Link shortener - coding task

Objective

Using **JavaScript** and **Vue**, your challenge is to build out a URL shortening web app, integrate with the shrtcode API, and get it looking as close to the design as possible. Feel free to use **TailwindCSS** to solve this challenge.
Please create a GitHub repository and push your code there.

Brief

URL shortening is a technique on the Web in which a Uniform Resource Locator (URL) may be made substantially shorter and still direct to the required page. This is achieved by using a redirect that links to the web page that has a long URL. For example, the URL "https://example.com/assets/category_B/subcategory_C/Foo/" can be shortened to "https://example.com/Foo", and the URL "http://example.com/about/index.html" can be shortened to "https://goo.gl/aO3Ssc ".

Your challenge is to build out this landing page, integrate with the shrtcode API and get it looking as close to the design as possible.

Your users should be able to:

- View the optimal layout for the site depending on their device's screen size
- Shorten any valid URL
- See a list of their shortened links, even after refreshing the browser
- Copy the shortened link to their clipboard in a single click
- Receive an error message when the form is submitted if:
    - The *input* field is empty

Your task is to build out the project to the designs inside the `/design` folder. You will find both a mobile and a desktop version of the design to work along with active states.

You will find all the required assets in the `/images` folder. The assets are already optimized. The designs are in JPG static format. This will mean that you'll need to use your best judgment for styles such as `font-size`, `padding`, and `margin`.

There is also a `style-guide.md` file containing the information you'll need, such as color palette and fonts.

Evaluation Criteria

- JavaScript best practices
- Show us your work through your commit history
- We're looking for you to produce working code, with enough room to demonstrate how to structure components in a small program
- Completeness: did you complete the features?
- Correctness: does the functionality act in sensible, thought-out ways?
- Maintainability: is it written in a clean, maintainable way?
- Testing: is the system adequately tested?

Deliverables

Make sure to include all source code in the GitHub repository.

Please organize, design, test, and document your code as if it were going into production - then push your changes.

Have fun building! 🚀

The e-Mission Team