## **Music Player Website Development using MERN Stack & Spotify API**

### **Welcome:**

Hello Students, welcome to the Full Stack web development course! In this session, we will explore the development of a Music Player Website using the Spotify API and the MERN Stack.

### **About the Project:**

The **Music Player Website** project is designed to create a platform where users can:

* Sign up and sign in securely.
* Search for songs using the **Spotify API**.
* Manage their favorite tracks by liking and pinning them to personalized playlists.

The MERN Stack (MongoDB, Express.js, React.js, Node.js) is utilized to ensure a smooth user experience, with MongoDB handling data storage, Express and Node.js managing the backend logic, and React providing an interactive frontend.

## **Learning Objectives:**

By the end of this course, students will be able to:

* Set up and understand the architecture of a full-stack **MERN** application.
* Implement **user authentication** and secure routes using **JWT**.
* Integrate third-party APIs (**Spotify API**) to fetch and display song data.
* Develop interactive features for managing user playlists (liked and pinned songs).

### **Problem Statement:**

With the growth of music streaming platforms, users demand customizable and personalized music experiences that are fast, responsive, and user-friendly. Many existing platforms lack flexibility, requiring users to adapt to their limitations. This project addresses the need for a solution where users can not only browse and listen to music but also create, pin, and manage their own playlists in a way that is highly interactive and user centric.

### **Proposed Solution:**

The **Music Player Website** offers a flexible, personalized experience for music lovers by integrating the **Spotify API** into a full-stack web application. Users will be able to:

* **Sign up and Sign in** with secure authentication via **JWT**.
* **Search and Browse Songs** using the Spotify API.
* **Like and Pin Tracks** to save favorite songs.
* **Manage Playlists** by viewing liked and pinned songs on separate pages.

The solution leverages the **MERN stack** for its scalability, speed, and flexibility, ensuring that the platform can handle many users and interactions efficiently.

**Software & Hardware Requirements:**

Software:

1. Node.js (includes npm)
2. MongoDB Community Server
3. Visual Studio Code

Hardware:

1. Processor: Intel i3 or above
2. RAM: 8 GB
3. Storage: 256 GB SSD
4. Operating System: Windows

### **WEEK 1 - Backend Development and Frontend Development**

### **WEEK 2 - Spotify API Integration**

### **WEEK 3 - Adding Styles for Home Page and Features**

### **Summary:**

This project explored the development of a **Music Player Website** using the **MERN Stack**. The backend (Node.js and Express.js) manages user authentication and preferences, while **MongoDB** offers scalable storage. The frontend, built with **React.js**, provides a responsive interface, and the **Spotify API** enables dynamic song fetching.

Throughout the project, we emphasized:

* **Secure user authentication** using **JWT**.
* **API integration** with the **Spotify API**.
* **User-centric design** and interactive features such as liking and pinning songs.

This project has provided a strong foundation for building complex, scalable web applications, preparing you for advanced challenges in full-stack development.

### **Future Scope:**

Potential enhancements for the platform could include:

* Personalized recommendations based on user preferences.
* Social sharing of playlists and tracks.
* In-app music streaming.
* Scalability improvements for handling large datasets and real-time interactions.