UNIVERSITÀ DEGLI STUDI DI SALERNO

Corso di Laurea in Informatica

Progetto Ingegneria del Software

TEST PLAN

VERSIONE 1.2

BookPad

Partecipanti

Nome	Matricola
Marica D'Alfonso	0512106258
Marianna Farina	0512109126

Sommario

1. Introduzione

Lo scopo del documento di test plan è quello di pianificare e gestire le attività di test per il sistema BookPad. Verranno indicate le componenti e le funzionalità da testare, le strategie di testing adottate e gli strumenti utilizzati. Lo scopo del testing è quello di rilevare errori in maniera pianificata all'interno del codice realizzato, affinché essi non si presentino durante l'utilizzo da parte dell'utente. I risultati prodotti dai test saranno utilizzati per comprendere dove intervenire per correggere gli errori o apportare modifiche per migliorare il sistema.

2. Relazione con altri documenti

Il test plan ha una forte correlazione con i documenti prodotti finora. Per verificare il corretto funzionamento del sistema verranno utilizzati dei test case individuati durante il processo di sviluppo del sistema, e basati sulle funzionalità individuate nel documento di raccolta e analisi dei requisiti (RAD).

RAD: la relazione con il Requirement Analysis Document riguarda in particolare i requisiti funzionali e non funzionali del sistema, i test verranno progettati tenendo in considerazione le specifiche espresse in tale documento.

SDD: il System Design Document rappresenta l'architettura del sistema, e nel progettare i test si terrà conto dell'architettura del software e della suddivisione in sotto-sistemi.

ODD: i test si baseranno sulle interfacce definite nell'Object Design Document.

3. Funzionalità

Abbiamo deciso di non testare i metodi *get* e *set* e i metodi con priorità bassa.
Di seguito vengono elencate le funzionalità introdotte nel sistema che saranno sottoposte a testing:

- **Gestione account**
  - Registrazione
  - Login
- **Operazioni sulle storie**
  - Pubblicazione di una storia
  - Ricerca di una storia
  - Commento della storia

4. Criteri di successo/insuccesso

I dati di input dei test case vengono raggruppati in sotto insiemi, dalle caratteristiche comuni in modo da avere dei dati che siano rappresentativi per una intera porzione del dominio.
Il testing ha successo se l'output osservato differisce dall'output atteso: ciò significa che la fase di testing avrà successo se individuerà una failure. In tal caso questa verrà analizzata e, se legata ad un fault, si procederà alla sua correzione.
Sarà infine iterata la fase di testing per verificare che la modifica non abbia impattato su altri componenti del sistema. La failure quindi è qualsiasi variazione del comportamento del sistema rispetto al comportamento atteso specificato in analisi dei requisiti.

5. Approccio

L'approccio alla fase di testing si compone di tre fasi, la prima avrà lo scopo testare le componenti del sistema singolarmente, poi verranno testate le componenti raggruppate integrandole insieme in gruppi, infine si testerà il funzionamento dell'intero sistema.

5.1 Testing di unità

Per i test di unità verrà utilizzata una strategia **black box**. Così facendo andremo ad esaminare le funzionalità dell'applicazione ed il comportamento input/output delle singole componenti senza tener conto della loro struttura interna. Essendo per la quasi totalità dei domini impossibile generare tutti i possibili input, verrà partizionato l'input utilizzando la tecnica del Category Partition. I risultati del testing verranno analizzati e usati per correggere gli errori che causano il fallimento del sistema.

5.2 Testing di integrazione

Dopo aver sottoposto ogni componente al testing di unità, ed aver corretto gli eventuali errori scaturiti dal test, essi verranno integrati in sottosistemi più grandi per sottoporli ad un test di integrazione. Il testing verrà effettuato seguendo la strategia "Bottom-Up" che prevede l'integrazione dal basso verso l'alto.

5.3 Testing di integrazione

Prima di essere pronto all'uso, il sistema affronterà l'ultima fase di testing, quello di sistema, per dimostrare che siano soddisfatti tutti i requisiti richiesti. Lo scopo di questa fase è testare le funzionalità più importanti, usate maggiormente e con maggior probabilità di fallimento. Per determinare se il sistema rispecchia tutti i requisiti funzionali e globali allora sarà effettuato il testing tramite il tool **Selenium**.

6. Sospensione e ripresa

6.1 Criteri di sospensione

La fase di testing del sistema verrà sospesa quando si otterranno i risultati attesi in accordo con i tempi di sviluppo previsti, tenendo sempre conto dei costi dell'attività di testing.
Questo processo verrà quindi portato avanti quanto più possibile nel tempo senza però rischiare di ritardare la consegna finale del progetto.

6.2 Criteri di ripresa

La fase di testing potrà riprendere in seguito a modifiche o correzioni che generano errori o fallimenti, i test case verranno, quindi, sottoposti nuovamente al sistema assicurandosi così di aver risolto effettivamente il Problema.

7. Materiali per il test

Gli strumenti utilizzati sono:

- Web Server Apache Tomcat
- MySQL Server
- JUnit
- Mockito
- Selenium IDE

8. Casi di test

8.1 Gestione account

TC_REG – Registrazione

Parametro: email	
CATEGORIE	**SCELTE**
Formato email [**fe**]	1. Non rispetta il formato [property **formatoEmailNonValido**] 2. Rispetta il formato [property **formatoEmailValido**]
Esistenza nel database [**ee**]	**1.** Esiste nel database [property **emailEsiste**] 2. Non esiste nel database [*if formatoEmailValido*] [property **emailNonEsiste**]

Parametro: password	
CATEGORIE	**SCELTE**
Lunghezza password [**lp**]	1. Lunghezza < 8 [property **passwordCorta**] 2. Lunghezza >= 8 [property **passwordLunga**]
Formato password [**fp**]	1. Non rispetta il formato [*if passwordLunga*] [property **formatoPassNonValido**] 2. Rispetta il formato [*if passwordLunga*] [property **formatoPassValido**]

Parametro: password2	
CATEGORIE	**SCELTE**
Coincide con password [**ep**]	1. Non coincide con *password* [*if formatoPassValido*] [property **nonCoincide**] **2.** Coincide [*if formatoPassValido*] [property **coincide**]

Parametro: username	
CATEGORIE	**SCELTE**
Lunghezza username [**lu**]	1. Lunghezza < 4 oppure lunghezza > 15 [property **usernameNonValido**] 2. Lunghezza >= 4 e lunghezza <= 15 [property **usernameValido**]
Formato username[**fu**]	3. Non rispetta il formato [*if usernameValido*] [property **formatoUsernameNonValido**] 4. Rispetta il formato [*if usernameValido*] [property **formatoUsernameValido**]

Codice	Combinazione	Esito
TC_REG_1.1	fe1	errore
TC_REG_1.2	fe2, ee1	errore
TC_REG_1.3	fe2, ee2, lp1	errore
TC_REG_1.4	fe2, ee2, lp2, fp1	errore
TC_REG_1.5	fe2, ee2, lp2, fp2, ep1	errore
TC_REG_1.6	fe2, ee2, lp2, fp2, ep2, lu1	errore
TC_REG_1.7	fe2, ee2, lp2, fp2, ep2, lu2,fu1	errore
TC_REG_1.8	fe2, ee2, lp2, fp2, ep2, lu2,fu2	successo

TC_LOG – Login

Parametro: email	
CATEGORIE	**SCELTE**
Presenza nel database [**ee**]	1. L'email non esiste nel database [property **emailNonEsiste**] 2. L'email esiste nel database [property **emailEsiste**]

Parametro: password	
CATEGORIE	**SCELTE**
Corrispondenza password [**cp**]	1. la password non coincide [property **passwordNonCorretta**] 2. la password coincide [property **passwordCorretta**]

Codice	Combinazione	Esito
TC_LOG_2.1	ee1	errore
TC_LOG_2.2	ee2, cp1	errore
TC_LOG_2.3	ee2, cp2	successo

8.2 Operazioni sulle storie

TC_PUB – Pubblica storia

Parametro: titolo	
CATEGORIE	**SCELTE**
Lunghezza titolo[**lt**]	1. Lunghezza <= 0 oppure lunghezza > 60 [property **lunghezzaTitoloErrata**] 2. Lunghezza > 0 e lunghezza <= 60 [property **lunghezzaTitoloCorretta**]

Parametro: trama	
CATEGORIE	**SCELTE**
Lunghezza trama[**ltr**]	1. Lunghezza <= 0 oppure lunghezza > 30000 [property **lunghezzaTramaErrata**] 2. Lunghezza > 0 e lunghezza <= 30000 [property **lunghezzaTramaCorretta**]

Parametro: genere	
CATEGORIE	**SCELTE**
Genere esiste **[ge]**	1. Il genere non è presente nel database [property **genereNonEsiste**] 2. il genere è presente nel database [property **genereEsiste**]

Parametro: titoli_capitoli	
CATEGORIE	**SCELTE**
Numero capitoli [**nc**]	1. Numero capitoli = 0 [property **primoCapitoloMancante**] 2. Numero capitoli > 0 [property **primoCapitoloOk**]
Lunghezza titoli **[ltc]**	1. Lunghezza <= 0 oppure lunghezza > 60 [i*f primoCapitoloOK*] [property **lunghezzaTitoloCapitoliErrata**] 2. Lunghezza > 0 e lunghezza <= 60 [i*f primoCapitoloOK*] [property **lunghezzaTitoloCapitoliCorretta**]

Parametro: contenuto_capitoli	
CATEGORIE	**SCELTE**
Numero contenuto capitoli corrisponde a numero capitoli [**ncc**]	1. Numero di contenuti != numero capitoli [property **numeroContenutiNonCorrispondente**] 2. Numero di contenuti == numero capitoli [property **numeroContenutiOk**]
Lunghezza contenuti capitoli [**lcc**]	1. Lunghezza <= 0 oppure lunghezza > 60000 [*if numeroContenutiOk*] [property **lunghezzaContenutiErrata**] 2. Lunghezza > 0 e lunghezza <= 60000 [*if numeroContenutiOk*] [property **lunghezzaContenutiCorretta**]

Codice	Combinazione	Esito
TC_PUB_3.1	lt1	errore
TC_PUB_3.2	lt2, ltr1	errore
TC_PUB_3.3	lt2, ltr2, ge1	errore
TC_PUB_3.4	lt2, ltr2, ge2, nc1	errore
TC_PUB_3.5	lt2, ltr2, ge2, nc2, ltc1	errore
TC_PUB_3.6	lt2, ltr2, ge2, nc2, ltc2, ncc1	errore
TC_PUB_3.7	lt2, ltr2, ge2, nc2, ltc2, ncc2, lcc1	errore
TC_PUB_3.8	lt2, ltr2, ge2, nc2, ltc2, ncc2, lcc2	successo

TC_RIC – Ricerca storia

Parametro: tipo	
CATEGORIE	**SCELTE**
Tipo valorizzato**[tv]**	1. tipo = null [property **tipoNonValorizzato**] 2. Tipo diverso da null [property **tipoValorizzato**]
Tipo corretto **[tc]**	1. Il tipo di ricerca non è tra ‘user’, ‘tag’ o ‘titolo’ [*if tipoValorizzato*] [property **tipoNonCorretto**] 2. Il tipo di ricerca è tra ‘user’, ‘tag’ o ‘titolo’ [*if tipoValorizzato*] [property **tipoCorretto**]

Parametro: testo	
CATEGORIE	**SCELTE**
Testo valorizzato**[txv]**	1. testo= null [property **testoNonValorizzato**] 2. testo diverso da null [property **testoValorizzato**]
Testo corretto **[txc]**	1. il testo è vuoto [*if testoValorizzato*] [property **testoNonCorretto**] 2. Il testo non è vuoto [*if testoValorizzato*] [property **testoCorretto**]

Codice	Combinazione	Esito
TC_RIC_4.1	tv1	errore
TC_RIC_4.2	tv2, tc1	errore
TC_RIC_4.3	tv2, tc2, txv1	errore
TC_RIC_4.4	tv2, tc2, txv2, txc1	errore
TC_RIC_4.5	tv2, tc2, txv2, txc2	successo

TC_COM– Aggiunta commento

Parametro: id	
CATEGORIE	**SCELTE**
Storia esiste [**se**]	1. non esiste nel database una storia con l'id [property **storiaNonEsiste**] 2. esiste nel database una storia con l'id [property **storiaEsiste**]

Parametro: testo	
CATEGORIE	**SCELTE**
Lunghezza testo [**lt**]	1. Lunghezza <= 0 oppure lunghezza > 500 [property **lunghezzaTestoErrata**] 2. Lunghezza > 0 e lunghezza <= 500 [property **lunghezzaTestoCorretta**]

Codice	Combinazione	Esito
TC_COM_6.1	se1	errore
TC_COM_6.2	se2, lt1	errore
TC_COM_6.3	se2, lt2	successo