ADVANCED PROGRAMMING LANGUAGES – A.A. 2023-24

Progetto “Airline Manager” – Relazione finale

Ravale Roberto 1000044127, Vitali Gabriele 1000010255

Sommario

1. Caratteristiche generali

Airline Manager si pone come applicazione per il supporto delle attività di gestione di una generica compagnia aerea e di un'ampia varietà di funzionalità da essa offerte al cliente.

Nell'ambito dello sviluppo del codice è stata dedicata particolare attenzione ai seguenti aspetti:

- favorire riusabilità del codice, definendo metodi parametrizzabili, comuni a entrambi i membri, nel caso di operazioni affini da realizzare;
- favorire estendibilità dell'applicazione, grazie ad un'opportuna modularità del codice;
- favorire leggibilità e comprensibilità del codice, a partire da un'accurata organizzazione di quest'ultimo e all'utilizzo di funzioni ad alta coesione;
- accurata validazione dei valori, sia lato client sia lato server, realizzata attraverso l'uso di apposite regex;
- rendere configurabile l'applicazione, attraverso il settaggio di opportuni parametri di funzionamento, in base alla differente logica di business che si desidera applicare;
- continua e proficua collaborazione tra i membri del gruppo, con un processo decisionale costante e condiviso.

2. Funzionalità

Si riporta di seguito l'elenco delle funzionalità offerte da ciascuno dei tre moduli di cui l'applicazione si compone.

Modulo Client (Python)

Utente:

- acquisto di un biglietto (singolo) per un volo aereo;
- operazione di check-in, in accordo con le politiche della compagnia aerea;
- modifica prenotazione (aggiunta bagaglio, selezione posto a bordo), in accordo con le politiche della compagnia aerea;
- cancellazione prenotazione, in accordo con le politiche della compagnia aerea;

Admin:

- gestione autenticazione;
- gestione volo: inserimento, modifica, visualizzazione ed eliminazione dei dati di base (numero di volo, aeroporto di partenza, aeroporto di arrivo) relativi ad un certo codice volo;
- gestione schedulazione volo: per ciascun numero di volo (AS1234) inserimento, modifica, visualizzazione ed eliminazione delle ricorrenze periodiche (esempio: volo AS1234 di ogni lunedì)

Modulo Server (C#)

- implementazione della logica applicativa e delle interrogazioni al DB necessarie per la realizzazione delle funzionalità offerte dal modulo client, tenendo conto, altresì, delle politiche di business della compagnia aerea definite per ciascuna operazione;
- meccanismo di autenticazione dell'admin;
- logica di determinazione del prezzo del biglietto, attraverso il calcolo di differenti sovrapprezzi (sulla base del numero di posti disponibili rimasti per una certa ricorrenza e sulla base della stagione dell'anno, bassa, media o alta, entro cui ricade la stessa);

Modulo Server Statistiche (C++)

S'intende calcolare e fornire le seguenti statistiche:

- statistica 1: frequenza mensile di un determinato volo per un determinato anno;
- statistica 2: percentuale dei posti a sedere venduti rispetto al totale, per ciascuna ricorrenza, relativamente ad uno specifico volo di linea;
- statistica 3: frequenza dei voli della compagnia che decollano da un aeroporto in un determinato anno;
- statistica 4: guadagno mensile della compagnia, relativamente ad uno specifico anno.

3. Suddivisione lavoro

Gabriele Vitali:

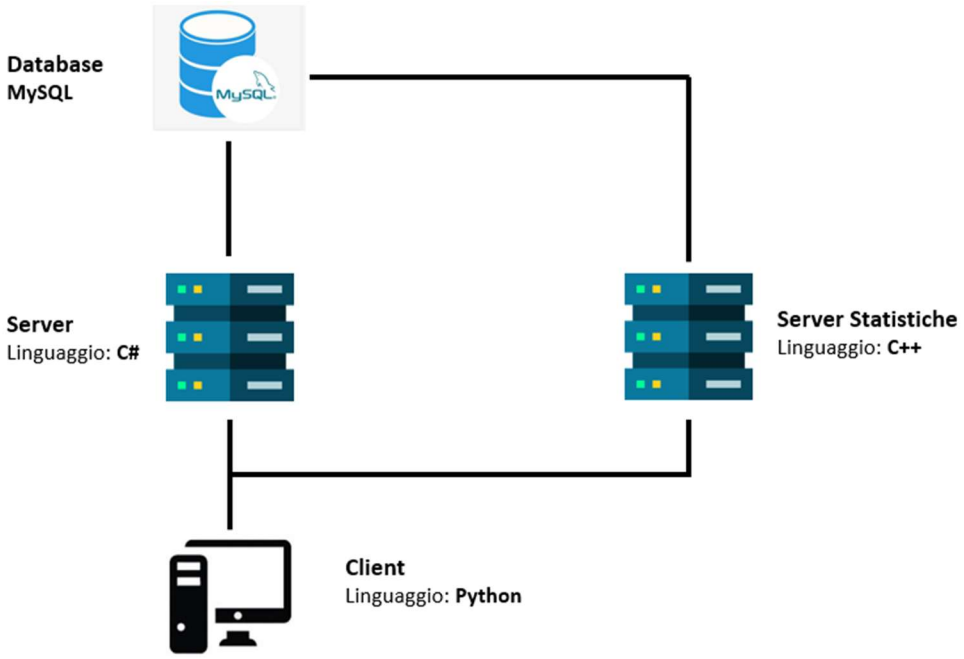
- acquisto di un biglietto aereo singolo (ricerca voli aerei, inserimento nuova prenotazione, inserimento nuovo passeggero, aggiornamento ricorrenza), con duplice modalità di ricerca dei voli e meccanismo di calcolo di opportuni sovrapprezzi;
- check-in;
- cancellazione di una prenotazione;
- visualizzazione carta d'imbarco;
- statistica 2;
- statistica 4;

Roberto Ravale:

- Modulo admin (Login, inserimento volo, eliminazione volo, inserimento ricorrenza, modifica ricorrenza, eliminazione ricorrenza e visualizzazione delle ricorrenze di un volo);
- Inserimento bagaglio;
- Modifica posto a sedere;
- Statistica 1;
- Statistica 3;

4. Architettura e meccanismo di comunicazione

L'elenco degli endpoint è definito all'interno dei file config.py per l'ambiente Python e Configurazione.cs per l'ambiente C#.
Il client, a seconda della specifica funzionalità richiesta, stabilisce una connessione con il server principale o con il server dedicato alle statistiche, interagendo con uno di essi per volta mediante protocollo HTTP
Entrambi i server accedono al medesimo database MySQL per il recupero dei dati necessari al completamento delle operazioni richieste.
Non esiste alcuna forma di comunicazione diretta tra il server principale e il server statistiche



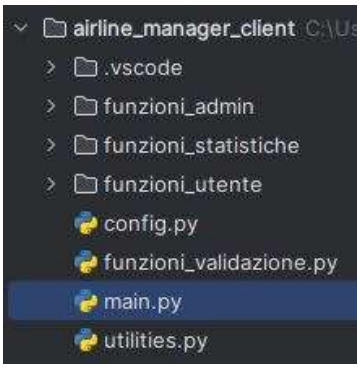
5. Organizzazione codice

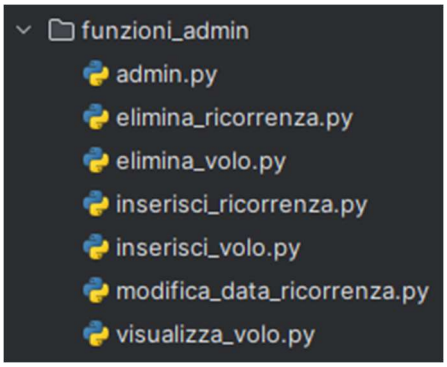
Si segnala che, all'interno dei file contenenti metodi in comune, è stato specificato mediante opportuni commenti quali dei due membri del gruppo abbia sviluppato un certo metodo (**RR**: Roberto Ravale; **GV**: Gabriele Vitali).

Codice Python

Le funzionalità offerte dal client sono state raccolte in tre macrogruppi distinti (utente, admin e statistiche), a cui corrispondono tre folder differenti.
Ciascuno di questi include:

- file contenente routine dedicata (utente.py, admin.py, statistiche.py), che implementa il menù attraverso cui è possibile accedere alle funzionalità specifiche di ciascun macrogruppo;
- un file per ciascuna funzionalità offerta.

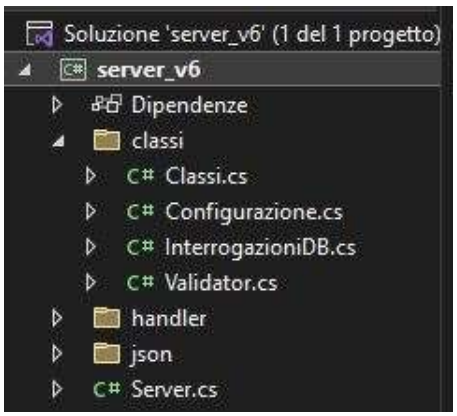




La cartella generale del progetto Python, invece, comprende:

- file **main.py**, contenente la routine di avvio, attraverso cui è possibile accedere alle tre distinte categorie di funzionalità precedentemente citate;
- file **config.py**, contenente elenco degli url e dei path necessari per contattare server e server statistiche;
- file **funzioni_validazione.py**, contenente un'ampia raccolta di funzioni di validazione utilizzate per la verifica della correttezza dei valori di input, lato client;
- file **utilities.py**, contenente una raccolta eterogenea di funzioni utilizzabili da entrambi i membri.

Codice C#



Il metodo **Main** è contenuto in **Server.cs**. In quest'ultimo vengono gestite le differenti richieste provenienti dal client, ciascuna delle quali è associata univocamente ad un certo listener, cui è stato assegnato uno specifico endpoint.

Il server è stato implementato in maniera da tale da supportare il multithreading: in particolare, avvia un nuovo thread per ogni richiesta ricevuta, permettendo così di gestire più richieste contemporaneamente.

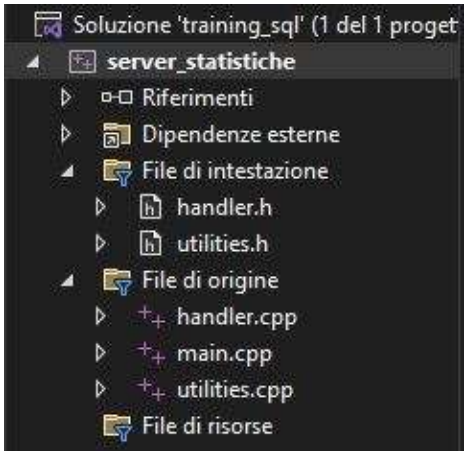
Ciascuna richiesta ricevuta dal server viene gestita da un apposito metodo offerto dalla classe **Handler**. Ciascuno di tali metodi è stato inserito in file .cs individuali (ad esempio, il file HandlerCheckIn.cs contiene il metodo handler che si occupa di servire la richiesta di check-in proveniente dal client). L'insieme di questi file è presente nel folder "Handler".

In quest'ultimo è presente anche il file **Handler.cs** che, invece, contiene una raccolta delle sottofunzioni sfruttate nei metodi handler sopra citati.

Nella cartella generale del progetto, si segnala anche il folder "Classi", contenente i seguenti file:

- **InterrogazioniDB.cs**: definisce la classe InterrogazioniDB, che offre un'ampia raccolta di metodi di interrogazione del database MySQL, sfruttati dai diversi metodi handler;
- **Configurazione.cs**: definisce la classe Configurazione, contenente l'elenco degli endpoint associati ai diversi listener e un'ampia serie di parametri di funzionamento, che è possibile impostare allo scopo sia di modificare sia di estendere il comportamento dell'applicazione, adattandola alla specifica logica di business che si desidera applicare;
- **Validator.cs**: definisce la classe Validator che contiene un'ampia raccolta di funzioni di validazione utilizzate per la verifica della correttezza dei valori provenienti dal client;
- **Classi.cs**: definisce un elenco di classi (Volo, Ricorrenza, Prenotazione, Passeggero, Admin) utilizzate in fase di deserializzazione del json contenuto all'interno delle richieste HTTP POST inviate dal client e ricevute dal server;

Codice C++



Il metodo **main** è contenuto nell'omonimo file .cpp.

L'organizzazione del codice del server statistiche c++ è del tutto analogo a quella del server in c#.

In questo caso le funzioni handler sono state raggruppate negli omonimi file .h e .cpp, mentre i metodi di interrogazioni del database MySQL sono stati definiti in utilities.h e utilities.cpp.

6. Guida all'installazione

Client Python

Per l'ambiente Python, è stata adottata la versione 3.12.1.

Al fine di garantire una gestione efficace della comunicazione HTTP con i server è richiesta l'installazione della libreria ***Request***

- Commando istallazione: *pip install requests*

Server C#

Per lo sviluppo del server principale, è stata utilizzata la versione .NET 8.0

Per utilizzare una versione .NET differente:

- Accedere alla directory del server e individuare il file ***"server_v6.csproj"***
- Modificare il contenuto del tag ***"<TargetFramework>"*** con la versione desiderata
    - Es: <TargetFramework>net7.0</TargetFramework>

Server C++

Per la configurazione del server statistiche, si richiede l'installazione dei seguenti componenti:

- **Mysql Connector**
    - Disponibile per il download al seguente indirizzo: https://dev.mysql.com/downloads/connector/cpp/
- Libreria **cpp-httplib** (versione: 0.14.1)
    - L'installazione può essere effettuata utilizzando il gestore di pacchetti **vcpkg** con il comando:
    - .\vcpkg install cpp-httplib
- Libreria **nlohmann-json** (versione: 3.11.3)
    - Analogamente, l'installazione può essere effettua mediante vcpkg:
    - .\vcpkg install nlohmann-json

Per configurare il MySQL Connector con Visual Studio 2022, seguire i seguenti passi:

1. Aprire il progetto “server statistiche” in Visual Studio.
2. Navigare in “*Progetto -> Proprietà di server_statistiche”*
3. Nelle impostazioni di configurazione, apportare le seguenti modifiche:
    a. C++ -> Generale -> Directory di inclusione aggiuntive
        i. Aggiungere il percorso di installazione del connettore MySQL
        ii. ES: C:\mysql-cpp-8.3.0\include\jdbc;
    b. C++ -> Preprocessore -> Definizioni Preprocessore
        i. Inserire: *STATIC_CONCPP;*
    c. Linker -> Generale -> Directory librerie aggiuntive
        i. Inserire: C:\mysql-cpp-8.3.0\lib64\vs14
    d. Linker -> Input ->Dipendenze Aggiuntive
        i. Inserire: mysqlcppconn-static.lib