UNIVERSIDADE ESTADUAL DE GOIÁS (UEG)

INSTITUTO ACADÊMICO DE CIÊNCIAS E TECNOLOGIA

UNIDADE DE PIRES DO RIO

CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS PARA INTERNET

Documento de Requisitos do Software

Sistema de Controle de Funcionários

**SUMÁRIO**

# 

[**INTRODUÇÃO 3**](#_x07ephjuvhu2)

[**TECNOLOGIAS 4**](#_plxvwwx677hy)

[Backend: 4](#_91692nlotduz)

[Frontend: 4](#_gb4pjabtpw53)

[**REQUISITOS FUNCIONAIS 6**](#_lhcz9u872vtc)

[Gestão de Funcionários: 6](#_iv4algsyo6qw)

[[RF.001] - Cadastro de Funcionários: 6](#_9b6vw64ru5mf)

[[RF.002] - Edição de Funcionários: 6](#_ic6a9t407y8b)

[[RF.003] - Visualização de Funcionários: 6](#_w1u7q5ofcubm)

[Gestão dos Caminhões: 6](#_y0kgh64li4p3)

[[RF.004] - Cadastro de Caminhões: 6](#_de0xu2uestbh)

[[RF.005] - Edição de Caminhões: 6](#_ud653dd5dblx)

[[RF.006] - Visualização de Caminhões: 6](#_27vp0qqql3ea)

[Escala de Funcionários: 7](#_3mmkla9nw70u)

[[RF.007] - Designação de Funcionários: 7](#_k8r1oedbf9yo)

[[RF.008] - Visualização da Escala: 7](#_h12hx48rzjwl)

[**REQUISITOS NÃO FUNCIONAIS 8**](#_dfkm18edvoex)

[Desempenho 8](#_r48jsiw90cq3)

[[NF.001] - Tempo de Resposta 8](#_fs836ri8xhlh)

[[NF.002] - Suporte a Usuários Simultâneos 8](#_n4egmmajelhq)

[Usabilidade 8](#_gef04mkde364)

[[NF.003] - Design Responsivo 8](#_zgdrpmaz94ux)

[[NF.004] - Interface Intuitiva 8](#_e1o6ti8m08i6)

[Autenticação e autorização 8](#_3829a72lnvok)

[[NF.005] - Login de Usuários 8](#_i0i2oyo08uwk)

[[NF.006] - Controle de Acesso 8](#_iv7uy05s5qxy)

[Compatibilidade 9](#_9y8tlhratvp1)

[[NF.007] Desktop 9](#_qhnuxxgj7inj)

[[NF.008] Mobile 9](#_wpfdhzjk5ifz)

[**CONCLUSÃO 10**](#_awaaov70xewd)

# 

Capítulo 1

# **INTRODUÇÃO**

Este documento descreve como os requisitos funcionais e não funcionais foram atendidos durante o desenvolvimento da aplicação de gestão de escalas de trabalho e frota da transportadora.

Capítulo 2

# **TECNOLOGIAS**

## **Backend:**

* **Spring Boot:** Framework utilizado para o desenvolvimento da API, que facilita a criação de microserviços e aplicações Java baseadas na arquitetura RESTful.
* **Spring Security:** Usado para implementar o controle de autenticação e autorização dos usuários, garantindo que apenas usuários autenticados e com permissões adequadas possam acessar recursos específicos.
* **JWT (JSON Web Token):** Usado para autenticação segura, permitindo a comunicação entre o frontend e o backend com segurança.
* **MySQL:** Banco de dados relacional utilizado para armazenar informações de funcionários, caminhões e escalas.
* **JPA (Java Persistence API):** Utilizado para a manipulação do banco de dados de forma orientada a objetos, com o auxílio do **Hibernate** para implementação de consultas e mapeamento de entidades.
* **Maven:** Ferramenta de gerenciamento de dependências e automação de builds, usada para gerenciar as dependências do Spring Boot e outras bibliotecas.

## **Frontend:**

* **React.js:** Biblioteca JavaScript utilizada para construir a interface do usuário de forma declarativa e baseada em componentes.
* **Material-UI:** Biblioteca de componentes React que implementa o design Material, proporcionando uma interface limpa, responsiva e fácil de usar.
* **Axios:** Biblioteca para realizar requisições HTTP, utilizada no frontend para interagir com a API e enviar/receber dados.
* **React Router:** Utilizado para implementar o roteamento na aplicação frontend, permitindo navegação entre as páginas de maneira dinâmica.
* **React Context API:** Usada para gerenciar o estado global da aplicação, especialmente para controle de autenticação e sessões de usuário.

Capítulo 3

# **REQUISITOS FUNCIONAIS**

## Gestão de Funcionários:

### [RF.001] - Cadastro de Funcionários:

Foi implementado um endpoint “POST /funcionarios” na API, permitindo que a administração cadastre novos funcionários. A interface frontend em React foi desenvolvida com formulários para capturar os dados dos funcionários e enviá-los para a API

### [RF.002] - Edição de Funcionários:

Foi implementado um endpoint “PUT /funcionarios/{id}” na API para atualização dos dados dos funcionários. O frontend permite que o administrador edite os dados de um funcionário, exibindo os dados atuais em um formulário que pode ser alterado e enviado para a API.

### [RF.003] - Visualização de Funcionários:

Foi criado o endpoint “GET /funcionarios” para retornar a lista de todos os funcionários cadastrados. No frontend, foi criada uma tabela que exibe as informações de todos os funcionários, permitindo à administração visualizar os dados.

## Gestão dos Caminhões:

### [RF.004] - Cadastro de Caminhões:

Implementação do endpoint “POST /caminhoes” para o cadastro de novos caminhões. A interface frontend contém um formulário onde a administração pode inserir os dados do caminhão.

### [RF.005] - Edição de Caminhões:

A API possui o endpoint “PUT /caminhoes/{id}” para editar os dados de caminhões. O frontend exibe os dados dos caminhões em um formulário que pode ser alterado e enviado para a API.

### [RF.006] - Visualização de Caminhões:

O endpoint “GET /caminhoes” foi criado para retornar todos os caminhões cadastrados. No frontend, a administração pode visualizar a lista completa de caminhões em uma tabela.

## Escala de Funcionários:

### [RF.007] - Designação de Funcionários:

Foi criado um endpoint “POST /escalas” para designar os funcionários para suas escalas de trabalho. O frontend permite que a administração selecione funcionários e caminhões e os associe a uma escala de trabalho.

### [RF.008] - Visualização da Escala:

Implementação do endpoint “GET /escalas” para que tanto a administração quanto os funcionários possam visualizar suas escalas. O frontend permite que a administração visualize todas as escalas e que os funcionários visualizem suas respectivas escalas.

Capítulo 4

## 

# REQUISITOS NÃO FUNCIONAIS

## 

## 

## Desempenho

### [NF.001] - Tempo de Resposta

Foi utilizado o React para garantir tempos de resposta rápidos, utilizando técnicas de cache.

### [NF.002] - Suporte a Usuários Simultâneos

Utilização recursos do Spring Boot e MySQL que garantem o desempenho em cenários de alta carga.

## Usabilidade

### [NF.003] - Design Responsivo

O frontend foi desenvolvido utilizando Material-UI, que garante uma interface responsiva para dispositivos móveis e desktop. A aplicação se adapta dinamicamente ao tamanho da tela do usuário.

### [NF.004] - Interface Intuitiva

A interface foi projetada para ser simples e fácil de usar. Os formulários de cadastro e edição foram implementados com validações claras, e a navegação entre as páginas foi organizada para tornar o sistema intuitivo para administradores e funcionários.

## Autenticação e [autorização](https://docs.google.com/document/d/1LUvHlRWAkTNh6GdL8SLkMN-ZcBG8J3c9WcPcMXFYLuc/edit#heading=h.12nu8pjydpir)

### [NF.005] - Login de Usuários

Foi implementado o sistema de autenticação via JWT na API com endpoints “POST /login” para que os usuários possam realizar login. O frontend gerencia a autenticação, armazenando o token JWT no localStorage e utilizando-o nas requisições subsequentes.

### [NF.006] - Controle de Acesso

A API implementa middleware para validar o token JWT e verificar o papel (admin ou user) do usuário. O frontend exibe as telas de acordo com o papel do usuário, restringindo funcionalidades como a edição de dados aos administradores.

## Compatibilidade

### [NF.007] Desktop

O frontend foi testado em vários navegadores, incluindo Chrome, Firefox, e Edge, garantindo a compatibilidade e funcionalidade da aplicação em diferentes plataformas.

### [NF.008] Mobile

O design responsivo implementado com Material-UI assegura que a aplicação seja utilizável em diversos dispositivos

Capítulo 5

# CONCLUSÃO

Os requisitos foram atendidos de acordo com as especificações fornecidas, utilizando tecnologias modernas como Spring Boot, React.js, JWT, Material-UI, MySQL, Axios e Maven. A aplicação é segura, escalável, e oferece uma interface intuitiva, atendendo às necessidades da Aliás Transportadora para gerenciar escalas de trabalho e frota.

Se houver necessidade de ajustes adicionais ou novas funcionalidades, o sistema foi projetado de forma modular, permitindo fácil manutenção e expansão.