[**Overview**] This assignment entails the development of a user interface (UI) that employs an interactive game. The objective is to demonstrate your understanding of HTML, CSS, and JavaScript (jQuery) concepts. You will create an engaging game that includes a user-friendly interface, captivates the player's attention, and promptly responds to their actions. You have the option to select one of the following game concepts or propose your own.

Game Concepts:

Create an interactive game by utilizing jQuery, CSS and HTML. This to be developed game should include a user-friendly interface, captivate the player's attention, and promptly respond to their actions. You have the option to select one of the following game concepts or propose your own.

- A **memory card game** that displays a grid of face-down cards. Players have the ability to disclose two cards simultaneously by selecting them with a click.
- A **word matching game** in which participants are presented with a grid of tiles facing downwards, with each tile holding a word or phrase. Players alternate between revealing two tiles simultaneously, aiming to find matching pairs of words. Possible categories for the game could encompass animals, food, or famous quotes.
- A **numbers game** where participants have to match pairs of numbers or mathematical equations. Each tile within a grid may contain an equation, and players are required to switch over tiles in order to locate matching pairs. The game may incorporate varying degrees of complexity, encompassing addition, subtraction, multiplication, or division.
- An **image matching game** where players have to match pairs of photos or icons. Every individual tile within the grid has the potential to display either a picture or an emblem. The objective for players is to uncover tiles by flipping them over in order to locate pairs that match.
- A **pattern matching game** in which players match pairs of patterns or forms. Every individual tile within the grid possesses a distinct pattern or shape, and participants are required to flip the tiles in order to locate matching pairings.
- A **letter matching game** where players need to match pairs of letters or words. The grid contains tiles, each of which can have either a letter or a word. Players have to flip over the tiles to locate matching pairs.
- An **emoji matching game** where players have to match pairs of emojis or emoticons. Each tile in the layout has a distinct emoji, and players need to flip them over to locate matching pairs.
- A **trivia quiz game** in which players respond to multiple-choice questions on a range of subjects including history, literature, science, news, and sports. Participants accumulate points for accurate responses and engage in competition with one another or against a time limit.
- A **puzzle game** in which players solve puzzles by arranging fragments or tiles to create a coherent picture or pattern.
- A **word search game** where players locate hidden words in a grid of letters. Players have the ability to search for words in a horizontal, vertical, diagonal, or backward direction within the grid.
- A **rhythm game** in which players synchronize their actions (e.g., mouse clicks) with the music by tapping or hitting buttons in sync with the beat.

When designing the assignment, ensure that the UI incorporates the following characteristics:

- Development:
  - Employ only JavaScript (jQuery), CSS and HTML to complete the requirements of this assignment.
  - Assignment should not utilize any backend program and/or web APIs.
  - Assignment should consist of three files: (a) HTML file that contains HTML structure (assign2.html), (b) JavaScript file that contains the JavaScript logic (assign2.js), and (c) a CSS styling (assign2.css).
  - Ensure your code follows modular code design and contains adequate documentation (e.g., structure your code logically and provide comments when necessary).
- User Interface (UI):
  - Provide clear instructions and game rules.
  - Ensure the interface is intuitive and user-friendly
- Interactive Gameplay and Responsive Design
  - UI should react to user activities like as clicking, tapping, and typing.
  - Ensure that the game responsive to different screen sizes and consider accessibility in your design (e.g., keyboard navigability, screen reader compatibility, and using accessible color contrasts).

    - Provide immediate feedback on user actions.
- Visual Feedback and Progress Indicators:
    - Display visual feedback and progress indicators. For example, this could include animations, score updates, and timer display, which can enhance the user experience.
    - UI provides a timer and a real-time score after every event by the user (e.g., flipping/clicking a card)
    - Use visual cues to enhance the gaming experience.
- Error Handling and User Experience:
    - Provide feedback to the user when something goes wrong (e.g., invalid inputs or game reset).
    - UI provides a way to restart the game once completed.
- CSS Style Sheet:
    - The UI is required to operate through a web browser and should include the use of CSS frameworks such as Bootstrap or Bootswatch or equivalent. The UI should be similar to that of Module 2 GitHub example.

Examples and Templates:

To assist you in getting started, a sample example or template for a simple card matching game is available on Canvas. The sample template is meant to offer a basic structure in order to get started with developing the assignment. You are not required to use the template for completing the assignment.

A Game Concept with Example for a Memory Card Game:

- Description: Create a game that presents a grid of face-down cards. The goal is for players to find matching pairs by flipping over two cards at a time.
- Example: Imagine a 4x4 grid (16 cards) with 8 pairs of matching cards. When the game starts, all cards are face down. The player clicks on one card to reveal it, and then clicks on a second card. If the two cards match, they stay face up; if they do not match, they are flipped back face down. The game continues until all pairs are found.
- Game Rules:
    - Players can flip two cards at a time.
    - If the cards match, they remain face up.
    - If the cards do not match, they are flipped back face down after a short delay.
    - The game ends when all pairs are found.
- Objective: Match all pairs in the fewest number of moves possible.
- Features:
    - Timer to track how long it takes to complete the game.
    - Move counter to track the number of attempts.
    - Restart button to reset the game.

Use Free Image Resources

You can use websites that offer free images. Some popular options include:

- **Unsplash**: A website with a large collection of free high-resolution photos.
- **Pexels**: Offers free stock photos and videos.
- **Pixabay**: Free images and videos you can use anywhere.

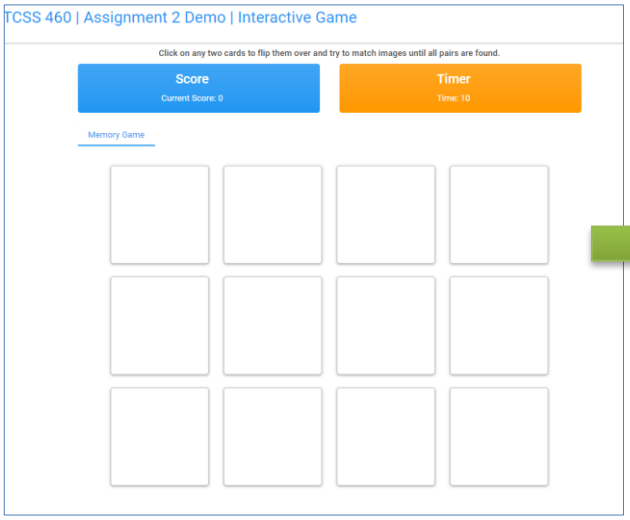
Example Steps Using Unsplash

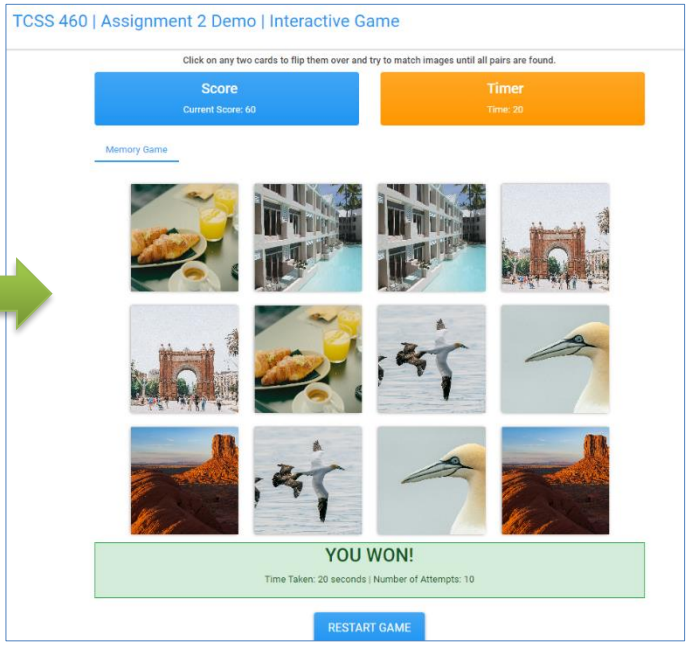
Assume that the UI will present **six** different images, which translates into **twelve** cards for a memory game.

1. **Visit Unsplash**: Go to unsplash.com.
2. **Search for Images**: Use the search bar to find images related to your game theme (e.g., "nature", "animals", "abstract").
3. **Download Images**: Select six different images and download them.
4. **Rename Images**: Rename the downloaded images to img1.jpg, img2.jpg, img3.jpg, img4.jpg, img5.jpg, and img6.jpg.
    - Note that the extension of the file may vary.

[Sample Screenshots of a Memory Game Example]

- left: when game first loaded into browser
- right: when game competes]





Tips and Hints for Coding and Implementation

- **Creating CSS Classes:**
  - Specify CSS classes to apply styling to elements such as cards, containers, and flipped states.
- **Adding and Removing CSS Classes with jQuery:**
  - Utilize jQuery methods to dynamically incorporate and eliminate CSS classes in response to user activities..
    - **addClass**(): Adds one or more classes to the selected elements.
    - **removeClass**(): Removes one or more classes from the selected elements.
    - Examples:

```
$('.card').on('click', function() {
    $(this).addClass('flipped');
});
```

```
firstCard.removeClass('flipped');
```

```
.card {
    width: 150px;
    height: 150px;
    margin: 10px;
    display: inline-block;
    vertical-align: top;
    cursor: pointer;
    text-align: center;
    font-size: 24px;
    border: 1px solid #ccc;
    background-color: #fff;
    transition: background-color 0.3s;
    position: relative;
    line-height: 150px;
}

.flipped {
    background-color: #f9f9f9;
    color: #000;
}
```

- **Game Logic Tips**
  - Shuffling the Cards:
    - Randomly shuffle the array of images to guarantee that the game is distinct on each occasion.
    - Utilize JavaScript's sort() method with a random comparator.
  - Handling Click Events:
    - Utilize the on method in jQuery to manage click events on the cards.
  - Matching Logic:
    - Store the references to the flipped cards and compare their data properties to determine if they match.
- **Debugging and Testing**
  - Console Logging:
    - Utilize the console.log() function to display variable values and the sequence of execution. This aids in comprehending the potential sources of error.
  - Incremental Development:
    - Ensure that you test your code gradually and in small increments. Prior to proceeding to the subsequent phase, it is vital to thoroughly test the minor portion that has been implemented. This facilitates the process of identifying and resolving problems.
  - Error Handling:
    - Anticipate possible faults and manage them in a smooth and skillful manner. For instance, make sure that only cards that have not been flipped can be selected by clicking on them.

Use of AI Tools and AI Code Generation Tools:

Although artificial intelligence (AI) tools can provide significant assistance in consulting, offering advice, and resolving issues, it is crucial to utilize them in a responsible manner. As students, it is imperative to prioritize learning and understanding the concepts rather than depending on AI tools to automatically produce code.

Please comply with the following guidelines when using AI tools:

- **Advice & Recommendations**: Utilize artificial intelligence tools to clarify uncertainties, obtain comprehensive explanations, and grasp optimal strategies in coding. These tools can facilitate the discovery of new ideas and enhance your coding skills.
- **Troubleshooting**: When you come across problems or errors in your code, AI tools might be valuable for identifying and resolving difficulties. Nevertheless, try to comprehend the fundamental reason behind the problem and the mechanisms of the solution.
- **Avoid Code Generation**: Avoid utilizing AI tools to generate full code segments or accomplish entire assignments. Assignments serve the objective of evaluating your understanding and proficiency in applying the principles presented throughout the course.
- **Using AI Tools**: In case AI tools were employed in anyway, please mention this in your video demonstration.
- **Learning Focus**: It is crucial to prioritize the acquisition and comprehension of coding skills. Engaging in manual coding will improve your problem-solving abilities and foster a more profound comprehension of the subject matter.

Submission Instructions:

Important Notes:

- Ensure that your code adheres to best practices, which include appropriate implementation of modularity through component lifecycle functions, and concepts of clean code.
- Utilize adequate comments and documentation to clarify your code and the decisions made during implementation.

What to Submit:

Compress all of the files for the assignment 2 into zip format. Call this file **userid_assign2.zip** *(where userid is your UW User ID).*

The compressed file should contain:

a) a folder called **code** which contains the following:

1. HTML File: **assign2.html** – This file contains the structure of your interactive game.
2. CSS File: **assign2.css** – This file contains the styles for your interactive game.
3. JavaScript File: **assign2.js** – This file contains the logic for your interactive game.
4. Media Files: Include any images or videos used in the assignment within the zip file.

b) A recorded video demonstration (3-5 minutes) file. Call this file **assign2.mp4** (or equivalent extension). The recorded video file should demonstrate a working assignment 2 via a browser. That is, the video should demonstrate:

- The working of the interactive game developed.
- A walkthrough and an explanation of the major coding implementation methodologies used in your assignment.
- A demonstration of a complete game with scores and time tracking.
- The video demonstration should cover key features like game start, gameplay, scoring, and game completion.

Note: Students are encouraged students to explicitly mention in their video demonstration if and how they used any **AI tools** during their development process.

Note: To preserve space on Canvas, consider compressing your video file. You can use online tools such as HandBrake, Clipchamp, or any other video compression tool to reduce the file size. Compressing your video file will help saving time during the uploading of your submission on Canvas.

Grading Rubric (20 points)

- [6 marks] **Functionality**: How well does the game works with no bugs and that all features implemented correctly.
- [3 marks] **Code quality**: How well the code is well-organized, clean, and follows best practices. Comments are used effectively.
- [3 marks] **CSS Styling**: How well styling is consistent, visually appealing, and enhances user experience.
- [2 marks] **Creativity and Innovation**: Game design shows creativity and innovation.
- [2 marks] **Visual Feedback and Progress Indicator**: UI provides visual feedback and progress indicators that enhance user experience.
- [4 marks] **Video Demonstration**: Video is clear, concise, and thoroughly explains the game and code implementation. Further, video demonstrates game with scores and time tracking.