About Dataset

It is a sales data of a restaurant company operating in multiple cities in the world. It contains information about individual sales transactions, customer demographics, and product details. The data is structured in a tabular format, with each row representing a single record and each column representing a specific attribute. This dataset can be commonly used for business intelligence, sales forecasting, and customer behaviour analysis.

Restaurant Sales Data Analysis

In [67]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [68]:
```
df=pd.read_csv(r'C:\Users\hi\Downloads\9. Sales-Data-Analysis.csv')
```

In [69]:
```
df.head()
```

Out[69]:

	Order ID	Date	Product	Price	Quantity	Purchase Type	Payment Method	Manager	City
0	10452	07-11-2022	Fries	3.49	573.07	Online	Gift Card	Tom Jackson	London
1	10453	07-11-2022	Beverages	2.95	745.76	Online	Gift Card	Pablo Perez	Madrid
2	10454	07-11-2022	Sides & Other	4.99	200.40	In-store	Gift Card	Joao Silva	Lisbon
3	10455	08-11-2022	Burgers	12.99	569.67	In-store	Credit Card	Walter Muller	Berlin
4	10456	08-11-2022	Chicken Sandwiches	9.95	201.01	In-store	Credit Card	Walter Muller	Berlin

In [70]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 254 entries, 0 to 253
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Order ID        254 non-null    int64  
 1   Date            254 non-null    object 
 2   Product         254 non-null    object 
 3   Price           254 non-null    float64
 4   Quantity        254 non-null    float64
 5   Purchase Type   254 non-null    object 
 6   Payment Method  254 non-null    object 
 7   Manager         254 non-null    object 
 8   City            254 non-null    object 
dtypes: float64(2), int64(1), object(6)
memory usage: 18.0+ KB
```

In [71]:
```
df.isnull().sum()
```

Out[71]:
```
Order ID          0
Date              0
Product           0
Price             0
Quantity          0
Purchase Type     0
Payment Method    0
Manager           0
City              0
dtype: int64
```

In [72]:
```
df.columns
```

Out[72]:
```
Index(['Order ID', 'Date', 'Product', 'Price', 'Quantity', 'Purchase Type',
       'Payment Method', 'Manager', 'City'],
      dtype='object')
```

In [73]:
```
df.describe()
```

Out[73]:

	Order ID	Price	Quantity
count	254.000000	254.000000	254.000000
mean	10584.133858	7.102323	460.611457
std	75.889181	4.341855	214.888699
min	10452.000000	2.950000	200.400000
25%	10520.250000	3.490000	201.010000
50%	10583.500000	4.990000	538.880000
75%	10649.750000	9.950000	677.440000
max	10713.000000	29.050000	754.430000

Data Preprocessing

In [74]:
```
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')  # 'coerce' will turn invalid formats into NaT
```

In [75]:
```
df['Total Amount']=df['Price']*df['Quantity']
```

In [76]:
```
df.head()
```

Out[76]:

	Order ID	Date	Product	Price	Quantity	Purchase Type	Payment Method	Manager	City	Total Amount
0	10452	2022-07-11	Fries	3.49	573.07	Online	Gift Card	Tom Jackson	London	2000.0143
1	10453	2022-07-11	Beverages	2.95	745.76	Online	Gift Card	Pablo Perez	Madrid	2199.9920
2	10454	2022-07-11	Sides & Other	4.99	200.40	In-store	Gift Card	Joao Silva	Lisbon	999.9960
3	10455	2022-08-11	Burgers	12.99	569.67	In-store	Credit Card	Walter Muller	Berlin	7400.0133
4	10456	2022-08-11	Chicken Sandwiches	9.95	201.01	In-store	Credit Card	Walter Muller	Berlin	2000.0495

In [77]:
```
# Create Total Amount
df['Total Amount'] = df['Price'] * df['Quantity']

# Extract day, month, weekday
df['Day'] = df['Date'].dt.day
df['Month'] = df['Date'].dt.month
df['Weekday'] = df['Date'].dt.day_name()
```

In [78]:
```
print(f"Duplicate rows: {df.duplicated().sum()}")
```

```
Duplicate rows: 0
```

In [79]:
```
df = df.drop_duplicates()
```

In [80]:
```
df.head()
```

Out[80]:

	Order ID	Date	Product	Price	Quantity	Purchase Type	Payment Method	Manager	City	Total Amount	Day	Month	Weekday
0	10452	2022-07-11	Fries	3.49	573.07	Online	Gift Card	Tom Jackson	London	2000.0143	11.0	7.0	Monday
1	10453	2022-07-11	Beverages	2.95	745.76	Online	Gift Card	Pablo Perez	Madrid	2199.9920	11.0	7.0	Monday
2	10454	2022-07-11	Sides & Other	4.99	200.40	In-store	Gift Card	Joao Silva	Lisbon	999.9960	11.0	7.0	Monday
3	10455	2022-08-11	Burgers	12.99	569.67	In-store	Credit Card	Walter Muller	Berlin	7400.0133	11.0	8.0	Thursday
4	10456	2022-08-11	Chicken Sandwiches	9.95	201.01	In-store	Credit Card	Walter Muller	Berlin	2000.0495	11.0	8.0	Thursday

In [81]:
```
df['Purchase Type'].nunique()
```

Out[81]:
```
3
```

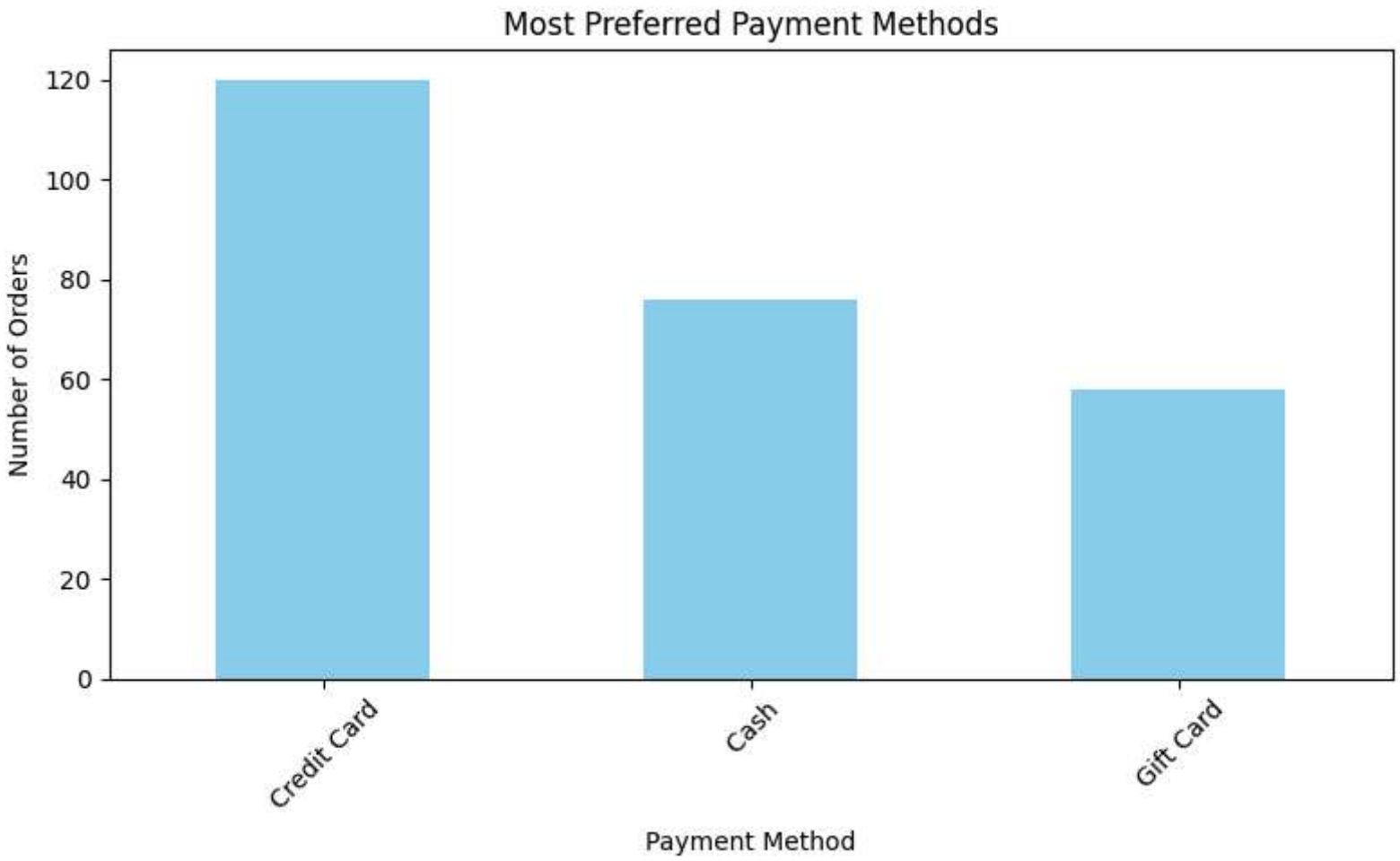
Q.1) Most Preferred Payment Method ?

In [82]:
```
payment_counts=df['Payment Method'].value_counts()
payment_counts
```

Out[82]:
```
Payment Method
Credit Card    120
Cash            76
Gift Card       58
Name: count, dtype: int64
```

In [83]:
```
plt.figure(figsize=(8,5))
payment_counts.plot(kind='bar', color='skyblue')
plt.title('Most Preferred Payment Methods')
plt.xlabel('Payment Method')
plt.ylabel('Number of Orders')
plt.xticks(rotation=45)
```

```
plt.tight_layout()
plt.show()
```



Q.2) Most Selling Product - By Quantity & By Revenue ?

In [84]:
```
# Total quantity sold per product
quantity_sold = df.groupby('Product')['Quantity'].sum().sort_values(ascending=False)

# Display top results
print("Most Selling Products by Quantity:\n", quantity_sold.head())
```

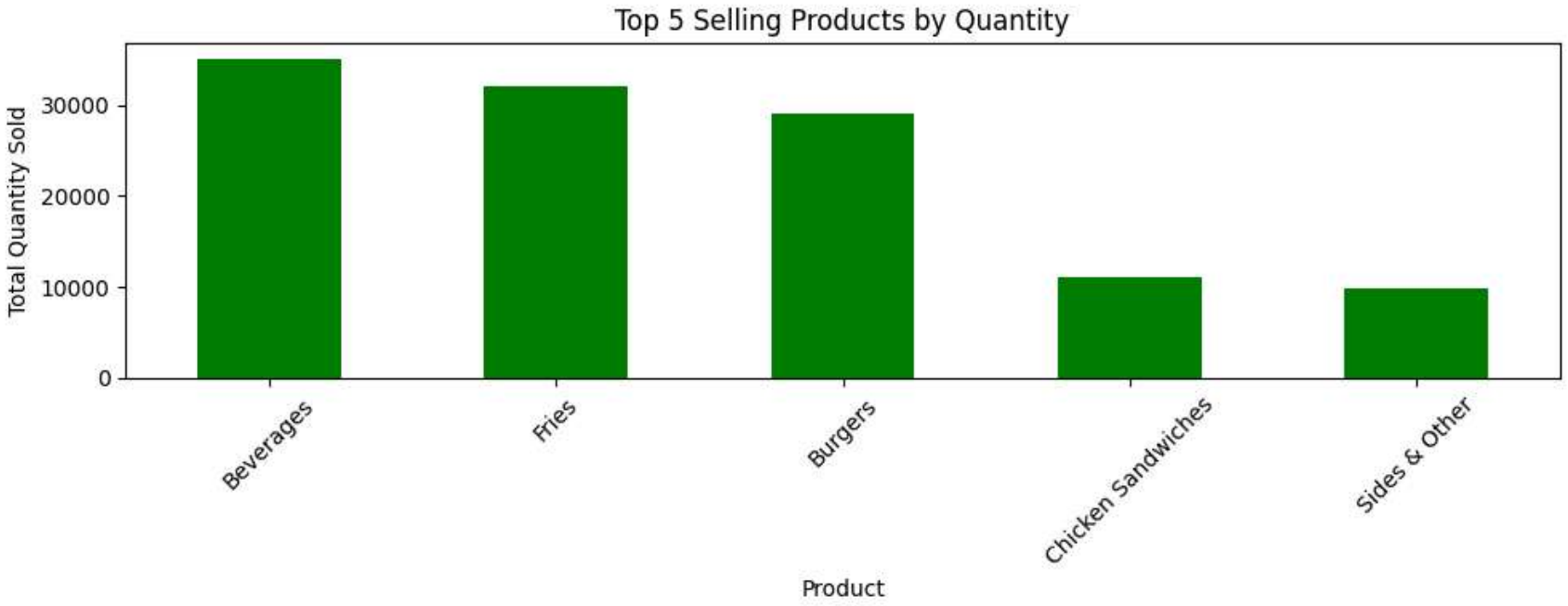
```
Most Selling Products by Quantity:
 Product
Beverages             34983.14
Fries                 32034.34
Burgers               29022.31
Chicken Sandwiches    11135.92
Sides & Other          9819.60
Name: Quantity, dtype: float64
```

In [85]:
```
revenue_by_product = df.groupby('Product')['Total Amount'].sum().sort_values(ascending=False)
```

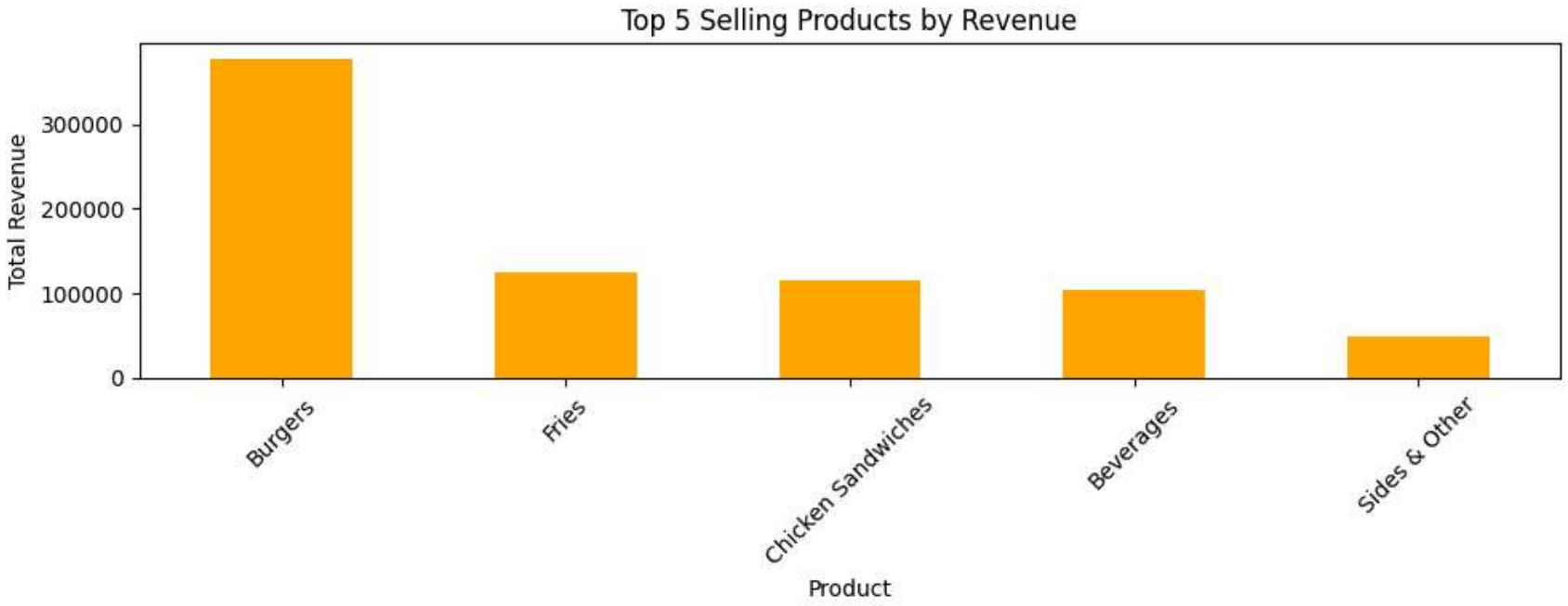
In [86]:
```
print("Most Selling Products by Revenue:\n", revenue_by_product.head())
```

```
Most Selling Products by Revenue:
 Product
Burgers               376999.8069
Fries                 125674.2903
Chicken Sandwiches    114641.6950
Beverages             103200.2630
Sides & Other          48999.8040
Name: Total Amount, dtype: float64
```

In [87]:
```
plt.figure(figsize=(10,4))
quantity_sold.head(5).plot(kind='bar', color='green')
plt.title('Top 5 Selling Products by Quantity')
plt.ylabel('Total Quantity Sold')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [88]:
```
plt.figure(figsize=(10,4))
revenue_by_product.head(5).plot(kind='bar', color='orange')
plt.title('Top 5 Selling Products by Revenue')
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Q.3) Which city had maximum revenue , or , Which Manager earned maximum revenue ?

In [89]:
```
#total revenue by city
revenue_by_city = df.groupby('City')['Total Amount'].sum().sort_values(ascending=False)
```

In [90]:
```
print('Revenur by city:\n', revenue_by_city)
```

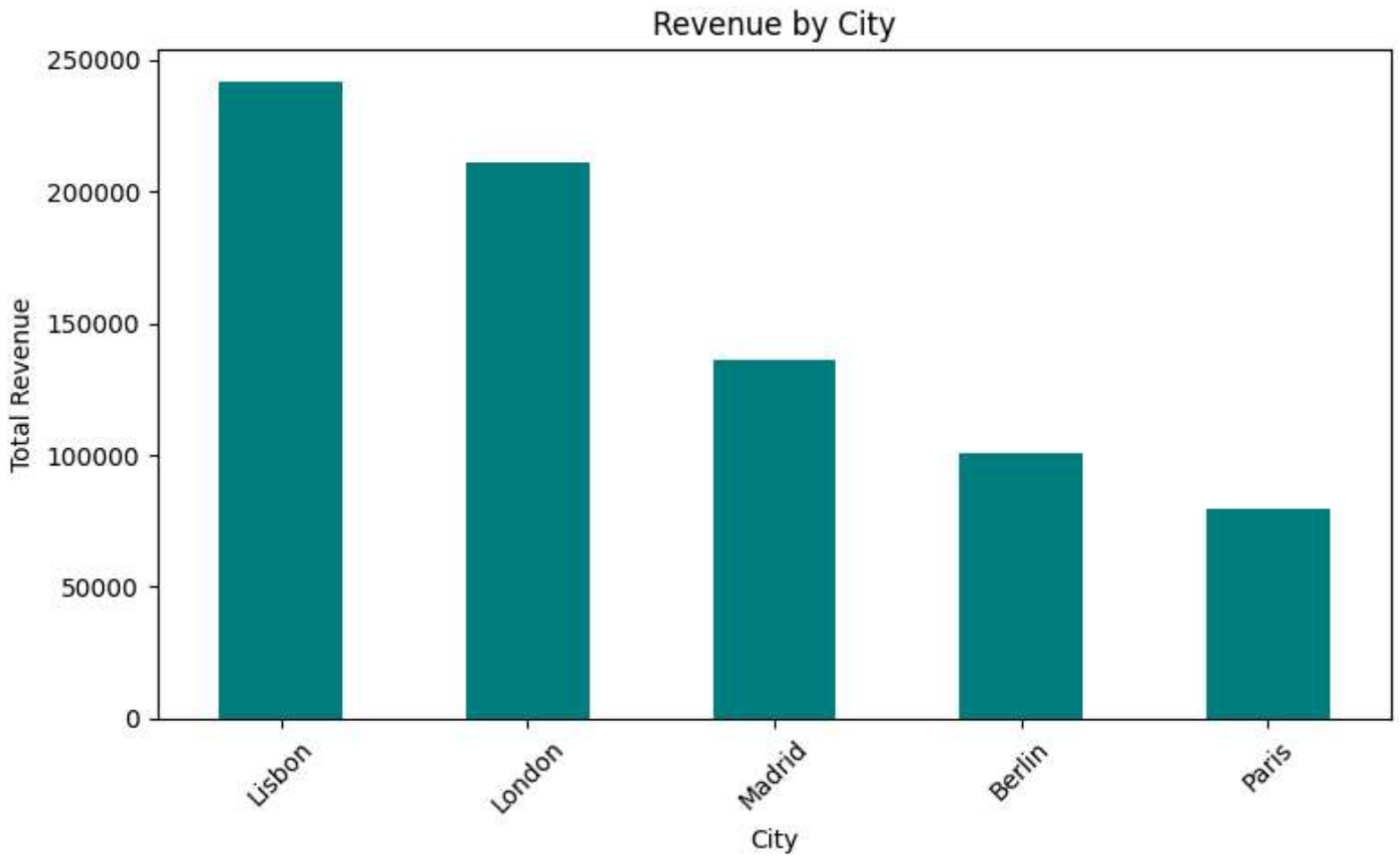
```
Revenur by city:
 City
Lisbon    241714.1157
London    211201.0406
Madrid    136200.2665
Berlin    100600.1313
Paris      79800.3051
Name: Total Amount, dtype: float64
```

In [91]:
```
top_city = revenue_by_city.idxmax()
top_city_revenue = revenue_by_city.max()
```

```
print(f"\n City with Maximum Revenue: {top_city} (₹{top_city_revenue:.2f})")
```

```
City with Maximum Revenue: Lisbon (₹241714.12)
```

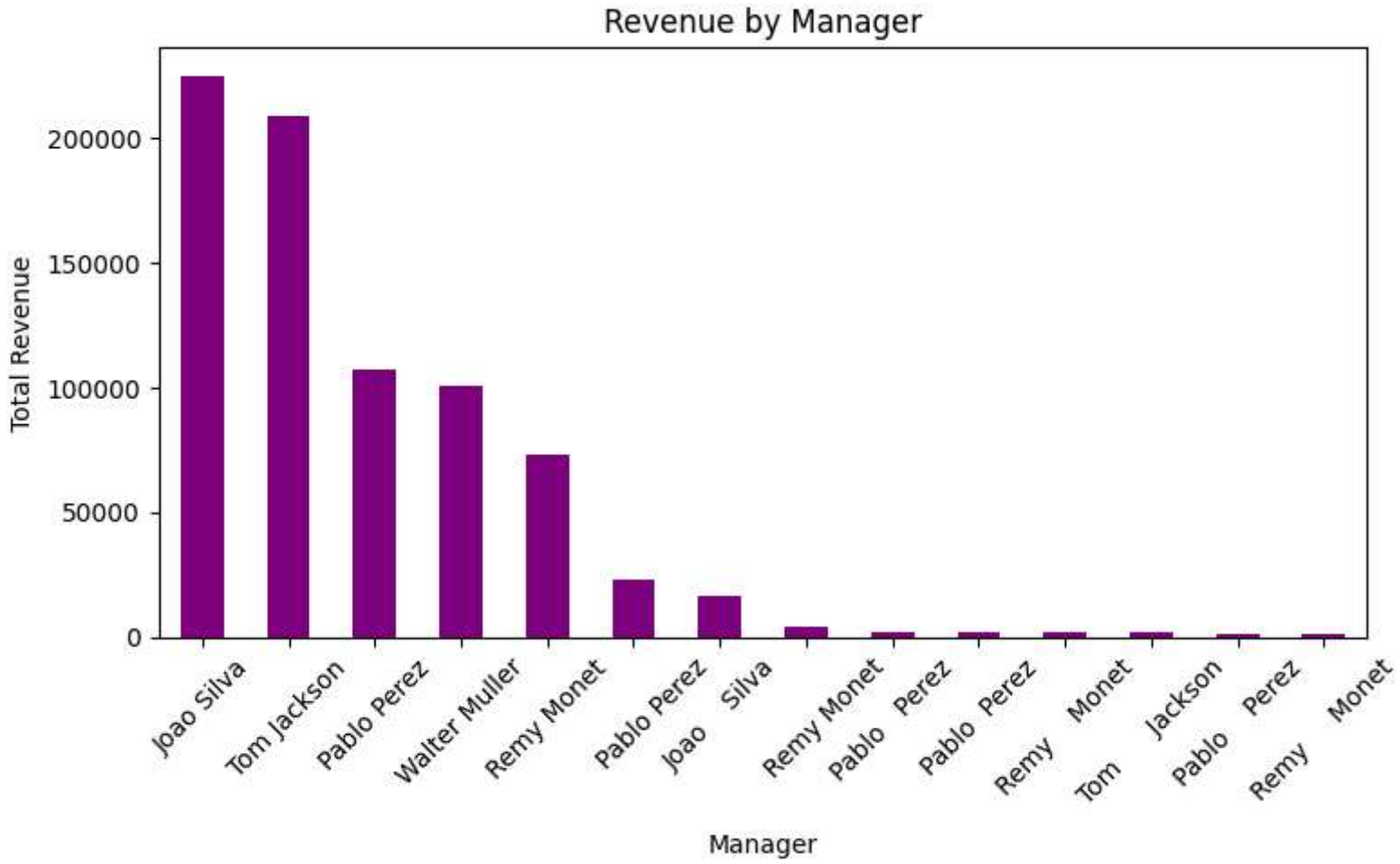
In [92]:
```
revenue_by_city.plot(kind='bar', color='teal', figsize=(8,5))
plt.title('Revenue by City')
plt.xlabel('City')
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [93]:
```
# Total revenue by manager
revenue_by_manager = df.groupby('Manager')['Total Amount'].sum().sort_values(ascending=False)

# Display result
print("Revenue by Manager:\n", revenue_by_manager)

# Top manager
top_manager = revenue_by_manager.idxmax()
top_manager_revenue = revenue_by_manager.max()
print(f"\n👨‍💼 Manager with Maximum Revenue: {top_manager} (₹{top_manager_revenue:.2f})")
```

```
Revenue by Manager:
 Manager
Joao Silva            225074.7665
Tom Jackson           209201.0263
Pablo Perez           107600.2549
Walter Muller         100600.1313
Remy Monet             72800.2690
       Pablo Perez      23200.0330
Joao    Silva           16639.3492
       Remy Monet        4000.0258
Pablo   Perez            2199.9913
Pablo  Perez             2199.9913
Remy    Monet            2000.0143
Tom      Jackson         2000.0143
Pablo    Perez            999.9960
Remy     Monet            999.9960
Name: Total Amount, dtype: float64

👨‍💼 Manager with Maximum Revenue: Joao Silva (₹225074.77)
```

In [94]:
```
revenue_by_manager.plot(kind='bar', color='purple', figsize=(8,5))
plt.title('Revenue by Manager')
plt.xlabel('Manager')
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



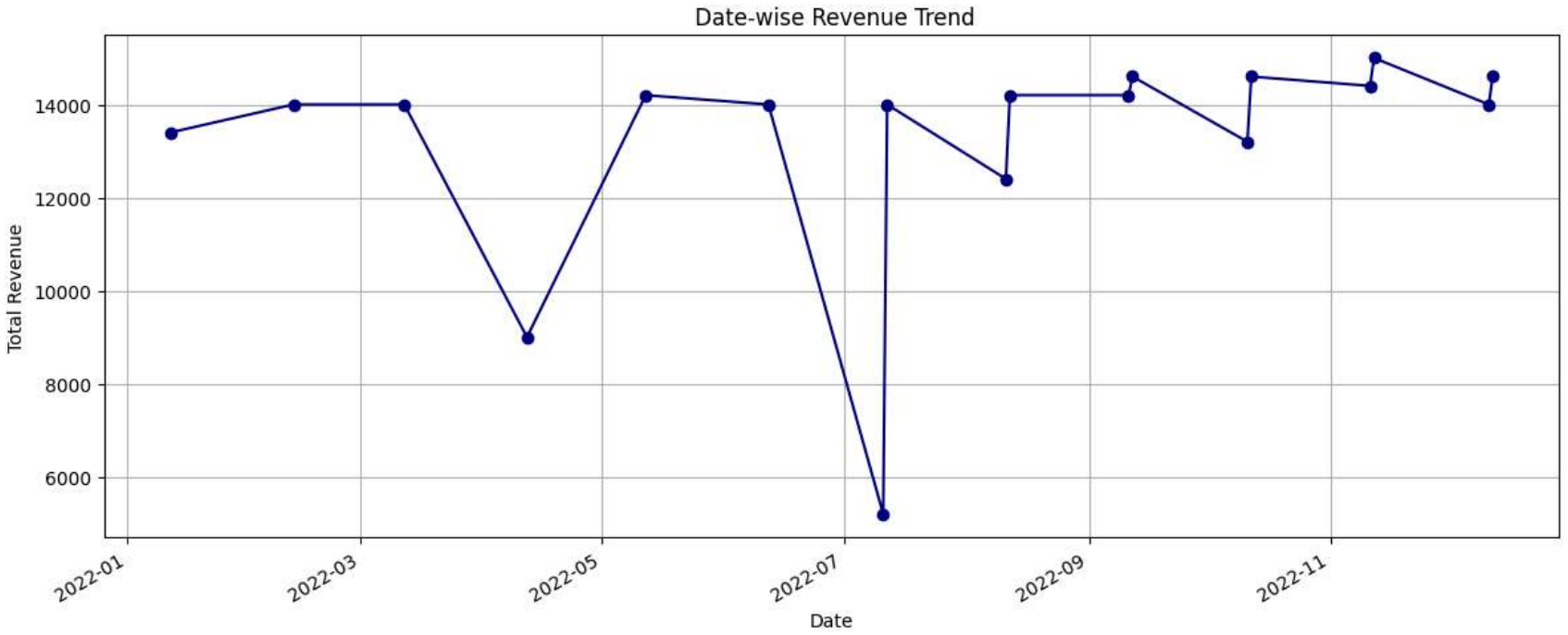
Q.4) Date wise revenue.

In [95]:
```
date_wise_revenue=df.groupby('Date')['Total Amount'].sum().sort_index()
```

In [96]:
```
print('date_wise_revenue:\n' ,date_wise_revenue.head())
```

```
date_wise_revenue:
 Date
2022-01-12    13400.1144
2022-02-12    14000.0535
2022-03-12    14000.0535
2022-04-12     9000.1007
2022-05-12    14200.0386
Name: Total Amount, dtype: float64
```

In [97]:
```
plt.figure(figsize=(12,5))
date_wise_revenue.plot(kind='line', marker='o', color='navy')
plt.title(' Date-wise Revenue Trend')
plt.xlabel('Date')
plt.ylabel('Total Revenue')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Q.5) Average Revenue.

In [98]:
```
Average_Revenue_per_order = df['Total Amount'].mean()
```

In [99]:
```
print('Average Revenue.\n', Average_Revenue_per_order)
```

```
Average Revenue.
 3029.589996850394
```

In [100…
```
daily_revenue = df.groupby('Date')['Total Amount'].sum()
```

In [101…
```
daily_revenue.mean()
```

Out[101…
```
13277.840822222222
```

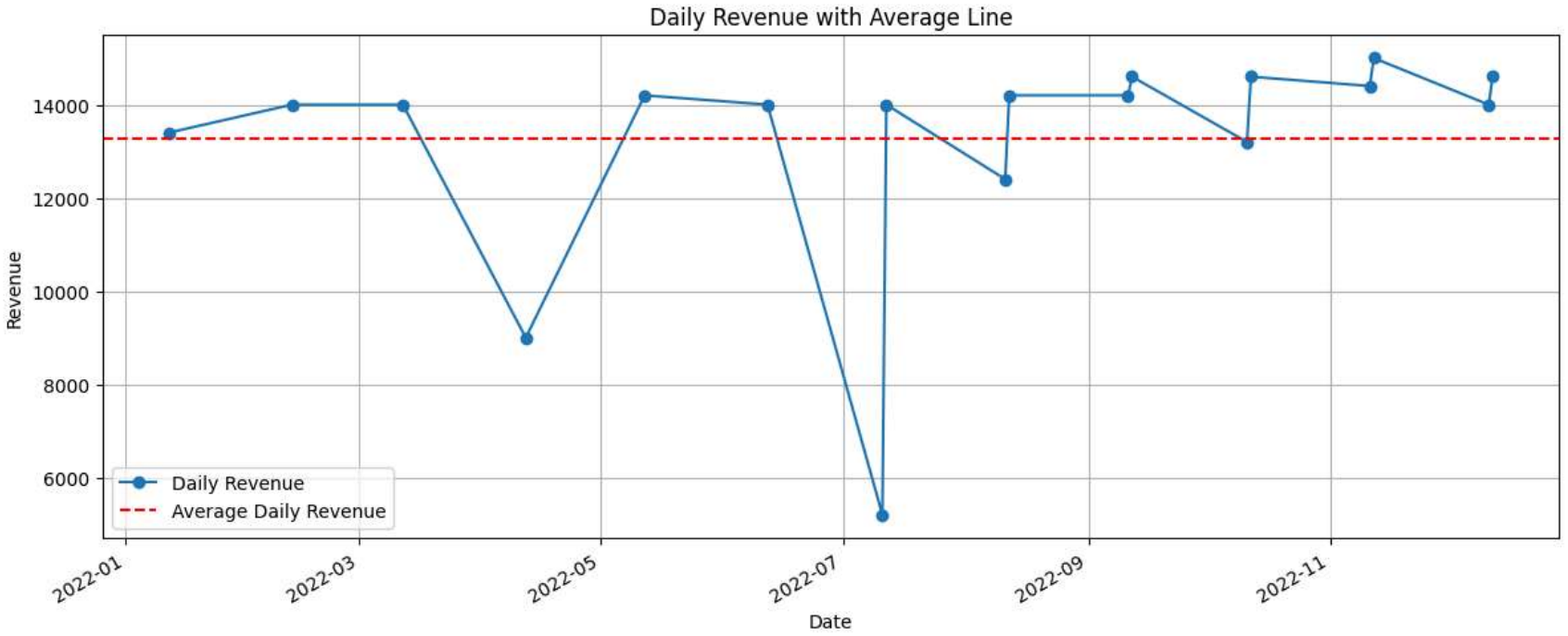
In [102…
```
average_revenue_per_day = daily_revenue.mean()
```

In [103…
```
manager_avg_revenue = df.groupby('Manager')['Total Amount'].mean()
print(" Average Revenue per Order by Manager:\n", manager_avg_revenue)
```

```
 Average Revenue per Order by Manager:
 Manager
       Pablo Perez    3314.290429
       Remy Monet     2000.012900
Joao    Silva         3327.869840
Joao Silva            3215.353807
Pablo    Perez         999.996000
Pablo   Perez         2199.991300
Pablo  Perez          2199.991300
Pablo Perez           2988.895969
Remy     Monet         999.996000
Remy    Monet         2000.014300
Remy Monet            3033.344542
Tom      Jackson      2000.014300
Tom Jackson           2827.040896
Walter Muller         3353.337710
Name: Total Amount, dtype: float64
```

In [104…
```
import matplotlib.pyplot as plt

plt.figure(figsize=(12,5))
daily_revenue.plot(label='Daily Revenue', marker='o')
plt.axhline(average_revenue_per_day, color='red', linestyle='--', label='Average Daily Revenue')
```

```
plt.title('Daily Revenue with Average Line')
plt.xlabel('Date')
plt.ylabel('Revenue')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



Q.6) Average Revenue of November & December month.

In [105…

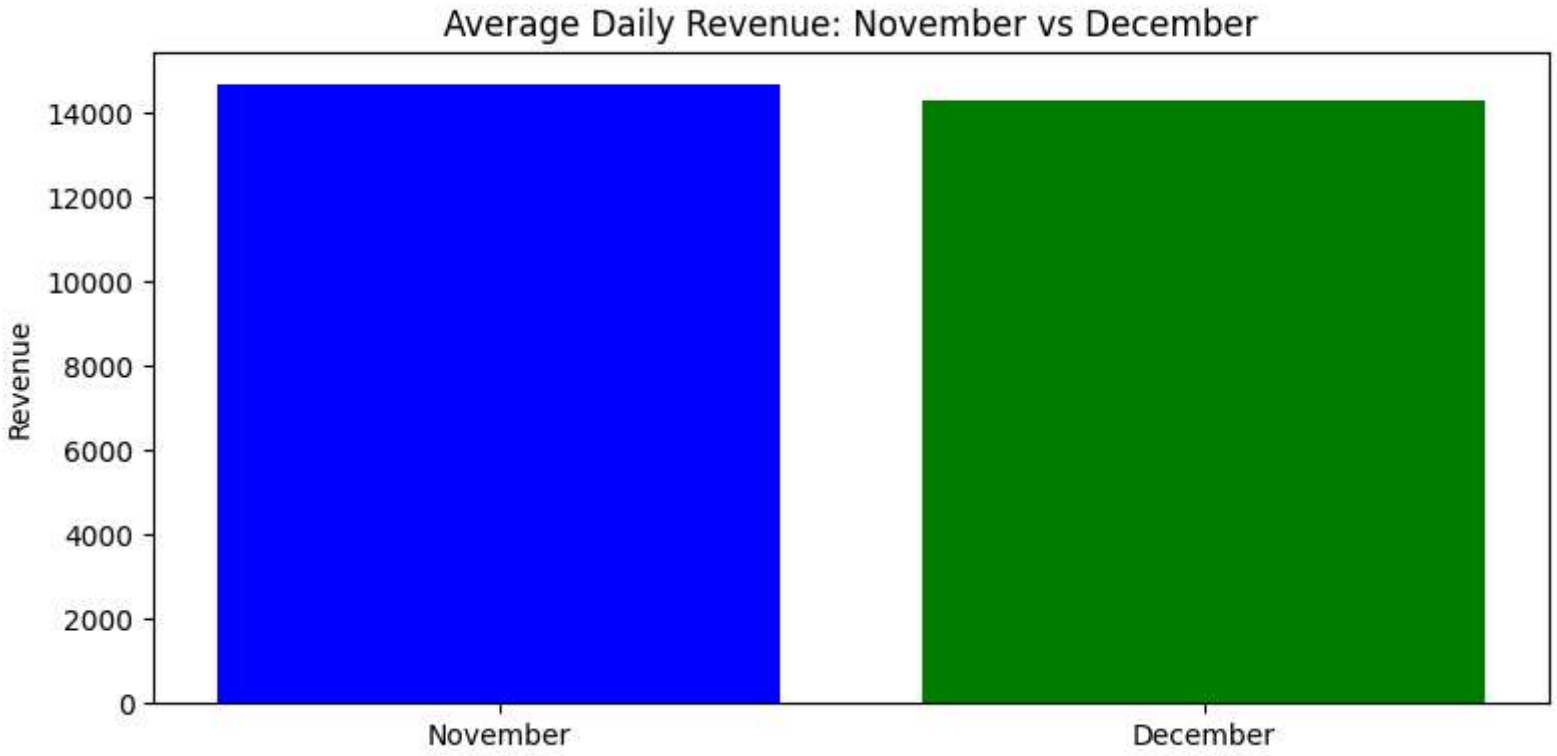
```
# Ensure Date column is datetime
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Extract month as a new column
df['Month'] = df['Date'].dt.month

# Filter for November (11) and December (12)
november_data = df[df['Month'] == 11]
december_data = df[df['Month'] == 12]
```

```
# Group by Date to get daily revenue
november_daily_revenue = november_data.groupby('Date')['Total Amount'].sum()
december_daily_revenue = december_data.groupby('Date')['Total Amount'].sum()

# Calculate average daily revenue for each month
avg_november_revenue = november_daily_revenue.mean()
avg_december_revenue = december_daily_revenue.mean()

print(f" Average Daily Revenue in November: ₹{avg_november_revenue:.2f}")
print(f" Average Daily Revenue in December: ₹{avg_december_revenue:.2f}")
```

```
Average Daily Revenue in November: ₹14700.05
Average Daily Revenue in December: ₹14300.06
```

In [106…

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8,4))
plt.bar(['November', 'December'], [avg_november_revenue, avg_december_revenue], color=['blue', 'green'])
plt.title('Average Daily Revenue: November vs December')
plt.ylabel('Revenue')
plt.tight_layout()
plt.show()
```



Q.7) Standard Deviation of Revenue and Quantity ?

In [107…

```
## # Standard Deviation of Total Revenue
std_revenue = df['Total Amount'].std()

# Standard Deviation of Quantity
std_quantity = df['Quantity'].std()

print(f" Standard Deviation of Revenue: ₹{std_revenue:.2f}")
print(f" Standard Deviation of Quantity: {std_quantity:.2f}")
```

```
Standard Deviation of Revenue: ₹2420.12
Standard Deviation of Quantity: 214.89
```

Q.8) Variance of Revenue and Quantity ?

In [108…
```python
# Variance of Total Revenue
var_revenue = df['Total Amount'].var()

# Variance of Quantity
var_quantity = df['Quantity'].var()

print(f"Variance of Revenue: ₹{var_revenue:.2f}")
print(f"Variance of Quantity: {var_quantity:.2f}")
```

```
Variance of Revenue: ₹5856972.96
Variance of Quantity: 46177.15
```

Q.9) Is revenue increasing or decreasing over time?

In [109…
```python
daly_revenue=df=df.groupby('Date')['Total Amount'].sum().sort_index()
```

In [110…
```python
plt.figure(figsize=(12,5))
daily_revenue.plot(kind='line', marker='o', color='darkgreen')
plt.title('Revenue Trend Over Time')
plt.xlabel('Date')
plt.ylabel('Total Revenue')
plt.grid(True)
plt.tight_layout()
plt.show()
```

Q.10) Average 'Quantity Sold' & 'Average Revenue' for each product ?

In [ ]:

In [ ]: