DOCUMENTAZIONE :

-REFACTORING SPOTIFY

-SECONDA CONSEGNA

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

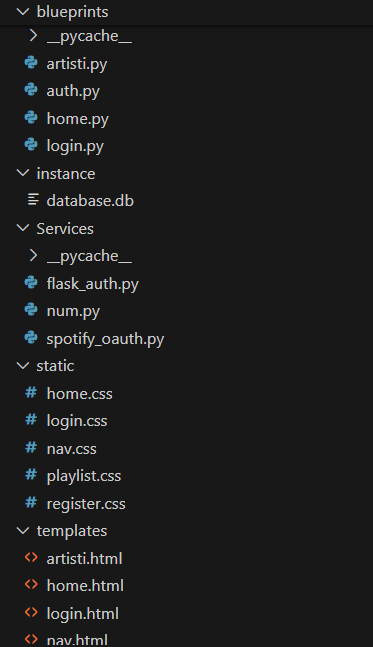
DESCRIZIONE PROGETTO

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)

REFACTORING SPOTIFY

TERZA CONSEGNA

Membri: D’Angelo Luca, Rossi Tommaso, La Barba 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.  
• Analizzare e confrontare playlist, grazie a grafici interattivi realizzati con Pandas e Plotly.

L'app è modulare grazie ai Blueprint e utilizza i template Jinja per la visualizzazione dinamica dei dati.

RISTRUTTURAZIONE PROGETTO

L'architettura è stata ulteriormente ottimizzata per supportare nuove funzionalità avanzate di analisi dati e confronto playlist, mantenendo al tempo stesso l'accesso semplice per gli utenti non autenticati.

GRAFICI E ANALISI – TERZA CONSEGNA

Distribuzione Temporale dei Brani *(Fatta da Tommy)*

Descrizione: Mostra quanti brani della playlist sono stati pubblicati ogni anno.  
Dati utilizzati: Data di pubblicazione dei brani estratta da Spotify.  
Motivazione grafico: Un grafico a barre è perfetto per visualizzare la quantità di brani pubblicati in ogni anno, rendendo evidente se la playlist ha una forte componente "nostalgica" o è più orientata a brani recenti.

Distribuzione della Durata dei Brani *(Fatta da Tommy)*

Descrizione: Analizza la variabilità della durata dei brani all'interno della playlist.  
Dati utilizzati: Durata di ciascun brano (in millisecondi).  
Motivazione grafico: Un istogramma permette di vedere se ci sono durate standardizzate o picchi particolari (es. brani molto brevi o lunghi).

Distribuzione della Popolarità *(Fatta da Luca)*

Descrizione: Illustra il livello di popolarità dei brani.  
Dati utilizzati: Valori di popolarità forniti da Spotify (0–100).  
Motivazione grafico: Un grafico a barre evidenzia la frequenza di ciascun livello di popolarità e consente di capire se la playlist contiene prevalentemente hit o brani di nicchia.

Distribuzione dei Generi Musicali *(Fatta da Luca)*

Descrizione: Mostra i generi musicali prevalenti nella playlist.  
Dati utilizzati: Generi associati agli artisti presenti nella playlist.  
Motivazione grafico: Un grafico a torta o a barre visualizza la varietà dei generi, utile per capire la diversità o l’uniformità stilistica della playlist.

Evoluzione della Popolarità nel Tempo *(Fatta da Gio)*

Descrizione: Unisce anno di pubblicazione e popolarità per ogni brano.  
Dati utilizzati: Data di pubblicazione + valore di popolarità.  
Motivazione grafico: Un grafico a linee è ideale per osservare tendenze temporali e variazioni nella scelta musicale: se la playlist mescola brani vecchi ma popolari o segue mode contemporanee.

ANALISI COMPARATIVE TRA PLAYLIST

Brani in Comune *(Fatta da Luca)*

Descrizione: Identifica i brani presenti in entrambe le playlist.  
Dati utilizzati: Titoli e ID dei brani.  
Output: Percentuale di somiglianza calcolata sul totale dei brani della playlist più piccola.  
Motivazione grafico: Un grafico a barre impilate visualizza il numero totale di brani, quelli in comune e la percentuale.

Artisti in Comune *(Fatta da Tommy)*

Descrizione: Evidenzia gli artisti presenti in entrambe le playlist.  
Dati utilizzati: Nomi degli artisti.  
Motivazione grafico: Un grafico a barre doppie mostra la frequenza di apparizione per ogni artista in ciascuna playlist.

Confronto della Popolarità Media dei Brani *(Fatta da Luca)*

Descrizione: Confronta la popolarità media delle due playlist.  
Dati utilizzati: Media dei valori di popolarità.  
Motivazione grafico: Un grafico a colonne affiancate consente di visualizzare chiaramente quale playlist è mediamente più popolare.

Confronto dei Generi Musicali *(Fatta da Luca)*

Descrizione: Mostra i generi musicali prevalenti nelle due playlist.  
Dati utilizzati: Generi associati agli artisti.  
Motivazione grafico: Un grafico a barre raggruppate mostra i generi comuni e le loro frequenze, evidenziando differenze di stile tra le playlist.

Distribuzione Temporale dei Brani (Comparativa) *(Fatta da Gio)*

Descrizione: Confronta le due playlist in base all’evoluzione temporale dei brani.  
Dati utilizzati: Anno di pubblicazione.  
Motivazione grafico: Un grafico a linee doppie mette in risalto la distribuzione cronologica dei brani nelle due playlist, utile per capire l’arco temporale coperto.

SCHEMA ER E LOGICO

Schema logico

User (Id, Username, Password)  
Playlist (Id, Name, Owner, Image, User\_id)  
Track (Id, Title, Artist, Duration, Popularity, ReleaseDate, Genre, Playlist\_id)

REFACTORING SPOTIFY

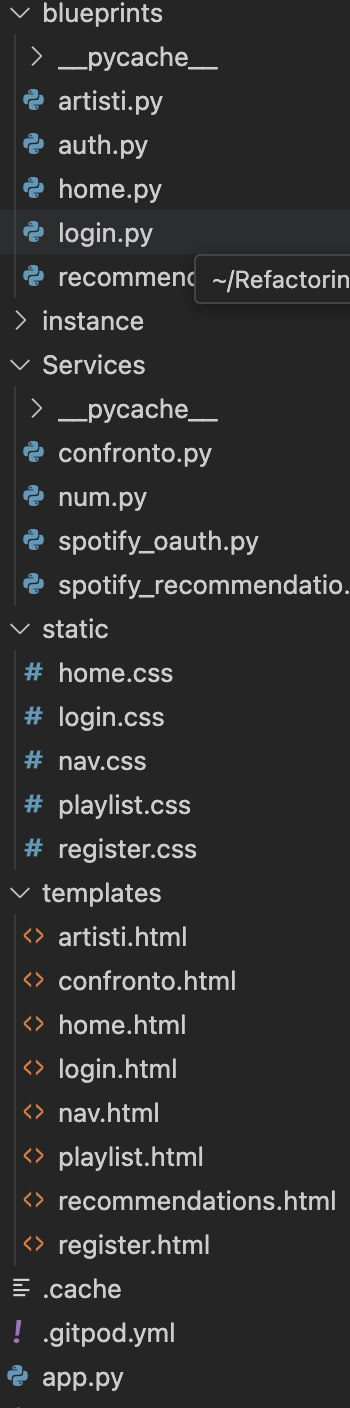
QUARTA CONSEGNA

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.  
• Analizzare e confrontare playlist, grazie a grafici interattivi realizzati con Pandas e Plotly.

L'app è modulare grazie ai Blueprint e utilizza i template Jinja per la visualizzazione dinamica dei dati.

• Possibilità di condividere una playlist copiando il suo link o di aprire la playlist in una pagina esterna direttamente su Spotify.

RISTRUTTURAZIONE PROGETTO



FUNZIONI RICHIESTE:

**Utilizzare l’endpoint GET /recommendations delle spotify web API (Fatto da Giovanni):**

grazie a questa funzione aggiunta ora l’utente, tramite gli id di un’artista o di una traccia può visualizzare dei brani consigliati utilizzando la funzione GET /recommendations che però a quanto pare spotify a disattivato come funzione

**Integrazione con altri servizi, Generazione di link condivisibili Spotify ( Fatto da Luca):**

**tramite questa funzione ora gli unteti possono copiare il link di una playlist o se vogliono possono anche aprire la playlist direttamente nella pagina di Spotify questo grazie al metodo playlist["external\_urls"]["spotify"].**