An Interactive Web-Based Blockchain Visualization Tool for Educational Purposes

Pranav Patil
Department of Information Technology
Vidyalankar Institute of Technology
Mumbai, India

Arihan Iyer
Department of Information Technology
Vidyalankar Institute of Technology
Mumbai, India

Aniket Mahajan
Department of Information Technology
Vidyalankar Institute of Technology
Mumbai, India

Sarvesh Pansare
Department of Information Technology
Vidyalankar Institute of Technology
Mumbai, India

***Abstract*—Understanding blockchain technology fundamentals, such as hashing, proof-of-work, immutability, and distributed consensus, can be challenging due to their abstract nature. This paper presents an interactive web-based application designed to demystify these concepts for educational purposes. The system combines a Python Flask backend, implementing a simplified blockchain model, with a React frontend utilizing Three.js for dynamic 3D visualization. Users can add transactions to a mempool, mine new blocks incorporating these transactions, observe the proof-of-work process, visualize the growing chain, and interactively explore block data. Crucially, the tool allows users to simulate data tampering within blocks and immediately visualize the consequences through a chain validation mechanism, highlighting the cryptographic linkage and immutability principles. The system also demonstrates difficulty adjustment based on block generation time. This application provides an intuitive, hands-on platform for students and enthusiasts to grasp core blockchain mechanics.**

Keywords— Blockchain, Visualization, Education, React, Three.js, Flask, Proof-of-Work, Hashing, Immutability, Interactive Learning, Difficulty Adjustment.

I. INTRODUCTION

Blockchain technology has emerged as a transformative force across various industries, promising enhanced security, transparency, and decentralization. However, the underlying mechanisms, including cryptographic hashing, distributed ledger concepts, consensus algorithms like Proof-of-Work (PoW), and the principle of immutability, are often complex for newcomers to grasp solely through theoretical explanations. There is a growing need for effective educational tools that can provide intuitive and interactive learning experiences. Traditional teaching methods often rely on static diagrams and textual descriptions, which may fail to convey the dynamic and interconnected nature of blockchain operations. Interactive visualizations can bridge this gap by allowing learners to actively engage with the system, observe cause-and-effect relationships, and build a more robust mental model of how blockchains function.

This paper introduces a web application developed specifically for blockchain education. It aims to provide a simplified yet functional simulation of a blockchain, focusing on visualizing core concepts in an interactive 3D environment. Users can perform fundamental actions like creating transactions, mining blocks, and observing the results in real-time. A key feature is the ability to simulate data tampering and witness the chain validation process break the cryptographic links, reinforcing the concept of immutability. The tool also visualizes the mempool and demonstrates automatic difficulty adjustments.

II. RELATED WORK

Several tools and approaches have been developed to help in understanding blockchain technology, ranging from conceptual simulations to specific data visualization platforms. Ocheja et al. provides a review focused specifically on visualizing educational blockchain data, highlighting trends and challenges, arguing that unique visualization approaches are needed for educational contexts compared to financial or technical analysis tools. Some tools focus on non-computer-based simulations, like "The Blockchain Game" described by IBM and Instructables, where participants manually act as nodes, calculate hashes (simplified), and achieve consensus, emphasizing the procedural aspects without requiring technical infrastructure. Web-based simulators like Bloxxgame and Bitcoin Simulator offer interactive environments where users can create transactions, mine blocks, and observe chain growth, like the tool presented here. Bloxxgame specifically targets educational use, allowing students to experiment with hashing, signing, and consensus. Other visualization tools focus on analyzing real blockchain data, such as Bitnodes (visualizing node distribution), Ethviewer (real-time Ethereum transactions), or commercial platforms like EY Blockchain Analyzer and Bitquery, which offer sophisticated tools for exploring transaction flows and patterns, often for compliance or forensic purposes.

While these analysis tools are powerful, they may be overwhelming for beginners trying to grasp fundamental mechanics. The tool presented in this paper differentiates itself by combining a simplified backend simulation focused on core concepts (PoW, hashing, linkage, difficulty adjustment) with an interactive 3D visualization built using React and Three.js. Its primary educational contribution lies in the explicit feature allowing users to *tamper* with block data and *immediately visualize* the resulting break in chain validity through distinct visual cues in the 3D environment, directly demonstrating the principle of immutability in a way not commonly emphasized in other simulation tools. Furthermore, it provides a complete,

self-contained web application integrating the backend logic, transaction handling, mining simulation, and visualization within a single cohesive educational package.

III. System Architecture

The application employs a client-server architecture, separating the blockchain logic (backend) from the user interface and visualization (frontend). The backend is developed using Python and the Flask micro-framework. It is responsible for managing the blockchain data structure, handling block creation, implementing the Proof-of-Work algorithm, performing chain validation, adjusting mining difficulty, maintaining a transaction mempool, and exposing a RESTful API for frontend interaction. The front end is built with React, a JavaScript library for building user interfaces, utilizing axios for communication with the backend API. A critical component of the frontend is its use of React Three Fiber and drei, which provide abstractions for creating and managing 3D scenes with Three.js directly within the React component model. This approach enables the dynamic 3D visualization of the blockchain. Communication between the frontend and backend occurs via HTTP requests to the Flask API endpoints, transferring data primarily in JSON format. This separation of concerns allows for modularity and facilitates independent development and potential future scaling or replacement of components.

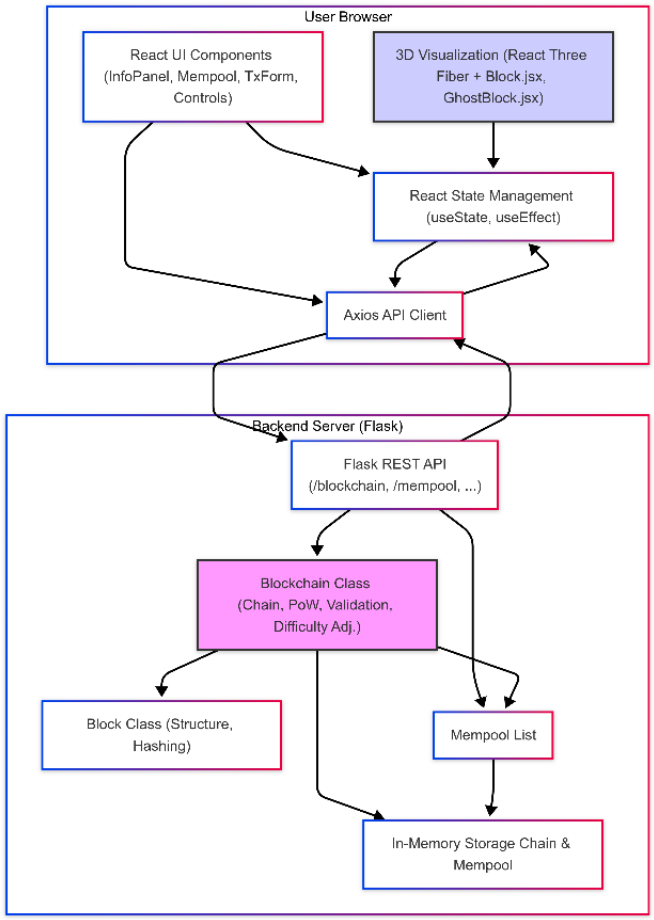

Fig. 1. System Architecture

A. Frontend

The React frontend is responsible for rendering the user interface and the interactive 3D visualization of the blockchain. It is structured using functional components to manage different aspects of the view and user interaction. The main App.jsx component serves as the central hub, managing the global application state, including the blockchain data (chain), the transaction mempool (mempool), details of any currently selected block, loading and mining status indicators, the current network difficulty, and the chain's validity status. It utilizes React hooks like useState, useEffect, and useCallback for state management and handling asynchronous operations like data fetching and API calls. The App component orchestrates user actions such as adding transactions, initiating mining, triggering backend validation, and simulating data tampering. The core visualization is handled by the BlockchainVisualizer.jsx component, which leverages React Three Fiber to declaratively build the 3D scene. It maps the chain state array to individual Block.jsx components, calculates their positions in 3D space, draws connecting lines between them, and renders the GhostBlock.jsx component representing the next potential block.

It also sets up the necessary 3D environment elements like cameras (OrthographicCamera), interactive controls (OrbitControls), and lighting. The Block.jsx component itself represents a single block as a 3D cube, whose visual appearance (color, emissive properties, interactivity) dynamically changes based on a status prop (valid, first-invalid, subsequent-invalid) passed down from the visualizer, which reflects the overall chain validity state. The GhostBlock.jsx component provides the visual cue and interaction point for mining new blocks, changing its appearance based on whether mining is in progress and whether there are transactions in the mempool. Complementing the visualization are UI panels: InfoPanel.jsx displays detailed information about the currently selected block (hash, nonce, timestamp, data, etc.) and crucially allows users to edit the data field for non-genesis blocks to simulate tampering, providing a save button that triggers validation. MempoolPanel.jsx lists pending transactions, and TransactionForm.jsx provides input fields for users to create and submit new transactions to the mempool.

The simulation of data tampering is a central interactive feature facilitated by the frontend architecture. When a user edits the data field of a selected block within the InfoPanel and saves the changes, the frontend first updates its local representation of that block within the chain state. Immediately following this local update, it sends the entire modified chain state as a JSON payload to the backend's /api/validate_chain_state endpoint. The backend performs the rigorous validation checks described earlier. The validation result (success or failure, and the index of the first invalid block) is sent back to the frontend. This response updates the chainValidity state managed by the App component. This updated validity status is then propagated down through props to the BlockchainVisualizer and

individual Block components. This propagation triggers conditional rendering logic within these components, causing the visual appearance of the blocks and connecting lines from the point of tampering onwards to change, clearly indicating the break in the chain's integrity.

B. Backend

The Flask backend manages the blockchain's core logic and data. The Block class defines each block's structure, including fields like index, ISO-formatted timestamp, transaction data (with amounts as strings for consistent hashing), previous hash, difficulty, nonce, and the block's hash. A separate mined_timestamp (Unix epoch float) aids in difficulty adjustments. The Blockchain class maintains the chain as a list, initializing it with a Genesis block. New blocks are added using add_block(data), which incorporates mempool transactions, sets the previous hash, performs Proof-of-Work via proof_of_work(block), and appends the mined block. Hashes are computed deterministically using sorted JSON serialization and SHA-256. The adjust_difficulty method modifies mining difficulty based on the time taken to mine recent blocks, ensuring a stable block generation rate.

The mempool, an in-memory list, temporarily stores user-submitted transactions before they're mined. The Flask app exposes RESTful API endpoints: GET /api/blockchain retrieves the blockchain's current state; GET /api/mempool lists pending transactions; POST /api/add_transaction adds new transactions to the mempool; POST /api/mine_block mines a new block with mempool transactions, updates the chain, and clears the mempool; and POST /api/validate_chain_state validates a potentially modified chain from the frontend by reconstructing Block objects and checking the chain's integrity, returning validation results and the index of any invalid block.

IV. Key Features and Educational Aspects

The design of this application prioritizes features that directly contribute to understanding fundamental blockchain concepts in an accessible manner. The interactive 3D visualization moves beyond static representations, offering a spatial context for the chain's structure, growth, and interconnectedness. Users can freely manipulate the view (rotate, pan, zoom) to inspect blocks and the links between them from various angles. This active exploration is complemented by a step-by-step interaction model where users directly initiate actions like adding transactions to the mempool and triggering the mining process. They can then immediately observe the consequences: the mempool count changes, transactions disappear from the pending list, and a new block visually appears and connects to the chain, making the process tangible. While the underlying cryptographic computations of Proof-of-Work are complex, the application demonstrates the concept by showing a "Mining..." status indicator, introducing a processing delay, and finally revealing the mined block with its associated nonce and difficulty values displayed in the information panel, representing the computational effort involved. Perhaps the most impactful educational feature is the simulation of immutability through data tampering. By allowing users to modify the data within a previously mined block and then triggering a validation check, the system provides immediate, clear visual feedback. The tampered block and all subsequent blocks and connecting lines change color, visually demonstrating how a single alteration breaks the cryptographic linkage established by the hash chain.

This vividly reinforces the concept of immutability and the inherent difficulty of rewriting blockchain history. The validation feedback is specific, highlighting the *first* block where the integrity checks failed and differentiating subsequent affected blocks, clarifying the cascading effect of tampering. Furthermore, the visualization of the mempool as a separate list of pending transactions reinforces the concept of transaction confirmation latency – transactions must wait to be selected and included in a block by a miner. Finally, the dynamic adjustment of mining difficulty, reflected in the UI and potentially observable as changes in the time required to mine subsequent blocks, provides insight into the blockchain's self-regulating mechanism designed to maintain a consistent block generation rate despite fluctuations in network hashing power.

V. Results

The effectiveness of the educational tool relies heavily on clear and intuitive visual cues within the 3D environment to represent the state and properties of the blockchain. Blocks are rendered as 3D cubes using the Block.jsx component. Visual differentiation is applied based on the block's type and validity status. The Genesis block is given a distinct color (e.g., Cyan) to mark it as the origin of the chain. Standard valid blocks share a consistent color (e.g., Green). When chain validation fails due to tampering, the block identified as the *first* invalid block adopts a bright warning color (e.g., Red), drawing immediate attention to the point of integrity failure. Blocks after the first invalid one are also colored differently, perhaps using a dimmer shade of the warning color (e.g., Dark Red), to indicate they are part of the invalidated segment but not the source of the break. Interaction cues are also employed; valid blocks typically exhibit subtle scaling effects on hover and selection, accompanied by an emissive glow to highlight the user's focus. Blocks within an invalidated segment might have these interaction effects reduced or disabled to visually reinforce their compromised status.

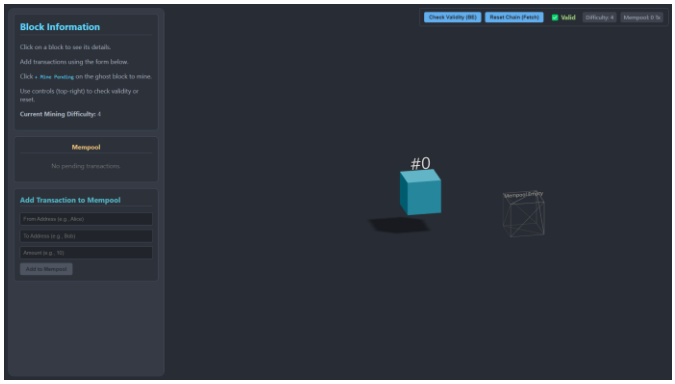

Fig. 2. Interface Genesis Block

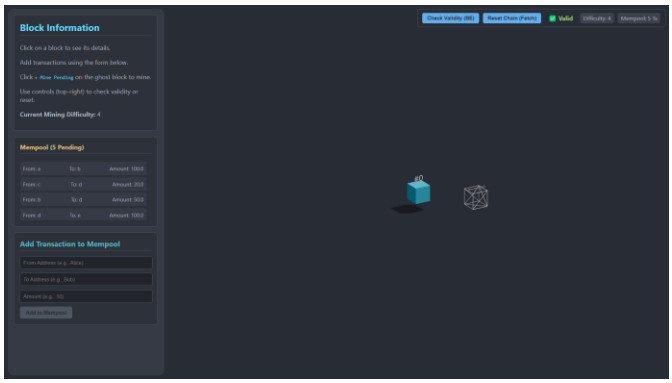

Fig. 3. Mempool Transactions

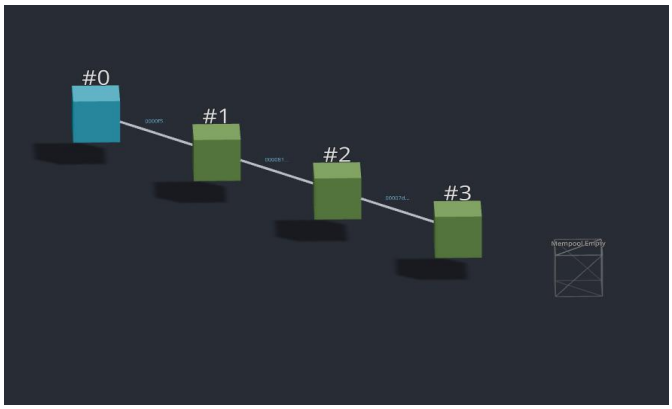

Fig. 4. Blockchain

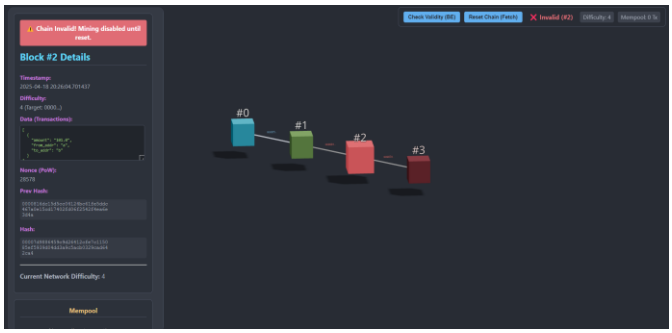

Fig. 5. Invalid/Tampered Blockchain

The connections between consecutive blocks, representing the crucial hash links, are visualized using lines rendered via drei Line. The color of these lines reflects the validity of the link they represent. A link leading to a valid block (or the first invalid block itself) is displayed in a standard neutral color (e.g., Grey/White). However, if a link points to a block that is part of the *subsequent* invalid segment (meaning the integrity broke in a preceding block), the line itself adopts a distinct warning color (e.g., Dark Grey or matching the invalid block color), visually signifying the broken chain continuity *before* that point. To further emphasize the hash linkage, abbreviated hash values are displayed as text labels (drei Text) near the midpoint of each connecting line. These labels represent the previous_hash value stored within the block *following* the line. The color of this hash text mirrors the validity of the link it sits upon – standard color (e.g., Blue) for valid links and a warning color (e.g., Red) for invalid links, reinforcing the connection between the visual link and the underlying cryptographic data.

Finally, the GhostBlock.jsx component serves as a visual placeholder and interaction point for creating the next block in the sequence. It typically appears as a distinct wireframe cube positioned after the last valid block. Its appearance dynamically changes based on the application's state. When the system is ready for mining (mempool has transactions, chain is valid), it might become slightly more prominent on hover. During the mining process, its appearance changes significantly – perhaps becoming solid, pulsing with light or color (e.g., Yellow/Orange), and displaying "MINING..." text – to represent the ongoing computational effort. If mining is not possible (e.g., the mempool is empty or the chain is invalid), the ghost block might appear dimmed, less interactive, or display text like "Mempool Empty", providing clear feedback on why mining cannot be initiated. Interaction with the ghost block (clicking) is enabled only when conditions for mining are met, guiding the user appropriately.

VI. Conclusion And Future scope

This project successfully demonstrates the utility of interactive 3D visualization as an effective educational tool for demystifying core blockchain concepts. By providing a hands-on, visually engaging environment, the application allows users to actively participate in the creation and validation of a simplified blockchain. Users can add transactions, initiate mining, observe the results of Proof-of-Work, and, most importantly, directly witness the consequences of data tampering through immediate and clear visual feedback on the chain's integrity. This experimental approach facilitates a deeper and more intuitive understanding of cryptographic hashing, block linking, the principle of immutability, and the chain validation process compared to purely theoretical explanations. The inclusion of dynamic difficulty adjustment further adds a layer of realism to the simulation, illustrating how blockchain networks maintain operational stability.

While the current implementation provides a solid foundation for educational purposes, several potential enhancements could further increase its value and scope. Future work could involve simulating a networked environment with multiple nodes, visualizing block propagation across the network, and demonstrating consensus mechanisms like the longest chain rule more explicitly. Implementing and visualizing alternative consensus algorithms, such as Proof-of-Stake (PoS), would allow for comparative learning. The visualization could be enhanced to provide more detailed insights into the transactions contained within each block, rather than just displaying the raw data structure. Introducing basic smart contract functionality, allowing users to deploy simple contracts and visualize their state changes on the blockchain, would cover another significant aspect of modern blockchain technology. Finally, for scalability and demonstrating longer-term behavior, performance optimizations for visualizing very large chains

could be explored, potentially using techniques like geometry instancing or implementing level-of-detail rendering for blocks farther away from the camera focus.

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[2] P. Ocheja, B. Flanagan, H. Ogata, and S. S. Oyelere, "Visualization of education blockchain data: trends and challenges," *Interactive Learning Environments*, pp. 1-25, Jan. 2022, doi: 10.1080/10494820.2021.2024431. [Online]. Available: https://doi.org/10.1080/10494820.2021.2024431.

[3] C. N. C. et al., "Exploring the Role of Visualization and Engagement in Computer Science Education," in *ITiCSE-WGR '02: Proceedings of the 7th annual conference on Innovation and technology in computer science education*, Aarhus, Denmark, Jun. 2002, pp. 13–24, doi: 10.1145/544414.544418.

[4] G. Andersen, "The Proof-of-Work Concept," Satoshi Nakamoto Institute, Jun. 24, 2013. [Online]. Available: https://satoshi.nakamotoinstitute.org/writings/proof-of-work-concept/.

[5] J. Scott, "The Blockchain Game: A great new tool for your classroom," IBM Blockchain Blog, Apr. 29, 2019. [Online]. Available: https://www.ibm.com/blogs/blockchain/2019/04/the-blockchain-game-a-great-new-tool-for-your-classroom/.

[6] MoldStud, "The Role of Visualization in Computer Science Education," Jan. 27, 2024. [Online]. Available: https://moldstud.com/articles/the-role-of-visualization-in-computer-science-education

[7] Bitcoin Simulator, "Blockchain | Bitcoin Simulator," Accessed: Apr. 18, 2025. [Online]. Available: https://bitcoinsimulator.net/blockchain