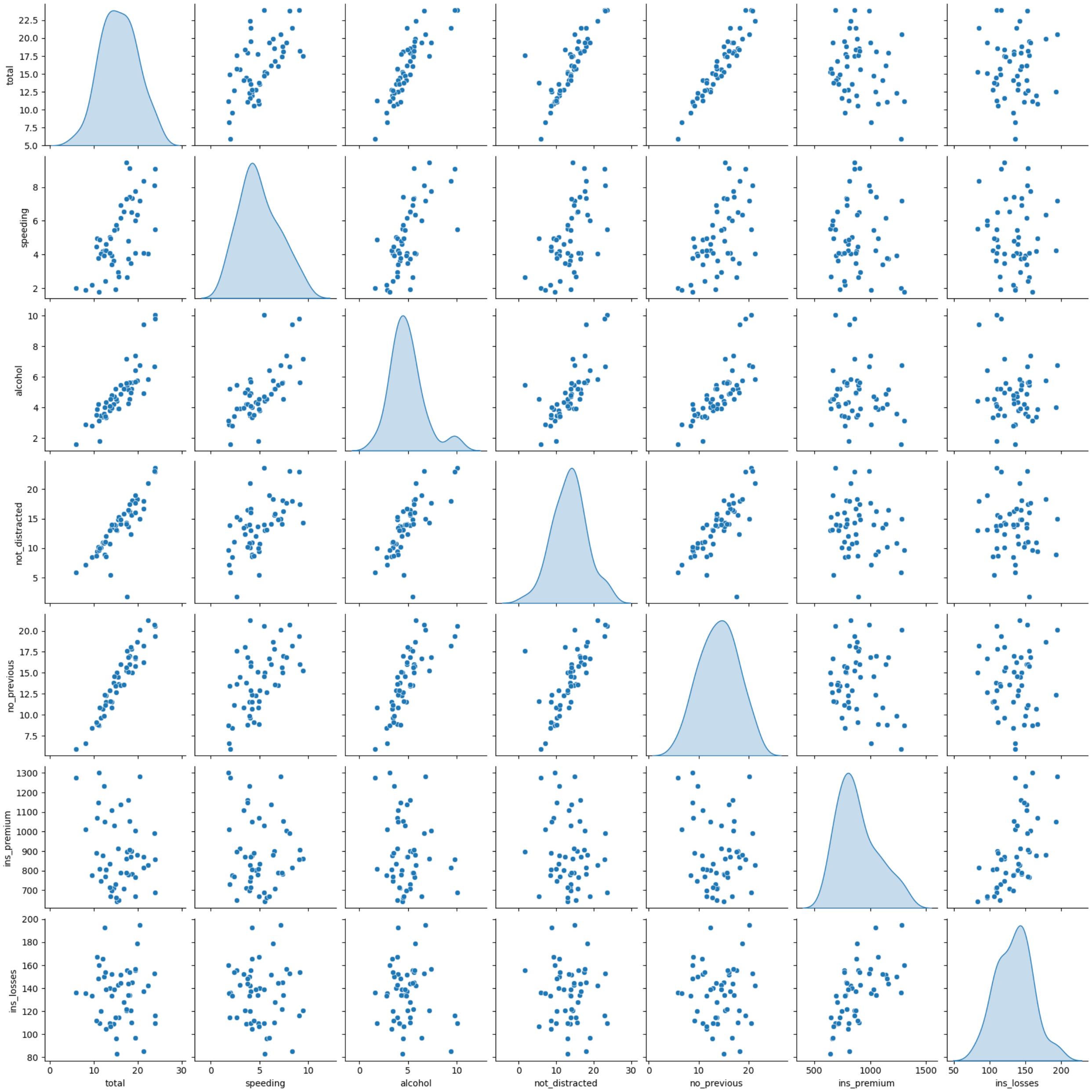
```
# Step 1: Import the necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Step 2: Import the dataset
df = sns.load_dataset ('car_crashes')

# Step 3: Handling null values (if any)
# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)

Missing Values:
 total             0
speeding          0
alcohol           0
not_distracted    0
no_previous       0
ins_premium       0
ins_losses        0
abbrev            0
dtype: int64

# No missing values found in this dataset, so no need for further
handling.

# Step 4: Separate Dependent and Independent Variables
X = df.drop(columns=['total', 'abbrev'])  # Independent variables
y = df['total']  # Dependent variable

# Step 5: Encoding
# The provided dataset appears to be entirely numeric after
preprocessing, so no encoding is required.

# Step 6: Splitting into training and testing set
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Step 7: Feature Scaling (if necessary)
# Depending on the algorithms you plan to use, you might need to scale
the features.
# For instance, you can use StandardScaler to scale the features.
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

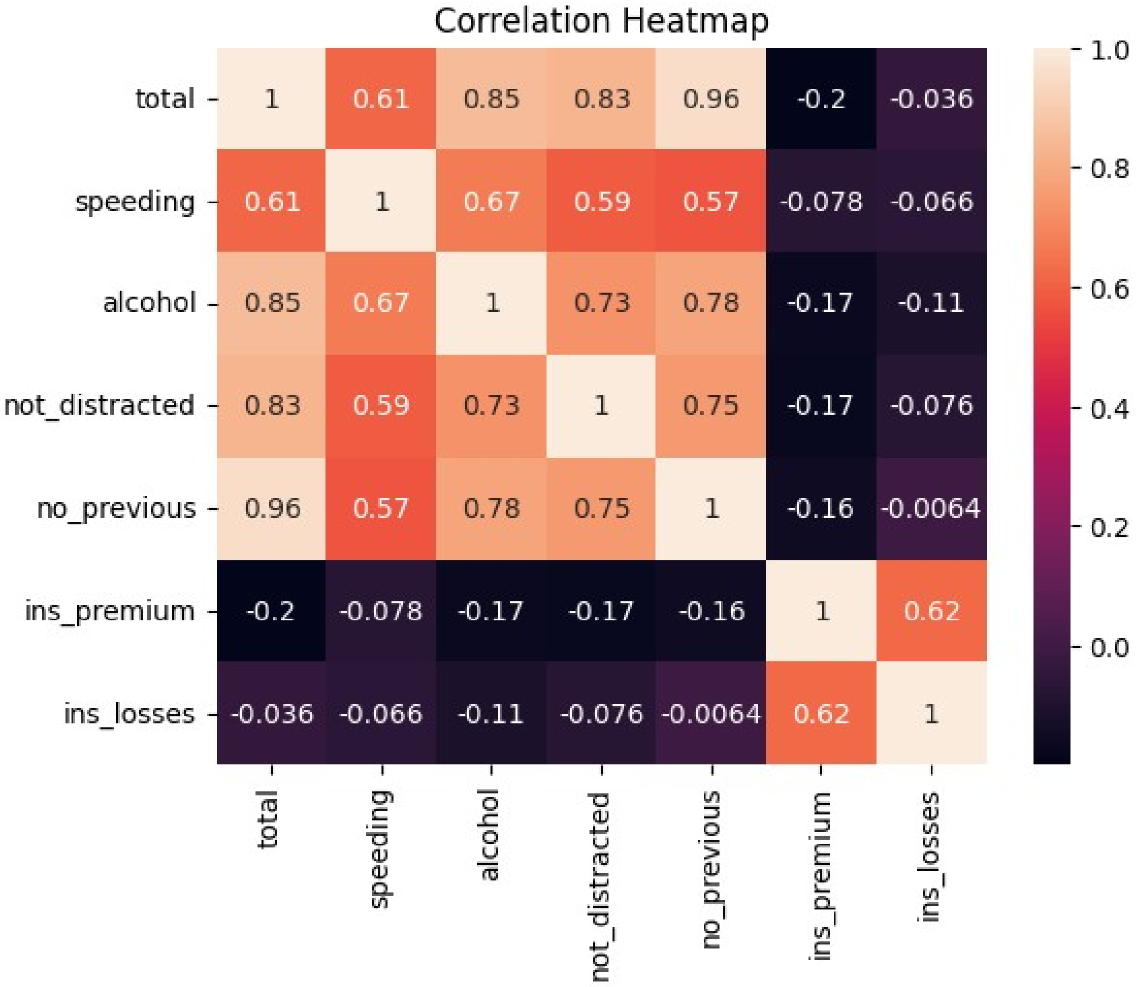
```
# Data Visualization:

#1. Plot a pair plot to visualize relationships between variables
sns.pairplot(df, diag_kind='kde')
plt.show()
```



```
# Conclusions:
# The pair plot helps us visualize the distribution of individual
variables and relationships between them.
# For example, you can observe how variables like 'alcohol' and
'speeding' correlate with 'total' and are approximately linear.
```

```
#2. Plot a heatmap to visualize the correlation matrix
correlation_matrix = df.corr()
sns.heatmap(correlation_matrix, annot=True)
plt.title("Correlation Heatmap")
plt.show()
```

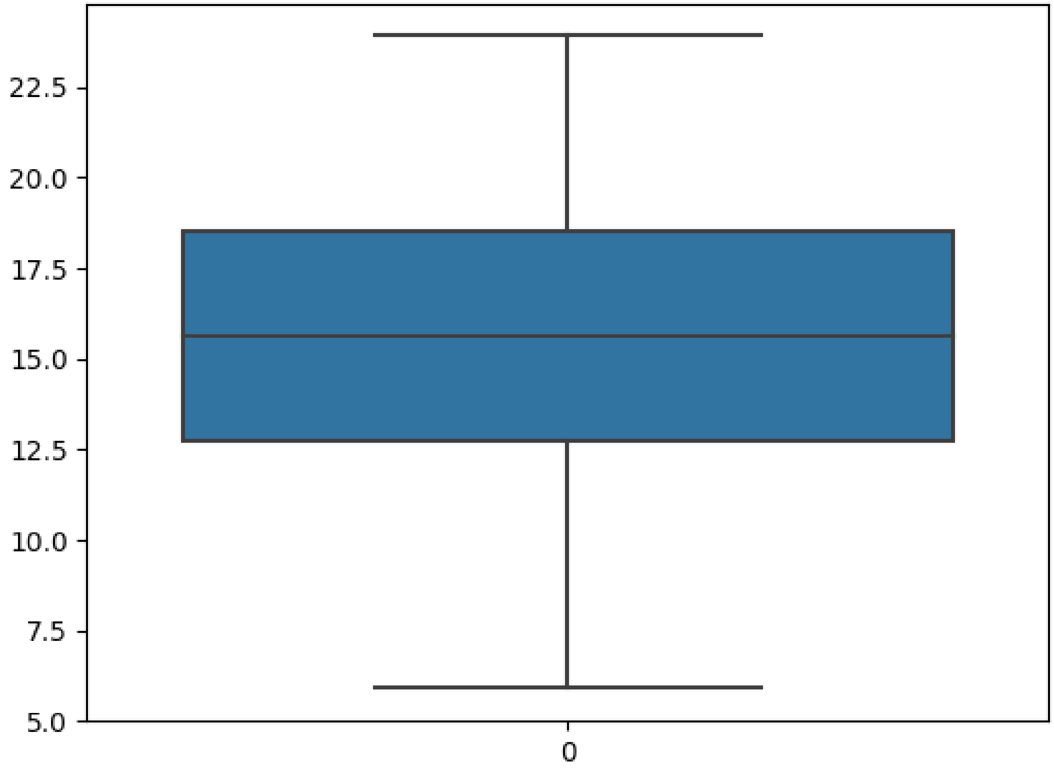
```
<ipython-input-19-dcd90b0c5a39>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  correlation_matrix = df.corr()
```



```
# Conclusions:
#The correlation heatmap provides insights into the strength and
direction of correlations between variables.
#Strong positive correlations are shown in warmer colors, while
negative correlations are in cooler colors.
#For instance, 'alcohol' and 'total' have a strong positive
correlation.
```

```
#3. Box plot for outliers
sns.boxplot(df.total)
#plt.title("Box Plot for 'total' Variable")
```

```
<Axes: >
```



```
#Outliers are data points that significantly deviate from the majority
of the data.
#From the above box plot for "total", we can say that there no
outliers.
```