**PHASE3-FIRSTPROJECT-WRITEUP**

**Searching for a Specific User and Updating the User Information.**

**Project objective:**

Create a Spring MVC web application that will retrieve users based on their user ID. The retrieved user data will then be edited in a form and updated in the database. The entire database processing has to be done using Hibernate. Front pages will be made in JSP.

**Background of the problem statement:**

As a part of developing an ecommerce web application, the admin backend requires a module that can retrieve users based on their user ID and update their information as required.

**You must use the following:**

● Eclipse as the IDE  
● Apache Tomcat as the web server  
● Spring MVC with Hibernate, log4j, and MySQL Connector

**Following requirements should be met:**

● Create a JSP page to take in a user ID  
● Create a controller that will validate the user ID. If it is invalid, it will display a JSP page with an error message. If it is valid, it will retrieve user details from the database and show an edit form in JSP  
● Once the edit form is submitted, the controller will update the details in the database. A confirmation JSP page will be shown  
● Follow the standard methodology of creating controllers, services, dao, and entity classes

This project has four subsections, namely:

1. Creating a Maven Project which is web enabled
2. Creating pom.xml for including the required components
3. Building the project
4. Pushing the code to your GitHub repositories

**1. Creating a Maven Project which is web enabled**

* Open Eclipse
* Go to the **File** menu. Choose **New->Maven Project**
* Uncheck **Create a Simple Project** and check **Use Default Workspace Location** and click on **Next**
* From the **archetype** list, choose the row that has **Artifact Id** as **maven-archetype-webapp** and click on **Next**
* Enter **Group Id** as **SpringmvcandHibernate** and **Artifact Id** as **User\_Manager** and click on **Finish**
* This will create the project files in the Project Explorer
* Before building the project, we need to confirm that the **servlet.jar** has been added to the project
* In the Project Explorer, right click and choose **Properties**
* Select **Java Build Path** from the options on the left
* Click on **Libraries** tab on the right
* If there is an existing entry for the **servlet.jar,** then click on **Cancel** and exit the window
* If it is not there, then click on **Classpath** entry and click on **Add External JARs** button on the right
* From the **file** list, select **servlet.jar** file and click on **Ok**
* Click on **Apply and Close**

**App.java**

**package** com.UserManager;

**import** org.springframework.context.annotation.ComponentScan;

**import** org.springframework.context.annotation.Configuration;

@Configuration

@ComponentScan(basePackages = {"com.UserManager"})

**public** **class** App {

}

**Initializer.java**

**package** com.UserManager;

**import** org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

**public** **class** Initializer **extends** AbstractAnnotationConfigDispatcherServletInitializer {

@Override

**protected** Class<?>[] getRootConfigClasses() {

// **TODO** Auto-generated method stub

**return** **null**;

}

@Override

**protected** Class<?>[] getServletConfigClasses() {

// **TODO** Auto-generated method stub

**return** **null**;

}

@Override

**protected** String[] getServletMappings() {

// **TODO** Auto-generated method stub

**return** **null**;

}

**MainController.java**

**package** com.UserManager;

**import** java.util.ArrayList;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.ModelMap;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RequestParam;

@Controller

**public** **class** MainController {

@Autowired

**private** Userdao userdao;

Logger logger = LoggerFactory.*getLogger*(MainController.**class**);

String currID = **null**;

@RequestMapping(value="/")

**public** String showIndexPage(ModelMap model,

@RequestParam(value="name", required=**false**, defaultValue="World") String name){

model.addAttribute("name", name);

**return** "index";

}

**public** **boolean** isNumber(String s)

{

**if**(s == **null**)

**return** **false**;

**try**

{

**double** db = Double.*parseDouble*(s);

}

**catch**(NumberFormatException e)

{

**return** **false**;

}

**return** **true**;

}

@RequestMapping("/update")

**public** String saveDetails(@RequestParam("id") String id, ModelMap modelMap) {

**try**

{

User user = userdao.GetUserById(Integer.*valueOf*(id));

ArrayList<User> userList = **new** ArrayList<>();

**if**(user != **null**)

{

userList.add(user);

Iterable<User> users = userList;

currID = id;

modelMap.put("user", users);

}

**else**

**return** "nouser";

}

**catch** (NumberFormatException e)

{

// **TODO** Auto-generated catch block

**return** "nouser";

}

**catch** (Exception e)

{

// **TODO** Auto-generated catch block

e.printStackTrace();

}

modelMap.put("ID", id);

**return** "update";

}

@RequestMapping("/update2")

**public** String updateDetails(@RequestParam("nameedit") String nameedit, @RequestParam("emailedit") String emailedit, @RequestParam("passwordedit") String passwordedit, ModelMap modelMap) {

ArrayList<User> userList = **new** ArrayList<>();

**try**

{

User u = userdao.GetUserById(Integer.*valueOf*(currID));

userdao.setUser(u, nameedit, emailedit, passwordedit);

userList.add(u);

Iterable<User> users = userList;

modelMap.put("user", users);

}

**catch** (NumberFormatException e)

{

e.printStackTrace();

}

**catch**(Exception e)

{

e.printStackTrace();

}

modelMap.put("IDedit", currID);

**return** "update2";

}

}

**UserController**

**package** com.UserManager;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.ModelMap;

**import** org.springframework.web.bind.annotation.RequestMapping;

@Controller

**public** **class** UserController {

@Autowired

**private** Userdao userdao;

Logger logger = LoggerFactory.*getLogger*(UserController.**class**);

@RequestMapping("/users")

**public** String showUsers(ModelMap model) {

logger.info("Getting all Users");

Iterable<User> users = userdao.GetAllUsers();

logger.info("Passing users to view");

model.addAttribute("users", users);

**return** "users";

}

}

**UserExceptionController**

**package** com.UserManager;

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.web.bind.annotation.ControllerAdvice;

**import** org.springframework.web.bind.annotation.ExceptionHandler;

@ControllerAdvice

**public** **class** UserExceptionController {

@ExceptionHandler(value = UserNotFoundException.**class**)

**public** ResponseEntity<Object> exception(UserNotFoundException exception) {

**return** **new** ResponseEntity<>("User not found", HttpStatus.***NOT\_FOUND***);

}

}

**User.java**

**package** com.UserManager;

**import** javax.persistence.Entity;

**import** javax.persistence.GeneratedValue;

**import** javax.persistence.GenerationType;

**import** javax.persistence.Id;

@Entity // This tells Hibernate to make a table out of this class

**public** **class** User {

@Id

@GeneratedValue(strategy=GenerationType.***AUTO***)

**private** Integer id;

**private** String name;

**private** String email;

**private** String password;

**public** Integer getId() {

**return** id;

}

**public** **void** setId(Integer id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public** String getPassword() {

**return** password;

}

**public** **void** setPassword(String password) {

**this**.password = password;

}

@Override

**public** String toString() {

**return** (id.toString() + " " + name + " " + email + " " + password);

}

}

**Userdao.java**

**package** com.UserManager;

**import** java.util.Optional;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Service;

@Service

**public** **class** Userdao {

@Autowired

**private** UserRepository userRepository;

**public** Iterable<User> GetAllUsers()

{

**return** userRepository.findAll();

}

**public** User GetUserByName(String name) {

User foundUser = userRepository.findByName(name);

**return** foundUser;

}

**public** User GetUserById(**int** id) **throws** Exception {

Optional<User> foundUser = userRepository.findById(id);

//**TODO**: we need to decide how to handle a "Not Found" condition

**if**(!foundUser.isPresent())

**return** **null**;

**return**(foundUser.get());

}

**public** **void** UpdateUser(User usertoUpdate) {

userRepository.save(usertoUpdate);

}

**public** **void** setUser(User u, String name, String email, String password) {

//u.setId(id);

u.setName(name);

u.setEmail(email);

u.setPassword(password);

UpdateUser(u);

}

}

**UserRepository**

**package** com.UserManager;

**import** org.springframework.data.repository.CrudRepository;

**public** **interface** UserRepository **extends** CrudRepository<User, Integer> {

**public** User findByName(String name);

}

**UserNotFoundException**

**package** com.UserManager;

**public** **class** UserNotFoundException **extends** RuntimeException {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

}

**Spring.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns = *"http://www.springframework.org/schema/beans"*

xmlns:xsi = *"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation = *"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-3.0.xsd"*>

<!--create an object for user , SI for the data-->

<bean id=*"userobj"* class=*"com.UserManager.User"*>

</bean>

<!--create an object for dao, Si for the template -->

<bean id=*"dao"* class=*"com.UserManager.Userdao"*>

<property name=*"temp"* ref=*"template"*></property>

</bean>

<!--create an obj for the predefined class of hibernate template -->

<bean id=*"template"* class=*"org.springframework.orm.hibernate3.HibernateTemplate"*>

<property name=*"sessionFactory"* ref=*"sf"*></property>

</bean>

<!--create an obj for the session factory,SI,Hibernate -->

<bean id=*"sf"* class=*"org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean"*>

<property name=*"dataSource"* ref=*"ds"*></property>

<property name=*"packagesToScan"* value=*"com.UserManager"*></property>

<property name=*"hibernateProperties"*>

<props>

<prop key=*"hibernate.dialect"*>org.hibernate.dialect.MySQLDialect</prop>

<prop key=*"hibernate.hbm2ddl.auto"*>update</prop>

<prop key=*"hibernate.show\_sql"*>true</prop>

</props>

</property>

</bean>

<bean id=*"ds"* class=*"org.apache.commons.dbcp.BasicDataSource"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"*></property>

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/springauth"*></property>

<property name=*"username"* value=*"root"*></property>

<property name=*"password"* value=*"root"*></property>

</bean>

</beans>

**JSP PAGES**

**error.jsp**

<html>

<head>

</head>

<body>

<h2>Error: Page not found</h2>

</body>

</html>

**index.jsp**

<html>

<head>

<style>

*.center* {

text-align: *center*;

}

</style>

</head>

<body style="background-color:*lightblue*;">

<div class=*"center"*>

<h1>User Manager</h1>

<h2 class=*"hello-title"*>Search for a User and Update Information</h2>

<a href=*"/users"*>View user table</a>

<br><br>

<form method=*"post"* action=*"update"*>

Enter an id from the user table: <input type=*"text"* id=*"id"* name=*"id"* placeholder=*"Type here"* required><input type=*"submit"* value=*"Enter"* />

</form>

</div>

</body>

**Nouser.jsp**

<html>

<head>

</head>

<body>

<h2>Error: User not found</h2>

</body>

</html>

**Update.jsp**

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"* %>

<html>

<head>

<style>

**table,** **th,** **td** {

border: *1px solid black*;

margin: *auto*;

}

*.center* {

text-align: *center*;

}

</style>

</head>

<body style="background-color:*lightblue*;">

<div class=*"center"*>

<h2>Update Table</h2>

<p> User ID: ${ID}</p>

<table style="float:*inherit*">

<tr><th>ID</th><th>Name</th><th>Email</th><th>Password</th></tr>

<c:forEach items=*"*${user}*"* var=*"userE"* varStatus=*"count"*>

<tr id=*"*${count.index}*"*>

<td>${userE.id}</td>

<td>${userE.name}</td>

<td>${userE.email}</td>

<td>${userE.password}</td>

</tr>

</c:forEach>

</table>

<br><br>

<form method=*"post"* action=*"update2"*>

<br><h3>Edit user: ${ID}</h3>

<input type=*"text"* id=*"nameedit"* name=*"nameedit"* placeholder=*"Name"* required>

<input type=*"text"* id=*"emailedit"* name=*"emailedit"* placeholder=*"Email"* required>

<input type=*"text"* id=*"passwordedit"* name=*"passwordedit"* placeholder=*"Password"* required>

<input type=*"submit"* value=*"Enter"* />

</form>

</div>

</body>

</html>

**Update2.jsp**

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"* %>

<html>

<head>

<style>

**table,** **th,** **td** {

border: *1px solid black*;

margin: *auto*;

}

*.center* {

text-align: *center*;

}

</style>

</head>

<body style="background-color:*lightblue*;">

<div class=*"center"*>

<h2>Successfully Updated User</h2>

<p> User ID: ${IDedit}</p>

<div>

<table style="float:*inherit*">

<tr><th>ID</th><th>Name</th><th>Email</th><th>Password</th></tr>

<c:forEach items=*"*${user}*"* var=*"userE"* varStatus=*"count"*>

<tr id=*"*${count.index}*"*>

<td>${userE.id}</td>

<td>${userE.name}</td>

<td>${userE.email}</td>

<td>${userE.password}</td>

</tr>

</c:forEach>

</table>

</div>

<br><br>

<h3>Return to Homepage</h3>

<div>

<a href=*"/"*>Return</a>

</div>

</div>

</body>

</html>

**Users.jsp**

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"* %>

<html>

<head>

<style>

**table,** **th,** **td** {

border: *1px solid black*;

margin: *auto*;

}

*.center* {

text-align: *center*;

}

</style>

</head>

<body style="background-color:*lightblue*;">

<div class=*"center"*>

<h2>Users</h2>

<table style="float:*inherit*">

<tr><th>ID</th><th>Name</th><th>Email</th><th>Password</th></tr>

<c:forEach items=*"*${users}*"* var=*"user"* varStatus=*"count"*>

<tr id=*"*${count.index}*"*>

<td>${user.id}</td>

<td>${user.name}</td>

<td>${user.email}</td>

<td>${user.password}</td>

</tr>

</c:forEach>

</table>

</div>

</body>

</html>

**MSQL TABLE**

create database springauth;

use springauth;

select \* from user;

**2.Creating pom.xml for including the required components**

* In the Project Explorer, expand **User\_Manager** and double click on **pom.xml**
* Add the following entries:

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>SpringmvcandHibernate</groupId>

<artifactId>User\_Manager</artifactId>

<packaging>war</packaging>

<version>0.0.1-SNAPSHOT</version>

<name>User\_Manager Maven Webapp</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-orm -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-tx -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

<!-- https://mvnrepository.com/artifact/commons-dbcp/commons-dbcp -->

<dependency>

<groupId>commons-dbcp</groupId>

<artifactId>commons-dbcp</artifactId>

<version>1.4</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>3.6.9.Final</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-entitymanager -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-entitymanager</artifactId>

<version>3.6.9.Final</version>

</dependency>

<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.29</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>4.2.9.RELEASE</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-war-plugin</artifactId>

<version>3.3.1</version>

</plugin>

</plugins>

<finalName>User\_Manager</finalName>

</build>

</project>

**3.Building the project**

* From the **Project** menu at the top, click on **Build**
* If any compile errors are shown, fix them as required

**4.Pushing the code to your GitHub repositories:**

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**