DODO.NET

TSR

Test Summory Report

|  |  |
| --- | --- |
| Riferimento |  |
| Versione | 1.0 |
| Data | 11/01/2022 |
| Destinatario | Prof.ssa F. Ferrucci |
| Presentato da | Alfonso Cuomo  Simone Farina |
| Approvato da |  |

RevisionHistory

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autori** |
| 11/01/2022 | 1.0 | Stesura documento | [Alfonso Cuomo] |

Sommario

1. Introduzione
2. Testing di unità

2.1 Classi testate

2.2 Package Bean

2.3 Package Model

2.4 Package Control

2.5 Riassunto del testing di Unità

3. Testing di Sistema

3.1 Test case eseguiti

3.2 Riassunto del testing del Sistema

4. Glossario

1. Introduzione

Nel seguente documento vengono riportati i risultati relativi all’esecuzione dei test case di unità delle componenti della piattaforma DodoNet. Per quanto riguarda i package model e controller è stato utilizzato JUnit, framework di unit testing per il linguaggio di programmazione Java. Il testing delle servlet, attraverso le richieste e risposte http, è effettuato mediante l’utilizzo del framework Mockito con l’ utilizzo di Spring per andare a simulare le richieste e risposte http.

2.Testing di Unità

Di seguito sono riportati i risultati del testing di Unità, effettuato con JUnit sulle classi Java appartenenti ai packages Controller e Model.

**2.1 Classi Testate**

Il testing di unità si è concentrato sulle seguenti classi:

• Package Bean:

1. UtenteBean

2. OrdineBean

3. ComposizioneBean

4. ProdottoBean

• Package Model:

1. ProdottoModelDm

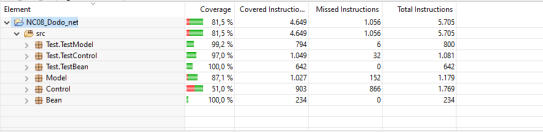
2. ComposizioneModelDm

3. OrdineModelDm

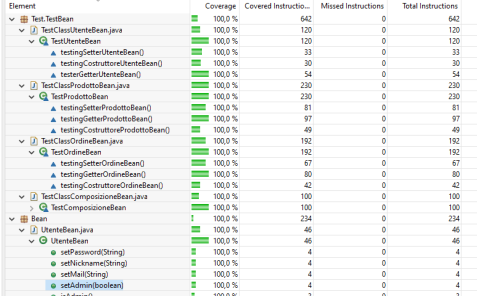
4. UtenteModelDm

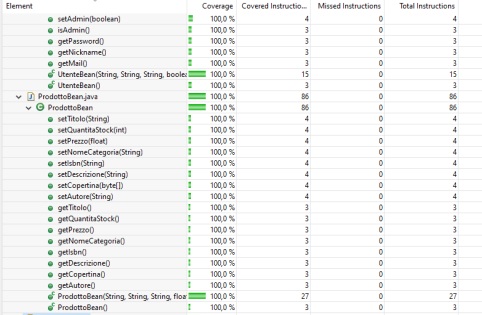
5.ConnectionSingleton

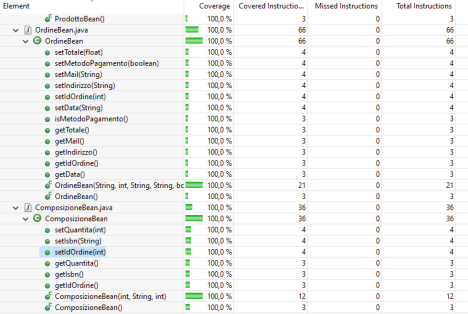
Viene riportata di seguito la branch coverage ottenuta testando l’intero progetto:



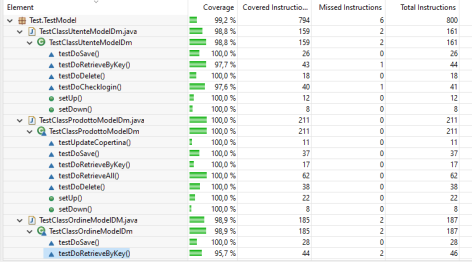
Viene riportata di seguito la branch coverage ottenuta testando il package Bean:

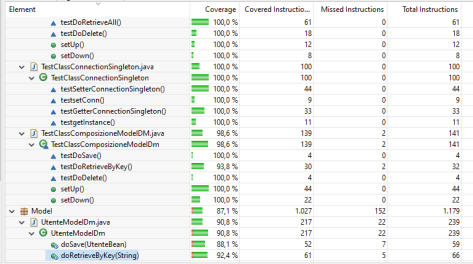


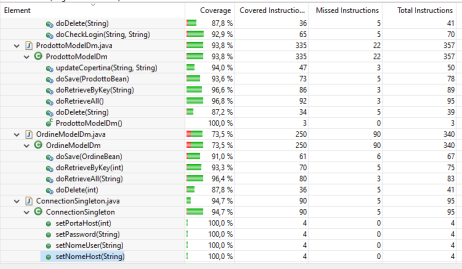


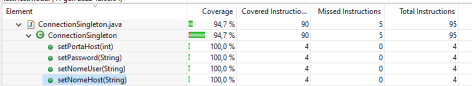


Viene riportata di seguito la branch coverage ottenuta testando il package Model:

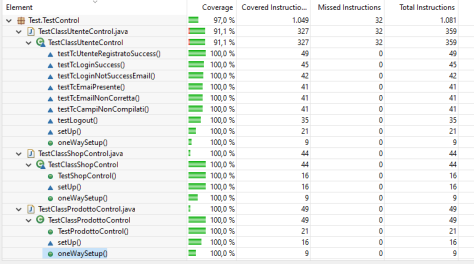


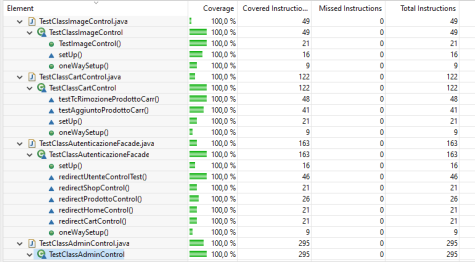


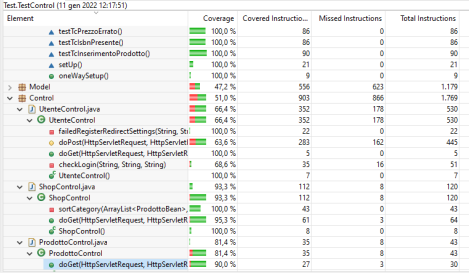


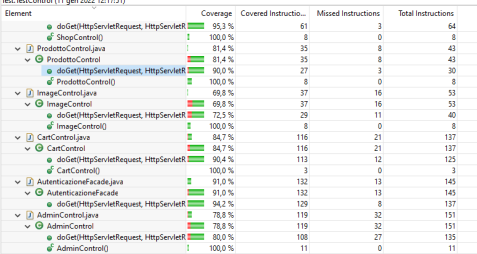


Viene riportata di seguito la branch coverage ottenuta testando il package Control:









**2.2 Riassunto del testing di Unità**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Componenti  testate | Errori trovati | Errori corretti | Componenti non testate |
| Model | 75 | 7 | 7 | 43 |
| Control | 105 | 11 | 11 | 72 |
| Bean | 50 | 0 | 0 | 0 |

3.Testing di Sistema

Di seguito vengono riportati i risultati ottenuti dal testing di sistema, effettuato con il plugin Katalon Recorder. Il testing di sistema viene effettuato seguendo i test case individuati all’interno del Test Case Specification Document.

**3.1 Test Case eseguiti**

Sono stati eseguiti i seguenti Test Case:

**TC\_1.1\_06**

**TC\_1.1\_07**

**TC\_1.1\_08**

**TC\_2.2\_04**

**TC\_2.2\_06**

**TC\_09.09\_05**

**TC\_09\_09\_03**

**TC\_09\_09\_07**

**3.2 Riassunto dei risultati del testing di sistema.**

Grazie al testing di sistema è stato possibile individuare un failure e un fault, e di conseguenza sono stati risolti i relativi faults. Tutte le failure individuate sono state riportate nel documento "Incident Report".

4.Glossario e definizioni

• RAD (Requirement Analysis Document): Documento ottenuto dalla fase di raccolta ed analisi dei requisiti;

• SDD (System Design Document): Documento ottenuto dalla fase di modellazione del sistema proposto;

• ODD (Object Design Document): Documento all’interno del quale si definiscono le scelte relative all’implementazione del sistema, scegliendo le possibili soluzioni ed analizzandone eventuali costi/benefici;

• TCD (Test Case Document): documento all’interno del quale vengono testate le singole funzionalità, osservando l’input fornito e l’output atteso, specificati prima dell’esecuzione del test;