



**YENEPOYA**

(DEEMED TO BE UNIVERSITY)

Recognized under Sec 3(A) of the UGC Act 1956

Accredited by NAAC with 'A' Grade

**YENEPOYA INSTITUTE OF ARTS, SCIENCE, COMMERCE AND MANAGEMENT  
YENEPOYA (DEEMED TO BE UNIVERSITY)  
BALMATTA, MANGALORE**

**AN PROJECT REPORT ON  
“SALES PERFORMANCE ANALYSIS”**

**SUBMITTED BY  
FATHIMATHUL HIBA ABDUL JABBAR**

**III BCA (DATA ANALYTICS, CLOUD COMPUTING AND CYBER SECURITY WITH  
IBM)  
22BDACC084**

**UNDER THE GUIDANCE OF  
MS. BHOOMIKA SUVARNA  
LECTURER  
DEPARTMENT OF COMPUTER SCIENCE**

**IN PARTIAL FULLFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE  
DEGREE OF  
BACHELORS IN COMPUTER APPLICATION**

**APRIL 2025**

## **CERTIFICATE**

This is to certify that the project work entitled “**Sales Performance Analysis**” has been Successfully carried out in the Graduate Studies and Research in Computer Science by **Fathimathul Hiba Abdul Jabbar** (Reg No:**22BDACC084**), student of III BCA (**Data Analytics, Cloud Computing and Cyber Security with IBM**), under the supervision and guidance of **Ms. Bhoomika Suvarna**

Internal Guide: **Ms. Bhoomika**

Chairperson:

Internal Examiner:

External Examiner:

PRINCIPAL

**Prof (Dr.) Arun A. Bhagawath**

The Yenepoya Institute of Arts, Science, Commerce and Management  
(Deemed to be University)

Submitted for the viva- voice

Place: Mangalore



## DECLARATION

I **Fathimathul Hiba Abdul Jabbar** bearing **Reg.22BDACC084** hereby declare that this project report entitled “**Sales Performance Analysis**” had been prepared by me towards the partial fulfilment of the requirement for the award of the Bachelor of Computer Application at Yenepoya (Deemed to be University) under the guidance of **Ms. Bhoomika Suvarna**, Department of Computer Science, Yenepoya Institute of Arts, Science, Commerce and Management.

I also declare that this field study report is the result of my own effort and that it has not been submitted to any university for the award of any degree or diploma.

**Place: Mangalore**

**Date:**

**Fathimathul Hiba Abdul Jabbar**

**III BCA (BDACC)**

**22BDACC084**

## **ACKNOWLEDGEMENT**

I would like to express my deepest gratitude to Principal and dean of science **Prof. Dr. Arun A. Bhagawath** and to Vice Principal **Dr. Shareena P, Dr. Jeevan Raj** and **Mr. Narayana Sukumar** for their kind permission and giving me an opportunity to do this study.

Special thanks are extended to **HOD Dr. Rathnakar Shetty P**, Department of Computer Science for his invaluable insights and encouragement.

I am profoundly thankful to my internal guide **Ms. Bhoomika**, for her continuous support, expertise, patience and mentorship which greatly contributed to the completion of my project, “**Sales Performance Analysis**”.

I would also like to extend my appreciation to all other faculty members and staff who have provided assistance and support in various capacities during the course of this project. Your contributions have been instrumental in shaping the outcome of this endeavor.

**Place: Mangalore**

**Date:**

**Fathimathul Hiba Abdul Jabbar**

**III BCA (BDACC)**

**22BDACC08**

# TABLE OF CONTENTS

<b>1. INTRODUCTION</b>	<b>9</b>
1.1 OVERVIEW OF THE PROJECT	9
1.2 OBJECTIVE OF THE PROJECT	9
1.3 PROJECT CATEGORY	9
1.4 TOOLS AND PLATFORM TO BE USED	9
1.5 OVERVIEW OF THE TECHNOLOGIES USED	10
1.5.1 Hardware Requirements	10
1.5.2 Software Requirements	10
1.6 STRUCTURE OF THE PROGRAM	10
1.7 STATEMENT OF THE PROBLEM	11
<b>2. LITERATURE REVIEW</b>	<b>12</b>
<b>3. SOFTWARE REQUIREMENTS SPECIFICATION</b>	<b>14</b>
3.1 INTRODUCTION	14
3.1.1 Purpose	14
3.1.2 Scope of the project	14
3.1.3 Intended audience and Reading Suggestions	14
3.1.4 Definitions, Acronyms and Abbreviations	14
3.1.5 References	16
3.1.6 Overview	16
3.2 OVERALL DESCRIPTION	16

3.2.1	Product Perspective	16
3.2.2	Product Features	17
3.2.3	User Characteristics	17
3.2.4	Operating Environment	17
3.2.5	Design and Implementation Constraints	17
3.2.6	General Constraints	18
3.2.7	Assumptions and Dependencies	18
3.3	SPECIFIC REQUIREMENTS	18
3.3.1	External Interface Requirements	18
3.3.1.1	User Interface	18
3.3.1.2	Hardware Interface	19
3.3.1.3	Software Interface	19
3.3.2	Functional Requirements	19
3.3.3	Performance Requirements	19
3.3.4	Design Constraints	19
3.3.5	Other Requirements	20
<b>4.</b>	<b>SYSTEM ANALYSIS AND DESIGN</b>	<b>21</b>
4.1	INTRODUCTION	21
4.2	METHODOLOGY	21
4.3	DATA FLOW DIAGRAM	22
4.4	TABLE RELATIONSHIP	23
4.5	TABLE DESCRIPTION	23

4.6 SYSTEM DESIGN IMPLEMENTATION	24
4.7 USER INTERFACE DESIGN	25
<b>5. TESTING</b>	<b>28</b>
5.1 INTRODUCTION	28
5.2 TESTING OBJECTIVE	28
5.3 TEST CASES	28
5.3.1 Login functionality	28
5.3.2 Dashboard Rendering	29
5.3.3 Data Accuracy	29
<b>6. SYSTEM SECURITY</b>	<b>30</b>
6.1 INTRODUCTION	30
6.2 SOFTWARE SECURITY	30
<b>7. CONCLUSION</b>	<b>32</b>
<b>8. FUTURE ENHANCEMENTS</b>	<b>33</b>
<b>9. BIBLIOGRAPHY</b>	<b>34</b>
<b>10. APPENDIX</b>	<b>35</b>

## LIST OF IMAGES

Image no	Particular	Page no
1	Table Relationships	
2	Login Page	
3	Home Page	
4	Profit Analysis Page	
5	Performance Insights Page	



# **1.INTRODUCTION**

## **1.1. OVERVIEW OF THE PROJECT**

In today's competitive business environment, real-time data analysis has become essential. This project, Sales Performance Analysis, focuses on resolving the inefficiencies faced by a hardware company in interpreting sales data. Previously dependent on static Excel reports, the organization lacked actionable insights and real-time analytics. This project introduces an interactive Sales Insights Dashboard using Power BI, enabling data-driven decision-making by visualizing key sales metrics and trends.

## **1.2. OBJECTIVE OF THE PROJECT**

The primary objective of this project is to enhance sales decision-making using Business Intelligence (BI) tools.

Specific objectives include:

- Extracting and cleaning sales data from a MySQL database.
- Developing interactive dashboards using Power BI.
- Calculating metrics like revenue, profit margin, and sales quantity.
- Providing actionable insights to stakeholders.
- Integrating the dashboard into a Django-based web portal.

## **1.3. PROJECT CATEGORY**

This project falls under the Data Analytics and Business Intelligence category, with practical implementation in Web Development using Django and Power BI integration.

## **1.4. TOOLS AND PLATFORM TO BE USED**

BI Tool: Microsoft Power BI

Database: MySQL, WAMP Server

ETL Tool: Power Query (within Power BI)

Programming: Python

IDE: Visual Studio Code

Web Framework: Django

## **1.5. OVERVIEW OF TECHNOLOGIES USED**

This project integrates various technologies across data collection, analysis, and visualization:

### **1.5.1 HARDWARE REQUIREMENTS**

Intel Core i5 processor (or equivalent)

Minimum 8GB RAM (16GB recommended)

256GB SSD storage

Full HD display (1920×1080 resolution)

### **1.5.2 SOFTWARE REQUIREMENTS**

Power BI Desktop (v2.120.664 or higher)

MySQL and WAMP Server

Microsoft Excel

Python (for data preprocessing)

VS Code

Django framework (for deployment)

## **1.6. STRUCTURE OF THE PROGRAM**

The program follows a structured data pipeline:

Data Extraction: Sales data retrieved from MySQL.

Data Transformation: Performed using Power Query.

Data Modeling: Implemented a star schema with sales transactions as the fact table.

DAX Measures: Created metrics like profit margin %, YoY growth.

Visualization: Built multiple dashboard pages (Home, Profit Analysis, Performance Insights).

Web Integration: Embedded Power BI dashboard in a Django web portal for role-based access.

## **1.7. STATEMENT OF THE PROBLEM**

The company faced multiple issues:

Manual Reports: Dependency on Excel sheets for reporting led to inefficiency.

Lack of Real-Time Insights: Delayed data caused missed opportunities.

Subjective Regional Inputs: Reports varied in consistency and detail.

Sales Decline: Without analytical backing, performance issues went unnoticed.

Solution: A Power BI-based dashboard was created to automate reporting, visualize sales KPIs, enable data filters (by region/time/product), and provide real-time access via a Django-based web interface

## **2. LITERATURE REVIEW**

The evolution of business intelligence (BI) tools has transformed how companies analyze data and make decisions. The use of dashboards for sales analysis has become a standard in the modern business world. This literature review explores existing studies, articles, and practices related to sales analytics, Power BI, and data visualization techniques, which form the foundation of this project.

According to Turban et al. (2011), Business Intelligence (BI) refers to the tools, technologies, and practices used to collect, integrate, analyze, and present business data. BI plays a critical role in helping companies understand their operational performance, particularly in areas like sales, finance, and customer relationship management.

Sales analysis helps organizations monitor product performance, regional effectiveness, seasonal trends, and customer behaviors. A lack of proper visualization and insights can lead to poor decision-making, especially in large companies with distributed sales data across regions.

Power BI, developed by Microsoft, is a leading BI platform known for its flexibility, user-friendly interface, and advanced data modeling capabilities. It supports: Integration with various databases like MySQL, SQL Server, Excel, and Azure. Creation of interactive dashboards and real-time visualizations.

Use of DAX (Data Analysis Expressions) for advanced data calculations.

Multiple studies have highlighted Power BI's effectiveness in improving data interpretation and decision-making processes. For example, a case study by Singh & Sharma (2019) showed that implementing Power BI dashboards improved sales tracking efficiency by 60% in an Indian retail company.

Before analysis, data must be cleaned and transformed — a process known as ETL (Extract, Transform, Load). Kimball & Ross (2013) emphasized that over 60% of a data analyst's time is spent in this phase, reinforcing its importance.

In this project, Power Query in Power BI was used for data cleaning, handling null values, correcting currency mismatches, and structuring the dataset. These practices are consistent with data warehousing standards in the industry.

Dashboards that display key performance indicators (KPIs)—like revenue, profit margins, and sales volume—help decision-makers act quickly. According to Few (2006), dashboards should be concise, highly visual, and focused on the most relevant information.

A study by SAP (2020) on global organizations using BI tools highlighted that sales dashboards led to 35% faster decision-making. 40% improvement in identifying underperforming regions/products. Increased collaboration between sales and strategy teams.

Integration of BI dashboards with platforms like Django allows users to access dashboards via web portals. This concept aligns with modern software architecture trends, especially in enterprise environments where remote access and role-based views are necessary.

This review shows that using Power BI for sales insight is not only an industry standard but also an academically validated method for improving sales performance and decision-making. It also establishes the credibility of learning resources like Code basics and reinforces the significance of a structured methodology (like ETL and KPI dashboards) in business analytics projects.

## **3. SOFTWARE REQUIREMENTS SPECIFICATIONS**

### **3.1. INTRODUCTION**

#### **3.1.1. Purpose**

The purpose of this document is to define the software requirements for the Sales Performance Analysis system. It serves as a foundation for system design, development, and testing. The project aims to improve sales decision-making through interactive dashboards developed using Power BI, with data sourced from MySQL and integrated via Django.

#### **3.1.2 Scope of the Project**

The project focuses on the creation of a Sales Insights Dashboard for a hardware company that traditionally used Excel reports for sales tracking. This tool aims to automate data collection, perform advanced analytics, and visualize key sales performance metrics like revenue, profit margin, and sales trends. The solution also includes web-based access via Django, supporting real-time and role-based reporting.

#### **3.1.3 Intended Audience and Reading Suggestions**

This document is intended for:

- Project supervisors and evaluators
- Developers and data analysts
- Stakeholders seeking insights into project goals and functionalities
- Readers are advised to have a basic understanding of business intelligence tools, databases, and web technologies.

#### **3.1.4 Definitions, Acronyms, and Abbreviations**

The following terms and abbreviations are used throughout this document:

BI (Business Intelligence): A set of tools, technologies, and processes that convert raw data into meaningful and actionable insights to support business decision-making. Power BI is used

in this project for BI visualization.

**KPI (Key Performance Indicator):** A measurable value that demonstrates how effectively an organization is achieving key business objectives. Examples in this project include revenue, profit margin, and customer performance metrics.

**ETL (Extract, Transform, Load):** A data integration process that involves extracting data from a source, transforming it into a usable format, and loading it into a destination system for analysis.

**DAX (Data Analysis Expressions):** A formula language used in Power BI for creating custom calculations and aggregated metrics such as profit margin percentage and year-over-year growth.

**SQL (Structured Query Language):** A standard language used to manage and manipulate relational databases. MySQL is the chosen database system for this project.

**UI (User Interface):** The front-end interface through which users interact with the system. In this project, it refers to the embedded dashboard hosted on a Django web application.

**Power BI:** A Microsoft business analytics tool that enables the creation of interactive dashboards and reports from various data sources, supporting real-time decision-making.

**Dashboard:** A graphical interface that displays essential business metrics and key performance indicators in a concise and interactive format.

**MySQL:** An open-source relational database management system used for storing, retrieving, and managing structured sales-related data in this project.

**Star Schema:** A type of database schema used in data warehousing, where a central fact table is linked to multiple dimension tables, enhancing performance and scalability.

**Django:** A high-level Python web framework that enables secure and scalable web development. It is used in this project to host the Power BI dashboard through embedding.

**RLS (Row-Level Security):** A Power BI feature that allows the implementation of user-specific views by restricting access to rows in a dataset based on user roles.

### **3.1.5 References**

Turban, E., Sharda, R., & Delen, D. (2011). *Decision Support and Business Intelligence Systems* (9th ed.). Pearson Education.

Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed.). Wiley.

Few, S. (2006). *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly Media.

Russo, M., & Ferrari, A. (2021). *The Definitive Guide to DAX: Business Intelligence for Microsoft Power BI, SQL Server Analysis Services, and Excel* (2nd ed.). Microsoft Press.

Microsoft Corporation. (2023). *Power BI Documentation*. Retrieved from <https://learn.microsoft.com/en-us/power-bi/>

Gartner Inc. (2022). *Magic Quadrant for Analytics and Business Intelligence Platforms*. Retrieved from <https://www.gartner.com>

SAP. (2020). *The Business Value of BI and Analytics*. SAP Insights. Retrieved from <https://insights.sap.com>

Singh, R., & Sharma, S. (2019). "Enhancing Retail Sales Insights Using Power BI: A Case Study." *International Journal of Data Science*, 4(2), 45–52.

CodeBasics. (2024). *Power BI Full Course – YouTube Channel*. Retrieved from <https://www.youtube.com/@codebasics>

### **3.1.6 Overview**

The document is structured to first provide a high-level description of the system, followed by specific requirements including functional, performance, and interface details. It concludes with design and deployment considerations.

## **3.2 OVERALL DESCRIPTION**

### **3.2.1 Product Perspective**



The Sales Insights Dashboard is a standalone BI system that interfaces with MySQL for backend data storage and Power BI for visualization. It follows a modular architecture with a frontend embedded into a Django web portal for user accessibility.

### **3.2.2 Product Features**

- Real-time data connectivity with MySQL
- KPI calculations using DAX
- Interactive dashboards with filters for region, time, and product
- Visual analytics: bar charts, line graphs, maps, slicers
- Embedded dashboard in Django with user login

### **3.2.3 User Characteristics**

Intended users include:

Sales Managers: For trend identification and decision-making

Executives: For profitability and growth insights

Analysts: For deeper metric analysis and reporting

Users are expected to have basic familiarity with BI tools and dashboard navigation.

### **3.2.4 Operating Environment**

Hardware: Intel Core i5, 8–16GB RAM, 256GB SSD, Full HD Display

Software: Windows 10/11, Power BI Desktop, MySQL, Python, Django, VS Code

### **3.2.5 Design and Implementation Constraints**

Power BI limitations in free tier (e.g., collaboration and sharing)

Real-time integration limited by MySQL refresh intervals

Embedded iframe view subject to Power BI Service permissions

### **3.2.6 General Constraints**

Internet required for dashboard publishing and embedding

Data privacy must be maintained; no sensitive user/customer data included

Must operate efficiently on systems with minimum hardware specs

### **3.2.7 Assumptions and Dependencies**

Assumptions:

- Users will have a stable internet connection to access the dashboard.
- The MySQL database will be updated regularly with accurate sales data.
- Users will have valid login credentials and basic knowledge of dashboard usage.
- Required hardware and software resources will be available.
- Power BI Service will function without interruption during project use.

Dependencies:

- The system depends on Power BI for dashboard creation and data visualization.
- MySQL is required as the primary data source for all analytics.
- Django is used to host and manage the embedded dashboard.
- Availability of Power BI connectors and APIs is essential for real-time data refresh.
- Compatibility between browsers, operating systems, and Power BI is necessary for smooth performance.

## **3.3 SPECIFIC REQUIREMENTS**

### **3.3.1 External Interface Requirements**

#### **3.3.1.1 User Interface**

Users interact through a login-protected Django web application that embeds the Power BI dashboard, offering filters and drill-down capabilities.

#### **3.3.1.2 Hardware Interface**

The system operates on standard PC/laptop hardware with internet access and screen resolution of at least 1920×1080 pixels.

#### **3.3.1.3 Software Interface**

- MySQL for database operation
- Power Query for ETL
- DAX for KPI computation
- Django for web application embedding

### **3.3.2 Functional Requirements**

- Extract data from MySQL into Power BI
- Clean and transform data using Power Query
- Calculate metrics such as Revenue, Profit Margin %, YoY Growth
- Generate dynamic visuals for dashboards
- Embed dashboards within Django using iframe

### **3.3.3 Performance Requirements**

- Dashboards must load within 5 seconds on a standard internet connection
- Data refresh cycle should occur daily or in near real-time
- Must handle large datasets (thousands of transactions) without lag

### **3.3.4 Design Constraints**

- The star schema must be maintained in Power BI for performance
- User interface must remain consistent and mobile-friendly if needed
- Embedded dashboards must comply with Django framework constraints

### **3.3.5 Other Requirements**

- Validation of data through SQL queries and Excel
- Role-based access can be implemented in future using Power BI RLS or Django permissions
- The system should be scalable to accommodate more KPIs and additional regions/products

## **4.SYSTEM ANALYSIS AND DESIGN**

### **4.1 INTRODUCTION**

This section provides a detailed overview of the system analysis and design phase of the Sales Performance Analysis project. It focuses on how the system was structured, modeled, and implemented to meet business requirements. The primary goal of this phase is to translate project objectives into technical models that guide development and ensure the system is scalable, efficient, and user-friendly.

### **4.2 METHODOLOGY**

The methodology adopted for the Sales Performance Analysis project follows a structured approach based on the principles of data analytics and business intelligence. It involves six key phases, from data extraction to dashboard deployment.

#### **1. Data Extraction**

Sales-related data was sourced from a MySQL database using SQL queries. Five primary tables were extracted: sales\_transactions, customers, products, markets, and date. The WAMP server was used to host and manage the database locally.

#### **2. Data Cleaning and Transformation (ETL)**

The data was cleaned and transformed using Power Query in Power BI:

Null values were handled appropriately.

Currency formats were standardized (e.g., USD converted to INR).

Additional calculated columns were created: Profit Margin %, Cost Price.

Missing or incomplete entries were corrected using imputation techniques.

#### **3. Data Modeling**

A star schema was implemented in Power BI for efficient querying and performance:

Fact table: sales\_transactions

Dimension tables: customers, products, markets, date

Relationships were created with referential integrity to ensure consistency.

#### 4. KPI Definition using DAX

Custom measures were developed using DAX (Data Analysis Expressions) to generate key performance indicators:

- Total Revenue
- Profit Margin %
- Top Performing Customer
- Year-over-Year Growth

These metrics enable in-depth sales performance analysis.

#### 5. Dashboard Design

An interactive Power BI dashboard was created with three core pages:

Home Page: General overview of revenue, profits, and sales distribution.

Profit Analysis: Focus on cost price, target margin, and achieved profit.

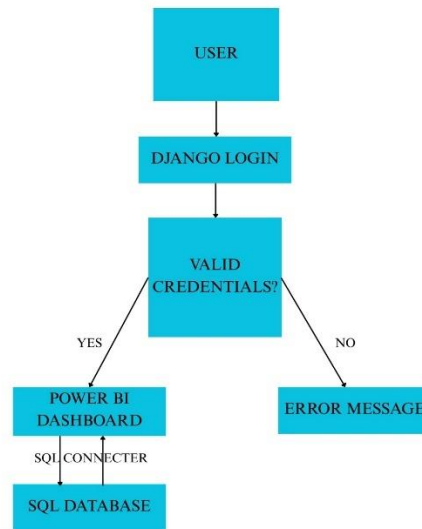
Performance Insights: Displays top/bottom customers, products, and regions.

Interactive features such as slicers, tooltips, and filters enhance user experience.

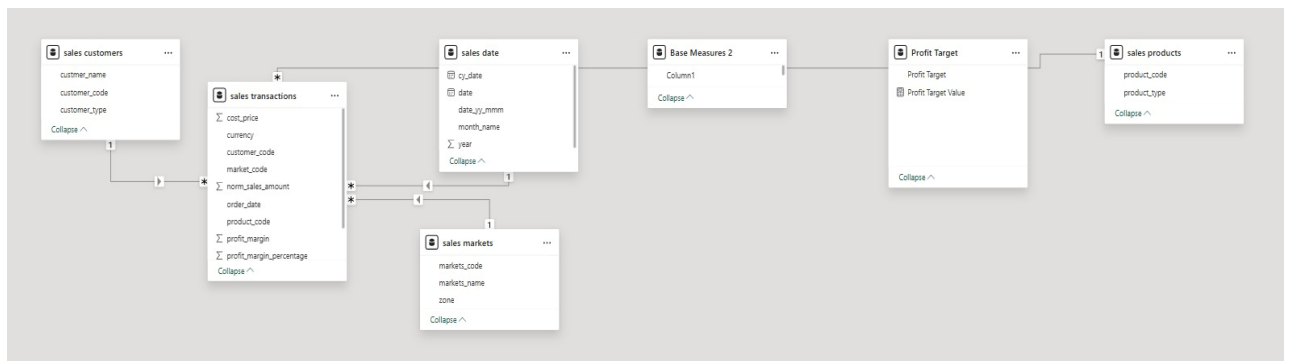
#### 6. Deployment via Django

To simulate a real-world enterprise use case, the dashboard was embedded into a Django web application using iframe integration. Users can log in through a secure HTML interface and access the dashboard online.

### **4.3 DATA FLOW DIAGRAM**



## 4.4 TABLE RELATIONSHIP



## 4.5 TABLE DESCRIPTION

TABLE NAME	DESCRIPTION
sales_transactions	This is the fact table containing all key transactional metrics including norm_sales_amount, cost_price, profit, profit_margin_percentage, currency, and associated foreign keys (customer_code, product_code, market_code, order_date). It drives the core of the analysis.
sales_customers	A dimension table holding customer-related details such as customer_name, customer_code, and customer_type. Linked to transactions via customer_code.
sales_products	Contains product metadata such as product_code and product_type. It helps

	filter and analyze product-wise sales performance.
sales_markets	Represents market-related geographic data like markets_code, markets_name, and zone. Useful for region-based filtering and comparison.
sales_date	A date dimension table that includes cy_date, month_name, year, and date_yy_mmm. Used for time-series analysis and filtering in the dashboard.
Profit Target	Holds target values for profitability. Includes Profit Target category and Profit Target Value, enabling comparison between actual and target profit margins.
Base Measures 2	Appears to be a supporting table, possibly used to store base metrics or calculations (e.g., thresholds, flags). The presence of a single column suggests a utility or helper role in DAX calculations or segmentation.

## 4.6 SYSTEM DESIGN IMPLEMENTATION

The system design implementation of the Sales Performance Analysis project focuses on how various components of the application interact to deliver a seamless experience for users accessing business insights.

The architecture integrates multiple technologies — MySQL for backend data storage, Power BI for data visualization, and Django for web interface delivery. The system follows a modular design approach where each component (data source, dashboard, user interface) functions independently but communicates through well-defined channels.

The implementation is user-centric. Once authenticated through the Django login page, users are presented with an embedded Power BI dashboard. The dashboard dynamically updates based on user-selected filters like region, time, and product category.

To model this interaction, the system includes:

- Use Case Diagram to represent how users interact with the system.
- Activity Diagram to show the step-by-step flow from login to dashboard usage.

This design ensures scalability, maintainability, and ease of use for both end-users and developers.



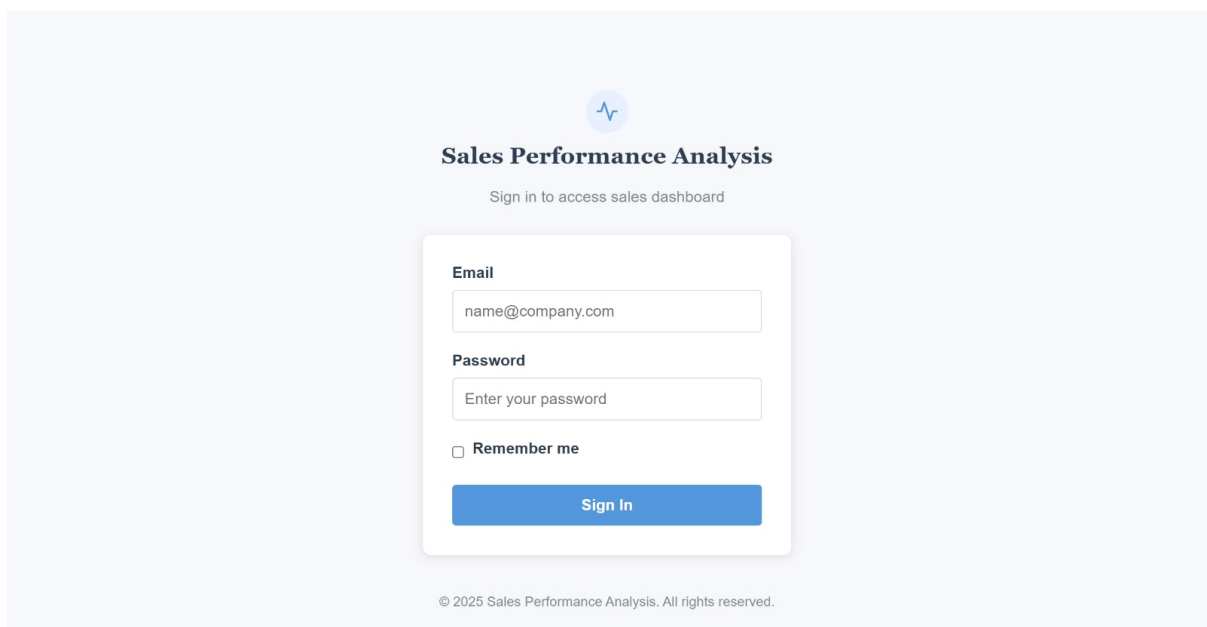
## 4.6 USER INTERFACE DIAGRAM

The user interface (UI) design of the Sales Performance Analysis system is built with simplicity and functionality in mind. The goal is to provide users with a clean, intuitive environment where they can easily view and interpret sales data.

The system includes:

Login Page:

Designed using HTML and CSS, the login interface ensures secure access to the dashboard. It includes fields for email and password with a professional layout and responsive design.

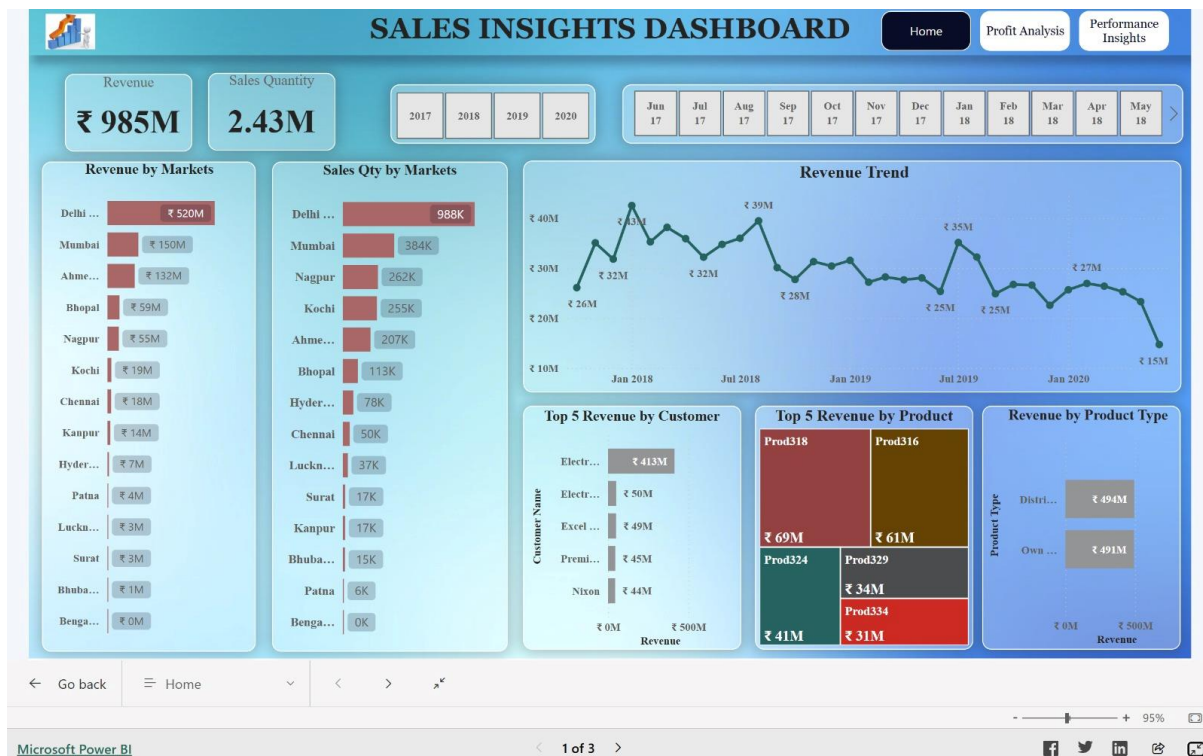


The screenshot displays a login page for 'Sales Performance Analysis'. At the top center is a blue circular icon with a white pulse line. Below it, the title 'Sales Performance Analysis' is written in bold, followed by the subtitle 'Sign in to access sales dashboard'. The main content is a white login form with a subtle shadow. Inside the form, there is an 'Email' label above a text input field containing 'name@company.com'. Below that is a 'Password' label above a text input field with the placeholder 'Enter your password'. A checkbox labeled 'Remember me' is positioned below the password field. At the bottom of the form is a blue 'Sign In' button. The footer of the page, centered, reads '© 2025 Sales Performance Analysis. All rights reserved.'

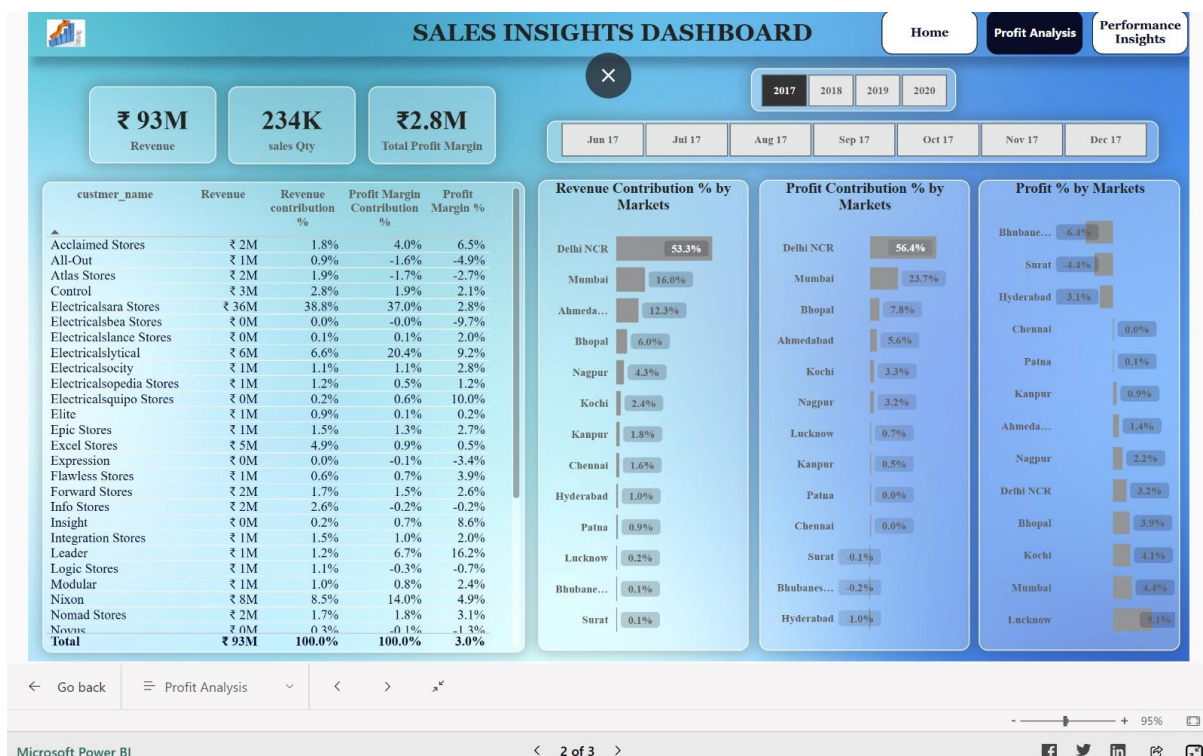
Dashboard Pages (Embedded via Power BI):

Once logged in, users are directed to the dashboard embedded using Power BI's iframe within Django. The dashboard consists of:

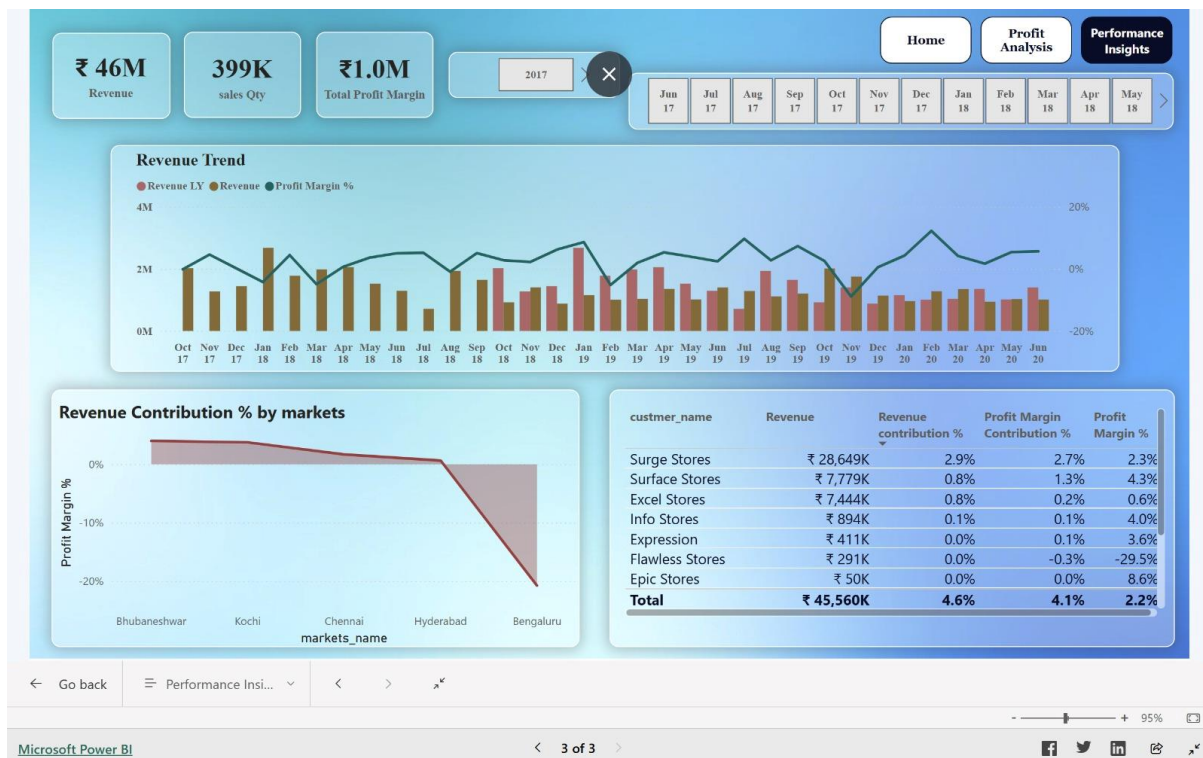
Home Page: Overview of total revenue, sales, and general performance.



Profit Analysis Page: Visualizations such as profit margins, cost price comparisons, and targets.



Performance Insights Page: Metrics like top customers, best-selling products, and under performing markets.



### Interactive Features:

- Slicers for filtering by year, month, region, and product.
- Tooltips on charts for detailed data.
- Responsive layout for different screen sizes.

This interface empowers users — especially managers and analysts — to explore data efficiently and derive actionable insights.

## 5.TESTING

### 5.1 INTRODUCTION

Testing is a critical phase in the software development life cycle to ensure the system performs as expected, is reliable, and is free of critical bugs. The Sales Performance Analysis system was tested to verify the functionality of the login module, data visualizations, filtering features, and dashboard responsiveness when embedded in the Django framework.

Both manual testing and data validation techniques were used to ensure accuracy and usability of the embedded Power BI dashboards.

### 5.2 Testing Objective

The main objectives of testing this project were:

- To ensure users can securely log in to the system.
- To verify that embedded Power BI dashboards are displayed correctly.
- To confirm slicers and filters function properly across all dashboard pages.
- To validate the accuracy of revenue, profit, and performance calculations.
- To ensure smooth performance across browsers and devices.

### 5.3 TEST CASES

#### 5.3.1 Login Functionality

Test Case ID	TC_001
Description	Verify that the user can log in with valid credentials.
Input	Email: user@example.com Password: correct_password
Expected Output	User is redirected to the dashboard.
Actual Output	User redirected successfully
Status	Pass

Test Case ID	TC_002
Description	Attempt login with incorrect credentials.
Input	Email: user@example.com Password: wrong_password
Expected Output	Display “Invalid credentials” message.
Actual Output	Error message displayed
Status	Pass

### 5.3.2 DASHBOARD RENDERING

Test Case ID	TC_003
Description	Verify that the embedded Power BI dashboard loads correctly.
Input	Access dashboard URL after login
Expected Output	Dashboard visuals are rendered in iframe
Actual Output	All visuals displayed properly
Status	Pass

### 5.3.3 DATA ACCURACY

Test Case ID	TC_004
Description	Verify revenue and profit calculations match backend SQL totals
Input	Cross-check totals in Power BI and SQL query
Expected Output	Totals should match
Actual Output	All totals matched successfully
Status	Pass

## **6. SYSTEM SECURITY**

### **5.1 INTRODUCTION**

System security is a crucial aspect of any web-based application to protect data integrity, prevent unauthorized access, and ensure safe user interaction. Since this project involves data visualization and user login, it implements basic yet essential security practices supported by the Django framework.

### **5.2 SOFTWARE SECURITY**

The following security measures were considered and implemented in the Sales Performance Analysis system:

#### **1. CSRF Protection**

Cross-Site Request Forgery (CSRF) protection is implemented using Django's built-in middleware. This ensures that malicious websites cannot submit forms on behalf of authenticated users without authorization.

How it works: Each form submission includes a hidden CSRF token. Django verifies this token on every POST request to prevent unauthorized commands.

Implementation: `{% csrf_token %}` is added in the HTML login form.

#### **2. Secure Authentication**

The login system uses Django's authentication mechanism which hashes passwords and validates credentials securely.

Inputs are validated on the server side.

Sessions are used to maintain user login state securely.

#### **3. Role-based Dashboard Embedding (Optional for Future)**

While not implemented now, the system architecture supports potential enhancements like Row-Level Security (RLS) in Power BI and Django role-based access to restrict dashboards based on user roles (e.g., Manager, Analyst).

#### 4. Data Isolation

The MySQL database connection is not exposed directly to the frontend. All data interactions are managed through Power BI and embedded securely via iframe within Django, limiting direct access to raw data.

## 7. CONCLUSION

This project successfully developed a comprehensive Sales Insights Dashboard that transforms raw sales data into actionable business intelligence. By leveraging PowerBI's robust visualization capabilities and implementing a structured data pipeline, we addressed the organization's critical challenge of manual, inefficient sales reporting.

This project not only delivered a working BI solution but also provided hands-on experience in end-to-end data analytics. It showed how business problems can be addressed using modern data tools like Power BI, turning raw numbers into clear insights.

- Some of the major takeaways from this project include:
- Understanding real-world business datasets and challenges.
- Gaining proficiency in MySQL, Power Query, DAX, and Power BI.
- Learning how to design effective dashboards for decision-makers.
- Experiencing a complete BI workflow from data to deployment.

The project demonstrates the impact of data visualization in enhancing business understanding and boosting organizational performance.



## **8. FUTURE ENHANCEMENTS**

### **Real-time Data Integration**

Currently, the data is static. Implementing real-time data refresh using APIs or live database connections would make the dashboard dynamic and current.

### **Predictive Analytics**

Integrate machine learning models to forecast future sales trends, customer demand, and product performance using Power BI + Azure ML or Python integration.

### **Role-Based Access**

Use Power BI service with row-level security (RLS) or embed dashboards in a Django web application with login-based access levels for Sales Directors, Managers, and Analysts.

### **Mobile Compatibility**

Optimize the dashboard layout for Power BI mobile app, making it easy to access insights on-the-go for field managers and executives.

### **Expand KPI Coverage**

Add additional KPIs like:

- Customer Lifetime Value (CLV)
- Customer Retention Rate
- Product Return Rate
- Sales Rep Performance

### **Drillthrough & What-If Analysis**

Use Power BI features like Drill

through Pages and What-If Parameters to allow deeper exploration and simulation of business scenarios.

## 9. BIBLIOGRAPHY

Turban, E., Sharda, R., & Delen, D. (2011). *Decision Support and Business Intelligence Systems* (9th ed.). Pearson Education.

Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed.). Wiley.

Few, S. (2006). *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly Media.

Russo, M., & Ferrari, A. (2021). *The Definitive Guide to DAX: Business Intelligence for Microsoft Power BI, SQL Server Analysis Services, and Excel* (2nd ed.). Microsoft Press.

Microsoft Corporation. (2023). *Power BI Documentation*. Retrieved from <https://learn.microsoft.com/en-us/power-bi/>

Django Software Foundation. (2023). *Django Documentation*. Retrieved from <https://docs.djangoproject.com/>

Singh, R., & Sharma, S. (2019). "Enhancing Retail Sales Insights Using Power BI: A Case Study." *International Journal of Data Science*, 4(2), 45–52.

CodeBasics. (2024). *Power BI Full Course – YouTube Channel*. Retrieved from <https://www.youtube.com/@codebasics>

SAP. (2020). *The Business Value of BI and Analytics*. SAP Insights. Retrieved from <https://insights.sap.com>

Gartner Inc. (2022). *Magic Quadrant for Analytics and Business Intelligence Platforms*. Retrieved from <https://www.gartner.com>

## 10. APPENDIX

### 1.LOGIN PAGE CODE

```
<form method="post">

    {% csrf_token %}

    <div class="form-group">

        <label for="email">Email</label>

        <input type="email" id="email" name="user" placeholder="Enter your email"
required>

    </div>

    <div class="form-group">

        <label for="password">Password</label>

        <input type="password" id="password" name="passwords" placeholder="Enter
your password" required>

    </div>

    <button type="submit" class="login-button">Sign In</button>

</form>
```

### 2. POWER BI DAX FORMULA

Measure	Formula
Revenue	Sum ( 'sales transactions'[norm_sales_amount])
Profit Margin %	DIVIDE([Total Profit Margin], [Revenue], 0)

Top Customer	TOPN(1,VALUES(customers[customer_name]), [Total Revenue])
--------------	--

### 3. SQL TABLE CREATION

**Eg:**

```
CREATE TABLE sales_transactions (
    transaction_id INT PRIMARY KEY,
    customer_id INT,
    product_id VARCHAR(10),
    market_id INT,
    order_date DATE,
    sales_qty INT,
    revenue DECIMAL(15,2),
    profit DECIMAL(15,2),
    currency VARCHAR(3),
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);
```

