In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
#to ignore warnings
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
car_store = pd.read_csv('C:/Users/dabir/Downloads/EDA FILE/data_set/used_cars_data.csv')
```

In [3]:

```
car_store.head()
```

Out[3]:

	S.No.	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Ty
0	0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	F
1	1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	F
2	2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	F
3	3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	F
4	4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Secc

will display the top 5 observations of the dataset

In [4]:

```
car_store.tail()
```

Out[4]:

	S.No.	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owne
7248	7248	Volkswagen Vento Diesel Trendline	Hyderabad	2011	89411	Diesel	Manual	
7249	7249	Volkswagen Polo GT TSI	Mumbai	2015	59000	Petrol	Automatic	
7250	7250	Nissan Micra Diesel XV	Kolkata	2012	28000	Diesel	Manual	
7251	7251	Volkswagen Polo GT TSI	Pune	2013	52262	Petrol	Automatic	
7252	7252	Mercedes-Benz E-Class 2009-2013 E 220 CDI Avan...	Kochi	2014	72443	Diesel	Automatic	

the last 5 observations of the dataset

In [5]:

```
car_store.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7253 entries, 0 to 7252
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   S.No.              7253 non-null   int64  
 1   Name               7253 non-null   object 
 2   Location           7253 non-null   object 
 3   Year               7253 non-null   int64  
 4   Kilometers_Driven  7253 non-null   int64  
 5   Fuel_Type          7253 non-null   object 
 6   Transmission       7253 non-null   object 
 7   Owner_Type         7253 non-null   object 
 8   Mileage            7251 non-null   object 
 9   Engine             7207 non-null   object 
 10  Power              7207 non-null   object 
 11  Seats              7200 non-null   float64
 12  New_Price          1006 non-null   object 
 13  Price              6019 non-null   float64
dtypes: float64(2), int64(3), object(9)
memory usage: 793.4+ KB
```

In [6]:

```
car_store.nunique()
```

Out[6]:

```
S.No.                7253
Name                 2041
Location               11
Year                   23
Kilometers_Driven    3660
Fuel_Type               5
Transmission            2
Owner_Type              4
Mileage               450
Engine                150
Power                 386
Seats                   9
New_Price             625
Price                1373
dtype: int64
```

In [7]:

```
car_store.isnull().sum()
```

Out[7]:

```
S.No.                   0
Name                    0
Location                0
Year                    0
Kilometers_Driven       0
Fuel_Type               0
Transmission            0
Owner_Type              0
Mileage                 2
Engine                 46
Power                  46
Seats                  53
New_Price            6247
Price                1234
dtype: int64
```

In [8]:

```
(car_store.isnull().sum()/(len(car_store)))*100
```

Out[8]:

```
S.No.                 0.000000
Name                  0.000000
Location              0.000000
Year                  0.000000
Kilometers_Driven     0.000000
Fuel_Type             0.000000
Transmission          0.000000
Owner_Type            0.000000
Mileage               0.027575
Engine                0.634220
Power                 0.634220
Seats                 0.730732
New_Price            86.129877
Price                17.013650
dtype: float64
```

The percentage of missing values for the columns New_Price and Price is ~86% and ~17%, respectively.

In [9]:

```
car_store = car_store.drop(['S.No.'], axis = 1)
car_store.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7253 entries, 0 to 7252
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Name               7253 non-null   object 
 1   Location           7253 non-null   object 
 2   Year               7253 non-null   int64  
 3   Kilometers_Driven  7253 non-null   int64  
 4   Fuel_Type          7253 non-null   object 
 5   Transmission       7253 non-null   object 
 6   Owner_Type         7253 non-null   object 
 7   Mileage            7251 non-null   object 
 8   Engine             7207 non-null   object 
 9   Power              7207 non-null   object 
 10  Seats              7200 non-null   float64
 11  New_Price          1006 non-null   object 
 12  Price              6019 non-null   float64
dtypes: float64(2), int64(2), object(9)
memory usage: 736.8+ KB
```

In [10]:

```
from datetime import date
date.today().year
car_store['Car_Age']=date.today().year-car_store['Year']
car_store.head(10)
```

Out[10]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	M
0	Maruti Wagon R LXI CNG	Mumbai	2010	72000	CNG	Manual	First	
1	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	
2	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	
3	Maruti Ertiga VDI	Chennai	2012	87000	Diesel	Manual	First	
4	Audi A4 New 2.0 TDI Multitronic	Coimbatore	2013	40670	Diesel	Automatic	Second	
5	Hyundai EON LPG Era Plus Option	Hyderabad	2012	75000	LPG	Manual	First	
6	Nissan Micra Diesel XV	Jaipur	2013	86999	Diesel	Manual	First	
7	Toyota Innova Crysta 2.8 GX AT 8S	Mumbai	2016	36000	Diesel	Automatic	First	
8	Volkswagen Vento Diesel Comfortline	Pune	2013	64430	Diesel	Manual	First	
9	Tata Indica Vista Quadrajet LS	Chennai	2012	65932	Diesel	Manual	Second	

In [11]:

```
car_store['Brand'] = car_store.Name.str.split().str.get(0)
car_store['Model'] = car_store.Name.str.split().str.get(1) + car_store.Name.str.split().s
car_store[['Name','Brand','Model']]
```

Out[11]:

	Name	Brand	Model
0	Maruti Wagon R LXI CNG	Maruti	WagonR
1	Hyundai Creta 1.6 CRDi SX Option	Hyundai	Creta1.6
2	Honda Jazz V	Honda	JazzV
3	Maruti Ertiga VDI	Maruti	ErtigaVDI
4	Audi A4 New 2.0 TDI Multitronic	Audi	A4New
...	...	...	...
7248	Volkswagen Vento Diesel Trendline	Volkswagen	VentoDiesel
7249	Volkswagen Polo GT TSI	Volkswagen	PoloGT
7250	Nissan Micra Diesel XV	Nissan	MicraDiesel
7251	Volkswagen Polo GT TSI	Volkswagen	PoloGT
7252	Mercedes-Benz E-Class 2009-2013 E 220 CDI Avan...	Mercedes-Benz	E-Class2009-2013

7253 rows × 3 columns

In [12]:

```
print(car_store.Brand.unique())
print(car_store.Brand.nunique())
```

```
['Maruti' 'Hyundai' 'Honda' 'Audi' 'Nissan' 'Toyota' 'Volkswagen' 'Tata'
 'Land' 'Mitsubishi' 'Renault' 'Mercedes-Benz' 'BMW' 'Mahindra' 'Ford'
 'Porsche' 'Datsun' 'Jaguar' 'Volvo' 'Chevrolet' 'Skoda' 'Mini' 'Fiat'
 'Jeep' 'Smart' 'Ambassador' 'Isuzu' 'ISUZU' 'Force' 'Bentley'
 'Lamborghini' 'Hindustan' 'OpelCorsa']
33
```

The brand name ‘Isuzu’ ‘ISUZU’ and ‘Mini’ and ‘Land’ looks incorrect.

In [13]:

```
searchfor = ['Isuzu' ,'ISUZU','Mini','Land']
car_store[car_store.Brand.str.contains('|'.join(searchfor))].head(5)
```

Out[13]:

	Name	Location	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type
13	Land Rover Range Rover 2.2L Pure	Delhi	2014	72000	Diesel	Automatic	First
14	Land Rover Freelander 2 TD4 SE	Pune	2012	85000	Diesel	Automatic	Second
176	Mini Countryman Cooper D	Jaipur	2017	8525	Diesel	Automatic	Second
191	Land Rover Range Rover 2.2L Dynamic	Coimbatore	2018	36091	Diesel	Automatic	First
228	Mini Cooper Convertible S	Kochi	2017	26327	Petrol	Automatic	First

In [14]:

```
car_store["Brand"].replace({"ISUZU": "Isuzu", "Mini": "Mini Cooper","Land":"Land Rover"},
```

In [16]:

```
car_store.describe().T
```

Out[16]:

	count	mean	std	min	25%	50%	75%
Year	7253.0	2013.365366	3.254421	1996.00	2011.0	2014.00	2016.00
Kilometers_Driven	7253.0	58699.063146	84427.720583	171.00	34000.0	53416.00	73000.00
Seats	7200.0	5.279722	0.811660	0.00	5.0	5.00	5.00
Price	6019.0	9.479468	11.187917	0.44	3.5	5.64	9.95
Car_Age	7253.0	9.634634	3.254421	4.00	7.0	9.00	12.00

- Years range from 1996- 2019 and has a high in a range which shows used cars contain both latest models and old model cars.
- On average of Kilometers-driven in Used cars are ~58k KM. The range shows a huge difference between min and max as max values show 650000 KM shows the evidence of an outlier. This record can be removed.
- Min value of Mileage shows 0 cars won't be sold with 0 mileage. This sounds like a data entry issue.
- It looks like Engine and Power have outliers, and the data is right-skewed.

- The average number of seats in a car is 5. car seat is an important feature in price contribution.
- The max price of a used car is 160k which is quite weird, such a high price for used cars, There may be an outlier or data entry issue.

In [17]:

```
car_store.describe(include='all').T
```

Out[17]:

	count	unique	top	freq	mean	std	min	2
Name	7253	2041	Mahindra XUV500 W8 2WD	55	NaN	NaN	NaN	N
Location	7253	11	Mumbai	949	NaN	NaN	NaN	N
Year	7253.0	NaN	NaN	NaN	2013.365366	3.254421	1996.0	201
Kilometers_Driven	7253.0	NaN	NaN	NaN	58699.063146	84427.720583	171.0	3400
Fuel_Type	7253	5	Diesel	3852	NaN	NaN	NaN	N
Transmission	7253	2	Manual	5204	NaN	NaN	NaN	N
Owner_Type	7253	4	First	5952	NaN	NaN	NaN	N
Mileage	7251	450	17.0 kmpl	207	NaN	NaN	NaN	N
Engine	7207	150	1197 CC	732	NaN	NaN	NaN	N
Power	7207	386	74 bhp	280	NaN	NaN	NaN	N
Seats	7200.0	NaN	NaN	NaN	5.279722	0.81166	0.0	
New_Price	1006	625	63.71 Lakh	6	NaN	NaN	NaN	N
Price	6019.0	NaN	NaN	NaN	9.479468	11.187917	0.44	
Car_Age	7253.0	NaN	NaN	NaN	9.634634	3.254421	4.0	
Brand	7253	32	Maruti	1444	NaN	NaN	NaN	N
Model	7252	726	SwiftDzire	189	NaN	NaN	NaN	N

In [18]:

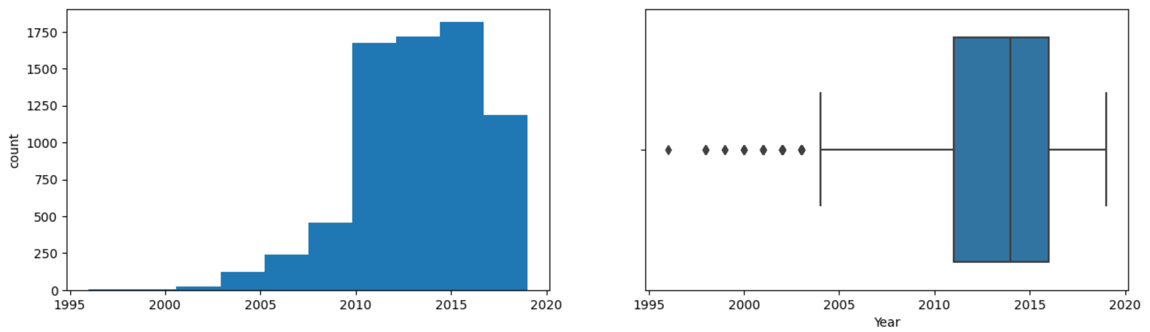
```
cat_cols=car_store.select_dtypes(include=['object']).columns
num_cols = car_store.select_dtypes(include=np.number).columns.tolist()
print("Categorical Variables:")
print(cat_cols)
print("Numerical Variables:")
print(num_cols)
```

```
Categorical Variables:
Index(['Name', 'Location', 'Fuel_Type', 'Transmission', 'Owner_Type',
       'Mileage', 'Engine', 'Power', 'New_Price', 'Brand', 'Model'],
      dtype='object')
Numerical Variables:
['Year', 'Kilometers_Driven', 'Seats', 'Price', 'Car_Age']
```

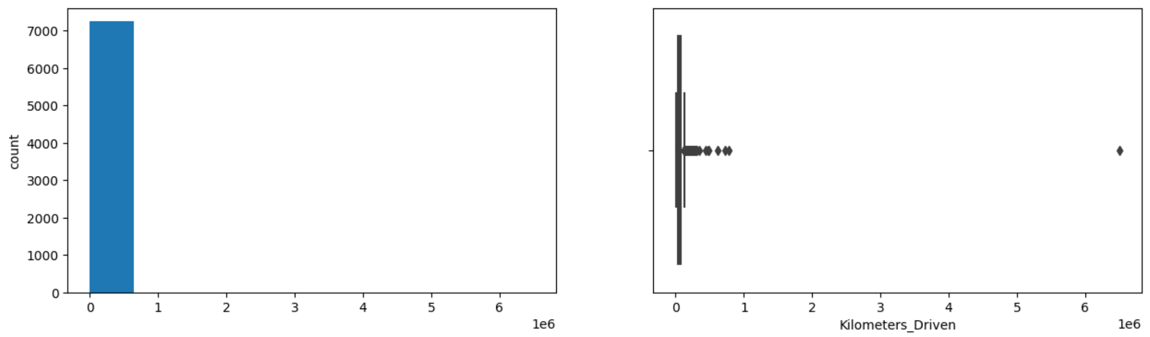
In [20]:

```
for col in num_cols:
    print(col)
    print('Skew :', round(car_store[col].skew(), 2))
    plt.figure(figsize = (15, 4))
    plt.subplot(1, 2, 1)
    car_store[col].hist(grid=False)
    plt.ylabel('count')
    plt.subplot(1, 2, 2)
    sns.boxplot(x=car_store[col])
    plt.show()
```

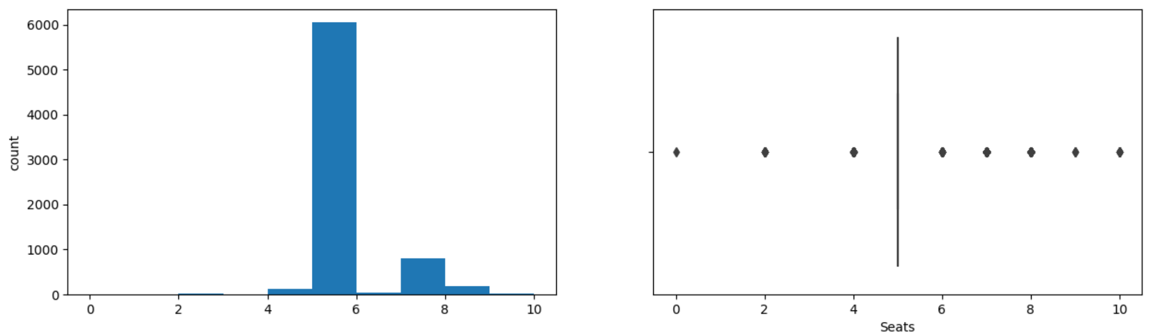
```
Year
Skew : -0.84
```



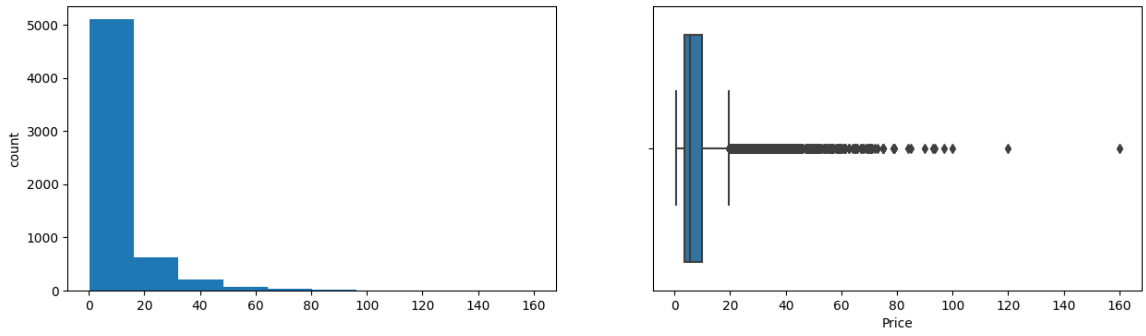

```
Kilometers_Driven
Skew : 61.58
```



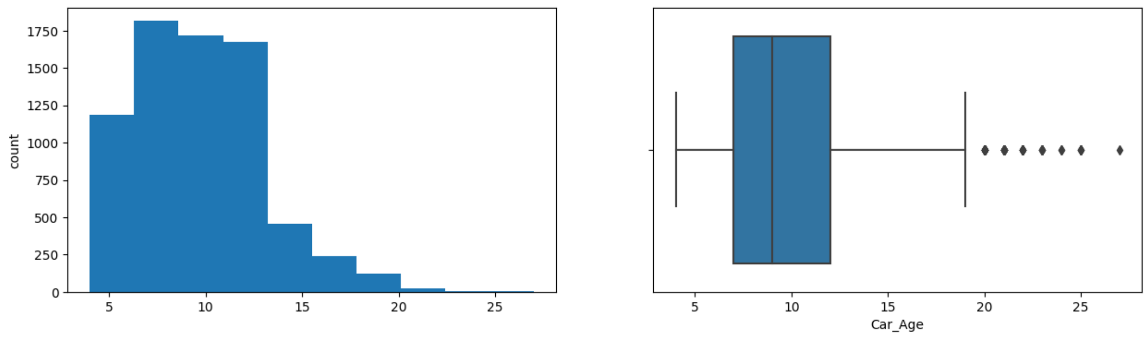

```
Seats
Skew : 1.9
```




```
Price
Skew : 3.34
```



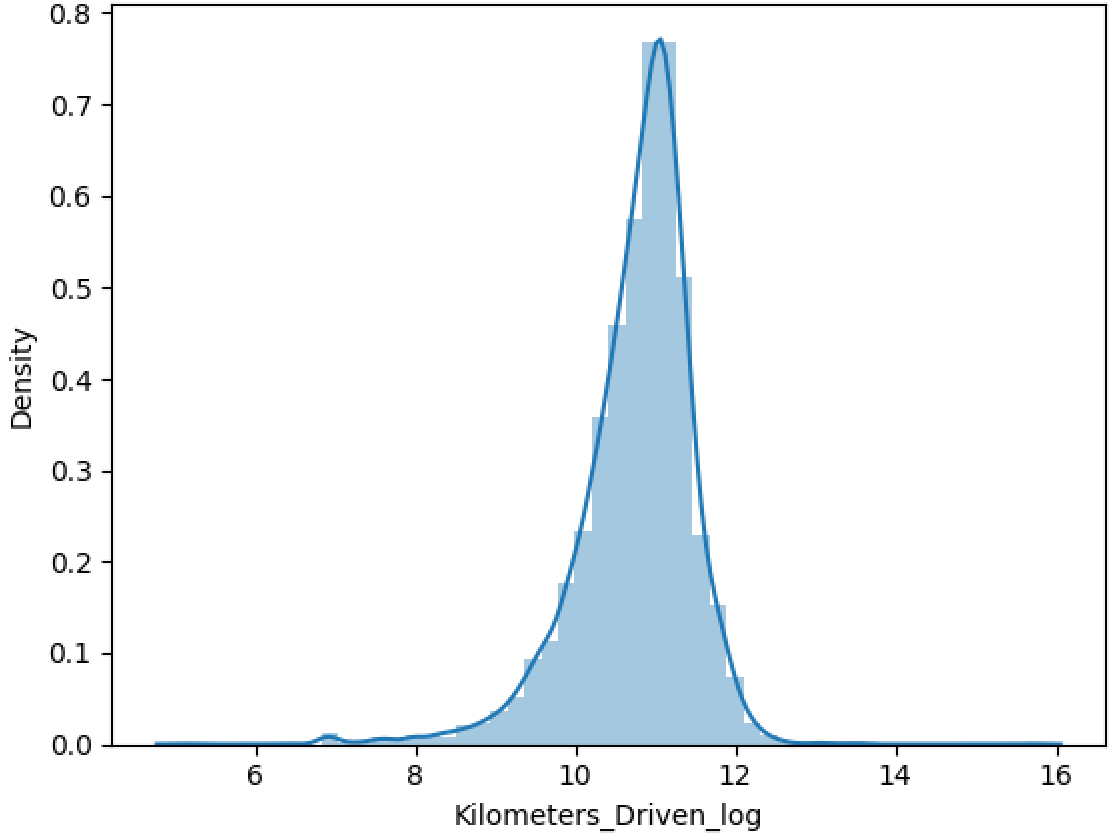

```
Car_Age
Skew : 0.84
```



In [25]:

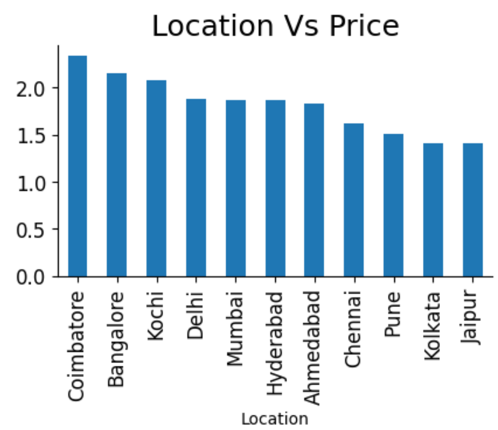
```
# Function for log transformation of the column
def log_transform(car_store,col):
    for colname in col:
        if (car_store[colname] == 1.0).all():
            car_store[colname + '_log'] = np.log(car_store[colname]+1)
        else:
            car_store[colname + '_log'] = np.log(car_store[colname])
    car_store.info()
log_transform(car_store,['Kilometers_Driven','Price'])
#Log transformation of the feature 'Kilometers_Driven'
sns.distplot(car_store["Kilometers_Driven_log"], axlabel="Kilometers_Driven_log");
```

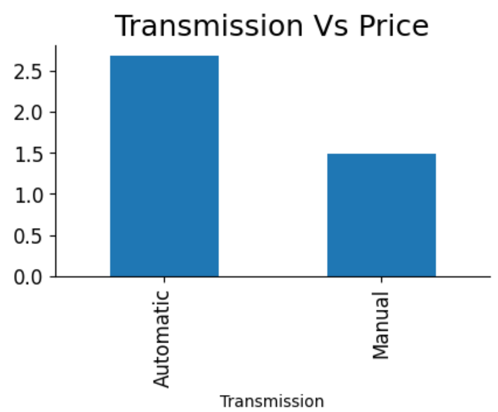
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7253 entries, 0 to 7252
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Name                   7253 non-null   object 
 1   Location               7253 non-null   object 
 2   Year                   7253 non-null   int64  
 3   Kilometers_Driven      7253 non-null   int64  
 4   Fuel_Type              7253 non-null   object 
 5   Transmission           7253 non-null   object 
 6   Owner_Type             7253 non-null   object 
 7   Mileage                7251 non-null   object 
 8   Engine                 7207 non-null   object 
 9   Power                  7207 non-null   object 
 10  Seats                  7200 non-null   float64
 11  New_Price              1006 non-null   object 
 12  Price                  6019 non-null   float64
 13  Car_Age                7253 non-null   int64  
 14  Brand                  7253 non-null   object 
 15  Model                  7252 non-null   object 
 16  Kilometers_Driven_log  7253 non-null   float64
 17  Price_log              6019 non-null   float64
dtypes: float64(4), int64(3), object(11)
memory usage: 1020.1+ KB
```

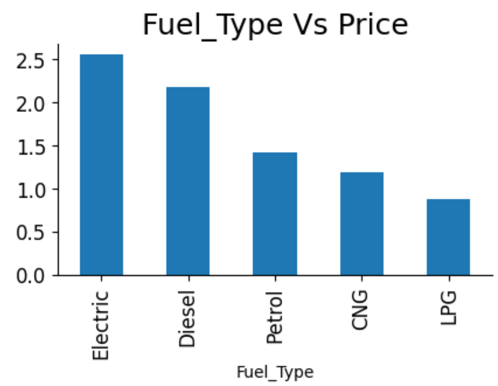

0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Density
6
8
10
12
14
16
Kilometers_Driven_log

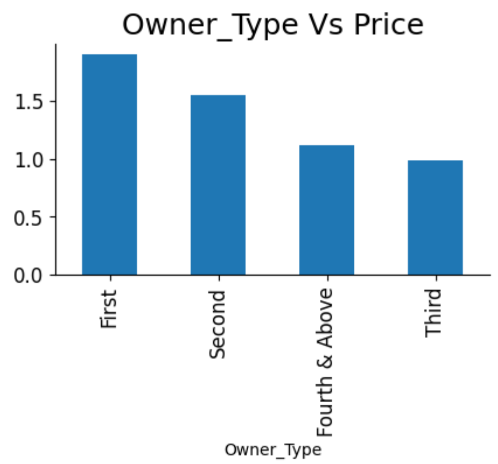
In [30]:

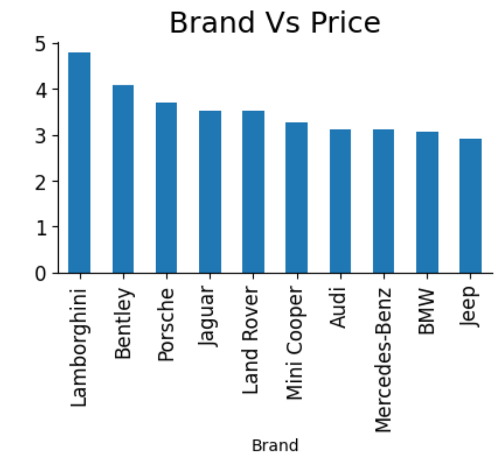
```
fig, axarr = plt.subplots(4, 2, figsize=(12, 18))
car_store.groupby('Location')['Price_log'].mean().sort_values(ascending=False).plot.bar(a
axarr[0][0].set_title("Location Vs Price", fontsize=18)
car_store.groupby('Transmission')['Price_log'].mean().sort_values(ascending=False).plot.b
axarr[0][1].set_title("Transmission Vs Price", fontsize=18)
car_store.groupby('Fuel_Type')['Price_log'].mean().sort_values(ascending=False).plot.bar(
axarr[1][0].set_title("Fuel_Type Vs Price", fontsize=18)
car_store.groupby('Owner_Type')['Price_log'].mean().sort_values(ascending=False).plot.bar
axarr[1][1].set_title("Owner_Type Vs Price", fontsize=18)
car_store.groupby('Brand')['Price_log'].mean().sort_values(ascending=False).head(10).plot
axarr[2][0].set_title("Brand Vs Price", fontsize=18)
car_store.groupby('Model')['Price_log'].mean().sort_values(ascending=False).head(10).plot
axarr[2][1].set_title("Model Vs Price", fontsize=18)
car_store.groupby('Seats')['Price_log'].mean().sort_values(ascending=False).plot.bar(ax=a
axarr[3][0].set_title("Seats Vs Price", fontsize=18)
car_store.groupby('Car_Age')['Price_log'].mean().sort_values(ascending=False).plot.bar(ax
axarr[3][1].set_title("Car_Age Vs Price", fontsize=18)
plt.subplots_adjust(hspace=1.0)
plt.subplots_adjust(wspace=.5)
sns.despine()
```

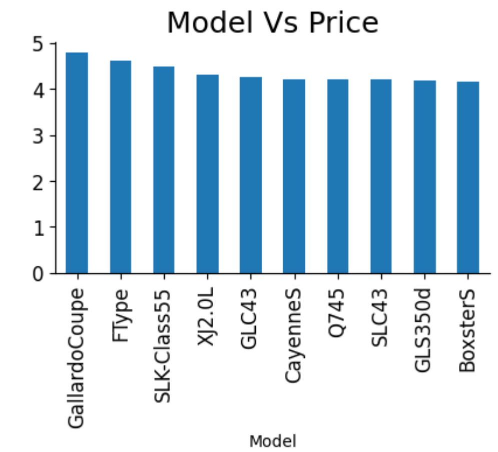


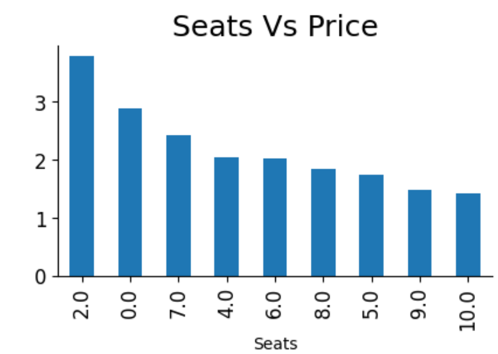


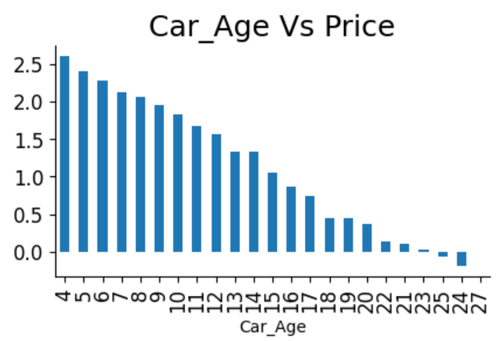












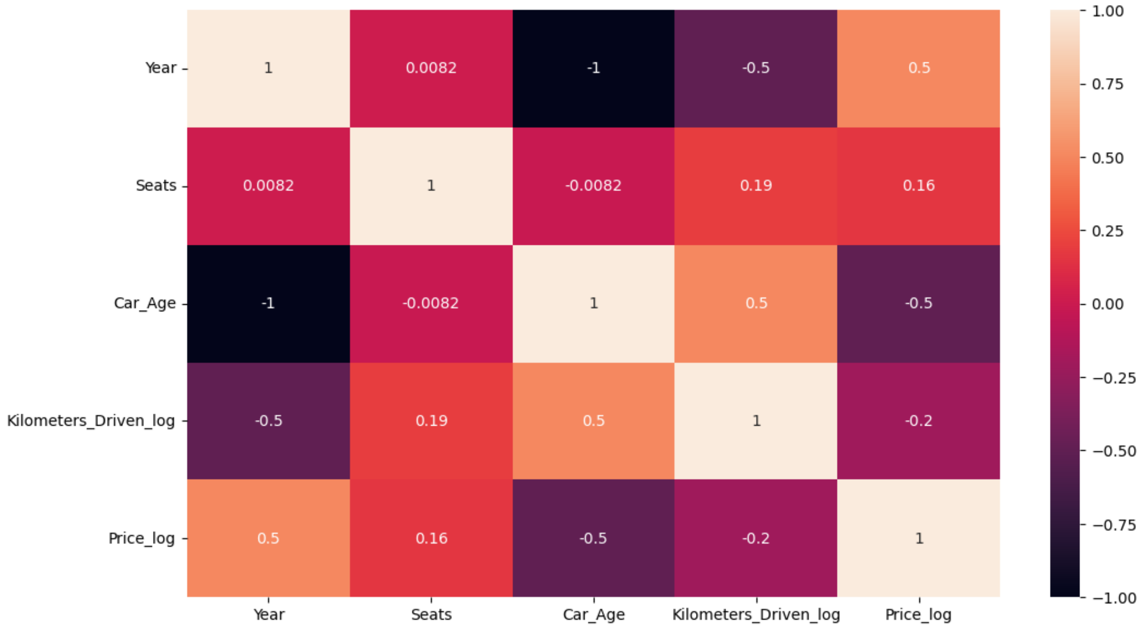
Observations

- The price of cars is high in Coimbatore and less price in Kolkata and Jaipur
- Automatic cars have more price than manual cars.
- Diesel and Electric cars have almost the same price, which is maximum, and LPG cars have the lowest price

- First-owner cars are higher in price, followed by a second .
- The third owner's price is lesser than the Fourth and above .
- Lamborghini brand is the highest in price
- Gallardocoupe Model is the highest in price
- 2 Seater has the highest price followed by 7 Seater
- The latest model cars are high in price

In [31]:

```
plt.figure(figsize=(12, 7))
sns.heatmap(car_store.drop(['Kilometers_Driven','Price'],axis=1).corr(), annot = True, vm
plt.show()
```



- The engine has a strong positive correlation to Power 0.86
- Price has a positive correlation to Engine 0.69 as well Power 0.77
- Mileage has correlated to Engine, Power, and Price negatively
- Price is moderately positive in correlation to year.
- Kilometer driven has a negative correlation to year not much impact on the price
- Car age has a negative correlation with Price *car Age is positively correlated to Kilometers-Driven as the Age of the car increases; then the kilometer will also increase of car has a negative correlation with Mileage this makes sense.

Conclusion

- Most of the customers prefer 2 Seat cars hence the price of the 2-seat cars is higher than other cars.
- The price of the car decreases as the Age of the car increases.
- Customers prefer to purchase the First owner rather than the Second or Third.
- Due to increased Fuel price, the customer prefers to purchase an Electric vehicle.
- Automatic Transmission is easier than Manual.

In [ ]:

```

```