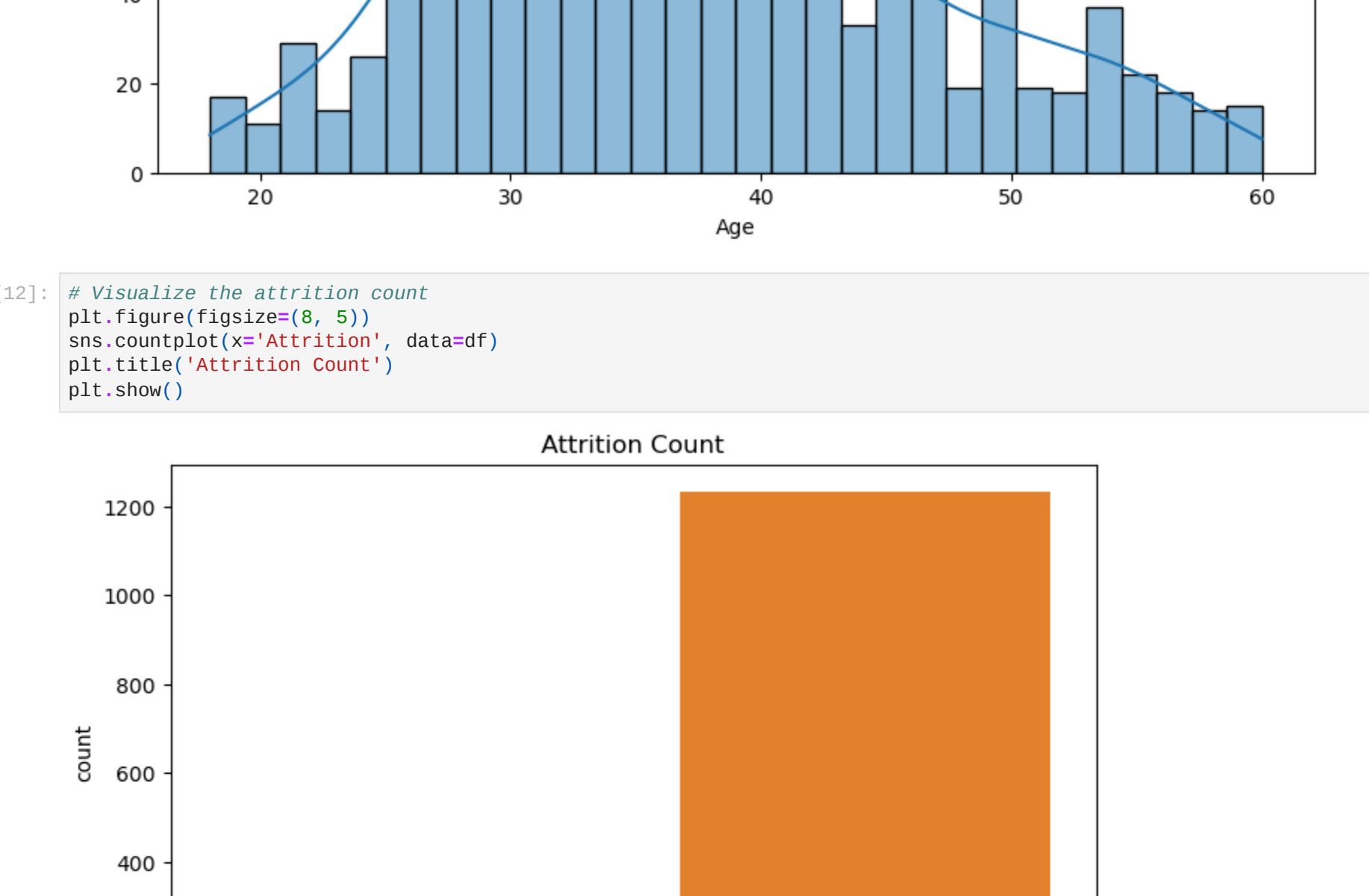
Data Loading & Cleaning and Preprocessing

In [1]:
```python
import pandas as pd
import numpy as np

df = pd.read_csv(r"C:\Users\rizwa\OneDrive\Desktop\Intern\Final project\HR-Employee-Attrition.csv")

print("Data Information:")
print(df.info())

# Check for missing values
print("\nMissing Values:")
print(df.isnull().sum())

# Handle missing values (if any)
df = df.dropna()

# Display basic statistics after handling missing values
print("\nSummary Statistics:")
print(df.describe())

# Check for duplicates
print("\nDuplicate Rows:")
print(df.duplicated().sum())

# Handle duplicates (if any)
# Example: Drop duplicate rows
df = df.drop_duplicates()

# Confirm changes
print("\nData Information After Cleaning:")
print(df.info())
```

```
Data Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64 
 9   EmployeeNumber            1470 non-null   int64 
 10  EnvironmentSatisfaction   1470 non-null   int64 
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64 
 13  JobInvolvement            1470 non-null   int64 
 14  JobLevel                  1470 non-null   int64 
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64 
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64 
 19  MonthlyRate               1470 non-null   int64 
 20  NumCompaniesWorked        1470 non-null   int64 
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64 
 24  PerformanceRating         1470 non-null   int64 
 25  RelationshipSatisfaction  1470 non-null   int64 
 26  StandardHours             1470 non-null   int64 
 27  StockOptionLevel          1470 non-null   int64 
 28  TotalWorkingYears         1470 non-null   int64 
 29  TrainingTimesLastYear     1470 non-null   int64 
 30  WorkLifeBalance           1470 non-null   int64 
 31  YearsAtCompany            1470 non-null   int64 
 32  YearsInCurrentRole        1470 non-null   int64 
 33  YearsSinceLastPromotion   1470 non-null   int64 
 34  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
None

Missing Values:
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64

Summary Statistics:
               Age    DailyRate  DistanceFromHome    Education  EmployeeCount  \
count  1470.000000  1470.000000       1470.000000  1470.000000         1470.0   
mean     36.923810   802.485714          9.192517     2.912925            1.0   
std       9.135373   403.509100          8.106864     1.024165            0.0   
min      18.000000   102.000000          1.000000     1.000000            1.0   
25%      30.000000   465.000000          2.000000     2.000000            1.0   
50%      36.000000   802.000000          7.000000     3.000000            1.0   
75%      43.000000  1157.000000         14.000000     4.000000            1.0   
max      60.000000  1499.000000         29.000000     5.000000            1.0   

       EmployeeNumber  EnvironmentSatisfaction   HourlyRate  JobInvolvement  \
count     1470.000000              1470.000000  1470.000000     1470.000000   
mean      1024.865306                 2.721769    65.891156        2.729932   
std        602.024335                 1.093082    20.329428        0.711561   
min          1.000000                 1.000000    30.000000        1.000000   
25%        491.250000                 2.000000    48.000000        2.000000   
50%       1020.500000                 3.000000    66.000000        3.000000   
75%       1555.750000                 4.000000    83.750000        3.000000   
max       2068.000000                 4.000000   100.000000        4.000000   

          JobLevel  ...  RelationshipSatisfaction  StandardHours  \
count  1470.000000  ...               1470.000000         1470.0   
mean      2.063946  ...                  2.712245           80.0   
std       1.106940  ...                  1.081209            0.0   
min       1.000000  ...                  1.000000           80.0   
25%       1.000000  ...                  2.000000           80.0   
50%       2.000000  ...                  3.000000           80.0   
75%       3.000000  ...                  4.000000           80.0   
max       5.000000  ...                  4.000000           80.0   

       StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear  \
count       1470.000000        1470.000000            1470.000000   
mean           0.793878          11.279592               2.799320   
std            0.852077           7.780782               1.289271   
min            0.000000           0.000000               0.000000   
25%            0.000000           6.000000               2.000000   
50%            1.000000          10.000000               3.000000   
75%            1.000000          15.000000               3.000000   
max            3.000000          40.000000               6.000000   

       WorkLifeBalance  YearsAtCompany  YearsInCurrentRole  \
count      1470.000000     1470.000000         1470.000000   
mean          2.761224        7.008163            4.229252   
std           0.706476        6.126525            3.623137   
min           1.000000        0.000000            0.000000   
25%           2.000000        3.000000            2.000000   
50%           3.000000        5.000000            3.000000   
75%           3.000000        9.000000            7.000000   
max           4.000000       40.000000           18.000000   

       YearsSinceLastPromotion  YearsWithCurrManager  
count              1470.000000           1470.000000  
mean                  2.187755              4.123129  
std                   3.222430              3.568136  
min                   0.000000              0.000000  
25%                   0.000000              2.000000  
50%                   1.000000              3.000000  
75%                   3.000000              7.000000  
max                  15.000000             17.000000  

[8 rows x 26 columns]

Duplicate Rows:
0

Data Information After Cleaning:
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64 
 9   EmployeeNumber            1470 non-null   int64 
 10  EnvironmentSatisfaction   1470 non-null   int64 
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64 
 13  JobInvolvement            1470 non-null   int64 
 14  JobLevel                  1470 non-null   int64 
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64 
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64 
 19  MonthlyRate               1470 non-null   int64 
 20  NumCompaniesWorked        1470 non-null   int64 
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64 
 24  PerformanceRating         1470 non-null   int64 
 25  RelationshipSatisfaction  1470 non-null   int64 
 26  StandardHours             1470 non-null   int64 
 27  StockOptionLevel          1470 non-null   int64 
 28  TotalWorkingYears         1470 non-null   int64 
 29  TrainingTimesLastYear     1470 non-null   int64 
 30  WorkLifeBalance           1470 non-null   int64 
 31  YearsAtCompany            1470 non-null   int64 
 32  YearsInCurrentRole        1470 non-null   int64 
 33  YearsSinceLastPromotion   1470 non-null   int64 
 34  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(26), object(9)
memory usage: 413.4+ KB
None
```

Data Analysis and Modeling (Analyzed the cleaned HR data to identify patterns, trends, and correlations,EDA: Exploratory Data Analysis)

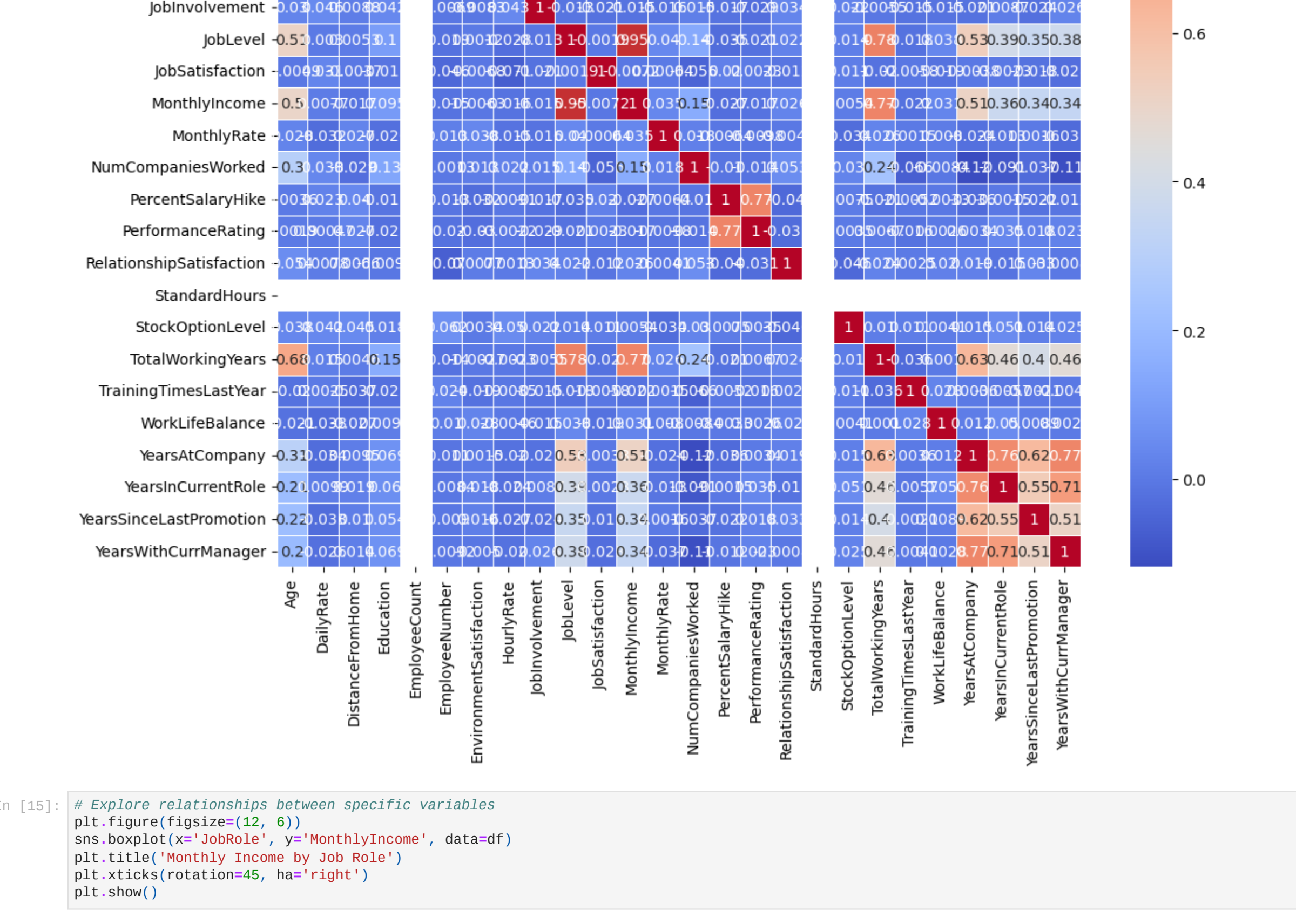
In [11]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Visualize the distribution of Age
plt.figure(figsize=(10, 6))
sns.histplot(df['Age'], bins=30, kde=True)
plt.title('Distribution of Age')
plt.show()
```



In [12]:
```python
# Visualize the attrition count
plt.figure(figsize=(8, 5))
sns.countplot(x='Attrition', data=df)
plt.title('Attrition Count')
plt.show()
```

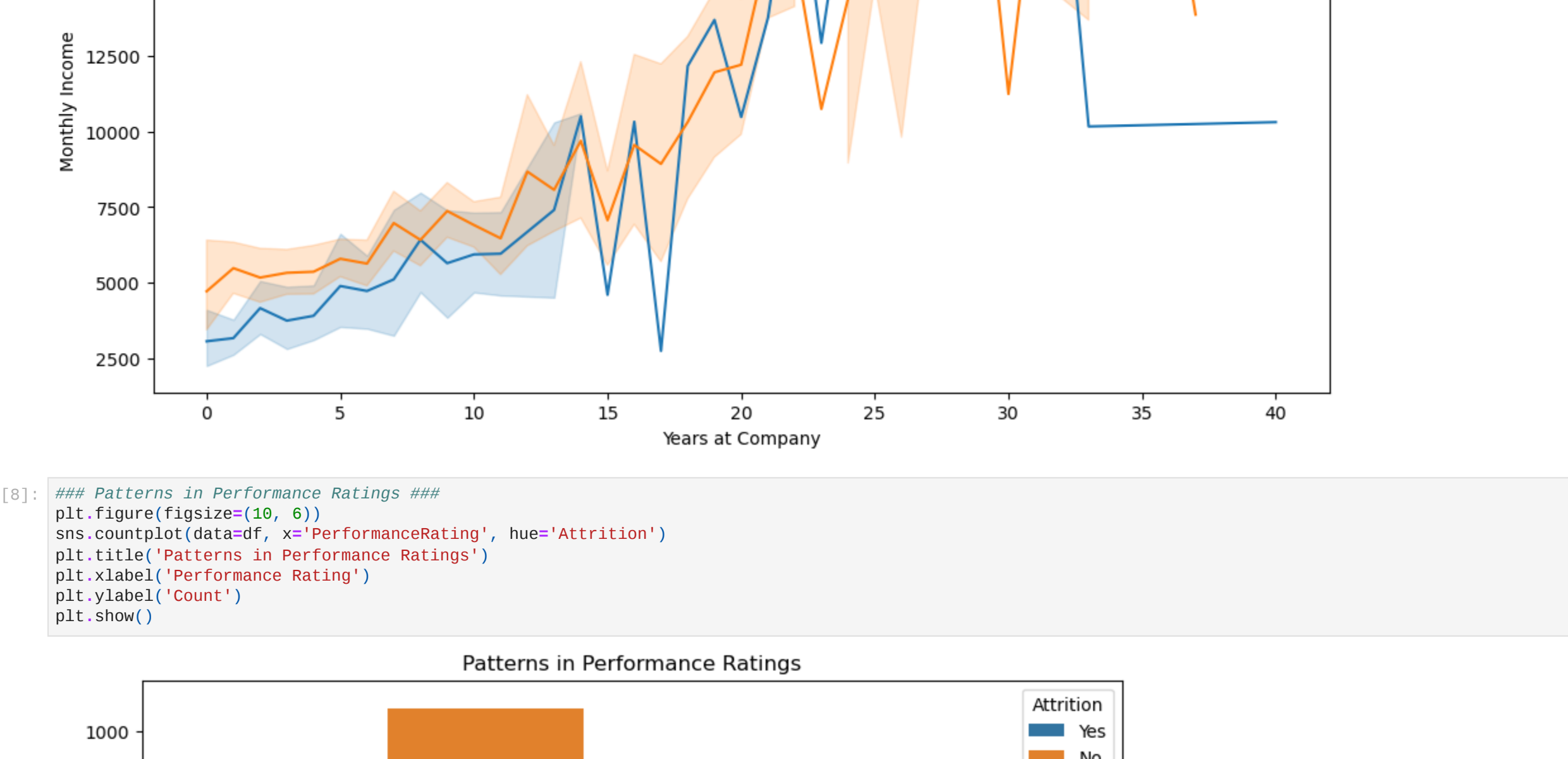
In [13]:
```python
# Explore the correlation matrix
correlation_matrix = df.corr()
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```

```
C:\Users\rizwa\AppData\Local\Temp\ipykernel_1516\4129031465.py:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  correlation_matrix = df.corr()
```



In [15]:
```python
# Explore relationships between specific variables
plt.figure(figsize=(12, 6))
sns.boxplot(x='JobRole', y='MonthlyIncome', data=df)
plt.title('Monthly Income by Job Role')
plt.xticks(rotation=45, ha='right')
plt.show()
```

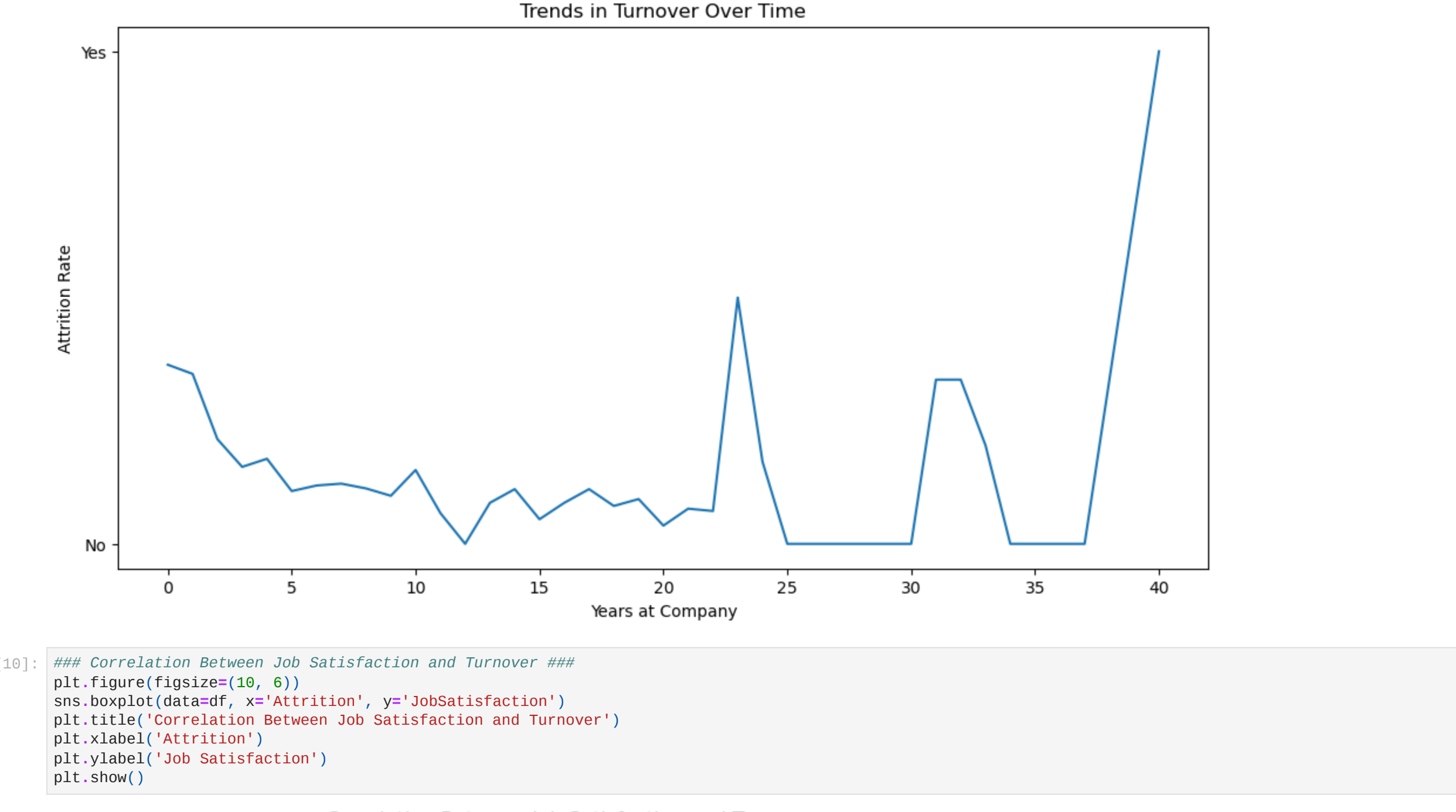
In [7]:
```python
### Trends in Salary Over Time ###
plt.figure(figsize=(12, 6))
sns.lineplot(data=df, x='YearsAtCompany', y='MonthlyIncome', hue='Attrition')
plt.title('Trends in Salary Over Time')
plt.xlabel('Years at Company')
plt.ylabel('Monthly Income')
plt.show()
```



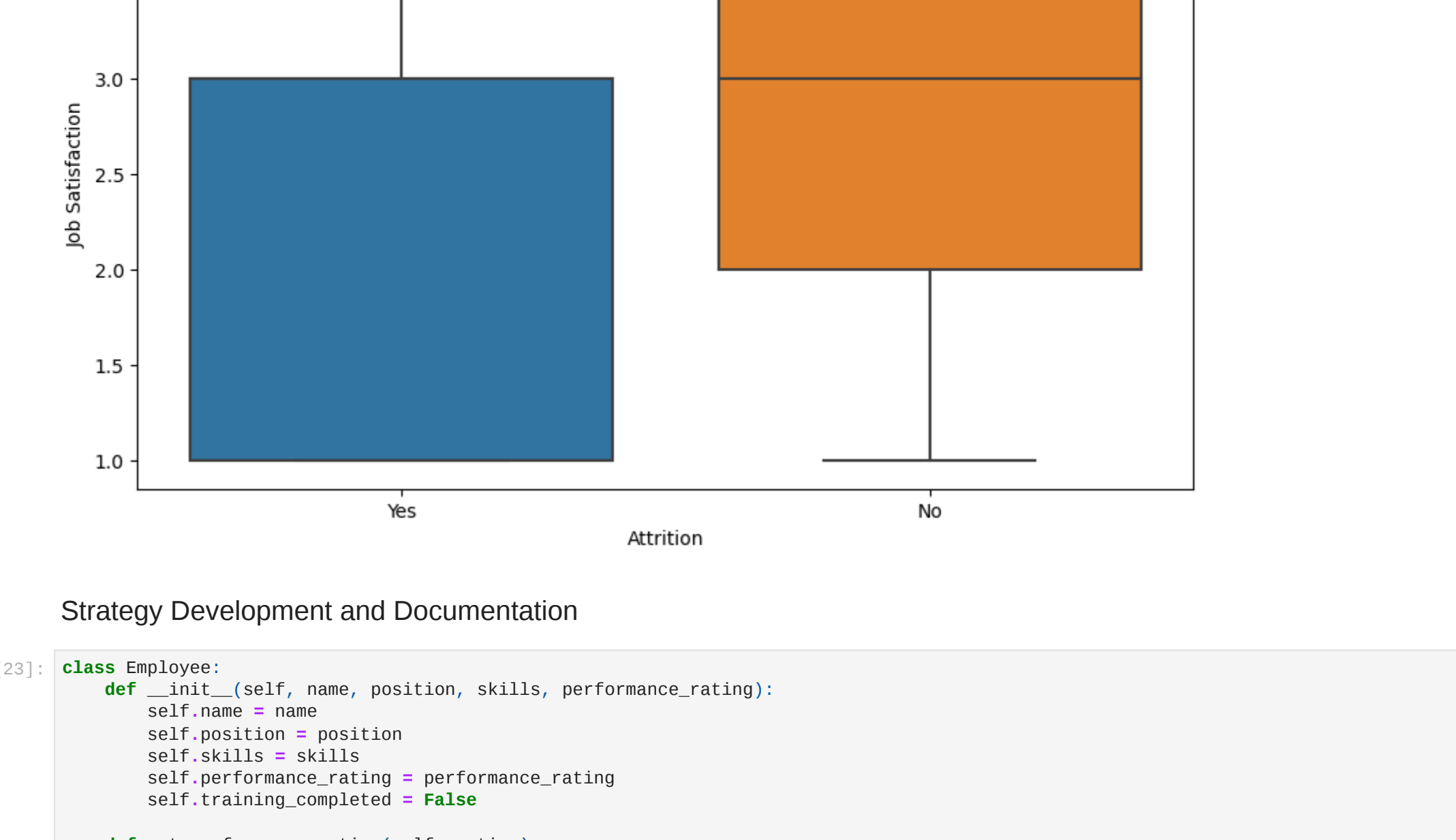
In [8]:
```python
### Patterns in Performance Ratings ###
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='PerformanceRating', hue='Attrition')
plt.title('Patterns in Performance Ratings')
plt.xlabel('Performance Rating')
plt.ylabel('Count')
plt.show()
```

In [9]:
```python
### Trends in Turnover Over Time ###
plt.figure(figsize=(12, 6))
sns.lineplot(data=df, x='YearsAtCompany', y='Attrition', estimator='mean', ci=None)
plt.title('Trends in Turnover Over Time')
plt.xlabel('Years at Company')
plt.ylabel('Attrition Rate')
plt.show()
```

```
C:\Users\rizwa\AppData\Local\Temp\ipykernel_1516\3951006875.py:3: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sns.lineplot(data=df, x='YearsAtCompany', y='Attrition', estimator='mean', ci=None)
```



In [10]:
```python
### Correlation Between Job Satisfaction and Turnover ###
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Attrition', y='JobSatisfaction')
plt.title('Correlation Between Job Satisfaction and Turnover')
plt.xlabel('Attrition')
plt.ylabel('Job Satisfaction')
plt.show()
```



Strategy Development and Documentation

In [23]:
```python
class Employee:
    def __init__(self, name, position, skills, performance_rating):
        self.name = name
        self.position = position
        self.skills = skills
        self.performance_rating = performance_rating
        self.training_completed = False

    def set_performance_rating(self, rating):
        self.performance_rating = rating

    def complete_training(self):
        self.training_completed = True
        print(f"{self.name} has completed targeted training.")

    def adjust_compensation(self, new_salary):
        print(f"Compensation for {self.name} adjusted to {new_salary}.")

    def promote(self, new_position):
        print(f"{self.name} has been promoted to {new_position}.")
        self.position = new_position


class PerformanceEnhancementStrategy:
    def __init__(self):
        self.employees = []

    def add_employee(self, employee):
        self.employees.append(employee)

    def implement_strategy(self):
        for employee in self.employees:
            if employee.performance_rating < 4.0:
                # Offer targeted training for employees with low performance
                employee.complete_training()
            elif 4.0 <= employee.performance_rating < 4.5:
                # Consider compensation adjustment for employees with moderate performance
                employee.adjust_compensation(1.05)
            elif employee.performance_rating >= 4.5:
                # Consider career progression for high-performing employees
                employee.promote("Senior " + employee.position)



employee1 = Employee("Ekta Agrawal", "Software Developer", ["Python", "JavaScript"], 3.8)
employee2 = Employee("Harshvardhan", "Data Analyst", ["SQL", "Excel"], 4.2)
employee3 = Employee("Rizwan Siddiqui", "UX Designer", ["UI/UX", "Sketch"], 4.8)


strategy = PerformanceEnhancementStrategy()


strategy.add_employee(employee1)
strategy.add_employee(employee2)
strategy.add_employee(employee3)

# Implement the performance enhancement strategy
strategy.implement_strategy()

# Print the results
for employee in strategy.employees:
    print(f"{employee.name} - Position: {employee.position}, Performance Rating: {employee.performance_rating}")
```

```
Ekta Agrawal has completed targeted training.
Compensation for Harshvardhan adjusted to 1.05.
Rizwan Siddiqui has been promoted to Senior UX Designer.
Ekta Agrawal - Position: Software Developer, Performance Rating: 3.8
Harshvardhan - Position: Data Analyst, Performance Rating: 4.2
Rizwan Siddiqui - Position: Senior UX Designer, Performance Rating: 4.8
```

In [ ]: