### **Challenge #4: JavaScript - Memory Card Matching Game (Part 1) (40 minutes)**

You will create a memory card matching game where users flip cards to find matching pairs. The game should keep track of flipped cards and notify the player when they find a match or fail to do so. The game ends when all pairs are matched. This challenge will test your logical reasoning and ability to manage complex game states using DOM manipulation and event handling.

In this exercise, we will complete **Part 1** of this game and learn the concepts of:

* Array operations
* Retrieval of user inputs (e.g. multi-select dropdown menu)
* Template literals (See [W3Schools tutorial](https://www.w3schools.com/js/js_string_templates.asp) for more details)

#### **Instructions**

1. **Skeleton Files**:
   * Go to **eLearn → Content → Week 4 → In Class → (unzip) Week4\_InClass → Challenge4**
   * You will find **3 skeleton files**:
     + home.html: The main HTML file.
     + styles.css: The CSS file used to style the main page.
     + script.js: The JavaScript file you’ll use to add interactivity to the main page.
2. Don’t forget to use **Chrome Browser’s InCognito mode**.
3. Please remember to **hard-reload/refresh** (the main HTML page in the web browser upon making changes to the HTML file (if any), the CSS file (if any), and the JavaScript file (you will make changes here for sure).
   * **Windows**: CTRL + SHIFT + R
   * **Mac**: command + SHIFT +R
4. Go to home.html and inspect the code. Understand what elements are of importance in terms of DOM manipulation you will perform later on using JavaScript.
5. Go to script.js and complete the function generate\_board() as follows:
   * **Task 1**
     + In the HTML page, notice how the **select (dropdown menu)** allows for the selection of **multiple options**.
     + Add more code in the function so that the code can retrieve the user-selected option(s).
     + Display the options in **Developer Tools → Console**.
   * **Task 2**
     + We have a total of **24 customized cards** (image files) available for your use in this game. See in the sub-folder cards/. You will see such image files as apple\_nick.png, kiwi\_darryl.png, and so on. For **each possible option** (friend’s name), there are **4 fruit cards**.
     + Now that you have retrieved **one or more friend names** and given that there are **4 fruit cards** for each friend, we can start **populating the game board**.  
         
       *Please see in the* ***next page*** *for card examples !!!*

| ***4 Fruit Cards* for the friend *Darryl*** |
| --- |
| | **apple\_darryl.png** | **banana\_darryl.png** | **kiwi\_darryl.png** | **orange\_darryl.png** | | --- | --- | --- | --- | |  |  |  |  | |

* + - This is a **matching game**, thus, we will want to display a card **twice** in the game board.
    - Add more code to the function (you can certainly modularize this logic out into a separate function should you wish to) to generate cards and store it into an **Array** where the **cards are randomized** in order. This way, in **Task 3**, we can simply retrieve each card from this array in sequence.
  + **Task 3**
    - If you look at the HTML page, you will see the following code segment. The **innerHTML** of this **<div>** is currently an **empty string**.

| **<div id="game-board">**  **A new game board will replace this text**  **</div>** |
| --- |

* + - In Task 3, you will add more code to the function such that the **innerHTML** of the above **<div>** will be replaced with **HTML code**.
    - **Assumption**: The game board will always have **4 columns** and **N rows**, where **N** is:
      1. **# of user-selected friends** *times* 2.
      2. *Why 2?* Recall that for each **card**, we want to add 1 more of it so that the user can attempt to **match**.
    - In completing **Task 3**, there are **2 approaches**:
      1. You can use **creatElement**, **createTextNode**, and so on (see [this W3Schools “To Do List” example](https://www.w3schools.com/howto/howto_js_todolist.asp)).
      2. You can leverage **Template Literals** (See [W3Schools tutorial](https://www.w3schools.com/js/js_string_templates.asp)).

*See below for example output screens*

| home.html (upon loading this page for the first time - the user hasn’t clicked on any buttons) |
| --- |
|  |

*See more examples in the next pages*

| home.html (user selects **Brandon** and **Darryl** - and then clicks on the button) |
| --- |
|  |

*See more examples in the next pages*

| home.html (subsequently, user selects **Nick** and **Yin Kit** - and then clicks on the button) |
| --- |
|  |

*See more examples in the next pages*

| home.html (subsequently, user selects **Chein** - and then clicks on the button) |
| --- |
|  |