COGNIZANT

Digital Nurture 4.0

Deep Skilling - Java FSE

WEEK-2 HANDS ON

By Kaviya P

**Mocking Dependencies in Spring Tests using Mockito**

Exercise 1: Mocking a Service Dependency in a Controller Test

\*\*Task:\*\* Write a unit test for a Spring controller that uses a service to fetch data. Mock the

service dependency using Mockito.

**User.java**

**package** com.kaviyaP.SpringBootFirst.model;

**import** jakarta.persistence.Entity;

**import** jakarta.persistence.GeneratedValue;

**import** jakarta.persistence.GenerationType;

**import** jakarta.persistence.Id;

**import** jakarta.persistence.Table;

@Entity

@Table(name = "users")

**public** **class** User {

@Id

**private** Long id;

**private** String name;

**public** User() {}

**public** User(Long id, String name) {

**this**.id = id;

**this**.name = name;

}

// Getters and Setters

**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;

}

}

**MyUserService.java**

**package** com.kaviyaP.SpringBootFirst.service;

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

**import** org.springframework.stereotype.Service;

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.repository.UserRepository;

@Service

**public** **class** MyUserService {

@Autowired

**private** UserRepository userRepository;

**public** User getUserById(Long id) {

**return** userRepository.findById(id).orElse(**null**);

}

}

**UserController.java**

**package** com.kaviyaP.SpringBootFirst.controller;

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

**import** org.springframework.http.ResponseEntity;

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

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.service.MyUserService;

**import** com.kaviyaP.SpringBootFirst.service.UserService;

@RestController

@RequestMapping("/users")

**public** **class** UserController {

@Autowired

**private** MyUserService userService;

@GetMapping("/{id}")

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

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

}

}

**UserControllerTest.java**

**package** com.kaviyaP.SpringBootFirst;

**import** **static** org.mockito.Mockito.*mock*;

**import** **static** org.mockito.Mockito.*when*;

**import** org.junit.jupiter.api.BeforeEach;

**import** org.junit.jupiter.api.Test;

**import** org.mockito.Mockito;

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

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

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

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

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

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

**import** org.springframework.http.MediaType;

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

**import** **static** org.assertj.core.api.Assertions.~~assertThat~~;

**import** **static** org.hamcrest.Matchers.*is*;

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

**import** **static** org.junit.jupiter.api.Assertions.*assertThrows*;

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

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

**import** java.util.List;

**import** java.util.Optional;

**import** com.kaviyaP.SpringBootFirst.controller.UserController;

**import** com.kaviyaP.SpringBootFirst.exception.UserNotFoundException;

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.repository.UserRepository;

**import** com.kaviyaP.SpringBootFirst.service.MyUserService;

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

**import** **static** org.springframework.test.web.servlet.request.MockMvcRequestBuilders.\*;

**import** com.fasterxml.jackson.databind.ObjectMapper;

@WebMvcTest(UserController.**class**)

**public** **class** UserControllerTest {

@Autowired

**private** MockMvc mockMvc;

@~~MockBean~~

**private** MyUserService userService;

@Test

**public** **void** testGetUser() **throws** Exception {

User mockUser = **new** User(1L, "John Doe");

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

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

.accept(MediaType.***APPLICATION\_JSON***))

.andExpect(*status*().isOk())

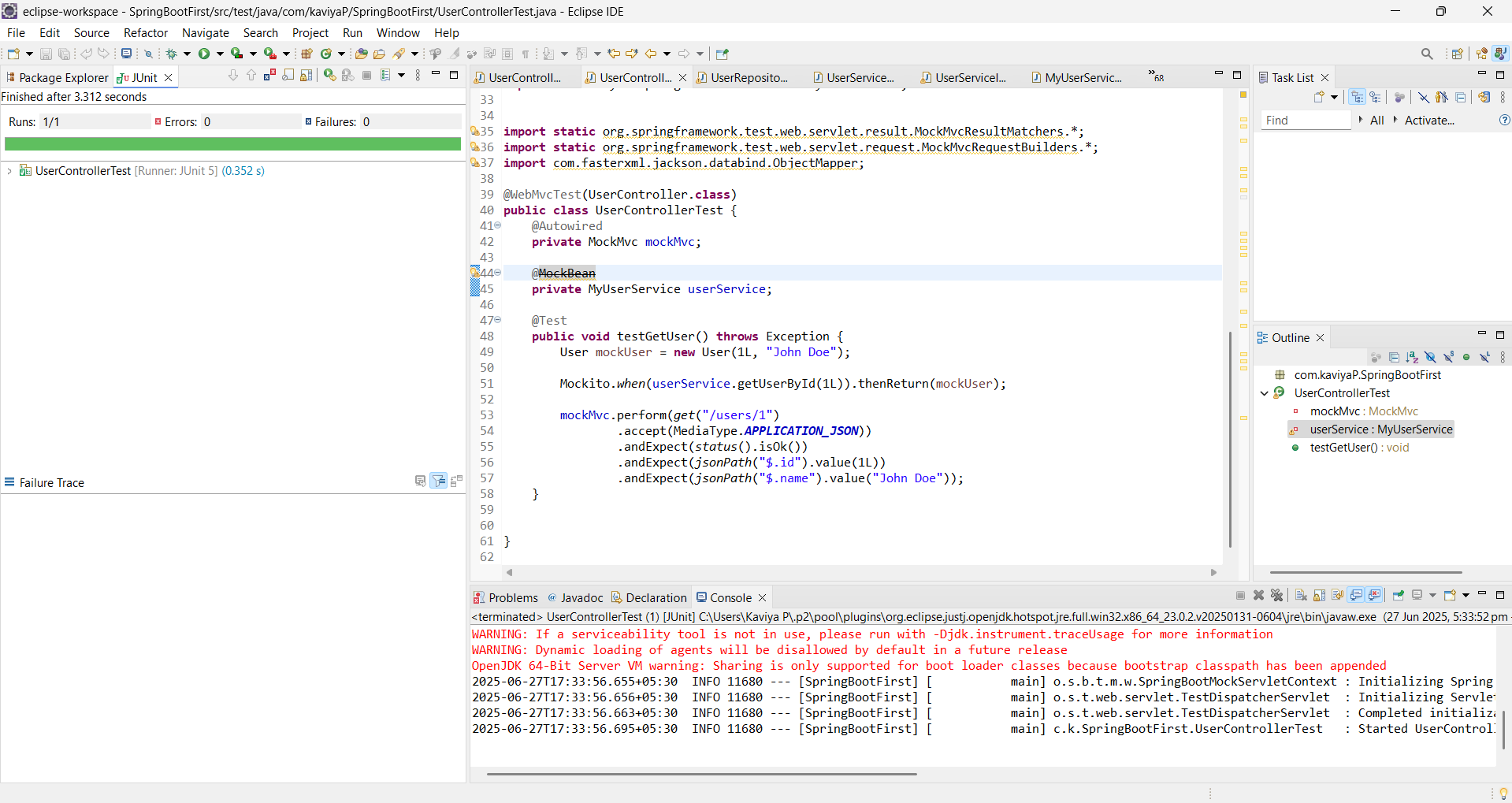
.andExpect(*jsonPath*("$.id").value(1L))

.andExpect(*jsonPath*("$.name").value("John Doe"));

}

}

OUTPUT



Exercise 2: Mocking a Repository in a Service Test

\*\*Task:\*\* Write a unit test for a Spring service that uses a repository to fetch data. Mock the

repository dependency using Mockito.

**User.java**

**package** com.kaviyaP.SpringBootFirst.model;

**import** jakarta.persistence.Entity;

**import** jakarta.persistence.GeneratedValue;

**import** jakarta.persistence.GenerationType;

**import** jakarta.persistence.Id;

**import** jakarta.persistence.Table;

@Entity

@Table(name = "users")

**public** **class** User {

@Id

**private** Long id;

**private** String name;

**public** User() {}

**public** User(Long id, String name) {

**this**.id = id;

**this**.name = name;

}

// Getters and Setters

**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;

}

}

**UserRepository.java**

**package** com.kaviyaP.SpringBootFirst.repository;

**import** java.util.List;

**import** org.springframework.data.jpa.repository.JpaRepository;

**import** com.kaviyaP.SpringBootFirst.model.User;

**public** **interface** UserRepository **extends** JpaRepository<User, Long> {

}

**UserService.java**

**package** com.kaviyaP.SpringBootFirst.service;

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

**import** org.springframework.stereotype.Service;

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.repository.UserRepository;

@Service

**public** **class** MyUserService {

@Autowired

**private** UserRepository userRepository;

**public** User getUserById(Long id) {

**return** userRepository.findById(id).orElse(**null**);

}

}

**UserControllerTest.java**

**package** com.kaviyaP.SpringBootFirst;

**import** **static** org.mockito.Mockito.*mock*;

**import** **static** org.mockito.Mockito.*when*;

**import** org.junit.jupiter.api.BeforeEach;

**import** org.junit.jupiter.api.Test;

**import** org.mockito.Mockito;

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

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

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

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

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

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

**import** org.springframework.http.MediaType;

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

**import** **static** org.assertj.core.api.Assertions.~~assertThat~~;

**import** **static** org.hamcrest.Matchers.*is*;

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

**import** **static** org.junit.jupiter.api.Assertions.*assertThrows*;

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

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

**import** java.util.List;

**import** java.util.Optional;

**import** com.kaviyaP.SpringBootFirst.controller.UserController;

**import** com.kaviyaP.SpringBootFirst.exception.UserNotFoundException;

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.repository.UserRepository;

**import** com.kaviyaP.SpringBootFirst.service.MyUserService;

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

**import** **static** org.springframework.test.web.servlet.request.MockMvcRequestBuilders.\*;

**import** com.fasterxml.jackson.databind.ObjectMapper;

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

**import** **static** org.mockito.Mockito.\*;

**public** **class** UserControllerTest {

@Test

**public** **void** testGetUserById() {

// Create mock UserRepository

UserRepository userRepository = *mock*(UserRepository.**class**);

// Create service and inject mock

MyUserService userService = **new** MyUserService();

// Use reflection or constructor to inject mock if necessary

// For simplicity, setting field directly using reflection here

**try** {

**var** field = MyUserService.**class**.getDeclaredField("userRepository");

field.setAccessible(**true**);

field.set(userService, userRepository);

} **catch** (Exception e) {

**throw** **new** RuntimeException(e);

}

// Prepare data

User user = **new** User(1L, "Alice");

*when*(userRepository.findById(1L)).thenReturn(Optional.*of*(user));

// Execute

User result = userService.getUserById(1L);

// Verify

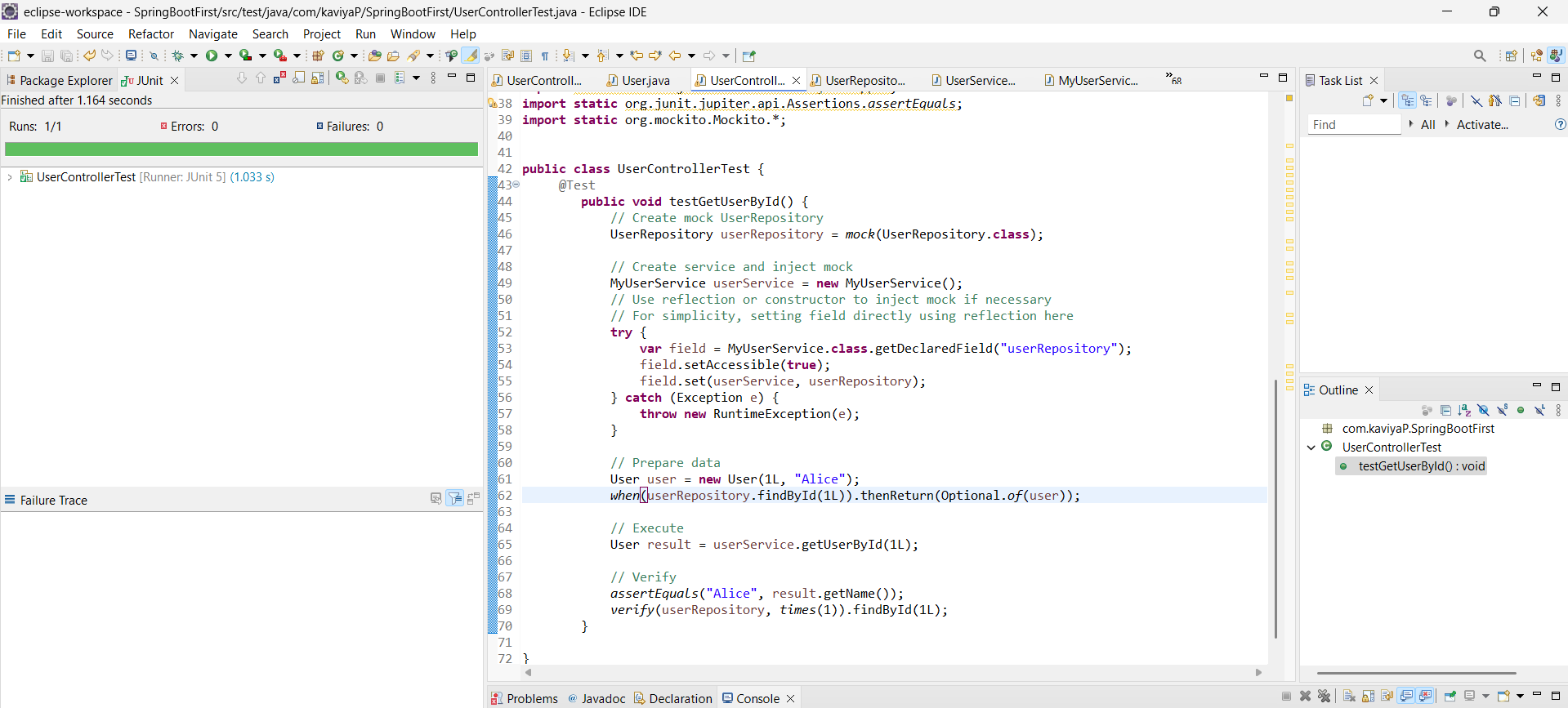
*assertEquals*("Alice", result.getName());

*verify*(userRepository, *times*(1)).findById(1L);

}

}

OUTPUT



Exercise 3: Mocking a Service Dependency in an Integration Test

\*\*Task:\*\* Write an integration test for a Spring Boot application that mocks a service

dependency using Mockito.

**User.java**

**package** com.kaviyaP.SpringBootFirst.model;

**import** jakarta.persistence.Entity;

**import** jakarta.persistence.GeneratedValue;

**import** jakarta.persistence.GenerationType;

**import** jakarta.persistence.Id;

**import** jakarta.persistence.Table;

@Entity

@Table(name = "users")

**public** **class** User {

@Id

**private** Long id;

**private** String name;

**public** User() {}

**public** User(Long id, String name) {

**this**.id = id;

**this**.name = name;

}

// getters and setters

**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; }

}

**UserRepository.java**

**package** com.kaviyaP.SpringBootFirst.repository;

**import** java.util.List;

**import** org.springframework.data.jpa.repository.JpaRepository;

**import** com.kaviyaP.SpringBootFirst.model.User;

**public** **interface** UserRepository **extends** JpaRepository<User, Long> {}

**MyUserService java**

**package** com.kaviyaP.SpringBootFirst.service;

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

**import** org.springframework.stereotype.Service;

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.repository.UserRepository;

@Service

**public** **class** MyUserService {

@Autowired

**private** UserRepository userRepository;

**public** User getUserById(Long id) {

**return** userRepository.findById(id).orElse(**null**);

}

}

**UserController.java**

**package** com.kaviyaP.SpringBootFirst.controller;

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

**import** org.springframework.http.ResponseEntity;

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

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.service.MyUserService;

**import** com.kaviyaP.SpringBootFirst.service.UserService;

@RestController

@RequestMapping("/users")

**public** **class** UserController {

@Autowired

**private** MyUserService userService;

@GetMapping("/{id}")

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

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

}

}

**UserControllerTest.java**

**package** com.kaviyaP.SpringBootFirst.controller;

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

**import** org.springframework.http.ResponseEntity;

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

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.service.MyUserService;

**import** com.kaviyaP.SpringBootFirst.service.UserService;

@RestController

@RequestMapping("/users")

**public** **class** UserController {

@Autowired

**private** MyUserService userService;

@GetMapping("/{id}")

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

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

}

}

**package** com.kaviyaP.SpringBootFirst;

**import** **static** org.mockito.Mockito.*mock*;

**import** **static** org.mockito.Mockito.*when*;

**import** org.junit.jupiter.api.BeforeEach;

**import** org.junit.jupiter.api.Test;

**import** org.mockito.Mockito;

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

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

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

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

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

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

**import** org.springframework.http.MediaType;

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

**import** **static** org.assertj.core.api.Assertions.~~assertThat~~;

**import** **static** org.hamcrest.Matchers.*is*;

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

**import** **static** org.junit.jupiter.api.Assertions.*assertThrows*;

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

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

**import** java.util.List;

**import** java.util.Optional;

**import** com.kaviyaP.SpringBootFirst.controller.UserController;

**import** com.kaviyaP.SpringBootFirst.exception.UserNotFoundException;

**import** com.kaviyaP.SpringBootFirst.model.User;

**import** com.kaviyaP.SpringBootFirst.repository.UserRepository;

**import** com.kaviyaP.SpringBootFirst.service.MyUserService;

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

**import** **static** org.springframework.test.web.servlet.request.MockMvcRequestBuilders.\*;

**import** com.fasterxml.jackson.databind.ObjectMapper;

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

**import** **static** org.mockito.Mockito.\*;

@SpringBootTest

@AutoConfigureMockMvc

**public** **class** UserControllerTest {

@Autowired

**private** MockMvc mockMvc;

@~~MockBean~~

**private** MyUserService userService;

@Test

**void** testGetUserById() **throws** Exception {

User mockUser = **new** User(1L, "John Doe");

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

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

.accept(MediaType.***APPLICATION\_JSON***))

.andExpect(*status*().isOk())

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

.andExpect(*jsonPath*("$.name").value("John Doe"));

}

}

OUTPUT

