**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.

**MockMvcTestingApplication.java :-**

package org.example1;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class MockMvcTestingApplication {  
 public static void main(String[] args) {  
 SpringApplication.*run*(MockMvcTestingApplication.class, args);  
 }  
}

**User.java :-**

package org.example1;  
  
import jakarta.persistence.Entity;  
import jakarta.persistence.Id;  
  
@Entity  
public class User {  
 @Id  
 private Long id;  
 private String name;  
  
 // Constructors  
 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;  
 }  
}

**UserController.java :-**

package org.example1;  
  
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) {  
 User user = userService.getUserById(id);  
 return ResponseEntity.*ok*(user);  
 }  
}

**UserRepository.java :-**

package org.example1;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface UserRepository extends JpaRepository<User, Long> {  
}

**UserService :-**

package org.example1;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface UserRepository extends JpaRepository<User, Long> {  
}

**UserControllerTest.java :-**

package org.example1;  
  
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.\*;  
  
@WebMvcTest(UserController.class)  
public class UserControllerTest {  
  
 @Autowired  
 private MockMvc mockMvc;  
  
 @MockBean  
 private UserService userService;  
  
 @Test  
 public void testGetUser() throws Exception {  
 User mockUser = new User(1L, "Naresh");  
 *when*(userService.getUserById(1L)).thenReturn(mockUser);  
  
 mockMvc.perform(*get*("/users/1"))  
 .andExpect(*status*().isOk())  
 .andExpect(*jsonPath*("$.name").value("Naresh"))  
 .andExpect(*jsonPath*("$.id").value(1));  
 }  
}

**pom.xml :-**

<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>3.2.5</version> <!-- Use the latest stable version -->  
 <relativePath/> <!-- Required for parent resolution -->  
 </parent>  
  
 <groupId>org.example</groupId>  
 <artifactId>logback-multiappender-demo</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <packaging>jar</packaging>  
  
 <properties>  
 <java.version>17</java.version>  
 </properties>  
  
 <dependencies>  
 <!-- Spring Boot Web -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
  
 <!-- Spring Boot Data JPA -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
 </dependency>  
  
 <!-- Spring Boot Test -->  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- Jackson JSON -->  
 <dependency>  
 <groupId>com.fasterxml.jackson.core</groupId>  
 <artifactId>jackson-databind</artifactId>  
 </dependency>  
  
 <!-- Jakarta Persistence -->  
 <dependency>  
 <groupId>jakarta.persistence</groupId>  
 <artifactId>jakarta.persistence-api</artifactId>  
 <version>3.1.0</version>  
 </dependency>  
 </dependencies>  
  
 <build>  
 <plugins>  
 <!-- Compiler Plugin -->  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.11.0</version>  
 <configuration>  
 <source>17</source>  
 <target>17</target>  
 </configuration>  
 </plugin>  
 </plugins>  
 </build>  
</project>

Output :-

