AUTHENTICATION APPLICATION:

package com.example.authentication;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class AuthenticationApplication {

public static void main(String[] args) {

SpringApplication.run(AuthenticationApplication.class, args);

}

}

LOGIN CONTROLLER:

package com.example.authentication.controllers;

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

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

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

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

@Controller

public class LoginController {

@GetMapping("/")

public String showGreeting(ModelMap map) {

return "greeting";

}

@GetMapping("/login")

public String showLogin(ModelMap map) {

return "login";

}

@PostMapping("/login")

public String submitLogin(@RequestParam String username, @RequestParam String password){

//TODO:

return "Success";

}

}

USER:

package com.example.authentication.models;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

@Entity

public class User {

@Id

@GeneratedValue(strategy= GenerationType.AUTO)

private int id;

private String name;

private String password;

private String email;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

USER REPOSITORY:

package com.example.authentication.repositories;

import com.example.authentication.models.User;

import org.springframework.data.repository.CrudRepository;

public interface UserRepository extends CrudRepository<User, Integer> {

public User findByName(String name);

}

LOGIN SERVICE:

package com.example.authentication.services;

import com.example.authentication.models.User;

import com.example.authentication.repositories.UserRepository;

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

import org.springframework.stereotype.Service;

@Service

public class LoginService {

@Autowired

private UserRepository userRepository;

public boolean AddUser(User newUser) {

if (newUser == null) return false;

userRepository.save(newUser);

return true;

}

public boolean AuthenticateUser(User user, String enteredPassword)

{

return (user.getPassword().equals(enteredPassword));

}

public User GetUserByName(String name) {

Iterable<User> users = userRepository.findAll();

for (User user : users) {

if(user.getName().equals(name)) {

return user;

}

}

return null;

}

}

APPLICATION PROPERTIES:

#JSP view resolver support

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

spring.mvc.view.suffix=.jsp

server.port= 10000

#Database H2

spring.datasource.url=jdbc:h2:C:/temp-project/testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.hibernate.ddl-auto=update

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

AUTHENTICATION APPLICATION TEST:

package com.example.authentication;

import org.junit.jupiter.api.Test;

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

@SpringBootTest

class AuthenticationApplicationTests {

@Test

void contextLoads() {

}

}

AUTHENTICATION TESTS:

package com.example.authentication;

import com.example.authentication.models.User;

import com.example.authentication.repositories.UserRepository;

import com.example.authentication.services.LoginService;

import org.junit.jupiter.api.Test;

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

import org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTest;

import org.springframework.boot.test.autoconfigure.orm.jpa.TestEntityManager;

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

import static org.junit.jupiter.api.Assertions.assertEquals;

@DataJpaTest

public class AuthenticationTests {

@Autowired

private TestEntityManager entityManager;

@Autowired

private UserRepository userRepository;

@Test

public void whenFindByName\_thenReturnUser() {

// given

User dummyUser = new User();

dummyUser.setName("Dummy");

dummyUser.setEmail("test@test.com");

dummyUser.setPassword("password");

entityManager.persist(dummyUser);

entityManager.flush();

// when

User found = userRepository.findByName(dummyUser.getName());

// then

assertEquals(found.getName(), dummyUser.getName());

}

}

AUTHENTICATION WEB TESTS:

package com.example.authentication;

import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;

import static org.springframework.test.web.servlet.result.MockMvcResultHandlers.print;

import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.status;

import org.junit.jupiter.api.Test;

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

import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;

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

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

import org.springframework.boot.test.mock.mockito.MockBean;

import org.springframework.test.web.servlet.MockMvc;

import com.example.authentication.controllers.LoginController;

@SpringBootTest(webEnvironment = WebEnvironment.RANDOM\_PORT)

@AutoConfigureMockMvc

public class AuthenticationWebTests {

@Autowired

private MockMvc mockMvc;

@Test

public void shouldReturnDefaultMessage() throws Exception {

this.mockMvc.perform(get("/")).andDo(print()).andExpect(status().isOk());

}

}

LOGIN SERVICE TEST:

package com.example.authentication;

import org.junit.jupiter.api.Test;

import org.mockito.Mock;

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

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

import org.springframework.boot.test.mock.mockito.MockBean;

import com.example.authentication.models.User;

import com.example.authentication.repositories.UserRepository;

import com.example.authentication.services.LoginService;

import static org.junit.jupiter.api.Assertions.assertTrue;

import static org.mockito.Mockito.when;

@SpringBootTest

public class LoginServiceTest {

@Autowired

private LoginService loginService;

@MockBean

private UserRepository userRepository;

@Test

public void testAddUser() {

User user = new User();

user.setId(1);

user.setName("Dummy");

user.setEmail("dummy@dummy.com");

user.setPassword("password");

when(userRepository.save(user)).thenReturn(user);

assertTrue(loginService.AddUser(user));

}

@Test

public void testAuthenticateUser() {

User user = new User();

user.setId(1);

user.setName("Dummy");

user.setEmail("dummy@dummy.com");

user.setPassword("password");

when(userRepository.save(user)).thenReturn(user);

String password = "password";

assertTrue(loginService.AuthenticateUser(user, password));

}

}

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>3.1.0</version>

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

</parent>

<groupId>com.ecommerce</groupId>

<artifactId>HandlingUserAuthentication</artifactId>

<version>0.0.1-SNAPSHOT</version>

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

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

<scope>test</scope>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<version>1.4.200</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>