In [4]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:

```
df = pd.read_csv(r"C:\Users\Maj Mortuza\Downloads\supermarket_sales.csv")
```

In [6]:

```
df.head(10)
```

Out[6]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7.0	26.1415	548.
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5.0	3.8200	80.
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7.0	16.2155	340.
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8.0	23.2880	489.
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7.0	30.2085	634.
5	699-14-3026	C	Naypyitaw	Normal	Male	Electronic accessories	85.39	7.0	29.8865	627.
6	355-53-5943	A	Yangon	Member	Female	NaN	68.84	6.0	20.6520	433.
7	315-22-5665	C	Naypyitaw	Normal	Female	NaN	73.56	10.0	36.7800	772.
8	665-32-9167	A	Yangon	Member	Female	NaN	36.26	2.0	3.6260	76.
9	692-92-5582	B	Mandalay	Member	Female	NaN	54.84	3.0	8.2260	172.

In [7]:

```
df.dtypes
```

Out[7]:

```
Invoice ID                 object
Branch                     object
City                       object
Customer type              object
Gender                     object
Product line               object
Unit price                float64
Quantity                  float64
Tax 5%                    float64
Total                     float64
Date                       object
Time                       object
Payment                    object
cogs                      float64
gross margin percentage   float64
gross income              float64
Rating                    float64
dtype: object
```

In [8]:

```
df['Date']
```

Out[8]:

```
0        1/5/19
1        3/8/19
2        3/3/19
3       1/27/19
4        2/8/19
         ...
998     2/22/19
999     2/18/19
1000    2/18/19
1001    3/10/19
1002    1/26/19
Name: Date, Length: 1003, dtype: object
```

In [9]:

```
df['Date'] = pd.to_datetime(df['Date'])
```

In [11]:

```
df['Date']
```

Out[11]:

```
0      2019-01-05
1      2019-03-08
2      2019-03-03
3      2019-01-27
4      2019-02-08
          ...
998    2019-02-22
999    2019-02-18
1000   2019-02-18
1001   2019-03-10
1002   2019-01-26
Name: Date, Length: 1003, dtype: datetime64[ns]
```

In [10]:

```
df.dtypes
```

Out[10]:

```
Invoice ID                       object
Branch                           object
City                             object
Customer type                    object
Gender                           object
Product line                     object
Unit price                      float64
Quantity                        float64
Tax 5%                          float64
Total                           float64
Date                     datetime64[ns]
Time                             object
Payment                          object
cogs                            float64
gross margin percentage         float64
gross income                    float64
Rating                          float64
dtype: object
```

In [12]:

```
#Put date as index column

df.set_index('Date', inplace = True)
```

In [14]:

```
df.head()
```

Out[14]:

Date	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%
2019-01-05	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7.0	26.1415
2019-03-08	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5.0	3.8200
2019-03-03	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7.0	16.2155
2019-01-27	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8.0	23.2880
2019-02-08	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7.0	30.2085

In [15]:

```
df.describe()
```

Out[15]:

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	in
count	996.000000	983.000000	1003.000000	1003.000000	1003.000000	1.003000e+03	1003.0
mean	55.764568	5.501526	15.400368	323.407726	308.007358	4.761905e+00	15.4
std	26.510165	2.924673	11.715192	246.019028	234.303836	6.131488e-14	11.7
min	10.080000	1.000000	0.508500	10.678500	10.170000	4.761905e+00	0.5
25%	33.125000	3.000000	5.894750	123.789750	117.895000	4.761905e+00	5.8
50%	55.420000	5.000000	12.096000	254.016000	241.920000	4.761905e+00	12.0
75%	78.085000	8.000000	22.539500	473.329500	450.790000	4.761905e+00	22.5
max	99.960000	10.000000	49.650000	1042.650000	993.000000	4.761905e+00	49.6

In [45]:

```
df.duplicated()
```

Out[45]:

```
Date
2019-01-05    False
2019-03-08    False
2019-03-03    False
2019-01-27    False
2019-02-08    False
              ...
2019-02-22    False
2019-02-18    False
2019-02-18     True
2019-03-10     True
2019-01-26     True
Length: 1003, dtype: bool
```

In [46]:

```
#Let's investigate duplicated rows

df[df.duplicated()==True]
```

Out[46]:

Date	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	
2019-02-18	849-09-3807	A	Yangon	Member	Female	Fashion accessories	88.34	7.0	30.919	6
2019-03-10	745-74-0715	A	Yangon	Normal	Male	Electronic accessories	NaN	2.0	5.803	1
2019-01-26	452-04-8808	B	Mandalay	Normal	Male	Electronic accessories	87.08	NaN	30.478	6

In [48]:

```
df.drop_duplicates(inplace=True)
```

In [49]:

```
df.duplicated().sum()
```

Out[49]:

```
0
```

In [50]:

```
#Check for missing value

df.isna().sum()
```

Out[50]:

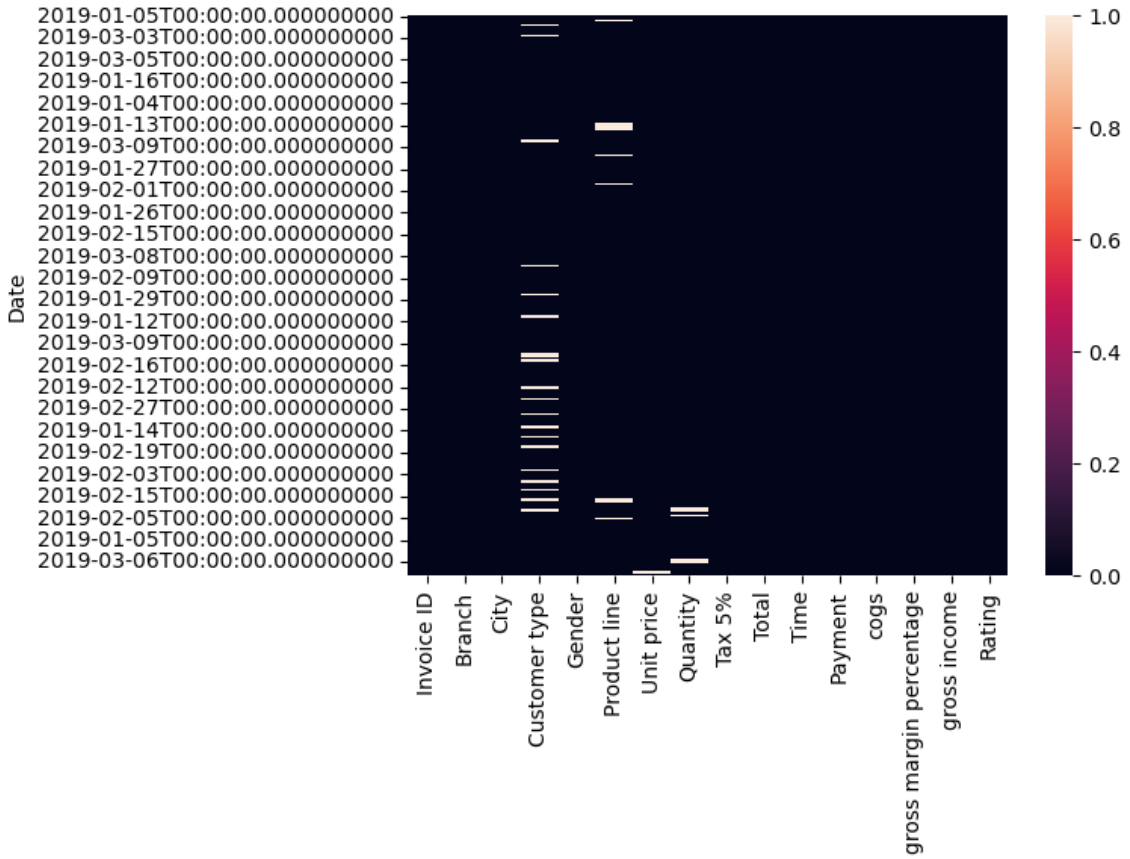
```
Invoice ID                 0
Branch                     0
City                       0
Customer type             79
Gender                     0
Product line              43
Unit price                 6
Quantity                  19
Tax 5%                     0
Total                      0
Time                       0
Payment                    0
cogs                       0
gross margin percentage    0
gross income               0
Rating                     0
dtype: int64
```

In [51]:

```
sns.heatmap(df.isnull())
```

Out[51]:

```
<Axes: ylabel='Date'>
```



In [56]:

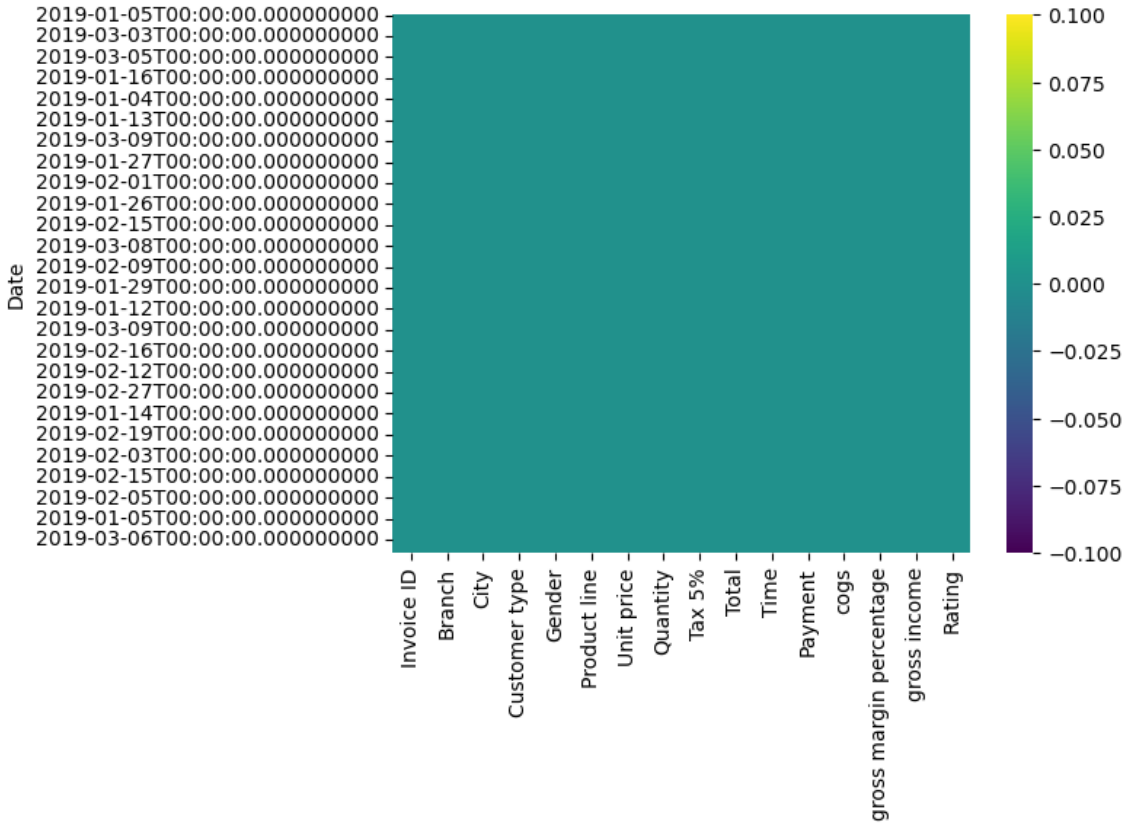
```
df.fillna(df.mean(numeric_only=True), inplace=True)
```

In [55]:

```
df.fillna(df.mode().iloc[0], inplace=True)
```

In [61]:

```
sns.heatmap(df.isnull(), cmap='viridis')
plt.show()
```



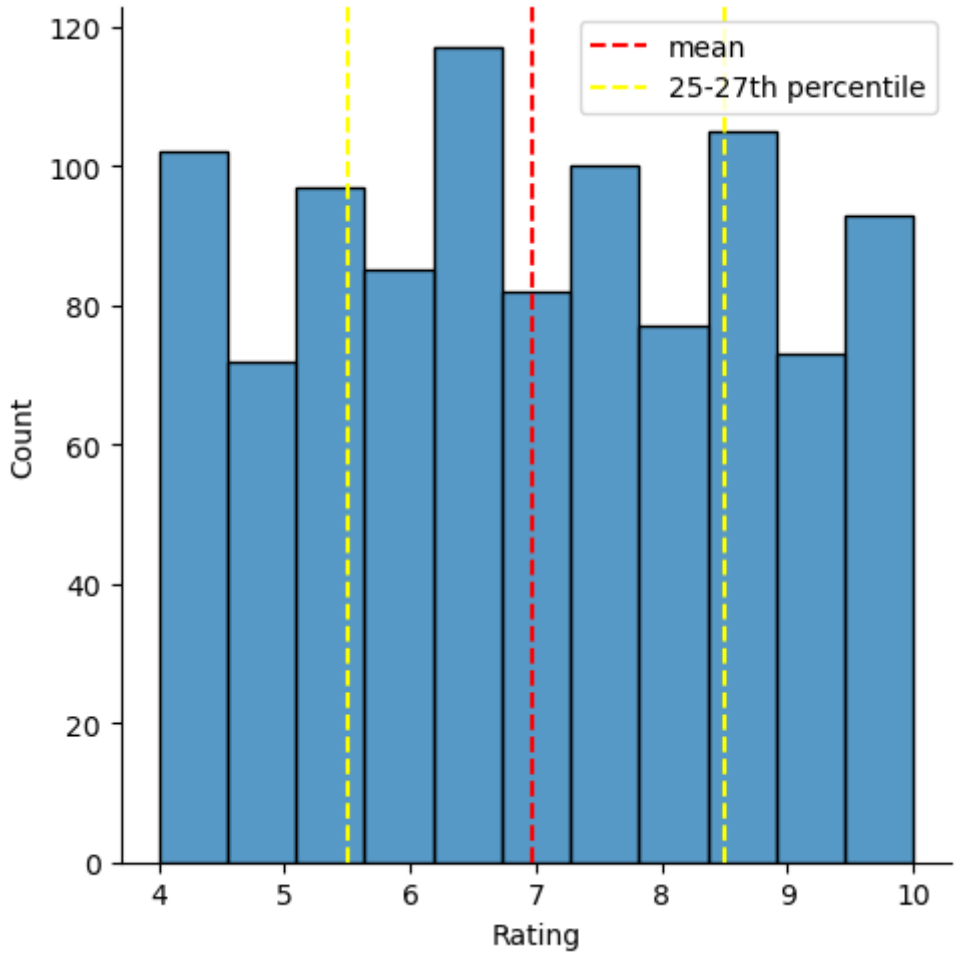
In [28]:

```
#Let's see how the distribution of customer ratings look like and find meand & 25-75th p

sns.displot(df['Rating'])
plt.axvline(x=np.mean(df['Rating']), c = 'red', ls = '--', label = 'mean')
plt.axvline(x=np.percentile(df['Rating'], 25), c = 'yellow', ls = '--', label = '25-27th
plt.axvline(x=np.percentile(df['Rating'], 75), c = 'yellow', ls = '--')
plt.legend()
```

Out[28]:

```
<matplotlib.legend.Legend at 0x1dcc0533110>
```

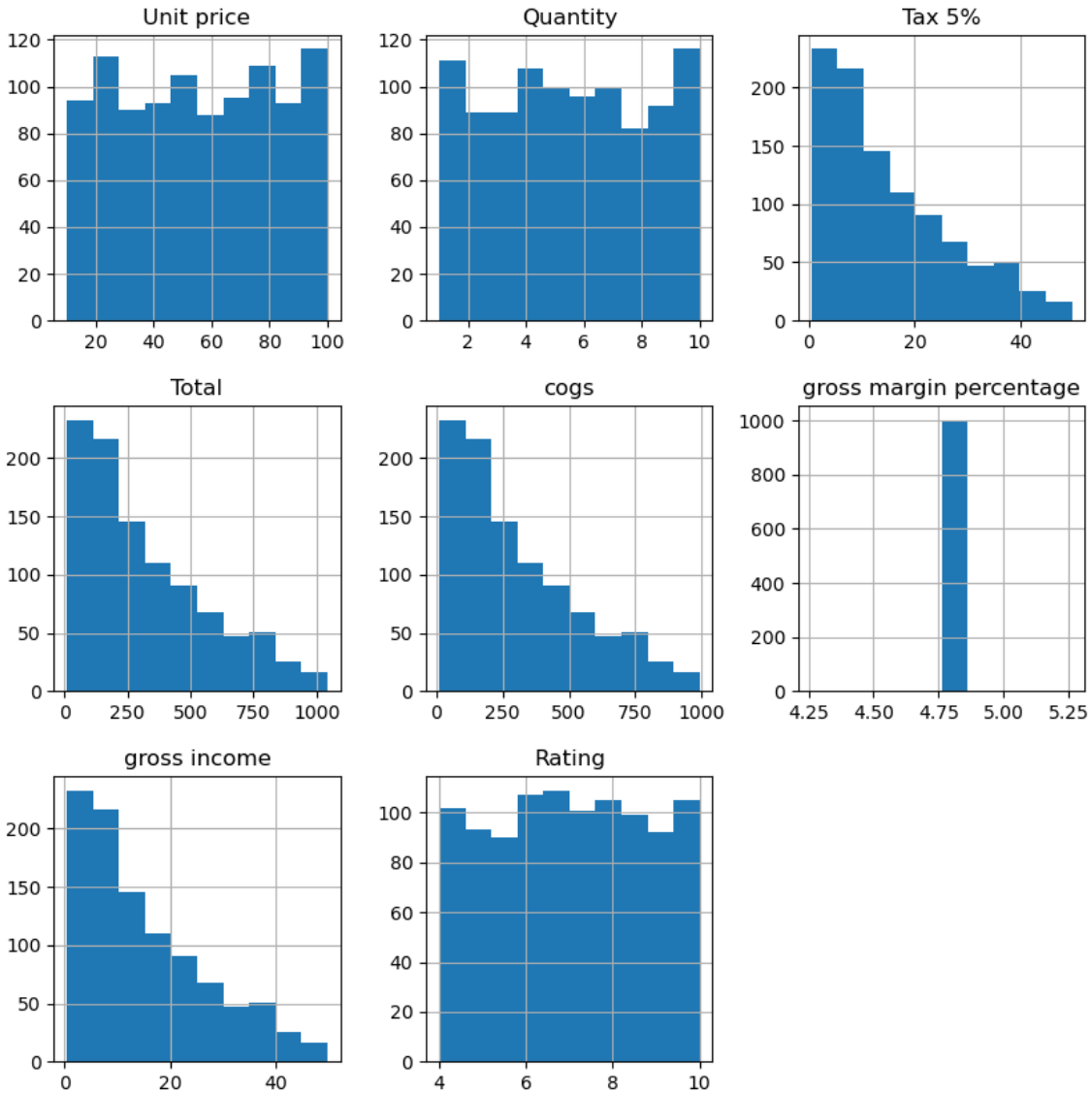


In [31]:

```
df.hist(figsize=(10,10))
```

Out[31]:

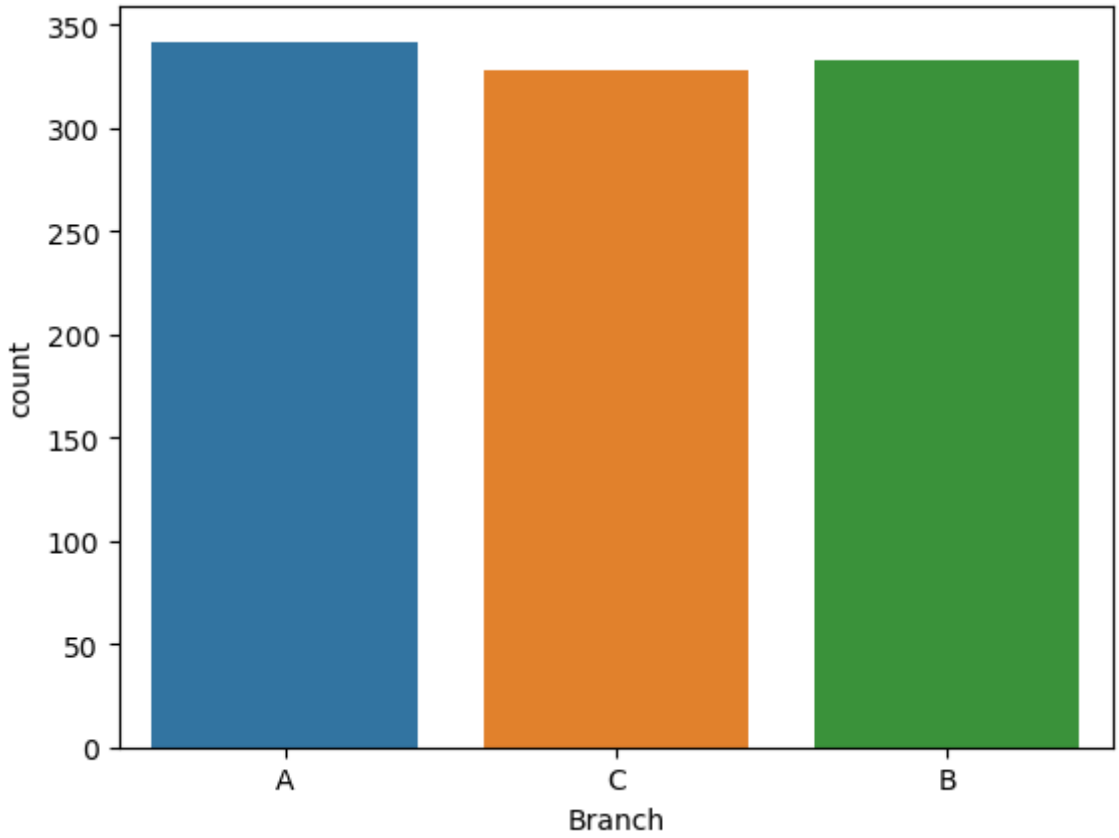
```
array([[<Axes: title={'center': 'Unit price'}>,
        <Axes: title={'center': 'Quantity'}>,
        <Axes: title={'center': 'Tax 5%'}>],
       [<Axes: title={'center': 'Total'}>,
        <Axes: title={'center': 'cogs'}>,
        <Axes: title={'center': 'gross margin percentage'}>],
       [<Axes: title={'center': 'gross income'}>,
        <Axes: title={'center': 'Rating'}>, <Axes: >]], dtype=object)
```



In [33]:

```
#Let's check whether sales in different branches

sns.countplot(x='Branch', data=df)
plt.show()
```



In [36]:

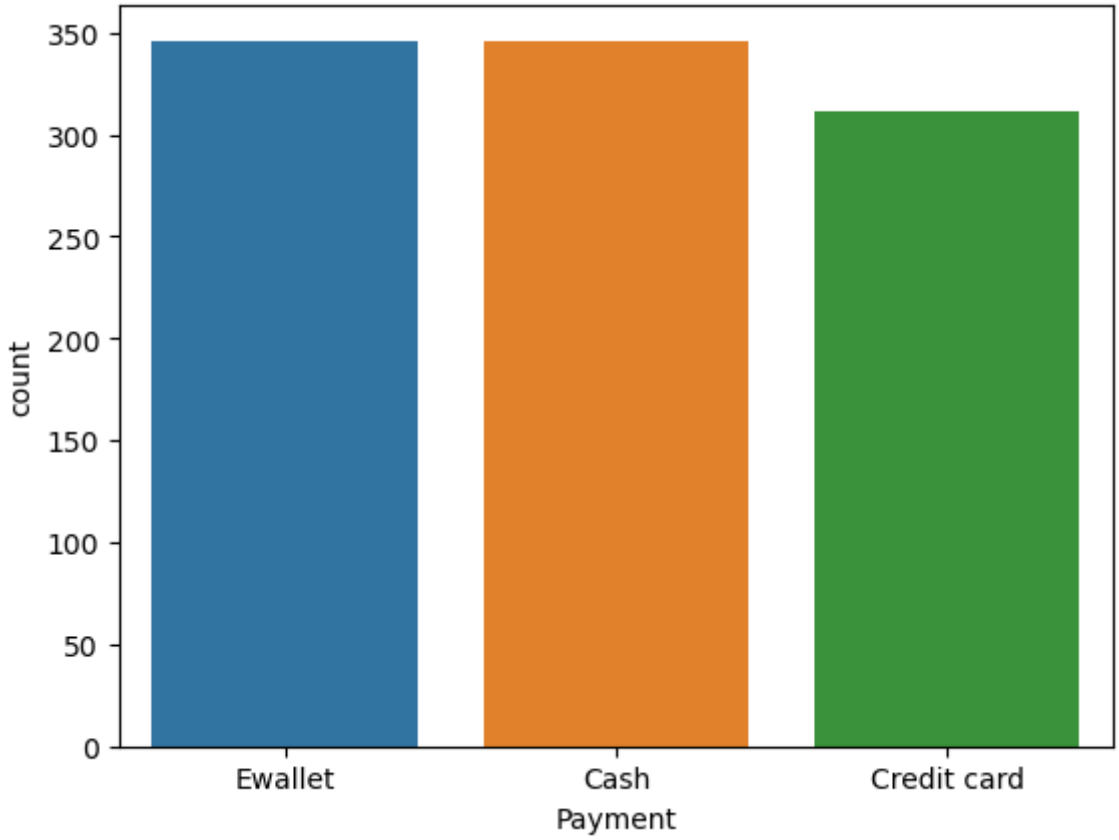
```
df['Branch'].value_counts()
```

Out[36]:

```
A    342
B    333
C    328
Name: Branch, dtype: int64
```

In [37]:

```
sns.countplot(x='Payment', data=df)
plt.show()
```

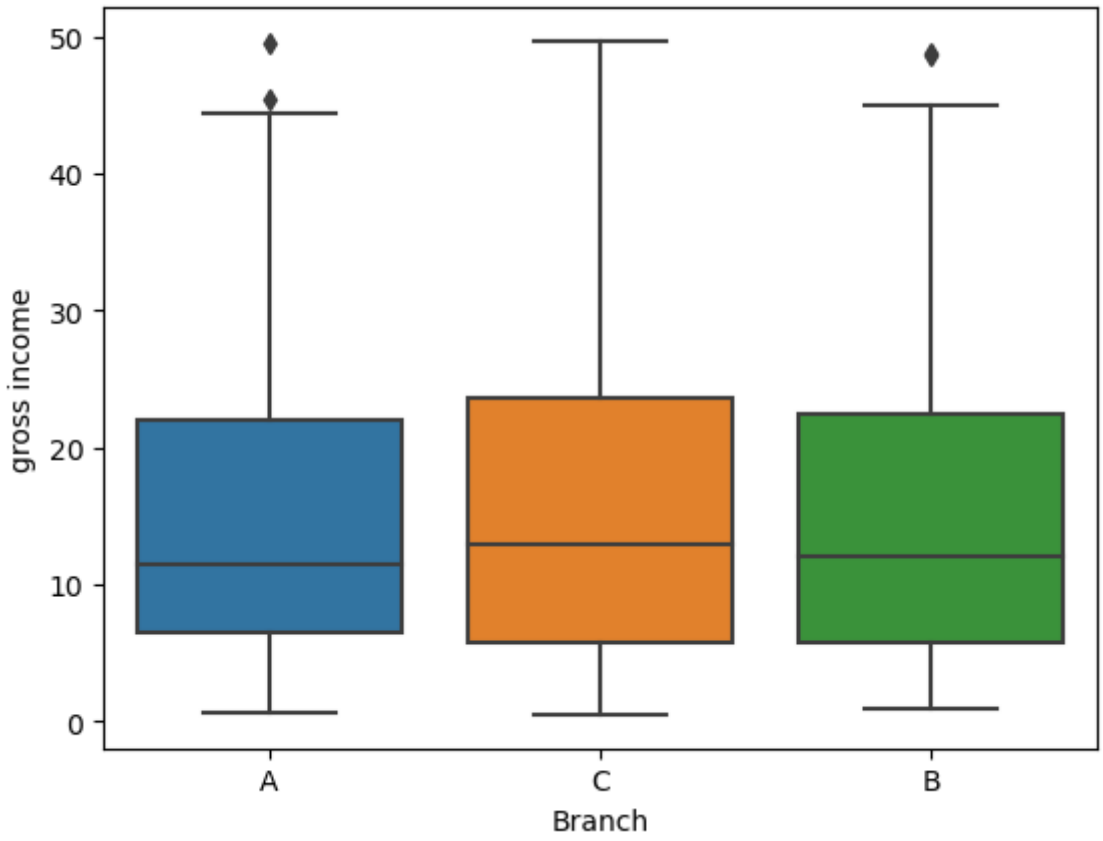


In [38]:

```
sns.boxplot(x=  df['Branch'], y = df['gross income'])
```

Out[38]:

```
<Axes: xlabel='Branch', ylabel='gross income'>
```

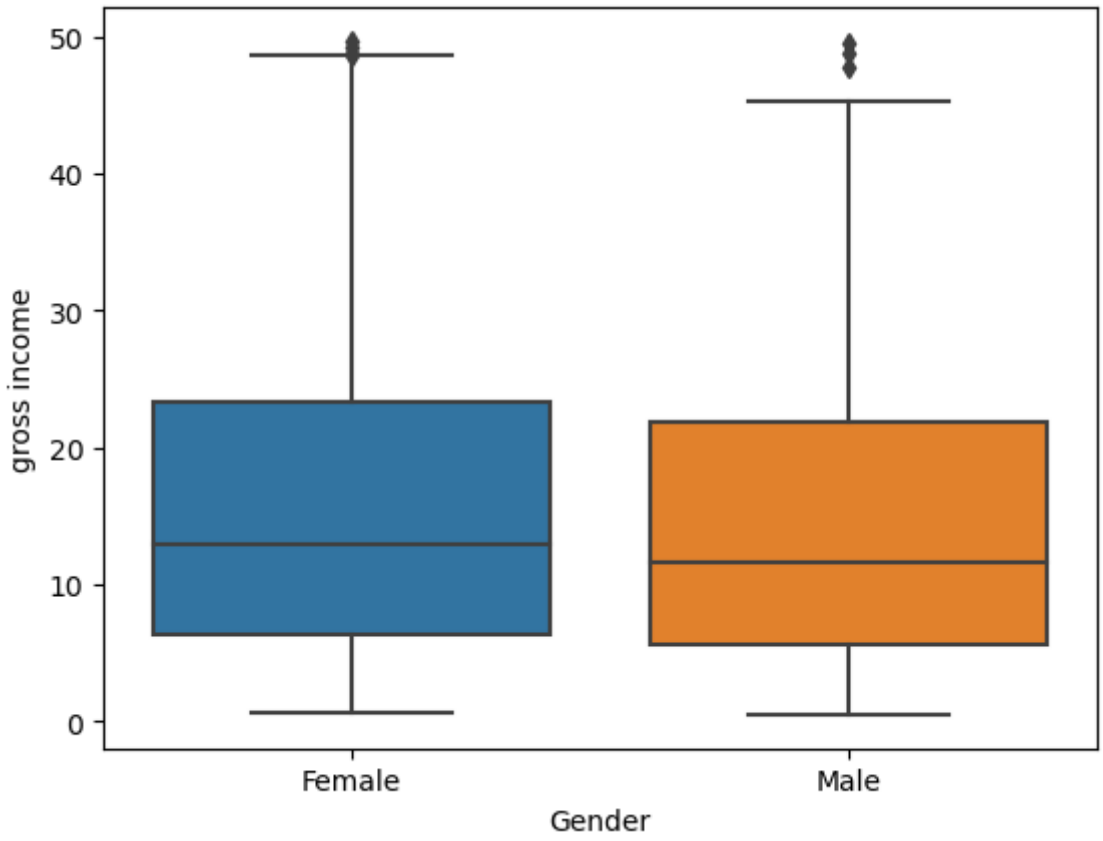


In [39]:

```
sns.boxplot(x=  df['Gender'], y = df['gross income'])
```

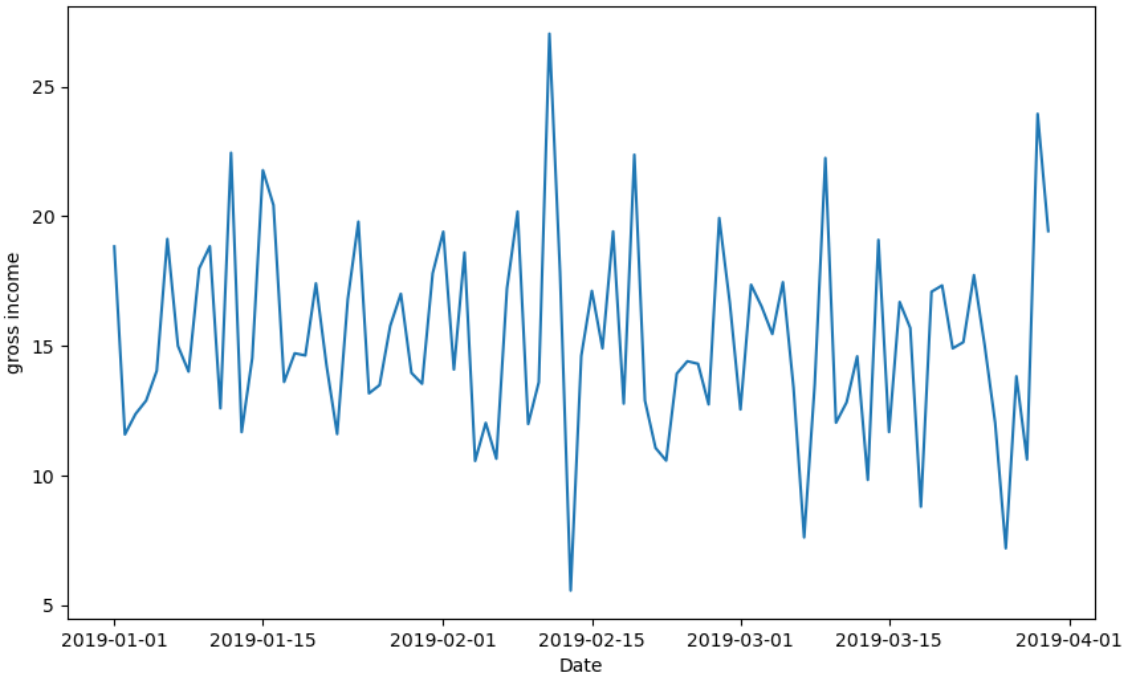
Out[39]:

```
<Axes: xlabel='Gender', ylabel='gross income'>
```



In [42]:

```
df_mean = df.groupby(df.index).mean(numeric_only=True)
plt.figure(figsize=(10, 6))
sns.lineplot(x=df_mean.index, y=df_mean['gross income'])
plt.show()
```



In [74]:

```
corr_matrix = df.corr(numeric_only=True)
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, cmap='coolwarm', center=0, annot=True, fmt='.2f', linewidths=0.
plt.show()
```