SEARCHING FOR SPECIFIC USER AND UPDATING USER INFORMATION

Open pom.xml

```
<?xmI version="10" 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">

<modeIVersion>4.0.0</modelVersion>

<parent>

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

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

<version>2.4.3</version>

<relativePath f>

</parents

<groupId>com.exampIe</groupId>

<artifactId>UserManager</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>UserManager</name>

<description>Searching for a Specific User and Updating the User Information.</description>

<properties>

<java.version>1.8</java.versions

</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>org.projectIombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optionaI>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

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

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

<scope>provided</scope>

</dependency>



<dependency>

<groupId>javax.xml.bind</groupId>

<artifactId>jaxb-api</artifactId>

</dependency>

<dependency>

<groupId>org.javassist</groupId>

<artifactId>javassist</artifactId>
```

```xml
<version>3.25.0-GA</version>

</dependency>



<dependency>

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

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optionaI>

</dependency>

</dependencies>



<buiId>

<plugins>

<pIugin>

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

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

<configuration>

<excludes>

<exclude>

<groupId>org.projectIombok</groupId>

<artifactId>lombok</artifactId>

</excIude>

</excIudes>

</configuration>

</plugin>

</pIugins>
```

```
</buiId>



</project >
```

Create UserManagerApplication.java

```
package com.example. UserManager;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootAppIication
public class UserManagerApplication {
public static void main(String[] args) {
SpringApplication.run(UserManagerApplication.class, args);

}
}
```

Create package com.example.UserManager.controller

Create AppErrorController.java

```
package com.example.UserManager.controller;

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() {

//do something like logging
return "error";

}
@Override

public String getErrorPath() {
return null;

}
}
```

Create MainController.java

```
package com.example.UserManager.controller;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;
@ControIIer
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";

}
}
```

Create UserController.java

```
package com.example.UserManager.controller;
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.ModelAttribute;
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 com.example. UserManager.entities.User;
import com.example.UserManager.services.UserService;
@ControIIer
public class UserController {

//controls the functionality of the user entity

@Autowired

private UserService userservice;

Logger logger = LoggerFactory.getLogger(UserController.class);
@GetMapping("/users")
public String showUsers(ModeIMap model) { logger.info("Getting all users");
IterabIe<User> users = userService.GetAIlUsers();
logger.info("Passing users to view");
model.addAttribute("users", users );
return "users";
}
@RequestMapping(value ="/search/{id}", method = RequestMethod.POST) public String
searchUser(ModelMap model, @RequestParam("id") int id) ( logger.info("Searching for a user");
User user = userservice.GetUserById(id);
logger.info("Passing Searched User to View");
model.addAttribute("userSearch", user);
return "search";

}
@PostMapping("search/update")

public String updateUser(ModelMap model, @ModelAttribute("update") User user) {
logger.info("Updating a User");
userservice. UpdateUser(user);
model.addAttribute("updatedUser", user);
return "update";
}
}
```

Create UserExceptionController.java

```
package com.example.UserManager.controller;
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=UserNotFound Exception.class)

public ResponseEntity<Object> exception(UserNotFoundException ex) {
return new ResponseEntity<>("Product not found", HttpStatus.NOT_FOUND);

}
}
```

Create package com.example.UserManager.entities

Create User.java

```
package com.example.UserManager.entities;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
@Entity

public class User { //The Entity of a User; What it is.
 @Id
@GeneratedValue(strategy=GenerationType.AUTO)
private Integer id;
private String name;
private String email;
private String password;
```

```
public User() {
super();

}


public User(Integer id, String name, String email, String password) {
super();
this.id = id;
this.name = name;

this.email = email;
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;

}
public String getPassword() {
return password;
}
public void setPassword(String password) {
this.password = password;
}
@Override

public String tostring() {
```

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

}
}
```

Create package com.example.UserManager.exceptions

Create UserNotFoundException.java

```
package com.example. UserManager.exceptions;



public class UserNotFoundException extends RuntimeException {
private static final long
serialVersionUlD = 1L;
```

Create package com.example.UserManager.repositories

Create UserRepository.java

```
package com.example.UserManager.repositories;

import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;
import com.example. UserManager.entities.User;
@Repository
public interface UserRepository extends CrudRepository<User, Integer> {
public User findByName(String name);
}
```

Create package com.example.UserManager.services

Create UserService.java

```
package com.example.UserManager.services;
import java.utiI.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.findAII();

}
public User GetUserByName(String name) {
return userRepository.findByName(name);
}

public User GetUserById(Integer id) {

Optional<User> foundUser = userRepository.findById(id);
if(!found User.isPresent()) throw new UserNotFound Exception();
return foundUser.get();
}

public User UpdateUser(User userToU pdate) {
return userRepository.save(userToUpdate);
}
}
```

Src/main/resources

application.proprties

```
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysqI://IocaIhost:3306/mywork
spring.datasource.username=root
spring.datasource.password=password
logging.level.org.springframework.web: DEBUG
spring.mvc.view.prefix=/WEB-INF/jsp/
spring.mvc.view.suffix=.jsp
server.port=8080
```

src/main/webapp/WEB-INF/jsp

Create error.jsp

```
<!DOCTYPE htmI>

<htmI>

<body>

<h1>Something went wrong! </h1>

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

<a href=”/”>Go Home</a>
```

```
</body>

</htmI>
```

Create index.jsp

```
<htmI>

<body>

<h2>Spring AppIication</h2>



<h2 class=”hello-title“>HeIlo ${name}!</h2>

<a href=”osers”>List Users</a>

<form action=”search/{id)” method=”post”>

Enter ID Number: <input name=”id” type=”text” id=”id” placeholder=”1” required/>

<input name=”Submit" type=”submit”f>

</form>

</body>

</htmI>
```

Create search.jsp

```
<%@ page language=”java" contentType="text/html;
charset=UTF-8" pageEncoding=”UTF-8"%>
<%@ taglib prefix=“form" uri="http://www.springframework. org/tags/form"%>

<!DOCTYPE htmI PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/htmI4/loose.dtd">


<htmI>

<styIe>
table {
float: left;
```

```
th {

border—bottom: 1px solid black;

text-align: left;

}

</style>

<body>



<h2>Search for User</h2>



<tabIe>

<tr>

<th>ID</th>

<th>Name</th>

<th>Email</th>

<th>Password</th>

</tr>

<tr>

<td>$(userSearch.id)</td>

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

<td>$(userSearch.emaiI)</td>

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

</tr>

</table>



<br/>

<br/>
```

```
<br/>

<br/>

<br/>

<br/>

<form:form action="update" method="post” commandName=“update”>

<h3>Update This User?</h3>

<p>User ID: $(userSearch.id)</p>

<input type=”hidden” name=”id” id=”id” value=”$(userSearch.id)” required/>

<label for=”nome“>New Name:</label><br/>

<input type=”text” name="name" id="name" vaIue=”$(userSearch.name}” required/><br/>

<label for=”emoi/”>New Email:</label> <br/>

<input type=”text” name=”email” id=”email” value=”${userSearch.email)” required/><br/>

<label for=”password”>New Password:</label><br/>

<input type=”text“ name=”password“ id=”password" vaIue=”$(userSearch.password}” required/><br/
><br/>

<input type=”submit” value=”Submit”/>

</form:form>

<br />

<br />

<a href=”/”>Return to Menu</a>

</body>

</htmI>
```

Create update.jsp

```
<htmI>

<body>

<h2>Update Successful</h2>
```

```
Updated User Credentials: $(updatedUser.tostring())

<br/><br/>

<a href=”/”>Return to Menu</a>

</body>

</htmI>
```

Create users.jsp

```
<%@ taglib uri=”h/tp://java. sun.com/jsp/jstl/core” prefix="c"%>

<htmI>

<styIe>
table {
float: left;

}


table, th, td {


border: 1px solid black;



</style>

<head></head>

<body>

<h2>Users Page</h2>

<table>

<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>

</body>

</htmI>
```