DOCUMENTAZIONE :

-REFACTORING SPOTIFY

-SECONDA CONSEGNA

Membri: D’Angelo Luca, Rossi Tommaso, Labarba Samuel, Grassi Giovanni

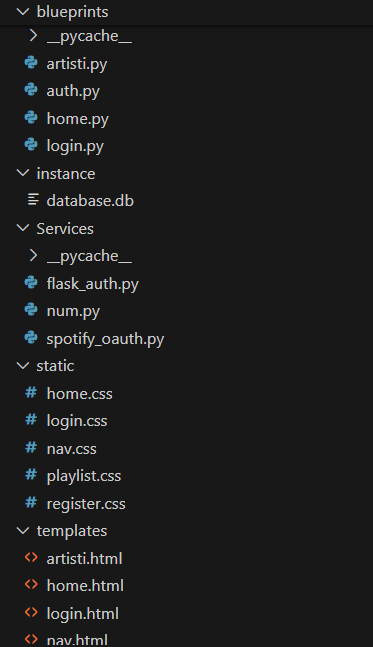
DESCRIZIONE PROGETTO

Questo progetto è una web app sviluppata in Flask che si integra con l'API di Spotify tramite OAuth. Gli utenti possono registrarsi e autenticarsi con credenziali locali, per poi collegare il proprio account Spotify. Una volta autenticati, l'applicazione permette di:

* **Visualizzare le proprie playlist Spotify:** vengono recuperate e mostrate informazioni quali nome, numero di brani e copertina.
* **Ricercare playlist pubbliche:** con la possibilità di salvare le playlist preferite nel database, evitando duplicati.
* **Esplorare artisti popolari:** è presente una sezione dedicata che visualizza una classifica dei 50 artisti globali più popolari, completa di immagine e informazioni sulla popolarità.

Il progetto è strutturato in maniera modulare tramite Blueprint, utilizzando template Jinja per la visualizzazione dinamica dei dati musicali.

RISTRUTTURAZIONE PROGETTO



REFACTORING SPOTIFY

FUNZIONE AGGIUNTA (fatta dal Giovanni)

La funzione implementata nella pagina **"artisti.html"** serve a visualizzare una lista dei **50 artisti più popolari** a livello globale, ottenuti tramite una query all'API di Spotify. Quando un utente visita questa pagina, viene effettuata una ricerca che recupera gli artisti più popolari del 2025, ordinati per popolarità e per ogni artista, viene mostrato:

* **Nome** dell'artista
* **Immagine** dell'artista (se disponibile)
* La **popolarità** dell'artista

FUNZIONI RICHIESTE

### **1.Ristrutturare l’architettura dell’app per far in modo che l’autenticazione utente con OAuth2 sia opzionale (fatta da Samu):**

Per rendere l'autenticazione utente con OAuth2 opzionale, sono state effettuate modifiche. Ora l'app consente agli utenti di esplorare funzionalità come la ricerca delle playlist pubbliche e la visualizzazione degli artisti senza la necessità di effettuare il login tramite Spotify. Se l'utente sceglie di autenticarsi, viene concesso l'accesso a funzionalità avanzate come la visualizzazione delle proprie playlist. L'autenticazione è gestita tramite sessioni e token di accesso, ma non è obbligatoria per l'uso delle funzioni di base.

### **2. Creare una barra di ricerca per cercare le playlist pubbliche e visualizzarle (fatta da Luca):**

È stata aggiunta una **barra di ricerca** nella homepage, che consente agli utenti di cercare **playlist pubbliche** su Spotify. L'utente può inserire una query, e l'app invia una richiesta all'API di Spotify per ottenere le playlist corrispondenti. I risultati della ricerca vengono poi visualizzati sotto forma di un elenco, con dettagli come il nome della playlist, il proprietario e l'immagine di copertura (se disponibile). Gli utenti possono anche cliccare su ogni playlist per visualizzare i brani al suo interno.

SECONDA CONSEGNA

FUNZIONI RICHIESTE:

### **1.Aggiungere un sistema di autenticazione tramite flask-login (Fatta da Tommy):**

Per implementare l'autenticazione, è stato integrato Flask-Login, che gestisce in modo sicuro il login, logout e la protezione delle rotte. Il sistema consente agli utenti di registrarsi e autenticarsi utilizzando username e password, e sfrutta decorator come @login\_required per garantire l'accesso esclusivo alle funzionalità riservate. Il meccanismo di sessione mantiene lo stato di autenticazione, migliorando la sicurezza e l'usabilità dell'applicazione.

**2.Creare un db relazionale dove salvare i dati degli utenti e le playlist salvate per ogni utente tramite pymysql(fatta da Luca)**

È stato realizzato un database relazionale che utilizza PyMySQL per connettersi a MySQL (o MariaDB). La struttura del database prevede tabelle dedicate agli utenti e alle playlist, con relazioni definite tramite chiavi esterne per associare ogni playlist salvata all'utente corrispondente. Questo approccio assicura l'integrità dei dati e consente di gestire e interrogare le informazioni in modo efficiente e scalabile.

### **3.L'analisi dei dati delle playlist, con visualizzazione tramite grafici creati con Pandas e una libreria scelta per creare e visualizzare i grafici (es. Plotly).(fatta da Samu):**

È stata implementata una funzionalità per l'analisi dei dati delle playlist utilizzando Pandas per l'elaborazione e Plotly per la visualizzazione interattiva. I dati relativi a metriche come il numero di brani e la popolarità vengono trasformati e rappresentati graficamente, offrendo agli utenti insight visuali chiari e dinamici. I grafici interattivi permettono di esplorare in dettaglio le informazioni, migliorando l'esperienza utente e l'interpretazione dei dati musicali.

SCHEMA ER E LOGICO



Schema logico

User (Id, Username, Password)

Playlist (Id, Name, Owner, image User\_id)