In [1]:
```
import sklearn
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import scipy
import statistics
from sklearn import model_selection
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import label_binarize
import os
```

In [29]:
```
path = os.getcwd()
iris_df = pd.read_csv(path+'\\Learn Dataset\\iris_dataset_missing.csv')
iris_df_nona = iris_df.dropna()
iris_df_nona["Class"] = list(iris_df_nona.loc[:,"species"].values)
iris_df_nona["Class"]=iris_df_nona["Class"].replace("Iris-versicolor",0).replace("Iris-setosa",1).replace("Iris-virginica",2)
heart_df = pd.read_csv(path+'\\Learn Dataset\\heart_disease_missing.csv')
heart_df_nona = heart_df.dropna()
features = ["exang","thal","slope","cp"]
heart_df_sub = heart_df_nona.copy()
for i in heart_df_nona.columns:
    if i not in features and i not in ["target"]:
        heart_df_sub.drop(columns = [i], inplace=True)
```

CM3

Noticable outliers

Iris Dataset

Iris data set has several missing values which have been discarded which is one method of dealing with missing values

In [30]:
```
iris_df.dropna().describe()
```

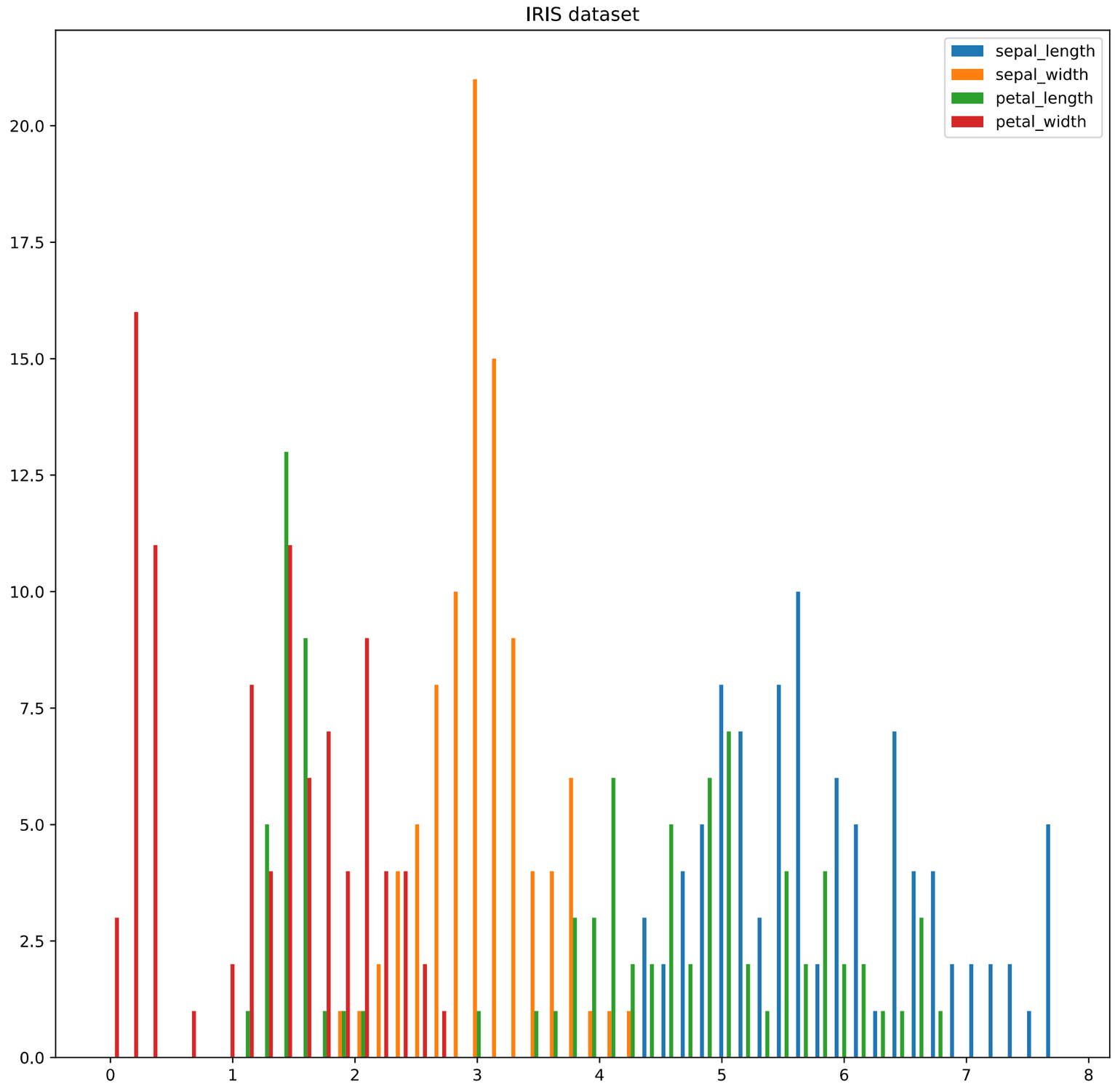
Out[30]:

	sepal_length	sepal_width	petal_length	petal_width
count	93.000000	93.000000	93.000000	93.000000
mean	5.867894	3.054935	3.808118	1.209826
std	0.892271	0.439463	1.811399	0.793656
min	4.344007	1.946010	1.033031	-0.072203
25%	5.152435	2.794790	1.541564	0.333494
50%	5.636744	3.049459	4.192791	1.369266
75%	6.478961	3.239682	5.098860	1.837925
max	7.795561	4.249211	6.768611	2.603123

Looking at the summary of the dataset, we can notice that petal_width has negative value(s) which is(are) clearly an outlier since length and width are non negative real numbers. Such negatives values in this case can be removed. However, a histogram can paint a much better picture in this regard.

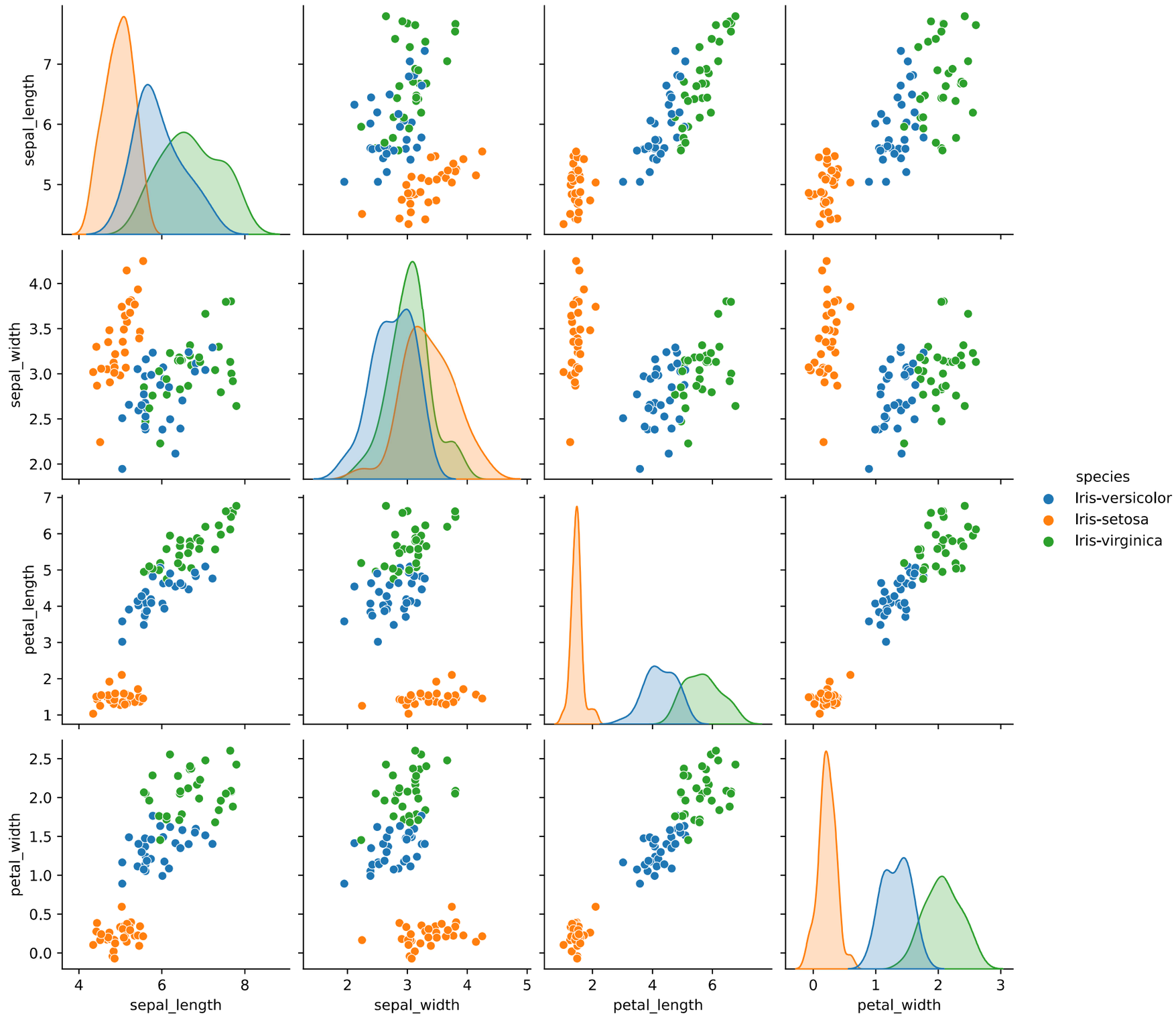
In [31]:
```
iris_df_X = iris_df.copy()
```

```
iris_df_X = iris_df.dropna().drop(columns=['species'])
iris_df_Y = iris_df.dropna().drop(columns=["sepal_length","sepal_width","petal_length","petal_width"])
n_bins = 50
fig1 = plt.figure(figsize=(12,12))
plt.hist([iris_df_X.iloc[:,0],iris_df_X.iloc[:,1],iris_df_X.iloc[:,2], iris_df_X.iloc[:,3]],n_bins)
plt.title('IRIS dataset')
plt.legend(["sepal_length","sepal_width","petal_length","petal_width"])
plt.show()
```



As discussed, negative values in petal width is an outlier. Apart from that, notice how there are multiple peaks petal_width, and also, the presence of a peak far right in case of sepal_length. One mustn't decide them as outliers without further investigation. Let us compare it with auto-correlation plots (diagonal plots) in pairplot figure. Looking at the auto-corr plot for sepal_length, we can see that such peaks are associated with specific species and hence they are NOT outliers. The only outlier is the negative values which need to be removed.

In [32]:
```
sn = sns.pairplot(iris_df_nona.drop(columns=["Class"]),hue='species', dropna=True)
```



In [33]:
```
outlier_ = iris_df_nona[iris_df_nona['petal_width']<0]
iris_df_nona.drop(index = list(outlier_.index), inplace=True)
iris_df_nona.describe()
```

Out[33]:

	sepal_length	sepal_width	petal_length	petal_width	Class
count	91.000000	91.000000	91.000000	91.000000	91.000000
mean	5.890573	3.054944	3.859047	1.237675	1.000000
std	0.888618	0.444313	1.797814	0.779370	0.829993
min	4.344007	1.946010	1.033031	0.020731	0.000000
25%	5.183002	2.783261	1.552927	0.341349	0.000000
50%	5.695405	3.049459	4.276817	1.400355	1.000000
75%	6.486531	3.265000	5.141844	1.860809	2.000000
max	7.795561	4.249211	6.768611	2.603123	2.000000

We have removed the negative entries in the petal_width attribute

Heart Dataset

Similar to the previous case, we are dealing with missing values in our heart disease dataset by removing nan values.

In [34]:
```
heart_df_nona.describe()
```

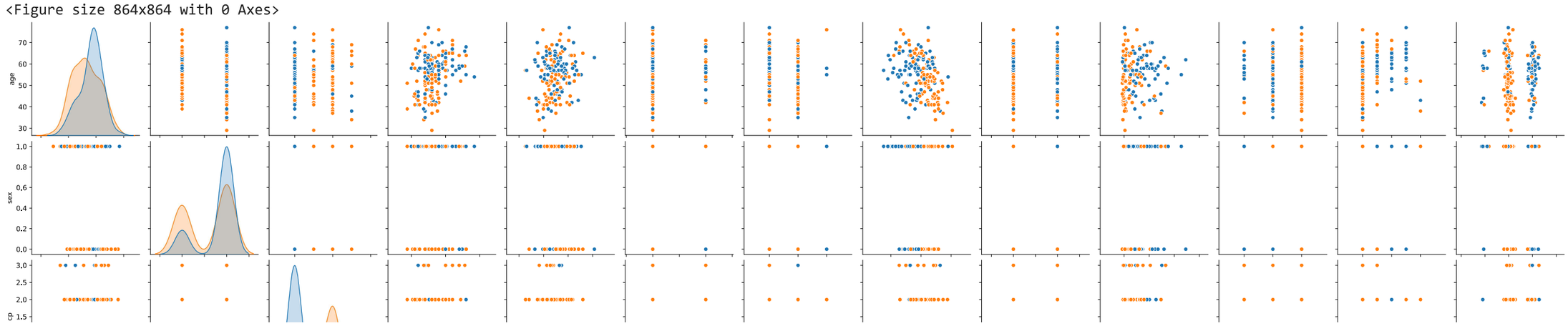
Out[34]:

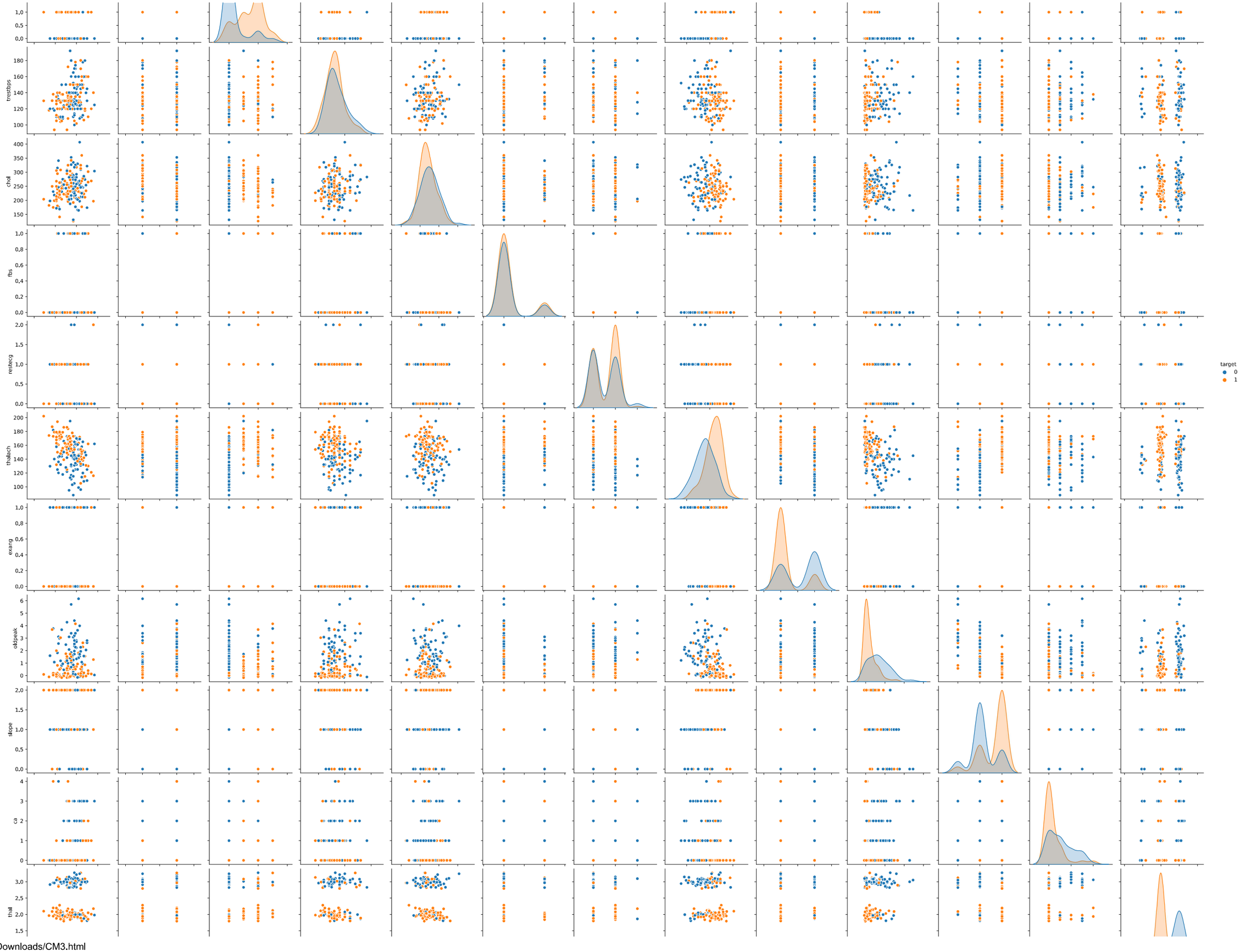
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000
mean	54.649425	0.695402	0.948276	132.852160	246.142975	0.137931	0.563218	149.446186	0.367816	1.096423	1.396552	0.747126	2.351880	0.528736
std	9.275368	0.461565	1.015870	18.438982	46.541790	0.345823	0.541912	22.059644	0.483603	1.279873	0.634454	1.077552	0.613963	0.500614
min	29.000000	0.000000	0.000000	93.944184	126.085811	0.000000	0.000000	88.032613	0.000000	-0.176438	0.000000	0.000000	0.858554	0.000000
25%	47.250000	0.000000	0.000000	120.026582	213.267299	0.000000	0.000000	134.528882	0.000000	0.056498	1.000000	0.000000	1.963829	0.000000
50%	55.000000	1.000000	1.000000	130.021392	242.960083	0.000000	1.000000	151.490680	0.000000	0.676122	1.000000	0.000000	2.077903	1.000000
75%	61.750000	1.000000	2.000000	140.097261	273.983074	0.000000	1.000000	165.713572	1.000000	1.744327	2.000000	1.000000	2.970481	1.000000
max	77.000000	1.000000	3.000000	192.020200	406.932689	1.000000	2.000000	202.138041	1.000000	6.157114	2.000000	4.000000	3.277466	1.000000

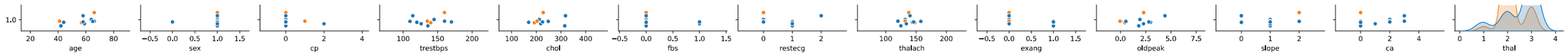
One outlier that we notice here is that thal, which is supposed to be categorical seems to be corrupted by noise which needs to be corrected. We will correct the same by rounding it off to the nearest whole number.

In [35]:
```
fig1 = plt.figure(figsize=(12,12))
sn = sns.pairplot(heart_df_nona,hue='target', dropna=True)
```

```
<Figure size 864x864 with 0 Axes>
```




trestbps
chol
fbs
restecg
thalach
exang
oldpeak
slope
ca
thal
target
0
1



In [36]:
```
thal = [round(x) for x in list(heart_df_nona.loc[:,"thal"].values)]
heart_df_nona["thal"] = thal
heart_df_nona.describe()
```

Out[36]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000	174.000000
mean	54.649425	0.695402	0.948276	132.852160	246.142975	0.137931	0.563218	149.446186	0.367816	1.096423	1.396552	0.747126	2.356322	0.528736
std	9.275368	0.461565	1.015870	18.438982	46.541790	0.345823	0.541912	22.059644	0.483603	1.279873	0.634454	1.077552	0.598206	0.500614
min	29.000000	0.000000	0.000000	93.944184	126.085811	0.000000	0.000000	88.032613	0.000000	-0.176438	0.000000	0.000000	1.000000	0.000000
25%	47.250000	0.000000	0.000000	120.026582	213.267299	0.000000	0.000000	134.528882	0.000000	0.056498	1.000000	0.000000	2.000000	0.000000
50%	55.000000	1.000000	1.000000	130.021392	242.960083	0.000000	1.000000	151.490680	0.000000	0.676122	1.000000	0.000000	2.000000	1.000000
75%	61.750000	1.000000	2.000000	140.097261	273.983074	0.000000	1.000000	165.713572	1.000000	1.744327	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	192.020200	406.932689	1.000000	2.000000	202.138041	1.000000	6.157114	2.000000	4.000000	3.000000	1.000000

Here, we have made corrections in terms of noise fixing in 'thal' attribute