

Introducción breve:

SuperChess64 es una aplicación de ajedrez en línea permite a los usuarios jugar entre sí o contra una inteligencia artificial. Implementa autenticación con Firebase y gestiona dinámicamente la navegación entre las páginas de login y juego.

Resumen de los tests

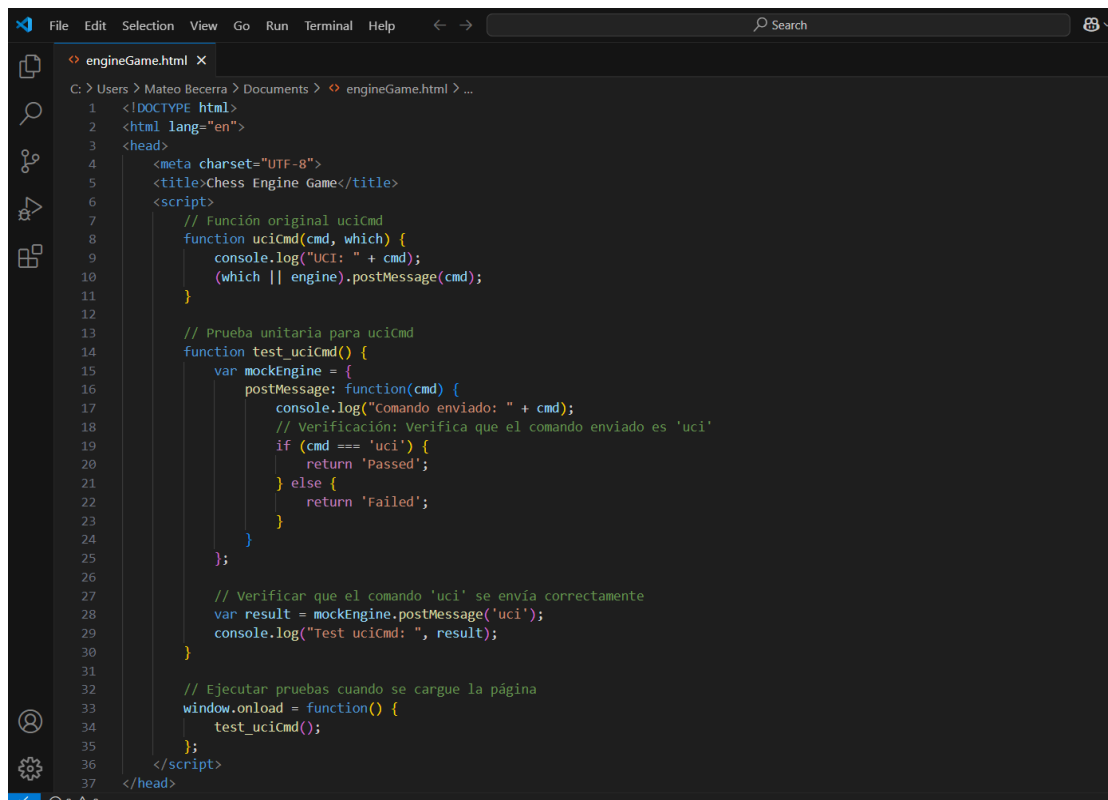
Mateo Becerra Porras

Tipo de prueba realizada: Prueba Unitaria

Descripción breve del componente probado: Se probó la función `uciCmd` que se encarga de enviar comandos al motor de ajedrez utilizando el protocolo UCI (Universal Chess Interface).

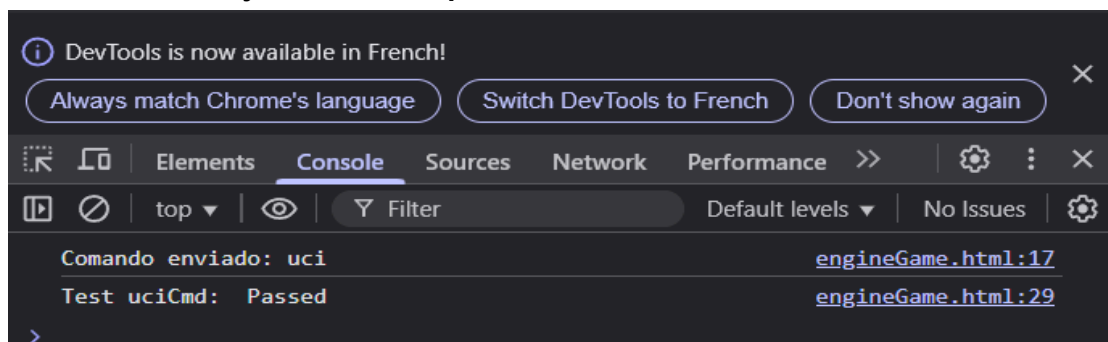
Herramienta o framework usado: No se utilizó un framework específico, se realizó utilizando el entorno del navegador y funciones integradas de JavaScript.

Código de la prueba:



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Chess Engine Game</title>
6   <script>
7     // Función original uciCmd
8     function uciCmd(cmd, which) {
9       console.log("UCI: " + cmd);
10      (which || engine).postMessage(cmd);
11    }
12
13    // Prueba unitaria para uciCmd
14    function test_uciCmd() {
15      var mockEngine = {
16        postMessage: function(cmd) {
17          console.log("Comando enviado: " + cmd);
18          // Verificación: Verifica que el comando enviado es 'uci'
19          if (cmd === 'uci') {
20            return 'Passed';
21          } else {
22            return 'Failed';
23          }
24        }
25      };
26
27      // Verificar que el comando 'uci' se envía correctamente
28      var result = mockEngine.postMessage('uci');
29      console.log("Test uciCmd: ", result);
30    }
31
32    // Ejecutar pruebas cuando se cargue la página
33    window.onload = function() {
34      test_uciCmd();
35    };
36  </script>
37 </head>
```

Resultado de la ejecución de la prueba:



```
DevTools is now available in French!
Always match Chrome's language Switch DevTools to French Don't show again

Elements Console Sources Network Performance >>
top Filter Default levels No Issues

Comando enviado: uci engineGame.html:17
Test uciCmd: Passed engineGame.html:29
```

Integrante del grupo: Roger Fabian Bonilla Caro

Tipo de prueba realizada: Unitaria

Comprobante probado: buscarPartida.js

Descripción del componente probado: La función maneja la cola de emparejamiento escuchando el evento 'buscarPartida' donde si no hay jugadores esperando pone el jugador en espera (encolado) y si encuentra uno se desencola a ambos, se genera un código aleatorio de partida y los jugadores entran a jugar. Para el test, en el código se crean dos sockets que simulan los usuarios en busca de la partida.

Herramienta o framework usado para las pruebas: jest

Código de la prueba

```
let server, ioServer;
let port = 3037;

beforeAll((done) => {
  const httpServer = http.createServer();
  ioServer = new Server(httpServer);
  let waitingPlayer = null;

  ioServer.on("connection", (socket) => {
    socket.on("buscarPartida", () => {
      if (!waitingPlayer) {
        waitingPlayer = socket;
        socket.emit("searching");
      } else {
        let gameCode = Math.random().toString(36).substr(2, 8);
        let colors = Math.random() < 0.5 ? ["white", "black"] : ["black", "white"];
        waitingPlayer.emit("partidaEncontrada", { code: gameCode, color: colors[0] });
        socket.emit("partidaEncontrada", { code: gameCode, color: colors[1] });
        waitingPlayer = null;
      }
    });
  });

  httpServer.listen(port, () => {
    server = httpServer;
    done();
  });
});

afterAll(() => server.close());

test("Emparejamiento de dos jugadores", (done) => {
  const client1 = io('http://localhost:$port');
  const client2 = io('http://localhost:$port');
  let gameData = [];

  const handleMatch = (data) => {
    gameData.push(data);
    if (gameData.length === 2) {
      expect(gameData[0].code).toBe(gameData[1].code);
      expect(gameData[0].color).not.toBe(gameData[1].color);
      client1.disconnect();
      client2.disconnect();
      done();
    }
  };

  client1.on("partidaEncontrada", handleMatch);
  client2.on("partidaEncontrada", handleMatch);
  client1.emit("buscarPartida");
  client2.emit("buscarPartida");
});
```

Resultado de la ejecución de la prueba

```
● > npx jest buscarPartida.test.js
PASS ./buscarPartida.test.js
  ✓ Emparejamiento de dos jugadores (49 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        0.276 s, estimated 1 s
Ran all test suites matching /buscarPartida.test.js/i.
```

Integrante del grupo: Julian Esteban Cadena Rojas

Tipo de prueba realizada: Unitaria

Comprobante probado: connection, disconnect

Descripción del componente probado: Los componentes probados 'connection' y 'disconnect' son en el contexto de Socket.IO, los eventos fundamentales para gestionar la comunicación entre el cliente y el servidor. El evento 'connect' se dispara automáticamente cuando un cliente establece una conexión con el servidor de sockets; por otro lado el evento 'disconnect' se dispara cuando un cliente se desconecta del servidor por cualquier motivo. En el código de testing con Jest, probamos la conexión y desconexión del socket con socket.io-client.

Herramienta o framework usado para las pruebas: jest

Código de la prueba

```
const { Server } = require("socket.io");
const Client = require("socket.io-client");
const gameServer = require("./io"); // Importa tu código

describe("Socket.IO Server", () => {
  let io, serverSocket, clientSocket;

  beforeAll((done) => {
    io = new Server(3001, { cors: { origin: "*" } }); // Servidor en el puerto 3001
    gameServer(io);

    clientSocket = Client("http://localhost:3001");
    io.on("connection", (socket) => {
      serverSocket = socket;
    });

    clientSocket.on("connect", done);
  });

  afterAll(() => {
    io.close();
    clientSocket.close();
  });

  test("El cliente se conecta correctamente", () => {
    expect(clientSocket.connected).toBeTruthy();
  });

  test("El cliente se desconecta correctamente", (done) => {
    clientSocket.on("disconnect", () => {
      expect(clientSocket.connected).toBeFalsy();
      done();
    });

    clientSocket.close(); // Forzamos la desconexión
  });
});
```

Resultado de la ejecución de la prueba

```
• [julian@archlinux Software-Engineering-I]$ npx jest
  console.log
    New socket connection

      at Namespace.log (Proyecto/SocketAndStockFish/chess-site/server/sockets/io.js:5:17)

  console.log
    socket disconnected

      at Socket.log (Proyecto/SocketAndStockFish/chess-site/server/sockets/io.js:52:21)
      at Map.forEach (<anonymous>)
      at Array.map (<anonymous>)

  PASS Proyecto/SocketAndStockFish/chess-site/server/sockets/server.test.js
    Socket.IO Server
      ✓ El cliente se conecta correctamente (1 ms)
      ✓ El cliente se desconecta correctamente (1 ms)

  Test Suites: 1 passed, 1 total
  Tests:       2 passed, 2 total
  Snapshots:   0 total
  Time:        0.189 s
  Ran all test suites.
```

Integrante del grupo: Jacel Thomas Enciso Pinzon

Tipo de prueba realizada: prueba de integración

Comprobante probado: Autenticación y Redirección

Descripción del componente probado: Verifica si el usuario se autentica correctamente a través de Firebase Authentication utilizando el método signInWithPopup con GoogleAuthProvider. Además, comprueba que, tras una autenticación exitosa, el sistema redirige automáticamente al usuario desde la ruta /login a la interfaz principal del juego en /game, asegurando una transición fluida y segura dentro de la aplicación.

Herramienta o framework usado para las pruebas: jest

Código de la prueba

```
const puppeteer = require("puppeteer");

describe("Redirección de login al juego al autenticar", () => {
  let browser;
  let page;

  beforeAll(async () => {
    browser = await puppeteer.launch({ headless: true });
    page = await browser.newPage();
  });

  afterAll(async () => {
    await browser.close();
  });

  test("La página redirige de /login a /game al autenticar correctamente", async () => {
    await page.goto("http://localhost:5173");

    // Simular la redirección
    await page.evaluate(() => {
      window.location.href = "http://localhost:3037";
    });

    // Esperar a que la navegación ocurra
    await page.waitForNavigation();

    // Verificar que la URL cambió
    const currentUrl = page.url();
    expect(currentUrl).toBe("http://localhost:3037/");
  }, 10000);
});
```

```
> proyecto@1.0.0 test
> jest

PASS tests/authGame.test.js
  Redirección de login al juego al autenticar
    ✓ La página redirige de /login a /game al autenticar correctamente (1227 ms)
```

```
Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        3.14 s, estimated 4 s
Ran all test suites.
```

PS C:\Users\USER\Documents\UNAL JTEP\Proyectos\Ingsoft I proyecto\Software-Engineering-I\Proyecto>

Lecciones aprendidas y dificultades:

Aprendimos a utilizar la consola del navegador para ejecutar y depurar nuestras pruebas de manera rápida y efectiva, sin necesidad de configurar un entorno de pruebas completo, así mismo se aprendió de herramientas nuevas de testeo para JavaScript como lo es "jest" y nos enfrentamos a la dificultad de probar entornos asíncronos, como en la simulación de navegación y autenticación con Firebase o pruebas en tiempo real con Socket.IO.