

# 🎵 Music Streaming Web App (Digital Media Player)

## 📌 Project Aim

To design and develop a **full-stack music and podcast streaming web application** that allows users to stream audio content, manage playlists, and enjoy a modern digital media player experience similar to Spotify Lite. This project focuses on real-world architecture, clean UI, and industry-relevant skills within **2 weeks**.

---

## 👥 Use Cases

### User

- Register & login
- Browse music & podcasts
- Search audio content
- Stream music/podcasts
- Create & manage playlists
- Resume last played audio
- Like / favorite tracks

### Admin (Basic)

- Upload audio files
  - Add metadata (title, artist, category, cover image)
  - Manage tracks & podcasts
- 





## Tech Stack

### Frontend

- [React.js](#) or its framework
- Tailwind CSS or shadCN
- React Router
- HTML5 Audio API

### Backend

- Node.js
- Express.js

### Database & Services

- Supabase (PostgreSQL, Auth, Storage) or firebase or convex

### Deployment (Optional)

- Frontend: Vercel / Netlify
  - Backend: Render / Railway
- 



## Database Tables (High Level)

- users
  - tracks
  - podcasts
  - playlists
  - playlist\_tracks
  - recently\_played
-

17

# 2-WEEK DETAILED STEP-BY-STEP GUIDE

---

## ◆ WEEK 1: FOUNDATION & CORE FEATURES

### Day 1 – Project Setup & Planning

#### What to do:

- Finalize project features (music, podcast, playlist, player)
- Create GitHub repository
- Setup frontend using React (Vite or CRA)
- Install Tailwind CSS
- Setup backend with Node.js & Express
- Create Supabase project

#### Deliverables:

- Running frontend & backend servers
  - Project folder structure
- 

### Day 2 – Authentication System

#### What to do:

- Enable Supabase authentication (email/password)
- Build Signup & Login UI in React
- Integrate Supabase Auth SDK
- Create protected routes using React Router
- Store user session globally

#### Deliverables:

- Working authentication
  - User redirected after login
-



## Day 3 – Database Design & Backend APIs

### What to do:

- Design database tables in Supabase
- Create Express APIs:
  - GET /tracks
  - GET /podcasts
  - GET /categories
- Connect backend with Supabase client
- Test APIs using Postman

### Deliverables:

- Functional APIs
  - Data fetched successfully
- 



## Day 4 – Audio Player Core Logic

### What to do:

- Implement HTML5 Audio element
- Create global audio context/state
- Add Play / Pause / Seek functionality
- Create fixed bottom mini-player

### Deliverables:

- Audio plays across pages
  - Seek bar working
- 



## Day 5 – Music Listing & Playback

### What to do:

- Create music card components
- Fetch music list from backend
- Implement category filtering
- Play selected track via player

### Deliverables:

- Music browsing page
  - Play from list
- 

## Day 6 – Podcast Module

### What to do:

- Create podcast listing page
- Podcast detail page with episodes
- Play podcast episodes via player

### Deliverables:

- Podcast section functional
- 

## Day 7 – UI Refinement

### What to do:

- Improve UI spacing & colors
- Add loaders & empty states
- Make app fully responsive

### Deliverables:

- Polished UI
  - Mobile-friendly app
- 

## ◆ WEEK 2: ADVANCED FEATURES & DEPLOYMENT

## Day 8 – Playlist Feature

### What to do:

- Create playlist table & APIs
- Allow users to create playlists
- Add/remove tracks from playlist
- Playlist detail page

**Deliverables:**

- Fully working playlists
- 

 **Day 9 – Recently Played & Resume**

**What to do:**

- Track user listening history
- Save last playback time
- Resume from last position

**Deliverables:**

- Resume listening feature
- 

 **Day 10 – Admin Upload Panel**

**What to do:**

- Create admin-only upload page
- Upload audio files to Supabase Storage
- Save metadata to database

**Deliverables:**

- Admin content upload system
- 

 **Day 11 – Search Functionality**

**What to do:**

- Create search input UI
- Backend search API
- Debounced search queries

**Deliverables:**

- Search working for music & podcasts
-



## Day 12 – Optimization & Cleanup

### What to do:

- Reusable components
- Lazy loading routes
- Error handling
- Code cleanup

### Deliverables:

- Optimized production-ready code
- 



## Day 13 – Deployment

### What to do:

- Deploy frontend on Vercel/Netlify
- Deploy backend on Render
- Add environment variables

### Deliverables:

- Live project URL
- 



## Day 14 – Documentation & Resume Prep

### What to do:

- Write README.md
- Add screenshots
- API documentation
- Resume-ready project description

### Deliverables:

- Complete submission-ready project
-



## Learning Resources

- React Docs – react.dev
  - Supabase Docs – supabase.com/docs
  - Traversy Media (YouTube)
  - JavaScript Mastery (YouTube)
  - MDN Audio API
- 



## Industry Evaluation

**Industry Rating: 8.5 / 10**

This project demonstrates real-world skills, media handling, authentication, and full-stack development aligned with junior to mid-level developer expectations.

👉 If you add:

- Waveform visualization
- Recommendations
- Analytics dashboard

➡ It easily becomes **9 / 10**

---



## Future Enhancements

- Audio waveform visualization
  - Recommendations system
  - Analytics dashboard
  - Offline support
- 



This project is **resume-strong, interview-relevant, and industry-aligned**.