Introduction

Our engineer will be expected to:

- Make technical choices regarding the frontend architecture of our website including the framework to serve assets, html, images, javascript, and css to the client browser.
- Make our site beautiful, performant, and intuitive on a variety of devices and browsers.
- Work with our design and product team to create a great user experience with pixel-perfect designs and detailed transitions

This exercise is a way to demonstrate frontend engineering skills on a small project that should show your comfort level with modern JavaScript and web frameworks, and in shipping frontend architecture while taking into account customer experience and best practices for web user experience.

Exercise

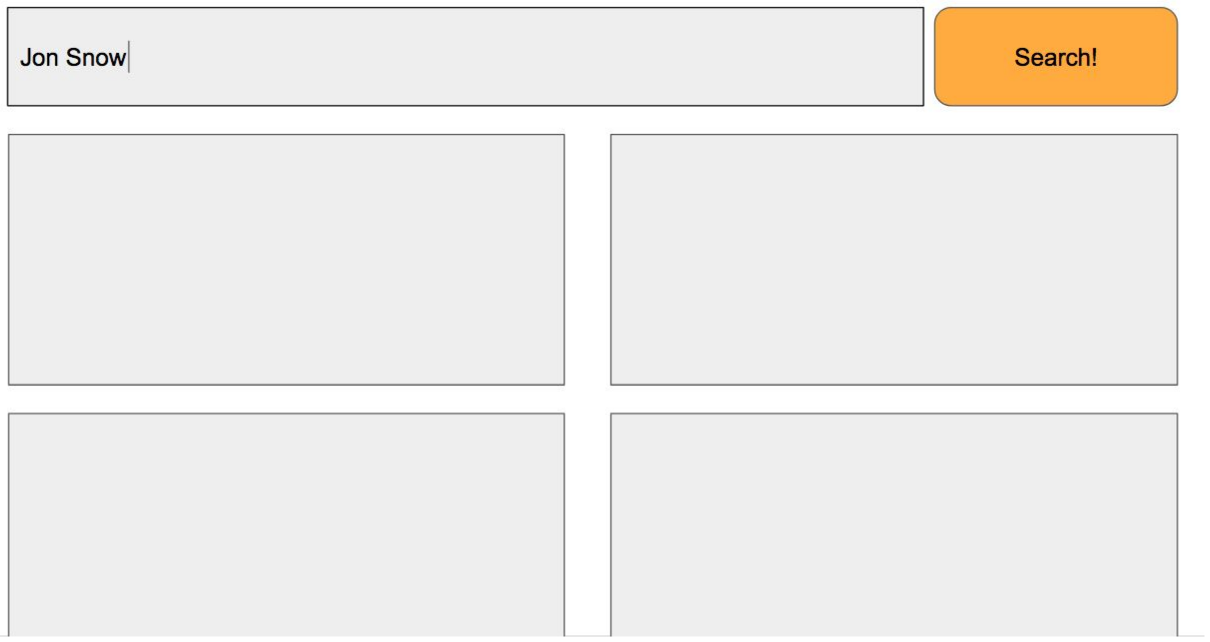
Create a web application to search for animated GIFs using the giphy api. Giphy is a repository of animated GIFs. You can use their public api key "`dc6zaTOxFJmzC`" for this exercise. You will need to familiarize yourself with their various endpoints based on the documentation on their github README.

Deliverables

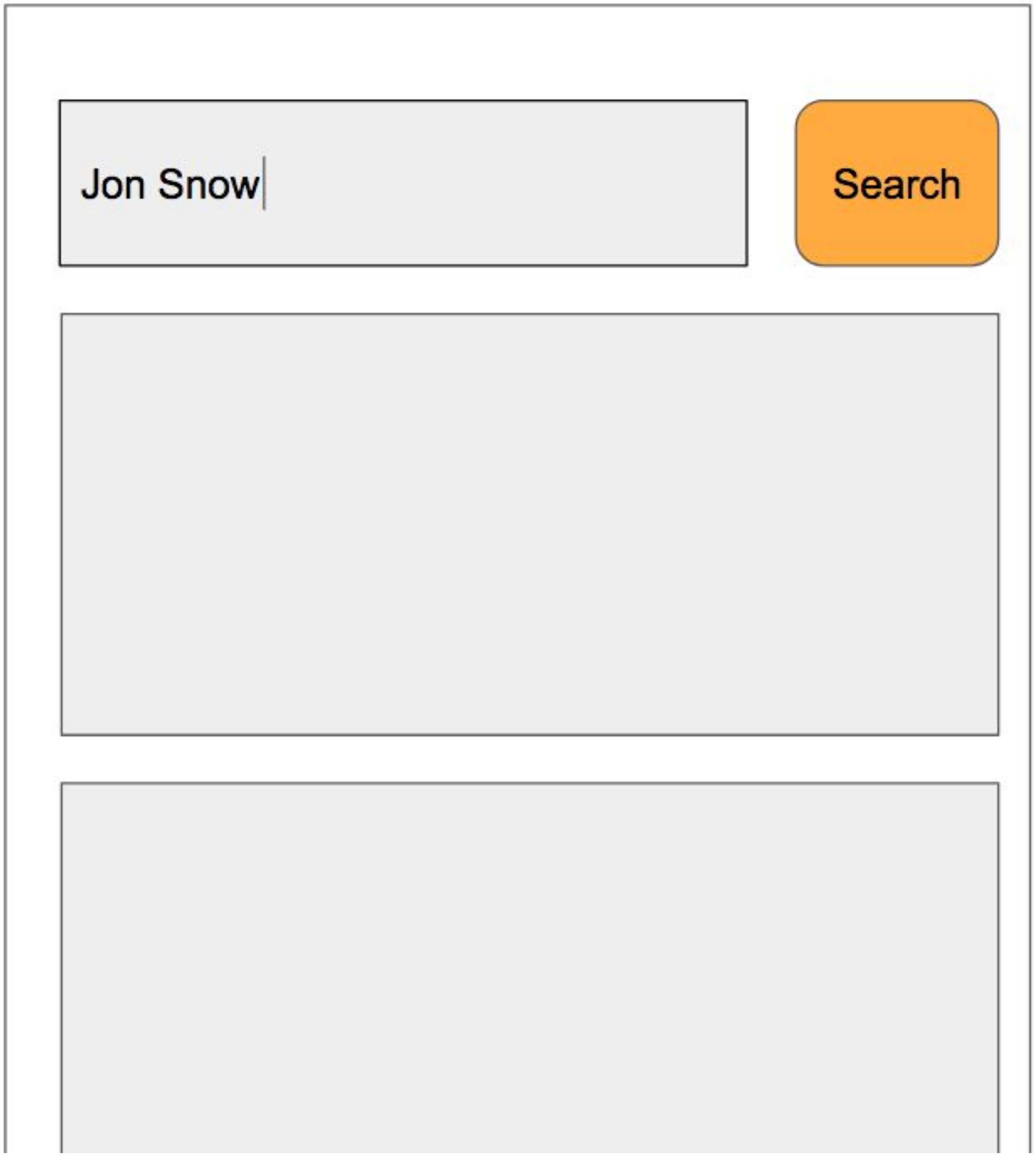
- Use any framework you like to generate and implement this web application- we want to see how you like to work!
- Provide clear instructions on how to install all dependencies to build and run this application. Assume that the user who will be running your application has nothing installed other than the base OSX installation.
- Create a readme in the provided format below with instructions on how to build, run, and test your application. Also update the changelog with the steps you took to generate and build any parts of the application.
- The deliverable format will be a .zip file containing the root directory structure for your application. To run the application, I will unzip your .zip file into a new folder on a mac machine, cd to that folder, look for a README or README.md file in that folder, and follow the README's instructions on installing dependencies, building, and running the web application server. If you cannot attach a zip file to the email, you can send me a link to the zip file on dropbox, box, or any hosting provider of your choice.
- You should NOT bundle dependency sources in your deliverable. For example, if you are using node modules as a dependency, the deliverable zip file should include the package.json with your node dependencies and versions, but not the contents of your node_modules directory.

- Your web application will be expected to look great when visited by a browser on a mobile phone at 320 px and on a desktop at 1280px. All sizes in between are nice to have, but not necessary for this deliverable.
- You may choose to include additional screens and routes to your application as you see fit to complement the functionality of this application.

Routes and Behavior

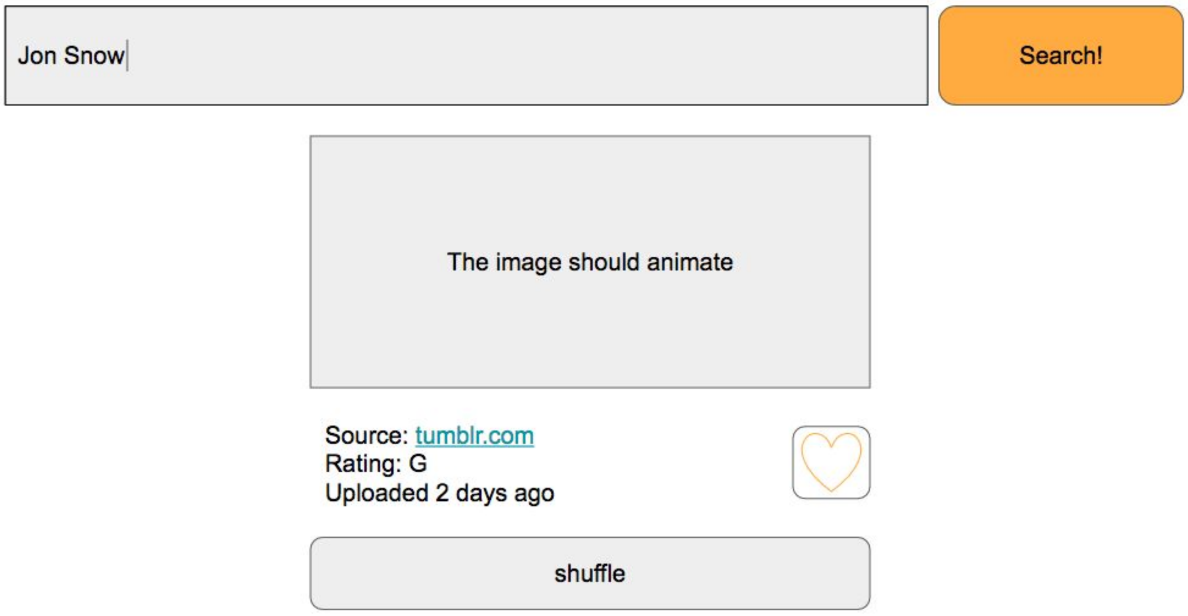
- / - the blank search screen
  - a search box and a submit button which will send you to /<search term>.
- /<search_term> (for example “/kittens”)
  - The user is displayed results for that search term.
  - On desktop the image should display in a 2 column grid.
  - 

  - On mobile, the images should just display in one column



  - If a user clicks any of the images, they will be taken to a full page view of the gif id (suggested route /<search_term>/<gif id>
- /<search_term>/<giphy gif id>
  - The user is shown a large display of the animated gif, which will be looping constantly. Below the gif, they will be shown metadata about the image.

    - A button below that will randomly navigate them to another giphy gif id that matches this search term in the format /<search term>/<giphy gif id> from the top 20 search results for this term.
    - A user should be able to mark/unmark an image as a favorite. Favorites should be stored locally on the user's browser using the storage mechanism of your choosing. There's no need to persist this data in a backend store, or to have user accounts or login mechanisms.



    -
- /favorites
    - A user should be able to see their favorites - the layout and behavior of favorite images should be similar to the search results screen on both desktop and mobile.
    - This should be stored locally on the client browser.

README format

Your readme should contain the following sections:

- Dependencies (Include your instructions to install all dependencies required to build, test, and run this project on a freshly installed Mac running OSX here)
- Build Instructions
- Run Instructions
- Test Instructions
- Changelog (The steps that you took to create this project, particularly if you used a third party scaffolding tool, or code generation tool)