Assignment 3

This file document the Wordle game updtes with php and Ajax.

Overview

We used php to build a top 5 best score board, which will use `$_SESSION` array to store the best 5 attempts to solve the Wordle game. (We choose 5 since Wordle is different from the game Yatzy). And a global variable `attempts` to keep track the total number of attempts of user, when a user win the wordle, the attempts will increase by one, and decrease by 1 if the player loses all the chances.

Ajax

We use the axis to call perform the Ajax functions, the details are in the comment of the code.

```
import axios from 'axios'

// Create a object to send the ajax request to my php server
const apiClient = axios.create({
  baseURL: 'http://localhost:8080',
  // Include the cookies,
  withCredentials: true,
  // Format is json
```

```
  baseURL: 'http://localhost:8080',
  // Include the cookies,
  withCredentials: true,
  // Format is json
  headers: {
    Accept: 'application/json',
    'Content-Type': 'application/json'
  }
})

export default {
  // Get the $_SESSION array from the player.php, to get all the user attempts
  getMessage() {
    return apiClient.get('/player.php')
  },
  // Put new attempts of a player into the $_SESSION array
  sendMessage(player) {
    return apiClient.post('/player.php', player)
  },
  // Close and destroy the $_SESSION array, to refresh the score board
  destroySession() {
    return apiClient.get('/quit.php')
  },
  // To update the global variable attempts into the $_SESSION array
  postAttempts(attempts) {
    apiClient.post('/attempts.php', JSON.stringify({ attempts }))
  },
  // Get the global variable attempts from $_SESSION array
  getAttempts() {
    return apiClient.get('/attempts.php')
  }
}
```

Global Variable: `Attempts`

This variable is aim to track the total number of attempts the player since the game starts. SInce it's stored in the `$_SESSION` array, when u refresh the page, it will not be reset to 0.

We update the variable when the user either win or lose a game:

```
const resetBoard = () => {
  ...
  // Pass the variable to the attempts.php
  api.postAttempts(php_attempts.value++)
  ...
}
```

attempts.php

```
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $json = file_get_contents("php://input");
    // Get the updated variable's value
    $data = json_decode($json, true);

    // Initialize Attempt_Number if not already set
    if (!isset($_SESSION["Attempt_Number"])) {
        $_SESSION["Attempt_Number"] = 1;
    }

    if (json_last_error() !== JSON_ERROR_NONE) {
        http_response_code(400);
        die(json_encode(["error" => "Invalid JSON: " . json_last_error_msg()]));
    }
```

```
    if (isset($data["attempts"] )) {
        // Update the variable
        $_SESSION["Attempt_Number"] += $data["attempts"];
    }

} elseif ($_SERVER["REQUEST_METHOD"] === "GET"){
    echo json_encode($_SESSION);
}
```

Scoreboard

In the `player.php` we store a object:

```
let player = {
  a_number: php_attempts.value,
  a_attempts: 7 - life.value
}
```

into the session. In the method handleKeyDown, we pass this object to the player.php:

```
const handleKeydown = (event: KeyboardEvent) => {
  ...
  if (inputWord.value[currentRow.value].join('') === answer.value.toUpperCase()) {
  ...
    // Construct a player object
    let player = {
      a_number: php_attempts.value,
      a_attempts: 7 - life.value
    }
    // Send the pbject
    api.sendMessage(player)
```

In the player.php:

```php
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $json = file_get_contents("php://input");
    // Get the player object
    $data = json_decode($json, true);

    if (json_last_error() !== JSON_ERROR_NONE) {
        http_response_code(400);
        die(json_encode(["error" => "Invalid JSON: " . json_last_error_msg()]));
    }

    if (isset($data['a_number']) && isset($data['a_attempts'])) {
        // Store the number of attempts and the sucess attempts into new variables
        $attempt_number = $data['a_number'];
        $attempts = $data['a_attempts'];
        // Put into the $_SESSION array
        $_SESSION["Attempt_$attempt_number"] = $attempts;
        // Return this array (optional)
        $response = ["Attempt " . $attempt_number => $_SESSION["Attempt_$attempt_number"]];
        echo json_encode($response);

    }
} elseif ($_SERVER["REQUEST_METHOD"] === "GET") {
    // Get the $_SESSION array
    echo json_encode($_SESSION);
}
```

Lastly, we will retrieve the both the global variable and the player attempts from the session:

Lastly, we will retrieve the both the global variable and the player attempts from the session:

```
async function fetchMessage() {
  try {
    const session = await api.getMessage()
    // Get session's data
    const session_object = session.data
    const tbody = document.getElementsByTagName('tbody')[0]

    // Filter out 'Attempt_Number' and then sort the attempts in ascending order
    const session_array = Object.entries(session_object)
      .filter((player) => player[0] !== 'Attempt_Number')
      .sort((a, b) => a[1] - b[1])

    // Only keep the top 5 attempts
const top_5 = session_array.slice(0, 5)

// Put each attempts into the table, except the global variable
top_5.forEach((player) => {
  // Ignore
  if (player[0] == 'Attempt_Number') {
    return
  }
  // Create the table elements
  const tr = document.createElement('tr')
  const td_key = document.createElement('td')
  const td_value = document.createElement('td')

  td_key.innerText = player[0].substring(8)
  td_value.innerText = player[1]
```

```
        tr.appendChild(td_key)
        tr.appendChild(td_value)

        tbody.appendChild(tr)
      })
    } catch (error) {
      console.error('Error fetching message:', error)
    }
  }
```

Clean Scoreboard

We also include a button to clean the session, so the player can refresh their score board!

```
<?php
// Start
session_start();
...

if ($_SERVER["REQUEST_METHOD"] === "GET") {
    // Unset first
    session_unset();
    // Destroy the session
    session_destroy();
}
echo json_encode(["status" => "All session variables destroied"]);
?>
```

Button

```
function destroy() {
```

Button

```
function destroy() {
  // Use the api object to call the method
  api.destroySession()
}
```

Api

```
import axios from 'axios'

const apiClient = axios.create({
  baseURL: 'http://localhost:8080',
  withCredentials: true,
  headers: {
    Accept: 'application/json',
    'Content-Type': 'application/json'
  }
})

export default {
  ...
  // Method to destroy the session
  destroySession() {
    return apiClient.get('/quit.php')
  }
  ...
}
```

State of the game

1 Initial State

Run the following command to start the game

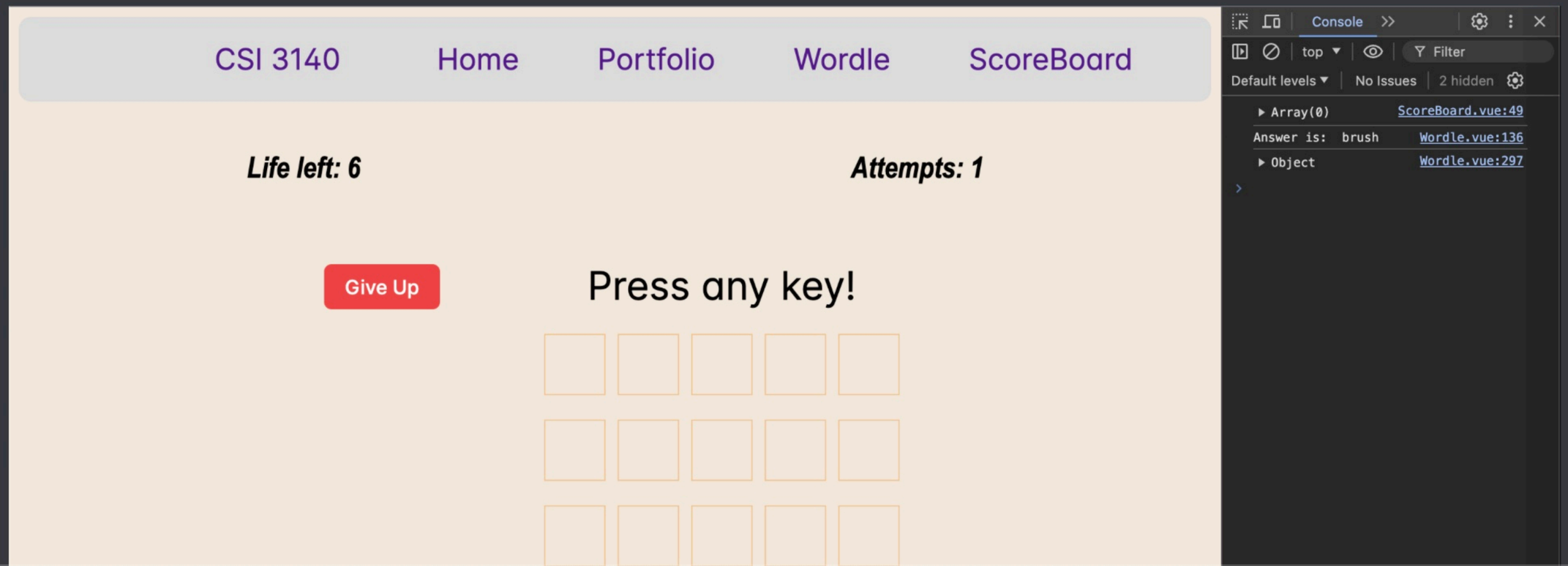
```
Wordle git:(main) ✗ npm run dev
```

Run this to start the php server

```
→  Wordle git:(main) ✗ php -S localhost:8080 -t src/api

[Wed Jul 10 15:49:04 2024] PHP 8.3.8 Development Server (http://localhost:8080) started
```

The game should look like

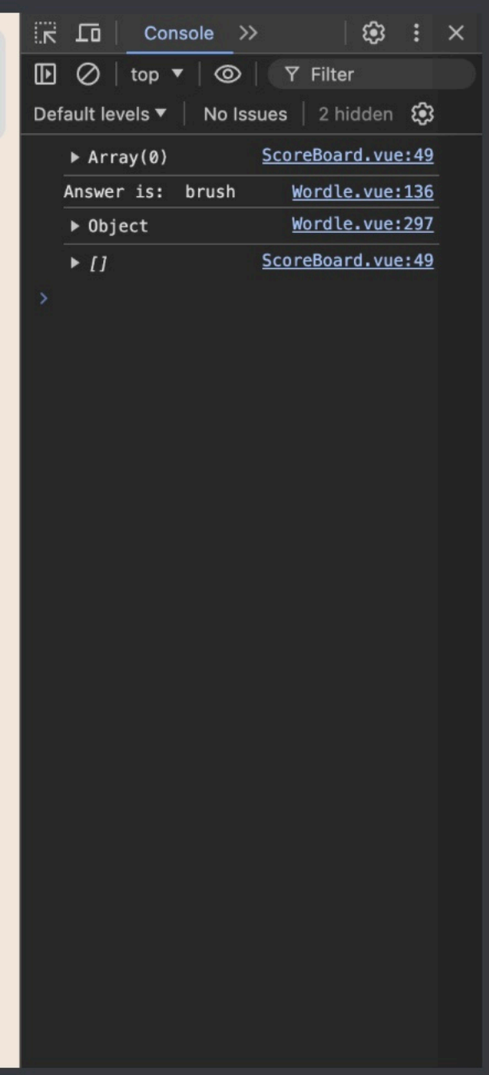


Top 3 Scoreboard

Table

Attempt Number	Attempts

Clean Board

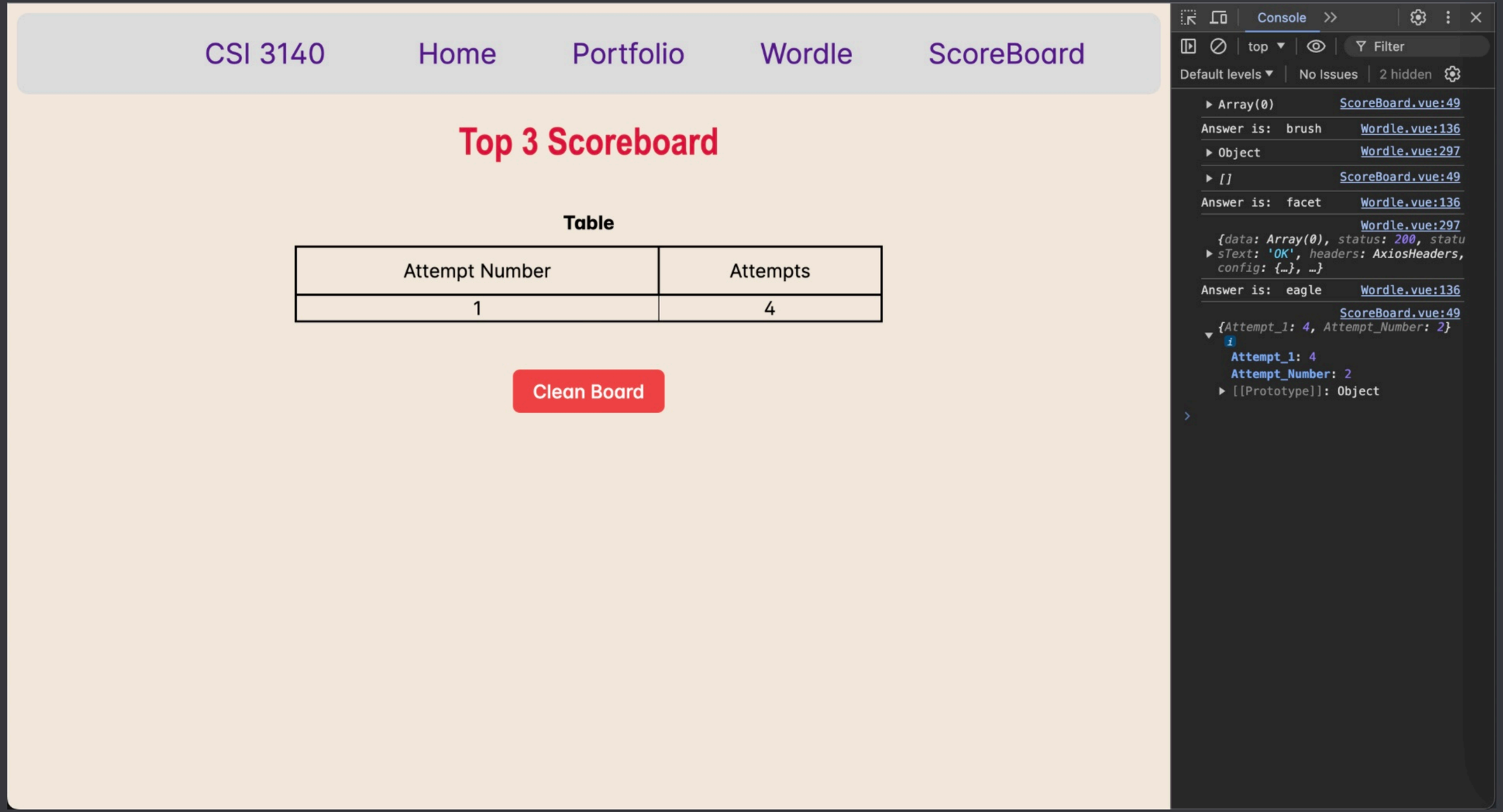


2 Playing

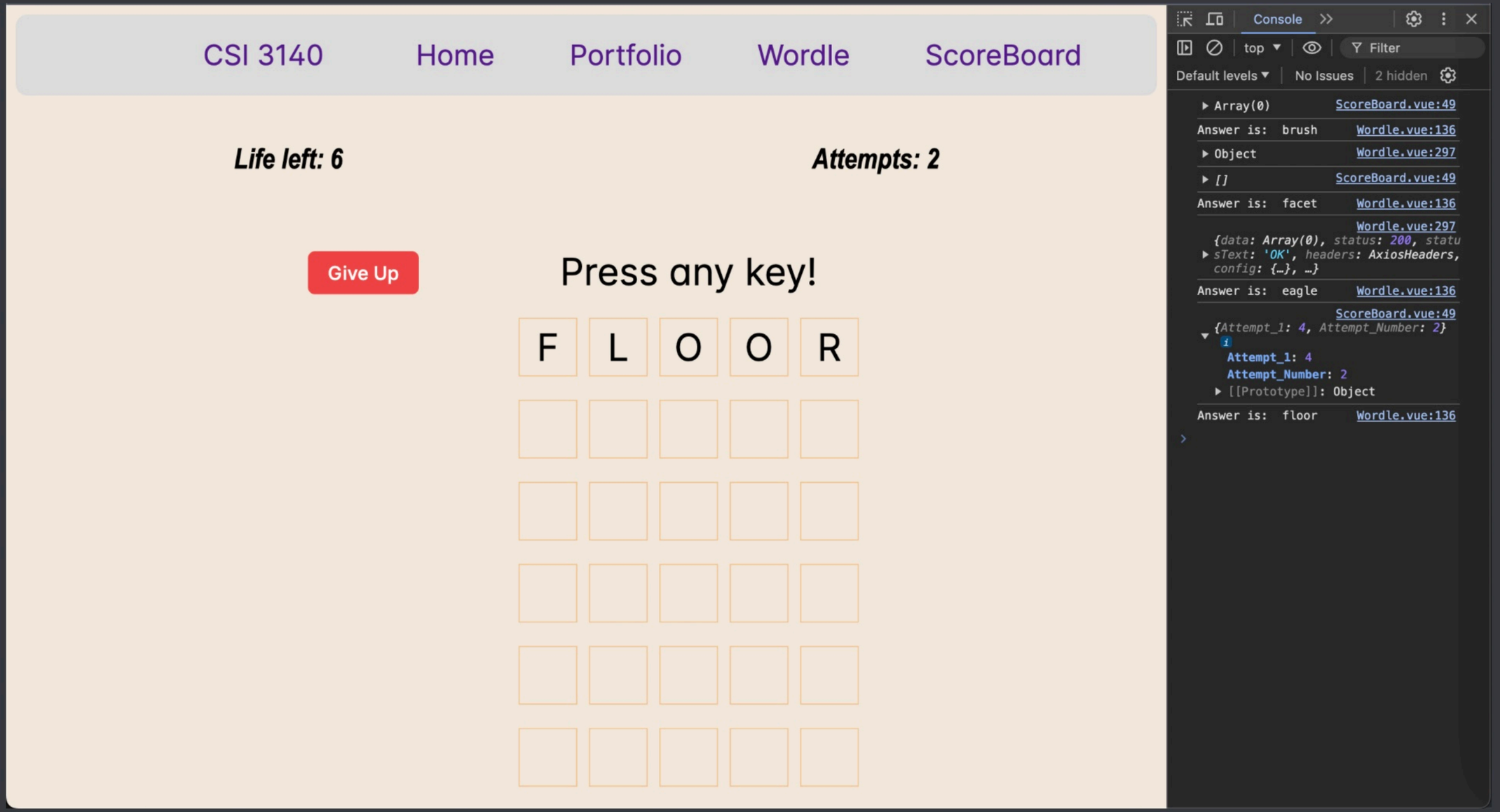
Now start our first trial

You can see on the terminal, the correct answer is: facet, and we have used 3 life, if we enter the correct answer in the 4th line, then our attempt number is 1(1st round), and the attempts is 4.

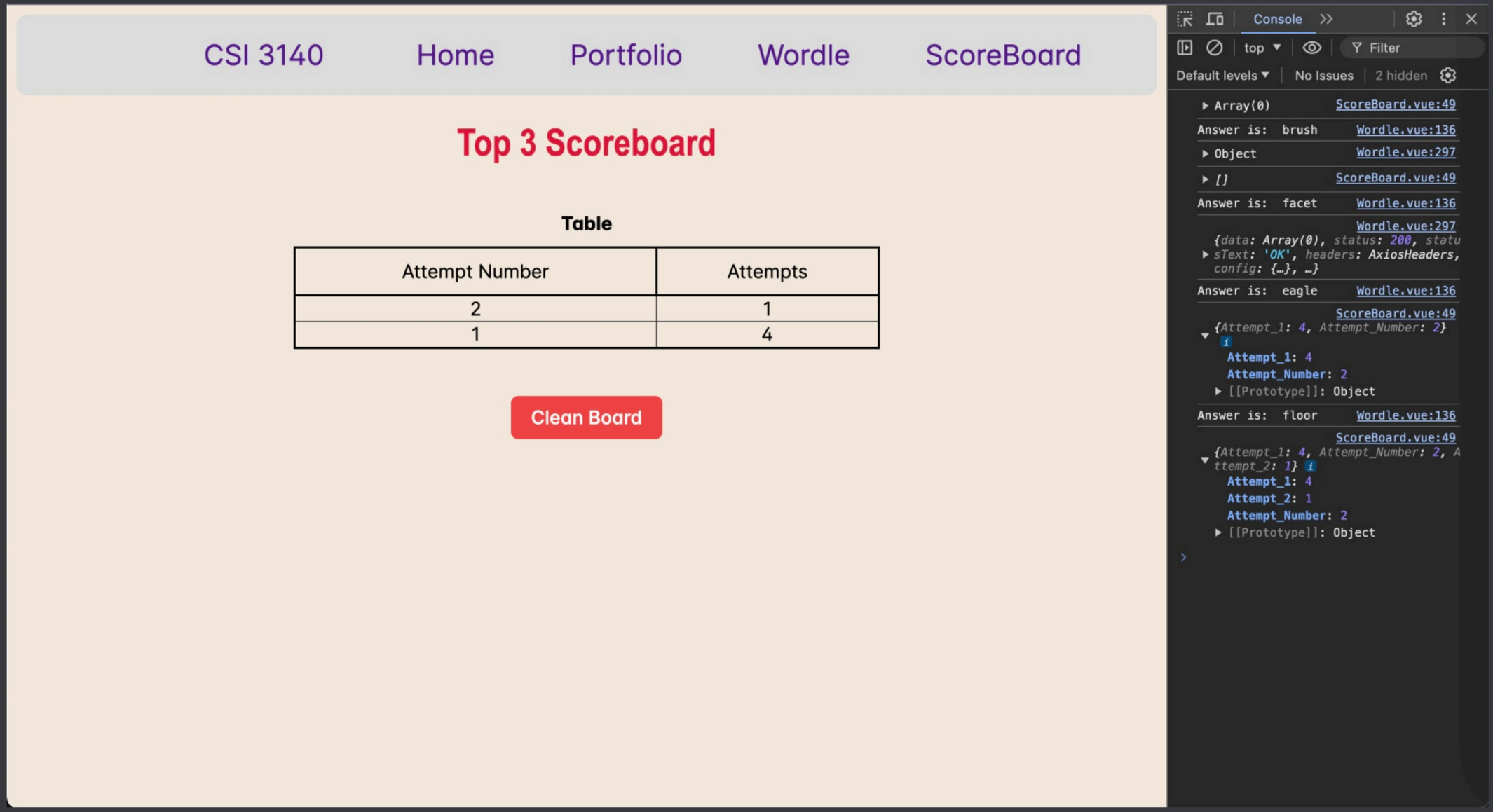
Lets check out scoreboard, you can see the table has been updated, the Attempt number indicates n-th round of player's result, and the right column is the number of attempts player used.


CSI 3140
Home
Portfolio
Wordle
ScoreBoard
Top 3 Scoreboard
Table
Attempt Number
Attempts
1
4
Clean Board
Console
top
Filter
Default levels
No Issues
2 hidden
Array(0) ScoreBoard.vue:49
Answer is: brush Wordle.vue:136
Object Wordle.vue:297
[] ScoreBoard.vue:49
Answer is: facet Wordle.vue:136
Wordle.vue:297
{data: Array(0), status: 200, statu
sText: 'OK', headers: AxiosHeaders,
config: {…}, …}
Answer is: eagle Wordle.vue:136
ScoreBoard.vue:49
{Attempt_1: 4, Attempt_Number: 2}
Attempt_1: 4
Attempt_Number: 2
[[Prototype]]: Object

Let's play again, now attempt number is 2 since we are in our 2nd round. And we are try to enter the correct answer in the first try!



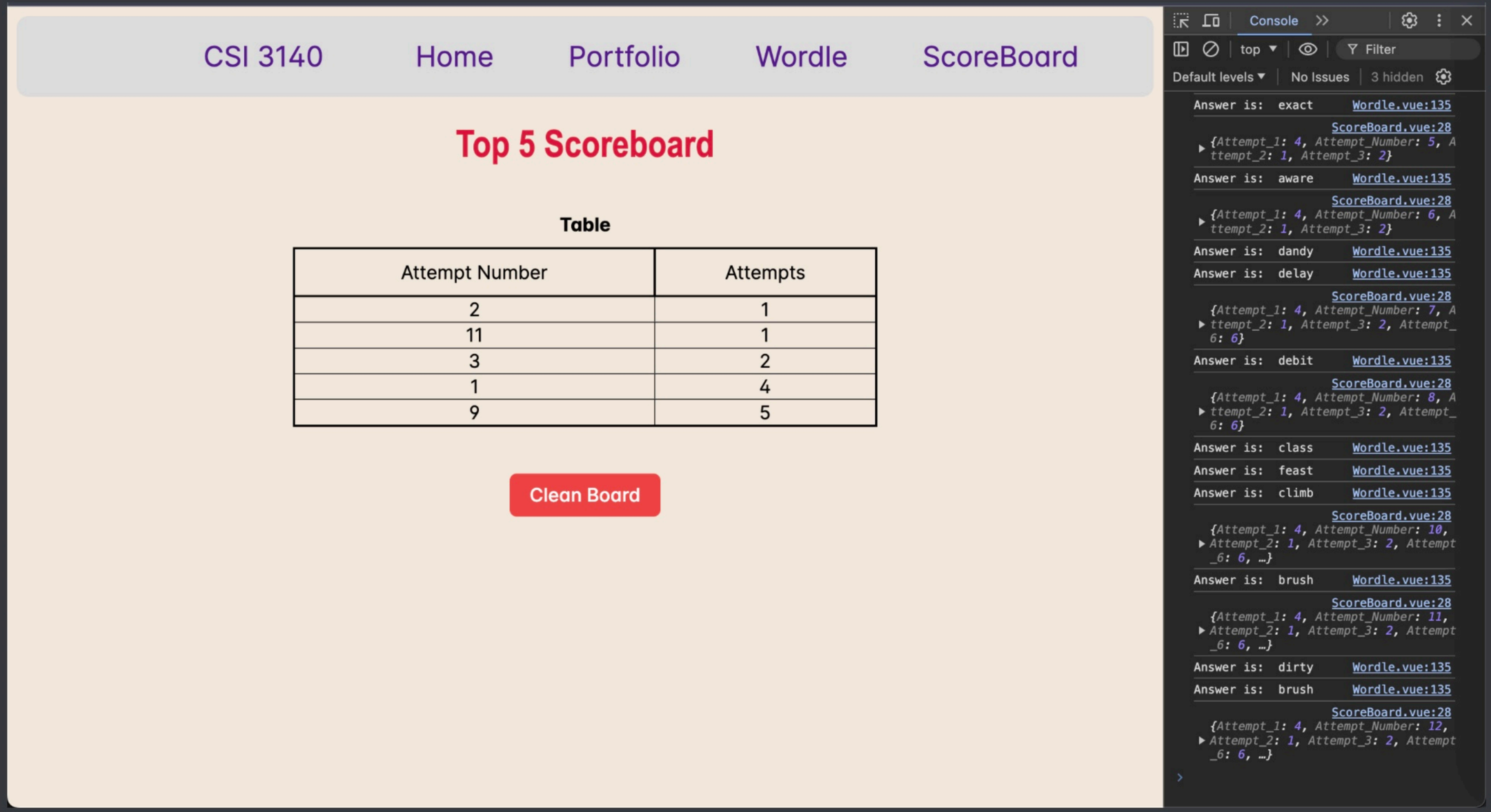
Now on the scoreboard, the table will display the 2nd try first since it has a lower(better) attempts.



> The table should write as: Top 5 Scoreboard. I didn't realize it by far...

3 Keep playing

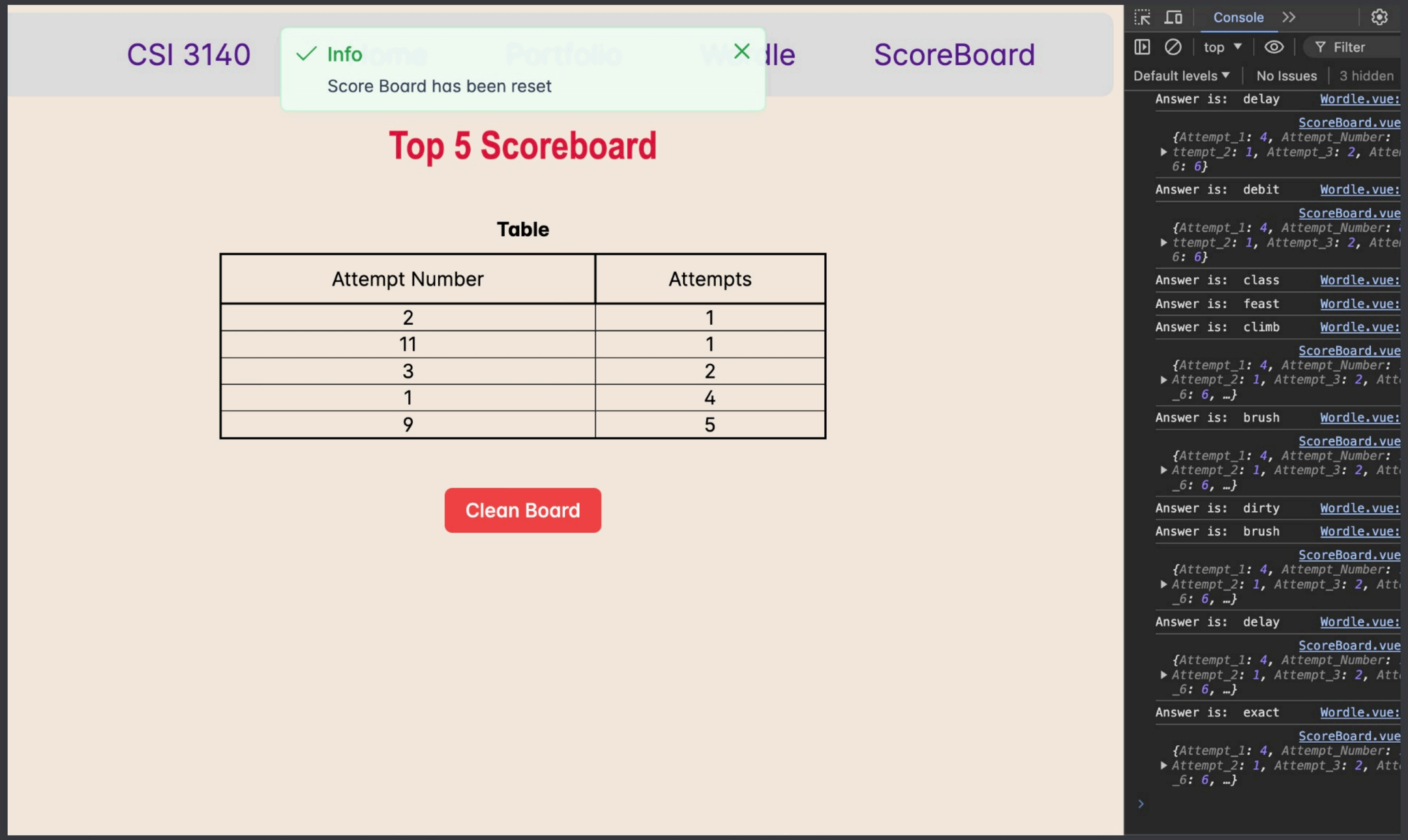
Lets keep try a few time, and see the results.



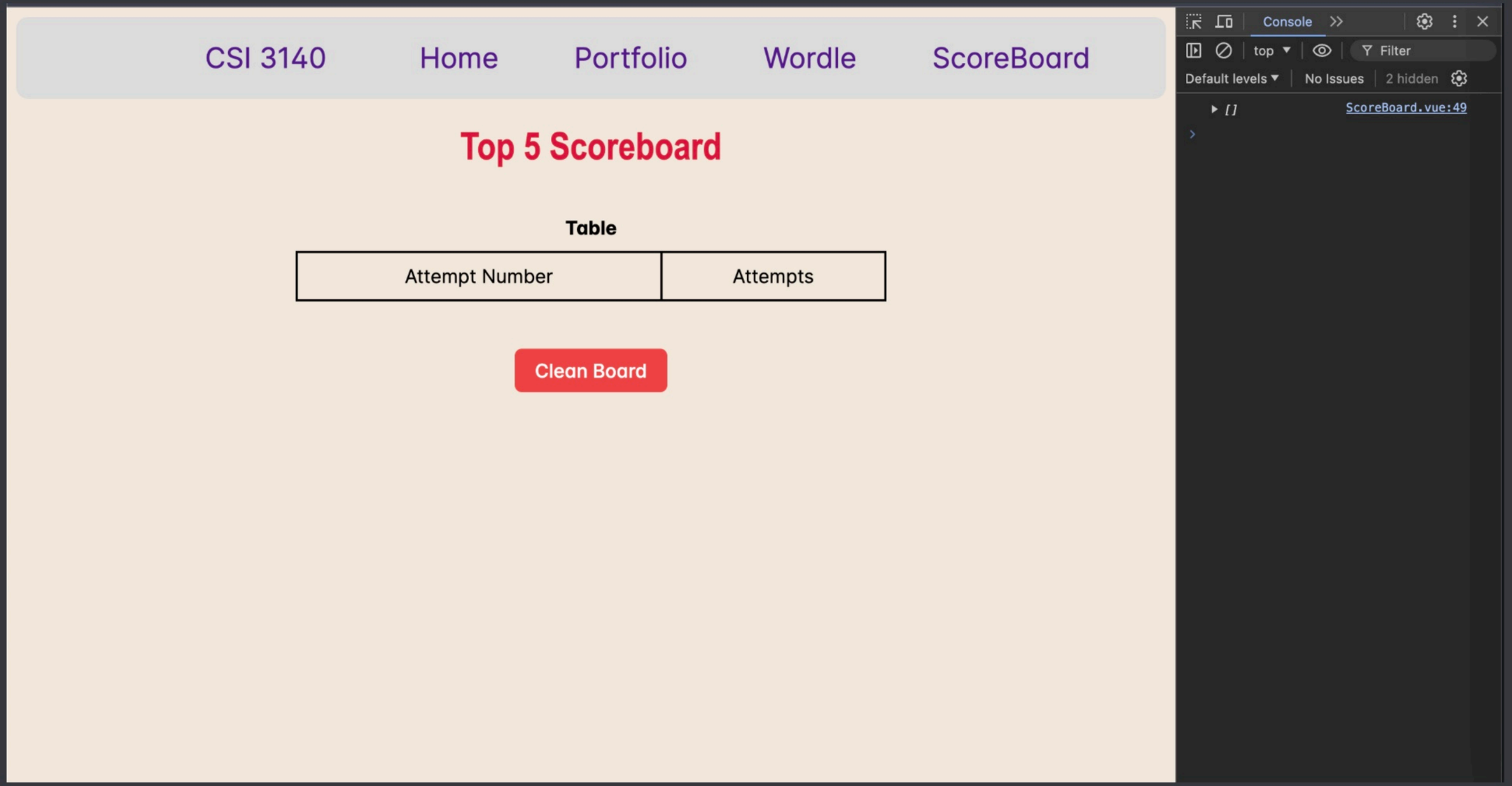
As the table shows, no matter how many attempts we try, it will only store the top 5 scores!

4 Clean Board

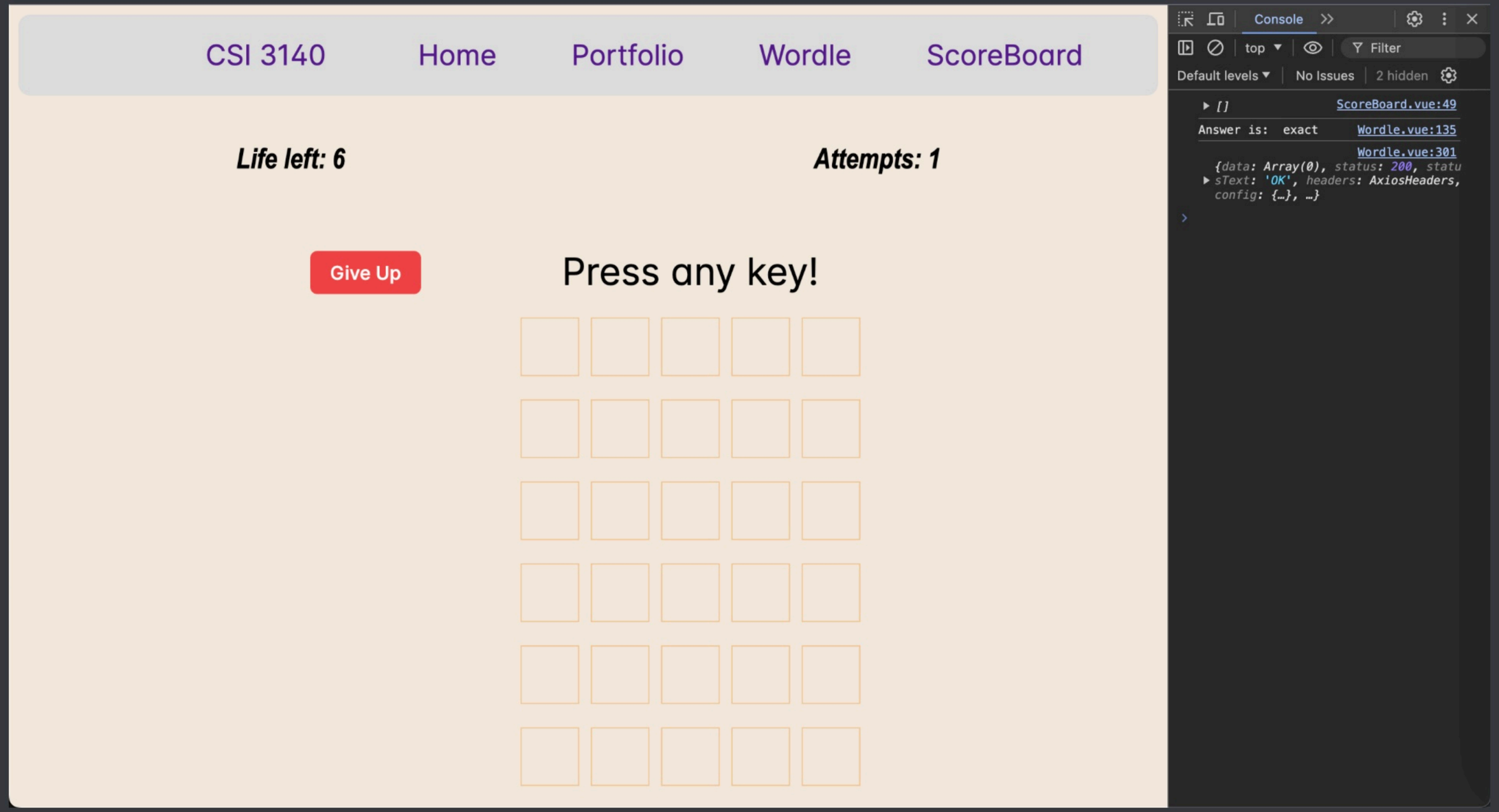
If you want to challenge yourself again, you can press the `clean board` button



After a few seconds:



The board is clean! Lets go back to the game:



The Attempt has been reset to 1, that means the button works!