In [1]:
```
import pandas as pd
import seaborn as sns
```

In [2]:
```
df = sns.load_dataset('iris')
df
```

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

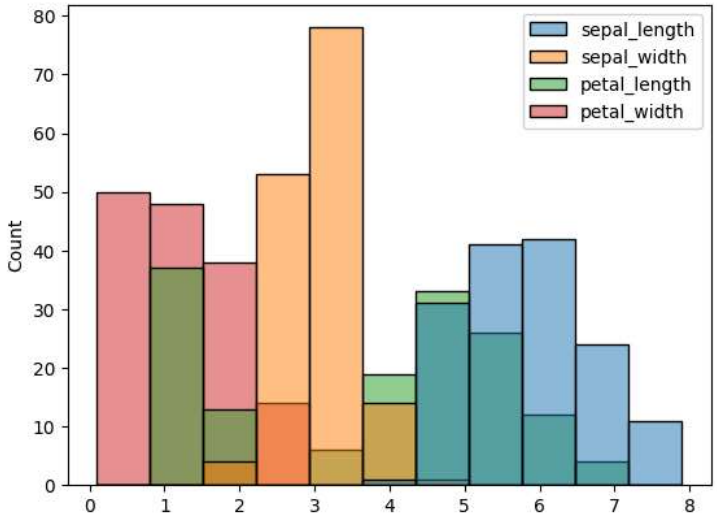
150 rows × 5 columns

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

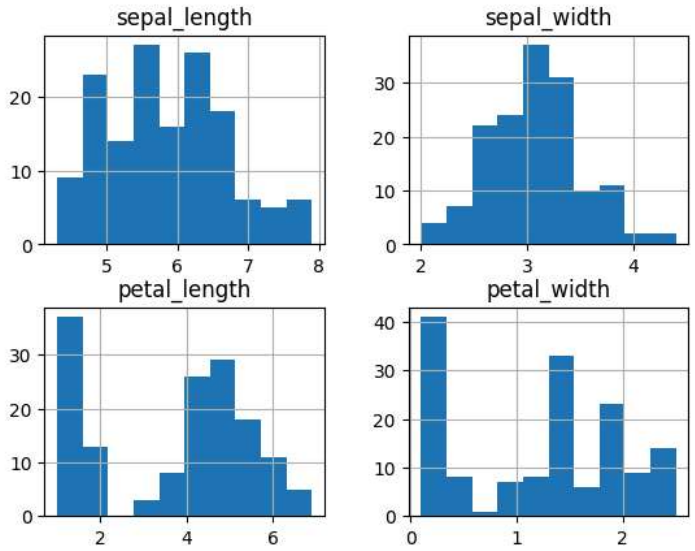
In [4]:
```
# No need fro categorical data i.e. no 'x' and no 'y'
sns.histplot(data=df)
```

Out[4]:
```
<Axes: ylabel='Count'>
```

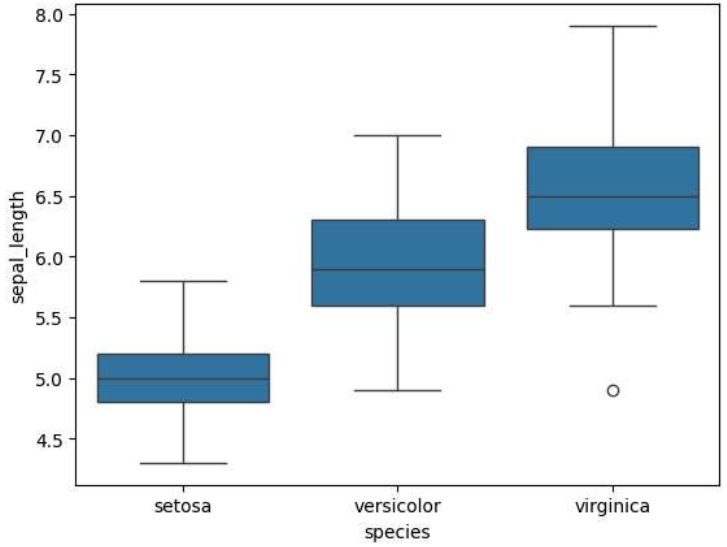
In [5]:
```
df.hist()
```

Out[5]:
```
array([[<Axes: title={'center': 'sepal_length'}>,
        <Axes: title={'center': 'sepal_width'}>],
       [<Axes: title={'center': 'petal_length'}>,
        <Axes: title={'center': 'petal_width'}>]], dtype=object)
```



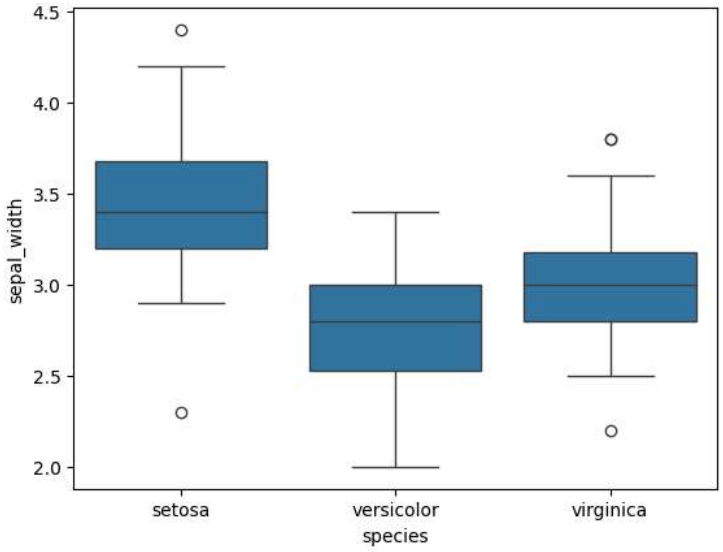
In [6]:
```
sns.boxplot(x='species', y='sepal_length', data=df)
```

Out[6]: <Axes: xlabel='species', ylabel='sepal_length'>



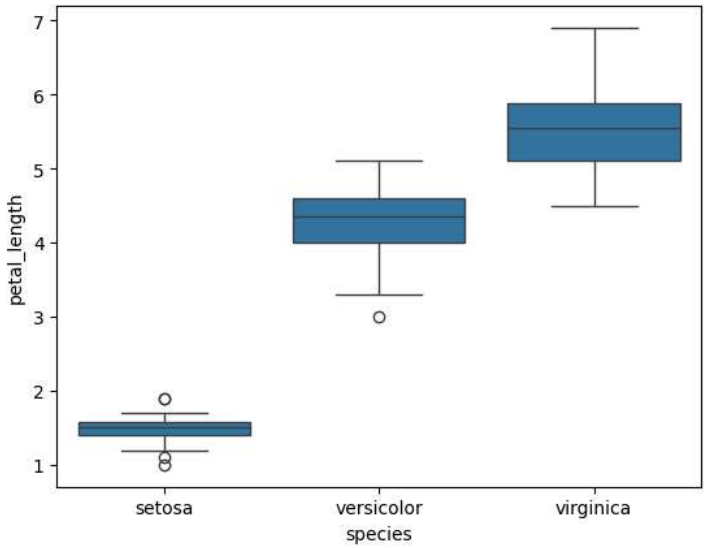
In [7]:
```
sns.boxplot(x='species', y='sepal_width', data=df)
```

Out[7]: <Axes: xlabel='species', ylabel='sepal_width'>



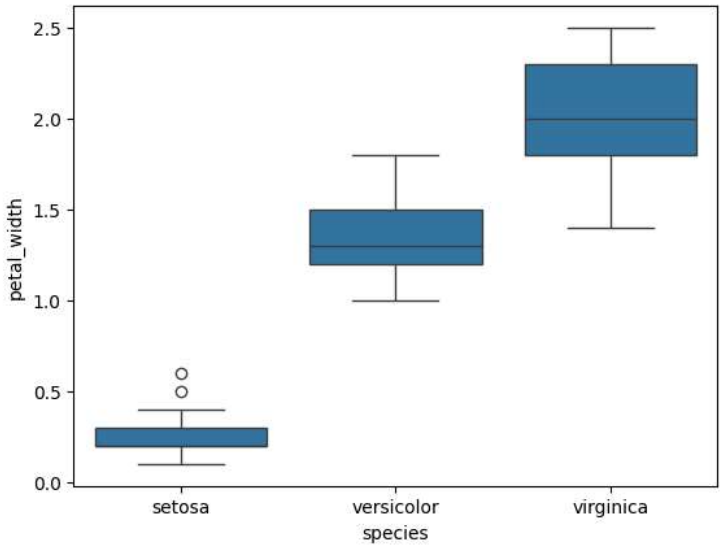
In [8]:
```
sns.boxplot(x='species', y='petal_length', data=df)
```

Out[8]: <Axes: xlabel='species', ylabel='petal_length'>



In [9]:
```
sns.boxplot(x='species', y='petal_width', data=df)
```

Out[9]: <Axes: xlabel='species', ylabel='petal_width'>



In [10]:
```
def fun(x):
    upper_fence = q3(x) + (iqr(x)*1.5)
    lower_fence = q1(x) - (iqr(x)*1.5)
    outliers = []
    for i in x:
        if ((i>upper_fence) or (i<lower_fence)):
            outliers.append(i)

    print(outliers)

def q3(x):
    return x.quantile(0.75)

def q1(x):
    return x.quantile(0.25)

def iqr(x):
    a=q3(x)
    b=q1(x)
    return a-b
```

In [11]:
```
# Testing the function
t = df[df['species']=='virginica']
# t
r = t['sepal_length']
# r
fun(r)
```

```
[4.9]
```

In [12]:
```
for i in df['species'].unique():
    print("\n"+i+" species :")
    for j in ['sepal_length','sepal_width','petal_length','petal_width']:
        print(j+" arguement")
        t = df[df['species']==i]
        r = t[j]
        fun(r)
```

```
setosa species :
sepal_length arguement
[]
sepal_width arguement
[4.4, 2.3]
petal_length arguement
[1.1, 1.0, 1.9, 1.9]
petal_width arguement
[0.5, 0.6]

versicolor species :
sepal_length arguement
[]
sepal_width arguement
[]
petal_length arguement
[3.0]
petal_width arguement
[]

virginica species :
sepal_length arguement
[4.9]
sepal_width arguement
[3.8, 2.2, 3.8]
petal_length arguement
[]
petal_width arguement
[]
```