

Propósito Projeto CoChess:

O projeto CoChess foi desenvolvido com o objetivo de proporcionar uma experiência de xadrez cooperativo e interativo para os usuários. A aplicação permite partidas online, sugestões de jogadas, histórico de partidas e gerenciamento de perfil. Este documento apresenta os principais artefatos do sistema, incluindo diagramas de classes e telas da aplicação.

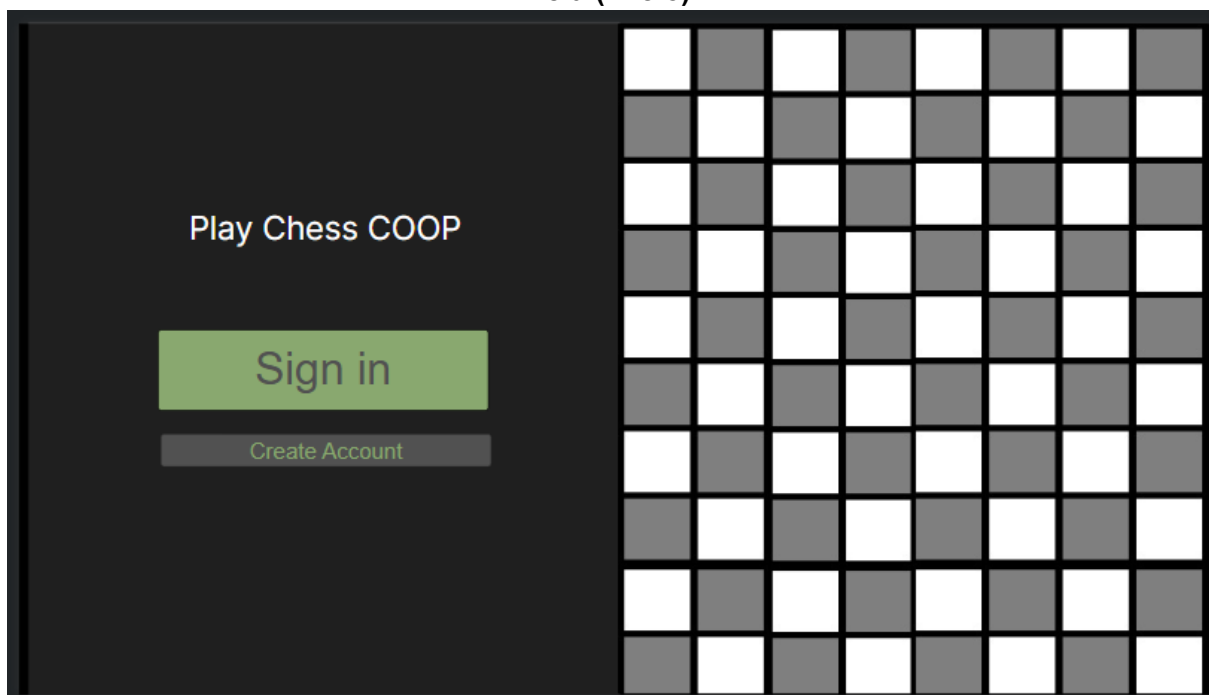
Projeto : CoChess

Feito por :

Matheus Vidal Pereira, João Caio Oliveira Lins e João Henrique Rodrigues Lopes

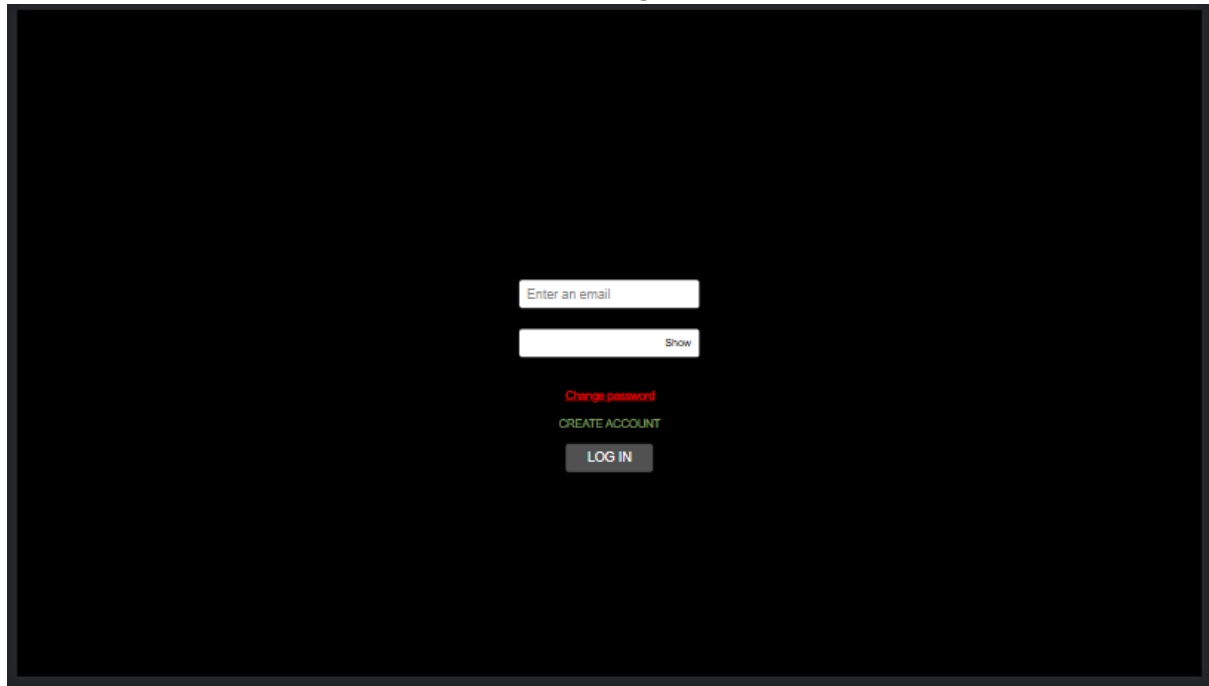
Diagrama de Classes

Tela (Início)



Classe Usuario: A tela inicial pode exibir opções de login ou cadastro, onde os usuários devem se autenticar.

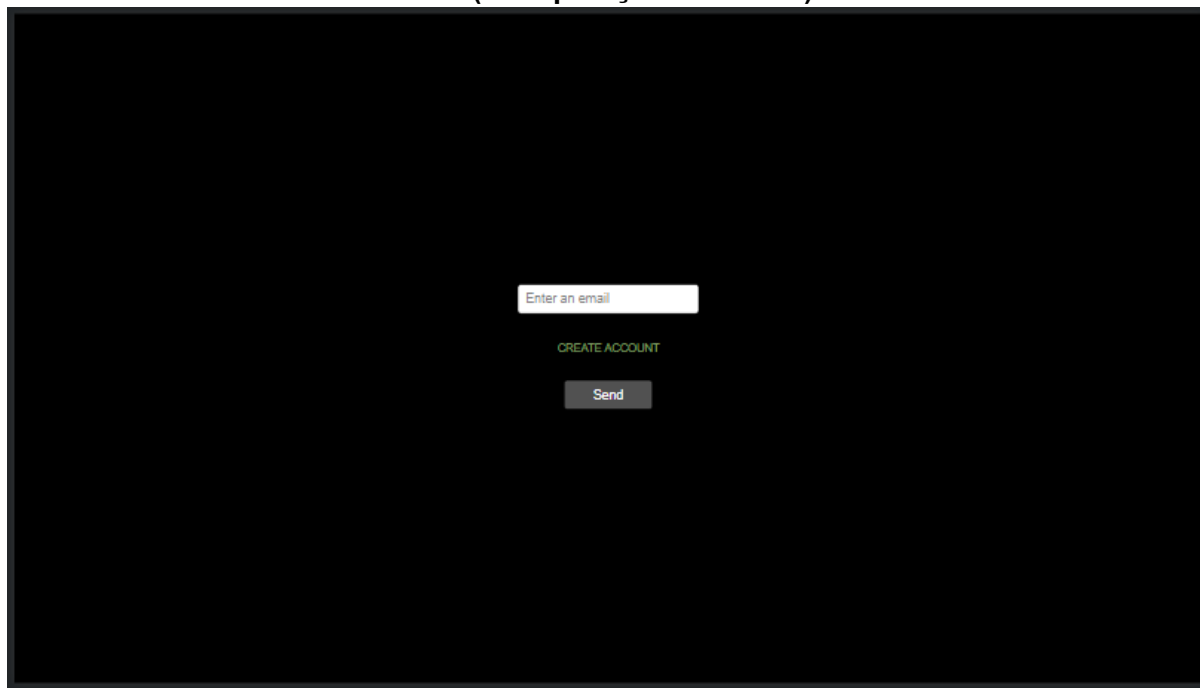
Tela (Log in)

A login screen with a dark gray background. In the center, there are two white input fields. The top field contains the placeholder text "Enter an email". Below it is a second white field with a small "Show" link to its right. Underneath these fields are three links: "Change password" in red, "CREATE ACCOUNT" in green, and a gray "LOG IN" button.

Classe Usuario: Aqui, o usuário autentica seu acesso ao sistema

- Atributos: id, nome, email, senha
- Métodos: cadastrar(), autenticar(), recuperarSenha(), editarPerfil()

Tela (Recuperação de Senha)



Enter an email

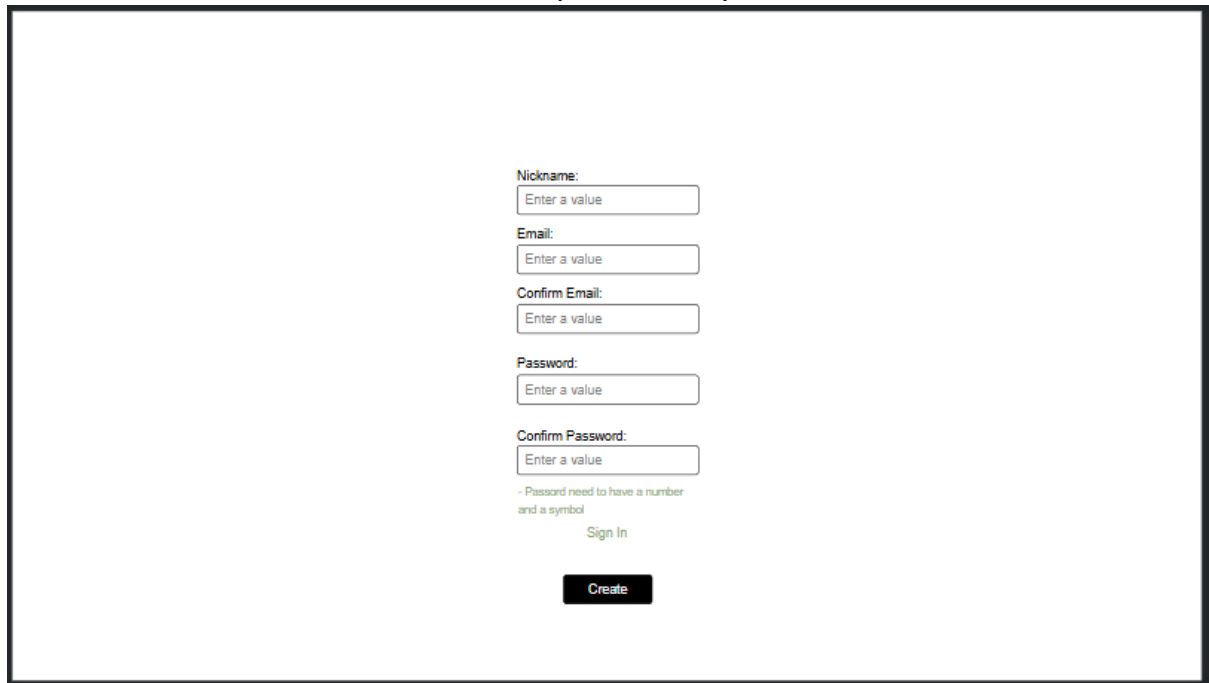
[CREATE ACCOUNT](#)

Send

Classe Usuario: Utiliza recuperarSenha() para redefinir sua senha a partir do email cadastrado.

- Atributos: id, nome, email, senha
- Métodos: cadastrar(), autenticar(), recuperarSenha(), editarPerfil()

Tela (Criar Conta)



A user registration form titled "Tela (Criar Conta)". It contains five input fields for "Nickname:", "Email:", "Confirm Email:", "Password:", and "Confirm Password:", each with a placeholder "Enter a value". Below the password field is a green note: "- Password need to have a number and a symbol". A "Sign In" link is positioned below the note, and a black "Create" button is at the bottom center.

Nickname:

Email:

Confirm Email:

Password:

Confirm Password:

- Password need to have a number
and a symbol

[Sign In](#)

[Create](#)

Classe Usuario: Chama `cadastrar()` para permitir que novos usuários entrem no sistema.

- Atributos: id, nome, email, senha
- Métodos: `cadastrar()`, `autenticar()`, `recuperarSenha()`, `editarPerfil()`

Tela (Mudar Senha)

The screenshot shows a web form for changing a password. It is centered on a white background within a black rectangular frame. The form consists of the following elements from top to bottom: a label 'Password:' followed by a text input field containing the placeholder 'Enter a value'; a label 'Confirm Password:' followed by another text input field with the same placeholder; a green error message '- Password need to have a number and a symbol' (note the typo 'need'); a green link 'Sign In'; and a black button with the white text 'Change'.

Classe Usuario: Utiliza `editarPerfil()` para alterar a senha e garantir a segurança da conta.

- Atributos: id, nome, email, senha
- Métodos: `cadastrar()`, `autenticar()`, `recuperarSenha()`, `editarPerfil()`

Tela (Editar Perfil)

Nickname:

Password:

Confirm Password:

- Password need to have a number and a symbol

Edit

Classe Usuario : Permite modificar nome, email e senha através de `editarPerfil()`

- Atributos: id, nome, email, senha
- Métodos: `cadastrar()`, `autenticar()`, `recuperarSenha()`, `editarPerfil()`

Tela (Lobby)

PLAY

History

Points:
9999

Search: Enter a value

- Junkioch
- Lordwatar
- Itsvioletn

Edit Profile Self

Classe Jogador: Exibe informações sobre ranking, jogadores disponíveis e opções de jogo.

-

[illegible]

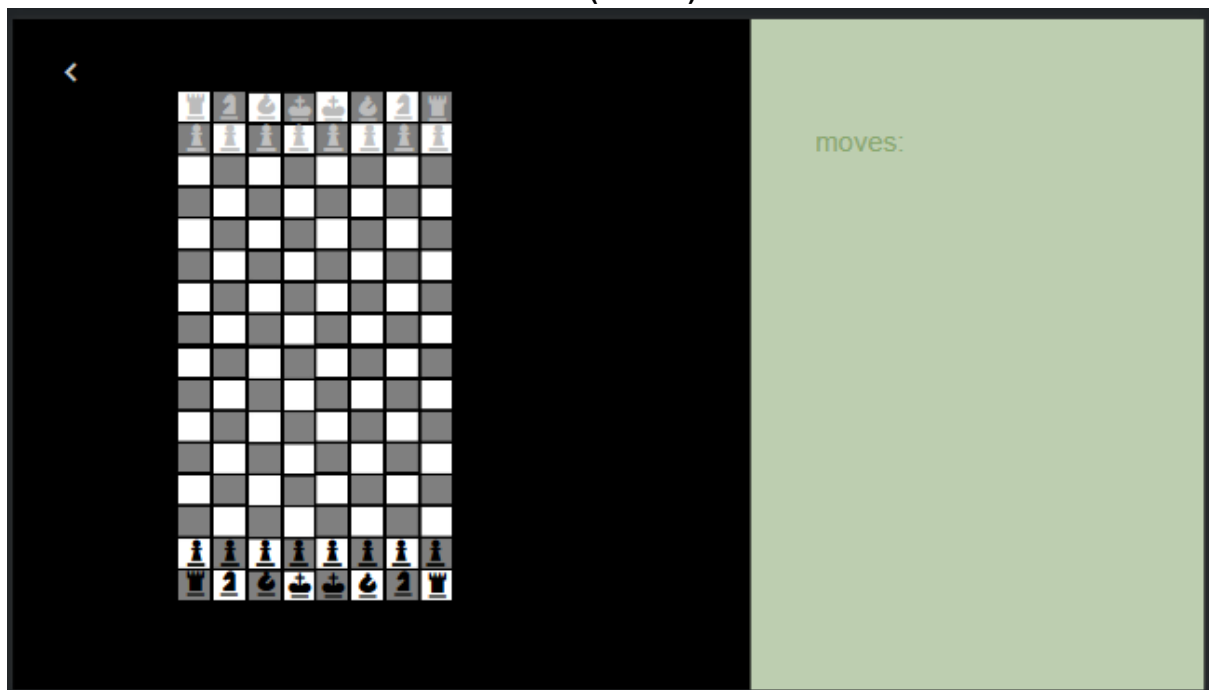
Classe Ranking: Permite análise de desempenho por meio de `visualizarRanking()`.

- Atributos: id, jogador, pontuacao, vitorias, derrotas
- Métodos: `atualizarRanking()`, `visualizarRanking()`

Classe Partida : Apresenta os detalhes das partidas jogadas anteriormente através de `verHistoricoPartida()`.

- Atributos: id, jogadores, tabuleiro, status, historicoMovimentos
- Métodos: `iniciarPartida()`, `finalizarPartida()`, `registrarMovimento()`, `verHistoricoPartida()`

Tela (Treino)



Classe SugestaoJogada : Sugere jogadas para aprimorar as habilidades dos jogadores através de `gerarSugestao()`.

- Atributos: id, jogador, movimentoSugerido
- Métodos: `gerarSugestao()`, `aceitarSugestao()`

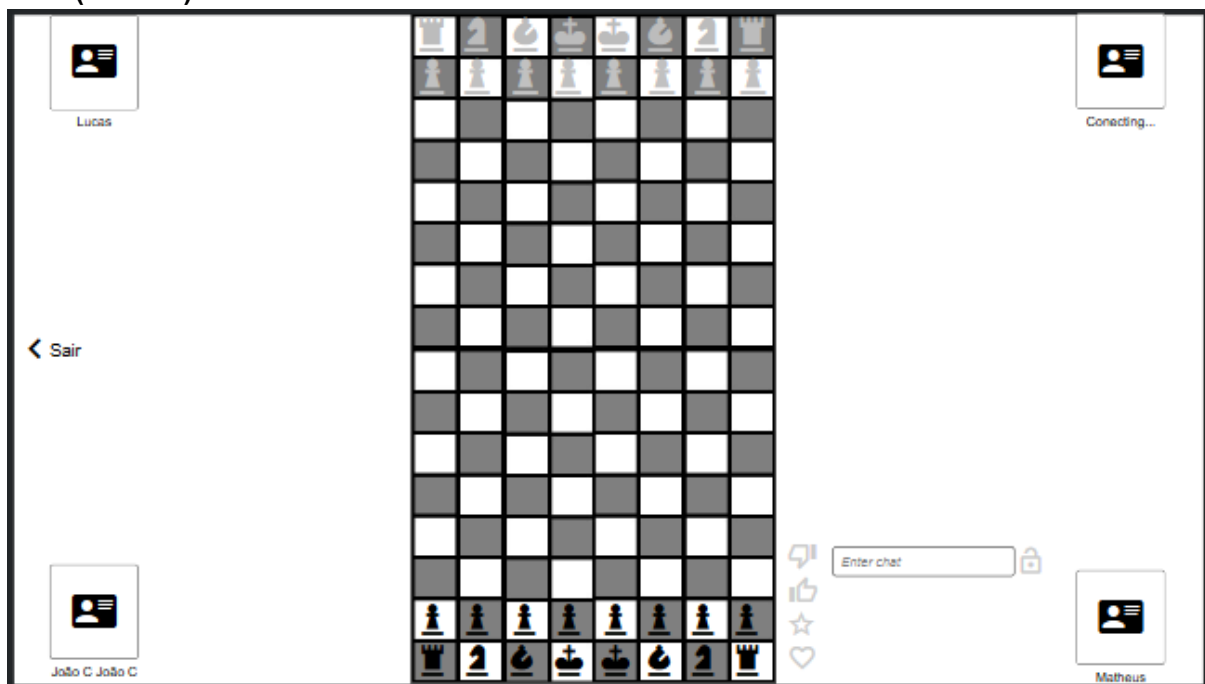
Classe Movimento: Armazena os movimentos realizados no treino com `validarMovimento()`.

- Atributos: id, peca, posicaoInicial, posicaoFinal, valido
- Métodos: validarMovimento(), executarMovimento()

Classe Partida: Gerencia o andamento do jogo com registrarMovimento() e finalizarPartida().

- ✧ Atributos: id, jogadores, tabuleiro, status, historicoMovimentos
- ✧ Métodos: iniciarPartida(), finalizarPartida(), registrarMovimento(), verHistoricoPartida()

Tela (Partida)



Classe SugestaoJogada : Caso ativado, pode fornecer sugestões de jogadas durante a partida..

- Atributos: id, jogador, movimentoSugerido
- Métodos: gerarSugestao(), aceitarSugestao()

Classe Movimento: Armazena os movimentos realizados com validarMovimento().

- Atributos: id, peca, posicaoInicial, posicaoFinal, valido
- Métodos: validarMovimento(), executarMovimento()

Classe Partida: Gerencia o andamento do jogo com registrarMovimento() e finalizarPartida().

- Atributos: id, jogadores, tabuleiro, status, historicoMovimentos

- Métodos: iniciarPartida(), finalizarPartida(), registrarMovimento(), verHistoricoPartida()

Classe Jogador (herda de Usuario)

- Atributos: ranking, time
- Métodos: visualizarRanking()

Código dos diagramas no plantUML:

@startuml

```
class Usuario {
```

```
+id: int
```

```
+nome: String
```

```
+email: String
```

```
+senha: String
```

```
+cadastrar()
```

```
+autenticar()
```

```
+recuperarSenha()
```

```
+editarPerfil(nome: String, email: String, senha: String)
```

```
}
```

```
class Jogador {
```

```
+ranking: int
```

```
+time: String
```

```
+visualizarRanking()
```

}

class Partida {

+id: int

+jogadores: Jogador[4]

+tabuleiro: Tabuleiro

+status: String

+historicoMovimentos: Movimento[]

+iniciarPartida()

+finalizarPartida()

+registrarMovimento()

+verHistoricoPartida()

}

class Movimento {

+id: int

+peca: String

+posicaoInicial: String

+posicaoFinal: String

+valido: boolean

+validarMovimento()

+executarMovimento()

}

class SugestaoJogada {

+id: int

+jogador: Jogador

+movimentoSugerido: Movimento

+gerarSugestao()

+aceitarSugestao()

}

class Ranking {

+id: int

+jogador: Jogador

+pontuacao: int

+vitorias: int

+derrotas: int

+atualizarRanking()

+visualizarRanking()

}

class Administrador {

+gerenciarSistema()

+corrigirErros()

}

Usuario <|-- Jogador

Usuario <|-- Administrador

Partida "1" --> "4" Jogador : Participa

Partida "1" --> "*" Movimento : Registra jogadas

Jogador "1" --> "1" Ranking : Está associado

Jogador "1" --> "*" SugestaoJogada : Pode sugerir jogadas

Movimento "1" --> "1" SugestaoJogada : Ligado às sugestões

@enduml