In [2]:

```
import pandas as pd

# Load the dataset
data = pd.read_csv('Sport car price.csv')  # Make sure the filename matche

# Convert the 'Price (in USD)' column to float after removing commas
data['Price (in USD)'] = data['Price (in USD)'].replace('[\$,]', '', regex

# Calculate the average price for each car make
average_prices = data.groupby('Car Make')['Price (in USD)'].mean().reset_i

# Define the price categories
price_categories = {
    'Budget-Friendly Brands': (0, 50000),
    'Mid-Range Brands': (50001, 100000),
    'Premium Brands': (100001, 300000),
    'Ultra-Premium / Luxury Brands': (300001, 1000000),
    'Hypercar Brands': (1000001, float('inf'))
}

# Categorize each brand
def categorize_brand(price):
    for category, (low, high) in price_categories.items():
        if low <= price <= high:
            return category
    return 'Uncategorized'

average_prices['Category'] = average_prices['Price (in USD)'].apply(catego

# Sort and display the results
average_prices.sort_values(by='Price (in USD)', ascending=False)
```

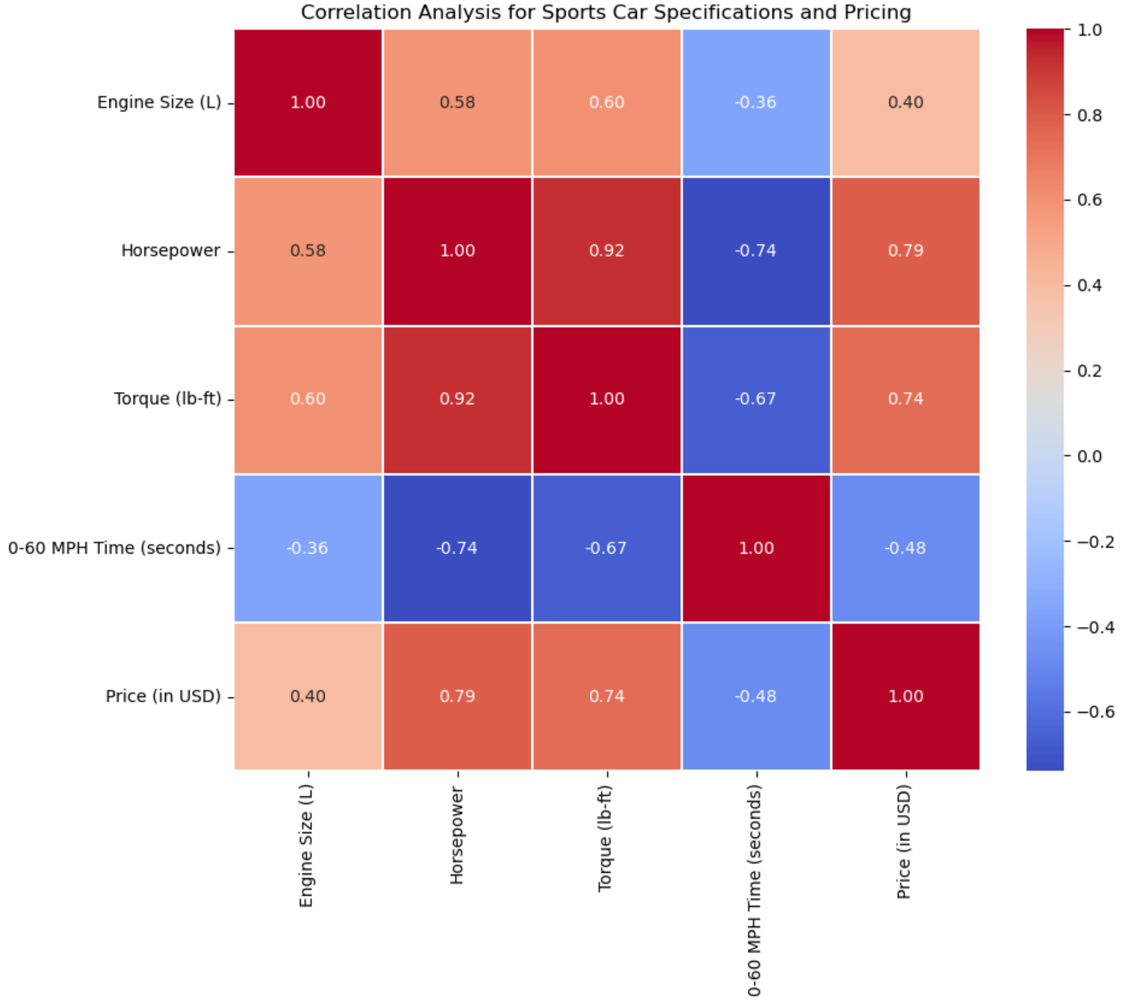
Out[2]:

	Car Make	Price (in USD)	Category
8	Bugatti	3.251957e+06	Hypercar Brands
15	Koenigsegg	2.906667e+06	Hypercar Brands
25	Pagani	2.791667e+06	Hypercar Brands
26	Pininfarina	2.500000e+06	Hypercar Brands
29	Rimac	2.400000e+06	Hypercar Brands
37	W Motors	2.216667e+06	Hypercar Brands
31	Shelby	1.000000e+06	Ultra-Premium / Luxury Brands
18	Lotus	5.084359e+05	Ultra-Premium / Luxury Brands
16	Lamborghini	4.259472e+05	Ultra-Premium / Luxury Brands
11	Ferrari	4.100991e+05	Ultra-Premium / Luxury Brands
12	Ford	3.688295e+05	Ultra-Premium / Luxury Brands
30	Rolls-Royce	3.332350e+05	Ultra-Premium / Luxury Brands

	Car Make	Price (in USD)	Category
21	McLaren	2.978079e+05	Premium Brands
36	Ultima	2.200000e+05	Premium Brands
7	Bentley	2.156290e+05	Premium Brands
4	Aston Martin	2.150791e+05	Premium Brands
22	Mercedes-AMG	1.693636e+05	Premium Brands
23	Mercedes-Benz	1.646614e+05	Premium Brands
34	Tesla	1.625274e+05	Premium Brands
0	Acura	1.578741e+05	Premium Brands
27	Polestar	1.550000e+05	Premium Brands
19	Maserati	1.476562e+05	Premium Brands
33	TVR	1.405000e+05	Premium Brands
28	Porsche	1.294784e+05	Premium Brands
5	Audi	9.387493e+04	Mid-Range Brands
17	Lexus	9.322885e+04	Mid-Range Brands
13	Jaguar	8.311833e+04	Mid-Range Brands
6	BMW	8.013413e+04	Mid-Range Brands
3	Ariel	7.500000e+04	Mid-Range Brands
1	Alfa Romeo	7.413406e+04	Mid-Range Brands
2	Alpine	7.150000e+04	Mid-Range Brands
10	Dodge	7.097683e+04	Mid-Range Brands
9	Chevrolet	5.524692e+04	Mid-Range Brands
14	Kia	5.220000e+04	Mid-Range Brands
24	Nissan	5.075216e+04	Mid-Range Brands
35	Toyota	4.307200e+04	Budget-Friendly Brands
32	Subaru	3.817000e+04	Budget-Friendly Brands
20	Mazda	2.683000e+04	Budget-Friendly Brands

In [7]:

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
data = pd.read_csv('Sport car price.csv')

# Clean and convert data
data['Price (in USD)'] = data['Price (in USD)'].replace('[\$,]', '', regex
data['Engine Size (L)'] = pd.to_numeric(data['Engine Size (L)'], errors='c
data['Horsepower'] = pd.to_numeric(data['Horsepower'], errors='coerce')
data['Torque (lb-ft)'] = pd.to_numeric(data['Torque (lb-ft)'], errors='coe
data['0-60 MPH Time (seconds)'] = pd.to_numeric(data['0-60 MPH Time (secon

# Check for any missing values and fill or drop them
data.dropna(inplace=True)

# Analyzing correlations
correlation_matrix = data[['Engine Size (L)', 'Horsepower', 'Torque (lb-ft
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", li
plt.title('Correlation Analysis for Sports Car Specifications and Pricing'
plt.show()
```

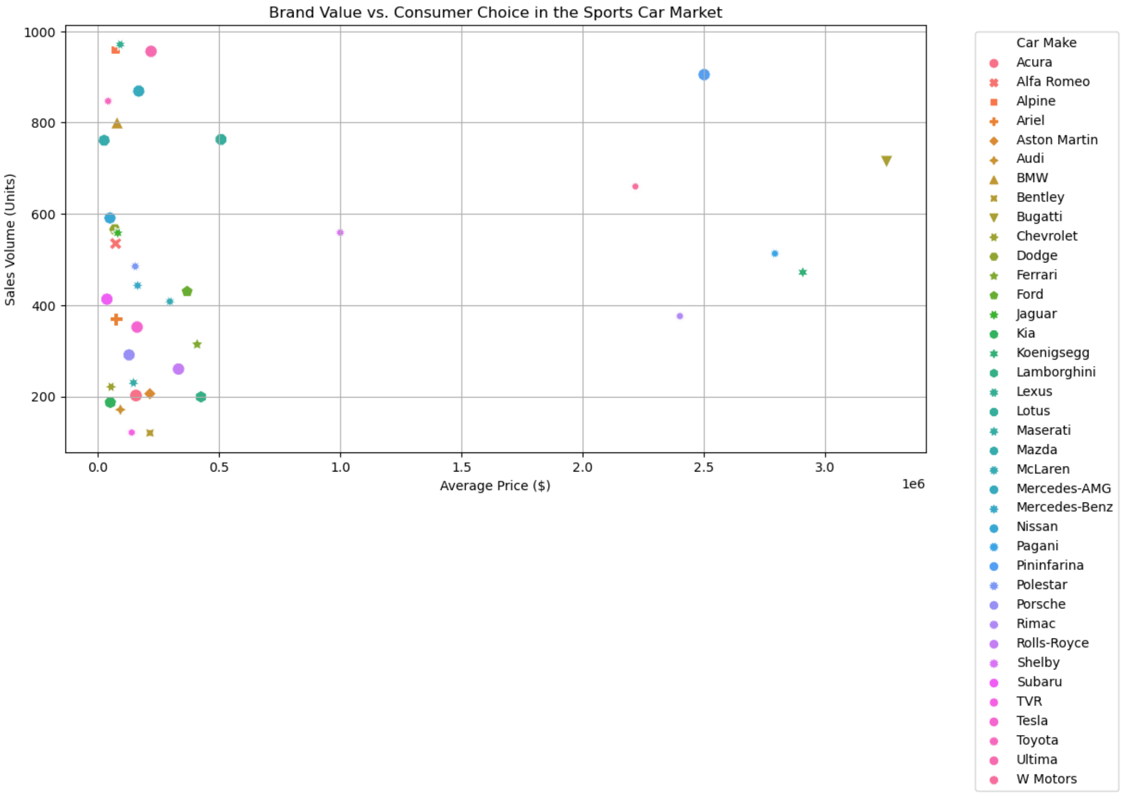


In [8]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
data = pd.read_csv('Sport car price.csv')

# Convert price data to numeric, removing any non-numeric characters
data['Price (in USD)'] = data['Price (in USD)'].replace('[\$,]', '', regex

# Calculate the average price for each car brand
average_price_by_brand = data.groupby('Car Make')['Price (in USD)'].mean()

# Assuming there's a column for sales volume or popularity (like number of
# Here, I'll create a sample column for illustrative purposes. Replace this
# Example: data['Sales Volume'] = [sample data values]

# For illustration, we will simulate a 'Sales Volume' column (Please provic
import numpy as np
np.random.seed(42)  # For reproducibility
average_price_by_brand['Sales Volume'] = np.random.randint(100, 1000, size

# Visualizing the relationship between average price and sales volume
plt.figure(figsize=(12, 6))
sns.scatterplot(x='Price (in USD)', y='Sales Volume', data=average_price_by
plt.title('Brand Value vs. Consumer Choice in the Sports Car Market')
plt.xlabel('Average Price ($)')
plt.ylabel('Sales Volume (Units)')
plt.legend(title='Car Make', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.grid(True)
plt.show()
```

```
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
```

```
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\vpram\anaconda3\anaconda\Lib\site-packages\seaborn\_oldcore.py:1
498: FutureWarning: is_categorical_dtype is deprecated and will be remove
d in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```



```
In [ ]:
```