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

my_walmart= pd.read_excel('/content/Walmart Sales.xlsx')

my_walmart.shape
```

```
(1000, 12)
```

```
my_walmart.isnull()
```


```
{"summary":"{\n  \"name\": \"my_walmart\",\n  \"rows\": 1000,\n
\"fields\": [\n    {\n      \"column\": \"Invoice ID\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Branch\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"City\",\n      \"properties\": {\n
\"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\n
\"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Customer type\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Gender\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Product line\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Unit price\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Quantity\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 1,\n        \"samples\": [\n          false\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Date\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
```

```
n    },\n    {\n      \"column\": \"Time\",\n      \"properties\": {\n
\"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\n
\"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Payment\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Rating\",\n      \"properties\":
{\n        \"dtype\": \"boolean\",\n        \"num_unique_values\": 1,\
n        \"samples\": [\n          false\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```


```
my_walmart.isnull().sum()
```

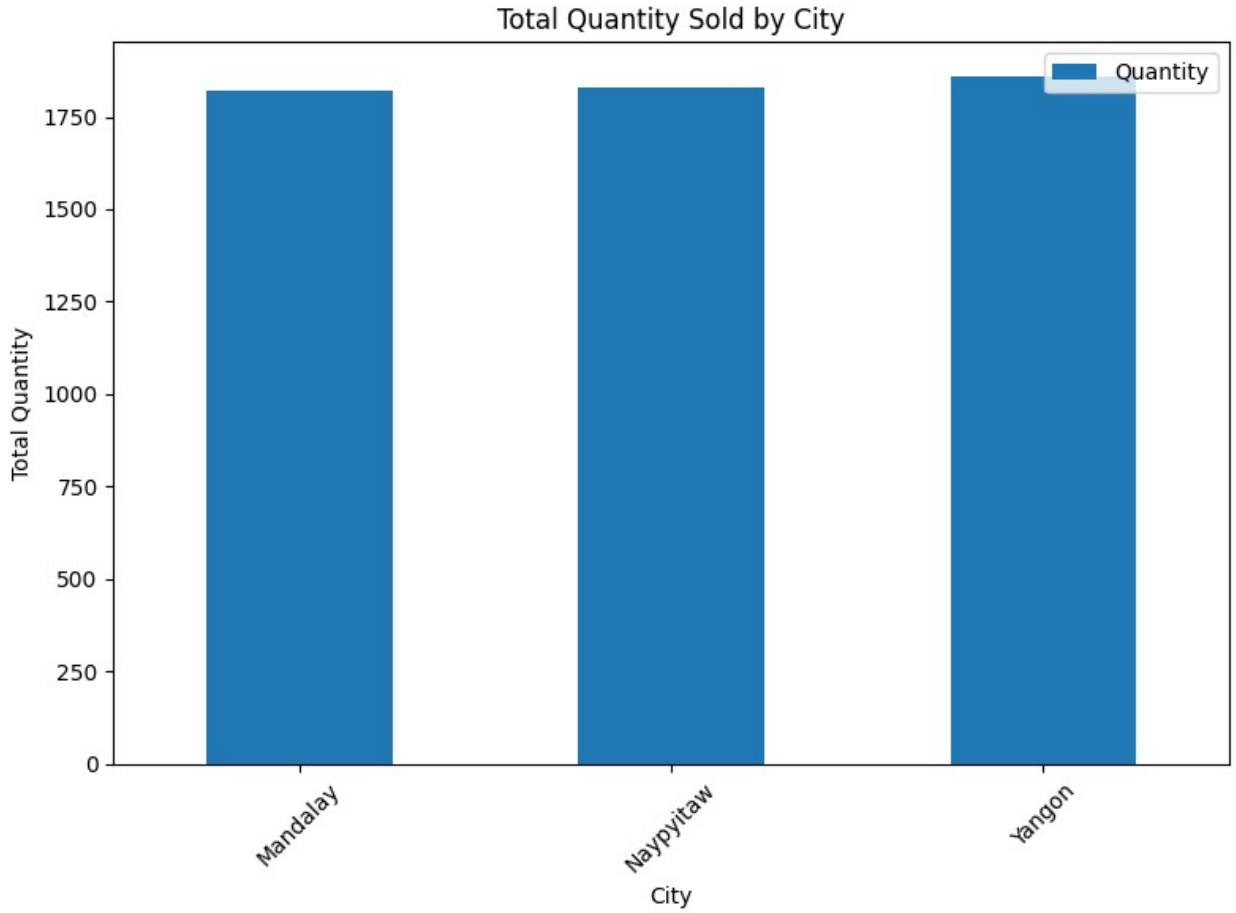
```
Invoice ID       0
Branch           0
City             0
Customer type    0
Gender           0
Product line     0
Unit price       0
Quantity         0
Date             0
Time             0
Payment          0
Rating           0
dtype: int64
```

```
#1. Walmart Sales Analysis:

#A. Analyze the performance of sales and revenue at the city and
branch level

City_Re=my_walmart.groupby('City')['Quantity'].sum().reset_index()
print(City_Re)
```

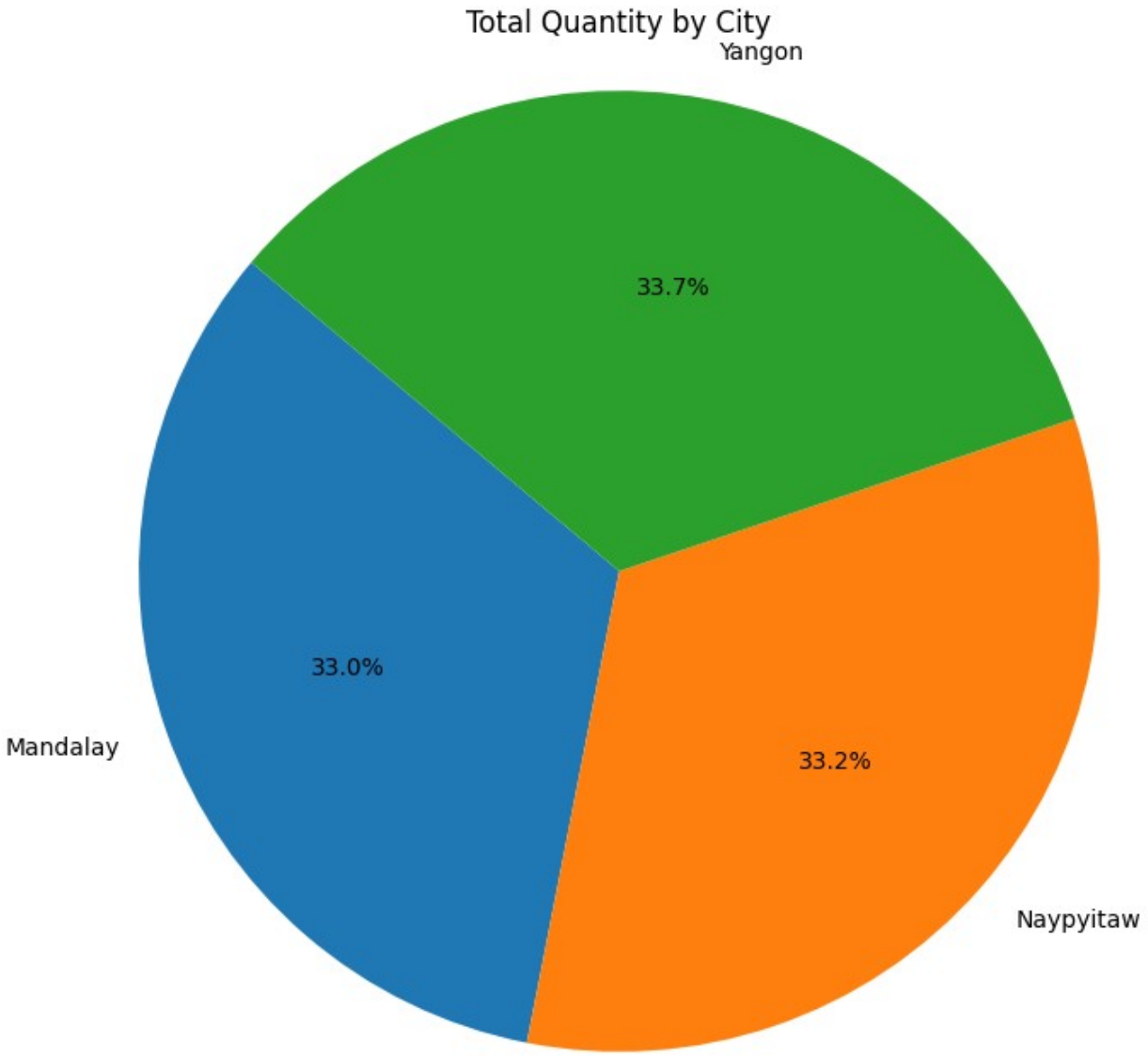
```
        City  Quantity
0   Mandalay      1820
1  Naypyitaw      1831
2     Yangon      1859
```

```
#Sales Quantity for City-wise:

City_Re.plot(kind='bar', x='City', y='Quantity', figsize=(8, 6))
plt.xlabel('City')
plt.ylabel('Total Quantity')
plt.title('Total Quantity Sold by City')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
```

```
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```




```
plt.figure(figsize=(8, 8))
plt.pie(City_Re['Quantity'], labels=City_Re['City'], autopct='%1.1f%
%', startangle=140)
plt.title('Total Quantity by City')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a
circle.
plt.show()
```

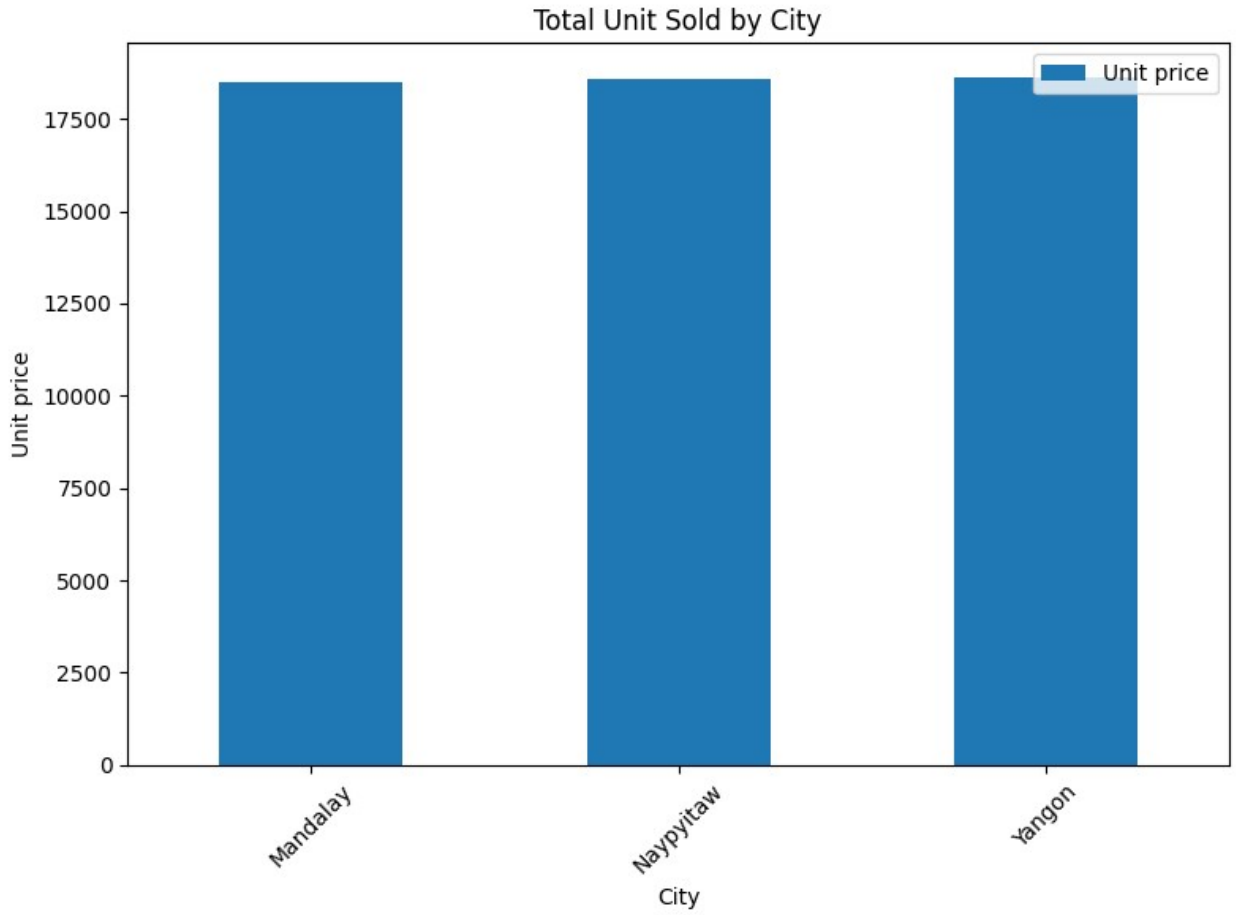



```
#Sales Unit price for City-wise:

City_Re_1=my_walmart.groupby('City')['Unit price'].sum().reset_index()
print(City_Re_1)
```

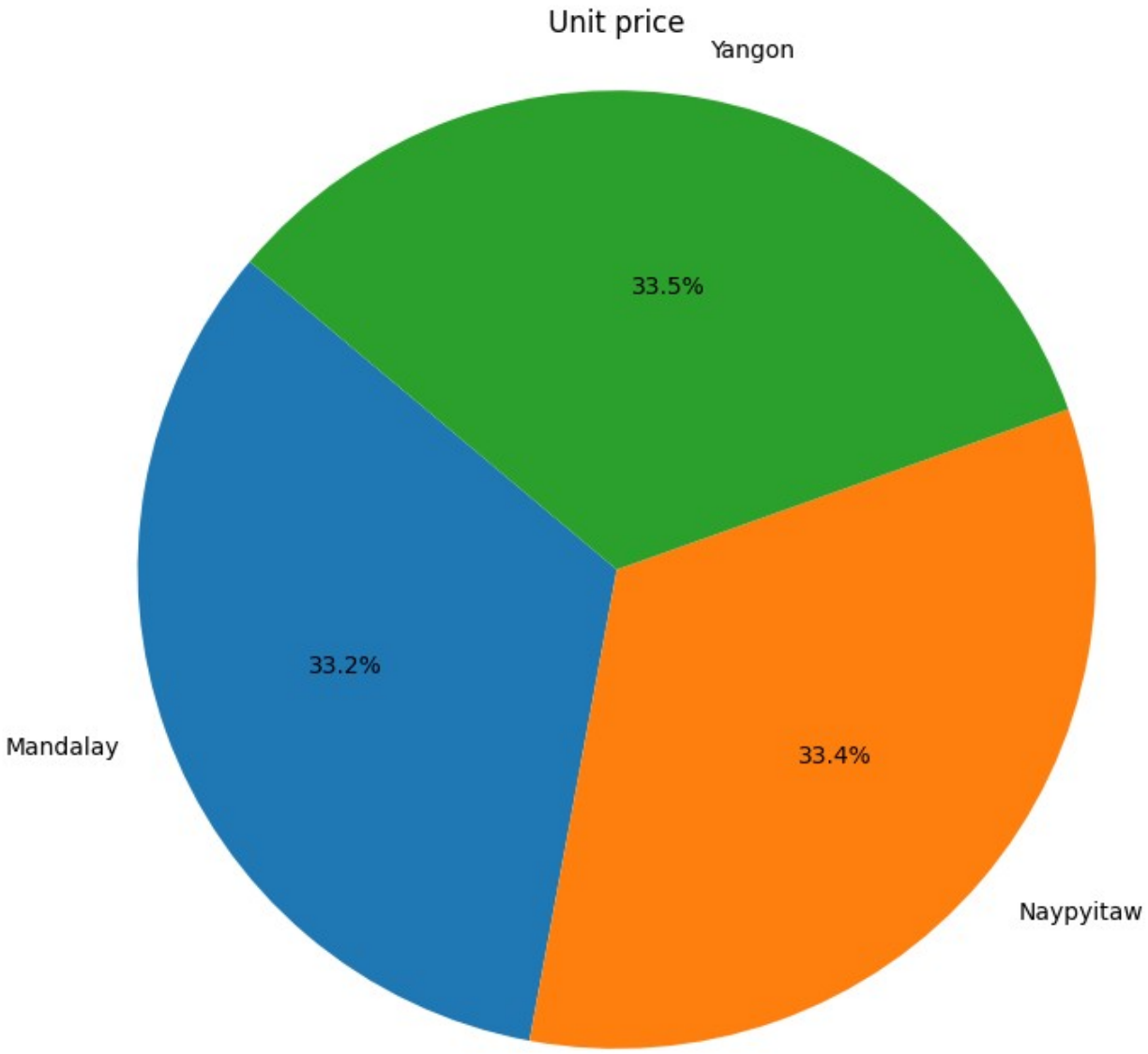
```
        City  Unit price
0   Mandalay    18478.88
1  Naypyitaw    18567.76
2     Yangon    18625.49
```

```
City_Re_1.plot(kind='bar', x='City', y='Unit price', figsize=(8, 6))
plt.xlabel('City')
plt.ylabel('Unit price')
plt.title('Total Unit Sold by City')
```

```
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



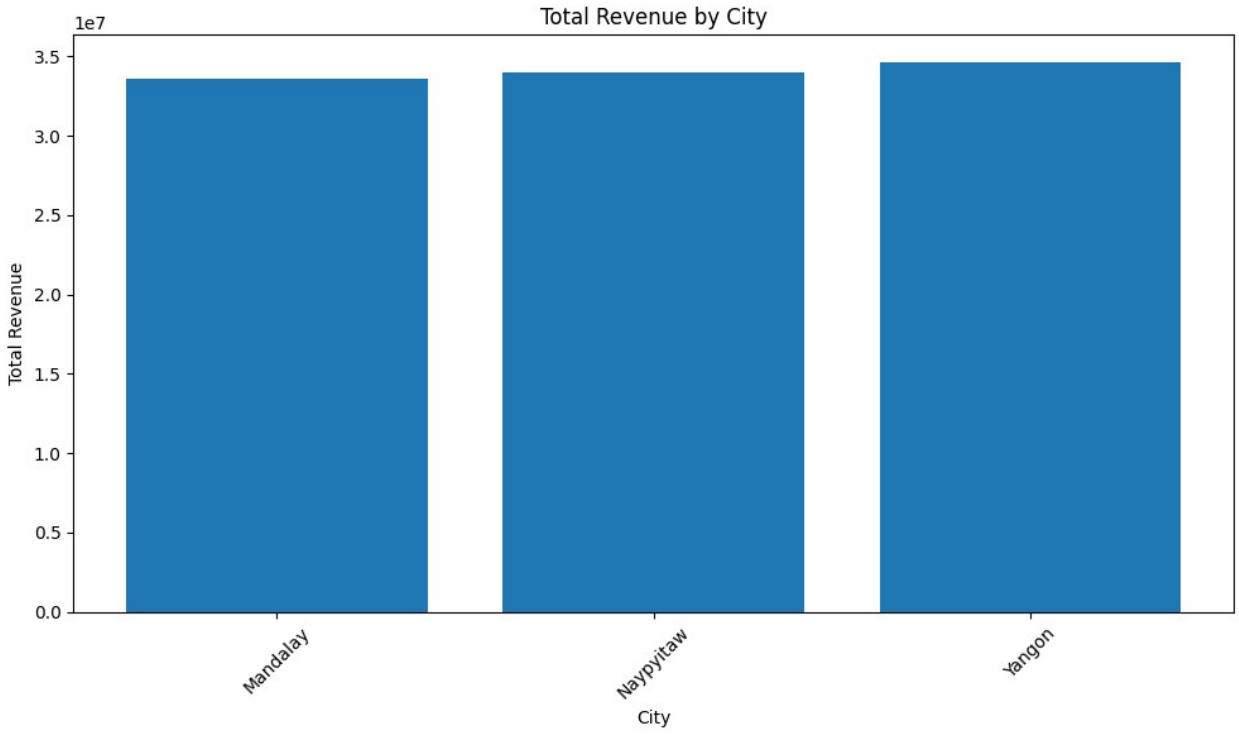

```
plt.figure(figsize=(8, 8))
plt.pie(City_Re_1['Unit price'], labels=City_Re_1['City'],
autopct='%1.1f%%', startangle=140)
plt.title('Unit price')
plt.axis('equal')
plt.show()
```



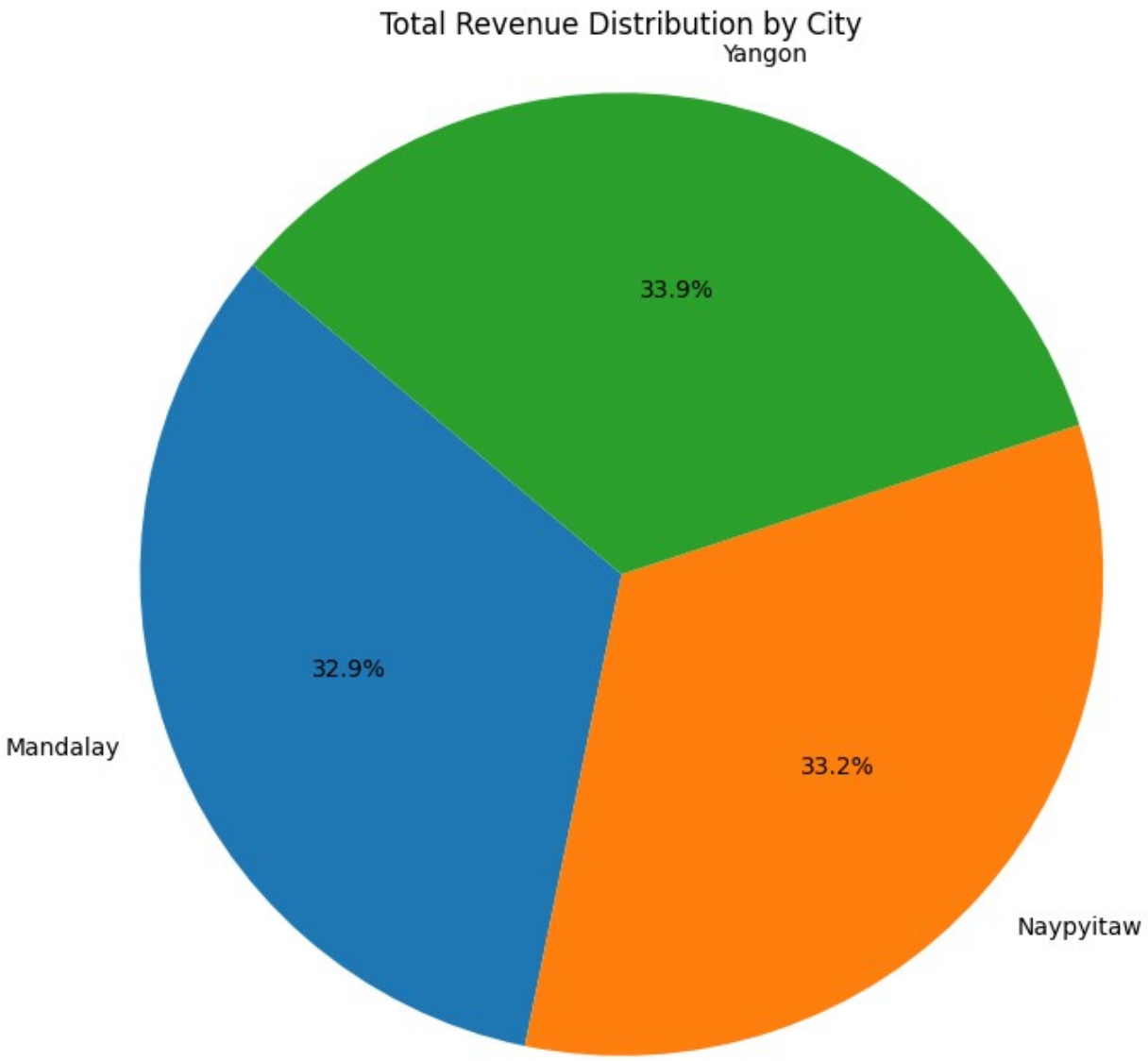

```
#Sales of Total Revenue for city-wise

City_Re['Total Revenue'] = City_Re['Quantity'] * City_Re_1['Unit
price']

print('Total Revenue by City:')
print(City_Re[['City', 'Total Revenue']])
```

```
Total Revenue by City:
        City  Total Revenue
0   Mandalay    33631561.60
1  Naypyitaw    33997568.56
2     Yangon    34624785.91
```

```
plt.figure(figsize=(10, 6))
plt.bar(City_Re['City'], City_Re['Total Revenue'])
plt.xlabel('City')
plt.ylabel('Total Revenue')
plt.title('Total Revenue by City')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(8, 8))
plt.pie(City_Re['Total Revenue'], labels=City_Re['City'],
autopct='%1.1f%%', startangle=140)
plt.title('Total Revenue Distribution by City')
plt.axis('equal')
plt.show()
```

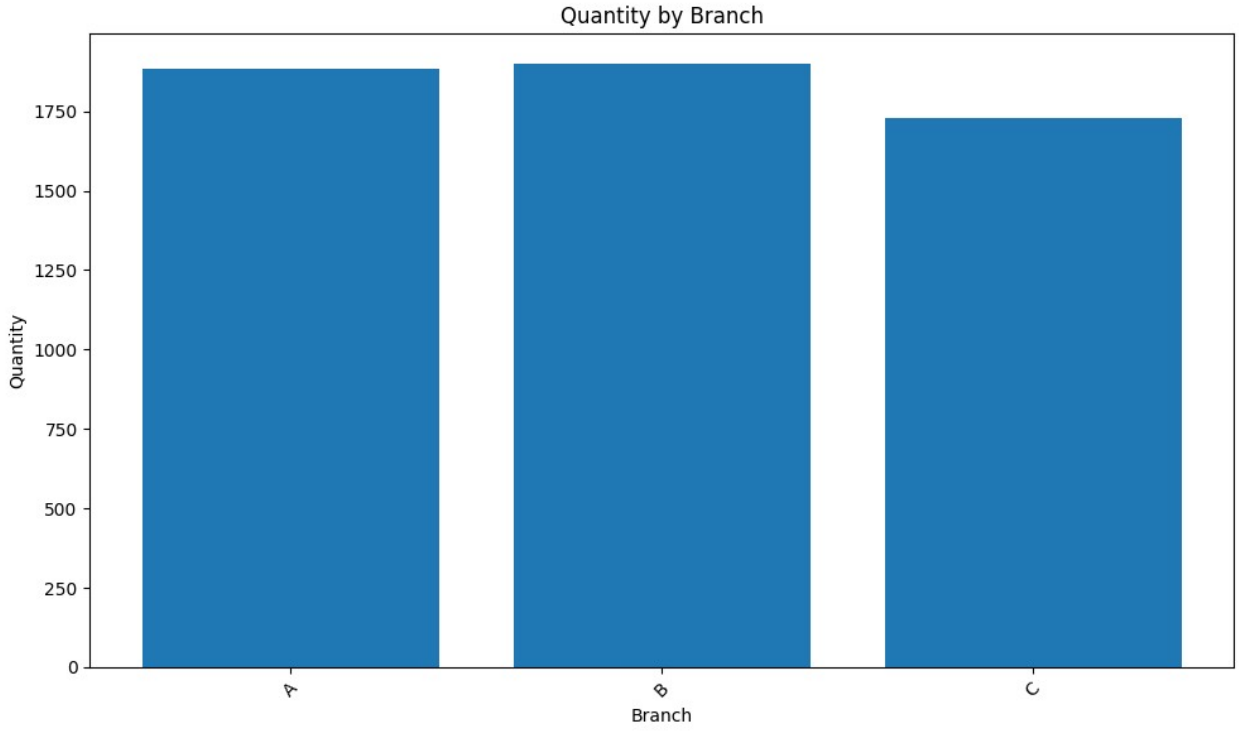



```
#Branch Wise Revenue

Branch_Wise_Revenue=my_walmart.groupby('Branch')
['Quantity'].sum().reset_index()
print(Branch_Wise_Revenue)
```

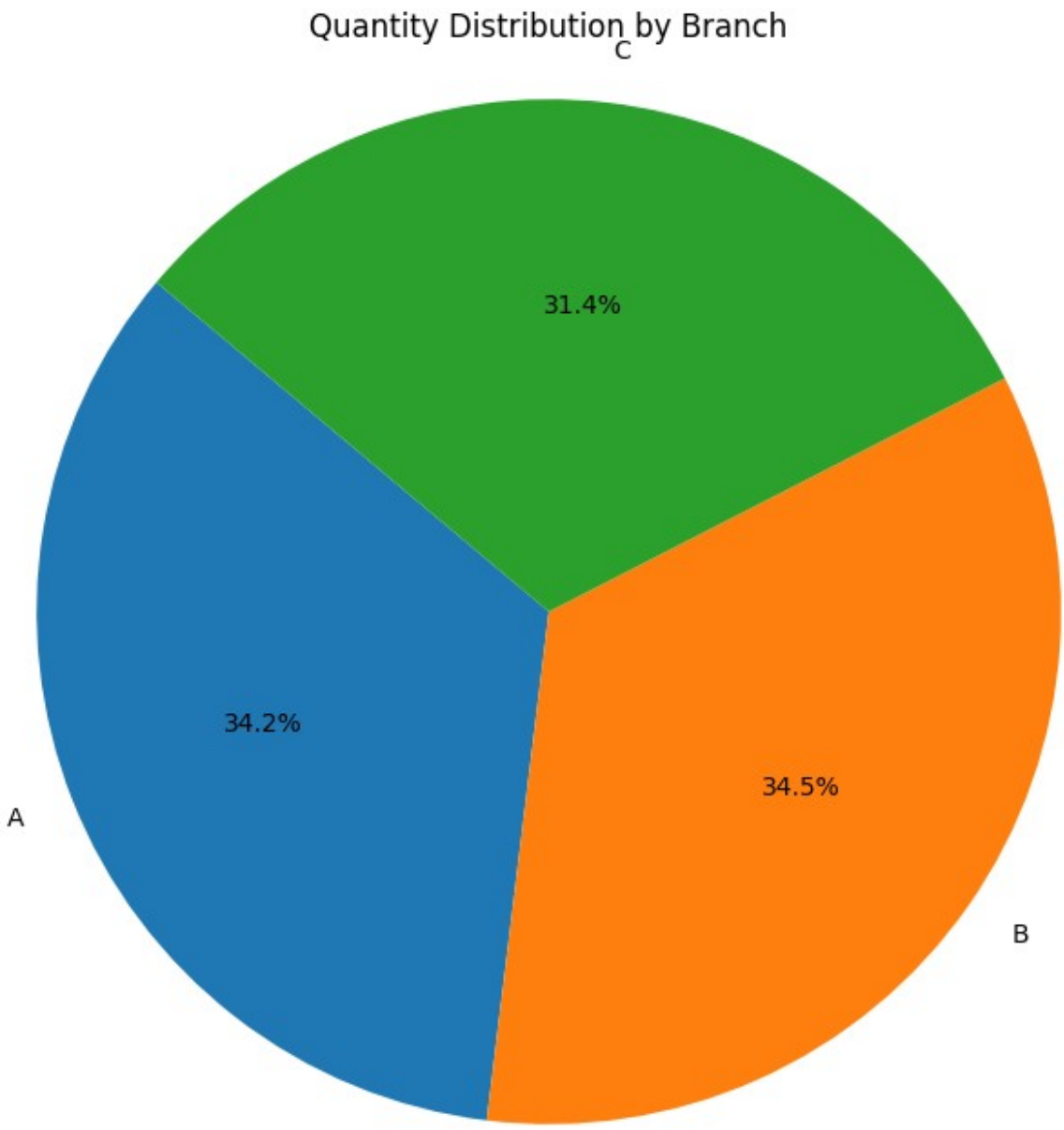
```
  Branch  Quantity
0      A      1883
1      B      1899
2      C      1728
```

```
#Sales Quantity for Branch-wise:

plt.figure(figsize=(10, 6))
plt.bar(Branch_Wise_Revenue['Branch'],
```

```
Branch_Wise_Revenue['Quantity'])
plt.xlabel('Branch')
plt.ylabel('Quantity')
plt.title('Quantity by Branch')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



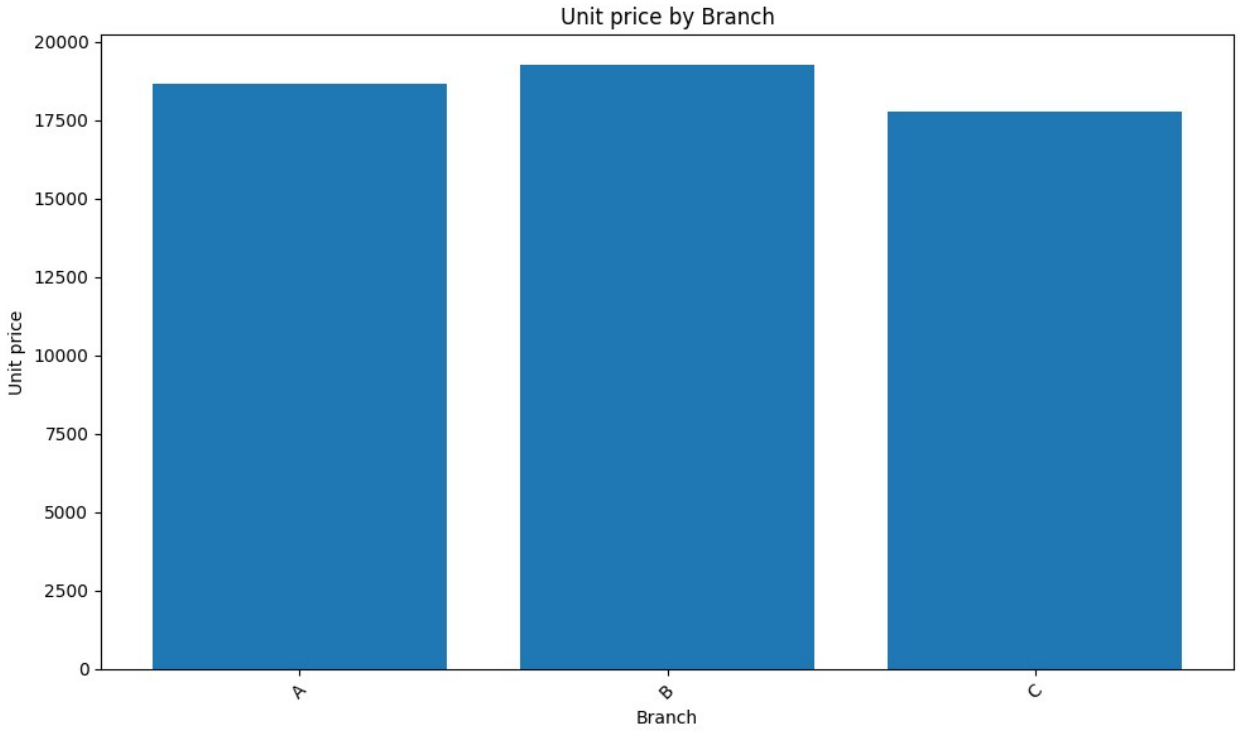

```
plt.figure(figsize=(8, 8))
plt.pie(Branch_Wise_Revenue['Quantity'],
labels=Branch_Wise_Revenue['Branch'], autopct='%1.1f%%',
startangle=140)
plt.title('Quantity Distribution by Branch')
plt.axis('equal')
plt.show()
```



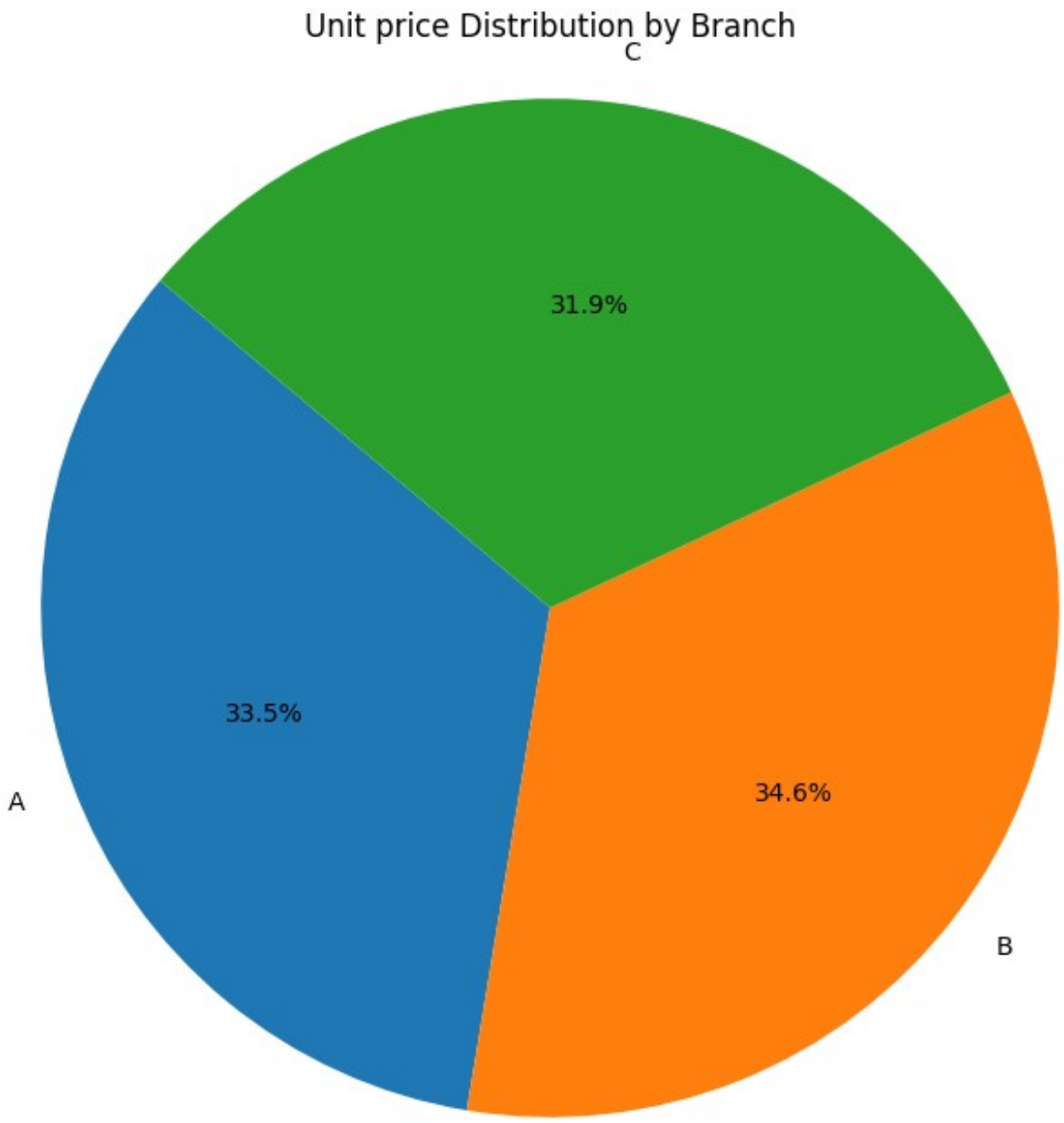

```
#Sales Unit Price for Branch-wise:

Branch_Wise_Revenue1=my_walmart.groupby('Branch')['Unit
price'].sum().reset_index()
print(Branch_Wise_Revenue1)
```

```
  Branch  Unit price
0      A    18645.54
1      B    19251.62
2      C    17774.97
```

```
plt.figure(figsize=(10, 6))
plt.bar(Branch_Wise_Revenue1['Branch'], Branch_Wise_Revenue1['Unit
price'])
plt.xlabel('Branch')
plt.ylabel('Unit price')
plt.title('Unit price by Branch')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



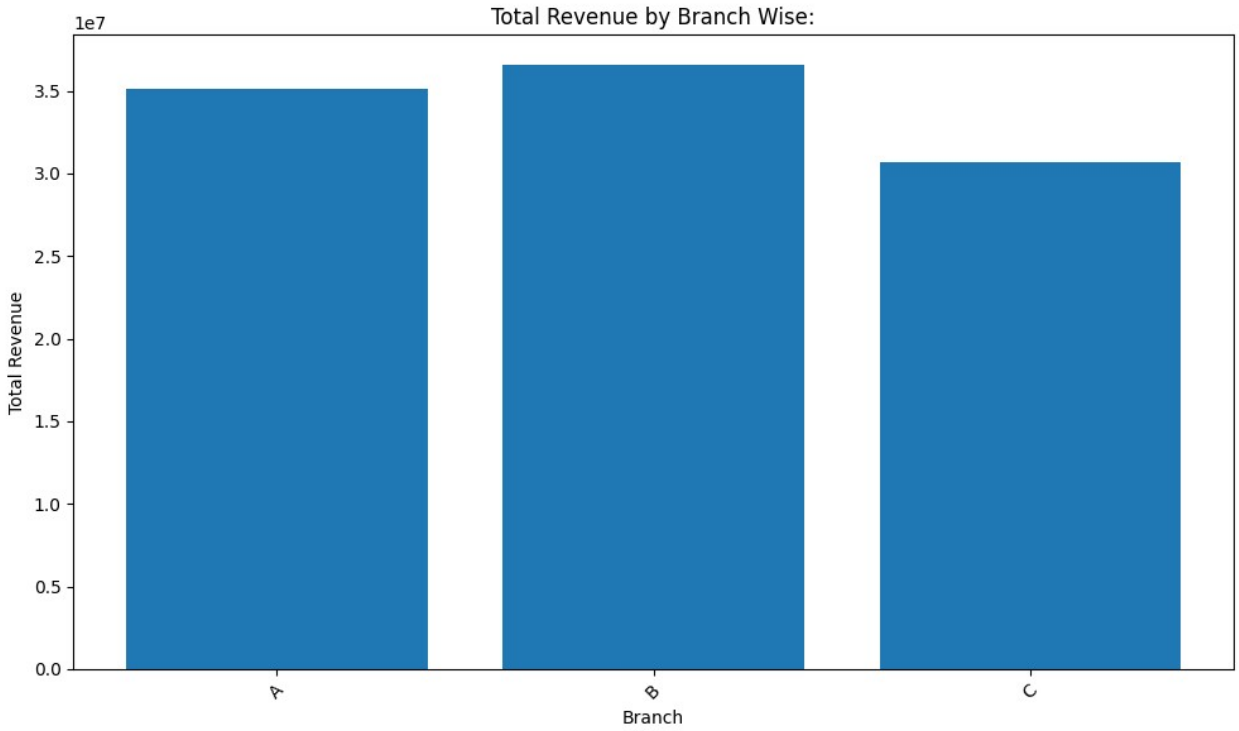

```
plt.figure(figsize=(8, 8))
plt.pie(Branch_Wise_Revenue1['Unit price'],
labels=Branch_Wise_Revenue1['Branch'], autopct='%1.1f%%',
startangle=140)
plt.title('Unit price Distribution by Branch')
plt.axis('equal')
plt.show()
```



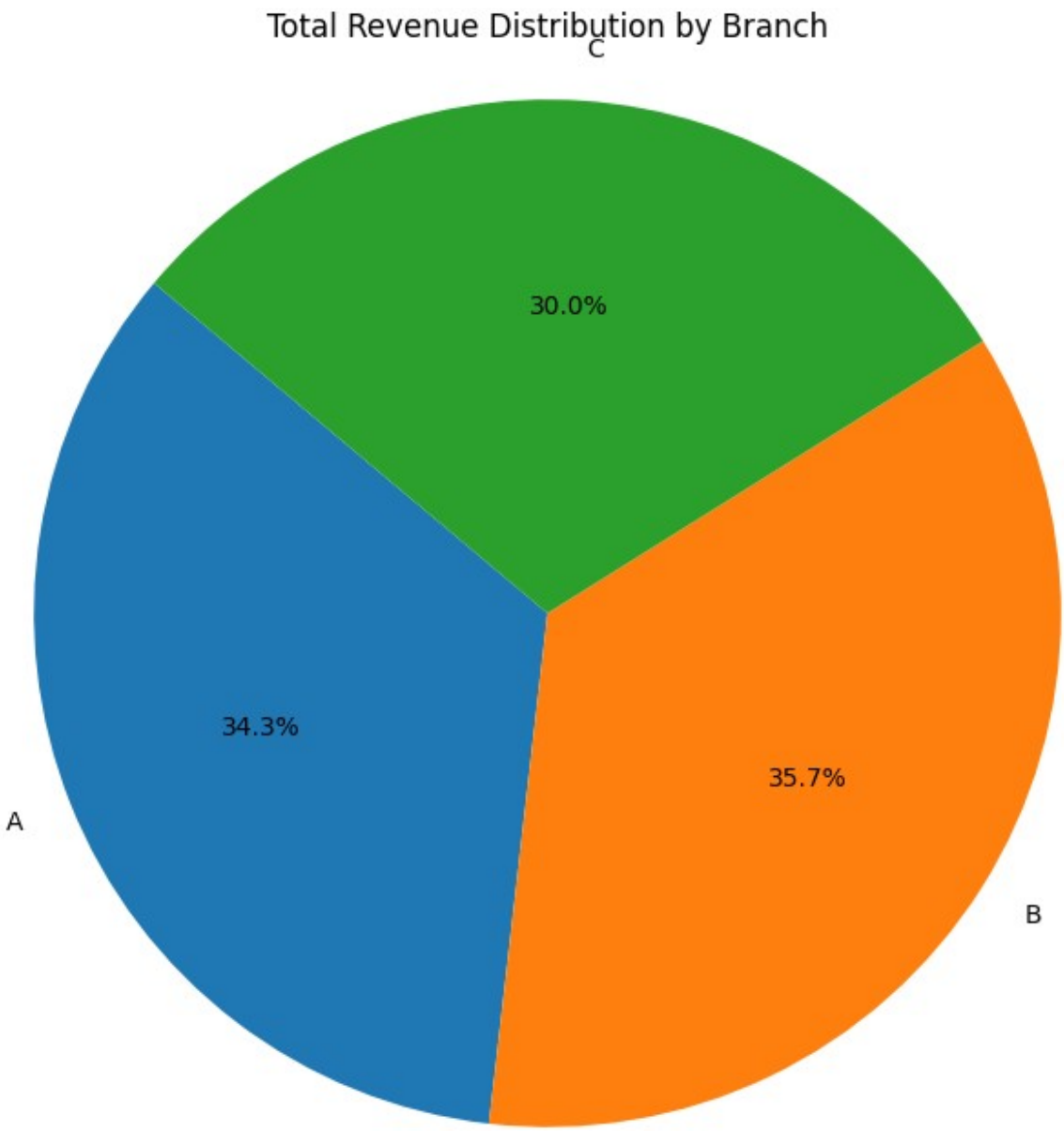

```
#Sales Total Revenue for Branch-wise:

Branch_Wise_Revenue['Total Revenue'] = Branch_Wise_Revenue['Quantity']
*Branch_Wise_Revenue1['Unit price']
print('Total Revenue by Branch Wise:')
print(Branch_Wise_Revenue[['Branch', 'Total Revenue']])
```

```
Total Revenue by Branch Wise:
  Branch  Total Revenue
0      A    35109551.82
1      B    36558826.38
2      C    30715148.16
```

```
plt.figure(figsize=(10, 6))
plt.bar(Branch_Wise_Revenue['Branch'], Branch_Wise_Revenue['Total
Revenue'])
plt.xlabel('Branch')
plt.ylabel('Total Revenue')
plt.title('Total Revenue by Branch Wise:')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(8, 8))
plt.pie(Branch_Wise_Revenue['Total Revenue'],
labels=Branch_Wise_Revenue['Branch'], autopct='%1.1f%%',
startangle=140)
plt.title('Total Revenue Distribution by Branch')
plt.axis('equal')
```

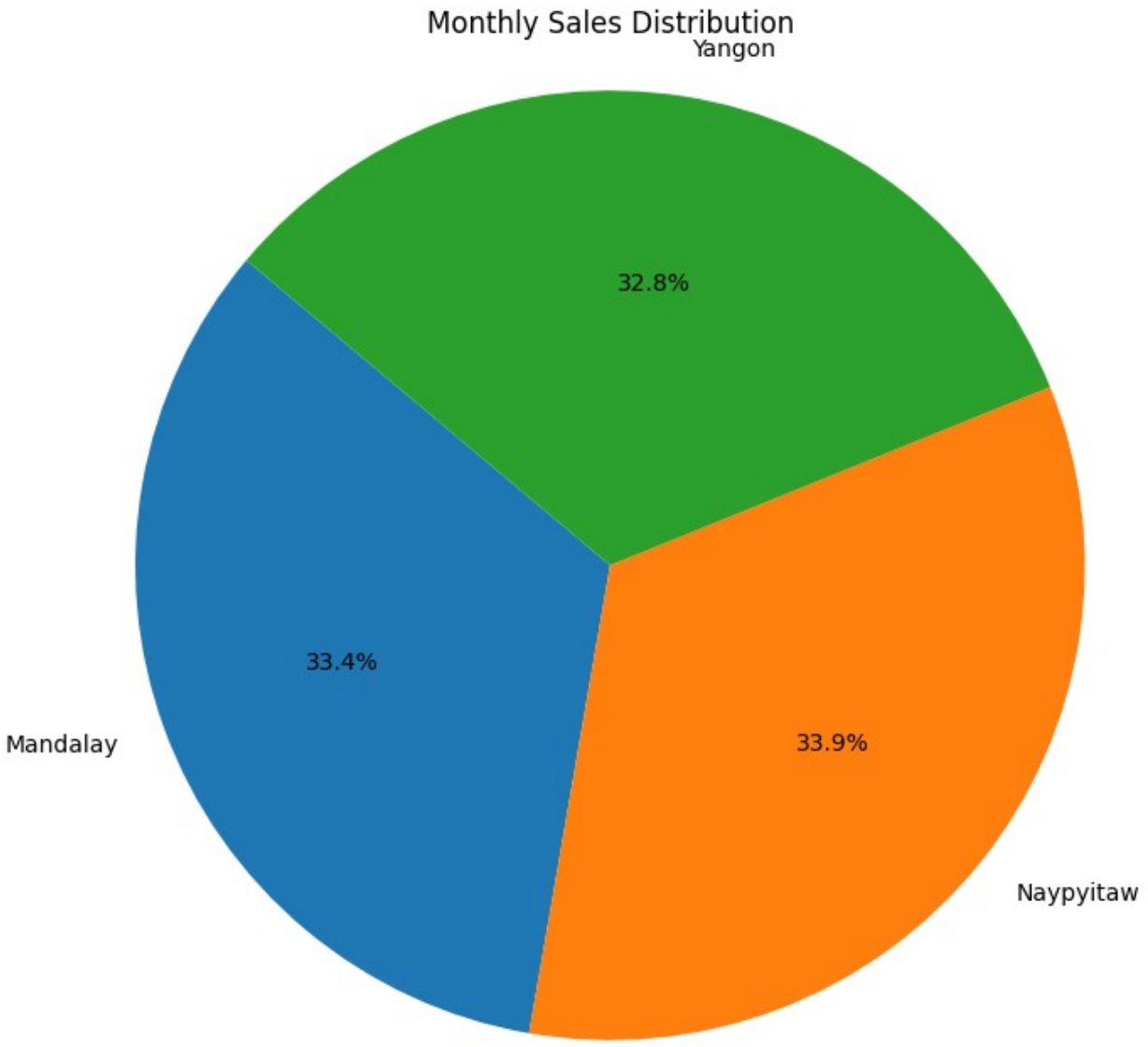



```
# b)The average price of an item sold at each branch of the city

Avg_price = my_walmart.groupby(['City', 'Branch'])['Unit
price'].mean().reset_index()
print('Average Price by Branch:')
print(Avg_price)
```

```
Average Price by Branch:
        City Branch  Unit price
0   Mandalay      A   53.353866
1   Mandalay      B   56.133305
2   Mandalay      C   57.958316
```

```
3  Naypyitaw      A   54.123182
4  Naypyitaw      B   57.785688
5  Naypyitaw      C   57.941009
6     Yangon      A   55.639298
7     Yangon      B   56.011062
8     Yangon      C   52.684602
```

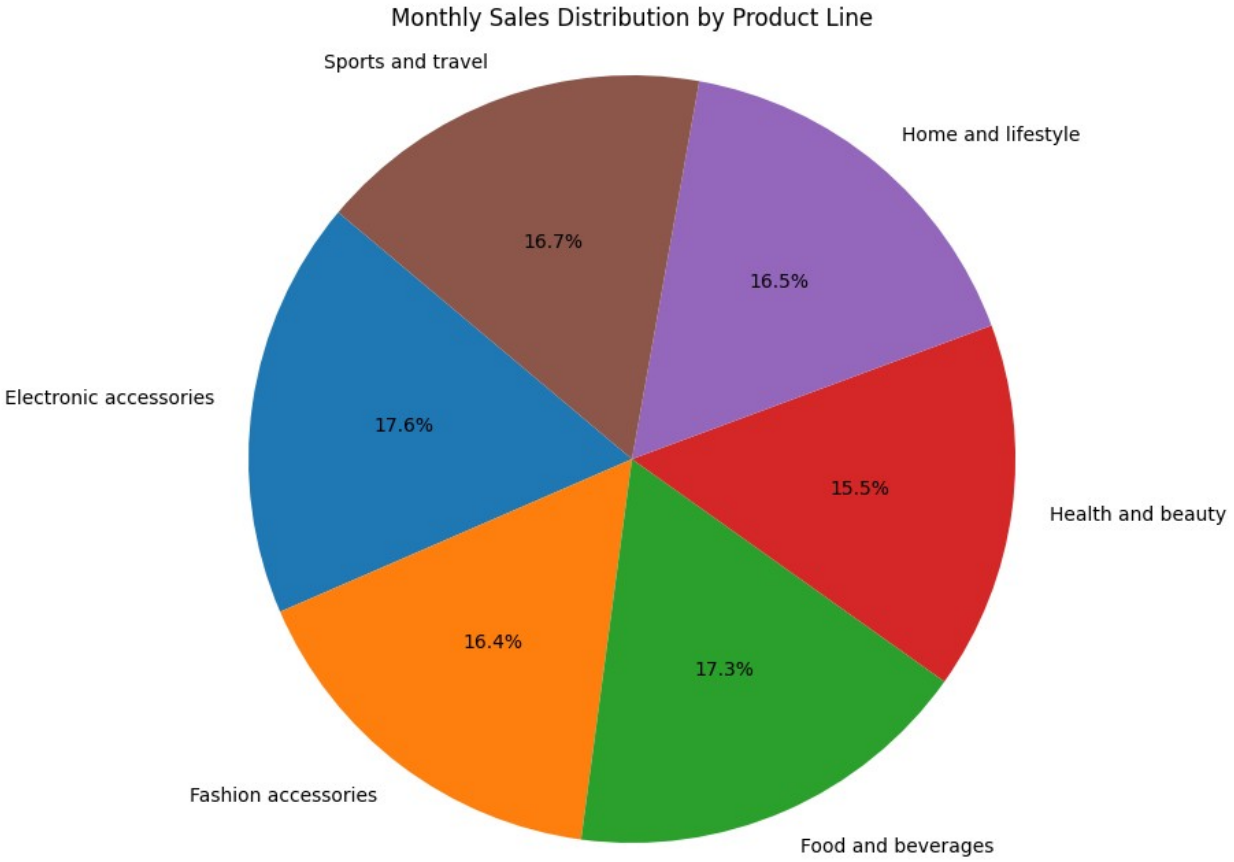
```
plt.figure(figsize=(8, 8))
plt.pie(Avg_price.groupby('City')['Unit price'].sum(),
labels=Avg_price['City'].unique(), autopct='%1.1f%%', startangle=140)
plt.title('Monthly Sales Distribution')
plt.axis('equal')
plt.show()
```



```
# c) Analyze the performance of sales and revenue,Month over Month
across the Productline, Gender, and Payment Method, and identify the
focus areas to get better sales forApril 2019

my_walmart['Date'] = pd.to_datetime(my_walmart['Date'])

# Add month column
my_walmart['Month'] = my_walmart['Date'].dt.month

# Monthly sales by product line
monthly_sales = my_walmart.groupby(['Month', 'Product line'])
['Quantity'].sum().reset_index()
print('Monthly Sales by Product Line:')
print(monthly_sales)
```

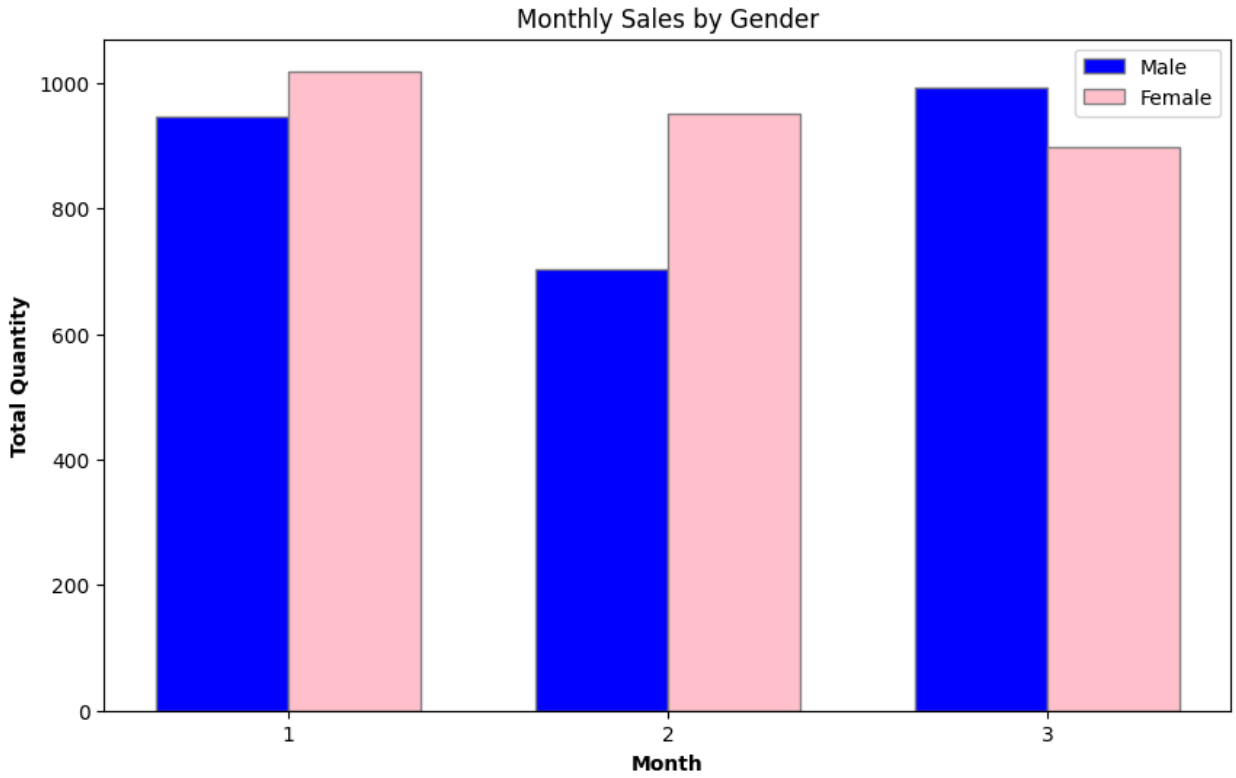
```
Monthly Sales by Product Line:
    Month            Product line  Quantity
0       1  Electronic accessories       333
1       1     Fashion accessories       336
2       1      Food and beverages       325
3       1       Health and beauty       254
4       1      Home and lifestyle       342
5       1       Sports and travel       375
6       2  Electronic accessories       313
7       2     Fashion accessories       295
8       2      Food and beverages       349
9       2       Health and beauty       266
10      2      Home and lifestyle       205
11      2       Sports and travel       226
12      3  Electronic accessories       325
13      3     Fashion accessories       271
14      3      Food and beverages       278
15      3       Health and beauty       334
16      3      Home and lifestyle       364
17      3       Sports and travel       319
```

```
# Pivot the data for better visualization
product_pivot = monthly_sales.pivot(index='Month', columns='Product
line', values='Quantity')

# Plotting the pie chart
plt.figure(figsize=(10, 8))
plt.pie(monthly_sales.groupby('Product line')['Quantity'].sum(),
labels=monthly_sales['Product line'].unique(), autopct='%1.1f%%',
startangle=140)
plt.title('Monthly Sales Distribution by Product Line')
plt.axis('equal')
plt.show()
```

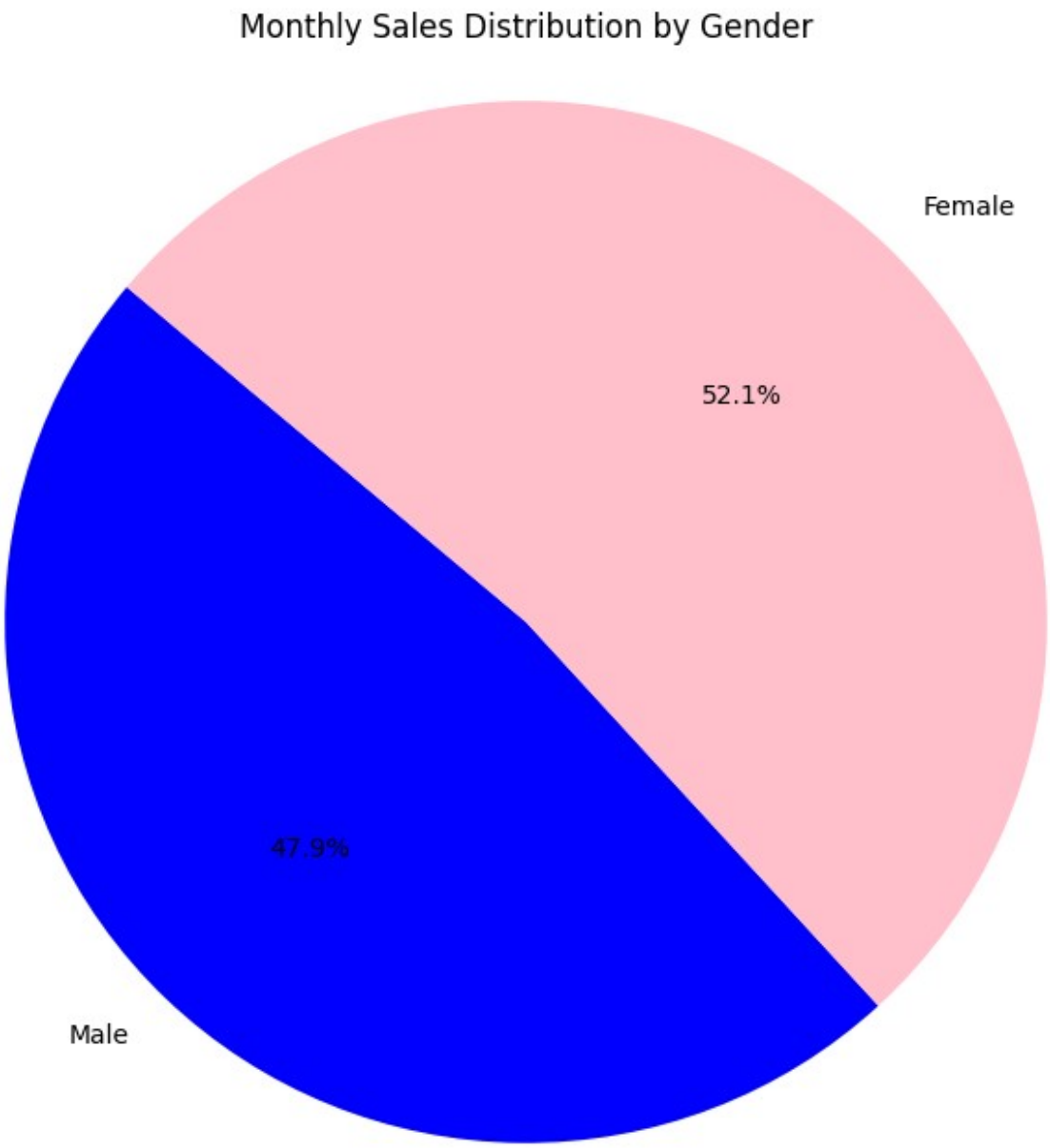



```
# Create separate dataframes for each gender
male_data = Gender_monthly[Gender_monthly['Gender'] == 'Male']
female_data = Gender_monthly[Gender_monthly['Gender'] == 'Female']

# Set the width of the bars
bar_width = 0.35

# Set the position of the bars on the x-axis
r1 = range(len(male_data))
r2 = [x + bar_width for x in r1]

# Plotting the bar chart
plt.figure(figsize=(10, 6))
plt.bar(r1, male_data['Quantity'], color='blue', width=bar_width,
edgecolor='grey', label='Male')
plt.bar(r2, female_data['Quantity'], color='pink', width=bar_width,
edgecolor='grey', label='Female')

# Add xticks on the middle of the group bars
plt.xlabel('Month', fontweight='bold')
plt.ylabel('Total Quantity', fontweight='bold')
plt.xticks([r + bar_width / 2 for r in range(len(male_data))],
male_data['Month'])
```

```
# Create legend & Show graphic
plt.legend()
plt.title('Monthly Sales by Gender')
plt.show()
```




```
# Separate data for each gender
male_data = Gender_monthly[Gender_monthly['Gender'] == 'Male']
female_data = Gender_monthly[Gender_monthly['Gender'] == 'Female']

# Labels for the pie chart
labels = ['Male', 'Female']

# Data for the pie chart
sizes = [male_data['Quantity'].sum(), female_data['Quantity'].sum()]

# Colors for each section
colors = ['blue', 'pink']

# Plotting the pie chart
plt.figure(figsize=(8, 8))
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%',
startangle=140)
plt.title('Monthly Sales Distribution by Gender')
plt.axis('equal')
plt.show()
```




```
# Monthly sales by payment method
Payment_monthly = my_walmart.groupby(['Month', 'Payment'])
['Quantity'].sum().reset_index()
print('\nMonthly Sales by Payment Method:')
print(Payment_monthly)
```

```
Monthly Sales by Payment Method:
   Month      Payment  Quantity
0      1         Cash       708
1      1  Credit card       622
2      1      Ewallet       635
```

```
3      2         Cash       596
4      2  Credit card       505
5      2      Ewallet       553
6      3         Cash       592
7      3  Credit card       595
8      3      Ewallet       704
```

```
# Pivot the data for better visualization
payment_pivot = Payment_monthly.pivot(index='Month',
columns='Payment', values='Quantity')

# Plotting the bar chart
payment_pivot.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.xlabel('Month')
plt.ylabel('Total Quantity')
plt.title('Monthly Sales by Payment Method')
plt.xticks(rotation=45)
plt.legend(title='Payment Method')
plt.tight_layout()
plt.show()
```


```
plt.figure(figsize=(8, 8))
plt.pie(Payment_monthly.groupby('Payment')['Quantity'].sum(),
labels=Payment_monthly['Payment'].unique(), autopct='%1.1f%%',
startangle=140)
plt.title('Monthly Sales Distribution by Payment Method')
```

```
plt.axis('equal')
plt.show()
```