# Testing plan
# Progetto
# Sorting Hat

| | |
|---|---|
| Riferimento | |
| Versione | 1.1 |
| Data | 14/02/2022 |
| Destinatario | Docente Ingegneria del Software 2021/22 |
| Presentato da | Ascione Josef, Di Gregorio Emanuele, Di Sarno Davide, Sacco Matteo |
| Approvato da | |

## Revision History

| Data | Versione | Descrizione | Autori |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 26/11/2021 | 0.1 | Prima stesura | Ascione J., Di Gregorio E., Di Sarno D., Sacco M. |
| 27/11/2021 | 0.2 | Modifica panoramica | Ascione J., Di Gregorio E., Di Sarno D., Sacco M. |
| 07/12/2021 | 0.3 | Raffinamento criteri Pass/Fail | Ascione J., Di Gregorio E., Di Sarno D., Sacco M. |
| 09/12/2021 | 0.4 | Aggiunta Cicli | Ascione J., Di Gregorio E. |
| 10/12/2021 | 0.5 | Pianificazione testing e specifica approccio | Di Sarno D., Ascione J. |
| 11/12/2021 | 1.0 | Revisione Documento | Ascione J., Di Gregorio E. |
| 14/02/2022 | 1.1 | Revisione Finale | Di Gregorio E., Di Sarno D. |

# Sommario

# 1. Introduzione

Il test plan è stato creato per documentare lo scopo, gli approcci, le risorse e il programma delle attività di test.

Include le feature da testare e da non testare, i criteri di successo e fallimento, gli strumenti per il testing e i test case da sviluppare per poter ottimizzare il prodotto finale.

# 2. Relazione con gli altri documenti

- **Libro: Object-Oriented Software Engineering (Using UML,Patterns,and Java) Third Edition**
  Autori: Bernd Bruegge & Allen H.Dutoit
- **Relazione con il System Design Document(SDD)**

  La relazione tra test plan e SDD riguarda la suddivisione del sistema in sottosistemi

  poiché i test dei vari componenti devono rimare fedele a questa partizione.

- **Relazione con il Requirement Analysis Doument(RAD)**

  Relazione che consegue dalla presenza nel RAD dei requisiti funzionali e non funzionali testati dai test case.

## 3. Panoramica del Sistema

Il sistema fornisce tutte le sue funzionalità attraverso una Web Application. Per essere certi che tutte le funzionalità funzionino a dovere verranno testate, se queste funzionalità otterranno risultato positivi dai test (che vuol dire che eseguono le loro funzionalità) possiamo intendere che il Sistema SortingHat ha raggiunto gli obiettivi prefissati.

## 4. Funzionalità del Sistema

Feature to be tested:

- Registrazione nuovo utente ordinario
- Registrazione nuovo utente universitario
- Login
- Aggiunta discussioni

Feature not to be tested:

- Visualizzazione statistiche
- Validazione form(MODULO IA)

## 5. Pass/Fail criteria

Fault e bug saranno documentati e analizzati in seguito ai risultati dei test. Considereremo e segneremo il testing come “pass” (riuscito) quando non si avranno deviazioni dal comportamento atteso sia nell’interazione con l0utente e sia nel comportamento del sistema alle funzioni prese in esame.

Un testing sarà segnalato come “fail”(fallimento) quando, tenendo conto che per lo standard IEEE per “fail” si intende il successo del test, incontreremo deviazioni del comportamento del sistema o operazioni che porteranno comunque allo stato di failure.

Infine, annoteremo un testing in esecuzione come “WIP” (Work in progress) oppure “blocked” quando l’esecuzione non riuscirà a terminare.

# 6. Approccio

Il testing del sistema comprenderà :

- il testing di unità per poter garantire il comportamento voluto nei singoli sottosistemi ;
- il testing di integrazione per la verifica delle interazioni tra i sottosistemi, necessaria per verificare il corretto funzionamento del sistema quando più unita collaborano;
- Il testing di sistema per definire se i requisiti funzionali sono stati soddisfatti. Inoltre il Testing di Sistema include anche il Testing di Performance e il Testing Pilota.

## 6.1 Testing di unità

In questa fase sono analizzati i metodi e gli oggetti utilizzati per svolgere le funzionalità richieste nei vari sottosistemi.

Il testing del codice è stato realizzato usufruendo dei tool presenti nell’ambiente di sviluppo e gli strumenti riportati nel punto 7.

La strategia prevista per affrontare tale attività consiste nel testare i metodi delle classi del Sistema e, in particolare, quelli che permettono le validazioni dell’input dell’utente, come descritto nei test cases. I casi di test saranno definiti attraverso l’uso del framework JUnit. Inoltre , per isolare le componenti da testare è stato utilizzato Mockito, mentre per il calcolo della Branch Coverage è stato sfruttato JoCoCo.

## 6.2 Testing di integrazione

L’attività di Testing di Integrazione consiste nell’integrare le componenti testate singolarmente in sottosistemi più grandi, rilevando bug non evidenziati con il Testing di unità. La strategia prevista è stata del testing Bottom-Up, poiché ritenuta migliore per software Objected-Oriented ovvero il paradigma su cui si basa il nostro Sistema. Sono stati testati tutti i metodi relativi alle classi DAO, ovvero i metodi contenenti le query per le interazioni con il database, e i formMapper che invocano questi metodi.

Le tecnologie sono le stesse di quelle del Testing di unità.

## 6.3 Testing di sistema

In questa fase sono testati i requisiti funzionali(Functional Testing) per individuare gli errori definiti dagli input degli utenti.

L'approccio utilizzato per testare i requisiti funzionali definiti nel documento "RAD", è stato quello del "Black-box Testing". Per individuare e descrivere i casi di test si è scelto di utilizzare la tecnica della "Category Partition" , la quale si basa sulla partizione dell'insieme dei possibili input dell'utente in classi di equivalenza.

L'attività di Testing di sistema è effettuata tramite Selenium IDE e consiste nell'utilizzare tale tool per verificare che i test case descritti nel Testing funzionale non rilevino errori.

## 7. Strumenti per il testing (hardware/software)

Lo strumento hardware utilizzato di testing è il computer e, poiché il Sistema non è ancora rilasciato, non deve essere connesso necessariamente ad Internet.

Gli strumenti software utilizzati per le attività di testing sono:

- JaCoCo: libreria per il Code Coverage in ambito Java
- Selenium IDE: tool open source utilizzato per la gestione automatizzata dei browser, utilizzato come framework di testing. Permette di registrare le azioni che un utente può effettuare nell'interazione con il Sistema, così da poter eseguire i casi di test.
- Mockito : framework utilizzato per la realizzazione di oggetti mock durante il Test di Unità.
- JUnit: framework open source per scrivere ed eseguire Testing di Unità per Java

## 8. Test cases

L'esecuzione dei test case verrà utilizzato il metodo del Category partition. Questo metodo permetterà di testare le funzioni del sistema individuando parametri suddivisi in categorie distinte che saranno in seguito divise in scelte con valori precisi.
Il Test Cases Specification(TCS) documenterà ogni test.

## 9. Specifica dei test cases

## 9.1 Gestione cicli di dottorato

### TC_1.1 Aggiunta Discussione

| **Parametro:** Corpo<br>**Formato:** [0-9,a-z,A-Z]{2-10000} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza li** | **1:** lunghezza >2 [errore]<br>**2:** lunghezza >=2 && lunghezza <=10000 [property **lunghezzaLIok**]<br>**3:** lunghezza >10000[errore] |
| **formato fi** | **1:** rispetta il formato [if **lunghezzaLIok**] [property **formatoFIok**]<br>**2:** non rispetta il formato [if **lunghezzaLIok**] [errore] |

| **Parametro:** Titolo<br>**Formato:** [0-9,a-z,A-Z]{2-50} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza ld** | **1:** lunghezza >2 [errore]<br>**2:** lunghezza >= 2 && <=50[property **lunghezzaLDok**]<br>**3:** lunghezza > 50 [errore] |

| **Codice** | **Combinazione** | **Esito** |
|---|---|---|
| TC_1.1_01 | li1 | errore |
| TC_1.1_02 | li3 | errore |
| TC_1.1_03 | li2.fi2 | errore |
| TC_1.1_04 | li2.fi1.ld1 | errore |
| TC_1.1_05 | li2.fi1.ld3 | errore |

| | | |
|---|---|---|
| TC_1.1_06 | Li2.fi1.ld2 | inserimento |

**TC_1.2 Registrazione Utente Ordinario**

| **Parametro:** Nome<br>**Formato:** [a-z, A-Z] {2-20} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza ln** | **1:** lunghezza <2 [errore]<br>**2:** lunghezza >=2 && lunghezza <=20 property **lunghezzaLnok**]<br>**3:** lunghezza >20 [errore] |
| **formato fn** | **1:** rispetta il formato [if **lunghezzaLnok**] [property **formatoFnok**]<br>**2:** non rispetta il formato [if **lunghezzaLnok**] [errore] |

| **Parametro:** Cognome<br>**Formato:** [a-z, A-Z] {2-20} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza lc** | **1:** lunghezza <2 [errore]<br>**2:** lunghezza >=2 && lunghezza <=20 [property **lunghezzaLcok**]<br>**3:** lunghezza >20 [errore] |
| **formato fc** | **1:** rispetta il formato [if **lunghezzaLcok**] [property **formatoFcok**]<br>**2:** non rispetta il formato [if **lunghezzaLcok**] [errore] |

| **Parametro:** email<br>**Formato:** [a-z, A-Z,0-9, parola@parola.parola ] {6-50} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza le** | 1: lunghezza <6 [errore] |

| | |
|---|---|
| | 2: lunghezza >=6 && lunghezza <=50 [property lunghezzaLeok]<br>3: lunghezza >50 [errore] |
| **formato fe** | 1: rispetta il formato [if lunghezzaLeok] [property formatoFeok]<br>2: non rispetta il formato [if lunghezzaLeok] [errore] |
| **esiste ee** | 1: esiste nel DB [if lunghezzaLeok and formatoFeok] [errore]<br>2: non esiste nel DB [if lunghezzaLeok and formatoFeok] [property esisteEEok] |

| **Parametro:** Password<br>**Formato:** [a-z, A-Z,0-9] {2-20} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza lp** | **1:** lunghezza <2 [errore]<br>**2:** lunghezza =>2 or =<20 [property **lunghezzaLpok**]<br>**3:** lunghezza >20 [errore] |
| **formato fp** | **1:** rispetta il formato [if **lunghezzaLpok**] [property **formatoFpok**]<br>**2:** non rispetta il formato [if **lunghezzaFpok**] [errore] |

| **Codice** | **Combinazione** | **Esito** |
|---|---|---|
| TC_1.2_01 | ln1 | errore |
| TC_1.2_02 | ln3 | errore |
| TC_1.2_03 | ln2.fn2 | errore |
| TC_1.2_04 | ln2.fn1.lc1 | errore |
| TC_1.2_05 | ln2.fn1.lc3 | errore |
| TC_1.2_06 | ln2.fn1.lc2.fc2 | errore |
| TC_1.2_07 | ln2.fn1.lc2.fc1.le1 | errore |

| | | |
|---|---|---|
| TC_1.2_08 | ln2.fn1.lc2.fc1.le3 | errore |
| TC_1.2_09 | ln2.fn1.lc2.fc1.le2.fe2 | errore |
| TC_1.2_10 | ln2.fn1.lc2.fc1.le2.fe1.ee1 | errore |
| TC_1.2_11 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp1 | errore |
| TC_1.2_12 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp3 | errore |
| TC_1.2_13 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp2.fp2 | errore |
| TC_1.2_14 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp1.fp1 | Registrazione |

**TC_1.3 Registrazione Utente Universitario**

| **Parametro:** Nome<br>**Formato:** [a-z, A-Z]{2-20} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza ln** | **1:** lunghezza <2 [errore]<br>**2:** lunghezza >=2 && lunghezza <=20 [property **lunghezzaLnok**]<br>**3:** lunghezza >20 [errore] |
| **formato fn** | **1:** rispetta il formato [if **lunghezzaLnok**] [property **formatoFnok**]<br>**2:** non rispetta il formato [if **lunghezzaLnok**] [errore] |

| **Parametro:** Cognome<br>**Formato:** [a-z, A-Z] {2-50} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza lc** | **1:** lunghezza <2 [errore]<br>**2:** lunghezza >=2 && lunghezza <=50 [property **lunghezzaLcok**] |

| | |
|---|---|
| | **3:** lunghezza >50 [errore] |
| **formato fc** | **1:** rispetta il formato [if **lunghezzaLcok**] [property **formatoFcok**]<br>**2:** non rispetta il formato [if **lunghezzaLcok**] [errore] |

| **Parametro:** email<br>**Formato:** [a-z, A-Z,0-9, parola@studenti.unisa.it ] {6-50} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza le** | 1: lunghezza <6 [errore]<br>2: lunghezza >=6 && lunghezza <=50 [property lunghezzaLeok]<br>3: lunghezza >50 [errore] |
| **formato fe** | 1: rispetta il formato [if lunghezzaLeok] [property formatoFeok]<br>2: non rispetta il formato [if lunghezzaLeok] [errore] |
| **esiste ee** | 1: esiste nel DB [if lunghezzaLeok and formatoFeok] [errore]<br>2: non esiste nel DB [if lunghezzaLeok and formatoFeok] [property esisteEEok] |

| **Parametro:** Password<br>**Formato:** [a-z, A-Z,0-9] {2-20} | |
|---|---|
| **Categorie** | **Scelte** |
| **lunghezza lp** | **1:** lunghezza <2[errore]<br>**2:** lunghezza =>2 or =<20 [property **lunghezzaLpok**]<br>**3:** lunghezza >20[errore] |
| **formato fp** | **1:** rispetta il formato [if **lunghezzaLpok**] [property **formatoFpok**] |

| | |
|---|---|
| | **2:** non rispetta il formato [if **lunghezzaFpok**] [errore] |

| Codice | Combinazione | Esito |
|---|---|---|
| TC_1.3_01 | ln1 | errore |
| TC_1.3_02 | ln3 | errore |
| TC_1.3_03 | ln2.fn2 | errore |
| TC_1.3_04 | ln2.fn1.lc1 | errore |
| TC_1.3_05 | ln2.fn1.lc3 | errore |
| TC_1.3_06 | ln2.fn1.lc2.fc2 | errore |
| TC_1.3_07 | ln2.fn1.lc2.fc1.le1 | errore |
| TC_1.3_08 | ln2.fn1.lc2.fc1.le3 | errore |
| TC_1.3_09 | ln2.fn1.lc2.fc1.le2.fe2 | errore |
| TC_1.3_10 | ln2.fn1.lc2.fc1.le2.fe1.ee1 | errore |
| TC_1.3_11 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp1 | errore |
| TC_1.3_12 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp3 | errore |
| TC_1.3_13 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp2.fp2 | errore |
| TC_1.3_14 | ln2.fn1.lc2.fc1.le2.fe1.ee2.lp1.fp1 | Registrazione |

## TC_1.4 Login

| |
|---|
| **Parametro:** email<br><br>**Formato:** [a-z, A-Z,0-9, parola@parola.parola ] {6-50} |

| Categorie | Scelte |
|---|---|
| **lunghezza le** | **1:** lunghezza <6 [errore]<br>**2:** lunghezza >=6 && lunghezza <=50 [property **lunghezzaLeok**]<br>**3:** lunghezza >50 [errore] |
| **formato fe** | **1:** rispetta il formato [if **lunghezzaLeok**] [property **formatoFeok**]<br>**2:** non rispetta il formato [if **lunghezzaLeok**] [errore] |
| **esiste ee** | **1:** esiste nel DB [if **lunghezzaLeok** and **formatoFeok**] [property **esisteEeok**]<br>**2:** non esiste nel DB [if **lunghezzaLeok** and **formatoFeok**] [errore] |

**Parametro:** Password

**Formato:** [a-z, A-Z,0-9] {2-20}

| Categorie | Scelte |
|---|---|
| **lunghezza lp** | **1:** lunghezza <2 [errore]<br>**2:** lunghezza =>2 or =<20 [property **lunghezzaLpok**]<br>**3:** lunghezza >20 [errore] |
| **formato fp** | **1:** rispetta il formato [if **lunghezzaLpok**] [property **formatoFpok**]<br>**2:** non rispetta il formato [if **lunghezzaFpok**] [errore] |
| **esiste ep** | **1:** esiste nel DB [if **lunghezzaLpok** and **formatoFpok**] [property **esisteEpok**]<br>**2:** non esiste nel DB [if **lunghezzaLpok** and **formatoFpok**][errore] |
| **equivalente vp** | **1:** l'email e la password inseriti sono equivalenti a quelli registrati nel DB [if **esisteEpok**] [property **equivalenteVpok**]<br>**2:** l'email e la password inseriti non sono equivalenti a quelli registrati nel DB [if **esisteEpok**] [errore] |

| Codice | Combinazione | Esito |
|---|---|---|
| TC_1.4_01 | le1 | errore |
| TC_1.4_02 | le3 | errore |
| TC_1.4_03 | le2.fe2 | errore |
| TC_1.4_04 | le2.fe1.ee2 | errore |
| TC_1.4_05 | le2.fe1.ee1.lp1 | errore |
| TC_1.4_06 | le2.fe1.ee1.lp3 | errore |
| TC_1.4_07 | le2.fe1.ee1.lp2.fp2 | errore |
| TC_1.4_08 | le2.fe1.ee1.lp2.fp1.ep2 | errore |
| TC_1.4_09 | le2.fe1.ee1.lp2.fp1.ep1.vp2 | errore |
| TC_1.4_10 | le2.fe1.ee1.lp2.fp1.ep1.vp1 | accesso |

# 10. Pianificazione del testing e Assegnazione dei ruoli

Tutte le specifiche dei Test Case sono state svolte nel Test Case Specification.

# 11. Testing schedule

I test verranno eseguiti ed analizzati durante l'implementazione delle porzioni di codice interessati e dopo l'implementazione finale del sistema per garantire nei report finali la massima correttezza e completezza del prodotto.

Essendo un team low-budget le fasi di testing saranno suddivise in modo equilibrato tra i membri con ruoli definiti in fase di implementazione.


Corso Triennale di Informatica – Università di Salerno
Corso di studi di Ingegneria del Software
Sorting Hat