Albmer
By
Os Álbmers

Jack Zhao

Kunaal Kumar

Michael Wright

Abstract

Albmer scrapes various music related sites for album reviews and ratings. This data is displayed to users for Billboard top music charts and for individual artists or albums available through custom searches. Additionally, there is an API that is well documented and allows for developers to integrate Albmer into their apps and services.

AWS VM URLs

Jack Zhao - AWS Educate reject my request, I mentioned that to Karthik Karanth

Kunaal Kumar - ec2-54-162-77-42.compute-1.amazonaws.com

Michael Wright - ec2-54-92-140-225.compute-1.amazonaws.com

Introduction

Albmer provides a way for users to find new music and search for ratings for specific albums they are interested in. The site consists of a home page, search result page, artist page and album page. Starting at the home page, users see a search form where they can search for artists or albums. Additionally, when the home page loads, the Billboard Top Albums Chart is scraped by the application and displayed in a table that is generated dynamically row by row. The album title in each row is a link that routes the user to the search results page. When the user searches for an artist or album they are also redirected to the search results page.

The search result page receives the type of search (“Artist” or “Album”) as well as the search query. The page then hits the MusicBrainz API which is an external API that provides details about musicians, albums, and songs. MusicBrainz then returns a list of artists or albums with unique MusicBrainzIds that can be used for getting more details. When the user clicks on an artist or album link they are then redirected to the artist page or album page respectively.

The album and artist pages receive the unique MusicBrainzId, and make a second API call to MusicBrainz to get details about the album or artist. Included in the details are links

to several sites that contain music ratings. These links are used to scrape ratings from the corresponding sites. By using the MusicBrainz links we can reduce scraping time because we can use light-weight html parsers without needing a browser. The artist and album pages show details and ratings as well as links to associated albums. The album page also shows images of the album cover.

The album and artist pages shows ratings from several sites and saves the user time by providing details and ratings all in one place.

Feature Table

Feature Name	Scope	Primary Programmer	Time Spent	File/Function	LoC
MusicBrainz API callbacks	Back-end	Kunaal	10.5 hours	API Controller to fetch, parse and provide data from MusicBrainz	626
Database Caching	Back-end	Kunaal	12 hours	Cache calls from API endpoints to lessen network load, avoid rate limiting and speed up processing time	212
API documentation	Documentation	Kunaal	1 hour	Documentation for each API endpoint containing description, result, parameters and examples.	-
Overview Page / Landing Page	Front-end	Kunaal	1 hour	Pages detailing purpose of Albmer,, features and steps to use it.	40
Created and Seeded the HighScore Database Table	Back-end	Kunaal	1.5 hrs	Data/db_initializer.cs	50

Billboard Top Album Scraper	Back-end	Mike	3 hrs	ScraperController/ScrapeAlbumChart()	82
Billboard Top Album UI and Routing	Front-end	Mike	6 hrs	Home/Index.cshtml, billboard.js/All site.js/All	161
Search Result Page and Routing	Front-end	Mike	10 hrs	search.js/All Home/SearchResults.cshtml	44
All Music Scraper	Front-end	Mike	5 hrs	ScraperController/AllMusicScraper.cs	57
Routing Between Search, Billboard, and Artist/Album	Front-end	Mike	4 hr	search.js/All	8
Album Details Page	Front-end	Jack	8 hr	details.js/All Album.cshtml	150
Artist Details Page	Front-end	Jack	7 hr	details.js/All Artist.cshtml	150
UI Enhancements	Front-end	Jack	4 hr	distributed js and cshtml files	80
Rate Your Music Scraper	Back-end	Jack	3 hr	ScraperController/RateYourMusicRatings()	40
Discogs Scraper	Back-end	Jack	3.5 hr	ScraperController/DiscogsRatings()	45

Individual Contribution

Kunaal was responsible for writing the API controller endpoints which serialized data from MusicBrainz API for music metadata and CoverArtArchive for album art. He was also responsible for caching results in the database for faster load times and to potentially avoid rate limiting by MusicBrainz. Due to his work with the database his commits included migrations which makes his “lines of code committed” higher than everyone else's.

Mike was responsible for the Billboard chart scraper and front-end, the AllMusic Scraper, the Search Results Page, and overall testing of the API endpoints. A large portion of his

time was in testing and integrating everyone's features. This resulted in fewer lines of code, but was important to the app. He created core features as well.

Jack was responsible for the Details Page for Artist and Album. As well as the DiscogsRatings and the RateYourMusicRatings scrapers. He also improved the UI for the site and make the interaction nicer.

Individual Contribution

Team Member	Time Spent on Proj	Lines of Code Committed
Jack	25.5 hours	465
Kunaal	24.5 hours	42,577
Michael	28 hours	352

Summary

The Albmer team went the extra mile and completed a project with scrapers that can access album information on 4 different sites and hit multiple external API's. The user experience is intuitive and the application provides a useful service. Our team performed at a Superior level. The product we've created delivers value to anyone that has looked for new artists and albums.