

# CS 416

## Homework 3

### 1. Objectives

The purpose of this assignment is to get you familiar with creating dynamic web pages and improve your skills in the areas below:

- Creating a web page using HTML, and CSS
- Using Bootstrap
- Making a web page that is responsive for various screen sizes
- Making a web page dynamic using JavaScript and jQuery

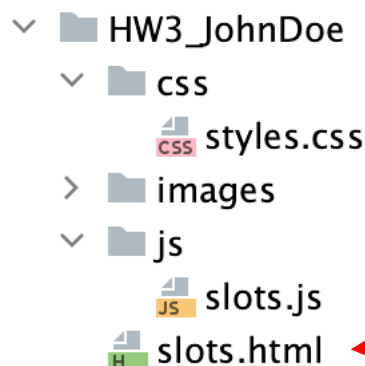
### 2. What to do?

In this assignment, you are given several screenshots and a description of a particular web page to replicate. Using **Bootstrap** (you must use the latest Bootstrap version, **5.2**), you will recreate this web page exactly as shown in the provided screenshots and video below. Using **JavaScript** and **jQuery** (you must use the latest jQuery version, **3.6**), you will also implement the required functionality.

In particular, you will create a basic slot machine that will have three slots displaying three images. There will be a **Spin** button to play this game. When the **Spin** button is clicked, it will randomly select a new set of images (no spin animation is required, each image just needs to change to a new random image). In addition to creating the basic slot machine, you will add code to keep track of the money the person has.

#### 2.1 Getting Started

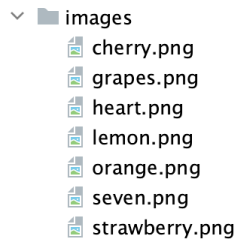
Create 3 directories/folders for CSS, images and JavaScript and name them as `css`, `images`, and `js` as shown below. For the folder name, replace your name with John Doe, unless your name is John Doe.



Note that the `html` file is **not** located in the `js` folder. Make sure to put your `html` file as shown here.

Subsequently, create the following files.

1. An HTML file named **slots.html** that will contain the HTML markup of your page. It may be a good idea to get the Bootstrap started template for this page.
2. A CSS file named **styles.css** that contains the styles for this page. Similar to HW-2, you will be utilizing Bootstrap CSS classes rather than implementing them yourself, so **styles.css** should contain very few CSS rules. This file should be located in the **css** directory you created in the first step.
3. A JavaScript file named **slots.js** that will contain the JavaScript code. This file should be located in the **js** directory you created in the first step.
4. Download the images from Blackboard and put them in the **images** folder you created in the first step so that **images** directory should look like below.



Add images using a relative path as shown below

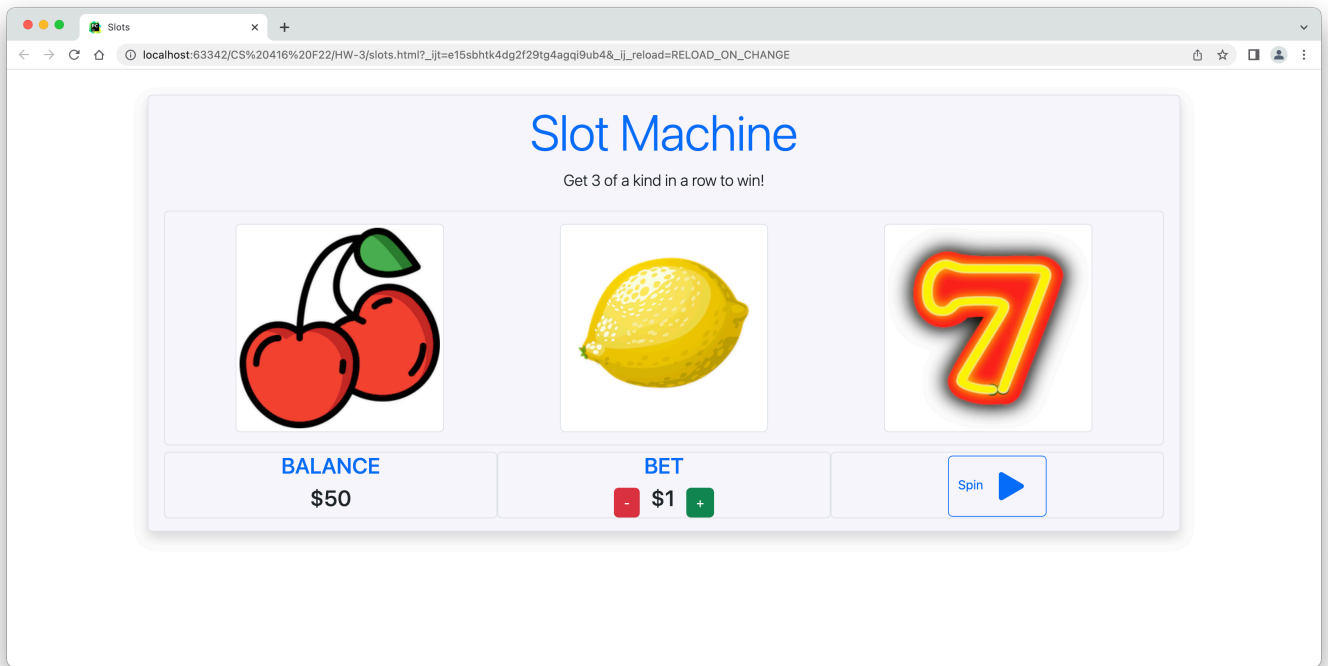
```

```

## 2.2 Overall appearance

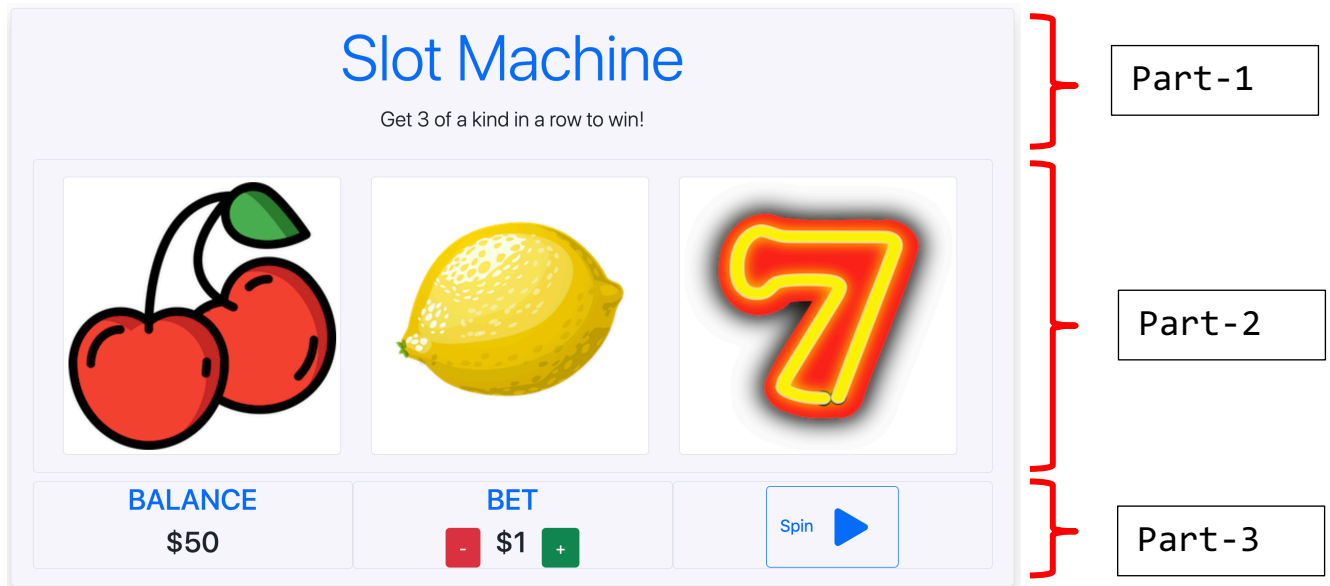
The image below shows a screenshot of the website you need to recreate in this homework. A full-size screenshot of the entire page is attached with this assignment on Blackboard.

Additionally, you can **watch the video** on Blackboard showing the look and behavior of this page (see the video on Blackboard).



## 2.3 Appearance Details

As shown in the image below, there is a container that consists of three rows (i.e., sections or parts). The details of these sections are explained below.



### Container

- A padding should be given to body so that the page maintains some margins from top, left and right.
- The container should have a background color as shown above (bg-light class from Bootstrap can be used).
- There is a rounded border and shadow as shown above (border, rounded and shadow classes from Bootstrap can be used).
  - See the link for more information:
    - <https://getbootstrap.com/docs/5.2/utilities/borders/#additive>
    - <https://getbootstrap.com/docs/5.2/utilities/borders/#radius>
- There should be a padding within this container from top and bottom (py-2 class from Bootstrap can be used).
- For the content in this container, you must use the Bootstrap Grid System. In particular, the container should have rows of content, and those rows should have one column for the part-1 part, and three columns for each of the part-2 and the part-3, respectively.
  - See the link for more information:
    - <https://getbootstrap.com/docs/5.2/layout/grid/>

### Part-1

- This section should have a row containing a column such as col-lg-6.
- Using the Bootstrap Grid System, this section should justify the content to the center.

- For more information, you can see the link below:
  - <https://getbootstrap.com/docs/5.2/layout/columns/#alignment>
- This section should contain one heading and a paragraph as follows:
  - The heading should be large and contain a text “Slot Machine”, and it should be blueish color
    - Hint: one of the Bootstrap’s display classes (e.g., display-3) can be used for large heading, and text-primary can be used for the blue color
      - <https://getbootstrap.com/docs/5.2/content/typography/#display-headings>
  - The paragraph should be located under the heading and include the following text “**Get 3 of a kind in a row to win!**”.
    - You can use lead class to make this paragraph stand out.
    - The content in this heading will also be updated dynamically in your JavaScript code. So, it may be a good idea to give an id for this element so that it would be easier to access and change its content in your JavaScript code.

## Part-2

- Using the Bootstrap Grid System, this section should have a row containing three columns where each contains an image. The content in this section should be centered as well.
- There is a rounded-border around the row as shown above (border and rounded classes can be used for the row).
- For this section, there is a small margin (m-2 class can be used for the row), and a padding (p-3 class can be used for the row).
- The images must scale such that they are larger on a wide screen, and shrink to be narrow enough to fit small screen.
  - **Hint:** You can use .img-fluid class to implement this. See the link below for more information.
    - <https://getbootstrap.com/docs/5.2/content/images/#responsive-images>
- There should be a rounded 1px border for each of the three images.
- Hint: img-thumbnail class can be used for the rounded border.
  - <https://getbootstrap.com/docs/5.2/content/images/#image-thumbnails>

## Part-3

- Using the Bootstrap Grid System, this section should have a row containing three columns.
- All the content in this section should be centered.
- For this section, there is a small margin (m-2 class can be used for the row).
- There should be a rounded-border around each of three columns.
- When the screen size is smaller than the small screen size (i.e., -sm), these columns must stack on top of each other rather than going side by side. The details of each column are described below:
  - **Sub-section-1 (BALANCE):**
    - There should be a heading (e.g., h3) for **BALANCE** with a bluish color

- Hint: text-primary class from Bootstrap can be used
- There should be a tag (e.g., h3) for dolor amount. You may also consider using a span tag inside this tag for just the \$ amount portion, which can make it easier when changing the content via JavaScript later.

```
<h3> $ <span id="money">100</span> </h3>
```

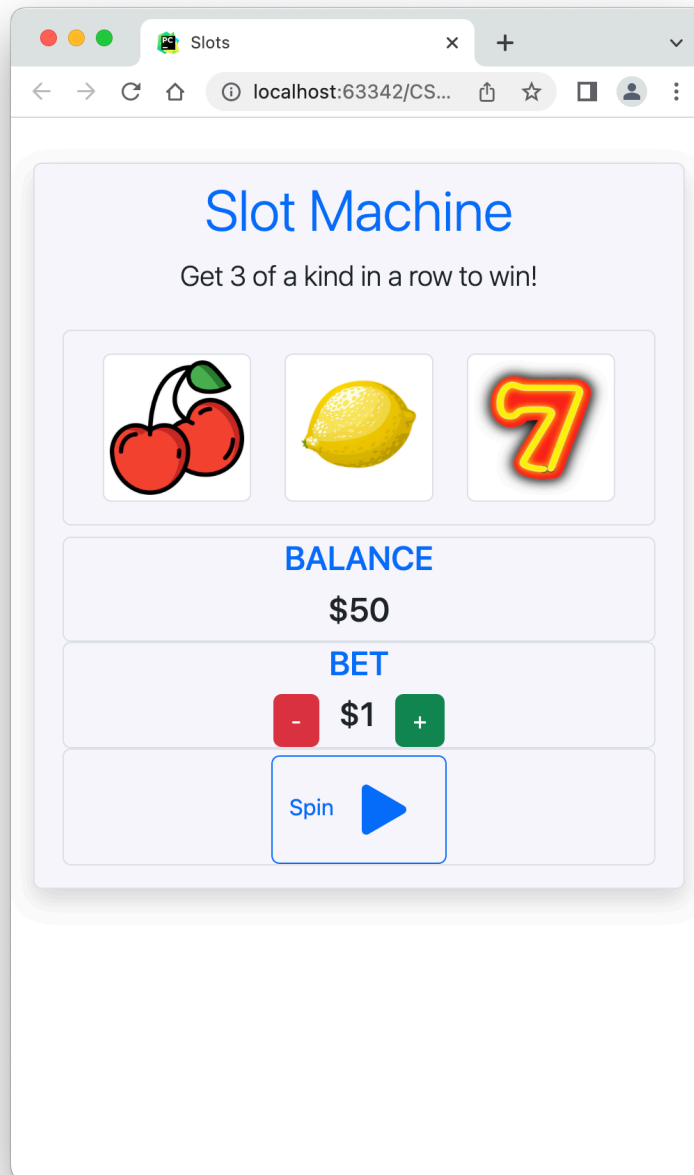
○ **Sub-section-2 (BET):**

- There should be a heading for **BET** amount with a bluish color (similar to Balance)
- There should be two buttons to increment (green) and decrement (red) the bet amount.
  - Hint: btn btn-success and btn btn-danger classes can be used to change their colors accordingly.
- You should use h3 heading tag for bet amount. The two buttons and bet amount should appear side by side.
  - Hint-1: You may observe that buttons and bet amount do not appear side by side, which is an expected behavior of block-level elements (e.g., h3). So, you can consider changing this style property so that they appear side by side.
  - Hint-2: You may also consider using a span tag inside this h3 tag for just the \$ amount portion, which can make it easier when changing the content via JavaScript later.

○ **Sub-section-3 (SPIN):**

- There should be a button containing an icon from Bootstrap.
  - Hint: You can use the following icon and change the width and height values to match closely with the sample implementation (or copy and paste the code into an appropriate tag)
    - <https://icons.getbootstrap.com/icons/play-fill/>
- The icon should have a width and height of 64.
- For this button, there should be a small margin from the top.

- Overall, your page should be responsive and viewable from a mobile device such as shown below.



- Please note that though the specifications above can be helpful, they may not cover all the details. Thus, you should design your page in accordance to the given full-size screenshots and the video on Blackboard.

## 2.4 Overall Behavior

- There should be a spin button to play the game. When the user clicks on the spin button, all three slots' images should be randomly selected from at least 7 different images (such as those on Blackboard with the assignment) and displayed on the page.
  - Hint: The code below can be used to generate a random number in a certain range  $[0, \text{myRange}]$ . You can then use this randomly generated number as an index value to choose an image from an array that stores image names (e.g., `lemon.png`). You can create a function and put this code in that function so that you can call this function as many times as you need it without redundancy.
    - `Math.floor( myRange * Math.random() );`
- If the 3 images are the same (i.e., hit the jackpot), the user wins **15\*betAmount** and the following text should be shown under the blue "slot machine" heading.
  - "Congratulations! You won!"
  - The text should also be red and blink for a short amount of time.
    - Hint: `fadeTo()` function from jQuery can be used to achieve this. You can also chain multiple `fadeTo(speed, opacity)` functions for the element selected by jQuery selectors in order to create this effect such as below:
      - `.fadeTo(100, 0.1).fadeTo(200, 1.0);`
      - See more information at the link below:
        - <https://api.jquery.com/fadeTo/>
      - If you get the following error when you use `fadeTo()` function, this may be because of using jQuery slim version (some functions are not included in this version) which comes with the Bootstrap's starter template. To fix it, you need to insert a regular jQuery version that you can get from jQuery website.  
`$(...).text(...).fadeTo is not a function`

### jQuery 3.x

- jQuery Core 3.5.1 - [uncompressed](#), [minified](#), [slim](#), [slim minified](#)

# Slot Machine

Congratulations! You won!

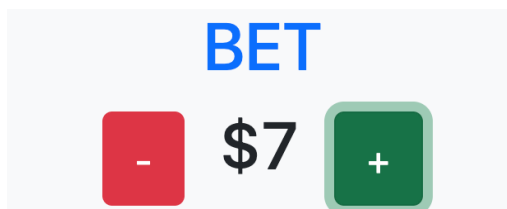
- If the 3 images are not the same, the user loses **betAmount**. The following text should be shown under the blue "slot machine" text.

- “You lost, spin again.”
- The text should also be red and blink for a short amount of time (see the video on Blackboard).

# Slot Machine

You lost, spin again.

- The HTML tag under Balance should show the money the user currently has. When the user first loads the page, the money should be \$50 by default.
- When the user first loads the page, the bet amount should be \$1 by default. The user should be able to decide what the bet amount will be before clicking the spin button. Towards that, there are two buttons to increment and decrement bet amount. Clicking the button with ( - ) sign should decrement the bet amount by 1 dollar. Clicking the button with ( + ) sign should increment the bet amount by 1 dollar.



- Your program needs to get the bet amount from the HTML tag under Bet section (i.e., part-2), which could be any number between 1 and the current balance. So, the bet amount cannot be a negative number or a value that is higher than the user's current balance.
- After each spin, the current balance should be updated (as described above).
- If the user has no money left, the following text “**You lost all your money!**” should be shown rather than allowing him/her to spin.
  - Make sure your program **does not** show “you lost, spin again” message instead of the above message for this situation (i.e., the user has no money left).

# Slot Machine

You lost all your money!

- If the user tries to bet an amount that is more than he/she already has, the following message should be shown
  - “**Invalid bet amount, you do not have enough money to bet X\$**”.
  - The text should also be red and blink for a short amount of time.



- For example, if the user has 7\$, but he/she tries to bet 10\$, the message should state *"Invalid bet amount, you do not have enough money to bet 10\$"*. If you are wondering how this situation is possible, as the plus button prevents exceeding the current balance. This can happen because the bet has not changed if the user loses despite having enough money for that bet amount in the previous round.

## Slot Machine

Invalid bet amount, you do not have enough money to bet \$7

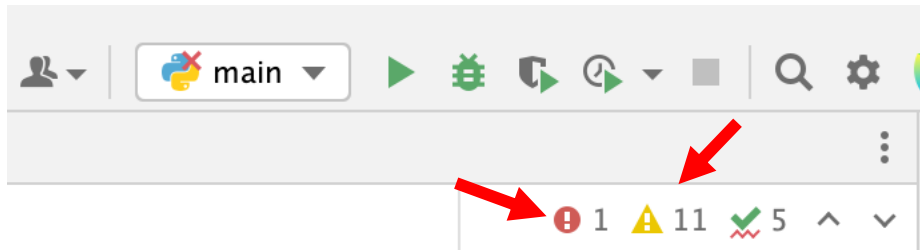
### Extra Credit (OPTINAL)

Up to **10** bonus points will be given if the images in slots are animated as shown in the extra-credit video on Blackboard. In particular, images in each slot changes quickly, and each slot rotates through all the images in your `images` folder. The 3 images begin to change simultaneously and the first image is determined first, then the second image and finally the third image.

**Hint:** You can generate 3 random numbers for each slot in an increasing order and use those numbers for setting an interval value (e.g., using `setInterval()` function) to keep changing the images. For example, if you use `setInterval(myFunction, 100)`; with the following random numbers 30, 43, 58, the animation code in `myFunction` takes 0.3 sec, 0.43 sec, and 0.58 sec for the first, second and third slot, respectively. In `myFunction`, you can change an image to another image by altering `src` attribute.

## 2.5 Validate Your Code

While writing your code in PyCharm, PyCharm can help you identify the problems in your code. On the upper right corner, you can see an indicator something like below. Once you click on the warning icon or error icon, it can help you identify the issues, if any.



Furthermore, after writing your code, you should validate your code to receive full credit. To do so, you can check your HTML code using [“Nu Html Checker”](#)

- [https://validator.w3.org/#validate\\_by\\_input](https://validator.w3.org/#validate_by_input)

After opening the link above, you can copy and paste your HTML code, and then click “Check” button. The page will tell you if the code is valid. If it is valid, you can go ahead and submit your code to Blackboard. If not, the page will tell you specific issue(s), and you should fix them before uploading your code to Blackboard in order to avoid losing points.

A screenshot of the W3C Markup Validation Service website. The header shows the W3C logo and the text 'Markup Validation Service' with a subtitle 'Check the markup (HTML, XHTML, ...) of Web documents'. Below the header are three tabs: 'Validate by URI', 'Validate by File Upload', and 'Validate by Direct Input'. The 'Validate by Direct Input' tab is selected. Below the tabs is a section titled 'Validate by direct input' with a text area for entering markup. The text area contains the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Examples with CSS</title>
  <link rel="stylesheet" href="exercise-2-styles.css">
</head>
<body>
  <h1 class="my-heading">CS 416 Web Programming</h1>
  <h2>Section 01</h2>
  <h4>Fall 2022</h4>
</body>
```

Below the text area is a link that says 'More Options'. At the bottom right of the form is a 'Check' button, which is highlighted with a red arrow.



This indicated that you are good to go. Here is an example message.



This indicates that you need to fix the issues. Here is an example message (intentionally removed the closing tag of h4 to demonstrate an example issue):

**Document checking completed. No errors or warnings to show.**

### Source

1. <!DOCTYPE html>↵
2. <html lang="en">↵
3. <head>↵
4.     <meta charset="UTF-8">↵
5.     <title>Examples with CSS</title>↵

Check by  ☒ CSS

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Examples with CSS</title>
  <link rel="stylesheet" href="exercise-2-styles.css">
</head>
<body>

<h1 class="my-heading">CS 416 Web Programming</h1>
<h2>Section 01</h2>
<h4>Fall 2022

<a href="#">Click here!</a>
```

Check

Use the Message Filtering button below to hide/show particular messages,

## Message Filtering



1. **Error** Heading cannot be a child of another heading.

From line 18, column 1; to line 18, column 23

```
re!</a><h1 class="my-heading">Lorem
```

## Assignment Checklist:

The following is a checklist that you should refer to before submitting your assignment. This will prevent you making mistakes that you would otherwise lose points.

- Make sure that the web page you submit includes the html, head, title, body, and **doctype** tags.
- Make sure that your HTML file is well formed, meaning all the html, head, title and body tags are opened must be closed. For example, if you have `<body>` tag, you must have `</body>`.
  - You should check your HTML code using "[Nu Html Checker](#)"
    - [https://validator.w3.org/#validate by input](https://validator.w3.org/#validate_by_input)
- Try to avoid redundancy in your CSS as much as possible. You should not have too many redundant styles, if there are ways to use inheritance or special selectors to concisely define style rules. Points will be deducted if styles used for this page are too redundant.
- File names (e.g., html file, images, ccs file) should be all lower-case with no space to separate words, instead use `-` to separate words (e.g., recipe-styles.css).
- It is best practice to use relative file paths when describing location of image files. You can create an image directory, put images in this directory, and use a relative path as follows:
  - ``  Good
  - ``  Avoid
  - Using relative file path, as shown on the first example above, allows the link to work on your computer as well as on other computers (e.g., public domain).
- Make sure you follow the best HTML practices, which you can find under Lecture-2 on Blackboard.
- Make sure your JavaScript code is not repetitive.
  - **DRY (Don't Repeat Yourself):** If you ever find yourself copying and pasting the same code again and again, STOP and think long and hard – you're probably doing something wrong
    - **Source:** <https://medium.com/the-andela-way/how-to-start-writing-high-quality-code-at-any-point-of-your-programming-journey-d434cb0ba8ca>

## What to submit?

Submit a **single zip file** containing the following on Blackboard (please name this zip file with your first name and last name such as **HW3\_John Smith.zip**):

- The HTML file (**slots.html**), CSS file (**styles.css**) and JavaScript file (**slots.js**) **along with the image files**.
- 3 screenshots (in PNG, JPG, JPEG or PDF format) showing your web page in your web browser for each of 3 different screen sizes (desktop size display, medium size display, and small size display). You can use **Firefox, Opera, Chrome** or similar to take a screenshot of full page (see the tutorial under "Tutorial" section on Blackboard)).
- Fill out the attached self-evaluation form and submit it along with the rest of the files.

If Blackboard does not allow you to submit these files, submit a zip file that contains your files (if you do not know how to zip your files, please see the tutorial on Blackboard).

If for some reason you are still not able to submit your files, email your files to me before the deadline. Please attach your files to this email.

## Getting help

Start your assignment early. If you need help, send an email to me or make an online appointment with me during my office hours.