Front End Developer Exam

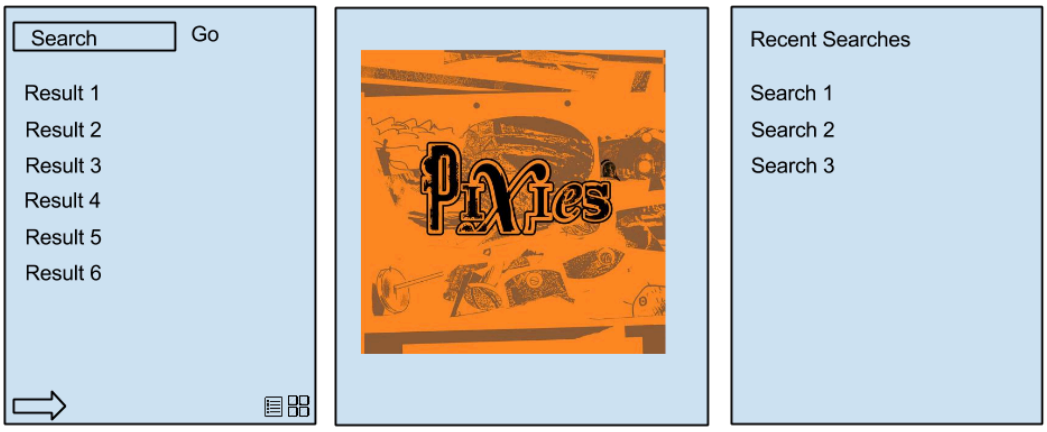
Introduction
This exam will use the SoundCloud API to fetch data and display it on a page. SoundCloud is a platform which gives users access to a library of tracks. These tracks can be embedded and played on other web pages. The SoundCloud API allows you to query the tracks.

Apart from the SoundCloud API, you are allowed to make use of any library you see fit to help you complete the task. Assume you are developing this for the latest Chrome release, so no need to support old browsers.

The SoundCloud documentation can be found here:
https://developers.soundcloud.com/docs/api/guide

When asked for a client_id please use: **ggX0UomnLs0VmW7qZnCzw**

The Layout



Create an HTML page with the following items:
1. Search container which contains
    1. A text box
    2. A button or link to trigger the search
    3. A container for the search results
    4. 3 buttons at the bottom
        i. Next button
        ii. List button
        iii. Tile button
2. Image container
3. Recent Searches Container

The Task
1. Use the SoundCloud API to allow the user to search for anything entered in the search box. The names of the results should be displayed as a list below the textbox. Only fetch 6 results at a time.
2. When clicked, the next button should display the next 6 results instead of the current results

3. Keep a history of the searches for the user on the Recent Searches Container. This history should display the last 5 searches and should be available for the user in subsequent visits.
4. When clicked, a search item should instigate a new search.
5. When a search result is clicked, it should fly to the Image Container, fade out and the image for the search result should fade in instead.
6. When clicking the central image, embed the track below the image and make it play
7. BONUS: When clicking the tile button tile the search results should tile in rows of using their images. Clicking the list button should toggle back to list view. This selection should be remembered for the user's next visit.

What we're looking for

- Well written code broken into logical components that interact with each other.
- Framework implementation (React.js, Backbone, Angular, etc) or "vanilla" JavaScript implementation a plus.
- Avoid a 1 page jQuery solution.