|  |
| --- |
| **Università degli Studi di Salerno Corso di Ingegneria del Software** |

**BizBong  
Testing  
Versione 1.1**

**LOGO PROGETTO**

Data: 27/01/2017

**Partecipanti:**

|  |  |
| --- | --- |
| Nome | Matricola |
| **Vladyslav Sikorskyy** | **0512103106** |
| **Renato Matarazzo** | **0512103256** |
| **Pietro Clemente** | **0512102136** |
| **Michele Citro** | **0512103064** |

|  |  |
| --- | --- |
| **Scritto da:** | **Vladyslav Sikorskyy,Renato Matarazzo,Pietro Clemente,Michele Citro** |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Descrizione | Autore |
| **27/01/2017** | **1.0** | **Sono stati inseriti i seguenti campi:**  **-Introduzione**  **-Funzionalità da testare**  **-Approccio**  **-Testing di unità**  **-Testing di integrazione**  **-Testing di sistema** | **Vladyslav Sikorskyy,Renato Matarazzo** |
| **27/01/2017** | **1.1** | **Sono stati inseriti i seguenti campi:**  **-Sospensione e ripresa**  **-Criteri di sospensione**  **-Criteri di ripresa**  **-Materiale per il testing**  **-Test cases** | **Vladyslav Sikorskyy,Renato Matarazzo,Michele Citro,Pietro Clemente** |
| **27/01/2017** | **1.2** | **Sono stati inseriti i seguenti campi:**  **-Classi di equivalenza** | **Vladyslav Sikorskyy,Renato Matarazzo,Michele Citro,Pietro Clemente** |

Indice

**1. INTRODUZIONE 4**

**2. Funzionalità da testare…………………………………………………………..4**

**2.1 Approccio……………………………………………………………………..4**

**2.2 Testing di unità…………………………………………………………….4-5**

**2.3 Testing di integrazione………………………………………………………5**

**2.4 Testing di sistema……………………………………………………………5**

**3. Sospensione e ripresa……………………………………………………………5**

**3.1 Criteri di sospensione……………………………………………………….5**

**3.2 Criteri di ripresa…………………………………………………………….5**

**3.3 Materiale per il testing……………………………………………………...6**

**4. Test cases…………………………………………………………………………6**

**5. Classi di equivalenza…………………………………………………………….7**

**1. Introduzione**

Lo scopo del documento è quello di definire i test case su cui verranno

testate le funzionalità del sistema. Per ogni funzionalità saranno forniti un

numero sufficiente di istanze di input in modo tale da fornire almeno un

test case composto da dati corretti.

Il testing è una tecnica che ha lo scopo di rompere il sistema, cioè quello di identificare errori con l'obiettivo di evitare che si presentino durante l'utilizzo del software da parte dell'utente, e non quello di garantire che il codice sia corretto. In altre parole si cerca di creare quanti più fallimenti o stati erronei in modo pianificato per consentire agli sviluppatori di correggerli prima che il prodotto sia consegnato al cliente. Lo scopo di questo documento è quello di descrivere e pianificare l'attività di testing su determinate funzionalità del sistema.

**2. Funzionalità da testare**

Le attività di test sono state pianificate per le seguenti gestioni:

· Gestione Registrazione

· Gestione Login

· Gestione Partita

· Gestione Connessione

· Gestione Sessione

**2.1 Approccio**

Le tecniche di testing adottate riguarderanno inizialmente il testing di unità

dei singoli componenti, in modo da testare nello specifico la correttezza di

ciascuna unità.

Seguirà il testing d’integrazione, che focalizzerà l’attenzione

principalmente sul test delle interfacce delle suddette unità.

Infine verrà eseguito il testing di sistema, che vedrà come oggetto di

testing l’intero sistema assemblato nei suoi componenti. Quest’ ultimo

servirà soprattutto a verificare che il sistema soddisfi le richieste del

committente.

**2.2. Testing di Unità**

Durante questa fase, verranno ricercate le condizioni di fallimento isolando

i componenti ed usando test driver e test stub, cioè implementazioni

parziali di componenti che dipendono o da cui dipendono le componenti

da testare. La strategia utilizzata per il testing si baserà esclusivamente

sulla tecnica BlackBox, che si focalizza sul comportamento Input/Output,

ignorando la struttura interna della componente. Al fine di minimizzare il

numero dei test cases, i possibili input verranno partizionati in classi di

equivalenza e per ogni classe verrà selezionato un test case . Gli stati

erronei scovati in questa, come in qualsiasi altra fase di testing, che

comporteranno un fallimento del sistema, dovranno essere

tempestivamente comunicati agli implementatori al fine di correggerli e

ripristinare il testing al più presto.

**2.3. Testing d’integrazione**

In questa fase si procederà all’integrazione delle componenti di una

funzionalità che verranno testate nel complesso attraverso una strategia di

Bottom-Up. Si passerà, poi, alla funzionalità successiva fino ad esaurire le

funzionalità implementate. Quest’approccio mira principalmente a ridurre

le dipendenze tra funzionalità differenti e a facilitare la ricerca di errori

nelle interfacce di comunicazione tra sottosistemi.

**2.4. Testing di Sistema**

Lo scopo di questa fase del testing è quello di dimostrare che il sistema

soddisfi effettivamente i requisiti richiesti e che sia, quindi, pronto all’uso.

Come per il testing di unità, si cercherà di testare le funzionalità più

importanti per l’utente e quelle che hanno una maggiore probabilità di

fallimento. Si noti che, come per il testing di unità, si procederà attraverso

la tecnica BlackBox.

**3. Sospensione e Ripresa**

Di seguito si elencano i casi di sospensione e ripresa dell’applicazione e di come quest’ultima reagisce in questi specifici casi.

**3.1 Criteri di sospensione**

La fase di Testing del sistema verrà sospesa quando si raggiungerà un

compromesso tra qualità del prodotto e costi dell'attività di Testing. Il

Testing verrà quindi portato avanti quanto più possibile nel tempo senza

però rischiare di ritardare la consegna finale del progetto.

**3.2 Criteri di ripresa**

In seguito ad ogni modifica o correzione delle componenti che

genereranno errori o fallimenti, i test case verranno sottoposti nuovamente

al sistema assicurandoci così di aver risolto effettivamente il problema.

**3.3 Materiale per il testing (Requisiti**

**Hardware/software)**

L’hardware necessario per l’attività di test è un dispositivo smartphone/tablet Android, di versione minima 4.2.

E’ necessario predisporre di una connessione ad internet per accedere alla maggior parte dei contenuti all’interno dell’applicazione.

**4. Test Cases**

Le varie fasi di testing necessiteranno ognuna di test case utili ad

individuare errori ed anomalie sia analizzando il codice che provando la

sua esecuzione.

Si sono individuate varie classi di equivalenza per ogni input che possa

essere immesso per l’utilizzo di una o più componenti. In tal modo, è

possibile sviluppare test case con input delle tipologie identificate per

testare una o più unità.

Di seguito sono elencate le classi di equivalenza che saranno prese in

considerazione durante i successivi documenti di Testing per sviluppare i

test case.

**5. Classi di equivalenza**

· Gestione Registrazione

· Gestione Login

· Gestione Partita

· Gestione Connessione

· Gestione Sessione

1. **Registrazione**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Classe** | **Valido** | **Classe** | **Non Valido** |
| Nickname | CE01 | Nickname≠ ‘ ’ | CE02 | Nickname= ‘ ’ |
|  |  | Nickname≠NickEsistente |  |  |
| Password | CE03 | Password ≠ ‘ ’ | CE04 | Password = ‘ ’ |
|  |  |  |  |  |
|  |  |  |  |  |
| E-Mail | CE05 | E-Mail≠ ‘ ’  E-Mail≠E-MailEsistente | CE06 | E-Mail= ‘ ’  E-Mail=E-MailEsistente |