Backend Task Description — Mini Spotify Clone (3:30 Hours)

Objective

Build a backend REST API for a mini Spotify-like platform with **two roles**:

- **Admin** – Can add and delete albums and songs.
- **User** – Can browse albums, follow albums, and get notifications when new songs are added.

The project must include **authentication, authorization, caching, filtering, and aggregation**.

Key Features

1. Authentication & Authorization

- Implement user registration and login.
- Use JWT authentication (access + refresh tokens).
- Only **admins** can add, update, or delete albums/songs.
- Users can only perform browsing, following, and notification actions.

---

2. Admin Functionalities

- **Add Album** – Create a new album with metadata (title, artist, genre, year, etc.).
- **Delete Album** – Remove an album and its songs.
- **Add Song to Album** – Append songs to an existing album.
- **Delete Song** – Remove a song from an album.

---

3. User Functionalities

- **Browse Albums & Songs**
    - Filter by genre, artist, year.
    - Search by album title or artist name.
    - Sort results (e.g., by release year, popularity, song count).
    - Pagination for large datasets.
- **Follow/Unfollow Albums**
    - Follow an album to get updates when new songs are added.
    - Unfollow albums at any time.
- **View Followed Albums**
    - Get a list of albums the user follows.
- **Notifications**
    - Receive a list of recent updates (new songs) for followed albums.

---

4. Caching (node cache)

- Cache frequently accessed endpoints:
    - Album list
    - Album details
- Invalidate cache when albums or songs are modified.
- Use short TTL for caching (e.g., 2–5 minutes).

---

5. Aggregations & Analytics

- **Top Albums** – Return albums ranked by number of followers.
- **Genre Statistics** – Show total albums per genre.

- **Artist Timeline** – Show album release counts per year for a specific artist.

---

6. Validation & Security

- Validate all request bodies and query params.
- Rate limit requests to prevent abuse. (opt)
- Secure API with CORS and Helmet.
- Return consistent error responses.

---

7. Deliverables

- Fully working backend API with authentication, album management, following, and notifications.
- Caching and cache invalidation logic.
- Aggregation endpoints for analytics.
- Clear API documentation with request/response examples.
- `.env.example` file with all required environment variables.
- Short README with setup instructions.
