# Deena 20104016

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pp
```

# Problem Statement

# LINEAR REGRESSION

In [2]:
```
a = pd.read_csv("vehicle.csv")
```

Out[2]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | l |
|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | lounge | 51.0 | 882.0 | 25000.0 | 1.0 | 44.907242 | 8.6115598 |
| **1** | 2.0 | pop | 51.0 | 1186.0 | 32500.0 | 1.0 | 45.666359 | 12.241889 |
| **2** | 3.0 | sport | 74.0 | 4658.0 | 142228.0 | 1.0 | 45.503300 | 11.417 |
| **3** | 4.0 | lounge | 51.0 | 2739.0 | 160000.0 | 1.0 | 40.633171 | 17.634609 |
| **4** | 5.0 | pop | 73.0 | 3074.0 | 106880.0 | 1.0 | 41.903221 | 12.495650 |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **1544** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | len |
| **1545** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | con |
| **1546** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | Null valu |
| **1547** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | f |
| **1548** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | sea |

1549 rows × 11 columns

# HEAD

In [24]:
```
a.head(8)
```

Out[24]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | lon |
|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | lounge | 51.0 | 882.0 | 25000.0 | 1.0 | 44.907242 | 8.611559868 |
| **1** | 2.0 | pop | 51.0 | 1186.0 | 32500.0 | 1.0 | 45.666359 | 12.24188995 |
| **2** | 3.0 | sport | 74.0 | 4658.0 | 142228.0 | 1.0 | 45.503300 | 11.41784 |
| **3** | 4.0 | lounge | 51.0 | 2739.0 | 160000.0 | 1.0 | 40.633171 | 17.63460922 |
| **4** | 5.0 | pop | 73.0 | 3074.0 | 106880.0 | 1.0 | 41.903221 | 12.49565029 |
| **5** | 6.0 | pop | 74.0 | 3623.0 | 70225.0 | 1.0 | 45.000702 | 7.68227005 |
| **6** | 7.0 | lounge | 51.0 | 731.0 | 11600.0 | 1.0 | 44.907242 | 8.611559868 |
| **7** | 8.0 | lounge | 51.0 | 1521.0 | 49076.0 | 1.0 | 41.903221 | 12.49565029 |

# Data Cleaning and Preprocessing

In [23]:
```
b=a.head(8)
b
```

Out[23]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | lon |
|---|---|---|---|---|---|---|---|---|
| **0** | 1.0 | lounge | 51.0 | 882.0 | 25000.0 | 1.0 | 44.907242 | 8.611559868 |
| **1** | 2.0 | pop | 51.0 | 1186.0 | 32500.0 | 1.0 | 45.666359 | 12.24188995 |
| **2** | 3.0 | sport | 74.0 | 4658.0 | 142228.0 | 1.0 | 45.503300 | 11.41784 |
| **3** | 4.0 | lounge | 51.0 | 2739.0 | 160000.0 | 1.0 | 40.633171 | 17.63460922 |
| **4** | 5.0 | pop | 73.0 | 3074.0 | 106880.0 | 1.0 | 41.903221 | 12.49565029 |
| **5** | 6.0 | pop | 74.0 | 3623.0 | 70225.0 | 1.0 | 45.000702 | 7.68227005 |
| **6** | 7.0 | lounge | 51.0 | 731.0 | 11600.0 | 1.0 | 44.907242 | 8.611559868 |
| **7** | 8.0 | lounge | 51.0 | 1521.0 | 49076.0 | 1.0 | 41.903221 | 12.49565029 |

In [22]:
```
a.describe()
```

Out[22]:

| | ID | engine_power | age_in_days | km | previous_owners | lat |
|---|---|---|---|---|---|---|
| **count** | 1538.000000 | 1538.000000 | 1538.000000 | 1538.000000 | 1538.000000 | 1538.000000 |
| **mean** | 769.500000 | 51.904421 | 1650.980494 | 53396.011704 | 1.123537 | 43.541361 |
| **std** | 444.126671 | 3.988023 | 1289.522278 | 40046.830723 | 0.416423 | 2.133518 |
| **min** | 1.000000 | 51.000000 | 366.000000 | 1232.000000 | 1.000000 | 36.855839 |
| **25%** | 385.250000 | 51.000000 | 670.000000 | 20006.250000 | 1.000000 | 41.802990 |
| **50%** | 769.500000 | 51.000000 | 1035.000000 | 39031.000000 | 1.000000 | 44.394096 |
| **75%** | 1153.750000 | 51.000000 | 2616.000000 | 79667.750000 | 1.000000 | 45.467960 |
| **max** | 1538.000000 | 77.000000 | 4658.000000 | 235000.000000 | 4.000000 | 46.795612 |

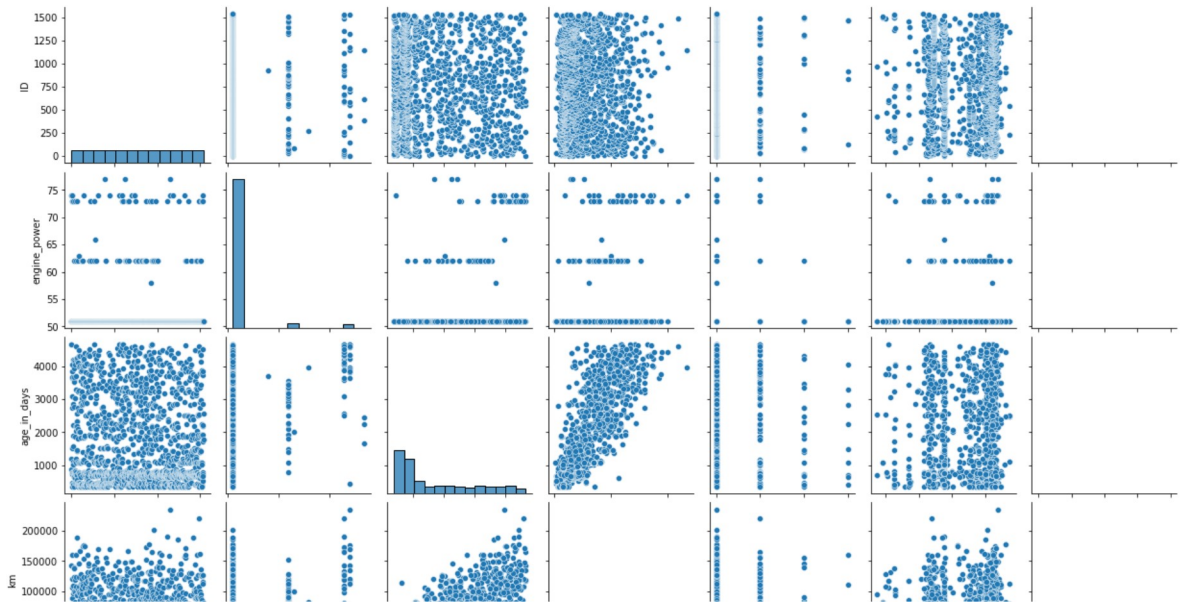
# To display heading

In [21]:
```
a.columns
```

Out[21]:
```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price', 'Unnamed: 9', 'Unnamed: 10'],
      dtype='object')
```

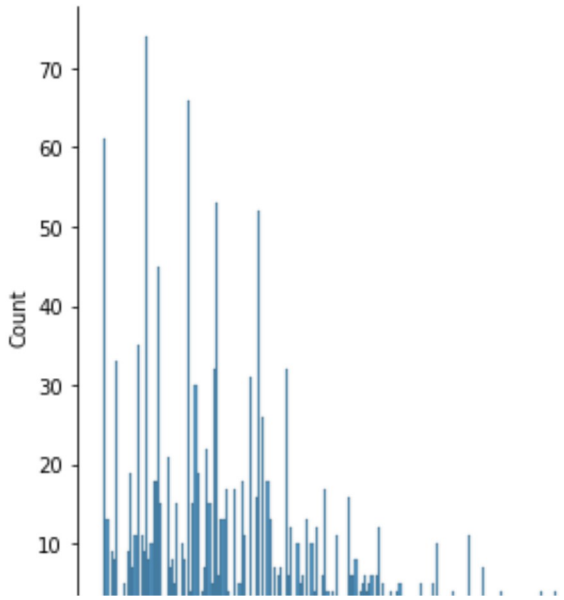
In [20]:
```
sns.pairplot(a)
```

Out[20]:
```
<seaborn.axisgrid.PairGrid at 0x2a6fbf47ac0>
```



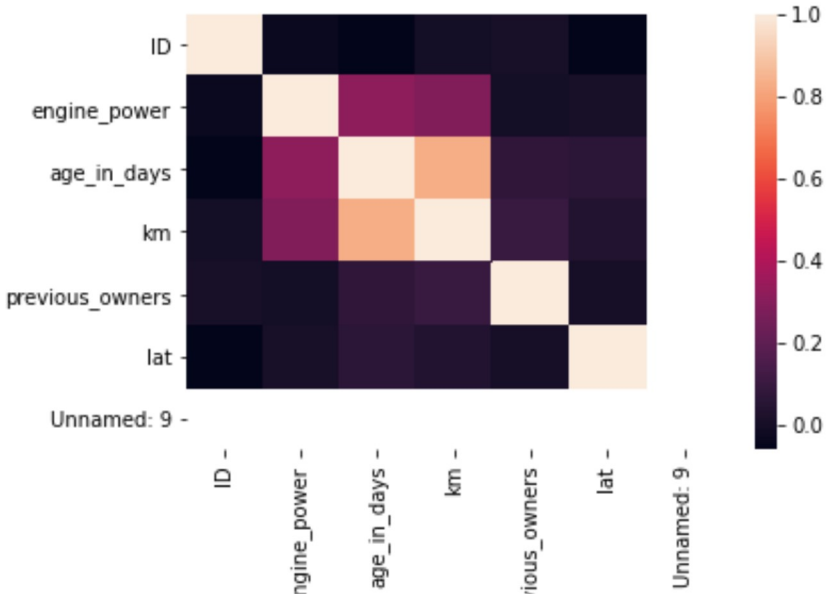
In [5]:
```
sns.displot(a['price'])
```

Out[5]: <seaborn.axisgrid.FacetGrid at 0x2a6fa5d1190>




In [19]:
```
sns.heatmap(a.corr())
```

Out[19]: <AxesSubplot:>




# TO TRAIN THE MODEL - MODEL BUILDING

In [12]:
```
x = b[['price']]
```

In [14]:
```
# to split my dataset into training and test data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [15]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[15]:
```
LinearRegression()
```

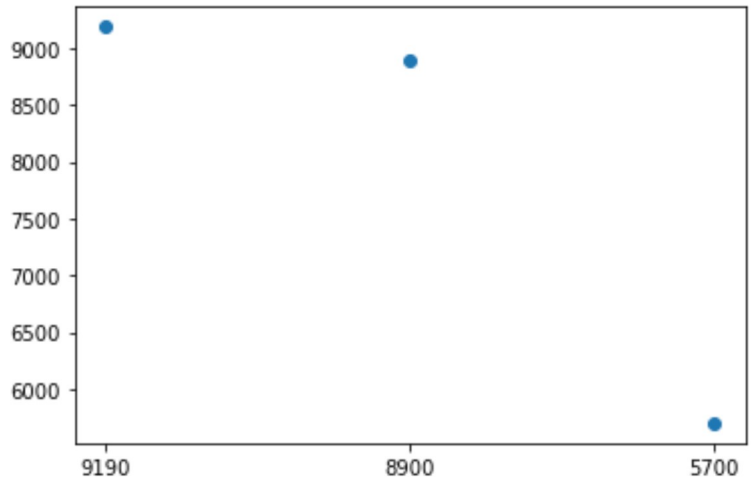
In [16]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[16]:

| | Co-efficient |
|---|---|
| **price** | 1.0 |

In [17]:
```
prediction= lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[17]:
```
<matplotlib.collections.PathCollection at 0x2a6fbe960d0>
```




In [18]:
```
lr.score(x_test,y_test)
```

Out[18]:
```
1.0
```