Lightweight iTunes Catalog

Description

Build a web application that allows users to search for songs, tv shows, movies, etc. (anything that iTunes API provides). This web app should allow for users to see their search split into different categories and allow them to favorite any item they'd like. Their list of favorites should persist on a page refresh, however no need to use a database. Two services should be created, a backend server and a client server. The client server should call the backend server to retrieve the data. Below are a list of requirements that should be completed.

1. Back-End API

Build an API that takes in a search term and uses that value to retrieve list of songs, movies, etc. from the iTunes Search API. Before returning the result, sort each of the results into categories based on media type. The response of this API should be a JSON object, where each field are the different media types and inside each field is an array of objects. The response should match the below structure.
Required fields: id, name, artwork, genre, url

```
{
	'song': [{
		id: Integer, // trackId (ID of
entity)
		name: String, // name of entity
		artwork: String, // URL of the
artwork
		genre: String, // Genre of entity
		url: String // trackViewUrl
	},
	...
	]
	,
	'feature-movie': [{
		id: Integer, // trackId (ID of
entity)
		name: String, // name of entity
		artwork: String, // URL of the
artwork
		genre: String, // Genre of entity
		url: String // trackViewUrl
	},
	...
	]
}
```

`List of different media types from documentation (link below): [book, album, coached-audio, feature-movie, interactive- booklet, music-video, pdf podcast, podcast-episode, software-package, song, tv-episode, artist]`

iTunes Search API Documentation:
https://affiliate.itunes.apple.com/resources/documentation/itunes-store-web-service-search-api/
Example API call to iTunes Search API:
https://itunes.apple.com/search?term=jack+johnson

2. Front-End

Build a view that allows a user to type in a search query and will display the results on the page **(do this by calling the back-end API you created above in #1)**. The results should be split into different sections based on the media type of entity. (ex. Songs will all be together sectioned off, feature-movies will be in another section, etc). If a certain media type section doesn't have any entries, do not show that section.
Required data to be shown on the view: picture of the artwork, name, genre, link to iTunes

3. Favorites

Allow items to be marked as "favorites" that will be saved and persisted per browser accessing your site. These favorites can be a mix of different media types of entities and should always be accessible on the page (even when no search has been entered)