Vechile Insurance Analysis

- Domain : Insurance

- Objective :

The primary objective of this project is to conduct an in-depth Exploratory Data Analysis (EDA) on a dataset related to vehicle insurance. Through this analysis, students will gain valuable insights into the patterns, trends, and factors influencing insurance claims. The project encompasses various aspects of data preprocessing, visualization, and statistical analysis.

- Dataset Overview :

The dataset contains information related to vehicle insurance, including details about insured individuals, their vehicles, and insurance claims. Students will explore columns such as age, gender, region, insurance premiums, policy types, and more. The ultimate goal is to derive meaningful insights that can inform decision-making processes within the insurance domain.

- Project flow :

1. Data Collection
2. Data loading and preprocessing
   - Handling Missing values
   - Handling outliers
   - Handling categorical variable
3. EDA

```
###importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

1. Data Loading and Inspection:

- Understand the structure of the dataset.
- Identify the types of information available.

```
data = pd.read_csv("Vehicle_Insurance.csv")

data.shape
```

```
(381109, 12)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381109 entries, 0 to 381108
Data columns (total 12 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   id                    381109 non-null  int64  
 1   Gender                381109 non-null  object 
 2   Age                   381109 non-null  int64  
 3   Driving_License       381109 non-null  int64  
 4   Region_Code           381109 non-null  float64
 5   Previously_Insured    381109 non-null  int64  
 6   Vehicle_Age           381109 non-null  object 
 7   Vehicle_Damage        381109 non-null  object 
 8   Annual_Premium        381109 non-null  float64
 9   Policy_Sales_Channel  381109 non-null  float64
 10  Vintage               381109 non-null  int64  
 11  Response              381109 non-null  int64  
dtypes: float64(3), int64(6), object(3)
memory usage: 34.9+ MB
```

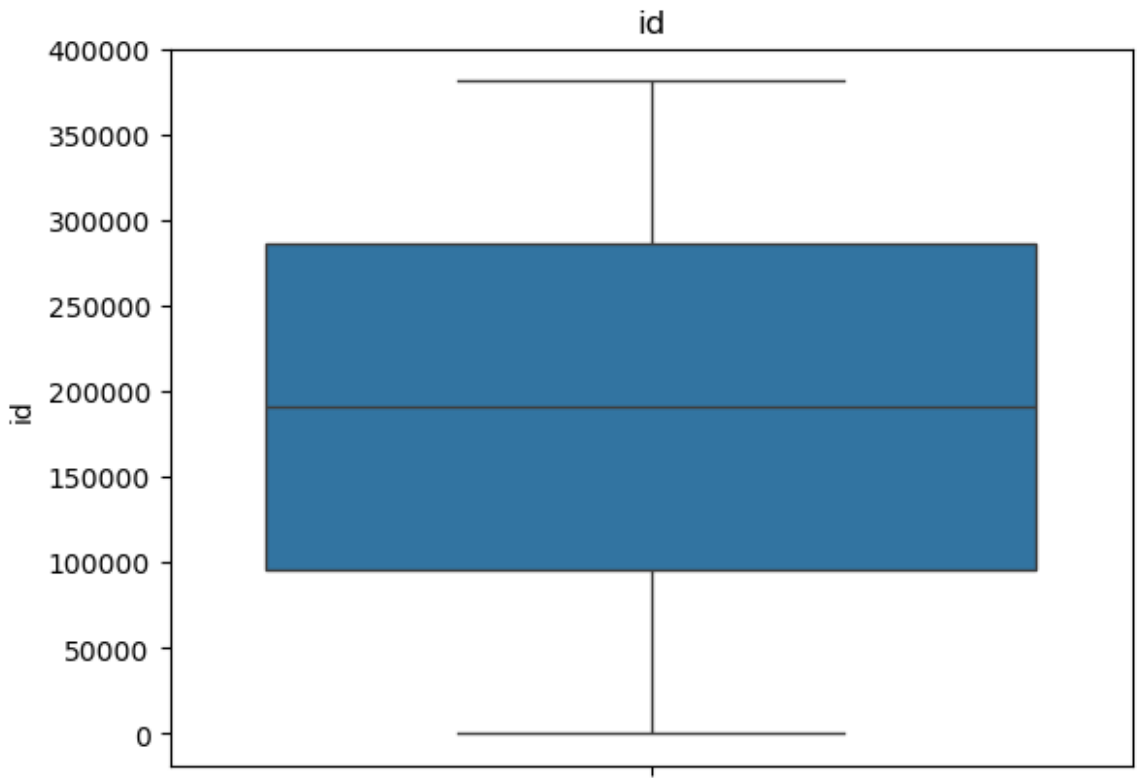
2. Data Cleaning:

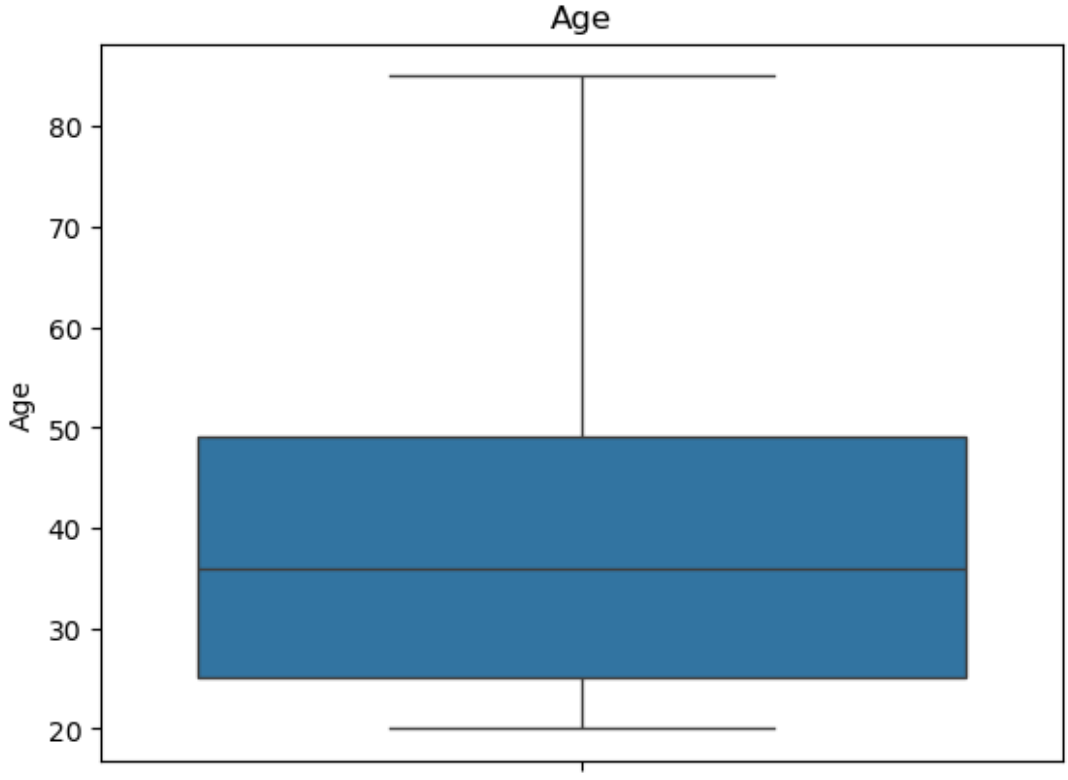
- Handle missing values and outliers appropriately

```
data.isnull().sum()
```

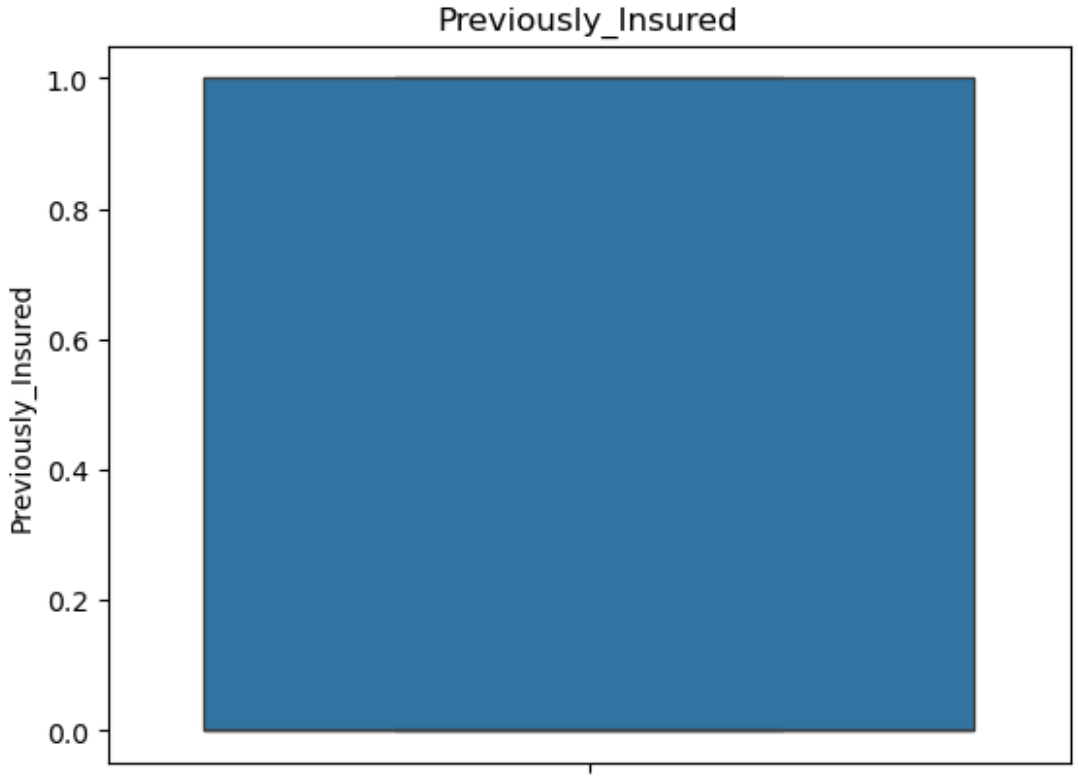
```
id                      0
Gender                  0
Age                     0
Driving_License         0
Region_Code             0
Previously_Insured      0
Vehicle_Age             0
Vehicle_Damage          0
Annual_Premium          0
Policy_Sales_Channel    0
Vintage                 0
Response                0
dtype: int64
```

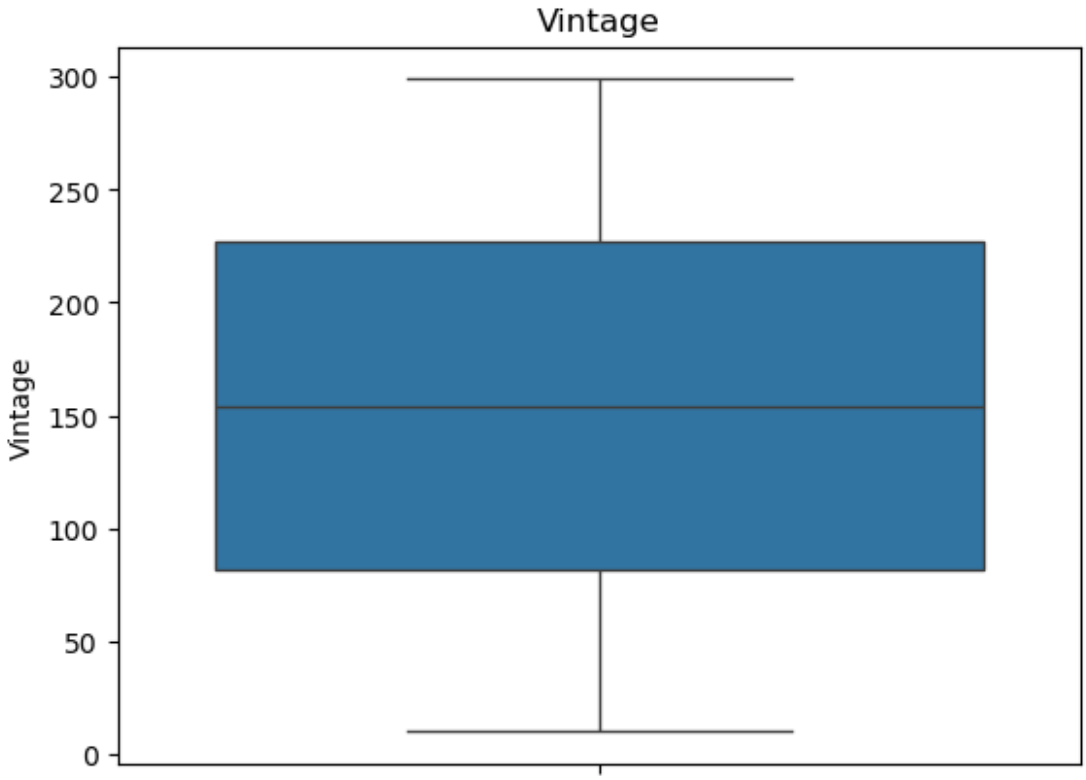
```
for i in data.select_dtypes("int64", "float64"):
    sns.boxplot(data[i])
    plt.title(f"{i}")
    plt.show()
```

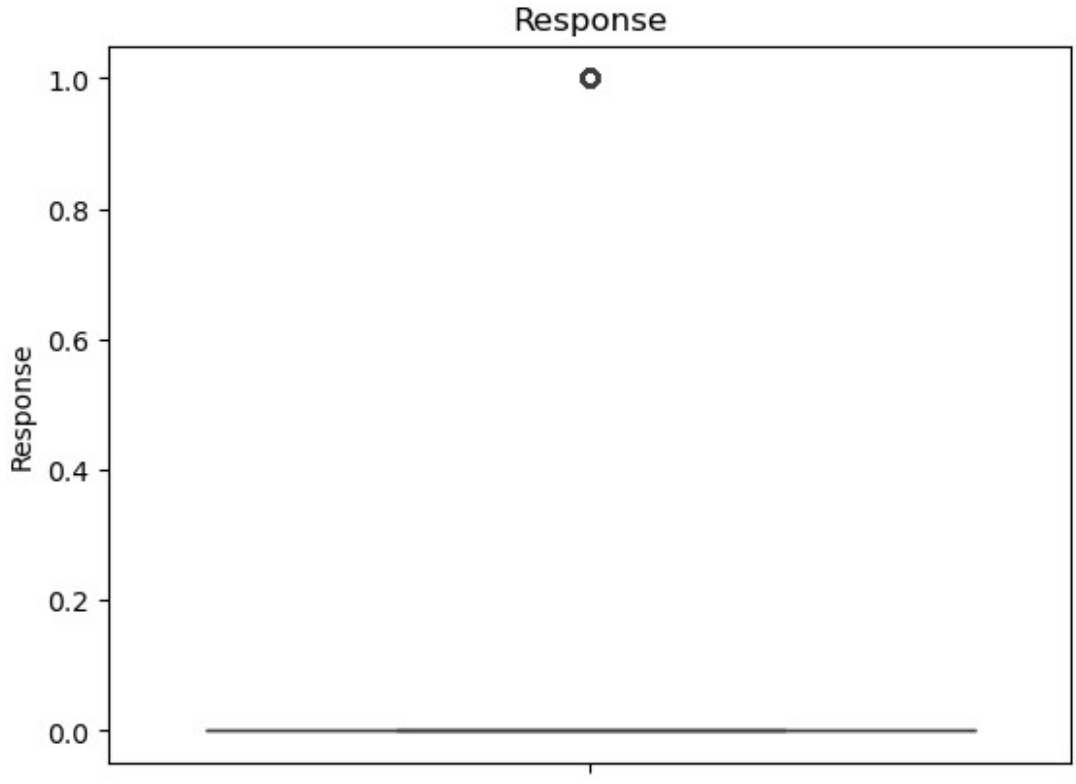

id
400000
350000
300000
250000
200000
150000
100000
50000
0
id


Age
80
70
60
50
40
30
20
Age

Driving_License
Driving_License
1.0
0.8
0.6
0.4
0.2
0.0


Previously_Insured
Previously_Insured
1.0
0.8
0.6
0.4
0.2
0.0



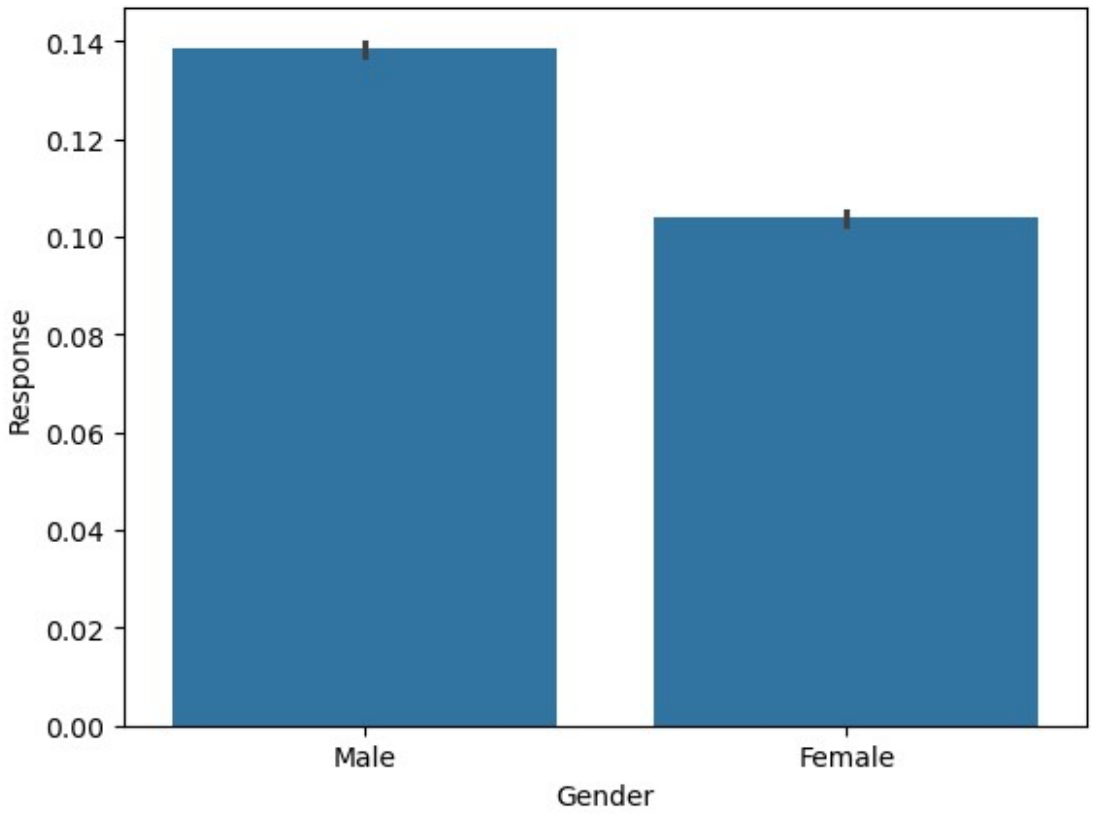



```
### There are no missing values and no outliers.
```

3. Data Visualization:

- Utilize various visualization techniques to explore the distribution of key variables.

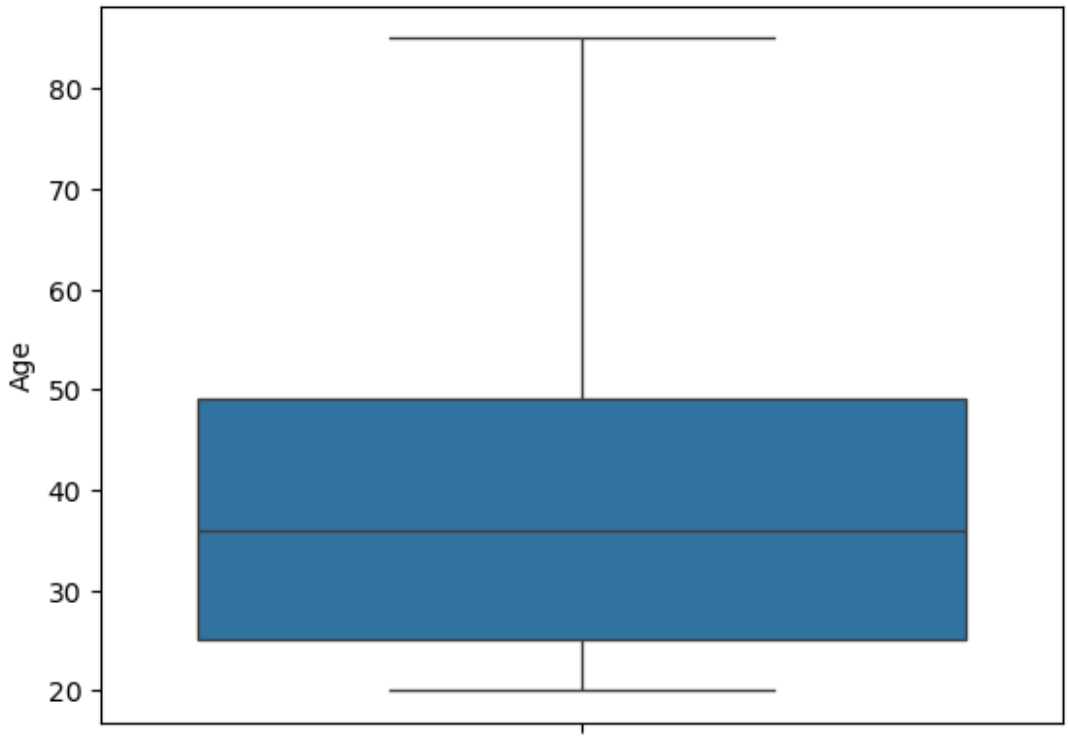
```
### Which gender is giving more response
sns.barplot(data, x = "Gender", y = "Response")
plt.show()
```



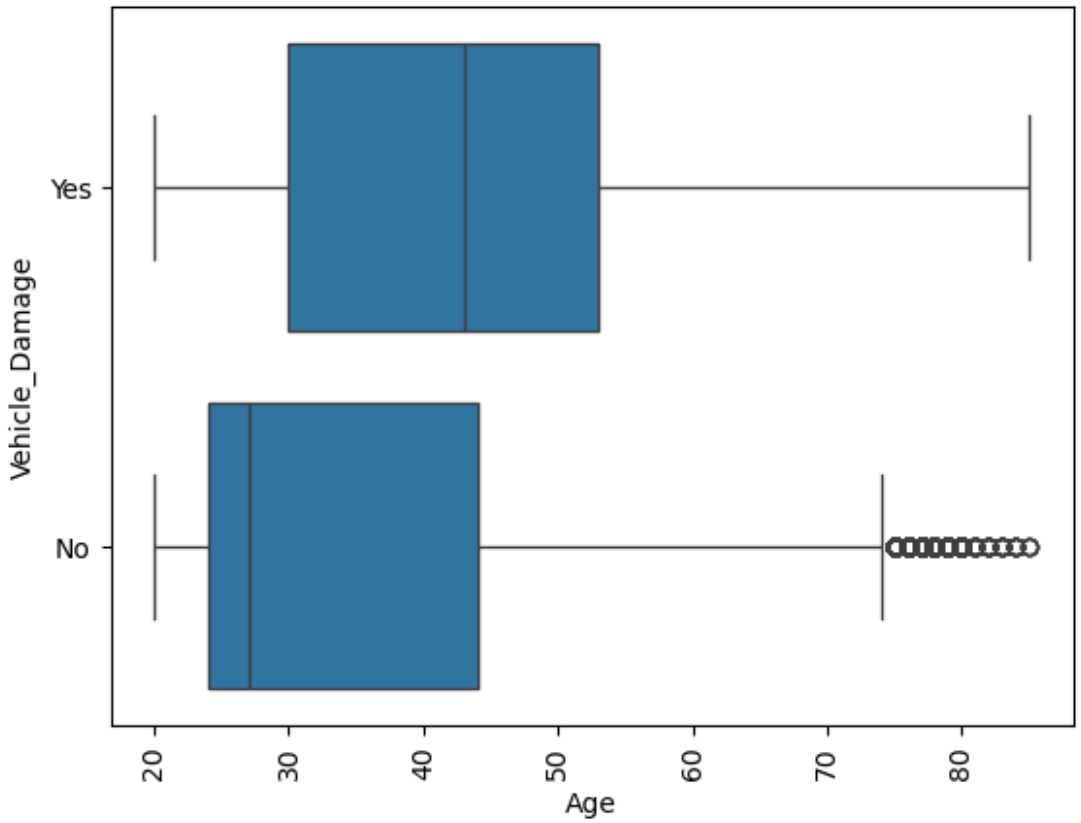
4. Age Distribution:

- Analyze the age distribution within the dataset and its impact on insurance claims.

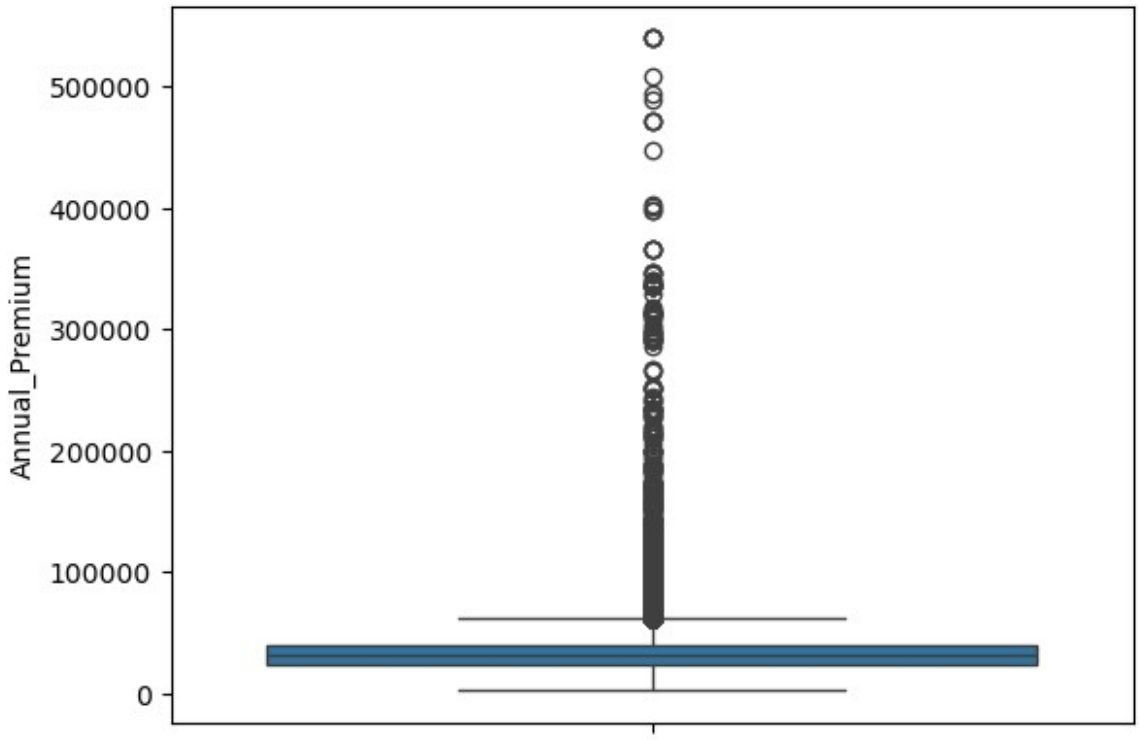
```
sns.boxplot(data["Age"])
plt.show()
```



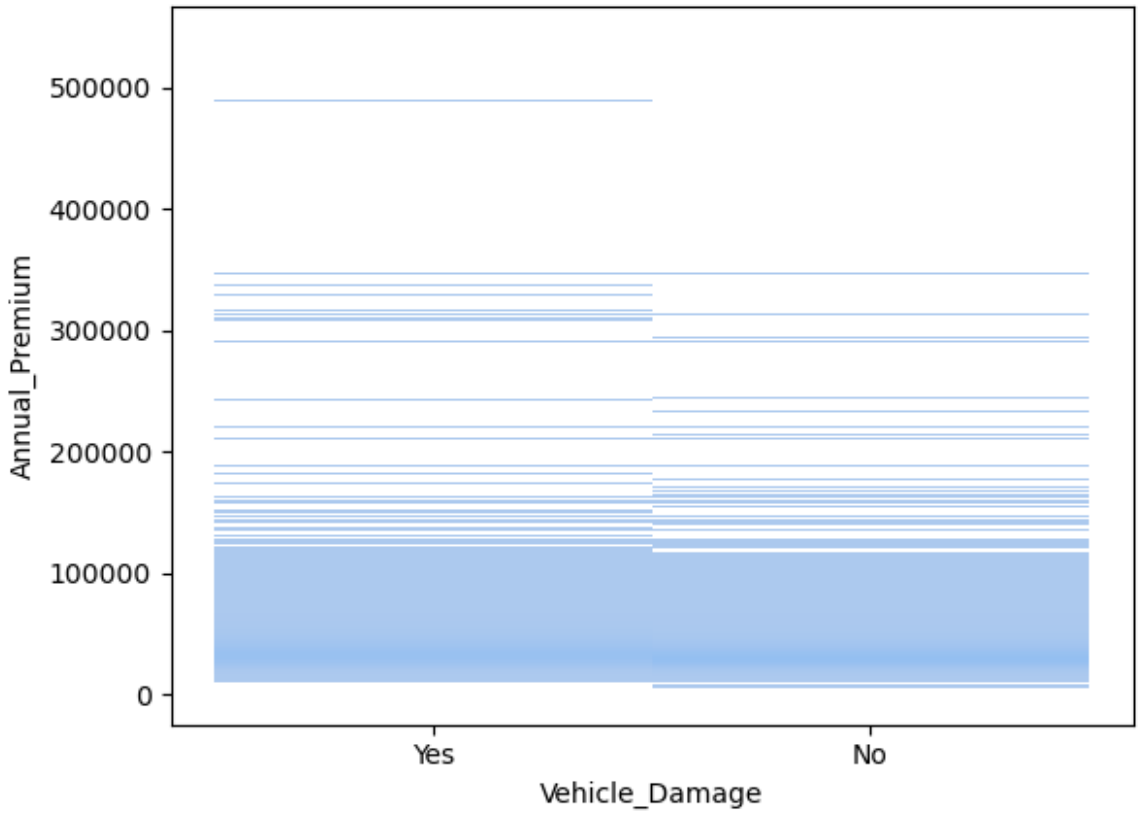

```
sns.boxplot(data, x = "Age", y = "Vehicle_Damage")
plt.xticks(rotation = 90)
plt.show()
```




```
sns.boxplot(data["Annual_Premium"])
plt.show()
```




```
sns.histplot(data, x = "Vehicle_Damage", y = "Annual_Premium")
plt.show()
```

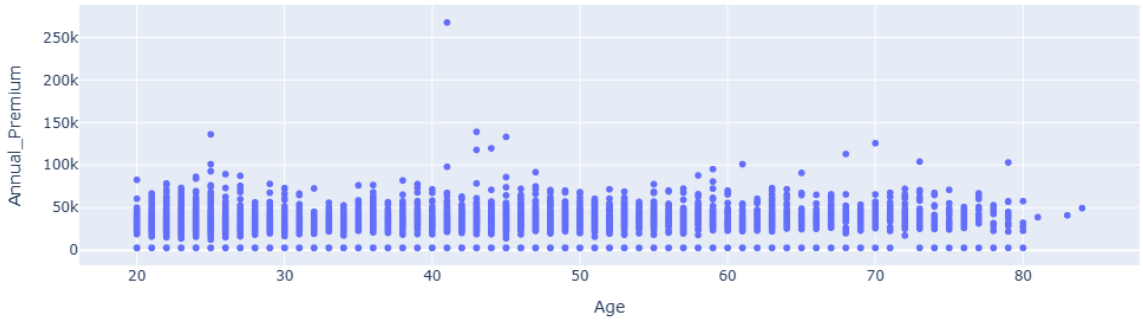



```
!pip install plotly
```

```
Requirement already satisfied: plotly in c:\users\hp\anaconda3\lib\
site-packages (5.24.1)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\hp\
anaconda3\lib\site-packages (from plotly) (8.2.3)
Requirement already satisfied: packaging in c:\users\hp\anaconda3\lib\
site-packages (from plotly) (24.1)
```

```
import plotly.express as px

figure = px.scatter(data.head(5000), x = "Age", y = "Annual_Premium")
figure.show()
```


250k
200k
150k
100k
50k
0
Annual_Premium
20
30
40
50
60
70
80
Age