

Università degli Studi di Salerno

Corso di Ingegneria del Software

UNISA-TCG

System Design Document



Data: 25/11/2025

Progetto: UNISA-TCG	Versione: 1.0
Documento: System Design Document	Data: 24/11/2025

Coordinatore del progetto:

Nome	Matricola
Sepe Giuseppe	0512119386

Partecipanti:

Nome	Matricola
Senatore Francesco	0512120568
Sepe Giuseppe	0512119386
Sullo Diego	0512119455

Revision History

Data	Versione	Descrizione	Autore
25/11/2025	1.0	System Design Document	Francesco Senatore, Diego Sullo, Giuseppe Sepe

Indice

1. Introduzione.....	5
1.1 Scopo del Sistema.....	5
1.2 Obiettivi di Design (Design Goals - DG).....	5
1.3 Trade-off.....	7
1.4 Definizioni, Acronimi e Abbreviazioni.....	8
1.5 Riferimenti.....	10
1.6 Panoramica.....	10
2. Architettura Software Corrente (Architetture di Riferimento).....	10
2.1 Analisi delle Architetture Simili (Contesto di Mercato).....	10
3. Architettura Software Proposta	
3.1 Panoramica.....	11
3.2 Scomposizione in Sottosistemi.....	12
3.3 Mappatura Hardware/Software (Deployment).....	14
3.4 Gestione dei Dati Persistenti.....	15
3.4.1 Scelta del Database Management System (DBMS).....	15
3.4.2 Strategia di Hosting (Server Locale).....	15
Implicazioni Operative.....	16
3.4.3 Riferimento allo Schema Logico.....	16
3.5 Sicurezza e Matrice degli Accessi (Matrice Globale).....	16
3.5.1 Matrice degli Accessi.....	17
3.5.2 Misure di Sicurezza Aderenti al GDPR.....	18
3.6 Controllo Globale del Software.....	18
3.6.1 Iniziazione e Propagazione delle Richieste.....	18
3.6.2 Gestione della Concorrenza e Sincronizzazione.....	19
Transazioni e Commit.....	20
3.7 Boundary Conditions.....	20
3.7.1 Comportamento di Avvio (Start-up).....	20
3.7.2 Comportamento di Arresto (Shutdown).....	20
3.7.3 Comportamento di Errore.....	21
A. Errori a Livello di Transazione.....	21
B. Errori a Livello di Sistema.....	21
4. Servizi dei Sottosistemi.....	21
4.1 Sottosistema Autenticazione & Sicurezza.....	21
4.2 Sottosistema Gestione Profilo Utente.....	22
4.3 Sottosistema Gestione Catalogo & Ricerca.....	23
4.4 Sottosistema E-commerce & Carrello.....	23
4.5 Sottosistema Gestione Ordini.....	24
4.6 Sottosistema Gestione Pagamenti.....	25
4.7 Sottosistema Gestione Recensioni.....	25
4.8 Sottosistema Gestione Amministrativa.....	26
4.9 Sottosistema Persistenza Dati (DAO Layer).....	26

1. Introduzione

1.1 Scopo del Sistema

Il progetto **UNISA-TCG (Unisa Cardshop)** consiste nello sviluppo di una **piattaforma web** dedicata alla compravendita e gestione di carte collezionabili (Pokémon, Yu-Gi-Oh!, Magic: The Gathering).

L'applicazione nasce dall'esigenza di superare la **frammentazione del mercato TCG** (Trading Card Game), caratterizzato da cataloghi disomogenei, informazioni incomplete e processi di acquisto poco uniformi.

L'obiettivo è realizzare un ambiente **unificato, affidabile e facilmente utilizzabile**. Il sistema è implementato come un'applicazione web **responsive**, basata su **Java (Servlet/JSP)** e **MySQL**, e distribuita su **Apache Tomcat**.

Obiettivi del progetto:

- Fornire un ambiente unico e intuitivo per la **ricerca, il filtraggio e l'acquisto** di prodotti TCG.
- Garantire la **sicurezza** delle transazioni e la **protezione dei dati personali** nel rispetto del **GDPR**
- Permettere agli utenti di **recensire e valutare** i prodotti .
- Offrire ai **Gestori** la possibilità di amministrare cataloghi, ordini e prezzi da un'interfaccia centralizzata.
- Assicurare **prestazioni elevate**, con tempi di risposta inferiori a un secondo per le operazioni principali .
- Garantire un **uptime minimo del 99%**.

1.2 Obiettivi di Design (Design Goals - DG)

Questi obiettivi di design derivano dai requisiti non funzionali (RNF) identificati nel RAD.

Rango	ID	Design Goal	Descrizione	Categoria	RNF di origine
1	DG_1	Sicurezza & GDPR	Il sistema deve garantire la massima sicurezza delle transazioni e il rispetto del Regolamento UE 2016/679 per i dati.	Dependabilità	RNF su Sicurezza, Legalità

2	DG_2	Tempo di Risposta	Le operazioni principali (es. ricerca) devono avere un tempo di risposta inferiore a un secondo	Performance	RNF su Prestazioni
3	DG_3	Usabilità & Coerenza	L'interfaccia deve essere intuitiva, coerente e responsive , adattandosi a desktop e mobile.	End User	RNF su Usabilità
4	DG_4	Affidabilità & Uptime	Il sistema deve assicurare un funzionamento stabile e continuo con un uptime minimo del 99%	Dependability	RNF su Affidabilità
5	DG_5	Manutenibilità & Modularità	Il codice deve essere modulare e documentato, e il DB normalizzato per facilitare estensioni e manutenzione.	Maintenance	RNF su Supportability
6	DG_6	Compatibilità Browser	L'interfaccia deve essere compatibile con i browser moderni (Chrome, Firefox, Safari, Edge).	Dependability	RNF su Interfaccia
7	DG_7	Gestione Errori	In caso di guasto, il sistema deve fornire messaggi d'errore significativi e gestire i rollback delle transazioni.	Dependability	RNF su Affidabilità

1.3 Trade-off

Trade-off	Descrizione
Tempo di Risposta vs. Completezza Dati	Per garantire un tempo di risposta <1s nella ricerca, si potrebbe ottimizzare la struttura del DB anche a scapito di una completa normalizzazione in ogni aspetto, o limitare il numero di risultati iniziali da mostrare.
Modularità (Java/Tomcat) vs. Velocità Sviluppo	Si è privilegiata la robustezza e manutenibilità di un'architettura Java (Servlet/JSP) su Tomcat, nonostante framework più leggeri potessero offrire una velocità di sviluppo maggiore.
Affidabilità vs. Manutenzione	Per garantire l'affidabilità, sono previsti backup periodici . Tuttavia, la manutenzione potrebbe essere interattiva (approccio standard per WAR su Tomcat) se non si adotta un'architettura più complessa.
Prestazioni (Velocità Ricerca) vs. Accuratezza & Completezza Dati	Per garantire tempi di risposta inferiori a un secondo (DG_2) su un catalogo vasto, potrebbero essere necessarie tecniche di caching o denormalizzazione dei dati di ricerca. Questo può talvolta limitare la freschezza o la precisione assoluta dei risultati rispetto a una query complessa in tempo reale.
Sicurezza (Crittografia/Hashing) vs. Latenza Computazionale	L'implementazione rigorosa di misure di sicurezza (come crittografia HTTPS end-to-end, validazione dei token e hashing complesso delle password per il GDPR) introduce un overhead computazionale. Si accetta una minima latenza di processo per garantire la massima sicurezza (DG_1) .
Usabilità (Semplicità UI) vs. Flessibilità Funzionale (Filtri)	Per mantenere l'interfaccia utente intuitiva (DG_3) e pulita, si è scelto di presentare solo i filtri di ricerca essenziali. Questo può limitare la capacità dei collezionisti professionisti di eseguire query altamente complesse, privilegiando la semplicità d'uso per l'utente medio.

1.4 Definizioni, Acronimi e Abbreviazioni

Termino/Acronimo	Definizione
ACID	(Atomicity, Consistency, Isolation, Durability). Proprietà fondamentali delle transazioni di database che garantiscono l'integrità dei dati, cruciali per il Checkout (UC3).
CRUD	(Create, Read, Update, Delete). Le quattro operazioni fondamentali di manipolazione dei dati utilizzate dal sottosistema Persistenza Dati e gestite dal Gestore sul Catalogo (UC5).
DAO	Data Access Object. Pattern software (Layer di Astrazione) che gestisce l'accesso ai dati, separando la logica di business dal database (L3).
DBMS	Database Management System. Software utilizzato per gestire e amministrare il database relazionale (MySQL).
GDPR	General Data Protection Regulation (Regolamento UE 2016/679). Normativa sulla protezione dei dati personali a cui il sistema deve conformarsi.
HTTPS	Hypertext Transfer Protocol Secure. Protocollo di comunicazione utilizzato per crittografare i dati in transito tra L1 e L2.
JDBC	Java Database Connectivity. API standard Java utilizzata dall'Application Server (L2) per connettersi al Database Server (L3).

L1, L2, L3	I tre livelli dell'architettura (3-Tier): L1 (Presentazione), L2 (Logica Applicativa), L3 (Dati).
PSP	Payment Service Provider. Fornitore di servizi di pagamento esterno (simulato) utilizzato per autorizzare le transazioni finanziarie.
RAD	Requirements Analysis Document. Documento che precede l'SDD e contiene la specifica dei requisiti e i modelli (Classi, Casi d'Uso, Sequenza).
Rollback	Operazione eseguita dal DBMS che annulla tutte le modifiche apportate durante una transazione in caso di fallimento (Gestione Errori 3.7).
SDD	System Design Document. Il presente documento che specifica l'architettura e la progettazione del sistema.
TCG	Trading Card Game. Il tipo di prodotto gestito dal sistema (es. carte collezionabili).
Tomcat	Apache Tomcat. L'Application Server (container di servlet) utilizzato come ambiente di esecuzione per la logica L2.
UC	Use Case (Caso d'Uso). Sequenza di azioni definita nel RAD (es. UC3: Checkout e Pagamento).
Uptime	Percentuale di tempo in cui il sistema è completamente operativo e disponibile agli utenti.

WAR	Web Archive. Formato di file utilizzato per impacchettare i componenti web dell'applicazione per il deployment su Tomcat.
------------	--

1.5 Riferimenti

- Requirements Analysis Document (RAD) UNISA-TCG
 - System Design-Documentation
-

1.6 Panoramica

Questo documento segue la struttura definita dal documento di riferimento ed è formato dalle seguenti sezioni:

- **Introduzione:** Descrive lo scopo del sistema UNISA-TCG, gli obiettivi di design e i compromessi (Trade-off) stabiliti.
- **Architettura software corrente:** Il sistema UNISA-TCG è sviluppato *da zero*. Non esiste un'architettura software preesistente che il progetto sia destinato a sostituire.
- **Architettura software proposta:** Descrive la partizione del sistema in sottosistemi (livelli), il mapping Hardware/Software e la gestione dei dati persistenti (Schema DB).
- **Servizi dei sottosistemi:** Descrive i compiti, le funzionalità e le interazioni chiave tra i moduli software, basandosi sui modelli dinamici e oggetti (UML) del RAD.

2. Architettura Software Corrente (Architetture di Riferimento)

2.1 Analisi delle Architetture Simili (Contesto di Mercato)

Il sistema **UNISA-TCG** è stato concepito per competere e superare i limiti delle piattaforme esistenti nel mercato delle carte collezionabili (TCG). L'analisi di fattibilità ha tenuto conto di piattaforme quali **CardMarket**, **TCGPlayer** ed **eBay**.

Queste piattaforme tendono a operare tipicamente con un'**Architettura Multilivello (Tiered Architecture)**, essenziale per la gestione di ampi cataloghi, un'elevata concorrenza di utenti e la sicurezza delle transazioni e-commerce.

- **Livello Presentazione (Front-end):** Interfaccia Web Responsive (necessaria per la compatibilità con dispositivi mobili).
- **Livello Applicativo (Back-end):** Logica di business complessa per gestire ricerca, filtri unificati, carrelli e transazioni.
- **Livello Dati (Database):** Necessario per la persistenza di cataloghi massivi, dati utente e storico ordini.

3. Architettura del sistema Proposto

L'architettura proposta per UNISA-TCG si allinea a questo modello, ma mira a risolvere i problemi di **frammentazione e disomogeneità** dei dati che affliggono i sistemi attuali

3.1 Panoramica

L'architettura selezionata per il sistema UNISA-TCG è l'**Architettura a 3 Livelli (3-Tier Architecture)**. Questa scelta è in linea con le best practice per lo sviluppo di applicazioni e-commerce web-based e garantisce una chiara separazione delle responsabilità, come richiesto dal requisito di **modularità (DG_5)** e **sicurezza (DG_1)**.

1. **Livello Presentazione (Client/Front-end):** Responsabile dell'interfaccia utente (UI) e della gestione delle interazioni del cliente (browser). Implementato con **HTML, CSS e JavaScript**.
2. **Livello Logica Applicativa (Back-end/Business Logic):** Contiene l'intera logica di business (ricerca, gestione carrello, calcolo totale, ecc.). Implementato in **Java (Servlet/JSP)** e distribuito su **Apache Tomcat**. Questo livello funge da intermediario, garantendo che il client non acceda mai direttamente al database, migliorando la sicurezza.
3. **Livello Dati (Database/Data Tier):** Responsabile della persistenza, memorizzazione e recupero dei dati. Implementato con **MySQL**.

Centralizzando la logica di business nel livello intermedio, l'architettura 3-Tier facilita la **manutenibilità (DG_5)** e la **scalabilità (DG_3)**, permettendo modifiche ai moduli di business o al database senza intaccare l'interfaccia utente, e viceversa.

3.2 Scomposizione in Sottosistemi

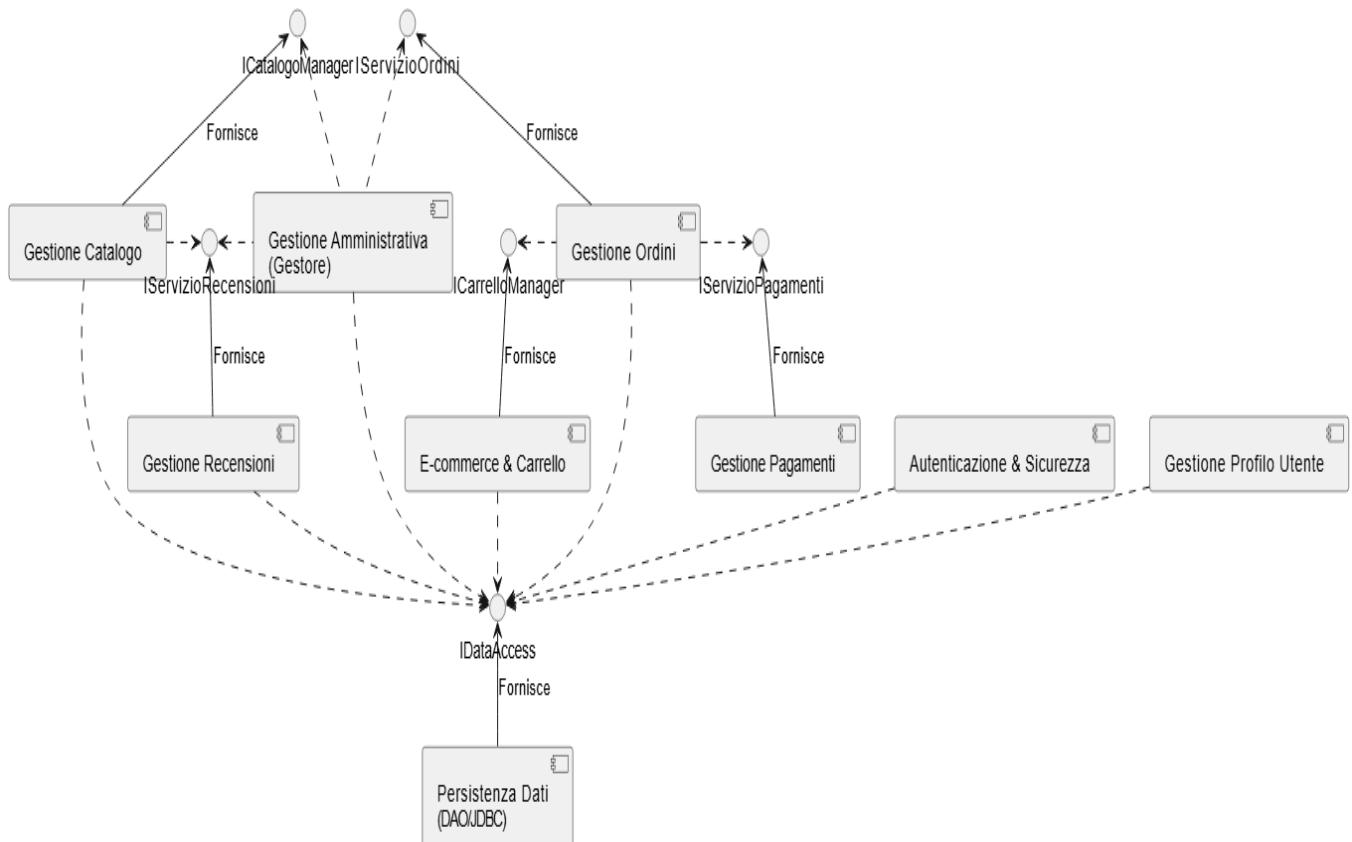
Il sistema UNISA-TCG è partizionato in sottosistemi logici (moduli), ciascuno con responsabilità ben definite, che implementano la logica di business e di controllo definita nel RAD.

Sottosistema	Funzionalità Principali	Componenti Logici Chiave (UML/Pattern)

Autenticazione & Sicurezza	Login, Logout, Registrazione, Reset Password. Gestione della logica di sicurezza: verifica credenziali, monitoraggio login falliti e blocco account.	Servizio Autenticazione, Utente, Utente Registrato, Gestore.
Gestione Profilo Utente	Visualizzazione, modifica e aggiornamento dei dati dell'utente.	Servizio Account, Utente Registrato.
Gestione Catalogo	Ricerca, Filtri e Visualizzazione dei prodotti. Dal lato Gestore, gestisce le operazioni CRUD (Create, Read, Update, Delete) su Prodotto e Categoria.	Catalogo Manager, Prodotto, Categoria.
E-commerce & Carrello	Logica per l'aggiunta/rimozione articoli, calcolo del subtotale e gestione delle quantità disponibili in tempo reale.	Gestione Carrello Control, Carrello.
Gestione Ordini	Creazione, registrazione e persistenza dell'istanza Ordine. Gestione del flusso di Checkout e aggiornamento dello stato dell'ordine e della Spedizione.	Servizio Ordini, Ordine,, Spedizione.
Gestione Pagamenti	Interfaccia per la simulazione della transazione con il PSP esterno e la sua registrazione sul sistema.	Servizio Pagamenti (simulato), Pagamento.
Gestione Recensioni	Gestisce la sottomissione e la moderazione di recensioni . Implementa la logica di business per il ricalcolo automatico del rating medio del Prodotto (come da	Servizio Recensioni, Recensione,Gestore.

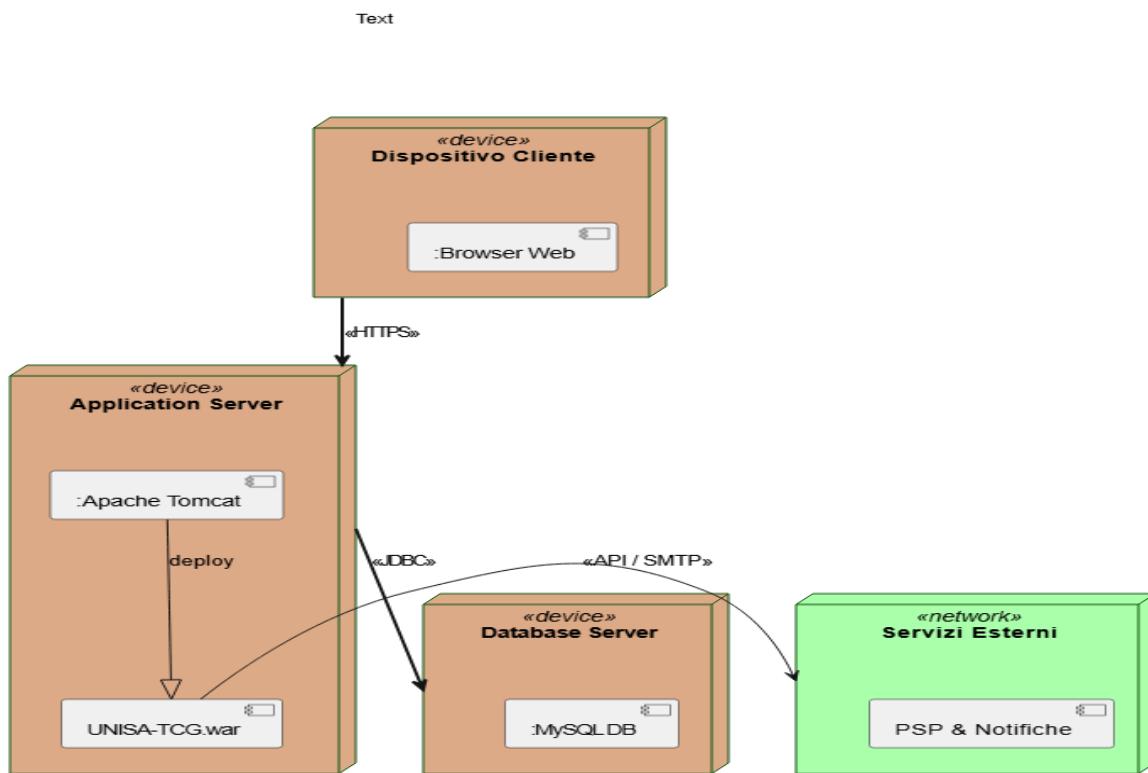
	diagramma dinamico entry/Ricalcola rating).	
Gestione Amministrativa (Gestore)	Fornisce i Servizi Amministrativi per la gestione degli Utenti, dello stato degli Ordini e la gestione del Catalogo.	Servizio Amministrazione, Gestore.
Persistenza Dati	Livello di accesso ai dati (livello Dati 3-Tier). Responsabile della mappatura tra oggetti Java e record del DB MySQL.	Data Access Objects (DAO), JDBC Layer .

Sono mostrate di seguito le dipendenze tra i sottosistemi attraverso un component diagram UML:



3.3 Mappatura Hardware/Software (Deployment)

Il sistema UNISA-TCG sarà implementato come **applicazione web responsive** e distribuito su una tipica architettura 3-Tier, che garantisce modularità, scalabilità e separazione delle responsabilità.



3.4 Gestione dei Dati Persistenti

Questa sezione definisce la strategia di persistenza e i criteri di gestione per i dati del sistema UNISA-TCG, essenziali per garantire la consistenza, l'affidabilità e la sicurezza delle informazioni.

3.4.1 Scelta del Database Management System (DBMS)

Per la gestione e il salvataggio dei dati persistenti, è stato scelto un **Database Management System (DBMS) relazionale**, basato sulla tecnologia **MySQL**. Questa scelta è in linea con l'architettura tecnica del Back-end, che utilizza Java e JDBC.

L'utilizzo di un DBMS relazionale è stato scelto per rispettare gli obiettivi di progettazione relativi alla gestione dei dati, offrendo:

- **Integrità dei Dati:** Il DBMS applica vincoli (es. chiavi primarie, chiavi esterne, vincoli di dominio) per garantire l'integrità dei dati e la coerenza tra le diverse entità (es. Ordine e Prodotto), verificandoli durante ogni aggiornamento del database.
- **Privacy dei Dati:** Il sistema sfrutterà i meccanismi di sicurezza del DBMS per consentire l'accesso differenziato ai dati, garantendo che i diversi utenti (Acquirenti, Gestori) possano accedere solo a specifiche parti del database, in ottemperanza ai requisiti **GDPR**.
- **Affidabilità dei Dati:** Il DBMS supporta la creazione di **backup periodici** e offre metodi per il ripristino immediato del database in caso di guasti software o hardware, garantendo un funzionamento stabile e continuo.
- **Atomicità delle Operazioni:** Le transazioni (come l'UC3: Checkout e Pagamento) vengono eseguite interamente (commit) o non vengono eseguite affatto (rollback), mantenendo la coerenza del database.

3.4.2 Strategia di Hosting (Server Locale)

Abbiamo deciso di adottare una strategia di hosting **locale (On-Premise)** per il Database Server MySQL.

Questa soluzione implica che il Database Server sia installato e gestito direttamente su un'infrastruttura hardware controllata dal committente o dal team di sviluppo, offrendo i seguenti vantaggi operativi e strategici:

- **Pieno Controllo dell'Infrastruttura:** L'hosting locale garantisce il **pieno controllo** sull'ambiente operativo del database, permettendo la personalizzazione, l'ottimizzazione specifica del tuning del DBMS e la gestione diretta delle risorse hardware (CPU, RAM, Storage) per allinearsi esattamente alle esigenze di performance del sistema.
- **Sicurezza e Conformità:** Il controllo diretto sulla posizione fisica dei dati (Data Sovereignty) può semplificare il rispetto di determinate politiche di sicurezza o requisiti di conformità legale.
- **Ottimizzazione dei Costi a Lungo Termine:** Sebbene i costi iniziali di setup siano maggiori, l'hosting locale elimina i costi operativi ricorrenti (abbonamento al servizio cloud) offrendo un **risparmio sui costi a lungo termine**.
- **Latenza di Rete Minima:** Se l'Application Server e il Database Server sono collocati all'interno della stessa rete locale o struttura hardware, si ottiene una **latenza minima** nella comunicazione JDBC tra L2 e L3, contribuendo al rispetto del requisito di **tempi di risposta inferiori a un secondo** per le operazioni principali.

Implicazioni Operative

L'adozione di un server locale rende l'infrastruttura e la sua manutenzione responsabilità diretta del team operativo. Pertanto, per garantire il requisito di **uptime minimo del 99%** e l'affidabilità richiesta, dovranno essere implementate soluzioni interne robuste, tra cui:

- Configurazione e monitoraggio dei processi di **backup periodico** e ripristino dei dati.
- Gestione della sicurezza del server MySQL.

3.4.3 Riferimento allo Schema Logico

Lo schema del database relazionale è derivato direttamente dal **Modello a Oggetti (Class Diagram)** definito nella Sezione 3.4.3 del Requirements Analysis Document.

Le tabelle del database rifletteranno le classi logiche del sistema, quali **Prodotto, Ordine, Utente Registrato, Carrello, e Recensioni**, con l'implementazione delle opportune associazioni.

3.5 Sicurezza e Matrice degli Accessi (Matrice Globale)

La matrice seguente modella l'accesso per i principali ruoli del sistema sui sottosistemi di Logica Applicativa (L2), seguendo i principi di **controllo degli accessi sulle classi/sottosistemi**.

3.5.1 Matrice degli Accessi

Sottosistema (Risorsa)	Non Registrato (Ospite)	Acquirente (Registrato)	Gestore
Autenticazione & Sicurezza	W (Register), E (Login)	E (Login, Logout, Reset Pwd)	R (Log Accessi) / W (Sblocco Account)
Gestione Profilo Utente	-	R, W (Visualizza/Aggiorna <i>Proprio Profilo</i>)	R, W (Visualizza/Aggiorna <i>Tutti i Profili</i>)

Gestione Catalogo	R (Ricerca, Filtri, Dettagli)	R	R, W (CRUD Prodotto/Categoria - UC5)
E-commerce & Carrello	W (Crea Carrello di sessione)	R, W (Modifica Carrello - UC2)	R (A scopo di debug/analisi)
Gestione Ordini	R (Tracking/Stato, se noto)	R, E (Crea Ordine, Visualizza <i>Propri</i> Ordini)	R, W (Visualizza <i>Tutti</i> gli Ordini, Aggiorna Stato - UC4)
Gestione Pagamenti	E (Esegue Transazione)	E (Esegue Transazione - UC3)	R (Log Transazioni)
Gestione Recensioni	-	R, W (Visualizza e Pubblica - UC7)	R, W (Moderazione e Rimozione)
Gestione Amministrativa	-	-	E (Esegue Servizi Amministrativi)
Persistenza Dati	-	-	R (Backup, Log, Manutenzione)

Legenda Permessi: **R** (Read), **W** (Write/Update/Create), **E** (Execute operazione).

3.5.2 Misure di Sicurezza Aderenti ai GDPR

Le politiche di accesso definite nella matrice sono supportate dalle seguenti misure di sicurezza, essenziali per la protezione dei dati personali:

- **Integrità del Sistema:** L'accesso al database (Livello Dati) è strettamente limitato al solo Livello Logica Applicativa (L2) tramite il layer **DAO**, impedendo l'accesso diretto ai client.
- **Crittografia:** Tutte le comunicazioni sensibili avvengono tramite **HTTPS**.
- **Hashing:** Le password sono memorizzate in formato hash per garantire la sicurezza anche in caso di violazione del database.
- **Principio del Minimo Privilegio:** Agli attori sono assegnati solo i permessi minimi necessari per svolgere le loro funzioni, come specificato nella matrice.

3.6 Controllo Globale del Software

Questa sezione descrive come le richieste degli utenti vengono iniziate (iniziazione) e propagate attraverso i sottosistemi (propagazione), e come il sistema gestisce la sincronizzazione e i problemi di concorrenza.

3.6.1 Iniziazione e Propagazione delle Richieste

Il sistema UNISA-TCG adotta uno stile di controllo **basato su Eventi e Invocatione Sincrona (Procedural Control)**, tipico delle architetture a livelli web (3-Tier).

- **Iniziazione (Livello 1):** Una richiesta viene iniziata quando l'utente (Acquirente/Gestore) interagisce con l'**Interfaccia Utente (Frontend)** (L1), generando un evento (es. click sul pulsante "Aggiungi al Carrello" o invio del form di Login).
- **Propagazione (Livello 2):** Il browser invia una richiesta **HTTP/S sincrona** all'**Application Server (L2)**. La richiesta viene intercettata dal Servlet/Controller appropriato (ad esempio, il Gestione Carrello Control).
- **Orchestrazione:** Il Controller (L2) invoca sequenzialmente i metodi sui sottosistemi di business necessari (es. Gestione Ordini invoca Gestione Pagamenti). La sincronizzazione tra i sottosistemi è implicita nella natura **procedurale** delle chiamate Java (un metodo aspetta il ritorno dell'altro).
- **Livello Dati (L3):** Ogni sottosistema L2 che necessita di persistenza invoca in modo sincrono il layer **Persistenza Dati (DAO)** tramite **JDBC**, attendendo la conferma dell'operazione.

3.6.2 Gestione della Concorrenza e Sincronizzazione

Le principali problematiche di concorrenza emergono quando più utenti tentano di modificare le stesse risorse critiche contemporaneamente, in particolare l'inventario (Quantità del Prodotto) e lo stato di una transazione (Ordine).

Problema di Concorrenza	Sottosistema Coinvolto	Soluzione Implementata
Consistenza dell'Inventario	E-commerce & Carrello, Gestione Ordini	Locking Ottimistico o Pessimistico: Durante il checkout, il sistema deve bloccare o verificare la disponibilità delle scorte (Quantità del Prodotto) immediatamente prima della finalizzazione dell'ordine per evitare vendite multiple dello stesso articolo.
Integrità Transazionale	Gestione Ordini, Gestione Pagamenti	Transazioni Atomiche (ACID): L'intero processo di Checkout (dalla sottrazione delle scorte alla registrazione dell'Ordine) sarà eseguito come un'unica transazione di database per garantire l' Atomicità . Se il pagamento fallisce, tutte le modifiche (es. allo stato dell'inventario) vengono annullate (Rollback).
Accesso Dati Condivisi	Persistenza Dati	Locking del DBMS (MySQL): Il DBMS gestirà automaticamente il locking a livello di riga o tabella per garantire che due scritture simultanee non corrompano i dati.

Per garantire l'integrità, il controllo globale stabilisce che la sequenza di operazioni che modifica lo stato critico del sistema deve essere eseguita atomicamente:

1. Verifica finale e blocco delle risorse (Prodotto.Quantità).
2. Delegazione a Gestione Pagamenti.
3. Registrazione dell'oggetto Ordine e DettagliOrdine.
4. **Commit** o **Rollback** dell'intera operazione.

Questo approccio garantisce che il sistema sia affidabile e che gli errori (come indicato nel RAD) siano gestiti con un **rollback** per prevenire la perdita di dati.

3.7 Boundary Conditions

Questa sezione definisce il comportamento del sistema nei suoi limiti operativi: avvio, arresto controllato e reazione agli errori.

3.7.1 Comportamento di Avvio (Start-up)

Il sistema **UNISA-TCG** segue un avvio sequenziale per garantire che il Livello Logica Applicativa (L2) non tenti di accedere al database prima che la connessione sia disponibile:

1. **Avvio dell'Application Server:** L'operatore avvia **Apache Tomcat**.
 2. **Deployment WAR:** Tomcat esegue il *deployment* del file **UNISA-TCG.war**.
 3. **Inizializzazione Servizi L2:** I componenti Java di business vengono caricati.
 4. **Verifica Connettività:** Il layer **Persistenza Dati (DAO)** inizializza il pool di connessioni **JDBC** verso **MySQL**.
 5. **Pronto all'Uso:** Il sistema è considerato "Online" quando la connessione al DB è stabilita e i servizi L2 sono pronti a rispondere alle richieste HTTP/S.
-

3.7.2 Comportamento di Arresto (Shutdown)

L'arresto del sistema è un processo controllato e graduale per preservare l'integrità dei dati e prevenire transazioni incomplete.

1. **Disabilitazione Accettazione:** L'Application Server smette di accettare nuove richieste dal Front-end.
 2. **Svuotamento Transazioni:** Il sistema attende il completamento delle transazioni attive (es. finalizzazione dell'ordine, commit/rollback).
 3. **Rilascio Risorse:** Il layer **Persistenza Dati** chiude in modo pulito il pool di connessioni JDBC.
 4. **Arresto:** Tomcat e i servizi L2 vengono spenti in sicurezza.
-

3.7.3 Comportamento di Errore

Il sistema deve gestire le eccezioni in modo robusto per garantire l'Affidabilità e l'Integrità dei Dati.

A. Errori a Livello di Transazione

Questi errori sono gestiti dal controllo atomico del sistema (Sezione 3.6):

- **Failure:** In caso di errore nel processo di Checkout (es. fallimento del PSP o esaurimento scorte), il sistema deve eseguire un **Rollback** di tutte le modifiche apportate alla base di dati per ripristinarne lo stato precedente.
- **Notifica all'Utente:** Vengono forniti **messaggi d'errore significativi** tramite l'Interfaccia Utente, in linea con i requisiti del RAD.

B. Errori a Livello di Sistema

- **Perdita di Connessione DB:** Se la connessione JDBC fallisce, il sistema L2 (Logica Applicativa) deve notificare un errore di sistema (500 Internal Server Error) finché la connessione non viene ristabilita.
- **Guasto Critico:** Per i guasti hardware o software maggiori, la ripresa del servizio dipende dai **backup periodici** e dalle procedure di ripristino stabilite per il Database MySQL.

4. Servizi dei Sottosistemi

In questa sezione vengono descritti i servizi esposti da ciascun sottosistema del Livello Logica Applicativa (L2) elencato nella Sezione 3.2. Questi servizi definiscono i confini e le interfacce per la collaborazione tra i moduli.

4.1 Sottosistema Autenticazione & Sicurezza

Servizio	Descrizione	Operazioni
Registrazione	Permette a un nuovo utente di creare un account.	RegisterUser
Login	Permette a un utente di autenticarsi nel sistema.	Login
Logout	Permette a un utente di effettuare l'uscita dal sistema.	Logout

Recupera Password	Permette a un utente di richiedere il reset della password tramite email (UC8).	ResetPassword, GenerateResetLink
Controllo Accessi	Gestisce la logica di monitoraggio e blocco degli account in caso di accessi falliti.	BlockAccount, UnblockAccount

4.2 Sottosistema Gestione Profilo Utente

Servizio	Descrizione	Operazioni
Visualizza Profilo	Permette all'utente di visualizzare i propri dati anagrafici e lo storico delle attività.	ViewProfile
Modifica Dati	Permette all'utente di modificare i propri dati anagrafici e l'indirizzo di spedizione.	UpdateUserData, UpdateAddress

4.3 Sottosistema Gestione Catalogo & Ricerca

Servizio	Descrizione	Operazioni
Ricerca e Filtraggio	Permette all'utente di cercare prodotti per nome e applicare filtri (categoria, condizione, prezzo) (UC1).	SearchProducts, ApplyFilters

Visualizza Dettagli	Permette di visualizzare la scheda dettagliata di un prodotto.	GetProductDetails
Gestione Catalogo (Admin)	Permette al Gestore di aggiungere, modificare o rimuovere prodotti (CRUD) (UC5).	AddProduct, UpdateProductDetails, RemoveProduct
Gestione Categorie	Permette al Gestore di gestire le categorie di TCG (Pokémon, Yu-Gi-Oh!, MTG).	AddCategory, UpdateCategory, RemoveCategory

4.4 Sottosistema E-commerce & Carrello

Servizio	Descrizione	Operazioni
Gestione Articoli	Permette all'utente di aggiungere e rimuovere prodotti dal carrello (UC2).	AddItemToCart, RemoveItemFromCart
Modifica Quantità	Permette all'utente di modificare la quantità di un prodotto nel carrello (UC2).	UpdateItemQuantity
Calcolo Totale	Calcola il totale provvisorio dell'ordine.	CalculateSubTotal

4.5 Sottosistema Gestione Ordini

Servizio	Descrizione	Operazioni
Inizializzazione Checkout	Crea un'istanza d'Ordine in stato provvisorio.	CreateOrderInstance
Finalizza Ordine	Finalizza la transazione, regista l'ordine definitivo e invia conferma (UC3).	FinalizeTransaction
Aggiorna Stato (Admin)	Permette al Gestore di aggiornare lo stato di spedizione dell'ordine (UC4).	UpdateOrderStatus, SetTrackingLink
Visualizza Dettagli Ordine	Permette al Gestore o all'utente di visualizzare i dettagli di un ordine specifico (UC4).	GetOrderDetails

4.6 Sottosistema Gestione Pagamenti

Servizio	Descrizione	Operazioni
Esegui Transazione	Delega l'autorizzazione dei dati sensibili al PSP esterno (simulato) (UC3).	ExecutePaymentTransaction
Registrazione Pagamento	Registra il risultato della transazione nel database dopo la conferma del PSP.	RecordPaymentSuccess, RecordPaymentFailure

Gestione Metodi Pagamento	Permette all'utente di aggiungere/rimuovere i propri metodi di pagamento (se memorizzati).	AddPaymentMethod, RemovePaymentMethod
----------------------------------	--	--

4.7 Sottosistema Gestione Recensioni

Servizio	Descrizione	Operazioni
Pubblica Recensione	Permette all'Acquirente di pubblicare una recensione e una valutazione (UC7).	PublishReview, SubmitRating
Aggiorna Rating	Ricalcola il rating medio del prodotto dopo una nuova recensione.	RecalculateAverageRating
Moderazione (Logica)	Contiene il metodo logico per approvare o rimuovere una recensione.	ModerateReview, DeleteReview

4.8 Sottosistema Gestione Amministrativa

Servizio	Descrizione	Operazioni
Gestione Utenti	Permette la visualizzazione, il ban o lo sbannaggio degli utenti.	BanUser, UnbanUser, ViewAllUsers
Gestione Ordini	Permette al Gestore di visualizzare tutti gli ordini, monitorare il ciclo di vita e aggiornarne lo stato.	GetElencoOrdini, UpdateOrderStatus (Invoca I_Ordini.UpdateStatus)
Gestione Catalogo	Permette al Gestore di modificare prezzi e quantità di prodotti.	UpdateProductPrice, UpdateProductQuantity

4.9 Sottosistema Persistenza Dati (DAO Layer)

Servizio	Descrizione	Operazioni
Recupero Dati (Read)	Permette ai sottosistemi L2 di recuperare entità dal database (es. Prodotto, Ordine).	fetchObject(ID), fetchList(QueryCriteria)
Salvataggio Dati (Create)	Permette ai sottosistemi L2 di creare nuove entità persistenti nel database.	saveObject(Object)
Aggiornamento Dati (Update)	Permette ai sottosistemi L2 di modificare lo stato delle entità	updateObject(Object)

	esistenti (es. Utente, Quantita Prodotto).	
Cancellazione Dati (Delete)	Permette ai sottosistemi L2 di rimuovere entità specifiche dal database.	deleteObject(ID)
Gestione Transazioni	Gestisce le transazioni JDBC per garantire l'atomicità delle operazioni (Commit/Rollback).	beginTx(), commitTx(), rollbackTx()