WEB222 Final Assessment – Fall 2022

Overview

The WEB222 final assessment is equivalent to a final exam but can be done remotely and submitted online.

The assessment has three parts and builds on your knowledge from the course to create websites. Specifically, you will be asked to do the following:

1. Create a Developer Profile Website with information about yourself, your skills, and your expertise (worth 10%)
2. Research and Implement a Static Hosting solution for your final assignment, so it can be viewed online (worth 5%)
3. Create a Reflection Video Walk-Through about the work you did in steps 1 & 2 and upload it to YouTube (worth 5%)

Submission

All 3 parts of your final assessment are due by **Wednesday December 7 by Midnight**. In your submission to Blackboard, please include:

* A .zip file of your complete web site code (part 1)
* A link to your publicly hosted web site (part 2)
* A link to your private YouTube video (part 3). NOTE: make sure you publish the video **unlisted/private**, so only the people who have the link can view the video

You **may** consult your notes, class recordings, and use the web, but you **may not** discuss any part of this final assessment with other students. All work must be your own. You may not collaborate or copy material from other sources or students.

You **may not** use existing web sites. All HTML, CSS, and JavaScript must be your own work.

Honesty Statement:

*I declare that my assessment is wholly my own work in accordance with Seneca Academic Policy. No part of this assessment has been copied manually or electronically from any other source (including web sites) except for the information supplied by the WEB222 instructors and / or made available in this assessment for my use. I also declare that no part of this assignment has been distributed to other students.*

Part 1. Website Design (10%)

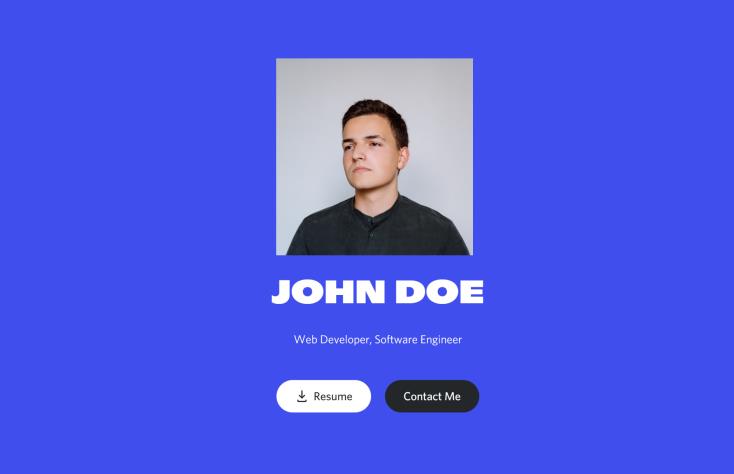
You are asked to create a developer profile website to show off your skills as a developer to possible employers. You are responsible for researching and implementing all aspects of this website on your own and may not include code from other students.

Your website will be expected to have the following structure and elements:

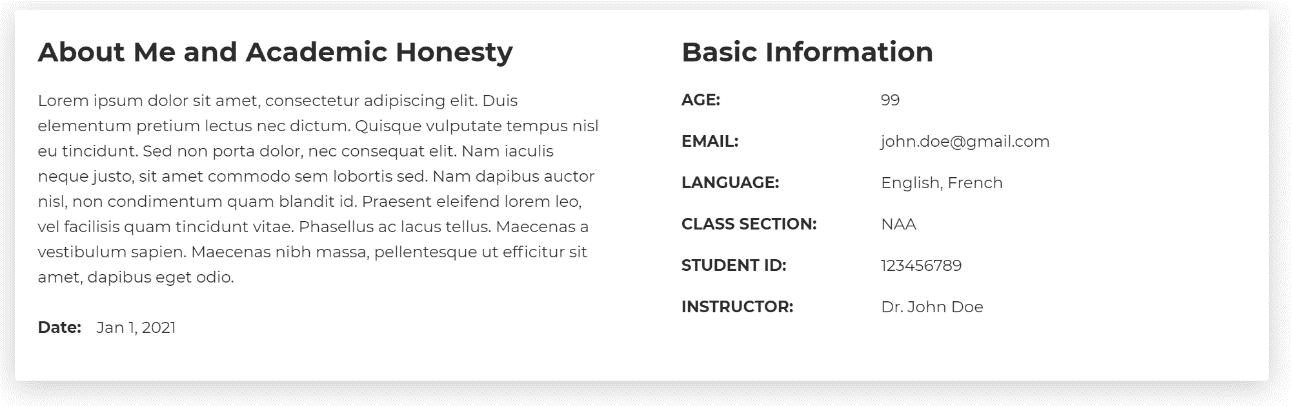
* Make sure you choose the most appropriate, **semantic HTML5 elements** for all of your page’s content (e.g., not everything should be a <div> or <p>). Your page must be valid HTML5.
* All CSS and JavaScript should be put in separate .css or .js files (i.e., not embedded in the HTML), and use proper indentation, formatting, and include appropriate comments.
* All website content should be centered in the viewport, having equal margins on the left and right. The page content should cover 90% of the viewport width (i.e., the width of the browser window). However, the content width should be limited to 1,100 pixels, never growing larger than that.
* Your site should use elements of a **Responsive Design**. That is, it should work equally well, and optimize space and sizes, on both desktop screens (more than 400 px wide) and mobile phones (less than 400 px wide). Research and use CSS Media Queries to define classes and rules that work on a narrow screen (400 pixels or less) vs. your full desktop (greater than 400 pixels). For example, font sizes, margins, layout choices (e.g., removing or moving elements), image sizes, etc. could all be different if viewed on a phone vs. desktop. There are lots of ways to implement a Responsive web site. Research and use a few of these techniques in order to accomplish this requirement.

* Graphical user interface, website

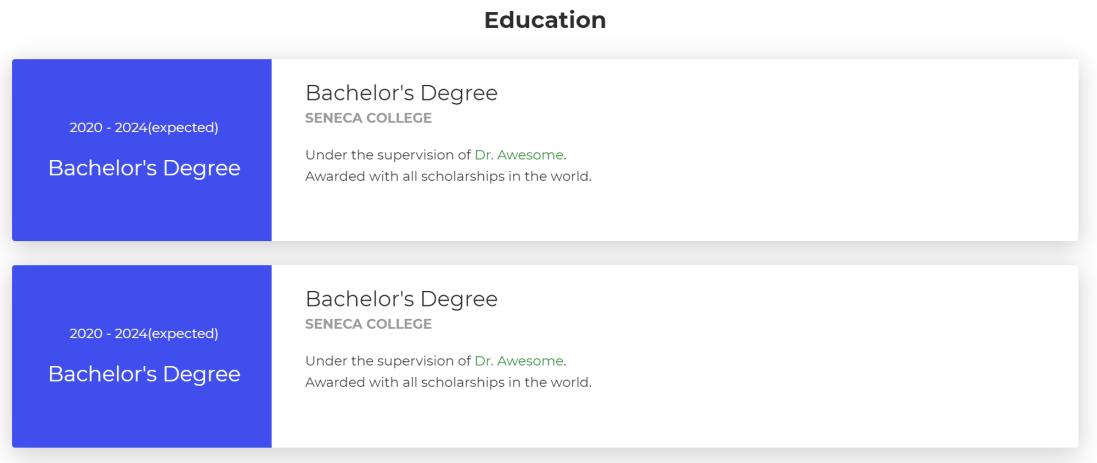
  Description automatically generatedThe website should be divided into sections using different background colours. On the right is an example image of a website that uses this “coloured sections” design approach. Use a palette from palettelist.com to help pick your main and accent colours. Make sure all text uses good contrast for accessibility.
* The top section of your page should have the following structure. NOTE: the exact styling you use is up to you, make it your own, but convey the same general information with the general structure:
  + an image of you (or an image that you like which represents some aspect of your personality). Make sure you have the rights to use any images you include
  + a professional title you hope to get in the future
  + two links
    - the Resume link should download a PDF version of your resume (see the download attribute for an anchor element)
    - the Contact Me link should take you to a Contact Me form on the same page, which will be discussed in more detail below.



* Use at least two fonts from Google Fonts and icons from Font Awesome to improve the website’s aesthetics.
* Create an infographic section that tells us about you. Again, the exact styling is up to you, make it your own, but try to recreate roughly this structure:



* + Your name
  + The current date
  + This course
  + Your section
  + Your professor’s name and use the Seneca honesty statement found at the **bottom of this document**.
* Create an education section that tells us about your education. Use your own styles (i.e., it doesn’t need to look like this) to convey the same information as below, with a similar structure.



* Create a skills section that lists your favourite programming languages and technologies. You should include technologies that you are hoping to learn while you are studying at Seneca, not just those you already know.
* Create a section with a Contact Me HTML Form. Users will use this to fill out their information along with a message to you. Your form should include the following, and all fields should be mandatory unless otherwise noted, and **use the most appropriate <input> type** and **labels**:
* Name
* Email Address
* Street Address
* City
* Province (use a <datalist> for autocomplete of all Canadian provinces and territories)
* Postal Code (must be a valid Canadian Postal Code, with or without a space, use a regex pattern)
* Radio Button Options to specify what this is about: one of "Job Offer", "School", or "Other". If the user selects "Other", dynamically reveal (with JavaScript) another input box to enter the topic. In the case of "Job Offer” or "School" hide this input box
* A large area to enter the message that they want to send (it can show at least 5 lines of text at the same time)
* Choose at least 1 more input control of your own choosing and make it fit thematically with the rest of the form
* The form should submit to https://httpbin.org/post using the POST method, and only allow the user to submit when all data has been entered and there aren't any validation errors. The data posted to httpbin.org should include all data defined in the form above.

Part 2. Static Hosting (5%)

You are asked to research and implement a static hosting solution for your web site, so that it is accessible via a public URL. You do not need to spend any money to achieve this, since many free hosting services exist:

1. Netlify - <https://www.netlify.com/>
2. Vercel - <https://vercel.com/>

Please submit the public URL for your project. All pages, images, etc. must work and not return 404s or other errors.

Part 3. Reflection Video (5%)

You are asked to create a Reflection video that is 5 minutes or less in length and upload it as a private video to YouTube (i.e., only the people you share the URL with will be able to see it).

Your video should include audio of you talking and a screen-capture or screenshots of all the elements you discuss (i.e., you don’t have to be on camera if you aren’t comfortable doing so, it’s up to you). Use the video to give a walk-through of your website and the technologies you used.

In your reflection, please discuss and demo all of the following:

* Give a tour of your website, discussing all areas.
* Discuss the techniques, technologies, and patterns you used?
* Which other web sites did you use as inspiration and why?
* Demonstrate and explain how you made your web site responsive, so that it works well on both phone and desktop browsers. Show how your website adjusts to window sizes less than, and more than, 400 px.
* Explain some of the main features of HTML Forms and Input elements you used to help make sure the user’s input was valid for your forms, and to keep them from submitting mistakes.
* How did you test that your responsive design and web form were both working, and did this testing find any bugs in your implementation? How did you address these?
* What advice would you give to other web developers building forms for their pages based on your own experience? What lessons did you learn?