COMP 4610 GUI I

HW 1
Set up, GitHub and First Web Page

PART 1: Setting Up to Take This Course

(1) Complete Your Course Registration, join Piazza and Blackboard

a. Begin by completing your registration for this class using the "Syllabus Certification" form posted on blackboard.
b. Next, join the Piazza class discussion forum so that you can get help from your classmates, our teaching assistant (TA), and me if you need it, and likewise provide help to others where you can. We will use this forum throughout the semester for all course-related communications outside of class, so it is critical that you register on it or you will miss important class announcements.
c. If you have not already done so, please register/Sign-in at Blackboard Site (semester sessions access here: https://lowell.umassonline.net/webapps/login/ and summer sessions access here: https://uml.umassonline.net/). I will be posting most teaching materials here, so it is also critical that you register on it or you will miss important class materials/announcements. Please contact help@uml.edu if you need any help.

What to submit: submit a snapshot/scanned copy of your completed registration for this class.

(2) Reading

Read/watch the following video and article:

- Roadmap to becoming a web developer (read frontend and full stack)
- A Guide to Becoming a Full-Stack Developer in 2017 (old goodie)
- Video: Web Development In 2022 - A Practical Guide (I really liked this video)

What to submit: Write a short summary of what you have learned from this reading. It should be at least more than 200 words.

(3) Installing Editor

You may choose to install one of the following editors for working on your web development during this course. If you are not familiar with any of them, I would suggest starting with Visual Studio Code.

1. Visual Studio Code - Visual Studio Code is a powerful source code editor that comes with a range of tools for JavaScript development. The IDE comes with built-in support for JavaScript, TypeScript, and Node.js. It also has plenty of extensions for other languages (such as C++, C#, Python, and PHP). Developed by Windows, Visual Studio

Code is great for new programmers as it explains everything from HTML tags to syntax and error handling.

2. Brackets - Brackets is a powerful, but lightweight editor that comes with a set of great visual tools and preprocessor supports that allow for easy designing in the browser. The open-source project is free of charge and has a thriving community that is always there to lend a hand. The iDE offers live HTML, CSS, and JavaScript coding and supports programming in Perl, Ruby, Java, Python, and many other languages.
3. Atom - Atom is a fantastic IDE for JavaScript programming. Because it's created by GitHub, it means that there is a thriving community to turn to if you run into any issues. It works with Mac, Windows, and Linux and ships with a package manager for installing new packages. The app is highly customizable, but can also be used well without configuring or customizing anything.

   Note: If you do not plan to use any of the above and have an editor of your preference. Please let me know.

What to submit: Nothing to submit as long as your editor is ready to use.

PART 2: First Web Page

In this part of the assignment you will create your first web page.

Make sure your page have the basic html5 structure discussed in class. It can also be found at: https://www.sitepoint.com/a-minimal-html-document-html5-edition/

For your ease of use, I have also created a template for you to start with here.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Example Website</title>
    <!-- <link rel="stylesheet" href="style.css"> -->
  </head>
  <body>
        <h1>This is the Main Heading</h1>
        <!-- <p>This text might be an introduction to the rest of the page. And
if the page is a
             long one it might be split up into several sub-headings.<p>
        <h2>This is a Sub-Heading</h2>
        <h2>Another Sub-Heading</h2>
        <p>Here you can see another sub-heading.</p> -->
    </body>
    <!-- <script src="script.js"></script> -->
</html>
```

Make sure you properly document your code. Documentation are expected for every section of your code. They should clearly demonstrate what each section are designed for.

Make sure that your file contains the following documentation within <!-- ... --> tags at the top of the file:

a. your full name
b. the date on which you created the file
c. a short description of what your Web page does (this can be very short for this simple page)

This documentation is very important and will be a major part of your grade for this assignment. Most students who do poorly on this assignment simply don't put enough effort into the documentation.

Here is an example of proper documentation at the top of an HTML file.

```
<!--
File: index.html
GUI Assignment:  Creating Your First Web Page
Wenjin Zhou, UMass Lowell Computer Science, wzhou@cs.uml.edu
Copyright (c) 2021 by Wenjin.  All rights reserved.  May be freely copied or
excerpted for educational purposes with credit to the author.
updated by WZ on July 7, 2021 at 9:23 AM
-->
```

Style the text on the page you have chosen as you see fit. Explore the various CSS text properties and their possible values discussed and listed at http://www.w3schools.com/cssref/default.asp#text. If you have a friend who is artistic, you are welcome to ask him or her for help, of course naming that person in your code documentation to give him or her credit for his or her input.

Make sure the content is NON-TRIVIAL, that is, your page contains more than I would expect from a non-major taking 91.113 Introduction to the Internet. It should show that you have put in the effort and the page should minimally include the following HTML contents:

- Some informative reading/interesting site to browse. At least 150 words per page.
- Headings. Those **<h1>** to **<h6>** tags. Put a heading before things that make sense to have a heading.
- Paragraphs.
- Tables. Think about creative ways to use tables. Normally tables are used on websites to keep things looking organized.
- Linked in images. Try to avoid copying image off the web. Take your own photo or sketch something. It doesn't have to look good. If you absolutely prefer online images, please properly cite them on your source code.
- HTML entities. See list here.
- Some simple styling with CSS.

It's crucial for a website to feature meaningful content and not merely appear as a mishmash of random HTML tags.

Code Criteria

- Your CSS code must be in a separate file from your HTML file, in a subdirectory name **css**. This is the industry-standard way to organize websites. You might use inline CSS and/or an internal style sheet during development, but you must move all your CSS to a separate, external file before you submit your assignment. That external file should have an extension of .css.
- Your CSS code must be documented to the same standard we have used for HTML. That is, major sections must be documented to identify what they are styling. You should ***not*** document each individual CSS rule. Assume that that reader of your documentation knows CSS, but that he or she is not familiar with the way you're using it.
- Basic **file organization** is expected. For example, all images should be in their own folder.

Test both your web page and your CSS code using the W3C Validators:

- HTML5 Markup Validation Service: http://validator.w3.org or http://validator.nu
- W3C CSS Validation Service: http://jigsaw.w3.org/css-validator

Your code MUST PASS both validators using the github link from part 3. When it does, add the CSS W3C validation icons to your page (HTML5 does not have one, please reference here).

What to submit: Please submit all related files in a zip.

PART 3: GitHub

This part of the assignment is simple. All you have to do is repost your solution to part 2 on GitHub in such a way that we can view it in a browser. The page must be published using GitHub Pages so that we can get to it via the URL:
http://<*your_user_name*>.github.io/<*your_repository_name*>/<*your_file_name*>

The information you need to do this assignment is in a video created by Curran Kelleher, previous Teaching Assistant for this course. You should watch that video and follow along the steps that it outlines. You will submit your github.io URL using the standard procedure, that is, the course blackboard.

** We highly recommend that you download the GitHub Desktop Software **

1. Watch Curran's Introduction to GitHub YouTube Video at:
   https://www.youtube.com/watch?v=Q6HbQRWAMM4

   As a reference for this video, all the commands you will need to execute for this assignment are listed in the Introduction to GitHub README file at:

https://github.com/curran/screencasts/tree/gh-pages/introToGitHub

Below are links to specific times in the video that explain each step.

- Creating a new repository in GitHub
- Install Git on your system
- Setting up SSH keys
    - you don't need to do this if you use the HTTPS protocol rather than SSH
    - however, with HTTPS you'll need to enter your user name and password each time you push
- Cloning a repository
- Using git status
- Using git add
- Committing files
- Configuring your user name and email with Git
- Pushing changes to GitHub
- GitHub Pages
- Creating the gh-pages branch
- Checking out the gh-pages branch
- Pushing the gh-pages branch to GitHub
- Adding an index.html file
- Updating index.html and adding a JavaScript file (full Git workflow cycle)

2. Follow the instructions in the video to post all the files you created for Part 2 to GitHub Pages.

3. Test your page to ensure that is displays properly using your http://*your_user_name*.github.io URL.

What to submit: Submit your github.io URL as a text file titled "github link"

Submitting Your Assignment for Grading

Please follow these directions carefully. Incorrectly submitted assignments will not be given appropriate credit. To submit this assignment you must go to our course blackboard and submit prior to the deadline.

How You Will Be Graded

This part will be graded on a 30-point system with points awarded as follows. Please note that the lists of features provided below are not meant to be exhaustive. They are merely representative of the types of things we are looking for in each grading category.

Criteria	***Possible Points***
Part 1	

• Completion of Course Registration	1
• Reading - a short summary of what you have learned from this reading. It should be at least more than 200 words.	*3*
Part 2	
Program Integrity / Design • Page exhibits creativity and effort, content is non-trivial, that is, your page contains more than I would expect from a non-major taking 91.113 Introduction to the Internet. Meet minimal requirements with reasonable design. (10 points) • all content is written in proper English with correct spelling (1point) • source code for all files is visible on the web and/or accessible via links on the assignment or index page (1 point) • all CSS resides in a separate file with an extension of .css in a subdirectory named css (2 points) • page validates using the W3C Markup Validation Service (2 points) • page validates using the W3C CSS Validation Service (2 points)	*18*
Source Code Documentation and Formatting • top of the file documentation • code is logically organized, any sources used are cited in comments embedded within code, all files contain adequate explanatory documentation, all documentation is meaningful and does not merely echo code • all files have adequate white space for readability, all files are properly indented and formatted	*3*
Part 3	
Page Displays from GitHub Pages page displays properly and without error from the github.io server	*5*