UDACITY

‹ Return to Classroom

FEND Capstone - Travel App

REVIEW

CODE REVIEW 9

HISTORY

Meets Specifications

Awesome 🎉

After all that hard work... **YOU PASSED**!

Congratulations on completing your project and achieving a significant milestone. You've not only grasped the foundations of Webpack but also learned to make client-to-server API calls, a fundamental skill in web development. This experience will be invaluable in your coding journey. Keep up the excellent work and keep exploring new challenges! Your dedication and hard work are paying off.

So what's next?

We've barely scratched the surface of JavaScript build tools and API services with this project! So, the answer is quite simple:

STUDY MORE!

There are so many different build tools and you have to figure out which ones are useful for you.

My recommendations are:

- Understanding how Webpack works internally
- FreeCodeCamp Backend Dev and APIs Certification - Highly recommend!
- FreeCodeCamp FEND Curriculum
- Future of JavaScript Build Tools 2023

This Frontend Developer Roadmap provides a huge roadmap to becoming a Frontend Developer. Explore all the different concepts, frameworks, and tools that we use in the industry. All the best on your journey to becoming a front-end developer!

Development Environment and Architecture

The project should have a structure like the one shown below. All files shown must be present (Webpack may be split into multiple config files, and names may differ) and the app must successfully render a home page with clear design and functionality added when index.html is loaded in the browser. The project should not contain errors in the browser console.

```
- Root:
  - `package.json`
  - `readme.md`
  - `webpack.config.js`
  - src folder
    - server folder
      - `server.js` (name will vary)
    - client folder
      - `index.js`
      - html/views folder
        - `index.html`
      - js folder
        - `app.js` (name will vary)
      - styles folder
        - `style.scss` (name will vary - may be broke into partials)
```

✓ MEETS SPECIFICATION

Keeping code files separated and files organized allows you to maintain your project without headaches. A well-designed folder structure will help you quickly navigate files in your codebase.

Webpack config should contain at least 3 scripts, express server, build and test. Additionally, dev server may be included.

✓ MEETS SPECIFICATION

One of the most powerful features of the `package.json` file is the ability to define custom scripts, which can automate common tasks like building, testing, and deploying your project. They can also help to standardize development practices across your team by providing a consistent set of commands for building, testing, and deploying your project.

RESOURCES

- **npm documentation**: The official npm documentation provides detailed information about package.json and scripting. Visit the following link: npm - package.json. It covers how to define scripts, the available script lifecycle events, and how to run scripts.
- **Yarn documentation**: Yarn is another popular package manager that uses package.json. The Yarn documentation has a section dedicated to package.json and scripts. Check it out here: Yarn - package.json. It explains how to define scripts and use lifecycle hooks.

- Check if Jest has been installed and `npm run test` script is implemented to run Jest.
- There should be at least one test for the express server.
- There should be at least one test for the application javascript client.

✓ MEETS SPECIFICATION

Jest provides a simple, yet powerful API for writing and executing tests, and it includes built-in support for test coverage, mocking, and code snapshots.

There are several reasons why we need Jest tests:

1. Tests help to catch errors and bugs early in the development process
2. Tests help to ensure that the code functions as intended
3. Tests make it easier to refactor code without introducing new bugs or breaking existing functionality
4. Tests encourage developers to write cleaner, more modular code that is easier to test and maintain

Here are some links to Jest tutorials:

- Jest official documentation: https://jestjs.io/docs/getting-started
- Testing React with Jest and Enzyme: https://medium.com/@rossbulat/testing-react-with-jest-and-enzyme-20505fec4675
- Jest crash course for beginners: https://www.youtube.com/watch?v=7r4xVDI2vho

The project must have service workers installed.

✓ MEETS SPECIFICATION

Great work adding service workers to your project. Service workers are JavaScript files that run in the background of web browsers and enable features like offline caching, push notifications, and background sync.

RESOURCES

1. MDN Web Docs - Service Worker API: The MDN Web Docs provide a comprehensive guide to the Service Worker API, explaining its concepts, lifecycle, and usage.
2. Google Developers - Introduction to Service Workers: Google Developers' documentation offers an introduction to service workers, covering key concepts and examples. It also provides guidance on using service workers with Webpack.
3. Workbox Documentation: Workbox is a powerful library for managing service workers. The official Workbox documentation provides detailed information on using service workers with Webpack, along with practical examples and best practices.

HTML & CSS

- **All features are usable across modern desktop and phone browsers.**

- **Ensure the HTML elements, eg. texts and buttons, are proportionate and readable in small screen devices.**

✅ MEETS SPECIFICATION

Responsive web design is incredibly important nowadays! We must be mindful of all the different screen and device sizes. We also need to be careful with font-sizes and colors for accessibility.

Having a responsive design offers a much better user experience and can help drive more attention to your site. In general, you have about 2 seconds to capture your visitors. If they need to zoom or scroll to see or navigate to certain areas, you are jeopardizing the overall experience and impression of your site.

ADDITIONAL RESOURCES

- Implementations Tips: https://learn.shayhowe.com/advanced-html-css/responsive-web-design/
- Trends in Responsive Web Design: https://webflow.com/blog/responsive-web-design

Styling is set up in a logical way. All interactive elements have hover states.

✅ MEETS SPECIFICATION

Check for responsive units like `rem`, `em`, `%`, and `vh/vw`. If only pixels are used, consider migrating to responsive units to enhance responsive development and accommodate different screen sizes.

RESOURCES ON RESPONSIVE UNITS:

1. MDN Web Docs - CSS Units: The MDN Web Docs provide a comprehensive guide to CSS units, including responsive units like `rem`, `em`, `%`, and viewport units (`vh/vw`).
2. CSS-Tricks - Lengths of CSS: CSS-Tricks offers a guide to CSS units, explaining the different types of units and their usage in responsive web development.
3. FreeCodeCamp - CSS Unit Guide: This article discusses CSS units in the context of responsive development, highlighting the benefits of using responsive units and providing practical examples.

HTML structure should be indented properly with classes and ID's that make sense.

✅ MEETS SPECIFICATION

Great work formatting your code and following best code practices!

ADDITIONAL RESOURCES

- 5 tips on how to organize your JavaScript file: https://www.bdo.com/digital/insights/application-development/5-tips-organize-javascript-without-framework
- Basic JS Standard Conventions: https://www.w3schools.com/js/js_conventions.asp
- Airbnb's style guide is widely used in the industry: https://github.com/airbnb/javascript
- Google JS style guide: https://google.github.io/styleguide/jsguide.html

The design should clearly be different from the design used in projects 3 and 4.

✓ MEETS SPECIFICATION

API and JS Integration

`server.js` file should be taken directly from passed project 3 with the addition of added member: value pairs and the required API keys.

✓ MEETS SPECIFICATION

- At least one function should be imported.
- At least one event listener should be imported.
- (styles referenced in html/css)

✓ MEETS SPECIFICATION

1. There should be URLS and API Keys for at least 3 APIs, including Geonames, Weatherbit, and Pixabay. You can feel free to use more than 3 APIs.
2. There should be a primary object with placeholder member value pairs.
3. There should be a primary function that is exported to `index.js` (index.js file should import the functions from other files).

✅ MEETS SPECIFICATION

Great work effectively incorporating the required URLs and API keys. To further enhance your project's security, I recommend exploring the best practices for managing environment variables in a Node.js project. You can refer to resources like the dotenv package for guidance on securely managing sensitive information in your application.

Documentation

A `README` file is included detailing the app and all dependencies.

Other requirements:
The Readme file should have non-default text in it that is specific to this project. It doesn't have to be thorough, but should have some basic info. Bonus points if correct markdown is used.

✅ MEETS SPECIFICATION

Knowing how to write a great README is an important skill to have. It helps other developers and your future self with the setup of your project.

ADDITIONAL RESOURCES

- Beginners Guide to Writing a Kickass README: https://medium.com/@meakaakka/a-beginners-guide-to-writing-a-kickass-readme-7ac01da88ab3
- Udacity FREE Course: https://www.udacity.com/course/writing-readmes--ud777

Comments are present and effectively explain longer code procedure when necessary.

✅ MEETS SPECIFICATION

Good work! Your code is well-commented and understandable.

ADDITIONAL RESOURCES

- Clarifying Code with JavaScript Comments by Udacity: https://www.udacity.com/blog/2021/05/javascript-comments.html
- JSDoc is a popular way to comment on your code: https://www.youtube.com/watch?v=YK-GurROGIg

Code is formatted with consistent, logical, and easy-to-read formatting as described in the Udacity JavaScript Style Guide.

✅ MEETS SPECIFICATION

DOWNLOAD PROJECT

9 CODE REVIEW COMMENTS ›

RETURN TO PATH

Rate this review

START