In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris
```

Loading DataSet

In [13]:
```python
df = pd.read_csv('Iris.csv')
```

In [14]:
```python
df
```

Out[14]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

In [15]:
```python
df['species'] = pd.Categorical.from_codes(data.target, data.target_names)
```

1. features and their types

In [16]:
```python
print("Dataset Features and Types:")
print(df.dtypes)
print("\nFirst few rows of the dataset:")
print(df.head())
```

```
Dataset Features and Types:
Id                  int64
SepalLengthCm     float64
SepalWidthCm      float64
PetalLengthCm     float64
PetalWidthCm      float64
Species            object
species          category
dtype: object

First few rows of the dataset:
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species  \
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa

  species
0  setosa
1  setosa
2  setosa
3  setosa
4  setosa
```

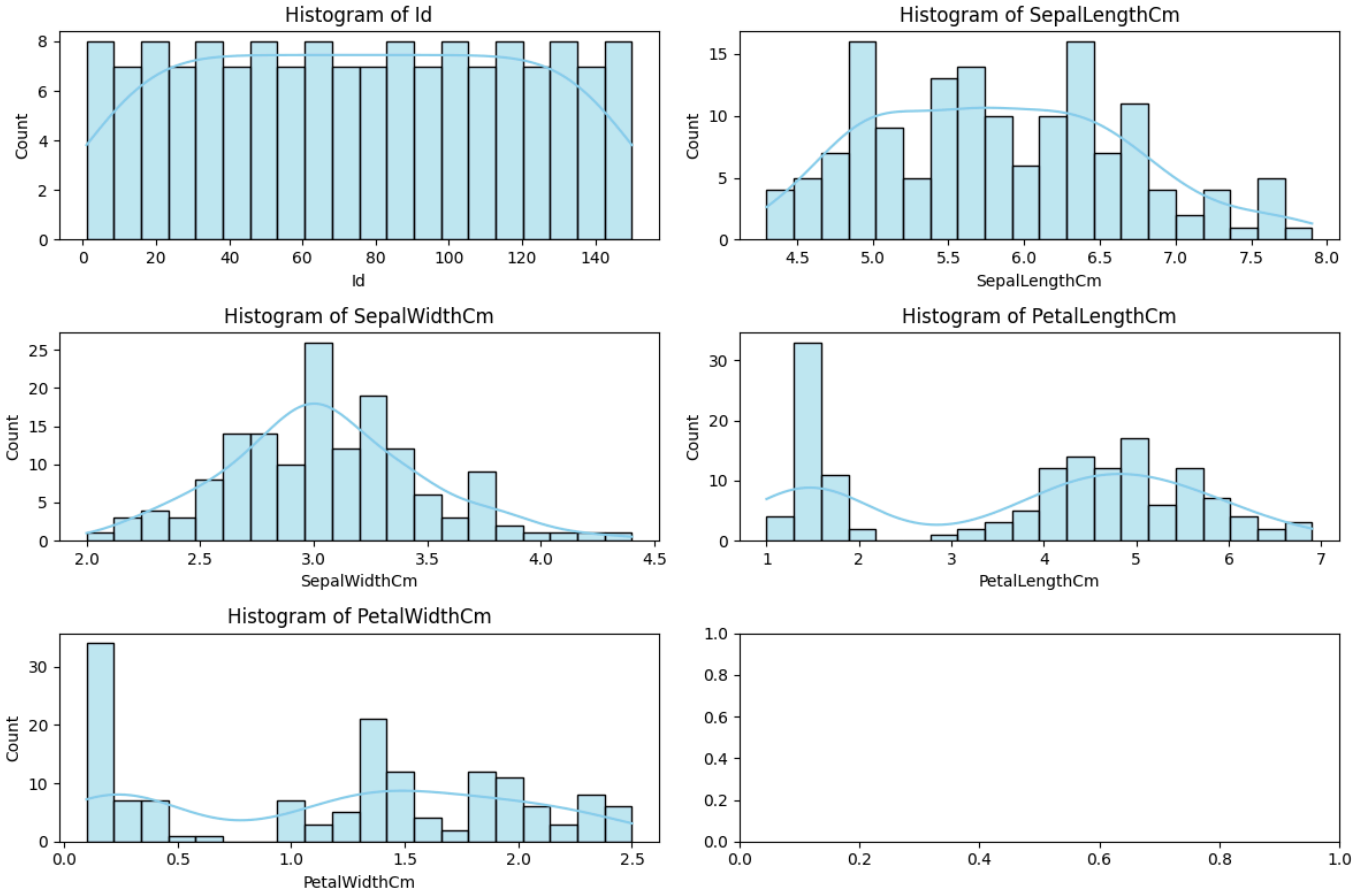
histograms for each numeric feature

In [20]:
```python
num_features = len(numeric_features)
rows = (num_features // 2) + (num_features % 2)  # Adjust rows dynamically

fig, axes = plt.subplots(rows, 2, figsize=(12, 8))  # Create a grid layout
axes = axes.flatten()  # Flatten the axes array to iterate over it easily

for i, col in enumerate(numeric_features):
    sns.histplot(df[col], kde=True, bins=20, color='skyblue', ax=axes[i])  # Assign each subp
    axes[i].set_title(f'Histogram of {col}')

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

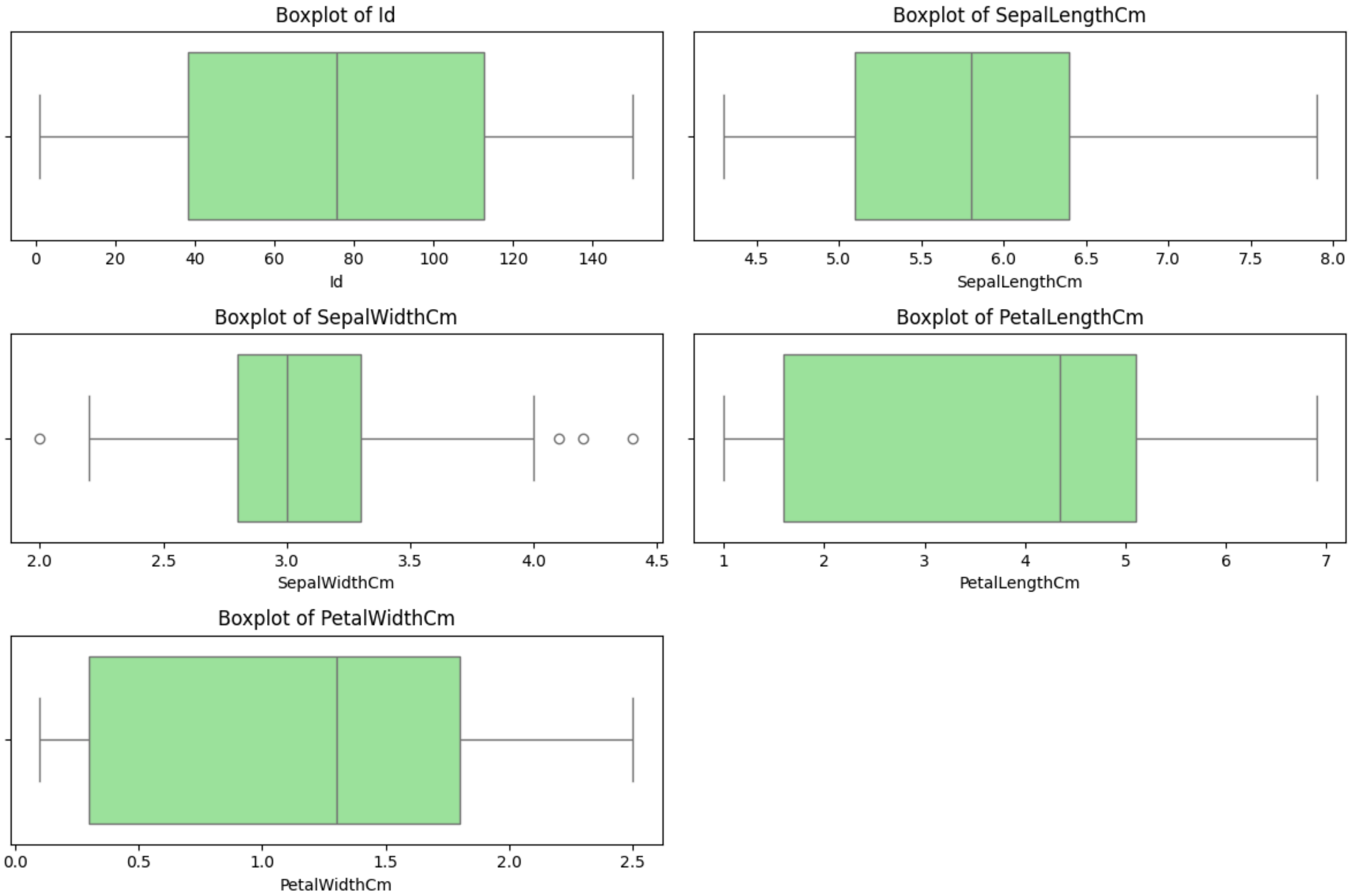


boxplots for each numeric feature

In [19]:
```
plt.figure(figsize=(12, 8))
num_features = len(numeric_features)
rows = (num_features // 2) + (num_features % 2)  # Adjust rows dynamically

for i, col in enumerate(numeric_features, 1):
    plt.subplot(rows, 2, i)  # Dynamically adjust the number of rows
    sns.boxplot(x=df[col], color='lightgreen')
    plt.title(f'Boxplot of {col}')

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



Outlier Detection and removal

In [22]:

```
def detect_outliers(df, feature):
    Q1 = df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[feature] < lower_bound) | (df[feature] > upper_bound)]

for col in numeric_features:
    outliers = detect_outliers(df, col)
    print(f"\nOutliers detected in {col}:")
    if not outliers.empty:
        print(outliers[[col]])
    else:
        print("No outliers detected.")


df_cleaned = df.copy()
for col in numeric_features:
    Q1 = df_cleaned[col].quantile(0.25)
    Q3 = df_cleaned[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    df_cleaned = df_cleaned[(df_cleaned[col] >= lower_bound) & (df_cleaned[col] <= upper_boun

print("\nShape before handling outliers:", df.shape)
print("Shape after handling outliers:", df_cleaned.shape)
```

```
Outliers detected in Id:
No outliers detected.

Outliers detected in SepalLengthCm:
No outliers detected.

Outliers detected in SepalWidthCm:
    SepalWidthCm
15           4.4
32           4.1
33           4.2
60           2.0

Outliers detected in PetalLengthCm:
No outliers detected.

Outliers detected in PetalWidthCm:
No outliers detected.

Shape before handling outliers: (150, 7)
Shape after handling outliers: (146, 7)
```