TEST PLAN

Indice

[1. Introduzione 3](#_Toc536032336)

[2. Documenti correlati 3](#_Toc536032337)

[2.1. Relazioni con il Requirement Analysis System (RAD) 3](#_Toc536032338)

[2.2. Relazioni con il System Design Document (SDD) 3](#_Toc536032339)

[2.3. Relazioni con l’Object Design Document (ODD) 3](#_Toc536032340)

[3. Panoramica del sistema 3](#_Toc536032341)

[4. Testing Scope 4](#_Toc536032342)

[4.1. Funzionalità da testare 4](#_Toc536032343)

[4.2. Funzionalità da non testare 4](#_Toc536032344)

[5. Pass/Fail criteria 4](#_Toc536032345)

[6. Approccio 4](#_Toc536032346)

[6.1. Test di Unità 4](#_Toc536032347)

[6.2. Test d’Integrazione 5](#_Toc536032348)

[6.3. Test di Sistema 5](#_Toc536032349)

[7. Criteri di sospensione e ripresa 5](#_Toc536032350)

[7.1. Criteri di sospensione 5](#_Toc536032351)

[7.2. Criteri di ripresa 5](#_Toc536032352)

[8. Test Delirevables 5](#_Toc536032353)

[9. Materiale per il testing 6](#_Toc536032354)

[10. Test Case 6](#_Toc536032355)

# Introduzione

Lo scopo di questo documento è quello di pianificare l’attività di testing del sistema EsteticaMente identificando le componenti e le funzionalità da testare e le tipologie di testing che verranno adottate. Questo al fine di verificare se il comportamento atteso coincide con quello conseguente all’implementazione. Con questa attività si potranno così rilevare eventuali errori e quindi apportare correzioni in maniera tale da fornire all’utente un prodotto in linea con le sue aspettative.

In questa attività saranno testate le seguenti gestioni:

* Gestione Utente
* Gestione Autenticazione
* Gestione Registrazione
* Gestione Ordine
* Gestione Prodotto
* Gestione Pagamento

Tali gestioni sono quelle individuate nel documento dell’ODD e verranno testate le rispettive funzionalità.

Le attività di test sono strettamente legate ai documenti fino ad ora prodotti che ne costituiscono un punto di partenza.

# Documenti correlati

## Relazioni con il Requirement Analysis System (RAD)

La relazione del Test Plan con il RAD riguarda i requisiti funzionali e non funzionali del sistema. I test delle funzionalità prenderanno in considerazione le specifiche espresse nel RAD.

## Relazioni con il System Design Document (SDD)

Il testing delle componenti del sistema rispecchierà la suddivisione in sottosistemi riportata nel SDD.

## Relazioni con l’Object Design Document (ODD)

Il test di integrazione e di unità faranno riferimento alle class interfaces dichiarate nell’ODD.

# Panoramica del sistema

Come stabilito nel SDD la struttura di EsteticaMente è divisa secondo un’architettura three-tier: Presentation Layer, Application Layer, Storage. Il livello più alto interagisce con il livello applicativo che a sua volta si occuperà di eseguire le operazioni nel database. Abbiamo quindi identificato i principali sottosistemi:

* Gestione Utente
* Gestione Autenticazione
* Gestione Registrazione
* Gestione Ordine
* Gestione Prodotto
* Gestione Pagamento

Di questi sottosistemi andremo a testare le principali funzionalità.

# Testing Scope

## Funzionalità da testare

Di seguito andremo ad elencare le funzionalità, per ogni gestione, che andranno testate:

* Gestione Utente
* Modifica Utente
* Eliminazione Utente
* Gestione Autenticazione
* Login
* Gestione Registrazione
* Registrazione
* Gestione Prodotti
* Modifica Prodotto
* Eliminazione Prodotto
* Inserimento Prodotto
* Ricerca Prodotto
* Gestione Pagamento
* Effettua Pagamento
* Gestione Ordine
* Modifica stato Ordine
* Elimina Ordine

## Funzionalità da non testare

Le funzionalità da non testare sono principalmente le interfacce lato utente, interfacce hardware e logica di database nonché tutte le funzionalità precedentemente non menzionate non saranno testate poiché non sono considerate critiche per il nostro sistema. Non testiamo i bean, le modalità di visualizzazione

# Pass/Fail criteria

Un test si reputerà superato se l’output risultante dallo specifico test case sarà quello atteso e precedentemente determinato dal responsabile dei test. In sostanza il test sarà ritenuto superato quando effettivamente nella sua esecuzione non si riscontreranno errori, altrimenti si riscontrerà un insuccesso.

# Approccio

Il testing prevederà prima di tutto il test delle unità dei singoli componenti, così da assicurarsi della loro correttezza.  
In seguito sarà effettuato il test di integrazione che si concentrerà principalmente sul test delle interfacce delle unità di cui prima.  
Infine verrà svolto il test di sistema che verterà nel testare l’intero sistema assemblato con tutte le componenti.

## Test di Unità

Per il test di unità sarà utilizzato testing Black-Box, ovvero le componenti verranno testate senza conoscere la loro struttura interna, ma tenendo conto solo dell’input e dell’output.  
In particolare verrà utilizzato il metodo di Category Partition:

* nel sistema vengono individuate le funzioni che possono essere testate individualmente
* si identificano i parametri di ogni funzione
* per ogni parametro si definiscono delle categorie (proprietà o caratteristiche importanti)
* le categorie vengono divise in scelte, ovvero nei possibili valori che possono assumere
* vengono identificati i vincoli delle varie scelte
* vengono costruiti i Test Frames che contengono tutte le possibili combinazioni di scelte nelle categorie, ovvero tutti i possibili test case.

L’esecuzione dei test case sarà poi riportata nel documento di Test Execution Report ed eventuali errori verranno segnalati nel documento di Test Incident Report. Un report dell’attività di testing completo verrà riportato nel Test Summary Report.

## Test d’Integrazione

In questa fase si provvederà all’integrazione delle componenti del sistema, testandole insieme con strategia bottom-up così da testare prima e più a lungo i moduli base del nostro sistema, che spesso risultano essere i più critici, e quindi avere vantaggi sulla localizzazione degli errori. Ogni componente verrà integrata dopo essere stata testata con successo nella fase di test di unità: nell’ordine verranno integrati dapprima i bean poi model, poi control e le view poiché sono stati sviluppati seguendo quest’ordine.

## Test di Sistema

In questa fase andremo a testare che il sistema funzioni nel complesso e sia effettivamente pronto all’utilizzo. Verranno testate le funzionalità di maggiore importanza per l’utente del nostro sistema sempre con una tecnica Black-Box utilizzando il software Selenium.

# Criteri di sospensione e ripresa

## Criteri di sospensione

L’attività di testing sarà sospesa quando saranno stati effettuati tutti i test e risolti gli eventuali errori riscontrati, fino al raggiungimento di un buon compromesso con la qualità del prodotto finale ed entro i limiti di tempo prestabiliti per la consegna. L’attività di testing verrà sospesa nel momento in cui viene riscontrato un errore così da permettere la sua risoluzione.

## Criteri di ripresa

A valle di ogni correzione di bug ed errori riscontrati durante questa attività, i casi di test verranno nuovamente sottoposti al sistema onde verificarne l’effettiva risoluzione.

# Test Delirevables

Al termine di questa attività verranno stilati i seguenti documenti:

* Test Plan, ovvero questo stesso documento
* Unit Test Report, contenente i dettagli relativi ai test di unità
* Test Case Specification, contenente i dettagli dell’attività relativi ad ogni test case
* Test Execution Report, contenente l’esecuzione di tutti i test case
* Test Incident Report, contenente i report degli errori riscontrati
* Test Summary Report, contenente le conclusioni della fase di testing

# Materiale per il testing

Per eseguire l’attività di testing utilizzeremo un PC con connessione ad internet e configurato con il database locale del nostro sistema. Sarà inoltre necessario l’utilizzo di JUnit e Selenium.

# Test Case

Per i Test Case si faccia riferimento al documento di Test Case Specification del nostro sistema.