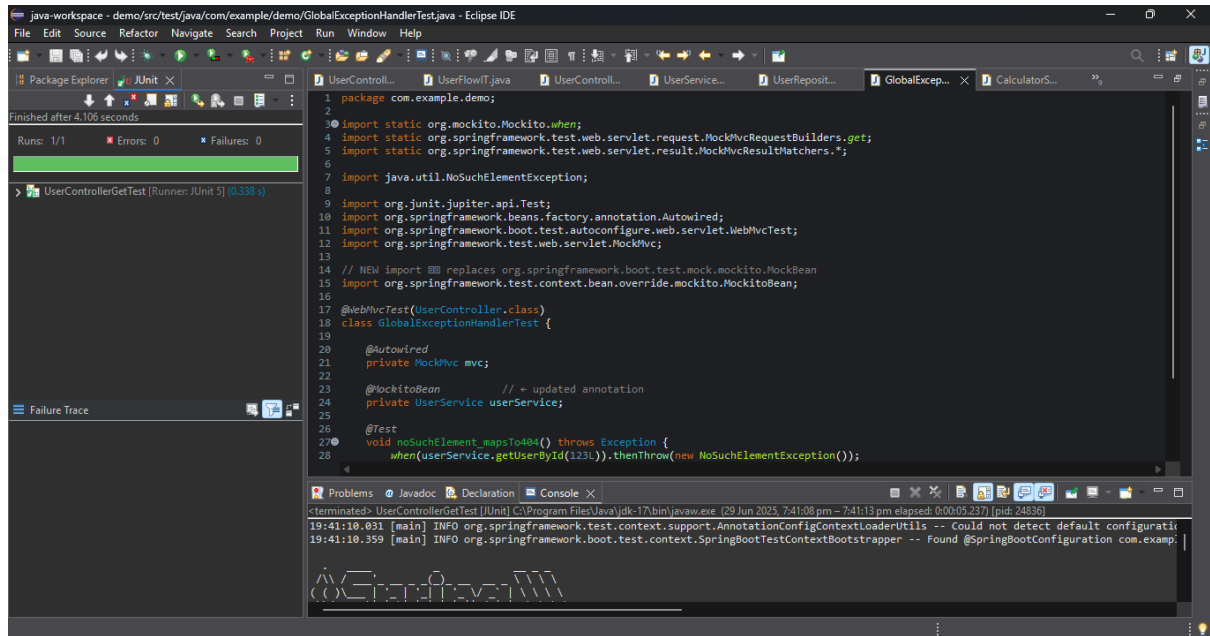


Use `@ExtendWith(MockitoExtension.class)` and `@Mock + @InjectMocks` (or `@DataJpaTest` with `@MockBean`) to isolate service code from the database.

### Exercise 3: Testing a REST Controller with MockMvc

Output:



```
1 package com.example.demo;
2
3 import static org.mockito.Mockito.when;
4 import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;
5 import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.*;
6
7 import java.util.NoSuchElementException;
8
9 import org.junit.jupiter.api.Test;
10 import org.springframework.beans.factory.annotation.Autowired;
11 import org.springframework.boot.test.autoconfigure.web.servlet.MockMvcTest;
12 import org.springframework.test.web.servlet.MockMvc;
13
14 // NEW import replaces org.springframework.boot.test.mock.mockito.MockBean
15 import org.springframework.test.context.bean.override.mockito.MockitoBean;
16
17 @WebMvcTest(UserController.class)
18 class GlobalExceptionHandlerTest {
19
20     @Autowired
21     private MockMvc mvc;
22
23     @MockitoBean // + updated annotation
24     private UserService userService;
25
26     @Test
27     void noSuchElement_mapsTo404() throws Exception {
28         when(userService.getUserById(123L)).thenReturn(new NoSuchElementException());
29     }
30 }
```

Finished after 4.106 seconds

Runs: 1/1 Errors: 0 Failures: 0

UserControllerGetTest [Runner: JUnit 5] (0.338 s)

Failure Trace

Problems Javadoc Declaration Console

<terminated> UserControllerGetTest [JUnit] C:\Program Files\Java\jdk-17\bin\javaw.exe (29 Jun 2025, 7:41:08 pm - 7:41:13 pm elapsed: 0:00:05.237) [pid: 24836]

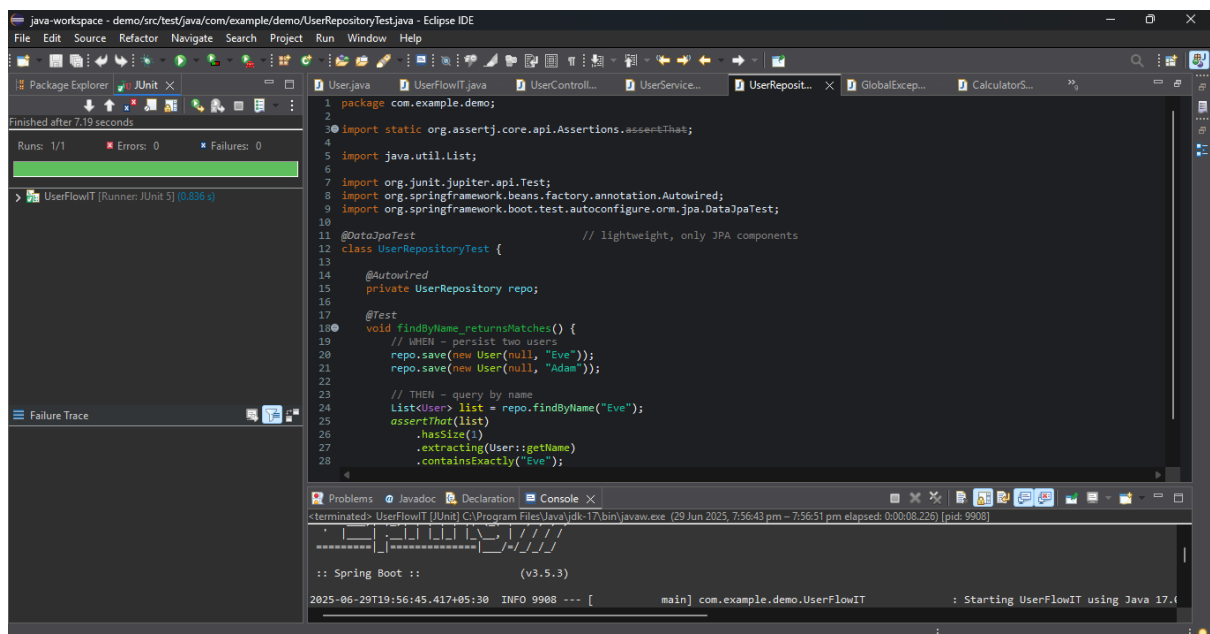
19:41:10.031 [main] INFO org.springframework.test.context.support.AnnotationConfigContextLoaderUtils -- Could not detect default configuration

19:41:10.359 [main] INFO org.springframework.boot.test.context.SpringBootTestContextBootstrapper -- Found @SpringBootTestConfiguration com.example

Spin up only the web layer (@WebMvcTest) and assert JSON/body results with MockMvc—no full context needed.

### Exercise 4: Integration Test with Spring Boot

Output:



```
1 package com.example.demo;
2
3 import static org.assertj.core.api.Assertions.assertThat;
4
5 import java.util.List;
6
7 import org.junit.jupiter.api.Test;
8 import org.springframework.beans.factory.annotation.Autowired;
9 import org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTest;
10
11 @DataJpaTest // lightweight, only JPA components
12 class UserRepositoryTest {
13
14     @Autowired
15     private UserRepository repo;
16
17     @Test
18     void findByName_returnsMatches() {
19         // WHEN - persist two users
20         repo.save(new User(null, "Eve"));
21         repo.save(new User(null, "Adam"));
22
23         // THEN - query by name
24         List<User> list = repo.findByName("Eve");
25         assertThat(list)
26             .hasSize(1)
27             .extracting(User::getName)
28             .containsExactly("Eve");
29     }
30 }
```

Finished after 7.19 seconds

Runs: 1/1 Errors: 0 Failures: 0

UserFlowIT [Runner: JUnit 5] (0.836 s)

Failure Trace

Problems Javadoc Declaration Console

<terminated> UserFlowIT [JUnit] C:\Program Files\Java\jdk-17\bin\javaw.exe (29 Jun 2025, 7:56:43 pm - 7:56:51 pm elapsed: 0:00:08.226) [pid: 9908]

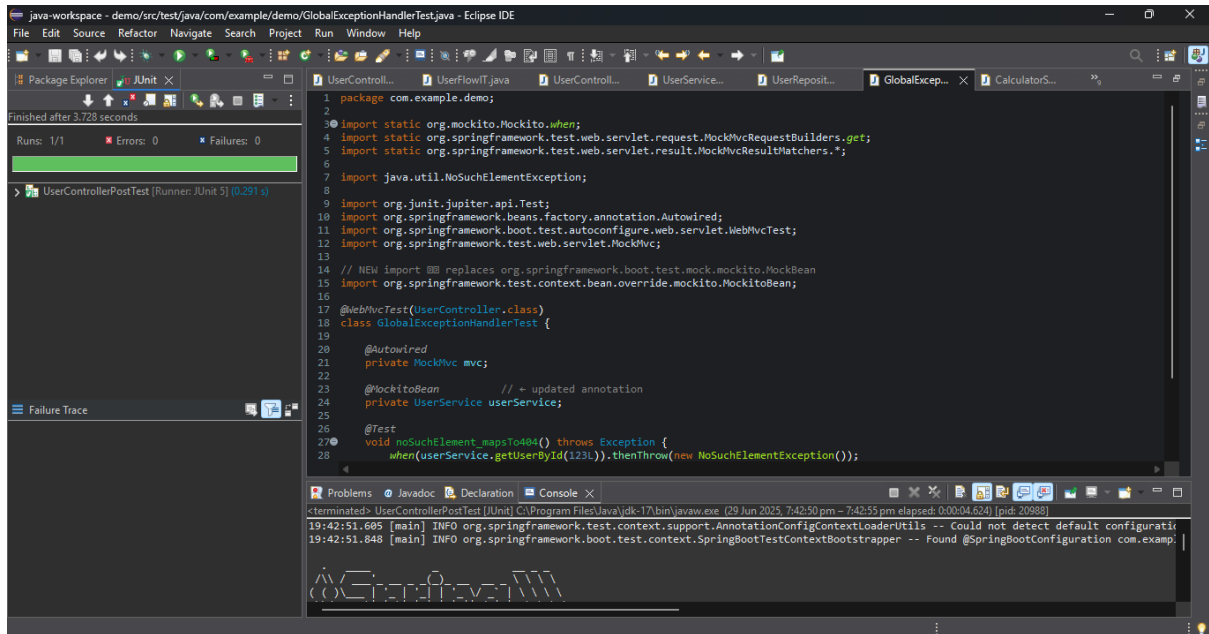
Spring Boot :: (v3.5.3)

2025-06-29T19:56:45.417+05:30 INFO 9908 --- [main] com.example.demo.UserFlowIT : Starting UserFlowIT using Java 17.0

Annotate with @SpringBootTest, hit the real HTTP endpoint, and assert against an in-memory DB to verify the entire stack.

## Exercise 5: Test Controller POST Endpoint

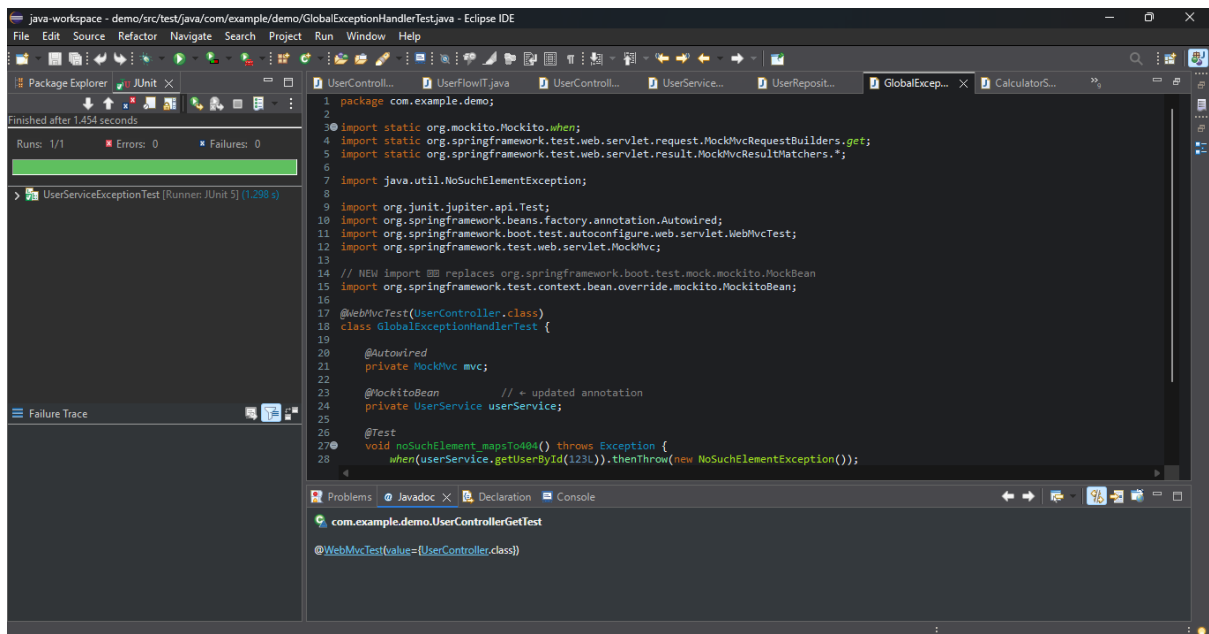
### Output:



Use MockMvc's .post() with a JSON payload, validate status().isOk() and returned body to guarantee create flow.

## Exercise 6: Test Service Exception Handling

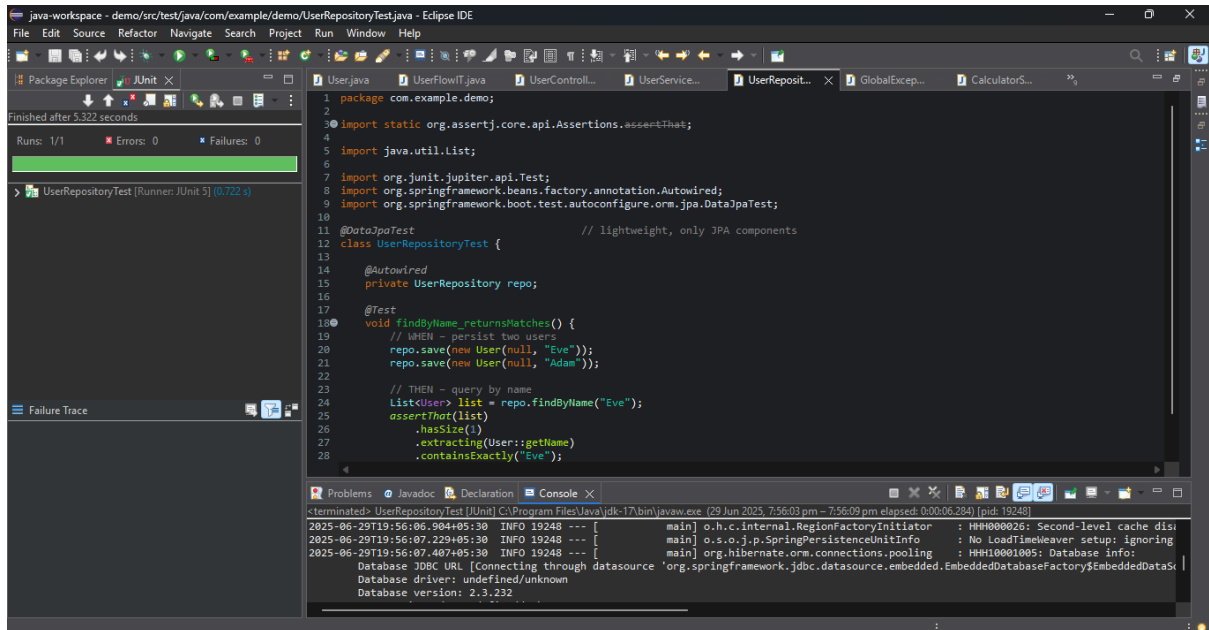
### Output:



Wrap the call in assertThrows (or MockMvc for REST) to make sure the service signals “user not found” as designed.

## Exercise 7: Test Custom Repository Query

### Output:



```
1 package com.example.demo;
2
3 import static org.assertj.core.api.Assertions.assertThat;
4
5 import java.util.List;
6
7 import org.junit.jupiter.api.Test;
8 import org.springframework.beans.factory.annotation.Autowired;
9 import org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTest;
10
11 @DataJpaTest // lightweight, only JPA components
12 class UserRepositoryTest {
13
14     @Autowired
15     private UserRepository repo;
16
17     @Test
18     void findByName_returnsMatches() {
19         // WHEN - persist two users
20         repo.save(new User(null, "Eve"));
21         repo.save(new User(null, "Adam"));
22
23         // THEN - query by name
24         List<User> list = repo.findByName("Eve");
25         assertThat(list)
26             .hasSize(1)
27             .extracting(User::getName)
28             .containsExactly("Eve");
29     }
30 }
```

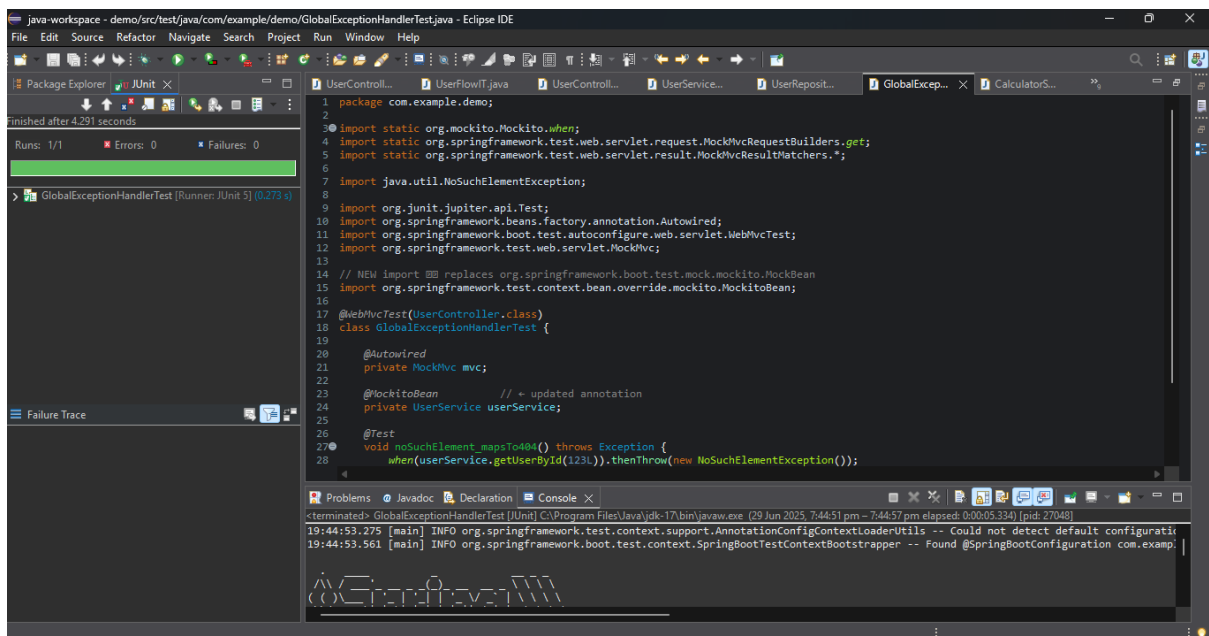
Finished after 5.322 seconds  
Runs: 1/1 Errors: 0 Failures: 0  
UserRepositoryTest [Runner: JUnit 5] (0.722 s)

Database JDBC URL [Connecting through datasource 'org.springframework.jdbc.datasource.embedded.EmbeddedDatabaseFactory\$EmbeddedDataS  
Database driver: undefined/unknown  
Database version: 2.3.232

Populate the H2 test DB, invoke `findByName`, and assert the result list—proves your JPQL/derived query works.

## Exercise 8: Test Controller Exception Handling

### Output:



```
1 package com.example.demo;
2
3 import static org.mockito.Mockito.when;
4 import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;
5 import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.*;
6
7 import java.util.NoSuchElementException;
8
9 import org.junit.jupiter.api.Test;
10 import org.springframework.beans.factory.annotation.Autowired;
11 import org.springframework.boot.test.autoconfigure.web.servlet.WebMvcTest;
12 import org.springframework.test.web.servlet.MockMvc;
13
14 // NEW import @ replaces org.springframework.boot.test.mock.mockito.MockBean
15 import org.springframework.test.context.bean.override.mockito.MockitoBean;
16
17 @WebMvcTest(UserController.class)
18 class GlobalExceptionHandlerTest {
19
20     @Autowired
21     private MockMvc mvc;
22
23     @MockitoBean // + updated annotation
24     private UserService userService;
25
26     @Test
27     void noSuchElement_mapsTo404() throws Exception {
28         when(userService.getUserById(123L)).thenReturn(new NoSuchElementException());
29     }
30 }
```

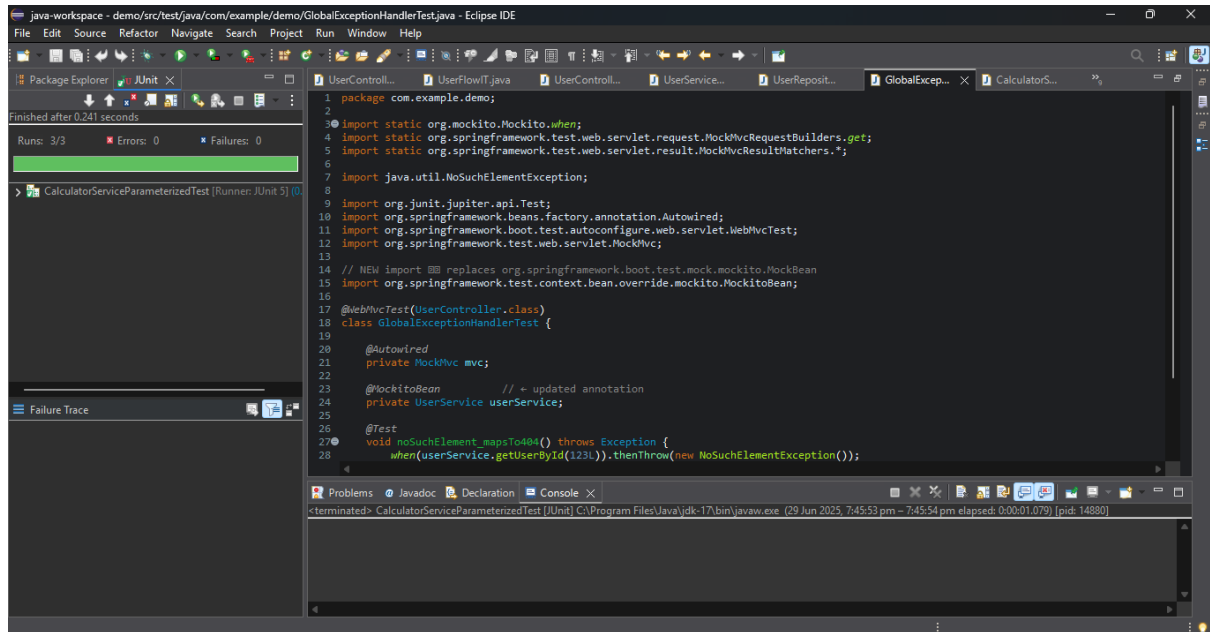
Finished after 4.291 seconds  
Runs: 1/1 Errors: 0 Failures: 0  
GlobalExceptionHandlerTest [Runner: JUnit 5] (0.723 s)

19:44:53.275 [main] INFO org.springframework.test.context.support.AnnotationConfigContextLoaderUtils -- Could not detect default configurati  
19:44:53.561 [main] INFO org.springframework.boot.test.context.SpringBootTestContextBootstrapper -- Found @SpringBootConfiguration com.examp

Trigger a failing request and expect 404 plus body text “User not found”, confirming `@ControllerAdvice` wiring.

## Exercise 9: Parameterized Test with Junit

### Output:



Use `@ParameterizedTest` + `@CsvSource` (or similar) to run the same assertion over many (input,expected) pairs with minimal code.