**Handling User Authentication**

**Openpom.xml add dependencies**

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

<projectxmlns="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.project</groupId>

<artifactId>Authentication</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>Authentication</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-jersey</artifactId>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

<scope>runtime</scope>

<optional>true</optional>

</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>

<version>3.25.0-GA</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok-maven-plugin</artifactId>

<version>1.18.18.0</version>

<type>maven-plugin</type>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

<configuration>

</configuration>

</plugin>

<plugin>

<groupId>org.projectlombok</groupId>

<artifactId>lombok-maven-plugin</artifactId>

<version>1.18.18.0</version>

</plugin>

</plugins>

</build>

</project>

**Create packagecom.project.Authentication**

**Create AuthenticationApplication.java**

**package**com.project.Authentication;

**import**org.springframework.boot.SpringApplication;

**import**org.springframework.boot.autoconfigure.SpringBootApplication;

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

**import** com.project.Authentication.controllers.AuthenticationController;

**import**com.project.Authentication.entities.User;

**import**com.project.Authentication.exceptions.UserNotFoundException;

**import**com.project.Authentication.services.AuthenticationService;

@SpringBootApplication

@Import({

AuthenticationController.**class**,

UserNotFoundException.**class**,

AuthenticationService.**class**,

User.**class**

})

**publicclass**AuthenticationApplication {

**publicstaticvoid** main(String[] args) {

SpringApplication.*run*(AuthenticationApplication.**class**, args);

}

}

**Create package com.project.Authentication.controller**

**Create AuthenticationController.java**

package com.project.Authentication.controllers;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import org.springframework.stereotype.Controller;

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

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

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

import com.project.Authentication.entities.User;

import com.project.Authentication.services.AuthenticationService;

@Controller

public class AuthenticationController {

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

@Autowired

AuthenticationServiceauthService;

@GetMapping("/")

public String showGreeting() {

return "greeting";

}

@GetMapping("/Auth")

public String showLogin() {

return "authenticate";

}

@PostMapping("/Auth")

public String authenticateUser(@RequestParam("username") String username, @RequestParam("password") String pswd) {

User user = authService.GetUserByName(username);

logger.info(user.getName() + " attempted to login with " + user.getPassword());

String path = (authService.isValidPassword(pswd, user.getPassword())) ? "success" : "failure";

logger.info("The path return: " + path);

return path;

}

}

**Create package com.project.Authentication.entities**

**Crate User.java**

package com.project.Authentication.entities;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import javax.validation.constraints.NotNull;

@Entity

@Table(name = "user")

public class User {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

@NotNull

private Integer id;

@Column(name = "name")

@NotNull

private String name;

@Column(name = "email")

@NotNull

private String email;

@Column(name = "password")

@NotNull

private String password;

public User() {

super();

}

public User(@NotNull String name, @NotNull String password) {

this.name = name;

this.password = password;

}

public User(@NotNull String name, @NotNull String email, @NotNull String password) {

super();

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 "User [id=" + id + ", name=" + name + ", email=" + email + ", password=" + password + "]";

}

}

**Create package com.project.Authentication.exceptions**

**Create UserNotFoundException.java**

**package**com.project.Authentication.exceptions;

**publicclass**UserNotFoundException**extends**RuntimeException {

**privatestaticfinallong*serialVersionUID*** = 1L;

}

**Create package com.project.Authentication.repositories**

**Create AuthenticationRepository.java**

package com.project.Authentication.repositories;

import java.util.Optional;

import org.springframework.data.repository.CrudRepository;

import org.springframework.stereotype.Repository;

import com.project.Authentication.entities.User;

@Repository

public interface AuthenticationRepository extends CrudRepository<User, Integer> {

public Optional<User>findUserByName(String name);

}

**Create package com.project.Authentication.services**

**Create AuthenticationService.java**

package com.project.Authentication.services;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import com.project.Authentication.entities.User;

import com.project.Authentication.exceptions.UserNotFoundException;

import com.project.Authentication.repositories.AuthenticationRepository;

@Service

public class AuthenticationService {

@Autowired

AuthenticationRepositoryauthRepo;

public User GetUserByName(String name) {

Optional<User> found = authRepo.findUserByName(name);

if(found.isPresent()) return found.get();

else throw new UserNotFoundException();

}

public Boolean isValidPassword(String cmp, String actual) {

return ((cmp.equals(actual)) ? true : false);

}

}

**Src/main/resources**

**application.properties**

spring.jpa.hibernate.ddl-auto=update

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

**Open src/main/webapp/jsp**

**Create authenticate.jsp**

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

pageEncoding=*"ISO-8859-1"*%>

<%@taglibprefix=*"form"*uri=*"http://www.springframework.org/tags/form"*%>

<!DOCTYPEhtml>

<html>

<head>

<metacharset=*"ISO-8859-1"*>

<title>Authentication Page</title>

</head>

<h2>Login Page</h2>

<body>

welcome to the authentication page

<form:formaction=*"Auth"*method=*"post"*commandName=*"login"*>

<labelfor=*"username"*>Username:</label>

<inputname=*"username"*id=*"username"*type=*"text"*placeholder=*"Username"*required/>

<labelfor=*"password"*>Password:</label>

<inputname=*"password"*id=*"password"*type=*"password"*placholder=*"Password"*required/>

<inputtype=*"submit"*name=*"Submit"*/>

</form:form>

</body>

</html>

**Create failure.jsp**

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

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPEhtml>

<html>

<head>

<metacharset=*"ISO-8859-1"*>

<title>Failed Login</title>

</head>

<body>

<h1>You failed your login pal!

</h1><br/>

<ahref=*"/Auth"*>Attempt Login again</a>

</body>

</html>

**Create greeting.jsp**

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

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPEhtml>

<html>

<head>

<metacharset=*"ISO-8859-1"*>

<title>Landing Page</title>

</head>

<h2>Welcome Page</h2>

<body>

you reached the landing page

<ahref=*"Auth"*>Login</a>

</body>

</html>

**Create success.jsp**

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

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPEhtml>

<html>

<head>

<metacharset=*"ISO-8859-1"*>

<title>Successful Login Page</title>

</head>

<body>

<h1>Successful Login</h1>

</body>

</html>