In [44]:
```
#Importing the necessary libraries:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

In [45]:
```
#Importing the csv file
df = pd.read_csv("Wine_quality.csv")
#Printing the first 5 rows
df.head()
```

Out[45]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4

In [46]:
```
#rows and columns
df.shape
```

Out[46]:
```
(1599, 12)
```

In [47]:
```
#checking for null values
df.isna().sum()
```

Out[47]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    1
density                 0
pH                      1
sulphates               0
alcohol                 0
quality                 1
dtype: int64
```

In [48]:
```
#Particular rows with null values
print(df[df.isnull().any(axis = 1)])
```

```
     fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
9              7.5              0.50         0.36             6.1      0.071
123            8.0              0.71         0.00             2.6      0.080
184            6.7              0.62         0.21             1.9      0.079

     free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
9                   17.0                   NaN   0.9978  3.35       0.80
123                 11.0                  34.0   0.9976  3.44       0.53
184                  8.0                  62.0   0.9970   NaN       0.58

     alcohol  quality
9       10.5      5.0
123      9.5      NaN
184      9.3      6.0
```

In [49]:
```
#renaming the "total sulphur dioxide" column for convinience
df.rename(columns = {'total sulfur dioxide':'TotalSulphurDioxide'}, inplace = True
```

In [55]:
```
#Removing the row with null value in the quality column
df = df.dropna(subset=['quality'])
#Replacing the null values with the mean values of the particular column
df.pH.fillna(df.pH.mean(), inplace=True)
df.TotalSulphurDioxide.fillna(df.TotalSulphurDioxide.mean(), inplace=True)
```

In [56]:
```
df.isna().sum()
#All the null values are hence removed
```

Out[56]:
```
fixed acidity          0
volatile acidity       0
citric acid            0
residual sugar         0
chlorides              0
free sulfur dioxide    0
TotalSulphurDioxide    0
density                0
pH                     0
sulphates              0
alcohol                0
quality                0
dtype: int64
```

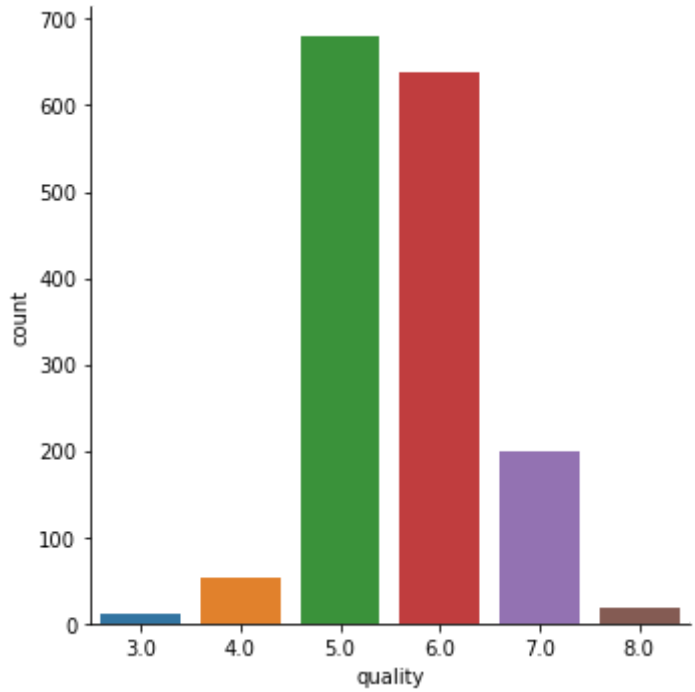
In [57]:
```
#Data analysis and visualization
#Statistical measures
df.describe()
```

Out[57]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	TotalSulphur
count	1598.000000	1598.000000	1598.000000	1598.000000	1598.000000	1598.000000	159
mean	8.319837	0.527707	0.271145	2.538767	0.087471	15.877972	4
std	1.741623	0.179058	0.194744	1.410369	0.047080	10.462720	3
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	2
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	3
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	6
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	28

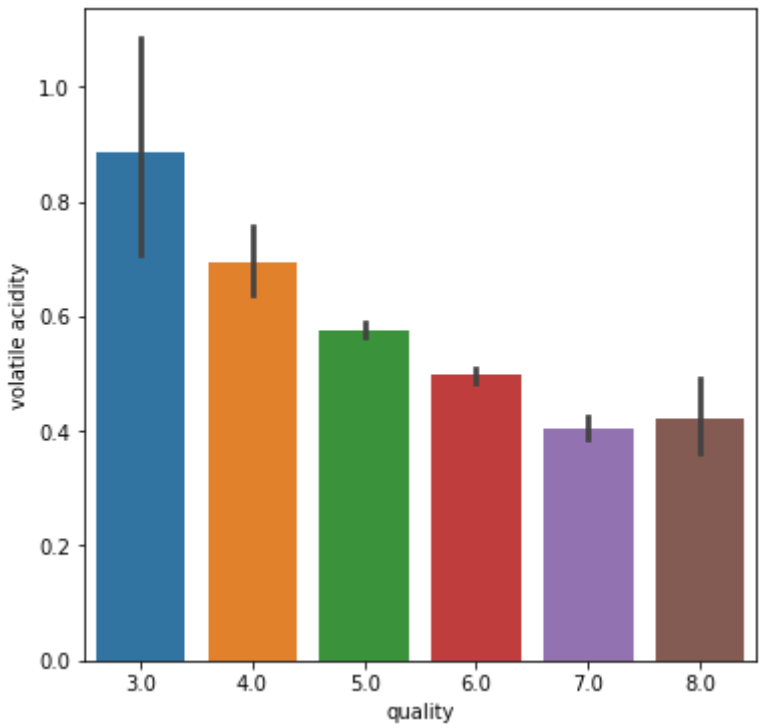
In [58]:
```
#number of values for each quality
sns.catplot(x ='quality', data = df , kind = 'count' )
```

Out[58]:
```
<seaborn.axisgrid.FacetGrid at 0x18f77754f10>
```



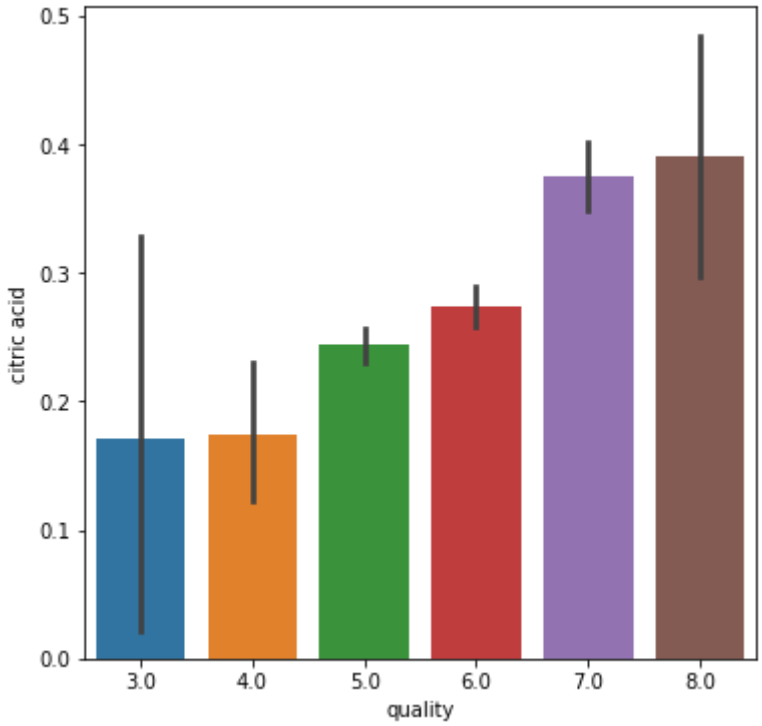
In [59]:
```
#volatile acidity and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "volatile acidity", data = df)
```

Out[59]:
```
<AxesSubplot:xlabel='quality', ylabel='volatile acidity'>
```



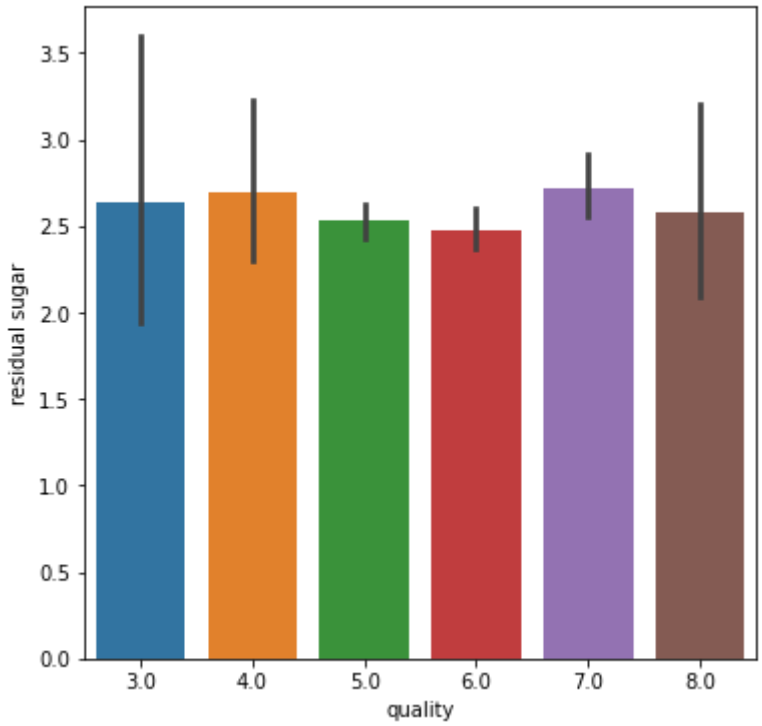
In [60]:
```
#citric acid and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "citric acid", data = df)
```

Out[60]:
```
<AxesSubplot:xlabel='quality', ylabel='citric acid'>
```



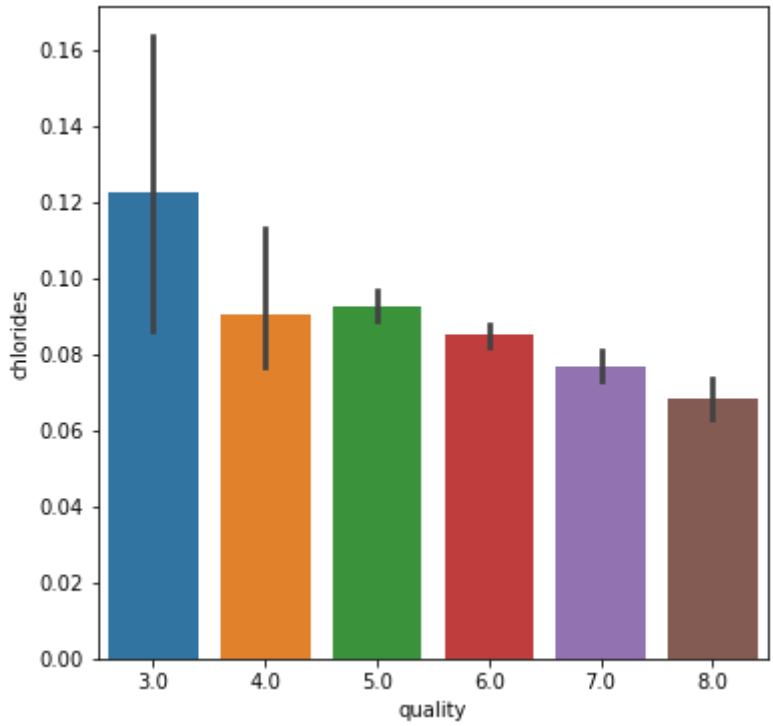
In [61]:
```
#residual sugar and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "residual sugar", data = df)
```

Out[61]:
```
<AxesSubplot:xlabel='quality', ylabel='residual sugar'>
```



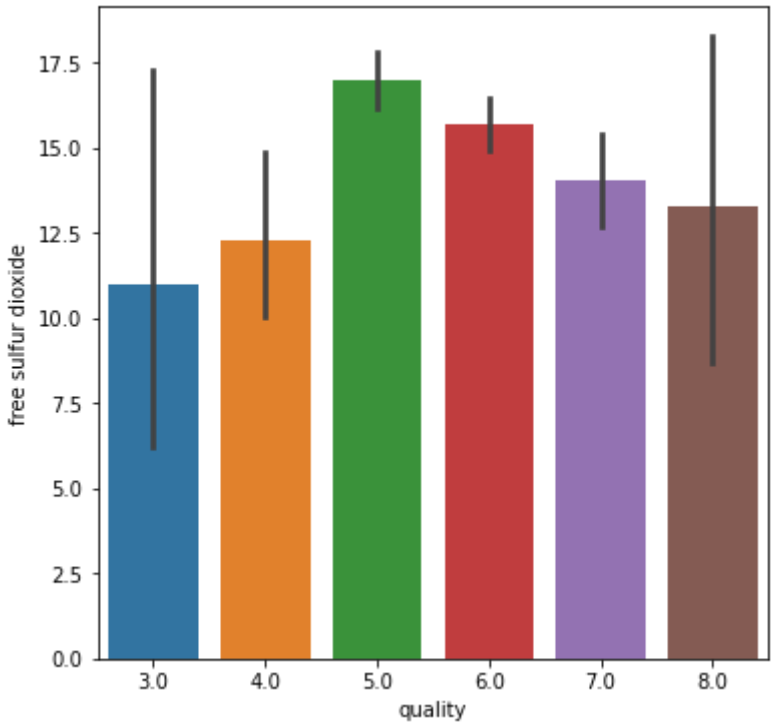
In [62]:
```
#chlorides and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "chlorides", data = df)
```

Out[62]:
```
<AxesSubplot:xlabel='quality', ylabel='chlorides'>
```



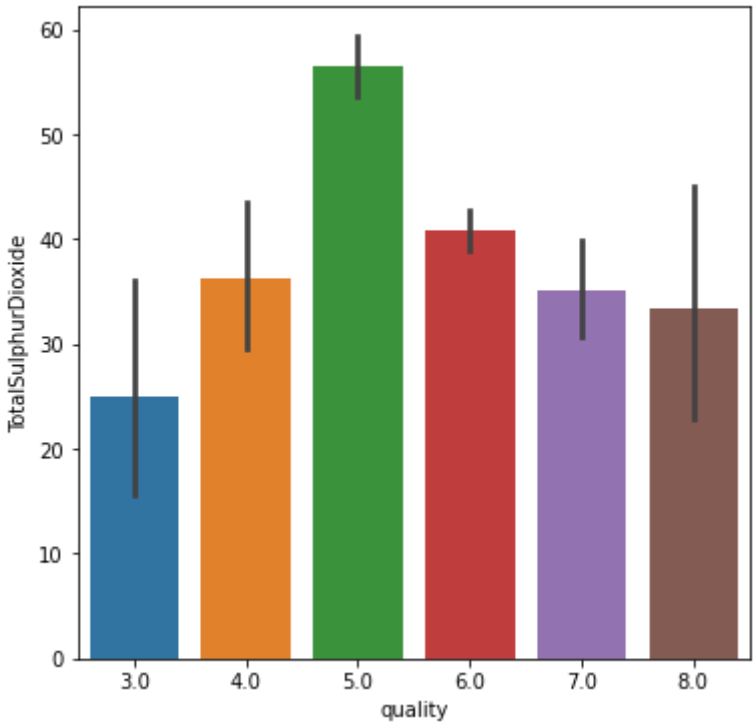
In [65]:
```
#free sulphur dioxide and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "free sulfur dioxide", data = df)
```

Out[65]:
```
<AxesSubplot:xlabel='quality', ylabel='free sulfur dioxide'>
```



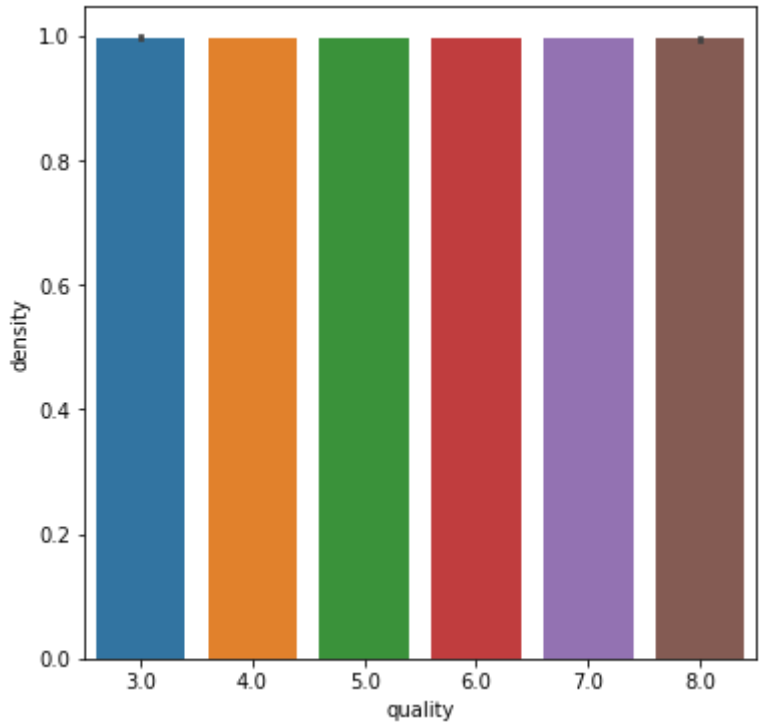
In [67]:
```
#TotalSulphurDioxide and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "TotalSulphurDioxide", data = df)
```

Out[67]:
```
<AxesSubplot:xlabel='quality', ylabel='TotalSulphurDioxide'>
```



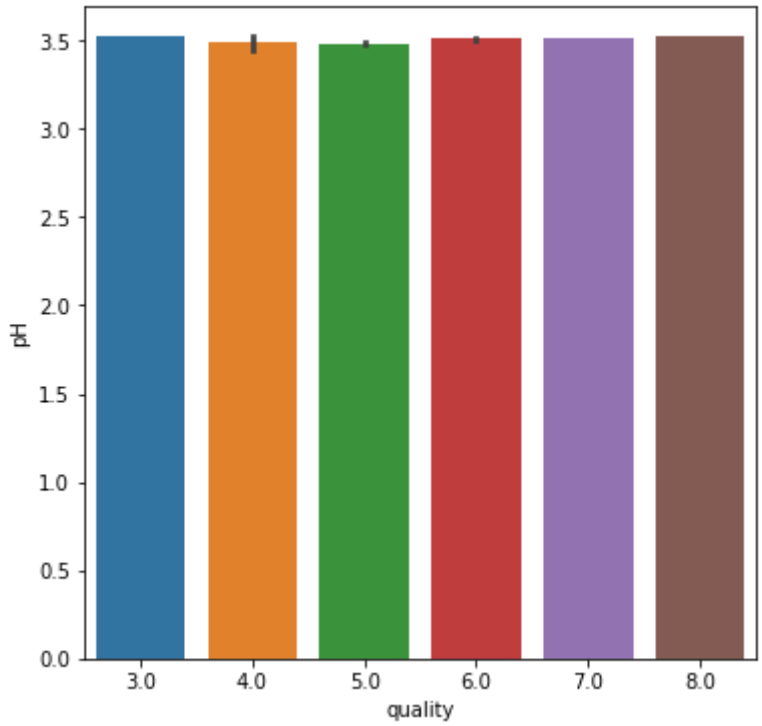
In [68]:
```
#density and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "density", data = df)
```

Out[68]:
```
<AxesSubplot:xlabel='quality', ylabel='density'>
```



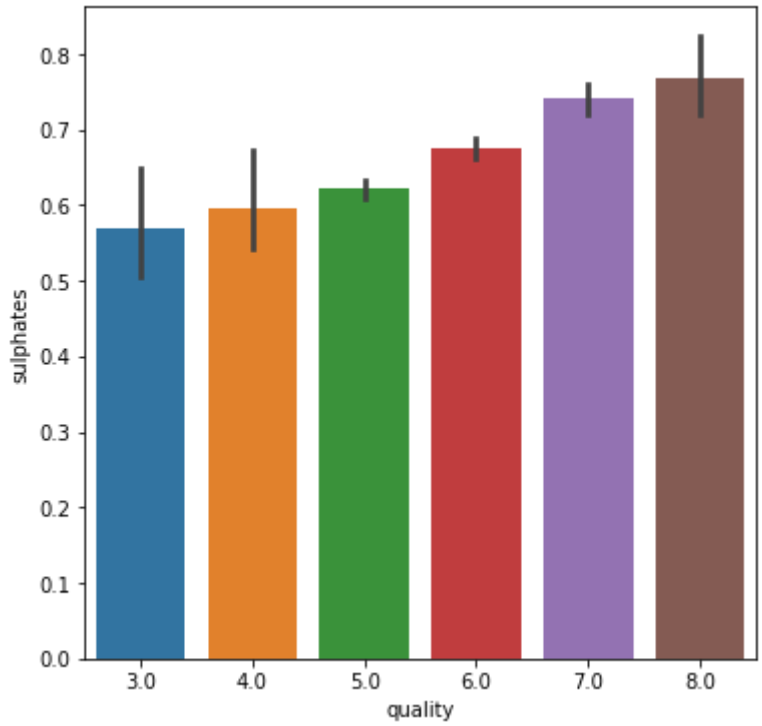
In [69]:
```
#pH and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "pH", data = df)
```

Out[69]:
```
<AxesSubplot:xlabel='quality', ylabel='pH'>
```



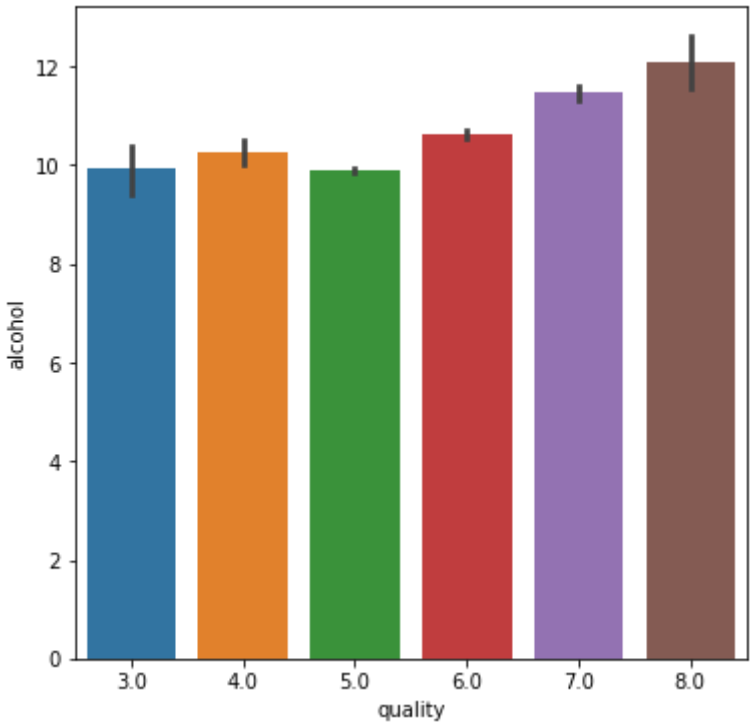
In [70]:
```
#sulphates and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "sulphates", data = df)
```

Out[70]:
```
<AxesSubplot:xlabel='quality', ylabel='sulphates'>
```



In [72]:
```
#alcohol and quality
plot = plt.figure(figsize = (6,6))
sns.barplot(x = 'quality', y = "alcohol", data = df)
```

Out[72]:
```
<AxesSubplot:xlabel='quality', ylabel='alcohol'>
```



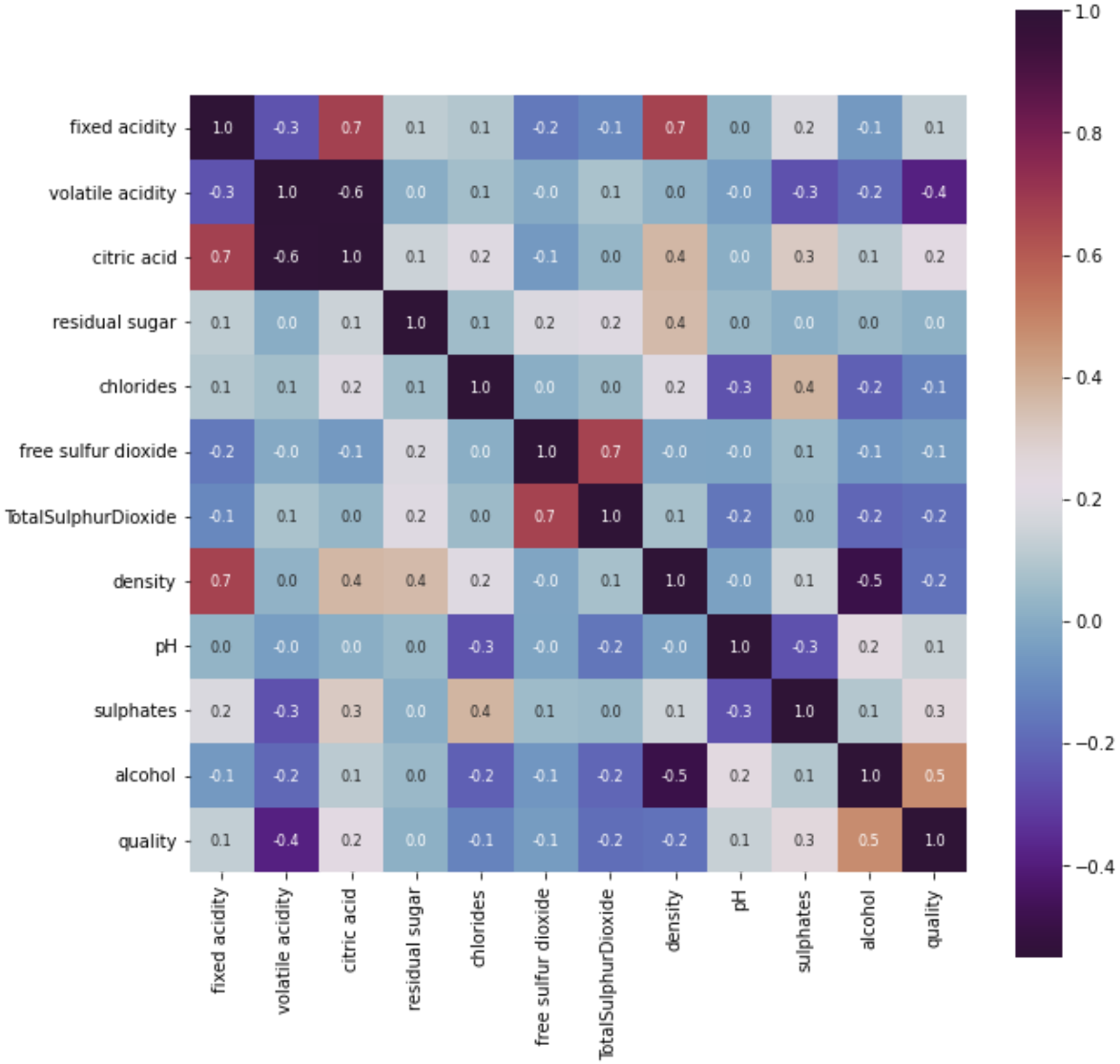
In [82]:
```
#Finding the correlation of all the columns to the quality column
correlation = df.corr()
correlation
```

Out[82]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	TotalSulphurD
fixed acidity	1.000000	-0.256100	0.671958	0.114783	0.093689	-0.153860	-0.
volatile acidity	-0.256100	1.000000	-0.552123	0.001891	0.061419	-0.010211	0.(
citric acid	0.671958	-0.552123	1.000000	0.143702	0.203810	-0.061425	0.(
residual sugar	0.114783	0.001891	0.143702	1.000000	0.055614	0.187074	0.2
chlorides	0.093689	0.061419	0.203810	0.055614	1.000000	0.005516	0.(
free sulfur dioxide	-0.153860	-0.010211	-0.061425	0.187074	0.005516	1.000000	0.(
TotalSulphurDioxide	-0.112834	0.076972	0.034774	0.200557	0.047778	0.668114	1.(
density	0.668149	0.021747	0.365586	0.355294	0.200692	-0.021817	0.(
pH	0.027466	-0.046010	0.007785	0.040037	-0.258759	-0.025604	-0.
sulphates	0.182953	-0.260636	0.312357	0.005549	0.371255	0.051450	0.(
alcohol	-0.061783	-0.201849	0.109241	0.042109	-0.221280	-0.069682	-0.2
quality	0.123987	-0.390258	0.225867	0.013756	-0.129011	-0.050899	-0.

In [95]:
```
#constructing a heatmap to understand the correlation between the columns
plt.figure(figsize = (10,10))
sns.heatmap(correlation,cmap = "twilight_shifted", cbar = True, square = True, fmt
```

Out[95]:
```
<AxesSubplot:>
```



In [ ]:
```
#Data preprocessing
```

In [91]:
```
#separate the data and the label
X = df.drop('quality',axis = 1)
X
```

Out[91]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	TotalSulphurDioxide	density	pH	sul
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	
...	...	...	...	...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.52	
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.52	
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.52	
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.52	

1598 rows × 11 columns

In [92]:
```
#Label Binarization
Y = df['quality'].apply(lambda y_value:1 if y_value>=7 else 0)
Y
```

Out[92]:
```
0       0
1       0
2       0
3       0
4       0
       ..
1594    0
1595    0
1596    0
1597    0
1598    0
Name: quality, Length: 1598, dtype: int64
```

In [93]:
```
#Train and test Split
X_train, X_test, Y_train,Y_test = train_test_split(X,Y,test_size = 0.2, random_sta
```

In [94]:
```
print(Y.shape,Y_train.shape,Y_test.shape)
```

```
(1598,) (1278,) (320,)
```

In [96]:
```
#Model Training
#Random Forest Classifier model
model = RandomForestClassifier()
```

In [97]:
```
model.fit(X_train, Y_train)
```

Out[97]:
```
RandomForestClassifier()
```

In [98]:
```
#Model Evaluation
#accuracy on test data
```

```
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

In [100…

```
print("Accuracy: " , test_data_accuracy)
```

```
Accuracy:  0.903125
```

In [105…

```
#Building a predictive system
input_data = (7.9,0.6,0.06,1.6,0.069,15,59,0.9964,3.3,0.46,9.4)

#changing the input data to a numpy array
input_data_np = np.asarray(input_data)

#reshape the data as we are predicting the label for only one instance
input_data_reshaped = input_data_np.reshape(1,-1)

prediction = model.predict(input_data_reshaped)
print(prediction)

if prediction[0] == 1:
    print("Good quality Wine")
else:
    print("Bad quality Wine")
```

```
[0]
Bad quality Wine
C:\Users\Anujit\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X do
es not have valid feature names, but RandomForestClassifier was fitted with featur
e names
  warnings.warn(
```

In [ ]: