MusicIndex documentation

# 1.- Header

Header component is present outside router-view, It will be visible in all the App, it contains the name of the application, which on click will redirect to the home page.

# 2.- Footer

Footer is a component outside router-view, It will be visible in all the App, It contains contact information retrieved using the function from ‘AboutUtils.js’ script, which only reads data from the about element in the ‘config.json’ config file. You can find both of these files in the Utils and config folder.

# 3.- SearchView

View Component related to path ‘/’, It splits the grid in 2 rows, one for the search bar, and other por the results section, this view implements the searchSong method, this method retrieves the result of the search and will display an alert message if no results have been found. This method works by using the searchSong method from the ‘ITunesAPIUtil.js’ script.

The method in the ITunesAPI util script performs a fetch to the Itunes store URL using the text receive from the user as the term parameter, the type ‘song’ as the entity parameter, It doesn’t use the limit parameter as by default ITunes will send a limit of 50 results unless we specify otherwise.

The result from the searchSong method is stored in the results data, this information works with the songs list, a sub component of the second row of the UI.

# 4.- SearchBar

An input component followed by a search button, when the search button is clicked the input will be cleared and an emit will be emitted to trigger the search in searchView.

# 5.- SongsList

This is the element which displays the result of the search, it contains 2 rows, one for the result and other for the pagination control.

Each page contains a top of 3 songs, the songs to display are chosen with the help of the page data, and the computed property renderedSongs, this property will return the proper songs depending on the pagination number.

A watcher controls that if another search is triggered the page will be set to 0.

In the last row we have the previous and next page buttons, this buttons depend of 2 computed properties to appear:

* isThereNextPage : returns true if it is not the las page
* isTherePrevPage: returns true if the page is not 0

Using the method get Price and getYear, some data is filtered and merged to match the requirement of the component SongElement, the one responsible to display each song.

# 5.- SongElement

Element responsible to render songs information, it renders an image related to the song (artWorkURL100 received from ITunes), the song name, artist, album, year of release and price. Price is displayed in a button that will open a new tab with the Itunes page related to the song.

# 6.- Testing

Unit testing has been used to assure the proper functionalities of the components, the only component without unit tests is SongElement component.

To run the tests use command npm run test:unit

There are 9 unit tests in the application which 9 are successful on execution.