In [59]:
```
from pyDOE2 import bbdesign
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Design:

- A --> WingLength: -1: 6.5 cm, 0: 8 cm, 1: 9.5 cm
- B --> BodyLength: -1: 6.5 cm, 0: 8 cm, 1: 9.5 cm
- C --> BodyWidth: -1: 4 cm, 0: 5 cm, 1: 6 cm
- D --> PaperClip: -1: no (fixed)
- E --> Tape: -1:no (fixed)

Box-Behnken

In [60]:
```
# Define factor levels
three_level_factors = {
    "A": [-1, 0, 1], # WingLength
    "B": [-1, 0, 1], # BodyLength
    "C": [-1, 0, 1] # BodyWidth
}

two_level_factors = {
    "D": -1,  # -1: No PaperClip
    "E":-1       # -1: No Tape
}
```

In [61]:
```
bbd_matrix = bbdesign(len(three_level_factors), center=3)
```

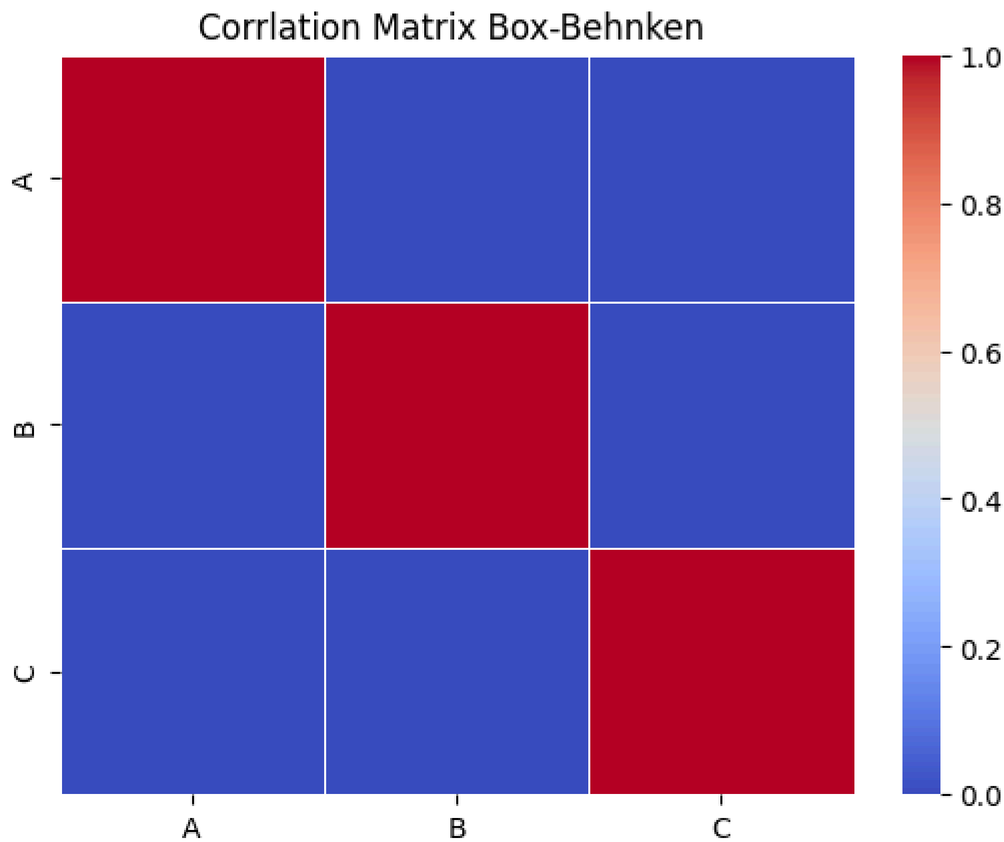
In [62]:
```
bbd_df = pd.DataFrame(bbd_matrix, columns=three_level_factors.keys())

print(bbd_df)
```

```
      A    B    C
0  -1.0 -1.0  0.0
1   1.0 -1.0  0.0
2  -1.0  1.0  0.0
3   1.0  1.0  0.0
4  -1.0  0.0 -1.0
5   1.0  0.0 -1.0
6  -1.0  0.0  1.0
7   1.0  0.0  1.0
8   0.0 -1.0 -1.0
9   0.0  1.0 -1.0
10  0.0 -1.0  1.0
11  0.0  1.0  1.0
12  0.0  0.0  0.0
13  0.0  0.0  0.0
14  0.0  0.0  0.0
```

In [63]:
```
print(bbd_df.corr())
```

```
              A             B             C
A  1.000000e+00 -2.775558e-17 -1.001484e-33
B -2.775558e-17  1.000000e+00  0.000000e+00
C -1.001484e-33  0.000000e+00  1.000000e+00
```

In [64]:
```
sns.heatmap(bbd_df.corr(), annot=False, cmap="coolwarm",  linewidths=0.5)
plt.title("Corrlation Matrix Box-Behnken ")
plt.show()
```



Mapping Actual Values Experiment

In [65]:
```
actual_values = {
    "A": { -1: 6.5, 0: 8, 1: 9.5 },
    "B": { -1: 6.5, 0: 8, 1: 9.5 },
    "C": { -1: 4, 0: 5, 1: 6 }
}

for col in three_level_factors:
    bbd_df[col] = bbd_df[col].map(actual_values[col])

print(bbd_df)
```

```
      A    B  C
0   6.5  6.5  5
1   9.5  6.5  5
2   6.5  9.5  5
3   9.5  9.5  5
4   6.5  8.0  4
5   9.5  8.0  4
6   6.5  8.0  6
7   9.5  8.0  6
8   8.0  6.5  4
9   8.0  9.5  4
10  8.0  6.5  6
11  8.0  9.5  6
12  8.0  8.0  5
13  8.0  8.0  5
14  8.0  8.0  5
```

In [66]:
```
# Radomize the order of the rows
random_order = bbd_df.sample(frac = 1, random_state=123).reset_index(drop=True)
print(random_order)
```

```
      A    B  C
0   9.5  8.0  6
1   8.0  6.5  6
2   6.5  8.0  4
3   6.5  6.5  5
4   9.5  8.0  4
5   8.0  9.5  4
6   8.0  6.5  4
7   8.0  9.5  6
8   9.5  9.5  5
9   9.5  6.5  5
10  6.5  8.0  6
11  8.0  8.0  5
12  6.5  9.5  5
13  8.0  8.0  5
14  8.0  8.0  5
```

Analysis Summary

In [67]:
```
results_df = pd.read_csv("Response_surf.csv")
results_df.head()
```

Out[67]:

	NID	Date	Time	DropNumber	HelicopterID	WingLength	BodyLength	BodyWic
0	da703145	2025-03-02	1500	1	H8	9.5	8.0	
1	da703145	2025-03-02	1501	2	H11	8.0	6.5	
2	da703145	2025-03-02	1502	3	H5	6.5	8.0	
3	da703145	2025-03-02	1503	4	H1	6.5	6.5	
4	da703145	2025-03-02	1504	5	H6	9.5	8.0	

In [68]:
```
# Change tape and paper clip values to -1

results_df['PaperClip'] = results_df['PaperClip'].map({'n': -1, 'y': 1})
results_df['Tape'] = results_df['Tape'].map({'n': -1, 'y': 1})
```

In [69]:
```
print(results_df.head())
```

```
        NID        Date  Time  DropNumber HelicopterID  WingLength  \
0  da703145  2025-03-02  1500           1           H8         9.5
1  da703145  2025-03-02  1501           2          H11         8.0
2  da703145  2025-03-02  1502           3           H5         6.5
3  da703145  2025-03-02  1503           4           H1         6.5
4  da703145  2025-03-02  1504           5           H6         9.5

   BodyLength  BodyWidth  PaperClip  Tape  DropHeight  FlightTime Notes
0         8.0          6         -1    -1           2        1.71  Home
1         6.5          6         -1    -1           2        1.76  Home
2         8.0          4         -1    -1           2        1.34  Home
3         6.5          5         -1    -1           2        1.70  Home
4         8.0          4         -1    -1           2        1.62  Home
```

In [70]:
```
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression

X = results_df[['WingLength', 'BodyLength', 'BodyWidth']]
y = results_df['FlightTime']
```

In [71]:
```
poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(X)

model = LinearRegression().fit(X_poly, y)

feature_names = poly.get_feature_names_out(['WingLength', 'BodyLength', 'BodyWidth'
coefficients = model.coef_

summary_df = pd.DataFrame({
    'Term': feature_names,
```

```
    'Coefficient': coefficients
})
```

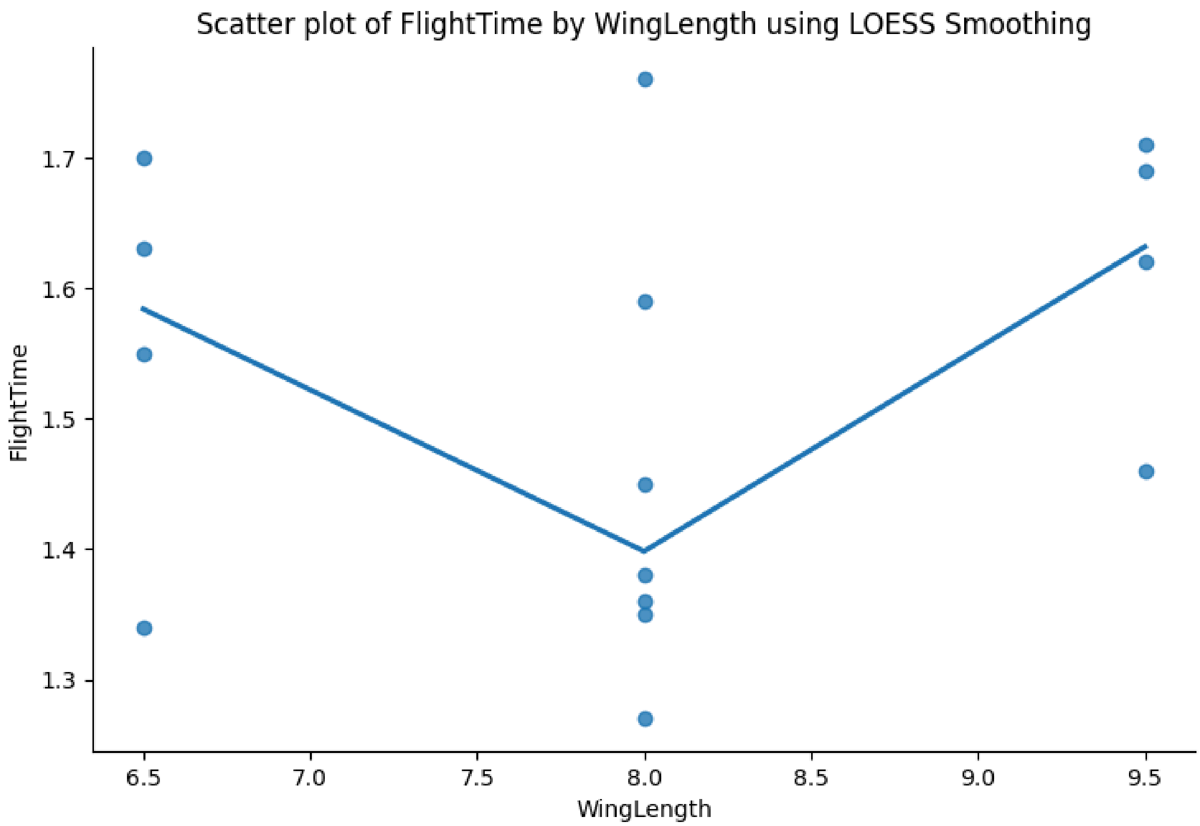
In [72]:
```
print(summary_df)
```

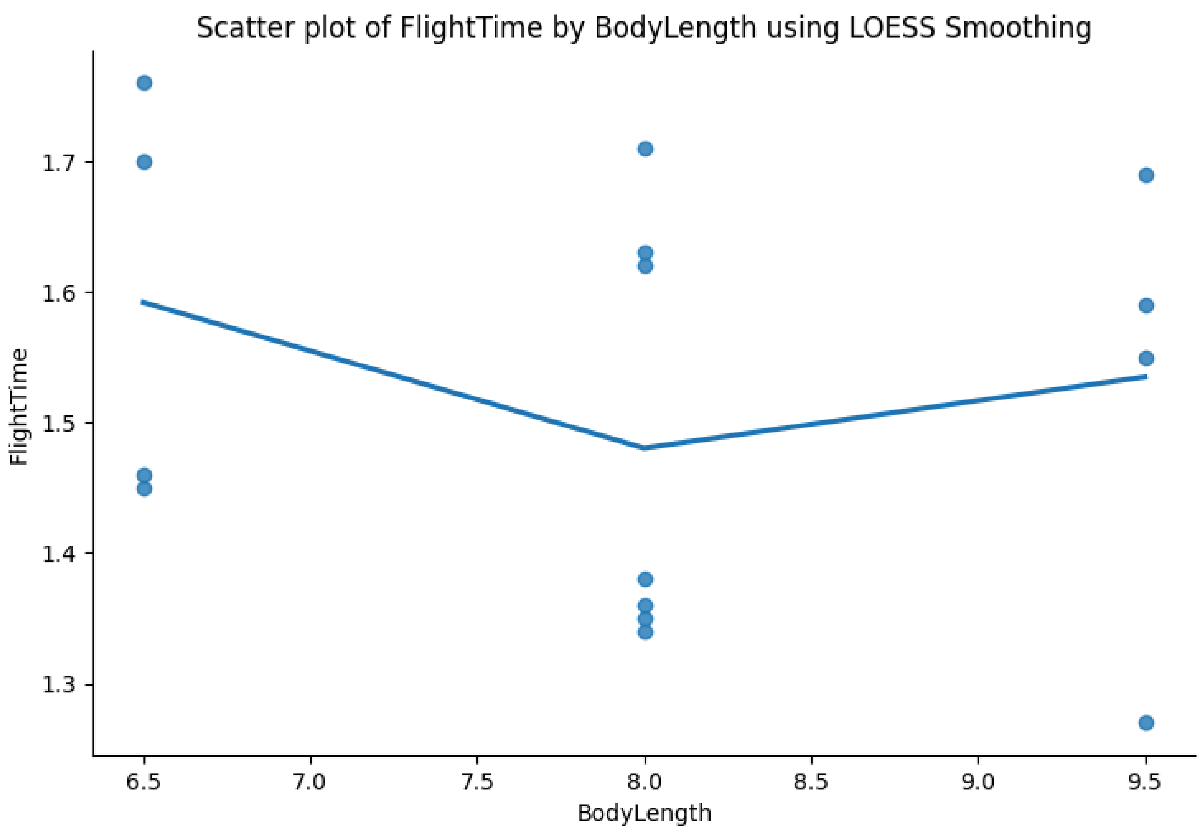
```
                      Term  Coefficient
0                        1     0.000000
1               WingLength    -1.195370
2               BodyLength    -0.472315
3                BodyWidth     0.507083
4             WingLength^2     0.065370
5    WingLength BodyLength     0.042222
6     WingLength BodyWidth    -0.033333
7             BodyLength^2     0.039815
8     BodyLength BodyWidth    -0.105000
9              BodyWidth^2     0.064583
```

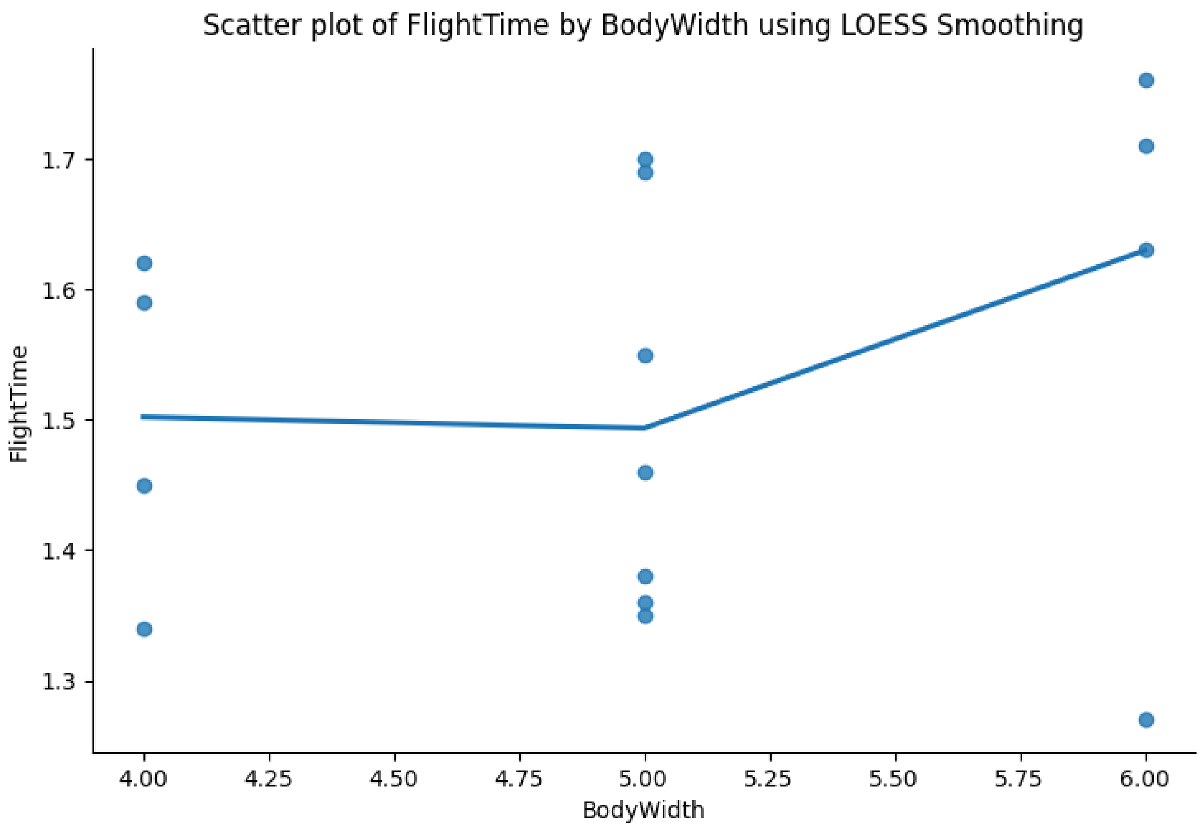
- Based on the **Linear terms** and their coefficients (WingLenght, BodyLength and BodyWidth) It suggest that max value of wings decreases the amount of flight time, the same effect happends with BodyWith where a long body or max value decreases the flight time but for Bodywith it suggest that wider body makes the helicopter stay on the air longer. Therefore, as the value increases for wing and body length it decreases the flight time but larger values of body with help the helicopter stay longer.
- Now, for the **squared terms** and their coefficients (^2) for Winglength it seems that a certain point larger wings having might actually increase the flight time. The same thing seems to happen for BodyLength where at a certain point a long body can increase the flight time. For BodyWidth, wider body seems to maintain the helicopter longer in the air increasing the flight time but it becomes less signficant.
- For **interaction terms** and their coefficients. The combination of having long wings and long body it leads to an increase of flight time. The combination of longer wings and wider body slightly reduced the flight time. Finally, when BodyLength and BodyWidth increase together, the flight time decreases which means that longer and wider bodies shorter flight time.

In [73]:
```
# Visualize Scatter Plot
for j in X:
    # Create scatter plot with LOESS smoothing
    sns.lmplot(
        data=results_df,
        x=j,  # Independent variables
        y='FlightTime',  # Dependent variable
        lowess=True,  # Add LOESS smoothing
        height=5,  # Set height of the plot
        aspect=1.5  # Set aspect ratio
    )
    plt.title(f'Scatter plot of FlightTime by {j} using LOESS Smoothing')
    plt.xlabel(j)
```

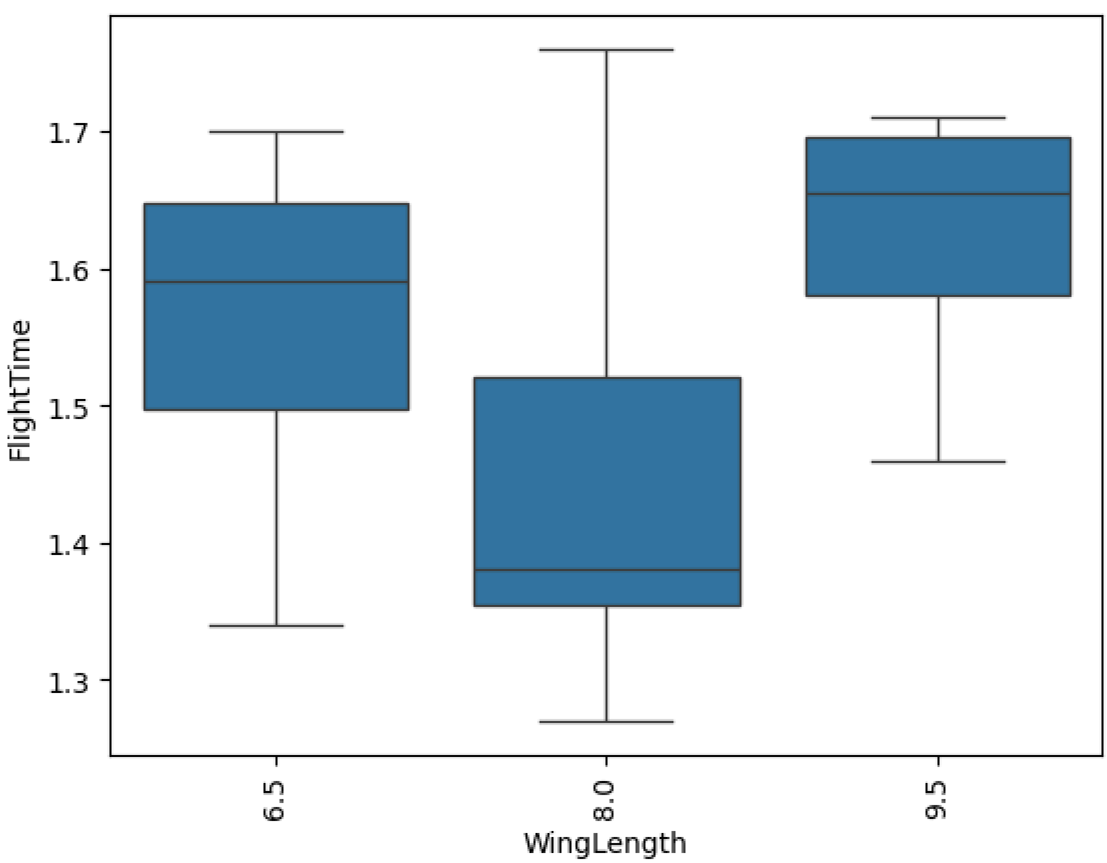
```
plt.ylabel('FlightTime')
plt.show()  # Display the plot
```



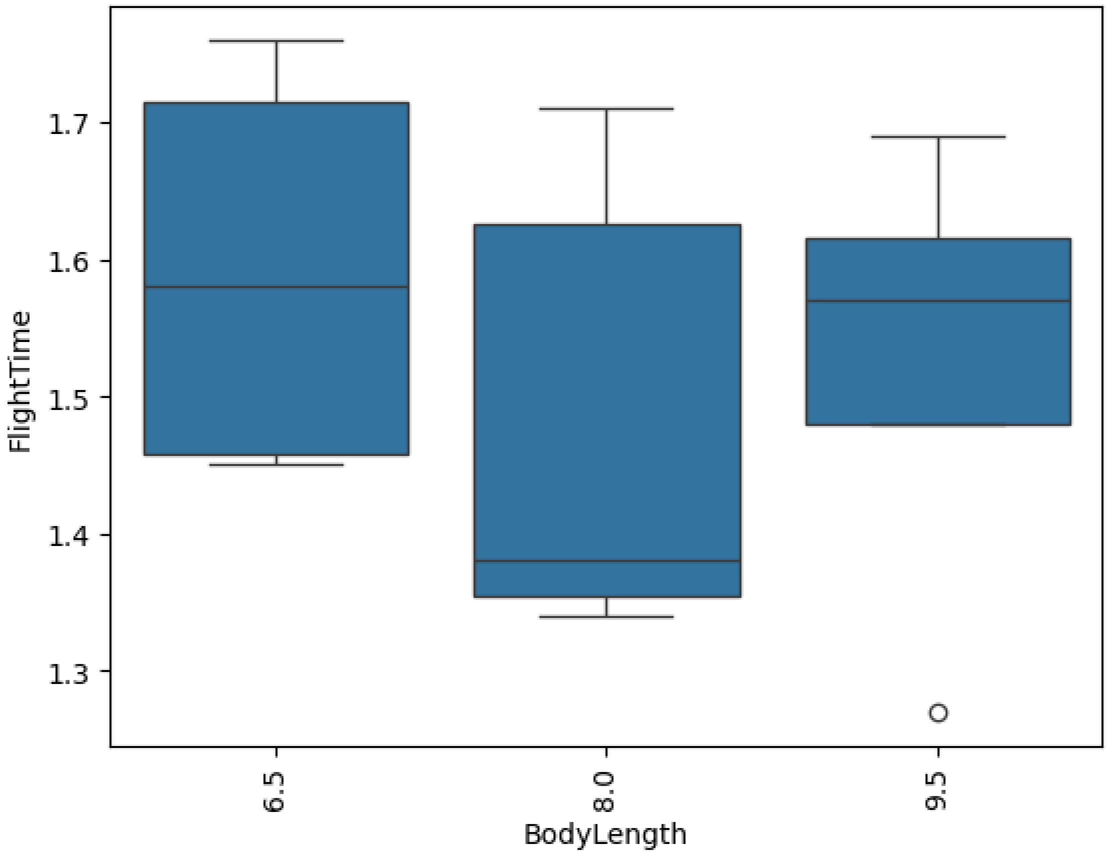




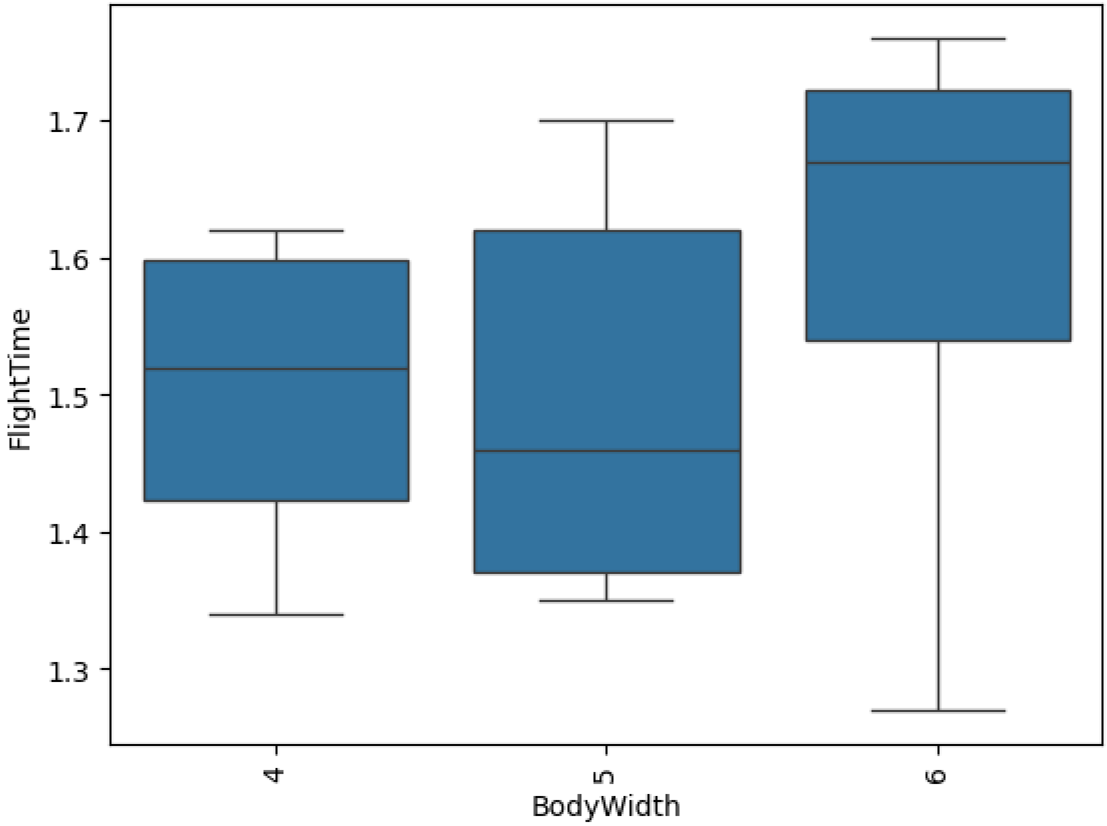
In [74]:
```
sns.boxplot(x='WingLength', y='FlightTime', data=results_df)
plt.xticks(rotation=90)
plt.show()
```



In [75]:
```
sns.boxplot(x='BodyLength', y='FlightTime', data=results_df)
plt.xticks(rotation=90)
plt.show()
```



In [76]:
```
sns.boxplot(x='BodyWidth', y='FlightTime', data=results_df)
plt.xticks(rotation=90)
plt.show()
```


FlightTime
1.7
1.6
1.5
1.4
1.3
4
5
6
BodyWidth