Test Summary Report

EasyCoach

Cronologia delle revisioni

Data	Versione	Descrizione	Autori
01/02/2025	0.1	Prima stesura del documento	NV, SF, FD, RF
02/02/2025	0.2	Aggiornamento del report dei test	NV
05/02/2025	1.0	Revisione finale	NV, SF, FD, RF

Membri del Team

Nome	Ruolo del progetto	Acronimo	Informazioni di Contratto
Nello Valentino	Membro del team	NV	n.valentino6@studenti.unisa.it
Simone Fausto	Membro del team	SF	s.fausto5@studenti.unisa.it
Francesco D'Arco	Membro del team	FD	f.darco6@studenti.unisa.it
Roberto Fiorenza	Membro del team	RF	r.fiorenza3@studenti.unisa.it

1. Introduzione

EasyCoach è una piattaforma progettata per rendere il mentoring accessibile, sicuro e organizzato, ottimizzando l'interazione tra mentor e mentee.

L'obiettivo del **Test Summary Report** è fornire un riepilogo delle attività di testing condotte sulla piattaforma, evidenziando i risultati ottenuti, la copertura del codice e gli eventuali problemi riscontrati durante i test.

Le attività di testing si sono concentrate sui seguenti sottosistemi:

- **Autenticazione**
- **Sessione**
- **Prenotazione**

Questo documento riassume l'esecuzione dei **test di unità**, **integrazione** e **sistema**, basandosi sui risultati ottenuti durante la fase di validazione della piattaforma.

2. Relazione con altri documenti

Il **Test Summary Report** è strettamente collegato ai seguenti documenti:

- **Test Plan (TP):** descrive l'approccio e la strategia di testing adottata per il progetto EasyCoach
- **Test Case Specification (TCS):** contiene i dettagli di ogni test case eseguito, con input, output atteso e oracoli di validazione
- **Test Incident Report (TIR):** riassume eventuali problemi critici riscontrati durante l'esecuzione dei test, con relative risoluzioni

3. Testing unitario e di integrazione

Durante lo sviluppo, i test unitari e di integrazione sono stati scritti e organizzati nelle classi corrispondenti ai Service/Servlet da testare.

Ogni membro del team, prima di eseguire un **push**, ha verificato che tutti i test relativi alle classi modificate superassero il controllo con successo. Dopo il **push** nel repository **GitHub,** il sistema **CI/CD** di **GitHub Actions** ha eseguito automaticamente i test, impedendo la fusione del codice in caso di fallimenti

Gli strumenti utilizzati sono:

- **JUnit:** per l'esecuzione dei test unitari
- **Mockito:** per il mocking delle dipendenze nei test di integrazione
- **JaCoCo:** per la misurazione della coverage nel codice

Di seguito vengono riportati i risultati della coverage dei test

Line Coverage	Branch Coverage
84%	77%

Per maggiori dettagli si lascia il link al report di JaCoCo:
https://github.com/Nellow04/EasyCoach-v.1.0

4. Testing di Sistema

I test di sistema sono stati eseguiti in un ambiente di testing simile a quello di produzione. Sono stati utilizzati **scenari reali** per validare il corretto funzionamento della piattaforma.

Questa fase di testing è stata **eseguita manualmente** dal team di sviluppo, simulando scenari d’uso reali e verificando il corretto funzionamento delle funzionalità chiave