ML EDA 2 Car sales(Module 2)

Questions

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

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

Out[2]:

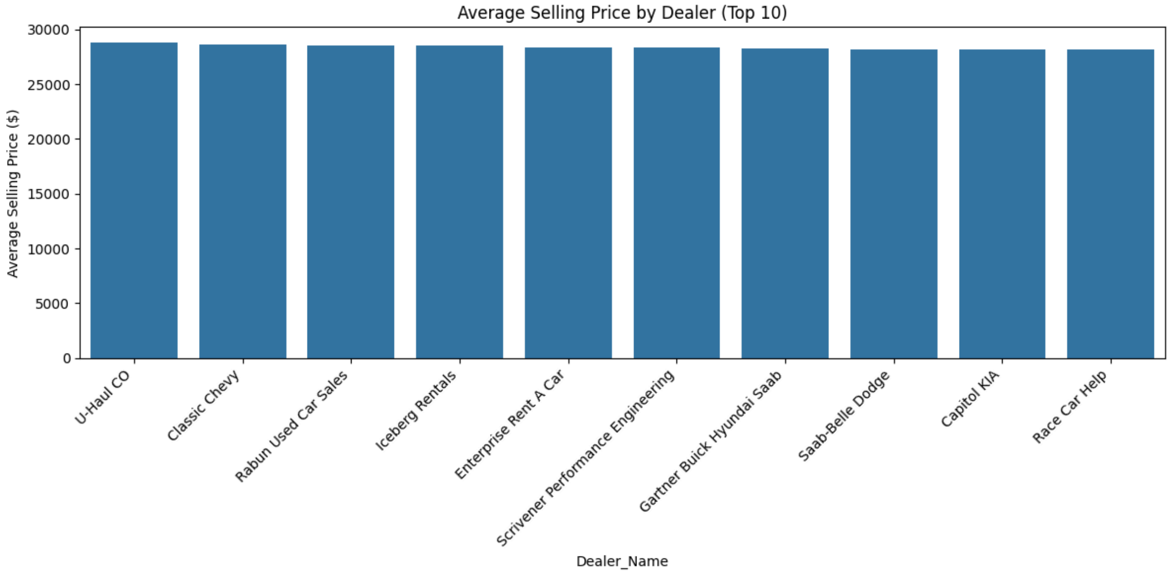
	Car_id	Date	Customer Name	Gender	Annual Income	Dealer_Name	Company
0	C_CND_000001	1/2/2022	Geraldine	Male	13500	Buddy Storbeck's Diesel Service Inc	Forc
1	C_CND_000002	1/2/2022	Gia	Male	1480000	C & M Motors Inc	Dodge
2	C_CND_000003	1/2/2022	Gianna	Male	1035000	Capitol KIA	Cadillac
3	C_CND_000004	1/2/2022	Giselle	Male	13500	Chrysler of Tri-Cities	Toyota
4	C_CND_000005	1/2/2022	Grace	Male	1465000	Chrysler Plymouth	Acura
...	...	...	...	...	...	...	..
23901	C_CND_023902	12/31/2023	Martin	Male	13500	C & M Motors Inc	Plymouth
23902	C_CND_023903	12/31/2023	Jimmy	Female	900000	Ryder Truck Rental and Leasing	Chevrole
23903	C_CND_023904	12/31/2023	Emma	Male	705000	Chrysler of Tri-Cities	BMW
23904	C_CND_023905	12/31/2023	Victoire	Male	13500	Chrysler Plymouth	Chevrole
23905	C_CND_023906	12/31/2023	Donovan	Male	1225000	Pars Auto Sales	Lexus

23906 rows × 16 columns

In [3]:

```
# Q1 What is the average selling price of cars for each dealer, and how does it
# Calculate the average selling price for each dealer
avg_price_per_dealer = df.groupby('Dealer_Name')['Price ($)'].mean().sort_values
print(avg_price_per_dealer)

# Visualize the comparison across different dealers (top 10 for clarity)
plt.figure(figsize=(12,6))
sns.barplot(x=avg_price_per_dealer.head(10).index, y=avg_price_per_dealer.head(1
plt.xticks(rotation=45, ha='right')
plt.ylabel('Average Selling Price ($)')
plt.title('Average Selling Price by Dealer (Top 10)')
plt.tight_layout()
plt.show()
```

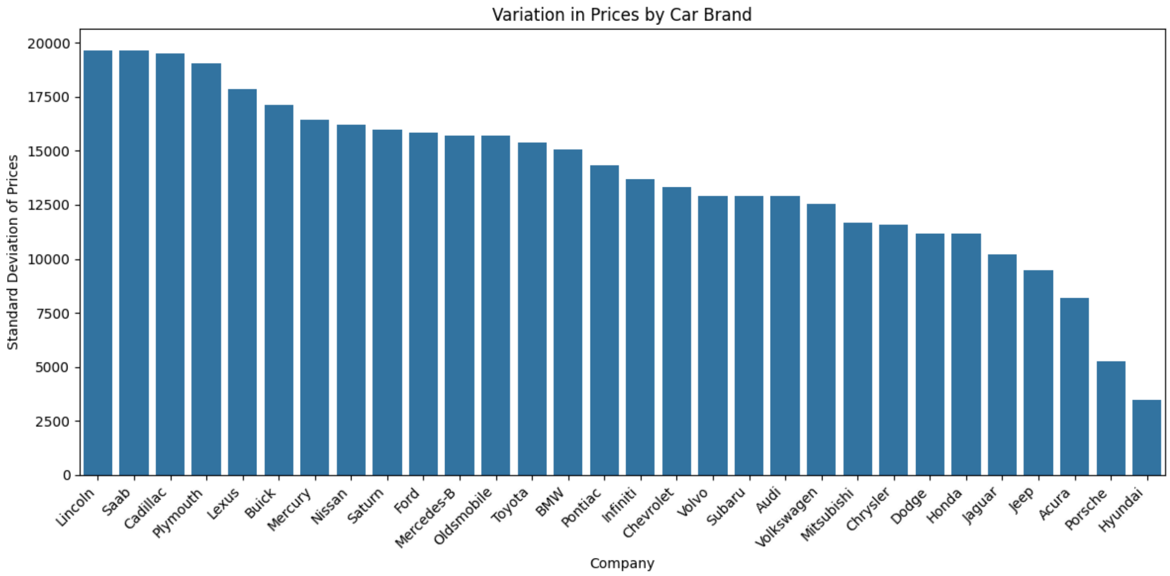
```
Dealer_Name
U-Haul CO                                            28769.919006
Classic Chevy                                        28602.014446
Rabun Used Car Sales                                 28527.536177
Iceberg Rentals                                      28522.958533
Enterprise Rent A Car                                28312.580800
Scrivener Performance Engineering                    28297.371589
Gartner Buick Hyundai Saab                           28247.621019
Saab-Belle Dodge                                     28190.139888
Capitol KIA                                          28189.703822
Race Car Help                                        28163.372706
Chrysler of Tri-Cities                               28123.091054
Star Enterprises Inc                                 28113.055244
Suburban Ford                                        28112.206758
C & M Motors Inc                                     28111.755200
Tri-State Mack Inc                                   28095.562050
Pars Auto Sales                                      28013.060317
Diehl Motor CO Inc                                   27993.929487
Motor Vehicle Branch Office                          27956.739617
Ryder Truck Rental and Leasing                       27914.988782
Progressive Shippers Cooperative Association No      27884.264036
New Castle Ford Lincoln Mercury                      27867.131955
Hatfield Volkswagen                                  27853.712242
Nebo Chevrolet                                       27818.889415
Clay Johnson Auto Sales                              27816.027113
McKinney Dodge Chrysler Jeep                         27684.096979
Chrysler Plymouth                                    27555.526400
Pitre Buick-Pontiac-Gmc of Scottsdale                27404.248408
Buddy Storbeck's Diesel Service Inc                  27217.261563
Name: Price ($), dtype: float64
```



In [4]:
```
# Q2 Which car brand (company) has the highest variation in prices, and what doe

# Calculate the standard deviation of prices for each car brand
std_dev_per_brand = df.groupby('Company')['Price ($)'].std().sort_values(ascendi
print(std_dev_per_brand)

# Visualize the variation in prices by car brand
plt.figure(figsize=(12,6))
sns.barplot(x=std_dev_per_brand.index, y=std_dev_per_brand.values)
plt.xticks(rotation=45, ha='right')
plt.ylabel('Standard Deviation of Prices')
plt.title('Variation in Prices by Car Brand')
```

```
plt.tight_layout()
plt.show()
```

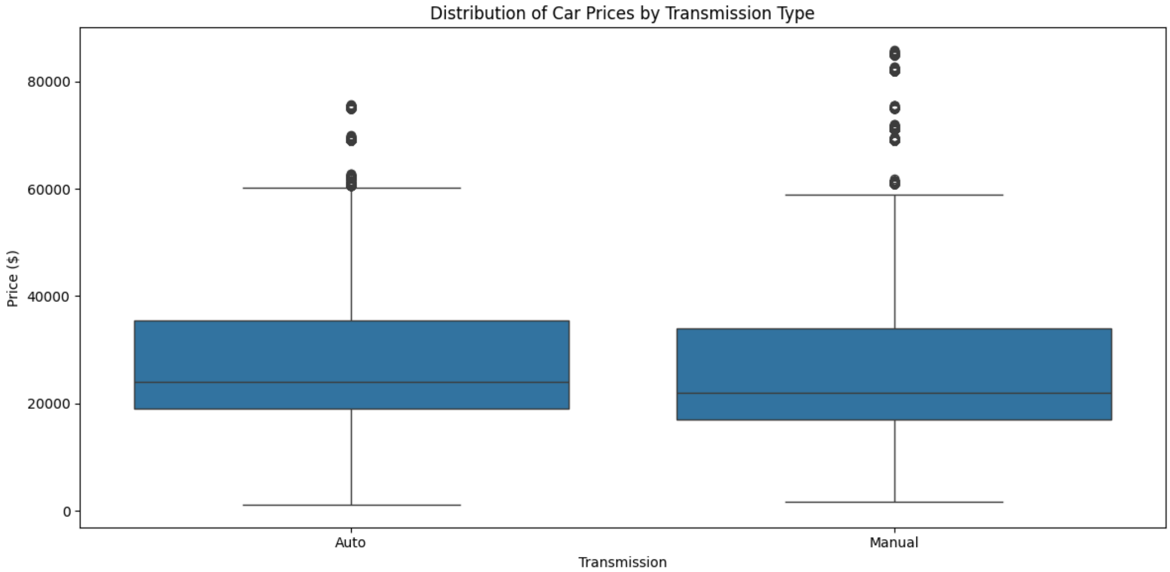
```
Company
Lincoln        19658.050211
Saab           19653.740089
Cadillac       19517.120220
Plymouth       19065.997338
Lexus          17852.923492
Buick          17142.232626
Mercury        16445.172195
Nissan         16214.264017
Saturn         15990.223671
Ford           15849.090227
Mercedes-B     15722.807459
Oldsmobile     15711.345857
Toyota         15367.131714
BMW            15065.578723
Pontiac        14348.963592
Infiniti       13696.332844
Chevrolet      13311.063223
Volvo          12933.790185
Subaru         12920.771620
Audi           12904.243867
Volkswagen     12527.124011
Mitsubishi     11671.343035
Chrysler       11583.286811
Dodge          11187.592085
Honda          11148.629062
Jaguar         10222.531533
Jeep            9459.834418
Acura           8183.046414
Porsche         5261.839206
Hyundai         3485.982649
Name: Price ($), dtype: float64
```



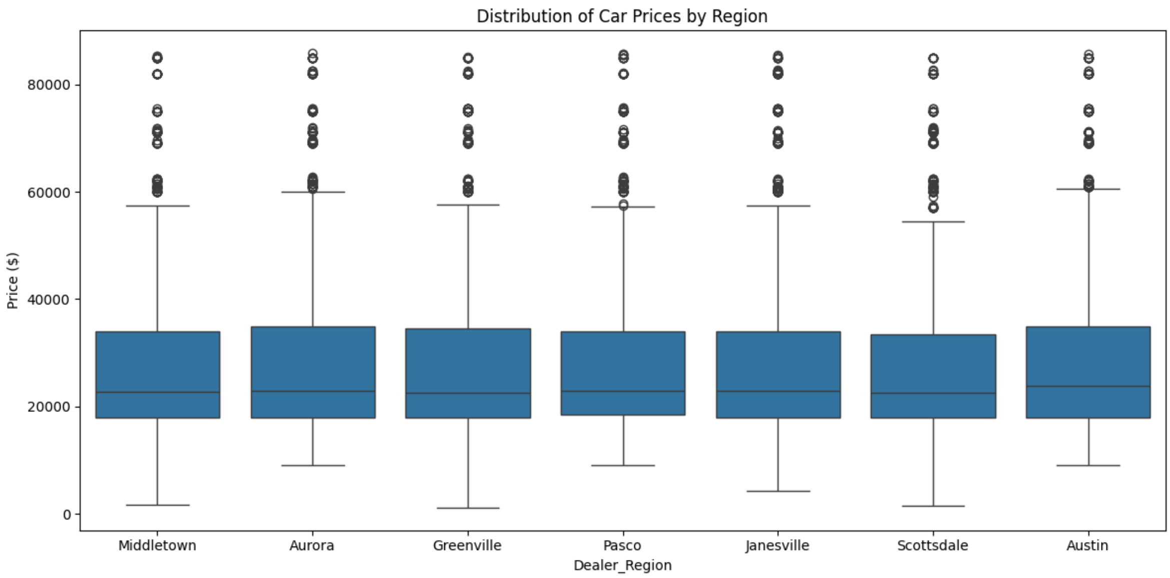
In [6]:
```
# Q3 What is the distribution of car prices for each transmission type, and how

# Calculate the interquartile range (IQR) for each transmission type
iqr_per_transmission = df.groupby('Transmission')['Price ($)'].agg(
        count='count',
        mean='mean',
        std='std',
```

```
        min='min',
        q25=lambda x: x.quantile(0.25),
        q50=lambda x: x.quantile(0.50),
        q75=lambda x: x.quantile(0.75),
        max='max'
)
print(iqr_per_transmission)

# Visualize the distribution of car prices for each transmission type
plt.figure(figsize=(12,6))
sns.boxplot(x='Transmission', y='Price ($)', data=df)
plt.title('Distribution of Car Prices by Transmission Type')
plt.tight_layout()
plt.show()
```

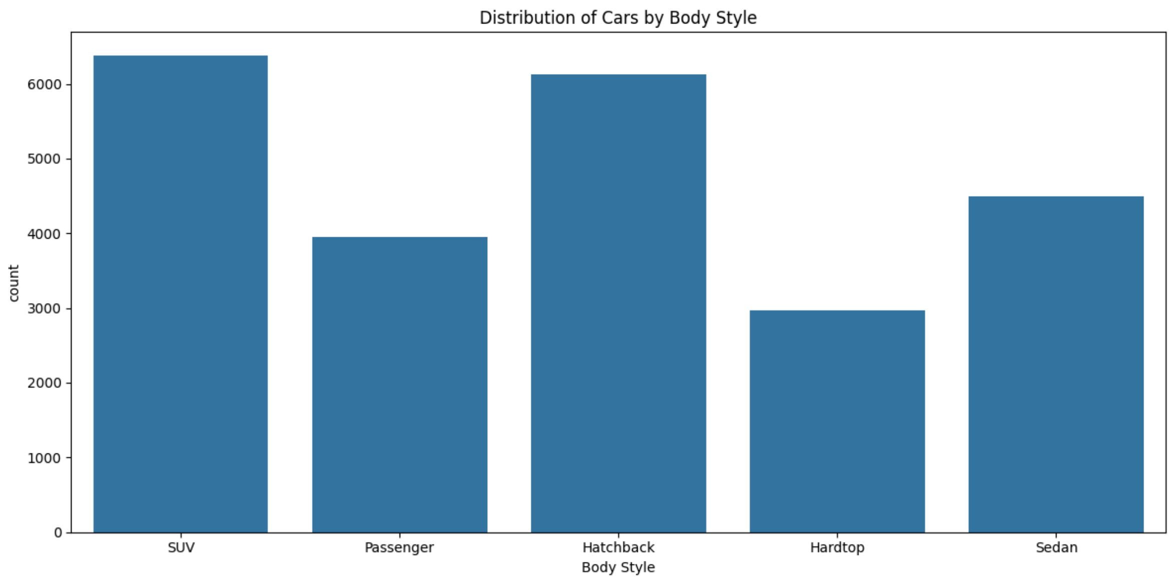
```
              count          mean           std   min     q25     q50  \
Transmission
Auto          12571  28248.525972  13747.070597  1200  19000.0  24000.0
Manual        11335  27914.710631  15862.871978  1700  17000.0  22001.0

                  q75    max
Transmission
Auto          35500.0  75700
Manual        34000.0  85800
```



In [8]:
```
# Q4 What is the distribution of car prices across different regions?

# Visualize the distribution of car prices across different regions
plt.figure(figsize=(12,6))
sns.boxplot(x='Dealer_Region', y='Price ($)', data=df)
plt.title('Distribution of Car Prices by Region')
plt.tight_layout()
plt.show()
```

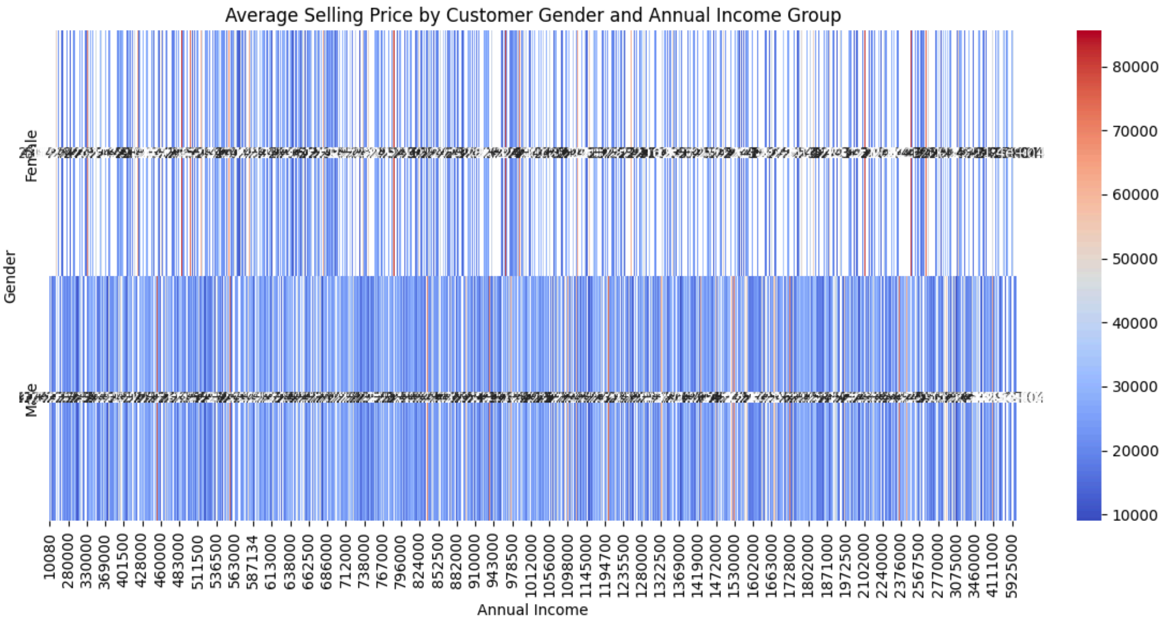


In [10]:
```
# Q5 What is the distribution of cars based on body styles?

# Visualize the distribution of cars based on body styles
plt.figure(figsize=(12,6))
sns.countplot(x='Body Style', data=df)
plt.title('Distribution of Cars by Body Style')
plt.tight_layout()
plt.show()
```



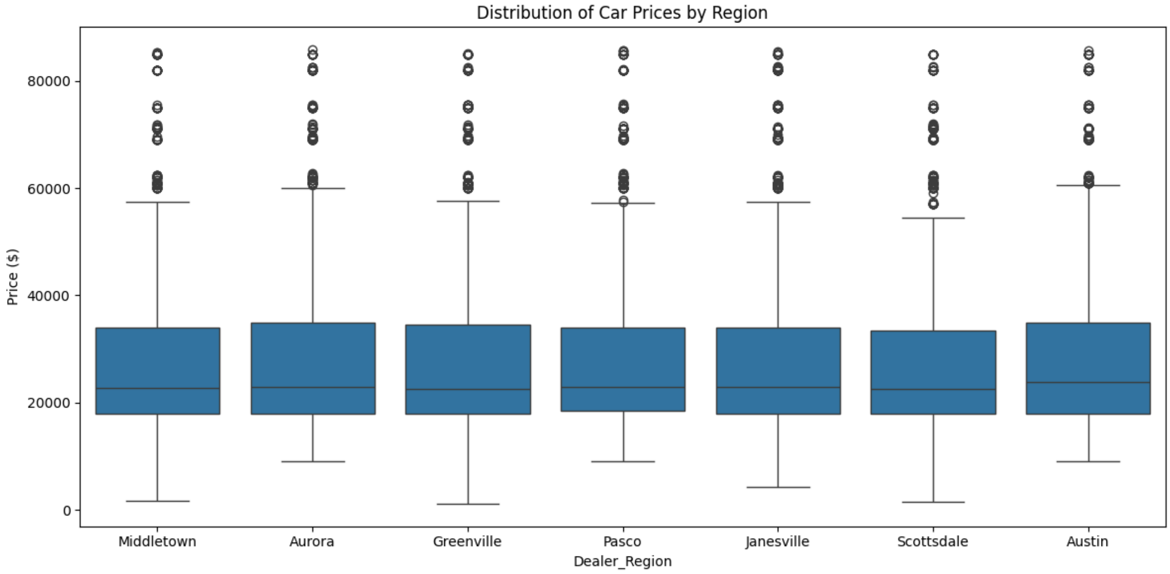
In [12]:
```
# Q6 How does the average selling price of cars vary by customer gender and annu

# Calculate the average selling price for each customer gender and annual income
avg_price_per_gender_income = df.groupby(['Gender', 'Annual Income'])['Price ($)
print(avg_price_per_gender_income)

# Visualize the average selling price by customer gender and annual income group
plt.figure(figsize=(12,6))
sns.heatmap(avg_price_per_gender_income, annot=True, cmap='coolwarm')
plt.xlabel('Annual Income')
plt.ylabel('Gender')
plt.title('Average Selling Price by Customer Gender and Annual Income Group')
plt.tight_layout()
plt.show()
```

```
Annual Income  10080         13500      24000      85000      106000     121000    \
Gender
Female           NaN  28132.038732        NaN        NaN     46001.0    20000.0
Male         22801.0  27809.493111    61001.0    43000.0         NaN        NaN

Annual Income  131000     145000     160000     170000     ...  6125000   \
Gender                                                     ...
Female           NaN        NaN        NaN        NaN     ...      NaN
Male         17000.0    16500.0    18334.0    14500.0     ...  19501.0

Annual Income  6240000    6400000    6460000    6500000    6600000    6800000   \
Gender
Female       42000.0    32001.0    14000.0        NaN        NaN        NaN
Male             NaN    71000.0        NaN    25000.0    39000.0    15000.0

Annual Income  7650000    8000000    11200000
Gender
Female           NaN        NaN        NaN
Male         21000.0    85000.0    26001.0

[2 rows x 2508 columns]
```



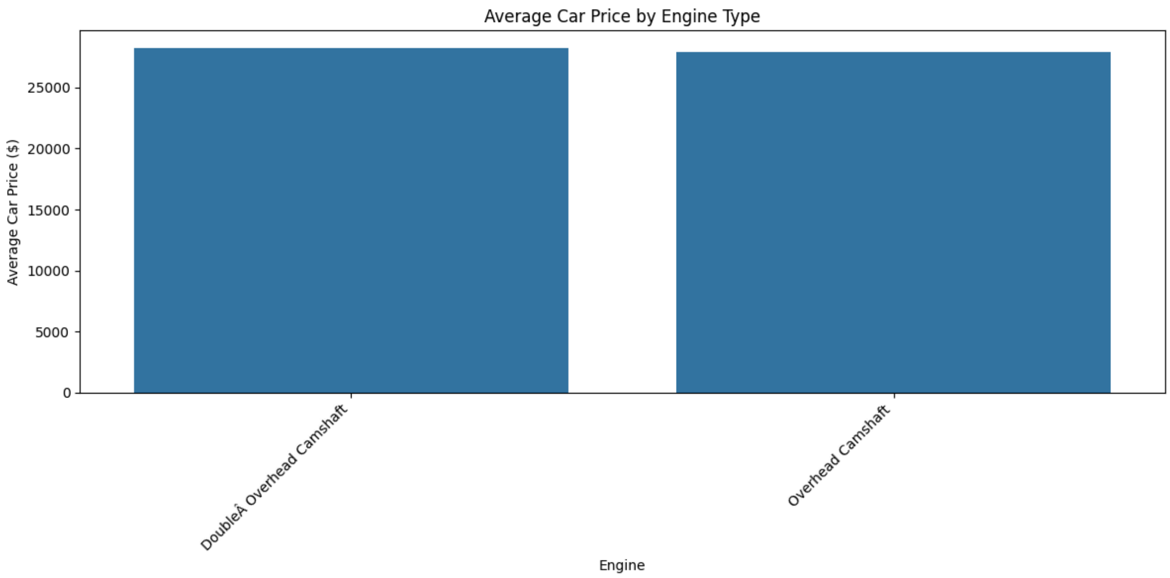
In [13]:
```python
# Q7 What is the distribution of car prices by region, and how does the number o

# Calculate the number of cars sold for each region
cars_sold_per_region = df.groupby('Dealer_Region')['Price ($)'].count().sort_val
print(cars_sold_per_region)

# Visualize the distribution of car prices by region
plt.figure(figsize=(12,6))
sns.boxplot(x='Dealer_Region', y='Price ($)', data=df)
plt.title('Distribution of Car Prices by Region')
plt.tight_layout()
plt.show()
```

```
Dealer_Region
Austin        4135
Janesville    3821
Scottsdale    3433
Pasco         3131
Aurora        3130
Greenville    3128
Middletown    3128
Name: Price ($), dtype: int64
```



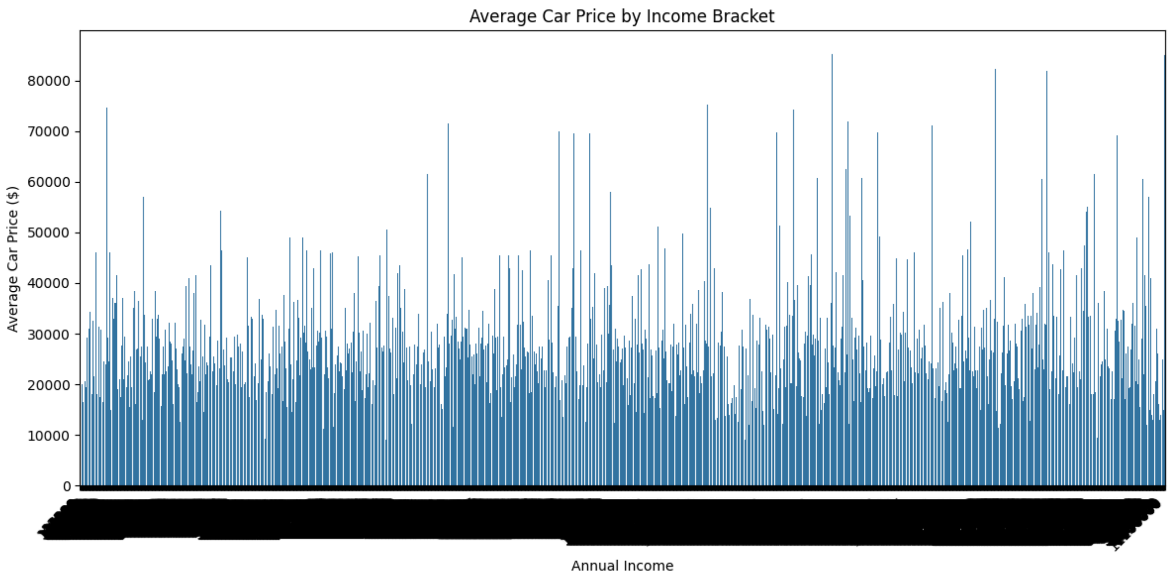
In [15]:
```
# Q8 How does the average car price differ between cars with different engine si

# Calculate the average car price for each engine type
avg_price_per_engine = df.groupby('Engine')['Price ($)'].mean().sort_values(asce
print(avg_price_per_engine)

# Visualize the average car price by engine type
plt.figure(figsize=(12,6))
sns.barplot(x=avg_price_per_engine.index, y=avg_price_per_engine.values)
plt.xticks(rotation=45, ha='right')
plt.ylabel('Average Car Price ($)')
plt.title('Average Car Price by Engine Type')
plt.tight_layout()
plt.show()
```

```
Engine
DoubleÂ Overhead Camshaft    28248.525972
Overhead Camshaft            27914.710631
Name: Price ($), dtype: float64
```



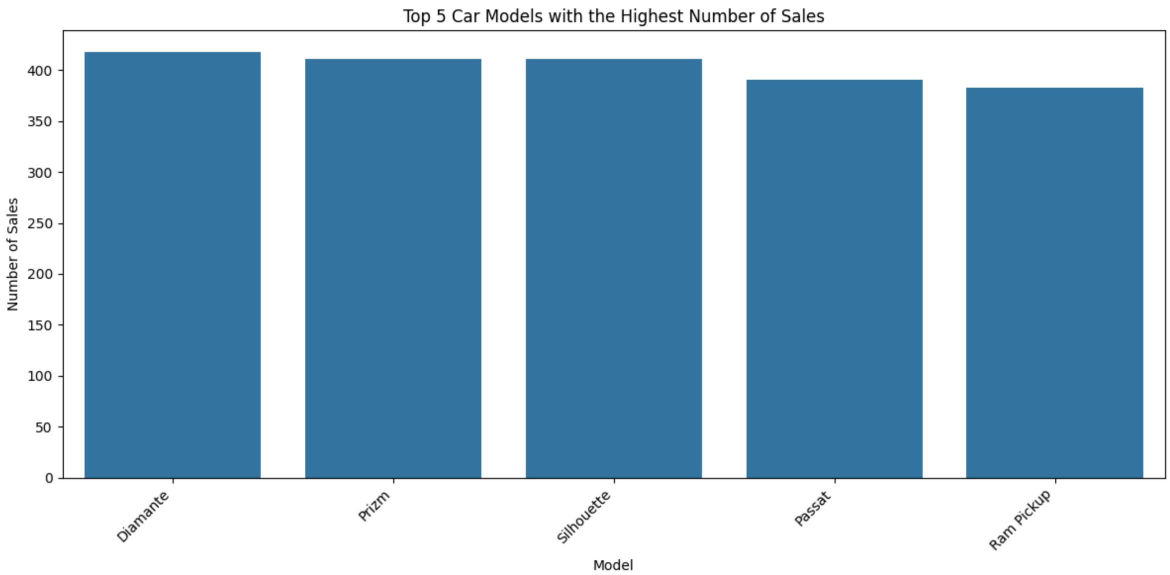
In [16]:
```
# Q9 How do car prices vary based on the customer's annual income bracket?

# Calculate the average car price for each income bracket
avg_price_per_income = df.groupby('Annual Income')['Price ($)'].mean().sort_valu
print(avg_price_per_income)

# Visualize the average car price by income bracket
plt.figure(figsize=(12,6))
sns.barplot(x=avg_price_per_income.index, y=avg_price_per_income.values)
plt.xticks(rotation=45, ha='right')
plt.ylabel('Average Car Price ($)')
plt.title('Average Car Price by Income Bracket')
plt.tight_layout()
plt.show()
```

```
Annual Income
5046000    85601.0
1414000    85400.0
1483000    85301.0
8000000    85000.0
785500     82500.0
            ...
2151000     9100.0
1281000     9100.0
273000      9001.0
679000      9000.0
338000      9000.0
Name: Price ($), Length: 2508, dtype: float64
```



In [17]:
```
# Q10 What are the top 5 car models with the highest number of sales, and how do

# Calculate the number of sales for each car model
sales_per_model = df.groupby('Model')['Price ($)'].count().sort_values(ascending
print(sales_per_model)

# Visualize the top 5 car models with the highest number of sales
plt.figure(figsize=(12,6))
sns.barplot(x=sales_per_model.index, y=sales_per_model.values)
plt.xticks(rotation=45, ha='right')
plt.ylabel('Number of Sales')
plt.title('Top 5 Car Models with the Highest Number of Sales')
plt.tight_layout()
plt.show()
```

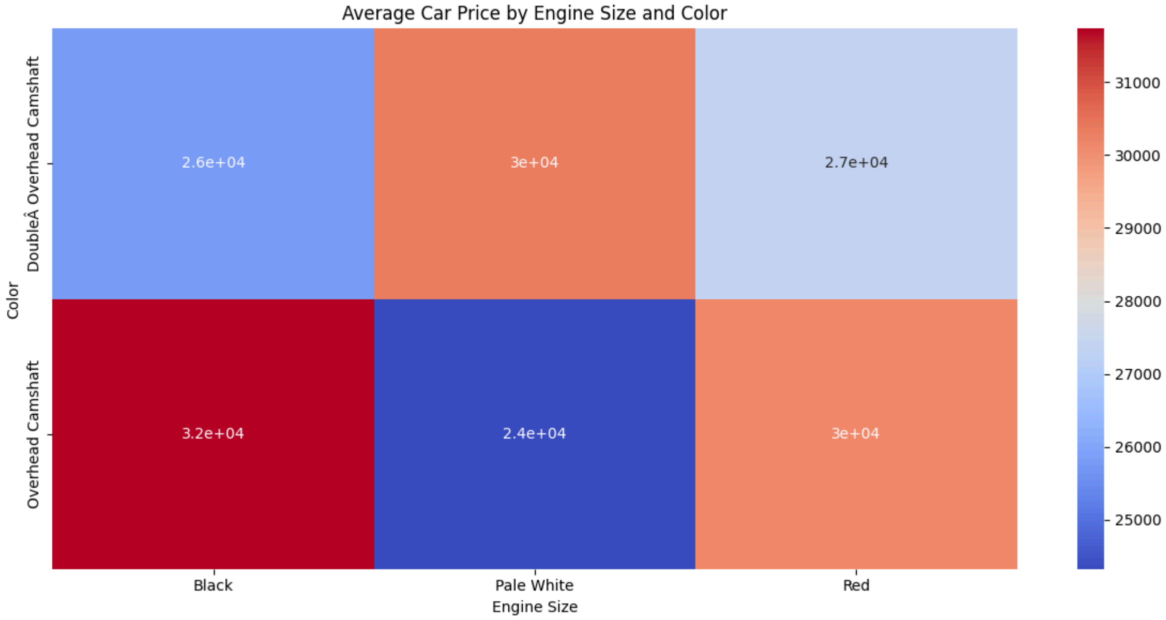
```
Model
Diamante      418
Prizm         411
Silhouette    411
Passat        391
Ram Pickup    383
Name: Price ($), dtype: int64
```



In [18]:
```
# Q11 How does car price vary with engine size across different car colors, and
```

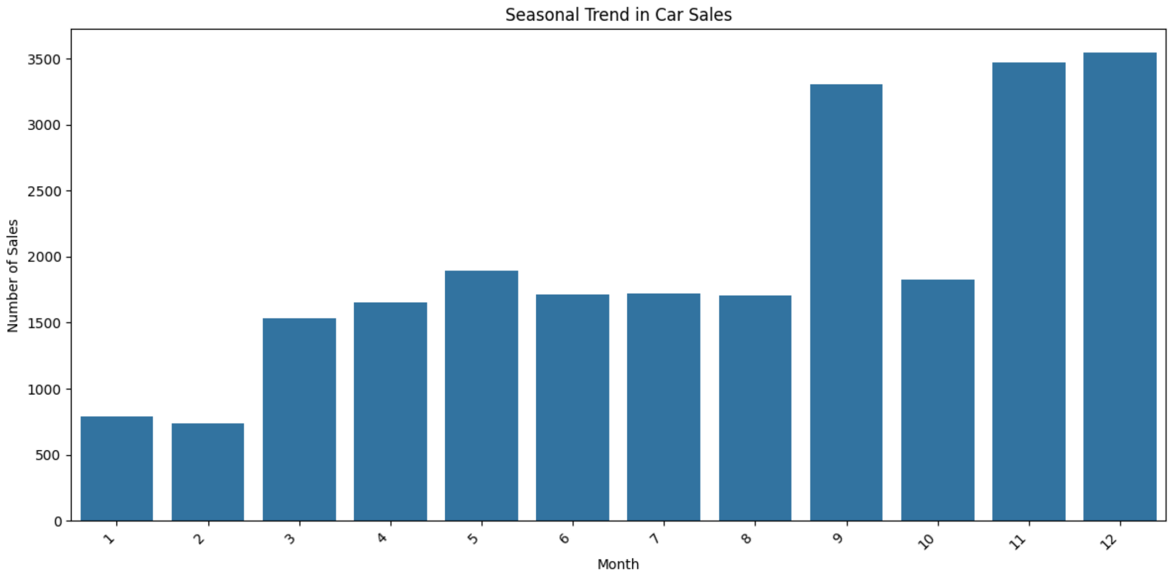
```
# Calculate the average car price for each engine size and color
avg_price_per_engine_color = df.groupby(['Engine', 'Color'])['Price ($)'].mean()
print(avg_price_per_engine_color)

# Visualize the average car price by engine size and color
plt.figure(figsize=(12,6))
sns.heatmap(avg_price_per_engine_color, annot=True, cmap='coolwarm')
plt.xlabel('Engine Size')
plt.ylabel('Color')
plt.title('Average Car Price by Engine Size and Color')
plt.tight_layout()
plt.show()
```

```
Color                             Black    Pale White           Red
Engine
DoubleÂ Overhead Camshaft  25766.451103  30347.987460  27434.392314
Overhead Camshaft          31732.967164  24327.767694  30153.405229
```



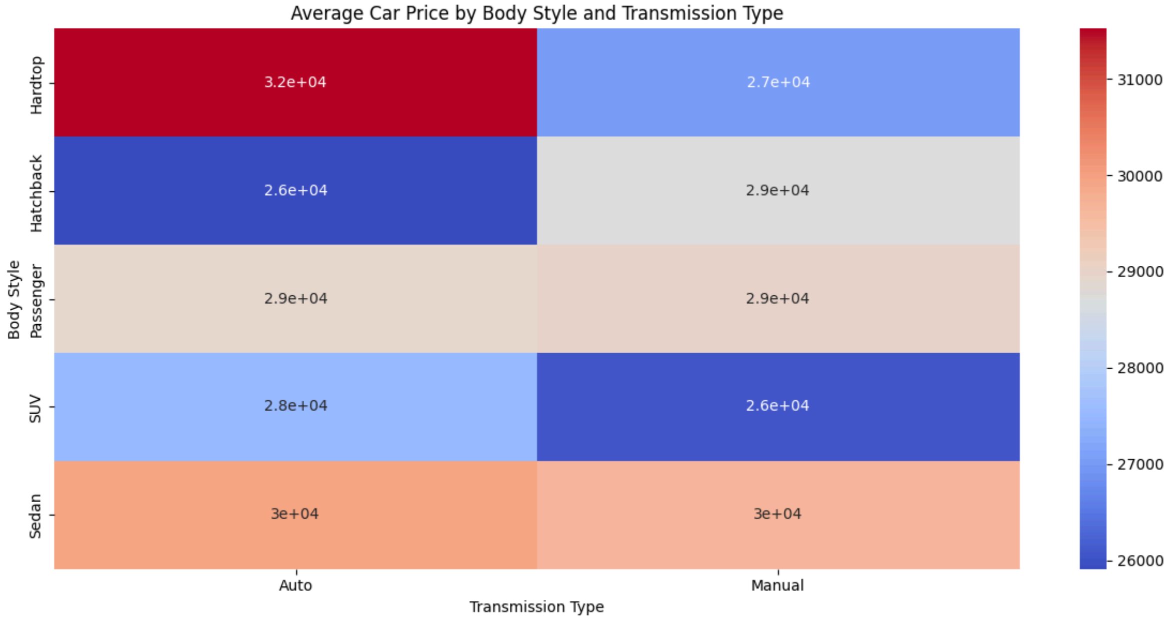
In [19]:
```
# Q12 Is there any seasonal trend in car sales based on the date of sale?

# Convert the 'Date' column to datetime format
df['Date'] = pd.to_datetime(df['Date'])

# Extract the month from the 'Date' column
df['Month'] = df['Date'].dt.month

# Calculate the number of sales for each month
sales_per_month = df.groupby('Month')['Price ($)'].count().sort_values(ascending
print(sales_per_month)

# Visualize the seasonal trend in car sales
plt.figure(figsize=(12,6))
sns.barplot(x=sales_per_month.index, y=sales_per_month.values)
plt.xticks(rotation=45, ha='right')
plt.ylabel('Number of Sales')
plt.title('Seasonal Trend in Car Sales')
plt.tight_layout()
plt.show()
```

```
Month
12    3546
11    3470
9     3305
5     1895
10    1830
7     1725
6     1715
8     1705
4     1655
3     1535
1      790
2      735
Name: Price ($), dtype: int64
```



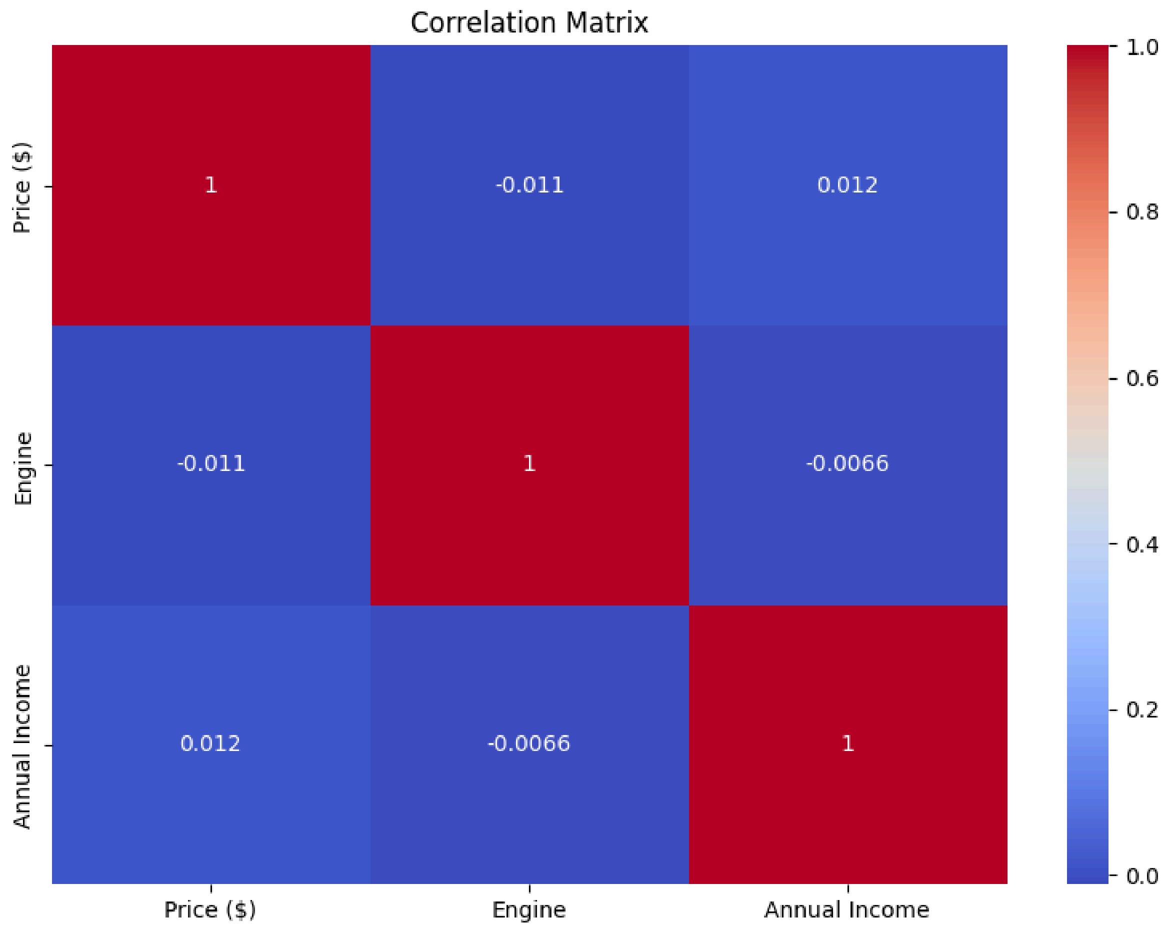
In [20]:
```
# Q13 How does the car price distribution change when considering different comb

# Calculate the average car price for each combination of body style and transmi
avg_price_per_body_transmission = df.groupby(['Body Style', 'Transmission'])['Pr
print(avg_price_per_body_transmission)

# Visualize the car price distribution by body style and transmission type
plt.figure(figsize=(12,6))
sns.heatmap(avg_price_per_body_transmission, annot=True, cmap='coolwarm')
plt.xlabel('Transmission Type')
plt.ylabel('Body Style')
plt.title('Average Car Price by Body Style and Transmission Type')
plt.tight_layout()
plt.show()
```

```
Transmission          Auto        Manual
Body Style
Hardtop       31520.188210  27016.943698
Hatchback     25910.544824  28702.550562
Passenger     28915.835149  28969.521039
SUV           27501.404407  26079.019161
Sedan         29955.294344  29664.271572
```



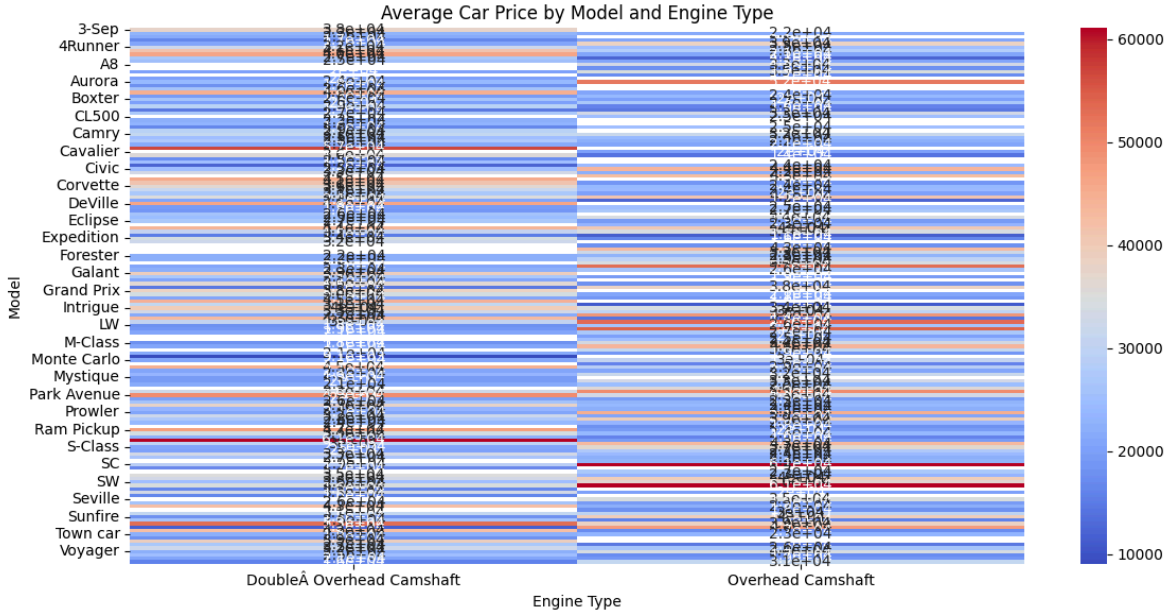
In [22]:
```
# Q14 What is the correlation between car price, engine size, and annual income

# Convert 'Engine' column to numeric using label encoding
df['Engine_encoded'] = df['Engine'].astype('category').cat.codes

# Calculate the correlation matrix
corr_matrix = df[['Price ($)', 'Engine_encoded', 'Annual Income']].corr()
print(corr_matrix)

# Visualize the correlation matrix
plt.figure(figsize=(8,6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', xticklabels=['Price ($)',
plt.title('Correlation Matrix')
plt.tight_layout()
plt.show()
```

```
                Price ($)  Engine_encoded  Annual Income
Price ($)        1.000000       -0.011271       0.012065
Engine_encoded  -0.011271        1.000000      -0.006598
Annual Income    0.012065       -0.006598       1.000000
```



In [23]:
```
# Q15 How does the average car price vary across different car models and engine

# Calculate the average car price for each model and engine type
avg_price_per_model_engine = df.groupby(['Model', 'Engine'])['Price ($)'].mean()
print(avg_price_per_model_engine)

# Visualize the average car price by model and engine type
plt.figure(figsize=(12,6))
sns.heatmap(avg_price_per_model_engine, annot=True, cmap='coolwarm')
plt.xlabel('Engine Type')
plt.ylabel('Model')
plt.title('Average Car Price by Model and Engine Type')
plt.tight_layout()
plt.show()
```

```
Engine     DoubleÂ Overhead Camshaft  Overhead Camshaft
Model
3-Sep                  37986.380117                NaN
3000GT                 22764.326923       21770.659864
300M                   21394.888889                NaN
323i                   16744.632287       21038.162162
328i                   21069.149606       38676.177215
...                             ...                ...
Viper                  32118.479167       26052.375000
Voyager                22066.026316       33995.678322
Windstar               25145.636364       17100.272727
Wrangler               21145.294737       18742.942029
Xterra                 15940.459459       31072.500000

[154 rows x 2 columns]
```


Average Car Price by Model and Engine Type
Model
3-Sep
4Runner
A8
Aurora
Boxter
CL500
Camry
Cavalier
Civic
Corvette
DeVille
Eclipse
Expedition
Forester
Galant
Grand Prix
Intrigue
LW
M-Class
Monte Carlo
Mystique
Park Avenue
Prowler
Ram Pickup
S-Class
SC
SW
Seville
Sunfire
Town car
Voyager
DoubleÂ Overhead Camshaft
Overhead Camshaft
Engine Type
60000
50000
40000
30000
20000
10000

In [ ]: