

---

# **Software Requirements Specification**

**for**

## **ReTruck**

**Versão 2.1 aprovada**

**8180378-Hugo Silva**

**Escola Superior de Tecnologia e Gestão**

**29/08/2022**

# Índice

<b>1. Introdução .....</b>	<b>5</b>
1.1 Propósito .....	5
1.2 Sugestões de Audiência e Leitura Previstas .....	5
1.3 Âmbito do projeto .....	5
1.4 Referências .....	5
<b>2. Descrição Global.....</b>	<b>6</b>
2.1 Perspetiva de Produto .....	6
2.2 Funcionalidades do Produto .....	6
2.3 Classes e Características dos Utilizadores .....	7
2.4 Ambiente Operacional .....	8
2.5 Pressupostos e dependências.....	8
<b>3. Requisitos de Interface Externa.....</b>	<b>9</b>
3.1 Interface de Utilizador .....	9
3.2 <i>Software</i> Interfaces .....	18
3.3 Interfaces de Comunicação .....	18
<b>4. Use Cases.....</b>	<b>19</b>
4.1 Gestão de Utilizadores e Organizações .....	19
4.2 Gestão de Ausências.....	20
4.3 Gestão de Camiões .....	21
4.4 Gestão de Avarias .....	22
4.5 Transportes .....	23
4.6 Análise de Transportes e Gestão de Serviços .....	24
4.7 <i>Dashboard</i> .....	25
4.8 <i>Diagramas Use Cases</i> .....	26
<b>5. Requisitos Não Funcionais .....</b>	<b>29</b>
<b>6. Anotações .....</b>	<b>30</b>
6.1 <i>Super Admin</i> .....	30
6.2 Algoritmo de análise e seleção .....	30
6.3 Algoritmo de análise e seleção de camiões, pensa duas vezes.....	31
<b>7. Arquitetura do Projeto .....</b>	<b>32</b>
7.1 Notas sobre a arquitetura .....	33
<b>8. Diagrama de Classes .....</b>	<b>34</b>
<b>9. Notas de Anexo .....</b>	<b>36</b>

## Histórico de Versões

Nome	Data	Motivos das Alterações	Versão
Hugo	03-08-2022	<i>Skeleton</i> do relatório	1.0
Hugo	04-08-2022	Adição de novos Requisitos	1.1
Hugo	05-08-2022	Adição de novos Requisitos e ambiente operacional	1.2
Hugo	05-08-2022	Perspetiva do Produto	1.3
Hugo	06-08-2022	Algoritmo de análise e seleção	1.4
Hugo	07-08-2022	XML e login <i>mobile</i>	1.5
Hugo	09-08-2022	<i>Mockups mobile</i>	1.6
Hugo	10-08-2022	<i>Mockups web</i>	1.7
Hugo	11-08-2022	Fim da descrição dos <i>mockups</i>	1.7
Hugo	12-08-2022	Diagrama de Classes	1.8
Hugo	13-08-2022	Alteração Diagrama de Classes	1.9
Hugo	20-08-2022	Adição dos novos serviços externos	2.0
Hugo	29-08-2022	Dados de configuração do modelo de análise	2.1

## Abreviaturas e Glossário

1. *Back-end*- componente de uma arquitetura de *software* que implementa as regras de negócio;
2. *Front-end*- Componente de uma arquitetura de *software* que fornece a *interface* aos utilizadores;
3. *RNF*- Requisito não funcional.
4. DTO– *Data Transfer Object*;

# 1. Introdução

## 1.1 Propósito

Este documento foi elaborado para o projeto ReTruck e contém a descrição de um sistema de *software* a ser desenvolvido para a disciplina de LDS. Entre o conteúdo que poderá ser encontrado no documento consta o propósito do *software*, o seu domínio, os requisitos, os *use cases*, os *mockups* e o diagrama de classes.

O projeto ReTruck nasceu para apoiar a tomada de decisão de uma empresa de caminhões que trabalha com 4 tipos de transportes diferentes. O objetivo é produzir um sistema que automatize a análise de transportes e o escalonamento dos caminhões.

## 1.2 Sugestões de Audiência e Leitura Previstas

Este documento destina-se ao aluno que irá desenvolver o projeto seguindo as práticas da metodologia SCRUM.

Para uma melhor percepção do projeto é aconselhado a leitura de artigos sobre a indústria 4.0 na área da logística. Em resumo várias áreas da indústria 4.0 atualmente possuem sensores que fornecem informação importante para a tomada de decisão. Neste contexto, os sensores serão simulados através de um ficheiro XML e irão fornecer dados cujo sistema ReTruck irá processar de modo a determinar se um transporte é ou não rentável/possível para a empresa, bem como o melhor caminhão/camhões para o realizar.

## 1.3 Âmbito do projeto

Existem empresas cujo a sua atividade é a realização de transportes de mercadorias para entidades externas. Um exemplo dessas empresas é a InterBrigas<sup>1</sup> que recebe pedidos de transporte com uma data, uma origem um destino e um valor. O processo de análise que determina se o transporte é ou não aceite envolve cálculos que são feitos de forma manual pelo Gestor. Dado essa situação e a falta de sistemas que apoiem este tipo de decisão o aluno resolveu criar uma plataforma que permita automatizar este processo.

## 1.4 Referências

*IEEE. Standard for Software Configuration Management Plans*

---

<sup>1</sup> <http://www.interbrigas.com/>

## 2. Descrição Global

### 2.1 Perspetiva de Produto

O projeto consiste num sistema de apoio à decisão para empresas de transportes que trabalham com serviços que lhes são propostos pelos clientes. Nesse sentido o sistema irá permitir que os clientes registem os seus transportes que serão depois analisados de uma forma automatizada. A análise que será feita irá ter em conta variáveis como por exemplo:

1. Consumo do camião.
2. Capacidade do camião.
3. Tipo do camião.
4. Número de Kms a percorrer.
5. Tipo do transporte.
6. Disponibilidade do camião no dia do serviço.
7. Valor ganho pelo serviço.

Estas variáveis serão processadas através de algoritmos que irão determinar se o serviço é rentável, possível e qual é o melhor camião para o fazer. No processo de análise existirá também um conjunto de dados que o utilizador pode parametrizar de acordo com as suas preferências/necessidades.

Além dessa análise o sistema irá também permitir que os condutores marquem as suas ausências de modo que isso seja tido em conta na seleção de um camião para um determinado transporte numa determinada data.

### 2.2 Funcionalidades do Produto

Com base nos diferentes tipos de perfil de utilizador é pretendido que no final o produto contenha as seguintes funcionalidades:

#### **Super-Admin**

1. Gestão de Utilizadores (CRUD);
2. Envio de *email* com os dados de acesso para os utilizadores;
3. Gestão de organizações;

#### **Manager**

1. Criação de um camião;
2. Edição de um camião;
3. Visualização de todos os camiões da organização;

4. Aceitar/Recusar férias pedidas pelos funcionários;
5. Visualizar todas as ausências<sup>2</sup> da empresa;
6. Efetuar Gestão de avarias dos camiões;
7. Visualização de informação relativa à atividade da empresa.
8. Aceitar/Recusar transportes;
9. Proceder à análise de transportes;
10. Aceitar / Recusar transportes;
11. Atribuir camiões a um transporte;

#### **Client**

1. Criar transportes;
2. Visualizar os seus transportes;

#### **Driver**

1. Pedir/Registar ausências;
2. Iniciar o seu serviço do dia;
3. Indicar a sua localização atual;
4. Terminar o seu serviço;

## **2.3 Classes e Características dos Utilizadores**

Existiram 4 tipo de utilizadores, no entanto o produto será desenvolvido em grande parte a pensar no papel do *Manager* de uma empresa de camiões.

1. *Manager*: Responsável pela frota, tem de decidir quais serviços aceita/rejeita e quais os camiões que os irão realizar.
2. *Client*: Efetua pedidos de transporte.
3. *Driver*: Regista ausências e inicia/termina serviços.
4. *Super-Admin*: Num contexto empresarial apenas um perfil de utilizador tem permissões de adicionar novos utilizadores ao sistema. O *Super-Admin* será o único ator responsável por essa tarefa no ReTruck.

---

<sup>2</sup> Férias é um tipo de ausência, existe outros como por exemplo faltas por motivos de doença ou motivos familiares

## 2.4 Ambiente Operacional

Este projeto irá correr em duas plataformas. A plataforma *WEB* que será utilizada num *browser* que corre num sistema operativo (por exemplo, Windows ou Linux) e será utilizada pelo *Manager* e pelo *Super-Admin*. Já a plataforma *MOBILE* irá correr num telemóvel *android* conectada com uma base de dados *Firebase* e será usada pelo *Driver* e pelo *Client*.

### Servidor/*Back-end*:

- API: .NET *framework* 6.0, linguagem C#
- Base De Dados: SQL Server
- Base De Dados: *Firebase*

### Client/*Front-end*:

- Website: React, linguagem JavaScript
- Mobile App: *Android*, linguagem Java

## 2.5 Pressupostos e dependências

A implementação do projeto apresenta as seguintes dependências:

- **Google API**
  - Distance Matrix API – Usada para calcular distância entre os locais.
  - Maps JavaScript API – Usada para imprimir o mapa na aplicação React.
  - SDK do Maps- Usada para simular a localização de um camião que será dada a partir da localização do telemóvel.
- **Firebase-** Base de dados que irá armazenar os dados da aplicação *android* que serão depois recolhidos pela API do *back-end*.
- **XML** – Ficheiro que contém informações sobre os camiões. Servirá para simular os dados recolhidos por um sensor embebido num camião. Esses dados na indústria 4.0 seriam recolhidos pela plataforma IOT<sup>3</sup>.
- **Holiday API**<sup>4</sup>– Usada para verificar se uma data corresponde a um feriado.
- **PositionStack API**<sup>5</sup>- Usada para converter uma morada em coordenadas geográficas.

---

<sup>3</sup> <https://www.rangel.com/pt/blog/industria-40-impacto-cadeias-de-abastecimento/>

<sup>4</sup> <https://holidayapi.com/>

<sup>5</sup> [https://positionstack.com/?utm\\_source=FirstPromoter&utm\\_medium=Affiliate&fpr=victor96](https://positionstack.com/?utm_source=FirstPromoter&utm_medium=Affiliate&fpr=victor96)



### 3. Requisitos de Interface Externa

#### 3.1 Interface de Utilizador

Dado as funcionalidades descritas no capítulo 2 deste documento, foi identificado que serão necessárias interfaces de utilizador em dois tipos de dispositivos diferentes, na aplicação *mobile* e na aplicação *web*.

Para a aplicação *mobile*, do ponto de vista do *Driver* tem-se 4 ecrãs, figura 1. O ecrã de *login*, a *homepage*, o registo de ausência e a interface para iniciar/terminar um serviço (que deverá corresponder ao serviço do dia). O primeiro ecrã mostrado deve ser na *app mobile* é o *login*.



Figura 1 Mockups mobile para o Driver

Do ponto de vista do *client* tem-se 4 ecrãs, figura 2. O ecrã de *login*, a *homepage*, o registo de transporte e a interface para visualizar o seu histórico de transportes.



Figura 2 Mockups mobile para o Client

### 3.1.1 Login Web

Na aplicação *web* a primeira interface que será apresentada ao *user* é a interface de *login*, cujo o *mockup* se encontra na figura 3. Formulário simples com dois *inputs* (*username* e *password*).

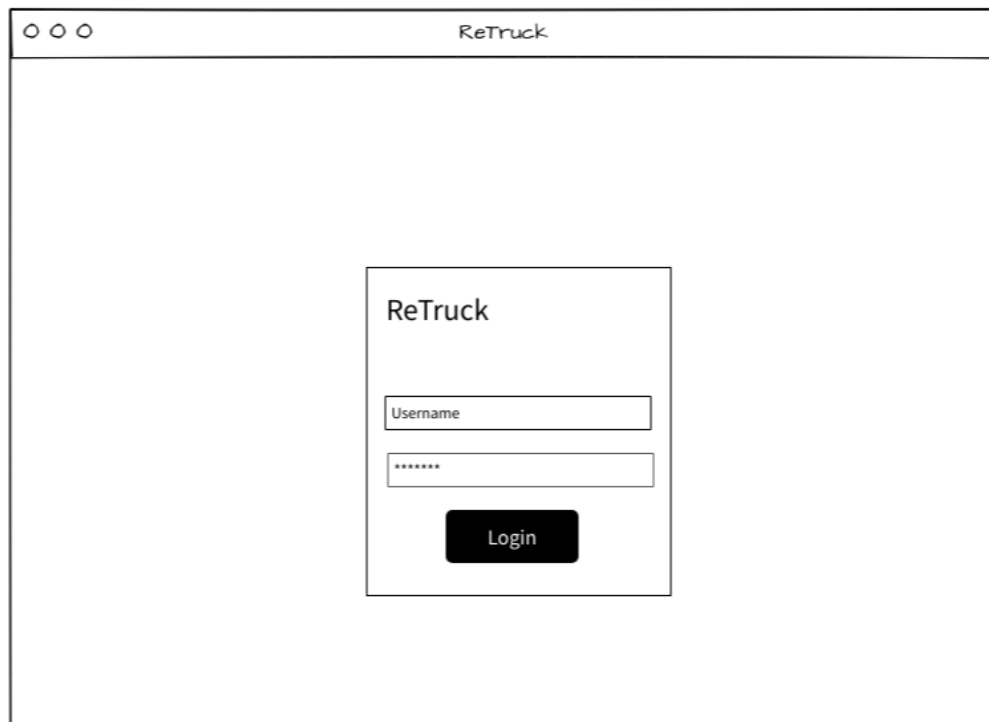


Figura 3 *Login* aplicação web

### 3.1.2 Homepage

Alguns gráficos com informação sobre a atividade da empresa, figura 4.

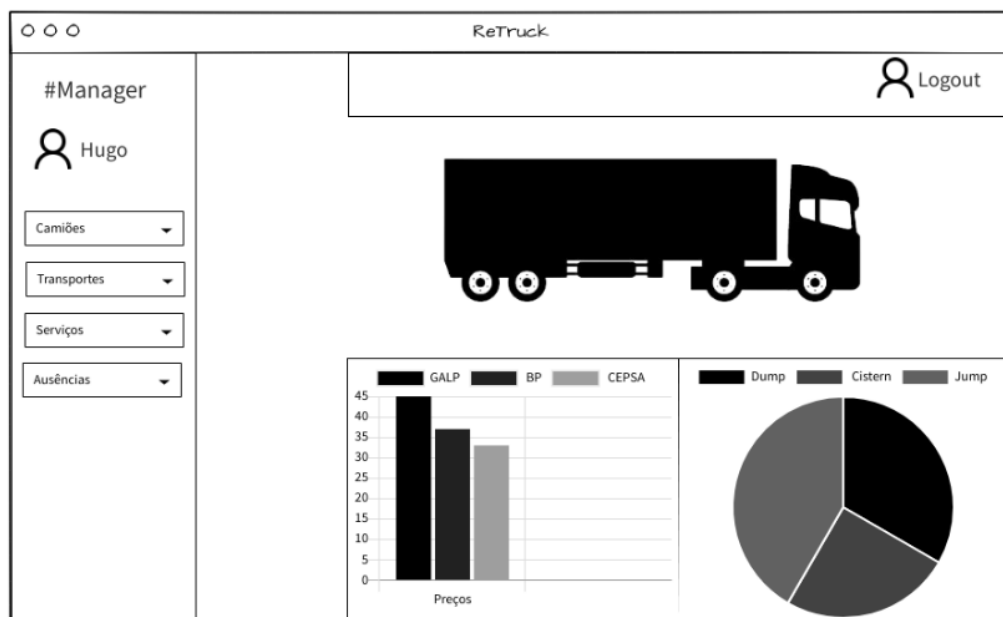


Figura 4 *Homepage*

### 3.1.3 Menus *Manager* e *Super Admin*

Os menus devem oferecer as funcionalidades por perfil de utilizador. Na figura 5 apresentam-se os dois tipos de menus necessários.



Figura 5 Menu de utilizadores

### 3.1.4 Criar organizações

Formulário com 3 *inputs* de texto e um botão, figura 6.

The image shows a mobile application screen for creating a new organization. The screen has a header bar with the title 'ReTruck' and a 'Logout' button with a user icon. On the left, there is a sidebar menu for '#Super Admin' user 'Hugo' with two visible items: 'Utilizadores' and 'Organizações'. The main content area is titled 'Criar Organização' and contains three text input fields labeled 'Nome', 'Localização', and 'Vatin'. Below these fields is a dark button labeled 'Criar Organização'.

Figura 6 Criar Organizações

### 3.1.5 Criar utilizadores

Formulário com 4 *inputs* de texto, um *select* com as opções que existem para perfil de utilizador e outro para as organizações. Por fim colocar um botão para o *upload* da foto do utilizador (campo opcional). Figura 7.

The screenshot shows a web application interface for 'ReTruck'. On the left is a sidebar for a '#Super Admin' user named 'Hugo', with navigation links for 'Utilizadores' and 'Organizações'. The main content area is titled 'Criar Utilizador' and contains the following fields: 'Username', 'Email', 'Nome', 'Password', 'Organização' (a dropdown menu), and 'Perfil de Utilizador' (a dropdown menu). Below these is a file upload section with a 'Foto' button and the text 'No File Chosen'. At the bottom right is a black button labeled 'Criar Utilizador'.

Figura 7 Formulário de criar utilizador

### 3.1.6 Criar Revisão

Formulário com 2 *inputs* de texto, um *select* com as opções todos os camiões que existem na organização e um *input* do tipo *date* para o utilizador selecionar a data em que foi/será feita a revisão. Figura 8.

The mockup shows a web application interface for 'ReTruck'. On the left is a sidebar for a user named '#Manager Hugo' with a profile icon and four dropdown menus: 'Camiões', 'Transportes', 'Serviços', and 'Ausências'. The top right corner has a 'Logout' button with a user icon. The main content area is titled 'Adicionar Revisão de um Camião'. It contains four input fields: 'Descrição' (text), 'Preço' (text), 'Camião' (dropdown), and a date field showing '12 May 2016' with a calendar icon. At the bottom is a black button labeled 'Criar Revisão'.

**Figura 8 Mockup Criar Revisão**

### 3.1.7 Criar camião

Formulário com 1 *input* de texto para a matrícula, um *select* para listar as categorias dos camiões e outro para listar as organizações. Por fim colocar um botão para o *upload* da foto do camião (campo opcional). Figura 9.

The mockup shows a web application interface for 'ReTruck'. On the left is a sidebar for a user named '#Manager Hugo' with a profile icon and four dropdown menus: 'Camiões', 'Transportes', 'Serviços', and 'Ausências'. The top right corner has a 'Logout' button with a user icon. The main content area is titled 'Criar Camião'. It contains three input fields: 'Matricula' (text), 'Organização' (dropdown), and 'Categoria' (dropdown). Below these is a file upload section with a 'Foto' button and the text 'No File Chosen'. At the bottom is a black button labeled 'Criar Camião'.

**Figura 9 Mockup criar camião**

### 3.1.8 Listagem de todos os camiões

Na listagem de camiões deve existir 3 colunas com um valor de 4 numa *row* de 12, cada coluna deve possuir uma foto do camião e a sua matrícula. A matrícula deve possuir um *link* para abrir a página de detalhes do camião. Fazer rota dinâmica pelo *id*.

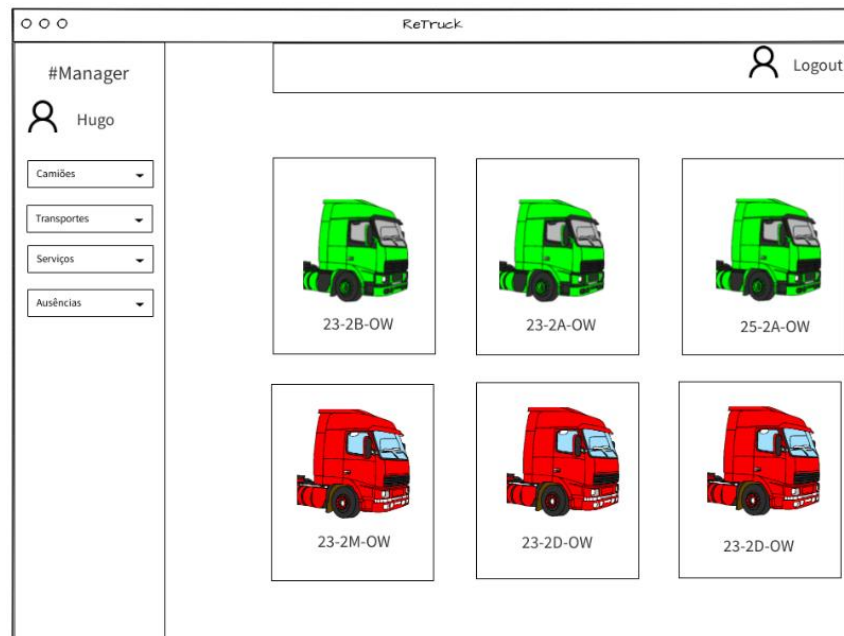


Figura 10 *Mockup* da Listagem de Camiões

### 3.1.9 Página de detalhes de um camião

Apresentação dos detalhes de um camião, figura 11.

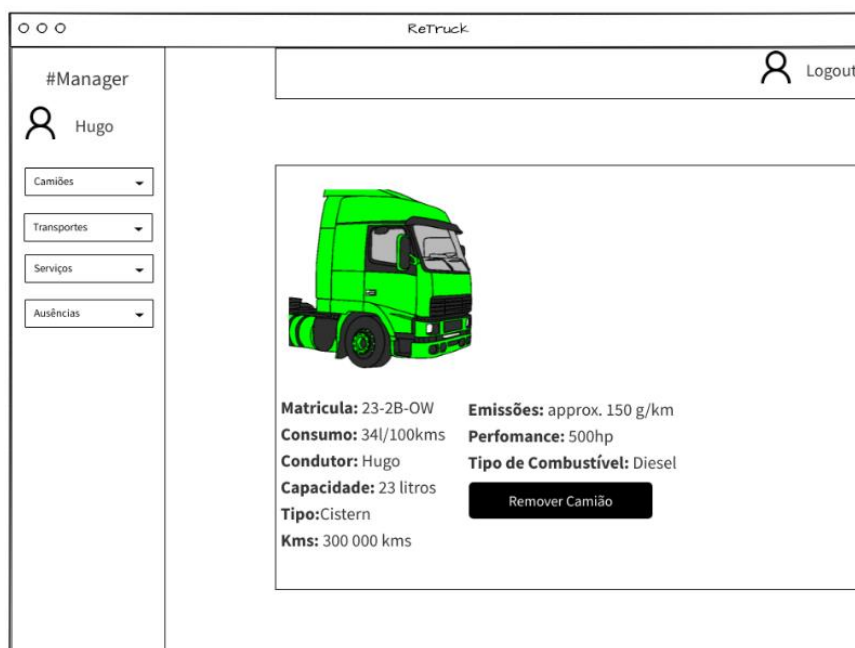


Figura 11 *Mockup* da Listagem de um camião

### 3.1.10 Listagem de avarias

Tabela com 4 colunas listando os dados relevantes das avarias. Devem também existir 2 colunas com botões de ações sobre as avarias (apagar/editar). Figura 12.

The mockup shows a web application interface for 'ReTruck'. On the left is a sidebar for a user named 'Hugo' with a '#Manager' role. It contains four dropdown menus: 'Camiónes', 'Transportes', 'Serviços', and 'Ausências'. The main content area has a 'Logout' button in the top right. Below it is a table titled 'Manutenções' with 6 columns: 'Data', 'Descrição', 'Custo', 'Matricula', 'Editar', and 'Delete'. The table contains three rows of data, each with a green 'Editar' button and a red 'Apagar' button.

Data	Descrição	Custo	Matricula	Editar	Delete
02/02/2022	Pneus	1000€	72-23-OF	Editar	Apagar
02/04/2022	Pneus	1000€	72-23-OF	Editar	Apagar
02/06/2022	Motor	1000€	72-23-OF	Editar	Apagar

Figura 12 Mockup da Listagem de avarias

### 3.1.11 Listagem de ausências

Tabela com 4 colunas listando os dados relevantes das ausências. Devem também existir 2 colunas com botões de ações sobre as ausências (aceitar/rejeitar). No fim da tabela colocar um botão para atualizar a lista de ausências. Figura 13.

The mockup shows the same 'ReTruck' application interface. The sidebar and top navigation are identical. The main content area features a table titled 'Ausências Pendentes' with 6 columns: 'Data', 'Funcionário', 'Motivo', 'Descrição', 'Aceitar', and 'Rejeitar'. The table contains three rows of data, each with a green 'Aceitar' button and a red 'Recusar' button. Below the table is a black button labeled 'Carregar Atualizações'.

Data	Funcionário	Motivo	Descrição	Aceitar	Rejeitar
02/02/2022	Marco	Doença	Covid Pesado	Aceitar	Recusar
02/02/2022	Marco	Doença	Covid Pesado	Aceitar	Recusar
02/02/2022	Marco	Doença	Covid Pesado	Aceitar	Recusar

Carregar Atualizações

Figura 13 Mockup da Listagem de ausências

### 3.1.12 Listagem de Transportes

Tabela com 6 colunas, listando os dados relevantes dos transportes. Devem também existir 1 coluna com um botão para analisar o transporte. No fim da tabela colocar um botão para atualizar a lista de transportes. Figura 14.

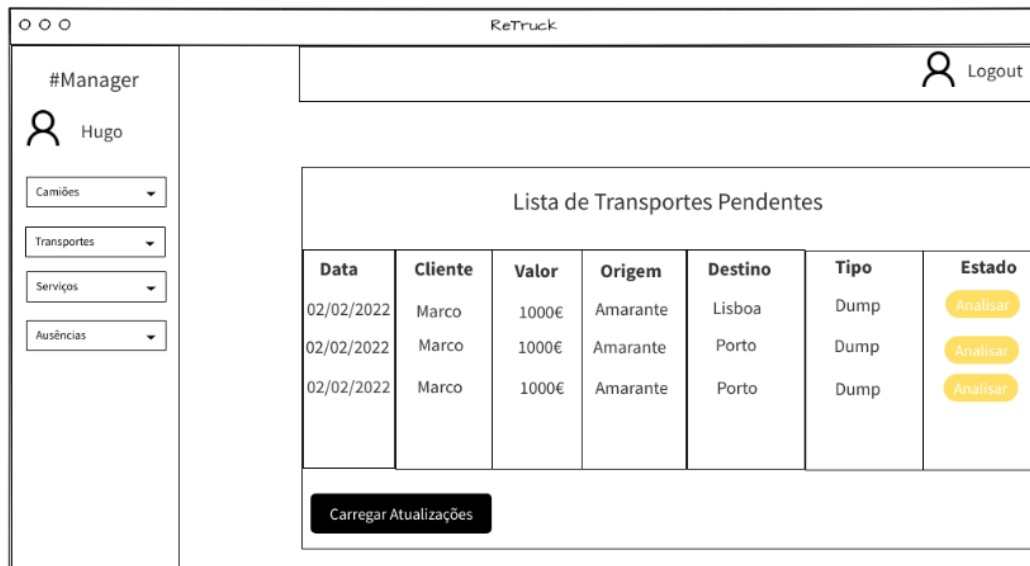


Figura 14 Mockup da Listagem de transportes

### 3.1.13 Detalhes de um transporte aceite

Dados do transporte e um mapa com a localização atual do camião. Figura 15.

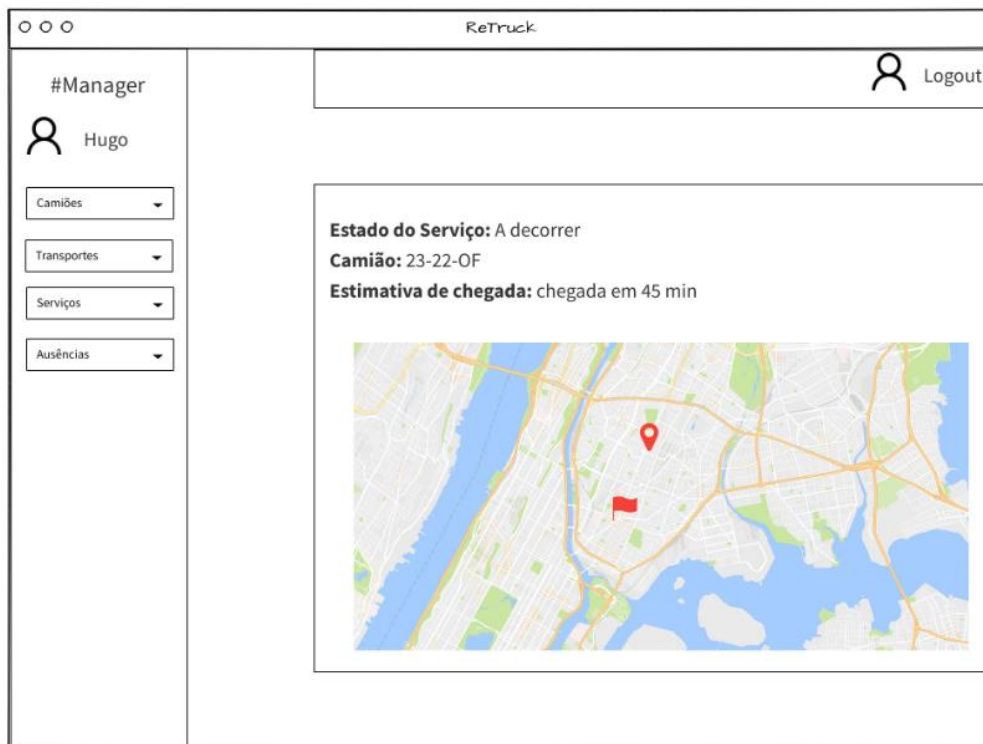


Figura 15 Mockup do serviço



### 3.1.14 Listagem de Organizações

Tabela com 4 colunas listando os dados relevantes das organizações. Devem também existir uma coluna com um botão para editar os dados de uma organização. Figura 16.

The mockup shows a web interface for 'ReTruck'. On the left is a sidebar with the user '#Super Admin' and 'Hugo', and two dropdown menus for 'Utilizadores' and 'Organizações'. The main content area has a 'Logout' button in the top right. Below it is a section titled 'Organizações' containing a table with 6 columns: '#', 'No', 'Nome', 'Localização', 'Vatin', and 'Ação'. The first row of data shows '#', '1', 'Demo', 'Amarante', '0', and an 'Editar' button.

#	No	Nome	Localização	Vatin	Ação
#	1	Demo	Amarante	0	<button>Editar</button>

Figura 16 *Mockup* Listagem de Organizações

### 3.1.15 Listagem de Utilizadores

Tabela com os dados relevantes de cada utilizador do sistema. A listagem deve estar dividida por perfis de utilizador de modo a facilitar a pesquisa por um *user*. Figura 17.

The mockup shows a web interface for 'ReTruck'. On the left is a sidebar with the user '#Super Admin' and 'Hugo', and two dropdown menus for 'Utilizadores' and 'Organizações'. The main content area has a 'Logout' button in the top right. Below it are two sections: 'Managers' and 'Drivers'. Each section contains a table with 6 columns: '#', 'No', 'Nome', 'Username', 'Email', and 'Ação'. The 'Managers' table has one row with a user icon, '1', 'Marco', 'marco1', 'hugsadf@gmail.com', and an 'Editar' button. The 'Drivers' table has one row with a user icon, '1', 'Marco', 'marco3', 'hugsadf@gmail.com', and an 'Editar' button.

#	No	Nome	Username	Email	Ação
	1	Marco	marco1	hugsadf@gmail.com	<button>Editar</button>

#	No	Nome	Username	Email	Ação
	1	Marco	marco3	hugsadf@gmail.com	<button>Editar</button>

Figura 17 *Mockup* da listagem de utilizadores

### 3.1.16 Resultado da análise de transporte

Página com o número de kms a percorrer (obtidos através da API do Google). Tabela com o melhor camião para realizar o serviço e por fim uma tabela que apresente a previsão de lucro que o serviço irá dar. Figura 18.

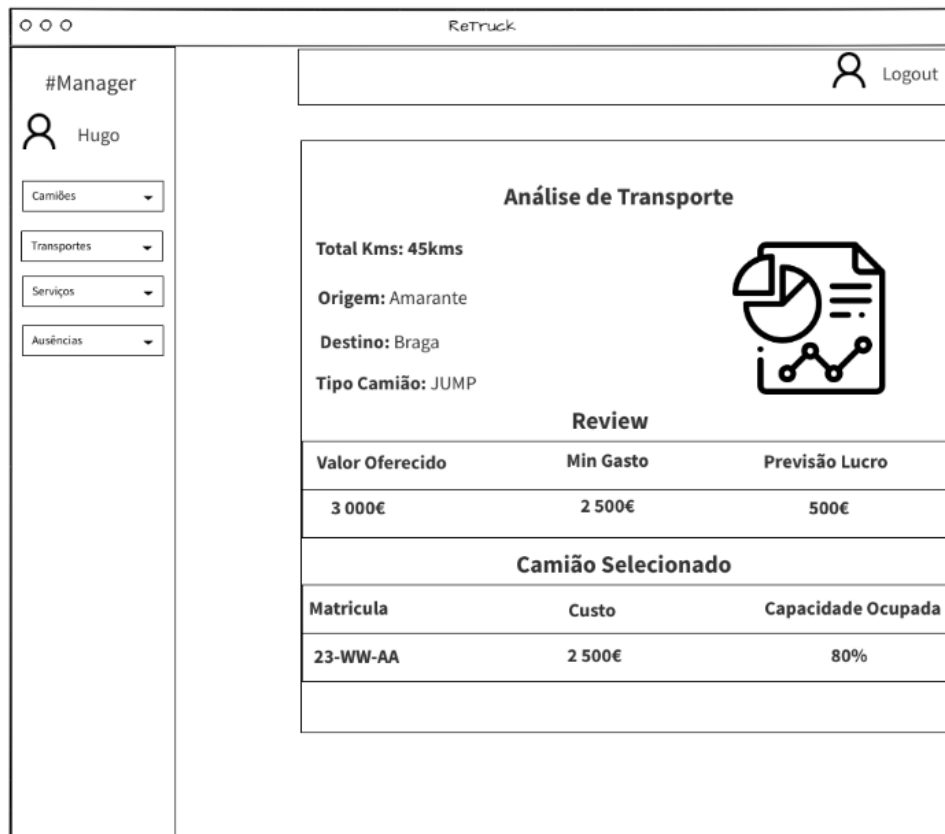


Figura 18 Mockup da análise de transporte

## 3.2 Software Interfaces

O desenvolvimento da API, com recurso à *framework* ASP.NET CORE deve cumprir o padrão MVC.

## 3.3 Interfaces de Comunicação

O sistema comunica com a API do Google Maps e com o Firebase através de HTTP. A resposta é recebida em formato JSON e será depois mapeada para os objetos dos *models* ou *dtos* que são usados no *back-end*. O sistema conta também com uma biblioteca para envio de *emails* designada de EmailJS que envia através do servidor SMTP um *email*. Dado que esta biblioteca só permite o envio de *emails* através de formulários, o *developer* responsável por essa tarefa deverá definir um procedimento para que o envio de *email* seja automático.

## 4. Use Cases

### 4.1 Gestão de Utilizadores e Organizações

#### 4.1.1 Descrição e Prioridade

Este *use case* tem como objetivo demonstrar a gestão de utilizadores e organizações. O ator deve poder criar utilizadores atribuindo-lhe um perfil, uma *password*, uma organização e um *username*. Por fim deve enviar para o *email* do novo utilizador os seus dados de acesso.

**Prioridade:** Essencial

**Ator:** *Super-Admin*

#### 4.1.2 Fluxo de Atividades, Criar utilizador

1. Ator efetua *login*.
2. Seleciona página de criação de utilizadores.
3. Insere os dados do novo utilizador.
4. Envia *Post* para o *back-end*.
5. Dados são guardados.
6. É mostrado um botão na interface de utilizador para o envio do *email*.
7. Ator clica no botão.
8. Email é enviado.

#### 4.1.3 Requisitos Funcionais

**REQ-1:** O sistema deve permitir efetuar a gestão de utilizadores.

**Critério de aceitação:** Qualquer utilizador que não seja *Super-Admin* não deve poder criar utilizadores.

**Critério de aceitação:** Não devem existir dois *users* com o mesmo nome.

**REQ-2:** O sistema deve permitir criar organizações.

**Critério de aceitação:** Não devem existir duas organizações com o mesmo nome.

**Critério de aceitação:** Não devem existir duas organizações com o mesmo *vatin*.

**Critério de aceitação:** Qualquer utilizador que não seja o *Super-Admin* não pode criar organizações.

**REQ-3:** O sistema deve permitir o envio de *emails*.

**REQ-4:** O sistema deve permitir listar os utilizadores do sistema por categorias (Cliente, Condutor e Gestores).

## 4.2 Gestão de Ausências

### 4.2.1 Descrição e Prioridade

Este *use case* tem como objetivo demonstrar a gestão das ausências dos funcionários. O registo de ausências irá permitir ao sistema fazer uma alocação de camiões para o serviço de uma forma mais realista.

**Prioridade:** Média

**Ator:** *Manager e Driver*

### 4.2.2 Fluxo de Atividades, criar uma ausência

1. *Driver* efetua *login* na aplicação *mobile*.
2. Seleciona página de registo de ausência.
3. Insere os dados da ausência (motivo, descrição e data).
4. Regista ausência.
5. *Manager* decide se aceita ou não a ausência.
6. Toma a decisão, ou clica em aceitar ausência ou em recusar.

### 4.2.3 Requisitos Funcionais

**REQ-5:** O sistema deve permitir criar ausências.

**Critério de aceitação:** O mesmo utilizador não pode ter duas ausências no mesmo dia.

**REQ-6:** O sistema deve permitir efetuar uma gestão de ausências (*Manager*).

**Critério de aceitação:** Ausência por doença ou motivos familiares deve ser aceite de forma automática.

**REQ-7:** O sistema deve permitir a listagem de todas as ausências de uma organização.

## 4.3 Gestão de Camiões

### 4.3.1 Descrição e Prioridade

Este *use case* tem como objetivo demonstrar a gestão de camiões no sistema.

**Prioridade:** Essencial

**Ator:** *Manager*

### 4.3.2 Fluxo de Atividades, criar camião

1. *Manager* faz *login* na aplicação.
2. Seleciona página de criar camião.
3. Insere a matrícula e o condutor.
4. Clica no botão de registar camião.
5. Dados são sobre o camião carregados do ficheiro XML (consumo, kms *etc*).
6. Camião é registado.

### 4.3.3 Fluxo de Atividades, criar camião não presente no XML

1. *Manager* faz *login* na aplicação.
2. Seleciona página de criar camião.
3. Insere a matrícula e o condutor.
4. Clica no botão de registar camião.
5. Dados não existem no ficheiro XML, retorna uma exceção e o camião não é criado.

### 4.3.3 Requisitos Funcionais

**REQ-8:** O sistema deve permitir criar camiões.

**Critério de aceitação:** Não devem existir dois camiões com a mesma matrícula.

**Critério de aceitação:** Apenas camiões registados no ficheiro XML devem ser criados.

**REQ-9:** O sistema deve permitir eliminar camiões.

**Nota:** Apenas o estado do camião deve ser atualizado para “*DISABLE*”.

**REQ-10:** O sistema deve permitir a listagem de todos os camiões de uma organização.

**Critério de aceitação:** Apenas camiões ativos na empresa devem ser listados ao utilizador.

## 4.4 Gestão de Avarias

### 4.4.1 Descrição e Prioridade

Este *use case* tem como objetivo demonstrar a gestão avarias e manutenções dos camiões. É importante no projeto pois irá fornecer informação para o módulo de alocação de transportes aos camiões.

**Prioridade:** Essencial

**Ator:** *Manager*

### 4.4.2 Fluxo de Atividades, Criar Avaria

1. *Manager* faz *login* na aplicação.
2. Seleciona página de criar avaria.
3. Insere os dados da avaria incluindo a data.
4. Clica no botão de registar avaria.

### 4.4.3 Fluxo de Atividades, Editar Avaria

1. *Manager* faz *login* na aplicação.
2. Seleciona página da listagem de avarias.
3. Seleciona a avaria a editar.
4. Altera os dados que entender.
5. Clica no botão de atualizar avaria.

### 4.4.4 Requisitos Funcionais

**REQ-11:** O sistema deve permitir criar avarias de camiões.

**Critério de aceitação:** Camião não tem duas avarias no mesmo dia.

**REQ-12:** O sistema deve permitir editar uma avaria.

**Nota:** Apenas os valores e descrição da avaria poderão ser atualizados.

**REQ-13:** O sistema deve permitir a listagem de todas as avarias nos camiões de uma organização.

**REQ-14:** O sistema deve permitir eliminar uma avaria de um camião.

**Nota:** Nesta situação a avaria deve ser mesmo eliminada do sistema.

**Exemplo:** Quando uma avaria é atribuída a um camião de forma errada, não existe necessidade de que esta continue na base de dados, contrariamente ao que acontece por exemplo na gestão de camiões quando um camião é eliminado.

## 4.5 Transportes

### 4.5.1 Descrição e Prioridade

Este *use case* tem como objetivo demonstrar a criação de um transporte.

**Prioridade:** Essencial

**Ator:** *Client*

### 4.5.2 Fluxo de Atividades, criar transporte

1. *Client* faz *login* na aplicação.
2. Seleciona página de criar transporte
3. Insere os dados do transporte incluindo a data.
4. Clica no botão de criar transporte.

### 4.5.3 Requisitos Funcionais

**REQ-15:** O sistema deve permitir criar transportes para os caminhões.

**Regras de Negócio:**

- Transporte tem sempre uma e apenas uma categoria associada.  
**Nota:** Sistema não deve fazer essa verificação, dado que o cliente pode decidir transportar por exemplo bebidas num transporte sem arca frigorífica.
- Transporte tem de ter um valor oferecido, só assim será possível efetuar a verificação se este deve ou não ser aceite.
- Transportes tem uma data, local de carga e descarga. São efetuados entre fabricas ou armazéns.

**REQ-16:** O sistema deve permitir listar todos os transportes pendentes de uma empresa.

## 4.6 Análise de Transportes e Gestão de Serviços

**Definição de serviço:** Transporte que foi aceite, ou seja passou a ser da responsabilidade da empresa.

### 4.6.1 Descrição e Prioridade

Este *use case* tem como objetivo demonstrar o processo de análise e seleção de um camião ou mais camiões para um transporte. Depois que o transporte é aceite passa a ser considerado um serviço e pode ser iniciado e terminado pelo *Driver*.

**Prioridade:** Essencial

**Ator:** *Manager e Driver*

### 4.6.2 Fluxo de Atividades, análise de transporte

1. *Client* faz *login* na aplicação.
2. Seleciona página de criar transporte
3. Insere os dados do transporte incluindo a data.
4. Clica no botão der criar transporte.
5. *Manager* recebe na *dashboard* notificação de um novo transporte.
6. Clica no botão para efetuar a review.
7. Seleciona uma das opções:

**Opção 1:** Rejeita o transporte.

- Atualizado o estado do transporte para rejeitado.

**Opção 2:** Aceita o transporte:

- Transporte passa a serviço.

**Nota:** Fluxo igual para todos os transportes.

### 4.6.3 Requisitos Funcionais

**REQ-17** O sistema deve permitir efetuar a análise de transportes.

**REQ-18** O sistema deve permitir atribuir um camião a um serviço.

**REQ-19** O sistema deve permitir ao condutor iniciar o serviço.

**REQ-20** O sistema deve permitir ao condutor terminar o serviço.

**REQ-21** O sistema deve permitir ao condutor informar a sua localização atual.



## 4.7 Dashboard

### 4.7.1 Descrição e Prioridade

Este *use case* tem como objetivo demonstrar dados sobre a atividade da empresa como por exemplo quanto foi faturado, quanto foi gasto em manutenções, *etc.*

**Prioridade:** Baixa

**Ator:** *Manager*

### 4.7.2 Fluxo de Atividades

1. *Manager* faz *login* no site.
2. Dados são mostrados na *homepage*.

### 4.7.3 Requisitos Funcionais

**REQ-21** O sistema deve permitir gerar informação através dos dados presentes na base de dados.

**Nota:** Dados podem ser apresentados sobre a forma de *labels* e gráficos. Para os gráficos aconselha-se a biblioteca React-Chart-JS<sup>6</sup>

### 4.7.3 Informações que podem ser geradas

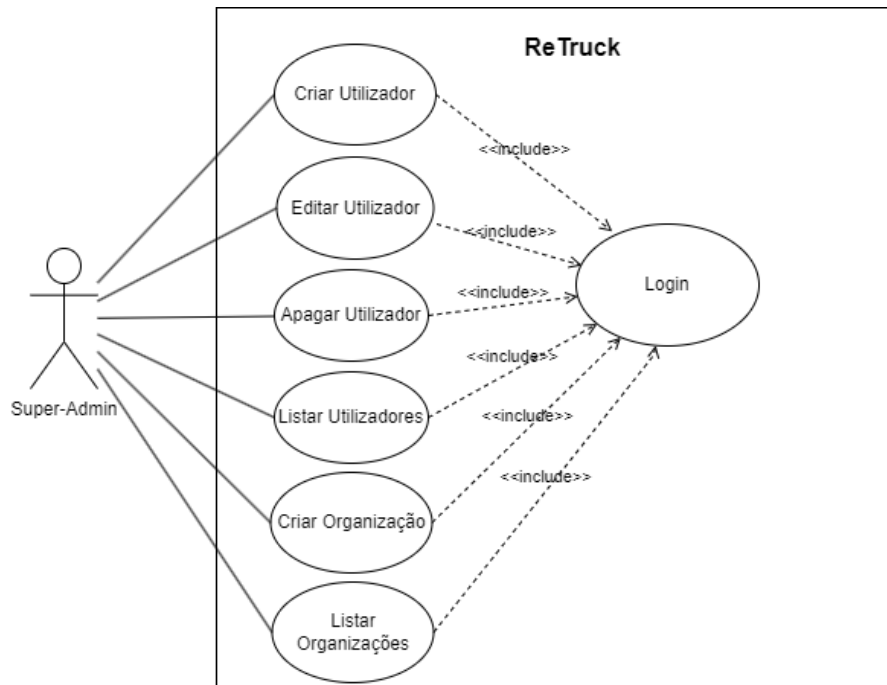
1. Quanto foi ganho com os serviços;
2. Quanto foi gasto em avarias/revisões;
3. Quantos camiões tenho, por categoria;
4. Quantos transportes rejeitei;
5. Quantos transportes aceitei;
6. Margem de lucro;
7. Preço do combustível por Petrolífera (Ex Galp e BP);

---

<sup>6</sup> <https://react-chartjs-2.js.org/>

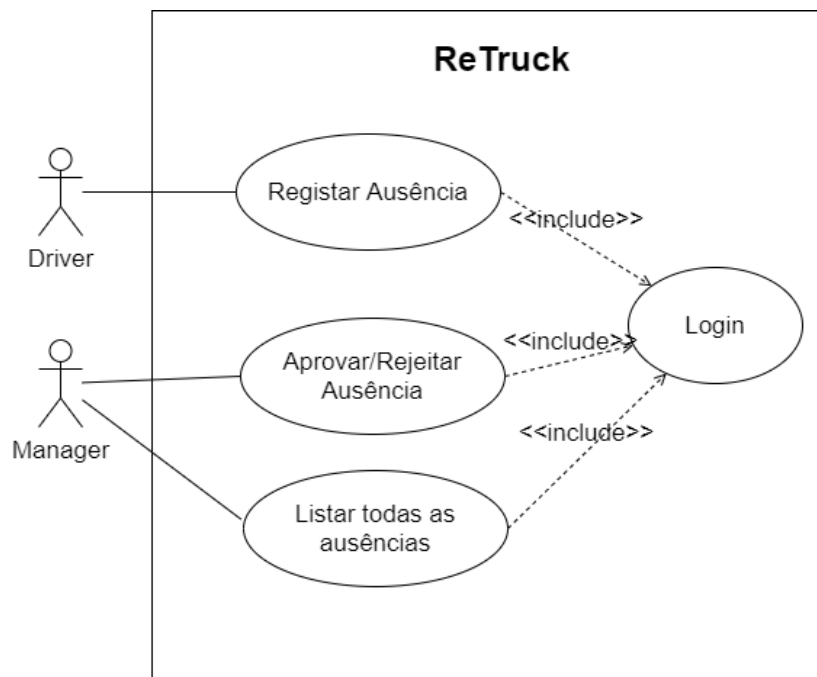
## 4.8 Diagramas Use Cases

A figura 19 demonstra as funcionalidades do ponto de vista do utilizador na gestão de utilizadores e organizações, capítulo 4.1 deste documento.



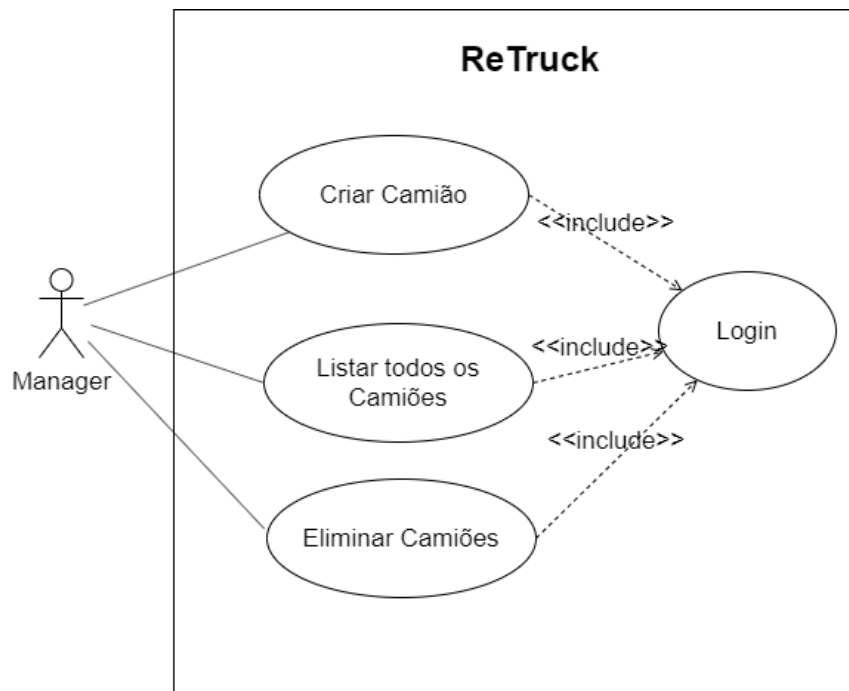
**Figura 19 Use Cases Super Admin**

A figura 20 demonstra as funcionalidades do ponto de vista do utilizador na gestão de ausências, capítulo 4.2 deste documento.



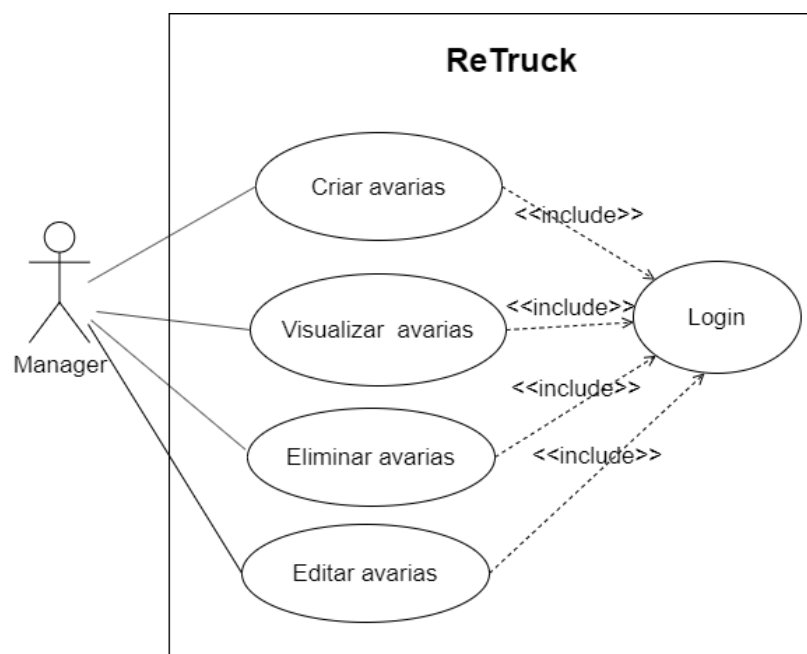
**Figura 20 Use Cases Gestão Ausências**

A figura 21 demonstra as funcionalidades do ponto de vista do utilizador na gestão de camiões, capítulo 4.3 deste documento.



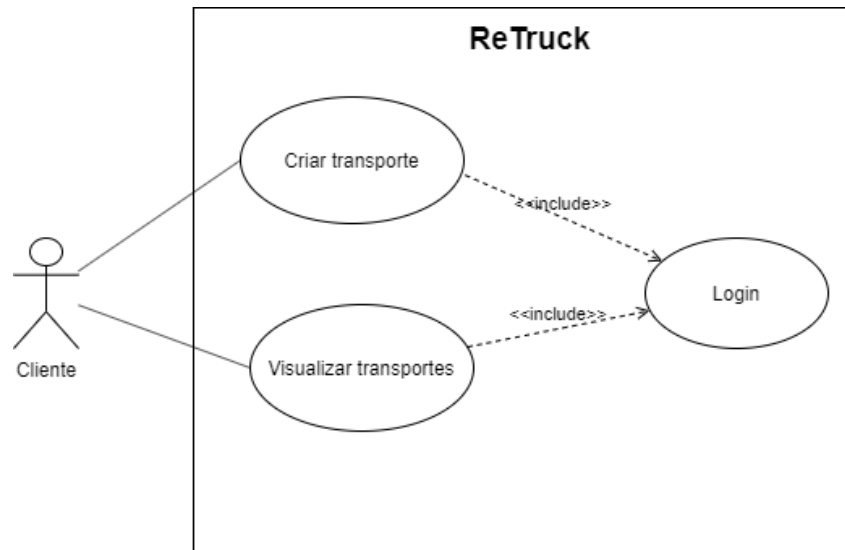
**Figura 21 Use Cases Gestão de Camiões**

A figura 22 demonstra as funcionalidades do ponto de vista do utilizador na gestão de avarias dos camiões, capítulo 4.4 deste documento.



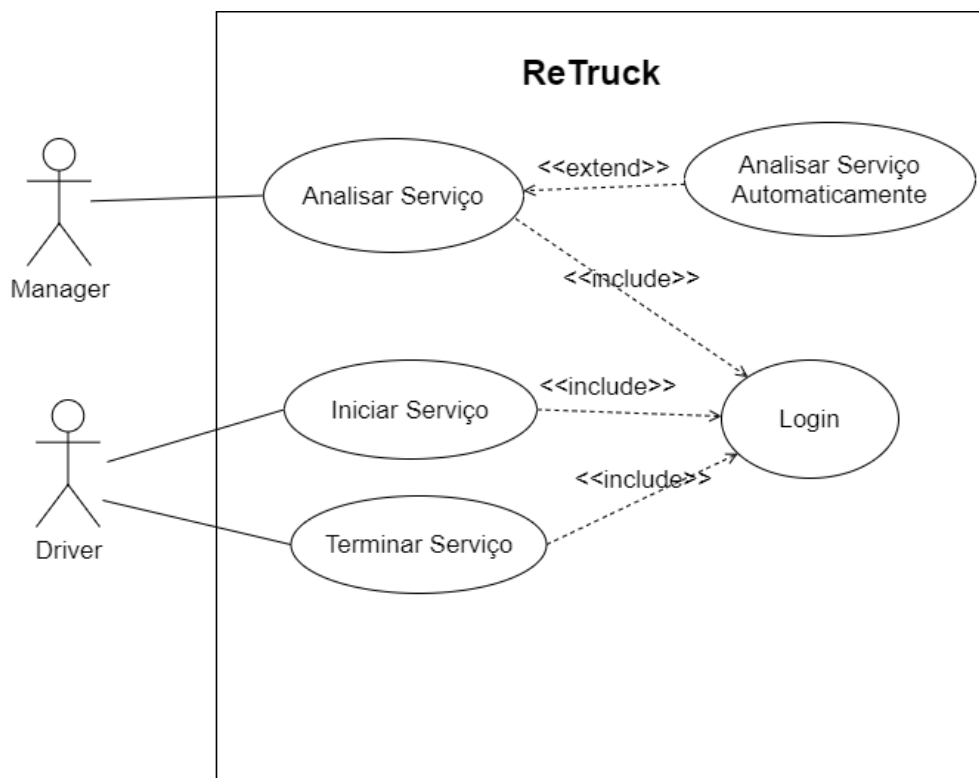
**Figura 22 Uses Case Gestão de Avarias**

A figura 23 demonstra as funcionalidades do ponto de vista do utilizador perante a classe transporte, capítulo 4.5 deste documento.



**Figura 23 Use Cases Transporte**

A figura 24 demonstra as funcionalidades do ponto de vista do utilizador na gestão de serviços, capítulo 4.6 deste documento.



**Figura 24 Use Cases Serviços**

## 5. Requisitos Não Funcionais

Os requisitos a seguir apresentados são os requisitos não funcionais. Por vezes denominados de atributos de qualidade estes requisitos tem um papel importante durante o desenvolvimento do projeto, nomeadamente na forma como se irá entregar as funcionalidades ao utilizador.

**RNF1:** O sistema deve ter características responsivas, ou seja, a interface deve ser adaptada a todas as plataformas (pc, tablet, *smartphone*)

**Categoria:** Responsividade

**Prioridade:** Essencial

---

**RNF2:** As informações pessoais dos utilizadores devem ser protegidas (por exemplo, a *password* deve ser encriptada).

**Categoria:** Proteção de dados

**Prioridade:** Essencial

---

**RNF3:** Apenas utilizadores autorizados podem fazer alterações à base de dados

**Categoria:** Segurança

**Prioridade:** Essencial

---

**RNF4:** O sistema deve estar disponível 365 dias por ano 24 horas por dia

**Categoria:** Fiabilidade / Disponibilidade

**Prioridade:** Baixa

---

**RNF5:** O *software* deve ser desenvolvido tendo em conta a possibilidade de o reutilizar.

A adição de novas funcionalidades deve ser simples.

**Categoria:** *Software* reutilizável

**Prioridade:** Baixa

---

**RNF6:** As interações do utilizador com o site devem ter um tempo de resposta no máximo de 400 milissegundos.

**Categoria:** Performance

**Prioridade:** Essencial

## 6. Anotações

### 6.1 Super Admin

O ator *Super-Admin* deve ser um utilizador inserido manualmente na base de dados. Para o desenvolvimento devem ser utilizados os seguintes dados:

**Username:** root    **Password:** string

**Nota:** A *password* deve ser inserida na base de dados encriptada de modo que funcione normalmente depois do *login* estar implementado.

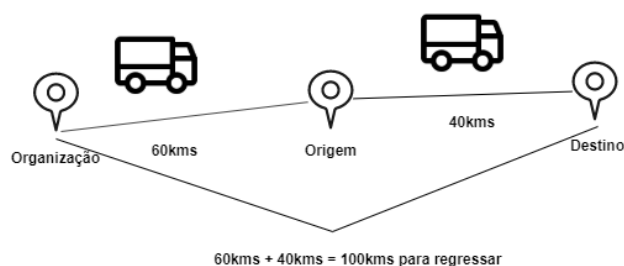
### 6.2 Algoritmo de análise e seleção

Todas as variáveis que devem ser usadas:

- Consumo do camião
- Capacidade do camião
- Tipo do camião
- Número de kms a percorrer (ida e volta)
- Tipo do transporte
- Disponibilidade do camião e do condutor para aquele dia
- Valor ganho pelo serviço
- Preço do combustível
- Número de avarias de um camião, camiões com mais avarias devem ser preteridos das escolhas.

**Nota Importante:** Cada camião tem uma organização, no cálculo dos km deve ser usado a distância da organização até à origem do serviço.

Exemplo: 60 kms da sede organização à origem do serviço, 40 kms o serviço e por fim mais 100kms para regressar de novo à sede da organização = 200kms, tudo isto deve ser calculado.



**Figura 25 Exemplo de Cálculo**

### 6.3 Algoritmo de análise e seleção de camiões, pensa duas vezes

Numa empresa manter os clientes por muitos e longos anos é essencial, por isso quando o algoritmo que faz a análise de um transporte e conclui que este não é rentável deve ser verificado o histórico daquele cliente, de modo a perceber se existe um saldo favorável. Caso o saldo seja favorável deve ser exibido um alerta para o *Manager*, dado que um cliente que ofereça 10 bons serviços e um mau não deixa de ser um bom cliente e, às vezes, um serviço recusado pode levar à perda do cliente.

Caso o saldo não seja favorável deve ser exibido um alerta remetendo para a rejeição do transporte, contudo a última decisão será sempre do utilizador.

A figura 26 demonstra o fluxo que o sistema deve ter no processo de análise de um transporte.

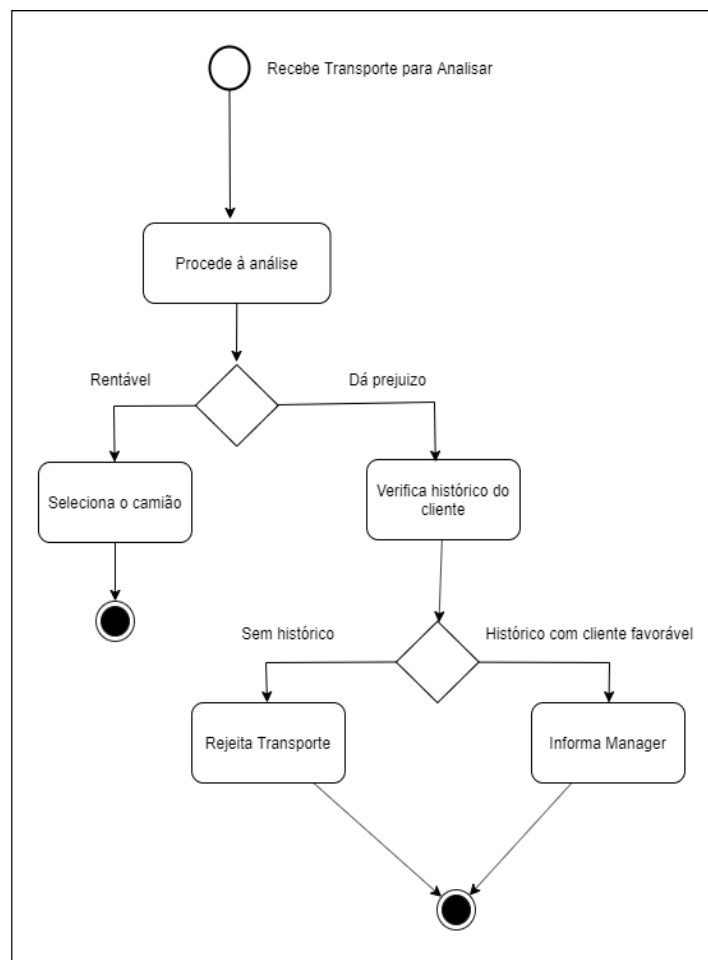


Figura 26 Diagrama de atividades

## 7. Arquitetura do Projeto

A figura 27 representa a arquitetura que será implementada no ReTruck. O sistema conta com 3 componentes principais.

**Back-end**, desenvolvido .NET *framework* 6.0, linguagem C# com base de dados SQL Server, responsável por:

- Efetuar ligação entre *front-end* e base de dados através de uma *Rest API*.
- Implementar as regras de negócio.

**Front-end**, desenvolvido em React para a *WEB* e em JAVA na aplicação *mobile*, responsável por.

- Permitir a interação dos utilizadores com o sistema.

**Base de dados:** SQL Server e Firebase, todos os dados da base de dados Firebase serão recolhidos pela API e inseridos na base de dados SQL.

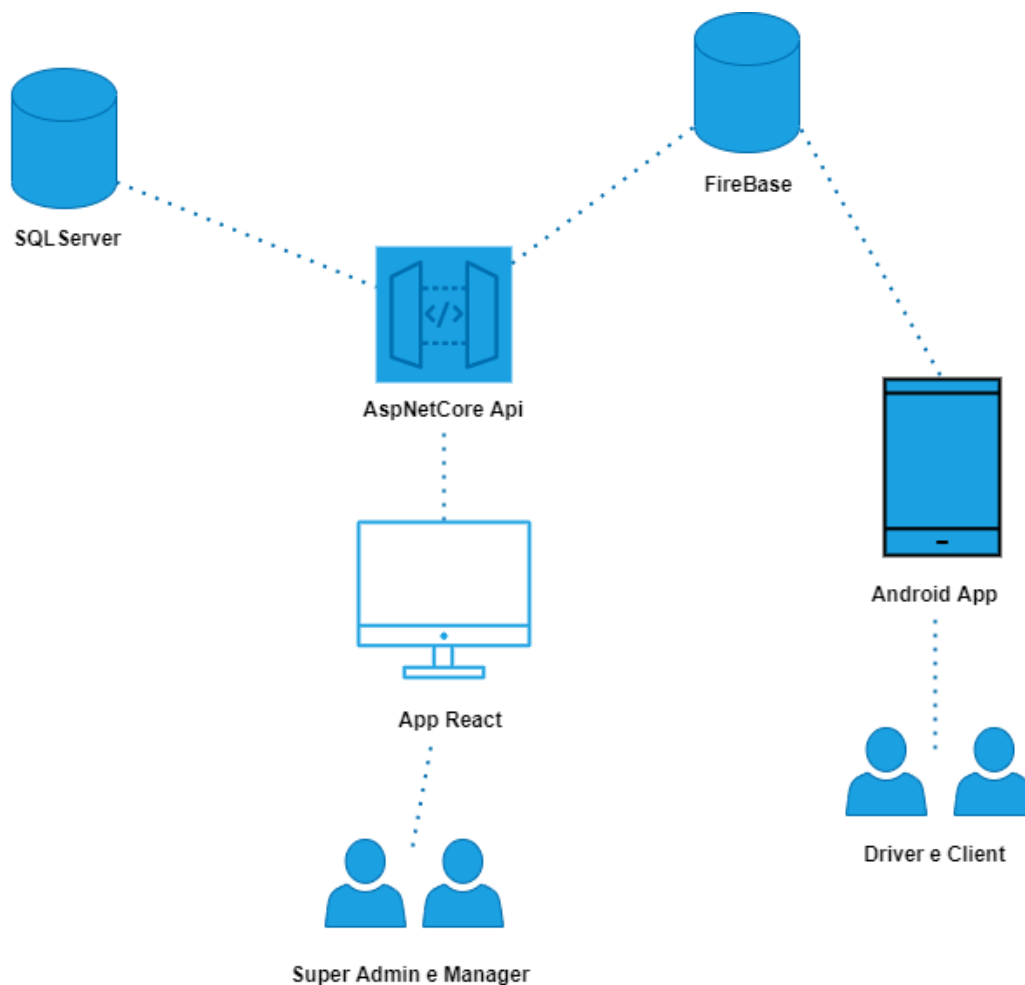


Figura 27 Arquitetura do ReTruck



## 7.1 Notas sobre a arquitetura

A aplicação *mobile* deve ter a funcionalidade de *login*, uma vez que o registo só é permitido através da aplicação *Web* os dados dos utilizadores devem ser exportados para a base de dados Firebase recorrendo à API.

Para o correto funcionamento do sistema a API deve também conter um método que faça a leitura e respetiva conversão em objetos de um ficheiro XML que permitam armazenar os dados relativos a cada camião. O ficheiro deve ter a mesma estrutura em todos os elementos, sendo que dependendo do tipo de camião a variável “capacity” terá interpretações diferentes.

**Como deve ser interpretada a variável *capacity*:**

- Camião de despejar fora com 1000 de *capacity*, representa o peso máximo que este pode transportar, 1000 kg.
- Camião-Cisterna com 1000 de *capacity*, representa o número de litros máximo que este pode transportar, 1000 litros.
- Camião frigorífico com 1000 de *capacity*, representa o volume máximo que este pode transportar, 1000 m<sup>3</sup>.

## 8. Diagrama de Classes

O diagrama apresentado, figura 28, representa as classes que são necessárias para a criação da base de dados, que deve ser feita através da abordagem *code first*. Recomenda-se a biblioteca EntityFramework Core para a realização deste processo.

1. Organization-User: Uma Organização tem **um conjunto** de utilizadores, um utilizador só pode estar associado **uma** organização.
2. Organization - Truck: Uma Organização tem **um conjunto** de camiões, um camião só pode estar associado **uma** organização.
3. Driver- Truck: Um camião tem sempre **um** condutor, um condutor só poderá estar associado a **um** camião.
4. Truck - ServiceTransport: Um serviço tem sempre **um** camião associado, um camião pode estar associado a **vários** serviços.

**Regras de Negócio:** **um** camião pode estar associado a vários serviços em dias diferentes, ou seja, por dia o camião apenas pode estar associado a um serviço, *requisito funcional no 18*.

5. Truck - ServiceTransport: Um camião possui um conjunto de avarias, uma avaria está associada apenas a **um** camião.
6. Transport- ServiceTransport: Um transporte pode estar associado a **vários** serviços, um serviço só está associado a **um** transporte.
7. User-License: Um *driver* é um *user* que possui **uma** licença de condução.
8. User-Absence: Um *user* tem **várias** ausências, no entanto uma ausência só pode estar associada a **um** *user*.
9. Service-ServiceCoor: Um serviço tem um **conjunto** de localizações, uma localização apenas está associada a **um** serviço.

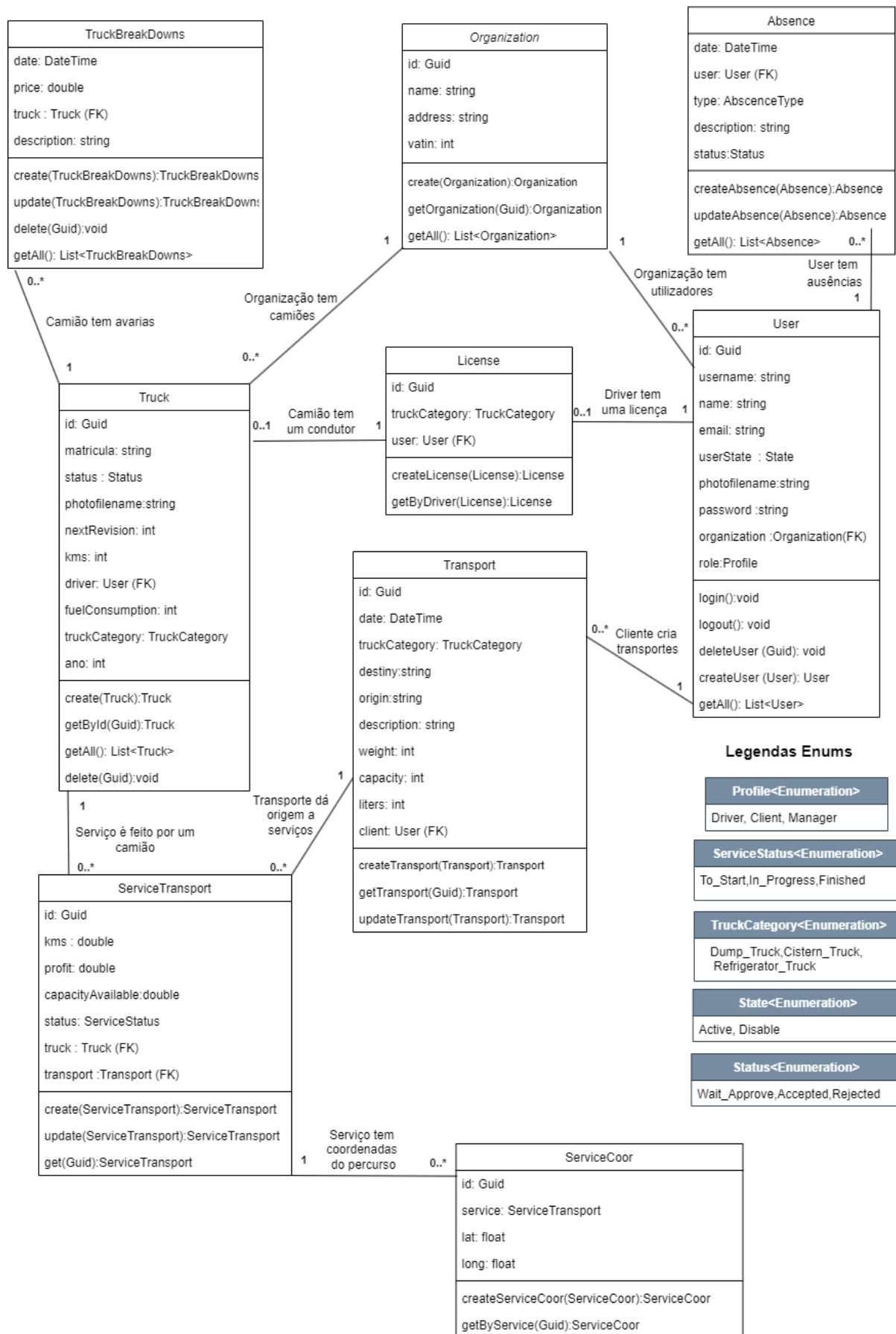


Figura 28 Diagrama de Classes

## 9. Notas de Anexo

Parâmetros a incluir no algoritmo de análise e seleção de camiões:

- Preço combustível;
- Preço da Portagem;
- Trabalho num feriado ou fim de semana leva a uma despesa maior? Se sim quanto?
- Tipo de análise;
- Avarias influenciam a pontuação do camião? Sim ou Não;
- Análise para o melhor camião para o serviço? o que considerar? Custo? Ocupação? Os dois?