Assessment Project: CB FSD – Backend and Database Development

Author: Rogelio Lotho

Goal: To build a Gym Membership application to handle and manage the data detail recordings of the participants and batches.

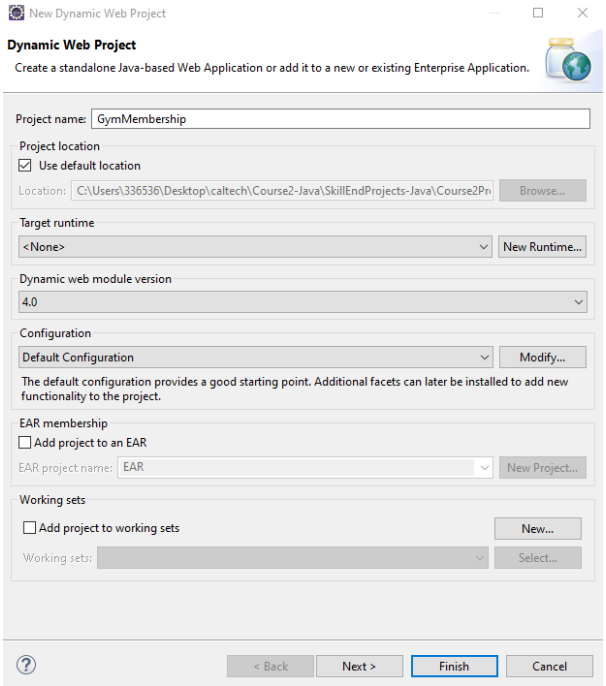
Feature: Design Server side Controller Structure with Servlets implementing Database connections to perform CRUD operation.

Tech Stack: Java, Servlets, JSP, Maven, JDBC, HTML, MySQL, Git and GitHub

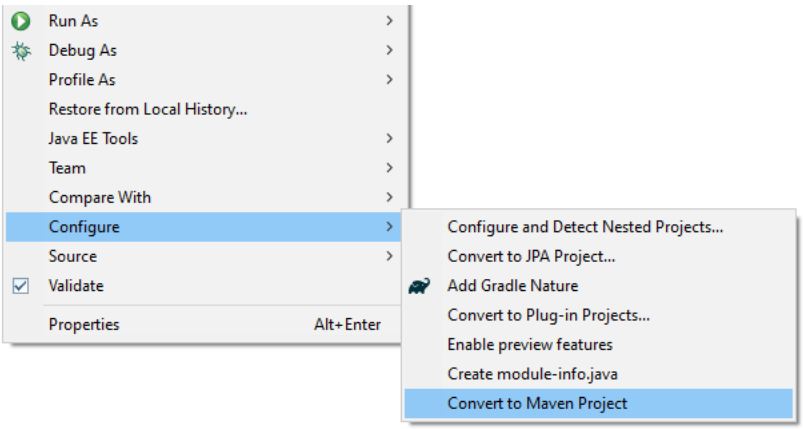
---

1. Create a Dynamic Web Project in Eclipses and configure it to Maven project

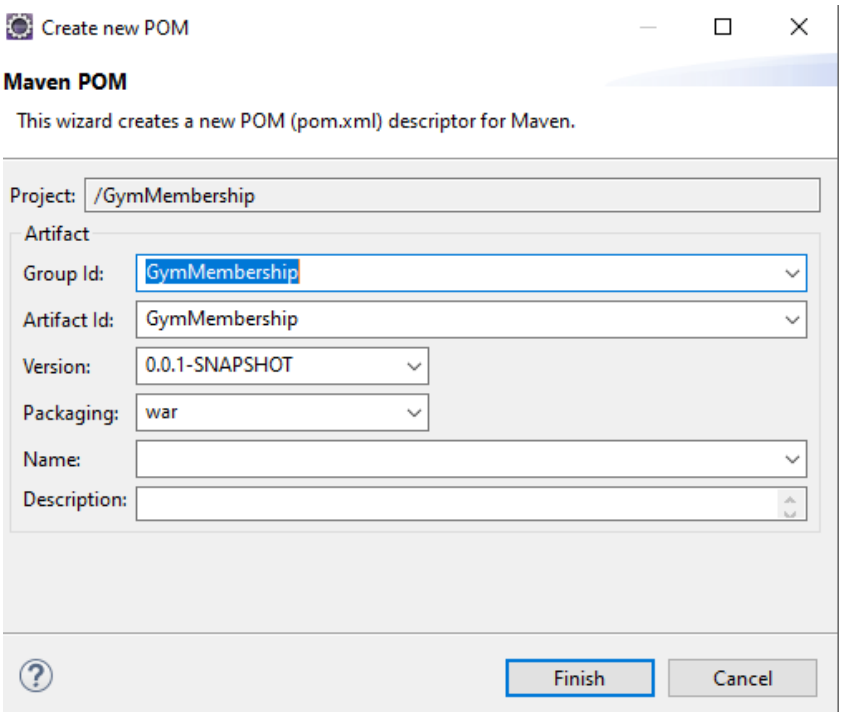
a. Dynamic Web Project



b. Configure to Maven Project

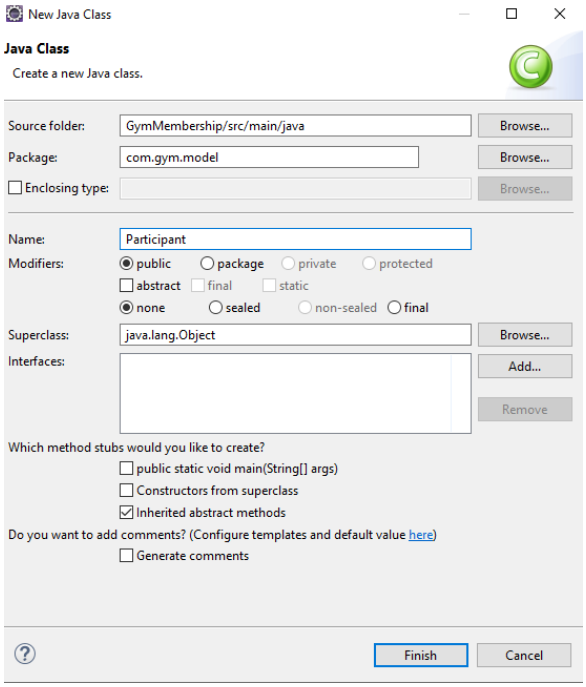


c. Maven POM

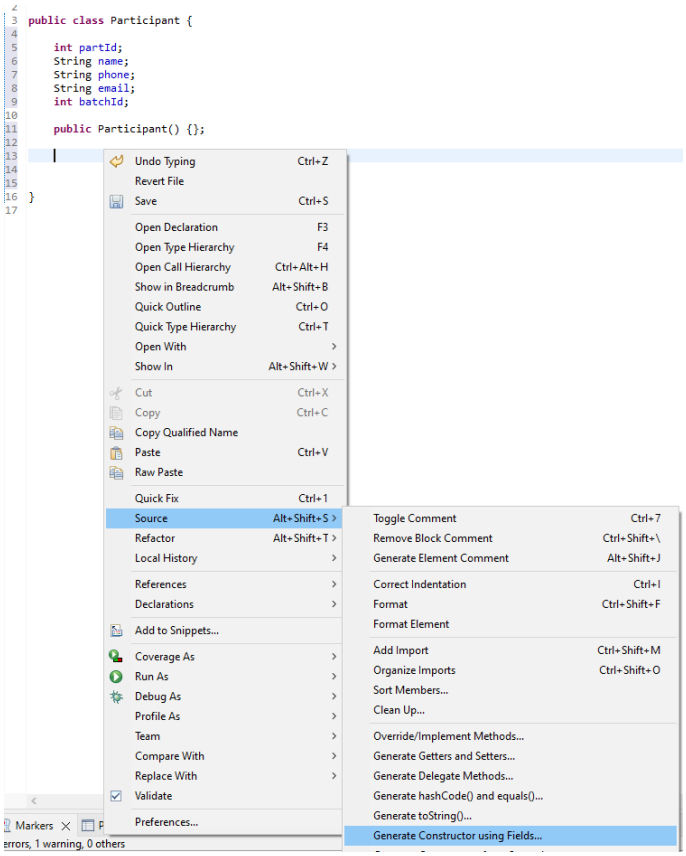


2. Create Java Classes

2.1 Create Participant class



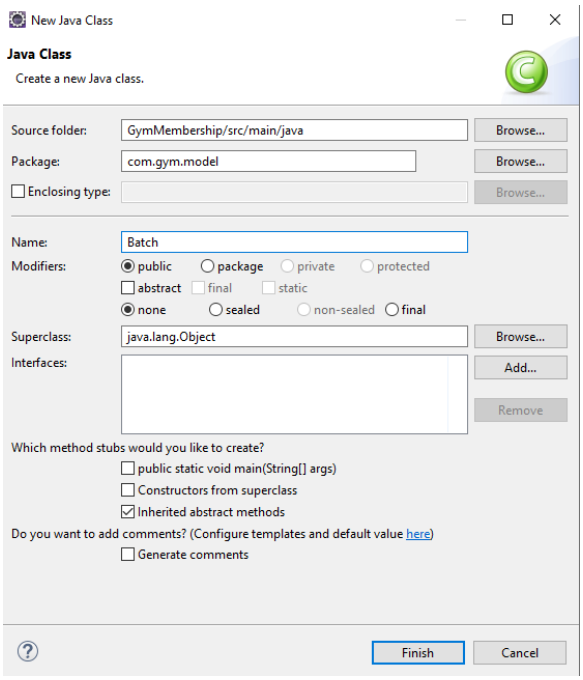
2.1.a. Generate Constructor and toString()



2.1.b Generated

```java
package com.gym.model;

public class Participant {

    int partId;
    String name;
    String phone;
    String email;
    int batchId;

    public Participant() {}

    public Participant(int partId, String name, String phone, String email, int batchId) {
        super();
        this.partId = partId;
        this.name = name;
        this.phone = phone;
        this.email = email;
        this.batchId = batchId;
    }

    @Override
    public String toString() {
        return "Participant [partId=" + partId + ", name=" + name + ", phone=" + phone + ", email=" + email
                + ", batchId=" + batchId + "]";
    };


}
```

2.2 Create Batch class



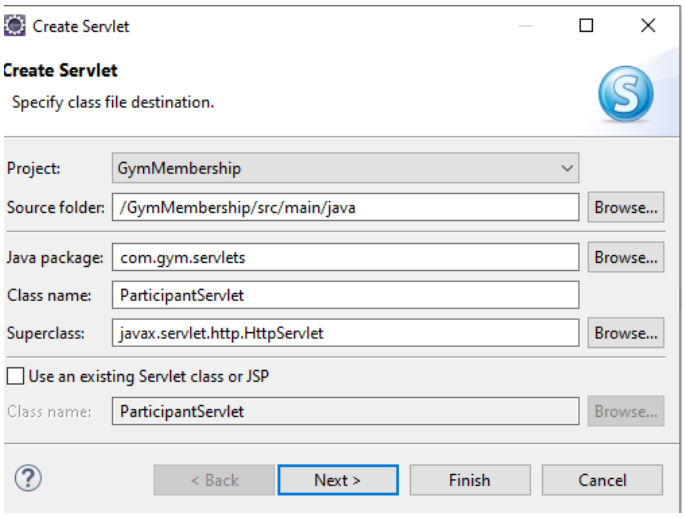
2.2.a. Batch Constructors

```java
package com.gym.model;

public class Batch {

    int bId;
    String batchname;
    String preferredTime;
    String slot;


    public Batch() {};

    public Batch(int bId, String batchname, String preferredTime, String slot) {
        super();
        this.bId = bId;
        this.batchname = batchname;
        this.preferredTime = preferredTime;
        this.slot = slot;
    }
    @Override
    public String toString() {
        return "Batch [bId=" + bId + ", batchname=" + batchname + ", preferredTime=" + preferredTime + ", slot=" + slot
                + "]";
    }


}
```

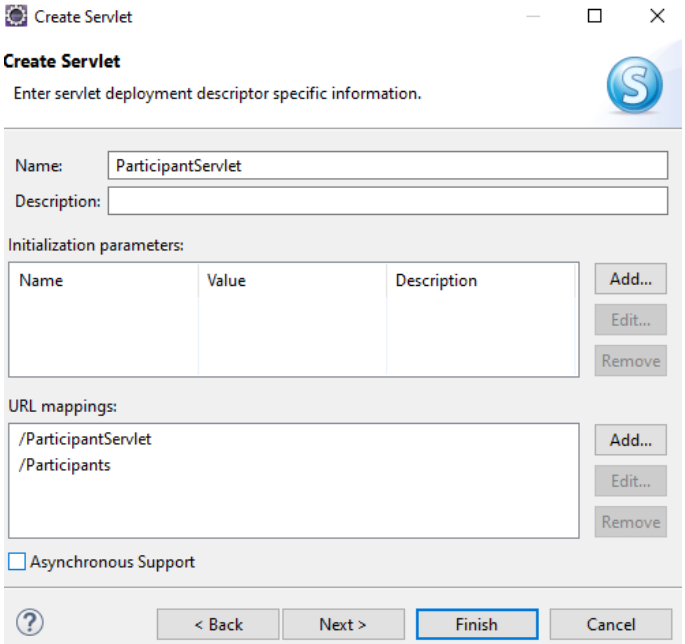
3. Create Servlets in the Project

```
<dependencies>
    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>4.0.1</version>
        <scope>provided</scope>
    </dependency>
    <!-- https://mvnrepository.com/artifact/javax.serv
    <dependency>
        <groupId>javax.servlet.jsp</groupId>
        <artifactId>jsp-api</artifactId>
        <version>2.2.1-b03</version>
        <scope>provided</scope>
    </dependency>
</dependencies>
```

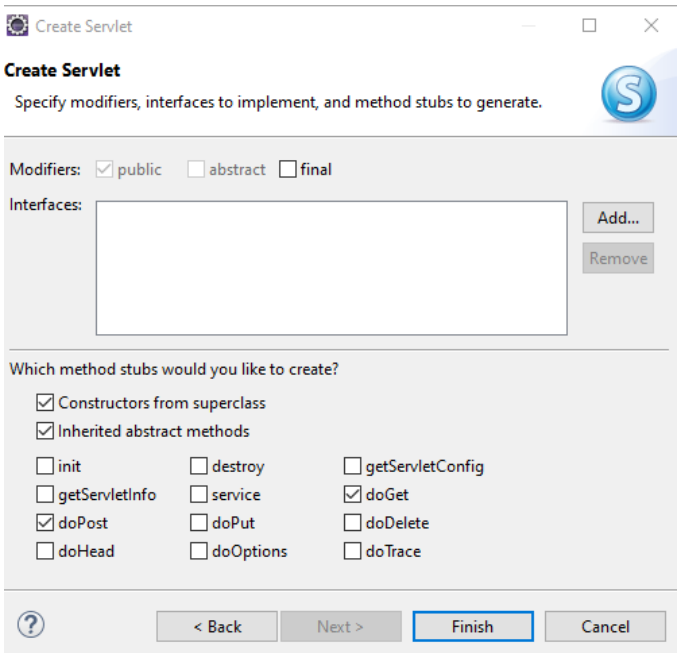
3.1. Create a Servlet for Participant



3.1.a. Create Participant mapping



3.1.b. Select Implementation and methods



3.1.c. Generate Code

```
package com.gym.servlets;

import java.io.IOException;

/**
 * Servlet implementation class ParticipantServlet
 */
@WebServlet({ "/ParticipantServlet", "/Participants" })
public class ParticipantServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ParticipantServlet() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletExceptic
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletExcepti
        // TODO Auto-generated method stub
        doGet(request, response);
    }

}
```

3.2. Create Servlet for Batch

```
package com.gym.servlets;

import java.io.IOException;

/**
 * Servlet implementation class Batch
 */
@WebServlet("/Batch")
public class Batch extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public Batch() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws Servl
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws Serv
        // TODO Auto-generated method stub
        doGet(request, response);
    }

}
```

4. Create HTML Pages

4.1 Create a Welcome Page with Navigation Menu

Welcome to the Gym Membership App

[Add]	Add Participant
[Add]	Add Batch
[Action]	View/Update/Delete Participants
[Action]	View/Update/Delete Batches
[Action]	View/Update/Delete List of Participants query Batches

@ Rogelio Lotho 2022

4.2 Create a HTML page to Add Batch

Batch Entry

Enter Batch Class: ex. Zumba, Circuit Traini

Enter Preferred Times: 07:00

Enter Slot: am or pm

Submit Query

@ Rogelio Lotho 2022

4.3 Create a HTML page to Add Participant

Participant Entry

Enter Name: ex. John Watson

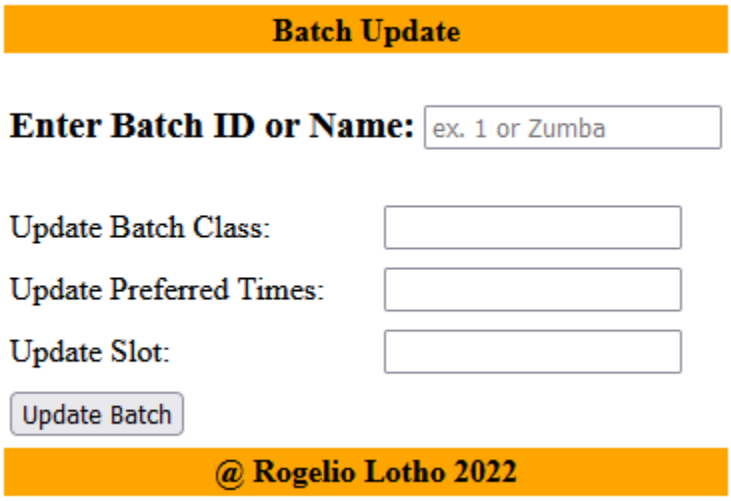
Enter Phone: ex +310 585-5537

Enter Email: ex john@example.com

Add Participant

@ Rogelio Lotho 2022

4.4 Create a HTML page to Update Batch



4.5 Create a HTML page to Update Participant

5. Perform CRUD Operations in JDBCCreate Database and Table in MySQL

5.1.a Mysql Database

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| estore             |
| gymmembership      |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
```

5.1.b Mysql Participant and Batch table

```
mysql> create table Participant (
    ->   pid int primary key auto_increment,
    ->   name varchar(200),
    ->   phone varchar(20),
    ->   email varchar(100),
    ->   bid int);
Query OK, 0 rows affected (0.01 sec)

mysql> describe Participant
    -> ;
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| pid   | int(11)      | NO   | PRI | NULL    | auto_increment |
| name  | varchar(200) | YES  |     | NULL    |                |
| phone | varchar(20)  | YES  |     | NULL    |                |
| email | varchar(100) | YES  |     | NULL    |                |
| bid   | int(11)      | YES  |     | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> describe Batch
    -> ;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| bid           | int(11)      | NO   | PRI | NULL    | auto_increment |
| batchname     | varchar(100) | YES  |     | NULL    |                |
| preferredTime | varchar(10)  | YES  |     | NULL    |                |
| slot          | varchar(5)   | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)

mysql> _
```

5.1 Configure JDBC Dependencies for MySQL

```
<!-- https://mvnrepository.com/artifact/mysql/mysql-connec
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.30</version>
</dependency>
```

5.2 Implement a DAO Design Pattern

```
package com.gym.db;

import java.util.ArrayList;

public interface DAO {

    void createConnection();
    void closeConnection();

    //Declare methods for Batch
    void createBatch(Batch batchclass);
    void updateBatch(Batch batchclass);
    ArrayList<Batch>  getBatch();

    //Declare methods for Participant
    void createParticipant(Participant aparticipant);
    void updateParticipant(Participant aparticipants);
    ArrayList<Participant>  getParticipant();


}
```

5.3 Create a Repository Class which uses DAO to perform DB Interactions

```java
package com.gym.db;

import java.sql.Connection;

public class DB implements DAO {

    Connection conn;
    Statement stmt;
    final String TAG = "["+getClass().getSimpleName()+"] ";
    public DB() {

try {

            Class.forName("com.mysql.cj.jdbc.Driver");

        } catch (ClassNotFoundException e) {
            System.out.println("[DB] MySQL Driver not found! "+e);
        }

        System.out.println("MySQL JDBC Driver Registgered");

    }



    @Override
    public void createConnection() {
        // TODO Auto-generated method stub
        try {
            String url= "jdbc:mysql://localhost/gymmembership";
            conn = DriverManager.getConnection(url,"admin","root");
            System.out.println("Connection Created");

        } catch (Exception e) {
            // TODO: handle exception
            System.out.println("[DAO] Create Connection Exception Occured "+e);
            }
        }


    @Override
    public void closeConnection() {
```

5.4 In Participant Servlet implement CRUD Operations using doGet, doPost, doDelete and doPut Http Methods

```
mysql> select * from participant;
+-----+-------------+----------------+--------------------+------+
| pid | name        | phone          | email              | bid  |
+-----+-------------+----------------+--------------------+------+
|   1 | Rogelio L   | (310) 568-5537 | lothor@example.com |    1 |
|   2 | John Watson | (568) 321-4393 | john@exampl.ecom   |    4 |
|   3 | Kira K.     | (323) 568-4312 | kira@yahoo.com     |    2 |
+-----+-------------+----------------+--------------------+------+
3 rows in set (0.00 sec)
```

Participants Entry

Enter Name: Kira K.

Enter Phone: (323) 568-4312

Enter Email: kira@yahoo.com

Enter Batch No. to Attend: 2

Add Participant

@ Rogelio Lotho 2022

```
[ParticipantServlet] Initialized
[AddParticipant Servlet doPost executed] Details: null null null batchId: null
Step 1: [DB] Driver Loaded
Step 2: [DB] Connection Created
Step 3: [Participant Servlet] Participant Entry Created Successfuly
Step 4: [DB] Connection Closed. Close Status: true
```

AddParticipant Servlet

Added Participant: Hari Morning (510) 4343-34343 hari@simplilearn.com batchId: 8

Participant [pId=0, name=Hari Morning, phone=(510) 4343-34343, email=hari@simplilearn.com, bId=8]

[AddParticipantServlet] Updated Successful

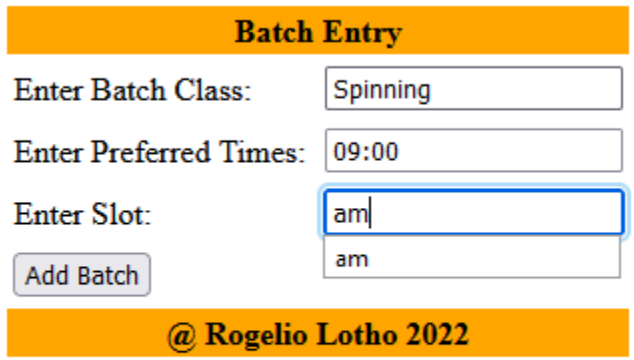
[UpdateParticipantServlet]

5.5 In Batch Servlet implement CRUD Operations using doGet, doPost, doDelete and doPut Http Methods

5.5.a Batch doPost

```
*/
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    //doGet(request, response);
    String batchname = request.getParameter("txtBatchName");
    String preferredTime = request.getParameter("txtPreferredTime");
    String slot = request.getParameter("txtSlot");

    System.out.println("[AddBatch Servlet doPost executed] Details: "+batchname+" "+preferredTime+" "+slot);

    Batch bc = new Batch();
    bc.batchname = request.getParameter("txtBatchName");
    bc.preferredTime = request.getParameter("txtPreferredTime");
    bc.slot = request.getParameter("txtSlot");

    DB db = new DB();
    db.createConnection();
    int result = db.addBatch(bc);

    response.setContentType("text/html");
    String loginTimeStamp = new Date().toString();

    String htmlResponse = "<center>[BatchServlet doPost] Thank you for adding Batch <h3>"+batchname+"</h3> Preferred Time: <h3>"
    PrintWriter pw = response.getWriter();

    String message = (result >= 0 ) ? "Batch Entry Created Successfuly": "[BatchServlet doPost] Insert failed error";

    pw.println("Step3: [Batch Servlet] "+message+" result: "+result);
    System.out.println("Step3: [Batch Servlet] "+message);
    pw.println(htmlResponse);
```

5.5.b Run Batch doPost




```
[BatchServlet] Initialized
[AddBatch Servlet doGet executed] Details: Spinning 09:00 pm
[AddBatch Servlet doPost executed] Details: Spinning 09:00 pm
Step 1: [DB] Driver Loaded
Step 2: [DB] Connection Created
Step 3 [DAO] Append to Batch table Successful
[DB] Connection Closed. Close Status: true
```

```
mysql> select * from batch;
+-----+----------+---------------+------+
| bid | batchname | preferredTime | slot |
+-----+----------+---------------+------+
|   1 | batchname | preferredTime | slot |
|   2 | Karate    | 07:00         | am   |
|   3 | Spinning  | 09:00         | pm   |
+-----+----------+---------------+------+
```

6 Create JSP Pages in the Project

6.1 Create a JSP Page to view the list of Participants with delete option

List of Participants

ID	Name	Phone	Email	BatchID	Class	Update/Delete
6	John Watson	(543) 545-454	john@example.com	4		Update | Delete
7	Kim Ducacy	(310) 533-4343	kimd@yahoo.com	6		Update | Delete
8	Rogelio Lotho	(505) 394-3434	rlotho@yahoo.com	10		Update | Delete
9	Kartina Mak	(213) 434-34343	kartinak@hotmial.com	3		Update | Delete
11	Travi Kubusinki	(504) 89-4343	traviw@gmail.com	5		Update | Delete
12	Anna Poli	(433) 343-34343	anna@example.com	3		Update | Delete

[UpdateParticipant JSP]

```
        </tr>
    <%

    String id = request.getParameter("id");

    Class.forName("com.mysql.jdbc.Driver");
    String url = "jdbc:mysql://localhost/gymmembership";
    Connection conn = DriverManager.getConnection(url, "admin", "root");
    System.out.println("[Update Participant JSP] Connection Created");

    Statement stmt = conn.createStatement();
    String query = "select * from Participant where pid ="+id;
    ResultSet rs=stmt.executeQuery(query);
while(rs.next())
{


        <form action="UpdateParticipant" method="Post">
        <input type="hidden" name="txtPid" value=<%=rs.getInt("pId")%> />
    <tr>
            <td>update Name:</td>
            <td><input type="text" name="txtName"  value="<%=rs.getString("name")%>" /></td>
        </tr>
        <tr>
            <td>update Phone:</td>
            <td><input type="text" name="txtPhone" value="<%=rs.getString("phone")%>" /></td>
        </tr>
        <tr>
            <td>update Email:</td>
            <td><input type="text" name="txtEmail" value="<%=rs.getString("email")%>" /></td>
        </tr>
        <tr>
            <td>update BatchId:</td>
            <td><input type="text" name="txtBid" value="<%=rs.getString("bId")%>" /></td>
        </tr>
        <tr>
            <td colspan="2"><input type="submit" value="Update Participant"></td>
        </tr>
        </form>
```

[UpdateBatch Servlet]

Update Batch Servlet

Update id = 9 Details: Circuit Training II 10:35 am

Batch [bId=0, batchname=Circuit Training II, preferredTime=10:35, slot=am]

[UpdateBatchServlet] Updated Successful

[UpdateParticipant Servlet]

```
package com.gym.servlets;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

import com.gym.db.DB;
import com.gym.model.Participant;


/**
 * Servlet implementation class UpdateParticipant
 */
public class UpdateParticipant extends HttpServlet {
    private static final long serialVersionUID = 1L;


    /**
     * @see HttpServlet#doPut(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
        Participant pct = new Participant();
        int pid = Integer.parseInt(request.getParameter("txtPid"));
        String name = request.getParameter("txtName");
        String phone = request.getParameter("txtPhone");
        String email = request.getParameter("txtEmail");
        String bid = request.getParameter("txtBid");
        System.out.println("[UpdateParticipant Servlet doPost executed] Details: "+name+" "+phone+" "+email+" batchId: "+bid);

        pct.name=name;
        pct.phone=phone;
        pct.email=email;
        pct.bId=Integer.parseInt(bid);
        pct.pId=pid;


        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();
        pw.print("<center><h3>Update Participant Servlet</h3>");
```

[DeleteParticipant Servlet]

```java
 */
public class DeleteParticipant extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
     */
    protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub

        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();

        pw.print("<center>");

        int id = Integer.parseInt(request.getParameter("id"));

        DB db = new DB();
        db.createConnection();
        int result = db.deleteParticipant(id);

        if (result>=0) {
            pw.print("<h3>[DeleteServlet] Participant Deleted Successfully</h3>");
        } else {
            pw.print("<h3>[DeleteServlet] Participant was not deleted!</h3>");
        }
        pw.print("</center>");
```

6.2

```
    <table>
        <thead>
            <tr>
                <th>ID</th>
                <th>Name</th>
                <th>Phone</th>
                <th>Email</th>
                <th>BatchID Class</th>
                <th>Update/Delete</th>
            </tr>
        </thead>
        <tbody>
        <%

        DB db = new DB();

        Class.forName("com.mysql.jdbc.Driver");
        String url = "jdbc:mysql://localhost/gymmembership";
        Connection conn = DriverManager.getConnection(url, "admin", "root");
        System.out.println("Step 2: [DB] Connection Created");

        Statement stmt = conn.createStatement();
        String query = "select * from Participant";
        ResultSet rs=stmt.executeQuery(query);
    while(rs.next())
    {

%>

                <tr>
                    <td><%=rs.getInt("pId")%></td>
                    <td><%=rs.getString("name")%></td>
                    <td><%=rs.getString("phone")%></td>
                    <td><%=rs.getString("email")%></td>
                    <td><%=rs.getInt("bId")%></td>
                    <td><a href="Update?id=<%=rs.getInt("pId")%>">Update</a>
                        | <a href="Delete?id=<%=rs.getInt("pId")%>">Delete</a></td>
                </tr>
            <% } %>
                </tbody>
    </table>
</body>
</html>
```

localhost:9090/gymmembership/Delete?id=1

I) Doma... We're sorry, but somet...

[DeleteServlet] Participant Deleted Successfully

```
mysql> select * from participant;
+-----+-------------+----------------+--------------------+------+
| pid | name        | phone          | email              | bid  |
+-----+-------------+----------------+--------------------+------+
|   1 | Rogelio L   | (310) 568-5537 | lothor@example.com |    1 |
|   2 | John Watson | (568) 321-4393 | john@exampl.ecom   |    4 |
|   3 | Kira K.     | (323) 568-4312 | kira@yahoo.com     |    2 |
|   4 | Jim Beam    | (474) 394-3433 | jimb@gmail.com     |    2 |
+-----+-------------+----------------+--------------------+------+
4 rows in set (0.00 sec)

mysql> select * from participant;
+-----+-------------+----------------+------------------+------+
| pid | name        | phone          | email            | bid  |
+-----+-------------+----------------+------------------+------+
|   2 | John Watson | (568) 321-4393 | john@exampl.ecom |    4 |
|   3 | Kira K.     | (323) 568-4312 | kira@yahoo.com   |    2 |
|   4 | Jim Beam    | (474) 394-3433 | jimb@gmail.com   |    2 |
+-----+-------------+----------------+------------------+------+
3 rows in set (0.00 sec)
```

6.3 Create a JSP page to view the list of Batches with delete option

List of Batch Classes

Batch ID	Class Name	Preferred Time	Slot	Update/Delete
2	Karate	07:00	am	Update | Delete
3	Spinning	09:00	pm	Update | Delete
5	Spinning Beg.	11:00	am	Update | Delete
6	Spinning Adv	12:30	pm	Update | Delete
7	Circuit Training	10:30	pm	Update | Delete
9	Circuit Training II	10:30	am	Update | Delete
10	JuiJitsu	6:30	pm	Update | Delete
11	Dance II	10:30	am	Update | Delete

6.4 Create a JSP Page to view the lists of Particpants in a Batch using query parameter

List of Participants

ID	Name	Phone	Email	BatchID	Class Name	Preferred Time	Slot	Update/Delete
7	Kim Ducacy	(310) 533-4343	kimd@yahoo.com	6	Spinning Adv	12:30	pm	Update | Delete
8	Rogelio Lotho	(505) 394-3434	rlotho@yahoo.com	10	JuiJitsu	6:30	pm	Update | Delete
9	Kartina Mak	(213) 434-34343	kartinak@hotmial.com	3	Spinning	09:00	pm	Update | Delete
11	Travi Kubusinki	(504) 89-4343	traviw@gmail.com	5	Spinning Beg.	11:00	am	Update | Delete
12	Anna Poli	(433) 343-34343	anna@example.com	3	Spinning	09:00	pm	Update | Delete

7 Build and Run the Project on Apache Tomcat webserver

```
45            int result = db.addBatch(bc);
46
47            response.setContentType("text/html");
48            String loginTimeStamp = new Date().toString();
49
```

Markers | Properties | Servers | Data Source Explorer | Snippets | Terminal | Console

Tomcat v10.0 Server at localhost [Started, Synchronized]

gymmembership(gymmembership-0.0.1-SNAPSHOT) [Synchronized]

8 Validate the working of project

```
[AddParticipantServlet] Initialized
[AddParticipant Servlet] Details: Hari Morning (510) 4343-34343 hari@simplilearn.com batchId: 8
Step 1: [DB] Driver Loaded
Step 2: [DB] Connection Created
Step 4: [DB] Connection Closed. Close Status: true
Step 1: [DB] Driver Loaded
[ViewParticipant JSP] Connection Created
Step 1: [DB] Driver Loaded
[ViewBatch JSP] Connection Created
[UpdateBatch JSP] Connection Created
Update Batch SErvelet serated
[UpdateBatchServlet] Details: Circuit Training II 10:35 am
Step 1: [DB] Driver Loaded
Step 2: [DB] Connection Created
Step 3: [DB] Update Batch executed
Step 4: [DB] Connection Closed. Close Status: true
[ViewParticipant JSP] Connection Created
[Update Participant JSP] Connection Created
```

9 Package the project as a Jar file using maven Package Goal

```
INFO] --------------------< gymmembership:gymmembership >---------------------
INFO] Building gymmembership 0.0.1-SNAPSHOT
INFO] --------------------------------[ war ]---------------------------------
INFO]
INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ gymmembership ---
INFO] Deleting C:\Users\336536\Desktop\caltech\Course2-Java\SkillEndProjects-Java\Course2Project\gymmembership\target
INFO] ------------------------------------------------------------------------
INFO] BUILD SUCCESS
INFO] ------------------------------------------------------------------------
INFO] Total time:  0.220 s
INFO] Finished at: 2022-08-02T22:58:30-06:00
INFO] ------------------------------------------------------------------------
```