Decentralized Connect4: A Blockchain-Based Gaming Experience

Genesis Anne Villar (Undergraduate)

Computer Science, San Diego State University

CS596: Fundamentals of Cryptography with Applications to Blockchain

Dr. Kai Li

05/08/2025

Application Purpose and Target Users

My blockchain-based Connect4 game addresses a fundamental challenge in online gaming: establishing trust between players without relying on a central authority. Traditional online games depend on centralized servers that are vulnerable to downtime, hacking, or manipulation. By implementing this classic game on the blockchain, I've created a fully transparent, decentralized gaming experience that demonstrates the power of smart contracts for trustless player interactions.

I chose to develop this application because Connect4's simple ruleset makes it an ideal candidate for blockchain implementation while still providing engaging gameplay. The classic nature of the game makes it accessible to a wide audience, while the blockchain implementation demonstrates practical applications of cryptography and distributed systems beyond cryptocurrency.

This project matters because it showcases how blockchain can transform even simple interactions by removing the need for trust between parties. The game serves as an educational tool to help people understand blockchain technology through a familiar context, making complex concepts more approachable.

The primary beneficiaries of this application include:

- Casual gamers looking for a fair, manipulation-free gaming experience
- Students and educators exploring blockchain applications
- Developers seeking examples of smart contract implementation for interactive applications
- Blockchain enthusiasts interested in practical use cases beyond financial transactions

System Functionalities and Workflow

System Inputs and Outputs

Inputs:

- Player wallet connections (MetaMask integration)
- Game creation parameters (timeout period selection)
- Player moves (column selection)
- Timeout claims when opponents are inactive

Outputs:

- Visual representation of the game board
- Game state information (active player, piece counts, timing)
- Win/loss/draw determinations
- Transaction confirmations and blockchain interactions

System Workflow

1. **Game Initialization:**
    - Players connect their MetaMask wallet to the application
    - The system verifies connection to the MegaETH testnet
    - Players create new games by setting a timeout period or join existing games
2. **Matchmaking Process:**
    - Players can create a game and wait for an opponent
    - Players can join available games from the lobby
    - Players can use auto-matchmaking to join an existing game or create a new one
3. **Gameplay Loop:**
    - Players take turns selecting columns to drop their pieces
    - Each move is submitted as a blockchain transaction
    - The smart contract validates moves and updates the game state
    - The frontend visualizes the current board state and player information
4. **Game Resolution:**
    - Win detection: The smart contract checks for four connected pieces (horizontal, vertical, diagonal)
    - Timeout handling: If a player doesn't move within the timeout period, their opponent can claim a win
    - Draw detection: If the board fills with no winner, the game ends in a draw
5. **Post-Game:**
    - Game results are permanently recorded on the blockchain
    - Players can return to the lobby to start or join new games

The workflow emphasizes transparency and fairness, with all game rules enforced by the smart contract rather than a central server. This ensures that neither player can cheat or manipulate the game state.

System Architecture

The system follows a client-server architecture where the "server" is the decentralized blockchain network rather than a traditional centralized server:

Architecture Components

1. **Frontend (Client)**
    - React.js application providing the user interface
    - Ethers.js library for blockchain interaction
    - Component-based design for game board, lobby, and informational pages
    - Local state management for UI responsiveness
2. **Blockchain Layer (Decentralized Backend)**
    - Solidity smart contract deployed on MegaETH testnet

    - Game logic implementation (move validation, win detection, timeout handling)
    - Storage of game states and player information
    - Event emissions for frontend notifications
3. **Web3 Integration Layer**
    - MetaMask wallet connection for user authentication
    - Transaction signing and submission
    - Event listening for game state updates
    - Error handling for blockchain interactions

Component Interactions

The components interact through the following flow:

1. User actions on the frontend trigger function calls to the Web3 integration layer
2. The Web3 layer formats these calls into transactions for the blockchain
3. MetaMask prompts users to sign transactions
4. Transactions are submitted to the blockchain network
5. The smart contract processes transactions, updates game state, and emits events
6. The frontend listens for events and updates the UI accordingly

This architecture ensures that all critical game logic is executed on the blockchain, while the frontend handles visualization and user interaction. The separation of concerns provides a responsive user experience while maintaining the security and transparency benefits of blockchain technology.

Demonstration

A video demonstration of the project is available at
https://www.youtube.com/watch?v=c88FVhDqRFs

Additionally, the live application can be accessed at: https://gilded-crepe-8a7bda.netlify.app/

To interact with the application:

1. Ensure you have MetaMask installed with MegaETH testnet configured (Chain ID: 6342)
2. Visit the application link and connect your wallet
3. Create a game or join an existing one to experience the blockchain-based Connect4 gameplay

Future Plans

I'm excited about where this project could go:

1. **In-Game Chat**: Adding a chat feature so players can communicate during matches. Imagine the trash talk possibilities!
2. **Multi-Testnet Support**: Making the game compatible with various test networks so more people can play without needing specific tokens. Ethereum testnet, Polygon Mumbai, you name it – the more the merrier.
3. **Universal Test Token Compatibility**: Implementing a system that accepts different test tokens across networks, making it truly accessible to anyone learning blockchain.
4. **Tournament Mode**: Because who doesn't love a good competition with brackets and leaderboards?
5. **Achievement System**: Blockchain-recorded achievements that you actually own. "First to win 100 games" would be pretty cool to have permanently recorded.

This project started as a learning exercise but has grown into something I'm genuinely proud of. It shows that blockchain isn't just about money – it's about creating systems we can trust without having to trust each other. And that's pretty revolutionary, even for a simple game like Connect4.