



Progetto **MYTHOS**

Test Execution Report Piattaforma Locale

Partecipanti

Egidio Mario 0512105122

Pagliaro Vincenzo 0512105546

Project Manager

De Lucia Andrea

Pecorelli Fabiano

Sommario

1. Testing di unità 3

2. Testing di integrazione 5

3. Test di sistema..... 8

1. Testing di unità

Il test di unità è stato eseguito sulle seguenti classi DAO:

- DetailDAO
- OrderDAO
- OrderedProductDAO
- ProductDAO
- ReservationOptionDAO
- RoleDAO
- TableDAO
- UserDAO

Per ogni classe DAO è stata generata una classe Test.

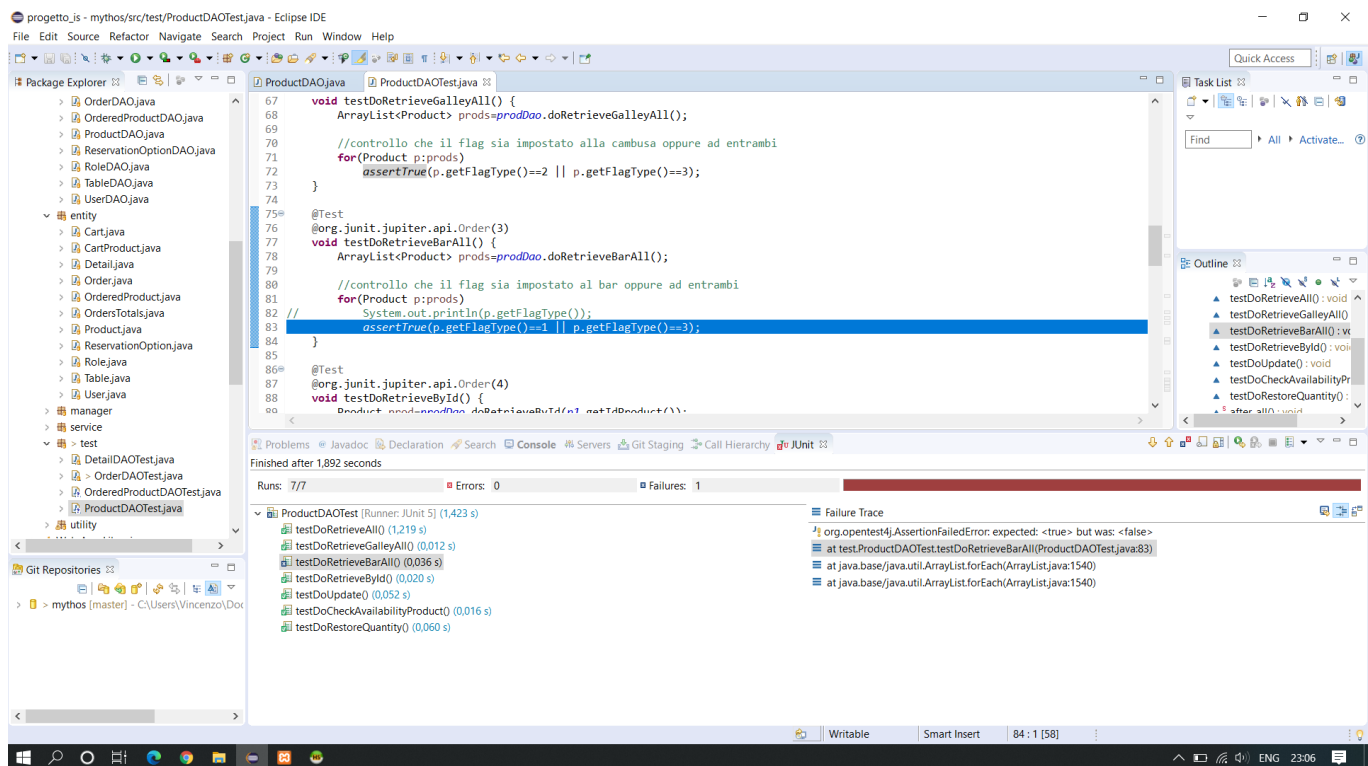
È stata testata, Inoltre, anche la entity “cart”.

Qui vengono riportati i risultati dei test eseguiti da una classe “TestSuite” la quale si occupa di richiamare tutti i test case di ogni classe di test (dei corrispettivi DAO):

Prima esecuzione:

E' stato rilevato una failure durante l'esecuzione del metodo “doRetrieveBarAll” della classe “ProductDAO”;

l'errore è stato risolto ed è stato eseguito nuovamente il test.



Seconda esecuzione:

progetto_js - mythos/src/test/unitTest/TestSuite.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- DAO
- entity
- manager
- service
- testIntegrationTest
 - AddProductToCartTest.java
 - EditProductTest.java
 - EditReservationPriceTest.java
 - EditReservationTest.java
 - LoginTest.java
 - ReserveTableTest.java
- testUnitTest
 - CartTest.java
 - DetailDAOTest.java
 - OrderDAOTest.java
 - OrderedProductDAOTest.java
 - ProductDAOTest.java
 - ReservationOptionDAOTest.java
 - RoleDAOTest.java
 - TableDAOTest.java
 - TestSuite.java
 - UserDAOTest.java
- utility
 - ConnectionPool.java
 - ManageReceipt.java
 - ParametersCheck.java
 - PrintToPrinter.java
- Web App Libraries

Git Repositories

- mythos [master] - C:\Users\Vincenzo\Documents

TestSuite.java

```
1 package test.unitTest;
2
3 import org.junit.platform.runner.JUnitPlatform;
4
5
6
7 @RunWith(JUnitPlatform.class)
8 @ScaleOf(1000)
9 class TestSuite {
10     // ...
11 }
```

JUnit

Finished after 6.412 seconds

Runs: 43/43 Errors: 0 Failures: 0

testUnitTest.TestSuite [Runner: JUnit 4] (5.316 s)

- JUnit Jupiter (5.287 s)
 - CartTest (0.217 s)
 - testAddProductToCart1() (0.074 s)
 - testAddProductToCart2() (0.009 s)
 - testRemoveProductFromCart() (0.007 s)
 - testAddDetailToCart() (0.022 s)
 - testContainsDetail() (0.021 s)
 - testRemoveDetailFromCart() (0.083 s)
 - DetailDAOTest (3.026 s)
 - testDoRetrieveById() (2.908 s)
 - testDoRetrieveAll() (0.118 s)
 - OrderDAOTest (0.561 s)
 - testDoClearTable() (0.120 s)
 - testMakeNewOrderByUserAndTable() (0.028 s)
 - testDoRetrieveMaxIdByUser() (0.014 s)
 - testDoRetrieveById() (0.054 s)
 - testDoUpdate() (0.030 s)
 - testDoRetrieveByTableNumber() (0.042 s)
 - testDoRetrieveExtraPaymentByTable() (0.075 s)
 - testDoDeleteByOrder() (0.198 s)
 - OrderedProductDAOTest (0.737 s)
 - testDoClearTable() (0.067 s)
 - testDoSave() (0.108 s)
 - testDoRetrieveAll() (0.085 s)
 - testDoRetrieveByOrder() (0.050 s)
 - testDoRetrieveTableTot1() (0.146 s)
 - testDoRetrieveTableTot2() (0.278 s)

progetto_js - mythos/src/test/unitTest/TestSuite.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- DAO
- entity
- manager
- service
- testIntegrationTest
 - AddProductToCartTest.java
 - EditProductTest.java
 - EditReservationPriceTest.java
 - EditReservationTest.java
 - LoginTest.java
 - ReserveTableTest.java
- testUnitTest
 - CartTest.java
 - DetailDAOTest.java
 - OrderDAOTest.java
 - OrderedProductDAOTest.java
 - ProductDAOTest.java
 - ReservationOptionDAOTest.java
 - RoleDAOTest.java
 - TableDAOTest.java
 - TestSuite.java
 - UserDAOTest.java
- utility
 - ConnectionPool.java
 - ManageReceipt.java
 - ParametersCheck.java
 - PrintToPrinter.java
- Web App Libraries

Git Repositories

- mythos [master] - C:\Users\Vincenzo\Documents

TestSuite.java

```
1 package test.unitTest;
2
3 import org.junit.platform.runner.JUnitPlatform;
4
5
6
7 @RunWith(JUnitPlatform.class)
8 @ScaleOf(1000)
9 class TestSuite {
10     // ...
11 }
```

JUnit

Finished after 6.412 seconds

Runs: 43/43 Errors: 0 Failures: 0

testUnitTest.TestSuite [Runner: JUnit 4] (5.316 s)

- JUnit Jupiter (5.287 s)
 - ProductDAOTest (0.275 s)
 - testDoRetrieveAll() (0.051 s)
 - testDoRetrieveGalleyAll() (0.068 s)
 - testDoRetrieveBarAll() (0.033 s)
 - testDoRetrieveById() (0.023 s)
 - testDoUpdate() (0.034 s)
 - testDoCheckAvailabilityProduct() (0.015 s)
 - testDoRestoreQuantity() (0.051 s)
 - ReservationOptionDAOTest (0.051 s)
 - testDoRetrieveConfig() (0.013 s)
 - testDoUpdate() (0.038 s)
 - RoleDAOTest (0.030 s)
 - testRetrieveRoleById() (0.029 s)
 - TableDAOTest (0.340 s)
 - testDoClear() (0.065 s)
 - testDoUpdate() (0.031 s)
 - testDoRetrieveAll() (0.079 s)
 - testDoRetrieveNotAssigned() (0.034 s)
 - testDoRetrieveById() (0.018 s)
 - testDoRetrieveAssigned() (0.023 s)
 - testDoSetNotAssignedById() (0.063 s)
 - testDoEditReservation() (0.027 s)
 - UserDAOTest (0.050 s)
 - testDoRetrieveByUsernamePassword() (0.017 s)
 - testDoRetrieveByUsername() (0.033 s)

2. Testing di integrazione

I test di integrazione sono stati realizzati sui control che si occupano dell'invocazione dei servizi specificati nel category partition.

I control che sono stati testati sono i seguenti:

- AddProductToCart: Control che si occupa di far inserire dal manager un prodotto nel carrello
- EditProduct: Control che si occupa di gestire la richiesta di modifica di un prodotto nel catalogo.
- EditReservationPrice: Control che si occupa di gestire la richiesta di aggiornamento dei prezzi delle prenotazioni(standard e luxus)
- EditReservation: Control che si occupa di gestire la richiesta di aggiornamento di una prenotazione esistente.
- LoginTest: che si occupa di gestire la richiesta di autenticazione da parte di un utente
- ReserveTable: che si occupa di gestire la richiesta di registrazione di una nuova prenotazione

Per ognuno di questi è stata generata una pagina Test che prevedeva l'invio di parametri ai control per verificarne il corretto funzionamento.

Qui vengono riportati i risultati dei test per ogni pagina.

AddProductToCartTest

The screenshot displays the Eclipse IDE interface. The Package Explorer on the left shows the project structure, with the 'control' package expanded. The main editor shows the 'AddProductToCartTest.java' file, which contains the following code:

```
26 private MockHttpSession session;  
27 private Cart c;  
28 private User u;  
29 private Role r;  
30  
31 @BeforeEach  
32 public void setUp() throws ServletException {  
33  
34     servlet = new AddProductToCart();  
35     request = new MockHttpServletRequest();  
36     response = new MockHttpServletResponse();  
37     session = new MockHttpSession();  
38     c = new Cart();  
39     u = new User();  
40     r = new Role();  
41     u.setRole(r);  
42  
43     r.setIdRole(1);  
44  
45     session.setAttribute("userLogged", u);  
46     session.setAttribute("cartList", c);  
47     request.setSession(session);  
48  
49     request.setParameter("id", "1");  
50 }
```

The JUnit test results are shown at the bottom of the IDE. The test suite 'AddProductToCartTest' (Runner: JUnit 5) (1,531 s) passed all tests. The results are as follows:

Test Case	Duration (s)
test_caseOne()	0.282
test_caseTwo()	1.180
test_caseFinal()	0.031
test_caseThree()	0.038

EditProductTest

The screenshot shows the Eclipse IDE with the `EditProductTest.java` file open. The file is part of the `mythos` project, specifically in the `src` directory under `test/integrationTest`. The code defines a `test_caseFour()` method that tests the `editProduct` method of the `EditProduct` class. The test sets parameters for price, warehouse, galley, and bar, and asserts that the response is "Campo prezzo non valido".

```
108 public void test_caseFour() throws ServletException, IOException {
109     request.setParameter("price", "90");
110     request.setParameter("warehouse", "");
111     request.setParameter("galley", "");
112     request.setParameter("bar", "");
113
114     String result="";
115     try {
116         servlet.doGet(request, response);
117     } catch (IllegalArgumentException e) {
118         result=e.getMessage();
119     }
120
121     assertEquals("Campo prezzo non valido", result);
122 }
```

The test results show that the test passed successfully. The test results are as follows:

Test Case	Duration (s)
test_caseFive()	0.286 s
test_caseFour()	0.013 s
test_caseNine()	0.013 s
test_caseOne()	0.008 s
test_caseSix()	0.009 s
test_caseTen()	1.213 s
test_caseTwo()	0.009 s
test_caseEight()	0.008 s
test_caseSeven()	0.009 s
test_caseThree()	0.045 s

EditReservationPriceTest

The screenshot shows the Eclipse IDE with the `EditReservationPriceTest.java` file open. The file is part of the `mythos` project, specifically in the `src` directory under `test/integrationTest`. The code defines a `test_caseFive()` method that tests the `editReservationPrice` method of the `EditReservationPrice` class. The test sets parameters for price, warehouse, galley, and bar, and asserts that the response is "Campo prezzo non valido".

```
108 public void test_caseFive() throws ServletException, IOException {
109     request.setParameter("price", "90");
110     request.setParameter("warehouse", "");
111     request.setParameter("galley", "");
112     request.setParameter("bar", "");
113
114     String result="";
115     try {
116         servlet.doGet(request, response);
117     } catch (IllegalArgumentException e) {
118         result=e.getMessage();
119     }
120
121     assertEquals("Campo prezzo non valido", result);
122 }
```

The test results show that the test passed successfully. The test results are as follows:

Test Case	Duration (s)
test_caseFive()	0.323 s
test_caseFour()	0.008 s
test_caseOne()	0.012 s
test_caseSix()	0.009 s
test_caseTwo()	0.008 s
test_caseSeven()	1.005 s
test_caseThree()	0.040 s

EditReservationTest

The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays the project structure, with the 'src' folder expanded. The main editor shows the 'EditReservationTest.java' file, which contains two test methods: 'test_caseOne()' and 'test_caseTwo()'. The 'test_caseOne()' method is currently selected and highlighted. The JUnit console at the bottom shows the test results, indicating that all tests passed successfully. The console output includes the following details:

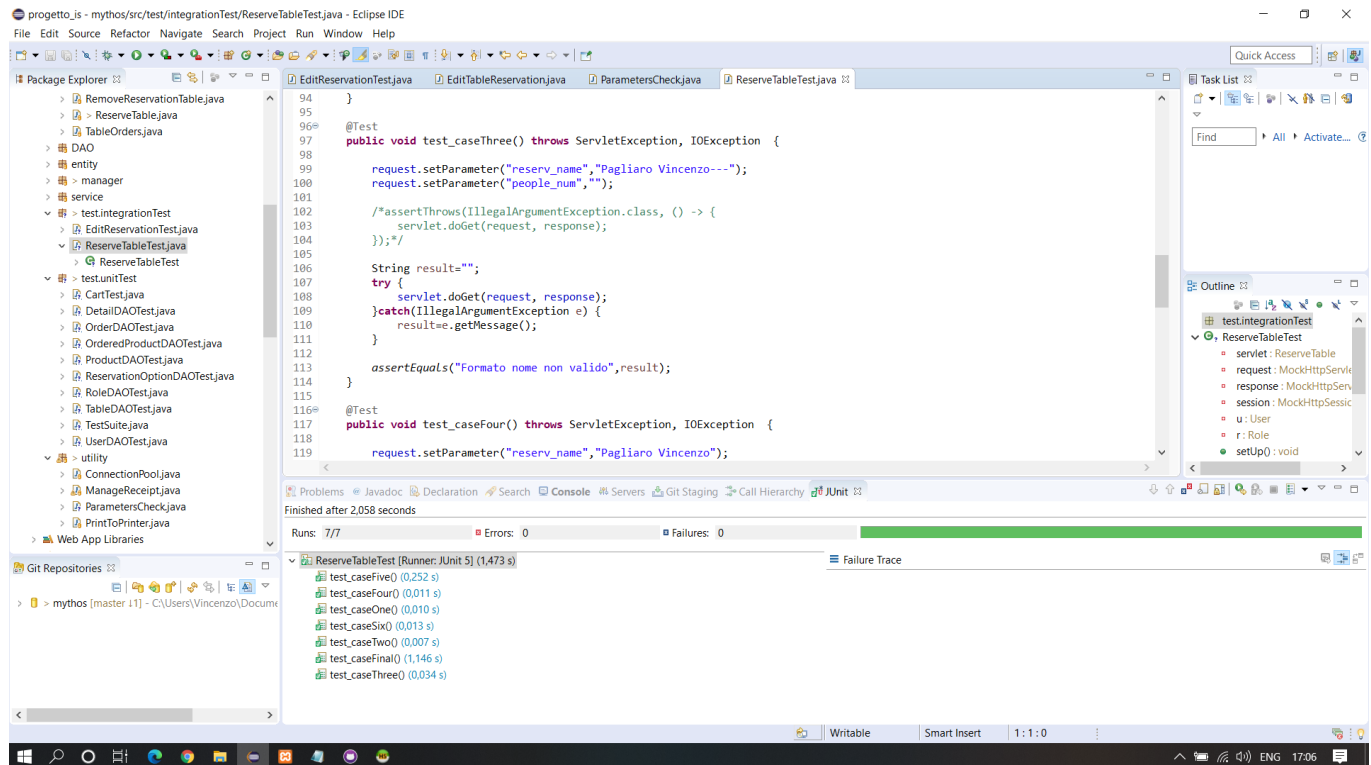
- Runs: 7/7
- Errors: 0
- Failures: 0
- Test Results:
 - test_caseFive() (0.272 s)
 - test_caseFour() (0.014 s)
 - test_caseOne() (0.011 s)
 - test_caseSix() (0.008 s)
 - test_caseTwo() (0.008 s)
 - test_caseFinal() (1.202 s)
 - test_caseThree() (0.040 s)

LoginTestTest

The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays the project structure, with the 'src' folder expanded. The main editor shows the 'LoginTestTest.java' file, which contains two test methods: 'test_caseFour()' and 'test_caseFive()'. The 'test_caseFour()' method is currently selected and highlighted. The JUnit console at the bottom shows the test results, indicating that all tests passed successfully. The console output includes the following details:

- Runs: 5/5
- Errors: 0
- Failures: 0
- Test Results:
 - test_caseFive() (1.370 s)
 - test_caseFour() (0.013 s)
 - test_caseOne() (0.007 s)
 - test_caseTwo() (0.012 s)
 - test_caseThree() (0.030 s)

ReserveTableTest



3. Testing di sistema

Il testing di sistema è stato realizzato utilizzando il framework **Selenium** installato come plug-in al browser(Google Chrome nel nostro caso).

Il testing con Selenium è basato sui dati di input eseguiti per il test di integrazione, che sono gli stessi specificati nel test case.

Inoltre, sono stati registrati dei test totali che mostrano, per ogni tipo di utente, tutte le operazioni che possono svolgere con l'applicativo.