Università degli Studi di Salerno
Corso di Ingegneria del Software

SpeedScale
Test Plan
Versione 0.1

Data: 15/12/2024

Coordinatore del progetto:

Nome	Matricola

Partecipanti:

Nome	Matricola
Sepe Gennaro	0512116971
La Marca Antonio	0512117826

Scritto da:	Sepe Gennaro

Revision History

Data	Versione	Descrizione	Autore
15/12/2024	0.1	Creazione e scrittura del documento	Sepe Gennaro

Indice

1. Introduzione

Questo documento rappresenta il piano di test per il sistema SpeedScale, un'applicazione progettata per fornire una gestione efficiente di utenti, prodotti, ordini e carrelli. Il piano di test è stato concepito per garantire che il sistema soddisfi i requisiti funzionali e non funzionali definiti nei documenti di specifica. L'obiettivo primario di questo piano è fornire una guida chiara e strutturata per la pianificazione e l'esecuzione dei test. Attraverso un insieme di strategie, criteri e casi di test ben definiti, si intende identificare e correggere eventuali difetti, assicurando che il sistema funzioni in modo affidabile e conforme alle aspettative. I test seguiranno un approccio strutturato che includerà verifiche a livello unitario, di integrazione e di sistema. Inoltre, saranno definiti criteri per l'accettazione e il rifiuto dei test, specificando in che modo i risultati influiranno sul processo di sviluppo complessivo. La qualità del sistema sarà garantita attraverso una copertura completa dei test, che verificheranno ogni aspetto critico dell'applicazione.

2. Relazione con altri documenti

La progettazione di SpeedScale si è avvalsa di risorse teoriche e pratiche che hanno guidato la definizione dei requisiti, la progettazione del sistema e la stesura dei relativi documenti. Come riferimento principale, è stato adottato il manuale "Object Oriented Software Engineering Using UML, Patterns, and Java™" di Bernd Bruegge e Allen H. Dutoit. Questo testo ha fornito le linee guida per l'utilizzo di UML, la scrittura di documenti tecnici e l'applicazione di pattern di progettazione, risultando fondamentale per garantire una progettazione chiara e coerente. Il processo di sviluppo è iniziato con la creazione di un Problem Statement, che ha consentito di delineare con precisione gli obiettivi del sistema e le esigenze del mercato di riferimento. Successivamente, è stato realizzato un Requirements Analysis Document (RAD), che ha permesso di ottenere una visione iniziale e strutturata delle funzionalità e delle specifiche del sistema, servendo da base per le fasi successive. La progettazione ha tratto ulteriore valore dagli insegnamenti del corso universitario "Ingegneria del Software", tenuto dal professore Andrea De Lucia, che ha fornito una solida metodologia per la gestione strutturata del ciclo di vita dello sviluppo software. Questi insegnamenti hanno contribuito a definire approcci di qualità sia nella progettazione del sistema che nel suo testing, garantendo una piattaforma robusta ed affidabile. Un altro documento chiave per il test è l'Object Design Document (ODD). Questo documento contiene una descrizione dettagliata delle classi, delle loro responsabilità e delle loro relazioni, offrendo una guida essenziale per individuare i casi di test e per stabilire la copertura dei test rispetto ai requisiti. L'ODD rappresenta il punto di riferimento per il collegamento tra i test unitari e le funzionalità implementate. Infine, tutti i test descritti in questo piano sono strettamente correlati ai requisiti funzionali e non funzionali delineati nei documenti di specifica. Per mantenere una tracciabilità rigorosa, è stato adottato uno schema di denominazione standard che permette di stabilire con chiarezza il legame tra i requisiti e i relativi test. Questo garantisce una verifica completa e sistematica del sistema SpeedScale, assicurando che ogni requisito sia coperto da un caso di test appropriato.

3. Panoramica del sistema

Il sistema SpeedScale è stato progettato seguendo un'architettura multilivello, strutturata in modo da garantire modularità, separazione delle responsabilità e facilità di manutenzione. L'architettura si articola in tre livelli principali: il livello di presentazione, il livello applicativo e il livello dei dati. Il livello di presentazione rappresenta l'interfaccia utente del sistema, attraverso cui gli utenti interagiscono direttamente con la piattaforma. Questo livello è implementato utilizzando le pagine JSF, basate su Java EE 7, e integra il framework Bootstrap per offrire un design reattivo e un'esperienza utente moderna. Le funzionalità accessibili a questo livello comprendono tutte le operazioni principali, dalla gestione del profilo utente fino all'esplorazione del catalogo e alla gestione del carrello. Il livello applicativo costituisce il nucleo logico del sistema ed è responsabile dell'elaborazione delle richieste provenienti dall'interfaccia utente. Questo livello è composto da servizi modulari progettati per assolvere a specifiche funzionalità. Ad esempio, il servizio di autenticazione gestisce l'accesso degli utenti, le loro credenziali e le sessioni; il servizio di profilo si occupa delle informazioni personali, degli indirizzi di spedizione, dei metodi di pagamento e dello storico degli ordini; il servizio catalogo permette la gestione dei prodotti, mentre il servizio carrello supporta tutte le operazioni relative agli acquisti, come l'aggiunta di prodotti e la loro conversione in ordine. Ogni servizio è stato sviluppato come una classe indipendente, progettata per garantire un'interfaccia chiara e un'interazione efficace con gli altri componenti del sistema. Il livello dei dati è invece dedicato alla gestione della persistenza, affidata a un database relazionale, come MySQL, utilizzando JPA (Java Persistence API) per garantire un accesso efficiente e sicuro alle informazioni. Le entità del dominio, tra cui Utente, Prodotto e Ordine, sono mappate direttamente alle tabelle del database, assicurando una chiara corrispondenza tra i dati logici e quelli fisici. Per garantire l'affidabilità del sistema, i test unitari si concentrano sui moduli fondamentali che compongono i principali sottosistemi, come l'autenticazione, la gestione del profilo, il catalogo e il carrello. Ogni modulo verrà analizzato in modo dettagliato per assicurarsi che le operazioni offerte siano conformi ai requisiti. Successivamente, i test di integrazione verificheranno il corretto funzionamento del sistema nel suo complesso, esaminando il flusso di interazione tra i livelli. La granularità dei componenti e le loro dipendenze saranno analizzate utilizzando un diagramma UML delle classi, che faciliterà la pianificazione delle attività di verifica. Questa panoramica fornisce il contesto strutturale del sistema, delineando gli aspetti fondamentali della sua progettazione e le basi per la definizione di una strategia di test efficace.

4. Caratteristiche da testare/non testare

Il piano di test per il sistema SpeedScale si concentra sulla verifica delle funzionalità essenziali e delle caratteristiche critiche che contribuiscono al raggiungimento degli obiettivi progettuali. L'attenzione è posta su tutte le operazioni principali che determinano l'efficienza e l'affidabilità del sistema, garantendo una copertura significativa dei requisiti funzionali e non funzionali specificati nei documenti di progettazione. Tra le caratteristiche principali che saranno sottoposte a test vi sono tutte le operazioni relative all'autenticazione, tra cui il login, la gestione delle credenziali e il mantenimento della sessione utente. La gestione del profilo è un'altra area centrale, con particolare attenzione alla modifica dei dati personali, agli indirizzi di spedizione, ai metodi di pagamento e allo storico degli ordini. Il catalogo prodotti sarà testato per garantire la corretta visualizzazione, modifica e gestione dei prodotti, mentre il carrello sarà sottoposto a verifiche per assicurare un'esperienza di acquisto fluida, compresa la gestione degli articoli selezionati e la conversione in ordine. Tutte queste caratteristiche saranno analizzate sia dal punto di vista della logica applicativa che per il loro comportamento nell'interfaccia utente. Dal lato non funzionale, si presterà particolare attenzione alla sicurezza, soprattutto nella protezione delle credenziali e nella gestione delle sessioni, oltre che alla performance, verificando che le operazioni principali siano eseguite in tempi accettabili anche in condizioni di carico elevato. Non saranno invece oggetto di test le funzionalità che non fanno parte del core attuale del sistema, come caratteristiche sperimentali o pianificate per versioni future. Analogamente, non verranno testate componenti di terze parti già certificate e ampiamente utilizzate, come il server di database, le librerie di Bootstrap e i framework standard di Java EE 7. Questa scelta è giustificata dall'affidabilità consolidata di tali componenti e dalla necessità di concentrare le risorse di test sugli sviluppi specifici della piattaforma SpeedScale. La distinzione tra le caratteristiche da testare e quelle escluse permette di ottimizzare il processo di verifica, evitando la dispersione delle risorse e focalizzando l'attenzione sulle funzionalità che hanno un impatto diretto sull'esperienza dell'utente e sulla qualità complessiva del sistema.

5. Criteri di accettazione/rifiuto

La definizione dei criteri di accettazione e rifiuto rappresenta un aspetto cruciale per garantire la qualità del sistema SpeedScale. Questi criteri forniscono un quadro di riferimento chiaro per valutare l'esito dei test, stabilendo le condizioni necessarie per considerare una funzionalità accettabile o per identificare le aree che richiedono ulteriori interventi. Per essere accettato, ogni test deve essere completato con successo, senza errori o criticità evidenti. Le operazioni principali devono dimostrare piena conformità ai requisiti funzionali e non funzionali descritti nel Requirements Analysis Document (RAD) e nel System Design Document (SDD). L'esecuzione dei test deve coprire integralmente i requisiti classificati come critici, mentre i requisiti secondari devono essere verificati in misura non inferiore al 90%. La conformità ai criteri di performance, sicurezza e usabilità è inoltre indispensabile per garantire un'esperienza utente ottimale e una robustezza adeguata. Un caso di test viene considerato rifiutato se si verificano errori bloccanti, ovvero problemi che impediscono l'esecuzione delle operazioni principali. Lo stesso vale per i difetti che causano deviazioni significative rispetto ai requisiti definiti. Anche i malfunzionamenti ricorrenti durante i test di regressione, se non risolti, rappresentano motivo di rifiuto. Questi errori sono documentati in un rapporto dettagliato e vengono classificati in base alla loro gravità per agevolare il processo di correzione e successiva verifica. I criteri stabiliti non solo garantiscono la qualità del sistema, ma permettono anche di definire obiettivi chiari per il completamento dei test.

6. Approccio

L’approccio adottato per il processo di test del sistema SpeedScale si basa su una strategia metodica e strutturata, che combina tecniche di testing manuale e automatico per garantire una copertura completa delle funzionalità e delle interazioni tra i componenti. L’obiettivo principale è verificare la conformità del sistema ai requisiti specificati, assicurando che ogni componente soddisfi le aspettative in termini di correttezza, efficienza e robustezza. Il piano di test si articola in diverse fasi, ciascuna delle quali affronta specifici livelli di granularità. Il testing unitario si concentra sui singoli moduli, isolandoli per verificare la correttezza delle loro funzionalità indipendentemente dagli altri componenti. Questo approccio consente di identificare e correggere rapidamente eventuali errori a livello di singola unità, minimizzando il rischio di propagazione di difetti. Per questa fase, si utilizzano strumenti come **JUnit** per l’automazione dei test e per garantire una copertura ottimale del codice. Il testing di integrazione si occupa di verificare le interazioni tra i diversi moduli. L’approccio seguito è principalmente di tipo bottom-up, in cui i moduli di livello più basso vengono testati per primi e, successivamente, vengono combinati con i moduli di livello superiore. Questa strategia permette di rilevare eventuali incongruenze o malfunzionamenti nei punti di interfaccia tra i componenti. Le dipendenze tra i moduli sono state chiaramente rappresentate tramite diagrammi UML durante la fase di progettazione, facilitando l’individuazione delle aree critiche da testare. Il testing di sistema rappresenta la fase finale ed è progettato per verificare il comportamento del sistema nel suo complesso, simulando scenari realistici di utilizzo. In questa fase vengono eseguiti test funzionali, di sicurezza, di performance e di usabilità per garantire che il sistema sia pronto per l’implementazione nel contesto operativo previsto. Il testing viene condotto utilizzando strumenti come **Selenium** per l’automazione dei test. La scelta di questa strategia di test è motivata dalla complessità del sistema e dalla necessità di garantire una qualità elevata in tutte le sue componenti. L’approccio adottato permette di gestire in modo efficace le dipendenze tra i moduli, assicurando che il sistema funzioni correttamente sia a livello di singoli componenti che nel loro insieme.

7. Sospensione e ripresa

Nel contesto del piano di test per il sistema SpeedScale, la sospensione dei test può diventare necessaria in situazioni particolari che ostacolano il proseguimento regolare delle attività di testing. Tali situazioni potrebbero includere, ad esempio, la rilevazione di difetti critici che compromettono il funzionamento di funzionalità essenziali, come quelle relative alla gestione degli utenti, dei prodotti o degli ordini, oppure problemi tecnici legati a malfunzionamenti hardware o software che impediscono l'esecuzione dei test stessi. Altri fattori che giustificano una sospensione possono essere la necessità di una revisione delle specifiche o la presenza di incertezze che richiedono una rivalutazione dei requisiti da parte del team di sviluppo. In caso di sospensione, è fondamentale stabilire con chiarezza le condizioni necessarie per la ripresa delle attività di test. La ripresa avverrà non appena i problemi che hanno causato l'interruzione siano stati risolti. Ogni test interrotto dovrà essere rieseguito interamente per verificare che le correzioni non abbiano introdotto nuovi difetti o influenzato negativamente altre aree del sistema. La ripresa include anche un riesame delle risorse e degli strumenti necessari, per garantire che siano disponibili e funzionanti, e l'aggiornamento della documentazione relativa ai test, così da mantenere una traccia chiara delle attività svolte. È importante che il processo di sospensione e ripresa venga gestito in modo da non compromettere l'integrità e la qualità complessiva del sistema. Ogni fase del testing, sia in caso di interruzione che di ripresa, dovrà essere accuratamente documentata, includendo una registrazione dei problemi riscontrati, delle azioni correttive intraprese e dei risultati ottenuti nella fase di verifica successiva. In questo modo, si garantisce che i test possano continuare in modo efficace, con la massima attenzione alla qualità e all'affidabilità del sistema.

8. Materiali di prova (requisiti hardware/software)

Per l'esecuzione dei test sul sistema, è sufficiente l'uso di un comune computer fisso o portatile che soddisfi i requisiti hardware e software minimi. In particolare, l'hardware necessario comprende un dispositivo con almeno 8 GB di RAM e una CPU moderna (ad esempio, Intel i5 o equivalente) per garantire che i test vengano eseguiti in modo fluido, anche in presenza di carichi elevati. Per quanto riguarda i software, il sistema di test può essere eseguito su qualsiasi macchina che supporti i requisiti di esecuzione di Java EE 7, poiché il sistema SpeedScale è sviluppato in questo ambiente. Pertanto, sono necessari i seguenti software:

1. **Java Development Kit (JDK)**: Per eseguire i test sui componenti sviluppati in Java, è necessaria l'installazione di una versione compatibile con Java EE 7 (ad esempio JDK 8 o superiore);
2. **Database Relazionale (MySQL)**: SpeedScale utilizza un database relazionale, come MySQL, per gestire la persistenza dei dati. È necessario un ambiente di test che consenta l'esecuzione e la gestione di un database di test, configurato con le entità del dominio;
3. **Strumenti di automazione dei test**: Per i test automatizzati, è consigliato l'uso di strumenti come **JUnit** per i test unitari e **Selenium** per i test di sistema e di integrazione, particolarmente per l'interazione con l'interfaccia utente. Questi strumenti richiedono l'installazione di Java e di browser web compatibili;
4. **Sistema operativo**: Il sistema di test può essere eseguito su sistemi operativi come Windows, macOS o Linux, purché compatibili con le versioni di software sopra indicate.

L'ambiente di test non richiede configurazioni hardware particolarmente avanzate. Un semplice computer, fisso o portatile, che rispetti i requisiti sopra indicati è sufficiente per eseguire tutti i test, garantendo che le verifiche vengano svolte in modo efficiente e senza necessità di risorse hardware complesse.

9. Casi di test

I casi di test sono uno strumento fondamentale per garantire che tutte le funzionalità del sistema siano implementate correttamente e che il software rispetti i requisiti funzionali e non funzionali previsti. Ogni caso di test è stato progettato tenendo conto delle specifiche del sistema e delle esigenze di business, al fine di coprire i principali flussi operativi. I test coprono una serie di operazioni cruciali, tra cui il login, la gestione del carrello, l'elaborazione degli ordini e la ricerca dei prodotti. Inoltre, per ogni caso di test sono definiti i relativi requisiti di ingresso, i risultati attesi, le dipendenze tra i test e le configurazioni ambientali necessarie per eseguire i test. Il capitolo è suddiviso in singole tabelle per ogni test, ognuna delle quali include informazioni dettagliate riguardo all'identificativo del test, agli input, agli output e alle eventuali dipendenze. Questa struttura facilita una lettura chiara e organizzata dei casi di test, rendendo semplice la gestione e l'esecuzione delle attività di verifica. L'esecuzione corretta di questi test garantirà che il sistema sia pronto per la fase di rilascio e per l'uso da parte degli utenti finali.

Test 1: Verifica del Login Utente

Identificatore	***TEST_LOGIN_01***
Descrizione	Verifica che l'utente possa effettuare il login con credenziali corrette.
Test items	Autenticazione
Specifiche di ingresso	E-mail: mrossi@email.it Password: "password123"
Specifiche di uscita	L'utente è autenticato e reindirizzato alla home page.
Dipendenze	-
Esigenze ambientali	-
Requisiti procedurali speciali	-

Test 2: Verifica Errore di Login con Credenziali Errate

Identificatore	***TEST_LOGIN_02***
Descrizione	Verifica che il sistema mostri un errore di login quando vengono inserite credenziali errate.
Test items	Autenticazione
Specifiche di ingresso	E-mail: mrossi@email.it Password: "tsw"
Specifiche di uscita	Il sistema visualizza un errore di credenziali non valide.
Dipendenze	TEST_LOGIN_01
Esigenze ambientali	-
Requisiti procedurali speciali	-

Test 3: Aggiunta di un Prodotto al Carrello

Identificatore	***TEST_CARRELLO_01***
Descrizione	Verifica che un prodotto possa essere aggiunto correttamente al carrello.
Test items	Carrello
Specifiche di ingresso	Prodotto da aggiungere: "Prodotto A"
Specifiche di uscita	Il prodotto "Prodotto A" viene aggiunto al carrello, e il totale del carrello si aggiorna.
Dipendenze	-
Esigenze ambientali	-
Requisiti procedurali speciali	-

Test 4: Rimozione di un Prodotto dal Carrello

Identificatore	***TEST_CARRELLO_02***
Descrizione	Verifica che un prodotto possa essere rimosso correttamente dal carrello.
Test items	Carrello
Specifiche di ingresso	Prodotto da rimuovere: "Prodotto A"
Specifiche di uscita	Il prodotto "Prodotto A" viene rimosso dal carrello, e il totale si aggiorna.
Dipendenze	-
Esigenze ambientali	-
Requisiti procedurali speciali	-

Test 5: Creazione di un Ordine

Identificatore	***TEST_ORDINE_01***
Descrizione	Verifica che l'ordine venga creato correttamente a partire dai prodotti nel carrello.
Test items	Carrello
Specifiche di ingresso	Prodotti nel carrello: "Prodotto A", "Prodotto B" Indirizzo di spedizione: "Via Roma 10, Milano" Metodo di pagamento: “XXXX, YY/YY, ZZZ”
Specifiche di uscita	L'ordine viene creato correttamente e l'utente riceve una conferma di acquisto.
Dipendenze	TEST_CARRELLO_01, TEST_LOGIN_01
Esigenze ambientali	-
Requisiti procedurali speciali	L'utente deve essere loggato e avere prodotti nel carrello.

Test 6: Verifica del Carrello VUOTO

Identificatore	***TEST_CARRELLO_03***
Descrizione	Verifica che il sistema segnali correttamente un carrello vuoto.
Test items	Carrello
Specifiche di ingresso	Nessun prodotto nel carrello
Specifiche di uscita	Il sistema visualizza il messaggio "Il carrello è vuoto" e il pulsante per procedere all'acquisto non è cliccabile.
Dipendenze	-
Esigenze ambientali	-
Requisiti procedurali speciali	-

Test 7: Verifica dell'Accesso alla Pagina del Profilo

Identificatore	***TEST_PROFILO_01***
Descrizione	Verifica che l'utente loggato possa accedere correttamente alla pagina del suo profilo.
Test items	GestioneProfilo
Specifiche di ingresso	L'utente è loggato
Specifiche di uscita	Il sistema reindirizza l'utente alla pagina del profilo.
Dipendenze	TEST_LOGIN_01
Esigenze ambientali	-
Requisiti procedurali speciali	L'utente deve essere loggato.

10. Programma di test

I tester devono avere una conoscenza approfondita degli strumenti di testing, come JUnit e Selenium, e una comprensione completa del sistema, in particolare della piattaforma Java EE su cui il sistema è basato. Nel corso del programma di test, si possono manifestare alcuni rischi che potrebbero compromettere l'efficacia del processo. Un rischio principale riguarda la complessità delle interazioni tra i moduli del sistema, che potrebbero non funzionare correttamente durante la fase di integrazione. È essenziale, pertanto, eseguire test di integrazione accurati per verificare che tutti i componenti del sistema comunichino in modo efficace. Un altro rischio significativo è legato alla performance del sistema, in particolare alla sua capacità di gestire carichi elevati. Test di stress e performance saranno quindi necessari per simulare un uso intensivo del sistema e verificarne la reattività. Inoltre, potrebbero sorgere problemi di compatibilità con componenti di terze parti, come il database MySQL o altre librerie di framework. In questi casi, è fondamentale monitorare attentamente la stabilità e la compatibilità del sistema. Infine, modifiche non previste al sistema potrebbero influenzare la validità dei test già eseguiti, rendendo necessario l'uso di test di regressione per verificare che le modifiche non abbiano introdotto nuovi difetti. Il programma di test è suddiviso in diverse fasi. Inizialmente, sarà necessario un periodo di preparazione per configurare gli ambienti di test. Successivamente, si procederà con il test unitario, che si concentrerà sulla verifica dei singoli moduli del sistema. Seguiranno poi i test di sistema, che includeranno la verifica delle funzionalità, della sicurezza, delle performance e dell'usabilità. Infine, si eseguiranno test di regressione e conclusione, per garantire che le modifiche apportate durante lo sviluppo non abbiano introdotto nuovi problemi.