Checklist: Project 3

Here is the checklist for the third project. It consists of two parts, just like the previous checklists. The first part lists the cases in which your project can be rejected for review, while the second part covers all the criteria that must be met for your project to be accepted.

Project rejected without review

General:

- Required corrections from a previous iteration (or iterations) haven't been completed.
- There are questions addressed to the reviewers in the project.

Layout mistakes:

- One or several blocks are incomplete or displayed incorrectly in Google Chrome or Firefox.
- Five or more elements are missing.
- One or more sections are missing.
- Blocks are not ordered according to the brief requirements.
- Alignment or CSS layout techniques (flexbox and positioning elements) are not used.
- The project has no media query and a responsive layout is not implemented.
- A font is specified incorrectly.
- Five or more text elements have a fixed height.

Mistakes in the code structure:

- More than five sections of code have HTML nesting errors.
- More than five sections of code have incorrectly written CSS declarations.

Incorrect application of BEM principles:

- Class names contain numbers, as in `<div class="block__element-1">`, or numerals, like `<div class="block__element-one">` or `<div class="block__first-element">`.
- Not all files are organized according to Flat BEM principles.

Project requirements

General

- The project contains:
  - A `blocks` directory with BEM blocks stored in separate CSS files (Flat BEM)
  - An `images` directory with all the images
  - A `pages` directory with an `index.css` file stored inside it
  - A `vendor` directory with `normalize.css`, `fonts.css` files, and a `fonts` directory stored inside it
  - The `index.html` file
  - A `README.md` file
  - A `.prettierignore` file, which tells Prettier to ignore `normalize.css`
  - A `.gitignore` file, which tells Git to ignore `.DS_Store`
- Stylesheets are connected in a separate file.
- All the sections of the design have been coded.
- The `README.md` file contains the following:
  1. The project's name
  2. A description of the project and its functionality
  3. A description of the technologies and techniques used
  4. Pictures, GIFs, or screenshots that detail the project features (highly recommended)
  5. The link to your deployed project on GitHub Pages
- There are no typos in the HTML and CSS, and the code is valid. We recommend using the W3C Markup Validator to check your code for validity.

- The code is well-formatted using the Prettier.
- The page has six photo cards. The photos and location names are up to you.

HTML/CSS

General

- `normalize.css` is imported into `index.css` before other CSS files.
- `viewport` is set correctly, `title` and `lang` are used.
- The layout doesn't break between breakpoints:
  - Media queries are properly used so that the layout is maintained at all resolutions.
  - Displays with a width of `320px` or more don't have horizontal scrolling.
  - No text overflows its corresponding block boundary.
- Aside from the `<header>`, elements should not touch the sides of the container on any screen larger than `320px`.
- The cumulative value of the `width`, `padding`, and `margin` properties for each element is specified to be within `30px` of the design requirements at the `1440px` breakpoint.
  - It should also be within `10px` of the design requirements at `320px`.
  - In the intermediate dimensions, the layout should look similar to either the desktop or mobile view.

Semantics

- Headings are made using the appropriate tags (from `<h1>` to `<h6>`). Text blocks are made with `<p>` tags. The `<header>`, `<main>`, `<footer>`, `<section>`, `<ul>`, and `<li>` elements are present in the code and they are used for their intended purposes.
- `<b>`, `<br>`, and `<i>` elements are not used.
- Elements are not wrapped in `<div>` tags unless required for alignment purposes.

Headings

- The page has a first-level heading. The hierarchy of headings is consistent, and there are no headings missing.

The BEM Methodology

- Stylesheets do not use tag selectors to set CSS rules.
- No more than two selectors are nested in any CSS rule.
- Stylesheets and images are located in separate folders and organized into blocks. Files are all organized within a Flat BEM file structure.
- Blocks are imported into the corresponding page file. CSS rules for elements and modifiers are declared in the corresponding CSS block file.
- Modifiers do not contain duplicate styles of the element or block being modified. They contain only the changing properties.
- There are no class attributes with a BEM modifier as their only value. For example, this is wrong: `<div class="block__element_mod-name_mod-value">`. This is correct: `<div class="block__element block__element_mod-name_mod-value">`.
- Components that share similar styles or functions constitute a single BEM entity, i.e., a block or an element.

CSS

- The page content is centered.
- The page has no white margins.
- `!important` is not used.
- The `Poppins` font family is used. Font sizes, weights, and colors are the same as in the design.
- System fonts are connected as alternatives to each of your fonts. No extra fonts are connected, and the `fonts` folder doesn't contain any extra font files.
- Relative values and appropriate CSS properties are used for setting block sizes. For example, use `max-width` instead of `width` for text elements.
- The padding for each section is set separately; negative values of the `margin` are not used.

- Margins are not used unnecessarily (for positioning elements on the page).
- Text elements don't have a fixed height, only have a fixed width if specified in the design, and will expand if more text is added to them.
- Absolute positioning is only used in instances where static or relative positioning can't be used.
- For elements with absolute positioning, `top` and `bottom` declarations are not used simultaneously, and neither are `left` and `right`.
- A horizontal scroll bar does not appear when changing the width of the preview window in the debugger.
- Breakpoints are grouped together. If there's a small difference in pixels between breakpoints in the `@media` at-rule, e.g., `max-width: 1044px` and `max-width: 1080px`, it's better to change the resolution in one of the media queries so that they have the same value.
- The project visually matches the design and is displayed correctly for all screen resolutions, as shown in the design.
- The same styles are not duplicated across different media queries. Each `@media` at-rule must only describe styles that are different from another `@media` at-rule.
- The maximum width is set for the content according to the design.
- To prohibit text wrapping and stretching of the card’s title, use the following properties:

```
.block {
  text-overflow: ellipsis;
  white-space: nowrap;
  overflow: hidden;
}
```

- To clamp the maximum rows of text for the profile elements and hide overflow text, use the following properties:

```css
.block {
  text-overflow: ellipsis;
  overflow: hidden;
  display: -webkit-box;
  -webkit-line-clamp: 3;
  -webkit-box-orient: vertical;
}
```

Interface accessibility

- All links and interactive elements have a `:hover` state.
- All `<img>` elements have an `alt` attribute containing a description in the page language.

Video checklist

General

- Video and sound is clear `/ 10.0`
- The video is at least 5 minutes long `/ 10.0`

Questions

1. Brief description of the project
   a. Project features `/ 10.0`
   b. Technologies used `/ 10.0`
2. Cards layout explanation
   a. Describe the chosen approach: flex or grid `/ 5.0`
   b. Tell what properties have been used `/ 2.5`
   c. Show how cards layout behaves at different resolutions `/ 2.5`
3. Adaptive design explanation
   a. Show how your design looks on desktop, tablet, and mobile resolutions `/ 5.0`

    b. Show examples in the code of how to change the display at different resolutions `/ 2.5`

4. Semantic tags usage
    a. Show examples of semantic tags in your code `/ 2.5`
    b. Explain why they should be used `/ 5.0`
5. BEM
    a. What is a block/element/modifier? `/ 5.0`
    b. How do you structure files and CSS code according to BEM? `/ 5.0`
6. CSS files connection
    a. Explain why the stylesheets in the `index.css` are connected in this order `/ 5.0`
    b. Explain why we need `normalize.css` file `/ 5.0`
7. Fonts usage
    a. Show how you connected the fonts in your project `/ 5.0`
    b. Show alternative fonts you have used `/ 2.5`
    c. Explain why do we need alternative fonts `/ 2.5`
8. Challenges
    a. Tell what the challenges were in doing the project and what you learned from it `/ 5.0`

These checklists will get stricter as time passes so that the requirements match those of an actual project. This checklist and the project description are already very similar to something you'd expect to receive for a real job. From the next sprint onward, checklists will only contain the requirements without accompanying text. This will become normal to you by the end of the course.