Test Plan

## Introduzione

Il presente documento descrive il piano dei test relativi alle componenti software del progetto “Bollicine”.

L’obiettivo di questo documento è la formalizzazione del dettaglio dei test funzionali del sistema in oggetto.

I risultati dei test che verranno eseguiti saranno cruciali nell’analisi delle failure e delle loro cause (fault) per individuare dove bisognerà intervenire per correggere gli errori o apportare modifiche per il miglioramento dei vari sottosistemi.

Come linea guida seguiremo gli use case (con priorità 1) implementati nel RAD con l’aiuto del concetto di classe di equivalenza. Per ogni use case viene prodotto un **test case** dove vengono individuate le classi di equivalenza di ogni input. Per ogni input vi sarà una tabella che descriverà le scelte per ogni classe di equivalenza e, infine, vi sarà una tabella che riassumerà tutte le possibili combinazioni di errore (e non) per il test case. Per ogni test case vi sarà un **test case specification** che descriverà alcune delle combinazioni possibili per quello use case

## Relazione con altri documenti

* Requirement Analisis Document

I test che andremo a rappresentare sono frutto dei requisiti funzionali che abbiamo nel RAD con priorità 1 (in quanto sono stati implementati).

* Object Design Document

Nell’ODD possiamo individuare i componenti che successivamente indicheremo come “testati in modo unitario”

* System design document

Facciamo riferimento alla scomposizione in sottosistemi che abbiamo definito nell’SDD

## Panoramica del sistema

Il nostro sistema prevede 5 attori principali (vedi RAD/2.4.2 Diagramma utenti):

* Cliente: Il cliente è un utente registrato che ha confermato il proprio account
* Gestore catalogo: Il gestore del catalogo è colui che può inserire, modificare ed eliminare prodotti dal catalogo
* Gestore ordini: Il gestore degli ordini è colui che può visualizzare tutti gli ordini
* Admin: L’admin è colui che registra i gestori del catalogo e i gestori degli ordini
* Guest: il guest è un utente non registrato

Dalla scomposizione in sottosistemi che abbiamo nell’SDD individuiamo i seguenti sistemi nel layer di applicazione:

* Product Manager: definisce tutti i servizi riguardanti i prodotti
* Order Manager: definisce tutti i servizi inerenti agli ordini
* User Manager: definisce tutte le operazioni che riguardano gli account degli utenti e dei loro componenti associati

## Funzionalità da testare

Il testing funzionale riguarderà le seguenti funzionalità definite nei sottosistemi che le realizza:

* Product Manager
  + Inserimento di un prodotto nel catalogo
  + Modifica di un prodotto nel catalogo
* User manager
  + Registrazione del cliente
  + Login utente
  + Aggiunta di un nuovo indirizzo di spedizione in fase di acquisto
  + Aggiunta di una nuova carta di credito in fase di acquisto
  + Registrazione di un nuovo gestore del catalogo da parte dell’admin
  + Registrazione di un nuovo gestore degli ordine da parte dell’admin

Non saranno testate invece:

* Order Manager
* Altri servizi in quanto non vi sono le condizioni di possibile errore derivate da altri tipi di test, al di fuori del testing funzionale

## Criteri Pass/Failed

Determinando un insieme di possibili input che possono scovare errori nel sistema, il test ha **successo se il comportamento osservato è diverso dal comportamento specificato** nei requisiti funzionali.

Gli obiettivi posti durante questa fase verranno raggiunti se il test individuerà dei fault nel sistema.

Nel caso in cui verranno individuati dei fault, saranno analizzati i sottosistemi che riguardano i fault con l’applicazione di modifiche che riguardano il caso analizzato. Successivamente verrà iterato il test per verificare che le modifiche effettuate siano state efficaci e che non abbiano avuto un impatto negativo su altre componenti del sistema.

Il testing **fallirà se non si riscontreranno errori**.

## Approccio

Seguiremo le linee guida standard del testing che delinea tre attività principali:

* Trovare errori in una singola componente
* Testare le funzionalità che confluiscono dall’integrazione dei vari sottosistemi
* Testare l’intero sistema al fine di verificare le funzionalità globali che verranno sfruttate dal cliente finale

Strategie utilizzate per l’esecuzione delle fasi di cui sopra (Test di unità, Test d’integrazione e Test di sistema).

6.1 Testing di unità

Abbiamo la necessità di testare singole componenti al fine di evidenziare errori, basando la nostra attenzione sul comportamento di un singolo componente utilizzando modalità come testing black-box o white-box.

6.2 Testing di integrazione

A seguito del consolidamento dell’effettivo funzionamento delle singole componenti, possiamo integrarle nei sottosistemi in cui vengono utilizzate. Possiamo quindi tesare i sottosistemi in cui queste componenti, testate singolarmente, sono utilizzate.

La strategia utilizzata è quella del **bottom-up** in cui i sottosistemi più in basso nella gerarchia sono testati individualmente. I successivi sottosistemi da testare sono quelli che utilizzano i sottosistemi testati in precedenza. Si ripete questo processo finché non sono estati tutti i sottosistemi. I test driver sono usati per simulare le componenti dei layer più in alto che non sono stati ancora integrati.

6.3 Testing di sistema

Testando i possibili input degli utenti verifichiamo le funzionalità del sistema. La riduzione dei casi di test è attuata tramite l’adozione del category partition.

Per questa tipologia di test ci affidiamo al framework Selenium al fine di osservare il comportamento del sistema in presenza di combinazioni di input non ammesse.

## Sospensione e ripresa

* 1. Criteri di sospensione

Il test viene sospeso se almeno il 10% dei casi di test causano errore.

Se viene raggiunta la percentuale del 10% il team provvede a correggere i fault e a procedere con il testing di nuove funzionalità

* 1. Criteri di ripresa

Il test riprenderà se saranno aggiunte nuove funzionalità al sistema.

* 1. Criteri di terminazione

L’attività di testing termina quando tutti i casi di test somministrati riporteranno esito negativo.

## Materiale per il testing

Il testing verrà effettuato su un database mySql.

I framework utilizzati in ambienti java sono: JUnit, Mikito e Spring. Per il test di sistema verrà utilizzato Selenium.

## Test Cases

Andremo a testare le funzionalità dei sottosistemi individuati al punto 4, di seguito avremo una suddivisione dei test per ogni sottosistema.

Ad ogni funzionalità verrà associata una tabella in cui, per ogni parametro, saranno definite le relative categorie insieme alle possibili scelte.

Nella tabella verrà individuato il parametro, che indica il valore di input, con il formato che deve seguire.

* 1. Product manager

Per questo sottosistema verrà testato l’**inserimento del prodotto nel catalogo** e la **modifica del prodotto nel catalogo**. Entrambe le funzionalità prevedono la sottomissione dei dati del prodotto all’interno del form da parte del gestore del catalogo.

I dati del prodotto sono i seguenti:

* Nome: stringa con un numero di caratteri compreso tra 1 e 45
* Categoria: la categoria è rappresentata da una select che comprende le categorie possibili de prodotto. Il campo non sarà testato perché è impossibile non selezionare una categoria per via della select
* Descrizione: stringa con un numero di caratteri compreso tra 1 e 500
* Tipo: Stringa con un numero di caratteri compreso tra 1 e 45
* Annata: intero compreso tra 0 e 4000
* Prezzo: double con valore minimo 1
* Disponibilità: intero di valore minimo 0
* Foto: file

Ogni campo è obbligatorio

|  |  |
| --- | --- |
| Parametro | Nome |
| LN - Lunghezza | 1. < 1 or > 45 [error] 2. >= 1 and <= 45 [property LN\_OK] |

|  |  |
| --- | --- |
| Parametro | Descrizione |
| LD - Lunghezza | 1. < 1 or > 500 [error] 2. >= 1 and <= 500 [property LD\_OK] |

|  |  |
| --- | --- |
| Parametro | Tipo |
| LT - Lunghezza | 1. < 1 or > 45 [error] 2. >= 1 and <= 45 [property LT\_OK] |

|  |  |
| --- | --- |
| Parametro | Annata |
| ANN - Valore | 1. < 0 or > 4000 [error] 2. >= 0 and <= 4000 [property ANN\_OK] |

|  |  |
| --- | --- |
| Parametro | Prezzo |
| PR - Valore | 1. < 1 [error] 2. >= 1 [property PR\_OK] |

|  |  |
| --- | --- |
| Parametro | Disponibilità |
| DISP - Valore | 1. < 0 [error] 2. >= 0 [property DISP\_OK] |

|  |  |
| --- | --- |
| Parametro | Foto |
| IMG - Valore | 1. Se non caricata [error] 2. Se caricata [property IMG\_OK] |

|  |  |  |
| --- | --- | --- |
| Codice | Combinazione | Esito |
| TC\_IMP\_1:1 | LN1 | Error |
| TC\_IMP\_1:2 | LN2.LD1 | Error |
| TC\_IMP\_1:3 | LN2.LD2.LT1 | Error |
| TC\_IMP\_1:4 | LN2.LD2.LT2.ANN1 | Error |
| TC\_IMP\_1:4 | LN2.LD2.LT2.ANN2.PR1 | Error |
| TC\_IMP\_1:5 | LN2.LD2.LT2.ANN2.PR2.DISP1 | Error |
| TC\_IMP\_1:6 | LN2.LD2.LT2.ANN2.PR2.DISP2.IMG1 | Error |
| TC\_IMP\_1:7 | LN2.LD2.LT2.ANN2.PR2.DISP2.IMG2 | Inserimento/  Modifica |

* 1. User manager

Per questo sottosistema verranno testate:

* + Registrazione del cliente
  + Login utente
  + Aggiunta di un nuovo indirizzo di spedizione in fase di acquisto
  + Aggiunta di una nuova carta di credito in fase di acquisto
  + Registrazione di un nuovo gestore del catalogo da parte dell’admin
  + Registrazione di un nuovo gestore degli ordine da parte dell’admin

Utilizzeremo gli stessi input per la **registrazione del cliente**, **registrazione di un nuovo gestore catalogo da parte dell’admin** e **registrazione di un nuovo gestore del catalogo** **da parte dell’admin** dato che i dati inseriti dall’utente finale, a prescindere dal ruolo (admin o cliente), sono gli stessi in tutte e tre le funzionalità.

I dati sono gli stessi perché i dati che differiscono nei vari tipi di utente sono settati automaticamente dal sistema a seconda del form che viene compilato.

* + 1. Registrazione

I dati inseriti dal cliente e dall’admin sono i seguenti:

* Cognome: Stringa con un numero di caratteri compreso tra i 2 e i 30 caratteri
* Nome: Stringa con un numero di caratteri compreso tra i 2 e i 30 caratteri
* Email: Stringa che deve rispettare il formato email
* Password: Stringa con un numero di caratteri compreso tra 4 e 15

|  |  |
| --- | --- |
| Parametro | Cognome |
| LC - Lunghezza | 1. < 2 or > 30 [error] 2. >= 2 and <= 30 [property LC\_OK] |

|  |  |
| --- | --- |
| Parametro | Nome |
| LN - Lunghezza | 1. < 2 or > 30 [error] 2. >= 2 and <= 30 [property LN\_OK] |

|  |  |
| --- | --- |
| Parametro | Password |
| LPW - Lunghezza | 1. < 4 or > 15 [error] 2. >= 4 and <= 15 [property LN\_OK] |

|  |  |
| --- | --- |
| Parametro | Email |
| Formato | [A-z0-9\.\+\_-]+@[A-z0-9\.\_-]+\.[A-z]{2,6} |
| LE - Lunghezza | 1. < 0 or > 30 [error] 2. >= 0 and <= 256 [property LE\_OK] |
| FE - Formato | 1. Non rispetta il formato [error] 2. Rispetta il formato [if LE\_OK] [property FE\_OK] |
| EE – Esistenza | 1. Esistenza nel sistema [if LE\_OK] [if FE\_OK] [errore] 2. Non esiste nel database [property EE\_OK] |

|  |  |  |
| --- | --- | --- |
| Codice | Combinazione | Esito |
| TC\_RU\_1:1 | LC1 | Error |
| TC\_RU\_1:2 | LC2.LN1 | Error |
| TC\_RU\_1:3 | LC2.LN2.LPW1 | Error |
| TC\_RU\_1:4 | LC2.LN2.LPW2.LE1 | Error |
| TC\_RU\_1:5 | LC2.LN2.LPW2.LE2.FE1 | Error |
| TC\_RU\_1:6 | LC2.LN2.LPW2.LE2.FE2.EE1 | Error |
| TC\_RU\_1:7 | LC2.LN2.LPW2.LE2.FE2.EE2 | Registrazione |

* + 1. Aggiunta nuovo indirizzo di spedizione in fase di acquisto

I dati inseriti dal cliente sono i seguenti:

* Cognome: Stringa con un numero di caratteri compreso tra i 2 e i 30 caratteri
* Nome: Stringa con un numero di caratteri compreso tra i 2 e i 30 caratteri
* Indirizzo: Stringa con un numero di caratteri compreso tra i 2 e i 50 caratteri
* Città: Stringa con un numero di caratteri compreso tra i 2 e i 50 caratteri
* Cap: intero di valore massimo 99999
* Provincia: Stringa con un numero di caratteri compreso tra i 2 e i 50 caratteri
* Alias: Stringa con un numero di caratteri compreso tra i 2 e i 30 caratteri

|  |  |
| --- | --- |
| Parametro | Cognome |
| LC - Lunghezza | 1. < 2 or > 30 [error] 2. >= 2 and <= 30 [property LC\_OK] |

|  |  |
| --- | --- |
| Parametro | Nome |
| LN - Lunghezza | 1. < 2 or > 30 [error] 2. >= 2 and <= 30 [property LN\_OK] |

|  |  |
| --- | --- |
| Parametro | Indirizzo |
| LI - Lunghezza | 1. < 2 or > 50 [error] 2. >= 2 and <= 50 [property LI\_OK] |

|  |  |
| --- | --- |
| Parametro | Città |
| LCI - Lunghezza | 1. < 2 or > 50 [error] 2. >= 2 and <= 50 [property LCI\_OK] |

|  |  |
| --- | --- |
| Parametro | Cap |
| CAP - Valore | 1. > 99999 [error] 2. <= 99999 [property Cap\_OK] |

|  |  |
| --- | --- |
| Parametro | Provincia |
| LP - Lunghezza | 1. < 2 or > 50 [error] 2. >= 2 and <= 50 [property LP\_OK] |
| Parametro | Alias |
| LA - Lunghezza | 1. < 2 or > 30 [error] 2. >= 2 and <= 30 [property LA\_OK] |

|  |  |  |
| --- | --- | --- |
| Codice | Combinazione | Esito |
| TC\_IS\_1:1 | LC1 | Error |
| TC\_IS\_1:2 | LC2.LN1 | Error |
| TC\_IS\_1:3 | LC2.LN2.LI1 | Error |
| TC\_IS\_1:4 | LC2.LN2.LI2.LCI1 | Error |
| TC\_IS\_1:5 | LC2.LN2.LI2.LCI2.CAP1 | Error |
| TC\_IS\_1:6 | LC2.LN2.LI2.LCI2.CAP2.LP1 | Error |
| TC\_IS\_1:7 | LC2.LN2.LI2.LCI2.CAP2.LP2.LA1 | Error |
| TC\_IS\_1:8 | LC2.LN2.LI2.LCI2.CAP2.LP2.LA2 | Inserimento indirizzo |