**Documentazione di HermesChat’s Server**

Il server che sta alla base del progetto di HermesChat è stato sviluppato in Go, un linguaggio open source sviluppato da Google, che utilizza il framework Gin-gonic per gestire le chiamate API e la libreria Gorilla Websocket per i socket. Riguardo al salvataggio dei dati si appoggia a due tecnologie Database che sono MongoDB (gestito dal mio compagno e collega Molino) e Redis.

Il database Redis viene hostato in locale su un container Docker e si occupa principalmente di salvare gli UUID di ogni socket legato ad un utente per permettere di inviare i vari messaggi e le varie notifiche a tutti i dispositivi su cui lui è loggato.

I dati degli utenti vengono invece salvati per l’appunto sulla piattaforma Atlas di MongoDB.

Il server da me realizzato, come ogni progetto in Go, è suddiviso in package che sono:

* Main 🡪 che si occupa di avviare il progetto, l’HTTPS Server e di richiamare le varie funzioni del pacchetto config e per il setup delle API routes;
* Config 🡪 che si occupa di carica i valori delle variabili nel file .env, collegarsi al database di MongoDB, quello di Redis, ottenere l’indirizzo IP locale e aggiungere l’indirizzo pubblico del dispositivo che hosta il server nel campo A del DNS di Godaddy (tramite la loro API) per fare in modo che, se c’è stato il binding sul router della porta con l’indirizzo IP e porta del pc, si aggiorna il collegamento a https://api.hermeschat.it che è l’indirizzo del server;
* Router 🡪 che si occupa di collegare le routes ai loro handler e di aggiungere il middleware delle CORS policy e quello dell’autenticazione con JWT;
* Controllers 🡪 che si occupa di verificare che i parametri presi in input siano effettivamente corretti, successivamente vengono inviati alle varie funzioni presenti nel package utils. Inoltre al suo interno sono presenti i commenti che serviranno per la creazione della documentazione API con Swagger
* Models 🡪 che si occupa di fornire delle strutture ben definiti che contengono le variabili passate come parametri nelle varie chiamate (principalmente quelle POST)
* Utils 🡪 è dove sono scritte le principali funzioni del server e la maggior parte del codice che si occupa di tutto, dall’aggiunta di record nel DB alla gestione della ricerca e dell’invio dei dati richiesti.

Ogni package viene suddiviso ovviamente in più file, principalmente per ordine e per funzionalità, come ogni file nel package utils contiene determinate funzioni, c’è quello del calendario, quello delle chat e così via.

<https://192.168.1.100:8090/docs/index.html> 🡪 Swagger API Documentation