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.

Step-by-Step Solution:

1. **Create the User Entity:**

```
@Entity

public class User {

    @Id

    private Long id;

    private String name;

    // getters and setters

}
```

2. **Create the UserService:**

```
@Service

public class UserService {

    @Autowired

    private UserRepository userRepository;

    public User getUserById(Long id) {

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

    }

}
```

3. **Create the UserController:**

```
@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));

   }

}
```

4. **Create the UserControllerTest:**

Write code for this.

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.

Step-by-Step Solution:

1. **Create the User Entity:**

```
@Entity

public class User {

   @Id

   private Long id;

   private String name;

   // getters and setters

}
```

2. **Create the UserRepository:**

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

}
```

3. **Create the UserService:**

```
@Service
public class UserService {
    @Autowired
    private UserRepository userRepository;
    public User getUserById(Long id) {
        return userRepository.findById(id).orElse(null);
    }
}
```

4. **Create the UserServiceTest:**

Write code for this.

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.

Step-by-Step Solution:

1. **Create the User Entity:**

```
@Entity
public class User {
    @Id
    private Long id;
    private String name;
    // getters and setters
}
```

2. **Create the UserService:**

```
@Service
public class UserService {
    @Autowired
    private UserRepository userRepository;
    public User getUserById(Long id) {
        return userRepository.findById(id).orElse(null);
    }
}
```

3. **Create the UserController:**

```
@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));
    }
}
```

4. **Create the UserIntegrationTest:**

Write code for this.

Hint: Use ‘@SpringBootTest’, ‘@AutoConfigureMockMvc’