**COME UTILIZZARE IL BOT**

* **Funzione search\_song:** Separa la logica di ricerca in una funzione per una migliore organizzazione.
* **Utilizzo di FFmpegAudioSource:** Per una maggiore compatibilità con diversi formati audio.
* **Comando !skip:** Permette di saltare alla canzone successiva.
* **Comando !queue:** Mostra la coda delle canzoni.
* **Evento on\_song\_end:** Gestisce la riproduzione automatica della prossima canzone nella coda.
* **Gestione degli errori:** Include una gestione di base degli errori per evitare crash del bot.

**Come usare il bot:**

1. **Invitalo nel tuo server Discord.**
2. **Entra in un canale vocale.**
3. **Digita !play + nome della canzone o URL per aggiungere una canzone alla coda.**
4. **Utilizza !skip, !pause, !resume e !queue per controllare la riproduzione.**

**Considerazioni:**

* La riproduzione di contenuti protetti da copyright potrebbe essere illegale.
* Prova a chiedere al bot “Come stai?” oppure “Chi Sei?”, sono domande che gli piacciono.

**Spiegazione codice**

**Librerie Importate**

* **os:** Interagisce con il sistema operativo (per caricare variabili d'ambiente).
* **discord:** Libreria principale per creare bot Discord.
* **dotenv:** Carica variabili d'ambiente da un file .env.
* **flask:** Framework web per creare l'applicazione.
* **requests:** (Non usato nel codice fornito) Per fare richieste HTTP a siti web.
* **youtube\_dl:** Scarica video da YouTube (utilizzato per ottenere informazioni sulle canzoni).
* **discord\_player:** Libreria specializzata per gestire la riproduzione musicale nei canali vocali di Discord.
* **asyncio:** Per gestire operazioni asincrone (utilizzato per mettere in pausa la riproduzione tra le canzoni).

**Applicazione Flask (app)**

* **@app.route("/"):** Definisce la pagina principale dell'applicazione web. Quando visiti l'URL principale dell'app, vedrai il contenuto del file pagina.html.

**Funzionalità Bot Discord (bot)**

* **search\_song(search):** Funzione asincrona che cerca una canzone su YouTube utilizzando youtube\_dl. Restituisce un dizionario con l'URL della sorgente audio e il titolo della canzone.
* **@bot.command():** Decoratore che indica che le funzioni seguenti sono comandi del bot Discord.
  + **play(ctx, \*, search):** Comando !play. Si connette al canale vocale dell'utente, cerca la canzone richiesta, la aggiunge alla coda del player e invia un messaggio di conferma.
  + **skip(ctx):** Comando !skip. Salta alla canzone successiva nella coda.
  + **pause(ctx):** Comando !pause. Mette in pausa la riproduzione.
  + **resume(ctx):** Comando !resume. Riprende la riproduzione.
  + **queue(ctx):** Comando !queue. Mostra la lista delle canzoni in coda.
* **@bot.event:** Decoratore che indica che le funzioni seguenti sono gestori di eventi del bot.
  + **on\_ready():** Evento attivato quando il bot è connesso e pronto. Stampa un messaggio di conferma.
  + **on\_song\_end(ctx, song):** Evento attivato quando una canzone finisce di essere riprodotta. Aspetta un secondo e poi, se la coda non è vuota, riproduce la prossima canzone.

**Caricamento del Token e Avvio del Client Discord**

* **load\_dotenv():** Carica le variabili d'ambiente dal file .env.
* **TOKEN = os.getenv('DISCORD\_TOKEN'):** Legge il token del bot Discord dal file .env.
* **intents = discord.Intents.default() e intents.message\_content = True:** Configura gli intents del client Discord per consentirgli di accedere ai contenuti dei messaggi.
* **client = discord.Client(intents=intents):** Crea un nuovo client Discord.

**Comandi di esempio su discord:**

* Digita !play nome della canzone per cercare e riprodurre una canzone.
* Digita !skip per saltare alla canzone successiva.
* Digita !pause per mettere in pausa la riproduzione.
* Digita !resume per riprendere la riproduzione.
* Digita !queue per vedere la lista delle canzoni in coda.