problem statement

a real esate agent want help to predict the house price for regions in USA. he gave us the daataset to work on to use liner regression model create a model that help him to estimate of what the house would sell sell for

In [ ]:

```
```

DATA COLLECTIN

In [5]:

```
# IMPORT LIBRARIES
import numpy  as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [6]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\fiat500_VehicleSelection_Dataset.csv")
a
```

Out[6]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	l
0	1	lounge	51	882	25000	1	44.907242	8.6115
1	2	pop	51	1186	32500	1	45.666359	12.2418
2	3	sport	74	4658	142228	1	45.503300	11.4178
3	4	lounge	51	2739	160000	1	40.633171	17.6346
4	5	pop	73	3074	106880	1	41.903221	12.4956
...	...	...	...	...	...	...	...	
1533	1534	sport	51	3712	115280	1	45.069679	7.7049
1534	1535	lounge	74	3835	112000	1	45.845692	8.6668
1535	1536	pop	51	2223	60457	1	45.481541	9.4134
1536	1537	lounge	51	2557	80750	1	45.000702	7.6822
1537	1538	pop	51	1766	54276	1	40.323410	17.5682

1538 rows × 9 columns

DATA CLEANING AND PRE-

In [7]:

```
# to find
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1538 entries, 0 to 1537
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ID               1538 non-null   int64  
 1   model            1538 non-null   object 
 2   engine_power     1538 non-null   int64  
 3   age_in_days      1538 non-null   int64  
 4   km               1538 non-null   int64  
 5   previous_owners  1538 non-null   int64  
 6   lat              1538 non-null   float64
 7   lon              1538 non-null   float64
 8   price            1538 non-null   int64  
dtypes: float64(2), int64(6), object(1)
memory usage: 108.3+ KB
```

In [8]:

```
# to display summary of statastic
a.describe()
```

Out[8]:

	ID	engine_power	age_in_days	km	previous_owners	la
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

In [9]:

```
# to display colum heading
a.columns
```

Out[9]:

```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owner
s',
       'lat', 'lon', 'price'],
      dtype='object')
```

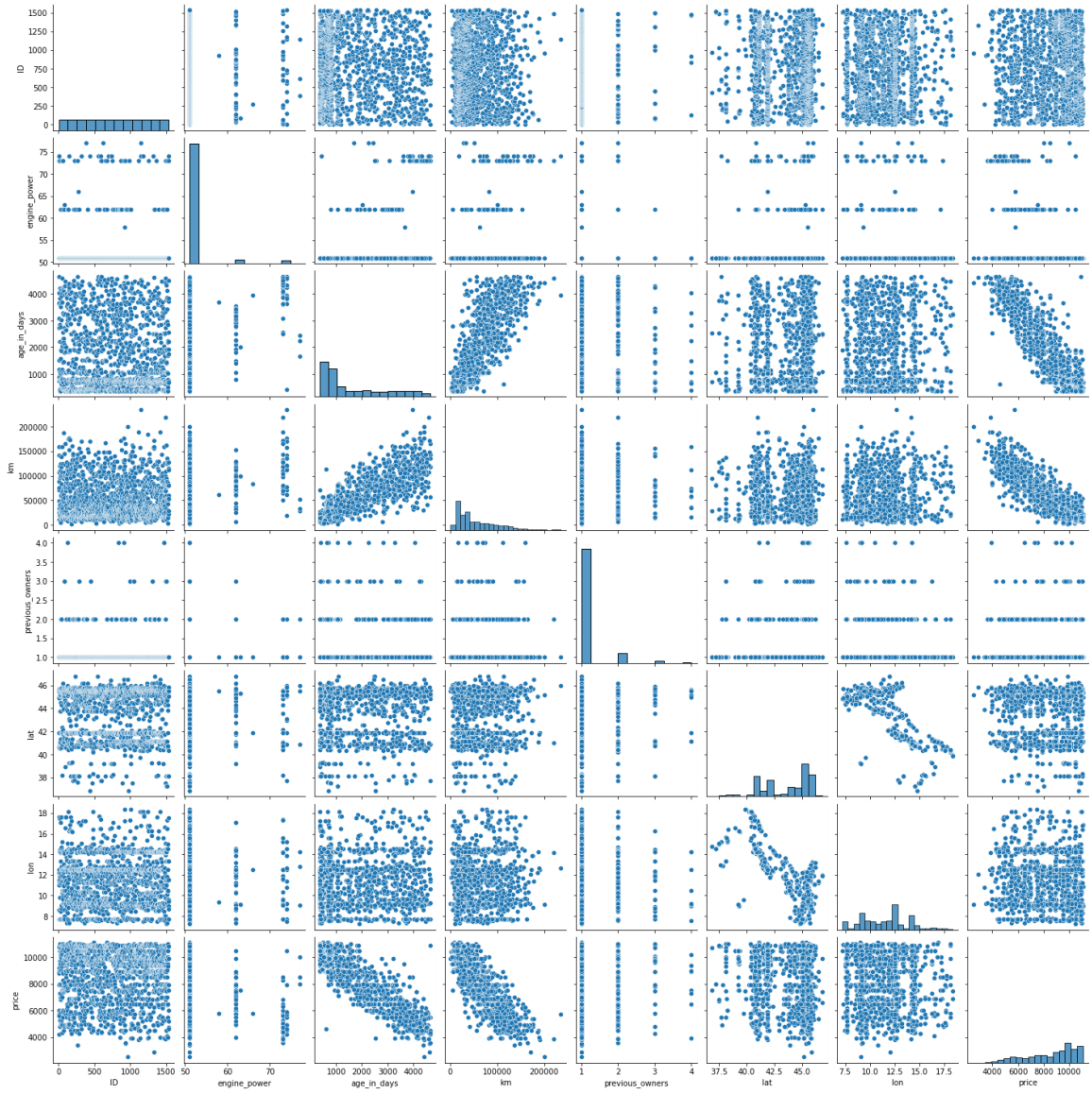
EDA and VISUALIZATION

In [10]:

```
sns.pairplot(a)
```

Out[10]:

```
<seaborn.axisgrid.PairGrid at 0x20cd095ea60>
```

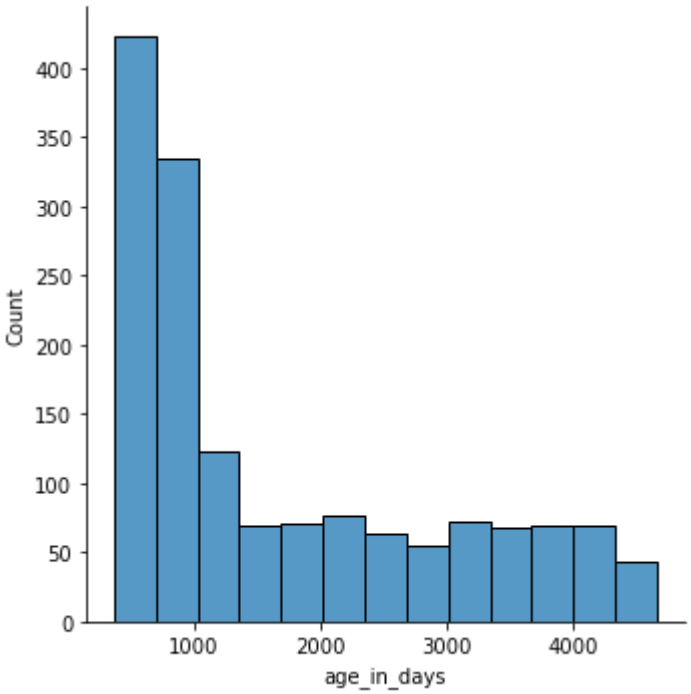


In [12]:

```
sns.displot(a["age_in_days"])
```

Out[12]:

```
<seaborn.axisgrid.FacetGrid at 0x20cd26a4ac0>
```



In [15]:

```
b=a[['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price']]
b
```

Out[15]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	l
0	1	lounge	51	882	25000	1	44.907242	8.6115
1	2	pop	51	1186	32500	1	45.666359	12.2418
2	3	sport	74	4658	142228	1	45.503300	11.4178
3	4	lounge	51	2739	160000	1	40.633171	17.6346
4	5	pop	73	3074	106880	1	41.903221	12.4956
...	...	...	...	...	...	...	...	
1533	1534	sport	51	3712	115280	1	45.069679	7.7049
1534	1535	lounge	74	3835	112000	1	45.845692	8.6668
1535	1536	pop	51	2223	60457	1	45.481541	9.4134
1536	1537	lounge	51	2557	80750	1	45.000702	7.6822
1537	1538	pop	51	1766	54276	1	40.323410	17.5682

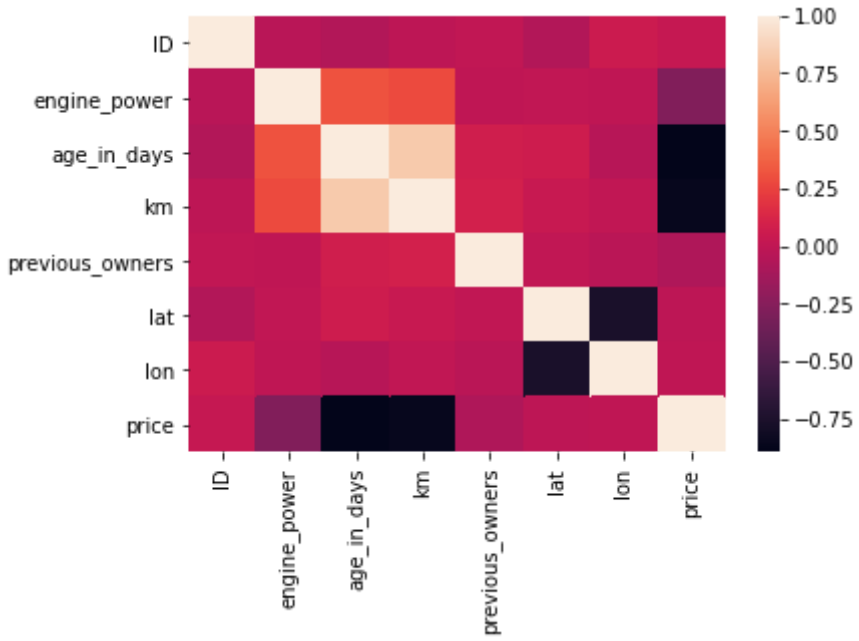
1538 rows × 9 columns

In [16]:

```
sns.heatmap(b.corr())
```

Out[16]:

<AxesSubplot:>



id train the model- model bulding

we are going to train liner hegression model ; we to split out data into two varialbe x and y where x is independent variable (input) and y is depending on x(output) we could ignore address column as it is not required for our model

In [22]:

```
x=a[['ID', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price']]
y=a['age_in_days']
```

In [23]:

```
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 [24]:

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[24]:

```
LinearRegression()
```

In [25]:

```
lr.intercept_
```

Out[25]:

```
2.0463630789890885e-12
```

In [26]:

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

Out[26]:

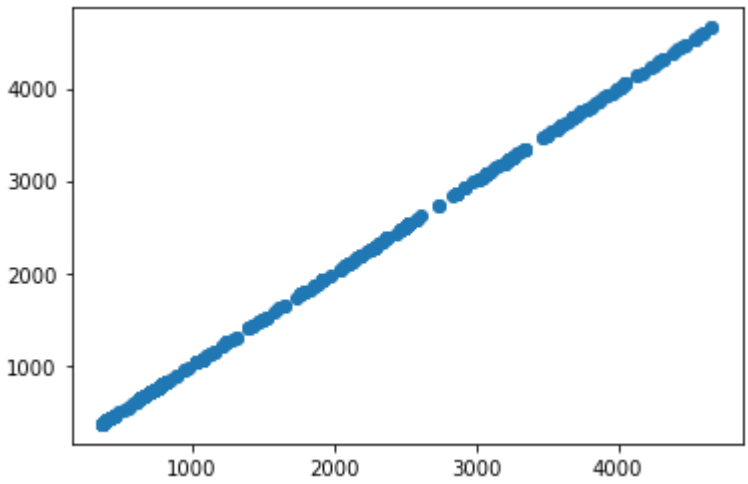
	Co-efficient
ID	-1.758564e-16
engine_power	4.539333e-15
age_in_days	1.000000e+00
km	2.202124e-17
previous_owners	7.998609e-15
lat	-7.561522e-16
lon	1.102698e-14
price	-3.553947e-16

In [27]:

```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[27]:

```
<matplotlib.collections.PathCollection at 0x20cd53bc1f0>
```



In [28]:

```
lr.score(x_test,y_test)
```

Out[28]:

```
1.0
```