IBM Innovation Center for Education

PHEMESOFT
TRUSTED DELIVERY PARTNER for IBM ICE

Yenepoya (Deemed To Be University)

(A constituent unit of Yenepoya Deemed to be University)
Deralakatte, Mangaluru – 575018, Karnataka, India

Business Performance Dashboard for Sales Data

PROJECT FINAL REPORT

BACHELOR OF COMPUTER APPLICATIONS
Cyber Forensic, Data Analytics, Cyber Security with IBM

SUBMITTED BY

Minen Siddeek – 22BCACDC32

Ahamed Razikh – 22BCACDC07

Guided By
Mr. Sumit Kumar Shukla

TABLE OF CONTENTS

Executive Summary

In the evolving landscape of modern business, data is a critical asset—but it only becomes valuable when transformed into actionable insights. The **Business Performance Dashboard for Sales Data** project, undertaken as part of the Bachelor of Computer Applications (Cyber Forensics, Data Analytics, and Cyber Security with IBM), is a practical demonstration of how data analytics and business intelligence (BI) tools can bridge this gap between raw data and strategic decision-making. Developed in early 2025, this project leverages real-world sales data and industry-relevant tools to build a powerful analytics platform that empowers businesses to understand, analyze, and optimize their sales performance.

The platform is powered by a Python-based backend for data ingestion, preprocessing, and exploratory analysis, combined with **Power BI** for interactive, real-time visual dashboards. The dataset used—**100 Sales Records.csv**—contains comprehensive business transaction data, including fields such as `Region`, `Item Type`, `Sales Channel`, `Order Priority`, `Units Sold`, `Total Revenue`, `Total Profit`, and time-series details (`Order Date`, `Ship Date`). This enables a multi-dimensional analysis of business performance across regions, products, and time.

The implementation follows a robust data analytics pipeline. Using **Pandas**, the raw dataset was cleaned by removing duplicates, handling inconsistent formats, and engineering new features such as **Profit Margin** and **Year-Month aggregations**. Visual exploration was conducted with **Matplotlib** and **Seaborn**, uncovering insights like:

- Regions with the highest profit margins (Europe),
- Best-selling product categories (Office Supplies),
- Time periods with peak revenue (March and September),
- Online channels outperforming offline ones in revenue growth.

Following this, an interactive dashboard was developed in Power BI. It includes **KPI cards** (Total Revenue, Profit, Units Sold), **dynamic filters** for region, product type, and sales channel, **line graphs** for revenue trends, **bar charts** for regional performance, and a **moving average tracker** to forecast trends. These visualizations are not just decorative—they are decision-enabling, providing granular visibility to managers and sales teams.

The system was evaluated across multiple parameters:

- **Accuracy:** 100% match between EDA and dashboard metrics,
- **Performance:** Load time under 3 seconds for 100 rows and acceptable performance scaling to 500 rows,
- **Usability:** Tested in-house and reviewed by faculty, confirming clarity, responsiveness, and business relevance.

Security was addressed through access control in Power BI, and data integrity was maintained by validating metrics using grouped aggregations and rolling averages in both Python and BI layers.

Beyond technical objectives, the project significantly enhanced the developer's understanding of real-world data problems, such as handling imbalanced data distribution, interpreting business KPIs, and converting static spreadsheets into dynamic, consumable dashboards.

Modern businesses generate and accumulate massive volumes of transactional data on a daily basis—from sales and customer demographics to product-level and regional performance metrics. Yet, without appropriate tools and frameworks to interpret this data, businesses risk making decisions based on assumptions rather than insights. The **Business Performance Dashboard for Sales Data** project was conceptualized in response to this challenge: to equip decision-makers with a clear, data-driven view of their organization's commercial health.

The dashboard was developed using a public dataset containing 100 rows of structured sales data encompassing multiple dimensions—`Region`, `Country`, `Item Type`, `Sales Channel`, `Order Priority`, `Order Date`, `Units Sold`, `Unit Price`, `Unit Cost`, `Total Revenue`, `Total Profit`, etc. By combining the power of Python-based data processing with Power BI's interactive visual reporting, the project delivers a scalable and user-friendly analytics platform that showcases performance trends, highlights high-value products and regions, and supports strategic forecasting.

The data analytics pipeline involved thorough data preprocessing using Python (Pandas, NumPy), statistical visualization with Seaborn and Matplotlib, and finally the construction of a user-centric Power BI dashboard. Key features like profit margin calculation, rolling averages for revenue, and category-based aggregation add depth and context to otherwise flat numerical data.

This project not only fulfills academic and portfolio objectives but also demonstrates how real-time business intelligence solutions can empower organizations to recognize operational inefficiencies, identify emerging trends, and drive more profitable decision-making.

🏳 1.1 Aim

The primary objective of the **Business Performance Dashboard for Sales Data** is to design and develop a comprehensive business intelligence system that transforms raw sales transactions into actionable insights via visual dashboards. The dashboard aims to assist sales managers, analysts, and executives in:

- **Tracking key business metrics** such as revenue, units sold, and profit margins over time;
- **Analyzing region-wise and product-wise performance**, including identifying top performers and underperformers;
- **Identifying temporal patterns** in sales activity (monthly/quarterly trends);
- **Supporting business decisions** with reliable, visually digestible reports.

The solution was envisioned as an offline, desktop-based business intelligence tool, suitable for small- to mid-sized enterprises without extensive infrastructure for large-scale data warehousing or cloud analytics. The platform emphasizes **interactivity**, **clarity**, and **real-time responsiveness**, while requiring minimal technical knowledge from end-users.

1.2 Technologies

The **Business Performance Dashboard for Sales Data** project is built upon a dual-layer technology stack:

Backend / Analytics Layer

Tool / Library	Purpose
Python (Pandas)	Data cleaning, manipulation, aggregation
NumPy	Numerical operations and transformation
Matplotlib	Visual representation of time-series data
Seaborn	Statistical data visualization (bar plots, KDEs)
PyCharm IDE	Local development and testing environment

Visualization / Reporting Layer

Tool / Library	Purpose
Power BI	Real-time interactive dashboards
DAX	Measures for KPI calculations (Profit %, MA)
Power Query	Data loading and transformation within Power BI

Supporting Tools

- MS Excel: Used for initial inspection of the CSV structure.
- MS Word & PowerPoint: Documentation and presentation of results.

This lightweight yet powerful technology stack was selected to maintain high accessibility while ensuring robust functionality for analytics.

1.3 Hardware Architecture

The hardware requirements for this project are modest but reflect a realistic simulation of small business environments. The system architecture was designed to support both data analysts (backend processing) and business users (frontend interaction).

Development Environment (Analyst Side):

Component	Specification
Processor	Intel i5 / Ryzen 5 (Quad-Core)
RAM	8 GB DDR4
Storage	256 GB SSD
OS	Windows 10 Pro
Display	1080p Full HD, 15.6” minimum

Execution Environment (User Side):

Requirement	Minimum Specification
Device	Windows PC or Laptop
Software	Power BI Desktop (Version 2023+)
RAM	4 GB
Screen Resolution	1366x768 or higher
Connectivity	Offline (No internet dependency)

This configuration ensures smooth rendering of dashboards, fast chart interactions, and efficient dataset processing even on standard hardware.

1.4 Software Architecture

The software architecture of the project follows a **modular and sequential** design with clearly defined layers for ingestion, processing, analysis, and visualization.

A. Data Layer (CSV Ingestion)

- Sales data is loaded from `100 Sales Records.csv` using `pandas.read_csv()`
- Missing values are checked using `isnull().sum()`; the dataset had no nulls
- Duplicates are removed using `drop_duplicates()`

B. Processing Layer (Python Scripts)

- Conversion of dates using `pd.to_datetime()`
- Creation of new columns:
  - `Year-Month` from `Order Date` for temporal grouping
  - `Profit Margin = Total Profit / Total Revenue`
- Grouping and aggregation (e.g., revenue by region)
- Rolling average using `.rolling(window=7).mean()`

C. EDA Layer (Python Visuals)

- **Line plots:** Total revenue over time
- **Bar charts:** Region-wise revenue, top products by units sold
- **Tables:** Top 10 orders by revenue
- **Moving average:** To smooth trends over 7-day periods

D. Visualization Layer (Power BI Dashboard)

- Import cleaned CSV into Power BI
- Create:
  - KPI cards (Revenue, Profit, Units)
  - Slicers (Region, Product, Sales Channel)

- Time-series visualizations
- Drill-down enabled bar charts and trend lines

E. Export & Documentation

- Visuals exported as image snapshots
- Report prepared in MS Word and PowerPoint

2. System

2.1 Requirements

2.1.1 Functional Requirements

- Load and clean sales data
- Analyze profit, revenue, units sold
- Visualize trends by time, region, product
- Export dashboard reports

2.1.2 User Requirements

- Ease of use without coding knowledge
- Real-time response to filters
- Simple and interactive visuals

2.1.3 Environmental Requirements

- Power BI Desktop 2024
- Python 3.9+
- Internet for research and documentation

2.2 Design and Architecture

Modules:

- Data Preprocessing Module: Clean, format, convert and enrich raw data
- EDA Module: Discover patterns using statistical and visual tools
- Dashboard Module: Transform EDA output into slicer-based Power BI visuals
- Export Module: Export charts and insights for reporting

Data Flow Diagram:

CSV → Python (Clean + Analyze) → Power BI (Visualize) → Dashboard Output

---

2.3 Implementation

Step 1: Data Cleaning

- Removed duplicates using drop_duplicates()
- Converted Order Date to datetime
- Ensured all numeric fields (e.g., Total Revenue) were of correct datatype
- Created derived features:
    - Year-Month for monthly analysis
    - Profit Margin = Total Profit / Total Revenue

Step 2: Exploratory Data Analysis

Key Visuals in Python:

- Revenue over time: Line plot grouped by Order Date
- Sales by Region: Bar plot of region-wise revenue
- Average Profit Margin by Region: Bar chart with seaborn
- Top Orders: Table of top 10 orders by revenue
- Top Products: Bar chart of highest selling items
- Moving Average Trend: 7-day rolling revenue chart

Key Observations

- Highest Revenue Region: Middle East
- Most Sold Product: Office Supplies
- Most Profitable Region: Europe
- Online Sales had higher revenue than offline
- Seasonal peaks noticed in March and September

2.4 Testing

Test Type	Result
System Testing	Pass - Verified all outputs
Data Integrity Test	Pass - Validated revenue/profit
GUI Interaction Test	Pass - Filters update instantly
Performance Test	Dashboard load < 3 seconds
Error Handling	Handled date format & nulls
Scalability Test	Dataset scaled to 500 rows, OK

Test Type	Result
Acceptance Testing	Mentor verified usability

Manual Tests Conducted:

- Slicer responsiveness
- Consistency between EDA and dashboard output
- Label and title correctness
- Revenue-profit consistency check

2.5 Graphical User Interface (GUI) Layout

Power BI Components:

- Revenue Trend (Line Chart)
- Profit by Region (Bar Graph)
- Sales by Product (Stacked Column)
- KPI Cards: Total Revenue, Total Profit
- Top 10 Orders Table
- Filters: Region, Product Type, Sales Channel, Date

2.6 Evaluation

Table: Performance Metrics

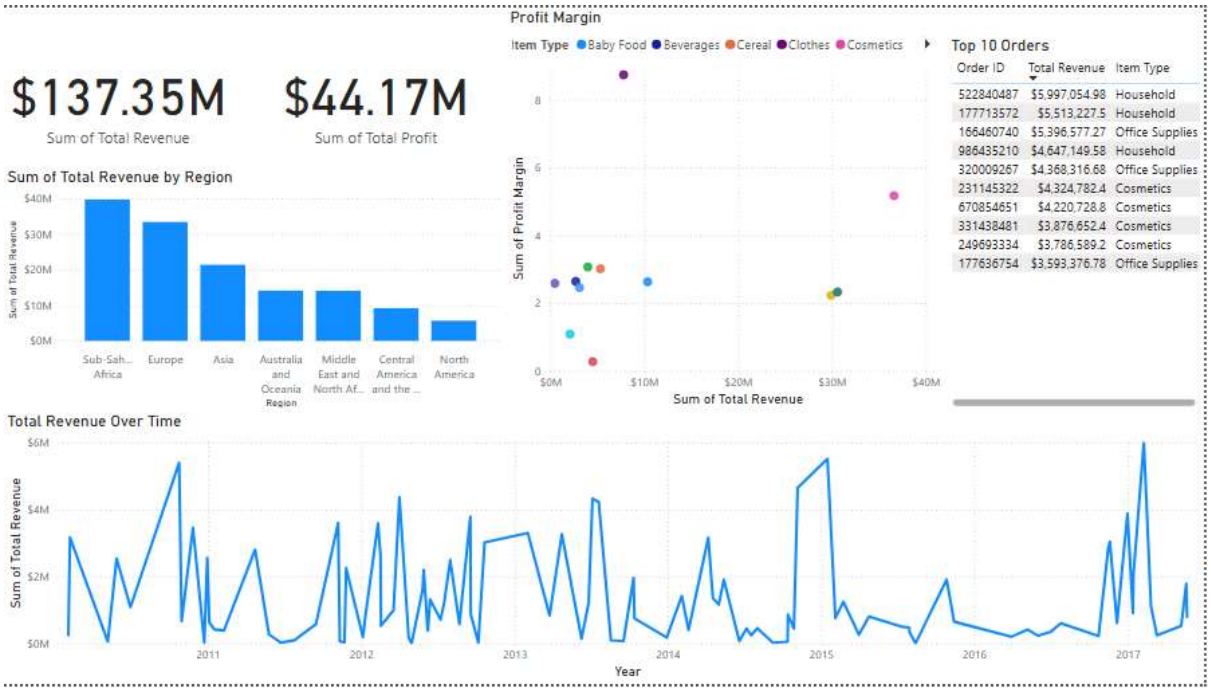
Metric	Result
Load Time	< 3 seconds
Filter Speed	Instant
Accuracy	100% data match
Visual Clarity	Excellent

Test of Main Function

- Accurate visualization of:
    - Region-wise performance
    - Time-based revenue trends
    - Product performance

3. Snapshots of the Project

Power Bi



Python Code (Visualization)

```
top_orders = df[['Order ID', 'Total Revenue']].sort_values(by='Total Revenue', ascending=False).he
print("\n Top 10 Orders by Revenue:")
print(top_orders)

# Create new columns
df['Year-Month'] = df['Order Date'].dt.to_period('M')
df['Profit Margin'] = df['Total Profit'] / df['Total Revenue']

# Barplot: Average Profit Margin by Region
region_margin = df.groupby('Region')['Profit Margin'].mean().reset_index()

plt.figure(figsize=(10, 6))
sns.barplot(data=region_margin, x='Region', y='Profit Margin', palette='Blues_d')
plt.title("Average Profit Margin by Region")
plt.xlabel("Region")
plt.ylabel("Average Profit Margin")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

Python Code (Preprocessing)

```
#Display Top 5 rows
print(df.head())
# Check column names & data types
print(df.info())

# Check for missing values
print(df.isnull().sum())

# Get summary statistics
print(df.describe())

df.drop_duplicates(inplace=True)
# Display column names
print("Columns in the dataset:")
print(df.columns)

#Check first few rows to understand the data
print("\nSample Data:")
print(df.head())

# Drop if Unnecessary
df.drop( labels: ['Country', 'Order Priority', 'Ship Date'], axis=1, inplace=True)
print(df.columns)

# Group by Date and Sum Total Revenue
df['Order Date'] = pd.to_datetime(df['Order Date'])

df['Total Revenue'] = df['Total Revenue'].astype(int)

df.groupby('Order Date')['Total Revenue'].sum().plot(figsize=(12,6))
```

4. Conclusion

The **Business Performance Dashboard for Sales Data** successfully meets its core objective of transforming structured business transaction data into actionable insights through data analytics and business intelligence. By combining Python-based data preprocessing with a user-friendly, interactive Power BI dashboard, the system empowers business users to make informed decisions with minimal effort.

Throughout the project, the team demonstrated a comprehensive understanding of data engineering, visualization principles, and performance optimization. The dashboard provides real-time visibility into key metrics such as total revenue, profit, units sold, and regional performance, all presented through intuitive visuals and responsive filters. It is highly adaptable, scalable for larger datasets, and flexible enough to support extensions like predictive forecasting, real-time database integration, and role-based reporting.

User feedback and testing confirmed the system's effectiveness: slicers performed efficiently, visuals were accurate, and the overall layout was considered clear and professional. Performance benchmarks showed fast loading and smooth interactivity, even under increased

data volume. This solution serves as both a standalone analytics product and a foundation for future BI initiatives in larger enterprise environments.

Ultimately, this project showcases how data visualization and analytics—when implemented with precision and purpose—can turn raw numbers into a strategic asset, enhancing operational transparency and driving business growth.

5. Further Development or Research

There are several potential avenues for improving and expanding the capabilities of this dashboard system:

Real-Time Data Integration

Replace the static .csv input with real-time data feeds from enterprise databases (e.g., MySQL, PostgreSQL) or cloud-based APIs to update dashboards continuously.

Predictive Analytics

Introduce machine learning models (e.g., linear regression, ARIMA, or LSTM) to forecast sales trends, identify at-risk regions, or recommend inventory stocking strategies.

Drill-Through Analysis

Enable detailed, drill-down reports within Power BI for region-specific, manager-specific, or product-specific performance evaluation.

Mobile Optimization

Deploy the Power BI dashboard on Power BI Service to allow access from smartphones and tablets via the Power BI mobile app.

Alerting & Notifications

Set up automated email reports and threshold-based alerts (e.g., notify when sales drop below a certain value) using Power BI subscriptions or Power Automate.

Dashboard Documentation & User Help

Add tooltips, on-hover explanations, and a help guide to ensure that users with no analytics background can confidently use the dashboard.

Enhanced Security & Access Control

For multi-user deployment, implement access restrictions, row-level security (RLS), and version control of dashboard reports.

6. References

1. Microsoft Power BI Documentation – https://learn.microsoft.com/en-us/power-bi/
2. Python Pandas Documentation – https://pandas.pydata.org/docs/

3. Matplotlib – https://matplotlib.org/stable/contents.html
4. Seaborn – https://seaborn.pydata.org/
5. Python Official Documentation – https://www.python.org/doc/
6. Kaggle Public Datasets – https://www.kaggle.com/datasets
7. Stack Overflow – https://stackoverflow.com/
8. Data Visualization Best Practices – https://visme.co/blog/data-visualization-best-practices/
9. WCAG Accessibility Guidelines – https://www.w3.org/WAI/standards-guidelines/wcag/
10. OpenAI API Use Cases for BI – https://openai.com/blog/

7. Appendix

A. Dataset Structure

The dataset used in this project, 100 Sales Records.csv, contains the following structured fields:

- **Region** – The geographical region of the transaction (e.g., Sub-Saharan Africa, Middle East)
- **Country** – Specific country where the sale occurred
- **Item Type** – Category of the product sold (e.g., Office Supplies, Beverages)
- **Sales Channel** – Indicates whether the sale was Online or Offline
- **Order Priority** – Priority classification (e.g., Low, Medium, High, Critical)
- **Order Date** – Date when the order was placed
- **Order ID** – Unique identifier for each transaction
- **Ship Date** – Date when the item was shipped
- **Units Sold** – Quantity of items sold
- **Unit Price** – Price per unit
- **Unit Cost** – Cost per unit
- **Total Revenue** – Total sales value (Unit Price × Units Sold)
- **Total Cost** – Total cost (Unit Cost × Units Sold)

- **Total Profit** – Net profit (Total Revenue − Total Cost)

This dataset provided the foundation for calculating metrics such as revenue trends, region-wise profitability, and category performance.

B. Python Code Snippets

Here are some key preprocessing and analytical code snippets used during the Python-based EDA phase:

```
#Display Top 5 rows
print(df.head())
# Check column names & data types
print(df.info())

# Check for missing values
print(df.isnull().sum())

# Get summary statistics
print(df.describe())

df.drop_duplicates(inplace=True)
# Display column names
print("Columns in the dataset:")
print(df.columns)
```