

Smart Agro RAF API

Henrique Fan¹, Rafael Nogueira¹, Bruno B. Neves², Fábio R. da Silva¹,
Roben C. Lunardi³, Diego Kreutz¹, Rodrigo B. Mansilha¹

¹Universidade Federal do Pampa - Campus Alegrete (UNIPAMPA)

²Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)

³Instituto Federal do Rio Grande do Sul (IFRS)

{henriquefan.aluno, rafaelnogueira.aluno}@unipampa.edu.br

neves.bruno@edu.pucrs.br

roben.lunardi@restinga.ifrs.edu.br

{fabiosilva.aluno, diegokreutz, rodrigomansilha}@unipampa.edu.br

Abstract. *The Smart Agro RAF API is an architecture and implementation designed to assist family farming in meeting the traceability requirements established by Instrução Normativa Conjunta (INC) No. 02/2018. The solution leverages blockchain technology to ensure data transparency, auditability, and security. The API functions as an abstraction layer, providing a unified interface that simplifies interaction with the distributed infrastructure and mitigates the technical complexity associated with using ERC-721 and ERC-1155 contracts without requiring modifications to the client logic. The solution was evaluated through an automated unit testing suite, which validated compliance with functional requirements and enabled traceability operations.*

Resumo. *Neste trabalho propomos uma arquitetura, denominada Smart Agro RAF API, para auxiliar a agricultura familiar a cumprir os requisitos de rastreabilidade impostos pela Instrução Normativa Conjunta (INC) nº 02/2018. A solução utiliza a tecnologia de contratos inteligentes baseados em blockchain para assegurar a transparência, auditabilidade e segurança dos dados. A API atua como uma camada de abstração, fornecendo uma interface unificada que simplifica a interação com a infraestrutura distribuída e permite mitigar a complexidade técnica associada ao uso de contratos baseados nos padrões ERC-721 e ERC-1155 sem a necessidade de modificar a lógica do cliente. A avaliação da solução foi realizada por meio de uma suíte de testes unitários automatizados, que validou a aderência aos requisitos funcionais, viabilizando as operações de rastreabilidade.*

1. Introdução

A agricultura familiar, que constitui a maioria (77%) dos estabelecimentos rurais brasileiros e é economicamente vital para a grande parte dos municípios menores [IBGE 2017], é um segmento de escala global, com pequenas propriedades representando cerca de 84% das mais de 608 milhões de fazendas no mundo [Lowder et al. 2021]. Apesar de sua relevância estratégica, esse setor enfrenta obstáculos significativos para a inovação em seus processos [Santos et al. 2024b, Santos et al. 2024a]. Este desafio é

agravado por exigências regulatórias como a Instrução Normativa Conjunta (INC) nº 02/2018 [BRASIL 2018], que impõe requisitos de rastreabilidade de vegetais. Tais normas impactam diretamente políticas públicas dependentes desse setor, como o Programa Nacional de Alimentação Escolar (PNAE), que destinou aproximadamente R\$ 1,65 bilhão da agricultura familiar em 2023, cumprindo a determinação legal de aplicar ao menos 30% dos recursos (\$5,5 bilhões totais) nessa modalidade de produção [BRASIL 2009, BRASIL 2023]. O projeto Proraf¹ tem como objetivo o desenvolvimento de um sistema de rastreabilidade para a agricultura familiar, com base na exploração de tecnologias emergentes para superar os desafios anteriormente citados.

Recentemente, propomos e avaliamos a economicidade da adoção de tecnologia de contratos inteligentes baseados em *blockchain* [Silva et al. 2025] como uma solução para auxiliar a agricultura familiar no cumprimento dos requisitos da INC 02/2018. A natureza descentralizada e imutável da *blockchain* [Chervinski et al. 2019, Al-Jaroodi and Mohamed 2019] oferece os pilares fundamentais de transparência, auditabilidade e segurança dos dados, aumentando a resiliência das informações de rastreabilidade [Dedeoglu et al. 2020]. Essa infraestrutura de confiança tem o potencial de agregar valor a toda a cadeia produtiva. Ademais, as características de não repúdio e autenticidade inerentes ao livro-razão distribuído podem facilitar significativamente a execução e auditoria de políticas públicas vinculadas ao setor [Lunardi et al. 2024].

Para viabilizar a integração de contratos inteligentes com aplicações web e móveis, é fundamental o desenvolvimento de uma API web. Com esse objetivo, este trabalho propõe uma arquitetura e a implementação de uma API denominada *Smart Agro RAF API*. Essa camada de abstração disponibiliza funcionalidades críticas, como criação, consulta e transferência de ativos digitais, ao mesmo tempo em que reduz a complexidade técnica do uso direto da infraestrutura de *blockchain*. Além disso, a API favorece a interoperabilidade e a escalabilidade do sistema, permitindo a integração transparente de diferentes aplicações à solução de rastreabilidade e ocultando a complexidade inerente à lógica dos contratos inteligentes e aos mecanismos de baixo nível da infraestrutura distribuída. O trabalho apresenta a análise de requisitos e a arquitetura proposta, cuja avaliação inicial é realizada por meio de testes sistemáticos.

O restante deste trabalho está organizado como segue. Iniciamos com discussão de conceitos fundamentais para tornar o trabalho auto-contido e discussão de esforços similares na literatura na Seção 2. Apresentamos uma solução e uma avaliação nas seções 3 e 4, respectivamente. Considerações finais são apresentadas na Seção 5.

2. Fundamentação Técnica e Teórica

2.1. Trabalhos Relacionados

No estudo de [Doan and Thiemann 2021], é apresentada uma solução para a limitação dos contratos inteligentes, que não interagem nativamente com sistemas de software tradicionais. Os autores destacam que as interfaces de chamada de procedimento remoto (Remote Procedure Call - RPC) existentes, como as da Ethereum e da Tezos, operam em baixo nível, exigem manipulação manual de dados em formato JSON e não oferecem garantias de segurança de tipo estática. A proposta implementada permite interações

¹<https://proraf.com.br>

programáticas e seguras com a *blockchain* por meio de uma API projetada para suportar tanto a criação e invocação de contratos inteligentes quanto a observação do estado da rede. A abordagem também introduz um modelo de execução que garante consistência e segurança nas interações entre sistemas externos e a infraestrutura distribuída.

Já [Park et al. 2023] busca reduzir a complexidade do uso direto de contratos inteligentes com o uso de uma camada intermediária baseada em APIs RESTful. O foco recai sobre a reutilização e o gerenciamento de contratos inteligentes em ambientes Hyperledger Fabric, estruturando-os por meio de *tags* que possibilitam busca, comparação e adaptação a diferentes contextos. Essa estratégia contribui para a padronização e para uma gestão mais sistemática de contratos em redes permissionadas, favorecendo reuso e eficiência no desenvolvimento.

Por sua vez, [Pasdar et al. 2023] discute o chamado “problema do oráculo”, uma das principais limitações dos contratos inteligentes, decorrente da incapacidade das *blockchains* de acessarem dados externos (*off-chain*). O trabalho analisa o uso de APIs como camada de abstração para integrar informações externas aos contratos *on-chain*. Os autores classificam diferentes abordagens de oráculos, explorando modelos *push* e *pull*, arquiteturas centralizadas e descentralizadas, além de mecanismos de autenticação e validação. A discussão contempla impactos em termos de segurança, confiabilidade, latência e escalabilidade, ressaltando as limitações específicas de cada alternativa.

Apesar das diferenças de escopo e aplicação, há convergência entre os trabalhos mencionados e a proposta aqui apresentada. Todos reconhecem a complexidade do uso direto de contratos inteligentes e propõem soluções baseadas em APIs ou camadas intermediárias para atenuar essa barreira. Neste estudo, assim como em [Park et al. 2023], são propostos mecanismos que simplificam a interação de aplicações externas com contratos inteligentes, seja por meio da abstração de operações específicas ou pela organização e reutilização de contratos. De forma semelhante, [Pasdar et al. 2023] e [Doan and Thiemann 2021] destacam o papel das interfaces programáticas como ponto de integração, seja para conectar contratos a dados externos por meio de oráculos, seja para garantir interações seguras e tipadas com a infraestrutura distribuída. Em conjunto, tais iniciativas reforçam a relevância de abstrações de alto nível para tornar a *blockchain* mais acessível, interoperável e confiável em diferentes domínios. A particularidade deste trabalho é seu foco na agricultura familiar, explorando o uso de contratos inteligentes em um contexto social e econômico específico.

2.2. Conceitos

A seguir apresentamos alguns conceitos para tornar o trabalho auto-contido.

- **INC 02/2018:** define um conjunto de informações que devem ser registradas e mantidas durante as transações de compra e venda de produtos vegetais, conforme apresentado na Tabela 1. Observe-se que os campos são basicamente os mesmos para entrada e saída, alterando apenas os termos sublinhados dependendo do ponto de vista da pessoa que registra (comprador ou fornecedor).
- **Requisitos Funcionais:** Para realizar rastreamento da cadeia produtiva da agricultura familiar em conformidade com a INC 02/2018 usando *smart contracts* consideramos os requisitos funcionais [Silva et al. 2025] listados na Tabela 2.

Tabela 1. Dados necessários conforme INC 02/2018 [BRASIL 2018].

1 - Informações do Produto Vegetal	1 - Informações do Produto Vegetal
1.1 - Nome do produto vegetal	1.1 - Nome do produto vegetal
1.2 - Variedade ou cultivar	1.2 - Variedade ou cultivar
1.3 - Quantidade do produto <u>recebido</u>	1.3 - Quantidade do produto <u>expedido</u>
1.4 - Identificação do lote	1.4 - Identificação do lote
1.5 - Data do <u>recebimento</u> do produto vendido	1.5 - Data da <u>expedição</u> do produto vendido
2 - Informações do <u>Fornecedor</u>	2 - Informações do <u>Comprador</u>
2.1 - Nome ou Razão Social	2.1 - Nome ou Razão Social
2.2 - CPF, I.E ou CNPJ ou CGC/MAPA	2.2 - CPF, I.E ou CNPJ ou CGC/MAPA
2.3 - Endereço completo, ou quando localizado em zona rural, coordenada geográfica ou CCIR	2.3 - Endereço completo, ou quando localizado em zona rural, coordenada geográfica ou CCIR

Tabela 2. Requisitos Funcionais do Sistema.

ID	Descrição do Requisito Funcional
RF1	Inserção de <i>tokens</i> dos ativos gerados pelo produtor. Antes que um produto seja movimentado (por exemplo, para venda), ele deve ser agrupado em um lote de produção e receber um conjunto de informações junto a um <i>token</i> único. <i>Tokens</i> derivados podem então ser gerados a partir do <i>token</i> /lote original para identificar itens específicos dentro do lote.
RF2	Revogação de <i>tokens</i>. Em caso de erro ou quando um produto/lote deixa de existir, deve haver uma funcionalidade para revogar o <i>token</i> correspondente.
RF3	Transferência de <i>tokens</i>. Lotes podem ser movimentados entre entidades gerando uma cadeia de rastreabilidade. Para isso, é necessário que as informações de transação sejam registradas para rastreamento.
RF4	Consulta de movimentação de <i>tokens</i>. Lotes e produtos podem ser identificados e rastreados baseados no <i>token</i> e registros de transferência de <i>tokens</i> . As informações dos produtos, lotes e movimentações devem ser públicas.

- **Padrões ERC (*Ethereum Request for Comments*):** estabelecem convenções para o desenvolvimento de contratos inteligentes na Ethereum, garantindo interoperabilidade, segurança e adoção consistente de funcionalidades pela comunidade. Neste contexto, escolhemos dois padrões candidatos (ERC-721 e ERC-1155), ambos amplamente utilizados e auditados no ecossistema. Cada um atende a necessidades específicas de rastreabilidade e escala, mas compartilham fundamentos comuns essenciais para a representação confiável de ativos digitais na *blockchain*, conforme exigido pela INC. A Tabela 3 detalha as características desses padrões, evidenciando tanto seus aspectos compartilhados quanto suas particularidades.

Tabela 3. Comparação entre os padrões ERC-721 e ERC-1155 implementados.

Aspecto	ERC-721	ERC-1155
Otimização	Rastreamento de lotes únicos e individualizados	Eficiência operacional em larga escala
Vantagem	Garante singularidade e rastreabilidade	Agrega múltiplos itens sob um mesmo contrato
Complexidade de gerenciamento	Menor	Maior

3. Smart AgroRAF API

A *Smart Agro RAF API* funciona como uma camada de abstração entre as aplicações clientes e os contratos inteligentes, simplificando significativamente a interação com a infraestrutura distribuída subjacente. Esta seção detalha inicialmente os requisitos que orientaram seu desenvolvimento, para em seguida apresentar sua arquitetura e componentes propostos.

3.1. Análise de Requisitos

A Figura 1 ilustra uma visão geral de como a *Smart Agro RAF API* interage com os demais componentes do sistema, destacando os requisitos funcionais que orientam sua arquitetura. No contexto do RF1, a API opera como uma camada de abstração que permite alternar entre os diferentes padrões de contratos (ERC-721 ou ERC-1155), sem demandar alterações na lógica do cliente. O RF2 contempla *endpoints* destinados à leitura, que não exigem autorização, permitindo consultas a informações sobre ativos digitais e históricos de transações, reforçando a transparência e a rastreabilidade inerentes ao uso da tecnologia *blockchain*.

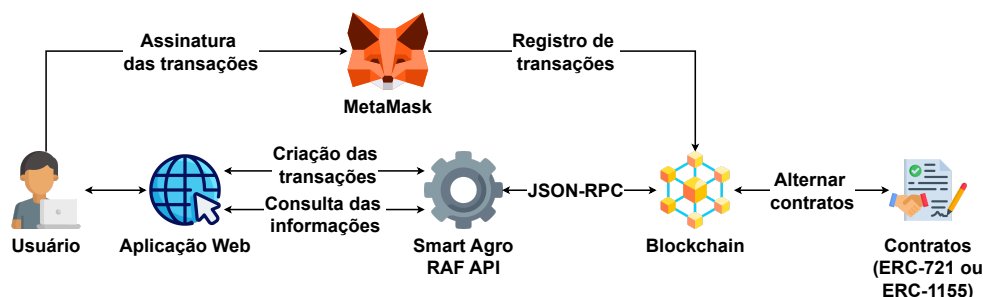


Figura 1. Visão Geral da Proposta.

Por fim, o RF3 destaca o papel da API na geração de transações, que são encaminhadas ao *frontend* para assinatura pelo usuário. Esse processo é viabilizado pelo gerenciador de carteiras MetaMask², extensão de navegador amplamente consolidada no ecossistema Ethereum, garantindo segurança e restrição às operações apenas para usuários autorizados pelo endereço de suas carteiras. Consideramos os seguintes requisitos funcionais no projeto da *Smart Agro RAF API*.

RF1: Alternância de Padrão de Contrato. A API deve fornecer uma interface unificada que permita alternar entre diferentes padrões de contrato sem necessidade de modificar a lógica do cliente.

RF2: Operações de Leitura. A API deve expor endpoints para consultar dados de ativos digitais e histórico de transações de forma eficiente e consistente.

RF3: Criação de Transações. A API deve gerar transações prontas para serem assinadas pelos usuários no *frontend* como o MetaMask, sem executar diretamente alterações na *blockchain*. A criação de transações deve ser restrita a usuários autorizados, garantindo controle e segurança no processo. Além de tratar e informar erros de forma clara, garantindo *feedback* confiável para o cliente.

²<https://metamask.io/>

3.2. Arquitetura Proposta

A API é estruturada segundo o paradigma REST, disponibilizando *endpoints* para operações de criação, revogação, transferência e consulta de *tokens* digitais, assegurando aderência às exigências da INC. Com isso, reduz-se a complexidade técnica associada ao uso direto da *blockchain*, fornecendo às aplicações do Produtor e do Consumidor os dados necessários para suas operações.

No modelo arquitetural em que as transações são pré-construídas no *backend* e assinadas no cliente, a responsabilidade pela estruturação da transação é atribuída ao *backend*, que define parâmetros fundamentais como o endereço do contrato inteligente, a codificação da chamada de função (*data*), o valor a ser transferido (*value*) e as estimativas de consumo computacional (*gas*). Uma vez preparada, a transação é enviada ao *frontend*, onde é apresentada ao usuário para assinatura e posterior submissão à rede *blockchain* por meio de uma carteira compatível, como a MetaMask. Essa abordagem centraliza a lógica de negócios e assegura a conformidade das transações com as regras da aplicação, ao mesmo tempo em que preserva a segurança das chaves privadas, já que a assinatura e o envio da transação ocorrem exclusivamente no ambiente do cliente.

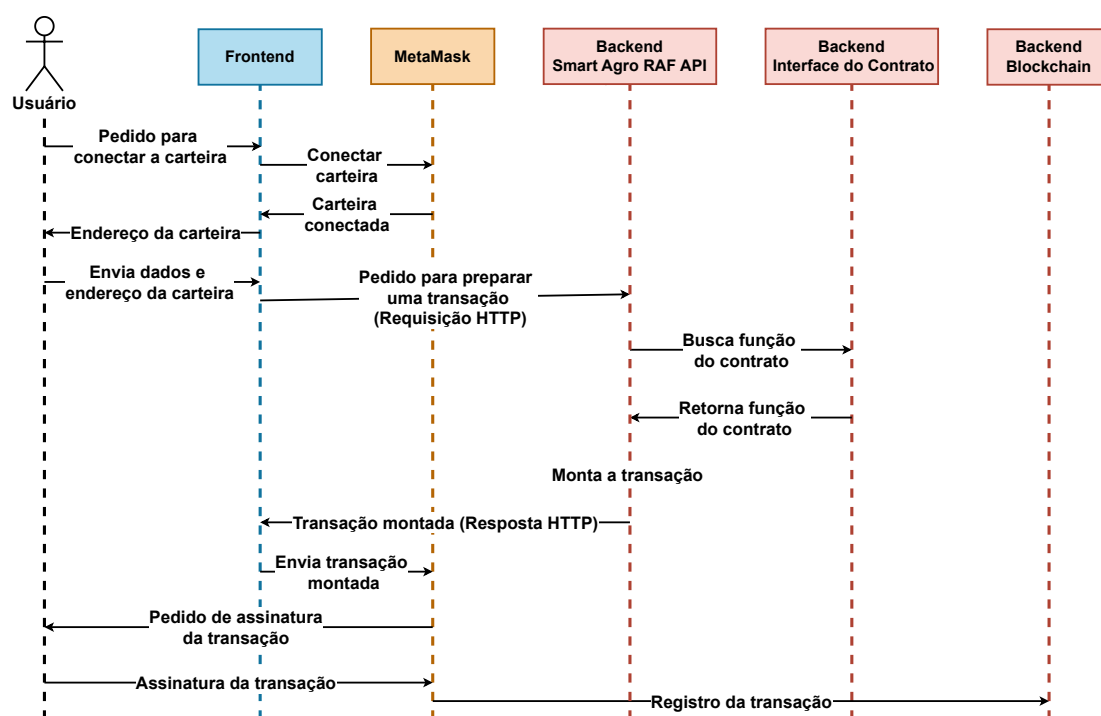


Figura 2. Diagrama de sequência ao registrar uma informação.

A Figura 2 apresenta o diagrama de sequência que descreve o fluxo de integração entre os componentes da solução proposta durante a construção e execução de uma transação na *blockchain*. O processo tem início com a solicitação do usuário para conexão da carteira digital, realizada pelo *frontend* e intermediada pela MetaMask, que retorna o endereço da carteira. Após a coleta dos dados necessários, o *frontend* envia uma requisição à API, responsável por interagir com a interface do contrato inteligente. Esta localiza e estrutura a chamada da função correspondente, utilizando os parâmetros fornecidos. A API então monta a transação, encapsulando elementos como endereço do

contrato, dados da função, valor e estimativa de consumo computacional, retornando-a ao *frontend* via resposta HTTP. A transação é apresentada ao usuário para assinatura local via MetaMask, garantindo que a chave privada permaneça sob seu controle. Após a assinatura, a transação é enviada à *blockchain*, finalizando o fluxo com o registro imutável da informação na rede distribuída.

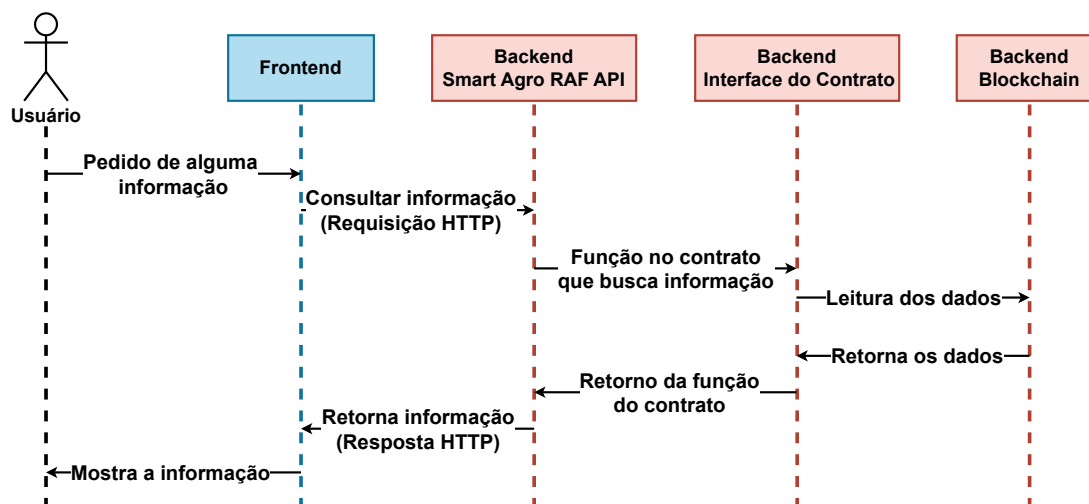


Figura 3. Diagrama de sequência ao consultar uma informação.

A Figura 3 ilustra o fluxo de execução de uma operação de leitura de dados armazenados em um contrato inteligente. Esse processo representa uma chamada de função que não envolve transações ou alterações de estado, permitindo apenas a obtenção de dados *on-chain*. O fluxo inicia-se com a solicitação do usuário por uma informação específica, realizada por meio do *frontend*. Em seguida, o *frontend* encaminha a requisição diretamente à API, que interage com a interface do contrato acionando a função correspondente. Os dados retornados são enviados de volta à API e propagados até o *frontend*, onde são apresentados ao usuário.

4. Avaliação

A avaliação da *Smart Agro RAF API* foi conduzida por meio de uma suíte de testes unitários automatizados. Foram validadas funcionalidades críticas, incluindo a instância de contratos inteligentes, a geração e codificação de transações, a manipulação de *tokens* nos padrões ERC-721 e ERC-1155, bem como a consistência das operações de leitura e escrita. Adicionalmente, a camada de API foi avaliada quanto ao tratamento de erros e padronização de respostas, assegurando conformidade com protocolos HTTP. Os resultados indicaram cobertura satisfatória dos cenários de uso e aderência da solução aos requisitos funcionais definidos, demonstrando que a API abstrai de forma eficaz a complexidade da interação com a *blockchain* e viabiliza operações de rastreabilidade de maneira segura e confiável.

4.1. Implementação

A API foi implementada em TypeScript, linguagem amplamente utilizada e alinhada às boas práticas de engenharia de software. A tipagem estática oferecida pelo TypeScript

contribui para maior segurança, modularidade, escalabilidade e facilidade de manutenção. Para interação com a *blockchain*, adotou-se a biblioteca `ethers.js`, reconhecida pela maturidade e integração eficiente com o ecossistema Ethereum, formando em conjunto com o TypeScript uma base consolidada para o desenvolvimento de aplicações descentralizadas (DApps). Embora o modelo de concorrência do Node.js apresente algumas limitações, estas não comprometem de forma significativa o desempenho ou a escalabilidade da API. O ambiente de desenvolvimento foi configurado com o Hardhat, cuja integração nativa com o `ethers.js` permite automação de testes e *deployments*, reduzindo a complexidade do ciclo de desenvolvimento. A escolha desse conjunto de tecnologias buscou equilibrar produtividade, robustez e suporte comunitário, assegurando bom desempenho e manutenção a longo prazo.

4.2. Metodologia

A suíte de testes foi desenvolvida em TypeScript e executada no ambiente Node.js com o *framework* Jest, utilizando o `ts-jest` para transpilação. A metodologia abrange diferentes níveis de abstração, garantindo robustez e corretude funcional. Nos testes unitários, a interação com contratos inteligentes foi verificada por meio de `mocks` que isolam a lógica de negócio, contemplando aspectos como instanciamento de contratos, validação de endereços Ethereum, carregamento de metadados, tratamento de erros e codificação de transações. Foram avaliadas tanto funcionalidades genéricas quanto específicas dos padrões ERC-721 e ERC-1155, incluindo consultas de *tokens* e formatação de dados. Na camada de API, construída com `Express.js`, os testes asseguraram o mapeamento correto de *endpoints* e a resposta adequada a falhas, garantindo rejeição de requisições inválidas com HTTP 400 e padronização de exceções com HTTP 500. Adicionalmente, os esquemas de validação e transformação de dados foram testados para garantir consistência entre a lógica dos contratos e a aplicação.

4.3. Resultados

A Tabela 4 apresenta os resultados da cobertura de testes entre os diferentes módulos da *Smart Agro RAF API*. As declarações e linhas alcançaram em grande parte índices superiores a 90%, destacando-se a cobertura total nos módulos relacionados a *middleware*, tipagens e esquemas, o que assegura a confiabilidade das estruturas fundamentais da aplicação. Nos módulos de interação com contratos inteligentes, a cobertura de funções foi de 100% tanto para ERC-721 quanto para ERC-1155, validando a implementação dos requisitos funcionais críticos de rastreabilidade e manipulação de *tokens*. Embora o módulo de rotas apresente menor cobertura de funções (60%), ainda mantém resultados satisfatórios em declarações (80,95%) e linhas (85%), indicando pontos de melhoria, mas sem comprometer a robustez geral. Em síntese, a suíte de testes garante alto grau de confiabilidade e solidez à solução, cobrindo tanto aspectos genéricos quanto específicos da aplicação.

5. Considerações Finais

A agricultura familiar desempenha um papel relevante tanto social quanto economicamente, mas enfrenta desafios na adoção da Instrução Normativa INC 02/2018. Para apoiar esse processo, desenvolvemos uma solução baseada em contratos inteligentes em *blockchain*. Neste trabalho, apresentamos uma API aberta projetada para facilitar o desenvolvimento de sistemas de rastreabilidade que utilizam contratos inteligentes. A solução será

Tabela 4. Cobertura dos Testes.

Arquivos	Declarações		Fluxos		Funções		Linhas	
router	80.95%	17/21	100%	0/0	60%	3/5	85%	17/20
blockchain/config	92.3%	12/13	50%	1/2	100%	0/0	92.3%	12/13
blockchain/interact	94.23%	98/104	68.18%	15/22	100%	26/26	94.11%	96/102
blockchain/interact/erc1155	97.36%	37/38	75%	3/4	100%	11/11	97.36%	37/38
blockchain/interact/erc721	100%	13/13	100%	0/0	100%	3/3	100%	13/13
router/middleware	100%	15/15	100%	4/4	100%	5/5	100%	15/15
types	100%	4/4	100%	0/0	100%	2/2	100%	4/4
types/schemas	100%	21/21	100%	0/0	100%	2/2	100%	21/21

aplicada no projeto Proraf³, e espera-se que outras iniciativas de rastreabilidade também possam adotar a proposta. A avaliação inicial do *Smart Agro RAF API* baseou-se em testes unitários, que asseguraram a corretude funcional e a robustez das camadas lógicas e dos contratos inteligentes. Embora eficazes para validar regras de negócio e fluxos de entrada e saída de forma isolada, tais testes não permitem avaliar a viabilidade operacional da solução, restringindo a análise de seu desempenho em cenários reais.

Como trabalhos futuros, propõe-se a realização de experimentos para avaliar a arquitetura do sistema, incluindo medições de latência e *throughput* sob diferentes volumes de transações, requisitos de resposta rápida e restrições de infraestrutura. Pretende-se comparar as implementações baseadas nos padrões ERC-721 e ERC-1155, testando cargas transacionais simultâneas que simulem cenários típicos de registro de colheitas e movimentação de lotes. Serão analisadas diferentes topologias de rede, centralizadas e distribuídas com múltiplos nós, a fim de produzir dados que orientem a evolução e a aplicação prática da solução. Além disso, está prevista uma avaliação em campo para examinar aspectos de usabilidade, barreiras de adoção e o uso de carteiras digitais na assinatura de transações.

Agradecimentos

Este trabalho recebeu apoio da Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS), por meio dos editais 02/2022, 08/2023, 09/2023 e dos termos de outorga 22/2551-0000841-0, 24/2551-0001368-7 e 24/2551-0000726-1, da CAPES (Código de Financiamento 001) e da Rede Nacional de Ensino e Pesquisa no âmbito do projeto Ilíada. Ainda, Roben recebeu apoio do IFRS e do EDITAL FAPERGS 14/2022.

Referências

- Al-Jaroodi, J. and Mohamed, N. (2019). Blockchain in industries: A survey. *IEEE access*, 7:36500–36515.
- BRASIL (2009). Lei nº 11.947, de 16 de junho de 2009. https://www.planalto.gov.br/ccivil_03/_ato2007-2010/2009/lei/111947.htm. Acesso: 25 ago. 2025.
- BRASIL (2018). Instrução normativa conjunta - inc nº 2, de 7 de fevereiro de 2018. https://www.in.gov.br/materia/-/asset_publisher/

³<https://proraf.com.br>

Kujrw0TZC2Mb/content/id/2915263/doi-10.12915/2018-02-08-instrucao-normativa-conjunta-inc-n-2-de-7-de-fevereiro-de-2018-2915259. Acesso em: 25 ago. 2025.

BRASIL (2023). Governo repassa R\$ 2,5 bilhões em seis meses para alimentação escolar. <https://www.gov.br/secom/pt-br/assuntos/noticias/2023/07/governo-repassa-r-2-5-bilhoes-em-seis-meses-para-alimentacao-escolar>. Acesso em: 25 ago. 2025.

Chervinski, J. O., Melchior, F., Fernandes, R., Sá, G., Antunes, L., Kreutz, D., and Mansilha, R. (2019). Introdução aos blockchains: Teoria e prática. Minicursos da XVII Escola Regional de Redes de Computadores, pages 1–21.

Dedeoglu, V., Dorri, A., Jurdak, R., Michelin, R. A., Lunardi, R. C., Kanhere, S. S., and Zorzo, A. F. (2020). A journey in applying blockchain for cyberphysical systems. In 2020 International Conference on COMMunication Systems & NETWORKS (COMSNETS), pages 383–390. IEEE.

Doan, T. T. H. and Thiemann, P. (2021). A typed programmatic interface to contracts on the blockchain. In Asian Symposium on Programming Languages and Systems, pages 222–240. Springer.

IBGE (2017). Censo agropecuário 2017. <https://www.ibge.gov.br/estatisticas/economicas/agricultura-e-pecuaria/21814-2017-censo-agropecuario.html>. Acesso em: 25 ago. 2025.

Lowder, S. K., Sánchez, M. V., and Bertini, R. (2021). Which farms feed the world and has farmland become more concentrated? World Development, 142:105455.

Lunardi, R. C., Michelin, R. A., Alharby, M., Dedeoglu, V., Nunes, H. C., Arruda, E., Zorzo, A. F., and van Moorsel, A. (2024). When blockchain meets smart cities: Opportunities, security and future research. Blockchains: A Handbook on Fundamentals, Platforms and Applications, 105:423.

Park, J., Jeong, S., and Yeom, K. (2023). Smart contract broker: Improving smart contract reusability in a blockchain environment. Sensors, 23(13):6149.

Pasdar, A., Lee, Y. C., and Dong, Z. (2023). Connect api with blockchain: A survey on blockchain oracle implementation. ACM Computing Surveys, 55(10):1–39.

Santos, N., Goulart, A., Kersten, D., Cornelio, J., Basso, F., and Mansilha, R. (2024a). Evangelização tecnológica da agricultura familiar. In Anais Estendidos do XX Simpósio Brasileiro de Sistemas de Informação, pages 291–296, Porto Alegre, RS, Brasil. SBC.

Santos, N., Goulart, A., Kersten, D., Cornelio, J., Basso, F., and Mansilha, R. (2024b). Proraf: Rastreamento da agricultura familiar. In Anais Estendidos do XX Simpósio Brasileiro de Sistemas de Informação, pages 241–244, Porto Alegre, RS, Brasil. SBC.

Silva, F., Neves, B., Fan, H., Lunardi, R., Kreutz, D., and Mansilha, R. (2025). Smart contracts para rastreamento da agricultura familiar. In Anais do VIII Workshop em Blockchain: Teoria, Tecnologias e Aplicações, pages 112–125, Porto Alegre, RS, Brasil. SBC.