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

KawaiiComix

Problem Statement

Versione 1.2

**Immagine che contiene fiore, rosa, Elementi grafici, testo

Descrizione generata automaticamente**

Data: 13/10/2023

|  |  |
| --- | --- |
| Nome | Matricola |
| Simone D’Assisi | 0512113584 |
| Davide Del Franco Natale | 0512113233 |
| Giovanni Sicilia | 0512107458 |

**Coordinatore del progetto**

**Partecipanti**

|  |  |
| --- | --- |
| Nome | Matricola |
| Simone D’Assisi | 0512113584 |
| Davide Del Franco Natale | 0512113233 |
| Giovanni Sicilia | 0512107458 |

|  |  |
| --- | --- |
| **Scritto da:** | Giovanni Sicilia |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autore** |
| 13/10/2023 | 1.0 | Prima stesura del documento | Giovanni Sicilia |
| 13/10/2023 | 1.1 | Aggiunta degli scenari | Simone D’Assisi |
| 13/10/2023 | 1.2 | Ambiente di destinazione e scadenze | Davide Del Franco Natale |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Indice**

[1. Introduzione 4](#_Toc151482127)

[*1.1 Scopo del sistema* 4](#_Toc151482128)

[*1.2 Obbiettivi di progettazione* 4](#_Toc151482129)

[*1.2.1 Criteri di prestazione* 4](#_Toc151482130)

[*1.2.2 Criteri di affidabilità* 4](#_Toc151482131)

[*1.2.3 Criteri di supportabilità* 5](#_Toc151482132)

[*1.2.4 Criteri usabilità* 5](#_Toc151482133)

[*1.3 Riferimenti* 6](#_Toc151482134)

[2. Architettura del sistema proposto 6](#_Toc151482135)

[*2.1 Decomposizione in sottosistemi* 6](#_Toc151482136)

[*2.2 Mapping Hardware/Software* 6](#_Toc151482137)

[*2.3 Gestione della persistenza* 6](#_Toc151482138)

[*2.4 Controllo degli accessi* 7](#_Toc151482139)

[*2.5 Controllo globale del software* 7](#_Toc151482140)

[*2.6 Condizioni limite* 7](#_Toc151482141)

# 1. Introduzione

## *1.1 Scopo del sistema*

Lo scopo del sistema è fornire un e-commerce classico con tutte le principali funzionalità offerte dai maggiori competitor.

## *1.2 Obbiettivi di progettazione*

## *1.2.1 Criteri di prestazione*

* **Throughput**

Il sistema deve essere in grado di gestire almeno cento utenti contemporaneamente, necessità dunque di avere un sistema completamente scalabile.

**Priorità**: Bassa

* **Memoria**

Tutti i dati relativi al catalogo, ai magazzini, agli ordini effettuati e agli utenti registrati sono conservati in un database relazionale. Lo spazio occupato dal sistema dipende quindi dalle dimensioni di quest’ultimo.

**Priorità**: Alta

* **Tempo di risposta**

Il sistema deve garantire un tempo di risposta massimo agli input dell’utente di due secondi. Per fare ciò ci si affiderà ai meccanismi di compressione e decompressione dati del server container utilizzato e quelli di default dei browser, solo in un secondo momento e in seguito ad un’analisi prestazionale verrà preso in considerazione l’uso di tool e librerie esterne per i suddetti meccanismi.

**Priorità**: Bassa

## *1.2.2 Criteri di affidabilità*

* **Robustezza del sistema**

Il sistema deve, in modo da evitare utilizzi impropri e pericolosi delle sue funzionalità, effettuare controlli sui dati inseriti dall’utente.

**Priorità**: Alta

* **Attendibilità**

Il sistema deve garantire l’attendibilità dei servizi messi a disposizione (e.g. Se un ordine viene effettuato con successo bisogna garantire la disponibilità di quel prodotto e allo stesso tempo mantenere in tempo reale il dato aggiornato per gli altri possibili acquirenti).

**Priorità**: Alta

* **Sicurezza**

Il sistema deve garantire l’accesso alle proprie componenti esclusivamente agli utenti autorizzati (e.g. Le sezioni di gestione del sito non devono essere accedute in alcun modo da utenti sprovvisti di un account da gestore).

**Priorità**: Alta

## *1.2.3 Criteri di supportabilità*

* **Modello MVC**

Il sistema deve avere un’architettura a tre livelli che favorisca la manutenibilità. L’idea è di usare un modello MVC nel quale la logica di business, che risulta limitarsi a controlli su dati e permessi e agli accessi alla base di dati, è prevalentemente gestita dai DAO.

**Priorità**: Alta

* **Estensibilità**

Il sistema deve essere facilmente estensibile per permettere l’aggiunta di funzionalità in futuro ma anche per assicurare la possibilità di un lancio dell’applicazione in fase intermedia, priva dunque di tutte le funzionalità progettate ma già interamente utilizzabile per il suo scopo primario. Ciò viene assicurato dall’utilizzo del modello MVC e da una certa componente di paradigma object-oriented.

**Priorità**: Alta

## *1.2.4 Criteri usabilità*

* **User-friendliness**

Il sistema deve essere user-friendly per garantire un’esperienza utente piacevole ed intuitiva. Ciò verrà garantito tramite la GUI grazie alla presenza di una barra di navigazione e di icone familiari per servizi come il carello e l’area utente.

Il sistema deve inoltre aiutare l’utente nella navigazione e nell’utilizzo dei servizi offerti dall’applicazione. Esso verrà infatti continuamente assistito tramite messaggi di errore che lo guideranno nel corretto utilizzo dell’applicazione e nella corretta compilazione dei vari form quando necessario.

**Priorità**: Alta

* **Responsive**

Il sistema deve essere utilizzabile dagli utenti tramite qualsiasi browser e da qualsiasi dispositivo connesso a Internet. Ciò è assicurato dal deployment dell’applicazione su un server container e dalle caratteristiche responsive del sistema che verranno ottenute tramite l’utilizzo di media query.

**Priorità**: Alta

## *1.3 Riferimenti*

* **R.A.D.**: si farà riferimento ai requisiti non funzionali presenti nel R.A.D.

# 2. Architettura del sistema proposto

## *2.1 Decomposizione in sottosistemi*

Il sistema risulta principalmente suddiviso in tre layer: presentation, business logic e data (suddivisione eredita dal modello MVC).

Ognuno di questi layer è formato da uno o più sottosistemi generalmente caratterizzati da una forte coesione ma debolmente accoppiati tra loro.

Il seguente diagramma UML mostra un overview dei sottosistemi e delle loro relazioni:

**/\*\*\*\*\*\*\*\* INSERIRE QUI DIAGRAMMA \*\*\*\*\*\*\*\*/**

* **View:** è il sottosistema che si occupa di gestire ciò che l’utente visualizza a schermo.
* **AcquistoManagement:** è il sottosistema che si occupa di gestire la logica dietro l’acquisto di uno o più prodotti: pagamento, carrello, stato dell’ordine etc.
* **UtenteManagement:** è il sottosistema che si occupa di gestire la logica dietro l’autenticazione dei vari tipi di utente e i loro permessi.
* **CatalogoManagement:** è il sottosistema che si occupa di gestire il catalogo del sistema: prodotti, magazzini, scorte etc.
* **Storage:** è il sottosistema che si occupa dell’interazione con la base di dati (transazioni, consistenza dei dati etc.). Per via della complessità richiesta parte, o il totale, di queste operazioni verranno delegate ad un’istanza di un DBMS relazionale.

## *2.2 Mapping Hardware/Software*

L’architettura del sistema è Client-Server con la gestione dei dati persistenti delegata ad un’istanza di un DMBS relazionale, quale MySQL, che si trova sulla stessa macchina del server.

La comunicazione tra il Browser, sulla ClientMachine, e il sistema, sulla HostMachine, avviene attraverso richieste e risposte HTTP.

Il deployment del sistema, impacchettato in un file war, viene fatto su di un server Apache Tomcat installato sulla HostMachine, che comunica con il DBMS attraverso un driver JDBC.

**/\*\*\*\*\*\*\*\* INSERIRE QUI DIAGRAMMA \*\*\*\*\*\*\*\*/**

## *2.3 Gestione della persistenza*

I dati del sistema che devono essere resi persistenti sono:

* Le informazioni degli utenti;
* Le informazioni sui prodotti;
* Le informazioni sul carrello;
* Gli ordini e tutto ciò che li riguarda (e.g. pagamento);
* I magazzini .

Come già detto questo compito verrà delegato ad un’istanza di MySQL.

## *2.4 Controllo degli accessi*

**/\*\*\*\*\*\*\*\* INSERIRE QUI MATRICE ACCESSI \*\*\*\*\*\*\*\*/**

## *2.5 Controllo globale del software*

Il controllo globale del software è di tipo event-based. Essendo un applicazione web, sarà il Web Server ad occuparsi dello smistamento delle varie richieste HTTP verso delle apposite Servlet che si occuperanno di gestire la richiesta, interagire con le altre componenti del sistema ed elaborare una risposta.

## *2.6 Condizioni limite*

Le *boundary condition* del sistema sono le seguenti:

* **Installazione del sistema:** l’installazione del sistema verrà eseguita da un sistemista ed un programmatore che avranno anche il compito di riempire la base di dati con gli account dei vari gestori ed il catalogo. In particolare, il server container Apache Tomcat, sul quale verrà fatto il deployment del sistema, verrà installato su di una macchina remota, per questo ci si affiderà ad un’azienda di web hosting.
* **Avvio del sistema:** il sistema verrà avviato da un sistemista in seguito all’installazione. Dopo l’avvio viene attivato il DBMS MySQL, la cui connessione col sistema verrà stabilita mediante driver JDBC.
* **Manutenzione del sistema:** per l’applicazione di un aggiornamento al sistema si dovrà scegliere una finestra oraria (di circa 1-2 ore) nella quale il sistema andrà offline, saranno applicate le dovute modifiche e poi verrà eseguito nuovamente l’avvio. Queste operazioni verranno eseguite da un sistemista e un programmatore. La finestra oraria di riferimento sarà dalle 04:00 alle 05:00 UTC+1, fascia orario che si prevede essere di minor attività per il sistema (si potrà decidere una nuova finestra studiando i dati di attività del sistema dopo il primo anno dall’avvio).
* **Fallimento del sistema:** nel caso il sistema fallisca a causa di un problema relativo al servizio di hosting (guasto della macchina remota, crash dei server etc.) sarà premura della nostra squadra fare pressioni all’azienda per risolvere la situazione quanto prima. Nel caso il fallimento sia dovuto ad un’eccezione non gestita dal nostro sistema un sistemista lo riavvierà il prima possibile per poi segnalare il problema alla nostra squadra di programmatori che si occuperà di risolverlo in maniera permanente.