KEY-KNOWLEDGE

Nome Documento

Test Plan 2.0

TABELLA REVISIONI

DATA	VERSIONE	DESCRIZIONE	AUTORE
8/03/2021	1.0	Prima versione del Test Plan	Crescenzo Manzone Franco Nicola Fernando Giovanni Battista Mercurio
12/05/2021	2.0	Revisione completa	Crescenzo Mazzone

1. Introduzione

Il sistema è stato pensato come un'app mobile multiplayer dove gli utenti si sfidano in quiz. La particolarità del sistema è che le domande al quiz sono scelte da un AI per ogni utente. L'intelligenza artificiale ha l'obiettivo di fornire all'utente domande sempre più difficili in base a quelle che sono le conoscenze dell'utente.

Quindi, l'obiettivo del nostro sistema è quello di spingere gli utenti a migliorare le proprie conoscenze passo dopo passo. Potrebbe essere utile a persone che vogliono prepararsi per concorsi o test.

Lo scopo principale della fase di testing è quello di controllare che i requisiti funzionali definiti in fase di analisi siano effettivamente funzionanti. L'obiettivo di questa fase è quindi quello di trovare quanti più fault possibili, in maniera da poter migliorare il sistema prima di rilasciarlo all'utente finale.

2. Relazioni con altri documenti

- RAD
- SDD
- Problem Statement

3. Panoramica del sistema

Come definito nel System Design Document, il sistema è stato suddiviso in tre livelli: Model, View e Controller.

I componenti che verranno testati sono i seguenti:

- Gestione Account

- Gestione Partite

Queste sono le componenti necessarie a fornire una demo del sistema.

4. Funzionalità da testare

In seguito sono elencate le funzionalità da testare , suddivise per sottosistema:

Gestione Account

- Login-DEMO

Gestione Partite

- Crea Partita-DEMO
- Rispondi Domanda-DEMO

5. Funzionalità da non testare

UserManagement

Logout: non vi è nessun rischio di fallimento dell'operazione o omissione degli input forniti dall'utente, dato che si tratta semplicemente di confermare l'operazione.

Playing

Fine Partita: non c'è alcun rischio di errore o omissione di input da parte dell'utente.

6. Criteri di successo e fallimento

Il testing ha successo se l'output osservato è diverso da quello atteso, ossia si parla di successo quando il testing rileva una failure. In questo caso, la failure verrà analizzata e corretta nel caso sia causata da un bug. Viceversa, il testing fallisce se non viene rilevata nessuna failure.

7. Approccio

L'approccio scelto per il testing prevede la suddivisione in tre fasi: test di unità, test di integrazione e test di sistema. In questo modo si avrà la possibilità di testare ogni sottosistema e di trovare e correggere eventuali bug rilevati.

Test di unità

Questa fase prevede il testing delle singole funzionalità implementate dai vari sottosistemi nel layer Model. Attraverso i framework JUnit e Mockito verranno testati i DAO, ossia le classi che si occupano di gestire gli oggetti del sistema, i JavaBean.

Test di integrazione

Per questa fase è stato scelto di procedere col testing in maniera Bottom-Up, che prevede prima il test dei sottosistemi indipendenti e successivamente verranno testati i sottosistemi che utilizzano i servizi di quelli testati precedentemente.
I sottosistemi dipendenti sono costituiti da quelli situati nel layer Controller.
In questa fase di test di integrazione verranno testate le funzionalità offerte dai sottosistemi presenti nel layer Controller, implementate dalle diverse classi Java, attraverso i framework JUnit e Mockito.

Test di sistema

Questa fase prevede il testing dell'intero sistema, attraverso Espresso test Recorder , offerto dall'ambiente di sviluppo android studio, sfruttando la tecnica di Black-Box, dividendo gli input di test in classi.

8. Materiale per il testing

Come supporto alla fase di testing di Sistema si utilizzerà Espresso test Recorder , offerto dall'ambiente di sviluppo android studio.
Per la fase di testing di integrazione abbiamo utilizzato il framework JUnit e Mockito. Per la fase di testing di unità abbiamo utilizzato il framework JUnit.

9. Test Cases

In seguito sono elencati i casi di test, che verranno descritti in dettaglio nel documento di Test Case Specification:

Login-DEMO

Parametro: Username		
C01:Username non presente	C02:Username presente	
Parametro: Password		
C03:Password errata	C04:Password corretta	
Codice	Combinazioni	Esiti
1.0	C01	Errore
1.1	C02&C03	Errore
1.2	C02&C04	Successo

Crea Partita-DEMO

Parametro: Utente		
C01:utente non trovato	C02:utente trovato	
Codice	Combinazioni	Esiti
2.0	C01	Errore
2.1	C02	Successo

Rispondi Domanda-DEMO

Parametro:Risposta		
C01:risposta non selezionata	C02:risposta selezionata	
Codice	Combinazioni	Esiti
3.0	C01	Errore
3.1	C02	Successo