Università degli Studi di Salerno

Corso di Ingegneria del Software

UStrike!

Test Plan and Specifications

Versione 2.0

Data: 20/01/2026

Informazioni sul Progetto

Progetto: UStrike!
Documento: **Test Plan and specifications**
Versione: 2.0
Data: 20/1/2026

Coordinatore del progetto:
Nome
Matricola

Partecipanti:

Nome	Matricola
Russo Serena	0512119098
Stefanile Andrea	0512119557
Valito Marcello	0512119014

Scritto da:
Russo Serena, Stefanile Andrea, Valito Marcello

Revision History

Data	Versione	Descrizione	Autore
19/12/2025	0.1	Stesura iniziale del documento	Russo Serena, Stefanile Andrea
20/12/2025	0.2	Inserimento dei primi test cases e test frames	Stefanile Andrea, Valito Marcello
21/12/2025	0.3	Ultimazione di tutti i test cases e test frames e correzione di alcuni errori	Russo Serena, Stefanile Andrea, Valito Marcello
22/12/2025	1.0	Revisione e consegna del documento.	Russo Serena, Stefanile Andrea, Valito Marcello
18/01/2026	1.5	Revisione del documento e adattamento dei test ai test effettivi	Stefanile Andrea
20/01/2026	2.0	Revisione finale	Stefanile Andrea

1.Introduzione 3
2.Relazioni con altri documenti 4
3.Funzionalità da testare 4
3.1Gestione Account () 4
3.2Gestione Prenotazioni () 4
4.Criteri Pass/Fail Testing 4
5.Approccio e Strategie di Testing 4
5.1Unit Testing 4
5.2Integration Testing 5
5.3System Testing (Selenium) 5
5.4Sospensione e Ripresa 5
5.5Materiali per il testing 5
6.Test Cases and Test frames 6
6.1 – Login 6
Test Frame 6
Test cases 6
6.2 – Aggiungi Prenotazione 7
Test Frame 8
Test cases 8
6.3-Registrazione 10
Test Frame 10
Test cases 11
6.4-Rifiuta Prenotazione 14
Test Frame 15
Test Cases 15

1.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 UStrike! 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.

NB verranno testate anche funzionalità non implementate nella demo.

2.Relazioni con altri documenti

Il test planning è in stretta relazione con i documenti prodotti fino ad ora:

- RAD (Requirement Analysis Document): i test case sono basati sulle funzionalità individuate negli Use Cases e nella requirement elicitation.
- SDD (System Design Document): la definizione dei sottosistemi.
- ODD (Object Design Document): le classi e le interfacce definiscono il scope esatto del testing

3.Funzionalità da testare

Le funzionalità sottoposte a test, suddivise per gestione del sistema:

3.1Gestione Account ()

- Login (cliente, staff, manager)
- Registrazione(cliente)
- Modifica profilo personale(cliente)
- Modifica password(cliente)

3.2Gestione Prenotazioni ()

- Aggiungi prenotazione (cliente)
- Annulla prenotazione (cliente)
- Accetta prenotazione (staff)
- Rifiuta prenotazione (staff)

4.Criteri Pass/Fail Testing

Pass: L'output osservato non corrisponde all'output atteso secondo le post-condizioni specificate nell'ODD.
Fail: L'output osservato è uguale all'output atteso.
Una volta riscontrata una failure:

1. Si registra il difetto
2. Si interviene per correggerlo
3. Si effettua il testing nuovamente per appurare che non abbia prodotto effetti collaterali

5.Approccio e Strategie di Testing

L'approccio della fase di testing si compone di tre livelli:

5.1Unit Testing

In questa fase abbiamo testato ogni singola unità a sé stante, abbiamo utilizzato un approccio di white-box.
I DAO nella fase di unit test sono stati testati usando la tecnica White-Box questo per assicurarci che sintassi e mapping siano corrette. È molto utile anche per l'isolamento essendo che così potremo testare i DAO anche a database spento. Inoltre, in questo modo ci assicuriamo che la logica Java che abbiamo inserito nei DAO sia funzionante essendo che possiamo simulare alcune condizioni difficili da avere nel database.
Le servlet sono state testate anche esse utilizzando la tecnica White-Box. La scelta è dovuta alla velocità con il quale può essere fatto un White-Box testing e per poter irrobustire ogni singola riga delle servlet.

L'approccio ha permesso di gestire situazioni critiche come le sessioni nulle, parametri di richiesta mancanti o ruoli utenti non autorizzati.
I service sono stati testati in utilizzando la tecnica White-Box. La scelta è dovuta alla possibilità di isolare il testing e assicurarci una fault tolerance maggiore, inoltre grazie all'isolamento delle dipendenze siamo riusciti a verificare che eventuali errori nei test fossero dovuti al service e non a problemi del database. Abbiamo testato anche i DTO per scrupolo.

5.2Integration Testing

In questa fase abbiamo preferito utilizzare un approccio Black-box in particolare il category partition. Abbiamo seguito una strategia bottom up partendo dai DAO che sono la base del nostro progetto fino ad arrivare ai Service. La scelta di non includere le Servlet in questa fase è dettata dalla volontà di isolare e consolidare prima l'interazione tra logica applicativa e database MySQL reale. Essendo che le servlet fungono principalmente da intermediari di controllo la loro corretta integrazione è stata verificata nel System Testing dove il sistema è stato validato a mano tramite l'interfaccia utente.

I test di integrazione del PrenotazioneService sono stati progettati seguendo una metodologia **Black-Box** basata sul **Category Partitioning**. Questa scelta ha permesso di derivare i casi di test (Good e Bad Path) direttamente dai requisiti di business e dai vincoli di integrità del database. L'esecuzione dei test segue tuttavia un approccio **Gray-Box**, in quanto la verifica dei risultati (Oracolo) non si limita al valore di ritorno dei metodi, ma valida la persistenza dello stato direttamente sul layer di database attraverso i DAO.

5.3System Testing (Selenium)

Livello: Sistema completo end-to-end

Tecnica: Functional Testing

- Test dei flussi utente completi (UC)
- Interazione con browser (Chrome, Firefox)
- Verifica della UI response e comportamento

5.4Sospensione e Ripresa

La fase di testing può essere sospesa se si riscontra un difetto critico nel sistema per essere poi ripresa dopo aver risolto il problema riscontrato. I criteri di sospensione:

- Critico: Errore che impedisce il funzionamento di un UC (es. DB non connesso)
- Maggiore: Errore che causa risultati errati (es. mora non calcolata)
- Minore: Errore estetico o di validazione (es. messaggio di errore incompleto)

Solo i difetti Critico e Maggiore sospendono il testing.

5.5Materiali per il testing

Infrastruttura:

- Web Server: Apache Tomcat 9 in locale
- Client-web: Browser (Chrome, Firefox)
- Database: MySQL 8.0
- IDE: NetBeans o IntelliJ IDEA

Strumenti di Testing:

- JUnit
- Mockito
- MySQL Workbench - Gestione DB test
- Postman (opzionale) - API testing manuale

Dati di Test:

- Utenti di test per ogni ruolo (Cliente, Staff,)

6.Test Cases and Test frames

6.1 – Login

Parametro: Email	
Categorie	Scelte
Username nel database - EM	1. Non esiste nel database [property NotEM] 2. Username esistente [property EM]

Parametro: Password	
Categorie	Scelte
Password associata a username nel database - PA	1.Password associata a username nel database [property PA] 2.Password non associata a username nel database [property NotPA]

Test Frame

Codice	Combinazione	Esito
TC1	EM1.PA2	Visualizzazione pagina di autenticazione con messaggio "username o password non corretta"
TC2	EM2.PA1	Visualizzazione pagina utente registrato
TC3	EM2.PA2	Visualizzazione pagina di autenticazione con messaggio "username o password non corretta"

Test cases

Test case ID:	TC1
Precondizione:	
Non esiste alcun utente con l'email fornita.	
Flusso di eventi:	
L'utente compila il form:	

Input	Valore
Email	Non.esisto@fail.com
Password	anyPass

L’utente clicca sul pulsante Accedi

Oracolo
Il metodo restituisce null. Nessuna sessione viene creata e l'accesso è negato.

Test case ID:	TC3
Precondizione:	
Esiste un utente registrato nel database con email mario.rossi@integration.it. La password attualmente memorizzata (hashata) non corrisponde a quella fornita in input.	
Flusso di eventi:	

L’utente compila il form:

Input	Valore
Email	mario.rossi@integration.it
Password	PasswordErrata999

L’utente clicca sul pulsante Accedi

Oracolo
L'autenticazione fallisce poiché il componente PasswordHasher rileva la mancata corrispondenza tra il valore in chiaro fornito e l'hash nel database.

6.2 – Aggiungi Prenotazione

Parametro: numero partecipanti	
Categorie	Scelte
Numero partecipanti - NP	1.formato numero partecipanti errato [property Not NP] 2.formato numero partecipanti corretto [property NP]

Parametro: data	
Categorie	Scelte
Data nel database - DD	1. Data non disponibile [property NotDD] 2. Data disponibile [property DD]

Parametro: fascia oraria	
Categorie	Scelte
Fascia oraria nel database - FD	1. Fascia oraria non disponibile [property

	NotFD] 2. Fascia oraria disponibile [property FD]

Test Frame

Codice	Combinazione	Esito
TC4	NP2, DD1, FD2	Visualizzazione pagina di prenotazione con messaggio "Data non disponibile"
TC5	NP2, DD2, FD1	Visualizzazione pagina di prenotazione con messaggio "Fascia oraria non disponibile"
TC6	NP1, DD2, FD2	Visualizzazione pagina di prenotazione con messaggio "Formato numero partecipanti errato"
TC7	NP2, DD2, FD2	Prenotazione avvenuta con successo ed inoltro della richiesta

Test cases

Test case ID:	TC4
Precondizione:	
L'utente si trova sulla pagina di prenotazione bowling.	
Flusso di eventi:	
L'utente seleziona: Input: Valore idServizio: 1 data: 2025-12-25 L'utente clicca sul pulsante Conferma	
Oracolo	
Prenotazione fallita in quanto la data inserita "2025-12-25" non è disponibile.	

Test case ID:	TC5
Precondizione:	
L'utente si trova sulla pagina di prenotazione bowling.	
Flusso di eventi:	
L'utente compila il form:	

Input	Valore
idServizio	1
data	2025-12-25
fascia	12.00-13.00

L’utente clicca sul pulsante Conferma

Oracolo
Prenotazione fallita in quanto la fascia oraria “12.00-13.00” non è disponibile.

Test case ID:	TC6
Precondizione:	
L’utente si trova sulla pagina di prenotazione bowling.	
Flusso di eventi:	

L’utente compila il form:

Input	Valore
idServizio	1
data	2025-12-25
fascia	12.00-13.00
Numero partecipanti	-1

L’utente clicca sul pulsante Conferma

Oracolo
Prenotazione fallita in quanto il formato del numero dei partecipanti non è corretto

Test case ID:	TC7
Precondizione:	
L’utente si trova sulla pagina di prenotazione bowling.	
Flusso di eventi:	

L’utente compila il form:

Input	Valore
idServizio	1
data	2025-12-25
fascia	pomeriggio
partecipanti	2
Nome partecipante1	“Giovanni”
Nome partecipante2	“Andrea”

L’utente clicca sul pulsante Conferma

Oracolo
Procedura di prenotazione avvenuta con successo.

6.3-Registrazione

Parametro: Email	
Categorie	Scelte
Email nel database – EM	1. Email esiste nel database [property EM] 2.Email non esiste nel database [property NotEM]
Formato email – FE	1.Formato email corretto [property FM] 2.Formato email non corretto [property NotFM]

Parametro: Nome	
Categorie	Scelte
Formato nome – FN	1.Formato nome corretto [property FN] 2.Formato nome non corretto [property NotFN]

Parametro: Cognome	
Categorie	Scelte
Formato cognome – FC	1.Formato cognome corretto [property FC] 2.Formato cognome non corretto [property NotFC]

Parametro: Password	
Categorie	Scelte
Formato password- FP	1.Formato password corretto [property FP] 2.Formato password non corretto [property NotFP]
Lunghezza password- LP	1.Lunghezza password corretta [property LP] 2.Lunghezza password non corretta [property not LP]

Parametro: Conferma password	
Categorie	Scelte
Conferma password uguale a Password- CP	1.Le due password sono uguali [property CP] 2.Le due password non sono uguali [property NotCP]

Test Frame

Codice	Combinazione	Esito
TC9	EM1.FE1.FN1.FC1.FP1.LP1.CP1	Registrazione avvenuta con

		successo
TC10	EM2.FE1.FN1.FC1.FP1.LP1.CP1	Visualizzazione pagina di Registrazione con messaggio "email già presente"
TC11	EM1.FE2.FN1.FC1.FP1.LP1.CP1	Visualizzazione pagina di autenticazione con messaggio "formato email errato"
TC12	EM1.FE1.FN2.FC1.FP1.LP1.CP1	Visualizzazione pagina di autenticazione con messaggio "formato nome errato"
TC13	EM1.FE1.FN1.FC2.FP1.LP1.CP1	Visualizzazione pagina di autenticazione con messaggio "formato cognome errato"
TC14	EM1.FE1.FN1.FC1.FP2.LP1.CP1	Visualizzazione pagina di autenticazione con messaggio "formato password errato"
TC15	EM1.FE1.FN1.FC1.FP2.LP2.CP1	Visualizzazione pagina di autenticazione con messaggio "password troppo corta"
TC16	EM1.FE1.FN1.FC1.FP1.LP1.CP2	Visualizzazione pagina di autenticazione con messaggio "le due password devono combaciare"

Test cases

Test case ID:	TC9
Precondizione:	
L'email mario.rossi@integration.it non è associata ad alcun account esistente nelle tabelle Cliente o Staff	
Flusso di eventi:	
L'utente compila il form:	

Input	Valore
Nome	Mario
Cognome	Rossi
Email	mario.rossi@integration.it
Password	PasswordSicura123!
Conferma password	PasswordSicura123!

L'utente clicca sul pulsante Registrati
Oracolo

Registrazione avvenuta correttamente

Test case ID:	TC10
Precondizione:	
L'utente invia una richiesta di registrazione con un'email che non rispetta il formato standard (es. manca il dominio o l'estensione).	
Flusso di eventi:	
L’utente compila il form:	

Input	Valore
Nome	Francesco
Cognome	Bianchi
Email	f.bia2e2e2e
Password	Fbianchi12!
Conferma password	Fbianchi12!

L’utente clicca sul pulsante Registrati
Oracolo
Registrazione fallita, formato email non rispettato

Test case ID:	TC11
Precondizione:	
L'email mario.rossi@integration.it è già presente nel database	
Flusso di eventi:	
L’utente compila il form:	

Input	Valore
Nome	Secondo
Cognome	User
Email	mario.rossi@integration.it
Password	pass2
Conferma password	pass2

L’utente clicca sul pulsante Registrati
Oracolo
Registrazione fallita, email già presente nel database

Test case ID:	TC12
Precondizione:	
La mail f.bianchi@mail.com non è registrata.	
Flusso di eventi:	
L’utente compila il form:	

Input	Valore

Nome	Fr!fd!!!!!!
Cognome	Bianchi
Email	f.bianchi@mail.com
Password	Fbianchi12!
Conferma password	Fbianchi12!

L'utente clicca sul pulsante Registrati

Oracolo
Registrazione fallita, formato nome errato

Test case ID:	TC13
Precondizione:	
La mail f.bianchi@mail.com non è registrata	
Flusso di eventi:	

L'utente compila il form:

Input	Valore
Nome	Francesco
Cognome	B!!!!!!!.....
Email	f.bianchi@mail.com
Password	Fbianchi12!
Conferma password	Fbianchi12!

L'utente clicca sul pulsante Registrati

Oracolo
Registrazione fallita, formato cognome errato

Test case ID:	TC14
Precondizione:	
La mail f.bianchi@mail.com non è registrata	
Flusso di eventi:	

L'utente compila il form:

Input	Valore
Nome	Francesco
Cognome	Bianchi
Email	f.bianchi@mail.com
Password	adwsad
Conferma password	adwsad

L'utente clicca sul pulsante Registrati

Oracolo
Registrazione fallita, formato password errato

Test case ID:	TC15
Precondizione:	
La mail f.bianchi@mail.com non è registrata	
Flusso di eventi:	
L'utente compila il form: Input: Valore Nome: Francesco Cognome: Bianchi Email: f.bianchi@mail.com Password: Fa12! Conferma password: Fa12! L'utente clicca sul pulsante Registrati	
Oracolo	
Registrazione fallita, password troppo corta	

Test case ID:	TC16
Precondizione:	
La mail f.bianchi@mail.com non è registrata	
Flusso di eventi:	
L'utente compila il form: Input: Valore Nome: Francesco Cognome: Bianchi Email: f.bianchi@mail.com Password: Fbianchi12! Conferma password: Fbianchi13! L'utente clicca sul pulsante Registrati	
Oracolo	
Registrazione fallita, le due password non corrispondono	

6.4-Rifiuta Prenotazione

Parametro: ID Prenotazione	
Categorie	Scelte
Esistenza nel database - EP	1. Non esiste nel database [property NotEP] 2. Esiste nel database [property EP]

Parametro: Motivazione Rifiuto	
Categorie	Scelte
Lunghezza Motivazione - LM	1. Lunghezza = 0 – campo vuoto [property

	invalidLMValue] 2. Lunghezza >= 1 [property validLMValue]

Parametro: Stato Prenotazione	
Categorie	Scelte
Stato prenotazione - SP	1. Stato = "In attesa" [property validSPValue] 2. Stato != "In attesa" [property invalidSPValue]

Test Frame

Codice	Combinazione	Esito
TC20	EP2.LM2.SP1	Visualizzazione pagina di prenotazione con messaggio "Prenotazione rifiutata con successo"
TC21	EP2.LM1.SP2	Visualizzazione pagina di prenotazione con messaggio "Motivazione: campo obbligatorio"

Test Cases

Test case ID:	TC20
Precondizione:	
Lo staff si trova sulla pagina di gestione prenotazioni e una prenotazione con stato "In attesa" è selezionata.	
Flusso di eventi:	
Lo staff compila il form di rifiuto: Input: idPrenotazione = 5; motivazione = "Pista/tavolo danneggiata/o" Lo staff clicca sul pulsante "Rifiuta"	
Oracolo	
Prenotazione rifiutata con successo. Stato cambia a "Rifiutata", motivazione salvata nel DB, cliente riceve aggiornamento di rifiuto, risorsa liberata.	

Test case ID:	TC21
Precondizione:	

Lo staff si trova sulla pagina di gestione prenotazioni.

Flusso di eventi:

Lo staff tenta di rifiutare una prenotazione senza inserire motivazione:

Input	Valore
idPrenotazione	5
motivazione	(vuoto)

Lo staff clicca sul pulsante "Rifiuta"

Oracolo

Errore di validazione: "Motivazione: campo obbligatorio". Form ripristinato, prenotazione non modificata.