Phase-2 Project:

Course-end Project 2: Gym Management System Backend

Introduction:

The Gym Management System is a backend solution designed to assist Gym Instructor Yamin in managing his participants and batches for Zumba sessions. The project involves developing a Java Enterprise application with Servlets, connecting to a MySQL database, and implementing CRUD operations.

Project Setup:

1. Dynamic Web Project Creation:

Created a Dynamic Web Project in Eclipse IDE and configured it as a Maven project for efficient build and dependency management.

Java Classes:

1. Participant and Batch Classes:

Developed Java classes for Participant and Batch entities to represent the data model. These classes encapsulate properties such as participant/batch ID, name, and session time.

Servlets:

1. Participant and Batch Servlets:

Created Servlets to handle CRUD operations for Participants and Batches.

Implemented methods for handling HTTP GET, POST, PUT, and DELETE requests to perform corresponding operations on the database.

HTML Pages:

1. Frontend HTML Pages:

Designed HTML pages to provide user interfaces for adding, updating, and deleting Participants and Batches.

Developed a Welcome Page with a navigation menu for easy access to different functionalities.

JDBC and Database Setup:

1. CRUD Operations using JDBC:

Utilized JDBC (Java Database Connectivity) for database interactions.

Implemented CRUD operations including creating, reading, updating, and deleting records in the MySQL database.

1. Database Configuration:

Created a MySQL database schema and tables to store Participant and Batch information.

Configured JDBC dependencies in Maven to establish database connectivity.

DAO Design Pattern:

1. DAO Implementation:

Implemented the DAO (Data Access Object) design pattern to separate data access logic from business logic.

Developed Repository classes to interact with the database using DAOs, ensuring better modularity and maintainability.

JSP Pages:

1. JSP Pages for Presentation:

Created JSP pages to display lists of Participants and Batches with options to delete records.

Designed a JSP page to view Participants in a specific Batch using query parameters.

Build and Deployment:

1. Project Deployment:

Built and deployed the project on Apache Tomcat web server for testing and validation.

Ensured the project's functionality by testing CRUD operations and frontend interactions.

Conclusion:

The Gym Management System backend project successfully addresses the requirements of managing participants and batches for Zumba sessions. By leveraging Java Enterprise technologies, JDBC for database connectivity, and Maven for project management, a robust and efficient solution has been developed. The implementation of CRUD operations, DAO design pattern, and frontend integration ensures a user-friendly and scalable application.

Execution steps:

Step 1: Download Eclipse:

1. Go to the Eclipse download page: https://www.eclipse.org/downloads/
2. Choose the Eclipse IDE for Java EE Developers package and download it.
3. Extract the downloaded zip file to a folder of your choice.

Step 2: Create a Maven Project:

1. Open Eclipse IDE.
2. Go to "File" -> "New" -> "Other".
3. In the "New" dialog box, expand "Maven" and select "Maven Project", then click "Next".
4. Check the box "Create a simple project (skip archetype selection)" and click "Next".
5. Enter the "Group Id" and "Artifact Id" for your project and click "Finish". This will create a basic Maven project structure.

Step 3: Install Tomcat Server:

1. Go to the Apache Tomcat download page: https://tomcat.apache.org/download-90.cgi (replace '90' with the appropriate version).
2. Download the binary distribution of Tomcat by clicking on the "zip" link under "Core" for your desired version.
3. Extract the downloaded zip file to a folder of your choice.

Step 4: Configure Tomcat Server in Eclipse:

1. In Eclipse, go to "Window" -> "Preferences".
2. Expand "Server" and select "Runtime Environments".
3. Click "Add..." to add a new server runtime environment.
4. Select "Apache Tomcat" and click "Next".
5. Browse and select the directory where you extracted Tomcat in the previous step.
6. Click "Finish" to complete the configuration.

Step 5: Import and Configure Project:

1. In Eclipse, right-click in the "Project Explorer" and select "Import".
2. Choose "Existing Maven Projects" and click "Next".
3. Browse to the directory where you created your Maven project and select it.
4. Click "Finish" to import the project into Eclipse.

Step 6: Run the Project:

1. Right-click on your project in Eclipse and select "Run As" -> "Run on Server".
2. Choose "Tomcat" as the server and click "Next".
3. Select the Tomcat server you configured earlier and click "Finish".
4. Eclipse will deploy your project to the Tomcat server, and you should be able to access it by opening a web browser and navigating to http://localhost:8080/yourprojectname.