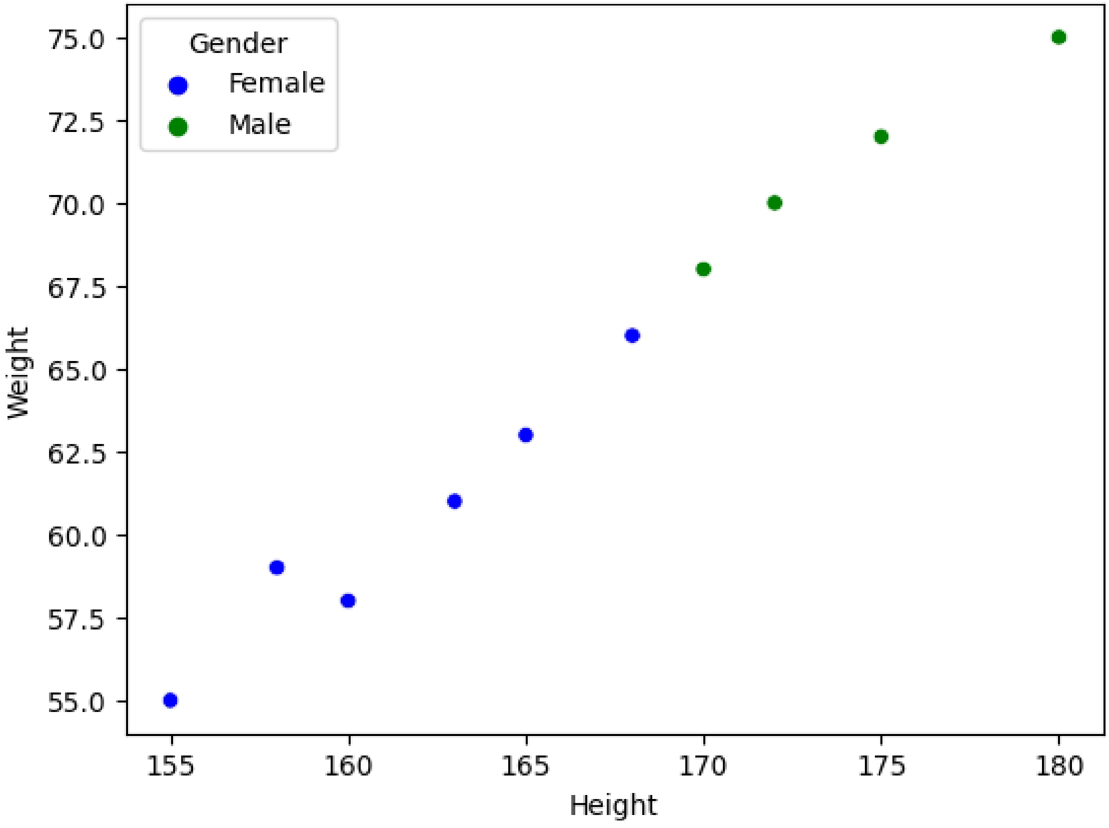
Seaborn is a Python data visualization library built on top of Matplotlib. It is designed to make data visualization more attractive and informative with minimal effort. Seaborn provides a higher-level interface to create a variety of statistical graphics, making it especially useful for data exploration, analysis, and presentation.

In [5]:

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Sample data
data = {
    'Height': [160, 165, 155, 172, 180, 158, 163, 170, 175, 168],
    'Weight': [58, 63, 55, 70, 75, 59, 61, 68, 72, 66],
    'Gender': ['Female', 'Female', 'Female', 'Male', 'Male', 'Female', 'Fer

}

df = pd.DataFrame(data)

palette_colors = {"Male": "green", "Female": "blue"}

sns.scatterplot(x='Height', y='Weight', data=df,hue="Gender",palette=palett

plt.show()
```



In [ ]:

```
sns.set_style()
sns.set_palette()
sns.set_context()
```

In [ ]:

```
g.fig.subtitle()
g.set_title()
```

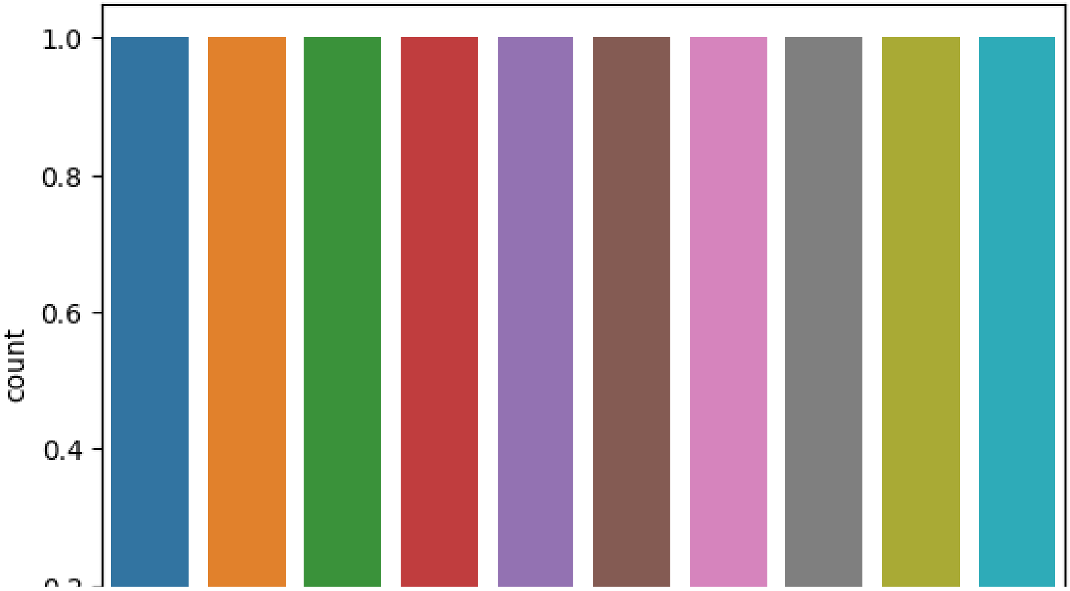
Count Plot

In [5]:

```
sns.countplot(x=weight)
```

Out[5]:

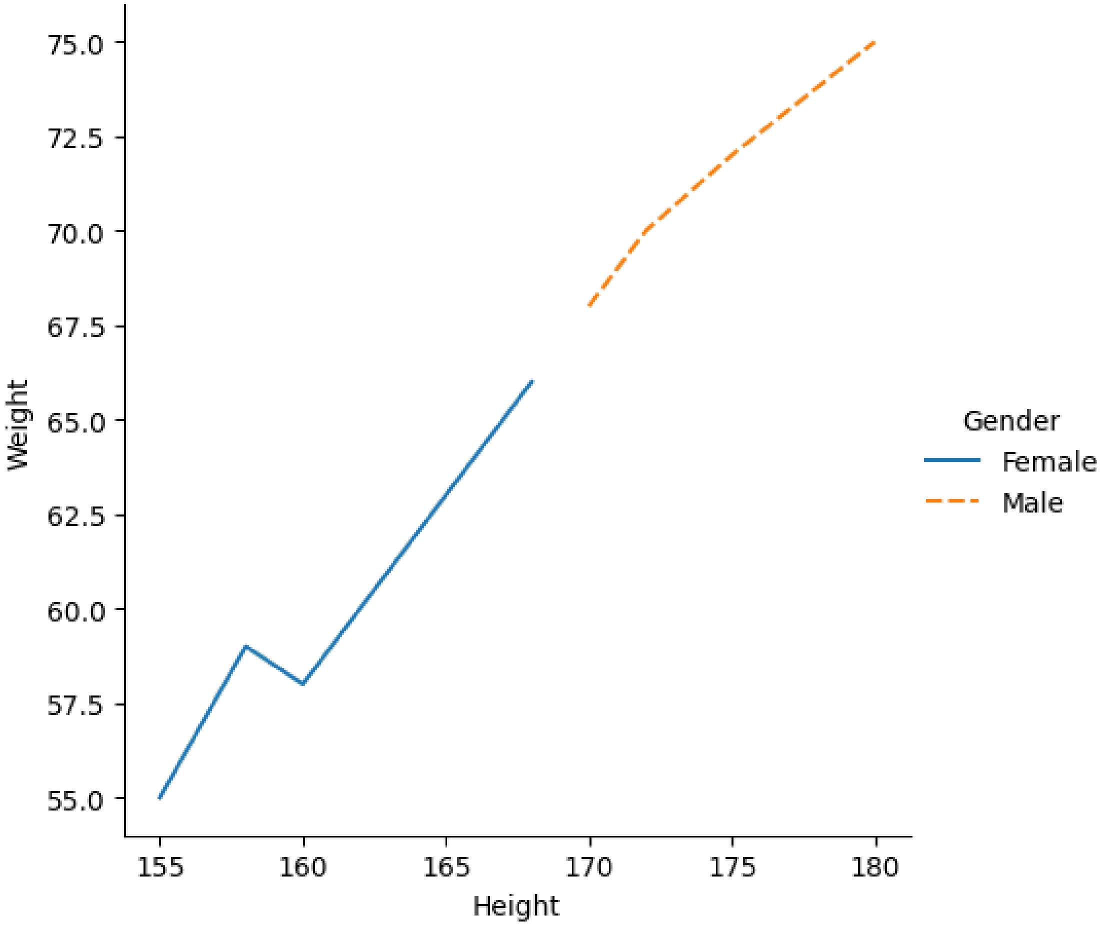
```
<Axes: ylabel='count'>
```



Line Plot

In [21]:
```
sns.relplot(x='Height', y='Weight', data=df,kind='line',errorbar='sd',hue='
```

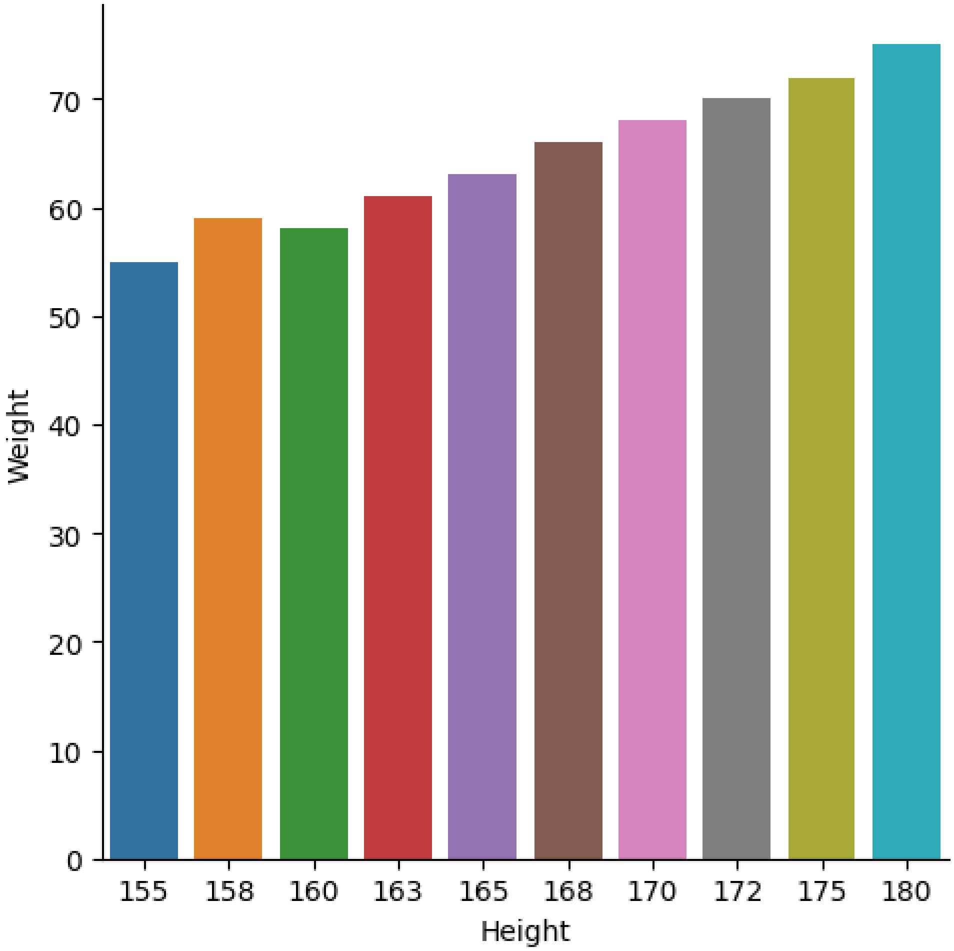
Out[21]:
```
<seaborn.axisgrid.FacetGrid at 0x19cff6268f0>
```



Bar Plot

```
In [22]: sns.catplot(x='Height', y='Weight', data=df,kind='bar')
```

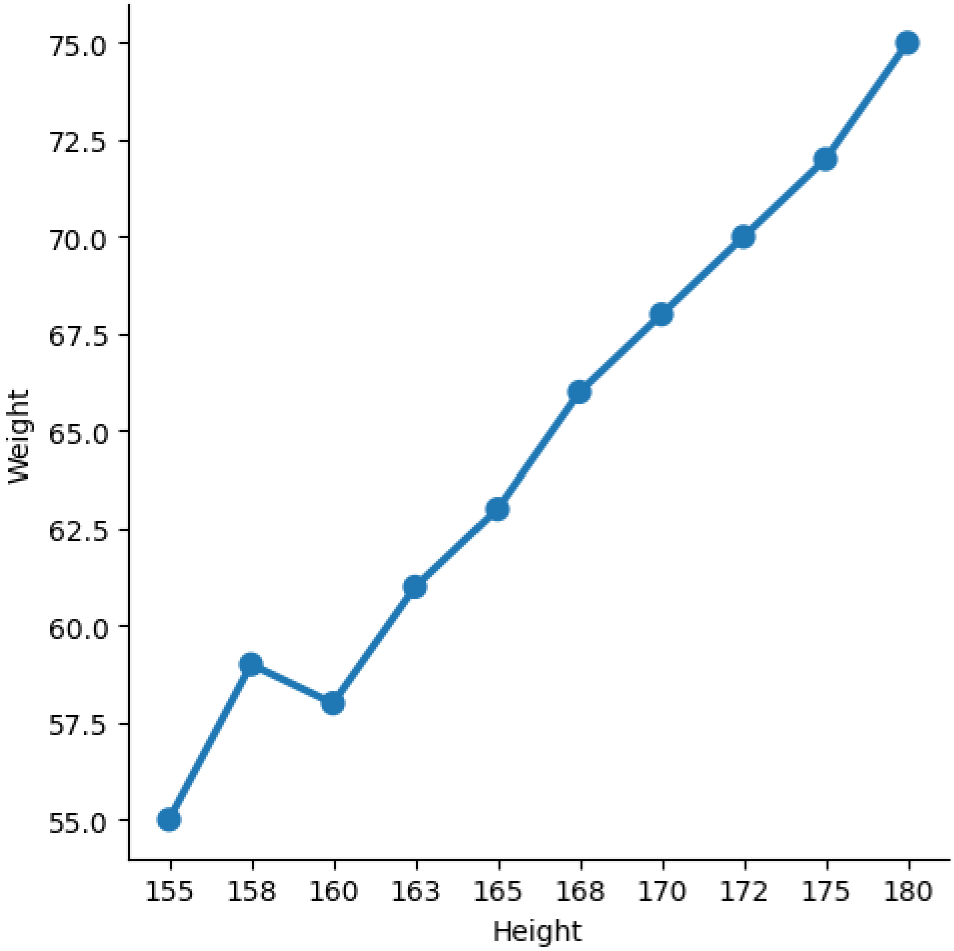
```
Out[22]: <seaborn.axisgrid.FacetGrid at 0x19cffc8eef0>
```



Point Plot

```
In [24]:  sns.catplot(x='Height', y='Weight', data=df,kind='point')
```

```
Out[24]: <seaborn.axisgrid.FacetGrid at 0x19c81086530>
```



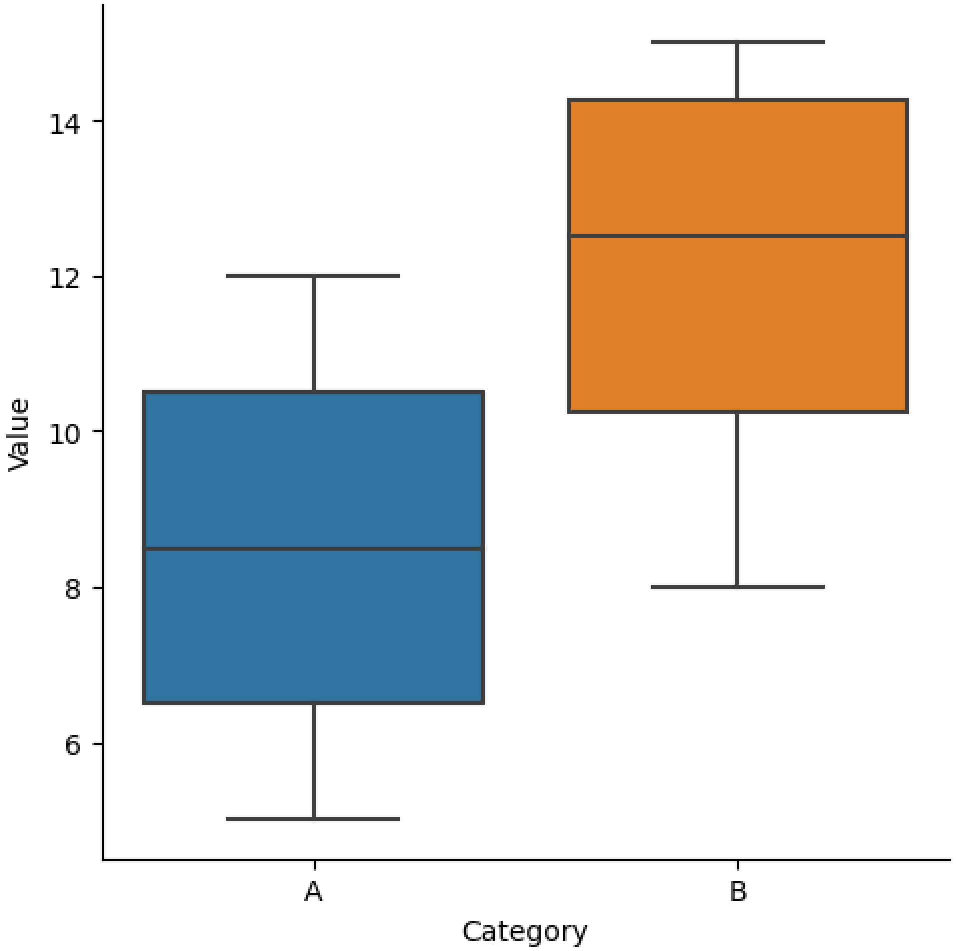
Box Plot

In [6]:

```
data1 = {
    'Category': ['A', 'B', 'A', 'B', 'A', 'B', 'A', 'B'],
    'Value': [5, 8, 12, 15, 10, 14, 7, 11]
}

df1=pd.DataFrame(data1)
sns.catplot(x='Category', y='Value', data=df1,kind='box')
```

Out[6]:

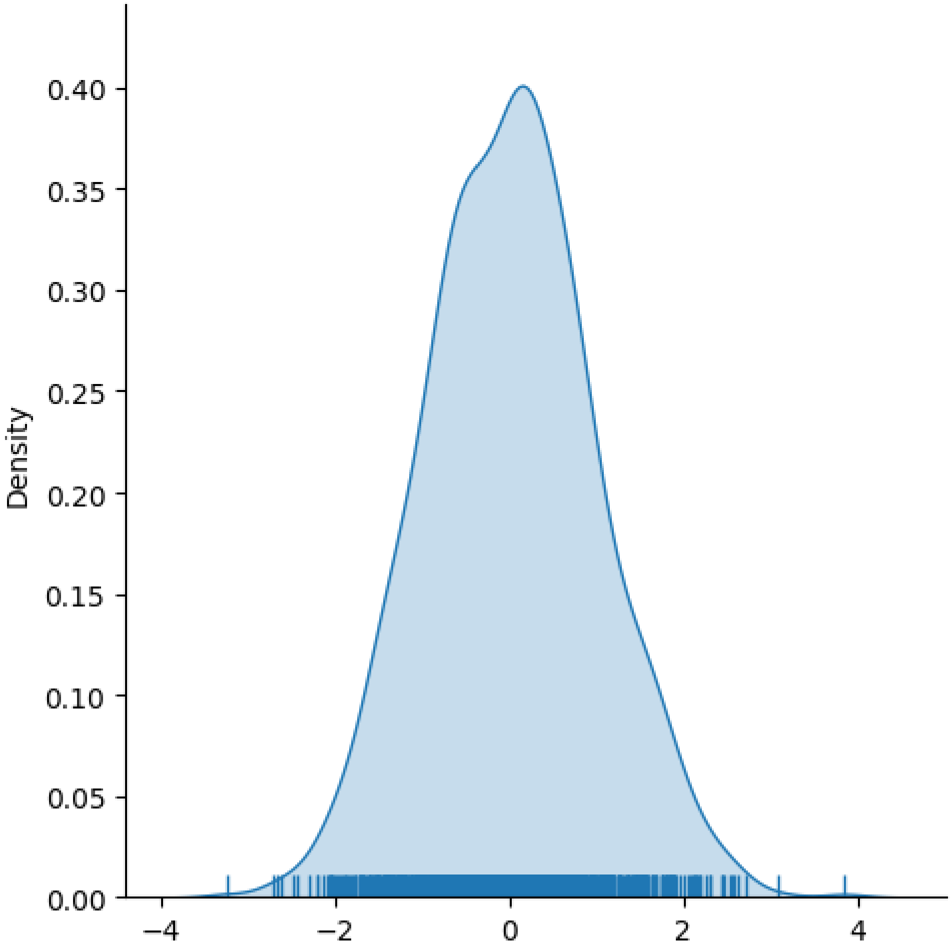
```
<seaborn.axisgrid.FacetGrid at 0x26b01b78e50>
```



Displot

In [10]:

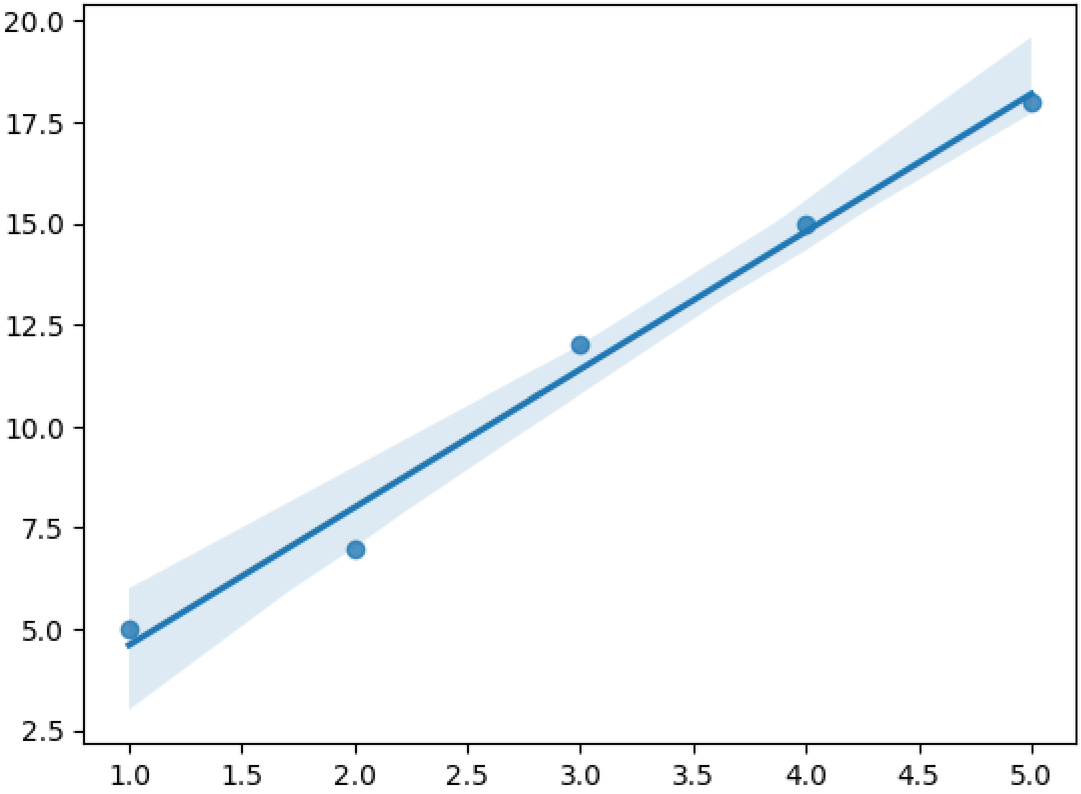
```
# Generate random data for demonstration
np.random.seed(42)
data = np.random.normal(loc=0, scale=1, size=1000)

# Create a displot of the generated data
sns.displot(data,
            kind='kde',   # Shaded KDE plot
            rug=True,      # Add a rug plot
            fill=True)     # Fill the area under the KDE plot

# Show the plot
plt.show()
```



Regression Plot

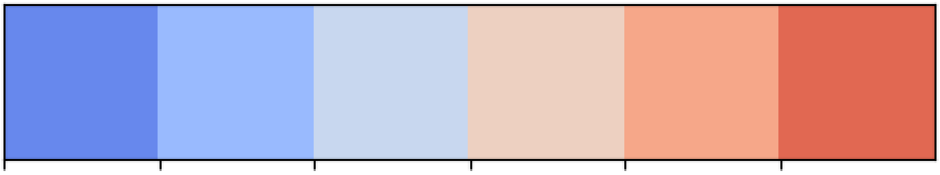
In [12]:

```
# Sample data
x = [1, 2, 3, 4, 5]
y = [5, 7, 12, 15, 18]

# Create a scatter plot with linear regression line
sns.regplot(x=x, y=y)

# Show the plot
plt.show()
```



In [13]:

```
# Create the Purples palette
sns.palplot(sns.color_palette("coolwarm", 6))
plt.show()
```



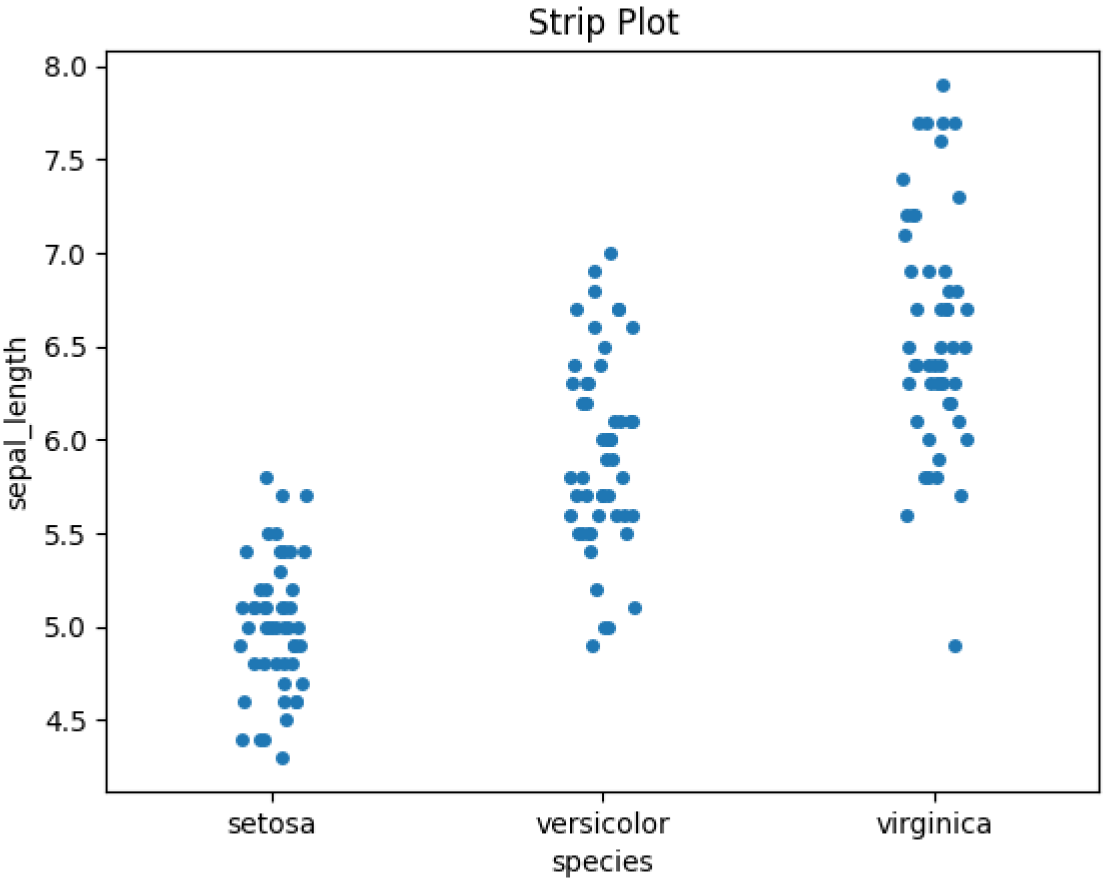
In [15]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
data = sns.load_dataset("iris")
```

Strip plot

In [16]:

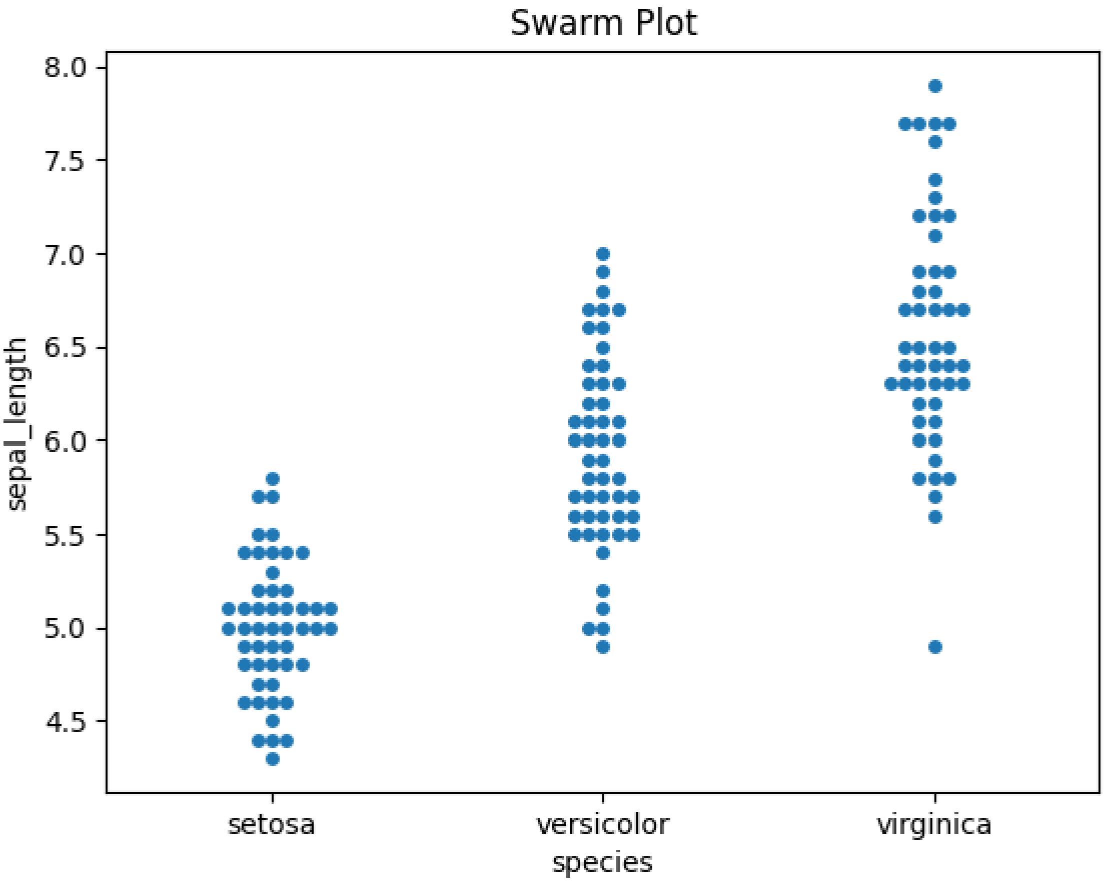
```
sns.stripplot(x="species", y="sepal_length", data=data)
plt.title("Strip Plot")
plt.show()
```



Swarm plot

In [17]:

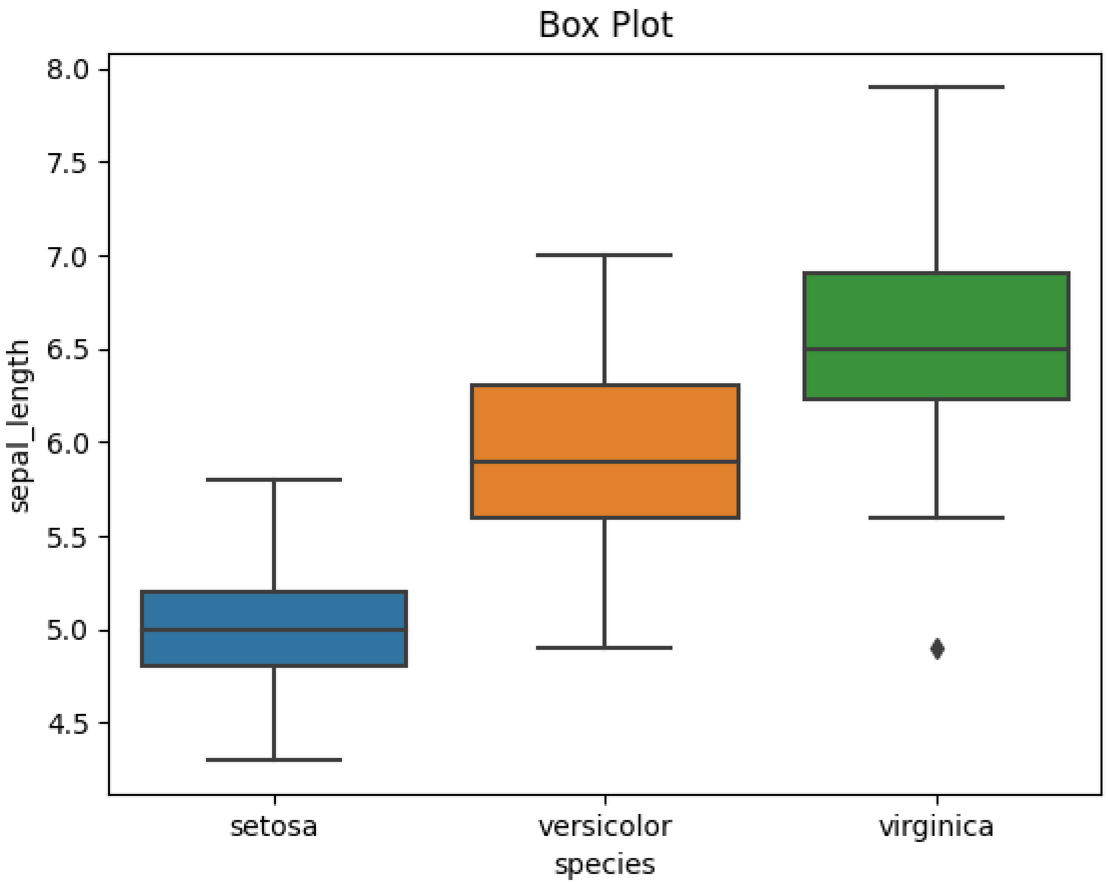
```
sns.swarmplot(x="species", y="sepal_length", data=data)
plt.title("Swarm Plot")
plt.show()
```



Box plot

In [18]:

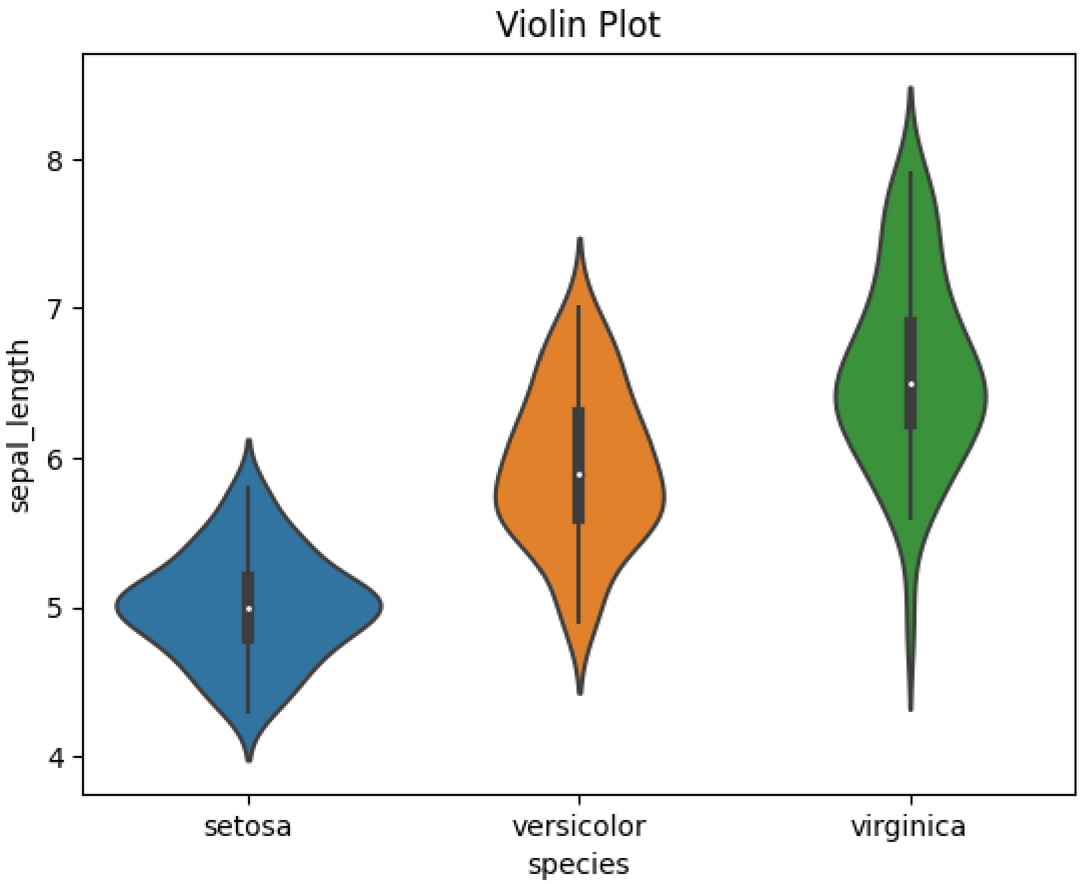
```
sns.boxplot(x="species", y="sepal_length", data=data)
plt.title("Box Plot")
plt.show()
```



Violin plot

In [19]:

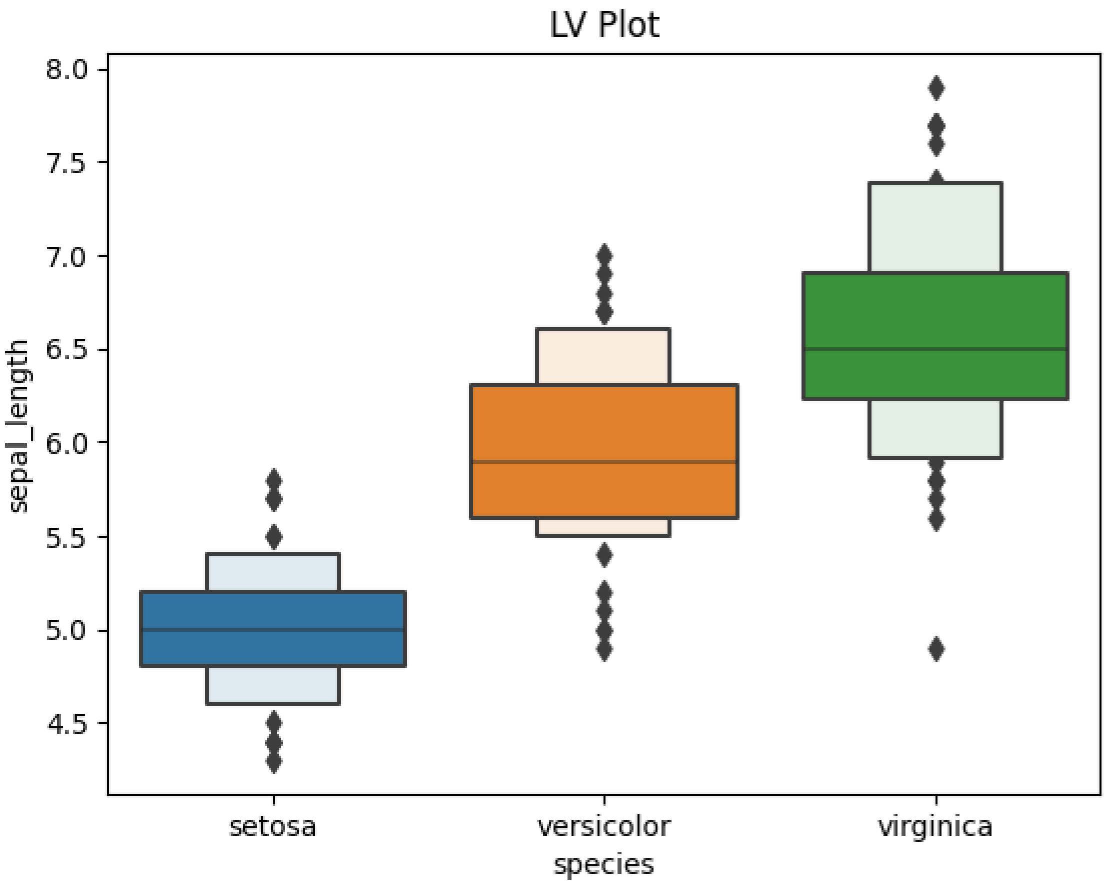
```
sns.violinplot(x="species", y="sepal_length", data=data)
plt.title("Violin Plot")
plt.show()
```



LV plot (Letter-Value plot)

In [27]:

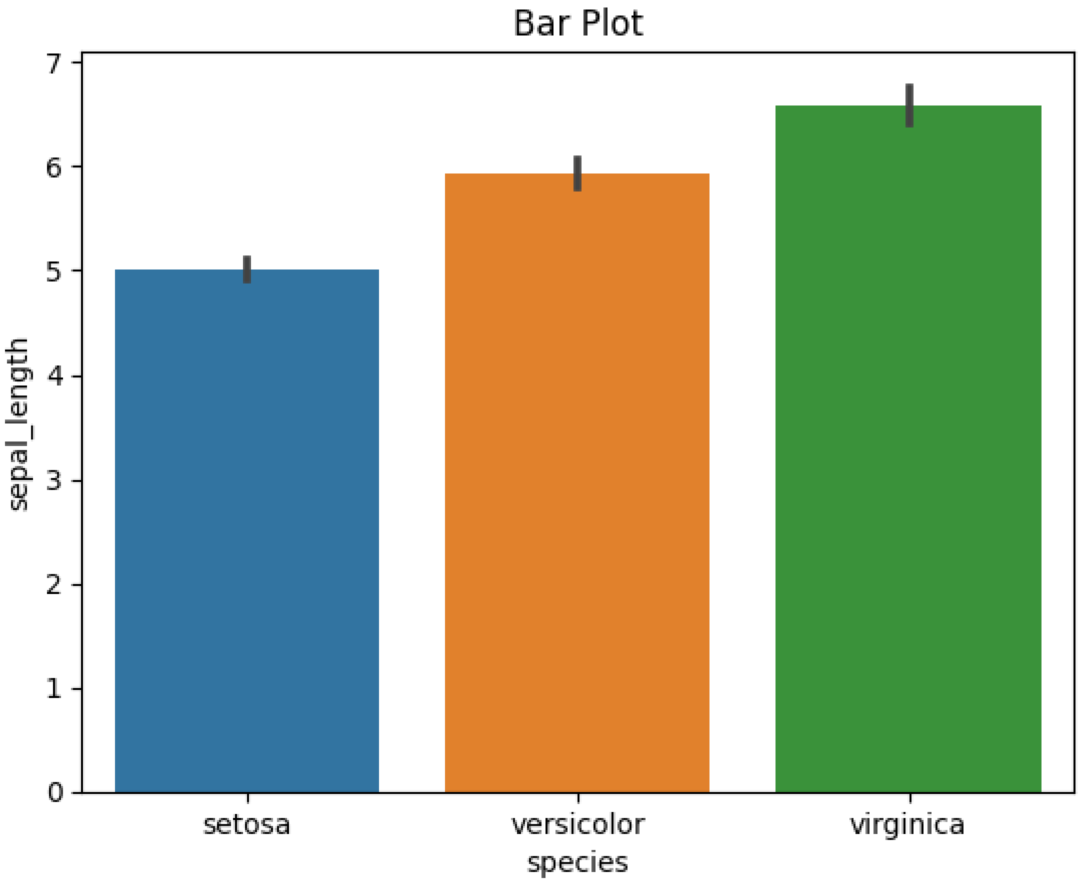
```
sns.boxenplot(x="species", y="sepal_length", data=data)
plt.title("LV Plot")
plt.show()
```



Bar Plot

In [21]:

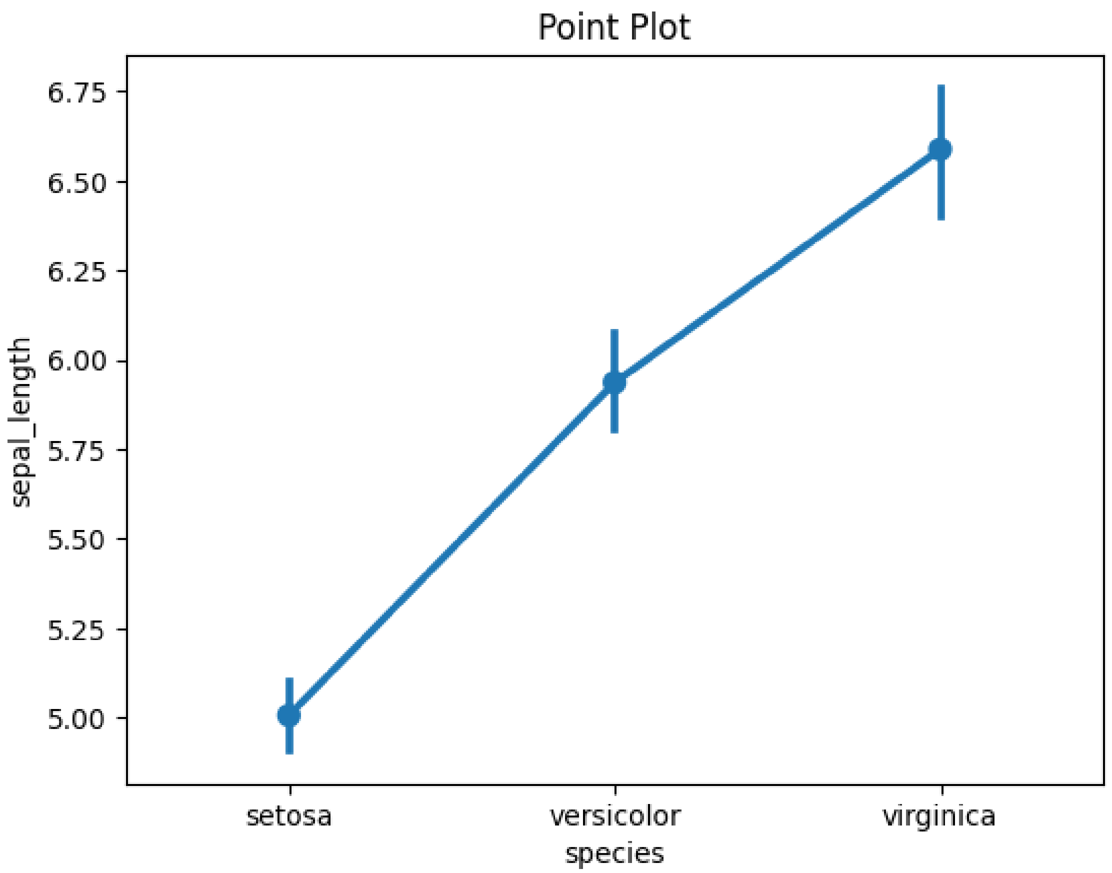
```
sns.barplot(x="species", y="sepal_length", data=data)
plt.title("Bar Plot")
plt.show()
```



Point plot

In [22]:

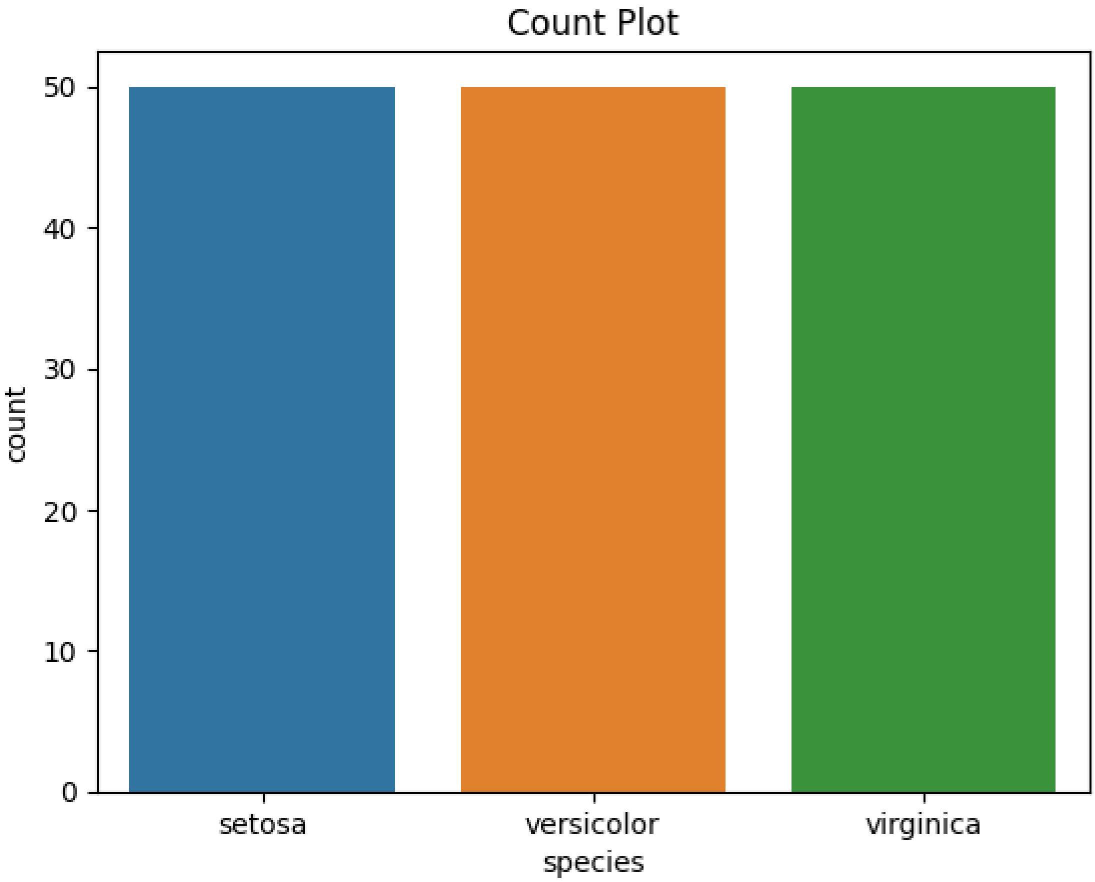
```
sns.pointplot(x="species", y="sepal_length", data=data)
plt.title("Point Plot")
plt.show()
```



Count plot

In [23]:

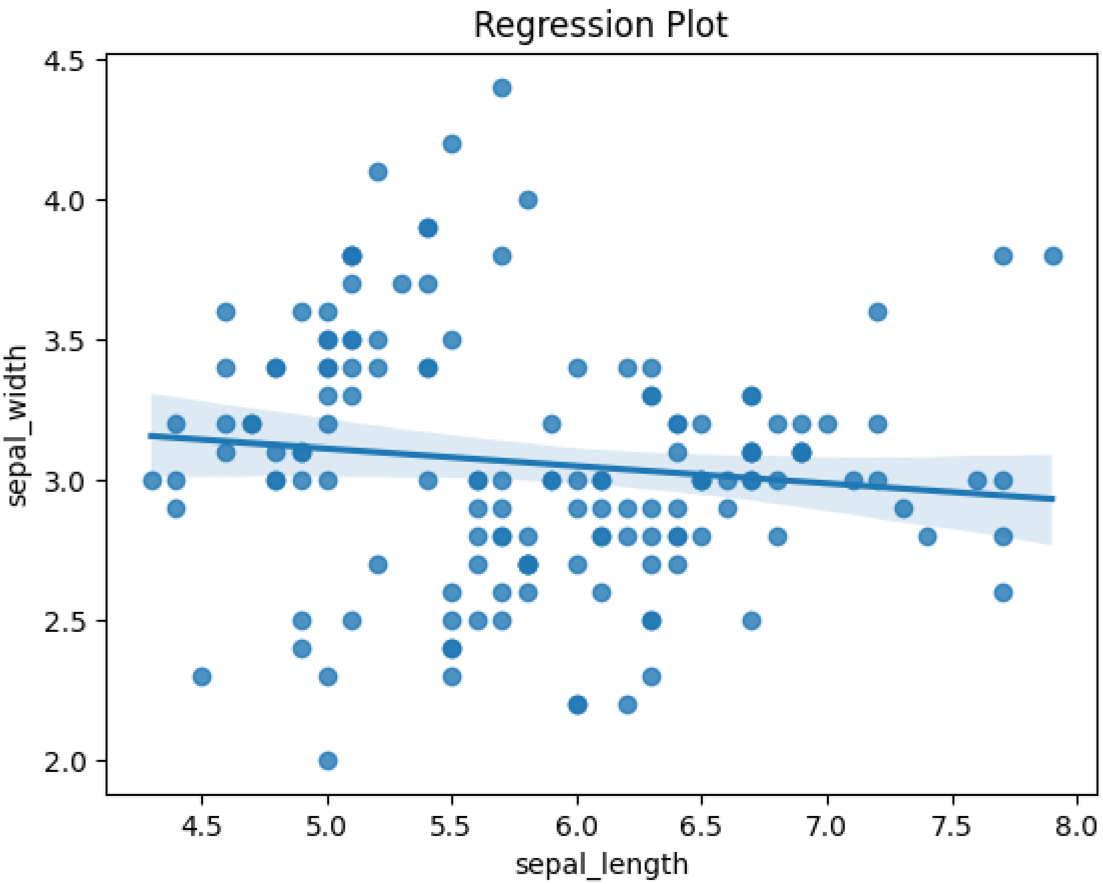
```
sns.countplot(x="species", data=data)
plt.title("Count Plot")
plt.show()
```



Regression plot

In [24]:

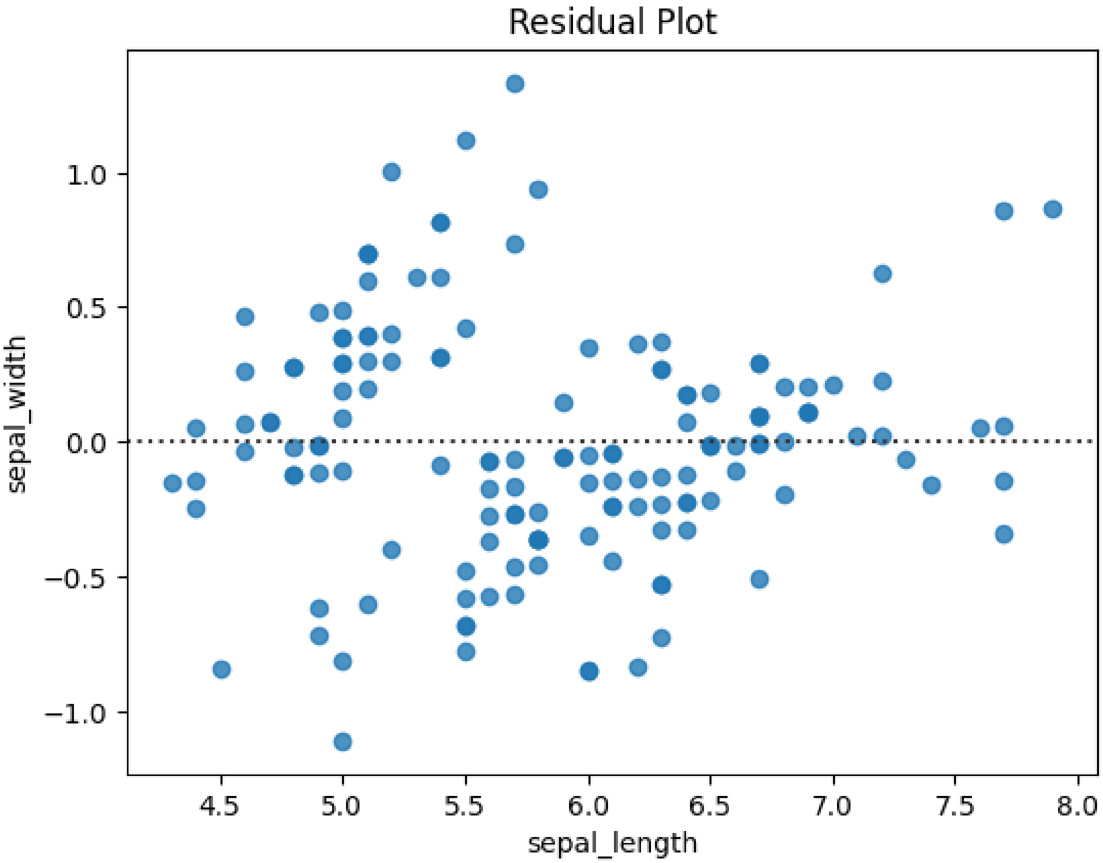
```
sns.regplot(x="sepal_length", y="sepal_width", data=data)
plt.title("Regression Plot")
plt.show()
```



Residual plot

In [25]:

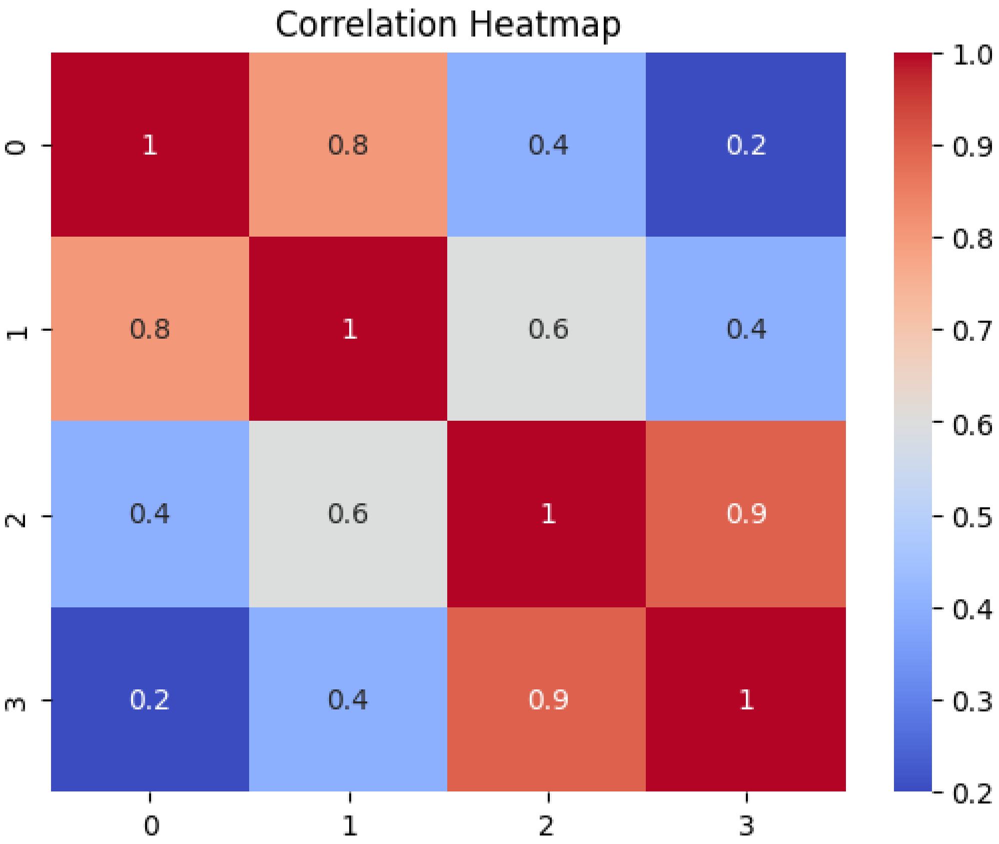
```
sns.residplot(x="sepal_length", y="sepal_width", data=data)
plt.title("Residual Plot")
plt.show()
```



Heatmap

In [29]:

```
# Sample correlation data
correlation_data = [[1.0, 0.8, 0.4, 0.2],
                    [0.8, 1.0, 0.6, 0.4],
                    [0.4, 0.6, 1.0, 0.9],
                    [0.2, 0.4, 0.9, 1.0]]

# Create a heatmap
sns.heatmap(correlation_data, annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```

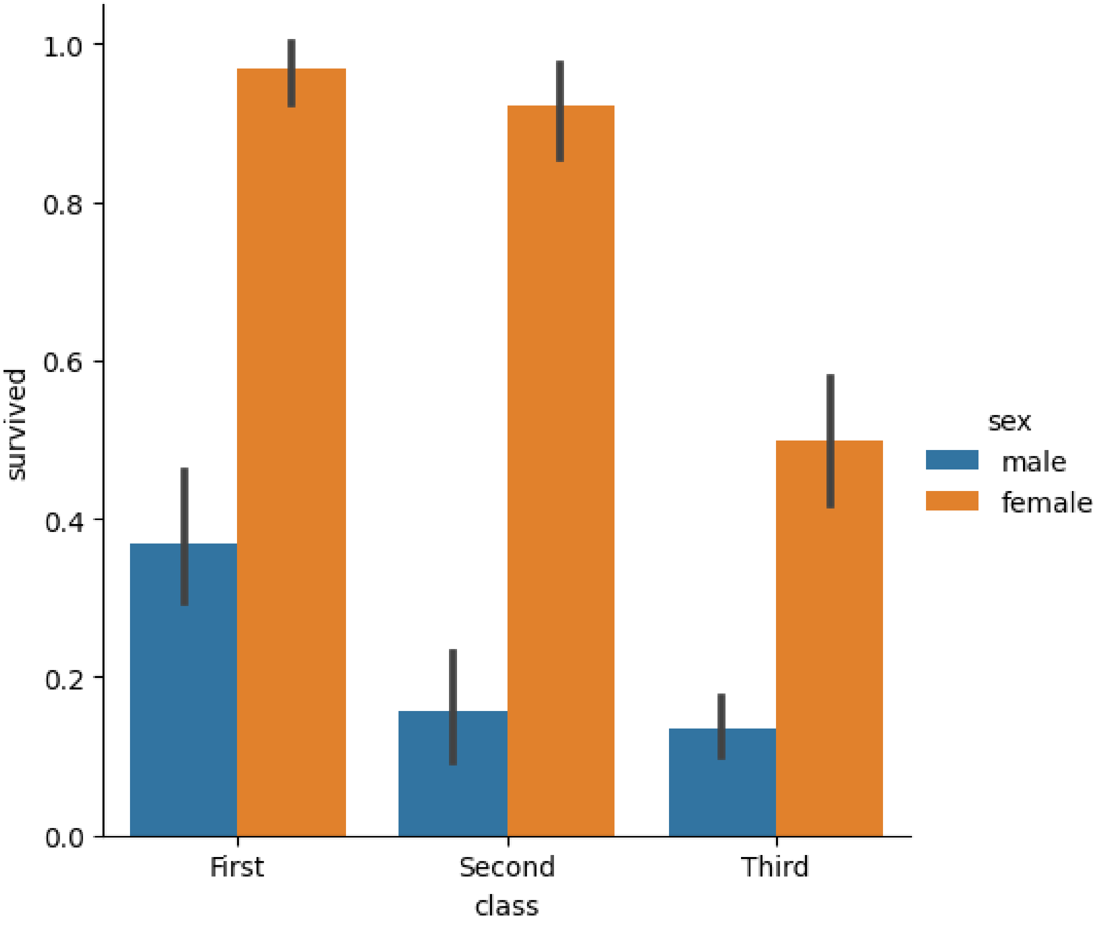


In [30]:

```
# Sample DataFrame
data = sns.load_dataset("tips")

# Create a FacetGrid
g = sns.FacetGrid(data, col="time", row="sex")
g.map(sns.scatterplot, "total_bill", "tip")

# Show the plots
plt.show()
plt.clf()
```

In [39]:

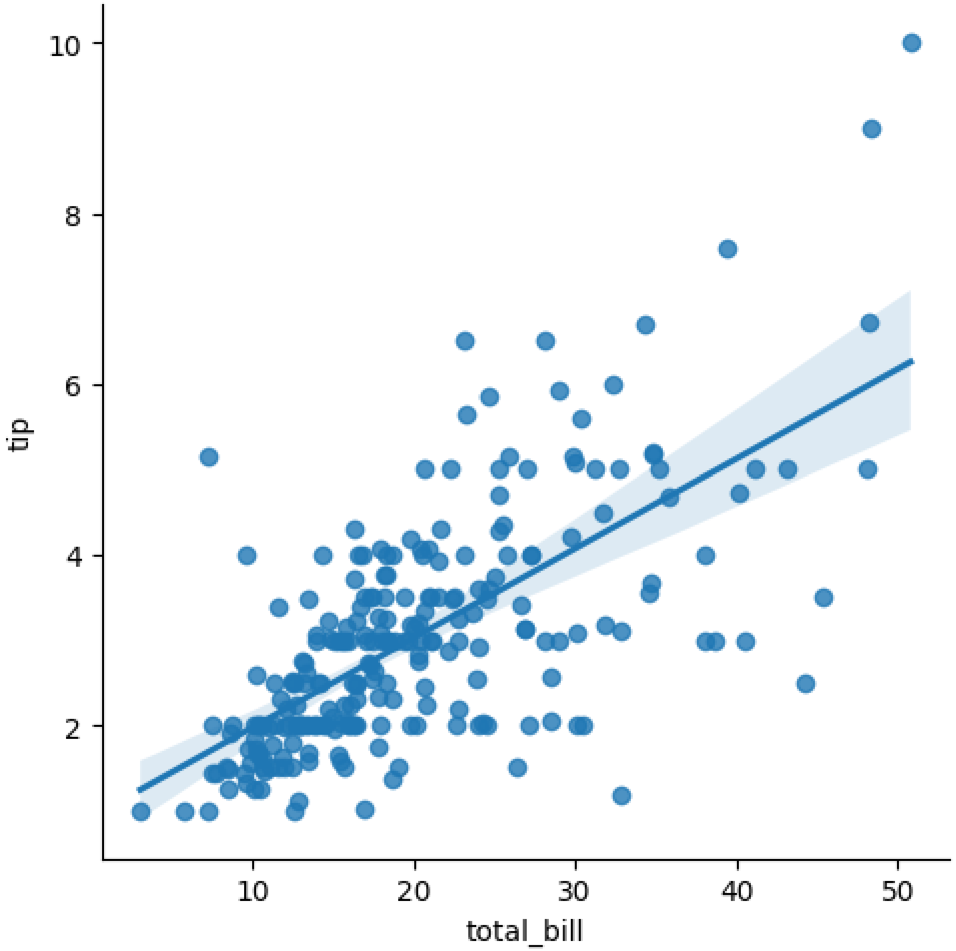
```
# Sample DataFrame
data = sns.load_dataset("titanic")

# Create a FactorPlot (catplot)
g = sns.catplot(x="class", y="survived", hue="sex", data=data, kind="bar")

# Show the plot
plt.show()
plt.clf()
```



```
<Figure size 640x480 with 0 Axes>
```

In [38]:

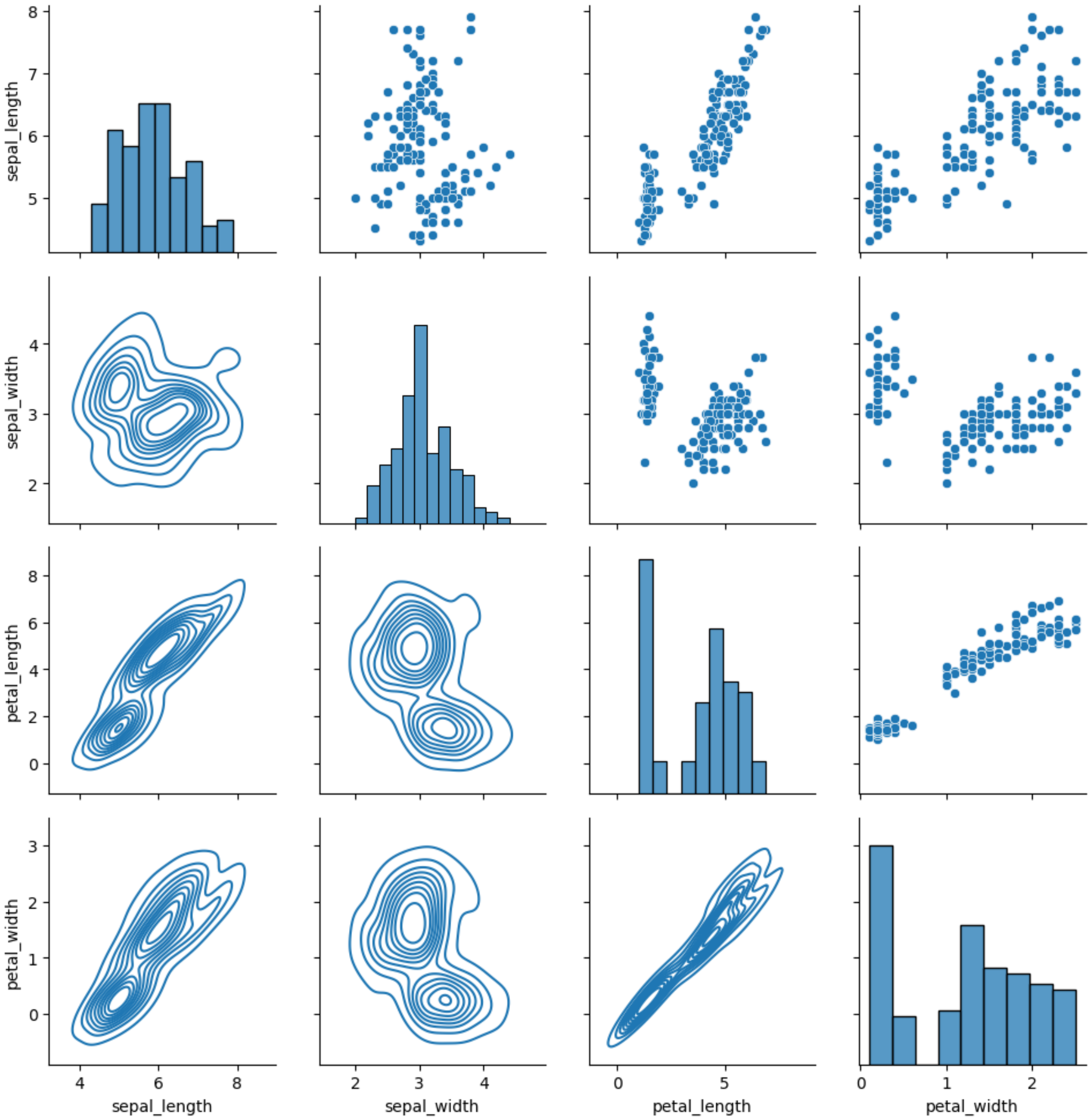
```python
# Sample DataFrame
data = sns.load_dataset("tips")

# Create an lmplot
sns.lmplot(x="total_bill", y="tip", data=data)

# Show the plot
plt.show()
plt.clf()
```



```
<Figure size 640x480 with 0 Axes>
```

In [37]:

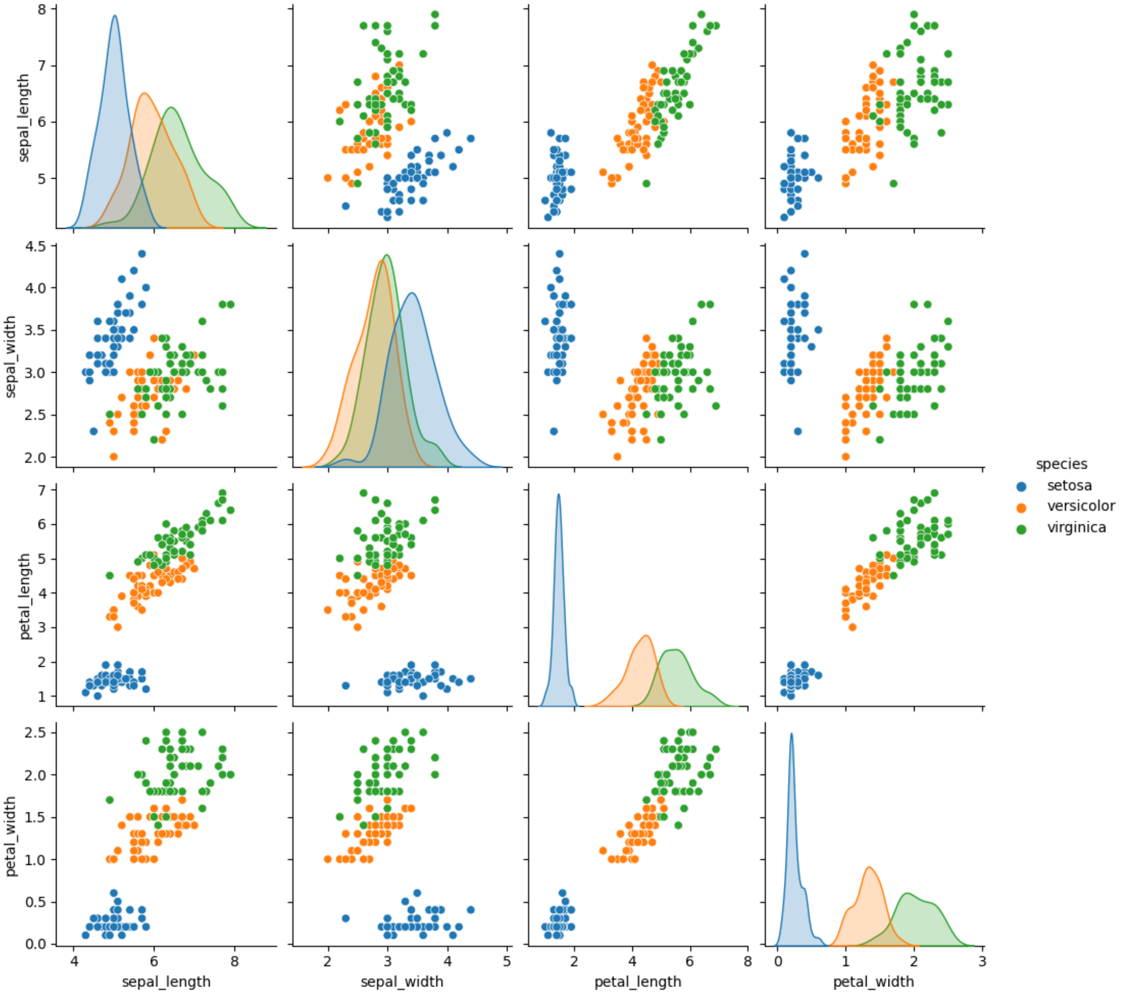
```
# Sample DataFrame
data = sns.load_dataset("iris")

# Create a PairGrid
g = sns.PairGrid(data)
g.map_upper(sns.scatterplot)
g.map_lower(sns.kdeplot)
g.map_diag(sns.histplot)

# Show the plots
plt.show()
plt.clf()
```



```
<Figure size 640x480 with 0 Axes>
```

In [36]:

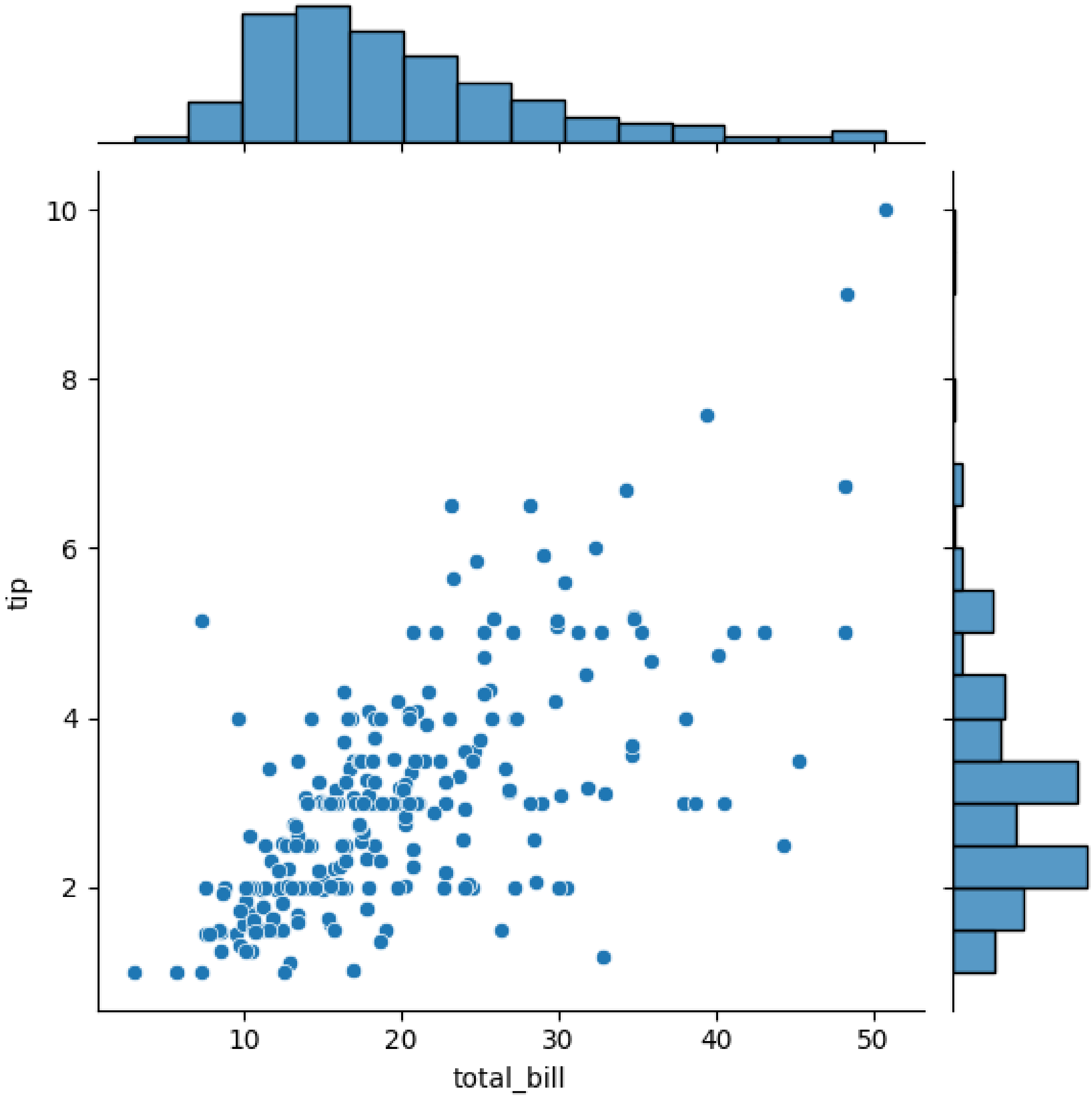
```
# Sample DataFrame
data = sns.load_dataset("iris")

# Create a PairPlot
sns.pairplot(data, hue="species")

# Show the plot
plt.show()
plt.clf()
```



```
<Figure size 640x480 with 0 Axes>
```

In [35]:

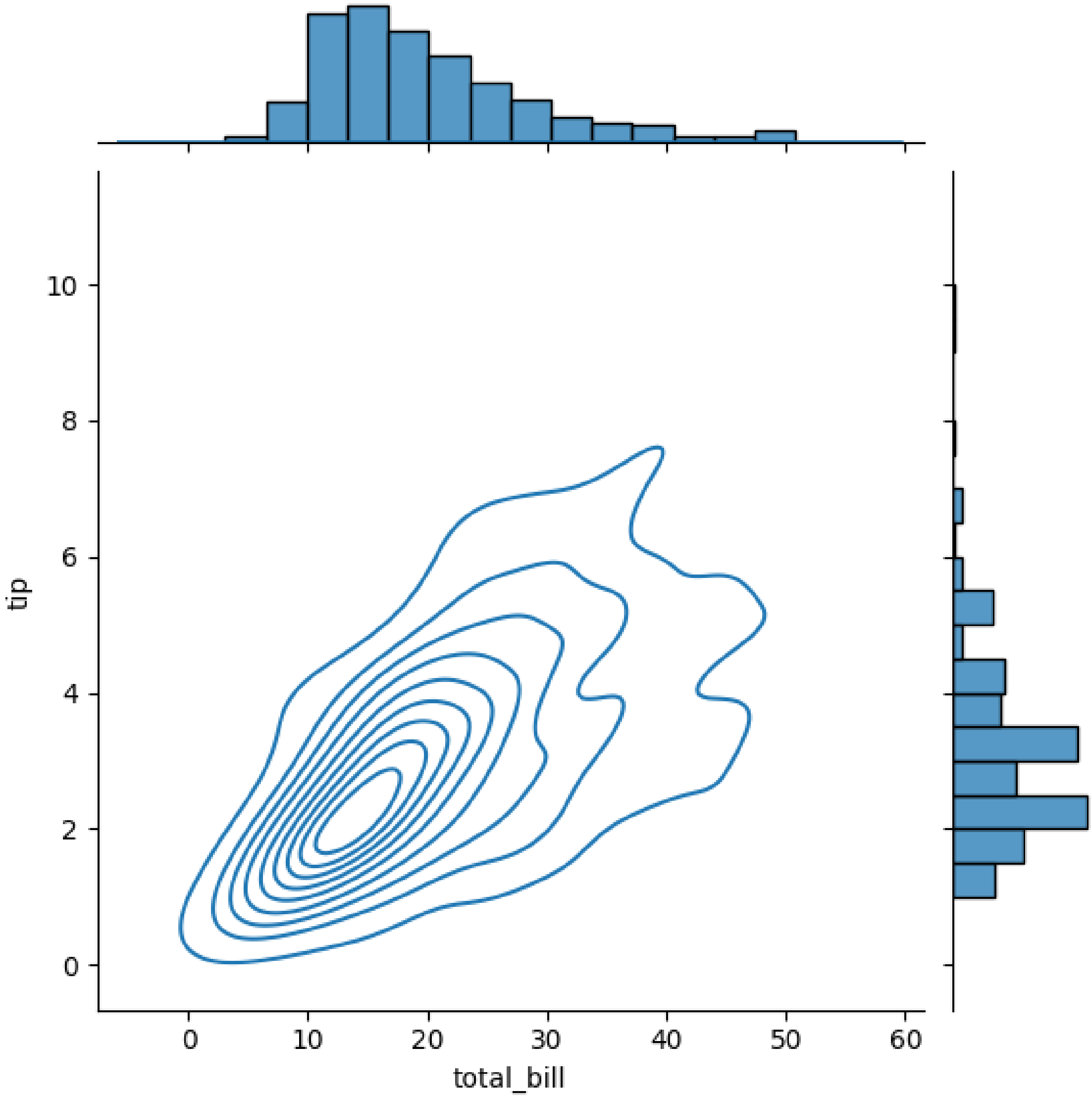
```
# Sample DataFrame
data = sns.load_dataset("tips")

# Create a JointPlot
sns.jointplot(x="total_bill", y="tip", data=data, kind="scatter")

# Show the plot
plt.show()
plt.clf()
```



```
<Figure size 640x480 with 0 Axes>
```

In [34]:

```
# Sample DataFrame
data = sns.load_dataset("tips")

# Create a Complex JointPlot
g = sns.jointplot(x="total_bill", y="tip", data=data, kind="kde")
g.plot_marginals(sns.histplot)

# Show the plot
plt.show()
plt.clf()
```



```
<Figure size 640x480 with 0 Axes>
```