

**T**est

**C**ase

**S**ummary **R**eport

|  |  |
| --- | --- |
| Riferimento |  |
| Versione | 0.1 |
| Data | 20/01/2020 |
| Destinatario | Prof.ssa F. Ferrucci |
| Presentato da | **H**ermann Senatore, **I**van Carmine Adamo, **L**orenzo Criscuolo, **O**razio Cesarano |
| Approvato da |  |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| DATA | Versione | Cambiamenti | Autori |
| 20/01/2020 | 0.1 | Prima Stesura | [tutti] |

Sommario

[1. Introduzione 3](#_Toc62134212)

[1.1. **Scopo** 3](#_Toc62134213)

[**1.2.** **Riferimenti** 3](#_Toc62134214)

[2. Risultati con Junit per il Model 4](#_Toc62134215)

[2.1. **Features testate** 4](#_Toc62134216)

[**2.2.** **Panoramica dei risultati del test delle classi** 4](#_Toc62134217)

[**2.2.1.** **EffettuaAs** 5](#_Toc62134218)

[**2.2.2.** **EffettuaAsManager** 5](#_Toc62134219)

[**2.2.3.** **EffettuaP** 5](#_Toc62134220)

[**2.2.4.** **EffettuapManager** 6](#_Toc62134221)

[**2.2.5.** **EmbeddedEffettuaP** 6](#_Toc62134222)

[**2.2.6.** **Swab** 6](#_Toc62134223)

[**2.2.7.** **SwabManager** 7](#_Toc62134224)

[**2.2.8.** **StatisticsManager** 7](#_Toc62134225)

[**2.2.9.** **EmbeddedPagina** 7](#_Toc62134226)

[**2.2.10.** **EmbeddedPaginaDiarioClinico** 8](#_Toc62134227)

[**2.2.11.** **EmbeddedRadiografia** 8](#_Toc62134228)

[**2.2.12.** **Pagina** 8](#_Toc62134229)

[**2.2.13.** **PaginaDiarioClinico** 9](#_Toc62134230)

[**2.2.14.** **PaginaDiarioClinicoManager** 9](#_Toc62134231)

[**2.2.15.** **PaginaManager** 9](#_Toc62134232)

[**2.2.16.** **Patient** 10](#_Toc62134233)

[**2.2.17.** **PatientManager** 10](#_Toc62134234)

[**2.2.18.** **Radiografia** 10](#_Toc62134235)

[**2.2.19.** **RadiografiaManager** 11](#_Toc62134236)

[**2.2.20.** **HealthWorker** 11](#_Toc62134237)

[**2.2.21.** **HealthWorkerManager** 11](#_Toc62134238)

[**2.2.22.** **Afferire** 12](#_Toc62134239)

[**2.2.23.** **AfferireManager** 12](#_Toc62134240)

[**2.2.24.** **EmbeddedAfferire** 12](#_Toc62134241)

[**2.2.25.** **Struttura** 13](#_Toc62134242)

[**2.2.26.** **StrutturaManager** 13](#_Toc62134243)

[3. Risultati di Junit e Mockito per il controller 13](#_Toc62134244)

[**3.1.** **Features Testate con Junit** 13](#_Toc62134245)

[**3.2.** **Panoramica dei risultati del test delle classi** 13](#_Toc62134246)

[**3.2.1.** **HealthWorkerController** 14](#_Toc62134247)

[**3.2.2.** **LoginController** 14](#_Toc62134248)

[**3.2.3.** **PatientController** 14](#_Toc62134249)

[**3.3.** **Features Testate con Mockito** 14](#_Toc62134250)

[**3.4.** **Panoramica dei risultati del test delle classi** 14](#_Toc62134251)

[**3.4.1.** **FaçadeController** 15](#_Toc62134252)

[4. Risultati di Katalon per la View 15](#_Toc62134253)

[**4.1.** **Features testate** 15](#_Toc62134254)

[**4.2.** **Panoramica feature testate** 15](#_Toc62134255)

[5. Resoconto Coverage 16](#_Toc62134256)

[6. Riepilogo del testing 16](#_Toc62134257)

[7. Glossario 16](#_Toc62134258)

# Introduzione

Il Test Summary Report è un documento contenente i risultati dell’esecuzione dei test case di unità delle varie componenti della piattaforma Hilo.

Le classi dei package model e controller sono testate con l’utilizzo di JUnit (nel controller si utilizza anche Mockito per alcune classi); successivamente, viene effettuato il testing di sistema con Katalon.

* 1. **Scopo**

Lo scopo di questo documento è quello di fornire una presentazione dei casi di test di unità per la piattaforma Hilo. I vari membri del team si sono impegnati nel verificare che le singole unità (classi e metodi) abbiano il comportamento atteso.

## **Riferimenti**

* NC\_04\_TCD

# Risultati con Junit per il Model

Di seguito sono riportati i risultati dei test di unità per le classi del package model.

* 1. **Features testate**

Con testing di unità per il package model si vuole effettuare il testing delle singole classi inserite e di seguito vengono riportate le classi testate:

* EffettuaAs;
* EffettuaAsManager;
* EffettuaP;
* EffettuapManager;
* EmbeddedEffettuaP;
* Swab;
* SwabManager;
* StatisticsManager;
* EmbeddedPagina;
* EmbeddedPaginaDiarioClinico;
* EmbeddedRadiografia;
* Pagina;
* PaginaDiarioClinico;
* PaginaDiarioClinicoManager;
* PaginaManager;
* Patient;
* PatientManager;
* Radiografia;
* RadiografiaManager;
* HealthWorker;
* HealthWorkerManager;
* Afferire;
* AfferireManager;
* EmbeddedAfferire;
* Struttura;
* StrutturaManager;

## **Panoramica dei risultati del test delle classi**

Per alcune delle classi testate sono stati riscontrati fault/failure che sono stati prontamente risolti e di seguito sono riportati i report dell’esecuzione dei test

### **EffettuaAs**

Immagine che contiene testo, monitor, screenshot, interni

Descrizione generata automaticamente

### **EffettuaAsManager**

Immagine che contiene testo

Descrizione generata automaticamente

### **EffettuaP**

Immagine che contiene testo, monitor, screenshot, argento

Descrizione generata automaticamente

### **EffettuapManager**

Immagine che contiene testo

Descrizione generata automaticamente

### **EmbeddedEffettuaP**

Immagine che contiene testo, monitor, screenshot, schermo

Descrizione generata automaticamente

### **Swab**

Immagine che contiene testo

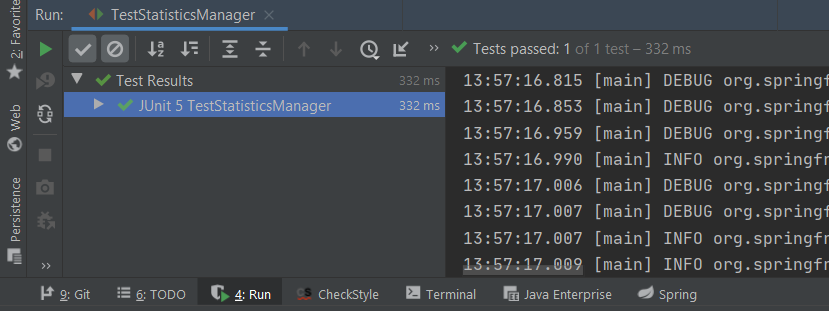
Descrizione generata automaticamente

### **SwabManager**

Immagine che contiene testo, screenshot, monitor, argento

Descrizione generata automaticamente

### **StatisticsManager**



### **EmbeddedPagina**

Immagine che contiene testo, monitor, screenshot, schermo

Descrizione generata automaticamente

### **EmbeddedPaginaDiarioClinico**

Immagine che contiene testo, interni, screenshot

Descrizione generata automaticamente

### **EmbeddedRadiografia**

Immagine che contiene testo

Descrizione generata automaticamente

### **Pagina**

Immagine che contiene testo

Descrizione generata automaticamente

### **PaginaDiarioClinico**

Immagine che contiene testo, screenshot, monitor, schermo

Descrizione generata automaticamente

### **PaginaDiarioClinicoManager**

Immagine che contiene testo, screenshot, monitor, nero

Descrizione generata automaticamente

### **PaginaManager**

Immagine che contiene testo, monitor, screenshot, argento

Descrizione generata automaticamente

### **Patient**

Immagine che contiene testo, screenshot, monitor, nero

Descrizione generata automaticamente

### **PatientManager**

Immagine che contiene testo

Descrizione generata automaticamente

### **Radiografia**

Immagine che contiene testo

Descrizione generata automaticamente

### **RadiografiaManager**

Immagine che contiene testo

Descrizione generata automaticamente

### **HealthWorker**

Immagine che contiene testo, monitor, screenshot

Descrizione generata automaticamente

### **HealthWorkerManager**

Immagine che contiene testo, screenshot, monitor, nero

Descrizione generata automaticamente

### **Afferire**

Immagine che contiene testo

Descrizione generata automaticamente

### **AfferireManager**

Immagine che contiene testo, screenshot, schermo

Descrizione generata automaticamente

### **EmbeddedAfferire**

Immagine che contiene testo

Descrizione generata automaticamente

### **Struttura**

Immagine che contiene testo

Descrizione generata automaticamente

### **StrutturaManager**

Immagine che contiene testo, screenshot, monitor, argento

Descrizione generata automaticamente

# Risultati di Junit e Mockito per il controller

Di seguito sono riportati i test di integrazione per le classi per il package controller:

## **Features Testate con Junit**

Con testing di unità per il package controller si vuole effettuare il testing delle singole classi inserite. Sono state testate le feature principali per il sistema. Di seguito sono riportate le classi che sono state testate con Junit:

* HealthWorkerController;
* LoginController;
* PatientController;

## **Panoramica dei risultati del test delle classi**

Per alcune delle classi sono stati riscontrati dei fault/failure ma sono stati prontamente risolti per far rispettare alle classi il comportamento atteso

### **HealthWorkerController**

Immagine che contiene testo, screenshot, monitor, interni

Descrizione generata automaticamente

### **LoginController**

Immagine che contiene testo

Descrizione generata automaticamente

### **PatientController**

Immagine che contiene testo

Descrizione generata automaticamente

## **Features Testate con Mockito**

Si è visto necessario testare una classe con Mockito per via della sua gestione delle richieste HTTP e della sessione. Anche per questa classe sono stati riscontrati fault/failure che sono stati prontamente risolti per far rispettare alla classe il comportamento atteso. Di seguito la classe testata con Mockito:

* FaçadeController;

## **Panoramica dei risultati del test delle classi**

Per la classe testata sono stati riscontrati dei fault/failure che sono stati prontamente risolti e di seguito è riportato il report:

### **FaçadeController**

Immagine che contiene testo

Descrizione generata automaticamente

# Risultati di Katalon per la View

Di seguito sono riportati i test di sistema per le View contenenti JS contenute nella cartella “resources/templates”

## **Features testate**

Con il testing di sistema l’obiettivo è quello di andare a testare partendo dalle View (quindi “dall’alto”) per andare a coinvolgere anche le sue componenti del Controller e Model (quindi “verso il basso”). I test che sono stati eseguiti corrispondono a quelli definiti nel TCD, dal momento che definiscono dei test con un’interazione con l’utente massima, per cui più soggetta a problematiche dovute ad input errati o click sbagliati. I test di sistema sono stati eseguiti con Katalon e in particolare sono stati effettuati i seguenti test:

* TC\_1;
* TC\_2;
* TC\_3;
* TC\_4;

## **Panoramica feature testate**

Resoconto totale di tutti e 4 i test case specificati

Immagine che contiene testo

Descrizione generata automaticamente

# Resoconto Coverage

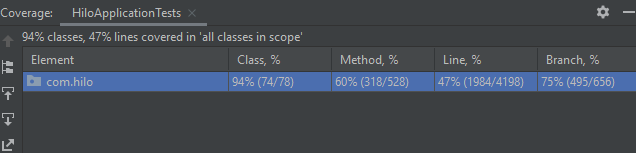


Immagine che contiene testo, screenshot, monitor, schermo

Descrizione generata automaticamente

# Riepilogo del testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Numero componenti testate | Numero errori trovati | Numero errori corretti | Numero componenti non testate |
| Model | 26 | 3 | 3 | 19 |
| Controller | 4 | 6 | 6 | 3 |
| View | 4 | 13 | 13 | 10 |

# Glossario

* “JUnit”, un framework per Java utilizzato al fine di automatizzare il testing;
* “Mockito”, un framework per Java utilizzato per emulare delle componenti di una classe al fine di eseguire il testing;
* “Katalon”, un’estensione web che permette di registrare le azioni da effettuare al fine di automatizzare il testing di sistema;
* “Feature”, funzionalità offerta dal sistema;
* “HTTP”, acronimo di “Hypertext Transfer Protocol” ed è il protocollo standard utilizzato per comunicare in rete;
* “sessione”, ovvero un insieme di richieste HTTP effettuate con un singolo utente in un determinato intervallo di tempo;
* “JS”, acronimo di “JavaScript” ovvero un linguaggio di scripting utilizzato per rendere dinamiche le pagine Web;
* “TCD”, acronimo di “Test Case Document” ovvero il documento nel quale sono presenti le specifiche per effettuare il testing e soprattutto la scelta delle componenti da testare;