



FACULDADE DE TECNOLOGIA DO IPIRANGA
CURSO DE ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

CRISTIAN DANIEL ESTEVEZ CACHI
EMILY VICTORIA DA SILVA CHAGAS
ISABEL SANTOS MARQUES
JONAS GODOY DIAMANTINO
THIAGO DOS SANTOS SANTANA
VINICIUS BEZERRA MEDEIROS

**SECONDBITE: SISTEMA WEB MOBILE-FIRST PARA COMERCIALIZAÇÃO DE
ALIMENTOS PERECÍVEIS**

SÃO PAULO

2026

CRISTIAN DANIEL ESTEVEZ CACHI

EMILY VICTORIA DA SILVA CHAGAS

ISABEL SANTOS MARQUES

JONAS GODOY DIAMANTINO

THIAGO DOS SANTOS SANTANA

VINICIUS BEZERRA MEDEIROS

**SECONDBITE: SISTEMA WEB MOBILE-FIRST PARA COMERCIALIZAÇÃO DE
ALIMENTOS PERECÍVEIS**

Trabalho de conclusão de curso
apresentado à faculdade de Tecnologia do
Ipiranga, como requisito parcial para
obtenção do grau de Tecnólogo em
Análise e Desenvolvimento de Sistemas

SÃO PAULO

2026

AGRADECIMENTOS

Agradecemos às nossas famílias, que sempre acreditaram em nosso potencial. Aos nossos professores, somos gratos pelo conhecimento que compartilharam conosco, pela dedicação e orientação ao longo dessa jornada acadêmica. E a todos que, de alguma forma, contribuíram para que este trabalho se concretizasse.

RESUMO

O desperdício de alimentos representa um desafio crítico nas dinâmicas de comercialização contemporâneas, gerando prejuízos econômicos diretos aos comerciantes e impactando negativamente o meio ambiente e a gestão de resíduos urbanos. Diante desse cenário, o presente trabalho propõe o desenvolvimento do SecondBite, um sistema web responsivo com abordagem *mobile-first* inspirada nos princípios da economia circular, que conecta consumidores e feirantes para a comercialização de alimentos próximos ao vencimento ou fora do padrão estético, mas ainda próprios para consumo. A solução atua especificamente no ecossistema de feiras livres, promovendo o aproveitamento de alimentos que seriam descartados e incentivando o consumo consciente por meio da tecnologia. O objetivo é fornecer uma plataforma acessível que mitigue as perdas financeiras dos feirantes e reduza ativamente o descarte de matéria orgânica nas cidades, integrando tecnologia e sustentabilidade.

Palavras-chave: Desperdício de alimentos; feiras livres; economia circular; sustentabilidade; desenvolvimento web.

LISTA DE FIGURAS

- Figura 1 - Diagrama de Modelagem do Banco de Dados.
- Figura 2 - Diagrama de Casos de Uso.
- Figura 3 - Diagrama de Classes.
- Figura 4 - Diagrama de Sequência.
- Figura 5 - Diagrama de Arquitetura.
- Figura 6 - Protótipo do Sistema - Login e Registro.
- Figura 7 - Protótipo do Sistema - Tela de Início Perfil Consumidor.
- Figura 8 - Protótipo do Sistema - Busca e Filtro de Produtos.
- Figura 9 - Protótipo do Sistema - Detalhes do Produto.
- Figura 10 - Protótipo do Sistema - Vitrine do Feirante.
- Figura 11 - Protótipo do Sistema - Avaliações de Feirante.
- Figura 12 - Protótipo do Sistema - Mapa de Feirantes.
- Figura 13 - Protótipo do Sistema - Carrinho de Compras.
- Figura 14 - Protótipo do Sistema - Meus Pedidos.
- Figura 15 - Protótipo do Sistema - Detalhes do Pedido Perfil Consumidor.
- Figura 16 - Protótipo do Sistema – Tela de Início Perfil Feirante.
- Figura 17 - Protótipo do Sistema – Modo Xepa.
- Figura 18 - Protótipo do Sistema – Clone de Produtos.
- Figura 19 - Protótipo do Sistema – Gestão de Estoque e Produtos.
- Figura 20 - Protótipo do Sistema – Configuração de Desconto Automático.
- Figura 21 - Protótipo do Sistema – Sugestão de Desconto Inteligente.
- Figura 22 - Protótipo do Sistema – Gestão de Pedidos.
- Figura 23 - Protótipo do Sistema – Consultar Avaliações.
- Figura 24 - Protótipo do Sistema – Relatório da Banca.
- Figura 25 - Gráfico de aceitação de alimentos próximos ao vencimento.
- Figura 26 - Gráfico de adesão à plataforma SecondBite.

LISTA DE TABELAS

Tabela 1 - CRUD do Consumidor e Login

Tabela 2 - Consulta de Produtos (Busca e Filtros)

Tabela 3 - Visualização e Avaliação de Feirante

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CRUD	Create, Read, Update, Delete
HTTP	Hypertext Transfer Protocol
JPA	Java Persistence API
JSON	JavaScript Object Notation
JWT	JSON Web Token
MER	Modelo Entidade-Relacionamento
MVP	Minimum Viable Product
MVC	Model-View-Controller
PIX	Pagamentos Instantâneos
REST	Representational State Transfer
RF	Requisito Funcional
RNF	Requisito Não-Funcional
SGBD	Sistema de Gerenciamento de Banco de Dados
SPA	Single Page Application

SUMÁRIO

1 INTRODUÇÃO.....	9
1.1 Objetivo geral.....	10
1.2 Objetivos específicos.....	10
1.3 Justificativa.....	11
2 PROJETO TÉCNICO.....	13
2.1 Regras de negócios.....	13
2.2 Requisitos funcionais.....	15
2.3 Requisitos não funcionais.....	18
2.4 Modelagem do banco de dados.....	20
2.5 Diagrama de casos de uso.....	21
2.6 Diagrama de classes.....	22
2.7 Diagrama de sequência.....	23
2.8 Diagrama de arquitetura.....	24
3 IMPLEMENTAÇÃO.....	27
3.1 Tecnologias e ferramentas utilizadas.....	27
3.2 Descrição e estrutura da aplicação.....	27
3.2.1 Padrões de projeto e camadas.....	28
3.2.2 Módulos principais do sistema.....	28
3.3 Práticas de desenvolvimento e versionamento de código.....	29
3.4 Implantação.....	30
3.5 Roteiro de testes.....	30
3.6 Telas do sistema.....	33
3.6.1 Telas do perfil do consumidor.....	34
3.6.2 Telas do perfil de feirante.....	43
4 RESULTADOS OBTIDOS.....	53
4.1 Perspectivas futuras.....	55
REFERÊNCIAS.....	57

1 INTRODUÇÃO

Nas dinâmicas contemporâneas de produção, distribuição e consumo, o desperdício de alimentos e a insegurança alimentar destacam-se como desafios globais e locais significativos. O Programa das Nações Unidas para o Meio Ambiente (PNUMA, 2024) aponta que mais de 1 bilhão de toneladas de alimentos são desperdiçadas globalmente a cada ano, o que equivale a cerca de 132 quilos per capita. No Brasil, o cenário é igualmente preocupante, com estimativas indicando que cerca de 27 milhões de toneladas de alimentos são descartadas anualmente, impactando diretamente toda a cadeia de suprimentos nacional.

Nos grandes centros urbanos, como a região metropolitana de São Paulo, o problema do desperdício torna-se ainda mais evidente no segmento de comercialização e abastecimento, com especial destaque para as feiras livres. Estudos recentes apontam que as feiras livres são importantes pontos de geração de resíduos orgânicos. Nesses locais, uma parcela significativa dos alimentos descartados ainda possui valor nutricional e está própria para consumo, mas acaba sendo rejeitada devido a imperfeições estéticas, excesso de oferta ou danos gerados no transporte e armazenamento (UNESP, 2022). As principais causas desse desperdício estão, portanto, associadas à logística inadequada, ao manuseio brusco e à exposição prolongada dos produtos nas barracas.

Nesse cenário, diversos atores sociais são impactados negativamente. Os feirantes enfrentam prejuízos financeiros diretos com o descarte das mercadorias não vendidas ao fim do dia. Simultaneamente, consumidores lidam com o aumento do custo de vida e barreiras para acessar alimentos de qualidade. Ademais, o volume expressivo de resíduos orgânicos despejados nas vias públicas sobrecarrega a gestão urbana, contribuindo para problemas ambientais e de saúde pública, como enchentes e proliferação de pragas urbanas. Para mitigar esses efeitos, é essencial a adoção de ações fundamentadas na economia circular, voltadas para a redução do desperdício na fonte e a integração tecnológica entre as pontas do consumo (ARAÚJO; CAPELLA; SCHNEIDER, 2024).

Diante dessa problemática, este trabalho propõe o desenvolvimento de uma plataforma web colaborativa, com interface otimizada para o uso móvel (*mobile-first*), denominada SecondBite. O sistema atua como um *marketplace* voltado especificamente para conectar feirantes e consumidores locais, visando otimizar a comercialização de alimentos in natura excedentes e próximos à perda de frescor.

O SecondBite atua como uma ferramenta tecnológica que permite aos feirantes registrarem de forma ágil os produtos que necessitam de escoamento rápido na barraca, através de descontos. Integrando serviços de geolocalização, o sistema aproxima esses comerciantes do público consumidor da sua região. Dessa forma, a aplicação deseja promover o engajamento social, incentivar o consumo sustentável e auxiliar cidadãos na busca por alimentos mais acessíveis, combatendo ativamente o desperdício nas feiras livres.

1.1 Objetivo Geral

Desenvolver uma aplicação web responsiva, com foco na experiência *mobile*, denominada SecondBite, que atue como um *marketplace* para conectar feirantes e consumidores locais, com o intuito de facilitar a comercialização de alimentos in natura excedentes e reduzir o desperdício de alimentos, fomentando práticas de economia circular.

1.2 Objetivos Específicos

Para que o objetivo geral fosse alcançado, foram definidos os seguintes objetivos específicos:

- Implementar módulos de cadastro e autenticação de usuários, permitindo a separação clara entre os perfis de consumidores e feirantes.
- Criar uma interface responsiva (*mobile-first*) para o gerenciamento ágil de produtos em ambiente de feira.
- Desenvolver recursos de *marketplace* que permitam aos feirantes registrarem, gerenciarem e ofertarem produtos, priorizando vendas rápidas.

- Integrar geolocalização para conectar consumidores às barracas de feira da região.
- Construir um painel de controle (*dashboard*) para os feirantes com gerenciamento do estoque, acompanhamento de pedidos e visualização do impacto gerado pela redução do desperdício.

1.3. Justificativa

O desperdício de alimentos na região metropolitana de São Paulo atinge níveis alarmantes, impactando negativamente tanto a economia local quanto o meio ambiente. Segundo levantamento recente da Aliança pela Alimentação Adequada e Saudável (2024), são desperdiçadas cerca de 33 mil toneladas de alimentos por ano apenas nas mais de 900 feiras livres da capital paulista. Estima-se que cerca de metade de todo o resíduo gerado nestes locais seja composto por alimentos que ainda poderiam ser consumidos.

Além dos problemas de manuseio e logística, estudos indicam que as etapas finais da cadeia de abastecimento, como as feiras, são pontos críticos de perda (ARAÚJO; CAPELLA; SCHNEIDER, 2024). Um agravante para essa situação é o fator comportamental: muitos consumidores rejeitam produtos que apresentam pequenas imperfeições estéticas, mesmo que estejam nutricionalmente perfeitos e próprios para o consumo (UNESP, 2022).

Pensando nisso, o trabalho buscou oferecer uma ferramenta tecnológica que combata essa problemática de forma direta e local. O sistema SecondBite surge como uma plataforma voltada para reinserir esses alimentos na cadeia de consumo antes que se tornem lixo, conectando os feirantes, que precisam escoar o estoque excedente ou "fora do padrão estético", com os consumidores da região, que buscam alimentos frescos a preços mais acessíveis.

O grande diferencial da plataforma reside em sua capacidade de otimizar a conexão entre comerciantes e moradores da região através de recursos de geolocalização e um sistema de descontos. Ao incentivar a venda rápida de itens próximos ao fim da vida útil na feira, o sistema diminui o prejuízo financeiro dos

comerciantes e reduz drasticamente o volume de resíduos orgânicos descartados nas ruas, aliviando a gestão de zeladoria urbana.

Sendo assim, este trabalho pretende oferecer uma resposta prática, escalável e sustentável para um problema real que afeta a segurança alimentar e o ecossistema das cidades todos os dias, promovendo a economia circular de forma aplicada.

2 PROJETO TÉCNICO

Este capítulo descreve os requisitos funcionais e não-funcionais do software, bem como as regras de negócio, modelagem do banco de dados e os diagramas.

2.1 Regras de Negócio

Segundo Pressman e Maxim (2021), as regras de negócio representam as políticas, fluxos e restrições que governam o comportamento de uma aplicação em resposta a eventos específicos do domínio. Elas são essenciais para garantir que o software cumpra os objetivos organizacionais e os requisitos operacionais para os quais foi projetado.

A seguir, são identificadas as regras de negócio que norteiam o funcionamento do sistema SecondBite:

- **[RN001] Exclusividade de Feirante no Carrinho:** Para que um pedido seja finalizado (*checkout*), o sistema deve exigir que todos os produtos presentes no carrinho do consumidor pertençam a um único feirante.
- **[RN002] Bloqueio de *Checkout* Vazio:** É estritamente bloqueada a inicialização do processo de *checkout* e a criação de um pedido caso o carrinho do consumidor não possua nenhum item.
- **[RN003] Garantia e Trava de Estoque no Carrinho:** O sistema deve impedir a adição de um novo produto ao carrinho, ou o aumento de sua quantidade, caso o valor total desejado ultrapasse a quantidade atualmente disponível no estoque do feirante.
- **[RN004] Remoção Automática por Quantidade Zerada:** Se o consumidor atualizar a quantidade de um produto no carrinho para zero (ou um valor negativo), o item deve ser removido automaticamente do carrinho.
- **[RN005] Fluxo Rígido de Aceite de Pedido:** Apenas o usuário com perfil de Feirante pode aceitar um pedido. O sistema só permite esta transição se o pedido estiver com o status de pendente de aceite.
- **[RN006] Restrição de Cancelamento pelo Consumidor:** Um consumidor só tem permissão para cancelar seu pedido se o status ainda for de aguardo de confirmação do pedido por parte do feirante. Se o feirante já o aceitou ou o

pedido já está pronto para coleta, o cancelamento por parte do consumidor é bloqueado.

- **[RN007] Restituição Automática de Estoque:** Sempre que um pedido for cancelado (seja pelo feirante ou pelo consumidor), as quantidades de todos os itens daquele pedido devem ser devolvidas e somadas imediatamente ao estoque ativo do produto.
- **[RN008] Fluxo Rígido de Conclusão de Pedido:** Um pedido só pode ser finalizado caso o seu status anterior seja, obrigatoriamente, "Pronto para Coleta".
- **[RN009] Geração Dinâmica de Código de Entrega:** Ao processar o *checkout*, o sistema gera automaticamente um código de segurança de 4 dígitos para o pedido, utilizando os últimos 4 dígitos do telefone cadastrado do consumidor.
- **[RN010] Autenticação de Retirada (Baixa de Pedido):** Para dar baixa na entrega e finalizar o pedido presencialmente, o feirante deve obrigatoriamente informar o "Código de Entrega" idêntico ao que foi gerado no aplicativo do consumidor.
- **[RN011] Restrição de Propriedade de Produto:** Apenas o feirante titular, que cadastrou um determinado produto, deve possuir autorização do sistema para alterá-lo, cloná-lo, excluí-lo ou manipular seu estoque.
- **[RN012] Estoque Não-Negativo:** A quantidade em estoque de um produto jamais pode assumir um valor negativo. Operações que resultam em valores menores que zero são barradas ou ajustadas automaticamente para zero.
- **[RN013] Sobrescrita de Promoção Xepa:** Ao aplicar o "modo xepa" a um grupo de produtos, o sistema deve desativar automaticamente a regra de "Desconto Automático" e impor o novo percentual de desconto manual definido pelo feirante.
- **[RN014] Unicidade de Avaliação por Feirante:** Um consumidor pode ter o direito de registrar apenas uma avaliação para cada feirante. O sistema deve bloquear tentativas de avaliações duplicadas.
- **[RN015] Unicidade de Identidade (E-mail):** O sistema deve impedir o cadastro de novas contas (sejam consumidores ou feirantes) utilizando um

endereço de e-mail que já possua registro na base de dados, garantindo a identidade única no login.

2.2 Requisitos Funcionais

RF001 - Cadastro Completo de Produtos

Descrição: O sistema deve permitir que o feirante cadastre novos produtos, informando nome, categoria, preço original, quantidade em estoque e fazendo o upload de múltiplas imagens.

RF002 - Configuração Híbrida de Descontos

Descrição: Durante a criação/edição do produto, o sistema deve permitir ao feirante definir se o desconto aplicado será "Automático" (calculado pela plataforma) ou "Manual" (onde ele dita a porcentagem exata).

RF003 - Modo Xepa (Flash Sale)

Descrição: O sistema deve fornecer uma interface dedicada para o “Modo Xepa”, para que o feirante selecione múltiplos produtos em lote e aplique uma liquidação imediata, sobrescrevendo configurações de desconto anteriores.

RF004 - Sugestões Inteligentes de Desconto

Descrição: O sistema deve exibir Sugestões de Desconto no painel do feirante, auxiliando na precificação dinâmica para evitar o desperdício de itens próximos ao fim da feira.

RF005 - Clonagem Rápida de Produtos

Descrição: O sistema deve oferecer um botão de "Clonar" para que o feirante duplique o cadastro de um produto existente, facilitando a adição de variações rápidas de mercadorias.

RF006 - Atualização Dinâmica de Estoque

Descrição: O sistema deve permitir que o feirante adicione ou reduza rapidamente a quantidade de itens no estoque através de botões rápidos na sua lista de produtos.

RF007 - Mapa de Feirantes

Descrição: O sistema deve exibir uma interface de mapa interativo para o consumidor, com "Pins" mostrando a localização exata das barracas e feirantes ativos próximos.

RF008 - Seleção de Localização

Descrição: O sistema deve permitir que o feirante cadastre ou atualize a localização física da sua barraca arrastando um marcador (Pin) sobre um mapa interativo durante a configuração do perfil.

RF009 - Vitrine do Feirante

Descrição: O sistema deve exibir uma página pública de perfil para cada feirante, listando todos os seus produtos disponíveis, sua nota média e informações de contato.

RF010 - Busca e Filtro de Catálogo

Descrição: O sistema deve permitir que o consumidor pesquise produtos por nome e filtre resultados por Categorias específicas (ex: Frutas, Legumes, Verduras).

RF011 - Gestão Dinâmica do Carrinho

Descrição: O sistema deve permitir que o consumidor adicione itens, altere as quantidades ou remova completamente os produtos do carrinho, calculando o subtotal em tempo real.

RF012 - Validação Contínua de Estoque

Descrição: O sistema deve verificar o estoque disponível sempre que o consumidor tentar adicionar ou aumentar a quantidade de um item no carrinho, impedindo a ação se não houver estoque suficiente.

RF013 - Efetivação de Pedido (*Checkout*)

Descrição: O sistema deve converter o carrinho ativo de um consumidor em um Pedido com status de pendente de confirmação, vinculando-o ao feirante responsável.

RF014 - Geração de Código de Retirada

Descrição: O sistema deve gerar e exibir um código numérico de 4 dígitos (baseado no telefone do consumidor) logo após o *checkout*, servindo como "senha" de segurança para a retirada do pedido na barraca.

RF015 - Painel de Pedidos (Feirante):

Descrição: O sistema deve permitir que o feirante visualize e altere o status dos pedidos recebidos, desde a separação da sacola até o status de "Pronto para Coleta".

RF016 - Conclusão Segura de Pedido

Descrição: O sistema deve exigir que o feirante digite o "Código de Entrega" correto fornecido pelo consumidor presencialmente para dar baixa e finalizar um pedido.

RF017 - Cancelamento de Pedidos

Descrição: O sistema deve permitir o cancelamento do pedido e deve retornar imediatamente os itens cancelados de volta ao estoque visível do feirante.

RF018 - Acompanhamento do Consumidor

Descrição: O sistema deve fornecer uma aba/tab "Meus Pedidos" onde o consumidor pode acompanhar se a sua sacola já está sendo separada ou se já está pronta para retirada.

RF019 - Painel de Impacto Ambiental

Descrição: O sistema deve possuir um *dashboard* na tela inicial do feirante exibindo métricas de desempenho e calculando o impacto positivo gerado pelas vendas da xepa (ex: quantidade de comida salva do desperdício).

RF020 - Sistema de Avaliações e Reputação

Descrição: O sistema deve permitir que o consumidor envie uma nota de avaliação para o feirante, compondo a nota média pública exibida na vitrine do vendedor.

RF021 - Painel Analítico e Exportação de Relatórios

Descrição: O sistema deve fornecer uma área dedicada a relatórios gerenciais para o feirante, apresentando o histórico de vendas, evolução do faturamento, ticket

médio por item e estimativa de impacto ecológico (CO₂ evitado) por períodos selecionáveis (ex: 7 dias ou mês). Além disso, deve permitir a exportação detalhada dessas vendas em um arquivo de planilha no formato CSV (*Comma-Separated Values*).

2.3 Requisitos não-funcionais

RNF001 - Autenticação e Autorização (Segurança)

Descrição: O sistema deve gerenciar a sessão dos usuários (feirantes e consumidores) de forma "sem estado" (*stateless*), utilizando tokens JWT (*JSON Web Token*) trafegados em rotas protegidas, garantindo escalabilidade e segurança na API.

RNF002 - Criptografia de Dados Sensíveis (Segurança)

Descrição: O sistema não deve armazenar senhas de usuários em texto plano no banco de dados. Todas as credenciais devem ser criptografadas via algoritmos de hash seguros (ex: BCrypt, configurado no *Spring Security*) antes da persistência.

RNF003 - Feedback Visual Imediato (Usabilidade)

Descrição: O sistema deve fornecer *feedback* visual instantâneo para todas as ações críticas do usuário (sucesso, erro, avisos) por meio de notificações não-intrusivas na tela (utilizando a biblioteca *Sonner* no *frontend*), garantindo que o usuário saiba o resultado de sua operação.

RNF004 - Tratamento Padronizado de Exceções (Confiabilidade / Resiliência)

Descrição: O *backend* deve capturar todos os erros inesperados ou de validação através de um manipulador global e retornar para o *frontend* mensagens de erro padronizadas e amigáveis, impedindo o travamento da aplicação cliente.

RNF005 - Paginação de Volumes de Dados (Desempenho)

Descrição: O sistema deve implementar paginação obrigatória nas consultas de listagem que podem crescer indefinidamente, como o catálogo de produtos. Dessa forma limita o consumo de banda e memória.

RNF006 - Comunicação via Padrão RESTful (Interoperabilidade)

Descrição: Toda a comunicação entre o aplicativo cliente (*frontend*) e o servidor (*backend*) deve ocorrer por meio do protocolo HTTP, utilizando requisições REST e trocando dados estritamente no formato JSON.

RNF007 - Gerenciamento Inteligente de Cache Client-Side (Desempenho)

Descrição: O *frontend* deve utilizar mecanismos de cache e sincronização assíncrona (via *TanStack Query*) para armazenar temporariamente dados buscados da API (como informações de perfil e produtos), reduzindo o número de requisições redundantes ao servidor.

RNF008 - Responsividade da Interface de Usuário (Usabilidade)

Descrição: A interface web do sistema deve ser fluida e responsiva, principalmente em dispositivos móveis, adaptando-se automaticamente aos mais diversos tamanhos de tela (*smartphones*, *tablets* e monitores *desktop*).

RNF009 - Tipagem Estrita e Prevenção de Falhas (Manutenibilidade)

Descrição: O código fonte deve empregar tipagem estática rigorosa em ambas as frentes da aplicação (TypeScript no *frontend* e Java no *backend*), garantindo a detecção de erros de contratos de dados em tempo de compilação, antes de ir para produção.

RNF010 - Compatibilidade Cross-Browser (Interoperabilidade / Compatibilidade)

Descrição: A interface de usuário deve ser plenamente funcional e visualmente consistente nos principais e mais atualizados navegadores modernos do mercado (Google Chrome, Mozilla Firefox, Microsoft Edge e Apple Safari).

RNF011 - Arquitetura de Código em Camadas (Manutenibilidade)

Descrição: O *backend* deve respeitar uma arquitetura de camadas estrita (*Controllers*, *Services* e *Repositories*), garantindo uma clara separação de responsabilidades para facilitar testes automatizados e a introdução de novas funcionalidades.

RNF012 - Renderização Otimizada de Mapas Geográficos (Desempenho / Usabilidade)

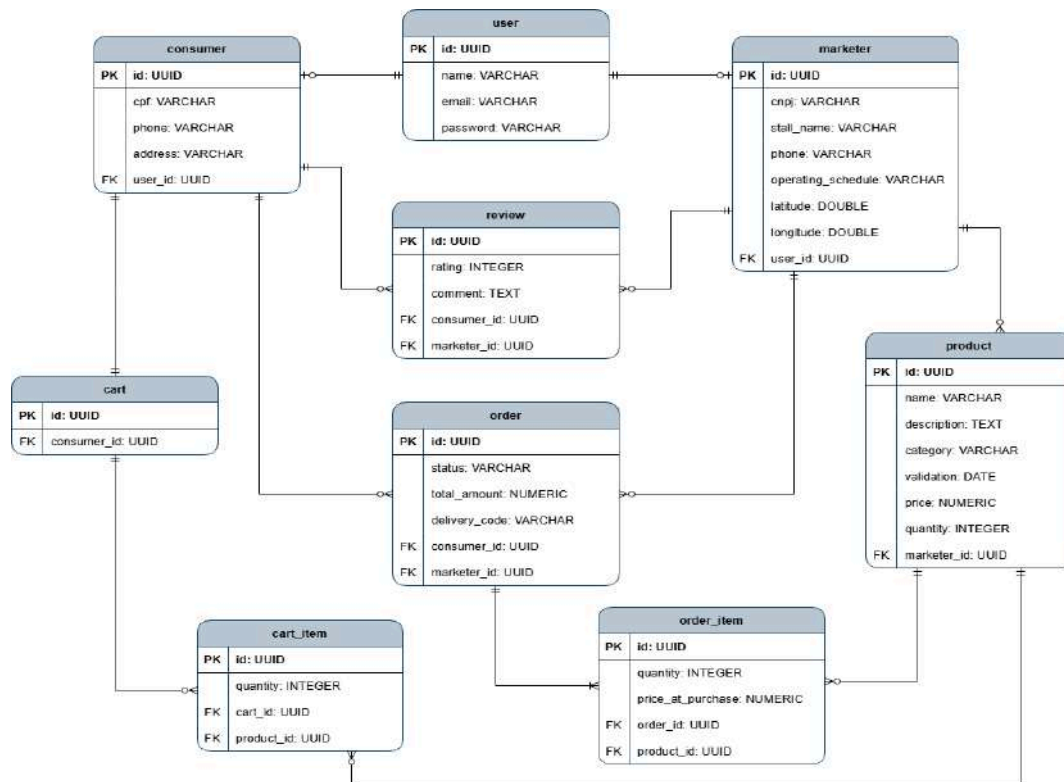
Descrição: A renderização dos componentes de mapa interativo (rastreamento de feirantes) no navegador do consumidor não deve bloquear a *thread* principal (UI), utilizando bibliotecas de mapeamento vetorial aceleradas por *hardware* (MapLibre GL) para garantir uma navegação suave ao arrastar ou dar zoom.

2.4 Modelagem do Banco de Dados

Para Elmasri e Navathe (2011), o modelo conceitual é uma descrição de alto nível da estrutura do banco de dados, sendo independente de detalhes de armazenamento físico ou implementação. Este modelo permite que projetistas e usuários finais visualizem as entidades fundamentais e seus respectivos relacionamentos de forma clara e intuitiva.

Na Figura 1, é apresentado o Modelo Entidade-Relacionamento (MER) do sistema:

Figura 1 - Diagrama de Modelagem do Banco de Dados.



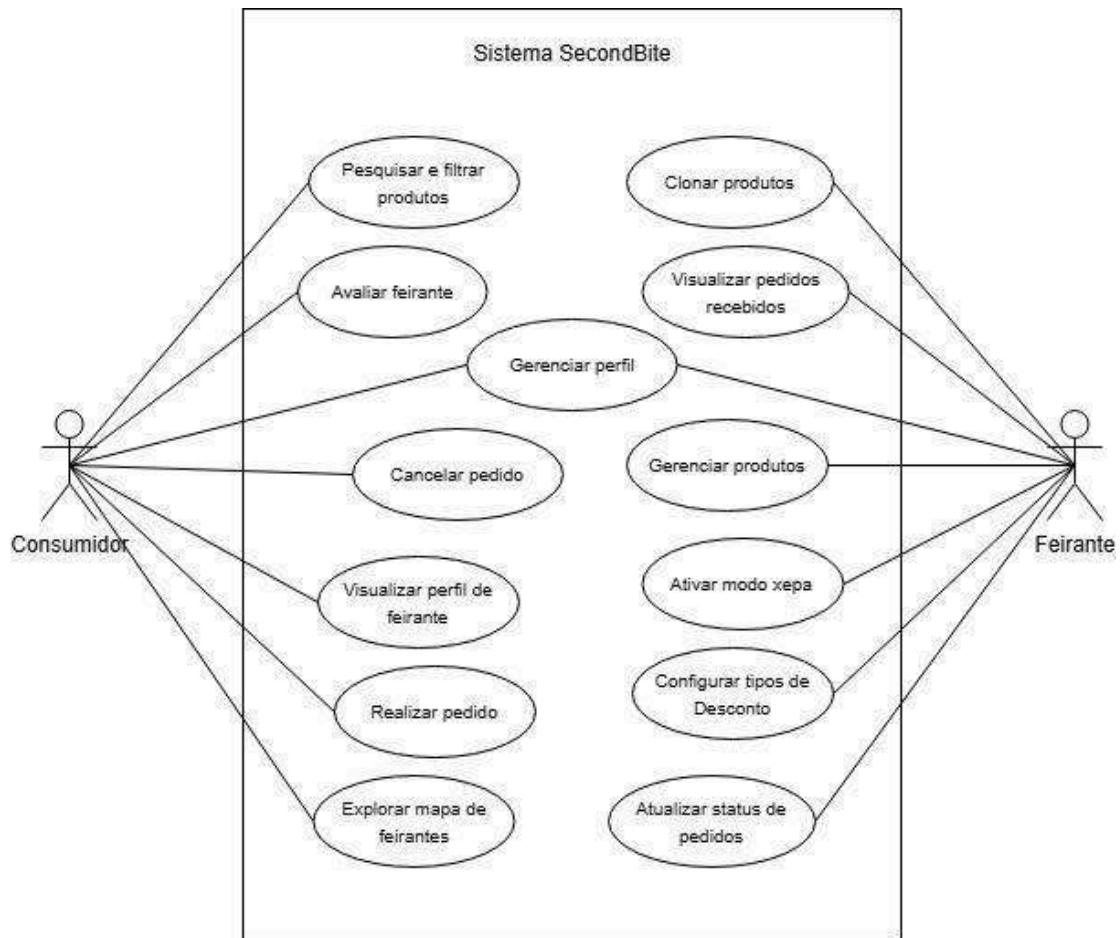
Fonte: Autores (2026).

2.5 Diagrama de Casos de Uso

Conforme definido por Booch, Rumbaugh e Jacobson (2006), o diagrama de casos de uso descreve o comportamento do sistema sob a ótica do usuário final. Ele detalha as sequências de ações que o software executa para fornecer um resultado de valor observável a um ator específico, facilitando a compreensão das interações externas.

A figura 2 abaixo ilustra as principais funcionalidades e interações dos usuários com o SecondBite:

Figura 2 - Diagrama de Casos de Uso.



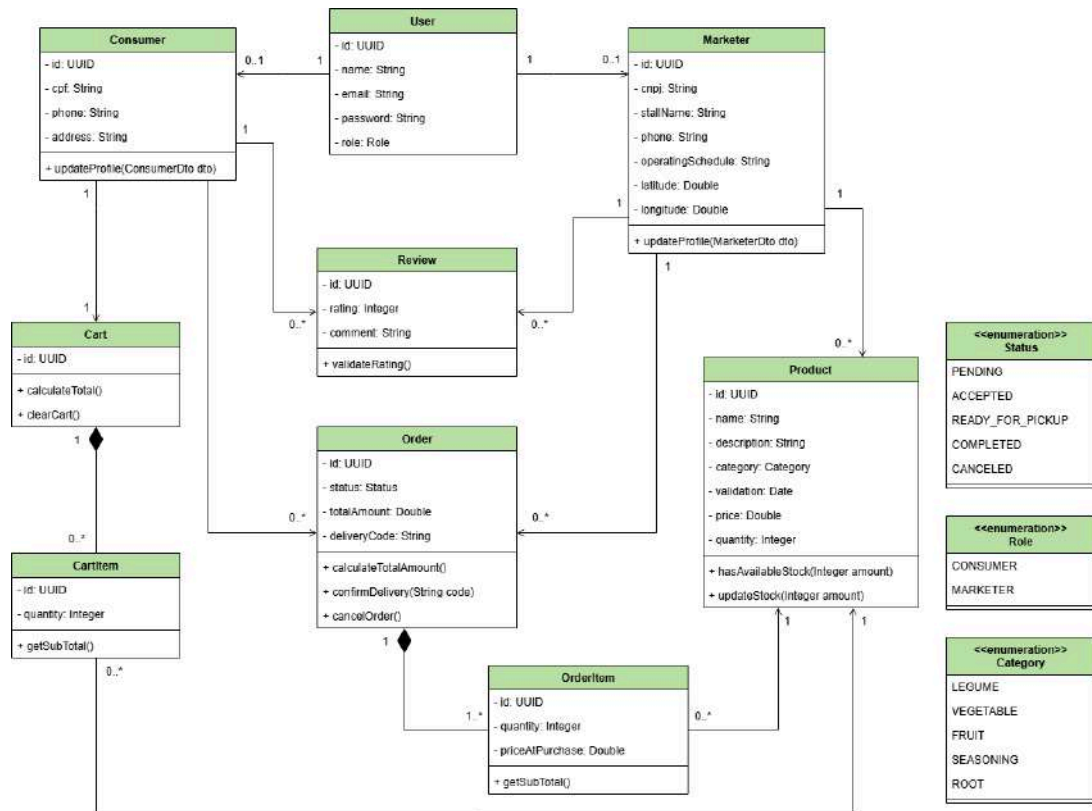
Fonte: Autores (2026).

2.6 Diagrama de Classes

De acordo com Larman (2007), o diagrama de classes constitui a ferramenta central da modelagem orientada a objetos, sendo utilizado para ilustrar a estrutura estática da aplicação. Nele, são representadas as classes do software, seus atributos, métodos e as associações que definem a lógica estrutural do sistema.

A estrutura lógica das classes do projeto está representada na Figura 3:

Figura 3 - Diagrama de Classes.



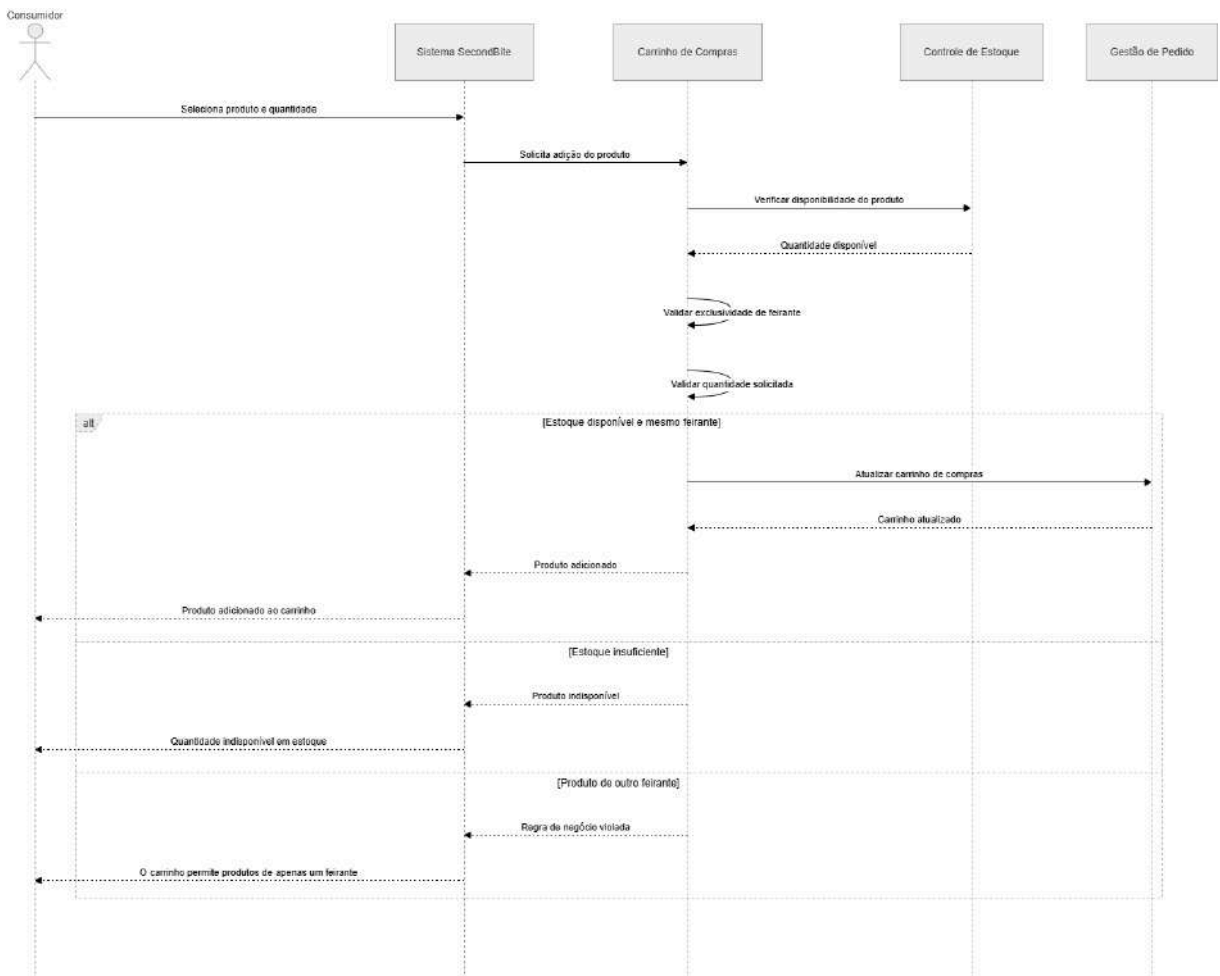
Fonte: Autores (2026).

2.7 Diagrama de Sequência

O diagrama de sequência, segundo Pressman (2016), foca na dimensão temporal das interações entre os objetos. Ele representa como os componentes colaboram entre si através de mensagens trocadas em uma ordem cronológica específica para realizar uma determinada funcionalidade ou caso de uso do sistema.

Na figura 4 abaixo, detalha-se a dinâmica de interação para os fluxos principais do sistema:

Figura 4 - Diagrama de Sequência.



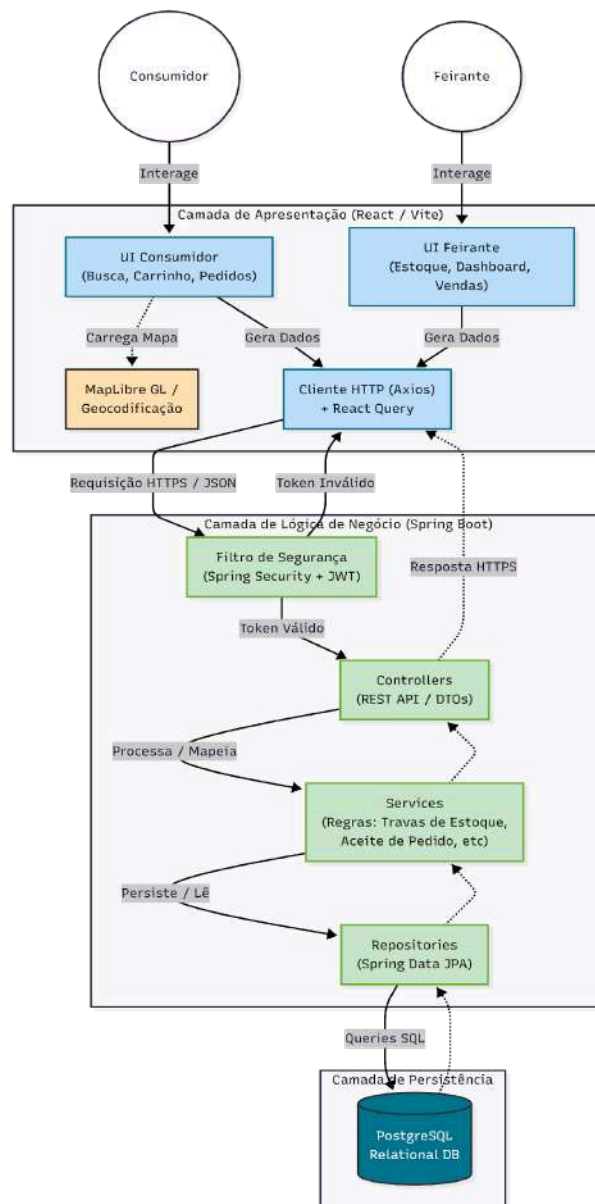
Fonte: Autores (2026).

2.8 Diagrama de Arquitetura

De acordo com Bass, Clements e Kazman (2021), a arquitetura de software é a estrutura fundamental de um sistema, composta por seus elementos, propriedades externamente visíveis e os relacionamentos entre eles. Ela serve como o projeto mestre que orienta o desenvolvimento, garantindo atributos de qualidade como escalabilidade e manutenção.

A organização geral dos componentes e camadas do SecondBite é apresentada na figura 5 a seguir:

Figura 5 - Diagrama de Arquitetura.



Fonte: Autores (2026).

O diagrama apresentado ilustra o fluxo de comunicação e a topologia do sistema SecondBite. A arquitetura foi concebida sob o modelo cliente-servidor, dividida em duas pontas principais. De um lado, os dispositivos móveis dos usuários interagem com a interface do sistema (*front-end*). Do outro, as ações do usuário disparam requisições HTTP (via API) para o servidor (*back-end*). O servidor atua como orquestrador do sistema, validando permissões, processando o fluxo lógico de compras e ofertas, e comunicando-se de forma isolada e segura com o banco de

dados. Essa separação garante que a interface de usuário evolua independentemente da lógica de armazenamento.

3 IMPLEMENTAÇÃO

Este capítulo tem como objetivo detalhar o processo de construção e implementação do sistema SecondBite. São apresentadas as tecnologias, os *frameworks* e as bibliotecas adotadas para o desenvolvimento da aplicação. Posteriormente, são demonstrados os principais fluxos de tela e as funcionalidades entregues aos usuários finais.

3.1 Tecnologias e Ferramentas Utilizadas

A plataforma foi estruturada sob o modelo cliente-servidor, adotando uma arquitetura baseada em APIs RESTful. O núcleo lógico e o processamento de dados (*Back-end*) foram desenvolvidos utilizando a linguagem Java 21 com o *framework* Spring Boot, adotando o PostgreSQL como Sistema de Gerenciamento de Banco de Dados (SGBD) e o padrão JWT (*JSON Web Token*) para garantir a segurança e a autenticação das rotas.

A interface de usuário (*Front-end*) foi projetada como uma *Single Page Application* (SPA) com foco na usabilidade em dispositivos móveis (*mobile-first*). Para isso, adotou-se a biblioteca React com TypeScript e Vite, utilizando o *framework* Tailwind CSS para a estilização ágil e responsiva. O gerenciamento de estados globais foi realizado pelo Zustand, enquanto a comunicação com a API e a integração de componentes de mapas e geolocalização foram viabilizadas pelas bibliotecas Axios e TanStack Query. Por fim, todo o planejamento visual e a prototipagem das telas foram concebidos previamente no Figma, e o versionamento do código-fonte foi controlado de forma colaborativa por meio do Git e GitHub.

3.2 Descrição e Estrutura da Aplicação

A aplicação SecondBite foi concebida como um sistema web responsivo, com design orientado prioritariamente a dispositivos móveis (*mobile-first*). Para suportar a agilidade exigida por transações de *marketplace* e geolocalização, a aplicação foi estruturada de forma modular e desacoplada, separando claramente as responsabilidades de interface e de processamento lógico.

3.2.1 Padrões de Projeto e Camadas

No lado do servidor, a aplicação não se limita a um padrão único, adotando uma mescla do padrão arquitetural MVC (*Model-View-Controller*) com uma arquitetura orientada a serviços (*Service Layer Pattern*). Essa abordagem híbrida é ideal para o fornecimento de APIs REST:

- **Controllers:** Camada responsável por expor os *endpoints* da API, interceptar as requisições HTTP vindas do *Front-end* e validar os dados de entrada.
- **Services:** Onde reside o coração da aplicação. Esta camada isola toda a regra de negócio, como o cálculo de descontos, validação de limite de estoque e checagem de geolocalização, garantindo que os *controllers* fiquem limpos e que a lógica possa ser reaproveitada.
- **Repositories e Entities:** Camada responsável pelo mapeamento dos objetos da aplicação para o banco de dados relacional, utilizando o Spring Data JPA.
- **View:** Ao invés de o servidor renderizar telas HTML tradicionais, a responsabilidade de "View" foi totalmente delegada ao *Front-end*, que consome os dados via formato JSON.

No lado do cliente, a interface web foi desenvolvida seguindo o padrão de Arquitetura Baseada em Componentes do ecossistema React. A interface gráfica foi construída a partir de blocos isolados e reutilizáveis, facilitando a manutenção e proporcionando uma experiência visual fluida e adaptável para o uso em *smartphones* por meio dos navegadores, dispensando a necessidade de instalação de aplicativos nativos.

3.2.2 Módulos Principais do Sistema

Para organizar a execução lógica do código e atender aos diferentes públicos, a plataforma foi estruturada em três macro-módulos funcionais:

- **Módulo de Autenticação e Gestão de Perfis:** Gerencia o ciclo de vida dos usuários. É responsável pelo cadastro, *login* e geração de *tokens* JWT. Este

módulo assegura que um "Consumidor" tenha acesso à interface de compras, enquanto um "Feirante" seja direcionado ao painel de administração de sua barraca.

- **Módulo do Feirante (Painel Administrativo):** Focado na operação do comerciante. Permite o cadastro de novos produtos *in natura*, o controle de estoque e a ativação do “Modo Xepa” (Descontos em massa), aplicando descontos rápidos para evitar o desperdício ao fim do horário de feira.
- **Módulo do Consumidor (Marketplace e Mapas):** Voltado para a experiência do cliente final. Integra os serviços de geolocalização para listar barracas cadastradas no sistema. Engloba as funcionalidades de busca, carrinho de compras e acompanhamento do *status* do pedido para retirada presencial.

3.3 Práticas de Desenvolvimento e Versionamento de Código

Para garantir a organização, a rastreabilidade e a colaboração eficiente durante a construção do SecondBite, o projeto adotou práticas modernas de engenharia de software e gerenciamento de configuração, alinhadas aos princípios da metodologia ágil Scrum.

A plataforma escolhida para a hospedagem e o gerenciamento remoto do código-fonte foi o GitHub, operando em conjunto com o sistema de controle de versão distribuído Git. Para manter a integridade da ramificação principal do projeto, a equipe adotou uma estratégia de *Feature Branching*. Sob essa abordagem, cada nova funcionalidade, correção ou incremento idealizado para a *sprint* vigente era desenvolvido em uma *branch* isolada. Ao término do desenvolvimento e validação da tarefa, o código era integrado à ramificação principal, garantindo entregas contínuas de pacotes funcionais ao fim de cada ciclo do Scrum.

Para manter o histórico de alterações legível e semântico, estabeleceu-se o uso do padrão *Conventional Commits*. Essa prática consiste em prefixar as mensagens de *commit* com classificadores padronizados. Isso facilitou a compreensão cronológica da evolução do sistema e a identificação rápida das modificações realizadas por cada membro no repositório.

No que tange às práticas de infraestrutura e entrega, o projeto adotou a prática de containerização para garantir a padronização do ambiente de execução. O sistema foi empacotado utilizando o Docker e a imagem resultante da aplicação foi armazenada no *Docker Hub*. O uso do Docker eliminou inconsistências de configuração entre as máquinas de desenvolvimento e o ambiente de produção, assegurando que o sistema, suas dependências e variáveis de ambiente funcionassem de maneira idêntica independentemente da infraestrutura hospedeira.

3.4 Implantação

A implantação do SecondBite foi realizada com o uso do Docker, utilizando contêineres para padronizar o ambiente da aplicação e evitar problemas de compatibilidade entre diferentes máquinas e sistemas operacionais. As imagens da aplicação foram armazenadas no Docker Hub e atualizadas conforme o avanço das sprints do projeto.

Durante o desenvolvimento, cada nova entrega gera uma versão da aplicação publicada no repositório público do projeto no Docker Hub, facilitando o controle e o acompanhamento das versões implementadas.

Ao final do projeto, foi disponibilizada a versão final do MVP (*Minimum Viable Product*), reunindo as funcionalidades desenvolvidas ao longo das *sprints* em uma imagem estável e pronta para utilização.

3.5 Roteiro de Testes

Os testes do SecondBite foram organizados por *sprints*, acompanhando o mesmo fluxo adotado durante o desenvolvimento da aplicação. Para cada *sprint*, foi elaborado um plano de testes contendo cenários, passos de execução, dados de entrada e resultados esperados. Todos os casos de teste executados apresentaram resultados satisfatórios.

Sprint 1 — CRUD de Consumidor e Login

Essa *sprint* contemplou o fluxo completo de gerenciamento da conta do consumidor. Os cenários testados incluíram tentativa de login com perfil inexistente, cadastro de novo usuário com perfil consumidor, alteração de dados cadastrais, autenticação com credenciais válidas, exclusão de conta e validação do bloqueio de acesso após a exclusão do perfil. Todos os 17 passos apresentados na Tabela 1 e definidos no plano de testes foram executados com sucesso.

Tabela 1 - CRUD de Consumidor e Login

ID	Cenário	Passo	Descrição do Passo	Resultado Esperado
1.0	Login inválido	1	Acessar tela de login com perfil não criado	Dados Incorretos
1.1	Cadastro	2	Acessar tela de cadastro	Tela de Cadastro exibida
1.1	Cadastro	3	Selecionar perfil de Consumidor	Perfil Consumidor selecionado
1.1	Cadastro	4	Preencher dados do usuário	Dados aceitos
1.1	Cadastro	5	Confirmar cadastro	Usuário cadastrado com sucesso
1.2	Alteração de Dados	6	Acessar perfil	Dados do usuário exibidos
1.2	Alteração de Dados	7	Clicar em "editar"	Dados do usuários liberados para editar
1.2	Alteração de Dados	8	Alterar nome	Novo nome
1.2	Alteração de Dados	9	Alterar telefone	Novo CPF
1.2	Alteração de Dados	10	Clicar em "Salvar alterações"	Dados do usuário alterado
1.3	Login	11	Clicar em "sair da conta"	Sair da conta
1.3	Login	12	Acessar tela de login	Tela de login exibida
1.3	Login	13	Inserir credenciais	Login realizado com sucesso
1.4	Exclusão de Conta	14	Acessar perfil	Tela de perfil exibida
1.4	Exclusão de Conta	15	Clique em "Excluir conta"	Sistema solicita confirmação

1.4	Exclusão de Conta	16	Confirmar exclusão	Conta excluída com sucesso
1.5	Login inválido	17	Tentar login após exclusão	Dados Incorretos

Fonte: Autores (2026)

Sprint 2 — Consulta de Produtos (Busca e Filtros)

Essa *sprint* validou as funcionalidades de navegação e filtragem de produtos disponíveis na plataforma. Os testes abrangeram o acesso à tela de busca, filtragem por categorias, exibição completa dos produtos cadastrados e pesquisa por nome específico. Os 7 passos previstos na Tabela 2 foram concluídos sem falhas.

Tabela 2 - Consulta de Produtos (Busca e Filtros)

ID	Cenário	Passo	Descrição do Passo	Resultado Esperado
2.0	Login	1	Acessar tela de login	Login Permitido
2.1	Produto	2	Acessar tela de buscar	Tela de Produto exibida
2.1	Produto	3	Clicar em frutas	Só frutas sendo exibida
2.1	Produto	4	Clicar em vegetais	Só verduras sendo exibida
2.2	Produto	5	Clicar em tudo	Todos os produtos sendo exibidos
2.2	Busca	6	Clicar na barra buscar	Banana sendo exibida
2.2	Busca	7	Clicar na barra buscar	Uva sendo exibida

Fonte: Autores (2026)

Sprint 3 — Visualização e Avaliação de Feirante

Essa *sprint* teve como objetivo validar o fluxo de acesso ao perfil do feirante por meio de um produto e o funcionamento do sistema de avaliações. Os cenários contemplaram visualização do perfil do feirante, registro de avaliação com nota e

comentário, edição da avaliação realizada e exclusão da avaliação. Todos os 11 passos ilustrados na Tabela 3 apresentaram comportamento conforme o esperado.

Tabela 3 - Visualização e Avaliação de Feirante

ID	Cenário	Passo	Descrição do Passo	Resultado Esperado
3.0	Login	1	Acessar tela de login	Login Permitido
3.1	Produtos	2	Acessar tela de buscar	Tela de busca exibida
3.1	Produtos	3	Clicar em melancia	Perfil da melancia é exibido
3.2	Feirante	4	Clicar no feirante	Feirante " Ana Frutas" exibido
3.3	Avaliação	5	Clicar em Avaliação	Avaliação exibida
3.3	Avaliação	6	Clicar na nota	Nota 4 preenchida
3.3	Avaliação	7	Clicar em comentário	Comentário preenchido
3.3	Avaliação	8	Clicar em enviar	Avaliação feita
3.3	Avaliação	9	Clicar em editar	Nota 5 preenchida
3.3	Avaliação	10	Clicar em salvar	Avaliação editada
3.3	Avaliação	11	Clicar em excluir	Avaliação excluída

Fonte: Autores (2026)

3.6 Telas do Sistema

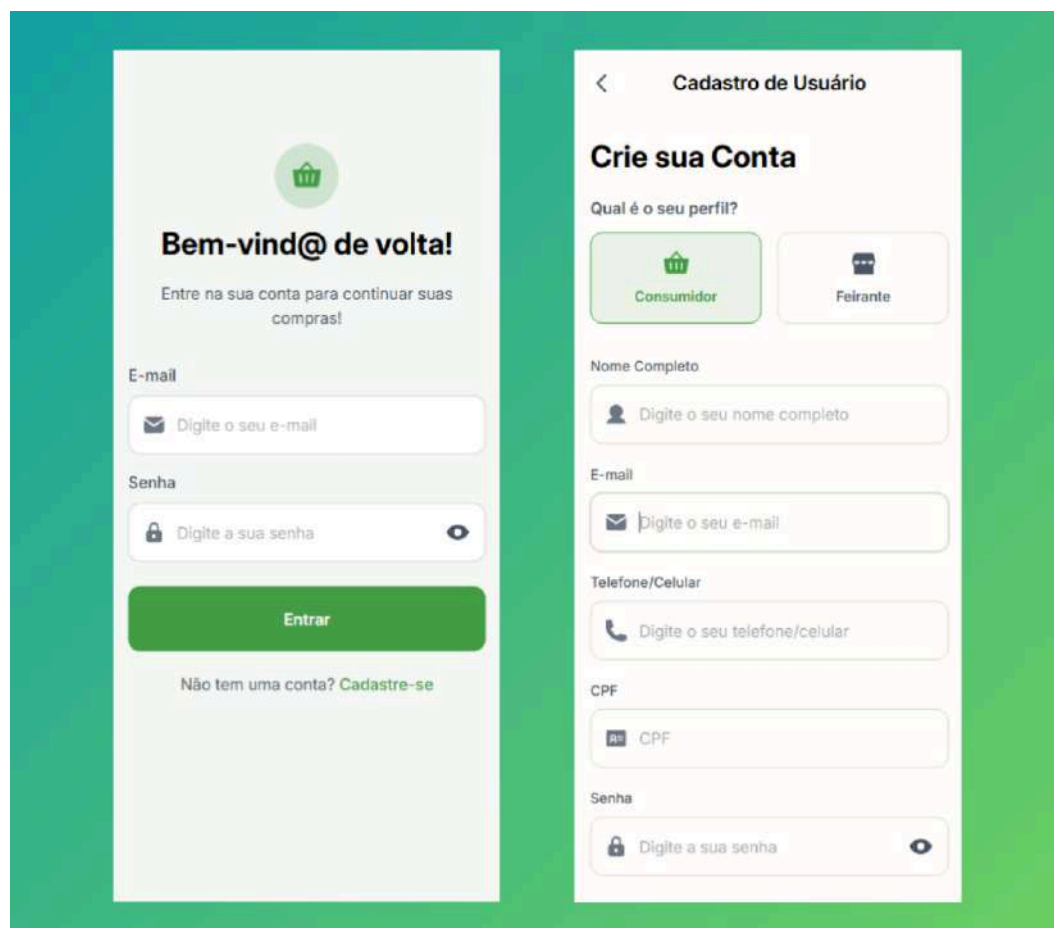
Este subcapítulo tem como objetivo apresentar as interfaces do sistema SecondBite, demonstrando a concretização do design e o funcionamento da aplicação na prática. As telas foram desenvolvidas seguindo a abordagem *mobile-first*, priorizando a usabilidade, a acessibilidade e a fluidez em dispositivos móveis, características essenciais para o uso em feiras livres. A seguir, as interfaces são detalhadas e correlacionadas diretamente com os requisitos funcionais, não-funcionais e regras de negócio que orientaram o seu desenvolvimento, comprovando o atendimento ao escopo do projeto.

3.6.1 Telas do Perfil de Consumidor

As interfaces a seguir descrevem a jornada do usuário consumidor na plataforma SecondBite, abrangendo desde o acesso inicial até a conclusão do processo de reserva de produtos.

A porta de entrada do sistema é composta pelas interfaces de autenticação e registro, conforme ilustrado na Figura 6. A tela permite a criação de novas contas e o acesso seguro de usuários já cadastrados. Em conformidade com o **RNF001**, o acesso é protegido por *tokens* JWT. No processo de registro, o sistema valida a **RN015** (Unicidade de Identidade), impedindo cadastros duplicados com o mesmo e-mail.

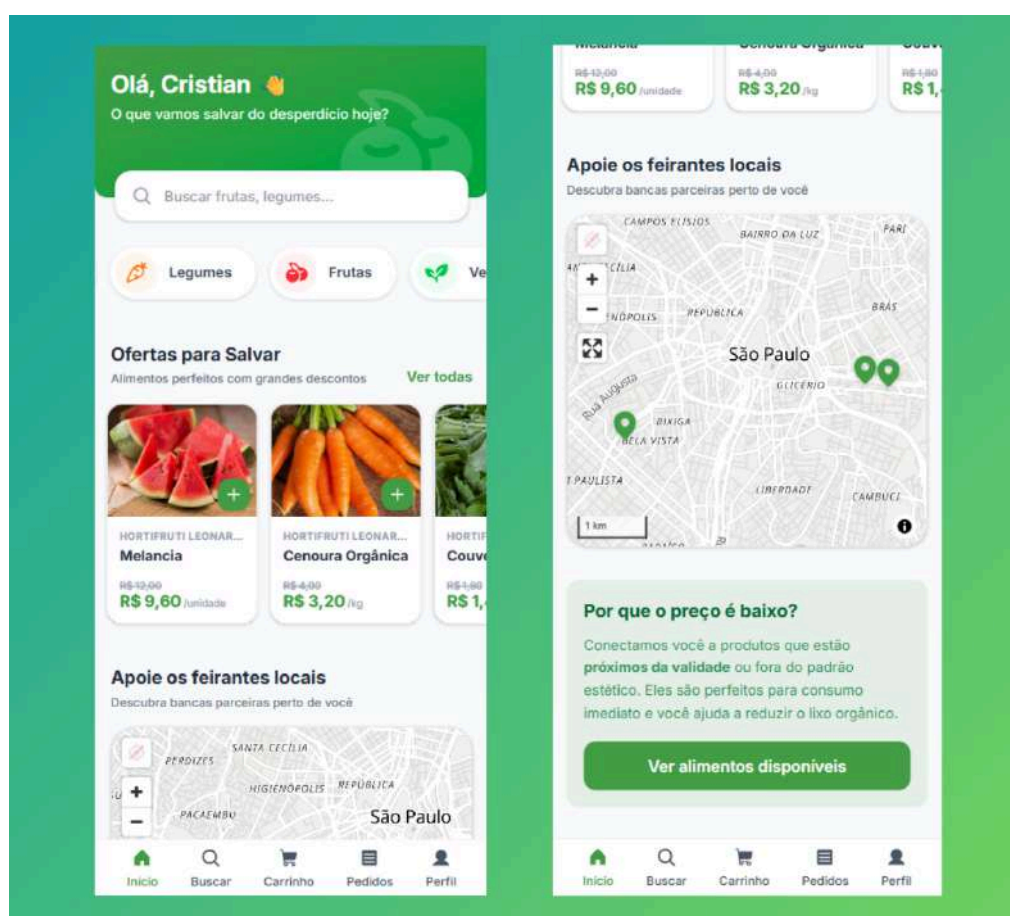
Figura 6 - Protótipo do Sistema - Login e Registro.



Fonte: Autores (2026).

Após o acesso, o usuário é direcionado à tela inicial, apresentada na Figura 7. Esta interface centraliza as principais categorias de produtos e ofertas em destaque. O design foi estruturado para atender ao **RF010**, permitindo que o consumidor inicie sua busca de forma intuitiva através de filtros por categorias como frutas, legumes e verduras.

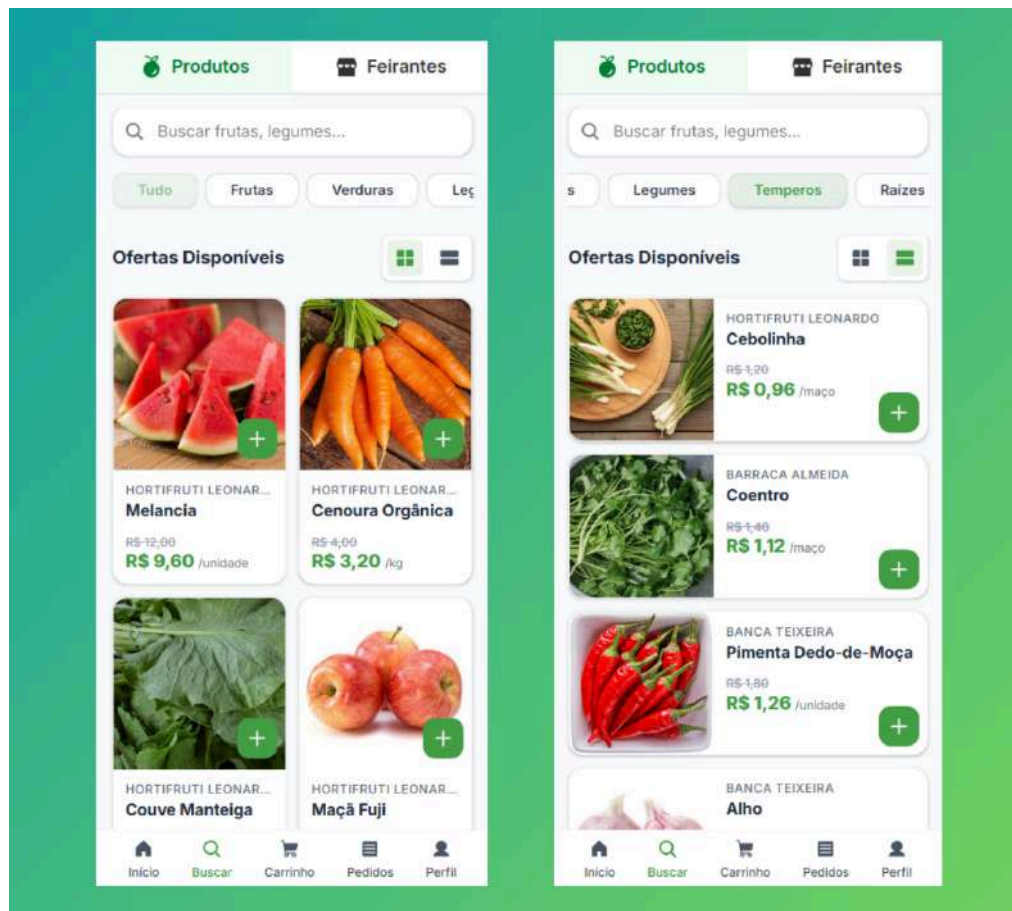
Figura 7 - Protótipo do Sistema - Tela de Início Perfil Consumidor.



Fonte: Autores (2026).

A Figura 8 exibe a interface de busca e listagem de produtos. Esta tela concretiza a funcionalidade prevista no **RF010** (Busca e Filtro de Catálogo), permitindo ao consumidor localizar itens específicos por nome ou categoria, exibindo os preços atualizados e informações básicas de cada produto.

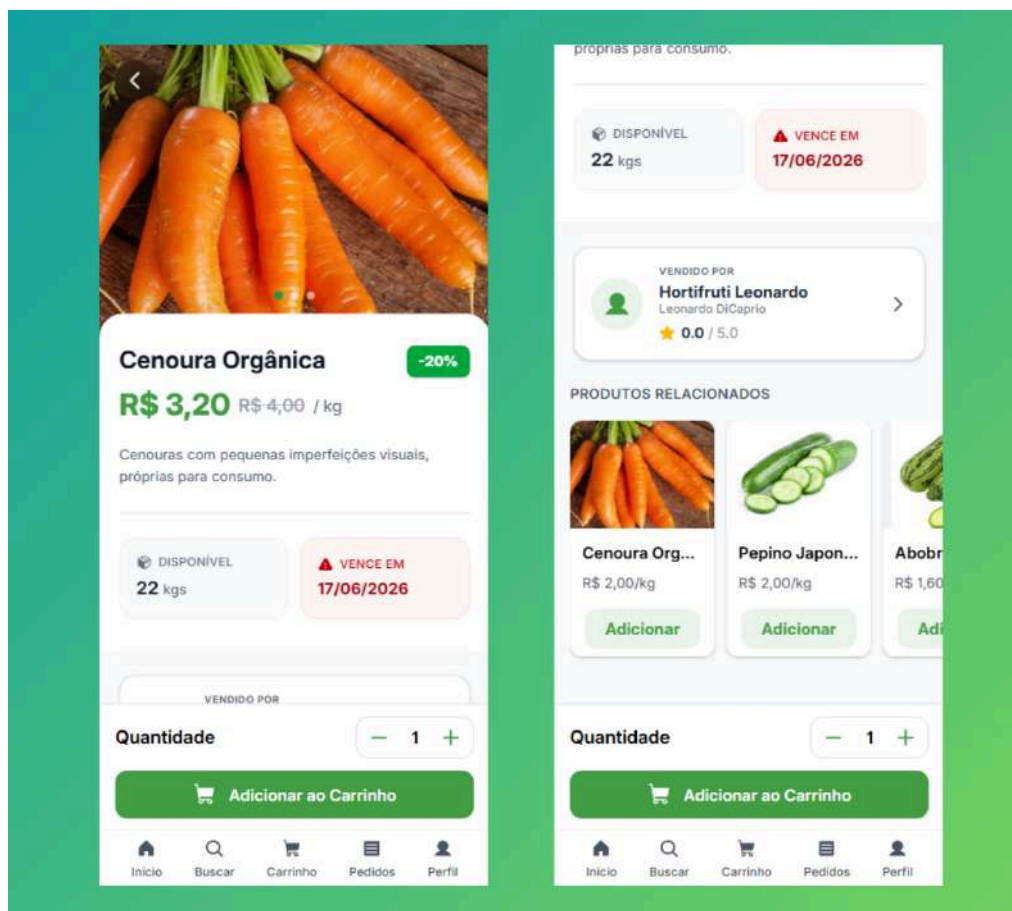
Figura 8 - Protótipo do Sistema - Busca e Filtro de Produtos.



Fonte: Autores (2026).

Ao selecionar um produto específico, a aplicação apresenta as informações detalhadas, conforme ilustrado na Figura 9 . Além da descrição e imagens do item, a tela permite definir a quantidade desejada. O sistema aplica nesta etapa a **RN003** (Garantia e Trava de Estoque), impedindo que o consumidor adicione quantidades superiores às disponíveis no inventário do feirante.

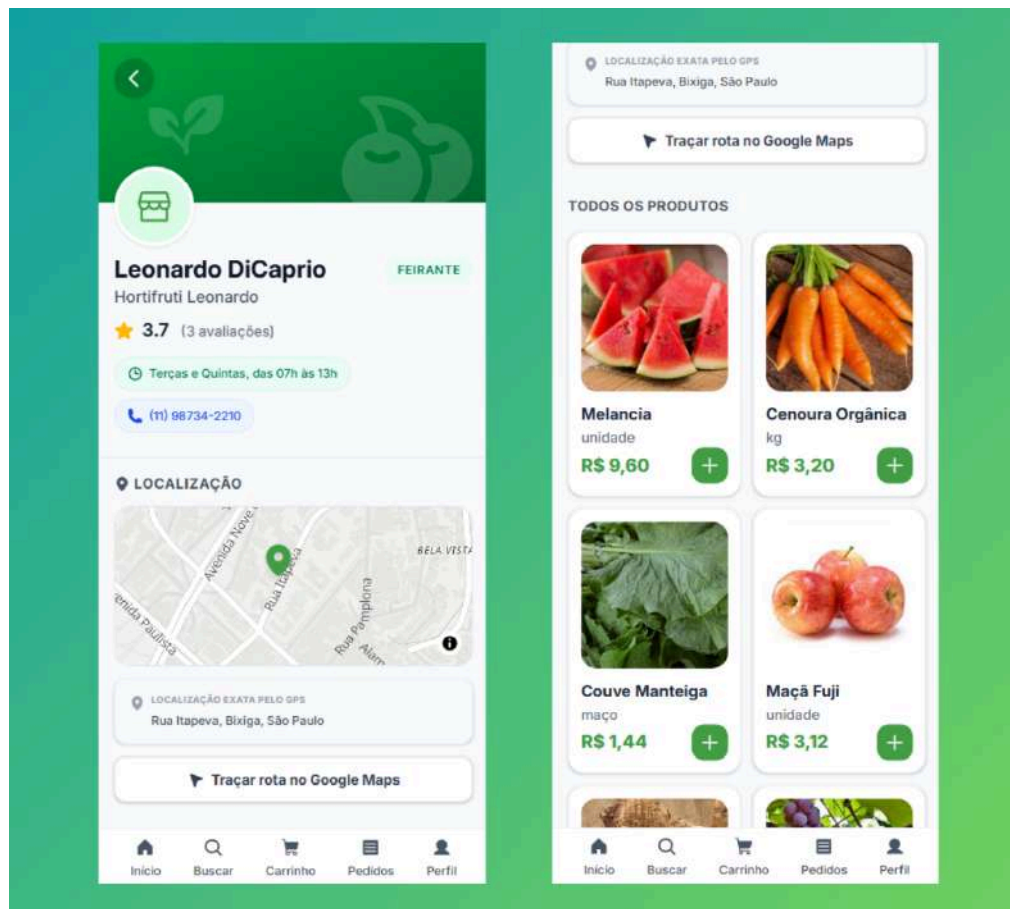
Figura 9 - Protótipo do Sistema - Detalhes do Produto



Fonte: Autores (2026).

A Figura 10 ilustra a vitrine pública do feirante. Atendendo ao **RF009**, esta página consolida todos os produtos ofertados por um vendedor específico, apresentando sua nota média de reputação e meios de contato, fortalecendo a transparência no *marketplace*.

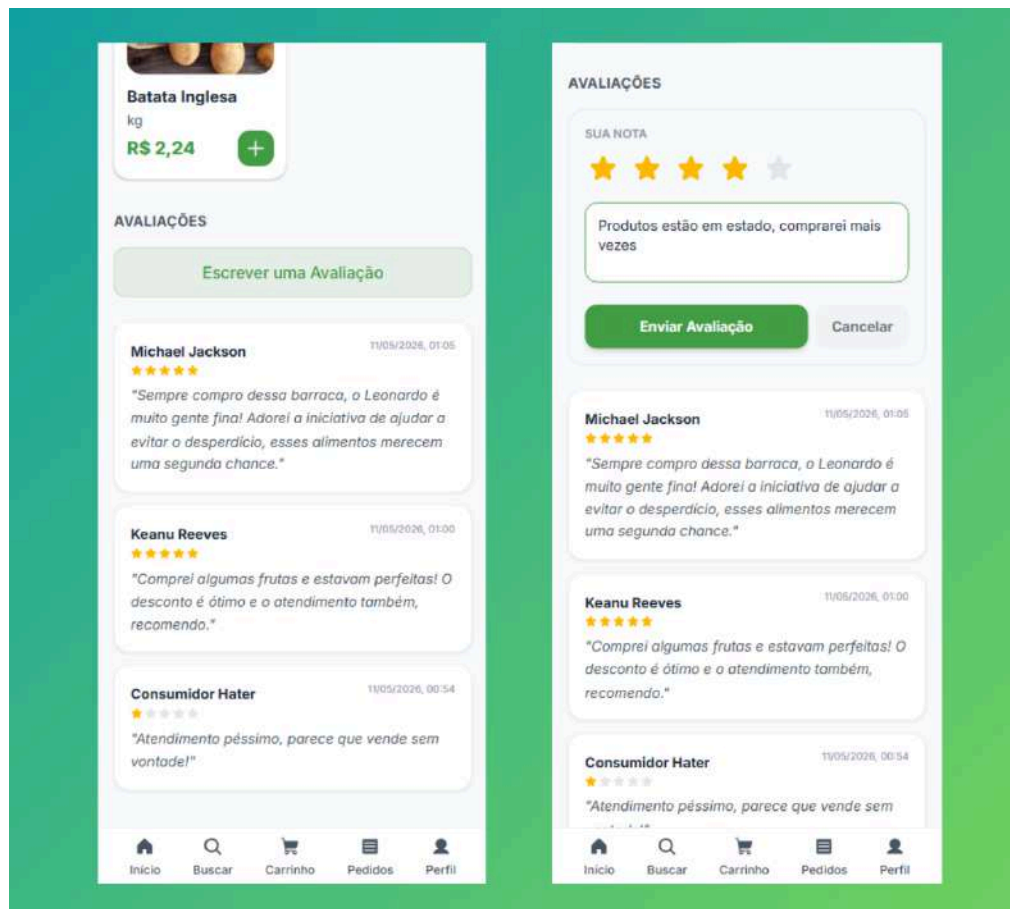
Figura 10 - Protótipo do Sistema - Vitrine do Feirante.



Fonte: Autores (2026).

Para assegurar a confiabilidade entre as partes, o sistema permite a consulta detalhada das avaliações, como mostrado na Figura 11. Esta funcionalidade cumpre o **RF020**. A submissão de novas notas é regida pela **RN014**, que garante a unicidade da avaliação por consumidor para cada feirante.

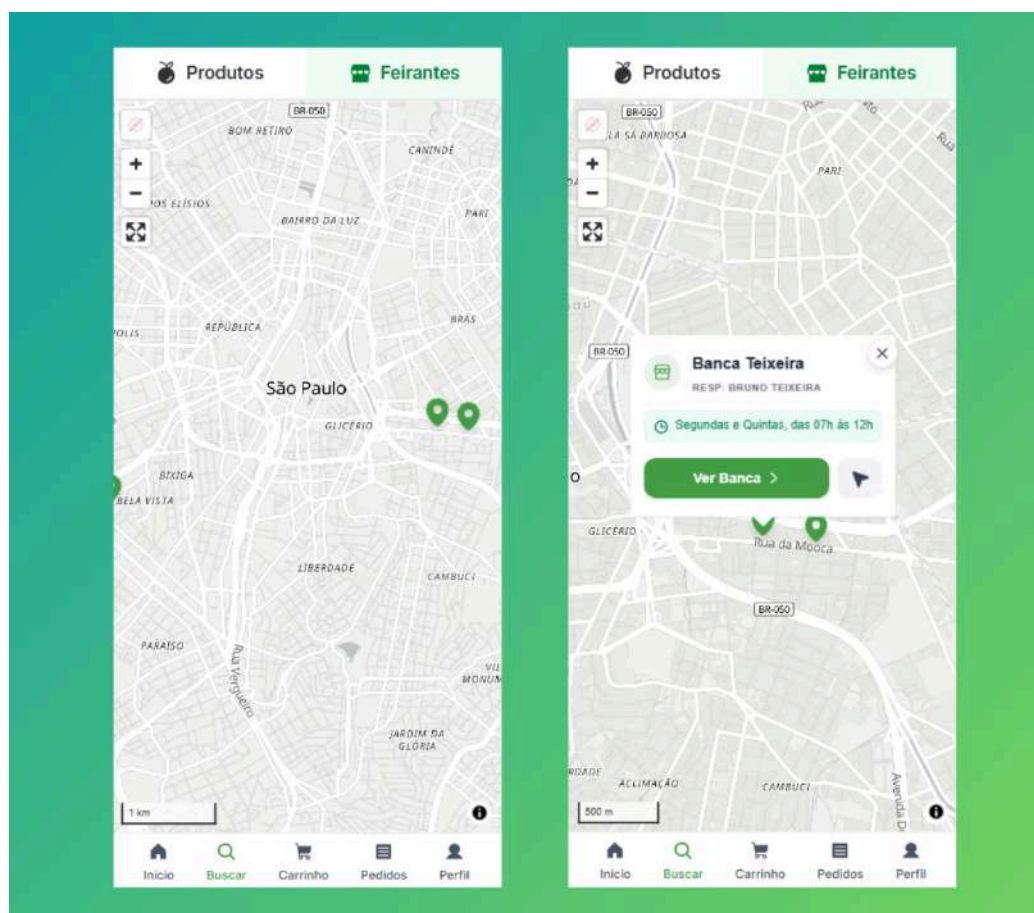
Figura 11 - Protótipo do Sistema - Avaliações de Feirante.



Fonte: Autores (2026).

A conexão geográfica entre feirantes e consumidores é viabilizada pela interface de mapa, conforme apresentado na Figura 12. Em atendimento ao **RF007**, a tela utiliza serviços de geolocalização para exibir marcadores (*pins*) das barracas ativas nas proximidades da localização do usuário.

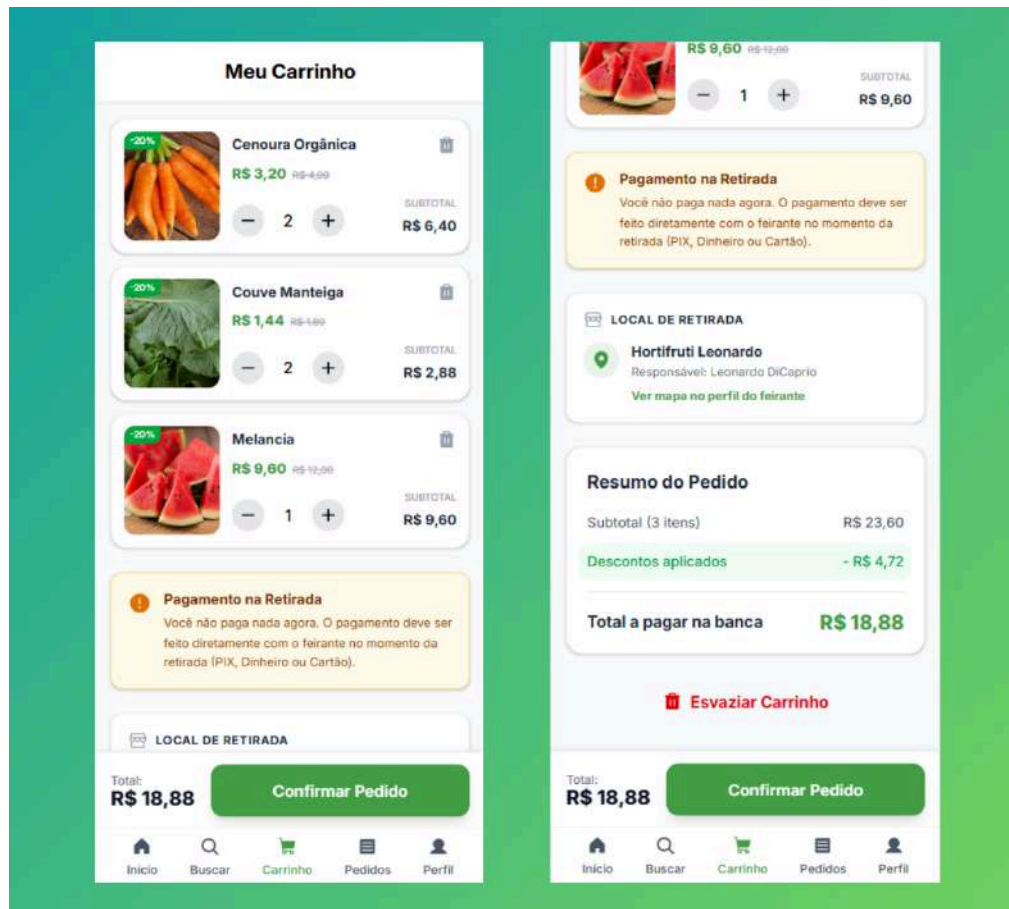
Figura 12 - Protótipo do Sistema - Mapa de Feirantes.



Fonte: Autores (2026).

A Figura 13 apresenta o carrinho de compras, onde são consolidados os itens selecionados para reserva. Esta interface valida múltiplas regras de negócio: a **RN001** garante a exclusividade de um único feirante por pedido; a **RN002** bloqueia o prosseguimento caso o carrinho esteja vazio; e a **RN004** remove automaticamente itens com quantidade zerada. A funcionalidade atende plenamente ao **RF011** e **RF012**.

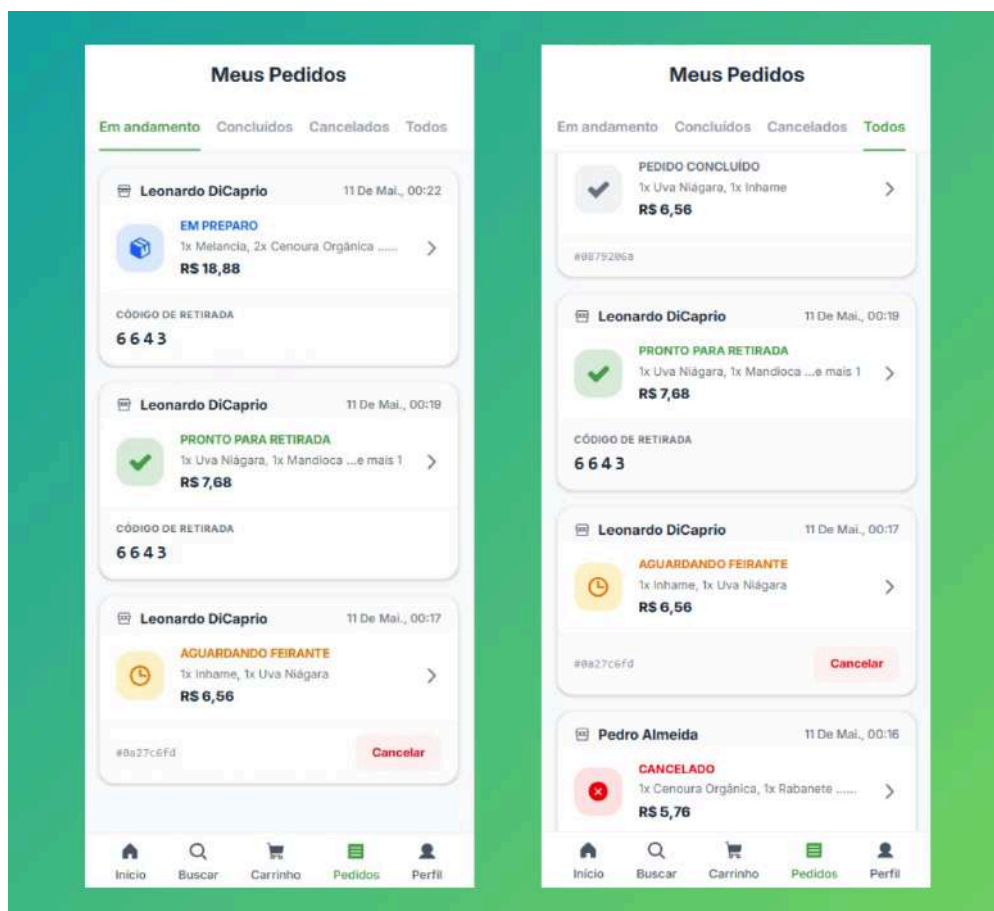
Figura 13 - Protótipo do Sistema - Carrinho de Compras.



Fonte: Autores (2026).

O acompanhamento das ordens de compra é realizado através da tela "Meus Pedidos", exibida na Figura 14. Esta interface permite ao consumidor monitorar o *status* de suas sacolas em tempo real (como "Aguardando Feirante" ou "Pronto para Coleta"), conforme definido no **RF018**. Além disso, a opção de cancelamento exibida respeita a **RN006**, sendo bloqueada caso o feirante já tenha aceitado o pedido.

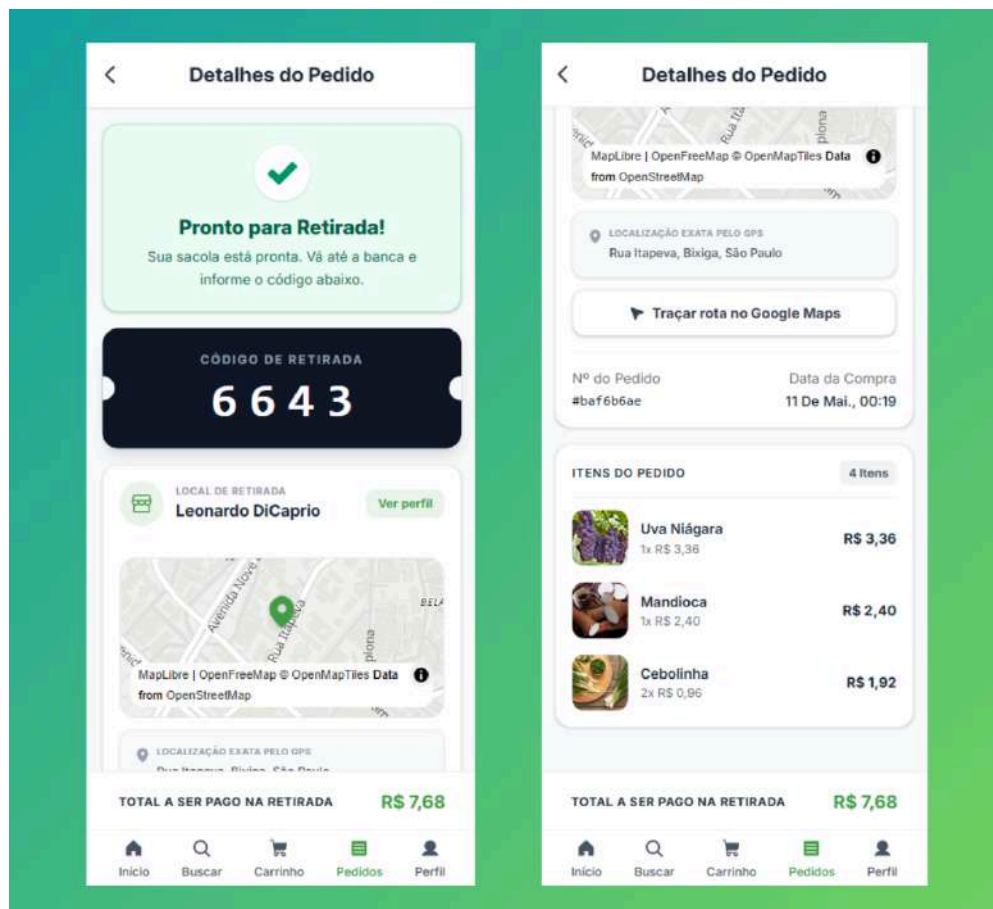
Figura 14 - Protótipo do Sistema – Meus Pedidos.



Fonte: Autores (2026).

Por fim, a Figura 15 detalha um pedido finalizado. A interface apresenta o resumo da transação e o código de segurança de 4 dígitos gerado dinamicamente (RN009), atendendo ao requisito RF014 para garantir a retirada segura na barraca.

Figura 15 - Protótipo do Sistema - Detalhes do Pedido Perfil Consumidor.



Fonte: Autores (2026).

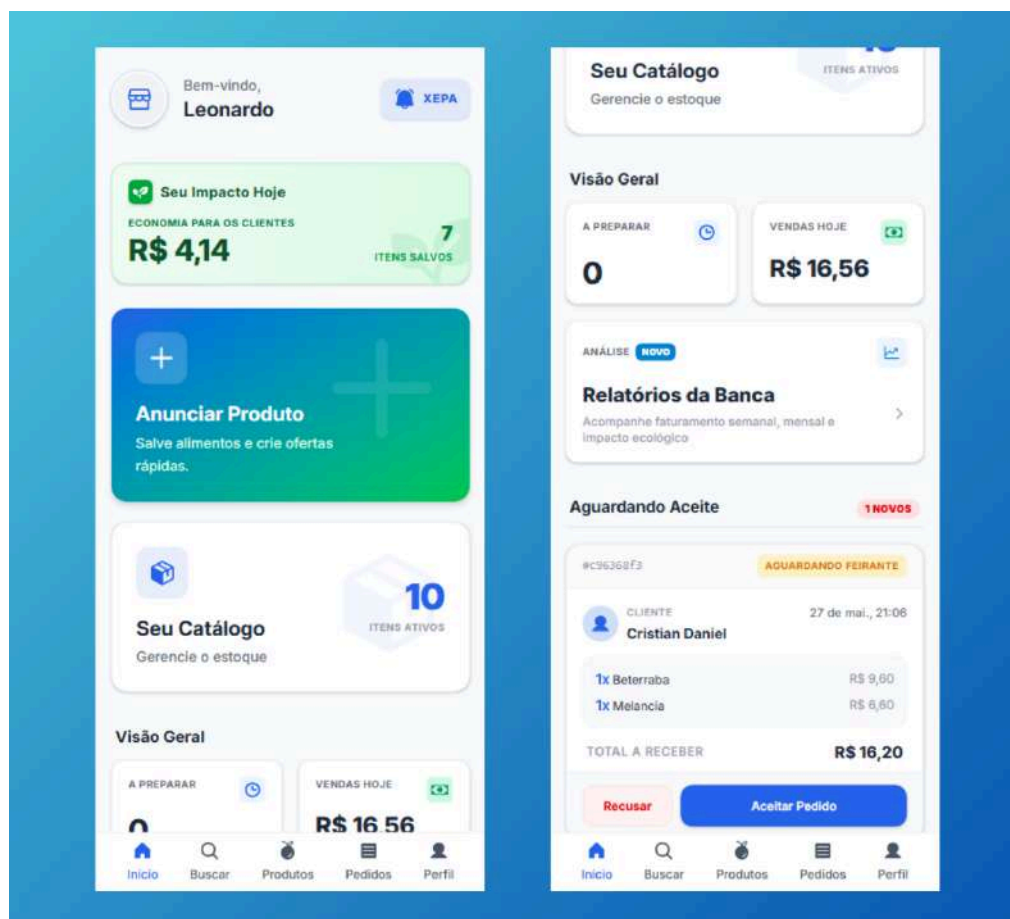
3.6.2 Telas do Perfil de Feirante

As interfaces destinadas ao feirante foram projetadas para oferecer agilidade operacional no ambiente dinâmico das feiras livres, permitindo o gerenciamento de estoque, o fluxo de pedidos e a ativação de liquidações, tudo sob a premissa de interface responsiva e adaptável a dispositivos móveis ditada pelo **RNF008**.

A interface principal do feirante é o *Dashboard*, apresentado na Figura 16. Esta tela atende de forma direta ao **RF019** (Painel de Impacto Ambiental), centralizando métricas de desempenho e calculando o impacto positivo gerado,

exibindo a quantidade de alimentos salvos do desperdício através das vendas na plataforma.

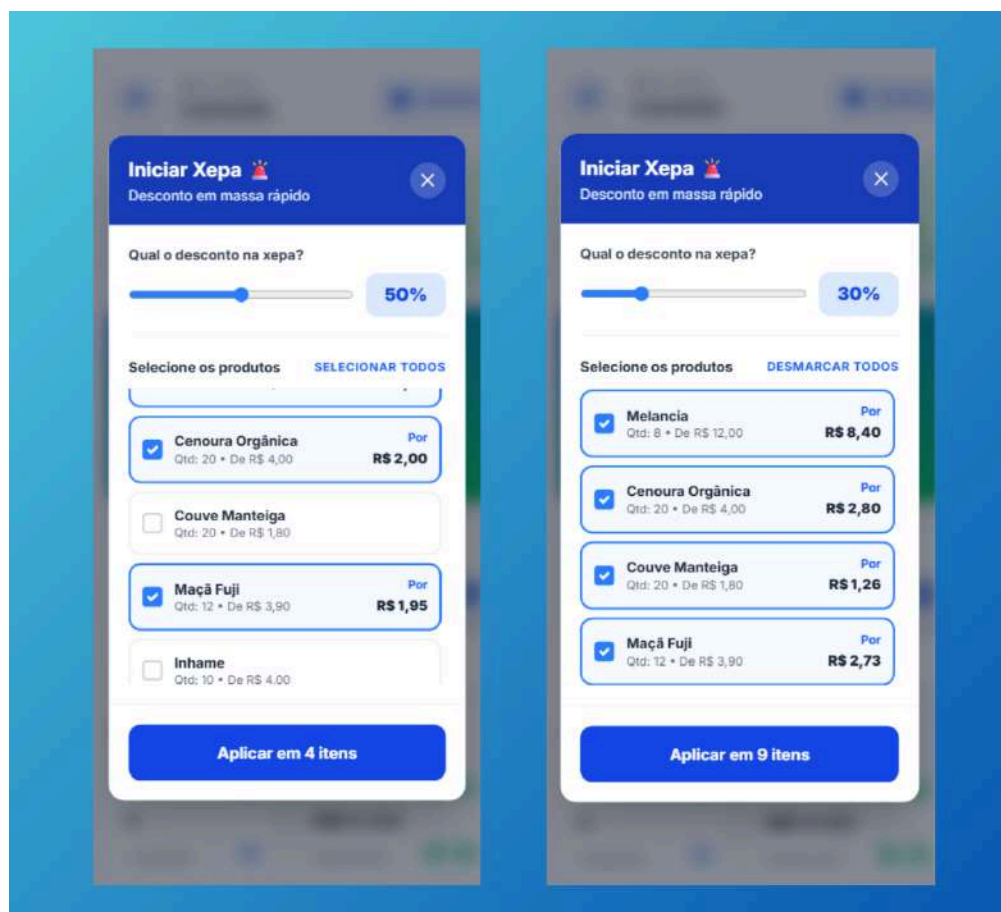
Figura 16 - Protótipo do Sistema – Tela de Início Perfil Feirante.



Fonte: Autores (2026).

A Figura 17 exibe a interface do "Modo Xepa", contemplando a funcionalidade exigida no **RF003**. Esta tela permite a aplicação de descontos agressivos em múltiplos itens simultaneamente. Ao aplicar a liquidação, o sistema aciona a **RN013** (Sobrescrita de Promoção Xepa), que se encarrega de desativar configurações de desconto automáticas prévias e impor o novo percentual manual para escoar os produtos próximos ao fim da feira. Adicionalmente, esta interface oferece apoio à precificação através do **RF004** (Sugestões Inteligentes de Desconto).

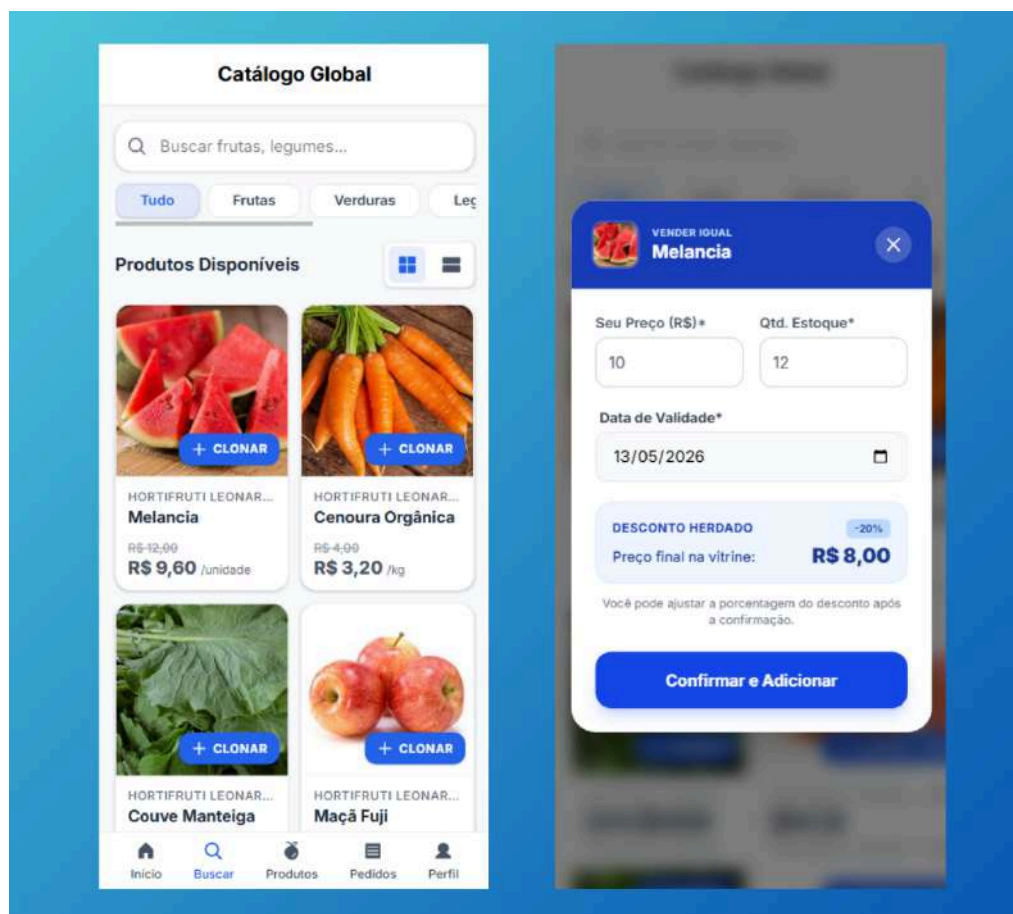
Figura 17 - Protótipo do Sistema – Modo Xepa.



Fonte: Autores (2026).

Para otimizar o tempo de cadastro, o sistema viabiliza a ferramenta de clonagem, iniciando-se pela interface de busca na Figura 18. Esta tela faz parte do fluxo do **RF005** (Clonagem Rápida de Produtos), permitindo que o feirante localize barracas e catálogos de concorrentes ou parceiros na plataforma.

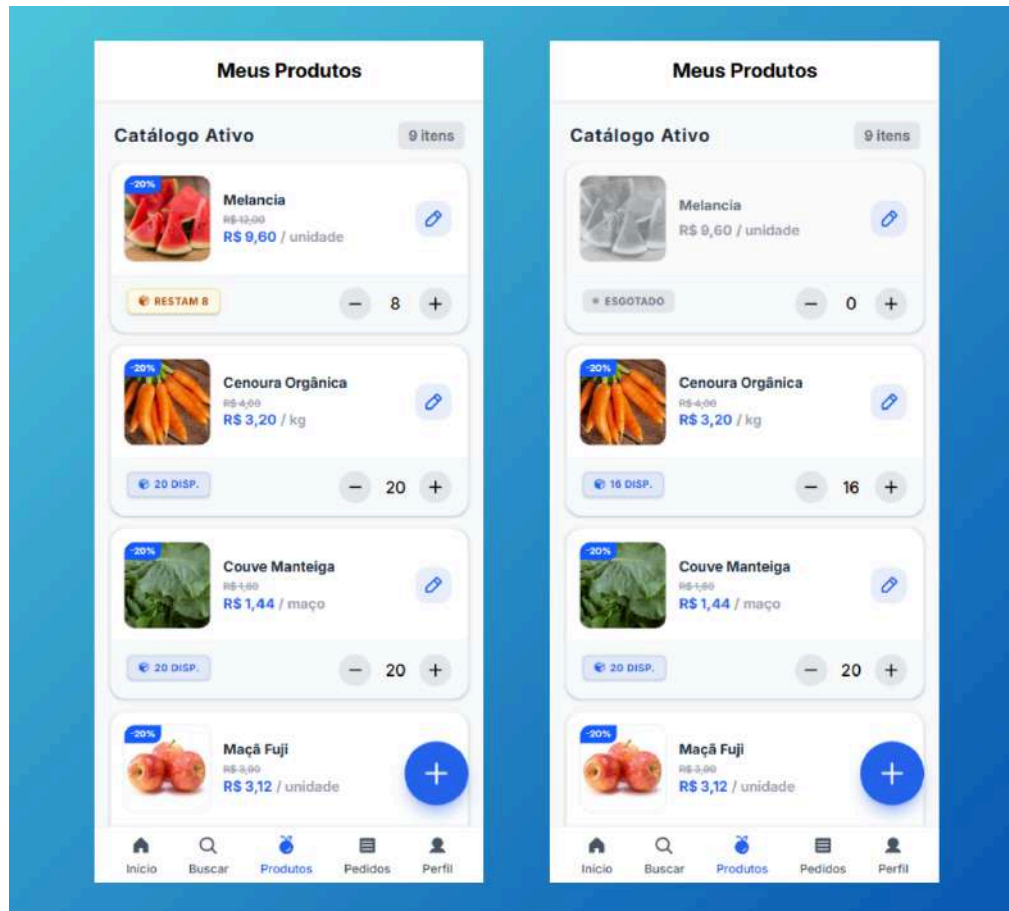
Figura 18 - Protótipo do Sistema – Clone de Produtos.



Fonte: Autores (2026).

A gestão do catálogo de mercadorias é realizada através da interface de listagem, ilustrada na Figura 19. Esta tela apoia o **RF001** e permite ao feirante atualizar seu inventário através de botões de manipulação rápida, cumprindo o **RF006**. As operações de edição e atualização do catálogo respeitam ativamente a **RN011**, e o sistema não permite em nenhuma hipótese que a inserção resulte em quantidades negativas (**RN012**).

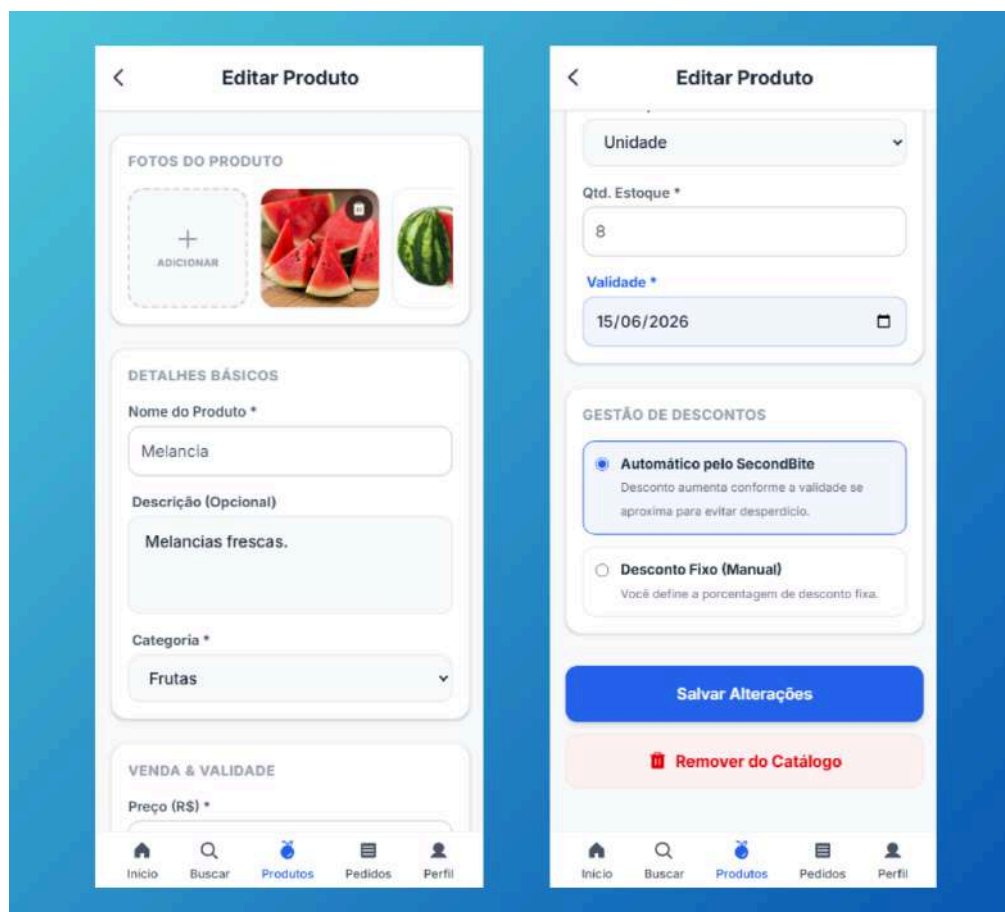
Figura 19 - Protótipo do Sistema – Gestão de Estoque e Produtos.



Fonte: Autores (2026).

A Figura 20 ilustra a interface de configuração detalhada de um item, com foco na definição da estratégia de preços. Atendendo diretamente ao **RF002** (Configuração Híbrida de Descontos), o sistema permite que o feirante defina se a redução de preço do produto seguirá uma lógica de "Desconto Automático" ou um desconto percentual manual, garantindo flexibilidade comercial.

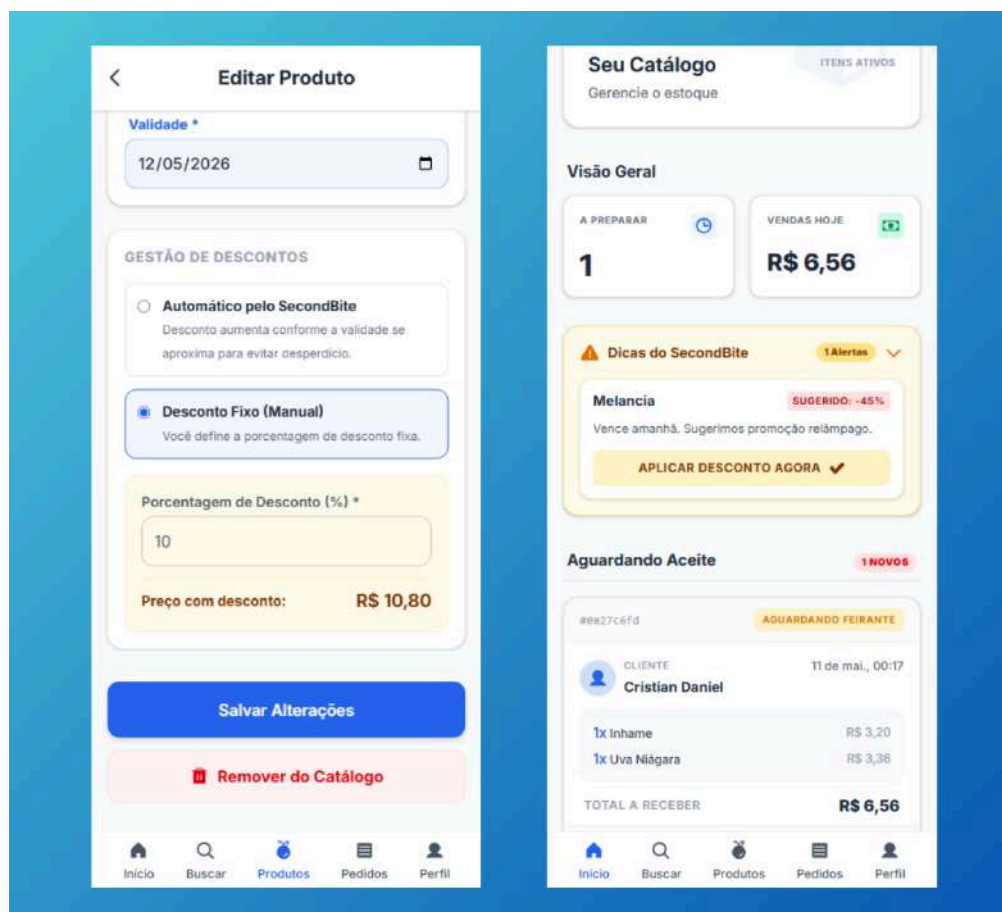
Figura 20 - Protótipo do Sistema – Configuração de Desconto Automático.



Fonte: Autores (2026).

Como apoio à tomada de decisão para evitar perdas, a plataforma fornece as “Dicas do SecondBite”, apresentadas na Figura 21. Esta tela concretiza o **RF004** (Sugestões Inteligentes de Desconto), auxiliando o feirante na precificação dinâmica de itens que estão próximos ao fim de sua vida útil na feira, maximizando a probabilidade de venda.

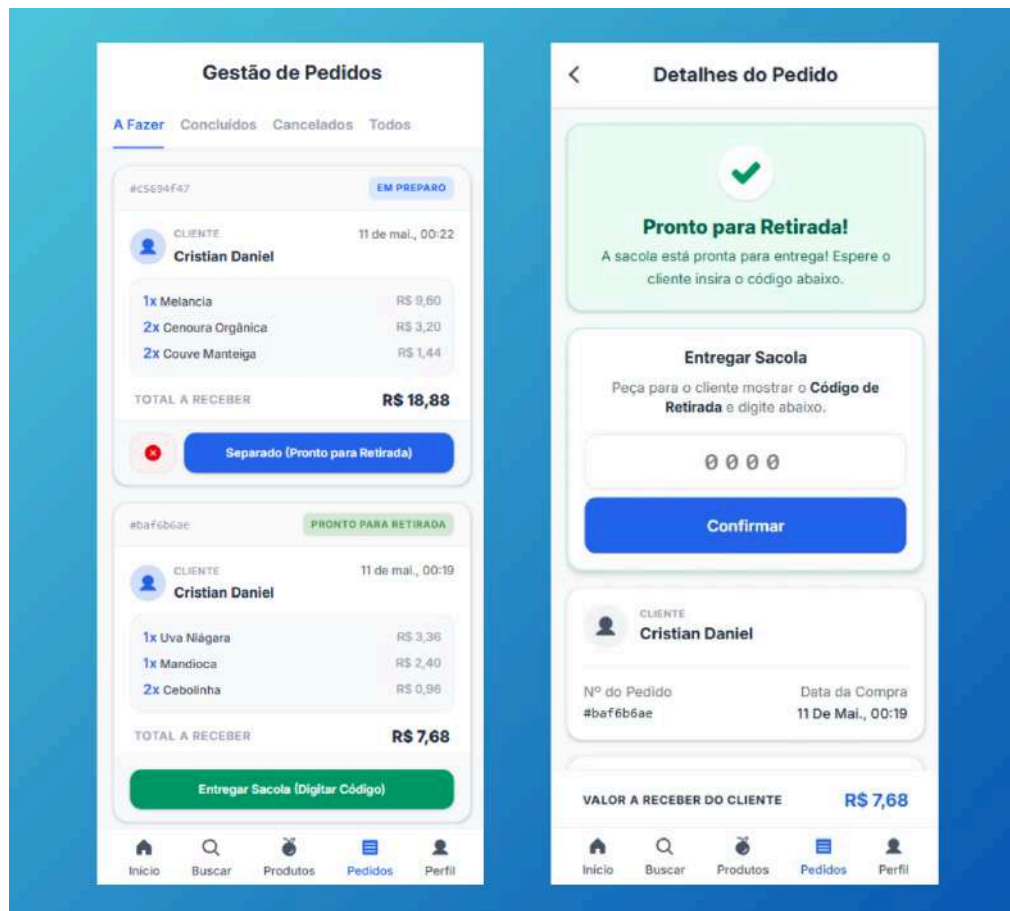
Figura 21 - Protótipo do Sistema – Sugestão de Desconto Inteligente.



Fonte: Autores (2026).

A Figura 22 atua como centralizador operacional do comerciante, ilustrando o painel de gerenciamento de pedidos recebidos. Conforme estabelecido no **RF015**, esta tela concentra validações de regras críticas: o feirante é o único com permissão para aceitar uma nova solicitação (**RN005**); para finalizar a entrega presencial, ele deve obrigatoriamente digitar a "senha" fornecida pelo consumidor (**RF016** e **RN010**); e o sistema impede a conclusão do pedido caso este não possua o *status* anterior de "Pronto para Coleta" (**RN008**). Adicionalmente, caso ocorra o cancelamento de uma sacola a partir desta tela, os itens voltam imediatamente ao catálogo disponível por meio da **RN007** (Restituição Automática de Estoque).

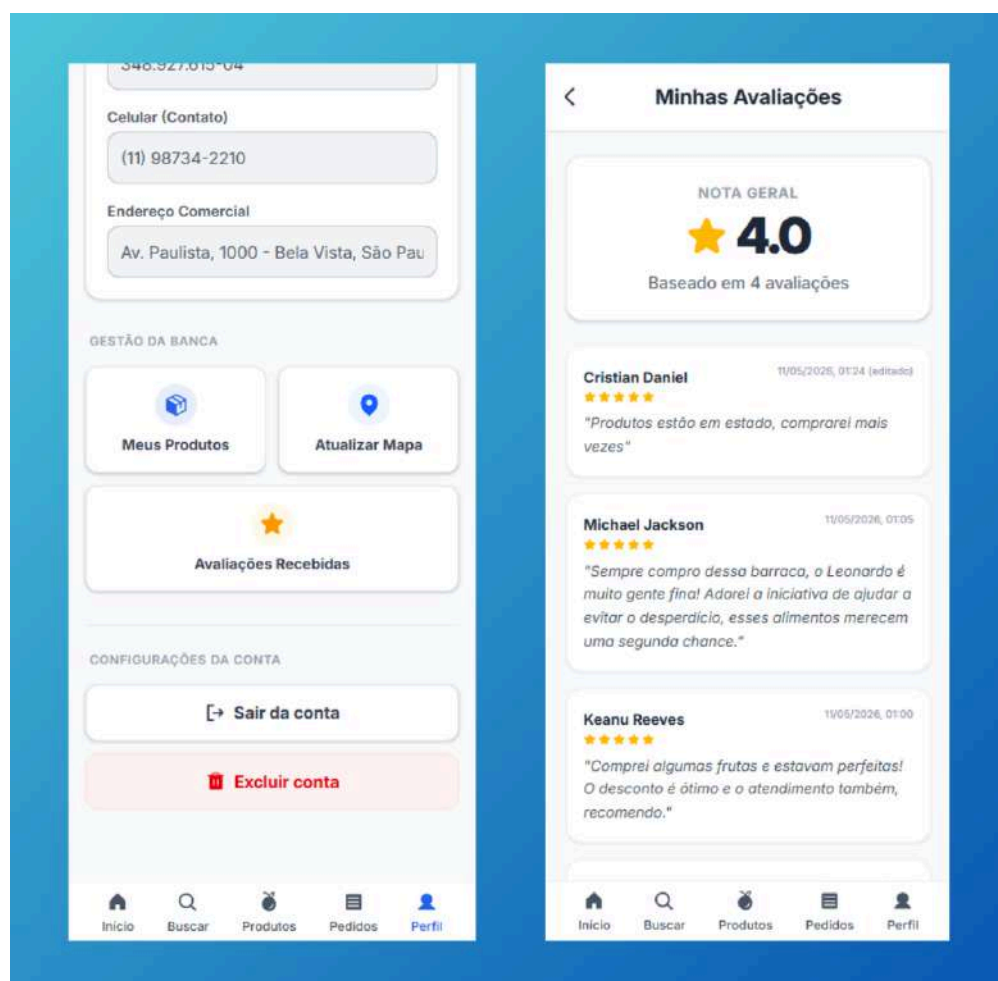
Figura 22 - Protótipo do Sistema – Gestão de Pedidos.



Fonte: Autores (2026).

Na Figura 23 exibe a interface onde o feirante acompanha as métricas qualitativas do seu serviço. Esta tela reflete os dados gerados pelo **RF020** (Sistema de Avaliações e Reputação), permitindo que o comerciante consulte as notas médias e os comentários deixados pelos consumidores na sua vitrine, garantindo o acompanhamento de sua reputação na plataforma.

Figura 23 - Protótipo do Sistema – Consultar Avaliações.

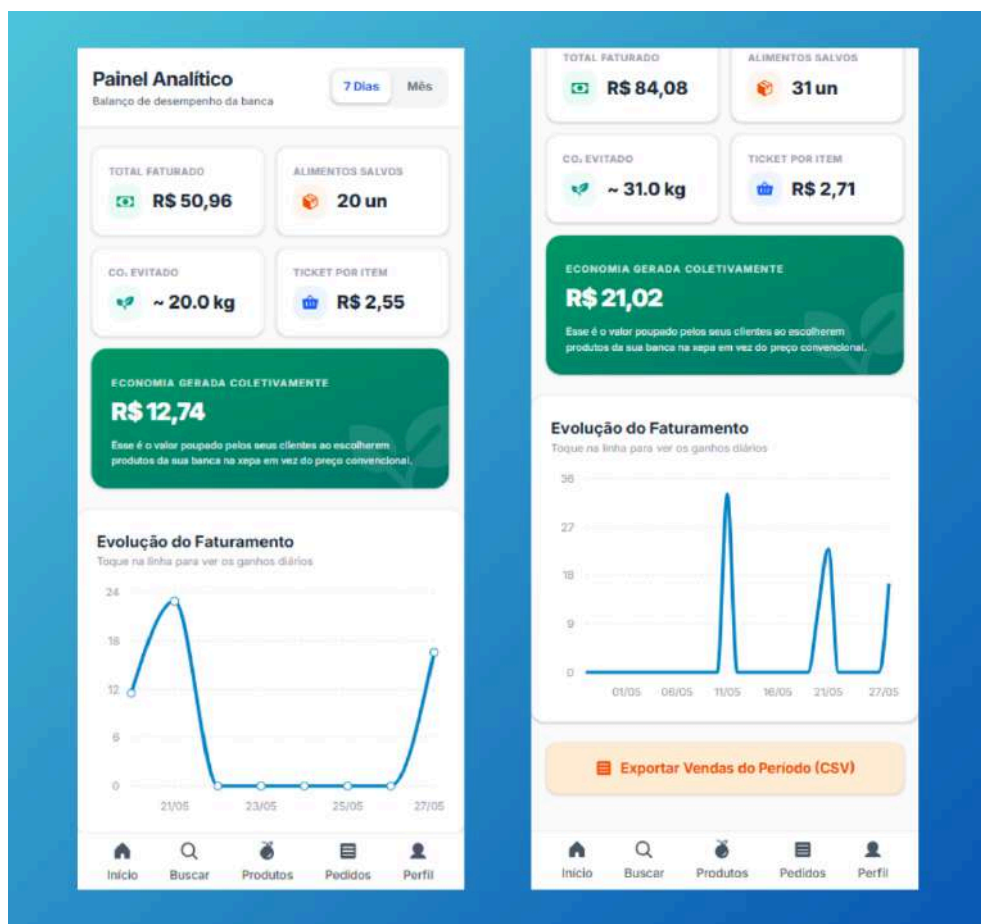


Fonte: Autores (2026).

Por fim, a Figura 24 ilustra o painel analítico avançado do comerciante. Esta interface atende diretamente ao **RF021** (Painel Analítico e Exportação de Relatórios), servindo como uma extensão gerencial das métricas diárias apresentadas no painel inicial (**RF019**). A tela fornece uma visão profunda do negócio através de um gráfico responsivo de evolução de faturamento e consolida indicadores de inteligência estratégica, como o *ticket* médio por item e a conversão do impacto sustentável (CO₂ evitado). Adicionalmente, a interface disponibiliza um recurso de exportação, permitindo que o feirante realize o download de todo o histórico detalhado de vendas do período em uma planilha (CSV) diretamente em

seu dispositivo, facilitando o controle contábil e a gestão financeira externa da banca.

Figura 24 - Protótipo do Sistema – Relatório da Banca.



Fonte: Autores (2026).

4 RESULTADOS OBTIDOS

O desenvolvimento do SecondBite demonstrou que a tecnologia pode ser utilizada como uma ferramenta eficaz no combate ao desperdício de alimentos e na promoção de práticas sustentáveis alinhadas aos princípios da economia circular. A plataforma proposta integra feirantes e consumidores em um ambiente digital acessível, permitindo que produtos ainda próprios para consumo, mas próximos do vencimento ou fora do padrão estético comercial, sejam comercializados com preços reduzidos, gerando benefícios econômicos, sociais e ambientais.

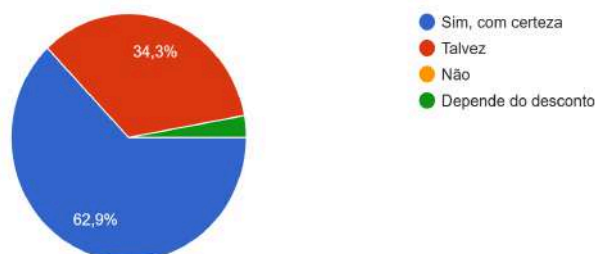
Ao longo deste trabalho, foram levantados os requisitos funcionais e não funcionais do sistema, definidas as regras de negócio, elaborada a modelagem do banco de dados e desenvolvida a implementação da aplicação utilizando tecnologias modernas, como React, Spring Boot e PostgreSQL. O resultado foi uma solução web responsiva, com abordagem *mobile-first*, capaz de oferecer funcionalidades de cadastro de produtos, busca com geolocalização, carrinho de compras, gestão de pedidos, promoções e sistema de avaliações. Para verificar se essa solução corresponderia às expectativas do público-alvo, foi realizada uma pesquisa de validação de mercado, cujos resultados são apresentados a seguir.

Os resultados da pesquisa reforçam de forma significativa o potencial de aceitação e a viabilidade do SecondBite. Quando questionados se comprariam alimentos próximos da validade, desde que estivessem em boas condições e com desconto, 62,9% dos participantes responderam "Sim, com certeza" e 34,3% responderam "Talvez". Apenas uma pequena parcela indicou que a decisão dependeria do valor do desconto, e nenhum participante respondeu negativamente. Como ilustrado na Figura 25 abaixo, esses dados demonstram que existe uma forte predisposição dos consumidores em adotar uma solução como a proposta, desde que sejam garantidas condições adequadas de qualidade e um benefício financeiro perceptível.

Figura 25 - Gráfico de aceitação de alimentos próximos ao vencimento.

Você compraria alimentos próximos da validade se estivessem em boas condições e com desconto?

35 respostas



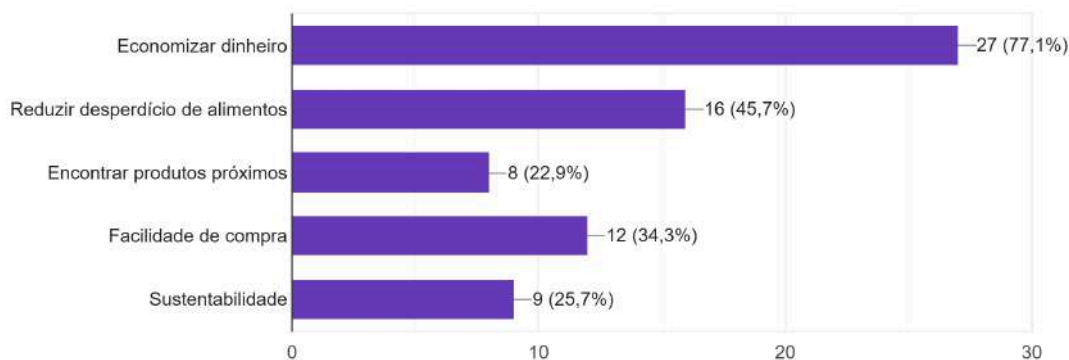
Fonte: Autores (2026).

Em relação aos motivos para utilizar a plataforma, os dois fatores mais relevantes foram economizar dinheiro, apontado por 77,1% dos respondentes, e reduzir o desperdício de alimentos, citado por 45,7%. Conforme apresentado na Figura 26 abaixo, esses resultados são especialmente importantes, pois correspondem justamente às duas principais metas do SecondBite: oferecer produtos com preços mais acessíveis aos consumidores e, ao mesmo tempo, contribuir para o aproveitamento de alimentos que poderiam ser descartados.

Figura 26 - Gráfico de adesão à plataforma SecondBite.

Quais seriam os seus principais motivos para usar uma plataforma como o SecondBite?

35 respostas



Fonte: Autores (2026).

As respostas abertas também mostraram que a confiança no aplicativo estaria relacionada principalmente à transparência das informações, à disponibilização de fotos reais e recentes dos produtos e às avaliações de outros usuários. Esses elementos, somados aos dados quantitativos apresentados, indicam que a proposta apresenta potencial de aceitação por parte do público-alvo.

Considerando a hipótese de que consumidores demonstraram interesse em adquirir alimentos com desconto e que feirantes estariam dispostos a utilizar a plataforma como alternativa para reduzir perdas e aumentar suas vendas, o SecondBite mostra-se viável tanto do ponto de vista tecnológico quanto mercadológico. Essa receptividade reforça a relevância da solução como instrumento de apoio ao consumo consciente e à geração de valor a partir de produtos que, de outra forma, seriam descartados.

Conclui-se, portanto, que o SecondBite atende aos objetivos propostos e evidencia como sistemas de informação podem contribuir de forma concreta para a redução do desperdício de alimentos, para o fortalecimento da sustentabilidade e para a ampliação do acesso da população a alimentos de qualidade por preços mais acessíveis. Além de beneficiar consumidores e feirantes, a plataforma também pode colaborar com a diminuição dos impactos ambientais associados ao descarte inadequado de resíduos orgânicos, consolidando-se como uma solução inovadora e socialmente relevante.

4.1 Perspectivas futuras

Como trabalhos futuros, pretende-se ampliar as funcionalidades da plataforma, incorporando meios de pagamento online, como Pix e cartões de crédito, bem como sistemas de notificação em tempo real para alertar consumidores sobre promoções e produtos próximos do vencimento. Também poderá ser implementado um mecanismo de recomendação baseado no histórico de compras e preferências dos usuários, oferecendo uma experiência mais personalizada.

Adicionalmente, recomenda-se a realização de um projeto piloto em feiras livres da cidade de São Paulo, com o objetivo de validar a solução em ambiente real

e coletar *feedback* de feirantes e consumidores. Caso a aceitação observada se confirme na prática, o SecondBite poderá evoluir para uma plataforma de maior alcance, contribuindo significativamente para a transformação digital do setor alimentício e para a construção de uma cadeia de consumo mais sustentável e eficiente.

REFERÊNCIAS BIBLIOGRÁFICAS

ALIANÇA PELA ALIMENTAÇÃO ADEQUADA E SAUDÁVEL. São Paulo desperdiça 33 mil toneladas de alimentos por ano em feiras livres. *Gazeta de S. Paulo*, São Paulo, 6 dez. 2024. Disponível em: <https://www.gazetasp.com.br/cotidiano/sao-paulo-desperdica-33-mil-toneladas-alimentos-por-ano/1147397/>. Acesso em: 9 abr. 2026.

ARAÚJO, G. P. de; CAPELLA, A. P.; SCHNEIDER, F. Coalizões contra o desperdício de alimentos. *Portal Embrapa – Embrapa Alimentos e Territórios*, Maceió, 26 abr. 2024. Disponível em: <https://www.embrapa.br/busca-de-noticias/-/noticia/88791018/coalizoes-contra-o-desperdicio-de-alimentos>. Acesso em: 9 abr. 2026.

BASS, L.; CLEMENTS, P.; KAZMAN, R. *Software Architecture in Practice*. 4. ed. Boston: Addison-Wesley, 2021.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *UML: guia do usuário*. 2. ed. Rio de Janeiro: Elsevier, 2006.

ELMASRI, R.; NAVATHE, S. B. *Sistemas de Banco de Dados*. 6. ed. São Paulo: Pearson, 2011.

LARMAN, C. *Utilizando UML e Padrões: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo*. 3. ed. Porto Alegre: Bookman, 2007.

PNUMA – Programa das Nações Unidas para o Meio Ambiente. *Food Waste Index Report 2024*. Nairobi: PNUMA, 2024.

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

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

UNESP – Universidade Estadual Paulista. Desperdício de alimentos em feiras livres de São Paulo é tema de estudo pioneiro da Unesp. *Jornal da Unesp*, São Paulo, 27 jul. 2022. Disponível em: <https://jornal.unesp.br/2022/07/27/desperdicio-de-alimentos-em-feiras-livres-de-sao-paulo-e-tema-de-estudo-pioneiro-da-unesp/>. Acesso em: 9 abr. 2026.