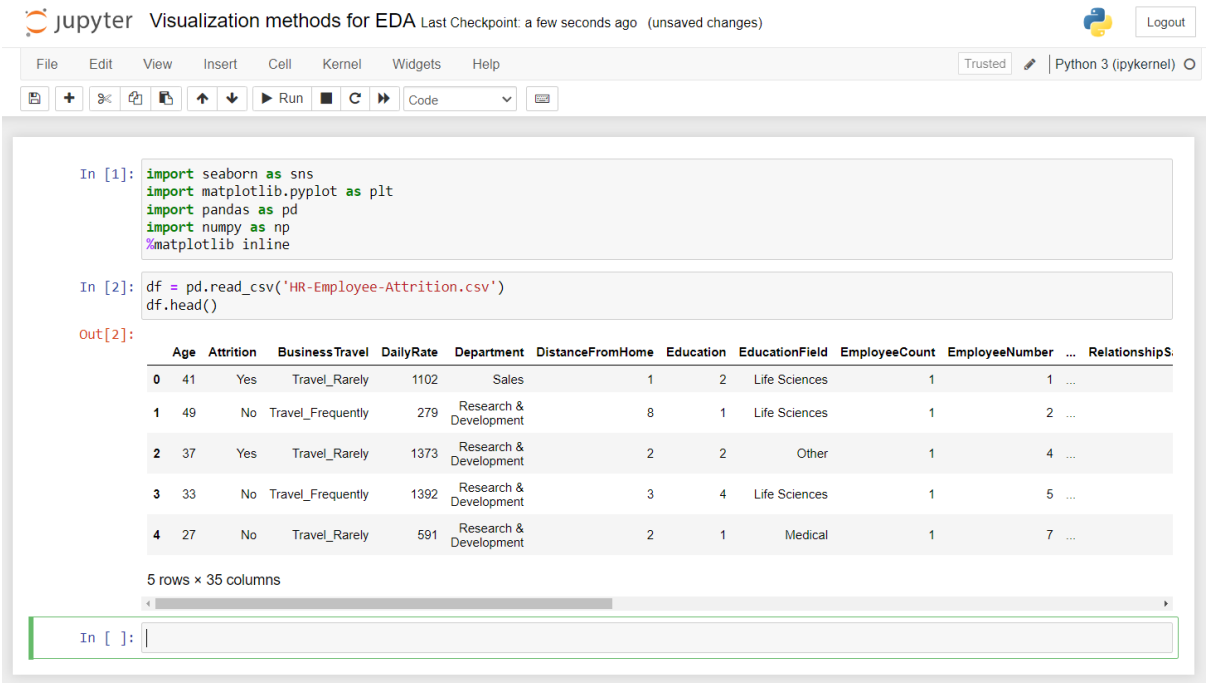
😊 Visualization methods for EDA

1. For this lab you need to upload a new file to your EDA Lab folder in Jupyter Notebook with the name HR-Employee-Attrition.csv as you can see below.

2. After that create a new Python 3 kernel in your Notebook. So, the first thing we need to do is import some important libraries as you can see below then load our CSV file in the notebook then read the data from it.



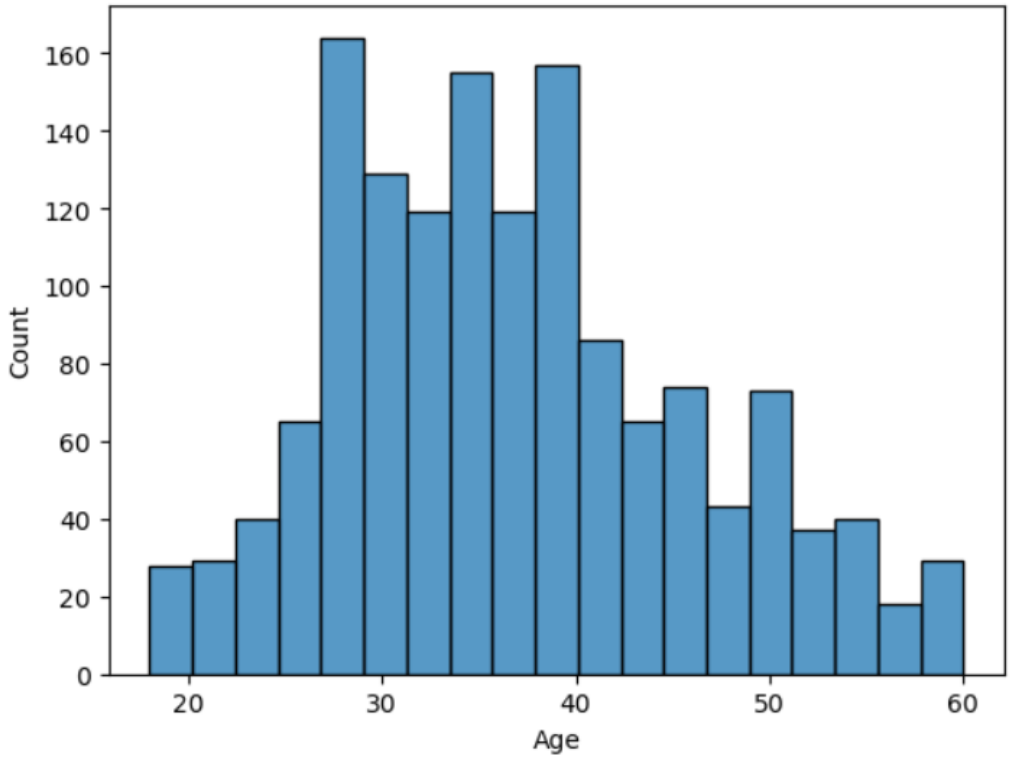
3. Now we are going to describe the data type of this dataset as you can see below.

In [3]:
```
df.info()
```

```
<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
```

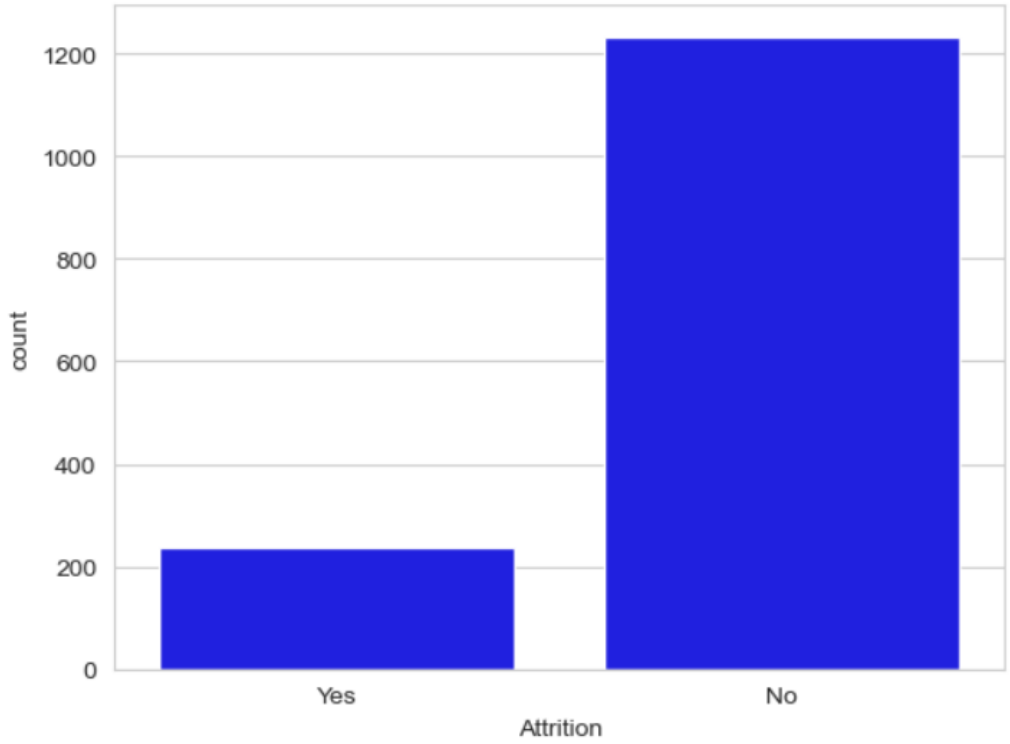
4. Here we have created a histogram chart for the Age column. From the dataset, we can see that most employees fall within the age range of 30 to 50, with the highest concentration between 20 and 40 years old. This indicates that the organization has a significant number of employees in this age group.

```
In [4]: # Univariate
        # Histogram
        sns.histplot(data=df,x='Age')
        plt.show()
```



5. To understand the attrition percentage in a company, you can look at the proportion of employees who leave the organization over a specific period. This is typically expressed as a percentage of the total workforce.
6. For analyzing attrition, which is a categorical variable (e.g., "Yes" or "No" for whether an employee has left), a common visualization method is a count plot. A count plot displays the frequency of each category, allowing you to easily see how many employees have left compared to those who have stayed. This can help you gain insights into employee turnover and identify any trends or patterns related to attrition within the organization.
7. You can use a count plot or a bar plot for analyzing categorical data, as they effectively show the frequency of each category. For univariate analysis of numerical data, you can utilize a histogram or similar plots to visualize the distribution of values.
8. In summary:
9. **Numerical Data**: Use histograms to understand the distribution.
10. **Categorical Data**: Use count plots or bar plots to analyze the frequency of categories.
11. This approach helps you visually interpret and compare the data effectively.

```
In [5]: sns.set_style('whitegrid')
        sns.countplot(data=df,x='Attrition',color='blue')
Out[5]: <Axes: xlabel='Attrition', ylabel='count'>
```



12. From the data, it is evident that approximately **84% of employees remained with the organization**, while **16% chose to leave**. This attrition percentage highlights the retention level within the company, indicating that the majority of employees decided to stay.

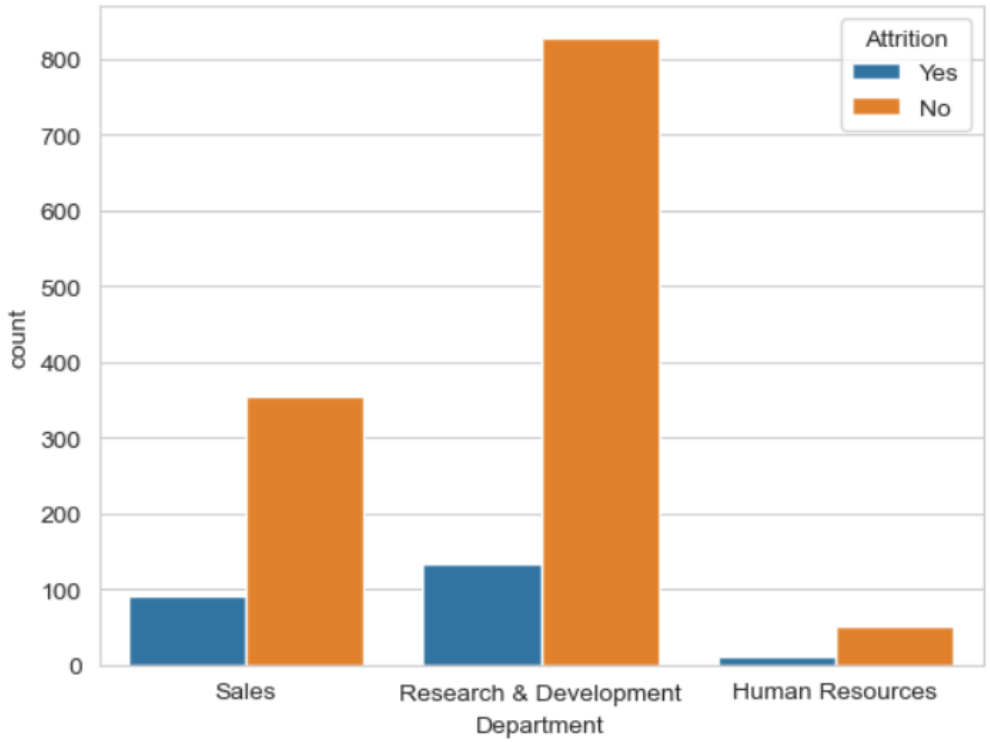
```
In [6]: df['Attrition'].value_counts(normalize=True)
Out[6]: Attrition
        No     0.838776
        Yes    0.161224
        Name: proportion, dtype: float64
```

13. If you want to analyze employee attrition by department, you can create a count plot or bar plot that shows the number of employees who left each department. This will help you identify which departments have higher attrition rates. By comparing the counts of employees who quit in each department, you can gain insights into potential issues within specific areas of the organization and develop targeted strategies for retention.

14. From your observations, it appears that in the Sales department, close to 100 employees have left the organization, indicating a noticeable level of attrition. In contrast, the Research and Development (R&D) department has a larger employee base, but the attrition rate is relatively lower compared to Sales. This suggests that while Sales may be experiencing significant turnover, R&D is retaining more of its staff despite having

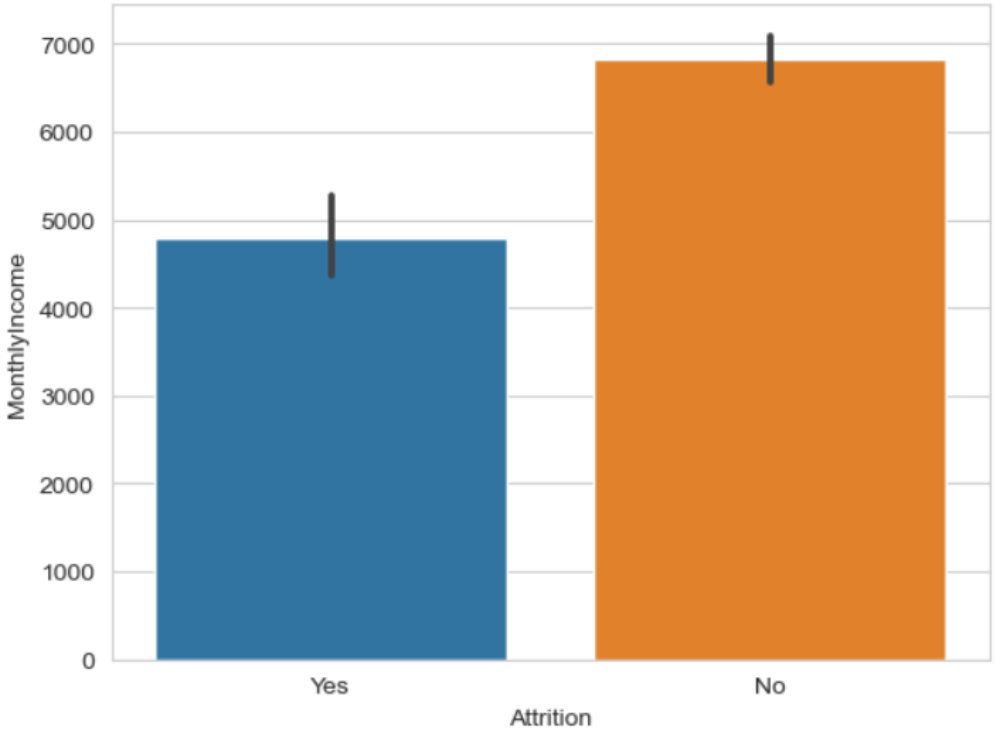
a higher overall population. This insight could prompt further investigation into the reasons behind the differences in attrition rates between these departments.

```
In [7]: sns.countplot(data=df,x='Department',hue='Attrition')
Out[7]: <Axes: xlabel='Department', ylabel='count'>
```



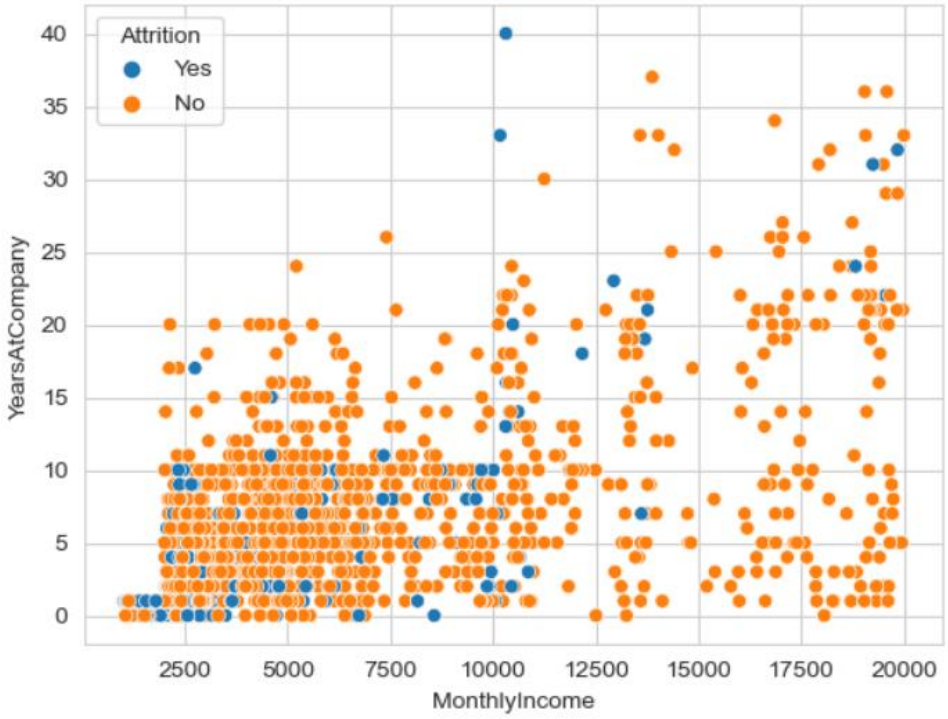
15. In this case, you conducted a bivariate analysis focusing on two categorical variables. To visualize the relationship between these variables, you used a count plot. This type of plot is effective for displaying the frequency of observations across different categories, allowing you to compare how the values of one categorical variable vary with those of another. This approach can provide valuable insights into patterns and trends within the data, such as how attrition rates differ across various departments.
16. To compare a numerical column, like salary, with a categorical column, such as employee attrition, you can conduct a bivariate analysis using a bar plot.
17. This visualization helps you understand whether salary levels have any impact on employee turnover. By displaying the average salary for employees who left the organization versus those who stayed, you can identify trends or patterns, such as whether higher or lower salaries are associated with higher attrition rates.
18. This approach allows you to gain insights into how salary influences employee decisions to leave the company.
19. So the monthly income has a direct impact on the person, whether he is leaving the job or not.

```
In [8]: # 1 is numerical & another is categorical
        sns.barplot(data=df,x='Attrition',y='MonthlyIncome')
        plt.show()
```



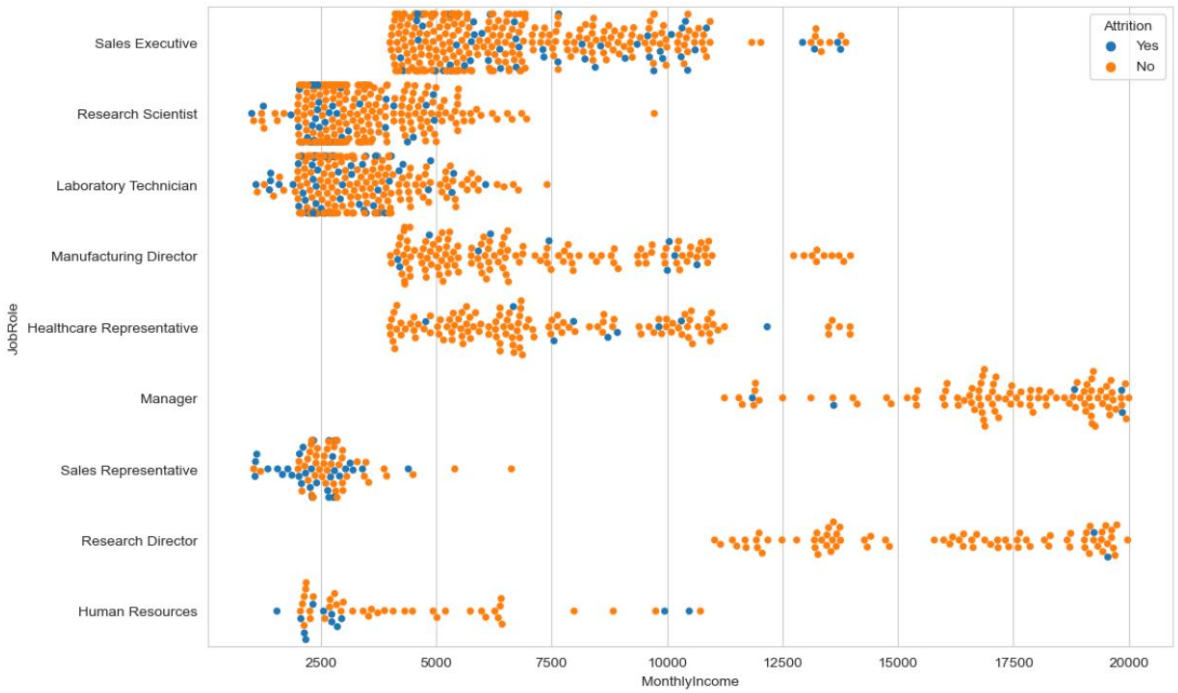
20. To analyze the relationship between two numerical variables, such as experience and monthly income, you can use a scatter plot. This type of visualization allows you to see how one continuous variable affects another. In this case, you can plot the years of experience on the x-axis and the monthly income on the y-axis.
21. By examining the scatter plot, you can identify patterns, trends, or correlations between experience and income, such as whether employees with more experience tend to earn higher monthly salaries. This analysis provides valuable insights into how experience levels relate to income distribution within the organization.
22. It seems you're observing a positive relationship between the years of experience at the company and an increase in salary. This suggests that as employees gain more experience, their monthly income tends to rise.
23. You can conclude that most employees earn between 2,500 and 10,000, indicating this income range is the most common salary band within the organization.

```
In [9]: # 2 continuous
sns.scatterplot(data=df,x='MonthlyIncome',y='YearsAtCompany',hue='Attrition')
plt.show()
```



24. A swarm plot is a powerful visualization tool available in the Seaborn library that helps to display the distribution of data points in a categorical context while also showing individual data points. Unlike a scatter plot, which can overlap points, the swarm plot arranges points to avoid overlap, allowing you to see the density and distribution of values within each category clearly.
25. This is particularly useful when analyzing the relationship between a numerical variable (like salary) and a categorical variable (like department), as it provides a detailed view of how values are spread across different categories.

```
In [10]: # swarm plot
# scatter plot for categorical and numerical data
plt.figure(figsize=(12,8))
sns.swarmplot(data=df, y='JobRole',x='MonthlyIncome',hue='Attrition')
plt.show()
```

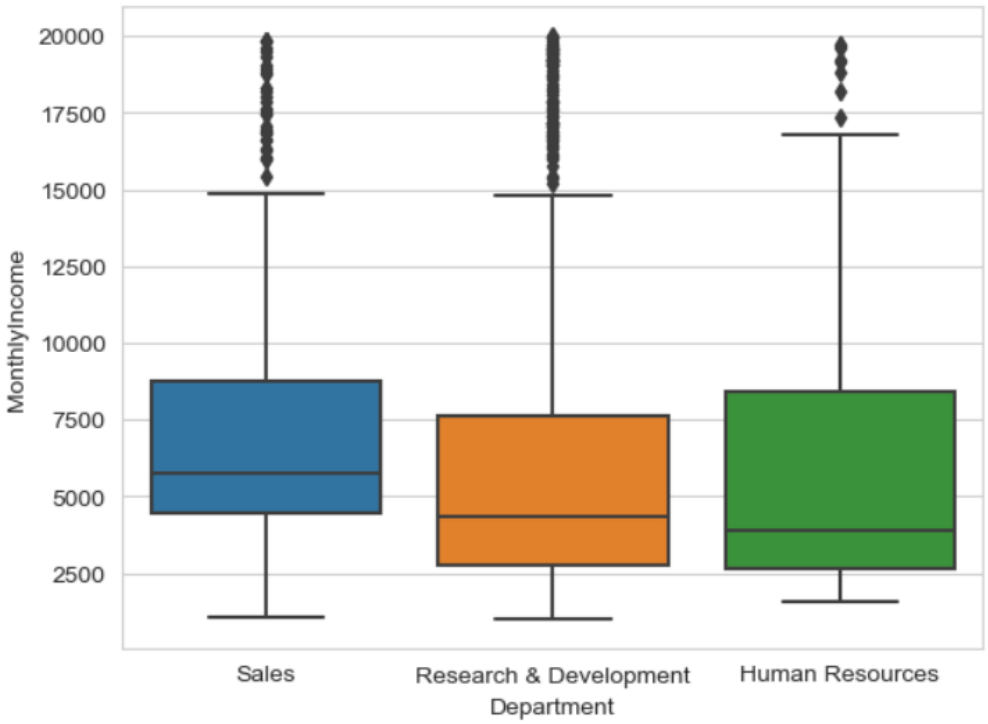


26. A box plot, also known as a whisker plot, is a great tool for visualizing the distribution of a dataset. It provides insights into the central tendency, variability, and potential outliers of the data.

Box plots can be utilized for both univariate and multivariate analysis:

- **Univariate Analysis**:
    - In this context, a box plot can visualize the distribution of a single numerical variable. It helps to display the median, quartiles, and any outliers present within that variable, allowing for a clear understanding of its overall distribution.
- **Multivariate Analysis**:
    - When combined with categorical data, box plots become even more informative. They allow for the comparison of a numerical variable across different categories, revealing patterns or trends associated with each group. This helps in understanding how the numerical values differ based on categorical classifications.

```
In [11]: sns.boxplot(x='Department',y='MonthlyIncome',data=df)
         plt.show()
```



27. Another type of plot commonly used in data analysis is the **joint plot**. Here's how it functions:

- **Scatter Plot for Two Continuous Values**: A joint plot allows you to visualize the relationship between two continuous variables. It creates a scatter plot that helps identify trends, correlations, and patterns between these variables.
- **Univariate Analysis**: In addition to providing insights into the relationship between two variables, a joint plot also includes histograms or density plots on the axes. This feature allows for univariate analysis on each of the continuous variables, giving you a better understanding of their individual distributions.

In [12]:
```
sns.jointplot(data=df, x='MonthlyIncome',y='Age',kind='scatter')
```

Out[12]:
```
<seaborn.axisgrid.JointGrid at 0x23de61aa4d0>
```