Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

Sistema de Votação Eletrônica Auditável Utilizando Blockchain

Vinicius de Castro Sampaio

JUIZ DE FORA
JUNHO, 2024

Sistema de Votação Eletrônica Auditável Utilizando Blockchain

Vinicius de Castro Sampaio

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Bacharelado em Ciência da Computação

Orientador: Edelberto Franco Silva

JUIZ DE FORA
JUNHO, 2024

SISTEMA DE VOTAÇÃO ELETRÔNICA AUDITÁVEL UTILIZANDO BLOCKCHAIN

Vinicius de Castro Sampaio

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Edelberto Franco Silva
Doutor em Ciência da Computação

Alex Borges Vieira
Doutor em Ciência da Computação

Luciano Jerez Chaves
Doutor em Ciência da Computação

JUIZ DE FORA
11 DE JUNHO, 2024

À minha família, por sua capacidade de acreditar em mim e investir em mim. Mãe, seu cuidado e dedicação foi que deram, em alguns momentos, a esperança para seguir. Pai, sua presença significou segurança e certeza de que não estou sozinho nessa caminhada. Irmã, pelo apoio e carinho e puxões de orelha durante minha caminhada. Aos amigos e colegas, pelos momentos descontraídos.

Resumo

Este trabalho aborda a importância e evolução dos sistemas de votação, destacando a necessidade de mais transparência e confiabilidade. Para isso, abordamos a relevância das votações eletrônicas, tanto em contextos oficiais e em eventos como consultas públicas. Abordamos também o problema da transparência e integridade dos votos, evidenciando falta de mecanismos para que os votantes auditem o seu própio voto. A tecnologia *blockchain* é apresentada como motivação para trazer mais transparência às votações online. Para isso, esse trabalho vem propor a criação de um sistema de votação online que utiliza a tecnologia *blockchain*. Com o uso dessa tecnologia, permitimos que o votante audite o seu própio voto e também entenda o funcionamento da votação através do contrato inteligente que possui código fonte aberto. Para a construção desse sistema, será realizado a criação de um contrato inteligente, utilizando a linguagem *Solidity* e a criação de um sistema para realizar o voto e acompanhar os resultados da votação.

Palavras-chave: Votação, *blockchain*, código fonte, contrato inteligente.

Abstract

This work addresses the importance and evolution of voting systems, highlighting the need for more transparency and reliability. To this end, we address the relevance of electronic voting, both in official contexts and in events such as public consultations. We also address the problem of transparency and integrity of votes, highlighting a lack of mechanisms for voters to audit their own vote. *blockchain* technology is presented as motivation to bring more transparency to online voting. To this end, this work proposes the creation of an online voting system that uses *blockchain* technology. Using this technology, we allow voters to audit their own vote and also understand how voting works through a smart contract that has open source code. To build this system, a smart contract will be created, using the *Solidity* language and a system will be created to carry out the vote and monitor the voting results.

Keywords: Voting, *blockchain*, source code, smart contract.

Agradecimentos

A toda minha família, pelo encorajamento, apoio nos momentos mais difíceis e presença em minha caminhada.

Aos meus amigos e colegas de trabalho, que me ajudaram a evoluir como profissional da área de tecnologia e tornaram esse período mais divertido.

Ao professor Edelberto Franco pela orientação, amizade e principalmente, pela paciência, sem a qual este trabalho não se realizaria.

Aos professores do Departamento de Ciência da Computação pelos seus ensinamentos e aos funcionários do curso, que durante esses anos, contribuíram de algum modo para o nosso enriquecimento pessoal e profissional.

"Não importa quanto a vida possa ser ruim, sempre existe algo que você pode fazer, e triunfar. Enquanto há vida, há esperança".

Stephen Hawking

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

DCC	Departamento de Ciência da Computução
UFJF	Universidade Federal de Juiz de Fora
ICE	Instituto de Ciências Exatas
TSE	Tribunal Superior Eleitoral
IBGE	Instituto Brasileiro de Geografia e Estatística
PoW	*Proof of Work* (Prova de trabalho)
PoS	*Proof of Stake* (Prova de participação)
PoA	*Proof of Authority* (Prova de autoridade)
BTC	*Bitcoin*
ETH	*Ether*
ABI	*Application Binary Interface* (Interface binária do aplicativo)
NFT	*Non-Fungible Token* (Token não-fungível)
DApp	*Decentralized application* (Aplicativo descentralizado)
IDE	*Integrated development environment* (Ambiente de desenvolvimento integrado)
CRUD	*Create, Read, Update and Delete* (Criação, leitura, atualização e remoção)
DER	Diagrama Entidade-Relacionamento
API	*Application Programming Interface* (Interface de programação de aplicativos)
ABI	*Application Binary Interface* (Interface Binária de Aplicação)
UTFPR	Universidade Tecnológica Federal do Paraná

1 Introdução

Atualmente, existem diferentes formas para realizar uma votação, sendo a mais comum o voto em cédulas físicas, onde o votante vai até a urna e deposita o seu voto que foi preenchido em um papel, e depois é realizada a contagem desses votos. Esse meio de votação inclusive é utilizado em diversos países em realização de eleições oficiais, onde são eleitos pessoas para cargos públicos de liderença. Além da votação em cédulas físicas, existem sistemas de votação que mesclam o físico e digital, que é o caso das eleições no Brasil, onde o eleitor vai fisicamente e realiza o seu voto em uma urna digital, e esse voto é armazenado em um dispositivo de memória e depois é realizado a contagem dos votos.

Além dos métodos de votação mais tradicionais, existe a opção da votação eletrônica, também conhecida como *e-voting*: amplamente utilizada em votações online, como votações corporativas e *reality shows* como *Big Brother Brasil* e A Fazenda. A votação eletrônica proporciona benefícios como facilidade na realização do voto e precisão na contagem dos votos, tornando-o mais eficiente e acessível para um grande número de participantes. Esse método também reduz significativamente a possibilidade de erros humanos na contagem dos votos, garantindo assim uma maior confiabilidade.

Hoje na internet há uma variedade de ferramentas disponíveis para a realização de votações e enquetes. No entanto, neste trabalho, iremos explorar uma solução inovadora que tem conquistado cada vez mais destaque: o uso da tecnologia blockchain em sistemas de votação eletrônica em geral. Essa tecnologia ajuda a tornar a votação mais segura e também mais transparente.

1.1 Contextualização

As votações *online* têm grande importância em nossa sociedade. Um exemplo disso são as Consultas Públicas realizadas pelo Senado no portal e-Cidadania[1], conforme a Figura 1.1. Essas consultas públicas são enquetes nas quais os brasileiros podem opinar sobre projetos

[1] ⟨https://www12.senado.leg.br/ecidadania/principalmateria⟩

de lei, projetos de emendas constitucionais e sugestões criadas por cidadãos brasileiros. O resultado dessas consultas são importantes pois demonstram o posicionamento de parte da população para os Senadores com relação a projetos que estão sendo tramitados naquela casa.

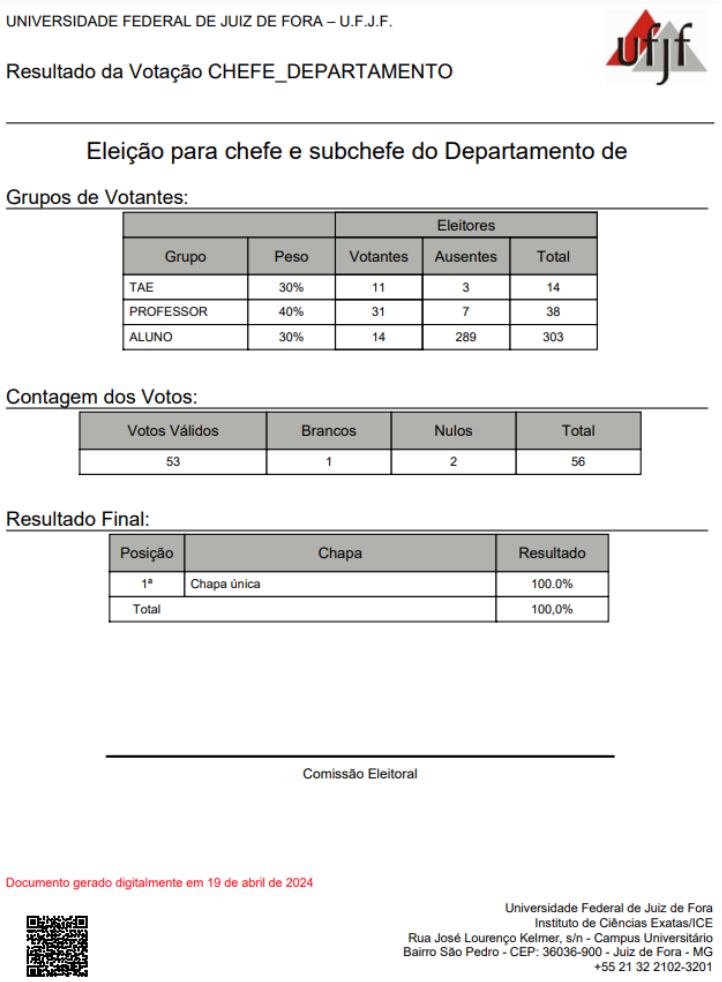
Figura 1.1: Consulta pública de sugestão no portal e-Cidadania

Outro exemplo de votação *online* são as votações do ICE (Instituto de Ciência Exatas) da UFJF. Elas são realizadas através do sistema Integra[2], e podem ser acessadas pelos professores, técnicos administrativos e estudantes. As votações que são realizadas no Integra são responsáveis por eleger a coordenação dos cursos do ICE, os chefes de departamentos do ICE entre outros.

1.2 Descrição do Problema

Um dos principais tópicos de discussão dos atuais meios de votação é referente a transparência. Como é possível garantir para o votante que o seu voto foi recebido pelo sistema e computado corretamente? Essa é uma discussão que vai além da votação eletrônica, e está presente até mesmo nos meios de votação tradicional, onde o resultado é obtido a partir da contagem de cédulas de votos físicas. Um exemplo disso é que uma das principais discussões atuais no Brasil é com relação a integridade das urnas eletrônicas (RUEDIGER, 2020). Segundo uma pesquisa do Datafolha, realizada em maio de 2022, cerca de 27% dos

[2] ⟨https://integra.nrc.ice.ufjf.br/⟩



UNIVERSIDADE FEDERAL DE JUIZ DE FORA – U.F.J.F.

Resultado da Votação CHEFE_DEPARTAMENTO

Eleição para chefe e subchefe do Departamento de

Grupos de Votantes:

		Eleitores		
Grupo	Peso	Votantes	Ausentes	Total
TAE	30%	11	3	14
PROFESSOR	40%	31	7	38
ALUNO	30%	14	289	303

Contagem dos Votos:

Votos Válidos	Brancos	Nulos	Total
53	1	2	56

Resultado Final:

Posição	Chapa	Resultado
1ª	Chapa única	100.0%
Total		100,0%

Comissão Eleitoral

Documento gerado digitalmente em 19 de abril de 2024

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas/ICE
Rua José Lourenço Kelmer, s/n - Campus Universitário
Bairro São Pedro - CEP: 36036-900 - Juiz de Fora - MG
+55 21 32 2102-3201

Figura 1.2: Documento com resultado da votação realizada no Integra para eleger o chefe e subchefe do Departamento de Química

brasileiros desconfiam do sistema do sistema eletrônico de votação das eleições do Brasil (EXAME, 2022).

Em alguns sistemas de *e-voting* que temos disponíveis atualmente, não há uma forma do eleitor verificar o seu próprio voto. Um exemplo disso são as votações realizadas pelo Integra, conforme a Figura 1.2. Nesse sistema não existem mecanismos para o votante auditar o seu própio voto.

Outros sistemas de votações eletrônicas que não são totalmente transparentes com o público são as votações de grandes *reality shows*. O votante, em geral, acessa o site disponibilizado, se autentica quando necessário, e depois deposita o seu voto. Ao fim da votação é realizada a divulgação do resultado da votação. Sendo assim, não é possível que o votante consiga verificar se o seu voto foi corretamente computado.

1.3 Motivação

A tecnologia *blockchain* surgiu a partir do Bitcoin, primeira criptomoeda lançada por Satoshi Nakamoto em 2008 (NAKAMOTO, 2008). Desde então, essa tecnologia tem ganhado popularidade e vem sendo utilizada em diferentes vertentes, sendo uma delas o mercado financeiro. Em 2024 o mercado de criptomoedas atingiu o valor de 2 trilhões de dólares de capitalização (COINTELEGRAPH, 2024).

Algumas das principais características dessa tecnologia são: a transparência, a descentralização e a segurança. Todos os registros de uma *blockchain* pública são acessíveis por qualquer usuário, isso traz transparência. Além disso, a *blockchain* usa de mecanismos de criptografia avançada para garantir segurança dos dados armazenados.

Diante disso, a tecnologia *blockchain* é um recurso promissor para aprimorar os sistemas de votação, oferecendo maior transparência. Além de tornar o processo mais aberto, a *blockchain* proporciona maior resistência a fraudes, fortalecendo a confiança no sistema e garantindo a integridade dos resultados. A *blockchain* também é capaz de trazer um mecanismo muito importante: a possibilidade do votante auditar o seu próprio voto.

1.4 Objetivos

1.4.1 Objetivo Geral

Este trabalho propõe desenvolver um mecanismo de votação eletrônica auditável utilizando a tecnologia *blockchain*. O principal objetivo é criar um sistema de votação que seja auditável, transparente e seguro, aproveitando as características da tecnologia *blockchain*, como imutabilidade e rastreabilidade.

O escopo deste trabalho se concentra na implementação de um sistema de voto eletrônico confiável e verificável, garantindo que cada voto possa ser auditado sem comprometer a integridade do processo. Destacamos que este estudo não abordará o anonimato do voto, focando exclusivamente na transparência e segurança do sistema de votação.

1.4.2 Objetivos Específicos

Neste trabalho será apresentada uma contextualização da tecnologia *blockchain* e contratos inteligentes, e também será realizada a implementação de um sistema para realizar a votação. Sendo assim, os objetivos específicos do trabalho são:

- Contextualização do funcionamento da *blockchain*;
- Contextualização dos contratos inteligentes;
- Apresentação de um contrato inteligente na linguagem de programação *Solidity* para realizar uma votação;
- Apresentação de uma interface para realizar o voto e acompanhar a votação utilizando a linguagem de programação JavaScript e *framework React*;

1.5 Metodologia

Inicialmente será realizada uma pesquisa aplicada através de estudos bibliográficos sobre a tecnologia *blockchain* e os atuais sistemas de votação. Após isso, será realizada a segunda etapa do trabalho que consiste em desenvolver um mecanismo de votação eletrônica auditável. Para desenvolver o sistema de votação eletrônica utilizando a tecnologia de *blockchain* será necessária a criação de um sistema com algumas composições, conforme a Figura 1.3.

O primeiro elemento dessa composição é o contrato inteligente, que será explicado na Seção 2.2.4. É nele que serão criadas as regras de como a votação irá funcionar, e como os candidatos e votantes serão cadastrados. Esse contrato tem o seu código-fonte aberto. Além do contrato, será criada uma interface para comunicação com o contrato inteligente. Essa interface será um sistema *web*, que permite cadastrar candidatos, realizar o voto, e acompanhar o resultado da votação.

Para a criação do contrato inteligente será utilizada a linguagem de programação *Solidity*, que será introduzida na Seção 2.3. O desenvolvimento desse contrato pode ser feito através da ferramenta de desenvolvimento Remix (Seção 2.3). Utilizando essa ferramenta, é possível realizar testes, e até mesmo a publicação em uma *blockchain*.

Após a criação do contrato inteligente, será implementada uma *interface* para realização do voto, utilizando a linguagem de programação Javascript e o *framework React*. Nessa *interface* será realizada a comunicação com o contrato inteligente, utilizando a biblioteca *ethers*. Essa biblioteca permite enviar e obter informações do contrato inteligente. Nessa mesma *interface* será exibida a contagem dos votos em tempo real.

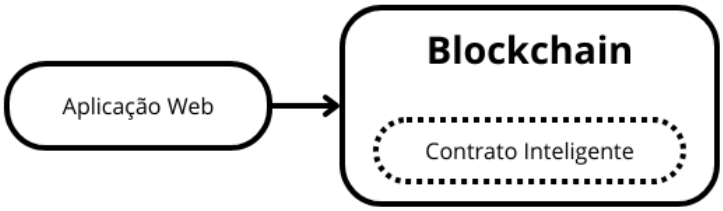


Figura 1.3: Comunicação entre os componentes

1.6 Organização

Esta monografia está organizada em cinco capítulos. O primeiro capítulo introduz o tema, explicando sobre os atuais sistemas de votação *online* e como é possível utilizar a tecnologia *blockchain* para torna-los mais transparentes e permitir o votante auditar o seu voto. No Capítulo 2, são apresentados conceitos importantes para a compreensão do trabalho, como: *Blockchain*, *Ethereum* e Contratos Inteligentes. O Capítulo 3 apresenta os trabalhos relacionados que foram utilizados para o desenvolvimento dessa pesquisa. No Capítulo 4 é apresentado o sistema que foi desenvolvido. Por fim, no Capítulo 5 é apresentada uma conclusão do trabalho, juntamente com discussões e possibilidades de trabalhos futuros.

2 Fundamentação Teórica

Este capítulo está organizado da seguinte forma: na Seção 2.1 os conceitos referentes a *blockchain* são introduzidos, sendo explicados o seu funcionamento, os diferentes tipos de *blockchain*, como são realizadas as transações e, além disso, os protocolos de consenso de uma *blockchain*. Já na Seção 2.2, são apresentados alguns conceitos da plataforma *Ethereum* onde são apresentados os componentes dessa plataforma, e características como as taxas de transação, que são chamadas de *Gas* (Gás), sobre as contas que os usuários utilizam para receber e transferir *Ether*. Será discutido também o conceito de contratos inteligentes, que é um dos componentes deste trabalho, além de exemplos do que pode ser construído com base nessa tecnologia. No fim da Seção 2.2.5, será apresentado uma ferramenta que permite visualizar transações, contratos e contas da plataforma *Ethereum*. Por fim, na Seção 2.3 é apresentada a linguagem de programação de contratos inteligentes da *Ethereum*.

2.1 Conceitos sobre *blockchain*

2.1.1 Introdução

A *blockchain* é uma tecnologia de armazenamento distribuído que funciona de maneira semelhante a um livro de registros. Sua estrutura é composta por blocos, onde cada bloco aponta para o bloco anterior, formando uma cadeia que remonta ao bloco inicial, conhecido como bloco gênesis. Dentro de cada bloco, há uma lista de transações, que constituem os registros da *blockchain*. Quando um novo bloco é criado, ele armazena as transações mais recentes (SOLORIO KEVIN; KANNA, 2019). Uma vez inserido na *blockchain*, um registro não pode ser alterado ou removido, tornando todos os registros imutáveis.

Essa tecnologia foi criada para trabalhar dentro de redes descentralizadas. Dessa forma, existem inúmeros nós (computadores) que fazem parte de uma *blockchain*, e todos

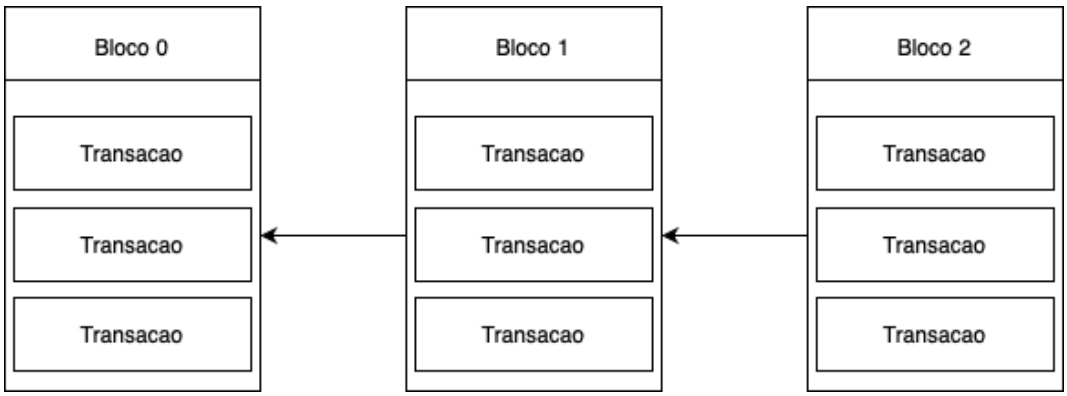


Figura 2.1: Estrutura da *blockchain*. Baseado em SOLORIO Kevin; KANNA (2019).

esses nós possuem a mesma cópia de toda cadeia de blocos (SOLORIO KEVIN; KANNA, 2019). No caso de uma nova transação inserida em um bloco falhar na validação em um desses nós, a transação não é concluída (SOLORIO KEVIN; KANNA, 2019). Dessa forma, todos os nós de uma *blockchain* compõem a *blockchain*, e são responsáveis por processar as transações em consenso.

2.1.2 Tipos de *blockchain*

Existem diferentes classificações para uma *blockchain*. Logo abaixo citamos alguns tipos de *blockchain* e sua descrição.

***Blockchain* pública**

Uma *blockchain* é pública quando qualquer pessoa pode fazer parte dela sendo um nó e quando seu funcionamento é completamente transparente e aberto (SOLORIO KEVIN; KANNA, 2019). Esse tipo de *blockchain* é utilizado por projetos totalmente abertos, como por exemplo: *Bitcoin* e *Ethereum*.

***Blockchain* privada ou permissionada**

Uma *blockchain* é privada quando seu conteúdo e permissão de fazer parte do nó são totalmente privados (SOLORIO KEVIN; KANNA, 2019), dessa forma, é possível ter acesso a rede somente com mecanismos de autenticação. Esse tipo de *blockchain* é mais utilizado no meio corporativo, onde as empresas não desejam expor os seus dados.

***Blockchain* semi privada, híbrida ou federada**

Uma *blockchain* é do tipo semiprivado quando possui dados abertos, porém não todos. Dessa forma, alguns dados são públicos e outros dados requerem acesso por meio de uma ferramenta de autenticação.

2.1.3 Transações

Quando uma transação é criada, inicialmente ela é transferida para um conjunto de transações não confirmadas (SOLORIO KEVIN; KANNA, 2019). Após isso, é criado um bloco onde irá conter a transação em questão (neste bloco pode conter outras transações também). A transação será processada e validada, e quem processa vai depender de qual Algoritmo de Consenso é utilizado (será explicado na seção 2.1.3). Após a transação ser validada e processada, todos os nós da *blockchain* precisam confirma-la, garantindo que está em conformidade de acordo com o livro de registros (todos os blocos da *blockchain*). Caso um nó não confirme a transação, ela não é realizada. Com isso, você garante que todos os nós da rede validaram a transação e que não há inconformidade nela.

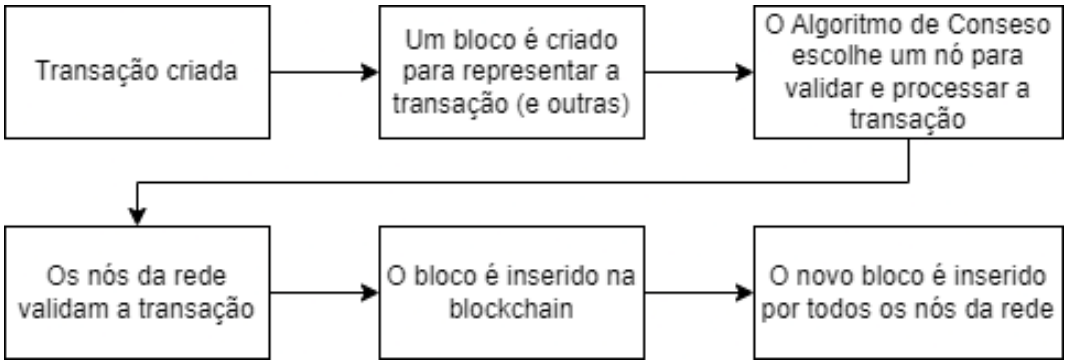


Figura 2.2: Fluxo da criação de uma nova transação em uma *blockchain*

2.1.4 Protocolos de Consenso

Os protocolos de consenso são responsáveis por coordenar a forma como os nós distribuídos irão, em conjunto, validar as transações (SOLORIO KEVIN; KANNA, 2019).

Proof of work **(PoW)**

O protocolo de consenso PoW, traduzindo, prova de trabalho, é um protocolo onde os nós realizam uma competição de quebra criptográfica. Para isso, é necessário poder computacional, para realizar o cálculo hash inúmeras vezes até encontrar uma hash com zeros suficientes na frente (SOLORIO KEVIN; KANNA, 2019). Esse protocolo pode ser chamado também de força bruta.

O protocolo de PoW é utilizado atualmente na *blockchain* da criptomoeda *Bitcoin*, onde os nós que competem para validar a transação utilizam de energia e poder computacional para tentar resolver o problema.

Um dos principais problemas do protocolo de PoW é o seu alto gasto energético e computacional. Estima-se que para validar uma única transação de *Bitcoin* são gastos 1418,05 kWh. Essa quantidade é suficiente para abastecer uma casa de uma família média dos Estados Unidos por 48,6 dias (DIGICONOMIST, 2022).

Proof of stake **(PoS)**

O protocolo de consenso PoS, traduzindo, prova de participação ou prova de montante, é um protocolo que surgiu como alternativa para o PoW. Nesse protocolo, os participantes reservam parte de seus ativos para serem bloqueados temporariamente para participar no processo de validação de transações e criação de novos blocos. Dessa forma, o algoritmo irá escolher um dos participantes aleatoriamente, porém quem possui maior quantia de ativos reservados, têm mais chances de ser escolhido. Dessa forma, esse algoritmo prioriza quem possui maior quantidade de ativos.

A criação do algoritmo PoS varia entre as *blockchains*. Diversos mecanismos são desenvolvidos para aumentar a probabilidade de participação daqueles com menores quantias de criptoativos. Um exemplo é a redução da chance de seleção imediata de um competidor recém-escolhido, favorecendo assim outros participantes. Atualmente, esse protocolo é utilizado na plataforma *Ethereum* (ETHEREUM, 2024).

***Proof of Authority* (PoA)**

O protocolo de consenso PoA, traduzindo, prova de autoridade, é um protocolo normalmente utilizado em *blockchain*s onde a criação de blocos (validação de transações) são de responsabilidade de determinadas entidades. Dessa forma, somente entidades selecionadas serão responsáveis por validar as transações (SOLORIO KEVIN; KANNA, 2019). Esse tipo de protocolo é frequentemente utilizado em *blockchain*s privadas. Dessa forma, o protocolo PoA utiliza a confiança das entidades, que são responsáveis pelos nós onde são validadas as transações.

2.1.5 *Bitcoin*

O *Bitcoin* é uma criptomoeda descentralizada, que utiliza a tecnologia *blockchain* com protocolo de consenso de PoW (NAKAMOTO, 2008). Essa moeda foi citada em um artigo em 2008, e posteriormente lançada em 2009. O autor do artigo publicado é nomeado como Satoshi Nakamoto, porém até a data deste artigo não se conhece a pessoa que o escreveu. Dessa forma, especula-se que esse nome trata-se de um nome falso criado para representar o criador.

Figura 2.3: Gráfico representando a unidade de *Bitcoin* (1 BTC) em Reais (R$) (GOOGLE, 2024)

Desde sua criação até os dias de hoje a moeda se popularizou e teve uma enorme valorização. De novembro de 2015 até abril de 2024, o *Bitcoin* acumula alta de aproximadamente 26.672,95% (GOOGLE, 2024). Atualmente, o BTC é a criptomoeda com maior valor de mercado, atingindo capitalização de mercado de US$ 1,24 trilhões. (COINMARKETCAP, 2024).

Atualmente, o BTC se popularizou e diversas pessoas o adotaram como forma de investimento, devido a sua alta volatilidade. Nos dias de hoje, é possível investir em fundos de investimentos que contém BTC (B3, 2021) e até mesmo comprar e vender a criptomoeda através de alguns bancos (NUBANK, 2022).

2.2 Conceitos de *Ethereum*

Ethereum é uma plataforma descentralizada, que utiliza a tecnologia *blockchain* com o protocolo de consenso de PoS (ETHEREUM, 2024). Essa plataforma surgiu em 2014 e teve sua publicação no mesmo ano, pelo Vitalik Buterin (ETHEREUM, 2024). Nessa plataforma, a criptomoeda utilizada é o *Ether* (ETH), que teve uma alta valorização e atualmente é a segunda maior criptomoeda com capitalização de mercado de US$ 395 bilhões (COINMARKETCAP, 2024).

A principal diferença que o *Ethereum* trouxe quando foi lançado em 2014, foi a tecnologia de contratos inteligentes (será explicado na Seção 2.2.4), dessa forma foi possível a criação de *Tokens*, *NFTs* e até mesmo a criação de aplicações descentralizadas.

2.2.1 *Gas* (Gás)

Para realizar transações na plataforma *Ethereum*, é necessário pagar uma taxa em *Ether* (ETH) para os validadores (SOLORIO KEVIN; KANNA, 2019). O algoritmo de consenso utilizado atualmente é o PoS, dessa forma, as pessoas que deixarem ETH em *stake* irão receber o valor referente às taxas de transação. O valor da taxa varia constantemente, devido à demanda dinâmica na rede: quando há mais transações, os preços sobem para priorizar confirmações, e quando a demanda cai, os preços diminuem. Além disso, atualizações e eventos específicos também influenciam essa variação. Uma forma de visualizar o custo atual e até mesmo o histórico dessa taxa é através do *Ethereum Gas Tracker* (Rastreador de gás da Ethereum) [3].

[3] ⟨https://etherscan.io/gastracker⟩

2.2.2 Contas

Uma conta na plataforma *Ethereum*, é representada por um número hexadecimal, por exemplo: 0xaab27b150451726ec7738aa1d0a94505c8729bd1 (SOLORIO KEVIN; KANNA, 2019). A maioria das transações de ETH são transações entre contas na plataforma *Ethereum* (SOLORIO KEVIN; KANNA, 2019), dessa forma, caso eu deseje transferir ETH da minha conta para a conta de outra pessoa, eu preciso do endereço da outra conta. Na plataforma *Ethereum*, é possível também transferir ETH para contratos inteligentes (SOLORIO KEVIN; KANNA, 2019).

2.2.3 *Metamask*

Metamask é uma carteira digital que é instalada como extensão de um navegador. Essa carteira permite que os usuários interajam com a rede *Ethereum* e com *DAPPs* (aplicativos descentralizados). A *Metamask* é utilizada em *DAPPs* para que o usuário interaja com contratos inteligentes e também é usada para assinar transações.

2.2.4 Contratos Inteligentes

Contratos inteligentes são programas lógicos que são armazenados na *blockchain*. Na plataforma da *Ethereum*, eles são programados na linguagem de programação *Solidity* (será explicado na Seção 2.3), e uma vez publicados na *blockchain*, não podem ter o seu código alterado. Dessa forma, os contratos inteligentes são imutáveis. Portanto, é possível construir aplicações que estarão distribuídas na *blockchain* e com código aberto.

Quando um contrato é publicado na plataforma *Ethereum*, é gerado uma hash única que é o seu identificador, o código do contrato. Dessa forma, quando você deseja procurar por alguma informação de um determinado código de contrato você pode buscar por esse identificador no *Ethereum blockchain Explorer* (será explicado na Seção 2.2.5). Além do código de contrato, é gerado também a ABI do contrato, que é a descrição da interface do contrato. A ABI do contrato é necessário para outras aplicações (não necessariamente publicadas na *blockchain* da *Ethereum*) comuniquem com o contrato que você criou, dessa forma, você pode criar aplicações web que utilizam a ABI para obter

dados do contrato ou inserir/atualizar dados.

Token

Um token é um criptoativo que é criado em uma *blockchain* a partir de um contrato inteligente. Dentre alguns tokens mais conhecidos estão o Tether (USDT), que é um token que tem seu valor lastreado no dólar, dessa forma, o USDT é uma forma de transferir ou acumular dólar na plataforma do *Ethereum* (TETHER, 2024). Dessa forma, o Tether possui um contrato inteligente, com o código em *Solidity*.

NFT

Um NFT é muito semelhante a um token, porém a sua diferença é que um NFT é um token não-fungível, dessa forma, um NFT é algo que é exclusivo e não pode ser repartido. Por exemplo, token podem ser utilizados para representar moedas como o dólar, já NFTs não podem. NFTs são utilizados para representar bens que não podem ser repartidos, como obras de arte. Da mesma forma que os tokens, os NFTs são programados e possuem um contrato inteligente.

Token Standards (Padrão de Tokens)

Padrões de tokens são padrões que os programadores podem seguir quando estão implementando um contrato inteligente (ETHEREUM, 2023). Dessa forma, caso um programador deseje criar um Token para representar uma moeda, ele pode implementar a interface ERC-20 dentro do seu contrato inteligente, que é o padrão para tokens de moedas. Na Tabela 2.1 é possível verificar alguns padrões de tokens da *Ethereum*.

DApp

DApps são aplicativos que funcionam em redes descentralizadas, por exemplo, aplicativos que operam na *blockchain* da *Ethereum*, e componentes desses aplicativos podem ser construídos em contratos inteligentes (ETHEREUM, 2022). Um exemplo de um DApp, é a OpenSea, um mercado de comercialização de NFTs. Através dessa plataforma, é possível comprar e vender NFTs, utilizando os contratos inteligentes que foram programados para

Tabela 2.1: Padrões de tokens (Traduzido de Ethereum (2023))

Padrão de Token	Descrição
ERC-20	Uma interface padrão para tokens fungíveis (intercambiáveis), como tokens de votação, tokens de staking ou moedas virtuais.
ERC-721	Uma interface padrão para tokens não fungíveis, como uma escritura de arte ou uma música.
ERC-777	ERC-777 permite que as pessoas criem funcionalidades extras em cima de tokens, como um contrato de mixagem para melhorar a privacidade das transações ou uma função de recuperação de emergência para salvá-lo se você perder suas chaves privadas.
ERC-1155	ERC-1155 permite negociações mais eficientes e agrupamento de transações, economizando custos. Esse padrão de token permite a criação de tokens utilitários (como $BNB ou $BAT) e tokens não fungíveis como CryptoPunks.
ERC-4626	Um padrão de cofre tokenizado projetado para otimizar e unificar os parâmetros técnicos de cofres com rendimento.

realizar esse comércio (OPENSEA, 2022).

2.2.5 *Ethereum blockchain Explorer* (Explorador da *blockchain Ethereum*)

O explorador da *blockchain Ethereum*, conhecido como *Etherscan* (Figura 2.4), é uma aplicação web onde é possível obter todas as informações referentes a *blockchain*, como os blocos e transações recentes. É possível também acessar contratos inteligentes, visualizar transações do contrato, obter ABI dos contratos e até mesmo visualizar o código fonte dos contratos. Através do explorador, é possível obter também o montante de *Ether* que um determinado usuário possui, quais tokens ele possui e também quais são as transações que esse usuário realizou.

2.2.6 *Sepolia*

A *Sepolia* é uma *blockchain* pública de *PoS*. É uma das principais rede de teste da *Ethereum*, onde os desenvolvedores podem obter *Ether* gratuitamente através de *faucets* (sites onde você resgata ETH de forma gratuita), para desenvolverem *DAPPs* e testarem antes de subirem para a rede principal da *Ethereum*.

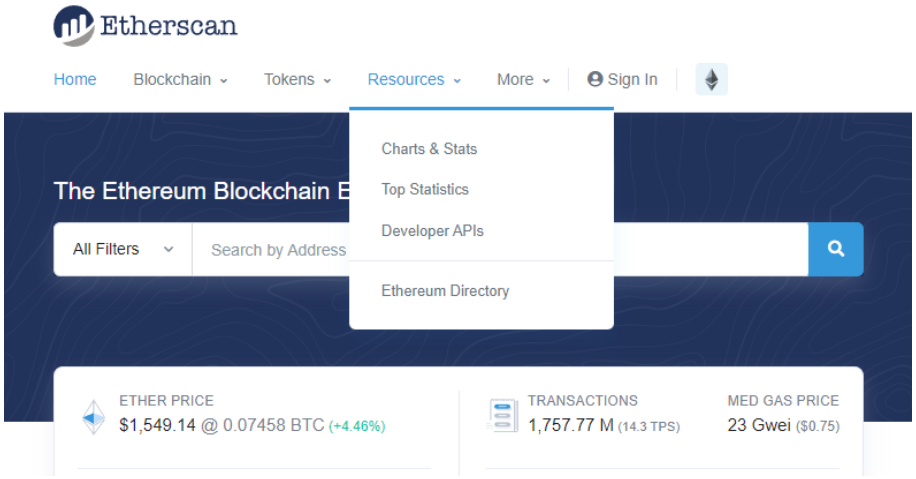


Figura 2.4: Etherscan (Imagem reproduzida através de Etherscan (2022))

2.3 Conceitos de *Solidity*

Solidity é uma linguagem de programação orientada a objetos criada para utilização especificamente em desenvolvimento de contratos digitais (SOLIDITY, 2022). Essa linguagem é utilizada em diversas plataformas como *Ethereum*, *Binance Smart Chain* e *Tron*, sendo *Ethereum* a mais conhecida. Essa linguagem tem como principal influência a linguagem C++ (SOLIDITY, 2022).

O *Remix* é uma IDE online para desenvolver contratos inteligentes na linguagem de programação *Solidity* para a plataforma *Ethereum* (REMIX PROJECT, 2024). A partir dela, é possível identificar problemas léxicos e semânticos do código, realizar testes de desenvolvimento e até mesmo realizar publicação em plataformas como *Ethereum* entre outras.

Segue um exemplo de um contrato inteligente implementado em *Solidity*. Nota-se que na primeira linha do código é definido a linguagem e qual versão dela é utilizada. Após isso é criado o contrato chamado *SimpleStorage* e dentro dele é criado um estado chamado *storedData*, uma função chamada *set* que altera o estado e uma função *get* que obtém o estado. Dessa forma, esse pequeno trecho de código é um contrato onde se armazena uma estado do tipo *uint*, e é possível alterar o valor desse estado e obtê-lo.

```
1 pragma solidity ^0.4.0;
2 contract SimpleStorage {
3     uint storedData;
4     function set(uint x) public {
```

```
5         storedData = x;
6     }
7     function get() public view returns (uint) {
8         return storedData;
9     }
10 }
```

Exemplo de um contrato inteligente implementado em Solidity obtido de Remix Project (2024)

2.4 Considerações finais

Neste capítulo, foram apresentados os principais conceitos relacionados à *blockchain* e *Ethereum*. Através desses conceitos, foi possível compreender o funcionamento de uma *blockchain* e quais os tipos de *blockchain* existentes. Também foram apresentados diferentes *blockchain*s, como o *Bitcoin*, que é uma criptomoeda que teve alta valorização desde a sua criação, e o *Ethereum*, que trouxe a novidade dos contratos inteligentes que é um dos componentes desse trabalho. Foi possível compreender que os contratos inteligentes trouxeram inúmeras possibilidades, como a criação de tokens e NFTs, além do desenvolvimento de aplicações descentralizadas.

Neste capítulo foi apresentado também a linguagem de programação *Solidity*, que é a linguagem utilizada para implementar contratos digitais. Foi apresentado também um exemplo de contrato inteligente implementado em *Solidity* e uma explicação do que o contrato faz. Dessa forma, é possível compreender um pouco a sintaxe da linguagem.

3 Trabalhos Relacionados

Esse capítulo traz uma comparação entre trabalhos relacionados a sistemas de votação utilizando *blockchain*. Nas Seções 3.1 à 3.7 são apresentados alguns trabalhos relacionados e na Seção 3.8 é feita uma comparação entre todos os trabalhos. Os trabalhos apresentados nesta seção têm como objetivo construir um sistema de votação utilizando a tecnologia *blockchain*. Em alguns artigos é proposto um sistema para as eleições brasileiras e em outros é proposto um sistema para votações corporativas.

3.1 Sistema de voto eletrônico baseado em *blockchain*

O trabalho Niwa (2019) tem como objetivo criar um sistema de votação eleitoral onde é utilizado a tecnologia *blockchain*. Inicialmente, o autor menciona os sistemas de votação utilizados atualmente no mundo, que são o de cédulas físicas (utilizado na maioria dos países) e o de votação eletrônica (utilizado atualmente no Brasil). Ele menciona também que um dos principais problemas das cédulas físicas é que em regiões onde há grande volume de votos, a contagem dos votos pode levar mais tempo. Além disso, as cédulas físicas são passíveis de fraudes, através de contagem de votos equivocada e cédulas mal impressas.

Como motivação do trabalho, o autor utilizou de problemas de segurança que ocorreram nas urnas eletrônicas nos últimos anos. Sendo um desses problemas, um caso que aconteceu em Marília, SP no ano de 2004, onde algumas seções eleitorais enviaram os resultados eleitorais antes mesmo do início da votação (CUNHA S. S. D.; MARCACINI, 2010). É comentado também o grande número de abstenções nas eleições de 2018, citando que o voto pode ser feito através de casa pelo sistema que ele propõe.

Na parte de desenvolvimento, o autor cita que para funcionamento do sistema, seria necessário a criação de uma *blockchain* que seria um *fork* da *blockchain* do *Bitcoin*, onde o governo seria responsável pela infraestrutura da rede: máquinas que compõe os nós da rede. Além disso, o governo iria precisar criar todas as cédulas de votação na

rede e distribuir os *Bitcoins* para que essas cédulas realizassem votos. Seria necessário mudanças no protocolo do Bitcoin originalmente copiado no *fork*, como a diminuição da dificuldade do processamento das transações, pois devido ao alto volume de votos, a rede precisa estar preparada para processar esses votos em maior volume. Ele também propõe alterar o protocolo para permitir a criação de tokens, semelhante ao protocolo *Ethereum*. É possível verificar o desenho que exemplifica a arquitetura na Figura 3.1.

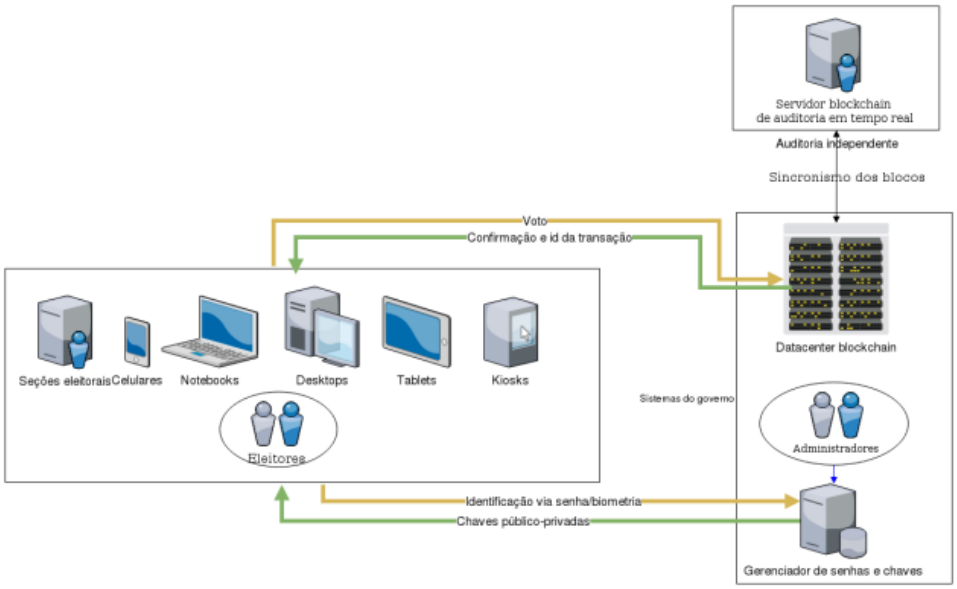


Figura 3.1: Arquitetura do sistema (NIWA, 2019)

O autor realizou testes utilizando um servidor e duas máquinas. Durante os testes de simulação de votos, ele não conseguiu ultrapassar a vazão de 2 milhões de votos por hora, processando ao fim de 24h no máximo 43 milhões de votos. Ele cita que essa limitação se deve principalmente aos equipamentos utilizados para simulação. Os testes de apuração conseguiram bons resultados, conseguindo trazer 50 milhões de votos apurados em tempo de 176,852 segundos.

O uso da tecnologia para processamento dos votos mostrou resultados bem positivos, visto que com testes realizados com máquinas de poder computacional médio, foi possível processar 2 milhões de votos por hora. O autor sugere um *fork* do *Bitcoin* com alterações para criação de tokens, talvez uma opção melhor seria um *fork* da *Ethereum*, visto que ela já tem implementada essa e outras funcionalidades que pode aprimorar o trabalho.

3.2 Sistema de eleição utilizando contratos inteligentes

O trabalho Martinho e Neto (2019) tem como principal objetivo propor um sistema de votação utilizando a tecnologia *blockchain*, mais precisamente utilizando contratos inteligentes no protocolo *Ethereum*. O artigo inicia com uma introdução explicando a história de criação do *Bitcoin* e do *Ethereum*, após isso é realizada uma explicação breve do que são os contratos inteligentes e como eles funcionam no protocolo da *Ethereum*. O autor também comenta um pouco sobre *e-voting* (Eleições eletrônicas), no qual ele menciona a Estônia, onde foi implementado um sistema de *e-voting* para sua eleição nacional.

A motivação que o autor utilizou foram as características da tecnologia blockchain, que são propositivas para utilização em sistemas eleitorais. Sendo uma dessas características a segurança no armazenamento de informações e facilidade de acesso, anonimidade das transações e transparência. É mencionado também a questão de que eleições com cédulas de papel pode haver fraude, diferentemente do sistema utilizando o protocolo *Ethereum*, que seria fisicamente impossível, segundo o autor.

Na parte do desenvolvimento, o autor desenvolveu um aplicativo descentralizado, com os seguintes objetivos: autenticidade dos eleitores, apenas eleitores aptos poderão votar, não haja duplicidade de votos, voto criptografado e anônimo e resultados públicos. Para isso, o autor desenvolveu um contrato inteligente para a plataforma da *Ethereum* na linguagem de programação *Solidity*. No contrato inteligente foram codificadas todas as regras de acordo com os objetivos mencionados. Além disso, foi criado uma interface web que se comunica com o contrato inteligente para realizar o voto.

Tabela 3.1: Tabela de comparativo de gastos (MARTINHO; NETO, 2019)

Simulação	Criação do contato	Cadastro candidatos	Cadastro eleitores e votação
3 candidatos 8 eleitores	0.002358989 ETH R$ 2,47	0.000440315 ETH R$ 0,46	0.001698432 ETH R$ 1,78
13 candidatos 143,3 milhões de eleitores	0.003576 ETH R$ 3,41	0.00190803166 ETH R$ 2,00	30423.1632 ETH R$ 31.914.585,91

No trabalho Martinho e Neto (2019) também é realizado um estudo financeiro de quanto iria custar para o governo realizar eleições utilizando a *blockchain* da *Ethe-*

reum, conforme a Tabela 3.1. Supondo que haveria o cadastro de 13 candidatos e 143,3 milhões de eleitores, o custo seria de 30.423,1632 ETH (Ether), que estava cotado em R$ 31.914.585,91 durante a criação do artigo. É realizado uma comparação com os custos atuais do sistema brasileiro atual, porém no cálculo de comparação, foram levados em conta apenas o auxílio alimentação dos voluntários. Custos como logística, manutenção das urnas eletrônicas e toda infraestrutura não foram levados em conta. Segundo o artigo, os custos com mesários na eleição de 2018 foi de aproximadamente R$ 35 milhões, dessa forma, o sistema que eles propõem teria uma economia de aproximadamente 8,82%.

O autor conclui que um sistema de autenticação online não é viável, pois pode facilitar a fraude durante a votação. Portanto, seria necessário criar uma abordagem para que viabilize o funcionamento do sistema que ele propôs. Além disso, o sistema proposto teria um alto custo para utilização. Talvez, uma opção para resolver esse problema, seria o governo criar um *fork* da plataforma *Ethereum* e montar uma rede para viabilização do sistema proposto. Dessa forma, os valores gastos para criação do contrato e transações na *blockchain*, seriam gastos para infraestrutura do governo criar uma rede para auditar os votos.

3.3 Aplicabilidade da *blockchain* no sistema de eleição departamental da UTFPR

No trabalho Barcelos e Martins (2020), o autor tem como objetivo utilizar a tecnologia *blockchain* para estudar a viabilidade de uso no processo eleitoral de escolha de chefe de departamento da UTFPR campus Ponta Grossa. Para isso, foi criada uma *blockchain* privada e uma interface web para realizar a votação.

Na introdução do trabalho, o autor comenta sobre a tecnologia do *Bitcoin*, explica a diferença sobre *blockchain* pública e privada e além disso comenta sobre os protocolos de consenso PoW (*Proof of Work*) e PoS (*Proof of Stake*). Após isso, o autor comenta que o objetivo do trabalho é utilizar uma *blockchain* privada, onde diferentes pessoas irão ser os nós para garantir a lisura do processo, como diferentes departamentos e diretórios. Dessa forma, os objetivos do trabalho é demonstrar a aplicabilidade da tecnologia *blockchain* no

sistema de votação acadêmico.

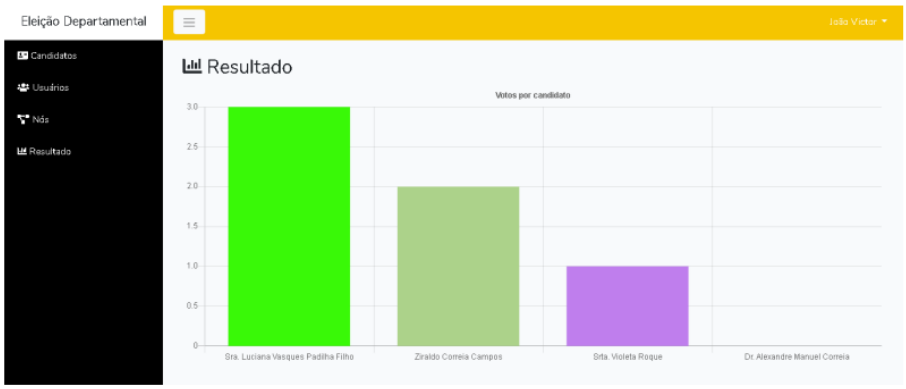


Figura 3.2: Tela de resultados (BARCELOS; MARTINS, 2020)

Na parte de desenvolvimento do trabalho, foram utilizados o *framework* Laravel (PHP) para desenvolvimento da interface e um banco de dados SQL para armazenar os candidatos à eleição e os usuários do sistema que poderão votar. Além disso, foram implementados na interface as operações de *CRUD* (Criação, Leitura, Atualização e Remoção) para candidatos e eleitores. Na interface também foi criado uma tela para registro de nós que poderão participar da *blockchain*, ou seja, caso um departamento tenha interesse em fazer parte da *blockchain* privada para garantir maior lisura do processo, é preciso que seja cadastrado antes. Por fim, foi criada a tela para visualização do resultado da eleição.

Foi criado também uma *blockchain* privada com protocolo de consenso PoW, no qual o autor utilizou como base um código implementado em *Python* com licença MIT. Foram feitas algumas mudanças nessa *blockchain*, sendo as principais: mudança da dificuldade da rede (foi alterada a dificuldade da rede para aumentar a vazão do processamento dos votos), retirada da recompensa pela mineração (visto que é uma *blockchain* privada no qual o objetivo não é recompensar os nós da rede) e registro de nós confiáveis através do administrador do sistema.

Na parte de resultados, o autor cria três subseções nas quais ele comenta os resultados com base nos objetivos específicos do trabalho. Na subseção de endereços privados, o autor comenta que não foi necessário a criação do par de chaves, visto que o sistema que foi criado possui um sistema de identificação centralizador, onde as informações dos eleitores/candidatos estão salvas em um banco de dados. Ele comenta que uma forma de tornar o sistema mais seguro do rastreio de votos seria criptografar as informações do

banco que possam identificar o eleitor, o nome por exemplo. Após isso, o autor comenta sobre os resultados referentes ao anonimato do sistema. Ele explica que foi decidido não transacionar na *blockchain* o eleitor do voto durante a votação, para impedir associação de votos a um eleitor. Além disso, ele comenta que é preciso colocar um atraso aleatório para o transacionamento do voto na *blockchain* com o armazenamento no banco, visto que um usuário com acesso ao banco de dados poderia comparar o *timestamp* do voto na *blockchain* com a alteração do campo *jaVotou* no banco de dados do eleitor. Por fim, na subseção de Segurança e Desempenho, o autor comenta que o desempenho poderia ser melhor se não se utilizasse a *blockchain* com PoW, porém o mesmo diz que a diferença de desempenho é pequena em caso de eleições substanciais.

3.4 Votechain, um sistema de votação implementada com a tecnologia *blockchain*

O trabalho Souza (2019) tem como objetivo apresentar uma alternativa ao modelo atual de votação utilizado no Brasil. Para isso, foi feita a criação de uma *blockchain* privada com protocolo de consenso de PoW, utilizando o *framework* Django da linguagem de programação *Python*, onde todos os nós irão armazenar e processar os votos. Para isso, o autor utilizou do *Docker* para simular a operação de múltiplos nós na rede.

O autor introduz o trabalho comentando sobre o protocolo *Bitcoin*, explicando sobre a sua criação e sobre o sucesso que ocorreu ao fim de 2017, chegando ao valor de R$ 69.700 a unidade do BTC. Após isso, ele usa como motivação a desconfiança de parte da população brasileira com o sistema atual e cita que esse problema faz com que outras pessoas até mesmo peçam pela volta do voto impresso, que seria até pior. Diante disso, ele comenta que países como Japão e Estados Unidos já possuem estudos para adoção de sistemas eleitorais utilizando *blockchain*.

O autor desenvolveu dois componentes que representam o sistema desenvolvido: um componente será o cliente, composto por uma interface web que o eleitor acessa para realizar o voto e o outro componente é o nó que compõe a *blockchain*, que será uma API REST.

Para a estrutura da *blockchain*, o autor realizou a criação de um DER (Diagrama Entidade-Relacionamento), onde ele criou quatro tabelas: *No*, *Usuário*, *Voto* e *Bloco*. Na tabela *No* estão cadastrados todos os nós da rede, contendo IP e Porta de cada um deles. Nas outras tabelas de *Usuário* e *Voto* estarão as informações do usuário e o seu voto. Por fim, na tabela *Bloco* irá conter os votos que foram armazenados no bloco, além disso terá as informações do bloco que compõe a *blockchain*, como *timestamp* e a *hash* do bloco. Foi criado também uma função que define a dificuldade de mineração do bloco atual, isso foi feito para que o autor tenha controle sobre o tempo de processamento dos votos.

Na aplicação cliente, o autor permite que o eleitor realize o cadastro, dessa forma, quando ocorrer a criação de um cadastro, será gerado uma chave pública e privada, podendo o eleitor, salvar a chave privada em um arquivo do seu computador/dispositivo. Essa chave privada será requisitada quando o eleitor for realizar o voto (Figura 3.3). A chave privada e senha em si não são salvas no banco de dados, são salvos somente o resumo dos mesmos. Sendo assim, quando a aplicação não quiser validar as informações, eles receberão o resumo gerado pela aplicação e o cliente validará o resumo recebido com o resumo salvo no banco de dados. Depois de cadastrado, o usuário consegue realizar o voto, informando o cargo do voto, o número do seu candidato e a chave privada.

Figura 3.3: Tela da aplicação cliente durante a votação (SOUZA, 2019)

Na aplicação nó foram utilizadas as tecnologias *Celery* e *RabbitMQ*. Sendo o *Celery* um agendador/gerenciador de tarefas em segundo plano, que foi utilizado para atividades como a mineração dos blocos. O *RabbitMQ* é um gerenciador de filas que foi responsável por receber o voto a ser processado, e em caso de falha/indisponibilidade temporária de um nó, quando o mesmo voltar a ativa, o *RabbitMQ* garante que esse voto será processado na sequência. Durante o processamento do voto, a aplicação verifica se o eleitor já realizou o voto para um determinado cargo, se já foi realizado o voto, o voto mais recente é ignorado.

Por fim, o autor exibe os resultados de um teste realizado em uma máquina com *Docker*, onde foram criadas as instâncias para execução do sistema. Foi criado um contêiner contendo os serviços do *RabbitMQ* e *Celery*, que serão utilizados pelos contêineres nós da rede. Além disso, foram criados outros dois contêineres que representam o cliente da rede (interface web onde são realizados os votos). E também são criados três contêineres que representam os nós da rede, onde esses se comunicam com o contêiner que possui os serviços do *RabbitMQ* e *Celery*. No teste em questão, o autor simula a realização de votos, auditoria em tempo real e exibição de resultados, e chega a conclusão de que o sistema é satisfatório e robusto, sendo suficiente para ser utilizado como base para projetos maiores.

3.5 Aplicação para votação utilizando a tecnologia *blockchain*

O trabalho Jesus e GonÇalves (2018) tem como objetivo aplicar o funcionamento da tecnologia *blockchain*, criando um sistema de votação para ser utilizado internamente dentro de organizações, de forma transparente e confiável. A *blockchain* que o autor utilizou é pública com algoritmo de consenso de PoS (durante a escrita do artigo a plataforma da Ethereum utilizava o protocolo de consenso PoW, porém a partir de 2022 a *Ethereum* passou a usar protocolo de consenso de PoS (MERCADO BITCOIN, 2024)).

Para desenvolver o trabalho, foi criado um contrato inteligente em *Solidity* que funciona dentro da plataforma da *Ethereum* e também uma aplicação para dispositivos

móveis que foi criada utilizando o *framework Ionic*. Dessa forma, o autor criou dois componentes, no qual o aplicativo do dispositivo móvel irá se comunicar com a *blockchain* da *Ethereum*.

No desenvolvimento do contrato foram implementadas as funções de abrir votação, impedindo que os eleitores votem antes da execução dessa votação e também a implementação da função encerrar votação, impedindo votos após a execução da mesma. Foi implementada a função alertar funcionário, na qual o funcionário visualiza uma notificação no aplicativo que indica que ele deve procurar o seu gestor direto. Por fim, foi implementado a função de Registrar voto, na qual é realizado o armazenamento do voto na *blockchain* da *Ethereum*. Para realizar essa comunicação, o autor menciona que foi necessário utilizar da ABI do contrato inteligente que foi implementado, além disso, foi utilizada a biblioteca *ethers*.

Figura 3.4: Tela de votação (JESUS; GONCALVES, 2018)

Por fim, o autor comenta que o trabalho ainda encontra-se em fase de testes e que após a realização dos testes, eles esperam concluir que a votação realizada é segura e pública. Dessa forma, não é possível observar muitos resultados pois o sistema proposto pelo autor encontra-se em desenvolvimento.

3.6 Verificação de Eleição utilizando *blockchain*

O trabalho Macelai (2019) tem como objetivo desenvolver um sistema de validação de votação utilizando a plataforma da *Ethereum*. Além disso, o autor comenta que o objetivo também é realizar um estudo de custos para funcionamento desse sistema que ele propõe.

Na introdução do trabalho, o autor comenta sobre a desconfiança que algumas pessoas têm com o sistema eleitoral atual e que o sistema atual não é totalmente seguro. É citado que um dos maiores problemas do sistema atual é a falta de auditabilidade por se tratar da eleição ocorrer em um ambiente fechado e controlado, dessa forma, não há forma de verificar se um voto em si foi realmente computado. Como forma de resolver esse problema, o autor comenta da tecnologia *blockchain*, que pode auxiliar para aumentar a confiança do sistema por parte da população, garantindo imutabilidade do sistema, onde os votos registrados não podem ser alterados, e além disso garantir que toda população possa auditar as eleições, por se tratar de uma *blockchain* pública. É comentado também a possibilidade de rodar algoritmos complexos em uma *blockchain* garantindo imutabilidade dos dados através dos contratos inteligentes.

O autor pretende desenvolver um módulo para um sistema online de eleição, com foco na auditoria e verificação dos votos a partir de uma *blockchain*. Para isso, o autor espera contribuir para o sistema eleitoral brasileiro, trazendo mais transparência e auditabilidade do processo. Também deseja identificar o limite de volume superior de processamento dos votos, dado que a tecnologia *blockchain* tem uma limitação de transações por segundo e também obter um estudo de custos para utilização desse sistema.

Na parte de desenvolvimento, o autor comenta que pretende evoluir o sistema de votação *Helios*, criando novos *endpoints* e atrelando o funcionamento do sistema a *blockchain* pública da *Ethereum*. Para isso, o autor realiza a criação de três contratos inteligentes:

- Contrato *Voters*: contrato inteligente responsável por armazenar os eleitores aptos a participar da votação. Nele contém um estado onde é salvo a lista de eleitores e uma função para inclusão de eleitores nessa lista;
- Contrato *Votes*: contrato inteligente responsável por armazenar os votos dos elei-

tores aptos a votar. Nele contém um estado para salvar a lista de votos e uma função para incluir novos votos. Além disso, o objeto salvo na lista não contém a informação do votante;

- Contrato *Election*: contrato inteligente onde é armazenado as informações sobre a eleição, como nome, data de criação, data de início da votação, entre outros. Também é armazenada a lista de questões (perguntas). Além disso, são armazenados neste contrato os identificadores dos contratos de *Voters* e *Votes*, além de uma função para adicionar questões (perguntas);

- Contrato *Result*: contrato inteligente onde é armazenado o resultado do voto individual. Nesse contrato é salvo a lista de perguntas com o resultado. De forma que ao fim da eleição é possível resgatar o resultado da votação. Isso é feito através de uma função para incluir questões na lista do estado do contrato;

Por fim, o autor mostra como foi feita a integração do sistema de votação do *Helios* com os contratos criados e publicados na blockchain. Após a eleição feita no sistema *Helios* ser concluída, o autor alterou a interface do sistema e incluiu um botão chamado *publish on blockchain* (publicar na *blockchain*), onde os dados são transmitidos para os contratos inteligentes e podem ser visualizados por todos. O autor chegou a criar um diagrama de sequência que exemplifica o funcionamento do sistema:

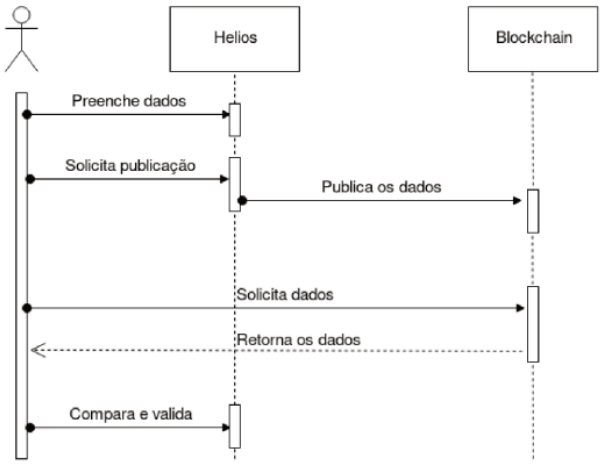


Figura 3.5: Diagrama de sequência (MACELAI, 2019)

Diante disso, é possível que os eleitores e qualquer pessoa tenha acesso a auditoria

gerada pela *blockchain* após a realização das eleições no sistema *Helios*, por se tratar de uma *blockchain* pública. Porém, é preciso destacar que todo o processo de votação que ocorre dentro do sistema *Helios* é em um ambiente fechado e controlado.

3.7 Comparações entre os trabalhos

Os sistemas desenvolvidos entre os trabalhos tiveram algumas diferenças bem interessantes. O Niwa (2019) utilizou da criação de uma *blockchain* pública de PoW que seria um *fork* do *Bitcoin*, enquanto o Martinho e Neto (2019), Jesus e GonÇalves (2018) e Macelai (2019) utilizaram da criação de contratos inteligentes na plataforma da *Ethereum*, uma *blockchain* pública de PoS. O Barcelos e Martins (2020) e Souza (2019) optaram por criar uma *blockchain* privada de PoW.

Alguns trabalhos chegaram a simular cenários com grande volume de votantes, como foi feito no Niwa (2019), onde foi realizado testes de carga para simular grande volume de votação. Nesse cenário em questão, foi conseguido atingir um volume de vazão de 2 milhões de votos por hora, alcançando valor de 43 milhões de votos em 24h. O Martinho e Neto (2019) chegou a realizar um cálculo de custo no caso de utilização do sistema proposto no cenário de eleição brasileiro, onde 143,3 milhões de pessoas votam. Nesse caso, o custo somente do sistema de votação para funcionar na *blockchain* da *Ethereum* para o cenário de 13 candidatos foi de R$ 31.914.585,91.

No trabalhos Barcelos e Martins (2020) e Souza (2019), foi proposto o uso de uma *blockchain* privada, nos resultados finais, a parte de auditoria se dava por parte dos administradores da eleição. Dessa forma, talvez o uso da tecnologia *blockchain* não traga tanta transparência igual aos trabalhos que utilizam uma *blockchain* pública.

Somente os trabalhos Niwa (2019) e Martinho e Neto (2019) propõe o sistema deles para utilização em eleições no Brasil. Os outros trabalhos propõem o sistema desenvolvido, para utilização em votações em ambientes corporativos, em votações em sistemas de departamentos de universidades e outros cenários de votações de pequena/média escala.

Este trabalho visa desenvolver um sistema de votação eletrônica utilizando a *blockchain* pública da Ethereum, semelhante aos trabalhos Martinho e Neto (2019), Jesus e GonÇalves (2018) e Macelai (2019). Contudo, diferente desses outros trabalhos, o foco

está na possibilidade do votante verificar o seu voto e conseguir acompanhar o resultado da votação em tempo real.

Tabela 3.2: Tabela de comparação entre os trabalhos relacionados

Trabalho	Tipo de *blockchain*	Protocolo de Consenso	Contratos inteligentes	Resultados
Niwa (2019)	pública	PoW	Não há	Vazão nos testes de carga.
Martinho e Neto (2019)	pública (*Ethereum*)	PoS	Há um contrato inteligente	Sistema auditável porém possui alto custo.
Barcelos e Martins (2020)	privada	PoW	Não há	Sistema auditável somente por pessoas restritas.
Souza (2019)	privada	PoW	Não há	Sistema auditável somente por pessoas restritas e utilizou de tecnologias centralizadas.
Jesus e GonÇalves (2018)	pública (*Ethereum*)	PoS	Há um contrato inteligente	Sistema auditável, utilizado para votações corporativas porém não foi desenvolvido completamente.
Macelai (2019)	pública (*Ethereum*)	PoS	Há utilização de diferentes contratos inteligentes	Validação dos votos de um sistema centralizado e alto custos.
Este trabalho (2024)	pública (*Ethereum*)	PoS	Há um contrato inteligente	Sistema auditável e resultados da votação em tempo real.

3.8 Considerações finais

Os trabalhos exibidos aqui nessa seção utilizaram de diferentes abordagens e também tiveram objetivos diferentes. Alguns utilizaram de *blockchains* públicas para trazer mais confiança e tornar o sistema mais auditável, porém outras optaram por utilizar *blockchains* privadas, onde o sistema é mais fechado e a auditoria fica restrita a quem tem acesso a *blockchain*. Outro fator interessante foi o estudo de viabilidade que foram realizados. Alguns trabalhos mostraram que optar por uma *blockchain* pública acaba elevando o

custo e inviabilizando o seu uso em grandes votações.

4 Desenvolvimento e Resultados

Neste capítulo, é apresentado o desenvolvimento do sistema de votação utilizando a tecnologia *blockchain*, detalhando como foi o projeto, quais tecnologias foram utilizadas e quais funções foram desenvolvidas. Serão exibidos também imagens que mostram o sistema em funcionamento. O código fonte desenvolvido pode ser acessado através do repositório público hospedado no Github[4].

4.1 Arquitetura do sistema

O projeto foi concebido com a seguinte estrutura: uma aplicação web servirá como interface de comunicação para o usuário. Nesta aplicação, o usuário utilizará a Metamask (carteira de criptoativos) para autenticar-se e assinar suas transações. Em outras palavras, através da Metamask, ele realizará o login na aplicação e executará diversas funções que modificam os estados do contrato inteligente, como realizar votos, cadastrar candidatos, entre outras ações.

A aplicação web estabelecerá uma comunicação direta com a *blockchain* da *Sepolia*, onde o contrato inteligente da votação está hospedado. Dessa forma, ela utilizará a biblioteca *ethers* para invocar as funções necessárias, como listar candidatos, obter o status da votação, entre outras, e com base nessas informações, apresentará os dados de forma clara e acessível para o usuário. A Figura 4.1 resume o funcionamento do sistema.

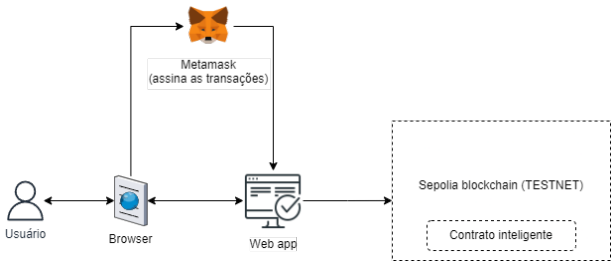


Figura 4.1: Arquitetura do sistema

[4] ⟨https://github.com/outlookcast/e-voting⟩

4.2 Contrato inteligente

Para o desenvolvimento do sistema de votação proposto, é necessário a criação de um contrato inteligente. Nesse contrato, é programado as regras e como será o funcionamento da votação. A linguagem de programação utilizada para desenvolver esse contrato é a *Solidity* - uma linguagem de programação popular para o desenvolvimento de *DAPPs*, e vamos utilizar a blockchain pública de testes da *Ethereum* para publicar o contrato desenvolvido: a *blockchain Sepolia*.

O contrato inteligente foi desenvolvido na ferramenta *Remix*, que permite de forma prática e fácil desenvolver e testar um contrato inteligente. Através dessa ferramenta, conseguimos testar o contrato inteligente sem necessidade de publicar o mesmo em *blockchains* públicas, pois o *Remix* tem máquinas virtuais que permitem a realização de testes durante o desenvolvimento. Essa ferramenta facilita o desenvolvimento de contratos inteligentes, pois uma vez publicado em uma *blockchain* pública, um contrato é imutável, sendo assim, não é possível alterar o seu código fonte. Portanto, a publicação do contrato inteligente na *blockchain* da *Sepolia* será somente após a conclusão do seu desenvolvimento.

4.2.1 Informações armazenadas no contrato inteligente

No contrato inteligente, serão armazenados as seguintes informações:

- O endereço público da conta do criador do contrato (ou chamado de *owner*): Essa informação é resgatada no momento que o contrato é criado (através do construtor) e é salva no exato momento;
- *Array* de candidatos e número de candidatos cadastrados: Cada candidato cadastrado possui um nome, URL com a sua foto e um contador de votos;
- Dicionário de contas que já votaram: Essa informação é utilizada para evitar que uma conta vote mais de uma vez;
- Status da votação: É utilizado para armazenar o status atual da votação;

4.2.2 Funções do contrato inteligente

Nesta subseção, será exibido as principais funções do contrato inteligente e como será o seu funcionamento.

Adicionar candidato

Na função de adicionar candidato, armazenamos as informações do candidato em um *array* de candidatos, adicionando seu nome, URL da sua foto e seu contador de votos, inicialmente zerado.

Essa função pode ser chamada apenas pelo criador do contrato, e somente quando a votação está em estado de "Não iniciada". Dessa forma, não será possível adicionar candidatos durante a votação e após o fim da votação.

Iniciar votação

Nessa função, o status da votação é alterado de "Não iniciado"para "Iniciado". Somente o criador do contrato pode chamar essa função e ela só pode ser executada quando a votação estiver no status "Não iniciado".

Verificação se pode votar

Qualquer usuário pode chamar essa função para verificar se está apto a votar. Essa função valida o status da votação e também verifica se o usuário já realizou um voto, a fim de inibir mais de um voto do mesmo usuário.

Votar

Qualquer conta da blockchain pode realizar um voto, porém esse voto pode ser realizado uma única e exclusiva vez por conta. Para realizar o voto a votação precisa estar em status de "Iniciado".

Finalizar votação

O criador do contrato chama essa função quando deseja que a votação chegue ao fim. Somente o criador do contrato pode chamar essa função, e após ela ser executada, nenhum

voto novo será armazenado no contrato.

Listar candidatos

Qualquer conta da blockchain pode chamar essa função, ela retorna o *array* de candidatos juntamente com o contador de votos. Como essa é uma função que não altera o estado da blockchain, ela não necessita de *Gas* para execução.

4.2.3 Eventos do contrato inteligente

Os eventos em contratos inteligentes são utilizados para notificar ocorrência de ações dentro de contratos inteligentes e funcionam também como uma ferramenta de registro de logs. Sendo assim, são uma ferramenta de log para registrar atividades importantes de mudanças que ocorreram dentro do contrato inteligente.

Esses eventos podem ser inclusive escutados por aplicações. Cada vez que um evento é lançado, a aplicação que está escutando o evento recebe uma notificação e pode tomar uma ação. No nosso contrato inteligente foram criados os seguintes eventos:

Evento de mudança de status da votação

Cada vez que o status da votação tem uma mudança, emitimos o evento *VotingStatusChanged*, que registra qual o novo status da eleição que foi modificado. Dessa forma, é possível que uma aplicação escute esse evento e permita os votos assim que o status for alterado para votação iniciada.

Candidato adicionado

Quando um candidato é adicionado na votação, temos o evento *CandidateAdded* que informa que um novo candidato foi adicionado. Nesse evento registramos o seu nome e foto. Dessa forma, toda vez que um novo candidato for incluído, haverá um registro desse evento.

Voto adicionado

A cada voto registrado, emitimos um evento onde salvamos o endereço público do votante e o seu voto. Esse evento pode ser utilizado para verificar o voto e até mesmo para aplicações escutarem esse evento e obter a contagem dos votos em tempo real.

4.2.4 *Deploy* do contrato inteligente na *Sepolia*

Após desenvolver o contrato utilizando a ferramenta de desenvolvimento *Remix*, precisamos realizar o *deploy* do contrato na *blockchain* da *Sepolia*. Para isso, será necessário ter *ether* na rede da *Sepolia* para pagar a taxa de rede (*Gas*) da criação de contrato. Uma forma de conseguir *ether* na rede da *Sepolia* é utilizar de *faucets* online, que são sites que disponibilizam gratuitamente *ether* e *tokens* em redes de testes.

Uma vez que temos *ether* em nossa conta, utilizamos a ferramenta *Remix* para realizarmos o *deploy* na rede da *Sepolia*. Feito o deploy, o contrato está publicado na *blockchain* de testes da *Ethereum*, sendo assim, conseguimos consumir informações da *blockchain* e conseguimos chamar funções que alteram o estado da *blockchain*.

Para verificar que o contrato foi publicado corretamente na blockchain podemos utilizar do *Ethereum blockchain explorer* da rede da *Sepolia*. Basta acessar o explorer e pesquisar pelo endereço que realizou o *deploy* do contrato na rede da *Sepolia*. Acessando a página, basta ir em transações, onde é possível encontrar a transação de criação de contrato e lá tem o endereço que foi gerado para o contrato criado.

Após o contrato ser publicado na blockchain da *Sepolia*, precisamos obter a ABI do contrato publicado, para que a aplicação Web, através da biblioteca *ethers*, realize chamada de funções do contrato inteligente que foi publicado.

4.3 Aplicação *Web* e telas desenvolvidas

Além do contrato inteligente, foi desenvolvido uma aplicação *Web* utilizando da linguagem de programação Javascript e o *framework React*. Nessa aplicação *Web*, é feita uma comunicação com a *blockchain* da *Sepolia* para obter as informações da votação e também é realizado interações com o contrato, para realização de cadastro de candidatos, iniciar

a votação, realizar voto e encerrar a votação.

Para o desenvolvimento da aplicação *Web* foi utilizado a biblioteca *ethers*: para realizar a autenticação utilizando a *Metamask* e para a realização de chamadas a funções do contrato que realizam alterações nos estados do contrato, como adição de candidatos e realizar voto. Abaixo estão listadas as telas que foram desenvolvidas.

4.3.1 Tela de login

A primeira tela desenvolvida foi a tela de login, onde o usuário utiliza a Metamask (instalada em seu navegador) para realizar a autenticação e para assinar as transações futuras. Nessa tela há um botão que, após clicado, utiliza a biblioteca *ethers* para solicitar ao usuário que aceite que a Metamask conecte com o sistema web e após isso coletamos o endereço publico do usuário logado e direcionamos para a tela de votação.

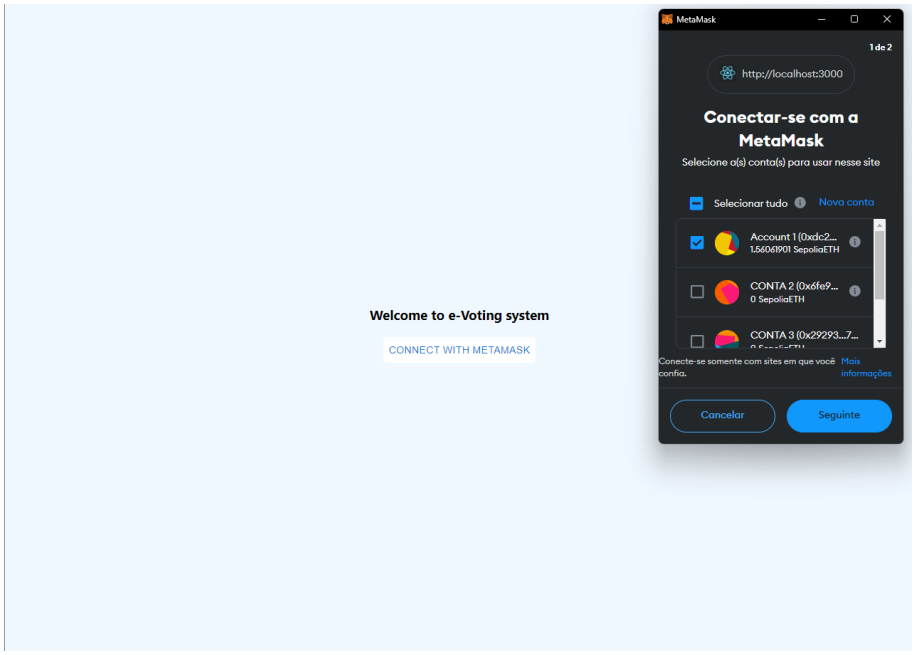


Figura 4.2: Tela de login

4.3.2 Tela inicial

Após realizar login, o sistema leva para a tela inicial. Nessa tela existem duas visões possíveis: uma que o dono do contrato terá e outra que os outros usuários em geral terão. O dono do contrato terá uma visão a mais que os outros usuários, que haverá

dois botões: um de criar candidato e outro de iniciar e finalizar a votação. Isso ocorre porque definimos como regra do contrato inteligente que somente o dono do contrato pode realizar tais ações.

Ao carregar a tela inicial, realizamos diversas chamadas ao contrato inteligente. Chamamos a função *getCandidateList* para obter a lista de candidatos participantes da votação, junto com seu número de votos. Obtemos também os valores de alguns estados do contrato, sendo eles o *votingStatus* e *candidatesNumber* para obter o status da votação e o número de candidatos.

Durante o carregamento, realizamos a chamada da função do contrato *canVote*, essa função é para que o contrato inteligente verifique se o usuário logado pode votar. Essa função checa o status da votação e se o usuário já realizou o voto. Caso a função retorne *false*, então desabilitamos o botão de votar, para impedir que o usuário chame a função de votar do contrato.

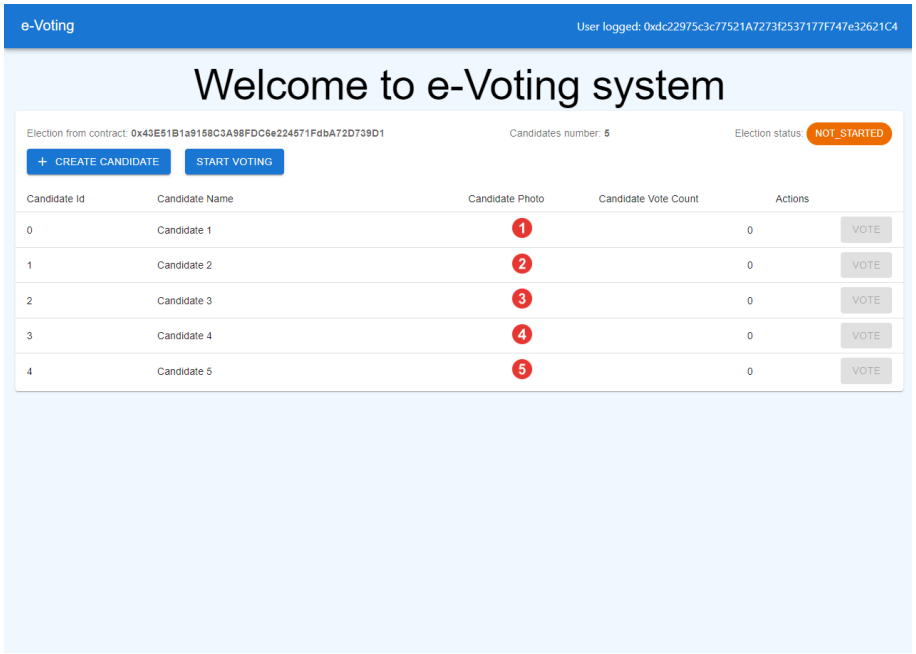


Figura 4.3: Tela inicial do sistema

Na tela inicial, é possível que o dono do contrato inicie a votação. Para que o dono inicie a votação, será solicitada uma confirmação de transação na *Metamask* para que seja chamado a função *startVoting* do contrato inteligente.

Após o dono do contrato aceitar a transação e ela ser confirmada, o sistema irá carregar todas as informações do contrato na tela, e o botão de votar será permitido para

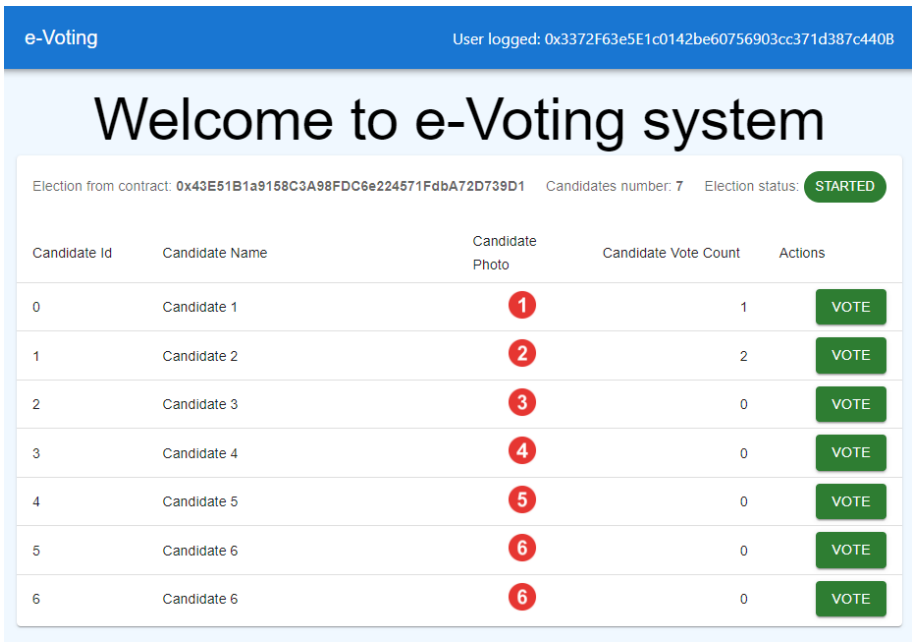


Figura 4.4: Tela inicial após dono do contrato iniciar votação

todos os usuários do sistema que ainda não realizaram o voto. Não será mais possível que o dono do contrato adicione candidatos durante a votação, essa foi uma regra que foi definida no contrato inteligente e foi incluída na interface também. Sendo assim, o botão de adicionar candidato que o dono do contrato tem visão ficará desabilitado.

Para realizar o voto, basta o usuário clicar em Vote e após isso a *Metamask* irá solicitar que ele aceite a transação e pague a taxa de rede (*Gas*). Após a transação ser confirmada o sistema irá recarregar as informações do contrato mostrando contagem de votos atualizada e o botão de votar estará desabilitado para esse usuário que já votou, pois a função *canVote* irá retornar *false*, visto que ele já realizou o voto.

A qualquer momento após o início da votação, o dono do contrato pode finalizar a votação clicando no botão de finalizar votação. Esse botão irá aparecer somente quando a votação estiver em andamento.

4.3.3 Tela de cadastro de candidato

Essa é uma tela que é protegida, sendo assim, somente o dono do contrato consegue acessá-la. Para acessar, basta clicar no botão de adicionar candidato na tela inicial. Nessa tela, há duas caixas de texto onde devem ser informados o nome do candidato e uma URL de sua foto. Essas informações serão utilizadas para cadastrar o candidato na *blockchain*.

Após preencher as informações e clicar em criar, a *Metamask* irá solicitar para que o usuário aceite a transação, que será enviada para a *blockchain*. Essa transação irá cobrar uma taxa de rede (*Gas*) para que seja executada. Caso o usuário rejeite a transação pela *Metamask*, a interface irá exibir um alerta informando que ele deve confirmar a transação.

Depois de confirmar a transação, o usuário é levado para a tela inicial onde é recarregado todas as informações do contrato, e o novo candidato adicionado é exibido na tela.

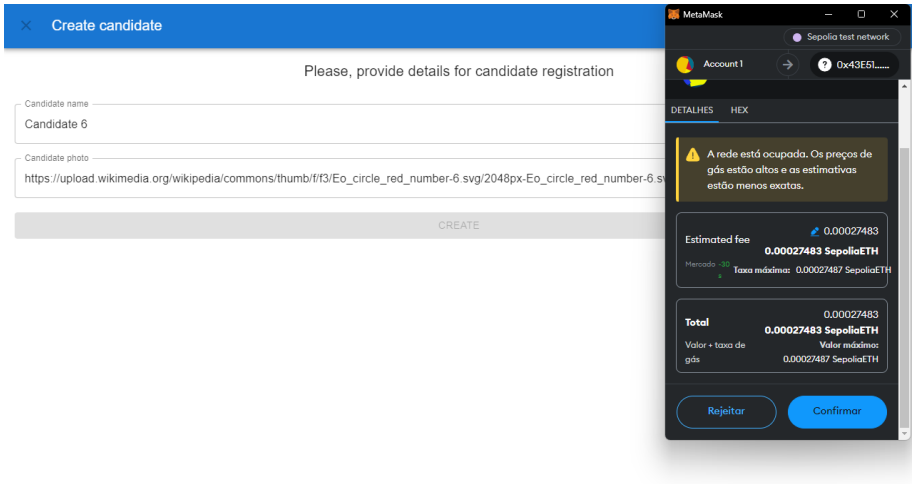


Figura 4.5: Tela de cadastro de candidato com caixa de confirmação de transação da Metamask

Após todos os usuários realizarem seus votos e o dono do contrato finalizar a votação, basta acessar a tela inicial da votação e visualizar o candidato com o maior número de votos. Outra forma de verificar o vencedor da votação é acessando o contrato inteligente e visualizando os eventos de voto e realizando a contagem dos votos.

4.4 Verificação do voto

Se tratando de um sistema de votação utilizando a *blockchain* pública da *Sepolia*, é possível confirmar o seu voto. Essa ação permite que o voto realizado seja auditável. Para tanto, basta acessar o Explorador da *blockchain* da *Sepolia* e acessar a página referente à sua

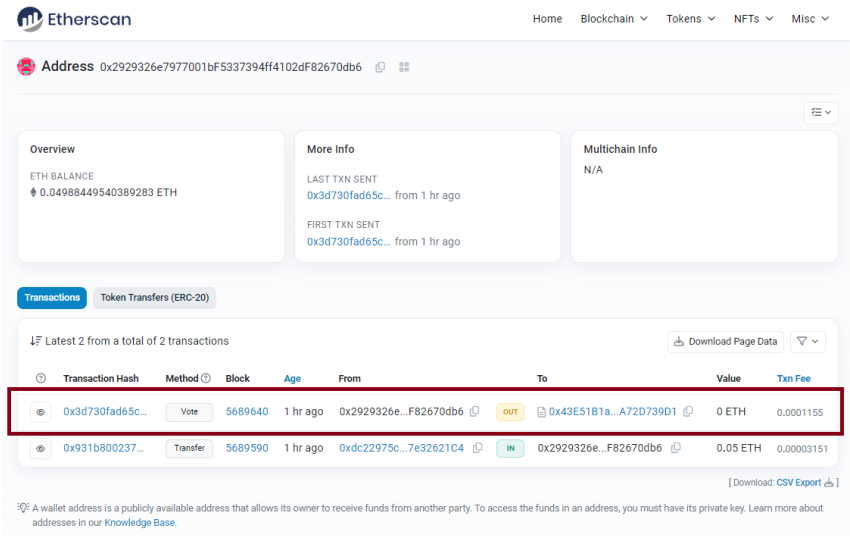


Figura 4.6: Transações do usuário 0x2929326e7977001bF5337394ff4102dF82670db6 na *blockchain* da *Sepolia*

conta, para isso, copie seu endereço publico e clique na busca.

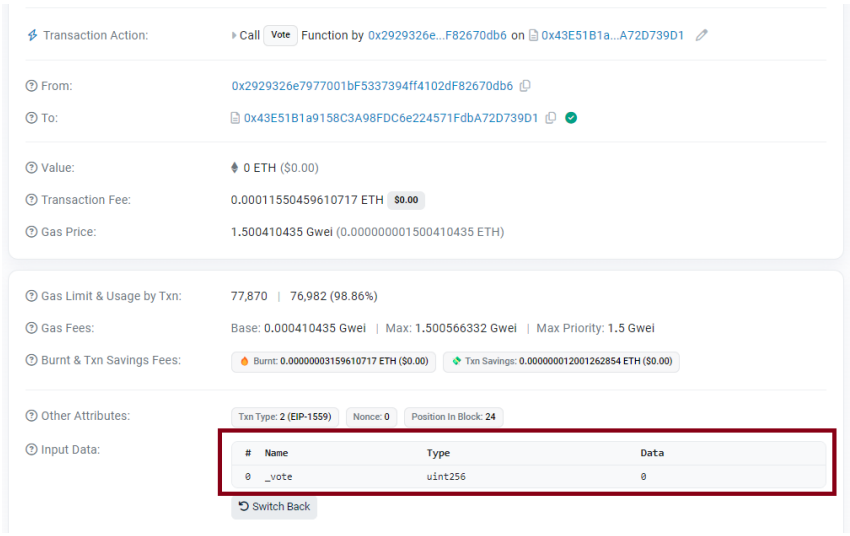


Figura 4.7: Transação de voto realizada

Após acessar, todas as transações realizadas pelo usuário serão exibidas. Procure na lista a transação que foi enviada pelo usuário para o código do contrato e que tenha utilizado o método de voto. Ao encontrar essa transação, clique nela para abrir os detalhes da mesma, onde será possível verificar no campo *Input Data* se o valor enviado corresponde ao candidato desejado, neste caso, 0 (identificador do Candidato 1). Desta maneira, qualquer usuário do sistema poderá validar o seu voto utilizando o seu endereço público.

5 Conclusões e Trabalhos Futuros

Com este trabalho foi possível verificar que a *blockchain* é uma tecnologia que possui características interessantes para utilização em sistemas de *e-voting*, trazendo mais transparência, confiança na votação e a oportunidade do votante auditar o seu próprio voto.

Foi desenvolvido um sistema de votação com regras bem definidas e com código fonte totalmente aberto. Além disso, o sistema é auditável, uma vez que se apoia em uma *blockchain* pública onde qualquer usuário tem acesso às informações. Sendo assim, o sistema desenvolvido representa uma evolução aos meios de votação tradicionais, trazendo como principais destaques a possibilidade e facilidade de validação do voto pelos usuários/eleitores. Isso torna o sistema de votação mais confiável e mais transparente.

O sistema demonstra versatilidade em diversas áreas de aplicação. Um dos exemplos são as votações internas do ICE da UFJF, que atualmente são realizadas pelo sistema Integra. Até mesmo em votações realizadas em pesquisas públicas pelo Senado.

5.1 Discussões

Alguns pontos de discussão importantes que essa pesquisa trouxe são com relação aos custos de implementação do sistema proposto. Visto que foi utilizada uma *blockchain* pública, todas as transações que alteram o estado da *blockchain* como votar, ou cadastrar candidato, por exemplo, possuem um custo de *Gas*. Esse é um dos desafios enfrentados se fosse adotada uma *blockchain* da *Ethereum*, visto que o preço do *Gas* está elevado devido a muitas transações que acontecem na rede.

Outro desafio é com relação à adaptação dos atuais sistemas de votação para a utilização da *blockchain*. Será preciso criar um mecanismo para que os usuários tenham uma conta na *blockchain* utilizada para realizar o voto. Além disso, esses usuários deverão ter *Ether* em suas contas para que possam pagar a taxa de rede (*Gas*) para realizar o voto.

Outro desafio diz respeito à privacidade das informações, especialmente em con-

textos de votação onde é essencial garantir o anonimato dos eleitores. Este trabalho não aborda essa questão e o sistema proposto não assegura o anonimato dos votantes.

5.2 Trabalhos futuros

No contexto das oportunidades futuras, há diversas possibilidades para o trabalho desenvolvido. Por exemplo, é possível adicionar regras para o contrato inteligente e molda-lo à votação de acordo com o desejado de forma que o usuário informe personalize o contrato e sua forma de aplicação. Sendo assim, é possível propor um mecanismo onde o usuário que deseja iniciar uma votação defina as suas regras através de mecanismos de interfaces, criando o contrato inteligente para a sua votação com base nas regras escolhidas previamente.

É possível também realizar estudos de aplicabilidade para utilização da tecnologia *blockchain* em eleições oficiais no Brasil. Esse inclusive, é um tema que está sendo estudado pelo TSE (Tribunal Superior Eleitoral) (EXAME, 2020). Ou ainda, de maneira mais simples, aplicá-lo como uma prova de conceito no âmbito de um instituto ou unidade acadêmica.

Por fim, outras tecnologias de redes *blockchain* podem ser avaliadas, assim como seu desempenho e custo associados para a aplicação.

Bibliografia

B3. *B3 inicia negociação do primeiro ETF da Hashdex*. [S.l.], 2021. Disponível em: ⟨https://www.b3.com.br/pt_br/noticias/b3-inicia-negociacao-do-primeiro-etf-da-hashdex.htm⟩. Acesso em: 10 de Junho, 2024.

BARCELOS, F. A.; MARTINS, J. V. de L. *Aplicabilidade da blockchain no sistema de eleição departamental da UTFPR campus Ponta Grossa*. [S.l.], 2020.

COINMARKETCAP. *Coinmarketcap*. [S.l.], 2024. Disponível em: ⟨https://coinmarketcap.com/⟩. Acesso em: 27 de Abril, 2024.

COINTELEGRAPH. *Capitalização do mercado de criptomoedas ultrapassa US$2 trilhões com preço do Bitcoin desafiando US$ 57.000*. [S.l.], 2024. Disponível em: ⟨https://br.cointelegraph.com/news/crypto-recaptures-2-trillion-market-cap-bitcoin-climbs-above-57000⟩. Acesso em: 10 de Junho, 2024.

CUNHA S. S. D.; MARCACINI, A. T. R. C. M. A. F. C. T. S. J. R. P. A. D. d. B. F. A. M. F. V. d. C. M. A. M. d. T. M. C. *Relatório sobre o Sistema Brasileiro de Votação Eletrônica*. [S.l.], 2010. Disponível em: ⟨https://www.brunazo.eng.br/voto-e/textos/CMind-1-Brasil-2010.pdf⟩. Acesso em: 10 de Junho, 2024.

DIGICONOMIST. *Bitcoin Energy Consumption Index*. [S.l.], 2022. Disponível em: ⟨https://digiconomist.net/bitcoin-energy-consumption⟩. Acesso em: 10 de Junho, 2024.

ETHEREUM. *Ethereum-powered tools and services*. [S.l.], 2022. Disponível em: ⟨https://ethereum.org/pt/dapps/⟩. Acesso em: 10 de Junho, 2024.

ETHEREUM. *TOKEN STANDARDS*. [S.l.], 2023. Disponível em: ⟨https://ethereum.org/pt/developers/docs/standards/tokens/⟩. Acesso em: 10 de Junho, 2024.

ETHEREUM. *PoS*. [S.l.], 2024. Disponível em: ⟨https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/⟩. Acesso em: 25 de Junho, 2024.

ETHERSCAN. *The Ethereum Blockchain Explorer*. [S.l.], 2022. Disponível em: ⟨https://etherscan.io/⟩. Acesso em: 10 de Junho, 2024.

EXAME. *TSE testa projetos para 'eleições do futuro' baseados em blockchain*. [S.l.], 2020. Disponível em: ⟨https://exame.com/future-of-money/blockchain-e-dlts/tse-testa-projetos-para-eleicoes-do-futuro-baseados-em-blockchain/⟩. Acesso em: 10 de Junho, 2024.

EXAME. *Datafolha: índice de confiança das urnas é de 73[S.l.], 2022. Disponível em: ⟨https://extra.globo.com/noticias/brasil/datafolha-indice-de-confianca-das-urnas-de-73-25518899.html⟩. Acesso em: 10 de Junho, 2024.*

GOOGLE. *Preço Bitcoin*. [S.l.], 2024. Disponível em: ⟨https://www.google.com/search?q=preco+bitcoin⟩. Acesso em: 27 de Abril, 2024.

JESUS, J. B. A. de; GONCALVES, J. P. de B. Aplicação para votação utilizando a tecnologia blockchain. p. 1–4, 2018. Disponível em: ⟨http://anais.eati.info:8080/index.php/2019/article/view/288/300⟩. Acesso em: 10 de Junho, 2024.

MACELAI, V. Verificação de eleição utilizando blockchain. p. 1–81, 2019. Disponível em: ⟨https://bibliotecadigital.tse.jus.br/xmlui/bitstream/handle/bdtse/6869/2019_macelai_verificacao_eleicao_blockchain.pdf?sequence=1&isAllowed=y⟩. Acesso em: 10 de Junho, 2024.

MARTINHO, L. L.; NETO, F. C. J. Sistema de eleições desenvolvido com a tecnologia de contratos inteligentes baseado em blockchain. p. 1–16, 2019.

MERCADO BITCOIN. *Ethereum 2.0 ou Merge: o que é, data de lançamento e mudanças*. [S.l.], 2024. Disponível em: ⟨https://blog.mercadobitcoin.com.br/ethereum-2-0⟩. Acesso em: 10 de Junho, 2024.

NAKAMOTO, S. *Bitcoin: A peer-to-peer electronic cash system*. [S.l.], 2008. Disponível em: ⟨https://bitcoin.org/bitcoin.pdf⟩. Acesso em: 10 de Junho, 2024.

NIWA, H. Um sistema de voto eletrônico baseado em blockchain. 2019.

NUBANK. *Nubank lança solução para comprar e vender criptomoedas para todos os clientes*. [S.l.], 2022. Disponível em: ⟨https://blog.nubank.com.br/nubank-lanca-nubank-cripto-para-todos-clientes/⟩. Acesso em: 10 de Junho, 2024.

OPENSEA. *Explore, collect, and sell NFTs*. [S.l.], 2022. Disponível em: ⟨https://opensea.io/⟩. Acesso em: 10 de Junho, 2024.

REMIX PROJECT. *DEPLOY RUN TRANSACTIONS IN THE BLOCKCHAIN*. [S.l.], 2024. Disponível em: ⟨https://remix-project.org/⟩. Acesso em: 10 de Junho, 2024.

RUEDIGER, M. A. Desinformação on-line e eleições no brasil: a circulação de links sobre desconfiança no sistema eleitoral brasileiro no facebook e no youtube (2014-2020). 2020.

SOLIDITY. *Solidity Documentation*. [S.l.], 2022. Disponível em: ⟨https://docs.soliditylang.org/en/v0.8.17/⟩. Acesso em: 10 de Junho, 2024.

SOLORIO KEVIN; KANNA, R. H. D. H. *Hands-on Smart Contract Development with Solidity and Ethereum: From Fundamentals to Deployment*. Sebastopol, Califórnia, EUA: O'Reilly Media, 2019. 1-267 p., 7 x 0.57 x 9.19 inches. Bibliografia: p. 131–132. ISBN 978-1492045267.

SOUZA, T. M. P. de. Votechain, uma solução mais segura, acessível e inovadora para as eleições, implementada com a tecnologia blockchain. 2019.

TETHER. *Tether*. [S.l.], 2024. Disponível em: ⟨https://tether.to/⟩. Acesso em: 10 de Junho, 2024.