Use sci-kit-learn to implement linear regression on a dataset to predict a continuous target variable.

In [10]:
```
from sklearn.datasets import load_diabetes
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import numpy as np

# Load dataset
diabetes = load_diabetes()

# Select only one feature for 2D visualization (e.g., BMI feature at index 2)
X = diabetes.data[:, np.newaxis, 2]
y = diabetes.target

# Split into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
print("Intercept:", model.intercept_)
print("Coefficient:", model.coef_[0])
print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
print("R² Score:", r2_score(y_test, y_pred))

# Sort values for a clean line plot
sorted_idx = np.argsort(X_test[:, 0])
X_test_sorted = X_test[sorted_idx]
y_pred_sorted = y_pred[sorted_idx]

# Plot scatter and best-fit line
plt.figure(figsize=(7,5))
plt.scatter(X_test, y_test, color='blue', label='Actual data')
```

```
plt.plot(X_test_sorted, y_pred_sorted, color='red', linewidth=2, label='Best-fit line')
plt.xlabel("BMI Feature")
plt.ylabel("Disease Progression")
plt.title("Linear Regression - Best Fit Line (Diabetes Dataset)")
plt.legend()
plt.show()
```

```
Intercept: 152.00335421448167
Coefficient: 998.5776891375593
Mean Squared Error: 4061.8259284949268
R² Score: 0.23335039815872138
```