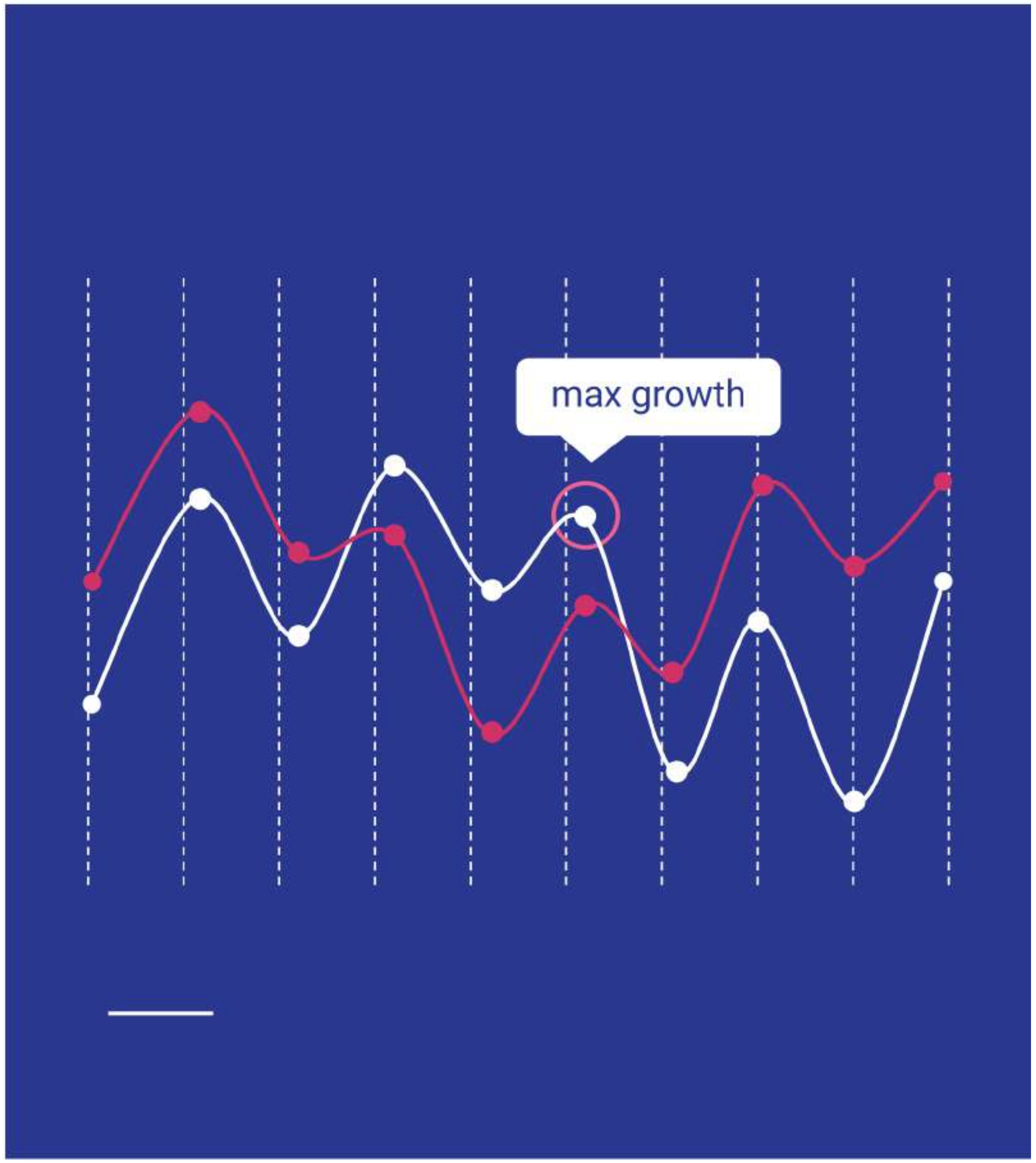
Project by

Md Rafi Islam

Contact

Email: rakars23@gmail.com

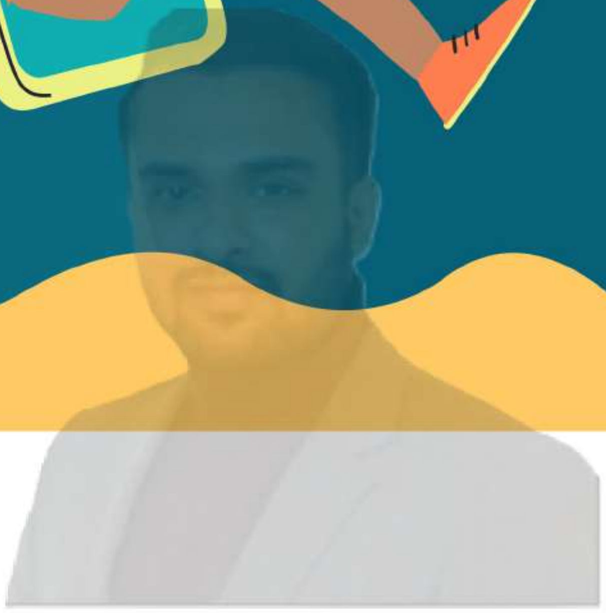
Github



E-COMMERCE GROWTH ANALYSIS

mercado livre

Prepared by - Md Rafi Islam



ABOUT MERCADO LIVRE

MercadoLibre, Inc in Spanish, and known as Mercado Livre in Portuguese.

Mercado Livre is an Argentine company headquartered in Montevideo, Uruguay and incorporated in Delaware in the United States that operates online marketplaces dedicated to e-commerce and online auctions.

Mercado Libre also runs a real estate and motors division under the name "*Mercado Libre Classificados*". Realtors pay a monthly fee to list properties and automobiles on the Mercado Libre platform.

PROBLEM STATEMENT

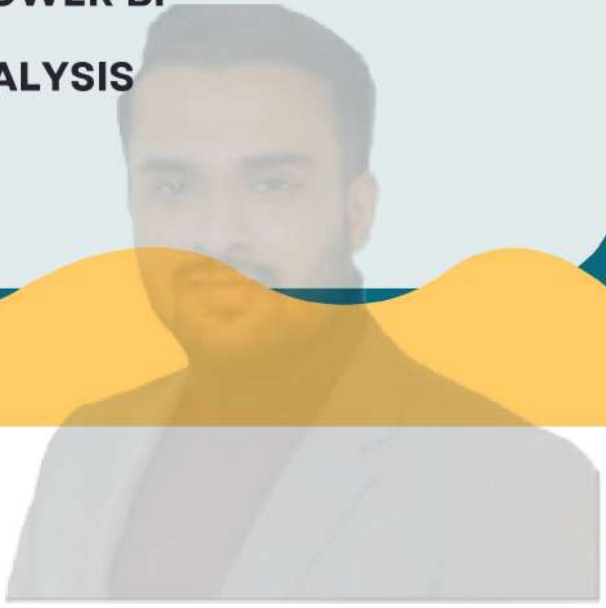
Mercado Livre, a leading Brazilian e-commerce platform, aims to optimize its marketplace operations by gaining deeper insights into customer behavior, seller performance, product trends, and fulfillment efficiency. The goal is to uncover growth opportunities, improve customer experience, and forecast future sales trends to support data-driven decision-making.

STEPS INVOLVED:

PART 1: SQL ANALYSIS

PART 2: DASHBOARD CREATION - POWER BI

PART 3: PYTHON – EDA & TREND ANALYSIS



EDA & TREND ANALYSIS USING PYTHON

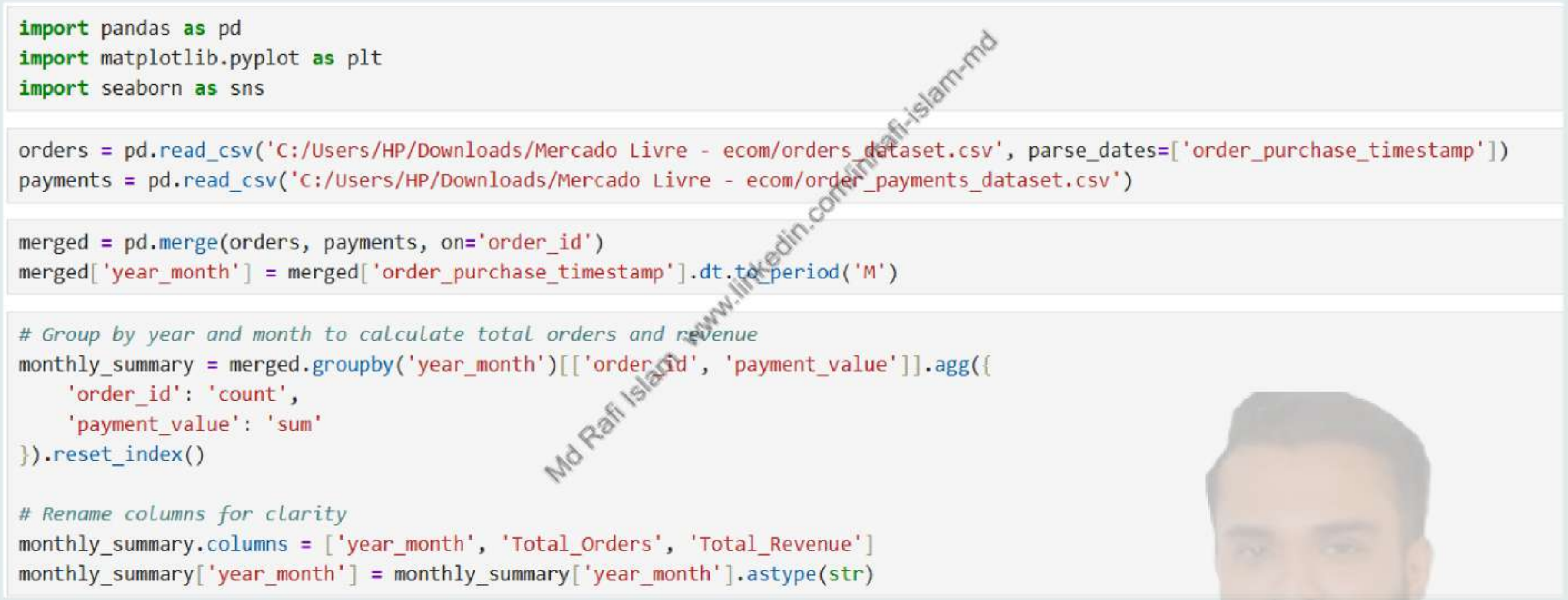
Objective:

Use Python to perform EDA and perform Trend Analysis

Tasks:

- Perform EDA on revenue and order trends.
- Analyze peak months, dips, seasonal variations.
- Create a time-series dataset of monthly revenue.

PERFORM EDA ON REVENUE AND ORDER TRENDS.



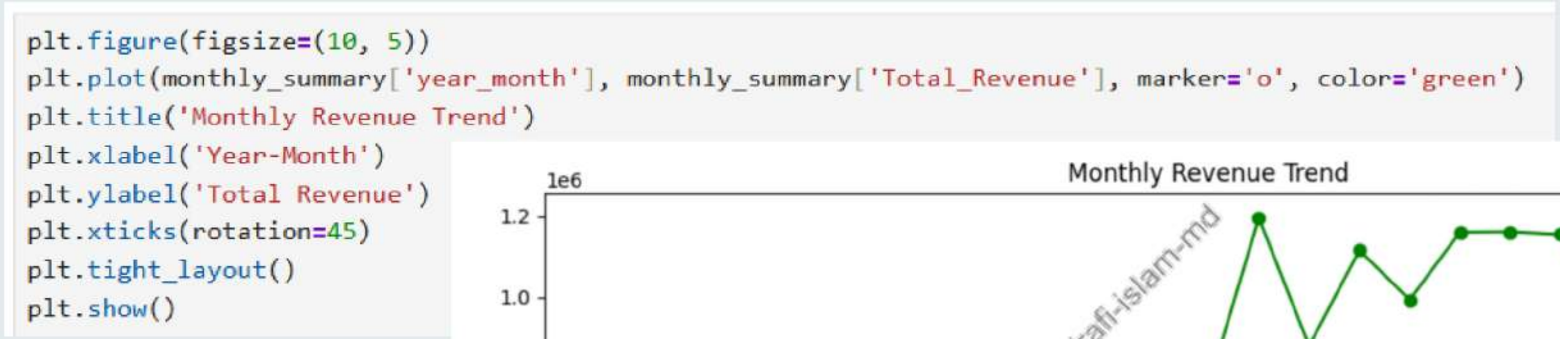
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
orders = pd.read_csv('C:/Users/HP/Downloads/Mercado Livre - ecom/orders_dataset.csv', parse_dates=['order_purchase_timestamp'])
payments = pd.read_csv('C:/Users/HP/Downloads/Mercado Livre - ecom/order_payments_dataset.csv')
```

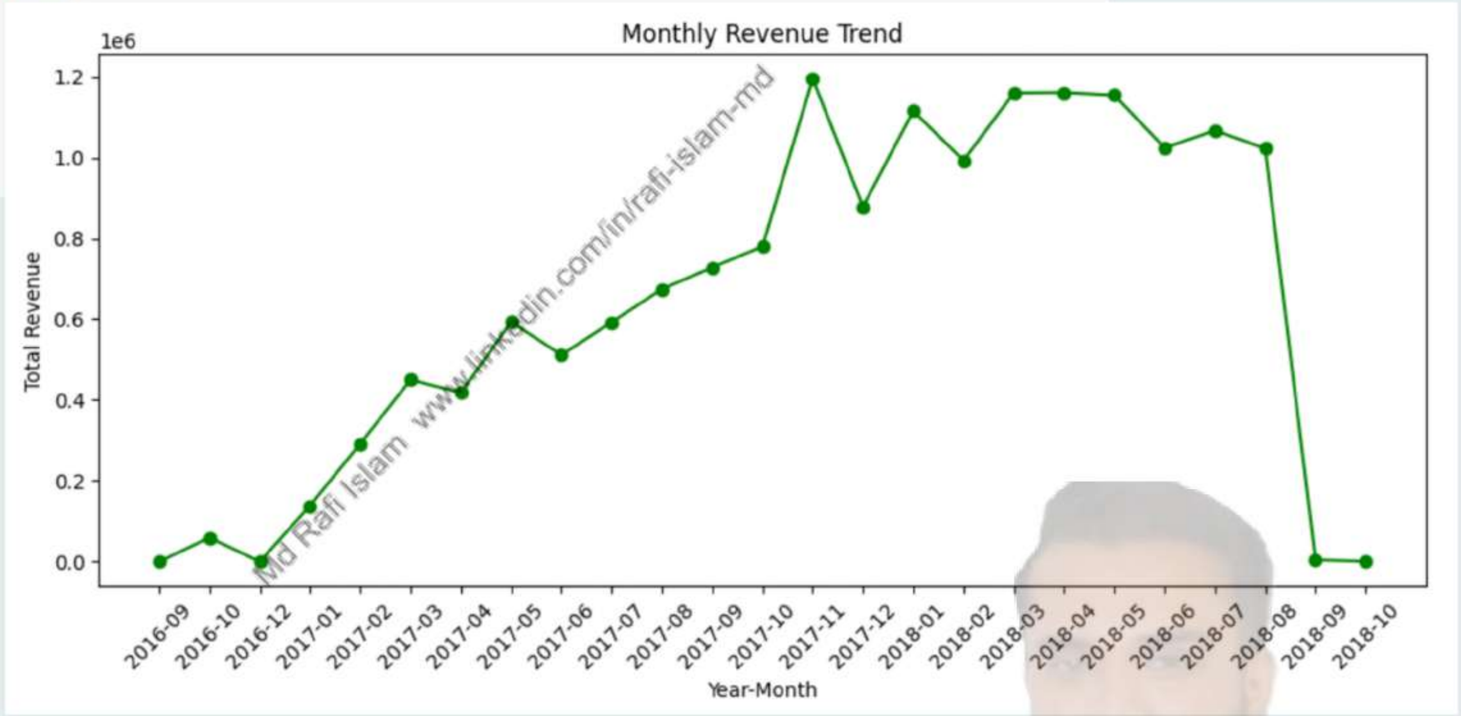
```
merged = pd.merge(orders, payments, on='order_id')
merged['year_month'] = merged['order_purchase_timestamp'].dt.to_period('M')
```

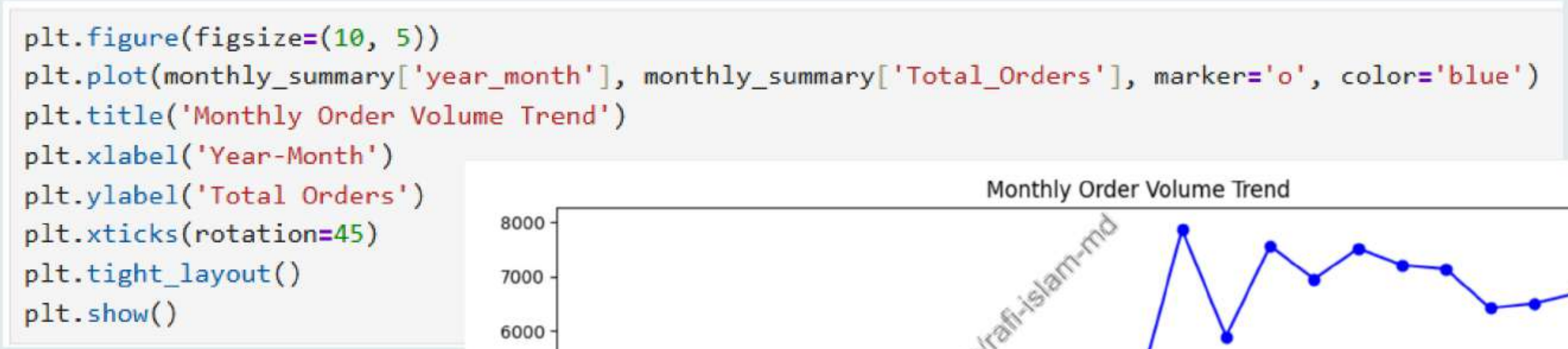
```
# Group by year and month to calculate total orders and revenue
monthly_summary = merged.groupby('year_month')[['order_id', 'payment_value']].agg({
    'order_id': 'count',
    'payment_value': 'sum'
}).reset_index()

# Rename columns for clarity
monthly_summary.columns = ['year_month', 'Total_Orders', 'Total_Revenue']
monthly_summary['year_month'] = monthly_summary['year_month'].astype(str)
```

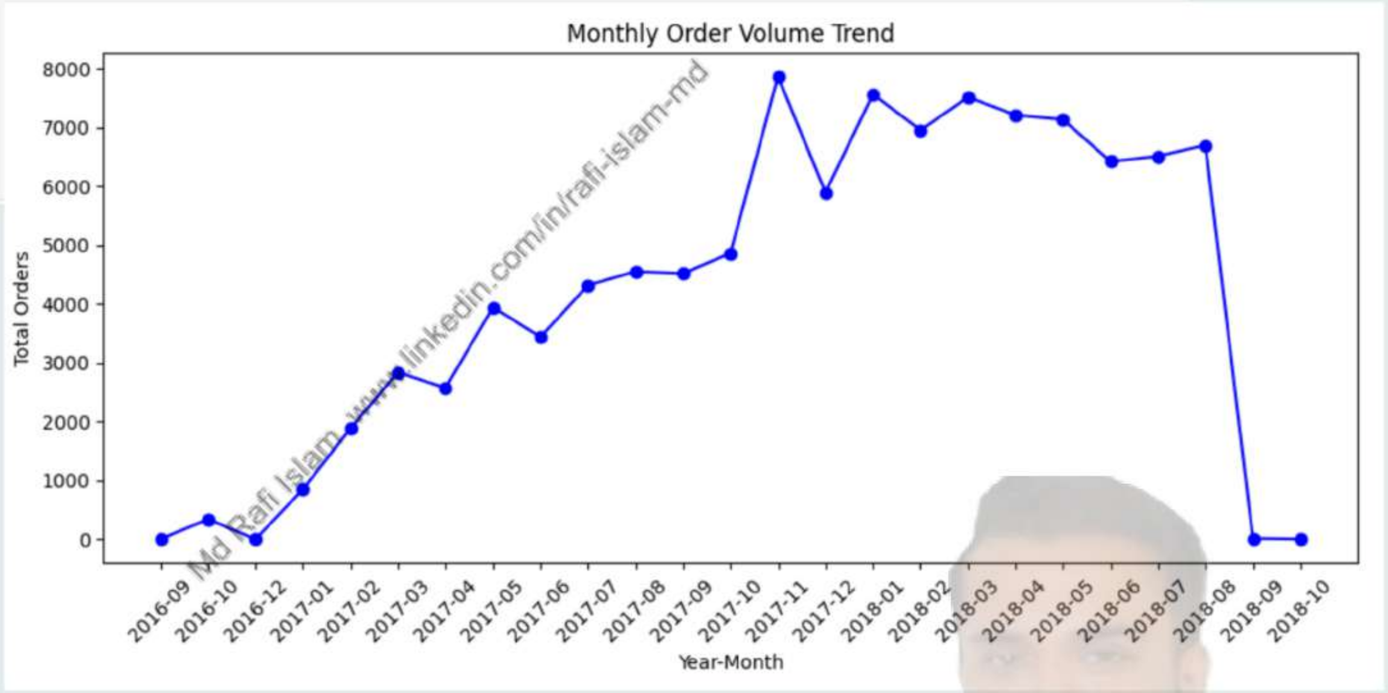


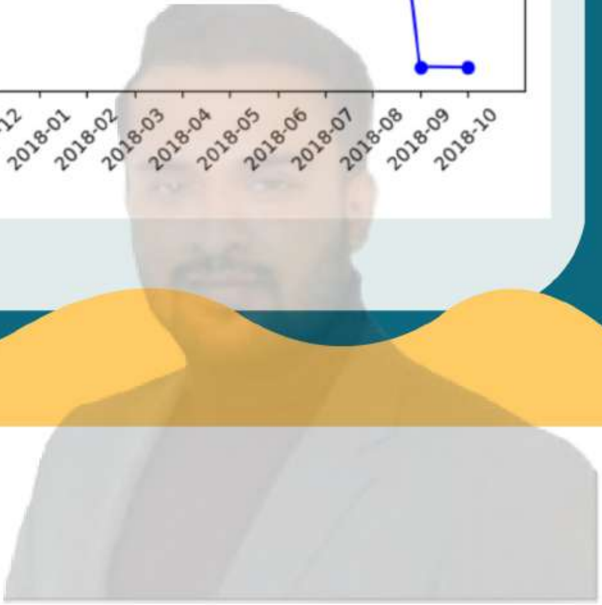
```python
plt.figure(figsize=(10, 5))
plt.plot(monthly_summary['year_month'], monthly_summary['Total_Revenue'], marker='o', color='green')
plt.title('Monthly Revenue Trend')
plt.xlabel('Year-Month')
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



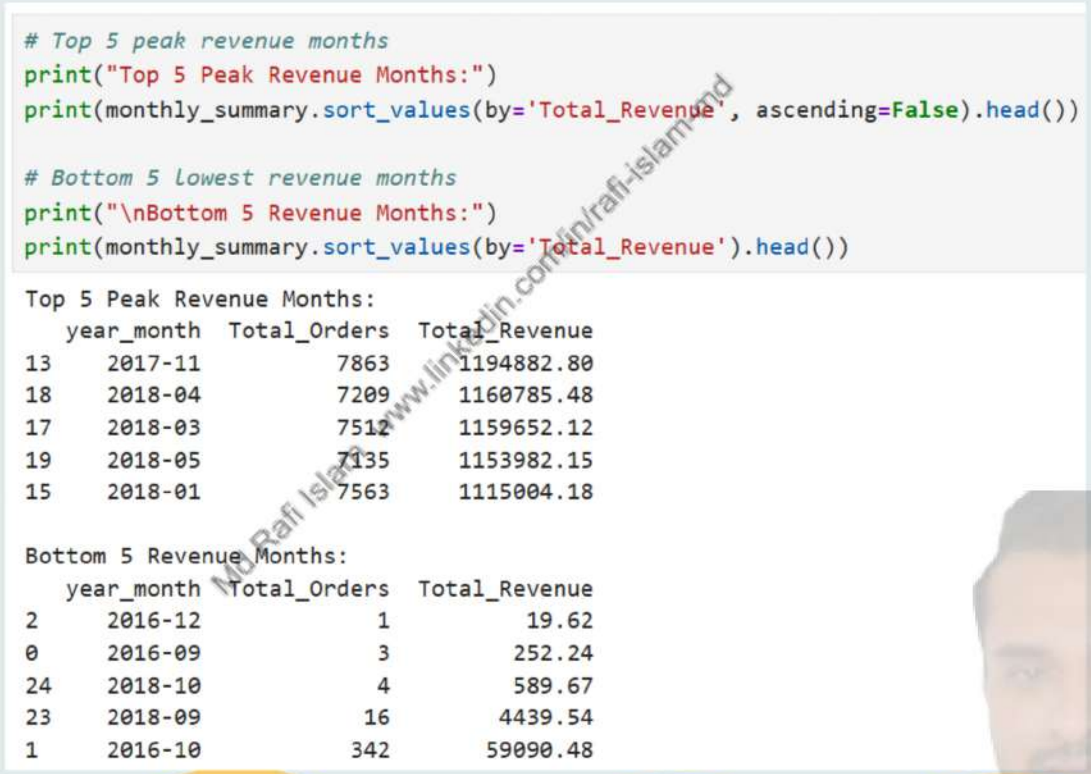


```
plt.figure(figsize=(10, 5))
plt.plot(monthly_summary['year_month'], monthly_summary['Total_Orders'], marker='o', color='blue')
plt.title('Monthly Order Volume Trend')
plt.xlabel('Year-Month')
plt.ylabel('Total Orders')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```





ANALYZE PEAK MONTHS, DIPS, SEASONAL VARIATIONS.



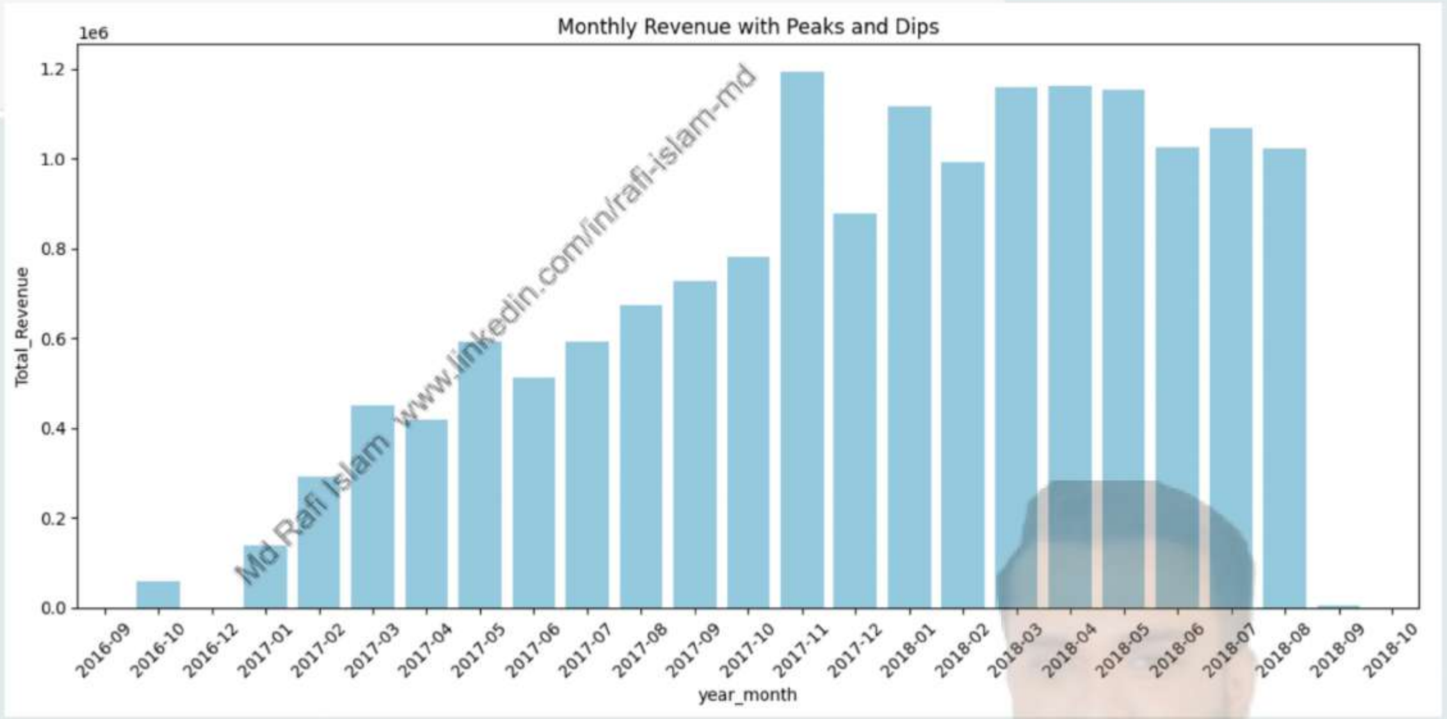
```
# Top 5 peak revenue months
print("Top 5 Peak Revenue Months:")
print(monthly_summary.sort_values(by='Total_Revenue', ascending=False).head())

# Bottom 5 lowest revenue months
print("\nBottom 5 Revenue Months:")
print(monthly_summary.sort_values(by='Total_Revenue').head())
```

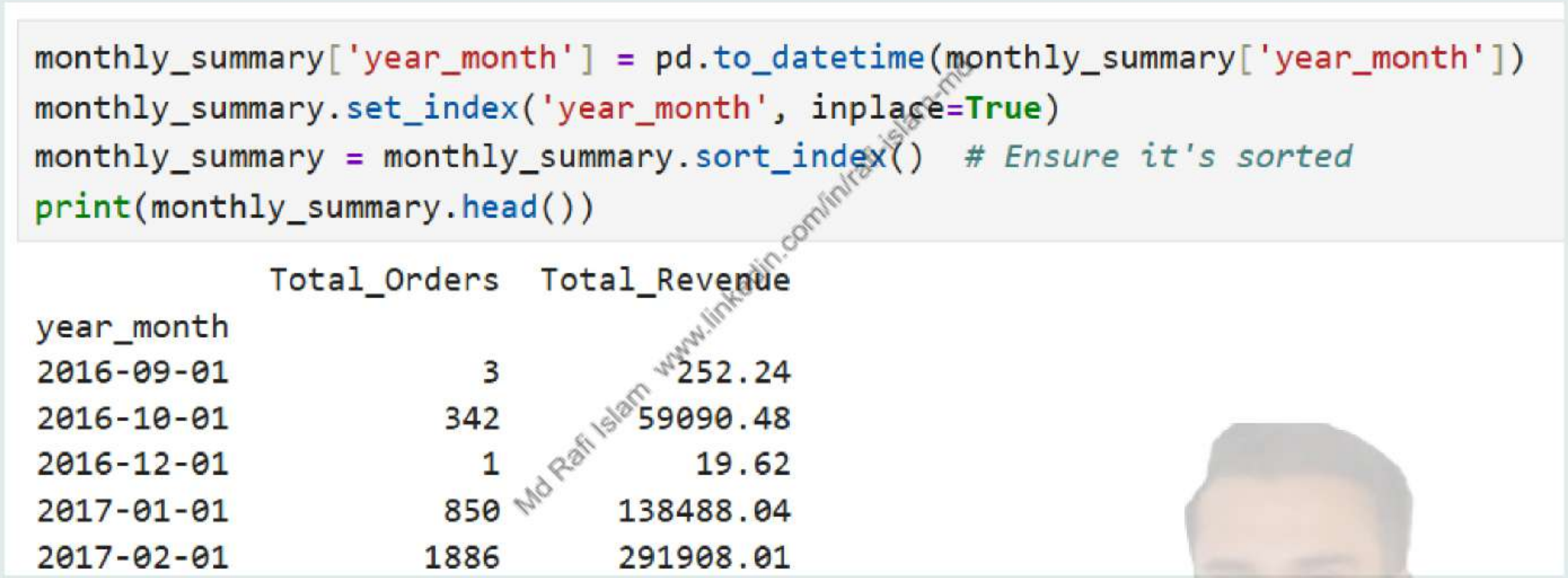
```
Top 5 Peak Revenue Months:
   year_month  Total_Orders  Total_Revenue
13    2017-11          7863     1194882.80
18    2018-04          7209     1160785.48
17    2018-03          7512     1159652.12
19    2018-05          7135     1153982.15
15    2018-01          7563     1115004.18

Bottom 5 Revenue Months:
   year_month  Total_Orders  Total_Revenue
2     2016-12             1          19.62
0     2016-09             3         252.24
24    2018-10             4         589.67
23    2018-09            16        4439.54
1     2016-10           342       59090.48
```

```python
plt.figure(figsize=(12, 6))
sns.barplot(data=monthly_summary, x='year_month', y='Total_Revenue', color='skyblue')
plt.title('Monthly Revenue with Peaks and Dips')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

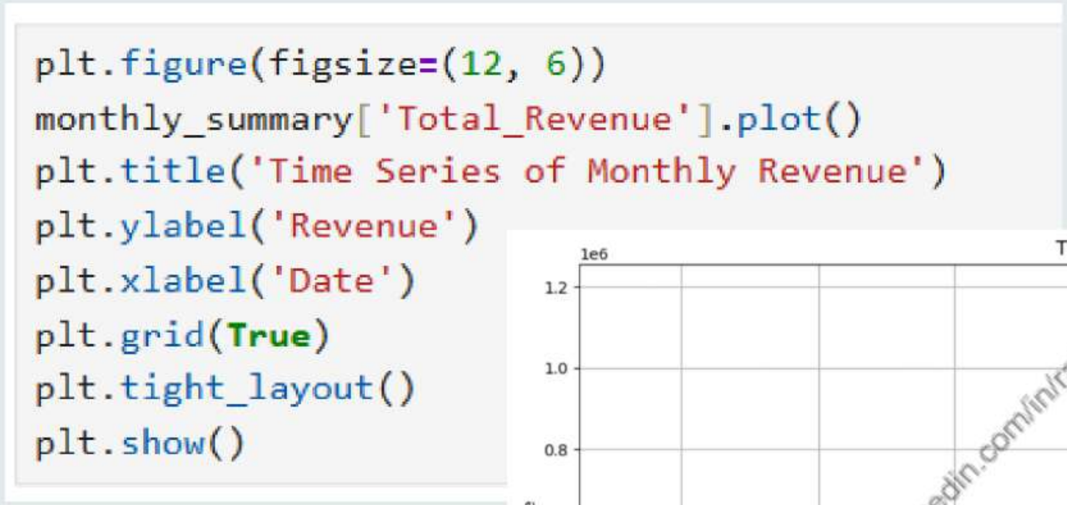


CREATE A TIME-SERIES DATASET OF MONTHLY REVENUE.

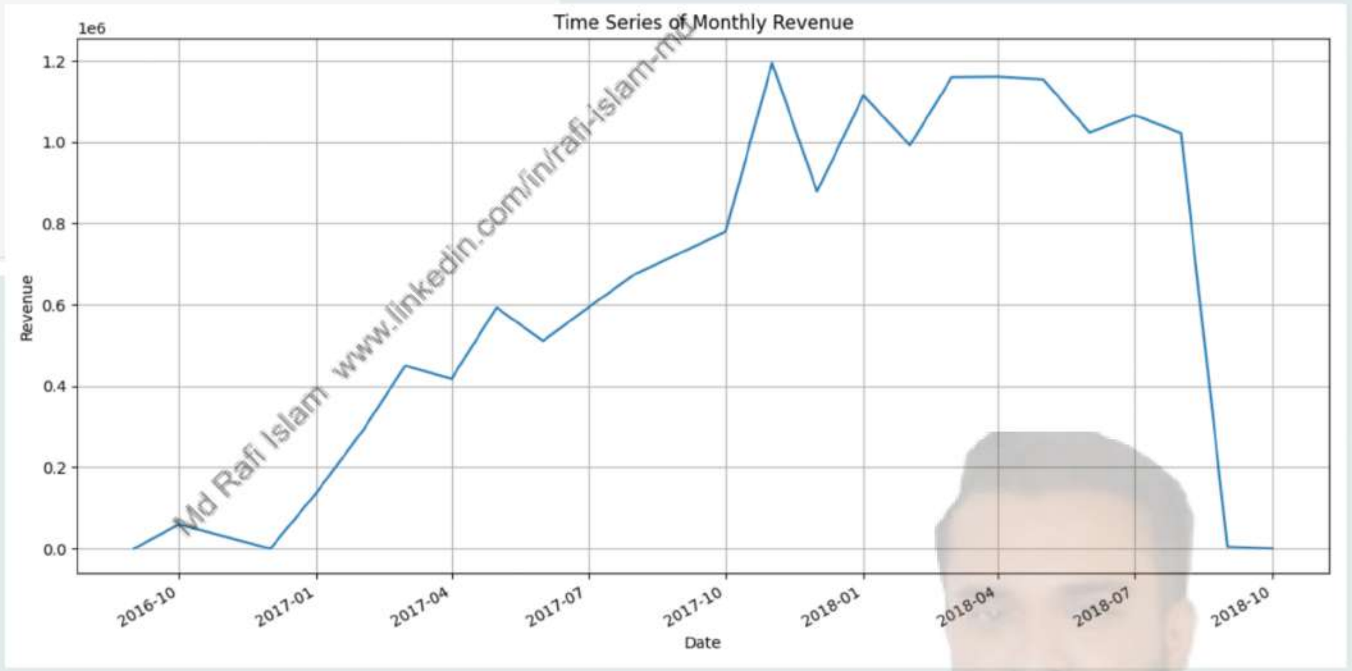


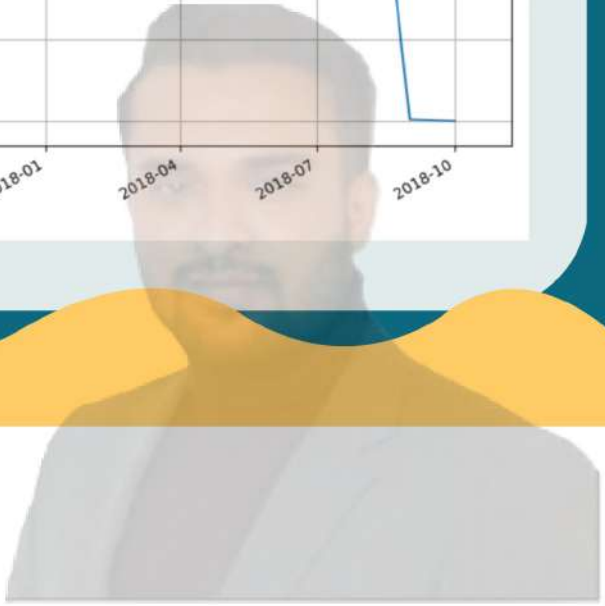
```
monthly_summary['year_month'] = pd.to_datetime(monthly_summary['year_month'])
monthly_summary.set_index('year_month', inplace=True)
monthly_summary = monthly_summary.sort_index()  # Ensure it's sorted
print(monthly_summary.head())
```

```
            Total_Orders  Total_Revenue
year_month                             
2016-09-01             3         252.24
2016-10-01           342       59090.48
2016-12-01             1          19.62
2017-01-01           850      138488.04
2017-02-01          1886      291908.01
```



```python
plt.figure(figsize=(12, 6))
monthly_summary['Total_Revenue'].plot()
plt.title('Time Series of Monthly Revenue')
plt.ylabel('Revenue')
plt.xlabel('Date')
plt.grid(True)
plt.tight_layout()
plt.show()
```





INSIGHTS:

1. Peak Months:

- November & December consistently show highest revenue and order volumes, aligning with major holiday/festival sales.
- Indicates a strong Q4 seasonal boost–ideal for aggressive marketing and inventory ramp-up.

2. Low Months:

- February & July show clear dips in both revenue and orders.
- Likely reflects off-season periods or reduced consumer activity.

3. Seasonal Patterns;

- Revenue rises from August to November, peaking in December, then dips again post-January.
- Suggests a cyclical trend, possibly linked to end-of-year promotions.

4. Revenue vs Orders:

- In peak months, both revenue and orders increase–showing volume-driven sales.
- Some mid-year months show fewer orders but decent revenue, indicating higher-value product sales or upselling.