In [40]:

```
# Importing Required Libraries
import numpy as ns
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,classification_report
```

In [41]:

```
# Loading the Dataset
df = pd.read_csv("D:\\Dataset\\PENGUINS.csv")
df
```

Out[41]:

	Unnamed: 0	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	1	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male
1	2	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female
2	3	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female
3	4	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	5	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female
...	...	...	...	...	...	...	...	...
339	340	Chinstrap	Dream	55.8	19.8	207.0	4000.0	male
340	341	Chinstrap	Dream	43.5	18.1	202.0	3400.0	female
341	342	Chinstrap	Dream	49.6	18.2	193.0	3775.0	male
342	343	Chinstrap	Dream	50.8	19.0	210.0	4100.0	male
343	344	Chinstrap	Dream	50.2	18.7	198.0	3775.0	female

344 rows × 8 columns

In [20]:

```
# Data Cleaning using Pandas
df.rename(columns={'Unnamed: 0' : 'S.NO'},inplace=True)
df
```

Out[20]:

	S.NO	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	1	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male
1	2	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female
2	3	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female
3	4	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	5	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female
...	...	...	...	...	...	...	...	...
339	340	Chinstrap	Dream	55.8	19.8	207.0	4000.0	male
340	341	Chinstrap	Dream	43.5	18.1	202.0	3400.0	female
341	342	Chinstrap	Dream	49.6	18.2	193.0	3775.0	male
342	343	Chinstrap	Dream	50.8	19.0	210.0	4100.0	male
343	344	Chinstrap	Dream	50.2	18.7	198.0	3775.0	female

344 rows × 8 columns

In [42]:

```
df.head(9)
```

Out[42]:

	Unnamed: 0	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	1	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male
1	2	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female
2	3	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female
3	4	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	5	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female
5	6	Adelie	Torgersen	39.3	20.6	190.0	3650.0	male
6	7	Adelie	Torgersen	38.9	17.8	181.0	3625.0	female
7	8	Adelie	Torgersen	39.2	19.6	195.0	4675.0	male
8	9	Adelie	Torgersen	34.1	18.1	193.0	3475.0	NaN

In [43]:

```
df.tail(12)
```

Out[43]:

	Unnamed: 0	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
332	333	Chinstrap	Dream	45.2	16.6	191.0	3250.0	female
333	334	Chinstrap	Dream	49.3	19.9	203.0	4050.0	male
334	335	Chinstrap	Dream	50.2	18.8	202.0	3800.0	male
335	336	Chinstrap	Dream	45.6	19.4	194.0	3525.0	female
336	337	Chinstrap	Dream	51.9	19.5	206.0	3950.0	male
337	338	Chinstrap	Dream	46.8	16.5	189.0	3650.0	female
338	339	Chinstrap	Dream	45.7	17.0	195.0	3650.0	female
339	340	Chinstrap	Dream	55.8	19.8	207.0	4000.0	male
340	341	Chinstrap	Dream	43.5	18.1	202.0	3400.0	female
341	342	Chinstrap	Dream	49.6	18.2	193.0	3775.0	male
342	343	Chinstrap	Dream	50.8	19.0	210.0	4100.0	male
343	344	Chinstrap	Dream	50.2	18.7	198.0	3775.0	female

In [44]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Unnamed: 0         344 non-null    int64  
 1   species            344 non-null    object 
 2   island             344 non-null    object 
 3   bill_length_mm     342 non-null    float64
 4   bill_depth_mm      342 non-null    float64
 5   flipper_length_mm  342 non-null    float64
 6   body_mass_g        342 non-null    float64
 7   sex                333 non-null    object 
dtypes: float64(4), int64(1), object(3)
memory usage: 21.6+ KB
```

In [45]:
```
df.isna().sum()
```

Out[45]:
```
Unnamed: 0           0
species              0
island               0
bill_length_mm       2
bill_depth_mm        2
flipper_length_mm    2
body_mass_g          2
sex                 11
dtype: int64
```

In [71]:
```
# Drop rows with missing values in relevant columns
df = df.dropna(subset=['flipper_length_mm', 'body_mass_g', 'bill_length_mm', 'bill_depth_mm', 'sex']
```

In [47]:
```
df.shape
```

Out[47]:
```
(333, 8)
```

In [48]:
```
df.columns
```

Out[48]:
```
Index(['Unnamed: 0', 'species', 'island', 'bill_length_mm', 'bill_depth_mm',
       'flipper_length_mm', 'body_mass_g', 'sex'],
      dtype='object')
```

In [49]:
```
df.size
```

Out[49]:
```
2664
```

In [50]:
```
df[df.index==99]
```

Out[50]:

	Unnamed: 0	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
99	100	Adelie	Dream	43.2	18.5	192.0	4100.0	male

In [51]:
```
df[df.index.isin(range(4,19))]
```

Out[51]:

	Unnamed: 0	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
4	5	Adelie	Torgersen	36.7	19.3	193.0	3450.0	female
5	6	Adelie	Torgersen	39.3	20.6	190.0	3650.0	male
6	7	Adelie	Torgersen	38.9	17.8	181.0	3625.0	female
7	8	Adelie	Torgersen	39.2	19.6	195.0	4675.0	male
12	13	Adelie	Torgersen	41.1	17.6	182.0	3200.0	female
13	14	Adelie	Torgersen	38.6	21.2	191.0	3800.0	male
14	15	Adelie	Torgersen	34.6	21.1	198.0	4400.0	male
15	16	Adelie	Torgersen	36.6	17.8	185.0	3700.0	female
16	17	Adelie	Torgersen	38.7	19.0	195.0	3450.0	female
17	18	Adelie	Torgersen	42.5	20.7	197.0	4500.0	male
18	19	Adelie	Torgersen	34.4	18.4	184.0	3325.0	female

In [52]:

```
df.loc[100]
```

Out[52]:

```
Unnamed: 0                 101
species                 Adelie
island                  Biscoe
bill_length_mm            35.0
bill_depth_mm             17.9
flipper_length_mm        192.0
body_mass_g             3725.0
sex                     female
Name: 100, dtype: object
```

In [53]:

```
df.describe()
```

Out[53]:

	Unnamed: 0	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
count	333.000000	333.000000	333.000000	333.000000	333.000000
mean	174.324324	43.992793	17.164865	200.966967	4207.057057
std	98.386547	5.468668	1.969235	14.015765	805.215802
min	1.000000	32.100000	13.100000	172.000000	2700.000000
25%	90.000000	39.500000	15.600000	190.000000	3550.000000
50%	173.000000	44.500000	17.300000	197.000000	4050.000000
75%	259.000000	48.600000	18.700000	213.000000	4775.000000
max	344.000000	59.600000	21.500000	231.000000	6300.000000

In [54]:

```
df.dtypes
```

Out[54]:

```
Unnamed: 0             int64
species               object
island                object
bill_length_mm       float64
bill_depth_mm        float64
flipper_length_mm    float64
body_mass_g          float64
sex                   object
dtype: object
```

In [56]:

```
df.drop(4)
```

Out[56]:

	Unnamed: 0	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	1	Adelie	Torgersen	39.1	18.7	181.0	3750.0	male
1	2	Adelie	Torgersen	39.5	17.4	186.0	3800.0	female
2	3	Adelie	Torgersen	40.3	18.0	195.0	3250.0	female
5	6	Adelie	Torgersen	39.3	20.6	190.0	3650.0	male
6	7	Adelie	Torgersen	38.9	17.8	181.0	3625.0	female
...	...	...	...	...	...	...	...	...
339	340	Chinstrap	Dream	55.8	19.8	207.0	4000.0	male
340	341	Chinstrap	Dream	43.5	18.1	202.0	3400.0	female
341	342	Chinstrap	Dream	49.6	18.2	193.0	3775.0	male
342	343	Chinstrap	Dream	50.8	19.0	210.0	4100.0	male
343	344	Chinstrap	Dream	50.2	18.7	198.0	3775.0	female

332 rows × 8 columns

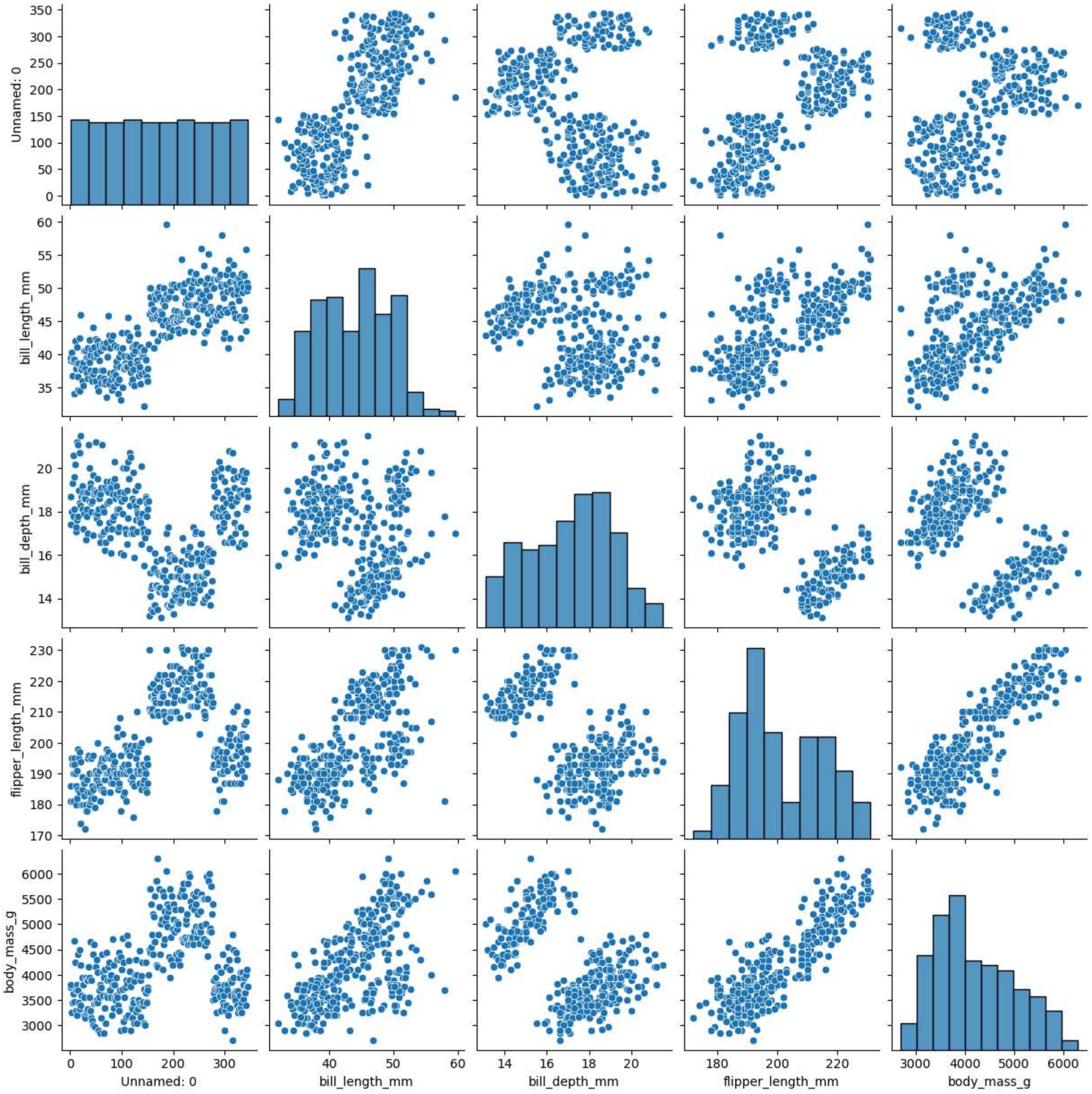
In [57]:

```
# Data Visualization using Matplotlib & Seaborn
import pandas as pd
import seaborn as sns
df = pd.read_csv("D:\\Dataset\\PENGUINS.csv")
import seaborn as sns
sns.pairplot(df)
```

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

Out[57]:

```
<seaborn.axisgrid.PairGrid at 0x21d491de610>
```

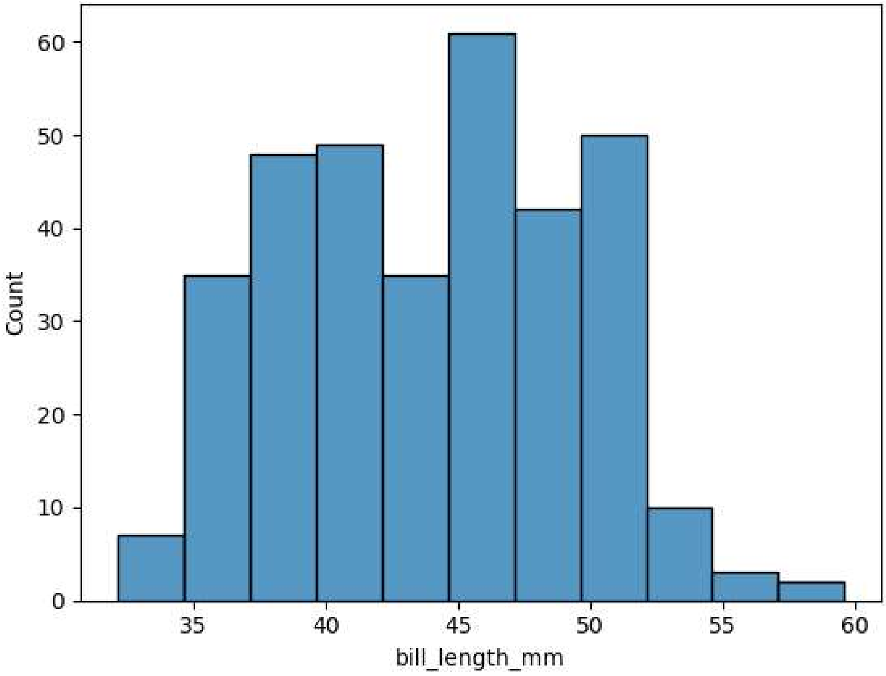


In [58]:

```
sns.histplot(df["bill_length_mm"])
```

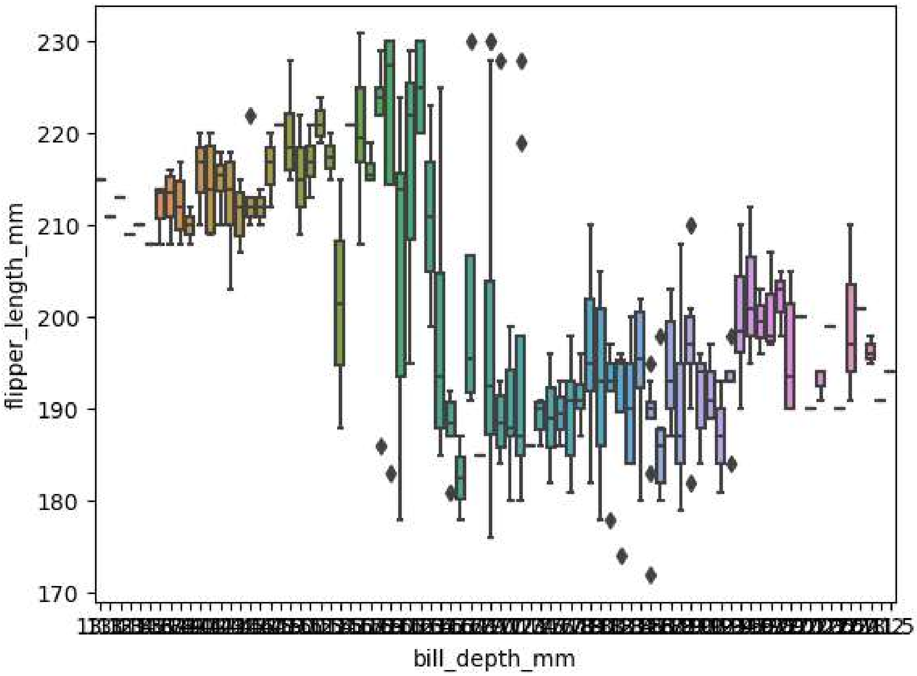
Out[58]:

```
<Axes: xlabel='bill_length_mm', ylabel='Count'>
```



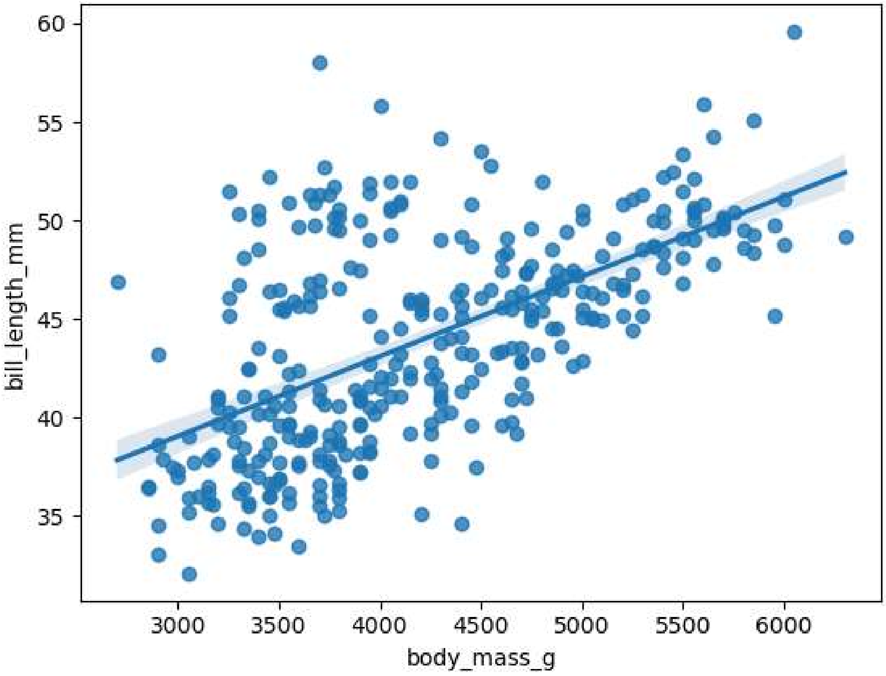
In [59]:

```
import matplotlib.pyplot as plt
sns.boxplot(x="bill_depth_mm",y="flipper_length_mm",data=df)
plt.show()
```



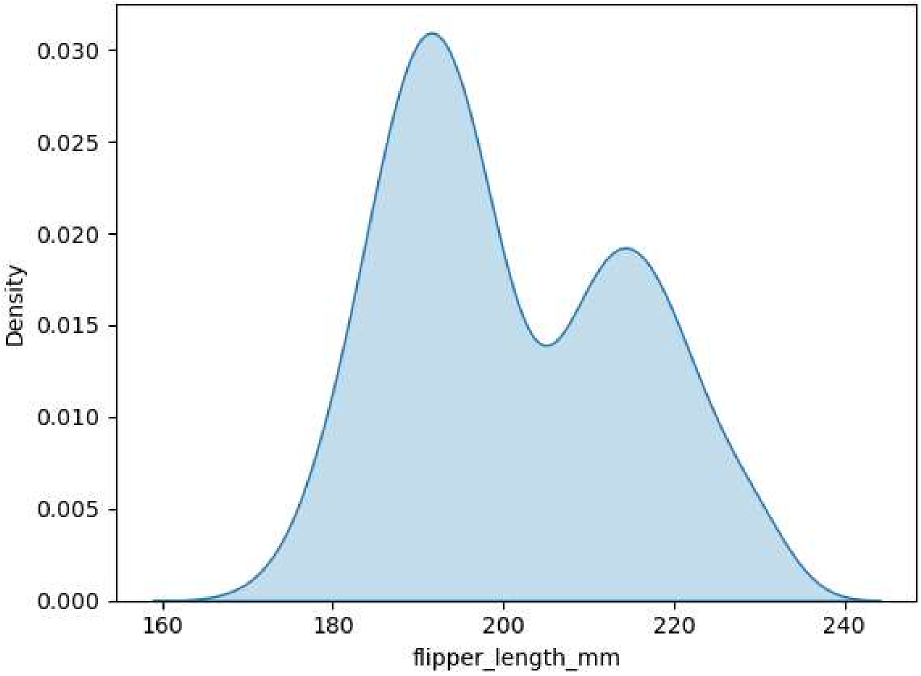
In [60]:

```
sns.regplot(x="body_mass_g",y="bill_length_mm",data=df)
plt.show()
```



In [61]:

```
sns.kdeplot(df["flipper_length_mm"],fill=True)
plt.show()
```



In [85]:

```
features = ['flipper_length_mm', 'body_mass_g', 'bill_length_mm', 'bill_depth_mm']
X = df[features]
y = df['species']
```

In [86]:

```
# Train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=42)
```

In [87]:

```
# Standardize the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [78]:

```
# Logistic Regression
log_reg = LogisticRegression(max_iter=1000, random_state=42)
log_reg.fit(X_train_scaled, y_train)
y_pred_log_reg = log_reg.predict(X_test_scaled)
print("Logistic Regression")
print("Accuracy:", accuracy_score(y_test, y_pred_log_reg))
print(classification_report(y_test, y_pred_log_reg))
```

```
Logistic Regression
Accuracy: 1.0
              precision    recall  f1-score   support

      Adelie       1.00      1.00      1.00        31
   Chinstrap       1.00      1.00      1.00        18
      Gentoo       1.00      1.00      1.00        18

    accuracy                           1.00        67
   macro avg       1.00      1.00      1.00        67
weighted avg       1.00      1.00      1.00        67
```

In [79]:

```
# Decision Tree
decision_tree = DecisionTreeClassifier(random_state=42)
decision_tree.fit(X_train, y_train)
y_pred_decision_tree = decision_tree.predict(X_test)
print("Decision Tree")
print("Accuracy:", accuracy_score(y_test, y_pred_decision_tree))
print(classification_report(y_test, y_pred_decision_tree))
```

```
Decision Tree
Accuracy: 1.0
              precision    recall  f1-score   support

      Adelie       1.00      1.00      1.00        31
   Chinstrap       1.00      1.00      1.00        18
      Gentoo       1.00      1.00      1.00        18

    accuracy                           1.00        67
   macro avg       1.00      1.00      1.00        67
weighted avg       1.00      1.00      1.00        67
```

In [80]:

```
# Random Forest
random_forest = RandomForestClassifier(random_state=42)
random_forest.fit(X_train, y_train)
y_pred_random_forest = random_forest.predict(X_test)
print("Random Forest")
print("Accuracy:", accuracy_score(y_test, y_pred_random_forest))
print(classification_report(y_test, y_pred_random_forest))
```

```
Random Forest
Accuracy: 1.0
              precision    recall  f1-score   support

      Adelie       1.00      1.00      1.00        31
   Chinstrap       1.00      1.00      1.00        18
      Gentoo       1.00      1.00      1.00        18

    accuracy                           1.00        67
   macro avg       1.00      1.00      1.00        67
weighted avg       1.00      1.00      1.00        67
```

In [81]:

```
# Support Vector Machine (SVM)
svm = SVC(random_state=42)
svm.fit(X_train_scaled, y_train)
y_pred_svm = svm.predict(X_test_scaled)
print("Support Vector Machine (SVM)")
print("Accuracy:", accuracy_score(y_test, y_pred_svm))
print(classification_report(y_test, y_pred_svm))
```

```
Support Vector Machine (SVM)
Accuracy: 0.9701492537313433
              precision    recall  f1-score   support

      Adelie       0.94      1.00      0.97        31
   Chinstrap       1.00      0.89      0.94        18
      Gentoo       1.00      1.00      1.00        18

    accuracy                           0.97        67
   macro avg       0.98      0.96      0.97        67
weighted avg       0.97      0.97      0.97        67
```

In [ ]: