U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Bases de Dados

Aluguer de Carros

LEIC012 – Grupo 1205

(11 de outubro de 2023)

Leonardo Garcia **up202200041**@fe.up.pt

Marcel Medeiros **up202200042**@fe.up.pt

Gonçalo Sousa **up202207320**@fe.up.pt

Índice

Introdução

No âmbito da Unidade Curricular de 2º Ano Bases de Dados, iremos, com este projeto, descrever e implementar a base de dados de uma empresa fictícia de aluguer de carros.

Desta forma, o grupo recorreu a técnicas de representação UML com o intuito de detalhar o modelo conceptual.

Contexto da Base de Dados

Como nosso cliente temos uma empresa que quer construir a base de dados para um negócio de aluguer de carros. Com esta base de dados, a empresa pretende armazenar não só dados relativos aos clientes como também à sua frota de veículos e o ato de alugar o carro.

É importante frisar que esta base de dados não descreve uma aplicação de aluguer de carros, ou seja, o nosso foco com este projeto não é definir uma interface que possibilite, através dela, um cliente efetuar o aluguer. É a recolha dos supracitados dados para armazenamento que nos interessa.

Pessoa

Pessoa é uma generalização que define a parte humana da base de dados. Desta importa saber o nome, a data de nascimento, a morada, o contacto telefónico e o e-mail. As pessoas podem ser divididas em duas especializações – Cliente e Funcionário.

Cliente

Acerca do Cliente, observamos que, tal como a classe Funcionário, é uma especialização de Pessoa. Decidimos fazer a divisão dados os atributos que se repetiriam entre o Cliente e o Funcionário.

Os dados da Carta de Condução de cada cliente são relevantes, desta forma, o ID da Carta de Condução, bem como a validade e a sua data de emissão são recolhidos e armazenados como atributos desta classe.

Funcionário

A cada funcionário que trabalha na empresa é lhe associado, por motivos de identificação, um ID. O número de horas semanais bem com o salário por hora de cada um é armazenado como atributo. Isto serve para a empresa saber quanto tem de pagar a cada funcionário no mês.

Marca

A entidade marca possui como atributo apenas um nome. Esta associar-se-à com a classe Modelo, podendo ter vários modelos associados à mesma marca.

Modelo

A classe Modelo é importante para especificar o tipo de carro no qual aquele modelo se encaixa. Cada modelo só pode ter uma única marca associada e um modelo pode se associar com um ou muitos carros.

Seguradora

É importante guardarmos na nossa base as Seguradoras parceiras, para isso, informações como seu nome, morada, telefone e e-mail serão armazenadas como atributos. A função principal desta entidade é fornecer à empresa seguros variados, que por sua vez podem ser fornecidos por diversas seguradoras.

Seguro

A classe Seguro mostra-se necessária para garantir ao cliente a cobertura de danos a vários níveis. Esta informação, aliada aos benefícios de viagem e acomodação que um seguro de automóvel pode oferecer, irá ajudar a definir o tipo do plano que, por sua vez, irá ditar seu preço. É importante destacar que um seguro será obrigatoriamente associado ao ato de aluguer.

Carro

Os carros pertencentes à empresa serão armazenados de forma a que a base guarde informações da matrícula e do estado do veículo. Cada carro possui um único modelo associado, e de acordo com o tipo de carro especificado pela entidade Modelo, faz parte de um único plano de aluguer. Decidimos utilizar as classes Modelo e Marca, ao invés de apenas os colocar como atributos da classe Carro, para evitar várias repetições de marca e modelo, uma vez que vários carros podem ter o mesmo modelo e, consequentemente a mesma marca

Plano de Aluguer

Os planos que a empresa oferece serão ditados pelo carro associado ao plano e seu respectivo tipo de modelo, desta forma, um carro de categoria mais alta terá um preço mais elevado.

Extras

Os extras são adicionais específicos à necessidade do cliente e podem ser adicionados de acordo com a escolha pessoal daquele que irá contratar o plano. Exemplos de extras podem ser "cadeirinha de bébé" ou "sistema *Bluetooth*", ou seja, partes adicionais do carro que agregam valor a experiência, e claro, acrescentam um preço extra.

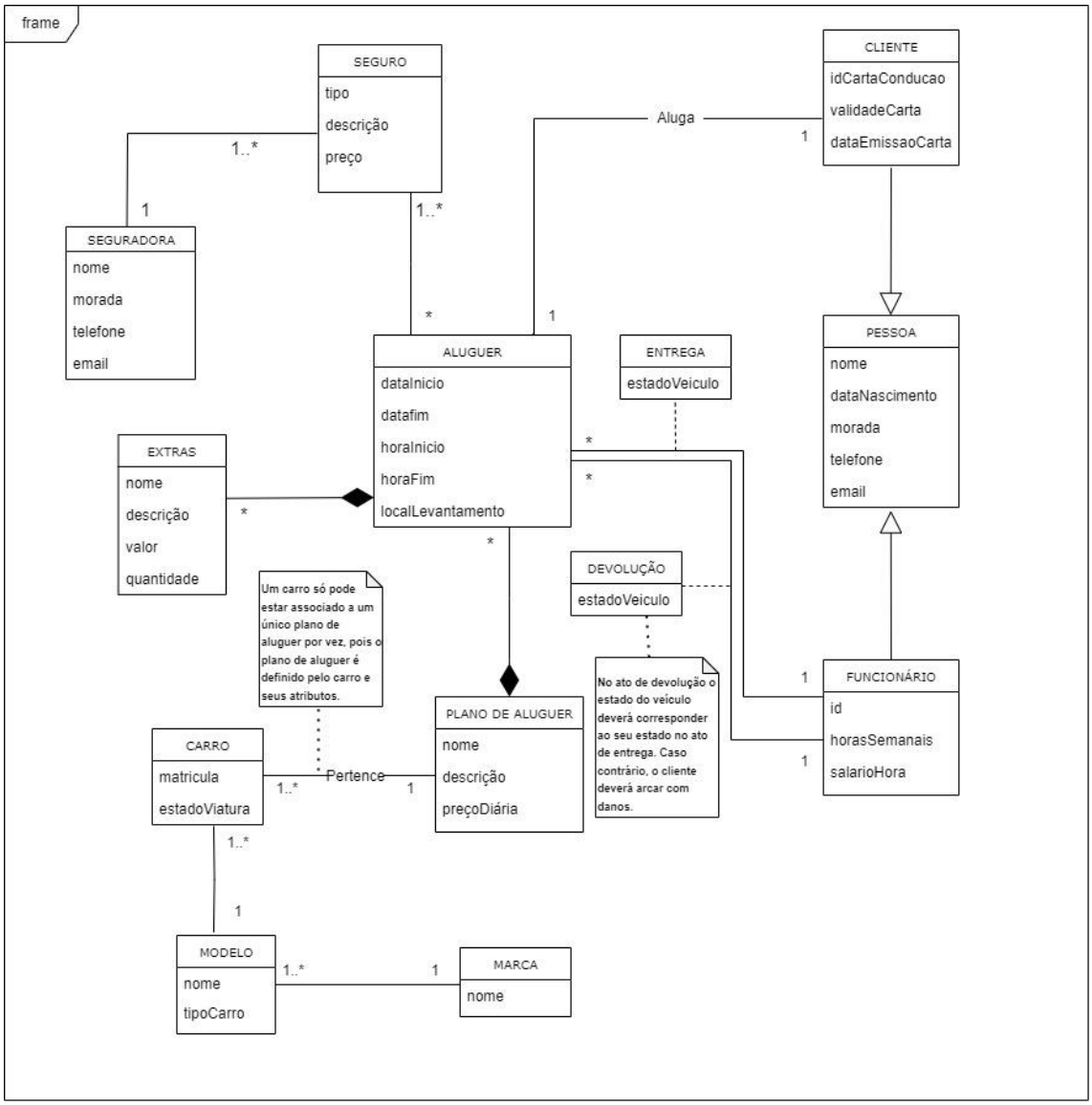
Aluguer

A classe Aluguer define o ato de alugar um veículo. Desta forma, é importante registar tanto a data e hora do início do contrato, como a data e hora de entrega do veículo, ou seja, do fim do contrato. Como as empresas podem ter um ou mais locais de levantamento de viaturas, também armazenaremos esta informação.

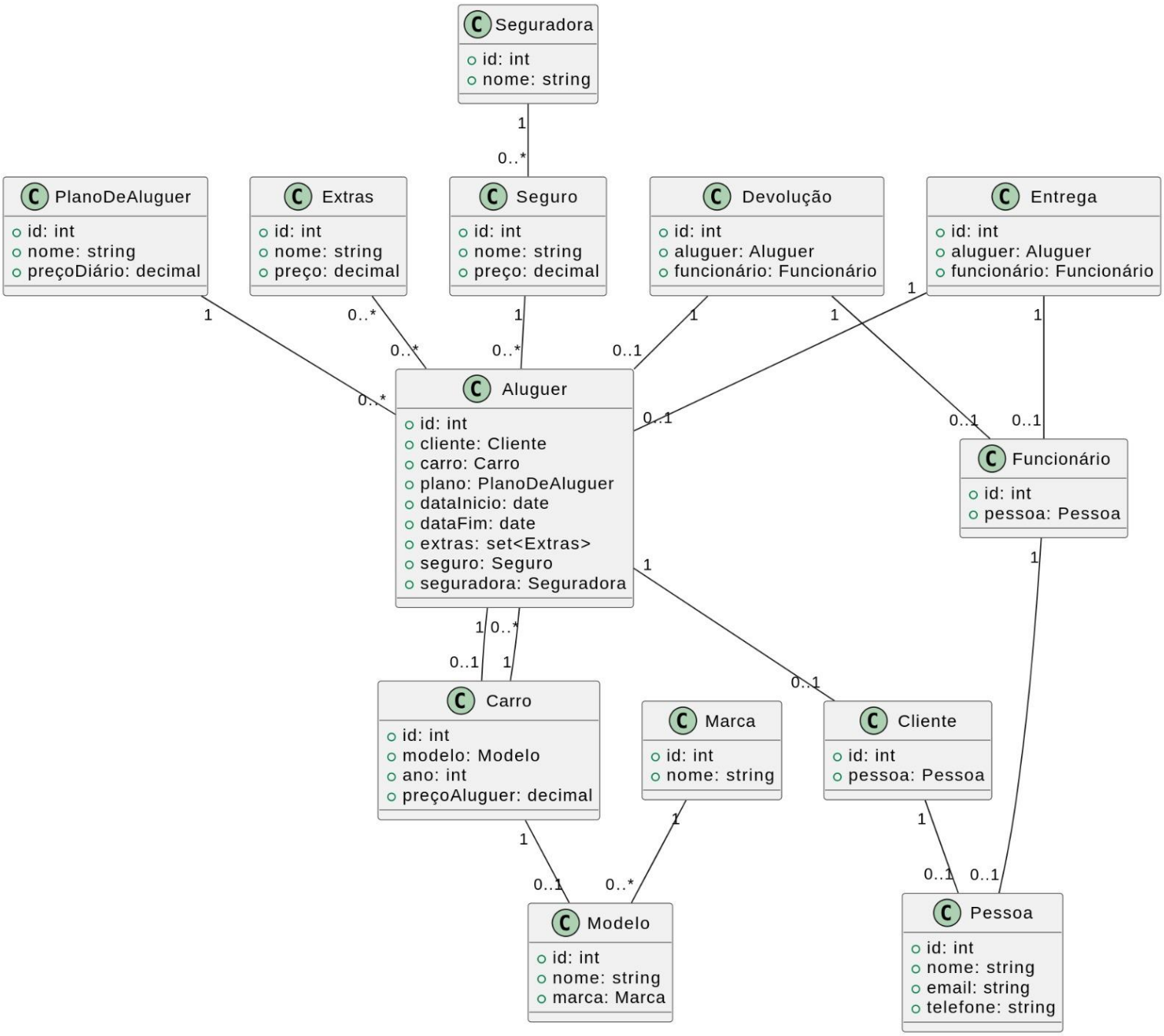
Entrega e Devolução

A necessidade de especificar o ato de entrega e devolução surgiu para termos como identificar o funcionário que exercerá cada função, além de guardar o estado do veículo em ambos os momentos.

Diagrama de Classes – UML Versão Original



Diagramas de Classe UML – Versão IA INPUT



Descrição do Processo de Integração IA

A integração IA surge como uma ferramenta para analisarmos as melhorias que poderíamos ter feito no nosso projeto e, de acordo com nossa lógica, julgar se a implementação do ponto de vista da inteligência artificial faria sentido. Para isso usamos a ferramenta **ChatUML** da seguinte maneira:

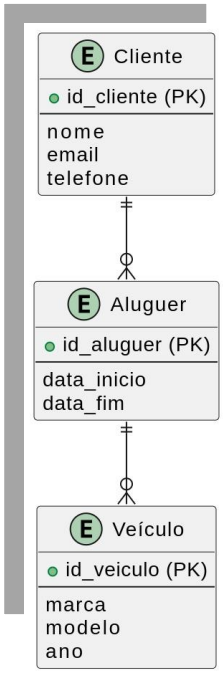
1º Prompt -

No âmbito da Unidade Curricular de 2º Ano Bases de Dados, iremos, com este projeto, descrever e implementar a base de dados de uma empresa fictícia de aluguer de carros.

Desta forma, o grupo recorreu a técnicas de representação UML com o intuito de detalhar o modelo conceitual. Como nosso cliente, temos uma empresa que quer construir a base de dados para um negócio de aluguer de carros. Com esta base de dados, a empresa pretende armazenar não só dados relativos aos clientes como também à sua frota de veículos e o ato de alugar o carro.

É importante frisar que esta base de dados não descreve uma aplicação de aluguer de carros, ou seja, o nosso foco com este projeto não é definir uma interface que possibilite, através dela, um cliente efetuar o aluguer. É a recolha dos supracitados dados para armazenamento que nos interessa.

output que nos foi dado inicialmente foi este:

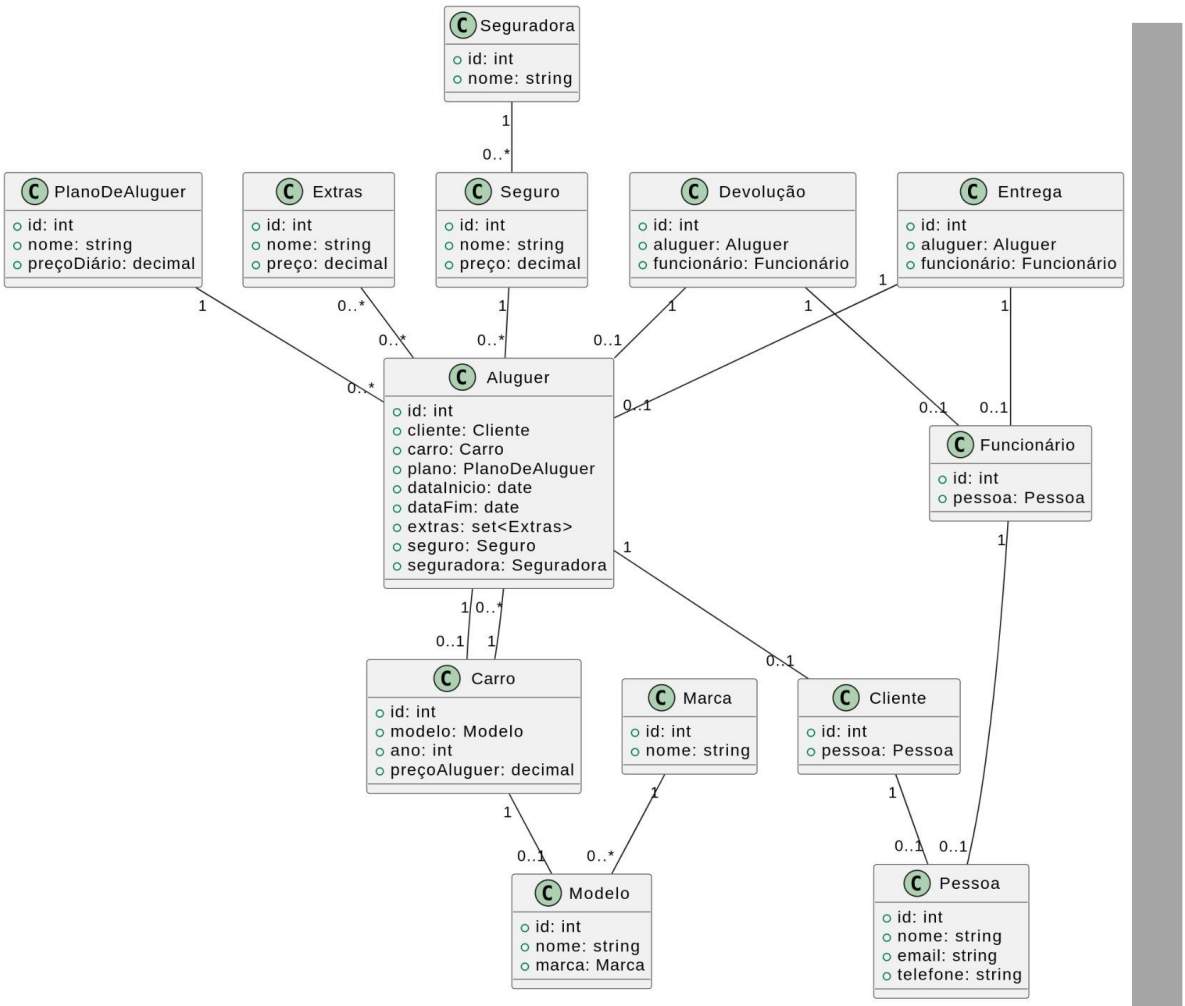


Assim, sentimos necessidade de ser mais expressivos com a IA:

2º Prompt -

Ok, mas ficou muito básico para o que temos em mente no nosso projeto. Queremos trabalhar com as seguintes classes: Pessoa, Cliente, Funcionário, Marca, Modelo, Carro, Plano de Aluguer, Extras, Seguro, Seguradora, Aluguer, Entrega e Devolução. Entrega e Devolução devem conter informações sobre o funcionário que realizou essas duas operações.

Então o ChatUML deu-nos um diagrama mais incorpado e com mais classes:



Finalmente, pedimos para a IA implementar o que ela achasse que faria sentido ser adicionado ao nosso projeto, porém o resultado foi o mesmo do anterior...

Análise Crítica dos Resultados e Considerações Finais

Dada a despretensiosidade dos resultados obtidos com o primeiro *input*, achamos desnecessário fazer uma análise sobre o *output*. Embora relativamente bem detalhada, a descrição que fizemos não foi suficiente para obtermos um diagrama com potencial.

O mesmo não acontece com o segundo *input*. Após fazermos uma referência exata das classes que queríamos adicionar, a IA conseguiu fazer um diagrama bastante similar ao nosso. Inclusive adicionou os *Data Types* de cada atributo, algo que não fizemos por agora.

Embora a semelhança com o nosso diagrama por consequência da objetividade desta técnica de representação, que não dá grande espaço para "inventar", achamo-lo um pouco confuso ao início – talvez por não ser da nossa autoria. Outra nota importante de realçar é a referência a relações entre classes como um atributo da classe de partida. Por exemplo, na classe Carro há um atributo modelo do tipo Modelo. Isto dificulta a leitura e interpretação do diagrama, acabando mesmo por ser um pouco redundante, uma vez que essa relação já está representada e é facilmente identificável.

Algo que também nos chamou à atenção foi a forma de representar algumas relações. Associações que seriam do tipo composição estão representadas em inconformidade com o modelo UML. Embora para efeitos práticos a composição esteja subentendida na carnalidade, achamos que seria mais correto recorrer à forma convencionada, tal como nós fizemos.

Tendo em conta também que Cliente e Funcionário são especializações da generalização Pessoa, a relação entre elas deveria estar representada com setas que apontam para a generalização de modo a respeitar o modelo UML.

A carnalidade das relações nem sempre foi a correta. Passo a enunciar algumas relações que deveriam ser retificadas – todas as '0..*' podem ser substituídas por '*' dado que '*' já inclui a possibilidade do 0 ocorrer; da classe Aluguer para a classe Carro, se o aluguer existe então alugar-se-á sempre um carro, ou seja, devemos trocar '0..1' por '1'; entre outras.

A inserção de um atributo ID em todas as classes foi algo que achamos interessante e de certa forma pertinente. De facto, usar um ID dá à classe um atributo único que pode ser usado como chave primária e permite a sua identificação em qualquer circunstância.

Dito isto, podemos concluir que mesmo tendo feito alguns acrescentos relevantes, a IA ainda não garante uma precisão de 100% no que toca a este tipo de trabalhos. Pode e deve ser usado como uma ferramenta de auxílio, mas nunca como substituto do trabalho humano.