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

**GAP  
System Design Document**

****

Data: 30/10/2021

**Partecipanti:**

|  |  |
| --- | --- |
| Nome | Matricola |
| Giammarino Emanuele | 0512108088 |
| Adinolfi Giacinto | 0512107764 |

**Indice**

1. Dominio del problema ……………………………………………………………………4
2. Requisiti Funzionali…………………………………………………………………………5
3. Requisiti non funzionali…………………………………………………………………..6
4. Target Environement………………………………………………………………………7
5. Scenari……………………………………………………………………………………………7

**MODIFICARE**

**1. Introduzione**

Gap nasce come un e-commerce per la vendita di arredamento per interni. Mr. Gap ha deciso di voler avviare la sua attività, ma data la corrente pandemia che ha causato molte restrizioni , e grazie all’ incremento dell'utilizzo della tecnologia e degli acquisti on-line ha optato per la scelta di un e-commerce escludendo l’idea di un negozio fisico. Viene prevista, quindi, una metodologia di consegna totalmente presso un domicilio. Il sito web vuole essere rivolto a tutte le tipologie di visitatori che hanno l’intendo di arredare la propria abitazione.

* 1. **Obiettivi del sistema**

Gli obiettivi del sistema (design goals) rappresentano le qualità desiderate di GAP e forniscono un consistente insieme di criteri che devono essere considerati mentre si fanno decisioni sul design.

Sono stati individuati i seguenti design goals:

• **Performance**

-Garantire tempi di risposta brevi (sotto i 100ms) per le pagine che si visitano.

-Servire più utenti contemporaneamente

• **Affidabilità**

-Garantire la sicurezza dei dati sensibili negando gli accessi non autorizzati.

-Prevenire crash del sistema a fronte degli errori commessi dagli utenti in un eventuale utilizzo scorretto della piattaforma

-La piattaforma dovrà essere disponibile 24 ore su 24 e 7 giorni su 7

• **Manutenzione**

-Il sistema deve poter essere facilmente modificabile per poter correggere eventuali errori

-Il sistema deve gestire in modo corretto eventuali eccezioni software

**1.2 Definizioni, acronimi e abbreviazioni**

**RAD**: Requirement Analysis Document

**SDD**: System Design Document

**MVC**: Model View Controller

**JSP**: Java Servlet Page

**1.3 Riferimenti**

Il documento valuta dei punti trattati specificatamente nel Requirement Analysis Document.

**1.4 Panoramica**

Il documento corrente mostra i dettagli della progettazione del sistema GAP. Innanzitutto, verranno illustrati gli obiettivi di design del sistema e la sua architettura software proposta. Successivamente, verrà analizzata la struttura software adottata correntemente e i sistemi esistenti che adottano un’idea di struttura che sia simile al progetto proposto, evidenziandone i punti cardine. Infine, verrà descritta l’architettura software del sistema proposto.

**2. Architettura software attuale**

Non è presente nessuna architettura software attuale.

**3. Architettura software proposta**

**3.1 Architettura scelta**

Per la decomposizione del sistema di GAP è stato scelto lo stile architetturale MODEL/VIEW/CONTROLLER. I sottosistemi saranno quindi separati in 3 livelli differenti:

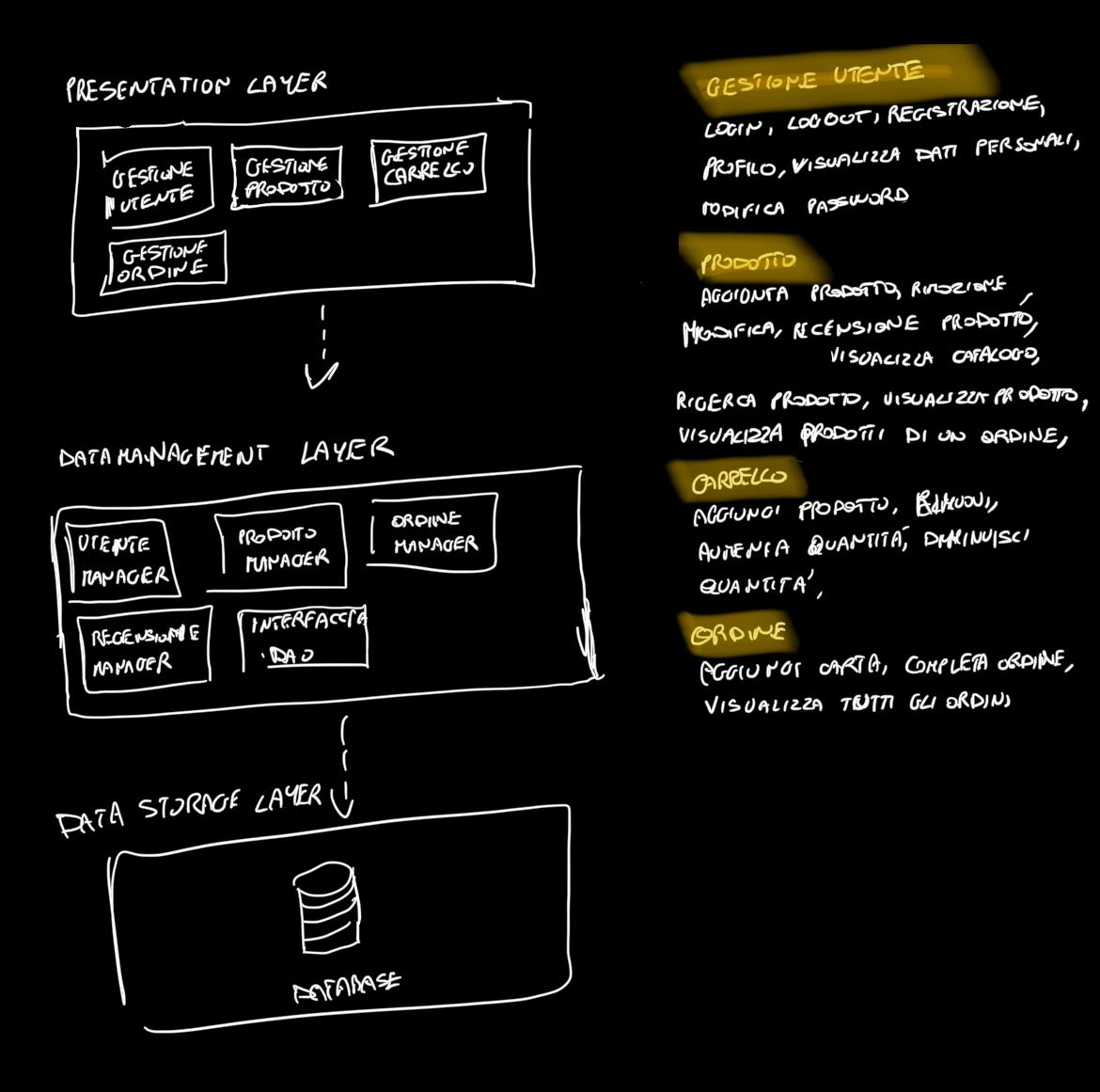
• **Model** – sottosistemi che hanno conoscenza del dominio applicativo e si occupano della memorizzazione dei dati e dell’interazione con la base di dati;

• **View** – sottosistemi che si occupano di mostrare all’utente i dati del dominio applicativo;

• **Controller** – sottosistemi che gestiscono la logica applicativa del sistema e l’elaborazione dei dati.

Tale decomposizione seguendo il modello MVC viene realizzata con l’uso congiunto di Servlet e pagine JSP.

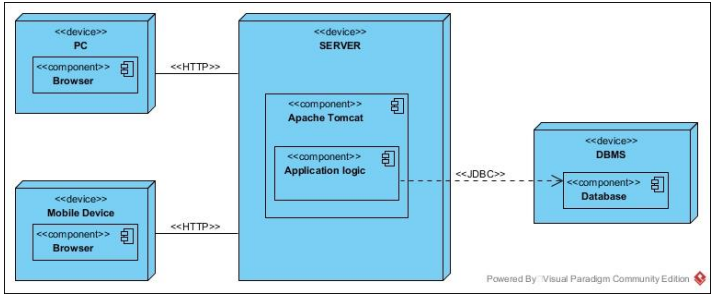
**3.2 Decomposizione sistema in sosttosistemi**



|  |  |
| --- | --- |
| **Gestione Utente** | Comprende le funzionalità di: login, logout, registrazione, visualizzazione profilo, visualizzazione dei dati personali, modifica password. |
| **Gestione Prodotto** | Comprende le funzionalità di: visualizzazione prodotto, ricerca prodotto, visualizzazione prodotto di un dato ordine, aggiunta prodotto, rimozione prodotto, modifica prodotto, visualizzazione catalogo. |
| **Gestione Carrello** | Comprende le funzionalità di: aggiunta prodotto al carrello, rimozione prodotto dal carrello, aumento quantità di un prodotto, decremento quantità di un prodotto, visualizzazione del carrello. |
| **Gestione Ordine** | Comprende le funzionalità di: completa ordine, aggiungi carta di pagamento, visualizza tutti gli ordini. |

**4. Mappatura Hardware/Software**

Il sistema è basato su una architettura di tipo Client/Server. Dal lato server, Apache Tomcat 9.0 offrirà la funzione di WebServer attraverso Java Servlet per la gestione della logica applicativa, e attraverso JSP (Java Servlet Pages) per realizzare l’interfaccia utente del sistema. Dal lato client, il web browser del cliente realizzerà l’interazione tra utente e sistema. Le componenti comunicano attraverso richieste al server e risposte al client attraverso il protocollo HTTP. Il server comunica con i dati persistenti attraverso le API JDBC.



**5. Dati persistenti**

È stato creato un documento separato per la rappresentazione dei dati persistenti.

**6. Controllo degli accessi e sicurezza**

Essendo GAP un sito di e-commerce sono obbligatorie misure di autenticazione e sicurezza. La piattaforma, infatti, prevede un sistema di registrazione per i nuovi utenti e di Login (tramite e-mail e password) per quelli già registrati. Sarà poi il sistema, in base ai dati ricevuti dal Database tramite le credenziali inserite, a limitare o a fornire eventuali operazioni volte a gestire dati critici del sito (catalogo, gestione degli ordini ecc.).

I permessi sono espressi nella Global Access Table seguente:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Ospite** | **Cliente** | **Gestore catalogo** | **Gestore**  **ordini** |
| **Profilo** |  |  |  |  |
| **Carrello** |  |  |  |  |
| **Prodotto** |  |  |  |  |
| **Ordini** |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**7. Controllo software globale**

stato creato

**8. Servizi dei sottosistemi**