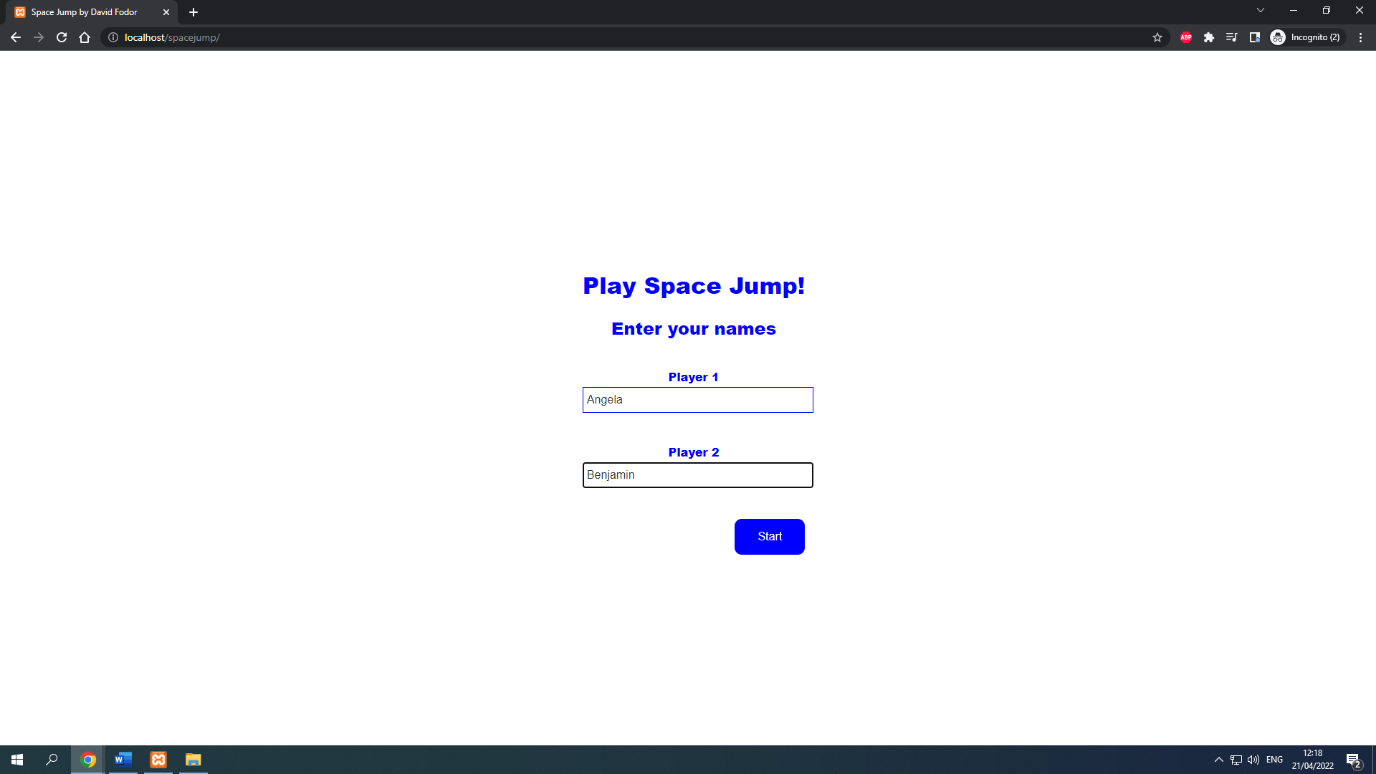
**COM549 Web Application Development  
Assignment – Space Jump**

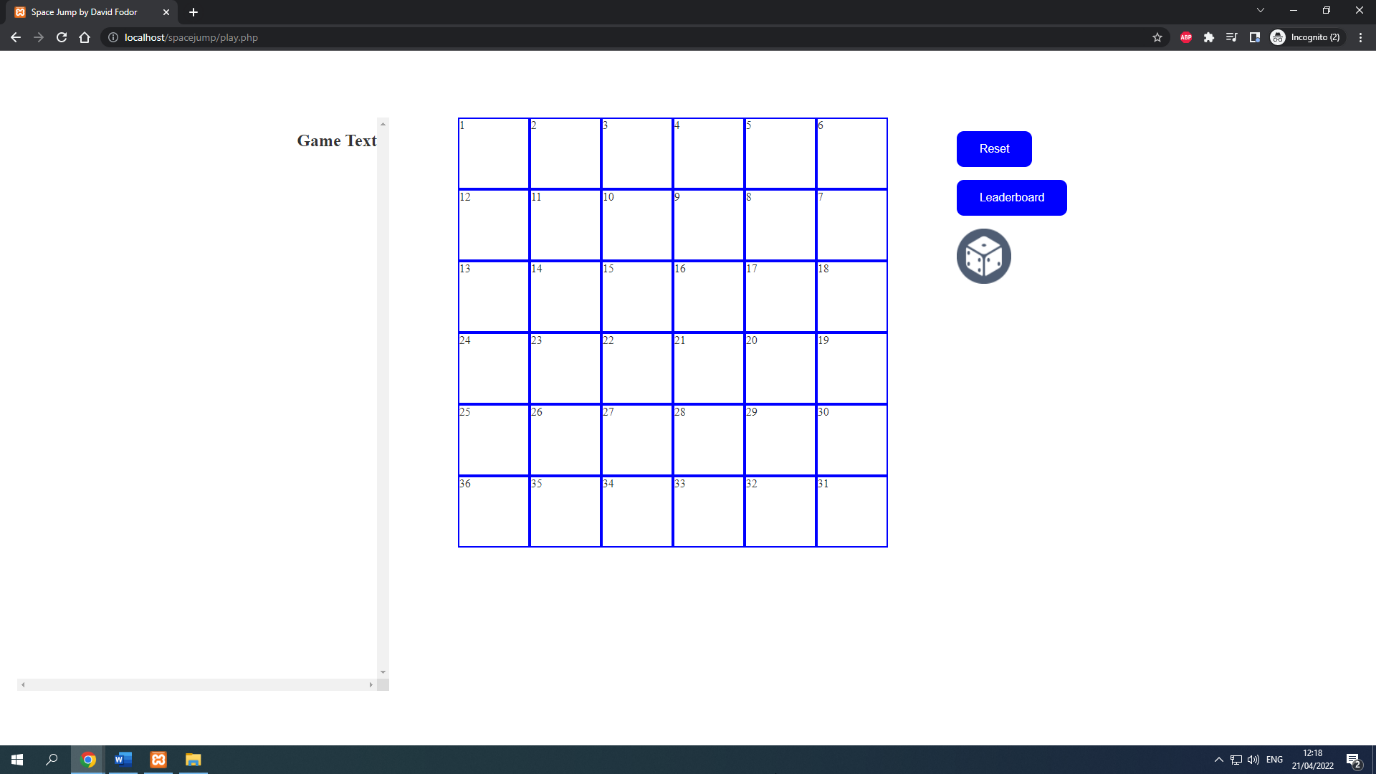
**Name:** David Fodor  
**Student ID:** B00796884

The goal of this assignment was to make a ‘Space Jump’ game, which is a variant of the game known as Snakes and Ladders. It was expected to be achieved by using server-side code through PHP and with an emphasis put on the possibility to regain the game state in case of an interruption to the page, and with communicating with a database to read and write values from and into it.

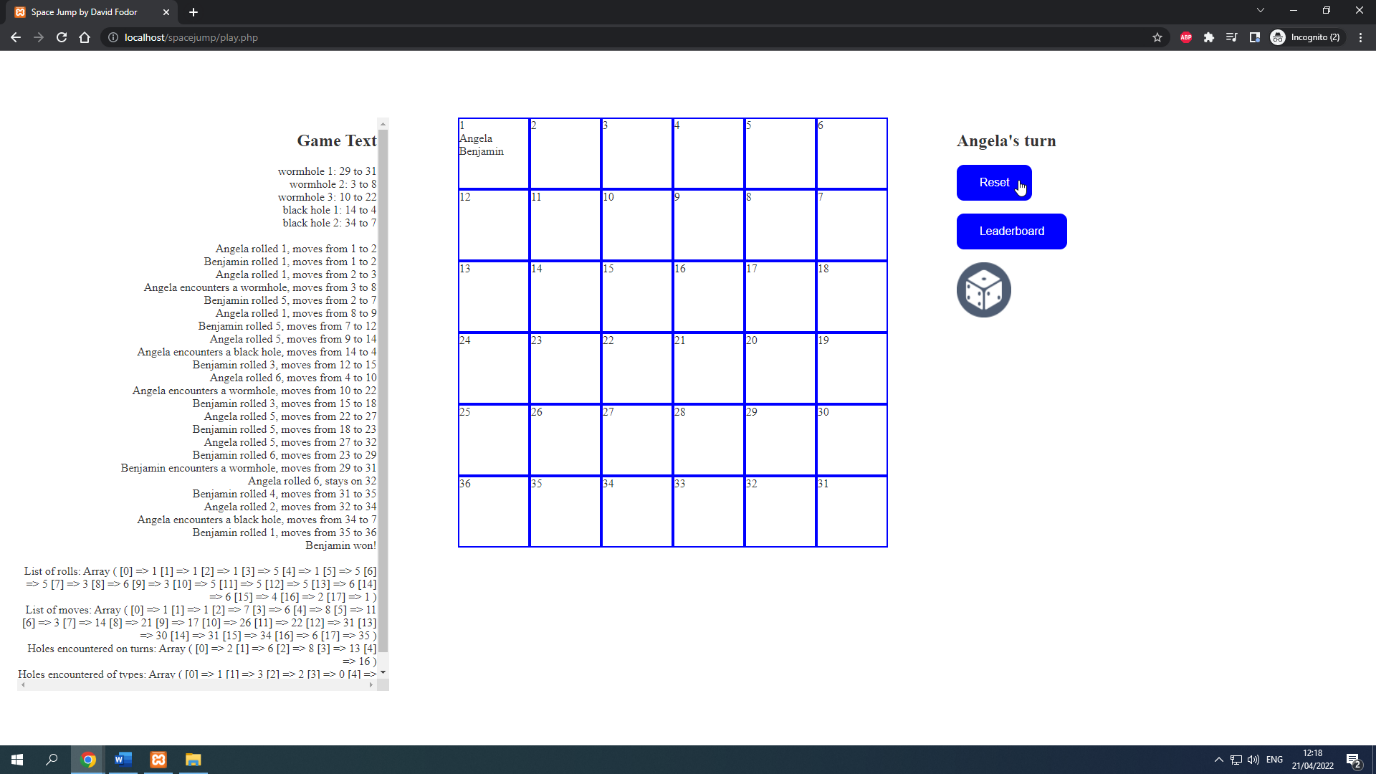
Below is a walkthrough of my solution, with some important things pointed out at each step. My code was run in Google Chrome, and by using XAMPP.



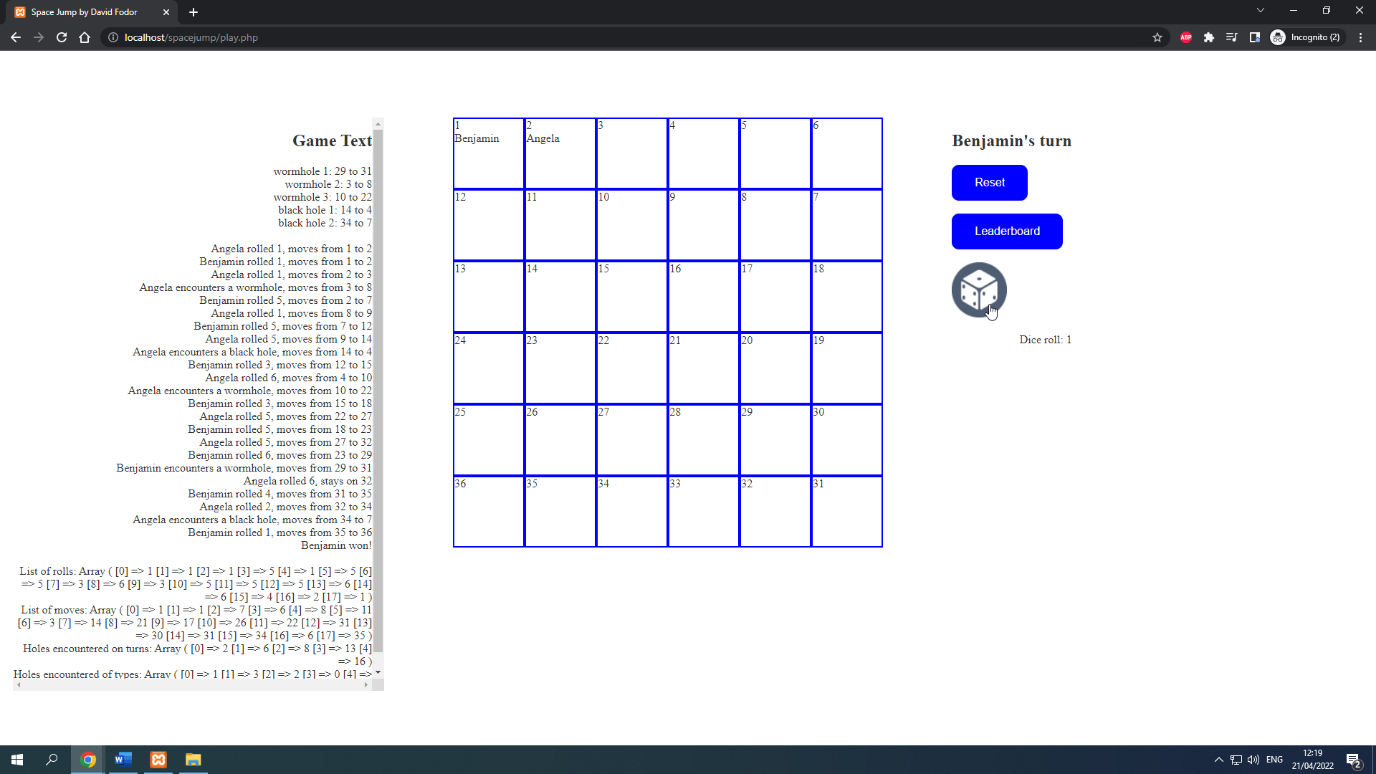
The users are greeted with this as the index page. They are prompted to enter their names, which are stored in session variables, in order for them to be retrievable later on. By clicking the Start button, the users are taken to the game’s page.



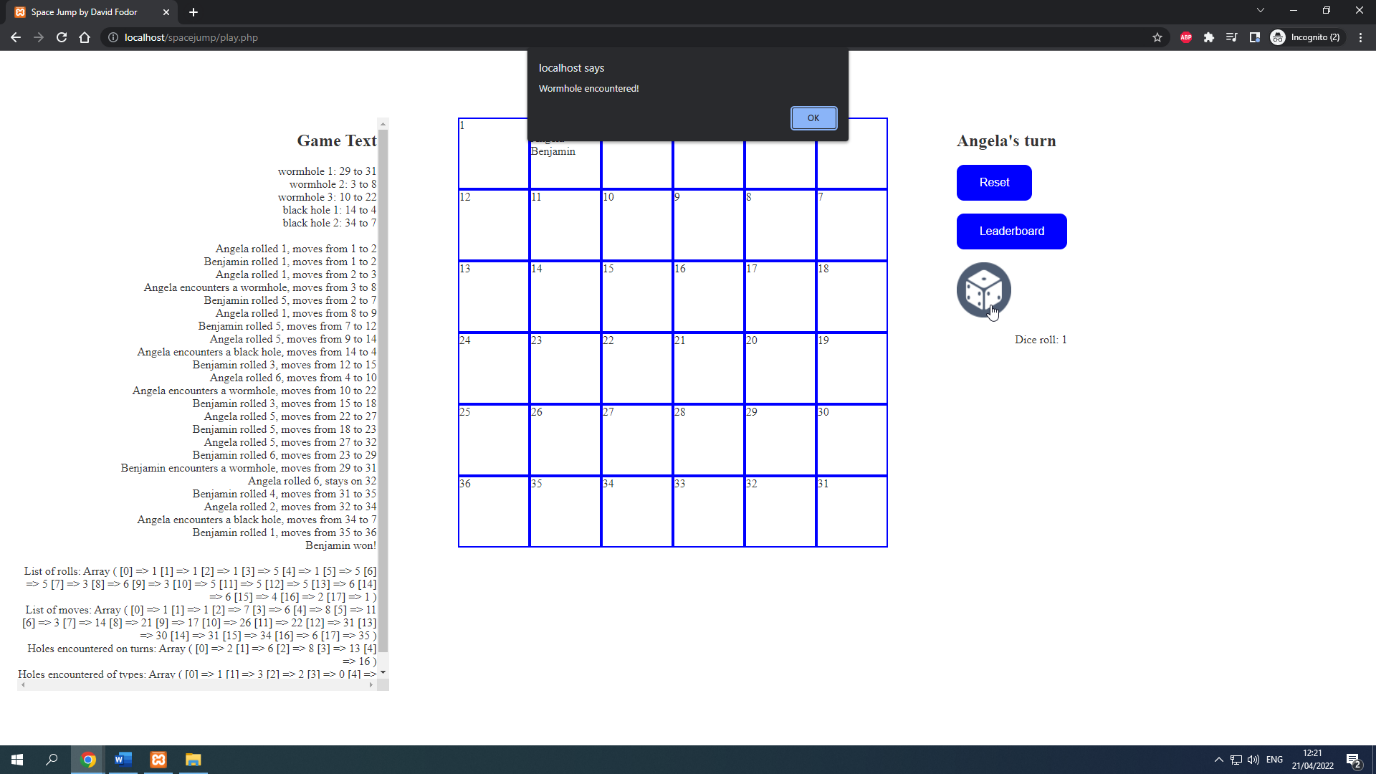
The users are greeted by an empty 6x6 board, with each of its cells numbered from 1 to 36.



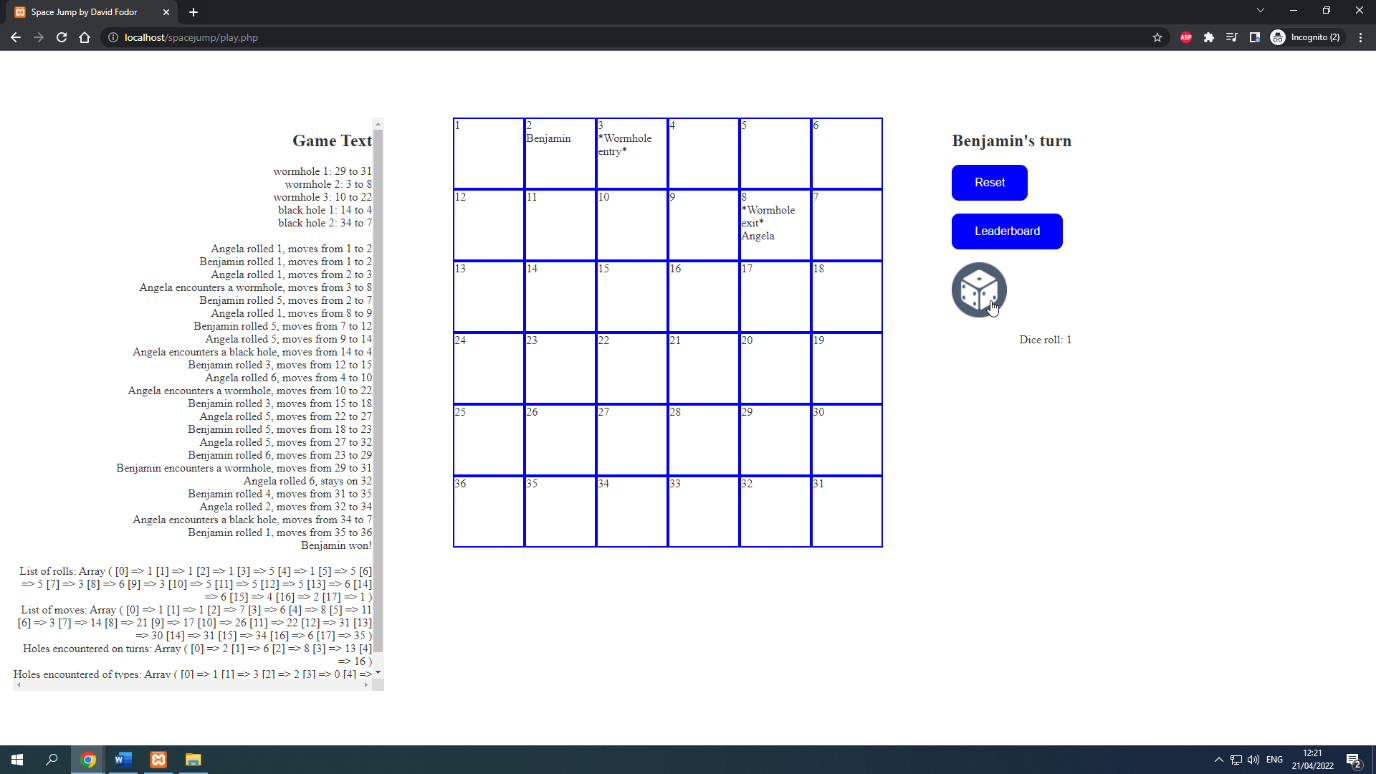
By clicking the Reset button, a new game is generated, and the previously entered names are both put on the starting square of the board. Snakes and Ladders is a game where the only influence that a player has on the gameplay, is rolling a dice.   
There is no significance to when the dices are being rolled; from the perspective of the player, all dice rolls can be made at the very start of the game, and they would not be able to tell the difference between that, and generating only one random number at each turn. Therefore, for simplicity’s sake, I have used the former solution. The outcome of the game is determined before the players visually reach the final square, and clicking the dice roll button only takes them through the pre-recorded turns.  
These pre-recorded turns are displayed in text form on the left side of the screen. For this solution, it has been left in, to illustrate the working of the game. Ideally, we would not write out the outcome of the game for the player to see upfront, in order to uphold the illusion of chance.  
Three wormholes and two black holes have been generated, each with an entry and an exit square, which are randomly determined, but in a way that they do not result in an infinite loop. The holes are initially not visible on the board, until a player steps on them.  
The game can be reset at any point by clicking the Reset button, which generates a new board. Until the game is reset, the board upholds its state even when the player refreshes the page, or closes it. This is achieved by session variables.



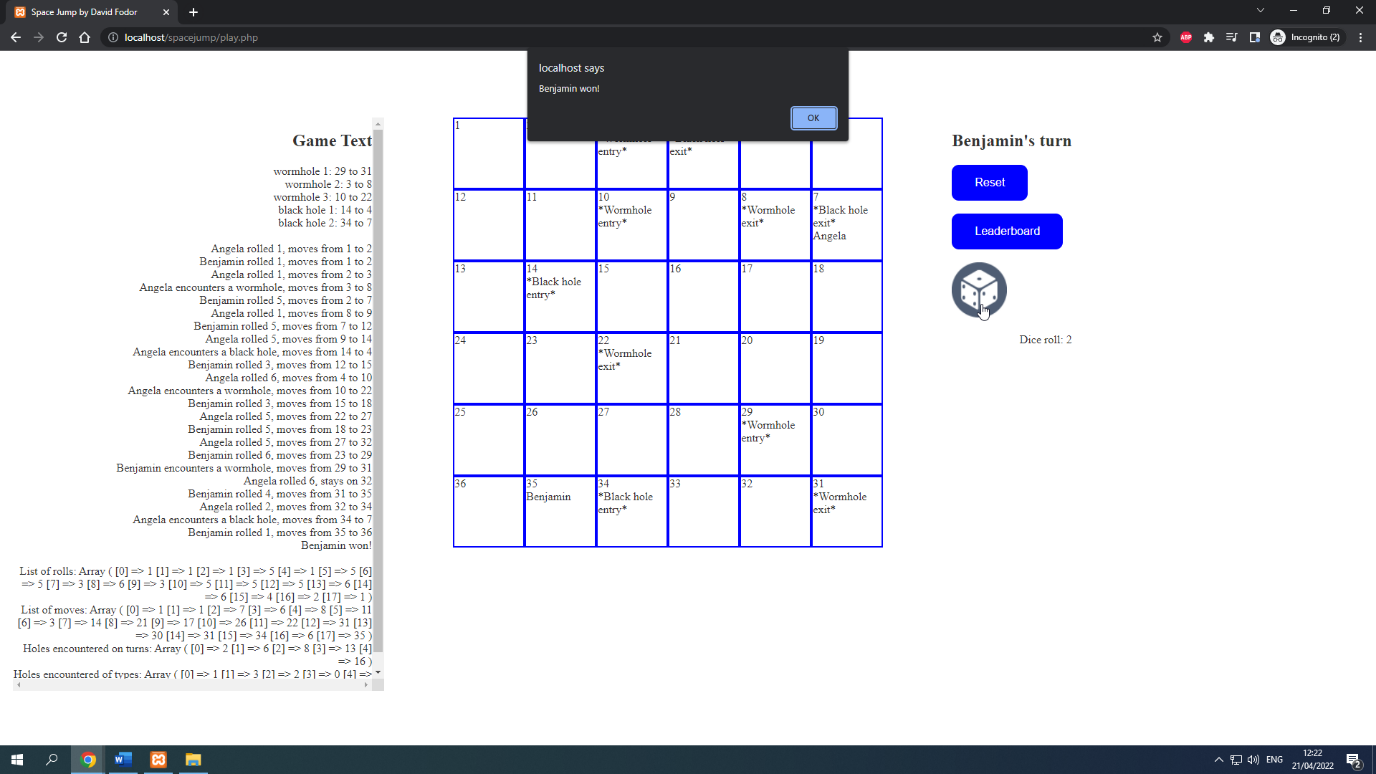
When clicking the dice button, a random number between 1 and 6 is generated, which turned out to be 1 in the picture above. The first player, Angela has been moved forward by 1 square in response. The text just right of the board now reads that it is Benjamin’s turn, who is the other player.



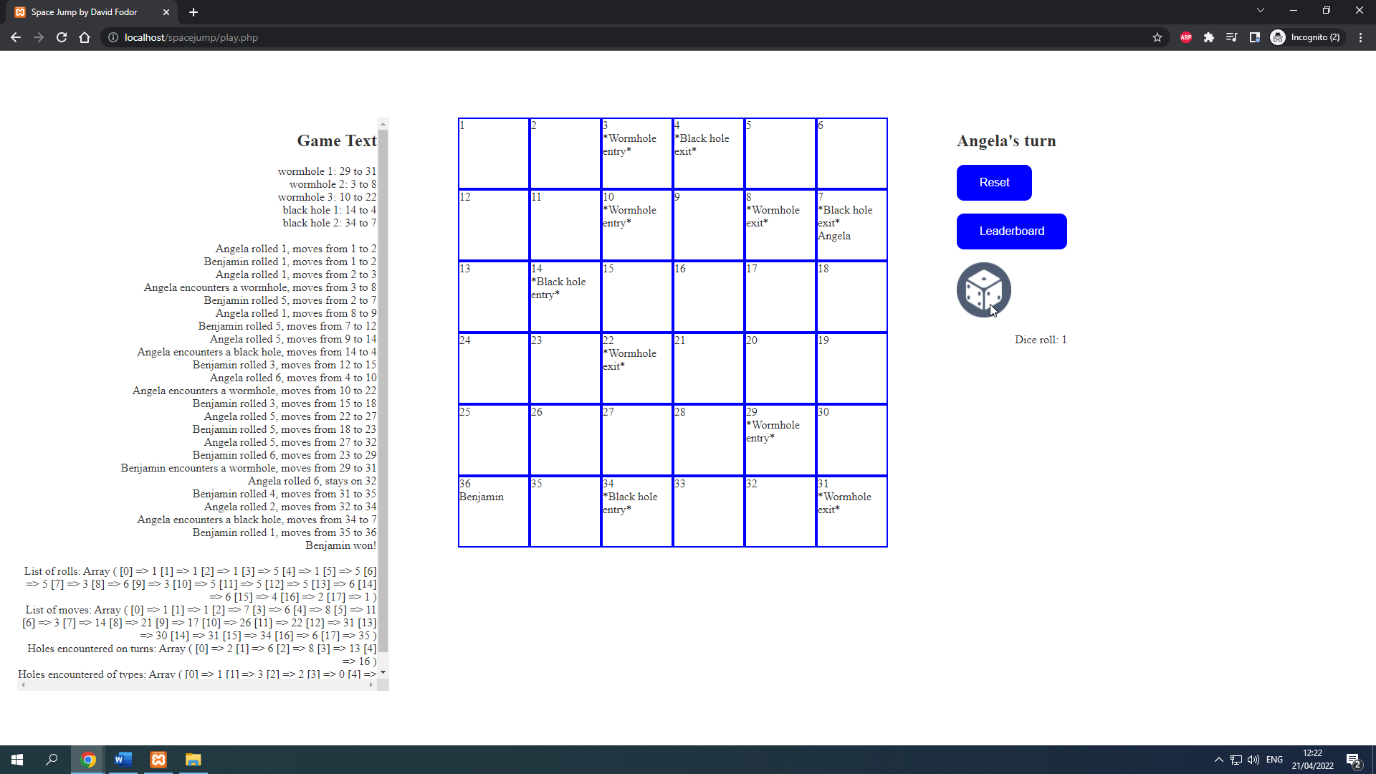
A few turns into the game, and a player has stepped on a field that had a hole on it. To grab the players’ attention, an alert has been implemented. The visuals get updated only after the alert has been acknowledged by clicking the OK button.



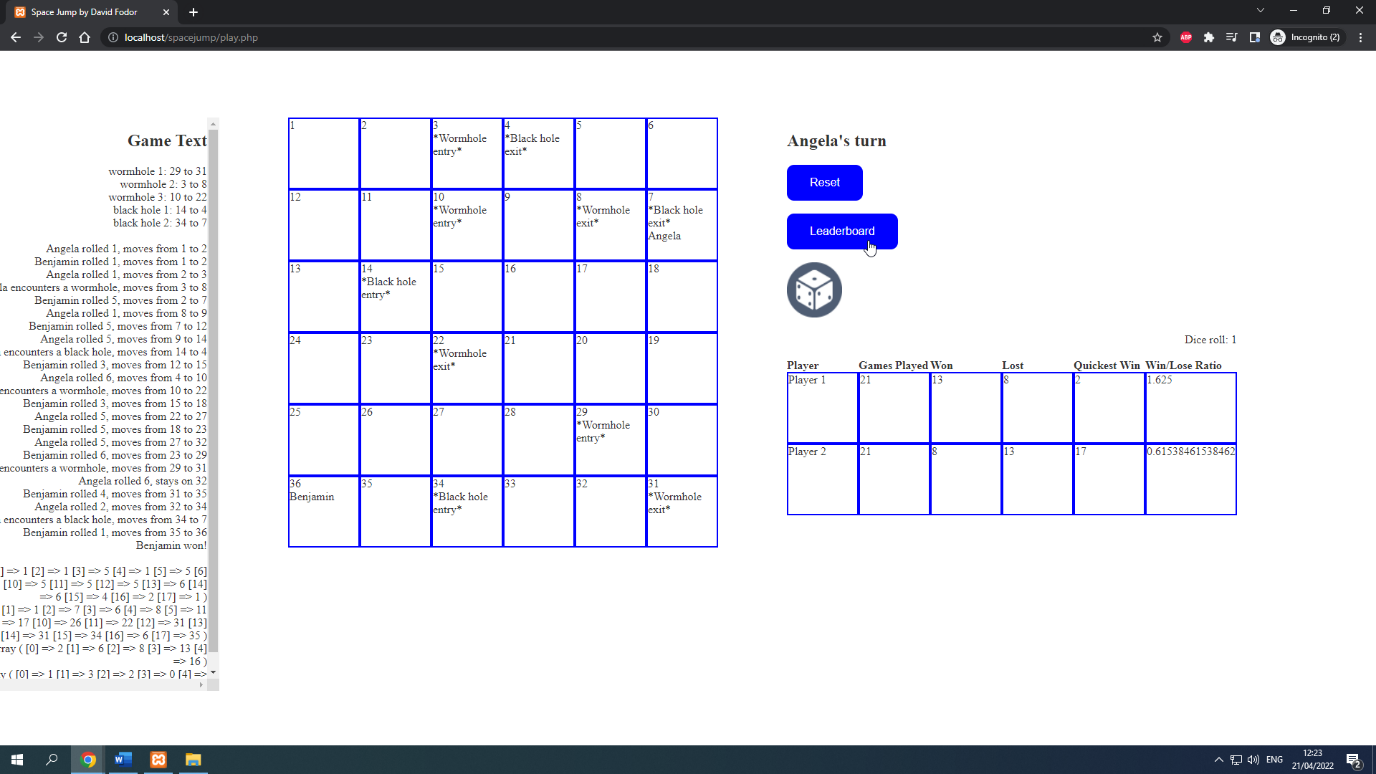
The board has now been updated to show the entry and exit square of the hole that has been revealed. The player has been transported to the hole’s exit square.



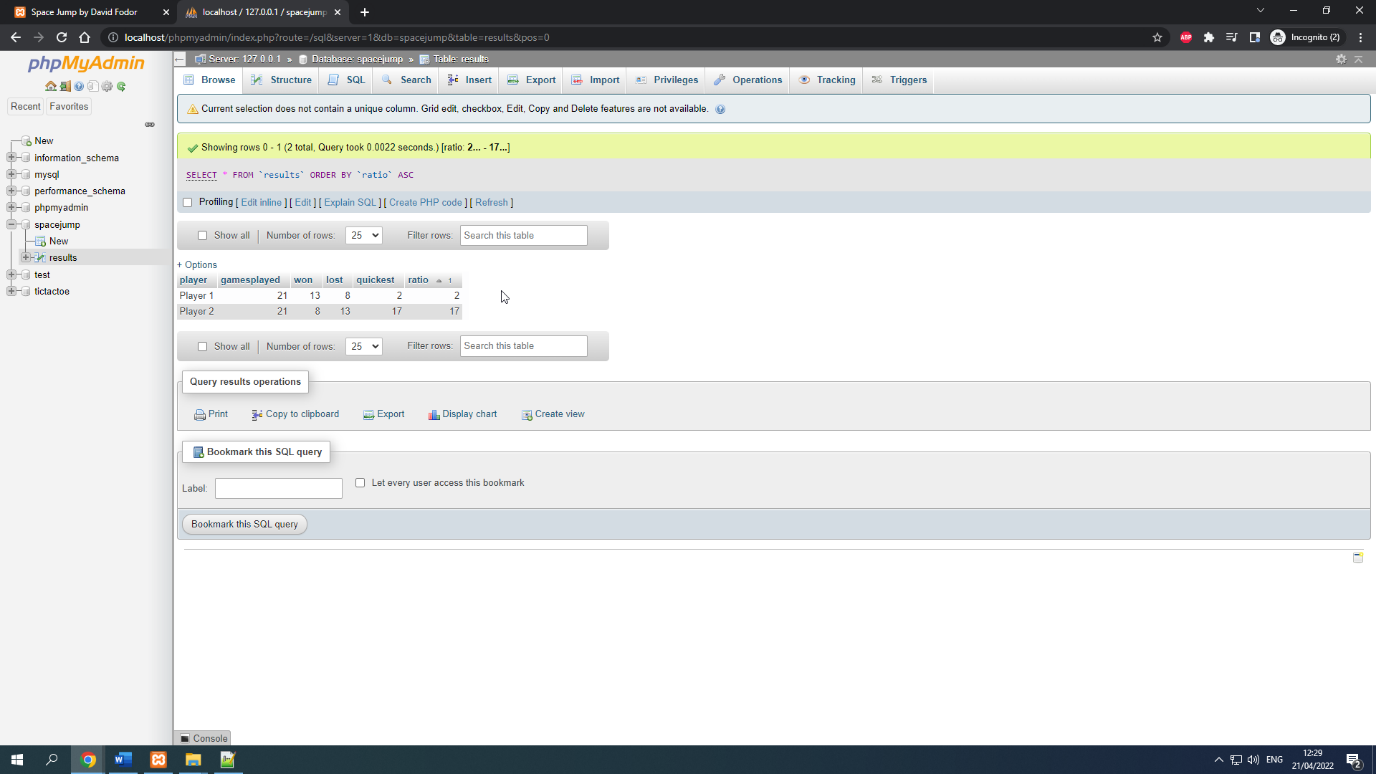
The game goes on, and at one point, Angela is only 4 squares away from the final square, but rolls 6, so she has to stay at her square. Then Benjamin reaches the final square, so an alert appears.



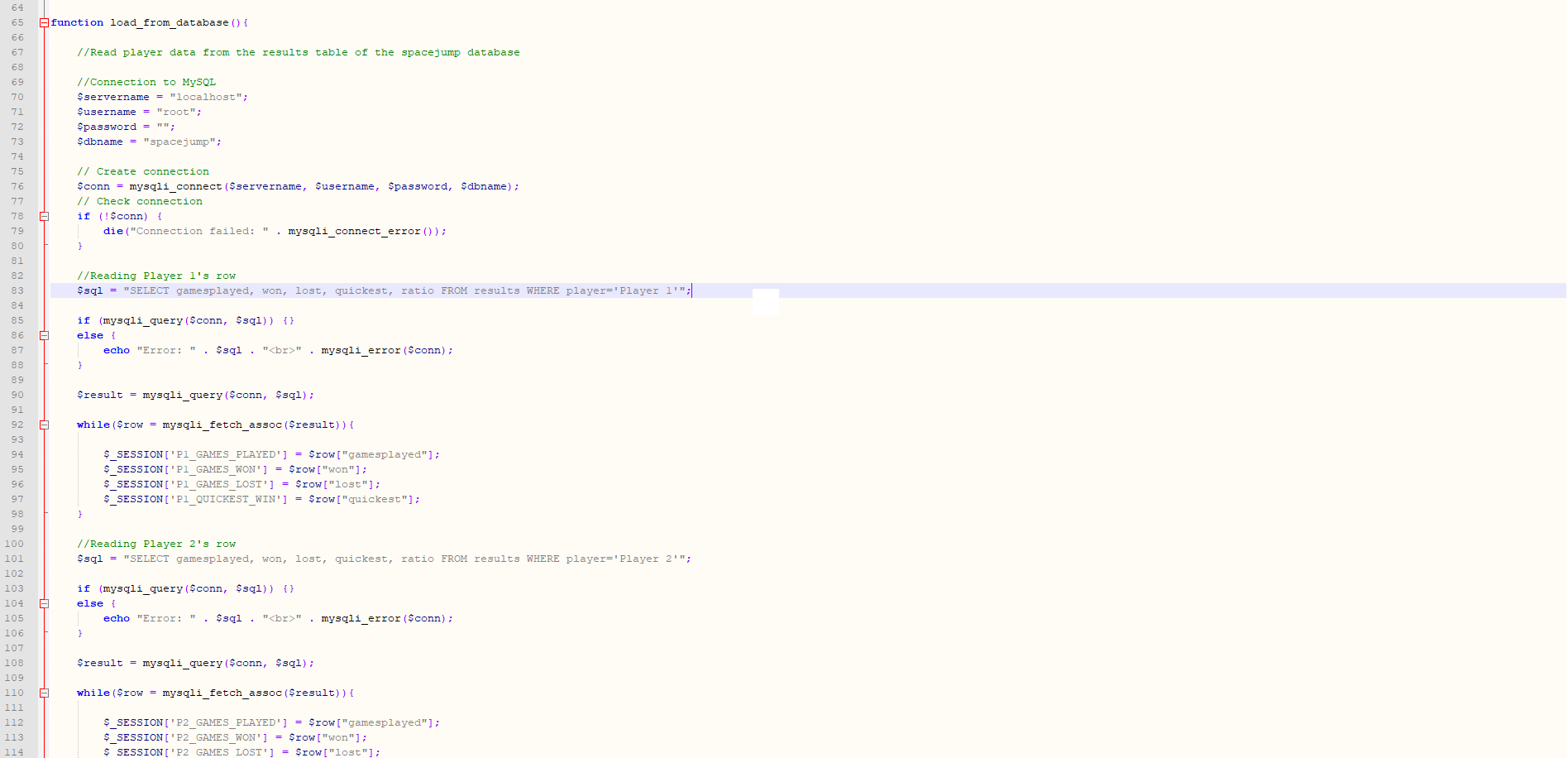
Once the alert is acknowledged, the final state of the board is shown, with Benjamin occupying the final square. At this point, the roll button gets disabled and stays that way until a new game is started.



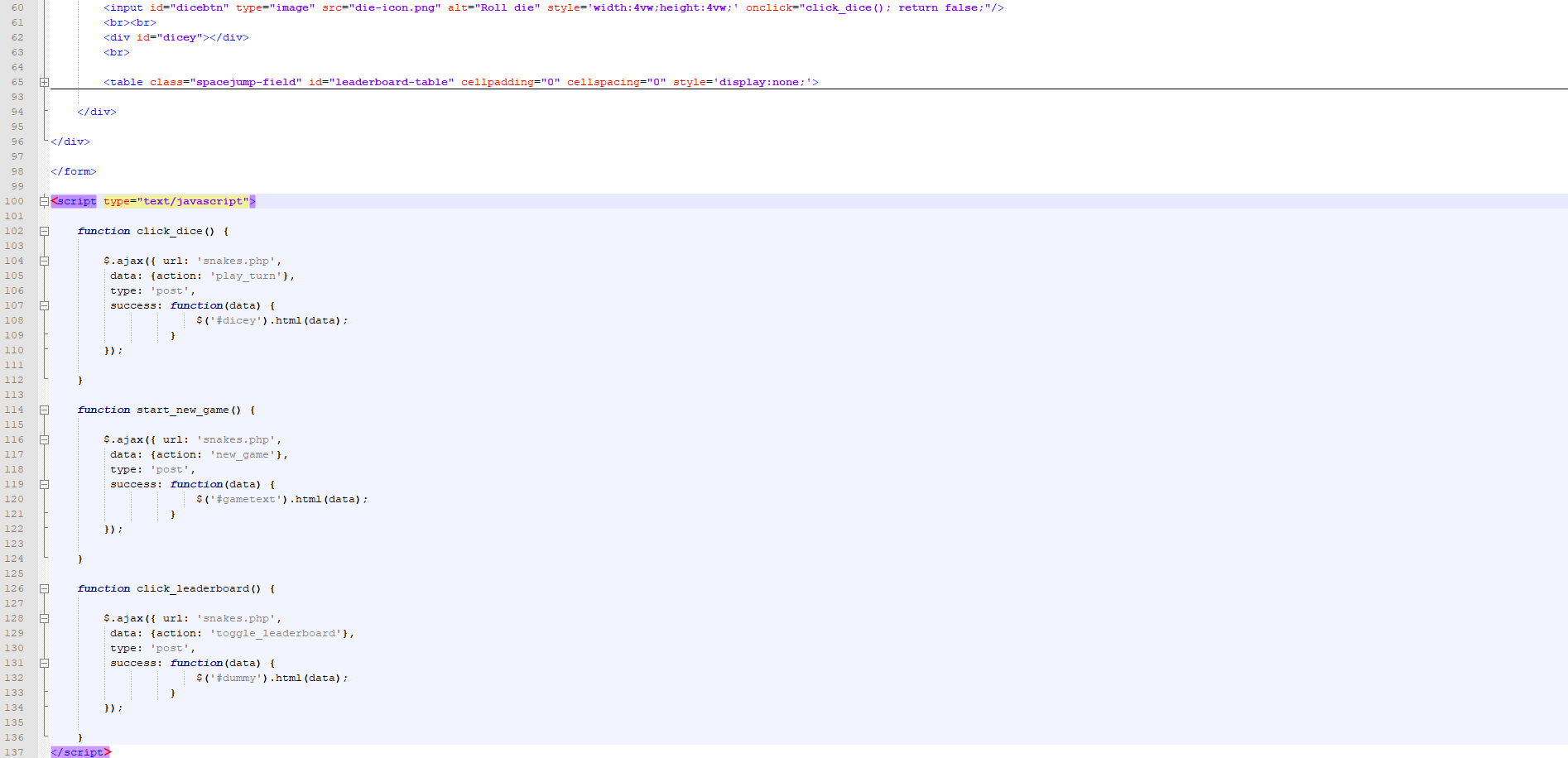
Either after the game is over, or during the course of the game, the leaderboard can be accessed by clicking the leaderboard buton. The leaderboard is initially hidden, and clicking the button toggles it on or off. Each time the button is clicked, the values of the leaderboard get updated by reading them from the database table where they are stored. The database gets updated each time a game is over.



Said database looks like this in phpMyAdmin. By taking a look at the values inside the table, and the values inside the leaderboard table, we can conclude that these values are the same, therefore the update has been successful.



Communication with the database in PHP is made possible with a few key commands, like mysqli\_connect, mysqli\_query, and mysqli\_fetch\_assoc. An $sql varable is introduced to store the SQL query that we want to make to the database.



During the implementation of the game, it was important to be able to update only parts of the webpage, without reloading the whole page. This can be achieved by AJAX, which is a technique that stands for Asynchronous JavaScript and XML.   
Provided we have jQuery declared in our HTML document, we can use the ajax() method for this task. We include some parameters inside the brackets of the method, written using the JSON format. One such parameter that we need in this case, is the name of the PHP function that we want to execute upon a click of the button, and also the name of the PHP file which includes said function.



The content of the PHP file that we are referring to looks like this. There is a switch case which directs us to the requested method, and from there, we can reach back to the HTML element by using JavaScript inside our PHP function.  
The picture above also includes some session variables, and the session\_start() method, which starts the session to begin with.

**Instructions for testing the game**

1. Move the *'spacejump'* folder inside the *'htdocs'* folder of XAMPP, *'C:\xampp\htdocs\'* by default

2. Start *'XAMPP Control Panel'*

3. Start *'Apache'* and *'MySQL'* inside *'XAMPP Control Panel'*

4. Open your browser and go to *'localhost/phpmyadmin'*

5. Click the *'Import'* tab

6. Click the *'Choose file'* button

7. Locate the *'db.sql'* file and open it

8. Press the *'Go'* button

The *'spacejump'* database with the 'results' table has now been created.

9. Go to *'localhost/spacejump'* in your browser

The game is now ready to be played.