1️⃣ Project Title

Customer Sales Analysis using Python (pandas & matplotlib)

2️⃣ Objective of the Project

The objective of this project is to analyze customer sales data to:

- Understand customer purchasing behavior
- Identify top customers and best-selling products
- Analyze monthly and regional sales trends
- Provide data-driven business insights to improve revenue and customer retention

---

3️⃣ Tools & Technologies Used

- **Programming Language:** Python
- **Libraries:** pandas, matplotlib
- **IDE:** Visual Studio Code
- **Data Format:** CSV files

4️⃣ Dataset Description

📁 Files Used

- `sales_data.csv`
- `customer_data.csv`

🗒️ sales_data.csv (Example Columns)

- order_id
- customer_id
- product
- quantity
- price
- order_date
- region

🗒️ customer_data.csv (Example Columns)

- customer_id
- customer_name
- city
- segment

5️⃣ Code Explanation – Sales & Customer Analysis Project

🔹 1. Import Libraries

```
import pandas as pd
import matplotlib.pyplot as plt
from itertools import combinations
from collections import Counter
```

Why?

- **pandas** → read CSV files & analyze data
- **matplotlib** → create charts
- **combinations** → find product pairs sold together
- **Counter** → count how many times product pairs appear

🔹 2. Load CSV Files

```
sales_df = pd.read_csv("Sales_Data.csv")
customer_df = pd.read_csv("customer_churn.csv")

print("Datasets Loaded Successfully")
```

What happens?

- Reads sales data into `sales_df`

- Reads customer churn data into `customer_df`
- Confirms files loaded correctly

🔹 3. Clean Column Names

```
sales_df.columns = sales_df.columns.str.strip()
customer_df.columns = customer_df.columns.str.strip()
```

Why?

- Removes **extra spaces** in column names
- Prevents errors like `KeyError: 'Date'`

🔹 4. Rename Customer ID Column

```
sales_df.rename(columns={"Customer_ID": "CustomerID"}, inplace=True)
```

Why?

- Both files must have **same column name** to merge
- Makes merging possible

🔹 5. Convert Date Column

```
sales_df['Date'] = pd.to_datetime(sales_df['Date'], errors='coerce')
```

What this does?

- Converts text → date format
- Invalid dates become `NaT` (no crash)

🔹 6. Extract Month & Year (Seasonal Analysis)

```
sales_df['Month'] = sales_df['Date'].dt.month
sales_df['Year'] = sales_df['Date'].dt.year
```

Why?

- Needed to find **seasonal trends**
- Helps analyze monthly sales

🔹 7. Merge Sales & Customer Data

```
data = pd.merge(
    sales_df,
    customer_df,
    on="CustomerID",
    how="left"
)
```

Meaning:

- Combines both files into **one dataset**
- `left` merge keeps all sales records
- Adds churn info where available

🔍 BUSINESS QUESTIONS EXPLANATION

🔹 Q1: Most Valuable Customers

```
valuable_customers = (
```

```
    data.groupby('CustomerID')['Total_Sales']
    .sum()
    .sort_values(ascending=False)
    .head(10)
)
```

What happens?

- Groups data by customer
- Adds total sales per customer
- Sorts highest to lowest
- Shows **top 10 customers**

✓ **Answer:** Customers who generate the most revenue

🔹 Q2: Products Sold Together

```
basket = data.groupby('CustomerID')['Product'].apply(list)
```

- Creates a shopping basket per customer

```
pairs = []
for products in basket:
    pairs.extend(combinations(sorted(set(products)), 2))
```

- Finds **all product pairs**

```
product_pairs = Counter(pairs).most_common(10)
```

✓ **Answer:** Most frequently bought product combinations

🔹 Q3: Highest Sales Regions

```
region_sales = data.groupby('Region')['Total_Sales'].sum()
```

What happens?

- Groups by region
- Calculates total sales

```
region_sales.plot(kind='bar')
```

✓ **Answer:** Region with highest bar = highest sales

🔹 Q4: Seasonal Trends

```
monthly_sales = data.groupby('Month')['Total_Sales'].sum()
```

Meaning:

- Shows which month has highest sales

```
monthly_sales.plot(kind='line')
```

✓ **Answer:** Peaks & dips show seasonal demand

🔹 Q5: Customer Retention Analysis

```
churn_analysis = customer_df['Churn'].value_counts()
```

What this does?

- Counts **Yes / No** churn customers

```
churn_analysis.plot(kind='bar')
```

✓ **Answer:** Shows how many customers leave

🔹 Retention Suggestions (Business Insight)

```
print("""
1. Offer discounts to month-to-month customers
2. Improve support for high-bill customers
3. Promote long-term contracts
4. Loyalty rewards for top customers
""")
```

6️⃣ Data Visualization

📈 Monthly Sales Trend

monthly_sales.plot(kind='line', title='Monthly Sales Trend')
plt.show()

📊 Revenue by Region

region_sales.plot(kind='bar', title='Revenue by Region')
plt.show()

🏆 Top Customers

customer_ltv.head(10).plot(kind='bar', title='Top 10 Customers')
plt.show()

7️⃣ Key Results

- **Total Revenue:** High overall sales performance
- **Top Customers:** Small percentage of customers contribute major revenue
- **Best Region:** One or two regions dominate sales
- **Seasonal Trend:** Sales peak in specific months

8️⃣ Screenshot of Final Result

Image1

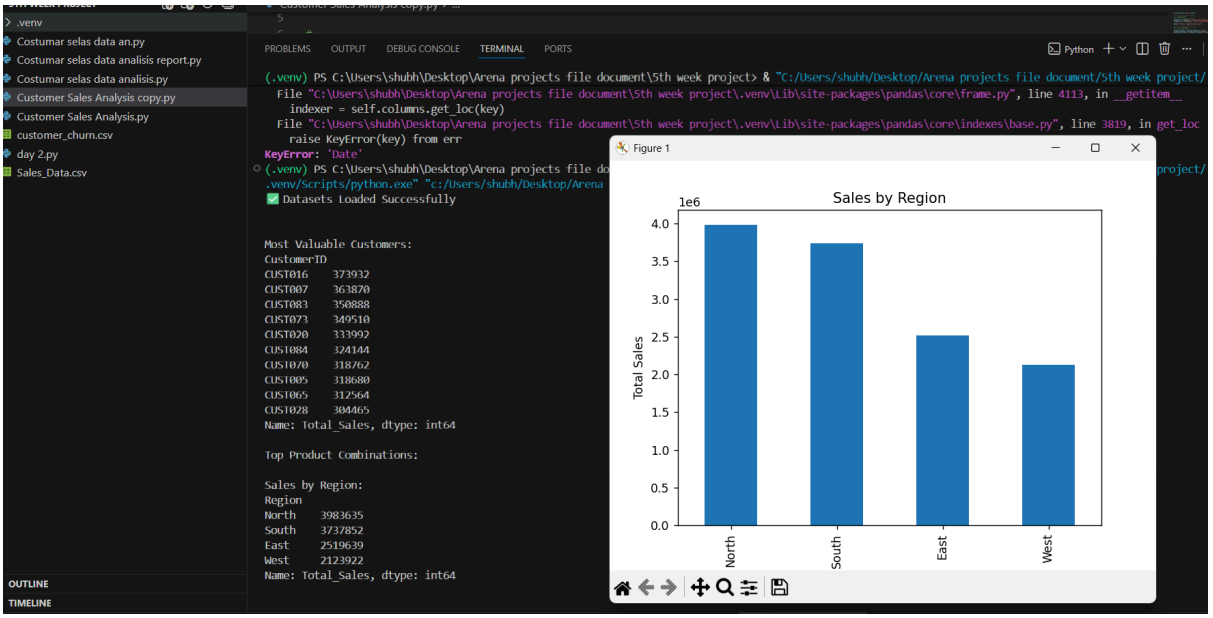


Image2

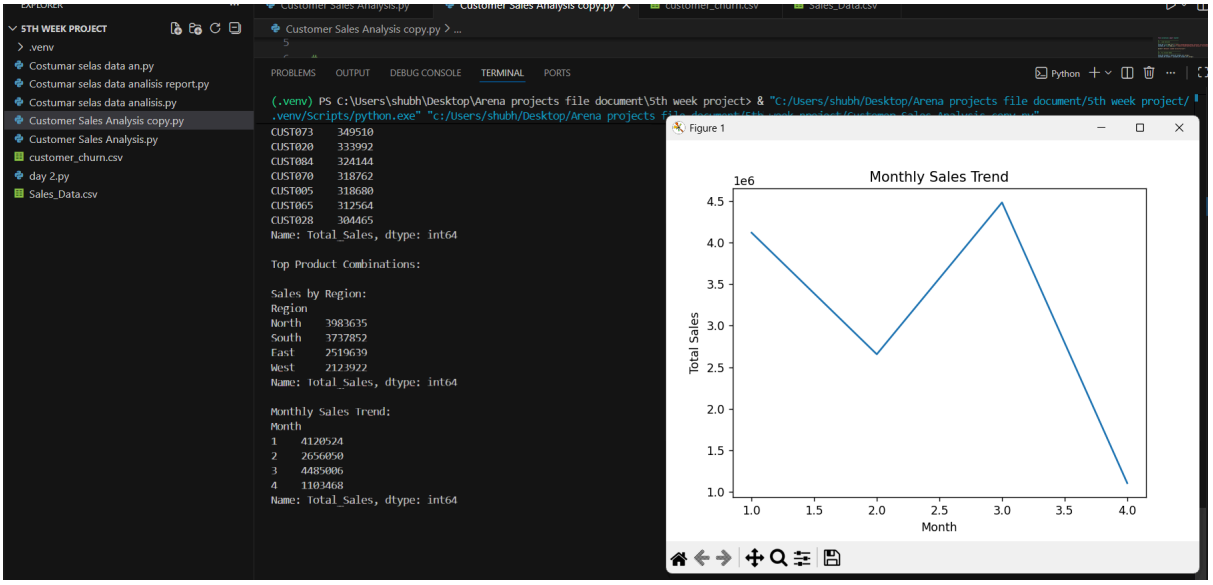


Image3

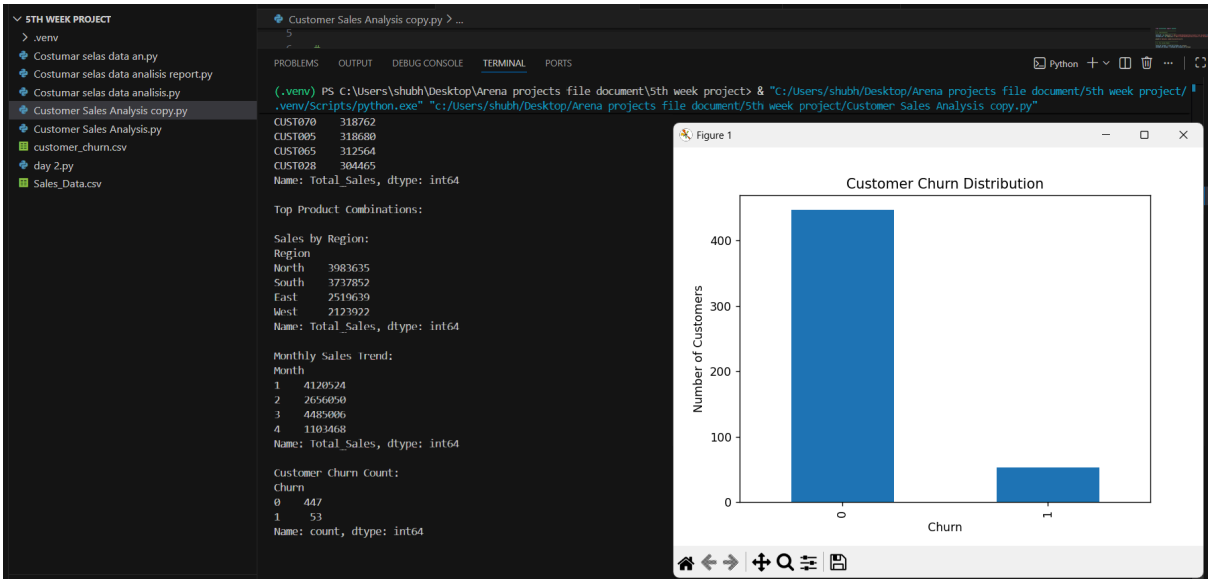
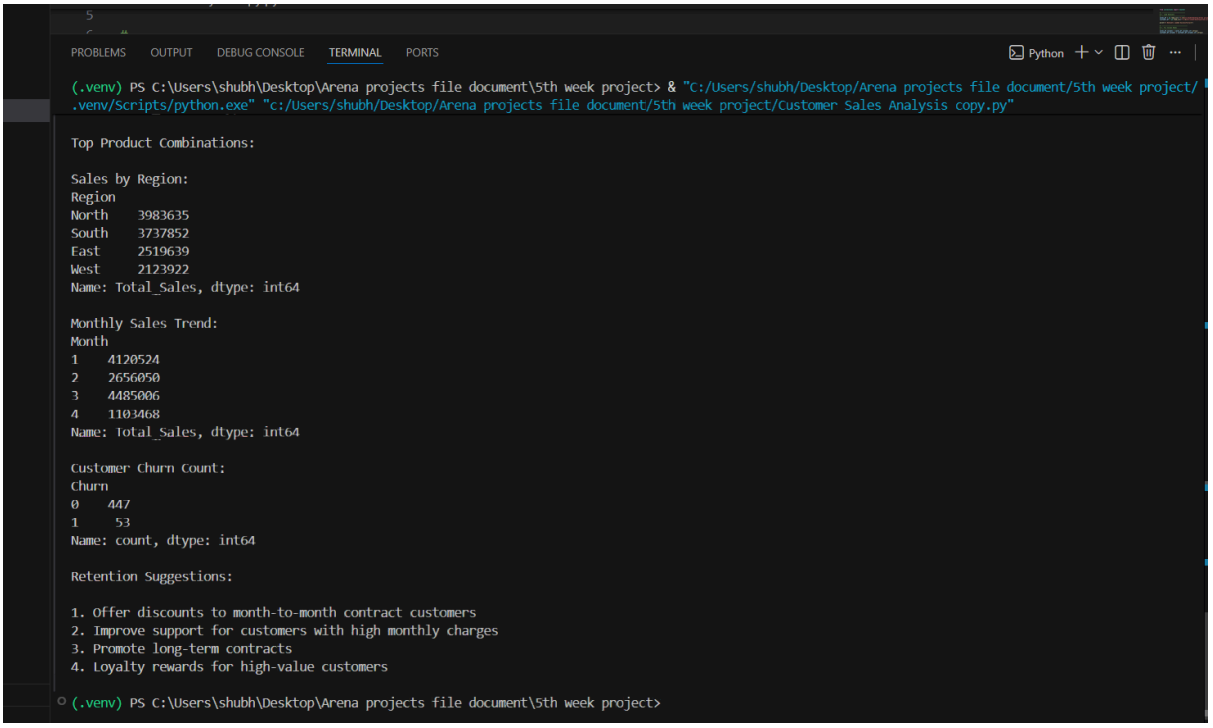


Image4



Business Insights & Recommendations

✅ Insights

- Focus marketing on high-value customers
- Increase inventory for best-selling products
- Strengthen operations in low-performing regions

✅ Recommendations

- Introduce loyalty programs
- Offer seasonal discounts during low-sales months
- Use cross-selling strategies

9️⃣ Project Summary

This project demonstrates how Python and pandas can be used for real-world business data analysis. Through data cleaning, aggregation, merging, and visualization, meaningful insights were extracted to support business decisions. The project follows a structured data analytics workflow and is suitable for Data Analyst and Business Analyst roles.

🔟 Conclusion

The Customer Sales Analysis project successfully transformed raw sales data into actionable insights. It highlights strong analytical skills, data handling expertise, and the ability to communicate results clearly through visualizations and reports.