UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

IESLEY BEZERRA DOS SANTOS

UM SISTEMA DE BILHETES BASEADO EM BLOCKCHAIN

QUIXADÁ

2023

IESLEY BEZERRA DOS SANTOS

UM SISTEMA DE BILHETES BASEADO EM BLOCKCHAIN

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Dr. Emanuel Ferreira Coutinho

QUIXADÁ

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S235s Santos, Iesley Bezerra dos.
Um sistema de bilhetes baseado em blockchain / Iesley Bezerra dos Santos. – 2023.
62 f.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2023.
Orientação: Prof. Dr. Emanuel Ferreira Coutinho.

1. Contrato inteligente. 2. Ethereum. 3. Blockchains (Base de dados). I. Título.

CDD 004

IESLEY BEZERRA DOS SANTOS

UM SISTEMA DE BILHETES BASEADO EM BLOCKCHAIN

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em: __/__/____.

BANCA EXAMINADORA

_____
Prof. Dr. Emanuel Ferreira Coutinho (Orientador)
Universidade Federal do Ceará (UFC)

_____
Prof. Dr. Michel Sales Bonfim
Universidade Federal do Ceará (UFC)

_____
Prof. Dr. Leonardo Oliveira Moreira
Universidade Federal do Ceará (UFC)

A minha família pelo apoio e o incentivo a minha formação superior.

AGRADECIMENTOS

A minha família pelo apoio e o incentivo a minha formação superior.

"O verdadeiro alívio é a paz que sentimos quando finalmente concluímos algo que nos preocupava. É como se um peso fosse retirado dos nossos ombros, e podemos finalmente respirar livremente." - Maya Angelou

RESUMO

A facilidade do meio digital economiza tempo, especialmente na compra de ingressos para eventos. Antes era necessário ir aos pontos de venda e enfrentar filas, mas agora é possível comprar ingressos online com apenas alguns cliques. No entanto, é importante mencionar os cambistas, que revendem ingressos com lucro. Eles utilizam diversos meios, como adquirir bilhetes em bilheterias e contar com uma rede de contatos para obter ingressos mais baratos em quantidades maiores. Além disso, a falsificação de ingressos é um problema real e que persiste com a venda digital de ingressos. Como uma forma de dificultar a atuação de cambistas e evitar a venda de ingressos falsificados, este trabalho propõe um sistema de comercialização de bilhetes usando tecnologia blockchain e que possibilita aos organizadores de eventos limitar ou proibir a revenda de ingressos por parte dos compradores, garantindo assim a disponibilidade dos bilhetes para o público-alvo. Além disso, foram implementados contratos inteligentes que regulam a revenda de ingressos, proporcionando maior controle e transparência nas transações. Para isso, a aplicação utilizou os contratos inteligentes na ethereum. Foi realizado um teste de facilidade de uso no sistema implementado, e que constatou que o sistema é fácil de usar.

Palavras-chave: Contrato inteligente; Ethereum; Blockchains (Base de dados).

ABSTRACT

The ease of the digital medium saves time, especially when purchasing event tickets. Before it was necessary to go to the points of sale and face queues, but now it is possible to buy tickets online with just a few clicks. However, it is important to mention the exchangers, who resell tickets for a profit. They use different means, such as purchasing tickets at ticket offices and relying on a network of contacts to obtain cheaper tickets in larger quantities. Additionally, ticket counterfeiting is a real and persistent problem with digital ticket sales. As a way to make it difficult for scalpers to act and to avoid the sale of counterfeit tickets, this work proposes a ticket sales system using blockchain technology that allows event organizers to limit or prohibit the resale of tickets by buyers, thus guaranteeing the availability of tickets to the target audience. In addition, smart contracts were implemented to regulate the resale of tickets, providing greater control and transparency in transactions. For this, the application used smart contracts on ethereum. An ease of use test was carried out on the implemented system, and found that the system is easy to use.

Keywords: Smart contract; Ethereum; Blockchains (Database).

LISTA DE QUADROS

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

As facilidades que o meio digital nos oferece nos ajudam também a poupar tempo. A compra de ingressos para eventos é um grande exemplo de como a internet facilita nossa vida, pois uma atividade que antes demandava locomoção aos pontos de venda e espera em filas, hoje só demanda alguns cliques, graças a internet (JUNIOR, 2013).

A comercialização de ingressos por meios online proporciona aos usuários conforto e comodidade, pois a compra pode ser feita através de qualquer dispositivo com acesso a internet e que suporte a aplicação que comercializa o ingresso, geralmente sites web ou aplicações mobile (JUNIOR, 2013). O usuário pode comprar de qualquer lugar, desde que esteja conectado. Além das vantagens já mencionadas, o comerciante tem mais facilidade na hora de aplicar descontos ou promoções especiais, pois a identificação do comprador é facilitada e em tempo real, além de ter uma economia com os custos de comercialização visto que não precisará manter pontos físicos de venda, o que no fim pode significar em menos custos ao comprador.

Ao falar de venda de ingressos, não podemos deixar de mencionar os cambistas, que basicamente são profissionais informais que atuam na porta de eventos que demandam entrada por meio de ingresso. Seu objetivo é lucrar com a revenda dos ingressos, onde eles vendem por um preço maior pelo que pagaram (RIBEIRO, 2014).

Para adquirir os ingressos, os cambistas usam de vários meios. Tradicionalmente, eles adquirem em bilheterias, onde se passam por compradores comuns e ficam repetindo o ato de comprar por várias vezes, até atingir seu objetivo de compra ou o limite possível. Comumente os cambistas possuem uma rede de contatos que os ajudam a conseguir bilhetes mais baratos e em quantidades acima dos limites por pessoa (RIBEIRO, 2014).

De acordo com as estatísticas da AARP (American Association of Retired Persons), mais de 5 milhões de pessoas nos EUA por ano, já tiveram a experiência de comprar ingressos falsificados (CHA et al., 2018).

Os ingressos atuais tem como identificadores códigos de barras ou QR codes. Tais identificadores acarretam riscos aos usuários, pois qualquer registro em imagem de tais identificadores abre a possibilidade destes serem extraídos, e replicados em outros ingressos falsos. Além disso, a revenda por parte de compradores que precisam se desfazer do ingresso, é dificultada, pois não tem como validar se o ingresso já foi usado ou revendido (TACKMANN, 2017).

Blockchain é um termo que surgiu em 2008, junto com a publicação da

criptomoeda bitcoin (NAKAMOTO, 2008). O blockchain da bitcoin trata-se de um livro contábil público e distribuído onde são registradas as transações na ordem em que ocorreram, formando um histórico que resulta na ordem atual do livro onde cada nó da rede possui uma cópia do livro contábil.

Com a tecnologia blockchain podemos armazenar informações sobre eventos e ingressos em uma rede blockchain. Assim, pessoas devidamente autorizadas podem usar as informações contidas na rede e verificar as propriedades dos ingressos com a garantia da integridade das informações que são garantidas pela blockchain.

Este trabalho propõe um sistema de bilhetes, implementado em blockchain com o ethereum, tendo como público alvo os compradores de ingressos online. Um sistema de bilhetes em blockchain traz para os compradores de ingressos a segurança na integridade das informações e a possibilidade de averiguar o real estado do ingresso comprado, evitando ingressos duplicados. Na solução proposta, o estado dos bilhetes é armazenado na blockchain, e só podem ser alterados por quem tiver autorização.

1.1 Objetivos

1.1.1 Objetivo Geral

Desenvolver um sistema de comercialização de ingressos que faça uso da tecnologia blockchain para garantir a integridade dos dados e a descentralização da aplicação.

1.1.2 Objetivos Específicos

- Desenvolver uma solução arquitetural que permita ao organizador do evento limitar ou proibir a revenda de ingressos por parte dos compradores de ingressos.
- Implementar contratos inteligentes que limitem ou proíbam a revenda de ingressos por parte dos compradores de ingressos.
- Avaliar a facilidade de uso da aplicação.

2 TRABALHOS RELACIONADOS

Nesta seção são apresentados os trabalhos relacionados à pesquisa. Foram selecionados três trabalhos que abordam o problema dos bilhetes digitais, e implementam uma solução usando blockchain.

2.1 *Secure Event Tickets on a Blockchain*

O artigo de Tackmann (2017) tem como objetivo propor uma solução para a comercialização de ingressos, que faça o uso da tecnologia blockchain para alcançar a conveniência de ingressos padrão com segurança aprimorada.

A solução desenvolvida trabalha com três agentes: os vendedores de ingressos, os clientes e os organizadores de eventos. A ideia central da solução é armazenar os identificadores únicos (IDs) de ingressos, juntamente com a identidade criptográfica do atual proprietário no blockchain. Cada transação pode gerar um novo bilhete, transferir um bilhete para um novo proprietário, ou invalidar um bilhete, quando o atual proprietário do bilhete decide usar o ingresso para entrar no local do evento. Todas as ações mudam o estado do bilhete na blockchain.

A solução foi desenvolvida tendo o Hyperledger Fabric V1 como sua plataforma de blockchain. As assinaturas digitais são ECDSA com curva secp256. O *chaincode* é escrito em Go usando a ligação padrão do Fabric e o armazenamento de valor-chave LevelDB fornecido.

O aplicativo usado por clientes e organizadores é programado em Swift e funciona em dispositivos iOS. A transmissão de dados nas etapas de venda e verificação são implementadas por meio da geração e leitura de códigos QR. Um comprador de bilhete simplesmente apresenta um código QR que contém a chave pública de assinatura pkb para o vendedor; o vendedor a escaneia e gera a solicitação de venda.

A solução usa assinaturas digitais para proteger todas as transações e permite que os usuários gerenciem, vendam e usem os ingressos com um aplicativo em seu *smartphone*. A segurança do sistema se baseia em dois pilares principais: a consistência garantida da blockchain e a validação das assinaturas digitais. Em resumo, a consistência do blockchain garante que cada bilhete apenas faça transições de estado válido e a validação das assinaturas digitais, por sua vez, garante que as solicitações enviadas para a blockchain só possam ser

geradas pela parte relevante.

2.2 *A Smart Contract-Based Mobile Ticketing System with Multi-Signature and Blockchain*

O artigo de Lin et al. (2019) propõe o uso da tecnologia blockchain para desenvolver um sistema descentralizado de bilhetagem móvel usando a auto-aplicação de contratos inteligentes em blockchain para garantir a execução correta das transações. A solução usa um mecanismo de múltiplas assinaturas para garantir a autenticidade e propriedade dos bilhetes, e para autorização das transferências de dinheiro envolvidas nas transações.

O sistema proposto funciona da forma descrita a seguir. O organizador do evento cria um evento como um contrato inteligente na blockchain para vender ingressos. O vendedor obtém ingressos do organizador do evento com um plano de vendas escrito na forma de um contrato inteligente. O organizador do evento e o agente assinam o contrato para acionar a execução do plano de vendas. O plano de vendas descreve os preços de venda e outros detalhes. Os bilhetes são emitidos na forma de código QR. Os contratos inteligentes garantem a inclusão da assinatura digital do organizador do evento nos ingressos emitidos para evitar os ingressos falsos. No entanto, nesta fase, o tíquete do código QR é apenas parcial. O tíquete de código QR ainda precisa da assinatura digital do consumidor. Durante a passagem pelo portão, o código QR é assinado pelo consumidor, e isso acionará o evento de entrar na bilheteria e transferir o dinheiro por meio dos contratos inteligentes.

O sistema proposto é implementado em EOSIO, que é uma rede de blockchain com permissão usando a prova delegada of-stake (DPoS) com consenso de tolerância a falhas bizantino.

2.3 *A Blockchain-based Privacy Preserving Ticketing Service*

O artigo de Cha et al. (2018) propõe um serviço de venda de bilhetes baseado em blockchain com preservação da privacidade, abreviado para BB Tickets. O estudo projeta e implementa o serviço BB Tickets com a blockchain Ethereum e usa tecnologias de contrato inteligente para construir os principais componentes do serviço.

No serviço BB Tickets, os organizadores de eventos podem colocar informações do evento na rede blockchain. Portanto, um usuário pode navegar pelos eventos disponíveis e

solicitar a compra de ingressos para um evento específico de um vendedor de ingressos.

Para proteger a privacidade do usuário, o referido estudo utiliza o esquema *Non Interactive Zero-Knowledge* (NIZK), que permite aos usuários provar que são compradores de ingressos, sem revelar suas informações identificáveis.

O serviço BB Tickets define dois grandes contratos inteligentes. O primeiro é o contrato inteligente *Ticket Issuer*, que fornece informações de um evento e trata dos pedidos de ingresso, compra e validação sobre o evento. O segundo, é o contrato inteligente do ingresso, que armazena as informações do ingresso e é obtido depois que um usuário compra um ingresso.

2.4 Análise Comparativa

Em relação ao trabalho de Cha et al, (2018), este trabalho permite que os criadores de ingresso limitem a revenda de ingressos, além de prover um qr code único por ingresso, usando assinatura digital. Já em relação a Lin et al. (2019), este trabalho desenvolve um sistema que permite aos organizadores de evento limitar a revenda de ingressos por parte dos compradores. Por fim, em relação ao trabalho de Tackmann (2017), este trabalho desenvolve um sistema que permite aos organizadores de eventos limitar a revenda de ingressos por parte dos compradores.

Os principais critérios de comparação entre os artigos foram:

- Limite na revenda de ingressos.
- Uso seguro do *QR Code* para representar ingresso.

A seguir apresentamos o Quadro 1, que resume as análises feitas nesta seção entre os trabalhos selecionados.

Quadro 1 - Análise comparativa entre trabalhos relacionados e este trabalho

Trabalho	Limita a Revenda de ingressos	QR Code
Blockchain-based Privacy Preserving Ticketing Service	Não	Não
A Smart Contract-Based Mobile Ticketing System with Multi-Signature and Blockchain	Não	Sim

Secure Event Tickets on a Blockchain	Não	Sim
Este Trabalho	Sim	Sim

Fonte: Fornecido pelo Autor.

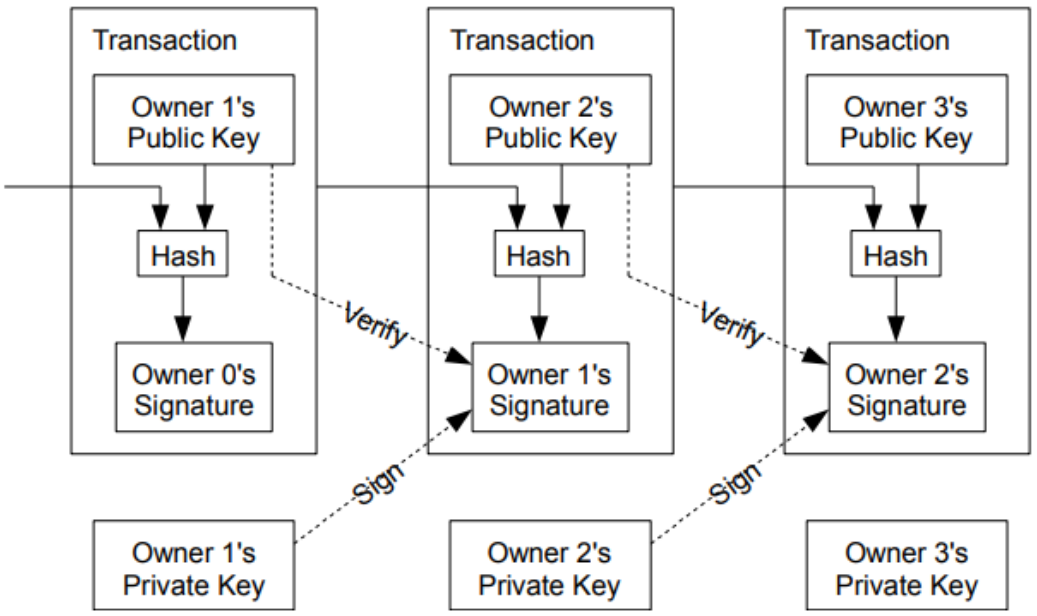
3 FUNDAMENTAÇÃO TEÓRICA

Esta seção trata da fundamentação teórica deste trabalho. São abordados os conceitos de blockchain no geral, e também é falado sobre criptografia.

3.1 Blockchain

Com o surgimento do bitcoin em 2008, a tecnologia blockchain que foi usada no seu desenvolvimento, ficou bastante conhecida. A blockchain consiste em uma cadeia de blocos, onde cada bloco, com exceção do primeiro e do último, faz referência a outro bloco (NAKAMOTO, 2008). Essa cadeia é publicada e distribuída para todos os nós que compõem a rede. Cada bloco contém registros seguros e imutáveis de transações que ocorreram conforme pode ser visto na figura 1. A cadeia cresce à medida que novas transações vão surgindo e vão sendo registradas.

Figura 1 - Esquema ilustrativo de blocos encadeados

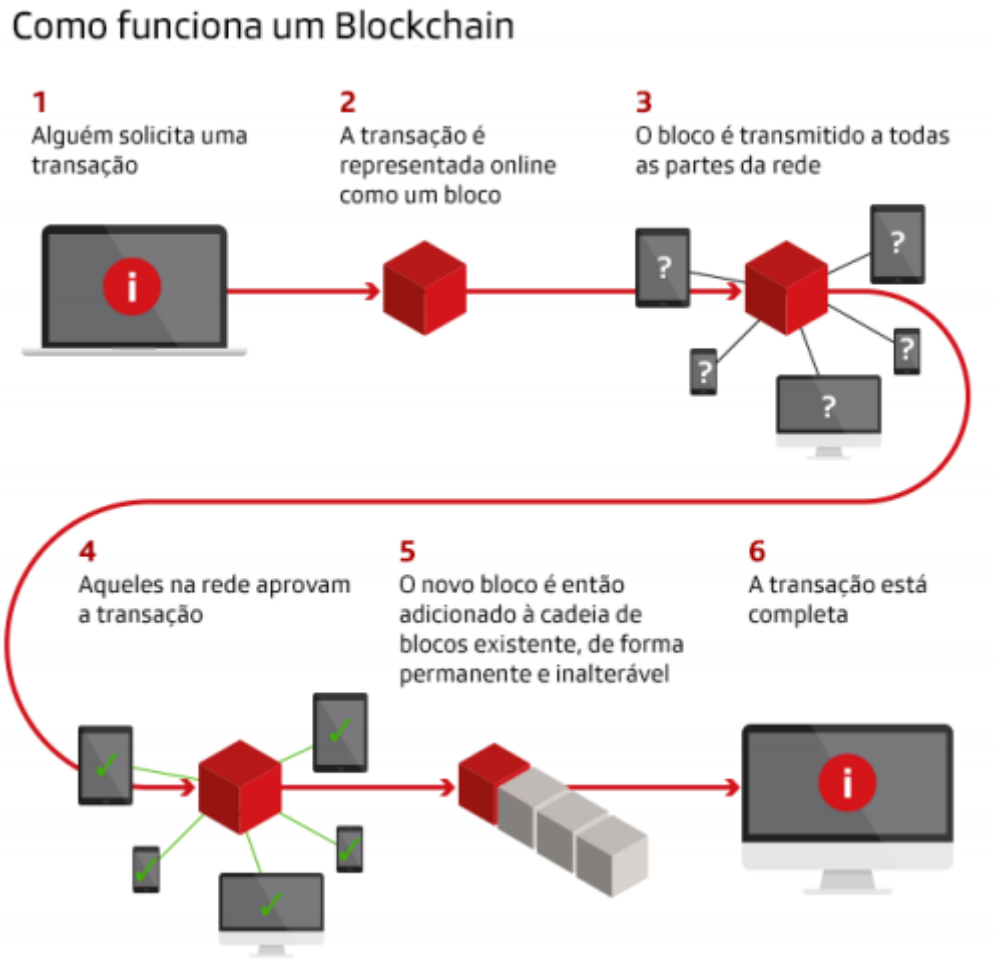


Fonte: (NAKAMOTO, 2008)

Na figura 2 temos a representação com alto nível de abstração do funcionamento de uma blockchain, desde a solicitação de uma transação até seu término. No passo 1 alguém

solicita uma transação; no passo 2, essa transação passa a ser representada como um bloco; no passo 3, esse bloco é disseminado para todos os nós da rede; no passo 4, os nós da rede aprovam (ou reprovam) a transação; no passo 5 caso tenha sido aprovado, o novo bloco é adicionado na blockchain de forma permanente e imutável e no passo 6 a transação é concluída.

Figura 2 - Esquema ilustrativo do funcionamento de uma blockchain



Fonte: Financial Times, PwC Estados Unidos

Cada blockchain tem o seu contexto e no caso do bitcoin, os blocos são criados através do processo de mineração por blocos denominados mineradores, que competem entre si e através do algoritmo de prova de consenso (PoW) é determinado quais nos mineradores terão suas transações publicadas e serão recompensados com bitcoin. Após a criação do bloco ele é disseminado pela rede e só é aceito e adicionado caso seja aprovado por mais da metade da rede.

O algoritmo de prova de consenso PoW (*Proof Of Work*) é baseado em dois princípios: a prova de trabalho tem que ser difícil e trabalhosa, mas não impossível, e a verificação da prova de trabalho deve ser fácil e rápida (NAKAMOTO, 2008). No bitcoin os mineradores são os responsáveis por realizarem a prova de trabalho para validarem as transações. O minerador que realizar mais rápido a prova de trabalho ganha uma certa quantidade de bitcoins.

No caso do bitcoin, a prova de trabalho envolve a verificação de um valor que, quando encriptado por exemplo em SHA-256, o *hash* começa com um certo número de bits zero. O trabalho médio necessário é exponencial em número de zero bits necessários e podem ser verificados executando um único *hash*. Para se realizar a mineração do bloco, busca um valor nonce que forneça um hash do bloco com o número de 0s iniciais determinados.

A complexidade e o custo computacional do problema a ser resolvido na prova de trabalho, é o que garante a segurança e a imutabilidade dos dados, pois como cada bloco faz referência ao anterior, uma alteração em um bloco anterior exige uma alteração em todos os blocos posteriores para se ter uma cadeia válida (onde todos os blocos tem um certa quantidade de 0s iniciais).

3.2 Protocolos de Consenso

Protocolos de consenso já são utilizados em sistemas distribuídos para a coordenação de processos independentes que precisam concordar sobre um mesmo valor, e no desenvolvimento de protocolos de comunicação (RIBEIRO; MENDIZABAL, 2019).

Em blockchain, os protocolos de consenso são utilizados com o intuito de garantir decisões descentralizadas e uniformes sobre o bloco que deve ser adicionado à cadeia. Existem diferentes abordagens, mas as principais são:

- Qualquer nó pode propor um bloco e o protocolo se encarregar de escolher qual será adicionado.
- O protocolo define quais nós podem propor blocos.

Em blockchains públicas, as decisões sobre escolhas dos nós que vão entrar na rede não pode ser restrita a um pequeno grupo de nós, visto que dessa forma o princípio da descentralização seria violado.

3.2.1 Prova de Participação

Prova de participação é um protocolo que surgiu como uma alternativa ao protocolo prova de trabalho, e que exige menos esforço computacional e deixa o processo de consenso mais rápido e barato (KING; NADAL, 2012).

Neste protocolo, novos blocos só podem ser adicionados a cadeia ou validados, pelos blocos validadores. Os blocos validadores são escolhidos através de um processo aleatório, mas para ser elegível a validador, um bloco precisa realizar uma transação especial que consiste em enviar um valor x de unidade monetária, que ficará retido e servirá como prova de participação. Os validadores são registrados na cadeia.

Em alguns sistemas o valor enviado aumenta as chances de escolha, mas mesmo assim o protocolo não fica vulnerável visto que um usuário malicioso para assegurar o direito de propor blocos teria que dispor de uma quantidade impraticável de fundos.

Os principais algoritmos de consenso que se relacionam com prova de participação são *chain-based proof of stake* e *BFT-style proof of stake*.

3.2.2 Prova de tempo decorrido

Prova de tempo decorrido (POeT) é um algoritmo de consenso utilizado em blockchains privadas e permissivas, que apresenta um menor consumo de energia em relação aos algoritmos mais comuns e não demanda hardware especializado porém seu funcionamento é atrelado a instruções de processadores (OLSON et al, 2018).

A ideia do algoritmo é a seguinte: cada bloco validador solicita um tempo de espera, esse tempo é gerado aleatoriamente; a uma função atrelada a blockchain chamada de enclave. O nó que recebe o menor tempo é eleito o líder e pode gerar um bloco, além disso lhe é atribuído um certificado, e após o bloco gerado ser validado pela rede este é adicionado a cadeia.

3.2.3 Tolerância prática a falhas bizantinas

Tolerância prática a falhas bizantinas é um algoritmo capaz de tolerar falhas bizantinas. No contexto das blockchains, ter um protocolo de consenso com tolerância a falhas bizantinas é muito importante devido à sua capacidade de se obter consenso em meio a nós maliciosos. Porém esse protocolo só apresenta bom desempenho para um número

reduzido de nós, sendo interessante para uso em blockchains privadas (VUKOLIĆ, 2016).

3.2.4 Prova de Trabalho

O Algoritmo de Prova de Trabalho é o protocolo de consenso que foi adotado na rede do bitcoin. Neste algoritmos, tem-se os mineradores que são os nós que competem entre si para ver quem resolve mais rápido o desafio criptográfico proposto, aquele que ganha tem o direito de produzir um novo bloco que caso seja aceito pela rede, o nó minerador vencedor será recompensado com uma certa quantia de criptomoedas (NAKAMOTO, 2008).

O desafio criptográfico consiste em encontrar o valor nonce que gera um código hash das transações e demais informações do bloco, que contenha uma certa quantidade de 0s iniciais, essa quantidade de 0s é a dificuldade da rede. Caso mais de um nó consiga resolver o desafio ao mesmo tempo, o bloco adicionado a rede é o que tem a cadeia mais longa, e consequentemente o que teve mais esforço computacional envolvido.

3.3 Tipos de Blockchain

Apesar de parecer uma coisa só, existem diferentes tipos de blockchains que podem ser usadas para diferentes situações, cada uma com capacidades e características únicas (GUEGAN, 2017). As blockchains existentes pertencem a um dos seguintes tipos: pública, privada ou permissionada e híbridas.

3.3.1 Blockchain Pública

Foi o primeiro tipo de blockchain que surgiu. São publicamente acessíveis pela internet e tem o Bitcoin e a Ethereum como as mais conhecidas (GUEGAN, 2017). Apesar de ter seus dados e seu código disponíveis publicamente, elas são bem seguras. Tem como principais características:

- Permitir que qualquer pessoa faça parte.
- Ter uma rede que funcione de forma transparente e aberta.
- Descentralizada.

3.3.2 Blockchain Permissionada

Com o desenvolvimento e crescimento do blockchain, surgiu o interesse empresarial pela tecnologia, o que levou ao desenvolvimento de blockchains privadas, onde se tem uma unidade central que concede acesso à rede (LAI; CHUEN, 2018). Os principais exemplos são o Hyperledger da Foundation Linux, e o Quorum da JPMorgan. Esse tipo de blockchain tem como principais características:

- O livro de transações e demais informações geradas pela blockchain é privado.
- Muitas vezes não possuem criptomoedas ou mineração.

3.3.3 Blockchain Híbrida

A blockchain híbrida é uma fusão entre a pública e a privada, juntando o melhor dos dois mundos. Na blockchain híbrida o acesso aos recursos da rede é controlado por uma ou mais entidades como na Permissionada, mas o acesso ao livro razão é público (LAI; CHUEN, 2018). Tem como principais características:

- O acesso à rede se dá por meio da autorização das unidades de controle.
- As informações geradas pela blockchain são públicas.

3.4 Plataformas Blockchain

3.4.1 Ethereum

Ethereum é uma plataforma de blockchain pública que foi pioneira nas plataformas que permitem o desenvolvimento de contratos inteligentes. Nela os contratos inteligentes são desenvolvidos em solidity (WOOD, 2015).

Por ser pública, tantos os contratos inteligentes como os dados armazenados são públicos. A plataforma ainda tem uma criptomoeda própria chamada de ether, que é usada para cobrar pelas transações e execução de contratos inteligentes na rede.

3.4.2 Hyperledger

Hyperledger é uma plataforma de blockchain permissionada que permite o desenvolvimento de contratos inteligentes em linguagens de programação comuns como Node JS, Java e Go (ANDROULAKI et al, 2018). Por ser permissionada , seus contratos e dados se encontram públicos somente aos participantes da rede.

3.5 Criptografia de Chave Assimétrica

Durante muito tempo, o ramo da criptografia viu o surgimento de diversos algoritmos que apesar das diferenças na forma como realizavam a criptografia dos dados, usavam uma única chave para realizar os processos de encriptação e desencriptação. O uso de uma única chave acarretava desvantagens como o fato de o destinatário de uma dada mensagem encriptada já ter de saber de antemão a chave para poder descriptografar e entender a mensagem (STALLINGS, 2015, p. 201-206).

A criptografia de chave assimétrica, introduziu o uso de duas chaves no processo criptográfico, onde uma pode ser usada para encriptar e outra para desencriptar, e consequentemente uma das chaves pode ser pública sem afetar a confidencialidade da outra chave.

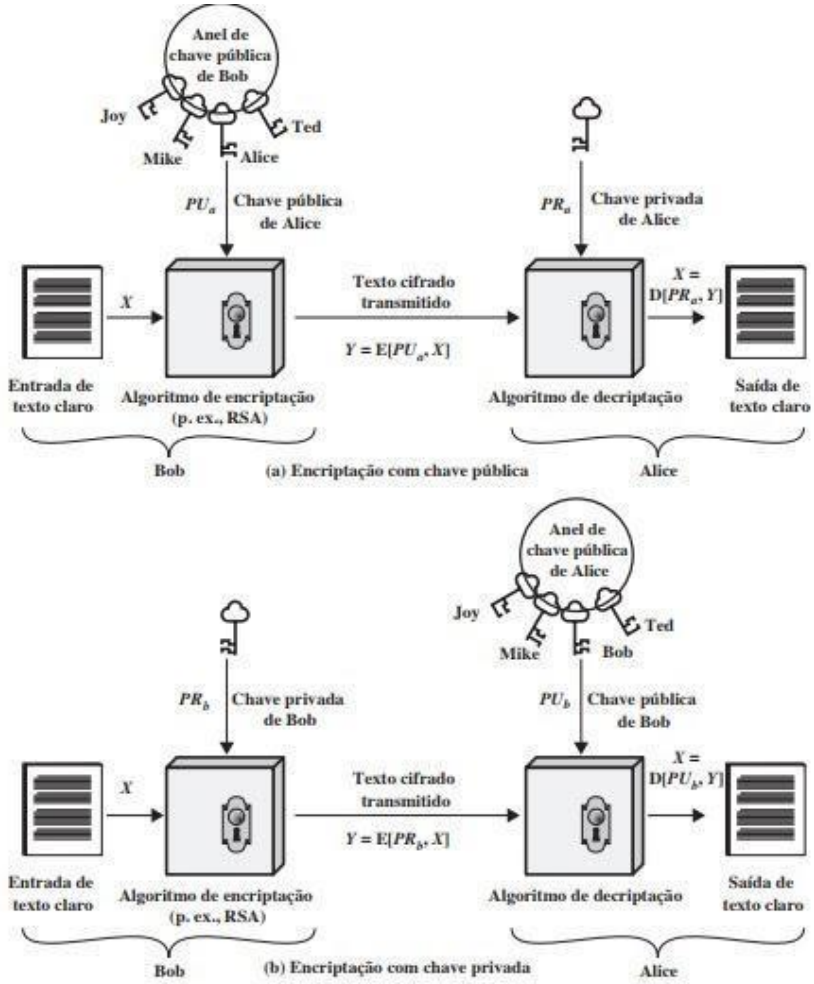
Os algoritmos de chave assimétrica, contam com duas chaves, uma chave para encriptação e uma chave para decriptação. No geral os algoritmos de criptografia de chave pública, tornam inviável computacionalmente a determinação de uma chave, mesmo que se tenha a outra chave e o conhecimento do algoritmo. Além disso, algoritmos como o RSA, permitem o uso de qualquer uma das duas chaves para a encriptação, desde que a outra seja usada para decriptação.

Na figura 3 temos um esquema de criptografia de uma mensagem com uso de chaves públicas e privadas e um algoritmo de chave assimétrica. As etapas essenciais são as seguintes:

1. Cada usuário gera um par de chaves.
2. Cada usuário torna umas das chaves públicas, compartilhando-a com os demais, e guarda a outra somente para si, sendo esta a chave privada.
3. Se o Bob, quer enviar uma mensagem para Alice e garantir que somente ela possa lê-la, ele encripta a mensagem com a chave pública de Alice.

4. Quando Alice recebe a mensagem, ela a decripta com sua chave privada, ou seja, a mensagem só pode ser lida por quem tiver a chave privada de Alice, no caso somente ela.
5. Se Bob quiser enviar uma mensagem para Alice, e assegurar que ele é o autor da mensagem, ele encripta a mensagem com sua chave privada.
6. Quando Alice recebe a mensagem, ela a decripta com a chave pública do Bob, e garante que a mensagem foi enviada pelo Bob, pois uma mensagem encriptada com uma chave privada só pode ser decriptada com a chave pública correspondente.

Figura 3 - Esquema de criptografia de mensagem, com uso de chave pública e privada



Fonte: William Stallings (2015, p. 202)

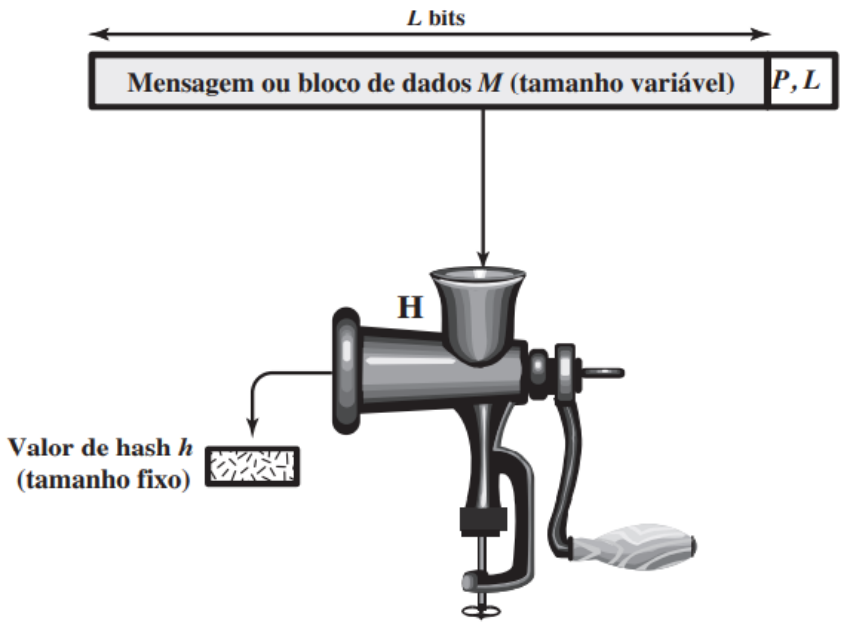
3.6 Função *Hash*

Função *hash* é o nome dado aos algoritmos que mapeiam dados de tamanho variável para dados de tamanho fixo. O valor retornado pelas funções *hash*, são chamados de valores *hash*, somas *hash*, códigos *hashes* ou simplesmente *hashes* (STALLINGS, 2015, p. 247-250).

Uma função *hash*, pode ser descrita como h = H(m), onde m são dados de tamanho variável, h são dados de tamanho fixo e H é a função *hash* que realiza a transformação. O principal objetivo de uma função *hash* é garantir a integridade dos dados, de modo que boas funções *hash* tem como propriedade o fato de produzirem saídas igualmente distribuídas e aleatórias para um grande conjunto de dados de entrada.

As funções *hash* criptográficas são uma aplicação das funções *hash* no campo da segurança da informação. As *hash* criptográficas visam tornar inviável computacionalmente que se consiga descobrir (1) dois objetos de dados que quando aplicados a uma função *hash* resultem no mesmo código *hash* (propriedade livre de colisão), ou (2) um objeto de dados que quando aplicado a uma função *hash*, resulte em um código *hash* pré determinado (propriedade da mão única). A figura 4 ilustra o que se espera na entrada e na saída de uma função *hash*.

Figura 4 - Esquema ilustrativo de aplicação de função *hash* a um bloco de dados M



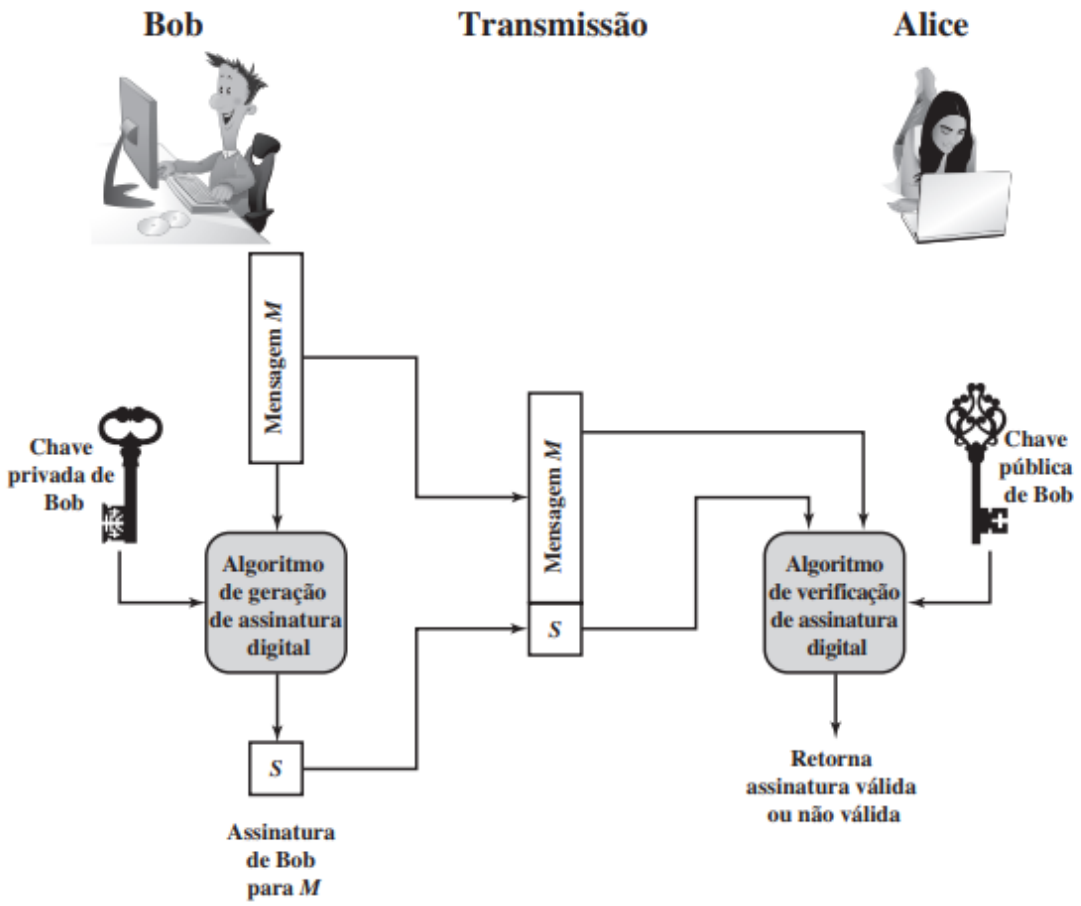
P, L = preenchimento mais campo de tamanho

Fonte: William Stallings (2015, p. 247)

3.7 Assinatura Digital

A assinatura digital é uma forma de autenticação digital que serve para atestar que uma dada mensagem foi criada por determinada pessoa. A criptografia de chave pública foi o avanço criptográfico que possibilitou o surgimento do esquema de assinatura digital, que faz uso da chave privada de um usuário para autenticar uma mensagem (STALLINGS, 2015, p. 310-312).

Figura 5 - Diagrama de esquema de assinatura digital simples



Fonte: William Stallings (2015, p. 310)

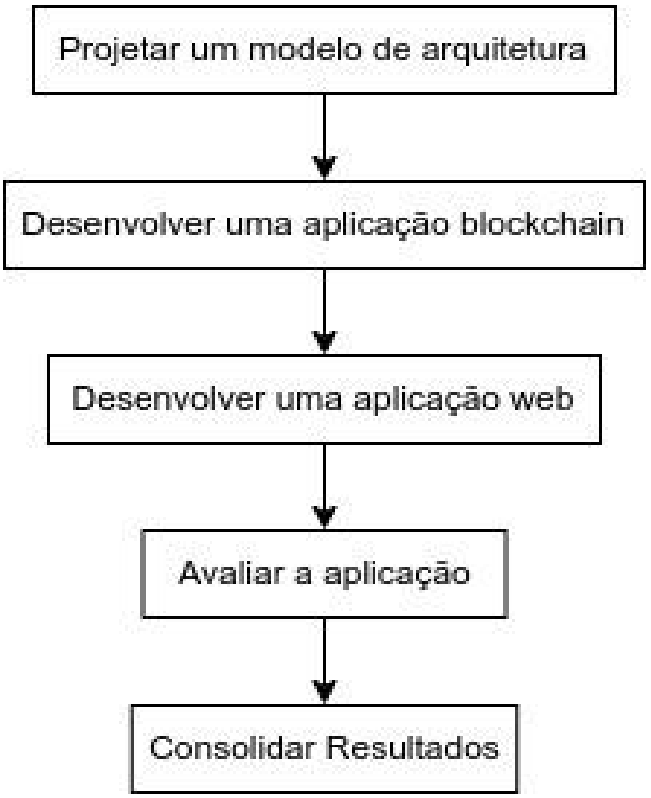
A figura 5 apresenta um modelo simples de criação e uso de assinaturas digitais.

Bob assina uma mensagem usando sua chave privada, para isso ele entra com a chave e a mensagem em um algoritmo de geração de assinatura digital. Alice pode verificar se a mensagem é assinada por Bob, usando sua chave pública e a mensagem como entradas em um algoritmo de verificação de assinatura digital.

4 PROCEDIMENTOS METODOLÓGICOS

O trabalho foi dividido em algumas etapas com o intuito de cumprir os objetivos. As etapas iniciais estão relacionadas com a melhor compreensão do tema e dos conceitos abordados, e são: levantamento bibliográfico, estudo de trabalhos relacionados e uma análise de ferramentas e tecnologias. Na figura 6 podemos ver as etapas em uma figura ilustrativa.

Figura 6 - Fluxo das Atividades da Metodologia



Fonte: Elaborado pelo Autor

As demais etapas estão diretamente relacionadas com o desenvolvimento da solução proposta para os problemas levantados na introdução, e são:

4.1 Projetar um modelo de arquitetura

É necessário se ter uma definição de como será o fluxo dos dados e qual o papel que cada camada da solução terá. Dessa forma, foi implementado um modelo de arquitetura

que engloba a camada blockchain e a aplicação *web*; a fim de se organizar as interações entre entidades, o fluxo dos dados e quais dados serão disponibilizados na blockchain. Com isso tivemos uma estrutura base para o desenvolvimento da aplicação blockchain.

4.2 Desenvolver uma aplicação blockchain

Após o desenvolvimento da arquitetura, foi iniciado o desenvolvimento do contrato inteligente na blockchain. O contrato inteligente é o que garante que a comercialização de ingressos ocorra conforme o previsto, além de persistirem na blockchain, as informações relacionadas.

4.3 Desenvolver uma aplicação web

Uma vez criado o contrato inteligente que persiste e realiza as regras de negócios em uma blockchain, foi desenvolvida uma aplicação web que se comunica com o contrato inteligente e é uma parte fundamental na lógica da aplicação como um todo, pois através do aplicativo web que o usuário final valida o seu ingresso digital e assim tem acesso ao evento em questão.

4.4 Avaliar a aplicação

Após o desenvolvimento da aplicação e com uma versão funcional, a aplicação foi avaliada através de um evento fictício, onde a aplicação foi utilizada para aquisição e uso de ingressos para o evento.

A aplicação foi testada por um pequeno grupo de pessoas que através da aplicação *web* interagiram com a blockchain. Os participantes tiveram acesso a uma lista de funcionalidades a serem avaliadas. Após o teste, os participantes preencheram um questionário. E avaliaram o funcionamento da aplicação como um todo.

4.5 Consolidar Resultados

Em posse do resultado dos questionários, foram produzidos gráficos que mostram e comparam os resultados obtidos. Além disso, os resultados foram analisados de forma quantitativa e qualitativa com base nos valores percentuais, absolutos, nas médias, e nas análises dos textos das respostas.

5 DESCRIÇÃO GERAL DO SISTEMA

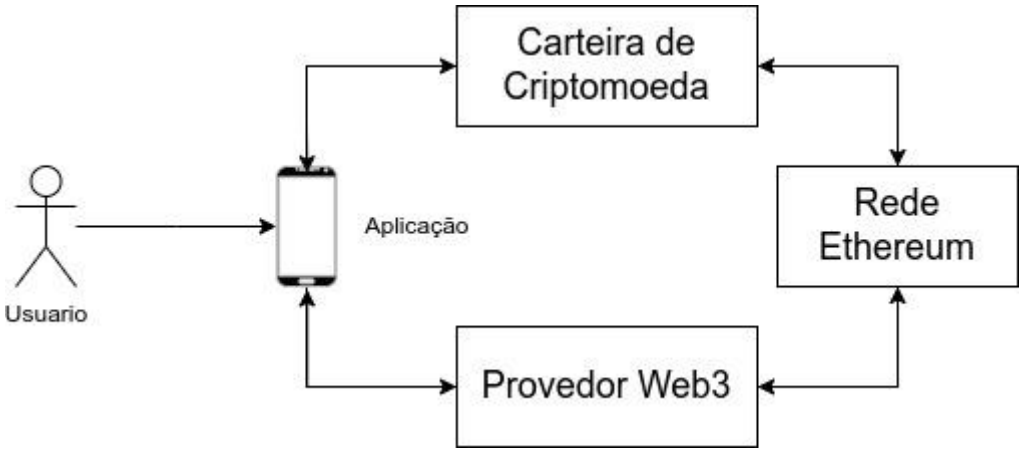
Nesta seção será descrita uma visão geral do sistema, a arquitetura e as funcionalidades implementadas.

5.1 Objetivo

A aplicação descrita neste artigo, tem como objetivo ser uma plataforma para a comercialização de ingresso, de uma forma segura, moderna e descentralizada, fazendo uso da tecnologia blockchain por meio de contratos inteligentes.

5.2 Arquitetura

Figura 7 - Fluxo de arquitetura



Fonte: Elaborado pelo autor

Como mostra a Figura 7, podemos destacar 3 componentes principais que fazem a aplicação funcionar, são eles: Carteira de Criptomoedas, que é responsável por identificar o usuário em questão, assinar transações e enviá-las para uma rede ethereum; Provedor Web3, seria uma aplicação de terceiros, que fornece acesso a um nó da rede, possibilitando a leitura de informações do contrato inteligente; Rede Ethereum, seria alguma das redes da ethereum, onde ficam os dados e onde as transações são validadas.

5.3 Principais Funcionalidades

Esta seção descreve as principais funcionalidades da aplicação, informando como elas podem ser feitas e fala um pouco sobre o uso que faz do contrato inteligente e sobre algumas outras tecnologias usadas. As telas da aplicação estão no apêndice B, e apesar de estarem em um formato mobile, trata-se das fotos de uma aplicação web sendo executada em um navegador da carteira de criptomoedas, em um celular

5.3.1 Conectar Carteira

Para usar as demais funcionalidades do sistema, o usuário deve conectar a sua carteira de criptomoedas ao aplicativo. Para conectar a carteira, basta clicar no botão com o nome “CONECTAR CARTEIRA”. Ao se conectar, desde que se tenha a extensão da carteira no navegador, ela abrirá para que o usuário confirme na carteira a conexão com a aplicação. Após a conexão, o nome do botão exibirá o endereço da conta conectada e com essa informação o aplicativo poderá executar as demais funcionalidades.

5.3.2 Criar Ingressos

Essa funcionalidade gera uma transação que altera os dados do contrato inteligente e é enviada para a rede por meio da carteira conectada. Para criar ingressos, basta clicar no card expansivo com o nome “Criar Ingressos”, que será exibido um formulário com as seguintes informações para serem preenchidas:

- Nome do Evento;
- Valor do Ingresso em ether;
- Limite de Transferência de cada ingresso;
- Quantidade de ingressos a serem criados;

Após preencher as informações, basta clicar no botão com o nome “CRIAR INGRESSO”, que a aplicação vai criar e enviar uma solicitação de transação para a carteira conectada, que exibirá uma página com o valor atual do gás e solicitará permissão para confirmar a transação. Com a transação confirmada, será exibido na aplicação um modal com o status atual da transação, e que informará se a transação foi concluída com sucesso ou não.

5.3.3 Listar Eventos

Essa funcionalidade apenas lê os dados no contrato inteligente sem alterá-los; os dados são obtidos por meio do provedor web3, que no caso deste aplicativo foi o provedor *Alchemy*. Para listar eventos, basta clicar no card expansivo com o nome “Listar Eventos”, que será exibido em uma tabela os eventos disponíveis com as seguintes informações:

- ID do evento;
- Nome do evento;
- Endereço da conta do organizador;
- Valor dos ingressos em ether;
- Quantidade de ingressos com o organizador e que estão disponíveis;

5.3.4 Comprar Ingresso

Essa funcionalidade gera uma transação que altera os dados do contrato inteligente e é enviada para a rede por meio da carteira conectada. A função executada no contrato inteligente, realiza uma transferência para a conta do proprietário do ingresso antes da transação. A compra de ingresso pode ser feita na aplicação, através de um formulário dentro do card expansível com nome “Comprar Ingresso” ou através do botão “COMPRAR” que aparece junto de cada evento na parte de “Listar Eventos”, e que ao ser clicado, direciona o usuário para o formulário e preenche os dados automaticamente. No formulário, serão solicitadas as seguintes informações:

- ID do evento;
- Endereço da conta do proprietário do ingresso;
- Valor do ingresso em ether;
- Habilitar ou não o ingresso para venda, após a compra;

Após preencher as informações, basta clicar no botão com o nome “Comprar Ingresso”, que a aplicação vai criar e enviar uma solicitação de transação para a carteira conectada, que exibirá uma página com o valor atual do gás e o valor que será enviado para o contrato inteligente (referente ao valor do ingresso) e solicitará permissão para confirmar a transação. Com a transação confirmada, será exibido na aplicação um modal com o status

atual da transação, e que informará se a transação foi concluída com sucesso ou não.

5.3.5 Listar meus ingressos

Essa funcionalidade apenas lê os dados no contrato inteligente sem alterá-los; os dados são obtidos por meio do provedor web3, que no caso deste aplicativo foi o provedor *Alchemy.* Para o usuário ter acesso aos ingressos em sua posse, basta clicar no card expansível com nome “Meus Ingressos”, e clicar no botão “LISTAR MEUS INGRESSOS” que será listado os ingressos em sua posse, com as seguintes informações:

- ID do ingresso;
- ID do evento;
- Nome do evento;
- Endereço da conta do organizador;
- Valor do ingresso em ether;
- Número de transferências do ingresso;
- Limite de transferências do ingresso;
- Disponibilidade para venda;

Também é possível filtrar a listagem somente para ingressos possíveis de serem transferidos ou não, para isso, basta clicar no botão “TRANSFERIVEIS” até ficar verde, para filtrar os disponíveis para transferência; ou vermelho para filtrar os não disponíveis para transferência. Também é possível filtrar a listagem somente para ingressos disponíveis para a venda ou não, para isso, basta clicar no botão “A VENDA” até ficar verde, para filtrar os disponíveis para venda; ou vermelho para filtrar os não disponíveis para venda.

5.3.6 Exibir Qr Code do Ingresso

Essa funcionalidade não realiza qualquer interação com a rede ethereum, apenas usa a carteira conectada para assinar a mensagem que faz referência ao ingresso. Para exibir o qr code de um ingresso, basta acessar a tabela com a listagem dos ingressos em posse do usuário. Na coluna “QrCode” terá o botão “ABRIR”, que ao ser clicado, vai abrir a carteira com uma solicitação de assinatura, essa assinatura é o conteúdo do qr code e faz referência ao id do ingresso. Ao confirmar a assinatura, será exibido em um modal o qr code. O qr code foi

gerado pela aplicação web, usando a biblioteca “react-qr-code”.

5.3.7 Validar o qr code

Essa funcionalidade gera uma transação que altera os dados do contrato inteligente e é enviada para a rede por meio da carteira conectada. Para validar o qr code de um ingresso, basta clicar no card expansivo com o nome “Validar Ingresso” e clicar no botão “VALIDAR INGRESSO”, que será exibido um modal que solicitará permissão para acessar a câmera, e ao ser concedida; vai abrir dentro do modal, a visualização da leitura do qr code. Quando o qr code for lido, vai aparecer no modal o botão “VERIFICAR” que ao ser clicado, exibe a carteira para confirmar o envio da transação, que ao ser confirmada é enviada e o seu status é exibido no modal, que atualiza para sucesso ou erro.

5.4 Tecnologias usadas

Para o desenvolvimento do sistema web, foi utilizado o framework react, além de html, javascript e css para cuidar da parte visual. Para a conexão com a carteira de criptomoedas foi utilizado a biblioteca web3.js. Para gerar o qr code foi utilizada a biblioteca react-qr-code, e para a leitura do qr code, foi utilizada a biblioteca react-qr-reader.

6 CONTRATO INTELIGENTE

Nesta seção, apresentaremos o desenvolvimento de um contrato inteligente denominado "TicketContract" que visa facilitar a venda e validação de ingressos para eventos. O contrato foi implementado em Solidity, uma linguagem de programação utilizada para escrever contratos inteligentes na plataforma Ethereum.

6.1 Funcionalidades do Contrato

O contrato inteligente "TicketContract" possui as seguintes funcionalidades:

- 1. Criação de Ingressos: O contrato permite que qualquer pessoa crie ingressos para um determinado evento. O criador especifica o nome do evento, a quantidade de ingressos a serem criados, o valor de cada ingresso e um limite de transferências. Os ingressos são armazenados em uma estrutura de dados chamada "Ticket", cada um com id exclusivo, e com o mesmo id de evento caso sejam criados juntos. É permitido criar até 1000 ingressos de uma única vez.

- 2. Compra de Ingresso: O contrato permite que qualquer pessoa possa comprar ingressos para um evento específico, bastando fornecer o ID do evento, o endereço da conta do proprietário do ingresso e um valor booleano para indicar se após a compra, o ingresso estará disponível para venda. O contrato verifica se ainda há ingresso do evento e do proprietário especificado, disponível para a venda e se podem ser transacionados. Além disso, o comprador deve enviar o valor do ingresso em wei para o contrato inteligente. Após a compra bem-sucedida, o ingresso é transferido para o endereço do comprador, e o valor do ingresso, descontada a taxa do contrato(um valor fixo em 140000000000000 wei); é transferida para o antigo proprietário do ingresso.

- 3. Listagem de Ingressos por Proprietário: O contrato tem uma função chamada “filterTicketsByOwner” que recebe e filtra os ingressos pelos seguintes parâmetros: id do evento, endereço da conta do proprietário, disponibilidade para transferência, habilitado para venda.

- 4. Listagem de Grupos de Ingressos: O contrato tem uma função chamada “getGroupTickets” que agrupa os ingressos transferíveis e disponíveis para venda, cujo proprietário é o criador; e retorna essas informações em uma estrutura de dados chamada

“GroupTickets”.

- 5. Verificar Ingresso: O contrato tem uma função chamada “verifyTicket” que recebe como parâmetros: o id do ingresso, o endereço da conta do proprietário atual do ingresso, e informações que compõem uma assinatura digital no seguinte formato “ticketId={id do ingresso}”, feita pelo proprietário do ingresso. A função verifica se as informações que compõem a assinatura digital estão condizentes, se a assinatura foi gerada pelo proprietário do ingresso, se o id de ingresso informado, se refere a um ingresso existente, cujo proprietário é o informado e cujo criador é a conta que chamou o contrato inteligente; se tudo estiver de acordo a informação referente ao ingresso é removida do contrato inteligente e a execução é concluída com sucesso, do contrário será revertida.

6.2 Desenvolvimento do Contrato Inteligente

O contrato inteligente “TicketContract” foi desenvolvido usando a linguagem Solidity para rodar na Ethereum Virtual Machine. Foi utilizado também o Remix, uma IDE online que serviu para editar o código, compilar, implantar e testar em diversas redes. O código completo do contrato inteligente, pode ser visto no apêndice A.

7 AVALIAÇÃO DOS RESULTADOS

Nesta seção serão detalhados os testes que foram realizados com usuários e os resultados obtidos.

7.1 Perfil dos Candidatos

Participaram da pesquisa 6 pessoas, em sua maioria com idade entre 18 e 25 anos, onde metade delas eram da área de tecnologia da informação e conhecia blockchain. A grande maioria das pessoas nunca tinham tido acesso a alguma carteira de criptomoedas. Todos os participantes que responderam a pesquisa, concordaram com o termo de consentimento livre e esclarecido.

7.2 Preparação

A estrutura necessária para a realização dos testes foi de um dispositivo móvel com a carteira de criptomoedas Metamask instalada com acesso à internet estável, e um computador com navegador com a extensão do metamask instalada, além de duas contas criadas no metamask e na rede Sepolia Test Network, com saldo de pelo menos 0.1 SepoliaETH e o link da aplicação. As atividades envolveram os fluxos das principais funcionalidades do aplicativo, onde foram descritos em formas de cenários que simulavam situações reais. A meta buscada com os testes foi a de avaliar o correto funcionamento da aplicação, sua integração com a carteira de criptomoedas e sua facilidade de uso.

7.2.1 Atividade para conectar a carteira com a aplicação

O cenário descrito foi de que o usuário deseja conectar sua carteira de criptomoedas com a aplicação de comercialização de ingressos. O objetivo dessa atividade é verificar se o usuário consegue estabelecer a conexão da carteira com a aplicação de forma correta e segura.

7.2.2 Atividade para criar ingressos

O cenário descrito foi de que um usuário deseja criar ingressos para um evento

específico. O objetivo dessa atividade é verificar se o sistema permite ao usuário criar ingressos, inserindo as informações corretas.

7.2.3 Atividade para comprar ingressos

O cenário descrito foi de que um usuário deseja comprar ingressos para um determinado evento e de um determinado proprietário. O objetivo dessa atividade é verificar se o sistema permite ao usuário visualizar os ingressos disponíveis para o evento e o proprietário específicos, e efetuar a compra utilizando sua carteira de criptomoedas.

7.2.4 Atividade para gerar qr code do ingresso

O cenário descrito foi de que um usuário deseja exibir o qr code de um ingresso em específico. O objetivo dessa atividade é verificar se o sistema permite ao usuário gerar o qr code do ingresso.

7.2.5 Atividade para verificar o ingresso

O cenário descrito foi de que o usuário é um criador de ingressos e deseja verificar o qr code de outro usuário. O objetivo dessa atividade é verificar se o sistema permite ao usuário criador, realizar a verificação com sucesso.

7.3 Execução

Os participantes estavam cientes das atividades que realizaram e tiveram livre acesso a aplicação. As atividades que seriam executadas estavam com a devida descrição e orientação no formulário de pesquisa, e foi solicitado que avaliassem a dificuldade de para realizar cada atividade. Além disso, os participantes puderam emitir seu feedback sobre a aplicação.

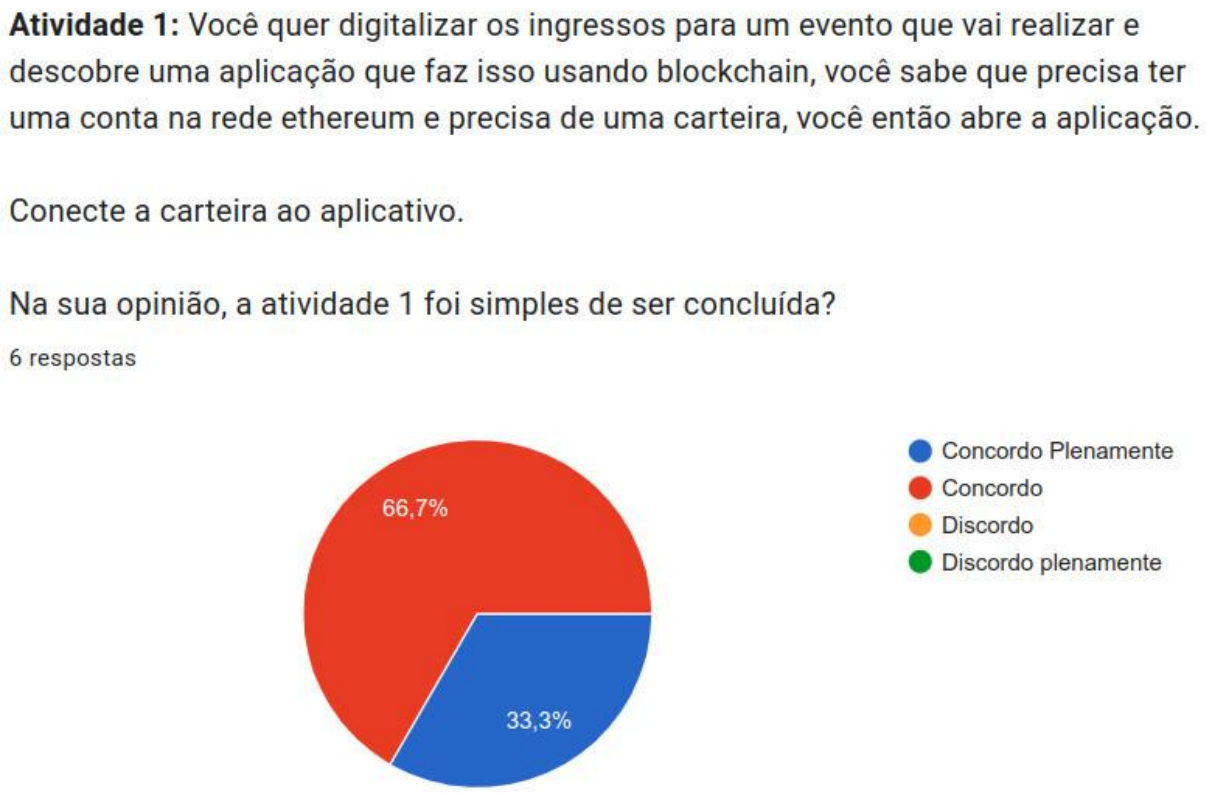
7.4 Análise dos Resultados

Nesta seção serão apresentados os resultados e análises de cada atividade do teste de facilidade de uso de forma isolada.

7.4.1 Análise da atividade para conectar a carteira com a aplicação

Nesta seção, foi analisado se o usuário conseguiria conectar a carteira com a aplicação. A resposta do teste foi positiva, com todos os participantes respondendo com "Concordo" ou "Concordo Plenamente" que conseguiram executar a atividade com facilidade, como mostrado na figura 8.

Figura 8 - Resultado da atividade de conectar carteira ao aplicativo

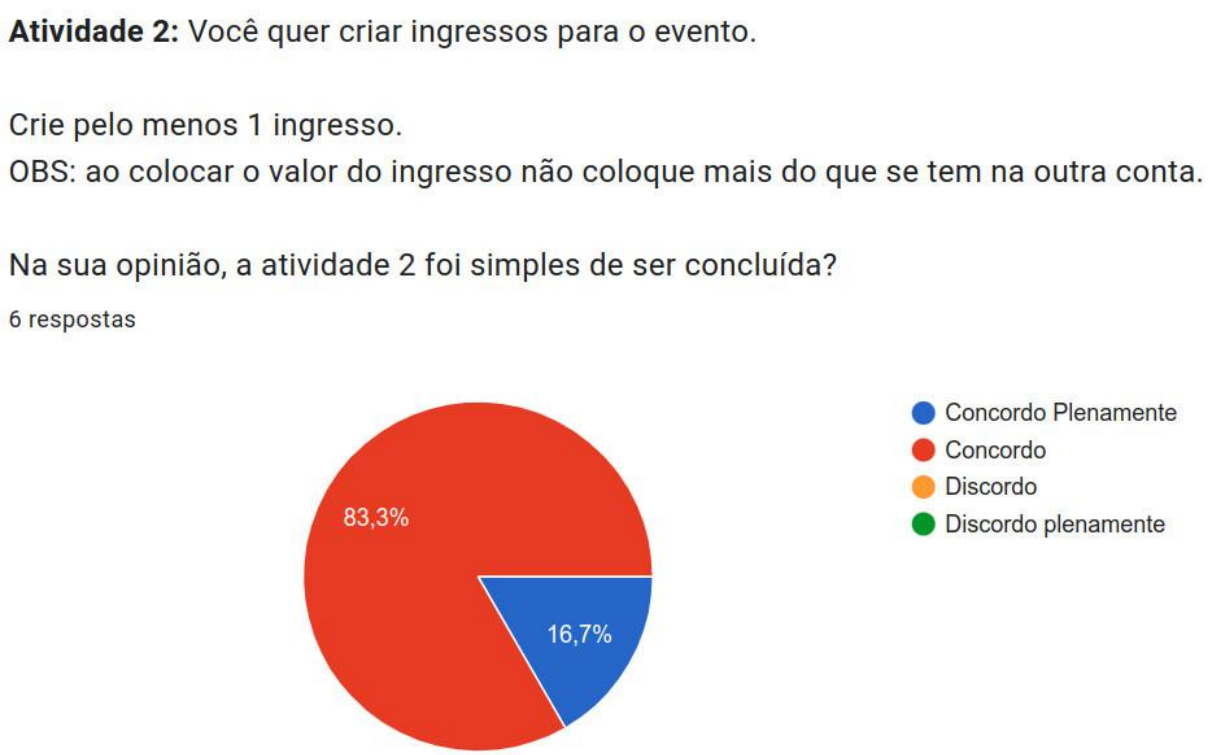


Fonte: Elaborado pelo autor

7.4.2 Análise da atividade para comprar ingressos

Nesta seção, foi analisado se o usuário conseguiria criar ingressos.. A resposta do teste foi positiva, com todos os participantes respondendo com "Concordo" ou "Concordo Plenamente" que conseguiram executar a atividade com facilidade, como mostrado na figura 9.

Figura 9 - Resultado da atividade de criar ingresso

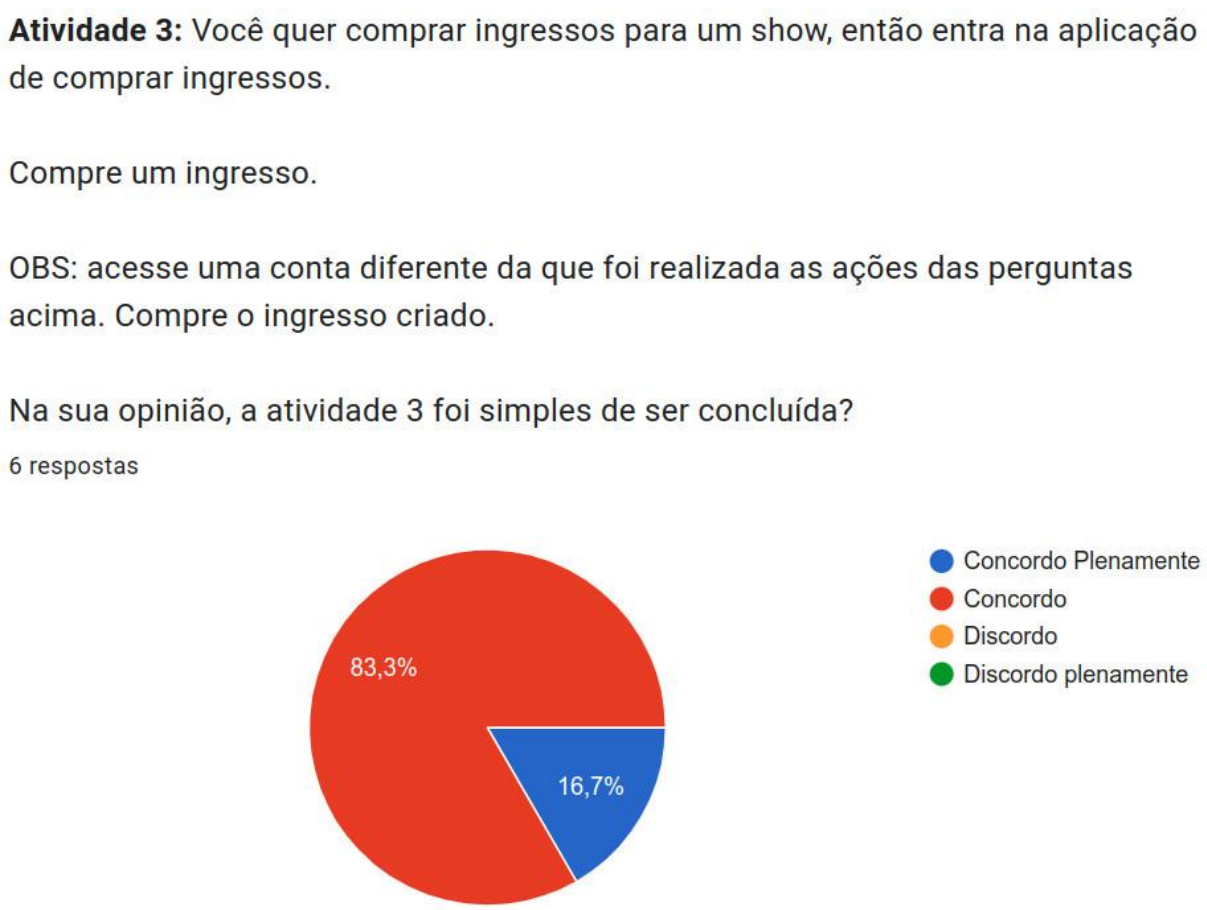


Fonte: Elaborado pelo autor

7.4.3 Análise da atividade para gerar qr code do ingresso

Nesta seção, foi analisado se o usuário conseguiria comprar ingressos. A resposta do teste foi positiva, com todos os participantes respondendo com “Concordo” ou “Concordo Plenamente” que conseguiram executar a atividade com facilidade, como mostrado na figura 10.

Figura 10 - Resultado da atividade de comprar ingresso

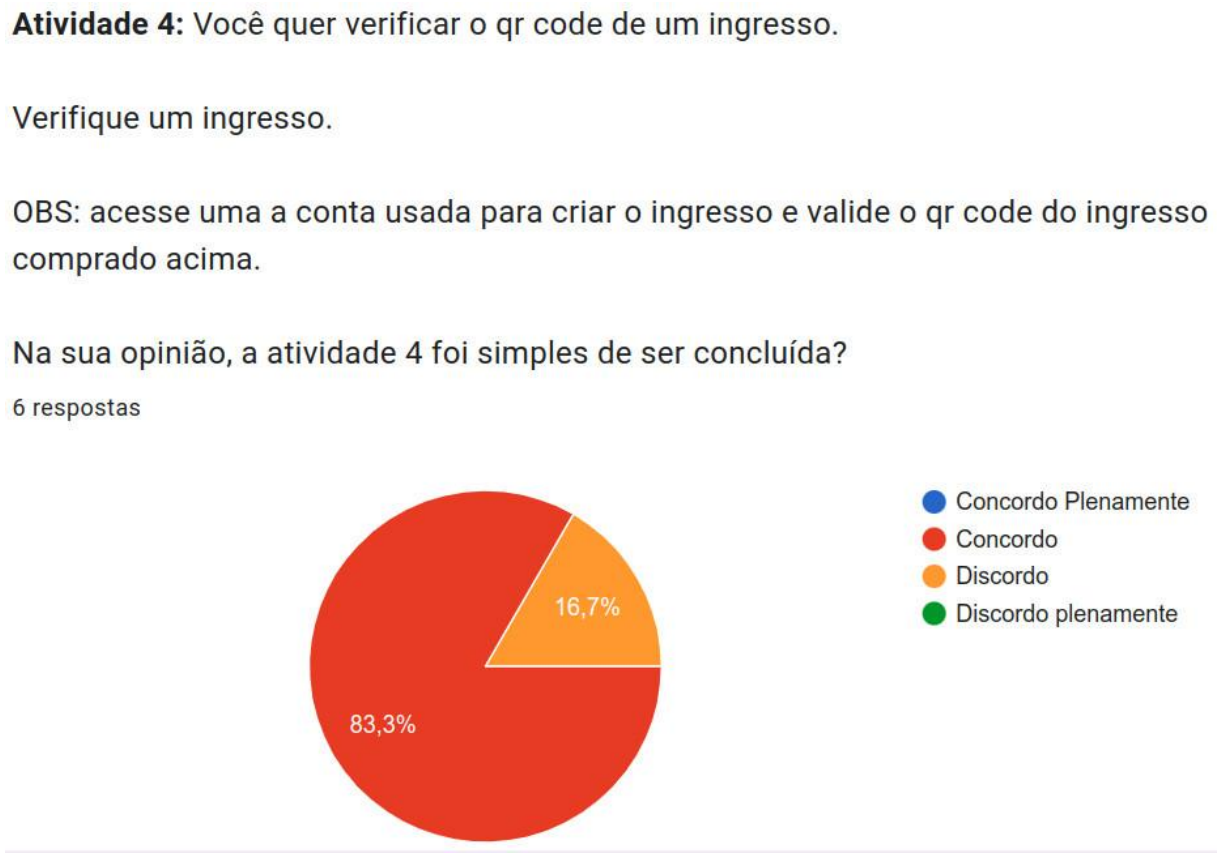


Fonte: Elaborado pelo autor

7.4.4 Análise da atividade para verificar ingresso

Nesta seção, foi analisado se o usuário conseguiria comprar ingressos. A resposta do teste foi boa, com boa parte dos participantes respondendo com “Concordo” que conseguiram executar a atividade com facilidade, como mostrado na figura 11.

Figura 11 - Resultado da atividade de verificar ingresso



Fonte: Elaborado pelo autor

7.5 Discussões sobre os resultados

Apesar dos resultados positivos, em que a grande maioria achou o sistema simples de ser utilizado, melhorias puderam ser levantadas a partir do feedback dos participantes. A experiência de comprar um ingresso pode ser melhorada, fazendo uma pagina para os eventos, em que suas informações fiquem mais fáceis de serem visualizadas, além disso, a listagem dos eventos pode receber filtros por nome e valor por exemplo, o que facilita aos compradores , acharem o evento desejado.

A experiência de mostrar o ingresso em *qr code*, pode ser melhorada, através da criação de uma página exclusiva para os ingressos do usuário, e com uma tela que consiga exibir todas as informações e o botão de exibir *qr code*, sem que precise realizar *scroll* quando estiver sendo utilizada em celulares.

Diante do que foi levantado por meio de feedbacks, fica claro que as melhorias se concentram na aplicação web e que a execução das funcionalidades, da conexão com a carteira e com a ethereum não entraram em pauta, o que mostra como funcionaram bem.

8 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho, foi feita a implementação e um teste de facilidade de uso de uma aplicação de comercialização de ingressos, baseada em blockchain. A Ethereum foi a principal tecnologia utilizada, pois ficou responsável pelo contrato inteligente que cuidou das regras de negócio e do armazenamento dos dados.

A principal dificuldade enfrentada durante o desenvolvimento foi o aprendizado do ethereum, o que incluiu uma nova linguagem que foi a solidity, que apesar das semelhanças com linguagens populares, apresentava particularidades por ser feita para contratos inteligentes.

A solução arquitetural que permite ao organizador do evento limitar ou proibir a revenda de ingressos por parte dos compradores de ingressos, foi feita; bem como o contrato inteligente que implementou a arquitetura. A avaliação de facilidade de uso foi feita com sucesso e obteve um feedback positivo, dessa forma este trabalho atingiu todos os objetivos propostos.

8.1 Conclusões

O teste de facilidade de uso mostrou que a aplicação é simples de usar e que todas as funcionalidades estão funcionando como o esperado. Porém os feedbacks coletados dos participantes da pesquisa, mostrou que a aplicação web ainda pode ser melhorada para que fique mais simples e esteticamente melhor.

8.2 Trabalhos futuros

Para trabalhos futuros, é pretendido uma evolução da aplicação como um todo. Para a aplicação web, pretendemos remodelar o design, de modo a deixar cada funcionalidade com uma tela única, além disso, é pretendido criar uma aplicação que guarde informações a mais além do que se tem no contrato inteligente. como mais informações dos eventos, além de imagens, de modo que a aplicação web, chame essa nova aplicação para obter informações dos eventos ao invés de chamar o contrato inteligente de forma direta. Além disso, pretende-se criar uma aplicação mobile para que não seja preciso usar o navegador da carteira para usar a aplicação no celular.

Pretende-se também implementar a possibilidade de os ingressos serem NFTs(Non-fungible Tokens), com uma imagem associada. Dessa forma, além das funcionalidades já implementadas no ingresso, teríamos a dele ser um NFT, cujo ingresso associado possa ser usado mas o NFT nunca possa ser apagado.

REFERÊNCIAS

ANDROULAKI, E.; BARGER, A.; BORTNIKOV, V.; CACHIN, C.; CHRISTIDIS, K.; De CARO, A.; ENYEART, D.; FERRIS, C.; LAVENTMAN G.; MANEVICH, Y.; MURALIDHARAN, S.; MURTHY, C.; NGUYEN, B.; SETHI, M.; SINGH, G.; SMITH, K.; SORNIOTTI, A.; STATHAKOPOULOU, C.; VUKOLIĆ, M.; COCCO, S. W.; YELLICK, J. Hyperledger fabric: a distributed operating system for permissioned blockchains. *In*: **Proceedings of the Thirteenth EuroSys Conference**, 2018, New York, ACM, 2018, p. 1-15.

CHA, S.; PENG, W.; HSU, T.; CHANG, C.; LI, S. A Blockchain-Based Privacy Preserving Ticketing Service. *In*: **Global Conference on Consumer Electronics**, 7. ed., 2018, Nara, IEEE, 2018, p. 585-587.

GUEGAN, D. **Public Blockchain versus Private blockchain**. 2017. Disponível em: https://econpapers.repec.org/paper/msecesdoc/17020.htm. Acesso em: 21 jun. 2021.

FELIZARDO JUNIOR, B. J. **E-COMMERCE**: UM ESTUDO DE CASO DA VENDA DE INGRESSOS ONLINE PARA ESTUDANTES DO CURSO DE ADMINISTRAÇÃO DE EMPRESAS DO PERÍODO NOTURNO DA UNIVERSIDADE DO EXTREMO SUL CATARINENSE. Monografia (Graduação em Administração) - Universidade do Extremo Sul Catarinense, Criciúma, 2013.

JUNQUEIRA, N. R. **Concessão de Permissão a Dados de Saúde Baseada em Contratos Inteligentes em Plataforma de Blockchain**. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Goiás, Goiânia, 2020.

KING, S.; NADAL, S. **Ppcoin**: Peer-to-peer crypto-currency with proof-of-stake. 2012. Disponível em: http://people.cs.georgetown.edu/~clay/classes/fall2017/835/papers/peercoin-paper.pdf. Acesso em: 22 jun. 2021.

LAI, R.; CHUEN, D. L. K. Blockchain – From Public to Private. *In*: **Handbook of Blockchain, Digital Finance, and Inclusion**. 2. vol. Academic Press, 2018. p. 145-177.

LIN, K.; CHANG, Y.; WEI, Z.; SHEN, C.; CHANG, M. A Smart Contract-Based Mobile Ticketing System with Multi-Signature and Blockchain. *In*: **Global Conference on Consumer Electronics**, 8. ed., 2019, Osaka, IEEE, 2019, p. 231-232.

NAKAMOTO, S. **Bitcoin**: A Peer-to-Peer Electronic Cash System. 2008. Disponível em: https://bitcoin.org/bitcoin.pdf. Acesso em: 20 jun. 2021.

OLSON, K.; BOWMAN, M.; MITCHELL, J.; AMUNDSON, S.; MIDDLETON, D.; MONTGOMERY, CIAN. **Sawtooth**: An Introduction. 2018. Disponível em: https://www.hyperledger.org/wp-content/uploads/2018/01/Hyperledger_Sawtooth_WhitePaper.pdf. Acesso em: 25 jun. 2021.

PREECE, J. D.; EASTON, J. M. Blockchain Technology as a Mechanism for Digital Railway Ticketing. *In*: **International Conference on Big Data**, 2019, Los Angeles, IEEE, 2020, p. 3599-3606.

RIBEIRO, C. M. A. **OS CAMBISTAS E OS PREÇOS DE INGRESSOS EM JOGOS DE FUTEBOL: O CASO DO E. C. BAHIA NO CAMPEONATO BRASILEIRO 2012**. Trabalho de conclusão de curso (Graduação em Ciências Econômicas) - Universidade Federal da Bahia, Salvador, 2014.

RIBEIRO, L.; MENDIZABAL, O. **Introdução à Blockchain e Contratos Inteligentes**: Apostila para Iniciante. 2019. Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/221495/RT-INE2021-1.pdf. Acesso em: 20 jun. 2021.

STALLINGS, W. **Criptografia e segurança de redes:** princípios e práticas. 6. ed. São Paulo: Pearson, 2015.

TACKMANN, B. Secure Event Tickets on a Blockchain. In: **Data Privacy Management, Cryptocurrencies and Blockchain Technology**, 2017, Oslo, Springer, 2017, p. 437-444.

VUKOLIĆ, M. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. *In*: **Open Problems in Network Security**, 2015, Zurich, Springer, 2016, p. 112–125.

WOOD, G. **Ethereum**: A secure decentralised generalised transaction ledger. 2015. Disponível em: http://gavwood.com/paper.pdf. Acesso em: 20 ago. 2021.

APÊNDICE A - CÓDIGO DO CONTRATO INTELIGENTE

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.19;

contract TicketContract {

  struct Ticket {
    uint256 id;
    uint256 eventId;
    string eventName;
    address owner;
    address organizer;
    uint256 age;
    uint256 limit;
    uint256 value;
    bool sale;
  }

  struct GroupTickets {
    uint256 eventId;
    string eventName;
    address organizer;
    uint256 value;
    uint256 quantity;
  }

  Ticket[] public tickets;

  uint256 private countEvent;
  uint256 private taxTransfer = 140000000000000;
  uint16 private maxQuantityTicketCreated = 1000;
```

```
  constructor() {
    countEvent = 1;
  }

  function createTickets(string memory _eventName, uint256 _limit,
uint256 _value, uint16 _quantity) public {
    if (bytes(_eventName).length == 0) {
      revert("Event name is required");
    }

    if (!(_quantity > 0 && _quantity <= maxQuantityTicketCreated)) {
      revert(string(abi.encodePacked("Quantity must be between 1 and ",
maxQuantityTicketCreated)));
    }

    if (_value < taxTransfer) {
      revert(string(abi.encodePacked("Value cannot be less than ",
taxTransfer, " wei")));
    }

    uint256 newEventId = countEvent;
    countEvent++;

    uint256 newId = tickets.length + 1;

    for (uint16 i = 0; i < _quantity; i++) {
      Ticket memory newTicket = Ticket(newId, newEventId, _eventName,
msg.sender, msg.sender, 0, _limit, _value, true);
      tickets.push(newTicket);
      newId++;
    }
  }
```

```
  function buyTicket(uint256 _eventId, address _owner, bool _sale)
public payable returns (uint256) {
    for (uint256 i = 0; i < tickets.length; i++) {
      if (tickets[i].eventId == _eventId && tickets[i].owner == _owner
&& tickets[i].age < tickets[i].limit && tickets[i].sale) {
        if (msg.value < tickets[i].value) {
          revert("Insufficient payment sent.");
        }

        address payable seller = payable(tickets[i].owner);
        seller.transfer(tickets[i].value - taxTransfer);
        tickets[i].age++;
        tickets[i].owner = msg.sender;
        tickets[i].sale = _sale;

        return tickets[i].id;
      }
    }
    revert("Tickets not available");
}

  function filterTicketsByOwner(uint256 _eventId, address _owner, bool
transferable, bool _sale) public view returns (Ticket[] memory) {
    if (_owner == address(0)) {
      revert("Owner address is required.");
    }
    Ticket[] memory filteredTickets = new Ticket[](tickets.length);
    uint256 numberTickets = 0;

    bool byEventId = _eventId != 0;

    for (uint256 i = 0; i < tickets.length; i++) {
```

```
      if (byEventId) {
        if (tickets[i].eventId != _eventId) {
          continue;
        }
      }
      if (tickets[i].owner != _owner) {
        continue;
      }
      if (tickets[i].sale != _sale) {
        continue;
      }

      if (transferable) {
        if (tickets[i].age == tickets[i].limit) {
          continue;
        }
      }

      filteredTickets[numberTickets] = tickets[i];
      numberTickets++;
    }

    assembly {
      mstore(filteredTickets, numberTickets)
    }

    return filteredTickets;
  }

 function getGroupTickets() public view returns (GroupTickets[] memory)
{
    GroupTickets[] memory groupTickets = new
```

```
    GroupTickets[](tickets.length);
    uint256 numberGroupTickets = 0;
    for (uint256 i = 0; i < tickets.length; i++) {
      if(!(tickets[i].age < tickets[i].limit && tickets[i].owner ==
tickets[i].organizer)) {
        continue;
      }
      bool exist = false;
      for (uint256 j = 0; j < groupTickets.length; j++) {
        if (groupTickets[j].eventId == tickets[i].eventId) {
          exist = true;
          groupTickets[j].quantity++;
          break;
        }
      }

      if (!exist) {
        GroupTickets memory newGroupTicket =
GroupTickets(tickets[i].eventId, tickets[i].eventName,
tickets[i].organizer, tickets[i].value, 1);
        groupTickets[numberGroupTickets] = newGroupTicket;
        numberGroupTickets++;
      }
    }

    assembly {
      mstore(groupTickets, numberGroupTickets)
    }

    return groupTickets;
  }

  function verifyTicket(uint256 _ticketId, bytes32 _hashedMessage,
```

```
address _owner, uint8 _v, bytes32 _r, bytes32 _s) public {
    bytes32 hashMessage = keccak256(abi.encodePacked("ticketId=",
_uint256ToString(_ticketId)));
    if (hashMessage != _hashedMessage) {
      revert();
    }
    bytes memory prefix = "\x19Ethereum Signed Message:\n32";
    bytes32 prefixedHashMessage = keccak256(abi.encodePacked(prefix,
_hashedMessage));
    address recoveredSigner = ecrecover(prefixedHashMessage, _v, _r,
_s);
    if (recoveredSigner != _owner) {
      revert();
    }

    for (uint256 i = 0; i < tickets.length; i++) {
      if (tickets[i].id == _ticketId) {
        if (tickets[i].owner != _owner) {
          revert();
        }
        if (tickets[i].organizer != msg.sender) {
          revert();
        }

        tickets[i] = tickets[tickets.length - 1];
        tickets.pop();

        return;
      }
    }

    revert();
}
```

```
  function _uint256ToString(uint256 number) internal pure returns
(string memory) {
    if (number == 0) {
      return "0";
    }
    uint256 tempNumber = number;
    uint256 digits;
    while (tempNumber != 0) {
      digits++;
      tempNumber /= 10;
    }
    bytes memory buffer = new bytes(digits);
    while (number != 0) {
      digits -= 1;
      buffer[digits] = bytes1(uint8(48 + number % 10));
      number /= 10;
    }
    return string(buffer);
  }
}
```

APÊNDICE B - TELAS DA APLICAÇÃO

Figura 12 - Tela inicial com carteira não conectada

Fonte: Elaborado pelo autor

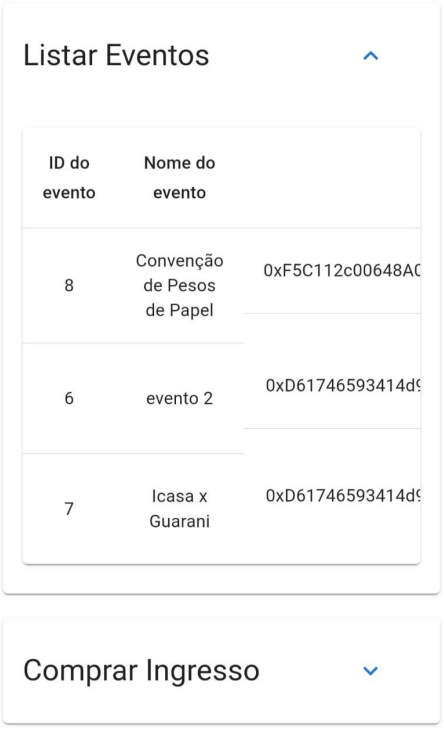
Figura 13 - Tela inicial com a carteira conectada

Fonte: Elaborado pelo autor

Figura 14 - Parte de criar ingressos expandida

Fonte: Elaborado pelo autor

Figura 15 - Parte de listar eventos expandida, com foco no nome do evento

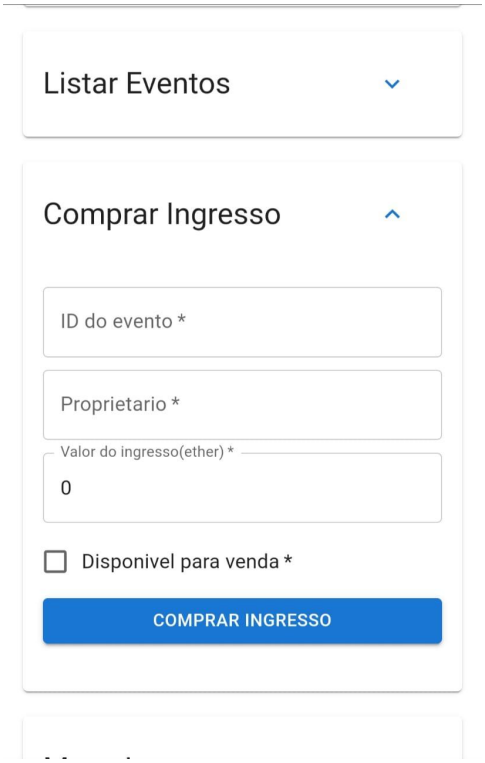


Fonte: Elaborado pelo autor

Figura 16 - Parte de listar eventos expandida, com foco no botão de comprar ingresso

Fonte: Elaborado pelo autor

Figura 17 - Parte de comprar ingresso expandida



Fonte: Elaborado pelo autor

Figura 18 - Parte de listar meus ingressos expandida

Fonte: Elaborado pelo autor

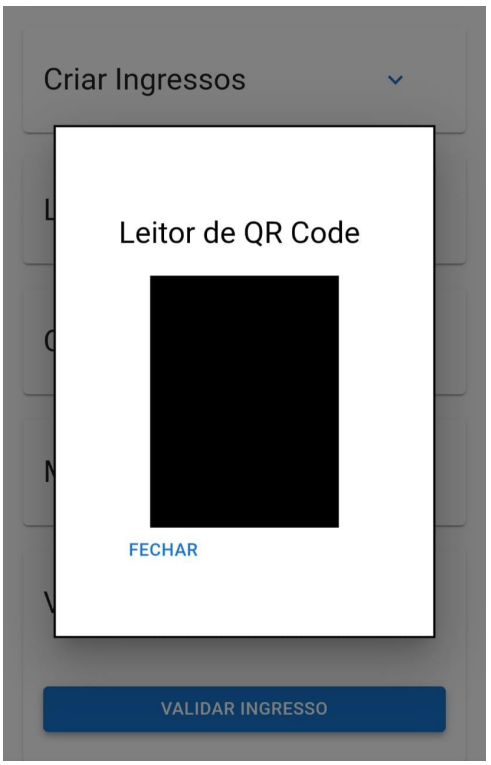
Figura 19 - Modal com o qr code do ingresso

Fonte: Elaborado pelo autor

Figura 20 - Botão de verificar ingresso

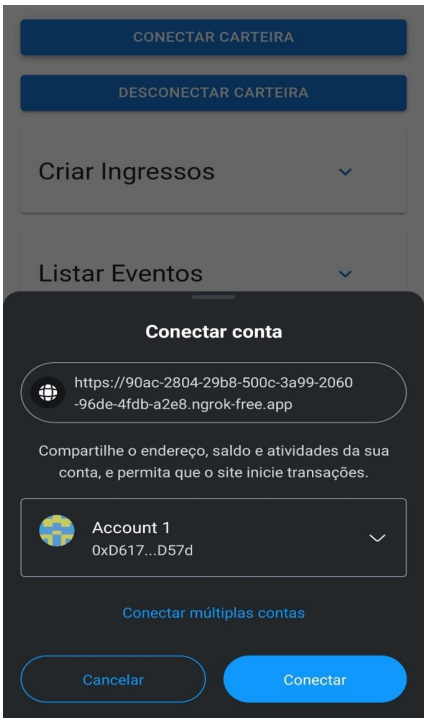
Fonte: Elaborado pelo autor

Figura 21 - Modal com o leitor de qr code



Fonte: Elaborado pelo autor

Figura 22 - Conectando a carteira



Fonte: Elaborado pelo autor