

UNIVERSIDADE FEDERAL DE UBERLÂNDIA  
FACULDADE DE ENGENHARIA MECÂNICA

CAIRO AMARAL FONSECA

Randomização, Aplicação e Inserção On-Chain de NFTs na Blockchain Solana: Uma  
Abordagem de Engenharia de Software

Uberlândia, Minas Gerais

2024

CAIRO AMARAL FONSECA

Projeto de fim de Curso apresentado à  
Faculdade de Engenharia Mecânica da  
Universidade Federal de Uberlândia como  
requisito parcial para obtenção do título  
de bacharel em Engenharia Mecânica

Orientador: Eustáquio São José de Faria

Uberlândia, Minas Gerais

2024

CAIRO AMARAL FONSECA

Randomização, Aplicação e Inserção On-Chain de NFTs na Blockchain Solana: Uma  
Abordagem de Engenharia de Software

Projeto de fim de Curso apresentado à  
Faculdade de Engenharia Mecânica da  
Universidade Federal de Uberlândia como  
requisito parcial para obtenção do título  
de bacharel em Engenharia Mecânica.

Uberlândia, dezembro de 2024

Banca Examinadora:

---

Eustáquio São José de Faria – Doutor UFU

---

Luciano José Arantes – Doutor UFU

Dedico este trabalho aos meus pais, pelo estímulo, carinho e compreensão, por nunca terem medido esforços para me proporcionar um ensino de qualidade durante todo o meu período escolar, ao meu irmão, por sempre me apoiar e estar ao meu lado em todos os momentos e a minha companheira, por me dar forças de sempre continuar lutando pelos meus objetivos.

“Ensinar não é transferir conhecimento, mas criar as possibilidades para a sua própria produção ou construção.”

(Paulo Freire)

## RESUMO

Este trabalho apresenta o desenvolvimento de um sistema para a geração e inserção on-chain de NFTs na blockchain Solana, utilizando um algoritmo de randomização de traits. O sistema foi desenvolvido para automatizar a criação de NFTs únicas, combinando elementos gráficos pré-definidos, como fundo, base, peito, cabeça, boca e olhos. Após a montagem da imagem da NFT, o sistema realiza o minting, registrando a NFT na blockchain Solana. A metodologia incluiu a implementação de um script em Node.js para gerenciar o processo de randomização e geração de imagens, e a integração com a biblioteca web3/solana para realizar o minting on-chain. Os resultados demonstram a eficácia do sistema em criar e registrar NFTs de forma automatizada e segura, sem a necessidade de interfaces complexas ou testes avançados, focando na funcionalidade e na eficiência do processo de minting.

**Palavras-chave:** NFTs, Randomização de traits e Blockchain Solana.

## **ABSTRACT**

This project presents the development of a system for generating and inserting NFTs on-chain on the Solana blockchain, using a trait randomization algorithm. The system was developed to automate the creation of unique NFTs by combining predefined graphic elements such as background, base, chest, head, mouth, and eyes. After assembling the NFT image, the system performs the minting process, registering the NFT on the Solana blockchain. The methodology included the implementation of a Node.js script to manage the randomization and image generation process, and the integration with the web3/solana library to mint NFTs on-chain. The results demonstrate the system's effectiveness in automatically and securely creating and registering NFTs without the need for complex interfaces or advanced testing, focusing on the functionality and efficiency of the minting process.

**Palavras-chave:** NFTs, Trait randomization and Solana blockchain.

## LISTA DE ILUSTRAÇÕES

Figura 1- Transações em blockchain.....	15
Figura 2 - Processo procedural de formação da imagem da NFT.....	30
Figura 3 - Exemplo da interface com dados fictícios .....	36
Figura 4 - Logs do backend do processo de criação da coleção. ....	37
Figura 5 - NFTs geradas pelo script com dados fictícios .....	39



## **LISTA DE ABREVIATURAS E SIGLAS**

- NFTs – Non-Fungible Tokens (Tokens Não Fungíveis)
- PoH – Proof of History (Prova de História)
- dApps – Decentralized Applications (Aplicações Descentralizadas)
- SOL – Solana (Criptomoeda da Blockchain Solana)
- PK – Private Key (Chave Privada)
- TPS – Transactions Per Second (Transações por Segundo)
- JSON – JavaScript Object Notation (Notação de Objeto JavaScript)
- URI – Uniform Resource Identifier (Identificador Uniforme de Recursos)
- IDE – Integrated Development Environment (Ambiente de Desenvolvimento Integrado)
- PoW – Proof of Work (Prova de Trabalho)
- SOL – Token da Solana Blockchain
- UI – User Interface (Interface do Usuário)

## SUMÁRIO

1. INTRODUÇÃO.....	12
1.1. Objetivo Geral .....	13
1.2. Objetivos Específicos .....	13
2. REVISÃO BIBLIOGRÁFICA .....	13
2.1. Blockchain .....	13
2.1.1. Segurança das transações em Blockchains .....	15
2.2. Smart contracts .....	16
2.3. Tokens e suas classificações .....	17
2.3.1. Token fungível.....	17
2.3.2. Token não fungível (NFT).....	18
2.4. Criptomoedas .....	18
2.5. Solana .....	19
2.5.1. Diferenças entre Solana e Bitcoin .....	20
2.5.2. Ambientes de operação.....	21
2.5.3. Diferenças entre ambientes.....	21
3. METODOLOGIA.....	22
4. DESENVOLVIMENTO.....	23
4.1. Softwares utilizados.....	23
4.2. Backend .....	24
4.2.1. Bibliotecas utilizadas .....	25
4.2.2. Criação e organização do projeto .....	26
4.2.3. Processos .....	27
4.2.4. Criação da coleção de NFTs .....	27
4.2.5. Criação dos metadados da NFT .....	28
4.2.6. Geração das Imagens .....	29

4.2.7.	Registro das NFTs e validação de coleção .....	31
4.3.	Frontend.....	33
4.3.1.	Bibliotecas utilizadas .....	34
4.3.2.	Criação e organização do projeto .....	34
4.3.3.	Criação dos arquivos e processos .....	34
4.3.4.	Serviço de requisição HTTP .....	35
4.3.5.	Homepage da aplicação .....	35
5.	TESTE DO PROCESSO .....	35
6.	RESULTADOS E CONCLUSAO .....	38
7.	REFERÊNCIAS BIBLIOGRAFICAS .....	40

## 1. INTRODUÇÃO

A tecnologia blockchain, inicialmente desenvolvida para sustentar o Bitcoin, rapidamente se expandiu para diversas áreas além das criptomoedas, tornando-se um pilar essencial da economia digital moderna. Sua adoção crescente tem impulsionado tanto empresas quanto indivíduos a explorar novas formas de investimento e negócios, especialmente com o surgimento dos NFTs (Tokens Não Fungíveis). Dados recentes apontam que cerca de um em cada dez americanos já investe em criptomoedas, ilustrando o crescimento contínuo desse mercado (REINICKE, 2021).

Os NFTs revolucionaram a forma como se entende a propriedade no mundo digital, permitindo que itens como arte, música e até bens físicos sejam representados de maneira única e intransferível na blockchain. Um exemplo marcante desse impacto é a venda da obra "Everydays: The First 5000 Days" do artista Beeple, leiloadada por impressionantes 69 milhões de dólares na Christie's, uma das maiores casas de leilão do mundo. Essa venda consolidou os NFTs no mercado tradicional de arte e destacou a legitimidade da arte digital no cenário global (NFTMETRIA, 2023).

Além de permitir que artistas digitais monetizem suas criações de forma autêntica e segura, os NFTs também democratizaram o acesso ao mercado de arte, removendo a necessidade de intermediários tradicionais como galerias ou agentes. Isso abriu novas oportunidades tanto para artistas emergentes quanto para colecionadores em busca de obras exclusivas. Artistas agora têm a possibilidade de comercializar suas criações diretamente com o público, criando um ambiente mais inclusivo e acessível para a criação e venda de arte (NFTMETRIA, 2023; BLOCKAPPS INC., 2023).

Essas mudanças indicam que o mercado de arte digital, impulsionado pelos NFTs, continuará evoluindo e transformando profundamente a forma como a humanidade consome, compra e entende a arte. O impacto dessas tecnologias vai além do mundo da arte, abrangendo também o setor tecnológico, o colecionismo e até o comércio global, com um potencial de crescimento significativo nos próximos anos (NFTMETRIA, 2023; CULT MTL, 2023).

### **1.1. Objetivo Geral**

O objetivo deste projeto é desenvolver uma plataforma acessível e eficiente para a criação e inserção de NFTs (Tokens Não Fungíveis) na blockchain Solana, automatizando um processo que, até então, exige conhecimento técnico avançado. O sistema visa facilitar a geração de coleções de NFTs ao combinar automaticamente elementos visuais (conhecidos como traits), garantindo que cada NFT seja única e registrada de forma segura na blockchain.

### **1.2. Objetivos Específicos**

O foco principal é simplificar o acesso a esse ambiente tecnológico que, apesar de sua popularidade crescente, ainda apresenta barreiras para muitos usuários. O projeto tem como objetivo não apenas tornar a criação de NFTs mais rápida e acessível, mas também assegurar que o processo seja modular, escalável e de fácil manutenção, permitindo a futura expansão do sistema.

A plataforma incluirá um algoritmo que fará a randomização das traits e gerará tanto os metadados quanto as imagens associadas a cada NFT, permitindo que o usuário, com apenas alguns cliques, possa criar e registrar coleções inteiras. Além disso, o sistema aproveitará a escalabilidade e os baixos custos da Solana, tornando a mintagem (inserção na blockchain) viável mesmo para grandes volumes de NFTs.

## **2. REVISÃO BIBLIOGRÁFICA**

### **2.1. Blockchain**

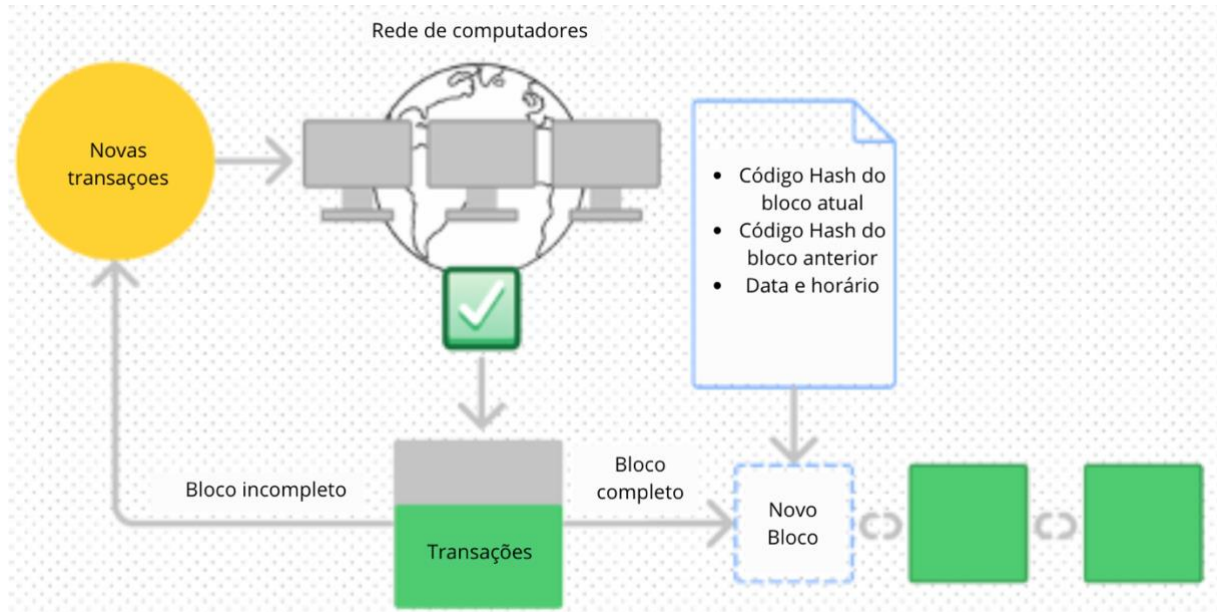
A blockchain é uma tecnologia de registro distribuído (DLT, do inglês Distributed Ledger Technology) que permite o armazenamento de informações de maneira descentralizada, imutável e transparente. A tecnologia foi primeiramente idealizada em 1991 pelos matemáticos Stuart Haber e W. Scott Stornetta, que desejavam criar um sistema capaz de impedir que hackers adulterassem a data e a hora de documentos digitais, utilizando técnicas criptográficas para garantir a integridade dos dados (HAYES, 2022). No entanto, a primeira implementação prática da blockchain ocorreu somente em 2008, quando Satoshi Nakamoto propôs sua aplicação no Bitcoin, a primeira criptomoeda descentralizada (NAKAMOTO, 2008).

A blockchain funciona como um livro-razão distribuído, onde transações são registradas de maneira segura e imutável por meio de blocos interligados. Cada bloco contém uma lista de transações verificadas, um registro do bloco anterior e um hash criptográfico que assegura a integridade dos dados (ANTONPOULOS, 2017). A imutabilidade da blockchain é garantida por um mecanismo de consenso distribuído, que valida as transações antes que elas sejam adicionadas à cadeia. Esse mecanismo elimina a necessidade de uma autoridade central, como bancos, proporcionando um sistema seguro e confiável para o registro de dados e transações.

Um dos mecanismos de consenso mais utilizados é o Proof of Work (PoW), utilizado no Bitcoin, que exige que computadores da rede resolvam problemas matemáticos complexos para validar transações. Contudo, blockchains mais recentes, como a Solana, utilizam mecanismos alternativos, como o Proof of History (PoH), criado por Anatoly Yakovenko em 2020, que permite uma escalabilidade e velocidade de transações significativamente maiores (YAKOVENKO, 2020).

Essa tecnologia tem permitido o desenvolvimento de diversas aplicações descentralizadas (dApps), como contratos inteligentes e NFTs. As dApps operam diretamente na blockchain, garantindo a transparência e a segurança nas transações (Figura 1), sem a necessidade de intermediários (PÉTERFAY, 2022). Dessa forma, a blockchain continua a ser uma solução robusta e segura para o desenvolvimento de aplicações modernas no ecossistema digital.

Figura 1- Transações em blockchain



Fonte: Péterfy (2022) - Traduzido

### 2.1.1. Segurança das transações em Blockchains

A segurança das transações realizadas em uma blockchain pode ser entendida observando as etapas que ocorrem durante a adição de novos blocos à cadeia. Cada transação feita em uma blockchain é registrada em um bloco que, por sua vez, é armazenado de forma linear e cronológica. Isso significa que novos blocos sempre são adicionados ao final da blockchain. Cada bloco contém um hash exclusivo, o hash do bloco anterior e um registro de carimbo de data e hora (timestamp), o que cria uma ligação imutável entre os blocos subsequentes (PÉTERFAY, 2022).

Esse mecanismo de segurança é fundamental para manter a integridade da blockchain. Se um hacker tentasse modificar um bloco anterior da cadeia, o hash desse bloco seria alterado, assim como o timestamp. Como os blocos seguintes dependem do hash do bloco anterior, qualquer tentativa de alteração geraria inconsistências em toda a cadeia subsequente. As outras cópias da blockchain armazenadas pelos nós na rede notariam essa discrepância e rejeitariam a versão adulterada (HAYES, 2022). Isso torna a blockchain imutável, o que dificulta a modificação retroativa de transações.

A única maneira de um ataque bem-sucedido ocorrer seria se o hacker conseguisse controlar mais de 51% das cópias da blockchain na rede (o chamado ataque de 51%). No entanto, devido ao tamanho crescente das redes de blockchain, especialmente em redes de criptomoedas populares como Bitcoin e Solana, tal ataque é extremamente difícil de realizar, já que seria necessário alterar mais da metade de todas as cópias simultaneamente (RADOCCCHIA, 2018). Além disso, a manipulação precisaria ocorrer antes que os nós restantes detectassem e corrigissem as discrepâncias.

Esse modelo de segurança, baseado na validação distribuída e imutabilidade dos dados, é um dos principais motivos pelos quais grandes empresas, como Walmart, Pfizer, AIG e Siemens, têm adotado o uso de blockchain para registrar transações de forma segura e confiável (HAYES, 2022). Assim, a estrutura da blockchain torna praticamente impossível modificar dados já registrados, o que aumenta significativamente a confiabilidade do sistema e a proteção contra fraudes.

## **2.2. Smart contracts**

Os smart contracts são programas autoexecutáveis armazenados em uma blockchain. O código desses contracts define as condições e ações que devem ser tomadas de forma autônoma quando certas condições são atendidas. Em essência, eles funcionam com base em instruções do tipo "se/quando isso acontecer, então faça aquilo", automatizando transações sem a necessidade de intermediários, como advogados ou bancos. Isso permite uma execução mais rápida e transparente das obrigações contratuais entre as partes envolvidas, eliminando a necessidade de confiança em uma entidade central (IBM, 2023).

Uma vez implantados em uma blockchain, os contratos inteligentes tornam-se imutáveis, ou seja, o código que os define não pode ser alterado após sua implantação. Essa imutabilidade garante que as condições originalmente definidas no contrato sejam sempre respeitadas, fornecendo um alto nível de segurança e confiança. A execução dos contratos é garantida pela própria rede distribuída da blockchain, que valida as transações e registra as interações de forma transparente e auditável, sem a necessidade de intervenção de terceiros (PETERS & PANAYI, 2016).



Como dito anteriormente, isso cria um ambiente robusto para a execução de acordos sem a necessidade de um intermediário central, como instituições financeiras ou entidades governamentais.

No entanto, um dos maiores desafios dos contratos inteligentes reside no risco de falhas no código. Uma vez que o código é imutável, qualquer erro ou vulnerabilidade presente no contrato pode ser explorado por atacantes, potencialmente resultando em perdas financeiras ou em transações incorretas. Um exemplo notório é o ataque à organização autônoma descentralizada (DAO) em 2016, onde uma falha no código do smart contract resultou na perda de milhões de dólares em criptomoeda. Esse caso destacou a importância da revisão e auditoria rigorosa dos contratos inteligentes antes de sua implantação em uma blockchain. Ferramentas de verificação formal e auditorias de segurança têm sido adotadas para minimizar esses riscos e garantir que o código funcione conforme o esperado (ATZORI, 2015; TSOUTSOS & MAKRIS, 2018).

### **2.3. Tokens e suas classificações**

Os tokens são representações digitais de ativos que podem ser transferidos ou armazenados em uma blockchain. Esses ativos podem representar tanto valores financeiros quanto bens tangíveis ou intangíveis, como propriedade intelectual, acesso a serviços ou ativos digitais como criptomoedas e NFTs. Em termos simples, um token é uma unidade de valor emitida por uma entidade e registrada na blockchain para permitir transações seguras e transparentes (HAYES, 2022; WANG et al., 2021).

#### **2.3.1. Token fungível**

Um token fungível é um ativo digital que pode ser trocado por outro de igual valor, de maneira que cada unidade é indistinguível de outra. Um bom exemplo de tokens fungíveis são as criptomoedas, como Bitcoin e Ether. Um Bitcoin, por exemplo, tem o mesmo valor e funcionalidade que qualquer outro Bitcoin, assim como o Ether é intercambiável dentro do sistema Ethereum (WANG et al., 2021). Esses tokens são utilizados para transações financeiras, pagamentos e contratos inteligentes, servindo essencialmente como uma moeda digital.

### **2.3.2. Token não fungível (NFT)**

Em contraste, um token não fungível (NFT) é um ativo digital único que não pode ser trocado por outro de valor equivalente, pois cada NFT possui características distintas que o tornam único. Enquanto os tokens fungíveis podem ser divididos e trocados entre si, os NFTs são indivisíveis e não intercambiáveis. Eles são comumente usados para representar a propriedade de ativos digitais únicos, como obras de arte, itens colecionáveis, músicas ou vídeos, sendo registrados em blockchains como Ethereum e Solana. Cada NFT tem seus próprios metadados que comprovam sua autenticidade e singularidade, o que o diferencia dos tokens fungíveis tradicionais (CHEN, 2021).

A principal diferença entre os dois tipos de tokens está, portanto, na fungibilidade. Enquanto tokens fungíveis, como o Bitcoin, podem ser trocados por outros da mesma forma e valor, os NFTs são únicos e não podem ser trocados por outro ativo idêntico. Isso faz com que os NFTs sejam amplamente usados para certificar a propriedade de ativos únicos no mundo digital, abrindo novos mercados para colecionáveis digitais e arte.

### **2.4. Criptomoedas**

As criptomoedas são moedas digitais ou virtuais que utilizam criptografia para garantir a segurança das transações e controlar a criação de novas unidades. A primeira e mais conhecida criptomoeda é o Bitcoin, que foi introduzida em 2008 por uma pessoa ou grupo de pessoas sob o pseudônimo de Satoshi Nakamoto. Nakamoto publicou o famoso artigo intitulado "Bitcoin: A Peer-to-Peer Electronic Cash System", onde descreveu como seria possível criar uma moeda digital que não dependesse de uma entidade central para validar transações (NAKAMOTO, 2008).

O Bitcoin foi criado como uma resposta às fraquezas do sistema financeiro tradicional, buscando oferecer uma alternativa descentralizada para o armazenamento e a transferência de valor. Utilizando a tecnologia blockchain, o Bitcoin permite que transações sejam verificadas e registradas de forma transparente e imutável por uma rede distribuída de computadores (nodes), sem a necessidade de intermediários como bancos ou governos (ANTONPOULOS, 2017).

O primeiro bloco de Bitcoin, conhecido como bloco gênese, foi minerado em 3 de janeiro de 2009, marcando o início da rede Bitcoin. Desde então, o Bitcoin cresceu

exponencialmente em popularidade e valor, abrindo caminho para a criação de milhares de outras criptomoedas, muitas delas baseadas em princípios semelhantes de descentralização e segurança (WANG et al., 2021).

O sucesso do Bitcoin inaugurou a era das criptomoedas, que continuam a revolucionar o setor financeiro ao permitir transações seguras, rápidas e sem fronteiras. Outras criptomoedas, como o Ethereum, surgiram posteriormente, trazendo inovações como contratos inteligentes e aplicações descentralizadas (dApps), que expandiram ainda mais as possibilidades do uso de criptomoedas.

## **2.5. Solana**

A Solana é uma blockchain de alto desempenho criada em 2020 por Anatoly Yakovenko, um ex-engenheiro da Qualcomm e Dropbox. Seu objetivo era resolver os problemas de escalabilidade enfrentados por blockchains mais antigas, como o Bitcoin e o Ethereum. A Solana foi projetada para suportar um grande número de transações por segundo, com baixas taxas, o que a torna ideal para aplicações descentralizadas (dApps) e mercados de NFTs (YAKOVENKO, 2020).

A Solana se destaca por suas inovações tecnológicas, sendo uma das blockchains mais rápidas disponíveis atualmente. As propriedades que tornam a Solana única incluem:

- **Proof of History (PoH):** Uma das inovações mais notáveis da Solana é o mecanismo de consenso PoH, que funciona em conjunto com o Proof of Stake (PoS). O PoH cria um registro de tempo verificável e descentralizado, permitindo que a rede processe transações em paralelo. Isso aumenta drasticamente a velocidade da rede, possibilitando a Solana lidar com mais de 65.000 transações por segundo (TPS), sem sacrificar a segurança ou descentralização (YAKOVENKO, 2020).
- **Baixas Taxas de Transação:** Enquanto blockchains como o Ethereum e Bitcoin enfrentam problemas de congestionamento e altas taxas, as taxas de transação na Solana permanecem extremamente baixas, o que a torna atraente para desenvolvedores e usuários de dApps.

- Escalabilidade: A arquitetura da Solana foi criada para escalar com o aumento da adoção, sem depender de soluções secundárias como "layer 2" ou fragmentação. Isso significa que à medida que mais usuários se juntam à rede, sua capacidade de processamento de transações pode aumentar sem comprometer a descentralização.

### **2.5.1. Diferenças entre Solana e Bitcoin**

- Mecanismo de Consenso: Enquanto o Bitcoin utiliza o Proof of Work (PoW), que requer que mineradores resolvam problemas matemáticos complexos para validar transações, a Solana utiliza uma combinação de Proof of Stake (PoS) e Proof of History (PoH). O PoW do Bitcoin consome grandes quantidades de energia, tornando-o mais lento, enquanto o PoH da Solana aumenta a eficiência e a velocidade da rede sem demandar tanta energia (YAKOVENKO, 2020; ANTONOPOULOS, 2017).
- Velocidade: O Bitcoin pode processar apenas cerca de 7 transações por segundo, enquanto a Solana é capaz de processar mais de 65.000 TPS (transações por segundo), tornando a Solana muito mais rápida e escalável para grandes volumes de transações (YAKOVENKO, 2020).
- Taxas: As taxas de transação no Bitcoin podem ser altas, especialmente durante períodos de congestionamento da rede. Em contraste, as taxas na Solana são extremamente baixas, com cada transação custando frações de centavo, independentemente do volume de transações na rede (WANG et al., 2021).
- Finalidade de Uso: O Bitcoin foi criado principalmente como uma reserva de valor e um sistema de pagamento digital descentralizado. Por outro lado, a Solana foi desenvolvida com foco em ser uma plataforma robusta para aplicações descentralizadas (dApps) e mercados de NFTs, além de oferecer transações rápidas e de baixo custo (YAKOVENKO, 2020).

### **2.5.2. Ambientes de operação**

A blockchain Solana opera em três principais ambientes chamados de clusters: Mainnet, Devnet e Testnet. Esses clusters fornecem diferentes funcionalidades, desde testes de desenvolvimento até a operação da rede principal, cada um com uma finalidade específica.

A Mainnet da Solana é a rede principal e totalmente funcional, onde as transações reais e o staking <sup>1</sup> de tokens acontecem. Nessa rede, usuários e desenvolvedores podem executar contratos inteligentes, mintar <sup>2</sup> e realizar transações de maneira segura e imutável. Todas as transações que ocorrem na Mainnet são definitivas e registradas permanentemente na blockchain, com valor econômico real envolvido. É nesse ambiente que as soluções de produção são implantadas, e onde os tokens SOL têm valor de mercado (YAKOVENKO, 2020).

A Devnet é um ambiente de desenvolvimento voltado para desenvolvedores testarem seus contratos inteligentes, aplicativos descentralizados (dApps) e transações antes de movê-los para a Mainnet. É uma rede segura para experimentação e aprendizado, onde os desenvolvedores podem usar tokens SOL de teste, sem valor econômico real, para testar suas implementações sem risco financeiro. Embora seja similar à Mainnet em termos de funcionalidade, ela é usada exclusivamente para testes e desenvolvimento (SOLANA FOUNDATION, 2021).

A Testnet é uma rede de testes pública utilizada para experimentos em larga escala e validação da estabilidade da rede antes de uma implementação final na Mainnet. Nela, desenvolvedores e validadores podem testar novas atualizações de software, mudanças na arquitetura ou otimizações de desempenho. A Testnet é crucial para garantir que as atualizações não afetem negativamente a Mainnet. Assim como na Devnet, os tokens SOL aqui não têm valor econômico (YAKOVENKO, 2020).

### **2.5.3. Diferenças entre ambientes**

- Mainnet: Rede principal, com transações reais e tokens de valor econômico.
- Devnet: Ambiente de desenvolvimento, com tokens de teste, usado para desenvolvimento de novos projetos.

---

<sup>1</sup> Processo de bloquear criptomoedas em uma rede blockchain para apoiar suas operações, como validação de transações, em troca de recompensas.

<sup>2</sup> Processo de criar um token ou NFT em uma blockchain, registrando-o de forma única e imutável na rede.

- Testnet: Rede de testes para validar atualizações e experimentar mudanças de larga escala na rede antes de implementações na Mainnet.

Esses clusters permitem que desenvolvedores criem, testem e implementem seus projetos de maneira segura e eficaz, garantindo que a blockchain Solana continue a evoluir sem interrupções ou falhas inesperadas.

### **3. METODOLOGIA**

A metodologia utilizada para o desenvolvimento deste projeto seguiu os princípios fundamentais da Engenharia de Software, com foco em modularidade, escalabilidade e automação. As decisões técnicas foram guiadas pela necessidade de criar um sistema eficiente e seguro para a geração, randomização e inserção on-chain de NFTs na blockchain Solana.

O sistema foi estruturado de maneira modular, separando claramente as responsabilidades entre o frontend (interface) e o backend (operacional). A modularidade foi uma escolha estratégica para permitir que cada parte do sistema pudesse ser desenvolvida, testada e escalada de forma independente, o que facilita futuras manutenções e expansões.

O backend foi desenhado para lidar com as tarefas mais críticas, como a randomização das traits e o processo de minting <sup>3</sup> das NFTs na blockchain.

O frontend foi mantido simples, servindo apenas como uma interface para que os usuários forneçam os dados necessários ao backend, que realiza a maior parte do processamento.

Para garantir que o sistema fosse escalável, tanto em termos de volume de NFTs gerados, quanto de desempenho, foram utilizadas tecnologias que suportam alto throughput <sup>4</sup>. A blockchain escolhida, Solana, foi selecionada por sua capacidade de processar milhares de transações por segundo com baixas taxas de transação, o que permite uma operação rápida e econômica durante o processo de minting.

---

<sup>3</sup> Processo de registrar e criar tokens ou NFTs em uma blockchain, tornando-os ativos digitais únicos e disponíveis para uso ou transação.

<sup>4</sup> Capacidade de um sistema processar e completar um número específico de operações ou transações em um determinado período.

O projeto também levou em consideração a possibilidade de expansão futura, com o backend preparado para suportar novos tipos de coleções de NFTs ou até mesmo integração com outros blockchains, se necessário.

A automação foi um pilar central da metodologia do projeto. Todas as etapas do processo, desde a geração das traits <sup>5</sup>, criação das imagens, até o minting e registro on-chain, foram automatizadas para minimizar a intervenção manual e reduzir o tempo de execução. Scripts foram desenvolvidos para garantir que o sistema pudesse gerar grandes volumes de NFTs de forma eficiente e com mínima supervisão.

A escolha das ferramentas foi feita com base na capacidade de integrar com a blockchain Solana e suportar o processo de criação de NFTs. Tecnologias como Node.js e web3.js foram adotadas para o backend, enquanto Metaplex e Canvas foram utilizados para o minting e geração de imagens. A estrutura escolhida também garante que as operações sejam seguras e que todos os dados sejam processados com a imutabilidade e segurança garantida pela blockchain.

Antes de implementar o sistema na Mainnet (rede principal), o sistema foi extensivamente testado na Devnet da Solana, o ambiente de desenvolvimento que simula a blockchain real sem envolver custos com transações. Testes foram conduzidos para verificar a corretude do algoritmo de randomização, a criação correta dos metadados e imagens, e o processo de minting. Os testes também incluíram a verificação de que o processo de geração e minting de NFTs funcionava conforme o esperado em diferentes cenários de carga.

## **4. DESENVOLVIMENTO**

### **4.1. Softwares utilizados**

- Windows 11: Sistema operacional utilizado no desenvolvimento do projeto, fornecendo a base necessária para a execução de todas as ferramentas e bibliotecas. O ambiente Windows foi configurado para suportar o desenvolvimento tanto do backend quanto do frontend, além de permitir a interação direta com a blockchain Solana e a execução de testes locais.
- Visual Studio Code (VS Code): IDE (Integrated Development Environment) utilizada para escrever e gerenciar o código do projeto. O VS Code foi escolhido

---

<sup>5</sup> Características ou atributos únicos que compõem um NFT, como cor, estilo ou acessórios, geralmente utilizados para definir sua aparência e raridade dentro de uma coleção.

devido à sua flexibilidade, ampla gama de extensões e suporte robusto para linguagens como TypeScript, JavaScript. A IDE facilitou a organização do projeto, a depuração do código e a integração com bibliotecas externas.

- Postman: Utilizado para testar as requisições do backend. Essa ferramenta permitiu a simulação de chamadas HTTP para verificar se as rotas e endpoints do backend estavam funcionando conforme esperado. O Postman foi essencial durante a fase de testes para garantir que os dados enviados e recebidos entre o frontend e o backend fossem processados corretamente e que o sistema estivesse preparado para lidar com diferentes tipos de entradas.
- Chrome Web Browser: Utilizado como navegador para testar a interface e a interação com o sistema. O Chrome, por ser um dos navegadores mais populares e compatíveis com padrões web modernos, foi escolhido para garantir que a interface estivesse funcionando de maneira adequada. O navegador também permitiu a visualização das respostas do backend e a análise de desempenho da aplicação no lado do cliente.

## **4.2. Backend**

O backend é a parte central responsável pelo processamento e execução das funções mais críticas do sistema, como a criação de coleções de NFTs e a inserção on-chain desses ativos na blockchain Solana. Ele gerencia todo o fluxo de dados, realizando operações essenciais para garantir que as NFTs sejam geradas, combinadas e registradas corretamente. Além disso, o backend é projetado para lidar com a randomização das traits visuais, a geração das imagens e a mintagem dos tokens.

O fluxo de trabalho do backend segue as seguintes etapas:

1. Criação da Coleção de NFTs
2. Criação dos metadados da NFT
3. Geração das Imagens
4. Registro das NFTs e validação de coleção



As etapas 2 e 3 são repetidas para cada NFT, conforme a quantidade solicitada. Ao final do processo, o backend retorna uma mensagem ao frontend indicando o sucesso ou falha da operação, juntamente com os detalhes das NFTs geradas.

Para uma visualização completa de todo o código que implementa essas funcionalidades, consulte o Apêndice A, onde estão detalhados os scripts que suportam o backend do sistema.

#### **4.2.1. Bibliotecas utilizadas**

- **mpl-token-metadata:** Esta biblioteca faz parte do conjunto de ferramentas do Metaplex, amplamente utilizada para a criação e gestão de NFTs na blockchain Solana. Ela é responsável por gerenciar os metadados dos tokens, como o nome, a descrição e a URI das imagens. No contexto do projeto, essa biblioteca é essencial para garantir que cada NFT tenha seus metadados gerados e organizados corretamente antes de ser mintada e registrada na blockchain.
- **Umi:** O Umi é uma biblioteca desenvolvida pela Metaplex Foundation que simplifica o desenvolvimento de contratos inteligentes e NFTs na blockchain Solana. Ela fornece uma abstração de baixo nível para interagir com contratos e realizar operações mais complexas na blockchain. Neste projeto, o Umi facilita o processo de minting das NFTs, além de fornecer uma interface mais amigável para desenvolvedores que trabalham com a Solana, ajudando a organizar os processos de criação e inserção dos NFTs na blockchain.
- **umi-bundle-defaults:** Essa biblioteca trabalha em conjunto com o Umi, fornecendo um conjunto padrão de ferramentas e configurações para facilitar o desenvolvimento de projetos na Solana. Ela ajuda a estruturar o projeto e configurar o ambiente de forma otimizada, permitindo um fluxo de trabalho mais simples e eficiente na criação de NFTs e contratos inteligentes.
- **umi-uploader-irys:** Essa biblioteca é uma extensão do Umi, especificamente voltada para o upload de metadados e arquivos, como imagens e documentos, necessários para o minting de NFTs. Ela simplifica o processo de upload para plataformas descentralizadas, como o Arweave, garantindo que os arquivos das NFTs sejam armazenados de forma segura e acessível no futuro.

- **solana-developers/helpers:** Essa biblioteca oferece uma série de funções utilitárias para desenvolvedores que trabalham com a blockchain Solana. Ela simplifica tarefas comuns, como a criação de transações, consulta de saldos e interações com contas na blockchain. Neste projeto, essas ferramentas são usadas para facilitar a comunicação entre o backend e a blockchain, automatizando algumas operações essenciais.
- **web3.js:** Web3.js é a principal biblioteca usada para interagir diretamente com a blockchain Solana. Ela facilita a criação e o envio de transações, a comunicação com contratos inteligentes e a consulta de informações da blockchain. Essa biblioteca é o núcleo que permite que o backend conecte, execute e registre NFTs na Solana de maneira eficiente e segura.
- **Arweave:** O Arweave é um protocolo de armazenamento descentralizado usado para guardar arquivos de forma permanente. É utilizado para armazenar metadados e imagens das NFTs, garantindo que esses arquivos estejam sempre disponíveis.
- **bs58:** É uma biblioteca usada para codificar e decodificar dados em Base58, um formato de codificação que é amplamente utilizado em sistemas blockchain, como Bitcoin e Solana. Ela é utilizada no projeto para manipular endereços e transações em um formato que seja compatível com os requisitos da blockchain.
- **Canvas:** A biblioteca Canvas permite a manipulação de gráficos diretamente no backend, sendo usada para gerar as imagens das NFTs com base nos atributos visuais combinados aleatoriamente. Ela oferece ferramentas para renderizar e sobrepor imagens, facilitando a criação gráfica de cada NFT.

#### **4.2.2. Criação e organização do projeto**

O desenvolvimento do backend começou com a utilização de uma linha de comando para gerar a estrutura inicial do projeto:

```
npm express-generator app-backend
```

Esse comando gerou automaticamente um repositório baseado no Node.js com o framework Express, estruturando a base do backend para lidar com requisições

HTTP e a lógica do servidor. Após a criação do repositório, todas as bibliotecas e dependências necessárias para o projeto, mencionadas anteriormente, foram instaladas utilizando o Yarn, garantindo um gerenciamento eficiente de pacotes.

#### **4.2.3. Processos**

O núcleo da aplicação no backend gira em torno de um script denominado `mintController.ts`, que gerencia todo o processo de geração e mintagem das NFTs. Dentro desse script, a função `generateNFTs` desempenha um papel central, recebendo como parâmetros `name`, `symbol`, `description`, `quantity` e `PK`. É essa função que coordena todas as etapas essenciais, desde a criação da coleção, passando pela geração dos metadados e das imagens, até a mintagem final das NFTs na blockchain Solana.

#### **4.2.4. Criação da coleção de NFTs**

Essa função recebe como parâmetros *name*, *symbol*, *uri* e *PK* e segue os seguintes passos:

1. Decodificação da Chave Privada: A chave privada fornecida (PK) é decodificada de base58 e usada para gerar uma conta de usuário (Keypair), que será utilizada para assinar transações na blockchain Solana.
2. Conexão à Blockchain: A função cria uma conexão com a Devnet da blockchain Solana, que é o ambiente de testes utilizado durante o desenvolvimento.
3. Configuração do Umi: O Umi é configurado com a conexão estabelecida e o plugin `mplTokenMetadata` é adicionado para interagir com os metadados dos NFTs. O usuário é autenticado por meio de sua chave secreta para que ele possa assinar transações.
4. Geração do Mint para a Coleção: Um novo signer (responsável por criar o token da coleção) é gerado. Esse mint será usado para a criação da coleção de NFTs.
5. Criação da Coleção: A função `createNft` é chamada com os parâmetros fornecidos, como nome, símbolo e URI. Esse passo cria a coleção de NFTs, que é registrada na blockchain Solana como um ativo digital.
6. Confirmação da Transação: Após criar a coleção, a transação é enviada e confirmada na blockchain.

7. Retorno do Mint da Coleção: A função retorna o mint da coleção criada, permitindo que seja utilizado posteriormente no processo de mintagem dos NFTs.

#### **4.2.5. Criação dos metadados da NFT**

Para criar os metadados de cada NFT, foi criado *createMetadata.ts*, seguindo as seguintes etapas:

1. Importação de Módulos: São importados os módulos necessários para manipulação de arquivos e caminhos no sistema, garantindo operações consistentes independentemente do ambiente.
2. Mapeamento de Traits: Um objeto mapeia as categorias de traits em plural (como "backgrounds", "eyes", "mouths") para suas versões no singular (ex: "background", "eye", "mouth"). Isso será usado nos metadados das NFTs.
3. Definição dos Caminhos: São definidos os caminhos absolutos para:
  - A pasta onde estão armazenadas as traits visuais (imagens usadas para as NFTs).
  - A pasta onde os arquivos de metadados gerados serão salvos.

Após isso, o arquivo segue o seguinte fluxo:

1. Seleção Aleatória de Traits: Uma função percorre as pastas de cada categoria de traits (como fundos, olhos, cabeças) e seleciona aleatoriamente um arquivo de imagem de cada pasta. Isso garante que cada NFT gerada seja única.
2. Geração de Traits: As traits selecionadas aleatoriamente são combinadas em um array de atributos. Cada atributo inclui o tipo da trait e o nome do arquivo da trait selecionada, que será usado nos metadados da NFT.
3. Criação de Metadados: Para cada NFT, os metadados são gerados com base no nome, símbolo, descrição, e número da NFT, além dos atributos selecionados aleatoriamente. Esses metadados incluem todas as informações necessárias para a criação da NFT.
4. Salvamento dos Metadados: Os metadados gerados para cada NFT são salvos como arquivos JSON. Os arquivos são nomeados sequencialmente e

armazenados no diretório de saída, onde estarão prontos para serem usados no processo de mintagem da NFT.

5. Verificação e Criação de Diretórios: Se o diretório onde os metadados serão salvos não existir, ele é criado automaticamente antes de salvar os arquivos.

#### **4.2.6. Geração das Imagens**

O processo de geração de imagens das NFTs no projeto utiliza a biblioteca Canvas, que permite a criação e manipulação de gráficos diretamente no servidor. A imagem de cada NFT é gerada por meio da sobreposição de camadas visuais (assets), onde cada camada representa uma característica visual da NFT, como o fundo, a base, os olhos, e a roupa.

Para gerar as imagens, o sistema segue uma ordem predefinida de camadas, garantindo que os elementos visuais sejam desenhados na sequência correta. As camadas são carregadas a partir de arquivos de imagem armazenados localmente, e, utilizando o Canvas, o sistema desenha cada camada no canvas 2D, um por vez, começando pela camada de fundo e progredindo até os detalhes finais, como os olhos e a boca.

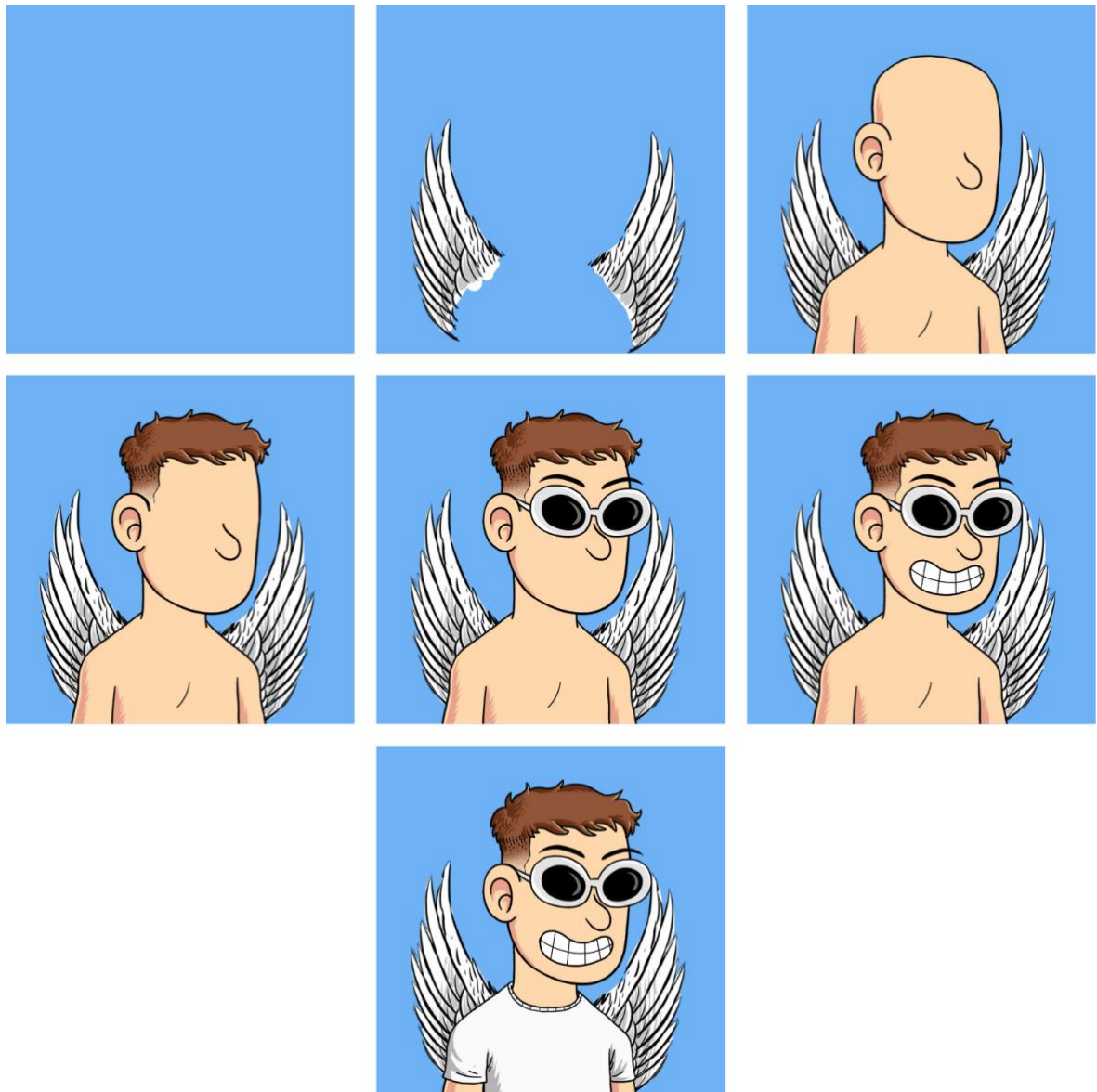
O fluxo de geração da imagem para a NFT segue desta maneira:

1. Verificação e Criação do Diretório de Imagens: O sistema verifica se o diretório onde as imagens serão salvas já existe. Caso contrário, ele cria o diretório automaticamente.
2. Criação do Canvas: Um quadro de 512x512 pixels é criado para servir como base da imagem da NFT. O contexto de desenho (2D) do quadro é definido para que as imagens possam ser desenhadas sobre ele.
3. Sobreposição de Traits por Ordem: O sistema percorre a lista de camadas, chamada `layerOrder` (Figura 2). Para cada camada ele localiza a `trait` correspondente nos metadados da NFT e carrega o arquivo de imagem associado a essa `trait`.
4. Desenho das Imagens no Canvas: Cada imagem de `trait` (como "background", "mouth", "head") é carregada e desenhada sobre o canvas na posição (0, 0),

sobrepondo as camadas na ordem correta. Isso garante que as traits sejam combinadas corretamente para formar a imagem final.

5. Conversão para Buffer: Após todas as camadas terem sido desenhadas, o canvas é convertido para um buffer de imagem no formato PNG.
6. Salvamento da Imagem: A imagem gerada é salva no diretório de saída com um nome sequencial correspondente ao índice da NFT.

Figura 2 - Processo procedural de formação da imagem da NFT



#### **4.2.7. Registro das NFTs e validação de coleção**

Após a criação das imagens e metadados de cada NFT, o sistema passa para a etapa final: a mintagem. Essa fase é crucial, pois é onde os dados gerados anteriormente são combinados e registrados de forma permanente na blockchain Solana. O processo envolve o envio dos arquivos (imagens e metadados) para um armazenamento descentralizado e a criação de uma transação que registra esses dados na blockchain.

Para garantir a integridade e autenticidade das NFTs, o sistema utiliza o Umi, uma ferramenta robusta que facilita a criação e o gerenciamento de tokens na Solana. O sistema também faz uso de um uploader descentralizado para garantir que as imagens e metadados sejam armazenados de maneira acessível e imutável. A mintagem é feita de forma automática para cada NFT, e a verificação é realizada para garantir que todos os tokens foram inseridos corretamente na blockchain:

1. **Decodificação da Chave Privada:** O processo começa com a decodificação da chave privada (PK) no formato base58. Essa chave é fundamental para gerar um par de chaves (Keypair), que será utilizado para assinar e validar as transações na blockchain Solana, garantindo que elas sejam autenticadas corretamente.
2. **Conexão com a Solana Devnet:** O sistema então se conecta à Devnet da Solana, que é o ambiente de testes utilizado durante o desenvolvimento. Esse ambiente simula as operações da blockchain real, permitindo que todo o processo de mintagem seja testado antes de ser realizado na rede principal.
3. **Configuração do Umi:** Em seguida, o Umi, uma ferramenta usada para facilitar a criação de NFTs, é configurado para se comunicar diretamente com a blockchain Solana. Além disso, são adicionados plugins importantes, como o mplTokenMetadata, que lida com os metadados das NFTs, e o irysUploader, que cuida do upload das imagens e arquivos para um armazenamento descentralizado.
4. **Processo de Mintagem por NFT:**

- Para cada NFT, o sistema começa lendo o arquivo de metadados JSON correspondente, que contém informações como o nome, a descrição e os atributos visuais da NFT.
  - Em seguida, a função `uploadUri` é chamada para fazer o upload da imagem associada à NFT, utilizando o serviço `irysUploader`. A imagem é carregada do diretório de saída e enviada para um serviço de armazenamento descentralizado, como o Ardrive.
  - Depois, o URI (link) da imagem armazenada é adicionado aos metadados da NFT. Em seguida, esses metadados atualizados são enviados ao serviço de upload para obter o URI final, que será vinculado ao token da NFT.
5. Geração do Mint da NFT: Para cada NFT gerada, um novo signer (ou mint) é criado. Esse mint é a identidade única da NFT na blockchain, sendo necessário para registrar o token como um ativo digital na rede.
  6. Criação e Envio da Transação: A função `createNft` é então chamada para registrar oficialmente a NFT na blockchain Solana. A transação inclui o URI dos metadados, o nome, o símbolo, e a chave da coleção à qual a NFT será associada. A transação é enviada para a blockchain, onde será confirmada.
  7. Verificação do Ativo Digital:
    - Após o envio da transação, o sistema tenta recuperar os dados da NFT diretamente da blockchain, usando a função `fetchDigitalAsset`.
    - Caso haja dificuldades na recuperação, o sistema faz várias tentativas para garantir que os dados da NFT sejam obtidos corretamente.
  8. Verificação e Registro da NFT: Uma vez que a NFT é criada com sucesso, a função `verifyNFT` é chamada para confirmar que a NFT foi inserida corretamente na coleção, validando o processo. O sistema também exibe o endereço público da NFT criada na blockchain
  9. Tratamento de Erros: Se ocorrer algum problema durante qualquer parte do processo, o sistema exibe uma mensagem de erro para que o problema possa ser identificado e resolvido.

Depois que o processo de validação da coleção é concluído e todas as NFTs são registradas corretamente na blockchain, o backend envia uma confirmação de sucesso ao frontend. Essa mensagem é enviada via requisição HTTP, informando que



todo o processo foi finalizado com sucesso. Assim, o frontend pode tranquilamente informar ao usuário que suas NFTs estão prontas e registradas na Solana, completando o ciclo de criação.

### **4.3. Frontend**

O frontend é a interface que conecta o usuário ao sistema, facilitando o envio das informações necessárias para a criação das NFTs. Ele foi projetado para ser simples e intuitivo, permitindo que o usuário insira dados como o nome da coleção e a quantidade de NFTs a serem geradas de maneira prática. Todo o fluxo no frontend é focado em garantir que o processo de interação com o backend seja eficiente e sem complicações, oferecendo uma experiência fluida ao usuário.

Além de coletar as informações, o frontend envia esses dados ao backend e, em seguida, recebe uma resposta que informa se o processo de criação e mintagem das NFTs foi concluído com sucesso. O objetivo principal é garantir que o usuário tenha uma visão clara do status da operação e, ao final, dos detalhes das NFTs geradas.

Para uma visualização completa de todo o código que implementa essas funcionalidades, consulte o Apêndice B onde estão detalhados os scripts que suportam o frontend do sistema.

O fluxo de trabalho do frontend segue as seguintes etapas:

1. Coleta dos dados do usuário
2. Envio das informações ao backend via requisição HTTP
3. Recebimento da confirmação de sucesso ou falha do backend
4. Exibição dos detalhes das NFTs geradas para o usuário

Ao final do processo, o frontend exibe ao usuário uma mensagem indicando o sucesso ou a falha da operação, junto com as informações completas das NFTs criadas.

#### **4.3.1. Bibliotecas utilizadas**

- **axios:** Biblioteca para fazer requisições HTTP. É amplamente utilizada para comunicação entre o frontend e o backend, permitindo que o frontend envie e receba dados, como as informações das NFTs geradas.
- **tailwindcss:** Um framework de utilitários CSS que permite a criação rápida de interfaces de usuário. Ele fornece classes pré-definidas que facilitam a estilização de componentes de forma eficiente e responsiva.

#### **4.3.2. Criação e organização do projeto**

O desenvolvimento do frontend começou com a criação de uma aplicação utilizando Next.js com TypeScript e Tailwind CSS. O comando utilizado para gerar a estrutura inicial do projeto foi:

```
npx create-next-app@latest tcc-frontend --typescript
```

Esse comando gerou automaticamente a estrutura básica do frontend, integrando o framework Next.js com TypeScript, o que facilita o desenvolvimento de interfaces reativas e otimizadas para a web. Em seguida, o Tailwind CSS foi configurado para gerenciar a estilização dos componentes, oferecendo uma abordagem baseada em utilitários para criar interfaces rápidas e responsivas.

Após a criação da aplicação, todas as bibliotecas e dependências mencionadas anteriormente foram instaladas utilizando o Yarn, garantindo um gerenciamento eficiente dos pacotes e uma boa manutenção do código ao longo do projeto.

#### **4.3.3. Criação dos arquivos e processos**

O frontend foi construído utilizando a estrutura padrão gerada pelo comando de criação do Next.js mencionado anteriormente. A partir dessa estrutura base, foram feitas apenas algumas modificações simples para adequar o projeto às necessidades do sistema. Foi incluída uma plataforma para realizar as requisições HTTP ao backend, permitindo o envio dos dados inseridos pelo usuário. Além disso, o arquivo `index.tsx` foi modificado para receber essas informações, coletando os dados

necessários, como o nome da coleção e a quantidade de NFTs, e os enviando ao backend para processamento.

#### **4.3.4. Serviço de requisição HTTP**

Foi criado o arquivo `api.ts` dentro da pasta `services` para gerenciar as requisições HTTP que o frontend realiza ao backend. Esse arquivo utiliza a biblioteca `axios` para configurar uma instância de API, definindo a base URL como `http://localhost:8080`, que é o endereço do backend. A partir dessa instância, o frontend pode realizar requisições de maneira centralizada e organizada, garantindo que todas as interações com o backend, como envio de dados e recebimento de respostas, sejam feitas de forma eficiente e com o mínimo de código repetitivo.

#### **4.3.5. Homepage da aplicação**

O arquivo `index.tsx`, responsável pela página principal, foi modificado para receber e gerenciar os dados inseridos pelo usuário. Ele utiliza a função `useState` do React para armazenar os valores de entrada, como nome, símbolo, descrição, quantidade e chave privada (PK), e oferece uma interface amigável para que o usuário possa preencher esses dados.

Além disso, foi implementada a lógica de envio de requisições HTTP ao backend. Quando o formulário é enviado, os dados são capturados e uma requisição POST é feita à API configurada no arquivo `api.ts`. Dependendo do resultado da resposta, o sistema exibe mensagens no console para confirmar se a coleção de NFTs foi criada com sucesso ou se houve algum erro durante o processo.

Essas alterações garantem que o frontend colete as informações necessárias de maneira intuitiva e faça a comunicação correta com o backend, facilitando todo o processo de criação das NFTs.

### **5. TESTE DO PROCESSO**

Para validar o sistema desenvolvido, foi realizado um teste prático utilizando dados fictícios (Figura 3). A interface do frontend foi utilizada para a coleta dos dados necessários à criação das NFTs. Foram inseridas informações de exemplo, como o nome da coleção, o símbolo, a descrição e a quantidade de NFTs desejada. Em

seguida, foi acionado o botão "Criar coleção" para iniciar o processo de geração e mintagem das NFTs.

Figura 3 - Exemplo da interface com dados fictícios



**Criar Coleção de NFTs**

Nome da coleção  
Colecao exemplo

Símbolo da coleção  
UFU

Descrição da coleção  
Colecao criada como exemplo para projeto de fim de curso

Quantidade de NFTs  
10

Private Key  
5EL5RJmtCU1nmQ34SeJCD51TnsRj17WkQqcpE

**Criar Coleção**

Os logs do backend detalham o processo de comunicação entre o frontend e o backend durante a geração e mintagem das NFTs na blockchain Solana (Figura 4). Inicialmente, o sistema recebe os dados enviados pelo frontend, que são utilizados para criar a coleção. Em seguida, o *mint address* da coleção é retornado, seguido pela criação e retorno dos *mint addresses* individuais das NFTs geradas. Inicialmente, é possível observar o recebimento dos dados enviados pelo frontend, que servem como entrada para a criação da coleção e das NFTs. Em seguida, o sistema gera a coleção e retorna o mint address dela, evidenciando o sucesso dessa etapa. Logo abaixo, são

exibidos os mint addresses individuais das NFTs, que foram criadas e inseridas na blockchain com sucesso.

Além disso, os logs registram algumas falhas que ocorreram durante o processo, indicando instabilidade momentânea na rede Solana. No entanto, o script desenvolvido foi capaz de lidar com essas falhas, retomando o processo automaticamente até que todas as NFTs fossem mintadas corretamente. Isso demonstra a robustez e resiliência do sistema, que conseguiu concluir o processo mesmo diante de oscilações na rede.

Figura 4 - Logs do backend do processo de criação da coleção.

```
Generating NFTs... {
  name: 'Colecao exemplo',
  symbol: 'UFU',
  description: 'Colecao criada como exemplo para projeto de fim de curso',
  quantity: '10',
  PK: '5EL5RJmtCU1nmQ34SejCD51TnsRj17WkQqcpEyVkGDfnHqDrRexrsKtAE5YG9rRaSNQdrPAvxrq546WPtTuWT98w'}

Creating collection...
Collection mint: 8HR9Bfna88gd2LrmSXgaiHkKEWY6ZhEUd6gHdAJ7Ets1
Created NFT 3iAaXAJGVn2FmYAVgGCUPHE1KVc9DNKDXB5VgZYJfEDP
Attempt 1 failed. Retrying...
Attempt 2 failed. Retrying...
Attempt 3 failed. Retrying...
Attempt 4 failed. Retrying...
Created NFT 8WRtagGnpd3Yeb1ZbyLMV2UKmNTaamXPxZv7U63MvBtd
Created NFT 74yZYBiGcSjQYahWcVEVkwRebFkVhSdB4BSD11NrPrc6
Attempt 1 failed. Retrying...
Attempt 2 failed. Retrying...
Attempt 3 failed. Retrying...
Attempt 4 failed. Retrying...
Created NFT GMxTXx4Bwx5DHgz2zsDkqSxSW97UyW8YvxeZ7pQc9iFS
Attempt 1 failed. Retrying...
Attempt 2 failed. Retrying...
Attempt 3 failed. Retrying...
Attempt 4 failed. Retrying...
Created NFT 2bDBCvC2QAqtpSW828ReqzrH5J8yrSczqGDaUpfWGYUA
Created NFT 6ce5aNE2Mc8ewVQGEZT7ph1WEz3BEuFiDh1yKffb5VDi
Created NFT Bwtfc1Dm89aDpgPvt3EvvG2naZzMitvpnHLz2GdR9N9w
Attempt 1 failed. Retrying...
Attempt 2 failed. Retrying...
Attempt 3 failed. Retrying...
Attempt 4 failed. Retrying...
Created NFT VTXdAViJGyA4r3BjmJ9SRvEqov54PQAQkBwFiQdXbtf
Created NFT 43aZqdZSrino6oko61q4sCo77Jj7FVTvVsvXxWugj6gg
Created NFT 9S8gt26eReMTkqzCAQ57VgKeDPR9cqjtTP2CaZyxthCF
```

## 6. RESULTADOS E CONCLUSÃO

O sistema desenvolvido funcionou de maneira eficaz em todos os testes realizados. O frontend conseguiu coletar os dados inseridos pelo usuário de forma simples e prática, como o nome da coleção, o símbolo, a descrição e a quantidade de NFTs, e transmitiu essas informações corretamente para o backend. Este, por sua vez, processou os dados com eficiência e deu início ao processo de mintagem das NFTs na blockchain Solana.

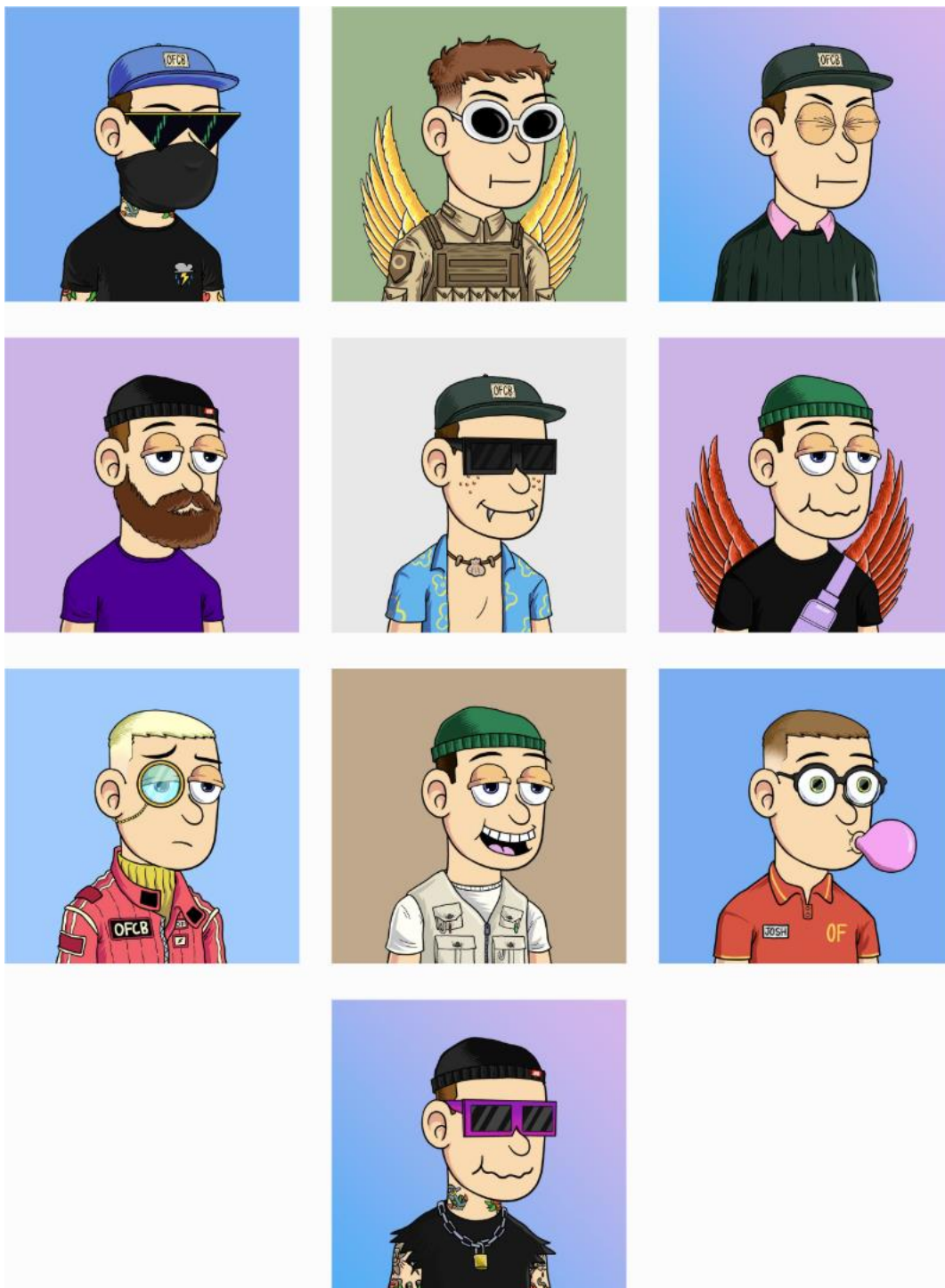
Apesar de o sistema ser robusto, a rede Solana apresentou algumas instabilidades durante os testes, o que pode causar atrasos ou até falhas temporárias no processo de mintagem. Mesmo assim, o sistema foi projetado para lidar bem com essas situações, e a maioria das operações foi concluída com sucesso.

Importante destacar que o código foi testado na Devnet da Solana, uma rede de testes, o que significa que não houve custos reais envolvidos na mintagem das NFTs. No entanto, caso o sistema seja migrado para a Mainnet (a rede principal da Solana), será necessário prestar atenção aos custos associados às transações.

Em resumo, os resultados indicam que o sistema é capaz de gerar e registrar NFTs de maneira confiável e eficiente, cumprindo os objetivos propostos.

Em resumo, os resultados indicam que o sistema é capaz de gerar e registrar NFTs de maneira confiável e eficiente, cumprindo os objetivos propostos. A Figura 6 representa o resultado do teste feito com dados fictícios, mostrando as 10 imagens geradas pelo script.

Figura 5 - NFTs geradas pelo script com dados fictícios



## 7. REFERÊNCIAS BIBLIOGRAFICAS

ANTONPOULOS, Andreas M. **Mastering Bitcoin: Unlocking Digital Cryptocurrencies**. 2. ed. Sebastopol: O'Reilly Media, 2017.

ATZORI, M. **Blockchain Technology and Decentralized Governance: Is the State Still Necessary?** Journal of Peer Production, v. 7, p. 1-14, 2015.  
<https://doi.org/10.2139/ssrn.2709713>

BLOCKAPPS INC. **NFT Art and Digital Ownership**. 2023. Disponível em: <https://www.blockapps.net/nft-art-and-ownership>. Acesso em: 5 out. 2024.

CHEN, K. **CryptoPunks and the NFT Revolution: A Historical Analysis**. Journal of Blockchain Research, v. 4, n. 2, p. 130-145, 2021.

CULT MTL. **NFTs: Empowering Artists and Democratizing the Art World**. 2023. Disponível em: <https://cultmtl.com/nfts-empowering-artists>. Acesso em: 5 out. 2024.

HAYES, Adam. **Blockchain Definition: What You Need to Know**. Investopedia, 2022. Disponível em: <https://www.investopedia.com/terms/b/blockchain.asp>. Acesso em: 8 out. 2024.

IBM. **What Are Smart Contracts on Blockchain?** 2023. Disponível em: <https://www.ibm.com/topics/smart-contracts>. Acesso em: 8 out. 2024.

NAKAMOTO, Satoshi. **Bitcoin: A Peer-to-Peer Electronic Cash System**. 2008. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acesso em: 8 set. 2024.

NFTMETRIA. **The Art World Revolution: NFTs and Their Impact on Traditional Art Markets**. 2023. Disponível em: <https://nftmetria.com>. Acesso em: 5 out. 2024.

PÉTERFAY, Eszter. **Creating a Website for a Programmatically Generated NFT Collection: Solana Network**. HAMK University, 2022.



PETERS, G. W.; PANAYI, E. **Understanding Modern Banking Ledgers through Blockchain Technologies: Future of Transaction Processing and Smart Contracts on the Internet of Money.** Journal of Banking and Finance, v. 6, n. 2, p. 15-22, 2016. <https://doi.org/10.2139/ssrn.2692487>

REINICKE, C. **One in 10 people in the U.S. are invested in cryptocurrencies.** CNBC. 2021. Disponível em: <https://www.cnbc.com/2021/08/23/one-in-10-people-in-the-us-are-invested-in-cryptocurrencies.html>. Acesso em: 5 out. 2024.

SOLANA FOUNDATION. **Developer Resources: Solana Clusters.** 2021. Disponível em: <https://docs.solana.com/clusters>. Acesso em: 8 out. 2024.

TSOUTSOS, N.; MAKRIS, Y. **Secure FPGA-based Execution of Smart Contracts for Blockchain Systems.** Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018.

WANG, Q. et al. **Non-fungible token (NFT): Overview, evaluation, opportunities and challenges.** Journal of Information Technology, v. 5, p. 78-94, 2021.

YAKOVENKO, Anatoly. **Solana: A New Architecture for a High Performance Blockchain.** 2020. Disponível em: <https://solana.com/solana-whitepaper.pdf>. Acesso em: 8 set. 2024.