|  |
| --- |
| **Università degli Studi di Salerno Corso di Ingegneria del Software** |

***SocialBook  
Documento Test Plan***

****

**Partecipanti:**

|  |  |
| --- | --- |
| Nome | Matricola |
| Barbato Alessia | 0512105858 |
| Proietto Angelica | 0512105762 |
| Russo Luca | 0512105840 |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Descrizione | Autore |
|  |  |  |  |

Indice

1.Introduzione 4

2. Relazione con altri documenti 4

2.1 Relazioni con il Requirements Analysis Document 4

2.2 Relazioni con il System Design Document 4

2.3 Relazioni con l’Object Design Document 4

3. Panoramica del sistema 4

4. Features da testare/da non testare 5

5. Pass/fail criteria 6

6. Approccio 6

7. Relazioni con il System Design Document 4

8. Relazioni con l’Object Design Document 4

Test Plan 1. Introduction 2. Relationship to other documents 3. System overview 4. Features to be tested/not to be tested 5. Pass/Fail criteria 6. Approach 7. Suspension and resumption 8. Testing materials (hardware/software requirements) 9. Test cases 10. Testing schedule

# Introduzione

Lo scopo di questo documento è l’analisi e la gestione dell’attività di testing riguardante il sistema SocialBook. Viene verificato il corretto funzionamento del sistema analizzando i singoli casi presi dai test e comparando il loro output con il presunto oracolo dei risultati. Da questa comparazione sarà possibile riscontrare gli eventuali errori e problemi da dover correggere con delle opportune modifiche. Il processo di testing viene iterato fino a che non si soddisfano in maniera accettabile i requisiti funzionali e non funzionali del sistema.

# Relazione con gli altri documenti

Per l’individuazione corretta dei casi di test si è fatto riferimento ai documenti precedentemente stilati. Infatti, ogni documento precedente è un punto di partenza indispensabile per poter effettuare un testing corretto ed adeguato e per verificare che il sistema desiderato sia simile a quello proposto.

## Relazioni con il Requirements Analysis Document (RAD)

Al fine di verificare la correttezza di SocialBook sono stati predisposti dei test basati sulle funzionalità individuate nel RAD, in particolare riguardanti i requisiti funzionali e non funzionali del sistema.

## Relazioni con il System Design Document (SDD)

Al fine di verificare la tracciabilità di SocialBook sono stati predisposti dei test sulle singole decomposizioni del sistema individuate nell’SDD, con maggior accortezza per il sottosistema “Gestione Utente” in quanto composto dal maggior numero di requisiti funzionali.

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

Al fine di verificare le performance di SocialBook sono stati predisposti dei test per la verificabilità del funzionamento delle interfacce specificate nell’ODD.

# Panoramica del sistema

L’obiettivo del sistema è quello di offrire a tutti gli appassionati di lettura una piattaforma online sicura che permetta all’utente non solo di acquistare libri (in formato digitale), ma anche di organizzarli come meglio si preferisce, recensirli, e comunicare con gli altri utenti.

L’obiettivo della piattaforma è quella di fornire un servizio veloce e sempre aggiornato in modo da garantire l’esperienza migliore possibile agli utenti.

Inoltre la piattaforma sarà completamente gestita da tre categorie di Admin:

-Il System Manager si occupa della manutenzione del codice

-Il Costumer Manager si occupa della gestione degli utenti registrati e delle annesse recensioni

-Il Catalogue Manager si occupa della gestione del catalogo e del singolo libro

# Features da testare/da non testare

## Gestione Utente

● TC\_GU1 Registrazione

● TC\_GU2 Log-in

● TC\_GU3 Creazione Booklist

● TC\_GU4 Modifica Booklist

● TC\_GU5 Elimina Booklist

● TC\_GU6 Recensione prodotto

### NON TESTATI

● Log-out

● Visualizzazione area utente

● Modifiche utente

● Elimina recensione

● Visualizzazione Booklist personali

● Visualizzazione, inserimento e rimozione Preferiti

## Gestione Acquisto

● TC\_GA1 Inserimento prodotto in carrello

● TC\_GA2 Rimozione libro da carrello

● TC\_GA3 Acquisto

### NON TESTATI

● Visualizzazione carrello

● Visualizzazione ordini

## Gestione Visualizzazione

● TC\_GV1 Ricerca

● TC\_GV2 Visualizzazione profilo utente

### NON TESTATI

● Visualizzazione Catalogo

● Visualizzazione Libro

## Gestione Interazione

● TC\_GI1 Segui utente

### NON TESTATI

● Smetti di seguire utente

● Segui Booklist

● Smetti di seguire Booklist

● Visualizzazione seguiti e seguaci

● Visualizzazione Booklist

## Gestione Ticket

### NON TESTATI

● Interazione Ticket

● Visualizzazione Ticket

## Gestione Customer

### NON TESTATI

● Rimozione Utente

● Rimozione Recensione

● Visualizzazione Utenti

## Gestione Libri

### NON TESTATI

● Aggiunta al catalogo

● Rimozione dal catalogo

● Modifica al prezzo libro

● Creazione nuovo libro

# Pass/Fail criteria

Un test avrà successo (pass) se, dato l’input al sistema, l’output osservato sarà diverso dall’output attesto (oracolo). Al contrario, un test fallirà (fail) quando il risultato osservato è uguale a quello atteso. Il testing sarà considerato valido se tutti i seguenti vincoli saranno rispettati: testare tutti i requisiti fondamentali, effettuare test di regressione ogni volta che si introducono nuove caratteristiche al sistema o vengono modificate quelle presenti, infine raggiungere un branch coverage non inferiore al 75%.

# Approccio

Per il sistema SocialBook, il testing si compone tre fasi. Nella prima fase, verranno eseguiti i test di unità dei singoli componenti, in modo da testare nello specifico la correttezza di ciascuna unità andando a constatare il corretto funzionamento. Nella seconda fase, verrà effettuato il testing di integrazione in cui si andrà a testare l’integrazione dei vari sottosistemi. Infine, verrà eseguito il testing di sistema che vedrà come oggetto di testing l’intero sistema assemblato nei suoi componenti. Nella sessione di testing del sistema verrà utilizzato un approccio “BLACK BOX” ed una strategia bottom-up: tale strategia prevede che i sottosistemi nel layer più in basso della gerarchia vengano testati individualmente, successivamente vengono testati e integrati i layer più in alto. Si ripete questo passo finché tutti i sottosistemi sono testati.

## Test di unità

Per effettuare il testing di unità andremo a testare ogni metodo di ciascuna classe del sistema. Si definiranno i casi di test in modo funzionale (black-box). Similmente ai casi di test di integrazione, per la definizione dei casi di test unitari verrà utilizzato il framework JUnit. Verrà definita una classe di test per ciascuna classe sorgente da testare. Se l'esecuzione di tutti i test unitari non porta alla branch coverage minima richiesta (75%).

## Test di integrazione

Verrà utilizzato un approccio bottom-up (più adatto al software Object Oriented). La definizione dei casi di test avverrà tramite il framework JUnit.

## Test di sistema

Per questo tipo di testing sarà utilizzato il tool Selenium IDE, che permette di registrare delle interazioni utente su un browser, e così di implementare ed eseguire i casi di test di sistema. Durante il testing di sistema, il server sarà deployato in localhost.