In [1]:
```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [17]:
```python
# reading the data set
df=pd.read_csv('car data.csv')
```

In [18]:
```python
df.head()
```

Out[18]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

In [19]:
```python
df.shape
```

Out[19]:
```
(301, 9)
```

In [20]:
```python
print(df['Seller_Type'].unique())
print(df['Transmission'].unique())
print(df['Owner'].unique())
print(df['Fuel_Type'].unique())
```

```
['Dealer' 'Individual']
['Manual' 'Automatic']
[0 1 3]
['Petrol' 'Diesel' 'CNG']
```

In [21]:
```python
df.isnull().sum()
```

Out[21]:
```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Kms_Driven       0
Fuel_Type        0
Seller_Type      0
Transmission     0
Owner            0
dtype: int64
```

In [22]:
```python
df.describe()
```

Out[22]:

	Year	Selling_Price	Present_Price	Kms_Driven	Owner
count	301.000000	301.000000	301.000000	301.000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980	0.043189
std	2.891554	5.082812	8.644115	38886.883882	0.247915
min	2003.000000	0.100000	0.320000	500.000000	0.000000
25%	2012.000000	0.900000	1.200000	15000.000000	0.000000
50%	2014.000000	3.600000	6.400000	32000.000000	0.000000
75%	2016.000000	6.000000	9.900000	48767.000000	0.000000
max	2018.000000	35.000000	92.600000	500000.000000	3.000000

In [23]:
```python
df.columns
```

Out[23]:
```
Index(['Car_Name', 'Year', 'Selling_Price', 'Present_Price', 'Kms_Driven',
       'Fuel_Type', 'Seller_Type', 'Transmission', 'Owner'],
      dtype='object')
```

In [24]:
```python
final_dataset=df[['Year', 'Selling_Price', 'Present_Price', 'Kms_Driven', 'Fuel_Type', 'Seller_Type', 'Transmission', 'Owner']]
```

In [25]:
```python
# adding new column known as current_year.
final_dataset['current_year']=2021
```

In [26]:
```python
final_dataset.head()
```

Out[26]:

	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	current_year
0	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0	2021
1	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0	2021
2	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0	2021
3	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0	2021
4	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0	2021

In [27]:
```python
# subtracting Year from current_year to get no._year
final_dataset['no._year']=final_dataset['current_year']-final_dataset['Year']
```

In [28]:
```python
final_dataset.head()
```

Out[28]:

	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	current_year	no._year
0	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0	2021	7
1	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0	2021	8
2	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0	2021	4
3	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0	2021	10
4	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0	2021	7

In [29]:
```python
# dropping year & current_year as we have no._year
final_dataset.drop(['Year'],axis=1,inplace=True)
final_dataset.drop(['current_year'],axis=1,inplace=True)
```

In [30]:
```python
final_dataset.head()
```

Out[30]:

	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	no._year
0	3.35	5.59	27000	Petrol	Dealer	Manual	0	7
1	4.75	9.54	43000	Diesel	Dealer	Manual	0	8
2	7.25	9.85	6900	Petrol	Dealer	Manual	0	4
3	2.85	4.15	5200	Petrol	Dealer	Manual	0	10
4	4.60	6.87	42450	Diesel	Dealer	Manual	0	7

In [31]:
```python
final_dataset=pd.get_dummies(final_dataset,drop_first=True)
final_dataset.head()
```

Out[31]:

	Selling_Price	Present_Price	Kms_Driven	Owner	no._year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
0	3.35	5.59	27000	0	7	0	1	0	1
1	4.75	9.54	43000	0	8	1	0	0	1
2	7.25	9.85	6900	0	4	0	1	0	1
3	2.85	4.15	5200	0	10	0	1	0	1
4	4.60	6.87	42450	0	7	1	0	0	1

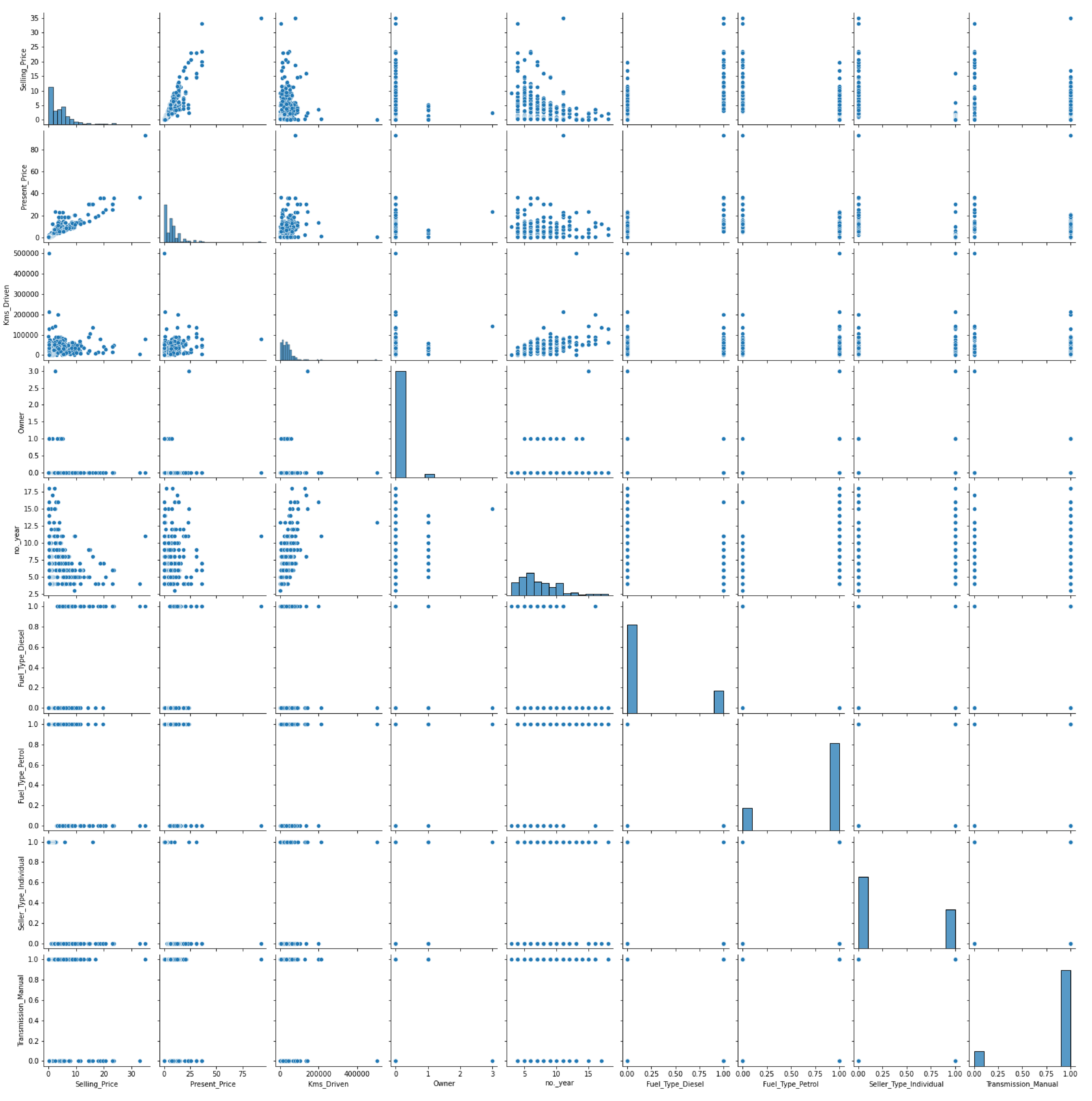
In [32]:
```python
final_dataset.corr()
```

Out[32]:

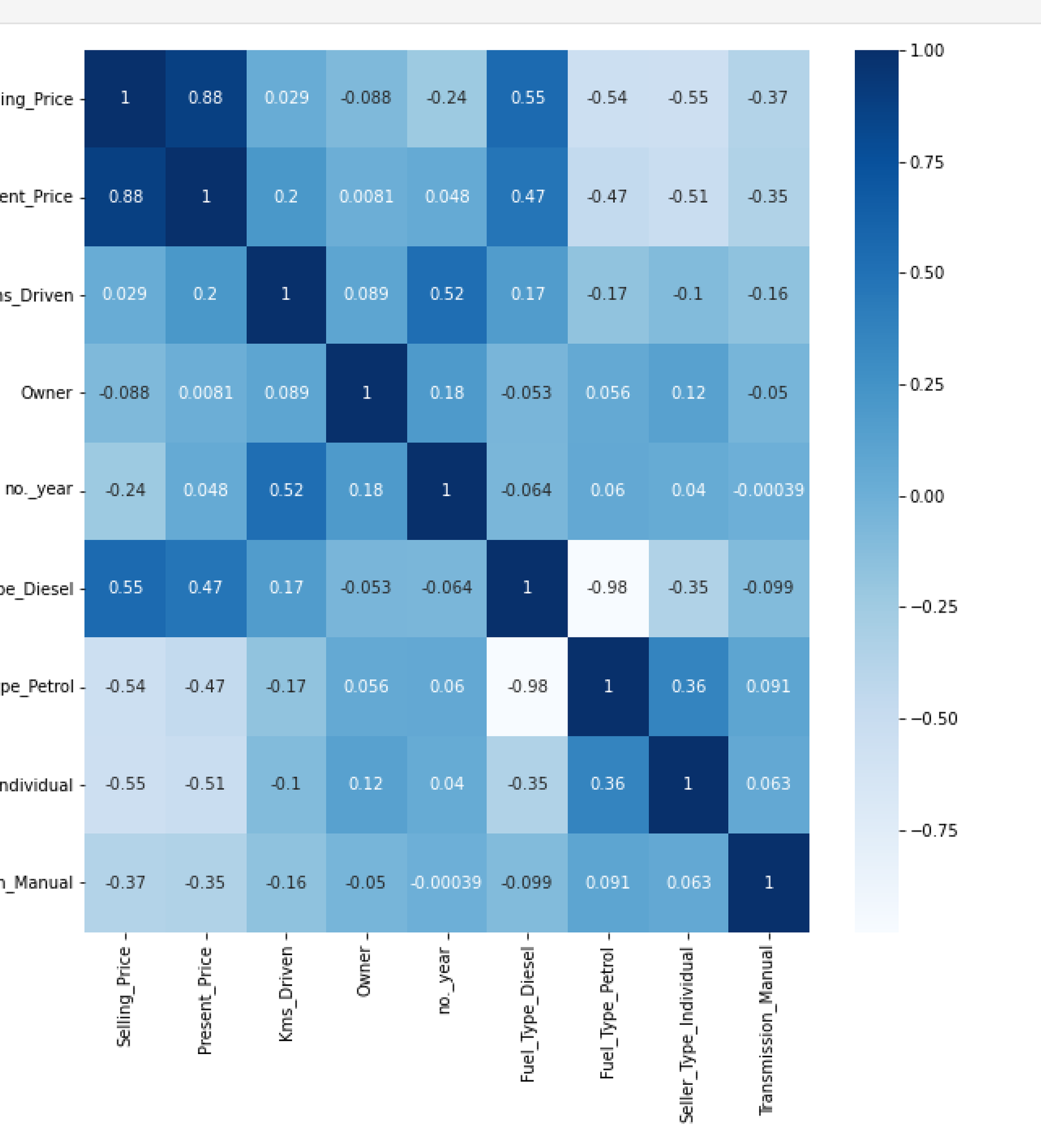
	Selling_Price	Present_Price	Kms_Driven	Owner	no._year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
Selling_Price	1.000000	0.878983	0.029187	-0.088344	-0.236141	0.552339	-0.540571	-0.550724	-0.367128
Present_Price	0.878983	1.000000	0.203647	0.008057	0.047584	0.473306	-0.465244	-0.512030	-0.348715
Kms_Driven	0.029187	0.203647	1.000000	0.089216	0.524342	0.172515	-0.172874	-0.101419	-0.162510
Owner	-0.088344	0.008057	0.089216	1.000000	0.182104	-0.053469	0.055687	0.124269	-0.050316
no._year	-0.236141	0.047584	0.524342	0.182104	1.000000	-0.064315	0.059959	0.039896	-0.000394
Fuel_Type_Diesel	0.552339	0.473306	0.172515	-0.053469	-0.064315	1.000000	-0.979648	-0.350467	-0.098643
Fuel_Type_Petrol	-0.540571	-0.465244	-0.172874	0.055687	0.059959	-0.979648	1.000000	0.358321	0.091013
Seller_Type_Individual	-0.550724	-0.512030	-0.101419	0.124269	0.039896	-0.350467	0.358321	1.000000	0.063240
Transmission_Manual	-0.367128	-0.348715	-0.162510	-0.050316	-0.000394	-0.098643	0.091013	0.063240	1.000000

In [33]:
```python
sns.pairplot(final_dataset)
```

Out[33]:
```
<seaborn.axisgrid.PairGrid at 0x7f9935b4abb0>
```



In [34]:
```python
cormat=final_dataset.corr()
top_corr_features=cormat.index
plt.figure(figsize=(10,10))
g=sns.heatmap(final_dataset[top_corr_features].corr(),annot=True,cmap="Blues")
```



In [35]:
```python
x=final_dataset.iloc[:,1:]
y=final_dataset.iloc[:,0]
```

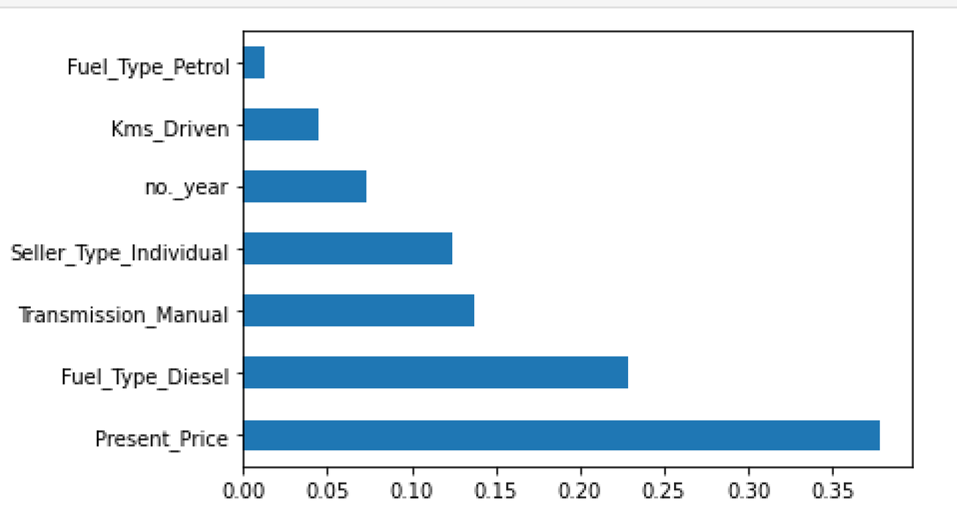
In [36]:
```python
from sklearn.ensemble import ExtraTreesRegressor
model=ExtraTreesRegressor()
model.fit(x,y)
```

Out[36]:
```
ExtraTreesRegressor()
```

In [37]:
```python
print(model.feature_importances_)
```

```
[0.37842217 0.04415011 0.0012481  0.07362488 0.22859809 0.01248109
 0.12465392 0.13682164]
```

In [38]:
```python
feature_importances=pd.Series(model.feature_importances_,index=x.columns)
feature_importances.nlargest(7).plot(kind='barh')
plt.show()
```



In [43]:
```python
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

Extra Trees Regressor

In [45]:
```python
from sklearn.ensemble import ExtraTreesRegressor
model=ExtraTreesRegressor()
model.fit(x,y)
```

Out[45]:
```
ExtraTreesRegressor()
```

In [46]:
```python
model.score(x_train, y_train)
```

Out[46]:
```
1.0
```

In [47]:
```python
predm=model.predict(x_test)
```

In [48]:
```python
model.score(x_test,y_test)
```

Out[48]:
```
1.0
```

In [49]:
```python
from sklearn import metrics
```

In [50]:
```python
metrics.r2_score(y_test,predm)
```

Out[50]:
```
1.0
```

Random Forest Regressor

In [51]:
```python
from sklearn.ensemble import RandomForestRegressor
rf_random=RandomForestRegressor()
```

In [52]:
```python
rf_random.fit(x,y)
```

Out[52]:
```
RandomForestRegressor()
```

In [53]:
```python
rf_random.score(x_train, y_train)
```

Out[53]:
```
0.9889117569050078
```

In [54]:
```python
pred1=rf_random.predict(x_test)
```

In [55]:
```python
rf_random.score(x_test,y_test)
```

Out[55]:
```
0.9952829860566341
```

In [56]:
```python
metrics.r2_score(y_test,pred1)
```

Out[56]:
```
0.9952829860566341
```

We have used both ExtraTreesRegressor and RandomForestRegressor Algorithms both are giving good accuracy on training and testing data. So, Its totally fine to use any of the Algorithm Extra trees Regressor Algorithm is the best one.

In [ ]: