Second Project:
Advanced HTML and CSS

Stage 2. Creating a website layout

Practicum
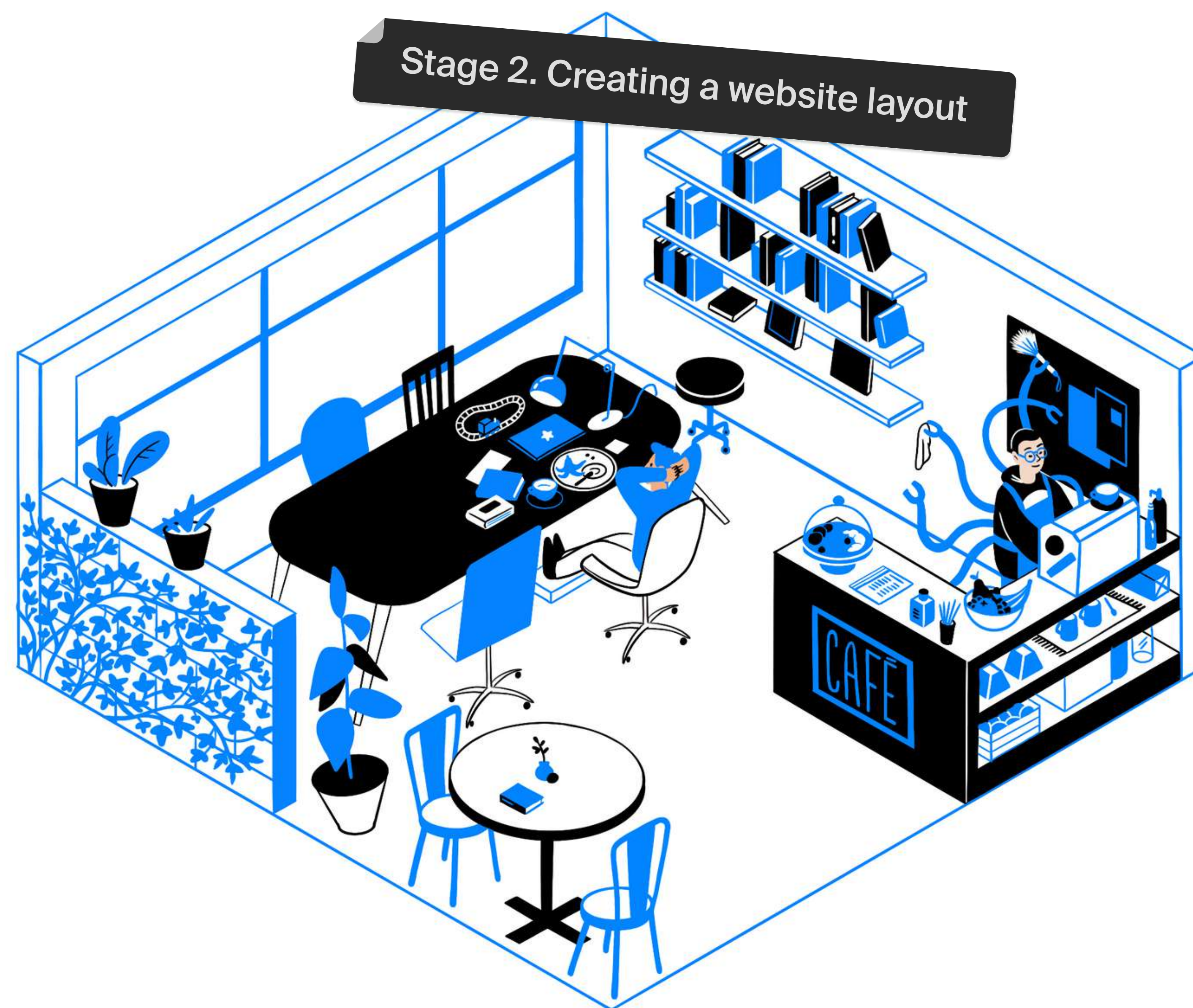
Introduction

The first thing you'll notice about this project is that it's similar to Project 1. Although you'll be building Project 2 from scratch, you're free to reuse blocks from your previous project and adapt the HTML and CSS accordingly. Finding similarities and reusing your code will help a lot! When planning out your project, consider how the structure and styles differ. This brief will give you all the dimensions, which should help simplify things.

This brief doesn't have many explicit step-by-step instructions compared to the previous project. You are becoming more advanced, so this lack of instructions is intentional.

As a budding software engineer interested in web development, you might be curious about working with web designers and why specific designs look the way they do. Here's a comment from our designer:

> *This is the website of the fictional coffee shop in the Practicum Library.*
> *The design follows the style of the Library website; however, since the coffee shop itself is less formal, this website is more colorful and uses more graphics.*
> *The circles in the background symbolize coffee tables and cups.*
> *I suggest animating one or two of these circles. The last circle in the "About" section should pulsate, representing the heart of the coffee shop.*
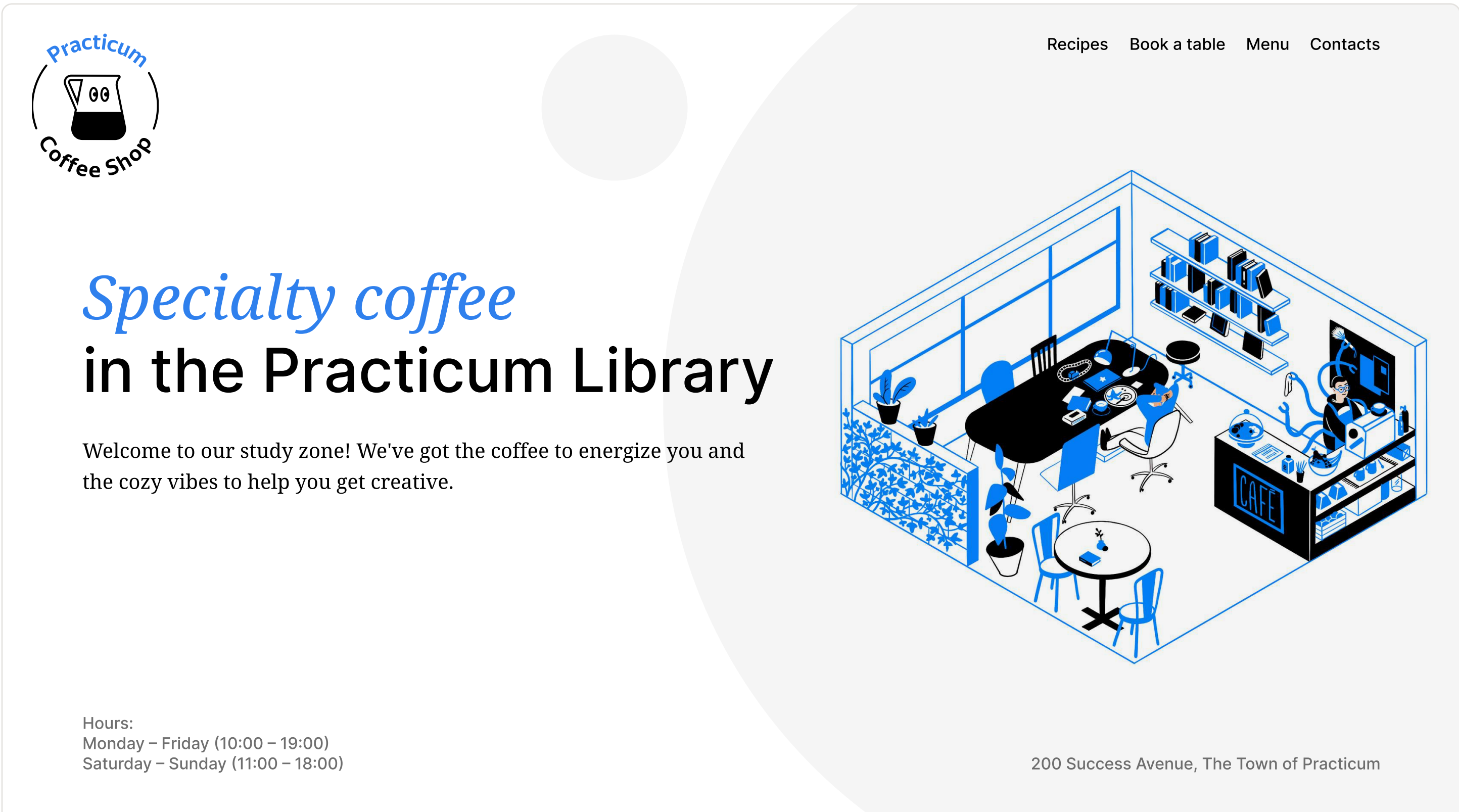
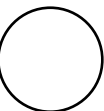*— Danila, Practicum designer*

Table of contents

Stage 2 Overview

In this stage of the project, you'll create the first two parts of your webpage: the header and the menu section. By the end of the stage, it should look like this:

Colors:

◯ #ffffff (header background, contrast font color)

⬤ #000000 (main font color)

⬤ #2f80ed (accent, button)

⬤ #838383 (input borders and placeholders)

⬤ #6F6F6F (hours and address)

⬤ #0A2750 (recipes background)

Recipes   Book a table   Menu   Contacts

Specialty coffee
in the Practicum Library

Welcome to our study zone! We've got the coffee to energize you and the cozy vibes to help you get creative.

Hours:
Monday – Friday (10:00 – 19:00)
Saturday – Sunday (11:00 – 18:00)

200 Success Avenue, The Town of Practicum

Recipes

Check out some recipes we've collected for your home-brewing convenience:

Aeropress recipe   ~5 min

French press recipe   ~15 min

1. Step-by-step implementation

Take the page content from the `texts.md` file that was provided on the project page. You don't need to add it to your repo, you'll just copy and paste the texts as required.

You'll work through each of the sections on the page one by one. For each section in the brief, we'll adhere to the following structure:

- Required elements and class names

- Sizes and spacing

- Typography

All the elements and class names shown in the brief are required, but you can create additional elements, such as wrappers, if needed. Just ensure that you adhere to the BEM naming principles.

Note that you can only use absolute positioning in a few cases, all of which are specified in the brief.

1.1. Header
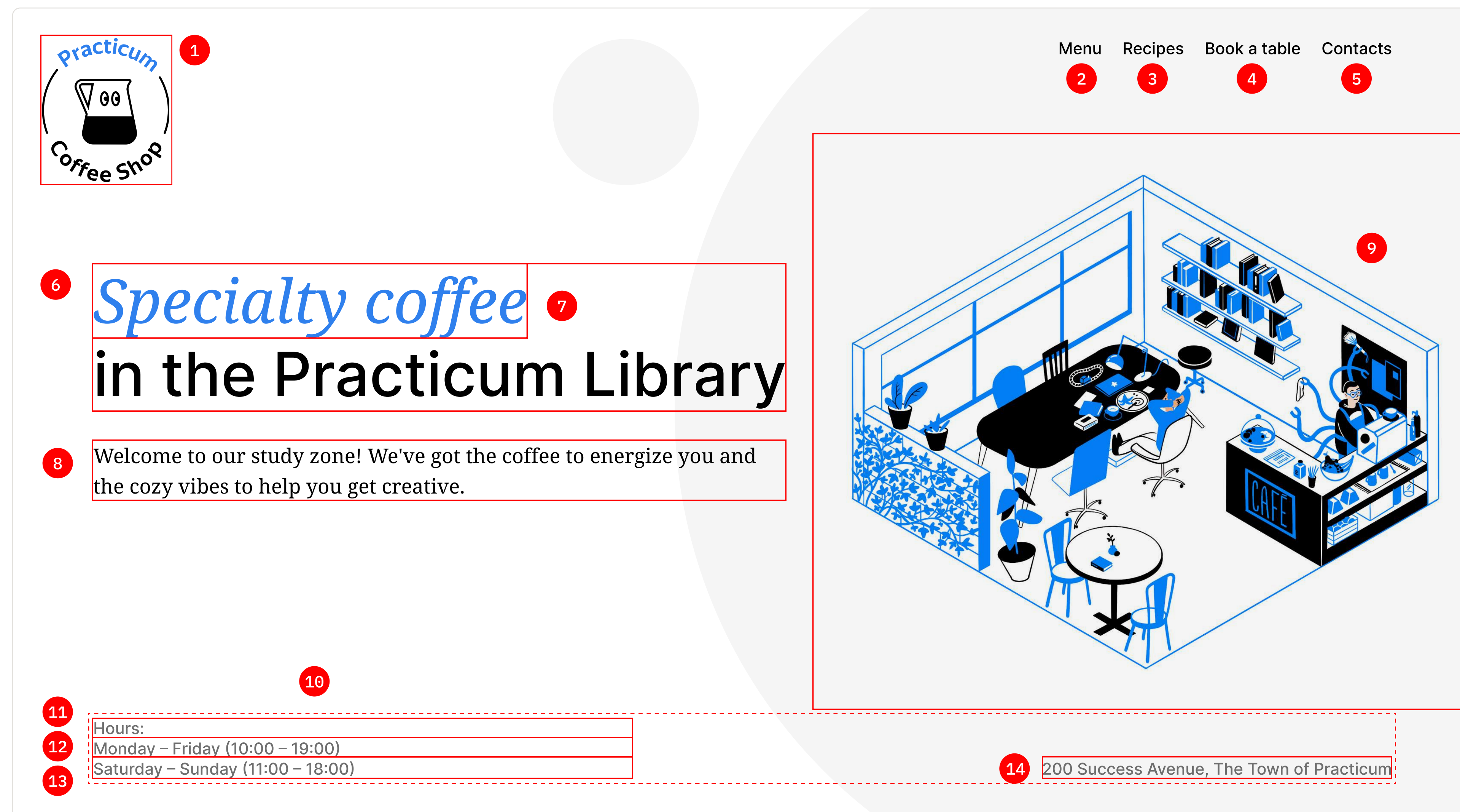
Pro tip:

How can you break lines of text into multiple lines and control the exact placement of those breaks? You could use `<br>` but this may cause accessibility issues due to the way `<br>` is handled by screen readers.

One trick is to wrap the desired lines in `<span>` elements with `display: block;` applied. Another option could be to use a non-breaking space: a special space character that can't be broken by a new line. Simply insert ` ` in your HTML between the words that should stick together.

Required elements and class names

Background: Set the background color to `#FFFFFF`. For the background image, provide the URL to the `background_header.svg` file. The background doesn't repeat and its position is centered. Set the size value to `cover`.

1. block: `logo`. Find the corresponding image in the `images` folder. You can use absolute positioning for this element.

2-5. block: `nav`, element: `link`. The instructions for writing class names will follow this format throughout the brief. In this case, it means that your class name should be written as follows: `nav__link`.

Remember to use the `nav` semantic element. You can build the navigation bar in a similar way to what you did in Project 1.

6. block: `header`, element: `title`. Since this is the title of our page, use an `<h1>` heading. Also, keep the text on separate lines, as shown in the design. It may be a small detail, but small details can have a big impact on the look of the site.

7. block: `header`, element: `span-accent`. This is for the blue text in italics that says "Specialty coffee." This text should be wrapped in a set of `<span>` tags. The `<span>` element is an inline tag often used to apply separate styles to part of a paragraph in order to make it stand out.

8. block: `header`, element: `description`.

9. block: `header`, element: `image`. You can use absolute positioning for this element.

10. block: `header`, element: `footer`. A wrapper for elements 10 through 13.

11 - 14. block: `header`, element: `paragraph`. Think about how you can group the three paragraphs as shown in the design. Hint: your knowledge of flexbox and a few divs can help.

30px

30px

Menu   Recipes   Book a table   Contacts

20px

80px

30px

124px

646x568px

202px

125x142px

80px

max-width:729px

5px

Specialty coffee
in the Practicum Library

80px

80px

28px

Welcome to our study zone! We've got the coffee to energize you and the cozy vibes to help you get creative.

max-width:683px

80px

Hours:
Monday – Friday (10:00 – 19:00)
Saturday – Sunday (11:00 – 18:00)

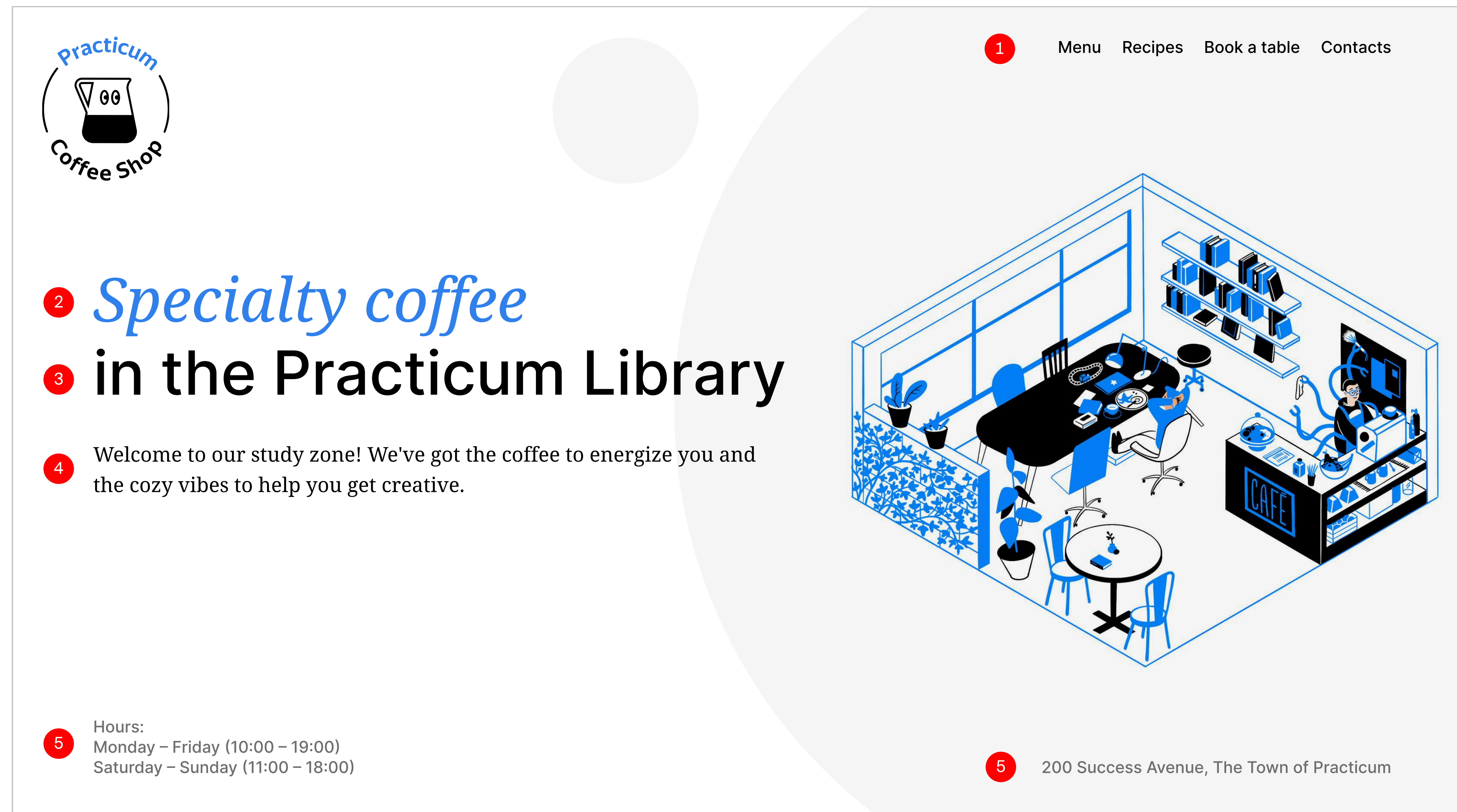200 Success Avenue, The Town of Practicum

80px

40px

40px

80px

Sizes and spacing

Minimum block height: 720px, maximum block height: 800px.

Set the regular screen height to 100vh (100% of the viewport) so that it fits screen sizes between 720px and 800px by default.

Add all the sizes according to the picture above.

1
```
font-family: "Inter";
font-style: normal;
font-weight: 500;
font-size: 16px;
line-height: 20px;
```

2
```
font-family: "Noto Serif";
font-style: italic;
font-weight: normal;
font-size: 60px;
line-height: 72px;
color: #2F80ED;
```

3
```
font-family: "Inter";
font-style: normal;
font-weight: 500;
font-size: 60px;
line-height: 72px;
```

4
```
font-family: "Noto Serif";
font-style: normal;
font-weight: normal;
font-size: 20px;
line-height: 30px;
```

5
```
font-family: "Inter";
font-style: normal;
font-weight: 500;
font-size: 16px;
line-height: 20px;
color: #838383
```

Menu   Recipes   Book a table   Contacts

Specialty coffee
in the Practicum Library

Welcome to our study zone! We've got the coffee to energize you and the cozy vibes to help you get creative.

Hours:
Monday – Friday (10:00 – 19:00)
Saturday – Sunday (11:00 – 18:00)

200 Success Avenue, The Town of Practicum

Typography

The numbered code snippets correspond to their respective numbered text elements. We don't need to add all these styles to each element — take a moment to consider the reason why.

First, remember the styles we assigned to the `page` block? Generally, its child elements will inherit these styles, so we don't need to give them the same styles again. There are exceptions to this, however, noted in the pro-tip below.

> **! Pro tip:**
>
> As we noted before, not all properties are inherited by all elements. For instance, heading tags like `<h1>`,`<h2>`, and `<h3>` will not inherit the `font-size` property, and `<a>` tags don't inherit `color`. In these cases, you can either set the value explicitly or instruct the element to inherit the property from its parent with the `inherit` keyword.

Second, you don't need to write values that are the same as the defaults, such as `font-style: normal;` and `font-weight: normal;` (which is equivalent to `font-weight: 400;`).

Lastly, keep in mind that you should always include appropriate fallback fonts.

- For Inter, use Arial as a fallback and use the generic sans-serif family in case the user doesn't have either of the other fonts.

- For Noto Serif, use Times New Roman as a fallback and use the generic serif family in case the user doesn't have either of the other fonts.

With all this in mind, write the necessary styles for the text elements in CSS.

1.2. Section: Recipes
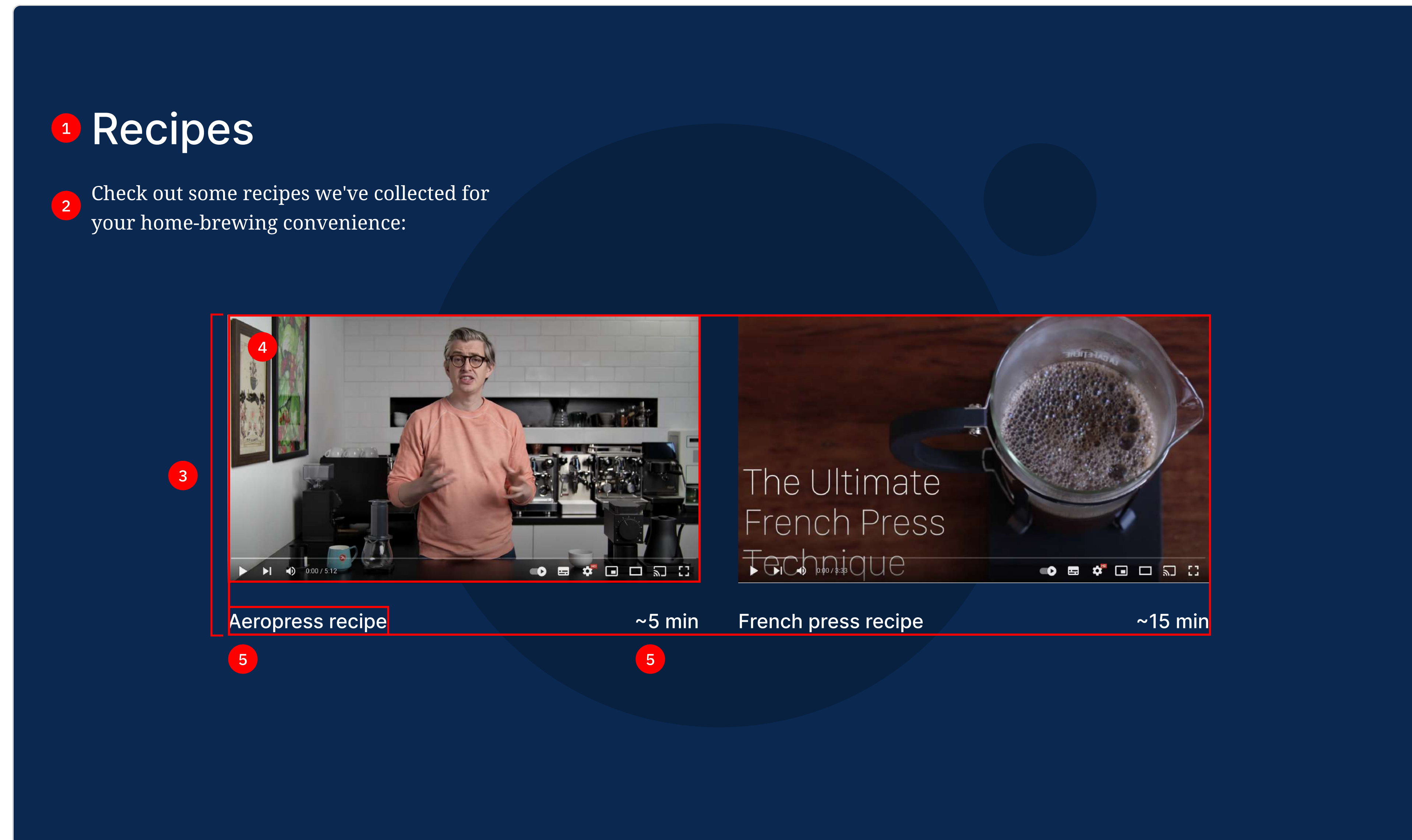
1. block: `recipes`, element: `title`.

2. block: `recipes`, element: `subtitle`.

3. block: `recipes`, element: `videos`.

4. block: `recipes`, element: `iframe`.

5. block: `recipes`, element: `video-caption`.

Required elements and class names

Background: Set the background color to `#0a2750`. For the background image, provide the URL to `background_recipes.svg`. The background doesn't repeat and its position is centred. Set the size value to `cover`.
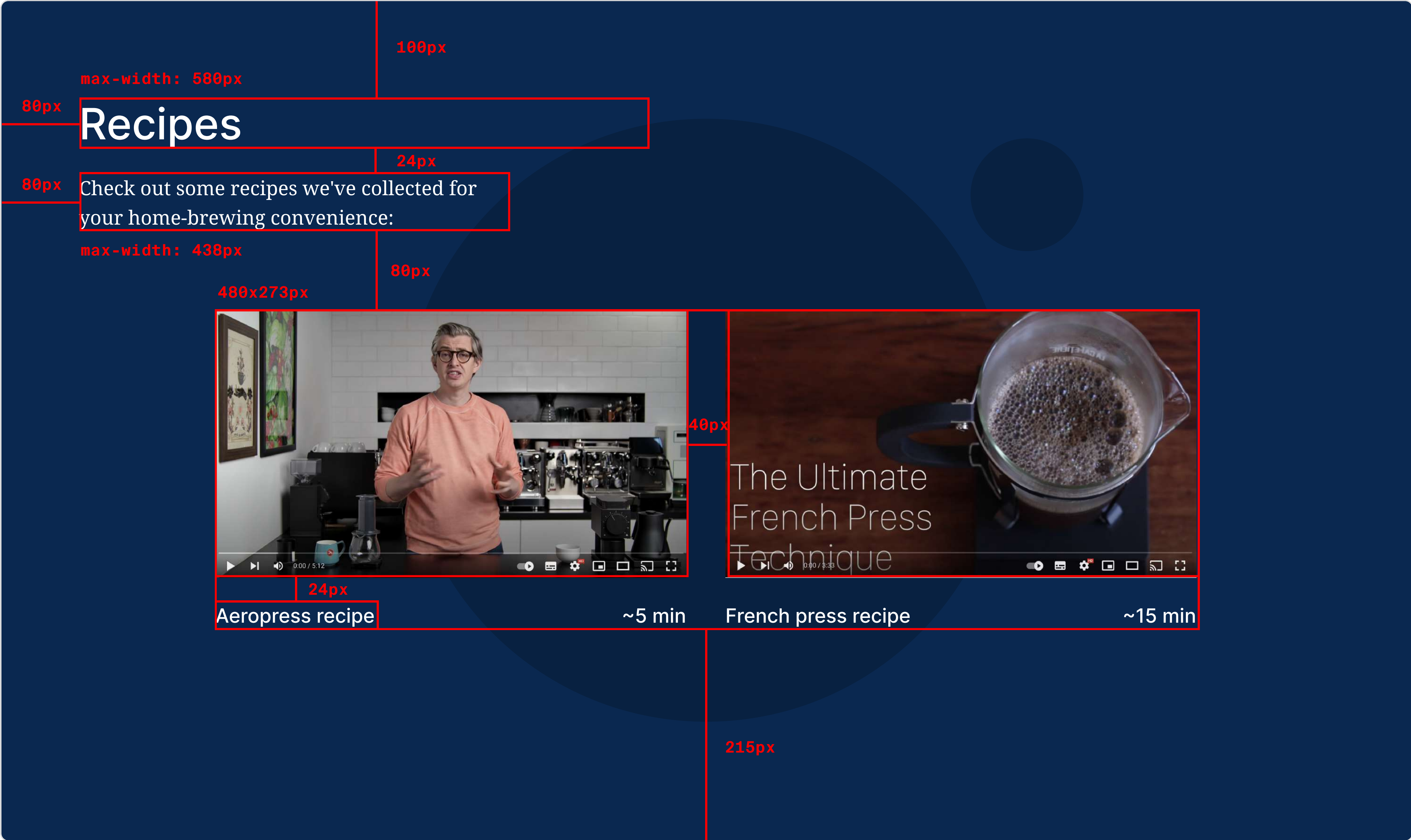
Use the `<iframe>` tag to add videos. You can use these links or choose your own (make sure to change the description and time accordingly):

https://www.youtube.com/watch?v=j6VlT_jUVPc

https://www.youtube.com/watch?v=st571DYYTR8

> ⊙ **Pro tip:**
>
> Make sure to use the "embed" link, and not the "watch" link. Refer to our YouTube API lesson if you need a reminder about the difference.

Sizes and spacing

Add all the sizes according to the picture above.

The `recipes__videos` element should be centered inside the section.

1 Recipes

2 Check out some recipes we've collected for
your home-brewing convenience:

The Ultimate
French Press
Technique

3 Aeropress recipe   ~5 min   French press recipe   ~15 min

1
```
font-family: "Inter";
font-style: normal;
font-weight: 500;
font-size: 44px;
line-height: 52px;
```

2
```
font-family: "Noto Serif";
font-style: normal;
font-weight: normal;
font-size: 20px;
line-height: 30px;
```

3
```
font-family: "Inter";
font-style: normal;
font-weight: 500;
font-size: 20px;
line-height: 30px;
```

Typography

The numbered code snippets correspond to their respective numbered
text elements.

1.3. Anchor tag functionality

Just like in the previous project, each anchor tag in the navigation bar should lead to the appropriate section. Remember that:

- The target sections of the webpage should have an appropriate `id`.

- The `href` of each anchor tag should be in the form `href="#section-id"`, where `section-id` is the `id` of the corresponding section.

2. Self-review

You are nearing the finish line for this project stage. Before submitting, there is one last crucial task: conducting the self-review. To help, we've prepared a checklist:

☐ Compare each section of your webpage against the brief (open your project side-by-side with the brief and go through each point. Consider jotting down notes for yourself with things that you want to remember).

☐ Check the validity of your code using a <u>validator</u>.

☐ Check your project against all the checklist items (you'll find this in the project description on the platform).

☐ Make sure to remove any redundant comment lines. Keep the comments that help you better understand the project structure and facilitate your understanding of complex features.

Once you're finished, you're ready to submit the project. Awesome job!

Practicum