Laurea Triennale in Informatica - Università degli Studi di Salerno

Corso di *Ingegneria del Software* - Prof. C. Gravino

System Design Document

ReVampAscent

Riferimento	ReVampAscent_SDD_V2.0
Versione	2.0
Data	25/11/2025
Destinatario	C. Gravino

Presentato da	Antonio Aliberti, Raffaella Di Pasquale, Vincenzo Martucci

Team composition

Ruolo	Nome	Acronimo	Contatti
Team Member	Antonio Aliberti	AA	a.aliberti56@studenti.unisa.it
Team Member	Raffaella Di Pasquale	RDP	r.dipasquale5@studenti.unisa.it
Team Member	Vincenzo Martucci	VM	v.martucci5@studenti.unisa.it

Sommario

Revision History

Data	Versione	Descrizione	Autori
26/11/2025	0.1	Prima stesura del documento	[TEAM MEMBER]
27/11/2025	0.2	Introduzione e Architettura del Sistema Corrente	[TEAM MEMBER]
30/11/2025	0.3	Mapping Hardware/Software, Condizioni Boundary, Servizi dei sottosistemi	AA
30/11/2025	0.4	Panoramica, Condizioni Boundary	AA
30/11/2025	0.5	Decomposizione in sottosistemi, Controllo degli accessi e sicurezza	AA e VM
30/11/2025	0.6	Gestione dati persistenti, Servizi dei sottosistemi	AA
30/11/2025	0.7	Controllo del flusso globale del sistema	RDP
01/12/2025	0.8	Diagramma Architetturale	VM
01/12/2025	0.9	Servizi dei sottosistemi	[TEAM MEMBER]
02/12/2025	0.10	Revisione Design Goals, aggiunta di un Trade-Off	RDP
03/12/2025	0.11	Revisione e Glossario	RDP
09/12/2025	1.0	Revisione documento	[TEAM MEMBER]
12/02/2025	2.0	Revisione finale documento	[TEAM MEMBER]

1. Introduzione

1.1 Scopo del Sistema

Negli ultimi anni, l'acquisto di arredi e complementi d'arredo ha subito una forte trasformazione digitale, ma molti utenti incontrano ancora difficoltà legate alla complessità delle piattaforme, alla scarsa chiarezza delle informazioni e ai processi di pagamento poco trasparenti o frammentati.
L'obiettivo del sistema ReVamp Ascent è quello di fornire un e-commerce integrato e intuitivo, che consenta una gestione completa dell'esperienza di acquisto online, dallo sfoglio del catalogo alla ricezione dell'ordine, riducendo al minimo la distanza tra l'esperienza d'acquisto fisica e quella virtuale.
Il sistema mira a semplificare l'interazione tra cliente e piattaforma, rendendo immediato l'accesso alle informazioni su prodotti, disponibilità, spedizione e pagamento. Ogni articolo è corredato da una scheda dettagliata contenente immagini ad alta risoluzione, descrizioni, dimensioni, materiali, colore, prezzo e scontistica applicata, affinché l'utente possa compiere una scelta consapevole.
L'acquisto avviene in modo guidato: l'utente autenticato può aggiungere articoli al carrello, procedere al checkout, inserire o modificare indirizzi di spedizione e scegliere il metodo di pagamento preferito. Il sistema gestisce ogni passaggio con conferme, validazioni e notifiche automatiche, garantendo trasparenza e affidabilità durante tutto il processo.
Parallelamente, la piattaforma consente agli amministratori di gestire in maniera centralizzata l'intero catalogo, con la possibilità di aggiungere, modificare o eliminare prodotti, aggiornare le immagini e i dettagli tecnici, oltre a monitorare ordini, pagamenti e recensioni dei clienti.
In questo modo, ReVamp Ascent assicura un flusso operativo continuo e tracciabile, con dati sempre coerenti tra interfaccia utente e database.
Dal punto di vista tecnico e progettuale, il sistema si propone come soluzione affidabile, sicura e scalabile, adottando protocolli di sicurezza TLS/HTTPS, sistemi di autenticazione e autorizzazione basati su credenziali cifrate, e strutture dati ottimizzate per la gestione contemporanea di più utenti.
L'interfaccia è pensata per essere user-friendly e responsive, adattandosi a dispositivi desktop, tablet e mobile, in modo da rendere l'esperienza d'acquisto accessibile ovunque.
Il progetto nasce quindi con lo scopo di ridefinire l'esperienza d'acquisto nel settore dell'arredamento online, garantendo un ambiente affidabile, coerente e integrato che permetta agli utenti di esplorare, scegliere e acquistare in modo sicuro e piacevole, mentre offre agli amministratori una piattaforma stabile e gestibile per coordinare l'intera filiera digitale del commercio.

1.2 Design Goals

In questo paragrafo vengono illustrati i Design Goals del sistema ReVamp Ascent, ovvero gli obiettivi progettuali che guidano le scelte architetturali, tecnologiche e di implementazione dell'intera piattaforma.

Essi derivano dai requisiti non funzionali definiti nel RAD e rappresentano le qualità prioritarie che il sistema deve garantire per essere considerato corretto, affidabile e conforme agli standard di un moderno e-commerce.

Nella tabella che segue, i design goal sono descritti in base ai seguenti campi, ognuno corrispondente a una colonna:

- **Rank**: ne indica la priorità rispetto agli altri design goal;
- **ID Design Goal**: un identificatore;
- **Descrizione**: ne descrive le caratteristiche;
- **Categoria**: raggruppa i design goal che descrivono qualità che rientrano nella stessa macroarea;
- **Origine**: il requisito non funzionale da cui è stato generato;
- **Trade off**: rappresenta una scelta tra due o più parametri, dove la diminuzione di uno costituisce l'aumento dell'altro.

In particolare, le categorie in cui sono stati suddivisi i design goal sono:

- **Performance**: indica, in maniera quantificabile, il livello di prestazioni del sistema software;
- **Dependability**: relativo all'affidabilità del sistema (gestione degli errori, resistenza ai crash…);
- **Cost**: indice dell'impatto economico dell'implementazione di una qualità sui costi di sviluppo;
- **Maintenance**: indica quanto effort è necessario per apportare modifiche al sistema dopo la prima release;
- **End user**: relativo ai canoni dell'interazione uomo-macchina e di ingente importanza per valutare la compatibilità di un sistema software con l'utenza.

Rank / Priorità	ID Design Goal	Descrizione design goal	Categoria	Origine	Trade-off

1	**DG_1** User Friendly	L'interfaccia dovrà essere intuitiva e di semplice utilizzo anche per utenti non esperti. Le azioni dovranno risultare chiare e prive di ambiguità.	End-user	**RNF_U.1**	Usability vs Security
2	**DG_2** Persistenza e integrità dei dati	I dati saranno memorizzati su database relazionale MySQL, con relazioni normalizzate e integrità referenziale.	Dependability	**RNF_I.5**	Data Integrity vs Performance
3	**DG_3** Riservatezza	Gli utenti autenticati potranno accedere solo alle informazioni a loro destinate (es. storico ordini personale).	Dependability	**RNF_A.6**	Security vs Performance
4	**DG_4** Completezza funzionale	Ogni modulo deve essere completamente operativo e integrato con le relative funzionalità correlate (es. aggiunta al carrello, checkout, storici ordini).	End user	**RNF_F.2**	Robustness vs Usability

5	**DG_5** Robustezza	Il sistema dovrà gestire input errati o incompleti senza crash o perdita di dati, mantenendo coerenza e stabilità.	Dependability	**RNF_A.4**	Robustness vs Usability

6	**DG_6** Portabilità	Il sistema deve poter essere utilizzato su hardware diversi sotto un unico codice sorgente.	Maintenance	**RNF_I.3**	Portability vs Performance
7	**DG_7** Accessibilità multipiattaforma	Il sito sarà ottimizzato per l'uso su PC, tablet e smartphone senza necessità di installazione.	Maintenance	**RNF_PA.1**	Responsive vs Development Complexity
8	**DG_8** Manutenibilità	Il sistema sarà facilmente manutenibile grazie all'elevata modularità del suo codice.	Maintenance	**RNF_S.1**	Maintainability vs Initial Cost/Time

9	**DG_9** Sicurezza dati sensibili	Tutte le comunicazioni saranno protette tramite protocollo HTTPS. Le password saranno cifrate tramite algoritmo di hashing sicuro (es. SHA-1).	Dependability	**RNF_A.2**	Security vs Performance
10	**DG_10** Responsive	Il sistema, siccome sarà reperibile su diverse piattaforme, dovrà adattarsi a tutti i diversi tipi di schermi.	End-user	**RNF_U.3**	Responsive vs Development Complexity
11	**DG_11** Evoluzione	Il sistema deve consentire la sua evoluzione, in previsione di future aggiunte di funzionalità	Maintenance	**RNF_S.2**	Maintainability vs Initial Cost/Time
12	**DG_12** Aggiornamenti continui	Gli aggiornamenti del sistema saranno effettuati server-side, senza necessità di download o reinstallazioni da parte dell'utente.	Maintenance	**RNF_PA.2**	Portability vs Performance

1.2 Trade-offs

Trade-off	Descrizione
Security vs Performance	La gestione di dati sensibili (credenziali, indirizzi, ordini, pagamenti) richiede l'adozione di misure avanzate di sicurezza quali HTTPS, hashing, tokenizzazione, validazioni server-side, controlli sui metodi di pagamento e sistemi di antimanomissione. Questi meccanismi, pur essenziali per la protezione degli utenti, comportano un naturale aumento della latenza nelle richieste, soprattutto durante le fasi critiche come login e checkout. Il progetto sceglie di privilegiare la sicurezza, mantenendo comunque i tempi di risposta entro limiti accettabili, consapevole del fatto che una protezione robusta comporta un incremento nei tempi di elaborazione.
Maintainability vs Initial Cost/Time	Per garantire che il sistema sia facilmente manutenibile ed estensibile in futuro (RNF_S.1, RNF_S.2), è stata adottata un'architettura rigorosamente modulare, con separazione netta tra logica di business, accesso ai dati e interfaccia. Scrivere codice così strutturato richiede molto più tempo e la creazione di numerosi file aggiuntivi rispetto a una programmazione "monolitica" o non strutturata. Si accetta un aumento dei Costi di Sviluppo (Tempo) nella fase iniziale per garantire una drastica riduzione dei costi di manutenzione futura. Si investe oggi per risparmiare domani
Responsive vs Development Complexity	Rendere l'interfaccia capace di adattarsi fluidamente a qualsiasi dimensione di schermo (PC, Tablet, Smartphone) richiede una logica di presentazione complessa (CSS Media Queries avanzate, layout fluidi, test su device multipli). Questo aumenta la complessità del codice Front-end rispetto a un sito con layout fisso per desktop. Si privilegia l'usabilità e l'esperienza dell'utente finale (End-user). Dato che una gran parte del traffico e-commerce proviene da mobile, si accetta la maggiore complessità implementativa pur di non perdere la fetta di utenza che naviga da smartphone.
Portability vs Performance	La scelta di sviluppare una Web Application basata su Java (Servlet container) garantisce che il software possa girare su qualsiasi sistema operativo (Windows, Linux, MacOS) e che gli aggiornamenti siano centralizzati (RNF_PA.2). Tuttavia, l'uso di un ambiente gestito (JVM) e l'astrazione del browser impediscono di ottenere le prestazioni assolute che si

	avrebbero con un'applicazione nativa compilata specificamente per l'hardware in uso (es. C++). Si privilegia la Portabilità e la facilità di distribuzione. Per un sistema e-commerce, l'accessibilità universale è un fattore critico di successo molto più rilevante rispetto all'ottimizzazione estrema delle performance hardware.
Usability vs Security	Il Design Goal DG_1 mira a rendere l'interazione il più fluida e priva di ostacoli possibile. Tuttavia, i requisiti di Sicurezza (DG_3, DG_9) impongono barriere necessarie, come la richiesta di password complesse (lettere, numeri, maiuscole), la ri-autenticazione per operazioni sensibili (es. modifica credenziali) e timeout di sessione brevi. In questo conflitto, il punto di equilibrio pende verso la Sicurezza nelle aree critiche (Login, Checkout, Area Admin), accettando una minore fluidità d'uso (es. dover reimmettere la password). Nelle aree pubbliche (Navigazione Catalogo), si privilegia invece l'Usabilità, evitando controlli invasivi finché non strettamente necessari.
Data Integrity vs Performance	Per garantire la consistenza dei dati (DG_2), il sistema utilizza un database relazionale con vincoli di integrità referenziale (Foreign Keys). Ogni volta che viene salvato un ordine, il DBMS esegue controlli multipli per assicurare che il cliente esista, che i prodotti siano disponibili e che il prezzo sia corretto. Questo processo di verifica introduce una latenza di scrittura che contrasta con l'obiettivo di massima velocità di risposta (DG_7). Si privilegia l'integrità dei dati. In un sistema e-commerce, la correttezza di un ordine e dei pagamenti è prioritaria rispetto al guadagno di pochi millisecondi. Le prestazioni vengono recuperate ottimizzando le operazioni di lettura (che sono più frequenti delle scritture) tramite indicizzazione.
Robustness vs Usability	Per soddisfare il DG_5, il sistema applica una validazione rigorosa sugli input (es. nella fase di registrazione), rifiutando qualsiasi dato che non rispetti il formato atteso. Questo comportamento "rigido" protegge il sistema dai crash, ma può risultare frustrante per l'utente (DG_1) che si vede rifiutare l'invio del modulo per errori formali. Si privilegia la robustezza. È inaccettabile che dati "sporchi" (es. lettere nel prezzo, quantità negative) entrino nel database. Per mitigare l'impatto negativo sull'usabilità, il sistema fornisce messaggi di errore parlanti (Feedback

	immediato) che guidano l'utente nella correzione, trasformando il vincolo tecnico in un aiuto alla compilazione.

1.3 Definizioni, acronimi e abbreviazioni

Acronimi:

- RAD = Requirement Analysis Document
- SDD = System Design Document
- COTS = Commercial Off-the-Shelf Definizioni:
- Sottosistema: corrisponde alla parte di lavoro che può essere svolta autonomamente da un singolo sviluppatore o da un gruppo di sviluppatori. Si ottiene decomponendo il sistema.
- Design Goal: proprietà del sistema sulla quale ci si concentra maggiormente.
- Trade-off: possibilità di ridurre una certa qualità per aumentare il valore di un'altra qualità e viceversa. Il termine è espresso talvolta come costo opportunità, riferendosi a più alternative alle quali si è preferito rinunciare a vantaggio di un'altra scelta.
- Mapping Hardware-Software: descritto per indicare i vari device hardware utilizzati dal sistema e la loro interazione con le componenti software.
- Dati Persistenti: dati che devono sopravvivere all'esecuzione singola dell'applicazione. Sono i dati che vengono salvati nel database.

1.4 Riferimenti

I riferimenti sono:

- Libro, "Object Oriented Software Engineering using UML, Patterns and Java"
  Edizione: 3rd Edition
  Anno: 2014
  Autori: Bernd Bruegge, Allen H. Dutoit

- Report.it_RAD
- Report.it_SOW
- Wikiwand
- Red Hat

1.5 Panoramica

Il presente documento è strutturato in quattro sezioni principali:

- **Introduzione**: in questa sezione del documento è possibile trovare una descrizione dello scopo del sistema, i vari design goals ordinati per priorità ed arricchiti di descrizione, i trade-offs ed informazioni circa il linguaggio ed i riferimenti utilizzati.
- **Architettura del sistema corrente**: in questa sezione del documento è possibile analizzare l'architettura del sistema attualmente esistente.
- **Architettura del sistema proposto**: in questa sezione del documento è possibile analizzare tramite una descrizione accurata l'architettura del sistema da noi proposto. Tale sezione è a sua volta suddivisa in: ○
  decomposizione dei sottosistemi; ○
  mapping hardware/software; ○
  gestione dati persistenti;
  ○ controllo degli accessi e sicurezza, del flusso globale del sistema; ○
  condizioni boundary.
- **Servizi dei sottosistemi**: in questa sezione del documento è possibile leggere la descrizione dei servizi di ogni sottosistema proposto.
- **Glossario**: sezione del documento che associa ad ogni sigla/termine menzionato/a all'interno del documento una definizione/descrizione per evitare che il lettore vada a ricercare informazioni all'esterno del documento, rendendolo autosufficiente sulla materia trattata.

2. Architettura del sistema corrente

Attualmente **non esiste alcun sistema centralizzato** che riproduca in modo completo e integrato l'insieme delle funzionalità offerte da **ReVamp Ascent**, ovvero una piattaforma unificata per la gestione del catalogo prodotti, la consultazione delle schede articolo, il carrello digitale, il checkout strutturato, la gestione degli indirizzi, i metodi di pagamento e la pubblicazione di recensioni, il tutto in un'unica applicazione coerente e progettata ad hoc.

Il processo di acquisto, in assenza di un sistema come ReVamp Ascent, risulta **frammentato**: l'utente è spesso costretto a confrontarsi con piattaforme che presentano **cataloghi poco aggiornati**, schede prodotto incomplete, metodi di pagamento non omogenei oppure procedure di checkout complesse e non guidate.
Sul versante amministrativo, non esiste un sistema unico che consenta la **gestione semplificata del catalogo**, l'**aggiornamento delle immagini**, la **modifica dei prodotti**, il controllo degli stock e il monitoraggio degli ordini, rendendo le operazioni di back-office lente e distribuite tra più strumenti non integrati.

Se volessimo individuare dei possibili concorrenti, potremmo fare riferimento a piattaforme e-commerce generaliste, ma nessuna di esse rappresenta un vero modello architetturale comparabile:
ogni soluzione offre solo **porzioni** delle funzionalità richieste (catalogo, o solo carrello, o solo gestione ordini), mentre ReVamp Ascent propone un flusso unificato che comprende **catalogo, carrello, checkout, ordine, pagamento, recensioni, amministrazione**, mantenendo **coerenza dei dati** e **continuità operativa**.

Pertanto, **non esiste una reale architettura esistente** che possa essere adottata come riferimento diretto o base comparativa: il sistema corrente è di fatto **inesistente**, e la piattaforma proposta nasce per colmare questa mancanza, offrendo un ecosistema completo, coerente e progettato specificamente per una gestione fluida dell'esperienza d'acquisto.

3. Architettura del sistema proposto

3.1 Panoramica

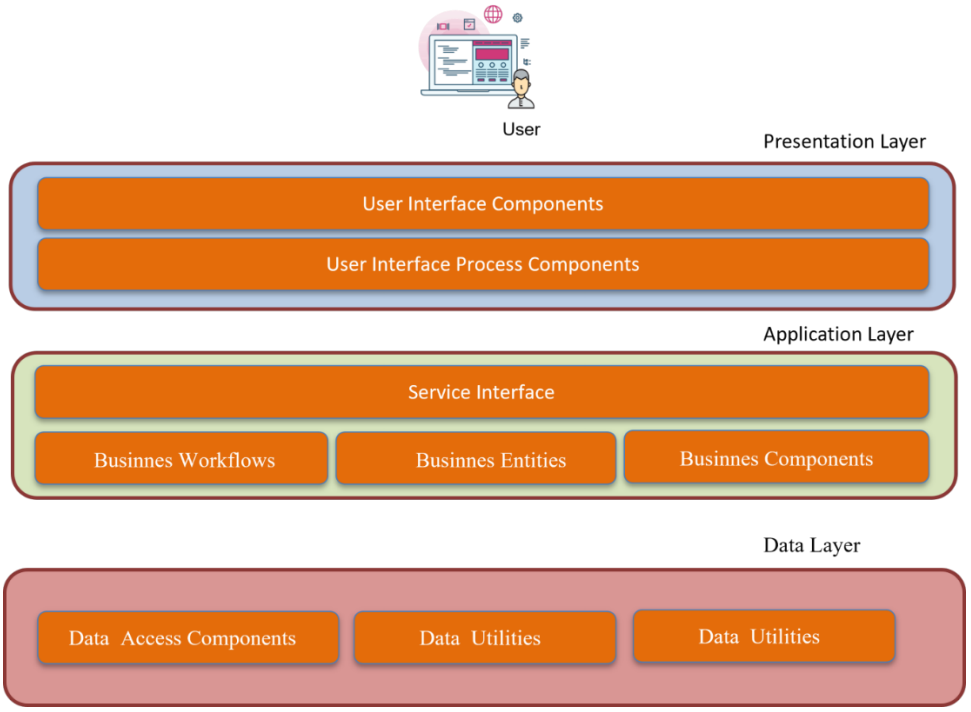
Il sistema proposto per ReVamp Ascent è strutturato secondo una chiara separazione tra livello di presentazione, logica applicativa e gestione dei dati, in modo da favorire manutenibilità, estendibilità e riuso del codice.

L'applicazione è una web application sviluppata in Java, in cui la parte di interazione con l'utente è gestita tramite pagine JSP/HTML, CSS per la formattazione grafica e JavaScript/AJAX per le chiamate asincrone e l'aggiornamento dinamico dei contenuti (ad esempio per operazioni su carrello e catalogo senza ricaricare l'intera pagina).

La logica di controllo e di business è implementata tramite Servlet Java e classi dedicate, che si occupano di ricevere le richieste dal front-end, validare i dati, orchestrare i flussi (es. login, gestione carrello, checkout, inserimento prodotti da parte dell'amministratore) e interagire con il livello di persistenza.

La gestione dei dati è affidata a un database relazionale MySQL, con tabelle dedicate a clienti, indirizzi, ordini, articoli, categorie, carrello, pagamenti e recensioni. L'accesso al database avviene attraverso apposite classi di accesso ai dati, che incapsulano le query SQL e centralizzano la logica di persistenza, riducendo la duplicazione del codice e facilitando eventuali modifiche allo schema dati.

Questa organizzazione consente di mantenere distinto ciò che riguarda la presentazione all'utente, la logica di elaborazione e la memorizzazione persistente delle informazioni, semplificando sia l'evoluzione futura del sistema (nuove funzionalità, modifiche al database, restyling grafico) sia le attività di test e manutenzione.


User
Presentation Layer
User Interface Components
User Interface Process Components
Application Layer
Service Interface
Businnes Workflows
Businnes Entities
Businnes Components
Data Layer
Data Access Components
Data Utilities
Data Utilities

3.2 Decomposizione in sottosistemi

Il sistema è stato decomposto nei seguenti sottosistemi:

- **Gestione Identità e Accessi**: Il sottosistema è garante dell'identificazione e dell'accesso sicuro al sistema da parte degli attori (Cliente e Admin) attraverso le funzionalità di Login e Logout. Si interessa inoltre della gestione del ciclo di vita dell'account, inclusa la registrazione di nuovi utenti, la modifica delle credenziali e la gestione dei dati anagrafici e di spedizione.
- **Gestione Catalogo**: Il sottosistema si interessa delle funzionalità relative alla presentazione dell'inventario digitale. In particolare, gestisce la navigazione tra le categorie, la ricerca dinamica dei prodotti in base al nome e la visualizzazione dettagliata delle schede tecniche degli articoli per l'Utente.
- **Carrello e Checkout**: Il sottosistema gestisce la fase transitoria d'acquisto. È garante delle funzionalità di aggiunta, rimozione e aggiornamento quantità degli articoli nel carrello virtuale e coordina l'intera procedura di Checkout, dalla selezione dell'indirizzo alla simulazione del pagamento finale.
- **Gestione Ordini e Storico**: Il sottosistema è responsabile della persistenza delle transazioni concluse. Si interessa delle funzionalità relative alla consultazione dello storico acquisto per il Cliente e offre all'Admin strumenti per il monitoraggio e il filtraggio temporale degli ordini presenti nel sistema.
- **Gestione Recensioni**: Il sottosistema si interessa delle funzionalità relative al feedback sui prodotti. Permette ai Clienti autenticati di inserire valutazioni e commenti e garantisce la visualizzazione di tali contenuti all'interno delle schede prodotto.
- **Amministrazione**: Il sottosistema raggruppa le funzionalità di back-office riservate esclusivamente all'attore Admin. È garante delle operazioni di manipolazione del catalogo (inserimento, modifica e cancellazione di articoli) e della visualizzazione dell'elenco globale degli utenti registrati.
- **Persistenza**: Il sottosistema gestisce la comunicazione diretta con la base di dati, fungendo da ponte tra l'applicazione e l'archivio fisico delle informazioni. Attraverso le classi DAO, si occupa di eseguire le interrogazioni e gli aggiornamenti dei dati (come inserimento, lettura,

modifica e cancellazione), trasformando i record del database in oggetti utilizzabili dal sistema. Si interfaccia con il DBMS MySQL per assicurare che le informazioni siano memorizzate in modo permanente e sicuro.

Component Diagram UML

Nel seguente Component Diagram UML sono mostrati i sottosistemi e le relative dipendenze tra essi

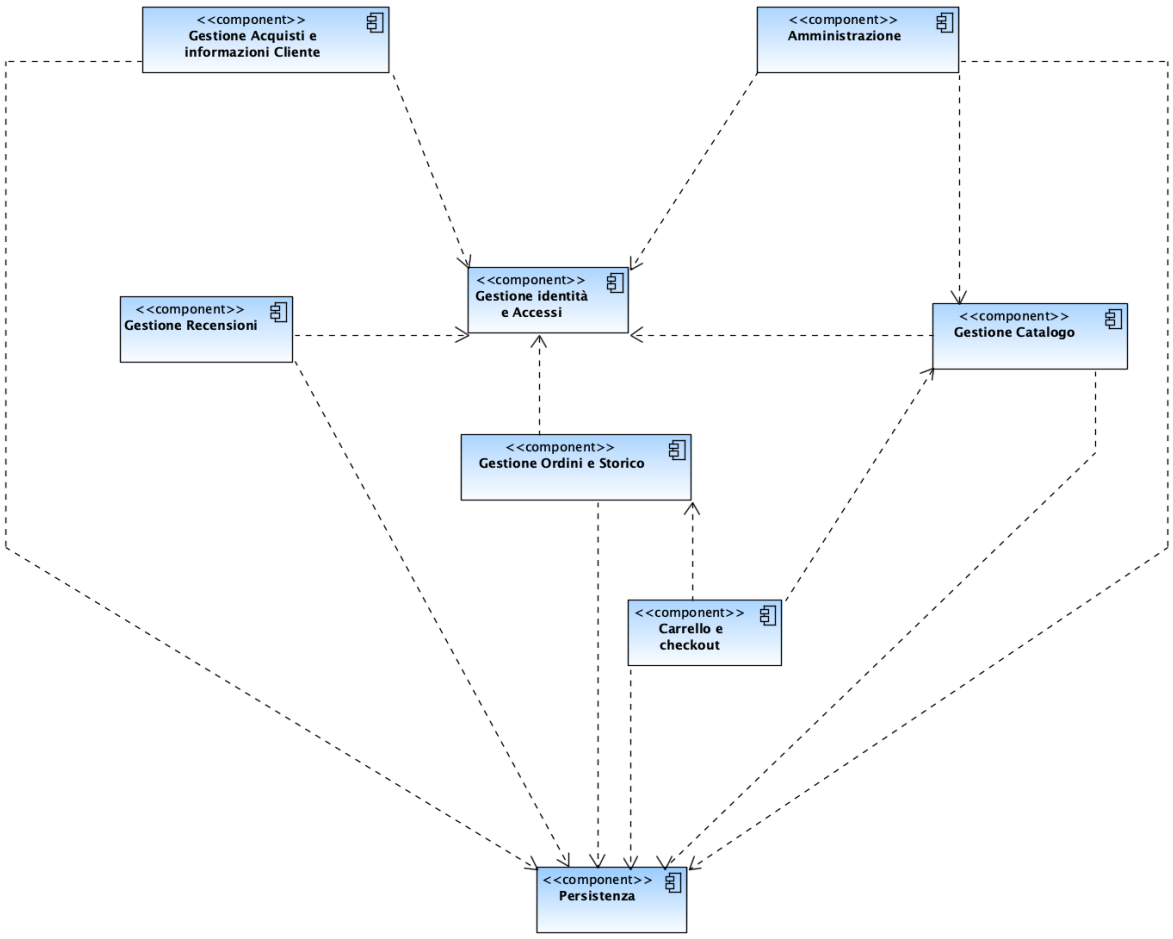
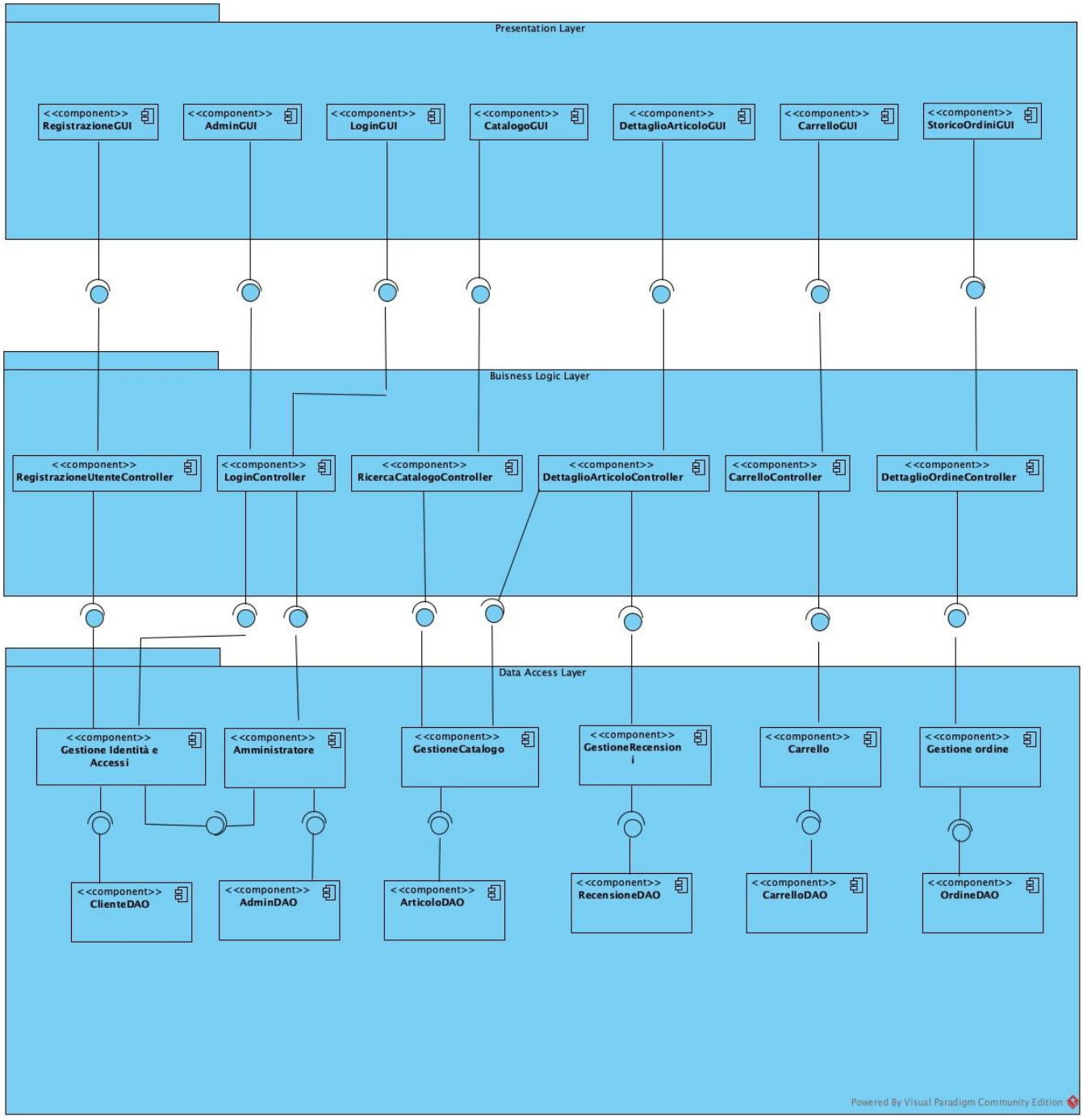


Diagramma Architetturale



3.3 Mapping Hardware/Software

La piattaforma ReVamp Ascent è una Web Application accessibile dagli utenti tramite un comune web browser (es. Chrome, Safari, Edge) su dispositivi Desktop, Tablet e Smartphone dotati di connessione Internet.

L'architettura fisica del sistema segue il modello Three-Tier (a tre livelli), così strutturato:

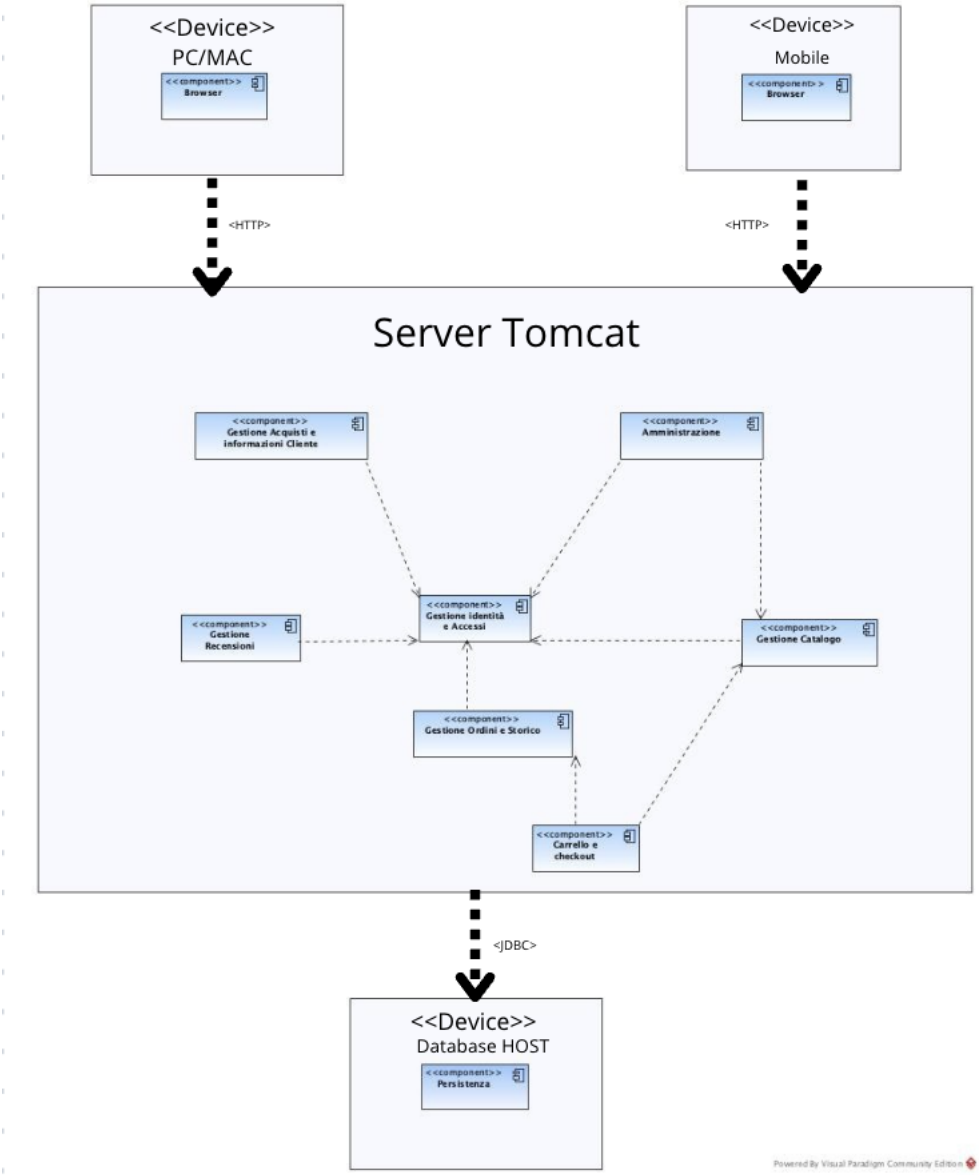
1. **Client Tier** (Presentation): Il dispositivo dell'utente finale che esegue il browser. Non richiede installazione di software specifico, ma si limita a renderizzare l'interfaccia HTML/CSS/JS fornita dal server.

2. **Middle Tier** (Application Server): Un server dedicato che ospita il Web Container (Apache Tomcat). Qui risiede la logica di business (Servlet, JSP, Bean) che elabora le richieste HTTP del client.

3. **Data Tier** (Database Server): Un server dedicato alla persistenza dei dati che ospita il DBMS MySQL.

Di seguito un UML Deployment Diagram che descrive il mapping hardware/software:

UML Deployment Diagram



3.4 Gestione dati persistenti

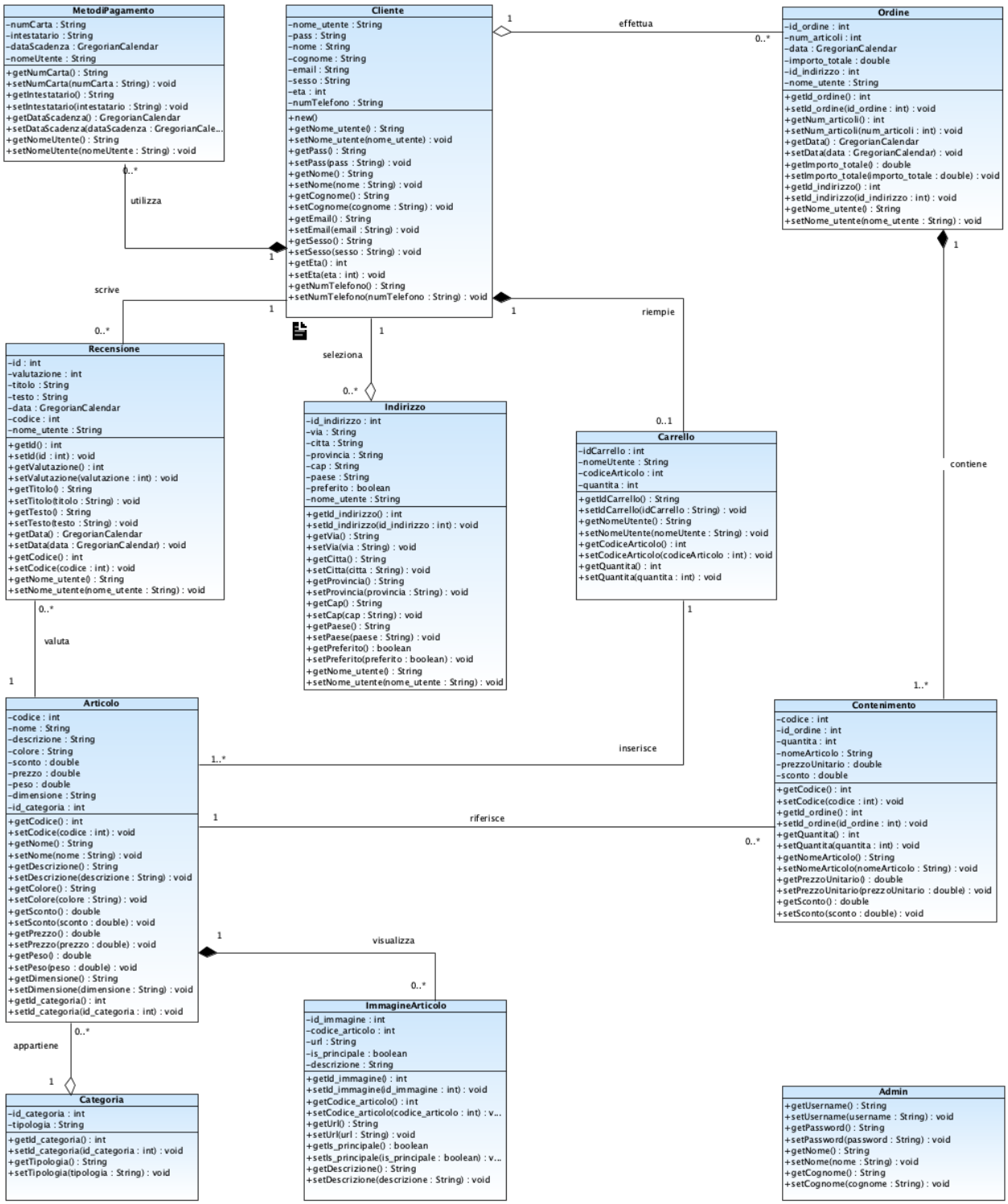
Per la gestione dei dati persistenti del sistema si è deciso di utilizzare un **database relazionale MySQL**, scelta naturale per un sistema di e-commerce in cui è fondamentale garantire **coerenza, integrità referenziale e transazioni affidabili** (es. ordini, pagamenti, aggiornamento dello stock).

L'adozione di un DB SQL come MySQL è stata motivata dai seguenti aspetti principali:

1. **Modello dati strutturato e relazioni chiare**
   Il dominio di ReVamp Ascent (Clienti, Indirizzi, Ordini, Articoli, Categorie, Carrello, Pagamenti, Recensioni) è fortemente relazionale.
   Un database SQL consente di modellare in modo esplicito tali relazioni tramite chiavi primarie/esterne, vincoli di integrità e normalizzazione, riducendo ridondanze ed errori nei dati.
2. **Integrità e coerenza delle transazioni (ACID)**
   Operazioni critiche come la conferma di un ordine, l'associazione del pagamento e l'aggiornamento delle giacenze devono essere eseguite in maniera atomica e consistente.
   MySQL supporta pienamente le proprietà ACID, garantendo che le transazioni vengano completate correttamente o annullate in caso di errore, evitando stati "a metà" (es. ordine creato ma stock non aggiornato).
3. **Query complesse e reportistica**
   L'e-commerce richiede spesso interrogazioni complesse (storici ordini per cliente, articoli più venduti, fatturato per periodo, filtri combinati su categoria/prezzo/sconto).
   Il linguaggio SQL di MySQL permette di esprimere tali query in modo potente ed efficiente, semplificando anche la futura integrazione con strumenti di analisi o reportistica.
4. **Maturità, portabilità e supporto**
   MySQL è un RDBMS ampiamente diffuso, stabile e ben documentato, supportato da un vasto ecosistema di strumenti (driver JDBC, client grafici, tool di backup, monitoraggio, migrazione).
   Ciò semplifica sia le fasi di sviluppo e testing, sia le attività di manutenzione e deploy in ambienti diversi (locale, server universitario, hosting esterno).

In sintesi, l'uso di **MySQL** consente al sistema di gestire in modo robusto e affidabile i dati critici dell'e-commerce, mantenendo un modello logico chiaro e facilmente estendibile, perfettamente allineato alle esigenze di tracciabilità e coerenza proprie di un progetto di questo tipo.

CD_SDD: Entity Class Diagram



3.5 Controllo degli accessi e sicurezza

Di seguito viene mostrata la matrice degli accessi per poter tenere traccia di quali attori possono accedere ai quali dei servizi offerti dal sistema.

Oggetti \ Attori	Ospite (Non Registrato)	Cliente (Registrato)
Gestione Identità e Accessi	Registrazione, Login	Logout, ModificaCredenziali, VisualizzaProfilo, GestioneIndirizzi, VisualizzaMetodiPagamento, AggiungiMetodoPagamento
Gestione Catalogo	VisualizzaCatalogo, RicercaProdotti, VisualizzaDettaglioArticolo, FiltraPerCategoria, VisualizzaImmagini	VisualizzaCatalogo, RicercaProdotti, VisualizzaDettaglioArticolo, FiltraPerCategoria, VisualizzaImmagini
Carrello e Checkout	AggiungiAlCarrello, VisualizzaCarrello, RimuoviDalCarrello, AggiornaQuantità	AggiungiAlCarrello, VisualizzaCarrello, RimuoviDalCarrello, AggiornaQuantità, ProcediAlCheckout, SimulaPagamento
Gestione Ordini e Storico	(Nessun Accesso)	VisualizzaStoricoPersonale, VisualizzaDettaglioOrdine
Gestione Recensioni	VisualizzaRecensioni	VisualizzaRecensioni, InserisciRecensione
Amministrazione	(Nessun Accesso)	(Nessun Accesso)

Attori / Oggetti	Amministratore (Admin)
Gestione Identità e Accessi	Login, Logout
Gestione Catalogo	VisualizzaCatalogo, RicercaProdotti, VisualizzaDettaglioArticolo, AggiungiArticolo, ModificaArticolo, EliminaArticolo
Carrello e Checkout	(Nessun Accesso)
Gestione Ordini e Storico	VisualizzaTuttiOrdini, FiltraOrdiniPerData, VisualizzaDettaglioOrdine
Gestione Recensioni	(Nessun Accesso)
Amministrazione	VisualizzaListaUtenti, VisualizzaDettagliCliente

3.6 Controllo del flusso globale del sistema

Il sistema ReVamp Ascent è una piattaforma software interattiva basata su interfaccia web; ciò significa che l'utente (Cliente o Admin), navigando le pagine del sito, impartisce comandi che attivano specifiche funzionalità di business.

Quando un attore invia una richiesta tramite l'interfaccia grafica, il sistema intercetta l'evento e lo indirizza al componente di controllo competente. Questo trasferisce il flusso allo strato logico (Business Logic), dove risiedono i moduli software incaricati di erogare i servizi dell'applicazione (come la gestione del carrello o l'elaborazione degli ordini). Infine, il processo coinvolge lo strato di persistenza (Data Layer), utilizzato per immagazzinare, recuperare e aggiornare le informazioni nel database

3.7 Condizioni Boundary

In questo paragrafo vengono illustrate le condizioni boundary riguardanti il fallimento del sistema e l'errore di accesso ai dati persistenti.

Fallimento del sistema

Identificativo	**UCBC_1 – Fallimento del sistema**	**Data**	28/11/2025
		Versione	1.0
		Autori	Antonio Aliberti
Descrizione	L'UC descrive il comportamento in caso di mancata risposta o crash del server web (es. Timeout, Errore HTTP 500).		
Attore principale	Utente (Cliente o Admin)		
Attori secondari	NA		
Entry condition	Il server non risponde alla richiesta HTTP entro il tempo limite (Time-out), OPPURE il server restituisce un errore critico imprevisto.		
Exit condition On success	Il servizio torna disponibile e l'utente riesce a ricaricare la pagina.		
Exit condition On failure	Il sito rimane irraggiungibile; l'utente non può proseguire la navigazione.		

Flusso di eventi principale		
1	Sistema	Rileva l'impossibilità di elaborare la richiesta e mostra una pagina di cortesia (o il browser notifica l'errore di connessione).
2	Utente	Ritenta la connessione riavviando l'applicazione.

Errore di accesso ai dati persistenti (Database Error)

Identificativo	**UCBC_2 – Errore di accesso ai dati persistenti**	**Data**	28/11/2025
		Versione	1.0
		Autori	Antonio Aliberti
Descrizione	L'UC descrive il comportamento del sistema quando non riesce a recuperare o salvare informazioni nel Database (es. durante il Login o il Checkout).		
Attore principale	Sistema		
Attori secondari	Utente (che riceve la notifica)		
Entry condition	Il sistema tenta una query SQL (lettura/scrittura) ma la connessione al DB fallisce o i dati risultano corrotti/inconsistenti.		
Exit condition On success	La connessione al DB viene ripristinata e l'operazione viene completata al tentativo successivo.		
Exit condition On failure	L'operazione (es. acquisto) viene annullata per preservare l'integrità dei dati.		

Flusso di eventi principale		
1	Sistema	Intercetta l'eccezione (es. SQLException), impedisce operazioni parziali e reindirizza l'utente a una pagina di errore dedicata o una pagina generica di servizio.
2	Utente	Prende atto dell'errore e riprova l'operazione in un secondo momento.

4. Servizi dei sottosistemi

In questa sezione vengono descritti i servizi di ogni sottosistema precedentemente elencati.

Sottosistema Gestione Identità e Accessi

Servizio	Descrizione	Interfaccia
Registrazione Utente	Questa funzionalità permette a un nuovo visitatore di creare un account personale fornendo i propri dati anagrafici.	UtenteService
Login	Questa funzionalità permette di effettuare l'accesso al sistema tramite username e password per sbloccare le funzionalità riservate (es. acquisto, recensioni).	UtenteService
Logout	Questa funzionalità permette di disconnettersi dal sistema	UtenteService
Modifica Credenziali	Questa funzionalità permette all'utente di aggiornare la propria password o il proprio username	UtenteService
Gestione Indirizzi	Questa funzionalità permette di visualizzare, aggiungere o rimuovere indirizzi di spedizione associati all'account.	UtenteService

Sottosistema Gestione Catalogo

Servizio	Descrizione	Interfaccia

Visualizza Catalogo	Questa funzionalità permette di visualizzare l'elenco completo dei prodotti o di filtrarli per categoria specifica	CatalogoService
Ricerca Prodotti	Questa funzionalità permette di cercare articoli digitando una parola chiave (ricerca testuale).	CatalogoService
Visualizza Dettaglio Articolo	Questa funzionalità permette di accedere alla scheda completa di un prodotto (prezzo, descrizione, disponibilità, immagini).	CatalogoService
Recupera Immagini	Questa funzionalità recupera e mostra la galleria fotografica associata a un determinato articolo.	CatalogoService

Sottosistema Carrello e Checkout

Servizio	Descrizione	Interfaccia
Aggiungi al Carrello	Questa funzionalità permette di inserire un prodotto nel carrello specificandone la quantità desiderata.	CarrelloService

Aggiorna Carrello	Questa funzionalità permette di modificare la quantità di un articolo già presente nel carrello o di rimuoverlo.	CarrelloService
Visualizza Carrello	Questa funzionalità mostra il riepilogo degli articoli selezionati con il calcolo del totale parziale.	CarrelloService
Procedi al Checkout	Questa funzionalità avvia la procedura di finalizzazione dell'acquisto, richiedendo la selezione di indirizzo e metodo di pagamento.	CheckoutService
Simula Pagamento	Questa funzionalità valida i dati della carta di credito e simula la transazione bancaria per confermare l'ordine.	CheckoutService

Sottosistema Gestione Ordini e Storico

Servizio	Descrizione	Interfaccia

Crea Ordine	Questa funzionalità registra nel sistema un nuovo ordine confermato, salvando i dettagli dei prodotti acquistati e l'importo pagato.	OrdineService
Visualizza Storico Personale	Questa funzionalità permette al Cliente di consultare l'elenco dei propri ordini passati e verificarne lo stato.	OrdineService
Filtra Ordini (Admin)	Questa funzionalità permette all'Amministratore di visualizzare e filtrare tutti gli ordini del sistema in base a un intervallo di date.	OrdineService
Visualizza Dettaglio Ordine	Questa funzionalità permette di esaminare le singole righe (prodotti) che compongono uno specifico ordine.	OrdineService

Sottosistema Gestione Recensioni

Servizio	Descrizione	Interfaccia
Inserisci Recensione	Questa funzionalità permette a un Cliente che ha acquistato un prodotto di lasciare un voto (rating) e un commento testuale.	RecensioneService
Visualizza Recensioni	Questa funzionalità recupera e mostra tutte le recensioni associate a un	RecensioneService

	prodotto nella pagina di dettaglio.	

Sottosistema Amministrazione

Servizio	Descrizione	Interfaccia
Aggiungi Articolo	Questa funzionalità permette all'Admin di inserire un nuovo prodotto nel catalogo, definendone tutte le proprietà.	AdminService
Modifica Articolo	Questa funzionalità permette all'Admin di aggiornare i dati (prezzo, stock, descrizione) di un prodotto esistente.	AdminService
Visualizza Lista Utenti	Questa funzionalità permette all'Admin di consultare l'elenco completo dei clienti registrati sulla piattaforma.	AdminService

Sottosistema di Persistenza

Servizio	Descrizione	Interfaccia
Gestione Connessione	Questa funzionalità gestisce l'apertura e la chiusura delle connessioni verso il DBMS MySQL (Connection Pool).	DriverManager

5. Glossario

Sigla/Termine	Definizione
Cliente (Guest/User)	Attore che interagisce con il sistema navigando il catalogo. Se registrato, può effettuare acquisti, gestire il carrello e visualizzare lo storico ordini.
Amministratore (Admin)	Attore con privilegi elevati responsabile della gestione dei contenuti del sito (Catalog Management), del monitoraggio degli ordini e della supervisione dell'utenza.
Carrello	Struttura dati temporanea (o persistente se loggato) che raccoglie gli articoli selezionati dall'utente con le relative quantità, prima della fase di pagamento.
Checkout	Procedura sequenziale che finalizza l'ordine, comprendente la scelta dell'indirizzo di spedizione, del metodo di pagamento e la conferma finale dell'acquisto.
DAO	Data Access Object. Pattern strutturale che fornisce un'interfaccia astratta verso il database. Nel progetto, le classi DAO (es. ArticoloDAO) gestiscono tutte le operazioni CRUD verso MySQL.
DBMS	Database Management System. Sistema software progettato per consentire la creazione, la manipolazione e l'interrogazione efficiente dei database. Nel progetto viene utilizzato MySQL.
Componenti COTS	Con il termine componenti COTS ci si riferisce a componenti hardware e software disponibili sul mercato per l'acquisto da parte di aziende di sviluppo interessate a utilizzarli nei loro progetti (https://www.wikiwand.com/it/COTS).
JSP	JavaServer Pages. Tecnologia utilizzata per creare pagine web dinamiche inserendo codice Java all'interno dell'HTML. Rappresenta il livello di "View" nel sistema.
Architettura 3-Tier	Architettura a 3 Livelli. Modello architetturale che suddivide l'applicazione in tre strati logici e fisici indipendenti: Presentation Tier (Interfaccia/JSP), Logic Tier (Business Logic/Servlet) e Data Tier (Database/MySQL). Questa separazione garantisce scalabilità, sicurezza e facilità di manutenzione.