package com.example.SpringSecurityManager;

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);

}

}

//controllers

package com.example.SpringSecurityManager.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() {

//do something like logging

return "error";

}

@Override

public String getErrorPath() {

return null;

}

}

package com.example.SpringSecurityManager.controllers;

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

import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import com.example.SpringSecurityManager.entities.User;

import com.example.SpringSecurityManager.services.UserService;

@Controller

public class MainController {

@Autowired

private UserService userService;

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

String currID = null;

@GetMapping(value="/")

public String showHomePage(ModelMap model,

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

model.addAttribute("name", name);

return "home";

}

@PostMapping(value="/index")

public String showIndexPage(@RequestParam("namelogin") String namelogin, @RequestParam("passwordlogin") String passwordlogin, ModelMap modelMap)

{

try {

User u = userService.GetUserByName(namelogin);

if(u.getName().equals(namelogin) && u.getPassword().equals(passwordlogin))

{

return "index";

}

else

{

return "home";

}

}

catch(NullPointerException e) {

return "home";

}

}

public boolean isNumber(String s)

{

if(s == null)

return false;

try

{

double db = Double.parseDouble(s);

}

catch(NumberFormatException e)

{

return false;

}

return true;

}

@PostMapping("/update")

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

try

{

User user = userService.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";

}

@PostMapping("/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 = userService.GetUserById(Integer.valueOf(currID));

userService.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";

}

}

package com.example.SpringSecurityManager.controllers;

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.ResponseBody;

import com.example.SpringSecurityManager.entities.User;

import com.example.SpringSecurityManager.services.UserService;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

@Controller

public class UserController {

@Autowired

private UserService userService;

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";

}

}

package com.example.SpringSecurityManager.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.SpringSecurityManager.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);

}

}

//entities

package com.example.SpringSecurityManager.entities;

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 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

package com.example.SpringSecurityManager.exceptions;

public class UserNotFoundException extends RuntimeException {

private static final long serialVersionUID = 1L;

}

//repositories

package com.example.SpringSecurityManager.repositories;

import org.springframework.data.repository.CrudRepository;

import com.example.SpringSecurityManager.entities.User;

public interface UserRepository extends CrudRepository<User, Integer> {

public User findByName(String name);

}

//services

package com.example.SpringSecurityManager.services;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import com.example.SpringSecurityManager.entities.User;

import com.example.SpringSecurityManager.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) 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);

}

}

//app properties

spring.jpa.hibernate.ddl-auto=update

spring.datasource.url=jdbc:mysql://${MYSQL\_HOST:localhost}:3306/userdb

spring.datasource.username=root

spring.datasource.password=Vipul@4567

logging.level.org.springframework.web: DEBUG

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

spring.mvc.view.suffix=.jsp

server.port=8090

<html>

<head>

</head>

<body>

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

</body>

</html>

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

<style>

.center {

text-align: center;

}

</style>

</head>

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

<div class="d-flex justify-content-center">

<div class="w-75 p-3">

<div class="center">

<h1 class="display-4">Spring Security</h1>

<div class="jumbotron">

<p class="lead">Login below to access the user's table</p>

<form method="post" action="index">

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

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

<input type="submit" value="Enter" class="btn btn-primary mb-2" />

</form>

</div>

</div>

</div>

</div>

</body>

</html>

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

<style>

.center {

text-align: center;

}

</style>

</head>

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

<div class="d-flex justify-content-center">

<div class="w-75 p-3">

<div class="center">

<h1 class="display-4">Search for a User By ID</h1>

<div class="jumbotron">

<h2 class="hello-title">Login Success</h2>

<p class="lead">View user table <a href="/users">here</a></p>

<br><br>

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

<p class="lead">Enter an id from the table: <p><input type="text" id="id" name="id" placeholder="Type here" required><input type="submit" value="Enter" class="btn btn-primary mb-2"/>

</form>

</div>

</div>

</div>

</div>

</body>

</html>

<html>

<head>

</head>

<body>

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

</body>

</html>

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

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

<style>

table, th, td {

border: 1px solid black;

margin: auto;

}

.center {

text-align: center;

}

</style>

</head>

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

<div class="d-flex justify-content-center">

<div class="w-75 p-3">

<div class="center">

<div class="jumbotron">

<h2 class="display-4">Update Table</h2>

<p class="lead"> 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" class="btn btn-primary mb-2"/>

</form>

</div>

</div>

</div>

</div>

</body>

</html>

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

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

<style>

table, th, td {

border: 1px solid black;

margin: auto;

}

.center {

text-align: center;

}

</style>

</head>

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

<div class="d-flex justify-content-center">

<div class="w-75 p-3">

<div class="center">

<h2 class="display-4">Successfully Updated User</h2>

<div class="jumbotron">

<p class="lead"> 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>

</div>

</div>

</div>

</body>

</html>

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

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

<style>

table, th, td {

border: 1px solid black;

margin: auto;

}

.center {

text-align: center;

}

</style>

</head>

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

<div class="d-flex justify-content-center">

<div class="w-75 p-3">

<div class="center">

<div class="jumbotron">

<h2 class="display-4">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>

</div>

</div>

</div>

</body>

</html>

//test

package com.example.UserManager;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

class UserManagerApplicationTests {

@Test

void contextLoads() {

}

}

//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>SpringSecurityManager</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>UserManager</name>

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

<properties>

<java.version>1.8</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>