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

```python
csv = pd.read_csv('sales data.csv')
csv.head()
```

	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	i
0	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26
1	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3
2	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	4.761905	16
					Health and										

Next steps: Generate code with `csv` | View recommended plots | New interactive sheet

```python
excel = pd.read_excel('Sales data.xlsx')
excel.head()
```

	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	i
0	A	Yangon	Normal	Male	Electronic accessories	51.69	7	18.0915	379.9215	1/26/2019	18:22	Cash	361.83	4.761905	18
1	B	Mandalay	Member	Female	Fashion accessories	54.73	7	19.1555	402.2655	3/14/2019	19:02	Credit card	383.11	4.761905	19
2	B	Mandalay	Member	Male	Home and lifestyle	27.00	9	12.1500	255.1500	3/2/2019	14:16	Cash	243.00	4.761905	12
3	C	Naypyitaw	Normal	Female	Electronic accessories	30.24	1	1.5120	31.7520	3/4/2019	15:44	Cash	30.24	4.761905	1

Next steps: Generate code with `excel` | View recommended plots | New interactive sheet

```python
json = pd.read_json('sales data.json')
json.head()
```

	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	gr inc
701	B	Mandalay	Normal	Male	Food and beverages	32.32	3	4.8480	101.8080	2019-03-27	19:11	Credit card	96.96	4.761905	4.8
702	B	Mandalay	Member	Female	Fashion accessories	19.77	10	9.8850	207.5850	2019-02-27	18:57	Credit card	197.70	4.761905	9.8
703	B	Mandalay	Member	Male	Health and beauty	80.47	9	36.2115	760.4415	2019-01-06	11:18	Cash	724.23	4.761905	36.2
704	B	Mandalay	Member	Female	Home and lifestyle	88.39	9	39.7755	835.2855	2019-03-02	12:40	Cash	795.51	4.761905	39.7
					Health and					2019-					

```python
def merge(dataframes):
  if dataframes:
    return pd.concat(dataframes)
```

```python
df = merge([csv,excel,json])
df.head()
```

	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	i
0	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	1/5/2019	13:08	Ewallet	522.83	4.761905	26
1	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2019	10:29	Cash	76.40	4.761905	3
2	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2019	13:23	Credit card	324.31	4.761905	16
					Health and										

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1000 entries, 0 to 999
Data columns (total 16 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Branch                   1000 non-null   object 
 1   City                     1000 non-null   object 
 2   Customer type            1000 non-null   object 
 3   Gender                   1000 non-null   object 
 4   Product line             1000 non-null   object 
 5   Unit price               1000 non-null   float64
 6   Quantity                 1000 non-null   int64  
 7   Tax 5%                   1000 non-null   float64
 8   Total                    1000 non-null   float64
 9   Date                     1000 non-null   object 
 10  Time                     1000 non-null   object 
 11  Payment                  1000 non-null   object 
 12  cogs                     1000 non-null   float64
 13  gross margin percentage  1000 non-null   float64
 14  gross income             1000 non-null   float64
 15  Rating                   1000 non-null   float64
dtypes: float64(7), int64(1), object(8)
memory usage: 165.1+ KB
```

```
df.describe()
```

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	gross income	Rating
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000	1.000000e+03	1000.000000	1000.00000
mean	55.672130	5.510000	15.379369	322.966749	307.58738	4.761905e+00	15.379369	6.97270
std	26.494628	2.923431	11.708825	245.885335	234.17651	6.131498e-14	11.708825	1.71858
min	10.080000	1.000000	0.508500	10.678500	10.17000	4.761905e+00	0.508500	4.00000
25%	32.875000	3.000000	5.924875	124.422375	118.49750	4.761905e+00	5.924875	5.50000
50%	55.230000	5.000000	12.088000	253.848000	241.76000	4.761905e+00	12.088000	7.00000
75%	77.935000	8.000000	22.445250	471.350250	448.90500	4.761905e+00	22.445250	8.50000
max	99.960000	10.000000	49.650000	1042.650000	993.00000	4.761905e+00	49.650000	10.00000

```
df.Date = pd.to_datetime(df.Date)
df.Time = pd.to_datetime(df.Time)
```

```
<ipython-input-10-520c7a075632>:2: UserWarning: Could not infer format, so each element will be parsed individually, falling back to
  df.Time = pd.to_datetime(df.Time)
```

```
df.head()
```

	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time	Payment	cogs	gross margin percentage	g in
0	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	2019-01-05	2024-11-04 13:08:00	Ewallet	522.83	4.761905	26.
1	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	2019-03-08	2024-11-04 10:29:00	Cash	76.40	4.761905	3.
2	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5255	2019-03-03	2024-11-04 13:23:00	Credit card	324.31	4.761905	16.
											2024-				

Next steps: Generate code with `df` | View recommended plots | New interactive sheet

```
df.dtypes
```

	0
Branch	object
City	object
Customer type	object
Gender	object
Product line	object
Unit price	float64
Quantity	int64
Tax 5%	float64
Total	float64
Date	datetime64[ns]
Time	datetime64[ns]
Payment	object
cogs	float64
gross margin percentage	float64
gross income	float64
Rating	float64

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

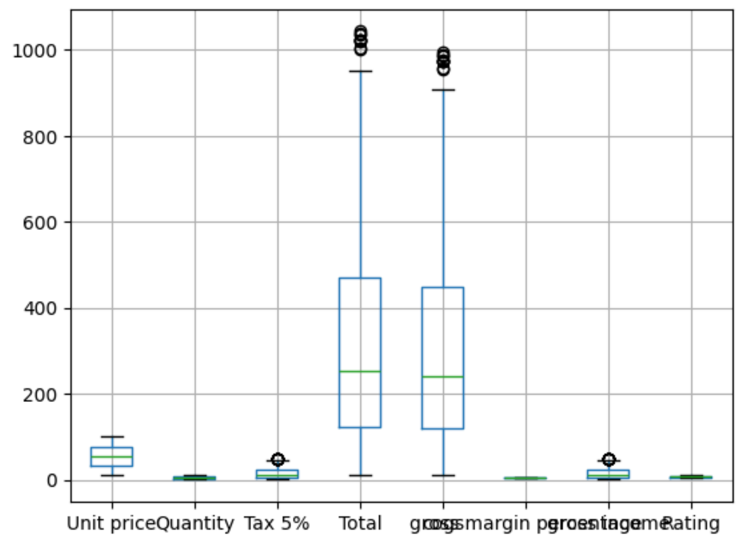
```
                          0
Branch                    0
City                      0
Customer type             0
Gender                    0
Product line              0
Unit price                0
Quantity                  0
Tax 5%                    0
Total                     0
Date                      0
Time                      0
Payment                   0
cogs                      0
gross margin percentage   0
gross income              0
Rating                    0

dtype: int64
```

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

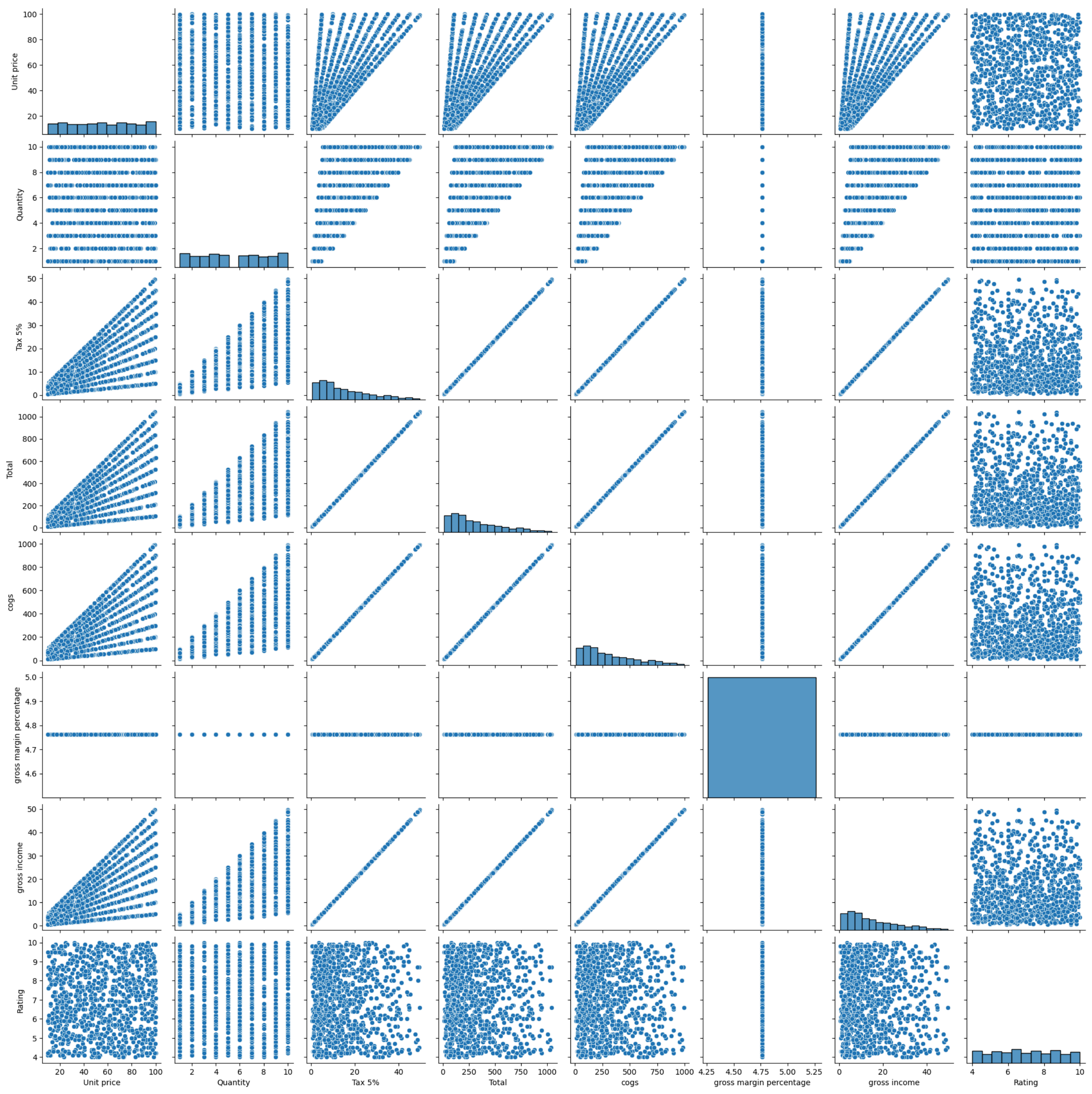
```
0
```

```
df.boxplot()
```

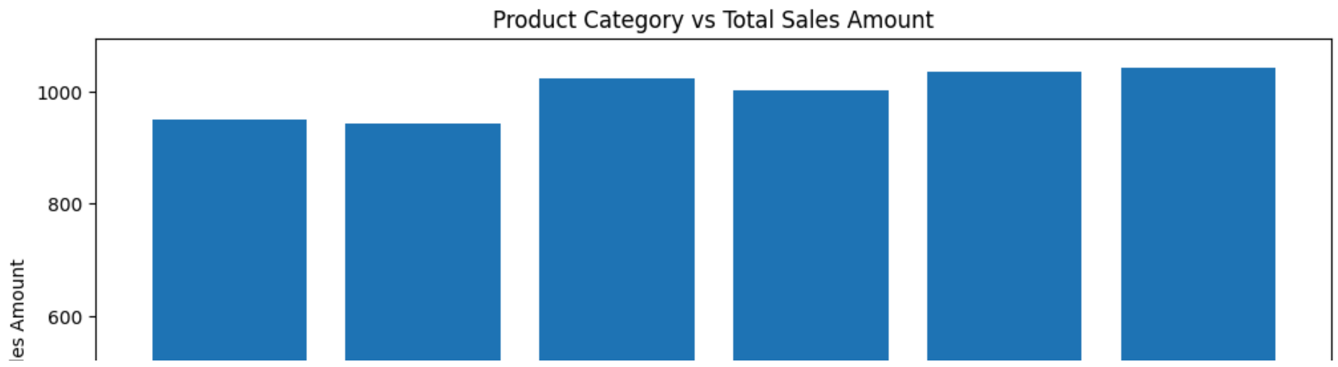
```
<Axes: >
```



```
sns.pairplot(df)
plt.show()
```



```
plt.figure(figsize=(12,6))
plt.bar(df['Product line'],df['Total'])
plt.title('Product Category vs Total Sales Amount')
plt.xlabel('Product Category')
plt.ylabel('Total Sales Amount')
plt.xticks(rotation=70)
plt.show()
```


Product Category vs Total Sales Amount
les Amount