In [1]:
```
# Importing necessary libraries

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

# Load the dataset
med_ins = pd.read_csv(r'C:\Users\Rohit kumar\Downloads\pythond\medical_insurance.csv

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(med_ins.head())
```

```
First few rows of the dataset:
   age  sex     bmi  children  smoker  region      charges
0   19    0  27.900         0       0       1  16884.92400
1   18    1  33.770         1       1       2   1725.55230
2   28    1  33.000         3       1       2   4449.46200
3   33    1  22.705         0       1       3  21984.47061
4   32    1  28.880         0       1       3   3866.85520
```

In [2]:
```
# Statistical summary of the dataset
print("Statistical summary of the dataset:")
print(med_ins.describe())

# Check for any missing values
print("\nNumber of missing values in each column:")
print(med_ins.isnull().sum())

```

```
Statistical summary of the dataset:
               age          sex          bmi     children       smoker  \
count  2772.000000  2772.000000  2772.000000  2772.000000  2772.000000
mean     39.109668     0.507215    30.701349     1.101732     0.796537
std      14.081459     0.500038     6.129449     1.214806     0.402647
min      18.000000     0.000000    15.960000     0.000000     0.000000
25%      26.000000     0.000000    26.220000     0.000000     1.000000
50%      39.000000     1.000000    30.447500     1.000000     1.000000
75%      51.000000     1.000000    34.770000     2.000000     1.000000
max      64.000000     1.000000    53.130000     5.000000     1.000000

            region       charges
count  2772.000000   2772.000000
mean      2.467532  13261.369959
std       1.103467  12151.768945
min       1.000000   1121.873900
25%       2.000000   4687.797000
50%       2.000000   9333.014350
75%       3.000000  16577.779500
max       4.000000  63770.428010

Number of missing values in each column:
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [3]:
```
#check the data type
print("Data type")
print(med_ins.dtypes)

```

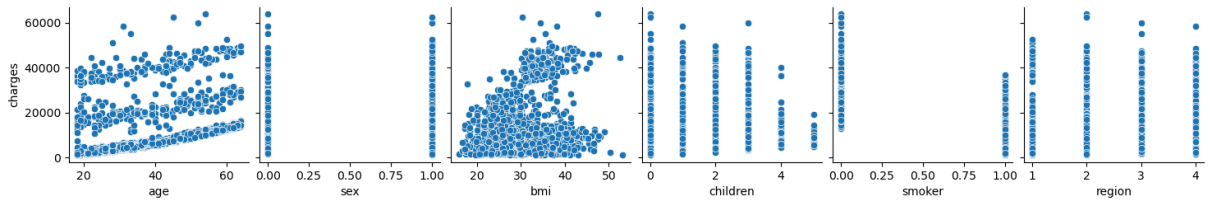
```
Data type
age           int64
sex           int64
bmi         float64
children      int64
smoker        int64
region        int64
charges     float64
dtype: object
```

In [4]:
```
X = med_ins.iloc[:, [0,1,2,3,4,5]].values
Y = med_ins.iloc[:, -1].values

print(X)
```

```
[[19.     0.    27.9    0.     0.     1.   ]
 [18.     1.    33.77   1.     1.     2.   ]
 [28.     1.    33.     3.     1.     2.   ]
 ...
 [19.     1.    26.03   1.     0.     3.   ]
 [23.     1.    18.715  0.     1.     3.   ]
 [54.     1.    31.6    0.     1.     1.   ]]
```

In [31]:
```
# Visualize the relationships between the independent variables and the target varial
sns.pairplot(med_ins, x_vars=['age', 'sex', 'bmi', 'children', 'smoker', 'region'], 
plt.show()

```

```
C:\Users\Rohit kumar\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```




In [7]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_stat
```

In [8]:
```
regressor = LinearRegression()
regressor.fit(X_train, Y_train)
```

Out[8]:
```
▾ LinearRegression
LinearRegression()
```

In [9]:
```
Y_pred = regressor.predict(X_test)
np.set_printoptions(precision=2)
print(np.concatenate((Y_pred.reshape(len(Y_pred),1), Y_test.reshape(len(Y_test),1)),
```

```
[[ 1731.92  2221.56]
 [   27.54 21595.38]
 [ 7754.66  5327.4 ]
 ...
 [ 9721.78  9301.89]
 [13086.02  7650.77]
 [ 4689.88  6640.54]]
```

In [29]:
```
# Define the input features for prediction
input_data = [[25, 1, 30, 3, 0, 1]]  # age=25, sex=male, bmi=30, children=3, smoker=
# Make prediction
predicted_charge = regressor.predict(input_data)[0]
predicted_charge_round= round(predicted_charge,2)
print("Predicted medical charge for the given scenario:", "Rs",predicted_charge_roun
```

```
Predicted medical charge for the given scenario: Rs 29054.92
```

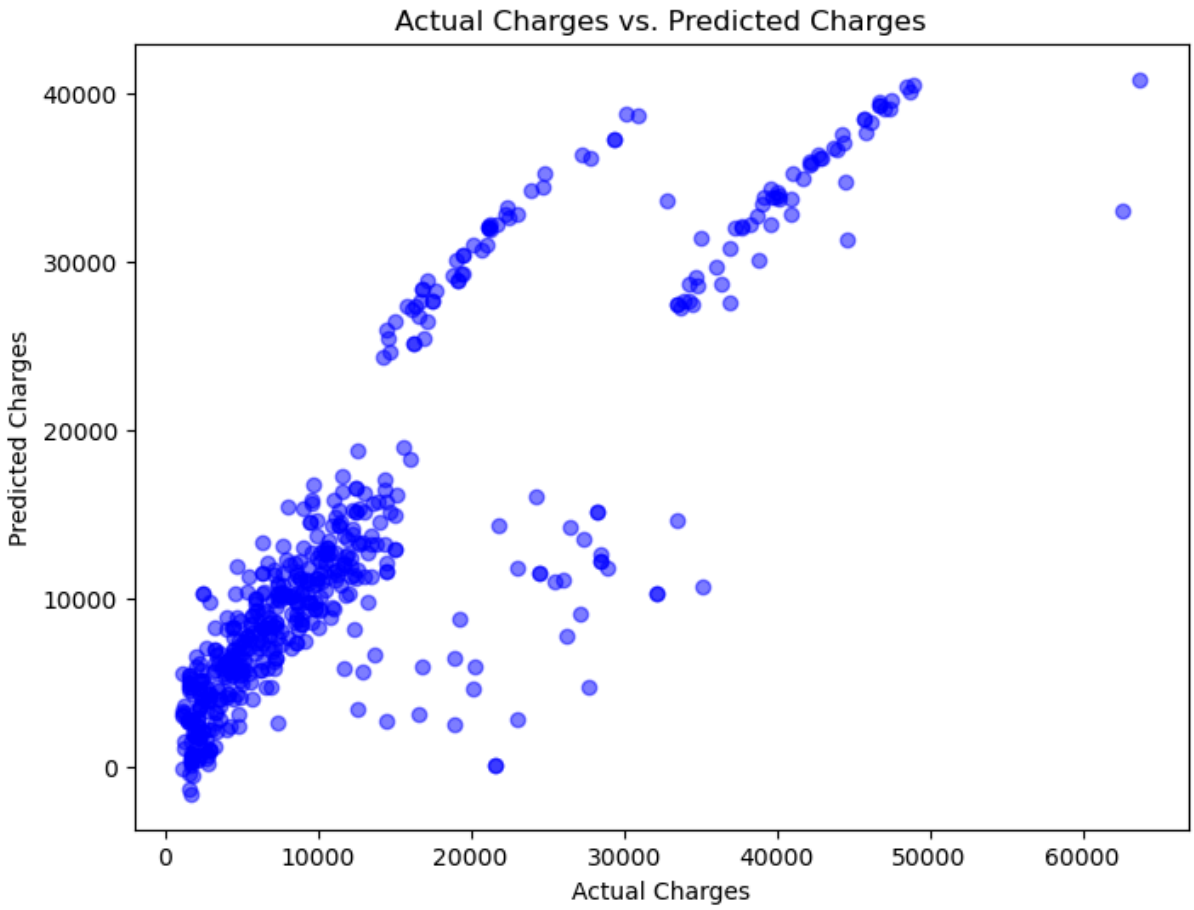
In [19]:
```
print(regressor.intercept_, regressor.coef_, regressor.score(X, Y))
```

```
11329.413044173816 [   253.28   -145.14    317.56    545.44 -24052.04    375.58] 0.7507
060330466141
```

In [25]:
```
result = r2_score(Y_test, Y_pred)
R2 = round(result, 2)
R2_percent = R2 * 100
print("The R^2 is:", R2_percent, "percent")
```

```
The R^2 is: 75.0 percent
```

In [32]:
```
# Make predictions on the test set
Y_pred = regressor.predict(X_test)

# Scatter plot of actual charges vs. predicted charges
plt.figure(figsize=(8, 6))
plt.scatter(Y_test, Y_pred, color='blue', alpha=0.5)
plt.title('Actual Charges vs. Predicted Charges')
plt.xlabel('Actual Charges')
plt.ylabel('Predicted Charges')
plt.show()

```




In [ ]:
```
1
```