In [1]:

```
import torch
```

In [2]:

```
```

```
Ones Tensor:
 tensor([[1., 1.],
        [1., 1.]])


Random Tensor:
 tensor([[0.8190, 0.8996],
        [0.1871, 0.3406]])
```

In [3]:

```
A = torch.tensor([[1.0, 2.0], [3.0, 4.0]])
print(A)
```

```
tensor([[1., 2.],
        [3., 4.]])
```

In [4]:

```
C = torch.Tensor([1.0, 2.0])
print(C)
```

```
tensor([1., 2.])
```

In [ ]:

```
```

torch张量的操作-拼接

In [5]:

```
tensor_1 = torch.tensor([[1,2,3,4]])
tensor_2 = torch.tensor([[5,6,7,8]])
print(torch.cat([tensor_1,tensor_2],dim=0))
print(torch.cat([tensor_1,tensor_2],dim=1))
```

```
tensor([[1, 2, 3, 4],
        [5, 6, 7, 8]])
tensor([[1, 2, 3, 4, 5, 6, 7, 8]])
```

In [ ]:

```
```

torch张量的操作-索引

In [6]:

```
tensor = torch.randn(4,4)

print(tensor)
print(f'zhuihouhang  :{tensor[0]}')
print(f'diyilie :{tensor[:,0]}')
print(f'zuihou :{tensor[...,-1]}')
tensor[:,1] = 0
print(tensor)
```

```
tensor([[ 0.1831, -0.0547,  0.1206,  0.1275],
        [ 0.2595,  0.3999,  1.3737, -1.0439],
        [ 1.3060, -0.5432,  1.8676,  2.5768],
        [ 0.3005,  0.3353, -0.8293,  0.6532]])
zhuihouhang :tensor([ 0.1831, -0.0547,  0.1206,  0.1275])
diyilie :tensor([0.1831, 0.2595, 1.3060, 0.3005])
zuihou :tensor([ 0.1275, -1.0439,  2.5768,  0.6532])
tensor([[ 0.1831,  0.0000,  0.1206,  0.1275],
        [ 0.2595,  0.0000,  1.3737, -1.0439],
        [ 1.3060,  0.0000,  1.8676,  2.5768],
        [ 0.3005,  0.0000, -0.8293,  0.6532]])
```

In [ ]:

```

```

torch张量的操作-数据类型转换

In [7]:

```
import torch
t = torch.ones(5)
n = t.numpy()
print(t)
print(n)
```

```
tensor([1., 1., 1., 1., 1.])
[1. 1. 1. 1. 1.]
```

In [8]:

```
import torch
import numpy as np
data = [[1,2],[3,4]]
np_array = np.array(data)
x_np = torch.from_numpy(np_array)
x_np
```

Out[8]:

```
tensor([[1, 2],
        [3, 4]], dtype=torch.int32)
```

In [ ]:

```

```

torch张量的操作-数据类型转换 图片转换为张量

In [10]:

```
from PIL import Image
from torchvision import transforms
image_path = r'form_tensor.jpg'
image = Image.open(image_path)

transform = transforms.ToTensor()
tensor_image = transform(image)

print(type(tensor_image))
```

```
<class 'torch.Tensor'>
```

In [ ]:

```

```

torch张量的操作-数据类型转换 张量转换为图片

In [11]:

```
import torch
from torchvision import transforms
from PIL import Image
```

```
# 生成随机张量
tensor_image = torch.randn(3, 224, 224)
# 将张量转换为图像
transformed_image = transforms.ToPILImage()(tensor_image)
# 保存图像的路径
save_path = r'form_tensor.jpg'
# 保存图像
transformed_image.save(save_path)
```

In [25]:

```
conda env list
```

```
# conda environments:
#
base                  *  C:\ProgramData\Anaconda3
008                      C:\ProgramData\Anaconda3\envs\008


Note: you may need to restart the kernel to use updated packages.
```

In [ ]:

```
```

github.com

In [14]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris
```

In [18]:

```
iris = load_iris()

iris_df = pd.DataFrame(data=iris.data,columns=iris.feature_names)

iris_df['species'] = pd.Categorical.from_codes(iris.target,iris.target_names)
```

In [26]:

```
print(iris_df.head())
```

```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

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

In [27]:

```
print('数据集的维度:',iris_df.shape)
```

```
数据集的维度: (150, 5)
```

In [28]:

```
iris_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 (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   species            150 non-null    category
dtypes: category(1), float64(4)
memory usage: 5.1 KB
```

In [29]:

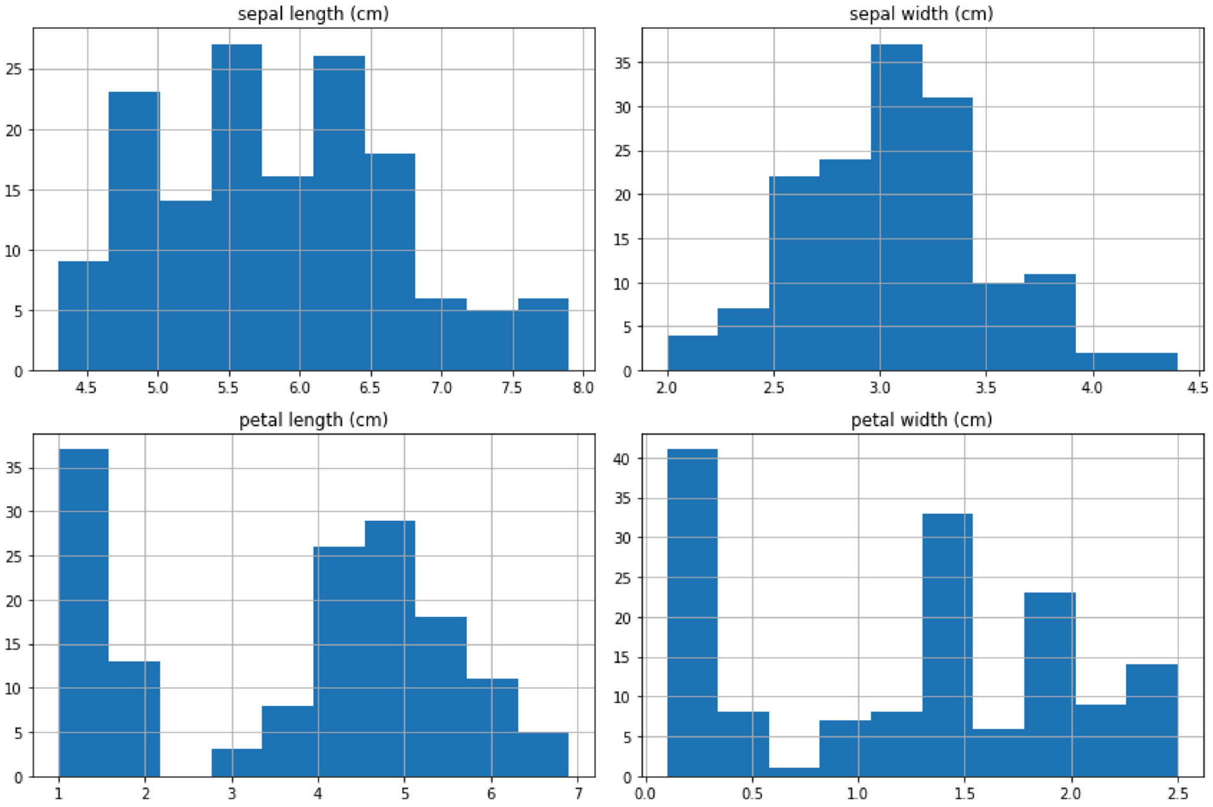
```
print(iris_df.describe())
```

```
       sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000
mean            5.843333          3.057333           3.758000
std             0.828066          0.435866           1.765298
min             4.300000          2.000000           1.000000
25%             5.100000          2.800000           1.600000
50%             5.800000          3.000000           4.350000
75%             6.400000          3.300000           5.100000
max             7.900000          4.400000           6.900000

       petal width (cm)
count        150.000000
mean           1.199333
std            0.762238
min            0.100000
25%            0.300000
50%            1.300000
75%            1.800000
max            2.500000
```

In [31]:

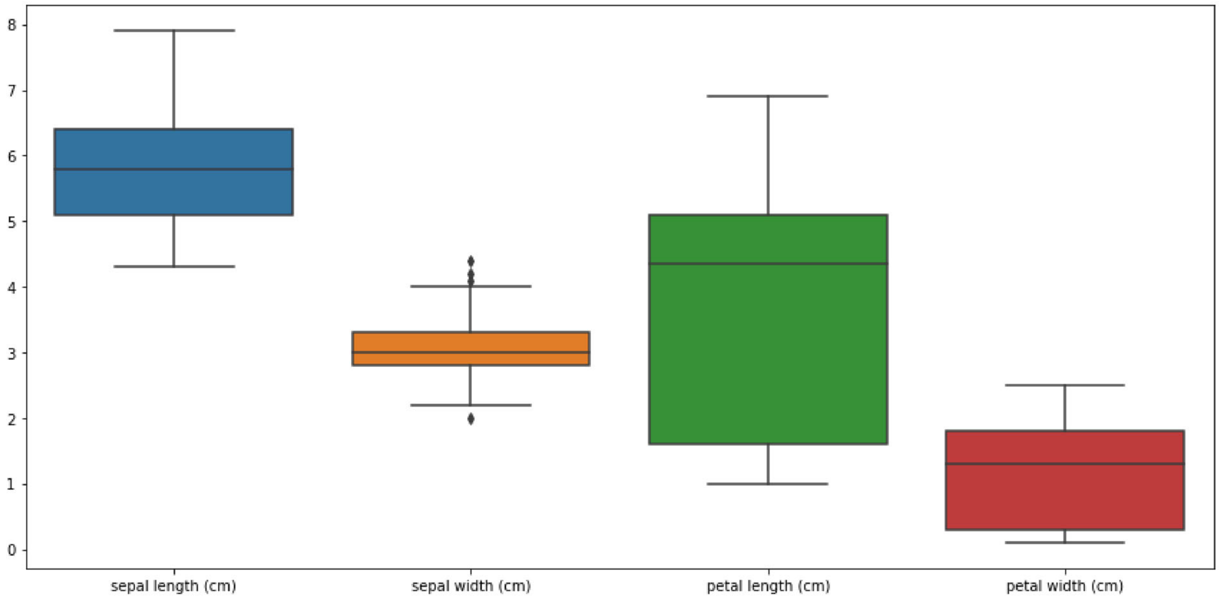
```
iris_df.hist(figsize=(12,8))
plt.tight_layout()
plt.show()
```



In [35]:

```
import seaborn as sns
plt.figure(figsize=(12,6))
sns.boxplot(data=iris_df)
plt.tight_layout()
plt.show()

plt.savefig('boxplot_features.png')
```

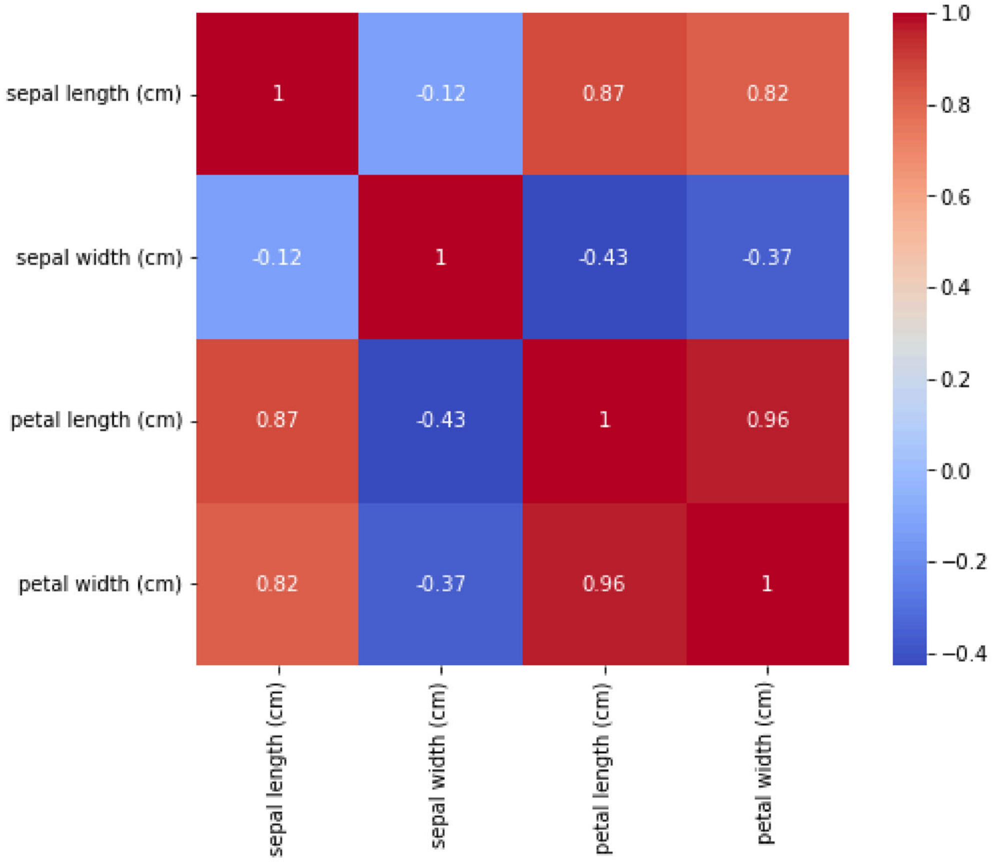


```
<Figure size 432x288 with 0 Axes>
```

In [40]:

```
correlation_matrix = iris_df.corr()


plt.figure(figsize=(8,6))
sns.heatmap(correlation_matrix,annot=True,cmap='coolwarm',square=True)
plt.show()
```

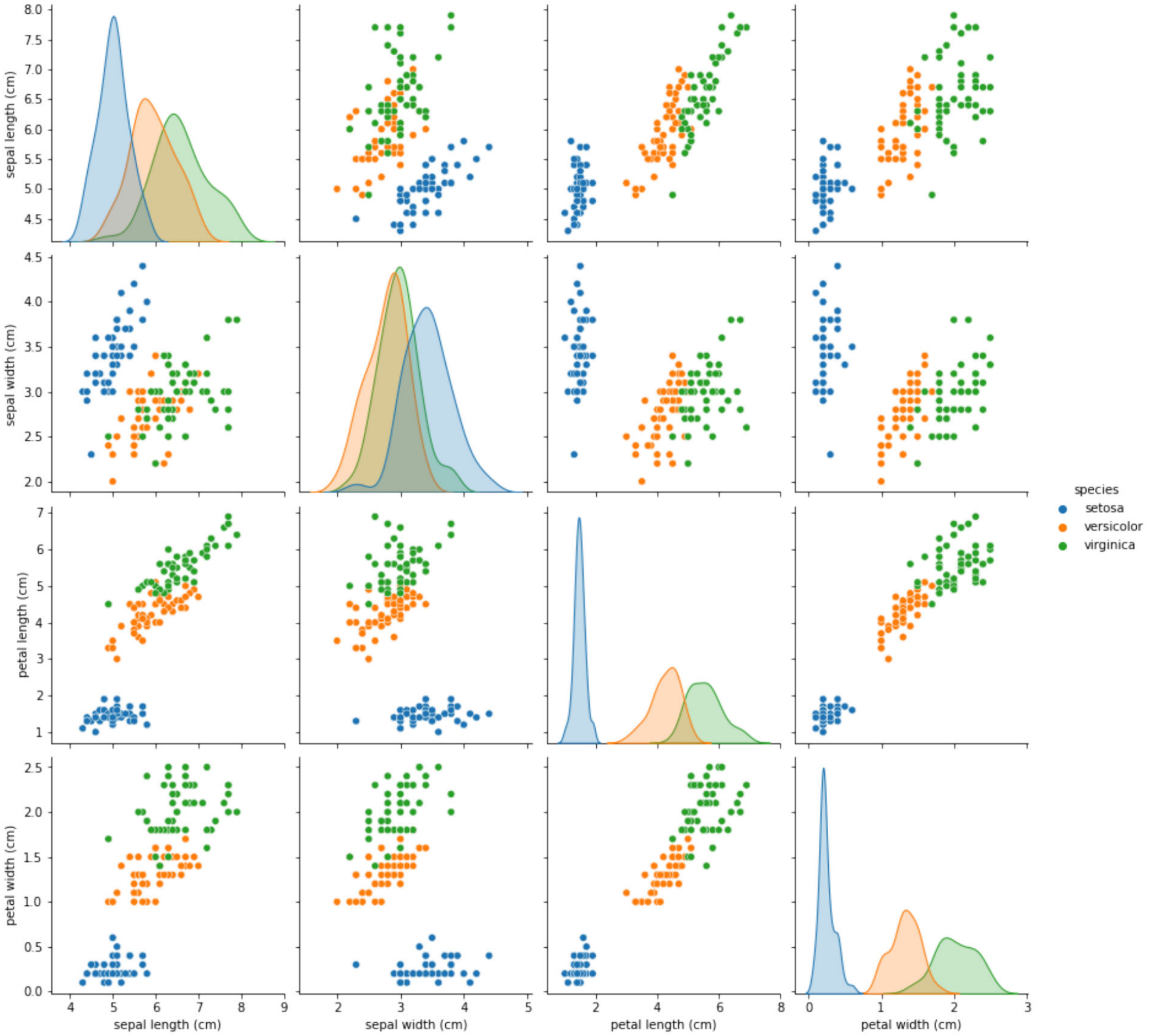


In [39]:

```
sns.pairplot(iris_df,hue = 'species',height=3)
```

Out[39]:

```
<seaborn.axisgrid.PairGrid at 0x22a043114f0>
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [7]:

```
import torch

print(torch.__version__) # pytorch版本
print(torch.version.cuda) # cuda版本
print(torch.cuda.is_available()) # 查看cuda是否可用
```

```
2.5.1+cpu
None
False
```

In [9]:

```
torch.Tensor(2,3)
```

Out[9]:

```
tensor([[2.8223e-37, 1.3424e-42, 0.0000e+00],
        [0.0000e+00, 0.0000e+00, 0.0000e+00]])
```

In [10]:

```
torch.Tensor([2,3])
```

Out[10]:

```
tensor([2., 3.])
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: