Test Plan
Progetto Fund.It

Riferimento	Gruppo G13
Versione	1.0
Scritto da	Luigi Crescenzo, Francesco Esposito, Sabato Genovese, Angelo Meo

Revision History

Data	Versione	Descrizione	Autori
04/01/2022	0.1	Prima stesura	Sabato Genovese
06/01/2022	0.2	Introduzione, relazione con altri documenti	Luigi Crescenzo, Francesco Esposito
06/01/2022	0.3	Panoramica del sistema	Angelo Meo
07/01/2022	0.4	Features da testare/da non testare	Francesco Esposito
07/01/2022	0.5	Revisione del lavoro svolto	Gruppo
07/01/2022	0.6	Approccio, Sospensione e ripristino	Sabato Genovese
08/01/2022	0.7	Materiale di testing, Test schedule	Francesco Esposito
08/01/2022	0.8	Test Cases	Gruppo
09/01/2022	0.9	Completati Test Cases	Francesco Esposito, Sabato Genovese
09/01/2022	1.0	Revisione del lavoro svolto	Gruppo

Sommario

Introduzione

Fund.It si propone di aiutare il finanziamento di progetti attraverso il meccanismo del crowdfunding, al fine di finanziare progetti creativi e sociali.

Il documento di Test Plan ha l'obiettivo di descrivere ed analizzare le attività di Testing per la piattaforma *Fund.It*. Il fine è quello di garantire che ogni aspetto della piattaforma funzioni in modo corretto.

All'interno del documento sono riportate le strategie di testing adottate, le funzionalità che saranno testate e gli strumenti scelti per la rilevazione degli errori, con lo scopo di presentare al cliente finale un sistema privo di malfunzionamenti.

Sono state pianificate attività di testing per le seguenti funzionalità:

- Autenticazione
- Gestione Utente
- Gestione Campagne
- Gestione Donazioni
- Gestione Segnalazioni
- Gestione Categorie

Relazione con altri documenti

Nel processo di individuazione dei test case adatti al sistema software in questione si farà uso della documentazione finora redatta.

In particolare, si farà uso del Requirements Analysis Document (RAD) dove sono presenti i requisiti funzionali e non funzionali che il sistema Fund.It dovrà implementare. La maggior parte dei test case pianificati saranno estrapolati da questo documento.

Si farà uso del System Design Document (SDD) nel suddividere i test case nei sottosistemi individuati in questo documento.

Panoramica del sistema

Il sistema da noi proposto si basa su architettura Three-Tier come già specificato nel documento SDD.

Si farà uso di HTML5, CSS, JavaScript, la libreria JavaScript jQuery e il framework Bootstrap per il front-end.

Per il back-end si farà uso di Java come linguaggio Object Oriented.

Per la gestione dei dati persistenti si farà uso di Azure di Microsoft e di SQL.

Features da testare/da non testare

Di seguito vengono mostrati tutti i sottosistemi su cui effettuare il testing Black Box

- Autenticazione
  - Login
- Gestione Utente
  - Modifica dati profilo
- Gestione Campagne
  - Creazione Campagna
  - Modifica Campagna
- Gestione Donazioni
  - Scrittura di un commento
- Gestione Segnalazioni
  - Segnalazione Campagna
- Gestione Categorie
  - Inserimento Categoria
  - Modifica Categoria

Non verranno testati con black box tutte quelle funzionalità a bassa priorità e quelle che non richiedono un input da parte dell'utente.

Pass/Fail criteria

Le attività di testing hanno come scopo il trovare le differenze tra il comportamento atteso specificato attraverso il modello del sistema e il comportamento osservato dal sistema implementato.

L'obiettivo è quello di trovare dati di test che massimizzano le probabilità di trovare errori durante l'esecuzione.

L'esito di un test è comparato ad un oracolo, se il risultato dato in output è uguale al risultato atteso il test ha successo, altrimenti viene segnalato come test fallito.

Il testing è considerato valido se vengono rispettati i seguenti vincoli:

- Raggiungere un branch coverage non inferiore al 75%;
- Testare tutti i requisiti funzionali;
- Effettuare test di regressione ogni volta che si introducono nuove caratteristiche al sistema o vengono modificate quelle presenti.
- Rispettare la soglia di errori rilevati dal tool Checkstyle (Soglia 20)

Approccio

Il testing del sistema è strutturato in tre fasi:

1. testing di sistema;
2. testing di integrazione;
3. testing di unità.

Prima dell'implementazione del sistema si procederà a scrivere i test case di sistema, mentre si procederà a progettare i test case di unità.
Inoltre, durante la stesura del codice saranno eseguite delle periodiche attività di revisione e controllo qualità sul codice prodotto.

Testing di sistema

Verifica se sono soddisfatti i requisiti funzionali e non funzionali, in questa fase verrà effettuato un controllo della conformità dell'intero sistema con i requisiti dell'utente finale. Il tool che verrà usato sarà Selenium IDE, che permette di registrare le azioni che l'utente deve effettuare per una determinata funzionalità, al fine di ripetere lo stesso test con input diversi e valutare le diverse risposte del sistema.

Test di integrazione

I test di integrazione hanno come fine l'individuazione di fault integrando diversi componenti insieme. Il processo di integrazione richiede che le componenti da integrare siano testate prima separatamente e poi si effettuano i test dopo aver integrato i due sottosistemi. Per la definizione dei test case si farà uso del framework JUnit, mentre per gli oggetti mock si farà uso di Mockito. Sarà inoltre usato Maven per automatizzare l'esecuzione dei test, mentre per generare un report sulla branch coverage verrà usato JaCoCo.

Per i test di integrazione si procederà con un approccio bottom-up, si prevede di testare dapprima le classi Service per poi passare a testare quelle Controller. I test di integrazione ovviamente comprenderanno quelli unitari.

Testing di unità

In questa fase si prevede di testare le classi e i metodi del sistema, ricercando le condizioni di fallimento per rintracciare il fault associato. Sono escluse dalle attività di testing di unità le classi entity in quanto classi POJO. Il testing di ogni singola componente verrà realizzata mediante la metodologia Black-Box, la documentazione una parte è inclusa all'interno del codice, mentre l'altra parte sarà generata dal framework JUnit. Con l'aiuto del Category Partition verranno realizzate delle classi di equivalenza che permettono di ottenere una buona copertura rispetto agli input.

Sospensione e ripristino

In questa sezione verranno esplicitati i criteri di sospensione del test e le attività da intraprendere al ripristino.

Criteri di sospensione

Le attività di testing sono sospese in caso di rilevazione di un errore o un failure come crash del database, server e connessione.

Criteri di ripristino

Le attività riprenderanno in seguito a modifiche attuate per risolvere i fault. In questo caso sarà eseguita una sessione di testing di regressione in modo da assicurarsi che le modifiche apportate non abbiano introdotto ulteriori difetti. Verranno effettuati nuovamente tutti i test, in particolare quelli che verificano l'effettiva risoluzione del problema.

Materiale di testing

L'hardware necessario per l'attività di test è un semplice computer dotato di browser e un web server, necessariamente connesso ad internet, in quanto il sistema utilizza i servizi della piattaforma Microsoft Azure per la persistenza dei dati.

Test cases

L'approccio utilizzato è quello esplicitato nella sezione del testing di unità. L'output atteso sarà dato da un oracolo umano basato sul giudizio a causa dell'assenza di specifiche formali.

1 Autenticazione

1.1 Login

Parametro: Email	
FORMATO **^[A-z0-9._%+-]+@[A-z0-9.-]+\.[A-z]{5,70}$**	
Nome Categoria	**Scelte per la categoria**
Formato Email[FE]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FE_OK]

Parametro: Password	
FORMATO **^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&._])[A-Za-z\d@$!%*?&._]{8,32}$**	
Nome Categoria	**Scelte per la categoria**
Formato Password[FP]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FP_OK]

Test case id	Test Frame	Esito
TC_1.1.1	**FE1**	Errore, formato email non corretto
TC_1.1.2	**FE2, FP1**	Errore, formato password non corretto
TC_1.1.3	**FE2, FP2**	Corretto

2 Gestione Utente

2.1 Modifica Profilo

Parametro: Nome	
FORMATO **^[A-Za-zà-ź \s]{2,50}$**	
Nome Categoria	**Scelte per la categoria**
Formato Nome[FN]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FN_OK]

Parametro: Cognome	
FORMATO **^[A-Za-zà-ź \s]{2,50}$**	
Nome Categoria	**Scelte per la categoria**
Formato Cognome[FC]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FC_OK]

Parametro: Password	
FORMATO **^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&._])[A-Za-z\d@$!%*?&._]{8,32}$**	
Nome Categoria	**Scelta per la categoria**
Formato Password[FP]	1. Formato non corretto = false [error] 2. Formato corretto = true [PROPERTY FP_OK]

Parametro: Email	
FORMATO **^[a-zA-Z0-9.!#$%&'*+\=?^_`{\|}~-]{1,29}+@[a-zA-Z0-9-]{1,29}+(?:\.[a-zA-Z0-9-]+){1,10}$**	
Nome Categoria	**Scelta per la categoria**
Formato Email[FE]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FE_OK]

Parametro: Telefono	
FORMATO **^\d{10}$**	
Nome Categoria	**Scelta per la categoria**
Formato Telefono [FT]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FT_OK]

Parametro: Strada	
FORMATO **[a-zA-Z0-9\s]{1,74}+, [0-9]{1,5}$**	
Nome Categoria	**Scelta per la categoria**
Formato Strada [FS]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FS_OK]

Parametro: Città	
FORMATO **^[A-Za-zà-ź \s]{2,50}$**	
Nome Categoria	Scelta Categoria
Formato Città[FCit]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FCit_OK]

Parametro: CAP	
FORMATO **^[0-9]{5}$**	
Formato CAP [FCAP]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FCAP_OK]

Parametro: Codice Fiscale	
FORMATO **^[A-Z]{6}[0-9]{2}[A-Z][0-9]{2}[A-Z][0-9]{3}[A-Z]$**	
Formato CF [FCF]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FCF_OK]

Parametro: Data di Nascita	
FORMATO **^(?:(?:31(\/|-|\.)(?:0?[13578]|1[02]))\1|(?:(?:29|30)(\/|-|\.)(?:0?[13-9]|1[0-2])\2))(?:(?:1[6-9]|[2-9]\d)?\d{2})$\|^(?:29(\/\|-\|\.)0?2\3(?:(?:(?:1[6-9]\|[2-9]\d)?(?:0[48]\|[2468][048]\|[13579][26])\|(?:(?:16\|[2468][048]\|[3579][26])00))))$|^(?:0?[1-9]|1\d|2[0-8])(\/|-|\.)(?:(?:0?[1-9])|(?:1[0-2]))\4(?:(?:1[6-9]|[2-9]\d)?\d{2})$**	
Formato Data [FDate]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FDate_OK]

Test case id	Test Frame	Esito
TC_2.1.1	**FN1**	Errore, formato nome non corretto
TC_2.1.2	**FN2, FC1**	Errore, formato cognome non corretto
TC_2.1.3	**FN2, FC2, FP1**	Errore, formato password non corretto
TC_2.1.4	**FN2, FC2, FP2, FE1**	Errore, formato email non corretto
TC_2.1.5	**FN2, FC2, FP2, FE2, FT1**	Errore, formato telefono non corretto
TC_2.1.6	**FN2, FC2, FP2, FE2, FT2, FS1**	Errore, formato strada non corretto
TC_2.1.7	**FN2, FC2, FP2, FE2, FT2, FS2, FCit1**	Errore, formato città non corretto
TC_2.1.8	**FN2, FC2, FP2, FE2, FT2, FS2, FCit2, FCAP1**	Errore, formato CAP non corretto
TC_2.1.9	**FN2, FC2, FP2, FE2, FT2, FS2, FCit2, FCAP2, FCF1**	Errore, formato CF non corretto
TC_2.1.10	**FN2, FC2, FP2, FE2, FT2, FS2, FCit2, FCAP2, FCF2, FDate1**	Errore, formato data non corretto
TC_2.1.11	**FN2, FC2, FP2, FE2, FT2, FS2, FCit2, FCAP2, FCF2, FDate2**	Corretto

3 Gestione Campagne

3.1 Creazione/Modifica Campagne

Parametro: Titolo	
Nome Categoria	**Scelte per la categoria**
Lunghezza [LNC]	1. Lunghezza > 50 OR Lunghezza < 5 = false[errore] 2. 5 <= Lunghezza <= 50 = true [PROPERTY LNC_OK]

Parametro: Descrizione	
Nome Categoria	Scelte per la categoria
Lunghezza [LD]	1. Lunghezza > 3000 OR Lunghezza < 15 = false[errore] 2. 15 <= Lunghezza <= 3000 = true[PROPERTY LD_OK]

Parametro: SommaTarget	
Nome Categoria	**Scelte per la categoria**
Valore [VST]	1. Valore < 1 = false[errore] 2. Valore >= 1 = true[PROPERTY VST_OK]

Parametro: Immagine	
FORMATO **^(.+)\/([^\/]+)$**	
Nome Categoria	**Scelte per la categoria**
Path Immagine [PI]	1. Formato non corretto = false[errore] 2. Formato corretto = true[PROPERTY PI_OK]

Test case id	Test Frame	Esito
TC_3.1.1	**LNC1**	Errore, lunghezza titolo non corretta
TC_3.1.2	**LNC2, LD1**	Errore, lunghezza descrizione non corretta
TC_3.1.3	**LNC2, LD2, VST1**	Errore, valore somma target non corretto
TC_3.1.4	**LNC2, LD2, VST2,PI1**	Errore, path immagine non corretto
TC_3.1.5	**LNC2, LD2, VST2,PI2**	Corretto

4 Gestione Donazioni

4.1 Commento a seguito di una donazione ad una campagna

Parametro: Commento	
FORMATO **^[A-zÀ-ù 0-9]{2,150}$**	
Nome Categoria	**Scelte per la categoria**
Formato Commento [FCom]	1. Formato non corretto = false[error] 2. Formato corretto = true [PROPERTY FCom_OK]

Test case id	Test Frame	Esito
TC_4.1.1	**FCom1**	Errore, formato commento non corretto
TC_4.1.2	**FCom2**	Corretto

5 Gestione Categorie

5.1 Aggiungi/Modifica Categoria

Parametro: Nome Categoria	
Formato **^[A-Za-zà-ź \s]{2,100}$**	
Nome Categoria	**Scelte per la categoria**
Formato Categoria[FCat]	1. Formato non corretto = false[error] 2. Formato corretto = true[PROPERTY FCAT_OK]

Test case id	Test Frame	Esito
TC_5.1.1	**FCat1**	Errore, formato categoria non corretto
TC_5.1.2	**FCat2**	Corretto

6 Gestione Segnalazioni

6.1 Segnala Campagna

Parametro: Descrizione	
Formato **^[A-zÀ-ù 0-9]{2,300}$**	
Nome Categoria	**Scelte per la categoria**
Formato Segnalazione[FSe]	1. Formato non corretto = false[error] 2. Formato corretto = true[PROPERTY FSe_OK]

Test case id	Test Frame	Esito
TC_6.1.1	**FSe1**	Errore, formato descrizione non corretto
TC_6.1.2	**Fse2**	Corretto

Testing schedule

La scrittura dei casi di test avverrà prima dello sviluppo del codice.

L'esecuzione dei test avverrà sia durante che dopo l'implementazione del sistema. Una volta concluso lo sviluppo, tutti i test saranno ri-eseguiti per garantire il corretto funzionamento e produrre i report finali.