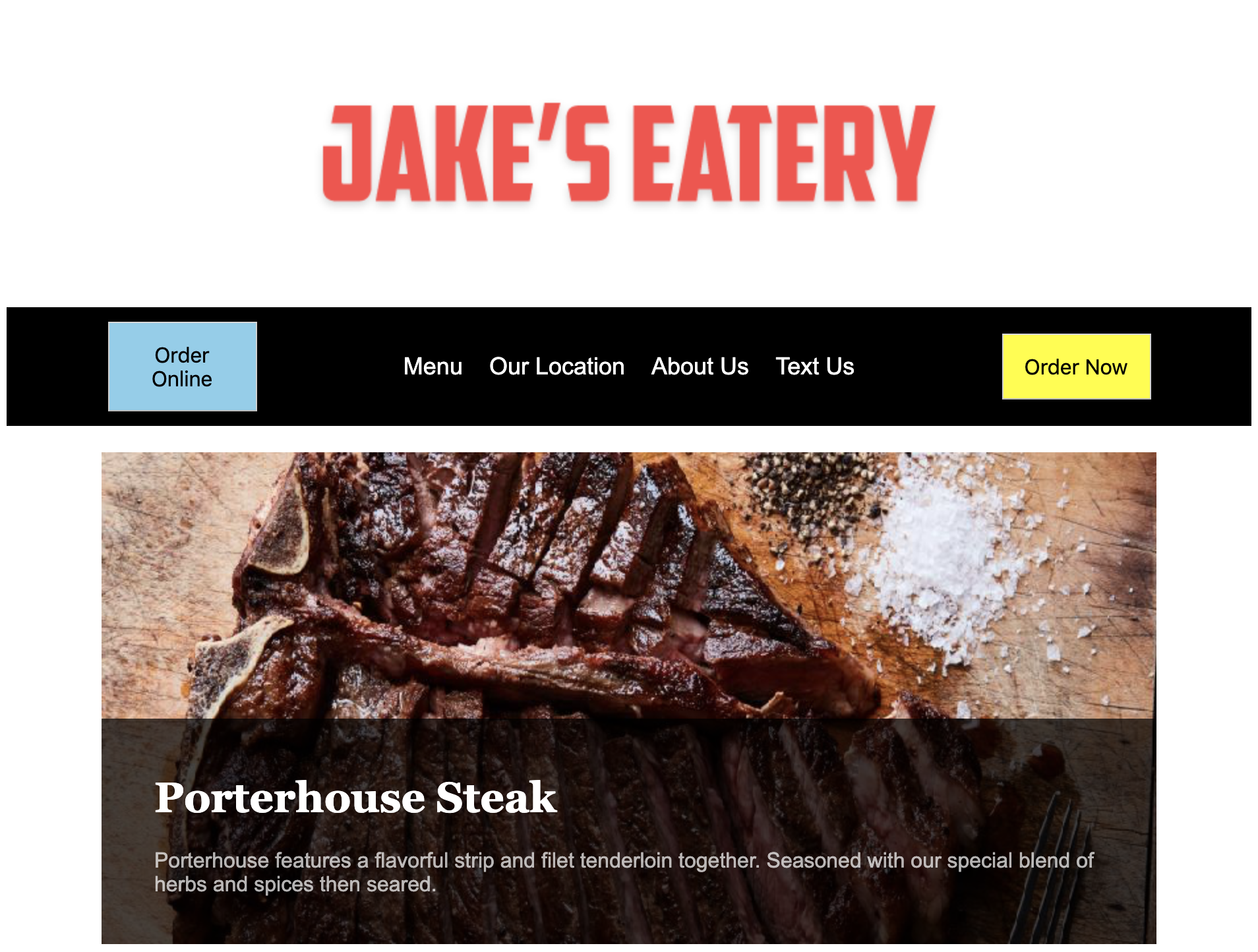
## **Overview**

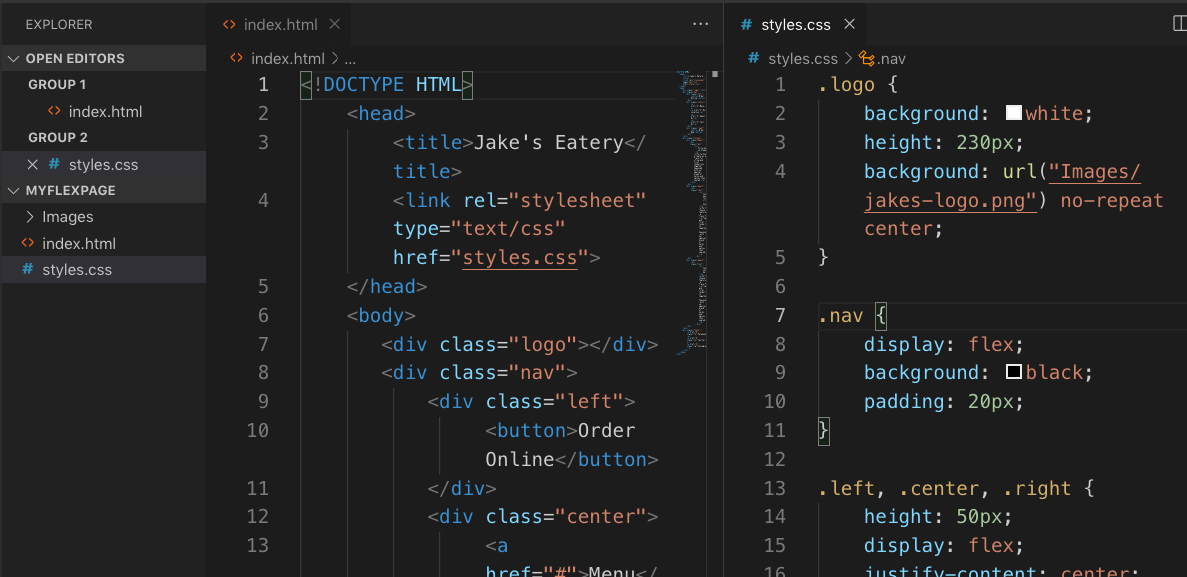
During this challenge, you will test your CSS Flexbox skills by building a webpage for your client, Jake’s Eatery.



You will also develop something else during this time: your workflow as a UX/UI developer.

## **Instructions**

**Step1: Set Up Your Workspace**

1. Open the files index.html and style.css.
   * These files are located in the Unit 18 Activities folder, inside **18.1\_activities.zip**.
2. Right-click on the tab for style.css and select **Split Right**. This will open the file in a new tab to the right of index.html.

**Step 2: Create the Navigation:**

1. Add a CSS selector that targets the logo class with the following properties:

| **Property** | **Value** |
| --- | --- |
| background | white |
| height | 230px |
| background-image | url("../images/jakes-logo.png") |
| background-repeat | no-repeat |
| background-position | center |

1. The HTML for the navigation area contains some hints for how to implement its layout.
   * Which <div> is the parent flex container? Which <div> is a child of the parent?

****

1. Add a selector that targets the nav class. This element will be the flex container for the navigation component of your layout.
   * Add the property display with a value of flex to make this your container.
   * Also add the properties background-color: black; and padding: 20px;.
2. Write a selector that targets three **different** classes all at once: left, center and right.
   * You can target multiple IDs, classes, and elements like so:

| .class1, .class2, .class3 {  ...  } |
| --- |

* + This will apply the same CSS to all selected IDs, classes, and elements and helps cut down the amount of duplicate CSS you have to write.
  + Add the properties height: 50px; and display: flex; to this selector.
  + Add the properties justify-content: center; and align-items: center; to center the elements both vertically and horizontally.

1. Next, create a selector that targets the left and right classes.
   * Add the property width: 25%; so that each element will occupy 25% of the container’s width.
2. Add a selector that targets the center class and set its width to 50%.
3. Write a selector that targets the <a> tags nested **inside** the center class.
   * You can target elements inside of IDs, classes, and elements class like so:

| .className elementName { } |
| --- |

* + Add the following properties to this selector:

| **Property** | **Value** |
| --- | --- |
| color | white |
| padding | 10px |
| font-size | 18px |
| text-decoration | none |
| font-family | Arial |

1. Next, create a selector to target the <button> element with the following properties:

| **Property** | **Value** |
| --- | --- |
| color | black |
| font-size | 16px |
| padding | 15px |
| width | 50% |

1. Lastly, create two selectors. One to target the <button> inside the right class. One to target the <button> inside the left class.
   * Add the property background-color: yellow; to the right button.
   * Add the property background-color: skyblue; to the left button.

**Step 3: Style the Content**

1. The HTML for the content area contains some hints for how to implement its layout. Can you tell which element is the parent flex container and which ones are the children?



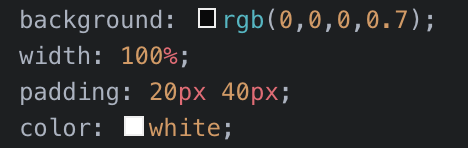
1. Add a selector that targets the parent flex container- the column class.
   * Add the following properties to create a column-based layout:

| **Property** | **Value** |
| --- | --- |
| display | flex |
| height | 1200px |
| width | 800px |
| flex-direction | column |
| margin | 0 auto |

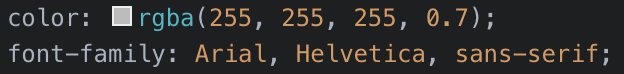
1. Next, add a selector that targets the meal1, meal2, and meal3 classes.
   * Add the following properties to create the children of the flex container:

| **Property** | **Value** |
| --- | --- |
| height | 100% |
| width | 100% |
| display | flex |
| align-items | flex-end |
| margin-bottom | 20px |
| background-repeat | no-repeat |
| background-position | center |

1. Create a selector that targets the meal1 class and add the margin-top: 20px; and background-image: url("../images/steak.png"); properties.
2. Target the meal2 class with a new selector. Add the background-image: url("../images/soup.png"); property.
3. Target the meal3 class with a new selector. Add the background-image: url("../images/cheesecake.png"); property.
4. Create a selector to target the text class and add the following properties:



1. Add a selector that targets the <p> element in the text class.
   * Add the following:



1. Add a selector that targets the <h1> element in the text class. Add the font-family: Georgia, 'Times New Roman', Times, serif; property.

**Step 4: Style the Footer**

1. Add the appropriate styling to make the footer area look like the image below:



**Bonus:**

1. How might you build this with CSS Flexbox? See if you can figure it out!