Centro Universitário Católica de Santa Catarina

Curso de Engenharia de Software

FLEETTRACK

Gabriel Knopka Ramos

Joinville, SC

2025

Resumo

Resumo: Este trabalho apresenta o desenvolvimento de um Sistema de Controle de Frotas, denominado FleetTrack, voltado para tornar a gestão de veículos mais simples e eficiente. O projeto busca oferecer uma solução prática e acessível para empresas que desejam organizar melhor suas operações, reduzir custos e ter mais controle sobre suas atividades diárias.

1 introdução

Contexto: A gestão de frotas é uma atividade fundamental para empresas que dependem do transporte de mercadorias ou prestação de serviços. O controle eficiente dos veículos e motoristas, bem como o acompanhamento das manutenções, são fatores essenciais para garantir segurança, reduzir custos operacionais e otimizar a logística.
Justificativa: A criação de uma ferramenta especializada para a gestão de veículos se mostra altamente relevante no campo da engenharia de software, pois oferece uma solução tecnológica capaz de automatizar processos, reduzir falhas humanas e melhorar a tomada de decisão. O sistema proposto busca atender a essa necessidade, alinhando-se às boas práticas de desenvolvimento e ao uso estratégico de tecnologias para resolver problemas reais.
Objetivos: O objetivo principal do projeto é criar uma ferramenta que ajude na gestão de veículos de forma prática, oferecendo recursos como o cadastro de veículos e motoristas, geração de relatórios de uso e desempenho, além do envio de alertas para situações que exigem atenção. Com isso, espera-se que o FleetTrack se torne um aliado importante no planejamento e na execução de atividades relacionadas à frota, impactando positivamente a logística e os resultados estratégicos das empresas.

2 descrição do projeto

O FleetTrack é pensado para empresas que precisam de uma gestão mais ágil e centralizada de suas frotas. A plataforma permitirá o registro de informações importantes sobre veículos e motoristas, a geração de relatórios de desempenho e o envio de alertas para situações críticas, tudo de maneira simples e acessível.

Muitas empresas enfrentam dificuldades como falta de dados consolidados, dificuldade de organizar as rotas, altos custos com operações ineficientes e consumo excessivo de combustível. Além disso, a ausência de um sistema dedicado pode comprometer a segurança e o planejamento. A proposta do FleetTrack é justamente enfrentar esses desafios, oferecendo uma solução que melhore o acompanhamento das operações, ajude a otimizar trajetos e promova um uso mais inteligente dos recursos.

Vale ressaltar que o projeto possui limitações: ele não incluirá funcionalidades de manutenção preditiva dos veículos nem integração com sensores mecânicos. O foco é atender exclusivamente a frotas terrestres, sem considerar operações aéreas ou marítimas. Assim, concentramos os esforços na eficiência operacional e na melhoria do suporte estratégico às empresas.

3 especificação técnica

Este projeto propõe a criação de um sistema de gestão de frotas que prioriza a organização e a eficiência, voltado para empresas que precisam de maior controle sobre seus veículos e operações.

Entre as principais funcionalidades estão: cadastro de veículos e motoristas, visualização de dados operacionais, geração de relatórios detalhados sobre a frota e alertas para eventos importantes. O objetivo é oferecer uma plataforma que seja prática, segura e que ajude as empresas a tomarem decisões mais assertivas.

3.1 requisitos de software

3.1.1 Requisitos funcionais

- **RF1 - Cadastro e Organização de Veículos**
  - RF1.1: O sistema deve permitir o cadastro de veículos, com informações
  - RF1.2: O sistema deve permitir o cadastro de informações adicionais sobre o estado de manutenção do veículo
  - RF1.3: O sistema deve permitir o cadastro de rotas.
  - RF1.4: Atribuir um destino ou rota específica a um veículo.
  - RF1.5: Permitir a atualização de informações de veículos cadastrados.
  - RF1.6: Permitir a exclusão ou desativação de veículos fora de operação.
- **RF2 - Cadastro e Organização de Motoristas**
  - RF02.1: O sistema deve permitir o cadastro de motoristas
  - RF2.2: Permitir a atualização de informações dos motoristas.
- **RF3 - Monitoramento e Relatórios Operacionais**
  - RF3.1: Ao atribuir uma viagem a um veículo, exibir uma previsão de gastos

- **RF4 - Painel Administrativo e Alertas**
  - RF4.1: Disponibilizar um painel com visão geral das operações
  - RF4.2: Implementar sistema de alertas automáticos para eventos crítico

O Diagrama de Casos de Uso(Figura 1) do sistema representa, de forma estruturada, as funcionalidades que podem ser acessadas pelo perfil Administrador no sistema, categorizadas por domínio funcional.

A seção Gestão de Usuários contempla ações administrativas relacionadas ao ciclo de vida de usuários do sistema, como criação, desativação, atualização de dados e gerenciamento de níveis de acesso. Na Gestão de Motoristas, o administrador pode realizar o cadastro, atualização e desativação de motoristas da frota.

A área Painel e Alertas fornece acesso ao painel administrativo e a alertas críticos, os quais são fundamentais para a tomada de decisão em tempo real. Por fim, a Gestão de Veículos permite operações completas sobre a frota, incluindo cadastro e atualização de veículos, desativação, atribuição de rotas, controle de manutenção, consulta de status e definição de destinos.

Este diagrama evidencia o papel central do administrador no sistema, tendo acesso completo às funcionalidades críticas que garantem o controle operacional e gerencial da frota.

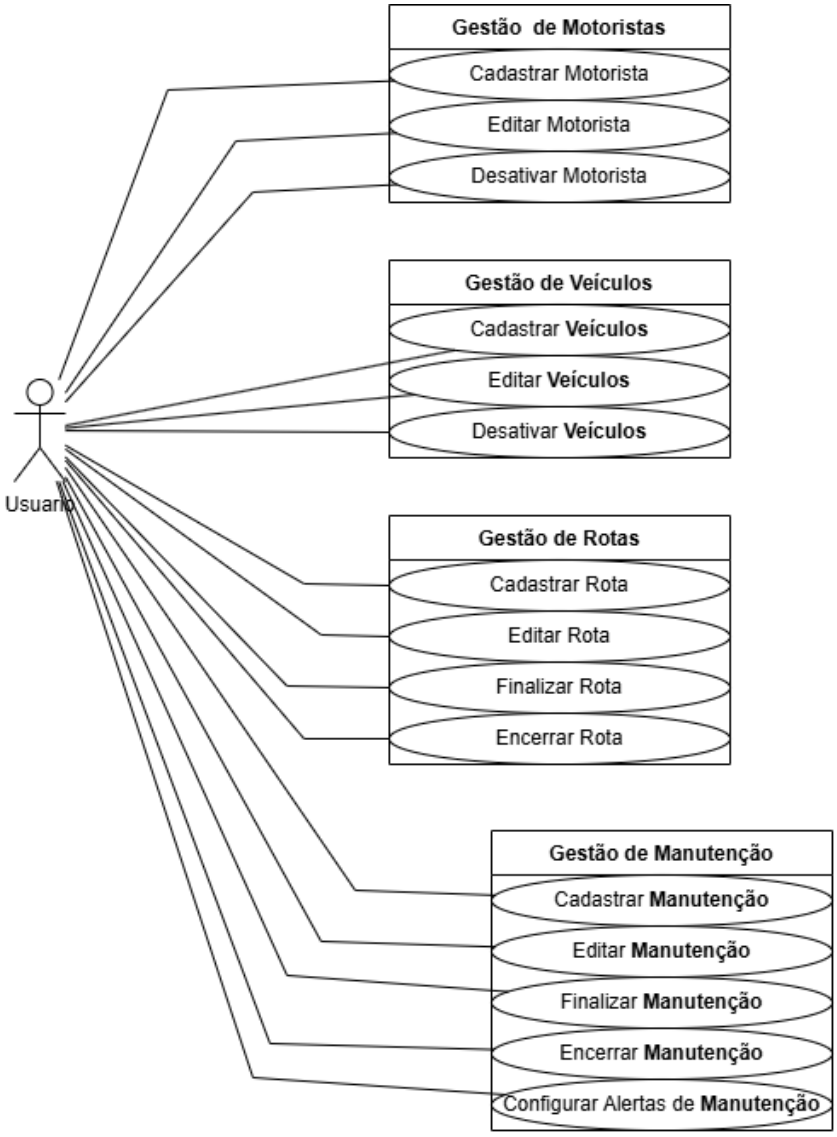

Figura 1: UML Administrador

3.1.2 Requisitos Não Funcionais

- RNF01: O sistema deve ser acessível via web.
- RNF02: A plataforma deve garantir respostas rápidas às solicitações dos usuários.
- RNF03: Os dados armazenados devem ser mantidos por no mínimo seis meses.
- RNF04: A interface do sistema deve ser intuitiva e amigável.

3.2 Considerações de design

3.2.1 Discussão sobre as escolhas de design

Durante o planejamento do FleetTrack, foram consideradas diversas alternativas de design para garantir uma solução escalável, segura e de fácil manutenção. A adoção de uma arquitetura baseada em três camadas (frontend, backend e banco de dados) foi preferida devido à sua simplicidade e eficiência em projetos.

Inicialmente, cogitou-se o uso de uma abordagem monolítica, pela facilidade de desenvolvimento e implantação, especialmente em projetos acadêmicos. Contudo, optou-se pela modularização através de microsserviços, visando maior escalabilidade e isolamento de falhas, facilitando a manutenção e futuras expansões.

O padrão MVC (Model-View-Controller) foi escolhido para o backend devido à sua ampla adoção, clareza na separação de responsabilidades e compatibilidade nativa com o framework Django.

3.2.2 Visão Inicial da Arquitetura

A arquitetura do FleetTrack será composta pelos seguintes componentes principais e suas respectivas interconexões:

- **Frontend:** Interface web desenvolvida com Django Templates, HTML, CSS e JavaScript, responsável pela interação com os usuários. Comunica-se com o backend via chamadas HTTP.

- **Backend:** API e lógica de negócios desenvolvida em Django, responsável pelo processamento de dados, regras de negócio e geração de relatórios.
- **Banco de Dados:** MySQL, destinado ao armazenamento persistente de informações relativas aos veículos, motoristas, trajetos e relatórios.
- **Serviço de Autenticação:** Implementação de autenticação e autorização baseada em Django Authentication, garantindo segurança na troca de informações.

A comunicação entre frontend e backend será realizada via APIs RESTful, utilizando o padrão HTTP, enquanto o backend se conecta ao banco de dados através do ORM (Object-Relational Mapping) do Django.

3.2.3 Padrões de Arquitetura

O projeto adota os seguintes padrões arquiteturais:

- **MVC (Model-View-Controller):** Para organizar a estrutura do backend, separando claramente as camadas de modelo de dados, lógica de negócio e apresentação.
- **RESTful APIs:** Para padronizar a comunicação entre frontend e backend, facilitando a integração com outros sistemas.

3.2.4 Modelos C4

Para uma melhor compreensão da arquitetura do FleetTrack, foi adotado o Modelo C4, que descreve o sistema em três níveis principais de abstração: Contexto, Contêineres e Componentes. Esses níveis possibilitam uma representação gradual da arquitetura, partindo de uma visão geral do sistema até o detalhamento das responsabilidades internas dos seus principais módulos

Contexto: Diagrama de Contexto(Figura 2), cuja finalidade é fornecer uma visão geral das interações do sistema com agentes externos. O FleetTrack é um sistema de gestão de frotas acessado por um único ator, o Administrador, que exerce funções operacionais e gerenciais através de uma interface web.

Além da interação humana, o sistema também realiza comunicação automatizada com o serviço externo OpenRouteService API, o qual fornece dados técnicos relacionados a rotas, como informações de distância, elevação e caminhos disponíveis. Essa integração tem como objetivo enriquecer funcionalidades internas, como o planejamento de itinerários e a geração de relatórios baseados em dados geográficos atualizados.

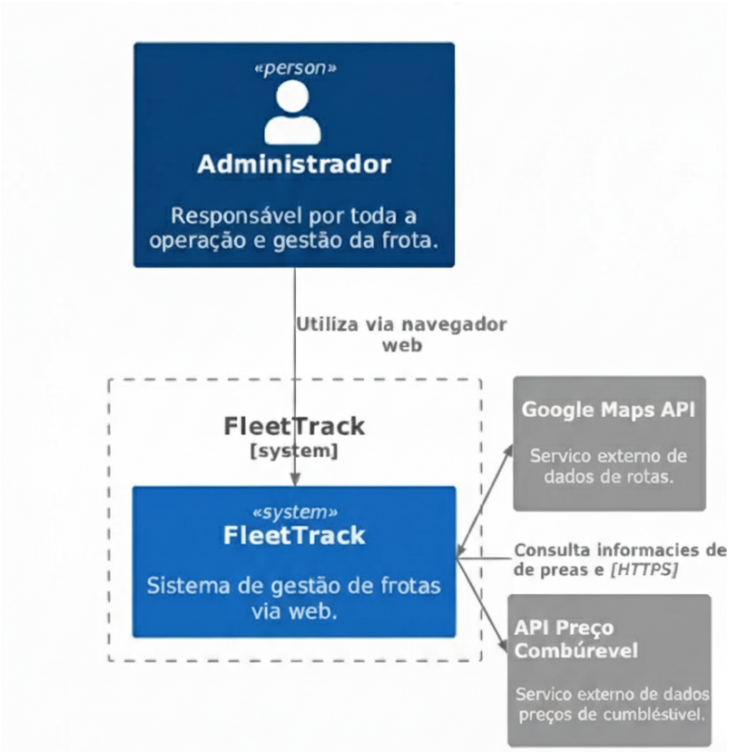

Figura 2: C4 Contexto

Contêineres: O sistema é dividido em três contêineres principais:

- **Frontend Web:** Aplicação baseada em Django Templates com HTML/CSS/JS. Responsável por renderizar páginas e interagir com o backend via requisições HTTP.
- **Backend API:** Serviço em Django, que expõe endpoints RESTful, processa as regras de negócio, autentica usuários via Django Authentication e interage com o banco de dados.
- **Banco de Dados Relacional:** MySQL, responsável pelo armazenamento persistente e seguro dos dados da frota.

Diagrama de Contêineres(Figura 3), evidenciando os principais blocos de software que o compõem e suas respectivas interações. O sistema é dividido em três contêineres principais: o Frontend Web, a Backend API e o Banco de Dados.

O Frontend Web, desenvolvido com Django Templates, representa a camada de apresentação acessada pelo usuário por meio de um navegador. Esse frontend se comunica com a Backend API, desenvolvida com Django e arquitetura RESTful, responsável por processar as regras de negócio, realizar autenticação de usuários e integrar serviços externos.

A persistência dos dados é realizada em um banco de dados relacional MySQL, que armazena informações como usuários, veículos, rotas e configurações do sistema. Complementarmente, a Backend API realiza requisições ao serviço externo OpenRouteService API para obtenção de dados técnicos sobre rotas, promovendo maior precisão e eficiência nos processos logísticos oferecidos pela aplicação.

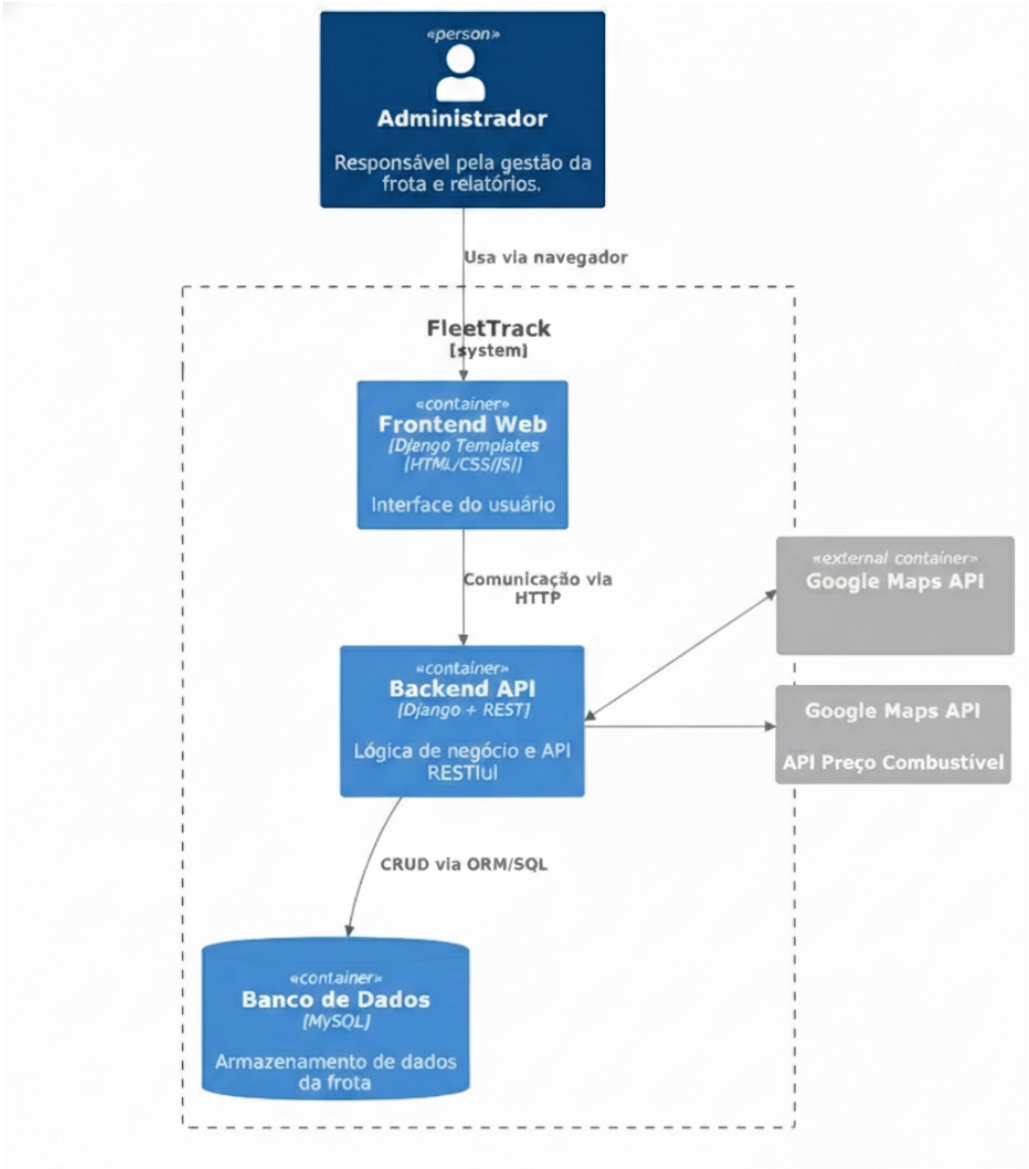

Figura 3: C4 Container

Componentes: O Diagrama de Componentes, representado na Figura 4, descreve a estrutura interna dos contêineres, detalhando como eles são compostos por módulos funcionais. No caso do FleetTrack, esse diagrama foca no backend da aplicação, desenvolvido em Django, e identifica seus principais

componentes lógicos.

Entre os componentes destacados estão o Serviço de Autenticação, o Módulo de Cadastro, que gerencia os dados de veículos; o Sistema de Geração de Relatórios, voltado à produção de informações gerenciais; e o Cliente de Integração Externa, que consome dados de rotas por meio de uma API pública. Esses componentes interagem entre si e com o banco de dados relacional, garantindo o funcionamento coeso do sistema.

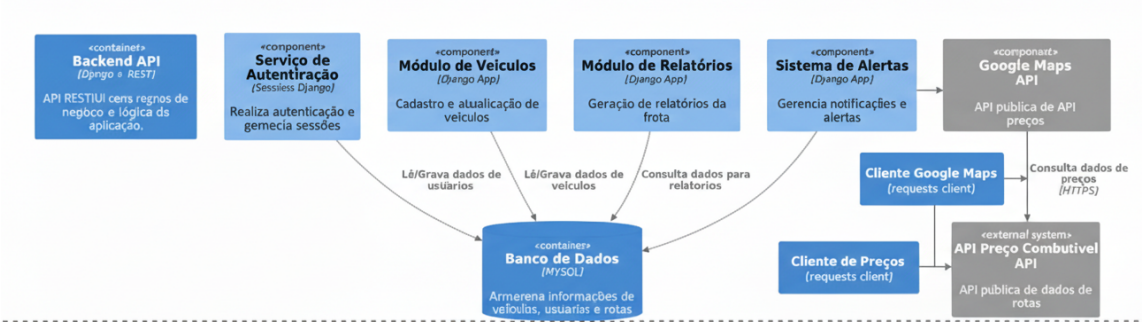

Figura 4: C4 Componentes

3.3 Stack Tecnológica

Linguagens de Programação:

- **Frontend:** Django Templates (HTML, CSS, JavaScript) para renderização dinâmica no lado do servidor.
- **Backend:** Django (Python) para desenvolvimento da aplicação.
- **Banco de Dados:** MySQL para armazenamento estruturado de informações.

Ferramentas de Desenvolvimento e Gestão de Projeto:

- **Git/GitHub**: para controle de versão e colaboração.
- **Postman**: para testes de APIs.
- **Google Cloud Platform (GCP):** Ambiente utilizado para o deploy e hospedagem da aplicação.

3.4 Considerações de Segurança

A segurança será um ponto central no desenvolvimento do sistema. Algumas medidas previstas:

- **Autenticação e autorização:** Controle de acesso utilizando Django Authentication.
- **Proteção contra vulnerabilidades:** Medidas para evitar ataques como SQL Injection, Cross-Site Scripting (XSS) e Cross-Site Request Forgery (CSRF).

4 Próximos Passos

Para garantir um desenvolvimento organizado e validado em todas as fases do projeto, os próximos passos estão estruturados conforme descrito a seguir:

4.1 Criação de mockups e fluxos de navegação

- Elaboração dos mockups das telas principais do sistema.
- Definição dos fluxos de navegação entre as páginas.

4.2 Desenvolvimento de protótipo

- Implementação de uma versão inicial com funcionalidades básicas.
- Testes preliminares com foco na navegação e fluxo de dados.
- Correção de eventuais falhas de usabilidade ou lógica.

4.3 Implementação dos módulos

- Desenvolvimento dos principais componentes do sistema:
  - Cadastro de motoristas e veículos;
  - Gerenciamento de viagens;
- Aplicação de boas práticas de segurança e desempenho.

4.4 Testes e validações

- Realização de testes unitários.
- Validação de funcionalidades conforme requisitos do projeto.
- Ajustes com base em feedback dos testes.

5 Referências

- **Django.** The web framework for perfectionists with deadlines. Disponível em: `https://www.djangoproject.com`. Acesso em: 14 fev. 2025.
- **MySQL.** MySQL: The world's most advanced open source relational database. Disponível em: `https://www.mysql.com`. Acesso em: 8 fev. 2025.
- **GitHub.** GitHub - Git repository hosting service. Disponível em: `https://github.com`. Acesso em: 29 fev. 2025.
- **Postman.** Postman - API platform for building and using APIs. Disponível em: `https://www.postman.com`. Acesso em: 20 fev. 2025.