GAME111: Web Design and Database Fundamentals

Final Assignment

Creating a Leaderboard

Total Marks: /15

Overview:

You will use your knowledge of HTML, CSS, JavaScript and AJAX api calls to create a leaderboard webpage that interacts with a NoSQL database (provided by your professor)

**Your submitted code MUST be you own.**

Information:

You will access a hosted web server through AJAX calls in your HTML page. The server is hosted here **https://lime-faithful-hippo.cyclic.app**.

The data returned will be in JSON format. To view a sample and the structure of the data check out the SampleData.json file linked in the project.

Tasks:

-**You will create 4 HTML pages** (or if you want to try a single flexible HTML page you can. Just run it by me first)

-The first page will fetch all the Game data from the server and display it in a table

-The second page will allow you to add a new game to the sever to add high scores

-The third page will fetch all the leaderboard scores for a game from the sever and display it in a table

-The final page will allow a user to enter new leaderboard data for the game

The Nitty Gritty:

Your tables will be JavaScript rendered. Meaning there should be no table tags in your HTML pages, the JavaScript page will render them after an AJAX call.

Your forms that you post should do some minor validation and cleaning of the data. **Use the trim function.**

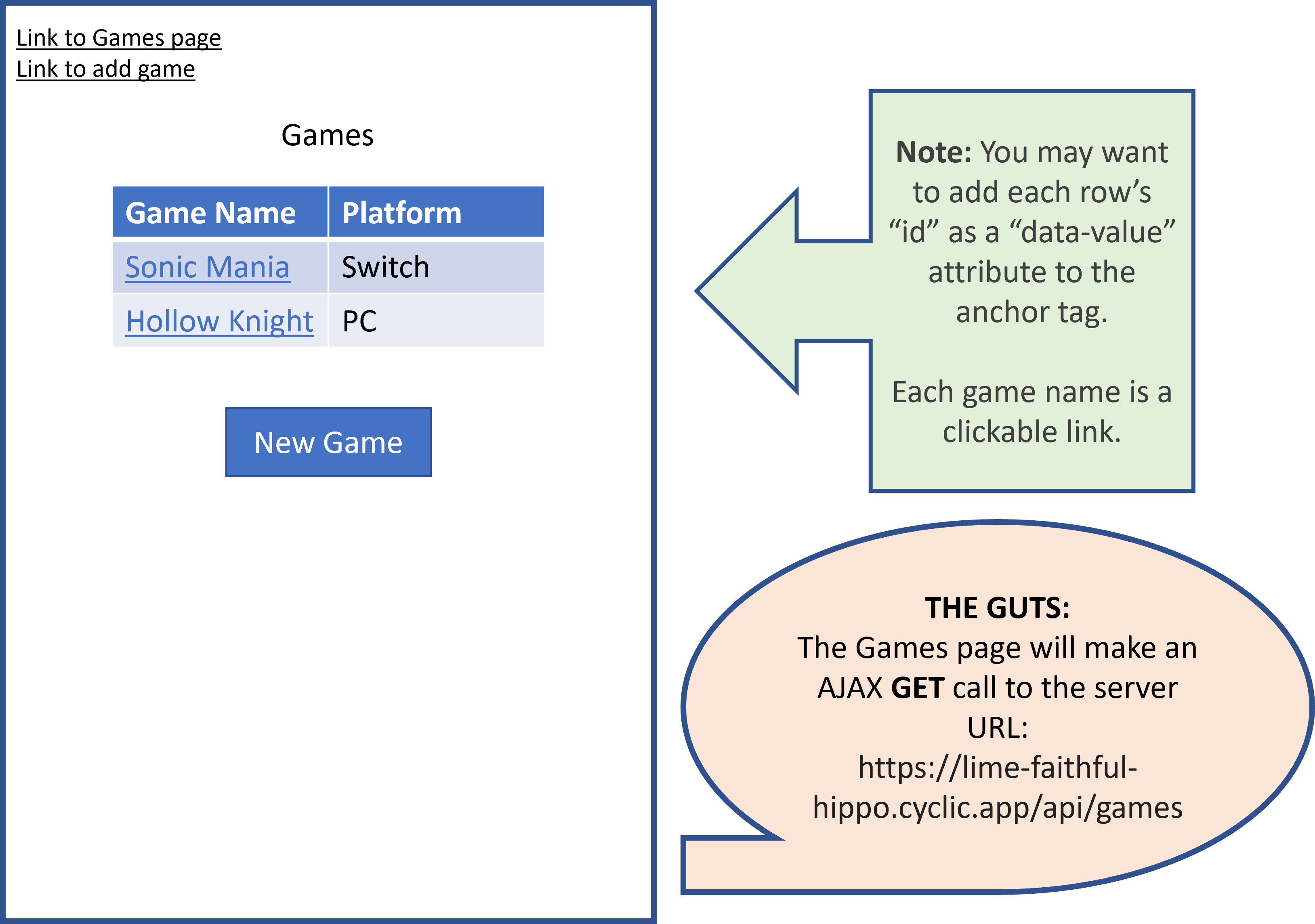
**Business Rules:**

These are made up rules that need to be met in order for our web pages and database to work.

1. Numbers must be numeric
2. Strings must be trimmed. Example. “Hi I’m Nate” is different than “Hi I’m Nate” you can see the first has leading and trailing whitespaces (aka spaces)
3. Time is a string in hr:min:sec:ms format
4. When posting to leader board you have to have a value in either “score” OR “time”

**Read Games Page:**

The read games page should look something like this:

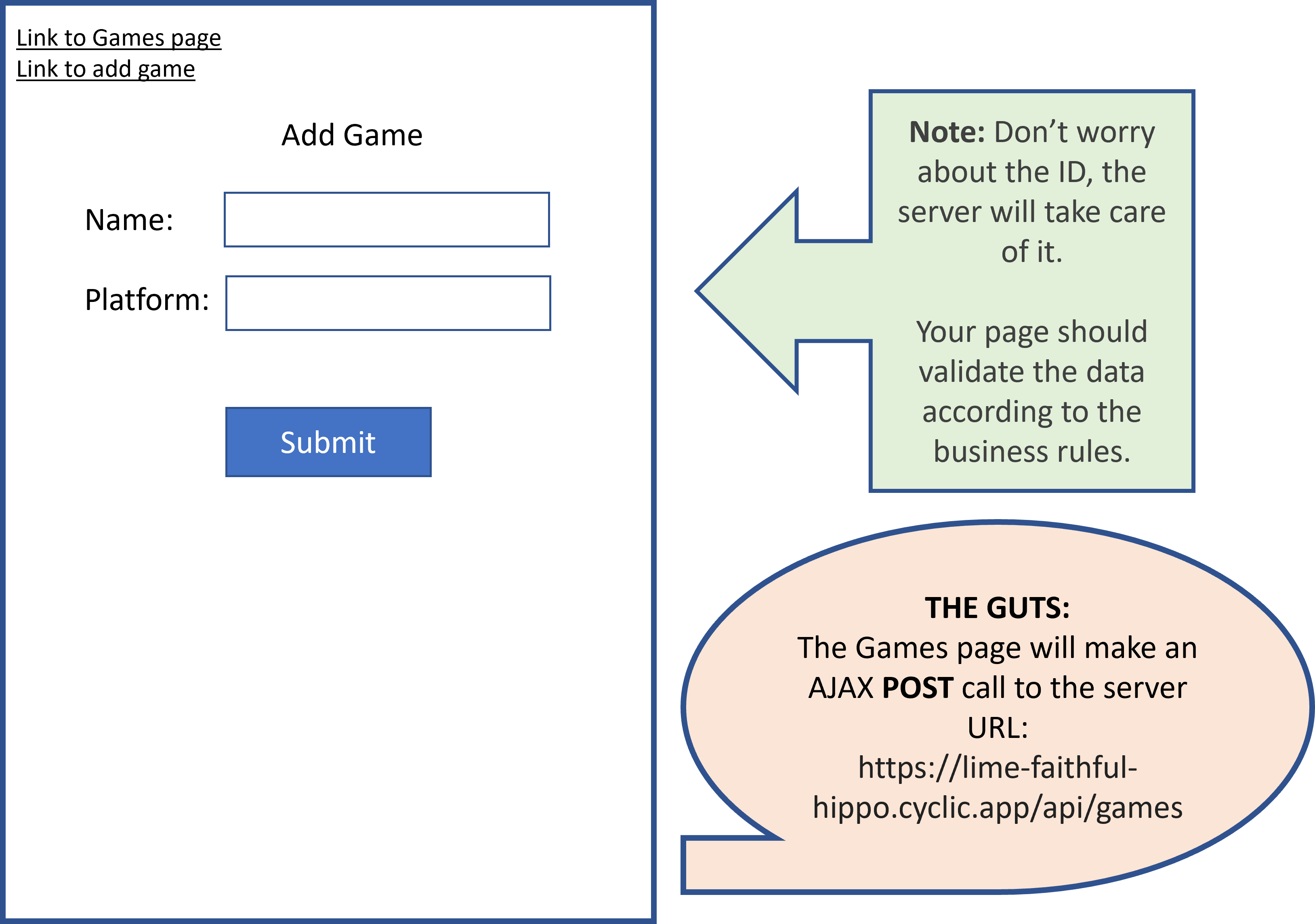


When a user clicks on the “New Game” button, it should take them to the “add Game/Post game” page.

Clicking on a game name should link to that game’s leaderboard.

When clicking it, you should pass the ID through the URL.

**Add Game Page:**



The Add Game page will have a form. However your code should just be input elements that have an “OnClick” button that retrieves the data in each input field.

Once you have valid input data, *(provide an error if the data is not valid describing this error),* then turn your data into a JSON string and pass it to the server.

The Server will respond with either a message saying success, or it will reply with an error.

If the server replied with an error, you will have to relay it back to the user.

If the server was OK with the post, then you will have to redirect back to the Read games page.

The data structure you pass back must look like the following:

{

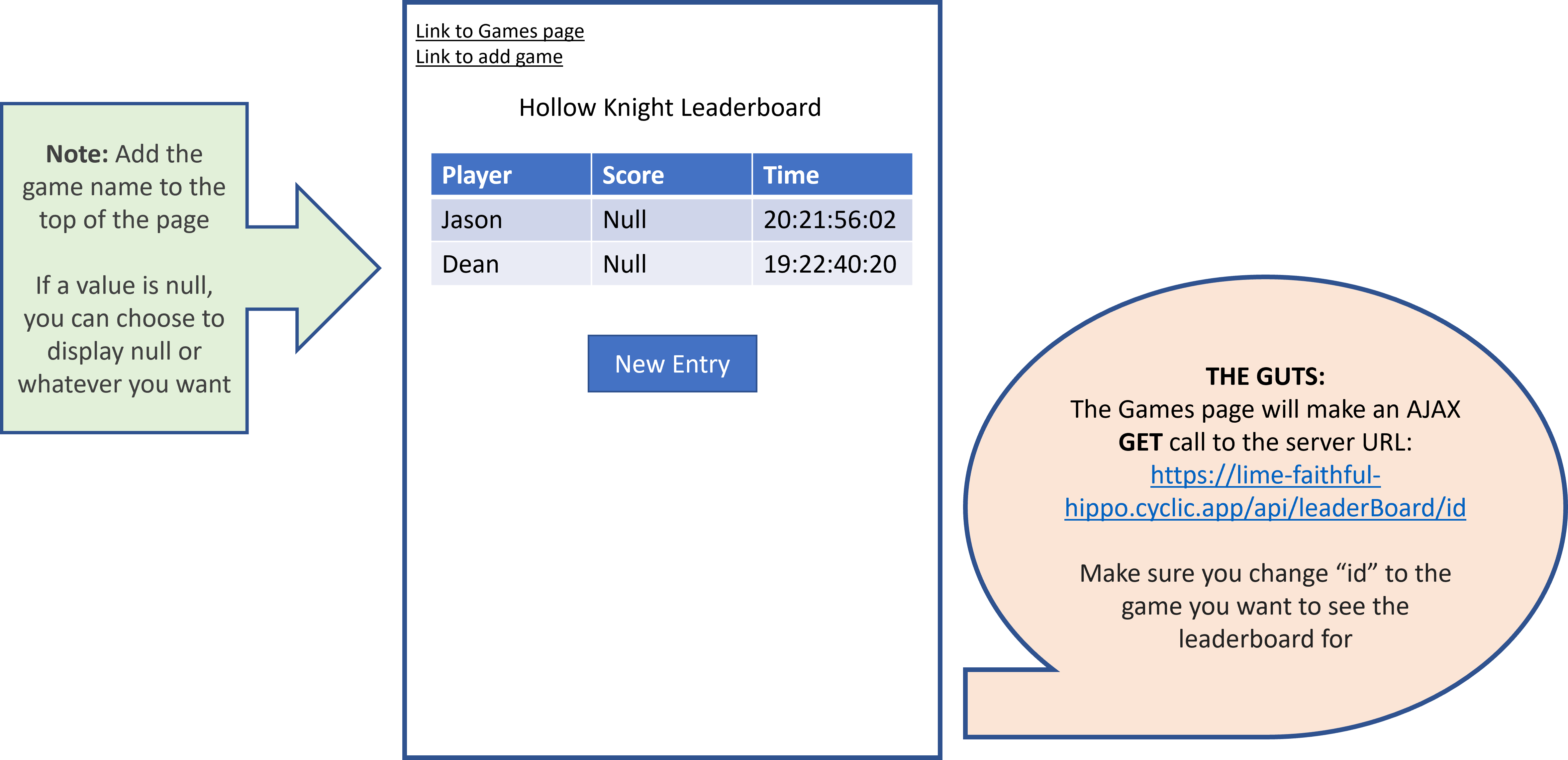
    "name" : "Game Name Here",

    "platform" : "Platform here"

}

Replace the orange text with the values from your inputs. **ENSURE** the blue names match exactly as they are shown above.

**Read Leaderboard:**



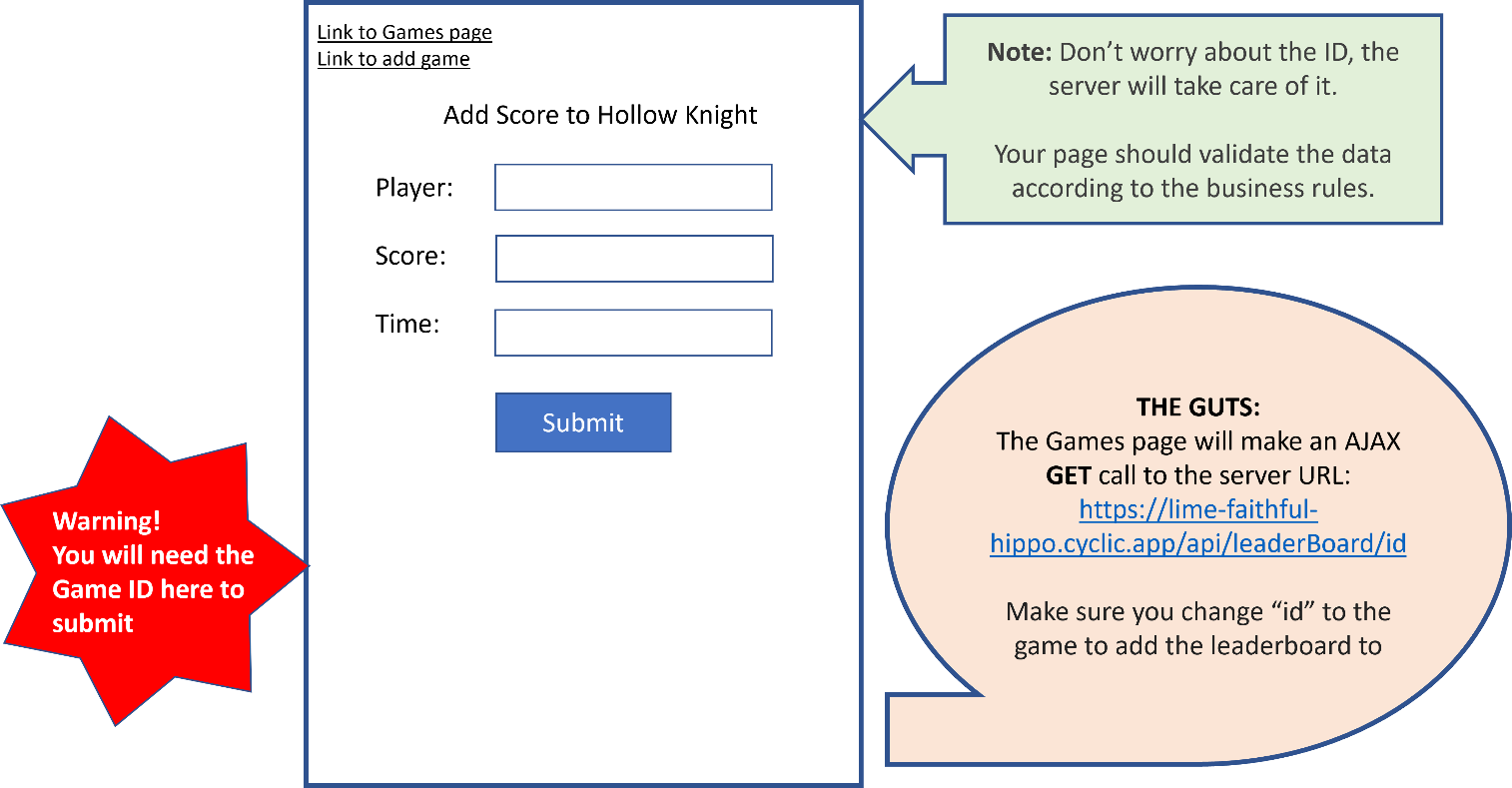
You will make an AJAX call to the server passing in the Game’s ID. It will then reply back with all of its data.

You will need to loop through each of the game’s “leaderboard” array and use that data to populate your table.

The “New Entry” button will take you to the “Add Leaderboard” page.

**Be sure to pass the “ID” of the game through the URL link in the button.**

**Add Leaderboard**



The Add Leaderboard page will have a form. However your code should just be input elements that have an “OnClick” button that retrieves the data in each input field.

Once you have valid input data, *(provide an error if the data is not valid describing this error),* then turn your data into a JSON string and pass it to the server.

The Server will respond with either a message saying success, or it will reply with an error.

If the server replied with an error, you will have to relay it back to the user.

If the server was OK with the post, then you will have to redirect back to the Read games page.

The data structure you pass back must look like the following:

{

    "gameID" : 3,

    "player" : "player name here",

    "score" : null,

    "time" : "xx:xx:xx:xx"

}

Replace the orange text with the values from your inputs. **ENSURE** the blue names match exactly as they are shown above.

Helpful Links:

Extract a parameter from a URL Address: <https://www.sitepoint.com/get-url-parameters-with-javascript/>

Video explaining fetch API: <https://youtu.be/ZTQcJWixB1k?t=258>

**Grading Rubric:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Description** | Present 100% | With Few Errors 70% | With Many Errors 40% | Not Present 0% | Points Total |
| HTML Read Games page |  |  |  |  | 2 |
| Read Games Fetch API |  |  |  |  | 1 |
|  |  |  |  |  |  |
| HTML Add Game Page |  |  |  |  | 2 |
| Add Game API |  |  |  |  | 1 |
|  |  |  |  |  |  |
| HTML Read Leaderboard |  |  |  |  | 3 |
| Read Leaderboard API |  |  |  |  | 1 |
|  |  |  |  |  |  |
| HTML Add Game |  |  |  |  | 3 |
| Add Game API |  |  |  |  | 2 |
|  |  |  |  |  |  |
| Bonuses |  |  |  |  | 1 Per Bonus |
| Web Pages hosted on Github pages |  |  |  |  |  |
| Web Page is one single dynamic Page |  |  |  |  |  |
| CSS added to make it look pretty |  |  |  |  |  |
| Any other add-ons you want to use. (talk to me first) |  |  |  |  |  |
|  |  |  |  |  |  |
| Total Marks |  |  |  |  | /15 |