**Test Plan**

**INTRODUZIONE**

In questo documento si specifica la pianificazione delle attività di Testing riguardante il sito Buy&See al fine di verificare se esistono differenze fra il comportamento atteso e il comportamento reale del sistema. In questa attività si andrà a rilevare gli eventuali errori prodotti dal codice scritto , per evitare che essi si presentano nel momento in cui il sistema viene messo ad esercizio. Le attività di testing sono state pianificate per le seguenti gestioni :

* Gestione autenticazione
* Gestione catalogo
* Gestione account
* Gestione visione contenuti

**DOCUMENTI CORRELATI**

Il test plan è strettamente collegato ai documenti prodotti fino ad ora , poiché prima di passare alla fase di testing abbiamo bisogno di avere una gran parte delle funzioni previste già implementate , queste sono state definite nei precedenti documenti. Di seguito verranno descritte le relazioni che ci sono fra il test plan e gli altri documenti.

**Relazoni con il Requirement Analysis Document (RAD)**

La relazione tra il test plan e il 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.

**Relazioni con il System Design Document (SDD)**

Nel System Design Document il sistema viene suddiviso in sottosistemi e l’architettura è MVC la quale è strutturata in tre componenti :

* Model
* View
* Control

Il test dei vari componenti deve rimanere fedele a queste suddivisioni il più possibile.

**Relazioni con l’Object Design Documenti (ODD)**

il test d’integrazione farà quanto più riferimento possibile alle interfacce delle classi definite nell’ODD.

**Panoramica**

Come è stato descritto nel System Design Documento(SDD) il sistema è costruito da un’architettura MVC , al fine di garantire basso accoppiamento e alta coesione fra le classi. Le tre componenti dell’architettura sono Model , View , Control. Queste tre componenti contengono rispettivamente la parte visuale del sistema , la parte applicativa e il database. Il sistema Bye&See è stato diviso per gestioni. Le gestioni che sono state individuate prevedono principalmente i metodi che effettuano operazioni CRUD(Create , Read , Update , Delete) e saranno queste le funzionalità che saranno testate.

**FUNZIONALITA’ DA TESTARE/NON TESTARE**

Di seguito vengono elencate le funzionalità da testare per ogni gestione

* Gestione autenticazione
* Registrazione
* Login
* Gestione catalogo
* Rimuovi film dal catalogo
* Aggiorna sezione novità
* Gestione account
* Modifica dati personali
* Invia segnalazioni
* Gestione visione contenuti
* Acquisto film
* Visione film

Di seguito vengono elencate le funzionalità da non testare per ogni gestione

* Gestione autenticazione
* Logout
* Recupero password
* Gestione catalogo
* Ricerca film
* Sfoglia catalogo per genere
* Sfoglia catalogo per anno
* Sfoglia catalogo per tutti i film
* Sfoglia catalogo per sezione novità
* Richiesta aggiunta film
* Modifica catalogo
* Gestione account
* Visualizza dati personali
* Modifica password
* Recupero password
* Visualizza segnalazioni
* Visualizza film acquistati
* Gestione visione contenuti
* Visione trailer
* Rilasciare recensione
* Rilasciare valutazione
* Modera recensioni
* Bannare utenti

**CRITERI PASS/FIELD**

I dati in input del test saranno suddivisi in classi di equivalenza , ovvero saranno raggruppati in insiemi con caratteristiche comuni , modo tale da poter testare solo uno degli elementi della classe. L’input avrà superato il test se l’output risultante è quello atteso , il risultato atteso sarà quello specificato dal membro del team che si occuperà del testing sul tale test case.

**APPROCCIO**

La fase di testing sarà suddivisa in tre fasi :

1. Testing di unità: verrà testata nello specifico il funzionamento di ogni singola unità del sistema
2. Testing di integrazione : dove verranno testate le interfacce delle suddette unità
3. Testing del sistema : dove verrà testato l’intero sistema assemblato

**Testing di unità**

Durante questa fase verranno ricercate le condizioni di fallimento , isolando i componenti. La strategia usata per il testing è la tecnica Black-Box , che si concentra sul comportamento Input/Output ignorando la struttura interna della componente. Per minimizzare il numero si test cases i possibili input verranno partizionati in classi di equivalenza e per ogni classe verrà usato un test case. Gli errori scovati in questa fase , e anche in tutte le altre , devono essere comunicati agli sviluppatori in modo tale che questi possano correggere l’errore e ripristinare la fase di testing.

**Testing di integrazione**

Questa face ha lo scopo di interagire tutte le componenti di una funzionalità al fine di testarle nel complesso utilizzando una strategia Bottom-Up.

**Testing di sistema**

In questa fase vogliamo dimostrare che il sistema soddisfi tutti i requisiti richiesti. Si cercherà di testare le funzionalità più utili per l’utente e quelle che hanno una maggiore probabilità di fallimento.

**Sospensione e ripresa**

**Criteri di sospensione**

La fase di testing verrà sospesa una volta raggiunto un giusto compromesso fra qualità del prodotto e costi dell’attività di testing. Il testing verrà portato avanti per quanto più tempo possibile senza rischiare di ritardare ulteriormente la consegna finale del progetto.

**Criteri di ripresa**

In seguito alle modifiche o alle correzioni delle componenti che introdurranno errori o fallimenti , i test case verranno di nuovo sottoposti al sistema in modo da assicurarsi che la modifica o la correzione sia servita effettivamente per risolvere l’errore o il fallimento.

**Materiale per il testing**

L’hardware necessario all’attività d testing è un PC con connessione ad internet dato che il database che viene usato dal sistema è accessibile da remoto in modo tale che tutti possano avere un’unica versione del db su cui lavorare.