Test Plan Specification

"Maggico Car & Motorbike Parts"

Raffaele Coscione 0512106006

Vincenzo Tortora 0512102104

Francesco Carotenuto 0512104798

Giovanni Renzulli 0512105730

Top Manager

Nome
Prof. Andrea De Lucia

Partecipanti

Nome	Matricola
Raffaele Coscione	0512106006
Vincenzo Tortora	0512102104
Francesco Carotenuto	0512104798
Giovanni Renzulli	0512105730

History

Data	Versione	Cambiamenti	Autore
30/01/2020	1.0	Stesura iniziale delle componenti	Francesco Carotenuto
06/02/2020	1.1	Completamento e revisione del documento	Francesco Carotenuto

Test Plan

Sommario

1. Introduzione

Lo scopo del sistema è quello di pianificare l'attività di testing all'interno della piattaforma YouLearn con lo scopo di verificare se esistono differenze tra il comportamento atteso) con il comportamento osservato. In questa attività andremo ad individuare i possibili errori nel codice sorgente causanti incident all'interno del sistema prima che l'utente finale esegua delle interazioni con il sistema. Le attività di testing sono organizzate in modo da poter verificare il funzionamento dei seguenti sottosistemi:

1. Gestione utente
2. Gestione Catalogo
3. Gestione Carrello

Si noti che non vi saranno attività di testing per i sottosistemi non descritti all'interno dell'Object design document e sviluppati in fase di implementazione.

Quindi i sottosistemi seguenti saranno esclusi dalle verifiche delle funzionalità di sistema:

1. Gestione pagamenti
2. Gestione Amministratore

I documenti precedenti al testing sono di grande importanza per la fase di testing, questo perché ogni documento, sviluppato secondo un differente livello di progettazione, saranno di grande importanza per la correttezza del testing.

2. Documenti Correlati

La documentazione precedente è strettamente correlata con la pianificazione dei test, questo perché già nei documenti precedenti abbiamo definito come alcuni servizi devono funzionare. Questa caratteristica prevede di rilevare informazioni come i comportamenti attesi durante l'esecuzione di alcune funzionalità. Vediamo, quindi, che la fase di testing prevede di verificare se ci siano differenze tra il funzionamento del sistema che si è progettato e il reale funzionamento del sistema implementato.

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

La relazione tra la fase di testing e la fase di analisi dei requisiti si basano sui requisiti funzionali e non funzionali descritti nel RAD che devono essere rispettati dal sistema durante la fase di testing.

2.2 Relazione con il System design document (SDD)

La relazione tra la fase di testing e la fase di design del sistema si basa sulla suddivisione del sistema in sottosistemi e la divisione in strati (Presentation layer, Application layer, Storage layer). Il testing deve essere fedele alla suddivisione progettata in fase di design in modo tale da rimanere coerente con il sistema che si è progettati.

2.3 Relazioni con l'Object design document (ODD)

La relazione fra la fase di testing e la fase di object design si basa principalmente sul riferimento della verifica dei contratti e dei componenti raffinati all'interno di tale documento.

3 Panoramica del Sistema

La struttura definita dentro il System design document (SDD) prevede un architettura clientserver three-trial basata sulla suddivisione in 3 strati:

- Presentation Layer – Strato che si occupa della presentazione dei risultati di una computazione agli utenti del sistema e di raccogliere gli input degli utenti
- Application layer – si occupa di fornire all'applicazione specifiche funzionalità
- Storage Layer – si occupa della gestione dei database di sistema

Il sistema, inoltre, è stato diviso in sottosistemi a secondo della gestione di una particolare categoria di servizi. La suddivisione è fatta nel seguente modo:

- Gestione utente
- Gestione Catalogo
- Gestione Carrello
- Gestione pagamenti
- Gestione Amministratore

Quasi tutte le gestioni prevedono operazioni di inserimento, modifica, cancellazione, visualizzazione e ricerca di dati. Saranno proprio tali funzionalità ad essere oggetto di testing del sistema. Ricordiamo, infine, che solamente i primi tre sottosistemi verranno testati prima del rilascio del progetto, qualsiasi servizio legato alla gestione dei pagamenti e di mail dipendenti dai sottosistemi da testare saranno sviluppati come stub o driver a seconda del tipo di dipendenza.

4 Funzionalità da testare e non testare

Di seguito saranno elencate tutte le funzionalità da testare in relazione al sottosistema a cui appartengono.

1. Gestione Autenticazione
    - Login
2. Gestione utente
    - Creazione Admin
    - Cambio password

3. Gestione Registrazione
    - Registrazione
4. Gestione Catalogo
    - Crea prodotto
    - Modifica prodotto
    - Visualizzazione prodotto
    - rimozione prodotto
    - Ricerca Prodotto
5. Gestione Carrello
    - Aggiungi prodotto
    - Rimozione prodotto
    - Checkout prodotto

5 Criteri Pass/Failed

I dati di test verranno raggrupati in classi di equivalenza in grado di raggruppare elementi con le stesse caratteristiche, per tali classi sarà sufficiente testare le funzionalità per un singolo elemento per classe. Un input supererà il test se il comportamento atteso e quello risultate combaceranno, ossia se l'output desiderato è uguale all'output ottenuto. Il responsabile del testing saprà a priori quale sia l'output desiderato.

6 Approccio

Verrà applicata una strategia di testing bottom-up. Si inzierà con il testing di unità dei singoli componenti, in modo da testare nello specifico le unità atomiche nella loro correttezza. Seguirà il testing di integrazione che focalizzerà l'attenzione sulle interfacce dell'unità. Infine, verrà eseguito il testing di sistema che vedrà la verifica del comportamento dell'interno sistema assemblato partendo dalle sue componenti principali. Il testing di sistema è importante per verificare se le caratteristiche richieste dal committente vengono rispettate o meno.

6.1 Testing di unità

Durante questa fase, verranno ricercate le condizioni di fallimento all'interno delle singole componenti e, usando test driver e stub raffiguranti implementazioni parziali di componenti che dipendono o da cui dipendono le componenti testate, si verificano i comportamenti della singola unità. La strategia utilizzata per il testing si baserà esclusivamente sulla tecnica Black-Box. Questa scelta strutturerà il testing unitario in un'analisi Input/Ouput delle singole componenti andando ad astrarre la verifica della struttura interna. Per minimizzare i casi di test, gli input verranno divisi in classi di equivalenza e ogni componente avrà un singolo caso di test per ogni classe di equivalenza strutturata. In questa fase, quindi, si avrà particolare attenzione sulla suddivisione delle classi degli input così da poter verificare ogni componente su ogni possibile tipo di input del dominio. La gestione del caso di errore, ossia lo stato in cui il comportamento atteso non è equivalente al comportamento ottenuto, comporterà un aggiornamento del documento di Incident Report; Tale documento verrà utilizzato per informare gli sviluppatori della presenza di un fault in modo tale da poterlo correggerlo tempestivamente per poi ripassare ad una verifica della correzione. La strategia del report di fault verrà estesa anche per le altre fasi di testing.

6.2 Testing di integrazione

Questa fase di test prevede l'aggregazione delle singole componenti e il loro testing adottando una strategia black-box. Tale fase prevede l'iterazione con l'interfaccia del sistema legato ai servizi delle funzionalità da testare, tale testing si basarà sulla ricerca di possibili fault all'interno delle funzionalità del sistema e, in particolare, nella logica applicativa del software.

6.3 Testing di sistema

In questa fase verrà utilizzata nuovamente la strategia black-box per il testing delle funzionalità dell'intero sistema e verificare se i requisiti e i vincoli di progettazione sono stati rispettati. In seguito a tale test, se superato in maniera corretta, si avrà un sistema pronto all'uso per l'utente finale. Ci si concentrerà principalmente sul testing delle funzionalità principali basate sulle priorità dei requisiti specificati durante la fase di analisi.

7 Sospensione e ripresa

7.1 Criteri di sospensione

La fase di testing di sistema verrà sospesa quando si raggiungerà un compromesso di qualità del prodotto e costi delle attività di testing. Il testing verrà propagato il più possibile in modo da garantire una migliore affidabilità del funzionamento del sistema andando, però, a considerare i tempi di consegna del progetto.

7.2 Criteri di ripresa

I criteri di ripresa rappresentano le azioni da apportare in seguito ad una modifica o ad una correzione di qualche componente durante la fase di testing. Tale attività prevede la creazione di nuovi test case sottoposti nuovamente al sistema per verificare la correttezza della modifica apportata.

8 Materiale per il testing

L'hardware necessario per l'attività di testing deve essere necessariamente un pc. Non vi è bisogno di una connessione poiché ogni componente può essere testata in locale.

9 Test Cases

9.1 Gestione Utente

9.1.1 Login

9.1.1.1 Category Partition

Parametro: Username Formato: [A-Za-z0-9._%+-]	
DatabaseRaggiungibile[DBOK]	• Database non raggiungibile [error] • Database raggiungibile[property DatabaseRaggiungibile DBOK
UsernamePresente[UNOK]	• Database raggiungibile[if DatabaseRaggiungibileDBOK] • Username non presente nel database[error] • Username presente nel database[UNOK]

Parametro: Password Formato: [A-Za-z0-9._%+-]	
DatabaseRaggiungibile[DBOK]	• Database non raggiungibile [error] • Database raggiungibile[property DatabaseRaggiungibileDBOK]
PasswordCorrispondente[PWOK]	• Database raggiungibile[if DatabaseRaggiungibileDBOK] • Password non corrispondente[error] • Password corrispondente[PWOK]

9.2 Gestione Utente

9.2.1 Cambio Password

9.2.1.1 Category Partition

Parametro: Password Formato: [A-Za-z0-9._%+-]	
DatabaseRaggiungibile[DBOK]	• Database non raggiungibile [error] • Database raggiungibile[property DatabaseRaggiungibileDBOK]
PasswordCorrispondente[PWCOK]	• Database raggiungibile[if DatabaseRaggiungibileDBOK] • Password non corrispondente[error] • Password corrispondente[PWCOK]

9.2.2 Creazione Admin

9.2.2.1 Category Partition

Parametro : Username Formato: [A-Za-z0-9._%+-]	
DatabaseRaggiungibile[DBOK]	• Database non raggiungibile [error] • Database raggiungibile[property DatabaseRaggiungibile DBOK
UsernamePresente[UNACOK]	• Database non raggiungibile [error] • Username non presente nel database[error] • Username presente nel database[UNACOK]

Parametro: Password Formato: [A-Za-z0-9._%+-]	
DatabaseRaggiungibile[DBOK]	• Database non raggiungibile [error] • Database raggiungibile[property DatabaseRaggiungibile DBOK
PasswordCorrispondente[PWACOK]	• Database raggiungibile[if DatabaseRaggiungibileDBOK] • Password non corrispondente[error] • Password corrispondente[PWACOK]

Parametro : eMail Formato: [A-Za-z0-9._%+-] @ Formato: [A-Za-z0-9._%+-]	
DatabaseRaggiungibile[DBOK]	• Database non raggiungibile [error] • Database raggiungibile[property DatabaseRaggiungibile DBOK]
eMailCorrispondente[EMACOK]	• Database non raggiungibile [error] • eMail non corrispondente[error] • email corrispondente[EMACOK]

9.3 Registrazione

9.3.1. Registrazione

9.3.1.1. Category Partition

Parametro: Username Formato: [A-Za-z0-9._%+-]	
Lunghezza[LTUSR]	• <5 AND >15 [error]

	• >= 5 AND <= 15 [property Lunghezza LTUSROK]
Formato[FTUSR]	•Rispetta la lunghezza[if property lunghezza LTUSROK] • Rispecchia il formato [A-Za-z09._%+-] [FTUSROK]

Parametro: Password Formato: [A-Za-z0-9._%+-]	
Lunghezza[LTPSW]	• <8 AND >40 [error] • >= 8 AND <= 40 [property Lunghezza LTPSWOK]
Formato[FTPSW]	• Rispetta la lunghezza[if property lunghezza LTPSWOK] • Rispecchia il formato [A-Za-z09._%+-] [FTPSWOK]

Parametro: eMail Formato: [A-Za-z0-9. _%+-] @ [A-Za-z0-9. -]. [A-Z]	
Lunghezza[LTEMA]	• <= 50 [property Lunghezza LTNMOK]
Formato[FTEMA]	• Rispetta la lunghezza[if property lunghezza LTEMAOK] • Rispecchia il formato [A-Za-z09._%+-] [FTEMAOK]

Parametro : Iban Formato: [IT-(2)0-9(1)A-Za-Z0-9]	
Lunghezza[LUIBN]	• =27[property Lunghezza LUIBNOK] • < 27 >[error]
Formato[FTIBN]	• Rispetta la lunghezza[if property lunghezza LUIBNOK] • Rispecchia il formato [IT-(2)0-9(1)A-Za-Z0-9] [FTIBNOK]

9.4 Gestione Catalogo

9.4.1 Crea Prodotto

9.4.1.1 Category Partition

Parametro: NomeProdotto Formato: [A-Za-z0-9._%+-]	
Lunghezza[LUNOP]	• 100>[error] • 100<=[property lunghezza LUNOPOK]
Formato[FTNOP]	• Rispetta la lunghezza[if property lunghezza LUIBNOK] • Rispetta il formato [A-Za-z0-9._%+-] [FTNOPOK]

Parametro: Descrizione Formato: [A-Za-z0-9._%+-]	
Lunghezza[LUDES]	• 100>[error] • 100<=[property lunghezza LUDESOK]
Formato[FTDES]	• Rispetta la lunghezza[if property lunghezza LUDESOK] • Rispetta il formato [A-Za-z0-9._%+-] [FTDESOK]

Parametro: prezzo Formato: [0-9]	
Formato [FPC]	• rispecchia il formato [0-9] [propertyformatoFPCOK] • non rispetta il formato [error]

Parametro: imgLink Formato: [[link]]	
Formato[FTIGL]	• rispecchia il formato • rispecchia il formato [link] [FTIGL]

Parametro: Quantità Formato: [0-9]	
Formato[FTQ]	• rispecchia il formato [[0-9] [propertyformatoFPCOK] • non rispetta il formato [error]

9.4.2 Aggiornamento Prodotto

9.4.2.1. Category Partition

Parametro: NomeProdotto Formato: [A-Za-z0-9._%+-]	
Lunghezza[LUNOP]	• 100>[error] • 100<=[property lunghezza LUNOPOK]
Formato[FTNOP]	• Rispetta la lunghezza[if property lunghezza LUIBNOK]

	• Rispetta il formato [A-Za-z0-9._%+-] [FTNOPOK]

Parametro: Descrizione Formato: [A-Za-z0-9._%+-]	
Lunghezza[LUDES]	• 100>[error] • 100<=[property lunghezza LUDESOK]
Formato[FTDES]	• Rispetta la lunghezza[if property lunghezza LUDESOK] • Rispetta il formato [A-Za-z0-9._%+-] [FTDESOK]

Parametro: prezzo Formato: [0-9]	
Formato [FPC]	• rispecchia il formato [0-9] [propertyformatoFPCOK] • non rispetta il formato [error]

Parametro: Quantità Formato: [0-9]	
Formato[FTQ]	• rispecchia il formato [[0-9] [propertyformatoFPCOK] • non rispetta il formato [error]

9.4.3. Ricerca Prodotto

9.4.3.1. Category Partition

Paramentro : Nome Prodotto	
DatabaseRaggiungibile[DBOK]	• Database non raggiungibile [error] • Database raggiungibile[property DatabaseRaggiungibile DBOK]
NomeProdottoPresente[NPPOK]	• Database non raggiungibile [error] • NomeProdtto non presente nel database[error] • NomeProdotto presente nel database[NPPOK]

9.4.4 Rimuovi prodotto

9.4.4.1 Category Partition

9.4.5 Visualizza prodotto

9.4.5.1 Category Partition

Il category partition per questo caso non è necessario in quanto non vi sono dati in input da verificare.

9.5 Gestione Carrello

9.5.1 Aggiungi al carello

9.5.1.1 Category Partition

Il category partition per questo caso non è necessario in quanto non vi sono dati in input da verificare.

9.5.2 Rimuovi dal Carrello

9.5.2.1 Category Partition

Il category partition per questo caso non è necessario in quanto non vi sono dati in input da verificare.

9.5.3 Checkout

9.5.3.1 Category Partition

Il category partition per questo caso non è necessario in quanto non vi sono dati in input da verificare.

10 Pianificazione dei test

Il team per il testing deve essere composto da persone che hanno una completa conoscenza del dominio applicativo e del dominio delle soluzioni del sistema. Inoltre, devono sapere in maniera completa tutte le tecniche di testing utilizzate e nominate all'interno del Test Plan e Test case specification. Le attività che comportano tale fase devono essere fatte nei tempi, nei costi e nei vincoli di qualità specificati. Solitamente gli sviluppatori e i tester non sono relazionati alla stessa persona, l'unico caso in cui tali ruoli combaciano con la stessa persona è quando si verifica un fault che porta ad un massivo cambiamento di una funzionalità. Il team dedicato al controllo dei vincoli di qualità sarà anche responsabile delle attività di testing e, quindi, della ricerca di possibili fault presenti nel sistema. La documentazione dei fault trovati verrà inviata agli sviluppatori per consentire la correzione del sistema. Il sistema revisionato dovrà, successivamente alla correzione, essere verificato attraverso altri casi di test per consentire di assicurarsi che le modifiche sono state

compiute in maniera corretta e verificare se tali cambiamenti hanno introdotto nuovi errori. L'attività di testing è fondamentale nello sviluppo di un sistema software in quanto la mancanza di tale attività o una cattiva gestione di essa può portare al completo fallimento del sistema e, in casi estremi, dell'intero progetto. Data l'importanza del testing, la schedulazione delle sue attività sono fondamentali.

10.1 Determinazione dei ruoli

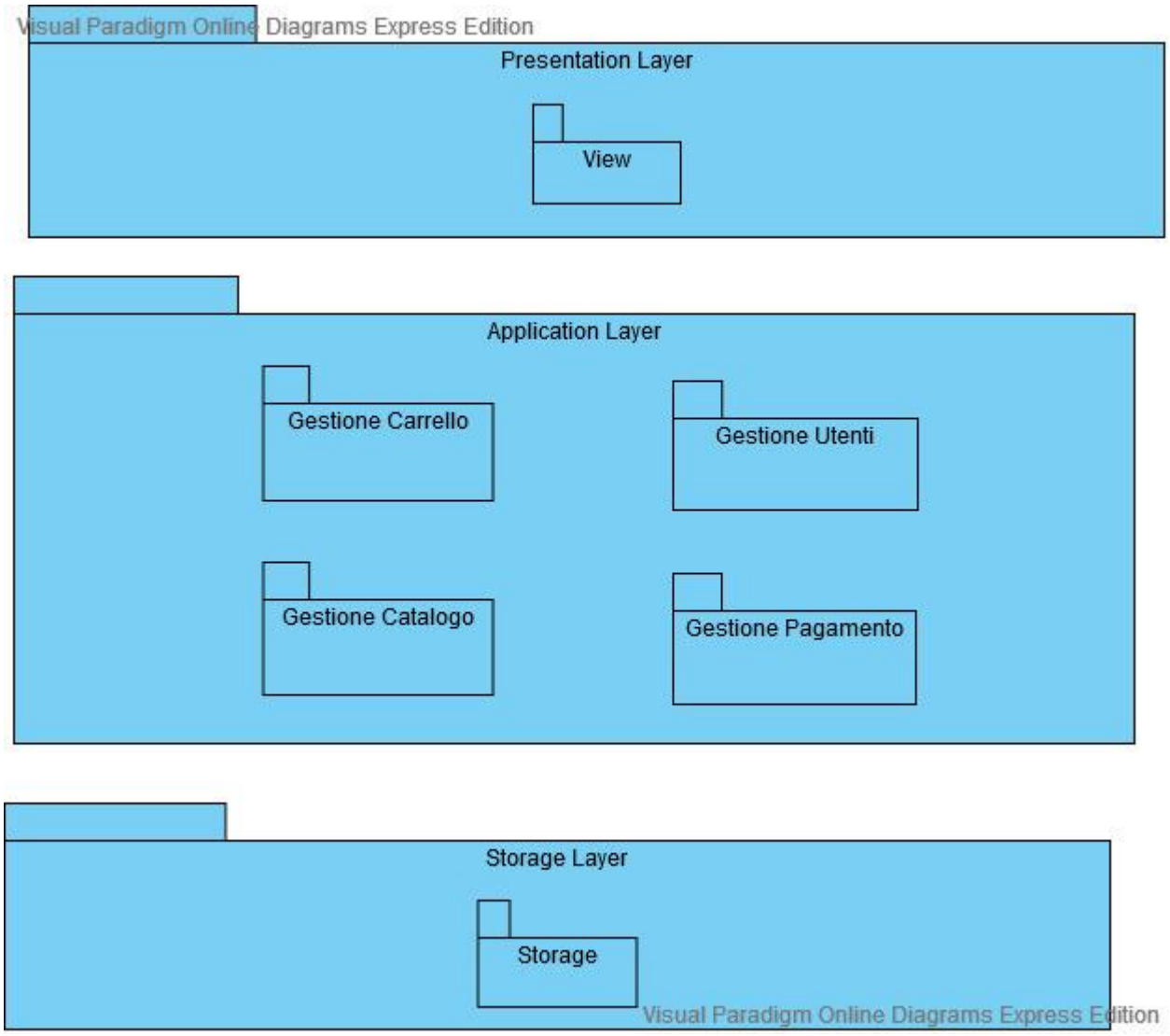
Le attività di testing verranno eseguite da Raffaele Coscione. Le attività relative al testing di unità come correzioni e cambiamenti verranno delegate agli sviluppatori che si occuperanno delle modifiche a livello implementativo. Tale organizzazione serve principalmente per allegerire il carico di lavoro da parte del tester che potrà dedicarsi maggiormente sul lavoro di testing funzionale.

10.2 Determinazione dei rischi

I rischi di un completo fallimento che prevedono la presenza di una quantità di errori elevata, può portare ad un ritardo del progetto. Per risolvere questa situazione, sempre se si verifichi, si è deciso di effettuare una pianificazione verticale dei test funzionale. Tale approccio ci permetterà di rilasciare un numero minore di funzionalità nei tempi previsti ma in maniera completa e funzionale.

10.3 Decomposizione gerarchica del sistema

La divisione gerarchica del sistema è stata mappata in 3 livelli gerarchici come nel seguente diagramma:



10.4 Organizzazione delle attività di testing

Le attività di testing verranno organizzate secondo uno schema che effettuerà una divisone 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 numerosi failure il sistema verrà rilasciato con mno componenti, ma interamente testate e funzionanti.