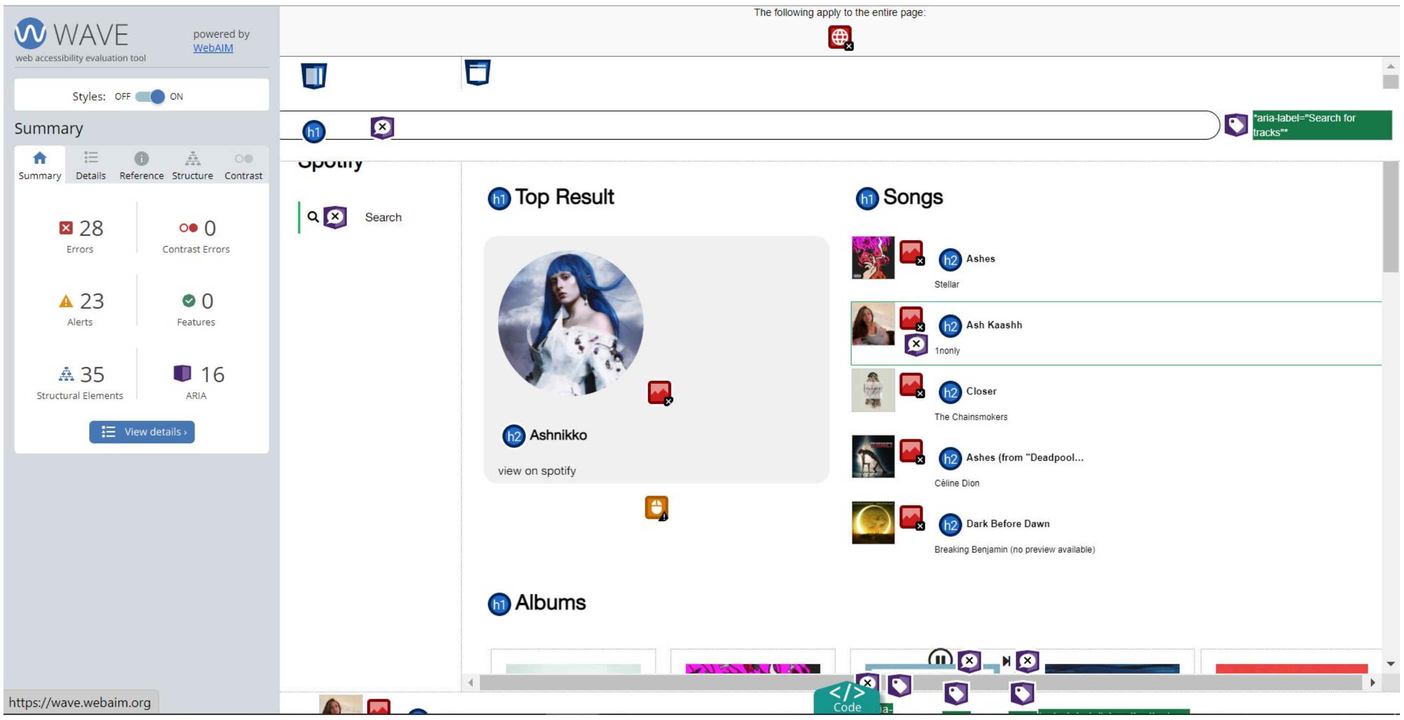
COMP_SCI130 Homework 4:

Name: Susan Jiayang Liao,   UID: 3405013

1. WAVE accessibility report (*search for "ash"*):

Before corrections:


WAVE web accessibility evaluation tool, powered by WebAIM

Styles: OFF ON

Summary

28 Errors | 0 Contrast Errors
23 Alerts | 0 Features
35 Structural Elements | 16 ARIA

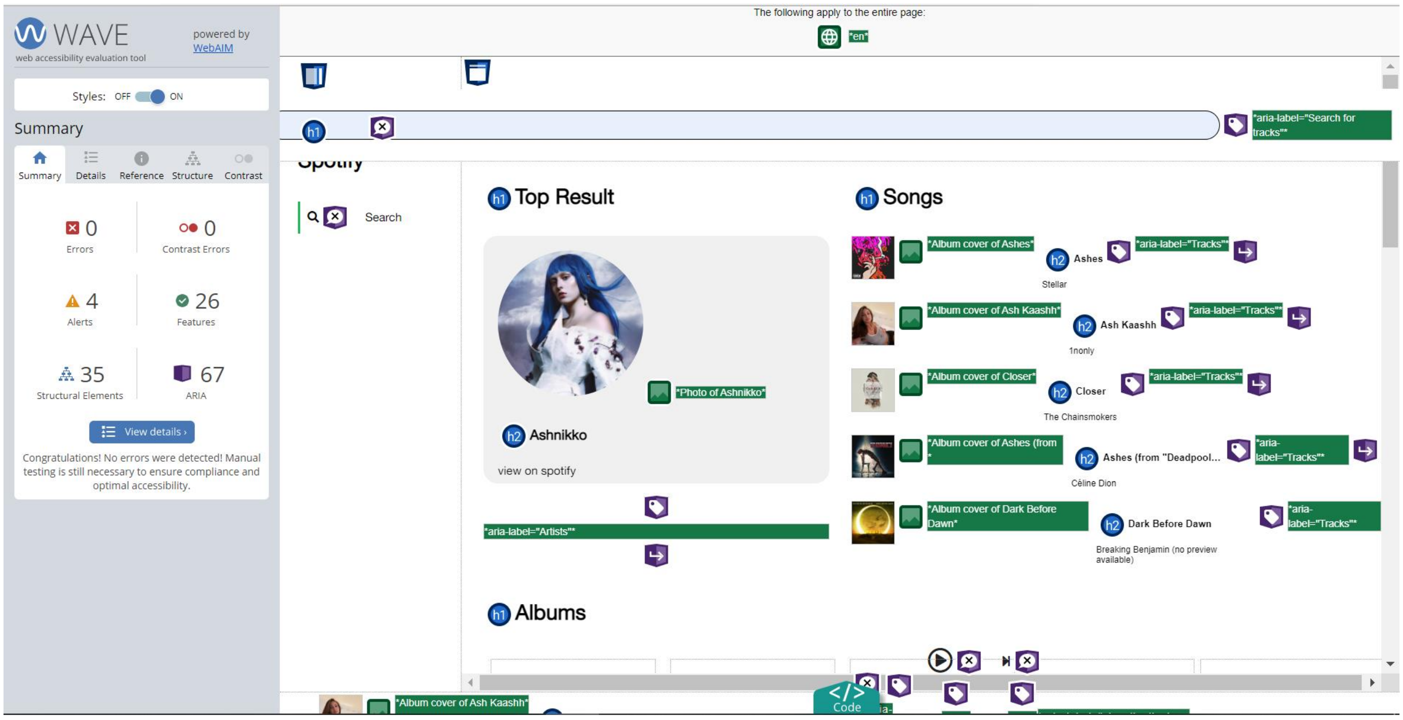
View details ›

https://wave.webaim.org

The following apply to the entire page:

"aria-label="Search for tracks""

After corrections:


WAVE web accessibility evaluation tool, powered by WebAIM

Styles: OFF ON

Summary

0 Errors | 0 Contrast Errors
4 Alerts | 26 Features
35 Structural Elements | 67 ARIA

View details ›

Congratulations! No errors were detected! Manual testing is still necessary to ensure compliance and optimal accessibility.

The following apply to the entire page:

"en"

"aria-label="Search for tracks""

"Photo of Ashnikko"

"aria-label="Artists""

"Album cover of Ashes" "aria-label="Tracks""

"Album cover of Ash Kaashh" "aria-label="Tracks""

"Album cover of Closer" "aria-label="Tracks""

"Album cover of Ashes (from "aria-label="Tracks""

"Album cover of Dark Before Dawn" "aria-label="Tracks""

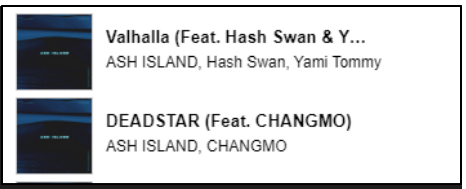
"Album cover of Ash Kaashh"

- What corrections did you have to make, and how did you make them?
  1. HTML file ("index.html")
     a. Add a language tag to specify the language of the entire page:
        Example: `<html> → <html lang="en">`
  2. JavaScript file ("script.js")
     a. Add alternative information for each image in the page, in case that the image is unable to display.
        Add <img src="..." **alt="Album cover of ${data.album.name}"**> in all HTML generators in script.js, under functions "getTracksHTML()","getArtistsHTML()", and "getAlbumsHTML()".
        Example: `<img src="…"  alt="Album cover of ${data.name}">`

     b. Use **tabindex="0"** for "onclick event handler" to make the keyboard navigable, so that users can now just use keyboard keys to do some functions, i.e., play and pause the music.
        Add <section class="..." **tabindex="0"**> in all HTML generators in script.js, under functions "getTracksHTML()","getArtistsHTML()", and "getAlbumsHTML()".
        Example: `<section class="…" id="…" onclick="…;" tabindex="0">`

     c. Add aria-labels for every "clickables" in the page, i.e., buttons, artist-cards, tracks, albums...
        Add <... **aria-label="(Corresponding contents)"**> in all HTML generators in script.js, under functions "getTracksHTML()","getArtistsHTML()", and "getAlbumsHTML()".
        Example:
        `<section class="…" id="…" onclick="…;" tabindex="0" aria-label="Artists">`
- Is this interface functional without the use of a mouse (i.e., just using keyboard keys)? Why or why not?
  - The interface is functional without the use of a mouse. You can use "Tab + Enter" to access all clickables in the page.
  - You can use the keyboard to type in the search box and press "Enter" to start a search.
  - Disadvantage: It's sometimes tedious to use "Tab" to go through all the clickables in the page if you just missed the wanted one. However, the "" and "" buttons here is used to scroll up and down the page. So, the most feasible option here is to use "Tab" to transverse all the clickable until find the desired one and press "Enter" there.
  -
- What are other accessibility tests or features you added or would be interested in learning about?
  1. Features
     a. I replace the attribute ${data.name} with a nameList generated by a for loop, so that we can display the names of all artists of a track.

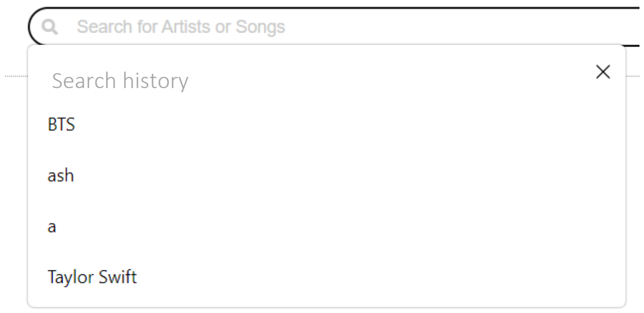
```
for (const track of Object.keys(topTracks)){
   nameList="";
   for (const artist of topTracks[track].artists){
       nameList+=artist.name;
       nameList+=", ";
   }
   nameList=nameList.slice(0,-2);
   elem.innerHTML += getAlbumTracksHTML(topTracks[track],nameList, albumImage);}
```


Valhalla (Feat. Hash Swan & Y...
ASH ISLAND, Hash Swan, Yami Tommy

DEADSTAR (Feat. CHANGMO)
ASH ISLAND, CHANGMO

     b. There is no image attributes when getting tracks from an album. Therefore, I display the image of album for each track if their own image is unavailable.
        - See function **const getAlbumTracksHTML = (data,nameList,image_url)** in script.js

2. Other accessibility tests.
    a. The page will store the search history, increasing the convenience.
    Question: Where is the search history stored, in the cookies or local cache memory?



BTS

ash

a

Taylor Swift

3. Other features
    a. I compared the documentation of Spotify API and the api tutor we used in extra credits. I found that for Spotify API, users are required to obtain an account and corresponding keys to authorize API calls. Hope we can explore some other real-life APIs in the future!
    b. I finished all the extra credit questions except the last one, due to some confusion and difficulties in using YouTube API. Looking forward to seeing some examples in tutorial!