

SongSurf

<https://github.com/comp195/SongSurf>

Shahbaj Sohal - s_sohal2@u.pacific.edu

William Balbuena - w_balbuena@u.pacific.edu

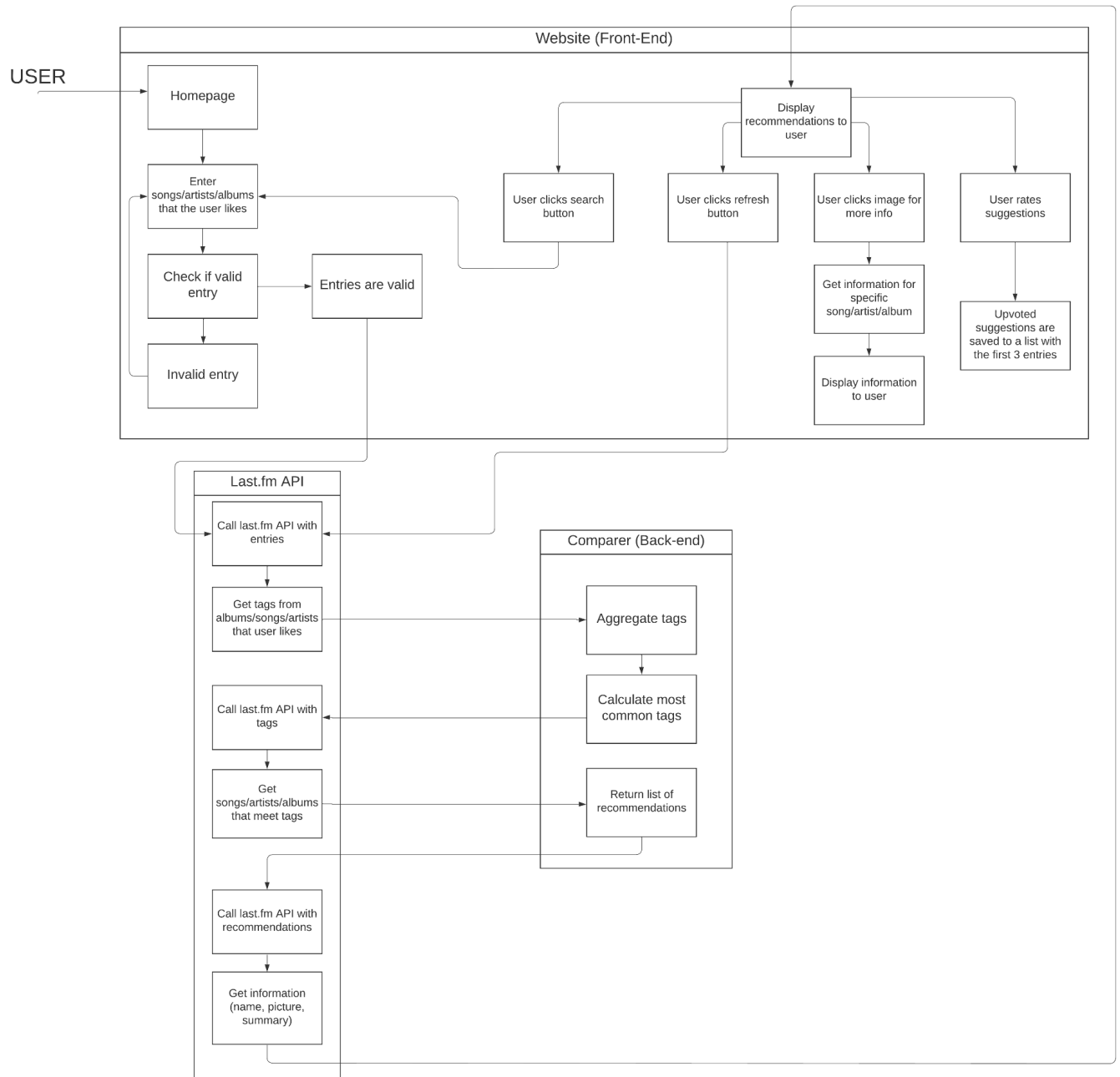
Patrick Nisperos - p_nisperos@u.pacific.edu

Feb 5, 2023

Introduction

SongSurf is a website/tool that music enthusiasts or someone looking for new music to listen to can utilize to find new artists, songs, or albums to listen to. As a group we have decided to create a website with a sleek and user-friendly user interface. Using API calls to a music database we will create recommendations based on what the user inputs into the website. The user will be able to input artists, albums, or songs that they enjoy into a text field and also select whether they want their recommendation to be an artist, album, or song and in return get a few recommendations based on their inputs. The user will also be able to select whether they like or dislike the given recommendation and continue to get more recommendations. Our goal as a team is to get an algorithm working to give the best possible recommendations and have an easy-to-use and a fantastic website design. We will be working together over the course of this semester to accomplish this goal using what we have learned from our previous classes.

System Architecture



Hardware, Software and System Requirements

Hardware Requirements

- ❖ **Display** - Any inch display, must have minimum resolution of 852x480 (480p)
- ❖ **Disk Storage** - N/A
- ❖ **Communication Network** - N/A
- ❖ **Input Devices** -
 - Mouse & Keyboard
- ❖ **Minimum network bandwidth** - Minimum 5MB/s

Software Requirements

- ❖ **Language** - Python, Javascript
- ❖ **Framework** - Flask
- ❖ **Libraries** - Python and Javascript standard libraries
- ❖ Last.fm API

System Requirements

- ❖ **Operating System** - Any OS since our project is a website

External Interfaces

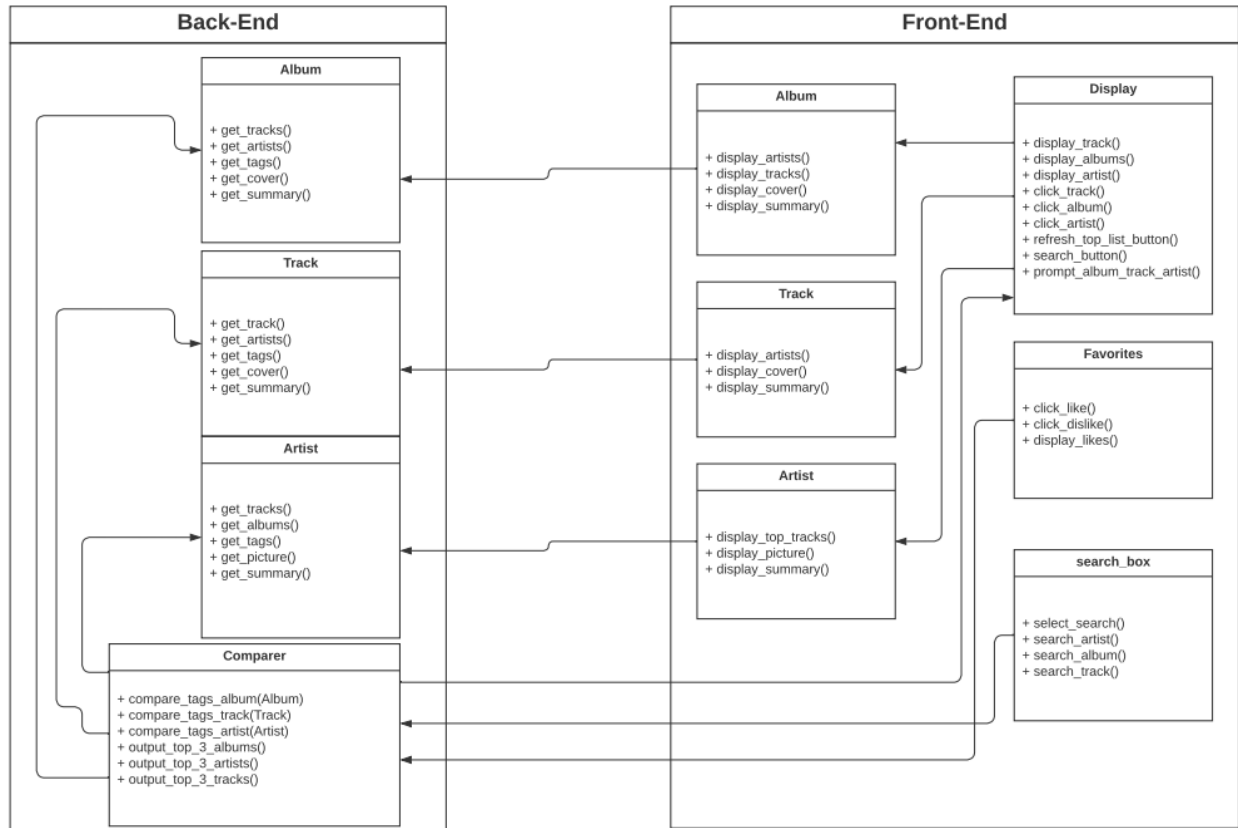
The Last.fm API is well documented and can be accessed through this link:

<https://www.last.fm/api#getting-started>. To utilize the Last.fm API, an API account is required.

When making calls to the API an identifiable User-Agent header and a set of parameters that include an API key are required for each request.

Software Design

Class Diagram and Specifications



Front End

❖ Display

The *Display* class is the most important class for the front-end, responsible for displaying all front-end interactable components to the end user. In addition to the foundational album/track/artist information, this class will also incorporate “refresh” functionality that will save the likes/dislikes of previous set of album/track/artist and account for them in the new set of album/track/artist it will produce. This class will communicate with the *album/artist/track* classes.

❖ Favorites

The *favorites* class is responsible for likes/dislikes functionality and interface. It will also be responsible for displaying all the user's likes. This class will communicate with the *comparer* class.

❖ **Search_box**

The *search_box* class is responsible for the essential searching functionality. Allowing the user to search a preferred artist/album/track. This class will communicate with the *comparer* class.

❖ **Album**

The *Album* class has the ability to display the artists, all the tracks in the album, the album summary, and the album cover picture. This class will communicate with the back-end *Album* class to fetch the raw data. It will also communicate with the *Display* class.

❖ **Track**

The *Track* class has the ability to display the artists, the album cover picture, and the track summary. This class will communicate with the back-end *Track* class to fetch the raw data. It will also communicate with the *Display* class.

❖ **Artist**

The *Artist* class has the ability to display the top tracks, the picture of the artist, and the artist summary/bio. This class will communicate with the back-end *Artist* class to fetch the raw data. It will also communicate with the *Display* class.

Back End

❖ **Comparer**

The *comparer* class is the most important class for the back-end responsible for the algorithms to most accurately discover the top albums/tracks/artists for the end user. This class will communicate with all the back-end classes to fetch tags, to *search_box* to get items searched for, and to *favorites* class to fetch likes/dislikes.

❖ **Album**

The *Album* class is responsible for fetching all data regarding tracks, artists, tags, and summary associated with the album. This fetching will utilize the last.fm API. This class will communicate with the *comparer*, and the front-end *Album* class.

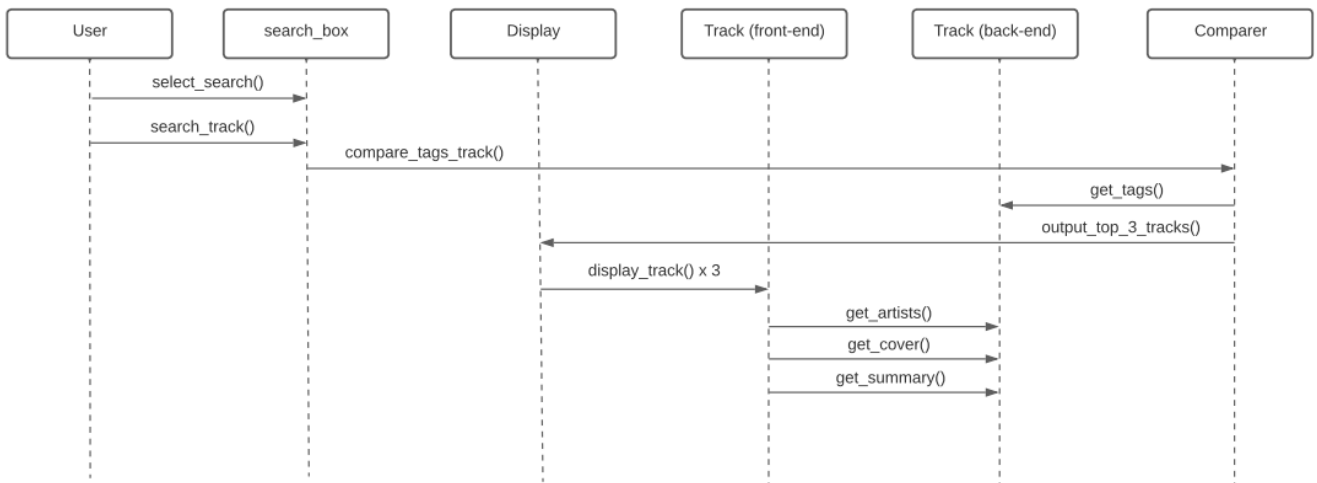
❖ **Track**

The *Track* class is responsible for fetching all data regarding album, artists, tags, and summary associated with the track. This fetching will utilize the last.fm API. This class will communicate with the *comparer*, and the front-end *Track* class.

❖ **Artist**

The *Artist* class is responsible for fetching all data regarding tracks, albums, tags, and summary/bio associated with the artist. This fetching will utilize the last.fm API. This class will communicate with the *comparer*, and the front-end *Artist* class.

Interactive Diagrams



Design Considerations

- ❖ Simple, straightforward functionality that leaves room for more features
- ❖ Making sure the user is inputting correct information (such as real artists or songs)
- ❖ Making an efficient website that is not super laggy
- ❖ Maintain a clean codebase that everyone can work with
- ❖ Did not want to create god class and split everything up into different classes
- ❖ Being specific with function definitions
- ❖ Wanted user to be able to like and dislike recommendations
- ❖ Also be able to see the recommendations that they liked

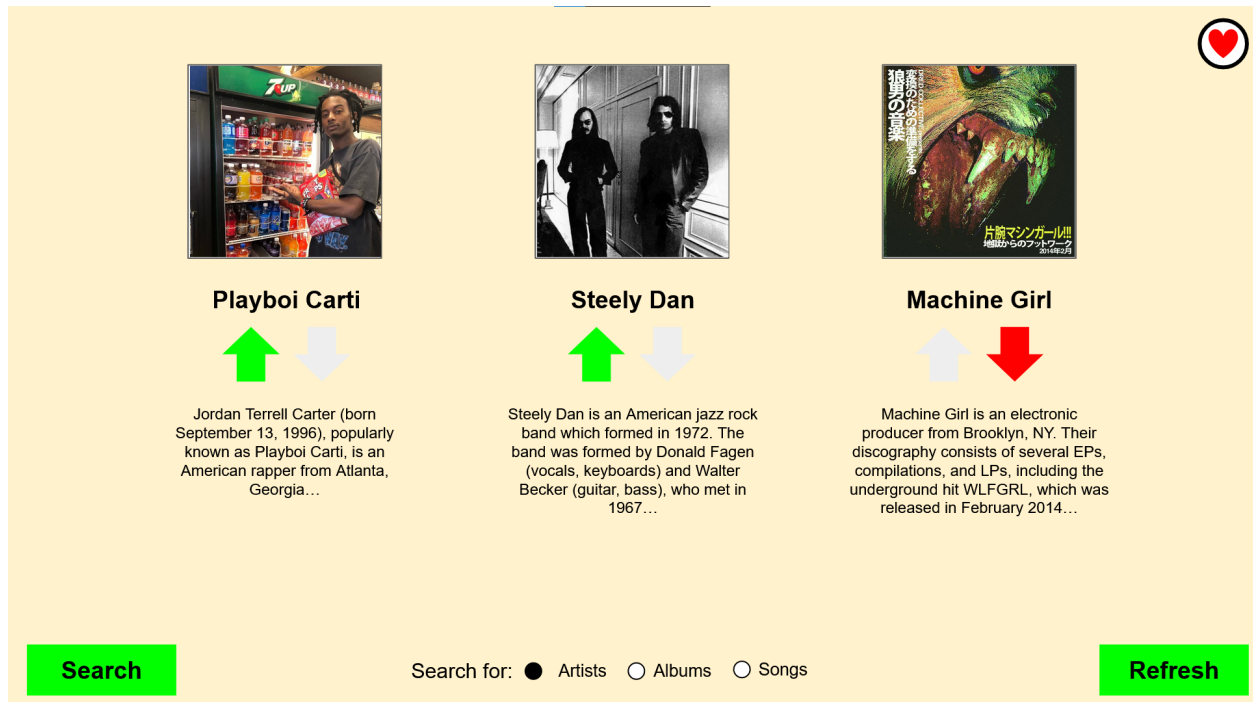
User Interface Design

The mockup shows a light yellow background with three input sections. Each section has a label 'Enter a song/artist/album:' followed by a text input field. The first field contains 'The Beach Boys'. The second field contains 'Lil Yachty - drive ME crazy!'. The third field contains 'Barack Obama' and is preceded by the text 'Invalid entry!' in red. Below these fields is a 'Search for:' section with three radio buttons: 'Artists' (selected), 'Albums', and 'Songs'. To the right of the radio buttons is a bright green rectangular button with the word 'Surf' in black text.

Mock up of entry page

When the user first opens the website, they will be prompted to enter up to three songs, albums, or artists that they enjoy. We will check whether or not their entries are valid and will notify them if such. They will also be asked to choose between either artists, songs, or albums to search for.

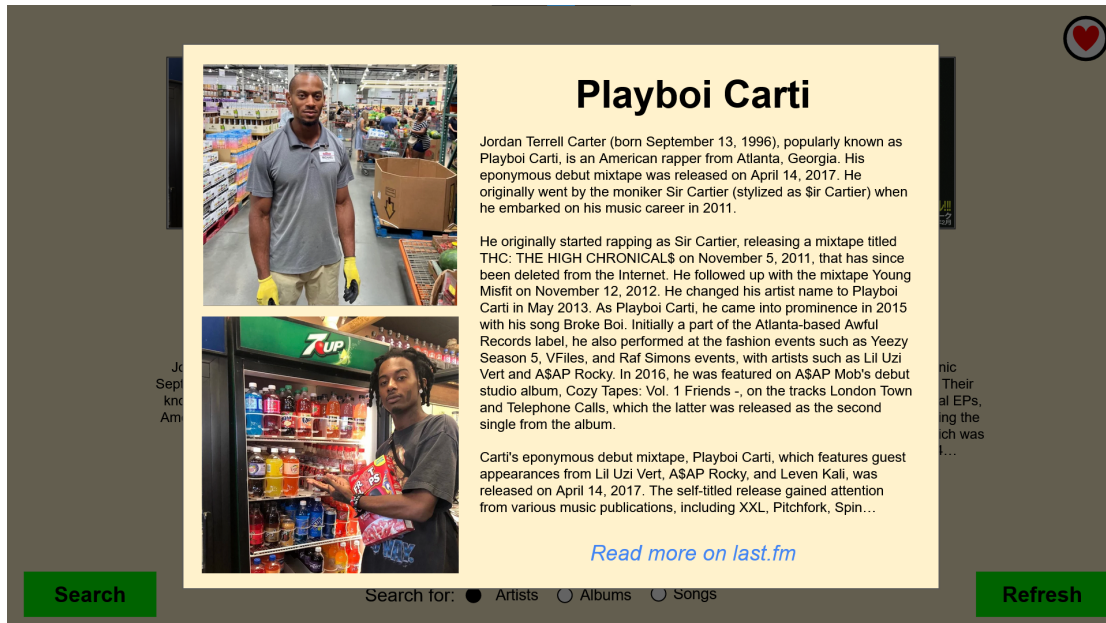
Once they have entered valid artists and have selected a category to search for, they will proceed to the next screen by clicking the “Surf” button.



Mock up of recommendation page for artists

After entering valid entries the user is then presented with three recommendations of songs/artists/albums depending on their choice. They are then able to vote on the recommendations presented to them and can refresh their selections. The next selections are impacted by their initial three entries as well as their reactions to prior recommendations. The user is also able to choose what category their new selections will provide.

The user is also able to restart their search by clicking the “Search” button which will bring them back to the initial screen where they’ll be able to start their search over from scratch.



Playboi Carti

Jordan Terrell Carter (born September 13, 1996), popularly known as Playboi Carti, is an American rapper from Atlanta, Georgia. His eponymous debut mixtape was released on April 14, 2017. He originally went by the moniker Sir Cartier (stylized as \$ir Cartier) when he embarked on his music career in 2011.

He originally started rapping as Sir Cartier, releasing a mixtape titled *THC: THE HIGH CHRONICAL\$* on November 5, 2011, that has since been deleted from the Internet. He followed up with the mixtape *Young Misfit* on November 12, 2012. He changed his artist name to Playboi Carti in May 2013. As Playboi Carti, he came into prominence in 2015 with his song *Broke Boi*. Initially a part of the Atlanta-based Awful Records label, he also performed at the fashion events such as Yeezy Season 5, VFiles, and Raf Simons events, with artists such as Lil Uzi Vert and ASAP Rocky. In 2016, he was featured on ASAP Mob's debut studio album, *Cozy Tapes: Vol. 1 Friends -*, on the tracks *London Town* and *Telephone Calls*, which the latter was released as the second single from the album.

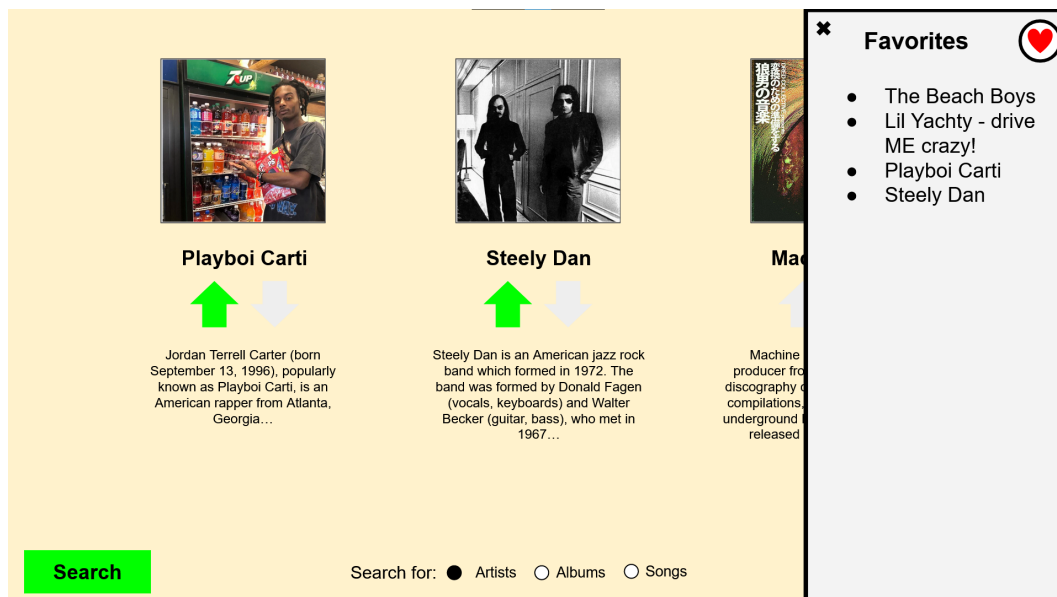
Carti's eponymous debut mixtape, *Playboi Carti*, which features guest appearances from Lil Uzi Vert, ASAP Rocky, and Leven Kali, was released on April 14, 2017. The self-titled release gained attention from various music publications, including *XXL*, *Pitchfork*, *Spin*...

[Read more on last.fm](#)

Search for: ☒ Artists ☐ Albums ☐ Songs

Mock up of summary page

On the selection screen the user is able to click the images of the songs/artists/albums to get more information. The summary also contains a link to its respective last.fm page where the user is able to find out more about the song/artist/album.



Playboi Carti

Jordan Terrell Carter (born September 13, 1996), popularly known as Playboi Carti, is an American rapper from Atlanta, Georgia...

Steely Dan

Steely Dan is an American jazz rock band which formed in 1972. The band was formed by Donald Fagen (vocals, keyboards) and Walter Becker (guitar, bass), who met in 1967...

Machine

Machine producer from discography of compilations, underground released

* Favorites

- The Beach Boys
- Lil Yachty - drive ME crazy!
- Playboi Carti
- Steely Dan

Search for: ☒ Artists ☐ Albums ☐ Songs

Mock up of recommendations page with favorites sidebar

The user is also able to click the favorites icon on the top right to keep track of what they've liked so far.

Glossary of Terms

algorithm - a set of instructions, used to solve problems or perform tasks, based on the understanding of available alternatives

application programming interface (API) - a way for two or more computer programs to communicate with each other

tag - a label attached to someone or something for the purpose of identification or to give other information

user interface (UI) - the means by which the user and a computer system interact, in particular the use of input devices and software

References

Grupman, Celeste. “Tutorial: Getting Music Data with the LAST.FM API Using Python.”

Dataquest, 7 Oct. 2019, www.dataquest.io/blog/last-fm-api-python/.

“Last.fm Music Discovery API.” *Last.fm*, Last.fm, www.last.fm/api#getting-started.

“Welcome to Flask.” *Welcome to Flask - Flask Documentation (2.2.x)*, Flask,

flask.palletsprojects.com/en/2.2.x/.