|  |
| --- |
| **Università degli Studi di Salerno Corso di Ingegneria del Software** |

**RAAF-GAMING**  
**System Design Document**  
**Versione 1.0**

Data: 11/11/2021

**Partecipanti:**

|  |  |
| --- | --- |
| Nome | Matricola |
| Rocco Iuliano | 0512106804 |
| Francesco Peluso | 0512107194 |
| Antonio De Lucia | 0512109225 |
| Antonio Maddaloni | 0512107890 |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Descrizione | Autore |
| 11/11/2021 | 1.0 | Prima stesura del documento | Membri del team |
|  |  |  |  |

Indice

1. Introduzione
   1. Obiettivo del sistema
   2. Design Goals
   3. Definizioni, acronimi e abbreviazioni
   4. Riferimenti
2. Architettura del Sistema Corrente
3. Architettura del sistema proposto
   1. Overview
   2. Decomposizione in sottosistemi
   3. Hardware/Software Mapping
   4. Persistent and Data Managment
   5. Access control and Security
   6. Global Software Control
   7. Boundary Conditions
4. Subsystem services
5. Introduzione
   1. Obiettivo del sistema

Lo sviluppo del web ha reso popolare la vendita e l'acquisto di oggetti tramite negozi online, il cosiddetto e-commerce, che durante gli ultimi anni ha cambiato radicalmente il modo in cui le persone fanno shopping. Questo sviluppo ha coinvolto anche il mondo videoludico e ha fatto sì che la richiesta d’acquisto di prodotti videoludici online crescesse in modo esponenziale. Per questo RAAF-GAMING ha l’obiettivo di proporsi in questo settore come un e-commerce di riferimento per la vendita di questi prodotti a prezzo scontato. Inoltre, al giorno d’oggi c’è una grande affluenza sui siti web che offrono la possibilità di acquistare noti videogiochi a prezzi scontati ma è anche molto richiesto una piattaforma web per la vendita di console e abbonamenti ad un prezzo ragionevole, proprio per questo RAAF-GAMING offre la possibilità di acquistare oltre ai videogiochi, le console e gli abbonamenti a prezzi scontati.

* 1. **Design Goals**

**Affidabilità**:

* *Messaggi di errore:*

Il sistema deve rispondere attraverso messaggi di errore in presenza di input non validi inseriti dall’utente. **Priorità=**alta.

* *Sicurezza della memorizzazione dati:*

Il sistema devefar sì che la password sia criptata. **Priorità=**alta.

* *Sicurezza nell’estrapolazione dei dati:*

Il sistema deve estrapolare i dati persistenti in modo sicuro.  Ovvero deve poter prevenire iniezioni malevole esterne sui dati persistenti. **Priorità=** alta.

**Performance:**

* *Carico di lavoro:*

Il sistema deve essere in grado di fornire servizio contemporaneamente ad almeno 100 utenti. **Priorità=**bassa.

* *Tempi di risposta:*

Il sistema deve fornire tempi di risposta minimizzati ad almeno 2 secondi per una maggiore fluidità. **Priorità=**bassa.

**Supportabilità:**

* *Architettura del sistema:*

il sistema deve avere un’architettura a 3 livelli per facilitare la manutenibilità. **Priorità=**alta.

**Usabilità:**

* *Graphic User Interface:*

Il sistema deve fornire un'interfaccia grafica.  **Priorità=**alta.

* *Menu contestuale:*

Il sistema devefornire un menu contestuale che permette di non smarrirsi all’interno del sito, cioè di sapere dove si trova in qualsiasi momento. **Priorità=**alta.

* *Nav-bar:*

Il sistema deve fornire una nav-bar grafica con una componente grafica che dia la possibilità di poter raggiungere la homepage. **Priorità=**alta.

* *Sistema Responsive:*

Il sito deve essere responsive, cioè adattarsi a vari dispositivi, ovvero: Smartphone, Tablet, Personal computer. **Priorità=** alta.

* 1. **Definizioni, acronimi e abbreviazioni**
  2. **Riferimenti**

Riferimento al R.A.D.

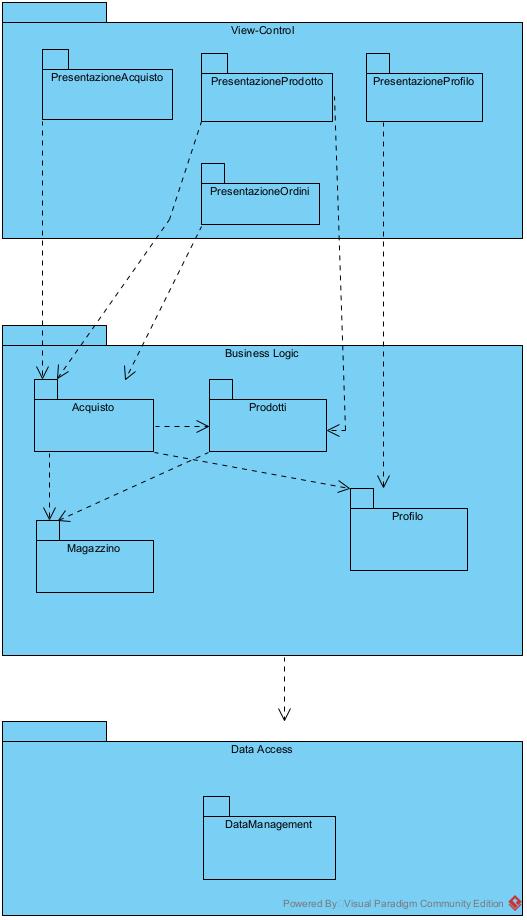
1. **Architettura del Sistema Corrente**

Non avendo un sistema esistente, non abbiamo nessun riferimento ad un’architettura corrente.

1. **Architettura del Sistema Proposto**
   1. **Overview**
   2. **Decomposizione in Sottosistemi**
      1. **Layering & Partitioning**

Decomponiamo il nostro sistema in 3 layers, che si occupano di gestire aspetti e funzionalità differenti. La decomposizione è basata su un particolare pattern per l’architettura software chiamato MVC

(MODEL-VIEW-CONTROL).



**View-Control**

**PresentazioneAcquisto** = visualizzazione e gestione del carrello poter procedere all'acquisto.

**PresentazioneProdotto =** visualizzazione e gestione: ricerca, recensione, aggiunta prodotto al carrello, visualizzare pagina informativa.

**PresentazioneProfilo =** gestione e visualizzazione di: pagina di autenticazione, pagina di registrazione, modifica dati personali.

**PresentazioneOrdini =** gestione e visualizzazione degli ordini.

**Business Logic**

**Acquisto =** Offre servizi per: fare un ordine, ottenere tutti gli ordini e gestire la consegna.

**Prodotti =** Offre servizi per: aggiungere prodotto al carrello, rifornire prodotto, aggiungere recensione, ottenere tutti i prodotti per una ricerca, vedere la disponibilità dei prodotti.

**Profilo =** Offre servizi per: autenticazione utente, registrazione, modifica dati personali.

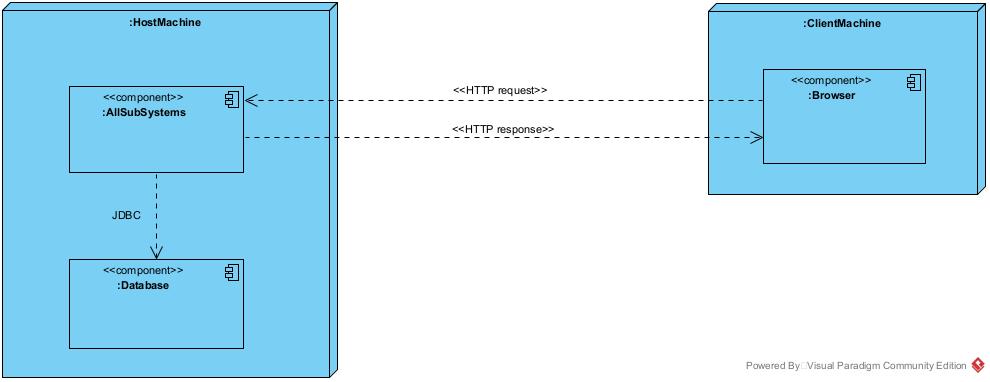
**Magazzino =** Offre servizi per controllare se un prodotto è disponibile e decrementare o aumentare la quantità di un prodotto.

**Data Access**

**DataManagement =** delegato dal DBMS.

* 1. Hardware/Software Mapping

RAAF-GAMING utilizza un’architettura Client-Server. Il Web Server è realizzato da Apache Tomcat ed è situato su una singola macchina, l’interfaccia utente è realizzata utilizzando pagine Java Server Page (JSP) e dalle Servlet che fanno da control. La logica di Business è implementata tramite DAO (Data Access Object) che permettono di accedere e modificare i dati del DB (si trova sulla stessa macchina server) tramite il driver JDBC (Java Database Connectivity) mentre i Bean ci permettono di manipolare i dati ottenuti dal DB tramite i DAO. Il Client è rappresentato dal Web Browser utilizzato dall’utente, che raggiunge il sito tramite un URL utilizzando un protocollo http, il Web Server accetta le richieste dei client reindirizzandoli alla homepage del sito.



fergfvrtbgvtr