Computer Management System

Name: S.H.L Welagedara

Reg No: SEU/IS/20/MIT/032

Year/Semester: 2nd Year 2nd Semester

SOUTH EASTERN UNIVERSITY OF SRI LANKA

Table of Contents

1. Introduction

The Computer Management System is a web-based application designed to manage a computer inventory database.
It provides users with an intuitive platform to perform CRUD (Create, Read, Update, Delete) operations on computer records.
The application is developed using Java Servlets, JSP, MySQL, Eclipse IDE, and Apache Tomcat, ensuring a robust and scalable system.

2. Objectives

The primary objectives of the Computer Management System are:
1. To simplify the management of computer inventory records.
2. To facilitate CRUD operations securely with backend integration.
3. To ensure scalability and data consistency using Servlets and JSP with MySQL.

3. Features of the System

1. Home Page:
   - A user-friendly interface for navigation and login/registration.

2. Dashboard:
   - Displays a list of computer records in tabular format with the following details:
     - Name, Brand, Type (Brand New, Used, Broken), Available.
     - Action buttons for editing or deleting records.

3. CRUD Operations:
   - Add New Computer: Allows users to add new records.
   - Edit Computer: Enables modifications to existing records.
   - Delete Computer: Deletes selected records.
   - View Computers: Lists all records in the database.

4. Technologies Used

- Programming Language: Java (Servlets & JSP)
- Development Environment: Eclipse IDE
- Database: MySQL
- Web Server: Apache Tomcat
- Frontend: HTML, CSS, JavaScript

5. System Design

Database Design

Table: computers

Column Name	Data Type	Constraints
id	INT	Primary Key, Auto-Increment
name	VARCHAR(50)	NOT NULL
brand	VARCHAR(50)	NOT NULL
type	ENUM('Brand New', 'Used', 'Broken')	NOT NULL
available	BOOLEAN	NOT NULL

6. Implementation

Prerequisites

- Eclipse IDE with Apache Tomcat configured.
- MySQL Database with the computers table.

Modules

1. Add New Computer:
   - JSP page (addComputer.jsp) collects user inputs.
   - Servlet (AddComputerServlet.java) inserts new records into the database.

2. View Computers:
   - JSP page (dashboard.jsp) displays all records.

3. Edit Computer:

- JSP page (editComputer.jsp) pre-fills existing data for editing.
- Servlet (EditComputerServlet.java) updates modified data.

4. Delete Computer:
   - Servlet (DeleteComputerServlet.java) removes selected records.

7. Sample Code

Servlet: Add Computer

```
@WebServlet("/addComputer")
public class AddComputerServlet extends HttpServlet {
   protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
      String name = request.getParameter("name");
      String brand = request.getParameter("brand");
      String type = request.getParameter("type");
      boolean available = request.getParameter("available") != null;

      try {
         Connection conn = DatabaseUtil.getConnection();
         String sql = "INSERT INTO computers (name, brand, type,
available) VALUES (?, ?, ?, ?)";
         PreparedStatement stmt = conn.prepareStatement(sql);
         stmt.setString(1, name);
         stmt.setString(2, brand);
         stmt.setString(3, type);
         stmt.setBoolean(4, available);
         stmt.executeUpdate();
         response.sendRedirect("dashboard.jsp");
      } catch (SQLException e) {
         e.printStackTrace();
      }
   }
}
```

JSP: Dashboard

```
<table>
  <tr>
    <th>ID</th>
    <th>Name</th>
    <th>Brand</th>
    <th>Type</th>
    <th>Available</th>
    <th>Actions</th>
  </tr>
  <%
    Connection conn = DatabaseUtil.getConnection();
    String sql = "SELECT * FROM computers";
    PreparedStatement stmt = conn.prepareStatement(sql);
    ResultSet rs = stmt.executeQuery();
    while (rs.next()) {
  %>
  <tr>
    <td><%= rs.getInt("id") %></td>
    <td><%= rs.getString("name") %></td>
    <td><%= rs.getString("brand") %></td>
    <td><%= rs.getString("type") %></td>
    <td><%= rs.getBoolean("available") ? "Yes" : "No" %></td>
    <td>
      <a href="editComputer.jsp?id=<%= rs.getInt("id") %>">Edit</a>
      <a href="deleteComputer?id=<%= rs.getInt("id") %>"
onclick="return confirm('Are you sure?')">Delete</a>
    </td>
  </tr>
  <% } %>
</table>
```

Servlet: Edit Computer

```
@WebServlet("/editComputer")
public class EditComputerServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        int id = Integer.parseInt(request.getParameter("id"));
        String name = request.getParameter("name");
        String brand = request.getParameter("brand");
        String type = request.getParameter("type");
        boolean available = request.getParameter("available") != null;

        try {
            Connection conn = DatabaseUtil.getConnection();
            String sql = "UPDATE computers SET name = ?, brand = ?, type = ?, available = ? WHERE id = ?";
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, name);
            stmt.setString(2, brand);
            stmt.setString(3, type);
            stmt.setBoolean(4, available);
            stmt.setInt(5, id);
            stmt.executeUpdate();
            response.sendRedirect("dashboard.jsp");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

8. Conclusion

The Computer Management System provides a scalable solution for efficiently managing computer inventory records.
By enabling CRUD operations through a web interface and leveraging Servlets and JSP for backend integration,
this project achieves a robust and user-friendly design.