

***Java Technologies for Web Applications***

**Lab Guides**

|  |  |
| --- | --- |
| Document Code | 25e-BM/HR/HDCV/FSOFT |
| Version | 1.1 |
| Effective Date | 20/11/2012 |

**Hanoi, 04/2019**

RECORD OF CHANGES

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Effective Date | Change Description | Reason | Reviewer | Approver |
|  | 25/Jun/2018 | Create a new Lab | Create new | DieuNT1 | VinhNV |
|  | 01/May/2019 | Update Fsoft Template | Update | DieuNT1 | VinhNV |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Contents

[Unit 2 - MVC Model and Session Tracking 4](#_Toc25869570)

|  |  |
| --- | --- |
|  | **CODE: NWEB.M.L201**  **TYPE: Medium**  **LOC:**  **DURATION: 60 MINUTES** |

# Unit 2 - MVC Model and Session Tracking

**Objectives**:

* Understand the basic concepts of web development technologies with java (JSP / Servlet)
* Able to write servlets using the Java programming language (Java servlets)
* Create dynamic HTML content with Servlets and JavaServer Pages, using the Expression Language, and the JSP Standard Tag Library (JSTL)
* Create robust web applications using MVC architecture, session management, filters, and database integration (JDBC)
* Make Servlets and JSP work together cleanly
* Create secure web applications using the features of the Java EE web container

**Problem Descriptions**:

**Step1: Create the following web pages**

**Link Bootstrap 4:**

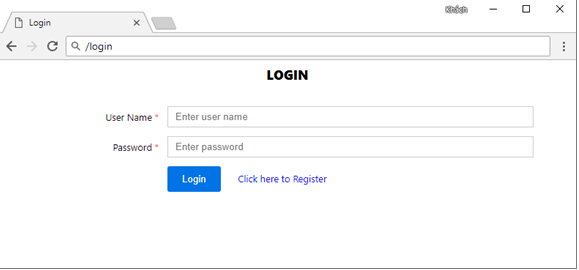
<link rel=*"stylesheet"*

href=*"https://maxcdn.bootstrapcdn.com/bootstrap/4.1.0/css/bootstrap.min.css"*>

**Link Font Awesome:**

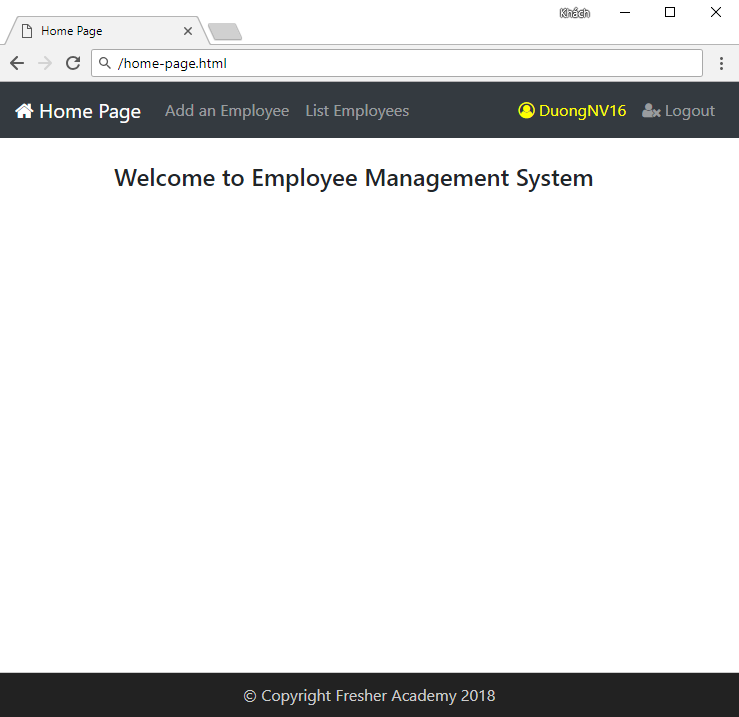
<link rel=*"stylesheet"* href=*"https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css"*>

* *login.jsp screen:*

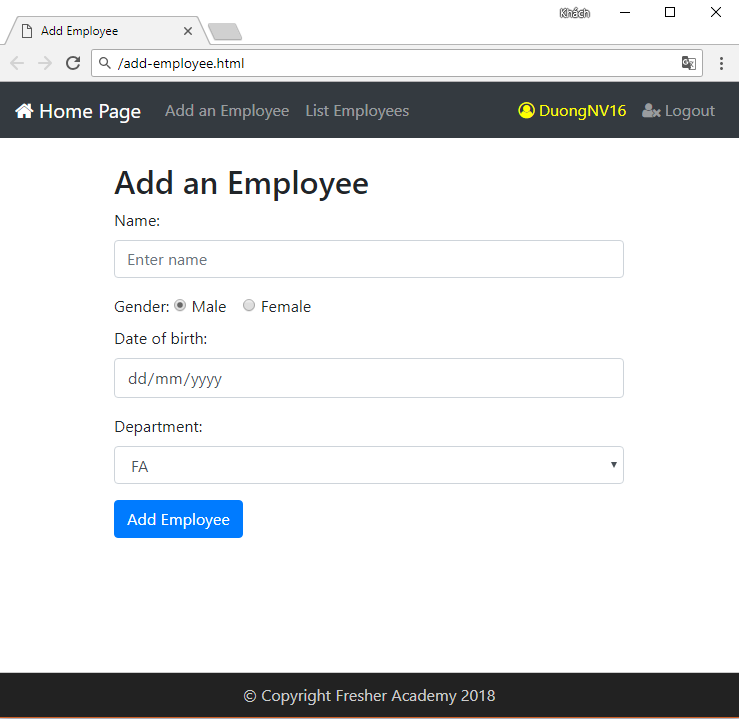


*Screen 01\_Layout 01*

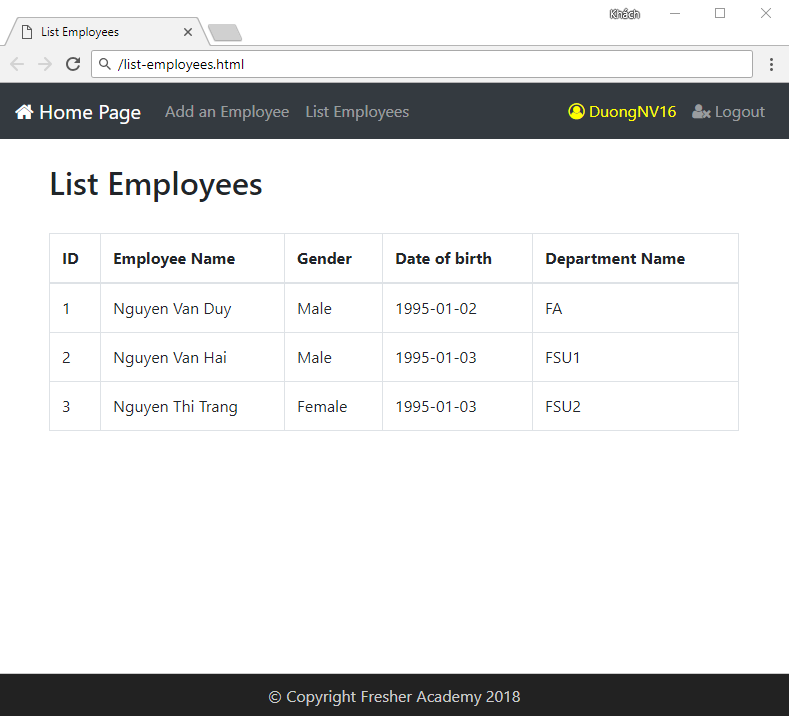
* *home-page.jsp screen:*



* *add-employee.jsp screen:*

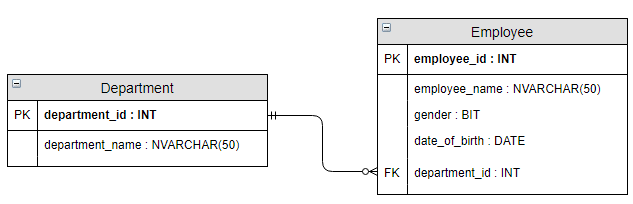


* *list-employees.jsp screen:*



**Step2: Create Database**

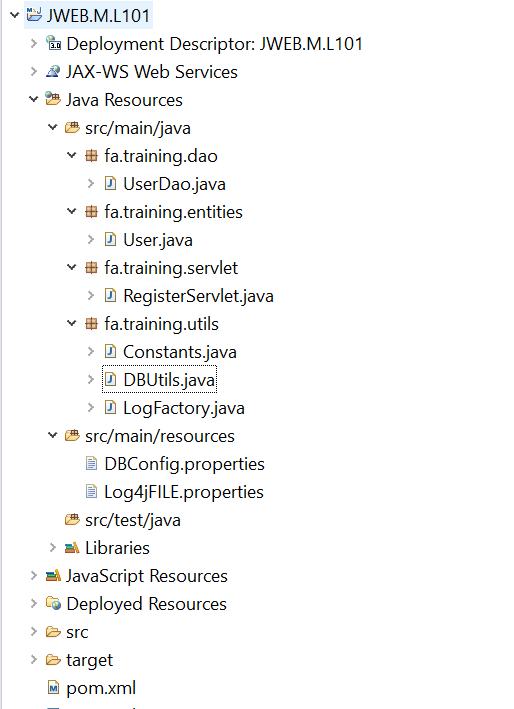
Create a database named “**JNWEBML201\_SMS**” có các bảng và quan hệ như sau:



Tạo stored procedure “**usp\_registerUser**” như sau:

1. CREATE PROC [dbo].[usp\_registerUser]
2. @firstName VARCHAR(50),
3. @lastName VARCHAR(50),
4. @email VARCHAR(100),
5. @userName VARCHAR(50),
6. @password VARCHAR(50)
7. AS
8. BEGIN
9. INSERT INTO Users VALUES (@firstName, @lastName, @email, @userName, @password)
10. END

**Step3: Create a maven project named “JWEB.M.L201”**:

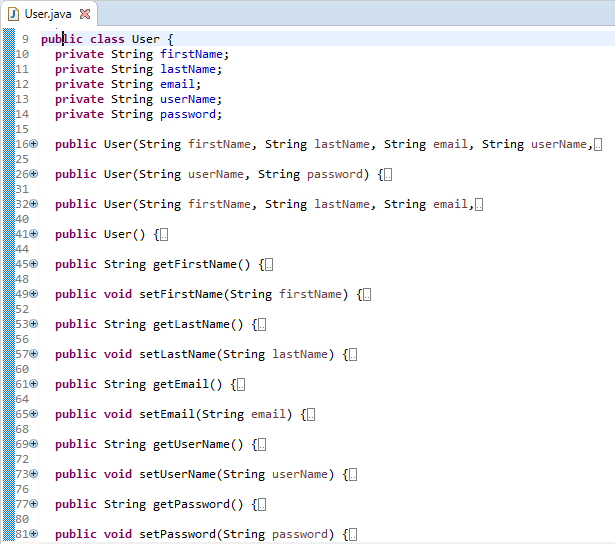


File **pom.xml**

1. <dependencies>
2. <dependency>
3. <groupId>javax.servlet</groupId>
4. <artifactId>javax.servlet-api</artifactId>
5. <version>3.1.0</version>
6. </dependency>
7. <dependency>
8. <groupId>com.microsoft.sqlserver</groupId>
9. <artifactId>mssql-jdbc</artifactId>
10. <version>7.0.0.jre8</version>
11. </dependency>
12. <dependency>
13. <groupId>log4j</groupId>
14. <artifactId>log4j</artifactId>
15. <version>1.2.17</version>
16. </dependency>
17. <dependency>
18. <groupId>javax.servlet</groupId>
19. <artifactId>jstl</artifactId>
20. <version>1.2</version>
21. </dependency>
22. </dependencies>
23. <build>
24. <finalName>JavaWeb\_P\_L002</finalName>
25. <plugins>
26. <plugin>
27. <groupId>org.apache.maven.plugins</groupId>
28. <artifactId>maven-compiler-plugin</artifactId>
29. <version>3.7.0</version>
30. <configuration>
31. <source>1.8</source>
32. <target>1.8</target>
33. </configuration>
34. </plugin>
35. <plugin>
36. <groupId>org.apache.maven.plugins</groupId>
37. <artifactId>maven-war-plugin</artifactId>
38. <version>3.2.2</version>
39. <configuration>
40. <warSourceDirectory>src/main/webapp
41. </warSourceDirectory>
42. <failOnMissingWebXml>
43. false
44. </failOnMissingWebXml>
45. </configuration>
46. </plugin>
47. </plugins>
48. </build>

**Step4: Create servlet class and config**

Create **User** class in package **fa.training.entities**:



Create **LoginServlet** class in package **fa.training.servlet** to handle login the following:

1. @WebServlet(urlPatterns = "/login")
2. **public** **class** LoginServlet **extends** HttpServlet {
3. private static final long *serialVersionUID* = 1L;
4. **private** **static** UserDao *userDao* = **new** UserDao();
5. @Override
6. **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response)
7. **throws** ServletException, IOException {
8. Log4J.*getLogger*().info("Running on doPost method of LoginServlet");
9. String userName = request.getParameter("userName");
10. String password = request.getParameter("password");
11. User user = **new** User(userName, password);
12. **try** {
13. **if** (*userDao*.login(user)) {
14. HttpSession session = request.getSession();
15. // if login successfully, save session user, who have just logined
16. session.setAttribute("userLogin", user);
17. response.sendRedirect(request.getContextPath() + "/home");
18. } **else** {
19. request.setAttribute("userRegister", user);
20. request.setAttribute("loginFail", "User name or password is incorrect");
21. request.getRequestDispatcher("/views/login.jsp").forward(request, response);
22. }
23. } **catch** (ClassNotFoundException e) {
24. Log4J.*getLogger*().
25. error("Class not found exception in method doPost of LoginServlet");
26. } **catch** (SQLException e) {
27. Log4J.*getLogger*().error("SQL exception in method doPost of LoginServlet");
28. }
29. }
30. @Override
31. **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response)
32. **throws** ServletException, IOException {
33. Log4J.*getLogger*().info("Running on doGet method of LoginServlet");
34. request.getRequestDispatcher("views/login.jsp").forward(request, response);
35. }
36. }

**Step4.** If the login is successful, it will save the logged-in session user, and redirect to request **/home** to display the home-page.

1. @WebServlet("/home")
2. **public** **class** HomePageServlet **extends** HttpServlet {
4. private static final long *serialVersionUID* = 1L;
5. @Override
6. **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response)
7. **throws** ServletException, IOException {
8. request.getRequestDispatcher("/views/home-page.jsp").forward(request, response);
9. }
10. }

* Tạo một Servlet có tên **RegisterServlet** trong package **fa.training.servlet** và override phương thức **doPost()** như sau:

1. @WebServlet(urlPatterns = "/register")
2. **public** **class** RegisterServlet **extends** HttpServlet {
3. **private** **static** **final** **long** ***serialVersionUID*** = 1L;
4. @Override
5. **protected** **void** doPost(HttpServletRequest request,
6. HttpServletResponse response) **throws** ServletException, IOException {
7. // Get data from the request using request.getParameter()
8. String firstName = request.getParameter("firstName");
9. String lastName = request.getParameter("lastName");
10. String email = request.getParameter("email");
11. String userName = request.getParameter("userName");
12. String password = request.getParameter("password");
14. // Set data for the user
15. User user = **new** User(firstName, lastName, email, userName, password);
16. **try** {
17. UserDao userDao = **new** UserDao();
18. // Call registerUser() method to insert user into DB
19. **if** (userDao.registerUser(user)) {
20. // Send a attribute name as "userRegister"
21. to register-user-process.jsp page
22. request.setAttribute("userRegister", user);
23. // Forward to register-user-process.jsp page
24. request.getRequestDispatcher("/views/login.jsp").
25. forward(request, response);
26. } **else** {
27. // send a attribute name as "message" to register-user.jsp page
28. request.setAttribute("message", Constants.***REGISTER\_FAIL\_MESSAGE***);
29. // forward to register-user.jsp page
30. request.getRequestDispatcher("/views/register-user.jsp").
31. forward(request, response);
32. }
33. } **catch** (ClassNotFoundException | SQLException e) {
34. // log error if exception occurs
35. LogFactory.*getLogger*().
36. error("An exception occurs while register user");
37. }
38. }
39. }

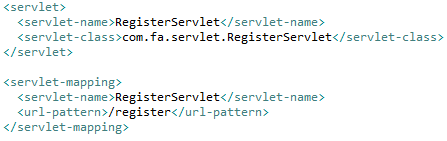
* **Cấu hình servlet để mapping với request, t**hay đổi **action** của form register thành:

<form action=*"*<%=request.getContextPath()%>*/register"* method=*"post"*

onsubmit="return validateRegister()" name=*"frm-register"*>

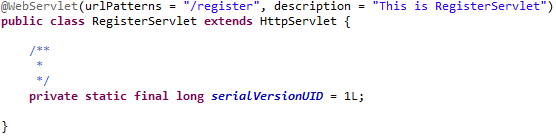
Có 2 cách cấu hình servlet để mapping với request tương ứng:

**Cách 1**: Cấu hình trong file **web.xml bằng cách thêm** đoạn code sau vào file **web.xml**



**Cách 2:** Sử dụng Annotation **@WebServlet**

Thêm Annotation **@WebServlet** trước class **RegisterServlet**



**Step5: Xử lý DAO**

Tạo method **registerUser(User user)** trong class **UserDao** để xử lý insert user vừa đăng ký vào bảng **Users** trong database.

1. package fa.training.dao;
2. import java.io.IOException;
3. import java.sql.CallableStatement;
4. import java.sql.Connection;
5. import java.sql.ResultSet;
6. import java.sql.SQLException;
7. import fa.training.entities.User;
8. import fa.training.utils.DBUtils;
9. /\*\*
10. \* The class contains methods to update and retrieve data from database
11. \*
12. \* @author FA
13. \*
14. \*/
15. public class **UserDao** {
16. /\*\*
17. \* The method to insert a new user into database.
18. \*
19. \* @param user an user object.
20. \* @return true if register successfully.
21. \* @throws SQLException
22. \* @throws IOException
23. \* @throws ClassNotFoundException
24. \*/
25. public boolean **registerUser**(User user)
26. throws ClassNotFoundException, IOException, SQLException {
27. Connection connection = null;
28. try {
29. connection = DBUtils.getConnection();
30. CallableStatement callableStatement =
31. connection.prepareCall("{call **usp\_registerUser**(?,?,?,?,?)}");
32. int param = 0;
33. callableStatement.setString(++param, user.getFirstName());
34. callableStatement.setString(++param, user.getLastName());
35. callableStatement.setString(++param, user.getEmail());
36. callableStatement.setString(++param, user.getUserName());
37. callableStatement.setString(++param, user.getPassword());
38. int result = callableStatement.executeUpdate();
39. if (result > 0) {
40. return true;
41. }
42. return false;
43. } finally {
44. DBUtils.closeConnection(connection);
45. }
46. }
47. }

**-- THE END --**