**KEY-KNOWLEDGE** 

# Nome Documento

Test Plan 1.0

[**KEY-KNOWLEDGE**  1](#_Toc66286943)

[Nome Documento 1](#_Toc66286944)

[TABELLA REVISIONI 2](#_Toc66286945)

[1. Introduzione 2](#_Toc66286946)

[2. Relazioni con altri documenti 2](#_Toc66286947)

[3. Panoramica del sistema 3](#_Toc66286948)

[4. Funzionalità da testare 3](#_Toc66286949)

[UserManagement 3](#_Toc66286950)

[Playing 3](#_Toc66286951)

[5. Funzionalità da non testare 3](#_Toc66286952)

[UserManagement 3](#_Toc66286953)

[Playing 3](#_Toc66286954)

[6. Criteri di successo e fallimeto 3](#_Toc66286955)

[7. Approccio 4](#_Toc66286956)

[Test di unità 4](#_Toc66286957)

[Test di integrazione 4](#_Toc66286958)

[Test di sistema 4](#_Toc66286959)

[8. Materiale per il testing 4](#_Toc66286960)

[9. Test Cases 4](#_Toc66286961)

[Login 5](#_Toc66286963)

[Crea Partita 5](#_Toc66286964)

[Rispondi Domanda 5](#_Toc66286965)

# ****TABELLA REVISIONI****

|  |  |  |  |
| --- | --- | --- | --- |
| **DATA** | **VERSIONE** | **DESCRIZIONE** | **AUTORE** |
| **8/03/2021** | **1.0** | **Prima versione del Test Plan** | **Crescenzo Manzone**  **Franco Nicola Fernando**  **Giovanni Battista Mercurio** |

# 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.

Loscopo 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.

# Relazioni con altri documenti

* RAD
* SDD
* Problem Statement

# 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:

* UserManagement
* Playing

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

# Funzionalità da testare

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

## UserManagement

* Login

## Playing

* Crea Partita
* Rispondi Domanda

Del sottosistema Playing saranno testate solo le funzionalità corrispondenti al requisito funzionale Playing-DEMO, indicato nel Problem Statement.

# 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.

Scegli Domanda: non fa parte del requisito funzionale Playing-DEMO

Crea Domanda: non fa parte del requisito funzionale Playing-DEMO

# Criteri di successo e fallimeto

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.

# 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 dal sottosistema Storage nel layer Model. Attraverso il framework JUnit 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.

Nel nostro sistema, il sottosistema Storage rappresenta l’unico sottosistema indipendente, mentre i sottosistemi dipendenti sono costituiti da quelli situati nel layer Controller.

Visto che il sottosistema Storage è stato già testato individualmente dalla fase di test di unità, in questa fase di test di integrazione verranno testate le funzionalità offerte dai sottosistemi presenti nel layer Controller, implementate dalle diverse Servlet, 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.

# Materiale per il testing

Come supporto alla fase di testing di Sistema si utilizzerà un browser Internet, la scelta è ricaduta su Google Chrome. Come strumenti di supporto verrà utilizzato: il framework Selenium aggiungendo a Google Chrome Selenium IDE.

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.

# Test Cases

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

## Login

|  |  |  |  |
| --- | --- | --- | --- |
| Parametro: Username | | | |
| C01:Username errato | | C02:Username corretto | |
| 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

|  |  |  |  |
| --- | --- | --- | --- |
| Parametro: Modalità | | | |
| C01:modalità non selezionata | | C02:modalità selezionata | |
| Parametro: Utente | | | |
| C03:utente non trovato | | C04:utente trovato | |
| Codice | Combinazioni | | Esiti |
| 2.0 | C01 | | Errore |
| 2.1 | C02&C03 | | Errore |
| 2.2 | C02&C04 | | Successo |

## Rispondi Domanda

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