Documentation

**Project Documentation: Retrieving Product Details Using Product ID**

**Overview**

This servlet-based web application provides a module to retrieve product details using a product ID. It consists of a form for users to enter a product ID, and upon submission, the application queries a MySQL database to display the product details. If the product is not found, an error message is displayed.

**Technologies Used:**

* **IDE**: Eclipse
* **Web Server**: Apache Tomcat
* **Database**: MySQL
* **JDBC**: MySQL Connector

**Setup and Configuration**

**Prerequisites:**

* Java Development Kit (JDK)
* Apache Tomcat
* Eclipse IDE
* MySQL Server
* MySQL Connector/J

**Project Structure:**

scssCopy code

project-root │ ├── src/main/java/com │ ├── ProductServlet.java │ ├── WebContent │ ├── productSearch.html (Product ID Input Page) │ ├── productDetails.jsp (Product Details Page) │ └── WEB-INF ├── lib (MySQL Connector JAR) ├── web.xml (Servlet Configuration)

**Configuration:**

1. **MySQL**: Create a database with a product table and prepopulate it with data.
2. **Eclipse**: Set up Apache Tomcat and add the MySQL Connector/J JAR to the project.
3. **web.xml**: Configure the servlet.
4. **HTML and JSP Pages**: Includes the pages for product ID input and displaying product details.

**Usage**

**Product Search Page:**

* URL: **http://localhost:8080/YourProjectName/productSearch.html**
* Enter Product ID and submit the form.

**Product Details Page:**

* Displays the product details or an error message if the product is not found.

**Step-by-Step Process:**

1. **Create MySQL Database and Table**:
   * Set up a MySQL database and create a product table with fields such as ID, name, description, price, etc.
   * Prepopulate the table with product data.
2. **Set Up Eclipse Project**:
   * Create a Dynamic Web Project in Eclipse.
   * Add Apache Tomcat as the server.
   * Include MySQL Connector/J in the project's lib folder.
3. **Create HTML Page**:
   * Design a simple HTML form to accept product ID.
4. **Create Servlet**:
   * Write a servlet that uses JDBC to connect to MySQL.
   * Query the database using the entered product ID.
   * Display the product details or an error message.
5. **Test the Application**:
   * Run the application on the Tomcat server.
   * Test by entering various product IDs.

**Troubleshooting**

* Check the JDBC URL and credentials.
* Ensure that MySQL Connector/J is properly added to the project.
* Verify the SQL query and product table structure.

**Conclusion**

This servlet-based application forms a critical part of an e-commerce backend, allowing admins to easily retrieve product information. The integration of HTML, servlets, and JDBC provides a robust and scalable solution.

By following the above steps and utilizing the provided resources, one can effectively build and test this application. The flexible design allows for easy modifications and scaling to suit specific needs.