FLAT

Test plan Document

Di Luccio Luca
Garofalo Antonio
Vitale Francesco

Data	Versione	Descrizione	Autore
16/01/2017	1.0	Stesura del test Plan	Tutti
25/01/2017	2.0	Revisione finale	Tutti

Sommario

Sommario

1. Introduzione

Lo scopo del documento di Test Plan è quello di andare a verificare se il comportamento atteso del sistema eguale quello osservato, alla luce di ciò, andremo a rilevare eventuali problemi dovuti ad errori presenti nel codice. Andremo quindi a testare tutte le funzionalità già specificate all'interno dell'ODD.

2. Relazione con gli altri documenti

Questo documento è chiaramente in stretta relazione con gli altri documenti:
Nel RAD sono specificati i requisiti funzionali e non funzionali che ci aiuteranno nell'esecuzione dei test;
Nell'SDD il sistema in uso è stato diviso in tre livelli, quali Presentation Layer, Application Layer, e Storage Layer, cercheremo dunque di mantenere il testing quanto più fedele a quest'architettura.
Infine, Il test d'integrazione farà quanto più riferimento possibile alle interfacce delle classi definite nell'ODD.

3. System Overview

Il nostro sistema è stato diviso in gestioni (consultabili nel punto 4), e ognuna di queste gestioni prevede principalmente operazioni di inserimento, modifica, cancellazione e ricerca. Queste sono le funzioni che verranno testate durante il testing del sistema.

4. Features da testare e non da testare

Di seguito saranno elencate per ogni tipo di gestione le funzioni che saranno testate:

1. Gestione autenticazione
    - Login
2. Gestione registrazione
    - Registrazione
3. Gestione Account
    - Modifica dati personali
4. Gestione recensioni
    - Aggiunta recensione
5. Gestione watchlist
    -
6. Gestione amministrazione
    -
7. Gestione ricerca
    - Ricerca film

5 Criteri di Pass/Fail

I dati in input del test saranno suddivisi in classi di equivalenza, ovvero verranno raggruppati in insiemi dalle caratteristiche comuni, per i quali sarà sufficiente testare un solo elemento rappresentativo. Un input avrà superato un test se l'output risultante sarà quello atteso, cioè quello che è stato specificato in precedenza dal membro che si occuperà del testing su quel determinato test case.

6 Approccio

L'attività di testing sarà suddivisa in tre categorie: Testing di Unità, Testing di Integrazione,e Testing di Sistema.

6.1 Testing di Unità

Durante questa fase, verranno cercate le condizioni di fallimento, isolando i componenti ed usando test driver e stub, cioè implementazioni parziali di componenti che dipendono o da cui dipendono le componenti da testare. La strategia utilizzata per il testing si baserà esclusivamente sulla tecnica Black-box, che si focalizza sul comportamento Input/Output, ignorando la struttura interna della componente. Al fine di minimizzare il numero di test cases, i possibili input verranno partizionati in classi di equivalenza e per ogni classe verrà selezionato un test case. Gli stati erronei scovati in questa, come in qualsiasi altra fase del testing, che comporteranno un fallimento del sistema dovranno essere tempestivamente corretti per poi ripristinare il testing al più presto, per comunicare i vari fallimenti identificati durante la fase di testing utilizzeremmo un documento chiamanto "Test incident report".

6.2 Testing di Integrazione

In questa fase si procederà all'integrazione delle componenti di una funzionalità che verranno testate nel complesso attraverso una strategia Bottom-Up. Si passerà, poi, alla funzionalità successiva fino ad esaurire le funzionalità implementate. Quest'approccio mira principalmente a ridurre le dipendenze tra funzionalità differenti e a facilitare la ricerca di errori nelle interfacce di comunicazione tra sottosistemi.

6.3 Testing di sistema

Lo scopo di questa fase è quello di procedere all'integrazione delle componenti di una funzionalità, che verranno poi testate nel comiplesso attraverso una strategia Bottom-up. Come prima, si passerà di volta in volta alla funzionalità successiva fino ad esaurire le funzionalità implementate. Quest'approccio mira principalmente a ridurre le dipendeze tra funzionalità differenti e a facilitare la ricerca di errori nelle interfacce di comunicazione tra sottosistemi.

7 Sospensione e Riassunzione

La fase di testing verrà sospesa verrà sospesa quando si raggiungerà un compromesso ta la qualità del prodotto e costi dell'attività di testing. Il testing verrà quindi portato avanti quanto più possibile nel tempo senza però rischiare di ritardare la consegna finale del progetto.

Inoltre, in seguito alle modifiche o correzioni delle componenti che introdurranno errori o fallimenti, i test case verranno sottoposti nuovamente al sistema facendo così in modo di risolvere definitivamente il problema.

8 Materiali di Testing

Per effettuare il testing servirà solamente un pc (dotato dei vari software appositi), non sarà neanche necessaria una connessione ad internet perché il database del sistema, che ci è stato fornito da una API, è salvato in locale.

9 Casi di Testing

9.1 Gestione login

9.1.1 Category partition

Parametro: Username **Formato:** [_.A-Za-z0-9]	
Lunghezza[LU]	1. <3 and >15 [error] 2. >=3 and <=15 [proprety lunghezzaLUOK]
Formato[FU]	1. Rispecchia il formato [iflunghezzaLUOK] [propretyformatoFUOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLUOK] [error]

Parametro: Password	
Match[MP]	1. Match con password Username=false [error] 2. Match con password Username=true [proprety MP_OK]

9.1.2 Casi di test

Codice	**Combinazione**	**Esito**
TC_1_1	LU1	Error
TC_1_2	LU2,FU2	Error
TC_1_3	LU2,FU1,MP1	Error
TC_1_4	LU2,FU1,MP2	**Correct**

9.2 Gestione registrazione

9.2.1 Category partition

Parametro: Nome **Formato:** [A-Za-z]	
Lunghezza[LN]	1. <2 and >15 [error] 2. >=2 and <=15 [proprety lunghezzaLNOK]
Formato[FN]	1. Rispecchia il formato [iflunghezzaLNOK] [propretyformatoFNOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLNOK] [error]

Parametro: Cognome **Formato:** [A-Za-z]	
Lunghezza[LC]	1. <2 and >15 [error] 2. >=2 and <=15 [proprety lunghezzaLCOK]
Formato[FC]	1. Rispecchia il formato [iflunghezzaLCOK] [propretyformatoFCOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLCOK] [error]

Parametro: Indirizzo Email **Formato:** [_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@ + [A-Za-z0-9-]+(\\.[A-Za-z0-9]+)*(\\.[A-Za-z]	
Lunghezza[LP]	1. <2 [error] 2. >=2 [proprety lunghezzaLPOK]
Formato[FP]	1. Rispecchia il formato [iflunghezzaLPOK] [propretyformatoFPOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLPOK] [error]
Valore(VP)	1. != RipetiPassword [ifformatoFPOK] [error] 2. == RipetiPassword [propretyvaloreVPOK, è uguale al valore password successivo]

Parametro: Username **Formato:** [._A-Za-z0-9]	
Lunghezza[LU]	3. <3 and >15 [error] 4. >=3 and <=15 [proprety lunghezzaLUOK]
Formato[FU]	3. Rispecchia il formato [iflunghezzaLUOK] [propretyformatoFUOK, rispecchia il formato] 4. Non rispecchia il formato [iflunghezzaLUOK] [error]

Parametro: Password **Formato:** [._A-Za-z0-9]	
Lunghezza[LP]	3. <3 and >15 [error] 4. >=2 and <=15 [proprety lunghezzaLPOK]
Formato[FP]	3. Rispecchia il formato [iflunghezzaLPOK] [propretyformatoFPOK, rispecchia il formato] 4. Non rispecchia il formato [iflunghezzaLPOK] [error]

Valore(VP)	3. != RipetiPassword [ifformatoFPOK] [error] 4. == RipetiPassword [propretyvaloreVPOK, è uguale al valore password successivo]

Parametro: ConfermaPassword **Formato:** [._A-Za-z0-9]	
Lunghezza[LCP]	1. <3 and >15 [error] 2. >=3 and <=15 [proprety lunghezzaLCPOK]
Formato[FCP]	1. Rispecchia il formato [iflunghezzaLCPOK] [propretyformatoFCPOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLCPOK] [error]
Valore(VCP)	1. != RipetiPassword [ifformatoFCPOK] [error] 2. == RipetiPassword [propretyvaloreVCPOK, è uguale al valore password precedente]

9.2.2 Casi di test

Codice	**Combinazione**	**Esito**
TC_2_1	LN1	Error
TC_2_2	LN2,FN2	Error
TC_2_3	LN2,FN1,LC1	Error
TC_2_4	LN2,FN1,LC2,FC2	Error
TC_2_5	LN2,FN1,LC2,FC1,LE1	Error
TC_2_6	LN2,FN1,LC2,FC1,LE2,FE2	Error
TC_2_7	LN2,FN1,LC2,FC1,LE2,FE1,LU1	Error
TC_2_8	LN2,FN1,LC2,FC1,LE2,FE1,LU2, FU2	Error
TC_2_9	LN2,FN1,LC2,FC1,LE2,FE1,LU2, FU1,LP1	Error
TC_2_10	LN2,FN1,LC2,FC1,LE2,FE1,LU2, FU1,LP2,FP2	Error
TC_2_11	LN2,FN1,LC2,FC1,LE2,FE1,LU2, FU1,LP2,FP1,LCP1	Error
TC_2_12	LN2,FN1,LC2,FC1,LE2,FE1,LU2, FU1,LP2,FP1,LCP2,FCP2	Error
TC_2_13	LN2,FN1,LC2,FC1,LE2,FE1,LU2, FU1,LP2,FP1,LCP2,FCP1	**Correct**

9.3 Gestione modifica dati personali

9.3.1 Category partition

Parametro: Nome **Formato:** [A-Za-z]	
Lunghezza[LN]	1. <2 and >15 [error] 2. >=2 and <=15 [proprety lunghezzaLNOK]
Formato[FN]	1. Rispecchia il formato [iflunghezzaLNOK] [propretyformatoFNOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLNOK] [error]

Parametro: Cognome **Formato:** [A-Za-z]	
Lunghezza[LC]	1. <2 and >15 [error] 2. >=3 and <=15 [proprety lunghezzaLNOK]
Formato[FC]	1. Rispecchia il formato [iflunghezzaLNOK] [propretyformatoFNOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLNOK] [error]

Parametro: Password **Formato:** [._A-Za-z0-9]	
Lunghezza[LP]	1. <3 and >15 [error] 2. >=3 and <=15 [proprety lunghezzaLPOK]
Formato[FP]	1. Rispecchia il formato [iflunghezzaLPOK] [propretyformatoFPOK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLPOK] [error]

9.3.2 Casi di test

Codice	**Combinazione**	**Esito**
TC_3_1	LN1	Error
TC_3_2	LN2,FN2	Error
TC_3_3	LN2,FN1,LC1	Error
TC_3_4	LN2,FN1,LC2,FC2	Error
TC_3_5	LN2,FN1,LC2,FC1,LP1	Error

TC_3_6	LN2,FN1,LC2,FC1,LP2,FP2	Error
TC_3_7	LN2,FN1,LC2,FC1,LP2,FP1	**Correct**

9.4 Gestione aggiunta recensione

9.4.1 Category partition

Parametro: Titolo	
Lunghezza[LT]	1. <5 [error] 2. >=5 [proprety lunghezzaLTOK]

Parametro: Voto	
Selezione[SV]	1. Un valore selezionato [proprety votoOK] 2. Nessun valore selezionato [error]

Parametro: Recensione **Formato:** [._A-Za-z0-9]	
Lunghezza[LR]	1. <10 [error] 2. >=10 [proprety lunghezzaLROK]

9.4.2 Casi di test

Codice	**Combinazione**	**Esito**
TC_4_1	LT1	Error
TC_4_2	LT2,SV2	Error
TC_4_3	LT2,SV1,LR1	Error
TC_4_4	LT2,SV1,LR2	**Correct**

9.5 Gestione ricerca

9.5.1 Category partition

Parametro: Ricerca **Formato:** [._A-Za-z0-9]

Lunghezza[LR]	1. 2< [error] 2. >=2 [proprety lunghezzaLROK]
Formato[FR]	1. Rispecchia il formato [iflunghezzaLROK] [propretyformatoFROK, rispecchia il formato] 2. Non rispecchia il formato [iflunghezzaLROK] [error]

9.5.2 Casi di test

Codice	Combinazione	Esito
TC_5_1	LR1	Error
TC_5_2	LR2,FR2	Error
TC_5_3	LR2,FR1	**Correct**

10 Testing schedule

Il team del testing sarà composto da persone con una conoscenza estremamente approfondita del sistema. Solitamente i componenti del team di testing non sono coinvolti nello sviluppo del sistema, in quanto non dovrebbero essere propriamente pratici, ma vista la dimensione del gruppo di progetto, ci vediamo costretti a dover agire necessariamente in questo modo.

Prima di tutto abbiamo cercato di identificare quanti più fault presenti possibili nel sistema, per poter successivamente correggerli. Successivamente abbiamo testato il sistema nuovamente, non solo per verificare che tutti i fault trovati in precedenza siano stati corretti, ma anche per verificare che non siano stati introdotti nuovi fault.

L'attività di testing è fondamentale per lo sviluppo corretto di un software, in quanto una tale mancanza potrebbe causare il fallimento dell'intero sistema. Per tale motivo, è fondamentale schedulare il testing.

10.1 Determinazione dei ruoli

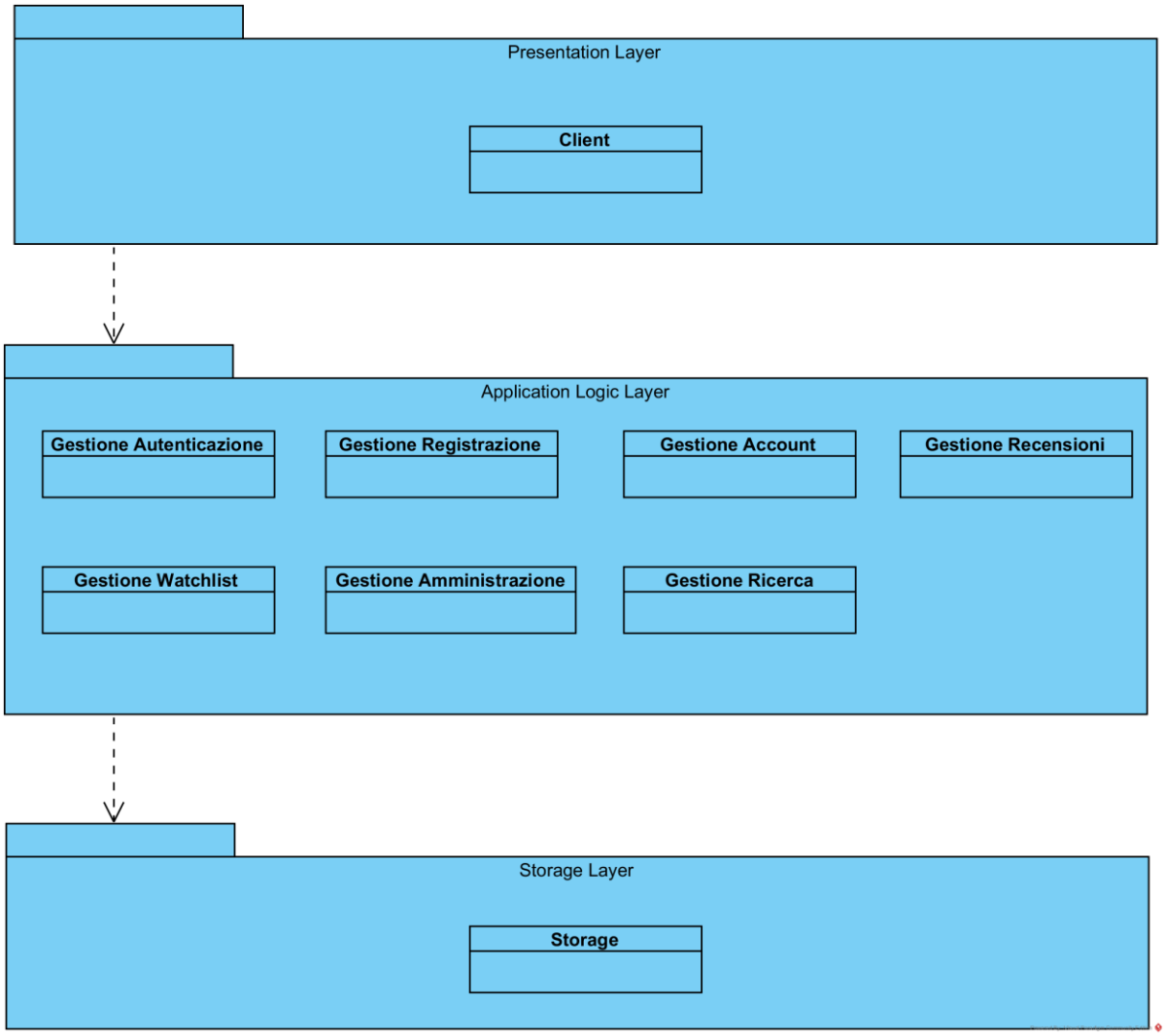
Per alleggerire il carico lavorativo, abbiamo deciso di dividere le attività di testing in parti uguali tra tutti i componenti. Per tanto ogni componente si occuperà dell'attività di testing di sistema e testing di unità.

10.2 Determinazione dei rischi

I rischi di fallimento verranno minimizzati effettuando una pianificazione verticale delle attività di testing funzionale. Questo tipo di approccio ci permetterà di rilasciare quante meno funzionalità possibili, però interamente testate, in caso di ritardi dovuti ad un numero elevato di failure.

10.3 Decomposizione gerarchica del sistema

La decomposizione in sottosistemi effettuata nell'attività di design è stata mappata sulla base di tre livelli gerarchici:



10.4 Organizzazione delle attività di testing

Le attività di testing verranno organizzate secondo uno schema che effettuerà una divisione funzionale di tipo verticale. In questo modo, al termine di ogni attività, si avrà una funzionalità completamente testata nei suoi livelli gerarchici. I vantaggi principali sono che in caso di ritardi dovuti al ritrovamento di numerose failure il sistema verrà rilasciato con meno componenti, ma interamente testate e funzionanti, come già specificato nella determinazione dei rischi.