Università Degli Studi Di Salerno

Progetto di Ingegneria del software 2018/2019

System Design Document

Sommario

Componenti del team di progetto

Partecipanti	Matricola
Cosimo Bacco	**0512104516**
Michele Castellaneta	**0512104804**
Domenico Trotta	**0512104882**

1.Introduzione

Questo documento descrive gli obiettivi di design del progetto. Illustra la scomposizione del sistema in sottosistemi e definisce le strategie adottate per il loro sviluppo.

1.1 Scopo del sistema

L'obiettivo del progetto è quello di realizzare un portale web che permetta ai clienti di acquistare videogiochi per diverse piattaforme (PS4, PS3, Xbox One, Nintendo Switch, …). Lo sviluppo di un sito web di questo tipo permetterebbe di ottimizzare i tempi di servizio e di aggiornare i clienti sullo stato degli ordini da essi effettuati. Tale sito web dovrebbe consentire una comunicazione più immediata ed efficace tra gli altri attori del sistema (visitatore, gestore degli ordini, gestore del catalogo, cliente).

Il sistema deve permettere:

1. la gestione degli account
2. la gestione degli ordini
3. la gestione del carrello
4. la gestione del catalogo

1.2 Design goals

Il sistema si avvale di una struttura grafica chiara e completa, con pulsanti, finestre di dialogo, icone, form per l'immissione dei dati e finestre scorrevoli. Le informazioni presentate sullo schermo saranno in grado di indirizzare l'utente verso le funzionalità a cui desidera accedere, cercando di volta in volta di isolare soltanto le informazioni necessarie per l'esecuzione della funzione richiesta. L'utente non dovrà necessariamente effettuare operazioni che richiedono una discreta conoscenza dell'applicazione, quindi l'utilizzo del sistema sarà guidato dall'interfaccia semplice e intuitiva. Il sistema GamesHub ha come struttura centrale un database, il quale sarà periodicamente aggiornato per garantire il corretto funzionamento del sistema stesso. Il sistema proposto cerca di rispettare tutti i criteri di design sotto elencati.

1.2.1 DG_1: Performance Criteria

Il sistema dovrebbe essere usabile e leggero, cosicché, nel caso di utilizzo in contemporanea da parte di più utenti, il sistema non risulti rallentato.
GamesHub si propone di rispettare i seguenti requisiti di qualità (rispetto alle prestazioni):

• ***DG_1.1 Tempo di risposta*** : le risposte dovrebbero essere date in un tempo accettabile a seguito dell'elaborazione dell'input.
• ***DG_1.2 Throughput*** : il sistema dovrebbe completare il maggior numero di operazioni nel minor tempo possibile, per garantire una maggiore interattività con i vari utenti connessi.
• ***DG_1.3 Memoria*** : il sistema necessita di una quantità di memoria dipendente da tutti i dati che saranno memorizzati all'interno della Web-Application realizzata. La quantità di memoria che verrà utilizzata da GamesHub non può essere stimata precisamente. In principio, il sistema dovrà essere sottoposto alla memorizzazione di almeno: 30 giochi.

1.2.2 DG_2: Dependability criteria

Il sistema dovrebbe garantire il corretto svolgimento delle proprie funzioni, gestendo i vari errori logici (quelli derivanti da una negligenza da parte dell'utente), che potranno verificarsi durante l'utilizzo, ed eventuali attacchi alla sicurezza.
GamesHub si propone, quindi, di rispettare i seguenti requisiti di qualità, relativi all'affidabilità:

● ***DG_2.1 Robustezza***: GamesHub dovrebbe offrire un buon grado di robustezza agli input invalidi forniti dagli utenti. Non verranno alterati i dati contenuti nel database: nel caso in cui l'utente sottometta dati errati al sistema, questo lancerà un messaggio d'errore per avvisare lo stesso utente che i dati inseriti sono invalidi;
● ***DG_2.2 Affidabilità*** : GamesHub dovrebbe garantire il corretto svolgimento delle proprie funzionalità, producendo unicamente l'output atteso;
● ***DG_2.3 Sicurezza*** : Ogni utente potrà accedere con una login e password personale; l'accesso al sistema sarà controllato da un apposito sistema di autenticazione, che permetterà ad ogni categoria di utente di eseguire il proprio lavoro senza intaccare o modificare quello altrui.

● ***DG_2.4 Tolleranza all'errore***: GamesHub deve essere capace di operare durante condizioni d'errore. Ciò sarà reso possibile tra tutte quelle sottoparti del sistema che hanno un grado di accoppiamento basso, in modo tale che l'errore in un sottosistema non intacchi le funzionalità di un altro.

1.2.3 DG_3: Maintenance criteria

Il sistema dovrebbe garantire un'alta manutenibilità.
GamesHub si propone, quindi, di rispettare i seguenti requisiti di qualità:

• ***DG_3.1 Estendibilità*** : il sistema dovrebbe essere realizzato in modo da poter garantire l'inserimento di nuove funzionalità in maniera semplice, senza doverne modificare altre.
• ***DG_3.2 Modificabilità*** : il sistema dovrebbe essere realizzato in modo da poter garantire la modifica di funzionalità già presenti all'interno del sistema, senza doverne apportare altre, quindi il grado di entropia del sistema deve essere mantenuto il più basso possibile.
• ***DG_3.3 Tracciabilità dei requisiti*** : tramite una buona documentazione sarà possibile risalire ai corrispettivi requisiti funzionali, cui faranno riferimento le varie classi e i metodi.
• ***DG_3.4 Leggibilità*** : Il codice sarà comodo da leggere grazie ad una accurata indentazione di quest'ultimo. Inoltre, sarà facile da comprendere le diverse parti di cui il codice è composto ci saranno opportuni commenti che ne spiegano il funzionamento.

1.2.4 DG_4: End-user criteria

Per quanto riguarda gli utenti, GamesHub si propone di garantire i seguenti requisiti di qualità:
• ***DG_4.1 Utilità*** : grazie ai requisiti funzionali raccolti, GamesHub dovrebbe supportare in pieno le esigenze delle varie tipologie di utenti.
• ***DG_4.2 Usabilità*** : il sistema dovrebbe essere semplice ed intuitivo e dopo un breve utilizzo dovrà consentire all'utente di compiere le operazioni nel minor tempo possibile. Inoltre, dovrebbero essere rispettati i requisiti non funzionali di usabilità del sistema (NFR_5, NFR_6 di GamesHub_RAD)

1.3 Definizioni, acronimi e abbreviazioni

Vengono di seguito esplicitati definizioni, acronimi e abbreviazioni che verranno incontrati all'interno del documento.

Acronimo	Descrizione
RAD	Requirement Analysis Document
GUI	Graphical User Interface
SW	Software
HW	Hardware
SQL	Structured Query Language
SDD	System Design Document
DBMS	Database Management System

1.4 Riferimenti

1. GamesHub_RAD
2. System Design Document – Gestione dati persistenti

1.5 Overview

Per garantire una migliore leggibilità, il System Design Document è stato diviso in tre sezioni:

1. **Introduzione**: viene riporta una descrizione del sistema specificando il motivo per cui è stato sviluppato, le sue caratteristiche e un accenno sull'utilizzo delle sue funzionalità.

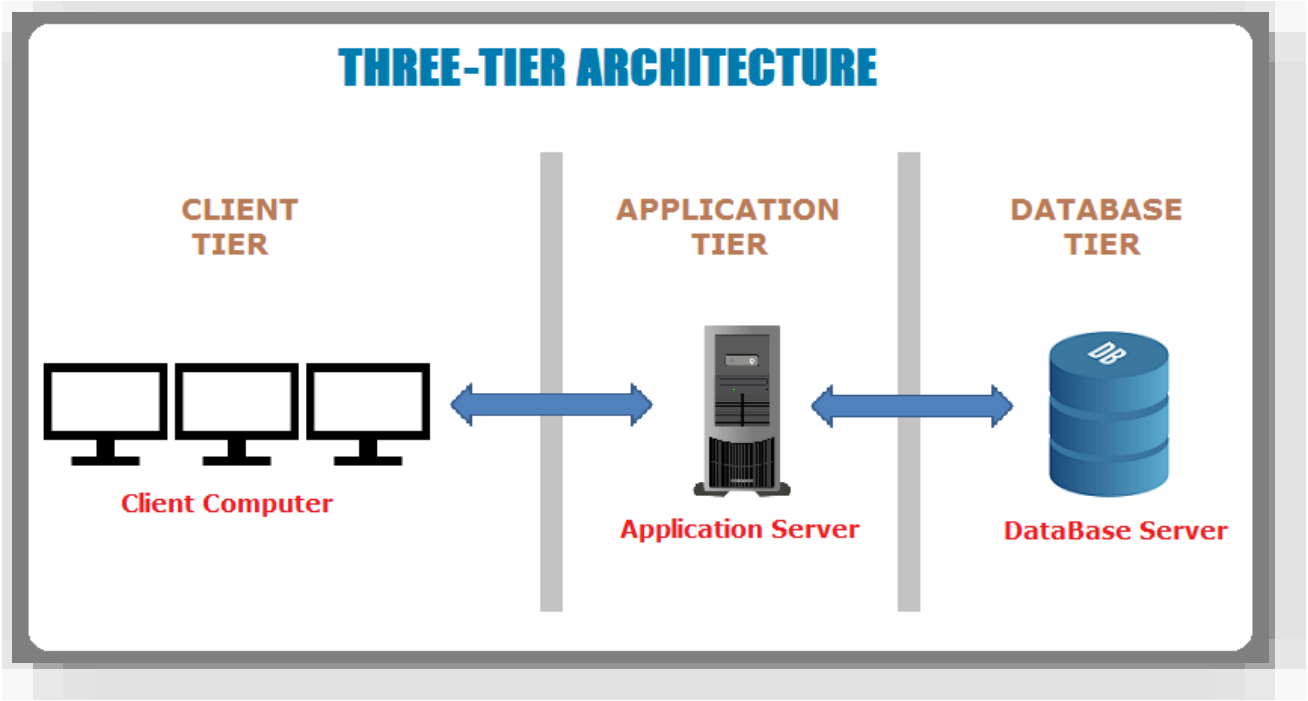
2. **Architettura Software proposta**: viene descritta l'architettura usata nel sistema, ed in particolare:

- **Decomposizione in sottosistemi**, nella quale è descritta la suddivisione del sistema in vari sottosistemi.
- **Hardware/software mapping**, in cui sono prese alcune decisioni sulle piattaforme hardware e software su cui il sistema dovrà girare e il mapping delle componenti su di esse.
- **Gestione dei dati persistenti**, in cui sono identificati gli oggetti persistenti e viene scelto il tipo di infrastruttura da usare per memorizzarli.
- **Controllo degli accessi e sicurezza**, che descrive il modello utente del sistema in termini di matrici di accesso e i problemi di sicurezza, come la scelta di un meccanismo di autenticazione.
- **Flusso di controllo globale**, esterno (tra client e server) ed interno, che descrive come il controllo globale del software è implementato, come le procedure di richiesta sono avviate e come si sincronizzano i sottosistemi.
- **Boundary Condition**, in cui sono descritte le condizioni limite del sistema.

3. **Servizi dei Sottosistemi**, dove viene riportata una descrizione dei sottosistemi individuati e i servizi offerti da ogni sottosistema in termini di operazioni.

2. Architettura Software Proposta

L'architettura scelta per il sistema GamesHub è quella Three-Tier. L'espressione architettura Three-Tier ("a tre strati") indica una particolare architettura software di tipo multi-tier per l'esecuzione di un'applicazione web. Essa prevede la suddivisione dell'applicazione in tre diversi moduli o strati dedicati rispettivamente all'interfaccia utente, alla logica funzionale e alla gestione dei dati persistenti. Tale architettura va tipicamente a mappare a livello fisico-infrastrutturale quella del sistema informatico ospitante l'applicazione da eseguire. Tali moduli interagiscono fra loro secondo le linee generali del paradigma Client-Server e utilizzando interfacce ben definite. In questo modo, ciascuno dei tre moduli può essere modificato o sostituito indipendentemente dagli altri, conferendo scalabilità e manutenibilità all'applicazione.



2.1 Overview

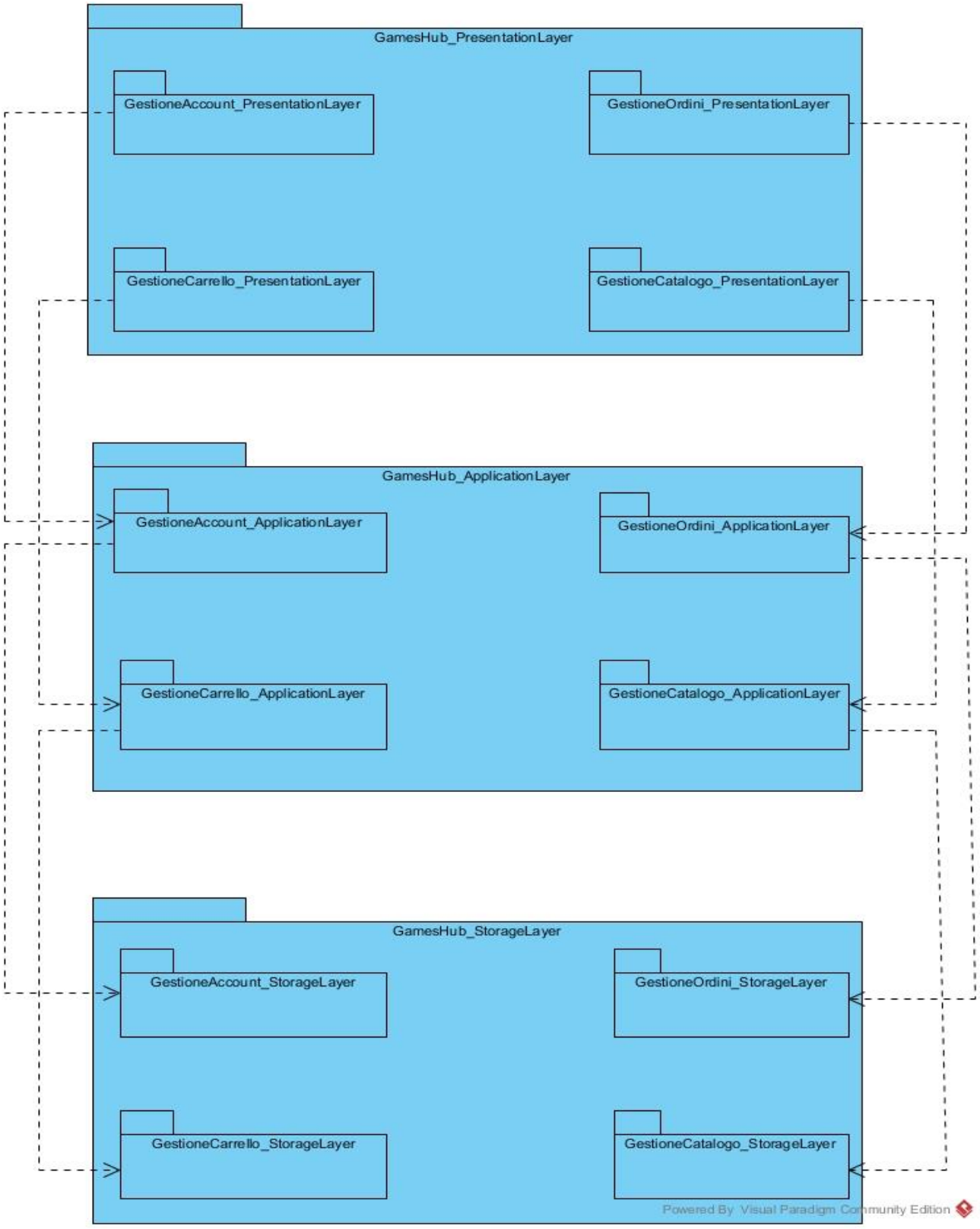
Il sistema che s'intende realizzare è, quindi, un sistema distribuito e gli utenti di GamesHub potranno interagire con esso tramite i propri terminali (client). I client, per realizzare le funzionalità richieste, dovranno comunicare con l'application server (che si occupa della logica di controllo), il quale, a sua volta, comunica con il database server, dove sono contenute tutte le informazioni.
Sul database server, risiede un DBMS che si occupa di recuperare, memorizzare ed interrogare i dati presenti nel database, elaborando, quindi, la richiesta degli utenti, inviata sottoforma di query da parte dell'application server. L'aspetto della concorrenza di accessi multipli al database sarà pertanto gestito dal DBMS stesso. Tale architettura conferisce all'intero sistema una maggiore manutenibilità e permette di gestire il problema della concorrenza degli accessi ai dati in maniera semplice ed efficace. Inoltre, grazie alla natura atomica delle transazioni col database, non si possono verificare problemi d'incoerenza nei dati (a meno di guasti ai supporti di memorizzazione secondari).

2.2 Decomposizione in sottosistemi

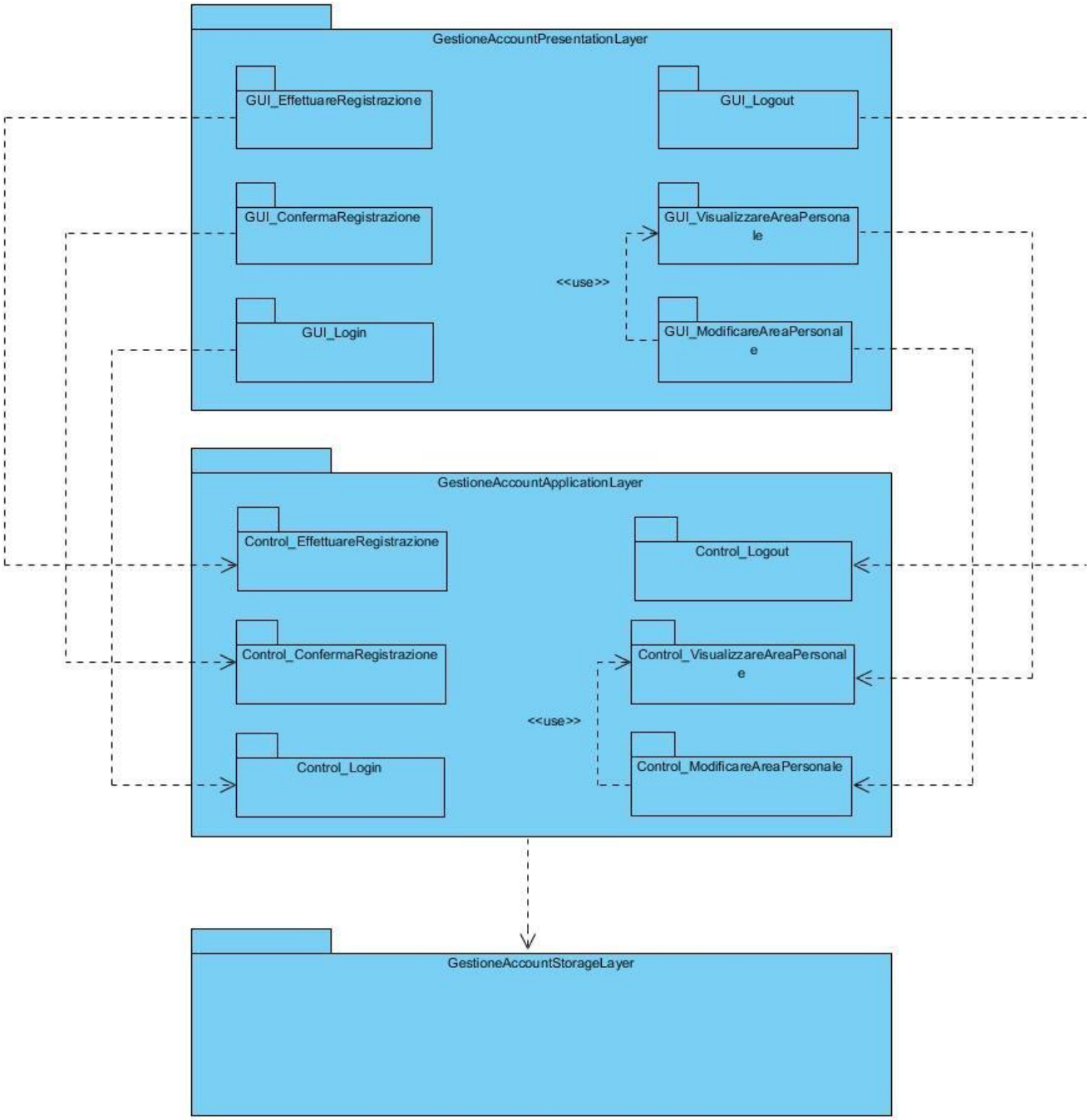
Per rendere il sistema più facile da progettare e aumentare il requisito di manutenibilità è stato deciso di decomporlo in sottosistemi, al fine di rendere la progettazione il più semplice possibile. L'architettura Three-Tier organizza i sottosistemi in tre strati: Interface Layer, ovvero tutti gli oggetti Boundary che interagiscono con l'utente. Application Logic Layer, ovvero tutti gli oggetti che riguardano il controllo e le entità del sistema, e Storage Layer, la quale funzione è quella di memorizzare, recuperare ed interrogare gli oggetti persistenti (i dati che sono presenti nei DB).
I tre livelli relativi all'architettura adottata per GamesHub, quindi, saranno:

- GamesHubPresentationLayer
- GamesHubApplicationLayer
- GamesHubStorageLayer

Diagramma generale



Sottosistema 0: Gestione Utente



GestioneAccountPresentationLayer

Include tutte le componenti dell'interfaccia grafica del sistema che offrono le funzionalità per la gestione dell'account. È riportata di seguito una breve descrizione di ognuna di esse:

- *GUI_EffettuareRegistrazione*: è un'interfaccia tramite la quale il visitatore si registra nel sistema.
- *GUI_ConfermaRegistrazione*: è un'interfaccia tramite la quale il visitatore conferma la registrazione.
- *GUI_Login*: è un'interfaccia tramite la quale visitatore effettua l'accesso nel sistema.
- *GUI_Logout*: è un'interfaccia tramite la quale l'utente registrato esce dal sistema.
- *GUI_VisualizzareAreaPersonale*: è un'interfaccia tramite la quale il cliente accede all'area personale.
- *GUI_ModificareAreaPersonale: è un'interfaccia tramite la quale il cliente modifica i propri dati nell'area personale.*

GestioneAccountApplicationLayer

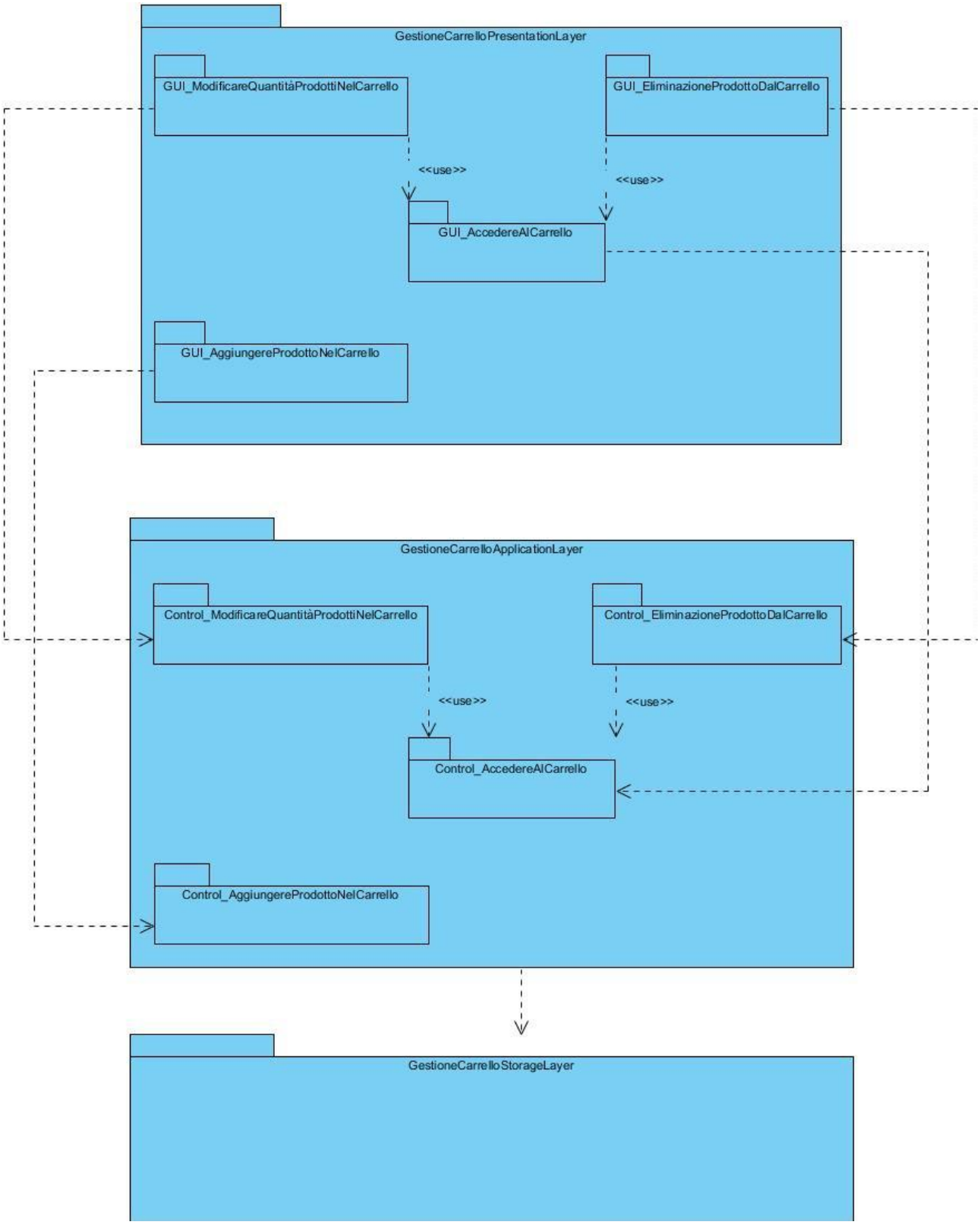
Comprende tutte le componenti logiche, responsabili del corretto funzionamento del sistema, e fa da tramite con il database per aggiornare i dati relativi alla gestione account.
È riportata di seguito una breve descrizione di ognuno di esse:

- Control_Effettuare*Registrazione*: si occupa di eseguire le operazioni necessarie alla registrazione di un visitatore del sistema.
- *Control_ConfermareRegistrazione*: si occupa di eseguire le operazione necessarie registrare il visitatore nel sistema.
- *Control_Login*: si occupa di eseguire le operazione necessarie all'autenticazione degli attori del sistema.
- *Control_Logout:* si occupa di eseguire le operazione necessarie al logout degli attori del sistema.
- *Control_VisualizzareAreaPersonale:* si occupa di eseguire le operazioni necessarie per far accedere l'utente registrato alla sua area personale.
- *Control_ModificareAreaPersonale*: si occupa di eseguire le operazione necessarie alla modifica dei dati di un utente registrato del sistema.

GestioneAccountDataLayer

Sottosistema che si occupa di rendere reperibili i dati, presenti all'interno del database, relativi alla gestione dell'account. Questo livello consente agli attori del sistema di effettuare l'autenticazione.

Sottosistema 1: Gestione Carrello



GestioneCarrelloPresentationLayer

Include tutte le componenti dell'interfaccia grafica del sistema che offrono le funzionalità per la gestione del carrello.
È riportata di seguito una breve descrizione di ognuna di esse:

- *GUI_AccedereAlCarrello*: è un'interfaccia tramite la quale il visitatore del sistema può accedere a tutte le operazioni legate alla visualizzazione del carrello.
- *GUI_AggiungereProdottoNelCarrello*: è un'interfaccia tramite la quale il visitatore può inserire un prodotto nel carrello.
- *GUI_EliminazioneProdottoDalCarrello*: è un'interfaccia tramite la quale visitatore del sistema può accedere a tutte le operazioni legate alla rimozione di un prodotto dal carrello.
- GUI_ModificareQuantitàProdottoNelCarrello: è un'interfaccia tramite la quale il visitatore del sistema può modificare la quantità di un prodotto dal carrello.

GestioneCarrelloApplicationLayer

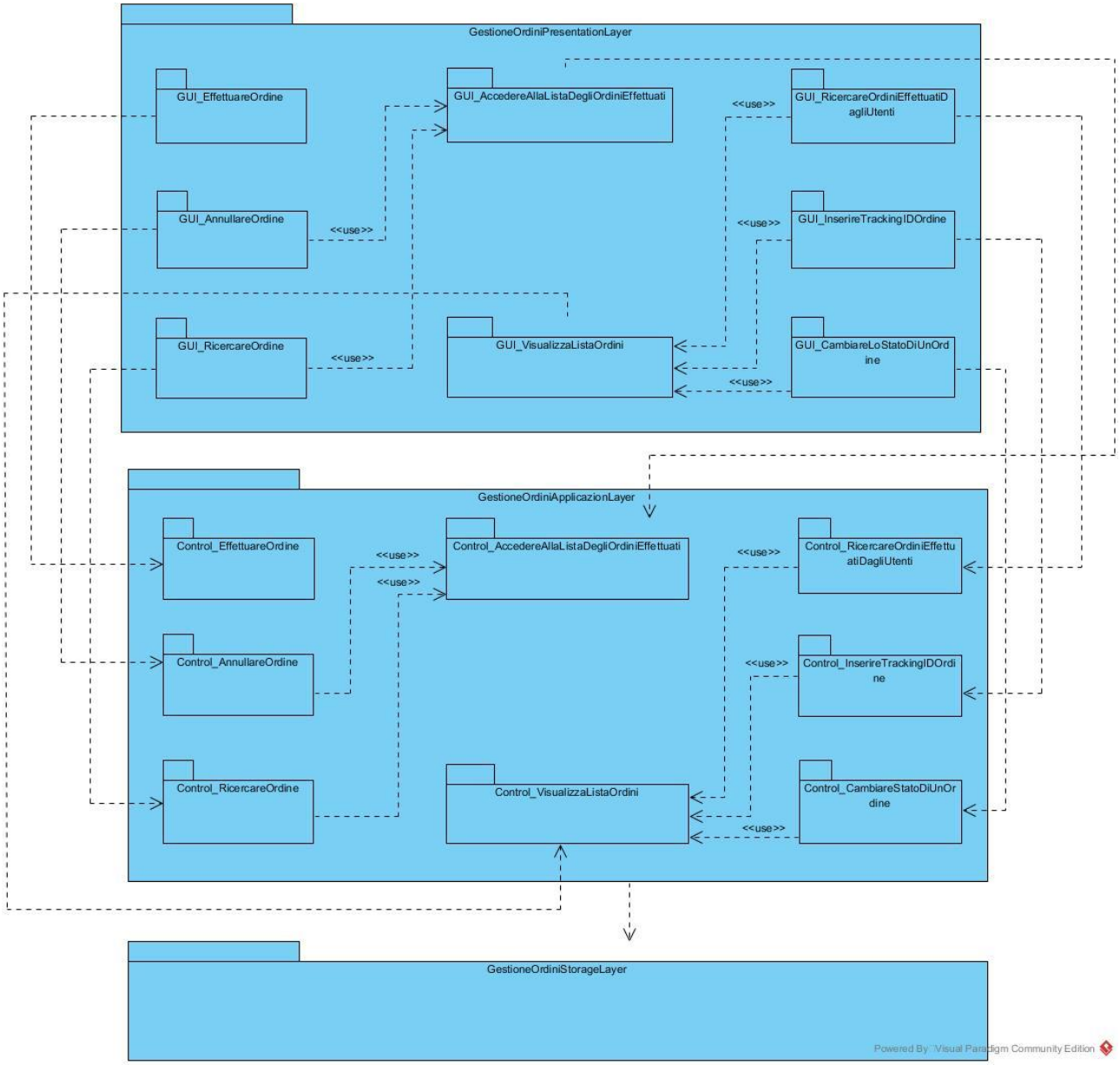
Comprende tutte le componenti logiche, responsabili del corretto funzionamento del sistema, e fa da tramite con il database per aggiornare i dati relativi alla gestione del carrello.
È riportata di seguito una breve descrizione di ognuno di esse:

- *Control_AccedereAlCarrello*: occupa di eseguire le operazioni necessarie alla visualizzazione del carrello.
- *Control_AggiungereProdottoNelCarrello*: si occupa di eseguire le operazioni necessarie all'aggiunta di un prodotto nel carrello.
- *Control_EliminazioneProdottoDalCarrello*: si occupa di eseguire le operazioni necessarie alla rimozione di un prodotto dal carrello.
- *Control_ModificareQuantitàProdottoNelCarrello*: si occupa di eseguire le operazioni necessarie per modificare la quantità di un prodotto nel carrello.

GestioneCarrelloStorageLayer

Sottosistema che si occupa di rendere reperibili i dati, presenti all'interno del database, relativi alla gestione del carrello.

Sottosistema 2: Gestione Ordini



GestioneOrdiniPresentationLayer

Include tutte le componenti dell'interfaccia grafica del sistema che offrono le funzionalità per la gestione degli ordini, accessibile da tutti gli utenti del sistema, il gestore degli ordini e il fattorino. È riportata di seguito una breve descrizione di ognuna di esse:

- *GUI_VisualizzaListaOrdini*: è un'interfaccia tramite la quale il gestore degli ordini, può accedere tutte le operazioni legate alla visualizzazione degli ordini.
- *GUI_AnnullaOrdine*: è un'interfaccia tramite la quale il cliente del sistema può accedere a tutte le operazioni legate all'annullamento dell'ordine.

- *GUI_ModificaStatoOrdine:* è un'interfaccia tramite la quale il gestore degli ordini può accedere a tutte le operazioni legate alla modifica dello stato degli ordini.
- *GUI_InserireTrackingOrdine*: è un'interfaccia tramite la quale il gestore degli ordini può inserire il tracking ID di un ordine che è stato effettuato
- *GUI_AccedereAllaListaDegliOrdiniEffettuati*: è un'interfaccia tramite la quale il cliente può accedere alla lista degli ordini che ha effettuato.
- *GUI_EffettuareOrdini*: è un'interfaccia tramite la quale il cliente può effettuare un ordine.
- *GUI_RicercareOrdine*: è un'interfaccia tramite la quale il cliente può ricercare un ordine che ha effettuato.
- *GUI_RicercareOrdiniEffettuatiDagliUtenti*: è un'interfaccia tramite la quale il gestore degli ordini può effettuare una ricerca degli ordini effettuati dagli utenti.

GestioneOrdiniApplicationLayer

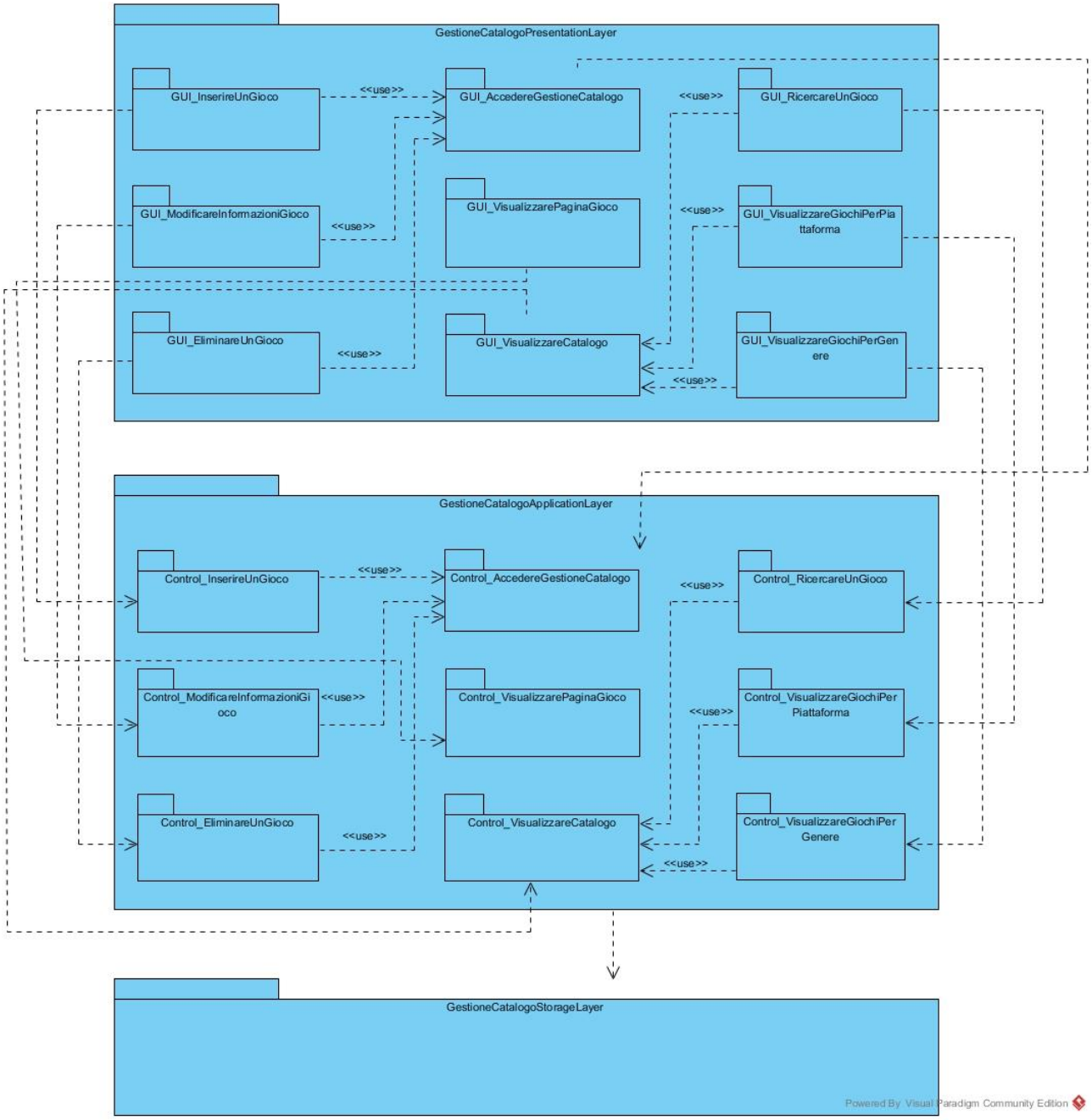
Comprende tutte le componenti logiche, responsabili del corretto funzionamento del sistema, e fa da tramite con il database per aggiornare i dati relativi alla gestione degli ordini.
È riportata di seguito una breve descrizione di ognuno di esse:

- *Control_VisualizzaListaOrdini*: si occupa di eseguire le operazioni necessarie alla
- visualizzazione degli ordini presenti all'interno del database.
- *Control_AnnullaOrdine:* si occupa di eseguire le operazioni necessarie all'annullamento degli ordini e memorizzarla all'interno del database.
- *Control_ModificaStatoOrdine:* si occupa di eseguire le operazioni necessarie alla
- modifica dello stato degli ordini e memorizzarla all'interno del database.
- *Control_InserireTrackingOrdine*: si occupa di eseguire le operazioni necessarie all'inserimento del tracking ID di un ordine e memorizzarlo all'interno del database.
- *Control_AccedereAllaListaDegliOrdiniEffettuati*: si occupa di eseguire le operazioni necessarie alla visualizzazione degli ordini effettuati da un utente presenti all'interno del database.
- *Control_EffettuareOrdini*: si occupa di eseguire le operazioni necessarie affinché venga effettuato un ordine e salvato all'interno del database.
- *Control_RicercareOrdine*: si occupa di eseguire le operazioni necessarie per effettuare una ricerca di un ordine effettuato da un cliente.
- *Control_RicercareOrdiniEffettuatiDagliUtenti*: si occupa di eseguire le operazioni necessarie per ricercare gli ordini effettuati dagli utenti.

GestioneOrdiniStorageLayer

Sottosistema che si occupa di rendere reperibili i dati, presenti all'interno del database, relativi alla gestione degli ordini. Questo livello consente di cancellare l'ordine, modificarne lo stato e il tempo di attesa nella coda e memorizzarle nel database.

Sottosistema 3: Gestione Catalogo



GestioneCatalogoPresentationLayer

Include tutte le componenti dell'interfaccia grafica del sistema che offrono le funzionalità per la gestione del catalogo.
È riportata di seguito una breve descrizione di ognuna di esse:

- *GUI_InserireUnGioco*: è un'interfaccia tramite la quale il gestore del catalogo può inserire un gioco nel catalogo.
- *GUI_ModificareInformazioniGioco*: è un'interfaccia tramite la quale il gestore del catalogo può modificare le informazioni di un gioco presente nel catalogo.

- *GUI_EliminareUnGioco:* è un'interfaccia tramite la quale il gestore del catalogo può accedere a tutte le operazioni legate all'eliminazione di un gioco dal catalogo.
- *GUI_AccedereGestioneCatalogo:* è un'interfaccia tramite la quale il gestore del catalogo può accede alla gestione del catalogo.
- *GUI_VisualizzareCatalogo:* è un'interfaccia tramite la quale il visitatore può accedere alla lista dei giochi presenti nel catalogo.
- *GUI_RicercareUnGioco*: è un'interfaccia tramite la quale il visitatore può ricercare un gioco all'interno del catalogo.
- *GUI_VisualizzareGiochiPerPiattaforma*: è un'interfaccia tramite la quale il visitatore può visualizzare i giochi per una piattaforma che ha scelto.
- *GUI_VisualizzareGiochiPerGenere: è un'interfaccia tramite la quale il visitatore può visualizzare i giochi per un genere che ha scelto.*
- *GUI_VisualizzarePaginaGioco: è un'interfaccia tramite la quale il visitatore può visualizzare la pagina del gioco che ha selezionato.*

GestioneCatalogoApplicationLayer

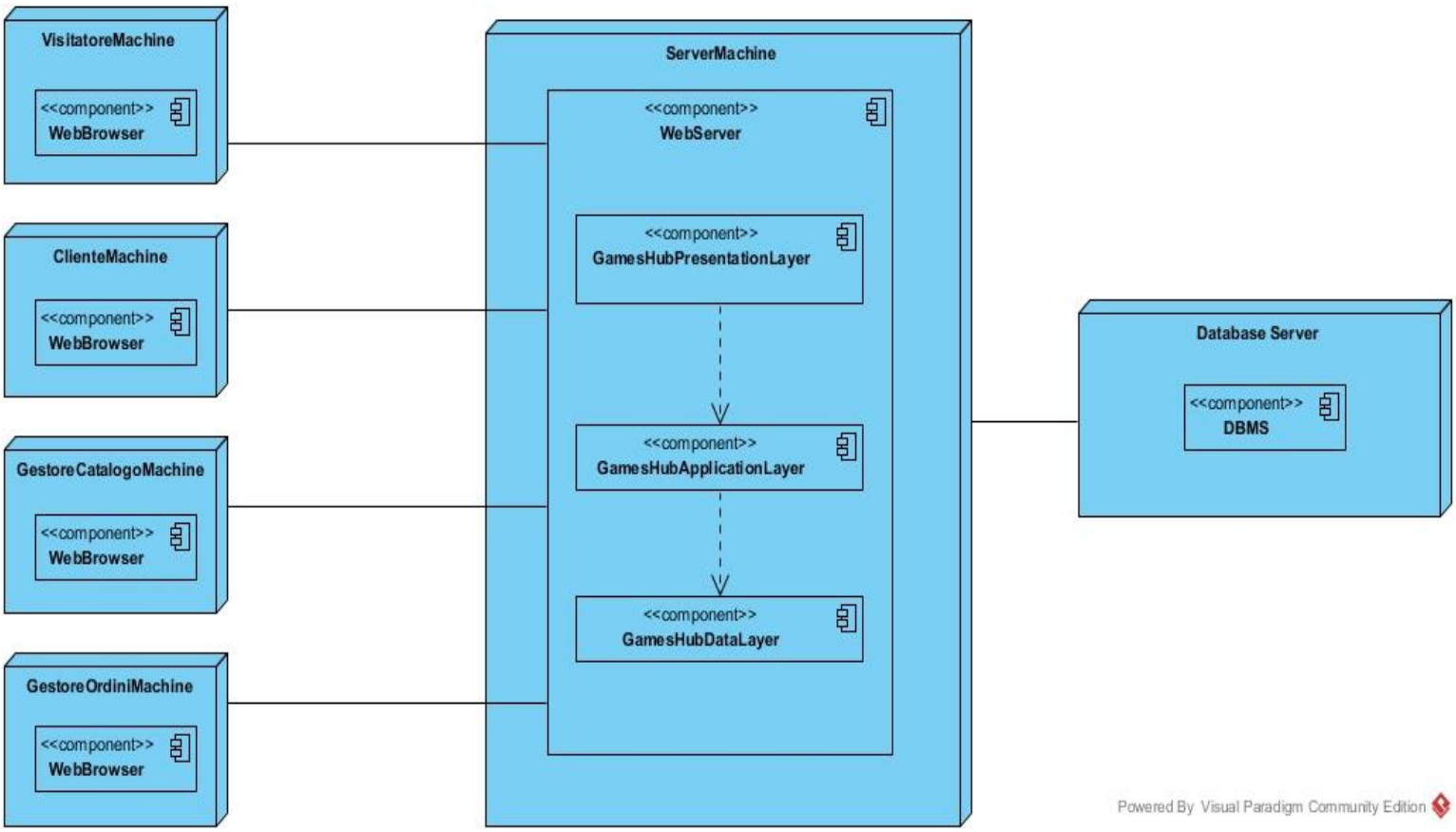
Comprende tutte le componenti logiche, responsabili del corretto funzionamento del sistema, e fa da tramite con il database per aggiornare i dati relativi alla gestione del catalogo.
È riportata di seguito una breve descrizione di ognuno di esse:

- *Control_InserireUnGioco*: si occupa di eseguire le operazioni necessarie affinché un nuovo gioco venga salvato nel database.
- *Control_ModificareInformazioniGioco*: si occupa di eseguire le operazioni necessarie affinché vengano aggiornate le informazioni di un gioco presente nel database.
- *Control_EliminareUnGioco:* si occupa di eseguire le operazioni necessarie affinché venga eliminato un gioco presente nel database.
- *Control_AccedereGestioneCatalogo:* si occupa di eseguire le operazioni necessarie affinché il gestore del catalogo possa visualizzare la gestione del catalogo.
- *Control_VisualizzareCatalogo:* si occupa di eseguire le operazioni necessarie affinché il visitatore possa visualizzare la lista dei giochi presenti nel catalogo.
- *Control_RicercareUnGioco*: si occupa di eseguire le operazioni necessarie per ricercare un gioco presente nel catalogo.
- *Control_VisualizzareGiochiPerPiattaforma*: si occupa di eseguire le operazioni necessarie affinché venga visualizzata una lista di giochi per una piattaforma scelta dal visitatore.
- *Control_VisualizzareGiochiPerGenere:* si occupa di eseguire le operazioni necessarie affinché venga visualizzata una lista di giochi per un genere scelto dal visitatore.
- *Control_VisualizzarePaginaGioco:* si occupa di eseguire le operazioni necessarie affinché venga visualizzata la pagina del gioco scelto dal visitatore.

GestioneCatalogoStorageLayer

Sottosistema che si occupa di rendere reperibili i dati, presenti all'interno del database, relativi alla gestione del catalogo.

2.3 Mapping Hardware\Software



Web Server
Il server utilizzato è Apache.
Interface layer
L'utente utilizza il sistema mediante un Browser installato all'interno del suo calcolatore (ad es. Opera, Firefox, Chrome).
Application Logic layer
Il sistema, e quindi le funzionalità, sono implementate in linguaggio Java.
Storage layer
Rappresenta il collegamento con il server da parte del sistema e si occupa di tutte le richieste di accesso e modifiche sui dati permanenti presenti nel database.
Database Server
Il DBMS usato è MySQL il quale presenta molte API che permettono l'interazione tra sistema e database.

2.4 Gestione dei dati persistenti

Il problema della persistenza dei dati nasce dalla necessità di rendere permanenti alcune informazioni all'interno di un sistema anche quando questo è spento. Molte volte capita di avere dei programmi con un gran numero di dati da gestire, dati che non possono essere reinseriti dopo un blackout o dopo lo spegnimento della macchina; si pensi ad un archivio di mille utenti contenente tutti i dati anagrafici. Sarebbe impensabile il reinserimento degli stessi ogni volta che la macchina sul quale risiedono i dati si spegne. Bisogna, quindi, trovare il modo per poter tener traccia dei dati utili anche quando l'applicazione che li ha utilizzati e/o creati ha
smesso di funzionare. I dati persistenti sono sempre lì e vengono cancellati solo se lo si richiede espressamente. Nel caso specifico, in previsione del gran numero di accessi al database e alla necessità di gestire anche più query contemporaneamente, si è optati per un DBMS relazionale, MySQL. La sigla SQL presente all'interno del nome sta a indicare che si tratta di un DBMS in grado di interpretare le istruzioni del linguaggio SQL (utile per le query). MySQL permette la creazione di "database relazionali" ossia consente la conservazione dei dati in tabelle separate anziché in un'unica grande entità. Questa sua particolare caratteristica consente di raggiungere una buona flessibilità e velocità di accesso ai dati ed una maggior modellazione delle basi dati. Semplicità d'uso, robustezza e velocità di esecuzione sono le caratteristiche principali di questo DBMS.
La sua struttura multi-thread consta di un thread fisso che controlla le connessioni in ingresso e un thread attivo per ogni connessione. I vari client non devono aspettare che le query di altri siano soddisfatte, possono lavorare simultaneamente. Il thread che controlla le connessioni
impedisce che due thread scrivano sulla stessa tabella nello stesso momento. In ogni tabella all'interno del nostro database sarà possibile accedere ai dati per compiere operazioni quali inserimento, modifica, cancellazione o semplicemente consultazione.

Riferimento System Design Document – Gestione dati persistenti.

2.5 Controllo degli accessi e sicurezza

GamesHub sarà realizzato in modo da interagire con diverse tipologie di utenti: visitatore, cliente, gestore degli ordini, gestore del catalogo.
Ognuno di essi potrà accedere esclusivamente ad un determinato tipo di funzionalità e dati, ovviamente previo autenticazione. I dati per effettuare quest'ultima saranno concordati nella fase di creazione dell'account, per tutti gli utenti tranne che per gli admin (Gestore degli ordini, Gestore del catalogo), i cui dati saranno impostati di default all'interno del database, non appena creato il sistema.

● Il visitatore può creare il suo account, visualizzare la lista dei giochi, ricercare un gioco, visualizzare un gioco, visualizzare il suo carrello, rimuovere un gioco dal carrello, svuotare il carrello.

● Il cliente può effettuare il login e il logout e modificare i dati del proprio account, visualizzare la lista dei giochi , ricercare un gioco, visualizzare un gioco, acquistare giochi, visualizzare il carrello, rimuovere un gioco dal carrello, svuotare il carrello, visualizzare l'ordine e annullare l'ordine, visualizzare i dettagli dell'ordine ,ricercare un ordine.

● Il Gestore degli ordini può effettuare il login e il logout del proprio account, visualizzare gli ordini e modificare lo stato dell'ordine in accettato, in preparazione o spedito, aggiungere un track-id.

● Il Gestore del catalogo può effettuare il login e il logout del proprio account, visualizzare la lista dei giochi, rimuovere un gioco dalla lista, modificare i giochi presenti nella lista, inserire un nuovo gioco e ricercare un gioco.

Per rappresentare in modo schematico e permettere una lettura immediata delle operazioni consentite agli attori sulle diverse entità del nostro sistema, utilizzeremo la matrice degli accessi, qui di seguito presentata.

2.5.1 Matrice di accesso

Attore	Oggetti				
	Utente	Ordine	Carrello	Carta	Gioco
Visitatore	registraAccount()		visualizzaCarrello() aggiungiGioco() rimuoviGioco()		visualizzaGioco() ricercaGioco() visualizzaCatalogo()
Cliente	login() logout() visualizzaDati() modificaDati()	creaOrdine() annullaOrdine() visualizzaOrdine()	visualizzaCarrello() aggiungiGioco() rimuoviGioco()	aggiungiCarta() eliminaCarta()	visualizzaGioco() ricercaGioco() visualizzaCatalogo()
Gestore catalogo	login() logout()				visualizzaCatalogo() aggiungiGioco() modificaGioco() rimuoviGioco() ricercaGioco()
Gestore ordini	login() logout()	visulizzaOrdini() modificaOrdini() ricercaOrdine()			

2.5.2 Sicurezza

La sicurezza del sistema è garantita dall'obbligo, di tutte le utenze che lo utilizzano, di autenticarsi, infatti GamesHub chiederà come forma di autenticazione l'immissione di un'username e di una password. Tali credenziali dovranno esser inserite ogni volta che si desidera utilizzare il sistema; tale sessione terminerà quando l'utente richiederà esplicitamente di effettuare logout. Nel caso l'accesso al sistema non abbia successo, GamesHub mostrerà una schermata in cui sarà specificato se c'è stato un errore di inserimento della password, o se l'username non esiste all'interno del database, consentendo, successivamente, all'utente di inserire le credenziali per effettuare un nuovo tentativo. Nello specifico la password deve avere una lunghezza minima di 6 caratteri alfanumerici e deve contenere almeno un carattere maiuscolo, uno minuscolo e un numero. In seguito all'autenticazione il GamesHub mostrerà diverse "viste" dello stesso sistema a seconda dell'utente, ovvero sarà mostrata una schermata contenente le sole funzionalità a cui una determinata tipologia di può accedere.

Per quanto riguarda la registrazione di un utente al sistema, GamesHub prevede un verifica in due passaggi. Dopo che i dati inseriti sono stati validati dal sistema, l'utente riceverà una e-mail contenente un link che gli permetterà di completare la sua registrazione a GamesHub.
Inoltre, il sistema cripterà password di ogni utente e anche eventuali dati sensibili delle carte inserite.

2.6 Flusso di controllo globale

2.6.1 Flusso di Controllo esterno

Il sistema GamesHub è principalmente un sistema event-driven, poichè tutte le funzionalità sono avviate in seguito ad un comando dell'utente. Il *WebServer* aspetta di ricevere richieste dal *WebBrowser.* L'utente, tramite la pressione di un pulsante o l'apertura di un menù scatena un evento, che sarà gestito correttamente dal giusto handler, il quale, a sua volta, indirizzerà il controllo del flusso del sistema alla classe corretta del sottosistema che si occupa della logica di controllo.

Alla ricezione di una nuova richiesta, il *WebServer* la processa e la indirizza alla servlet o JSP appropriata, risultando così in un flusso di controllo event-based. Il *WebServer* alloca un nuovo thread per ogni richiesta, permettendo la gestione contemporanea di più richieste. Ciò permette al sistema di essere più responsive, abilitando il *WebServer* a rispondere a singole richieste da parte del *WebBrowser* prima che altre richieste vengano completamente processate, e può incrementare il throughput abilitando l'elaborazione di una richiesta mentre un'altra sta attendendo la risposta dal database.

2.6.2 Controllo delle Concorrenza

Più utenti possono accedere contemporaneamente all'application server. Si rende necessaria, per il corretto funzionamento del sistema, la creazione di un nuovo thread per ogni utente che richiede un servizio. Sarà poi compito del DBMS occuparsi della concorrenza degli accessi al database. È necessario, a questo proposito, valutare le condizioni di concorrenza e prendere opportune precauzioni dove necessario. Per facilitare la manutenzione del sistema e gestire nella maniera più efficace la coordinazione della concorrenza degli accessi al database, sarà utilizzato sul database server un DBMS: il sistema permette l'accesso concorrente all'application server, ma la gestione della priorità d'accesso al database sarà gestita dal DBMS.

2.7 Boundary conditions

Le condizioni limite riguardano l'accensione, la gestione e lo spegnimento del sistema per quanto riguarda il lato Server.

UC: Start Server

Nome caso d'uso	**Start Server**
Attori partecipanti	Iniziato da: Admin
Flusso di eventi	1. L' Admin avvia il server attraverso la funzione "startServer". 2. Il sistema avvia il server, mettendo a disposizione dei client le varie funzionalità offerte.
Condizioni d'entrata	L'admin avvia il server.
Condizioni d'uscita	Il server è avviato con successo e le varie funzionalità sono messe a disposizione dei client in remoto
Eccezioni	Failure

UC: Shutdown Server

Nome caso d'uso	**Shutdown Server**
Attori partecipanti	Iniziato da: Admin
Flusso di eventi	1. L'Admin arresta il server attraverso la funzione "ShutdownServer." 2. Il sistema provvede ad arrestare il sistema, rimuovendo prima i servizi forniti dallo stesso server ai client.
Condizioni d'entrata	Il server è avviato.
Condizioni d'uscita	Il server è arrestato.

Eccezioni	Failure

UC: Configure Server

Nome caso d'uso	**Configure Server**
Attori partecipanti	Iniziato da: Admin
Flusso di eventi	1. L'Admin avvia la configurazione del server, attivando la funzionalità "ConfigureServer". 2. GamesHub visualizza una schermata con il pannello di controllo del server. 3. L'Admin controlla se si sono verificati errori nel sistema.
Condizioni d'entrata	Il server è avviato.
Condizioni d'uscita	L'Admin ha effettuato la configurazione del server e la correzione di un eventuale errore
Eccezioni	Failure

UC: Failure

Nome caso d'uso	**Failure**
Attori partecipanti	Iniziato da: Admin
Flusso di eventi	1. Il sistema rileva un crash nel sistema e presenta una schermata dove è possibile ripristinare il sistema allo stato precedente.
Condizioni d'entrata	Si verifica un crash nel sistema.
Condizioni d'uscita	L'Admin visualizza una notifica relativa al crash di sistema.

3. Servizi dei sottosistemi

3.1 Gestione utenti

Sottosistema	**Gestione utenti**
Descrizione	Sottosistema che gestisce la registrazione di un utente, l'autenticazione di tutti e tre gli utenti e le operazioni necessarie alla loro gestione.
Servizi offerti	
Servizio	**Descrizione**
Registra utente	Permette di inserire il profilo di uno studente nel database.
Login	Permette ad un utente di poter effettuare l'accesso al sistema.
Logout	Permette ad un utente di uscire dal sistema.
Visualizza utente	Permette di visualizzare a video il profilo di un utente.
Modifica Utente	Permette di raccogliere le informazioni di un utente presente nel database, modificarle e aggiornarle all'interno dello stesso.
Conferma e-mail	Permette di confermare l'e-mail di un utente.

3.2 Gestione carrello

Sottosistema	**Gestione carrello**
Descrizione	Sottosistema che gestisce le operazioni riguardanti il carrello come l'aggiunta di un prodotto al carrello, rimozione di un prodotto dal carrello e la modifica della quantità di un prodotto.
Servizi offerti	
Servizio	**Descrizione**
Accesso al carrello	Permette ad un utente di accedere al carrello.
Aggiungere prodotto	Permette ad un utente di poter aggiungere un prodotto al carrello
Rimuovere prodotto	Permette ad un utente di poter rimuovere un prodotto dal carrello.
Modifica quantità	Permette all'utente di poter modificare la quantità selezionata di un prodotto.

3.3 Gestione ordini

Sottosistema	**Gestione ordini**
Descrizione	Sottosistema che gestisce la visualizzazione degli ordini, la ricerca degli ordini e le operazioni necessarie alla loro gestione.
Servizi offerti	
Servizio	**Descrizione**
Visualizza lista ordini	Permette di visualizzare la lista degli ordini.
Annullare ordine	Permette ad un utente di poter annullare un ordine.
Modifica stato ordine	Permette di modificare lo stato di un ordine.
Inserimento tracking id	Permette di inserire il tracking id di un ordine e memorizzarlo all'interno del database.
Accesso alla lista ordini	Permette ad un utente di visualizzare la lista degli ordini effettuati.
Effettuare ordine	Permette ad un utente di creare un ordine e salvarlo nel database.
Ricercare ordine	Permette ad un utente di ricercare un ordine.

3.4 Gestione catalogo

Sottosistema	**Gestione catalogo**
Descrizione	Sottosistema che gestisce la visualizzazione dei giochi, la ricerca dei giochi e le operazioni necessarie alla loro gestione.
Servizi offerti	
Servizio	**Descrizione**
Visualizzare catalogo	Permette di visualizzare la lista dei giochi presenti nel database.
Visualizzare pagina di un gioco	Permette di visualizzare la pagina di un gioco
Inserire gioco	Permette di inserire un nuovo gioco nel database.
Eliminare gioco	Permette di eliminare un gioco dal database.
Accesso alla gestione del catalogo	Permette di gestire il catalogo dei giochi.
Ricercare gioco	Permette di ricercare un gioco.
Visualizzare giochi per piattaforma	Permette di visualizzare una lista di giochi di una determinata piattaforma selezionata dall'utente.
Visualizzare giochi per genere	Permette di visualizzare una lista di giochi di un determinato genere selezionato dall'utente.

Glossario

Termini	Descrizione
DBMS	Database Management System
DG	**Design-goals**