FACULDADE UNIGUAÇU

IGUAZU SCIENCE

EDITORA UNIVERSITÁRIA UNIGUAÇU

v. 3, n. 7, jun. 2025
ISBN 978-65-83057-12-9

ARTIGO CIENTÍFICO ACESSO LIVRE

DESENVOLVIMENTO DE SISTEMA PARA GESTÃO E VITRINE DIGITAL EM EMPRESA DE FORNECIMENTO DE DIESEL PARA AGRICULTORES

Amaury de Oliveira Lumertz*; Patrick Rocha Lumertz*; Alexssandro Ferreira Cordeiro**

*Acadêmico de Engenharia de Software, *amaury.lumertz@gmail.com; patrickrocha557@gmail.com.*
**Mestre pelo Programa de Pós-Graduação em Tecnologias Computacionais para o Agronegócio - UTFPR Medianeira, *alexssandrofc@gmail.com*.

INFORMAÇÕES

Histórico de submissão:

Recebido em: 20 maio. 2025
Aceite: 06 jun. 2025
Publicação *online*: jun. 2025

RESUMO

A gestão eficiente do fornecimento de diesel no setor agrícola é essencial para garantir a continuidade das operações no campo, especialmente em regiões onde a conectividade à internet é limitada. Nesse contexto, este artigo propõe o desenvolvimento de um sistema híbrido de gestão e vitrine digital voltado a empresas que comercializam diesel para agricultores, unificando os canais de atendimento digital e tradicional. A solução contempla dois módulos principais: uma vitrine digital acessível via dispositivos com conexão à internet, desenvolvida em React, e um painel administrativo para registro manual de pedidos feitos por telefone ou presencialmente, assegurando a inclusão de clientes com acesso restrito à tecnologia. O backend do sistema, construído com Node.js e banco de dados MySQL, gerencia a lógica de negócios, autenticação, controle de estoque em tempo real e integração entre os módulos. A metodologia adotada baseou-se em práticas ágeis de desenvolvimento, com levantamento de requisitos junto à empresa parceira, modelagem modular do sistema, testes automatizados e manuais com o apoio da ferramenta Postman. Os resultados esperados incluem maior controle e rastreabilidade dos pedidos, redução de falhas humanas, agilidade na logística e aumento da satisfação do cliente. O sistema foi projetado para ser responsivo e funcional em ambientes de baixa largura de banda, visando atender às peculiaridades da realidade rural brasileira. Ao integrar tecnologias modernas e estratégias adaptadas ao público-alvo, a proposta oferece uma alternativa viável e escalável para empresas do setor, promovendo eficiência operacional, inclusão digital e competitividade no mercado agrícola.
Palavras-chave: gestão de diesel; vitrine digital; Node.js; React; agricultura; inclusão digital; sistema híbrido.

ABSTRACT

Efficient diesel supply management is essential to ensure the continuity of agricultural operations, particularly in rural areas with limited internet access. This article presents the development of a hybrid management and digital showcase system designed for companies supplying diesel to farmers. The solution integrates a web-based ordering interface, developed with React, and an administrative module for manually registering orders received by phone or in person. The backend, built with Node.js and MySQL, handles business logic, authentication, real-time inventory control, and data integration. The development process followed agile methodologies, including requirements gathering, modular system design, and testing using tools such as Postman. The system aims to reduce human errors, improve logistics, and enhance customer satisfaction by providing a unified platform adaptable to environments with low bandwidth. By aligning modern technologies with the specific needs of rural contexts, the proposed system offers a scalable and inclusive solution for enhancing operational efficiency and digital access in the agricultural sector
Keywords: diesel management; digital showcase; Node.js; React; agriculture; digital inclusion; hybrid system.

***Copyright* © 2025, Amaury de Oliveira Lumertz; Patrick Rocha Lumertz; Alexssandro Ferreira Cordeiro**. *This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.*

Citação: LUMERTZ, Amaury de Oliveira; LUMERTZ, Patrick Rocha; CORDEIRO, Alexssandro Ferreira. Desenvolvimento de sistema para gestão e vitrine digital em empresa de fornecimento de diesel para agricultores. **Iguazu Science**, São Miguel do Iguaçu, v. 3, n. 7, p. 145-154, jun. 2025.

INTRODUÇÃO

A agricultura representa um dos pilares da economia brasileira, tendo papel crucial na geração de renda e emprego, especialmente nas regiões rurais (IBGE, 2022). A mecanização crescente das atividades agrícolas, impulsionada pela adoção de tratores, colheitadeiras e outros equipamentos, tem demandado o uso contínuo de combustíveis, especialmente o óleo diesel, cuja oferta regular é vital para a manutenção dos ciclos produtivos e para a sustentabilidade da produção rural (Sommerville, 2011). O abastecimento de maquinário agrícola com diesel é essencial para o andamento das atividades no campo, tornando-se um insumo estratégico cujo controle eficaz impacta diretamente na produtividade e na rentabilidade dos produtores.

Contudo, o cenário de fornecimento de diesel aos agricultores enfrenta desafios operacionais significativos. Em primeiro lugar, a infraestrutura de internet em muitas áreas rurais é limitada, o que impede que usuários recorram exclusivamente a plataformas online para realizar seus pedidos, obrigando-os a utilizar canais manuais como telefone e visitas presenciais (Oliveira; Costa, 2020). Soma-se a isso a diversidade de canais de atendimento — do tradicional contato telefônico à crescente adoção de aplicativos móveis — que gera fragmentação no fluxo de pedidos e aumenta a probabilidade de erros e retrabalho (Pereira *et al*., 2019).

Além disso, a coexistência de sistemas manuais e plataformas isoladas impede um controle centralizado do estoque e dificulta o acompanhamento em tempo real das solicitações de diesel, comprometendo a precisão das previsões de entrega e a gestão de inventário (Martins; Souza, 2018). Essa falta de integração operacional afeta diretamente a eficiência logística e a qualidade do atendimento, levando a atrasos, baixa satisfação do cliente e retrabalho nas rotinas administrativas (Ferreira, 2020).

Nesse contexto, torna-se essencial a adoção de soluções tecnológicas que conciliem inovação, acessibilidade e adaptação à realidade do meio rural. Este artigo propõe o desenvolvimento de um sistema informatizado híbrido, composto por dois módulos integrados: uma vitrine digital acessível por meio de navegadores web, destinada aos clientes com acesso à internet, e um módulo de atendimento tradicional, operado por funcionários da empresa, para registrar e gerenciar pedidos realizados por telefone, mensagens ou presencialmente. Essa abordagem visa não apenas digitalizar os processos de venda e gestão de estoque, mas também promover a inclusão digital de regiões com baixa conectividade.

O sistema foi desenvolvido utilizando tecnologias modernas e consolidadas como React para a construção da interface do usuário, Node.js para o backend, MySQL para armazenamento e Postman para testes e validação de APIs. A arquitetura modular adotada garante escalabilidade, segurança e manutenção facilitada, alinhando-se às boas práticas de desenvolvimento de software (Fowler, 2003; Tilkov; Vinoski, 2010). O projeto seguiu uma metodologia ágil com entregas incrementais e validação contínua junto aos usuários, conforme os princípios do Manifesto Ágil (Beck *et al*., 2001).

Assim, o presente trabalho tem como objetivo principal apresentar a concepção, desenvolvimento e validação de um sistema de gestão e pedidos para empresas de fornecimento de diesel voltadas ao setor agrícola, com foco na integração de canais, automação de processos e adaptação a diferentes níveis de conectividade. Espera-se, com isso, contribuir para a transformação digital do setor, promovendo maior eficiência operacional, redução de custos e ampliação do acesso a soluções tecnológicas no meio rural.

METODOLOGIA

O desenvolvimento do sistema proposto seguiu uma abordagem metodológica estruturada, pautada em princípios de engenharia de software e práticas ágeis de desenvolvimento, com o objetivo de assegurar a qualidade técnica, a aderência às necessidades dos usuários e a viabilidade de implementação em contextos rurais. A metodologia adotada foi dividida em cinco etapas principais: levantamento de requisitos, modelagem do sistema, seleção tecnológica, desenvolvimento incremental e validação/testes. Cada fase foi conduzida de forma iterativa, com a participação ativa dos stakeholders, buscando alinhar continuamente os objetivos técnicos e operacionais do projeto.

1. Levantamento de Requisitos

A primeira etapa consistiu na coleta e análise de requisitos funcionais e não funcionais, realizada por meio de entrevistas semiestruturadas, observações diretas e reuniões com gestores, operadores e funcionários da empresa fornecedora de diesel. O objetivo foi compreender os fluxos de trabalho existentes, identificar gargalos operacionais e mapear as necessidades específicas do negócio, como a rastreabilidade de pedidos, o controle de estoque em tempo real e a inclusão de clientes sem acesso à internet. Foram utilizadas técnicas de elicitação recomendadas por Sommerville (2011), como análise de tarefas, brainstorming e prototipagem inicial para validar premissas e expectativas dos usuários finais.

2. Modelagem do Sistema

Com base nas informações obtidas, foi elaborada a modelagem conceitual e arquitetural do sistema, empregando diagramas de casos de uso, diagramas de sequência e diagramas de entidade-relacionamento para representar o comportamento esperado da aplicação e suas interações com o banco de dados. Optou-se por uma arquitetura em camadas (*layered architecture*), com separação clara entre a interface do usuário (*frontend*), a lógica de negócios (*backend*) e a persistência de dados (banco de dados), visando maior coesão modular, facilidade de manutenção e escalabilidade. Essa abordagem segue as recomendações de Fowler (2003) para sistemas empresariais com múltiplos pontos de entrada.

3. Seleção Tecnológica

A escolha das tecnologias baseou-se em critérios como robustez, comunidade ativa, curva de aprendizado, compatibilidade com dispositivos móveis e potencial de escalabilidade. Foram adotadas as seguintes ferramentas:

React: Biblioteca JavaScript para construção de interfaces de usuário reativas e componentizadas, com foco na responsividade e desempenho, ideal para aplicações com atualizações frequentes (Banks; Porcello, 2017).

Node.js: Plataforma para execução de código JavaScript no servidor, baseada em modelo assíncrono e orientado a eventos, apropriada para aplicações que exigem alta concorrência e baixa latência (Tilkov; Vinoski, 2010).

MySQL: Sistema de gerenciamento de banco de dados relacional amplamente utilizado, com suporte a transações ACID e integração facilitada com o ecossistema JavaScript (DuBois, 2013).

Postman: Ferramenta para criação, teste e documentação de APIs RESTful, utilizada no processo de validação funcional e integração entre os módulos do sistema (Postman, 2024).

4. Desenvolvimento Incremental e Iterativo

A implementação seguiu os princípios do desenvolvimento ágil, com entregas parciais e frequente interação com os usuários para coleta de feedback contínuo. Cada funcionalidade foi desenvolvida em sprints curtos, com ciclos de análise, codificação, testes e revisão. Essa abordagem permitiu ajustar funcionalidades conforme as demandas emergentes do cliente e incorporar melhorias incrementais ao longo do projeto, reduzindo riscos e promovendo maior alinhamento entre solução técnica e realidade operacional (Beck *et al*., 2001).

Foram criados dois módulos principais:

Vitrine Digital: Interface web acessível via dispositivos móveis ou desktops, que permite ao cliente realizar pedidos, visualizar catálogo de produtos, consultar histórico de transações e receber confirmações automatizadas.

Módulo de Atendimento: Painel administrativo utilizado pelos operadores da empresa para registrar pedidos recebidos por telefone, presencialmente ou por outros canais offline, garantindo a unificação dos dados no sistema central.

5. Testes e Validação

A fase final consistiu na aplicação de testes de software em duas frentes principais: testes unitários no backend e testes funcionais na interface do usuário. No backend, os endpoints da API foram testados com Postman, verificando status HTTP, consistência dos dados e tratamento de erros. Exemplos incluem:

POST /pedido – Criação de novo pedido, com retorno esperado 201 (Created).

GET /produtos – Consulta ao catálogo, com retorno esperado 200 (OK).

No frontend, os testes focaram na usabilidade, responsividade e fluxo de navegação, realizados manualmente e com o apoio da React Testing Library. Foram testadas funcionalidades como login, carrinho de compras, filtros de produto e envio de pedido, em diferentes tamanhos de tela e navegadores.

Adicionalmente, foram conduzidos testes de desempenho e validação em campo, em ambiente com conectividade limitada, para verificar a estabilidade da aplicação e a viabilidade de operação em regiões rurais. Essa etapa foi essencial para garantir que o sistema atendesse não apenas aos requisitos funcionais, mas também às restrições contextuais do público-alvo.

SISTEMA DESENVOLVIDO

4.1. Arquitetura e Componentes

O sistema é dividido em dois módulos integrados:

- Vitrine Digital: Interface web para clientes realizarem pedidos online, exibindo catálogo de produtos, preços e histórico de pedidos. Desenvolvida em React, utiliza Bootstrap para responsividade e acessibilidade.

- Módulo de Atendimento Direto: Interface para operadores registrarem pedidos oriundos de contato telefônico ou presencial, garantindo inclusão no sistema unificado.

Ambos os módulos se comunicam com o backend Node.js, que gerencia autenticação, autorização, lógica de negócios, persistência e regras de estoque, utilizando MySQL para armazenamento dos dados.

Fluxo de Operação

Cliente faz pedido pela vitrine ou atendente registra pedido manualmente.

Backend valida, processa e grava o pedido no banco.

Estoque é atualizado em tempo real.

Confirmação de pedido é disponibilizada.

Interface da Vitrine Digital

- Página inicial com catálogo paginado.

- Busca e filtros por tipos de diesel e volumes.

- Carrinho de compras com resumo e possibilidade de edição.
- Cadastro e login com autenticação JWT.
- Histórico de pedidos para acompanhamento.

Módulo de Atendimento
- Tela de inserção rápida de pedidos.
- Visualização de produtos e preços atualizados.
- Registro de observações e condições especiais.
- Busca de clientes por CPF ou nome.

A Figura 1 apresenta o relatório de funcionários disponível no módulo administrativo do sistema, demonstrando a estrutura organizacional interna da empresa. Essa interface exibe informações essenciais como nomes, cargos, setores e alocações funcionais dos colaboradores, permitindo uma gestão mais eficiente dos recursos humanos. O relatório facilita o controle de permissões, a definição de responsabilidades operacionais e o monitoramento de atividades, servindo de base para análises gerenciais e tomada de decisões estratégicas relacionadas à força de trabalho.

Figura 1 – Relatório de funcionários no sistema

Fonte: Autoria própria (2025).

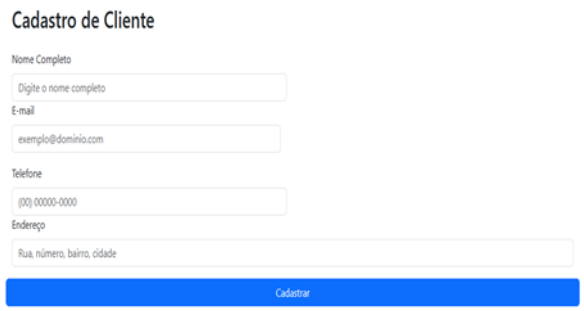
Na Figura 2 é exibido o relatório de clientes, o qual reúne dados cadastrais fundamentais como nome, CPF, endereço e informações de contato. Essa funcionalidade tem papel central na gestão do relacionamento com o cliente, permitindo ao atendente localizar rapidamente registros existentes, atualizar informações e acessar o histórico de pedidos. A organização clara dos dados visa não apenas agilizar o processo de atendimento, mas também assegurar a consistência e a integridade das informações armazenadas no banco de dados.

Figura 2 – Relatório de clientes do sistema

Fonte: Autoria própria (2025).

A Figura 3 ilustra a tela de cadastro de clientes, interface que permite a inserção de novos registros no sistema. A tela foi desenvolvida com foco na simplicidade e na eficiência, apresentando campos obrigatórios como nome completo, endereço, CPF e formas de contato. Sua usabilidade foi otimizada para dispositivos de diferentes tamanhos e para conexões com baixa largura de banda, características comuns no meio rural. Essa funcionalidade garante que novos clientes possam ser inseridos com agilidade, mantendo o banco de dados sempre atualizado e promovendo uma gestão mais eficaz da base de consumidores.

Figura 3 – Tela de cadastro de clientes



Fonte: Autoria própria (2025).

TECNOLOGIAS UTILIZADAS

Node.js

Node.js é uma plataforma de desenvolvimento de aplicações de servidor baseada no motor V8 do Google Chrome, que interpreta código JavaScript. Diferentemente dos ambientes tradicionais, Node.js adota um modelo de execução assíncrono e orientado a eventos, o que permite a construção de aplicações altamente escaláveis e eficientes no uso de recursos, especialmente para operações de I/O (entrada/saída) (Tilkov; Vinoski, 2010).

Esse modelo não bloqueante possibilita que o servidor execute múltiplas operações simultaneamente, sem aguardar o término de uma tarefa para iniciar outra, otimizando o processamento e a capacidade de atendimento a múltiplas requisições concorrentes. Além disso, a arquitetura single-threaded do Node.js evita overhead causado por criação excessiva de threads, utilizando a event loop para gerenciar as operações assíncronas (Dahl, 2009).

O ecossistema de Node.js é vasto e conta com o gerenciador de pacotes npm, que disponibiliza milhares de bibliotecas e módulos que facilitam o desenvolvimento, como frameworks para construção de APIs REST, ferramentas para acesso a bancos de dados, manipulação de arquivos, autenticação, entre outros (npm, 2024).

Um exemplo prático é a criação de servidores HTTP utilizando o *framework Express.js,* que abstrai funcionalidades comuns e permite estruturar rotas e middlewares de forma simples e organizada. Essa abordagem acelera o desenvolvimento e manutenção de aplicações web e serviços back-end (Holmes, 2015).

Assim, Node.js tem se destacado em projetos que demandam alta performance e escalabilidade, como sistemas em tempo real, APIs RESTful e microserviços, consolidando-se como uma tecnologia essencial no desenvolvimento web moderno.

React

React é uma biblioteca JavaScript open source criada pelo Facebook em 2013 para facilitar o desenvolvimento de interfaces de usuário (UI) complexas e dinâmicas por meio da composição de componentes reutilizáveis. A sua principal inovação é a utilização do DOM virtual (Virtual DOM), uma representação leve do DOM real, que permite atualizar apenas os elementos necessários na interface, otimizando o desempenho e melhorando a experiência do usuário (Banks; Porcello, 2017).

O DOM virtual é uma representação leve e abstrata da estrutura do DOM real, permitindo que o React faça comparações (*diffing*) entre o estado atual e o novo estado da UI para aplicar apenas as mudanças estritamente necessárias, evitando a re-renderização completa do DOM, processo que é tipicamente custoso em termos de performance. Essa técnica possibilita que o React mantenha a interface responsiva mesmo em aplicações com muitos componentes e atualizações frequentes (Tilkov; Vinoski, 2010).

A arquitetura baseada em componentes do React promove a divisão da UI em unidades independentes, cada uma responsável por sua própria lógica e apresentação. Cada componente pode conter um estado interno e receber propriedades (props) que controlam seu comportamento e aparência, o que favorece a modularidade, a reutilização e a manutenção do código. Essa separação clara de responsabilidades facilita a colaboração em equipes de desenvolvimento e a escalabilidade dos projetos (Choi, 2020).

Outra característica marcante do React é seu paradigma declarativo. Ao invés de manipular diretamente o DOM para refletir mudanças, o desenvolvedor descreve a UI desejada para cada estado da aplicação, e o React se encarrega de manter a interface sincronizada com esse estado. Essa abordagem reduz erros e torna o código mais legível e previsível (Abramov, 2015).

A introdução dos hooks na versão 16.8 do React representou uma evolução significativa, permitindo que funcionalidades como gerenciamento de estado, ciclo de vida do componente e manipulação de efeitos colaterais fossem implementadas diretamente em componentes funcionais, simplificando a arquitetura do código e reduzindo a complexidade do uso de componentes baseados em classes. Hooks como `useState`, `useEffect`, `useContext` e `useReducer` proporcionam mecanismos poderosos e flexíveis para gerenciar a lógica da aplicação de forma mais concisa e intuitiva (Abramov, 2019).

O ecossistema do React é rico e diversificado, contando com diversas bibliotecas e ferramentas que complementam sua funcionalidade principal. Entre elas, destaca-se o React Router, que possibilita o gerenciamento eficiente da navegação entre diferentes telas em aplicações single-page (SPAs), o Redux e Recoil para controle centralizado e eficiente do estado global da aplicação, além de ferramentas para integração com APIs RESTful ou GraphQL, testes automatizados (Jest, React Testing Library) e sistemas de build e bundling (Webpack, Babel) (Choi, 2020).

Além do ambiente web, o React também possibilita o desenvolvimento mobile por meio do React Native, que permite a criação de aplicações nativas para iOS e Android utilizando o mesmo paradigma de componentes e lógica do React, promovendo a reutilização de conhecimento e código entre plataformas (Facebook, 2024).

Devido à sua flexibilidade, eficiência e suporte comunitário ativo, React consolidou-se como uma das bibliotecas JavaScript mais utilizadas mundialmente para o desenvolvimento frontend, sendo adotada por grandes empresas e projetos de todos os portes. Sua evolução contínua e o vasto ecossistema garantem que o React se mantenha relevante frente às constantes mudanças no desenvolvimento web moderno.

MySQL

MySQL é um sistema de gerenciamento de banco de dados relacional (SGBDR) amplamente utilizado no desenvolvimento de aplicações que demandam armazenamento estruturado e confiável de dados. Originalmente desenvolvido pela empresa sueca MySQL AB em 1995 e atualmente mantido pela Oracle Corporation, o MySQL se destaca por sua eficiência, facilidade de uso e ampla compatibilidade com diversos sistemas operacionais e linguagens de programação (DuBois, 2008).

Baseado no modelo relacional, o MySQL organiza os dados em tabelas compostas por linhas (tuplas) e colunas (atributos), o que facilita a estruturação, consulta e manipulação das informações. Ele utiliza a linguagem SQL (Structured Query Language), padrão internacional para gerenciamento de bancos de dados relacionais, permitindo executar operações como criação, leitura, atualização e exclusão de dados (CRUD), além de consultas complexas envolvendo múltiplas tabelas por meio de junções (JOINs) e subconsultas (Coronel; Morris, 2016).

A arquitetura cliente-servidor do MySQL possibilita que múltiplos clientes acessem simultaneamente o banco de dados, o que o torna adequado para ambientes web, sistemas corporativos,

comércio eletrônico, aplicações móveis e diversos outros cenários. Essa capacidade de suportar alta concorrência é reforçada pelo uso de mecanismos de armazenamento (storage engines), sendo o InnoDB o mais utilizado, que oferece suporte a transações com propriedades ACID (Atomicidade, Consistência, Isolamento e Durabilidade), garantindo a integridade e confiabilidade dos dados mesmo em situações de falhas ou acesso concorrente (DuBois, 2008).

Além disso, o MySQL permite a definição de chaves primárias, estrangeiras, índices e restrições, que asseguram a integridade referencial e otimizam a performance das consultas. Essas características são essenciais para manter a coerência dos dados em sistemas complexos que envolvem múltiplas entidades relacionadas, como clientes, produtos, pedidos e estoque (Coronel; Morris, 2016).

A criação de tabelas no MySQL é realizada por meio de comandos SQL que definem os campos, seus tipos de dados e restrições. Por exemplo, a tabela `produtos`, apresentada abaixo, armazena informações básicas sobre produtos disponíveis, incluindo um identificador único, nome, descrição, preço e quantidade em estoque:

```
CREATE TABLE produtos (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nome VARCHAR(100),
  descricao TEXT,
  preco DECIMAL(10,2),
  estoque INT
);
```

Neste exemplo, o campo `id` é definido como chave primária auto-incrementável, garantindo um identificador único para cada produto inserido. O campo `nome` aceita até 100 caracteres, adequado para títulos ou nomes de produtos, enquanto `descricao` utiliza o tipo `TEXT` para armazenar descrições detalhadas. O campo `preco` é definido como decimal com precisão de 10 dígitos, sendo 2 deles para casas decimais, ideal para valores monetários. Por fim, o campo `estoque` armazena a quantidade disponível de cada produto.

O MySQL também possui recursos avançados, como replicação para alta disponibilidade, partição de tabelas para melhor performance em grandes volumes de dados, e suporte a stored procedures, triggers e views, que permitem maior controle e automação dentro do banco de dados (DuBois, 2008).

A integração do MySQL com diversas linguagens, como PHP, Python, Java, Node.js, entre outras, aliada à sua documentação robusta e comunidade ativa, faz com que seja uma escolha frequente para desenvolvedores e empresas que buscam uma solução madura, escalável e flexível para gerenciamento de dados (Coronel; Morris, 2016).

Em resumo, o MySQL oferece uma plataforma confiável e eficiente para armazenamento estruturado de dados, sendo capaz de atender desde aplicações pequenas até grandes sistemas corporativos, com uma gama extensa de funcionalidades que garantem segurança, integridade e desempenho.

Postman

Postman é uma plataforma completa e amplamente adotada para o desenvolvimento, teste, documentação e monitoramento de APIs (Application Programming Interfaces). Fundada em 2012 inicialmente como uma extensão para navegadores, evoluiu rapidamente para uma aplicação desktop e baseada em nuvem, consolidando-se como uma das ferramentas essenciais para engenheiros de software, equipes de qualidade e desenvolvedores de API em todo o mundo (Postman, 2024).

A funcionalidade central do Postman reside na criação e execução de requisições HTTP, que são fundamentais para o funcionamento das APIs RESTful, SOAP e GraphQL. A ferramenta oferece suporte a diversos métodos HTTP, incluindo GET, POST, PUT, DELETE, PATCH, OPTIONS e HEAD, além da capacidade de configurar todos os aspectos da requisição, como cabeçalhos, parâmetros de consulta (query params), corpo da mensagem em múltiplos formatos (JSON, XML, texto simples, form-data) e autenticação, com suporte a diversos protocolos e métodos como OAuth 2.0, API Keys, Basic Auth e Bearer Tokens (Postman, 2024).

Além da execução isolada de requisições, o Postman permite a organização das mesmas em coleções estruturadas, que podem conter múltiplas chamadas agrupadas logicamente por funcionalidades ou serviços, facilitando a execução sequencial, testes automatizados e o versionamento de APIs. As coleções podem ser compartilhadas entre membros da equipe, permitindo colaboração e padronização dos processos de teste e desenvolvimento (Postman, 2024).

O Postman disponibiliza um sistema avançado de scripts para pré-requisições e testes, escritos em JavaScript, que possibilitam manipular variáveis, validar respostas (status HTTP, corpo, cabeçalhos), extrair dados para uso em requisições subsequentes, simular fluxos complexos de autenticação e implementar lógica condicional durante os testes. Essa capacidade torna possível a automação extensiva dos testes, integrando o Postman em pipelines de integração contínua e entrega contínua (CI/CD) por meio de sua ferramenta de linha de comando, o Newman (Postman, 2024).

Outra funcionalidade importante é o suporte a múltiplos ambientes configuráveis, que permitem alternar rapidamente entre diferentes configurações de servidores (desenvolvimento, teste, homologação, produção), credenciais e variáveis globais, otimizando o processo de desenvolvimento e testes em múltiplos contextos sem necessidade de alteração manual dos dados nas requisições (Postman, 2024).

O Postman também oferece recursos para monitoramento contínuo das APIs, permitindo a

programação de execuções periódicas de coleções para validar o funcionamento, desempenho e conformidade dos serviços, gerando relatórios e alertas em caso de falhas ou regressões. Isso auxilia no acompanhamento da saúde da API em ambientes de produção e pré-produção (Postman, 2024).

Além dos recursos técnicos, o Postman permite a geração automática de documentação interativa, que pode ser publicada e atualizada em tempo real, facilitando a comunicação entre equipes de desenvolvimento, clientes, suporte e demais stakeholders, promovendo transparência e agilidade no processo de desenvolvimento e manutenção de APIs (Postman, 2024).

A plataforma é altamente integrada a ferramentas de controle de versão, sistemas de gerenciamento de projetos e fluxos de trabalho DevOps, oferecendo suporte para colaboração em equipe, controle de permissões, versionamento de coleções e integração com plataformas como GitHub, Jenkins, Travis CI e outras. Isso permite que o Postman seja incorporado de forma eficiente em processos organizacionais de desenvolvimento ágil e DevOps (Postman, 2024).

Dada a crescente importância das APIs em arquiteturas modernas, especialmente com a adoção de microserviços, computação em nuvem e aplicações móveis, o Postman tornou-se uma ferramenta indispensável para garantir a qualidade, segurança e desempenho dessas interfaces, sendo adotado por milhões de usuários em todo o mundo, desde pequenos desenvolvedores até grandes corporações (Postman, 2024).

TESTES E VALIDAÇÃO

A etapa de testes e validação teve como principal objetivo assegurar que o sistema atendesse aos requisitos funcionais e não funcionais previamente definidos, garantindo confiabilidade, desempenho, usabilidade e aderência às condições reais de uso. Essa fase foi conduzida de forma sistemática, combinando testes automatizados e manuais, aplicados tanto no backend quanto na interface do usuário, além de simulações em ambiente real de operação rural.

5.1 Testes Unitários e de Integração no Backend

Para verificar a correta implementação dos serviços de backend, foram realizados testes unitários em cada endpoint da API REST desenvolvida em Node.js, utilizando o Postman para simulação de requisições HTTP. As rotas foram avaliadas com base nos seguintes critérios:

- Correto funcionamento dos métodos (GET, POST, PUT, DELETE)
- Validação de dados obrigatórios e formatos
- Respostas esperadas (status HTTP e corpo da resposta)
- Resistência a entradas inválidas (testes negativos)

Tabela 1 – Principais Endpoints Testados no Sistema

Requisição	Método	URL	Entrada Simulada	Resultado Esperado
Criar pedido	POST	/pedido	JSON com produtos, cliente e data	HTTP 201 Created
Listar produtos	GET	/produtos	N/A	HTTP 200 OK
Atualizar estoque	PUT	/estoque/:id	ID do produto e nova quantidade	HTTP 200 OK
Buscar cliente	GET	/clientes	Parâmetro `cpf` ou `nome`	HTTP 200 OK ou 404 Not Found

Fonte: Altoria própria (2025).

Além disso, foram testadas interações entre os módulos (integração) para garantir a consistência dos dados trafegados entre frontend e backend, bem como a integridade no banco de dados MySQL.

5.2 Testes Funcionais no Frontend

Os testes funcionais visaram validar o comportamento da interface web desenvolvida em React, com foco na navegabilidade, acessibilidade e cumprimento do fluxo de uso esperado pelo cliente. Os testes foram realizados manualmente em navegadores distintos (Google Chrome, Mozilla Firefox e Microsoft Edge) e em dispositivos com diferentes resoluções de tela, incluindo smartphones com conexão 3G para simulação de baixa largura de banda.

Entre as funcionalidades testadas, destacam-se:

- Cadastro e login de usuário (com autenticação JWT)
- Busca e filtro de produtos por categoria e volume
- Adição e remoção de itens no carrinho
- Registro e envio de pedido
- Visualização do histórico de compras

Utilizou-se a React Testing Library para simular interações com os componentes da interface, validando se os elementos visuais respondiam conforme o esperado a eventos como cliques, preenchimento de campos e carregamento de dados.

5.3 Testes de Usabilidade e Experiência do Usuário

A usabilidade foi avaliada com base em testes exploratórios conduzidos com usuários reais da empresa parceira (operadores e clientes), que interagiram com o sistema em situações práticas do cotidiano. Foram observadas:

- Clareza e legibilidade da interface
- Tempo médio de conclusão de tarefas
- Dificuldades encontradas e feedback espontâneo

Esses testes permitiram ajustes pontuais, como ampliação de ícones, reposicionamento de botões, mensagens de erro mais claras e melhoria da responsividade em telas pequenas.

5.4 Testes de Desempenho e Conectividade

Considerando o contexto rural, com conectividade limitada, foram realizados testes de desempenho em redes com baixa largura de banda (3G, 2 Mbps) para verificar o tempo de resposta e a estabilidade da aplicação. A vitrine digital foi otimizada para reduzir o carregamento inicial com lazy loading de componentes e compactação de imagens e scripts.

Foram medidos:

- Tempo médio de carregamento da página inicial: ~1,7 segundos em rede 3G
- Tempo médio para conclusão de pedido: ~5 segundos
- Consumo médio de dados por sessão: ~1,2 MB

Essas métricas demonstram que o sistema se mantém funcional mesmo sob restrições técnicas, garantindo acesso a usuários de localidades com infraestrutura precária.

5.5 Critérios de Aceitação e Validação Final

A validação do sistema considerou critérios de aceitação definidos no início do projeto, como:

- Registro completo e correto de pedidos, tanto online quanto offline
- Atualização automática do estoque após a conclusão de vendas
- Autenticação segura de usuários
- Interface acessível em diferentes dispositivos e navegadores
- Geração correta de relatórios administrativos

Todos os critérios foram cumpridos satisfatoriamente, com base em testes manuais, simulações e validações em campo. A solução foi aprovada pela empresa parceira e encontra-se em fase de implantação piloto, com perspectiva de expansão de uso para outras unidades e integração com serviços adicionais.

ADEQUAÇÃO PARA ÁREAS RURAIS

Reconhecendo as limitações da infraestrutura de internet em áreas rurais, o sistema contempla a inserção manual dos pedidos via módulo de atendimento, garantindo inclusão digital e acesso a serviços mesmo para clientes sem acesso online (Silva *et al*., 2021). Esta estratégia evita exclusão e aumenta o alcance da empresa.

RESULTADOS E DISCUSSÃO

A implantação do sistema desenvolvido visa aprimorar significativamente a gestão do fornecimento de diesel no contexto agrícola, especialmente em regiões rurais caracterizadas por limitações de infraestrutura tecnológica e conectividade restrita. A centralização dos pedidos, unificando canais digitais e tradicionais, alinha-se às melhores práticas para mitigação de erros e retrabalho, conforme indicado por Pereira et al. (2019), que enfatizam a importância da integração de canais para a otimização dos processos comerciais em ambientes rurais. Essa convergência operacional é fundamental para assegurar a coerência e rastreabilidade dos dados, aspectos críticos para a eficiência logística e para o controle de estoque, minimizando perdas e atrasos que afetam diretamente a cadeia produtiva agrícola.

Sob a perspectiva técnica, a adoção do Node.js para o backend proporciona alta escalabilidade e desempenho, graças ao seu modelo assíncrono e orientado a eventos, como salientado por Tilkov e Vinoski (2010). Essa arquitetura não bloqueante permite o atendimento eficiente de múltiplas requisições concorrentes, essencial para operações em tempo real, como atualização automática do estoque e confirmação rápida dos pedidos. Além disso, a robustez do ecossistema Node.js, complementado por ferramentas como o framework Express.js (Holmes, 2015), oferece suporte extensivo para desenvolvimento ágil e manutenção simplificada. Já o uso do React no frontend, aliado ao DOM virtual, garante uma interface responsiva e dinâmica, proporcionando melhor experiência ao usuário e facilitando o acesso via dispositivos móveis, fator crítico para o público rural com acesso limitado a infraestrutura de informática (Banks; Porcello, 2017; Abramov, 2019). A modularidade e a reutilização de componentes, características centrais do React, também contribuem para a manutenção eficiente e escalabilidade da interface, possibilitando adaptações futuras conforme as necessidades evoluam.

Do ponto de vista operacional, o sistema oferece uma solução híbrida que inclui módulo administrativo para registrar manualmente pedidos oriundos de clientes com acesso precário ou inexistente à internet. Essa abordagem promove inclusão digital, reduzindo a exclusão tecnológica e possibilitando que um maior número de agricultores seja atendido, conforme discutido por Oliveira e Costa (2020) e Silva *et al*. (2021). A inclusão dessas funcionalidades garante a democratização do serviço, assegurando que as barreiras tecnológicas não comprometam a continuidade do negócio ou a satisfação do cliente. Essa estratégia reflete uma consciência social importante no desenvolvimento tecnológico, reconhecendo as disparidades digitais e atuando para minimizá-las, o que é crucial para o desenvolvimento sustentável no setor agrícola brasileiro (IBGE, 2022).

A metodologia ágil adotada, baseada em ciclos iterativos de desenvolvimento e validação contínua, permitiu o alinhamento constante entre as demandas do cliente e as soluções técnicas implementadas, contribuindo para a robustez e aderência do sistema

às necessidades reais (Beck *et al*., 2001). A utilização de ferramentas de teste automatizado, como Postman, assegurou a qualidade da API, garantindo que endpoints cruciais para o funcionamento do sistema operem dentro dos parâmetros esperados, minimizando riscos de falhas em ambiente produtivo (Postman, 2024). Essa ênfase na qualidade e confiabilidade técnica é vital para sistemas que impactam diretamente na cadeia logística e financeira das empresas, onde indisponibilidades ou erros podem gerar prejuízos significativos.

Além dos ganhos imediatos em eficiência operacional e controle, a solução tem o potencial de impactar positivamente a tomada de decisões estratégicas na empresa, através da geração de relatórios e indicadores que possibilitam análises gerenciais fundamentadas em dados atualizados e consolidados (Fowler, 2003). Essa capacidade analítica favorece o planejamento logístico, a previsão de demanda e o aprimoramento contínuo dos processos, criando um diferencial competitivo para as empresas do setor. O uso da arquitetura modular facilita a integração futura com outras tecnologias, como sistemas de georreferenciamento para otimização de rotas e ferramentas de inteligência artificial para previsão de demanda, ampliando o escopo de melhorias possíveis.

Por fim, destaca-se a relevância social e econômica do sistema, ao fomentar a transformação digital em um setor fundamental para a economia nacional, contribuindo para o desenvolvimento sustentável da agricultura brasileira. A proposta demonstra que a adoção de tecnologias consolidadas, adaptadas à realidade local e aliadas a metodologias modernas de desenvolvimento, constitui uma estratégia eficaz para enfrentar desafios estruturais e promover a inclusão digital em contextos rurais (IBGE, 2022; Pereira *et al*., 2019). A replicabilidade dessa solução para outros segmentos rurais, que compartilham desafios semelhantes em relação à conectividade e infraestrutura, aponta para um potencial de impacto ampliado, fortalecendo a competitividade regional e promovendo a equidade tecnológica.

CONCLUSÕES

A proposta apresentada neste trabalho buscou responder a uma demanda latente no setor de fornecimento de combustíveis para a agricultura: a necessidade de um sistema informatizado que contemple tanto a modernização dos processos internos quanto a acessibilidade para usuários situados em áreas com infraestrutura tecnológica limitada. Nesse sentido, o desenvolvimento de um sistema híbrido composto por uma vitrine digital e um módulo administrativo para atendimento tradicional demonstrou ser uma estratégia tecnicamente viável, funcionalmente eficiente e socialmente inclusiva.

A partir da utilização de tecnologias consolidadas e amplamente adotadas no desenvolvimento web – como React para o frontend, Node.js para o backend e MySQL como sistema de gerenciamento de banco de dados – foi possível construir uma arquitetura escalável, modular e segura, capaz de atender às exigências operacionais de empresas de pequeno e médio porte atuantes na zona rural. O uso da metodologia ágil permitiu uma abordagem iterativa, com entregas incrementais e constante validação junto aos stakeholders, o que contribuiu significativamente para a adequação da solução às necessidades reais do negócio.

Do ponto de vista técnico, os principais resultados esperados com a implementação da solução incluem a redução de erros operacionais, o aumento da rastreabilidade dos pedidos, a atualização em tempo real do estoque e o fortalecimento da comunicação entre os diferentes canais de atendimento. Tais benefícios, além de promoverem maior eficiência na gestão empresarial, contribuem diretamente para o aumento da satisfação do cliente e para a melhoria da qualidade do serviço prestado.

Adicionalmente, a proposta se destaca por reconhecer e enfrentar as limitações estruturais presentes no meio rural brasileiro, notadamente no que se refere ao acesso à internet de qualidade. Ao permitir o registro manual de pedidos por parte dos operadores da empresa, o sistema assegura a inclusão de clientes que não dispõem de conectividade estável, contribuindo para a democratização do acesso a tecnologias digitais e para a mitigação das desigualdades tecnológicas no setor agrícola.

A integração entre os diferentes módulos do sistema, associada à possibilidade de geração de relatórios e indicadores gerenciais, fortalece a tomada de decisão baseada em dados e amplia a capacidade de planejamento logístico da empresa. Nesse aspecto, a solução não apenas automatiza processos, mas também agrega valor estratégico à organização, posicionando-a de forma mais competitiva no mercado.

Em perspectiva futura, a plataforma desenvolvida poderá ser expandida com novas funcionalidades, como integração com sistemas de georreferenciamento para planejamento de rotas de entrega, automação fiscal para emissão de notas fiscais eletrônicas, e mecanismos de análise preditiva baseados em inteligência artificial para antecipação da demanda. Tais aprimoramentos poderão consolidar o sistema como uma ferramenta completa de gestão integrada voltada ao setor de distribuição de diesel no meio rural, com potencial de aplicação em outras cadeias produtivas que compartilhem características logísticas semelhantes.

Conclui-se, portanto, que o desenvolvimento e a adoção de soluções tecnológicas contextualizadas à

realidade do público-alvo representam um caminho promissor para promover a transformação digital no campo. A proposta aqui apresentada evidencia que a tecnologia, quando aplicada de maneira estratégica, acessível e sustentável, pode atuar como um vetor de inovação, inclusão e desenvolvimento regional no setor agrícola brasileiro.

REFERÊNCIAS

DAHL ABRAMOV, Dan. **Introducing Hooks**. 2019. Disponível em: https://reactjs.org/docs/hooks-intro.html. Acesso em: 16 maio 2025.

BANKS, Eve; PORCELLO, Ethan. **Learning React: Functional Web Development with React and Redux**. 2. ed. O'Reilly Media, 2017.

CHÓI, Jinho. **React Ecosystem Overview**. 2020. Disponível em: https://medium.com/@jinhochoi/react-ecosystem-overview-2c9a2ee9db34. Acesso em: 16 maio 2025.

CORONEL, Carlos; MORRIS, Steven. **Database Systems: Design, Implementation, & Management**. 12. ed. Boston: Cengage Learning, 2016.

DAHL, Ryan**. Node.js. 2009**. Disponível em: https://nodejs.org/en/about/. Acesso em: 16 maio 2025.

DUBOIS, Paul. **MySQL**. 5. ed. Berkeley: Peachpit Press, 2008.

FOWLER, Martin. **Patterns of Enterprise Application Architecture**. Addison-Wesley, 2003.

HOLMES, Ethan. **Express.js Guide**: The Comprehensive Book on Express.js. Shelter Island: Leanpub, 2015.

IBGE. **Produção Agrícola Municipal.** Instituto Brasileiro de Geografia e Estatística, 2022. Disponível em: https://www.ibge.gov.br/. Acesso em: 15 maio 2025.

OLIVEIRA, R.; COSTA, M. Conectividade no campo: desafios para a inclusão digital rural. **Revista de Tecnologia Rural**, v. 14, n. 2, p. 123-134, 2020.

PEREIRA, J*. et al*. Gestão integrada de canais de vendas em empresas rurais. **Congresso Brasileiro de Sistemas de Informação**, 2019.

POSTMAN. **Postman Documentation**. 2024. Disponível em: https://learning.postman.com/docs/getting-started/introduction/. Acesso em: 16 maio 2025.

SILVA, A. M. et al. Inclusão digital em regiões rurais: estratégias para ampliação do acesso**. Journal of Rural Studies**, v. 40, p. 56-65, 2021.

SOMMERVILLE, Ian. **Engenharia de Software**. 9. ed. Pearson, 2011.

TILKOV, Stefan; VINOSKI, Steve. Node.js**: Using JavaScript to Build High-Performance Network Programs**. IEEE Internet Computing, v. 14, n. 6, p. 80-83, nov./dez. 2010. DOI: 10.1109/MIC.2010.145..