Università degli studi di Salerno

*Corso di Laurea in Informatica*

**System Design Document**



**Autori:**

**Matricola**

0512104934

0512104814

0512104826

**Nome**

Acierno Erminio

Avallone Daniele

Paolillo Domenico

Sommario

[**1.Introduction** 3](#_Toc24992936)

[1.1 Purpose of the system 3](#_Toc24992937)

[1.2 Design Goals 3](#_Toc24992938)

[1.3 Definitions, acronyms and abbrevations 4](#_Toc24992939)

[1.4 References 4](#_Toc24992940)

[**2. Current system Architecture** 5](#_Toc24992941)

[**3. Proposed software architecture** 5](#_Toc24992942)

[3.1 Overview: 5](#_Toc24992943)

[3.2 Subsystem decomposition 6](#_Toc24992944)

[3.3 Hardware/software mapping 9](#_Toc24992945)

[3.4 Persistent data management 10](#_Toc24992946)

[3.5 Access Control and Security 13](#_Toc24992947)

[3.6 Boundary Conditions 14](#_Toc24992948)

# **1.Introduction**

## **1.1 Purpose of the system**

L’SDD definisce l’architettura e la progettazione del sistema per fornire al team di sviluppo una guida accurata sulla stessa architettura. L’obiettivo dell’implementazione che si propone è quello di costruire un sito web efficiente dedicato alla compagnia FasTrain che conceda agli utenti di acquistare biglietti del treno. Saranno implementate dunque le seguenti funzionalità:

-Pannello amministratore che gli permetta di gestire le corse presenti, quindi eliminarle, modificarle, o aggiungerne nuove;

-Registrazione e login per l’utente;

-Acquistare un biglietto;

-Ricerca corse;

-Visualizzazione offerte;

## **1.2 Design Goals**

Si definiscono i seguenti obiettivi di progettazione:

* **Utilizzo di java**: siccome il sito deve essere perfettamente disponibile su più browser possibili si è deciso di implementare utilizzando Java.
* **Usabilità**: L’interfaccia grafica deve essere pulita e intuitiva in modo da garantire all’utente sicurezza e permettergli di interagire con il sistema.
* **Robustezza**: Tutti i campi da compilare e le interazioni devono essere controllate in modo da non permette di inserire dati non validi
* **Tempo di risposta**: L’interazione con il sistema non deve superare i 30 secondi quando si effettua l’acquisto un biglietto
* **Multiutente**: Il sistema supporta operazioni che so eseguite da più utenti contemporaneamente
* **Sicurezza**: La sicurezza è garantita nella fase di login e nelle password criptate utilizzando il sistema fornito dal database.
* **Estendibilità**: E’ possibile aggiungere in futuro nuove classi e funzionalità al sistema che estendano quelle già esistenti.

## **1.3 Definitions, acronyms and abbrevations**

• SDD: Software Design Document;

• RAD: Requirements analysis document.

• BROWSER: Explorer, Chrome, Mozilla.

• WebBrowser: Client (utente che accede al sistema)

• WebServer: Server su cui sono memorizzate le risorse.

## **1.4 References**

* Problem statement
* RAD
* <https://www.italotreno.it/it>
* Bern Bruegge, Allen H. Dutoit, Object-Oriented Software Engineering - Using UML, Patterns, and JAVA, 3rd edition.

# **2. Current system Architecture**

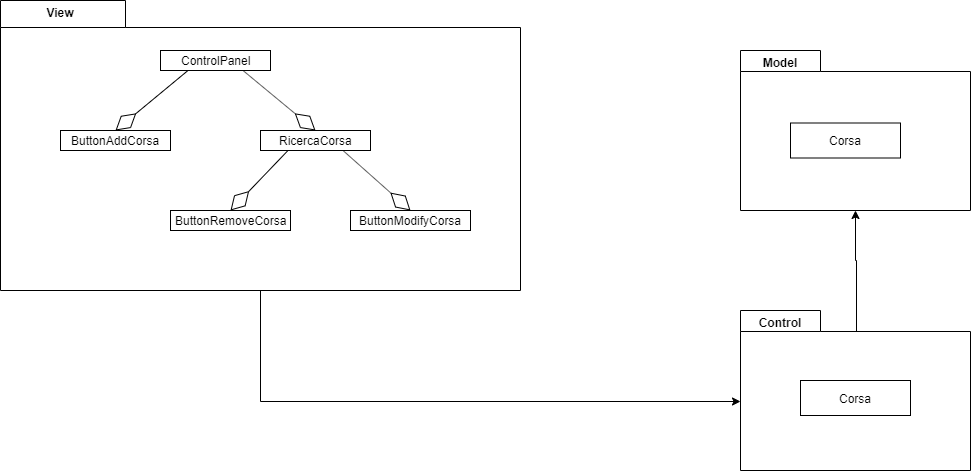
Il sistema presentato non è stato mai implementato prima d’ora, quindi le fasi di design e raccolta dei requisiti sono state compiute discutendo con il cliente, analizzando le sue richieste e facendo dei paragoni con altre piattaforme simili presenti in rete (<https://www.italotreno.it/it>).

# **3. Proposed software architecture**

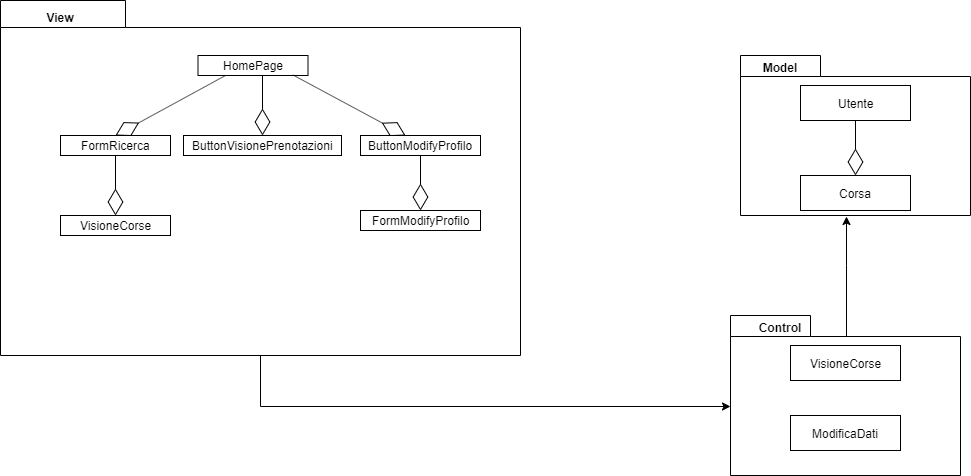
## **3.1 Overview:**

1. Il sottosistema individuato con il nome di “Gestione Gestore” coinvolge l’admin nella sezione dedicata nella gestione delle nuove corse, della modifica di una corsa già esistente e del cancellare una corsa.
2. Il sottosistema individuato con il nome di “Gestione Autenticazione” coinvolge il visitatore nella sezione dedicata alla registrazione di un nuovo utente, di login (qualora avesse già completato la registrazione).
3. Il sottosistema individuato con il nome di “Gestione Ricerca” coinvolge diverse figure: admin, visitatore, utente nella sezione dedicata per la ricerca di una corsa tramite una form.
4. Il sottosistema individuato con il nome di “Gestione Utente” coinvolge l’utente registrato nella sezione dedicata alla visualizzazione delle prenotazioni precedentemente fatte, alla modifica dei propri dati.
5. Il sottosistema individuato con il nome di “Gestione Prenotazione” coinvolge l’utente registrato nella sezione di acquisto di un biglietto relativo alla corsa da lui selezionata.

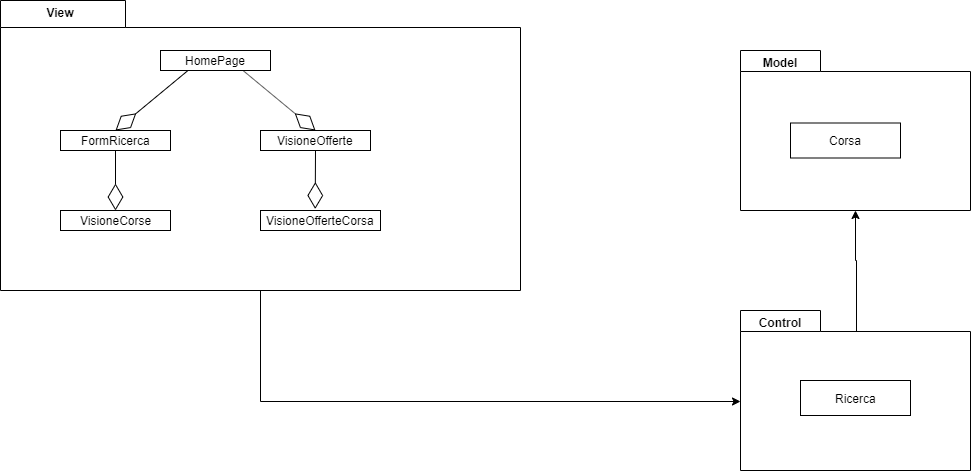
## **3.2 Subsystem decomposition**

**Gestione Gestore**

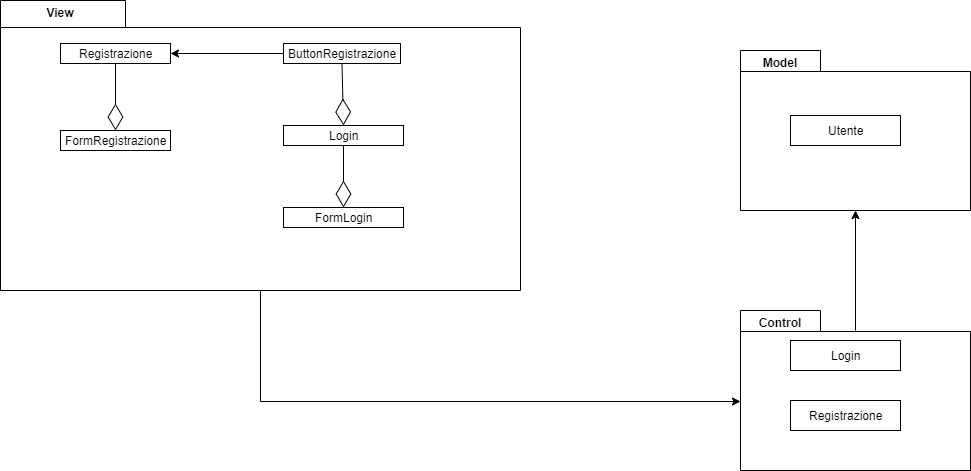
**Gestione Utente**

****

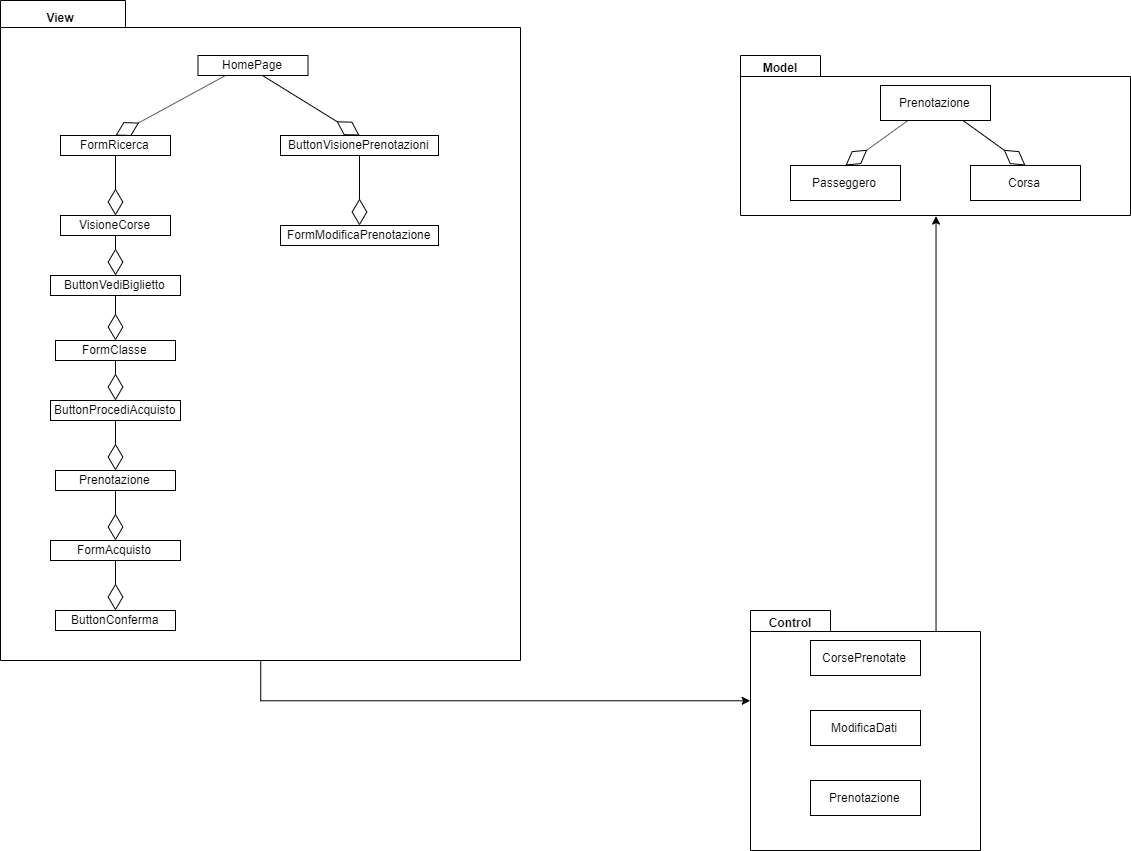
**Gestione Ricerca**

****

**Gestione Autenticazione**

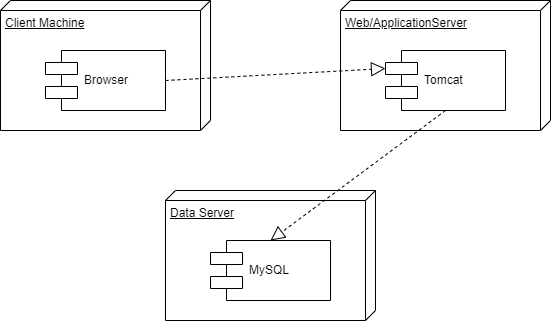
****

**Gestione Acquisto**

****

## **3.3 Hardware/software mapping**

Il seguente diagramma UML illustra la mappatura hardware e software per il sistema fasTrain.

****

## **3.4 Persistent data management**

## 

**Utente**

|  |  |  |
| --- | --- | --- |
| Attribute Name | Attribute Type | Attribute Size |
| Email | String | 35 |
| Password | String | 20 |
| Nome | String | 20 |
| Cognome | String | 20 |
| NumeroTelefono | String | 20 |
| Indirizzo | String | 30 |
| Città | String | 30 |
| Provincia | String | 30 |
| DataDiNascita | date | 10 |

**Prenotazione**

|  |  |  |
| --- | --- | --- |
| Attribute Name | Attribute Type | Attribute Size |
| CodicePrenotazione | int | 999 |
| Data | date | 10 |
| PrezzoTotale | double | 200 |
| Carta | int | 16 |
| ProprietarioCarta | String | 30 |
| Cvv | Int | 3 |
| ScadenzaCarta | Date | 10 |
| ClasseScelta | String | 20 |

**Passeggero**

|  |  |  |
| --- | --- | --- |
| Attribute Name | Attribute Type | Attribute Size |
| Nome | String | 20 |
| Cognome | String | 20 |
| CodiceDocumento | String | 10 |
| CodicePasseggero | Int | 3 |

|  |  |  |
| --- | --- | --- |
| Attribute Name | Attribute Type | Attribute Size |
| CodiceCorsa | int | 99999 |
| CodiceTreno | int | 99999 |
| StazionePartenza | String | 20 |
| StazioneArrivo | String | 20 |
| OraPartenza | String | 10 |
| OraArrivo | String | 10 |
| NumeroPostiEconomy | int | 999 |
| NumeroPostiBusiness | int | 999 |
| NumeroPostiPremium | int | 999 |
| PrezzoClasseEconomy | double | 200 |
| PrezzoClasseBusiness | double | 200 |
| PrezzoClassePremium | double | 200 |
| Sconto | int | 99 |
| Data | date | 10 |

**Corsa**

## 

## **3.5 Access Control and Security**

|  |  |  |
| --- | --- | --- |
| Oggetti | Attori | |
| Cliente | Amministratore |
| Account | Login ✓  Logout ✓  Visualizza Profilo ✓  Registrazione ✓  Modifica Profilo ✓ | Login ✓  Logout ✓  Visualizza Profilo ✓  Modifica Profilo ✓ |
| Prenotazione | Visualizza Prenotazioni ✓  Modifica Prenotazioni ✓ |  |
| Acquisto | Acquisto Biglietto ✓ |  |
| Corsa | Ricerca Corsa ✓  Visualizza Offerte Corse ✓  Vedi Info Corsa ✓ | Ricerca Corsa ✓  Modifica Corsa ✓  Cancella Corsa ✓  Aggiungi Corsa ✓ |

Il sistema sarà accessibile da browser e da WebServer, che come indicato sarà Apache Tomcat, il quale si occuperà della gestione in maniera concorrente gli accessi e le varie procedure.

Per la fase di autenticazione, quindi quando l’utente sottomette i propri dati durante la fase di registrazione, i dati saranno salvati in un DBMS, che come indicato sarà MySQL, il quale si occuperà delle interrogazioni necessarie per l’accesso.

La visualizzazione delle corse consiste nel completamento di una form che a seconda dei dati inseriti dall’utente proporrà dei risultati compatibili con essi, l’accesso ai dati qui sarà in sola lettura quindi non sarà necessario gestire la procedura in maniera concorrente.

La procedura di Acquisto di un biglietto invece è un’operazione complessa poiché può consistere di più use case; l’utente può acquistare un biglietto tramite la visualizzazione delle offerte, o eseguendo la ricerca. Solo nella fase di completamento dell’acquisto sarà necessario gestire la concorrenza.

Il visitatore invece può accedere solo a determinate sezioni, può sempre verificare le offerte presenti, può visualizzare le corse disponibili al momento ma, gli sarà negata la possibilità di effettuare acquisti senza previa registrazione/accesso.

Infine, il gestore, il quale sarà l’unico a poter accedere alle seguenti funzioni di rimozione di una corsa, di modifica di una corsa e di aggiunta di una corsa. Essendo l’unico attore a poter accedere a queste funzioni e quindi l’unico a manipolare i dati presenti nel database, non sarà necessario gestire in maniera concorrente le operazioni descritte precedentemente.

## **3.6 Boundary Conditions**

* **Inizializzazione**

Quando il sito sarà lanciato per la prima volta il sistema sarà sempre raggiungibile.

* **Terminazione**

Il sistema lato server solo in corrispondenza di guasti e malfunzionamenti sarà sospeso.

Lato client il sistema termina alla chiusura della pagina web che lo ospita.

* **Fallimento**

Qualora si verifichino casi eccezionali quali, mancanza di elettricità o guasti hardware, il sistema fallisce lato server.

Lato client il sistema può fallire qualora si verifichino dei guasti particolari come l’assenza di connessione internet.

* **Eccezioni**

Un down della web server rende il sito non agibile per un periodo di tempo non stimato, il sistema lato client mando un messaggio di errore “http 404 pagina al momento non disponibile”.