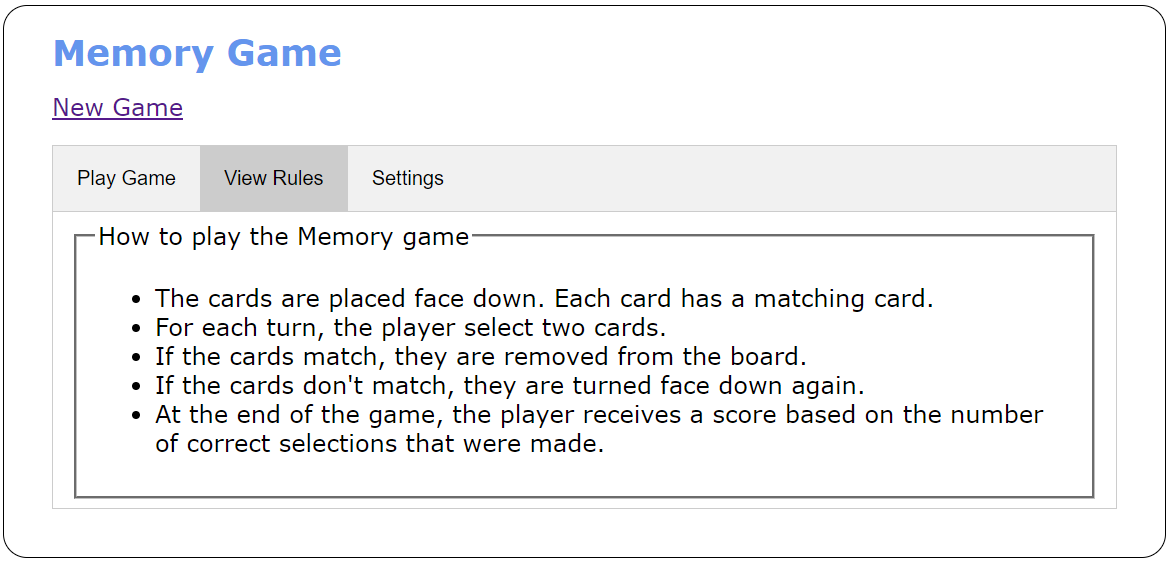
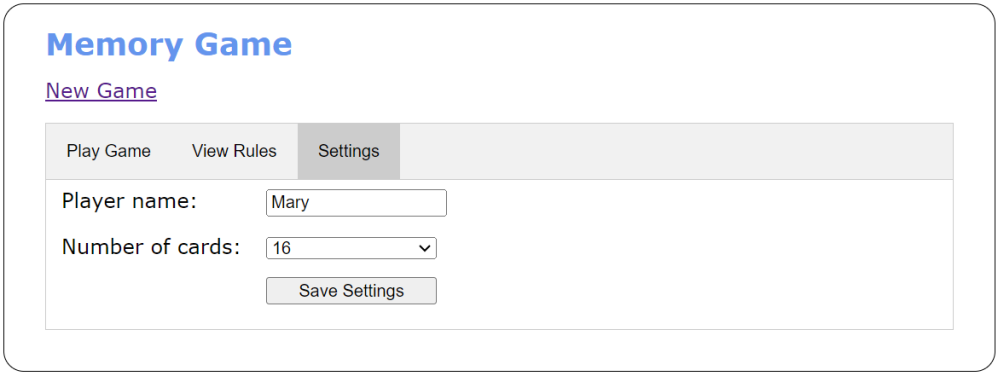
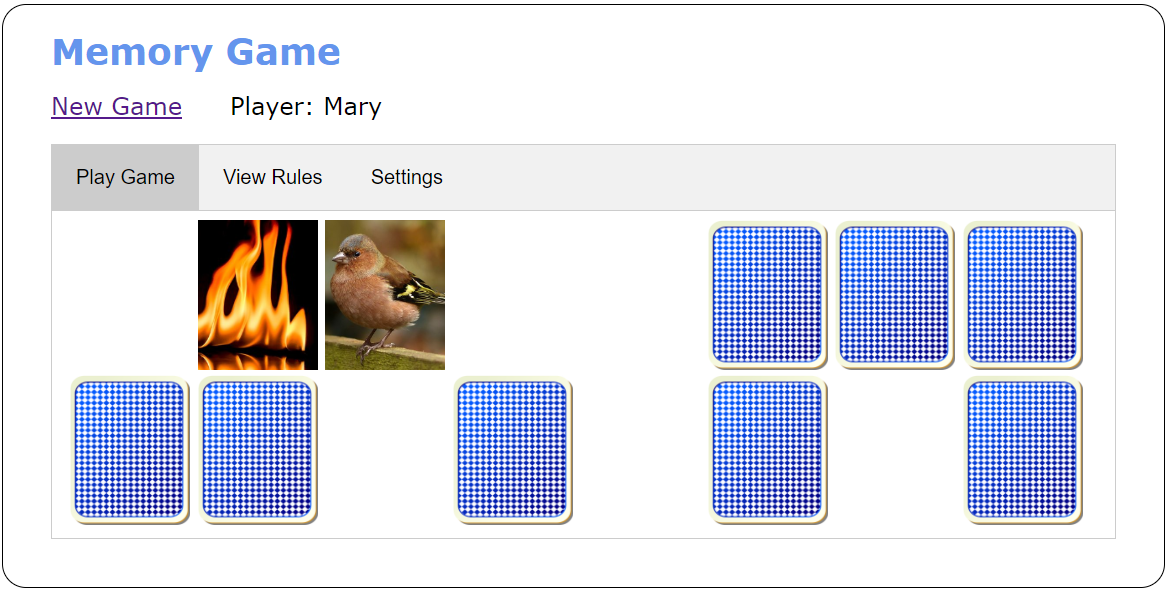
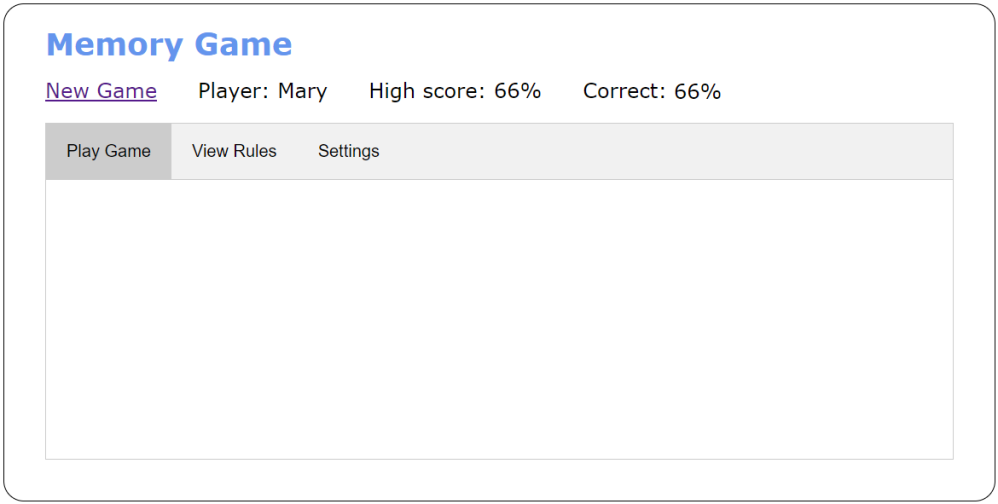
# Part 1: Create the Memory game

Part 1 of this case study has you create an app that allows the user to play the Memory game. The objective of this game is to turn over two cards at a time, trying to find cards that match until all cards have been matched.

Prerequisites: Chapters 1 to 11.

## User interface



## Specifications

* The main portion of the user interface should be displayed within the three tabs shown above.
* Images are provided for 24 cards, the back of the cards, and a blank card. All these images should be preloaded when the app starts.
* By default, the game should display six rows of cards with eight cards in each row, for a total of 48 cards (two of each card). The user can change the number of cards used as well enter a player name in the Settings tab.
* When the user clicks the Save Settings button, the player name and number of cards should be saved in local storage. In addition, the page should reload so the player’s name, the player’s high score if previous games have been played, and the correct number of cards are displayed. (You can use the reload() method of the location property of the window object to do this.)
* Create the HTML for the displayed cards by randomly selecting from the cards array. The cards for each row should be in a <div> element, and the <div> elements for all the rows should be in the <div> element with an id of “cards”.
* When the user clicks on a card whose back is displayed, it should flip to show the front of the card. The back of the card should fade out and the front of the card should fade in. (You can use the setInterval() method and the style.opacity property of an image to create a fade effect.)
* When the user clicks on a second card whose back is displayed, it should also flip as described above. Then, the second card should display for 1 second. If the cards match, they should flip to a blank image. If the cards don’t match, they should flip to show the back of the cards.
* Each time the user completes a game, the user’s high score should be updated and displayed. In addition, the percentage of correct selections for the game that was just completed should be calculated and displayed. The high score should also be stored in local storage so it can be compared against the score for the user’s next game.
* When the game ends, the user can click the New Game link to start another game.

## The HTML for each card

* The HTML for each card should consist of an <a> element with its id set to the src attribute for the card image and its href attribute set to “#”. The <a> element should contain an <img> element with its src attribute set to the image for the back of the card and its alt attribute set to “”.