Università degli Studi di Salerno

Corso di Ingegneria del Software

Core Shirt
Test-Plan
Versione 1.0

Data: 16/12/2016

Partecipanti:

Nome	Matricola
Ranieri Salvatore	0512103286
Russo Gennaro	0512103160
Turi Pasquale	0512102860
Urlo Mario Pio	0512103133

Indice

1. Introduzione

Lo scopo di questo documento è quello di analizzare e gestire lo sviluppo e le attività di testing riguardanti il software Core Shirt. Questa sessione di lavoro deve verificare il corretto funzionamento del sistema in diversi casi, studiati appositamente per mettere alla prova ogni singola funzionalità e caratteristica del sistema, al fine di ottenere un corretto funzionamento. I risultati di questi test saranno utilizzati per capire dove bisognerà intervenire, e quindi correggere eventuali errori o apportare modifiche per il miglioramento dei vari sottosistemi. Il processo verrà iterato fino a che non si otterranno i risultati attesi in accordo con i tempi di sviluppo previsti.

2. Relazione con gli altri documenti

Per verificare il corretto funzionamento del sistema sono stati predisposti dei test cases basati sulle funzionalità individuate sia nel documento inerente gli scenari che su quello di analisi dei requisiti (RAD).
Il documento a cui facciamo riferimento è quindi:
RAD 2.0;

3. Panoramica del sistema

Il sistema ha come obiettivo la creazione di un sito di vendita di T-shirt.
Il suo sviluppo è stato concepito in maniera modulare e pertanto è stato possibile dividerlo in sottosistemi (si veda System Design Document); questo approccio apporta vantaggi dal punto di vista della manutenzione e sarà fondamentale nella fase di testing, in quanto si potrà individuare un errore nel singolo sottosistema effettuando un numero limitato di test.
Una volta testati individualmente tutti i sottosistemi andremo a testare la loro integrazione utilizzando un approccio di tipo “Bottom-up”. Il sottosistema a livello gerarchico più basso sarà testato individualmente. Successivamente, sarà testata la sua integrazione con la componente a livello di gerarchia più alto. Questo procedimento sarà iterato finché non avremo integrato tutti i sottosistemi per verificare che l'intero sistema funzioni correttamente.

4. Funzionalità da testare e non

Le componenti prese in considerazione nella fase di testing rappresentano le funzionalità core del sistema, ovvero:

● ***Gestione Account***
Questa funzionalità permette ad un utente di effettuare l'accesso e di fare il logout dal sistema; permette agli admin, invece, di autenticarsi ed accedere alle funzionalità a loro consentite.
➢ Saranno testate le funzionalità di login.

● ***Gestione Carrello e Checkout***
Tale gestione consente di visualizzare, inserire ed eliminare dati relativi agli articoli presenti nel carrello e procedere successivamente al checkout per terminare l'acquisto.
➢ Saranno testate le funzionalità di inserimento di un nuovo elemento, della sua eliminazione e del procedimento per la terminazione dell'acquisto.

● ***Gestione Articoli***
Tale gestione consente di visualizzare, inserire, modificare ed eliminare dati relativi alle T-shirt.
➢ Sarà testata la funzionalità di inserimento delle quantità da parte di un utente di una determinata T-shirt.

● ***Gestione Acquisti***
Tale gestione consente di visualizzare ed evadere gli ordini in entrata ed in uscita.
➢ Sarà testata la funzionalità di visualizzazione dei dettagli di un ordine e della corretta evasione di un ordine.

● ***Gestione Dipendenti***
Tale gestione consente di visualizzare, inserire, modificare ed eliminare dati relativi ai dipendenti.
➢ Sarà testata la funzionalità di inserimento, di eliminazione e di modifica dei dati di un dipendente.

5. Pass/Fail criteria

Il testing ha successo se l'output osservato è diverso dall'output atteso: ciò significa che la fase di testing avrà successo se individuerà una fallimento. 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. Al contrario, il testing fallirà se l'output osservato sarà uguale all'oracolo.

6. Approccio

Nella sessione di testing del sistema verrà utilizzato un approccio di tipo “BLACK BOX”, che
prevede che i test vengano effettuati in maniera da non scendere nei dettagli del codice, ma basandosi sulle specifiche delle funzionalità da testare.
L’approccio alla fase di testing si compone di tre fasi:
- Testing di unità, che controlla i singoli componenti (classi, metodi)
- Testing di integrazione, che va a testare l'integrazione dei vari sottosistemi
- Testing di sistema: test funzionale, che andrà a verificare la funzionalità dell’intero sistema assemblato.

6.1 Testing di unità

Con il testing di unità verrà effettuato un controllo delle varie classi e metodi del sistema, quindi saranno ricercate le condizioni di fallimento andando ad evidenziare gli errori. Il testing di unità, sarà eseguito dal team di sviluppo attraverso l’implementazione di classi di test utilizzando il framework JUnit. In particolare, per ogni classe che esegue operazioni complesse sarà sviluppata la relativa classe JUnit.

6.2 Testing di integrazione

Con il testing di integrazione si effettuerà un controllo sull’integrazione delle varie componenti del sistema. Si adotterà una strategia di tipo “Bottom-up”. Per effettuare questi test di integrazione, spesso saranno necessari l’utilizzo di driver dato che tale strategia va ad integrare passo passo i sottosistemi partendo dal layer che si trova più in basso nella scala gerarchica. La lista dei casi di test verrà fuori dall’applicazione del metodo del Category Partition.

6.3 Testing di sistema

Con il testing di sistema verrà effettuato un controllo della correttezza dell’intero sistema. E’ da considerare il testing più critico, in quanto può risultare molto complesso andare alla ricerca di eventuali errori, essendo impegnati tutti i sottosistemi. Questo test sarà effettuato utilizzando il framework Selenium, che mette a disposizione strumenti per il controllo di sistemi web-based.

7. Test cases

Per sviluppare i test cases sarà utilizzato il metodo del Category Partition. Questo metodo consiste nell'identificare per ogni funzionalità da testare dei parametri; per ogni parametro verranno individuate delle categorie, le quali poi saranno suddivise in scelte. Alle scelte verrà assegnato un valore.
I test cases verranno definiti nel documento di Test Cases Specification (TCS).

8. Specifica dei test cases

8.1 Gestione Account

8.1.1 Login

Parametro→username	
Categorie	Scelte
lunghezza le	1: lunghezza < 6 [errore]
	2: lunghezza >6 [property **lunghezzaLEok** ,lunghezza username >6]
esiste ee	1: esiste nel DB [if **lunghezzaLEok**]
	2: non esiste nel DB [if **lunghezzaLEok**][errore]

Parametro→password	
Categorie	Scelte
lunghezza lp	1: lunghezza < 6 [errore]
	2: lunghezza >6 [property **lunghezzaLPok** ,lunghezza username >6]
esiste ee	1: esiste nel DB [if **lunghezzaLPok**]
	2: non esiste nel DB [if **lunghezzaLPok**][errore]

Codice	**Combinazione**	**Esito**
TC_Login_01	le2ee1lp2pp1	accesso
TC_Login_02	le2ee2	errore
TC_Login_03	le2ee1lp2pp2	errore
TC_Login_04	le2ee2lp2pp2	errore

8.2 Gestione carrello e Checkout

8.2.1 Checkout

Parametro → Nome	
formato: [A-Za-z]	
Categorie	**Scelte**
lunghezza ln	1: lunghezza < 2 [errore]
	2: lunghezza 2-25 [property **lunghezzaLNok**,lunghezza nome da 2 a 25]
	3: lunghezza > 25 [errore]
formato fn	1: rispetta il formato [if **lunghezzaLNok**] [property **formatoFNok,** rispetta il formato [A-Za-z]]
	2: non rispetta il formato [if **lunghezzaLNok**] [errore]
Parametro → Cognome	
formato: [A-Za-z]	
Categorie	**Scelte**
lunghezza lc	1: lunghezza < 2 [errore]
	2: lunghezza 2-30 [property **lunghezzaLCok**,lunghezza cognome da 2 a 30]

	3: lunghezza > 30 [errore]
formato fc	1: rispetta il formato [if **lunghezzaLCok**] [property **formatoFCok**, rispetta il formato [A-Za-z]]
	2: non rispetta il formato [if **lunghezzaLCok**] [errore]
Parametro → e-mail	
formato: [A-Za- z0-9]@[A-Za-z.]	
Categorie	**Scelte**
lunghezza le	1: lunghezza < 15 [errore]
	2: lunghezza 15-30 [property **lunghezzaLEok**,lunghezza e-mail da 15 a 30]
	3: lunghezza > 30 [errore]

formato fe	1: rispecchia il formato [if **lunghezzaLEok**][property **formatoFEok**,rispecchia il formato [A-Za- z0-9]@[A-Za-z.]]
	2: non rispetta il formato [if **lunghezzaLEok**][errore]
esiste ee	1: esiste nel DB [if **lunghezzaLEok** and **formatoFEok**][errore]
	2.non esiste nel DB [if **lunghezzaLEok** and **formatoFEok**][property nonEsiste**Eok**]

Parametro → Indirizzo Spedizione	
Categorie	**Scelte**
lunghezza li	1: lunghezza < 7 [errore]
	2: lunghezza 7-30 [property **lunghezzaLIok**,lunghezza indirizzo da 7 a 30]
	3: lunghezza > 30 [errore]
Parametro → CAP	
formato [0-9]	
Categorie	**Scelte**
lunghezza lca	1: lunghezza < 5 [errore]
	2: lunghezza = 5 [property **lunghezzaLCAok**, lunghezza CAP uguale a 5]
	3: lunghezza > 5 [errore]
formato fca	1: rispecchia il formato [if **lunghezzaLCAok**][property
	formatoFCAok,rispecchia il formato [0-9]]
	2: non rispetta il formato [if **lunghezzaLCAok**][errore]
Parametro → Numero Carta	

formato [0-9]	
Categorie	**Scelte**
lunghezza lnt	1: lunghezza < 16 [errore]
	2: lunghezza = 16 [property **lunghezzaLNTok**, lunghezza Numero Carta uguale a 16]
	3: lunghezza > 16 [errore]
formato fnt	1: rispecchia il formato [if **lunghezzaLNTok**][property **formatoFNTok**,rispecchia il formato [0-9]]
	2: non rispetta il formato [if **lunghezzaLNTok**][errore]

CODICE	**COMBINAZIONE**	**ESITO**
TC_Checkout_0 1	ln2fn1lc2fc1le2fe1ee2li2lca2fca1lnt2fnt1	registrazione
TC_Checkout_0 2	ln2fn1lc2fc1le2fe2	errore
TC_Checkout_0 3	ln2fn1lc2fc1le2fe1ee1	errore
TC_Checkout_0 4	ln1	errore
TC_Checkout_0 5	ln2fn1lc1	errore
TC_Checkout_0 6	ln2fn1lc2fc1le2fe1ee2li1	errore

TC_Checkout_0 7	ln2fn1lc2fc1le2fe1ee2li2lca1	errore
TC_Checkout_0 8	ln2fn1lc2fc1le2fe1ee2li2lca2fca1lnt1	errore
TC_Checkout_0 9	ln3	errore
TC_Checkout_1 0	ln2fn1lc3	errore
TC_Checkout_1 1	ln2fn1lc2fc1le2fe1ee2li3	errore
TC_Checkout_1 2	ln2fn1lc2fc1le2fe1ee2li2lca3	errore
TC_Checkout_1 3	ln2fn1lc2fc1le2fe1ee2li2lca2fca1lnt3	errore
TC_Checkout_1 4	ln2fn2	errore
TC_Checkout_1 5	ln2fn2lc2fc2	errore
TC_Checkout_1 6	ln2fn1lc2fc1le2fe1ee2li2lca2fca1lnt2fnt2	errore
TC_Checkout_1 7	ln2fn1lc2fc1le1	errore
TC_Checkout_1 8	ln2fn1lc2fc1le3	errore
TC_Checkout_1 9	ln2fn1lc2fc1le2fe1ee2li2lca2fca2	errore

8.3 Gestione Articoli

8.3.1 Inserimento Articolo

Parametro → Nome **formato: [A-Za-z, 0-9]**	
Categorie	**Scelte**
lunghezza ln	1: lunghezza < 2 [errore]
	2: lunghezza 2-25 [property **lunghezzaLNok**,lunghezza nome da 2 a 25]
	3: lunghezza > 25 [errore]
formato fn	1: rispetta il formato [if **lunghezzaLNok**] [property **formatoFNok,** rispetta il formato [A-Za-z, 0-9]]
	2: non rispetta il formato [if **lunghezzaLNok**] [errore]

Parametro → Quantità **formato: [0-9]**	
Categorie	**Scelte**
lunghezza lq	1: lunghezza <1 [errore]
	2: lunghezza 1-4 [property **lunghezzaLQok**,lunghezza nome da 1 a 4]
	3: lunghezza > 4 [errore]
formato fq	1: rispetta il formato [if **lunghezzaLQok**] [property **formatoFQok,** rispetta il formato [0-9]]

	2: non rispetta il formato [if **lunghezzaLQok**] [errore]

CODICE	COMBINAZIONE	ESITO
TC_Inserimento_01	ln2fn1fq1lq2	accesso
TC_Inserimento_02	ln1fn1fq1lq2	errore
TC_Inserimento_03	ln3fn1fq1lq2	errore
TC_Inserimento_04	Ln2fn2fq1lq2	errore
TC_Inserimento_05	Ln2fn1fq1lq1	errore
TC_Inserimento_06	Ln2fn1fq1lq3	errore
TC_Inserimento_07	Ln2fn2fq2lq2	errore

8.4 Gestione Dipendenti

8.4.1 Inserimento Dipendente

Parametro → Nome	
formato: [A-Za-z]	
Categorie	**Scelte**
lunghezza ln	1: lunghezza < 2 [errore]
	2: lunghezza 2-25 [property **lunghezzaLNok**,lunghezza nome da 2 a 25]
	3: lunghezza > 25 [errore]
formato fn	1: rispetta il formato [if **lunghezzaLNok**] [property **formatoFNok,** rispetta il formato [A-Za-z]]
	2: non rispetta il formato [if **lunghezzaLNok**] [errore]

Parametro → Cognome	
formato: [A-Za-z]	
Categorie	**Scelte**
lunghezza lc	1: lunghezza < 2 [errore]
	2: lunghezza 2-30 [property **lunghezzaLCok**,lunghezza cognome da 2 a 30]
	3: lunghezza > 30 [errore]
formato fc	1: rispetta il formato [if **lunghezzaLCok**] [property **formatoFCok**, rispetta il formato [A-Za-z]]
	2: non rispetta il formato [if **lunghezzaLCok**] [errore]
Parametro → Codice Fiscale	
formato: [A-Za-z, 0-9]	
Categorie	**Scelte**
lunghezza le	1: lunghezza <16 [errore]
	2: lunghezza =16 [property **lunghezzaLEok**,lunghezza codice fiscale 16]
	3: lunghezza >16 [errore]

formato fe	1: rispecchia il formato [if **lunghezzaLEok**][property **formatoFEok**,rispecchia il formato [A-Za-z0-9]
	2: non rispetta il formato [if **lunghezzaLEok**][errore]
Parametro → Stipendio	
formato [0-9]	
Categorie	**Scelte**
lunghezza lca	1: lunghezza < 3 [errore]
	2: lunghezza 3-4 [property **lunghezzaLCAok**, lunghezza Stipendio tra 3 e 4]
	3: lunghezza > 4 [errore]
formato fca	1: rispecchia il formato [if **lunghezzaLCAok**][property **formatoFCAok**, rispecchia il formato [0-9]]
	2: non rispetta il formato [if **lunghezzaLCAok**][errore]
Parametro → Username	
formato [A-Za-z]	
Categorie	**Scelte**
lunghezza lnt	1: lunghezza <2 [errore]
	2: lunghezza 2-16[property **lunghezzaLNTok**, lunghezza Username tra 2 e 16]

	3: lunghezza > 16 [errore]
formato fnt	1: rispecchia il formato [if **lunghezzaLNTok**][property **formatoFNTok**,rispecchia il formato [A-Za-z]
	2: non rispetta il formato [if **lunghezzaLNTok**][errore]

Parametro →Password	
formato [A-Za-z] [0-9]	
Categorie	**Scelte**
lunghezza lnt	1: lunghezza <8 [errore]
	2: lunghezza 8-16[property **lunghezzaLNTok**, lunghezza Password tra 8 e 16]
	3: lunghezza > 16 [errore]
formato fnt	1: rispecchia il formato [if **lunghezzaLNTok**][property **formatoFNTok**,rispecchia il formato [A-Za-z] [0-9]]
	2: non rispetta il formato [if **lunghezzaLNTok**][errore]