In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.formula.api import ols
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.ensemble import ExtraTreesClassifier

%config Completer.use_jedi = False
%matplotlib inline
```

1 Descripció del Dataset

Aquest dataset es tracta d'un conjunt de pacients en el qual se'ls hi recull un seguit de dades mèdiques i fisiològiques com per exemple edat, sexe, pulsacions màximes etc... i se'ls classifica segons si la probabilitat de tenir un atac de cor és alta (1) o simplement és baixa (0). Aquestes dades son importants per tal de realitzar un algoritme predictiu tal que extretes unes dades del pacient, se'l pugui classificar com a pacient amb risc d'atac de cor (i realitzar les accions oportunes abans que aquest passi) o com a pacient amb un baix risc.

2 Integració i selecció

En descarregar les dades del kaggle recomanat per la pràctica, s'observa que hi han dos datasets diferents. Un amb totes 303 linies on hi ha 14 variables diferents, i un altre on hi ha la saturació d'oxigen amb 3585 linies i 1 variable. La idea principal era integrar aquests dos datasets per tal de formar-ne un de conjunt. No obstant, el dataset de saturació no hi ha cap clau primaria que es pugui relaciónar amb el dataset principal. És per això que finalment no integraré els dos conjunts ja que podria fer associacions falses entre saturació de O2 i les dades de cada pacient.

In [2]:
```
# Pujada del arxiu amb les dades del heart
inicial = pd.read_csv("C:/Users/ricar/OneDrive/Escritorio/Màster Data Science UOC/Assign
print(inicial.shape)
inicial.head()
```

```
(303, 14)
```

Out[2]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

In [3]:
```
# Pujada de l'arxiu amb les dades de la saturació de 02

saturacio = pd.read_csv("C:/Users/ricar/OneDrive/Escritorio/Màster Data Science UOC/Assi
print(saturacio.shape)
saturacio.head()
```

```
(3585, 1)
```

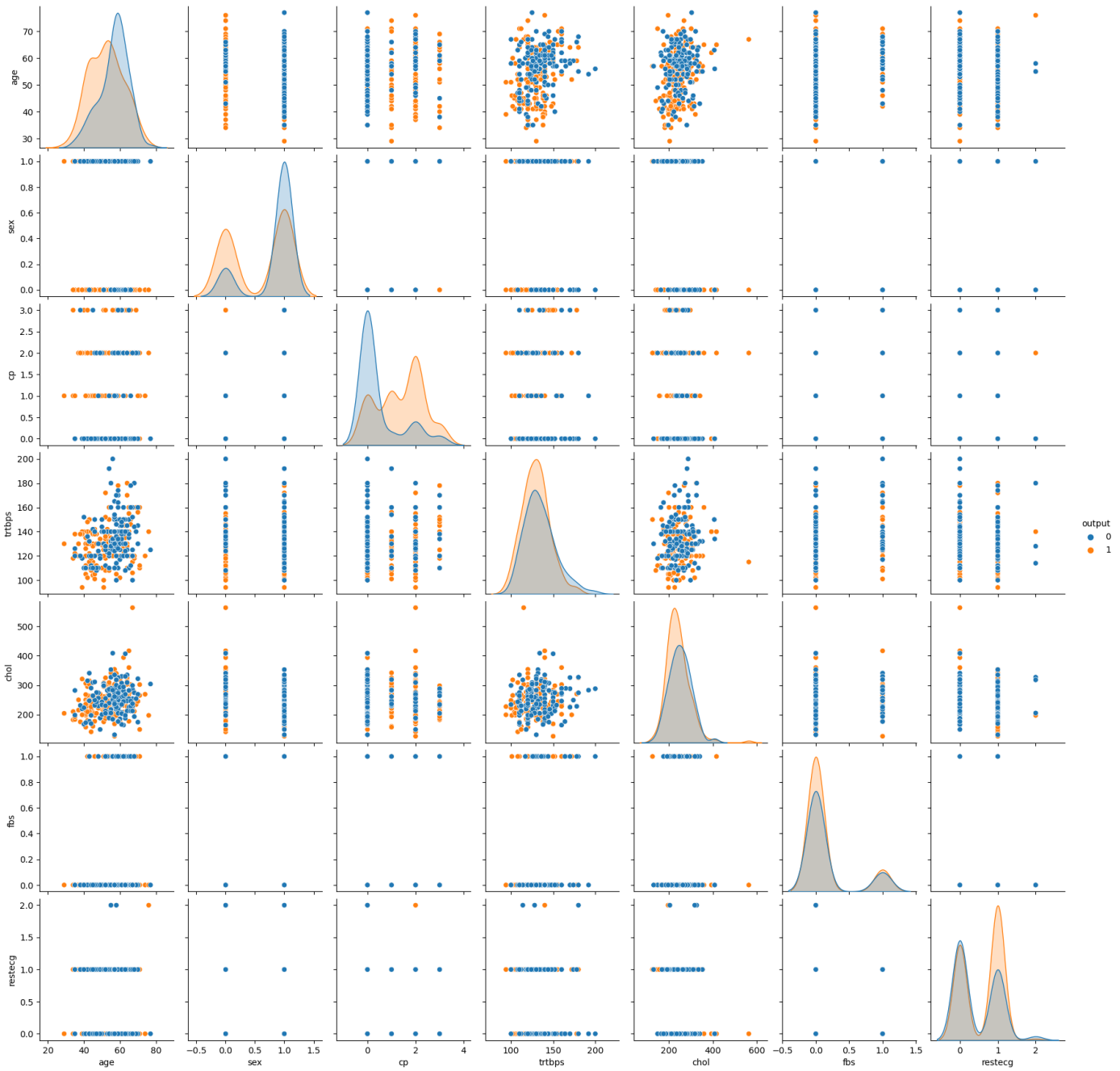
Out[3]:

	98.6
0	98.6
1	98.6
2	98.6
3	98.1
4	97.5

Per tal d'analitzar possibles relacions (tant lineals com no lineals) entre variables i poder descartar aquelles que ens aporten informació redundant, es realitza una matriu de scatterplots amb totes les variables distingides per si tenen probabilitat altes d'atac de cor o no.

In [4]:
```
sub1 = inicial.iloc[:,[0,1,2,3,4,5,6,13]]
sns.pairplot(sub1,hue="output")
# plt.show()
```

Out[4]:
```
<seaborn.axisgrid.PairGrid at 0x1d543a617f0>
```



In [5]:
```
sns.pairplot(inicial.iloc[:,[7,8,9,10,11,12,13]],hue="output")
```

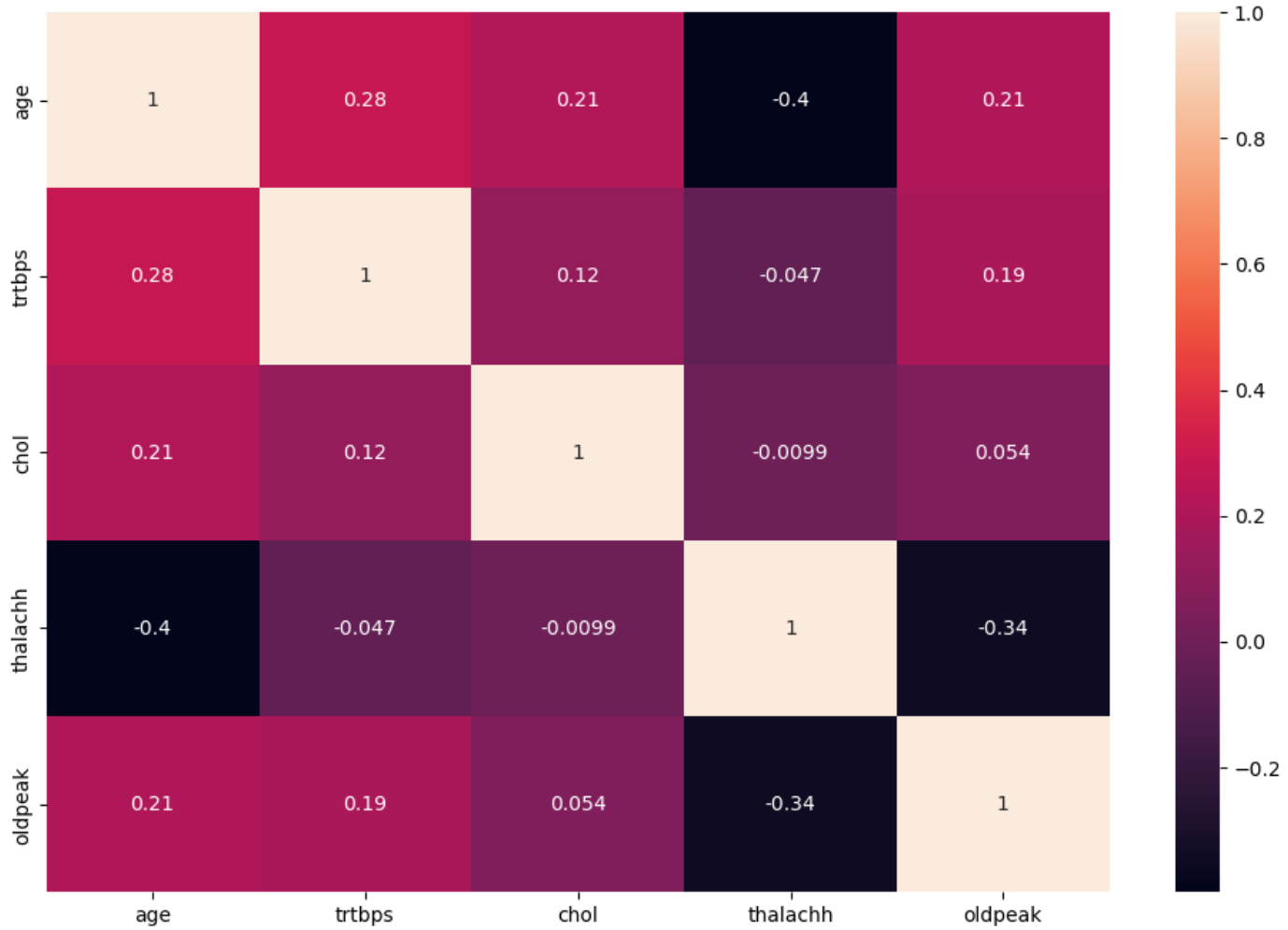
Out[5]:
```
<seaborn.axisgrid.PairGrid at 0x1d547d79d60>
```

Aquest primer gràfic no ens permet veure conclusions molt rellevants per lo que s'hauria de mirar amb més detall. Fent un heatmap de correlació de variables tenim (només agafem les numèriques):

In [6]:
```
numeriques = inicial.drop(columns=["sex","exng","caa","cp","fbs","restecg","output","slp

fig = plt.figure(figsize=(12,8))
sns.heatmap(numeriques.corr(),annot=True)
```

Out[6]:
```
<AxesSubplot:>
```



D'aquest heatmap s'extreuen les següents conclusions: hi ha una relació negativa entre les pulsacions màximes i 'edat, i també hi ha certa relació entre les màximes pulsacions i l'oldpeak, tot i així son correlacions "mitjanes".

A priori, abans de treure els outliers no es descarta cap variable ja que ho farem en apartats posteriors on seleccioanrem les dades més significatives.

3 Neteja de les dades

3.1. Les dades contenen zeros o elements buits? Gestiona cadascun d'aquests casos.

Cap dada conte valors nuls. En cas de contindre, faríem imputació de valors nulls per KNN per exemple

In [7]:
```
inicial.isna().sum()
```

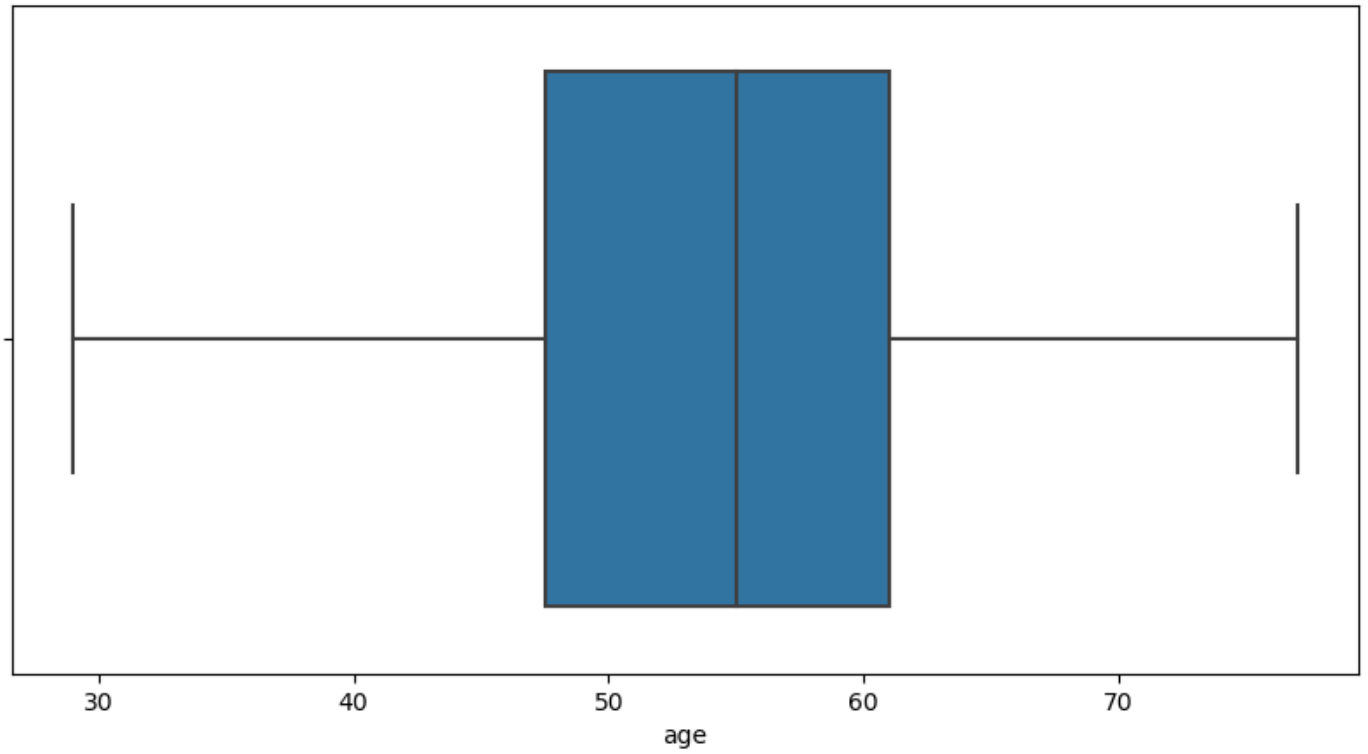
Out[7]:
```
age         0
sex         0
cp          0
trtbps      0
chol        0
fbs         0
restecg     0
thalachh    0
exng        0
oldpeak     0
slp         0
caa         0
```

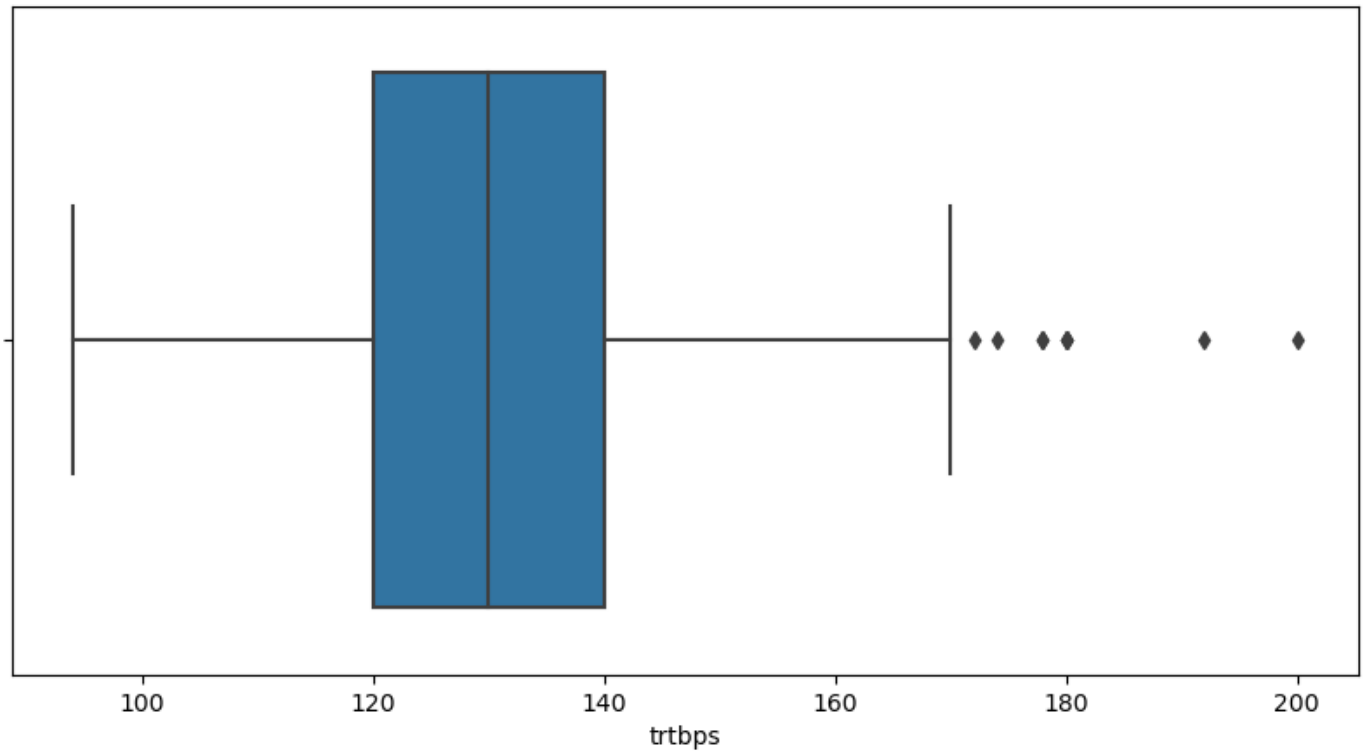
```
thall       0
output      0
dtype: int64
```

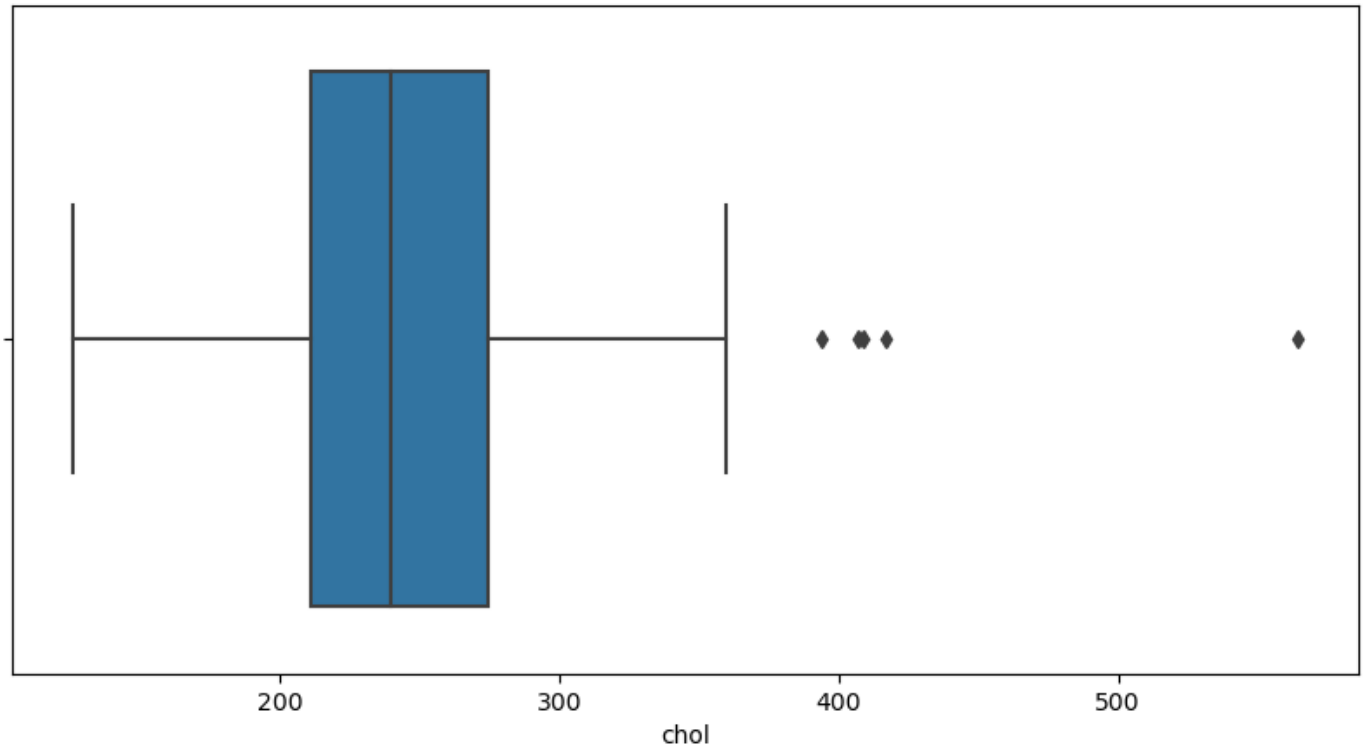
3.2. Identifica i gestiona els valors extrems.

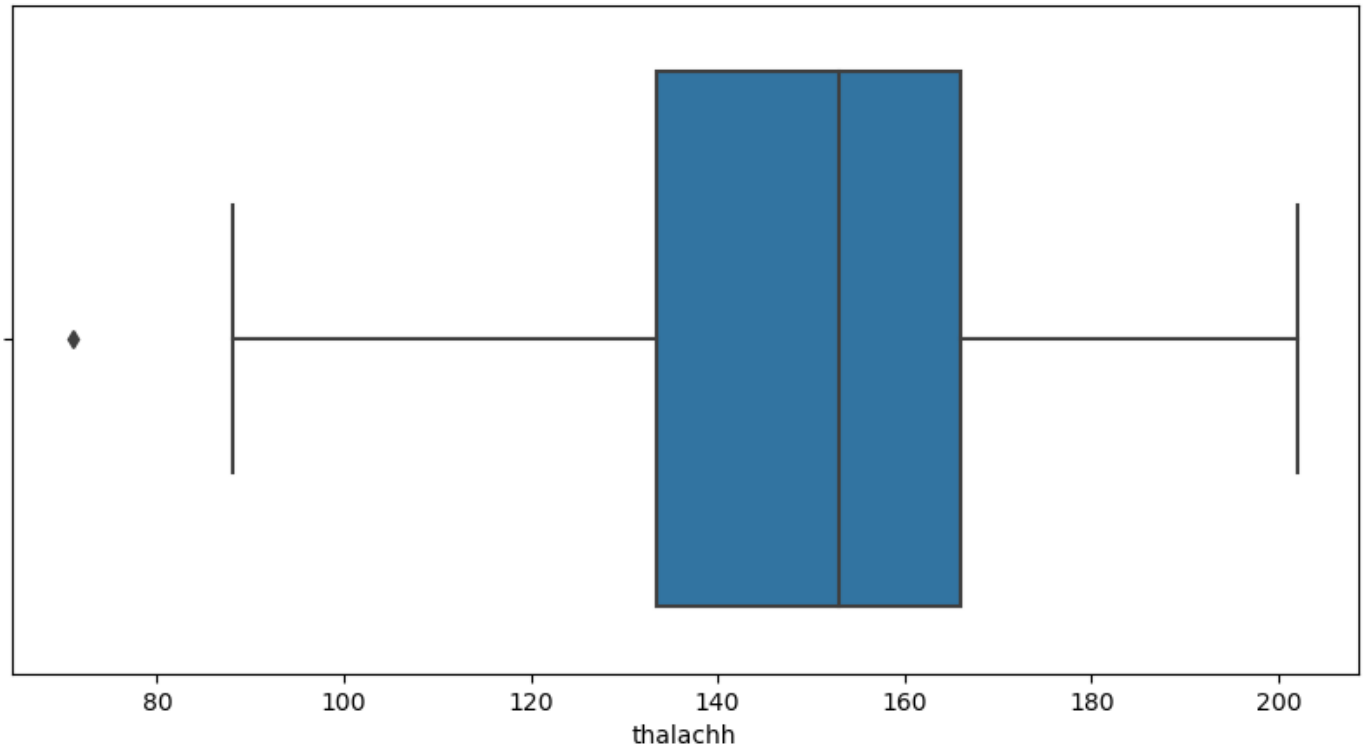
In [8]:
```
display(numeriques.shape)
for feature in numeriques.columns:
    plt.figure(figsize=(10,5))
    sns.boxplot(x=numeriques[feature])
    plt.show()
```

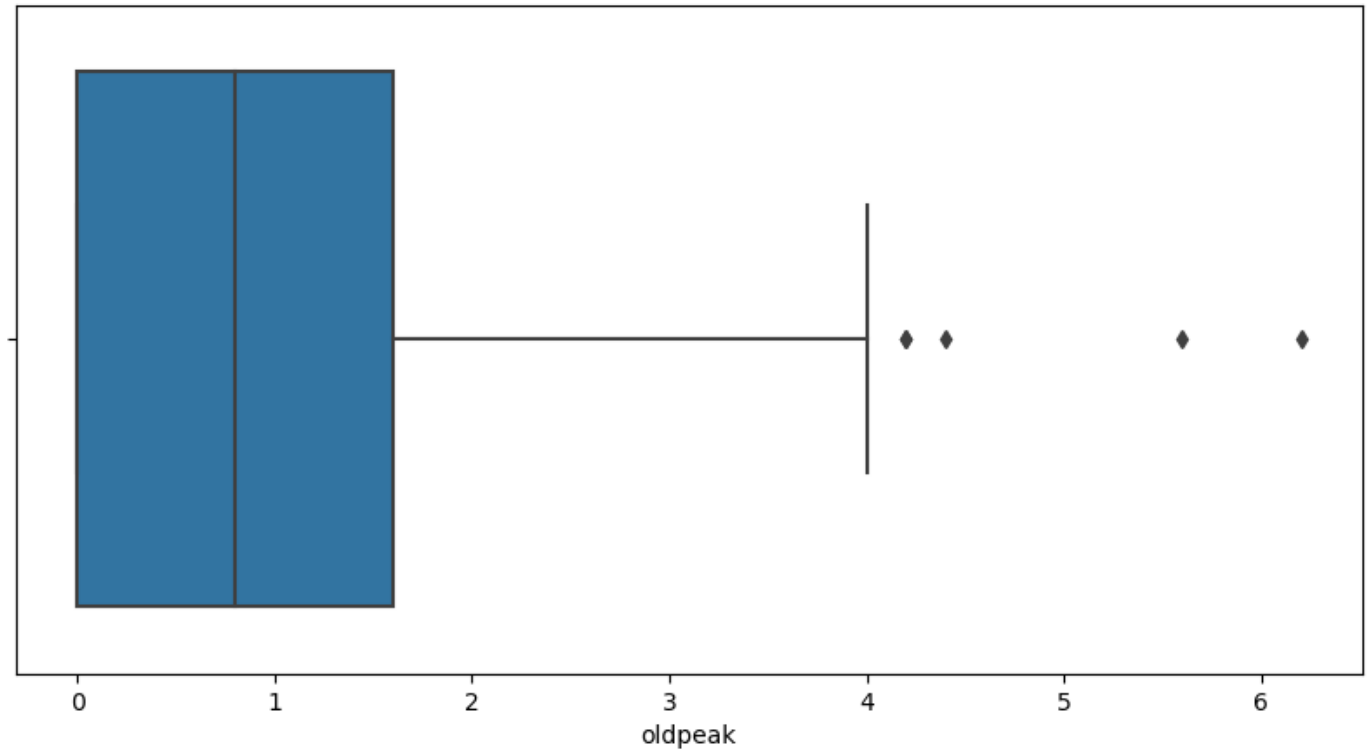
```
(303, 5)
```






200
300
400
500
chol


80
100
120
140
160
180
200
thalachh



Després de fer certes cerques, els valors que ens surten com a outliers en aquests boxplots, no es podria assegurar que ho fossin, ja que son valors poc comuns però tot i així s'han de tindre en compte perque son possibles i indicadors de malalties cardíaques.

4 Anàlisis de les dades

Primerament, es fa un anàlisis de les nostres variables amb algunes mesures de tendència central i disperció, d'aquesta manera també podem veure valors extrems en el min i max si s'allunya molt de la mitjana o dels quartils.

In [9]:
```
d=pd.concat([inicial.describe(),inicial.agg(["skew","mad","kurt","median"])])
d
```

Out[9]:

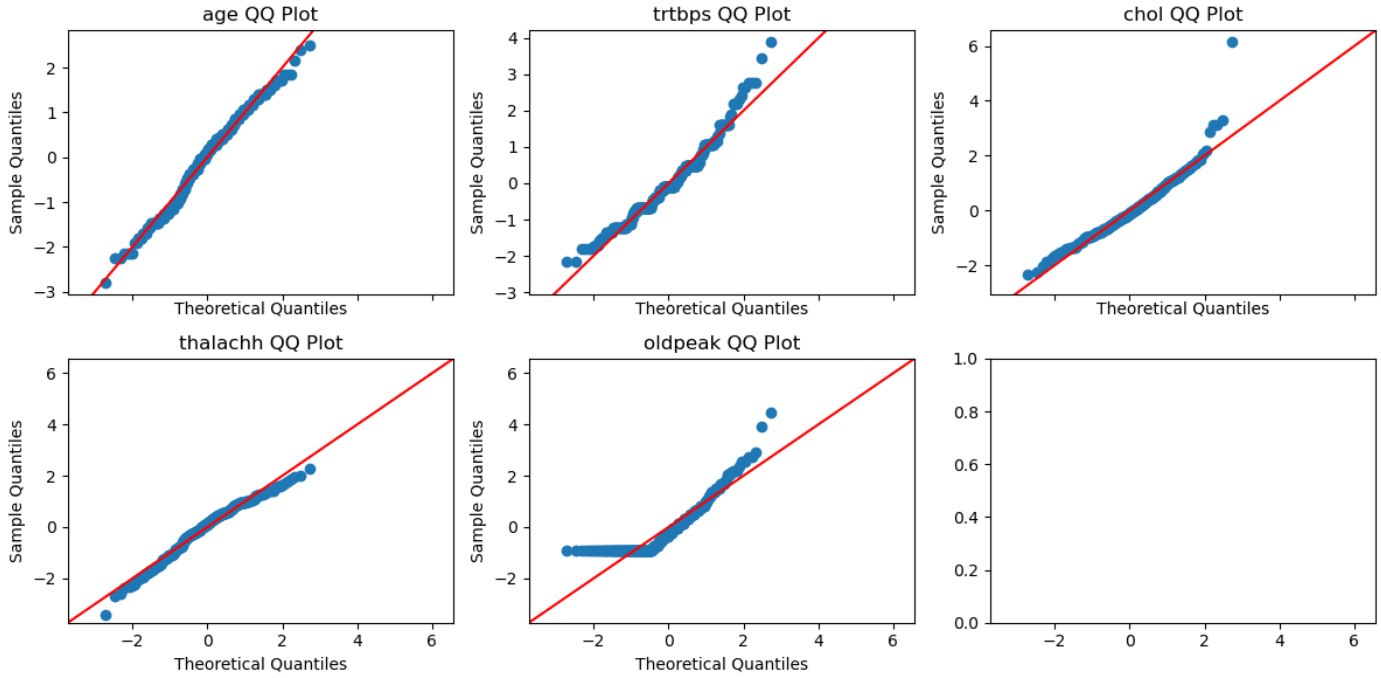
	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exi
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.3267
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.4697
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.0000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.0000
...	...	...	...	...	...	...	...	...	
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.0000
skew	-0.202463	-0.791335	0.484732	0.713768	1.143401	1.986652	0.162522	-0.537410	0.7425
mad	7.457112	0.432899	0.912743	13.522530	39.314141	0.252916	0.512368	18.484397	0.4399
kurt	-0.542167	-1.382961	-1.193071	0.929054	4.505423	1.959678	-1.362673	-0.061970	-1.4583
median	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.0000

A cotninuació es comprova la normalitat i homogenietat de la variança de les variables numèriques:

Com es pot veure, totes segueixen més o menys normalitat (Excepte oldpeak). Tot i així les cues de les distribución o estan sesgades, o els valors "outliers" les fan desviar.

A partir d'aquí, es procedeix a fer una selecció de variables utilitzant un model logístic, on a través de tests de contrast d'hipòtesis per saber si la variable és significativa en relació a l'output i testos de colinealitat entre variables, es determinarà quines son les variables més adequades.

In [10]:
```
fig, axes = plt.subplots(ncols=3, nrows=2, sharex=True, figsize=(4*3, 2*3))
for k, ax in zip(numeriques.columns, np.ravel(axes)):
    sm.qqplot(numeriques[k], line='45',fit = True, ax=ax)
    ax.set_title(f'{k} QQ Plot')
plt.tight_layout()
```



A primera vista, detectem variables com age, trtbps, chol, fbs, restcg, i slp que ens donen pvalors majors de 0.05 per lo que podríem interpretar que no son significatives. Tot i així, per creure aquests pvalors, s'hauria d'assegurar que no hi ha colinealitat entre les variables. Es revisa la colinealitat. Es fan les següents iteracions per anar descartant les variables (VIF >5):

Iteració 1 (es decarta trtbps):

	Variables	VIF
0	age	38.998305
1	sex	3.523211
2	cp	2.414403
3	trtbps	58.557335
4	chol	26.267365
5	fbs	1.268205
6	restecg	2.058206
7	thalachh	42.742178
8	exng	2.022527
9	oldpeak	3.062890
10	slp	10.072734
11	caa	1.808925
12	thall	17.165303

Iteració 2 (es descarta thalachh):

Variables	VIF
age	27.213596
sex	3.412645
cp	2.264790
chol	22.374105
fbs	1.248307
restecg	2.022210
exng	1.955987
oldpeak	2.965697
slp	8.372679
caa	1.760648
thall	15.963052

Iteració 3 (es descarta age):

	Variables	VIF
0	sex	3.404301
1	cp	2.211674
2	chol	14.917083
3	fbs	1.232428
4	restecg	2.006017
5	exng	1.947640
6	oldpeak	2.827322
7	slp	7.983150
8	caa	1.710828
9	thall	14.860553

Iteració 4 (es descarta chol):

	Variables	VIF
0	sex	3.348022
1	cp	2.154649
2	fbs	1.223893
3	restecg	2.005247
4	exng	1.915642
5	oldpeak	2.628953
6	slp	6.201591
7	caa	1.707658
8	thall	10.478889

Iteració 5 (es descarta thall):

	Variables	VIF
0	sex	3.038525
1	cp	2.112149
2	fbs	1.222870
3	restecg	1.957971
4	exng	1.795800
5	oldpeak	2.062440
6	slp	3.756703
7	caa	1.686170

A partir de la iteració 5, ja no tenim colinealitat entre les variables dependents, això vol dir que ens podem creure els contrastos d'hipòtesis del model i per tant les variables amb pvalors superiors a 0.05 són les que ens quedarem com a significatives. En aquest cas, eliminarem la variable fbs, ja que te un coeficient p-valor

de 0.6 i restcg que té un p-valor de 0.052.

	coef	std err	z	P>|z|	[0.025	0.975]
sex	-1.3762	0.373	-3.685	0.000	-2.108	-0.644
cp	0.9375	0.172	5.459	0.000	0.601	1.274
fbs	0.2581	0.492	0.524	0.600	-0.707	1.223
restecg	0.6033	0.311	1.940	0.052	-0.006	1.213
exng	-1.1637	0.361	-3.227	0.001	-1.871	-0.457
oldpeak	-0.5488	0.173	-3.179	0.001	-0.887	-0.210
slp	0.9698	0.212	4.574	0.000	0.554	1.385
caa	-0.8273	0.178	-4.655	0.000	-1.176	-0.479

In [11]:
```
# es separen les variables target (output) i la resta de variables independents
y=inicial.output
X = inicial.iloc[:,:-1]
X.drop(columns=["trtbps","thalachh","age","chol","thall","fbs","restecg"],inplace=True)
#fem un fit del model i veiem els resultats

logit_model=sm.Logit(y,X)
result=logit_model.fit()
print(result.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.394730
         Iterations 7
                           Logit Regression Results
==============================================================================
Dep. Variable:                 output   No. Observations:                  303
Model:                          Logit   Df Residuals:                      297
Method:                           MLE   Df Model:                            5
Date:                Sun, 08 Jan 2023   Pseudo R-squ.:                  0.4272
Time:                        20:51:36   Log-Likelihood:                -119.60
converged:                       True   LL-Null:                       -208.82
Covariance Type:            nonrobust   LLR p-value:                 1.157e-36
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sex           -1.3276      0.370     -3.590      0.000      -2.052      -0.603
cp             0.9541      0.167      5.699      0.000       0.626       1.282
exng          -1.0747      0.353     -3.042      0.002      -1.767      -0.382
oldpeak       -0.4905      0.167     -2.934      0.003      -0.818      -0.163
slp            1.1034      0.199      5.543      0.000       0.713       1.494
caa           -0.8004      0.174     -4.587      0.000      -1.142      -0.458
==============================================================================
```

In [12]:
```
#colinealitat

VIF_modelg = pd.DataFrame()
VIF_modelg["Variables"] = X.columns

#Es calculen els VIF per a les variables:

VIF_modelg['VIF'] = [variance_inflation_factor(X, i) for i in range(X.shape[1])]
print(VIF_modelg.to_string())
```

```
  Variables       VIF
0       sex  3.028078
1        cp  2.030867
2      exng  1.781966
3   oldpeak  2.007561
4       slp  3.191513
5       caa  1.637583
```

Finalment el grup de variables que s'han seleccionat són:

- sex
- cp
- exng
- oldpeak
- slp
- caa

Per a comprovar amb un altre mètode quines son les variables que prenen més importancia a l'hora de predir l'output, s'utilitza un ExtraTreesClassifier de sklearn per treure els "feature*importances*".

En aquest cas, si treguéssim les variables que amb un anàlisi de colinealitat ens donarien com a que son colineals, obtindríem el mateix resultat que en l'anterior mètode. Tot i així, els arbres de decisió no estan afectats per la colinealitat per tant, si féssim una predicció a través d'un mètode d'aquest tipus, es podrien seleccionar les següents variables (en cas de agafar-ne 6)

- caa
- cp
- exng
- thall
- oldpeak
- thalachh

In [13]:
```
X = inicial.iloc[:,:-1]
variables = X.columns
# s'extreuen les variables per importancia
model = ExtraTreesClassifier(n_estimators=100)
model.fit(X,y)
print(sorted(list(zip(variables,model.feature_importances_)),key= lambda x:x[1],reverse=
```

```
[('cp', 0.13834054782072447), ('caa', 0.12748266917345202), ('exng', 0.103597501752459
5), ('thall', 0.0937349737953743), ('oldpeak', 0.08683721872831603), ('thalachh', 0.0819
3736413386579), ('age', 0.0694440932964989), ('slp', 0.06548301498794187), ('trtbps', 0.
06285573078682934), ('chol', 0.0622101940973673), ('sex', 0.05385171853079835), ('restec
g', 0.03363610836733481), ('fbs', 0.020588864529037264)]
```

Resolució del problema

A partir dels resultats obtinguts, es podria conlcuir que les dades son útils per respondre a la pregunta de si a traves de les dades recollides, es pot detectar si un pacient és més propens a patir un atac de cor o no. Pel que fa a la selecció de variables, en funció de les hipòtesis que han de complir uns mètodes o altres, la selecció de variables hauria de ser diferent com s'ha pogut veure amb un mètode logístic, on les variables explicatives han de ser linealment independents i seguir normalitat en canvi amb un mètode d'arbres de decisió no colinealitat no afecta degut a que es realitza bootstraping en el dataset.