Test Unit Report

Di:

Sara Guerriero 0512103956
Torre Francesca 0512102366

Indice degli argomenti:

1.0 Introduzione 3
1.1 Scopo del Test 3
1.2 Strumentazione per il Test di unità 3

2.0 Classi da testare per sottosistema 4

3. 0 Report dei Test 5
3.1 Unità Account 5
3.2 Unità Amministratore 8
3.3 Unità ServiziUtente 10

4.0 Conclusioni e Valutazione 13

1.0 Introduzione

Il testing di unità rappresenta la fase di testing in cui si assicura che le componenti sviluppate funzionino in isolamento. Questo documento ha il compito di identificare la strategia di testing di unità per il sistema. In particolare, saranno specificate le componenti da testare e il modo in cui il testing dovrà essere eseguito.

1.1 Scopo del Test

Lo scopo dei Test è quello di riportare il risultato dei test di unità programmati nel documento e stilare una statistica dei casi di success, faliure ed error.

1.2 Strumentazione per il Test di unità

I test saranno effettuati con jUnit integrato con Eclipse. Saranno riportati in questo documento i risultati della console di jUnit con il numero di faliure, succes ed error e l'eventuale errore apparso nella console di Eclipse se dovessero verificarsi error.

I test saranno effettuati sulle classi java e jsp.

I sottosistemi analizzati saranno:

- DriverManagerConnectionPool
- Sottosistemi Servizi Utente
- Sottosistema Amministratore
- Sottosistema Account

2.0 Classi da testare per sottosistema

Database:

- DriverManagerConnectionPool

Per il database Account:

- UserDataDao;

Per il database Amministratore:

- ProdottiDataDao;

Per il database ServiziUtente:

- OrderDataDao;

Sottosistema Account:

- UserData.java

Sottosistema Amministratore:

- ProdottiData.java

Sottosistema ServiziUtente:

- CarrelloData.java

3. 0 Report dei Test

3.1 Unità Account

Introduzione:

Il test sarà effettuato sui seguenti metodi:

- doSave();
- doDelete ();
- ReturnUser() ;
- SearchUser();

Input Funzionalità:

- doSave(UserData user);
- doDelete(UserData user);
- returnUser();
- searchUser(String email);

Output Funzionalità:

- DoSave(): **return true;**
  - test DoSave(); Success;
  - testdoSaveUtenteFallito(user) : Success;

- doDelete (): **return false;**
  - test doDelete (): Success;

- ReturnUser() : **return null;**
  - testReturnUser() Success;

- SearchUser():**return null;**

  - testsearchUser(): Success;

Oracolo:

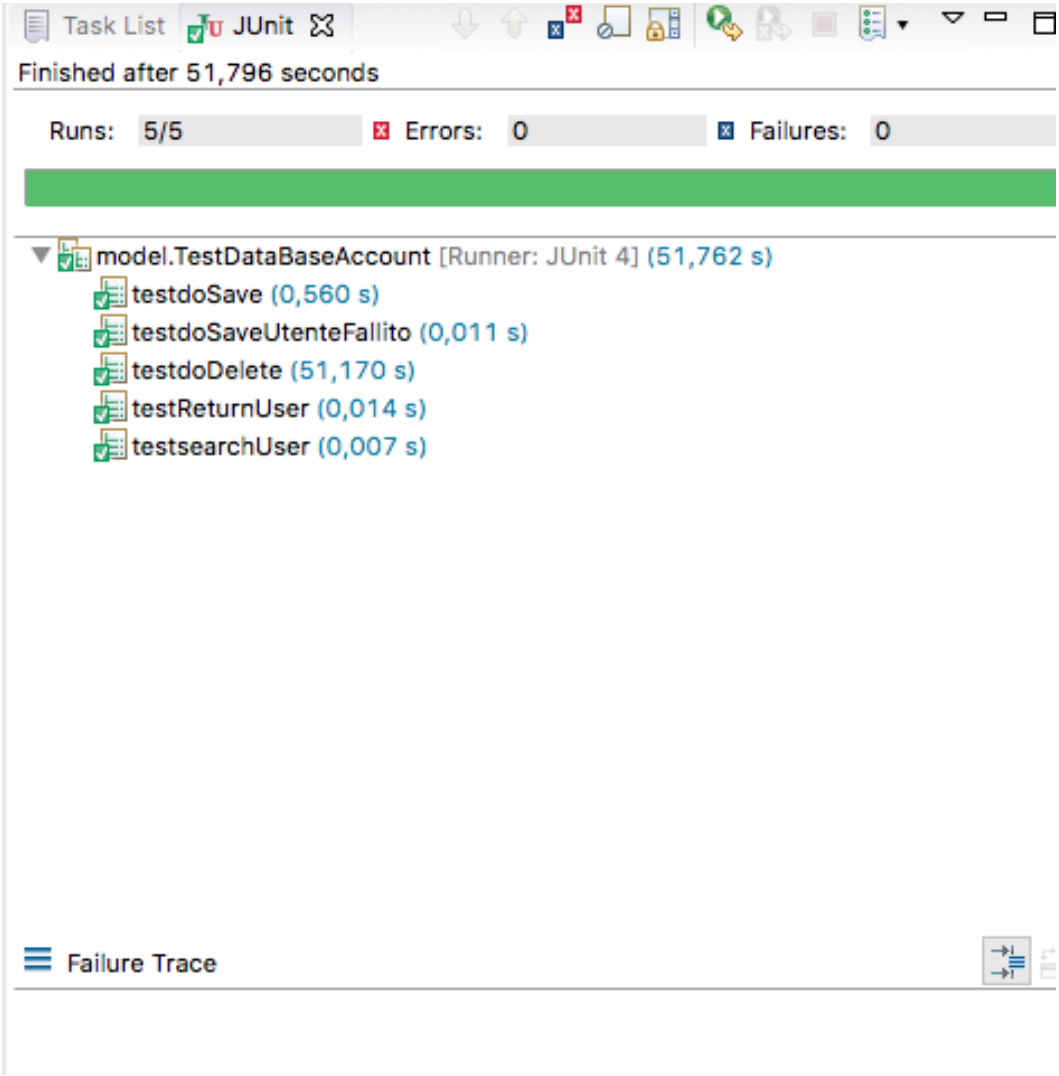
- doSave():Aggiunge un utente;
  - testdoSave(): Success;
  - testdoSaveUtenteFallito(): Success;

- doDelete () : Cancella un utente;

  - test doDelete (): Success;

- ReturnUser() : **Ottiene un ArrayList di Utenti**
    - testReturnUser() Success;
- SearchUser(): cerca un utente;
    - testsearchUser() success;

Test jUnit:



Razionale:

Sono stati eseguiti 5 Test. Tutti hanno avuto esito positivo quindi possiamo dire che il test d'unità per quando riguarda l'utenteè fallito.

3.2 Unità Amministratore

Introduzione:

Il test sarà effettuato sui seguenti metodi:

- doSave()
- doDelete()

- doRetrieveAll()
- doRetrieveByKey()
- doRetrieveByName()
- doRetrieveAllbySC()

Input Funzionalità:

- doSave (ProdottiData prodotti)
- doDelete(int code)
- doRetrieveAll(String order)
- doRetrieveByKey(int code)
- doRetrieveByName(String name)

Output Funzionalità:

- doSave(): **return Prodotti;**
  - testdoSave(); Success;
- doDelete() **return true;**
  - testDelete(); Success;
- doRetrieveAll() **return null;**
  - testdoRetrieveAll() Success;
- doRetrieveByKey()**return null;**
  - testdoRetrieveByKey(); Success;
- doRetrieveByName()**return null;**
  - testdoRetrieveByName(); Success;

Oracolo:

- doSave(): **aggiunge un nuovo prodotto;**
  - testdoSave(); Success;

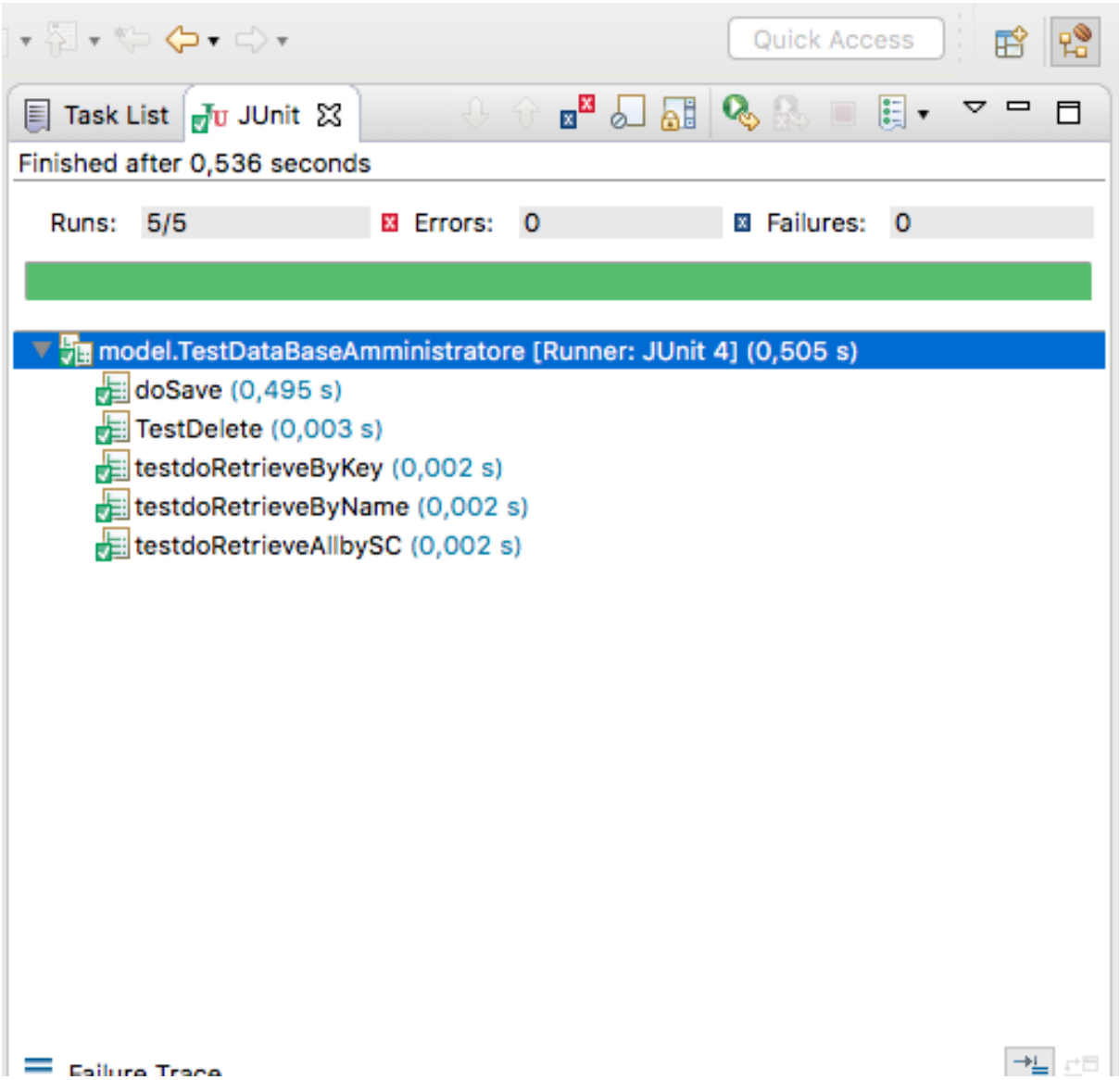
- doDelete() **cancella un prodotto;**
  - testDelete(); Success;

- doRetrieveAll() **restituisce tutti i prodotti;**
  - testdoRetrieveAll() Success;

- doRetrieveByKey() **restituisce i prodotti con un codice preciso;**
  - testdoRetrieveByKey(); Success;

- doRetrieveByName()**return i prodotti con un nome preciso;**
  - testdoRetrieveByName(); Success;

Test jUnit:



Razionale:

Sono stati eseguiti 5 Test. Tutti hanno avuto esito positivo quindi possiamo dire che il test d'unità per quando riguarda l'amministratore è fallito.

3.3 Unità ServiziUtente

Introduzione:

Il test sarà effettuato sui seguenti metodi:

- getOrder()
- getOrderByUser()

- getOrdiniByNumeroOrdine()

Input Funzionalità:

- getOrder()
- getOrderByUser(UserData user)
- getOrdiniByidOrdine(int nOrdine)

Output Funzionalità:

- getOrder() **return null;**
  - testgetOrder() Success;

- getOrderByUser() **return null;**

  - testgetOrderByUser() Success;

- getOrdiniByidOrdine() **return null;**
  - testgetOrdiniByidOrdine Success;

Oracolo:

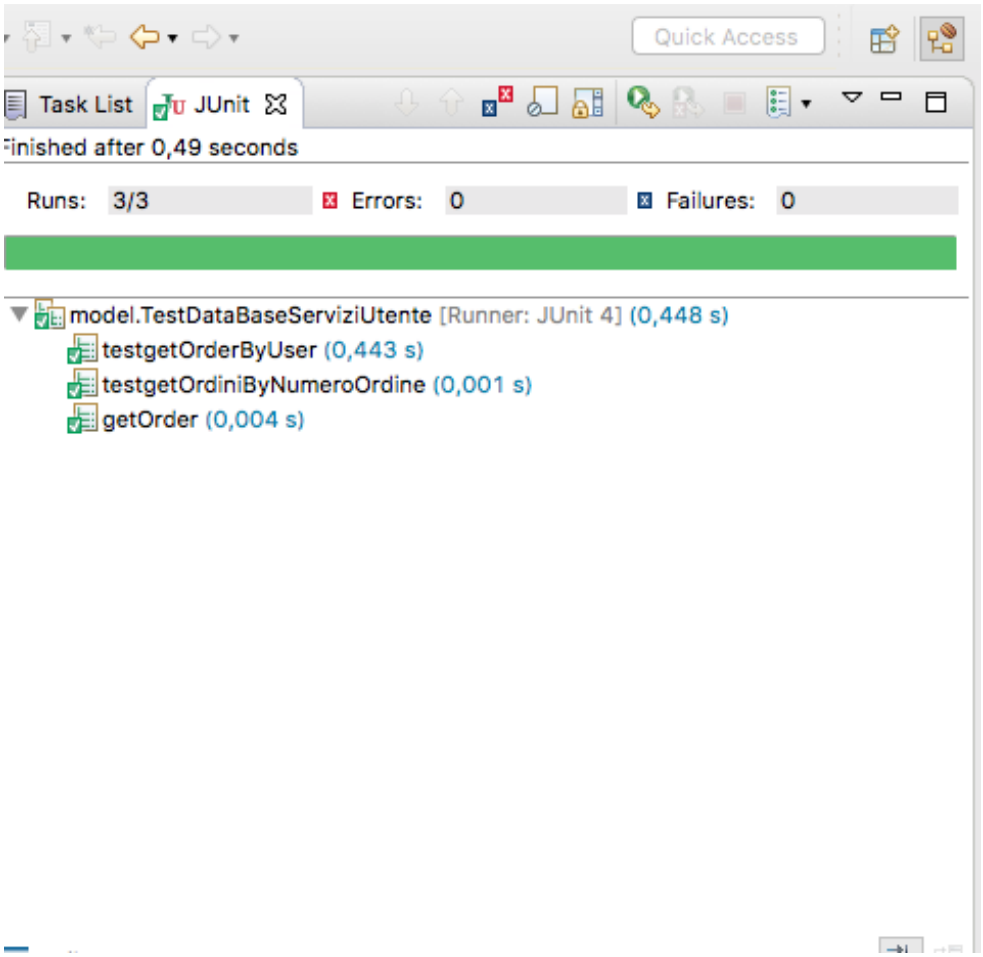
- getOrder() **restituisce tutti gli ordini;**
  - testgetOrder() Success;

- getOrderByUser() **restituisce gli ordini di un utente specifico ;**

  - testgetOrderByUser() Success;

- getOrdiniByidOrdine() **restituisce gli ordini in base all'id dell'ordine;**
    - testgetOrdiniByidOrdine Success;

Test jUnit:



Razionale:

Sono stati eseguiti 3 Test. Tutti hanno avuto esito positivo quindi possiamo dire che il test d'unità per quando riguarda i servizi utente è fallito.

4.0 Conclusioni e Valutazione

I test effettuati sulle unità database sono stati tutti negativi.