**Università degli Studi di Salerno**

**Corso di Ingegneria del Software**

Immagine che contiene Carattere, grafica, Elementi grafici, logo

Descrizione generata automaticamente

Test Case Specification

Car – Zone

Versione 0.2

13/02/2024

**Coordinatore del progetto:**

|  |  |
| --- | --- |
| **Nome** | **Matricola** |
| Francesco Pio Cataudo | 0512116773 |

**Partecipanti**:

|  |  |
| --- | --- |
| **Nome** | **Matricola** |
| Francesco Pio Cataudo | 0512116773 |
| Francesco Santoro | 0512117079 |
| Francesco Pio Bottaro | 0512118180 |
| Errico Aquino | 0512117730 |

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autori** |
| **13/12/2024** | 0.1 | Strutturazione documento ed informazioni iniziali. | Team |
| **13/02/2025** | 0.2 | Revisione | Team |

Sommario

[Revision History 3](#_Toc185017107)

[1.Test case specification identifier 5](#_Toc185017108)

[2. Test items 5](#_Toc185017109)

[3. Input specifications 6](#_Toc185017110)

[4. Output specifications 7](#_Toc185017111)

[5. Environmental needs 7](#_Toc185017112)

[6. Special procedural requirements 7](#_Toc185017113)

[7. Intercase dependecies 8](#_Toc185017114)

## 1.Test case specification identifier

Il Test Case Specification Identifier è il nome univoco assegnato ai casi di test, utilizzato per distinguerli tra loro. I nomi dei test sono definiti in base alle funzionalità o ai componenti testati, per facilitare il riferimento ai test durante le fasi di sviluppo e validazione.

L’identificatore di ogni caso di test segue la convenzione TCS-[Modulo]-[NumeroProgressivo] per garantire tracciabilità e organizzazione.

* Nome del caso di test:
  + **Test\_DBConnection\_001** – Verificare la connessione al database utilizzando DBConnect.java
  + **Test\_AddProduct\_002** – Validare l'aggiunta di un prodotto utilizzando addproduct.java
  + **Test\_UserAuthentication\_003** – Controllare la funzionalità di login per amministratori/clienti tramite checkadmin.java e checkcustomer.java
  + **Test\_CartFunctionality\_004** – Verificare il corretto funzionamento delle operazioni sul carrello utilizzando addtocart.java, removecart.java, addtocartnull.java, removecartnull.java
  + **Test\_OrderProcessing\_005** – Validare la creazione e la rimozione di ordini tramite orders.java, remove\_orders.java, order\_details.java, removetable\_order\_details.java
  + **Test\_ProductManagement\_006** – Verificare le operazioni di gestione dei prodotti come aggiornamento e eliminazione utilizzando updateproductname.java, updateproductprice.java, updateproductquantity.java, updateproductimage.java, deleteProduct.java
  + **Test\_CustomerManagement\_007** – Validare la creazione, l'autenticazione e l'eliminazione dell'account cliente utilizzando addcustomer.java, checkcustomer.java, deletecustomer.java
  + **Test\_ContactUsFunctionality\_008** – Assicurarsi che l'invio e la gestione del modulo di contatto funzionino correttamente tramite addContactus.java, addContactusc.java, remove\_contactus.java
  + **Test\_ShippingAddress\_009** – Validare la gestione degli indirizzi di spedizione utilizzando ShippingAddress2.java

Questi casi di test validano i componenti critici del sistema, garantendo il corretto funzionamento della connessione al database, gestione dei prodotti, autenticazione utente, operazioni sul carrello, elaborazione degli ordini, gestione clienti, gestione dei moduli di contatto e gestione degli indirizzi di spedizione.

## 2. Test items

* Componenti in test:
  + **Database Connection**: DBConnect.java
  + **Gestione Prodotti**: addproduct.java, updateproductname.java, updateproductprice.java, updateproductquantity.java, updateproductimage.java, deleteProduct.java
  + **Autenticazione Utente**: checkadmin.java, checkcustomer.java
  + **Gestione Carrello**: addtocart.java, removecart.java, addtocartnull.java, removecartnull.java
  + **Elaborazione Ordini**: orders.java, remove\_orders.java, order\_details.java, removetable\_order\_details.java
  + **Gestione Clienti**: addcustomer.java, checkcustomer.java, deletecustomer.java
  + **Gestione Contatti**: addContactus.java, addContactusc.java, remove\_contactus.java
  + **Gestione Indirizzi di Spedizione**: ShippingAddress2.java
* Funzionalità in test:
  + Verifica della connessione al database
  + Aggiunta, modifica e eliminazione di prodotti
  + Autenticazione degli utenti (admin e clienti)
  + Operazioni sul carrello (aggiunta, rimozione)
  + Creazione e gestione degli ordini
  + Registrazione e gestione dei clienti
  + Invio e gestione delle richieste di contatto
  + Gestione degli indirizzi di spedizione

## 3. Input specifications

Input richiesti:

* Credenziali utente (admin/customer) per autenticazione
* Dati del prodotto (nome, prezzo, quantità, immagine) per gestione prodotti
* ID prodotto per operazioni su carrello e gestione prodotti
* ID ordine per elaborazione e gestione degli ordini
* Dati cliente (nome, email, password) per registrazione e gestione clienti
* Informazioni di contatto (nome, email, messaggio) per modulo di contatto
* Indirizzo di spedizione per gestione delle consegne

## 4. Output specifications

Output attesi:

* Messaggi di successo o errore per autenticazione utente
* Conferma di aggiunta, aggiornamento o eliminazione di un prodotto
* Stato aggiornato del carrello dopo operazioni di aggiunta/rimozione
* Creazione confermata di un ordine con riepilogo dettagliato
* Notifica di registrazione avvenuta con successo per un cliente
* Conferma di ricezione del messaggio di contatto
* Verifica del corretto salvataggio di un indirizzo di spedizione

## 5. Environmental needs

* **Requisiti hardware:**
  + Server con specifiche minime: CPU quad-core, 16GB RAM, 500GB SSD
  + Dispositivi client compatibili con browser moderni
  + Rete stabile con connessione a internet
* **Requisiti software:**
  + Database MySQL/PostgreSQL configurato correttamente
  + Application server (Tomcat, GlassFish, o equivalente)
  + Ambiente di sviluppo (Java JDK 11+, Eclipse/IntelliJ)
  + Strumenti di test (JUnit, Selenium per test UI, Postman per API testing)

## 6. Special procedural requirements

**Vincoli:**

* Test da eseguire in orari di bassa attività per ridurre l’impatto sul sistema
* Simulazione di carichi di utenti per test di performance
* Backup del database prima di test distruttivi (es. eliminazione dati)
* Autorizzazione necessaria per eseguire test in ambienti di produzione

## 7. Intercase dependecies

**Dipendenze tra i test:**

* **Test\_DBConnection\_001** deve essere eseguito prima di qualsiasi altro test dipendente dal database.
* **Test\_UserAuthentication\_003** deve essere completato prima di eseguire test relativi alle operazioni utente.
* **Test\_AddProduct\_002** deve precedere i test relativi alla gestione del carrello e degli ordini.
* **Test\_CartFunctionality\_004** deve precedere **Test\_OrderProcessing\_005** per verificare l’acquisto di prodotti.
* **Test\_CustomerManagement\_007** deve essere eseguito prima di testare la gestione degli ordini e dell’autenticazione clienti.