In [25]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Ridge,RidgeCV,Lasso
from sklearn.preprocessing import StandardScaler
```

In [26]:
```
df=pd.read_csv(r"C:\Users\SASIDHAR ROYAL\Downloads\fiat500_VehicleSelection_Dataset.csv")
df
```

Out[26]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | lon | price |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | lounge | 51 | 882 | 25000 | 1 | 44.907242 | 8.611560 | 8900 |
| **1** | 2 | pop | 51 | 1186 | 32500 | 1 | 45.666359 | 12.241890 | 8800 |
| **2** | 3 | sport | 74 | 4658 | 142228 | 1 | 45.503300 | 11.417840 | 4200 |
| **3** | 4 | lounge | 51 | 2739 | 160000 | 1 | 40.633171 | 17.634609 | 6000 |
| **4** | 5 | pop | 73 | 3074 | 106880 | 1 | 41.903221 | 12.495650 | 5700 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **1533** | 1534 | sport | 51 | 3712 | 115280 | 1 | 45.069679 | 7.704920 | 5200 |
| **1534** | 1535 | lounge | 74 | 3835 | 112000 | 1 | 45.845692 | 8.666870 | 4600 |
| **1535** | 1536 | pop | 51 | 2223 | 60457 | 1 | 45.481541 | 9.413480 | 7500 |
| **1536** | 1537 | lounge | 51 | 2557 | 80750 | 1 | 45.000702 | 7.682270 | 5990 |
| **1537** | 1538 | pop | 51 | 1766 | 54276 | 1 | 40.323410 | 17.568270 | 7900 |

1538 rows × 9 columns

In [27]:
```
df.head()
```

Out[27]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | lon | price |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | lounge | 51 | 882 | 25000 | 1 | 44.907242 | 8.611560 | 8900 |
| **1** | 2 | pop | 51 | 1186 | 32500 | 1 | 45.666359 | 12.241890 | 8800 |
| **2** | 3 | sport | 74 | 4658 | 142228 | 1 | 45.503300 | 11.417840 | 4200 |
| **3** | 4 | lounge | 51 | 2739 | 160000 | 1 | 40.633171 | 17.634609 | 6000 |
| **4** | 5 | pop | 73 | 3074 | 106880 | 1 | 41.903221 | 12.495650 | 5700 |

In [28]:
```
df.tail()
```

Out[28]:

| | ID | model | engine_power | age_in_days | km | previous_owners | lat | lon | price |
|---|---|---|---|---|---|---|---|---|---|
| **1533** | 1534 | sport | 51 | 3712 | 115280 | 1 | 45.069679 | 7.70492 | 5200 |
| **1534** | 1535 | lounge | 74 | 3835 | 112000 | 1 | 45.845692 | 8.66687 | 4600 |
| **1535** | 1536 | pop | 51 | 2223 | 60457 | 1 | 45.481541 | 9.41348 | 7500 |
| **1536** | 1537 | lounge | 51 | 2557 | 80750 | 1 | 45.000702 | 7.68227 | 5990 |
| **1537** | 1538 | pop | 51 | 1766 | 54276 | 1 | 40.323410 | 17.56827 | 7900 |

In [29]:
```
df.describe()
```

Out[29]:

| | ID | engine_power | age_in_days | km | previous_owners | lat | lon | price |
|---|---|---|---|---|---|---|---|---|
| **count** | 1538.000000 | 1538.000000 | 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 | 11.563428 | 8576.003901 |
| **std** | 444.126671 | 3.988023 | 1289.522278 | 40046.830723 | 0.416423 | 2.133518 | 2.328190 | 1939.958641 |
| **min** | 1.000000 | 51.000000 | 366.000000 | 1232.000000 | 1.000000 | 36.855839 | 7.245400 | 2500.000000 |
| **25%** | 385.250000 | 51.000000 | 670.000000 | 20006.250000 | 1.000000 | 41.802990 | 9.505090 | 7122.500000 |
| **50%** | 769.500000 | 51.000000 | 1035.000000 | 39031.000000 | 1.000000 | 44.394096 | 11.869260 | 9000.000000 |
| **75%** | 1153.750000 | 51.000000 | 2616.000000 | 79667.750000 | 1.000000 | 45.467960 | 12.769040 | 10000.000000 |
| **max** | 1538.000000 | 77.000000 | 4658.000000 | 235000.000000 | 4.000000 | 46.795612 | 18.365520 | 11100.000000 |

In [30]:
```
df.shape
```

Out[30]:
```
(1538, 9)
```

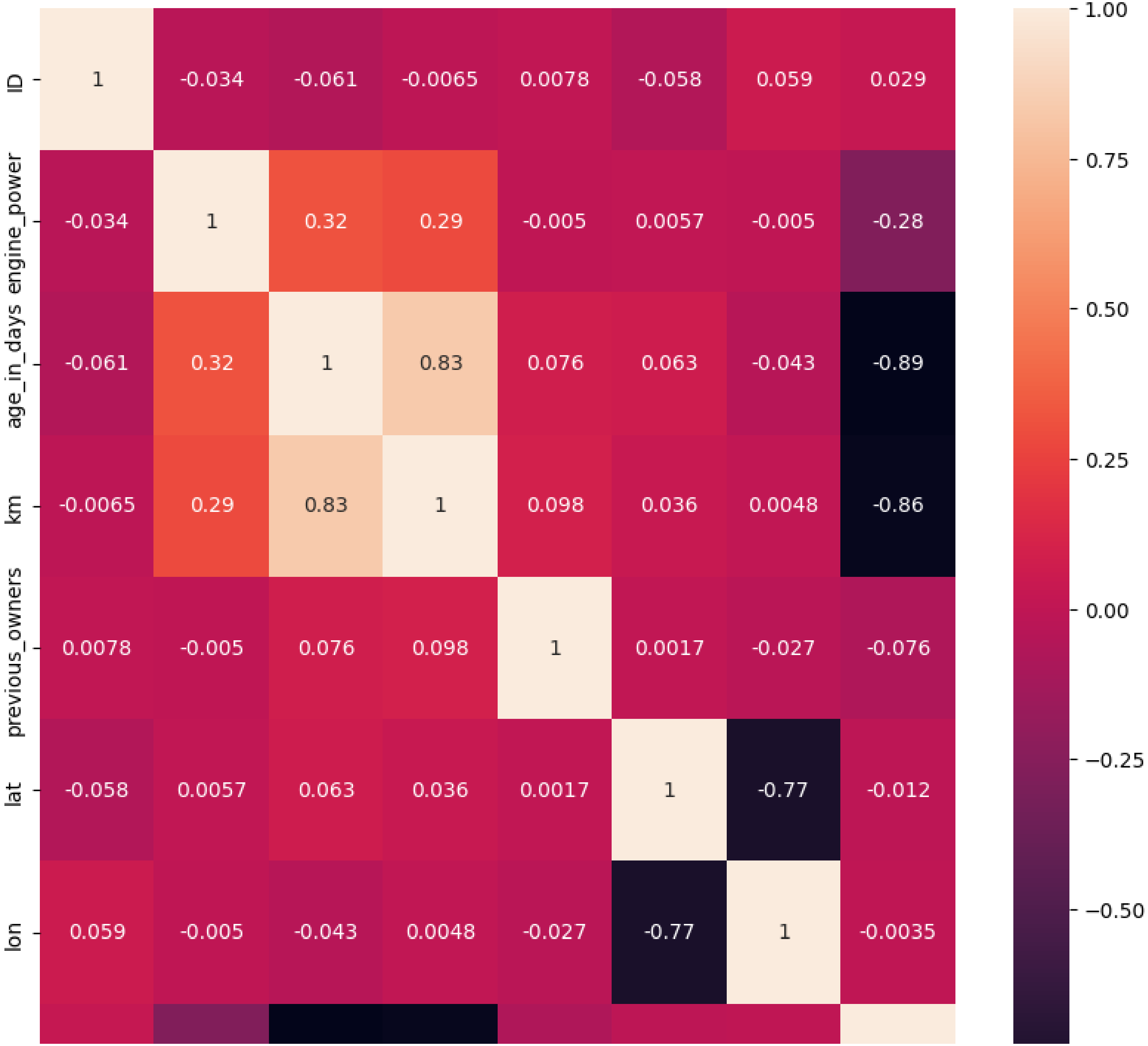
In [31]:
```
df.columns
```

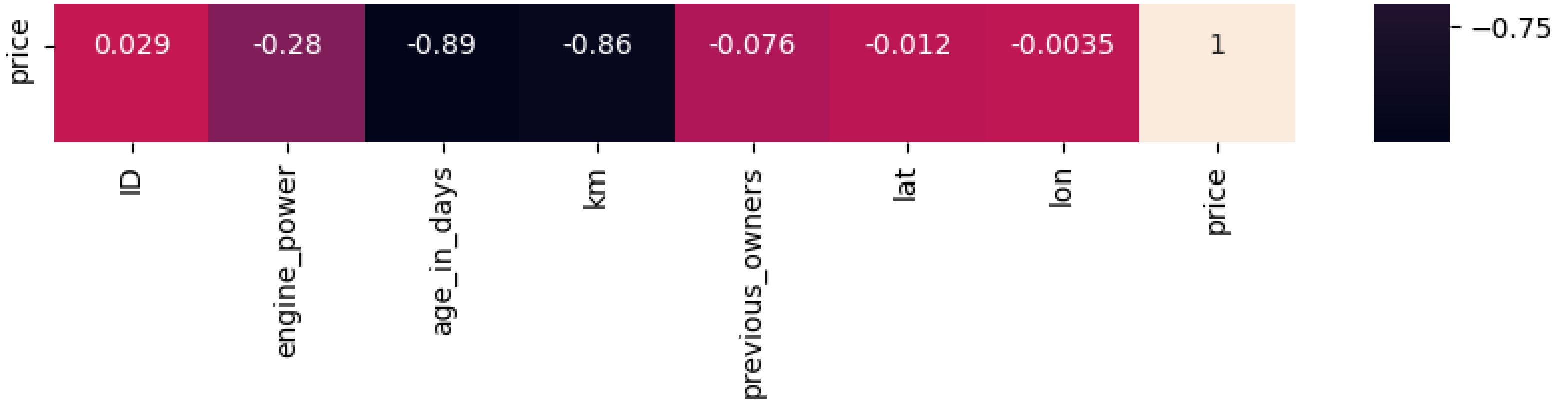
Out[31]:
```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price'],
      dtype='object')
```

In [32]:
```
a=df
a.drop(columns=["model"],inplace=True)
```

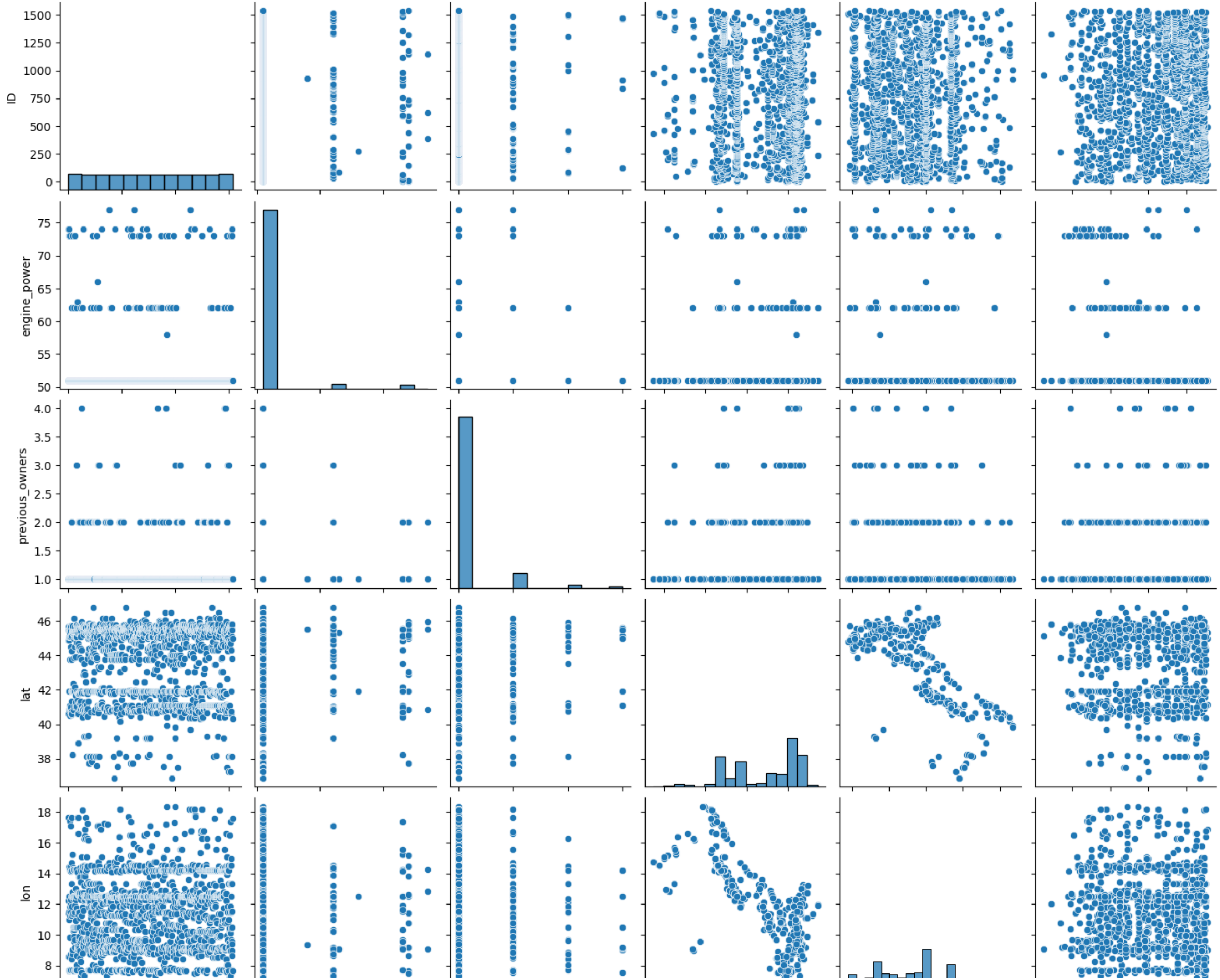
In [33]:
```
plt.figure(figsize = (10, 10))
sns.heatmap(df.corr(), annot = True)
```

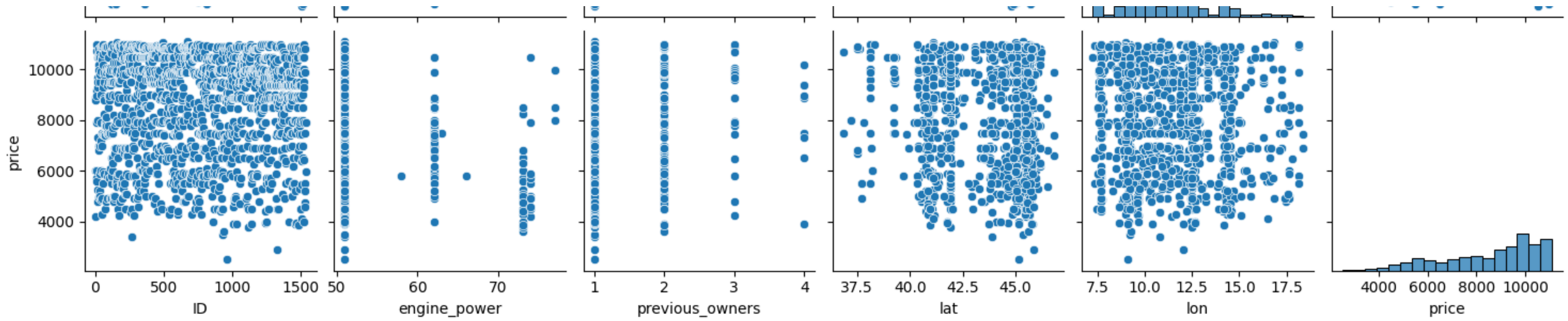
Out[33]: <Axes: >


ID
engine_power
age_in_days
km
previous_owners
lat
lon
1 -0.034 -0.061 -0.0065 0.0078 -0.058 0.059 0.029
-0.034 1 0.32 0.29 -0.005 0.0057 -0.005 -0.28
-0.061 0.32 1 0.83 0.076 0.063 -0.043 -0.89
-0.0065 0.29 0.83 1 0.098 0.036 0.0048 -0.86
0.0078 -0.005 0.076 0.098 1 0.0017 -0.027 -0.076
-0.058 0.0057 0.063 0.036 0.0017 1 -0.77 -0.012
0.059 -0.005 -0.043 0.0048 -0.027 -0.77 1 -0.0035
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50


price
0.029
-0.28
-0.89
-0.86
-0.076
-0.012
-0.0035
1
ID
engine_power
age_in_days
km
previous_owners
lat
lon
price
−0.75

In [34]:
```
df.drop(columns = ["km", "age_in_days"], inplace = True)
#pairplot
sns.pairplot(df)
df.price=np.log(df.price)
```


ID
engine_power
previous_owners
lat
lon




In [35]:
```
features = df.columns[0:2]
target = df.columns[-1]
#X and y values
X = df[features].values
y = df[target].values
#splot
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=17)
print("The dimension of X_train is {}".format(X_train.shape))
print("The dimension of X_test is {}".format(X_test.shape))
#Scale features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test=scaler.transform(X_test)
```

```
The dimension of X_train is (1076, 2)
The dimension of X_test is (462, 2)
```

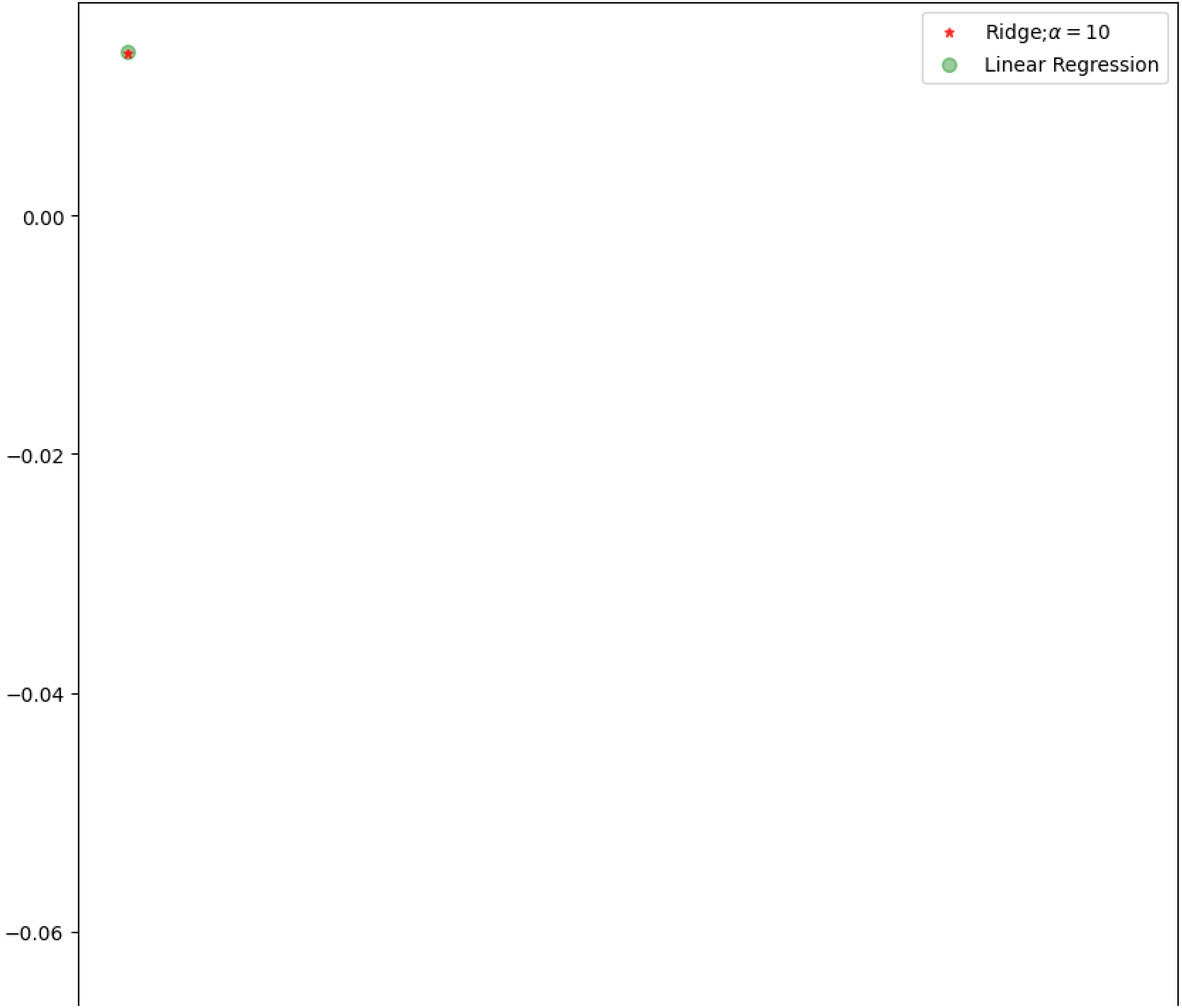
In [36]:
```
#Model
lr = LinearRegression()
#Fit model
lr.fit(X_train, y_train)
#predict
#prediction = lr.predict(X_test)
#actual
actual = y_test
train_score_lr = lr.score(X_train, y_train)
test_score_lr = lr.score(X_test, y_test)
print("\nLinear Regression Model:\n")
print("The train score for lr model is {}".format(train_score_lr))
print("The test score for lr model is {}".format(test_score_lr))
```

```
Linear Regression Model:

The train score for lr model is 0.07906758951709636
The test score for lr model is 0.08573839649638304
```

In [37]:
```
ridgeReg = Ridge(alpha=10)
ridgeReg.fit(X_train,y_train)
#train and test scorefor ridge regression
train_score_ridge = ridgeReg.score(X_train, y_train)
test_score_ridge = ridgeReg.score(X_test, y_test)
print("\nRidge Model:\n")
print("The train score for ridge model is {}".format(train_score_ridge))
print("The test score for ridge model is {}".format(test_score_ridge))
```

```
Ridge Model:

The train score for ridge model is 0.07906120163510788
The test score for ridge model is 0.08541192691344546
```

In [38]:
```
plt.figure(figsize=(10,10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,color='red',label=r'Ridge;$\alpha=
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='green',label='Linear Regression')
plt.xticks(rotation=90)
plt.legend()
plt.show()
```

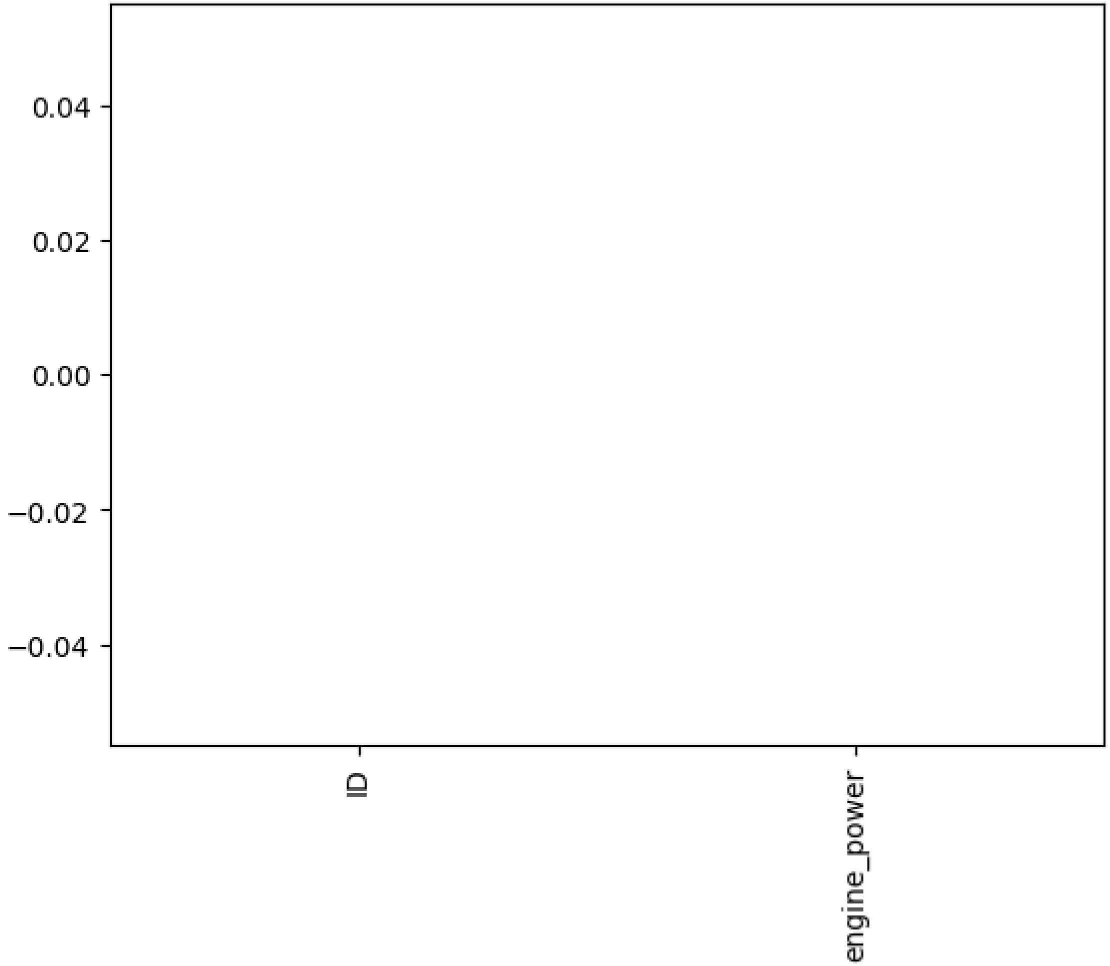

Ridge; $\alpha = 10$
Linear Regression
0.00
−0.02
−0.04
−0.06


In [39]:
```
lassoReg=Lasso(alpha=10)
lassoReg.fit(X_train,y_train)
train_score_lasso=lassoReg.score(X_train,y_train)
test_score_lasso=lassoReg.score(X_test,y_test)
print("\nRidge model\:\n")
print("The train score for lasso model is {}".format(train_score_ridge))
print("The train score for lasso model is {}".format(test_score_ridge))
```

```
Ridge model\:

The train score for lasso model is 0.07906120163510788
The train score for lasso model is 0.08541192691344546
```

In [40]:
```
pd.Series(lassoReg.coef_,features).sort_values(ascending=True).plot(kind="bar")
```

Out[40]: <Axes: >



In [41]:
```
from sklearn.linear_model import LassoCV
lasso_CV=LassoCV(alphas=[0.0001,0.001,0.01,0.1,1,10]).fit(X_train,y_train)
print("The train score for lasso model is{}".format(lasso_CV.score(X_train,y_train)))
print("The test score for lasso model is{}".format(lasso_CV.score(X_test,y_test)))
```

```
The train score for lasso model is0.07906730311134957
The test score for lasso model is0.08575009503364805
```

In [42]:
```
plt.figure(figsize=(10,10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,color='red',label=r'Ridge;$\alpha=
plt.plot(features,lasso_CV.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,color='blue',label=r'lasso,$\alpha
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='green',label='LinearRegression')
plt.xticks(rotation=90)
plt.legend()
plt.title("comparision plot of Ridge,Lasso and LinearRegression model")
plt.show()
```


comparision plot of Ridge,Lasso and LinearRegression model
Ridge;$\alpha = 10$
lasso,$\alpha = grid$
LinearRegression
0.00
−0.02
−0.04
−0.06


In [43]:
```
from sklearn.linear_model import RidgeCV
ridge_CV=RidgeCV(alphas=[0.0001,0.001,0.01,0.1,1,10]).fit(X_train,y_train)
print("The train score for ridge model is{}".format(ridge_CV.score(X_train,y_train)))
print("The test score for ridge model is{}".format(ridge_CV.score(X_test,y_test)))
```

```
The train score for ridge model is0.07906120163510788
The test score for ridge model is0.08541192691344601
```

In [ ]: