How do I Complete this Project?

This project is connected to the Intro to HTML and CSS course but depending on your background knowledge of HTML and CSS you may not need to take the whole thing to complete this project. Here's what you should do:

1. Download the design mockup file from the Downloadables section to your lower right hand corner and review it.
2. Identify the various boxes you will need to build in order to recreate this design.
3. Write your HTML and CSS files, continue to iterate until your work is representative of the design mockup.
4. Take the time to personalize your portfolio with custom colors, additional content and your own images.
5. Validate your HTML and CSS against the W3C's Validators. *Note: the validators consider the following errors, whereas Udacity accepts these errors as acceptable:*
    - HTML5: **Bad value X-UA-Compatible for attribute http-equiv on element meta** when using the X-UA-Compatible meta tag.
    - CSS3: **Property *[some property here]* is an unknown vendor extension** when using vendor prefixed properties (like -moz-box-sizing).

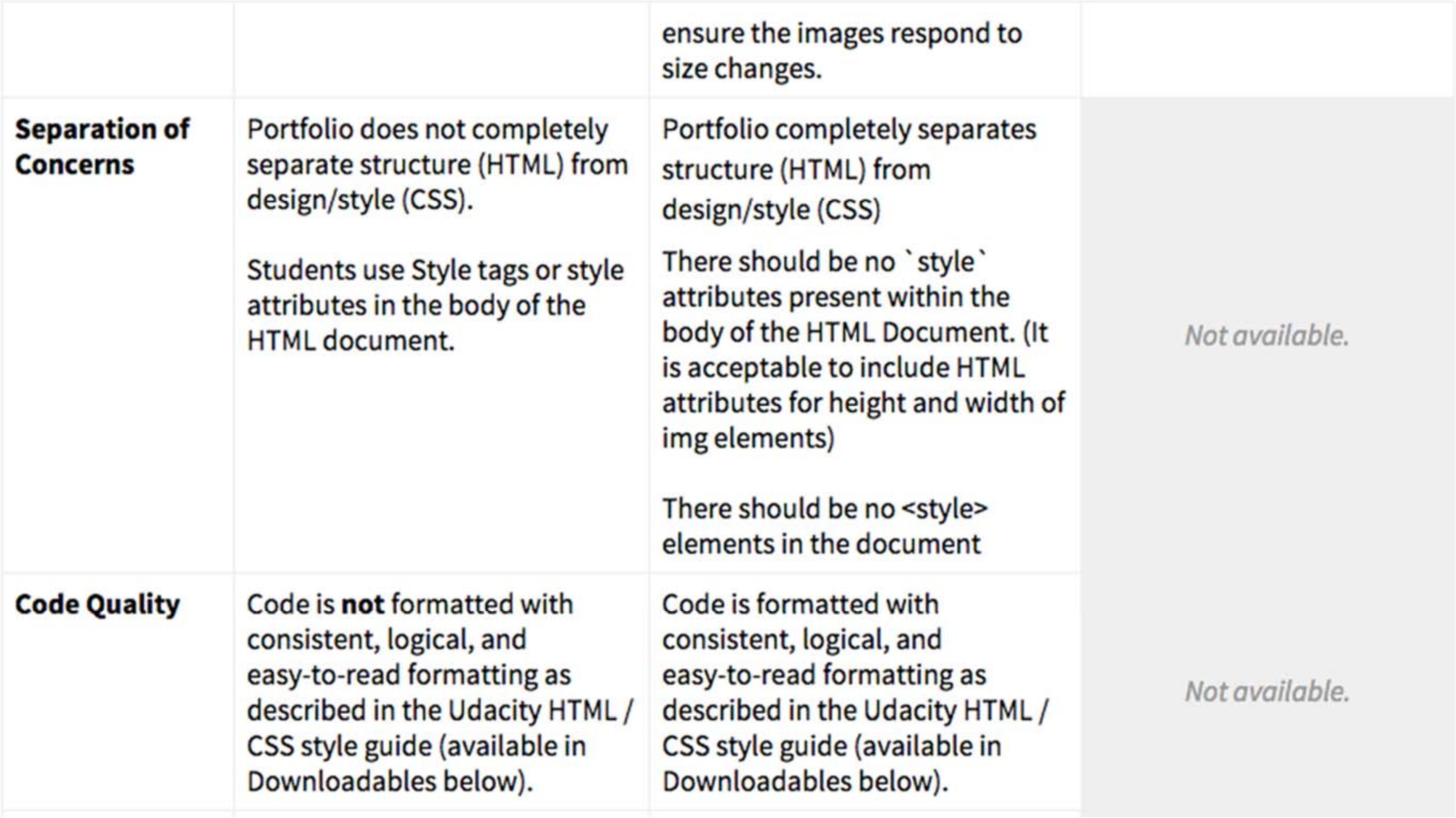
Evaluation

Your project will be evaluated by a Udacity Code Reviewer according to the rubric below. Be sure to review it thoroughly before you submit. All criteria must "meet specifications" in order to pass.

Build a Portfolio Site Project Rubric

Criteria	Does Not Meet Specifications	Meets Specifications	Exceeds Specifications (Completely Udacious)
Design	Page does not include at least 4 images, title text, paragraph text, or a logo. Some <div> tags are without a class or id. The page structure can not be interpreted from the DOM Student doesn’t utilize a grid based layout.	Page at a minimum includes at least 4 images, title text (i.e. <h1>, <h2>, …) , regular (paragraph) text (i.e. <p>), and a logo. No <div> tags are without a class or id. The page structure can be interpreted from the DOM. Student utilizes a grid based layout such that the body has an element that serves as a container for rows and columns.	Along with fulfilling everything in the ‘meets expectations’ students additionally: Use HTML5 Semantic tags such as <article>, <header>, <section>, etc. to add meaning to their code. Makes use of the flexbox layout module or a framework like Bootstrap, Foundation, etc.. for laying out their site. Personalizes the design beyond the provided mockup (e.g. grid layout, styles and images are customized.) Adds additional JavaScript functionality, while maintaining required components (Examples:

			Bootstrap Navbar, Polymer Component. Modals do not fulfill this specification.)
Responsiveness	No content is responsive nor displayed on one of the three required device sizes. Viewport meta tag is not included in the HTML (i.e. <meta name=”viewport” …)	All content is responsive and displays on all display sizes. (Desktop, Tablet, Mobile) **TIP:** Test responsiveness with Chrome Developer Tools device emulation by right-clicking anywhere on page, selecting ‘Inspect Element’, clicking the rectangle to the left of the Elements tab, select Apple iPad or Google Nexus 5 from Device drop-down list, and click reload. All content should be rendered on all three devices. It is not acceptable to hide content such as project images on mobile devices. Viewport meta tag must be included in HTML. If a CSS Framework is used (e.g. Bootstrap) the student uses classes provided by the framework to make images responsive, otherwise the student uses media-queries to	Along with fulfilling everything in the ‘meets expectations’ students additionally: Use srcset to provide optimized images to users on all device sizes.



		ensure the images respond to size changes.	
Separation of Concerns	Portfolio does not completely separate structure (HTML) from design/style (CSS). Students use Style tags or style attributes in the body of the HTML document.	Portfolio completely separates structure (HTML) from design/style (CSS) There should be no `style` attributes present within the body of the HTML Document. (It is acceptable to include HTML attributes for height and width of img elements) There should be no <style> elements in the document	*Not available.*
Code Quality	Code is **not** formatted with consistent, logical, and easy-to-read formatting as described in the Udacity HTML / CSS style guide (available in Downloadables below).	Code is formatted with consistent, logical, and easy-to-read formatting as described in the Udacity HTML / CSS style guide (available in Downloadables below).	*Not available.*

Style Guidelines

CSS REQUIRED Rules

- CSS - Use consistent indentation (tabs or spaces). (See: CSS General Formatting Rules-Indentation)

- CSS - Selectors, properties and property values (with the exception of strings) should be lowercase, including letters in hexadecimal color values. For example #f06c13 instead of #F06C13 and #ccc rather than #CCC. (See: CSS-General Formatting Rules-Capitalization.)
- CSS - Use IDs and class names that are meaningful or generic. (See: CSS-ID and Class Naming.)
- CSS - Use ID and class names that are as short as possible but as long as necessary. (See: CSS-ID and Class Naming.)
- CSS - Use Shorthand properties in all possible places for margin, border, padding, background. Examples: https://developer.mozilla.org/en-US/docs/Web/CSS/Shorthand_properties. (See: CSS Style Rules-Shorthand Properties.)
- CSS - Indent block content between curly braces. (See: CSS Formatting Rules-Block Content Indentation.)
- CSS - End all declarations with a semicolon. (See: CSS Formatting Rules-Declaration Stops.)
- CSS - Add single spaces after each property name's colon (See: CSS Formatting Rules-Property Name Stops.)
- CSS - Remove trailing white spaces from code. (See: CSS General Formatting Rules-Trailing Whitespace)
- CSS - there should be a single space after each colon. (See: CSS Formatting Rules-Property Name Stops.)
- CSS - Include a single space before the opening curly brace. (See: CSS-Declaration Block Separation.)
- CSS - ID and class names should not be used with type selectors. (See Style Guide: CSS Style Rules-Type Selectors.)
- CSS - Separate style rules with a blank line. (See: CSS-Rule Separations.)

CSS SUGGESTED Rules

- CSS - Remove units of measure after 0 values. (See: CSS Style Rules-0 and Units.)
- CSS - Use leading zeros for values in the code. (See: CSS Style Rules-Leading 0s.)
- CSS - Use 3 character hexadecimal notation for color in all possible places. For example #ccc rather than #cccccc. (See: CSS Style Rules-Hexadecimal Notation.)
- CSS - Separate words in ID and class names with a hyphen. (See: CSS Style Rules-ID and Class Name Delimiters.)
- CSS - Remove CSS hacks or user detection from code. (See: CSS Style Rules-Hacks.)
- CSS - Separate selectors and declarations with new lines. (See: CSS-Selector and Declaration Separation.)
- CSS - Use double quotes for all attribute selectors and property values. (See: CSS Formatting Rules-CSS Quotation Marks.)
- CSS - Identify groups of related style rules with section comments. (See: CSS Meta Rules-Section Comments.)

HTML REQUIRED Rules

- HTML - Element and attribute names should all be lowercase. (See: Capitalization.)
- HTML - Use HTML5. (See: HTML Style Rules-Document Type.)
- HTML - Use UTF-8 as character encoding. (See: General Meta Rules-Encoding.)
- HTML - Place all block/list/table elements on a separate line and indent child elements. (See: HTML-General Formatting.)
- HTML - Use semantic tags. (See: HTML Style Rules-Semantics.)

- HTML - Provide alternate content for multimedia. (See: HTML Style Rules-Multimedia Fallback.)
- HTML - Remove trailing white spaces from code. (See: HTML - General Formatting Rules - Trailing Whitespace.)
- HTML - Use consistent indentation (tabs or spaces) (See HTML - General Formatting Rules - Indentation)

HTML SUGGESTED Rules

- HTML - Use comments to explain the code. (See: General Meta Rules-Comments.)
- HTML - Mark action items with TODO. (See: General Meta Rules-Action Items.)
- HTML - Remove entity references. (See: HTML Style Rules-Entity References.)
- HTML - Remove type attribute from link and script elements. (See: HTML Style Rules-Type Attributes.)
- HTML - Use double rather than single quotation marks for attribute values. (See: HTML-HTML Quotation Marks.)

Submission

When you're ready to submit your project go back to your Udacity Home, click on Project 1, and we'll walk you through the rest of the submission process. Due to the high volume of submissions we receive, please allow up up to **7 business days** for your evaluation to be returned.

If you are having any problems submitting your project or wish to check on the status of your submission, please email us at **frontend-project@udacity.com** or visit us in the discussion forums.

What's Next?

You will get an email as soon as your reviewer has feedback for you. In the meantime, review your next project and feel free to get started on it or the courses supporting it!