Proyecto colaborativo

“Chess.mx”

Elaborado por:

- Hernández Muñoz, Diego Iván
- Ramírez Navarro, Angel Alexander
- De La Cruz Ascencio, Luis Angel

Mtro. Rubén Adrián Gil Rivera

Seminario de Ing. De Software

Sietemesinos

Índice

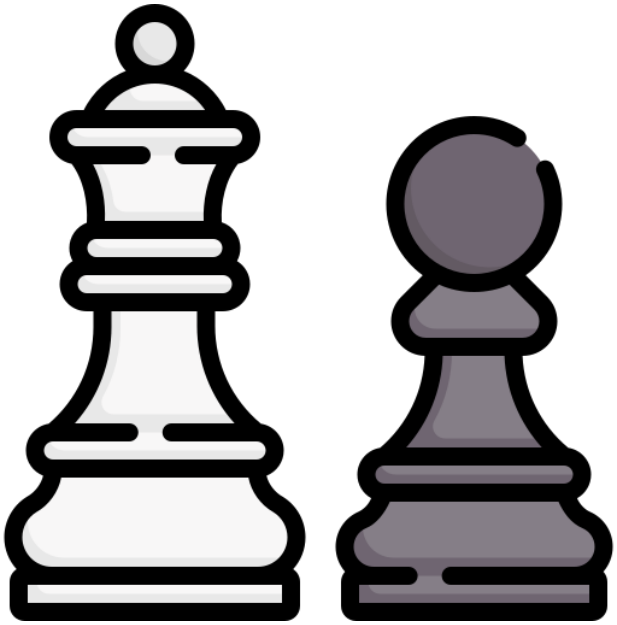
1. Descripción del proyecto de software 4
2. Justificación 5
3. Viabilidad 6
Viabilidad técnica 6
Viabilidad Económica 6
Viabilidad Operativa 6
Viabilidad legal 7
Viabilidad ambiental 7
4. Objetivo general 8
5. Objetivos específicos 8
6. Riesgos 9
7. Tabla de requerimientos funcionales 10
8. Tabla de requerimientos no funcionales 12
9. Timing Plan 13
10. Diagramas UML de Casos de uso 13
10.1 Diagrama de Casos de Uso 14
10.2. Diagrama de Secuencia 15
10.3. Diagrama de Clases de la arquitectura 15
10.4. Diagrama de Máquina de estados 16
11. Reporte de pruebas 17
11.1 Casos de prueba modulares 17
SMT_TestCase_1_DB_Connection 17
SMT_TestCase_2_Register_And_Login 18
SMT_TestCase_3_Board 20
SMT_TestCase_4_Pawn_Promotion 21
SMT_TestCase_5_En_Passant 22
11.2 Casos de prueba de Integración 23
SIT_TestCase_1_Auth_And_Access_To_Interfaces 23
SIT_TestCase_2_Player_Selection 24
SIT_TestCase_3_Game_Creation_In_DB 26
SIT_TestCase_4_Chess_r 27
SIT_TestCase_5_Leaderboard 29

11.3 Casos de prueba de Sistema 31
ST_TestCase_1_Authentication 31
ST_TestCase_2_User_Can_Do_Valid_Movements_Of_Pieces 32
ST_TestCase_3_Show_Check_Mate_Result 33
ST_TestCase_4_Show_LeaderBoard 34
ST_TestCase_5_Fast_Response_Of_Pieces 35
11.4 Total de pruebas 37
12. Manual de usuario 38
13. Conclusiones 38

1. Descripción del proyecto de software

Este proyecto será desarrollado utilizando el lenguaje de programación Java. Aunque actualmente no contamos con una gran experiencia en este lenguaje, lo hemos elegido por su adaptabilidad y relevancia en la industria del software.

El objetivo principal será crear un juego de ajedrez con funciones adicionales. Inicialmente, nos enfocaremos en implementar partidas entre dos jugadores en una misma pantalla, y posteriormente se añadirá la posibilidad de jugar contra la computadora. Además, los usuarios deberán crear un perfil para poder ingresar al juego, y los puntajes de las partidas serán almacenados en dicho perfil. Con base en estos puntajes, se generará una tabla de clasificación (leaderboard) que mostrará a los usuarios con mejores resultados.



2. Justificación

Si bien, en un principio, podría parecer que la creación de un juego de ajedrez no aborda problemáticas significativas más allá del entretenimiento del usuario, es importante destacar que la realización y el éxito de este proyecto contribuirán de manera sustancial al desarrollo de las habilidades técnicas de los desarrolladores, a la aplicación práctica de sus conocimientos y, por supuesto, al disfrute de los usuarios que hagan uso de él.

Dado que aún no se posee un dominio extenso del lenguaje Java y que este representa un entorno relativamente nuevo para el equipo, el desarrollo de este proyecto les permitirá adquirir competencias clave en programación orientada a objetos, así como aplicar diversos algoritmos y estructuras de datos para gestionar el comportamiento y los movimientos de cada una de las piezas del ajedrez. Además, pondrán en práctica sus conocimientos sobre diseño de interfaces gráficas para el desarrollo del tablero y la presentación visual del juego

Asimismo, se aprovecharán las habilidades adquiridas en la implementación de las reglas del ajedrez, incluyendo movimientos complejos como el enroque o el jaque mate, así como el manejo adecuado de excepciones, con el fin de crear una aplicación sólida que sirva como un valioso ejercicio de aplicación de teorías en un entorno práctico.

Por último, el entretenimiento será una consecuencia directa de la creación de **Chess.mx**, dado que el ajedrez es uno de los juegos más populares a nivel mundial. Al trasladarlo a una versión digital desarrollada en Java, no solo se proporcionará una herramienta de diversión, sino también una plataforma para el aprendizaje y la competición entre los usuarios.

3. Viabilidad

El desarrollo de **Chess.mx** es viable y factible, considerando los recursos con los que cuenta el equipo de desarrolladores.

Viabilidad técnica

El proyecto es técnicamente factible debido a que el lenguaje de programación que será utilizado para su desarrollo es versátil y ampliamente utilizado en el desarrollo de aplicaciones de escritorio y juegos. Las librerías que ofrece facilitan el diseño de interfaces gráficas llamativas e intuitivas, y al ser un lenguaje orientado a objetos es idóneo para el modelaje de las piezas de ajedrez y sus comportamientos.

Además, el equipo desarrollador cuenta con Entornos de Desarrollo Integrados, también conocidos como IDEs, que proporcionan soporte gratuito para llevar a cabo la programación en el lenguaje Java, así como herramientas de control de versiones que permitirán llevar una gestión correcta del proyecto, tal como lo es GitHub.

Y en cuanto a los conocimientos adquiridos, a pesar de que el equipo de desarrollo se encuentra en una etapa de aprendizaje a profundidad del lenguaje Java, se posee una base sólida en cuanto a los fundamentos de la programación, los algoritmos y las estructuras de datos.

Viabilidad Económica

El desarrollo de **Chess.mx** no conllevará de barreras económicas de gran amplitud debido a que los recursos que se estarán utilizando por parte de los desarrolladores serán de libre acceso. Por otro lado, el tiempo a invertir en el desarrollo del proyecto no será mayor a tres meses, por lo que costos adicionales ligados al tiempo invertido no se encuentran previstos.

Viabilidad Operativa

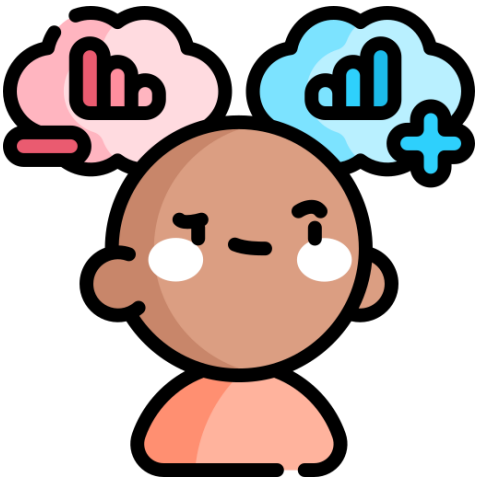
Todos los integrantes del equipo programador tienen la capacidad de operación para desarrollar e implementar el proyecto en los plazos de tiempo previstos, cumpliendo con lo establecido previamente. Se utilizarán los conocimientos adquiridos y se usarán los aprendidos durante la creación del juego. Además, se ha establecido un cronograma de actividades para el desarrollo del proyecto, dividiéndolo en fases manejables como el diseño de la interfaz gráfica, la implementación de las reglas de juego, el desarrollo de la lógica de los movimientos de las piezas y la fase de pruebas.

Viabilidad legal

Al pertenecer el juego de ajedrez al dominio público, no conllevará algún problema o restricción legal en cuanto a la creación de una versión digital. Además, toda librería o herramienta a utilizar cumplen con las licencias de uso.

Viabilidad ambiental

El software digital no requiere del uso de recursos materiales o energéticos más allá del necesario para su desarrollo en los equipos de cómputo, por lo que no conllevará de impactos ambientales significativos.



4. Objetivo general

Desarrollar un juego de ajedrez funcional para computadoras, que permita a dos usuarios jugar por turnos en la misma pantalla, con un sistema de puntuación integrado.

5. Objetivos específicos

- Desarrollar un sistema de inicio de sesión que permita a los usuarios registrarse y jugar una intyedrez.
- Implementar una tabla de clasificación (leaderboard) que muestre a los usuarios con los mejores puntajes de las partidas.
- Crear un sistema de almacenamiento que asocie los puntajes de las partidas con el perfil del usuario.
- Diseñar una interfaz intuitiva y atractiva, con botones grandes y colores y fuente adecuados, procurando que sea fácil de usar. El equipo de desarrollo intentará superar la curva de aprendizaje del lenguaje Java para lograr este objetivo.
- Desarrollar una base de datos optimizada y normalizada para garantizar tiempos de respuesta rápidos y eficientes.
- Cumplir con la entrega del software en la fecha establecida (25 de noviembre de 2024), asegurando que cumpla con los requisitos acordados.
- Garantizar la seguridad de los datos del usuario (nombre de usuario, correo y contraseña) para prevenir posibles ataques al sistema.
- Validar que la contraseña del usuario cumpla con los requisitos de seguridad adecuados para proteger su información.
- Verificar que no existan nombres de usuario duplicados en la base de datos, evitando la redundancia.

6. Riesgos

1. **Falta de conocimiento en Java**: El equipo no tiene experiencia previa en el lenguaje Java, lo que podría generar dificultades técnicas y desafíos de aprendizaje que ralenticen el avance del proyecto.
    - **Mitigación:** Dedicar tiempo a estudiar los conceptos básicos y practicar con pequeños proyectos antes de abordar objetivos más grandes.
2. **Sobrecarga de funcionalidades:** Incluir demasiadas funcionalidades complejas (como implementar un algoritmo de IA para poder lograr el juego contra la computadora) podría hacer que el proyecto se vuelva difícil de manejar, lo que pondría en riesgo el cumplimiento de los plazos.
    - **Mitigación:** Priorizar las funcionalidades esenciales, como el juego entre dos jugadores. Dejar funcionalidades más complejas como opcionales según el tiempo disponible.
3. **Problemas de integración de componentes:** Integrar diferentes módulos (sistema de log in, base de datos, y lógica del juego) podría causar errores imprevistos.
    - **Mitigación:** Probar cada módulo antes de combinarlo con otros. Utilizar control de versiones (Git) para manejar cambios.
4. **Problemas con la base de datos:** Un mal diseño o implementación de la base de datos podría generar tiempos de respuesta lentos o errores en la asociación de datos.
    - **Mitigación:** Diseñar la base de datos con anticipación, además de su respectiva normalización.
5. **Bugs no detectados o mal manejo de errores:** El juego podría presentar errores que afecten la jugabilidad o la experiencia del usuario.
    - **Mitigación:** Implementar pruebas unitarias y de integración de forma continua.
6. **Falta de documentación:** La falta de documentación del código podría dificultar su comprensión en un futuro, así como el mantenimiento y la adición de futuras mejoras del sistema.
    - **Mitigación:** Mantener una buena documentación desde el inicio, incluyendo comentarios en el código y manuales de usuario.
7. **No adoptar un estilo de codificación:** La ausencia de un estilo de codificación uniforme puede llevar a un código desordenado y difícil de leer.
    - **Mitigación:** Establecer y documentar un estilo de codificación claro y uniforme para todo el equipo. Esto puede incluir convenciones de nombres de variables, clases, estructura de archivos, formato de código, y comentarios.
8. **Gestión inadecuada del tiempo:** La falta de una buena planificación podría llevar a que el proyecto no se termine dentro del plazo establecido (25 de noviembre de 2024).

- **Mitigación:** Seguir, revisar y modificar el cronograma de actividades regularmente, según convenga.

7. Tabla de requerimientos funcionales

ID de requerimiento	***RF01***
Tipo	Funcional
Funcionalidad	Registro de usuario
Requerimiento	El sistema debe permitir que los jugadores se registren en *Chess.mx* creando un nombre de usuario y contraseña.
Comentarios	Validar que el nombre de usuario no exista previamente en la base de datos para evitar redundancia. Verificar que la contraseña cumpla con los siguientes requisitos: • Longitud mínima de ocho caracteres. • Contener al menos una letra mayúscula. • Incluir al menos un número.
Id_Test_Case_Satisfied	SMT_TestCase_2_Register_And_Login

ID de requerimiento	***RF02***
Tipo	Funcional
Funcionalidad	Inicio de sesión
Requerimiento	El sistema debe permitir a los usuarios registrados iniciar sesión en *Chess.mx* para poder jugar una nueva partida.
Comentarios	Si el nombre de usuario o contraseña es incorrecto, se debe mostrar un mensaje de error adecuado.
Id_Test_Case_Satisfied	SMT_TestCase_2_Register_And_Login

ID de requerimiento	***RF03***
Tipo	Funcional
Funcionalidad	Partida entre dos jugadores
Requerimiento	El sistema debe permitir que dos usuarios jueguen una partida de ajedrez por turnos de forma local (en la misma pantalla).
Comentarios	Mostrar el nombre del jugador en turno
Id_Test_Case_Satisfied	SIT_TestCase_4_Chess_Match

ID de requerimiento	***RF04***
Tipo	Funcional
Funcionalidad	Sistema de puntaje
Requerimiento	El sistema debe registrar el resultado de cada partida (victoria, derrota o empate) y asociarlo al perfil de los usuarios que participaron en la partida.
Comentarios	N/A
Id_Test_Case_Satisfied	N/A

ID de requerimiento	***RF05***
Tipo	Funcional
Funcionalidad	Tabla de clasificación
Requerimiento	Implementar una tabla de clasificación (leaderboard) que muestre a los usuarios con los mejores puntajes de las partidas.
Comentarios	La tabla debe permitir la visualización de los 10 mejores jugadores.
Id_Test_Case_Satisfied	SIT_TestCase_5_Leaderboard

ID de requerimiento	***RF06***
Tipo	Funcional
Funcionalidad	Iniciar una partida
Requerimiento	El sistema debe permitir al usuario iniciar una nueva partida de ajedrez desde la pantalla principal.
Comentarios	N/A
Id_Test_Case_Satisfied	SIT_TestCase_4_Chess_Match

ID de requerimiento	***RF07***
Tipo	Funcional
Funcionalidad	Tablero
Requerimiento	El sistema debe mostrar un tablero de ajedrez de 8x8 con las piezas en sus respectivas posiciones iniciales.
Comentarios	El tablero deberá tener sus respectivas guías de números y letras.
Id_Test_Case_Satisfied	SMT_TestCase_3_Board

ID de requerimiento	***RF08***
Tipo	Funcional
Funcionalidad	Piezas
Requerimiento	El sistema debe permitir al usuario mover las piezas del tablero según las reglas oficiales del ajedrez.
Comentarios	El sistema debe verificar que el movimiento realizado sea válido de acuerdo con las reglas del ajedrez. Debe actualizarse el estado del tablero después de cada movimiento.
Id_Test_Case_Satisfied	ST_TestCase_2_User_Can_Do_Valid_Movements_Of_Pieces

ID de requerimiento	***RF09***
Tipo	Funcional
Funcionalidad	Reglas de juego
Requerimiento	El sistema debe implementar movimientos especiales como enroque, coronación y captura al paso.
Comentarios	N/A
Id_Test_Case_Satisfied	SMT_TestCase_5_En_Passant

ID de requerimiento	***RF10***
Tipo	Funcional
Funcionalidad	Mostrar resultado de la partida
Requerimiento	El sistema debe mostrar un mensaje de victoria, derrota o empate al final de cada partida.
Comentarios	N/A
Id_Test_Case_Satisfied	ST_TestCase_3_Show_Check_Mate_Result

ID de requerimiento	***RF11***
Tipo	Funcional
Funcionalidad	Almacenamiento de resultado de una partida
Requerimiento	El sistema debe guardar los resultados de las partidas en una base de datos relacional
Comentarios	N/A
Id_Test_Case_Satisfied	ST_

8. Tabla de requerimientos no funcionales

ID de requerimiento	*RNF01*
Tipo	No funcional
Funcionalidad	Rendimiento
Requerimiento	El juego debe garantizar tiempos de respuesta rápidos para las acciones de los usuarios, como el inicio de sesión, el registro y el movimiento de las piezas durante una partida.
Comentarios	El tiempo de respuesta para las acciones no debe exceder de 3 segundos.
Id_Test_Case_Satisfied	ST_TestCase_5_Fast_Response_Of_Pieces

ID de requerimiento	*RNF02*
Tipo	No funcional
Funcionalidad	Usabilidad
Requerimiento	La interfaz del usuario debe ser funcional, intuitiva y fácil de usar para garantizar una experiencia de usuario agradable.
Comentarios	N/A
Id_Test_Case_Satisfied	SIT_TestCase_4_Chess_r

ID de requerimiento	*RNF03*
Tipo	No funcional
Funcionalidad	Mantenimiento
Requerimiento	El sistema debe ser fácilmente mantenible para permitir actualizaciones y mejoras futuras sin requerir grandes modificaciones.
Comentarios	Documentar el código y seguir un estilo de codificación para que el código sea legible.
Id_Test_Case_Satisfied	N/A

ID de requerimiento	*RNF04*
Tipo	No funcional
Funcionalidad	Seguridad
Requerimiento	El sistema debe de contar con cifrado para las contraseñas de los usuarios.
Comentarios	N/A
Id_Test_Case_Satisfied	ST_TestCase_1_Password_Encryption

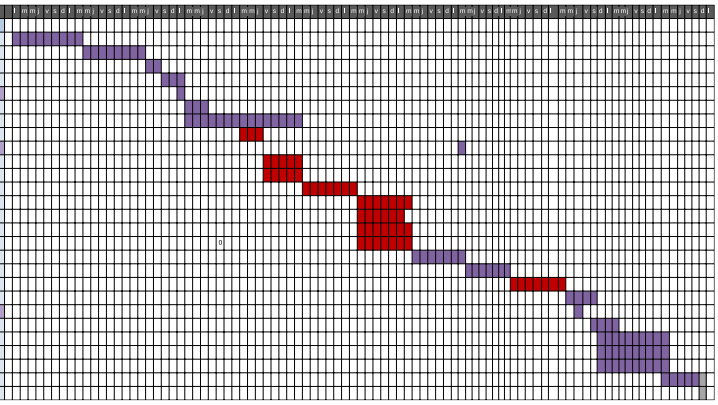
ID de requerimiento	*RNF05*
Tipo	No funcional
Funcionalidad	Estabilidad
Requerimiento	El sistema debe tener una conexión estable con una base de datos relacional SQL.
Comentarios	N/A
Id_Test_Case_Satisfied	N/A

ID de requerimiento	*RNF06*
Tipo	No funcional
Funcionalidad	GUI
Requerimiento	El tablero y la interfaz deben ser visualmente atractivos y con una gama de colores adecuado.
Comentarios	N/A
Id_Test_Case_Satisfied	SIT_TestCase_4_Chess_r

9. Diagrama de Gantt, timing plan y ruta crítica

A continuación, se presenta nuestro diagrama de Gantt, el cual refleja las actividades realizadas durante la elaboración del proyecto, incluyendo la asignación de tareas a los diferentes miembros del equipo. Además, se muestra la ***ruta crítica*** del proyecto, ya que cualquier retraso en dichas actividades podría afectar directamente la fecha de entrega. Las actividades que forman parte de la ruta crítica están resaltadas en color rojo para una mejor identificación.

ID	TAREA	ASIGNADO A	PROGRESO	INICIO	FIN
	Proyecto: Chess.mx				
1	Realización de Timing Plan	All team	100%	26-8-24	3-9-24
2	Recolección de requerimientos, objetivos y justificación	All team	100%	4-9-24	11-9-24
3	Asignación de roles y actividades	All team	100%	12-9-24	13-9-24
4	Documento de Arquitectura de Software	All team	100%	14-9-24	16-9-24
XX	**FIRST MILESTONE**		**100%**	**16-9-24**	**16-9-24**
4	Documento de Arquitectura de Software	All team	100%	17-9-24	19-9-24
5	Documentos de diseño software (Diagramas UML)	All team	100%	17-9-24	1-10-24
6	Definir la estructura de la base de datos	Alexander Ramirez	100%	24-9-24	26-9-24
XX	***SECOND MILESTONE***		***100%***	***22-10-22***	***10-24***
7	Creación de registro y log in de usuario	Iván Muñoz	100%	27-9-24	1-10-24
8	Conexión con base de datos	Alexander Ramirez	100%	27-9-24	1-10-24
9	Diseño de interfaz de usuario y tablero	Iván Muñoz	100%	2-10-24	8-10-24
10	Diseño de piezas	Angel De La Cruz, Alexander	100%	9-10-24	15-10-24
11	Creación de los algoritmos de las piezas	Angel De La Cruz	100%	9-10-24	14-10-24
12	Creación de restricciones de las piezas	All team	100%	9-10-24	15-10-24
13	Implementación de movimientos dentro del tablero	Iván Muñoz, Angel De La Cruz	100%	9-10-24	15-10-24
14	Implementación de lógica para juego por turnos	Alexander Ramirez	100%	16-10-24	22-10-24
15	Creación de algoritmos para diferentes resultados de	All team	100%	23-10-24	29-10-24
16	Implementación de sistema de puntuación y podio	Angel De La Cruz	100%	30-10-24	5-11-24
17	Implementación de juego contra PC (a tratar con el	Iván Muñoz, Alexander	100%	6-11-24	9-11-24
18	***THIRD MILESTONE***		***100%***	***7-11-24***	***7-11-24***
19	Implementación de juego contra PC (a tratar con el	Iván Muñoz, Alexander	100%	9-11-24	12-11-24
20	Diseño de pruebas	All team	100%	10-11-24	19-11-24
21	Ejecución de pruebas	All team	100%	10-11-24	19-11-24
22	Reporte de pruebas	All team	100%	10-11-24	19-11-24
23	Creación de manual de usuario	All team	100%	18-11-24	24-11-24
24	Entrega del proyecto	All team	100%	24-11-24	24-11-24



Clic aquí para ver el timing plan en la plataforma Trello.

10. Diagramas UML de Casos de uso

10.1 Diagrama de Casos de Uso

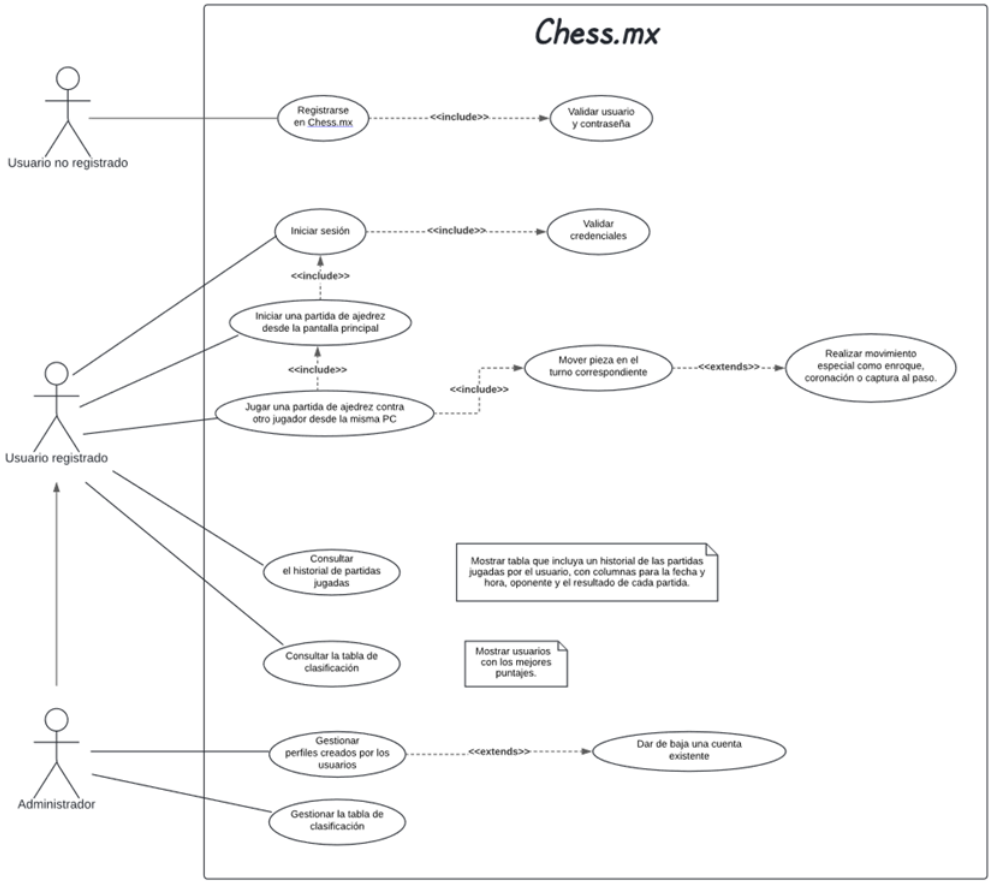


Figura 10.1.1 Diagrama de Casos de Uso que detalla las interacciones para tres tipos de usuarios: Usuario no registrado, Usuario registrado y Administrador. Incluye acciones como registrarse, iniciar sesión, jugar, consultar historial de partidas, entre otros, mostrando cómo los distintos usuarios interactúan con el juego.

10.2. Diagrama de Secuencia

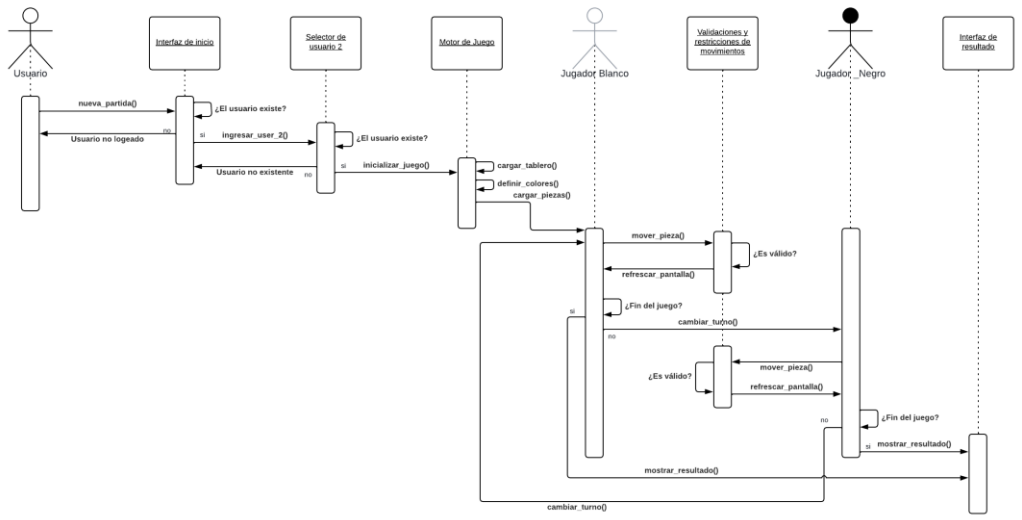


Figura 10.2.1 Diagrama de Secuencias, donde se muestra el flujo de interacción entre cada parte del sistema para gestionar una partida de ajedrez, desde el registro de usuarios, las acciones de los jugadores, la verificación de reglas, hasta la determinación del ganador y el almacenamiento de los resultados.

10.3. Diagrama de Clases de la arquitectura

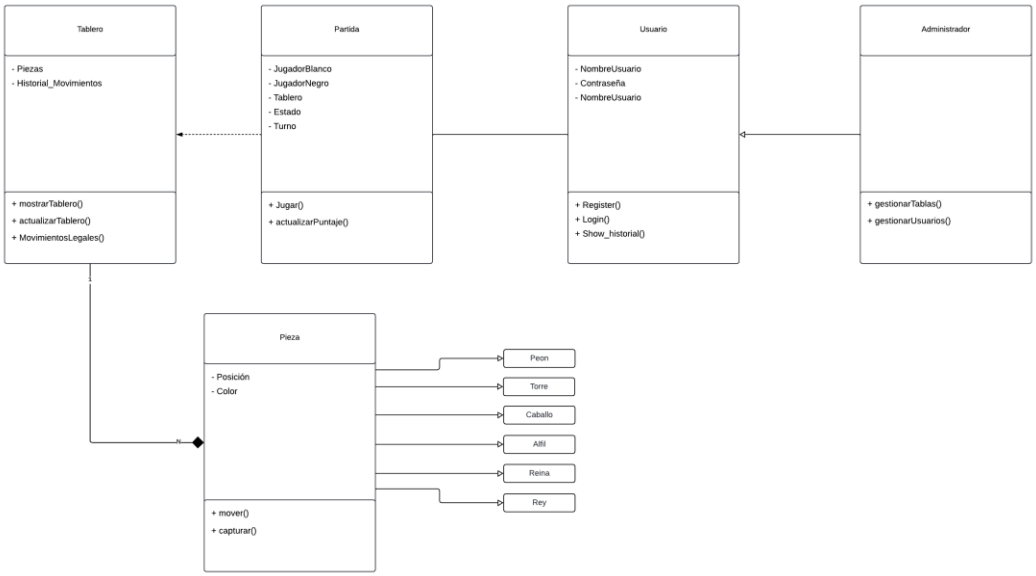


Figura 10.3.1 Diagrama de Clases 1.0 que muestra a manera generalizada las clases y cómo se relacionan una con otra dentro del software, además de integrar su cardinalidad, sin adentrarse a las funciones o variables específicas.

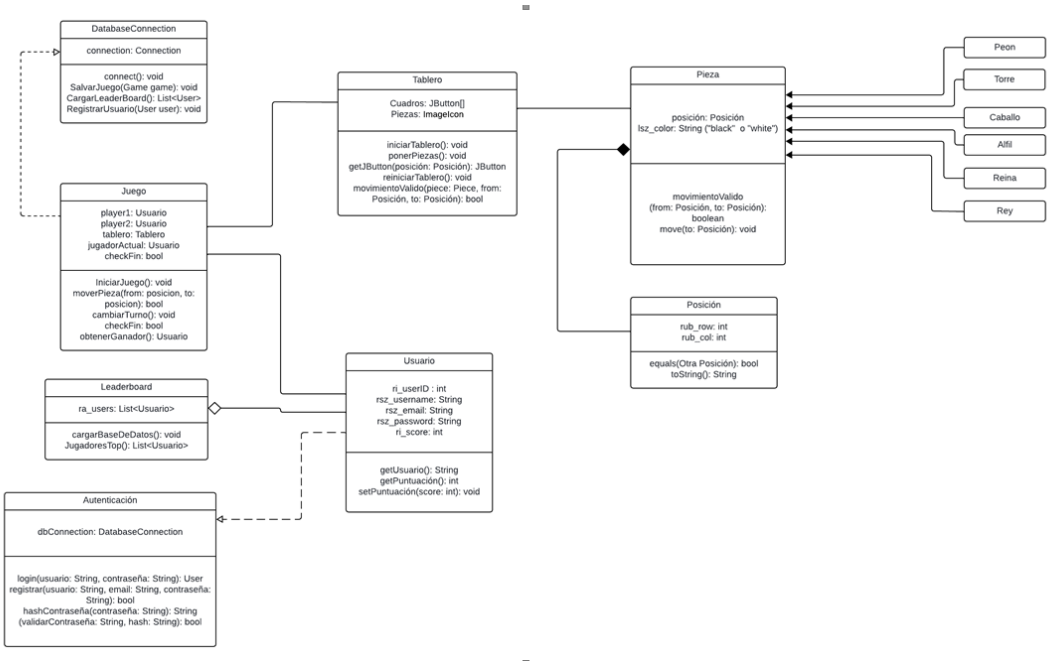


Figura 10.3.2 Diagrama de Clases 2.0, versión mejorada del diagrama de clases. Ahora se hace un análisis más a fondo donde se integran variables y clases específicas que son utilizadas dentro del código, explicando así su relación.

10.4. Diagrama de Máquina de estados

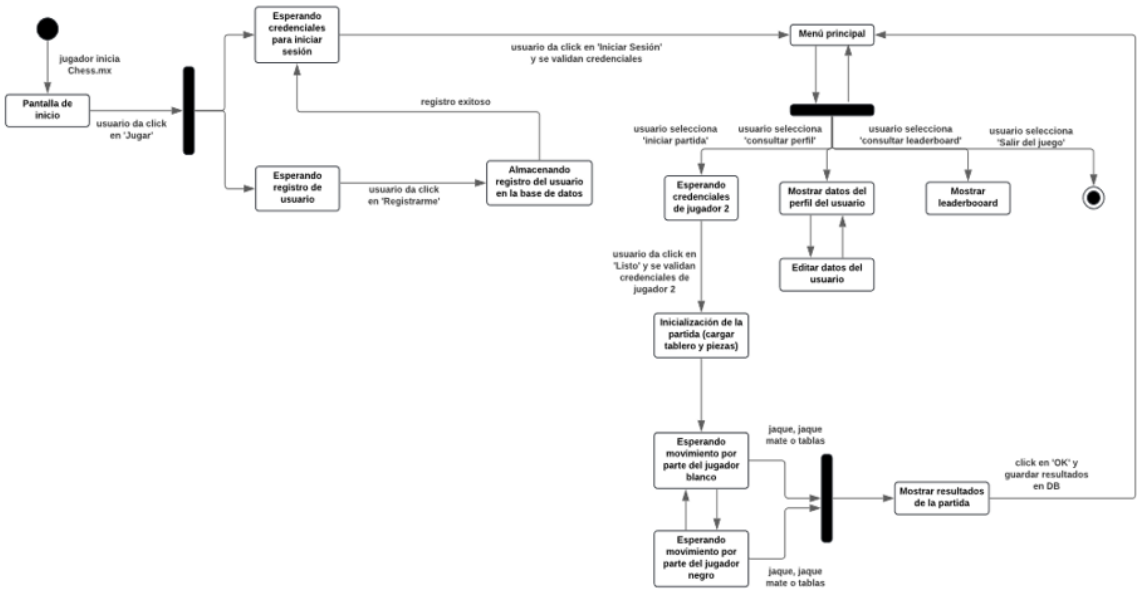


Figura 10.4.1 Diagrama de máquina de estados que describe los distintos estados del juego a medida que el usuario interactúa con él (mediante eventos), mostrando las transiciones que reflejan las decisiones del usuario entre los distintos estados.

11. Reporte de pruebas

11.1 Casos de prueba modulares

Conexión con base de datos	**Status**
SMT_TestCase_1_DB_Connection	**Complete**

Requirements covered

RNF05

Test Objective

- Comprobar la conexión del sistema con la base de datos relacional alojada en el gestor de BD phpMyAdmin.

Test Environment description

- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

1. La clase TestConnection llama al método GetConnection() de la clase DatabaseConnection que permite establecer la conexión a la base de datos.
2. La configuración de acceso debe incluir credenciales válidas de la base de datos.

Expected Results

1. El sistema establece una conexión exitosa a la base de datos sin lanzar excepciones.
2. Como las credenciales ingresadas fueron válidas, se tiene acceso.

Test Fit criteria and test execution results	**Test Results**

Conexión exitosa:
`--- exec:3.1.0:exec (default-cli) @ chess ---`
`Conexion exitosa a la base de datos`
`------------------------------------------------------------------------`
`BUILD SUCCESS`
`------------------------------------------------------------------------`
`Total time:  2.437 s`
`Finished at: 2024-11-05T22:53:55-06:00`
`------------------------------------------------------------------------`
	PASSED

Comments

- La conexión fue establecida con éxito.

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Creación de nuevo usuario	Status
SMT_TestCase_2_Register_And_Login	Complete

Requirements covered

RF01 and RF02

Test Objective

- Verificar que los usuarios puedan crear un usuario e iniciar sesión correctamente con credenciales válidas y que el sistema bloquee intentos de inicio de sesión con credenciales inválidas.

Test Environment description

- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

1. La clase TestUser pedirá un nombre de usuario nuevo y su respectiva contraseña para realizar el registro, y posteriormente se comprobará el inicio de sesión de este usuario.
2. La tabla users alojará los usuarios registrados, y de esta misma tabla se obtendrá la información para el inicio de sesión.
3. El sistema debe validar correctamente el nombre de usuario y la contraseña.

Expected Results

1. El usuario puede realizar se registra y su respectiva contraseña, creando un nuevo de usuario en el sistema.
2. El usuario puede iniciar sesión con credenciales válidas y acceder a su perfil.
3. Si las credenciales son incorrectas o duplicadas, el sistema muestra un mensaje de error y no permite el acceso.

Test Fit criteria and test execution results	Test Results
Registro exitoso:	PASSED

```
] --- exec:3.1.0:exec (default-cli) @ chess ---
 Registro de nuevo usuario:
 Nombre de usuario: angelll
 Correo: angelll@gmail.com
 Password: 1234
 Registro exitoso.
```

Inicio de sesión exitoso

```
Iniciar sesion:
Correo: angelll@gmail.com
Password: 1234
Inicio de sesion exitoso.
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```

Error por credenciales incorrectas al iniciar sesión:

```
Iniciar sesion:
Correo: angel@gmail.com
Password: 123456
Error en el inicio de sesion. Verifica tus credenciales.
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```

Comments

- El registro de un nuevo usuario y el inicio de sesión fueron exitosos.

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Carga de tablero de ajedrez	Status
SMT_TestCase_3_Board	Complete

Requirements covered

RF07

Test Objective

- Comprobar la carga de un tablero de ajedrez en conjunto de las piezas al iniciar una partida mediante la función iniciarJuego() e iniciarTablero(), cumpliendo con los requisitos y características iniciales de este juego.

Test Environment description

- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

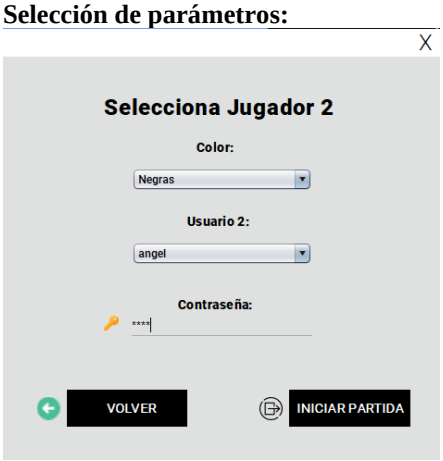
1. Al estar dentro de la ventana de Elegir jugador 2 se seleccionan los parámetros necesarios.
2. Una vez que se valide la contraseña del segundo usuario y el color de piezas se inicia el juego
3. Ahora se ejecuta el main() del archivo TableroAjedrez.
4. Se genera un tablero de ajedrez de 8x8 con las respectivas características elegidas.

Expected Results

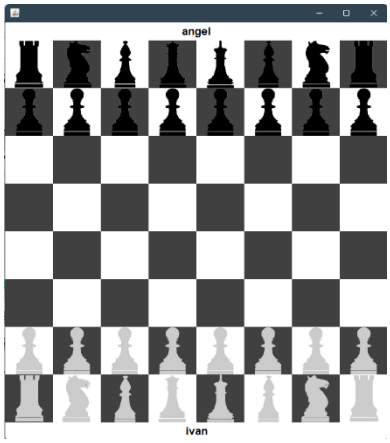
1. Se selecciona el jugador 2 y el color que este tendrá de piezas.
2. Se ingresa la contraseña del segundo usuario y al ser correcta se puede iniciar el juego.
3. Se abre la nueva ventana que genera el archivo TableroAjedrez.
4. Se genera el tablero de ajedrez completamente funcional para poder jugar entre usuarios.

Test Fit criteria and test execution results

Selección de parámetros:



Tablero:



Test Results

PASSED

Comments

- El tablero se mostró exitosamente al iniciar una partida.

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Promoción de peón	**Status**
SMT_TestCase_4_Pawn_Promotion	**Complete**

Requirements covered

RF08 and RF09

Test Objective

- Comprobar el funcionamiento correcto del movimiento especial “Promoción de Peón”, mediante la función coronacionPeon().

Test Environment description

- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

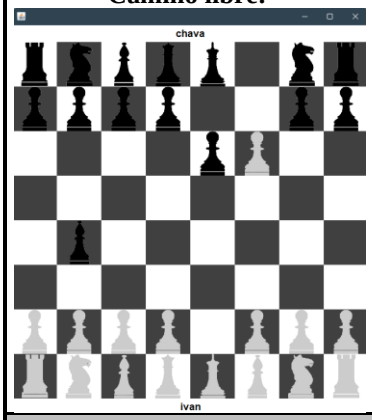
1. Ejecutar Chess.mx desde Netbeans.
2. Jugar de manera que un peón tenga el camino libre para llegar al lado contrario del tablero.
3. Una vez el peón del color contrario llegue al lado opuesto se podrá elegir la promoción indicada.
4. Al elegir la promoción especifica el peón se convierte en esa pieza y tiene sus movimientos actualizados.

Expected Results

1. El programa de Chess.mx se ejecuta correctamente y se inicia sesión y se sigue el flujo hasta la partida.
2. Se genera un camino libre para el peón hacia el lado opuesto.
3. Al llegar el peón al lado contrario aparece un menú donde seleccionara la pieza especifica.
4. Una vez elegida se observa como el peón se convierte en la pieza elegida.

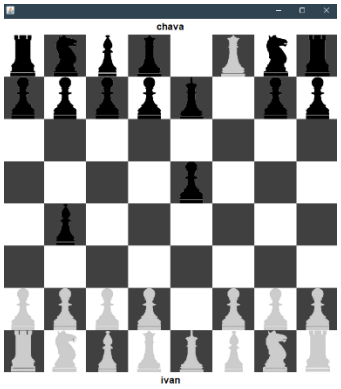
Test Fit criteria and test execution results

Camino libre:



Menú de selección:

Cambio de peón:



Test Results

PASSED

Comments

N/A

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Captura al paso	**Status**
SMT_TestCase_5_En_Passant	**Complete**

Requirements covered

RF08 and RF09

Test Objective

- Comprobar uno de los movimientos especiales del juego de ajedrez, en este caso es la captura al paso, mediante la función capturaAlPaso().

Test Environment description

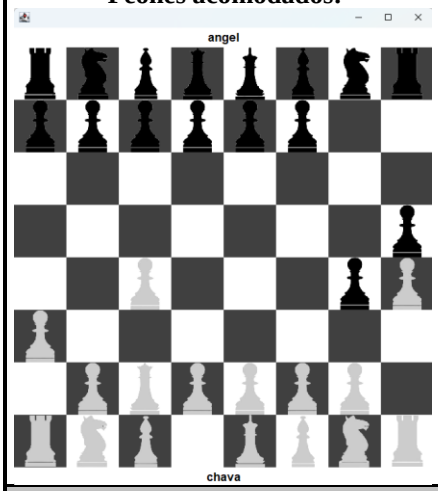
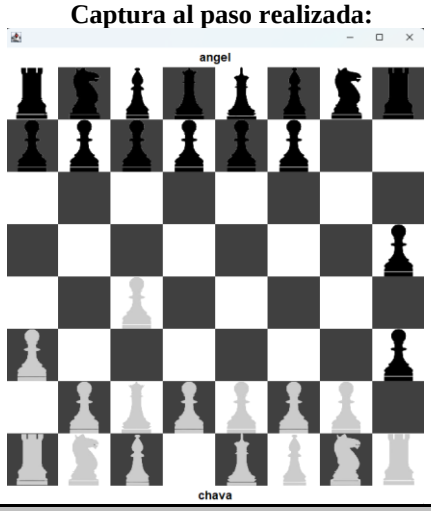
- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

1. Iniciar una nueva partida
2. Mover los peones relacionados con la prueba a sus respectivas posiciones.
3. Realizar una captura al paso.

Expected Results

1. La partida se inició correctamente.
2. Se acomodaron los peones a sus posiciones en las cuales se puede comprobar que la captura al paso sea funcional.
3. Se llevó a cabo el movimiento especial de captura al paso.

Test Fit criteria and test execution results	**Test Results**
Peones acomodados:  **Captura al paso realizada:** 	PASSED

Comments

N/A

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

11.2 Casos de prueba de Integración

Integración entre Autenticación y Acceso a las Interfaces	**Status**
SIT_TestCase_1_Auth_And_Access_To_Interfaces	**Complete**

Requirements covered

RF02

Test Objective

- Comprobar la correcta integración de los módulos de inicio de sesión y las distintas interfaces del sistema.

Test Environment description

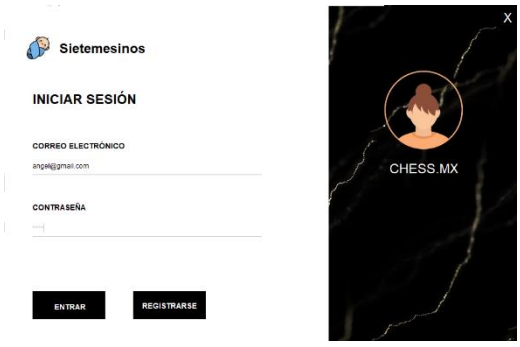
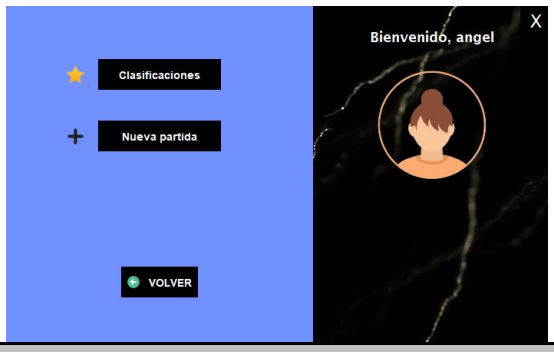
- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

1. Ejecutar Chess.mx desde Netbeans.
2. Iniciar sesión con credenciales válidas.
3. Redirigir a la página principal.
4. Acceder a las interfaces.

Expected Results

1. La aplicación de Chess.mx se ejecutó y abrió correctamente.
2. La sesión fue iniciada con datos alojados en la base de datos.
3. Se redirigió a la página principal después de iniciar sesión.
4. Se pudo acceder a las interfaces que aparecen en la página de inicio.

Test Fit criteria and test execution results	**Test Results**
Inicio de Sesión  **Página Principal** 	PASSED

Comments

- NA

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Validación de Selección de Jugadores	Status
SIT_TestCase_2_Player_Selection	Complete

Requirements covered

RF03

Test Objective

- Comprobar la correcta validación de los jugadores 1 y 2.

Test Environment description

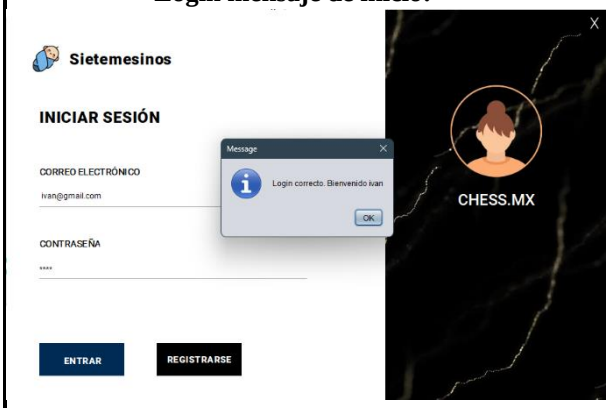
- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

1. Ejecutar Chess.mx desde Netbeans.
2. El jugador 1 inicia sesión dentro del login con sus credenciales previamente registradas.
3. El jugador 1 inicia una nueva partida.
4. Selección de Jugador 2 con credenciales correctas.
5. Selección de color para jugador 2.
6. Inicio de partida

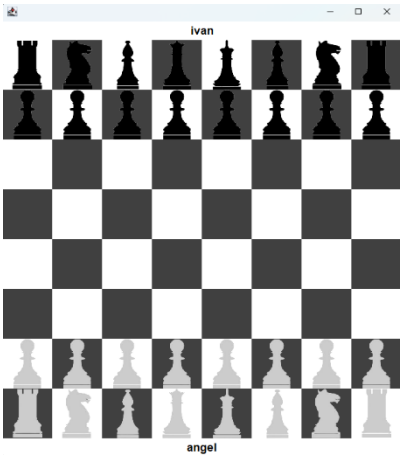
Expected Results

1. El programa de Chess.mx se ejecuta correctamente y se abre la pestaña de login.
2. El jugador 1 inició sesión correctamente.
3. Se presionó el botón de nueva partida y se abrió la pestaña de selección de jugador 2.
4. Se escribieron las credenciales correctas del jugador 2 según la base de datos.
5. Se seleccionó el color Negro para el jugador 2.
6. La partida fue iniciada.

Test Fit criteria and test execution results	Test Results
Login mensaje de inicio:  **Credenciales incorrectas de jugador 2:**	PASSED

Credenciales correctas de jugador 2:

Inicio de partida:



Comments

- Se seleccionaron correctamente los jugadores 1 y 2 para el inicio de una partida de ajedrez, ingresando sus respectivas credenciales.

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Creación de nuevas partidas en BD

Status

Complete

SIT_TestCase_3_Game_Creation_In_DB

Requirements covered

RF06 and RF11

Test Objective

- Comprobar que, al iniciar una nueva partida, esta se refleje correctamente en la base de datos, haciendo las relaciones a los jugadores que se están enfrentando en ella.

Test Environment description

- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

1. Ingresar a una nueva partida.
2. Verificar que la partida se generó correctamente en la base de datos con los jugadores relacionados a ella.

Expected Results

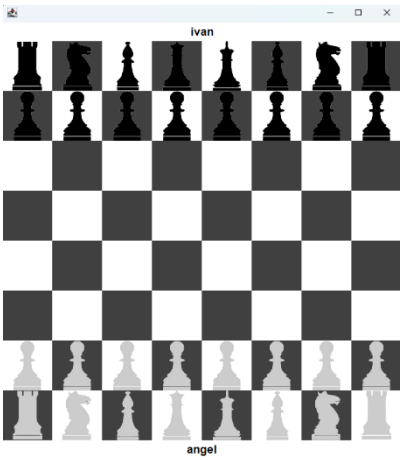
1. La partida se inició exitosamente.
2. En la base de datos, se generó una nueva fila en la tabla ‘games’, donde se especifica el jugador de piezas negras, el jugador de piezas blancas, la fecha, pero sin un jugador relacionado en el campo de ganador.

Test Fit criteria and test execution results

Test Results

PASSED

Inicio de partida



Creación de juego en BD

Opciones extra

	game_id	player_id_white	player_id_black	winner_id	game_date
Editar Copiar Borrar	61	7	8	NULL	2024-11-25

Comments

- NA

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Partida de Ajedrez	**Status**
SIT_TestCase_4_Chess_r	**Complete**

Requirements covered

RF07, RF08, RF09 and RF10

Test Objective

- Comprobar que el juego de ajedrez se lleve a cabo de principio a fin, siguiendo las reglas tradicionales de un juego de ajedrez, y mostrando el resultado obtenido.

Test Environment description

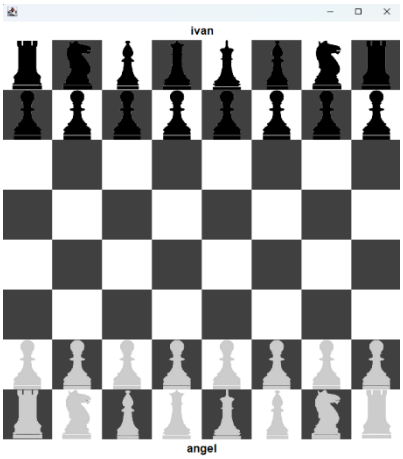
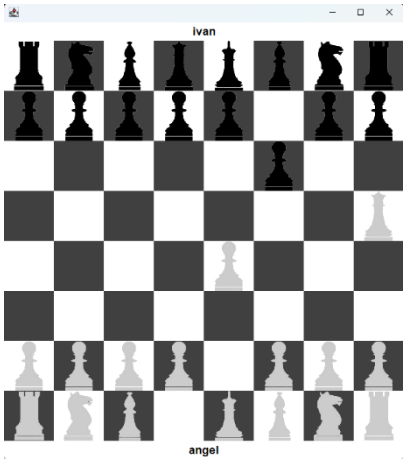
- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

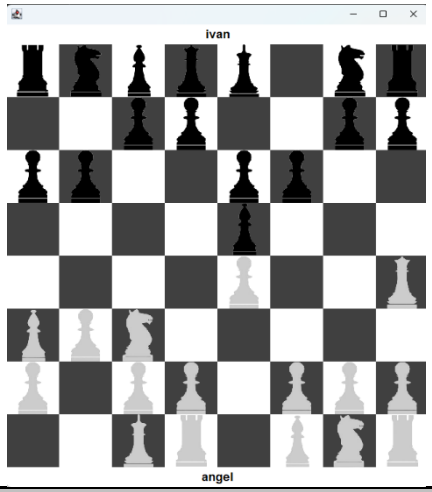
1. Iniciar un nuevo juego.
2. Mover las piezas según el turno del jugador al que le corresponda.
3. Probar movimientos especiales.
4. Comprobar resultado de partida.

Expected Results

1. El juego se inició correctamente.
2. Según el jugador al que le iba tocando en cada turno, se movían las piezas que se requirieran siguiendo las reglas tradicionales del ajedrez.
3. Al momento de realizar movimientos especiales, el juego permitió llevarlos a cabo.
4. Se obtuvo el resultado de victoria por parte del jugador negro.

Test Fit criteria and test execution results	**Test Results**
Inicio de partida  **Piezas movidas de posiciones iniciales** 	PASSED

Movimiento especial: enroque



Jaque Mate

Comments

- N/A

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Tabla de Clasificaciones	Status
SIT_TestCase_5_Leaderboard	Complete

Requirements covered

RF04 and RF05

Test Objective

- Verificar que los resultados de cada partida se actualicen en la base de datos, y que estos mismos se puedan reflejar en la tabla de clasificaciones que muestra a los jugadores con mayor cantidad de victorias.

Test Environment description

- **Entorno:** Sistema operativo Windows 11
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin
- **IDE:** NetBeans v23.

Test Requirement Details

1. Observar la tabla de clasificaciones antes de iniciar una nueva partida.
2. Iniciar y finalizar una partida con algún jugador victorioso.
3. Visualizar el mensaje de victoria.
4. Verificar que el contador en la leaderboard se ha actualizado.

Expected Results

1. Se observó que, en la tabla de clasificaciones, el jugador chava únicamente cuenta con dos victorias en su historial.
2. Chava, siendo el jugador de las piezas negras, obtuvo la victoria.
3. El mensaje de victoria apareció luego de la partida.
4. Se observó el cambio en la tabla de clasificaciones, aumentando el contador de partidas ganadas del jugador chava.

Test Fit criteria and test execution results		Test Results
Tabla de clasificaciones antes de iniciar partida	**Finalizar partida con jugador chava victorioso**	PASSED

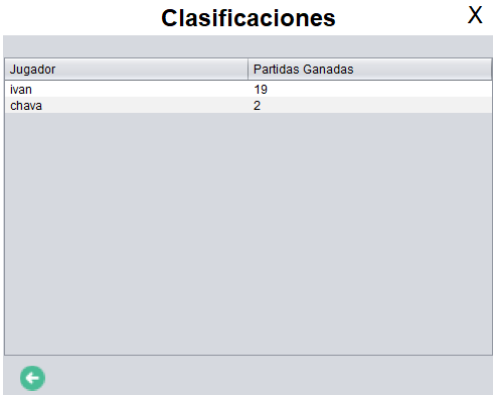


Tabla de clasificaciones actualizada

Comments

- NA

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

11.3 Casos de prueba de Sistema

Autenticación de Sesión	**Status**
ST_TestCase_1_Authentication	**Complete**

Requirements covered

RNF04

Test Objective

El objetivo de la prueba es verificar que, una vez el jugador se haya registrado en Chess.mx, su contraseña se almacene en la base de datos y .

Test Environment description

- **Entorno:** Sistema operativo Windows 10.
- **Base de datos:** MySQL v8. 0.30
- **DBMS:** phpMyAdmin

Test Requirement Details

1. Ejecutar Chess.mx.jar.
2. Clic en botón “Registrarse”.
3. Llenar campos de nombre de usuario, correo y contraseña.
4. Dar clic en botón “Guardar”.

Expected Results

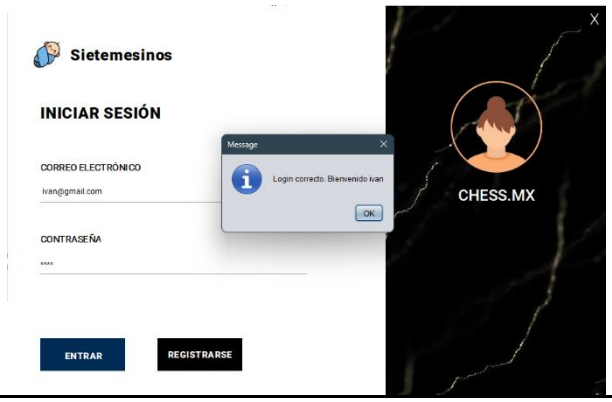
1. Chess.mx corriendo en primer plano
2. Se muestra ventana para registrar usuario.
3. N/A
4. Aparece mensaje de registro de usuario exitoso y la contraseña de cifra en la base de datos.

Test Fit criteria and test execution results	**Test Results**
Result: N/A	PASSED

Comments

- Puede verificar que la contraseña esté cifrada desde phpMyAdmin, consultando la tabla *users*.

user_id	username	email	password
7	angel	angel@gmail.com	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e...
8	ivan	ivan@gmail.com	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e...
9	mrchava	chava@gmail.com	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e...
12	profe	profe@gmail.com	03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e...
13	usuarioprueba	usuarioprueba@gmail.com	cdcb7422ca0fe077931b84e6fb7e6dfb7d6678dc7e9ae9c433...



References
• Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Jugador puede realizar movimientos válidos en el tablero	Status
ST_TestCase_2_User_Can_Do_Valid_Movements_Of_Pieces	Complete

Requirements covered

RF08

Test Objective

El objetivo de la prueba es verificar que durante una partida en Chess.mx, el jugador puede realizar movimientos válidos de las diferentes piezas y se actualice el tablero, y, si el movimiento no es válido, se muestre un mensaje de error y no se realice el movimiento.

Test Environment description

- **Entorno:** Sistema operativo Windows 10.

Test Requirement Details

1. Ejecutar Chess.mx.jar.
2. Ingresar campos de correo electrónico y contraseña válidos (es decir, un usuario ya registrado).
3. Clic en botón “Entrar”.
4. Dar clic en botón “Nueva Partida”.
5. Ingresar campos correspondientes al jugador 2 (usuario, contraseña y color que utilizará en la partida).
6. Clic en botón “Iniciar Partida”.
7. Jugador blanco intenta mover cualquier peón blanco dos casillas hacia adelante.
8. Jugador negro intenta mover cualquier peón negro una casilla hacia adelante.
9. Jugador blanco intenta mover un caballo blanco ubicado en (8,2) a casilla (6,4).

Expected Results

1. Chess.mx corriendo en primer plano
2. N/A
3. Aparece mensaje de inicio de sesión exitoso.
4. Aparece ventana para comenzar una nueva partida de ajedrez.
5. N/A
6. Se carga el tablero de ajedrez de 8x8 con las piezas de ambos colores en sus respectivas posiciones iniciales.
7. Sistema permite el movimiento porque es válido.
8. Sistema permite el movimiento porque es válido.
9. Sistema no permite el movimiento porque no es válido y muestra un mensaje de error.

Test Fit criteria and test execution results	Test Results
Result: N/A	PASSED

Comments

El sistema permitirá el movimiento siempre y cuando sea válido de acuerdo a las reglas del ajedrez.

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Mostrar el resultado de la partida en un escenario de Jaque Mate	**Status**
ST_TestCase_3_Show_Check_Mate_Result	**Complete**

Requirements covered

RF10 and RF03

Test Objective

El objetivo de la prueba es verificar que durante una partida habitual en Chess.mx, el jugador negro deje en Jaque Mate al jugador blanco, con lo cual se mostraría un mensaje indicando que las piezas negras han ganado la partida.

Test Environment description

- **Entorno:** Sistema operativo Windows 10.

Test Requirement Details

1. Ejecutar Chess.mx.jar.
2. Ingresar campos de correo electrónico y contraseña válidos (es decir, un usuario ya registrado).
3. Clic en botón “Entrar”.
4. Dar clic en botón “Nueva Partida”.
5. Ingresar campos correspondientes al jugador 2 (usuario, contraseña y color que utilizará en la partida).
6. Clic en botón “Iniciar Partida”.
7. Se juega una partida habitual entre jugador de piezas blancas y negras.

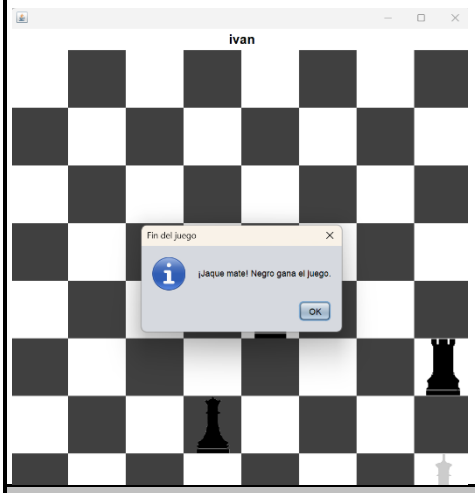
Expected Results

1. Chess.mx corriendo en primer plano
2. N/A
3. Aparece mensaje de inicio de sesión exitoso.
4. Aparece ventana para comenzar una nueva partida de ajedrez.
5. N/A
6. Se carga el tablero de ajedrez de 8x8 con las piezas de ambos colores en sus respectivas posiciones iniciales.
7. Jugador de piezas negras hace Jaque Mate a jugador de piezas blancas. Se muestra un mensaje en la pantalla indicando que fue un Jaque Mate y que el jugador ganador fue el de las piezas negras.

Test Fit criteria and test execution results	**Test Results**
Result: N/A	PASSED

Comments

El jaque mate es el resultado más común en una partida de ajedrez, aunque también se puede dar un escenario de Tablas por Material Insuficiente, pero está fuera del alcance de esta prueba.



References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Mostrar la tabla de clasificación	**Status**
ST_TestCase_4_Show_LeaderBoard	**Complete**

Requirements covered

RF05

Test Objective

El objetivo de la prueba es verificar que el sistema muestre correctamente la tabla de clasificación con los mejores jugadores en base a sus partidas ganadas.

Test Environment description

- **Entorno:** Sistema operativo Windows 10.

Test Requirement Details

1. Ejecutar Chess.mx.jar.
2. Ingresar campos de correo electrónico y contraseña válidos (es decir, un usuario ya registrado).
3. Clic en botón “Entrar”.
4. Dar clic en botón “Clasificaciones”.

Expected Results

1. Chess.mx corriendo en primer plano
2. N/A
3. Aparece mensaje de inicio de sesión exitoso.
4. Aparece ventana donde se muestra la tabla de clasificación con los mejores jugadores en base a sus partidas ganadas.

Test Fit criteria and test execution results	**Test Results**
Result: N/A	PASSED

Comments

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

Garantizar tiempos de respuesta rápidos para el movimiento de las piezas	**Status**
ST_TestCase_5_Fast_Response_Of_Pieces	**Complete**

Requirements covered

RNF01

Test Objective

El objetivo de la prueba es verificar que el sistema no tarde más de tres segundos cuando el usuario realice el movimiento de alguna pieza, con el fin de brindarle una experiencia de usuario agradable.

Test Environment description

- **Entorno:** Sistema operativo Windows 10.

Test Requirement Details

1. Ejecutar Chess.mx.jar.
2. Ingresar campos de correo electrónico y contraseña válidos (es decir, un usuario ya registrado).
3. Clic en botón “Entrar”.
4. Dar clic en botón “Nueva Partida”.
5. Ingresar campos correspondientes al jugador 2 (usuario, contraseña y color que utilizará en la partida).
6. Clic en botón “Iniciar Partida”.
7. Se juega una partida habitual entre jugador de piezas blancas y negras, verificando que el tiempo de respuesta del movimiento de cualquier pieza sea menor a tres segundos.

Expected Results

1. Chess.mx corriendo en primer plano
2. N/A
3. Aparece mensaje de inicio de sesión exitoso.
4. Aparece ventana para comenzar una nueva partida de ajedrez.
5. N/A
6. Se carga el tablero de ajedrez de 8x8 con las piezas de ambos colores en sus respectivas posiciones iniciales.

7. Experiencia de usuario no afectada.

Test Fit criteria and test execution results	Test Results
Result: N/A	PASSED

Comments

N/A

References

- Ivanpillow. (2024). Prueba-Ing-Software. Repositorio de GitHub. Recuperado de https://github.com/Ivanpillow/Prueba-Ing-Software

11.4 Total de pruebas

Al finalizar el desarrollo del proyecto, se llevaron a cabo un total de 15 pruebas con el objetivo de garantizar la calidad y el correcto funcionamiento del sistema. Estas pruebas se clasificaron en tres categorías principales descritas en la siguiente tabla:

Tipo de prueba	Descripción	Pruebas pasadas	Pruebas fallidas	Total de pruebas ejecutadas
Pruebas modulares	Verificar el correcto desempeño de componentes individuales y funciones específicas del sistema.	5	0	5
Pruebas de integración	Evaluar la interacción entre diferentes módulos para asegurar la correcta transmisión de datos entre ellos.	5	0	5
Pruebas de sistema	Comprobar el comportamiento global de la aplicación en situaciones reales de uso, simulando escenarios integrales y posibles casos de error.	5	0	5

Todas las pruebas realizadas fueron exitosas, obteniendo el estado PASSED en cada caso, lo que indica que los criterios establecidos para su ejecución y evaluación fueron cumplidos satisfactoriamente. Esto respalda la robustez, funcionalidad y confiabilidad de la solución desarrollada.

12. Manual de usuario

Clic aquí para ver el documento del manual de usuario.

13. Conclusiones

Diego Iván Hernández Muñoz:

Este fue un proyecto en general con experiencias muy amenas y satisfactorias, decidimos irnos por un lenguaje el cual fuera diferente y presentara algún tipo de reto por ello elegimos JAVA, también decimos irnos por una aplicación tipo juego para no hacer un proyecto tan relacionado a algún negocio.

Aprendí bastante en cuanto a la documentación ya que fue un proceso bastante extenso a lo largo de la elaboración de este proyecto, pero este mismo nos ayudó a mejorar la comunicación en equipo, y poder delegar actividades, errores y comentarios entre nosotros de manera eficiente, todavía pienso que el realizar tantos diagramas es muy extenuante pero dentro de las explicaciones del profesor le encontré sentido, el poder explicar el programa de manera general o ya sea muy específica llegando a interacciones físicas, variables y funciones, un aspecto el cual no tenía en mente antes.

Regresando al proyecto, según el timing plan que teníamos el tiempo era más que suficiente si íbamos al pie de la letra, pero al no tener en cuenta del todo varios aspectos externos como otros proyectos o exámenes de otras materias influyeron a tener ciertos contratiempos, pero con el trabajo en equipo y una buena división del trabajo pudimos lograrlo.

Angel Alexander Ramírez Navarro:

La realización de este proyecto representó un reto significativo para todo nuestro equipo, ya que ninguno de nosotros tenía experiencia previa con el lenguaje de programación JAVA. Sin embargo, contábamos con la ventaja de tener conocimientos en C y C++, lenguajes cuya forma de programar es similar a JAVA, lo que facilitó nuestra adaptación.

Aunque no fui el encargado de implementar gran parte de la interfaz gráfica (GUI) y la experiencia de usuario (UX), me concentré en desarrollar los movimientos básicos de las piezas de ajedrez, así como algunos movimientos especiales. Al principio, pensé que estos movimientos serían bastante complejos de implementar, pero sorprendentemente resultaron ser más sencillos de lo que imaginaba.

En conjunto con la interfaz gráfica y la correcta implementación del juego en JAVA, logramos ofrecer una experiencia de usuario agradable. Además, todos sentimos una gran satisfacción personal al haber completado este proyecto en un lenguaje nuevo para nosotros, lo que nos permitió aprender y superar un desafío significativo.

Luis Angel De La Cruz Ascencio:

El principal objetivo por el cual decidimos como equipo llevar a cabo este proyecto, además de el de crear un juego de ajedrez en lenguaje Java, fue para irnos familiarizando con el lenguaje de Java, ya que ninguno de nosotros los integrantes del equipo habíamos estado tan adentrados en este lenguaje, y las únicas veces que habíamos hecho uso de él fue de forma breve.

Al final se cumplió el objetivo, pudimos conocer e implementar las características básicas y algunas no tan básicas en una aplicación de ajedrez hecha a base de jFrames de Java. Cada ventana de la aplicación es un formulario que interactúa de una forma dinámica con la lógica del programa que se encuentra detrás. Durante el desarrollo, pudimos integrar funcionalidades como la gestión de los usuarios, el proceso y registro de partidas y la visualización de un sistema de clasificaciones generales de todos los jugadores.

El proyecto no solo cumplió su objetivo inicial, sino que también sirvió como una experiencia bastante completa para comprender mejor el desarrollo de aplicaciones gráficas en Java, su interacción con las bases de datos y la importancia del diseño de este tipo de lenguajes para el mundo laboral.