Exercise: Ensuring your application is accessible to users

Overview

The goal of this exercise is for you to review your application's accessibility and revise the project with improvements.

Scenario

It is very important to ensure any app you create is accessible to all users. As such, you need to ensure the accessibility of the app you have created for Little Lemon.

There are many ways that you can improve a React application's accessibility. One way that has been discussed in this course is the use of semantic markup. Another important accessibility improvement that ties in well with setting up semantic markup is the use of ARIA attributes

As you complete this exercise, you may find it helpful to review the following content:

- Web accessibility
- Designing for accessibility

You can also check out the **Accessibility** resource available at the official React docs website linked in the additional resources. This provides a nice overview of accessibility in React, including the points discussed here, plus a few additional points.

Instructions

Step 1: Improving the semantic markup you're using

Improve your app accessibility by improving the semantic markup you're using.

Note: This step's completion will depend on how much work you've already done on making your app's code semantic in the previous lesson items that were dedicated to this specific goal.

Step 2: Use ARIA attributes

Improve app accessibility by using ARIA attributes

Add at least the aria-label attribute and set it to **On Click**, as follows:

`aria-label="On Click"`

Tip: Remember, the aria-* attributes are an exception to the rule that all JSX properties and attributes use camelCase. In the case of ARIA attributes, it's proper JSX syntax to use the hyphen-cased syntax, just like in plain HTML.

Note: This step's completion will also depend on how much attention you've paid to the semantic structure of your app in the previous exercises.

Step 3: Labeling forms

To improve the accessibility of your forms, use the `label` element and the HTML `For` attribute, which maps to the id attribute of the actual form item, such as an input.

Note: This is important because it allows users to click on a specific form input label, and it will automatically focus into the given form input.

Conclusion

You have now practiced reviewing an application's accessibility and revising the project with improvements. Although these three steps are not all the different ways in which you can improve the accessibility of your React apps, you are three steps closer to ensuring your web app is accessible to all users.