



INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
CAMPUS PICOS
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JEAN CARLOS RODRIGUES SOUSA

VIABUS: PLATAFORMA PARA GESTÃO INTEGRADA E DIGITALIZAÇÃO DO
TRANSPORTE RODOVIÁRIO ALTERNATIVO DE PASSAGEIROS

PICOS, PIAUÍ
2025

JEAN CARLOS RODRIGUES SOUSA

VIABUS: PLATAFORMA PARA GESTÃO INTEGRADA E DIGITALIZAÇÃO DO
TRANSPORTE RODOVIÁRIO ALTERNATIVO DE PASSAGEIROS

Trabalho de Conclusão de Curso (Relatório Técnico de Software) apresentado como exigência parcial para obtenção do diploma do Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Picos.

Orientador: Prof. M^e. Aislan Rafael Rodrigues de Sousa

PICOS, PIAUÍ
2025

JEAN CARLOS RODRIGUES SOUSA

VIABUS: PLATAFORMA PARA GESTÃO INTEGRADA E DIGITALIZAÇÃO DO
TRANSPORTE RODOVIÁRIO ALTERNATIVO DE PASSAGEIROS

Trabalho de Conclusão de Curso (Relatório Técnico de Software) apresentado como exigência parcial para obtenção do diploma do Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Picos.

Aprovada em ____/____/____.

BANCA EXAMINADORA:

Prof. M^e. Aislan Rafael Rodrigues de Sousa(Orientador)
Instituto Federal do Piauí (IFPI)

Prof. M^e.
Instituto Federal do Piauí (IFPI)

Prof. M^e.
Instituto Federal do Piauí (IFPI)

PICOS, PIAUÍ
2025

Dedico este trabalho à minha família, cujo apoio e incentivo tornaram possível esta
nova jornada.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por me conceder força e perseverança ao longo desta caminhada.

Ao meu orientador, Prof. Me. Aislan Rafael Rodrigues de Sousa, pelo apoio, paciência e orientação fundamentais para a realização deste trabalho.

Aos meus pais, Josué José Filho e Maria de Fátima Rodrigues, por todo o apoio, suporte e incentivo ao longo da minha jornada acadêmica.

Aos professores do curso de Análise e Desenvolvimento de Sistemas do IFPI – Campus Picos, por compartilharem conhecimento e contribuírem para minha formação profissional.

E, com imensa gratidão, à minha parceira Karielly de Carvalho, por estar ao meu lado, apoiando-me em todos os momentos.

A todos que, de alguma forma, fizeram parte desta jornada, meu sincero agradecimento.

"Software é uma grande combinação de arte e engenharia."
Bill Gates

[illegible]

ABSTRACT

[illegible]

Keywords: Word 1; Word 2; Word 3.

LISTA DE ILUSTRAÇÕES

Figura 1 – Visão de alto nível da solução.	26
Figura 2 – Agregado organizacional - estrutura multiempresa.	28
Figura 3 – Agregado de configuração de rotas, paradas e horários.	28
Figura 4 – Agregado de recursos operacionais - frota e motoristas.	29
Figura 5 – Agregado de operação - viagens, recursos e vendas.	30
Figura 6 – Organização modular do backend ViaBus.	30
Figura 7 – Pipeline de processamento de requisições.	31
Figura 8 – Diagrama de sequência - fluxo de autenticação.	32
Figura 9 – Arquitetura multi-tenant: uma aplicação serve múltiplas empresas com isolamento automático de dados.	33
Figura 10 – Hierarquia de providers e contextos do frontend.	34
Figura 11 – Tela de login do sistema ViaBus.	37
Figura 12 – Formulário de criação de empresa.	38
Figura 13 – Painel de controle principal com métricas e navegação.	39
Figura 14 – Listagem de paradas com funcionalidades de busca e filtros.	39
Figura 15 – Formulário de cadastro de parada com mapa interativo.	40
Figura 16 – Interface de criação de rota com seleção de paradas.	41
Figura 17 – Listagem de veículos com status operacional.	42
Figura 18 – Interface de gerenciamento de motoristas.	42
Figura 19 – Processo de agendamento de viagem.	43
Figura 20 – Seleção de rota no wizard de passagens.	44
Figura 21 – Seleção de data e horário.	45
Figura 22 – Cadastro de informações do passageiro.	46
Figura 23 – Seleção de locais de embarque e desembarque.	47
Figura 24 – Confirmação final e geração da passagem.	48
Figura 25 – Resultado da pergunta sobre empresa ou serviço.	57
Figura 26 – Resultado da pergunta sobre tamanho da frota de veículos.	57
Figura 27 – Resultado da pergunta sobre número de rotas operadas.	57
Figura 28 – Resultado da pergunta sobre tipo de operação.	58
Figura 29 – Resultado da pergunta sobre gerenciamento atual de passagens.	58
Figura 30 – Resultado da pergunta sobre maiores desafios operacionais.	58
Figura 31 – Resultado da pergunta sobre valor de um sistema único.	59
Figura 32 – Resultado da pergunta sobre funcionalidades mais importantes.	59
Figura 33 – Resultado da pergunta sobre aplicativo como diferencial.	59
Figura 34 – Resultado da pergunta sobre modelo de parceria.	60
Figura 35 – Resultado da pergunta sobre interesse em testar a plataforma.	60

Figura 36 – Diagrama de classes detalhado do domínio ViaBus.	61
--	----

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
CRB	Conselho Regional de Biblioteconomia
IBGE	Instituto Brasileiro de Geografia e Estatística
IFPI	Instituto Federal de Educação, Ciência e Tecnologia do Piauí
UFPI	Universidade Federal do Piauí

LISTA DE SÍMBOLOS

%	Porcentagem
©	Copyright
®	Marca registrada
\$	Dólar
§	Seção
Γ	Letra grega Gama
Λ	Lambda
ζ	Letra grega minúscula zeta
∈	Pertence

SUMÁRIO

1	INTRODUÇÃO	15
1.1	Justificativa	15
1.2	Objetivos	16
1.2.1	Geral	16
1.2.2	Específicos	16
1.3	Metodologia	17
2	FUNDAMENTAÇÃO TEÓRICA	18
2.1	Software as a Service (SaaS)	18
2.1.1	Arquitetura e Modelo de Negócio	18
2.1.2	Vantagens e Desafios para PMEs	19
2.2	Tecnologia e Inovação no Transporte Rodoviário	19
2.3	Arquitetura Tecnológica da Solução Proposta	20
2.3.1	Tecnologias do Backend	20
2.3.1.1	NestJS e Arquitetura Modular	20
2.3.1.2	TypeORM e Mapeamento Objeto-Relacional (ORM)	20
2.3.1.3	Autenticação com JWT e Passport.js	20
2.3.2	Tecnologias do Frontend	21
2.3.2.1	Next.js, SSR e App Router	21
2.3.2.2	React, TypeScript e Ecossistema de UI	21
3	REQUISITOS DO SISTEMA	22
3.1	Requisitos Funcionais	22
3.2	Requisitos Não Funcionais	22
4	TECNOLOGIAS ENVOLVIDAS	24
4.1	Tecnologias do Backend	24
4.2	Tecnologias do Frontend	24
4.3	Infraestrutura e Ferramentas de Suporte	25
5	ARQUITETURA E MODELAGEM DO SISTEMA	26
5.1	Arquitetura da Solução	26
5.2	Modelagem do Banco de Dados	26
5.2.1	Diagramas UML Fracionados por Agregado	27
5.3	Arquitetura do Backend	30
5.3.1	Organização Modular	30
5.3.2	Pipeline de Processamento	31

5.3.3	Fluxo de Autenticação	31
5.3.4	Arquitetura Multi-tenant	32
5.4	Arquitetura do Frontend	33
5.4.1	Stack Tecnológico e Organização	33
5.4.2	Arquitetura de Roteamento	34
5.4.3	Gerenciamento de Estado e Contextos	34
5.4.4	Sistema de Autenticação e Comunicação	35
5.4.5	Arquitetura de Componentes	35
5.4.6	Tipagem e Validação	35
5.5	Estratégia de Implantação	35
6	MÓDULOS E FUNCIONALIDADES DO FRONTEND	37
6.1	Autenticação e Gestão de Empresa	37
6.1.1	Tela de Login e Registro	37
6.1.2	Fluxo de Criação da Empresa	38
6.2	Painel de Controle (Dashboard)	38
6.2.1	Visão Geral	39
6.3	Módulo de Paradas	39
6.3.1	Listagem de Paradas	39
6.3.2	Criação e Edição de Paradas	40
6.4	Módulo de Rotas	40
6.5	Módulo de Veículos	41
6.6	Módulo de Motoristas	42
6.7	Módulo de Viagens	42
6.8	Módulo de Venda de Passagens	43
6.8.1	Assistente de Venda de Passagens	43
6.8.1.1	1. Seleção de Rota	43
6.8.1.2	2. Seleção de Data e Horário	44
6.8.1.3	3. Informações do Passageiro	45
6.8.1.4	4. Seleção de Locais de Embarque e Desembarque	46
6.8.1.5	5. Confirmação e Pagamento	47
6.9	Características da Interface	48
6.9.1	Design Responsivo	48
6.9.2	Experiência do Usuário	48
6.9.3	Integração com Mapas	49
7	RESULTADOS E DISCUSSÃO	50
7.1	Verificação de Requisitos	50
7.2	Avaliação Heurística	51
7.3	Discussão dos Resultados	52

8	CONSIDERAÇÕES FINAIS	53
	APÊNDICE A – QUESTIONÁRIO DA PESQUISA DE MERCADO .	54
	APÊNDICE B – RESULTADOS DA PESQUISA DE MERCADO . .	57
	APÊNDICE C – DIAGRAMA DE CLASSE DO VIABUS	61
	APÊNDICE D – REPOSITÓRIOS DO CÓDIGO-FONTE	62

1 INTRODUÇÃO

Em um país de dimensões continentais, o transporte rodoviário de passageiros funciona como um elemento estratégico para a integração socioeconômica, conectando municípios e garantindo a mobilidade da população (**FGV2023**). Enquanto o sistema interestadual é regulamentado em nível federal pela Agência Nacional de Transportes Terrestres (ANTT), conforme estabelecido em sua lei de criação (**BRASIL2001**), uma vasta e heterogênea rede de transporte intermunicipal opera sob jurisdição estadual.

No estado do Piauí, esta modalidade é uma realidade consolidada, cuja organização é definida pela Lei Nº 8.562 de 2025, que dispõe sobre o Sistema de Transporte Rodoviário Intermunicipal de Passageiros do Estado do Piauí (STRIP/PI). A referida lei classifica o "serviço alternativo" como uma das categorias oficiais do sistema, designando a Secretaria dos Transportes (SETRANS) como o poder concedente (**PIAUI2025**). Este serviço, prestado por veículos de menor porte, surge para suprir as lacunas deixadas pelo sistema convencional. A existência de uma legislação específica demonstra a relevância do setor, mas também evidencia a necessidade de ferramentas de gestão que se adaptem às suas particularidades operacionais.

Diante desse cenário, este trabalho propõe o desenvolvimento de uma plataforma baseada no modelo Software como Serviço (*SaaS, Software as a Service*) para a gestão integrada de empresas de transporte rodoviário alternativo. A escolha por este modelo se fundamenta na definição de Chong e Carraro (2006 apud **melo2007software**) como um "software implementado como um serviço hospedado e acessado pela Internet", o que permite que empresas utilizem soluções baseadas em nuvem com custos operacionais reduzidos e alta escalabilidade — características especialmente vantajosas para o setor em foco.

Este cenário, onde um serviço regulamentado e essencial como o transporte alternativo ainda opera com uma gestão predominantemente analógica — fato constatado na pesquisa de mercado realizada para este trabalho (Apêndice B) —, evidencia os desafios para a modernização do setor. A ausência de sistemas de gestão integrados e a consequente dependência de processos manuais não apenas comprometem a eficiência operacional e a qualidade dos serviços prestados, mas também criam uma barreira para a inovação e o crescimento sustentável das empresas que atuam nesse segmento.

1.1 JUSTIFICATIVA

O transporte rodoviário alternativo de passageiros no Brasil, embora essencial para a conectividade regional, opera de forma majoritariamente analógica e fragmen-

tada. Uma pesquisa de mercado realizada para este estudo (Apêndice B) revelou que a gestão de frotas e a venda de passagens ainda são fortemente dependentes de processos manuais, como o uso de cadernos de anotações e planilhas. Essa carência de ferramentas digitais integradas limita a eficiência e o potencial de crescimento do setor. Neste contexto, a adoção de plataformas *SaaS* surge como uma solução estratégica para transformar esse cenário, otimizando recursos e reduzindo custos operacionais. Estudos de mercado mais amplos corroboram essa visão, indicando que tecnologias no setor de transportes podem elevar a eficiência logística em até 15% (**setcepar2023**).

Além disso, uma plataforma *SaaS* voltada para o transporte alternativo pode facilitar o acesso à inovação para pequenas e médias empresas, sem exigir altos investimentos. Esse modelo permite automação da bilhetagem, gestão integrada de rotas e melhor experiência para o passageiro. Essas soluções já demonstram impacto positivo em outros segmentos do transporte, aumentando a competitividade e garantindo serviços mais confiáveis (**prologapp2024**). Assim, a digitalização do setor não só fortalece as empresas, mas também melhora a mobilidade interurbana, tornando os serviços mais eficientes e acessíveis.

1.2 OBJETIVOS

1.2.1 Geral

Desenvolver um protótipo funcional de uma plataforma *SaaS* para a gestão integrada do transporte rodoviário alternativo de passageiros, verificando a implementação de suas funcionalidades essenciais e avaliando a usabilidade de sua interface a partir de princípios de design.

1.2.2 Específicos

- Realizar uma pesquisa de mercado para identificar as dores e os processos manuais de empresas do setor de transporte alternativo;
- Especificar os requisitos funcionais e não funcionais de um sistema de gestão com base nas necessidades levantadas na pesquisa;
- Desenvolver um protótipo funcional da plataforma ViaBus, implementando os módulos de gestão de rotas, paradas, veículos, motoristas e um fluxo para agendamento de passagens;
- Realizar uma verificação técnica do protótipo para analisar a aderência do software aos requisitos especificados e conduzir uma avaliação heurística para identificar possíveis melhorias de usabilidade na interface.

1.3 METODOLOGIA

A metodologia empregada para a concepção e desenvolvimento do sistema ViaBus foi estruturada em três etapas sequenciais: levantamento de requisitos, desenvolvimento do protótipo e avaliação da solução.

A primeira etapa, de levantamento de requisitos, utilizou uma abordagem mista, partindo da observação empírica do autor sobre os desafios do setor de transporte alternativo. Tais observações foram então validadas por meio de uma pesquisa de mercado qualitativa. Para tal, foi elaborado um questionário online (detalhado no Apêndice A) e aplicado junto a gestores de duas empresas do setor. As respostas, consolidadas na tabela do Apêndice B, foram essenciais para a especificação formal dos requisitos que nortearam o desenvolvimento.

A segunda etapa, de desenvolvimento do protótipo, seguiu um processo iterativo alinhado a práticas de prototipagem evolutiva, onde o software é construído e refinado em ciclos contínuos (**sommerville2011software**). Adotou-se a estratégia *Frontend-First*, que prioriza a construção da interface do usuário (UI) como guia para o desenvolvimento do sistema. Utilizando o framework Next.js e a biblioteca de componentes shadcn/ui, todas as telas da aplicação foram inicialmente desenvolvidas com dados estáticos (*mockados*), permitindo a validação dos fluxos de interação antes da implementação da lógica de negócio no backend.

Por fim, a terceira etapa consistiu na avaliação do protótipo. Devido à impossibilidade de realizar testes com usuários finais, optou-se por uma verificação interna em duas frentes. A primeira foi uma *Verificação de Requisitos*, na qual se analisou sistematicamente o atendimento aos requisitos funcionais e não funcionais. A segunda foi uma *Avaliação Heurística*, um método de inspeção consolidado para encontrar problemas de usabilidade em interfaces (**Nielsen1994**). Atuando como avaliador especialista, o autor inspecionou o sistema com base nas 10 heurísticas de usabilidade de Nielsen, cujos resultados detalhados são apresentados no Capítulo 5.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são apresentados os conceitos essenciais que fundamentam o desenvolvimento deste trabalho. A primeira seção aborda o modelo *Software as a Service* (SaaS), detalhando sua definição, arquitetura, vantagens e desafios. A segunda seção explora a aplicação de tecnologias no setor de transporte rodoviário, conectando a teoria à prática observada. Por fim, a terceira seção aprofunda-se na arquitetura e nas tecnologias específicas escolhidas para a construção do protótipo ViaBus.

2.1 SOFTWARE AS A SERVICE (SAAS)

O modelo *Software as a Service* (SaaS), ou Software como Serviço, representa uma mudança de paradigma na forma como o software é distribuído e consumido. Diferentemente do modelo tradicional *on-premise*, onde o cliente adquire licenças e é responsável pela infraestrutura, no modelo SaaS o cliente paga uma assinatura periódica para acessar a aplicação pela internet, que é hospedada na nuvem pelo provedor do serviço (**moveideias2025saas**).

Nesse modelo, toda a infraestrutura subjacente — servidores, armazenamento, redes e o próprio software — é gerenciada pelo provedor. Isso significa que o fornecedor é responsável pela manutenção, atualizações, segurança e disponibilidade da aplicação, permitindo que as empresas clientes foquem em suas atividades principais sem se preocupar com a complexidade da gestão de TI (**moveideias2025saas**).

2.1.1 Arquitetura e Modelo de Negócio

A arquitetura mais comum em soluções SaaS é a *multi-tenancy* (multilocação), onde uma única instância da aplicação e da infraestrutura serve a múltiplos clientes (locatários ou *tenants*) (**frontegg2021multitenant**). Embora compartilhem os mesmos recursos computacionais, os dados de cada cliente são mantidos isolados e seguros, garantindo a privacidade e a confidencialidade das informações. Essa abordagem permite que o provedor otimize os recursos e reduza os custos, o que se reflete em preços mais acessíveis para o cliente final.

O modelo de negócio é baseado em assinaturas, que podem variar em preço conforme o número de usuários, os recursos contratados ou o volume de uso. Essa flexibilidade oferece escalabilidade, permitindo que as empresas ajustem o serviço de acordo com seu crescimento e suas necessidades, pagando apenas pelo que utilizam (**prologapp2024saas**).

2.1.2 Vantagens e Desafios para PMEs

A adoção de plataformas SaaS oferece um conjunto significativo de vantagens estratégicas para as Pequenas e Médias Empresas (PMEs). A principal delas é a drástica redução de custos, pois o modelo de assinatura elimina a necessidade de altos investimentos de capital (CAPEX) em licenças e hardware, transformando-os em despesas operacionais (OPEX) previsíveis (**praxio2021vantagens**).

Além da otimização financeira, outros benefícios se destacam. O modelo introduz uma notável escalabilidade e flexibilidade, permitindo que as empresas aumentem ou diminuam facilmente a capacidade de uso do software para responder rapidamente às mudanças do mercado. Soma-se a isso a acessibilidade e mobilidade intrínsecas ao serviço, que, por ser acessado via internet, pode ser utilizado de qualquer lugar e em diferentes dispositivos — algo transformador para o setor de logística. Ao terceirizar a gestão da infraestrutura de TI, as empresas também ganham maior foco no *core business*, dedicando mais tempo e recursos à otimização de suas operações. Por fim, provedores de SaaS geralmente oferecem um nível de segurança e disponibilidade superior ao que uma PME poderia implementar por conta própria, garantido por Acordos de Nível de Serviço (SLAs) (**prologapp2024saas**; **praxio2021vantagens**).

Apesar das vantagens, o modelo também apresenta desafios, como a dependência de uma conexão estável com a internet e opções de personalização potencialmente mais limitadas. É justamente esse conjunto de benefícios, especialmente a redução de custos e a acessibilidade, que torna o modelo SaaS uma solução promissora para modernizar setores tradicionalmente analógicos, como o de transporte rodoviário.

2.2 TECNOLOGIA E INOVAÇÃO NO TRANSPORTE RODOVIÁRIO

O setor de transporte rodoviário de passageiros, apesar de sua importância socioeconômica, apresenta um ritmo lento na adoção de tecnologias digitais (**sestsenat2021relatorio**). A ausência de soluções tecnológicas integradas, especialmente no segmento alternativo, resulta em ineficiências como a falta de controle sobre os processos, a incapacidade de atender a picos de demanda e altos índices de reclamação de clientes (**fateczl2022impactos**). Essa carência é corroborada pela pesquisa de mercado conduzida para este trabalho (ver Apêndice B), que revelou que 100% dos gestores entrevistados ainda dependem de métodos como cadernos de anotações e mensagens de WhatsApp para gerenciar suas operações.

Plataformas SaaS surgem como uma solução ideal para democratizar o acesso a tecnologias avançadas neste setor. A análise de soluções comerciais existentes, como TOTVS, iTransport e Praxio, revela uma lacuna de mercado: as ferramentas ou são muito complexas e caras para PMEs, ou são focadas em nichos específicos

como o fretamento corporativo, não atendendo de forma integrada as necessidades do operador de transporte alternativo (**totvs2025passageiros; itransport2025gestao; praxioluna2025venda**).

2.3 ARQUITETURA TECNOLÓGICA DA SOLUÇÃO PROPOSTA

2.3.1 Tecnologias do Backend

O backend é o alicerce da plataforma, responsável pela lógica de negócios, persistência de dados e segurança. A *stack* escolhida foi projetada para ser modular, robusta e escalável, utilizando tecnologias consolidadas no ecossistema TypeScript.

2.3.1.1 NestJS e Arquitetura Modular

O framework escolhido para o desenvolvimento do backend foi o NestJS. Trata-se de um framework Node.js progressivo, construído com e para o TypeScript, que utiliza uma arquitetura fortemente modular (**nestjs2025framework**). O pilar do NestJS é o conceito de Módulos, que organizam o código em blocos coesos e funcionais (e.g., um módulo para autenticação, outro para gestão de rotas). Cada módulo encapsula seus próprios *controllers*, *providers* (serviços) e pode importar ou exportar funcionalidades (**nestjs2025modules**). Essa estrutura promove uma forte separação de responsabilidades, facilita a reutilização de código e a manutenção do sistema, sendo ideal para a plataforma proposta, onde funcionalidades complexas podem ser desenvolvidas como módulos independentes (**devanddeliver2024architecture**).

2.3.1.2 TypeORM e Mapeamento Objeto-Relacional (ORM)

Para a camada de persistência de dados, a escolha foi o TypeORM. Um *Object-Relational Mapper* (ORM) é uma técnica que cria uma ponte entre o paradigma orientado a objetos da aplicação e o paradigma relacional dos bancos de dados, permitindo que os desenvolvedores manipulem o banco de dados através de objetos e classes, abstraindo a necessidade de escrever consultas SQL manualmente (**logrocket2024typeorm**). O TypeORM é um ORM maduro para o ecossistema TypeScript, que utiliza intensivamente decoradores para definir Entidades (classes que mapeiam para tabelas) de forma declarativa e fortemente tipada. Isso acelera o desenvolvimento e reduz a probabilidade de erros relacionados a dados (**devto2024typeorm**).

2.3.1.3 Autenticação com JWT e Passport.js

A segurança da API é um requisito crítico. A estratégia de autenticação escolhida foi baseada em *JSON Web Tokens* (JWT), implementada com a biblioteca Passport.js. JWT é um padrão aberto (RFC 7519) para a criação de tokens de acesso

compactos e autossuficientes. Quando um cliente faz uma requisição a um recurso protegido, ele envia o JWT, e o servidor pode verificar a assinatura para autenticar o usuário sem precisar consultar um banco de dados de sessões, resultando em uma autenticação *stateless* ideal para APIs RESTful (**soshace2024jwt**). O Passport.js atua como um *middleware* de autenticação modular, e a estratégia *passport-jwt* é projetada especificamente para extrair e verificar a validade desses tokens, oferecendo uma solução robusta e padronizada para proteger as rotas da API (**passportjs2025jwt**).

2.3.2 Tecnologias do Frontend

A interface do usuário (*frontend*) é o principal ponto de contato com os gestores e passageiros. Sua arquitetura foi projetada com foco em performance e robustez, utilizando um ecossistema moderno baseado em React.

2.3.2.1 Next.js, SSR e App Router

O framework escolhido para o frontend é o Next.js. Uma de suas principais características é o suporte nativo à Renderização no Servidor (*Server-Side Rendering* - SSR), na qual a página HTML é gerada no servidor a cada requisição. Isso melhora o tempo de carregamento inicial da página e a otimização para motores de busca (SEO) (**medium2025ssr**). Com a introdução do *App Router*, o Next.js adotou por padrão o uso de *React Server Components*, que permitem que a busca de dados e a renderização de componentes não interativos ocorram exclusivamente no servidor. Isso resulta em uma redução significativa da quantidade de JavaScript enviada ao cliente e otimiza a performance geral da aplicação (**nextjs2025servercomponents**).

2.3.2.2 React, TypeScript e Ecossistema de UI

A base da interface é o React, uma biblioteca JavaScript para a construção de interfaces baseada em um modelo de componentes reutilizáveis. Para garantir a robustez e a manutenibilidade, o React é utilizado com o TypeScript, um superconjunto do JavaScript que adiciona tipagem estática. O TypeScript permite a detecção de erros em tempo de compilação e serve como uma forma de documentação viva, tornando o código mais fácil de entender e facilitando a colaboração (**dhiwise2024reacttypescript**).

Para a estilização, foi adotado o Tailwind CSS, um framework *utility-first* que acelera o desenvolvimento e garante consistência visual (**medium2025cssframeworks**). Complementando-o, a biblioteca de componentes *shadcn/ui* foi utilizada. Diferente de bibliotecas tradicionais, seus componentes são copiados para o código-fonte do projeto, dando ao desenvolvedor controle total sobre o código e permitindo personalizações profundas sem *vendor lock-in* (**shadcnui2025docs**).

3 REQUISITOS DO SISTEMA

A concepção do sistema ViaBus partiu de uma necessidade concreta, identificada na pesquisa de mercado (Apêndice B) realizada com gestores de empresas de transporte de passageiros. A pesquisa revelou um cenário operacional amplamente manual, onde ferramentas como *"cadernos de anotações"* e *"mensagens de WhatsApp"* são o padrão para tarefas críticas como a venda de passagens e a organização de listas de embarque. Essa dependência de métodos analógicos resulta nos principais desafios apontados pelos gestores: a dificuldade de *"controlar os assentos já vendidos"*, a *"perda de tempo organizando listas"* e a *"comunicação com os motoristas"*.

Diante desse diagnóstico, os requisitos do ViaBus foram definidos para atacar diretamente essas dores. Eles representam a digitalização e a centralização de um processo hoje fragmentado e suscetível a erros. Este capítulo detalha esses requisitos em duas perspectivas: os requisitos funcionais, que descrevem o que o sistema deve ser capaz de fazer para o usuário, e os requisitos não funcionais, que definem os atributos de qualidade e as condições sob as quais o sistema deve operar.

3.1 REQUISITOS FUNCIONAIS

Os requisitos funcionais foram agrupados em categorias que espelham o fluxo de trabalho de uma empresa de transporte. Eles traduzem as necessidades operacionais em funcionalidades concretas, conforme detalhado na Tabela 1.

3.2 REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais definem os critérios de qualidade que garantem uma boa experiência de uso e a confiabilidade do sistema. Eles são apresentados na Tabela 2.

Tabela 1 – Requisitos funcionais do sistema ViaBus

ID	Descrição da Funcionalidade
Organização da Empresa e Recursos	
RF01	O gestor deve poder cadastrar e gerenciar os dados da sua frota de veículos.
RF02	O gestor deve poder cadastrar, consultar e atualizar as informações de seus motoristas, como dados de contato e validade da CNH.
RF03	O sistema deve permitir o cadastro de todos os pontos de parada (embarque/desembarque) utilizados pela empresa, incluindo sua localização.
RF04	O gestor deve poder criar e organizar as rotas (itinerários) da empresa, definindo a sequência de paradas e os horários de operação.
Operação e Venda de Passagens	
RF05	O sistema deve permitir o agendamento de viagens futuras, associando uma rota, um veículo e um motorista.
RF06	Um vendedor ou gestor deve poder vender passagens de forma rápida e guiada, selecionando a rota e a data da viagem.
RF07	Ao vender uma passagem, o sistema deve registrar as informações essenciais do passageiro.
RF08	O sistema não deve permitir, em hipótese alguma, a venda de uma passagem para um assento que já está ocupado em uma viagem (evitar overbooking).
RF09	O sistema deve gerar uma lista de passageiros de fácil consulta para cada viagem, que possa ser usada no momento do embarque.
Controle e Análise	
RF10	O gestor deve poder visualizar rapidamente a taxa de ocupação de cada viagem para tomar decisões.
RF11	O sistema deve fornecer relatórios simples de vendas, mostrando o faturamento por rota ou por período.

Tabela 2 – Requisitos não funcionais do sistema ViaBus

Usabilidade e Acesso	
Usabilidade e Acesso	
RNF01	A interface do sistema deve ser clara, organizada e intuitiva, exigindo o mínimo de treinamento para um novo usuário.
RNF02	O sistema deve ser fácil de usar tanto em um computador no escritório quanto em um celular ou tablet na rodoviária.
Confiabilidade e Desempenho	
RNF03	O sistema deve ser rápido, com telas e informações carregando em poucos segundos.
RNF04	A plataforma deve estar disponível para uso a maior parte do tempo, sem quedas ou instabilidades frequentes.
Segurança	
RNF05	O acesso ao sistema deve ser protegido por login e senha.
RNF06	Os dados de uma empresa de transporte não podem, sob nenhuma circunstância, ser acessados pelos usuários de outra empresa.

4 TECNOLOGIAS ENVOLVIDAS

A construção do protótipo ViaBus foi fundamentada em um conjunto de tecnologias de mercado, selecionadas a partir de critérios objetivos como adequação aos requisitos do projeto, ecossistema e características técnicas específicas. Este capítulo apresenta as principais ferramentas, bibliotecas e frameworks utilizados, organizados em tabelas por categoria para facilitar a consulta e a compreensão da arquitetura da solução. Cada tabela detalha a aplicação específica da tecnologia no projeto e a justificativa técnica para sua escolha. Para consultar as versões exatas de cada dependência utilizada, veja o Apêndice D, que lista os repositórios do código-fonte do projeto.

4.1 TECNOLOGIAS DO BACKEND

As tecnologias do backend formam o alicerce da aplicação, responsáveis pela lógica de negócio, segurança e persistência dos dados. As principais escolhas estão detalhadas na Tabela 3.

Tabela 3 – Tecnologias utilizadas no Backend

Tecnologia	Aplicação no Projeto ViaBus	Justificativa da Escolha
NestJS	Framework principal para a construção da API RESTful, estruturando os módulos de negócio (rotas, motoristas, passagens, etc.).	A arquitetura modular e o sistema de injeção de dependência nativo facilitam a organização e a manutenção de um projeto com múltiplos domínios.
TypeORM	Mapeamento das classes do domínio para as tabelas do banco de dados PostgreSQL, abstraindo a escrita de consultas SQL.	Integração madura com o ecossistema TypeScript e suporte a <i>decorators</i> para a definição de entidades, o que aumenta a produtividade e a clareza do código.
Passport.js com JWT	Implementação da estratégia de autenticação e autorização, protegendo as rotas da API e gerenciando as sessões dos usuários.	Padrão de mercado para autenticação <i>stateless</i> em APIs. A modularidade do Passport.js permite uma implementação desacoplada e segura.

4.2 TECNOLOGIAS DO FRONTEND

O frontend é a camada de apresentação do sistema, com a qual o usuário interage diretamente. As tecnologias foram escolhidas com foco em performance e experiência de desenvolvimento, conforme a Tabela 4.

Tabela 4 – Tecnologias utilizadas no Frontend

Tecnologia	Aplicação no Projeto ViaBus	Justificativa da Escolha
Next.js	Framework principal para a construção da interface de usuário (UI), incluindo o painel de gestão e os fluxos de venda de passagens.	Suporte nativo a Server-Side Rendering (SSR) e Server Components, estratégias que melhoraram o tempo de carregamento inicial e a performance percebida pelo usuário.
React	Biblioteca base para a criação de toda a interface em um modelo de componentes reutilizáveis (botões, formulários, tabelas, etc.).	Ecossistema consolidado e vasta disponibilidade de bibliotecas, o que viabilizou a rápida prototipação da interface.
TypeScript	Utilizado em todo o desenvolvimento (backend e frontend) para adicionar tipagem estática ao JavaScript.	Aumenta a manutenibilidade do código e reduz a ocorrência de erros em tempo de execução, servindo como uma forma de documentação para a estrutura de dados.
Tailwind CSS	Framework CSS utilizado para a estilização de todos os componentes da interface.	A abordagem <i>utility-first</i> acelera o desenvolvimento da UI e facilita a manutenção de um design system consistente em toda a aplicação.

4.3 INFRAESTRUTURA E FERRAMENTAS DE SUPORTE

Este conjunto de ferramentas dá suporte ao desenvolvimento, à persistência dos dados e à implantação do sistema, garantindo consistência e qualidade ao longo do ciclo de vida do projeto (Tabela 5).

Tabela 5 – Infraestrutura e Ferramentas de Suporte

Tecnologia	Aplicação no Projeto ViaBus	Justificativa da Escolha
PostgreSQL	Sistema de Gerenciamento de Banco de Dados (SGBD) relacional para a persistência de todos os dados da aplicação.	SGBD de código aberto com funcionalidades avançadas, como suporte a dados geoespaciais (PostGIS), e alta compatibilidade com o TypeORM.
Docker	Containerização do backend, do frontend e do banco de dados para os ambientes de desenvolvimento e produção.	Garante a paridade entre os ambientes, simplifica a configuração do projeto e facilita a implantação em qualquer provedor de nuvem.
ESLint e Prettier	Ferramentas para análise estática e formatação automática do código-fonte.	Asseguram a padronização e a qualidade do código, independentemente do desenvolvedor, facilitando a leitura e a manutenção futura.

5 ARQUITETURA E MODELAGEM DO SISTEMA

5.1 ARQUITETURA DA SOLUÇÃO

Esta seção descreve a organização estrutural do sistema ViaBus, que provê funções de gestão de transporte para empresas: cadastro de empresas, usuários, motoristas, veículos, rotas, paradas, viagens e bilhetes. A solução adota uma arquitetura web em camadas, com separação clara entre apresentação (frontend), lógica de negócio e APIs (backend) e persistência (banco de dados).

No *frontend*, utiliza-se Next.js 15 com roteamento via App Router, estado de sessão via NextAuth (estratégia *JWT*) e componentes React tipados (TypeScript). O *frontend* consome APIs REST autenticadas, mantém contexto de usuário/empresa e incorpora mapas e edição geográfica com Leaflet/React-Leaflet para operações sobre rotas e paradas.

O *backend* é implementado com NestJS 11, estruturado por módulos de domínio (*auth, users, companies, drivers, vehicles, stops, routes, trips, tickets, etc.*). As regras de negócio são expostas por controladores REST, protegidos por autenticação *JWT* e autorização baseada em papéis. A multiempresa é tratada por *companyId* e por um interceptor que injeta o contexto de empresa a partir do token. A persistência usa TypeORM com PostgreSQL.

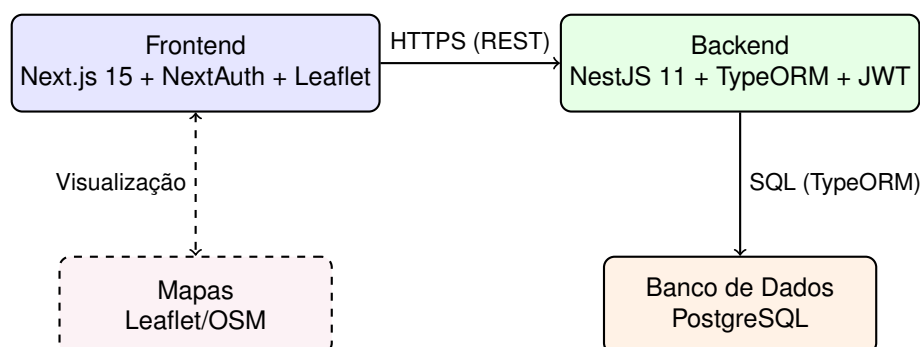


Figura 1 – Visão de alto nível da solução.

5.2 MODELAGEM DO BANCO DE DADOS

O modelo de dados foi projetado para refletir as agregações de negócio, garantindo a coesão e a integridade das entidades do domínio. A Tabela 6 sumariza as principais entidades e seus papéis. Para facilitar a análise, a modelagem será apresentada através de diagramas UML fracionados por agregado. A visão completa e detalhada de todos os relacionamentos do domínio está disponível para consulta no Apêndice C.

Entidade	Descrição resumida
<code>companies</code>	Empresa (razão social, nome fantasia, CNPJ, slug, e-mail, telefone, logo)
<code>users</code>	Usuário (nome, e-mail, senha, papel, status, telefone, foto, <code>company_id</code>)
<code>drivers</code>	Motorista (nome, CPF, CNH, categoria, validade, telefone, e-mail, data nascimento, contratação, status, contato emergência, endereço, observações, <code>company_id</code>)
<code>vehicles</code>	Veículo (placa, modelo, marca, ano, capacidade, categoria, conforto, tipo ônibus, data aquisição, hodômetro, manutenção, status, observações, <code>company_id</code>)
<code>addresses</code>	Endereço (CEP, logradouro, número, complemento, bairro, cidade, estado, longitude, latitude)
<code>stops</code>	Parada (nome, <code>address_id</code> , ativa, acessibilidade, abrigo, <code>company_id</code>)
<code>routes</code>	Rota (nome, descrição, ativa, duração estimada, distância, <code>company_id</code>)
<code>route_stops</code>	Associação rota–parada com ordem e horário de partida opcional
<code>route_schedules</code>	Agenda por dia da semana para a rota (0-6, domingo-sábado)
<code>trips</code>	Viagem (rota, horários partida/chegada, status, preço base, assentos total/disponível, auto-gerada, observações, <code>company_id</code>)
<code>trip_vehicles</code>	Associação viagem–veículo com motoristas primário/secundário
<code>tickets</code>	Bilhete (passageiro, documento, telefone, e-mail, assento, preço, status, pontos embarque/desembarque com coordenadas, observações, <code>company_id</code>)

Tabela 6 – Principais entidades do domínio.

5.2.1 Diagramas UML Fracionados por Agregado

Para facilitar a análise do domínio e a compreensão das decisões de modelagem, o esquema de dados foi decomposto em agregados lógicos. Um agregado é um cluster de objetos de domínio que podem ser tratados como uma única unidade, garantindo a consistência das regras de negócio. A seguir, cada agregado principal é detalhado, justificando-se as principais decisões de modelagem.

Agregado Organizacional: Empresas e Usuários

O pilar da arquitetura multi-tenant do sistema reside neste agregado. A entidade `Company` é a raiz de todos os dados operacionais. A decisão de incluir a chave estrangeira `companyId` na entidade `User` estabelece o vínculo fundamental que garante que cada usuário pertença inequivocamente a uma única empresa. Este design é a base para o filtro de isolamento de dados que será aplicado em todas as consultas subsequentes, conforme detalhado na seção de arquitetura do backend.

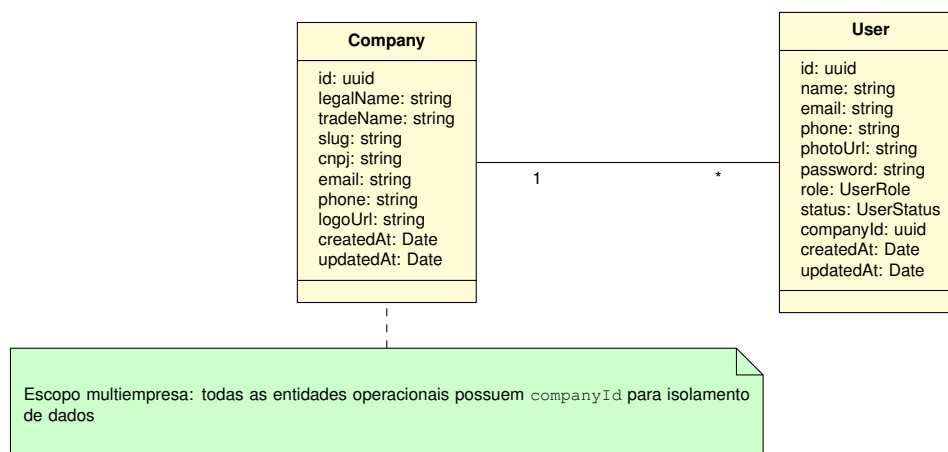


Figura 2 – Agregado organizacional - estrutura multiempresa.

Agregado de Configuração: Rotas e Paradas

Este agregado modela a estrutura logística fundamental de uma empresa de transporte. A decisão mais crítica aqui foi a criação da entidade associativa `RouteStop`. Em vez de uma relação simples, a `RouteStop` foi necessária para resolver o relacionamento N:N entre `Route` e `Stop`, permitindo que uma mesma parada (ex: "Rodoviária de Picos") possa pertencer a múltiplas rotas. Mais importante, esta entidade armazena o atributo `order`, que é a regra de negócio central para definir a sequência do trajeto, transformando um conjunto de pontos em um itinerário ordenado. Adicionalmente, a entidade `RouteSchedule` foi criada para desacoplar a definição da rota de sua frequência operacional, permitindo que uma mesma rota tenha horários diferentes dependendo do dia da semana.

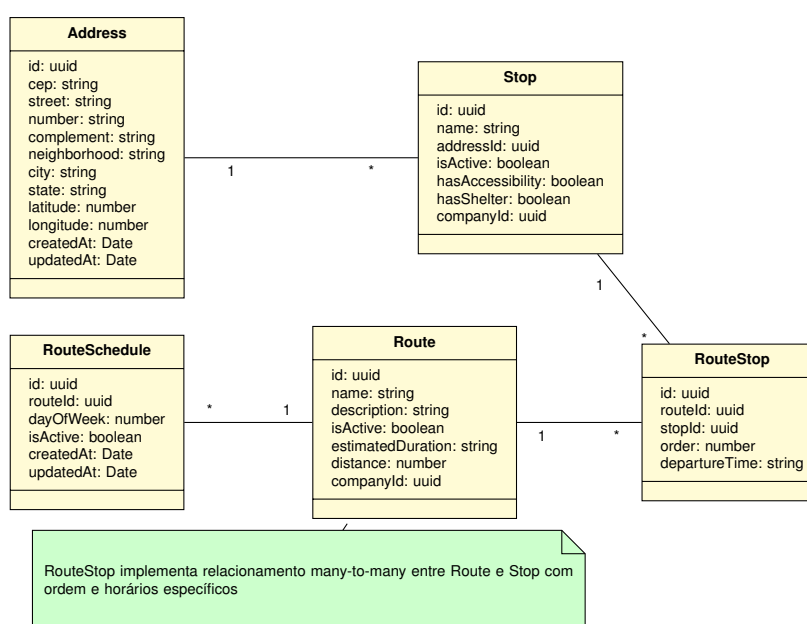


Figura 3 – Agregado de configuração de rotas, paradas e horários.

Agregado de Recursos Operacionais: Veículos e Motoristas

Este agregado representa os ativos físicos e humanos da empresa. As entidades `Vehicle` e `Driver` são recursos independentes que são dinamicamente alocados às viagens. A modelagem através da entidade associativa `TripVehicle` foi uma decisão deliberada para conferir flexibilidade operacional. Em vez de vincular um motorista fixo a um veículo, a `TripVehicle` permite que a alocação seja feita por viagem, refletindo a realidade das operações de transporte. A inclusão de campos para motorista primário e secundário (`primaryDriverId`, `secondaryDriverId`) nesta entidade já prevê a complexidade de viagens longas ou que necessitam de revezamento, demonstrando a escalabilidade do modelo.

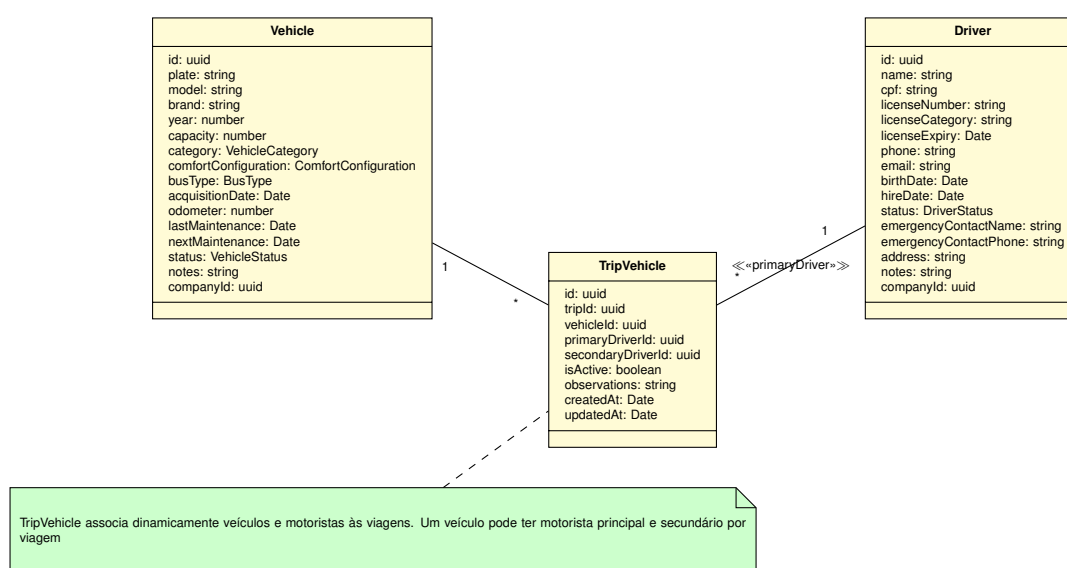


Figura 4 – Agregado de recursos operacionais - frota e motoristas.

Agregado de Operação e Vendas: Viagens e Bilhetes

Este é o agregado central que conecta a configuração (Rotas) com os recursos (Veículos/Motoristas) e a atividade comercial (Bilhetes). A entidade `Trip` é a instância de uma `Route` em uma data e horário específicos. Uma decisão de modelagem importante na entidade `Ticket` foi a inclusão de campos para registrar os pontos de embarque e desembarque (`boardingStopId`, `landingStopId`), permitindo a venda de bilhetes para trechos seccionados da viagem. Além disso, a flexibilidade foi aumentada ao permitir que os pontos de embarque/desembarque possam ser tanto uma parada pré-cadastrada quanto uma localização customizada (com descrição e coordenadas), atendendo à demanda do transporte alternativo onde pontos de encontro podem ser flexíveis.

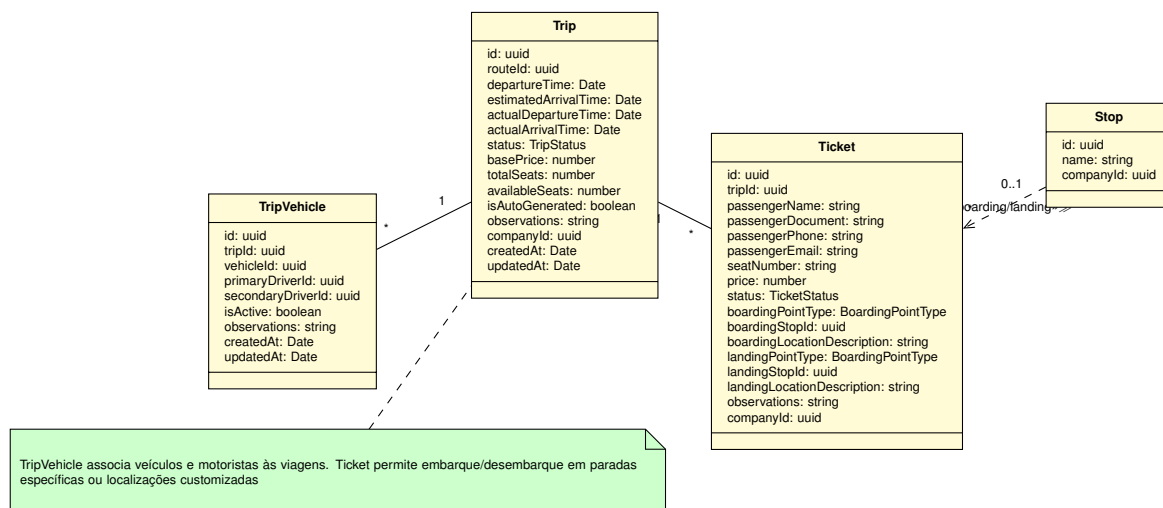


Figura 5 – Agregado de operação - viagens, recursos e vendas.

5.3 ARQUITETURA DO BACKEND

O backend implementa uma arquitetura modular baseada no framework NestJS 11, seguindo os princípios de *Separation of Concerns* e *Dependency Injection*. A aplicação é estruturada em 10 módulos de domínio, com camadas bem definidas de segurança, validação e persistência.

5.3.1 Organização Modular

O sistema está organizado em três grupos funcionais de módulos, conforme ilustrado na Figura 6.

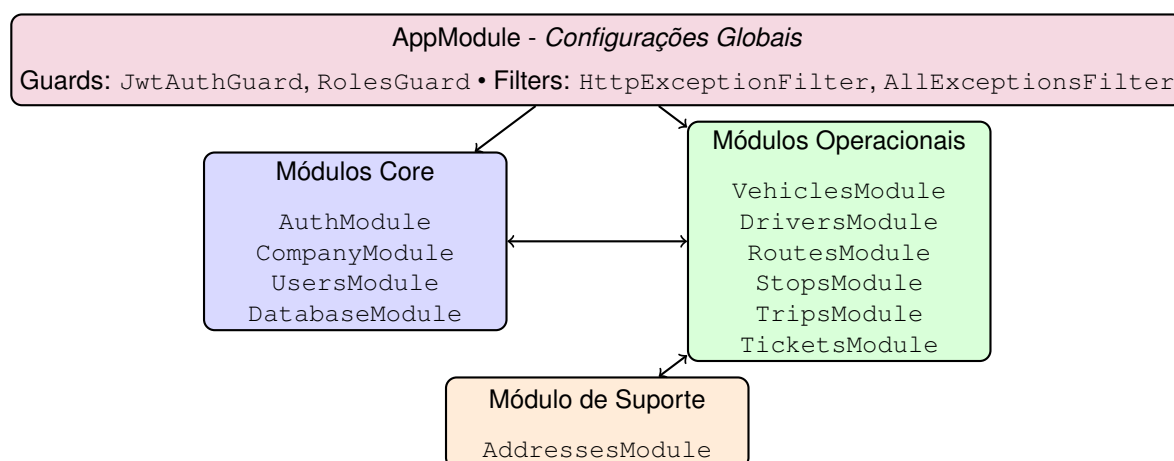


Figura 6 – Organização modular do backend ViaBus.

A organização modular agrupa os componentes por responsabilidade funcional. Os Módulos Core gerenciam autenticação, usuários, empresas e configuração de banco de dados. Os Módulos Operacionais implementam as regras de negócio específicas do domínio de transporte. O Módulo de Suporte fornece funcionalidades

auxiliares como geocodificação. O `AppModule` centraliza as configurações globais de segurança e tratamento de erros, aplicadas a todos os módulos.

5.3.2 Pipeline de Processamento

Cada requisição HTTP passa por um pipeline estruturado de middleware, guards, interceptors e filters, garantindo segurança e consistência.

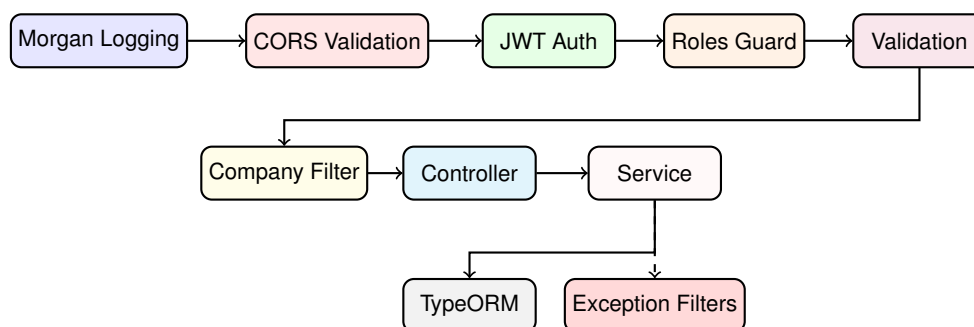


Figura 7 – Pipeline de processamento de requisições.

O pipeline garante que cada requisição passe por validações sequenciais obrigatórias. Inicialmente, o Morgan registra logs e o CORS valida origens permitidas. A autenticação via `JwtAuthGuard` verifica tokens válidos, seguida da autorização por `RolesGuard` que confirma permissões do usuário. O `ValidationPipe` valida DTOs de entrada. O `CompanyFilterInterceptor` injeta o contexto multiempresa antes do processamento pelo `Controller` e `Service`. Finalmente, o `TypeORM` persiste dados e os `Exception Filters` tratam erros de forma padronizada.

5.3.3 Fluxo de Autenticação

O sistema utiliza autenticação baseada em JWT com suporte a multi-tenancy. A Figura 8 detalha o processo completo.

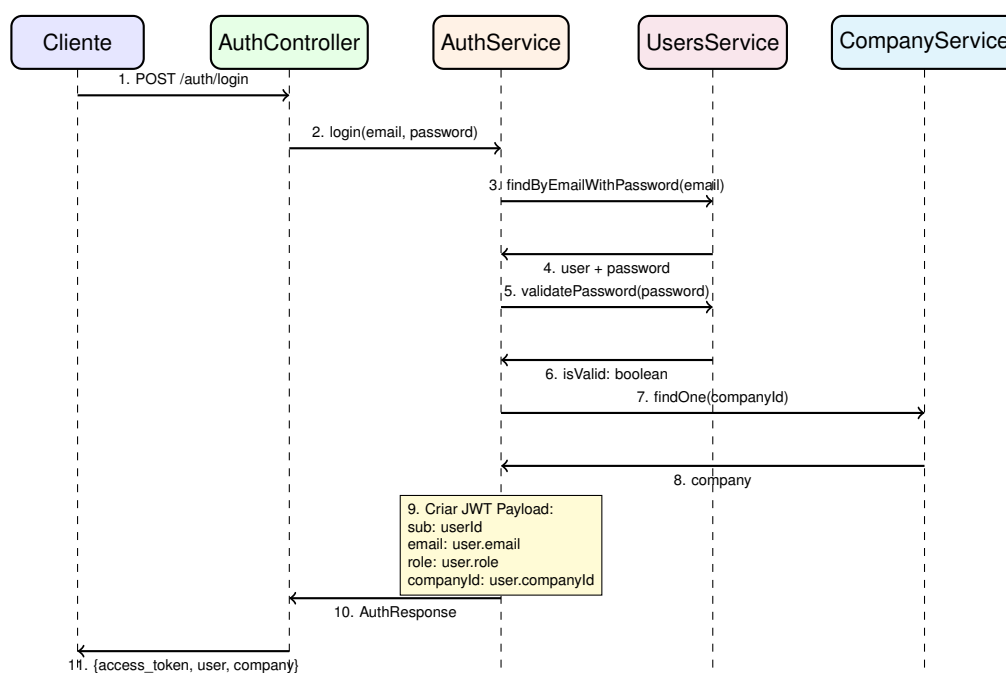


Figura 8 – Diagrama de sequência - fluxo de autenticação.

O fluxo de autenticação implementa um processo seguro e eficiente em 11 etapas. O cliente envia credenciais via POST para o `AuthController`, que delega ao `AuthService` a validação. O `UsersService` busca o usuário por email (incluindo senha hash), valida a senha fornecida e retorna o resultado. Confirmada a autenticação, o `AuthService` consulta os dados da empresa via `CompanyService` e cria o payload JWT contendo informações do usuário e contexto empresarial. A resposta final inclui o token de acesso, dados do usuário e informações da empresa, permitindo que o frontend mantenha o contexto de sessão multiempresa.

5.3.4 Arquitetura Multi-tenant

O sistema implementa isolamento de dados por empresa através de uma arquitetura multi-tenant baseada em filtros automáticos. A solução utiliza três componentes principais que trabalham em conjunto para garantir separação completa entre inquilinos:

- `BaseCompanyService<T>`: Classe abstrata que implementa operações CRUD com filtro automático por `companyId`
- `CompanyFilterInterceptor`: Interceptor que extrai o `companyId` do token JWT e o injeta no contexto da requisição
- `@CurrentUser`: Decorator que facilita o acesso aos dados do usuário autenticado

A implementação garante que todas as operações de banco de dados incluam automaticamente a cláusula `WHERE companyId = ?`, eliminando a possibilidade de

vazamento de dados entre empresas. O padrão é aplicado consistentemente em todos os módulos operacionais (usuários, motoristas, veículos, rotas, paradas, viagens e passagens).

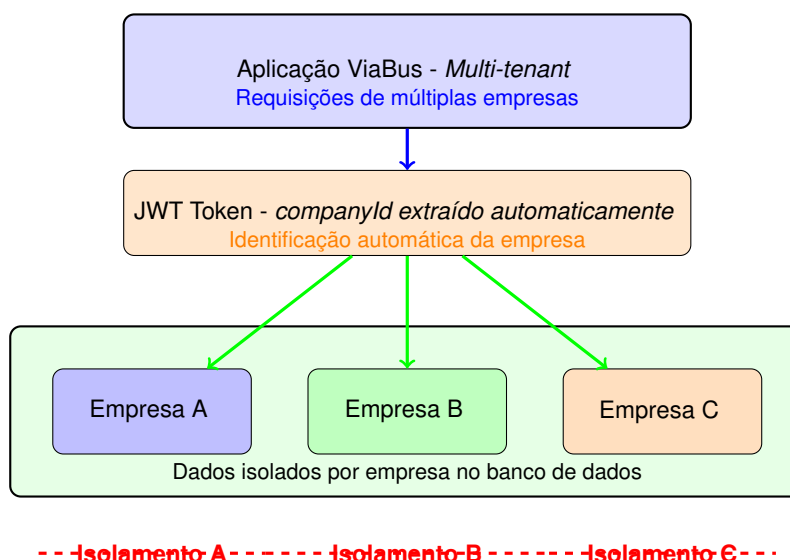


Figura 9 – Arquitetura multi-tenant: uma aplicação serve múltiplas empresas com isolamento automático de dados.

5.4 ARQUITETURA DO FRONTEND

O frontend implementa uma arquitetura moderna baseada em Next.js 15 (React 19) com seu *App Router*, utilizando TypeScript para garantir uma tipagem estática completa. A aplicação foi estruturada seguindo padrões de arquitetura limpa e uma organização orientada a domínios de negócio, visando a manutenibilidade e a escalabilidade da interface.

5.4.1 Stack Tecnológico e Organização

Para garantir performance, uma boa experiência de usuário e manutenibilidade, o frontend utiliza um conjunto integrado de tecnologias. A base da aplicação é o Next.js 15, que, junto ao React 19, oferece uma plataforma sólida para a renderização da interface. A construção da UI segue um *design system* customizado, empregando componentes primitivos do Radix UI para assegurar a acessibilidade e o Tailwind CSS para uma estilização *utility-first* ágil e consistente. A gestão de formulários é responsabilidade do React Hook Form combinado com o Zod para validação tipada de dados. A autenticação é gerenciada pelo NextAuth v4, enquanto as funcionalidades geográficas são implementadas com Leaflet. O estado global da aplicação, por sua vez, é controlado de forma eficiente pela Context API nativa do React.

5.4.2 Arquitetura de Roteamento

A aplicação utiliza o *App Router* do Next.js 15, que permite um roteamento dinâmico e baseado em arquivos. A estrutura de rotas foi projetada para suportar a natureza multiempresa do sistema, seguindo o padrão `/dashboard/[company]/recurso`. Essa abordagem garante o isolamento dos dados no nível da URL, onde cada empresa acessa seus próprios recursos, como `/motoristas` ou `/rotas`, dentro de seu próprio escopo. O sistema diferencia entre páginas públicas, como `/login` e `/criar-empresa`, e as rotas privadas do *dashboard*. A proteção de rotas é garantida por um componente de ordem superior (*wrapper*) chamado `ProtectedPage`, que valida a autenticação e as permissões do usuário antes de renderizar a página solicitada.

5.4.3 Gerenciamento de Estado e Contextos

O gerenciamento do estado global da aplicação é implementado através de uma hierarquia de *Contextos* React, conforme ilustrado na Figura 10. Essa abordagem permite um compartilhamento de dados eficiente e desacoplado entre os componentes. No topo da hierarquia, o `SessionProvider` gerencia a sessão do NextAuth e os tokens JWT. Abaixo dele, o `ThemeProvider` controla o tema visual da aplicação (claro/escuro). Em seguida, o `AuthProvider` expõe os dados do usuário autenticado e da empresa associada. O `CompanyProvider` mantém o estado específico da empresa que está sendo visualizada no *dashboard*, enquanto o `SidebarProvider`, na camada mais interna, gerencia estados de UI, como a visibilidade do menu lateral.

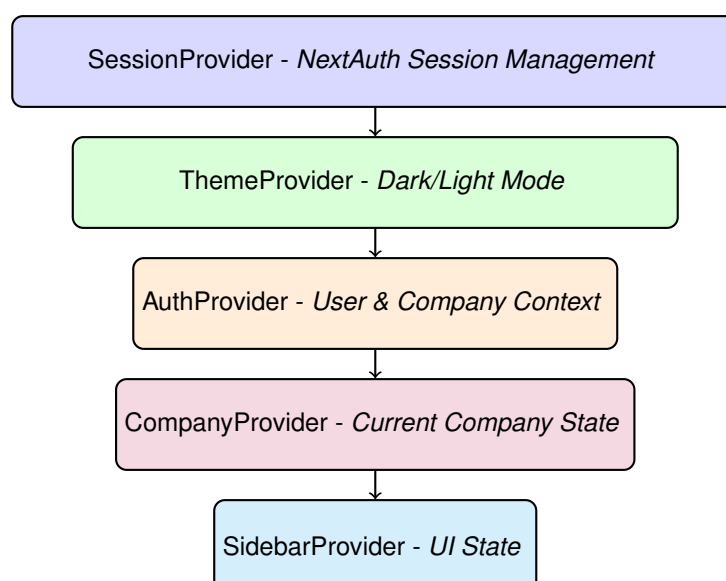


Figura 10 – Hierarquia de providers e contextos do frontend.

5.4.4 Sistema de Autenticação e Comunicação

A comunicação com o backend é centralizada para garantir consistência e segurança. A autenticação integra o NextAuth com a API NestJS através de tokens JWT, e todas as requisições HTTP são gerenciadas por um serviço centralizado, o `api.service.ts`. Este serviço encapsula responsabilidades críticas: ele anexa automaticamente os *Bearer tokens* de autenticação em todas as chamadas a rotas protegidas, trata respostas de erro de forma padronizada (redirecionando para a página de login em caso de erro 401, por exemplo), processa as respostas da API para extrair os dados e valida a sessão do usuário antes de cada requisição, otimizando a comunicação com o servidor.

5.4.5 Arquitetura de Componentes

O sistema de componentes foi projetado em camadas, seguindo uma lógica de composição que vai do mais simples ao mais complexo. Na base, estão os Primitivos de UI do Radix UI, que fornecem comportamento e acessibilidade sem estilização. Sobre eles, foi construído um Design System de componentes reutilizáveis e tipados (localizados em `components/ui/`), que definem a aparência visual da aplicação. Estes são, então, utilizados para compor os Componentes de Domínio, que são específicos de cada funcionalidade, como formulários de cadastro ou tabelas de dados. Por fim, as Páginas são montadas através da composição desses componentes, organizadas dentro de Layouts reutilizáveis que fornecem a estrutura de navegação e os provedores de contexto necessários.

5.4.6 Tipagem e Validação

O TypeScript é utilizado de forma extensiva para garantir a integridade dos dados em toda a aplicação. Um diretório central de tipos (*Types Directory*) contém as definições para cada entidade de negócio (`User`, `Company`, etc.), servindo como uma fonte única da verdade para a estrutura de dados. Essa tipagem se estende às respostas da API, garantindo que os dados recebidos do backend correspondam ao esperado. No lado do cliente, a validação de formulários é realizada com *Schemas Zod*, que asseguram a consistência dos dados antes do envio. Essa abordagem cria uma arquitetura com tipagem estática de ponta a ponta (*end-to-end*), desde o banco de dados até a interface do usuário.

5.5 ESTRATÉGIA DE IMPLANTAÇÃO

A implantação do sistema ViaBus foi planejada em uma arquitetura de nuvem, utilizando um conjunto de serviços da Google Cloud Platform (GCP) para automação e

escalabilidade. A estratégia adotada consistiu em quatro etapas principais: containerização da aplicação, implantação dos serviços em ambiente *serverless*, gerenciamento do banco de dados e automação da entrega (CI/CD).

Para garantir a portabilidade entre os ambientes, a aplicação foi containerizada com Docker. Foram criados arquivos `Dockerfile` específicos para o *backend* (NestJS) e o *frontend* (Next.js), utilizando a abordagem de *multi-stage builds* para gerar imagens otimizadas e seguras. A orquestração do ambiente de desenvolvimento local foi realizada com o Docker Compose.

Os contêineres da aplicação foram implantados na nuvem como dois serviços independentes no Google Cloud Run, uma plataforma de computação *serverless*. Essa configuração permite que o *frontend* e o *backend* escalem de forma autônoma com base na demanda de requisições. A comunicação externa é garantida via HTTPS, com certificados TLS/SSL gerenciados automaticamente pela plataforma.

Para a persistência dos dados, foi utilizada uma instância gerenciada do PostgreSQL no serviço Google Cloud SQL. A comunicação entre a API executada no Cloud Run e o banco de dados ocorre em uma rede segura e privada, com criptografia dos dados em trânsito e em repouso. A responsabilidade por tarefas de manutenção, como backups e atualizações, é delegada ao próprio serviço da GCP.

Por fim, o ciclo de vida da aplicação foi automatizado com práticas de Integração e Entrega Contínuas (CI/CD). Foram estruturados ambientes isolados de desenvolvimento e produção na GCP. O gerenciamento de informações sensíveis, como chaves de API e credenciais de banco de dados, foi centralizado no Google Secret Manager. Um *pipeline* no Google Cloud Build foi configurado para automatizar o processo de construção das imagens Docker e a implantação de novas versões no Cloud Run a cada atualização no repositório principal do Git.

6 MÓDULOS E FUNCIONALIDADES DO FRONTEND

O frontend do sistema ViaBus oferece uma interface moderna e intuitiva para gestão completa de operações de transporte rodoviário. A aplicação é desenvolvida em Next.js 15 com React 19, proporcionando uma experiência de usuário fluida e responsiva.

6.1 AUTENTICAÇÃO E GESTÃO DE EMPRESA

6.1.1 Tela de Login e Registro

A autenticação do sistema é realizada através de uma interface limpa e moderna. A tela de login apresenta campos para email e senha, com validação em tempo real e feedback visual para o usuário. O sistema utiliza NextAuth v4 para gerenciamento de sessão, garantindo segurança e persistência do estado de autenticação.




A imagem mostra a interface de login do sistema ViaBus. No topo, o logo "ViaBus" é exibido em negrito, seguido pelo subtítulo "Sistema de Gerenciamento de Transporte". Abaixo, há dois botões: "Entrar" (destacado em branco) e "Criar Conta" (em cinza). Seguem-se os campos de entrada: "Email" com o placeholder "seu@email.com" e ícone de envelope, e "Senha" com pontos para ocultar o texto e ícone de olho para alternar visibilidade. Um botão "Entrar" em azul sólido está posicionado abaixo dos campos. No rodapé, há o texto de copyright: "© 2025 ViaBus. Todos os direitos reservados."

Figura 11 – Tela de login do sistema ViaBus.

6.1.2 Fluxo de Criação da Empresa

Após o primeiro acesso, usuários sem empresa associada são direcionados para o fluxo de criação de empresa. Este processo inclui cadastro da empresa com informações básicas (nome, CNPJ, endereço), configuração inicial de parâmetros operacionais e validação automática de dados.



O formulário de criação de empresa é apresentado em uma interface limpa e moderna. No topo, há um ícone circular azul com um símbolo de documento. Abaixo dele, o título "Criar Empresa" é exibido em uma fonte bold, seguido por uma instrução: "Configure sua empresa para começar a usar o ViaBus". O formulário contém três campos de entrada: "Nome da Empresa" com o placeholder "Digite o nome da sua empresa", "CNPJ" com o placeholder "00.000.000/0000-00", e "Telefone" com o placeholder "(00) 00000-0000". Abaixo dos campos, há dois botões: um botão azul "Criar Empresa" e um botão branco "Voltar ao Login" com uma seta para trás. No rodapé, o texto "© 2024 ViaBus. Sistema de Gerenciamento de Transporte" é exibido em uma fonte menor.

Figura 12 – Formulário de criação de empresa.

6.2 PAINEL DE CONTROLE (DASHBOARD)

O dashboard principal oferece uma visão consolidada das operações da empresa, apresentando métricas importantes e acesso rápido aos módulos principais.

6.2.1 Visão Geral

O painel de controle exibe cards informativos com estatísticas em tempo real: total de viagens, passagens vendidas, veículos ativos, motoristas disponíveis, rotas ativas e paradas cadastradas.

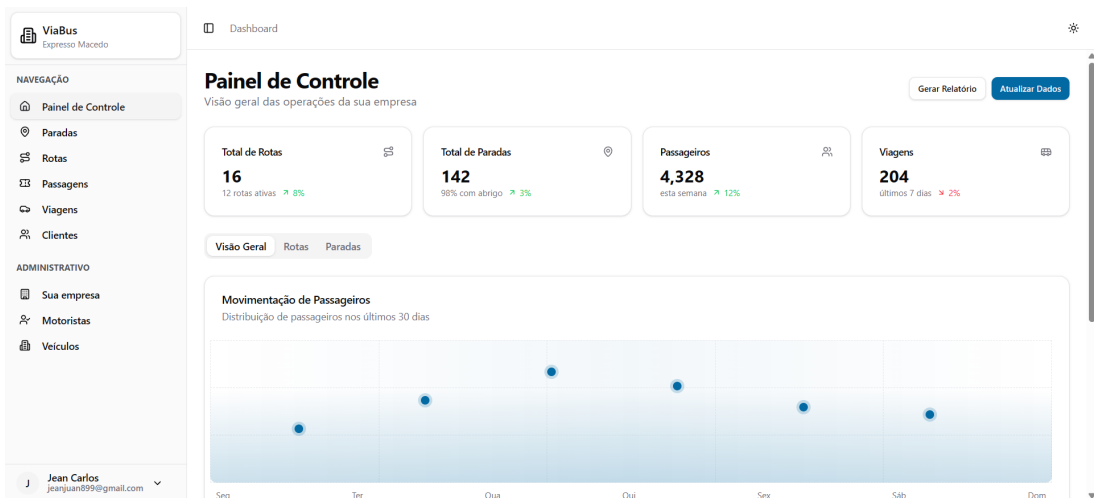


Figura 13 – Painel de controle principal com métricas e navegação.

6.3 MÓDULO DE PARADAS

O módulo de paradas permite o cadastro e gerenciamento de pontos de embarque e desembarque, com integração a mapas interativos.

6.3.1 Listagem de Paradas

A tela de listagem apresenta todas as paradas cadastradas em formato de tabela, com funcionalidades de busca, filtros e ações rápidas (editar, visualizar, excluir).

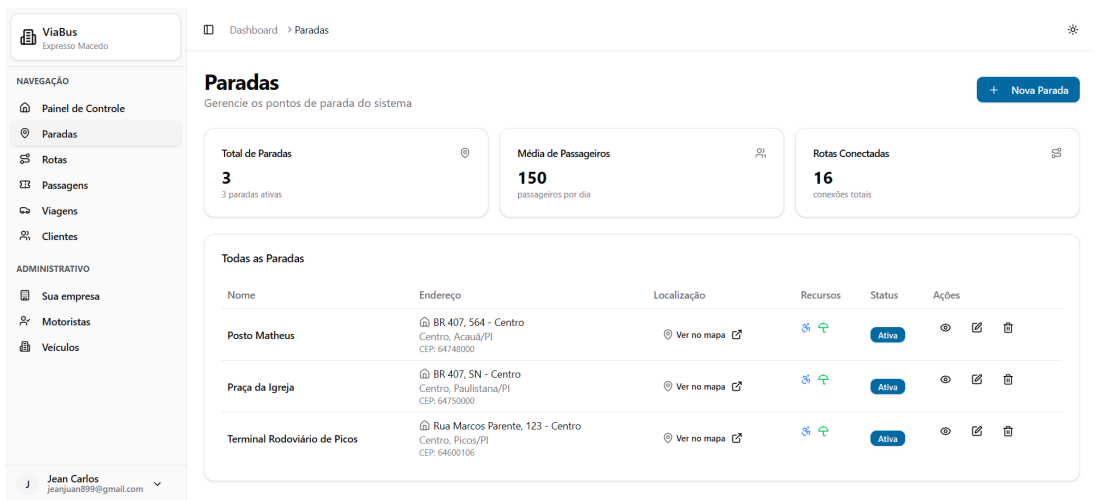


Figura 14 – Listagem de paradas com funcionalidades de busca e filtros.

6.3.2 Criação e Edição de Paradas

O formulário de cadastro de paradas integra um mapa interativo baseado em Leaflet, permitindo seleção geográfica por clique, arrastar marcador para ajuste fino, validação de endereço e informações detalhadas como nome, descrição e horários de funcionamento.

ViaBus
Expresso Macedo

Dashboard > Paradas > Nova

Nova Parada

Cadastre um novo ponto de parada no sistema

Formulário de Cadastro

Nome da Parada Ex: Terminal Central	CEP Ex: 64000000
Rua Ex: Avenida Frei Serafim	Número Ex: 123
Complemento Ex: Bloco A, Apto 101	Bairro Ex: Centro
Cidade Ex: Teresina	Estado Ex: PI

Localização no Mapa
Selecione o ponto exato da parada no mapa

Mapa interativo mostrando a localização da parada no mapa.

Figura 15 – Formulário de cadastro de parada com mapa interativo.

6.4 MÓDULO DE ROTAS

O módulo de rotas permite a criação de itinerários conectando paradas em sequência lógica, incluindo definição de paradas em ordem de visitação, cálculo automático de distâncias e configuração de preços por trecho.

Criar Nova Rota

Informações da Rota

Nome da Rota
Ex: Centro - Aeroporto

Status
☒ Rota Ativa

Descrição
Descrição da rota...

Paradas da Rota 3 paradas

Buscar parada por nome ou endereço...

1ª Parada **Posto Matheus** ×
BR 407, Centro
☐ Acessível ☐ Abrigo
Horário de Partida (opcional) --:--

2ª Parada **Praça da Igreja** ×
BR 407, Centro
☐ Acessível ☐ Abrigo
Horário de Partida (opcional) --:--

3ª Parada **Terminal Rodoviário de Picos** ×
Rua Marcos Parente, Centro
☐ Acessível ☐ Abrigo

Figura 16 – Interface de criação de rota com seleção de paradas.

6.5 MÓDULO DE VEÍCULOS

O módulo de veículos gerencia a frota da empresa, permitindo cadastro completo (placa, modelo, capacidade, ano), controle de status operacional (ativo, em manutenção, inativo), histórico de viagens e filtros avançados.

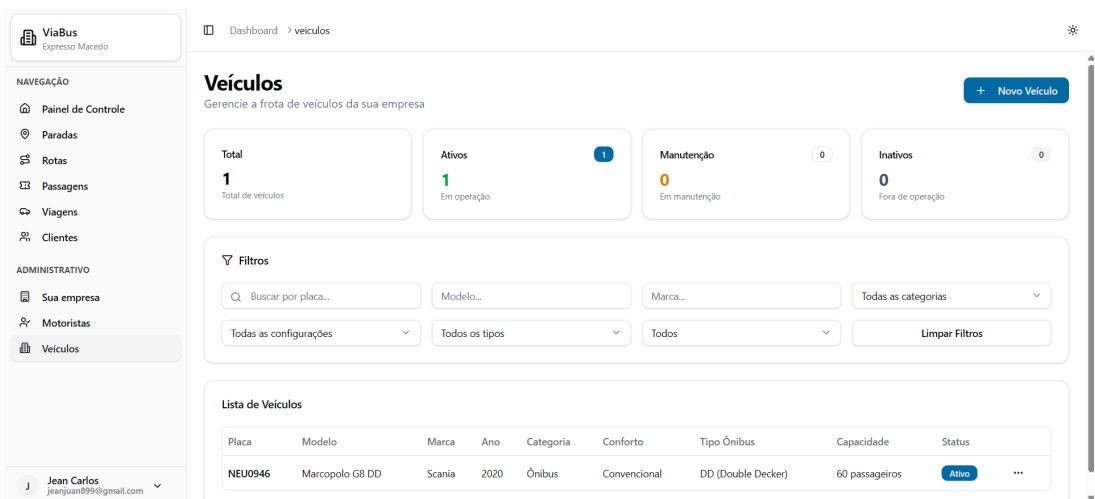


Figura 17 – Listagem de veículos com status operacional.

6.6 MÓDULO DE MOTORISTAS

O módulo de motoristas gerencia a equipe de condutores, incluindo dados pessoais (nome, CPF, CNH, contatos), informações profissionais (categoria da CNH, experiência), status de disponibilidade e histórico de viagens.

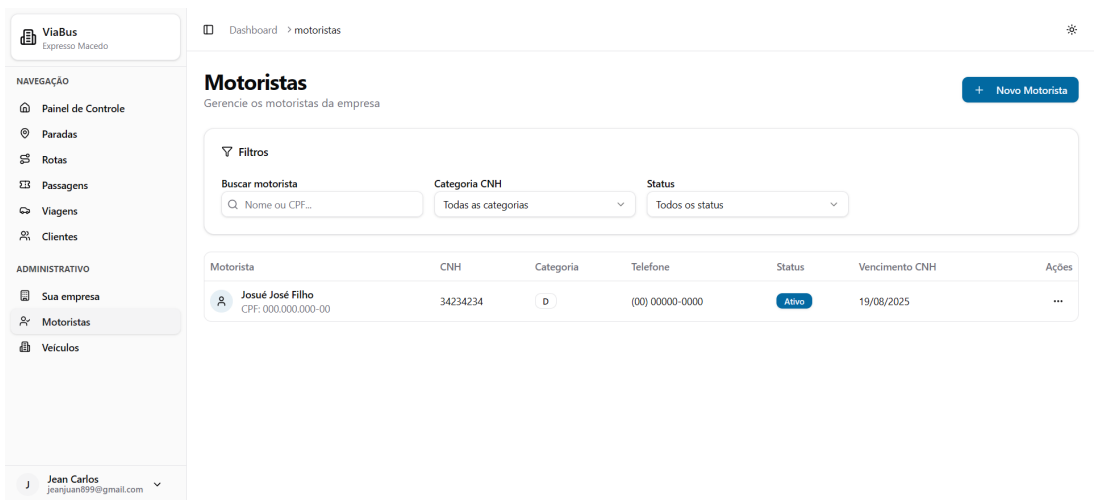


Figura 18 – Interface de gerenciamento de motoristas.

6.7 MÓDULO DE VIAGENS

O módulo de viagens permite o agendamento e gerenciamento de viagens através de um fluxo estruturado: seleção da rota, definição de data e horário, associação de veículo, designação de motorista, configuração de preços e confirmação final.

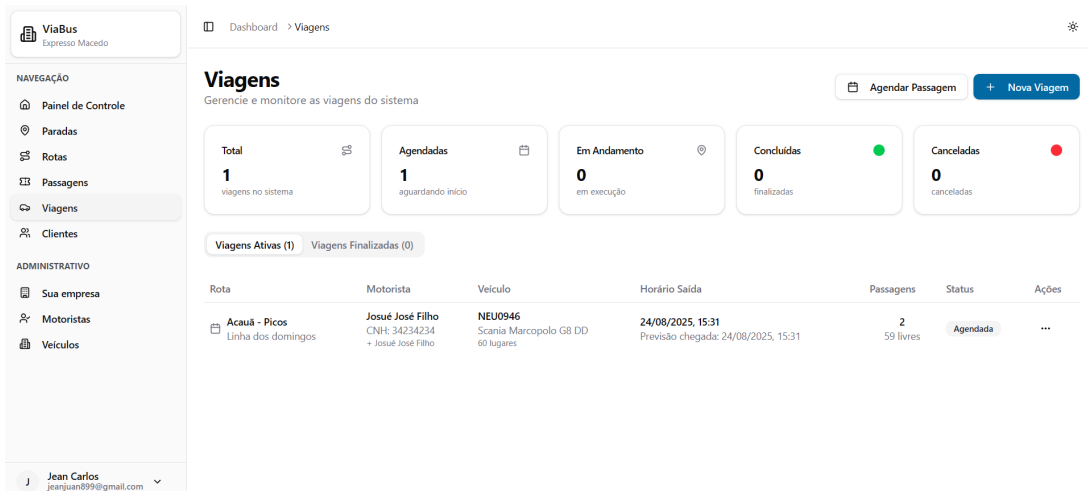


Figura 19 – Processo de agendamento de viagem.

6.8 MÓDULO DE VENDA DE PASSAGENS

O módulo de passagens implementa um assistente (wizard) intuitivo para venda de passagens, guiando o usuário através de cinco etapas sequenciais.

6.8.1 Assistente de Venda de Passagens

6.8.1.1 1. Seleção de Rota

Primeira etapa com listagem de rotas disponíveis, filtros por data, informações da rota (paradas, horários, preços) e verificação de disponibilidade de assentos.

Agendar Nova Passagem

✕

Selecione a rota, data e preencha os dados do passageiro. O sistema criará automaticamente uma viagem se necessário.

Rota → Data → Passageiro → Locais

Selecione a Rota

Acauã - Picos

● Disponível

Linha dos domingos

🕒 Saída: 18:31

Disponibilidade

4 datas

31/08 07/09 14/09 + 1 mais

Paradas

- Terminal Rodoviário de Picos 10:00
- Praça da Igreja 19:00

+ 1 paradas

Selecionar Rota

Figura 20 – Seleção de rota no wizard de passagens.

6.8.1.2 2. Seleção de Data e Horário

Segunda etapa com calendário interativo, horários disponíveis e informação de capacidade restante.

Agendar Nova Passagem

Selecione a rota, data e preencha os dados do passageiro. O sistema criará automaticamente uma viagem se necessário.

Rota → Data → Passageiro → Locais

Voltar

Selecione a Data

Acauã - Picos

Linha dos domingos

Horário de saída: 18:31

agosto 2025						
Dom	Seg	Ter	Qua	Qui	Sex	Sáb
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Próximas datas disponíveis:

domingo, 31 de agosto

18:31

domingo, 7 de setembro

18:31

domingo, 14 de setembro

18:31

Figura 21 – Seleção de data e horário.

6.8.1.3 3. Informações do Passageiro

Terceira etapa para coleta de dados pessoais (nome completo, CPF), contatos (telefone, email) e validação automática de CPF.

Agendar Nova Passagem ×

Selecione a rota, data e preencha os dados do passageiro. O sistema criará automaticamente uma viagem se necessário.

[Rota](#) → [Data](#) → [Passageiro](#) → [Locais](#) Voltar

Dados do Passageiro

Informações Pessoais

Nome Completo *	Telefone *
<input type="text" value="Digite o nome completo"/>	<input type="text" value="(11) 99999-9999"/>
CPF	Email
<input type="text" value="000.000.000-00"/>	<input type="text" value="email@exemplo.com"/>

● Campos obrigatórios

● Nome completo ● Telefone

Continuar

Figura 22 – Cadastro de informações do passageiro.

6.8.1.4 4. Seleção de Locais de Embarque e Desembarque

Quarta etapa com seleção de paradas oficiais ou definição de locais personalizados, cálculo de preço baseado na distância e observações para o motorista.

Agendar Nova Passagem

Selecione a rota, data e preencha os dados do passageiro. O sistema criará automaticamente uma viagem se necessário.

Rota → Data → Passageiro → Locais

Voltar

Pontos de Embarque e Desembarque

Ponto de Embarque

☒ Parada Oficial ☐ Local Específico

Selecione a parada

Escolha uma parada

Ponto de Desembarque

☒ Parada Oficial ☐ Local Específico

Selecione a parada

Escolha uma parada

Preço (R\$)

0

Observações

Informações adicionais sobre o embarque/desembarque...

Confirmar Agendamento

Figura 23 – Seleção de locais de embarque e desembarque.

6.8.1.5 5. Confirmação e Pagamento

Etapa final com resumo da viagem, detalhes da passagem, preço final e geração automática do bilhete.

Agendar Nova Passagem

Selecione a rota, data e preencha os dados do passageiro. O sistema criará automaticamente uma viagem se necessário.

Rota → Data → Passageiro → Locais

Voltar

Confirmação do Agendamento

Detalhes da Viagem

Data

domingo, 31 de agosto de 2025

Horário

18:31

Rota

Acauã - Picos

Dados do Passageiro

Nome

Jean Carlos

Telefone

89994077294

Pontos de Embarque e Desembarque

Embarque

Parada não selecionada

Desembarque

Parada não selecionada

Valor e Observações

Preço

R\$ 60,00

Confirmar Agendamento

Figura 24 – Confirmação final e geração da passagem.

6.9 CARACTERÍSTICAS DA INTERFACE

6.9.1 Design Responsivo

A interface é totalmente responsiva, adaptando-se a diferentes tamanhos de tela: desktop com layout completo, tablet com adaptação automática e mobile com interface otimizada.

6.9.2 Experiência do Usuário

O sistema prioriza a usabilidade através de navegação intuitiva, feedback visual com notificações e estados de carregamento, validação em tempo real e suporte à acessibilidade.

6.9.3 Integração com Mapas

A integração com mapas interativos (Leaflet) oferece visualização geográfica precisa, seleção por clique, funcionalidade de arrastar e soltar para ajuste fino e controles intuitivos de zoom e navegação.

O frontend do ViaBus proporciona uma experiência completa e profissional para gestão de operações de transporte, combinando funcionalidade avançada com interface intuitiva e moderna.

7 RESULTADOS E DISCUSSÃO

7.1 VERIFICAÇÃO DE REQUISITOS

A primeira etapa da avaliação consistiu na verificação sistemática do grau de implementação dos Requisitos Funcionais (RF) e Não Funcionais (RNF) definidos no Capítulo 3. A análise buscou constatar se as funcionalidades e características planejadas foram efetivamente traduzidas em software. As Tabelas 7 e 8 apresentam o resultado consolidado desta verificação.

Tabela 7 – Verificação dos Requisitos Funcionais (RF)

Requisito	Status	Observações e Evidências
RF01 – Registro	Implementado	Fluxo de cadastro de usuários disponível na API de autenticação, com validação de dados.
RF02 – Login	Implementado	Autenticação via e-mail e senha com emissão de JWT e proteção de rotas.
RF03 – Permissões	Implementado	Controle de acesso por perfis/roles com guards no backend e proteção de rotas no frontend.
RF04 – Empresas	Implementado	Criação e gestão de perfis de empresa disponíveis; página de criação de empresa no frontend.
RF05 – Isolamento	Implementado	Interceptor/serviço central aplica filtro por <i>companyId</i> do usuário em todas as requisições.
RF06 – Motoristas	Implementado	CRUD completo de motoristas no módulo dedicado.
RF07 – Veículos	Implementado	CRUD completo de veículos no módulo dedicado.
RF08 – Paradas	Implementado	CRUD de pontos de parada com armazenamento de geolocalização.
RF09 – Rotas	Implementado	Criação e manutenção de rotas com sequência de paradas.
RF10 – Horários	Parcialmente Implementado	Associação de horários e preços às rotas; serviços de <i>schedules</i> no frontend.
RF11 – Mapas	Parcialmente Implementado	Visualização de rotas e paradas disponível; ausência de mapa interativo avançado em algumas telas.
RF12 – Agendamento de Viagens	Implementado	Criação de viagens baseada em rotas e horários no módulo de viagens.
RF13 – Atribuição	Implementado	Associação de veículos e motoristas às viagens.
RF14 – Status de Viagens	Parcialmente Implementado	Atualização de status disponível; sem comunicação em tempo real por WebSocket.
RF15 – Wizard de Venda	Implementado	Fluxo guiado de venda com <i>wizard</i> no frontend.
RF16 – Embarque/Desembarque	Implementado	Seleção flexível de pontos ao longo da rota no ato da venda.
RF17 – Passageiros	Implementado	Coleta e armazenamento dos dados completos dos passageiros.
RF18 – Anti-Overbooking	Implementado	Verificação de disponibilidade antes da emissão; criação de ticket bloqueada com erro 409 quando lotado.
RF19 – Listas	Implementado	Geração/visualização de listas de passageiros por viagem nas telas operacionais.
RF20 – Busca	Parcialmente Implementado	Filtros múltiplos disponíveis nas listagens; oportunidades de ampliar critérios e combinações.
RF21 – Ocupação em Tempo Real	Parcialmente Implementado	Visualização de ocupação por viagem; ausência de atualização <i>push</i> em tempo real.

Tabela 8 – Verificação dos Requisitos Não Funcionais (RNF)

Requisito	Status	Observações e Evidências
RNF01 – Responsividade	Implementado	Interface responsiva, componentes padronizados e adaptação a diferentes tamanhos de tela.
RNF02 – Navegação	Parcialmente Implementado	Fluxos principais claros; avaliação heurística indica melhorias na prevenção de erros.
RNF03 – Acessibilidade	Parcialmente Implementado	Formulários com indicadores de progresso; faltam recursos avançados de acessibilidade.
RNF04 – Autenticação	Implementado	Login seguro com JWT e controle de sessão.
RNF05 – Autorização	Implementado	Acesso baseado em perfis com guards e verificação centralizada.
RNF06 – Isolamento	Implementado	Separação de dados por empresa garantida via filtro automático por <i>companyId</i> .
RNF07 – Validação	Implementado	Validação de entrada no backend (DTOs/validators) e no frontend.
RNF08 – Desempenho de Carregamento	Implementado	Páginas com tempos de resposta adequados nos fluxos críticos.
RNF09 – Otimização de Consultas	Parcialmente Implementado	Consultas indexadas nos módulos principais; há espaço para otimizações adicionais.
RNF10 – Escalabilidade	Parcialmente Implementado	Arquitetura modular; ausência de escalonamento horizontal automático e fila de mensagens.
RNF11 – Arquitetura	Implementado	Código organizado por módulos com camadas bem definidas.
RNF12 – Padrões	Implementado	Padrões de codificação consistentes e lint configurado.
RNF13 – Documentação	Parcialmente Implementado	Documentação básica presente; falta detalhamento abrangente de API e decisões de projeto.
RNF14 – Containerização	Implementado	Dockerfile no frontend e backend; orquestração via <i>docker-compose</i> .
RNF15 – Configuração de Ambientes	Implementado	Suporte a múltiplos ambientes com variáveis de ambiente.

7.2 AVALIAÇÃO HEURÍSTICA

A segunda etapa da avaliação do protótipo consistiu em uma Avaliação Heurística, método de inspeção de usabilidade consolidado por Nielsen (1994). Atuando como avaliador especialista, foram analisados os principais fluxos de interação do sistema, como o cadastro de rotas e a venda de passagens, com o objetivo de identificar potenciais problemas e pontos de aderência a boas práticas de design de interface. A seguir, são discutidos os achados mais relevantes, agrupados pelas heurísticas correspondentes.

Visibilidade do status do sistema (Heurística 1) O sistema se destaca na aplicação desta heurística. Em operações que demandam tempo, como o carregamento de tabelas de dados, a interface exibe componentes de skeleton loading (Figura X, a ser inserida por você, mostrando a tabela "carregando"), comunicando claramente ao usuário que uma ação está em progresso. Adicionalmente, ações de sucesso ou falha, como salvar um novo veículo, disparam notificações do tipo toast no canto da tela, como ilustrado na Figura ??, fornecendo feedback imediato e não intrusivo.

Correspondência entre o sistema e o mundo real (Heurística 2) A plataforma utiliza uma linguagem e iconografia familiar ao usuário-alvo. Termos como "Frota", "Motoristas", "Paradas" e "Rotas" são diretos e correspondem à terminologia do setor. A utilização de mapas interativos para a gestão de paradas e visualização de rotas, como visto na Figura ??, cria uma representação visual que espelha diretamente a operação

logística do mundo real, facilitando o entendimento e a manipulação dos dados.

Consistência e padronização (Heurística 4) Este é um dos pontos mais fortes do protótipo. A utilização sistemática da biblioteca de componentes shadcn/ui garante uma alta consistência visual e de interação. Elementos como botões, formulários, tabelas e modais, como o de cadastro de nova parada (Figura ??), mantêm uma identidade e um comportamento padronizados em todos os módulos. Essa consistência reduz a carga cognitiva do usuário e acelera a curva de aprendizado da ferramenta.

Prevenção de erros (Heurística 5) Neste ponto, foram identificadas oportunidades de melhoria. No formulário de criação de rotas (Figura ??), o sistema atualmente permite que o usuário avance e tente salvar uma rota sem ter selecionado nenhuma parada. Isso representa um erro potencial, pois uma rota sem paradas é funcionalmente inútil e pode gerar dados inconsistentes. Uma melhoria recomendada seria a implementação de uma validação no frontend que desabilite o botão "Salvar" ou exiba uma mensagem de alerta enquanto a lista de paradas da rota estiver vazia.

Reconhecimento em vez de memorização (Heurística 6) A interface faz um bom trabalho ao manter as opções e informações visíveis. O menu lateral persistente (Figura ??) permite que o usuário navegue entre os módulos sem precisar memorizar o caminho. No entanto, no fluxo de venda de passagens, o sistema poderia melhorar ao exibir um resumo da seleção (ex: "Rota: Picos x Teresina, Data: 29/08/2025") em todos os passos do assistente, para que o usuário não precise memorizar as informações inseridas nos passos anteriores.

Estética e design minimalista (Heurística 8) A interface do ViaBus é limpa e funcional. Não há excesso de informações ou elementos visuais que possam distrair o usuário de suas tarefas. O uso de espaços em branco e a hierarquia visual clara, como visto no Dashboard (Figura 13), contribuem para um design minimalista que prioriza o conteúdo e a funcionalidade.

7.3 DISCUSSÃO DOS RESULTADOS

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

8 CONSIDERAÇÕES FINAIS

Nesta seção, faça uma reflexão sobre o projeto como um todo. Discuta os resultados alcançados, os desafios enfrentados e as lições aprendidas. Avalie se os objetivos foram atingidos e sugira melhorias ou trabalhos futuros que possam ser realizados com base no que foi desenvolvido.

APÊNDICE A – QUESTIONÁRIO DA PESQUISA DE MERCADO

Este apêndice apresenta a transcrição fiel do questionário, desenvolvido na plataforma Google Forms, que foi utilizado para o levantamento de requisitos do sistema ViaBus.

Título do Formulário: ViaBus: simplificando a gestão do seu transporte de passageiros (pesquisa de mercado)

PERGUNTAS

1. Em qual empresa ou serviço você atua? (Opcional)

Campo para resposta curta.

2. Qual o tamanho da frota de veículos?

Marcar apenas uma oval.

- 1 veículo
- 2 a 5 veículos
- 6 a 10 veículos
- 11 a 20 veículos
- Mais de 20 veículos

3. Quantas rotas (trechos) diferentes a empresa opera regularmente?

Marcar apenas uma oval.

- 1 a 2 rotas
- 3 a 5 rotas
- 6 a 10 rotas
- Mais de 10 rotas
- Não opero rotas fixas (apenas fretamento)

4. A operação é principalmente:

Marcar apenas uma oval.

- Intermunicipal (dentro do mesmo estado)
- Interestadual (entre estados diferentes)
- Ambos

5. Como é gerenciada a venda de passagens e a lista de passageiros hoje?

Marque todas que se aplicam.

- Caderno de anotações / Planilha em papel
- Planilhas no computador (Excel, Google Sheets)
- Mensagens diretas (WhatsApp, Telegram)
- Ligação telefônica
- Diretamente com o motorista, na hora do embarque
- Utilizo outro sistema/software
- Outro

6. Quais são os maiores desafios operacionais hoje para a empresa?

Marque todas que se aplicam.

- Realizar o embarque de passageiros com planilhas manuais.
- Controlar quais assentos já foram vendidos e quais estão livres.
- Perda de tempo organizando listas de passageiros e viagens.
- Prejuízo com assentos vazios (baixa ocupação)
- Comunicação com os motoristas sobre a rota e os passageiros.
- Receber os pagamentos (muito dinheiro em espécie, dificuldade com Pix, etc.).
- Outro

7. Quão valioso seria ter um sistema único para gerenciar rotas, horários, veículos, motoristas e passagens em um só lugar, acessível pelo celular ou computador?

Marcar apenas uma oval, em uma escala de 1 a 5.

- 1 2 3 4 5

8. Quais funcionalidades de um sistema como o ViaBus seriam mais importantes para o seu negócio?

Marque todas que se aplicam.

- Embarque facilitado através de um aplicativo/site, substituindo planilhas manuais.
- Painel para cadastrar e organizar minhas rotas, horários e preços.
- Possibilidade de agendamento externo (ex.: prefeituras às quais a empresa presta serviços).

- Venda de passagens online (com pagamento por Pix e cartão).
- Cadastro e gestão de motoristas e veículos.
- Relatórios financeiros (vendas por período, por rota, etc.).
- Outro

9. Se você pudesse oferecer aos seus passageiros um aplicativo para eles mesmos comprarem e acompanharem a viagem (ver horário, paradas, etc.), você acredita que isso seria um diferencial contra seus concorrentes?

Marcar apenas uma oval.

- Sim, com certeza seria um grande diferencial.
- Talvez, poderia atrair mais clientes.
- Não, acho que não faria diferença.
- Tenho dúvidas se meus clientes usariam.

10. Qual modelo de parceria você consideraria mais justo para usar um sistema completo como este?

Marcar apenas uma oval.

- Uma pequena taxa fixa por mês.
- Uma pequena porcentagem (%) sobre cada passagem vendida pelo sistema.
- Um modelo misto (taxa mensal baixa + porcentagem pequena).
- Não tenho certeza.

11. Gostaria de receber um convite para testar a plataforma ViaBus em primeira mão e sem custos quando estiver disponível?

Marcar apenas uma oval.

- Sim, tenho interesse!
- Não, obrigado.
- Talvez, gostaria de mais informações antes.

APÊNDICE B – RESULTADOS DA PESQUISA DE MERCADO

Este apêndice apresenta os resultados visuais coletados através da pesquisa de mercado realizada com dois gestores de empresas de transporte de passageiros.

Em qual empresa ou serviço você atua? (Opcional)

2 respostas

Expresso Macedo

Marcio e Regis Transportes e Serviços Ltda

Figura 25 – Resultado da pergunta sobre empresa ou serviço.

Qual o tamanho da frota de veículos?

2 respostas

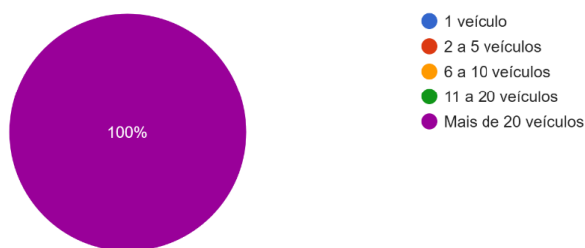


Figura 26 – Resultado da pergunta sobre tamanho da frota de veículos.

Quantas rotas (trechos) diferentes a empresa opera regularmente?

2 respostas

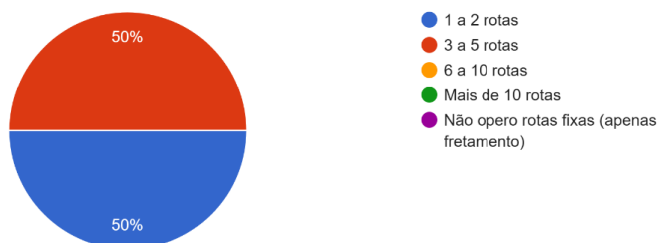


Figura 27 – Resultado da pergunta sobre número de rotas operadas.

A operação é principalmente:

2 respostas

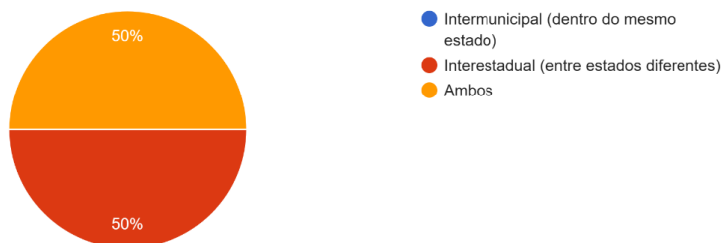


Figura 28 – Resultado da pergunta sobre tipo de operação.

Como é gerenciada a venda de passagens e a lista de passageiros hoje?

2 respostas

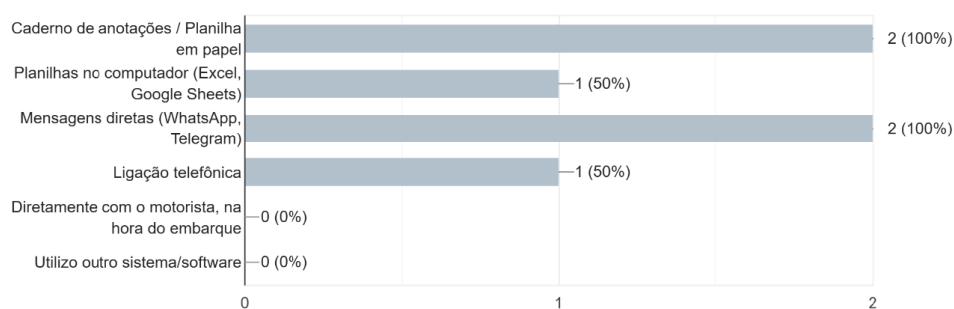


Figura 29 – Resultado da pergunta sobre gerenciamento atual de passagens.

Quais são os maiores desafios operacionais hoje para a empresa?

2 respostas

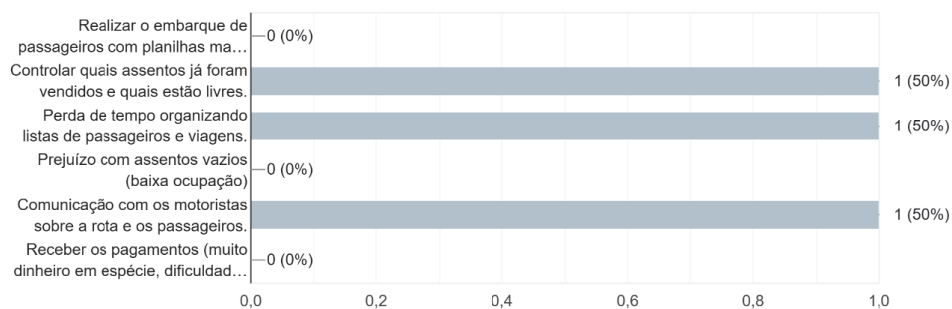


Figura 30 – Resultado da pergunta sobre maiores desafios operacionais.

Quão valioso seria ter um sistema único para gerenciar rotas, horários, veículos, motoristas e passagens em um só lugar, acessível pelo celular ...Marcar apenas uma oval, em uma escala de 1 a 5)
2 respostas

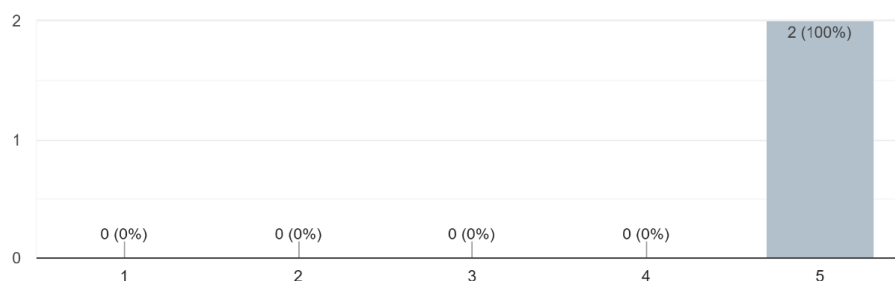


Figura 31 – Resultado da pergunta sobre valor de um sistema único.

Quais funcionalidades de um sistema como o ViaBus seriam mais importantes para o seu negócio?
2 respostas

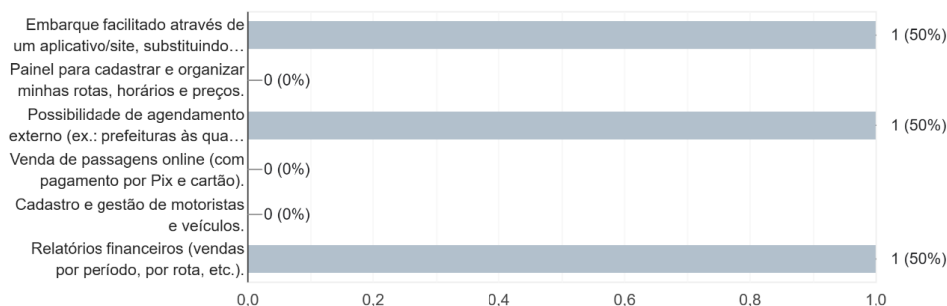


Figura 32 – Resultado da pergunta sobre funcionalidades mais importantes.

Se você pudesse oferecer aos seus passageiros um aplicativo para eles mesmos comprarem e acompanharem a viagem (ver horário, paradas, etc....)ra seus concorrentes? (Marcar apenas uma oval)
2 respostas

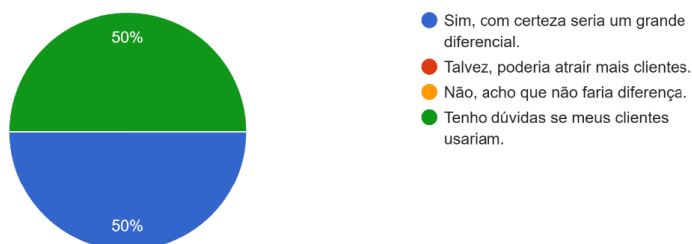


Figura 33 – Resultado da pergunta sobre aplicativo como diferencial.

Qual modelo de parceria você consideraria mais justo para usar um sistema completo como este?

2 respostas



Figura 34 – Resultado da pergunta sobre modelo de parceria.

Gostaria de receber um convite para testar a plataforma ViaBus em primeira mão e sem custos quando estiver disponível?

2 respostas

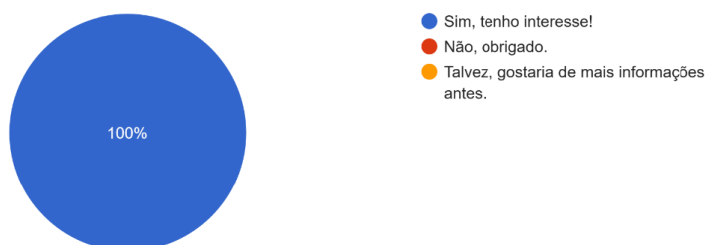


Figura 35 – Resultado da pergunta sobre interesse em testar a plataforma.

APÊNDICE C – DIAGRAMA DE CLASSE DO VIABUS

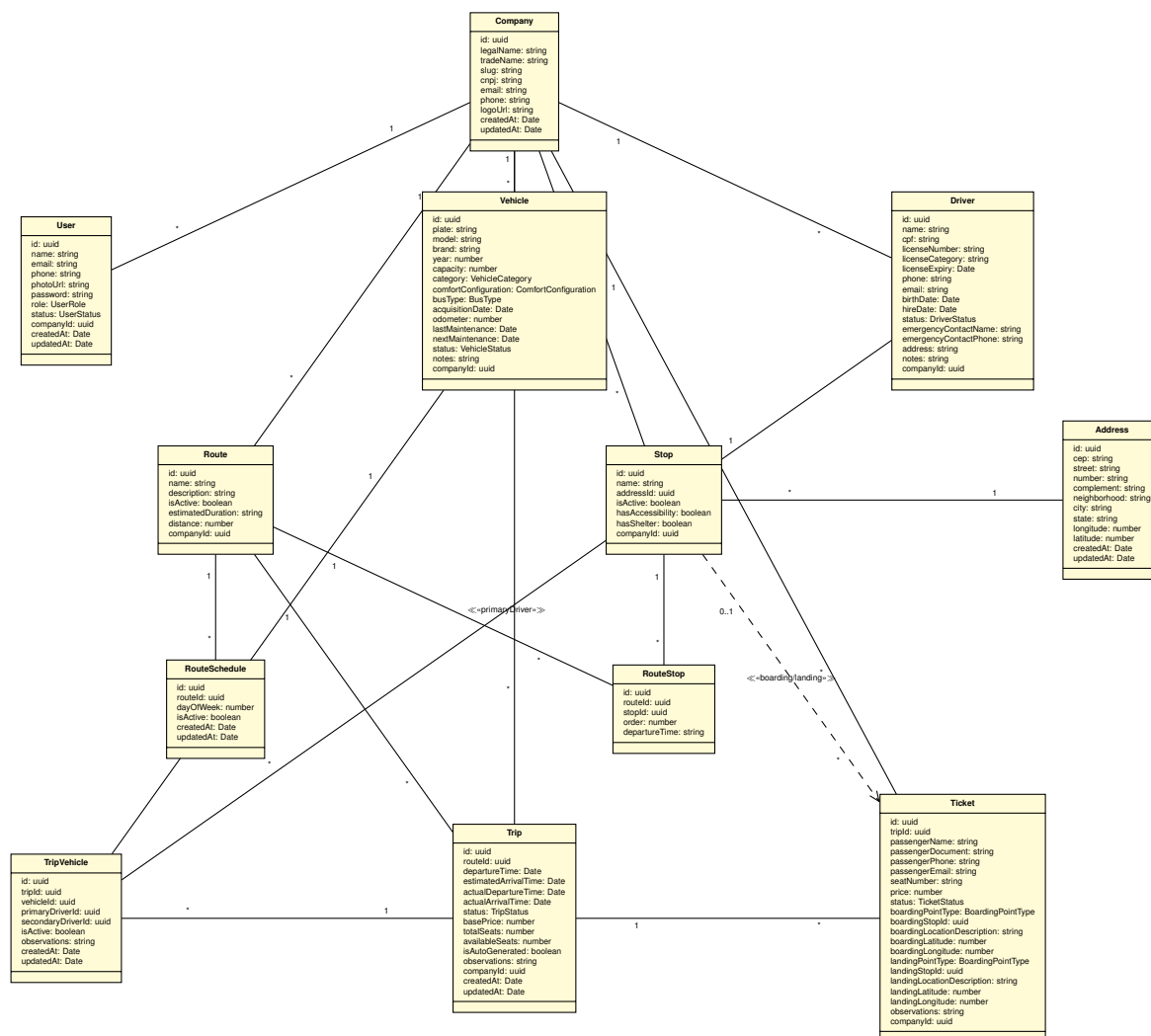


Figura 36 – Diagrama de classes detalhado do domínio ViaBus.

APÊNDICE D – REPOSITÓRIOS DO CÓDIGO-FONTE

O código-fonte completo do protótipo do sistema ViaBus, desenvolvido como parte deste trabalho, está publicamente disponível para consulta e análise nos seguintes repositórios da plataforma GitHub:

- **Repositório do Frontend (Interface do Usuário):**

<https://github.com/JeanCarlos899/viabus-front-end>

- **Repositório do Backend (API e Lógica de Negócio):**

<https://github.com/JeanCarlos899/viabus-back-end>