TP

Test Plan

LUPUS IN CAMPUS

Riferimento	NC12_RAD
Versione	0.12
Data	17/11/2024
Destinatario	Prof Carmine Gravino
Presentato da	NC12 Team
Approvato da	

Revision History

Data	Versione	Descrizione	Autori
27/12/2024	0.1	Prima stesura	Tutto il team
28/12/2024	0.2	Creazione del test case: Entrare in lobby con lista invito	S.G
28/12/2024	0.3	Creazione del test case: Entrare in lobby con codice	F.G
28/12/2024	0.4	Creazione del test case: Aggiunta amico tramite ricerca manuale	C.I
12/02/2025	1.4	Riscritti i test case	S.G
17/02/2025	1.4	Revisione	F.G

Team members

Nome	Acronimo	Informazioni di contatto
Angelo Ascione	AA	a.ascione19@studenti.unisa.it
Federica Graziuso	FG	f.graziuso1@studenti.unisa.it
Stefano Gagliarde	SG	s.gagliarde@studenti.unisa.it
Christian Izzo	CI	c.izzo43@studenti.unisa.it

Sommario

1 Introduzione

Il documento di Test Plan ha l'obiettivo di definire le strategie che verranno utilizzate per effettuare il testing del sistema

Sono state pianificate attività di testing per i seguenti use case:

- UC_RG: Registrare un Giocatore
- UC_ELOPM: Entrare in una Lobby Online Pubblica tramite Mostra lobby
- UC_CL: Creare una Lobby
- UC_IRRM: Inviare una Richiesta di amicizia tramite Ricerca Manuale

2 Relazione con altri documenti

Per la corretta individuazione dei test case, si fa riferimento ad altri documenti prodotti.

Relazioni con il Requirements Analysis Document (RAD)
I test case pianificati nel Test Plan sono elaborati in relazione ai requisiti funzionali presentati nel RAD.

3 Panoramica del sistema

Il sistema si base su una architettura MVC.

- Azure e SQL per il database su cloud in fase di deployment
- Per la logica applicativa (back-end) sarà utilizzato Java con Spring
- Per il front-end sarà utilizzato Java con Android-Studio

4 Features da testare/da non testare

Di seguito la lista delle features di cui si effettuerà il testing per le varie gestioni:

- **Registrazione e Autenticazione**
  - Registrare un Giocatore
- **Gestione amici**
  - Inviare una richiesta di amicizia tramite ricerca manuale
- **Gestione lobby**
  - Entrare in lobby tramite Mostra lobby
  - Creare una lobby

5 Pass/Fail criteria

Le attività di testing sono mirate ad identificare la presenza di faults (errori) all'interno del sistema, per effettuare un successivo intervento di eliminazione.
L'esito di un test case è valutato mediante un oracolo, inteso come il risultato atteso della sua esecuzione, basandosi sui requisiti. Un test ha successo (pass) se, dato un input al sistema, l'output ottenuto è uguale all'output dell'oracolo. Un test fallisce (fail) se, dato un input al sistema, l'output ottenuto è diverso dall'output dell'oracolo.

Tutto il testing sarà considerato valido se i vincoli saranno rispettati:

- Testare gli use case specificati (nel punto 1 di questo documento);
- Raggiungere un branch coverage non inferiore al 70%.

6 Approccio

Nel processo di testing unitario, ogni metodo inerente alle funzionalità degli use case delle classi del sistema sarà testato. I casi di test saranno progettati con un approccio black-box e documentati nel codice usando il framework JUnit per il testing delle classi Java.

Per ciascuna classe di produzione (*Production Class*) sarà creata una classe di test corrispondente, denominata con il formato NomeTest. Le classi di test verranno sviluppate parallelamente alle rispettive classi di produzione, in modo da garantire una buona copertura del codice.

Le tecnologie utilizzate includeranno:

- **Mockito**: Utilizzato per creare mock e stub e isolare le componenti da testare.
- **JaCoCo**: Consente di misurare metriche di copertura del codice, come la Branch Coverage.
- **Maven**: Permette di gestire la build del progetto e automatizzare l'esecuzione dei test lato server.
- **JUnit**: Permette di definire casi di test isolati, configurazioni pre e post-esecuzione.

7 Sospensione e Ripristino

Questa sezione descrive i criteri per la sospensione delle attività di test e le azioni necessarie per il ripristino del processo.

Criteri di sospensione
Le attività di testing continueranno fino al loro completamento, anche in presenza di failure. Tuttavia, il processo di testing potrà essere temporaneamente sospeso se, durante l'esecuzione, viene rilevato un errore nella definizione di uno o più test.

Criteri di ripristino
Il testing riprenderà una volta che i fault individuati saranno stati corretti.

8 Materiale di testing

L'hardware necessario per l'attività di test è un semplice computer poiché abbiamo deciso di testare solo il server. Questa scelta è stata fatta perché il server è il punto focale dell'applicazione.

9 Test cases

L'approccio per la definizione dei test frame sarà il category partition.

9.1 Registrazione e Autenticazione

9.1.1 Registrare un Giocatore

Parametro: Nickname	
Nome categoria	**Scelte per la categoria**
Valore [VN]	1. Valore = Empty [error] 2. Valore != Empty [PROPERTY VN_OK]
Registrato [RN]	1. Già registrato = True [error] 2. Già registrato = False [PROPERTY RN_OK]
Parametro: Email	
Formato: [a-z0-9._%+\\-]+@[a-z0-9.\\-]+\\.[a-z]{2,}$	
Nome categoria	**Scelte per la categoria**
Valore [VE]	1. Valore = Empty [error] 2. Valore != Empty [PROPERTY VE_OK]
Formato [FE]	1. Rispetta il formato = False [error] 2. Rispetta il formato = True [PROPERTY FE_OK]
Registrata [RE]	1. Già registrata = True [error]

	2. Già registrata = False [PROPERTY RE_OK]
Parametro: Password	
Nome categoria	**Scelte per la categoria**
Valore [VP]	1. Valore = Empty [error] 2. Valore != Empty [PROPERTY VP_OK]

Test case ID	Test frame	Esito
TC_1.1_1	VN1	Errato: Il campo nickname non può essere vuoto
TC_1.1_2	VN2, RN1	Errato: Il nickname è già in uso
TC_1.2_1	VN2, RN2, VE1	Errato: il campo email non può essere vuoto
TC_1.2_2	VN2, RN2, VE2, FE1	Errato: l'email non corrisponde al formato
TC_1.2_3	VN2, RN2, VE2, FE2, RE1	Errato: l'email è già registrata
TC_1.3_1	VN2, RN2, VE2, FE2, RE2, VP1	Errato: il campo password non può essere vuoto
TC_1.4	VN2, RN2, VE2, FE2, RE2, VP2	Corretto: registrazione effettuata

9.2 Gestione lobby

9.2.1 Entrare in una Lobby Online Pubblica tramite Mostra lobby

Parametro: Lista lobby pubbliche	
Nome categoria	**Scelte per la categoria**
Dimensione [DLL]	1. Dimensione = 0 [error] 2. Dimensione > 0 [PROPERTY DLL_OK]
Parametro: Numero di giocatori nella lobby scelta	
Nome categoria	**Scelte per la categoria**
Quantità [QTL]	1. Quantità giocatori >= 18 [error] 2. Quantità giocatori < 18 [PROPERTY QTL_OK]
Parametro: Stato delle lobby	
Nome categoria	**Scelte per la categoria**
Esito [ESL]	1. Esito = in corso [error] 2. Esito = attesa giocatori [PROPERTY ESL_OK]

Test case ID	**Test frame**	**Esito**
TC_2.1_1	DLL1	Errato: non ci sono lobby pubbliche attive
TC_2.2_1	DLL2, QTL1	Errato: lobby piena, non può entrare

TC_2.3_1	DLI2, QT2L, ESL1	Errato: partita già in corso, non può entrare
TC_2.4	DLI2, QTL2, ESL2	Corretto: il giocatore si è unito alla lobby

9.2.2 Creare una lobby

Parametro: Giocatore	
Nome categoria	**Scelte per la categoria**
Presenza [PG]	1. Presenza = True [error] 2. Presenza = False [PROPERTY PG_OK]
Parametro: Lobby	
Nome categoria	**Scelte per la categoria**
Tipo [TL]	1. Tipo != "Pubblica" or "Privata" [error] 2. TIpo = "Pubblica" or "Privata" [PROPERTY TL_OK]

Test case ID	Test frame	Esito
TC_3.1_1	PG1	Errato: il giocatore è già presente in una lobby, non può crearne una
TC_3.2_1	PG2, TL1	Errato: il tipo della lobby non è supportato

TC_3.3	PG2, TL2	Corretto: Lobby creata con successo

9.3 Gestione Amici

9.3.1 Inviare una Richiesta di amicizia tramite Ricerca Manuale

Parametro: Giocatore cercato	
Nome categoria	**Scelte per la categoria**
Trovato [TG]	1. Trovato = False [error] 2. Trovato = True [PROPERTY TG_OK]
Parametro: Relazione con Destinatario	
Nome categoria	**Scelte per la categoria**
Amicizia [AD]	1. Amicizia = True [error] 2. Amicizia = False [PROPERTY AD_OK]
Parametro: Richiesta di amicizia	
Nome categoria	**Scelte per la categoria**
Stato [SRA]	1. Stato = già inviata[error] 2. Stato != già inviata [PROPERTY SRA_OK]
Invio [IRA]	1. Invio = False [error] 2. Invio = True [PROPERTY IRA_OK]

Test case ID	**Test frame**	**Esito**

TC_4.1_1	TG1	Errato: il giocatore cercato non esiste
TC_4.2_1	TG2, AD1	Errato: il destinatario è già presente nella lista amici
TC_4.3_1	TG2, AD2, SRA1	Errato: richiesta di amicizia già inviata
TC_4.4	TG2, AD2, SRA2, IRA2	Corretto: Richiesta di amicizia inviata