Spotify.js

- this file sets up the foundation for connecting to the Spotify API

- the 'scopes' array contains permissions for the user so you can access various tools for the app The '.join(' ')' method concats all these permission strings into one string where each value is seperated by a space. This is the format needed for permissions with Spotify.

- const params = {scope: scopes}
  const queryParamString = new URLSearchParams(params)

  converting the scope permissions into a web url.

  scope=user-read-email%20...

- the login url is whats needed to create the web address that a user is going to visit to login via Spotify. This url is a combination of Spotify's offical login page + the permissions I defined earlier

  → const LOGIN_URL = `https://accounts.spotify.com/authorize?${queryParamString.toString()}`

  When the link is clicked, the user will be informed on what the app wants to access

- const spotifyAPI = new SpotifyWebApi({
  clientId: process.env.CLIENT_ID,
  clientSecret: process.env.CLIENT_SECRET
  })

  Creating a way to actually connect/communicate with Spotify

  export spotifyAPI instance and LOGIN_URL for use in other files.

---

route.ts

- this is the file that handles the actual login process

tokens are temporary passes

accessTokens lets us access Spotify data, but access will expire

refreshTokens don't expire as fast and can make new accessTokens

imports:

- NextAuth : the library handling auth
- Spotify : tells NextJS we are using Spotify for login
- spotifyApi, {LOGIN_URL} : from spotify.js
- JWT: type definition for tokens