Università degli studi di Salerno
Dipartimento di Informatica
Corso di Laurea in Informatica

Test Plan Document

"TomMASO"

Docente:
Andrea De Lucia

Tutor:
Manuel De Stefano

Studenti:

Nome	*Matricola*
Antonio Allocca	0512106933
Pasquale Di Costanzo	0512106675
Vincenzo Esposito	0512108127
Antonio Toppi	0512106882

Anno Accademico: 2021/22

Revision History

Data	Versione	Descrizione
02/01/2022	0.1	Prima stesura
03/01/2022	0.2	Aggiunte Funzionalità da testare
05/01/2022	0.3	Aggiunte Specifiche casi di test
07/01/2022	0.4	Aggiunti i casi d’uso e i relativi diagrammi

Indice

1. Introduzione

Il Test Plan Document di TomMASO include le informazioni necessarie per definire l'approccio che deve essere usato nella fase di testing del progetto. Lo scopo di questo documento è quello di gestire lo sviluppo e le attività di test di TomMASO per trovare eventuali errori all'interno del codice realizzato. I risultati dei test saranno poi utilizzati per trovare correzioni da apportare al codice, al fine di risolvere i problemi per migliorare il sistema.

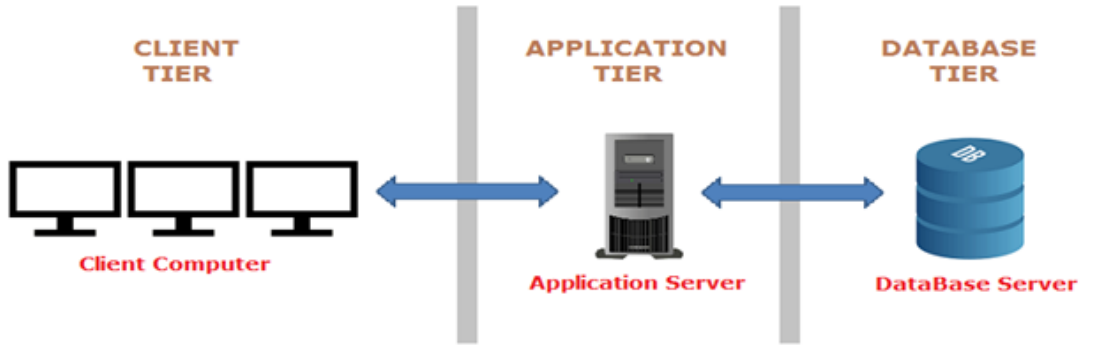
2. Riferimenti

RAD: dove verranno testati i casi d'uso
SDD: che include la definizione del sistema in sottosistemi e il controllo degli accessi

ODD: verranno testate le classi implementate

3. Panoramica del sistema

Il nostro sistema è basato su architettura a tre livelli.



4. Funzionalità da testare

Di seguito verranno riportate le funzionalità del sistema che saranno sottoposte a test:

- UC1. UserSignUp
- UC2. Login
- UC6. UploadFile
- UC7. EditMarkDownFile
- UC11. SendMessage
- UC12. SendComment
- UC15. DeleteBlog
- UC21. Search
- UC23. CreateBlog

5. Criteri di successo/insuccesso

La fase di testing verrà ritenuta “di successo” nel caso in cui tutti i test previsti abbiano un risultato positivo (il risultato del caso di test coincide con l’oracolo).

6. Approccio

Il testing che svolgeremo su questo sistema si compone di tre fasi:

- Testing di unità, con lo scopo di testare le componenti del sistema singolarmente
- Testing di integrazione, che servirà a testare le funzionalità dei vari componenti del sistema
- Testing di sistema, che mira a verificare che l’intero sottosistema soddisfi le richieste del cliente.

6.1. Test di unità

Per realizzare il testing di ogni singola componente verrà utilizzata la tecnica ***Black-Box***.

Per il partizionamento degli input verranno usate ***classi di equivalenza*** rendendo i test più efficienti. I risultati del testing verranno analizzati e usati successivamente per correggere eventuali errori del sistema.

6.2. Test di integrazione

Il test di integrazione serve a testare l'intero sistema e si sviluppa dopo il test di unità, quando ogni parte testata viene integrata con il resto del sistema.

Verrà usata la strategia ***bottom-up***: strategia di testing che permette di testare prima i componenti indipendenti, poi di integrarli con i componenti che ne dipendono. Si ripete questo processo finché non sono state testate tutte le classi.

6.3. Test di sistema

Il test di sistema ha lo scopo di testare tutte le funzionalità più importanti e usate maggiormente. Trattandosi di un sistema web-based verrà utilizzato ***Selenium.***

7. Sospensione e ripresa

La fase di testing del sistema verrà sospesa in caso siano rilevate failure. Dopo la loro correzione e convalida attraverso test di regressione si potrà riprendere il testing.

8. Materiali di prova (requisiti hardware/software)

Gli strumenti utilizzati nella fase di testing saranno:

- ***MariaDB*** per la gestione del database
- ***Apache Tomcat 9***, come web server
- ***JUnit, DBUnit, Mockito*** per il test di unità e di integrazione.
- ***Selenium*** per il test di sistema
- ***Firefox,*** come web browser
- **JDK 1.8+**

9. Specifiche casi di test

Specifica dei casi di test per il caso d'uso UC1. UserSignUp	
Parametri	username
	password1
	password2
Oggetti dell'ambiente	Database
Categorie	**Categoria 1:** formato username (/^[a-zA-Z][a-zA-Z0-9-.]{7,19}$/)
	Categoria 2: username già presente nel database
	Categoria 3: formato password (/.{8,}/)
	Categoria 4: password1 uguale a password2
Scelte	**Scelte per Categoria 1:** FU1: formato username non corretto FU2: formato username corretto
	Scelte per Categoria 2: UP1: username già presente UP2: username non presente
	Scelte per Categoria 3: FP1: formato password non corretto FP2: formato password corretto

	Scelte per Categoria 4: PU1: password1 uguale a password2 PU2: password1 diversa da password2
Vincoli	**FU1**: formato username errato
	FU2: formato username corretto [property formato_username_corretto]
	UP1: username già presente [if formato_username_corretto]
	UP2: username non presente [property username_non_presente]
	FP1: formato password errato [if formato_username_corretto and username_non_presente]
	FP2: formato password corretto [property formato_password_corretto]
	PU1: password1 diversa da password2 [if formato_username_corretto and username_non_presente and formato_password_corretto]
	PU2: password1 uguale a password2
Test Frame	**TC1:** FU2, UP2, FP2, PU2
	TC2: FU1, UP2, FP2, PU2
	TC3: FU2, UP1, FP2, PU2
	TC4: FU2, UP2, FP1, PU2
	TC5: FU2, UP2, FP2, PU1

Test Case	**TC1:** username="AAAAAAAA", password1="BBBBBBBB", password2="BBBBBBBB" - ORACOLO: Utente correttamente autenticato
	TC2: username="AAAAAAA!", password1="BBBBBBBB", password2="BBBBBBBB" - ORACOLO: Utente reindirizzato alla registrazione con il messaggio “Username non valido”
	TC3: username="XXXXXXXX", password1="BBBBBBBB", password2="BBBBBBBB" - ORACOLO: Utente reindirizzato alla registrazione con il messaggio “Impossibile registrare l'account”
	TC4: username="AAAAAAAA", password1="BBBBBBB", password2="BBBBBBB" - ORACOLO: Utente reindirizzato alla registrazione con il messaggio “La password deve contenere almeno 8 caratteri”
	TC5: username="AAAAAAAA", password1="BBBBBBBB", password2="CCCCCCCC" - ORACOLO: Utente reindirizzato alla registrazione con il messaggio “Le password non corrispondono”

Specifica dei casi di test per il caso d'uso UC2. Login	
Parametri	username
	password
Oggetti dell'ambiente	Database
Categorie	**Categoria 1:** username presente nel database
	Categoria 2: password associata a username nel database
	Categoria 3: permesso login
Scelte	**Scelte per Categoria 1:** UP1: username presente nel database UP2: username NON presente nel database
	Scelte per Categoria 2: PA1: password associata a username PA2: password NON associata a username
	Scelte per Categoria 3: PL1: l'utente ha il permesso di autenticarsi PL2: l'utente NON ha il permesso di autenticarsi
Vincoli	**UP1**: username presente nel database [property username_presente]

	UP2: username NON presente nel database [property username_non_presente]
	PA1: password associata a username [if username_presente] [property utente_esistente]
	UP2: username non presente [property username_non_presente]
	PA2: password NON associata a username
	PL1: l'utente ha il permesso di autenticarsi [if utente_esistente]
	PL2: l'utente NON ha il permesso di autenticarsi
Test Frame	**TC1:** UP1, PA1, PL1
	TC2: UP1, PA2
	TC3: UP2, PA2
	TC4: UP1, PA1, PL1
Test Case	**TC1:** username: "a.allocca", password: "Allocca123" – in DB presente la coppia (username = "a.allocca", password = "Allocca123" permessi="8") – ORACOLO: Utente autenticato e indirizzato alla sua home page
	TC2: username: "a.allocca", password: "DiCostanzo321" – in DB presente la coppia (username = "a.allocca", password = "Allocca123") – ORACOLO: Visualizzazione

	pagina di autenticazione con messaggio di errore “Credenziali errate”
	TC3: username: “a.delucia”, password: “AAAAAAAA” – in DB non presente nessun utente con username “a.delucia” – ORACOLO: Visualizzazione pagina di autenticazione con messaggio di errore “Credenziali errate”
	TC4: username: “utenteBannato”, password: “bannato12” – in DB presente la coppia (username = “a.allocca”, password = “Allocca123” permessi=”0”) – ORACOLO: Utente reindirizzato alla pagina di errore perché non dispone dei permessi necessari all’autenticazione
Specifica dei casi di test per il caso d’uso UC6. UploadFile	
Parametri	filename
Oggetti dell’ambiente	Filesystem
Categorie	**Categoria 1**: filename
Scelte	**Scelte per Categoria 1:** **NF1**: Nome file invalido **NF2**: Nome file valido
Test Frame	**TC1**: NF1
	TC2: NF2
Test Case	**TC1**: filename = "../AAA" - ORACOLO: Il file non viene caricato sulla piattaforma
	TC2: filename = "AAABBB" - ORACOLO: Il file viene caricato sulla piattaforma

Specifica dei casi di test per il caso d'uso UC7. EditMarkDownFile	
Parametri	filename
	user
Oggetti dell'ambiente	Filesystem
	Sessione
Categorie	**Categoria 1:** file esistente
	Categoria 2: utente proprietario
Scelte	**Scelte per Categoria 1:** FE1: il file esiste FE2: il file NON esiste
	Scelte per Categoria 2: UP1: utente proprietario del file UP2: utente NON proprietario del file
Vincoli	**FE1**: il file esiste [property file_esistente]
	FE2: il file NON esiste
	UP1: utente proprietario del file [if file_esistente]
	UP2: utente NON proprietario del file [if file_esistente]
Test Frame	**TC1:** FE1, UP1
	TC2: FE1, UP2
	TC3: FE2
Test Case	**TC1:** filename = "LaMiaPagina", username dell'utente in sessione = "p.dicostanzo", nel file system è presente tale file – ORACOLO: Le modifiche effettuate al file vengono correttamente salvate e l'utente viene reindirizzato alla pagina visualizzata correttamente.

	TC2: filename = “LaMiaPagina”, username dell’utente in sessione = “v.esposito”, nel file system è presente tale file – ORACOLO: Le modifiche effettuate al file non vengono salvate perché l’utente non è il proprietario del file e viene reindirizzato alla pagina di errore
	TC3: filename = “FileNonEsistente”, nel file system non è presente tale file – ORACOLO: Viene creato per l’utente user un file vuoto.

Specifica dei casi di test per il caso d’uso UC11. Send Message	
Parametri	message
Oggetti dell’ambiente	Database
Categorie	**Categoria 1:** formato messaggio
Scelte	**Scelte per Categoria 1:** **FM1**: formato messaggio corretto **FM2**: formato messaggio NON corretto
Vincoli	**FM1**: formato messaggio corretto [property messaggio_corretto]
	FM2: formato messaggio NON corretto
Test Frame	**TC1:** FM1
	TC2: FM2

Test Case	**TC1:** messaggio=”AAA” - ORACOLO: il messaggio viene inviato
	TC2: messaggio=” ” – ORACOLO: il messaggio non viene inviato

Specifica dei casi di test per il caso d'uso UC12. SendComment	
Parametri	commento
Oggetti dell'ambiente	Database
Categorie	**Categoria 1**: Lunghezza commento (per specifica: il commento deve avere almeno 3 caratteri)
Scelte	**Scelte per categoria 1**: **CLI1**: lunghezza commento corretta (>=3) **CLI2**: lunghezza commento NON corretta
Vincoli	**CL1**: lunghezza commento corretta [property commento_corretto]
	CL2: lunghezza commento NON corretta [property commento_non_corretto]
Test Frame	**TF1**: CL1
	TF2: CL2
Test Case	**TC1** - TF1 commento="testingiswonderful" - ORACOLO: Commento inviato e reindirizzazione alla pagina
	TC2 - TF2 commento="ab" - ORACOLO: Commento non inviato e reindirizzazione alla pagina e visualizzazione messaggio di errore

Specifica dei casi di test per il caso d'uso UC15. DeleteBlog	
Parametri	blogname
Oggetti dell'ambiente	Database
Categorie	**Categoria 1**: blog esistente
Scelte	**Scelte per la Categoria 1**: **BE1**: il blog esiste **BE2**: il blog NON esiste
Test Frame	**TC1:** BE1
	TC2: BE2
Test Case	**TC1**: blogname = "BlondeSalad" - ORACOLO: Blog eliminato
	TC2: blogname = "BlogInesistente" - ORACOLO: Blog non cancellato

Specifica dei casi di test per il caso d'uso UC21. Search	
Parametri	tipo
	testo
Oggetti dell'ambiente	Database
Categorie	**Categoria1**: blog esistente
	Categoria2: utente esistente
Scelte	**Scelte per tipo**: **T1**: utente **T2**: blog
	Scelte per Categoria1: **BE1**: il blog non esiste **BE2**: il blog esiste
	Scelte per Categoria2: **UE1**: l'utente non esiste **UE2**: l'utente esiste
Vincoli	**T1**: utente [property type_user]
	T2: blog [property type_blog]
	BE1: il blog non esiste [if type_blog]
	BE2: il blog esiste [if type_blog]
	UE1: l'utente non esiste [if type_user]
	UE2: l'utente esiste [if type_user]
Test Frame	**TC1**: T1, UE1
	TC2: T1, UE2

	TC3: T2, BE1
	TC4: T2, BE2
Test Case	**TC1**: tipo="utente", testo="CCCCCCCC" - ORACOLO: L'utente viene reindirizzato alla pagina di errore “Username non trovato”
	TC2: tipo="utente", testo="DDDDDDDD" - ORACOLO: L'utente viene reindirizzato al profilo dell’utente
	TC3: tipo="blog", testo="AAAAAAAA" - ORACOLO: L'utente viene reindirizzato alla pagina di errore “Blog non trovato”
	TC4: tipo="blog", testo="BBBBBBBB "- ORACOLO: L'utente viene reindirizzato al blog

Specifica dei casi di test per il caso d'uso UC23. CreateBlog	
Parametri	blogname
Oggetti dell'ambiente	Database
Categorie	**Categoria 1:** formato blogname (/^[a-zA-Z][a-zA-Z0-9-.]{7,19}$/)
	Categoria 2: blogname presente nel database
Scelte	**Scelte per Categoria 1:** **FB1**: formato blogname corretto **FB2**: formato blogname NON corretto
	Scelte per Categoria 2: **BD1**: blogname presente nel database **BD2**: blogName NON presente nel database
Vincoli	**FB1**: formato blogname corretto [property blogname_corretto]
	FB2: formato blogname NON corretto
	BD1: blogname presente nel database [if blogname_corretto]
	BD2: blogname NON presente nel database
Test Frame	**TC1:** FB1, BD1
	TC2: FB1, BD2

Test Case	**TC1:** blogname = ”IlMioBlog” - in DB non presente alcun blog con blogname = ”IlMioBlog” - ORACOLO: il blog viene creato
	TC2: blogname = ”IlMioBlog” - in DB già presente un blog con blogname=”IlMioBlog” - ORACOLO: viene segnalato all’utente la presenza di un blog con quel nome e viene reindirizzato alla pagina di creazione