**CODE:**

package com.example;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class DemoApplication {

public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

}

}

package com.example;

public class User {

private Long id;

private String name;

public User() {}

public User(Long id, String name) {

this.id = id;

this.name = name;

}

public Long getId() { return id; }

public void setId(Long id) { this.id = id; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

}

package com.example;

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

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

@RestController

@RequestMapping("/users")

public class UserController {

@Autowired

private UserService userService;

@GetMapping("/{id}")

public ResponseEntity<User> getUser(@PathVariable Long id) {

return ResponseEntity.ok(userService.getUserById(id));

}

}

package com.example;

import org.springframework.stereotype.Service;

@Service

public class UserService {

public User getUserById(Long id) {

return new User(id, "Sabarna");

}

}

package com.example;

import org.junit.jupiter.api.Test;

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

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

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

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

import static org.mockito.Mockito.when;

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

import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.\*;

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

@WebMvcTest(UserController.class)

public class UserControllerTest {

@Autowired

private MockMvc mockMvc;

@MockBean

private UserService userService;

@Test

void testGetUserById() throws Exception {

User mockUser = new User(1L, "Sabarna");

when(userService.getUserById(1L)).thenReturn(mockUser);

mockMvc.perform(get("/users/1"))

.andDo(print()) // 👉 this prints actual response in console

.andExpect(status().isOk())

.andExpect(jsonPath("$.id").value(1L))

.andExpect(jsonPath("$.name").value("Sabarna"));

}

}

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

<projectDescription>

<name>mvn-1</name>

<comment></comment>

<projects>

</projects>

<buildSpec>

<buildCommand>

<name>org.eclipse.jdt.core.javabuilder</name>

<arguments>

</arguments>

</buildCommand>

<buildCommand>

<name>org.eclipse.m2e.core.maven2Builder</name>

<arguments>

</arguments>

</buildCommand>

</buildSpec>

<natures>

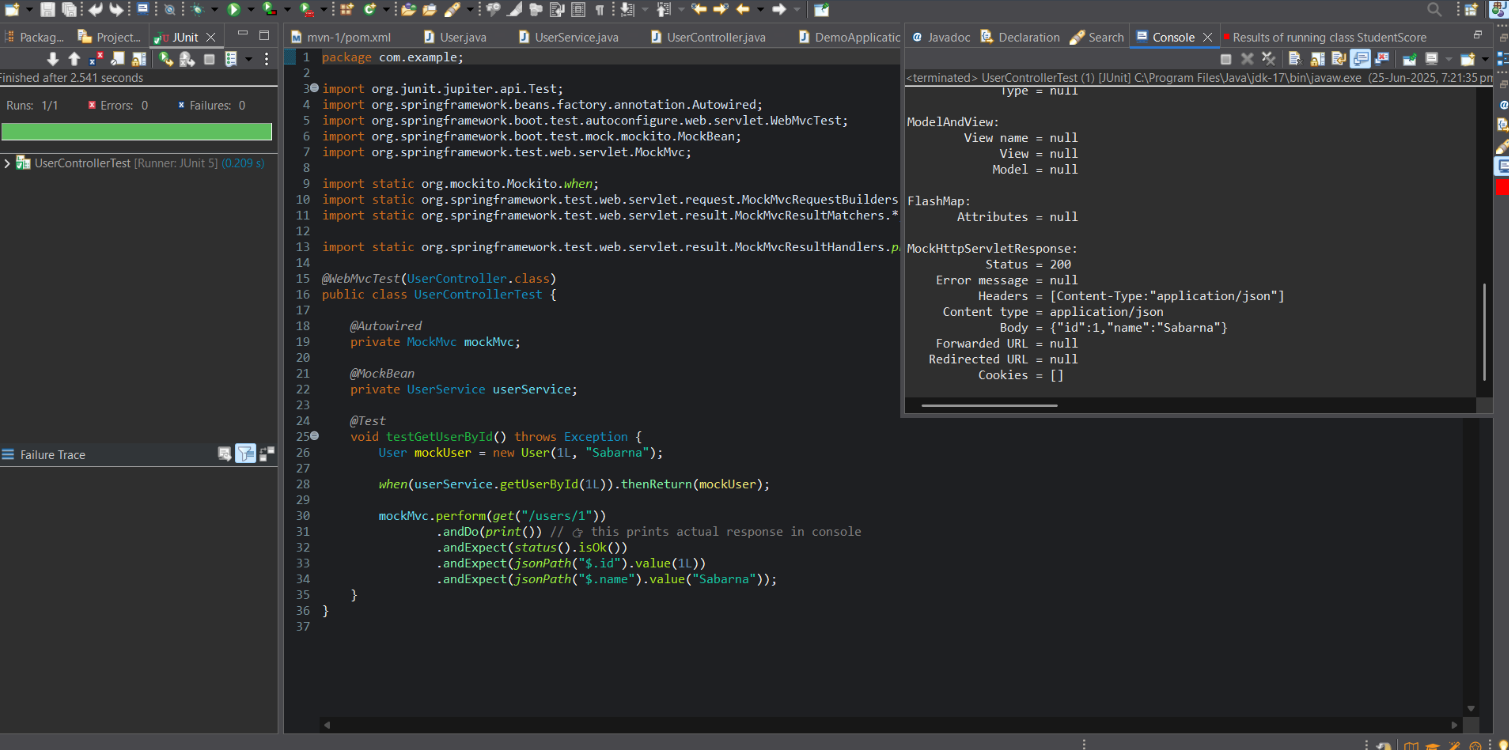
<nature>org.eclipse.jdt.core.javanature</nature>

<nature>org.eclipse.m2e.core.maven2Nature</nature>

</natures>

</projectDescription>

**OUTPUT:**

****