CS 4WW3/6WW3 – Web Systems and Web Computing

Project, Part 1 – Client-side, static

Worth: 10% (4WW3), 5% (6WW3)

Due: Thursday, October 3rd, 10PM

Specification date: Monday, September 16, 10AM

Synopsis

This part of the project will require you to design a website and implement some of the site's pages using HTML and CSS. In this part of the project, you will only implement static aspects of the client side of the website. In part 2, you will use Javascript to add dynamic aspects to the client side of the site; in part 3, you will implement the full dynamic server-side functionality. This part consists of several core programming tasks that all submissions must include, plus some add-on tasks for those enrolled in CS 6WW3. You must upload a ZIP file containing all of your files to Avenue by the deadline; see below for details on the contents of your ZIP file.

Project Overview

You will design and partially implement a website that allows users to browse and contribute reviews of geographically based object, similar to a restaurant review site like Yelp or Urbanspoon.

The theme of your website, and the "geographically based objects" which users can browse and review, can be anything you choose: Pokémon Go stops, parks, playgrounds, public art, restaurants, book stores, farmer's markets, WiFi hot spots, etc.

In combination with the second and third parts of the project, your site will:

- Allow users to register for accounts
- Allow registered users to provide a textual review and rating (numerical, thumbs up/down, stars, whatever…) for an existing object
- Allow registered users to contribute new objects, consisting of a name, geographic location, and uploaded image
- Allow all users (registered or unregistered) to search for items based on name, rating, or proximity to their current location
- Display results in a table and on a map

Your site will need to be functional for users on mobile devices and non-map-based functionality should work for users with a disability (i.e., using a screen-reader or text-based browser).

Core Programming Tasks

Before you begin coding, you should develop a sitemap to help you figure out the different pages your site will require and how they are related.

Pages to develop

You must create the following pages.

- A search form that allows users to search for objects by name (entered into a text box) or rating (selected from a drop-down box).
    - For part 1 of the project, your form does not have to submit the results anywhere, since you don't have a server side yet.
- A sample results page showing the results of a search (a) on a map, and (b) in a tabular format. From the results table, users should be able to link to a more detailed screen for individual objects.
    - For part 1 of the project, since you will not be implementing the database and dynamic server-side components, your sample results page should include a few sample results hard-coded into your HTML source file.
    - For part 1 of the project, since you will not be using Javascript, you should include a screenshot of a map, rather than actually embed a live map using Javascript.
- A sample individual object page, with details about the object itself, its location on a map, as well as a list of all reviews and ratings that have been entered by users.
    - For part 1 of the project, since you will not be implementing the database and dynamic server-side components, you only need to create 1 sample individual object page, and it should include a few sample reviews and ratings hard-coded into your HTML source file.
    - For part 1 of the project, since you will not be using Javascript, you should include a screenshot of a map, rather than actually embed a live map using Javascript.
- An object submission page, containing a form with which users could submit a new object. The form should have fields for the name of the object, a description of the object, and its location as a pair of latitude-longitude coordinates. The form should also allow users to upload an image for the object.
    - For part 1 of the project, your form does not have to submit the results anywhere, since you don't have a server side yet.
- A user registration page, containing a form in which users are asked to enter the information required to sign up for an account.
    - For part 1 of the project, your form does not have to submit the results anywhere, since you don't have a server side yet.

  Your registration page should include several at least 4 different HTML form elements, including text boxes and check boxes or radio items, at least one of which is an "HTML5" form element such as "type=email" or "type=search" or "type=date".

All of your pages should include a header, navigation menu, and footer. You may also choose have a sidebar if you think it is appropriate. Your navigation menu must contain at least 4 links, although some of them may be broken links because you will not have implemented all of your website in this assignment. Your navigation menu might be part of the header or part of the sidebar.

Design

You should consider your target audience when designing both the functionality and visual style of the web site. When designing your website you should also ensure that:

- Your site has a unified design with a consistent look and feel achieved via appropriate use of colours, fonts, images, etc. and a common page layout (including at least a header, footer, and menu).
- Your site should display well on a desktop browser as well as a mobile browser. You should use "responsive design" techniques in which the same HTML document is style differently for desktop and browser using @media queries in a CSS file.
    - For mobile, your design will be tested at the "iPhone 7" preset in Google Chrome Developer Tools, in both portrait and landscape modes (375x667 or 667x375).
    - For desktop, you can use either a flexible or centred page design that works even for users with screen resolutions as small as 1024 pixels wide.
    - Your site should also display reasonably at any size between 320 pixels wide and 1024 pixels wide, as well as anything larger.
- Your web site follows the WCAG guidelines and works reasonably for users with low-bandwidth connections.

Implementation

Your will need to write HTML and CSS to implement your web pages. Some requirements for your implementation are as follows.

- Your web pages must be strictly developed using the HTML5 standard. You should include the correct document type at the start of each HTML file indicating which version of HTML you are using. Your source must be consistently indented and formatted so that it is easy for humans to read and understand. This implies that you should either write the HTML source manually or at least use a tool that produces clear simple HTML source. In other words, you should not use tools such as Microsoft Word that generate HTML that is hideous to read.
- We will validate your HTML and CSS source code using validators, and you may lose marks if your code does not validate successfully. Note that you should ensure your tags are nested properly so that your pages pass validation.
    - HTML5 validator: http://html5.validator.nu
    - CSS validator: http://jigsaw.w3.org/css-validator/
- CSS must be used to perform all styling. Your pages should be formatted using div elements and positioned using CSS. Do not use tables to layout items on your pages – only use a table if you are displaying tabulated data.
    - You may use <b>, <i>, and <u> tags in your HTML.

- CSS code should be imported from external style sheets with only minimal use of inline style attributes and no use of internal style elements. All CSS code should be well formatted and commented.
- A **larger than normal amount of comments** are required to be included in your HTML and CSS code to demonstrate to your tutor (assignment marker) that you thoroughly understand the meaning of everything that you are using.

Add-on Programming Tasks

CS 4WW3 students can receive up to 2% extra credit for completing one add-on task.

CS 6WW3 students must complete both add-on tasks and will have to ensure these add-ons continue to work through parts 2 and 3 of the project.

Add-on task 1: Meta-data and microdata

In this add-on task, you will add meta-data and microdata to the sample individual object page so that it displays the way you want it to when shared on social media sites and so that search engines can extract semantic geographic and review data from your site.

1. Add metadata fields in the header for Facebook's Open Graph protocol, as well as Twitter Cards. Here are some resources to help you:
    a. Description of the Open Graph protocol: http://ogp.me
    b. Facebook's Open Graph debugger: https://developers.facebook.com/tools/debug/
    c. Description of Twitter Cards: https://developer.twitter.com/en/docs/tweets/optimize-with-cards/guides/getting-started
    d. Twitter's Cards validator: https://cards-dev.twitter.com/validator
    e. Google's structured data validator: https://search.google.com/structured-data/testing-tool
2. Add geographic microdata using the Place microdata schema to your sample individual object page so that advanced web parsers know where the item is geographically.
    a. Details of the Place microdata schema: http://schema.org/Place
    b. Some examples from Google on using Place (and other) microdata: https://developers.google.com/search/docs/data-types/local-business
3. Add microdata to your sample individual object page for each review so that advanced web parsers can recognize reviews and aggregate them into their search results.
    a. Details of the Review microdata schema: http://schema.org/Review
    b. Some examples of reviews in a nice tutorial on microdata: http://diveintohtml5.info/extensibility.html
4. Include metadata in your website so that, if a mobile user saves the page to their "home screen", it will be displayed intelligently.

a. Tutorial on configuring web site for iOS home screen: http://code.tutsplus.com/tutorials/configuring-an-iphone-web-app-with-meta-tags--mobile-2133
b. Tutorial on configuring web site for Android home screen: https://developer.chrome.com/multidevice/android/installtohomescreen
c. Favicon generator for many platforms: https://realfavicongenerator.net

Add-on task 2: HTML5 images and video

In this add-on task, you will use HTML5 to improve the multimedia capabilities of your website.

1. Add the ability to attach a video to an object.
    a. On the object submission page, you should include an additional upload field so the user can upload a video.
    b. On the sample individual object page, you should include a sample video. You should use HTML5 <video> tags to include your image, rather than any special plugin.
    c. Tutorials on using HTML5 video:
        i. http://www.html5rocks.com/en/tutorials/video/basics/
        ii. http://www.w3schools.com/HTML/html5_video.asp
2. Use the HTML5 <picture> and <source> tags to provide images customized to the screen size.
    a. For any graphics (logos, etc.) used in your website design, use the <picture> and <source> tags to provide at least two versions of the logo, for example one standard resolution (1x) and one for high-DPI / retina displays (2x) (so your ZIP should also have at least two versions of the file inside it).
    b. On the sample individual object page, you should include a sample image. You should use the <picture> and <source> tags to again provide at least two versions of the file (so your ZIP should also have at least two versions of the file inside it).
    c. Tutorials on using HTML5 images:
        i. http://www.html5rocks.com/en/tutorials/responsive/picture-element/
        ii. http://webdesign.tutsplus.com/tutorials/quick-tip-how-to-use-html5-picture-for-responsive-images--cms-21015
    d. Include answers to the following questions in your README.txt file:
        i. Describe briefly the different versions of graphics provided in step 2(a); include a sample of HTML code and explain how the different selectors work together.
        ii. List three positive goals that can be achieved using HTML5 <picture> and <source> attributes.
        iii. List one negative about using HTML5 <picture> and <source> attributes. How can this negative be mitigated?

Restrictions

In the first assignment, you cannot use any dynamic client-side components (Javascript) or server-side components (PHP, ASP.NET, AJAX, etc.): you should only use HTML and CSS.

You may not use any client-side technologies that require plugins, such as Java applets, Flash, or Silverlight.

See the FAQ at the end of this document for information about which external frameworks you are allowed to use.

Getting Help

To achieve top marks in this assignment, you will need to supplement what you have learned in lectures and tutorials with details from textbooks or online resources. Fortunately, there are many resources available on the Internet for web development. On Avenue, I have posted links to useful sites with HTML and CSS.

Your tutor will be able to provide guidance on the core programming tasks. You should be prepared to explain what you are trying and why you are stuck, rather than just asking "how do you do this?".

The add-on programming tasks should be seen as more independent tasks: these are functionalities you need to do a little bit of research on how to use, and your tutor will not provide as much guidance in this area.

You are welcome to ask questions of your peers either offline or on Avenue Discussion group. As a last resort, you can email me (doq4@mcmaster.ca) with a question or to request a consultation meeting.

Submission Instructions

SUBMISSION

(1) You must upload a ZIP file containing all your files to Avenue by the deadline; see details on project specification for the contents of your ZIP file.
(2) You should also have your files copied to your web server and set up so that they appear as a live webpage.
(3) The README file in your ZIP file will include a:
  a. link to your live server and
  b. git repository (add doq4@mcmaster.ca as a contributor).
(4) Submit on Avenue.

Your ZIP file should contain the following:

- README.txt: A text file containing:

    - Your name, student number, and whether you are enrolled in CS 4WW3 or 6WW3.
    - If you are CS 4WW3 student doing an add-on for extra credit, say which add-on task you are doing.
    - Answers to the questions in Add-on task 2, if you did it.
- search.html, results_sample.html, individual_sample.html, submission.html, registration.html: HTML files corresponding to the pages listed under "Pages to develop"
- Any additional files: CSS, images, fonts, sample video. You may organize these additional files into subfolders, as you like.

You should include all files required to view your site properly.

- If you use Google Fonts or another external webfonts provider, you do not need to include the fonts in your ZIP file.
- If you use a CSS reset stylesheet, you must include that in your ZIP file.
- If you use any third-party images or videos, you must include them in your ZIP file.
- Share your git repo with doq4@mcmaster.ca.

You should limit your ZIP file to 30 MB.

If you are uploading a sample video, it only needs to be a few seconds long to demonstrate that it works, so you should be able to keep it small.

Marking

Your website should work in any modern mainstream desktop browser: Chrome, Firefox, Internet Explorer / Edge, Opera, or Safari; as well as Safari on iOS and Android Chrome.

We will test your website at least on the most recent version of Chrome on the desktop, but we may also use another modern browser to check compatibility. For the mobile version, we will test your site using the (desktop) Chrome Developer Tools mobile simulator at the "iPhone 7" size as indicated above, but we may also test on an actual mobile phone or emulator to check compatibility.

Your website must follow the WCAG guidelines for accessibility.

Marks will be allocated as follows:

- 20 marks for the core functionality
- 15 marks for the quality of the HTML code
- 15 marks for the quality of the CSS code
- 10 marks for add-on task 1
- 10 marks for add-on task 2

Please see the separate assessment criteria sheet for the detailed marking criteria.

Academic Integrity and Copyright

Except where allowed below, the work you submit for this assignment must be your own.

A good way for beginners to learn about creating web pages is to examine the source of other high-quality pages available on the web and to learn how they do various things. It is unethical to plagiarise such code verbatim, but it is quite OK to examine them, understand how they work and apply the same techniques and approaches in your web site. Different web pages can provide different things; one web site might illustrate the use of a font or colour scheme that you find effective, while another might use CSS to achieve a page layout that you like. Others still might use JavaScript to achieve dynamic effects, such as drop down menus. Other web sites of the same genre might provide some ideas regarding content and functionality.

It is generally acceptable to use small snippets of code you find on the Internet—not more than 5 lines—without attribution. You should not use larger pieces of other people's code in this assignment.

You may not use any external frameworks, libraries, or templates in developing your website, except as indicated in the FAQ below. You are permitted to use a CSS reset stylesheet if you want.

If you want to use other people's images, fonts, or videos in your website, you may do provided you abide by all copyright restrictions. For images, this means you should only use images that you make yourself, or images that are either in the public domain or licensed under terms that allow reuse (e.g., certain Creative Commons licenses). For fonts, this means you should only use fonts that are licensed for web embedding, or web-font services such as Google Fonts. Note that when using properly licensed images in building your own website, it is generally considered polite to locally host the image to avoid placing bandwidth load on the original image provider.

We may use similarity-detecting tools to help identify submissions that share code.

Extensions and Late Assignments

Late assignments will be penalized at 15% per day to a maximum of 7 days. Students with extenuating circumstances should contact the instructor well in advance of the deadline. The McMaster Student Absence Form (MSAF) can be applied to academic work worth less than 25% of the final grade, i.e., a lab report. You must submit the MSAF online at http://www.mcmaster.ca/msaf/. I will automatically grant a 3 calendar day extension upon submission of an MSAF.

Frequently Asked Questions

1. *Can I use these beautiful HTML / CSS templates I found on the web?*
   ~~No~~ **YES!**, you must code your own HTML and CSS. While many organizations do use template systems, since this is a first web development unit it is important for you to learn how to write this code on your own.

2. *Can I use AngularJS / React / YUI / Bootstrap / <insert favourite HTML/CSS/JavaScript framework here>?*
   No. **YES!** Web development frameworks are a very dynamic area, and it is very likely that the frameworks that are popular today will be abandoned or obsolete in just a couple of years. As noted above, it is important for you to learn how to write your own code, and once you have those skills, you will be able to work with appropriate frameworks when the need arises. (If you have a very good reason to use a framework to do something super awesome in a way that doesn't defeat the purpose of the assignment, you can propose it to me and I will consider it.)
3. *Can I use Sass or LESS when I write my CSS code?*
   Yes, but it is recommended that you only do so if you have previous experience with CSS and can explain the problems with CSS that Sass or LESS aim to solve. Be advised that we will only mark your CSS code, so you should ensure that whatever CSS code is generated by Sass or LESS is human-readable and high-quality. (If you did not understand this question or answer, feel free to ignore it.)
4. *Can I use reset.css / normalize.css?*
   You may use a CSS reset stylesheet if you want to have more control over the base formatting of CSS objects. See http://www.cssreset.com for some possibilities.
5. *Can I use web fonts services like Google Fonts?*
   Yes. Please see the note above regarding copyright issues.
6. *Can I use sIFR for displaying fonts?*
   No, because sIFR requires Flash and you may not use Flash.
7. *Can I use an icon font like Font Awesome?*
   Yes.
8. *Which editor should I use?*
   See *Avenue -> Content -> Resources* for a discussion about text editors. Note in particular that you may use Adobe Dreamweaver as a text editor, but you should avoid using its WYSISYG / GUI features as they may not output readable code.