UNIVERSITÀ DEGLI STUDI DI SALERNO
FACOLTÀ DI SCIENZE MM.FF.NN.
CORSO DI LAUREA TRIENNALE IN INFORMATICA
CORSO DI INGEGNERIA DEL SOFTWARE

Gamerland

Test Plan

VERSIONE 1.4

Anno Accademico 2017/18

Top Manager:

Professore
Prof. De Lucia Andrea
Prof. Francese Rita

Partecipanti:

Nome	Matricola
Capone Natale	0512103586
Stanzione Roberto	0512103763
Iuliano Gianpaolo	0512103514

Revision History:

Data	Versione	Descrizione	Autore
16/02/2018	1.0	Inizio stesura introduzione	Tutto il team
17/02/2018	1.1	Stesura panoramica e funzionalità	Tutto il team
18/02/2018	1.2	Stesura approccio e test cases	Tutto il team
19/02/2018	1.3	Modifica e continuazione test cases	Tutto il team
21/02/2018	1.4	Fine stesura documento	Tutto il team

Sommario

1. Introduzione

Lo scopo di questo documento è di pianificare l'attività di test del sistema Gamerland al fine di verificare se esistono differenze tra il comportamento atteso e il comportamento osservato. In questa attività andremo a rilevare gli eventuali errori prodotti all'interno del codice, per evitare che essi si presentino nel momento in cui il sistema verrà utilizzato dall'utente finale. Le attività di test sono state pianificate per le seguenti gestioni:

1. **Gestione Autenticazione;**
2. **Iscrizione al sistema;**
3. **Gestione dei cataloghi;**
4. **Gestione profilo utente;**

Si noti, tuttavia, che verranno testate esclusivamente le funzionalità implementate e specificate nell'ODD, oltre alla gestione dei test delle funzionalità, vengono anche pianificate le responsabilità del team e lo scheduling del test. La fase di testing è strettamente legata alle fasi ad essa precedenti; ogni document, risultato delle differenti fasi di sviluppo, sarà un punto di partenza indispensabile per poter effettuare un testing corretto e adeguato.

2. Documenti Correlati

Il test plan ha ovviamente una stretta relazione con il resto dei documenti che sono stati prodotti finora, poiché prima di passare alla fase di testing, oltre ad aver implementato il sistema nella gran parte, esso era stato pianificato nelle precedenti documentazioni. Questo quindi permette di rilevare le eventuali differenze tra ciò che si desiderava e ciò che invece il sistema fa. Di seguito verranno riportate le relazioni tra il test plan e la documentazione precedente.

2.1 Relazioni con il documento di analisi dei requisiti (RAD)

La relazione tra test plan e RAD riguarda in particolare i requisiti funzionali e non funzionali del sistema poiché i test che saranno eseguiti su ogni funzionalità terranno conto delle specifiche espresse nel RAD.

2.2 Relazioni con il System Design Document (SDD)

Nel System Design Document abbiamo suddiviso il nostro sistema in sottosistemi e l'architettura in tre livelli: View Layer, Control Layer e Model Layer. Il test dei vari componenti deve rimanere fedele a queste suddivisioni il più possibile.

2.3 Relazioni con l'Object Design Document (ODD)

Il test d'integrazione farà quanto più riferimento possibile alle class interfaces definite nell'ODD.

3. Panoramica del sistema

Come stabilito nel System Design Document la struttura del nostro sistema è divisa secondo un'architettura "MVC" cioè a tre livelli: View Layer, Control Layer, Model Layer. In questo caso il livello più alto interagisce con il livello applicativo che a sua volta si occuperà di eseguire le operazioni nel database di Gamerland, cercando di garantire il più possibile basso accoppiamento e alta coesione tra le varie classi. Il sistema inoltre è stato suddiviso in sottosistemi più piccoli. Abbiamo infatti i seguenti sottosistemi:

1 ClienteModelDM;
2 ProdottoModelDM;
3 OrdineModelDM;
4 Carrello;

Quasi ognuno dei precedenti sottosistemi prevede principalmente operazioni di inserimento, modifica, cancellazione e visualizzazione e saranno proprio queste funzionalità ad essere testate nel corso della fase di testing del sistema.

4. Funzionalità da testare e da non testare

Di seguito saranno elencate per ogni gestione quali sono le funzionalità che saranno testate.

1 Gestione autenticazione;

- Login

2 Scelta ed acquisto dei prodotti nei vari cataloghi del sistema;

- Aggiunta e rimozione dei prodotti dal carrello
- Acquisto dei prodotti presenti nel carrello

3 Iscrizione al sistema;

- Registrazione

4 Gestione dei cataloghi;

- Aggiunta di nuovi prodotti nei cataloghi
- Rimozione prodotti dai cataloghi

5 Gestione del magazzino;

- Modifica stato dell'ordine

6 Visualizzazione ricavi

- Visualizza ricavi

7 Gestione profilo utente

- Visualizza stato degli ordini
- Modifica indirizzo di spedizione
- Modifica password
- Eliminazione account

5. Criteri Pass/Failed

I dati di 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 dai membri del team che si occuperanno del testing su tale test case, i responsabili del testing conoscono quale dovrebbe essere l'output corretto.

6. Approccio

Le tecniche di testing adottate riguarderanno inizialmente il testing di unità dei singoli componenti, in modo da testare nello specifico la correttezza di ciascuna unità. Seguirà il testing d'integrazione, che focalizzerà l'attenzione principalmente sul test delle interfacce delle suddette unità. Infine verrà eseguito il testing di sistema, che vedrà come oggetto di testing l'intero sistema assemblato nei suoi componenti. Quest'ultimo servirà soprattutto a verificare che il sistema soddisfi le richieste del committente.

6.1 Testing di unità

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. Gli stati erronei scovati in questa, come in qualsiasi altra fase di testing, che comporteranno un fallimento del sistema dovranno essere tempestivamente comunicati agli sviluppatori al fine di correggerli e ripristinare il testing al più presto. Effettueremo dei test ai metodi delle classi utilizzando DBUnit, poiché i metodi che andremo a testare si basano su utilizzo di query, restituendo i risultati di quest'ultime.

6.2 Testing d'integrazione

Per l'implementazione della piattaforma seguiremo l'approccio non incrementale, detto Big-bang Testing, che prevede l'implementazione completa del sistema che presenterà già tutti i servizi offerti in una singola fase, che sarà poi testata e via via raffinata fino al raggiungimento di un risultato soddisfacente.

6.3 Testing di sistema

Lo scopo di questa fase di testing è quello di dimostrare che il sistema soddisfi effettivamente i requisiti richiesti e sia, quindi, pronto all'uso. Come per il testing di unità, si cercherà di testare le funzionalità più importanti per l'utente e quelle che hanno una maggiore probabilità di fallimento. Al fine di minimizzare il numero di test cases, i possibili input verranno partizionati in category partition e per ogni classe verrà selezionato un test case. Si noti che, come per il testing di unità, si procederà attraverso tecnica Black-Box. Il testing verrà effettuato con SeleniumIDE.

7. Materiale per il testing

L'hardware necessario per l'attività di test è un pc, dato che il database del sistema è integrato nel pc.

8. Test Cases

8.1 Gestione Autenticazione

8.1.1 Login

Parametro: Username **Formato:** Qualsiasi Carattere	
Lunghezza[LU]	1. ==0 [error] 2. !=0 [property lunghezza LUOK]

Parametro: Password **Formato:** Qualsiasi Carattere	
Lunghezza[LP]	1. ==0 [error] 2. !=0 [property lunghezza LPOK]

Codice	Combinazione	Esito
TC_1.0_0	LU1	Errato
TC_1.0_1	LU2, LP1	Errato
TC_1.0_2	LU2, LP2	Corretto

8.2 Iscrizione al sistema

8.2.1 Registrazione

Parametro: Codice Fiscale **Formato:** [0-9a-zA-Z]	
Lunghezza[LF]	1. !=16 [error] 2. ==16 [property lunghezza LFOK]
Formato[FF]	1. Non rispetta il formato [iflunghezzaLFOK] [error] 2. Rispecchia il formato [iflunghezzaLFOK][property formato FFOK, rispecchia il formato [0-9a-zA-Z]]

Parametro: Nome **Formato:** [a-zA-Z]	
Lunghezza[LN]	1. <=2 [error] 2. >2 [property lunghezza LNOK]
Formato[FN]	1. Non rispetta il formato [iflunghezzaLNOK] [error] 2. Rispecchia il formato [iflunghezzaLNOK] [property formato FNOK, rispecchia il formato [a-zA-Z]]

Parametro: Cognome **Formato:** [a-zA-Z]	
Lunghezza[LC]	1. <=2 [error] 2. >2 [property lunghezza LCOK]
Formato[FC]	1. Non rispetta il formato [iflunghezzaLCOK] [error] 2. Rispecchia il formato [iflunghezzaLCOK] [property formato FCOK, rispecchia il formato [a-zA-Z]]

Parametro: Telefono **Formato:** [0-9]	
Lunghezza[LT]	1. !=10 [error] 2. ==10 [property lunghezza LTOK]
Formato[FT]	1. Non rispetta il formato [iflunghezzaLTOK] [error] 2. Rispecchia il formato [iflunghezzaLTOK] [property formato FTOK, rispecchia il formato [0-9]]

Parametro: Indirizzo **Formato:** [0-9a-zA-Z]	
Lunghezza[LI]	1. <6 or >=50[error] 2. >=6 and <50 [property lunghezza LIOK]
Formato[FI]	1. Non rispetta il formato [iflunghezzaLIOK] [error] 2. Rispecchia il formato [iflunghezzaLIOK] [property formato FIOK, rispecchia il formato [0-9a-zA-Z]]

Parametro: Città **Formato:** [a-zA-Z]	
Lunghezza[LCT]	1. <=1 or >=30[error] 2. >1 and <30[property lunghezza LCTOK]
Formato[FCT]	1. Non rispetta il formato [iflunghezzaLCTOK] [error] 2. Rispecchia il formato [iflunghezzaLCTOK] [property formato FCTOK, rispecchia il formato [a-zA-Z]]

Parametro: E-Mail **Formato:** \w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})	
Lunghezza[LE]	1. <7 [error] 2. >=7 [property lunghezza LEOK]
Formato[FE]	1. Non rispetta il formato [iflunghezzaLEOK] [error] 2. Rispecchia il formato [iflunghezzaLEOK] [property formato FEOK, rispecchia il formato**:** \w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})]

Parametro: ID Utente	
Lunghezza[LID]	1. <3 or >10 [error] 2. >=3 and <=10 [property lunghezza LIDOK]

Parametro: Password	
Lunghezza[LP]	1. <5 or >10 [error] 2. >=5 and <=10 [property lunghezza LPOK]

Codice	**Combinazione**	**Esito**
TC_2.2_0	LF1	Errato
TC_2.2_1	LF2, FF1	Errato
TC_2.2_2	LF2, FF2, LN1	Errato
TC_2.2_3	LF2, FF2, LN2, FN1	Errato
TC_2.2_4	LF2, FF2, LN2, FN2, LC1	Errato
TC_2.2_5	LF2, FF2, LN2, FN2, LC2, FC1	Errato
TC_2.2_6	LF2, FF2, LN2, FN2, LC2, FC2, LT1	Errato
TC_2.2_7	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT1	Errato
TC_2.2_8	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI1	Errato
TC_2.2_9	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI1	Errato
TC_2.2_10	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI2, LCT1	Errato
TC_2.2_11	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI2, LCT2, FCT1	Errato
TC_2.2_12	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI2, LCT2, FCT2, LE1	Errato
TC_2.2_13	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI2, LCT2, FCT2, LE2, FE1	Errato
TC_2.2_14	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI2, LCT2, FCT2, LE2, FE2, LID1	Errato
TC_2.2_15	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI2, LCT2, FCT2, LE2, FE2, LID2, LP1	Errato
TC_2.2_16	LF2, FF2, LN2, FN2, LC2, FC2, LT2, FT2, LI2, FI2, LCT2, FCT2, LE2, FE2, LID2, LP2	Corretto

8.3 Gestione dei cataloghi

8.3.1 Aggiunta nuovi prodotti nei cataloghi (accessorio)

Parametro: Nome Prodotto **Formato:** Qualsiasi Carattere	
Lunghezza[LNP]	1. ==0 [error] 2. != 0 [property lunghezza LNPOK]

Parametro: Descrizione **Formato:** Qualsiasi Carattere	
Lunghezza[LD]	1. ==0 [error] 2. != 0 [property lunghezza LDOK]

Codice	**Combinazione**	**Esito**
TC_3.3_0	LNP1	Errato
TC_3.3_1	LNP2, LD1	Errato
TC_3.3_2	LNP2, LD2	Corretto

8.3.2 Aggiunta nuovi prodotti nei cataloghi (videogioco)

Parametro: Nome Prodotto **Formato:** Qualsiasi Carattere	
Lunghezza[LNP]	1. ==0 [error] 2. != 0 [property lunghezza LNPOK]

Parametro: Genere **Formato:** [a-zA-Z]	
Lunghezza[LG]	1. ==0 [error] 2. !=0 [property lunghezza LGOK]
Formato[FG]	1. Non rispetta il formato [iflunghezzaLGOK] [error] 2. Rispecchia il formato [iflunghezzaLGOK] [property formato FGOK, rispecchia il formato [a-zA-Z]]

Parametro: Descrizione **Formato:** Qualsiasi Carattere	
Lunghezza[LD]	1. ==0 [error] 2. != 0 [property lunghezza LDOK]

Codice	Combinazione	Esito
TC_3.3_3	LNP1	Errato
TC_3.3_4	LNP2, LG1	Errato
TC_3.3_5	LNP2, LG2, FG1	Errato
TC_3.3_6	LNP2, LG2, FG2, LD1	Errato
TC_3.3_7	LNP2, LG2, FG2, LD2	Corretto

8.3.3 Aggiunta nuovi prodotti nei cataloghi (Console)

Parametro: Nome Prodotto **Formato:** Qualsiasi Carattere	
Lunghezza[LNP]	1. ==0 [error] 2. != 0 [property lunghezza LNPOK]

Parametro: Descrizione **Formato:** Qualsiasi Carattere	
Lunghezza[LD]	1. ==0 [error] 2. != 0 [property lunghezza LDOK]

Codice	Combinazione	Esito
TC_3.3_8	LNP1	Errato
TC_3.3_9	LNP2, LD1	Errato
TC_3.3_10	LNP2, LD2	Corretto

8.4 Gestione profilo utente

8.4.1 Modifica indirizzo di spedizione

Parametro: Via **Formato:** [0-9a-zA-Z]	
Lunghezza[LV]	1. <6 or >50 [error] 2. >=6 and <=50[property lunghezza LVOK]
Formato[FV]	1. Non rispetta il formato [iflunghezzaLVOK] [error] 2. Rispecchia il formato [iflunghezzaLVOK] [property formato FVOK, rispecchia il formato [0-9a-zA-Z]]

Parametro: Città **Formato:** [a-zA-Z]	
Lunghezza[LC]	1. <=1 or >30[error] 2. >1 and <=30 [property lunghezza LVOK]
Formato[FC]	1. Non rispetta il formato [iflunghezzaLCOK] [error] 2. Rispecchia il formato [iflunghezzaLCOK] [property formato FCOK, rispecchia il formato [a-zA-Z]]

Codice	**Combinazione**	**Esito**
TC_4.6_0	LV1	Errato
TC_4.6_1	LV2, FV1	Errato
TC_4.6_2	LV2, FV2, LC1	Errato
TC_4.6_3	LV2, FV2, LC2, FC1	Errato
TC_4.6_4	LV2, FV2, LC2, FC2	Corretto

8.4.2 Modifica password

Parametro: Vecchia Password **Formato:** Qualsiasi Carattere	
Lunghezza[LVP]	1. ==0 [error] 2. !=0 [property lunghezza LVPOK]
Uguaglianza[UVP]	1. != password corrente utente[iflunghezzaLVPOK] [error] 2. ==password utente [iflunghezzaLVPOK][property Uguaglianza UVPOK]

Parametro: Nuova Password **Formato:** Qualsiasi Carattere	
Lunghezza[LNP]	1. ==0 or >10 or <1 [error] 2. !=0 and <=10 and >1[property lunghezza LNPOK]
Disuguaglianza[DNP]	1. ==Vecchia Password [error] 2. != Vecchia Password [iflunghezza LNPOK][property disuguaglianza DNPOK]

Parametro: Conferma Password **Formato:** Qualsiasi Carattere	
Lunghezza[LCP]	1. ==0 [error] 2. !=0[property lunghezza LCOK]
Uguaglianza[UCP]	1. != Nuova Password[iflunghezzaLCPOK] [error] 2. ==Nuova Password [iflunghezzaLCPOK] [property Uguaglianza UCPOK]

Codice	**Combinazione**	**Esito**
TC_4.6_5	LVP1	Errato
TC_4.6_6	LVP2, UVP1	Errato
TC_4.6_7	LVP2, UVP2, LNP1	Errato
TC_4.6_8	LVP2, UVP2, LNP2, DNP1	Errato
TC_4.6_9	LVP2, UVP2, LNP2, DNP2, LCP1	Errato
TC_4.6_10	LVP2, UVP2, LNP2, DNP2, LCP2, UCP1	Errato
TC_4.6_11	LVP2, UVP2, LNP2, DNP2, LCP2, UCP2	Corretto

9. Pianificazione del testing

Il team per il testing deve essere composto da persone che hanno una completa e approfondita conoscenza del sistema e delle tecniche di testing con i documenti associati, quali Test plan e Test case specification. Tali tecniche devono essere applicate nei tempi, nel budget e nei vincoli di qualità stabiliti. Il team è responsabile dell'attività di testing e quindi della ricerca di fault. Il sistema revisionato è poi testato nuovamente non solo per verificare se gli errori trovati in precedenza sono stati eliminati ma soprattutto per verificare che non ne siano stati introdotti dei nuovi. L'attività di testing è fondamentale nello sviluppo di un sistema software in quanto la mancanza di tale attività o una cattiva interpretazione di essa può portare al completo fallimento del sistema. Il team dedicato all'attività di testing sarà composto dall'intero gruppo di progetto.