System Design Document

GameSquare

Sommario

[1. Introduzione 3](#_Toc63593596)

[1.1 Obiettivi del Sistema 3](#_Toc63593597)

[1.2 Design Goals 3](#_Toc63593598)

[1.2.1 Criteri di performance: 3](#_Toc63593599)

[1.2.2 Criteri di affidabilità: 3](#_Toc63593600)

[1.2.3 Criteri di costi: 3](#_Toc63593601)

[1.2.4 Criteri di manutenzione: 3](#_Toc63593602)

[1.2.5 Criteri di end user: 4](#_Toc63593603)

[1.3 Design Trade-offs 4](#_Toc63593604)

[1.3.1 Memoria vs Estendibilità: 4](#_Toc63593605)

[1.4 Definizioni, acronimi e abbreviazioni 4](#_Toc63593606)

[1.5 Riferimenti 4](#_Toc63593607)

[1.6 Panoramica 4](#_Toc63593608)

[Capitolo 1: 4](#_Toc63593609)

[Capitolo 2: 4](#_Toc63593610)

[Capitolo 3: 4](#_Toc63593611)

[Capitolo 4: 4](#_Toc63593612)

[2. Architettura del sistema corrente 4](#_Toc63593613)

[3. Architettura del sistema proposto 5](#_Toc63593614)

[3.1 Decomposizione in layer 5](#_Toc63593615)

[3.2 Decomposizione in sottosistemi 5](#_Toc63593616)

1. Introduzione

## Obiettivi del Sistema

GameSquare nasce come strumento di supporto all’utenza del medium videoludico. Questa ormai enorme comunità è molto presente su Internet ma non dispone di una piattaforma specializzata e centralizzata dove riunirsi per scambiare opinioni, chiedere consiglio o semplicemente tener traccia dei prodotti consumati e di quelli a cui si è interessati.  
GameSquare può essere usato da chiunque senza bisogno di registrazione, ma l’utilizzo delle sue piene funzionalità è rivolto alle seguenti categorie:  
- Videogiocatori di qualsiasi età e grado di esperienza.  
- Sviluppatori di videogiochi volenterosi di interfacciarsi con la propria clientela.

## Design Goals

### 1.2.1 Criteri di performance:

* *Tempo di risposta:* Il sito web deve fornire un tempo di risposta abbastanza basso da poter permettere una fluida navigazione a tutti gli utenti.
* *Memoria:* La memoria fornita dal DB dovrà essere scalabile per accomodare influssi di nuovi utenti e creazione di un numero qualsiasi di nuove entry.

### 1.2.2 Criteri di affidabilità:

* *Disponibilità:* Il sito web dovrà essere online 24/7.
* *Sicurezza e privacy:* Il sito richiederà e-mail e password per la registrazione e successive autenticazioni. Non sarà richiesto nessun tipo di dato sensibile agli utenti.
* *Robustezza:* Input non validi verranno prontamente individuati e bloccati dal sistema, e l’utente verrà notificato dell’errore.
* *Tolleranza ai malfunzionamenti:* Il sistema potrebbe essere soggetto a guasti di vario genere. Fallimenti relativi al database verranno prontamente riparati grazie a backup automatici, mentre fallimenti relativi alla raggiungibilità del sito saranno notificati agli utenti e la loro risoluzione sarà a carica del servizio di hosting.

### 1.2.3 Criteri di costi:

* *Sviluppo:* È stimato un costo complessivo di sviluppo di circa 150 ore (50 ore per ogni Project Member)
* *Mantenimento:* Sono previsti costi monetari per il mantenimento del servizio di hosting e del Database.

### 1.2.4 Criteri di manutenzione:

* *Estendibilità:* Sarà possibile aggiungere nuove funzionalità al sistema, in base alle esigenze dell’utenza o allo sviluppo del mercato videoludico.
* *Portabilità:* La portabilità sarà garantita in quanto l’interazione con il sistema avverrà tramite un browser web.
* *Adattabilità:* Nel caso se ne presentasse la necessità, il sistema potrà essere adattato per accomodare diversi tipi di medium ludici, come giochi da tavolo, visual novels, etc.

### 1.2.5 Criteri end user:

* *Usabilità:* Il sistema è di facile utilizzo e fornisce una guida per l’utente neofita.
* *Utilità:* Il sistema centralizza funzionalità previamente presenti su Internet in maniera frammentata e crea un metodo di discussione immediato e trasparente fra sviluppatore e consumatore.

## Design Trade-offs

### Memoria vs Estendibilità:

Le funzionalità del sistema sono fortemente basate su registrazione di nuove entry e catalogazione di quelle esistenti. Di conseguenza verrà data priorità alla scalabilità e all’affidabilità della memoria.

## Definizioni, acronimi e abbreviazioni

## Riferimenti

## Panoramica

### Capitolo 1:

Introduzione, presentazione degli obiettivi di sistema, design goals e trade-offs.

### Capitolo 2:

Presenta le funzionalità del sistema corrente (non pervenuto).

### Capitolo 3:

Presentazione dell’architettura del sistema proposto, decomposizione dello stesso in sottosistemi, hardware/software mapping, gestione dei dati persistenti, sicurezza e controllo degli accessi, condizioni limite.

### Capitolo 4:

Presentazione dei sottosistemi e dei loro servizi.

2. Architettura del sistema corrente

GameSquare punta a centralizzare alcune funzioni già presenti su altri siti come forum e social network, senza tuttavia andare a rimpiazzare un sistema già esistente.

3. Architettura del sistema proposto

## 3.1 Panoramica

Il sistema che andiamo a proporre è un’applicazione web, e il suo obiettivo, come già precedentemente accennato, è di fornire una piattaforma di discussione riguardante il mondo videoludico.

Il sistema prevede funzioni limitate per utenti non registrati, funzionalità più avanzate per utenti registrati e privilegi di gestione del catalogo e moderazione del sito per gli sviluppatori o i moderatori.

Verrà utilizzato il pattern MVC (Model-View-Controller) al fine di ridurre l’accoppiamento fra presentazione dei dati e logica di business.

## 3.2 Decomposizione in sottosistemi

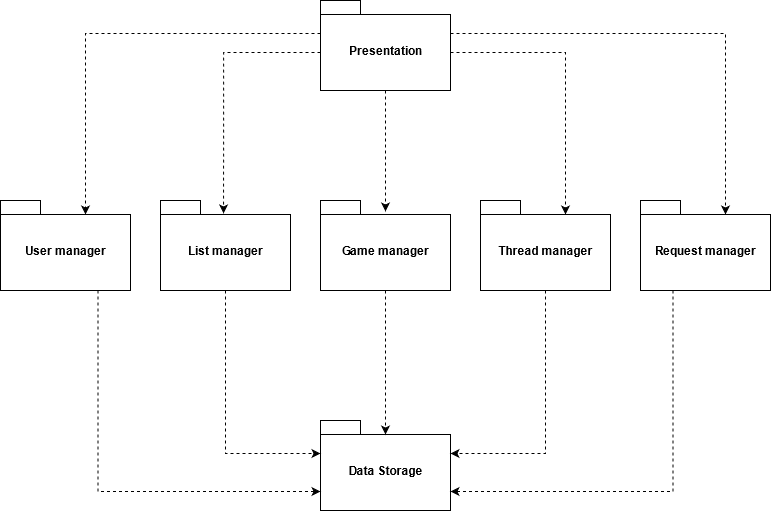
### 3.2.1 Decomposizione in Layer:

Il Sistema, dopo la sua decomposizione, è previsto di tre layer che lo compongono:

* **Presentation:** gestisce la visualizzazione dei dati (giochi, profili e discussioni del sistema) e la rappresentazione dei controlli necessari per le interfacce con i vari tipi di utenti.
* **Business layer:** raccoglie gli input dei vari utenti dalla Presentation, definisce e gestisce le entità principali e le loro relazioni e logiche applicative.
* **Data Storage:** tutto quello che riguarda la persistenza dei dati.

**3.2.2 Decomposizione in sottosistemi**

Il Sistema è così decomposto in sette sottosistemi che si occupano di gestire funzionalità differenti:



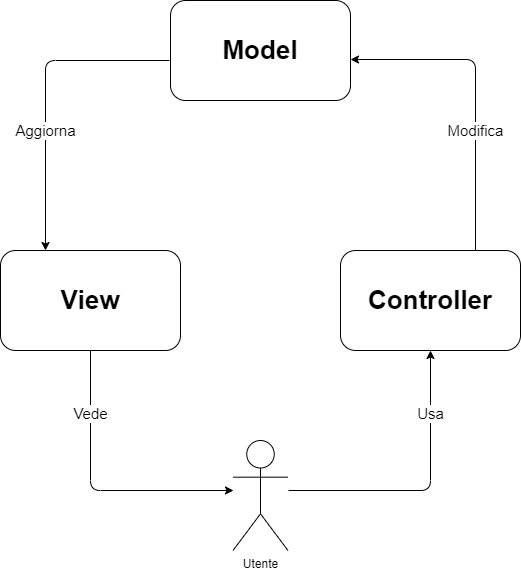
Il layer di Presentation è composto da un unico sottosistema che gestisce l’interfaccia grafica con l’utente.

Il layer di Business è così composto:

* User Manager si occupa della registrazione di nuovi utenti nel sistema; login e modifiche del profilo.
* List Manager gestisce le varie liste personali di ogni utente e di tutte le funzionalità ad esso legate.
* Game Manager riguarda l’aggiunta e la modifica dei giochi presenti nel catalogo.
* Thread Manager si occupa dell’aspetto social del sistema, gestendo discussioni relative ai giochi, commenti e like.
* Request Manager gestisce tutte le richieste fatte da un utente che arrivano all’admin, che sia di aggiunta di un gioco, di un thread o di una segnalazione.

Infine, il layer del Data Storage è il sottosistema che immagazzina e gestisce i dati persistenti.

## 3.3 Mapping hardware/software



### [da completare]

## 3.4 Gestione dei dati persistenti

Poiché il sistema necessita di un ampio spazio di memorizzazione, è stato scelto l’utilizzo di un database relazionale per la gestione dei dati persistenti, anche per garantire l’accesso concorrente ai dati da parte di diversi utenti. Per gestire il database si utilizza il DBMS MySQL.