Università degli Studi di Salerno

Corso di Ingegneria del Software

GINusJuice
Test Plan
Versione 1.0

Data: 14/12/2024

Coordinatore del progetto:

Nome	Matricola
D’Avino Salvatore	0512118435
Vitulano Antonio	0512116776

Partecipanti:

Nome	Matricola
ANTONIO VITULANO	0512116776
SALVATORE D’AVINO	0512118435

Scritto da:	ANTONIO VITULANO, SALVATORE D’AVINO

Revision History

Data	Versione	Descrizione	Autore
14/12/2024	1.0	Introduzione	Vitulano
14/12/2024	1.0	Rapporto con altri documenti	Vitulano
14/12/2024	1.0	Panoramica del sistema	Vitulano
14/12/2024	1.0	Caratteristiche da testare e non da testare	Vitulano
14/12/2024	1.0	Criteri successo e di fail	Vitulano
14/12/2024	1.0	Approccio	D’avino
14/12/2024	1.0	Sospensione e di ripresa	D’avino
14/12/2024	1.0	Testing e materials	D’Avino
14/12/2024	1.0	Testing Cases	D’Avino
14/12/2024	1.0	Testing Schedule	D’Avino
15/12/2024	1.0	Riunione per approvare	D’avino Vitulano

Sommario

INTRODUZIONE

Definizioni

• Branch Coverage: tecnica adoperata durante la fase di testing, che prevede l'esecuzione di tutti i rami del programma almeno una volta durante la fase di testing.
• Failure: mancata o scorretta azione di un determinato servizio atteso.
• Fault: causa che ha generato una failure.
• Model View Control: è un metodo architetturale che prevede la divisione dell'applicazione di tre parti. Tale divisione viene effettuata per separate la rappresentazione delle informazioni interne del sistema dal meccanismo in cui le informazioni sono presentate all'utente

1.1.2 Acronimi e Abbreviazioni

• RAD: Requirement Analysis Document.
• SDD: System Design Document.
• ODD: Object Design Document.
• TP: Test plan.
• MVC: Model View Controller.
• DB: Database.
• API: Application Programming Interface.
• GUI: Graphical User Interface.

1.1.3 Riferimenti

• Object-Oriented Software Engineering (Using UML, Patterns, and Java) Third Edition.
Autori: Bernd Bruegge & Allen H. Dutoit
• PMBOK Guide and Software Extention to the PMBOK Guide, Fifth Ed, Project Management Institute, 2013;
• Documentazione del progetto:
RAD,SDD,ODD

Rapporto con altri documenti

Questo documento è correlato a tutti i documenti prodotti fino al rilascio del sistema, quindi verranno modificati in futuro dopo il rilascio di altri documenti non ancora prodotti. I test case sono basati sulle funzionalità del sistema, individuate e raccolte nel RAD.e nei sottosistemi dell'SDD

Relazione con il documento di raccolta ed analisi dei requisiti (RAD)

La relazione tra questo documento e il RAD riguarda la fase di raccolta dei requisiti funzionali e non funzionali, infatti in questo documento oltre a essere presenti i requisiti si possono trovare anche scenari, casi d'uso, diagrammi di sequenza e mockups.

Relazione con il System Design Document (SDD)

In questo documento è presente l'architettura del sistema (MVC), la struttura dei dati e i servizi offerti da ogni sottosistema.

Relazione con l'Object Design Document (ODD)

Nell'ODD sono contenuti i package e le classi del sistema.

Panoramica del sistema

Come definito nel System Design Document, il sistema avrà una struttura MVC (Model View Controller).La componente fondamentale di questo approccio è il controller che si occuperà della logica esecutiva diogni sottosistema, nel model verranno indicate le entità persistente del DB, infine nella view verranno mostrate le interfacce utente.

Caratteristiche da testare/non da testare

Per trovare le caratteristiche da testare ci concentriamo sui requisiti funzionali del sistema presenti nel rad,e nelle varie funzionalità dei sottosistemi individuati nel SDD,in generale daremo molta più importanza alle funzionalità di sicurezza e che noi (Vitulano Antonio,D'Avino Salvatore) riteniamo importanti per il software che stiamo progettando come ad esempio il login e l'aggiunta al carrello ,le caratteristiche non andremo a testare sono quelle funzionalità ancora non implmentate e quei requisiti non funzionali che risultano difficili da testare per due studenti universitari.

Criteri di successo/fail

Dopo aver individuato tutti i dati di input del sistema, quest'ultimi verranno raggruppati insieme in base alle caratteristiche in comune. Questa tecnica ci servirà per poter diminuire il numero di test da dover effettuare. Diremo che la fase di test ha successo se viene individuata effettivamente una failure all'interno del sistema, cioè l'output atteso per quel determinato input non è lo stesso previsto dall'oracolo. Successivamente la failure sarà analizzata e si passerà eventualmente alla sua correzione e verranno eseguiti nuovamente tutti i test necessari per verificare l'impatto che la modifica ha avuto sull'intero sistema. Diremo invece che il testing fallirà se l'output mostrato dal sistema coincide con quello previsto dall'oracolo.

Approccio

Testing di unità

Lo scopo del testing di unità è quello di testare ogni singola funzione presente all'interno del sistema. Ogni funzione rappresenta quella che viene definita come "unità".

Approccio scelto

L'approccio scelto è di tipo white-box, con tale tipo di approccio quindi andremo a testare il sistema conoscendone il funzionamento interno.

Testing di integrazione

Lo scopo del testing di integrazione è quello di "mettere insieme" le componenti testate precedentemente tramite il test di unità per vedere come funzionano una volta integrate tra loro.

Approccio scelto

Per effettuare il testing di integrazione si è scelto di adoperare un approccio bottom-up. Il vantaggio fondamentale di questa tipologia di testing è quello della riusabilità del codice. Questo tipo di approccio prevede però la costruzione driver per simulare l'ambiente chiamante. È stato scelto quindi questo tipo di approccio perché sembra quello più intuitivo e semplice.

Testing di sistema

Lo scopo del testing di sistema è quello di verificare che i requisiti richiesti dal cliente siano stati effettivamente rispettati e che il cliente risulti soddisfatto del sistema stesso. In questo tipo di testing si vanno a verificare le funzionalità utilizzate più spesso da parte del cliente e quelle che risultano più "critiche".

Approccio scelto

Useremo dei file,andremo a utilizzare il sistema come dei veri e propri clienti .

Sospensione e ripresa

Criteri di sospensione

La fase di testing verrà sospesa nel momento in cui saranno raggiunti i risultati previsti in accordo con quello che è il budget a disposizione o dovremo andare all'esame

Criteri di ripresa

Tutte le attività di testing riprenderanno nel momento in cui verranno effettuate modifiche all'interno del sistema.

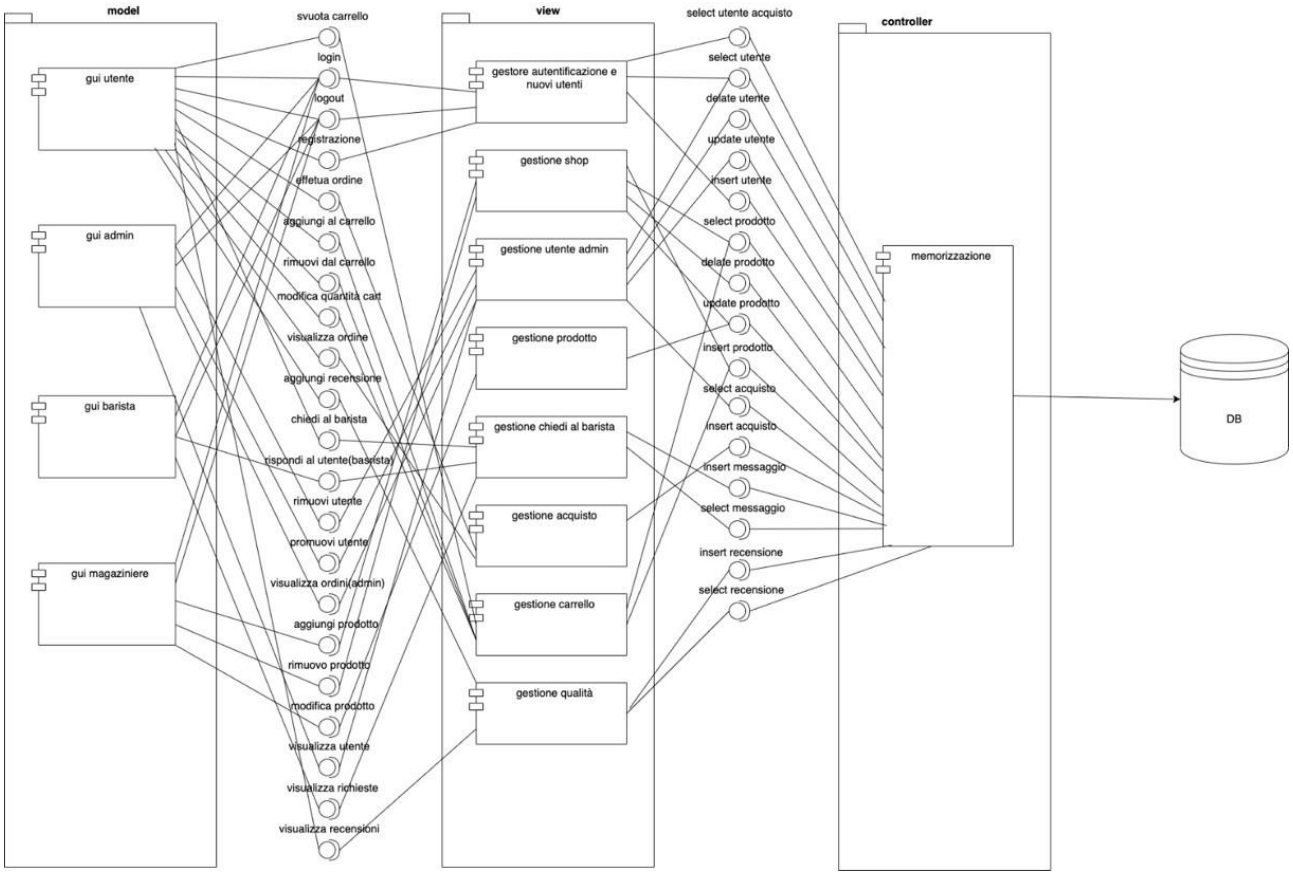
Testing materials

Per effettuare il testing non avremo bisogno di altro che un PC senza nemmeno la connessione ad internet in quanto ogni tipo di test verrà effettuato sull'applicazione che girerà in locale.

Test Cases

Tutti i casi che andremo a testare li prendiamo dal SDD in particolare andremo a testare:
Svuota Carrello
Login
Logout
registrazione
effettua ordine
aggiungi al carrello
rimuovi al carrello
modifica quantità
Visualizza ordine
Aggiungi recensione
Chiedi al barista
Rispondi all'utente
Rimuovi utente
Promuovi utente
Visualizza ordini
Aggiungi prodotto

Rimuovi prodotto
Modifica prodotto
Visualizza utente
Visualizza richieste
Visualizza recensioni
In particolare andremo a testare tutte le funzioni descritte in questa tabella



naturalemnte andremo a testare solamente le funzioni più importanti

Testing schedule

Il testing verrà effettuato parallelamente allo sviluppo del sistema, al fine di individuare e correggere errori nel momento in cui essi verranno introdotti. L'attività di testing è prerogativa di tutti i membri del team, i quali dovranno occuparsi della generazione e dell'esecuzione dei casi di test relativi alle funzionalità che hanno implementato. Tale attività è fondamentale nello sviluppo di un sistema software in quanto la sua mancanza può portare al completo fallimento del sistema. Data tale premessa ne risulta fondamentale la schedulazione e la successiva documentazione.