Decentralized Tic-Tac-Toe: A Transparent and Fair Blockchain Game

CS596 - Fundamentals of Cryptography and Blockchain
San Diego State University – Spring 2025
Team Members: Amine Boughou

1. Introduction

Through the introduction of transparent, trustless platforms, decentralized apps (DApps) have completely transformed digital interaction. With the help of a smart contract installed on the Ethereum blockchain, this project illustrates a Decentralized Tic-Tac-Toe game in which two players who are not mutually trusted compete equitably. By guaranteeing unchangeable movements, publicly verifiable results, and tamper-proof game rules, this solution removes the possibility of cheating that comes with centralized game servers.

2. Motivation

Despite being a simple game, Tic Tac Toe on a blockchain illustrates the real benefits of decentralization:

- Transparency: Every action on the blockchain is visible to all participants.Once a move is submitted, it cannot be changed.
- Fairness: The smart contract impartially enforces the game's rules.
- Lack of trust: There is no requirement for a centralized server or trustworthy third party.

3. System Overview

3.1 Supported Application

- Application: An Ethereum-based two-player tic tac toe game.
- Users: Any two participants who have MetaMask-capable browsers and Ethereum accounts.
  Its importance lies in its ability to teach equitable multiplayer interactions without the need for a central server.

3.2 Functionalities

- Player Registration (Player X and Player O).

- Player Moves (grid positions 0-8).

- Game State Validation (win, draw, ongoing).
- Transparent board state updates.

4. System Architecture

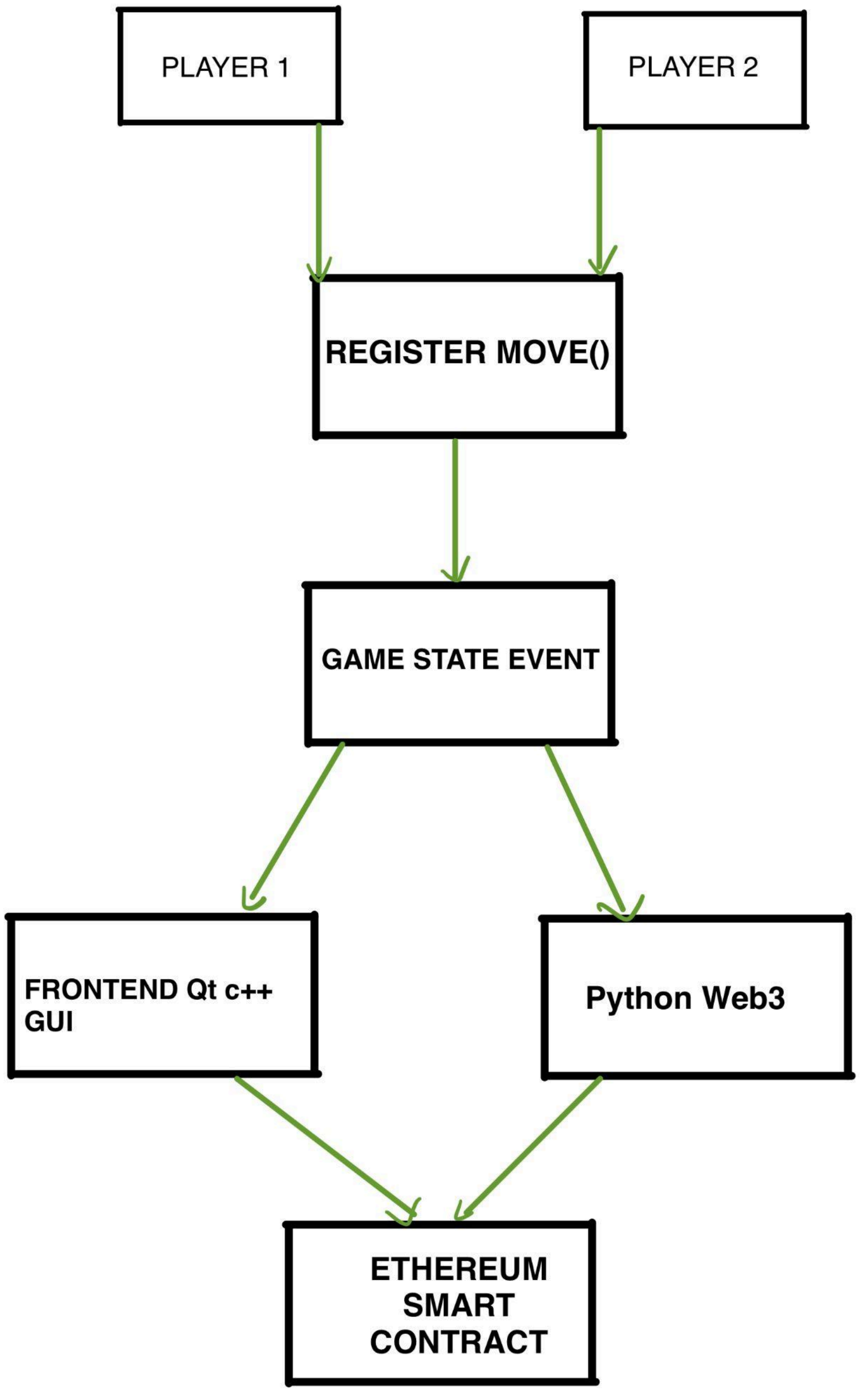
4.1 On-Chain Components

- **Smart Contract (Solidity)**:
    - Player registration.
    - Board state management.
    - Rule enforcement.
    - Result determination.
    - Event emission for off-chain UI updates.

4.2 Off-Chain Components

- **Frontend (Qt C++ GUI with Python Web3 Integration)**:
    - User registration UI.
    - Board visualization.
    - Move submission.
    - Real-time game status display.

Architecture Diagram



5. Implementation Details

5.1 Development Tools

- **Solidity** Smart Contract Development (Remix IDE).
- **Ganache CLI** Local Blockchain Testing.
- **Python Web3.py** blockchain interaction.
- **Qt C++ GUI** user interface.

5.2 Smart Contract Key Functions

- `registerPlayer()`: Register X and O players.
- `makeMove(uint8 position)`: Submit move.
- `getBoard()`: Fetch the board state.
- `checkWinner()`: Evaluate the winner.

5.3 Off-Chain Features

- PyQt GUI with **buttons representing the board**.
- Python script calling **Web3 functions** (`register`, `move`, `fetch state`).
- Qt connects **button presses to blockchain transactions**.

---

6. Demonstration Summary

A recorded demo or live class demonstration covers:

- Player registration.
- Move submission through GUI.

- Blockchain state updates with visible transaction hashes.
- Automatic winner detection by the contract.

7. Challenges Encountered

- Integrating Python Web3.py with the Qt GUI.
- FAcing multiple crash in QT GUI.
- Managing state synchronization between blockchain and GUI.

8. Future Improvements

- **Multiple Game Sessions**: Support multiple ongoing games.
- **Staking Mechanism**: Introduce Ether wagers for competitive play.
- **Decentralized Frontend**: Host the GUI on IPFS for full decentralization.
- **Gas Optimization**: Reduce unnecessary state writes.

9. Conclusion

This project successfully shows how even a game like tic tac toe can use blockchain's a transparant ,fair and trustless execution.It benefits for application creation ,blockchains and client integration . By guaranteeing unchangeable movements, publicly verifiable results, and tamper-proof game rules

10. References

1. CIS629 Fundamentals of Blockchain and Cryptocurrency, Dr. Yuzhe Tang, Syracuse University.
2. Ethereum Developer Documentation: https://ethereum.org/en/developers/docs/
3. Solidity Documentation: https://docs.soliditylang.org
4. Web3.py Documentation: https://web3py.readthedocs.io