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

FINAL PROJECT REPORT
ON EMPLOYEE
PRODUCTIVITY ANALYSIS

SUBMITTED BY

FATHIMA AULIYA (22BCACDC20)

LENA CHRISPIN FERNANDEZ (22BCACDC30)

ZIANA BINOY (22BCACDC69)

NANDHANA MANIKANDAN (22BCACDC55)

FATHIMA SHARAFUDEEN (22BSCFDC16)

BCA & BSC (CYBER FORENSICS DATA ANALYTICS AND CYBERSECURITY) WITH IBM

GUIDED BY
MR. SHASHANK

TABLE OF CONTENTS

LIST OF IMAGES

1.INTRODUCTION

1.1. OVERVIEW OF THE PROJECT

In the modern work environment, employee productivity is a critical metric for organizational success. This project focuses on analyzing employee productivity using a dynamic dashboard that enables HR and management to make informed decisions. A centralized portal allows secure login and real-time access to employee performance data, visualized through Power BI.

1.2. OBJECTIVE OF THE PROJECT

The objective of this project is to provide analytical insights into employee productivity. This includes.

Specific objectives include:

- Extracting and cleaning Employee data from a MySQL database.
- Developing interactive dashboards using Power BI.
- Calculating metrics like revenue, profit margin, and Employee 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: Employee data retrieved from MySQL.

Data Transformation: Performed using Power Query.

Data Modeling: Implemented a star schema with employee 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 organization encountered several challenges in monitoring and improving employee performance:

Manual Reports: HR teams relied heavily on Excel sheets and manual tracking, leading to inefficiency and inconsistencies.

Lack of Real-Time Insights: Delays in performance evaluation caused missed opportunities for timely intervention and support.

Inconsistent Data Tracking: Different departments maintained separate formats and criteria for performance evaluation, leading to inconsistent and subjective assessments.

Unnoticed Productivity Issues: Without analytical tools, underperformance and trends in productivity were difficult to identify or address effectively.

Solution:

An interactive Power BI dashboard was developed to automate data collection, visualize key employee productivity KPIs, enable dynamic filtering (by department, role, or time), and provide real-time insights through a secure web interface. This solution empowers HR and management to make data-driven decisions and foster continuous improvement.

2. LITERATURE REVIEW

The evolution of business intelligence (BI) tools has transformed how companies analyze data and make decisions. The use of dashboards for employee analysis has become a standard in the modern business world. This literature review explores existing studies, articles, and practices related to employee 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 employee, finance, and customer relationship management.

Employee 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 employee data across regions.

Power BI, developed by Microsoft, is a leading BI platform known for its flexibility, userfriendly 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 employee 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 employee 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 employee dashboards led to 35% faster decision-making. 40% improvement in identifying underperforming regions/products. Increased collaboration between employee 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 employee insight is not only an industry standard but also an academically validated method for improving employee performance and decisionmaking. 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 Employee Productivity Analysis system. It serves as a foundation for the design, development, and testing of the application. The goal of the project is to enhance workforce management by enabling real-time monitoring of employee performance through interactive dashboards developed using Power BI. The data is sourced from a MySQL database and visualized via a secure, loginbased interface.

3.1.2 Scope of the Project

The project involves the development of an Employee Performance Dashboard aimed at assisting HR and management in tracking productivity metrics. Traditionally, employee evaluation relied on manual reports and Excel-based tools, which lacked consistency and analytical depth. This solution automates data collection, enables advanced performance analytics, and provides visual insights into employee attendance, task efficiency, and overall productivity. The system includes secure web access for authorized users and supports dynamic filtering by department, role, and time period.

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 employee-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 Employee 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 Employee Performence 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:

HR Managers: To monitor employee attendance, task completion, and overall productivity for performance evaluation and decision-making.

Executives and Management: To gain insight into workforce trends, departmental performance, and organizational productivity levels

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 employee performance 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 Employee Productivity Analysis project. It outlines how the system was structured, modeled, and implemented to meet organizational performance monitoring requirements. The primary

objective of this phase is to translate the project goals into technical solutions that ensure the system is scalable, user-friendly, and efficient in providing actionable insights on employee productivity.

4.2 METHODOLOGY

The methodology adopted for this project follows a structured approach based on data analytics and business intelligence principles. It includes the following phases:

1. Data Extraction

Employee-related data—such as attendance, task logs, working hours, and departmental assignments—was sourced from a MySQL database using SQL queries. Key tables included: employee_info, task_details, attendance_records, performance_metrics, and date_dimension. The WAMP server was used to host and manage the database locally.

2. Data Cleaning and Transformation (ETL)

Using Power Query in Power BI, data was cleaned and transformed:

- Null or missing values were appropriately handled.
- Date formats were standardized for consistency.
- Calculated columns such as Task Efficiency %, Productivity Score, and Absentee Ratio were added.
- Incomplete records were fixed using logical imputation where feasible.

3. Data Modeling

A star scheme was implemented for better performance and scalability in Power BI:

- Fact Table: employee performance containing metrics like tasks completed, hours worked, attendance, and overall performance scores.
- Dimension Tables: employee_info, task_details, attendance_records, date dimension, and department.

Relationships were enforced with referential integrity for consistent data joins and filtering.

4. KPI Definition using DAX

Custom DAX measures were created to calculate essential KPIs such as:

- Task Completion Rate
- Attendance Percentage
- Productivity Index
- Average Task Duration

These metrics formed the basis for real-time productivity analysis and benchmarking across departments and roles.

5. Dashboard Design

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

Home Page: General overview of revenue, profits, and employee 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 TABLE DESCRIPTION

TABLE NAME	DESCRIPTION
All measures	A virtual table consisting of calculated measures using DAX. These are used for KPIs and aggregate metrics like percentages, counts, and totals.
Data	The primary table containing employee-related information. It includes demographics, job details, performance scores, and HR survey responses.

4.4 SYSTEM DESIGN IMPLEMENTATION

The system design implementation of the Employee Productivity Analysis project focuses on how various components of the application interact to deliver seamless experience for HR and management in accessing workforce 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.

4.5 USER INTERFACE DIAGRAM

The user interface (UI) design of the Employee Productivity Analysis system is built with simplicity, clarity, and ease of use in mind. The goal is to provide HR personnel and management with a clean, intuitive environment where they can quickly interpret employee data and draw actionable insights.

The system includes:

Login Page:

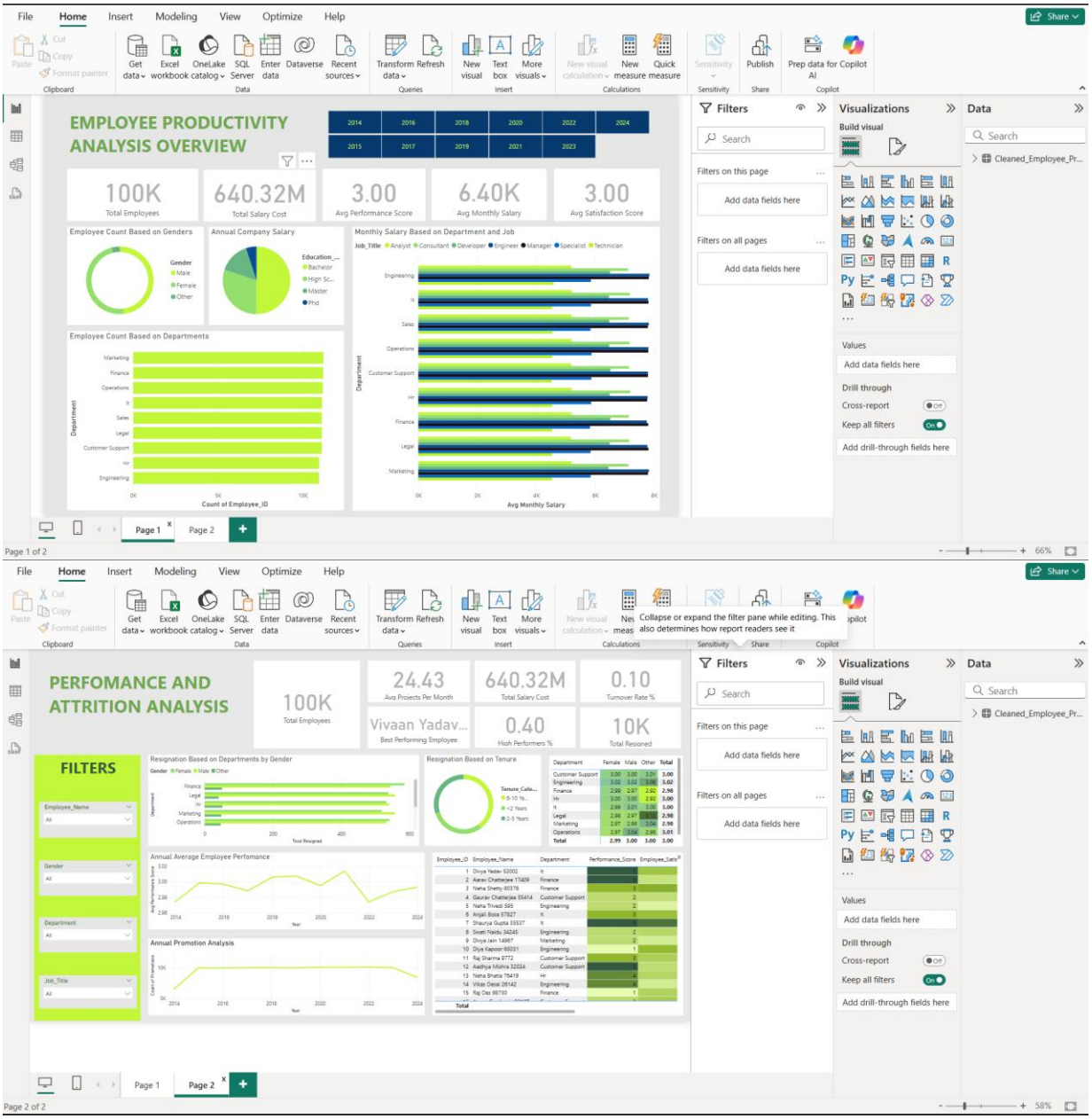
Designed using HTML and CSS, the login interface ensures secure access to the dashboard. It includes fields for email and password with a professional layout and responsive design.

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 Employee performance and Productivity Dashboard.

Employee Tracker page:



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 Employee 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 Employee 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. CONLUSION

This project successfully developed a centralized Employee Productivity Analysis dashboard. Using tools like Power BI and MySQL, the system allows HR to monitor and optimize employee performance. The login-enabled access ensures data privacy and structured visualization helps in data-driven decision-making.

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 trends in employee performance, such as potential attrition risk, engagement drop-off, or productivity spikes. By using Power BI with Azure Machine Learning 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 HR, Managers, and Analysts.

Mobile Compatibility

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

Expand KPI Coverage

Add additional KPIs like:

- Total Headcount
- Active Employees
- Employee Performance
- Employee Satisfaction
- Geographic Distribution

Drill through & What-If Analysis Use

Power BI features like Drill

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

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 Employee 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 %}
    <h2>Login</h2>
    <label>Username:</label>
    <input type="text" name="user" required>
    <label>Password:</label>
    <input type="password" name="passs" required>
    <input type="submit" value="Login">
</form>
```

2. POWER BI DAX FORMULA

Measure	Formula
Active%	DIVIDE([Active], [Headcount], 0)
Salary	SUM(data, [Salary],)
Terminated%	DIVIDE([Terminated], [Headcount], 0)

3. SQL TABLE CREATION

Eg:

```
CREATE TABLE IF NOT EXISTS hrdata (
  EmployeeID int DEFAULT NULL,
  EmployeeName varchar(512) DEFAULT NULL,
  Salary int DEFAULT NULL,
  Position varchar(512) DEFAULT NULL,
  State varchar(512) DEFAULT NULL,
  DateOfBirth varchar(512) DEFAULT NULL,
  Gender varchar(512) DEFAULT NULL,
  MaritalStatus varchar(512) DEFAULT NULL,
  HiringDate varchar(512) DEFAULT NULL,
  TerminationDate varchar(512) DEFAULT NULL, IN
  EmploymentStatus varchar(512) DEFAULT NULL,
  Department varchar(512) DEFAULT NULL,
  RecruitmentSource varchar(512) DEFAULT NULL,
  PerformanceScore varchar(512) DEFAULT NULL,
  EngagementSurvey double DEFAULT NULL,
  EmployeeSatisfaction int DEFAULT N
```