



Connect Four - Planejamento do Projeto



Estrutura do Projeto

```
bash
CopyEdit
/connect-four
├── /app                                # Nova estrutura do Next.js (App Router)
│   ├── /game                          # Página principal do jogo
│   │   ├── page.jsx                  # Página do jogo
│   │   └── styles.module.css         # Estilos específicos da página
│   ├── /layout.js                    # Layout global da aplicação
│   ├── /globals.css                  # Estilos globais
│   ├── /page.js                      # Página inicial (home)
│   ├── /api                          # Backend para multiplayer (se necessário)
│   │   └── route.js                  # API route para gerenciar partidas
│   └── /components                    # Componentes reutilizáveis
│       ├── Board.jsx                 # Tabuleiro do jogo
│       ├── Cell.jsx                  # Cada célula do tabuleiro
│       ├── Piece.jsx                 # Representação das peças (vermelha/amarela)
│       ├── PlayerInfo.jsx            # Exibe nome e status dos jogadores
│       ├── Timer.jsx                 # Contador de tempo por jogada
│       ├── SpecialCellIndicator.jsx  # Indica células de bônus
│       └── GameOverModal.jsx         # Tela de fim de jogo
├── /hooks                             # Custom Hooks para lógica do jogo
│   ├── useGameLogic.js              # Gerencia regras do jogo
│   └── useTimer.js                   # Gerencia o cronômetro
├── /context                           # Gerenciamento de estado global
│   └── GameContext.jsx               # Estado global do jogo (React Context API)
├── /utils                             # Funções auxiliares
│   ├── checkWinner.js                # Verifica vitória
│   ├── randomUtils.js                # Função para gerar células especiais
│   └── aiMove.js                     # Lógica para jogada do computador
├── /public                            # Assets como imagens, ícones, sons
├── /styles                            # Arquivos de estilo (Tailwind CSS)
│   └── globals.css
├── package.json
├── next.config.js
└── README.md
```



Stack Tecnológica

- ✓ **Next.js 14 (App Router)** → Estrutura moderna com SSR e otimizações.
- ✓ **React 18** → Core da aplicação.
- ✓ **Tailwind CSS** → Estilização rápida e modular.
- ✓ **Framer Motion** → Animação da peça deslizando (prioridade alta).
- ✓ **React Context API** → Gerenciamento de estado do jogo.

Lógica do Jogo



◆ Configuração Inicial

1. O usuário insere os nomes dos jogadores.
2. O primeiro jogador e a cor das peças são sorteados aleatoriamente.
3. O tabuleiro **6x7** é inicializado com células vazias e **5 células especiais aleatórias**.














◆ Regras do Jogo

- ✓ Cada jogador tem **10 segundos** para jogar. Se o tempo acabar, perde a vez.
- ✓ O jogador escolhe uma **coluna**, e a peça desliza até a última célula vazia.
- ✓ Se a peça cair em uma **célula especial**, o jogador ganha uma jogada extra.
- ✓ O jogo termina quando:
 - Alguém faz **4 peças em linha** (horizontal, vertical ou diagonal).
 - O tabuleiro fica cheio (**empate**).

◆ Fim do Jogo

-  Se houver um vencedor ou empate, um **modal aparece com opção de jogar novamente**.
-  Se o jogador escolher jogar de novo, o jogo reinicia com as configurações iniciais.

Funcionalidades Prioritárias (Baseadas na Avaliação)

Funcionalidade	Peso (%)	Prioridade
 Animação da peça deslizando (Framer Motion)	30%	 Alta
 Células especiais (bônus de jogada extra)	15%	 Alta
 Modo 1v1 e Modo vs. CPU (jogada aleatória)	15%	 Alta
 Identificação de fim de jogo/vencedor	10%	 Alta
 Solicitar e exibir nome dos jogadores	10%	 Alta
 Exibir tempo da jogada (10s por turno)	5%	Média
 Aspeto gráfico (design agradável)	10%	Média
 Permitir jogar novamente após o fim do jogo	2.5%	Baixa



Recursos Extras (Se houver tempo)



Efeitos sonoros para jogadas e vitória.



Modo escuro/claro para melhor experiência visual.



Histórico de partidas com pontuação.



Plano de Desenvolvimento

1. Setup do Projeto (Estrutura Inicial)



Criar projeto Next.js com App Router (`npx create-next-app@latest connect-four`).



Instalar dependências (Tailwind CSS, Framer Motion).



Configurar `app/layout.js` e `app/globals.css`.

2. Criar a Lógica Principal do Jogo (Tabuleiro e Peças)



Criar estado do jogo em `GameContext.jsx`.



Criar array bidimensional (6x7) para representar o tabuleiro.



Criar componentes principais:

- `Board.jsx` → Renderiza o tabuleiro.
- `Cell.jsx` → Cada célula do tabuleiro.
- `Piece.jsx` → Peça dos jogadores.
 - ✓ Implementar **inserção de peças corretamente** na coluna escolhida.
 - ✓ Implementar **verificação de vitória e empate** (`checkWinner.js`).

3. Implementar Animação da Peça (Prioridade Máxima - 30%)



Destacar a coluna ao passar o mouse.



Implementar **animação ao cair usando Framer Motion**.



Garantir que a peça desliza corretamente até a última célula vazia.

4. Implementar as Regras de Jogo

- ✓ Criar funcionalidade para **troca de turnos** entre os jogadores.
 - ✓ Implementar **temporizador** (cada jogada tem um limite de 10s).
 - ✓ Criar **lógica para perder a vez** caso o tempo expire.
-

5. Gerar e Implementar Células Especiais (Bônus - 15%)

- ✓ Gerar **5 células aleatórias** no tabuleiro.
 - ✓ Quando uma peça cair nessas células, **o jogador ganha uma jogada extra**.
 - ✓ Estilizar **as células especiais** para serem visualmente distintas.
-

6. Implementar Modo Contra Computador (IA Simples - 15%)

- ✓ Criar `aiMove.js` para definir jogadas do computador:
 - Escolher aleatoriamente **uma coluna válida**.
 - Esperar **1 segundo** antes de jogar (para parecer mais natural).
 - ✓ Garantir que o jogo **alterna entre jogador e computador corretamente**.
-

7. Criar Interface e Melhorar UX

- ✓ Criar `PlayerInfo.jsx` para destacar o jogador da vez.
 - ✓ Criar `Timer.jsx` para mostrar a contagem regressiva.
 - ✓ Criar `GameOverModal.jsx` para exibir o vencedor e opção de reiniciar.
 - ✓ Estilizar com **Tailwind CSS** para um **design agradável**.
-

8. Finalização e Testes

- ✓ Testar **todas as regras do jogo** para evitar bugs.
- ✓ Revisar **animações e experiência do usuário**.
- ✓ Melhorar detalhes visuais e **polir a interface**.