**PRODUCT MANAGEMENT**

***A Summer Internship Report submitted in partial fulfillment of the***

***requirements for the award of degree of***

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

## Submitted by:

KOPPISETTI BHARDWAJ SAI

22MH1A05I9



**Department Of Computer Science and Engineering**

**ADITYA ENGINEERING COLLEGE (A)**

**Approved by AICTE, Permanently affiliated to JNTUK & Accredited by NAAC with ‘A’ Grade**

**Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956**

**Aditya Nagar, ADB Road – Surampalem 533437, E.G. Dist., A.P.,**

**2023-2024.**

**ADITYA ENGINEERING COLLEGE (A)**

**Approved by AICTE, Permanently Affiliated to JNTUK & Accredited by NAAC with ‘A’ Grade**

**Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956**

**Aditya Nagar, ADB Road - Surampalem – 533437, E.G.Dist., A.P.,**

**Department Of Computer Science and Engineering**

****

**CERTIFICATE**

This is to certify that the Internship report entitled *“***PRODUCTMANAGEMENT***”* is being submitted by

**KOPPISETTI BHARDWAJ SAI (22MH1A05I9)**

In partial fulfillment of the requirements for award of the B.Tech degree in Computer Science and Engineering for the academic year 2023-2024.

**Internship Coordinator Head of the Department**

A. Hanumanth Rao, Ph.D. Dr. G.S.N. Murthy , Ph.D.

Assistant Professor Associate Professor

Department of CSE Department of CSE

**DECLARATION**

We hereby declare that the project entitled **“PRODUCT MANAGEMENT”** is a genuine project. This work has been submitted to the **ADITYA ENGINEERING COLLEGE,** Surampalem, permanently affiliated to **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA** in partial fulfillment of the **B.Tech** degree**.** We further declare that this project work has not been submitted in full or part of the award of any degree of this or any other educational institutions.

**by**

**KOPPISETTI BHARDWAJ SAI**

**(22MH1A05I9)**

**ACKNOWLEDGEMENT**

First I would like to thank the **Mr.** **Babji Neelam**, Surampalem for giving me the opportunity to do an internship within the **TechnicalHub**. I also would like all the people that worked along with me in **TechnicalHub,** Surampalem with their patience and openness they created an enjoyable working environment.

It is with immense pleasure that we would like to express our indebted gratitude to our Intenship coordinator **Mr. B.Pavan , Coding Trainer,** who has guided us a lot and encouraged us in every step of the intern project work, his valuable moral support and guidance throughout the Intern project helped us to a greater extent.

Our deepest thanks to **Dr. G.S.N Murthy, Associate Professor & Head of the Department** for inspiring us all the way and for arranging all the facilities and resources needed for our project.

We wish to thank **Dr. P.S.V.V.S.R Kumar, Professor** in CSE and Dean(Academics) for her support and suggestions during our project work.

We owe our sincere gratitude to **Dr. A. Ramesh, Principal** for providing a great support and for giving us the opportunity of doing the project.

We are thankful to our **College Management** for providing all the facilities in time to us for completion of our project.

Not to forget, **Faculty, Lab Technicians, non-teaching staff and our friends** who have directly or indirectly helped and supported us in completing our project in time.

# Abstract

# The Product Management system is a comprehensive Java-based application designed to enhance the efficiency of managing product information within a business. This project integrates Java's Abstract Window Toolkit (AWT) for creating an intuitive graphical user interface (GUI) and Java Database Connectivity (JDBC) for seamless interaction with a MySQL database. The system aims to streamline tasks such as inventory tracking, product categorization, sales management, and reporting, providing a robust solution for businesses to manage their product data effectively.

# At the core of the system lies the product catalog module, which allows users to perform CRUD (Create, Read, Update, Delete) operations on product data. The AWT-powered interface ensures that users can easily add new products, update existing ones, and delete obsolete entries. The application’s design focuses on user-friendliness, enabling even those with minimal technical expertise to navigate and manage the system effortlessly. By maintaining accurate and up-to-date product information, businesses can ensure efficient inventory management and reduce the risk of stockouts or overstocking.

# The system's business logic layer handles the application’s core functionalities and processes. This layer validates user inputs, performs necessary calculations, and ensures that data is accurately processed and stored. It acts as a mediator between the user interface and the database, maintaining the flow of data and enforcing business rules consistently. This separation of concerns not only simplifies maintenance but also allows for the easy addition of new features without disrupting the existing functionality.

# For database interactions, the system utilizes JDBC to connect to a MySQL database. This connection enables efficient execution of SQL queries for data retrieval, insertion, updates, and deletions. The use of JDBC ensures that the application can handle database operations securely and efficiently, maintaining data integrity and consistency. MySQL's robustness and support for complex queries and transactions make it an ideal choice for managing the extensive data requirements of the product management system.

# Security and scalability are key considerations in the system’s design. MySQL’s robust security features, including user authentication and access control, protect sensitive product data from unauthorized access. Regular backups and data recovery mechanisms ensure data reliability and availability. Additionally, MySQL’s support for replication and clustering allows the system to scale seamlessly as the volume of product data grows, ensuring that the application can handle increased load without compromising performance.

# Overall, the Product Management system provides a scalable, secure, and efficient solution for managing product information. By integrating AWT for an intuitive GUI and JDBC for reliable database interactions, the system enhances operational efficiency and supports informed decision-making. This comprehensive approach to product management makes the system an invaluable tool for businesses aiming to optimize their inventory processes and maintain accurate product data.

**Learning Objectives/Internship Objectives**

* Internships are generally thought of to be reserved for college students looking to gain experience in a particular field. However, a wide array of people can benefit from Training Internships in order to receive real world experience and develop their skills.
* An objective for this position should emphasize the skills you already possess in the area and your interest in learning more
* Internships are utilized in a number of different career fields, including architecture, engineering, healthcare, economics, advertising and many more.
* Some internships are used to allow individuals to perform scientific research while others are specifically designed to allow people to gain first-hand experience working.
* Utilizing internships is a great way to build your resume and develop skills that can be emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position.

# WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES

|  |  |  |  |
| --- | --- | --- | --- |
| **1st WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 13/5/24 | Monday | Reporting at office with all Photocopies of Documents.  Overview to Company Profile & Total Internship Schedule |
| 14/5/19 | Tuesday | Brief Introduction on Java Programming |
| 15/5/19 | Wednesday | Discuss PROS & CROS of Java |
| 16/5/19 | Thursday | Learning the different components |
| 17/5/19 | Friday | Learn how to write basic code |
| 18/5/19 | Saturday | Holiday |

|  |  |  |  |
| --- | --- | --- | --- |
| **2nd WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 19/5/19 | Sunday | Holiday |
| 20/5/19 | Monday | Learning connection of database |
| 21/5/19 | Tuesday | Working with java |
| 22/5/19 | Wednesday | Holiday |
| 23/5/19 | Thursday | Sample Projects |
| 24/5/19 | Friday | Sample Projects |

|  |  |  |  |
| --- | --- | --- | --- |
| **3rd WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 25/5/19 | Saturday | Holiday |
| 26/5/19 | Monday | Introduction to AWT |
| 27/5/19 | Tuesday | Configuring application with AWT tools |
| 28/5/19 | Wednesday | Holiday |
| 29/5/19 | Thursday | Sample Projects using AWT tools |
| 30/5/19 | Friday | Discuss about the latest projects |

|  |  |  |  |
| --- | --- | --- | --- |
| **4th WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 31/6/19 | Saturday | Discussion on left over Topics |
| 1/7/19 | Monday | Holiday |
| 2/7/19 | Tuesday | Holiday |
| 3/7/19 | Wednesday | Assigning Projects |
| 4/7/19 | Thursday | Implementation of Project |
| 5/7/19 | Friday | Project Presentation |
| 6/7/19 | Saturday | Submission of Project abstract & Presentation |

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **SNo.** |  | **Contents** | **Page** |
| 1. |  | Introduction | 11 |
| 2. |  | Project overview | 12 |
| 3. |  | Technologies Used | 12 |
| 4. |  | Project Architecture | 13 |
| 5. |  | Components Used | 14 |
|  | 5.1 | Java AWT | 14 |
|  | 5.2 | JDBC (Java Database Connectivity) | 14 |
|  | 5.3 | MYSQL Database | 15 |
| 6. |  | GUI Design | 16 |
| 7. |  | Implementing AWT Components | 17 |
|  | 7.1 | Frames and Panels | 17 |
|  | 7.2 | Buttons , TextFields , and Labels | 17 |
|  | 7.3 | Layouts | 18 |
| 8. |  | Database Connectivity |  |
|  | 8.1. | Configuring JDBC | 18 |
|  | 8.2 | Executing SQL Queries | 19 |
| 9. |  | Project Implementation | 19 |
| 10. |  | Source Code | 20 |
| 11. |  | Output | 42 |
| 12. |  | Conclusion | 43 |
| 13. |  | Bibliography & References | 44 |

**Introduction**

In today's fast-paced business environment, efficient product management is crucial for the success of any organization. The "Product Management" project aims to provide a comprehensive solution for managing various aspects of products, from inventory control to sales tracking. This project leverages the capabilities of Java, specifically utilizing Abstract Window Toolkit (AWT) for the user interface and Java Database Connectivity (JDBC) for database interactions. The combination of these technologies ensures a robust and user-friendly application that can streamline the product management process.

The Abstract Window Toolkit (AWT) in Java is a platform-independent windowing, graphics, and user-interface toolkit that provides a set of APIs for creating and managing graphical user interfaces (GUIs). In this project, AWT is employed to design an intuitive and responsive interface that allows users to easily navigate through different functionalities such as adding new products, updating existing product information, and generating reports. The choice of AWT ensures that the application can run seamlessly across different operating systems, providing a consistent user experience.

Java Database Connectivity (JDBC) is an API that enables Java applications to interact with a wide range of databases. For the Product Management project, JDBC is used to connect to the underlying database, allowing for efficient storage and retrieval of product data. This integration facilitates real-time updates and ensures data integrity, making it easier for businesses to maintain accurate and up-to-date product information. By utilizing JDBC, the project can support various database systems, offering flexibility and scalability to meet the growing needs of any organization.

Furthermore, the Product Management project emphasizes modularity and extensibility, allowing for future enhancements and customizations. The design of the application is structured in a way that separates the user interface, business logic, and data access layers. This separation of concerns not only simplifies maintenance but also enables the integration of additional features without significant changes to the existing codebase. For instance, businesses can easily add new functionalities like advanced analytics, customer feedback integration, or automated reordering systems.

**Project Overview**

The Product Management system is designed to streamline and optimize the management of products within a business environment. It provides a comprehensive suite of features aimed at simplifying tasks such as inventory tracking, product categorization, sales management, and reporting. The application is developed using Java, with a graphical user interface created using Abstract Window Toolkit (AWT) and robust database interactions managed through Java Database Connectivity (JDBC). This project is targeted towards businesses of all sizes looking to enhance their product management capabilities through a reliable and efficient software solution.

**Technologies Used**

The Product Management system leverages Java as its core programming language, chosen for its platform independence, robustness, and extensive library support. One of the key technologies used is the Abstract Window Toolkit (AWT), a part of Java's standard library for building graphical user interfaces. AWT provides a set of APIs that facilitate the creation of windows, buttons, menus, and other interactive components. In this project, AWT is utilized to develop an intuitive and responsive user interface, ensuring that users can easily navigate through the application’s features. Its event-driven architecture allows the application to handle user actions efficiently, providing a seamless and interactive experience across different operating systems.

Another crucial technology integrated into the system is Java Database Connectivity (JDBC), an API that enables Java applications to connect and interact with various database systems. JDBC is used in this project to manage all database operations, including data insertion, updates, deletions, and queries. This API ensures that the application can communicate with the underlying database in a secure and efficient manner, maintaining data integrity and consistency. JDBC's support for multiple database management systems offers flexibility in choosing the most suitable database for the application, whether it be MySQL, PostgreSQL, Oracle, or others. By using JDBC, the "Product Management" system can efficiently store, retrieve, and manipulate product data, ensuring that the information is always up-to-date and readily accessible for reporting and analytics.

**Project Architecture**

The architecture of the Product Management system is designed with a focus on modularity, scalability, and maintainability. The system is structured into three main layers: the presentation layer, the business logic layer, and the data access layer. This layered approach ensures a clear separation of concerns, allowing each layer to be developed, tested, and maintained independently. The presentation layer, built using Java's Abstract Window Toolkit (AWT), handles all user interactions and displays the graphical user interface (GUI). This layer is responsible for capturing user input, displaying product information, and generating reports, ensuring an intuitive and user-friendly experience.

The business logic layer serves as the core of the application, implementing the various functionalities and rules that govern the product management processes. This layer processes the user input received from the presentation layer, performs the necessary calculations and validations, and coordinates the flow of data between the presentation and data access layers. By encapsulating the business logic within this layer, the system ensures that all product management operations, such as adding new products, updating existing products, and generating sales reports, are performed consistently and correctly. This modular design also facilitates the addition of new features and enhancements, as changes to the business logic can be made without affecting the other layers.

The data access layer is responsible for interacting with the underlying database, utilizing Java Database Connectivity (JDBC) to perform all data-related operations. This layer manages the connection to the database, executes SQL queries, and handles the retrieval and storage of product information. By abstracting the database interactions within this layer, the system ensures that all data access is performed in a secure and efficient manner, maintaining data integrity and consistency. Additionally, this approach allows the application to support multiple database systems with minimal changes to the codebase. Overall, the three-layered architecture of the Product Management system ensures a robust, scalable, and maintainable solution that can adapt to the evolving needs of businesses.

**Java AWT**

A key element that the Product Management system uses to create its graphical user interface (GUI) is Java's Abstract Window Toolkit (AWT). A wide range of pre-built, cross-platform APIs are available with AWT to make it easier to construct interactive windows, buttons, text fields, menus, and other GUI elements. By ensuring that the program looks and feels the same on several operating platforms, this toolkit improves user experience. AWT is used to provide a user-friendly interface for the Product Management system, which enables users to rapidly navigate through a variety of features like controlling product stocks, seeing comprehensive product information, and producing reports. Because of AWT's event-driven architecture, the program can react quickly to user actions, which makes the system responsive and easy to use.

**JDBC (Java Database Connectivity)**

The Product Management system relies heavily on Java Database Connectivity (JDBC), which offers the required API for easy interaction with the underlying database. The application can manage database connections, carry out transactions, and run SQL statements thanks to JDBC. In order to guarantee that all changes are appropriately reflected in the database, this API is essential for carrying out CRUD (Create, Read, Update, Delete) actions on the product data. The application's ability to connect to numerous database management systems, including MySQL, PostgreSQL, and Oracle, is ensured by the use of JDBC, providing flexibility and scalability to satisfy various organizational needs. The system can maintain real-time data consistency and integrity by utilizing JDBC, which is necessary for precise inventory tracking and reporting.

JDBC's capacity to effectively handle database connections is essential to the Product Management system's dependability and performance. It ensures that the application functions properly even under heavy loads by handling the complexity of creating and terminating connections, maintaining connection pools, and carrying out intricate queries. Moreover, batch processing is supported via JDBC, which enables the program to run several SQL statements in a single session, improving performance and lowering network cost. JDBC's strong error-handling features aid in quickly locating and fixing database-related problems, maintaining the stability and dependability of the system.

**MYSQL Database**

The MySQL database is a cornerstone of the Product Management system, providing a reliable, high-performance, and scalable solution for managing product data. As an open-source relational database management system, MySQL offers numerous advantages, including strong support for SQL (Structured Query Language), which is used to manage and manipulate the data stored within the database. MySQL's robustness and efficiency make it an ideal choice for handling the complex data operations required by the product management application, such as tracking inventory levels, storing product details, and generating sales reports.

In the Product Management system, MySQL is used to store a wide range of product-related data, including product names, descriptions, prices, categories, stock quantities, and sales information. The database schema is designed to ensure data normalization, minimizing redundancy, and optimizing data integrity. Tables are carefully structured with primary and foreign keys to establish relationships between different data entities, enabling efficient querying and data retrieval. This structured approach not only ensures data consistency but also enhances the overall performance of the system, as queries can be executed quickly and efficiently.

One of the key features of MySQL that benefits the Product Management system is its support for transactions. Transactions ensure that a series of database operations are executed in a reliable and consistent manner, adhering to the ACID (Atomicity, Consistency, Isolation, Durability) properties. For example, when a new product is added to the inventory, multiple related tables may need to be updated simultaneously. MySQL transactions ensure that all these operations are completed successfully, and if any step fails, the entire transaction is rolled back to maintain data integrity. This transactional support is crucial for maintaining the accuracy and reliability of the product data.

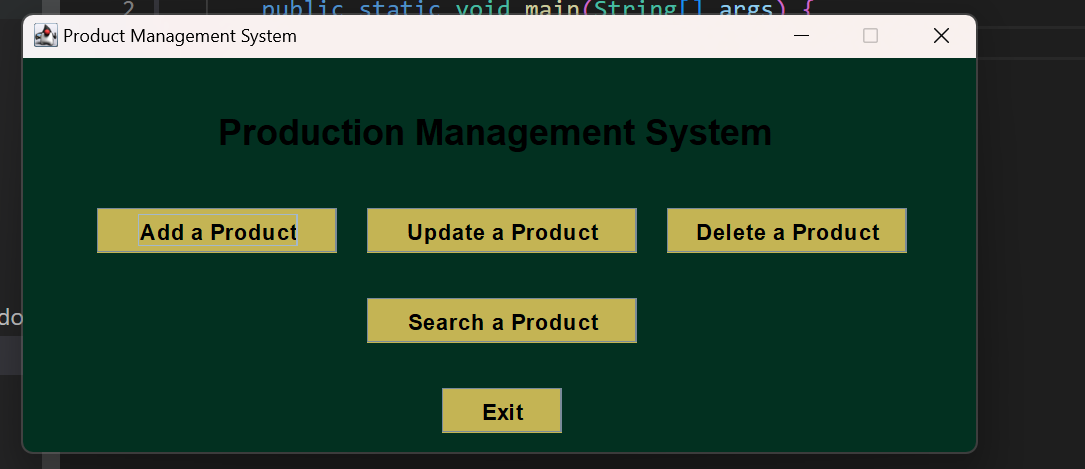
MySQL also provides robust security features, which are essential for protecting sensitive product information. The database management system includes user authentication, access control, and encryption capabilities, ensuring that only authorized users can access and manipulate the data. In the Product Management system, these security measures help safeguard the product data against unauthorized access and potential breaches. Regular backups and data recovery mechanisms further enhance the system's reliability, allowing for quick restoration of data in case of hardware failures or other unforeseen events.

**GUI Design**

Creating a table Products which consists:

1. Product Id
2. Product Name
3. Product Price
4. Category
5. Brand
6. Quantity

The main task appears like this



Buttons:

1. Add a Product:

It triggers a window which asks for the all the table categories and inserts them into the table

2. Update a Product:

It triggers a window same as add a product window which updates the product based on the product\_id

3. Delete a product:

It triggers a window which asks for the id to delete a particular product

4. Search a Product:

It triggers a window which has the buttons to search for using the categories of the table like search by using id or name or price or category or brand

By clicking in these buttons ,a window appears which asks for the specified category of the button we clicked and displays the table which contains the given information.

5. Exit: The exit button stops the execution of the program.

**Frames and Panels**

In the Product Management system, frames and panels play a crucial role in structuring the graphical user interface (GUI) using Java's Abstract Window Toolkit (AWT). A frame, which serves as the main window of the application, provides the primary container for all other UI components. Within this frame, panels are used to organize the layout, grouping related components together to create a cohesive and intuitive user experience. Panels act as sub-containers that can hold buttons, text fields, labels, and other interactive elements, making it easier to manage the placement and alignment of these components. By utilizing frames and panels, the Product Management system ensures a well-organized and user-friendly interface, enabling users to efficiently navigate through various functionalities, such as adding products, updating inventory, and generating reports.

**Buttons, Text Fields, and Labels**

In the Product Management system, buttons, text fields, and labels are essential components used to create an interactive and user-friendly graphical user interface (GUI). Buttons are crucial for initiating actions and commands within the application. They allow users to perform various tasks such as adding new products, updating existing product information, deleting items, and generating reports. Each button is designed to be easily recognizable and accessible, often accompanied by descriptive text or icons to clearly indicate its function. For example, a "Save" button enables users to store new product details in the database, while a "Delete" button removes selected products from the inventory. These buttons are strategically placed within the interface to provide a seamless and efficient workflow, enhancing overall user productivity.

Text fields and labels are equally important in the Product Management system for data input and display. Text fields provide users with editable areas where they can enter or modify product information, such as names, descriptions, prices, and quantities. These input fields are designed to handle various data types and ensure validation to prevent errors and inconsistencies. Labels, on the other hand, serve as static text elements that provide context and guidance to the users.. For instance, a label reading "Product Name" placed beside a text field indicates that the user should input the name of the product in that field. Together, buttons, text fields, and labels create an intuitive and organized interface that simplifies user interactions and enhances the overall usability of the "Product Management" system.

**Layouts**

Layouts in the Product Management system are fundamental for organizing the graphical user interface (GUI) components in a visually appealing and functional manner. Utilizing Java's Abstract Window Toolkit (AWT), various layout managers such as Border Layout, Flow Layout, and Grid Layout are employed to arrange buttons, text fields, labels, and other elements systematically. Each layout manager offers a unique way of positioning components; for instance, Border Layout divides the container into five regions (North, South, East, West, and Center), Flow Layout arranges components in a left-to-right flow, and Grid Layout places components in a grid of cells, ensuring consistent alignment and spacing. By effectively leveraging these layouts, the "Product Management" system ensures a clean, organized, and responsive interface, enhancing the user experience by making navigation and interaction straightforward and intuitive.

**Configuring JDBC**

Configuring JDBC in the Product Management system involves setting up the necessary components to establish a connection between the Java application and the MySQL database. This process begins with loading the JDBC driver, typically by including the appropriate MySQL JDBC driver JAR file in the existing project location and register it. Next, a connection string, which includes the database URL, username, and password, is used to establish a connection with the database using `DriverManager.getConnection()`. Once the connection is established, SQL statements can be executed through `Statement`, `PreparedStatement`, or `CallableStatement` objects, allowing for efficient interaction with the database. Proper configuration also involves handling exceptions and managing the connection lifecycle, including closing connections, statements, and result sets to prevent resource leaks. This careful setup ensures that the Product Management system can reliably and securely interact with the MySQL database, maintaining data integrity and performance.

**Executing SQL Queries**

Executing SQL queries in the Product Management system is a critical operation that allows the application to interact with the MySQL database effectively. Once the JDBC connection is established, SQL queries can be executed using `Statement` or `PreparedStatement` objects. For instance, to retrieve product data, a `SELECT` query can be executed, and the results are processed using a `ResultSet` object. For updating, inserting, or deleting records, `UPDATE`, `INSERT`, or `DELETE` queries are used, respectively, often executed with `executeUpdate()` for changes in the database. `PreparedStatement` is particularly useful for executing parameterized queries, enhancing security by preventing SQL injection attacks and improving performance with query precompilation. Exception handling ensures that any database errors are properly managed, and resources like statements and result sets are closed after use to maintain optimal performance and resource management. This process ensures that the "Product Management" system can perform all necessary database operations efficiently and securely, maintaining data consistency and reliability.

**Project Implementation**

The Product Management system is a robust Java-based application designed to streamline and optimize the management of product-related information within a business. Utilizing Java's Abstract Window Toolkit (AWT) for its graphical user interface and Java Database Connectivity (JDBC) for seamless database interactions, the system offers a comprehensive suite of features for tracking inventory, managing product details, and generating insightful reports. The application is built with a modular architecture, ensuring scalability and ease of maintenance. By efficiently handling tasks such as adding, updating, and deleting products, and providing real-time data access and analysis, the "Product Management" system enhances operational efficiency and supports informed decision-making, making it an invaluable tool for businesses of all sizes.

The Product Management system implementation involves integrating AWT for the GUI, JDBC for database connectivity, and MySQL for data storage, ensuring efficient product management and user interaction.

**Source Code**

**Java**

**public class PTrigger {**

**public static void main(String[] args) {**

**new PMain();**

**}**

**}**

**import java.awt.Color;**

**import javax.swing.JButton;**

**import javax.swing.JFrame;**

**import javax.swing.JLabel;**

**import javax.swing.JOptionPane;**

**import java.awt.Font;**

**import java.awt.event.ActionEvent;**

**import java.awt.event.ActionListener;**

**public class PMain extends JFrame implements ActionListener {**

**JButton b1, b2, b3, b4, b5, b6;**

**PMain() {**

**setSize(650, 300);**

**setTitle("Product Management System");**

**setResizable(false);**

**setLayout(null);**

**JLabel j1 = new JLabel("Product Management System");**

**j1.setForeground(Color.BLUE);**

**j1.setBounds(130, 10, 500, 80);**

**j1.setFont(new Font("Arial", Font.BOLD, 24));**

**Color code = new Color(2, 48, 32);**

**getContentPane().setBackground(code);**

**Color sadist\_blue = new Color(196, 180, 84);**

**b1 = new JButton("Add a Product");**

**b1.setBounds(50, 100, 160, 30);**

**b1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**b1.setForeground(Color.BLACK);**

**b1.setBackground(sadist\_blue);**

**b1.addActionListener(this);**

**b2 = new JButton("Update a Product");**

**b2.setBounds(230, 100, 180, 30);**

**b2.setFont(new Font("Pacifico", Font.BOLD, 15));**

**b2.setForeground(Color.BLACK);**

**b2.setBackground(sadist\_blue);**

**b2.addActionListener(this);**

**b3 = new JButton("Delete a Product");**

**b3.setBounds(430, 100, 160, 30);**

**b3.setFont(new Font("Pacifico", Font.BOLD, 15));**

**b3.setForeground(Color.BLACK);**

**b3.setBackground(sadist\_blue);**

**b3.addActionListener(this);**

**b4 = new JButton("Search a Product");**

**b4.setBounds(110, 160, 180, 30);**

**b4.setFont(new Font("Pacifico", Font.BOLD, 15));**

**b4.setForeground(Color.BLACK);**

**b4.setBackground(sadist\_blue);**

**b4.addActionListener(this);**

**b5 = new JButton("Exit");**

**b5.setBounds(270, 220, 100, 30);**

**b5.setFont(new Font("Pacifico", Font.BOLD, 15));**

**b5.setForeground(Color.BLACK);**

**b5.setBackground(sadist\_blue);**

**b5.addActionListener(this);**

**b6 = new JButton("View Data");**

**b6.setBounds(340, 160, 200, 30);**

**b6.setFont(new Font("Pacifico", Font.BOLD, 15));**

**b6.setForeground(Color.BLACK);**

**b6.setBackground(sadist\_blue);**

**b6.addActionListener(this);**

**add(j1);**

**add(b1);**

**add(b2);**

**add(b3);**

**add(b4);**

**add(b5);**

**add(b6);**

**setVisible(true);**

**setLocationRelativeTo(null);**

**setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);**

**}**

**public void actionPerformed(ActionEvent e) {**

**if (e.getSource() == b1)**

**new PAdd();**

**else if (e.getSource() == b2)**

**new PUpdate();**

**else if (e.getSource() == b3)**

**new PDelete();**

**else if (e.getSource() == b4)**

**new PSearch();**

**else if (e.getSource() == b5) {**

**int option = JOptionPane.showConfirmDialog(null, "Are you sure you want to exit?", "Exit Confirmation",**

**JOptionPane.YES\_NO\_OPTION);**

**if (option == JOptionPane.YES\_OPTION) {**

**System.exit(0);**

**}**

**} else if (e.getSource() == b6)**

**new PView();**

**}**

**}**

**import java.sql.Statement;**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.SQLException;**

**import javax.swing.JButton;**

**import javax.swing.JFrame;**

**import javax.swing.JLabel;**

**import javax.swing.JOptionPane;**

**import javax.swing.JTextField;**

**import java.awt.Color;**

**import java.awt.Font;**

**import java.awt.event.ActionListener;**

**import java.awt.event.ActionEvent;**

**public class PAdd extends JFrame {**

**JButton a1, a2;**

**PAdd() {**

**setSize(450, 360);**

**setTitle("Product Add");**

**setResizable(false);**

**setLayout(null);**

**Color code = new Color(2, 48, 32);**

**getContentPane().setBackground(code);**

**Color sadist\_blue = new Color(196, 180, 84);**

**JLabel j1 = new JLabel();**

**j1.setBounds(15, 10, 200, 30);**

**j1.setText("Enter ID of Product");**

**j1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j1.setForeground(Color.WHITE);**

**JTextField t1 = new JTextField();**

**t1.setBounds(220, 10, 110, 30);**

**t1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t1.setBackground(sadist\_blue);**

**JLabel j2 = new JLabel();**

**j2.setBounds(15, 50, 150, 30);**

**j2.setText("Enter Product Name");**

**j2.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j2.setForeground(Color.WHITE);**

**JTextField t2 = new JTextField();**

**t2.setBounds(220, 50, 200, 30);**

**t2.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t2.setBackground(sadist\_blue);**

**JLabel j3 = new JLabel();**

**j3.setBounds(15, 90, 150, 30);**

**j3.setText("Enter Product Price");**

**j3.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j3.setForeground(Color.WHITE);**

**JTextField t3 = new JTextField();**

**t3.setBounds(220, 90, 200, 30);**

**t3.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t3.setBackground(sadist\_blue);**

**JLabel j4 = new JLabel();**

**j4.setBounds(15, 130, 170, 30);**

**j4.setText("Enter Product Brand");**

**j4.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j4.setForeground(Color.WHITE);**

**JTextField t4 = new JTextField();**

**t4.setBounds(220, 130, 200, 30);**

**t4.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t4.setBackground(sadist\_blue);**

**JLabel j5 = new JLabel();**

**j5.setBounds(15, 170, 180, 30);**

**j5.setText("Enter Product Category");**

**j5.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j5.setForeground(Color.WHITE);**

**JTextField t5 = new JTextField();**

**t5.setBounds(220, 170, 200, 30);**

**t5.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t5.setBackground(sadist\_blue);**

**JLabel j6 = new JLabel();**

**j6.setText("Enter Product Quantity ");**

**j6.setBounds(15, 210, 180, 30);**

**j6.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j6.setForeground(Color.WHITE);**

**JTextField t6 = new JTextField();**

**t6.setBounds(220, 210, 200, 30);**

**t6.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t6.setBackground(sadist\_blue);**

**a1 = new JButton("Submit");**

**a1.setBounds(230, 280, 100, 30);**

**a1.setBackground(Color.WHITE);**

**a1.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e) {**

**try {**

**Connection con = DriverManager.getConnection(**

**"jdbc:mysql://localhost:3306/charan", "root",**

**"BhardwajSai@123");**

**Statement st = con.createStatement();**

**String s1 = t1.getText().toUpperCase();**

**String s2 = t2.getText().toUpperCase();**

**String s3 = t3.getText().toUpperCase();**

**String s4 = t4.getText().toUpperCase();**

**String s5 = t5.getText().toUpperCase();**

**String s6 = t6.getText();**

**String query = "insert into products values(" + s1 + ",'" + s2 + "'," + s3 + ",'" + s4 + "','" + s5**

**+ "'," + s6 + ")";**

**int result = JOptionPane.showConfirmDialog(null,**

**"Are you sure you want to submit this form?",**

**"Confirm Submit",**

**JOptionPane.YES\_NO\_OPTION);**

**if (result == JOptionPane.YES\_OPTION) {**

**try {**

**int rs = st.executeUpdate(query);**

**if (rs == 0) {**

**throw new SQLException(**

**"Data cannot be inserted");**

**}**

**else {**

**JOptionPane.showMessageDialog(null,**

**"Data is inserted in the table",**

**"Success",**

**JOptionPane.INFORMATION\_MESSAGE);**

**t1.setText("");**

**t2.setText("");**

**t3.setText("");**

**t4.setText("");**

**t5.setText("");**

**t6.setText("");**

**}**

**} catch (SQLException e1) {**

**JOptionPane.showMessageDialog(null,**

**"ERROR: " + e1.getMessage(),**

**"ERROR", JOptionPane.ERROR\_MESSAGE);**

**t1.setText("");**

**t2.setText("");**

**t3.setText("");**

**t4.setText("");**

**t5.setText("");**

**t6.setText("");**

**}**

**}**

**} catch (SQLException e3) {**

**}**

**}**

**});**

**a2 = new JButton("Clear");**

**a2.setBounds(90, 280, 100, 30);**

**a2.setBackground(Color.WHITE);**

**a2.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e) {**

**int clr = JOptionPane.showConfirmDialog(null, "Are you sure", "Clear",**

**JOptionPane.YES\_NO\_OPTION);**

**if (clr == JOptionPane.YES\_OPTION) {**

**t1.setText("");**

**t2.setText("");**

**t3.setText("");**

**t4.setText("");**

**t5.setText("");**

**t6.setText("");**

**}**

**}**

**});**

**add(j1);**

**add(j2);**

**add(j3);**

**add(j4);**

**add(j5);**

**add(j6);**

**add(t1);**

**add(t2);**

**add(t3);**

**add(t4);**

**add(t5);**

**add(t6);**

**add(a1);**

**add(a2);**

**setLocationRelativeTo(null);**

**setVisible(true);**

**setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);**

**}**

**}**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import java.sql.Statement;**

**import javax.swing.JButton;**

**import javax.swing.JFrame;**

**import javax.swing.JLabel;**

**import javax.swing.JOptionPane;**

**import javax.swing.JTextField;**

**import java.awt.Color;**

**import java.awt.Font;**

**import java.awt.event.ActionEvent;**

**import java.awt.event.ActionListener;**

**public class PUpdate extends JFrame {**

**String c1;**

**JLabel q1, q2, q3, q4, q5, q6;**

**JTextField jf1;**

**PUpdate() {**

**setSize(500, 630);**

**setTitle("Product Update");**

**setResizable(false);**

**setLayout(null);**

**Color code = new Color(2, 48, 32);**

**getContentPane().setBackground(code);**

**Color sadist\_blue = new Color(196, 180, 84);**

**JLabel cru = new JLabel();**

**cru.setText("Enter the Product Id to view details");**

**cru.setBounds(15, 30, 270, 30);**

**cru.setFont(new Font("Pacifico", Font.BOLD, 15));**

**cru.setForeground(Color.WHITE);**

**jf1 = new JTextField();**

**jf1.setBounds(300, 30, 80, 30);**

**jf1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**jf1.setBackground(sadist\_blue);**

**JButton m = new JButton();**

**m.setText("View");**

**m.setBounds(400, 30, 70, 30);**

**m.setBackground(Color.WHITE);**

**m.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e) {**

**try {**

**Connection con = DriverManager.getConnection(**

**"jdbc:mysql://localhost:3306/charan", "root",**

**"BhardwajSai@123");**

**Statement st = con.createStatement();**

**c1 = jf1.getText();**

**String query = "select \* from products where id=" + c1;**

**try {**

**ResultSet res = st.executeQuery(query);**

**if (res.next()) {**

**clearLabels();**

**setUpLabels();**

**int id = res.getInt("id");**

**String name = res.getString("PName");**

**int price = res.getInt("PPrice");**

**String brand = res.getString("PBrand");**

**String category = res.getString("PCategory");**

**int quantity = res.getInt("PQuantity");**

**q1.setText("ID: " + id);**

**q2.setText("Name: " + name);**

**q3.setText("Price: " + price);**

**q4.setText("Brand: " + brand);**

**q5.setText("Category: " + category);**

**q6.setText("Quantity: " + quantity);**

**revalidate();**

**repaint();**

**add(q1);**

**add(q2);**

**add(q3);**

**add(q4);**

**add(q5);**

**add(q6);**

**} else {**

**throw new SQLException("No Data Found");**

**}**

**} catch (SQLException e4) {**

**JOptionPane.showMessageDialog(null,**

**"ERROR: " + e4.getMessage(),**

**"ERROR", JOptionPane.ERROR\_MESSAGE);**

**}**

**} catch (SQLException e3) {**

**JOptionPane.showMessageDialog(null,**

**"ERROR: " + e3.getMessage() + c1,**

**"ERROR", JOptionPane.ERROR\_MESSAGE);**

**}**

**}**

**});**

**JLabel main = new JLabel("Enter the Details to be Updated");**

**main.setBounds(15, 230, 250, 30);**

**main.setFont(new Font("Pacifico", Font.BOLD, 15));**

**main.setForeground(Color.RED);**

**add(main);**

**JLabel j1 = new JLabel();**

**j1.setBounds(15, 270, 200, 30);**

**j1.setText("Enter ID of Product");**

**j1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j1.setForeground(Color.WHITE);**

**JTextField t1 = new JTextField();**

**t1.setBounds(220, 270, 110, 30);**

**t1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t1.setBackground(sadist\_blue);**

**JLabel j2 = new JLabel();**

**j2.setBounds(15, 310, 150, 30);**

**j2.setText("Enter Product Name");**

**j2.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j2.setForeground(Color.WHITE);**

**JTextField t2 = new JTextField();**

**t2.setBounds(220, 310, 200, 30);**

**t2.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t2.setBackground(sadist\_blue);**

**JLabel j3 = new JLabel();**

**j3.setBounds(15, 350, 150, 30);**

**j3.setText("Enter Product Price");**

**j3.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j3.setForeground(Color.WHITE);**

**JTextField t3 = new JTextField();**

**t3.setBounds(220, 350, 200, 30);**

**t3.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t3.setBackground(sadist\_blue);**

**JLabel j4 = new JLabel();**

**j4.setBounds(15, 390, 170, 30);**

**j4.setText("Enter Product Brand");**

**j4.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j4.setForeground(Color.WHITE);**

**JTextField t4 = new JTextField();**

**t4.setBounds(220, 390, 200, 30);**

**t4.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t4.setBackground(sadist\_blue);**

**JLabel j5 = new JLabel();**

**j5.setBounds(15, 430, 180, 30);**

**j5.setText("Enter Product Category");**

**j5.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j5.setForeground(Color.WHITE);**

**JTextField t5 = new JTextField();**

**t5.setBounds(220, 430, 200, 30);**

**t5.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t5.setBackground(sadist\_blue);**

**JLabel j6 = new JLabel();**

**j6.setText("Enter Product Quantity ");**

**j6.setBounds(15, 470, 180, 30);**

**j6.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j6.setForeground(Color.WHITE);**

**JTextField t6 = new JTextField();**

**t6.setBounds(220, 470, 200, 30);**

**t6.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t6.setBackground(sadist\_blue);**

**JButton a1 = new JButton("Submit");**

**a1.setBounds(230, 510, 100, 30);**

**a1.setBackground(Color.WHITE);**

**a1.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e) {**

**try {**

**Connection con = DriverManager.getConnection(**

**"jdbc:mysql://localhost:3306/charan", "root",**

**"BhardwajSai@123");**

**Statement st = con.createStatement();**

**String s1 = t1.getText().trim();**

**String s2 = t2.getText().trim();**

**String s3 = t3.getText().trim();**

**String s4 = t4.getText().trim();**

**String s5 = t5.getText().trim();**

**String s6 = t6.getText().trim();**

**StringBuilder query = new StringBuilder("update products set ");**

**boolean updateNeeded = false;**

**if (!s1.isEmpty()) {**

**try {**

**int id = Integer.parseInt(s1);**

**query.append("id=").append(id).append(", ");**

**updateNeeded = true;**

**} catch (NumberFormatException e1) {**

**JOptionPane.showMessageDialog(null,**

**"Please enter a valid number for ID",**

**"Input Error", JOptionPane.ERROR\_MESSAGE);**

**return;**

**}**

**}**

**if (!s2.isEmpty()) {**

**query.append("PName='").append(s2).append("', ");**

**updateNeeded = true;**

**}**

**if (!s3.isEmpty()) {**

**try {**

**int price = Integer.parseInt(s3);**

**query.append("PPrice=").append(price).append(", ");**

**updateNeeded = true;**

**} catch (NumberFormatException e1) {**

**JOptionPane.showMessageDialog(null,**

**"Please enter a valid number for Price",**

**"Input Error", JOptionPane.ERROR\_MESSAGE);**

**return;**

**}**

**}**

**if (!s4.isEmpty()) {**

**query.append("PBrand='").append(s4).append("', ");**

**updateNeeded = true;**

**}**

**if (!s5.isEmpty()) {**

**query.append("PCategory='").append(s5).append("', ");**

**updateNeeded = true;**

**}**

**if (!s6.isEmpty()) {**

**try {**

**int quantity = Integer.parseInt(s6);**

**query.append("PQuantity=").append(quantity).append(", ");**

**updateNeeded = true;**

**} catch (NumberFormatException e1) {**

**JOptionPane.showMessageDialog(null,**

**"Please enter a valid number for Quantity",**

**"Input Error", JOptionPane.ERROR\_MESSAGE);**

**return;**

**}**

**}**

**if (!updateNeeded) {**

**JOptionPane.showMessageDialog(null, "No fields to update", "Update Error",**

**JOptionPane.ERROR\_MESSAGE);**

**return;**

**}**

**query.setLength(query.length() - 2);**

**query.append(" where id=").append(c1);**

**int result = JOptionPane.showConfirmDialog(null,**

**"Are you sure you want to submit this form?",**

**"Confirm Submit",**

**JOptionPane.YES\_NO\_OPTION);**

**if (result == JOptionPane.YES\_OPTION) {**

**int rs = st.executeUpdate(query.toString());**

**if (rs == 0) {**

**throw new SQLException("Data cannot be updated");**

**} else {**

**JOptionPane.showMessageDialog(null,**

**"Data is updated in the table",**

**"Success",**

**JOptionPane.INFORMATION\_MESSAGE);**

**t1.setText("");**

**t2.setText("");**

**t3.setText("");**

**t4.setText("");**

**t5.setText("");**

**t6.setText("");**

**}**

**}**

**} catch (SQLException e3) {**

**JOptionPane.showMessageDialog(null,**

**"ERROR: " + e3.getMessage(),**

**"ERROR", JOptionPane.ERROR\_MESSAGE);**

**}**

**}**

**});**

**JButton a2 = new JButton("Clear");**

**a2.setBounds(90, 510, 100, 30);**

**a2.setBackground(Color.WHITE);**

**a2.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e) {**

**int clr = JOptionPane.showConfirmDialog(null, "Are you sure", "Clear",**

**JOptionPane.YES\_NO\_OPTION);**

**if (clr == JOptionPane.YES\_OPTION) {**

**t1.setText("");**

**t2.setText("");**

**t3.setText("");**

**t4.setText("");**

**t5.setText("");**

**t6.setText("");**

**}**

**}**

**});**

**add(cru);**

**add(jf1);**

**add(j1);**

**add(j2);**

**add(j3);**

**add(j4);**

**add(j5);**

**add(j6);**

**add(t1);**

**add(t2);**

**add(t3);**

**add(t4);**

**add(t5);**

**add(t6);**

**add(m);**

**add(a1);**

**add(a2);**

**setLocationRelativeTo(null);**

**setVisible(true);**

**setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);**

**}**

**private void setUpLabels() {**

**q1 = new JLabel();**

**q1.setBounds(30, 50, 100, 30);**

**q1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**q1.setForeground(Color.WHITE);**

**q2 = new JLabel();**

**q2.setBounds(30, 80, 150, 30);**

**q2.setFont(new Font("Pacifico", Font.BOLD, 15));**

**q2.setForeground(Color.WHITE);**

**q3 = new JLabel();**

**q3.setBounds(30, 110, 150, 30);**

**q3.setFont(new Font("Pacifico", Font.BOLD, 15));**

**q3.setForeground(Color.WHITE);**

**q4 = new JLabel();**

**q4.setBounds(30, 140, 150, 30);**

**q4.setFont(new Font("Pacifico", Font.BOLD, 15));**

**q4.setForeground(Color.WHITE);**

**q5 = new JLabel();**

**q5.setBounds(30, 170, 200, 30);**

**q5.setFont(new Font("Pacifico", Font.BOLD, 15));**

**q5.setForeground(Color.WHITE);**

**q6 = new JLabel();**

**q6.setBounds(30, 200, 200, 30);**

**q6.setFont(new Font("Pacifico", Font.BOLD, 15));**

**q6.setForeground(Color.WHITE);**

**add(q1);**

**add(q2);**

**add(q3);**

**add(q4);**

**add(q5);**

**add(q6);**

**}**

**public void clearLabels() {**

**if (q1 != null)**

**remove(q1);**

**if (q2 != null)**

**remove(q2);**

**if (q3 != null)**

**remove(q3);**

**if (q4 != null)**

**remove(q4);**

**if (q5 != null)**

**remove(q5);**

**if (q6 != null)**

**remove(q6);**

**revalidate();**

**repaint();**

**}**

**}**

**import java.awt.Color;**

**import java.awt.Font;**

**import java.awt.event.ActionListener;**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.SQLException;**

**import java.sql.Statement;**

**import java.awt.event.ActionEvent;**

**import javax.swing.JButton;**

**import javax.swing.JFrame;**

**import javax.swing.JLabel;**

**import javax.swing.JOptionPane;**

**import javax.swing.JTextField;**

**public class PDelete extends JFrame {**

**PDelete() {**

**setSize(500, 300);**

**setTitle("Product Delete");**

**setResizable(false);**

**setLayout(null);**

**Color code = new Color(2, 48, 32);**

**getContentPane().setBackground(code);**

**Color sadist\_blue = new Color(196, 180, 84);**

**JLabel l1 = new JLabel();**

**l1.setText("Enter the Product ID");**

**l1.setBounds(30, 100, 200, 30);**

**l1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**l1.setForeground(Color.WHITE);**

**JTextField t1 = new JTextField();**

**t1.setBounds(220, 100, 110, 30);**

**t1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**t1.setBackground(sadist\_blue);**

**JButton d = new JButton("Delete");**

**d.setBounds(250, 170, 100, 30);**

**d.setBackground(Color.WHITE);**

**d.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e) {**

**try {**

**Connection con = DriverManager.getConnection(**

**"jdbc:mysql://localhost:3306/charan", "root",**

**"BhardwajSai@123");**

**Statement st = con.createStatement();**

**String getid = t1.getText();**

**String query = "delete from products where id=" + getid;**

**int result = JOptionPane.showConfirmDialog(null,**

**"Are you sure you want to delete this Product?",**

**"Confirm Delete",**

**JOptionPane.YES\_NO\_OPTION);**

**if (result == JOptionPane.YES\_OPTION) {**

**try {**

**int rs = st.executeUpdate(query);**

**if (rs == 0) {**

**throw new SQLException(**

**"Table does not contains the data with the id:" + getid);**

**}**

**else {**

**JOptionPane.showMessageDialog(null,**

**"Data is deleted from the table",**

**"Success",**

**JOptionPane.INFORMATION\_MESSAGE);**

**t1.setText("");**

**}**

**} catch (SQLException e1) {**

**JOptionPane.showMessageDialog(null,**

**"ERROR: " + e1.getMessage(),**

**"ERROR", JOptionPane.ERROR\_MESSAGE);**

**t1.setText("");**

**}**

**}**

**} catch (SQLException e2) {**

**}**

**}**

**});**

**JButton d1=new JButton("Clear");**

**d1.setBounds(130, 170, 80, 30);**

**d1.setBackground(Color.WHITE);**

**d1.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e){**

**int clr = JOptionPane.showConfirmDialog(null, "Are you sure", "Clear",**

**JOptionPane.YES\_NO\_OPTION);**

**if (clr == JOptionPane.YES\_OPTION) {**

**t1.setText("");**

**}**

**}**

**});**

**add(d);**

**add(d1);**

**add(l1);**

**add(t1);**

**setLocationRelativeTo(null);**

**setVisible(true);**

**setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);**

**}**

**}**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import java.sql.Statement;**

**import java.util.Vector;**

**import javax.swing.JButton;**

**import javax.swing.JComboBox;**

**import javax.swing.JFrame;**

**import javax.swing.JLabel;**

**import javax.swing.JOptionPane;**

**import javax.swing.JScrollPane;**

**import javax.swing.JTable;**

**import javax.swing.JTextField;**

**import java.awt.Color;**

**import java.awt.Font;**

**import java.awt.event.ActionEvent;**

**import java.awt.event.ActionListener;**

**public class PSearch extends JFrame {**

**PSearch() {**

**setSize(800, 600);**

**setTitle("Product Search");**

**setLayout(null);**

**setResizable(false);**

**JLabel j1 = new JLabel("Category");**

**j1.setBounds(30, 30, 150, 30);**

**j1.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j1.setForeground(Color.BLACK);**

**add(j1);**

**JComboBox<String> categoryComboBox = new JComboBox<>();**

**categoryComboBox.setBounds(30, 60, 150, 30);**

**add(categoryComboBox);**

**JTextField minPriceField = new JTextField();**

**minPriceField.setBounds(200, 60, 100, 30);**

**add(minPriceField);**

**JLabel j2 = new JLabel("Min Price");**

**j2.setBounds(200, 30, 100, 30);**

**j2.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j2.setForeground(Color.BLACK);**

**add(j2);**

**JTextField maxPriceField = new JTextField();**

**maxPriceField.setBounds(320, 60, 100, 30);**

**add(maxPriceField);**

**JLabel j3 = new JLabel("Max Price");**

**j3.setBounds(320, 30, 100, 30);**

**j3.setFont(new Font("Pacifico", Font.BOLD, 15));**

**j3.setForeground(Color.BLACK);**

**add(j3);**

**JButton searchButton = new JButton("Search");**

**searchButton.setBounds(450, 60, 100, 30);**

**add(searchButton);**

**JScrollPane scrollPane = new JScrollPane();**

**scrollPane.setBounds(30, 120, 700, 400);**

**add(scrollPane);**

**JTable table = new JTable();**

**scrollPane.setViewportView(table);**

**try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/charan", "root",**

**"BhardwajSai@123");**

**Statement st = con.createStatement();**

**ResultSet rs = st.executeQuery("SELECT DISTINCT PCategory FROM products")) {**

**while (rs.next()) {**

**categoryComboBox.addItem(rs.getString("PCategory"));**

**}**

**categoryComboBox.setSelectedItem(null);**

**} catch (SQLException ex) {**

**JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage(), "Error", JOptionPane.ERROR\_MESSAGE);**

**}**

**searchButton.addActionListener(new ActionListener() {**

**public void actionPerformed(ActionEvent e) {**

**String category = (String) categoryComboBox.getSelectedItem();**

**String minPriceStr = minPriceField.getText();**

**String maxPriceStr = maxPriceField.getText();**

**String query = "SELECT \* FROM products WHERE 1=1";**

**if (category != null && !category.isEmpty()) {**

**query += " AND PCategory='" + category + "'";**

**}**

**if (!minPriceStr.isEmpty()) {**

**double minPrice = Double.parseDouble(minPriceStr);**

**query += " AND PPrice >= " + minPrice;**

**}**

**if (!maxPriceStr.isEmpty()) {**

**double maxPrice = Double.parseDouble(maxPriceStr);**

**query += " AND PPrice <= " + maxPrice;**

**}**

**try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/charan", "root",**

**"BhardwajSai@123");**

**Statement st = con.createStatement();**

**ResultSet rs = st.executeQuery(query)) {**

**Vector<Vector<Object>> data = new Vector<>();**

**Vector<String> columnNames = new Vector<>();**

**int columnCount = rs.getMetaData().getColumnCount();**

**for (int i = 1; i <= columnCount; i++) {**

**columnNames.add(rs.getMetaData().getColumnName(i));**

**}**

**while (rs.next()) {**

**Vector<Object> row = new Vector<>();**

**for (int i = 1; i <= columnCount; i++) {**

**row.add(rs.getObject(i));**

**}**

**data.add(row);**

**}**

**table.setModel(new javax.swing.table.DefaultTableModel(data, columnNames));**

**} catch (SQLException ex) {**

**JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage(), "Error",**

**JOptionPane.ERROR\_MESSAGE);**

**}**

**}**

**});**

**setLocationRelativeTo(null);**

**setVisible(true);**

**setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);**

**}**

**}**

**import java.sql.Statement;**

**import java.util.Vector;**

**import java.sql.Connection;**

**import java.sql.DriverManager;**

**import java.sql.ResultSet;**

**import java.sql.SQLException;**

**import javax.swing.JFrame;**

**import javax.swing.JOptionPane;**

**import javax.swing.JScrollPane;**

**import javax.swing.JTable;**

**// import javax.swing.table.TableColumn;**

**import java.sql.ResultSetMetaData;**

**public class PView extends JFrame {**

**JScrollPane sc;**

**PView() {**

**setSize(1300, 1000);**

**setTitle("All Products");**

**setResizable(false);**

**try {**

**Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/charan", "root",**

**"BhardwajSai@123");**

**Statement st = con.createStatement();**

**ResultSet rs = st.executeQuery("select \* from products");**

**Vector<Vector<Object>> data = new Vector<>();**

**Vector<String> columnNames = new Vector<>();**

**ResultSetMetaData metaData = rs.getMetaData();**

**int columnCount = metaData.getColumnCount();**

**for (int i = 1; i <= columnCount; i++) {**

**columnNames.add(metaData.getColumnName(i));**

**}**

**while (rs.next()) {**

**Vector<Object> row = new Vector<>();**

**for (int i = 1; i <= columnCount; i++) {**

**row.add(rs.getObject(i));**

**}**

**data.add(row);**

**}**

**JTable table = new JTable(data, columnNames);**

**// TableColumn c1 = table.getColumnModel().getColumn(1);**

**// c1.setPreferredWidth(200);**

**sc = new JScrollPane(table);**

**} catch (SQLException e) {**

**JOptionPane.showMessageDialog(null, "Error: " + e.getMessage(), "Error", JOptionPane.ERROR\_MESSAGE);**

**}**

**add(sc);**

**setVisible(true);**

**setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);**

**}**

**}**

**MYSQL**

create table products(id int,

PName varchar(15),

PPrice int,

PBrand varchar(15),

PCategory varchar(15),

PQuantity int);

**Output**

The output of the "Product Management" system encompasses a wide array of functionalities and data representations that significantly enhance the efficiency and effectiveness of product management within a business. Key outputs include detailed product listings, inventory status reports, sales data, and various analytical insights. Users can view comprehensive product catalogs that display crucial information such as product names, descriptions, prices, stock levels, and categories. These listings are dynamically updated, ensuring that users always have access to the most current product information. The intuitive AWT-based interface ensures that these outputs are presented in a clear and organized manner, making it easy for users to interpret and act upon the data.

Another significant output of the system is the generation of detailed reports and analytics. Users can produce reports that summarize sales performance, inventory turnover, and product demand trends. These reports provide valuable insights that aid in strategic decision-making, such as identifying best-selling products, forecasting future demand, and optimizing stock levels to reduce holding costs. The system’s ability to present these reports in various formats, including charts and graphs, enhances their readability and utility. This feature allows businesses to quickly assess their product management performance and make informed decisions to drive efficiency and profitability.

Additionally, the system outputs actionable alerts and notifications that help manage inventory effectively. For example, users can receive alerts for low stock levels, upcoming reorder points, or discrepancies between expected and actual inventory counts. These notifications ensure that potential issues are addressed promptly, preventing stockouts and overstock situations. By providing real-time updates and automated alerts, the "Product Management" system empowers businesses to maintain optimal inventory levels, improve customer satisfaction, and streamline their operations. Overall, the outputs generated by the system provide comprehensive, real-time data and insights that are crucial for effective product management and strategic planning.

**Conclusion**

In conclusion, the Product Management system stands as a vital tool for businesses aiming to streamline their product management processes. By leveraging the strengths of Java's Abstract Window Toolkit (AWT) and Java Database Connectivity (JDBC), the system provides a user-friendly interface and robust database interactions, ensuring efficient management of product information. The modular architecture, with clearly defined layers for the user interface, business logic, and data access, enhances the system's scalability and maintainability. This thoughtful design ensures that the application can adapt to evolving business needs, supporting a wide range of functionalities from inventory tracking to detailed reporting.

The integration of MySQL as the backend database further strengthens the system's reliability and performance. MySQL's robust transactional support and security features ensure data integrity and protection, while its scalability options cater to growing data requirements. Overall, the "Product Management" system not only improves operational efficiency but also supports strategic decision-making by providing accurate and real-time product data. This project exemplifies how effective use of modern technologies can lead to the development of comprehensive solutions that address the complex needs of businesses, making it an indispensable asset for any organization looking to enhance its product management capabilities.

**Bibliography & References**

[**https://docs.oracle.com/javase/8/docs/**](https://docs.oracle.com/javase/8/docs/)

[**https://docs.oracle.com/javase/8/docs/api/java/awt/package-summary.html**](Summer_Intern_report%5b1%5d.docx)

[**https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/**](Summer_Intern_report%5b1%5d.docx)