📄 Lyrix - Technical Documentation

Overview

Lyrix is a full-stack MERN (MongoDB, Express, React, Node.js) application that allows users to:

- Register and log in using JWT authentication.
- Search for song lyrics by title or artist.
- Save favorite song lyrics into custom-named playlists.
- Manage playlists entirely via **localStorage** on the frontend.

Architecture

Layer	Stack / Technology
Frontend	React (Vite + TypeScript)
Backend	Node.js, Express.js
Database	MongoDB Atlas
Authentication	JWT (JSON Web Tokens)
Storage	localStorage (browser) for playlists

Features

1. Authentication

- User **signup** and **login** using email and password.
- JWT token generated on successful login.
- Token stored in `localStorage` for authenticated requests.
- Protected API routes requiring token validation.

2. Song Lyric Search

- Users can search songs by **title** or **artist**.
- Results display song **title**, **artist**, and **lyrics**.
- Search handled by a RESTful `/api/songs` route on backend.

3. Playlist Management (Frontend only)

- Users can **save** searched songs into **custom playlists**.
- Users can **view** playlists and **delete** individual songs.
- Data persisted using **localStorage** (no backend storage yet).

Folder Structure

Backend (`/backend`)

```
/backend
├── models/
│   └── User.ts
│   └── Song.ts
├── routes/
│   └── auth.ts
│   └── songs.ts
├── middleware/
│   └── authMiddleware.ts
├── app.ts
├── server.ts
├── config/
│   └── db.ts
└── package.json
```

Frontend (`/frontend`)

```
/frontend
├── src/
│   ├── components/
│   │   └── Navbar.tsx
│   ├── pages/
│   │   └── Home.tsx
│   │   └── Signup.tsx
│   │   └── Login.tsx
│   │   └── Playlists.tsx
│   ├── context/
│   │   └── AuthContext.tsx
│   ├── services/
│   │   └── api.ts
│   ├── utils/
│   │   └── playlistStorage.ts
│   ├── types.ts
│   ├── App.tsx
│   └── main.tsx
├── index.html
└── package.json
```

Key APIs

POST `/api/auth/signup`

- Registers a new user.
- Body: `{ email, password }`
- Response: `{ token }`

POST `/api/auth/login`

- Logs in a user.
- Body: `{ email, password }`
- Response: `{ token }`

GET `/api/songs`

- Search songs.
- Query parameters: `title`, `artist`.
- Protected route (requires JWT token).

Authentication Flow

1. User signs up or logs in.
2. Backend generates JWT with user ID.
3. Token stored in `localStorage`.
4. Token attached to every protected API request via headers.
5. Backend verifies token with middleware.

Playlist Logic (Frontend only)

- `localStorage` key: `lyrix_playlists`
- Playlist structure:

```
interface Playlist {
  name: string;
  songs: Song[];
}
```

- Helper functions in `playlistStorage.ts`:
    - `getPlaylists()`
    - `savePlaylists()`
    - `addSongToPlaylist(playlistName: string, song: Song)`
    - `removeSongFromPlaylist(playlistName: string, songId: string)`

Future Enhancements

- 🎯 Move playlists to backend database for cross-device sync.
- 🎯 Add password hashing (bcrypt) for better security (if not already added).
- 🎯 Full text search and pagination on song lyrics.
- 🎯 Add favorites or liked songs feature.
- 🎯 Responsive design and improved UI/UX.

Setup Instructions

Backend

bash
CopyEdit

cd backend
npm install
npm run dev

Ensure MongoDB Atlas connection string in `.env`:

MONGO_URI=your_mongo_connection_string
JWT_SECRET=your_jwt_secret

Frontend

cd frontend
npm install
npm run dev

Author:

Developer: Peter Arcuri

Project: Lyrix