# Indice

-Obiettivo

-Modello E/R

-Modello relazionale

-Implementazione su Oracle SQL Developer:

- Creazione delle tabelle;

- Creazione dei Trigger;

- Esecuzione degli esercizi richiesti.

-Sviluppo di Entity, Service, Repository e Controller su Eclipse EE

- Visualizzazione su swagger

- Front-End

-pagina Home

-Login e Register

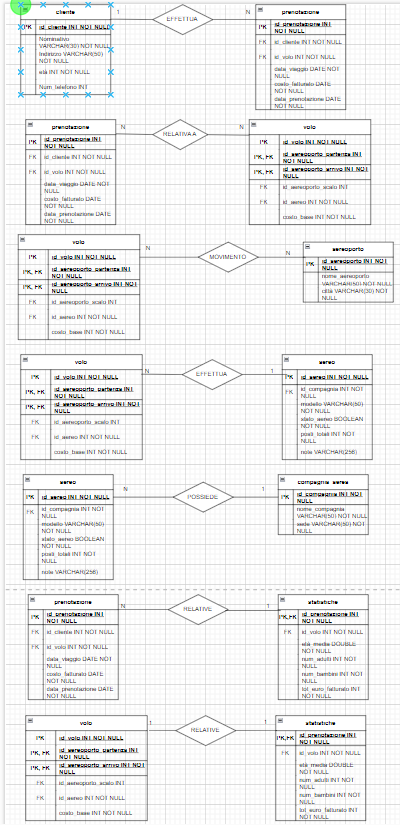
-Pagina con insert,update,delete e visualizzazione di prenotazione

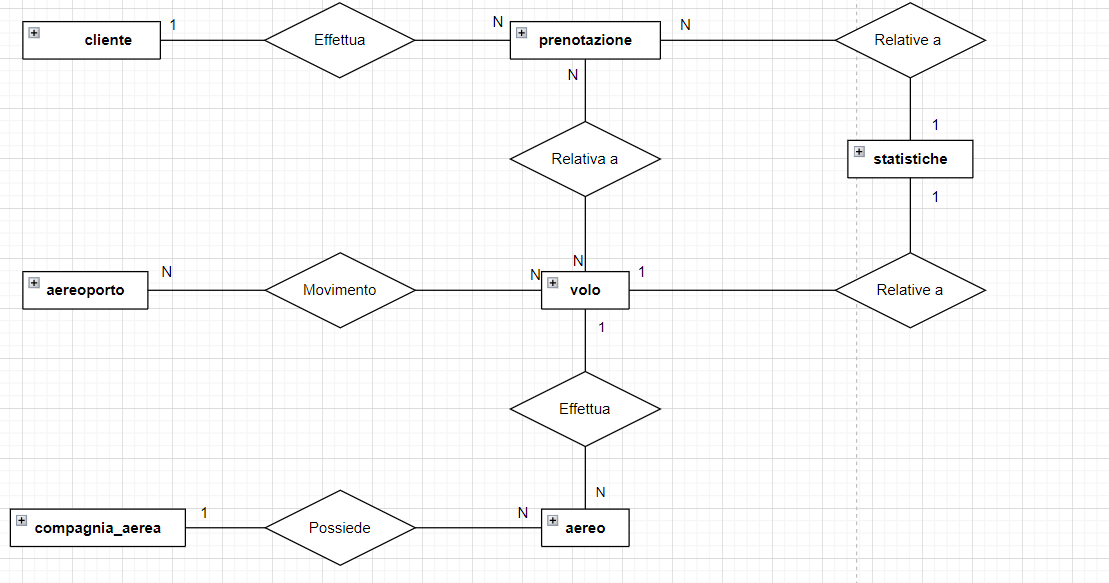
# Obiettivo

Lo scopo del progetto è creare un'applicazione web con la quale gli utenti potranno acquistare biglietti aerei. Il sistema si basa su un database relazionale con le sue funzioni di gestione voli e prenotazione. Avremo un database che supporterà la prenotazione su molteplici tratte in tutto il mondo, centinaia di voli per varie compagnie aeree.

# Modello E/R

Dopo aver letto la traccia fornitaci con tutte le specifiche richieste ho iniziato a sviluppare il modello E/R in 2 modi diversi: -Frammentato: Ogni tabella si collega tramite una relazione ad una sola tabella per volta.



-Completo: Tutte le tabelle collegate in un unico diagramma.

# Modello Relazionale

Ho normalizzato tutte le tabelle che avevo sviluppato nel precedente modello E/R con tutti i campi impostandoli nel modo migliore per svilupparli successivamente nel mio DB.

TABELLA VOLO

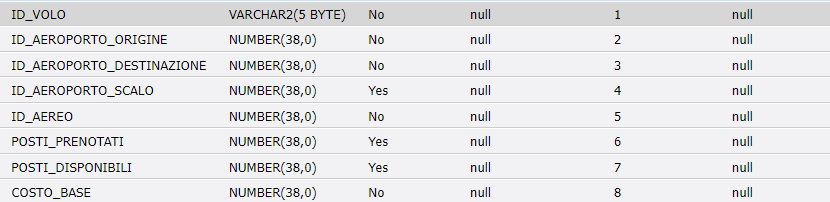


TABELLA AEREO

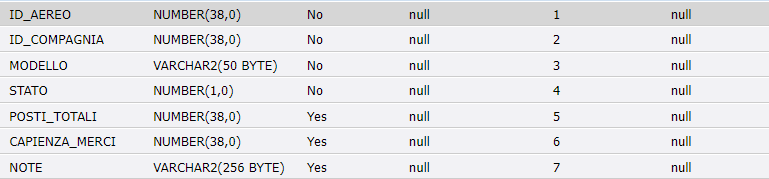


TABELLA AEREOPORTO



TABELLA CLIENTE

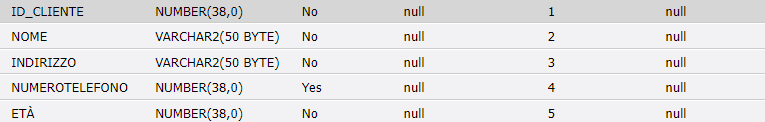


TABELLA COMPAGNIA AEREA



TABELLA PRENOTAZIONE

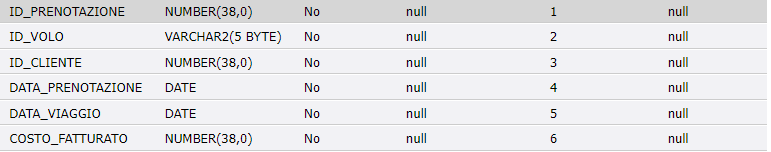
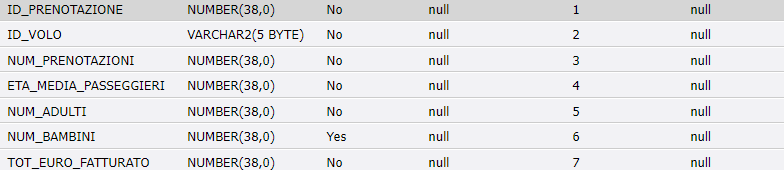


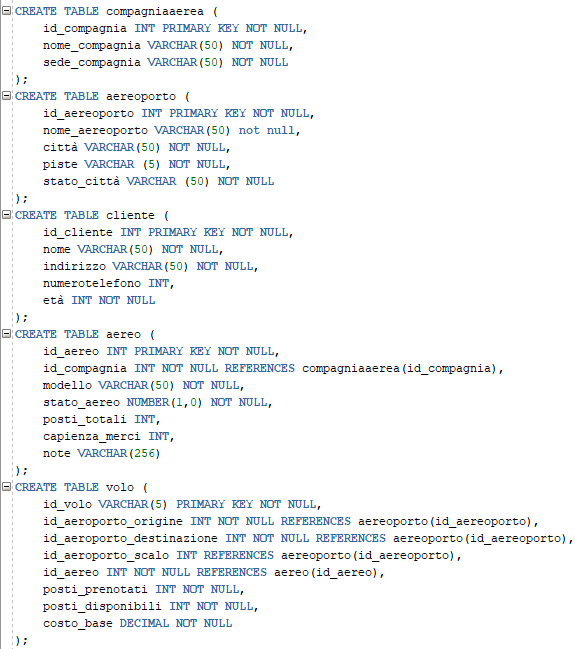
TABELLA STATISTICHE

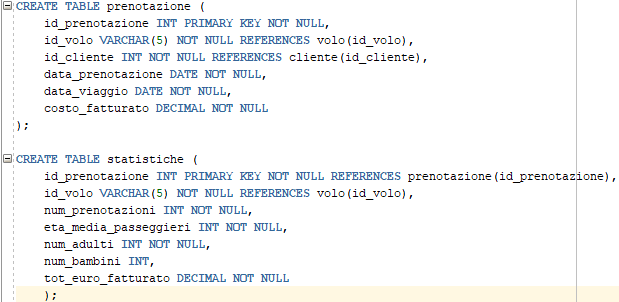


# Implementazione in Oracle SQL Developer

Parte 1.0:

Creazione delle tabelle seguendo il modello E/R e la normalizzazione delle stesse tabelle



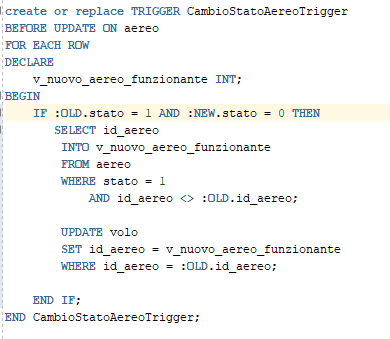


Dopo aver creato tutte le apposite tabelle ho sviluppato grazie all’ aiuto del AI Chat GPT gli insert per tutte le tabelle tranne per statistiche che dovrà essere una tabella di auto implement tramite un trigger apposito che si aggiornerà ogni qual volta viene inserito un nuovo valore nella tabella Volo.

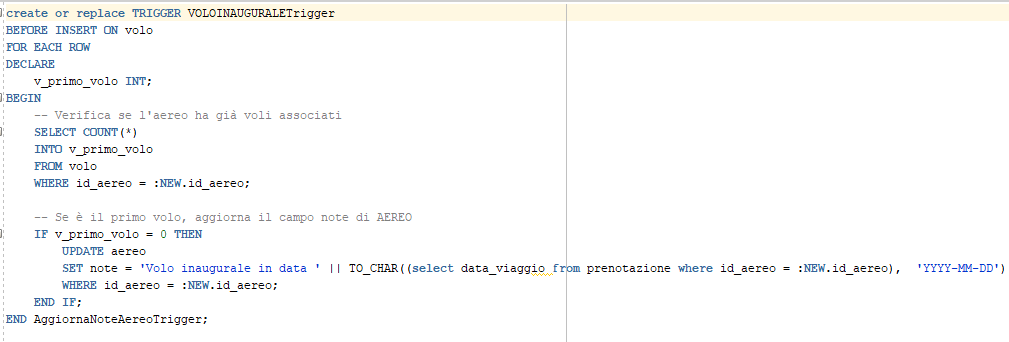
Parte 1.1

Dopo aver creato le tabelle mi sono mosso subito per la creazione inizialmente di 3 trigger richiesti dalla traccia:

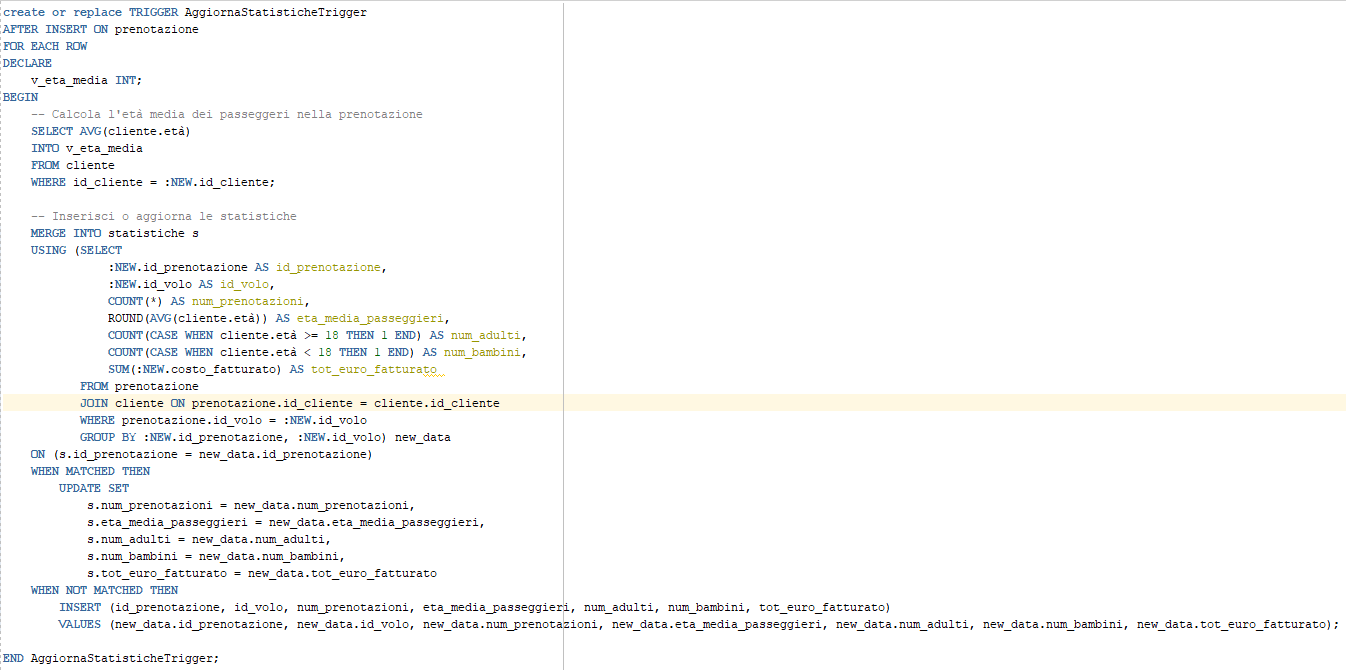
1. Al cambio di stato di un AEREO da funzionante a non funzionante, modifichi automaticamente tutti i VOLI collegati all’aereo in questione, e imposti ai VOLI in primo aereo utile al posto di quello non funzionante



1. All’inserimento di un nuovo VOLO, verificare che per l’aereo assegnatoli sia il primo volo, in questo caso sarà necessario aggiornare il campo note di AEREO, con la dicitura “*Volo inaugurale in data [data del volo inserito]*”



1. All’inserimento di una prenotazione aggiornare con un trigger tutte le informazioni della tabella STATISTICHE

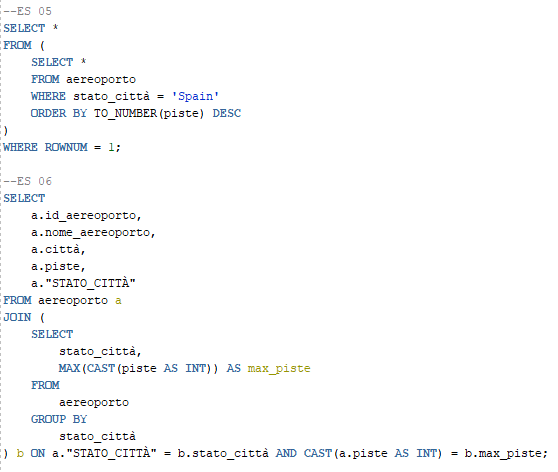


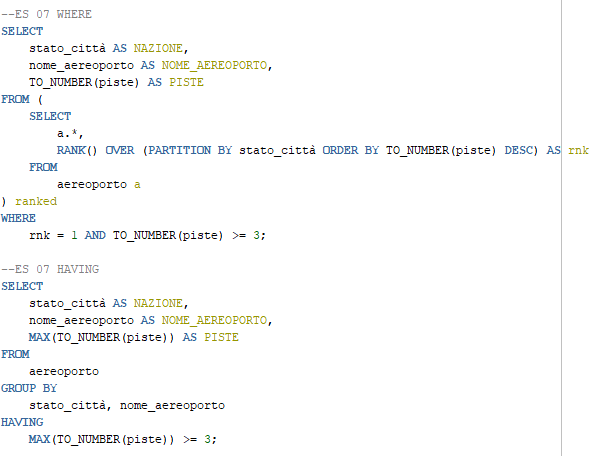
Per più info sul db visitare la seguente pagina: [index.html](index._files/index.html)

Parte 1.2

Dopo la creazione dei 3 trigger sono passato a crearmi le insert di tutte le tabelle presenti nel mio database e a sviluppare le query che ci hanno fornito i nostri tutor per vedere se il DB fosse sviluppato correttamente e qual ora ce ne fosse bisogno modificarlo per renderlo migliore.









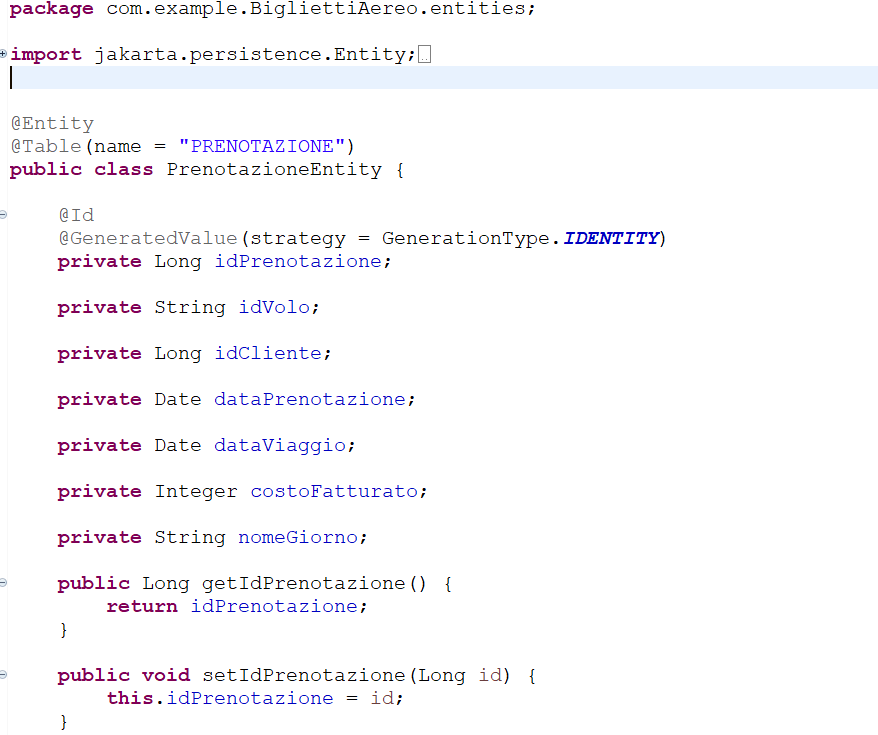
Creazione di Entity, Service, Repository e Controller su Eclipse EE

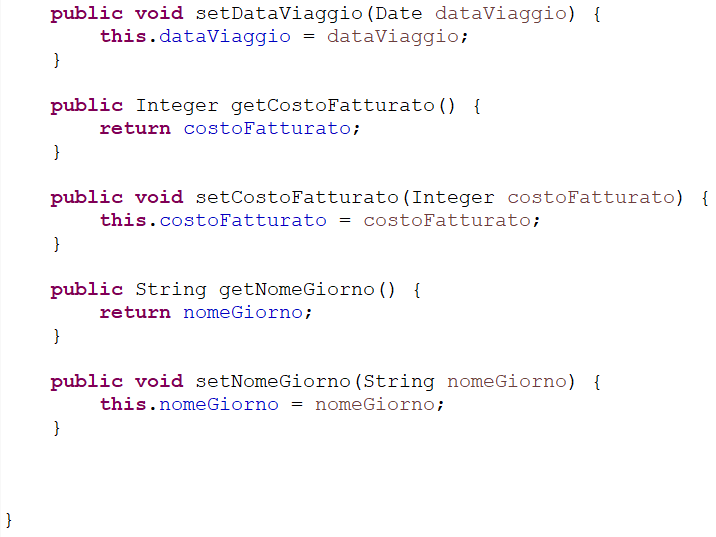
Dopo tutta la parte di sviluppo del database sono passato a sviluppare su java EE(EnterPrise) dove sfruttando Spring Boot ho creato il mio progetto con Maven e dove sono andato in seguito a sviluppare le entity i service i repository e i controller per permettermi di modificare il database usando Swagger o Postman per poi collegare il tutto alla pagina di Front-End.

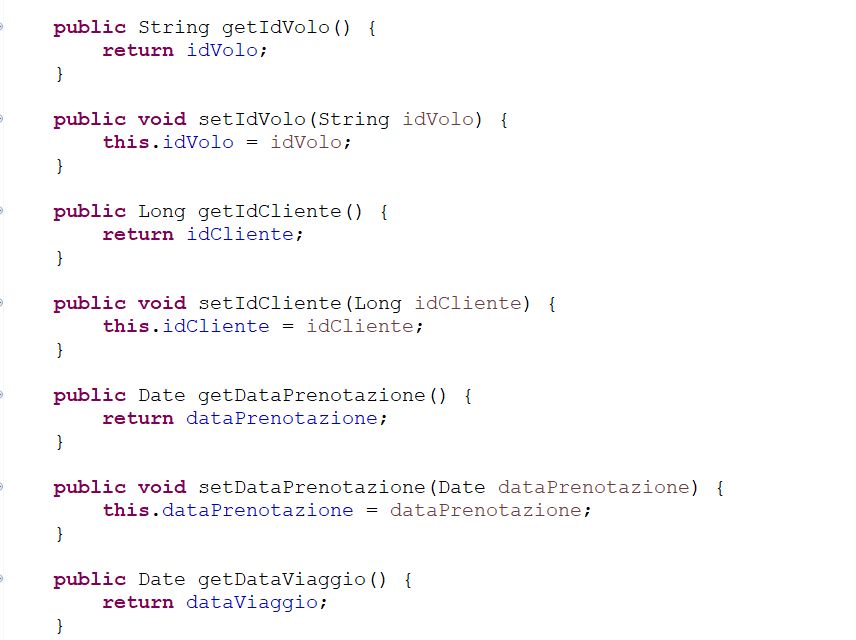
2.1 Sviluppo delle Entity

La prima cosa che ho sviluppato sono state le entity grazie alle quali possiamo collegarci alle tabelle del nostro database.

Esempio di Entity:



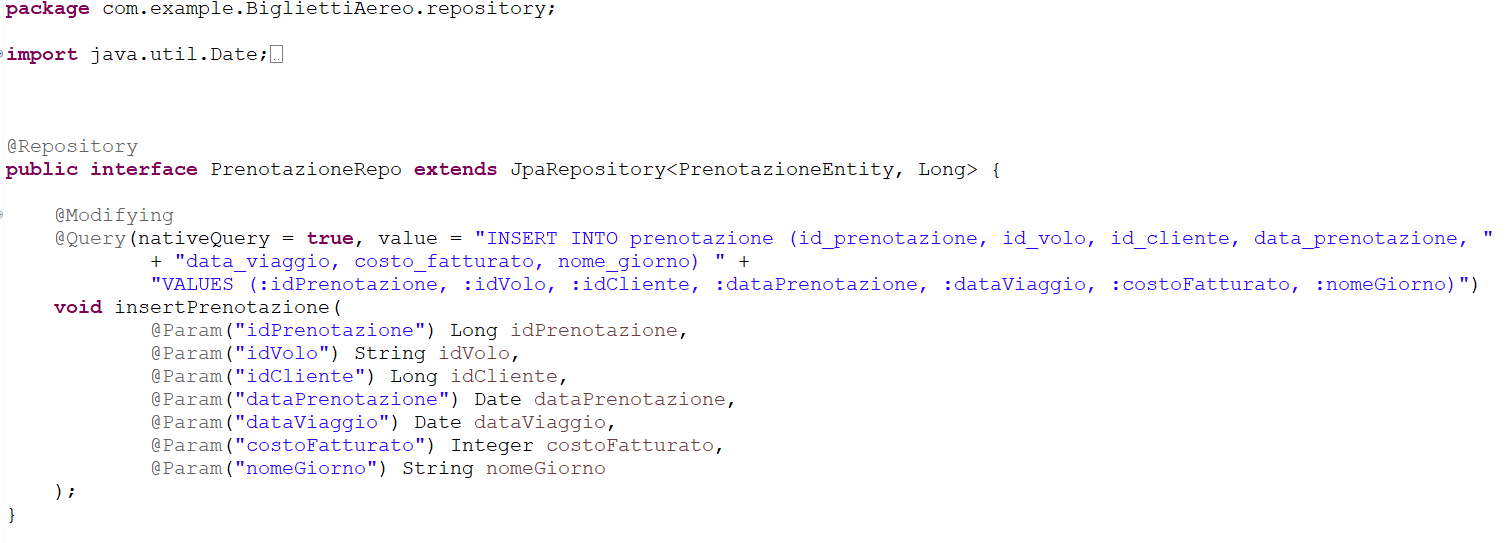




2.2 Sviluppo dei Repository

Dopo aver sviluppato tutte le Entity seguendo l’esempio caricato sono passato alla creazione di tutti i Repository dove ho creato la funzione per poter fare le insert.

Esempio di Repository:



2.3 Creazione di Service

Successivamente alla creazione dei miei repository sono passato alla creazione dei miei Service all’interno dei quali ho sviluppato tutti i servizi (Visualizzazione di tutti i campi di una tabella, ricerca tramite l’Id di una riga, Insert di una nuova riga, Update di una riga già esistente, Delete di una riga)

Esempio di un Service:

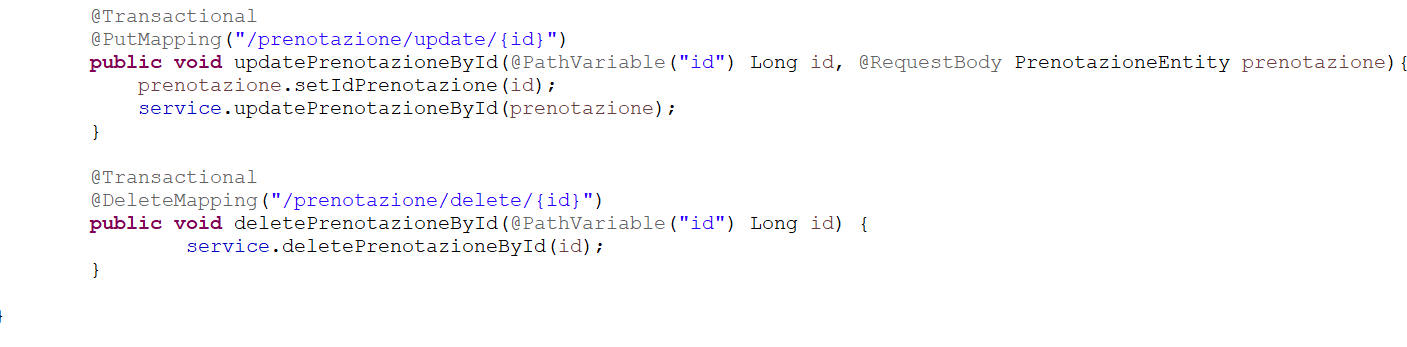


2.4 Creazione dei Controller

Infine, dopo aver creato tutti i Servizi che mi servivano sono passato a creare i controller per gestire le operazioni CRUD (Create, Read, Update, Delete).

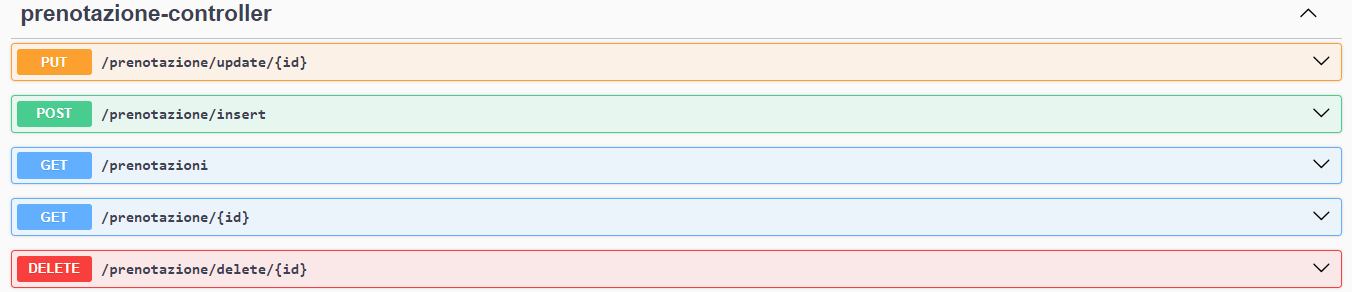
Esempio di controller:



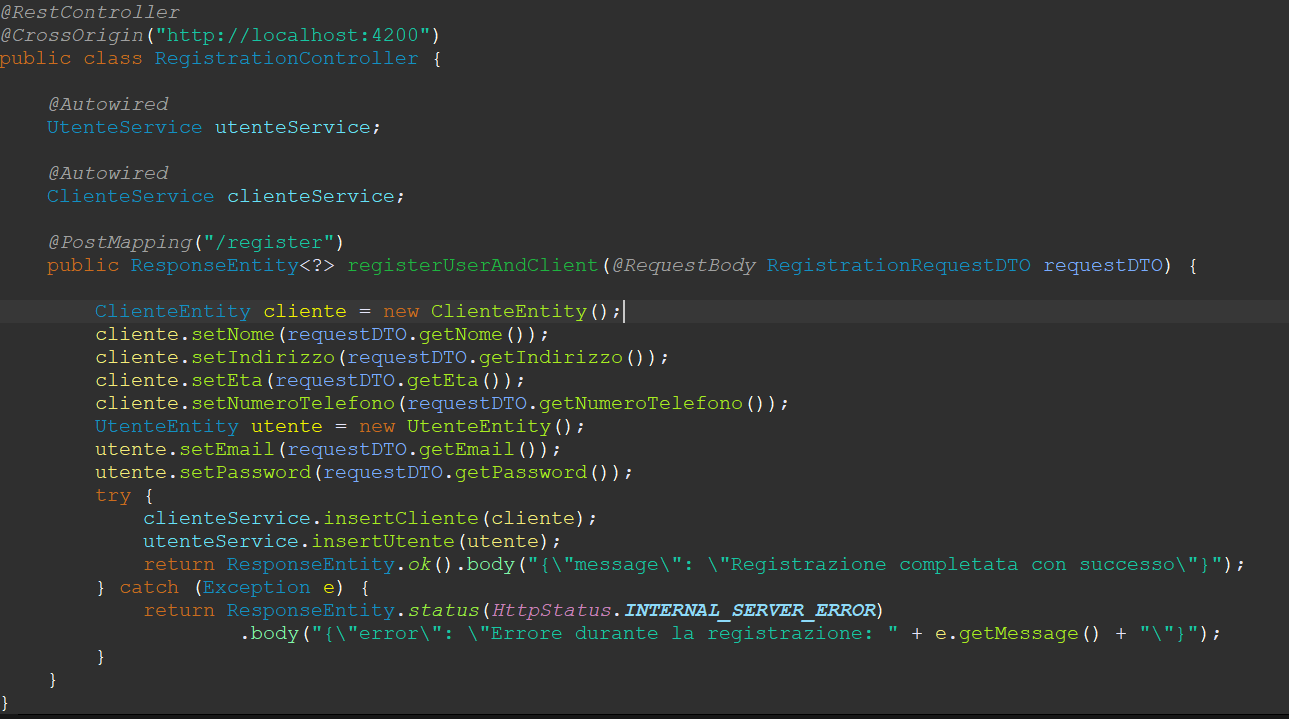


2.5 Visualizzazione ed esecuzione di tutti servizi su Swagger

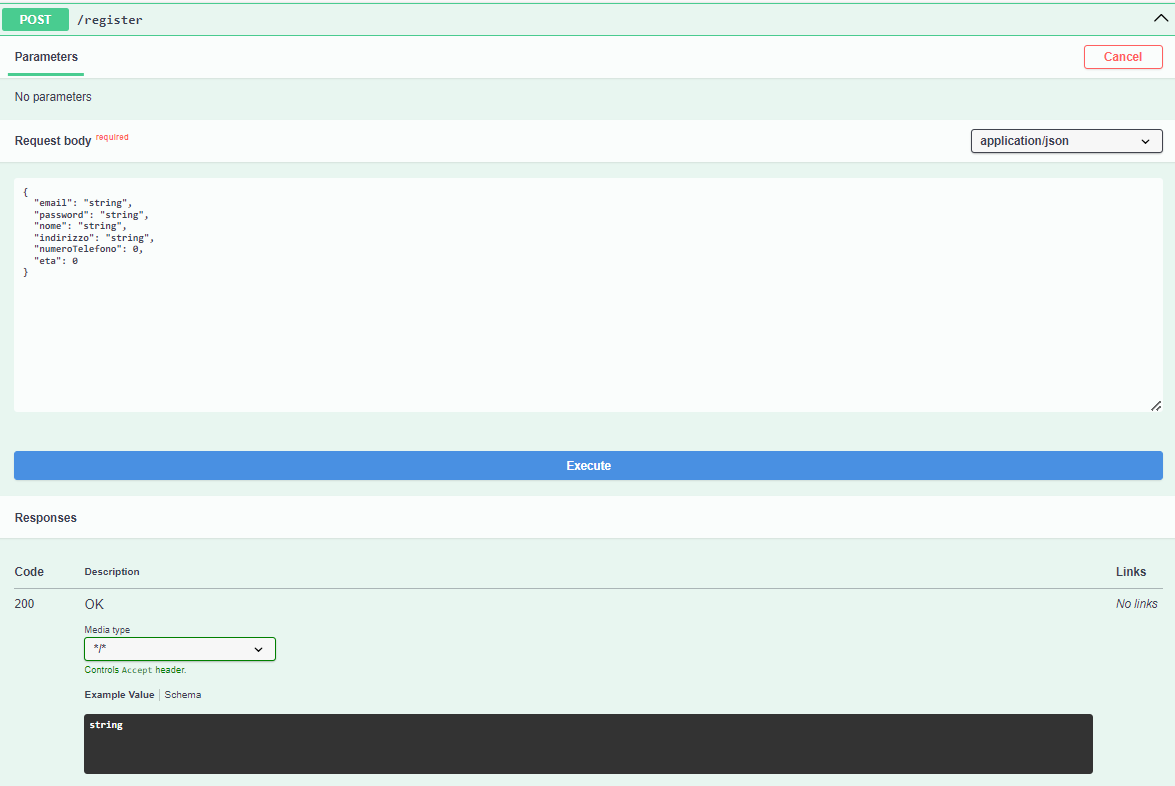
Dopo aver creato le entity, repository, service e controller possiamo mandare in esecuzione la nostra Java Application e usare swagger per verificare se funziona tutto correttamente e i campi vengono implementati nel database.



Dopo aver fatto questo procedimento per tutte le componenti che mi servivano ho sviluppato il controller per la registrazione che mi compila contemporaneamente Cliente e la nuova tabella utente che mi serivrà in seguito per il login i quanto contiene email e password.



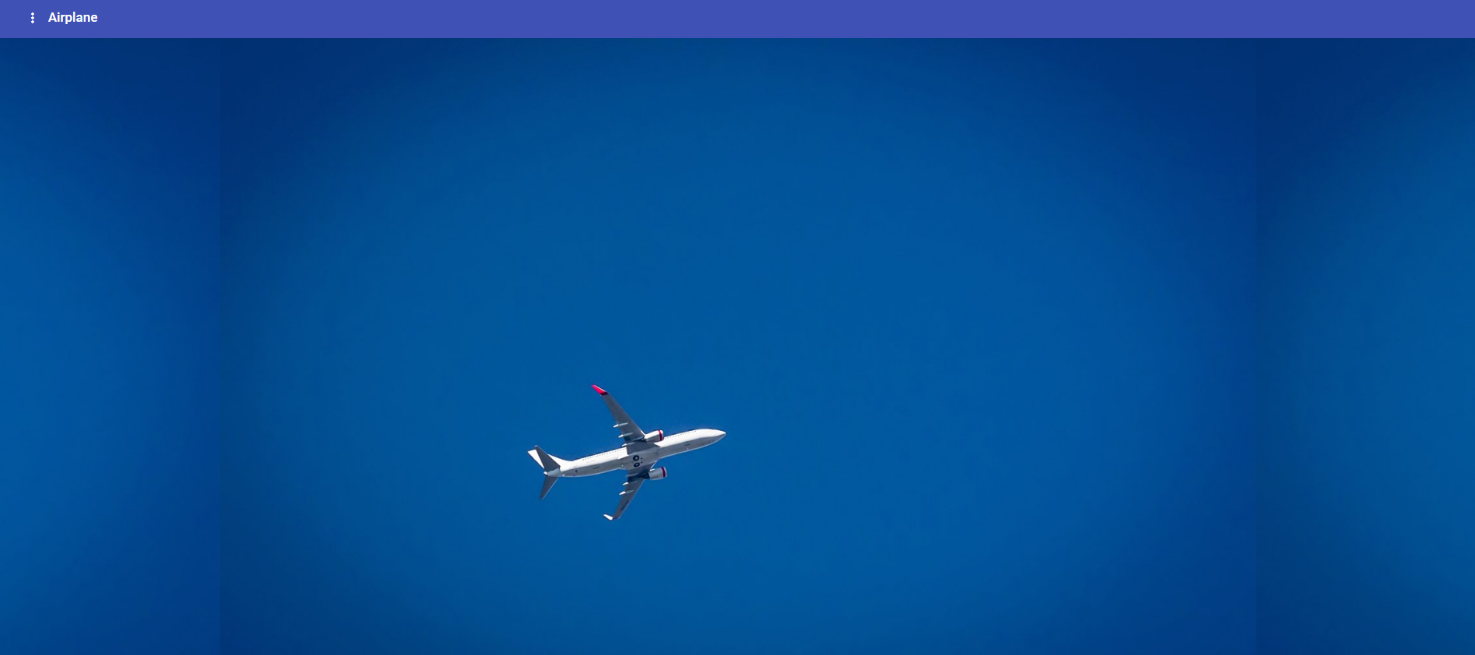
E su swagger si presenta in questo modo:



**FRONT-END**

* 1. pagina di home

Il front-end parte con la pagina di home



Qui troviamo un immagine di Back-Ground e in cima una toolBar istanziata dove e possibile aprire un menu dove troviamo la pagina di prenotazione, clienti, voli e anche login e registrati che invece di aprire una nuova pagina aprono una semplice finestra dove è possibile effettuare la registrazione e dopo di che il login.

