In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error,r2_score

import warnings
warnings.filterwarnings('ignore')
```

In [9]:

```
df = pd.read_csv('car data.csv')
df
```

Out[9]:

	Car_Name	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Selling_type	Trai
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	
...	...	...	...	...	...	...	...	
296	city	2016	9.50	11.60	33988	Diesel	Dealer	
297	brio	2015	4.00	5.90	60000	Petrol	Dealer	
298	city	2009	3.35	11.00	87934	Petrol	Dealer	
299	city	2017	11.50	12.50	9000	Diesel	Dealer	
300	brio	2016	5.30	5.90	5464	Petrol	Dealer	

301 rows × 9 columns

In [3]:

```
df.isnull().sum()
```

Out[3]:

```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Driven_kms       0
Fuel_Type        0
Selling_type     0
Transmission     0
Owner            0
dtype: int64
```

In [4]:

```
# Check Duplication
df.duplicated().sum()
```

Out[4]:

```
2
```

In [5]:

```
#Check datatype
df.dtypes
```

Out[5]:

```
Car_Name          object
Year               int64
Selling_Price    float64
Present_Price    float64
Driven_kms         int64
Fuel_Type         object
Selling_type      object
Transmission      object
Owner              int64
dtype: object
```

In [6]:

```
# Check the number of unique values of each column
df.nunique()
```

Out[6]:

```
Car_Name          98
Year              16
Selling_Price    156
Present_Price    148
Driven_kms       206
Fuel_Type          3
Selling_type       2
Transmission       2
Owner              3
dtype: int64
```

In [7]:

```
#Check statistics of data set
df.describe()
```

Out[7]:

	Year	Selling_Price	Present_Price	Driven_kms	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	8.642584	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 [10]:

```
#First 7 rows of dataset
print("bottom seven rows of dataset are: ")
df.tail(7)
```

```
bottom seven rows of dataset are:
```

Out[10]:

	Car_Name	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Selling_type	Trai
294	amaze	2014	3.75	6.80	33019	Petrol	Dealer	
295	city	2015	8.55	13.09	60076	Diesel	Dealer	
296	city	2016	9.50	11.60	33988	Diesel	Dealer	
297	brio	2015	4.00	5.90	60000	Petrol	Dealer	
298	city	2009	3.35	11.00	87934	Petrol	Dealer	
299	city	2017	11.50	12.50	9000	Diesel	Dealer	
300	brio	2016	5.30	5.90	5464	Petrol	Dealer	

In [12]:

```
df.shape
```

Out[12]:

```
(301, 9)
```

In [13]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Car_Name       301 non-null    object 
 1   Year           301 non-null    int64  
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Driven_kms     301 non-null    int64  
 5   Fuel_Type      301 non-null    object 
 6   Selling_type   301 non-null    object 
 7   Transmission   301 non-null    object 
 8   Owner          301 non-null    int64  
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

In [14]:

```
print(df.Fuel_Type.value_counts())
print(df.Selling_type.value_counts())
print(df.Transmission.value_counts())
```

```
Petrol    239
Diesel     60
CNG         2
Name: Fuel_Type, dtype: int64
Dealer        195
Individual    106
Name: Selling_type, dtype: int64
Manual       261
Automatic     40
Name: Transmission, dtype: int64
```

In [15]:

```
# encoding "Fuel_Type" Column
df.replace({'Fuel_Type':{'Petrol':0,'Diesel':1,'CNG':2}},inplace=True)

# encoding "Seller_Type" Column
df.replace({'Selling_type':{'Dealer':0,'Individual':1}},inplace=True)

# encoding "Transmission" Column
df.replace({'Transmission':{'Manual':0,'Automatic':1}},inplace=True)
```

In [16]:

```
df.head()
```

Out[16]:

	Car_Name	Year	Selling_Price	Present_Price	Driven_kms	Fuel_Type	Selling_type	Trans
0	ritz	2014	3.35	5.59	27000	0	0	
1	sx4	2013	4.75	9.54	43000	1	0	
2	ciaz	2017	7.25	9.85	6900	0	0	
3	wagon r	2011	2.85	4.15	5200	0	0	
4	swift	2014	4.60	6.87	42450	1	0	

In [18]:

```
X = df.drop(['Car_Name','Selling_Price'],axis=1)
Y = df['Selling_Price']
```

In [19]:

```
print(X)
```

```
     Year  Present_Price  Driven_kms  Fuel_Type  Selling_type  Transmissio
n  \
0    2014           5.59       27000          0             0
0
1    2013           9.54       43000          1             0
0
2    2017           9.85        6900          0             0
0
3    2011           4.15        5200          0             0
0
4    2014           6.87       42450          1             0
0
..    ...            ...         ...        ...           ...
...
296  2016          11.60       33988          1             0
0
297  2015           5.90       60000          0             0
0
298  2009          11.00       87934          0             0
0
299  2017          12.50        9000          1             0
0
300  2016           5.90        5464          0             0
0

     Owner
0        0
1        0
2        0
3        0
4        0
..     ...
296      0
297      0
298      0
299      0
300      0

[301 rows x 7 columns]
```

In [20]:

```
print(Y)
```

```
0       3.35
1       4.75
2       7.25
3       2.85
4       4.60
       ...
296     9.50
297     4.00
298     3.35
299    11.50
300     5.30
Name: Selling_Price, Length: 301, dtype: float64
```

In [21]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.1, random_state=2
```

In [22]:

```
lin_reg_model = LinearRegression()
```

In [23]:

```
lin_reg_model.fit(X_train,Y_train)
```

Out[23]:

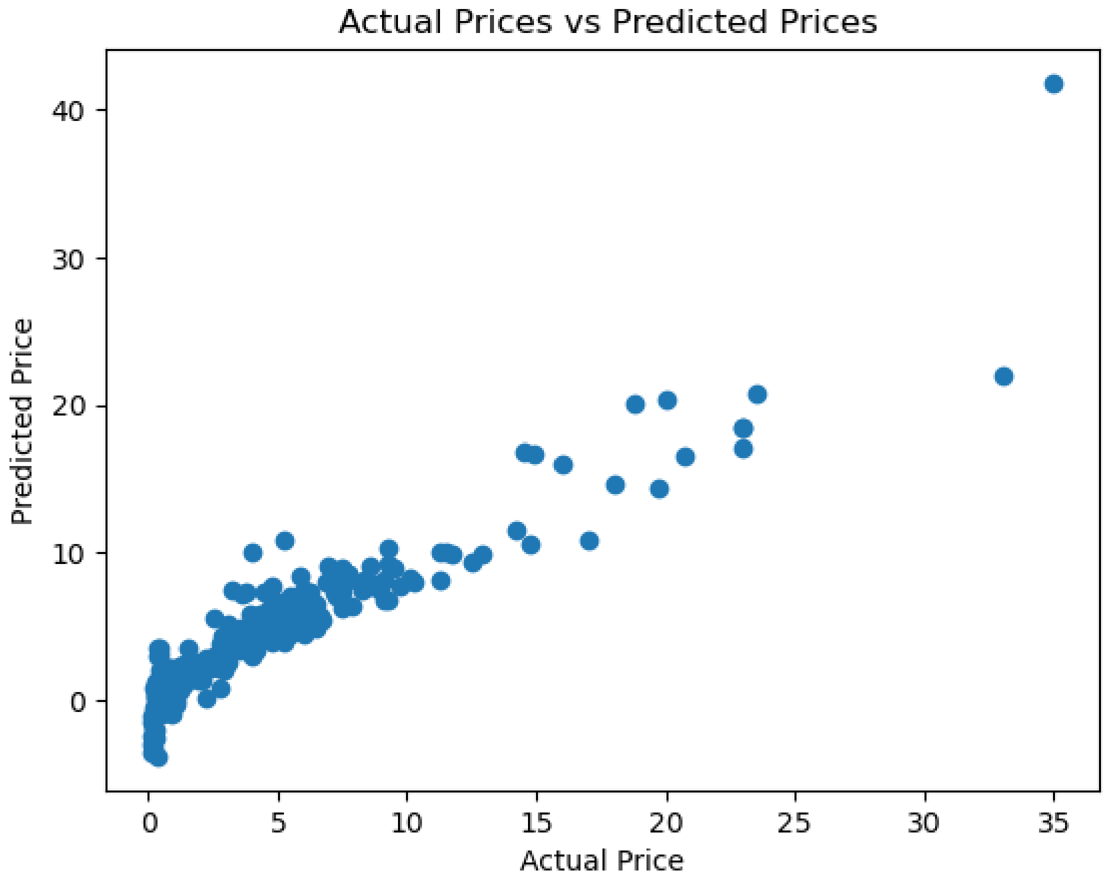
```
LinearRegression()
```

In [24]:

```
training_data_prediction = lin_reg_model.predict(X_train)
```

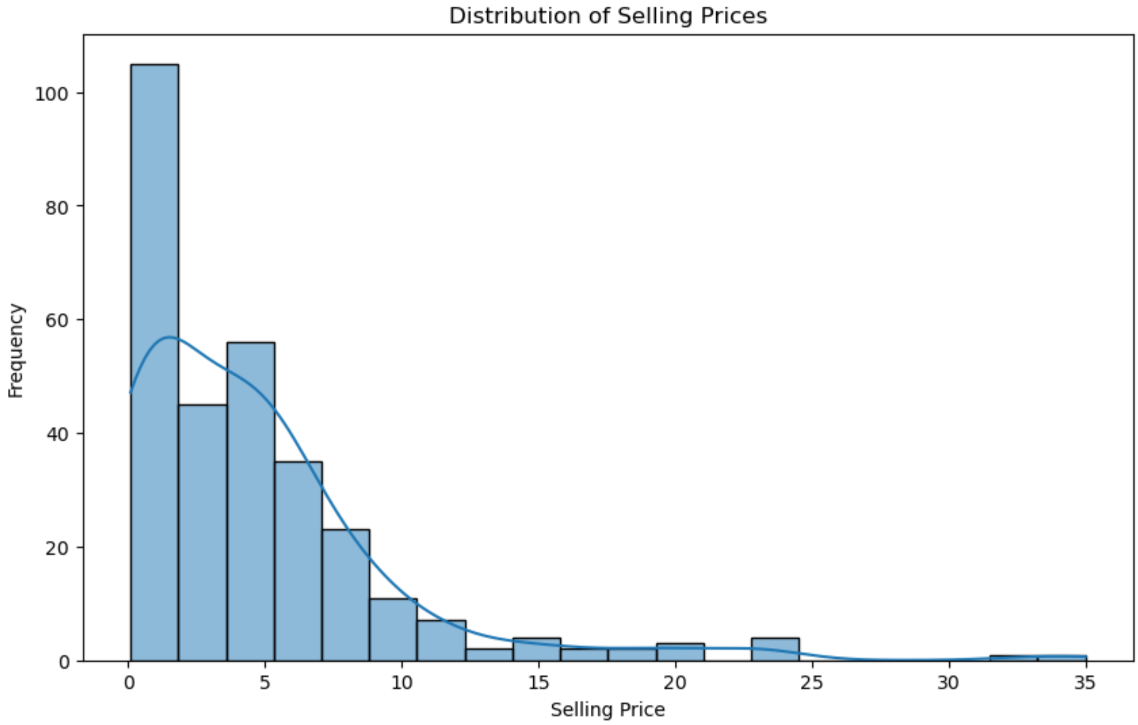
In [26]:

```
plt.scatter(Y_train, training_data_prediction)
plt.xlabel("Actual Price")
plt.ylabel("Predicted Price")
plt.title(" Actual Prices vs Predicted Prices")
plt.show()
```



In [28]:

```
plt.figure(figsize=(10, 6))
sns.histplot(df['Selling_Price'], bins=20, kde=True)
plt.xlabel('Selling Price')
plt.ylabel('Frequency')
plt.title('Distribution of Selling Prices')
plt.show()
```



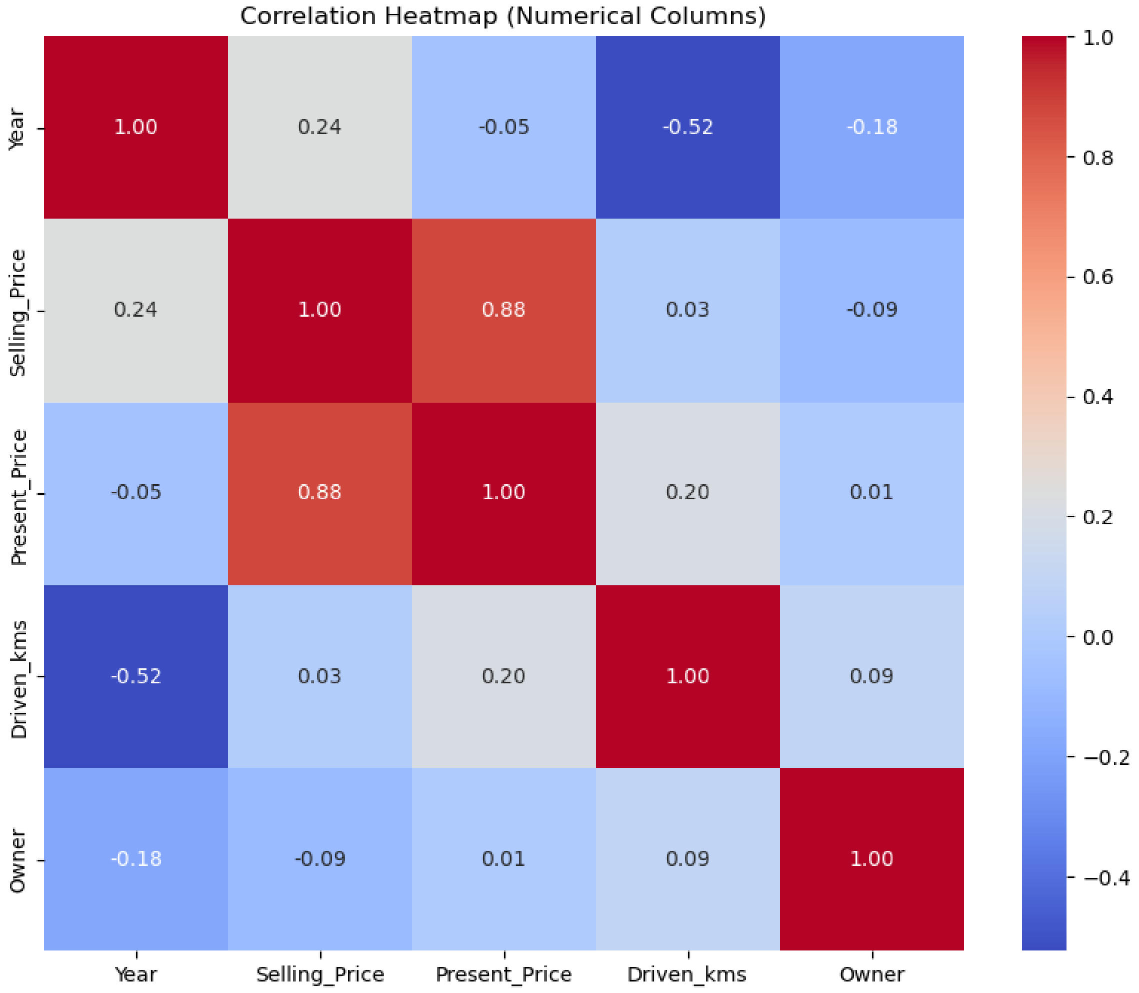
In [31]:

```
# Select numerical columns
numerical_columns = ['Year', 'Selling_Price', 'Present_Price', 'Driven_kms', 'Owner']

# Create a DataFrame containing only the numerical columns
numerical_df = df[numerical_columns]

# Calculate the correlation matrix for numerical columns
correlation_matrix = numerical_df.corr()

# Create a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap (Numerical Columns)')
plt.show()
```



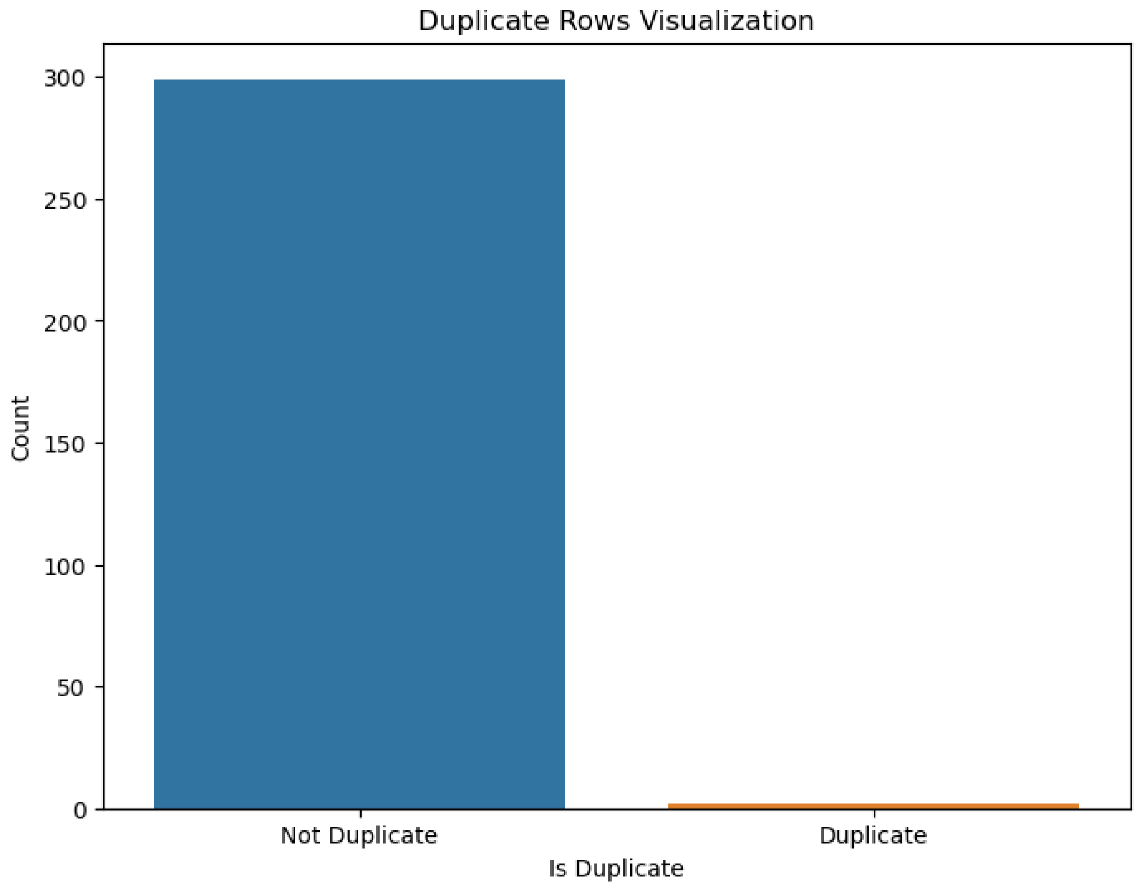
In [34]:

```
# Check for duplicates and create a new column "Is_Duplicate"
df['Is_Duplicate'] = df.duplicated()

# Import necessary libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Create a count plot to visualize duplicates
plt.figure(figsize=(8, 6))
sns.countplot(x='Is_Duplicate', data=df)
plt.xlabel('Is Duplicate')
plt.ylabel('Count')
plt.title('Duplicate Rows Visualization')
plt.xticks([0, 1], ['Not Duplicate', 'Duplicate'])  # Customize x-axis labels
plt.show()

# Drop the "Is_Duplicate" column if not needed
df.drop(columns=['Is_Duplicate'], inplace=True)
```


Duplicate Rows Visualization
Count
Not Duplicate
Duplicate
Is Duplicate