

Università degli Studi di Salerno
Corso di Ingegneria del Software

FantaUnisa
Test Plan Document
Versione 1.0



Data: 21/01/2026

Progetto: FantaUnisa	Versione: 1.0
Documento: Test Plan Document	Data: 21/01/2026

Partecipanti:

Nome	Matricola
Clavino Antonio	0512119692
Corona Francesco	0512119827
Sabetta Francesco	0512118990
Tiberini Monica	0512120226

Scritto da:	Tutti i partecipanti.
--------------------	-----------------------

Revision History

Data	Versione	Descrizione	Autore
20/12/2025	0.1	Stesura introduttiva del documento	Sabetta Francesco
21/12/2025	0.2	Stesura sezioni 4, 5 e 6	Sabetta Francesco
21/12/2025	0.3	Stesura finale del documento	Sabetta Francesco
21/01/2026	1.0	Revisione finale	Tutti i partecipanti

Indice

1. Introduction	4
2. Relationship to other documents.....	4
3. System overview.....	4
4. Features to be tested/not to be tested	4
5. Pass/Fail Criteria	5
6. Approach.....	5
7. Suspension and resumption.....	6
8. Testing materials (hardware/software requirements).....	6
9. Testing schedule	6

1. Introduction

Il presente documento descrive il piano di test di sistema di **FantaUnisa**. L'obiettivo principale è verificare che il sistema soddisfi i requisiti funzionali e non funzionali specificati nel **Requirement Analysis Document (RAD)** e che l'architettura implementata sia conforme a quanto descritto nel **System Design Document (SDD)** e nell'**Object Design Document (ODD)**.

2. Relationship to other documents

- **RAD (Requirement Analysis Document):** Definisce i requisiti funzionali che fungono da oracolo per i test.
- **SDD (System Design Document):** Definisce l'architettura e l'ambiente hardware/software di riferimento.
- **ODD (Object Design Document):** Fornisce i dettagli sulle classi e i metodi da testare.

3. System overview

Lo scopo di **FantaUnisa** è la progettazione e lo sviluppo di un'applicazione web che supporti i fantallenatori nella scelta della formazione ideale da schierare ogni settimana, attraverso l'elaborazione di statistiche aggiornate, l'utilizzo di un algoritmo configurabile ed una community che interagisce tramite i post dei vari utenti della piattaforma.

L'architettura è basata sul pattern **MVC** (Model-View-Controller) a tre livelli:

1. **Presentation Layer:** Pagine JSP e Servlet.
2. **Application Logic Layer:** Insieme dei sottosistemi/funzionalità del sistema.
3. **Data Storage Layer:** Database relazionale MySQL gestito tramite DAO.

4. Features to be tested/not to be tested

Le seguenti funzionalità saranno oggetto di verifica, raggruppate per sottosistema:

4.1 Gestione Accessi e Profilo

- Registrazione di un nuovo Fantallenatore (controllo unicità e-mail).
- Login e Logout (gestione sessione).
- Modifica del profilo utente (password).
- Eliminazione account.

4.2 Gestione Rosa (Team Management)

- Verifica vincoli di composizione rosa (max 25 giocatori: 3P, 8D, 8C, 6A).

4.3 Gestione Formazione e Modulo

- Selezione e persistenza del modulo tattico (es. 4-3-3, 3-5-2).
- Schieramento della formazione (verifica titolari e panchinari).

- **4.4 Community**

- Creazione e pubblicazione di un post.
- Inserimento di commenti.
- Aggiunta/Rimozione di reazioni ai post.

Le seguenti caratteristiche sono escluse da questo piano di test:

- **Gestione Dati Statistici (Admin):** Le funzionalità di upload, parsing dei file Excel e aggiornamento delle statistiche non saranno testate. Si assume che i dati presenti nel database siano stati caricati correttamente o siano pre-popolati per l'ambiente di test.
- **Performance sotto stress estremo:** Non verranno eseguiti test di carico (DDoS) oltre la capacità nominale del server di sviluppo.
- **Sicurezza hardware:** Guasti fisici ai server o alla rete.
- **Librerie di terze parti:** Il funzionamento interno del driver JDBC o del container Apache Tomcat è assunto come corretto.
- **Compatibilità mobile nativa:** Il sistema sarà testato come web-app responsive, non come app nativa.

5. Pass/Fail Criteria

In accordo con i principi di Software Testing adottati, stabiliamo i seguenti criteri per valutare l'esito dell'esecuzione dei casi di test:

Definizione di Successo e Fallimento del Test

- **Successo del Test (Test Success):** Un caso di test è considerato "di successo" se l'esecuzione rivela un malfunzionamento o una discrepanza rispetto alle specifiche. In questo scenario, il test ha raggiunto il suo scopo di identificare un difetto nel sistema (il codice contiene un *fault*).

Criterio: Output Effettivo \neq Output Atteso

- **Fallimento del Test (Test Failure):** Un caso di test è considerato "fallito" se l'esecuzione non rivela alcun errore. Il sistema si è comportato esattamente come previsto dalle specifiche; quindi, il test non è riuscito a trovare bug.

Criterio: Output Effettivo $=$ Output Atteso

6. Approach

Il nostro approccio al testing per il progetto **FantaUnisa** è diviso in tre fasi sequenziali: test di unità, test d'integrazione e test di sistema.

La prima fase, ovvero il **test di unità**, ha lo scopo di verificare le funzionalità dei singoli componenti software in isolamento. Per quanto riguarda il **test d'integrazione**, esso consiste nell'unire progressivamente i componenti secondo una strategia di integrazione definita, per testare il loro funzionamento e la correttezza dello scambio dati quando vengono combinati, procedendo fino a quando l'integrazione non incorpora tutti i moduli del sistema. Per ultimo abbiamo quindi il **test di sistema**, il cui scopo è quello di verificare il corretto comportamento dell'intera applicazione web nel suo ambiente operativo finale.

Per effettuare i test di unità utilizziamo una tecnica **Black-Box**, ovvero una tecnica che effettua la selezione dei casi di test basandosi esclusivamente sulle specifiche dei componenti. In particolare, la tecnica da noi scelta è il **Category Partition**, che divide i domini di input in categorie e scelte significative; questo ci permette di coprire sistematicamente i comportamenti rilevanti del sistema riducendo al contempo il numero di test necessari.

Per effettuare i test di integrazione utilizziamo la tecnica **Sandwich**, che suddivide l'architettura di **FantaUnisa** in **tre layer logici**: il layer superiore, quello inferiore e quello target. Il testing avviene in parallelo partendo dal livello superiore e da quello inferiore, con l'obiettivo di far convergere l'integrazione verso il target layer centrale.

Infine, per il testing di sistema utilizzeremo il tool "**Selenium IDE**" per poter simulare le interazioni reali dell'utente tramite browser e verificare se il sistema rispecchia fedelmente i requisiti funzionali e globali precedentemente definiti.

7. Suspension and resumption

- **Sospensione:** L'attività di testing viene sospesa se vengono rilevati difetti critici che impediscono l'esecuzione dei casi di test successivi (es. impossibilità di effettuare il login o crash del database).
- **Ripresa:** L'attività riprende solo dopo la risoluzione dei difetti critici e nel caso in cui il test di regressione non rilevi ulteriori difetti.

8. Testing materials (hardware/software requirements)

Hardware

I test verranno eseguiti sulla seguente configurazione hardware (ambiente di sviluppo/staging):

- **Client:** PC Desktop e Laptop standard.

Software

La configurazione software richiesta per l'ambiente di test è:

- **Web Server:** Apache Tomcat 10.1.36.
- **Database:** MySQL Server 8.0.
- **Java Environment:** JDK 22 o superiore.
- **Strumenti di Test:** JUnit 5, Selenium IDE e Mockito.

9. Testing schedule

Fase	Attività	Durata
Preparazione ambiente	Configurazione piattaforme e database	1-2 giorni

Testing funzionalità di base	Registrazione, Autenticazione, Gestione Profilo	3 giorni
Testing funzionalità inerenti alla formazione	Gestione Squadra, Formazione, Modulo	5 giorni
Testing funzionalità Community	Creazione Post, Creazione Commenti, Creazione Reazioni	5 giorni

Totale stimato: 15 giorni lavorativi.