UDACITY

Return to Classroom

Landing Page

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Dear Learner,
Building something dynamic is definitely a skill you will find useful throughout your Frontend development journey. Using Javascript in just skeleton code is like giving life to Pinocchio. Great job!!

There are some extra links for the other requirements, attached in the hope those will help you in learning more about the concepts.

Some extra optional resources

- https://code.tutsplus.com/tutorials/24-javascript-best-practices-for-beginners--net-5399
- https://blog.adtile.me/2014/01/16/a-dive-into-plain-javascript/

Congratulations on completing the project! Great work on this submission! I look forward to seeing your skills continue to grow in front-end development.

Keep Learning 😄

Interface and Architecture

✓

The project should have a structure like the one shown below. All files shown must be present and the app must successfully render a home page with clear design and functionality added when index.html is

loaded in the browser. No errors should display in console.

```
css
- styles.css
index.html
js
- app.js
README.md
```

✅ **Good job with the directory structure**

This is a good practice, so that in the future when the project starts to grow, we know what to find where? Also, as the number of pages or use cases increases, there can be sub-folders inside those folders like, if there are multiple pages, we can have a root style, a common style, and then page-specific style and the same goes for JS.

Read more here about how you can design better folder structures
https://medium.com/@nmayurashok/file-and-folder-structure-for-web-development-8c5c83810a5

✓

- All features are usable across modern desktop, tablet, and phone browsers.
- A Responsive layout of the landing page should be created to use across all devices.
- Make sure that the navigation bar is responsive too across all these devices.
- Responsiveness can be verified by inspecting the landing page using the Developer Tools option on Google Chrome Browser.

✅ **The implementation of responsiveness of the website is aptly implemented.**

No matter what size I make the browser screen, the features click, scroll, highlighting of the section, etc are working in all the sizes.

This is especially important to increase the reachability of your application. As we both know, there are many different types of screen sizes and resolutions, if our app doesn't render on someone's device, they won't be interested in using our product.

Here is a link to some basic rules for how to make UX better. These rules are also good for any product you make, not just websites.
https://www.springboard.com/blog/design/ux-design-principles/

✓

- Styling should be added for active states.
- Set CSS class active state when the element is in the viewport.
- The active section in the Navbar should be highlighted.

✅ **Styling has been added for the active states.**

This is a good thing, as the stateful design is a part of good UX. It makes the life of users easier. And helps them navigate your website much faster.

✓

There are at least 4 sections that have been added to the page.

✅ **There are at least 4 sections that have been added to the page.**

You added six sections and all are able to dynamically show in the navigation menu - great job!

Landing Page Behavior

Navigation is built dynamically as an unordered list. Start with empty `ul` and dynamically build navigation using `Append`, `appendChild`, and `innerHTML`.

✅ **Navigation is built dynamically as an unordered list.**

Good job for achieving this efficiently. I am impressed your use of `document.createDocumentFragment()`, this actually improves the speed of dynamically creating elements.

Extra Resource
Time comparison between different methods of element creation
https://howchoo.com/code/learn-the-slow-and-fast-way-to-append-elements-to-the-dom

It should be clear which section is being viewed while scrolling through the page.

> Tip: Detect the element location relative to the viewport using .getBoundingClientRect() built-in function.

✅ **It should be clear which section is being viewed while scrolling through the page.**

You have done smart and advanced programming here. Using `getBoundingClientRect` is some advance programming.

Tip: You can try some other way of doing this too. Read more about `intersectionObserver`
https://developer.mozilla.org/en-US/docs/Web/API/Intersection_Observer_API

When clicking an item from the navigation menu, the link should scroll to the appropriate section.
You can use the following methods to fulfill this criterion:

- Use addEventListener('click',....) to listen to the click event.
- Use preventDefault() as if there is a default event occurring we need to stop that.
- There are several javascript methods for scrolling, scroll(), scrollBy(), and scrollIntoView() are all acceptable.
- A smooth scrolling behavior is expected in the project.

✅ **When clicking an item from the navigation menu, the link should scroll to the appropriate section.**

✅ When clicking an item from the navigation menu, the link should scroll to the appropriate section.

✅ Should be using `addEventListener('click', ...)`.
✅ `scroll()`, `scrollBy()`, and `scrollIntoView()` are all acceptable.

Reading Material:
Why `preventDefault` is important https://developer.mozilla.org/en-US/docs/Web/API/Event/preventDefault

Documentation

✓

The ReadMe file should replace the given texts on the README template with specific information for this project. It doesn't have to be thorough, but should have some basic information, eg. project description, usage, dependencies, and use correct the markdown syntax.

References: markdown guide and example of README contents

It's certainly useful to summarize the project and note what skills it took to complete the project. This will only become more and more important as you build even more advanced projects for your portfolio.

There are some general conventions for what it should include and I have listed them here for you to refer to in the future:

1 General description of the project or content of the repository
2 List of what software, firmware and hardware you may require.
3 Installation instructions for the software and firmware.
4 List of files included in the project.
5 Copyright and licensing information.
6 Acknowledgements and credits for any resources or blogs that helped you create the project.
7 Known bugs

Here is also the GitHub help page on READMEs: https://help.github.com/articles/about-readmes/

✓

Comments should be present at the beginning of each procedure and class.

> **Bonus: Great to have comments before crucial code sections within the procedure.**
> **Refer to Udacity JavaScript Style Guide - Comments for standard best practices.**

You included useful comments throughout - this will be very helpful to anyone else (and even yourself!) who wants to check out your code later.

✓

Code should be formatted with consistent, logical, and easy-to-read formatting.

> **Tip: Carefully follow the best practices mentioned in the Code formatting section of the guide**

Nice work following consistent style as described in the style guide!

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START