Music Website using Django – Project Synopsis

1. Project Title

Online Music Streaming Website using Django

2. Objective

To develop a dynamic, user-friendly music streaming web application using Django that allows users to listen to songs, create playlists, search music by genre/artist, and manage their accounts.

3. Introduction

Music streaming has become a primary mode of entertainment in the modern digital era. This project aims to build a music website using Django, which serves as a backend framework to manage audio content, user accounts, and playlists efficiently. The platform will also feature user authentication, song uploads (admin), and responsive UI.

4. Technologies Used

- Frontend: HTML5, CSS3, Bootstrap, JavaScript
- Backend: Python with Django Framework
- Database: SQLite (development), PostgreSQL (optional for production)
- Media Storage: Django Media Files
- Audio Support: HTML5 Audio Player

5. Functional Requirements

User Module:

- Register/Login/Logout
- Browse songs by genre, artist, or album
- Search songs
- Play/pause songs (using HTML5 player)
- Create and manage playlists
- Like/favorite songs

Admin Module:

- Login to admin panel
- Add/edit/delete songs
- Manage artists, albums, and categories
- Monitor user activity

6. Modules Breakdown

User Auth: Handles user registration and secure login/logout
Music Manager: Stores music metadata: title, artist, genre, album, file path
Playlist Manager: Users can create, save, and delete playlists
Search: Django ORM-based search on title, artist, and genre
Admin Panel: Django admin for managing all music content

7. Advantages

- Clean UI with responsive design
- Fully dynamic – easy to update
- Scalable: add support for podcast or video in future
- Role-based access control (User/Admin)
- Secure file upload and management system

8. Future Scope

- Integration with Spotify or YouTube APIs

- Premium subscriptions using Razorpay/Stripe
- Lyrics viewer and karaoke mode
- AI-based music recommendations
- Mobile app version with REST API backend

9. Timeline Estimate

Requirements & Planning – 2 days
Django Setup & Models – 2 days
User Authentication – 1 day
Music Player & UI – 2 days
Admin Panel Setup – 1 day
Testing & Debugging – 1 day
Total Duration – ~9 Days

10. Conclusion

This project is a full-stack web application that demonstrates how Django can be used to create powerful media-based websites. It integrates backend logic, frontend design, and audio management in one smooth platform. It’s ideal for beginners/intermediates in Django looking to explore real-world project development.