TP Test Plan

English Validation Porting Android

Riferimento	
Versione	1.3
Data	
Destinatario	
Presentato da	
Approvato da	

Revision History

Data	Versione	Descrizione	Autori
01/12/2019	1.0	Aggiunta capitoli 1,2,5,6,7,8,9,10,11,12,13,14.	Carmine Brancaccio, Antonio Calabrese, Fiorenzo Carino, Michele Costabile, Marco Fresolone, Paolo Schiavo, Stefano Tulino, Simone Veneruso.
01/12/2019	1.1	Aggiunta capitoli 3,4	Stefano Tulino, Marco Fresolone.
02/12/2019	1.2	Correzione indentatura	Marco Fresolone.
02/12/2019	1.3	Correzioni generali	Marco Fresolone.

Contents

1. Introduzione

In questo documento verranno definiti gli approcci e le attività di testing riguardanti la piattaforma Porting mobile English Validation. Saranno identificati gli elementi da testare e da non testare, gli approcci e gli strumenti usati per l'attività di testing. Lo scopo del testing è quello

di rilevare fault e failure presenti nel sistema, ecco perchè diremo che il testing avrà avuto successo se riuscirà ad individuare dei fault o una failure. Quest'attività ci consentirà di scoprire degli errori prima del rilascio del sistema, in questo modo potremmo porre rimedio a questi errori e fare in modo che non si rivelino durante la messa in esercizio del sistema. Le gestioni, che saranno analizzate, contengono requisiti con priorità media o alta e sono le seguenti: Gestione Utente, Gestione Admin, Gestione Segreteria, Gestione Studente.

1.1 Definizioni, Acronimi e Abbreviazioni

1.1.1 Definizioni

Branch Coverage : metodo che assicura che tutti i rami del programma o gli stati condizionali siano almeno una volta, durante il processo di test;
Failure: mancata prestazione di un servizio atteso;
Fault: causa di un failure, insieme di informazion, errori algoritmici, che quando vengono processate generano un fallimento;
Model View Control: modello architetturale comunemente usato per lo sviluppo di interfacce utente che dividono un'applicazione in tre parti interconnesse. Questo viene fatto per separare le rappresentazioni interne di informazioni dai modi in cui le informazioni sono presentate e accettate dall'utente;

1.1.2 Acronimi E Abbreviazioni

TP_1.0: Utilizzata per indicare il TestPlan;
RAD: Abbreviazione utilizzata per indicare il Requirement Analysis Document;
SDD: Abbreviazione utilizzata per indicare il System Design Document;
ODD: Abbreviazione utilizzata per indicare il Object Design Document;
SOW: Abbreviazione utilizzata per indicare lo Statement Of Work;
BC: Abbreviazione utilizzata per indicare il Business Case;
TP: Abbreviazione utilizzata per indicare il Test Plan;
STC: Abbreviazione utilizzata per indicare il System Test Case;
MVC: Abbreviazione utilizzata per indicare l'architettura Model View Controller;
FB: Abbreviazione utilizzata per indicare FireBase;
API: Abbreviazione utilizzata per indicare le Application Programming Interface;

1.2 Riferimenti

PMBOK ® Guide and Software Extention to the PMBOK® Guide, Fifth Ed., Project Management Institute, 2013
Kathy Schwalbe, "Information Technology Project Management", International Edition7E, Cengage Learning, 2014
Bernd Bruegge, Allen H. Dutoit, "Object-Oriented Software Engineering Using UML, Patterns and Java", Third Ed., Pearson, 2010;
Documentazione di Progetto:
RAD_7.2;
SDD_ 1.5;

2. Documenti Correlati

Il presente documento è in stretta relazione con i documenti prodotti fino al rilascio della versione 1.0 del Test Plan, inoltre, esso è in forte relazione anche con documenti che saranno sviluppati e rilasciati in futuro, ciò comporterà modifiche ed aggiornamenti di quest'ultimo.
I test case sono basati sulle funzionalità individuate nel documento di raccolta ed analisi dei requisiti.

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

La relazione fra TP e RAD riguarda i requisiti funzionali e non funzionali del sistema, in quanto esso contiene la descrizione dettagliata delle funzionalità con scenari, use case, diagrammi e mockup e inoltre vi è indicata anche la priorità dei requisiti.

2.2 Relazione con il System Design Document (SDD)

Nell' SDD è presente la architettura del sistema (MVC), la struttura dei dati e i servizi dei sottosistemi individuati.

2.3 Relazione con l'Object Design Document (ODD)

Nell'ODD (ancora non sviluppato al momento del rilascio dell' TP) sono contenuti i package e le classi del sistema.

2.4 Relazione con lo Statement Of Work (SOW)

Nello Statement Of Work uno dei criteri di premialità, è quello di ottenere una branch coverage del 75%, i test case verranno strutturati in modo tale da poter soddisfare tale criterio.

3. Panoramica del Sistema

Come definito nell' SDD, la struttura del nostro sistema segue l'architettura MVC, il quale illustra i servizi utilizzando un approccio REST.
La componente fondamentale di questa architettura è il controller. Per ogni sottosistema ci sarà un controller che si occuperà della logica di business della specifica gestione.
Nel model verranno mappate gli oggetti permanenti memorizzati in Firebase, il quale sarà il nostro database non-relazionale di riferimento.
La view invece si occuperà semplicemente di mostrare le interfacce utente.
Nel sistema verrà usato anche un approccio REST, il quale, attraverso una comunicazione Http, interrogherà quando richiesto Firebase, per avere degli specifici risultati. Altro utilizzo del Rest, potrebbe essere la generazione di documenti, come pdf o Excel, in quanto non possono essere generati automaticamente dalla nostra app Android.
I sottosistemi nel nostro sistema saranno:
* Gestione Studente
* Gestione Admin
* Gestione Segreteria
* Gestione Utente

4. Possibili Rischi

ID	Rischio	Risvolto	Probabilità	Impatto
R1	Pianificazione delle attività di testing non adeguata	Negativo	Bassa	Alto
R2	Problemi finanziari organizzazione costringono a riduzioni bilancio progetto.	Negativo	Media	Alto

R3	Impossibile reclutare persone con competenze richieste per progetto.	Negativo	Bassa	Alto
R4	Il team non segue l'approccio definito	Negativo	Bassa	Alto
R3	Il team trova difficoltà nello specificare i casi di test	Negativo	Media	Alto
R4	Il team trova difficoltà nello usare i tool definiti per svolgere la attività di testing	Variabile	Media	Medio
R5	Base di dati usata in sistema non può gestire un numero di transazioni troppo elevato. Investigazione su acquisto di componenti e/o rivalutazione del codice usato	Variabile	Media	Alto
R6	Le attività di testing proseguono oltre i tempi preventivati	Variabile	Media	Medio/Alto
R7	I casi di test precedentemente illustrati non riescono a portare una branch coverage del 75%,requisito minimo per un buon software	Negativo	Media	Medio

5. Funzionalità da testare

Il team di sviluppo provvederà a testare, per le varie gestioni, i seguenti casi di priorità media o alta:

- Gestione studente
  - Registrazione
  - Compilazione primo form
  - Caricamento allegati
- Gestione utente
  - Accesso utente
- Gestione admin
  - Correzione errori admin
  - Ricercare certificati per ente o per nome
- Gestione segreteria
  - Aggiunta CFU
  - Correzione errori segreteria

6. Funzionalità da non testare

Le funzioni di priorità bassa e quelle associate all'invio di notifiche (gestite dal sistema operativo Android e da Firebase) non verranno testate.

7. Approccio

7.1 Testing di integrazione

In questa fase le unità vengono prese, combinate e testate in gruppo.
Per poter effettuare l'integration test è stata scelta la strategia bottom-up, in quanto consente di poter iniziare l'attività di testing non appena il primo modulo è stato specificato. Questo approccio, richiede la costruzione di driver per simulare l'ambiente chiamante.

7.2 Testing di sistema

Prima della messa in esercizio del sistema verrà effettuata una fase di testing di sistema per dimostrare che i requisiti commissionati dal cliente siano soddisfatti. In questo tipo di testing andremo a testare le funzionalità usate più frequentemente dal cliente e quelle che presentano maggiore possibilità di fallimento.

8. Criteri di pass/fail

Una volta individuati i vari dati di input del test, questi verranno raggruppati in base a caratteristiche comuni in insiemi. In questo modo ci sarà possibile diminuire ed ottimizzare il numero di test. La fase di test avrà successo se individuerà una failure, ovvero se l'output osservato sarà diverso da quello atteso. Ogni qual volta verrà individuata una failure, questa verrà analizzata e, se legata ad un fault, si procederà alla sua correzione. Una volta completata la correzione, si procederà, ad una nuova fase di test per verificare che la modifica non ha impattato su altri componenti del sistema, tutto in modo iterativo. Invece il testing fallirà se l'output osservato sarà uguale all'oracolo.

9. Criteri di Sospensione e Ripresa

9.1 Criteri di sospensione

La fase di testing verrà interrotta quando verranno raggiunti i risultati attesi in accordo con in tempi e i costi allocati alle attività di testing.

9.2 Criteri di ripresa

Le attività di testing riprenderanno in seguito a delle modifiche che potrebbero introdurre nuovi errori all'interno del sistema.

10. Test Deliverables

I documenti rilasciati durante e al termine della fase di test sono:
Test Plan;
System Test Case;
Test Case Specification;
Test Execution Report;

Test Incident Report;
Test Summary Report;

11. Materiale per il testing

Per svolgere le attività di testing è necessario un dispositivo mobile Android , Firebase con una collezione di dati adeguata e Node.js per fare il parsing dei dati in file Excel o Pdf.

12. Necessità di Training per lo Staff

Il team non ha mai utilizzato il DB non relazionale Firebase ed ha poca esperienza nella programmazione mobile Android. Per far fronte a questa esigenza i PM svolgeranno delle attività di tutorato mirate a colmare la mancanza di skill e successivamente forniranno esempi utili per rendere più chiari i concetti spiegati durante il tutorato.

13. Responsabilità

Ogni team member sarà resposabilie del testing della gestione alla quale è stato assegnato. Il team non verrà diviso in "tester" e "developer" per evitare overhead di comunicazione.

14. Glossario

- **Mockup :** modello dimostrativo di un oggetto originale che viene realizzato per dare un'idea dell'oggetto finito
- **Deliverables :** un oggetto materiale o immateriale realizzato (fornito/consegnato) come risultato di un'attività
- **PM :** Project Manager, una persona che ha la responsabilità generale per l'avvio, la pianificazione, la progettazione, l'esecuzione, il monitoraggio, il controllo e la chiusura di un progetto.
- **Tester :** Membro del team che si occupa di eseguire il testing sul codice sviluppato.
- **Developer :** Membro del team che si occupa dello sviluppo del codice e della sua implementazione
- **Overhead :** risorse accessorie, richieste in sovrappiù rispetto a quelle strettamente necessarie per ottenere un determinato scopo.