

SERGIO GARCÍA JIMÉNEZ

Jaque Mate



TUTOR: RICARDO VALENCIA

PROYECTO 2º DAW 23-24

IES MATEMÁTICO PUIG ADAM
GETAFE

Resumen del proyecto

Este proyecto está enfocado en mejorar las habilidades de los jugadores novatos, principiantes e iniciados en el ajedrez, especialmente en la fase de apertura, también mejorando su resolución de problemas de ajedrez y la posibilidad de practicar contra una IA para poner a prueba lo que han aprendido.

Para ello, he analizado tanto mi propia experiencia como jugador, partidas de jugadores principiantes, foros de ajedrez online, creadores de contenido orientados al ajedrez y libros.

El objetivo principal es dar una serie de herramientas a estos jugadores para que puedan practicar, aprender sus aperturas personalizadas, mediante la repetición y el ensayo y error, dejando total libertad al jugador de delimitar el número de jugadas a practicar, el tiempo necesario, pudiendo repetir y repasar las mismas cuanto desee, así como guardar sus propias configuraciones a su gusto.

Al ser una herramienta que yo mismo he utilizado para aprender, asentar e interiorizar algunas de mis aperturas, he comprobado la eficacia y la sencillez de la herramienta, que además no es una herramienta que abunde en internet o, las que he encontrado, no acaban de dar todas las facilidades necesarias o carecen de algunas herramientas importantes.

Palabras clave

Ajedrez

Chess

Aperturas ajedrez

Chess openings

Tácticas ajedrez

ÍNDICE

1. Objetivo y alcance del proyecto.....	7
1.1 Introducción.....	7
1.2 Alcance.....	9
1.3 Conceptos clave y aclaraciones técnicas sobre ajedrez.....	10
Mi idea es generar una página web que nos permita guardar varias aperturas como favoritas, siendo fundamental que podrás importar varios movimientos para crear tus propias aperturas o movimientos que quieras estudiar, y guardarlos, eligiendo los nombres y cada paso de forma totalmente personalizada.....	10
1.4 Excepciones.....	12
2. Primeros bocetos y diseños.....	13
2.1 Borradores de páginas web.....	13
2.2 Diseño de piezas.....	15
3. Antecedentes.....	17
3.1 Resumen.....	17
3.2 Tecnologías utilizadas.....	18
5. Análisis de Alternativas y Justificación de las Decisiones Adoptadas.....	20
5.1. Análisis de alternativas para el aprendizaje de ajedrez:.....	20
5.2. Análisis de alternativas para el desarrollo web.....	22
6. Estructura de la base de datos.....	23
7. Gestión de usuarios, administración y sesiones.....	24
8. Funcionalidades más relevantes de la aplicación.....	32
8.1 Datos curiosos.....	32
8.2 Diseños de piezas personalizados.....	33
8.3 Tablero en la página de inicio.....	34
8.4 Problemas.....	35
8.5 Bot.....	38
8.6 Aperturas.....	42
9. Conclusiones.....	51
9.1 Diseño y accesibilidad.....	51
9.2 Uso intuitivo.....	53
9.3 Funcionalidad.....	53
10. Programación temporal de los trabajos a realizar.....	54
11. Bibliografía, fuentes y otros recursos.....	55

1. Objetivo y alcance del proyecto

1.1 Introducción

En los últimos meses me he iniciado en el mundo del ajedrez, y una de las primeras necesidades que he tenido a la hora de tomarme más en serio el juego es la de estudiar. Hay muchas variables y posibilidades en ajedrez, como dato curioso, el número de partidas de ajedrez únicas posibles es mucho mayor que el número de electrones en el universo (10^{120} partidas contra 10^{79} electrones). Por tanto, en muchas ocasiones la visualización y memorización de patrones y posiciones es vital en el juego.

Una de las partes más importantes del juego es la fase de apertura, que consiste en los primeros movimientos de la partida. Hay algunos principios básicos que sirven como reglas generales u orientativas, algunas como:

- Abrir el juego con uno de los peones centrales.
- Desarrollar primero los caballos y luego los alfiles.
- Intentar controlar el centro del tablero.
- No mover la misma pieza más de una vez durante la apertura.

Y sin embargo, hay aperturas que rompen con estos principios por diversos motivos.

Al contrario de lo que mucha gente piensa, en ajedrez hay muchos movimientos llamados teóricos o “de libro”, esto significa lo que parece, que muchas veces son memorizados por el jugador, y estos movimientos se dan en la fase de apertura, en la fase intermedia de la partida es donde hay más “estilo libre” y en la fase final nuevamente, volvemos a muchos patrones o jugadas que se repiten y muchos jugadores saben de memoria.

Ahora bien, hay muchísimas aperturas en ajedrez, y a su vez, estas tienen muchísimas variantes, ¿cómo vas a memorizar cientos de miles de jugadas? Bueno, ni los grandes maestros de ajedrez pueden, lo habitual es especializarse en unas cuantas aperturas, y en unas cuantas defensas (esto es, aperturas desde el lado de las negras, ya que al jugar primero las blancas, normalmente se habla de aperturas para blancas y defensa para negras).

En mi camino de aprendizaje del juego, unos de los primeros pasos que tomé fue embarcarme en la búsqueda de un repertorio de aperturas y defensas para aplicar en mis propias partidas, y al hacerlo me di cuenta de lo fácil que era que las aperturas se te fueran de las manos, no recordarás movimientos clave, o los hicieras en el momento que no era el correcto, y una mala apertura normalmente condiciona el resto de la partida negativamente.

Por ello me surgió la idea de poder guardar ciertas aperturas que quisieras aprender o estudiar, y poder practicarlas con una guía que te fuera indicando si las jugadas que hacías eran las correctas o no, y aprender por repetición. Y eso es Jaque Mate, un entrenador de aperturas para jugadores, principiantes, intermedios o veteranos, de ajedrez.

1.2 Alcance

Si bien a priori la idea principal giraba en torno a la fase de apertura, a lo largo del desarrollo del proyecto he añadido algunas herramientas útiles adicionales para el entrenamiento de los usuarios. De esta manera, he acabado añadiendo una lista de problemas de uno o dos movimientos con soluciones únicas, para poder practicar tácticas y encontrar los mejores movimientos en cada posición.

Además, he añadido la posibilidad de poder jugar una partida en un solo dispositivo en la página principal, y en caso de no tener con quien jugar, una pequeña inteligencia artificial, que si bien no está optimizada, dará bastante competencia a jugadores novatos y principiantes y podrá servir para practicar algunas tácticas o aperturas a jugadores más experimentados.

Dado que el ajedrez no requiere de ningún idioma y los menús y contenido textual es relativamente sencillo, veo factible el hecho de poder simplemente traducir el proyecto para que pueda ser utilizado también por gente independientemente de que no hable castellano.

Estuve valorando la posibilidad de implementar un analizador de posiciones y jugadas, pero, debido a la complejidad y gran trabajo que requería, acabé descartándola, al menos de momento, ya que podría constituir casi un proyecto completo por sí solo. No obstante, no es una idea que no baraje constituir a lo largo de los próximos meses para la ampliación de la aplicación, ya que sería de gran utilidad.

1.3 Conceptos clave y aclaraciones técnicas sobre ajedrez

Mi idea es generar una página web que nos permita guardar varias aperturas como favoritas, siendo fundamental que podrás importar varios movimientos para crear tus propias aperturas o movimientos que quieras estudiar, y guardarlos, eligiendo los nombres y cada paso de forma totalmente personalizada.

Esto se consigue con la tecnología **PGN** que corresponde a las siglas en inglés Portable Game Notation (Notación portátil de juego en español). PGN es un formato de computadora para grabar partidas de ajedrez, tanto los movimientos como la información relacionada, es decir, si yo hago una partida de ajedrez, y muevo por ejemplo, el peón blanco a una casilla, el peón negro a otra, el caballo blanco a otra, y el alfil negro a otra, la mayoría de aplicaciones de ajedrez online nos permiten exportar la partida en formato PGN, nuestro ejemplo se vería tal que así:

1. e4 e5 2. Nc3 Bc5

Ahora bien, ¿qué significa o cómo se interpreta esto?

Turno 1.

e4 Indica la casilla (E4) donde se mueve la pieza, siempre el primer movimiento es del jugador blanco, si no pone ninguna letra mayúscula antes de la casilla, es un peón el que mueve

e5 Movimiento del jugador negro, otro peón, a E5

Turno 2.

Nc3 N de Knight, caballo, se denomina N y no K ya que la K corresponde a movimientos del rey, y la casilla donde se mueve, C3.

Bc5 B de Bishop, Alfil, del jugador negro, que se mueve a C5.

Como podemos ver, es un formato muy sencillo de entender, apenas ocupa espacio, ya que es texto plano, y sin embargo, podemos guardar cientos de movimientos y exportarlos en un archivo de texto. Esto nos permite recrear esos movimientos en un analizador de partidas o similares posteriormente, o, en este caso, en mi entrenador de aperturas.

El proceso sería el siguiente

1. Coges la lista de movimientos de tu archivo PGN, ya sea una partida jugada por ti, o una apertura que quieres practicar (hay PGNs totalmente gratuitos en Internet) y la exportas al entrenador.
2. Le dices al entrenador si quieres practicar como blancas o cómo negras.
3. El entrenador te va preguntando a cada movimiento, cuál sería el movimiento a realizar
4. Si el movimiento que haces coincide con el del PGN, es que has hecho el movimiento correcto que quieres aprender, te felicita y pasa al siguiente movimiento
5. Si lo haces mal te lo indica y puedes volver a intentar el movimiento correcto

Como detalle adicional, indicar que en el PGN también pueden almacenarse comentarios dentro de las jugadas con el formato {}.

Ne5 {"Aquí se mueve el caballo porque si no pueden hacernos jaque"}, por ejemplo

Y también datos de la partida en sí, como el lugar, el evento, la fecha...

[Event "FIDE World Championship"]

[Site "New York, NY USA"]

[Date "2016.11.11"]

[Round "1"]

[White "Carlsen, Magnus"]

[Black "Karjakin, Sergey"]

[Result "1/2-1/2"]

1.4 Excepciones

Si bien mi proyecto cuenta con el explorador de aperturas, que es una herramienta poco extendida entre la mayoría de aplicaciones para el estudio del ajedrez, hay algunos puntos que no están implementados por el momento, que me parece importante reseñar.

- No se incluye un análisis avanzado de partidas por medio de un motor de ajedrez, si bien es una funcionalidad que se implementará en el futuro.
- No se ha valorado por el momento la posibilidad de poder jugar en línea con otros jugadores.

2. Primeros bocetos y diseños

2.1 Borradores de páginas web

index.html



JaqueMate



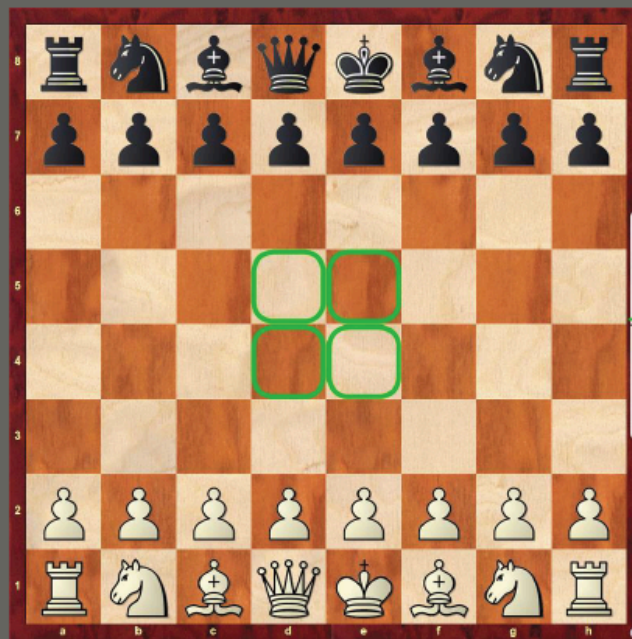
Problemas



Aperturas



Análisis



Dato
Curioso

Lorem Ipsum es simplemente el texto de relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de las industrias desde el año



 Problemas

 Aperturas

 Análisis

Apertura 1

Introducción


Variante 1

Variante 2

Variante 3

Apertura 1



 Mis aperturas

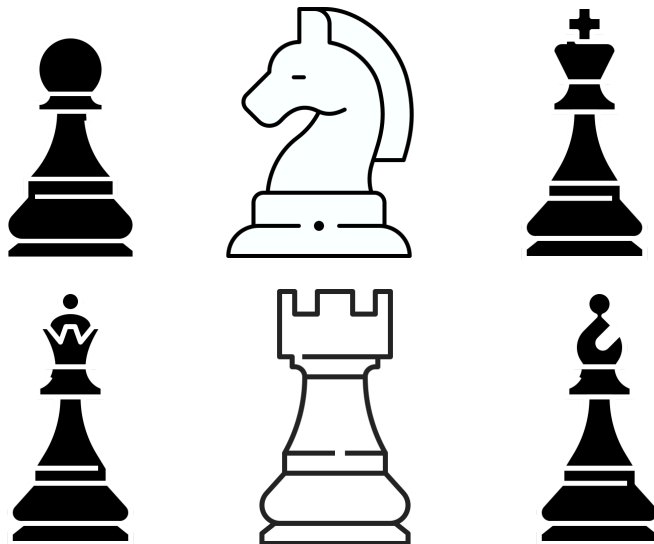
 Nueva apertura

 Guardar favorita

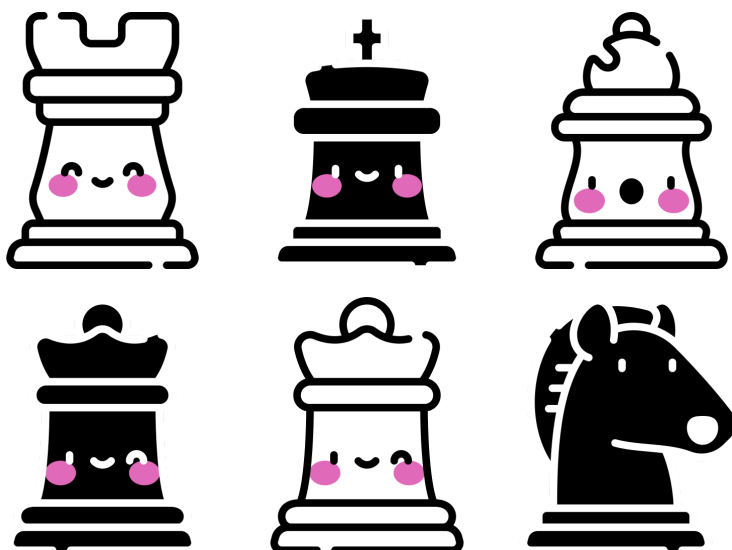


2.2 Diseño de piezas

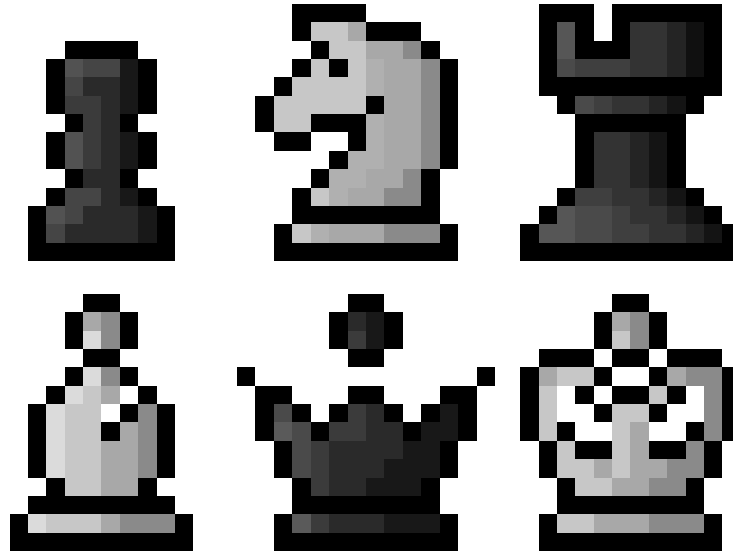
Diseño moderno



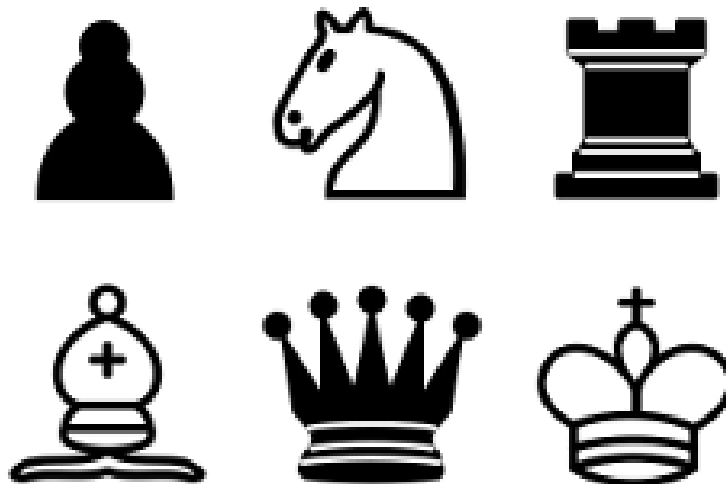
Diseño kawaii



Diseño retro



Diseño clásico



El diseño clásico es el que utilizan la mayoría de juegos online de ajedrez, son piezas que proporciona Wikipedia, pero quería añadir algo de personalización y variedad, por eso busqué algunos diseños alternativos y fuera de lo común.

3. Antecedentes

3.1 Resumen

El punto de partida del proyecto se sustenta en la carencia de herramientas específicas que se centren en la práctica de aperturas de ajedrez. Existen aplicaciones y plataformas que ofrecen análisis de partidas completas, problemas tácticos y partidas contra motores de ajedrez avanzados, pero pocas se centran en entrenar específicamente la fase de apertura.

Si bien existe contenido y material como vídeos, explicaciones y algunos ejercicios, no dan en mi opinión suficientes ejercicios prácticos y además no permiten la personalización y flexibilidad que permite mi herramienta, que se centra en la repetición. Al fin y al cabo, una apertura tiene muchas variantes y son muchos movimientos, cuya memorización es una tarea muy pesada y ardua, en cambio mediante la repetición de los mismos movimientos con una herramienta de ensayo y error, este aprendizaje será más fácil, más divertido y sobre todo más rápido, ya que acabaremos por reconocer y recordar los patrones de una apertura concreta.

La solución que planteo me parece tan efectiva porque conlleva un enfoque especializado y es accesible para todo tipo de jugadores, ya que cualquiera independientemente de su nivel va a encontrar esta herramienta útil si quiere aprender una nueva apertura o reforzar su conocimiento sobre una que ya conoce.

3.2 Tecnologías utilizadas

HTML 5

El estándar para la creación de páginas web, universal.

CSS

Idéntico al anterior, es un estándar universal para el diseño de una página web.

JavaScript

Para añadir las funcionalidades necesarias en la web.

APIs de ajedrez

En mi caso para el diseño gráfico del tablero y piezas, utilizaré la biblioteca chessboard.js, mientras que usaré chess.js para las reglas y funcionamiento del juego

JQuery

Para la utilización de las API de ajedrez de chessboard.js

PGN

Como he explicado en el punto anterior, es un estándar para cualquier aplicación de ajedrez, necesaria para la importación, exportación y análisis de partidas.

FEN

Otro tipo de notación de ajedrez, ligeramente diferente al PGN y que explicaré más adelante.

En el siguiente punto, se examinará con mayor profundidad el porqué utilizo cada tecnología.

4. Arquitectura web

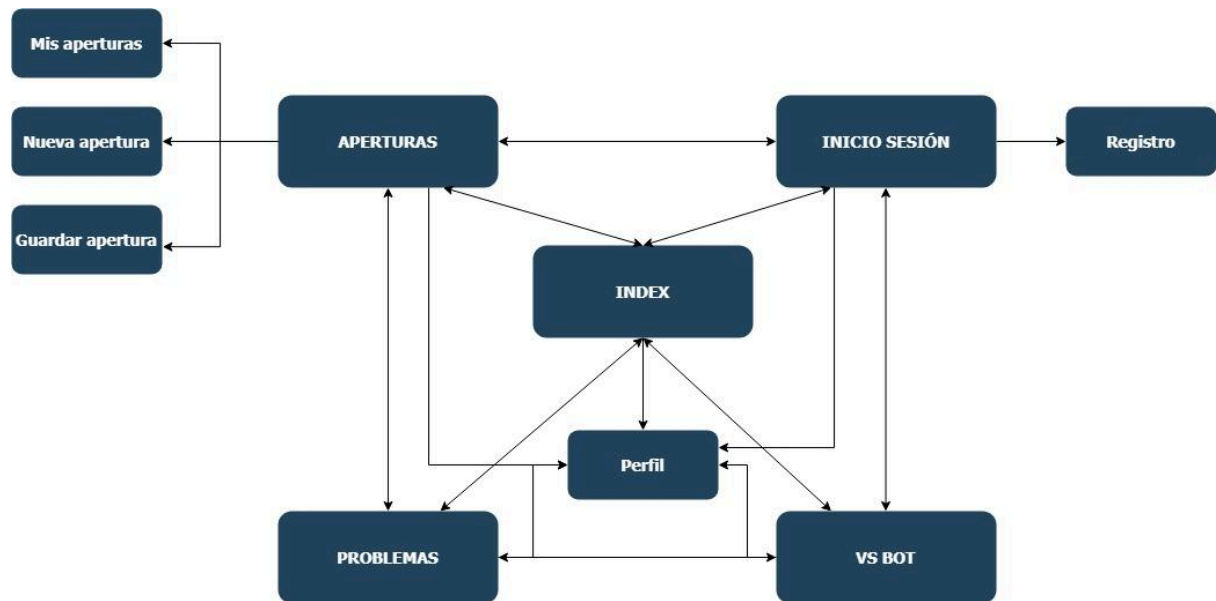
La arquitectura de la aplicación web sigue un modelo Cliente-Servidor, donde el navegador web (cliente) interactúa con la interfaz de usuario, y el servidor es donde se procesan las solicitudes del cliente, se accede a la base de datos y se envían las respuestas de vuelta al cliente, utilizando para ello PHP y gestionando la BBDD con MySQL Workbench y MySQL Server.

En el lado del cliente, utilizo HTML para estructurar el contenido de la web, CSS para el diseño y presentación, y JavaScript (incluyendo jQuery para funcionalidades como las ventanas modales) para la lógica de la aplicación, como manejar eventos del usuario y realizar peticiones de datos al servidor.

En el lado del servidor, se utiliza PHP para interactuar con la base de datos y procesar las peticiones de autenticación de usuarios, extracción de datos importantes en la BBDD, almacenamiento de aperturas, encriptación de contraseñas, recuperación de datos almacenados, etc.

La aplicación sigue un patrón Modelo-Vista-Controlador (MVC): el PHP actúa como el controlador que procesa las solicitudes del cliente, accede a los datos del modelo (la base de datos), y envía la respuesta a la vista (el HTML) que se presenta al usuario en el navegador.

DIAGRAMA DE LA RELACIÓN DE PÁGINAS



5. Análisis de Alternativas y Justificación de las Decisiones Adoptadas

5.1. Análisis de alternativas para el aprendizaje de ajedrez:

1. Uso de un motor de ajedrez avanzado de análisis, como Stockfish:

Stockfish es un motor de ajedrez, el más potente hasta el momento y también el más popular. Teniendo distintos niveles de profundidad, calcula los mejores movimientos en cada posición siendo capaz de ver más allá de 10 y 15 jugadas en adelante.

Sin embargo hay varias razones por las que Stockfish no me parece una herramienta idónea para la apertura. Principalmente, porque calcula en base a futuras jugadas que los jugadores principiantes son incapaces de visualizar, y en muchos casos, ni los veteranos, mientras que el fundamento de la apertura es tener una base de partida sólida, no necesariamente tener que ganar la partida en 20 movimientos. Además, hay muchas aperturas que tienen ciertas trampas, oportunidades o situaciones incómodas para el rival, que si bien Stockfish no considera ventajosas para el jugador que realiza la apertura, porque puede calcular la mejor respuesta, frente a un humano normal sin su capacidad de análisis, y especialmente en posiciones poco comunes o poco conocidas, puede suponer una enorme ventaja que provoque que el jugador rival haga movimientos fuera de lugar o cometa más errores al no conocer los trucos y motivaciones detrás de nuestra apertura. Por tanto el factor humano me parece clave en la fase de apertura y por este motivo un motor de análisis queda descartado. Además de ello, la complejidad de la integración del mismo, y el consumo de recursos que podría afectar al rendimiento de equipos menos potentes, no hace más que reforzar esta decisión.

2. Práctica de problemas existentes:

Si bien la visualización de patrones mediante repetición es una de las claves para mejorar en ajedrez, el problema de estos es que normalmente hay una variedad muy amplia, problemas para dar mate, o ganar piezas, o forzar movimientos malos para el rival. Y aunque en algunas partes hay problemas centrados en aperturas, normalmente no suele haber mucha variedad y estos suelen ser de aperturas específicas, sobre todo las más conocidas, lo que limita la personalización por parte del usuario, cosa que yo mismo he experimentado al buscar ejercicios para aperturas que no son tan populares cuando quería practicarlas. Por este motivo, aunque he incluido problemas en mi aplicación, para el aprendizaje de aperturas no son la herramienta principal al no tener esa personalización que me parece esencial, si bien la he incluido como complemento para el desarrollo de otras habilidades, visualización y tácticas.

3. Contenido multimedia, tutoriales o bases de datos de aperturas

Estas herramientas me parecen útiles y lógicamente son parte del proceso de aprendizaje de una apertura. Son un complemento para familiarizarte con la apertura, comprender como funciona y entender por qué funciona de esa manera o cuales son sus puntos principales. Sin embargo, a la hora de interiorizar los movimientos clave y recordar que movimiento hacer en cada posición, el hecho de verlo o estudiarlo hace mucho menos eficiente el proceso de aprendizaje que la practica. Por ello, son herramientas complementarias idóneas para empezar a aprender una apertura pero no para asentarla y consolidarla.

Examinadas las principales herramientas disponibles en la mayoría de páginas de ajedrez, podemos comprobar que una herramienta específica puede ser de tremenda utilidad para el propósito de aprender una nueva apertura.

5.2. Análisis de alternativas para el desarrollo web

Realmente, la variedad de lenguajes de programación, posibilidades en BBDD y servidores, IDEs, etc. es muy amplia, por lo que aquí no me explayaré tanto.

El IDE que he utilizado ha sido Visual Studio Code, un estándar totalmente conocido, sencillo, práctico y con el que hemos trabajado a lo largo del curso, además de ser más de mi agrado que otros que he probado como NetBeans o Eclipse.

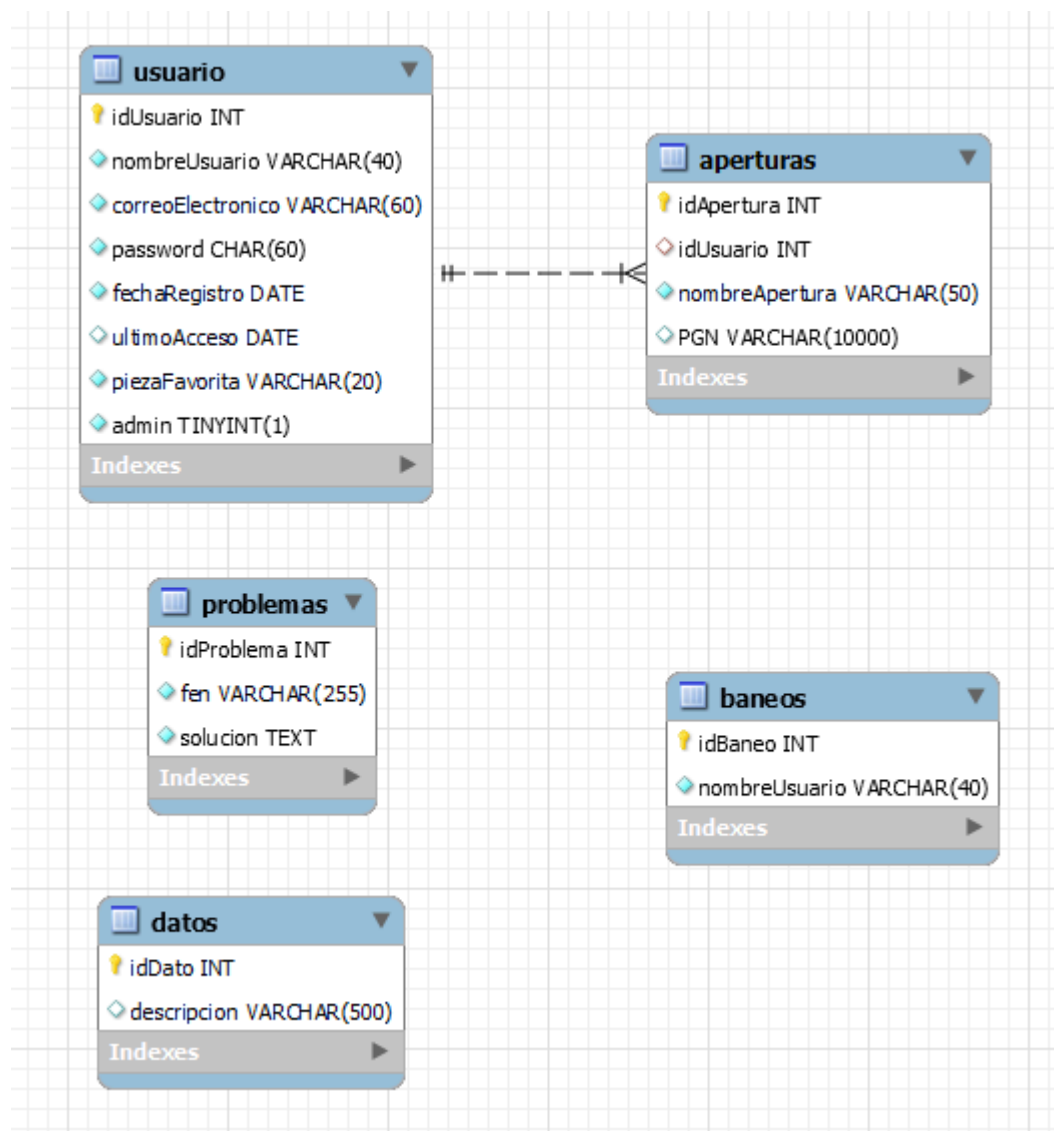
Lógicamente al ser una aplicación web, el uso del estándar universal HTML5 se da casi por sentado.

Debido a que es una herramienta con la que estoy familiarizado e iba a necesitar un gestor de base de datos, he utilizado MySQL Workbench y MySQL Server al tener una interfaz gráfica sencilla y ya conocida, y una sintaxis que conozco más en profundidad que otros servidores que utilizan SQL. El servidor por el momento funciona de manera local.

Además, para la implementación de las funcionalidades de ajedrez iba a necesitar Javascript, por lo que su implementación en el proyecto también era necesaria, además de servirme para funcionalidades como uso de botones u otras igualmente útiles.

Para el manejo de las funcionalidades y datos de la BBDD y su relación con mi aplicación web, el lenguaje utilizado es PHP, un lenguaje aprendido a lo largo del año que encaja perfectamente con las necesidades de mi proyecto.

6. Estructura de la base de datos



La BBDD es bastante sencilla. En realidad la mayoría de tablas son independientes, a excepción de aperturas que dependen del usuario. Los datos y problemas no requieren de otra información y el método de baneos, no necesita estar conectado con usuarios para funcionar.

7. Gestión de usuarios, administración y sesiones

Los usuarios se gestionan a través de la tabla usuario en la BBDD, y el registro de los mismos se realiza en la página registro.php con un formulario.

```
<!-- Formulario que recoge los datos para el registro -->
<form id="formRegistro" method="post" action="registroRespuesta.php">
  Correo electrónico:<input type="email" name="userMail" class="input">
  <span id="email-status"></span>
  Usuario:<input type="text" name="user" class="input">
  <span id="user-status"></span>
  Contraseña:<input type="password" name="pwd" class="input">
  Repite la contraseña:<input type="password" name="pwd2" class="input">
  <input id="registrarse" class="botonPerfil" type="submit" value="Registrarse">
</form>
```

Una vez enviados los datos los proceso en la página registroRespuesta.php que recibe los datos, comprueba todos los posibles fallos y ejecuta el registro o no según las condiciones. Primero compruebo que todos los campos están cumplimentados, y paso la función htmlspecialchars por ellos para evitar posibles errores con caracteres extraños o no reconocidos en la inserción en la BBDD. Además obtengo el día actual para posteriormente guardar la fecha de registro.

```
// Compruebo que he recibido todos los datos del formulario de registro
if (isset($_POST['userMail']) && isset($_POST['user']) && isset($_POST['pwd']) && isset($_POST['pwd2'])) {
  // Recibo y filtro los datos del formulario
  $email = htmlspecialchars($_POST['userMail']);
  $user = htmlspecialchars($_POST['user']);
  $pwd = htmlspecialchars($_POST['pwd']);
  $pwd2 = htmlspecialchars($_POST['pwd2']);
  // Obtengo el día de hoy
  $fechaActual = date('Y-m-d H:i:s');
```

Lo primero que compruebo es si las contraseñas coinciden. Si es así, paso importante, las encripto, de forma que la inserción en la BBDD sea un texto encriptado para que la privacidad del usuario esté protegida, y para que en caso de fuga de información, no se filtre información sensible.

```
// Verifico que las contraseñas coincidan
if ($pwd !== $pwd2) {
    $msg = "Las contraseñas no coinciden.";
} else {
    // Encripto la contraseña
    $hashedPwd = password_hash($pwd, PASSWORD_DEFAULT);
}
```

Posteriormente, compruebo que no haya un usuario registrado con el mismo nombre que el introducido en el formulario, así como el correo electrónico. También, añado un error genérico por si hay cualquier problema con la BBDD (conexiones, queries, etc.)

```
// Verificar si el nombre de usuario ya está en uso
$sql = "SELECT * FROM Usuario WHERE nombreUsuario = ?";
$stmt = $conn->prepare($sql);
// Error genérico en caso de fallo de la conexión o la BBDD
if (!$stmt) {
    die("Ha habido un error durante el registro, inténtalo más tarde.");
}
$stmt->bind_param("s", $user);
$stmt->execute();
$result = $stmt->get_result();
// Si está registrado, lo muestro
if ($result->num_rows > 0) {
    $msg = "El nombre de usuario ya está en uso, por favor elige otro.";
} else {
    // Verificar si el correo electrónico ya está registrado
    $sql = "SELECT * FROM Usuario WHERE correoElectronico = ?";
    $stmt = $conn->prepare($sql);
    // Error genérico en caso de fallo de la conexión o la BBDD
    if (!$stmt) {
        die("Ha habido un error durante el registro, inténtalo más tarde.");
    }
    $stmt->bind_param("s", $email);
    $stmt->execute();
    $result = $stmt->get_result();
    if ($result->num_rows > 0) {
        $msg = "El correo electrónico indicado ya está registrado.";
    }
}
```

Guardo en la variable msg el posible error para luego mostrarlo en caso de que lo haya y que el usuario sepa porque ha fallado el registro (usuario ya en uso, correo ya en uso, contraseña que no coincide o fallo genérico).

En caso de que todo sea correcto inserto todos los datos en la BBDD.

```
// Inserto en la base de datos si todo es correcto
$sql = "INSERT INTO Usuario (nombreUsuario, correoElectronico, password, fechaRegistro) VALUES (?, ?, ?, ?)";
$stmt = $conn->prepare($sql);
// Error genérico en caso de fallo de la conexión o la BBDD
if (!$stmt) {
    die("Ha habido un error durante el registro, inténtalo más tarde.");
}
$stmt->bind_param("ssss", $user, $email, $hashedPwd, $fechaActual);
if ($stmt->execute()) {
    $msg = "Usuario registrado correctamente. Ahora puedes iniciar sesión.";
    $redirect = "index.php";
} else {
    // Error genérico en caso de fallo de la conexión o la BBDD
    $msg = "Ha habido un error durante el registro, inténtalo más tarde.";
    $redirect = "registro.php";
}
```

A la hora de iniciar sesión, recogemos los datos por un formulario (hay varios en varias partes de la web)

```
<!-- Ventana modal para inicio sesión -->
<div id="modalSesion" class="ventanaModal">
    <div id="contenidoSesion">
        <span class="cerrar">&times;</span>
        <div class="registroLogo">
            
            <h2 id="tituloSesion">Iniciar Sesión</h2>
        </div>
        <form id="formInicioSesion" method="post" action="inicioSesion.php">
            <label for="usuario">Usuario:</label>
            <input type="text" id="usuario" name="userLog" required>
            <label for="password">Contraseña:</label>
            <input type="password" id="password" name="pwdLog" required>
            <input id="botonSesion" class="botonPerfil" type="submit" value="Iniciar Sesión">
        </form>
        <p>¿No tienes cuenta? <a href="registro.php" id="registrate">Regístrate</a></p>
    </div>
</div>
```

Si se rellena este formulario se comprueban los datos en inicioSesion.php, donde vemos si el usuario está baneado (esta es una opción que tiene exclusiva el administrador) y comprobamos que los datos son correctos (el usuario existe y la contraseña introducida coincide con la almacenada encriptada)

```
// Comprobamos que hemos recibido usuario y contraseña
if (isset($_POST['userLog']) && isset($_POST['pwdLog'])) {
    $userLog = $_POST['userLog'];
    $pwdLog = $_POST['pwdLog'];

    // Preparamos la consulta primero para verificar si el usuario está baneado
    $sql = "SELECT * FROM baneos WHERE nombreUsuario = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("s", $userLog);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        // Si es así, no iniciamos sesión y mostramos el mensaje
        $msg = "Este usuario está baneado.";
    } else {
        // Preparamos la consulta para verificar credenciales si el usuario no está baneado
        $sql = "SELECT idUsuario, password, admin FROM Usuario WHERE nombreUsuario = ?";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("s", $userLog);
        $stmt->execute();
        $result = $stmt->get_result();

        // Comprobamos si el usuario existe
        if ($result->num_rows > 0) {
            $row = $result->fetch_assoc();
            $hashedPwd = $row['password'];
            // Verificamos si la contraseña proporcionada coincide con la contraseña encriptada almacenada
            if (password_verify($pwdLog, $hashedPwd)) {
                if ($row['admin'] == 1) {
                    // Activa el modo administrador si el usuario es admin
                    $_SESSION['admin'] = true;
                }
                // Y asigna el id de usuario a la sesión
                $_SESSION['user'] = $row['idUsuario'];
                $msg = "Inicio de sesión correcto.";
            } else {
                $msg = "La contraseña introducida es incorrecta.";
            }
        } else {
            $msg = "El nombre de usuario indicado no existe.";
        }
    }
}
```

Mostramos el mensaje de éxito o fallo según el caso, y redirigimos a inicio.

Para la administración, en la tabla usuario hay un campo admin, que por defecto es false a la hora de registrar nuevos usuarios, y solo puede cambiarse desde el servidor de BBDD. Tengo un único usuario administrador por ahora (admin) y este tiene acceso a una parte exclusiva de la web donde puede ver algunas estadísticas de la web como usuario más antiguo o más nuevo, y también tiene la opción de banear/desbanear usuarios.

A la hora de iniciar sesión comprobamos si el usuario es admin para activar el modo admin en la sesión:

```
if($row['admin'] == 1){  
    // Activa el modo administrador si el usuario es admin  
    $_SESSION['admin'] = true;  
}
```

El administrador tiene acceso a la página admin.php aparte de a su perfil de usuario normal

```
<!-- Si el admin está activo mostramos su ventana de administración -->  
<?php if (isset($_SESSION['admin'])) : ?>  
      
<?php endif; ?>
```

Este apartado de la web, solo puede accederse si se es admin, aquí tenemos varias queries para recuperar datos de estadísticas de la BBDD

```
// Verificamos si el usuario es administrador  
if (isset($_SESSION['admin'])) {  
  
    // Obtenemos número de usuarios registrados  
    $sql = "SELECT COUNT(*)-1 AS numeroUsuarios FROM usuario";  
    $result = $conn->query($sql);  
    $numeroUsuarios = $result->fetch_assoc()['numeroUsuarios'];  
  
    // Obtenemos el usuario más antiguo  
    $sql = "SELECT nombreUsuario FROM usuario ORDER BY fechaRegistro ASC LIMIT 1";  
    $result = $conn->query($sql);  
    $usuarioAntiguo = $result->fetch_assoc()['nombreUsuario'];  
  
    // Obtenemos el usuario más nuevo  
    $sql = "SELECT nombreUsuario FROM usuario ORDER BY fechaRegistro DESC LIMIT 1";  
    $result = $conn->query($sql);  
    $usuarioNuevo = $result->fetch_assoc()['nombreUsuario'];  
  
    // Obtener apertura más jugada  
    $sql = "SELECT nombreApertura, COUNT(*) AS cantidad FROM aperturas GROUP BY nombreApertura ORDER BY cantidad DESC LIMIT 1";  
    $result = $conn->query($sql);  
    $aperturaJugada = $result->fetch_assoc()['nombreApertura'];  
}
```

Y también tenemos la opción de banear/desbanear usuarios a través de un formulario

```
<h2 id="subtitulo">Gestión de usuarios:</h2>
<form id="formBaneo" method="post" action="admin.php">
  Introduce el usuario a gestionar:<input type="text" name="userBan" class="input">
  <span id="userBan"></span>
  <div id="filaBoton">
    <input id="baneo" class="botonPerfil" type="submit" name="banear" value="Banear usuario">
    <input id="desbaneo" class="botonPerfil" type="submit" name="desbanear" value="Desbanear usuario">
  </div>
</form>
```

Formulario para baneo/desbaneo de usuarios

```
// Si se ha enviado el formulario para banear usuario
if (isset($_POST['banear'])) {
    $userBan = $_POST['userBan'];

    // Verificamos si el usuario ya está baneado
    $sql = "SELECT nombreUsuario FROM baneos WHERE nombreUsuario = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("s", $userBan);
    $stmt->execute();
    $result = $stmt->get_result();
    $row = $result->fetch_assoc();

    if ($row) {
        // Si ya hay resultado es que el usuario ya está baneado
        $msg = "El usuario ya está baneado.";
    } else {
        // Si no está baneado, lo insertamos en la tabla de baneos
        $sql = "INSERT INTO baneos (nombreUsuario) VALUES (?)";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("s", $userBan);
        $stmt->execute();
        $stmt->close();

        $msg = "Usuario baneado";
    }
}
```

Baneo de usuario

```

// Si se ha enviado el formulario para desbanear usuario
if (isset($_POST['desbanear'])) {
    $userDesban = $_POST['userBan'];

    // Verificamos si el usuario estaba baneado antes de desbanearlo
    $sql = "SELECT nombreUsuario FROM baneos WHERE nombreUsuario = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("s", $userDesban);
    $stmt->execute();
    $result = $stmt->get_result();
    $row = $result->fetch_assoc();

    if ($row){
        // Preparamos la consulta para desbanear usuario
        $sql = "DELETE FROM Baneos WHERE nombreUsuario = ?";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("s", $userDesban);
        $stmt->execute();
        $stmt->close();

        $msg = "Usuario desbaneado";
    }
    else {
        $msg = "El usuario no está en la lista de baneados";
    }
}

```

Desbaneo de usuario

En cada página, comprobamos si el usuario ha iniciado sesión y cargamos las opciones correspondientes, por ejemplo su pieza favorita, o sus aperturas, etc.

```
<?php

// Abrimos conexión a BBDD
include_once('conexion.php');

// Iniciamos la sesión
session_start();

// Inicializamos el usuario activo a false por defecto
$usuarioActivo = false;

// Verificamos si el usuario ha iniciado sesión
if (isset($_SESSION['user'])) {
    $userLog = $_SESSION['user'];
    $usuarioActivo = true;
    // Preparamos la consulta para sacar datos del usuario (pieza y admin)
    $sql = "SELECT piezaFavorita FROM Usuario WHERE idUsuario = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("s", $userLog);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        // Asignamos la pieza fav
        $piezaFavorita = $row['piezaFavorita'];
    } else {
        // Valor por defecto en caso de error o que no haya iniciada sesión
        $piezaFavorita = "Moderno";
    }
} else {
    // Valor por defecto en caso de error o que no haya iniciada sesión
    $piezaFavorita = "Moderno";
}
```

La variable `$_SESSION['user']` guarda el id de usuario cuando iniciamos sesión.

El cierre de sesión es bastante simple, se gestiona a través de `cerrarSesion.php`, destruye la sesión activa de forma que el `idUsuario` logado se borra, igual que el modo admin en caso de que fuera un admin el logado.

```
<?php
// Iniciamos la sesión
session_start();

// La desasignamos y destruimos para cerrar sesión
session_unset();
session_destroy();

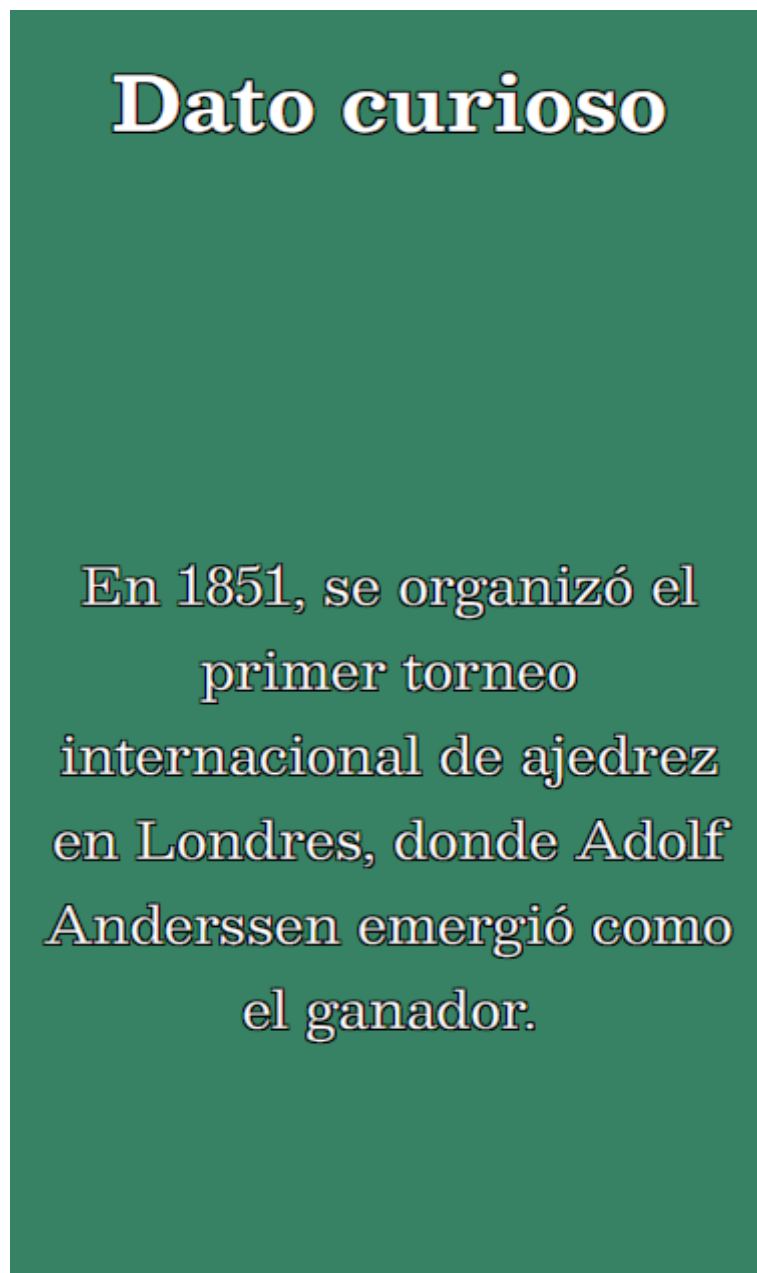
// Eliminamos también datos de formulario
$_POST = []

?>
```


8. Funcionalidades más relevantes de la aplicación

8.1 Datos curiosos

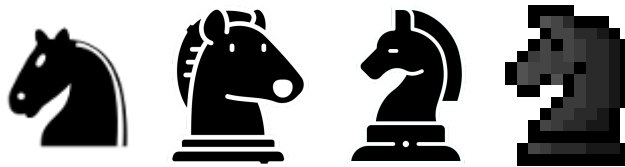
Como pequeño detalle adicional, he buscado e introducido en la base de datos, una tabla con datos curiosos acerca del mundo del ajedrez, con datos de todo tipo, que se van mostrando a los jugadores en la pantalla principal y en la del bot.



Ejemplo de dato curioso

8.2 Diseños de piezas personalizados

En el perfil del usuario, se nos muestra su fecha de registro y además, se nos da una opción para poder elegir el estilo de pieza que queremos tener de forma predeterminada para nuestras partidas y prácticas. La opción por defecto al registrarse y también en caso de no estar logado es el estilo “Moderno”.



Comparativa de caballos negros, de izquierda a derecha: “Clásico”, “Kawai”, “Moderno”, “Retro”.

Para ello, simplemente he introducido un campo en la tabla usuario de la BBDD, que por defecto viene determinado en Moderno, pero que se puede modificar en el perfil del usuario.

The image shows a user profile interface on a grey background. At the top left is an icon of a chessboard with a white king and a green queen. To its right is the word 'Perfil' in a large, bold, serif font. Below this, the text 'Estilo de piezas:' is followed by a dropdown menu. The dropdown menu is open, showing the word 'Moderno' in a blue font, with a small downward arrow on the right. At the bottom of the form is a large, rounded green button with the text 'Guardar cambios' in a black serif font.

Perfil de usuario con la opción de piezas a elegir.

8.3 Tablero en la página de inicio

En la página de inicio, he introducido un tablero completo que permite jugar partidas completas, tanto dos jugadores como si se quiere probar ciertas posiciones y luego exportar el archivo de esta partida.



Ejemplo de una partida de 18 movimientos (9 por jugador)

8.4 Problemas

Este es un pequeño añadido que iba a estar condicionado por el desarrollo del resto del proyecto. Al no ser una parte principal, sino un complemento, la complejidad de este apartado iba a estar determinado por los tiempos de desarrollo de las otras partes más relevantes.

Finalmente, he introducido varios problemas con solución de un solo movimiento, la mayoría mates en 1. Estos problemas los he obtenido del libro " Chess: 5334 Problems, Combinations and Games" de László Polgár, y los he introducido en una base de datos, separándolo en la posición inicial, y después "traduciendo" manualmente la solución a un formato que pudiera entender Javascript, me explico.

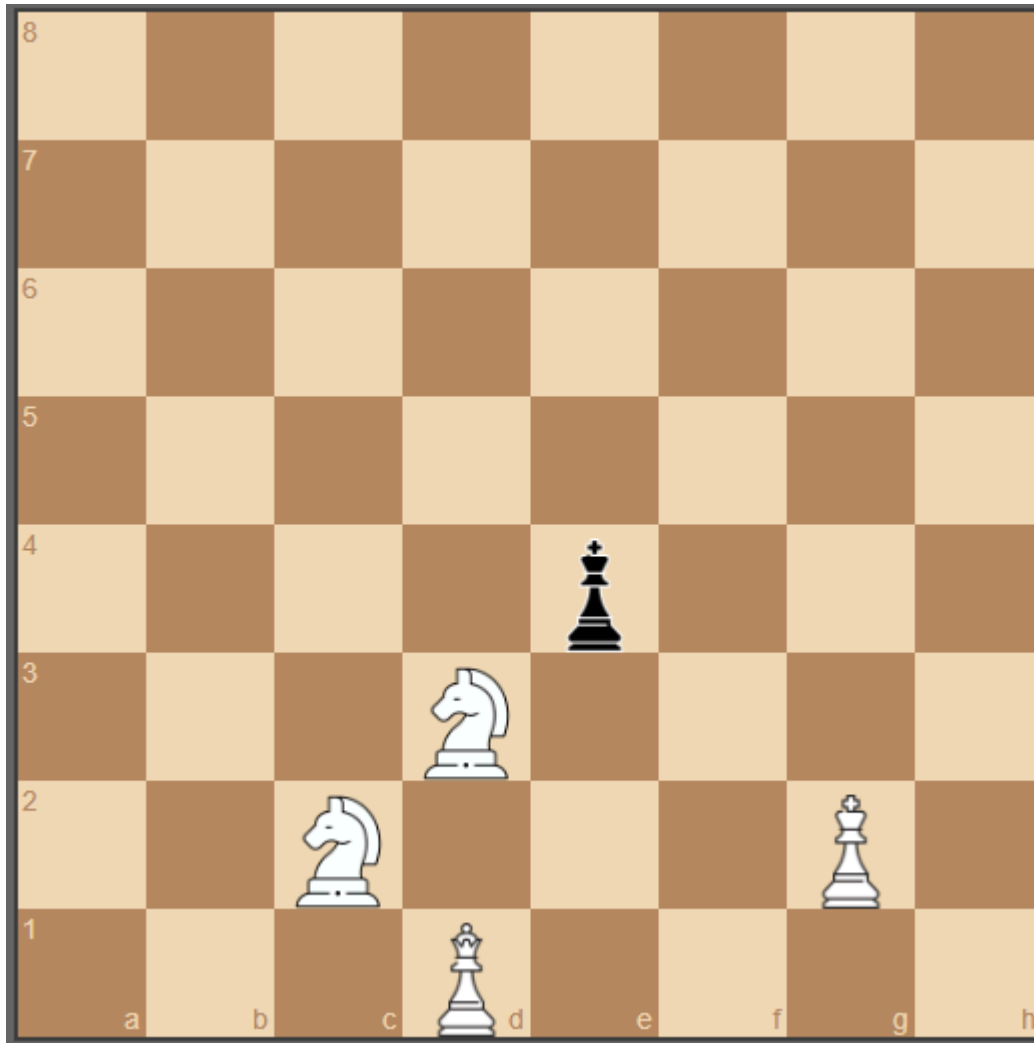
La posición de los problemas la importo en notación FEN que básicamente es una línea de texto que nos da la posición actual del tablero, siendo indiferente los movimientos anteriores (esta es la diferencia principal con la notación PGN). Esta notación se puede cargar a la biblioteca chess, haciendo que la posición inicial del tablero sea la del problema a resolver.

Ahora bien, las soluciones, vienen en notación PGN, por ejemplo, 1.Qxg7# (es decir, Reina se mueve a g7), pero esto no es una posición válida para poder comprobar si el movimiento que hace el jugador en mi aplicación es correcto.

¿Cómo he solucionado esto? Bueno, lo que he hecho es traducir a un sistema de coordenadas en el tablero la solución, de forma que mi solución tiene un formato tipo "a5-a1" esto quiere decir:

La posición de la pieza empieza en la casilla a5 y acaba en la casilla a1.

De esta manera, cojo de mi BBDD una posición inicial en FEN de mi base de datos, por ejemplo, 8/8/8/8/4k3/3N4/2N3K1/3Q4 w - - 0 1 , y cojo también la solución que sería "d1-f3".



Posición 8/8/8/8/4k3/3N4/2N3K1/3Q4 w - - 0 1

El motivo por el cual utilizo la notación FEN y no la PGN, es porque la mayoría de problemas vienen en esa notación, ya que el sentido de la notación PGN es saber los movimientos a lo largo de la partida, mientras que en un problema, no nos importa lo que haya ocurrido antes, solamente la posición. Esto hace la importación mucho más corta y sencilla al utilizar este formato.

Divido la solución con la función split por el valor "-", de forma que ya tengo la casilla de salida (d1) y la de destino (f3).

```
// Divido la solución en dos partes, casilla de inicio y casilla destino
var partesSolucion = solucion.split(/-/);

// El movimiento esperado es el que coincide con la solución
var movimientoEsperado = {
  from: partesSolucion[0],
  to: partesSolucion[1],
}
```

From determina la casilla de inicio y to la de destino, en nuestra variable movimientoEsperado, guardamos las casillas

Cuando el jugador hace un movimiento, compruebo si la pieza que ha movido parte de a5 y acaba en a1. Si es así, el jugador ha hecho el movimiento esperado, el problema está resuelto y sale un aviso felicitando al jugador, permitiéndole volver a ver el problema o pasar al siguiente.

```
movimiento = juegoAjedrez.move({
  from: source,
  to: target,
```

Esos son los valores que toma el movimiento que ha hecho el jugador

Si el movimiento realizado por el jugador no es con la pieza correcta (el valor de casilla de inicio no coincide) o no acaba en la posición correcta (la casilla destino no coincide) es que el jugador ha movido la pieza errónea o bien ha movido la pieza correcta pero a una casilla incorrecta. En ese caso el aviso indica que el movimiento es incorrecto y da la opción de reintentar o pasar a otro problema.

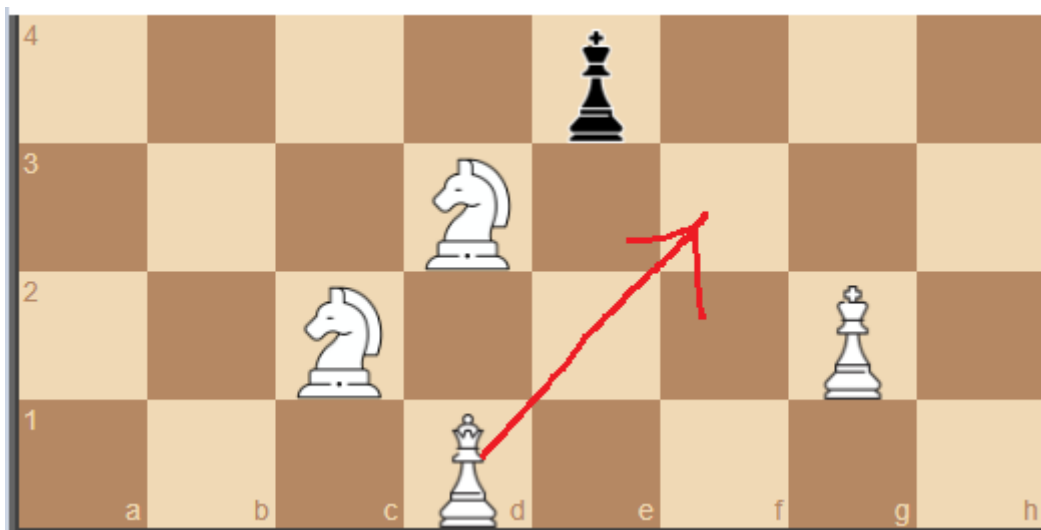
```

// Verifica si el movimiento es el esperado
if ([source === movimientoEsperado.from && target === movimientoEsperado.to
    && (!movimientoEsperado.promotion || movimiento.promotion === movimientoEsperado.promotion)]) {
    document.getElementById("textoModal").innerHTML = "<span class = 'finJuego'>¡Movimiento correcto!</span>";
    funcionAbrirModal();
} else {
    document.getElementById("textoModal").innerHTML = "<span class = 'finJuego'>¡Movimiento incorrecto! Inténtalo de nuevo</span>";
    funcionAbrirModal();
    // Ponemos sonido de error
    document.getElementById("error").play();
}

```

La parte del código que comprueba si el movimiento ha sido el esperado o no y actúa en consecuencia

En un futuro, mi idea es implementar problemas más complejos, de 2 o 3 movimientos, ya que esto da más variedad de tácticas y obliga al jugador a calcular varias jugadas futuras. Ahora bien, esto hace más compleja la solución e implica hacer que los movimientos del rival se automaticen respondiendo a los del jugador, por lo que por el momento solo he implementado problemas de un movimiento.



Este será el movimiento correcto, de la casilla D1 a la casilla F3.

8.5 Bot

Me he animado a introducir una pequeña inteligencia artificial, si bien no es demasiado compleja, es un reto para jugadores novatos e iniciados y sirva como práctica para jugadores intermedios. El bot siempre maneja las piezas negras.

En primera instancia, simplemente evaluaba la posición actual y sacaba una lista de movimientos posibles, teniendo en cuenta que fueran legales, es decir, no podía ignorar jaques o mover a casillas inalcanzables. Sin embargo, esto dejaba un resultado bastante pobre al simplemente hacer movimientos azarosos sin sentido, haciendo que apenas supusiera un reto.

Por ello, lo que hice a continuación fue evaluar las capturas disponibles en una posición dada. De esta manera, la IA no pasa por alto ninguna captura posible. Esto mejora sustancialmente su capacidad de ataque y contraataque, ya que en caso de capturarle piezas o dejar las nuestras expuestas, prioriza esto a otro movimiento, siendo capaz de hacer capturas que muchos jugadores pasan por alto. Si bien es cierto que a veces hace cambios de pieza no favorables (torre por alfil por ejemplo, o un caballo por un peón), creo que está balanceado con el hecho de que nunca “falla” ninguna captura por difícil que sea de ver.

```
// Comprueba que es el turno del ordenador (negras)
if ((juegoAjedrez.turn() === "b") && (juegoAjedrez.game_over() !== true)) {
  var movimientos = juegoAjedrez.moves(); // Obtener todos los movimientos legales
  var movimientoCaptura = movimientos.filter(function(move) {
    return move.includes('x'); // Comprueba si tiene capturas disponibles
  });
```

El bot siempre mueve piezas negras, y hace un filtro de todos los movimientos posibles, primero filtrando por los que incluyan una captura ('x')

```
if (movimientoCaptura.length > 0) {
  // Si hay uno o varios movimientos de captura, elige uno de ellos al azar
  var movimientoAleatorio = movimientoCaptura[Math.floor(Math.random() * movimientoCaptura.length)];
  juegoAjedrez.move(movimientoAleatorio);
  document.getElementById("comer").play();
  board.position(juegoAjedrez.fen()); // Actualizamos el tablero
```

Si hay varios movimientos de captura, coge uno al azar

No obstante, cuando no había capturas disponibles o en el final de la partida que no había tanto intercambio de piezas, los movimientos volvían a ser algo erráticos. Por lo que decidí que el bot también evaluara, tras comprobar que no había capturas, si había algún ataque al rey disponible (jaque) lo que aumentaba su amenaza hacia el rey del jugador.

No tenía mucho sentido dar un jaque que iba a ser neutralizado inmediatamente, porque esto provocaba que a veces diera un jaque al lado del rey y el rey simplemente comía la pieza, por lo que era un movimiento que debilitaba al propio bot más que amenazar al jugador en algunas ocasiones.

```
// Si no hay movimientos de captura disponibles, busca movimientos que pongan en jaque al rey oponente
var movimientosJaque = movimientos.filter(function(move) {
  // Filtra los movimientos que pongan al rey del oponente en jaque
  juegoAjedrez.move(move); // Intenta hacer el movimiento
  // Comprueba si el movimiento pone en jaque al rey
  var jaque = juegoAjedrez.in_check();
  // Si no prueba otro
  juegoAjedrez.undo();
  // Devolvemos el movimiento, si es que lo hay
  return jaque;
});
```

Si no hay capturas disponibles, el bloque else revisa los posibles jaques, devolviendo todos ellos

Por ello, hice una pequeña función para comprobar que movimientos ponían al rey enemigo en jaque, y después, comprobar que no permitían que la pieza que hacía el jaque fuera comida en el siguiente turno. De esta manera, el bot solo realiza jaques si no va a ser capturada inmediatamente después.

Además, para el caso de llegar con el peón al final del tablero, este promociona una pieza al azar automáticamente, ya que sus movimientos legales una vez llega al final tras mover un peón es elegir una pieza cualquiera, y por tanto elige una al azar. Podría configurarse para que promocione a una reina siempre, o incluso la mejor opción (no siempre es una reina) pero por el momento creo que está balanceado el hecho de que coja una pieza cualquiera.

```
// Filtramos los movimientos de jaque que no dejan la pieza que da el jaque en peligro
if (movimientosJaque.length > 0) {
    var movimientosJaqueSeguros = movimientosJaque.filter(function(move) {
        juegoAjedrez.move(move); // Hacer el movimiento
        var seguro = true;
        var movimientosOponente = juegoAjedrez.moves({ verbose: true });

        for (var i = 0; i < movimientosOponente.length; i++) {
            if (movimientosOponente[i].captured) {
                seguro = false;
                break;
            }
        }
        // Probamos todos
        juegoAjedrez.undo();
        // Devolvemos el movimiento, si es que lo hay
        return seguro;
    });
};
```

Dentro de los movimientos de jaque, filtra por cuales no ponen la pieza en peligro

Finalmente, si ninguna de estas condiciones se cumple, el bot hace un movimiento al azar. Había pensado implementar aquí también la posibilidad de que el bot solo hiciera movimientos al azar seguros, es decir, que no pusiera su pieza en peligro, pero creo que haría el juego algo complicado para los principiantes ya que el bot solo cometería errores a la hora de capturar, ya que no tiene un sistema de valores claro en cuanto a que piezas son más valiosas. De esta manera, tras haberlo probado muchas partidas y horas, he considerado que está bastante balanceado para tener una partida entretenida pero que no suponga un reto demasiado difícil, y que cometa algunos errores de los cuales el jugador se puede aprovechar.

```
if (movimientosJaqueSeguros.length > 0) {
    // Si hay movimientos de jaque seguros, elige uno de ellos al azar
    var movimientoAleatorio = movimientosJaqueSeguros[Math.floor(Math.random() * movimientosJaqueSeguros.length)];
    juegoAjedrez.move(movimientoAleatorio);
    document.getElementById("jaque").play();
    board.position(juegoAjedrez.fen()); // Actualizamos el tablero
} else {
    // Si no hay movimientos de jaque seguros, realiza un movimiento aleatorio
    var movimientoAleatorio = movimientos[Math.floor(Math.random() * movimientos.length)];
    juegoAjedrez.move(movimientoAleatorio);
    document.getElementById("mover").play();
    board.position(juegoAjedrez.fen()); // Actualizamos el tablero
}
```

Si hay movimientos de Jaque seguros, elige uno de ellos al azar. Si no, por último, hace un movimiento al azar ya que no hay capturas ni jaques disponibles

8.6 Aperturas

La última parte orientada al estudio del ajedrez, y la que es más diferencial en el proyecto y respecto a otras páginas orientadas al estudio del ajedrez.

Aquí lo fundamental es la capacidad del estudiante de poder importar cualquier lista de movimientos y luego poder repasarla como desee (de hecho, aunque está pensado para aperturas, es válido para aprender y repasar cualquier patrón de movimientos que el jugador desee).

Para ello, una vez iniciada sesión, podemos añadir nuevas aperturas, revisar las que tenemos actualmente y borrar la seleccionada. Si pinchamos en alguna de las opciones sin tener la sesión iniciada, la aplicación directamente abre la ventana modal para iniciar sesión.

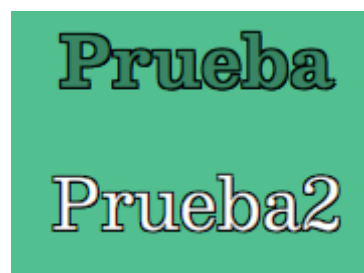


Al pulsar en Mis aperturas, si no tenemos ninguna guardada, sale el mensaje correspondiente



Así se ven las aperturas disponibles cuando tenemos algunas ya insertadas en la BBDD

Al pinchar en la apertura que queremos practicar, se resalta la opción seleccionada y se carga la posición PGN que está guardada en la BBDD asociada a esa apertura. Además se nos preguntará con que color vamos a practicar la apertura.

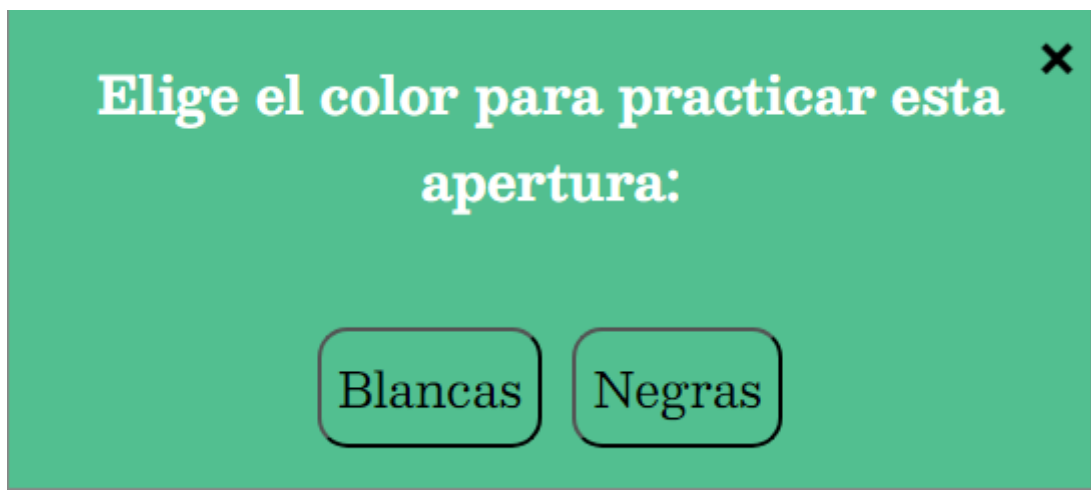


La opción Prueba es la activa actualmente

En la BBDD, guardamos las aperturas asociadas por id de usuario, y ahí se indica el nombre de la apertura y los movimientos que corresponden a la misma (el PGN).

	idApertura	idUsuario	nombreApertura	PGN
▶	1	5	Prueba	1. d4 d5 2. Nc3 Nc6
	2	5	Prueba2	1. e4 e5 2. Nc3 Nc6

Ejemplo de las aperturas asociadas al usuario con id 5



Ventana modal para saber que color es el que va a manejar el jugador

De esta manera, una vez hemos iniciado sesión, podemos ir a nuestras aperturas, seleccionar la que queremos practicar, y se carga la lista de movimientos a repasar, empezando por la posición 1. En ese momento, el entrenador nos preguntará por el siguiente movimiento a realizar, si lo realizamos y coincide con el que hay en la lista de movimientos, es que era el movimiento esperado, el programa nos indica que hemos acertado y realiza el movimiento del rival, tras lo cual sería nuestro turno nuevamente y tendremos que mover otra vez realizando el movimiento correcto.

En caso de que estemos practicando una apertura con negras, el ordenador es el que tiene que realizar el primer movimiento de la lista, moviendo nosotros después. Además, tiene en cuenta el color a la hora de orientar el tablero, ya que si queremos practicar con negras, pone el lado de las negras orientado hacia el jugador.

```

var board = ChessBoard("tablero", config);
// Función para seleccionar el color
$(".botonApertura").click(function () {
    colorJugador = $(this).data("color");
    $("#modalColor").hide();
    // Ajustamos la orientación del tablero según el color del jugador
    config.orientation = colorJugador;
    board = ChessBoard("tablero", config);
    // Comenzar el juego
    avanzarMovimiento();
});
});

```

Función para la orientación del tablero

Para la dinámica de movimientos, he creado una función avanzarMovimiento, que comprueba si hay movimientos en la lista del PGN, y en caso de haberlos, comprueba si es el turno del jugador o no. Si no lo es, sabe que el siguiente movimiento tiene que hacerlo el ordenador, posteriormente el jugador y así sucesivamente.

```

// Función para avanzar un movimiento del ordenador
function avanzarMovimiento() {
    if (indiceMovimiento < movimientos.length) {
        var movimiento = movimientos[indiceMovimiento];
        if (
            (colorJugador === "white" && movimiento.color === "b") ||
            (colorJugador === "black" && movimiento.color === "w")
        ) {
            juegoAjedrez.move(movimiento);
            indiceMovimiento++;
            setTimeout(function () {
                board.position(juegoAjedrez.fen());
                document.getElementById("mover").play();
                // Si es el último movimiento de la lista, felicitamos al jugador
                if (indiceMovimiento >= movimientos.length) {
                    document.getElementById("textoDialogo").innerHTML =
                        "¡Muy bien! Este era el último movimiento";
                } else {
                    avanzarMovimiento();
                }
            }, 500); // Esperamos 0.5 segundos para hacer el movimiento del ordenador
        }
    }
}

```

Función avanzarMovimiento

```

if (movimientoValido) {
    // Verificar si el movimiento realizado es el esperado
    if (movimiento.from === source && movimiento.to === target) {
        indiceMovimiento++;
        document.getElementById("mover").play();
        document.getElementById("textoDialogo").innerHTML = "¡Correcto!";
        setTimeout(function () {
            // Si es el último movimiento felicitamos al jugador
            if (indiceMovimiento >= movimientos.length) {
                document.getElementById("textoDialogo").innerHTML =
                    "¡Muy bien! Este era el último movimiento";
            } else {
                // Si no, preguntamos por el siguiente
                document.getElementById("textoDialogo").innerHTML =
                    "¿Siguiente movimiento?";
            }
            board.position(juegoAjedrez.fen());
            setTimeout(avanzarMovimiento, 250); // Comenzar el movimiento del ordenador después de 0.25 segundos
        }, 1000); // Mostrar el mensaje "¡Correcto!" durante 1 segundo
    }
}

```

Función que valida si el movimiento del jugador es el correcto

En caso de que el movimiento no sea el correcto, manejamos la condición con el bloque else, de manera que se deshace el movimiento que ha hecho el jugador junto con un mensaje de error por parte del entrenador, y nos permite volver a intentarlo.

```

} else {
    // Deshacemos el movimiento si no es el esperado
    juegoAjedrez.undo();
    document.getElementById("error").play();
    document.getElementById("textoDialogo").innerHTML =
        "¡Movimiento incorrecto! Inténtalo de nuevo";
}
} else {
    // Si el movimiento no es legal, no es necesario deshacer nada, solo reproducir el sonido de error
    document.getElementById("error").play();
}
}

```

Bloque else que controla el caso de fallo del jugador



El “entrenador” reacciona según si nuestros movimientos son correctos o no

Además, es posible que el jugador quiera revisar ciertos movimientos nuevamente, por lo que he añadido la funcionalidad de poder avanzar o ir hacia atrás en la lista de movimientos, o incluso al primero y último, con los botones de la cabecera.



Botones de avance y retroceso, y otro para exportar la posición actual hasta el momento


```

// Función para cargar el movimiento anterior
$("#atras").on("click", function () {
    if (indiceMovimiento > 0) {
        indiceMovimiento--;
        juegoAjedrez.undo();
        document.getElementById("mover").play();
        board.position(juegoAjedrez.fen());
        // Actualizamos el diálogo de la torre entrenadora
        document.getElementById("textoDialogo").innerHTML =
            "¿Siguiente movimiento?";
    } else {
        // Si no hay más movimientos que atrasar suena error y no hacemos nada
        document.getElementById("error").play();
    }
});

// Función para rehacer el último movimiento deshecho/avanzar un movimiento
$("#adelante").on("click", function () {
    if (indiceMovimiento < movimientos.length) {
        juegoAjedrez.move(movimientos[indiceMovimiento]);
        indiceMovimiento++;
        document.getElementById("mover").play();
        board.position(juegoAjedrez.fen());
        // Actualizamos el diálogo de la torre entrenadora
        document.getElementById("textoDialogo").innerHTML =
            "¿Siguiente movimiento?";
    } else {
        // Si no hay más movimientos que avanzar suena error y no hacemos nada
        document.getElementById("error").play();
    }
});

```

Funciones para ir atrás y adelante un movimiento respectivamente

```

// Ir al primer movimiento
$("#principio").on("click", function () {
    juegoAjedrez.reset();
    indiceMovimiento = 0;
    board.position(juegoAjedrez.fen());
    // Actualizamos el diálogo de la torre entrenadora
    document.getElementById("textoDialogo").innerHTML =
        "¿Siguiente movimiento?";
    // Si es el turno del ordenador, avanzamos su movimiento
    if (
        (colorJugador === "white" && juegoAjedrez.turn() === "b") ||
        (colorJugador === "black" && juegoAjedrez.turn() === "w")
    ) {
        setTimeout(avanzarMovimiento, 500);
    }
});

// Ir al último movimiento
$("#final").on("click", function () {
    while (indiceMovimiento < movimientos.length) {
        juegoAjedrez.move(movimientos[indiceMovimiento]);
        indiceMovimiento++;
    }
    board.position(juegoAjedrez.fen());
});

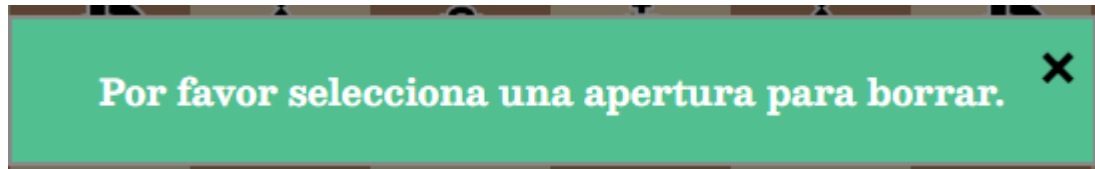
```

Funciones para ir al primer y último movimiento respectivamente

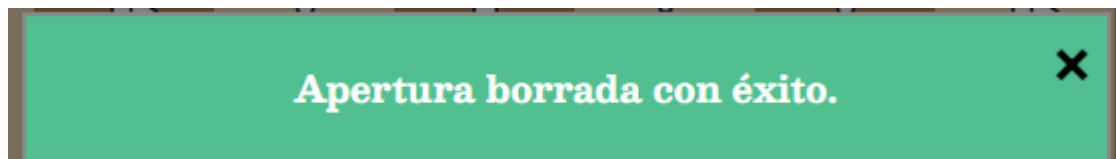
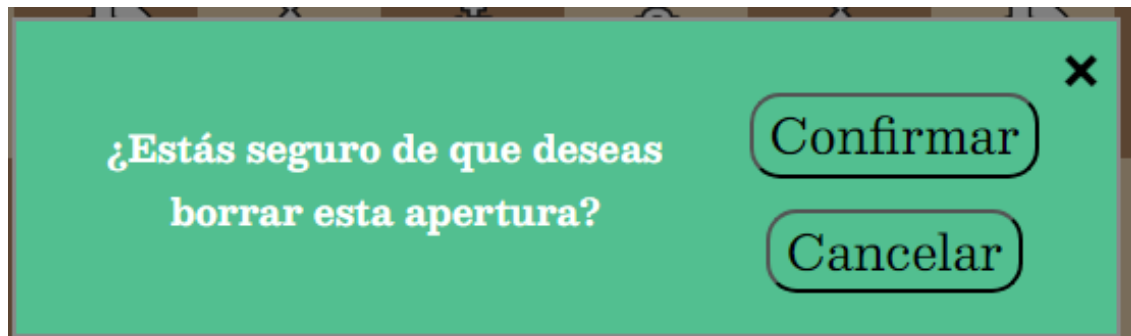
Además, la opción de ir al inicio, comprueba si el primer movimiento lo tiene que realizar el ordenador, por si queremos repetir la apertura nuevamente, tener esa posibilidad de empezar desde el primer movimiento.

Por último podemos borrar una apertura.

Al pulsar el botón de borrar apertura, comprobamos que efectivamente tenemos una apertura seleccionada, ya que el borrado hace referencia a la apertura activa. Si no hay ninguna seleccionada, se muestra un mensaje de error.



En caso de que haya una seleccionada, sale un mensaje de confirmación para el borrado de la apertura, y si se confirma se envía la id del usuario al script aperturas para que sepa que usuario es el que tiene iniciada la sesión y busque la apertura correspondiente para borrarla. Si no se confirma lógicamente no hace nada.



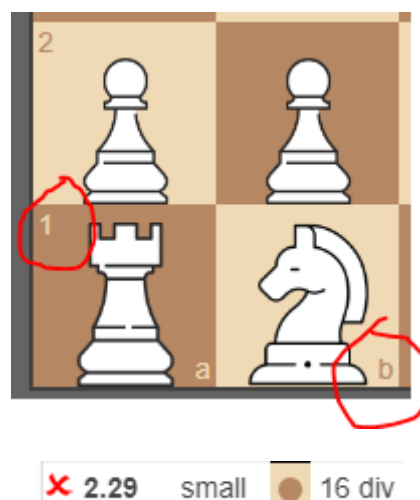
9. Conclusiones

Tras haber finalizado el desarrollo de la aplicación, podemos revisar punto por punto cada uno de los aspectos relevantes acerca de la misma, de menor a mayor importancia.

9.1 Diseño y accesibilidad

Aún siendo una parte que no ha sido prioritaria, he sabido mantener una buena relación entre sencillez pero atractivo, utilizando fuentes discretas pero estéticas, una paleta de colores oscuros, pero con contrastes, y que considero se ajustan a la temática del ajedrez. Tenemos varios elementos dinámicos como ventanas modales, algunas animaciones, pero nada demasiado saturante o exagerado. Predomina la funcionalidad y el contenido sobre el diseño, cosa que era lo que pretendía. Por supuesto, la aplicación es totalmente responsive.

El único “pero” que he encontrado a nivel de accesibilidad es, utilizando la herramienta Color Contrast Checker, me indica que hay poco contraste entre el fondo del tablero y los números y letras que marcan la casilla. No obstante, por el momento no tengo soluciones para hacer accesible el ajedrez para personas con visibilidad afectada o reducida.



También una de las mayores complicaciones respecto al diseño, fue el tamaño del tablero. Primero me encontré con que, a pesar de ser responsive, había que actualizar la página para que implementara los cambios, cosa que era bastante molesta y afectaba a la funcionalidad, descubriendo después que había una opción en la biblioteca de chessboard.js para hacerlo responsive al momento. No obstante, la segunda dificultad fue que solo era responsive a lo ancho, ya que requería una altura mínima. Estuve barajando la opción de dejar un scroll al lateral del tablero en caso de que no llegara al mínimo pero me pareció una solución demasiado pobre, y finalmente, lo que hice fue ajustar el tablero a un contenedor exterior en tiempo real, con el uso de esta función:

```
// Función para ajustar el tamaño del contenedor del tablero y del tablero según se reajusta la ventana al contenedor

function adjustBoardSize() {
  // Dejo un 90% del contenedor de ancho para que deje algo de margen
  var containerWidth = $("#contenedorTablero").width() * 0.9;
  // El alto es un 80% de la ventana
  var containerHeight = $(window).height() * 0.8;

  // Determinar el tamaño del tablero basado en el tamaño del contenedor
  var size = Math.min(containerWidth, containerHeight);

  // Ajustamos el tamaño del tablero
  $("#tablero").css({
    width: size + "px",
    height: size + "px",
  });

  // Redimensionamos el tablero acorde a los nuevos tamaños
  board.resize();
}

// Hacer el tablero dinámicamente responsive ancho y alto según se actualiza la ventana
$(window).resize(function () {
  adjustBoardSize();
});

// Ajustar el tamaño del tablero al cargar la página
adjustBoardSize();
```

Función para hacer el tablero responsive según se fuera cambiando el ancho o alto de la página

9.2 Uso intuitivo

La web tiene una navegación bastante sencilla y donde el usuario rápidamente identifica todas las posibilidades. Los botones, o bien son auto descriptivos, o tienen un significado universal (ej: el caso del botón de usuario para iniciar sesión). El resto está claramente identificado, siendo sencillo navegar de un lugar a otro, y acceder a cualquier sección desde cualquier parte de la página.

9.3 Funcionalidad

Todos los apartados hasta el momento incluidos en la aplicación son completamente funcionales y están completos. Ahora bien, muchos en el futuro están sujetos a cambios y mejoras, especialmente para aumentar sus posibilidades y profundidad, véase el caso de los problemas, donde tengo en mente ampliar la complejidad de los mismos, haciéndolos de varios movimientos, y la nueva funcionalidad de Análisis de las posiciones en el tablero, que será un apartado totalmente nuevo.

10. Programación temporal de los trabajos a realizar

Si bien el proyecto a día de hoy es un prototipo totalmente funcional y está listo para operar, quiero añadir algunos puntos relevantes que ya he mencionado en sus apartados correspondientes.

1. Inclusión de mayor número de problemas y con soluciones más complejas. Julio de 2024.
2. Inclusión de un apartado de análisis con un motor de ajedrez que permita revisar posiciones y analizarlas utilizando Stockfish. Septiembre de 2024.

De esta forma, el proyecto quedará como una página totalmente funcional y una de las más completas a día de hoy para el entrenamiento de ajedrez.

11. Bibliografía, fuentes y otros recursos

Páginas web

<https://www.flaticon.es/> Iconos de uso libre

<https://www.figma.com/> Diseño de bocetos de web

<https://pixabay.com/es/> Imágenes y audios de uso libre

<https://www.chess.com/> Ideas e inspiración, consulta de foros, preguntas de jugadores

<https://lichess.org/> Analisis de partidas

<https://es.chessbase.com/> Noticias y novedades sobre el ajedrez

<https://database.chessbase.com/> Base de datos con partidas y estadísticas

<https://chessboardjs.com/> Biblioteca para creación del tablero de ajedrez y su funcionamiento

<https://github.com/jhlywa/chess.js> Biblioteca para la implementación de funcionamiento y reglas de ajedrez (movimientos legales, jaques, mates, etc).

<https://es.wikipedia.org/> Información y datos sobre ajedrez

<https://www.canva.com/> Diseño de logos

<https://stackoverflow.com/> Preguntas y dudas resueltas

<https://chat.openai.com/> Dudas sobre el funcionamiento de alguna herramienta, ideas para el proyecto

<https://blog.davideai.dev/make-a-basic-chess-board-with-chessboardjs>

Blog donde explican la implementación de chessboardjs

<https://www.youtube.com/@ChessVibesOfficial> Creador de contenido de ajedrez

<https://www.youtube.com/@GothamChess> Creador de contenido de ajedrez

<https://www.youtube.com/@AnnaCramling> Creador de contenido de ajedrez

<https://www.youtube.com/@ReyEnigma> Creador de contenido de ajedrez

<https://opengameart.org/content/pixel-chess-pieces> Estilos de piezas de ajedrez

Libros

“Fundamentos de ajedrez” José Raúl Capablanca

“Como ganar al ajedrez” Levy Rozman

“Aperturas de Ajedrez” Miguel Illescas

“5334 Problems, Combinations, and Games” László Polgár