**Università degli Studi di Salerno**

Simone Cristiano 0512105070

Mario Offertucci 0512105184

Valeria Santarpia 0512105232

Smart Restaurant  
Test Summary Report

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autore** |
| 18/02/2020 | 1.0 | Introduzione e casi di test server | Valeria Santarpia |
| 18/02/2020 | 2.0 | Aggiunti casi di test client | Mario Offertucci |

# 

# 1. Introduzione

Il presente documento ha lo scopo di riportare i risultati dell’esecuzione dei processi di testing effettuati sul sistema Smart Restaurant dopo il completamento della fase di implementazione. In particolare, vengono analizzate le attività di testing pianificate prima della realizzazione del sistema, confrontandole con quelle effettivamente eseguite e con i risultati ottenuti.

# 2. Riepilogo testing

Così come descritto dal documento Test Plan, “TP\_SmartRestaurant”, le attività di testing che sono state pianificate durante la fase di progettazione del sistema si dividono in tre diverse categorie: test d’unità, test d’integrazione e test di sistema. Ognuna di queste, focalizzandosi su diversi aspetti del sistema e servendosi di approcci differenti, ha l’obiettivo di verificare che il comportamento del sistema rispetto quello atteso e pianificato durante lo sviluppo.

In particolare, sono state testate:

* le classi dei package entity e query del Server con l’utilizzo di JUnit;
* le classi dei package entity e server del client con l’utilizzo di JUnit;
* il sistema con Espresso.

Combinando tra loro questi diversi tipi di testing, si è tentato di garantire un rilevamento degli errori che risultasse quanto più completo possibile.

2.1 Testing d’unità

La fase di testing di unità prevede di testare in maniera isolata le diverse componenti del sistema realizzato. Per fare questo, si è deciso di testare i metodi delle diverse classi che compongono i vari sottosistemi. Così come viene specificato dal Test Plan, l’approccio scelto per questo test è di tipo black-box.

2.2 Testing d’Integrazione

La fase di testing di integrazione prevede di “mettere insieme” le componenti testate precedentemente tramite il test di unità per vedere come funzionano una volta integrate tra di loro.

2.3 Testing di Sistema

La fase del testing di sistema prevede di testare le diverse funzionalità implementate nel sistema. Trattandosi di un’app android il tool utilizzato è “Espresso” che, simulando un uso del sistema da parte di un utente, ci ha permesso di verificare la correttezza dell’esecuzione di tutte le funzionalità offerte. I risultati di questa fase sono riportati nel “TER\_SmartRestaurant”.

# 3. Risultati Unit Client

3.1 Risultati package entity

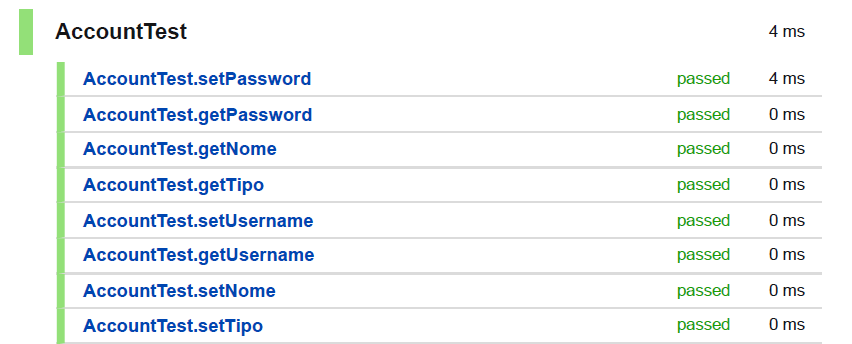
Di seguito sono riportati i risultati dei test di unità per i package entity. Le features testate sono:

* Account;
* Comanda;
* Notifica;
* Piatto;
* PiattoOrdinato;
* Tavolo;
* DbManager.

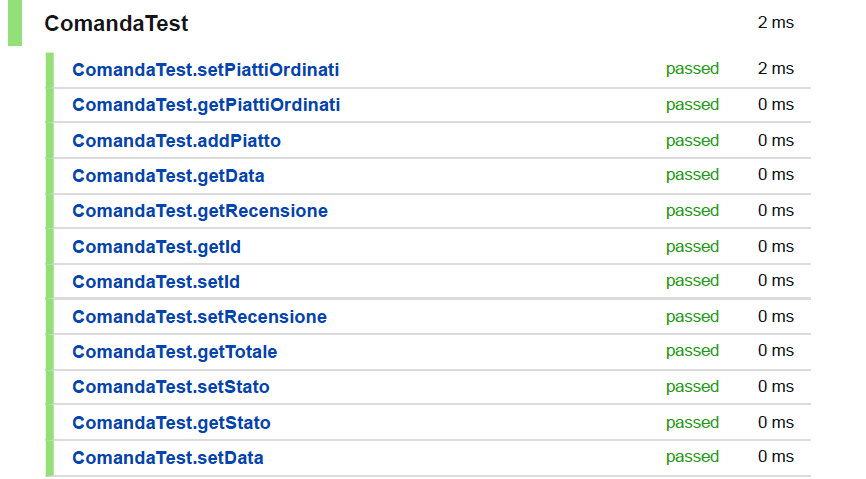
4.2 Panoramica dei risultati delle classi di test

In nessuna delle classi sono stati riscontrati error/failure. Di seguito sono riportati i risultati dell’esecuzione di ogni classe.

4.2.1 Classe Account



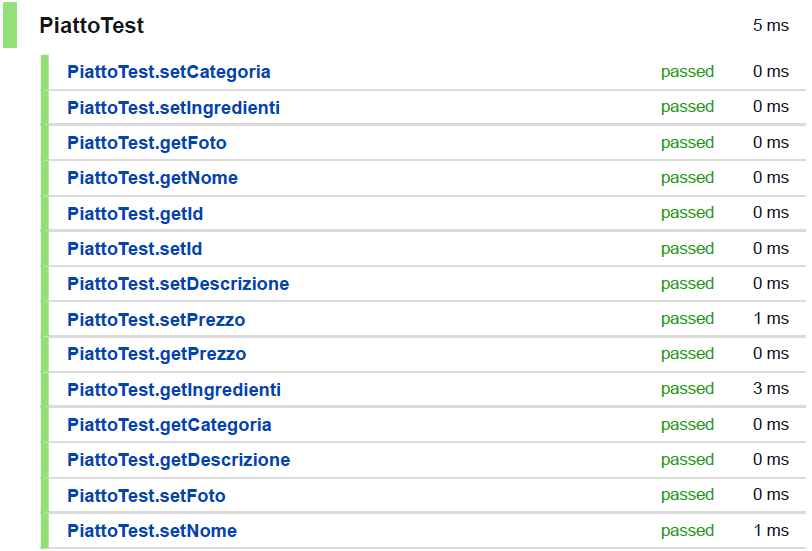
4.2.2 Classe Comanda



4.2.3 Classe Notifica



4.2.4 Classe Piatto



4.2.5 Classe PiattoOrdinato

Questo test è incluso nel test di ComandaTest perché ComandaTest utilizza PiattoOrdinato.

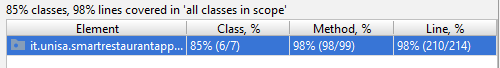
4.2.6 Classe Tavolo

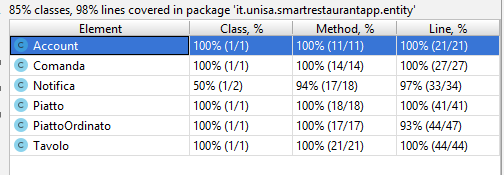


4.2.7 Classe DbManager

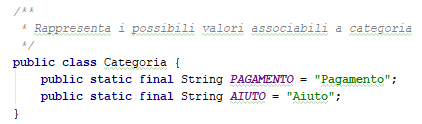


# 4.3 App coverage





La classe “Notifica” contiene una classe innestata che contiene semplicemente delle costanti per identificare lo stato della comanda. Nonostante questa classe sia stata utilizzata nel testing, il report non la conta.



# 5. Risultati Unit Server

5.1 Risultati package entity

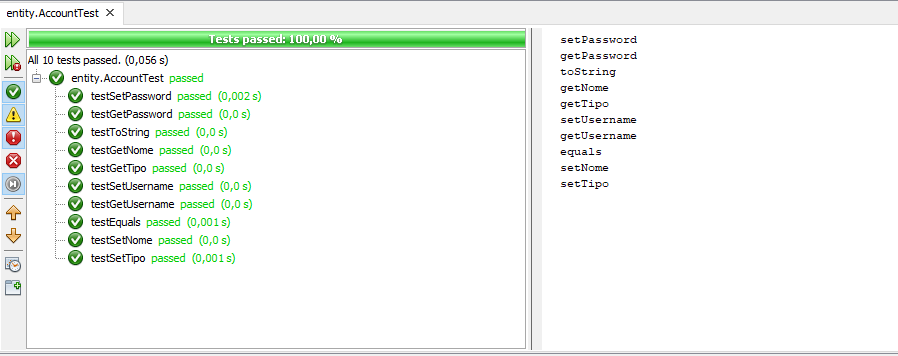
Di seguito sono riportati i risultati dei test di unità per i package entity. Le features testate sono:

* Account;
* Comanda;
* Notifica;
* Piatto;
* PiattoOrdinato;
* Tavolo.

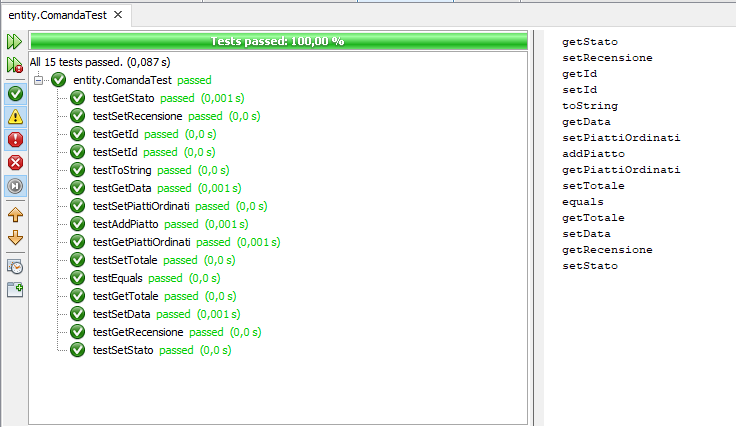
5.2 Panoramica dei risultati delle classi di test

In nessuna delle classi sono stati riscontrati error/failure. Di seguito sono riportati i risultati dell’esecuzione di ogni classe.

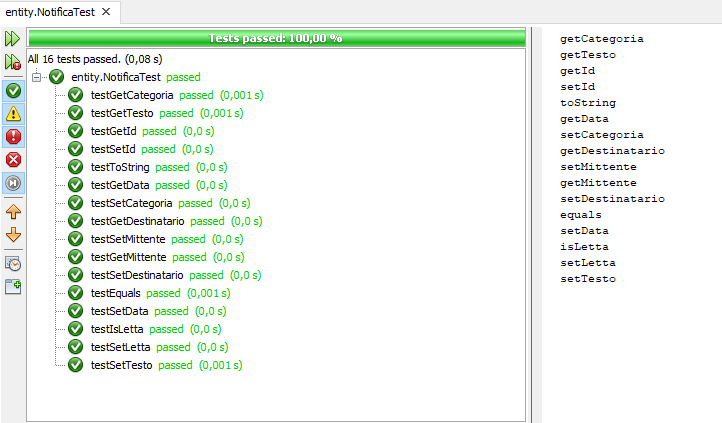
5.2.1 Classe Account



5.2.2 Classe Comanda



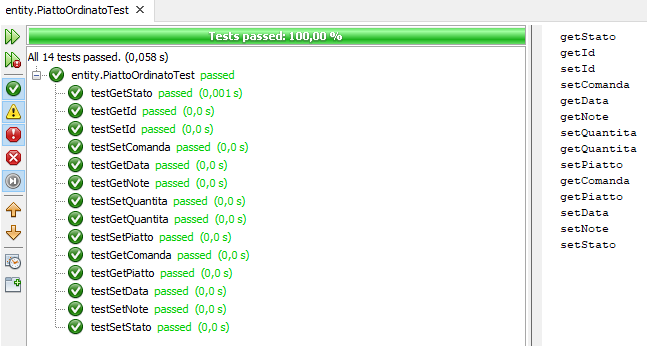
5.2.3 Classe Notifica



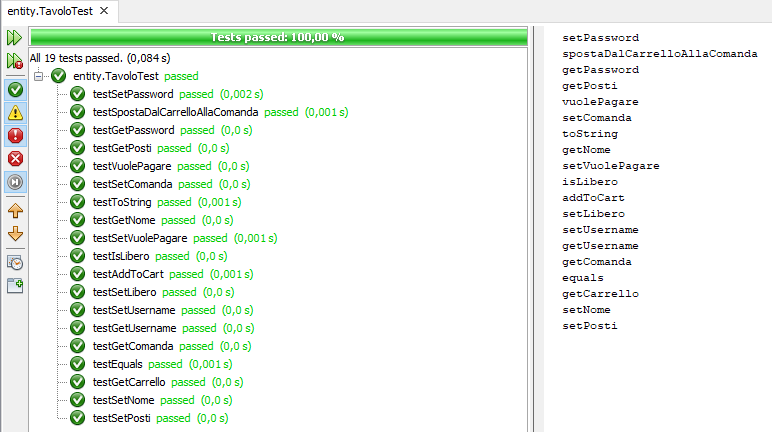
5.2.4 Classe Piatto



5.2.5 Classe PiattoOrdinato



5.2.6 Classe Tavolo



5.2 Risultati package query

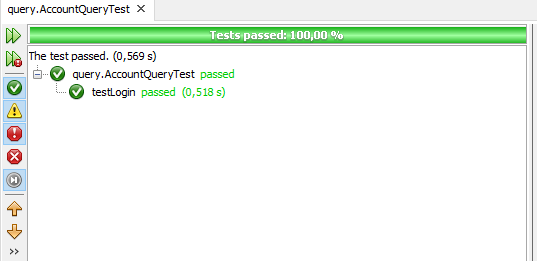
Di seguito sono riportati i risultati dei test di unità per i package entity. Le features testate sono:

* AccountQuery;
* ComandaQuery;
* NotificaQuery;
* PiattoQuery;
* PiattoOrdinatoQuery;
* TavoloQuery.

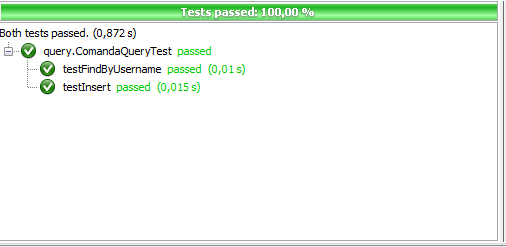
5.3 Panoramica dei risultati delle classi di test

In nessuna delle classi sono stati riscontrati error/failure. Di seguito sono riportati i risultati dell’esecuzione di ogni classe.

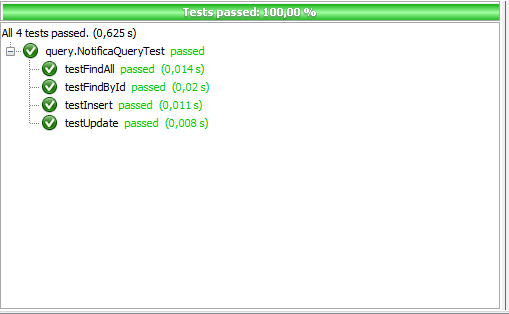
5.3.1 Classe AccountQuery



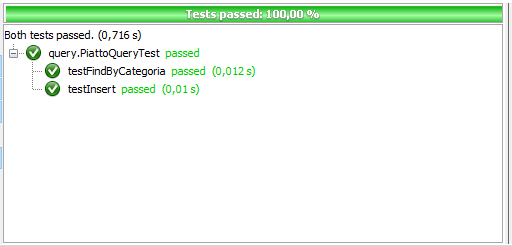
5.3.1 Classe ComandaQuery



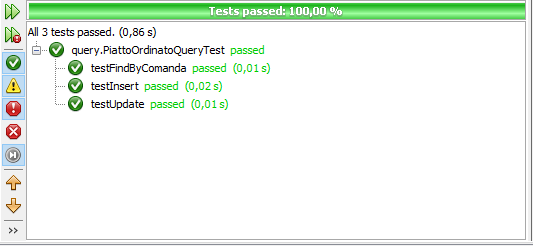
5.3.1 Classe NotificaQuery



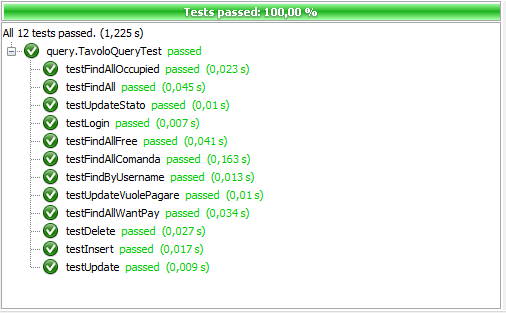
5.3.1 Classe PiattoQuery



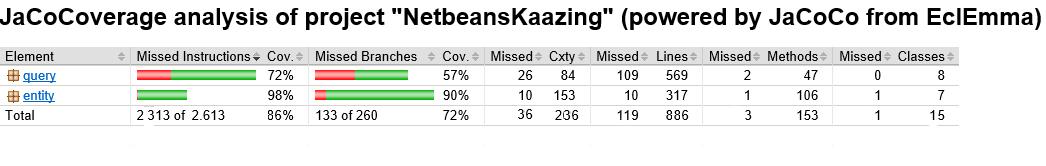
5.3.1 Classe PiattoOrdinatoQuery



5.3.1 Classe TavoloQuery



# 5.4 Server coverage



Nelle classi del package query non è stato possibile testare i branch relativi alla cattura delle eccezioni, poiché esse vengono lanciate solo in caso di problemi legati al codice che effettua la query.