UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

RIAN CUSTÓDIO DE SOUZA BELTRÃO

FASTFRETE: UMA APLICAÇÃO WEB PARA SOLICITAÇÃO E OFERTA DE FRETES URBANOS DE PEQUENO PORTE

GUARAPUAVA

2025

RIAN CUSTÓDIO DE SOUZA BELTRÃO

FASTFRETE: UMA APLICAÇÃO WEB PARA SOLICITAÇÃO E OFERTA DE FRETES URBANOS DE PEQUENO PORTE

FastFrete: A Web Application for Requesting and Offering Small Urban Freights

Projeto de Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Roni Fabio Banaszewski

Coorientador: Marcelo Marcos Vichar Junior

GUARAPUAVA

2025

4.0 Internacional

Esta licença permite compartilhamento, remixe, adaptação e criação a partir do trabalho, mesmo para fins comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

RESUMO

Este trabalho apresenta o desenvolvimento da FastFrete, uma aplicação web voltada para a realização de fretes urbanos de pequeno porte. A plataforma tem como propósito conectar clientes a fretistas locais, oferecendo um ambiente digital estruturado que substitui processos informais e pouco organizados de contratação de serviços de transporte. O sistema foi concebido inicialmente como um Produto Mínimo Viável (MVP), implementando apenas as funcionalidades essenciais para validar o modelo proposto. Essa abordagem permite avaliar a aceitação da solução, coletar feedback dos usuários e estabelecer uma base tecnológica sólida para futuras expansões. Entre as possíveis evoluções incluem-se a integração de pagamentos digitais, recursos de rastreamento de cargas e mecanismos de geolocalização em tempo real, ampliando a eficiência e a confiabilidade do serviço.

Palavras-chave: fretes urbanos; aplicação web; mvp; transporte de pequeno porte; possiveis evoluções.

ABSTRACT

This work presents the development of FastFrete, a web application focused on small-scale urban freight services. The platform aims to connect clients with local freight carriers, offering a structured digital environment that replaces informal and poorly organized processes for contracting transportation services. The system was initially conceived as a Minimum Viable Product (MVP), implementing only the essential functionalities to validate the proposed model. This approach allows for evaluating the solution's acceptance, collecting user feedback, and establishing a solid technological foundation for future expansions. Possible improvements include the integration of digital payments, cargo tracking features, and real-time geolocation mechanisms, increasing the efficiency and reliability of the service.

Keywords: urban freight; mvp; web aplication; small-scale; possible improvements.

LISTA DE FIGURAS

Figura 1 – Paleta de Cores do Sistema **27**
Figura 2 – Logotipo **27**
Figura 3 – Landing Page **28**
Figura 4 – Tela de login **29**
Figura 5 – Formulário de criação de anúncio **30**
Figura 6 – Diagrama do banco de dados **32**

LISTA DE ABREVIATURAS E SIGLAS

Siglas

ANTT	Agência Nacional de Transportes Terrestres
DBML	Database Markup Language - *Linguagem de Marcação de Banco de Dados*
CNT	Confederação Nacional do Transporte
MVP	Minimum Viable Product - *Produto Mínimo Viável*
NPM	Node Package Manager
ORM	Object Relational Mapping - *Mapeamento Objeto-Relacional*
PIB	Produto Interno Bruto
RNTRC	Registro Nacional de Transportadores Rodoviários de Cargas
TAC	Transportadores Autônomos de Carga

SUMÁRIO

1 INTRODUÇÃO 7

1.1 Objetivos 8

1.1.1 Objetivo geral 8

1.1.2 Objetivos específicos 8

2 TRABALHOS RELACIONADOS 9

2.1 Fretebras 9

2.2 TruckPad 9

2.3 SuperFrete 9

2.4 Considerações 10

3 REFERENCIAL TEÓRICO 11

3.1 Processos de desenvolvimento 11

3.1.1 Scrum 11

3.1.2 Kanban 11

3.1.3 MoSCoW 12

3.1.4 Histórias de Usuário 12

3.1.5 Test-Driven Development (TDD) 12

3.2 Ferramentas de desenvolvimento 13

3.2.1 Git e GitFlow 13

3.2.2 GitHub e GitHub Projects 13

3.2.3 Estratégia de monorepo 13

3.2.4 Docker e Docker Compose 14

3.3 Tecnologias do lado do cliente 14

3.3.1 TypeScript 14

3.3.2 Vue.js 14

3.3.3 Nuxt 15

3.3.4 Tailwind e Flowbite 15

3.4 Tecnologias do lado do servidor 15

3.4.1 Node.js 15

3.4.2 NestJS 15

3.4.3 Prisma ORM 16

3.5 Ferramentas de prototipação e modelagem 16
3.5.1 Figma 16
3.5.2 Dbdiagram 16
4 MATERIAIS E MÉTODOS 17
4.1 Materiais 17
4.2 Métodos 18
5 ANÁLISE DO SISTEMA 20
5.1 Visão geral do sistema 20
5.2 Perfis de usuários 20
5.3 Funcionalidades essenciais 21
5.4 Público-alvo 25
6 PROJETO DO SISTEMA 26
6.1 Design System 26
6.2 Prototipação das telas 27
6.3 Modelagem do banco de dados 30
7 CONCLUSÃO 33
REFERÊNCIAS 34
APÊNDICE A FUNCIONALIDADES CLASSIFICADAS COMO SHOULD HAVE, COULD HAVE E WON'T HAVE (MOSCOW) 37
A.1 Funcionalidades Should Have (Deveria Ter) 37
A.2 Funcionalidades Could Have (Poderia Ter) 37
A.3 Funcionalidades Won't Have This Time (Não Terá Desta Vez) 38

1 INTRODUÇÃO

O setor de transporte de cargas desempenha um papel fundamental na economia brasileira, representando cerca de 3,1% do Produto Interno Bruto (PIB) em 2025, segundo a Confederação Nacional do Transporte (CNT) (CNT, 2025). Além da relevância nacional, os fretes urbanos e de pequeno porte são essenciais para atender demandas cotidianas de pessoas físicas e pequenos negócios, desde entregas rápidas até mudanças residenciais de curta distância.

A predominância de Transportadores Autônomos de Carga (TAC) é uma característica marcante do setor. De acordo com dados da Agência Nacional de Transportes Terrestres (ANTT), dos 833 mil transportadores registrados ativos no Registro Nacional de Transportadores Rodoviários de Cargas (RNTRC), modal rodoviário, 610 mil são autônomos (ANTT, 2025). Essa parcela predominante de profissionais autônomos, normalmente é acompanhada pela informalidade na contratação, fazendo com que os profissionais tenham a necessidade de sempre buscar novos fretes.

A atuação desses profissionais, é apresentada em diferentes modalidades que atendem a uma variedade de necessidades logísticas. Para encomendas rápidas e de baixo volume, os pequenos fretes urbanos são realizados por motoboys ou veículos leves, como furgões e vans. Já as mudanças e cargas médias são feitas por caminhonetes e caminhões, ideais para o transporte de mercadorias e pequenas mudanças entre municípios próximos. O setor também abrange serviços específicos, como o transporte de veículos em guincho ou cargas frágeis, que exigem maior cuidado e, muitas vezes, especialização do transportador.

Apesar da variedade de opções disponíveis, a contratação de fretes urbanos ainda enfrenta diversos desafios para o usuário final. Entre eles, destaca-se a dificuldade de acesso, já que a maioria das pessoas não possui contatos organizados de transportadores e depende de indicações informais ou de buscas em redes sociais. Essa informalidade leva à disponibilidade incerta, pois o fretista pode já estar ocupado. Soma-se a isso o desajuste entre demanda e oferta, em que um cliente com uma carga pequena pode ter acesso apenas a um veículo grande. Por fim, há a perda de oportunidades de economia, pois a falta de comunicação impede que transportadores que retornam de outra cidade ofereçam fretes mais baratos.

Essas dificuldades demonstram que, embora o serviço de frete urbano seja amplamente necessário, não existem muitas soluções simples e estruturadas para aproximar clientes e transportadores locais. Nesse sentido, seria interessante a criação de uma aplicação digital que centralize as informações, facilite o contato entre as partes e torne a contratação mais ágil, confiável e acessível.

1.1 Objetivos

Nesta seção serão apresentados o objetivo geral e os objetivos específicos do presente trabalho.

1.1.1 Objetivo geral

Desenvolver uma aplicação web para transportes urbanos de pequeno porte, que conecte clientes e fretistas, visando facilitar a solicitação e oferta de serviços de transporte de forma simples, rápida e confiável.

1.1.2 Objetivos específicos

- **Implementar o sistema de autenticação e gerenciamento de usuários**, contemplando cadastro, login, recuperação de senha e controle de perfis (cliente, fretista e administrador).
- **Desenvolver funcionalidades para publicação e gerenciamento de fretes**, incluindo cadastro de anúncios, upload de fotos, definição de data ou janela de disponibilidade e acompanhamento pelo cliente.
- **Disponibilizar ferramentas para fretistas visualizarem e se candidatarem a fretes**, com filtros adequados, detalhes do anúncio e envio de propostas.
- **Oferecer meios de interação entre as partes**, permitindo contato direto para negociação e adotando pagamento externo à plataforma.
- **Implementar um sistema de avaliações e um painel administrativo**, possibilitando gerenciar usuários, fretes, avaliações e suspensões quando necessário.

2 TRABALHOS RELACIONADOS

O cenário atual do transporte de cargas é marcado pelo surgimento de plataformas digitais e aplicativos de frete que atuam como intermediadores entre empresas e transportadores. Essas soluções representam um avanço na modernização do setor, buscando otimizar a busca por cargas e a comunicação entre as partes. No entanto, cada uma apresenta pontos fortes e fracos em sua abordagem.

2.1 Fretebras

A plataforma do Fretebras é uma solução online e um aplicativo móvel que facilita a conexão direta entre empresas que precisam transportar cargas e uma vasta rede de caminhoneiros e frotistas autônomos. O funcionamento é simples: as empresas publicam suas cargas e fretes, e os transportadores buscam por fretes compatíveis com seus veículos. A negociação ocorre diretamente entre as partes, sem a necessidade de intermediários. O site oficial da empresa destaca que a plataforma tem mais de 4 milhões de downloads e é a preferida dos caminhoneiros, com alta avaliação. A Fretebras processa 20 mil cargas por dia, com mais de 78 mil fretes ativos e 900 mil motoristas cadastrados, ajudando-os a encontrar cargas para os trajetos de ida e de volta, evitando que viajem com o caminhão vazio (FRETEBRAS, 2025).

2.2 TruckPad

Uma outra solução bastante utilizada é o Truckpad, um aplicativo de frete que atua como uma plataforma digital que conecta transportadores autônomos a transportadoras de forma direta. Sua principal proposta é modernizar a logística do setor, eliminando a figura do intermediário e otimizando o processo de contratação. As vantagens da plataforma são claras para ambas as partes: para o caminhoneiro, ela oferece um ambiente para buscar fretes e negociar diretamente, o que aumenta a rentabilidade e resolve o problema do retorno com o veículo vazio. Já para a transpotadora, o aplicativo proporciona maior transparência na negociação e acesso a uma ampla rede de motoristas, agilizando o processo de contratação e permitindo o rastreamento da carga em tempo real (TRUCKPAD, 2025).

2.3 SuperFrete

A SuperFrete é uma plataforma logística brasileira voltada para o nicho de envio de encomendas no e-commerce, funcionando como ponte entre lojistas virtuais e transportadoras. O público-alvo da SuperFrete são pequenos e médios lojistas, especialmente aqueles que operam por meio de redes sociais ou plataformas de e-commerce, e que buscam escalabilidade

logística sem depender de grandes contratos ou volumes elevados. A SuperFrete se posiciona como uma solução estratégica e tecnologicamente sofisticada para lojistas de pequeno e médio porte que desejam profissionalizar sua logística de entrega, oferecendo economia e automação (SUPERFRETE, 2025).

2.4 Considerações

As plataformas Fretebras, TruckPad e SuperFrete são referências no setor de transportes, oferecendo funcionalidades robustas e evidenciando a viabilidade de otimização logística, como a promoção do frete de retorno. No entanto, a principal limitação identificada nesses modelos reside em seu foco predominante no mercado B2B (empresa para empresa). A ausência de soluções voltadas para o consumidor final e para o transporte de cargas de pequeno e médio porte cria uma lacuna significativa no mercado. Essa necessidade de atender a demandas diretas de clientes justifica a criação de uma nova plataforma.

3 REFERENCIAL TEÓRICO

O desenvolvimento da aplicação web *FastFrete*, voltada à solicitação e oferta de fretes urbanos de pequeno porte, fundamenta-se em princípios da engenharia de software e na adoção de metodologias ágeis e tecnologias modernas de desenvolvimento web. Este capítulo apresenta o conjunto de referenciais teóricos e técnicos que sustentam o projeto, abordando desde as metodologias de desenvolvimento e controle de versão até as tecnologias que compõem a pilha de implementação. O objetivo é oferecer a base conceitual necessária para compreender as decisões técnicas e metodológicas que serão detalhadas nos capítulos de *Materiais e Métodos* e *Projeto do Sistema*.

3.1 Processos de desenvolvimento

Antes de construir o sistema propriamente dito, é essencial compreender as metodologias que orientam o ciclo de desenvolvimento. Os métodos ágeis surgem como alternativas flexíveis aos modelos tradicionais, permitindo maior adaptação às mudanças e entregas contínuas de valor ao usuário. Nesta seção, são apresentados os principais frameworks e técnicas utilizados no gerenciamento do projeto, incluindo *Scrum*, *Kanban* e o método de priorização *MoSCoW*, que juntos formam a base metodológica do *FastFrete*.

3.1.1 Scrum

O *Scrum* é um framework ágil amplamente utilizado para o gerenciamento de projetos de software, baseado em ciclos curtos e iterativos denominados *sprints*. Cada sprint resulta em um incremento funcional do produto, permitindo inspeção contínua e adaptação rápida às mudanças. O método define papéis específicos — *Product Owner*, *Scrum Master* e *Development Team* —, bem como eventos formais como *Sprint Planning*, *Daily Scrum*, *Sprint Review* e *Retrospective*, que asseguram transparência e melhoria contínua (SCHWABER; SUTHERLAND, 2020).

3.1.2 Kanban

O Kanban é uma abordagem visual de gestão de fluxo de trabalho, que organiza as tarefas em colunas representando diferentes estágios de execução, como “A Fazer”, “Em Progresso” e “Concluído”. Essa metodologia permite visualizar o progresso do projeto, identificar gargalos e promover um fluxo contínuo de entrega de valor (ANDERSON, 2010). A aplicação do Kanban em projetos de software facilita a priorização das atividades e a transparência en-

tre os membros da equipe, especialmente quando integrado a ferramentas de versionamento e controle de tarefas.

3.1.3 MoSCoW

A técnica *MoSCoW* é um método de priorização de requisitos utilizado em projetos de software para organizar a importância das funcionalidades de forma clara e hierárquica. O método classifica os requisitos em quatro categorias: **Must Have**, que representam itens indispensáveis; **Should Have**, itens importantes, mas não essenciais; **Could Have**, funcionalidades desejáveis; e **Won't Have This Time**, que abrangem requisitos que não serão contemplados na versão inicial (DSDM Consortium, 2014).

Essa abordagem contribui para que equipes de desenvolvimento mantenham foco no que é essencial, evitando o fenômeno conhecido como *scope creep*, caracterizado pela expansão não planejada do escopo do projeto. Ao estruturar claramente o que deve ou não ser implementado em cada fase, o MoSCoW facilita o planejamento, a tomada de decisões e o gerenciamento eficiente de recursos durante o desenvolvimento de sistemas.

3.1.4 Histórias de Usuário

As histórias de usuário são um artefato amplamente utilizado em metodologias ágeis, como Scrum e Extreme Programming (XP), para representar requisitos de forma simples e centrada no usuário. Segundo Cohn (2004), uma história de usuário descreve uma funcionalidade de maneira breve, enfatizando o valor que ela entrega ao usuário final. Esse formato facilita a comunicação entre stakeholders, reduz ambiguidades e favorece a evolução incremental do sistema.

Tradicionalmente, as histórias são expressas por meio da estrutura: **COMO** [tipo de usuário], **QUERO** [ação ou necessidade], **PARA** [benefício ou objetivo]. Associados às histórias estão os *critérios de aceitação*, que especificam as condições que devem ser atendidas para que a funcionalidade seja considerada concluída. Esses critérios servem como base para validação, testes e alinhamento de expectativas entre usuário e equipe.

3.1.5 Test-Driven Development (TDD)

O *Test-Driven Development* (TDD), ou Desenvolvimento Orientado por Testes, é uma prática de engenharia de software proposta por Kent Beck no final da década de 1990, no contexto da metodologia *Extreme Programming* (XP) (BECK, 2003). O TDD baseia-se na criação de testes automatizados antes da implementação do código funcional, promovendo um ciclo de desenvolvimento iterativo e incremental conhecido como *Red–Green–Refactor*.

Nesse ciclo, o desenvolvedor inicialmente escreve um teste que falha (*Red*), em seguida implementa o código mínimo necessário para que o teste seja aprovado (*Green*), e, por fim, realiza a refatoração do código garantindo que os testes continuem passando (*Refactor*). Essa prática reforça a qualidade do software, melhora a legibilidade do código e reduz a incidência de erros regressivos. No contexto de metodologias ágeis, o TDD complementa frameworks como o Scrum, permitindo a integração contínua e a validação incremental das funcionalidades.

3.2 Ferramentas de desenvolvimento

O desenvolvimento colaborativo de software exige ferramentas que garantam versionamento, rastreabilidade e integração contínua entre as partes do sistema. Nesta seção, são apresentadas as ferramentas e práticas empregadas para o controle do ciclo de desenvolvimento, com destaque para o uso do Git, GitHub e Docker, além da estratégia de organização de código em *monorepo*, que contribui para a manutenção e escalabilidade do projeto.

3.2.1 Git e GitFlow

O Git é um sistema de controle de versão distribuído utilizado para registrar, gerenciar e recuperar o histórico de alterações de um projeto de software. Sua arquitetura distribuída permite colaboração simultânea entre desenvolvedores, rastreabilidade de mudanças e versionamento seguro, mesmo em ambientes descentralizados (CHACON; STRAUB, 2014).

Entre as estratégias de ramificação, destaca-se o modelo *GitFlow*, que organiza o desenvolvimento em ramos específicos, como *master*, *develop* e *feature branches*. Esse fluxo facilita o gerenciamento de versões, o controle de lançamentos e a separação clara entre desenvolvimento contínuo e entregas estáveis (DRIESSEN, 2010).

3.2.2 GitHub e GitHub Projects

O GitHub é uma plataforma de hospedagem de código baseada em Git, que oferece recursos para colaboração, revisão de código e integração contínua. Seu módulo *GitHub Projects* possibilita o gerenciamento de tarefas em formato de quadro Kanban, com automação de movimentação de cartões conforme o status das *issues*. Essa integração contribui para a organização e transparência do processo de desenvolvimento (GITHUB, 2025).

3.2.3 Estratégia de monorepo

A organização dos repositórios pode seguir diferentes modelos, sendo os dois mais comuns o *multirepo* e o *monorepo*.

No modelo *monorepo*, múltiplos componentes, serviços ou módulos são armazenados em um único repositório, permitindo centralização de dependências, padronização de ferramentas e versionamento unificado. Essa abordagem é frequentemente adotada por equipes que buscam integração consistente entre partes relacionadas de um sistema (FOWLER, 2019).

Já o modelo *multirepo* separa cada componente em repositórios independentes, favorecendo autonomia entre equipes, ciclos de versionamento distintos e modularidade mais rígida. A escolha entre essas estratégias depende do tamanho do time, do nível de acoplamento desejado e das características do produto em desenvolvimento.

3.2.4 Docker e Docker Compose

O Docker é uma tecnologia de virtualização leve baseada em contêineres, que permite empacotar aplicações e suas dependências em ambientes isolados e portáteis. Com o *Docker Compose*, é possível definir múltiplos serviços em um único arquivo de configuração (`docker-compose.yml`), automatizando a inicialização do ambiente completo (DOCKER, 2025). Essa combinação garante consistência entre os ambientes de desenvolvimento, teste e produção, reduzindo erros de configuração e aumentando a reprodutibilidade.

3.3 Tecnologias do lado do cliente

O desenvolvimento do lado do cliente (*frontend*) é responsável pela interface e experiência do usuário. Nesta seção, são descritas as tecnologias que compõem a camada de apresentação do sistema, abordando desde a linguagem principal até os frameworks e bibliotecas de estilização que garantem interatividade, responsividade e consistência visual.

3.3.1 TypeScript

O TypeScript é uma linguagem de programação baseada em JavaScript que adiciona tipagem estática opcional e recursos de orientação a objetos. Essa tipagem facilita a detecção precoce de erros e melhora a legibilidade e a manutenção do código. Por ser compatível com o ecossistema JavaScript e amplamente adotado no desenvolvimento de aplicações web modernas, o TypeScript contribui para a criação de sistemas mais estáveis, escaláveis e previsíveis (MICROSOFT, 2025).

3.3.2 Vue.js

O Vue.js é um framework progressivo para construção de interfaces de usuário reativas. Sua arquitetura baseada em componentes favorece a modularidade, a reutilização de código e

o desenvolvimento de interfaces dinâmicas e responsivas. A curva de aprendizado acessível e a documentação detalhada tornam o Vue uma das principais escolhas para aplicações modernas de página única (*Single Page Applications*) (YOU; TEAM, 2025).

3.3.3 Nuxt

O Nuxt é um framework construído sobre o Vue.js que oferece suporte a renderização do lado do servidor (SSR) e geração de sites estáticos. Ele adota uma estrutura modular que organiza rotas, páginas e middlewares de forma automatizada, melhorando o desempenho e o SEO da aplicação (LABS, 2025). Sua arquitetura opinativa contribui para padronizar o desenvolvimento do *frontend*, reduzindo a complexidade de configuração inicial.

3.3.4 Tailwind e Flowbite

O Tailwind CSS é um framework utilitário de estilos que fornece classes pré-definidas diretamente aplicáveis ao HTML, permitindo a construção rápida de interfaces responsivas e consistentes (TEAM, 2025b). Complementarmente, o Flowbite disponibiliza uma biblioteca de componentes de interface — como botões, tabelas e modais — baseados no Tailwind, acelerando o processo de prototipação e garantindo uniformidade visual (THEMESBERG, 2025).

3.4 Tecnologias do lado do servidor

O lado do servidor (*backend*) é responsável pela lógica de negócio, controle de acesso e persistência dos dados. Nesta seção, são apresentadas as tecnologias que compõem essa camada, explicando como cada uma contribui para a robustez, escalabilidade e integração do sistema.

3.4.1 Node.js

O Node.js é um ambiente de execução JavaScript orientado a eventos e não bloqueante, projetado para aplicações escaláveis e de alto desempenho. Ele permite que o mesmo ecossistema de linguagem seja utilizado tanto no lado do cliente quanto no servidor, simplificando a integração entre camadas (FOUNDATION, 2025).

3.4.2 NestJS

O NestJS é um framework para Node.js escrito em TypeScript, inspirado em arquiteturas modulares e orientadas a objetos. Ele utiliza princípios de injeção de dependência e programa-

ção reativa, oferecendo uma estrutura organizada para construção de APIs escaláveis e de fácil manutenção (MYŚLIWIEC; TEAM, 2025).

3.4.3 Prisma ORM

O Prisma é um ORM (*Object-Relational Mapping*) moderno que facilita a interação entre a aplicação e o banco de dados. Ele gera modelos fortemente tipados e oferece ferramentas de migração e inspeção de esquema, reduzindo a incidência de erros e aumentando a produtividade do desenvolvedor (TEAM, 2025a). Sua integração nativa com o NestJS reforça a consistência entre a lógica de negócio e a camada de persistência.

3.5 Ferramentas de prototipação e modelagem

O design de interfaces e a modelagem do banco de dados constituem etapas fundamentais no desenvolvimento de sistemas, pois garantem que a aplicação seja funcional, intuitiva e tecnicamente consistente. Antes da implementação, é essencial validar fluxos, usabilidade e aspectos estéticos por meio de ferramentas de prototipação colaborativa, bem como antecipar os recursos e estruturas necessários no banco de dados. Nesta seção, são apresentadas as ferramentas selecionadas para apoiar essa fase de planejamento.

3.5.1 Figma

O Figma é uma plataforma colaborativa de design de interfaces e prototipação, amplamente utilizada no desenvolvimento de sistemas web e mobile. Ele permite criar componentes reutilizáveis, definir padrões visuais e testar fluxos de navegação de forma interativa. Por possibilitar o trabalho simultâneo entre diferentes membros da equipe, o Figma contribui para a padronização visual e a eficiência na validação das interfaces do *FastFrete* (FIGMA, 2025).

3.5.2 Dbdiagram

O dbdiagram é uma ferramenta online voltada para a modelagem de bancos de dados por meio de diagramas de entidade-relacionamento. Ele permite definir tabelas, colunas e relacionamentos utilizando uma linguagem chamada Database Markup Language - *Linguagem de Marcação de Banco de Dados* (DBML), oferecendo uma maneira simples e estruturada de planejar a arquitetura do banco de dados. Por possibilitar a visualização clara dos esquemas e o compartilhamento com a equipe (DBDIAGRAM, 2025), o dbdiagram contribui para a organização e eficiência na etapa de modelagem do *FastFrete*.

4 MATERIAIS E MÉTODOS

Este capítulo apresenta os recursos, ferramentas e procedimentos metodológicos que serão utilizados na concepção e planejamento do sistema *FastFrete*. A seção de *Materiais* descreve as tecnologias, frameworks e ferramentas que comporão a base do projeto, enquanto a seção de *Métodos* expõe o processo de organização e acompanhamento do desenvolvimento, fundamentado em práticas ágeis e adaptadas à realidade de um projeto individual.

4.1 Materiais

Os materiais previstos para o desenvolvimento do *FastFrete* englobam linguagens de programação, frameworks, bibliotecas e ferramentas de apoio. Esses recursos foram selecionados considerando critérios de compatibilidade, escalabilidade, facilidade de manutenção e adequação às demandas de uma aplicação web moderna.

- **Linguagens de programação e frameworks**
  - TypeScript como linguagem principal, aplicada tanto no frontend quanto no backend;
  - Vue.js e Nuxt no desenvolvimento do frontend, responsáveis pela construção das interfaces e pelos fluxos de interação com o usuário;
  - NestJS no backend, oferecendo uma arquitetura modular, padronizada e com suporte nativo ao TypeScript;
  - Node.js como ambiente de execução do backend e para o gerenciamento de pacotes por meio do Node Package Manager (NPM);
  - Prisma como ORM, utilizado para a comunicação entre a aplicação e o banco de dados;
  - MySQL como sistema gerenciador de banco de dados, escolhido por sua robustez, estabilidade e ampla adoção no mercado.
- **Ambiente de desenvolvimento**
  - Sistema operacional Ubuntu;
  - Editor de código WebStorm;
  - Versionamento de código realizado com Git, utilizando repositório hospedado no GitHub;
  - Gerenciamento de dependências por meio do NPM.
- **Infraestrutura e execução**

  - Utilização de Docker e Docker Compose para padronização do ambiente de desenvolvimento e execução da aplicação;
  - Comunicação entre as camadas do sistema por meio de uma API REST, utilizando o formato JSON para troca de dados.

- **Ferramentas de apoio**
  - Figma para prototipação das interfaces e validação dos fluxos de navegação;
  - dbdiagram.io para a modelagem de dados e elaboração dos diagramas entidade-relacionamento (DER).

4.2 Métodos

Os métodos a serem empregados no projeto baseiam-se em práticas ágeis de desenvolvimento de software, com foco em organização, acompanhamento e entrega incremental. A metodologia será adaptada à realidade de um projeto conduzido individualmente, preservando os princípios de planejamento e revisão contínua.

- **Metodologia de desenvolvimento**

  O processo de desenvolvimento será conduzido com base no framework *Scrum*, em uma versão adaptada para um único desenvolvedor. Nessa adaptação, o aluno atuará como desenvolvedor principal, enquanto o professor orientador desempenhará os papéis de *Scrum Master* e *Product Owner*. Serão mantidos os elementos essenciais do método, como planejamento de sprints, acompanhamento das atividades e revisões periódicas, sem a necessidade de todos os rituais formais do Scrum.

- **Ciclos de trabalho**

  O trabalho será organizado em sprints com duração de aproximadamente duas semanas, possibilitando o acompanhamento contínuo da evolução do projeto e a revisão incremental de suas entregas. Cada sprint terá objetivos definidos a partir do *backlog do produto*, que reunirá as histórias de usuário elaboradas na etapa de análise do sistema. O controle das tarefas será feito por meio de quadros Kanban no GitHub Projects, com colunas que representarão o estado de cada atividade, como “A Fazer”, “Em Progresso”, “Em Revisão” e “Concluído”.

- **Priorização de funcionalidades**

  Para o planejamento do escopo inicial, será aplicada a técnica *MoSCoW*, que classifica os requisitos em quatro categorias: **Must Have**, **Should Have**, **Could Have** e **Won’t Have This Time**. Serão priorizados, nesta primeira etapa de concepção, os requisitos classificados como **Must Have**, que definirão o escopo do Minimum Viable Product

- *Produto Mínimo Viável* (MVP). Essa priorização permitirá concentrar esforços nas funcionalidades essenciais e reduzir o risco de expansão não controlada do escopo.

- **Estratégia de versionamento**

  O versionamento do projeto seguirá o modelo GitFlow, dividido em ramos específicos para desenvolvimento, funcionalidades e correções. Essa organização permitirá controlar o progresso, garantir estabilidade e suportar entregas incrementais.

- **Integração e validação**

  A etapa de testes incluirá validações funcionais dos fluxos principais e testes unitários baseados em TDD. A integração entre os módulos do sistema será verificada de modo incremental, acompanhando a evolução das funcionalidades. Ao final de cada sprint, será realizada uma revisão conjunta entre aluno e orientador, na qual serão avaliadas as entregas, o alinhamento com o backlog e os ajustes necessários para o ciclo seguinte.

- **Modelagem e prototipação**

  A modelagem dos dados será elaborada antes da implementação, empregando diagramas entidade-relacionamento (DER). Essa modelagem definirá entidades, atributos e relacionamentos que orientarão a construção do banco de dados. As interfaces do sistema serão prototipadas seguindo princípios de usabilidade, consistência e navegação clara. Os protótipos representarão os principais fluxos do sistema e serão validados antes da implementação.

5 ANÁLISE DO SISTEMA

Este trabalho consiste no desenvolvimento de uma aplicação web, denominada FastFrete, com foco em fretes urbanos de pequeno porte. O objetivo central é criar um ambiente digital que conecte clientes e fretistas locais, simplificando a contratação de serviços de transporte que, atualmente, são realizados de forma informal e pouco organizada.

O projeto será implementado inicialmente como um MVP, contemplando apenas as funcionalidades essenciais para validar a solução proposta. Dessa forma, será possível testar a aceitação do sistema, reunir feedback dos usuários e estabelecer uma base sólida para expansões futuras, como a integração de pagamentos digitais, rastreamento de cargas e geolocalização em tempo real. As demais funcionalidades poderão ser implementadas em desenvolvimentos posteriores.

5.1 Visão geral do sistema

O FastFrete organiza o processo de contratação de fretes urbanos de pequeno porte, conectando clientes e fretistas de maneira mais prática, transparente e segura do que os métodos informais atualmente utilizados.

No MVP, a solução será disponibilizada exclusivamente para o município de Guarapuava. De forma geral, o funcionamento do sistema envolve três etapas principais: (i) o cadastro de usuários; (ii) a publicação de fretes por parte dos clientes; e (iii) a visualização e candidatura dos fretistas às solicitações disponíveis.

A comunicação direta entre cliente e fretista ocorre apenas após o início de uma negociação ou seleção para execução do serviço. Ao final do frete, ambas as partes podem registrar avaliações, favorecendo a construção de reputação e aumentando a confiabilidade da plataforma.

5.2 Perfis de usuários

A plataforma contará com três perfis principais, cada um desempanhando funções específicas:

- **Cliente**: usuário responsável por solicitar fretes. Pode cadastrar anúncios, visualizar fretistas disponíveis, entrar em contato direto para negociação e avaliar o serviço após sua realização.
- **Fretista**: transportador autônomo que se cadastra no sistema informando seus dados e os veículos disponíveis. Pode visualizar as solicitações de frete publicadas e ser contatado diretamente por clientes interessados.

- **Administrador**: responsável por manter a organização da plataforma, gerenciando usuários, solicitações e avaliações. Possui autoridade para suspender usuários que descumprirem as regras de uso.

5.3 Funcionalidades essenciais

As funcionalidades descritas nesta seção correspondem aos requisitos classificados como *Must Have* pela técnica MoSCoW, compondo o MVP do sistema. As demais funcionalidades — classificadas como *Should Have*, *Could Have* e *Won't Have This Time* — encontram-se documentadas no Apêndice A, onde são apresentadas como possibilidades de expansão para versões futuras. As funcionalidades do MVP foram descritas utilizando o modelo de histórias de usuário. Para cada história de usuário apresentada, são definidos também os respectivos **critérios de aceitação**. A seguir, estão listadas as histórias descritas por ordem de prioridade, identificadas por um ID no formato **HUXX**, onde **XX** representa um número sequencial.

- **HU01 – Autenticação de Usuário:** *Como* cliente, fretista ou administrador, *quero* fazer login, logout e recuperar senha, *para* acessar as funcionalidades conforme meu papel.

  Critérios de Aceitação:

  1. Formulário com campos obrigatórios: e-mail e senha.
  2. Exibir mensagem de erro em caso de credenciais inválidas.
  3. Função "Esqueci minha senha" deve enviar link de redefinição por e-mail.
  4. Direcionar o usuário para sua área inicial após login.

- **HU02 – Cadastro de Cliente:** *Como* cliente, *quero* criar uma conta informando meus dados pessoais, *para* solicitar fretes.

  Critérios de Aceitação:

  1. Campos obrigatórios: nome, telefone, cidade, e-mail e senha.
  2. A cidade disponível deve ser apenas **Guarapuava**.
  3. Validar se o e-mail informado já está cadastrado.
  4. Exibir mensagem de confirmação após conclusão do cadastro.

- **HU03 – Cadastro de Fretista:** *Como* fretista, *quero* criar uma conta com meus dados pessoais, tipo de veículo e capacidade de carga, *para* receber solicitações de clientes.

  Critérios de Aceitação:

  1. Campos obrigatórios: nome, telefone, cidade, e-mail, senha, tipo de veículo e capacidade.

2. A cidade disponível deve ser apenas **Guarapuava**.
3. Possibilidade de cadastrar mais de um tipo de veículo.
4. Permitir atualização dos dados do perfil após o cadastro.

- **HU04 – Lista de Fretistas:** *Como* cliente, *quero* visualizar fretistas filtrando por tipo de veículo, *para* encontrar rapidamente quem pode atender meu frete.

  Critérios de Aceitação:

  1. Exibir nome, veículo, capacidade e contatos.
  2. Permitir filtro por categoria de veículo.
  3. Abrir o WhatsApp ao clicar no botão dedicado.

- **HU05 – Publicar Anúncio de Frete:** *Como* cliente, *quero* criar um anúncio com descrição e fotos, *para* que o fretista compreenda o serviço.

  Critérios de Aceitação:

  1. O anúncio deve solicitar os seguintes campos obrigatórios:
     - título;
     - descrição;
     - categoria de veículo;
     - endereço de origem (cidade, bairro, logradouro, número);
     - endereço de destino (cidade, bairro, logradouro, número).
  2. O cliente deve escolher entre dois formatos de data:
     a) **Data e horário exatos** (ex.: 30/11/2025 às 14:00);
     b) **Janela de disponibilidade**, podendo selecionar:
        - hoje;
        - amanhã;
        - esta semana;
        - este mês;
        - próximo fim de semana;
        - qualquer fim de semana;
        - data a combinar.
  3. O sistema deve permitir upload de até três fotos.
  4. Após o cadastro, o frete deve aparecer na tela “Meus Fretes”, exibindo:
     - status (ativo, concluído ou cancelado);
     - quantidade de fretistas candidatos.

- **HU06 – Contato Direto:** *Como* cliente, *quero* entrar em contato com o fretista por WhatsApp ou telefone, *para* negociar valores e detalhes.

  Critérios de Aceitação:

  1. Cada fretista deve exibir botões de “Ligar” e “WhatsApp”.
  2. Os aplicativos correspondentes devem abrir automaticamente.

- **HU07 – Pagamento Direto em Mãos:** *Como* cliente, *quero* pagar diretamente ao fretista, *para* evitar taxas e agilizar o processo.

  Critérios de Aceitação:

  1. Informar claramente que o pagamento ocorre fora da plataforma.
  2. O valor negociado não deve ser registrado no sistema.

- **HU08 – Avaliação Simples:** *Como* cliente ou fretista, *quero* avaliar a outra parte com nota e comentário, *para* orientar futuros usuários.

  Critérios de Aceitação:

  1. Solicitar avaliação após a finalização do frete.
  2. A nota (1 a 5) é obrigatória; comentário opcional.
  3. Exibir avaliações no perfil do usuário.

- **HU09 – Dashboard do Administrador:** *Como* administrador, *quero* gerenciar usuários, solicitações e avaliações, *para* manter a plataforma organizada.

  Critérios de Aceitação:

  1. Deve exibir lista de clientes e fretistas com filtros e campo de busca.
  2. Deve exibir lista de solicitações de frete com status (por exemplo, ativo, concluído, cancelado).
  3. Deve permitir visualizar as avaliações recebidas por cada usuário.

- **HU10 – Suspender Usuários Problemáticos:** *Como* administrador, *quero* suspender usuários que violem regras, *para* manter a confiabilidade da plataforma.

  Critérios de Aceitação:

  1. Permitir suspender e reativar contas.
  2. Usuários suspensos não podem acessar o sistema.
  3. Registrar o motivo da suspensão.

- **HU11 – Meus Fretes (Cliente):** *Como* cliente, *quero* visualizar uma lista dos fretes que cadastrei, *para* acompanhar meus anúncios e acessar seus detalhes.

  Critérios de Aceitação:

  1. A tela deve listar os fretes cadastrados pelo cliente logado.
  2. Os fretes devem ser organizados ao menos em “Ativos” e “Concluídos”, podendo os concluídos aparecer em uma seção ou página separada.
  3. Cada cartão de frete deve exibir informações principais, como título, categoria de veículo e status.
  4. Ao clicar em um frete, o sistema deve abrir a página de detalhes, onde o cliente pode visualizar os candidatos e, quando aplicável, sugestões de outros fretistas.
  5. O cartão do frete deve indicar, quando houver, a quantidade de fretistas candidatos.

- **HU12 – Lista de Fretes Disponíveis (Fretista):** *Como* fretista, *quero* visualizar fretes disponíveis cadastrados por clientes, *para* identificar oportunidades de serviço.

  Critérios de Aceitação:

  1. A tela inicial do fretista deve exibir uma lista de fretes disponíveis em formato de cartões.
  2. A lista deve ser filtrada, no mínimo, pela cidade associada ao fretista e pela categoria de veículo que ele possui.
  3. Cada cartão deve exibir informações básicas, como título, categoria de veículo e valor sugerido.
  4. Ao clicar em um cartão, o sistema deve abrir a página de detalhes do frete.

- **HU13 – Detalhes do Frete (Fretista):** *Como* fretista, *quero* visualizar os detalhes de um frete, *para* avaliar se desejo me candidatar.

  Critérios de Aceitação:

  1. A página de detalhes deve exibir a descrição completa do frete, categoria de veículo, data e horário.
  2. Quando houver, devem ser exibidas as fotos cadastradas pelo cliente.
  3. O endereço deve ser apresentado de forma parcial (por exemplo, bairro e rua sem número), sem expor dados sensíveis do cliente.
  4. Dados de contato do cliente (telefone, WhatsApp) não devem ser exibidos ao fretista nessa etapa.

- **HU14 – Candidatar-se a Frete:** *Como* fretista, *quero* me candidatar a um frete informando o veículo e o valor proposto, *para* disputar a prestação do serviço.

  Critérios de Aceitação:

  1. A partir da lista ou da página de detalhes de um frete, o fretista deve poder acionar a opção de candidatura.
  2. O fretista deve selecionar um dos veículos cadastrados em seu perfil para associar à candidatura.
  3. O fretista deve informar um valor proposto para o serviço.
  4. A candidatura deve ser registrada e ficar disponível para visualização pelo cliente na área de detalhes do frete.
  5. Dados de contato do cliente não devem ser exibidos ao fretista apenas pelo fato de se candidatar.

5.4 Público-alvo

O FastFrete é voltado principalmente para dois grupos:

- **Clientes finais**: pessoas físicas ou pequenos comerciantes que necessitam de transporte de itens como móveis, eletrodomésticos, pequenas mudanças ou entregas locais e não dispõem de uma rede de contatos confiável de fretistas.
- **Transportadores autônomos**: profissionais que atuam com veículos de pequeno e médio porte (motoboys, fiorinos, vans, caminhonetes, caminhões 3/4) e que desejam ampliar sua visibilidade e captar mais serviços na região de atuação.

O sistema também atende **administradores da plataforma**, responsáveis por supervisionar o uso e garantir que o ambiente se mantenha seguro e organizado.

6 PROJETO DO SISTEMA

A prototipação de interfaces, assim como a modelagem do banco de dados, constitui atividades essenciais no processo de desenvolvimento de sistemas, sendo normalmente realizadas durante a etapa de planejamento da aplicação. Diante de sua relevância para a definição da estrutura inicial do projeto, este capítulo apresenta os protótipos das interfaces desenvolvidas e o modelo de banco de dados elaborado para apoiar essa fase preparatória.

6.1 Design System

Antes da prototipação das telas, foi definido uma prévia do *design system* para padronizar os elementos visuais da aplicação *FastFrete*. Esse conjunto de diretrizes estabelece a identidade visual do projeto e orienta a construção dos protótipos, garantindo uniformidade estética, coerência entre componentes e facilidade de manutenção futura.

O design system contempla a paleta de cores, a tipografia aplicada no projeto, os padrões de espaçamento e arredondamento utilizados nas telas e os componentes reutilizáveis que servirão de base no desenvolvimento.

A paleta de cores, apresentada na Figura 1, inclui as seguintes cores:

- **Laranja Primário — #f58634**: utilizado em botões de ação principal e elementos de destaque.
- **Azul Principal — #1a4ddb**: aplicado em botões secundários, links e elementos informativos.
- **Vermelho Alerta — #d9534f**: reservado para mensagens de aviso ou erro.
- **Cinza Escuro (Texto) — #212529**: cor padrão para textos e títulos, garantindo legibilidade.
- **Fundo Creme — #fff9f5**: cor de fundo predominante das páginas.
- **Branco Puro — #ffffff**: utilizado em cards, inputs e áreas de conteúdo.

A tipografia adotada corresponde a uma fonte sem serifa moderna, da família *Inter*, reconhecida por sua alta legibilidade em interfaces digitais, distribuídos conforme a hierarquia textual entre títulos, botões e textos descritivos.

Para manter clareza e equilíbrio visual, foram definidos espaçamentos consistentes entre seções e componentes, adotando margens uniformes que facilitam a leitura e a navegação pelas telas. Os elementos também utilizam bordas arredondadas entre 12 px e 16 px, resultando em uma estética moderna e amigável.

Figura 1 – Paleta de Cores do Sistema

Fonte: Autoria própria (2025).

A construção dos componentes reutilizáveis adotará o *Flowbite* em conjunto com o *Tailwind CSS*, permitindo a criação de elementos padronizados e altamente configuráveis. Essa abordagem favorece a consistência visual, acelera o desenvolvimento das telas e facilita ajustes futuros, uma vez que modificações aplicadas aos componentes centrais podem ser automaticamente refletidas em toda a interface.

O logotipo, apresentado na Figura 2, utiliza as cores principais do projeto e incorpora um símbolo associado ao conceito de frete, reforçando a identidade visual da plataforma.

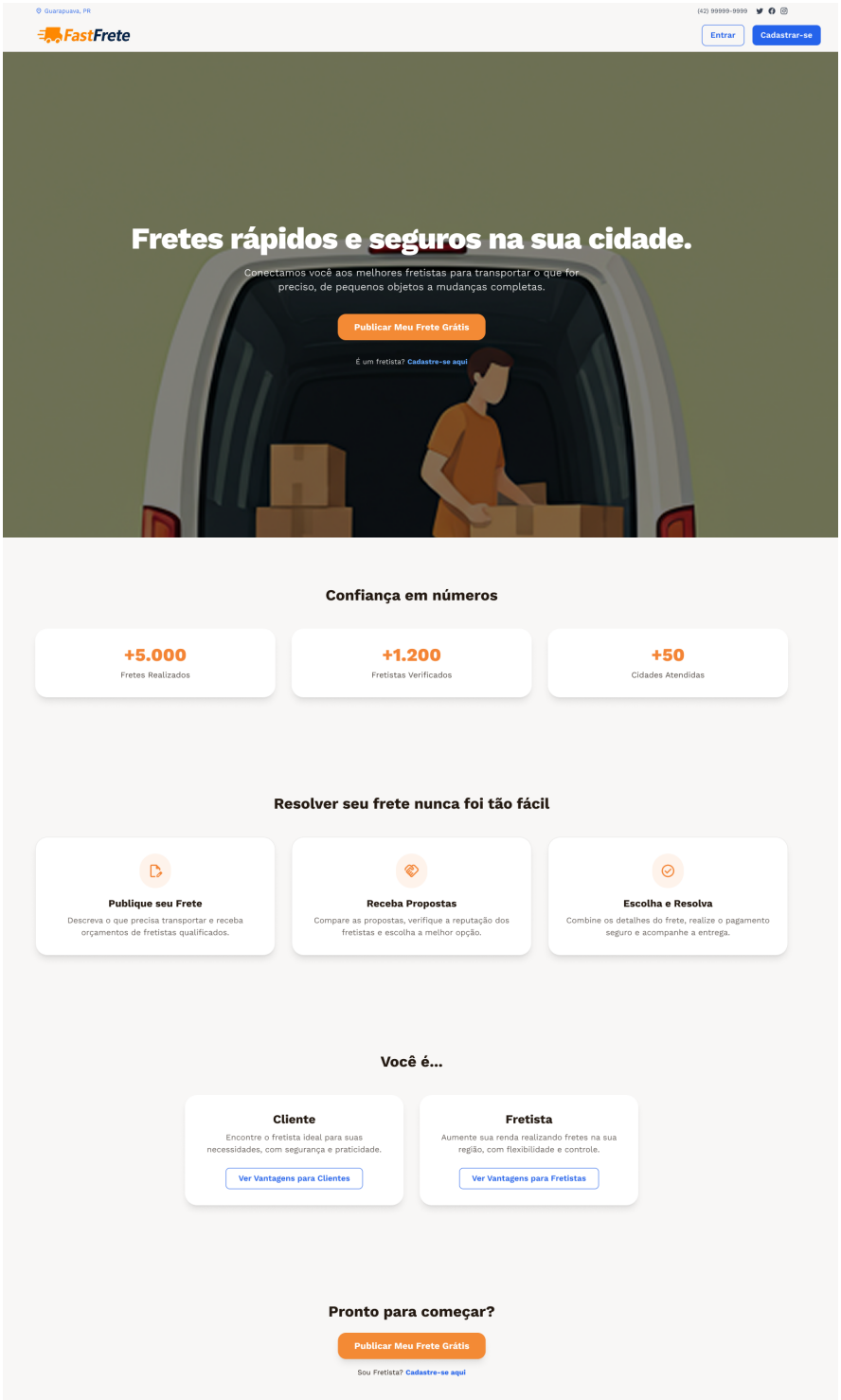
Figura 2 – Logotipo

Fonte: Autoria própria (2025).

6.2 Prototipação das telas

Com o objetivo de validar antecipadamente a organização visual e funcional das telas da aplicação web, foi adotado um design baseado em princípios de simplicidade, coerência e usabilidade, garantindo navegação intuitiva e reduzindo a carga cognitiva do usuário. Elementos como tipografia limpa, ícones padronizados e distribuição equilibrada dos espaços foram selecionados para favorecer a clareza das ações e facilitar a interação. Como resultado, foram criadas telas como a landing page mostrada na Figura 3.

Figura 3 – Landing Page



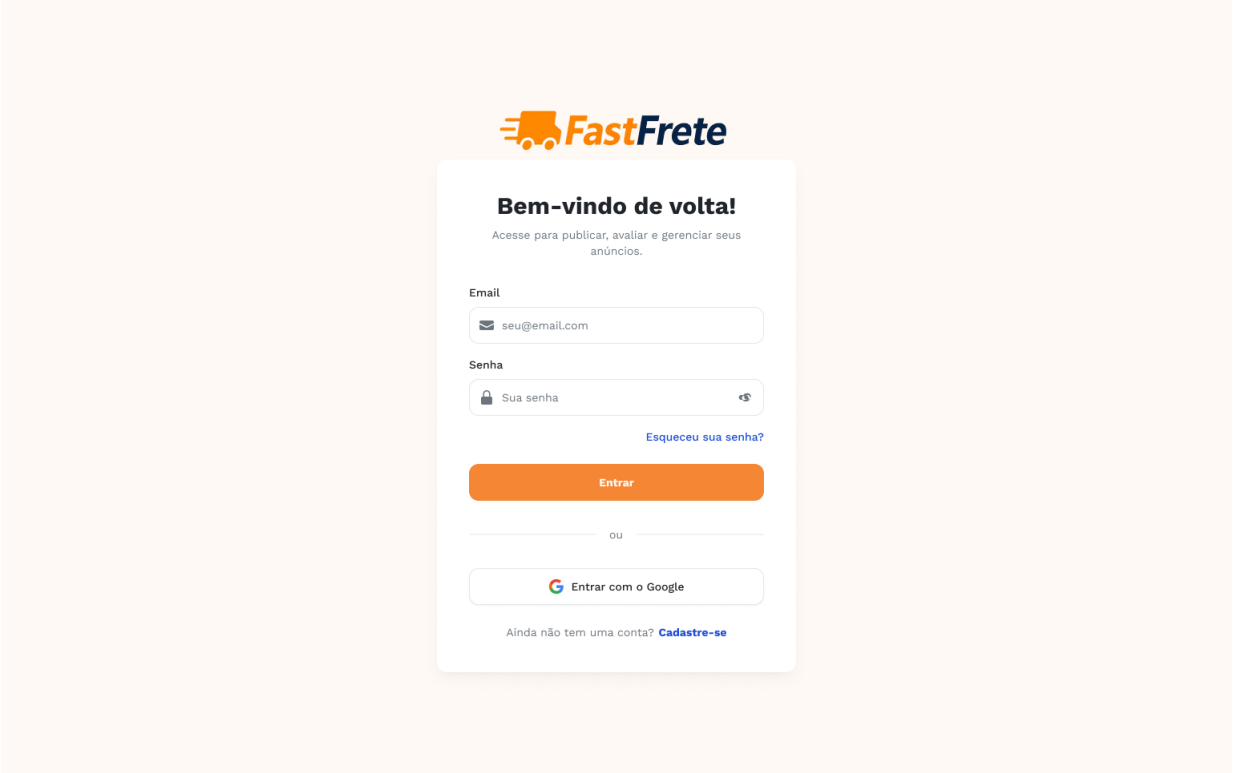
Fonte: Autoria própria (2025).

A landing page é uma página web que funcionará como um cartão de apresentação da plataforma e será utilizada especialmente para a captação de clientes. Por esse motivo, foi elaborado um protótipo que busca transmitir maior confiança ao usuário que ainda não possui cadastro no *FastFrete*. Nele foram adicionados números ilustrativos sobre clientes e fretes resolvidos, vantagens de utilizar a aplicação e opções de cadastro como cliente ou fretista.

Além disso, a estrutura da página foi planejada para evidenciar os diferenciais da aplicação, utilizando títulos destacados e chamadas de ação estratégicas. A hierarquia visual foi organizada de modo a guiar o usuário por informações relevantes, reduzindo dúvidas e favorecendo a tomada de decisão. O uso de elementos gráficos alinhados à identidade visual reforça a credibilidade da plataforma.

Fora a tela de apresentação da aplicação, também foram criadas outras páginas a fim de validar o fluxo de acesso do usuário, como por exemplo a tela de login na Figura 4.

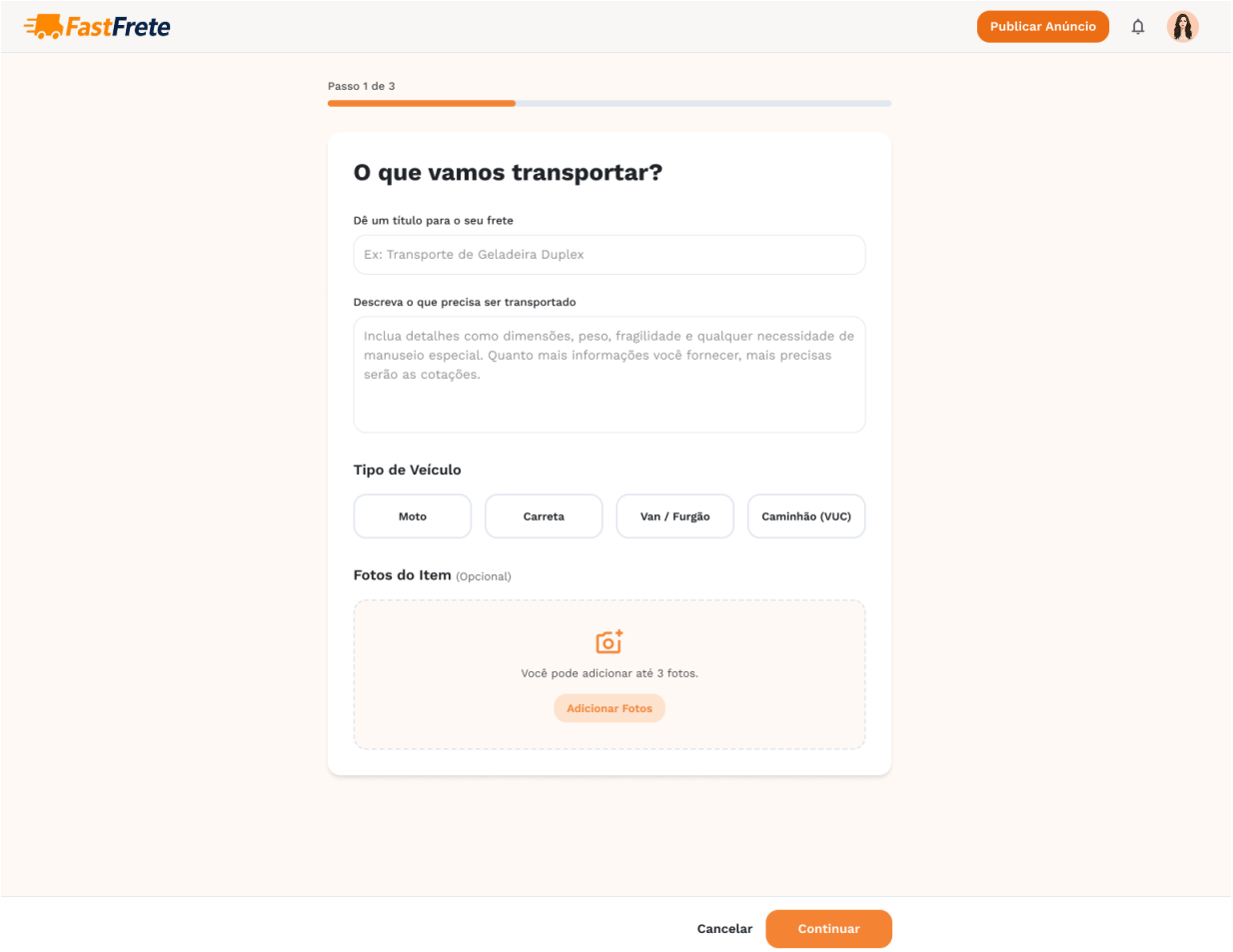
Figura 4 – Tela de login



Fonte: Autoria própria (2025).

Esta tela é a porta de entrada para todos os perfis de usuário e foi desenvolvida com foco na simplicidade e na facilidade de uso. Sua estrutura prioriza a visualização clara dos campos de autenticação, permitindo que o usuário insira seus dados sem ambiguidades. Além disso, a interface foi planejada para oferecer suporte em situações comuns, como a perda de senha ou a ausência de cadastro, apresentando links e instruções que facilitam a recuperação de acesso ou a criação de um novo cadastro.

Figura 5 – Formulário de criação de anúncio



Fonte: Autoria própria (2025).

Assim como a tela de login, outra interface desenvolvida para validar o fluxo do usuário é apresentada na Figura 5. Trata-se do formulário de criação de anúncio de frete, que será preenchido pelo cliente durante o processo de solicitação de um novo serviço. Esse formulário é dividido em várias etapas, permitindo que as informações sejam inseridas de maneira organizada e progressiva. Por ser responsável pelo registro inicial do frete, ele desempenha um papel essencial dentro do fluxo da plataforma.

6.3 Modelagem do banco de dados

Para a modelagem do banco de dados, foram considerados os requisitos funcionais do sistema para o MVP. A partir desses requisitos, identificaram-se as entidades e relações necessárias. Além disso, foram analisados aspectos relacionados à escalabilidade e à manutenção do sistema. Como resultado, obteve-se um diagrama composto por sete tabelas, apresentado na Figura 6, sendo elas:

- **Tabela *users***: Armazena todos os usuários cadastrados na plataforma, independentemente do papel desempenhado. Contém informações pessoais como *name*, *email*,

phone e *password_hash*, bem como o campo *role*, que define se o usuário é cliente, fretista ou administrador. Inclui também *active* e *suspension_reason* para controle gerencial, além do campo *city_id*, que referencia o município associado ao usuário.

- **Tabela *vehicles***: Registra os veículos pertencentes aos usuários do tipo fretista. Cada registro inclui dados como *vehicle_type*, *capacity_kg*, *plate*, *model* e *color*, estando vinculado a um usuário por meio de *user_id*. Permite que um mesmo fretista cadastre múltiplos veículos, com diferentes capacidades e características.

- **Tabela *freights***: Representa os anúncios de frete criados pelos clientes. Inclui informações como *title*, *description* e *vehicle_category*, além dos endereços de origem e destino (*origin_city_id*, *origin_neighborhood*, *origin_street*, *origin_number*, e equivalentes para destino). Também contempla os dados de disponibilidade por meio dos campos *date_mode*, *exact_datetime* e *availability_window*. Esses campos determinam, respectivamente, o formato escolhido (como “EXATA” ou “JANELA”), a data e horário exatos (por exemplo, *2025-11-30 14:00*) ou a janela de disponibilidade selecionada (como *hoje*, *amanhã*, *esta semana*, *próximo fim de semana* ou *data a combinar*). O campo *status* registra a situação atual do frete, como ativo, concluído ou cancelado.

- **Tabela *freight_photos***: Armazena as fotos associadas a um frete, permitindo até três imagens por anúncio. Cada registro referencia um frete por meio de *freight_id* e contém a URL do arquivo (*file_url*) e a ordem de exibição definida em *sort_order*.

- **Tabela *applications***: Representa as candidaturas feitas por fretistas a um determinado frete. Cada candidatura vincula um frete (*freight_id*), um fretista (*freighter_id*) e um veículo (*vehicle_id*). O campo *proposed_value* registra o valor ofertado pelo fretista, enquanto *status* indica a situação da candidatura (como pendente, selecionado ou cancelado).

- **Tabela *reviews***: Armazena as avaliações realizadas após a conclusão do frete. Cada avaliação indica quem avaliou (*reviewer_id*), quem foi avaliado (*reviewed_id*) e a qual frete se refere (*freight_id*). Inclui a nota atribuída (*rating*) e um comentário opcional (*comment*), contribuindo para a construção de reputação na plataforma.

- **Tabela *cities***: Armazena os municípios disponíveis no sistema. Embora o MVP esteja restrito a Guarapuava, esta tabela permite futuras expansões para múltiplas cidades. Contém os campos *name* e *state_code*, que representam o município e sua unidade federativa.

Figura 6 – Diagrama do banco de dados

Fonte: Autoria própria (2025).

7 CONCLUSÃO

O presente trabalho propõe o desenvolvimento do FastFrete, uma aplicação web voltada para fretes urbanos de pequeno porte. A proposta busca responder a uma necessidade recorrente da população: a dificuldade de encontrar, avaliar e contratar serviços de transporte de forma ágil, segura e transparente.

Por meio da implementação de um MVP, o projeto pretende validar a viabilidade de uma solução digital simples que conecte clientes e fretistas locais. O escopo inicial contempla funcionalidades essenciais, como cadastro de usuários, publicação de anúncios, lista de fretistas disponíveis, contato direto, avaliações e gerenciamento administrativo. Essas funcionalidades representam a base mínima necessária para que o sistema possa operar de maneira efetiva e oferecer valor real aos seus usuários.

Acredita-se que a aplicação contribuirá para aumentar a visibilidade dos transportadores autônomos, facilitar a vida de clientes que necessitam de pequenos fretes e organizar um mercado que hoje funciona majoritariamente de forma informal.

Embora este trabalho se restrinja ao MVP, sua realização abre espaço para expansões futuras, como a integração de meios de pagamento digitais, rastreamento em tempo real das cargas, uso de geolocalização para otimização de rotas e a inclusão de recursos avançados de segurança. Essas melhorias poderão consolidar o FastFrete como uma solução completa para o transporte urbano de cargas de pequeno porte.

Assim, a proposta representa uma iniciativa em direção à digitalização de serviços logísticos locais, com potencial de impacto positivo para clientes, fretistas e para a comunidade em geral.

REFERÊNCIAS

ANDERSON, D. J. **Kanban: Successful Evolutionary Change for Your Technology Business**. [*S.l.*]: Blue Hole Press, 2010.

ANTT. 2025. Disponível em: https://app.powerbi.com/view?r=eyJrIjoiMWNkOTdlNGYtZjQ1Mi00Mzk2LTg3M2UtZDEyYmZhMzY3ZTNhIiwidCI6Ijg3YmJlOWRlLWE4OTItN Acesso em: 4 set. 2025.

BECK, K. **Test-Driven Development: By Example**. [*S.l.*]: Addison-Wesley Professional, 2003. Disponível em: https://www.oreilly.com/library/view/test-driven-development/0321146530/. Acesso em: 06 nov. 2025. ISBN 9780321146533.

CHACON, S.; STRAUB, B. **Pro Git**. 2014. Disponível em: https://git-scm.com/book/en/v2. Acesso em: 06 nov. 2025.

CNT, A. C. T. A. **Investimentos em rodovias geram impactos positivos sobre o PIB do transporte, mostra estudo da CNT**. 2025. Disponível em: https://cnt.org.br/agencia-cnt/investimentos-em-rodovias-geram-impactos-positivos-sobre-o-pib-do-transporte-mostra-estudo-da-cnt. Acesso em: 4 set. 2025.

COHN, M. **User Stories Applied: For Agile Software Development**. Boston: Addison-Wesley, 2004.

DBDIAGRAM. 2025. Disponível em: https://docs.dbdiagram.io/. Acesso em: 21 nov. 2025.

DOCKER, I. **Docker Overview**. 2025. Disponível em: https://docs.docker.com/get-started/overview/. Acesso em: 06 nov. 2025.

DRIESSEN, V. **A successful Git branching model (GitFlow)**. 2010. Disponível em: https://nvie.com/posts/a-successful-git-branching-model/. Acesso em: 06 nov. 2025.

DSDM Consortium. **The DSDM Agile Project Framework Handbook**. Ashford, England: DSDM Consortium, 2014. ISBN 9780954886975.

FIGMA, I. **Figma Help Center**. 2025. Disponível em: https://help.figma.com/. Acesso em: 06 nov. 2025.

FOUNDATION, O. **Node.js Documentation**. 2025. Disponível em: https://nodejs.org/en/docs. Acesso em: 06 nov. 2025.

FOWLER, M. **Monorepo: An Anti-pattern?** 2019. Acesso em: 10 fev. 2025. Disponível em: https://martinfowler.com/articles/monorepo.html.

FRETEBRAS. 2025. Disponível em: https://www.fretebras.com.br/. Acesso em: 4 set. 2025.

GITHUB. **Planning and tracking with GitHub Projects**. 2025. Disponível em: https://docs.github.com/en/issues/planning-and-tracking-with-projects. Acesso em: 06 nov. 2025.

LABS, N. **Nuxt 3 Documentation**. 2025. Disponível em: https://nuxt.com/docs. Acesso em: 06 nov. 2025.

MICROSOFT. **TypeScript Documentation**. 2025. Disponível em: https://www.typescriptlang.org/docs/. Acesso em: 06 nov. 2025.

MYŚLIWIEC, K.; TEAM, N. C. **NestJS Documentation**. 2025. Disponível em: https://docs.nestjs.com/. Acesso em: 06 nov. 2025.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide**. 2020. Disponível em: https://scrumguides.org/. Acesso em: 06 nov. 2025.

SUPERFRETE. 2025. Disponível em: https://superfrete.com/blog/tag/sobre-a-plataforma. Acesso em: 21 nov. 2025.

TEAM, P. **Prisma ORM Documentation**. 2025. Disponível em: https://www.prisma.io/docs. Acesso em: 06 nov. 2025.

TEAM, T. C. **Tailwind CSS Documentation**. 2025. Disponível em: https://tailwindcss.com/docs. Acesso em: 06 nov. 2025.

THEMESBERG. **Flowbite Documentation**. 2025. Disponível em: https://flowbite.com/docs/. Acesso em: 06 nov. 2025.

TRUCKPAD. 2025. Disponível em: https://www.truckpad.com.br. Acesso em: 4 set. 2025.

YOU, E.; TEAM, V. C. **Vue.js Guide**. 2025. Disponível em: https://vuejs.org/guide/. Acesso em: 06 nov. 2025.

APÊNDICE A – Funcionalidades Classificadas como Should Have, Could Have e Won’t Have (MoSCoW)

A.1 Funcionalidades Should Have (Deveria Ter)

Essas funcionalidades descritas no modelo história de usuário que são importantes para aprimorar a experiência do usuário e facilitar a operação do sistema, porém a aplicação pode funcionar sem elas.

- **Suporte a múltiplas cidades: Como** cliente ou fretista **quero** poder utilizar a plataforma em diferentes municípios **para** ampliar a área de atendimento e permitir fretes em outras regiões.
- **Histórico de pagamentos: Como** fretista **quero** ver o histórico de solicitações e pagamentos **para** acompanhar meu trabalho no app.
- **Chat de intermediação: Como** cliente **quero** um chat direto com o fretista dentro do app **para** combinar detalhes rápidos sem precisar sair do app.
- **Frete rápido: Como** fretista **quero** aceitar corridas rápidas automáticas **para** pegar serviços sem precisar dar lances.
- **Agendamento de frete: Como** cliente **quero** agendar um frete **para** um dia específico **para** me organizar melhor.
- **Gerador de relatórios: Como** administrador **quero** gerar relatórios de uso **para** melhorar a plataforma.
- **Promoção de frete reverso:** . **Como** cliente **quero** poder ver se um frete é reverso **para** conseguir um preço mais baixo.
- **Fretes intermunicipais: Como** cliente **quero** poder solicitar um frete intermunicipal **para** expandir minhas opções de transporte.

A.2 Funcionalidades Could Have (Poderia Ter)

Essas funcionalidades descritas no modelo história de usuário são desejáveis e poderiam aprimorar ainda mais a experiência do usuário, porém também não são essenciais no início desta aplicação.

- **Verificação de fretista: Como** cliente **quero** escolher um fretista verificado **para** garantir maior segurança na minha entrega.
- **Pagamento intermediado: Como** fretista **quero** receber pagamento antecipado **para** evitar cancelamentos de última hora.

- **Segurança e proteção: Como** cliente **quero** contratar seguro **para** garantir proteção dos meus itens transportados.
- **Histórico de fretes: Como** cliente **quero** ter um histórico de fretes **para** consultar preços passados.
- **Parcerias com empresas: Como** administrador **quero** permitir parcerias com empresas **para** expandir a plataforma.
- **Leilão de fretes: Como** cliente **quero** poder iniciar um leilão **para** encontrar o melhor preço em fretes intermunicipais.

A.3 Funcionalidades Won't Have This Time (Não Terá Desta Vez)

Funcionalidades descritas no modelo história de usuário que serão deixadas de lado inicialmente, mas que podem ser desejáveis em versões futuras.

- **Localização em tempo real: Como** cliente **quero** rastrear a localização do frete em tempo real **como** no Uber.
- **Contrato digital: Como** fretista **quero** obter contrato formal assinado digitalmente **para** garantir compromisso legal.
- **Pagamento parcelado: Como** cliente **quero** pagar o fretista em parcelas **para** dividir o valor do serviço.
- **Serviços completos: Como** administrador **quero** oferecer serviços de mudança completa (embalagem e carregamento) **para** aumentar o alcance do app.