**Università degli Studi di Salerno**

**Corso di Ingegneria del Software**

**EdenJewelry TestPlanEdenJewelry**

**Versione 1.0**

****

Data: 16/01/2025

|  |  |
| --- | --- |
| Progetto: EdenJewelry | Versione: 1.0 |
| Documento: TestPlan\_EdenJewelry | Data: 16/01/2025 |

## Coordinatore del progetto:

|  |  |
| --- | --- |
| **Nome** | **Matricola** |
|  |  |
|  |  |

**Partecipanti:**

|  |  |
| --- | --- |
| **Nome** | **Matricola** |
| Gaetano D’Alessio | 0512110836 |
| Luigi Montuori | 0512117799 |
| Miriam Eva De Santis | 0512117121 |

**Revision History**

Luigi Montuori

**Scritto da:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autore** |
| 16/01/2025 | 1.0 | Prima stesura del file di Test Plan | Luigi Montuori |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |
| --- | --- | --- |
|  | Ingegneria del Software | Pagina 2 di 7 |

**Indice**

1. [INTRODUZIONE 3](#_TOC_250000)
   1. Paragrafo A 3
      1. Paragrafo A.X 3
      2. Paragrafo A.Y 3
   2. Paragrafo B 3

# Introduzione

In questo documento affronteremo la fase di test a cui il prodotto deve essere sottoposto.

Lo scopo del testing è quello di valutare il corretto funzionamento del sistema, tramite

alcune tecniche di test specifiche ed affrontate a lezione, e di scoprire valori di confine o operazioni che portano alla generazione di un comportamento anomalo o lontano dalle nostre aspettative. Tramite queste valutazioni, potremmo creare ed aggiungere delle soluzioni che contrastino questi errori.

# Riferimenti ad altri documenti

Illustriamo ora i riferimenti agli altri documenti sviluppati, durante le fasi di analisi e progettazione:

* Requirements Analysis Document (**RAD**): da questo documento ricaveremo i requisiti funzionali da testare;
* System Design Document (**SDD**): la visione di questo documento risulta essenziale per la suddivisione in sottosistemi in esso contenuta;
* Object Design Document (**ODD**): in quest’ultimo documento riprendiamo le dipendenze tra gli oggetti individuati.

# System overview

EdenJewelry è progettato per:

* Registrarsi ed effettuare l’accesso;
* Fornire un vasto catalogo di gioielli;
* Visualizzare i dettagli riguardo i propri gioielli preferiti;
* Acquistare gioielli;
* Visualizzare il proprio storico degli ordini;
* Permettere ai venditori di inserire e/o rimuovere gioielli;
* Assicurare la disponibilità dei prodotti.

Il sistema utilizza un'architettura a tre layer dove la trasparenza permette all’*upper* layer di accedere al *bottom* layer direttamente, per motivi di efficienza. Ogni strato raggruppa i moduli

concettualmente simili, senza ricorrere alla classica suddivisione “Presentation-Business-Data”.

I moduli da testare durante lo unit testing sono i seguenti:

* Gestione account
  + Autenticazione
  + AreaUtente
  + AreaVenditore
  + Registrazione
  + Utente
* Gestione prodotti
  + Homepage
  + Catalogo
  + Carrello
  + Prodotto
* Gestione ordini
  + Ordini
  + Acquisto

# Feature to be tested/not be tested

Le attività di testing interessano tutte le funzionalità che gli sviluppatori hanno reputato ad alta priorità.

Le funzionalità da testare sono:

* Gestione account
  + Autenticazione
    - Login (FR1)
  + Registrazione
    - Register (FR1)
  + AreaUtente
    - Visualizzazione area riservata all’utente
  + AreaVenditore
    - Visualizzazione area riservata al venditore
* Gestione prodotti
  + Homepage
    - Ricerca del prodotto
  + Catalogo
    - Visualizzazione del catalogo (FR2, FR3)
    - Visualizzazione stock dei prodotti (FR9)
    - Aggiunta e rimozione dei prodotti (FR10, FR11)
  + Carrello
    - Aggiunta e rimozione prodotti dal carrello
    - Modifica quantità di un prodotto del carrello
  + Prodotto
    - Selezione dei prodotti (FR4)
* Gestione ordini
  + Ordini
    - Visualizzazione dello storico (FR5)
    - Visualizzazione dello storico completo (FR12)
  + Acquisto
    - Acquisto dei prodotti (FR6)

Abbiamo escluso da questa lista:

* Il modulo Wishlist, in quanto avente priorità media è possibile svilupparlo e testarlo in un momento successivo;
* Le classi e le funzioni di check, non definiti nelle fasi di analisi (sono comunque ausiliare e trascurabili);

# Pass/Fail criteria

Allo scopo di stabilire se una certa componente testata ha superato o fallito il test, diremo:

* Ad una componente viene assegnato valore “fail” se il test non ha portato a comportamenti fuori dalle nostre aspettative. Dunque, il risultato ottenuto è anche in linea con quello atteso;
* Ad una componente viene assegnato valore “pass” se il test è riuscito a

compromettere il suo comportamento. Quindi, il risultato ottenuto è diverso da quello atteso;

# Approach

L’Approccio comprende

## Test Unitari

Verifica singoli componenti software.

## Test D’Integrazione

Validazione delle interazioni tra i vari componenti.

## Test di Sistema

Validazione e verifica del sistema complessivo.

# Suspension and resumption

## Criteri di sospensione

◆

## Criteri di ripresa

◆

# Testing materials

1. **Test cases**
   1. Accesso degli utenti
      * Parametri
        + email (email dell’utente)
        + password (password dell’utente)
      * Oggetti dell’ambienteù
        + database (contiene i dati degli utenti registrati)

|  |  |
| --- | --- |
| Categorie | Scelte |
| 1. Username presente | UP1. Email presente all’interno del database UP2. Email non presente |

|  |  |
| --- | --- |
| 2. Password presente | PP1. Password corretta (presente all’interno del database)  PP2. Password errata |

|  |  |
| --- | --- |
| TEST FRAME 1 | |
| Combinazioni | Oracolo |
| UP1, PP1 | Login effettuato. |
| UP1, PP2 | Messaggio “Password errata”. |
| UP2, PP1 | Messaggio “Email inesistente”. |
| UP2, PP2 | Messaggio “Email inesistente”. |

# Testing schedule

Il nostro testing schedule sarà strutturato in maniera tale da assicurare un processo di verifica completo.

Come già specificato nell’approccio, andremo a dividere la fase di testing in diverse parti.