In [2]:

```
pip install seaborn
```

```
Collecting seaborn
  Downloading seaborn-0.12.2-py3-none-any.whl (293 kB)
     ---------------------------------------- 0.0/293.3 kB ? eta -:--:--
     ---- ----------------------------------- 30.7/293.3 kB ? eta -:--:--
     ----- ---------------------------------- 41.0/293.3 kB 487.6 kB/s eta 0:00:01
     ----------------- ------------------- 143.4/293.3 kB 1.7 MB/s eta 0:00:01
     ---------------------------- -------- 225.3/293.3 kB 1.4 MB/s eta 0:00:01
     ------------------------------------  286.7/293.3 kB 1.4 MB/s eta 0:00:01
     ------------------------------------  286.7/293.3 kB 1.4 MB/s eta 0:00:01
     ------------------------------------  286.7/293.3 kB 1.4 MB/s eta 0:00:01
     ------------------------------------  286.7/293.3 kB 1.4 MB/s eta 0:00:01
     ---------------------------------- 293.3/293.3 kB 786.2 kB/s eta 0:00:00
Requirement already satisfied: numpy!=1.24.0,>=1.17 in c:\users\tia\appdata\local\progr
ams\python\python310\lib\site-packages (from seaborn) (1.24.1)
Requirement already satisfied: pandas>=0.25 in c:\users\tia\appdata\local\programs\pyth
on\python310\lib\site-packages (from seaborn) (2.0.1)
Requirement already satisfied: matplotlib!=3.6.1,>=3.1 in c:\users\tia\appdata\local\pr
ograms\python\python310\lib\site-packages (from seaborn) (3.7.1)
Requirement already satisfied: cycler>=0.10 in c:\users\tia\appdata\local\programs\pyth
on\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (0.11.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\tia\appdata\local\progr
ams\python\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (2.8.2)
Requirement already satisfied: packaging>=20.0 in c:\users\tia\appdata\local\programs\p
ython\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (23.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\tia\appdata\local\programs
\python\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (4.39.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\tia\appdata\local\programs
\python\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (1.4.4)
Requirement already satisfied: pillow>=6.2.0 in c:\users\tia\appdata\local\programs\pyt
hon\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\tia\appdata\local\programs
\python\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (3.0.9)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\tia\appdata\local\programs
\python\python310\lib\site-packages (from matplotlib!=3.6.1,>=3.1->seaborn) (1.0.7)
Requirement already satisfied: tzdata>=2022.1 in c:\users\tia\appdata\local\programs\py
thon\python310\lib\site-packages (from pandas>=0.25->seaborn) (2023.3)
Requirement already satisfied: pytz>=2020.1 in c:\users\tia\appdata\local\programs\pyth
on\python310\lib\site-packages (from pandas>=0.25->seaborn) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\tia\appdata\local\programs\python\p
ython310\lib\site-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.1->seabor
n) (1.16.0)
Installing collected packages: seaborn
Successfully installed seaborn-0.12.2
Note: you may need to restart the kernel to use updated packages.


[notice] A new release of pip is available: 23.0.1 -> 23.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [3]:

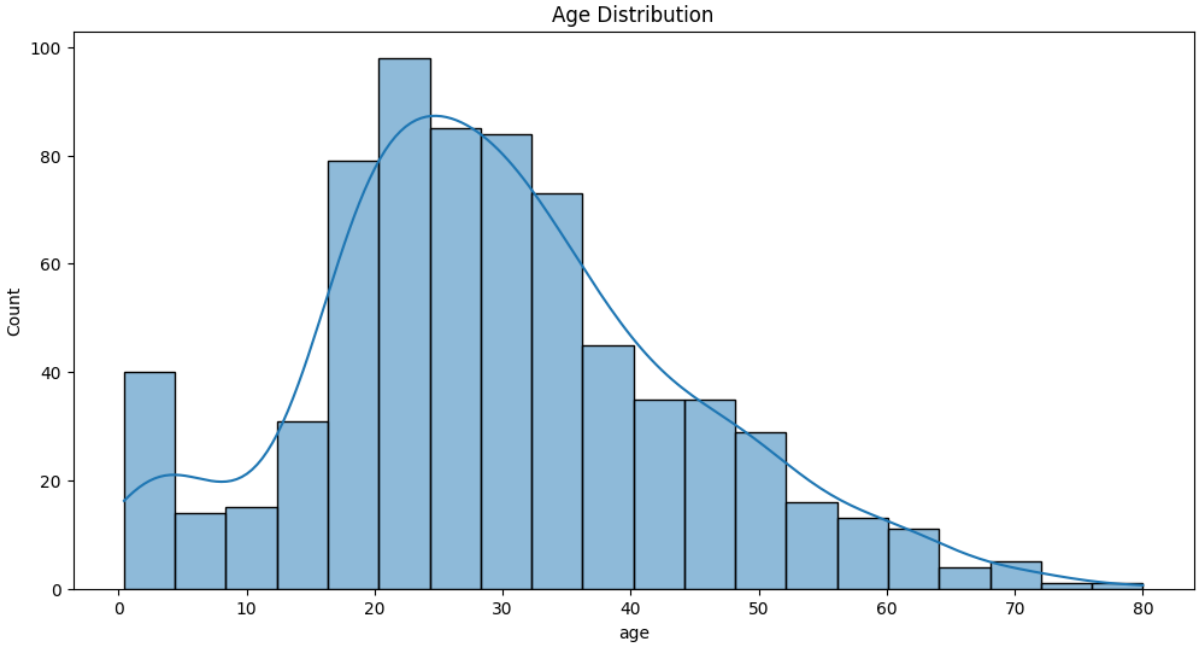
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [5]:

```
# 2. Load the dataset
df = pd.read_csv('titanic.csv')
```

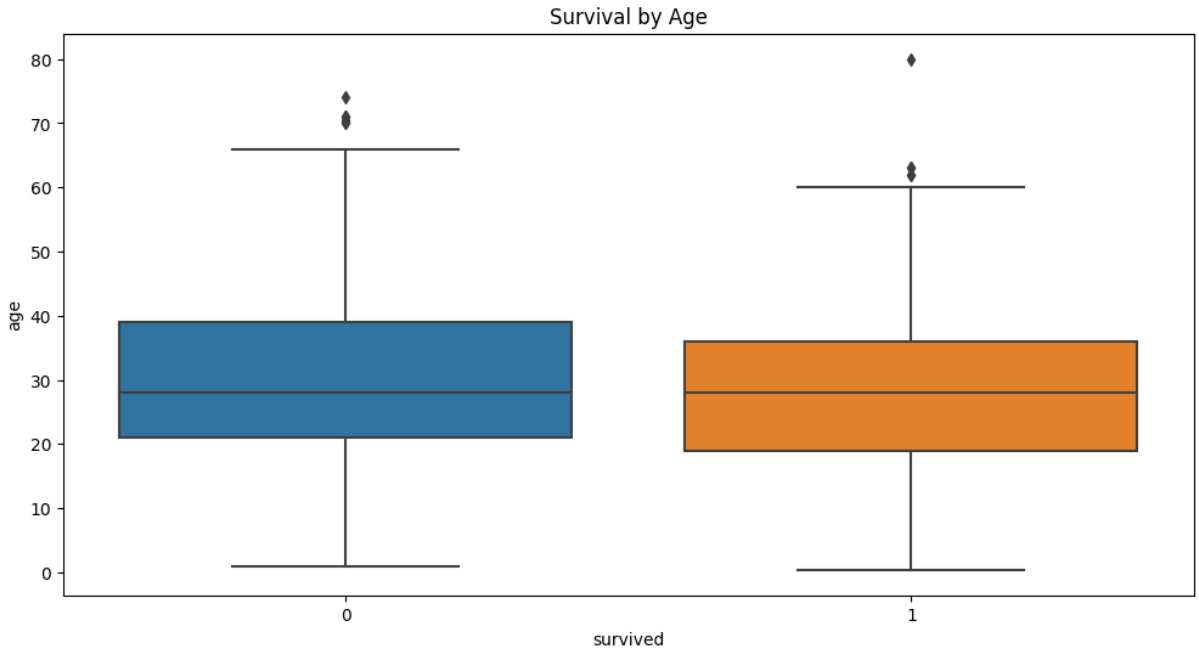
In [8]:

```
# 3. Perform Visualization
# Univariate Analysis
plt.figure(figsize=(12, 6))
sns.histplot(df['age'].dropna(), kde=True)
plt.title('Age Distribution')
plt.show()
```



In [10]:

```
# Bi-Variate Analysis
plt.figure(figsize=(12, 6))
sns.boxplot(x='survived', y='age', data=df)
plt.title('Survival by Age')
plt.show()
```



In [11]:

```
# Multi-Variate Analysis
plt.figure(figsize=(12, 6))
sns.violinplot(x='survived', y='age', hue='sex', data=df, split=True)
plt.title('Survival by Age and Gender')
plt.show()
```

In [12]:

```
# 4. Perform descriptive statistics
print(df.describe())
```

```
         survived      pclass         age       sibsp       parch        fare
count  891.000000  891.000000  714.000000  891.000000  891.000000  891.000000
mean     0.383838    2.308642   29.699118    0.523008    0.381594   32.204208
std      0.486592    0.836071   14.526497    1.102743    0.806057   49.693429
min      0.000000    1.000000    0.420000    0.000000    0.000000    0.000000
25%      0.000000    2.000000   20.125000    0.000000    0.000000    7.910400
50%      0.000000    3.000000   28.000000    0.000000    0.000000   14.454200
75%      1.000000    3.000000   38.000000    1.000000    0.000000   31.000000
max      1.000000    3.000000   80.000000    8.000000    6.000000  512.329200
```

In [14]:

```
# 5. Handle missing values
df.dropna(subset=['age'], inplace=True)  # Dropping rows with missing age values
```

In [17]:

```
# 6. Find and replace outliers (example using Z-score)
from scipy import stats
z_scores = stats.zscore(df['fare'])
threshold = 3
outliers = df.loc[z_scores > threshold]
df.loc[z_scores > threshold, 'fare'] = df['fare'].mean()
```

In [18]:

```
# 7. Check for categorical columns and perform encoding (example using one-hot encoding)
categorical_cols = ['sex', 'embarked']
df_encoded = pd.get_dummies(df, columns=categorical_cols)
```

In [19]:

```
# 8. Split the data into dependent and independent variables
X = df_encoded.drop('survived', axis=1)
y = df_encoded['survived']
```

In [34]:

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder

# Separate the features and target variable
X = df.drop('survived', axis=1)
y = df['survived']

# Define the categorical and numerical features
categorical_features = ['sex', 'embarked', 'class', 'who', 'adult_male', 'deck', 'embark_town', 'alone
numerical_features = ['pclass', 'age', 'sibsp', 'parch', 'fare']

# Create a column transformer for preprocessing
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numerical_features),
        ('cat', OneHotEncoder(), categorical_features)
    ])

# Apply preprocessing to the features
X_preprocessed = preprocessor.fit_transform(X)

# Print the preprocessed feature matrix
print(X_preprocessed)
```

```
[[ 0.91123237 -0.53037664  0.52457013 ...  0.          1.
   0.        ]
 [-1.47636364  0.57183099  0.52457013 ...  0.          1.
   0.        ]
 [ 0.91123237 -0.25482473 -0.55170307 ...  0.          0.
   1.        ]
 ...
 [-1.47636364 -0.73704057 -0.55170307 ...  0.          0.
   1.        ]
 [-1.47636364 -0.25482473 -0.55170307 ...  0.          0.
   1.        ]
 [ 0.91123237  0.15850313 -0.55170307 ...  0.          0.
   1.        ]]
```

In [36]:

```
# 10. Split the data into training and testing
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_preprocessed, y, test_size=0.2, random_state=42
```