INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA CAMPUS GUARABIRA
DIRETORIA DE DESENVOLVIMENTO DE ENSINO
COORDENAÇÃO DO CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

KEVIN CIRNE DE LACERDA CUSTÓDIO
MATHEUS SERAFIM DOS SANTOS

RELATÓRIO TÉCNICO
Projeto Integrador em Sistemas para Internet (PISI): RURALGEST

GUARABIRA - PB
2026

KEVIN CIRNE DE LACERDA CUSTÓDIO
MATHEUS SERAFIM DOS SANTOS

RELATÓRIO TÉCNICO
Projeto Integrador em Sistema para Internet (PISI): RURALGEST

Trabalho de Conclusão de Curso apresentado ao Curso de Tecnologia em Sistemas para Internet, do Instituto Federal da Paraíba – Campus Guarabira, em cumprimento às exigências parciais para a obtenção do título Tecnólogo em Sistemas para Internet.

Orientador(a): Alex Cabral de Britto.

Prof. Ms. Alex Cabral de Britto

Orientador (IFPB)

Prof. Dr. Rodrigo Leone Alves

Examinador (IFPB)

Prof. Ms. Daniel Marques Targino Guimarães

Examinador (IFPB)

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IFPB - GUARABIRA

C987p Custódio, Kevin Cirne de Lacerda
Projeto integrador em sistemas para internet (PISI): RURALGEST / Kevin Cirne de Lacerda Custódio; Matheus Serafim dos Santos.- Guarabira, 2026.
47f.; il.; color.

Trabalho de Conclusão de Curso (Tecnólogo em Sistemas para Internet). – Instituto Federal da Paraíba, Campus Guarabira. 2026.

"Orientação: Prof. Me. Alex Cabral de Britto"

Referências.

1.Sistema para internet 2. Desenvolvimento de software. 3. Gestão Rural. 4. Governo Digital. I. Título.

CDU 004.4(0.067)

Elaborada por Ana Carine da Costa Gonçalves - CRB 000676

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
CAMPUS GUARABIRA

ATA 7/2026 - CCSTSI/DDE/DG/GB/REITORIA/IFPB

ATA DE APRESENTAÇÃO E DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO

CCS de Tecnologia em Sistemas para Internet

Aos 26 de fevereiro de 2026, às 13h30, na Sala 8 do Bloco Acadêmico, reuniram-se os membros da banca avaliadora, Alex Cabral de Britto (Orientador), Daniel Marques Targino Guimarães (Examinador Interno) e Rodrigo Leone Alves (Examinador Interno), para avaliarem a apresentação do *Trabalho de Conclusão do Curso Superior de Tecnologia em Sistemas para Internet* (Relatório Final do Projeto Integrador em Sistemas para Internet - PISI) do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba (IFPB), *campus* Guarabira, desenvolvido pelos alunos **Kevin Cirne de Lacerda Custodio,** matrícula nº **202213810038 e Matheus Serafim dos Santos,** matrícula nº **202313810029**, intitulado "Projeto Integrador em Sistemas para Internet (PISI): RURALGEST ",protocolado para apresentação de acordo com os requisitos expostos no Projeto Pedagógico do Curso de Tecnologia em Sistemas para Internet. Após a apresentação, a banca apresentou, por unanimidade, pareceres a favor da aprovação do trabalho. Desta forma, o Trabalho de Conclusão de Curso foi **aprovado** e definiu-se a **nota final 85 (oitenta e cinco) pontos.**

Nada mais havendo a tratar, às 17h00, encerraram-se os trabalhos, determinando-se a lavratura desta ata, que, após lida e considerada conforme, será assinada pelos presentes. Eu, Alex Cabral de Britto, lavrei a presente ata.

Guarabira/PB, em 09 de março de 2026.

(assinado eletronicamente)
ALEX CABRAL DE BRITTO
Matrícula SIAPE: 2301362

Documento assinado eletronicamente por:

- **Alex Cabral de Britto, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 09/03/2026 11:11:22.
- **Rodrigo Leone Alves, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 09/03/2026 11:32:36.
- **Daniel Marques Targino Guimaraes, PROF ENS BAS TEC TECNOLOGICO-SUBSTITUTO**, em 09/03/2026 13:34:34.
- **Gabriela Guedes de Souza, COORDENADOR(A) DE CURSO - FUC1 - CCSTSI-GB**, em 09/03/2026 20:30:16.
- **Matheus Serafim dos Santos, DISCENTE (20231381002 9) DE TECNOLOGIA EM SISTEMAS PARA INTERNET - CAMPUS GUARABIRA**, em 10/03/2026 08:40:34.
- **Kevin Cirne de Lacerda Custodio, DISCENTE (20221381003 8) DE TECNOLOGIA EM SISTEMAS PARA INTERNET - CAMPUS GUARABIRA**, em 10/03/2026 15:45:29.

Este documento foi emitido pelo SUAP em 09/03/2026. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código 846921
Verificador: 31d2a79d3d
Código de Autenticação:

Dedicatória

A Deus. A meus pais e familiares, por todo apoio e carinho!

Dedico!

AGRADECIMENTOS

Por Kevin Cirne

Agradeço, primeiramente, a Deus e a minha família, que sempre me apoiaram durante toda a trajetória do curso. Agradeço aos amigos que fiz nesta caminhada, pois, com sua companhia, o trajeto tornou-se muito mais leve e divertido. Agradeço também a todos os professores, em especial ao professor Rhavy, por sua extrema paciência, dedicação e constante preocupação com o aprendizado dos alunos e pelas caronas quando eu não tinha ônibus disponível para voltar para casa.

Por Matheus Serafim

Agradeço a minha mãe, pelo apoio, dedicação e incentivo em todos os momentos desta caminhada. A minha namorada, Rayane Martins, pela paciência, carinho e motivação constantes, que foram fundamentais para que eu chegasse até aqui. Aos amigos que estiveram presentes oferecendo companheirismo e palavras de encorajamento. Aos que me deram carona e facilitaram minha rotina, contribuindo de forma prática e generosa para que eu pudesse concluir esta etapa. E, finalmente, a todos que, de alguma forma, participaram desta jornada, deixo aqui minha sincera gratidão.

Epígrafe

"Primeiro, resolva o problema. Depois, escreva o código."
*Autor — **John Johnson***

RESUMO

O presente trabalho apresenta o desenvolvimento do RuralGest, um sistema *web* de gestão agrícola projetado para a Secretaria de Agricultura de Pirpirituba. O projeto foi motivado pela necessidade de superar os desafios da gestão descentralizada e manual, que dificultava o controle de demandas e a prestação de contas dos serviços públicos municipais. O objetivo geral deste trabalho consistiu em desenvolver uma plataforma digital centralizada para aproximar agricultores e gestão pública, facilitando a solicitação, o monitoramento e a execução de serviços rurais. A metodologia aplicada caracterizou-se como uma pesquisa tecnológica de natureza aplicada, adotando uma abordagem de desenvolvimento interativa e incremental. Para a implementação da solução, utilizou-se uma arquitetura baseada em microsserviços e contêineres Docker, empregando a linguagem Python com o *framework* Flask no *back-end* e a biblioteca ReactJS com Chakra UI no *front-end*, garantindo a responsividade e a usabilidade do sistema. A persistência dos dados foi assegurada pelo Sistema Gerenciador de Banco de Dados PostgreSQL. Como resultados, obteve-se uma aplicação funcional que permite o cadastro unificado de produtores e propriedades, o gerenciamento de solicitações em tempo real e a geração de relatórios gerenciais com suporte a assinaturas digitais. Conclui-se que o RuralGest atendeu aos objetivos propostos, promovendo a modernização administrativa da secretaria, conferindo maior transparência aos processos e agilidade no atendimento ao produtor rural.

Palavras-Chave: Gestão Rural; Sistema Web; Governo Eletrônico; ReactJS; Python.

ABSTRACT

This work presents the development of RuralGest, a web-based agricultural management system designed for the Department of Agriculture of Pirpirituba. The project arose from the need to overcome the challenges of manual and decentralized management, which hindered demand control and accountability of municipal public services. The main objective was to create a centralized digital platform to bring farmers closer to public administration, facilitating the request, monitoring, and execution of rural services. The methodology adopted was characterized as applied technological research, following an iterative and incremental development approach. The solution was implemented with an architecture based on microservices and Docker containers, using Python with the Flask framework for the back-end and ReactJS with Chakra UI for the front-end, ensuring system responsiveness and usability. Data persistence was guaranteed by the PostgreSQL Database Management System. As a result, a functional application was obtained, enabling unified registration of producers and properties, real-time management of requests, and generation of managerial reports with support for digital signatures. It is concluded that RuralGest achieved the proposed objectives, promoting administrative modernization of the department, increasing process transparency, and speeding up service delivery to rural producers.

Keywords: Rural Management; Web System; Electronic Government; ReactJS; Python.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURA E SIGLAS (SE HOUVER)

ABNT – Associação Brasileira de Normas Técnicas

API – Application Programming Interface (Interface de Programação de Aplicação) CPF – Cadastro de Pessoas Físicas

CSS – Cascading Style Sheets (Folhas de Estilo em Cascata) DER – Diagrama Entidade-Relacionamento

E-Gov – Electronic Government (Governo Eletrônico) GTA – Guia de Transporte Animal

HTTP – Hypertext Transfer Protocol (Protocolo de Transferência de Hipertexto) IFPB – Instituto Federal da Paraíba

INCRA – Instituto Nacional de Colonização e Reforma Agrária ITE – Imposto Territorial

JSON – JavaScript Object Notation (Notação de Objetos JavaScript) PDF – Portable Document Format (Formato de Documento Portátil)

RBAC – Role-Based Access Control (Controle de Acesso Baseado em Função) REST – Representational State Transfer (Transferência de Estado Representacional) RF – Requisito Funcional

RNF – Requisito Não Funcional

SGBD – Sistema Gerenciador de Banco de Dados

SPA – Single Page Application (Aplicação de Página Única)

SQL – Structured Query Language (Linguagem de Consulta Estruturada) TI – Tecnologia da Informação

TIC – Tecnologias da Informação e Comunicação

UML – Unified Modeling Language (Linguagem de Modelagem Unificada)

SUMÁRIO

1 INTRODUÇÃO

A agricultura firma-se como uma das atividades mais relevantes para o Brasil, posicionando-se como um dos pilares do desenvolvimento econômico nacional. Segundo MATTEI (2014), a agricultura familiar representa a base produtiva da maioria dos estabelecimentos rurais no país, exigindo que a organização dos serviços municipais acompanhe essa relevância para prover o suporte adequado aos produtores. Bem como destaca, REZENDE (2017), a gestão eficiente das atividades agrícolas no âmbito municipal é um desafio que demanda maior transparência e controle dos processos administrativos.

Nesse cenário, a transição de processos manuais para o meio digital surge como uma necessidade estratégica para otimizar o fluxo de trabalho. Com essa automação, as demandas passam a ser processadas com maior agilidade e precisão. Essa modernização é fundamental para garantir que as informações administrativas sejam íntegras, seguras e facilmente acessíveis tanto para os gestores públicos quanto para os cidadãos atendidos.

Todavia, a realidade observada na Secretaria de Agricultura de Pirpirituba apresenta um cenário de fragmentação de dados e ineficiência operacional. Na atualidade, a gestão dos serviços agrícolas sofre problemas consideráveis de controle, visto que a comunicação depende de processos manuais, planilhas separadas e documentos físicos. Essa desorganização gera obstáculos para o produtor, que não possui um canal especializado para solicitar serviços e acompanhar demandas. Já por parte da gestão, a falta de integração dificulta o monitoramento das atividades e a repartição otimizada de recursos, tornando a geração de relatórios um processo lento e suscetível a falhas.

O objetivo central deste trabalho é propor a automação e centralização desses processos administrativos, da Secretaria de Agricultura de Pirpirituba, por meio de uma solução tecnológica dedicada. Como resposta aos problemas, anteriormente citados, apresenta-se o Sistema de Gestão Rural (RuralGest), uma plataforma digital desenvolvida para gerenciar as atividades entre gestores, técnicos e produtores rurais. A implementação desta ferramenta visa substituir a rotina manual por um ambiente digital integrado, promovendo a eficiência e a transparência na administração pública moderna, conforme os preceitos de REZENDE (2017).

1.1 Objetivos

Nesta seção serão citados os objetivos que guiam o trabalho

1.1.1 Objetivo Geral

Desenvolver um sistema web para o gerenciamento de serviços agrícolas no município de Pirpirituba-PB, propondo a organização da demanda de solicitações, o controle administrativo da gestão e o acompanhamento dos status dos serviços solicitados.

1.1.2 Objetivos Específicos

- Centralizar e digitalizar o cadastro de produtores rurais e propriedades, eliminando o uso de documentos físicos dispersos;
- Permitir o registro e o acompanhamento do status das solicitações em tempo real, garantindo transparência ao produtor;
- Disponibilizar um painel administrativo com indicadores de desempenho para auxiliar a gestão na tomada de decisões;
- Automatizar a geração de relatórios com suporte a assinaturas digitais, agilizando a prestação de contas da secretaria;
- Enviar notificações sobre o andamento dos serviços e avisos gerais, melhorando a comunicação entre a prefeitura e o campo;
- Controlar a execução de serviços, garantindo a atualização cadastral contínua e a organização da fila de espera;
- Implementar mecanismos de segurança da informação que assegurem a integridade e o controle de acesso aos dados armazenados.

1.2 Delimitação do Problema de Pesquisa

O presente projeto delimita-se ao desenvolvimento de uma ferramenta web voltada para a Secretaria de Agricultura de Pirpirituba-PB, focando na automação do fluxo de solicitações de serviços rurais e na organização de relatórios administrativos.

Perante a dependência de processos manuais e a dissociação das informações na Secretaria de Agricultura de Pirpirituba, como um sistema web pode otimizar o gerenciamento dos serviços rurais e auxiliar no acompanhamento do andamento das atividades?

1.3 Justificativa

A justificativa deste trabalho fundamenta-se na relevância social da agricultura familiar que, segundo MATTEI (2014), representa 80% dos estabelecimentos rurais no Brasil. Dessa forma garantir que esses produtores tenham acesso a serviços públicos de forma rápida e organizada é essencial para o desenvolvimento socioeconômico da região.

Atualmente, a gestão desses serviços em Pirpirituba enfrenta a lentidão de processos baseados em papeis, o que compromete o controle e a transparência administrativa. De acordo com REZENDE (2017), o uso da tecnologia da informação é um requisito indispensável para que a administração pública contemporânea seja eficiente e entregue resultados satisfatórios ao cidadão.

Do ponto de vista acadêmico, esta pesquisa oferece uma contribuição específica ao demonstrar, na prática, como arquiteturas *web* modernas podem ser adaptadas ao contexto de baixa maturidade digital no setor público rural. O trabalho evidencia que a aplicação de princípios da Engenharia de Software, neste cenário, exige priorizar requisitos de usabilidade e simplicidade para superar barreiras de letramento tecnológico, servindo, portanto, de modelo para futuras implementações de sistemas de Governo Eletrônico (*E-Gov*) em pequenas prefeituras.

2 REFERENCIAL TEÓRICO

A agricultura familiar é o meio de sustento para a maioria das famílias rurais brasileiras, representando cerca de 80% dos estabelecimentos do país, conforme indica Mattei (2014). Contudo, esse setor enfrenta desafios de gestão devido ao uso predominante de métodos manuais e analógicos, que limitam o crescimento e a transparência administrativa. Conforme REZENDE (2017), a modernização das instituições, por meio da tecnologia é uma exigência indispensável para garantir a eficiência pública, permitindo que processos lentos sejam substituídos por fluxos de dados ágeis e organizados.

Nesse cenário, o software deixa de ser apenas uma ferramenta técnica e passa a atuar como um transformador de informações, capaz de converter dados brutos em inteligência para a prefeitura e para o produtor, de acordo com Pressman e Maxim (2021). Para que essa transição digital ocorra de forma segura, o desenvolvimento deve seguir os princípios de confiabilidade e segurança da engenharia de software, garantindo a proteção das informações municipais (SOMMERVILLE, 2019).

2.1 A Agricultura Familiar e o desafio da informação no campo

A produção agrícola familiar, para além de sua expressiva presença no território nacional, consolida-se como uma força econômica vital para o sustento de diversas regiões e para o desenvolvimento dos municípios brasileiros. Conforme aponta PEDROSO (2017), a evolução desse setor no Brasil foi impulsionada por estratégias de crédito e pesquisa que visavam substituir métodos rudimentares, por meios de produção mais eficientes e adaptados às exigências do mercado.

Contudo, esse desenvolvimento não aconteceu de forma semelhante no que se refere ao acesso à informação. Existe uma grande necessidade de evolução na distribuição de tecnologias em áreas mais distantes dos grandes centros tecnológicos. Segundo SCHNEIDER et al. (2016), a agricultura familiar perpassa por diversas dificuldades relacionadas ao mercado e à gestão do negócio, sendo a falta de familiaridade com a tecnologia um dos principais obstáculos para o produtor.

Esse entrave tecnológico é sustentado tanto pela complexidade dos softwares quanto pelo acesso restrito a informações claras e íntegras. Como reforçam SOUZA et al. (2011) e DE PAULA et al. (2014), a dificuldade de uso das ferramentas digitais muitas vezes afasta o homem do campo das políticas de modernização administrativa.

Diante das dificuldades citadas, percebe-se uma lacuna no desenvolvimento de soluções que priorizem a simplicidade e a transparência para o contexto específico das secretarias de agricultura municipais. Dessa maneira o RuralGest busca preencher essa falta, oferecendo uma ferramenta que visa superar a complexidade dos softwares tradicionais e facilitar o acompanhamento das atividades para o produtor rural.

2.2 Engenharia de Software e Qualidade de Sistemas

O desenvolvimento do RuralGest fundamenta-se nos preceitos da Engenharia de Software para assegurar o rigor técnico, a manutenibilidade e a longevidade da ferramenta digital. Segundo PRESSMAN e MAXIM (2021), o software deve ser interpretado como um transformador de informações, apto a processar dados brutos e convertê-los em ativos estratégicos para um domínio específico. No escopo deste projeto, essa transformação materializa-se ao converter solicitações manuais e dispersas de serviços rurais em um fluxo de dados estruturado e organizado para o controle da gestão pública municipal.

Para que o sistema opere como uma solução robusta e profissional, a pesquisa adota os atributos de qualidade propostos por SOMMERVILLE (2019). A confiabilidade assegura que a plataforma execute suas funções de forma consistente, minimizando falhas inesperadas, enquanto a segurança é abordada por meio da implementação de mecanismos que protejam a integridade dos dados e garantam o controle de acesso às informações sensíveis dos produtores e da prefeitura. Assim, a aplicação desses princípios técnicos fundamenta uma arquitetura voltada à transparência na prestação de contas e à integridade dos processos administrativos rurais.

2.3 Sistemas para Internet e a modernização administrativa

A implementação de sistemas baseados na web no setor público fundamenta-se no conceito de Governança Digital, que utiliza as Tecnologias de Informação e Comunicação (TICs) para ampliar a transparência e a eficiência dos serviços prestados ao cidadão. Conforme aponta REZENDE (2017), a modernização administrativa mediada pela tecnologia permite que a gestão pública supere as limitações dos processos analógicos, centralizando o fluxo informativo e facilitando a tomada de decisão baseada em dados reais.

Para além da eficiência operacional interna, a eficácia de uma ferramenta como o

RuralGest está intrinsecamente atrelada à sua usabilidade. Dado que o contexto rural é frequentemente marcado por lacunas na inclusão digital, a fundamentação teórica sobre interface e Experiência do Usuário (UX) torna-se um requisito técnico imperativo. Um software voltado a esse domínio deve priorizar a simplicidade arquitetural e a clareza visual, visando reduzir a carga cognitiva do usuário e garantir que o agricultor acesse os serviços de forma intuitiva, independentemente de sua proficiência tecnológica prévia.

Desta forma, a convergência entre uma engenharia de software rigorosa e uma interface centrada no usuário encerra a base teórica necessária para o desenvolvimento de uma solução que atenda tanto às exigências técnicas da administração pública, quanto às necessidades práticas dos produtores rurais de Pirpirituba.

3 METODOLOGIA

Este capítulo descreve o percurso metodológico adotado para a concepção e o desenvolvimento do sistema RuralGest. A definição adequada da metodologia é crucial, pois, segundo Prodanov e Freitas (2013), o método é o conjunto de processos pelos quais se torna possível conhecer uma determinada realidade e produzir um determinado objeto.

3.1 Classificação da Pesquisa

Quanto à sua natureza, esta pesquisa classifica-se como **aplicada**, pois objetiva gerar conhecimentos para aplicação prática, dirigidos à solução de problemas específicos da Secretaria de Agricultura de Pirpirituba. Segundo Gil (2019), a pesquisa aplicada objetiva gerar conhecimentos para aplicação prática, dirigidos à solução de problemas específicos.

Do ponto de vista dos objetivos, trata-se de uma pesquisa **exploratória e descritiva**. Exploratória, Gil (2019) define que a pesquisa exploratória visa proporcionar maior familiaridade com o problema. Esta etapa concretizou-se através do levantamento de requisitos junto aos gestores para compreender o fluxo atual. Descritiva, Conforme Prodanov e Freitas (2013), a pesquisa descritiva expõe características de determinada população ou fenômeno. Aqui, documentam-se as características do *software* proposto, suas funcionalidades e regras de negócio.

Com relação aos procedimentos técnicos, o trabalho enquadra-se como uma pesquisa tecnológica, caracterizada pelo desenvolvimento de um produto final (o *software* RuralGest) que sistematiza o processo de gestão rural.

3.2 Processo de Desenvolvimento

Para a construção do sistema, optou-se por uma abordagem de desenvolvimento de software iterativa e incremental. De acordo com Pressman e Maxim (2016), o modelo incremental aplica sequências lineares de desenvolvimento à medida que o tempo avança, onde cada incremento entrega uma versão funcional do produto. O ciclo de vida do projeto foi estruturado nas seguintes etapas:

- Levantamento de Requisitos: Identificação das necessidades dos gestores, técnicos e produtores rurais, resultando na lista de Requisitos Funcionais e Não Funcionais, assim como

afirma Sommerville (2019), requisitos mal definidos são a principal causa de falhas em projetos de *software*.;

- Análise e Design: Modelagem do banco de dados (Modelo Entidade-Relacionamento) e estruturação das interfaces do usuário, focando na usabilidade;
- Implementação: Codificação do *front-end* e *back-end* utilizando as tecnologias selecionadas;
- Testes e Validação: Verificação das funcionalidades para garantir a integridade dos dados e a correta execução dos serviços.

3.3 Procedimentos Técnicos

Para a operacionalização do desenvolvimento, o ambiente de trabalho foi configurado utilizando a plataforma de conteinerização Docker. Esta escolha permitiu a padronização do ambiente de desenvolvimento, garantindo que as versões das linguagens e bancos de dados fossem as mesmas em todas as etapas do projeto, eliminando incompatibilidades entre sistemas operacionais distintos.

A arquitetura da aplicação foi dividida em duas camadas principais (*Back-end* e *Front-end*), comunicando-se via protocolo HTTP.

- No servidor (Back-end): Foi desenvolvida uma API RESTFUL (*Application Programming Interface*) utilizando a linguagem Python 3.9 com o *microframework* Flask. A aplicação implementa padrões de segurança como autenticação via Token e criptografia de senhas com a biblioteca Bcrypt. Além disso, integra-se a um sistema de cache distribuído (Redis) para otimização de consultas onerosas, com os indicadores da dashboard administrativa.
- Integração Contínua (DevOps): Foi implementada uma esteira de automação (CI/CD) utilizando o servidor Jenkins. A cada alteração no código, o Jenkins aciona automaticamente agentes Docker efêmeros que executam testes unitários e validações de qualidade, assegurando a estabilidade do sistema antes da disponibilização.
- No cliente (Front-end): A interface do usuário foi construída com a biblioteca ReactJS, utilizando CSS para a estilização. O uso do React permitiu a criação de uma *Single Page Application* (SPA), proporcionando uma navegação fluida e responsiva, requisito essencial para o uso em dispositivos móveis no campo.
- Persistência de Dados: Para o armazenamento das informações, utilizou-se o Sistema Gerenciador de Banco de Dados (SGBD) Relacional PostgreSQL, escolhido por sua robustez e conformidade com os padrões ACID, essenciais para garantir a integridade dos cadastros e das

solicitações de serviço.

3.4 REQUISITOS

A Engenharia de Requisitos é uma etapa fundamental no desenvolvimento de sistemas. Segundo Sommerville (2019), é o processo de estabelecer os serviços que o cliente requer do sistema e as restrições sob as quais ele opera e é desenvolvido. Neste projeto, o levantamento permitiu identificar as necessidades do RuralGest, classificadas em: Requisitos Funcionais e Não Funcionais.

3.4.1 Requisitos Funcionais (RF)

Os requisitos funcionais definem o comportamento do sistema. De acordo com Pressman e Maxim (2016), eles descrevem as funções que o software deve executar, as entradas que deve aceitar e as saídas que deve produzir. Em suma, representam o que o sistema fará para atender às necessidades do usuário.
Abaixo, listam-se os requisitos funcionais do RuralGest:

Tabela 1 – Requisitos Funcionais do Sistema

ID	Requisito Funcional	Descrição	Prioridade
RF01	Cadastro de Agricultores	Permitir o cadastro de agricultores contendo nome, CPF, comunidade, dados da propriedade, independente de registro no INCRA.	Alta
RF02	Edição de Agricultores	Editar dados cadastrais de agricultores já registrados no sistema, mantendo a integridade dos registros vinculados.	Alta
RF03	Exclusão de Agricultores	Excluir cadastros de agricultores, aplicando regras de verificação para não deixar registros órfãos (solicitações sem dono).	Alta

RF04	Validação Documental	Funcionalidade exclusiva para o Gestor validar documentação física (CCIR, ITR), liberando acesso do agricultor aos serviços.	Alta
RF05	Cadastro de Funcionários	Permitir o cadastro de funcionários (operadores, técnicos, atendentes) com definição de cargo e perfil de acesso	Alta
RF06	Edição de Funcionários	Permitir a alteração de dados de funcionários, como mudança de cargo ou atualização de contato.	Alta
RF07	Exclusão de Funcionários	Permitir a desativação ou exclusão de funcionários, revogando imediatamente seu acesso ao sistema.	Alta
RF08	Cadastro de Propriedades	Registrar propriedades rurais detalhando área total, área explorável (hectares), coordenadas geográficas e cultura principal.	Alta
RF09	Edição de Propriedades	Alterar dados da propriedade para corrigir áreas ou atualizar culturas, refletindo a realidade produtiva atual.	Média
RF10	Exclusão de Propriedades	Excluir propriedades, bloqueando a ação caso existam solicitações de serviço (pendentes ou concluídas) vinculadas ao imóvel.	Alta
RF11	Cadastro de Veículos	Registrar máquinas e veículos da frota (tratores, caminhões), informando modelo, placa e capacidade operacional.	Alta
RF12	Edição de veículos	Editar dados da frota, permitindo atualizações de quilometragem, horímetro ou correções de cadastro.	Alta
RF13	Exclusão de veículos	Excluir veículos da frota, mantendo histórico de quais serviços foram executados por eles antes da exclusão.	Alta

RF14	Gestão de Status do Veículo	Visualizar e alterar o status operacional do veículo em tempo real: Disponível, Em uso ou Em Manutenção.	Alta
RF15	Cadastro de Serviços	Registrar novos tipos de serviços no catálogo (ex: Corte de Terra, GTA), definindo limites de atendimento por hectare.	Média
RF16	Edição de Serviços	Alterar parâmetros dos serviços oferecidos, como descrição, requisitos ou maquinário necessário.	Média
RF17	Exclusão de Serviços	Remover serviços do catálogo tornando-os indisponíveis para novas solicitações	Média
RF18	Solicitação de Serviços	Permitir que o agricultor solicite serviços via interface própria, selecionando obrigatoriamente uma propriedade validada e inserindo observações.	Alta
RF19	Histórico de Solicitações	Exibir ao agricultor uma lista de seus pedidos anteriores e atuais com status detalhado e datas de execução.	Média
RF20	Regra de Anti-Duplicidade	Bloquear automaticamente uma nova solicitação caso já exista um pedido ‘Pendente’ ou ‘Em andamento’ para o mesmo serviço na mesma propriedade.	Alta
RF21	Atribuição de Operador/Veículo	Permitir que o Gestor atribua um funcionário responsável e um veículo específico para executar uma solicitação em aberto.	Alta
RF22	Visualização de Demanda	Exibir ao operador/técnico uma lista filtrada contendo apenas as solicitações atribuídas a ele (‘Meus Serviços’).	Alta
RF23	Registro de Execução	Permitir que o operador ou gestor altere o status para ‘Concluída’ registrando data/hora (Timestamp) automática e manual.	Alta

RF24	Geração de Documentos PDF	Gerar automaticamente o ‘Protocolo de Solicitação’ e o ‘Relatório de Conclusão’ com código de segurança para assinatura.	Alta
RF25	Notificações Automáticas	Enviar alertas ao agricultor sempre que o status de sua solicitação mudar (ex: Aprovada, Recusada, Concluída).	Média
RF26	Dashboard Gerencial	Painel visual com indicadores: total de hectares atendidos, serviços mais demandados e status da frota	Média
RF27	Auditoria (Logs)	Registrar trilha de auditoria de ações críticas (Quem cadastrou, quem excluiu, quem validou) para segurança.	Alta

3.4.2 Requisitos Não Funcionais (RNF)

Enquanto os funcionais tratam do "o que" o sistema faz, os requisitos não funcionais referem-se ao "como" ele deve se comportar. Conforme explica Sommerville (2019), são restrições impostas aos serviços ou funções oferecidas pelo sistema, como restrições de tempo, restrições sobre o processo de desenvolvimento e padrões. Eles definem atributos de qualidade como confiabilidade, tempo de resposta e requisitos de armazenamento.
Os requisitos não funcionais definidos para o projeto são:

Tabela 2 – Requisitos Não Funcionais (RNF)

ID	Requisito Não Funcional	Descrição
RNF01	Usabilidade	Interface responsiva e simplificada, minimizando cliques para atender usuários com diferentes níveis de letramento digital.
RNF02	Segurança (RBAC)	Controle de acesso baseado em perfis (Gestor, Agricultor, Operador) e criptografia de senhas (Bcrypt).

RNF03	Integridade de Dados	Uso de chaves estrangeiras (Foreign Keys) no banco para impedir registros órfãos e garantir consistência.
RNF04	Desempenho e Caching	Utilização de cache (Redis) para otimizar consultas pesadas (dashboards) e garantir baixo tempo de resposta.
RNF05	Disponibilidade	Arquitetura em contêineres (Docker) para facilitar implantação e garantir disponibilidade em servidores Linux/Debian.
RNF06	Auditoria Temporal	Registro automático de timestamps (creates_at, updated_at) em todas as tabelas críticas do sistema.
RNF07	Automação (CI/CD)	Esteira de integração contínua (Jenkins) para testes automatizados e validação de código antes do deplo

3.5 Perfis de Usuário

- **Administrador (Gestor):** Acesso total, visualização de *dashboards* (RF15), relatórios gerenciais e auditoria (RF18).
- **Técnico/Operador:** Registro de visitas (RF17), execução de serviços e atualização de *status*.
- **Produtor Rural:** Solicitação de serviços, acompanhamento de histórico e recebimento de notificações (RF14).

3.6 User Stories

Abaixo, algumas histórias de usuário representativas para o desenvolvimento:

- **US01:** "Como Gestor, quero visualizar uma dashboard com gráficos de serviços solicitados e status da frota, para que eu possa planejar a manutenção e o atendimento. "
- **US02:** "Como Assistente Administrativo, quero auxiliar o gestor cadastrando agricultores e lançando/validando solicitações no sistema, para que agilizar o atendimento na secretária."
- **US03:** "Como Gestor, quero gerenciar os cadastros administrativos (Funcionários, Veículos e Tipos de Serviço), adicionando, editando ou removendo registros, para manter o sistema atualizado com os recursos disponíveis."

- **US04:** "Como Produtor Rural, quero receber uma notificação quando minha solicitação mudar de status, para que eu possa me preparar para receber o serviço.”
- **US05:** "Como Gestor, quero visualizar a demanda de serviços pendentes e o status operacional de cada veículo (Disponível/ Em Manutenção), para que eu possa decidir qual trator enviar para cada propriedade."
- **US06:** "Como Sistema, quero bloquear automaticamente novos pedidos se já houver um pendente para a mesma propriedade, para evitar duplicidade e desperdício de recursos."
- **US07:** "Como Gestor, quero atribuir um operador e um veículo específico para uma solicitação, oficializando quem será o responsável pela execução."
- **US08:** "Como Produtor Rural, quero visualizar o histórico das minhas solicitações, para que eu possa acompanhar o que já foi pedido e o que ainda está pendente."

4 RESULTADOS E DISCUSSÃO

Este capítulo apresenta os resultados obtidos com o desenvolvimento do sistema **RuralGest**, detalhando a arquitetura de *software* implementada, as ferramentas tecnológicas utilizadas e os artefatos de modelagem que guiaram a construção da solução.

4.1 Sobre o Projeto

O **RuralGest** consolidou-se como uma plataforma *web* de gestão agrícola, projetada para atender às demandas específicas da Secretaria de Agricultura de Pirpirituba. O resultado final é um sistema que integra, em um único ambiente digital, o fluxo de solicitação de serviços, o cadastro de produtores rurais e o controle operacional de máquinas e técnicos.

A solução substitui os controles manuais e descentralizados por um processo informatizado e auditável. Para o gestor público, o sistema entrega transparência e capacidade de planejamento através de painéis de indicadores. Já para o produtor rural, oferece acessibilidade e agilidade na solicitação de demandas. O *software* foi desenvolvido com foco na usabilidade, garantindo que a complexidade tecnológica permaneça transparente para o usuário final.

4.2 Arquitetura do Sistema

A arquitetura do sistema foi estruturada seguindo o modelo cliente-servidor, desacoplando a interface do usuário (*Front-end*) da lógica de negócios (*Back-end*). Segundo Tanenbaum e Wetherall (2011), neste modelo, o dispositivo do usuário (cliente) solicita serviços e o servidor processa e retorna as informações, permitindo uma separação clara de responsabilidades. Esta separação de responsabilidades facilita a manutenção evolutiva e permite que diferentes equipes ou desenvolvedores trabalhem simultaneamente em camadas distintas da aplicação.

A comunicação entre as camadas ocorre através de uma API (*Application Programming Interface*), baseada no estilo arquitetural REST (*Representational State Transfer*). O cliente (*Front-end*) realiza requisições HTTP enviando e recebendo dados no formato JSON (*JavaScript Object Notation*), que são processados pelo servidor (*Back-end*). Todo o ambiente é orquestrado através de contêineres, assegurando a portabilidade da aplicação entre diferentes

infraestruturas de hospedagem. Todo esse ecossistema é orquestrado via Docker Compose, onde cada componente (API, Banco Cache) reside em seu próprio contêiner isolado, comunicando-se através de uma rede virtual interna segura.

4.3 Ferramentas e Tecnologias Utilizadas

Para a implementação da arquitetura proposta, foram selecionadas ferramentas de mercado consolidadas e de código aberto (*open source*), visando garantir a robustez, a segurança e o baixo custo de licenciamento do projeto.

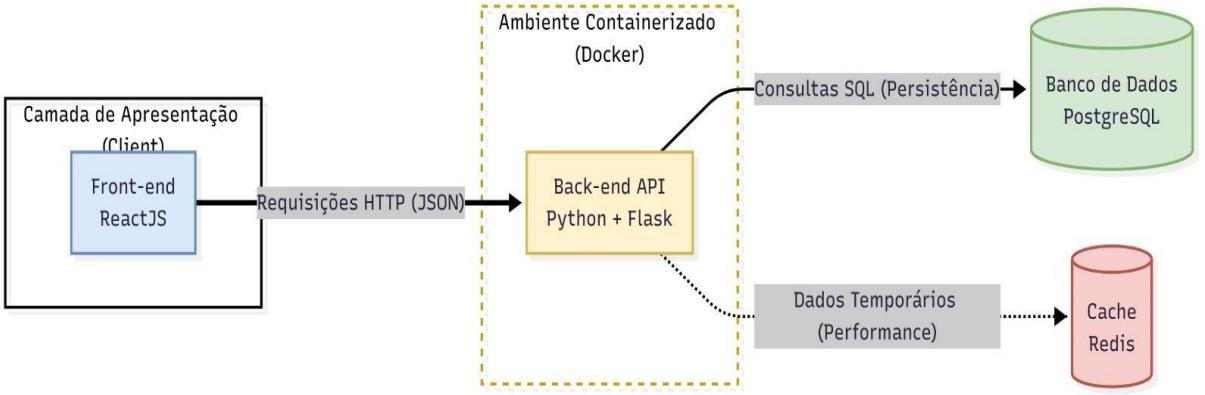
- **Linguagem de Back-end (Python):** Utilizou-se a linguagem Python devido à sua alta legibilidade e vasta biblioteca padrão. Sua sintaxe clara acelera o desenvolvimento e facilita a implementação de regras de negócio complexas.
- **Framework Web (Flask):** O Flask foi adotado como *framework* para o desenvolvimento da API. Por ser um *microframework*, ele oferece a flexibilidade necessária para estruturar a aplicação sem impor dependências excessivas, permitindo um controle granular sobre as rotas e o tratamento das requisições.
- **Linguagem de Front-end (ReactJS):** A interface foi construída com a biblioteca ReactJS. Baseada em componentes reutilizáveis e no conceito de *Virtual DOM*, essa tecnologia permite a criação de interfaces dinâmicas e de alta performance, essenciais para a experiência do usuário em dispositivos móveis.
- **Biblioteca de Componentes (Chakra UI):** Para a estilização e *design* da interface, utilizou-se o Chakra UI. Trata-se de uma biblioteca de componentes simples, modulares e acessíveis para React, que acelera o desenvolvimento ao fornecer elementos de UI prontos e customizáveis, garantindo conformidade com padrões de acessibilidade WAI-ARIA (SEGUN, 2024).
- **Sistema Gerenciador de Banco de Dados (PostgreSQL):** Para a persistência dos dados, optou-se pelo PostgreSQL. Trata-se de um banco de dados relacional objeto-relacional robusto, que oferece suporte avançado a transações, integridade referencial e consultas complexas, garantindo a segurança das informações cadastrais e operacionais.
- **Ambiente de Desenvolvimento (Docker):** A ferramenta Docker foi utilizada para a criação de ambientes isolados (contêineres). Isso assegura que o *software* execute de maneira idêntica nas máquinas de desenvolvimento e no servidor de produção, eliminando falhas decorrentes de diferenças de configuração de ambiente.

- **Sistema de Cache (Redis):** Foi implementado o Redis como banco de dados em memória (in-memory data structure store) para atuar como camada de cache. Sua utilização visa otimizar o desempenho do sistema, armazenando resultados de consultas onerosas (como indicadores da dashboard) para reduzir a latência e a carga sobre o banco de dados principal.
- **Integração Contínua (Jenkins):** Para automatizar o ciclo de vida do desenvolvimento, utilizou-se o servidor de automação Jeins. Configurado em contêiner, ele monitora o repositório de código e executa pipelines de teste e validação a cada nova alteração, garantindo a qualidade do código entregue.

4.4 Desenho da Arquitetura

A Figura 1 ilustra a arquitetura de alto nível da solução **RuralGest**, demonstrando o fluxo de dados entre as camadas da aplicação. O modelo adota uma abordagem de contêineres para garantir o isolamento e a reprodutibilidade do ambiente.

Figura 1 – Desenho da Arquitetura



Fonte: Autoria própria (2026).

4.5 Diagramas de Casos de Uso

Para modelar as interações funcionais do sistema, utilizou-se a linguagem de modelagem UML (*Unified Modeling Language*). Segundo Guedes (2018), a UML é a linguagem padrão para a elaboração da estrutura de projetos de *software*, permitindo visualizar, especificar e documentar sistemas. O Diagrama de Casos de Uso (Figura 2) apresenta os atores envolvidos e as principais funcionalidades do RuralGest. Bezerra (2014) explica que o diagrama de casos de uso descreve a funcionalidade proposta para o novo sistema, focando no "o que" o sistema faz, e não em "como" ele faz.

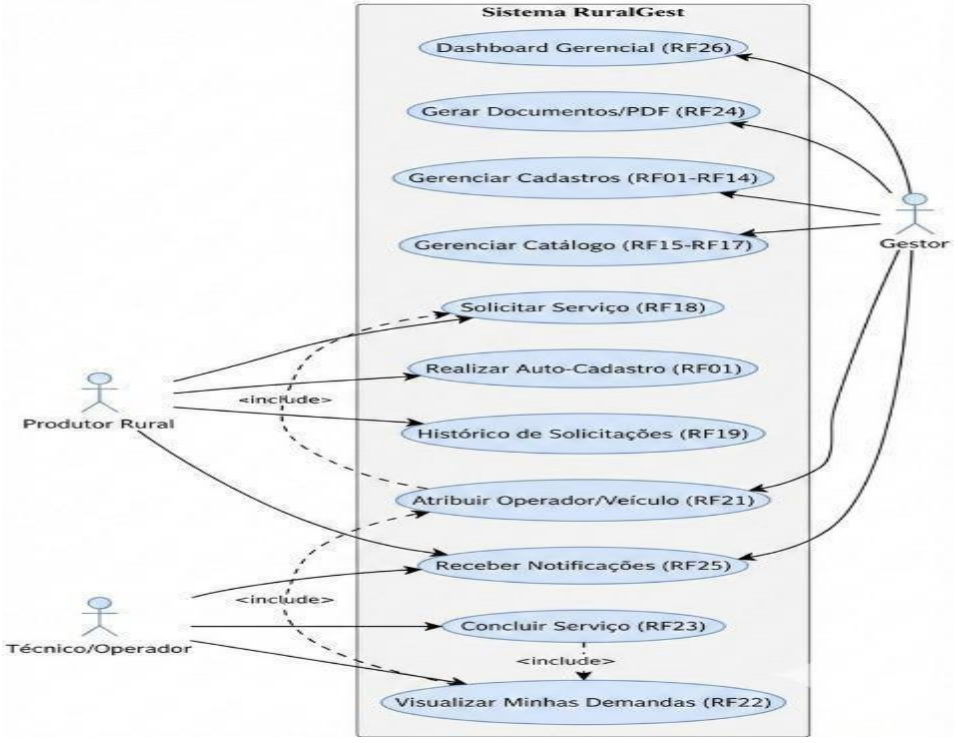
4.5.1 Visão Geral

Identificou-se a atuação de três atores principais no sistema:

- **Gestor (Secretário):** Ator estratégico com acesso total ao sistema. É responsável pela visualização de indicadores de desempenho (dashboards), gerenciamento de usuários do sistema e tomada de decisão baseada nos relatórios gerenciais.
- **Assistente Administrativo:** Ator operacional responsável pelo fluxo de entrada de dados. Suas funções incluem o cadastro e validação documental dos agricultores (conferência de CCIR/ITR, lançamento de solicitações presenciais e gestão da fila de atendimento.
- **Produtor Rural:** Usuário beneficiário do sistema. Interage com a plataforma para realizar seu autocadastro, solicitar serviços de forma remota, acompanhar o status dos pedidos em tempo real e receber notificações.
- **Operador/Técnico:** Ator responsável pela execução. Acessa o sistema para visualizar a lista de serviços atribuídos a ele ("Minhas Tarefas"), registrar a execução das atividades no campo e atualizar o status das solicitações para "Concluída".

Na figura 2 é apresentado o Diagrama de Casos de Uso do Rural Gest

Figura 2 – Diagrama de Casos de Uso do RuralGest



Fonte: Autoria própria (2026)

4.5.2 Alto Nível com Includes

Conforme observado na Figura 2, o sistema estrutura-se em torno de casos de uso centrais que possuem relacionamentos de inclusão (<<*include*>>), indicando dependências obrigatórias para a execução de tarefas complexas:

1. **Controle de Execução:** O caso de uso "Controlar Execução de Serviços (RF08)" atua como um hub central. Para ser executado, ele depende das informações provenientes da "Solicitação de Serviço" e integra-se diretamente ao "Gerenciamento de Cadastros (RF01-RF05)".
2. **Painel Administrativo:** A funcionalidade "Visualizar Painel Administrativo (RF15)" consome dados processados pelo controle de execução, permitindo ao gestor uma visão macro das operações.
3. **Funcionalidades Específicas:** O diagrama também destaca funções críticas como "Assinar Documentos Digitalmente (RF11)" pelo gestor e "Registrar Visita Técnica (RF17)" pelo agrônomo, atendendo aos requisitos de formalização e auditoria do sistema.

4.6 Modelo Lógico do Banco de Dados

A modelagem de dados foi realizada seguindo o paradigma relacional, visando garantir a integridade referencial e a consistência das informações armazenadas. O esquema do banco de dados foi normalizado até a terceira forma normal (3FN) para evitar redundâncias e anomalias de atualização.

As principais entidades que compõem o modelo de dados do RuralGest são:

- Usuário: Entidade central de segurança. Armazena as credenciais de acesso (Login e Senha criptografada com Bcrypt) e define o perfil de permissão (Role), controlando o acesso às funcionalidades do sistema.
- Agricultor: Contém os dados pessoais e cadastrais dos produtores rurais. Relaciona-se com a entidade Usuário (para acesso ao sistema) e inclui campos de controle administrativo, como documentação_validada (booleano) e data_atualizacao_cadastro para gestão da vigência do cadastro.
- Propriedade: Representa os imóveis rurais. Armazena dados técnicos essenciais para a alocação de máquinas, como a área_exploravel e coordenadas geográficas, vinculando-se obrigatoriamente a um Agricultor.

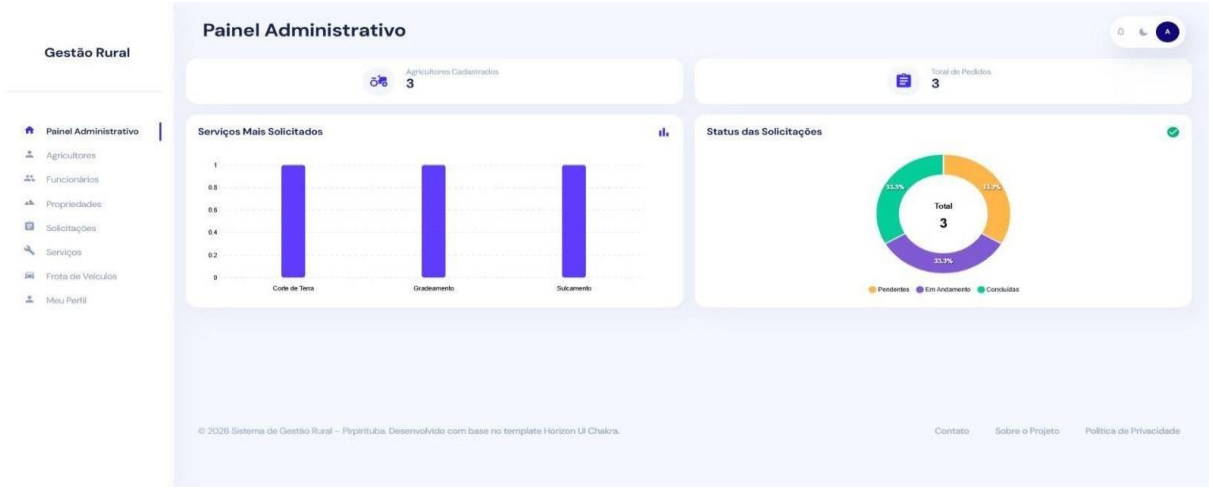
- Serviço: Catálogo de serviços oferecidos pela secretaria (ex: Corte de Terra, Ensilagem). Define regras de negócio como a capacidade_hectares e o tipo de veículo necessário para a execução.
- Veículo: Gerencia a frota municipal. Registra tratores e máquinas, controlando seu status operacional (Disponível, Em manutenção, Em Uso) para evitar conflitos de agendamento.
- Solicitação: Entidade associativa central que registra o pedido de serviço. Vincula um Agricultor, uma Propriedade, um Serviço, um Operador responsável e um Veículo. Armazena o clico de vida do pedido(status), datas de solicitação e execução, e observações.
- Documento: Armazena os metadados dos arquivos gerados pelo sistema (PDFs de Protocolo e Relatórios), incluindo o hash_seguranca para validação de autenticidade física.
- Notificação: Registra os alertas gerados pelo sistema para os usuários, permitindo o controle de leitura e o histórico de comunicação entre a gestão e o produtor.
- Auditoria: Entidade de segurança que registra uma trilha de logs, armazenando quem realizou ações críticas (Criação, Edição, Exclusão), quando ocorreu e qual registro foi afetado, garantindo a rastreabilidade das operações.

A persistência física foi implementada no SGBD PostgreSQL, utilizando chaves estrangeiras (*Foreign Keys*) e triggers automáticos para assegurar a consistência dos dados.

4.7 Sobre o Sistema

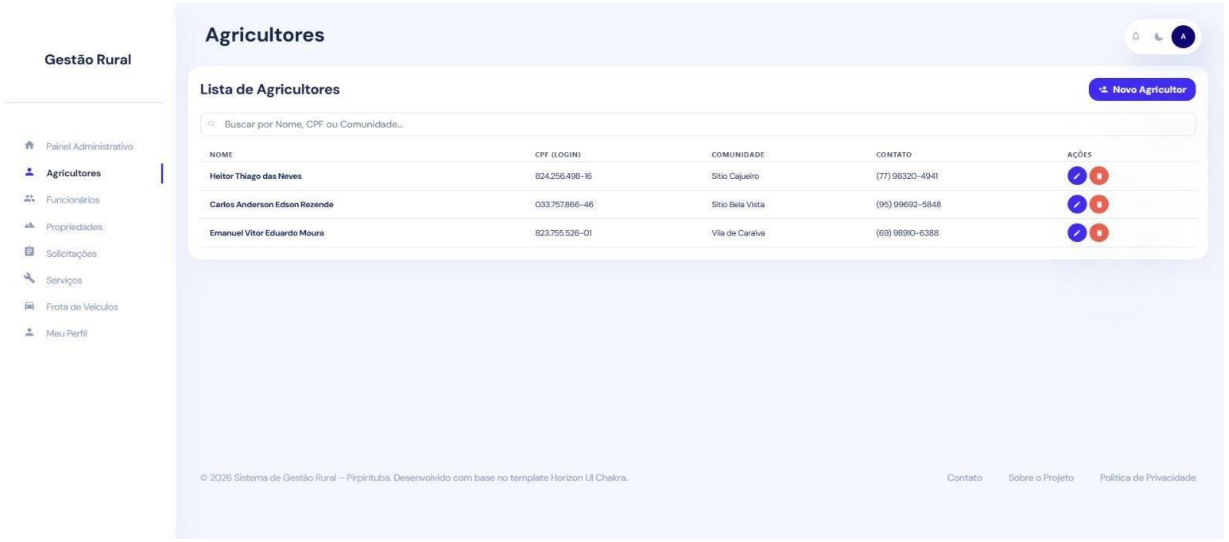
A Figura 1 representa o painel administrativo do gestor. Nela, o gestor tem uma visão geral sobre o sistema, observando a quantidade de agricultores cadastrados, total de pedidos, um gráfico sobre os serviços mais solicitados e um gráfico sobre os status das solicitações.

Figura 1 – Painel administrativo: Sobre o Sistema



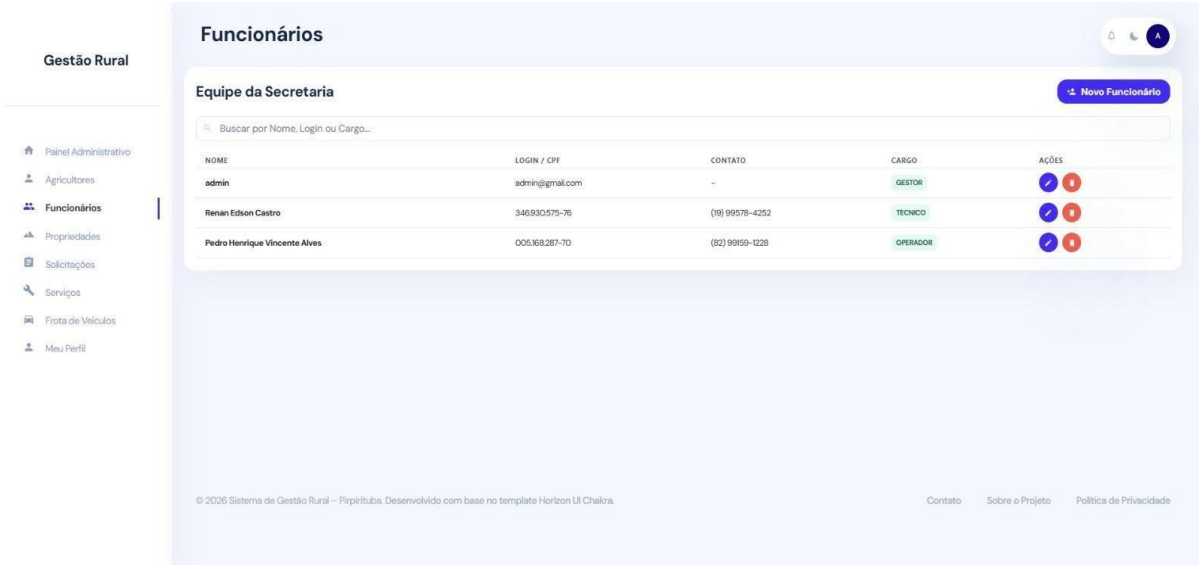
A figura 2 representa a Tela de Listagem de Agricultores. Os cadastros dos agricultores são feitos preenchendo os campos com dados como: Nome, Cpf, Comunidade e contato

Figura 2 – Tela de Listagem de Agricultores: Sobre o Sistema



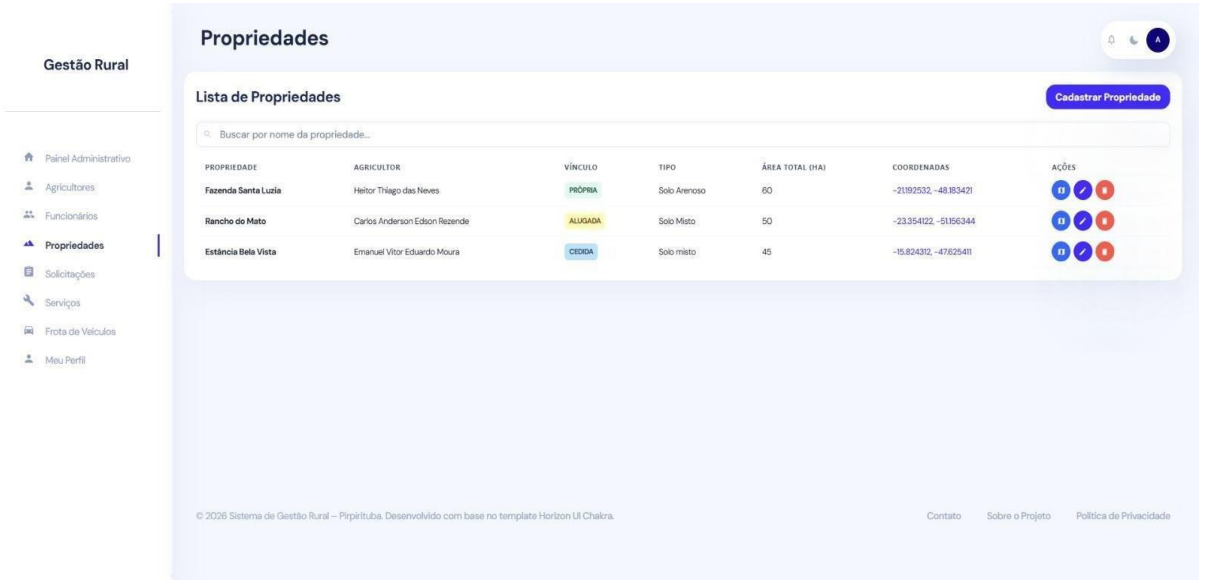
A figura 3 representa a Tela de Cadastro dos Funcionários. Nela os funcionários são cadastrados preenchendo os campos de Cpf, Contato e cargo corresponde na secretária.

Figura 3 –Tela de Cadastro de Funcionários: Sobre o Sistema



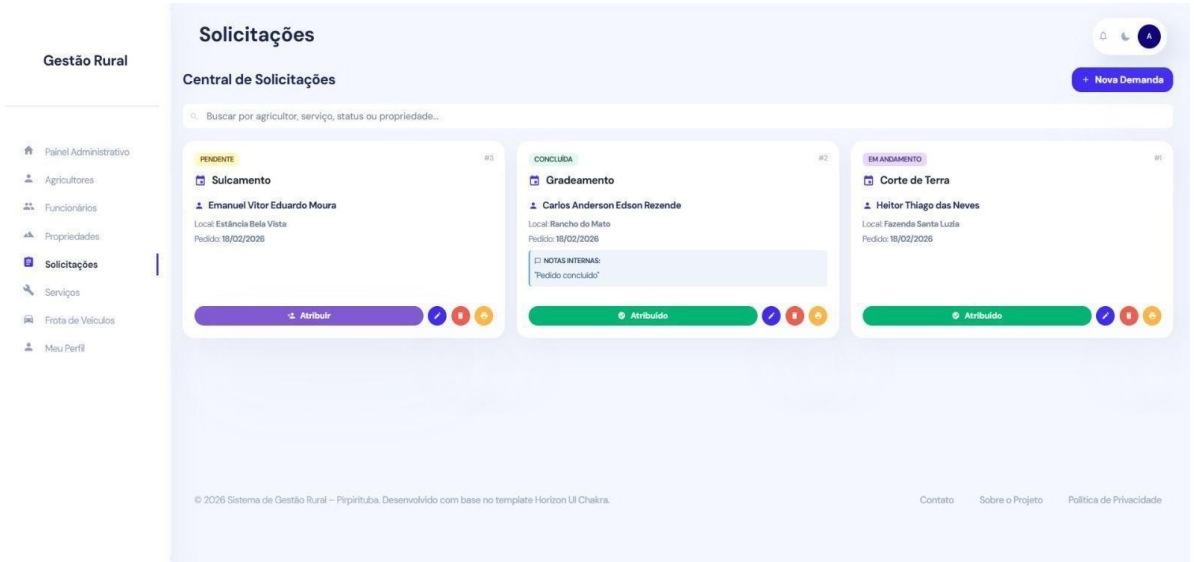
A figura 4 representa a Tela de Cadastro de Propriedades. As propriedades são cadastradas preenchendo os campos do Agricultor responsável pela propriedade, Tipo de Vínculo, Tipo do solo, Área Total e Coordenadas.

Figura 4 –Tela de Cadastro de Propriedades: Sobre o Sistema



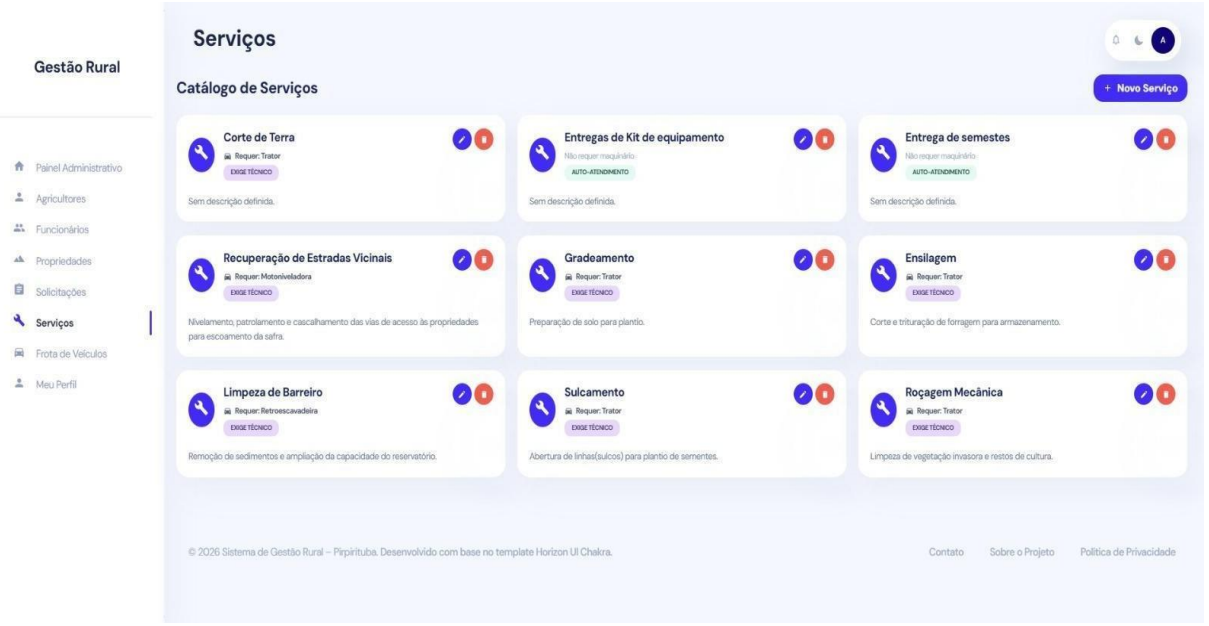
A figura 5 representa a Tela de Solicitação de Serviços. Os serviços solicitados pelo agricultor são recebidos e listados na tela, contendo informações do tipo de serviço solicitado, o nome do agricultor que está solicitando o serviço, o local onde o serviço será realizado, a data que o pedido foi solicitado e o status do serviço.

Figura 5 –Tela de Solicitação de Serviços: Sobre o Sistema



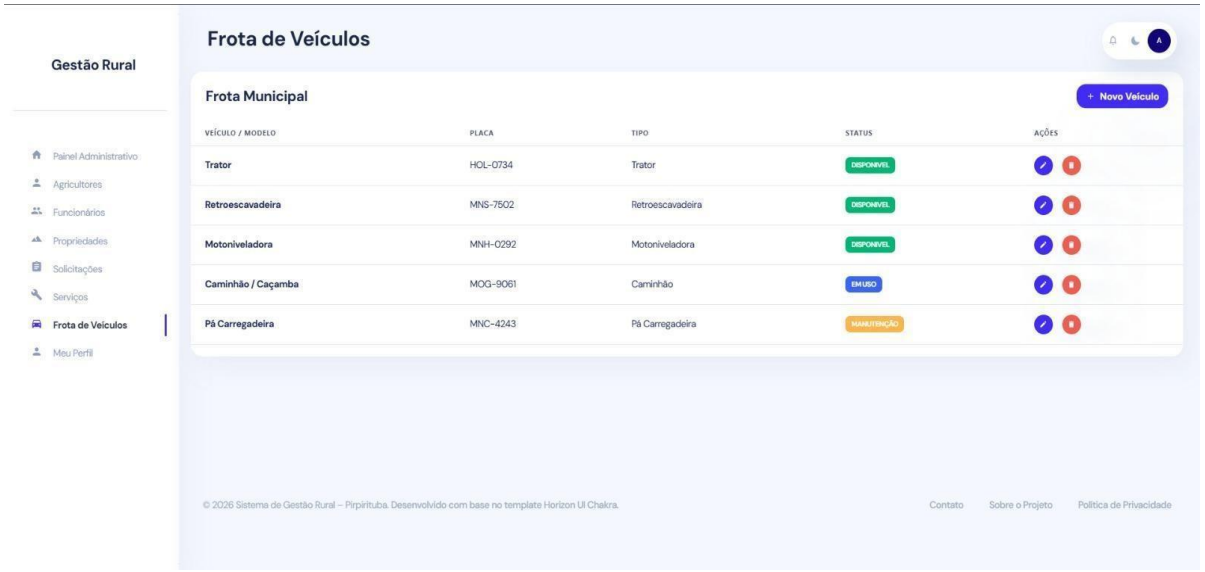
A figura 6 representa a Tela de Listagem de Serviços. Nela os serviços são listando contendo informações do tipo de serviço, qual tipo de veículo esse serviço necessita para ser feito, se precisa de algum técnico para realizar o serviço, e uma descrição sobre o serviço.

Figura 6 –Tela de Listagem de Serviços: Sobre o Sistema



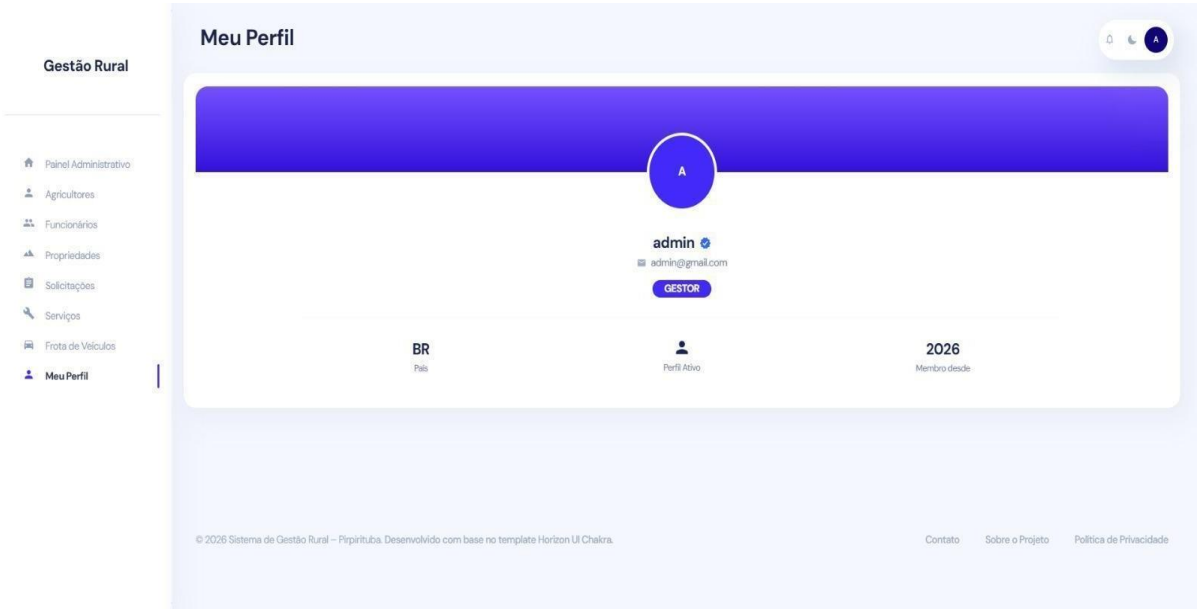
A figura 7 representa a Tela de Listagem de Veículos. Vão ser listados os veículos cadastrados com os campos de Modelo do veículo, Placa do veículo, tipo do veículo, o status do veículo.

Figura 7 –Tela de Listagem de Veículos: Sobre o Sistema



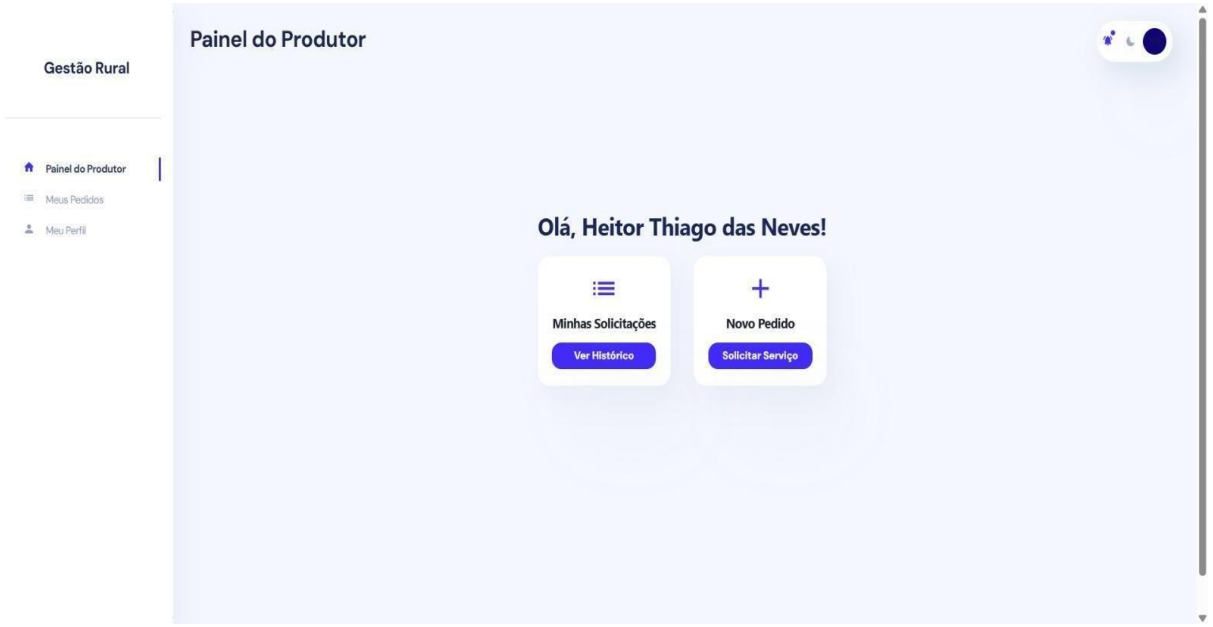
A figura 8 representa a Tela de Perfil do Administrador. Essa tela contem o nome do perfil, o email cadastrado, a função do usuário, e o ano que se cadastrou no sistema.

Figura 8 –Tela de Perfil do Administrador: Sobre o Sistema



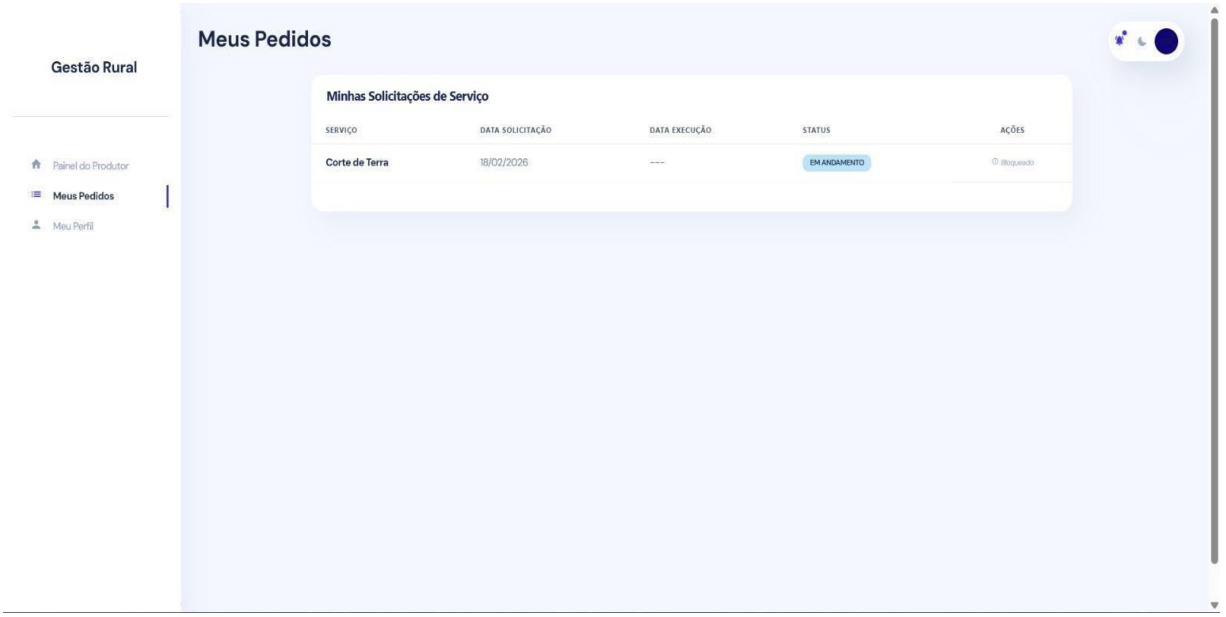
A figura 9 representa o Painel do Produtor. A tela contém dois botões, um de solicitar um serviço e um botão para ver suas solicitações de serviço.

Figura 9 –Painel do Produtor: Sobre o Sistema



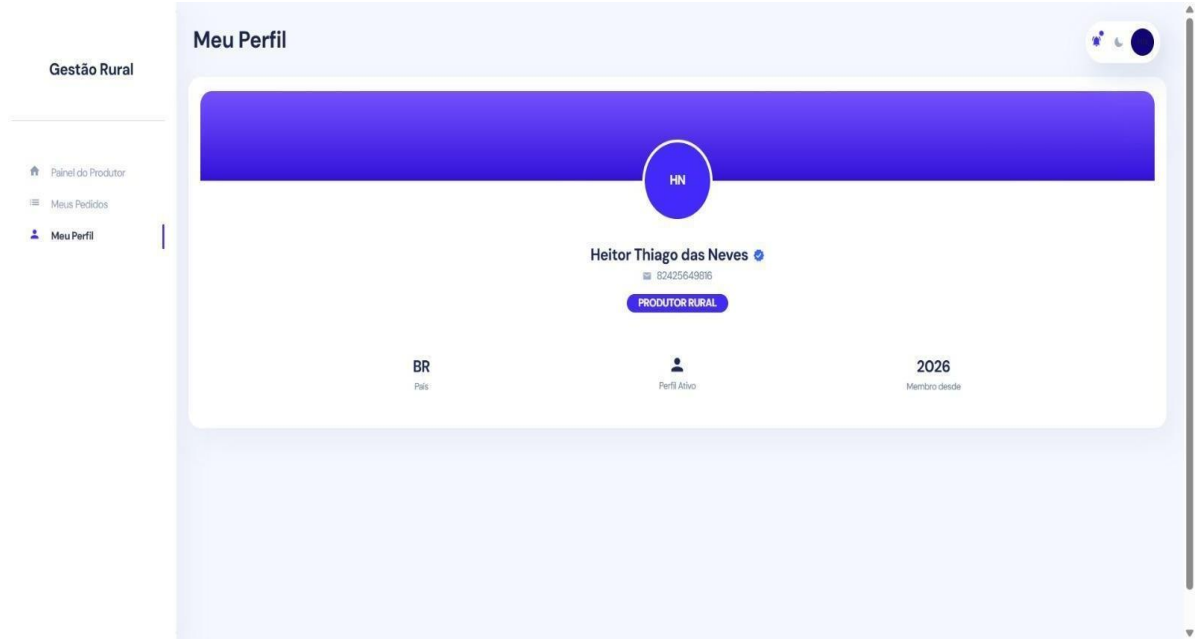
A figura 10 corresponde a Tela de Solicitações do Produtor onde aparecem os serviços solicitados pelo produtor com dados do tipo de serviços solicitado, data de solicitação do pedido, data de execução caso o pedido tenha sido concluído e o status do pedido.

Figura 10 –Tela de Solicitações do Produtor: Sobre o Sistema



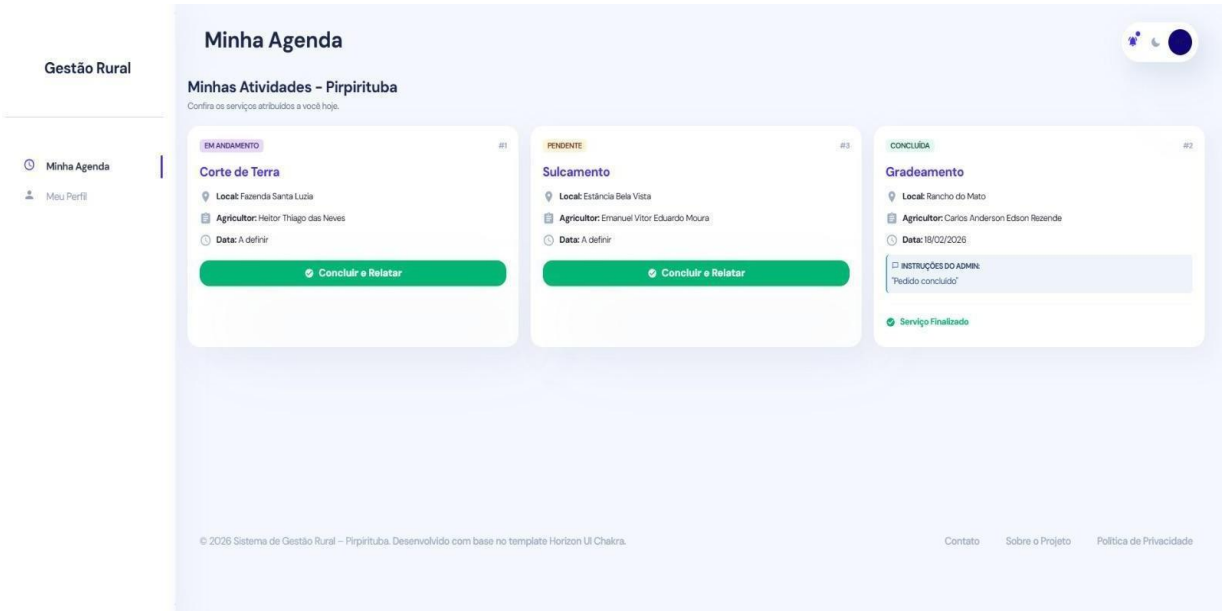
A figura 11 corresponde a Tela de Perfil do Produtor. Onde mostra o nome do produtor, o cpf cadastrado e o ano que a conta foi cadastrada no sistema.

Figura 11 –Tela de Perfil do Produtor: Sobre o Sistema



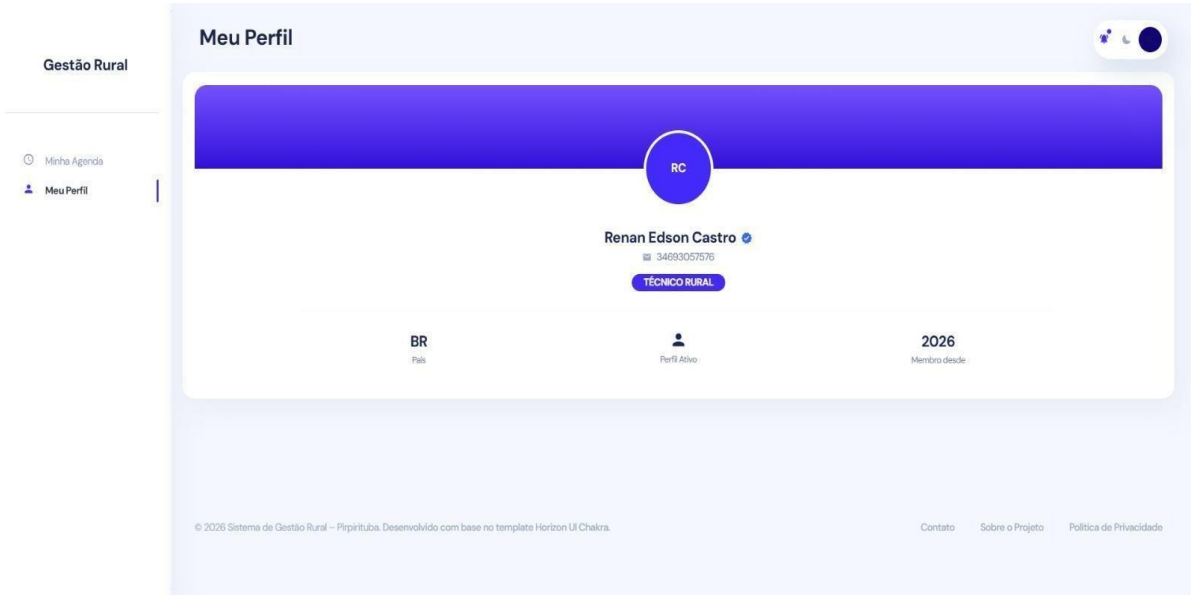
A figura 12 corresponde a Tela de Listagem de Serviços Atribuídos ao Funcionário. Nessa tela vão aparecer os serviços atribuídos ao funcionário, contendo dados do tipo de serviço solicitado, o agricultor que solicitou o serviço, a data que o serviço foi concluído e o status do serviço e um botão de conclui o serviço e relatar que concluiu para o administrador e produtor.

Figura 12 –Tela de Listagem de Serviços Atribuídos ao Funcionário: Sobre o Sistema



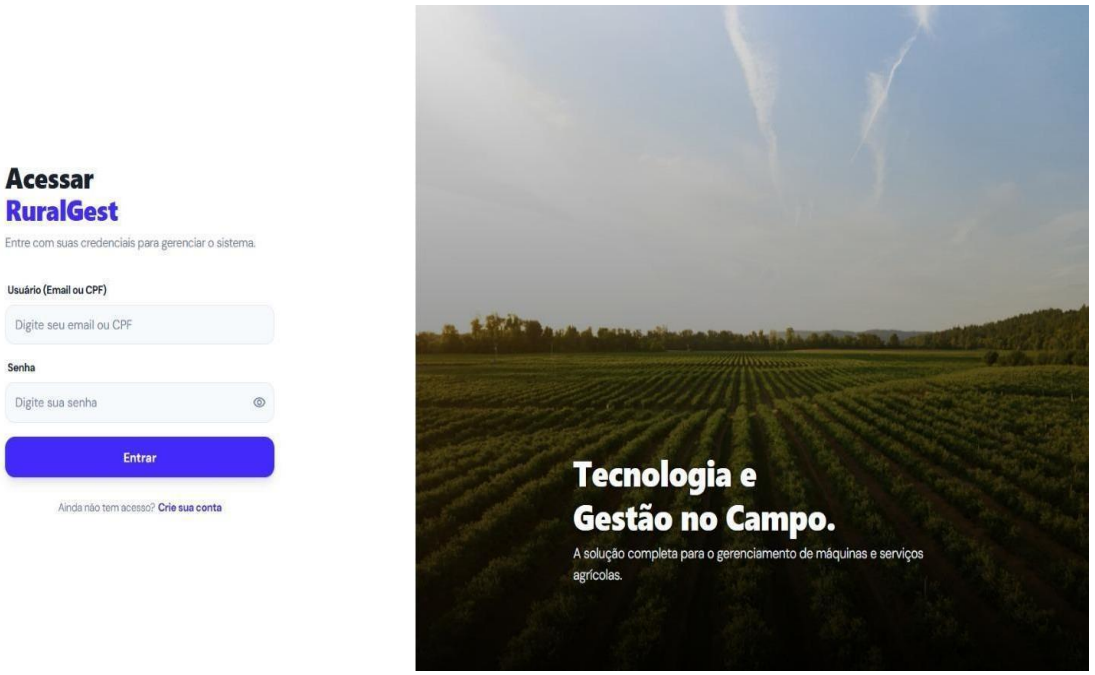
A figura 13 corresponde a Tela de Perfil do Funcionário. Essa tela contém informações sobre o nome do funcionário, cpf cadastrado, a função dele na secretária e o ano que cadastrou a conta.

Figura 13 –Tela de Perfil do Funcionário: Sobre o Sistema



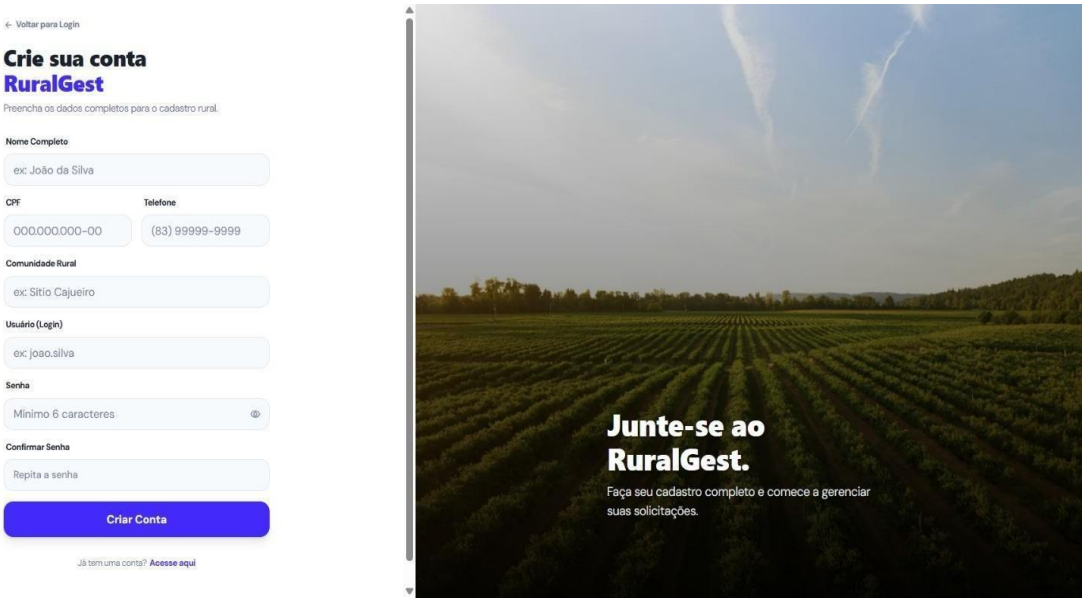
A figura 14 corresponde a Tela de Login contendo o campo de Usuário podendo ser Email ou cpf e a senha.

Figura 14 –Tela de Login: Sobre o Sistema



A figura 15 corresponde a Tela de Cadastro contendo campos de Nome completo, cpf, telefone, comunidade rural, usuário e senha.

Figura 15 –Tela de Cadastro: Sobre o Sistema



4.8 Testes do Jmeter

A fim de garantir que o sistema *RuralGest* possua a robustez necessária para operar no ambiente real da Secretaria de Agricultura, foram realizados testes de carga e estresse. O objetivo principal desta etapa foi validar o desempenho da API (Backend) sob condições de uso intenso, verificando se o tempo de resposta se mantém aceitável e se o servidor suporta múltiplas conexões simultâneas sem apresentar falhas.

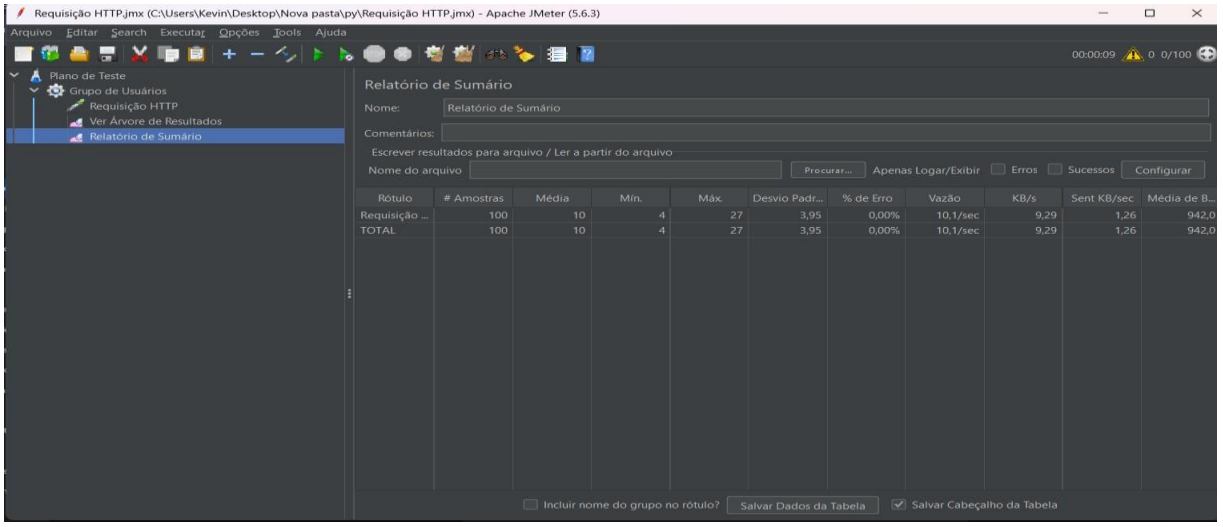
A ferramenta escolhida para a execução foi o Apache JMeter (versão 5.6.3), um software open-source amplamente utilizado na indústria para análise de performance.

4.9 Metodologia do Teste

O cenário configurado consistiu em requisições do tipo GET na rota /agricultores, que exige que o sistema processe a solicitação, consulte o banco de dados e retorne a lista de cadastros. Foram definidos três cenários progressivos para simular diferentes níveis de tráfego:

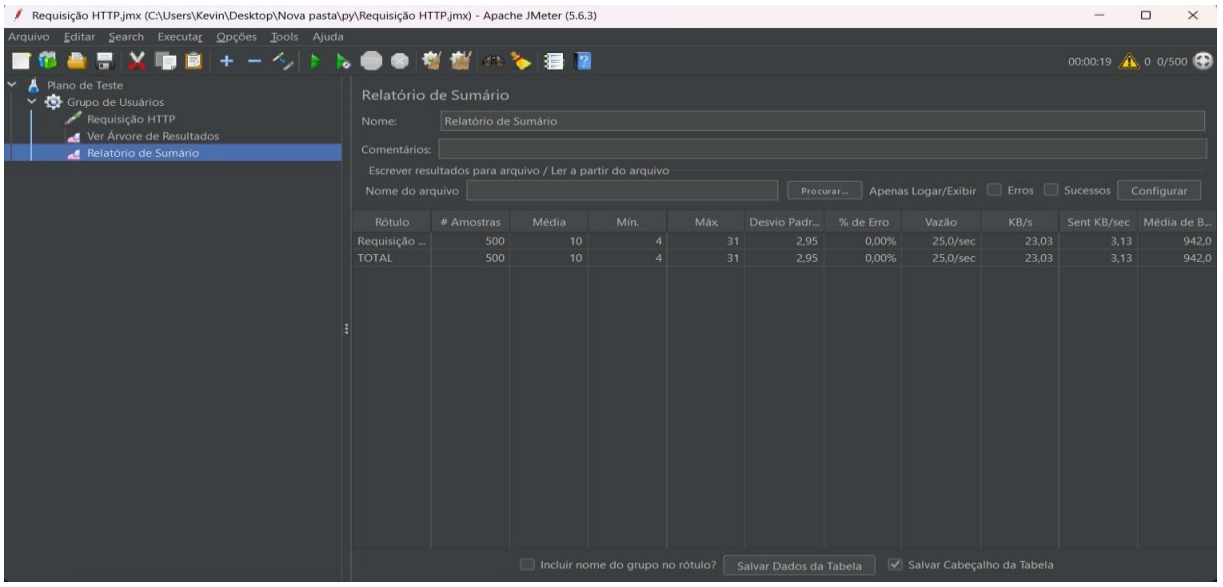
1. **Cenário de Carga Leve:** 100 usuários virtuais simultâneos (ramp-up de 10 segundos).
2. **Cenário de Carga Média:** 500 usuários virtuais simultâneos (ramp-up de 20 segundos).
3. **Cenário de Estresse Máximo**: 1.000 usuários virtuais simultâneos (ramp-up de 30 segundos).

Figura 1 – Cenário de Carga Leve (100 usuários): Testes do Jmeter



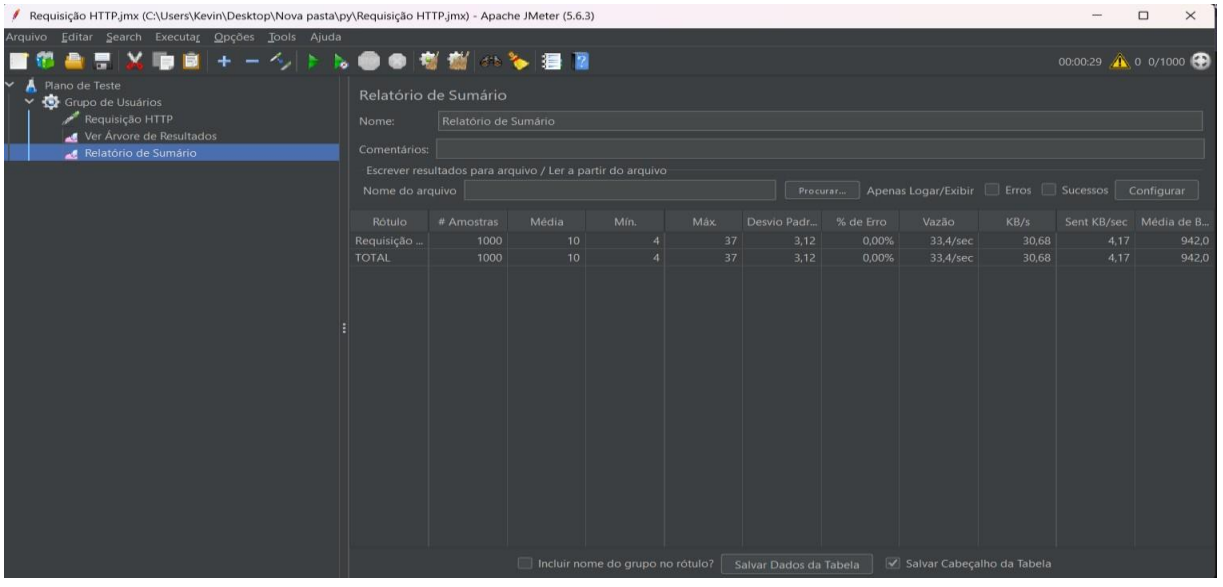
Fonte: Autoria própria (2026).

Figura 2 – Cenário de Carga Média (500 usuários): Testes do Jmeter



Fonte: Autoria própria (2026).

Figura 3 – Cenário de Carga Pesada (1000 usuários): Testes do Jmeter



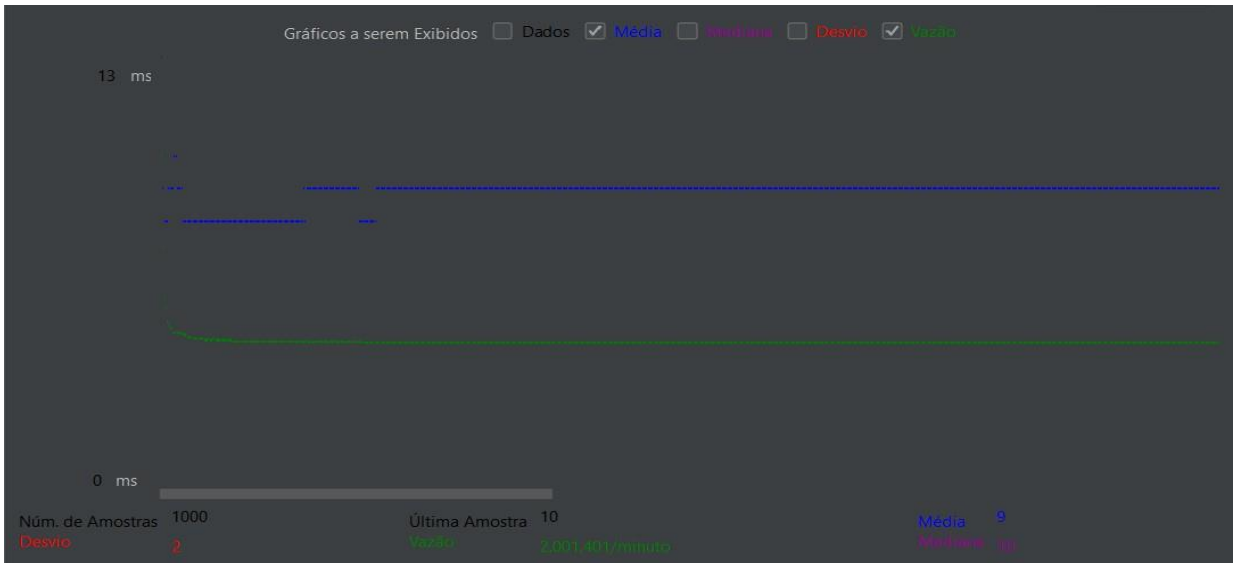
Fonte: Autoria própria (2026).

Tabela 1: Resultados Consolidados dos Testes de Carga: Testes do Jmeter

Cenário	Usuários	Tempo Médio	Vazão	Erro%
Carga Leve	100	10 ms	10,1s	0,00%
Carga Média	500	10 ms	25,0s	0,00%
Carga Pesada	1000	10 ms	33,4s	0,00%

Visualização da Estabilidade: Para validar a constância do servidor sob estresse máximo, foi gerado o gráfico de comportamento durante a execução de 1.000 usuários:

Figura 4 – Gráfico de Resultados de Estabilidade (1.000 usuários): Testes do Jmeter



Fonte: Autoria própria (2026).

Conclusão Técnica:

Análise de Desempenho: Os testes realizados demonstram uma alta estabilidade da API desenvolvida. Mesmo no cenário de estresse máximo (1000 usuários virtuais), o tempo médio de resposta manteve-se constante em 10ms, sem ocorrência de erros (HTTP 500 ou timeouts).

A vazão (throughput) aumentou linearmente conforme a carga, atingindo 33,4 requisições por segundo, o que indica que o servidor Flask foi capaz de gerenciar o escalonamento de threads de forma eficiente. O resultado valida a arquitetura para o uso previsto na Secretaria de Agricultura, suportando múltiplos acessos simultâneos sem degradação do serviço.

No gráfico a linha azul representa o tempo médio de resposta, que se manteve linear e abaixo de 15ms durante toda a execução do teste. De modo que a estabilidade visualizada comprova que não houve degradação de performance (memory leaks ou filas de processamento) mesmo sob carga máxima, validando a robustez da arquitetura em Flask.

5 CONCLUSÃO

O presente trabalho dedicou-se ao desenvolvimento do sistema RuralGest, uma solução tecnológica voltada para a modernização dos processos administrativos da Secretaria de Agricultura de Pirpirituba. O projeto surgiu da necessidade de superar as limitações impostas pelo gerenciamento manual e descentralizado, que dificultava o controle de demandas e a prestação de contas no setor público municipal.

Conclui-se que o objetivo geral foi alcançado com êxito. A implementação do sistema permitiu centralizar o cadastro de produtores rurais e propriedades, criando uma base de dados unificada e segura. A arquitetura *web* adotada, utilizando tecnologias modernas como ReactJS e Python, garantiu a entrega de uma ferramenta responsiva e acessível, que respeita as limitações de conectividade e o perfil dos usuários do campo.

Do ponto de vista técnico, a utilização de contêineres Docker e a estruturação de uma API RESTful demonstraram-se decisões acertadas, proporcionando um ambiente de desenvolvimento padronizado e escalável. O sistema não apenas digitalizou processos burocráticos, como a solicitação de serviços e a emissão de guias (GTA), mas também introduziu uma cultura de transparência e eficiência na gestão dos recursos públicos municipais.

Durante o desenvolvimento, o principal desafio enfrentado foi a adequação da interface aos requisitos de usabilidade para usuários com baixo letramento digital, o que exigiu um refinamento constante das telas e fluxos de navegação. Superada essa etapa, o **RuralGest** apresenta-se como uma ferramenta robusta, capaz de aproximar, o poder público do produtor rural.

5.1 Trabalhos Futuros

Para a evolução contínua do RuralGest, vislumbra-se a implementação de melhorias focadas na robustez da infraestrutura, na segurança da informação e no refinamento das regras de negócio.

Em relação à segurança e autenticação, pretende-se migrar o atual sistema de login para uma plataforma de gerenciamento de identidade mais robusta, como o Firebase Authentication. Essa mudança justifica-se pela necessidade de oferecer mecanismos modernos de recuperação de acesso, criptografia de ponta a ponta e gestão de sessões mais segura, retirando a complexidade da autenticação do servidor local e garantindo maior proteção aos dados sensíveis dos usuários.

No que tange à **arquitetura e escalabilidade**, planeja-se a transição para modelos de banco de dados em nuvem e a adoção de lógica *serverless* (*Cloud Functions*). O objetivo dessa atualização é otimizar o processamento de dados e permitir a sincronização de informações em tempo real entre os clientes conectados, preparando o sistema para suportar um volume crescente de requisições sem comprometer o desempenho do servidor principal.

Por fim, busca-se aprimorar a **integridade dos dados e a validade jurídica**. Futuras versões incluirão mecanismos automáticos para verificar a atualização cadastral dos agricultores, restringindo novas solicitações de usuários com dados obsoletos ou pendências documentais. Além disso, implementará-se a assinatura digital com códigos de verificação (*hash*) nos documentos gerados, garantindo a autenticidade e a segurança jurídica dos relatórios e protocolos emitidos pela Secretaria.

REFERÊNCIAS

DE PAULA, R. et al. **Políticas de modernização e inclusão digital no cenário rural**. São Paulo: Editora Acadêmica, 2014.

ISO/IEC. **ISO/IEC 25010**: Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. Geneva: ISO, 2011.

MATTEI, Lauro. **O papel da agricultura familiar no desenvolvimento do Brasil**. Florianópolis: UFSC, 2014.

NIELSEN, Jakob. **Usabilidade de software**. São Paulo: Elsevier, 2012.

PEDROSO, M. T. M. **A modernização da agricultura brasileira**: crédito e pesquisa. Brasília: Embrapa, 2017.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software**: uma abordagem profissional. 9. ed. Porto Alegre: AMGH, 2021.

REZENDE, Denis Alcides. **Engenharia de software e sistemas de informação**. 3. ed. Rio de Janeiro: LTC, 2017.

SCHNEIDER, S. et al. **As TICs e a agricultura familiar no Brasil**: dificuldades e oportunidades. Porto Alegre: UFRGS, 2016.

SOMMERVILLE, Ian. **Engenharia de software**. 10. ed. São Paulo: Pearson Education do Brasil, 2019.

SOUZA, J. et al. **Dificuldades de uso das ferramentas digitais no campo**. Revista Brasileira de Tecnologia Rural, v. 10, 2011.

BEZERRA, Eduardo. **Princípios de análise e projeto de sistemas com UML**. 3. ed. Rio de Janeiro: Elsevier, 2014.

FOWLER, Martin. **Microservices**: a definition of this new architectural term. 2014. Disponível em: https://martinfowler.com/articles/microservices.html.

GIL, Antônio Carlos. **Como elaborar projetos de pesquisa**. 6. ed. São Paulo: Atlas, 2019.

GUEDES, Gilleanes T. A. **UML 2: uma abordagem prática**. 3. ed. São Paulo: Novatec, 2018.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de Software**: uma abordagem profissional. 8. ed. Porto Alegre: AMGH, 2016.

PRODANOV, Cleber Cristiano; FREITAS, Ernani Cesar de. **Metodologia do trabalho científico**: métodos e técnicas da pesquisa e do trabalho acadêmico. 2. ed. Novo Hamburgo: Feevale, 2013.

SEGUN, Adebayo. **Chakra UI: Simple, Modular & Accessible UI Components for your**

React Applications. 2024. Disponível em: https://chakra-ui.com/.

TANENBAUM, Andrew S.; WETHERALL, David J. **Redes de Computadores**. 5. ed. São Paulo: Pearson Prentice Hall, 2011.