# Project Name: Searching for a Specific User and Updating the User Information.

## GitHub Link:

<https://github.com/tamasjit/myjavaproject/tree/master/Phase%203/Phase-3_Practice_Project/q1_UserMgn>

## Source Code:

### Jsp:

#### Index.jsp:

<html>

<body>

<h2>Spring Application</h2>

<h2 class="hello-title">Hello ${name}!</h2>

<a href="users">List Users</a>

</body>

</html>

#### Edit.jsp:

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

<html>

<head>

<style>

table, th, td {

    border: 1px solid black;

}

</style>

</head>

<body>

    <h2>Edit User</h2>

    <table style="float: left">

        <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>

    <form action="/edit" method="post">

    <br><br><br>

        <label for="name">Name: </label> <input type="text" id="name"

            name="name"><br> <br> <label for="email">Email:</label>

        <input type="text" id="email" name="email"><br> <br>

        <label for="password">Password:</label> <input type="text"

            id="password" name="password"><br> <br> <input

            type="submit" value="Enter">

    </form>

    <a href="users">Back</a>

</body>

</html>

#### Error.jsp:

<!DOCTYPE html>

<html>

<body>

<h1>Something went wrong! </h1>

<h2>Our Engineers are on it</h2>

<a href="/">Go Home</a>

</body>

</html>

#### Fail.jsp:

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Field is empty</title>

</head>

<body>

Field is empty

<br><br>

<a href="users">Back to Users</a>

</body>

</html>

#### Success.jsp:

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Update successful</title>

</head>

<body>

Update successful

<br><br>

<a href="users">Back to Users</a>

</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;

}

</style>

</head>

<body>

    <h2>Users</h2>

    <table style="float: left">

        <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>

    <form action="/users" method="post">

    <br><br><br><br><br><br><br><br><br><br>

        <label for="id">ID:</label> <input type="text" id="id" name="id"><br>

        <br>

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

    </form>

</body>

</html>

### Controller:

#### appErrorController.java:

package com.example.UserManager.controllers;

import org.springframework.boot.web.servlet.error.ErrorController;

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

public class AppErrorController implements ErrorController {

    @RequestMapping("/error")

    public String handleError() {

        //TODO: do something like logging

        return "error";

    }

    @Override

    public String getErrorPath() {

        return null;

    }

}

#### mainController.java:

package com.example.UserManager.controllers;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

import org.springframework.web.bind.annotation.GetMapping;

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

import org.springframework.web.bind.annotation.RequestMethod;

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

import org.springframework.web.bind.annotation.SessionAttributes;

@Controller

public class MainController {

     @GetMapping(value="/")

     public String showIndexPage(ModelMap model,

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

         model.addAttribute("name", name);

         return "index";

     }

}

#### UserController.java:

package com.example.UserManager.controllers;

import java.util.ArrayList;

import java.util.Arrays;

import org.hibernate.mapping.Map;

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.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

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

import org.springframework.web.bind.annotation.RequestMethod;

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

import org.springframework.web.bind.annotation.ResponseBody;

import com.example.UserManager.entities.User;

import com.example.UserManager.services.UserService;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

@Controller

public class UserController {

    @Autowired

    private UserService userService;

    private int id;

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

    @GetMapping("/users")

    public String showUsers(ModelMap model) {

        logger.info("Getting all Users");

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

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

        model.addAttribute("users", users);

        return "users";

    }

    @PostMapping("/users")

    public String enterID(ModelMap model, @RequestParam(value="id") int userID) {

        this.id = userID;

        logger.info("Getting user");

        User user = userService.GetUserById(userID);

        Iterable<User> users = Arrays.asList(user);

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

        model.addAttribute("users", users);

        return "edit";

    }

    @PostMapping("/edit")

    public String editUser(ModelMap model, @RequestParam(value="name") String userName,

            @RequestParam(value="email") String userEmail, @RequestParam(value="password") String userPass) {

        if(!userName.isEmpty() && !userEmail.isEmpty() && !userPass.isEmpty()) {

            User user = new User();

            user.setEmail(userEmail);

            user.setName(userName);

            user.setPassword(userPass);

            user.setId(id);

            userService.UpdateUser(user);

            return "success";

        } else return "fail";

    }

}

#### userExceptionController.java:

package com.example.UserManager.controllers;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

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

import com.example.UserManager.exceptions.UserNotFoundException;

@ControllerAdvice

public class UserExceptionController {

    @ExceptionHandler(value = UserNotFoundException.class)

    public ResponseEntity<Object> exception(UserNotFoundException exception) {

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

    }

}

### Entitles:

#### User.java:

package com.example.UserManager.entities;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

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

@Table(name = "users")

public class User {

    @Id

    @GeneratedValue(strategy=GenerationType.AUTO)

    private Integer id;

    private String name;

    private String email;

    private String password;

    public String getPassword() {

        return password;

    }

    public void setPassword(String password) {

        this.password = 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;

    }

    @Override

    public String toString() {

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

    }

}

### Exceptions:

#### userNotFoundException.java:

package com.example.UserManager.exceptions;

public class UserNotFoundException extends RuntimeException {

    private static final long serialVersionUID = 1L;

    /\*

    private final String userId;

    public UserNotFoundException() {

      super();

      this.userId = id;

    }

    public String getUserId() {

      return userId;

    }

    \*/

}

### Repositories:

#### userRepository.java:

package com.example.UserManager.repositories;

import org.springframework.data.repository.CrudRepository;

import com.example.UserManager.entities.User;

public interface UserRepository extends CrudRepository<User, Integer> {

    public User findByName(String name);

}

### Services:

#### userService.java:

package com.example.UserManager.services;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import com.example.UserManager.entities.User;

import com.example.UserManager.exceptions.UserNotFoundException;

import com.example.UserManager.repositories.UserRepository;

@Service

public class UserService {

    @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) {

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

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

        if (!foundUser.isPresent()) {

            System.out.println("user does not exist");

            throw new UserNotFoundException();

        }

        return(foundUser.get());

    }

    public void UpdateUser(User usertoUpdate) {

        userRepository.save(usertoUpdate);

    }

}

#### userManagementApplication.java:

package com.example.UserManager;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class UserManagerApplication {

    public static void main(String[] args) {

        SpringApplication.run(UserManagerApplication.class, args);

    }

}

### Resources:

#### Application.properties:

spring.jpa.hibernate.ddl-auto=update

spring.datasource.url=jdbc:mysql://localhost:3306/Auther

spring.datasource.username=root

spring.datasource.password=Abc@123

logging.level.org.springframework.web: DEBUG

spring.mvc.view.prefix=/WEB-INF/jsp/

spring.mvc.view.suffix=.jsp

server.port=8090

server.error.whitelabel.enabled=false

#### pom.xml:

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

<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>

    <parent>

        <groupId>org.springframework.boot</groupId>

        <artifactId>spring-boot-starter-parent</artifactId>

        <version>2.4.3</version>

        <relativePath/> <!-- lookup parent from repository -->

    </parent>

    <groupId>com.example</groupId>

    <artifactId>UserManager</artifactId>

    <version>0.0.1-SNAPSHOT</version>

    <name>UserManager</name>

    <description>Demo project for Spring Boot</description>

    <properties>

        <java.version>17</java.version>

    </properties>

    <dependencies>

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-data-jpa</artifactId>

        </dependency>

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-web</artifactId>

        </dependency>

        <dependency>

            <groupId>mysql</groupId>

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

            <scope>runtime</scope>

        </dependency>

        <dependency>

            <groupId>org.projectlombok</groupId>

            <artifactId>lombok</artifactId>

            <optional>true</optional>

        </dependency>

        <dependency>

            <groupId>javax.servlet</groupId>

            <artifactId>jstl</artifactId>

            <version>1.2</version>

        </dependency>

        <dependency>

             <groupId>org.apache.tomcat.embed</groupId>

             <artifactId>tomcat-embed-jasper</artifactId>

             <scope>provided</scope>

        </dependency>

        <dependency>

            <groupId>org.springframework.boot</groupId>

            <artifactId>spring-boot-starter-test</artifactId>

            <scope>test</scope>

        </dependency>

    </dependencies>

    <build>

        <plugins>

            <plugin>

                <groupId>org.springframework.boot</groupId>

                <artifactId>spring-boot-maven-plugin</artifactId>

                <configuration>

                    <excludes>

                        <exclude>

                            <groupId>org.projectlombok</groupId>

                            <artifactId>lombok</artifactId>

                        </exclude>

                    </excludes>

                </configuration>

            </plugin>

        </plugins>

    </build>

</project>

#### 