HENRIQUE YUKIO MURATA
MATHEUS RIBEIRO LIRA
OTÁVIO HENRIQUE MONTEIRO

CARBON21: DESENVOLVIMENTO DE UMA PLATAFORMA DE TOKENIZAÇÃO PARA AÇÕES DE PRESERVAÇÃO AMBIENTAL E REFLORESTAMENTO

São Paulo
2022

HENRIQUE YUKIO MURATA
MATHEUS RIBEIRO LIRA
OTÁVIO HENRIQUE MONTEIRO

CARBON21: DESENVOLVIMENTO DE UMA PLATAFORMA DE TOKENIZAÇÃO PARA AÇÕES DE PRESERVAÇÃO AMBIENTAL E REFLORESTAMENTO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Eletricista com ênfase em Computação.

São Paulo
2022

HENRIQUE YUKIO MURATA
MATHEUS RIBEIRO LIRA
OTÁVIO HENRIQUE MONTEIRO

CARBON21: DESENVOLVIMENTO DE UMA PLATAFORMA DE TOKENIZAÇÃO PARA AÇÕES DE PRESERVAÇÃO AMBIENTAL E REFLORESTAMENTO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Eletricista com ênfase em Computação.

Área de Concentração:

Computação

Orientador:

Prof. Dr. Marcos Antonio Simplicio Junior

São Paulo
2022

RESUMO

Diante do atual contexto de crescente preocupação da sociedade e do universo empresarial com a preservação do meio ambiente, o presente trabalho de formatura surge como uma tentativa de contribuir para esse objetivo. O projeto consiste em uma prova de conceito de uma plataforma de tokenização de ações de preservação ambiental e reflorestamento. Áreas de terra preservadas ambientalmente podem ser tokenizadas por NFTs, que por sua vez podem ser utilizados para compensação ambiental e rendem FTs periodicamente como "dividendos de preservação", chamados de Sylvas. Usuários podem transferir *tokens* entre si caso queiram e podem consultar sua carteira de NFTs e seu saldo de Sylvas. O usuário pode escolher entre gerar a sua chave privada utilizada para assinar transações localmente, em seu próprio navegador, ou por intermédio do servidor, sem tráfego dessa chave pela rede. Os principais componentes desenvolvidos são a rede *Blockchain Hyperledger* para execução, ordenação e registro descentralizado das transações, o *Website* por onde o usuário interage com todo o sistema e a API que recebe os pedidos do *Website*, realiza as chamadas à *Blockchain* e publica os metadados dos NFTs na rede IPFS (*InterPlanetary File System*). Existem perspectivas de continuidade para o projeto, visando a construção de soluções como um *Marketplace*, por onde os usuários podem comprar, vender e compensar *tokens*, e também de desenvolver integrações com órgãos ambientais para validação documental das áreas de terra para a correta emissão dos NFTs.

Palavras-Chave – Blockchain, Tokenização, Meio Ambiente, Preservação Ambiental, Reflorestamento, NFT, Hyperledger Fabric, Nodejs.

ABSTRACT

Given the current context of growing concern of society and of the business universe with environment preservation, the present graduation work emerges as an attempt to contribute towards this goal. The project consists of a Proof Of Concept of a tokenization platform for environmental preservation and reforestation actions. Environmentally preserved land areas can be tokenized via NFTs, which in turn can be used for environmental compensation and yield FTs periodically as "preservation dividends", called *Sylvas*. Users can transfer tokens amongst themselves, if they wish so, and are also able to view their portfolio of NFTs and their *Sylvas* balance. The user can choose between generating his private key used for signing transactions locally, in his own browser, or through server mediation, without having this key transmited through the network. The main components developed are the Hyperledger Blockchain network for decentralized execution, ordering and recording of transactions, the Website where the user interacts with the whole system and the API that receives the requests from the Website, makes the calls to the Blockchain and publishes the metadata of the NFTs on the IPFS (InterPlanetary File System) network. There are prospects of continuity for the project, aiming to build solutions such as a Marketplace, where users can buy, sell and compensate tokens, and also to develop integrations with environmental agencies for documentary validation of land areas for the correct issuance of NFTs.

Keywords – Blockchain, Tokenization, Environment, Environmental preservation, Reforestation, NFT, Hyperledger Fabric, Nodejs.

AGRADECIMENTOS

Aos familiares e amigos, por todo o apoio e pela ajuda, que muito contribuíram ao decorrer da realização deste trabalho.

Ao professor Marcos Simplício, nosso orientador, e ao professor Charles Miers, que participou ativamente do projeto, por toda a dedicação e amizade de ambos.

A todos que participaram, direta e indiretamente, do projeto Carbon 21, sem o qual este trabalho não existiria.

À Escola Politécnica da USP, essencial em nosso processo de formação profissional, pela dedicação, e por tudo o que aprendemos ao longo dos anos do curso.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O projeto de conclusão de curso consiste na especificação, desenvolvimento e implantação de uma plataforma de tokenização que promova o plantio de novas árvores e preservação de áreas florestadas, fazendo uso da tecnologia de *Blockchain* como meio para geração de valor para os envolvidos nesse tipo de iniciativa.

1.1 Motivação

O projeto tem como motivação o atual contexto de crescente preocupação do universo empresarial com aspectos como o meio ambiente e a responsabilidade social, representados pelo conceito ESG (*Environmental, Social, Governance* - Meio Ambiente, Social, Governança), cada vez mais difundido e adotado no mundo corporativo. Essa revolução relacionada ao impacto socioambiental empresarial visa regenerar recursos naturais, provendo a prosperidade compartilhada e diminuindo a desigualdade. O objetivo é criar "Um mundo focado não apenas em minimizar danos, mas em fazer o bem de forma mensurável" [1].

Em um âmbito global, o interesse cada vez maior pela temática ESG tem promovido grandes mudanças no mundo dos investimentos. Investidores tem se envolvido com o conceito de "investimento responsável", buscando alocar seus recursos em portfólios alinhados com pautas ambientais. Como consequência, as empresas também têm se movido na direção de adotar as práticas ESG [2].

No cenário brasileiro, o tema ESG ainda se encontra tímido; contudo, tem ganhado mais holofote nos últimos meses, o que pode ser percebido no comportamento dos investidores e nas empresas. Em relação aos investidores, estes estão começando a levar em conta fatores ESG em seus investimentos [2]. De acordo com o Retrato da Sustentabilidade no Mercado de Capitais feito pela ANBIMA (Associação Brasileira das Entidades dos Mercados Financeiro e de Capitais) em 2021, das 265 instituições entrevistadas, a maioria composta por bancos e gestoras de recursos, 35 % já implantaram processos de integração da Sustentabilidade nas decisões estratégicas do negócio com a exclusão de

oportunidades que não se enquadrem nesses critérios, enquanto 37 % estão em fase de implantação e disseminação [3]. Já no âmbito empresarial, a maioria das empresas está no início da adoção da prática de divulgar informações relacionadas a questões ESG [2].

Diante desse contexto de crescente preocupação com os impactos ambientais, surge a motivação para o presente trabalho; mais especificamente, no tópico de preservação ambiental e reflorestamento. O projeto busca tornar a atividade de reflorestamento economicamente interessante por meio da criação de uma plataforma de tokenização de ativos florestais. Esta plataforma se sustenta de forma independente e menos complexa do que soluções existentes como o mercado de crédito de carbono, além de buscar tirar proveito da legislação local, tendo o Brasil com caso de estudo. Assim, ela pode ser vista como um mecanismo complementar ao mercado de carbono tradicional, em oposição a tentar ser um concorrente.

Finalmente, do ponto de vista tecnológico, a plataforma tem como base a recente consolidação de tecnologias descentralizadas no cenário de tokenização de ativos, bem como em sua grande expansão, abrangendo ativos de Arte, Colecionáveis, Jogos, Utilitários, dentre outros [4]. Um dos exemplos mais relevantes nessa revolução do mercado de ativos digitais é a tecnologia de *Blockchain*, usada em plataformas como *Ethereum* (ETH).

1.2 Objetivo

O projeto consiste em avaliar a possibilidade de desenvolvimento e iniciar a implementação de uma plataforma de tokenização que promova o reflorestamento e a preservação das áreas reflorestadas. A contribuição do sistema para tais metas deve ocorrer por meio da geração de *NFTs* (*Non-Fungible Tokens* - *tokens* não-fungíveis), que são *tokens* que representam uma única entidade e dão a ela o seu valor intrínseco. No caso do projeto, os NFTs representam áreas de terra devidamente registradas no sistema e que são cobertas por florestas ou que serão reflorestadas. Os *NFTs* registrados devem render periodicamente *FTs* (*Fungible Tokens* - *tokens* fungíveis), ou como são chamados na plataforma, Sylvancoins ou simplesmente Sylvas, representando "dividendos de preservação", e assim fornecendo uma renda como forma de motivar aos proprietários de terra que queiram preservar a vegetação de suas terras. Além disso, os NFTs teriam uma segunda utilidade representada pela possibilidade de serem utilizados para fins de compensação ambiental, conforme legislação vigente no Brasil.

1.3 Justificativa

O reflorestamento promovido pelo projeto traria uma série de benefícios como desaceleração das mudanças climáticas, preservação da biodiversidade, combate à desertificação, sobrevivência de comunidades que dependem de recursos florestais, geração de empregos, melhoria da qualidade de vida, entre muitos outros benefícios [5]. Diferentemente de esforços tradicionais de reflorestamento, baseados primariamente no voluntariado, plataformas de tokenização fornecem um modelo de negócio claro aos interessados no plantio de árvores. De fato, existem alguns exemplos recentes de iniciativas desse tipo no mundo, as quais em parte servem de inspiração para a proposta aqui apresentada.

Como modelo mais tradicional, conta-se, por exemplo, com o mercado de créditos de carbono, que permite que países busquem atingir de forma colaborativa suas metas de redução da emissão de gases estufa. Essa cooperação envolve acordos envolvendo investimentos e transferência de tecnologia por parte de países desenvolvidos que não conseguiriam atingir suas metas de redução e acabam comprando créditos de carbono de países em desenvolvimento [6]. Contudo, ainda existem ao menos duas barrerias para sua ampla adoção como mecanismo para promover o reflorestamento: a alta complexidade no cálculo da geração de créditos de carbono, que muitas vezes exigindo consultoria especializada para auxiliar no gerenciamento desses créditos; e a necessidade de se dispor de amplas áreas de plantio (da ordem de 10 mil hectares) para que a geração de créditos de carbono seja viável [7].

Em relação a modelos recentes de negócio baseados em tecnologias descentralizadas de tokenização, iniciativas como a *Smart Forest Coin* [8] e a *Tree Cycle* [9] geram NFTs referentes às árvores, que criam lucros com seu corte e venda. Uma desvantagem dessa abordagem é não promover a preservação a longo prazo das árvores plantadas, visto que o objetivo final é o seu corte. Já outras plataformas, como *Forestcoin* [10], *Tree Defi* [11] e *Moss* [12] buscam explorar esses benefícios ecológicos de longo prazo. Isto é feito por meio de geração de NFTs por plantio e autoverificação, leilão de NFTs atrelados às árvores juntamente a *tokens* de CO_2 ou até mesmo por meio da facilitação do ingresso no mercado de carbono. Cada abordagem apresenta benefícios interessantes, mas também desafios e desvantagens, tais como a dificuldade em rastrear e validar a existência e saúde das árvores, ou a barreira imposta pela natureza do mercado de carbono, pouco adequado para pequenas propriedades [7].

Nesse contexto, a CARBON 21 visa tornar o reflorestamento um negócio lucrativo e sedutor para ambos os detentores de pequenas e grandes propriedades. Essencialmente,

a plataforma disponibiliza a criação de *tokens* que exploram diferentes formas de geração de renda vinculadas ao (re)florestamento: exploração de leis de compensação ambiental existentes no Brasil e em alguns outros países do mundo; exploração de mercado de manejo de madeira, com o corte e venda de árvores; e a valorização de benefícios ambientais trazidos diretamente por áreas de floresta, como captura de carbono e melhoria de microclima local, traduzida na cunhagem de moedas digitais apoiada pelas respectivas áreas florestais do mundo real. Para se beneficiarem da iniciativa CARBON 21, os proprietários de terras precisam registrar as suas terras na plataforma, provando que as autoridades competentes aprovaram a utilização das terras pretendidas para o cultivo de árvores.

O projeto fará uso de uma arquitetura baseada em tecnologia *Blockchain* consorciada, com o objetivo de fornecer total transparência aos valores gerados (e.g., impedir que entidades do sistema, incluindo administradores, gerem NFTs ou FTs sem respeitar regras pré-definidas para tal). Nessa plataforma, é permitido que qualquer usuário devidamente registrado realize transações e acesse o conteúdo completo da *Blockchain*; contudo, a inserção de novos blocos na *Blockchain* por meio da participação no mecanismo de consenso deve ficar restrita a poucas entidades consorciadas. Essa estratégia permite a adoção de mecanismos de consenso bastante leves e eficientes, em oposição ao que se observa com relação ao consumo de energia com sistemas abertos como o Bitcoin [13]. Ao mesmo tempo, a abertura de acesso ao conteúdo da *Blockchain* garante que qualquer tentativa de modificação ou criação de registros em desacordo com as regras do sistema possa ser rapidamente detectada por qualquer usuário da rede. Assim, mesmo que haja conluio entre diversos usuários da rede, a presença de uma única entidade honesta monitorando o sistema é capaz de identificar (e, assim, prevenir) ações maliciosas.

Os trabalhos realizados pelo grupo são um recorte de um projeto de escala maior, o CARBON 21, que envolve equipe multidisciplinar de pesquisadores da USP e alhures para tratar aspectos técnicos, econômicos e legislatórios do sistema. A equipe do projeto faz parte desse grupo, se concentrando em seus aspectos técnicos.

1.4 Organização do Trabalho

O restante deste trabalho está organizado nos seguintes capítulos. No capítulo 2, são abordados os aspectos conceituais envolvidos no trabalho. Como tópicos, são cobertos o conceito de *Blockchain*, contratos inteligentes, mecanismos de consenso e tipos de rede (pública ou permissionada). No capítulo 3, é descrita a metodologia do trabalho aplicada desde a concepção do projeto com seus requisitos até as definições de tecnologia e *MVP*,

que se concluem com a implementação e validação pretendidas. No capítulo 4, são exploradas mais a fundo as especificações de requisitos definidos para o projeto que nortearão o seu desenvolvimento. No capítulo 5, é descrito o desenvolvimento do trabalho, envolvendo as técnicas e tecnologias adotadas.

2 ASPECTOS CONCEITUAIS

Neste capítulo são abordados os principais conceitos técnicos necessários para o bom entendimento das tecnologias utilizadas e do projeto como um todo. São abordados desde tópicos como conceitos e serviços de segurança da informação até *Blockchain* e *password hashing*.

2.1 Conceitos de Segurança da Informação

É importante que, desde a sua concepção, a plataforma da *Carbon* se preocupe com a segurança dos seus processos, dos seus dados e dos seus usuários. Deseja-se proteger a plataforma de práticas como o vazamento de senhas, execução de transações por pessoas se passando por usuários legítimos, adulteração do registro de transações, entre outras. Para a melhor compreensão das medidas de segurança adotadas durante o desenvolvimento do projeto, são descritos a seguir alguns conceitos relacionados à Segurança da Informação.

2.1.1 Serviços de Segurança

Existem diferentes aspectos de segurança de um sistema de computadores. Tais aspectos são também denominados serviços de segurança, e podem ser divididos em Disponibilidade, Confidencialidade, Integridade, Autenticidade e Irretratabilidade.

2.1.1.1 Disponibilidade

O serviço de Disponibilidade busca garantir que os usuários legítimos consigam acessar as informações e recursos do sistema. Quanto mais crítico for o sistema, maior será a disponibilidade requerida. A interrupção de um serviço remove o acesso dos clientes às funcionalidades do sistema, bem como o acesso da administração aos recursos necessários para a condução de tarefas críticas. A perda do serviço pode significar prejuízos financeiros significativos por meio da diminuição da produtividade dos colaboradores e da potencial

perda de clientes [14].

2.1.1.2 Confidencialidade

O serviço de Confidencialidade busca garantir que informações armazenadas em um sistema de computação ou transmitidas em uma rede não sejam reveladas a usuários não-autorizados. Dessa forma, a informação armazenada ou transmitida é representada de maneira obscura e não compreensível; somente os usuários legítimos mediante um algoritmo podem torná-la clara. A perda desse serviço consiste, portanto, na obtenção da mensagem clara por pessoas não-autorizadas [14].

2.1.1.3 Integridade

A Integridade consiste na identificação de alterações em um conjunto de dados armazenados ou transmitidos. Esse serviço não impede a modificação dos dados, mas permite a sua detecção [14].

2.1.1.4 Autenticidade

A Autenticidade é a garantia de que o remetente de uma mensagem seja corretamente identificado pelo seu destinatário. Ela está relacionada com duas etapas:

- Identificação: quando um usuário apresenta suas informações de identificação ao sistema;
- Verificação: apresentar ou gerar informações de autenticação que servem de prova do seu vínculo com o usuário que deseja utilizar o sistema [15].

2.1.1.5 Irretratabilidade

A Irretratabilidade é a propriedade de um sistema que permite que ações em um sistema possam ser rastreadas até identificar o seu verdadeiro autor. Isso confere responsabilidade aos usuários, que podem ser penalizados por ações maliciosas como quebras de segurança e violação de regras [16].

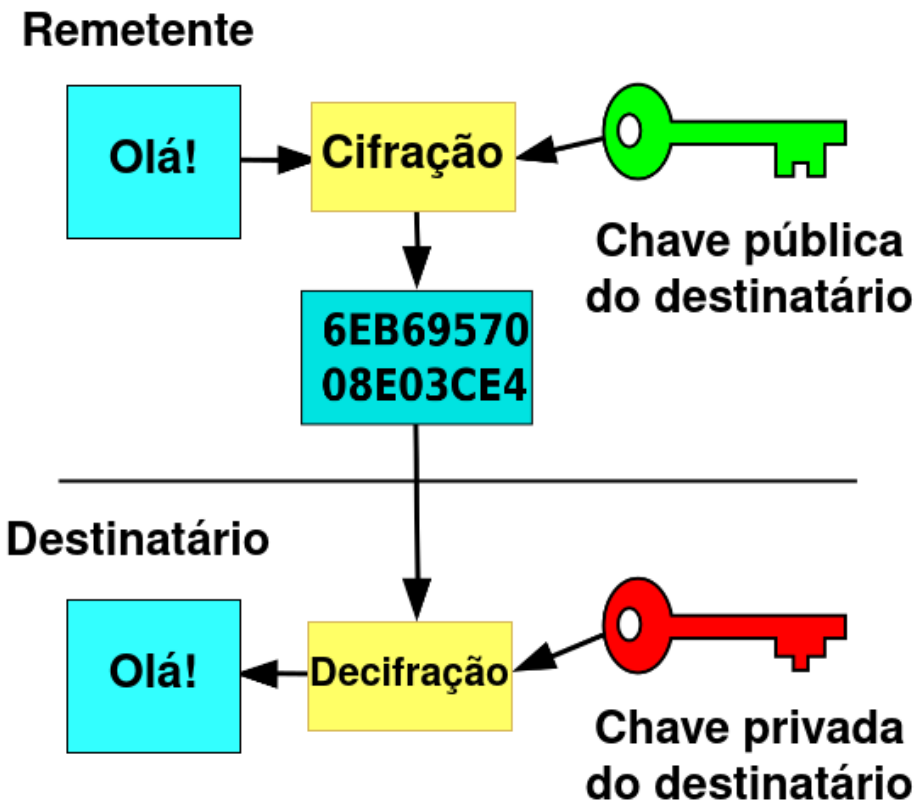
2.1.2 Criptografia Assimétrica

A criptografia assimétrica, também conhecida como criptografia de chave pública, é um sistema criptográfico que utiliza um par de chaves de forma que uma chave realiza a operação inversa que a outra provoca. Essa forma de criptografia permite a execução de serviços como a cifração e a decifração de dados e a criação e verificação de assinaturas digitais [17]. As chaves do par mencionado são chamadas de pública e privada, de forma que esta é armazenada somente pelo seu dono e não deve ser exposta a terceiros, enquanto que aquela é disponibilizada para a Internet.

2.1.3 Cifração e Decifração

A cifração é a transformação criptográfica de uma informação de forma a ocultar o seu significado por meio de uma representação de dados obscura, chamada de cifra. A decifração é o processo inverso, em que a cifra é transformada na informação original às claras [18]. A cifração, portanto, confere o serviço de Confidencialidade. Para o caso de criptografia assimétrica, o remetente deve cifrar a mensagem com a chave pública do destinatário, que por sua vez decifrará a mensagem recebida com a sua chave privada.

Figura 1: Cifração e decifração por criptografia assimétrica



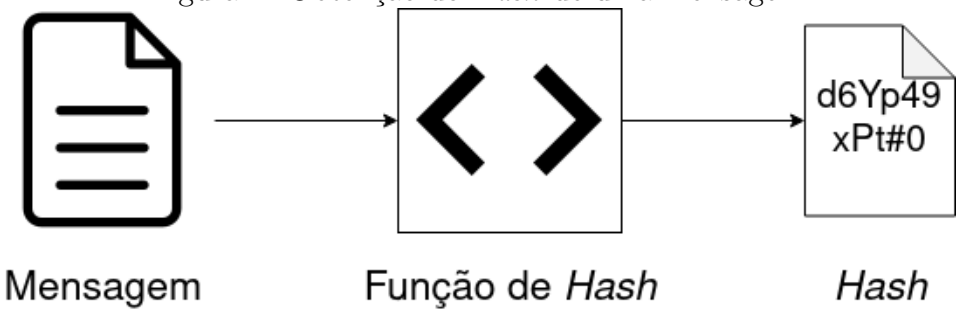
Fonte: adaptado de [19]

2.1.4 Autenticação de Mensagens e *Hash*

A autenticação de mensagens é a proteção contra ataques ativos como falsificação e adulteração. Uma mensagem é tida como autêntica se ela for íntegra e se ela de fato procede de quem ela afirma ter vindo (i.e., se a sua origem for autêntica). Uma das maneiras de se conferir autenticação de mensagens é por meio de uma função de *Hash*.

Uma função de *Hash* recebe uma mensagem M de tamanho variável e produz um resultado $H(M)$ de tamanho fixo como saída, denominado de *hash* e que representa a identidade da mensagem. Isso permite a verificação da integridade da mensagem, que é transmitida juntamente com o seu *hash* a um destinatário. Caso ocorra modificações na mensagem, o seu *hash* também apresentará diferença, que será identificada pela comparação entre o *hash* transmitido pelo remetente e o obtido pelo destinatário ao aplicar a função *Hash* na mensagem recebida.

Figura 2: Obtenção do *Hash* de uma mensagem

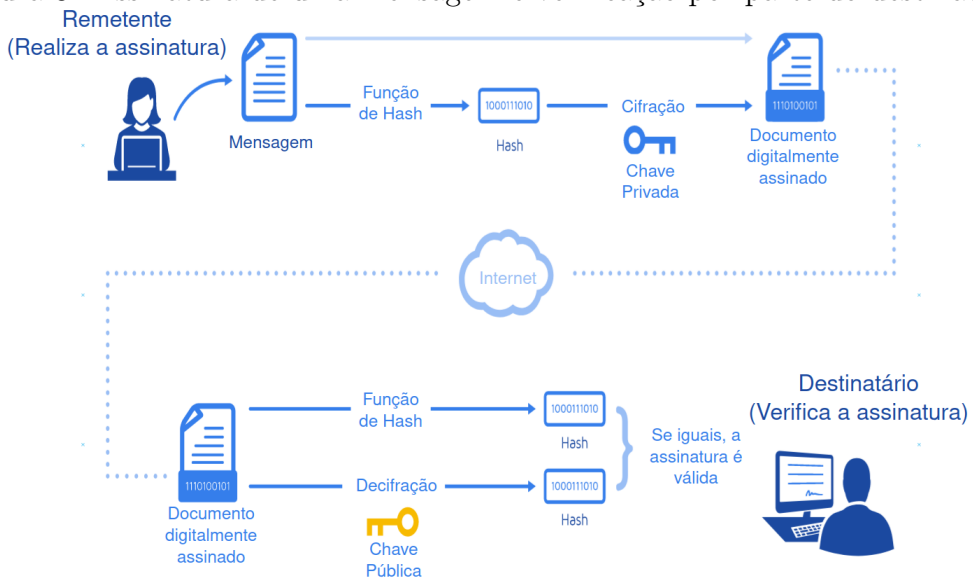


Fonte: Elaboração própria

2.1.5 Assinatura Digital

A assinatura digital é o resultado da cifração do *hash* de uma mensagem transmitida por meio da chave privada do remetente. A cifra resultante é enviada junto com a mensagem e, ao ser recebida pelo destinatário, é decifrada por meio da chave pública do remetente, obtendo-se o *hash* da mensagem original. O destinatário também obtém o *hash* da mensagem e realiza a comparação dos dois valores obtidos. Esse processo permite a verificação da identidade do remetente, uma vez que a chave pública só consegue decifrar aquilo que a sua chave privada correspondente tenha cifrado [20]. Além disso, é conferido se houve modificações na mensagem recebida pelo fato de que modificações alterarão o valor do *hash* a ser comparado com a decifração da assinatura. Portanto, a assinatura digital confere os serviços de Autenticidade, Irretratabilidade e Integridade.

Figura 3: Assinatura de uma mensagem e verificação por parte do destinatário



Fonte: adaptado de [21]

2.1.6 Certificados Digitais

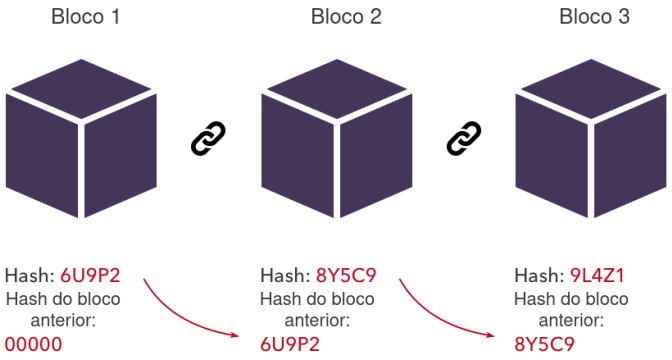
A criptografia de chave pública permite que um usuário possa divulgar a sua chave pública para qualquer outro participante na rede. Essa abordagem, no entanto, possui uma fragilidade: qualquer usuário pode assumir a identidade de outra pessoa e divulgar a sua chave pública, assim podendo se autenticar em sistemas e lendo as mensagens destinadas à identidade original até que seja descoberto [20].

Os certificados digitais, também chamados de certificados de chave pública, surgiram como solução para esse problema. Em suma, são certificados emitidos por uma entidade confiável aos membros de um sistema ou de uma rede. Contêm o identificador e a chave pública de um usuário, as informações da entidade confiável e, por fim, a assinatura de todo o bloco de informações mencionado pela chave privada dessa entidade, que é chamada de *Certificate Authority* (CA - Autoridade Certificadora). Qualquer usuário que precise da chave pública de uma identidade específica pode obtê-la e verificar a sua validade por meio da assinatura da CA emissora [20].

2.2 *Blockchain*

Uma *Blockchain* é essencialmente uma forma de implementação de uma Autoridade de Carimbo de Tempo de forma descentralizada [22]. Um uso comum dessa tecnologia consiste na construção de mecanismo para registro de transações - também chamado de livro-razão ou *ledger* - mantido por uma rede distribuída de nós, ou *peers*, que compartilham uma mesma visão do conteúdo armazenado e conseguem detectar tentativas de modificação do histórico por parte de seus pares. Para que isso seja possível, cada nó costuma manter uma cópia do *ledger* global, que é composto por uma sequência de transações realizadas, organizadas na forma de blocos encadeados sequencialmente. Essa cadeia se mantém unida pelo fato de que cada bloco contém o valor do *hash* do bloco anterior [23]. Essa característica permite a identificação de mudanças no *ledger*, pois a mínima alteração realizada em algum bloco provoca a mudança do seu *hash*, de forma que o bloco posterior, supostamente não modificado, deixaria de apontar para o bloco alterado. Dessa forma, a tecnologia *Blockchain* é um mecanismo distribuído que busca conferir integridade a um registro de transações (tanto em termos de ordem como de conteúdo) mantido pelos nós componentes da rede.

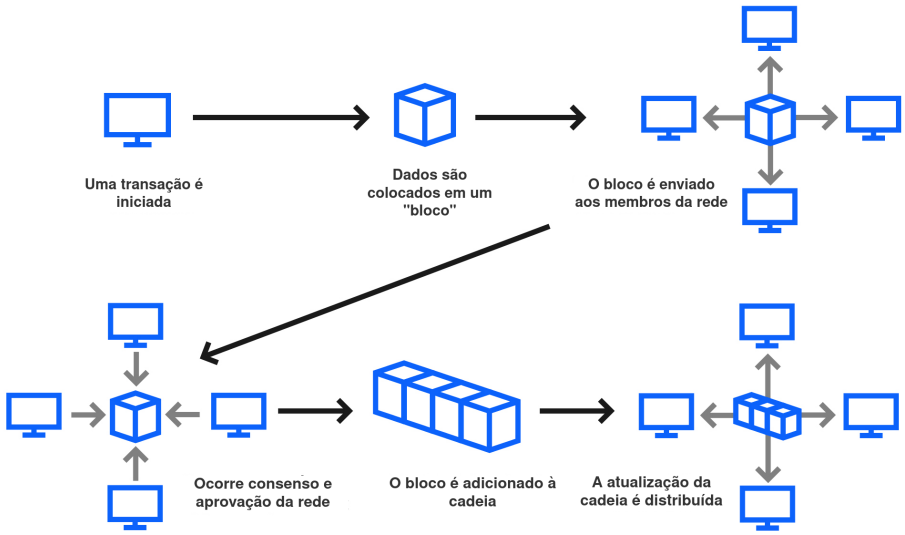
Figura 4: Estrutura geral de uma *Blockchain*



Fonte: adaptado de [24]

Para que sejam canonizadas em uma *Blockchain*, as transações precisam ser analisadas e aprovadas por um grupo de nós da rede mediante um mecanismo de consenso. Há diferentes tipos de mecanismo de consenso possíveis, cada um sendo mais adequado para aplicações específicas. Uma vez realizado o consenso sobre uma transação, esta é adicionada à cadeia de blocos e todos os nós que armazenam uma cópia do *ledger* são atualizados com o novo bloco. A figura a seguir resume o comportamento geral de uma rede *Blockchain*.

Figura 5: Funcionamento de uma rede *Blockchain* genérica



Fonte: adaptado de [25]

2.2.1 Contratos Inteligentes (*Smart Contracts*)

Contratos Inteligentes, *Smart Contracts*, ou ainda *Chaincodes* (como são chamados na plataforma do *Hyperledger Fabric*), são contratos escritos na forma de códigos que contêm a lógica programável de um sistema. Embora não precisem necessariamente ser associados a *Blockchains* até porque o termo data da década de 90 (anterior ao conceito do que hoje se convenciona chamar de blockchains), atualmente essa associação é bastante comum [26].

Em particular, quando associados a uma *Blockchain* que processa transações, contratos inteligentes permitem que tais transações, uma vez cumpridas as condições contratuais predefinidas, sejam executadas automaticamente sem a necessidade da presença de uma autoridade central confiável [27].

Contratos inteligentes podem ser escritos em linguagens de programação específicas, como é o caso do *Solidity* e do *Vyper* utilizados nos *Smart Contracts* executados pela EVM (*Ethereum Virtual Machine*), máquina virtual que compõe o ecossitema *Blockchain* da *Ethereum*. No entanto, eles também podem ser escritos em linguagens de programação de propósito geral, como Go, Java e JavaScript, como é o caso dos Chaincodes utilizados pelo *Hyperledger Fabric*.

2.2.2 Mecanismos de Consenso

Para que as transações entrem definitivamente para o *ledger* da *Blockchain*, deve existir um mecanismo de consenso pelo qual os nós entram em acordo sobre quais transações serão aceitas e qual será a sua ordenação. Isso se faz necessário para evitar cenários conhecidos como bifurcações, ou *forks*, em que nós diferentes possuem diferentes versões de *ledgers*.

Os mecanismos de consenso devem possuir tolerância a casos de falha, que podem ser divididos em duas categorias: falhas por indisponibilidade involuntária de nós, e falhas pela existência de nós maliciosos que desejam comprometer o funcionamento da rede. As primeiras são comumente tratadas por protocolos *CFT* (*Crash Fault Tolerance*), e as últimas por protocolos *BFT* (*Byzantine Fault Tolerance*).

Existem diversos tipos de mecanismo de consenso, sendo alguns bastante similares entre si, e outros cujos paradigmas utilizados para se atingir o consenso são bastante distintos. Por exemplo, alguns se baseiam em processamento computacional para atingir o consenso (como o *Proof-of-Work*), enquanto outros dependem diretamente da autoridade

dos nós envolvidos (como o *Proof-of-Authority*). Para o presente trabalho, o mecanismo de consenso utilizado é descrito na seção 5.1, onde são descritas as tecnologias utilizadas no projeto.

2.2.3 Permissionamento

As redes *Blockchain* podem ser classificadas em permissionadas e não-permissionadas. Diz-se que uma *Blockchain* é permissionada quando existem restrições quanto a quem pode se tornar membro e a permissões de ações de cada nó. Nesse tipo de rede, há uma configuração intrínseca que atribui papéis (*roles*) aos membros. A cada papel são atribuídas diferentes autorizações específicas, como escrever informações na *Blockchain* e aprovar a admissão de novos membros [28]. Já *Blockchains* não-permissionadas, ou públicas, são caracterizadas por permitirem a participação de qualquer um na rede. O registro de transações é compartilhado por todos os participantes e é atualizado por mineradores [28] por meio de mecanismos de consenso.

Pela sua natureza mais restrita e isolada, *Blockchains* permissionadas são mais adequadas para aplicações que precisem de privacidade e confidencialidade. Em uma rede não-permissionada, obter esse mesmo efeito exige técnicas criptográficas razoavelmente avançadas, devido ao fato de que todos os nós armazenam o mesmo *ledger* e, assim, possuem acesso aos mesmos registros de transações [23].

Em relação a mecanismos de consenso, as *Blockchains* não-permissionadas mais populares, como o *Bitcoin* e o *Ethereum*, costumam adotar mecanismos que dependem de mineração como o PoW. Embora essas abordagens proporcionem um grau de segurança interessante, elas requerem grande esforço computacional por parte dos nós mineradores. Isso acaba sendo um fator limitante para a escalabilidade da rede devido à baixa taxa temporal de gravação de transações na rede. As *Blockchains* permissionadas, por sua vez, são mais escaláveis pois permitem um número restrito de nós validadores, o que reduz o custo computacional da rede. Além disso, há uma flexibilidade maior na escolha mecanismos de consenso, o que permite a escolha de algoritmos menos exigentes computacionalmente e mais baseados em confiança nos nós validadores [29].

2.2.4 *Tokens* Fungíveis e Não-fungíveis

Ativos criptográficos têm uma série de características, incluindo se um ativo criptográfico é ou não fungível. A fungibilidade é uma característica de um ativo que de-

termina se itens ou quantidades do mesmo tipo são intercambiáveis e de igual valor quando transferidos ou utilizados. Se forem iguais ou similares, eles são *Tokens* Fungíveis (FTs) [30]. Já um Token Não-Fungível (*NFT*) é uma unidade de dados armazenados em uma *Blockchain* que certifica que um bem digital é único (i.e. não intercambiável), ao mesmo tempo em que oferece um certificado digital de propriedade.

Por meio de propostas *Ethereum Request for Comments* (*ERC*) são recomendados padrões e convenções, em nível de aplicação, para o desenvolvimento acerca de protocolos dos *tokens*. O padrão adotado foi o ERC-1155, que descreve a convenção para múltiplos tokens, tanto fungíveis quanto não-fungíveis [31].

3 MÉTODO DE TRABALHO

A seguir, é descrito o método empregado na condução do projeto.

3.1 Especificação de Requisitos

Primeiramente, foram levantados os requisitos funcionais e não-funcionais do sistema para melhor compreensão e delineamento da solução. Além disso, essa etapa também contribui para a identificação de possíveis restrições na escolha da *Blockchain* e das demais tecnologias utilizadas no projeto.

3.2 Levantamento de Casos de Uso

Primeiramente, foi realizado a identificação e documentação dos casos de uso. Foram elaborados um total de três documentos, um apresentando os casos de uso de usuários comuns, outro para os casos de uso de administradores da Carbon21 e, por fim, um para funcionários do IBAMA e da CETESB. Para cada caso de uso foram listadas pré-condições, passos necessários para cada cenário e pós condições

Em seguida, para melhor compreensão do comportamento do programa, foram desenvolvidos diagramas com base nos casos de uso levantados.

3.3 Definição do MVP

A próxima etapa foi a definição de um MVP (*Minimum Viable Product* - produto mínimo viável) para fins de prova de conceito. Estudaram-se os casos de uso inicialmente aplicáveis à arquitetura projetada para baterias de testes e simulação do funcionamento, por meio da utilização de *containers* que emulam a execução de nós da *blockchain*.

3.4 Escolha de Tecnologias

A próxima etapa consistiu na escolha das tecnologias que foram utilizadas para a implementação do projeto. A principal dessas escolhas foi a da *Blockchain*, cujo processo de escolha foi dividido em:

1. Pesquisa de *Blockchains*: foram pesquisadas e analisadas as principais plataformas *Blockchain* existentes atualmente;
2. Análise comparativa: as *Blockchains* pesquisadas que melhor se adequaram aos requisitos do sistema foram analisadas comparativamente entre si com base em diferentes critérios;
3. Escolha da *Blockchain* com base na análise feita na etapa anterior.

3.5 Implementação

Definido o MVP e as principais tecnologias, partiu-se para a etapa de implementação em uma linguagem de programação escolhida dentre as possibilidades permitidas pela *Blockchain* escolhida, com o auxílio de possíveis *frameworks*.

3.6 Testes e Validação

Após a implementação, realizaram-se testes de validação do sistema desenvolvido.

4 ESPECIFICAÇÃO DE REQUISITOS

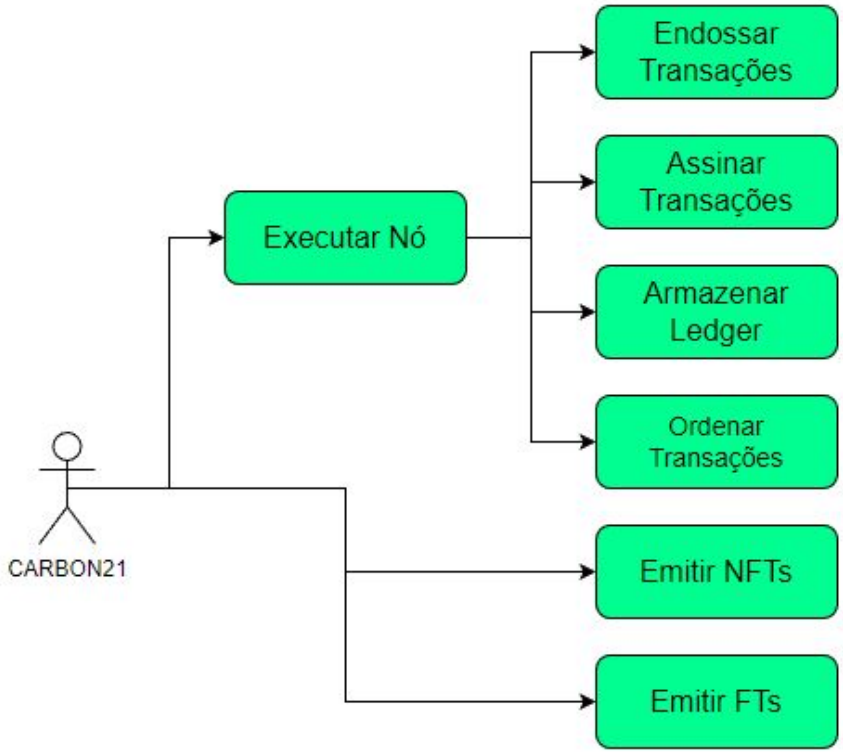
O presente trabalho consiste no desenvolvimento de um sistema, a saber, uma plataforma de tokenização de ações de reflorestamento e preservação ambiental. Assim, é de fundamental importância que, na etapa inicial do projeto, sejam levantados os seus requisitos funcionais e não-funcionais que nortearão todo o processo. Assim, no presente capítulo, são levantados, de forma mais abrangente, os requisitos identificados para a plataforma pensada.

4.1 Definição de Atores

O projeto conta com diferentes atores, cada um com suas características e funcionalidades particulares. Os principais perfis identificados na etapa conceitual do projeto são:

- *Carbon21*: são os principais responsáveis pela implementação e gestão da plataforma, realizando, também, sua divulgação e promovendo sua adoção por terceiros. A Carbon21 é, ainda, responsável pela manutenção do sistema, executando nós do *Blockchain* por meio dos quais endossam, assinam e ordenam transações. Por fim, é o ator responsável pela emissão de Sylvas e de *NFTs* na plataforma.

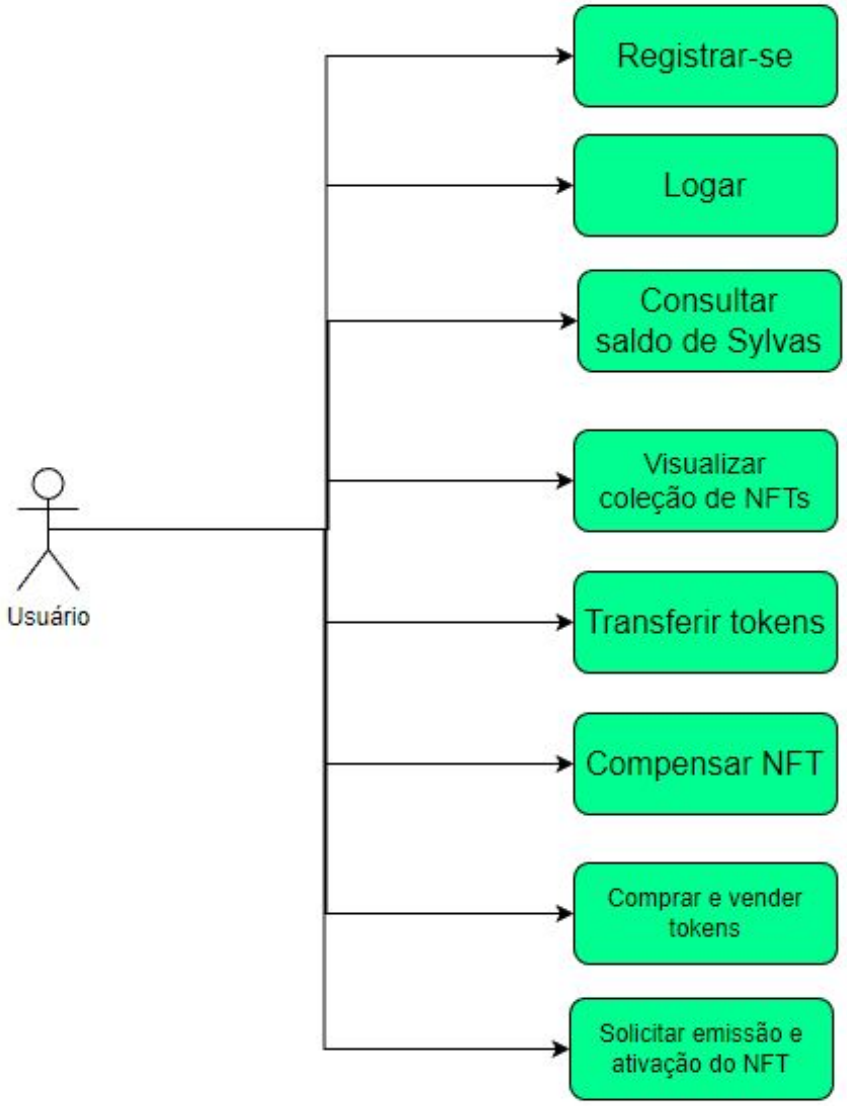
Figura 6: Ator Carbon21



Fonte: Elaboração própria

- Usuários: podem assumir diferentes papéis no sistema. Podem ser proprietários de terras que disponibilizam suas terras para reflorestamento e solicitam a emissão e ativação dos *NFTs*. Podem, ainda, assumir papel de investidores que adquirem e negociam os *tokens* da plataforma, esperando que estes valorizem futuramente. Consultas ao saldo de Sylvas e de *NFTs* são também interações que esses atores tem com a plataforma.

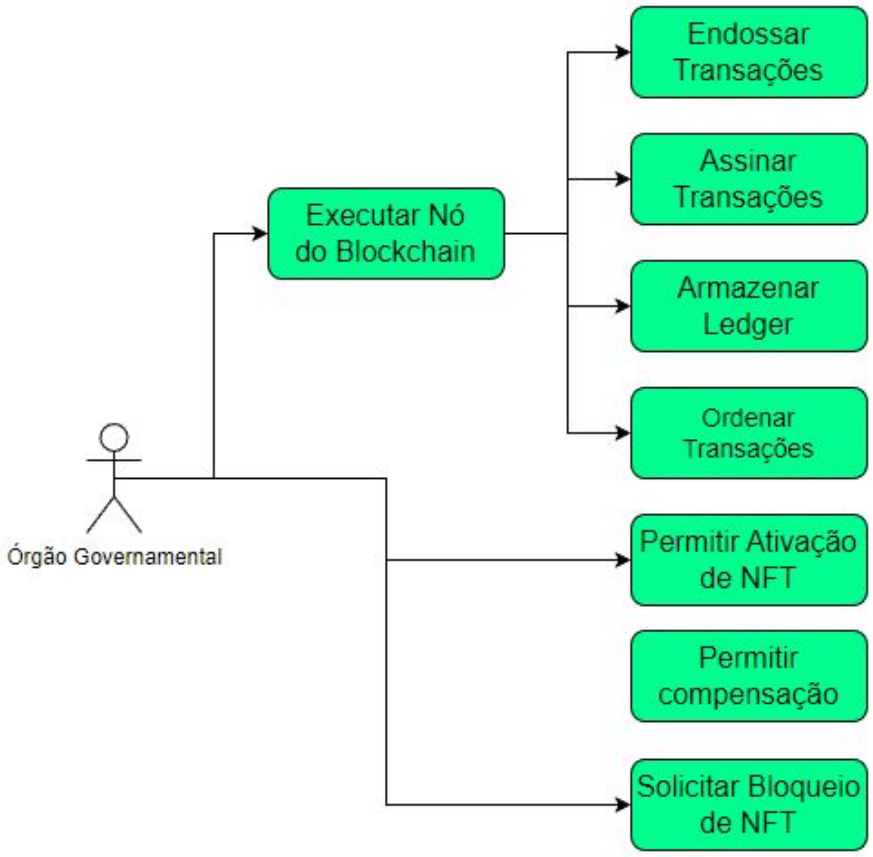
Figura 7: Ator usuário



Fonte: Elaboração própria

- Órgãos Governamentais: possuem duas principais áreas de atuação. A primeira é a validação de documentos que comprovem informações importantes para a integridade e legalidade do projeto, tais como Cotas de Reserva Ambiental (CRA), Escrituras, entre outros. Para isso cogita-se parcerias com órgão como IBAMA E CETESB. A segunda área de atuação é a de monitoramento do estado das propriedades, verificando, por exemplo, se não houve desmatamento de uma área *tokenizada*. Para isso, cogita-se parcerias com órgão como Embrapa e INPE. Caso seja de seu interesse, os órgãos governamentais também podem armazenar cópia do *Blockchain* e participar do endosso e ordenação de transações.

Figura 8: Ator órgão governamental



Fonte: Elaboração própria

4.2 Requisitos Funcionais

Os requisitos funcionais identificados são:

4.2.1 Emissão de *NFT*

O usuário que seja dono de uma terra elegível para ser *tokenizada* deve apresentar a documentação necessária acerca da área a ser reflorestada e solicitar a emissão de um *NFT*, o qual será inicialmente emitido com *status* de inativo.

4.2.2 Ativação e Bloqueio do *NFT*

Um *NFTs*, uma vez solicitada a ativação pelo seu detentor, só pode ser ativado pela Carbon21 após permissão emitida por um órgão ambiental responsável, que deverá veri-

ficar e validar toda a documentação apresentada da propriedade de terra em questão.

O sistema, caso seja detectada e comprovada infração das regras estipuladas pela plataforma como, por exemplo, o corte de árvores, deve contar com meios de bloquear o *NFT* ativo em questão. Um *NFT* bloqueado para de gerar dividendos periódicos, isto é, Sylvas e seu uso para compensação ambiental também é impossibilitado.

4.2.3 Funcionalidades do *NFT*

Quando ativo, um *NFT* deve render Sylvas, periodicamente, ao seu detentor. Além disso, os usuários da plataforma podem listar seus *NFTs* para venda e comprar *NFTs* listados de outros usuários dentro do *marketplace* da plataforma.

Cada *NFT* deve ter um direito de compensação a ele vinculado o qual pode ser utilizado para fins de compensação ambiental. Diferentemente do rendimento periódico de Sylvas, o uso para compensação ambiental deve ser único, isto é, uma vez compensado o usuário não poderá utilizar novamente do direito.

Tal como para ativação, o uso do direito de compensação de determinado *NFT* exige permissão de órgãos ambientais que verificarão, por exemplo, se a fitofisiologia da área em questão corresponde com a da área desmatada.

4.2.4 *Marketplace*

Deve existir, como já citado, um ambiente em que os usuários interessados possam ofertar e comprar *NFTs* e direitos de compensação.

4.2.5 Sistema Auditável

O sistema deve ser auditável, permitindo ao público a verificação da integridade do registro de transações, conferindo à plataforma transparência e rastreabilidade dos *tokens*.

Além do sistema ser auditável, é de particular interesse que esses auditores não tenham que executar nós completos da *Blockchain*, isto é, que não precisem manter cópias completas da mesma, tendo acesso somente ao necessário para verificação e acompanhamento de sua integridade.

4.2.6 Legislação Local

O processo de compensação florestal deve estar sujeito às legislações ambientais locais vigentes, levando-se em consideração sempre o cenário mais restritivo em caso de divergências entre regulamentos de diferentes órgãos governamentais;

4.3 Requisitos Não-Funcionais

Como requisitos não-funcionais, têm-se:

4.3.1 Escalabilidade

Visando escalabilidade para possivelmente atender dimensões nacionais no futuro, o sistema deve ser capaz de processar, sem prejuízo às suas operações, um volume de transações da ordem de milhares por segundo.

4.3.2 Padrões de *Tokens*

O sistema deverá utilizar, sempre que possível, padrões já estabelecidos que facilitem a integração com outras plataformas e soluções existentes. Entre os padrões mais utilizados atualmente estão o ERC-20 para *tokens* fungíveis, o ERC-721 para *tokens* não fungíveis e o ERC-1155 que suporta os dois tipos de *tokens*.

4.3.3 Integração

A princípio, é conveniente que haja integração, pelo menos, com a rede *Ethereum*, tendo em vista que um dos principais pontos do projeto é criar interesse em nossos *tokens* e a rede em questão é uma das mais populares voltadas à tokenização;

5 DESENVOLVIMENTO DO TRABALHO

Este capítulo é destinado a introduzir e explicar as tecnologias centrais utilizadas no projeto, discutir o processo de implementação da solução proposta e seus decorrentes resultados.

5.1 Tecnologias Utilizadas

A seguir discorre-se a respeito das principais tecnologias utilizadas no processo de desenvolvimento, sendo detalhados os principais aspectos de seu funcionamento e sua utilidade dentro do projeto.

5.1.1 *Hyperledger Fabric*

A plataforma escolhida para construção da *Blockchain* foi o *Hyperledger Fabric*. Trata-se de uma plataforma *open-source* de tecnologia distribuída de *ledgers* (DLT - *Distributed Ledger Technology*) permissionada e voltada para aplicações empresariais desenvolvida pela *Linux Foundation*. O *Fabric* conta com uma arquitetura modular e configurável, o que confere grande versatilidade para amplos casos de uso nos mais diversos setores comerciais.

Em alto nível, a plataforma é composta pelos seguintes componentes modulares [23]:

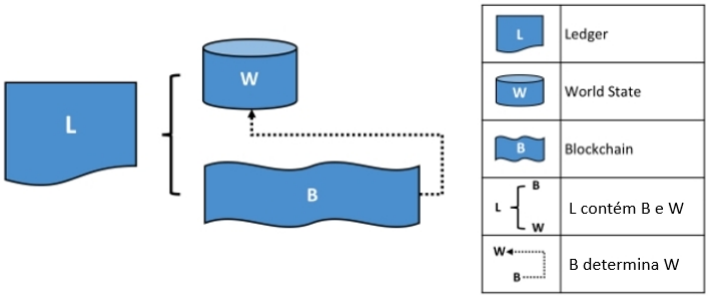
1. *Pluggable Ordering Service*: responsável por estabelecer o consenso na ordem das transações e realizar a transmissão dos blocos aos *peers*;
2. *Pluggable Membership Service Provider* (MSP): responsável por associar as entidades participantes da rede a identidades criptográficas;
3. *Pluggable Endorsement and Validation Policy*: responsável pela configuração das políticas de endosso e validação das transações, podendo ser configuradas de maneira independente de acordo com a aplicação;

4. *Chaincodes*: contratos inteligentes escritos em linguagens de programação de propósito geral como *Go, Java e JavaScript*.

5. *Peer-to-peer gossip service*: componente opcional que dissemina os blocos do *ordering service* para outros *peers*.

6. *Ledger*: é composta de um *World State*, isto é, uma *database* que contém o valor atual de um conjunto de atributos que descrevem o estado da *ledger* em determinado momento. Sendo composta também por uma *Blockchain*, que registra as transações e as decorrentes mudanças que levaram ao atual *world state*.

Figura 9: Ledger no Hyperledger Fabric



Fonte: adaptado de [23]

Além desses componentes, o *Hyperledger Fabric* utiliza mecanismos de consenso plugáveis, por padrão o Raft, um mecanismo CFT de consenso que será detalhado em capítulo posterior.

5.1.1.1 Nós e Canais

No *Hyperledger Fabric*, os nós são dispostos em organizações e podem ser classificados em três principais tipos de acordo com suas funções [32]:

1. *Clients*: submetem as propostas de transações a serem executadas e enviam as transações para os nós ordenadores;

2. *Peers*: responsáveis por executar propostas e validar transações. Cada *peer* mantém para si uma cópia da *ledger*, mas nem todos executam todas as propostas de transações, apenas um grupo de *peers* definidos por uma política pré-definida, os denominados *endorsing peers*, irão executá-las endossando-as ou não.

3. *Ordering Service Nodes* (OSN): são os nós que compõem o *ordering service*, sendo responsáveis por ordenar as transações e gerar novos blocos.

O *Fabric* possibilita que organizações participem, simultaneamente, de múltiplas *Blockchains* conectadas por um mesmo *Ordering Service* utilizando uma arquitetura de canais. Cada um dos canais pode ser encarado como uma sub-rede independente, em que apenas seus participantes têm acesso ao *Chaincode* e aos dados das transações, o que lhe confere uma maior privacidade e confidencialidade [23].

Cada organização deve contar com ao menos um *anchor peer* que será o responsável por receber os blocos dos nós ordenadores e enviá-los para os demais *peers* de sua organização.

5.1.1.2 Mecanismo de Consenso *Raft*

O *Raft* é um mecanismo *CFT* de consenso distribuído assimétrico em que os nós ordenadores podem assumir três diferentes funções: candidato, líder ou seguidor.

Todos os nós inicialmente são seguidores e possuem uma FSM (*finite-state machine*) cujas entradas serão comandos descritos por um *log*. Cada um dos nós seguidores também possui um *election timeout* aleatório que varia de 150 ms a 300 ms. Quando esse *timeout* espira e nenhum líder é escutado, o nó seguidor torna-se um candidato, inciando-se um novo processo de eleição. Incia-se também um novo *election term*, que são os períodos nos quais algoritmo divide o tempo, sendo cada um deles iniciado por um novo processo de eleição.

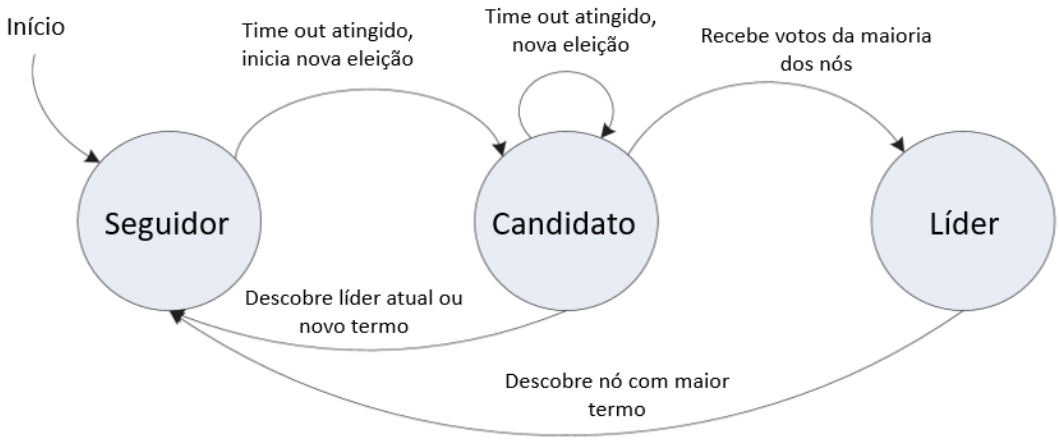
O nó candidato, então, vota em si mesmo e envia um *vote request* para os outros membros do *cluster*, que, se ainda não tiverem votado no atual *election term*, irão responder com seu voto ao candidato.

Se o nó candidato receber a maioria dos votos, ele se torna um líder, passando a ser responsável por receber novas entradas do *log*, inicialmente em *uncommitted state*, e replicá-las para os nós seguidores por meio de *append entries messages*, que são enviadas em intervalos determinados pelos chamados *heartbeat timeouts*.

O líder então espera que a maioria dos nós tenham recebido a nova entrada do *log* para que, então, esta entre em *committed state* e atualize-se o estado do líder que, por sua vez, irá enviar notificações para que os demais também atualizem seu estado. Dessa maneira o *cluster* chega a um consenso do estado e assim que os seguidores deixarem de

escutar os *hearbeats* do líder e o *election timeout* de um dos candidatos espirar, este se tornará um candidato dando-se início a um novo *election term* [33].

Figura 10: Funcionamento do Raft



Fonte: adaptado de [34]

5.1.1.3 Fluxo de Transações

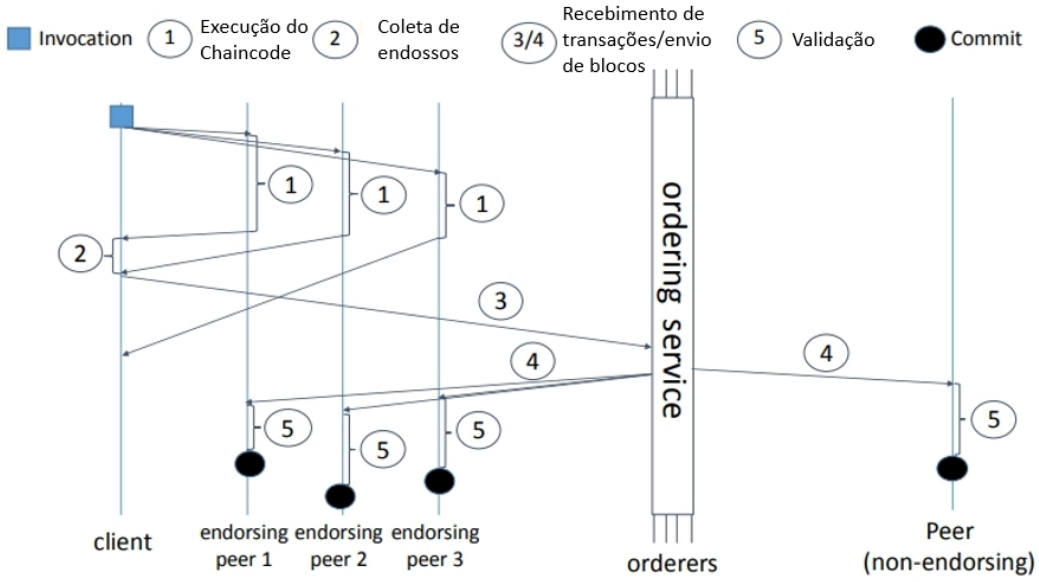
O fluxo de transações do Hyperledger Fabric pode ser dividido em três principais fases, sendo elas [32]:

1. Fase de execução: os clientes enviam propostas de transações assinadas para os *endorsing peers* que, por sua vez, simulam, isto é, executam as propostas a partir de seu estado local da *Blockchain* sem que as modificações persistam na *ledger*. Os *endorsers*, então, geram uma resposta assinada, o seu endosso, contendo um *readset* e um *writeset* e enviam para o cliente;

2. Fase de ordenação: uma vez coletados os endossos necessários definidos previamente pela política de endosso, os clientes agora enviam as transações endossadas para os nós ordenadores. Essas transações são ordenadas seguindo um mecanismo de consenso, no caso do Fabric, um Raft e então são gerados novos blocos. Por fim, esses blocos são enviados aos *anchor peers*;

3. Fase de validação: esta fase pode ser subdivida em três principais etapas. As duas primeiras consistem de novas verificações, sendo uma validação de endosso e uma checagem de possíveis conflitos no *read-write set*. Por fim, a terceira etapa consiste na atualização da *ledger* caso as transações sejam consideradas válidas.

Figura 11: Fluxo de Transações no *Hyperledger Fabric*



Fonte: adaptado de [32]

5.1.2 MySQL

O projeto conta com o suporte de um banco de dados relacional e Spara isso, utilizou-se do MySQL. Trata-se de um dos mais populares sistemas de gerenciamento de banco de dados (*SGBD*) relacional *open-source*, desenvolvido pela *Oracle Corporation* para lidar com ambientes de produção com alta demanda de maneira veloz, confiável e escalável [35].

5.1.3 Docker

O Docker é uma tecnologia de conteinerização, i.e, permite que softwares e aplicações completas sejam empacotadas com suas bibliotecas, *frameworks* e outras dependências em ambientes virtuais isolados. Neste sentido, esses ambientes, chamados de contêineres, funcionam de maneira similar às chamadas máquinas virtuais (*VMs*).

Contudo, ao contrário das *VMs* que executam um sistema operacional completo, incluindo seu kernel, os contêineres tiram proveito do compartilhamento do *kernel* do sistema operacional da máquina hospedeira, necessitando, consequentemente, de menos recursos computacionais.

Além disso, os contêineres do *Docker* apresentam uma outra vantagem com relação às *VMs*. Enquanto o tamanho das *VMs* está, tipicamente, na ordem de gigabytes, o tamanho dos contêineres estão, em sua maioria, na ordem dos megabytes, conferindo uma

maior escalabilidade ao projeto [36].

5.1.4 IPFS

O IPFS (Sistema de Arquivos Interplanetário) pode ser descrito como um protocolo descentralizado *peer-to-peer* para armazenar conteúdo - como dados, *websites*, arquivos e aplicações - bem como acessar esses dados de forma descentralizada [37]. Pode ser usado para várias finalidades diferentes, como eliminar os problemas de censura, ponto único de falha ou armazenamento distribuído.

As informações geralmente são organizadas na Internet através de locais, i.e. endereços, que disponibilizam dados de um servidor, por exemplo por meio de uma URL. O IPFS, por outro lado, utiliza "endereçamento baseado em conteúdo", onde os dados estão localizados com base no conteúdo em si por meio de um identificador exclusivo ou CID (*Content ID*), chamado *hash* (que é calculado sobre esse conteúdo).

Ao armazenar arquivos no IPFS, o sistema cria um objeto IPFS, que são limitados na quantidade de espaço que podem ocupar, definido em 256 KBs. Para arquivos maiores que 256 KBs é criado um objeto que referencia os demais que armazenam o conteúdo. Para encontrar dados dentro da rede, o IPFS utiliza Tabelas de *Hash* Distribuídas (DHTs). Uma tabela *hash* é uma estrutura que mapeia chaves para diferentes valores e esta tabela é distribuída pelos nós da rede, que após localizar o arquivo determinam quais pares o hospedam.

5.1.5 *Node.js*

O Node.js é um ambiente que permite execução de códigos *Javascript* do lado do servidor, isto é, independente de um navegador *web*. Possuindo uma arquitetura assíncrona e orientada por eventos tem como principal diferencial sua escalabilidade. [38]

Ao contrário do que ocorre em tecnologias como Java e *PHP*, o *Node.js* as requisições são tratadas em uma única *thread* de maneira assíncrona e não bloqueante, isto é, novas requisições não entram em uma fila de espera até que as operações de entrada e saída sejam resolvidas. Isso permite que o Node.js poupe recursos computacionais que seriam necessários para realização de *multi-threading* como, também, continua conseguindo tratar de requisições concorrentes.

O Node.js conta, ainda, com um gerenciador de pacotes, o *Node Packge Manager* (*NPM*), cuja disponibilidade de diversos *softwares* reutilizáveis e *frameworks* lhe confere

grande flexibilidade. Entre os principais *frameworks* utilizados atualmente, e que foi incorporado no projeto de formatura, está o *Express*, que facilita o desenvolvimento completo de aplicações *web* evitando que o programador tenha que trabalhar em baixo nível.

5.2 Projeto e Implementação

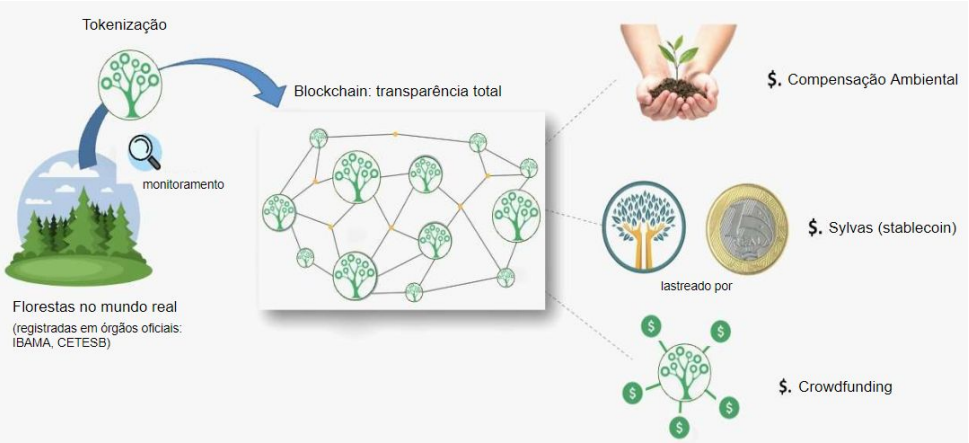
Neste capítulo discorre-se acerca das etapas de desenvolvimento e são apresentados os resultados obtidos ao longo dos trabalhos.

5.2.1 Modelagem do Modelo de Negócio

O principal produto da plataforma são os *NFTs* gerados a partir das propriedades de terra. Usuários detentores dos *NFTs*, não somente recebem periodicamente Sylvas, como também podem utilizar o direto de compensação a eles vinculados.

Uma das primeiras etapas do projeto, anterior ao início da implementação de fato, foi o desenvolvimento de um modelo de negócio, o que incluiu analisar o valor de uso dos *FTs* gerados, definir um ciclo de vida para os *NFTs* da plataforma e definir quais metadados o mesmo conteria.

Figura 12: Modelo de negócio da Carbon21



Fonte: Elaboração própria

5.2.1.1 Valor de uso dos Sylvas

O valor de uso dos *FTs* da plataforma, também chamados de Sylvas, está representado nos seguintes benefícios:

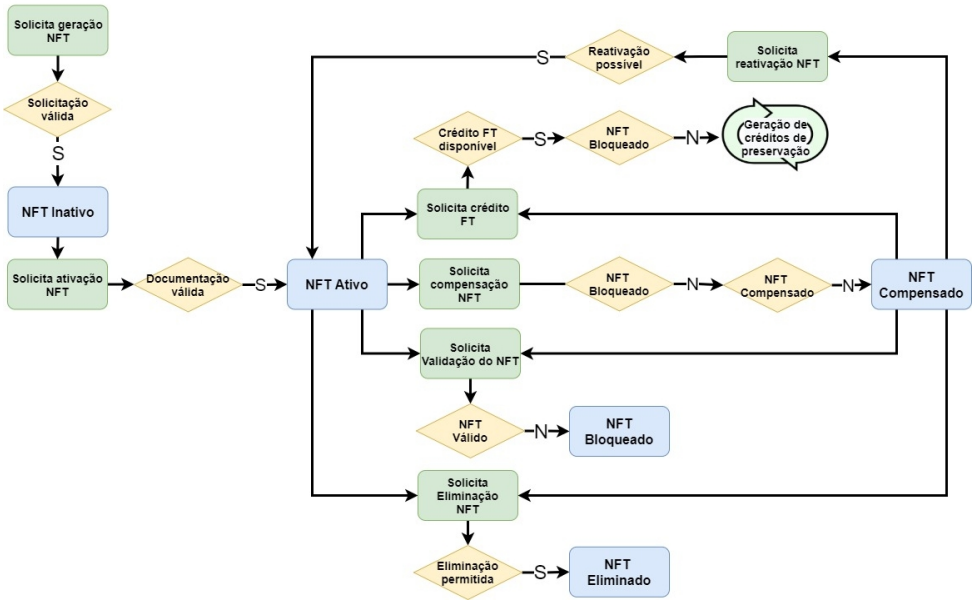
- Pagamento de contas: em um primeiro momento, por meio de parcerias com órgão públicos e entidades privadas interessadas na solução, os Sylvas podem ser aceitos para pagamento de contas como, por exemplo, contas de luz e de água ou mesmo para pagamento de impostos como IPTU rural. Sua aceitação por esses órgãos é essencial para que os Sylvas adquiram liquidez e consequentemente se crie um mercado de interessados.

- Pagamento de mercadorias: conforme aumentar aceitação dos Sylvas por esses órgãos públicos e estes adquirirem liquidez , acredita-se que os FTs da plataforma passem a ser aceitos, também, no pagamento de produtos e serviços.

- Investimento: a liquidez dos Sylvas garante que estes consigam ser convertidos pela *Carbon21* em moeda fiduciária. Com isso, adquirir os *FTs* e *NFTs* da plataforma passa a ser enxergado como uma forma de investimento.

5.2.1.2 Ciclo de Vida do NFT

Figura 13: Ciclo de vida do NFT Carbon21



Fonte: Elaboração própria

Uma vez solicitada a geração do NFT e enviados os documentos necessários pelo dono da terra por meio da plataforma, o *token* é gerado inicialmente em estado "Inativo", isto é, ainda sem render periodicamente FTs e sem a possibilidade de uso para compensação.

O dono do NFT, então, pode solicitar sua ativação e os documentos enviados na etapa anterior passarão a ser verificados por órgãos como a CETESB ou pelo IBAMA.

Validados os documentos, o NFT passa para o estado “Ativo” em que o usuário poderá solicitar início da geração de créditos FTs tal como solicitar sua compensação mediante permissão pelos mesmos órgãos acima citados. Uma vez que o direito de compensação atrelado a determinado NFT é utilizado, este não poderá ser utilizado novamente a menos que seja reativado como área de sub-bosque.

Caso seja comprovada alguma infração por parte do proprietário das terras, por exemplo, o desmatamento da área reflorestada, o NFT deve prontamente passar para o estado “Bloqueado” , cessando-se a geração de FTs e bloqueando seu uso para compensação se ainda não utilizado. Mesmo bloqueado o usuário pode, futuramente, solicitar seu desbloqueio e o NFT volta para seu estado de “Ativo”.

Caso seja do interesse do proprietário do NFT, este pode listá-lo para venda no *marketplace*, havendo, ainda, a possibilidade de venda à parte de seu direito de compensação.

5.2.1.3 Metadados dos NFTs

O *Schema* de metadados utilizado foi definido com base no Padrão EIP-1155 [39], visando também a compatibilidade com aplicações existentes de carteiras. Os parâmetros customizados foram escolhidos de forma a promover a lógica de negócio, compondo informações relevantes à investidores e demais *stakeholders*. Os parâmetros definidos inicialmente, estão apresentados na Tabela 1:

Tabela 1: Metadados dos NFTs

Atributo	Descrição
verifier	Órgão governamental que liberou a ativação do NFT
private_verifier	Indicação de outro órgão ou sistema que possui mais informações da terra
id	Id interno do NFT no sistema
land_owner	Dono da terra
compensation_owner	Detentor do direito de compensação
land	Área da terra (CRA, GPS... em aberto)
area_classification	Art. 46 Cod. Florestal
biome	Bioma/fitofisiologia
geolocation	Coordenadas da área
amount	Quantidade associada ao NFT
value	Valor do NFT listado para venda
sylvas_minted	Quantidade de FTs gerados
status	Inativo, Ativo, Bloqueado
compensation_state	Compensado, não-compensado
certificate	Certificado emitido pelo órgão governamental permitido
bonus_ft	Geração ou não de FTs bônus por sub-bosque
minter	Entidade que realizou a emissão do NFT (Carbon21)
queue	Lugar na fila de direito de gerar FTs
can_mint_sylvas	Booleano referente ao direito de gerar FTs
nft_type	Tipo de NFT (preservação / corte)
custom_notes	Anotações

Fonte: Elaboração própria

5.2.1.4 Remuneração do Sistema

A remuneração do sistema ocorre das seguintes maneiras:

1. Parte do rendimento anual de um *NFT* ativo, isto é, uma porcentagem fixa dos

Sylvas gerados anualmente por ele, são destinados ao sistema a título de remuneração e não ao seu proprietário.

2. Ao se realizar a venda de um *NFT*, parte do valor cobrado será utilizado de remuneração ao sistema, não sendo, o valor da venda, integralmente repassado ao vendedor.

3. Ao se realizar a compra de Sylvas na plataforma, parte dos Sylvas comprados é destinado ao sistema na forma de taxa.

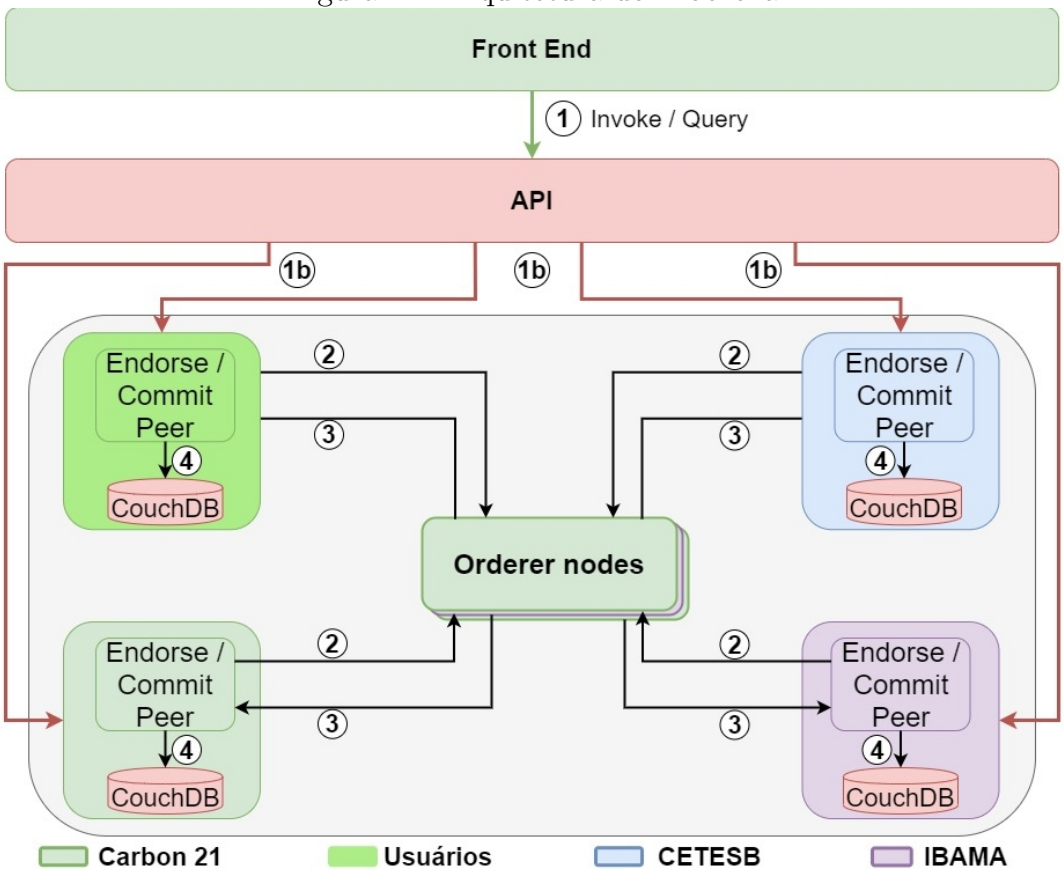
Atividades como a listagem de NFTs no *marketplace*, a princípio, são livres de taxações como forma de garantir a liquidez do *tokens* da plataforma.

5.2.2 Arquitetura

A arquitetura da plataforma conta com um *Front-End*, responsável pelo website da plataforma desenvolvido em *HTML*, *CSS* e *Javascript*. O *website* é acessado pelos usuários e, por meio dele, se cadastram e conseguem se comunicar com a *Blockchain*, realizando *invokes* e *queries*, por meio de uma API desenvolvida com NodeJS.

Com relação ao *Back-End*, o projeto conta com uma *Blockchain* permissionada funcional mantida por nós distribuídos em quatro diferentes organizações. São elas Carbon21, CETESB, IBAMA e uma organização de Usuários também mantida pela Carbon21. Cada organização possui um nó de endosso e *commit*, uma cópia da *ledger*, isto é, da *Blockchain* e da *database* (CouchDB) que armazena o *World State*, além de contar uma Autoridade Certificadora (CA). As organizações se comunicam entre si por meio de um canal único. Além dos quatro nós já citados, a arquitetura da aplicação, hoje, conta com um total de três nós ordenadores que participarão ativamente do *Raft*. Cada um dos sete nós foram implementados em contêineres isolados do *Docker*.

Figura 14: Arquitetura do Blockchain

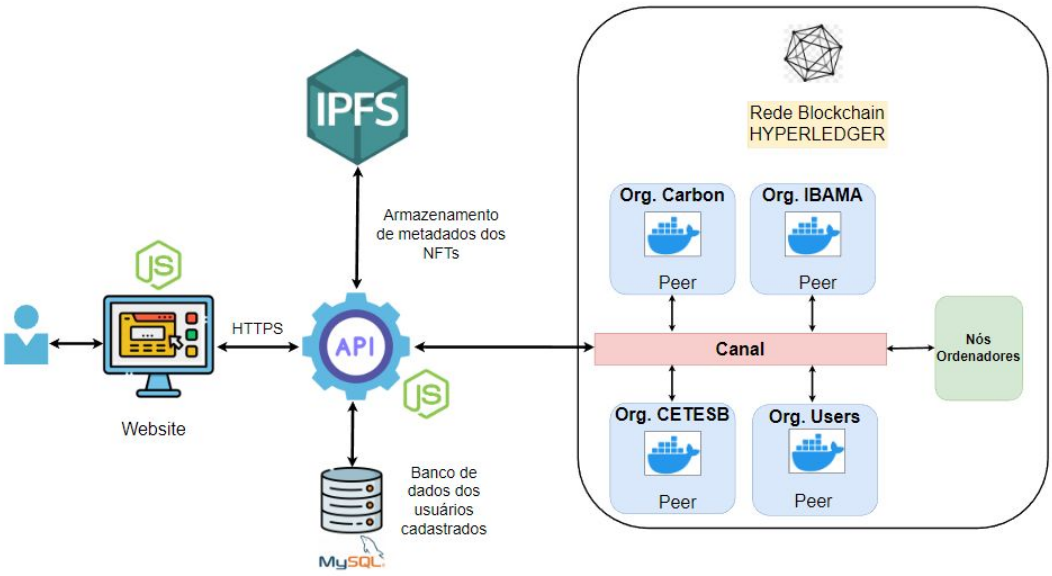


Fonte: Elaboração própria

Algumas informações de usuário, requeridas no ato do registro dentro da plataforma, não são estritamente necessárias para sua interação com a *Blockchain*, mas são necessárias para sua autenticação e são interessantes para sua identificação dentro da plataforma em um cenário permissionado. Dito isso, a arquitetura apresenta suporte de um banco de dados relacional (*MySQL*) à parte em que são armazenadas informações como o nome completo do usuário, seu CPF, seu *email*, o *hash* de sua senha de acesso a plataforma e um *seed*, utilizado para realização de *password hashing*.

Como já explicado em capítulos anteriores, os *NFTs* contam com metadados como geolocalização, estado da ativação e do direito de compensação, tipo de vegetação, representando por suaa fitofisiologia, tamanho da propriedade e dono do direito de compensação. Para o armazenamento destes metadados a solução faz uso do *IPFS*.

Figura 15: Arquitetura Completa da Solução



Fonte: Elaboração própria

5.2.3 Padrão de Token

Como base para o *Smart Contract* do projeto, ou *Chaincode*, como é chamado no contexto do *Hyperledger*, decidiu-se por adotar o padrão ERC-1155. Trata-se de um padrão que permite, a partir de um único contrato, descrever e realizar operações de múltiplos *tokens*, sendo eles fungíveis ou não fungíveis, suportando a maior parte das funcionalidades presentes tanto no ERC-20, como, também, no ERC-721.

Como um de seus principais diferenciais, além da questão de oferecer maior flexibilidade, o padrão permite operações em bateladas, ou *batch operations*, isto é, permite-se operar simultaneamente vários *tokens* em uma única transação, especificando-se seus identificadores (*IDs*).

Entre as funções implementadas pelo contrato base têm-se [31]:

1. *safeTransferFrom*: utilizada para transferir a quantidade especificada de *tokens* de um único tipo entre dois usuários.

2. *safeBatchTransferFrom*: utilizada para transferir as quantidades especificadas de variados tipos de *tokens* de um usuário para outro.

3. *balanceOf*: utilizado para consultar o saldo de determinado usuário para um único tipo de *token*

4. *balanceOfBatch*: utilizado para consultar o saldo de um ou mais usuários para múltiplos tipos de *token*

5. *setApprovalForAll*: permite que um terceiro, chamado de "operador", gerencie todos os *tokens* de uma determinada conta.

6. *isApprovedForAll*: consulta a situação de permissão de um operador para gerir os *tokens* de uma dada conta.

Além das funções citadas acima, há também algumas funções adicionais:

1. *Mint*: realiza a emissão da quantidade determinada de dado *token*. A sua versão *batch* (*mintBatch*) também está presente no contrato.

2. *Burn*: realiza a queima da quantidade especificada de determinado *token*). A sua versão *batch* (*burnBatch*) também está presente no contrato.

3. *SetURI*: permite que se defina o URI de metadados relacionados a um determinado *token*.

4. *GetURI*: retorna a URI dos metadados referentes a dado token.

5. *ClientAccountID*: utilizada para se obter o *Client ID* do usuário.

6. *ClientAccountBalance* é utilizada como um atalho para a função *balanceOf*.

5.2.4 Modificações no Chaincode

De modo a a adequar o contrato do ERC-1155 às necessidades e funcionalidades específicas do projeto, foram realizadas uma série de modificações ao contrato base.

Como já especificado, o usuário da plataforma irá receber dividendos de preservação periodicamente na forma de *FTs* caso detenha um *NFT* ativo. Para isso, foi criada uma função adicional nomeada de *FTFromNFT* que, ao ser acionada, realiza uma varredura do *World State* encontrando os *NFTs* existentes no *Blockchain* e gerando uma quantidade parametrizável de *FTs* para a conta correspondente

Além disso, percebendo-se que, apesar da possibilidade de consulta ao saldo de múltiplos tipos de *tokens* simultaneamente, ainda era necessário conhecer e fornecer na *query* a lista de *IDs* dos *tokens* cujo saldo queria se consultar. O *chaincode*, entretanto, carecia de um método capaz retornar uam lsita com os *tokens* na posse de um dado usuário. Procurando

evitar guardar uma lista de identificadores correspondentes a cada usuário no banco de dados relacional, o que poderia causar futuras divergências e conflitos com os dados guardados na própria *Blockchain*, foram criadas funções auxiliares. Essa funções tem como objetivo facilitar o fluxo de consultas e fazer com que as mesmas fossem realizadas diretamente ao *Blockchain* e ao *World State*. O seguintes métodos foram acrescentados ao *chaincode*:

1. *SelfBalanceNFT*: quando chamada retorna ao usuário devidamente registrado e autenticado a lista dos identificadores dos *NFTs* em sua posse.

2. *BalanceNFT*: para uso exclusivo de administradores do sistema, pertencentes a Carbon21. Dado o *email* de um usuário, a função retorna a lista de todos os *NFTs* do mesmo.

5.2.5 Implementação IPFS para Metadados

Neste projeto o IPFS é implementado por meio de contêiner *Docker ipfs/kubo*, que disponibiliza *gateways* e *APIs* para publicar e recuperar de dados da rede. Controladores da aplicação realizam requisição ao nó de ipfs, tratando o CDI retornado ou o objeto *JSON* contendo os metadados. A informação referente ao endereço dos metadados de cada NFT é registrado pelo chaincode de acordo com o padrão de *Universal Resource Identifier* (URI), e.g. *ipfs://QmV2gJwWbDh2E3zzVV1CRpR3prgC6fZ828Yszs9HYxmezp*.

O usuário habilitado a realizar *mint* de NFTs deve inserir nos campos apropriados dados referentes à propriedade, *stakeholders*, dentre outras informações relevantes, para que possam ser validados no *schema* de metadados e publicados na rede IPFS. Posteriormente os metadados podem ser recuperados pelos usuários finais em sua página de Coleção, que se utiliza dos identificadores de todos NFTs em sua conta para realizar uma busca na rede IPFS por meio do URI correspondente.

5.2.6 *API*

A API para comunicação com o *Back-End* da solução foi desenvolvida em Node.JS e extensivamente documentada no *Postman* do projeto. A tabela 2 apresenta as possíveis chamadas, sendo todas elas, com exceção da chamadas de registro e *login*, protegidas, sendo necessário se autenticar por meio do *jwt*.

Tabela 2: Chamadas para API

Chamada	Método	Descrição
Auth - Signup	POST	Registra usuário no sistema, retornando o jwt
Auth - Login	POST	Autentica usuário, retornando o jwt
Invoke - Mint	POST	Emite a quantidade determinada do token especificado
Invoke - MintFTFromNFT	POST	Emite Sylvas aos proprietários de NFTs
Invoke - TransferFrom	POST	Transfere certa quantidade de um dado tipo de token para outro usuário
Query - BalanceOf	GET	Retorna o saldo do usuário especificado para dado tipo de token
Query - SelfBalance	GET	Retorna o saldo do proprietário do jwt para dado tipo de token
Query - TotalSupply	GET	Retorna a quantidade existente de determinado tipo de token
Query - SelfBalanceNFT	GET	Retorna uma lista contendo todos os identificadores dos NFT na posse do proprietário do jwt
Query - ALLNFTID	GET	Retorna uma lista com todos os identificadores dos NFTs presentes no World State
Query - BalanceNFT	GET	Retorna uma lista de todos os identificadores dos NFTs na posse da conta especificada
Metadata - GetMetadata	GET	Recupera do IPFS os arquivos JSON dos metadados de determinado NFT
Metada - PostMetadata	POST	Publica no IPFS arquivos JSON contendo os metadados do NFT

Fonte: Elaboração própria

5.2.7 *Website*

A solução conta com um *website* funcional por meio do qual os usuários conseguem realizar chamadas à *API* do *Blockchain*, realizando transações, consultando saldos de *Sylvas*, visualizando sua coleção de *NFTs* ou mesmo compensando determinado *NFT*.

A parte superior de todas as telas contém uma barra de navegação pelo qual o usuário

pode acessar rapidamente as diferentes seções da página, a tela de *login* e a tela de registro. Para usuários autenticados com sucesso dão acesso às diferentes funcionalidade da plataforma.

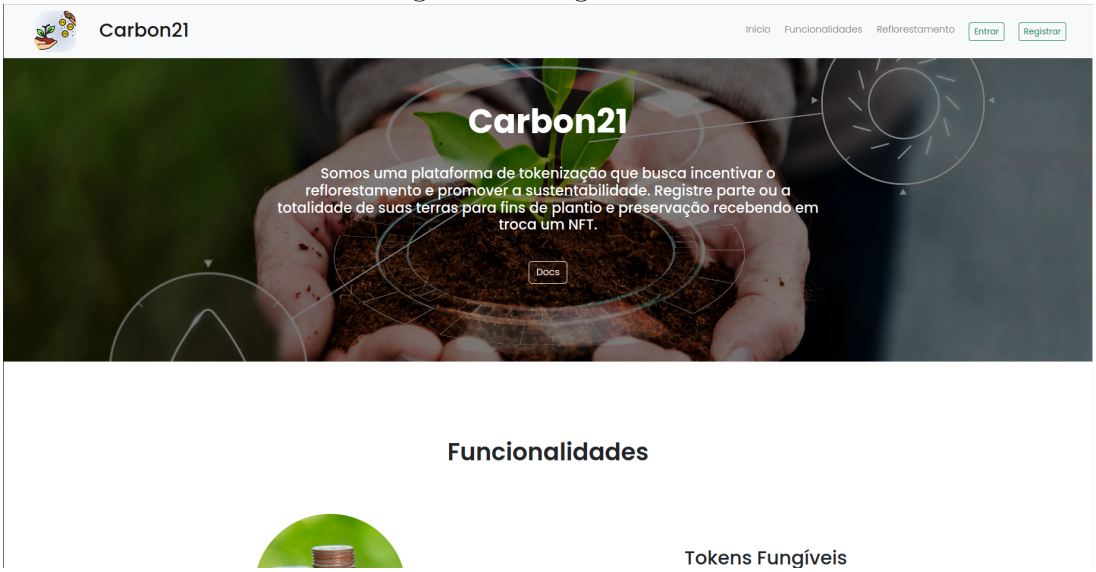
Há no *site* a diferenciação de dois tipos de usuários, sendo eles: usuários comuns e administradores da *Carbon21*. Este último possui acesso a funcionalidades reservadas, como emissão de *NFTs* e emissão de *Sylvas*.

Estas funcionalidades exclusivas para funcionários da Carbon21, apesar de ocultas na barra de navegação de um usuário comum, podem ser acessadas de maneira forçada caso o usuário conheça a *url* da página. Para isso foram criados *middlewares*, isto é, códigos que são executados entre a requisição do cliente e a execução da funcionalidade no servidor. Esses *middlewares* são responsáveis por redirecionar o atacante para a tela de login caso o mesmo tente acessar áreas não permitidas do *site*.

5.2.7.1 Página Inicial

O *website* conta com uma página inicial destinada à introdução do projeto a novos usuários, i.e, apresenta os principais objetivos da plataforma, as funcionalidades disponíveis até o presente momento, os possíveis benefícios decorrentes do reflorestamento e da preservação ambiental e as universidades envolvidas com o projeto.

Figura 16: Página inicial

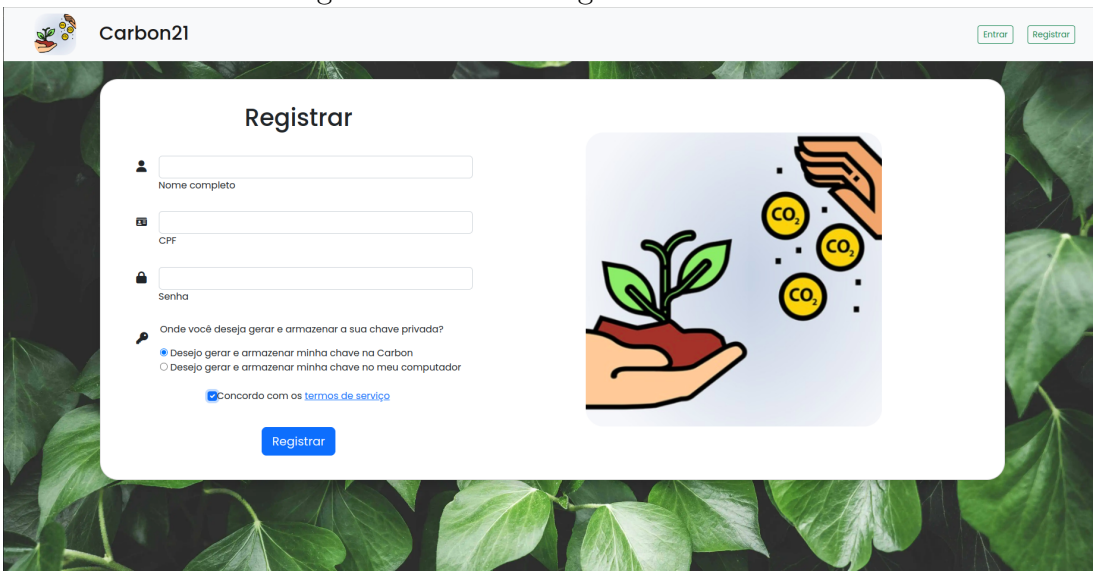


Fonte: Elaboração própria

5.2.7.2 Página de registro de usuário

Ao apertar o botão de registro da barra de navegação, o usuário tem acesso a tela de registro. Nela o usuário deve informar seu *email*, nome completo, documento de identificação (CPF) e uma senha de acesso a plataforma. Além disso, o usuário deve escolher se deseja guardar sua chave privada nos servidores da Carbon21 ou se deseja guardá-la em seu computador. Por fim, o usuário deve marcar se concordo ou não com os termos de serviço da plataforma. No ato do registro, é realizada uma verificação do *email* e do *CPF* informados no banco de dados de usuários já cadastrados. Se um novo usuário tentar se registrar com *email* ou CPF já utilizado a plataforma acusa um erro.

Figura 17: Tela de registro de usuários

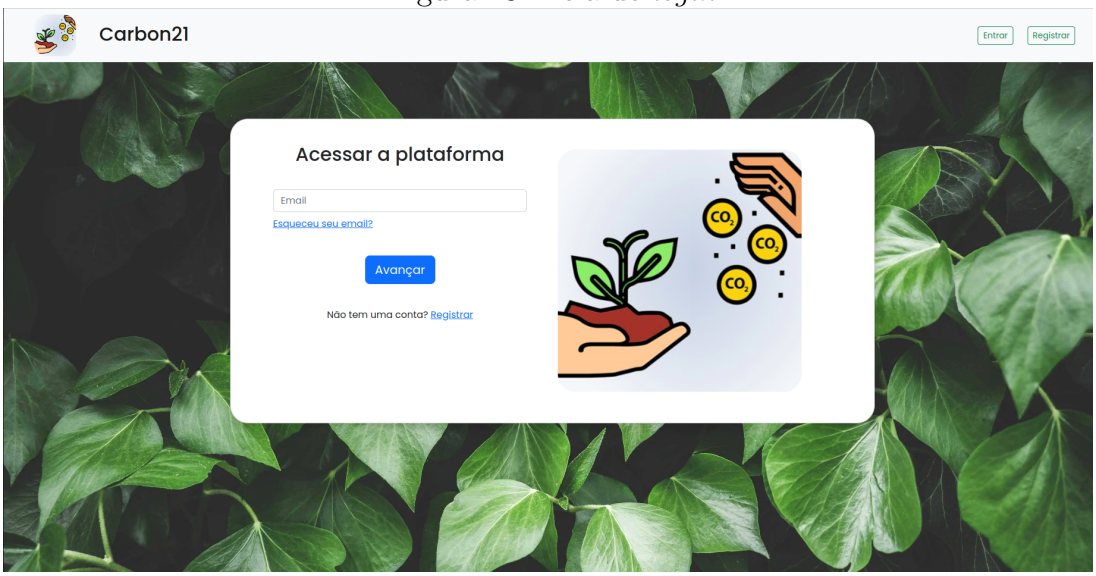


Fonte: Elaboração própria

5.2.7.3 Página de *login*

Ao apertar o botão de *login*, presente na barra de navegação, o usuário tem acesso a tela de login. Nela o usuário informa seu *email* e senha de acesso à plataforma. Caso o usuário entre com a combinação incorreta de *email* e senha, uma mensagem de erro surge na parte superior da tela. Caso o usuário consiga se autenticar corretamente, o mesmo é redirecionado à página inicial, já com acesso às funcionalidade básicas pela barra de navegação. O nome de usuário autenticado é mostrado ao lado do botão de finalizar sessão, até que o mesmo a finalize.

Figura 18: Tela de *login*

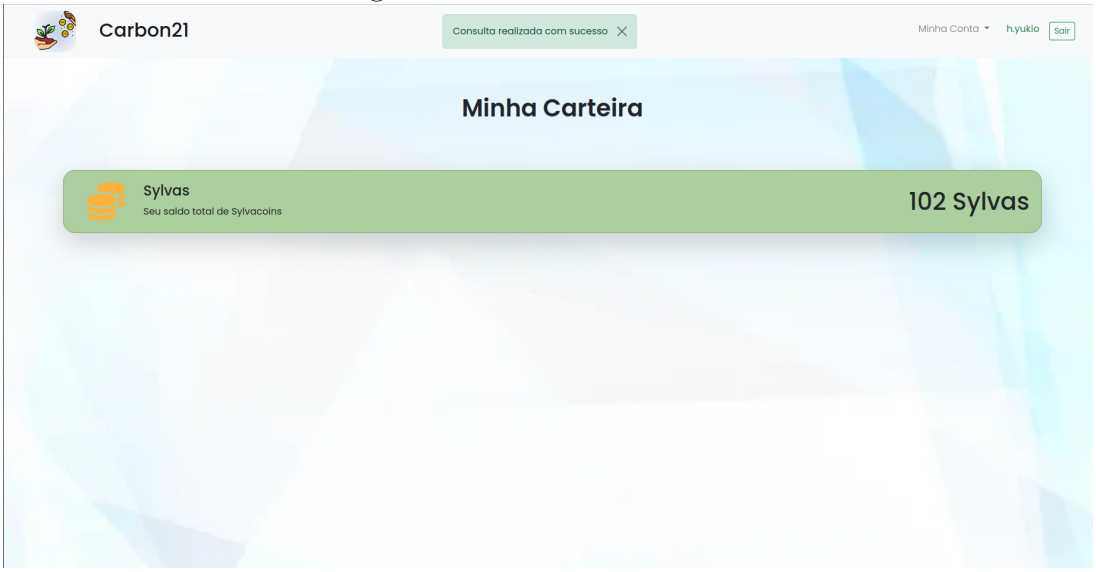


Fonte: Elaboração própria

5.2.7.4 Carteira do usuário

Um usuário devidamente autenticado em acesso a página de carteira do usuário. Esta página é destinada a visualização do saldo de Sylvas do usuário.

Figura 19: Carteira do Usuário

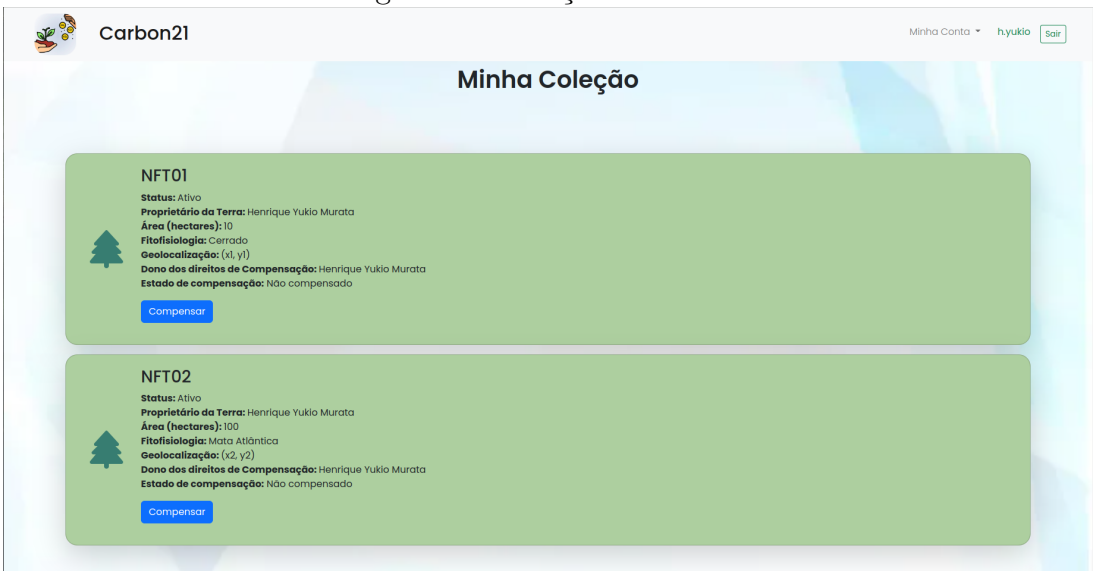


Fonte: Elaboração própria

5.2.7.5 Coleção do usuário

Ao usuário comum autenticado é permitido o acesso a página de coleção. Nela é possível visualizar a lista de todos os *NFTs* em sua posse. Também é possível expandir a visualização do *NFT* e conferir detalhes como os metadados relacionados ao mesmo.

Figura 20: Coleção do usuário



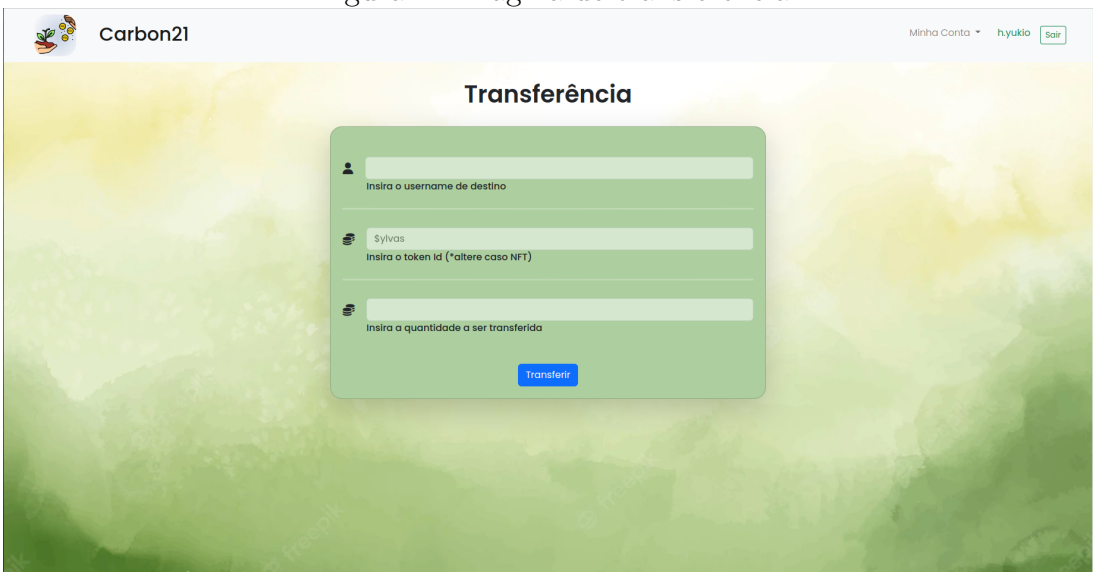
Fonte: Elaboração própria

NFTs cujo direito de compensação ainda não tenha sido utilizado, apresentam, juntamente aos seus metadados, um botão para a sua realização. Ao se compensar o *NFT*, o metadado relacionado ao estado de compensação é atualizado no *IPFS* e consequentemente na página de coleção do usuário.

5.2.7.6 Página de transferência

Qualquer usuário comum que possua saldo suficiente pode, por meio da tela de transferência, realizar a transação. Para isso é necessário que se informe o *email* do destinatário, a quantia a ser transferida e o identificador do tipo de *token* que será repassado.

Figura 21: Página de transferência



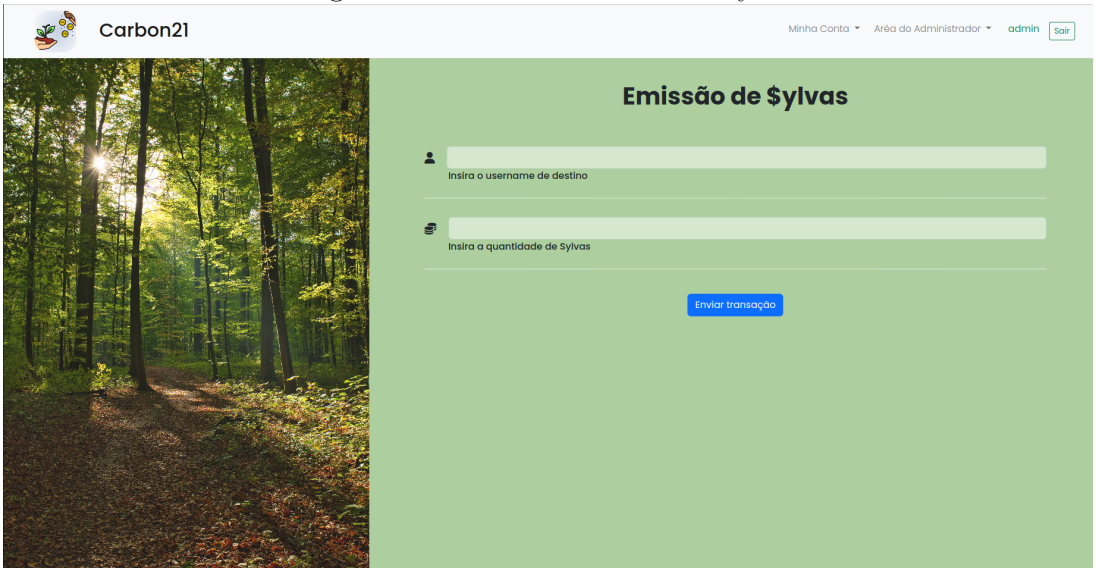
Fonte: Elaboração própria

5.2.7.7 Página de emissão de *Sylvas*

Um administrador devidamente autenticado dentro da plataforma tem acesso a tela de emissão de Sylvas. Para que operação seja realizada deve-se informar a quantia de *tokens* que se deseja emitir e o *email* do destinatário.

Com os dados informados, é realizada uma requisição à API do *Blockchain* que retorna uma mensagem informando se a operação foi bem sucedida ou não.

Figura 22: Tela de emissão de Sylvas



Fonte: Elaboração própria

5.2.7.8 Página de emissão de *NFTs*

Administradores da Carbon21 devidamente autenticados têm acesso a tela de emissão de *NFTs*. Para *tokenizar* uma propriedade de terra o administrador deve informar o *email* do destinatário, o identificador do *NFT*, o nome do proprietário da terra, o nome do proprietário do direito de compensação, a fitofisiologia da área, o tamanho em hectares da propriedade e sua localização. Não é necessário informar a quantidade do *NFT* a ser emitida, pois o mesmo deve ser único.

Assim como na tela de emissão de Sylvas, o administrador receberá um aviso indicando se operação ocorreu sem falhas.

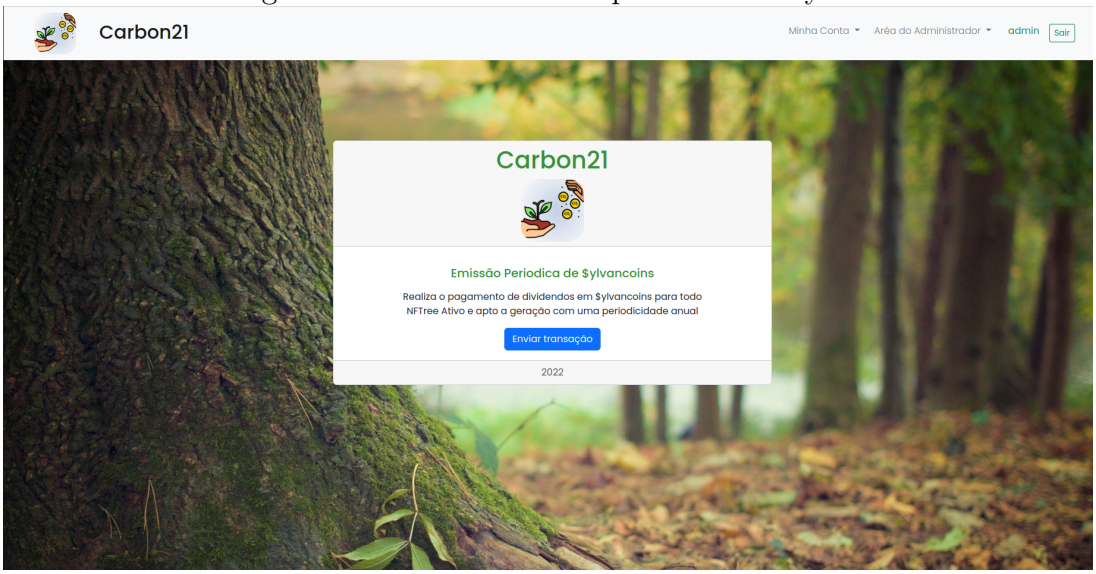
Figura 23: Tela de emissão de *NFT*

Fonte: Elaboração própria

5.2.7.9 Página de emissão periódica de Sylvas

A página de emissão periódica de Sylvas também é de acesso exclusivo de administradores da Carbon21. Nela estão contidas instruções para realização da emissão de Sylvas a partir dos *NFTs* ativos.

Figura 24: Tela de emissão periódica de Sylvas



Fonte: Elaboração própria

5.2.8 Decisões de Segurança

É importante que, já para a etapa de prova de conceito, exista uma preocupação com a segurança da plataforma e dos usuários. Algumas decisões, descritas a seguir, foram tomadas para tornar a solução segura.

5.2.8.1 Registro, Autenticação e Autorização

Como já dito nos capítulos anteriores, o projeto utiliza uma arquitetura de *Blockchain* permissionada, em que é possível identificar os usuários participantes da rede. Ao realizar o registro na plataforma, o usuário informa seu *email* que será utilizado como *username* e uma senha para acesso. Além disso, é necessário informar nome completo e CPF de modo a facilitar a identificação e possível responsabilização do usuário por possível ação maliciosa, permitindo a punição deste fora da plataforma, por ação judicial. Vale ressaltar que tanto o *email* quanto o CPF utilizado para registro devem ser únicos.

Usuários devidamente registrados podem acessar a plataforma informando seu *email* e senha. Caso as credenciais apresentadas estejam corretas, o servidor autenticará o usuário e enviará a ele um *token* de autorização JWT (*Json Web Token*), pelo qual o mesmo será autorizado a utilizar as funcionalidades oferecidas pela plataforma da *Carbon*.

5.2.8.2 *Password Hashing*

A proteção de senhas é fundamental para garantir a segurança dos usuários e dos seus dados. Para isso, uma prática importante é armazenar o *hash* da senha acrescida de uma *string* aleatória - chamada de sal, ou *salt* - no banco de dados, em vez de armazenar a senha às claras. Dessa maneira, ninguém que possua acesso ao banco consegue ler as senhas de acesso. Além disso, devido à concatenação do sal, senhas iguais e previsíveis produzem *hashes* diferentes, o que impede a utilização de tabelas de senhas e seus respectivos *hashes* - conhecidas como tabelas arco-íris - para a obtenção de uma determinada senha. Existem diferentes algoritmos que realizam o *hashing* de senhas com sal. Além disso, tais algoritmos combatem o ataque de força bruta, que é a realização de diversas tentativas autenticação no sistema com diferentes combinações de senha até que a combinação correta seja encontrada.

Para o projeto, foi utilizado o algoritmo *Argon2*, vencedor da competição *Password Hashing Competition* de 2015. Seu funcionamento é baseado em 5 parâmetros divididos em duas classes: primários e secundários. Os parâmetros primários são a mensagem e um *nonce*, respectivamente a senha do usuário e o *salt* utilizado no seu *hash*. Como parâmetros secundários, incluem-se três principais restrições responsáveis pelo custo computacional de execução do algoritmo, sendo elas: o grau de paralelismo que define o número de *threads* paralelas que podem ser executadas, o tempo de execução dado em número de iterações e a quantidade de memória exigida pelo algoritmo [40]. Esses parâmetros são os responsáveis por aumentar o tempo de processamento para a obtenção do *hash* da senha com sal, assim inviabilizando computacionalmente o ataque de força bruta. Por fim, para o projeto, utilizam-se os seguintes valores para as restrições: uso de memória de 16384 KiB, *thread* única e um total de 3 iterações.

Além da escolha do algoritmo de *Password Hashing*, foi decidido sobre como deve se dar a interação entre o cliente e o servidor durante o processo de autenticação. Tomando-se como o base o trabalho de Contini [41], o fluxo de autenticação ocorre da seguinte maneira:

1. O navegador do cliente envia um *request* com o *e-mail* ao servidor;

2. O servidor verifica se o *e-mail* em questão está registrado; nesse caso, é calculado o sal do usuário como um *hash* SHA-256 em função do *e-mail* e do *seed* do usuário armazenado. Caso o e-mail não esteja registrado, é calculado um *seed* em função do *e-mail* fornecido e de uma variável ambiente arbitrária denominada de *weed* e é

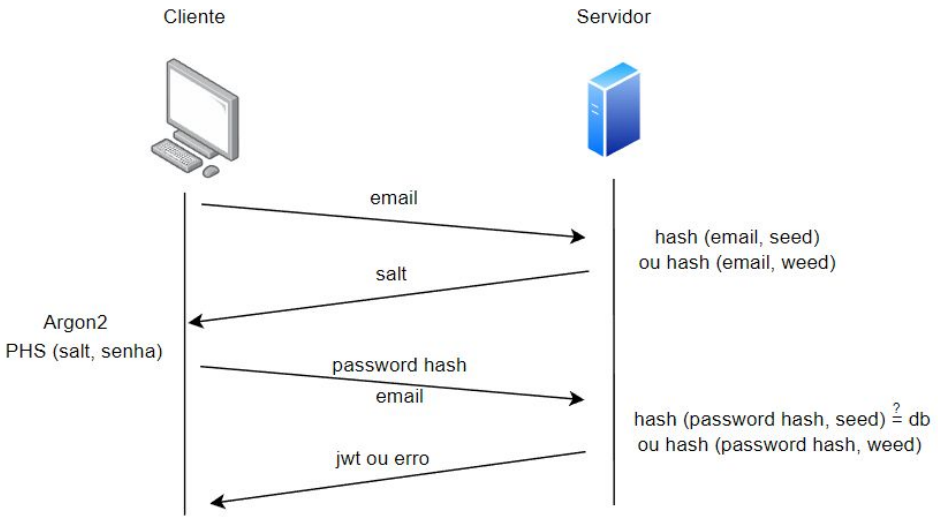
calculado um sal em função do *e-mail* e do *seed* obtido. Em qualquer caso, o sal calculado é retornado ao cliente;

3. O navegador do cliente calcula o *Password Hashing* via *Argon2* utilizando como parâmetros o sal retornado do servidor, a senha inserida pelo usuário e os parâmetros de processamento pré-definidos como grau de paralelismo, total de memória utilizada e total de iterações. O resultado é então enviado de volta ao servidor;

4. O servidor recebe o valor do *Password Hashing* e procura pelo usuário no banco de dados com base no *e-mail*. Se o usuário não existe, é calculado o *hash* entre o *Password Hashing* e o *weed* e, em seguida, o servidor retorna uma mensagem de erro ao cliente. Se o usuário existe, é calculado o *hash* entre o *Password Hashing* e seu *seed* e o valor resultante é comparado com a sua senha armazenada, autenticando o usuário em caso de igualdade ou retornando erro caso contrário.

O motivo pelo qual o servidor calcula o sal na etapa 2 e o *hash* do *Password Hashing* com o *weed* na etapa 4 para o caso de usuário não registrado é não fornecer pistas de informação a um atacante que esteja tentando autenticar no sistema por meio de diferentes *e-mails*. Caso tais valores não fossem calculados quando um usuário não está registrado, o tempo de resposta do servidor seria menor em comparação com o cenário de usuários registrados, de forma que o atacante passaria a procurar por outros usuários.

Figura 25: Password-Hashing



Fonte: Elaboração própria

5.2.8.3 Assinatura Local de Transações

Cada usuário registrado na plataforma possui um certificado digital emitido e assinado por uma Autoridade Certificadora, no qual está contido sua chave pública, e uma chave privada. Por meio do seu certificado, o usuário consegue se autenticar perante a plataforma e, por meio da sua chave privada, assinar propostas de transação que são enviadas para a rede *Blockchain*, que verifica a integridade.

Da forma como foi desenvolvido o protótipo atual, por maior simplicidade, as chaves privadas e os Certificate Signing Requests (CSR - Pedido de Assinatura de Certificado) dos usuários estão sendo geradas e armazenadas no lado do servidor. Entretanto, como possivelmente nem todos os clientes desejariam que a plataforma armazenasse as suas chaves privadas, tomou-se a decisão de implementar um segundo modo de operação, em que os clientes podem gerá-las localmente no momento do cadastro e enviar somente os CSRs gerados para o servidor, que efetua os registros e devolve os seus respectivos certificados digitais.

Para esse segundo modo de operação, as assinaturas podem ser assinadas localmente no lado do cliente sem que a sua chave privada trafegue pela *Internet*, de acordo com o seguinte fluxo de operação:

1. O cliente faz o *upload* de sua chave privada no *Website* da Carbon para a assinatura da transação (A chave não será transmitida para nenhum lugar);
2. O cliente, pelo *Website*, envia um request à API da Carbon solicitando a execução de uma transação;
3. O servidor responde com a proposta de transação não assinada;
4. O navegador do cliente realiza a assinatura local da transação e envia a proposta assinada para o servidor;
5. O servidor recebe a proposta de assinatura assinada, confere a integridade e executa a transação caso tenha sido validada.

Esse modo de operação visa atender clientes que desejam que suas chaves privadas sejam guardadas exclusivamente por eles, ao mesmo tempo em que evita o deslocamento dessas chaves pela *Internet*, assim promovendo um nível maior de segurança.

5.2.8.4 TLS e HTTPS

O TLS (*Transport Layer Security* - Segurança de Camada de Transporte) é um protocolo integrado a aplicações com o objetivo de proteger dados enviados em HTTP trocados entre cliente e servidor [42]. Construído sobre o procotolo TCP, fornece serviços de Confidencialidade, Integridade de dados, Irretratabilidade e Autenticação por meio de certificados digitais [43]. A utilização de TLS para a proteção do HTTP é chamada de HTTPS.

A conexão do TLS é divida etapas de *Handshake* e Transferência de Dados de Aplicação. O *Handshake* serve para iniciar a conexão. O cliente inicia com uma mensagem *ClientHello*, que é respondida pelo servidor que informa a versão do protocolo, o conjunto de algoritmos a serem utilizados e extensões. As demais mensagens do *Handshake* são utilizadas para autenticação via certificados X.509 e compartilhamento de chaves. Após esse compartilhamento, as mensagens passam a ser cifradas [43]. Ao final do *Handshake*, as aplicações estão prontas para se comunicarem entre si de maneira segura.

Diante da sua importância e utilização amplamente disseminada para promover comunicação segura entre sistemas via *Internet*, o TLS foi implementado para a comunicação entre o servidor e o cliente.

5.3 Testes e Avaliação

Para o desenvolvimento do código do projeto, foi utilizado o *Git* para versionamento de código e o *GitHub* para compartilhamento do repositório entre os membros da equipe. O sistema de versionamento de código unido a um repositório compartilhado é muito útil para estabelecer versões estáveis de código em um histórico, permitir o desenvolvimento paralelo de trabalho em *branches* separadas, promover controle e discussão e teste das modificações implementadas por cada usuário antes da sua incorporação ao código principal e possibilitar uma reversão do código para versões anteriores em casos como excesso de erros e instabilidade do código.

Ao longo de todo o desenvolvimento do projeto, os testes do sistema ocorreram de forma contínua. A cada nova funcionalidade construída, testes de validação foram feitos pelo seu autor e também por outros membros da equipe para a identificação e correção de *bugs* e, por fim, a validação do código para ser implementado definitivamente na *branch* principal no *GitHub*. Por fim, ao final de todo o desenvolvimento do *MVP*, realizou-se um teste final de validação de todo o sistema, que mostrou que a plataforma é funcional

e desempenha corretamente as funções construídas.

Devido ao atual estágio de desenvolvimento os testes não se focaram na performance do sistema, mas sim em suas funcionalidades. Dentre os principais e mais recorrentes testes realizados ao longo do projeto pode-se citar:

1. Testes de registro de novos usuários: realizou-se, primeiramente, o registro de um novo usuário utilizando *email* e CPF ainda não cadastrados. Ao fim do registro, verificou-se a presença do usuário e de seus corretos dados cadastrais no banco de dados. Realizaram-se, ainda, testes separados para tentativas de cadastro de usuários com *email* e CPF já cadastrados por outro usuário. Em ambos os testes a plataforma acusou erro no registro e o usuário não conseguiu se cadastrar. Tentativas de registro sem que fossem informados todos os dados obrigatório também foram barrados, antes mesmo que fossem realizadas chamadas à API.

2. Testes de *login* de usuário: uma vez cadastrados verificou-se que os usuários, informando as credencias corretas, recebiam o *jwt* e com isso o acesso à plataforma era permitido, conseguindo acessar todas as funcionalidades de um usuário comum. Ao informar senha ou *email* incorreto, o usuário recebia mensagem de erro e seu acesso à plataforma era prontamente negado devido a ausência do *jwt*. Além disso, verificou-se que, como o esperado, usuários administradores da *Carbon21* devidamente autenticados tinham acesso às funcionalidade de emissão de *tokens*.

3. Testes de emissão de *Sylvas*: consistia em acessar a plataforma com conta de administrador e emitir Sylvas a um usuário cadastrado. Verificou-se pelos teste que o usuário especificado realmente recebeu os Sylvas e na quantidade correta. Outro teste cobria o caso em que se tentava emitir *tokens* a contas inexistentes. Neste caso, a plataforma retornou mensagem de falha na execução da transação. Um teste final baseou-se na emissão prévia de *NFTs* para variadas contas, seguida da geração de Sylvas a partir desses *NFTs*. O teste serviu para confirmar que todos os usuário proprietários de *NFTs* receberam corretamente seus dividendos periódicos, enquanto usuários que não possuíam *NFTs* permaneceram com saldo inalterado.

4. Testes de emissão de *NFTs*: semelhante aos testes de emissão de Sylvas, entretanto com verificações extras que confirmavam a correta publicação dos metadados no *IPFS*.

5. Testes de consultas ao saldo de *Sylvas*: acessou-se a plataforma na conta de um usuário já cadastrado para o qual havia sido emitido Sylvas e verificava-se se o saldo

correspondia a quantidade emitida para o mesmo.

6. Testes de visualização de coleção: consistiu na verificação de que todos os *NFTs* de posse do usuário estavam listados em sua página de coleção. Verificou-se também se os metadados apresentados ao usuário eram, de fato, os que foram publicados na emissão do *NFT*. Para a página de coleção também foram realizados testes para comprovar o funcionamento da compensação. Verificou-se que ao utilizar o botão de compensação do *NFT* o seu estado de compensação era atualizado para "Compensado"e não era possível realizar nova compensação para o mesmo *NFT*. Ao se transferir o *NFT* seu estado mantinha-se.

7. Testes de transferência de *tokens*: testou-se a transferência para contas existentes e não existentes, verificando, neste último caso, que a plataforma não retirou os *tokens* da conta do usuário que tentou enviá-los. Verificou-se, ainda, que transferências com saldo insuficiente também eram bloqueadas.

6 CONSIDERAÇÕES FINAIS

Ao se desenvolver trabalho de potencial impacto socioambiental faz-se necessária a constante ponderação quando à viabilidade de soluções, sejam estas avaliadas sob a ótica econômica ou tecnológica, bem como os efeitos de cada decisão sobre os usuários finais.

Do modo geral o projeto obteve sucesso em implementar as técnicas e tecnologias, dentro do escopo delimitado anteriormente pelos objetivos almejados. Utilizando-se de padrões existentes, consolidou-se em sua arquitetura e funcionalidades básicas, implementando em natureza de prova de conceito técnicas específicas ao projeto e viabilizando melhorias e objetivos futuros.

Como explanado em seções anteriores, atingiram-se como marcos:

- Implantação de uma *Blockchain* por intermédio da plataforma permissionada *Hyperledger Fabric*, capaz de exercer as funcionalidades básicas já descritas por meio de nós distribuídos pelas organizações participantes.
- Desenvolvimento de *chaincodes* que dessem suporte integral às funcionalidades almejadas, juntamente à lógica de negócio.
- Adoção do padrão de tokens ERC-1155, que viabilizou o uso de tokens fungíveis e tokens não-fungíveis sem a necessidade de utilização de diferentes padrões, dentre outras vantagens.
- Emissão e transferência na *Blockchain* de tokens fungíveis (*FTs*) e tokens não-fungíveis (*NFTs*).
- Implantação de projeto *Node.js* juntamente ao *framework Express* e demais pacotes, viabilizando a construção do *Website* e da API da plataforma.
- *Landing page*, página de Registro de Usuário e Login.
- Páginas e funcionalidades de usuário, como Carteira e Coleção.

- Integração com a rede IPFS para armazenamento e exibição de metadados.
- Páginas e funcionalidades de administradores, como emissão e transferência de *tokens*.
- Considerações de segurança, como *TLS*, Assinatura Local e *JSON Web Token (JWT)*.

Por motivos de limitação de tempo e maior complexidade, algumas funcionalidades previstas para o *MVP* não foram implementadas e algumas simplificações foram feitas. As solicitações de emissão, ativação e emissão de *NFTs* não foram desenvolvidas por requerer a integração e interação da plataforma com órgãos ambientais como o IBAMA e o CETESB, responsáveis por permitir tais ações após validação documental. Em vez disso, a compensação foi simplificada para ser uma função que é chamada diretamente pelo dono do *NFT*, que atualiza o seu status correspondente para "Compensado". Além disso, o *Marketplace* não foi desenvolvido devido à sua dependência de uma fase mais avançada das funcionalidades já estabelecidas e das lógicas de solicitação não implementadas. Deseja-se, no entanto, implementar tais funções na continuidade do projeto.

6.1 Contribuições

O projeto tem, em sua concepção, aspirações de contribuir às praticas de preservação ambiental, concomitantemente ao aspecto social de possibilitar forma alternativa de ganhos econômicos para pequenos proprietários. Com a criação da plataforma almeja-se viabilizar a entrada de múltiplos usuários de diferentes conhecimentos e recursos em um ecossistema que estimule a troca de ativos e a promoção de ações como reflorestamento e compensação.

A Carbon 21 traz, juntamente à empreitada socioambiental, tecnologia em estado da arte fomentando o interesse no desenvolvimento acerca de assuntos como *Blockchain* e Criptoativos (e.g. *NFTs*), além de contribuir com o emprego de técnicas menos conhecidas pela população, como o *IPFS* e demais questões de segurança.

6.1.1 Contribuições da equipe à Carbon 21

A equipe do presente trabalho participou das etapas mais importantes do projeto junto com os demais participantes do grupo da *Carbon21*, composta também por profes-

sores, mestrandos e doutorandos. Definições de arquitetura, bem como definições e implementação de grandes funcionalidades foram realizadas em conjunto, mediante reuniões periódicas. Visto a natureza do gerenciamento ágil aplicado ao desenvolvimento, a contribuição dos participantes foi realizada em diferentes áreas dentro do projeto, envolvendo-se em todas as etapas. Tais contribuições coletivas, a saber, são as definições dos requisitos funcionais e não-funcionais, o levantamento dos casos de uso, a definição da arquitetura do sistema, a escolha de tecnologias e a implementação.

No que diz respeito à implementação, pode-se salientar contribuições majoritariamente desenvolvidas pelos integrantes da equipe, tais como a implementação do *Website* e criação das diversas páginas, antes e durante o desenvolvimento de suas funcionalidades. Juntamente às páginas, foram desenvolvidas pela equipe as lógicas de emissão de *tokens* (i.e. *Sylvas* e *NFTs*), as páginas e funcionalidades de transferência de *tokens*, Carteira e Coleção dos usuários e o registro e login de usuários.

Além disso, é de autoria da equipe a criação de módulos de funcionalidade da rede IPFS, bem como sua integração com os Metadados de *NFTs* (juntamente com rotas e *endpoints*), incluídos na página de Coleção. Outra contribuição voltada à segurança implementada pelo grupo foi assinatura local de transações, em que o cliente assina as transações da plataforma de forma local pela sua chave privada armazenada em seu computador.

Os professores participantes do grupo contribuíram com as suas orientações durante todo o projeto. Os demais membros, além das contribuições coletivas citadas anteriormente, participaram da implementação da estrutura da *Blockchain*, do desenvolvimento do Contrato Inteligente, da implementação do TLS, do desenvolvimento da API, da refatoração de códigos e do tratamento de erros comuns.

6.2 Perspectivas de Continuidade

O projeto *Carbon21* almeja a continuidade por meio da implementação de novas funcionalidades, bem como após revisitar técnicas já implementadas, porém de natureza de prova-de-conceito. Perspectivas de arquitetura serão reavaliadas a fim de melhorar a resiliência da solução juntamente ao ampliar e facilitar a participação de entidades externas, como de organizações ambientais e governamentais. A entrada de parceiros como o Ibama seria fundamental para a viabilização de funcionalidades e para o interesse de usuários no ingresso à plataforma, tendo em vista a integração com sistemas e bancos

de dados já existentes.

Foram mapeadas diversas funcionalidades a serem desenvolvidas em curto prazo, tal como a implementação da lógica de negócio responsável por permitir ao usuário final realizar um pedido de compensação, quando detentor do direito proveniente de *NFT*, juntamente aos órgãos responsáveis. Atualmente a emissão e compensação de *NFTs* são funcionalidades exclusivas de administradores da rede, limitando muito a entrada de novos participantes, por isso é relevante a criação de interfaces que permitam aos usuários finais solicitar a emissão ao inserir os dados necessários, bem como realizar a solicitação de ativação utilizando-se da devida documentação.

A transparência em plataformas que se utilizam de *Blockchain* é extremamente relevante, mas a participação com o armazenamento e manutenção de nós pode ser um dificultador, além de ser menos interessante em *Blockchains* permissionadas. Para lidar com tal aspecto pode-se implementar *logs* transparentes, facilmente verificáveis e armazenados externamente à rede. Juntamente a isso faz-se necessária a reavaliação da persistência de dados em organizações e na *Blockchain*. A fim de garantir a viabilidade econômica e auto-sustentação do projeto, vislumbra-se a implementação de mecanismos de taxação em transações.

Em longo prazo tem-se a perspectiva de criação de um *Marketplace* que permitiria a transação de *NFTs*, por meio da listagem, compras e vendas. Seriam também contemplados separadamente os direitos de compensação. A subdivisão de *NFTs* também pode ser avaliada, como permitido em novos padrões de tokens.

Além disso, melhorias e revisões de políticas de senhas e segurança podem tornar a solução do projeto ainda mais atraente ao público. Aliado a isso podem ser implementadas formas de verificações utilizando número da matrícula e geolocalização, bem como a integração com sistemas de monitoramento a distância (e.g. por satélite).

REFERÊNCIAS

[1] COHEN, R. ***On impact: A guide to the impact revolution***. [S.l.]: Ronald Cohen, 2018.

[2] UNGARETTI, M. ***ESG de A a Z: Tudo o que você precisa saber sobre o tema***. 2022. Acesso em: 08 junho 2022. Disponível em: <https://conteudos.xpi.com.br/esg/esg-de-a-a-z-tudo-o-que-voce-precisa-saber-sobre-o-tema/>.

[3] ANBIMA. ***Retrato da Sustentabilidade no Mercado de Capitais***. 2021. 17 p. Acesso em: 08 junho 2022. Disponível em: <https://www.anbima.com.br/pt_br/especial/sustentabilidade.htm>.

[4] NADINI, M. et al. **Mapping the NFT revolution: market trends, trade networks, and visual features**. *Scientific reports*, Nature Publishing Group, v. 11, n. 1, p. 1–11, 2021.

[5] ONE TREE PLANTED. ***Sustainable Development Goals***. [202?]. Acesso em: 10 maio 2022. Disponível em: <https://onetreeplanted.org/blogs/stories/reforestation-sustainable-development-goals>.

[6] IPAM. ***O que é e como funciona o mercado de carbono?*** [202?]. Acesso em: 10 maio 2022. Disponível em: <https://ipam.org.br/cartilhas-ipam/o-que-e-e-como-funciona-o-mercado-de-carbono/>.

[7] SUSTAINABLE CARBON. ***Minha propriedade pode gerar créditos de carbono?*** [202?]. Acesso em: 2 junho 2022. Disponível em: <https://www.sustainablecarbon.com/blog/minha-propriedade-pode-gerar-creditos-de-carbono/>.

[8] SMART FOREST COIN. ***Smart Forest Coin***. [202?]. Acesso em: 2 junho 2022. Disponível em: <https://www.smartforestcoin.com/>.

[9] TREECYCLE. ***Tokens and Blockchain - TREECYCLE — Powering Green Futures***. [202?]. Acesso em: 2 junho 2022. Disponível em: <https://treecycle.ch/en/tokens-and-blockchain/>.

[10] FORESTCOIN. ***Forestcoin Cryptocurrency — Plant trees to earn Forestcoin***. [202?]. Acesso em: 2 junho 2022. Disponível em: <https://forestcoin.earth/>.

[11] TREEDEFI. ***Treedefi - The Sustainable yield farm***. [202?]. Acesso em: 2 junho 2022. Disponível em: <https://treedefi.com/>.

[12] MOSS. ***Compense sua pegada de carbono com créditos de carbono da Moss***. [202?]. Acesso em: 2 junho 2022. Disponível em: <https://moss.earth/pt-br/>.

[13] O'DWYER, K. J.; MALONE, D. **Bitcoin mining and its energy footprint**. IET, 2014.

[14] STALLINGS, W.; BROWN, B. ***Computer Security: Principles and Practice***. [S.l.]: Pearson, 2015. 11, 22 p.

[15] SHIRLEY, R. ***Internet Security Glossary, Version 2***. RFC Editor, 2007. 26, 27 p. RFC 4949. (Request for Comments, 4949). Acesso em: 26 novembro 2022. Disponível em: <https://www.rfc-editor.org/info/rfc4949>.

[16] SHIRLEY, R. ***Internet Security Glossary, Version 2***. RFC Editor, 2007. 13 p. RFC 4949. (Request for Comments, 4949). Acesso em: 26 novembro 2022. Disponível em: <https://www.rfc-editor.org/info/rfc4949>.

[17] SHIRLEY, R. ***Internet Security Glossary, Version 2***. RFC Editor, 2007. 21, 22 p. RFC 4949. (Request for Comments, 4949). Acesso em: 26 novembro 2022. Disponível em: <https://www.rfc-editor.org/info/rfc4949>.

[18] SHIRLEY, R. ***Internet Security Glossary, Version 2***. RFC Editor, 2007. 119 p. RFC 4949. (Request for Comments, 4949). Acesso em: 26 novembro 2022. Disponível em: <https://www.rfc-editor.org/info/rfc4949>.

[19] GÖTHBERG, D. ***Public-key cryptography***. [2006]. Acesso em: 10 dezembro 2022. Disponível em: <https://en.wikipedia.org/wiki/Public-key_cryptography>.

[20] STALLINGS, W.; BROWN, B. ***Computer Security: Principles and Practice***. [S.l.]: Pearson, 2015. 60, 61 p.

[21] DOCUSIGN. ***Understanding digital signatures***. [2002?]. Acesso em: 10 dezembro 2022. Disponível em: <https://www.docusign.com/how-it-works/electronic-signature/digital-signature/digital-signature-faq>.

[22] NAKAMOTO, S. ***Bitcoin: A Peer-to-Peer Electronic Cash System***. 2008. Accesso em: 10 junho 2022. Disponível em: <https://bitcoin.org/bitcoin.pdf>.

[23] HYPERLEDGER. ***Introduction***. [202?]. Acesso em: 10 maio 2022. Disponível em: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatis.html>.

[24] IG. ***What is blockchain technology?*** [202?]. Acesso em: 17 maio 2022. Disponível em: <https://www.ig.com/ae/trading-strategies/what-is-blockchain-technology–200710>.

[25] MARTIN, T. ***How blockchain will disrupt your industry***. [202?]. Acesso em: 17 maio 2022. Disponível em: <https://www.slalom.com/insights/how-blockchain-will-disrupt-your-industry>.

[26] SCHULPEN, R. ***Smart contracts in the Netherlands***. [2018]. Acesso em: 10 junho 2022. Disponível em: <http://arno.uvt.nl/show.cgi?fid=146860>.

[27] ZHENG, Z. et al. An overview on smart contracts: Challenges, advances and platforms. *Future Generation Computer Systems*, Elsevier, v. 105, p. 475–491, 2020.

[28] LIU, M.; WU, K.; XU, J. **How Will Blockchain Technology Impact Auditing and Accounting: Permissionless versus Permissioned Blockchain**. *Current Issues in Auditing*, v. 13, n. 2, p. A19–A29, 08 2019. ISSN 1936-1270. Disponível em: <https://doi.org/10.2308/ciia-52540>.

[29] 101 BLOCKCHAINS. ***Permissioned vs Permissionless Blockchains***. 2020. Acesso em: 10 maio 2022. Disponível em: <https://101blockchains.com/permissioned-vs-permissionless-blockchains/>.

[30] EVANS, T. M. Cryptokitties, cryptography, and copyright. *AIPLA QJ*, HeinOnline, v. 47, p. 219, 2019.

[31] HYPERLEDGER. ***ERC-1155 Chaincode***. [2022]. Acesso em: 20 agosto 2022. Disponível em: <https://github.com/hyperledger/fabric-samples/blob/main/token-erc-1155/README.md>.

[32] IBM. ***Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains***. [2018]. Acesso em: 25 maio 2022.

[33] HYPERLEDGER. ***The Ordering Service***. [202?]. Acesso em: 3 maio 2022. Disponível em: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/orderer/ordering$_service.html¿$.

[34] HUANG, D.; MA, X.; ZHANG, S. **Performance Analysis of the Raft Consensus Algorithm for Private Blockchains**. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, v. 50, n. 1, p. 172–181, 2020.

[35] Oracle Corporation. ***MySQL 8.0 Reference Manual***. [2022]. Acesso em: 20 agosto 2022. Disponível em: <https://dev.mysql.com/doc/refman/8.0/en/>.

[36] IBM. ***Containers vs. Virtual Machines (VMs): What's the Difference?*** [2021]. Acesso em: 5 Dezembro 2022. Disponível em: <https://www.ibm.com/cloud/blog/containers-vs-vms>.

[37] BENET, J. **Ipfs-content addressed, versioned, p2p file system**. *arXiv preprint arXiv:1407.3561*, 2014.

[38] TILKOV, S.; VINOSKI, S. **Node.js: Using JavaScript to Build High-Performance Network Programs**. *IEEE Internet Computing*, v. 14, n. 6, p. 80–83, 2010.

[39] WITEK, R. et al. ***Eip-1155: Erc-1155 multi token standard. Ethereum Improvement Protocol, EIP-1155***. 2018.

[40] BIRYUKOV, A.; DINU, D.; KHOVRATOVICH, D. **Argon2: New Generation of Memory-Hard Functions for Password Hashing and Other Applications**. In: *2016 IEEE European Symposium on Security and Privacy (EuroSP)*. [S.l.: s.n.], 2016. p. 292–302.

[41] CONTINI, S. ***Method to Protect Passwords in Databases for Web Applications***. 2015. Cryptology ePrint Archive, Paper 2015/387. https://eprint.iacr.org/2015/387. Disponível em: <https://eprint.iacr.org/2015/387>.

[42] DAS, M. L.; SAMDARIA, N. **On the security of SSL/TLS-enabled applications**. *Applied Computing and Informatics*, v. 10, n. 1, p. 68–81, 2014. ISSN 2210-8327. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210832714000039>.

[43] HUSáK, M.; ČERMáK, M.; JIRSíK, T. e. a. **HTTPS traffic analysis and client identification using passive SSL/TLS fingerprinting**. *EURASIP Journal on Information Security*, v. 2016, n. 6, p. 4, 2016. Disponível em: <https://link.springer.com/article/10.1186/s13635-016-0030-7citeas>.