

**Università degli Studi di Salerno**

**Anno Accademico 2016/2017**

**Corso di Ingegneria del Software**

**System Design Document**

**V 1.0**

****

**Top Manager:**

*Prof. De Lucia Andrea*

**Team di sviluppo:**

|  |  |
| --- | --- |
| Nome e Cognome | Matricola |
| *Stefano Foresta* | 0512100504 |
| *Gennaro Franzese* | 0512100270 |

**Revision History:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Autore** | **Data** | **Descrizione** | **Versione** |
| *Gennaro Franzese* | *21/11/2016* | *Struttura documento* | *v 0.4* |
| *Stefano Foresta* | *21/11/2016* | *Stesura generale del documento* | *v 1.0* |

Sommario

[Introduzione 4](#_Toc467594835)

[Scopo del sistema 4](#_Toc467594836)

[Design goals 4](#_Toc467594837)

[Criteri di prestazione 4](#_Toc467594838)

[Criteri di affidabilità 4](#_Toc467594839)

[Criteri di manutenzione 5](#_Toc467594840)

[Overview 5](#_Toc467594841)

[Architettura del sistema 6](#_Toc467594842)

[Architettura del sistema proposto 6](#_Toc467594843)

[Overview 6](#_Toc467594844)

[Decomposizione in sottosistemi 7](#_Toc467594845)

[Interface Layer 7](#_Toc467594846)

[Application Logic Layer 7](#_Toc467594847)

[Storage Layer 7](#_Toc467594848)

[Hardware/Software mapping 7](#_Toc467594849)

[Gestione dei dati persistenti 7](#_Toc467594850)

[Controllo degli accessi e sicurezza 7](#_Toc467594851)

[Controllo degli accessi 7](#_Toc467594852)

# Introduzione

## Scopo del sistema

Il sistema, automatizzando la maggior parte delle operazioni che si svolgono in un normale negozio di informatica, renderà più fluido e leggero il carico di lavoro di chi è addetto all’amministrazione del suddetto negozio. Le operazioni di carico/scarico merci, di acquisto/vendita prodotti e di gestione documentazione (fatture in entrata/uscita) saranno gestite dal sistema che si verrà a creare e quindi i dati verranno conservati in un database relazionale creato ad-hoc e la gestione sarà affidata ad una web application disegnata su misura per il cliente.

## Design goals

Il sistema sarà progettato con un’interfaccia user-friendly in modo da agevolare l’utilizzo a persone che non hanno molta dimestichezza con le web-application di questo genere. Il software sarà ideato in modo da gestire al meglio le risorse del sistema evitando fastidiosi “crash”. Sarà un prodotto utile e semplice nell’interazione che permetterà ai propri utenti di trovarsi davanti un software facile da utilizzare. La gestione del database e le relative operazioni saranno completamente gestibili dall’interfaccia di gestione web che sarà creata ad-hoc secondo le richieste del cliente e attraverso l’invio di numerosi feedback per ogni operazione il proprietario sarà agevolato nel cercare di commettere meno errori possibili. Dal lato client l’interfaccia sarà il più semplice possibile per permettere una ricerca veloce ed intuitiva dei prodotti e per agevolare l’acquisto di un bene/servizio in maniera semplice e con una serie finita di passi.

### Criteri di prestazione

|  |  |
| --- | --- |
| **Criteri di design** | **Definizione** |
| ***Tempo di risposta*** | Il sistema deve essere efficiente, deve garantire una risposta in breve tempo, massimo 15 secondi, senza che si verifichino errori. |
| ***Throughput*** | Il sistema riesce a soddisfare massimo 50 task contemporaneamente in un tempo prefissato di un minuto. |
| ***Memoria*** | La memoria verrà gestita dal sistema il quale oltre a gestirne la memoria fisica (RAM), gestisce anche la memoria secondaria (Hard Disk). |

### Criteri di affidabilità

|  |  |
| --- | --- |
| **Criteri di design** | **Definizione** |
| ***Robustezza*** | Il sistema deve prevedere e gestire eventuali input errati senza bloccare il funzionamento del sistema quindi verrà fatta molta attenzione durante la fase di sviluppo per lo sviluppo della gestione degli errori.  Il sistema ha poi un layer-application che serve ad elaborare i dati inviati dalle varie interfacce. |
| ***Sicurezza*** | Per sicurezza, oltre a intendere la privacy del cliente, si intende l’inalterabilità dei dati dovuta da modifiche sbagliate, quindi in caso di modiche, il sistema tramite interfaccia GUI, manderà un messaggio di conferma a colui che utilizza il software. |
| ***Tolleranza crash*** | Questo è un aspetto fondamentale, il sistema non può operare in situazioni di errore, pertanto devono essere previste queste situazioni.  Al verificarsi di un crash verrà salvato il lavoro in corso in file temporanei ed il sistema software terminerà la sua esecuzione riavviandosi e cercando di mantenere inalterate le impostazioni.  Nel caso in cui ci siano errori che non permettono il riavvio del sistema deve essere prevista una macchina di supporto che rimanga attiva per tutto il tempo del ripristino mantenendo attive le funzionalità di base. |
| ***Affidabilità*** | Il sistema oltre alla gestione degli errori deve garantire il funzionamento 24h su 24h tranne nei tempi relativi alla manutenzione ordinaria che da contratto prevede un tempo massimo di stop di 12h. |

### Criteri di manutenzione

|  |  |
| --- | --- |
| **Criteri di design** | **Definizione** |
| ***Estendibilità*** | Ai fini di uno sviluppo futuro deve essere possibile aggiungere, in un secondo momento, altre funzionalità al sistema.  Per rendere possibile questo, la piattaforma verrà programmata nel modo più semplice possibile cercando di dividere al meglio le funzionalità in modo da poter, in futuro, apportare modifiche anche a singole parti e non per forza stravolgendo l’intero sistema. |
| ***Leggibilità*** | Il codice deve essere chiaro, quindi i nomi delle variabili e dei metodi che verranno utilizzati devono essere coerenti al contesto d’uso, dove è strutturato, per rendere più veloce una modifica o una estensione del sistema anche da parte di terzi che si avvicinano al progetto in un secondo momento. |

## Overview

Il presente documento SDD (System Design Document) si occupa di:

1. Definire gli obiettivi di design del progetto (identificando quali caratteristiche di qualità dovrebbero essere ottimizzate);
2. Decomporre il sistema in vari sottosistemi più piccoli che sono di più facile realizzazione;
3. Selezionare le strategie per costruire il sistema:
   1. Strategia Hardware/Software;
   2. Strategie che si riferiscono alla gestione dei dati persistenti;
   3. Il flusso di controllo globale;
   4. Le politiche di controllo degli accessi;
   5. La gestione delle condizioni limite;

L’output del system design è un modello del sistema che include quindi la decomposizione dello stesso in sottosistemi con la relativa descrizione per ognuna delle strategie utilizzate.

# Architettura del sistema

L’architettura attualmente proposta non va a sostituire nessuna architettura esistente, il sistema è alla sua prima versione.

# Architettura del sistema proposto

## Overview

Il software è composto seguendo lo schema client/server

* **Client:** il browser preferito del cliente.
* **Server:** le pagine PHP che implementano l’intero sistema.

Sono possibili tre configurazioni:

* **Single-Tier:** tutto il sistema è distribuito su una sola macchina che funzionerà da client e da server.
* **Two-Tier:** il sistema è distribuito ed eseguito su due macchine differenti; una macchina gestirà tutto il lato server e conterrà anche il database mentre l’altra macchina eseguirà il client.
* **Three-Tier:** a differenza della configurazione Two-Tier il database verrà gestito su una macchina separata.

La configurazione scelta è quella three-tier, così eventuali modifiche ad uno dei tre moduli non comporterà il blocco del sistema e la relativa modifica dei blocchi rimanenti rispetto a quello aggiornato. Tale configurazione è basata su tre layer:

1. **Interface Layer:** oggetti con cui l’utente interagisce (form, button, textbox, ecc.).
2. **Application Logic Layer:** livello che si occupa di gestire moduli che girano su un application server che genera i moduli per i contenuti dinamici e che riceve le richieste e le elabora.
3. **Storage Layer:** questo livello gestisce l’archiviazione persistente dei dati.

## Decomposizione in sottosistemi

I layer previsti sono: *Interface Layer, Application Logic Layer, Storage Layer*. Di seguito riportiamo, per ogni modulo, le componenti da cui è composto, che rappresentano nel dettaglio le operazioni di cui si occupa e i dati su cui operano.

### Interface Layer

|  |  |
| --- | --- |
| **Modulo** | **Descrizione** |
| ***Interfaccia web*** | Questo modulo descrive l’interfaccia grafica con cui l’utente interagisce e che invia al sistema le varie richieste. |

### Application Logic Layer

|  |  |
| --- | --- |
| **Modulo** | **Descrizione** |
| ***Gestione utente*** | Il modulo si occupa della gestione generale dell’utente. Creazione, modifica, gestione, cancellazione. |
|  |  |
|  |  |

### Storage Layer

|  |  |
| --- | --- |
| **Modulo** | **Descrizione** |
| ***Database*** | Gestisce le richieste di dati in entrata e in uscita. Si occupa del database e della sua gestione. |

## Hardware/Software mapping

Il software utilizzerà un’architettura Three-Tier in configurazione Client-Server-Database:

* **Client:** si intende lamacchina utilizzata dall’utente e dove gira il browser che quest’ultimo utilizza per navigare sul sito.
* **Server:** si intende la macchina su cui gira l’applicazione: genera i moduli per i contenuti dinamici e riceve ed elabora le richieste.
* **Database:** si intende la macchina su cui il database è installato e che gestisce quindi le richieste di dati sia in input che in output.

## Gestione dei dati persistenti

La scelta dei dati persistenti è stata fatta analizzando tutti gli oggetti rilevanti che in caso di chiusura controllata/incontrollata dovranno essere conservati.

## Controllo degli accessi e sicurezza

### Controllo degli accessi

Il sistema prevede due tipi di attori:

* **Admin:** è l’amministratore