Escola Superior de Tecnologia e Gestão

Licenciatura em Engenharia Informática

Escola Superior de Tecnologia e Gestão

Instituto Politécnico de Viana do Castelo

Unidade curricular de

Projeto I

Relatório de projeto

2022/2023

Gestão de uma empresa de transporte pesados de mercadorias

28257 – Diogo Manuel Machado Assunção,

28248 – David José Sousa Braga

Sumário

1. Introdução

1.1 - Apresentação do tema

Neste projeto desenvolvido na UC de Projeto I, temos como objetivo desenvolver diferentes modelos para melhorar a organização e a gestão dos meios disponíveis da empresa ReiPort.

ReiPort é uma empresa dedicada ao transporte de contentores em camiões (transportes pesados) na União Europeia.

Este projeto consiste na gestão dos pedidos de entrega (feitos pelos clientes) e dos meios para os referidos trabalhos (quer de logística, quer de mão de obra).

1.2 - Objetivos do projeto

Numa primeira fase do projeto, apresentamos os modelos BPMN e a descrição dos respetivos processos.

Numa segunda fase, apresentamos os diferentes tipos de utilizadores, bem como as suas ações no sistema, além de um levantamento de requisitos funcionais e não funcionais. Ainda se expôs os diferentes modelos e diagramas de design e modelação (Diagrama DER, Modelo de Domínio, Modelo de Classes e Modelo Relacional) e modelos e diagramas funcionais (Modelo de Casos de Uso, Templates de Caso de Uso, Diagrama de transição de estados, Diagramas de Sequência para 3 Templates de Caso de Uso e um Diagrama de Atividades).

Numa terceira fase, apresentamos o código utilizado no Postgres para criar as tabelas, os inserts, as views, os triggers, os indexes, os store procedures e por fim os updates e os deletes.

2. Apresentação do negócio

Nesta secção apresentaremos os modelos de negócio (em BPMN) e os diferentes processos.

2.1 - Âmbito e enquadramento do negócio

Esta empresa apenas realiza viagens de entrega, por exemplo: transporta a mercadoria para portos, fábricas, depósitos, empresas, comboios, aeroportos, etc. O rececionista recebe um pedido de trabalho, anota os dados e envia para o gestor, que verifica e agenda a entrega, além disso gere também os camiões e contentores a serem utilizados (se os camiões e contentores são disponibilizados pelo cliente ou pela empresa ReiPort), e os motoristas (se forem viagens de mais de 9 horas são necessários 2 motoristas). Ao receber a carga e antes de iniciar o transporte da mesma é realizada a inspeção e documentação da carga pelo/s motorista/s do pesado em questão, e no destino é realizada a verificação novamente para saber se a carga foi entregue em condições. Para os vários trabalhos, ReiPort dispõe do número necessário de camiões, contentores e motoristas para a realização dos trabalhos.

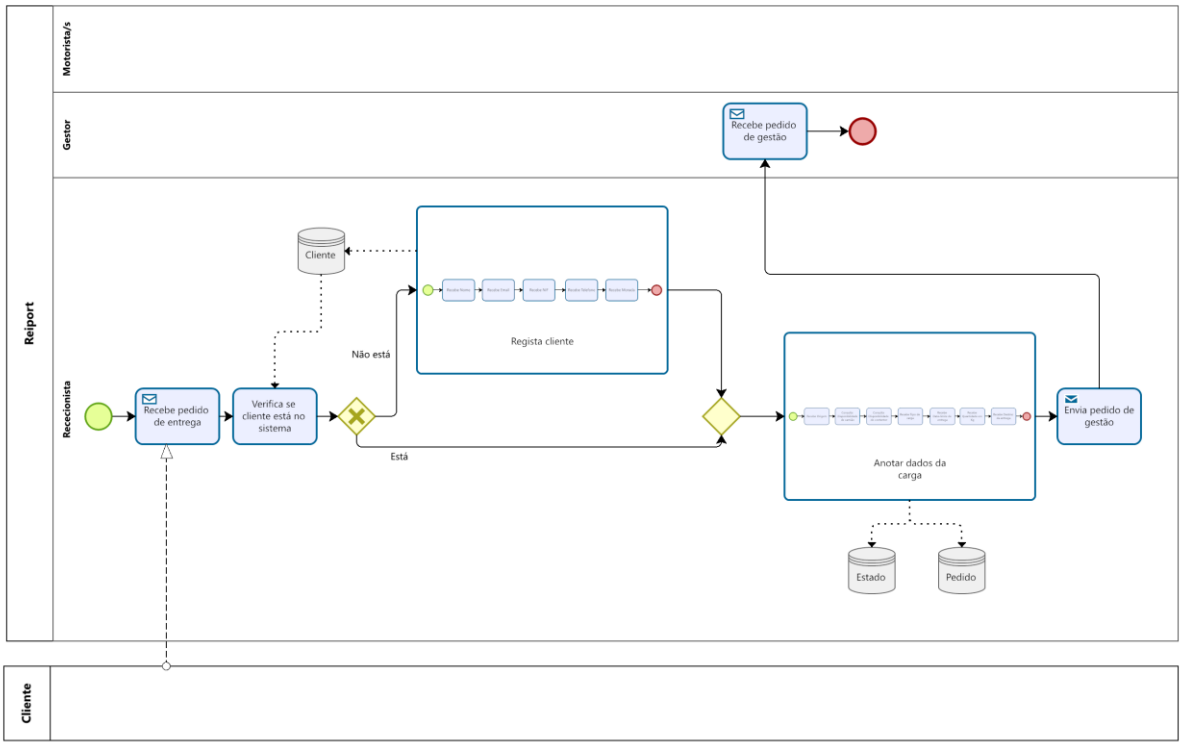
2.2 - Modelação dos Processos de negócio

Secção onde apresentaremos os modelos de negócio formulados.

2.2.1 Processo 1 – Processo de estabelecimento de mercadoria

Este processo consiste em receber um pedido de entrega, registá-lo e, posteriormente, enviá-lo para o gestor, para iniciar a fase seguinte.

Inicia-se com a receção de um novo pedido por parte de um cliente. O rececionista verifica se o cliente está na base de dados e, se não estiver, procede ao seu registo. No registo é pedido o nome, o email, o NIF, o telefone e, por fim, a morada. Depois regista a carga. Num diálogo com o cliente, configura os detalhes da entrega (origem da entrega, disponibilidade de camião, disponibilidade de contentor, dimensões, peso da carga, data-limite de entrega, tipo de carga, tamanho da carga, destino da entrega). De seguida, envia um pedido de gestão para o gestor e coloca o estado da entrega como suspensa.



2.2.2 Processo 2 – Processo de gestão e configuração da entrega

Este processo consiste em receber um pedido de gestão e proceder à organização e gestão dos meios disponíveis na empresa ou pelo cliente e a elaboração do CMR.

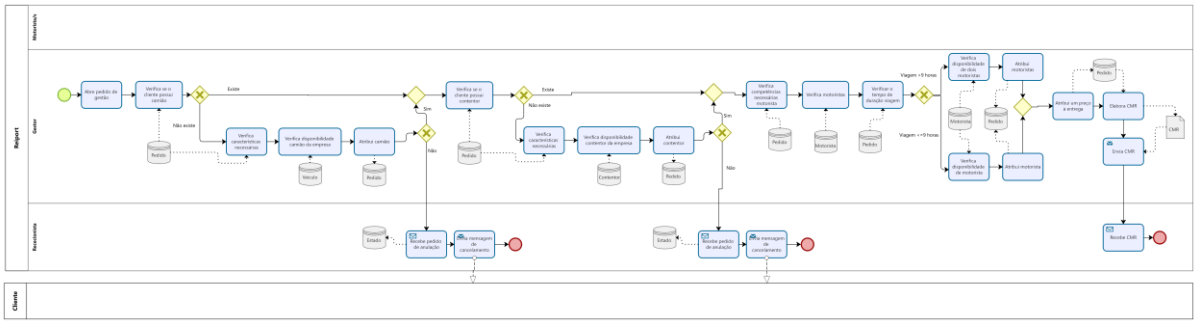
Inicia o seu trabalho verificando a disponibilidade de camião, isto é, confirma se o cliente fornece um camião para a entrega ou não. Se este não fornecer, é procurado um camião com as características pretendidas para o trabalho de acordo com o stock da empresa ReiPort. Se não for encontrado, o gestor envia um pedido de anulamento ao rececionista que envia uma mensagem de cancelamento ao cliente e altera o estado da entrega para anulada.

Procede, então, à verificação da disponibilidade de contentor onde executa o mesmo procedimento realizado anteriormente.

Verifica os motoristas com competências para transportar o tipo de carga a entregar (por exemplo: para viagens de transporte de materiais inflamáveis, o motorista necessita do ADR, …) e o tempo de duração da viagem. Em conformidade com a duração da viagem prevista, decide se o motorista necessita de um copiloto (viagens com mais de 9 horas).

Atribui, assim, um ou mais motoristas e um preço à entrega.

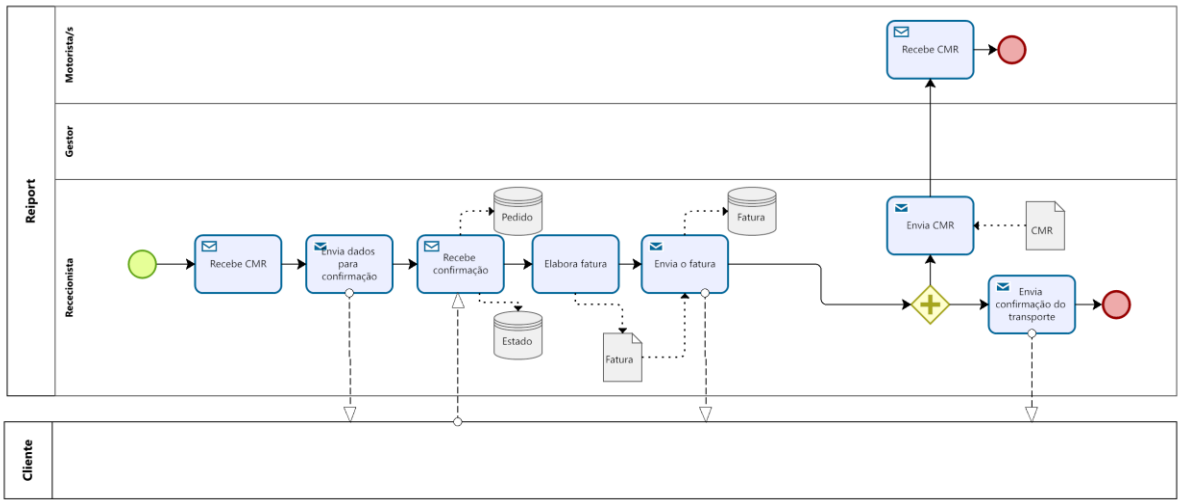
Elabora um CMR com toda a informação necessária para o/s motorista/s entregar/em a entrega. No fim envia o CMR ao rececionista.



2.2.3 Processo 3 – Processo de confirmação de dados e faturação

Este processo consiste na confirmação dos dados com o cliente, envio da data prevista da entrega e emissão da fatura.

Assim sendo, depois de enviar os dados recebe a confirmação do cliente e elabora a fatura que envia para o cliente, bem como altera o estado da entrega para agendada. Envia, também, o CMR para o/s motorista/s e a confirmação de transporte para o cliente.



2.2.4 Processo 4 – Processo de finalização do pedido de entrega e inspeção

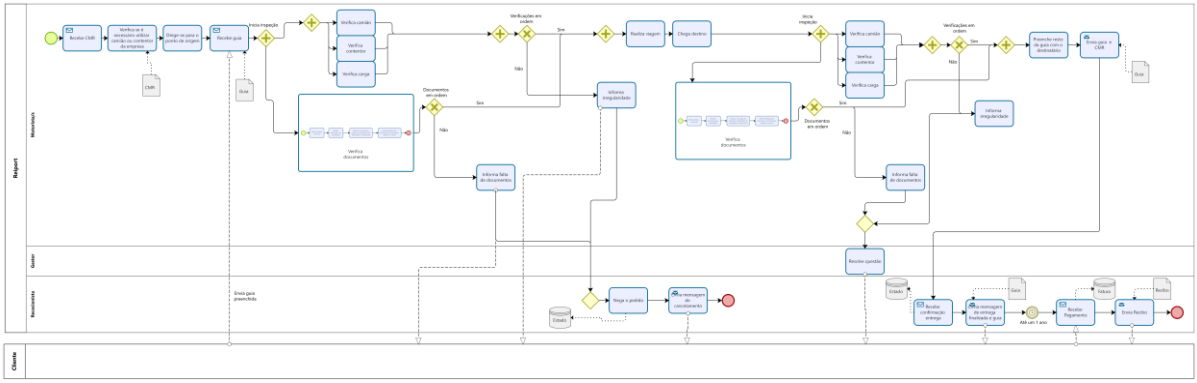
Este processo final consiste na viagem do motorista e das duas inspeções realizadas à carga, bem como a receção do pagamento.

O/s motorista/s recebe/m os detalhes enviados pelo rececionista (CMR) e procede/m ao levantamento do camião ou contentor, dependendo da verificação realizada pelo gestor quanto à utilização de camião ou contentor da empresa ou do cliente. Os motoristas dirigem-se para o ponto de origem da viagem de entrega. Na receção da mercadoria, recebem a guia devidamente preenchida pelo cliente. A carga é carregada e inicia-se o processo de inspeção da carga com a presença do cliente. Se o estado da carga e do camião e/ou contentor, bem como todos os documentos (necessários para o tipo de carga a transportar (fatura comercial, declaração de exportação, fotocópia do cartão de contribuinte do exportador e importador e por fim do parking-list (descrição da mercadoria, volumes e pesos))) estiverem em ordem, a entrega é realizada. Se por algum motivo, os documentos ou o estado de um dos três não estiver em ordem o cliente é informado, tal como o rececionista. Este nega o pedido e envia mensagem de cancelamento definitivo ao cliente.

O motorista realiza a entrega e, no destino, volta a verificar o estado da carga, do camião e/ou contentor, bem como todos os documentos. Se por algum motivo algo estiver incorreto, informa a/s irregularidade/s ao cliente e ao gestor, para que a situação seja resolvida.

Preenche o resto da guia com o destinatário e envia a guia para o rececionista.

O rececionista recebe a guia, altera o estado da entrega para concluída e envia para o cliente a guia. Por fim, o cliente paga o serviço e é enviado o recibo correspondente.



3. Levantamento de Requisitos

3.1.1 - Apresentação dos tipos de utilizador

- Rececionista

- Gestor

-Motorista

-Cliente

-Admin

3.1.2 - Ações de cada tipo de utilizador

O rececionista consegue registar novos clientes e fazer alterações, registar pedidos de entrega, visualizar os pedidos em procedimento, os pedidos agendados, envia ao cliente mensagens de cancelamento (se houver irregularidades no início da entrega) ou de conclusão do trabalho, atualiza o estado da entrega no sistema e recebe a confirmação destes detalhes, emite a fatura, envia a confirmação do início da entrega e o CMR ao motorista e recebe e envia a guia de transporte ao cliente. Consegue elaborar a fatura e recebe o pagamento do transporte, emite o recibo e envia-o para o cliente.

O gestor pode visualizar os pedidos em procedimento, os pedidos agendados, a logística da frota de camiões e contentores, os motoristas disponíveis, e definir o camião, contentor e motorista/s para o trabalho, além de atribuir um preço à entrega e emitir o CMR, que depois será utilizado pelo/s motorista/s. Além disso, tem de avaliar e corrigir qualquer irregularidade que surja durante a entrega. O gestor pode também registar o camião e/ou contentor para adicionar ao Stock.

O/s motorista/s recebem toda a informação necessária para o seu trabalho (na Guia de Transporte), além de realizar/em as inspeções à carga e aos meios utilizados no transporte. Também são responsáveis por receber e entregar as guias de transporte ao rececionista e, evidentemente, realizar a viagem.

O cliente pode verificar a qualquer altura o estado das suas encomendas, bem como alterar os seus dados, desde que efetue o seu registo no sistema.

O administrador pode gerir qualquer tipo de funcionário (rececionista, gestor e motorista).

3.2 - Requisitos Funcionais

Vocabulário	
CMR	É um documento de acompanhamento. É o documento preparado pela empresa para entregar ao motorista de um veículo.
Guia de transporte	Uma guia de transporte é um documento que declara que uma transportadora leva uma carga de A até B em nome do fornecedor. A guia de transporte declara as mercadorias contidas numa carga e o local onde serão entregues. A guia de transporte declara se devem ser cumpridas obrigações especiais para as mercadorias em questão (que tipo de habilitações são necessárias pelo motorista). O peso é também indicado e pode ser utilizado para verificar se o camião tem excesso de carga.

Requisitos Funcionais		
Referência	Descrição	Prioridade
RF01	Como rececionista, quero Registar um cliente para geri-lo no sistema	Alta
RF02	Como rececionista, quero Registar Pedido para os poder gerir	Alta
RF03	Como rececionista, quero poder Alterar os Dados dos Clientes e da sua carga para os atualizar	Média
RF04	Como rececionista, quero Visualizar todos os Pedidos (agendados, execução, suspensos, anulados e completos) para poder geri-los	Média
RF05	Como rececionista, quero poder enviar avisos de cancelamento, início do trabalho ou conclusão para o cliente	Baixa
RF06	Como rececionista, quero poder Alterar os Estados de Entrega para os atualizar	Alta
RF07	Como rececionista, quero enviar a confirmação do transporte para o cliente	Baixa
RF08	Como rececionista, quero poder Enviar Ficheiros para diferentes utilizadores do sistema	Alta
RF09	Como rececionista, quero poder aceder às guias de transporte para as enviar para os clientes	Alta

RF10	Como rececionista, quero Elaborar/Emitir Faturas para posteriormente enviar para os clientes	Média
RF11	Como rececionista, quero Elaborar/Emitir Recibo para posteriormente enviar para os clientes que já pagaram	Média
RF12	Como rececionista, quero Consultar os Ficheiros relativos à entrega para confirmar entregas com clientes	Alta
RF13	Como gestor, quero Visualizar todos os Pedidos (agendados, execução, suspensos, anulados e completos) para poder geri-los	Alta
RF14	Como gestor, quero Visualizar o Stock de Veículos e contentores da empresa ReiPort para poder geri-los	Alta
RF15	Como gestor, quero Consultar os Motoristas disponíveis para poder geri-los	Média
RF16	Como gestor, quero poder atribuir camião, contentor e motorista para a realização do trabalho	Alta
RF17	Como gestor, quero verificar se o cliente possui camião e/ou contentor para saber se é necessário atribuir camião da ReiPort	Média
RF18	Como gestor, quero verificar se a ReiPort possui camião para a entrega para anular o pedido se este não existir	Média
RF19	Como gestor, quero verificar se a empresa possui contentor para a entrega para anular o pedido se este não existir	Média
RF20	Como gestor, quero verificar motoristas com as habilitações necessárias para atribuir um trabalho onde estas são necessárias	Média
RF21	Como gestor, quero verificar se o tempo de duração da viagem é superior a nove horas para atribuir 2 motoristas	Baixa
RF22	Como gestor, quero calcular o preço de entrega de acordo com as condições definidas	Baixa
RF23	Como gestor, quero Elaborar CMRs para o enviar à rececionista e Visualizá-los	Alta
RF24	Como gestor, quero Resolver um aviso de Irregularidade para o poder, posteriormente, resolver	Alta
RF25	Como rececionista, quero poder aceder ao CMR para confirmar os dados com o cliente	Alta

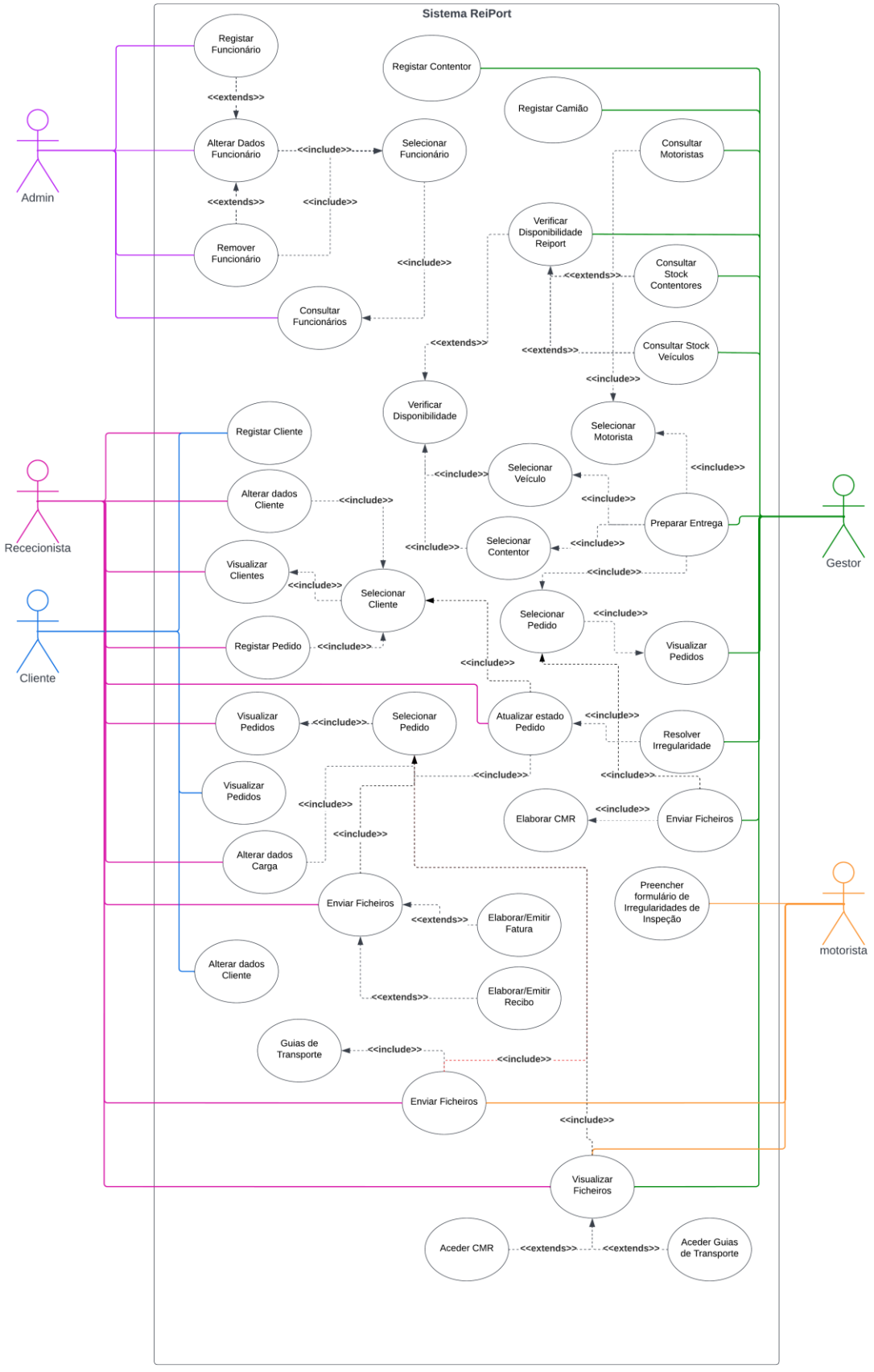
RF26	Como motorista, quero poder aceder a todos os ficheiros relativos ao transporte (Guia de transporte, CMR) para conhecer todos os aspetos da entrega	Alta
RF27	Como motorista, quero Preencher um Formulário de Irregularidades na inspeção do camião, contentor e/ou da carga para colocar no sistema	Alta
RF28	Como motorista quero Enviar Ficheiros (ex: guia de transporte preenchida) para o sistema	Alta
RF29	Como cliente, quero me poder Registar no sistema para previsão de serviço	Alta
RF30	Como cliente, quero Visualizar todos os meus Pedidos (agendados, execução, suspensos, anulados e completos) para ter conhecimento do progresso dos pedidos	Alta
RF31	Como cliente, quero poder Alterar os meus Dados para os atualizar	Alta
RF32	Como gestor, quero Registar Camião e/ou Contentor para os adicionar ao Stock	Alta
RF33	Como Admin, quero Registar, Alterar e Remover qualquer tipo de funcionário para fazer previsão do sistema	Alta

3.3 - Requisitos não funcionais

Requisitos Não Funcionais	
Referência	Descrição
RNF01	O sistema deve obrigar o gestor a colocar dois motoristas, quando são estimadas mais de nove horas de viagem
RNF02	O cliente tem até um ano para efetuar o pagamento
RNF03	Quando o motorista dá início ao transporte, pelo sistema, o estado da entrega atualiza, automaticamente, para Execução
RNF04	Um contentor apenas pode carregar um tipo de carga específico
RNF05	Um pedido apenas e só realiza uma entrega
RNF06	Quando o motorista preenche o formulário de irregularidades no transporte, pelo sistema, o estado da entrega atualiza, automaticamente, para Anulado

4. Design e Modelação

4.1 - Modelo de casos de uso



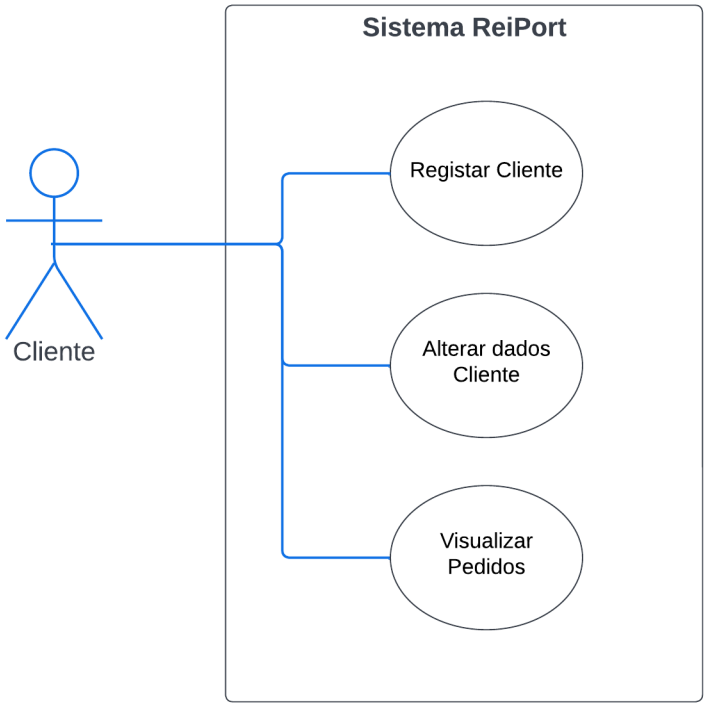


FIGURA 1 - CASO DE USO - CLIENTE

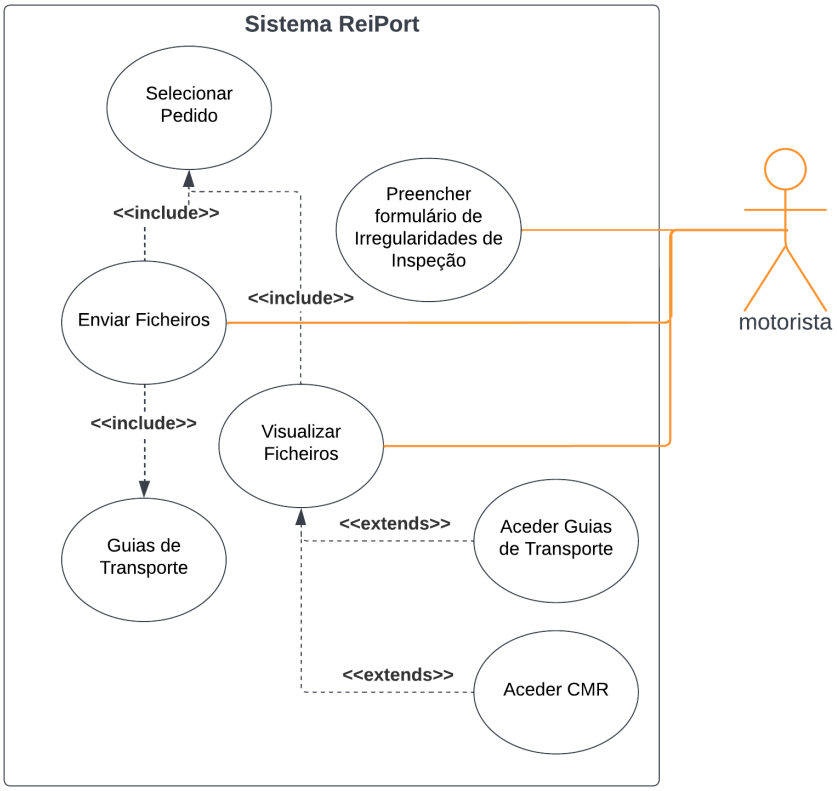


FIGURA 2 - CASOS DE USO - MOTORISTA

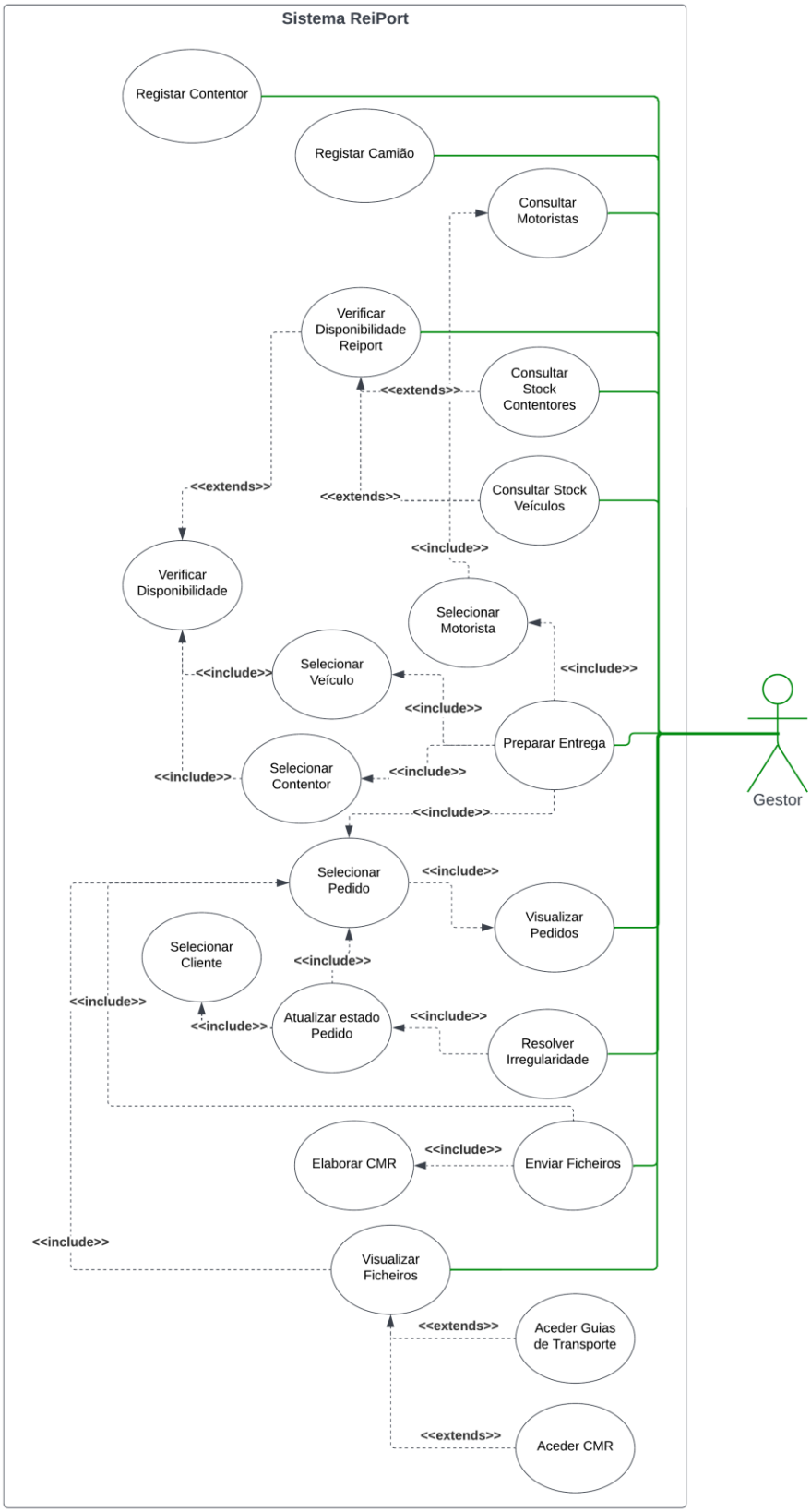


FIGURA 3 - CASO DE USO - GESTOR

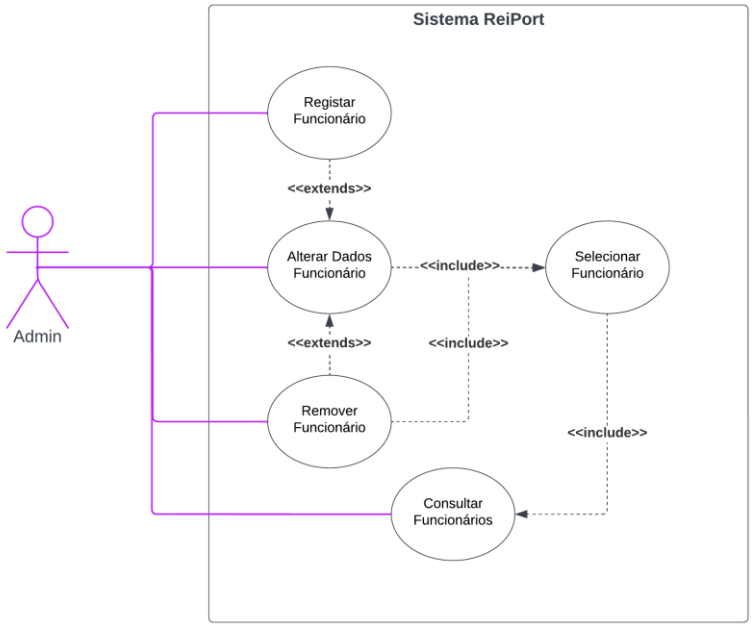


FIGURA 4 - CASO DE USO - ADMIN

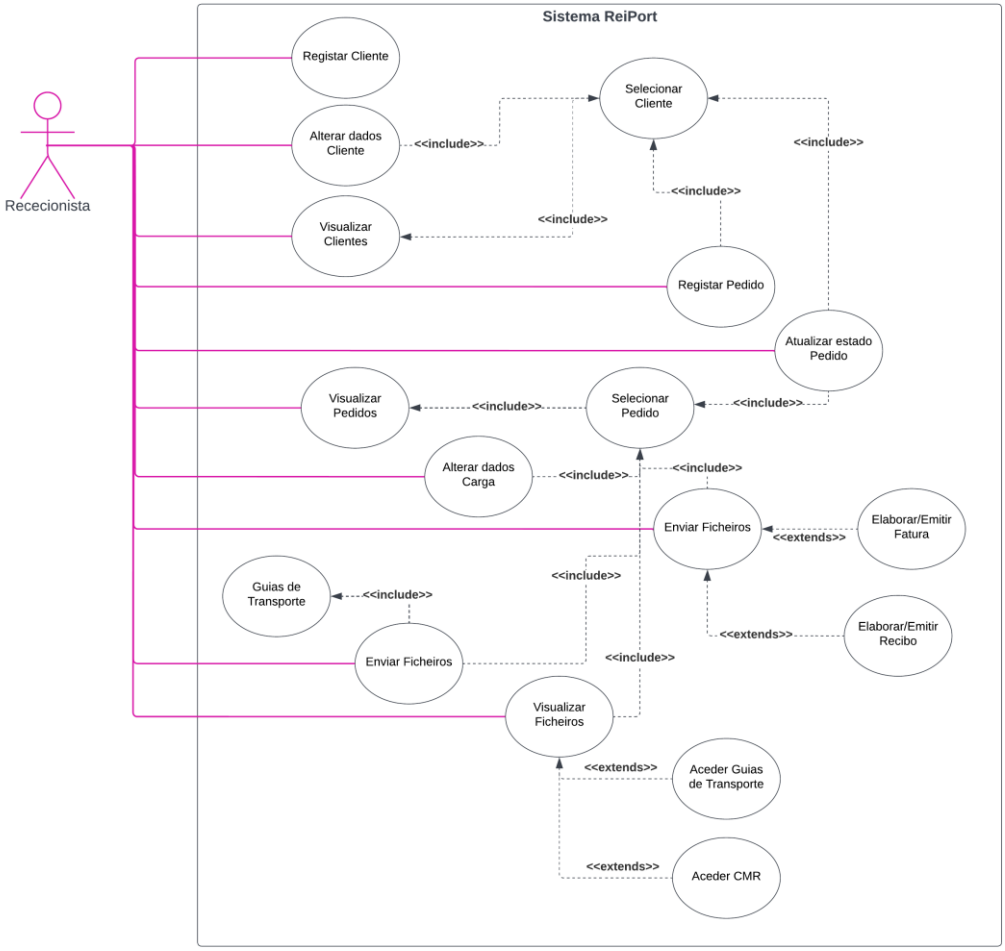


FIGURA 5 - CASO DE USO - RECECIONISTA

Templates dos Casos de Uso

Templates do Admin

Uso Caso 1:	Registar Funcionário
Ator Principal:	Admin
Requisitos	RF33
Pré-Requisitos:	Admin tem de estar registado no sistema
Pós-Requisitos:	É criado um funcionário no sistema
Cenário Principal:	1. Admin insere nome, data de nascimento, NIF, morada e telefone 2. Sistema apresenta dados e pede confirmação para criar Utilizador 3. Admin confirma 4. Sistema gera email e password 5. Sistema cria Utilizador
Cenários Alternativos:	1. Funcionário é Motorista a) Admin insere nome, data de nascimento, NIF, morada e telefone, número de cartão de cidadão, ADR e CAM 1. Funcionário é Motorista e não tem CAM a) Admin define campo CAM como Falso 1. Funcionário é Motorista e não tem ADR a) Admin define campo ADR como Falso
Exceções:	1. NIF inválido

Uso Caso 2:	Alterar Dados Funcionário
Ator Principal:	Admin
Requisitos	RF33
Pré-Requisitos:	Admin tem de estar registado no sistema
Pós-Requisitos:	Os dados do funcionário são alterados no sistema
Cenário Principal:	1. Admin pesquisa Funcionários 2. Sistema lista Funcionários por filtros definidos 3. Admin seleciona Funcionário 4. Admin altera os campos pretendidos 5. Sistema apresenta dados e pede confirmação para Alterar Dados de Funcionário 6. Admin confirma
Cenários Alternativos:	
Exceções:	3. Funcionário não existe 4. Dados por preencher 4. Dados inadmissíveis em campos

Uso Caso 3:	Remover Funcionário
Ator Principal:	Admin
Requisitos	RF33
Pré-Requisitos:	Admin tem de estar registado no sistema
Pós-Requisitos:	Os dados do funcionário são removidos do sistema
Cenário Principal:	1. Admin pesquisa funcionários 2. Sistema lista funcionários de acordo com filtros definidos 3. Admin seleciona Funcionário 4. Admin remove Funcionário 5. Sistema apresenta funcionário e pede confirmação para Remover 6. Admin confirma ação 7. Sistema remove Funcionário
Cenários Alternativos:	
Exceções:	1. Funcionário não existe

Uso Caso 4:	Consultar Funcionários
Ator Principal:	Admin
Requisitos	RF33
Pré-Requisitos:	Admin tem de estar autenticado
Pós-Requisitos:	Os funcionários são apresentados pelo sistema
Cenário Principal:	1. Admin pesquisa funcionários 2. Sistema apresenta funcionários por filtros definidos
Cenários Alternativos:	
Exceções:	

Templates do Motorista

Uso Caso 5:	Preencher Formulário de Irregularidades de Inspeção
Ator Principal:	Motorista
Requisitos	RF27
Pré-Requisitos:	Motorista tem de estar autenticado
Pós-Requisitos:	É submetido um formulário de irregularidades preenchido no sistema
Cenário Principal:	1. Motorista cria formulário de irregularidades 2. Motorista preenche formulário 3. Motorista submete 4. Sistema pede confirmação de submissão 5. Motorista confirma 6. Sistema armazena ficheiro
Cenários Alternativos:	
Exceções:	6. Ficheiro não enviado

Templates do Motorista, Gestor e Rececionista

Uso Caso 6:	Visualizar Ficheiros
Ator Principal:	Motorista, Gestor, Rececionista
Requisitos	RF09, RF12, RF23, RF25, RF26
Pré-Requisitos:	Utilizador tem de estar autenticado no sistema
Pós-Requisitos:	
Cenário Principal:	1. Utilizador pesquisa Pedidos 2. Sistema lista Pedidos por filtros definidos 3. Utilizador seleciona Pedido 4. Utilizador acede aos ficheiros relativos ao Pedido 5. Sistema apresenta ficheiros disponíveis 6. Utilizador seleciona ficheiro pretendido 7. Sistema apresenta ficheiro
Cenários Alternativos:	
Exceções:	1. Pedido não existe 6. Ficheiro não existe

Templates do Motorista e Rececionista

Uso Caso 7:	Enviar Ficheiros
Ator Principal:	Motorista, Rececionista
Requisitos	RF08, RF28
Pré-Requisitos:	Utilizador tem de estar autenticado
Pós-Requisitos:	Submete um ficheiro no sistema
Cenário Principal:	1. Utilizador pesquisa Pedidos 2. Sistema lista Pedidos por filtros definidos 3. Utilizador seleciona Pedido 4. Utilizador adiciona ficheiro ao Pedido 5. Sistema adiciona ficheiro 6. Sistema altera estado do pedido
Cenários Alternativos:	
Exceções:	3. Pedido não existe 5. Ficheiro não compatível 5. Ficheiro não enviado

Templates da Rececionista

Uso Caso 8:	Alterar Dados Cliente
Ator Principal:	Rececionista
Requisitos	RF03
Pré-Requisitos:	Rececionista tem de estar autenticado
Pós-Requisitos:	Os dados são alterados na conta do Cliente selecionada
Cenário Principal:	1. Rececionista pesquisa Clientes 2. Sistema lista Clientes por filtros definidos 3. Rececionista seleciona Cliente 4. Rececionista seleciona o parâmetro que pretende alterar e altera um a um 5. Sistema apresenta dados e pede confirmação para alterar dados do Cliente 6. Rececionista confirma
Cenários Alternativos:	
Exceções:	3. Cliente não existe

Uso Caso 9:	Visualizar Clientes
Ator Principal:	Rececionista
Requisitos	
Pré-Requisitos:	Rececionista tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Rececionista pesquisa Clientes 2. Sistema apresenta Clientes por filtros definidos
Cenários Alternativos:	
Exceções:	

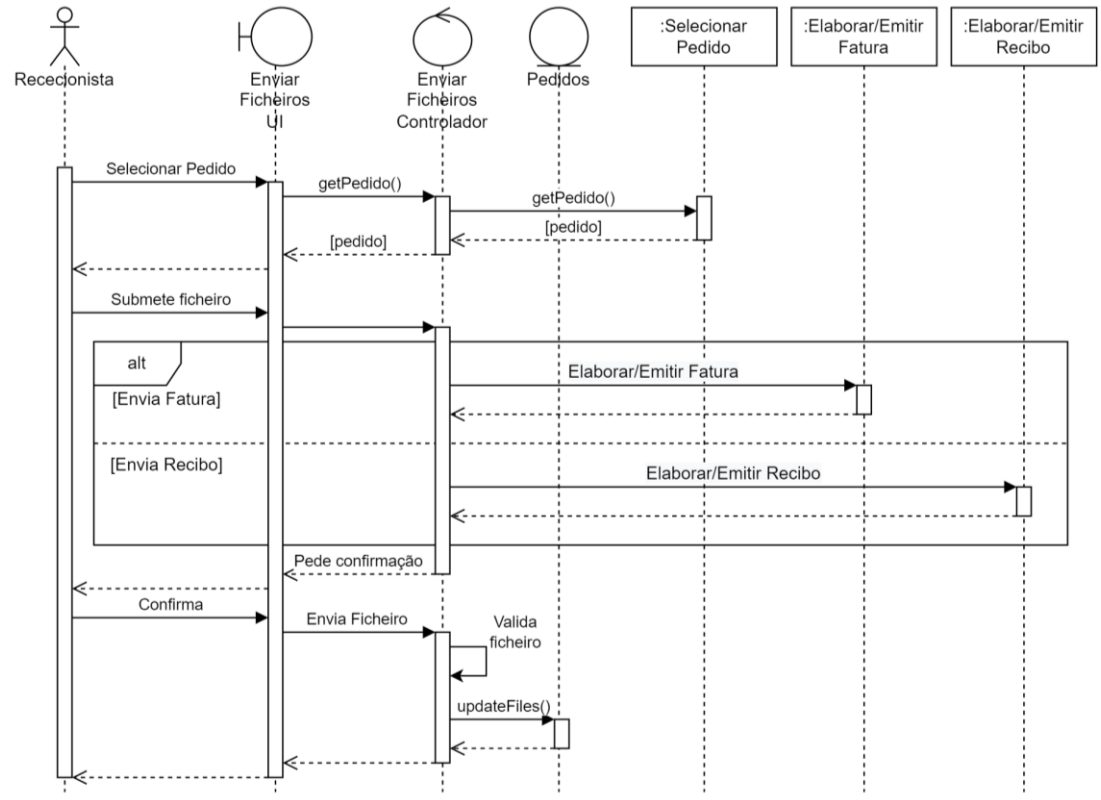
Uso Caso 10:	Registar Pedido
Ator Principal:	Rececionista
Requisitos	RF02
Pré-Requisitos:	Rececionista tem de estar autenticado
Pós-Requisitos:	Um Pedido é criado para um Cliente
Cenário Principal:	1. Rececionista pesquisa Clientes 2. Sistema lista Clientes por filtros definidos 3. Rececionista seleciona Cliente 4. Rececionista regista novo Pedido 5. Rececionista insere disponibilidade pelo Cliente de contentor e de veículo, peso da carga, data-limite, morada de origem de entrega e morada do destinatário no novo Pedido 6. Sistema apresenta dados e pede confirmação 7. Rececionista confirma
Cenários Alternativos:	
Exceções:	3. Cliente não existe

Uso Caso 11:	Visualizar Pedidos
Ator Principal:	Rececionista
Requisitos	RF04
Pré-Requisitos:	Rececionista tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Rececionista pesquisa Pedidos 2. Sistema apresenta Pedidos por filtros definidos
Cenários Alternativos:	
Exceções:	

Uso Caso 12:	Alterar Dados Pedido
Ator Principal:	Rececionista
Requisitos	RF03
Pré-Requisitos:	Rececionista tem de estar autenticado
Pós-Requisitos:	Os dados do pedido selecionado do Cliente são alterados
Cenário Principal:	1. Rececionista pesquisa Clientes 2. Sistema lista Clientes por filtros definidos 3. Rececionista seleciona Cliente 4. Rececionista pesquisa pedidos do Cliente 5. Sistema apresenta pedidos do Cliente 6. Rececionista seleciona pedido para efetuar alterações 7. Rececionista seleciona o parâmetro que pretende alterar e altera um a um 8. Sistema apresenta dados e pede confirmação 9. Rececionista confirma
Cenários Alternativos:	
Exceções:	3. Cliente não existe 6. Pedido não existe

Uso Caso 13:	Atualizar Estado do Pedido
Ator Principal:	Rececionista
Requisitos	RF06
Pré-Requisitos:	Rececionista tem de estar autenticado
Pós-Requisitos:	É alterado o estado de um Pedido associado a um Cliente
Cenário Principal:	1. Rececionista pesquisa Clientes 2. Sistema lista Cliente por filtros definidos 3. Rececionista seleciona Cliente 4. Rececionista pesquisa pedidos do Cliente 5. Sistema apresenta pedidos do Cliente 6. Rececionista seleciona pedido para efetuar alterações 7. Rececionista atualiza estado 8. Sistema apresenta dados e pede confirmação 9. Rececionista confirma
Cenários Alternativos:	
Exceções:	3. Cliente não existe 6. Pedido não existe

Uso Caso 14:	Enviar Ficheiros
Ator Principal:	Rececionista
Requisitos	RF08, RF10, RF11
Pré-Requisitos:	Rececionista tem de estar autenticado
Pós-Requisitos:	Submete um ficheiro no sistema
Cenário Principal:	1. Rececionista pesquisa Pedidos 2. Sistema lista Pedidos por filtros definidos 3. Rececionista seleciona Pedido 4. Rececionista Elabora Fatura/Recibo 5. Rececionista submete ficheiro 6. Sistema pede confirmação 7. Rececionista confirma 8. Sistema envia ficheiro
Cenários Alternativos:	
Exceções:	3. Pedido não existe 5. Ficheiro não compatível 8. Ficheiro não enviado



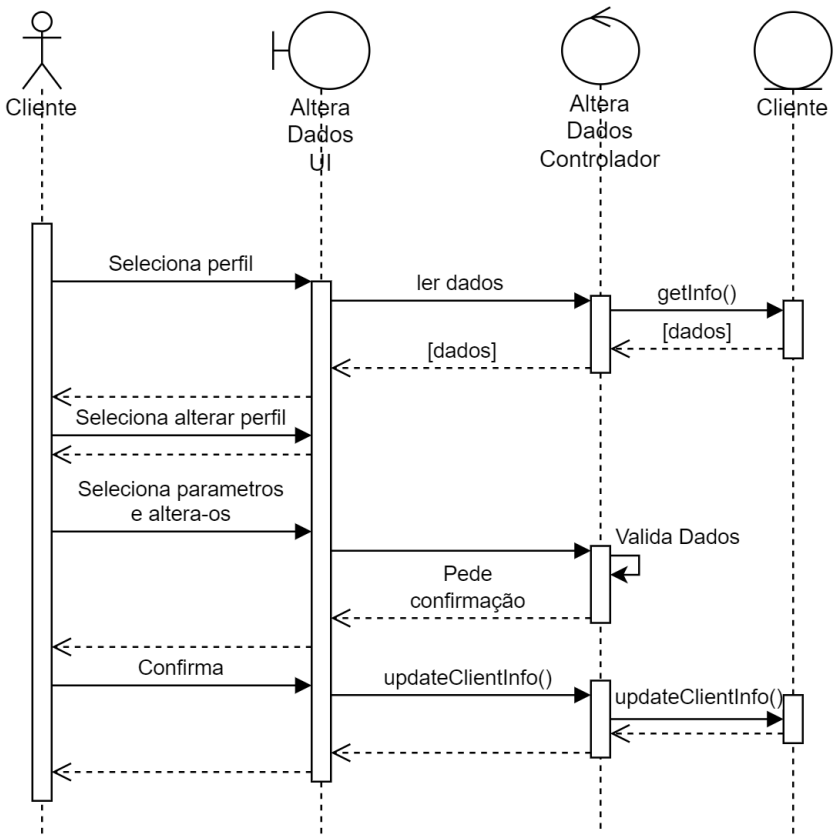
Templates da Rececionista e Cliente

Uso Caso 15:	Registar Cliente
Ator Principal:	Rececionista, Cliente
Requisitos	RF01, RF29
Pré-Requisitos:	Utilizador tem de estar autenticado
Pós-Requisitos:	Uma conta de Cliente é criada no sistema
Cenário Principal:	1. Utilizador insere nome próprio, apelido, data de nascimento, NIF, morada, email, password e telefone 2. Sistema apresenta dados e pede confirmação para criar Cliente 3. Utilizador confirma 4. Sistema regista Cliente
Cenários Alternativos:	
Exceções:	1. NIF inválido

Templates do Cliente

Uso Caso 16:	Visualizar Pedidos
Ator Principal:	Cliente
Requisitos	RF30
Pré-Requisitos:	Cliente tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Cliente pesquisa Pedidos 2. Sistema apresenta Pedidos
Cenários Alternativos:	
Exceções:	

Uso Caso 17:	Alterar Dados Cliente
Ator Principal:	Cliente
Requisitos	RF31
Pré-Requisitos:	Cliente tem de estar autenticado
Pós-Requisitos:	Os dados são alterados na conta do Cliente
Cenário Principal:	1. Cliente seleciona e editar perfil 2. Cliente seleciona o parâmetro que pretende alterar e altera um a um 3. Sistema apresenta dados e pede confirmação para alterar dados do Cliente 4. Cliente confirma 5. Sistema atualiza os dados do Cliente
Cenários Alternativos:	
Exceções:	1. NIF inválido



Templates do Gestor

Uso Caso 18:	Registar Contentor
Ator Principal:	Gestor
Requisitos	RF32
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	Um contentor é registado no sistema
Cenário Principal:	1. Gestor insere largura, comprimento, profundidade, cor, peso máximo suportado, marca, modelo e matrícula 2. Sistema apresenta dados e pede confirmação para criar Contentor 3. Gestor confirma 4. Sistema regista contentor
Cenários Alternativos:	
Exceções:	1. Matrícula inválida

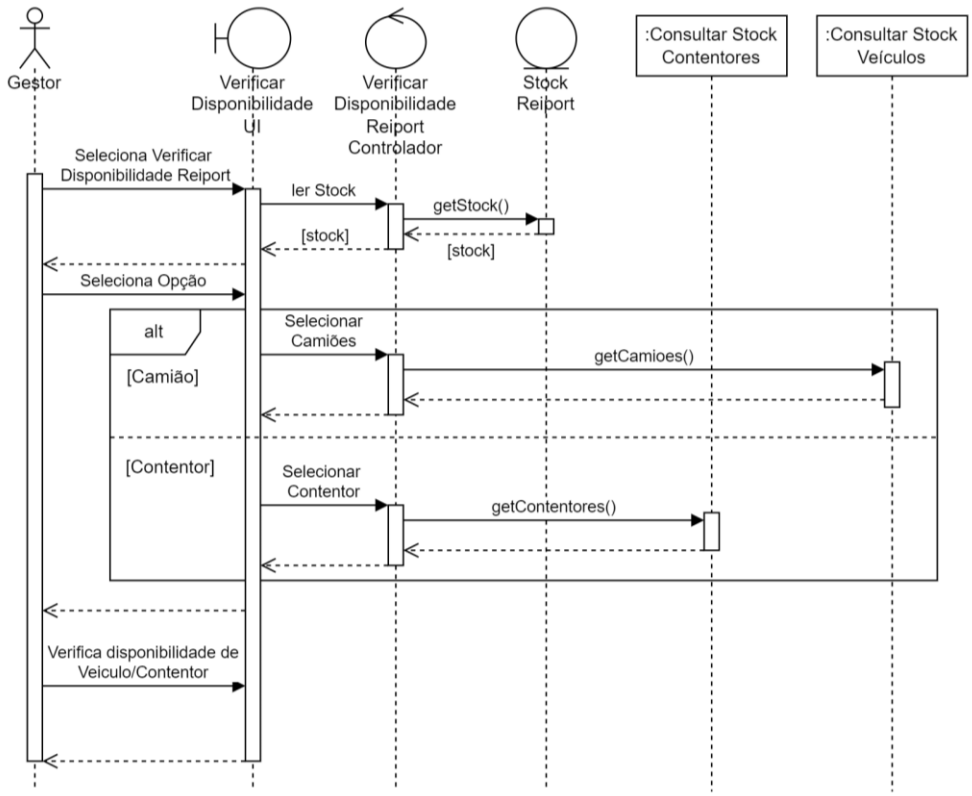
Uso Caso 19:	Registar Camião
Ator Principal:	Gestor
Requisitos	RF32
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	Um camião é registado no sistema
Cenário Principal:	1. Gestor insere potência, cilindrada, tanque, cor, peso máximo suportado, marca, modelo e matrícula 2. Sistema apresenta dados e pede confirmação para criar Camião 3. Gestor confirma 4. Sistema regista camião
Cenários Alternativos:	
Exceções:	1. Matrícula inválida

Uso Caso 20:	Consultar Stock Contentores
Ator Principal:	Gestor
Requisitos	RF14, RF19
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Gestor pesquisa Contentores 2. Sistema apresenta Contentores
Cenários Alternativos:	
Exceções:	

Uso Caso 21:	Consultar Stock Camiões
Ator Principal:	Gestor
Requisitos	RF14, RF18
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Gestor pesquisa Camiões 2. Sistema apresenta Camiões
Cenários Alternativos:	
Exceções:	

Uso Caso 22:	Consultar Motoristas
Ator Principal:	Gestor
Requisitos	RF15, RF20, RF21
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Gestor pesquisa Motoristas 2. Sistema apresenta Motoristas
Cenários Alternativos:	
Exceções:	

Uso Caso 23:	Verificar Disponibilidade Reiport
Ator Principal:	Gestor
Requisitos	RF17, RF18, RF19
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Gestor pesquisa Stock Reiport 2. Gestor seleciona Camião 3. Sistema apresenta Stock de Camiões Reiport 4. Gestor verifica se camião se encontra disponível para a entrega do pedido
Cenários Alternativos:	2. Selecionar Contentor a. Sistema apresenta Stock de Contentores Reiport b. Gestor verifica se contentor se encontra disponível para a entrega do pedido
Exceções:	1. Contentor ou Camião com estas especificações não existe 2. Contentor ou Camião com estas especificações está a ser utilizado



Uso Caso 24:	Visualizar Pedidos
Ator Principal:	Gestor
Requisitos	RF13
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Gestor pesquisa Pedidos 2. Sistema apresenta Pedidos
Cenários Alternativos:	
Exceções:	

Uso Caso 25:	Resolver Irregularidades
Ator Principal:	Gestor
Requisitos	RF24
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Gestor pesquisa Pedidos 2. Sistema lista Pedidos por filtros definidos 3. Gestor seleciona Pedido 4. Gestor resolve problema 5. Gestor altera estado do pedido
Cenários Alternativos:	
Exceções:	1. Pedido não existe

Uso Caso 26:	Preparar Pedido
Ator Principal:	Gestor
Requisitos	RF13, RF15, RF16, RF17, RF18, RF19, RF20, RF21, RF22, RF25
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	Pedido é preparado
Cenário Principal:	1. Gestor pesquisa Pedidos 2. Sistema lista Pedidos 3. Gestor seleciona Pedido 4. Gestor verifica disponibilidade de camião pelo Cliente 5. Gestor inicia "Registar Camião" 6. Gestor verifica disponibilidade de contentor ou contentores pelo Cliente 7. Gestor inicia "Registar Contentor" 8. Gestor pesquisa motoristas com habilitações necessárias 9. Gestor verifica tempo estimado da viagem 10. Gestor marca uma data de início de entrega 11. Gestor pesquisa e seleciona 1 motorista com disponibilidade 12. Atribui preço à entrega 13. Sistema apresenta dados para confirmação 14. Gestor confirma 15. Gestor inicia "Enviar Ficheiro"
Cenários Alternativos:	4. Camião do Cliente não existe a) Gestor verifica especificações de camião necessário b) Gestor inicia "Verificar Disponibilidade Reiport" c) Gestor seleciona camião da Empresa 6. Contentor do Cliente não existe a) Gestor verifica especificações de contentor necessário b) Gestor inicia "Verificar Disponibilidade Reiport" c) Gestor seleciona contentor da Empresa 6. Cliente disponibiliza um de dois contentores necessário a) Gestor verifica especificações de contentor necessário b) Gestor inicia "Verificar Disponibilidade Reiport" c) Gestor seleciona contentor do Stock da Reiport 6. Cliente necessita de dois contentores a) Gestor verifica especificações de contentores necessários b) Gestor inicia "Verificar Disponibilidade Reiport" c) Gestor seleciona contentores do Stock Reiport 9. Tempo Viagem maior que 9 horas a) Gestor seleciona 2 motoristas

Exceções:	3. Não existe Pedido 4. Não existe Camião, quer Cliente quer de Stock Reiport 6. Não existe Contentor, quer Cliente quer de Stock Reiport 11. Não existe Motorista

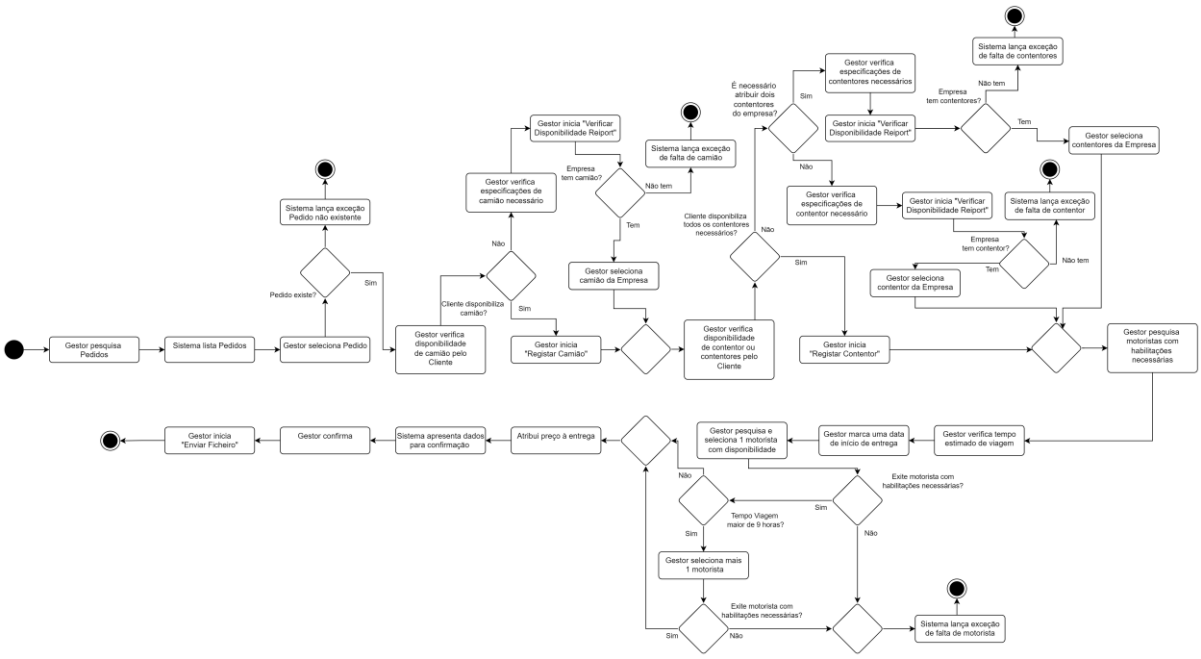


FIGURA 6 - DIAGRAMA DE ATIVIDADES DO RF26

Uso Caso 27:	Enviar Ficheiros
Ator Principal:	Gestor
Requisitos	RF23
Pré-Requisitos:	Gestor tem de estar autenticado
Pós-Requisitos:	
Cenário Principal:	1. Gestor pesquisa Pedidos 2. Sistema lista Pedidos 3. Gestor seleciona Pedido 4. Gestor Elabora CMR 5. Gestor submete ficheiro 6. Sistema pede confirmação 7. Gestor confirma
Cenários Alternativos:	
Exceções:	5. Ficheiro não compatível 7. Ficheiro não enviado

4.2 - Modelação de dados

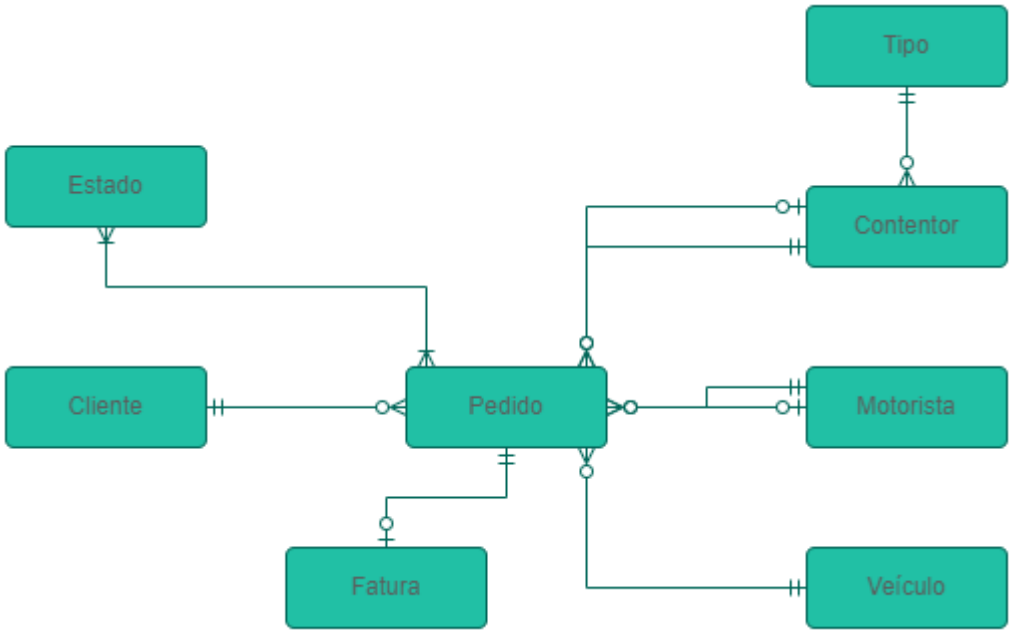


FIGURA 7 - DIAGRAMA DE ENTIDADES E RELACIONAMENTOS (DER) (1ª ABORDAGEM)

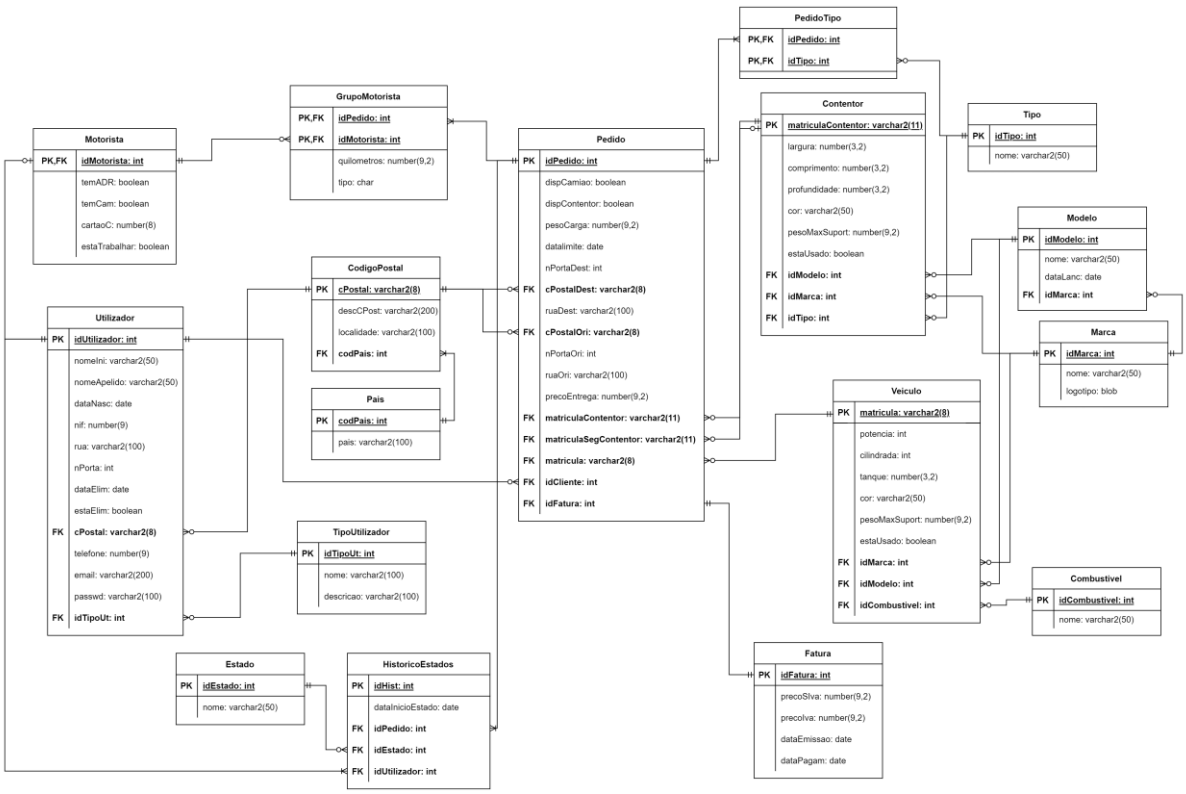


FIGURA 8 - MODELO RELACIONAL

4.3 - Modelo de classes

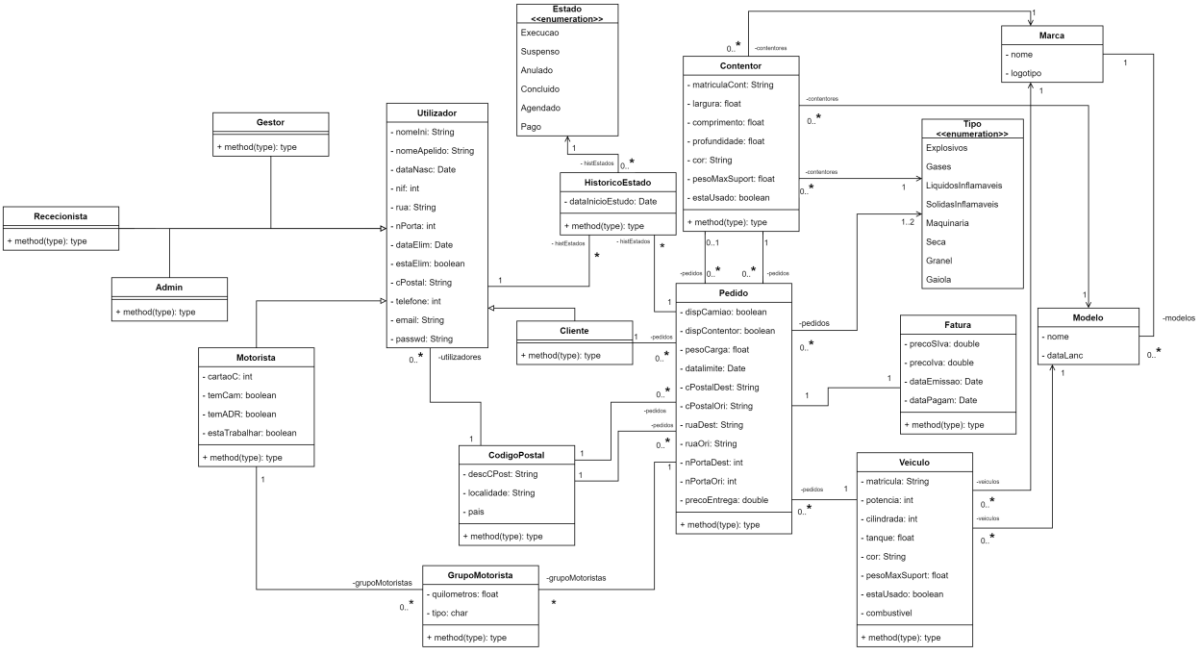


FIGURA 9 - MODELO DE CLASSES

4.4 - Modelo de domínio

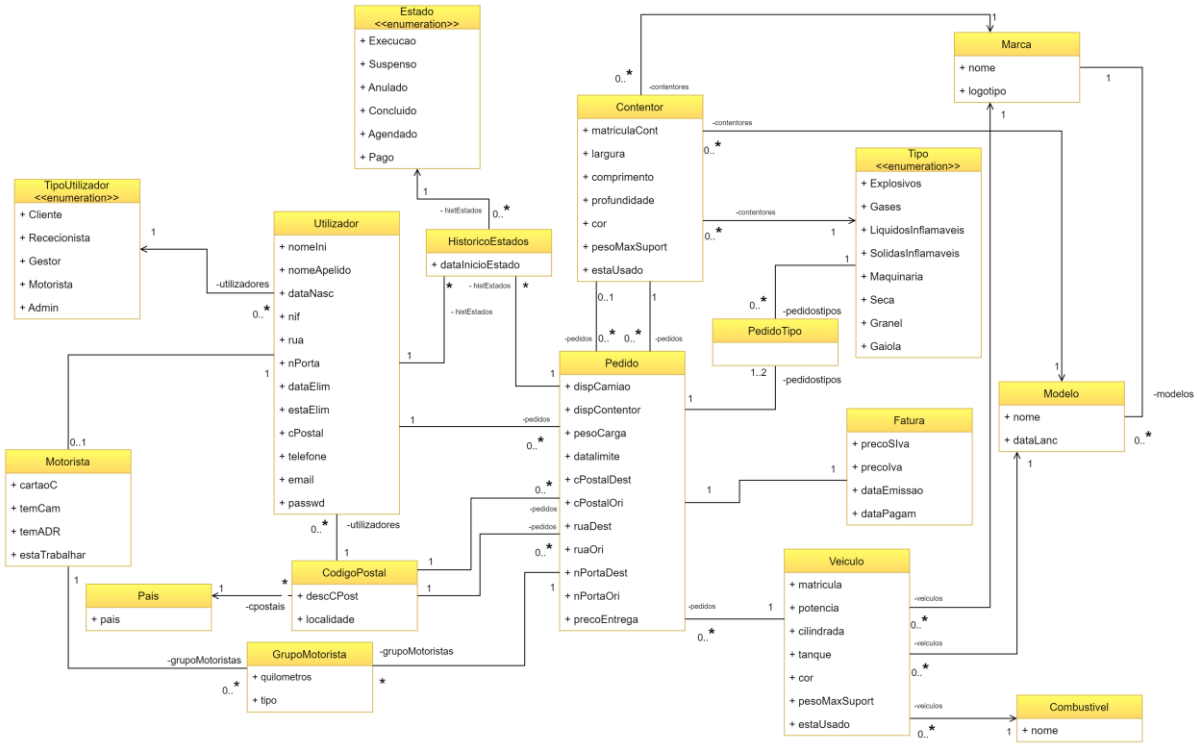


FIGURA 10 - MODELO DE DOMÍNIO

4.5 - Diagrama de transição de estados do Pedido

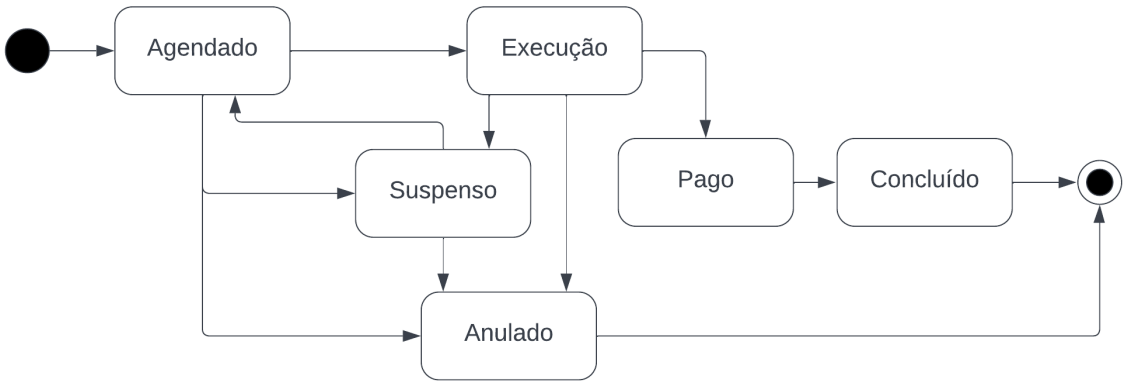
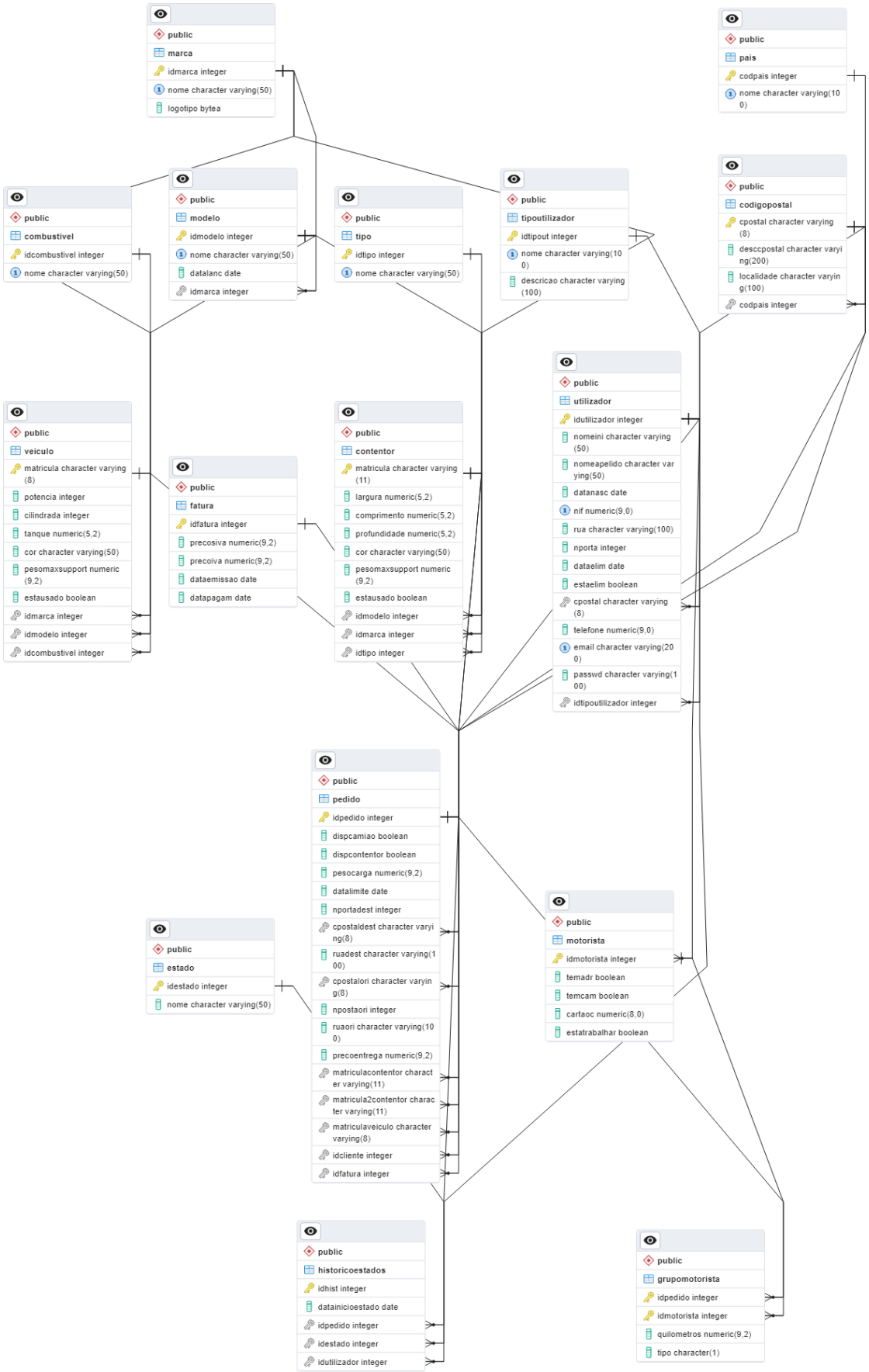


FIGURA 11 - DIAGRAMA DE TRANSIÇÃO DE ESTADOS

5. Implementação da BD



5.1 - Criação das tabelas

```
CREATE TABLE Pais(
   codPais SERIAL NOT NULL,
   pais VARCHAR(100) NOT NULL UNIQUE,
   PRIMARY KEY (codPais)
);

CREATE TABLE CodigoPostal(
   cPostal VARCHAR(8) NOT NULL,
   descCPostal VARCHAR(200) DEFAULT NULL,
   localidade VARCHAR(100) NOT NULL,
   codPais INT NOT NULL,
   PRIMARY KEY (cPostal),
   CONSTRAINT codPais_fk1
      FOREIGN KEY (codPais)
         REFERENCES Pais (codPais)
);

CREATE TABLE TipoUtilizador(
   idTipoUt SERIAL NOT NULL,
   nome VARCHAR(100) NOT NULL UNIQUE,
   descricao VARCHAR(100) DEFAULT NULL,
   PRIMARY KEY (idTipoUt)
);
```

```
CREATE TABLE Utilizador(
   idUtilizador SERIAL NOT NULL,
   nomeIni VARCHAR(50) NOT NULL,
   nomeApelido VARCHAR(50) NOT NULL,
   dataNasc DATE NOT NULL,
   nif NUMERIC(9) NOT NULL UNIQUE,
   rua VARCHAR(100) NOT NULL,
   nPorta INT NOT NULL,
   dataElim DATE DEFAULT NULL,
   estaElim BOOLEAN NOT NULL DEFAULT FALSE,
   cPostal VARCHAR(8) NOT NULL,
   telefone NUMERIC(9) NOT NULL,
   email VARCHAR(200) NOT NULL UNIQUE,
   passwd VARCHAR(100) NOT NULL,
   idTipoUt INT NOT NULL,
   PRIMARY KEY (idUtilizador),
   CONSTRAINT codPostal_fk1
      FOREIGN KEY (cPostal)
         REFERENCES CodigoPostal (cPostal),
   CONSTRAINT idTipoUt_fk1
      FOREIGN KEY (idTipoUt)
         REFERENCES TipoUtilizador (idTipoUt)
);

CREATE TABLE Motorista(
   idMotorista INT NOT NULL,
   temADR BOOLEAN NOT NULL DEFAULT FALSE,
   temCam BOOLEAN NOT NULL DEFAULT FALSE,
   cartaoC NUMERIC(8) NOT NULL,
   estaTrabalhar BOOLEAN NOT NULL DEFAULT FALSE,
   PRIMARY KEY (idMotorista),
   CONSTRAINT idMotorista_fk1
      FOREIGN KEY (idMotorista)
         REFERENCES Utilizador (idUtilizador)
);

CREATE TABLE Fatura(
   idFatura SERIAL NOT NULL,
   precoSIva NUMERIC(9,2) NOT NULL CHECK(precoSiva > 0),
   precoIva NUMERIC(9,2) NOT NULL CHECK(precoSiva > 0),
   dataEmissao DATE NOT NULl DEFAULT now(),
   dataPagam DATE DEFAULT NULL,
   PRIMARY KEY (idFatura)
);
```

```
CREATE TABLE Combustivel(
   idCombustivel SERIAL NOT NULL,
   nome VARCHAR(50) NOT NULL UNIQUE,
   PRIMARY KEY (idCombustivel)
);

CREATE TABLE Marca(
   idMarca SERIAL NOT NULL,
   nome VARCHAR(50) NOT NULL UNIQUE,
   logotipo bytea DEFAULT NULL,
   PRIMARY KEY (idMarca)
);

CREATE TABLE Modelo(
   idModelo SERIAL NOT NULL,
   nome VARCHAR(50) NOT NULL UNIQUE,
   dataLanc DATE NOT NULL,
   idMarca INT NOT NULL,
   PRIMARY KEY (idModelo),
   CONSTRAINT idMarca_fk1
      FOREIGN KEY (idMarca)
         REFERENCES Marca (idMarca)
);

CREATE TABLE Tipo(
   idTipo SERIAL NOT NULL,
   nome VARCHAR(50) NOT NULL UNIQUE,
   PRIMARY KEY (idTipo)
);
```

```
CREATE TABLE Contentor(
   matriculaContentor VARCHAR(11) NOT NULL,
   largura NUMERIC(5,2) NOT NULL CHECK(largura > 0),
   comprimento NUMERIC(5,2) NOT NULL CHECK(comprimento > 0),
   profundidade NUMERIC(5,2) NOT NULL CHECK(profundidade > 0),
   cor VARCHAR(50) NOT NULL,
   pesoMaxSuport NUMERIC(9,2) NOT NULL CHECK(pesoMaxSuport > 0),
   estaUsado BOOLEAN NOT NULL DEFAULT FALSE,
   idModelo INT NOT NULL,
   idMarca INT NOT NULL,
   idTipo INT NOT NULL,
   PRIMARY KEY (matriculaContentor),
   CONSTRAINT idModelo_fk1
      FOREIGN KEY (idModelo)
         REFERENCES Modelo (idModelo),
   CONSTRAINT idMarca_fk2
      FOREIGN KEY (idMarca)
         REFERENCES Marca (idMarca),
   CONSTRAINT idTipo_fk3
      FOREIGN KEY (idTipo)
         REFERENCES Tipo (idTipo)
);

CREATE TABLE Veiculo(
   matricula VARCHAR(8) NOT NULL,
   potencia INT NOT NULL CHECK(potencia > 0),
   cilindrada INT NOT NULL CHECK(cilindrada > 0),
   tanque NUMERIC(5,2) NOT NULL CHECK(tanque > 0),
   cor VARCHAR(50) NOT NULL,
   pesoMaxSuport NUMERIC(9,2) NOT NULL CHECK(pesoMaxSuport > 0),
   estaUsado BOOLEAN NOT NULL DEFAULT FALSE,
   idMarca INT NOT NULL,
   idModelo INT NOT NULL,
   idCombustivel INT NOT NULL,
   PRIMARY KEY (matricula),
   CONSTRAINT idMarca_fk1
      FOREIGN KEY (idMarca)
         REFERENCES Marca (idMarca),
   CONSTRAINT idModelo_fk2
      FOREIGN KEY (idModelo)
         REFERENCES Modelo (idModelo),
   CONSTRAINT idCombustivel_fk3
      FOREIGN KEY (idCombustivel)
         REFERENCES Combustivel (idCombustivel)
);
```

```
CREATE TABLE Pedido(
   idPedido SERIAL NOT NULL,
   dispCamiao BOOLEAN NOT NULL DEFAULT FALSE,
   dispContentor BOOLEAN NOT NULL DEFAULT FALSE,
   pesoCarga NUMERIC(9,2) NOT NULL CHECK(pesoCarga > 0),
   dataLimite DATE NOT NULL,
   nPortaDest INT NOT NULL,
   cPostalDest VARCHAR(8) NOT NULL,
   ruaDest VARCHAR(100) NOT NULL,
   cPostalOri VARCHAR(8) NOT NULL,
   nPortaOri INT NOT NULL,
   ruaOri VARCHAR(100) NOT NULL,
   precoEntrega NUMERIC(9,2) NOT NULL CHECK(precoEntrega > 0),
   matriculaContentor VARCHAR(11) NOT NULL,
   matriculaSegContentor VARCHAR(11) DEFAULT NULL,
   matricula VARCHAR(8) NOT NULL,
   idCliente INT NOT NULL,
   idFatura INT NOT NULL,
   PRIMARY KEY (idPedido),
   CONSTRAINT cPostalDest_fk1
        FOREIGN KEY (cPostalDest)
          REFERENCES CodigoPostal (cPostal),
   CONSTRAINT cPostalOri_fk2
        FOREIGN KEY (cPostalOri)
          REFERENCES CodigoPostal (cPostal),
   CONSTRAINT matriculaContentor_fk3
        FOREIGN KEY (matriculaContentor)
          REFERENCES Contentor (matriculaContentor),
   CONSTRAINT matricula2Contentor_fk4
        FOREIGN KEY (matriculaSegContentor)
          REFERENCES Contentor (matriculaContentor),
   CONSTRAINT matriculaVeiculo_fk5
        FOREIGN KEY (matricula)
          REFERENCES Veiculo (matricula),
   CONSTRAINT idCliente_fk6
        FOREIGN KEY (idCliente)
          REFERENCES Utilizador (idUtilizador),
   CONSTRAINT idFatura_fk7
        FOREIGN KEY (idFatura)
          REFERENCES Fatura (idFatura)
);
```

```
CREATE TABLE GrupoMotorista(
   idPedido INT NOT NULL,
   idMotorista INT NOT NULL,
   quilometros NUMERIC(9,2) NOT NULL CHECK(quilometros > 0),
   tipo CHAR NOT NULL DEFAULT 'p',
   PRIMARY KEY (idPedido, idMotorista),
   CONSTRAINT idPedido_fk1
         FOREIGN KEY (idPedido)
            REFERENCES Pedido (idPedido),
   CONSTRAINT idMotorista_fk2
         FOREIGN KEY (idMotorista)
            REFERENCES Motorista (idMotorista)
);

CREATE TABLE Estado(
   idEstado SERIAL NOT NULL,
   nome VARCHAR(50) NOT NULL,
   PRIMARY KEY (idEstado)
);

CREATE TABLE HistoricoEstados(
   idHist SERIAL NOT NULL,
   dataInicioEstado DATE NOT NULL DEFAULT now(),
   idPedido INT NOT NULL,
   idEstado INT NOT NULL,
   idUtilizador INT NOT NULL,
   PRIMARY KEY (idHist),
   CONSTRAINT idPedido_fk1
         FOREIGN KEY (idPedido)
            REFERENCES Pedido (idPedido),
   CONSTRAINT idEstado_fk2
         FOREIGN KEY (idEstado)
            REFERENCES Estado (idEstado),
   CONSTRAINT idUtilizador_fk3
         FOREIGN KEY (idUtilizador)
            REFERENCES Utilizador (idUtilizador)
);
```

5.2- Gestão dos dados na base de dados

5.2.1 - Inserção de dados

```
INSERT INTO pais (pais) VALUES ('Portugal');
INSERT INTO pais (pais) VALUES ('Austria');
INSERT INTO pais (pais) VALUES ('Belgica');
INSERT INTO pais (pais) VALUES ('Bulgaria');
INSERT INTO pais (pais) VALUES ('Chequia');
INSERT INTO pais (pais) VALUES ('Chipre');
INSERT INTO pais (pais) VALUES ('Croacia');
INSERT INTO pais (pais) VALUES ('Dinamarca');
INSERT INTO pais (pais) VALUES ('Eslovaquia');
INSERT INTO pais (pais) VALUES ('Eslovenia');
INSERT INTO pais (pais) VALUES ('Espanha');
INSERT INTO pais (pais) VALUES ('Estonia');
INSERT INTO pais (pais) VALUES ('Finlandia');
INSERT INTO pais (pais) VALUES ('Franca');
INSERT INTO pais (pais) VALUES ('Grecia');
INSERT INTO pais (pais) VALUES ('Hungria');
INSERT INTO pais (pais) VALUES ('Irlanda');
INSERT INTO pais (pais) VALUES ('Italia');
INSERT INTO pais (pais) VALUES ('Letonia');
INSERT INTO pais (pais) VALUES ('Lituania');
INSERT INTO pais (pais) VALUES ('Luxemburgo');
INSERT INTO pais (pais) VALUES ('Malta');
INSERT INTO pais (pais) VALUES ('Paises Baixos');
INSERT INTO pais (pais) VALUES ('Polonia');
INSERT INTO pais (pais) VALUES ('Alemanha');
INSERT INTO pais (pais) VALUES ('Romenia');
INSERT INTO pais (pais) VALUES ('Suecia');

INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4490-666', 'Povoa de Varzim', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('1000-139', 'Lisboa', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('3360-139', 'Cantanhede', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('2205-025', 'Abrantes', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4705-480', 'Braga', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4905-067', 'Barcelos', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('3040-474', 'Coimbra', 1);
```

```
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('3360-032', 'Penacova', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('3420-177', 'Tabua', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4200-014', 'Porto', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4475-045', 'Maia', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4795-894', 'Santo Tirso', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4480-330', 'Vila do Conde', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4900-281', 'Viana do Castelo', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4940-027', 'Paredes de Coura', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('2645-539', 'Cascais', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('3520-039', 'Nelas', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('7150-123', 'Borba', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('2610-181', 'Amadora', 1);
INSERT INTO codigopostal (cPostal, localidade, codPais) VALUES ('4490-251', 'Povoa de Varzim', 1);

INSERT INTO tipoutilizador (nome) VALUES ('Cliente');
INSERT INTO tipoutilizador (nome) VALUES ('Rececionista');
INSERT INTO tipoutilizador (nome) VALUES ('Gestor');
INSERT INTO tipoutilizador (nome) VALUES ('Motorista');
INSERT INTO tipoutilizador (nome) VALUES ('Admin');

INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, dataElim, estaElim,
cPostal, telefone, email, passwd, idTipoUt) VALUES
('Francisco', 'La Ventura', '1997-10-02', 940196036, 'Rua Ventura', 107, null, false, '4200-014',
914794541, 'franciscoventure@gmail.com', 'portomaior2', 1);
INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, dataElim, estaElim,
cPostal, telefone, email, passwd, idTipoUt) VALUES
('Ricardo', 'Machado', '1991-06-15', 323316608, 'Rua Dr. Andre Tomas', 7, null, false, '3520-039',
961123321, 'ricardinho@gmail.com', 'ric4rde25', 4);
INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, dataElim, estaElim,
cPostal, telefone, email, passwd, idTipoUt) VALUES
('Joao', 'Oliveira', '1968-08-14', 222837141, 'Avenida Nova Esperanca', 17, null, false, '4490-251',
931457891, 'oliveira@gmail.com', 'jonas?!100', 4);
INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, dataElim, estaElim,
cPostal, telefone, email, passwd, idTipoUt) VALUES
('Sofia', 'Ferreira', '1990-06-27', 851642243, 'Rua Diogo Cao', 5, null, false, '4475-045', 914789123,
'sofiaferreira@gmail.com', 'sofiferr!?191', 3);
INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, dataElim, estaElim,
cPostal, telefone, email, passwd, idTipoUt) VALUES
('Gabriel', 'Machado', '1972-07-20', 433492058, 'Rua da flor dourada', 45, null, false, '4900-281',
965741963, 'gabimachado@gmail.com', 'gabrielmachado101', 2);
```

INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, dataElim, estaElim, cPostal, telefone, email, passwd, idTipoUt) VALUES

('Rui', 'Alexandre', '1977-04-04', 243497058, 'Rua da alavanca', 106, null, false, '4900-281', 914651278, 'ruialex@gmail.com', 'supersecurepassword', 5);

INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, dataElim, estaElim, cPostal, telefone, email, passwd, idTipoUt) VALUES

('Joaquim', 'Silva', '2000-11-20', 123462731, 'Rua da Esperanca', 40, null, false, '4905-067', 934756280, 'joaquim@gmail.com', '?bl4sfemiafoicontada!#', 1);

INSERT INTO motorista (idmotorista, temADR, temCam, cartaoC, estaTrabalhar) VALUES (2, false, false, 78912345, false);

INSERT INTO motorista (idmotorista, temADR, temCam, cartaoC, estaTrabalhar) VALUES (3, true, true, 45612378, false);

INSERT INTO fatura(precoSIva, precoIva, dataPagam) VALUES (16260.16, 20000, null);

INSERT INTO fatura(precoSIva, precoIva, dataPagam) VALUES (32520.32, 40000, null);

INSERT INTO fatura(precoSIva, precoIva, dataPagam) VALUES (12195.12, 15000, null);

INSERT INTO fatura(precoSIva, precoIva, dataPagam) VALUES (37009.13, 45521.23, null);

INSERT INTO fatura(precoSIva, precoIva, dataPagam) VALUES (73170.73, 90000, null);

INSERT INTO combustivel (nome) VALUES ('Diesel');

INSERT INTO combustivel (nome) VALUES ('Gasolina');

INSERT INTO combustivel (nome) VALUES ('Eletrico');

INSERT INTO combustivel (nome) VALUES ('Gas');

INSERT INTO marca (nome) VALUES ('Iveco');

INSERT INTO marca (nome) VALUES ('Scania');

INSERT INTO marca (nome) VALUES ('Man');

INSERT INTO marca (nome) VALUES ('Volvo');

INSERT INTO marca (nome) VALUES ('Renault');

INSERT INTO marca (nome) VALUES ('Ford');

INSERT INTO marca (nome) VALUES ('Mercedes');

INSERT INTO marca (nome) VALUES ('EduCargas');

INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Actros L', '2021-07-01', 7);

INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Actros F', '2021-01-01', 7);

INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Actros', '1996-05-21', 7);

INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('S-WAY', '2022-11-26', 1);

INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('S-WAY Gas', '2022-11-30', 1);

```
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('V8', '2010-04-12', 2);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Linha S', '2014-07-25', 2);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Linha R', '2005-08-15', 2);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('TGX', '2022-03-11', 3);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('TGS', '2022-03-11', 3);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('TGM', '2022-03-11', 3);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Volvo FH', '1993-03-15', 4);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Volvo FM', '1998-08-11', 4);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('T High', '2013-06-13', 5);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('F-Max', '2018-04-17', 6);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Standard', '2000-04-17', 8);
INSERT INTO modelo (nome, dataLanc, idMarca) VALUES ('Open Top', '2000-04-17', 8);

INSERT INTO tipo (nome) VALUES ('Explosivo');
INSERT INTO tipo (nome) VALUES ('Gases');
INSERT INTO tipo (nome) VALUES ('LiquidosInflamaveis');
INSERT INTO tipo (nome) VALUES ('SolidasInflamaveis');
INSERT INTO tipo (nome) VALUES ('Maquinaria');
INSERT INTO tipo (nome) VALUES ('Seca');
INSERT INTO tipo (nome) VALUES ('Granel');
INSERT INTO tipo (nome) VALUES ('Gaiola');

INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor,
pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES
('XA-43-21', 2.35, 5.90, 33.20, 'Preto', 21.77, false, 16, 8, 7);
INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor,
pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES
('RK-ST-FG', 2.35, 12.04, 67.70, 'Amarelo', 26.78, true, 16, 8, 7);
INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor,
pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES
('YU-EF-AS', 2.35, 5.90, 33.20, 'Branco', 21.77, false, 16, 8, 7);
INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor,
pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES
('QA-CV-ZA', 2.35, 12.04, 67.70, 'Castanho', 26.78, true, 16, 8, 7);
INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor,
pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES
('PO-LK-JN', 2.31, 5.89, 32.23, 'Preto', 21.60, false, 17, 8, 6);
INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor,
pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES
('AF-GH-ZC', 2.35,  12.02, 65.50, 'Branco', 26.63, false, 17, 8, 6);
```

INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor, pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES

('12-FG-AB', 2.31, 5.89, 32.23, 'Castanho', 21.60, false, 17, 8, 6);

INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor, pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES

('19-FG-24', 2.35, 12.02, 65.50, 'Verde', 26.63, false, 17, 8, 6);

INSERT INTO contentor (matriculaContentor, largura, comprimento, profundidade, cor, pesoMaxSuport, estaUsado, idModelo, idMarca, idTipo) VALUES

('FT-ZX-12', 2.44, 6.06, 2.75, 'Verde', 24, false, 17, 8, 5);

INSERT INTO veiculo (matricula, potencia, cilindrada, tanque, cor, pesoMaxSuport, estaUsado, idMarca, idModelo, idCombustivel) VALUES

('QP-HP-12', 2600, 10.4, 720, 'Amarelo', 45, true, 7, 1, 1);

INSERT INTO veiculo (matricula, potencia, cilindrada, tanque, cor, pesoMaxSuport, estaUsado, idMarca, idModelo, idCombustivel) VALUES

('AS-US-12', 1600, 12.8, 720, 'Vermelho', 34.2, true, 7, 1, 1);

INSERT INTO veiculo (matricula, potencia, cilindrada, tanque, cor, pesoMaxSuport, estaUsado, idMarca, idModelo, idCombustivel) VALUES

('ZR-IP-IV', 2600, 390, 720, 'Preto', 32.7, true, 2, 6, 1);

INSERT INTO veiculo (matricula, potencia, cilindrada, tanque, cor, pesoMaxSuport, estaUsado, idMarca, idModelo, idCombustivel) VALUES

('VB-24-AQ', 2600, 390, 720, 'Verde', 38.19, true, 1, 5, 4);

INSERT INTO pedido (dispCamiao, dispContentor, pesoCarga, dataLimite, nPortaDest, cPostalDest, ruaDest, cPostalOri, nPortaOri, ruaOri, precoEntrega, matriculaContentor, matriculaSegContentor, matricula, idCliente, idFatura) VALUES

(false, false, 3, '2022-12-5', 7, '4480-330', 'Rua Rui Fonceca', '4795-894', 186, 'Rua das flores', 4000, 'XA-43-21', null, 'QP-HP-12', 1, 1);

INSERT INTO pedido (dispCamiao, dispContentor, pesoCarga, dataLimite, nPortaDest, cPostalDest, ruaDest, cPostalOri, nPortaOri, ruaOri, precoEntrega, matriculaContentor, matriculaSegContentor, matricula, idCliente, idFatura) VALUES

(true, true, 4.5, '2023-01-15', 15, '3360-032', 'Rua Dr. David Braga', '2645-539', 48, 'Rua Cargas e Descargas', 5700.53, 'AF-GH-ZC', null, 'VB-24-AQ', 1, 2);

INSERT INTO pedido (dispCamiao, dispContentor, pesoCarga, dataLimite, nPortaDest, cPostalDest, ruaDest, cPostalOri, nPortaOri, ruaOri, precoEntrega, matriculaContentor, matriculaSegContentor, matricula, idCliente, idFatura) VALUES

(false, false, 10.5, '2024-04-04', 89, '4940-027', 'Rua Ze Beto', '2610-181', 12, 'Rua Sao Patricio', 10000.99, 'AF-GH-ZC', null, 'AS-US-12', 1, 4);

INSERT INTO pedido (dispCamiao, dispContentor, pesoCarga, dataLimite, nPortaDest, cPostalDest, ruaDest, cPostalOri, nPortaOri, ruaOri, precoEntrega, matriculaContentor, matriculaSegContentor, matricula, idCliente, idFatura) VALUES

(false, true, 5.4, '2023-03-13', 19, '4490-666', 'Rua Da Junqueira', '2205-025', 172, 'Rua Dr. Rodrigo Lisboa', 15473.46, 'QA-CV-ZA', null, 'ZR-IP-IV', 1, 5);

```
INSERT INTO grupomotorista (idPedido, idMotorista, quilometros, tipo) VALUES
(1, 2, 70, 'p');
INSERT INTO grupomotorista (idPedido, idMotorista, quilometros, tipo) VALUES
(2, 3, 150, 'p');
INSERT INTO grupomotorista (idPedido, idMotorista, quilometros, tipo) VALUES
(2, 2, 30, 'c');
INSERT INTO grupomotorista (idPedido, idMotorista, quilometros, tipo) VALUES
(3, 3, 700, 'p');
INSERT INTO grupomotorista (idPedido, idMotorista, quilometros, tipo) VALUES
(3, 2, 560, 'c');
INSERT INTO grupomotorista (idPedido, idMotorista, quilometros, tipo) VALUES
(4, 3, 200, 'p');
INSERT INTO grupomotorista (idPedido, idMotorista, quilometros, tipo) VALUES
(4, 2, 150, 'c');

INSERT INTO estado (nome) VALUES ('Execucao');
INSERT INTO estado (nome) VALUES ('Suspenso');
INSERT INTO estado (nome) VALUES ('Anulado');
INSERT INTO estado (nome) VALUES ('Concluido');
INSERT INTO estado (nome) VALUES ('Agendado');
INSERT INTO estado (nome) VALUES ('Pago');

INSERT INTO HistoricoEstados (idPedido, idEstado, idUtilizador) VALUES
(1, 5, 4);
INSERT INTO HistoricoEstados (idPedido, idEstado, idUtilizador) VALUES
(1, 4, 4);
INSERT INTO HistoricoEstados (idPedido, idEstado, idUtilizador) VALUES
(1, 6, 1);
INSERT INTO HistoricoEstados (idPedido, idEstado, idUtilizador) VALUES
(2, 3, 1);
```

5.2.2 - Views

View com o objetivo de apresentar toda a informação relevante dos pedidos:

```
create view pedidoInfo
(
   idPedido,
   disponibilidadeCamiao,
   disponibilidadeContentor,
   pesoCarga,
   dataLimite,
   precoEntrega,
   nPortDest,
   ruaDest,
   codigopostaldestino,
   localidadeDestino,
   paisDestino,
   nPortaOri,
   ruaOri,
   codigopostalorigem,
   localidadeOrigem,
   paisOrigem,
   contentorMatricula,
   contentorSecundarioMatricula,
   camiaoMatricula,
   cliente,
   precoCIva,
   dataEmissao,
   dataPagamento
)
as select
   p.idPedido,
   p.dispCamiao,
   p.dispContentor,
   p.pesoCarga,
   p.datalimite,
   p.precoEntrega,
   p.nPortaDest,
   p.ruaDest,
   cpDest.cPostal as codigopostaldestino,
   cpDest.localidade as localidadeDestino,
```

```
    paisDest.pais as paisDestino,
    p.nPortaOri,
    p.ruaOri,
    cpOri.cPostal as codigopostalorigem,
    cpOri.localidade as localidadeOrigem,
    paisOri.pais as paisOrigem,
    contP.matriculaContentor as contentorMatricula,
    contS.matriculaContentor as contentorSecundarioMatricula,
    camiao.matricula as camiaoMatricula,
    ut.nomeIni as cliente,
    fat.precoiva as precoCIva,
    fat.dataEmissao as dataEmissao,
    fat.dataPagam as dataPagamento
from pedido p
inner join codigopostal cpDest
    on p.cPostalDest = cpDest.cPostal
inner join pais as paisDest
    on paisDest.codPais = cpDest.codPais
inner join codigopostal cpOri
    on p.cPostalOri = cpOri.cPostal
inner join pais as paisOri
    on paisOri.codPais = cpOri.codPais
inner join contentor contP
    on contP.matriculaContentor = p.matriculaContentor
left join contentor contS
    on contS.matriculaContentor = p.matriculaSegContentor
inner join veiculo camiao
    on camiao.matricula = p.matricula
inner join fatura fat
    on fat.idFatura = p.idFatura
inner join utilizador ut
    on ut.idUtilizador = p.idCliente
inner join tipoutilizador tpUt
    on tpUt.idTipoUt = ut.idTipoUt
where tpUt.nome = 'Cliente'
;
```

View com o objetivo de apresentar todos os motoristas de todos os pedidos:

```
create view pedidoMotorista
(
   idPedido,
   motorista,
   adr,
   cam,
   cc,
   estaTrabalhar,
   tipo,
   quilometros
)
as select
   pi.idPedido,
   ut.nomeIni || ' ' || ut.nomeApelido as motorista,
   m.temADR as adr,
   m.temCam as cam,
   m.cartaoC as cc,
   m.estaTrabalhar,
   case
      when gm.tipo = 'p' then 'Principal'
      when gm.tipo = 'c' then 'Co-Piloto'
   end as tipo,
   gm.quilometros
from pedidoInfo pi
inner join grupomotorista gm
   on pi.idPedido = gm.idPedido
inner join motorista m
   on m.idMotorista = gm.idMotorista
inner join utilizador ut
   on ut.idUtilizador = m.idMotorista
;
```

5.2.3 - Triggers

Trigger com o objetivo de atribuir um tipo aos motoristas de um pedido:

```
create or replace function log_alterar_tipo_motorista()
  returns trigger
  language plpgsql
  as
$$
declare
   _motoristas int;
begin

   select count(*)
   from grupomotorista
   where idPedido = new.idPedido
   into _motoristas;

         if _motoristas > 1 then
      update grupomotorista
      set tipo = 'c'
      where idPedido = new.idPedido and idMotorista = new.idMotorista;
   else
      update grupomotorista
      set tipo = 'p'
      where idPedido = new.idPedido and idMotorista = new.idMotorista;
         end if;

         return new;
end;
$$;

create or replace trigger alterar_tipo_motorista
  after insert
  on grupomotorista
  for each ROW
  execute procedure log_alterar_tipo_motorista();
```

5.2.4 - Indexes

Index para identificar pedido mais rapidamente:

```
create unique index indexPedidos
on pedido (idPedido);
```

Index para identificar contentor mais rapidamente:

```
create unique index indexContentor
on contentor (matriculaContentor);
```

Index para identificar veiculo mais rapidamente:

```
create unique index indexVeiculo
on veiculo (matricula);
```

5.2.5 - Selects

```
reiport=# select
    c.matriculaContentor as matricula,
    c.largura,
    c.comprimento,
    c.profundidade,
    c.cor,
    c.pesoMaxSuport,
    c.estaUsado,
    tp.nome as tipo,
    mod.nome as modelo,
    mar.nome as marca
from contentor c
inner join tipo tp
    on tp.idTipo = c.idTipo
inner join modelo mod
    on mod.idModelo = c.idModelo
inner join marca mar
    on mar.idMarca = c.idMarca
;
 matricula | largura | comprimento | profundidade |   cor    | pesomaxsuport | estausado |    tipo    |  modelo  |   marca
-----------+---------+-------------+--------------+----------+---------------+-----------+------------+----------+-----------
 XA-43-21  |    2.35 |        5.90 |        33.20 | Preto    |         21.77 | f         | Granel     | Standard | EduCargas
 RK-ST-FG  |    2.35 |       12.04 |        67.70 | Amarelo  |         26.78 | t         | Granel     | Standard | EduCargas
 YU-EF-AS  |    2.35 |        5.90 |        33.20 | Branco   |         21.77 | f         | Granel     | Standard | EduCargas
 QA-CV-ZA  |    2.35 |       12.04 |        67.70 | Castanho |         26.78 | t         | Granel     | Standard | EduCargas
 PO-LK-JN  |    2.31 |        5.89 |        32.23 | Preto    |         21.60 | f         | Seca       | Open Top | EduCargas
 AF-GH-ZC  |    2.35 |       12.02 |        65.50 | Branco   |         26.63 | f         | Seca       | Open Top | EduCargas
 12-FG-AB  |    2.31 |        5.89 |        32.23 | Castanho |         21.60 | f         | Seca       | Open Top | EduCargas
 19-FG-24  |    2.35 |       12.02 |        65.50 | Verde    |         26.63 | f         | Seca       | Open Top | EduCargas
 FT-ZX-12  |    2.44 |        6.06 |         2.75 | Verde    |         24.00 | f         | Maquinaria | Open Top | EduCargas
(9 rows)
```

FIGURA 12 - VISUALIZAR TODOS OS CONTENTORES

```
reiport=# select
reiport-#     v.matricula,
reiport-#     v.potencia,
reiport-#     v.cilindrada,
reiport-#     v.tanque,
reiport-#     v.cor,
reiport-#     v.pesoMaxSuport,
reiport-#     v.estaUsado,
reiport-#     com.nome as combustivel,
reiport-#     mod.nome as modelo,
reiport-#     mar.nome as marca
reiport-# from veiculo v
reiport-# inner join combustivel com
reiport-#     on com.idCombustivel = v.idCombustivel
reiport-# inner join modelo mod
reiport-#     on mod.idModelo = v.idModelo
reiport-# inner join marca mar
reiport-#     on mar.idMarca = v.idMarca
reiport-# ;
 matricula | potencia | cilindrada | tanque |   cor    | pesomaxsuport | estausado | combustivel |  modelo   |  marca
-----------+----------+------------+--------+----------+---------------+-----------+-------------+-----------+----------
 QP-HP-12  |     2600 |         10 | 720.00 | Amarelo  |          8.30 | t         | Diesel      | Actros L  | Mercedes
 AS-US-12  |     1600 |         13 | 720.00 | Vermelho |          7.20 | t         | Diesel      | Actros L  | Mercedes
 ZR-IP-IV  |     2600 |        390 | 720.00 | Preto    |          8.30 | t         | Diesel      | V8        | Scania
 VB-24-AQ  |     2600 |        390 | 720.00 | Verde    |          8.30 | t         | Gas         | S-WAY Gas | Iveco
(4 rows)
```

FIGURA 13 - VISUALIZAR TODOS OS VEÍCULOS

```
reiport=# select
reiport-#     pd.idPedido,
reiport-#     est.nome as estado,
reiport-#     hist.dataInicioEstado,
reiport-#     ut.nomeIni || ' ' || ut.nomeApelido as nome,
reiport-#     tu.nome as tipo
reiport-# from historicoestados hist
reiport-# inner join pedido pd
reiport-#     on hist.idPedido = pd.idPedido
reiport-# inner join estado est
reiport-#     on hist.idEstado = est.idEstado
reiport-# inner join utilizador ut
reiport-#     on hist.idUtilizador = ut.idUtilizador
reiport-# inner join tipoutilizador tu
reiport-#     on ut.idTipoUt = tu.idTipoUt
reiport-# ;
 idpedido |  estado   | datainicioestado |         nome          |  tipo
----------+-----------+------------------+-----------------------+---------
        1 | Agendado  | 2022-12-20       | Sofia Ferreira        | Gestor
        1 | Concluido | 2022-12-20       | Sofia Ferreira        | Gestor
        1 | Pago      | 2022-12-20       | Francisco La Ventura  | Cliente
        2 | Anulado   | 2022-12-20       | Francisco La Ventura  | Cliente
(4 rows)
```

FIGURA 14 - VISUALIZAR ESTADOS DOS PEDIDOS

```
reiport=# select
reiport-#     SUM(precoSiva)
reiport-# from fatura
reiport-# where dataPagam is not NULL
reiport-# ;
    sum
-----------
 110179.86
(1 row)
```

FIGURA 15 - VISUALIZAR TOTAL GANHO COM PEDIDOS, POR PARTE DA EMPRESA

```
reiport=# select
reiport-#     count(ut.*) as quantidade
reiport-# from utilizador ut
reiport-# inner join tipoutilizador tu
reiport-#     on ut.idTipoUt = tu.idTipoUt
reiport-# where tu.nome <> 'Cliente'
reiport-# ;
 quantidade
------------
          5
(1 row)
```

FIGURA 16- VISUALIZAR QUANTIDADE DE FUNCIONÁRIO AO EMBARGO DA REIPORT

```
reiport=# select
    pais
from pais
;
     pais
---------------
 Portugal
 Austria
 Belgica
 Bulgaria
 Chequia
 Chipre
 Croacia
 Dinamarca
 Eslovaquia
 Eslovenia
 Espanha
 Estonia
 Finlandia
 Franca
 Grecia
 Hungria
 Irlanda
 Italia
 Letonia
 Lituania
 Luxemburgo
 Malta
 Paises Baixos
 Polonia
 Alemanha
 Romenia
 Suecia
(27 rows)
```

FIGURA 17 - VISUALIZAR PAÍSES PARA OS QUAIS A REIPORT TRANSPORTA

```
reiport=# select
reiport-#     ut.nomeIni || ' ' || ut.nomeApelido as nome,
reiport-#     ut.dataNasc,
reiport-#     ut.nif,
reiport-#     ut.rua || ' ' || ut.nPorta as morada,
reiport-#     cp.cPostal,
reiport-#     cp.localidade,
reiport-#     ut.email,
reiport-#     m.temADR,
reiport-#     m.temCam,
reiport-#     m.cartaoC as cc,
reiport-#     m.estaTrabalhar
from motorista mreiport-# from motorista m
reiport-# inner join utilizador ut
reiport-#     on ut.idUtilizador = m.idMotorista
reiport-# inner join codigopostal cp
reiport-#     on cp.cPostal = ut.cPostal
reiport-# where m.temADR is TRUE
reiport-# ;
      nome       |  datanasc  |    nif    |         morada         | cpostal |    localidade    |          email          | temadr | temcam |    cc    | estatrabalhar
-----------------+------------+-----------+------------------------+---------+------------------+-------------------------+--------+--------+----------+---------------
 Ricardo Machado | 1991-06-15 | 323316608 | Rua Dr. Andre Tomas 7  | 3520-039 | Nelas            | ricardinho@gmail.com    | t      | t      | 45612378 | f
 Sofia Ferreira  | 1990-06-27 | 851642243 | Rua Diogo Cao 5        | 4475-045 | Maia             | sofiaferreira@gmail.com | t      | f      | 75312347 | f
 Gabriel Machado | 1972-07-20 | 433492058 | Rua da flor dourada 45 | 4900-281 | Viana do Castelo | gabimachado@gmail.com   | t      | t      | 96375315 | t
(3 rows)
```

FIGURA 18 - VISUALIZAR MOTORISTAS QUE POSSUEM ADR

```
reiport=# select
reiport-#     ut.nomeIni || ' ' || ut.nomeApelido as nome,
reiport-#     ut.dataNasc,
reiport-#     ut.nif,
reiport-#     ut.rua || ' ' || ut.nPorta as morada,
reiport-#     cp.cPostal,
reiport-#     cp.localidade,
reiport-#     ut.email,
reiport-#     m.temADR,
reiport-#     m.temCam,
reiport-#     m.cartaoC as cc,
reiport-#     m.estaTrabalhar
reiport-# from motorista m
reiport-# inner join utilizador ut
reiport-#     on ut.idUtilizador = m.idMotorista
reiport-# inner join codigopostal cp
reiport-#     on cp.cPostal = ut.cPostal
reiport-# where m.temCam is TRUE
reiport-# ;
      nome       |  datanasc  |    nif    |          morada          | cpostal |    localidade    |         email          | temadr | temcam |    cc    | estatrabalhar
-----------------+------------+-----------+--------------------------+---------+------------------+------------------------+--------+--------+----------+---------------
 Ricardo Machado | 1991-06-15 | 323316608 | Rua Dr. Andre Tomas 7    | 3520-039 | Nelas            | ricardinho@gmail.com   | t      | t      | 45612378 | f
 Joao Oliveira   | 1968-08-14 | 222837141 | Avenida Nova Esperanca 17 | 4490-251 | Povoa de Varzim  | oliveira@gmail.com     | f      | t      | 15957461 | f
 Gabriel Machado | 1972-07-20 | 433492058 | Rua da flor dourada 45   | 4900-281 | Viana do Castelo | gabimachado@gmail.com  | t      | t      | 96375315 | t
(3 rows)
```

FIGURA 19 - VISUALIZAR MOTORISTAS QUE POSSUEM CAM

```
reiport=# select

    sumreiport-#     gm.idPedido,
reiport-#     sum(gm.quilometros) as quilometros
reiport-# from grupomotorista gm
reiport-# inner join motorista mt
reiport-#     on mt.idMotorista = gm.idMotorista
reiport-# inner join utilizador ut
reiport-#     on ut.idUtilizador = mt.idMotorista
reiport-# group by (gm.idPedido)
reiport-# order by gm.idPedido
reiport-# ;
 idpedido | quilometros
----------+-------------
        1 |       70.00
        2 |      180.00
(2 rows)
```

FIGURA 20 - VISUALIZAR OS QUILÓMETROS REALIZADOS EM CADA PEDIDO

```
reiport=# select
   gm.idreiport-#     gm.idPedido,
reiport-#     count(gm.idMotorista) as qtdMotoristas
reiport-# from grupomotorista gm
reiport-# group by gm.idPedido
reiport-# order by gm.idPedido
reiport-# ;
 idpedido | qtdmotoristas
----------+---------------
        1 |             1
        2 |             2
(2 rows)
```

FIGURA 21 - VISUALIZAR A QUANTIDADE DE MOTORISTAS EM CADA PEDIDO

```
reiport=# select
reiport-# count(v.*) as qtd
reiport-# from veiculo v
reiport-# inner join marca mar
reiport-# on v.idMarca = mar.idMarca
reiport-# where v.pesoMaxSuport > 7.45
reiport-# ;
 qtd
-----
   3
(1 row)
```

FIGURA 22 - VISUALIZAR A QUANTIDADE DE CAMIÕES QUE SUPORTAM MAIS DE 7,45 TONELADAS

```
reiport=# select
 ' ' ||reiport-# ut.nomeIni || ' ' || ut.nomeApelido as nome,
reiport-# sum(ft.precosiva) as total
reiport-# from motorista mt
reiport-# inner join utilizador ut
reiport-#     on ut.idUtilizador = mt.idMotorista
reiport-# inner join grupomotorista gm
reiport-# on gm.idMotorista = mt.idMotorista
reiport-# inner join pedido pd
reiport-# on pd.idPedido = gm.idPedido
reiport-# inner join fatura ft
reiport-# on ft.idFatura = pd.idFatura
reiport-# where ft.datapagam is not null
reiport-# group by (nome)
reiport-# ;
      nome       |   total
-----------------+-----------
 Joao Oliveira   | 110179.86
 Ricardo Machado | 110179.86
(2 rows)
```

FIGURE 23 - VISUALIZAR OS MOTORISTAS QUE GANHARAM MAIS DINHEIRO COM OS SEUS PEDIDOS

```
reiport=# select
reiport-#     pd.idPedido,
reiport-#     sum(fat.precosiva) as totalPago
reiport-# from fatura fat
reiport-# inner join pedido pd
reiport-#     on pd.idFatura = fat.idFatura
reiport-# where extract(year from fat.dataPagam) > 2020
reiport-# group by pd.idPedido
reiport-# order by pd.idPedido
reiport-# ;
 idpedido | totalpago
----------+-----------
        3 |  37009.13
        4 |  73170.73
(2 rows)
```

FIGURE 24 - VISUALIZAR OS PEDIDOS PAGOS DEPOIS DO ANO 2020

5.2.6 - Atualização de dados

5.2.6.1 - Updates

Update com o objetivo de alterar a data da fatura com idFatura = 5:

```
update fatura
set dataPagam = '2023-01-01'
where idFatura = 5
;
```

Update com o objetivo de alterar a data da fatura com idFatura = 4:

```
update fatura
set dataPagam = '2022-12-25'
where idFatura = 4
;
```

Update com o objetivo de alterar a morada do cliente com idUtilizador = 5:

```
update utilizador
set rua = 'Avenida Boa Esperanca', nPorta = 32, cPostal = '2610-181'
where idUtilizador = 5
;
```

Update com o objetivo de alterar o tipo de utilizador de cliente para motorista, do cliente com idUtilizador = 7:

```
update utilizador
set idTipoUt = 4
where idUtilizador = 7
;
```

5.2.6.2 Deletes

Delete realizado com o objetivo de eliminar Utilizador com idUtilizador = 7:

```
delete from utilizador where idUtilizador = 7;
```

Delete realizado com o objetivo de eliminar Contentor com matriculaContentor = 'FT-ZX-12':

```
delete from contentor where matriculaContentor = 'FT-ZX-12';
```

5.2.7 Stored Procedures

Stored Procedure para adicionar um Cliente:

```
Create OR replace PROCEDURE addCliente (
   c_nomeIni VARCHAR(50),
   c_nomeApelido VARCHAR(50),
   c_dataNasc DATE,
   c_nif NUMERIC(9),
   c_rua VARCHAR(100),
   c_nPorta INT,
   c_cPostal VARCHAR(8),
   c_telefone NUMERIC(9),
   c_email VARCHAR(200),
   c_passwd VARCHAR(100)
)
LANGUAGE plpgsql AS
$$ BEGIN

   INSERT INTO utilizador (nomeIni, nomeApelido, dataNasc, nif, rua, nPorta, cPostal, telefone, email,
passwd, idTipoUt) VALUES
   (c_nomeIni, c_nomeApelido, c_dataNasc, c_nif, c_rua, c_nPorta, c_cPostal, c_telefone, c_email,
c_passwd, 1);

END $$;
```

Call para chamar store procedure:

```
call addCliente('Maia', 'Maria', '1985-02-07', 837502849, 'Rua Cesario Verde', 220, '3420-177',
917503249, 'maiaMaria@gmail.com', 'noonewillfindout12?!');
```

Stored Procedure para adicionar um HistoricoEstado:

```
Create OR replace PROCEDURE addHistorico (
   c_idUtilizador int,
   c_idFatura int
)
LANGUAGE plpgsql AS
$$
DECLARE
   _pedido int;
BEGIN

   update fatura
   set dataPagam = now()::date
   where idFatura = c_idFatura;

   select
      p.idPedido
   from fatura f
   inner join pedido p
      on p.idFatura = f.idFatura
   into _pedido;

   insert into historicoestados
   (idPedido, idEstado, idUtilizador)
   values (1, 6, c_idUtilizador);

END $$;
```

Call para chamar store procedure:

```
call addHistorico(2,5);
```

6. Conclusão

Este relatório redigido no âmbito da Unidade Curricular de Projeto I, demonstra o trabalho realizado pelo nosso grupo, bem como o nosso percurso de aprendizagem, conhecimento adquirido nas aulas e de dúvidas esclarecidas pelos professores. Assim sendo, foram realizadas ao longo do trabalho várias atualizações a modelos, casos de uso e diagramas já entregues.

Durante todo o trabalho prático surgiram dúvidas e obstáculos, que pensamos que foram ultrapassados e nos fizerem perceber melhor certos modelos, diagramas ou códigos utilizados.

Para que fosse possível trabalhar com a base de dados Postgresql, foi utilizada a ferramenta Docker de forma a atingir o seu isolamento. Além disso, utilizou-se a WSL do Windows para obter um ambiente Linux que permitisse a utilização da ferramenta psql para a execução dos scripts formulados no desenvolvimento do trabalho prático.

Conseguimos, assim, complementar a matéria alvo de aprendizagem nas aulas das Unidades Curriculares de Engenharia de Software I e Base de Dados, além de desenvolver a nossa capacidade de trabalhar em equipa, para obter o melhor resultado possível.