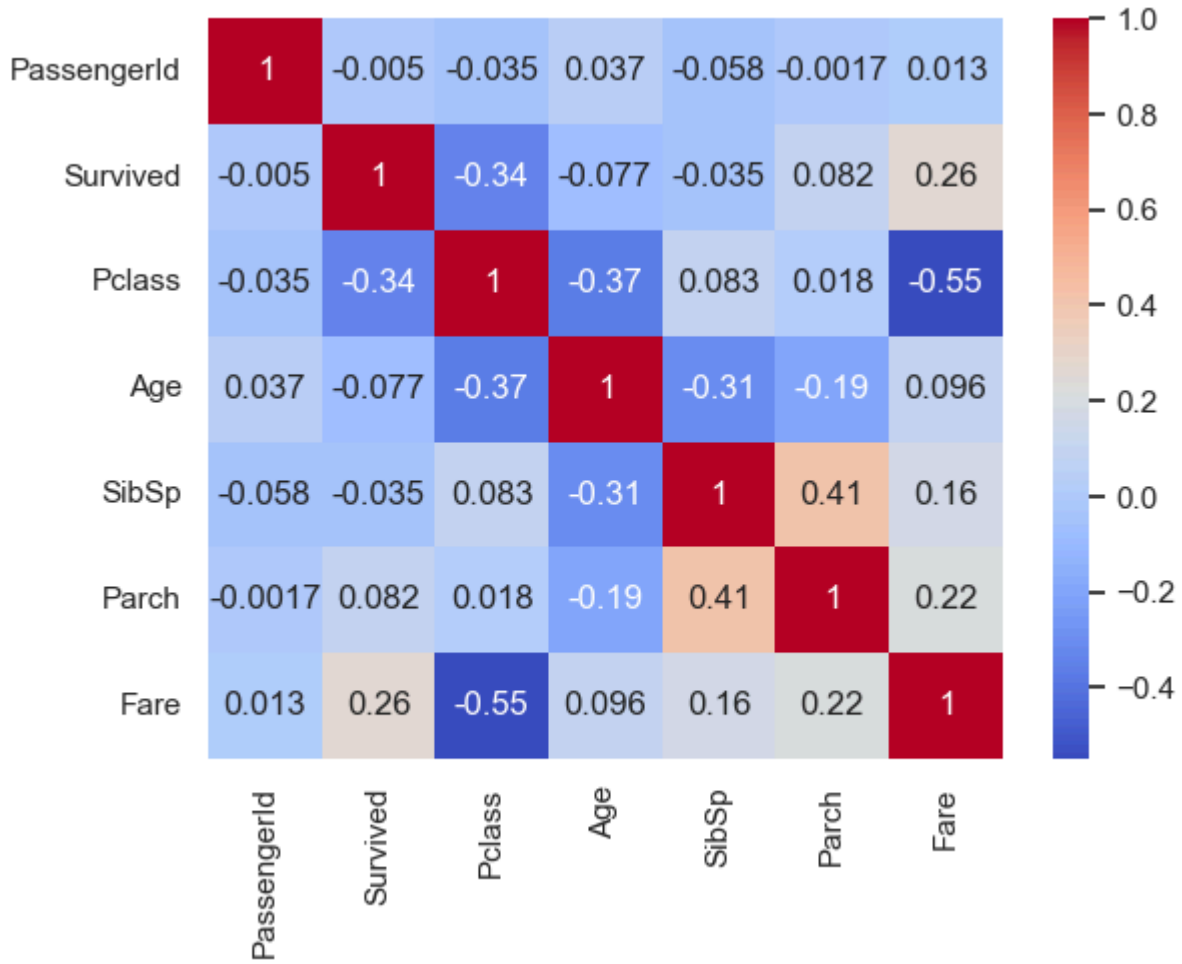


In [10]:
```
sns.heatmap(df.corr(),annot=True,cmap="coolwarm")
```

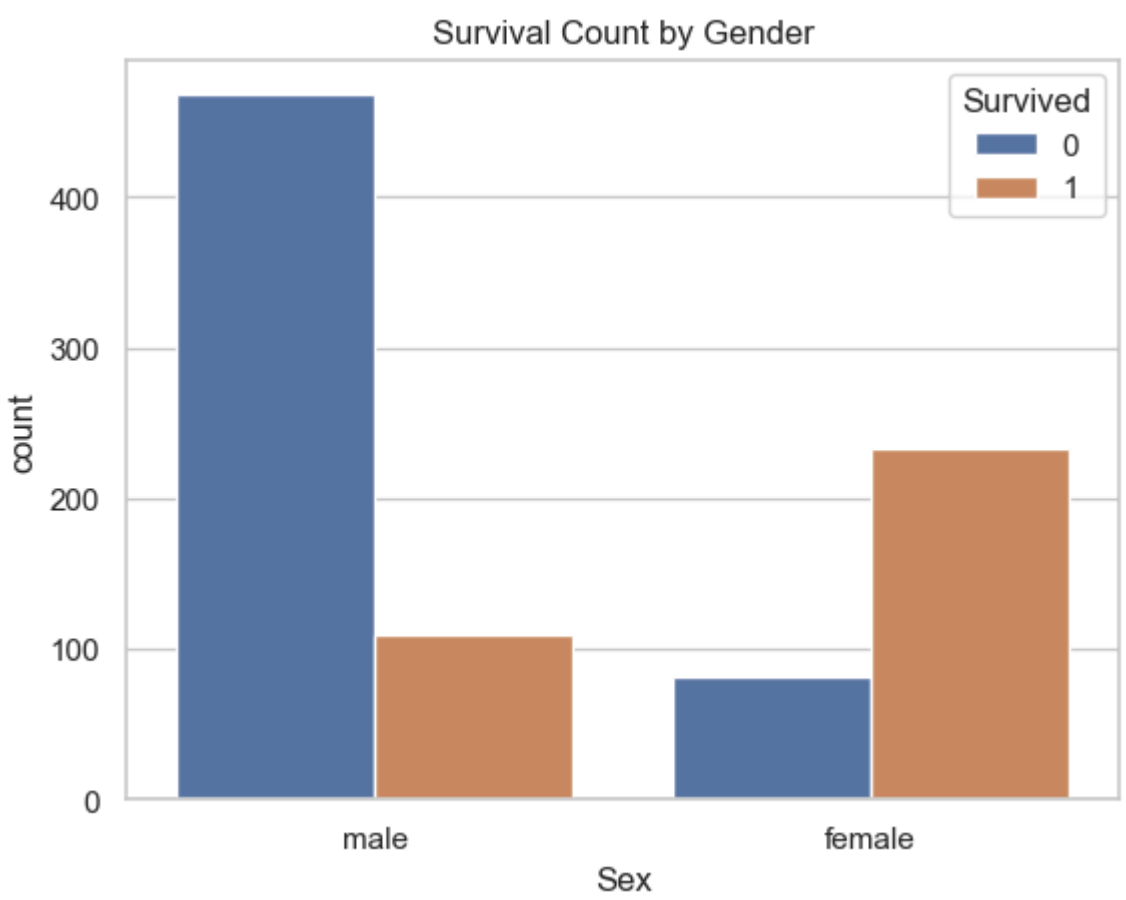
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[10], line 1
----> 1 sns.heatmap(df.corr(),annot=True,cmap="coolwarm")

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\pandas\core\frame.py:11076, in DataFrame.corr(self, method, min_periods, numeric_only)
  11074 cols = data.columns
  11075 idx = cols.copy()
> 11076 mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)
  11078 if method == "pearson":
  11079     correl = libalgos.nancorr(mat, minp=min_periods)

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\pandas\core\frame.py:2002, in DataFrame.to_numpy(self, dtype, copy, na_value)
   2000 if dtype is not None:
   2001     dtype = np.dtype(dtype)
-> 2002 result = self._mgr.as_array(dtype=dtype, copy=copy, na_value=na_value)
   2003 if result.dtype is not dtype:
   2004     result = np.asarray(result, dtype=dtype)

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\pandas\core\internals\managers.py:1713, in BlockManager.as_array(self, dtype, copy, na_value)
   1711         arr.flags.writeable = False
   1712 else:
-> 1713     arr = self._interleave(dtype=dtype, na_value=na_value)
   1714     # The underlying data was copied within _interleave, so no need
   1715     # to further copy if copy=True or setting na_value
   1717 if na_value is lib.no_default:

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\pandas\core\internals\managers.py:1772, in BlockManager._interleave(self, dtype, na_value)
   1770     else:
   1771         arr = blk.get_values(dtype)
-> 1772     result[rl.indexer] = arr
   1773     itemmask[rl.indexer] = 1
   1775 if not itemmask.all():

ValueError: could not convert string to float: 'Braund, Mr. Owen Harris'
```

In [13]:
```
numeric_df = df.select_dtypes(include=['int64','float'])
sns.heatmap(numeric_df.corr(),annot=True,cmap="coolwarm")
```

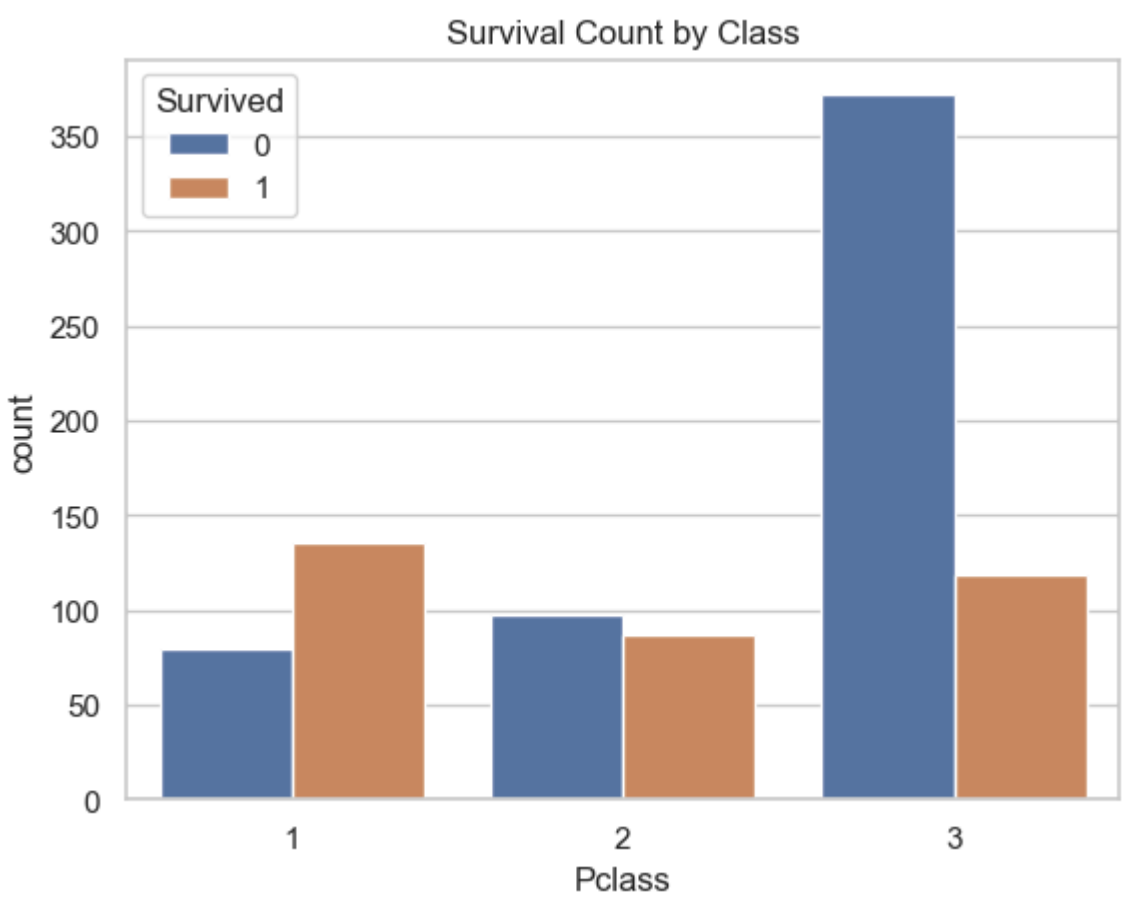
Out[13]: <Axes: >



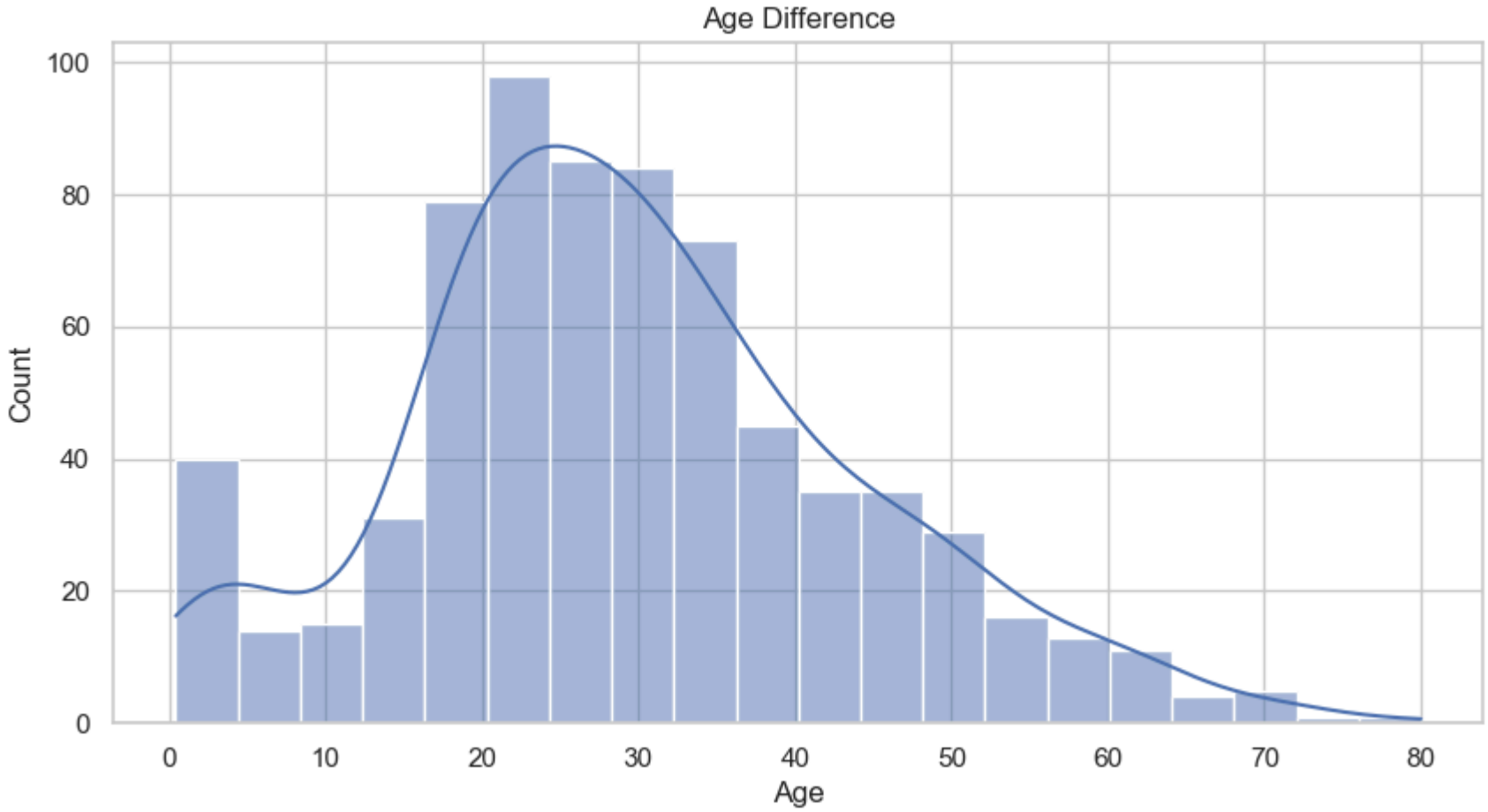
In [16]:
```
sns.countplot(data=df,x="Sex",hue='Survived')
plt.title('Survival Count by Gender')
plt.show()
```



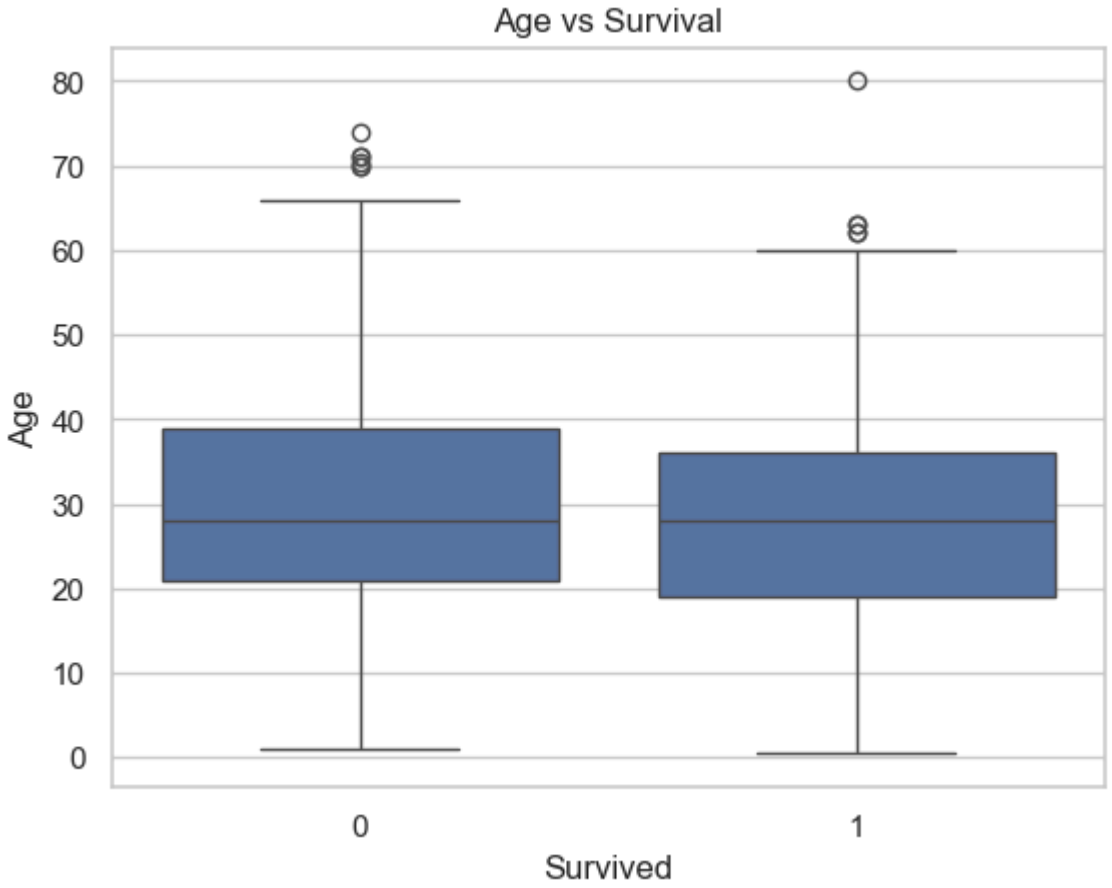
In [17]:
```
sns.countplot(data=df,x="Pclass",hue='Survived')
plt.title('Survival Count by Class')
plt.show()
```



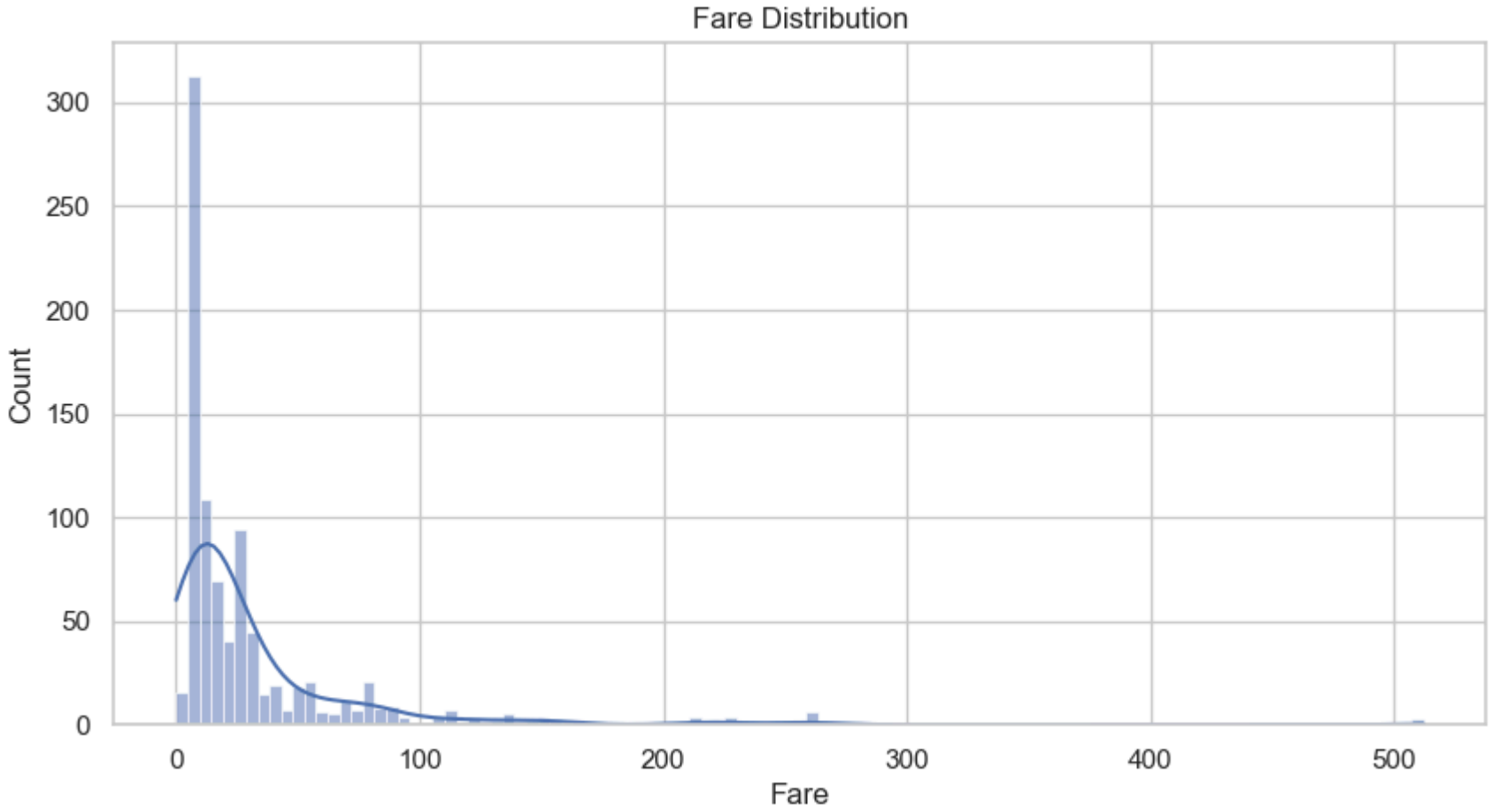
In [19]:
```
plt.figure(figsize=(10,5))
sns.histplot(df['Age'],kde=True)
plt.title('Age Difference')
plt.show()
```



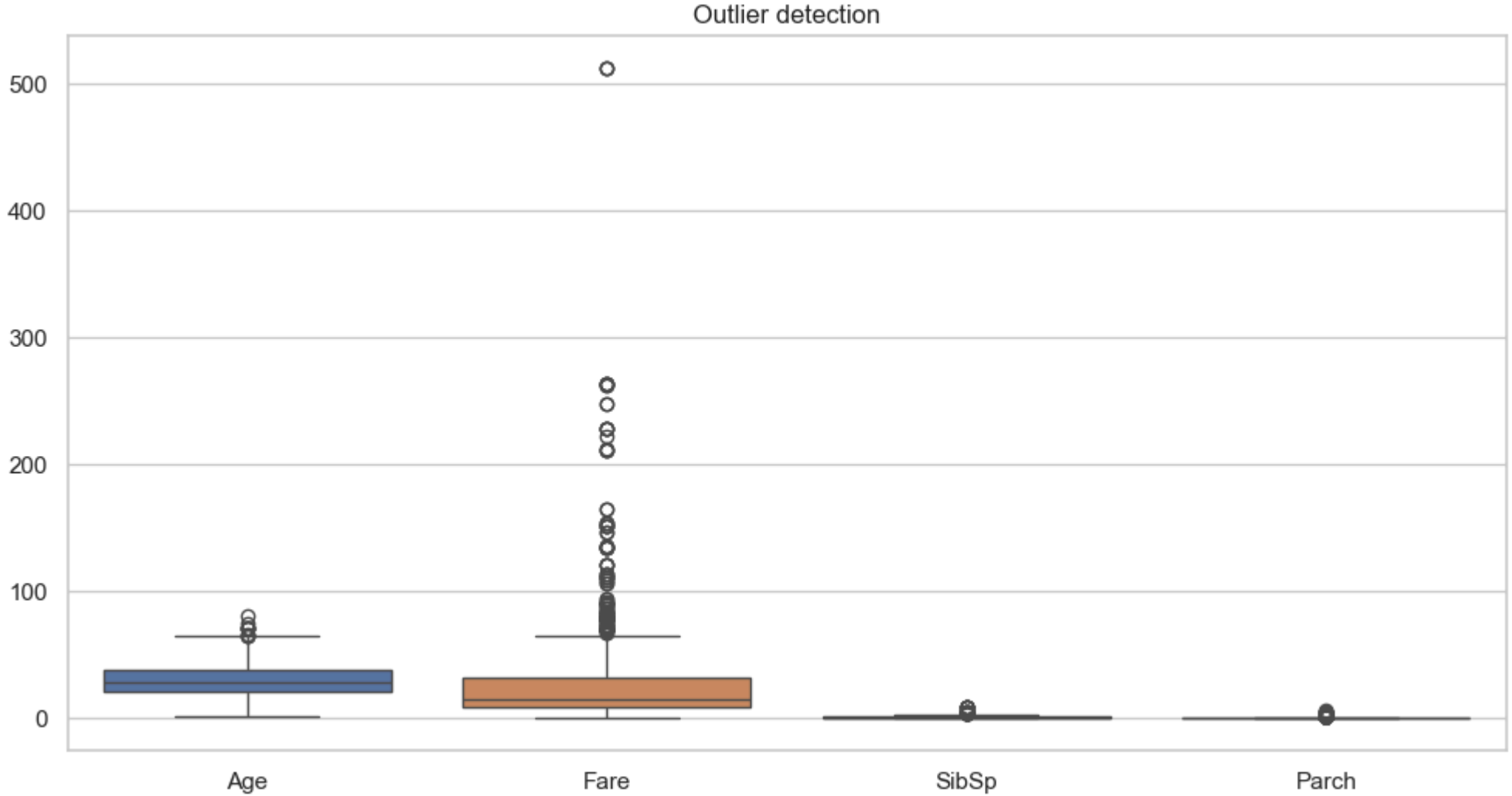
In [20]:
```
sns.boxplot(x='Survived',y='Age',data=df)
plt.title("Age vs Survival")
plt.show()
```



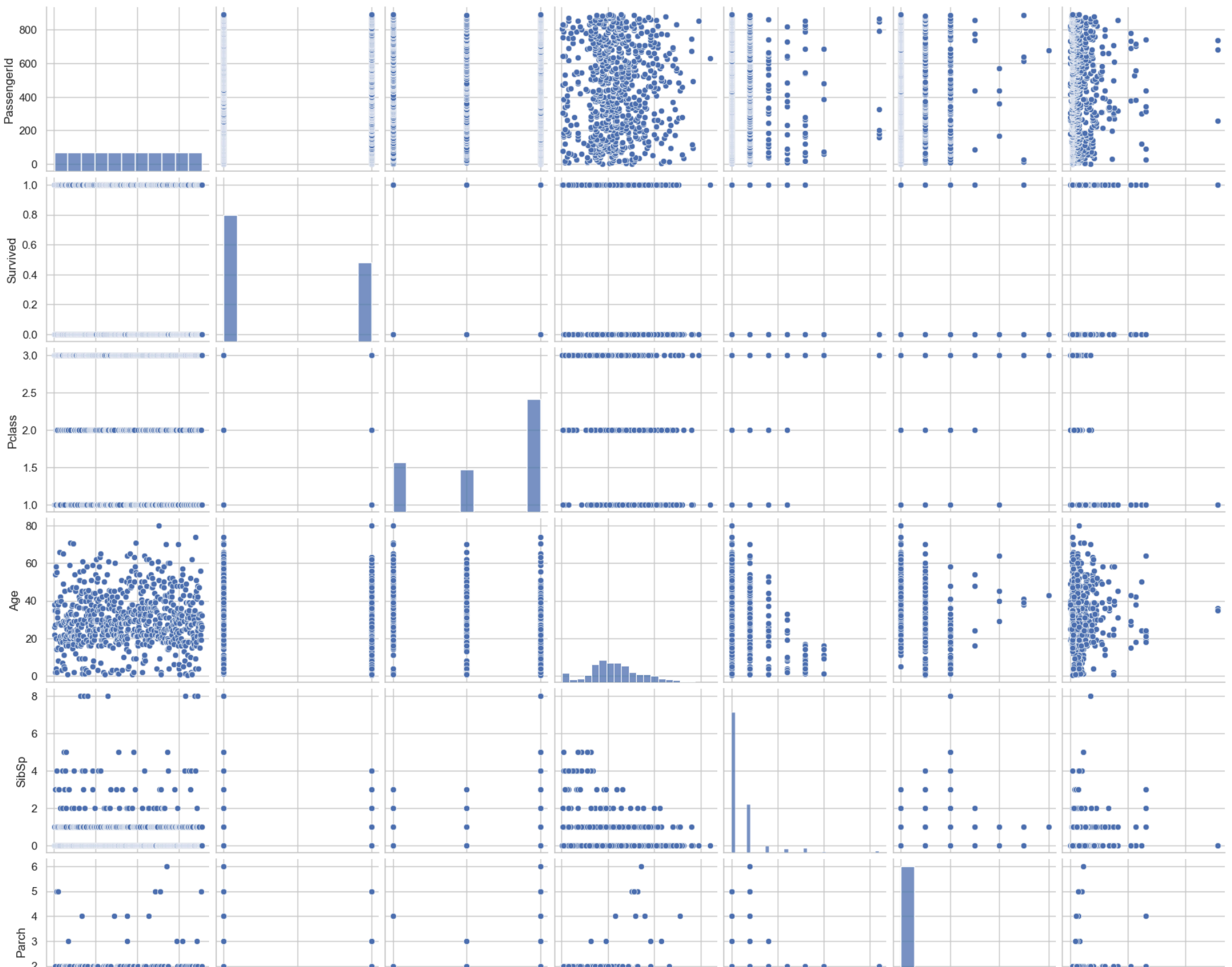
In [22]:
```
plt.figure(figsize=(10,5))
sns.histplot(df['Fare'],kde=True)
plt.title("Fare Distribution")
plt.show()
```

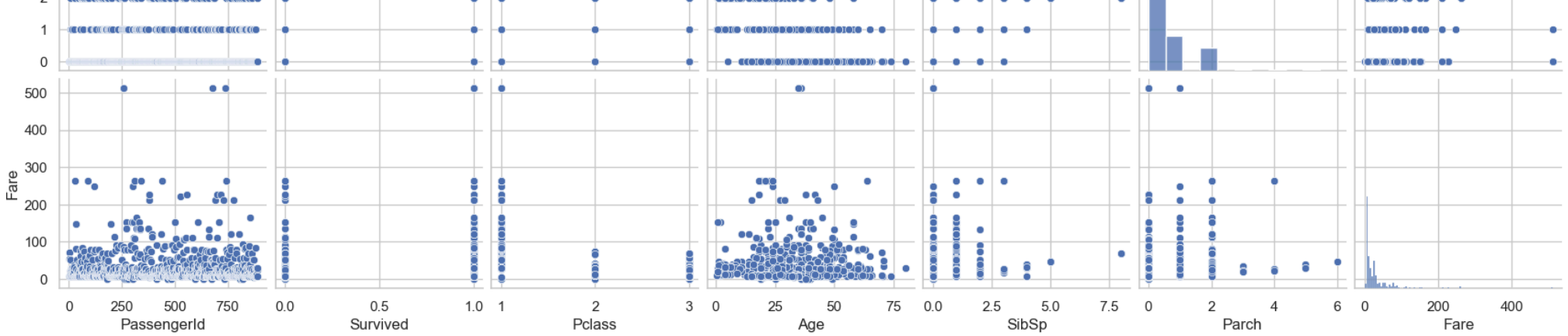


In [23]:
```
plt.figure(figsize=(12,6))
sns.boxplot(data=df[['Age','Fare','SibSp','Parch']])
plt.title("Outlier detection")
plt.show()
```

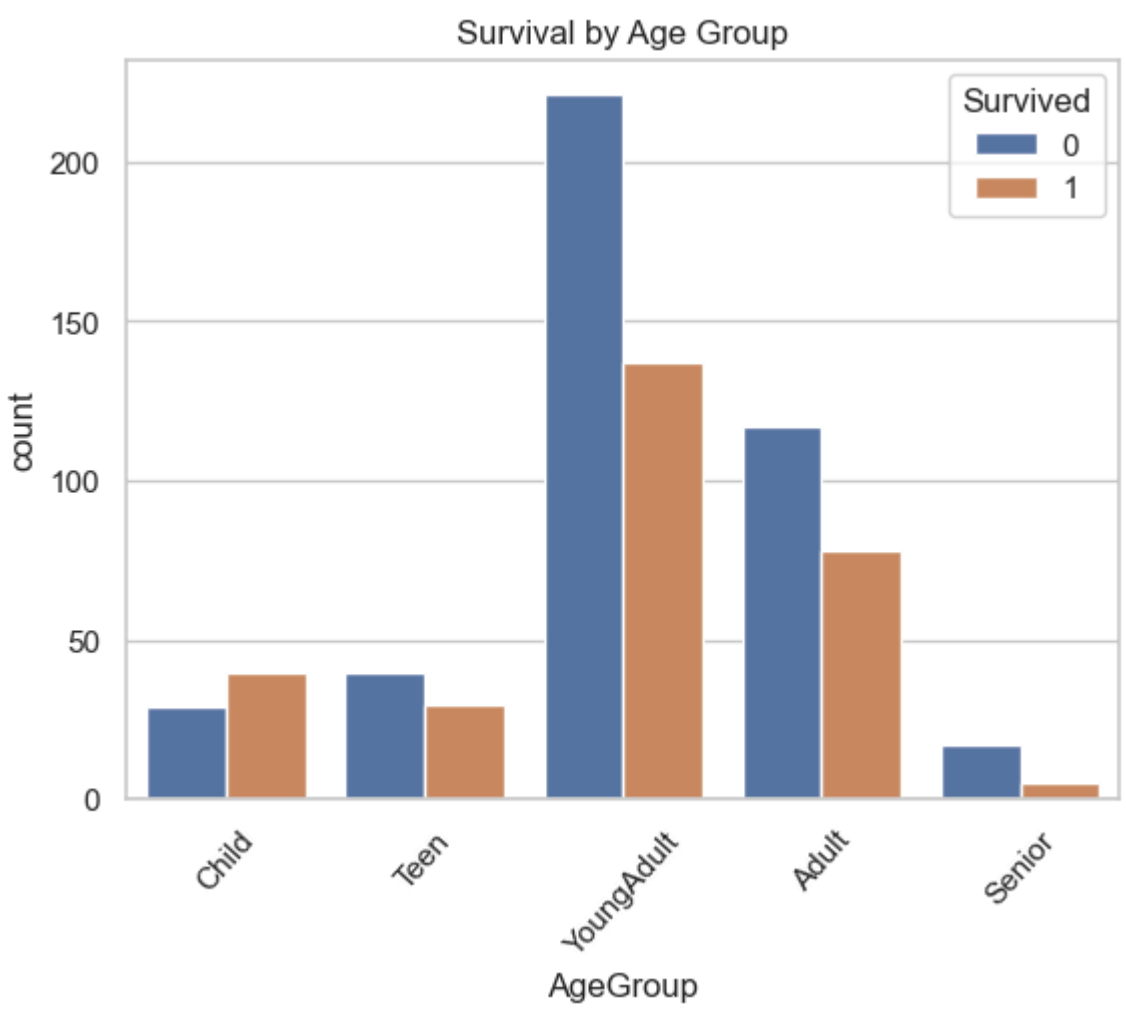


In [24]:
```
sns.pairplot(df.select_dtypes(include="number"))
plt.show()
```


PassengerId
Survived
Pclass
Age
SibSp
Parch



In [27]:
```
df['AgeGroup']=pd.cut(df['Age'],bins=[0,12,18,35,60,100],labels=['Child','Teen','YoungAdult','Adult','Senior'])
sns.countplot(data=df,x='AgeGroup',hue="Survived")
plt.xticks(rotation=48)
plt.title("Survival by Age Group")
plt.show()
```

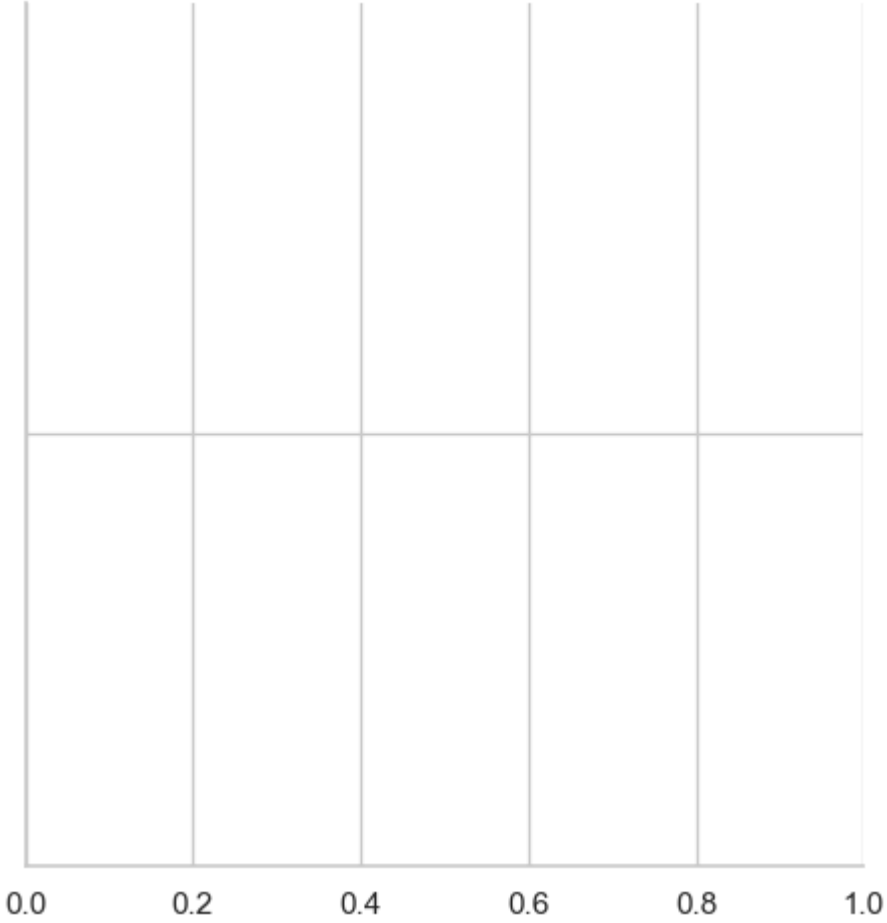


In [29]:
```
df["FamilySize"]=df['SibSp']+df['Parch']+1
sns.countlplot(data=df,x='FamilySize',hue="Survived")
plt.title("Survival by Family Size")
plt.show()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[29], line 2
      1 df["FamilySize"]=df['SibSp']+df['Parch']+1
----> 2 sns.countlplot(data=df,x='FamilySize',hue="Survived")
      3 plt.title("Survival by Family Size")
      4 plt.show()

AttributeError: module 'seaborn' has no attribute 'countlplot'
```

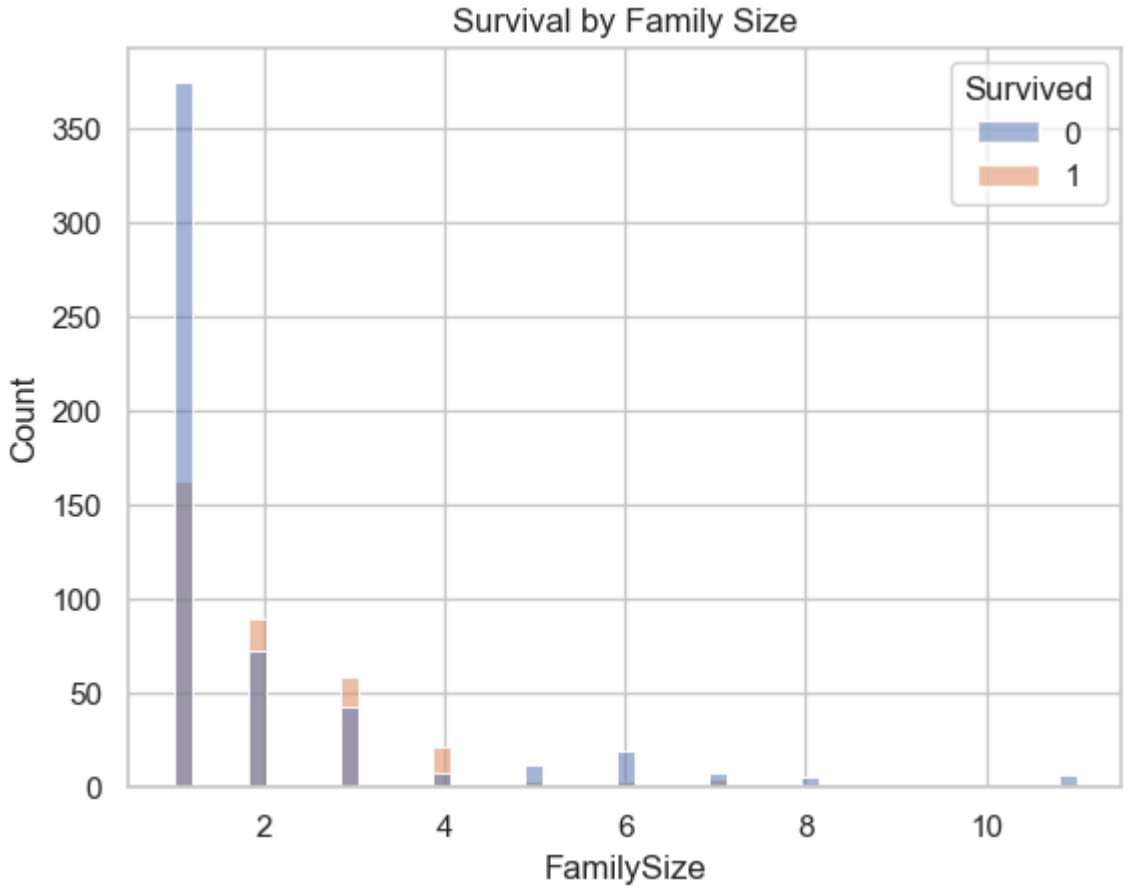
In [31]:
```
sns.catplot(data=df,x="FamilySize",hue="Survived",kind="Count")
plt.title("Survival by Family Size")
plt.show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[31], line 1
----> 1 sns.catplot(data=df,x=             ,hue=          ,kind=       )
      2 plt.title("Survival by Family Size")
      3 plt.show()

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\seaborn\categorical.py:3110, in catplot(data, x, y, hue, row, col, kind, es
timator, errorbar, n_boot, seed, units, weights, order, hue_order, row_order, col_order, col_wrap, height, aspect, log_scale, native_scale,
formatter, orient, color, palette, hue_norm, legend, legend_out, sharex, sharey, margin_titles, facet_kws, ci, **kwargs)
   3105 else:
   3106     msg = (
   3107         f"Invalid `kind`: {kind!r}. Options are 'strip', 'swarm', "
   3108         "'box', 'boxen', 'violin', 'bar', 'count', and 'point'."
   3109     )
-> 3110     raise ValueError(msg)
   3112 for ax in g.axes.flat:
   3113     p._adjust_cat_axis(ax, axis=p.orient)

ValueError: Invalid `kind`: 'Count'. Options are 'strip', 'swarm', 'box', 'boxen', 'violin', 'bar', 'count', and 'point'.
```

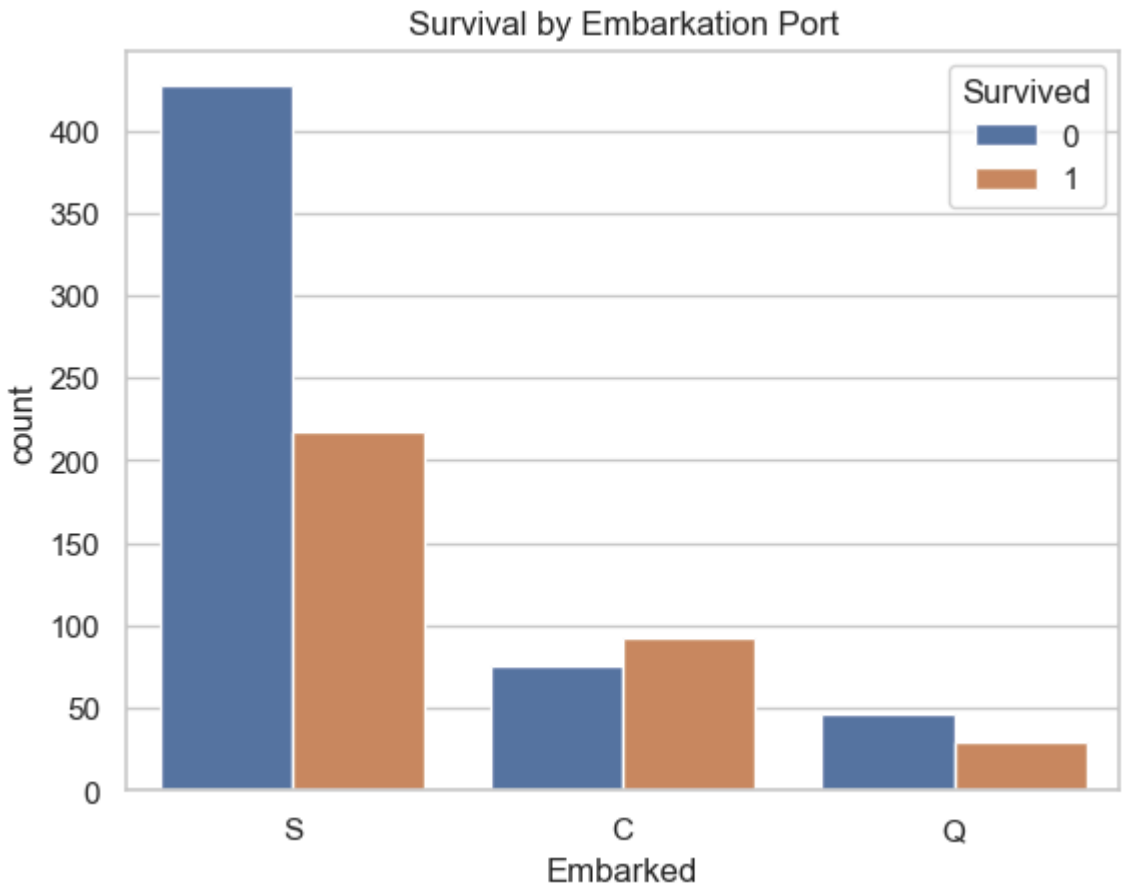


In [34]:
```
df["FamilySize"]=df["SibSp"]+df['Parch']+1
```

In [35]:
```
df["FamilySize"]=df['SibSp']+df['Parch']+1
sns.histplot(data=df,x='FamilySize',hue="Survived")
plt.title("Survival by Family Size")
plt.show()
```



In [36]:
```
sns.countplot(data=df,x='Embarked',hue="Survived")
plt.title("Survival by Embarkation Port")
plt.show()
```



In [38]:
```
df.isnull().sum()
```

Out[38]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
AgeGroup       177
FamilySize       0
dtype: int64
```

In [39]:
```
df['Age']=df['Age'].fillna(df['Age'].median())
```

In [41]:
```
df['Embarked']=df['Embarked'].fillna(df['Embarked'].mode()[0])
```

In [43]:
```
df=df.drop(columns=["Cabin"])
```

In [44]:
```
df['AgeGroup']=pd.cut(df['Age'],bins=[0,12,18,35,60,100],labels=['child','Teen','YoungAdult','Adult','Senior'])
```

In [45]:
```
df.isnull().sum()
```

Out[45]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
AgeGroup       0
FamilySize     0
dtype: int64
```

In [46]:
```
sns.countplot(data=df,x='Sex',hue='Survival')
pls.title("Survival Count by Gender")
plt.show()
```

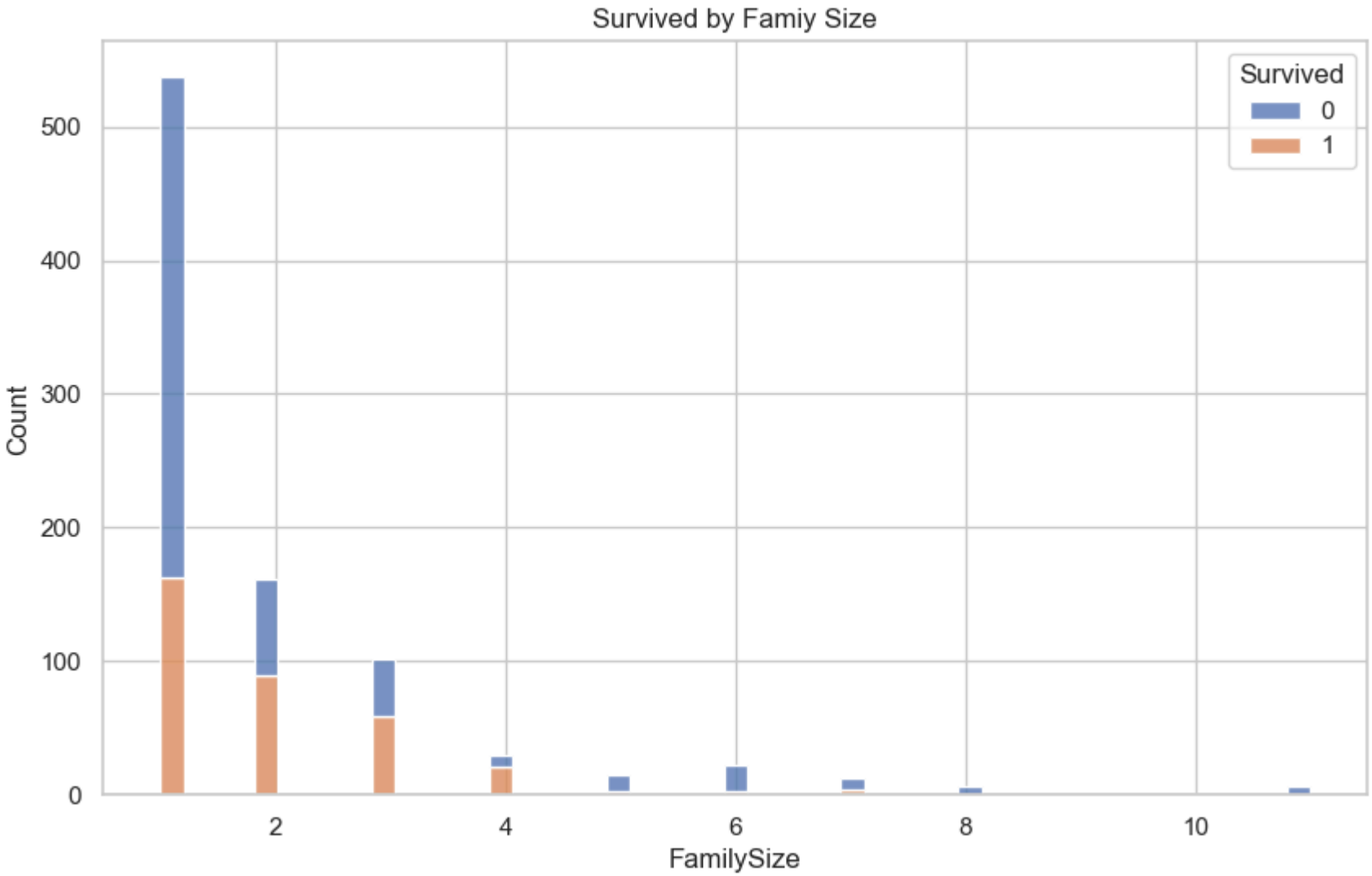
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[46], line 1
----> 1 sns.countplot(data=df,x=     ,hue=          )
      2 pls.title("Survival Count by Gender")
      3 plt.show()

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\seaborn\categorical.py:2631, in countplot(data, x, y, hue, order, hue_orde
r, orient, color, palette, saturation, fill, hue_norm, stat, width, dodge, gap, log_scale, native_scale, formatter, legend, ax, **kwargs)
   2628 elif x is not None and y is not None:
   2629     raise TypeError("Cannot pass values for both `x` and `y`.")
-> 2631 p = _CategoricalAggPlotter(
   2632     data=data,
   2633     variables=dict(x=x, y=y, hue=hue),
   2634     order=order,
   2635     orient=orient,
   2636     color=color,
   2637     legend=legend,
   2638 )
   2640 if ax is None:
   2641     ax = plt.gca()

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\seaborn\categorical.py:67, in _CategoricalPlotter.__init__(self, data, vari
ables, order, orient, require_numeric, color, legend)
     56 def __init__(
     57     self,
     58     data=None,
   (...)     64     legend="auto",
     65 ):
---> 67     super().__init__(data=data, variables=variables)
     69     # This method takes care of some bookkeeping that is necessary because the
     70     # original categorical plots (prior to the 2021 refactor) had some rules that
     71     # don't fit exactly into VectorPlotter logic. It may be wise to have a second
   (...)     76     # default VectorPlotter rules. If we do decide to make orient part of the
     77     # _base variable assignment, we'll want to figure out how to express that.
     78     if self.input_format == "wide" and orient in ["h", "y"]:

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\seaborn\_base.py:634, in VectorPlotter.__init__(self, data, variables)
    629 # var_ordered is relevant only for categorical axis variables, and may
    630 # be better handled by an internal axis information object that tracks
    631 # such information and is set up by the scale_* methods. The analogous
    632 # information for numeric axes would be information about log scales.
    633 self._var_ordered = {"x": False, "y": False}  # alt., used DefaultDict
--> 634 self.assign_variables(data, variables)
    636 # TODO Lots of tests assume that these are called to initialize the
    637 # mappings to default values on class initialization. I'd prefer to
```

```
    638 # move away from that and only have a mapping when explicitly called.
    639 for var in ["hue", "size", "style"]:

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\seaborn\_base.py:679, in VectorPlotter.assign_variables(self, data, variables)
    674 else:
    675     # When dealing with long-form input, use the newer PlotData
    676     # object (internal but introduced for the objects interface)
    677     # to centralize / standardize data consumption logic.
    678     self.input_format = "long"
--> 679     plot_data = PlotData(data, variables)
    680     frame = plot_data.frame
    681     names = plot_data.names

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\seaborn\_core\data.py:58, in PlotData.__init__(self, data, variables)
     51 def __init__(
     52     self,
     53     data: DataSource,
     54     variables: dict[str, VariableSpec],
     55 ):
     57     data = handle_data_source(data)
---> 58     frame, names, ids = self._assign_variables(data, variables)
     60     self.frame = frame
     61     self.names = names

File ~\AppData\Local\Programs\Python\Python313\Lib\site-packages\seaborn\_core\data.py:232, in PlotData._assign_variables(self, data, variables)
    230     else:
    231         err += "An entry with this name does not appear in `data`."
--> 232     raise ValueError(err)
    234 else:
    235
    236     # Otherwise, assume the value somehow represents data
    237
    238     # Ignore empty data structures
    239     if isinstance(val, Sized) and len(val) == 0:

ValueError: Could not interpret value `Survival` for `hue`. An entry with this name does not appear in `data`.
```

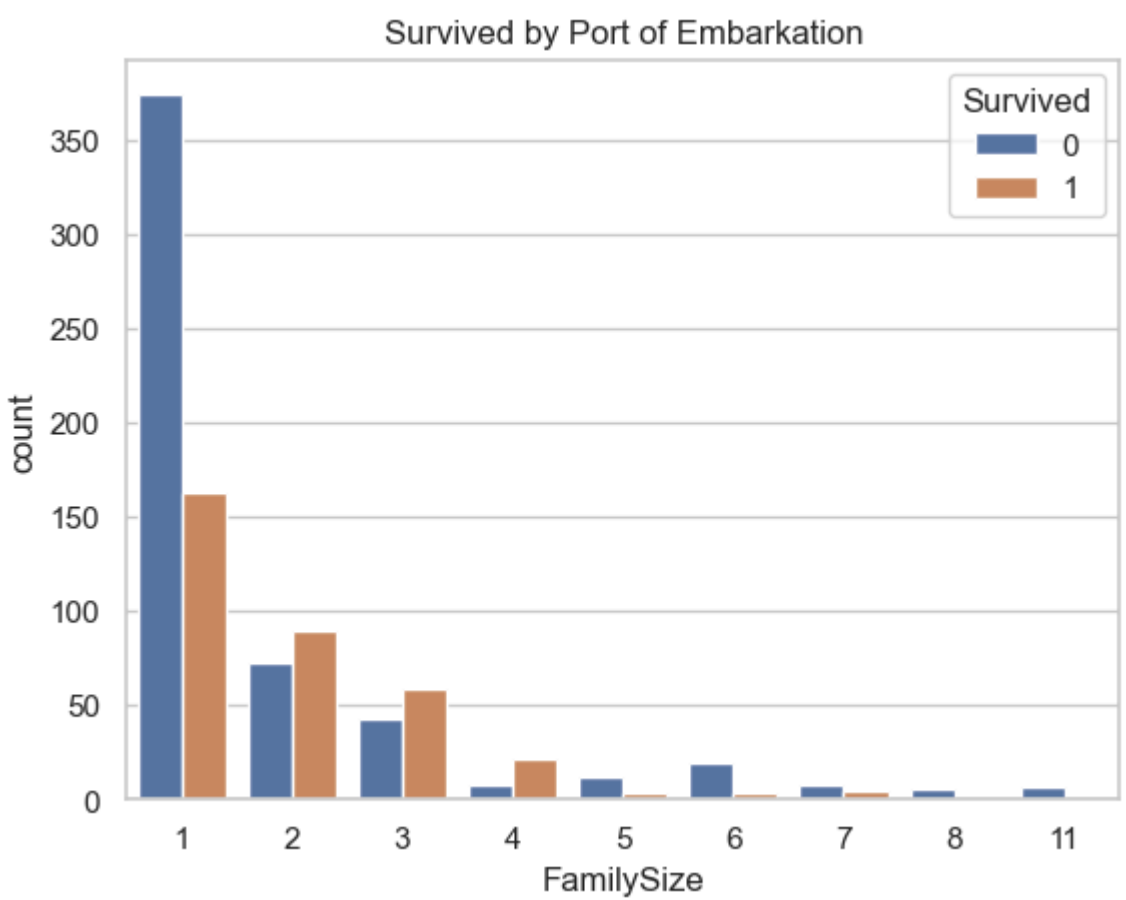
In [47]:
```
df.columns
```

Out[47]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Embarked', 'AgeGroup', 'FamilySize'],
      dtype='object')
```

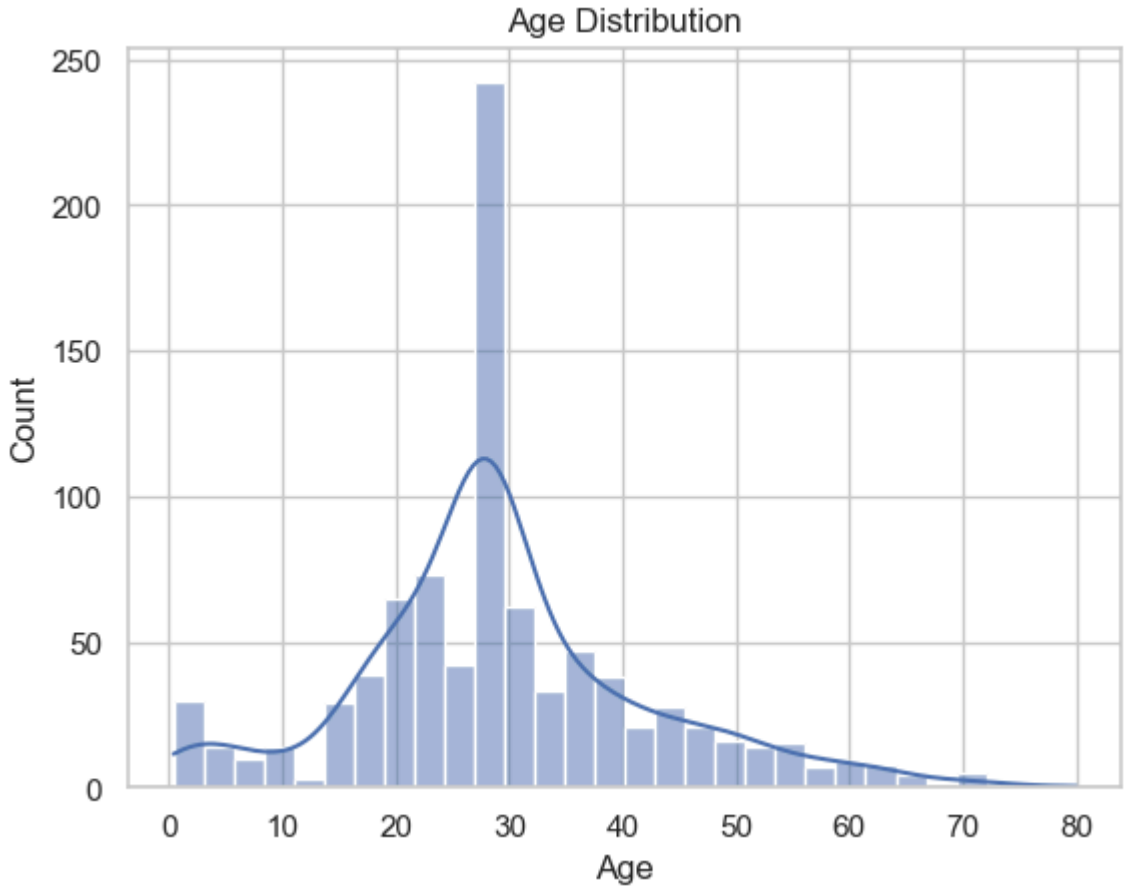
In [49]:
```
plt.figure(figsize=(10,6))
sns.histplot(data=df,x="FamilySize",hue="Survived",multiple="stack")
plt.title("Survived by Famiy Size")
plt.show()
```



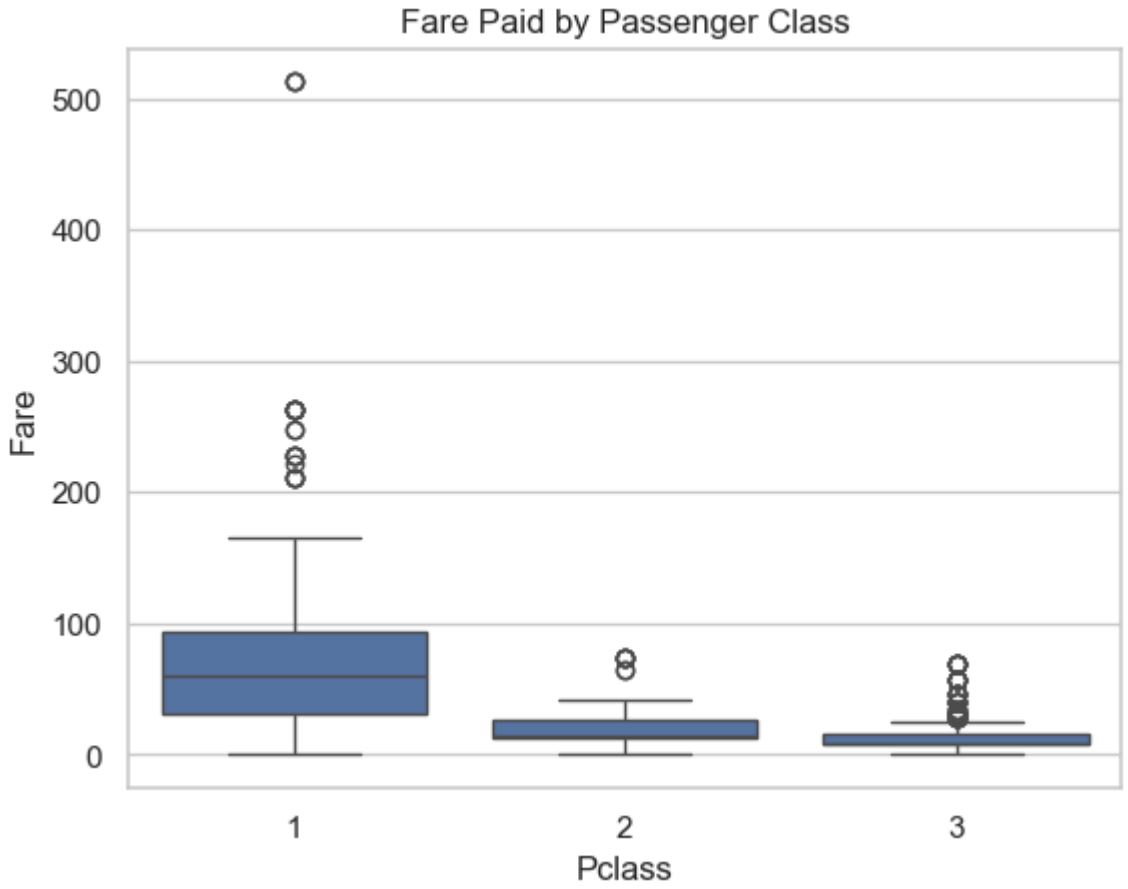
In [50]:
```
sns.countplot(data=df,x="FamilySize",hue="Survived")
plt.title("Survived by Port of Embarkation")
plt.show()
```



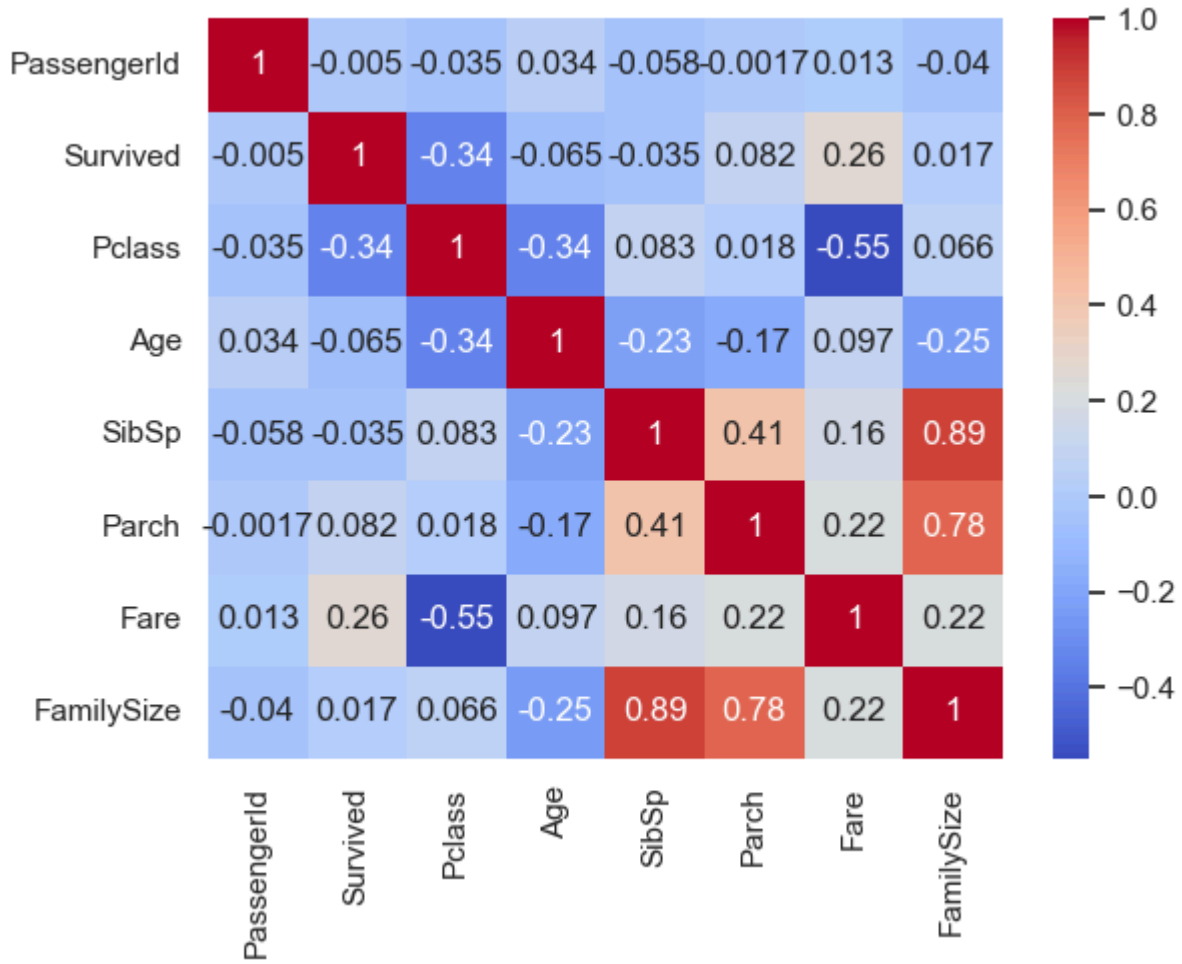
In [52]:
```
sns.histplot(df['Age'],kde=True)
plt.title("Age Distribution")
plt.show()
```



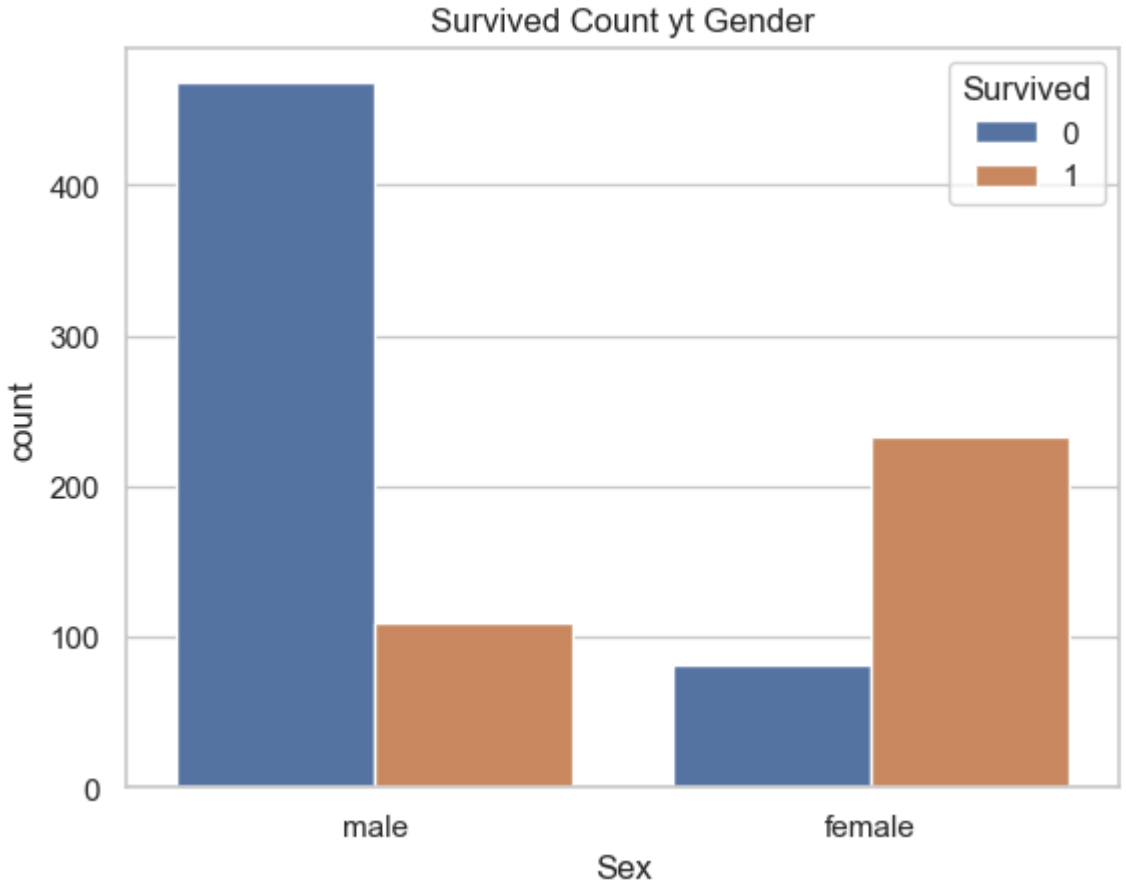
In [54]:
```
sns.boxplot(x="Pclass",y="Fare",data=df)
plt.title("Fare Paid by Passenger Class")
plt.show()
```



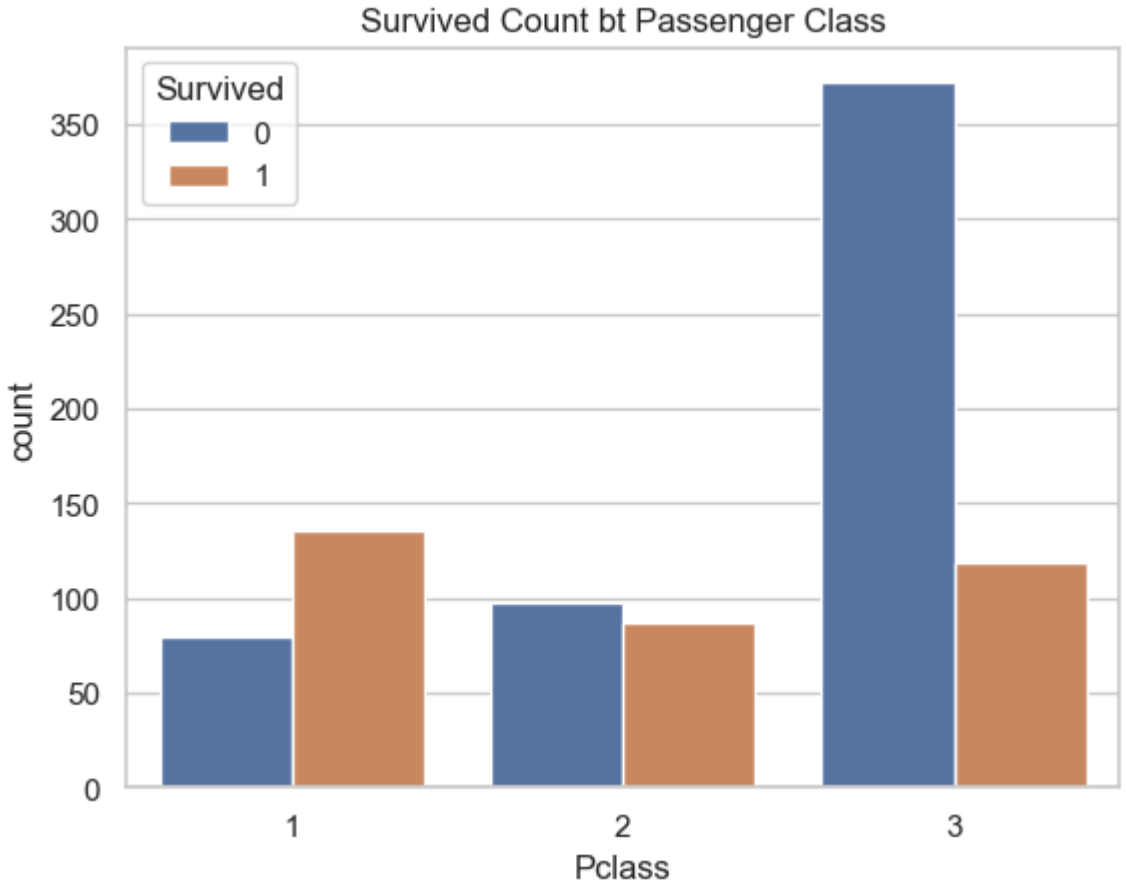
In [55]:
```
numeric_df=df.select_dtypes(include=['number'])
sns.heatmap(numeric_df.corr(),annot=True,cmap="coolwarm")
plt.show()
```



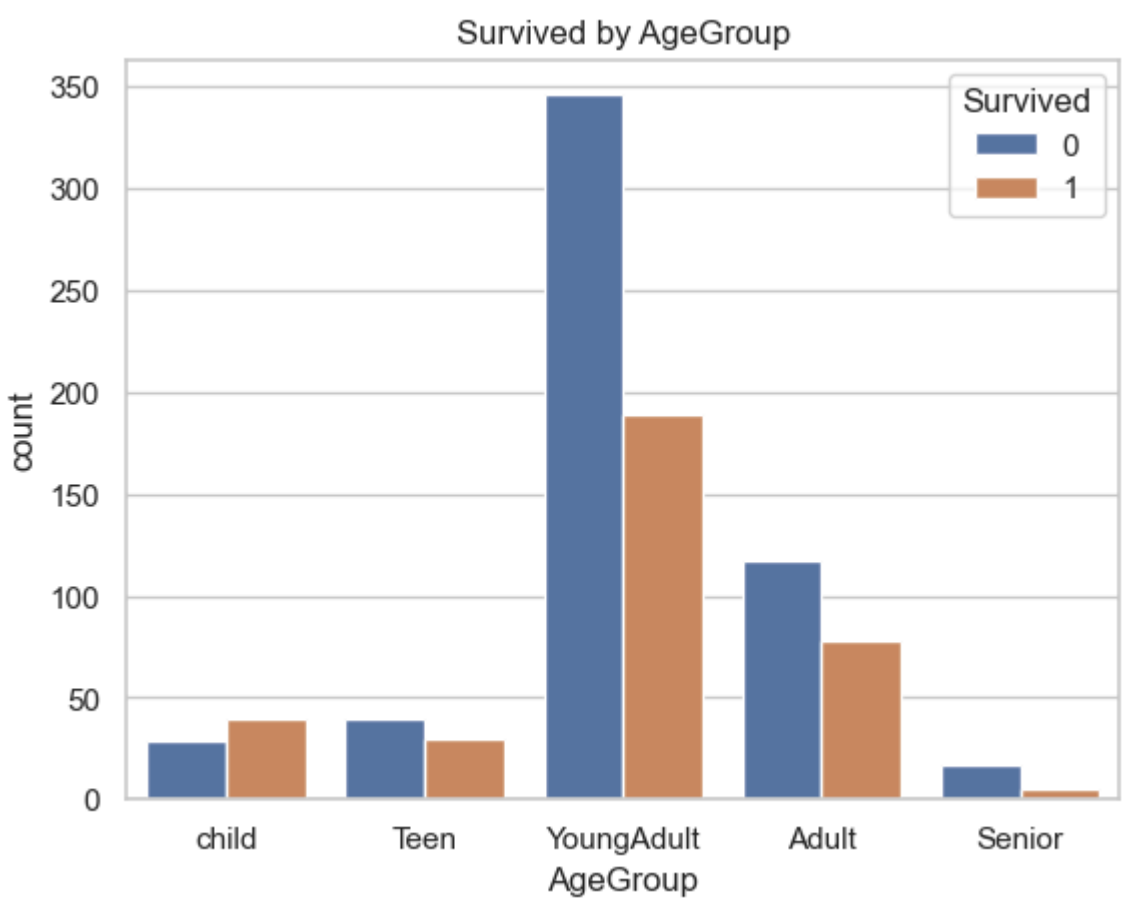
In [56]:
```
sns.countplot(data=df,x='Sex',hue='Survived')
plt.title("Survived Count yt Gender")
plt.show()
```



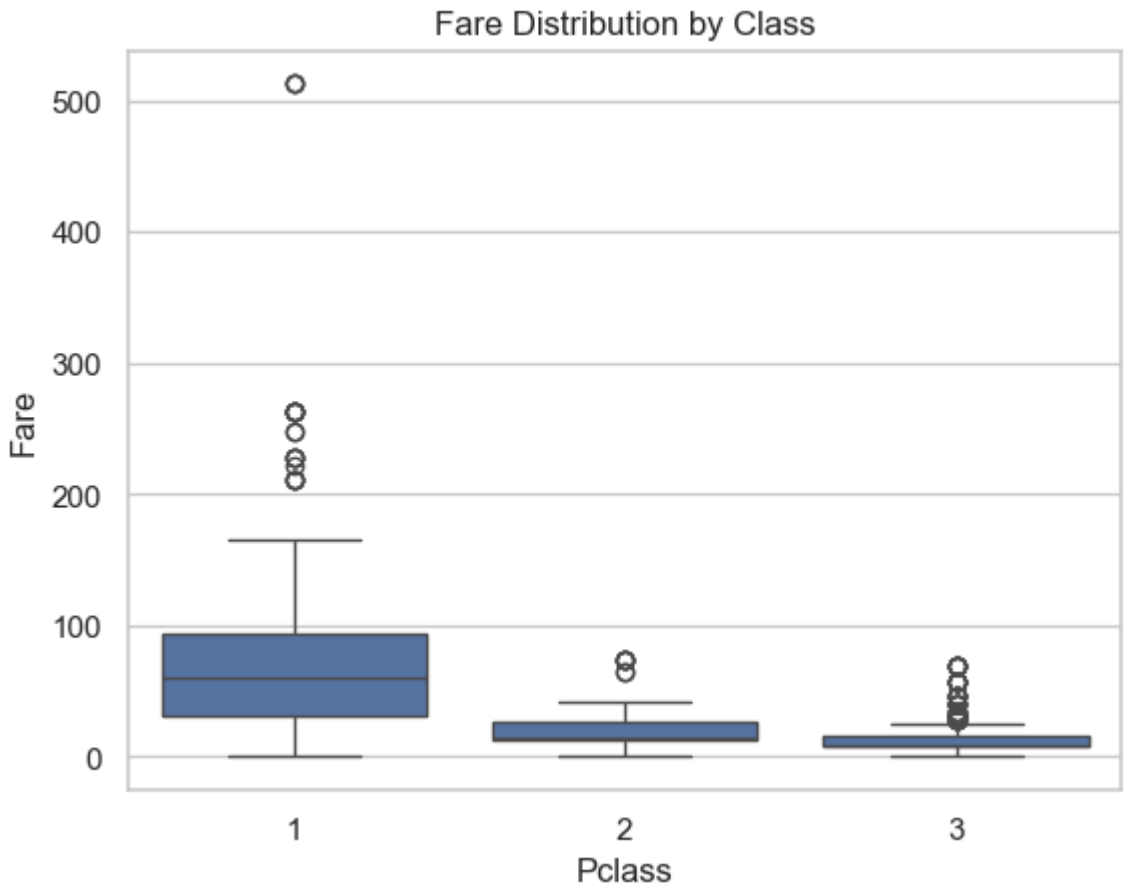
In [57]:
```
sns.countplot(data=df,x='Pclass',hue='Survived')
plt.title("Survived Count bt Passenger Class")
plt.show()
```



In [58]:
```
sns.countplot(data=df,x='AgeGroup',hue='Survived')
plt.title("Survived by AgeGroup")
plt.show()
```



In [59]:
```
sns.boxplot(data=df,x='Pclass',y='Fare')
plt.title("Fare Distribution by Class")
plt.show()
```



🧠 Exploratory Data Analysis (EDA) on Titanic Dataset

This report presents a detailed Exploratory Data Analysis (EDA) of the Titanic dataset.
The goal is to understand the data structure, identify patterns, visualize trends,
detect missing values, and derive meaningful insights related to passenger survival.

In [ ]:
```
📁 Dataset Overview

- Total Rows: 891
- Total Columns: 12
- Target Variable: **Survived**
- Key Features: Pclass, Sex, Age, Fare, SibSp, Parch, Embarked
- Dataset contains categorical, numerical, and mixed-type data.
```

In [ ]:
```
🧩 Missing Values Handling

- **Age** had 177 missing values → Filled using **median** age.
```

- **Embarked** had 2 missing values → Filled using **mode** ('S', 'C', or 'Q').
- **Cabin** had 687 missing values → Dropped due to excessive missing data.
- After cleaning, the dataset contains **zero missing values**.

In [ ]:
👩 Gender vs Survival

- Female passengers had a significantly higher survival rate.
- Male passengers had a much lower chance of survival.
- Gender is one of the strongest predictors of survival.

In [ ]:
🚄 Passenger Class vs Survival

- 1st class passengers survived the most.
- 3rd class passengers had the lowest survival rate.
- Socio-economic status strongly influenced survival chances.

In [ ]:
👶👴 Age Group vs Survival

- Children (0–12 years) showed higher survival rates.
- Young adults and adults had moderate survival.
- Senior passengers had low survival chances.
- Age is an important but secondary predictor.

In [ ]:
💵 Fare Analysis

- Fare distribution is **right-skewed** (most passengers paid low fares).
- Higher fare-paying passengers (1st class) had higher survival rates.
- Fare is moderately correlated with survival.

In [ ]:
👨‍👩‍👧‍👦 Family Size vs Survival

- Very large families (FamilySize > 4) had low survival chances.
- Medium-sized families (2–4 members) survived more.
- Solo travelers had moderate survival rates.

In [ ]:
⚓ Embarked vs Survival

- Passengers boarding from **Cherbourg (C)** survived more.
- Southampton (S) passengers had lower survival.
- Embarked location shows noticeable survival patterns.

In [ ]:
🔗 Correlation Summary

- **Pclass** has strong negative correlation with **Fare** and **Survival**.

- **Fare** and **Survival** have moderate positive correlation.
- **FamilySize** and **Survival** show weak correlation.
- Most numerical features have weak inter-correlation.

In [ ]:
```
🏁 Final Conclusion

Based on the EDA of the Titanic dataset:

- Females and children had significantly higher survival rates.
- Higher-class passengers (1st class) were more likely to survive.
- Fare plays an important role in survival prediction.
- Large families and passengers from 3rd class had poor survival chances.
- After cleaning missing values, the dataset is suitable for modeling.

This completes the exploratory analysis, providing insights into
factors affecting survival on the Titanic.
```