Federal University of Technology - Paraná - Brazil - UTFPR
Academic Department of Electronics – DAELN
Academic Department of Informatics – DAINF
Computer Engineering Course
Integration Workshop 3 (ELEX22) – S71 – 2024/2

Technical Report
Blockchurna: A Blockchain-Based Voting System

Arthur Fenili da Silva – arthurfenili@alunos.utfpr.edu.br
Diogo da Silva Gouveia – diogogouveia@alunos.utfpr.edu.br
Lucas Henrique Pereira Flores – lucasflores@alunos.utfpr.edu.br
Matheus Augusto Burda – matheusburda@alunos.utfpr.edu.br
Victor Hugo Garrett – victorgarrett@alunos.utfpr.edu.br

February 2025

Abstract

Blockchurna is a decentralized voting system that leverages blockchain technology to eliminate reliance on central authorities for vote counting and provide voters with a way to verify their votes. The system provides the election with more security, anonymity, and transparency through a public blockchain that records anonymized votes. By decentralizing vote validation and storage, Blockchurna reduces risks of manipulation and enhances voter trust in electoral processes. This report describes in detail the core components of Blockchurna, including software, hardware, and mechanical design. The software consists of blockchain-based vote recording, secure cryptographic verification, and a voter authentication system. The hardware includes a voting machine equipped with a biometric scanner, a thermal printer that prints the verification receipt, which is not enough to verify the vote by itself, and a secure USB storage mechanism. Additionally, this report discusses the system's integration, covering the entire voting process from authentication to blockchain insertion and verification. By addressing these aspects, Blockchurna aims to enhance the credibility and efficiency of modern elections while maintaining the standards of security and user accessibility.

1 Introduction

Electronic voting has become a widely used method for conducting elections, offering efficiency and accessibility. However, many current systems rely on centralized authorities to manage the vote-counting process, creating potential vulnerabilities. Voters must place their trust in election officials as they lack direct means to verify whether their votes were counted correctly. Additionally, concerns about data security, potential vote manipulation, and system failures raise doubts about the transparency and integrity of electronic voting.

Many electronic voting systems also lack verifiability. Although some offer printed receipts, these are often not used in the final vote count. Furthermore, in large-scale elections, a single point of failure in the centralized system could lead to disastrous consequences, such as election fraud or system malfunctions. Verifiability is also a problem, as many possible implementations would also allow vote buying and coersion.

1.1 Proposed Solution

To address these issues, Blockchurna introduces a blockchain-based electronic voting system that decentralizes the vote-counting process while maintaining voter anonymity and data integrity. The operation of the Blockchurna system is illustrated in Figure 1 and follows these steps:

1. Voters cast their votes using a secure voting machine with biometric authentication to prevent fraud, as already done in the current elections in Brazil.

2. A receipt is printed for each voter with a random cryptographic key, allowing voters to later verify their vote **in an authorized section**.

3. Votes are recorded in an immutable blockchain ledger, ensuring security and transparency, in which a decentralized network of nodes verifies each vote before adding it to the blockchain.

4. A Block Explorer application retrieves and displays anonymous voting data, allowing public counting of results.

5. Using their voter identification and the cryptographic key generated at the time of casting their vote, along with the Government key —the authorities' encryption key— voters can verify their votes on the blockchain in an authorized section.

Using blockchain technology, Blockchurna election results are transparent, auditable, and resistant to external interference. The integration of this system with secure voting machines improves usability while maintaining compliance with electoral regulations. Ultimately, the goal is to create a voting system that increases public trust and confidence in democratic processes.

2 Project Specification

2.1 Functional Requirements

The system is divided into three main categories:

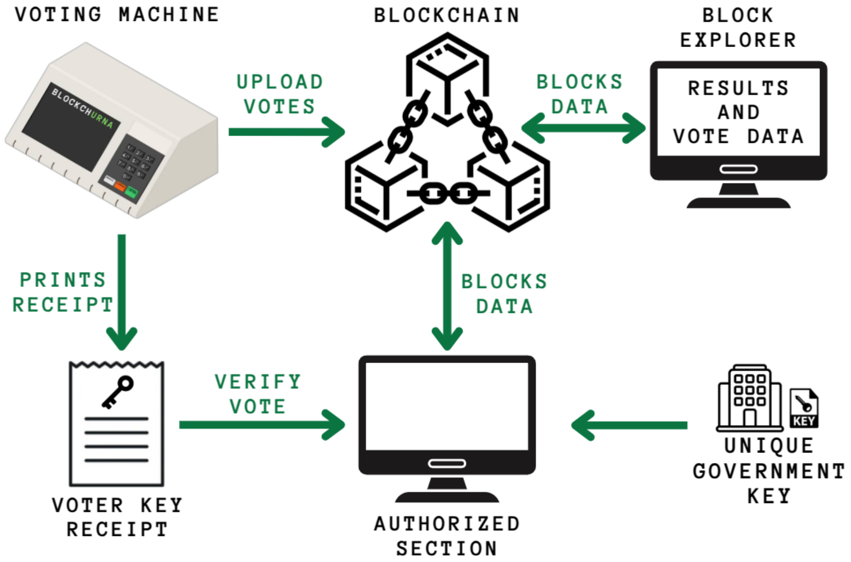


Figure 1: Overview diagram of the project's operation.

2.1.1 Mechanical Design

The mechanical design, inspired in the Brazilian UE2015 voting machine [1], ensures physical security, ease of use, and durability. The structure must accommodate all the electronic components. The design includes features such as Braille support and audio feedback to ensure accessibility for visually impaired users.

2.1.2 Hardware

The hardware design of the voting system is built around a Raspberry Pi 3 [2] embedded computer, which serves as the main processing unit, controlling and interfacing with all peripheral components. The system integrates the following key hardware elements:

- **Raspberry Pi 3:** Serves as the core computer responsible for processing voting data and controlling connected devices.
- **LCD Screen:** Displays voting-related information such as candidate details, photos, and voting instructions.
- **Thermal Printer:** Prints a receipt with a unique Key that voters can later use for vote verification in government authorized sessions.
- **Pendrive:** Securely stores all votes and candidate data in anonymized form.
- **Biometric Sensor:** Provides voter authentication by verifying fingerprints.

- **Keyboard:** Functions as the primary input device, allowing voters to type in their votes.
- **Power System:** Ensures stable operation regulated down to 12V and 5V with L7812 and L7805 voltage regulators.

A schematic of these components is shown in Appendix C C.

2.1.3 Software

- Secure blockchain for vote recording and verification.
- Firmware for managing biometric authentication and vote processing.
- A Web interface for decentralized vote counting and authorized vote verification.

2.2 Functional Requirements

ID	Requirement Description
FR101	The voting system must authenticate voters using biometric identification.
FR102	The system must generate a vote hash using cryptographic keys.
FR103	The blockchain must reject blocks with invalid vote signatures.
FR104	Voters should be able to verify their vote using a secure interface.
FR105	Blocks will only be added after validation by 50% of network nodes.

Table 1: Main Functional Requirements

3 Development

3.1 Mechanical Development

The project development started with the CAD design of the structure, keys and key stand. It was developed using the software SolidWorks [3] and the 3D model of the Blockchurna is shown in Figure 2. The technical drawing of the structure, including all necessary cutouts and holes, is presented in Figure 9 in Appendix A.

3.1.1 Mechanical Structure

The structure is made of polystyrene (PS) and includes all the following components:

- Front Panel:

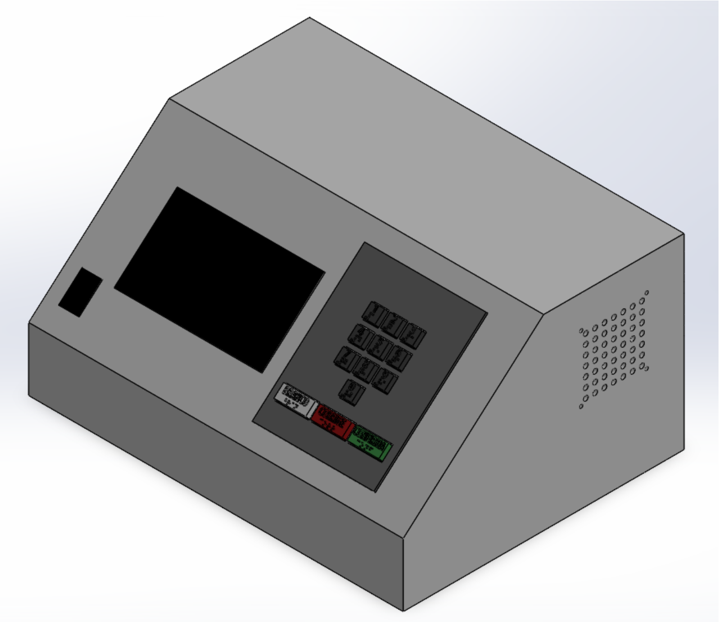
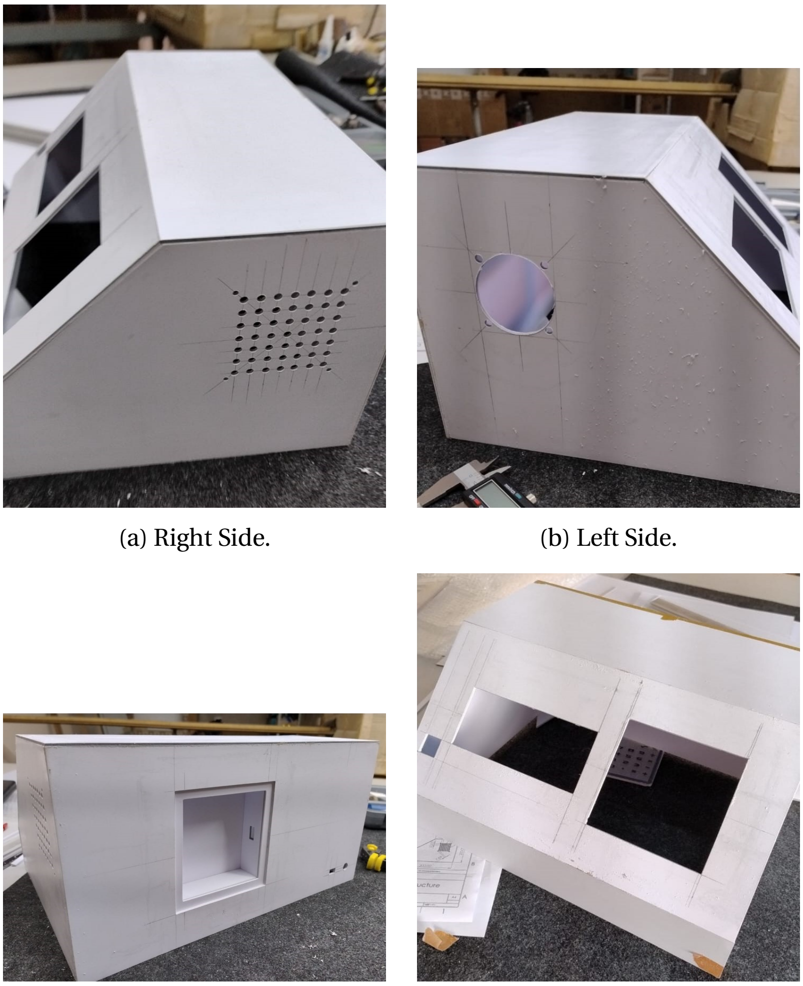


Figure 2: Blockchurna 3D Model.

  - A 7-inch LCD screen for vote selection.
  - A fingerprint scanner for biometric authentication.
  - A keyboard with Braille support for accessibility.

- Right Side:
  - Ventilation holes to accommodate an 80 mm cooler for thermal management.
- Back Panel:
  - A power supply connector.
  - A USB port for connecting the thermal printer.
  - A secure chamber for storing a USB drive containing vote data.
- Left Side:
  - A speaker for audio feedback.
  - A P2 connector for headphones to assist visually impaired users.

The assembled structure is shown in in Figure 3, with detailed views of each side.



(a) Right Side.

(b) Left Side.

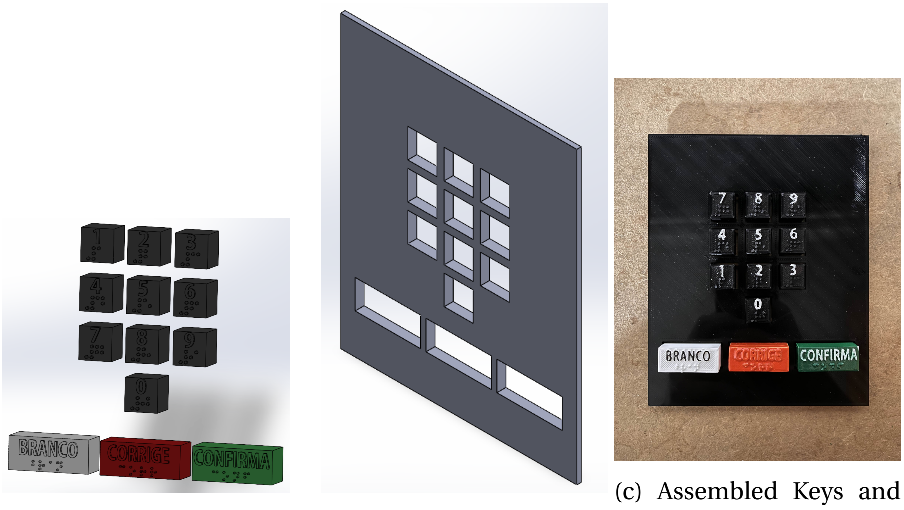
(c) Back Side.

(d) Front Side.

Figure 3: Assembled Structure.

3.1.2 Keyboard Design

The keyboard is designed with Braille support for accessibility. The keys are 3D-printed using PLA filament with the numeric keys (0-9) printed in black and the functional keys ("BRANCO", "CORRIGE" and "CONFIRMA") printed in white. The "CORRIGE" key is painted orange, and the "CONFIRMA" key is painted green for easy identification. The keys are mounted on a keyboard stand, the 3D models are shown in Figure 4b. The keys and the keystand were manufactured using 3D printing technology with PLA filament. The assembled parts can be seen in Figure 4c and the technical drawing is shown in Figure 10 in Appendix B.



(a) Keys. (b) Keyboard Stand. (c) Assembled Keys and Keyboard Stand.

Figure 4: 3D model of the keys and keystand, along with the final printed result of the keyboard.

3.1.3 Pendrive Chamber

As seen on the Figure 3c, the structure has a chamber on the back to safely store the pendrive that will contain the votes made on that voting machine. The chamber will only be open by the poll worker when the section ends.

3.2 Electronic

The electronic system has a 7-inch LCD screen connected via the DSI interface to display voting information, such as candidate details and voting instructions. A thermal printer, connected via USB, that provides voters a printed receipt containing a unique Key for vote confirmation. Data storage is managed through a pendrive, which stores all votes securely. For voter authentication, a biometric sensor is connected to the Raspberry Pi's GPIO pins. A USB keyboard serves as the primary input device for voters to cast their votes. The power system ensures stable operation, utilizing a 12V input regulated down to 5V via L7805 voltage regulator to power the Raspberry Pi and other components. Figure 11 in Appendix C, presents the full electronic schematic, illustrating all connections and power distribution.

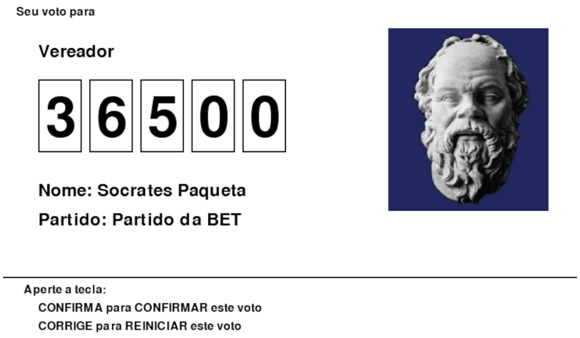
3.3 Software

The Blockchurna voting system is structured into three main software layers: firmware, responsible for managing voter interactions and vote encryption; a blockchain-based backend, which securely records and verifies votes in a decentralized manner; and a web interface, designed for auditing and verification of election results.

3.3.1 Firmware

The firmware of the Blockchurna voting system is responsible for managing the user interface, biometric authentication, cryptographic key generation, vote encryption, and local storage before blockchain submission. It is developed in Python using PyGame [4] for rendering the interface and handling user interactions.

The system consists of several modules that interact with different hardware components. The User Interface Module provides a graphical interface for the voter, like the voting screen in figure 5. The Biometric Authentication Module interfaces with the AS608 fingerprint sensor for voter identification. The Cryptographic Module is responsible for generating and managing keys used to encrypt the vote. The Storage Module handles temporary local storage of votes before they are transferred to the blockchain. The Printing Module sends verification receipts to a thermal printer, and the Vote Submission Module prepares the vote data for secure blockchain upload.



Seu voto para

Vereador

3 6 5 0 0

Nome: Socrates Paqueta

Partido: Partido da BET

Aperte a tecla:

CONFIRMA para CONFIRMAR este voto

CORRIGE para REINICIAR este voto

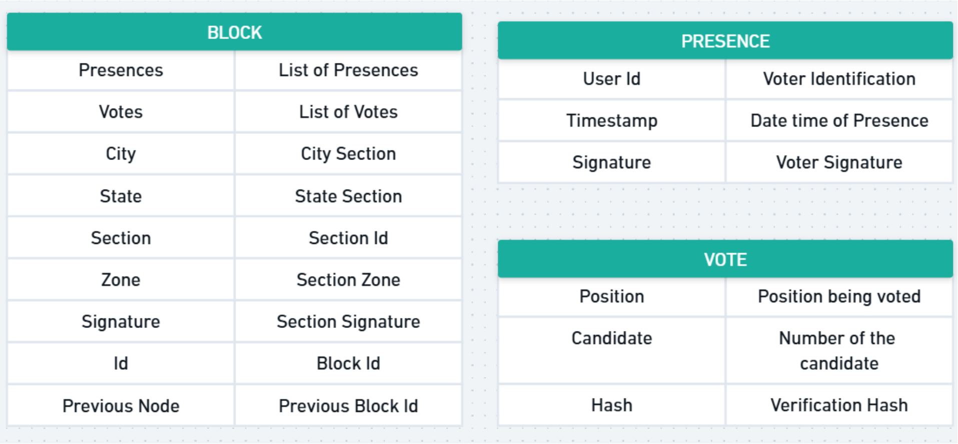
Figure 5: Voting Screen.

3.3.2 Blockchain Implementation

The blockchain component of Blockchurna ensures a decentralized and immutable vote recording system. Implemented in Golang [5], it uses libp2p [6] for peer-to-peer communication. The blockchain is structured into several key classes responsible for its core functionalities:

Block: Contains the list of transactions (votes), a list of presences, a signature, and data about the section. It is a data structure similar to the one in Figure 6.

Blockchain: Manages the chain of blocks, including adding new blocks and validating the chain's integrity.



BLOCK	
Presences	List of Presences
Votes	List of Votes
City	City Section
State	State Section
Section	Section Id
Zone	Section Zone
Signature	Section Signature
Id	Block Id
Previous Node	Previous Block Id

PRESENCE	
User Id	Voter Identification
Timestamp	Date time of Presence
Signature	Voter Signature

VOTE	
Position	Position being voted
Candidate	Number of the candidate
Hash	Verification Hash

Figure 6: Block Structure.

Vote and Presence: It is inside each block, the vote includes voter identification (anonymized) and candidate selection. The presence has a user identification, the timestamp, and the signature. It's important to note that the timestamp is associated with the voter presence signature, but the votes' order is randomized, so there's no way of using that information to determine who voted for whom.

Node: Represents a participant in the network, capable of adding transactions and maintaining a copy of the blockchain.

3.4 Peer-to-Peer (P2P) Network

The blockchain operates over a P2P network [6], enabling decentralized communication between nodes. Node discovery is achieved through a Distributed Hash Table (DHT), which ensures efficient lookup and routing of peers in the network. Additionally, the system defines distinct message types to handle essential operations, including block propagation, transaction broadcasting, and consensus mechanisms. A new node follows the diagram in Figure 7 to enter the network.

3.5 Block Validation Process

Block validation exists to ensure blockchain security by preventing fraudulent, malformed, or tampered blocks from being added. The process starts with an integrity check, confirming the block's structure and required fields. Invalid blocks are rejected immediately. Next, presence signatures are verified to ensure authorized participants have signed the block. If invalid, the block is discarded.

Ballot signatures are then checked to authenticate individual transactions. If any fail, the block is rejected. Hash verification follows, confirming the block's cryptographic integrity.

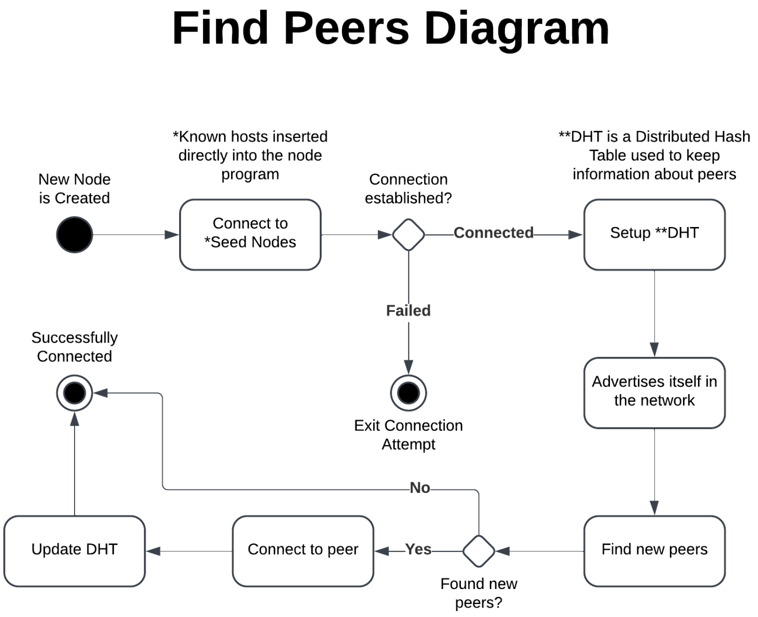


Figure 7: Find Peers Diagram.

Finally, the system detects any modifications. If the block is unchanged, it is added to the blockchain and broadcast to the network. This process is described in the diagram in 8.

3.6 Block Explorer

The block explorer is a web application built with React [7] and TypeScript [8], styled using Tailwind CSS [9]. It is deployed in the cloud via Vercel [10] to manage hosting and scalability. The application provides users with an interface to view blockchain data and analyze voting results.

Viewing Blockchain Data
The block explorer allows users to view detailed information for each block in the blockchain, such as the hash, timestamp, and transactions. This enables users to examine the structure and contents of individual blocks.

Aggregating Voting Data
Voting results are aggregated based on parameters such as city, voting session, and candidate. This data is presented in a structured format, showing aggregated vote counts for each candidate along with their registered number, name, and picture.

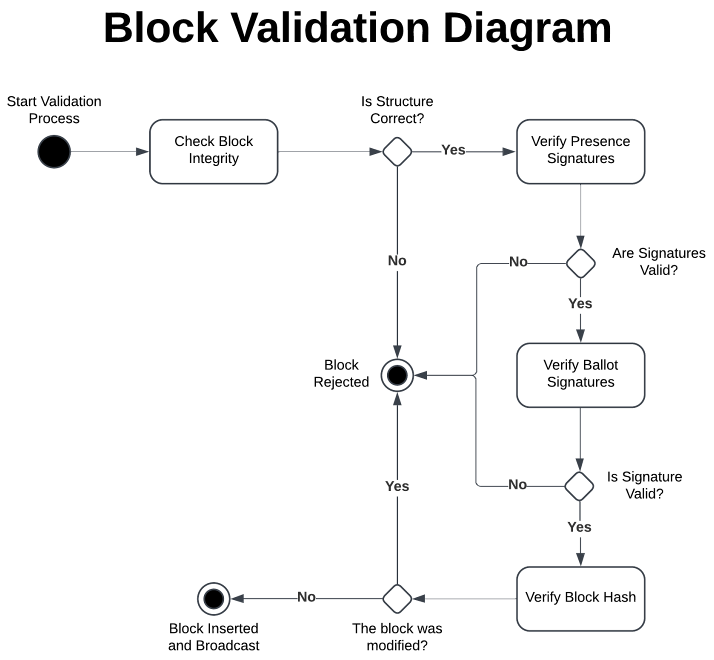


Figure 8: Block Validation Diagram.

Anonymizing Voting Data
The application ensures that voting data remains anonymized. While aggregated voting results are displayed, no personally identifiable information about individual voters is exposed in the system.

Real-Time Updates
The block explorer is designed to update automatically when new blocks are added to the blockchain, ensuring that the displayed data reflects the latest information without requiring manual page refreshes.

3.7 Vote Verification

Voters can verify their vote by proceeding to an authorized section with the printed receipt. After biometric authentication, three pieces of data are required to decrypt and so verify a vote:

- **User ID:** voter identification, present on the printed receipt.
- **Voter Key:** a cryptographic key, randomly generated at the time voter casts their vote, also present on the printed receipt.
- **Government Key:** a cryptographic key, randomly generated at the time

voter casts their vote, but kept exclusively by the authorities.

Without these three pieces of information combined, the data cannot be decrypted or verified, preserving the security and integrity of the election process.

4 Results

This section is dedicated to presenting the results obtained from the project. It begins with an analysis of the budget, comparing the predicted and actual costs. Following this, the timeline of completed tasks is reviewed. Finally, the functional and non-functional requirements that were met are presented.

4.1 Budget

The budget initially stipulated for the project was not fully utilized, as the team already possessed most of the required components needed for the system's development. In addition, several key components were generously borrowed from colleagues, further reducing the need for additional funding. This allowed the project to proceed without exceeding the originally planned budget and made efficient use of existing resources.

Table 2: Cost breakdown of items

Item Description	**Predicted Cost (BRL)**	**Actual Cost (BRL)**
LCD Screen	270.00	0.00 (borrowed)
Thermal Printer	108.00	108.00
Flash Memory	25.00	0.00 (borrowed)
Keyboard	12.00	12.00
Battery	135.00	0.00 (borrowed)
Raspberry Pi 3	200.00	0.00 (borrowed)
Hall Effect Sensor	15.00	20.00
Structure	100.00	100.00
Biometric Sensor	102.00	130.00
3D Printing	50.00	50.00
12V Power Source	50.00	40.00
P2 Extensor	15.00	15.00
Speakers + Amplifier	40.00	45.00
Risk Response Plan	100.00	0.00
TOTAL	**1222.00**	**520.00**

4.2 Project Timeline

The time spent developing the project averaged 69 hours per week for the entire team, which translates to approximately 13.8 hours per person per week. The team gathered primarily on weekends, building strong synergy and collaboration throughout the development process. For the most part, the team stayed on track with the planned schedule and successfully completed tasks on time, maintaining a steady and efficient work pace throughout the project. This is a factor that positively contributed to the success of the project.

4.3 Functional Requirement Completion

Fifty-one functional requirements were initially set to ensure the system's full functionality. Some of these requirements were optional, marked with a "*" next to their ID in the tables below. Unfortunately, three of the optional requirements could not be completed due to various reasons, which will be discussed afterwards.

Table 3: Completed Functional Requirements for Blockchain

ID	Requirement Description
FR1201	The distributed system must receive requests to insert blocks on the blockchain.
FR1202	The distributed system must maintain all of the public keys for the ballots and voter signature verification.
FR1203	The distributed system must validate the signatures in the blocks using the respective public keys and reject the block if one of the present signatures is not compatible with the expected public keys.
FR1204	The distributed system must maintain a copy of the blockchain on each network node.
FR1205	The voter should be able to consult the voting records on the blockchain without identifying the voters.
FR1206	The blockchain system will not allow block changes or deletions.
FR1207	The distributed network must have seed nodes to start the network.
FR1208	The seed nodes must have all the ballots and the voters' prerecorded public keys.
FR1209	The distributed system must keep the blockchain available if at least one node is still online.
FR1210 *	The prototype must handle a distributed network of at least 7 nodes.
FR1211	The distributed system must synchronize any new nodes attempting to enter the network.
FR1212	The distributed system must allow nodes to find other non-seed nodes.
FR1213	The blocks must contain information about the electoral session of origin.

Table 4: Functional Requirements for Block Explorer

ID	Requirement Description
FR1301	The block explorer system must allow users to search blocks by hash through the blockchain.
FR1302 *	The block explorer system must allow users to see the votes and signatures of each block.

ID	Requirement Description
FR1303 *	The block explorer system must show the sum of vote results for each candidate.
FR1304 *	The block explorer system must allow users to group vote data for each voting session.
FR1305 *	The block explorer system must show on the main page the latest blocks inserted.

Table 5: Functional Requirements for Hardware Aspects

ID	Requirement Description
FR201	The hardware system must print the Voter Key when the vote is confirmed.
FR202	The hardware system must provide a biometric sensor to identify voters.
FR203 *	**Not completed -** The hardware system must detect separation of the magnet and the hall effect sensor.
FR204 *	The hardware system must provide a speaker to play vote confirmation audio.
FR205 *	The hardware system must provide an earbud interface to play voting instructions audio for blind people.
FR206 *	The hardware system must not be reachable from the outside via wireless connections.
FR207	The hardware system must store the votes on the flash memory.
FR208 *	The prototype must record at least 10 fingerprints.
FR209 *	The prototype must identify at least 5 different voters.
FR210 *	The prototype must hold at least 10 prerecorded candidates.
FR211 *	**Not completed -** The prototype must run 5 votings of at most 5 minutes without an external power supply by a battery.
FR212 *	The hardware system must include a cooler for the main computer.

Table 6: Functional Requirements for Mechanical Aspects

ID	Requirement Description
FR301 *	**Not completed -** The voting machine must have a seal that can be broken for the purpose of removing the USB drive containing voting data.
FR302 *	All of the keys on the keypad must have braille for accessibility.
FR303	The structure must accommodate an LCD screen, keyboard, and biometric sensor exposed to the outside.
FR304	The structure must accommodate a Raspberry Pi inside it.
FR305	The structure must provide a port for removing the result media.
FR306	The structure must provide an air vent.

Requirements **FR203**, **FR211**, and **FR301** were not completed. **FR203** was not fulfilled because the separation of the magnet could not be accurately detected, and it could be bypassed by placing a stronger magnet on the outside. **FR211** was not met due to complications with the power circuit after integrating the battery. Once the issue was resolved, no further changes were made to avoid potentially introducing new complications. **FR301** was not achieved as a suitable seal that couldn't be bypassed could not be found at the budget estipulated. As a temporary solution, tape was used to secure the USB drive chamber lid in place.

5 Conclusions

In conclusion, this project developed a blockchain-based voting machine inspired by Brazil's electronic ballot system, integrating biometric authentication and blockchain technology for secure, transparent, and efficient vote counting and verification (without compromising anonymity). The development focused on three key areas: a) hardware integration, ensuring cohesive functionality of the biometric sensor, LCD screen, keyboard, and speakers; b) blockchain implementation, enabling secure data upload, validation, and public verification; and c) user experience, providing vote verification receipts and accessibility features.

The project required approximately 420 hours, exceeding the initial estimate of 350 hours. Most requirements were successfully implemented, with only five optional features (22%) remaining uncompleted.

For future improvements, a more advanced biometric sensor, security measures that a real ballot has that were beyond of the scope of this prototype and secure, government-authorized vote verification sessions would be necessary for a production-ready system, ensuring controlled and secure vote verification, so vote buying isn't possible.

In general, this project represents a significant step toward modernizing voting systems by combining established technologies with new ones to improve efficiency and transparency.

References

[1] Justiça Eleitoral. `https://www.justicaeleitoral.jus.br/urna-eletronica/`.

[2] Raspberry Pi. `https://www.raspberrypi.com/documentation/`.

[3] Solid Works. `https://www.solidworks.com/product/solidworks-3d-cad`.

[4] Pygame. `https://www.pygame.org/docs/`.

[5] Google Golang. `https://go.dev/doc/`.

[6] Libp2p. `https://libp2p.io/`.

[7] Meta Platforms. `https://react.dev/reference/react`.

[8] Microsoft. `https://www.typescriptlang.org/docs/`.

[9] Tailwind Labs. `https://tailwindcss.com/docs/`.

[10] Vercel Inc. `https://vercel.com/docs`.

A Technical Drawing of the Structure Modeling

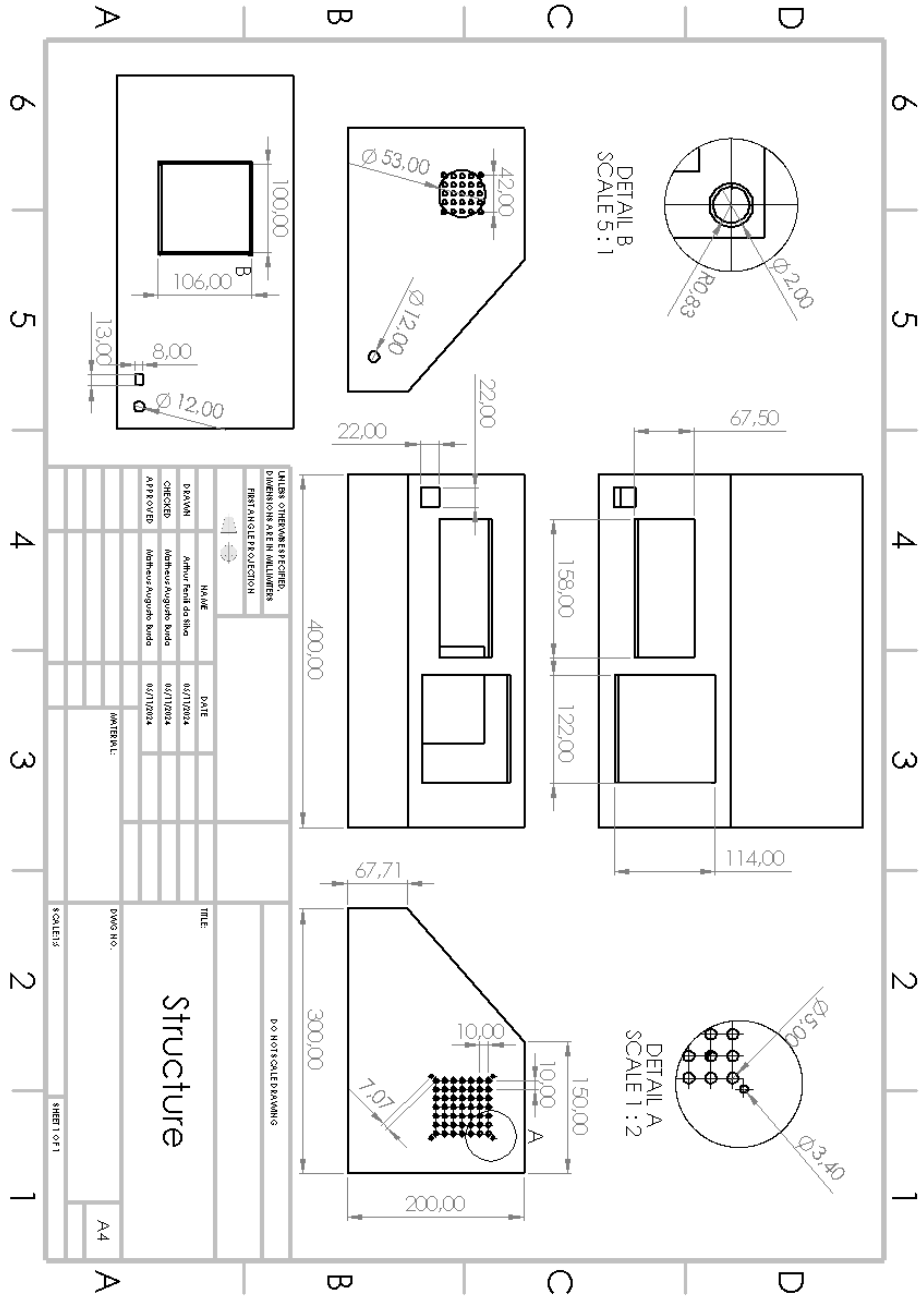


Figure 9: Structure Technical Drawing.

B Technical Drawing of the Keys and Keyboard Modeling

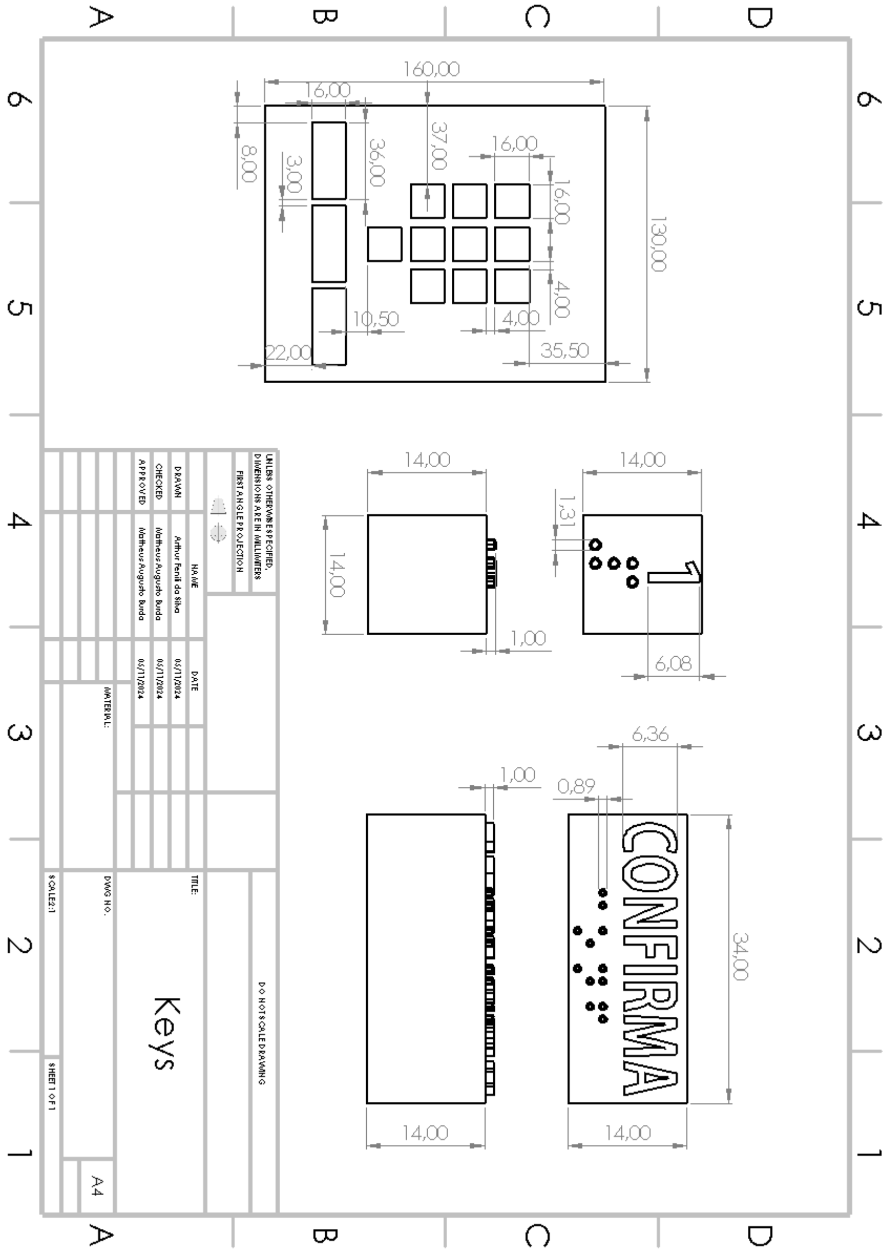


Figure 10: Keyboard Technical Drawing.

C Full Electronic Schematic

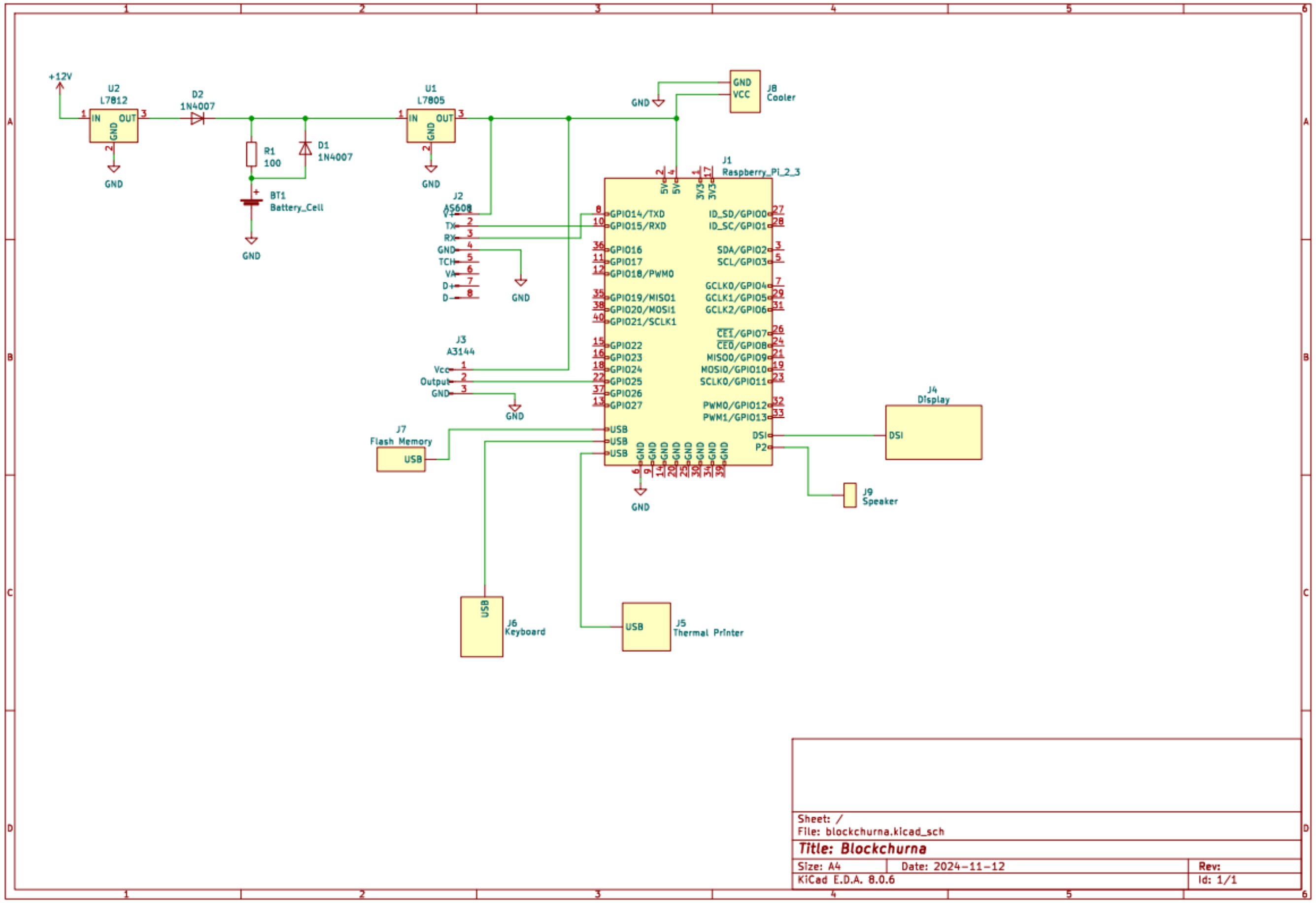


Figure 11: Electronic Schematic.

D Final Video Link

https://youtu.be/0Bhs-ADDU0M

E Blog Link

https://www.notion.so/BLOCKCHURNA-S-BLOG-116acce0dcf98052abc6dfef060d4064