Unit 8: React – Assessment #7

Assigned: 02/22/23

Due: 02/22/23

Completed: 02/2/23

**LMS:** <https://lms.grandcircus.co/mod/assign/view.php?id=22850>

**Google Doc:** <https://docs.google.com/document/d/1iVaGm8Ye9b3PFj4LAX_Fm_n_EQzQyhkkhr4XwCELG-8/preview>

**GitHub:** <https://github.com/gc-submissions/assessment-7-clintmsmith>

**Task:** Using the starter code, build a React application that displays a list of players and their scores. It includes a form that lets the user add players and a button that removes all players.

**Setup and Submission:**

See readme.md for setup instructions. Commit and push to submit.

**Build Specifications:**

The application consists of three components in addition to the App component. Display the **PlayerList** component in the App component. It also has a **Player** interface for the model.

**Player Interface**

* Represents a player object in the array of players.
* Has two properties: name (string) and score (number)

**PlayerList Component**

* Displays an **HTML table** of players, including two columns: **Name and Score**.
* The rows of this table must be rendered using the **PlayerRow** component (see below).
* The data for the table comes from an array in the *state* of PlayerList. Set the initial values for the array to [ { name: "Fox", score: 12 }, { name: "Turtle", score: 5 } ].
* Includes a "Clear List" button. When the "Clear List" button is clicked, the array is cleared and the table appears empty.
* **PlayerList** displays a **PlayerForm** component, which is used to add players to the array.
* Use CSS to add a 10px margin to the PlayerList component.
* Includes an <h1> that says "Players". Use CSS to make the text of this <h1> DarkOrchid in color.

**PlayerRow Component**

* Takes a **player** prop of type **Player**.
* Renders a single table row with two cells to display the name and score of that player.
* **PlayerList** must use **PlayerRow** when rendering the table.

**PlayerForm Component**

* Renders a form with an input for name, an input for score, and a submit button.
* Takes an **onAddPlayer** callback prop. **onAddPlayer** has one parameter, a **Player** with name and score values from the form.
* When the form is submitted it should call the onAddPlayer callback.
* When the form is submitted it should also clear the form, setting both inputs to empty.
* In **PlayerList** handle the **onAddPlayer** callback and add the **Player** to the list.

**Example Screenshot:**

Here is a sample screenshot to illustrate the task.

Graphical user interface, text, application

Description automatically generated

**Grading Criteria:**

This assessment is graded out of 10 points.

1. There is a correctly-typed interface named Player that includes name and score fields.
2. There is a PlayerRow component that renders a <tr> with two <td>s.
3. PlayerRow has a player prop of type Player. It uses this data to render the name and score.
4. PlayerList displays the names and scores of the players from the array.
5. PlayerList displays the names and scores using a correct HTML table format with a header row. And it uses PlayerRow for the rows of the table.
6. PlayerList has a 10px margin and it's "Player List" <h1> is DarkOrchid.
7. The "Clear List" button in PlayerList resets the array of players to an empty array.
8. PlayerForm is displayed and includes inputs for name and score and a submit button.
9. PlayerForm does not reload the page, but does clear out the form fields on submit.
10. Using a callback prop, a Player with correct name and score is added to PlayerList when PlayerForm is submitted.