**Writeup for Railway Crossing Status**

**Railway Crossing Status Management System**

**Overview:**

The Railway Crossing Status Management System is a web application designed to manage information about trains, crossings, and user interactions. The system is implemented using JavaServer Pages (JSP), Java Servlets, and a MySQL database.

**Database Schema:**

The application utilizes a MySQL database with the following tables:

1. **trains:**
   * id (Primary Key, Auto-incremented)
   * train\_name
   * crossing\_status
   * person\_in\_charge
   * train\_schedule
   * landmark
   * address
2. **users:**
   * user\_id (Primary Key, Auto-incremented)
   * username
   * password
   * email
3. **user\_favorite:**
   * user\_id (Foreign Key referencing users)
   * train\_id (Foreign Key referencing trains)

**Web Pages:**

1. **index.jsp:**
   * Landing page with navigation links to login and register.
2. **login.jsp:**
   * User login page with input fields for username and password.
   * Includes a link to register.jsp for new users.
   * Handles authentication and redirects to admin-dashboard.jsp or user-dashboard.jsp based on credentials.
3. **register.jsp:**
   * User registration page with input fields for username, password, and email.
   * Inserts new user details into the 'users' table in the database.
   * Redirects to login.jsp after successful registration.
4. **admin-dashboard.jsp:**
   * Admin dashboard displaying train details in a table.
   * Provides functionality to add trains to the 'trains' table.
   * Each train entry includes an "Add to Favorite" button.
5. **user-dashboard.jsp:**
   * User dashboard displaying train details in a table.
   * Provides functionality to add trains to the 'user\_favorite' table.
   * Each train entry includes an "Add to Favorite" button, linking to favorite.jsp.
6. **favorite.jsp:**
   * Displays favorite trains for the logged-in user from the 'user\_favorite' table.
   * Each favorite entry includes a "Remove" button.

**Servlets:**

1. **LoginServlet:**
   * Handles user authentication during login.
   * Redirects to admin-dashboard.jsp for admin users and user-dashboard.jsp for regular users.
2. **AdminDashboardServlet:**
   * Retrieves and displays train details for the admin dashboard.
   * Handles adding new trains to the 'trains' table.
3. **UserDashboardServlet:**
   * Retrieves and displays train details for the user dashboard.
   * Handles adding favorite trains to the 'user\_favorite' table.
4. **FavoriteServlet:**
   * Retrieves and displays favorite trains for the logged-in user.
   * Handles removing favorite trains from the 'user\_favorite' table.

**JDBC Connection:**

* Establishes a connection to the MySQL database using JDBC in each servlet.
* Executes SQL queries and updates to interact with the 'trains', 'users', and 'user\_favorite' tables.

**Session Management:**

* Utilizes HttpSession to manage user sessions and store user information.

**Recommendations:**

* The program emphasizes modularity and separation of concerns.
* Best practices, such as parameterized queries, are employed to prevent SQL injection.