

Corso di Laurea in Informatica, prof. A. De Lucia, a.a. 2021-22

Progetto di Ingegneria del Software

Immagine che contiene testo

Descrizione generata automaticamente  
  
[» GitHub Repository](https://github.com/is-shodan-21-22)

Scadenza: 07-10-2021❯ Project Proposal

Scadenza: 14-10-2021❯ Problem Statement

Scadenza: 28-10-2021❯ Requisiti e casi d’uso

Scadenza: 11-11-2021❯ Requisiti e casi d’uso

Scadenza: 25-11-2021❯ System Design Document

Scadenza: 16-12-2021  
➥ **Piano di test e specifica interfacce moduli**

Scadenza: alla consegna del progetto❯ Altri documenti

❯ Esecuzione dei test

❯ Object Design

❯ Codice sorgente

❯ Report sul lavoro individuale

**Shodan** [IS-2021/2022-DE LUCIA]  
Partecipanti del progetto

|  |  |  |
| --- | --- | --- |
| Nome | Matricola | E-mail |
| Antonio Gravino | 05121 07161 | a.gravino3@studenti.unisa.it |
| Dario Trinchese | 05121 07479 | d.trinchese2@studenti.unisa.it |
| Raffaele Zheng | 05121 09015 | r.zheng@studenti.unisa.it |
| Carmine Fabbri | 05121 07353 | c.fabbri@studenti.unisa.it |
| Carmine Napolitano | 05121 06417 | c.napolitano44@studenti.unisa.it |

Questo documento tratta esclusivamente il **Test Plan** del progetto.  
Per ulteriore documentazione valida ai fini dell’esame, consultare la repository *docs*.

**Introduzione**

Il test plan è un documento che si focalizza sugli aspetti manageriali del testing: gestisce lo sviluppo e le attività di testing effettuate sul sistema creato. Saranno identificati gli elementi e le funzionalità da testare, le strategie di testing, gli strumenti utilizzati per effettuarlo. Lo scopo del testing è quello di rilevare gli errori in maniera pianificata all’interno del codice realizzato, in modo che essi non si ripetano durante l’utilizzo da parte dell’utente finale. I risultati dei test servono per intervenire nei punti in cui sono presenti défaillance.

**Relazioni con altri documenti**

Il test planning è in stretta relazione con i documenti prodotti fino; infatti, i test case sono basati sulle funzionalità individuate in questi, particolarmente nel “documento di raccolta ed analisi dei requisiti” -(RAD).

La definizione dei sottosistemi ed i servizi da porre al testing sono stati individuati tramite il “System Design Document” -(**SDD**) e le classi sono state definite sulla base dell’“Object Design Document” -(**ODD**).

**Funzionalità da testare**

Di seguito le funzionalità introdotte nel sistema che saranno sottoposte a test, suddivise per ogni gestione del sistema:

**Gestione Account**

* Registrazione utente
* Login
* Logout
* Modifica password
* Modifica e-mail

**Gestione Blog**

* Aggiungi un articolo
* Rimuovi un articolo

**Gestione Game**

* Aggiungi un titolo
* Rimuovi un titolo
* Aggiorna un titolo
* Ricerca un titolo

**Gestione Payment**

* Ricarica saldo
* Aggiungi carta
* Paga ora

**Gestione Cart**

* Aggiungi al carrello
* Rimuovi dal carrello
* Svuota carrello

# **Criteri pass / fail testing**

Lo scopo del testing è trovare una failure nel programma, quindi, il testing va a buon fine se l’output osservato è diverso dall’output atteso; una volta riscontrata una failure (nel caso sia presente nel sistema), si interviene per correggerla e si effettua di nuovo il testing per appurare che non abbia prodotto effetti collaterali, producendo altre failure.

**Approccio**

L’approccio della fase di testing si compone di tre fasi. Si inizia con il testing di unità che ha lo scopo di testare le componenti del sistema singolarmente, poi si passa al testing di integrazione in cui le componenti del sistema verranno combinate e testate come un unico gruppo ed infine, si effettua il testing di sistema che mira a verificare il funzionamento dell’intero sistema.

Per il testing di unità si utilizzerà la tecnica “BLACK-BOX”. Con il “BLACK-BOX” testing ci focalizzeremo sul comportamento dell’input/output delle singole componenti senza tener conto della loro struttura interna. A causa della mancanza di fattibilità di effettuare un test esaustivo per l’ingente quantitativo di dati di input, verrà utilizzata la strategia del “Category Partition”, che consente di decomporre lo spazio di input in categorie per poi partizionare le “categorie” in classi di equivalenza chiamate “scelte”. Al termine saranno specificate le “combinazioni” delle scelte da testare creando delle istanze di casi di test specificando i valori dei dati effettivi per ciascuna scelta e determinare i risultati corrispondenti. Mediante il “Category Partition” otterremo, quindi, un test efficiente e privo di ridondanze.

Per il testing d’integrazione si utilizzerà la tecnica “WHITE-BOX”. La “WHITE-BOX” testing viene utilizzata per testare la logica interna del sottosistema o dell’oggetto considerato, infatti, per trovare un errore nel codice bisogna usare dei dati che percorrono la parte errata del programma. Per testare una parte di programma si introduce il concetto di “cammino”, ovvero, una sequenza di istruzioni attraversata durante un’esecuzione. Non esiste un criterio capace di testare ogni singolo cammino (dato l’elevato numero di questi ultimi), è possibile trovare un numero finito di cammini indipendenti che combinati tra loro forniscono la maggior parte dei restanti cammini.

Per determinare se il sistema rispecchia tutti i requisiti funzionali e globali allora sarà effettuato il testing di sistema tramite il tool “Selenium”.

**Sospensione e ripresa**

La fase di testing può essere sospesa se si riscontrasse un difetto nel sistema per essere poi ripresa dopo aver risolto il problema riscontrato.

**Materiale per il testing**

Gli strumenti utilizzati sono:

* Web Server Apache Tomcat 9 in locale, il quale gira sul sistema
* Client-web per effettuare richieste al server
* MySQL che gestisce il database
* Selenium IDE per il test di sistema
* JUnit per il test di unità ed il test di integrazione

*TEST GESTIONE ACCOUNT*

**TC\_Registrazione**

|  |  |
| --- | --- |
| Parametro: Username | |
| Categorie | Scelte |
| Lunghezza Username - LU | 1. Lunghezza = 0 – campo vuoto [property **invalidLUValue**]  2. Lunghezza >= 1 [property **validLUValue**] |
| Formato Username – FU | 1. Non rispetta il formato [*if validLUValue*] [property **invalidFUValue**]  2. Rispetta il formato [*if validLUValue*] [property **validFUValue**] |
| Esiste nel database - EXUS | 1. Non esiste nel database [*if validFUValue*] [property **NotEXUS**]  2. Esiste nel database [*if validFUValue*] [property **EXUS**] |

|  |  |
| --- | --- |
| Parametro: Password | |
| Categorie | Scelte |
| Lunghezza Password – LP | 1. Lunghezza = 0 – campo vuoto [property **invalidLPValue**]  2. Lunghezza >= 1 [property **validLPValue**] |
| Formato Password – FP | 1. Non rispetta il formato [*if validLPValue*] [property **invalidFPValue**]  2. Rispetta il formato [*if validLPValue*] [property **validFPValue**] |

|  |  |
| --- | --- |
| Parametro: Ripeti password | |
| Categorie | Scelte |
| Lunghezza Ripeti Password – LRP | 1. Lunghezza = 0 – campo vuoto [property **invalidLRPValue**]  2. Lunghezza >= 1 [property **validLRPValue**] |
| Formato Ripeti Password – FRP | 1. Non rispetta il formato [*if validLRPValue*] [property **invalidFRPValue**]  2. Rispetta il formato [*if validLRPValue*] [property **validFRPValue**] |
| Coincidenza - OVLP | 1. Il campo non coincide con il campo Password [*if validFRPValue*] [property **NotOverlap**]  2. il campo coincide con il campo Password [*if validFRPValue*] [property **Overlap**] |

|  |  |
| --- | --- |
| Parametro: E-mail | |
| Categorie | Scelte |
| Lunghezza E-mail – LE | 1. Lunghezza = 0 – campo vuoto [property **invalidLEValue**]  2. Lunghezza >= 1 [property **validLEValue**] |
| Formato E-mail – FE | 1. Non rispetta il formato [*if validLEValue*] [property **invalidLEValue**]  2. Rispetta il formato [*if validLEValue*] [property **validFEValue**] |
| Esiste nel database - EXEM | 1. Non esiste nel database [*if validFEValue*] [property **NotEXEM**]  2. Esiste nel database [*if validFEValue*] [property **EXEM**] |

|  |  |  |
| --- | --- | --- |
| CODICE | COMBINAZIONE | ESITO |
| TC\_Registrazione1 | LU2.FU2.EXUS2.LP2.FP2.LRP2.FRP2.OVLP2.LE2.FE2.EXEM2 | Registrazione avvenuta con successo! |
| TC\_Registrazione2 | LU2.FU2.EXUS2.LP2.FP2.LRP2.FRP2.OVLP2.LE2.FE2.EXEM1 | E-mail già esistente |
| TC\_Registrazione3 | LU2.FU2.EXUS2.LP2.FP2.LRP2.FRP2.OVLP2.LE2.FE1 | Formato e-mail non valido |
| TC\_Registrazione4 | LU2.FU2.EXUS2.LP2.FP2.LRP2.FRP2.OVLP2.LE1 | E-mail - campo obbligatorio |
| TC\_Registrazione5 | LU2.FU2.EXUS2.LP2.FP2.LRP2.FRP2.OVLP1.LE2.FE2.EXEM2 | Le password non coincidono |
| TC\_Registrazione6 | LU2.FU2.EXUS2.LP2.FP2.LRP2.FRP1.LE2.FE2.EXEM2 | Formato ripeti password non valido |
| TC\_Registrazione7 | LU2.FU2.EXUS2.LP2.FP2.LRP1.LE2.FE2.EXEM2 | Ripeti password - campo obbligatorio |
| TC\_Registrazione8 | LU2.FU2.EXUS2.LP2.FP1.LRP2.FRP2.OVLP2.LE2.FE2.EXEM2 | Formato password errato |
| TC\_Registrazione9 | LU2.FU2.EXUS2.LP1.LRP2.FRP2.OVLP2.LE2.FE2.EXEM2 | Password - campo obbligatorio |
| TC\_Registrazione10 | LU2.FU2.EXUS1.LP2.FP2.LRP2.FRP2.OVLP2.LE2.FE2.EXEM2 | Username già esistente |
| TC\_Registrazione11 | LU2.FU1.LP2.FP2.LRP2.FRP2.OVLP2.LE2.FE2.EXEM2 | Formato username non valido |
| TC\_Registrazione12 | LU1.LP2.FP2.LRP2.FRP2.OVLP2.LE2.FE2.EXEM2 | Username - campo obbligatorio |