

Faculdade de Engenharia da Universidade do Porto



Universidade do Porto

FEUP Faculdade de Engenharia

Livraria

Base de Dados

Licenciatura Engenharia Informática e Computação

Carla Alexandra Teixeira Lopes
Michel Celestino Paiva Ferreira
Mariana Ferreira Dias

Estudantes e Autores

2LEIC08 - Grupo 806

Ana Carolina Brandão, up202004461@up.pt

Pedro Marques, up202107525@up.pt

Marlon Belo, up202209531@up.pt

Outubro 2023
Porto, Portugal

Índice

Índice	2
Introdução	3
Contexto	3
Diagrama UML	4
Integração de inteligência artificial generativa	5
Diagrama UML antes do chat gpt:	9
Considerações finais:	10

Introdução

O objetivo deste projeto é obter experiência prática no design e implementação de uma base de dados relacional em SQLite. Mais concretamente, pretende-se organizar e modelar as interações relativas a uma loja de livros, uma livraria.

Contexto

A **Livraria** é identificada pelo seu nome e morada, possui também disponível um contacto telefónico. Funciona consoante dois **horários de atendimento** diferentes, um para os dias úteis da semana e outro para fins de semana e feriados.

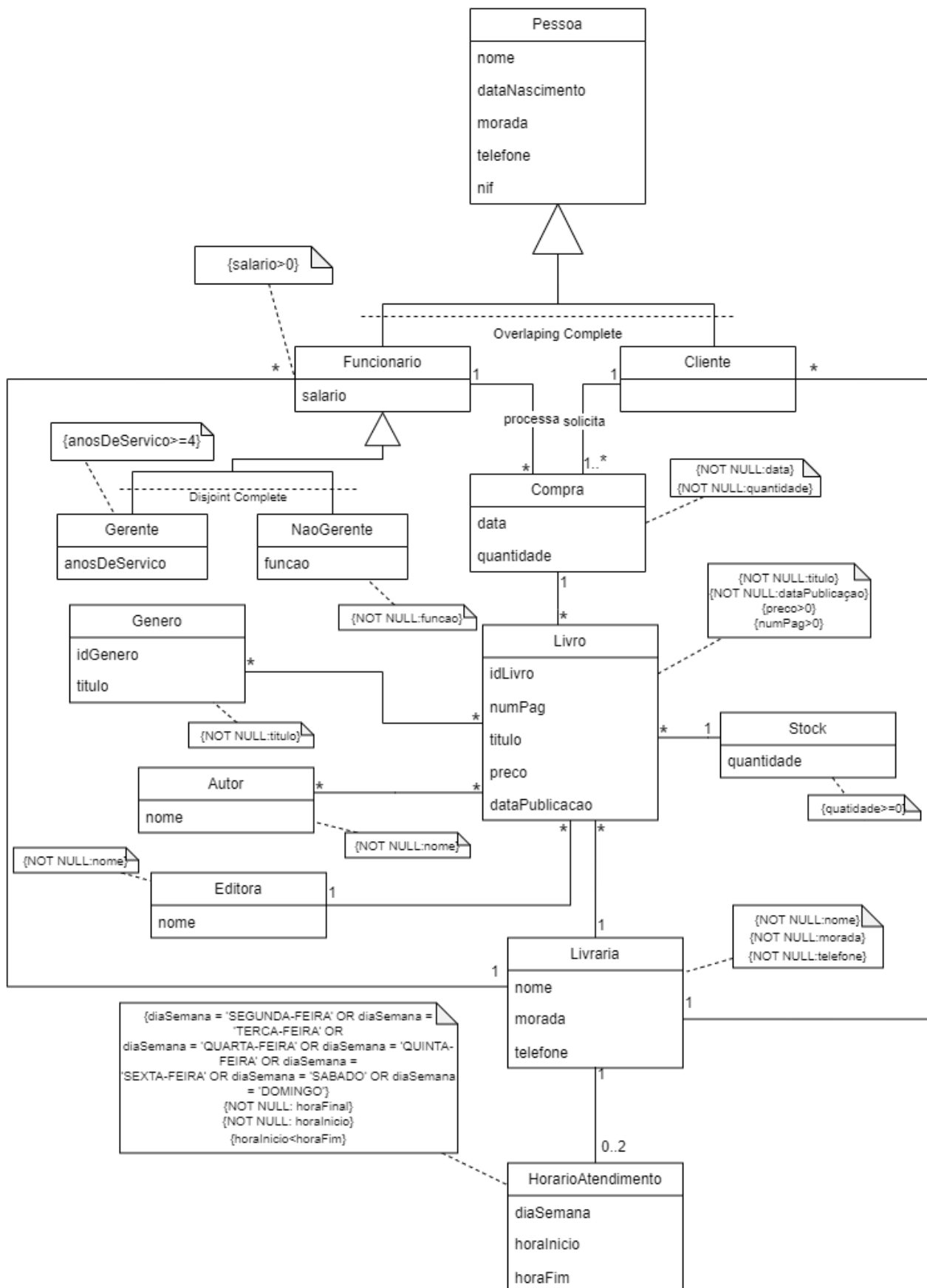
Tanto os **clientes** como os **funcionários** são identificados no sistema informático da livraria pelo seu nome, data de nascimento, morada, telefone e nif. Sendo os funcionários diferenciados dos clientes devido a possuírem um salário.

Os funcionários podem ou não ocupar o cargo de **gerência**, sendo que para tal aconteça, devem ser cumpridas certas condições. Apenas é possível a ocupação do cargo de gerente por um funcionário de cada vez. É necessário que o funcionário tenha no mínimo 4 anos de serviço.

Cada **compra** é processada por um funcionário e tem associada a quantidade de livros comprados e a data.

Um cliente pode comprar vários **livros**, desde que estejam presentes em **stock**. Livros estes de diversos géneros, autores e editoras. Um livro é identificado no sistema informático pelo seu id (idLivro) e possui as seguintes informações associadas: Título, número de páginas, preço e data de publicação.

Diagrama UML



Integração de inteligência artificial generativa

Para esta fase do Projeto foi utilizada primeiramente a ferramenta chat gpt-3.5 pertencente à empresa OpenAi [ChatGPT \(openai.com\)](https://openai.com).

Segue abaixo a nossa conversa com a Inteligência artificial, a **rosa** as prompts utilizadas e a **verde** as respostas obtidas juntamente com as nossas apreciações da mesma.

“Olá. Pretende-se organizar e modelar as interações relativas a uma loja de livros, uma livraria.

Poderias por favor fazer um esquema em UML para este efeito?

Aqui vai a descrição: *Contexto”

Dado este input o ChatBot gerou as classes que considerou necessárias para a modelagem do nosso enunciado das quais:

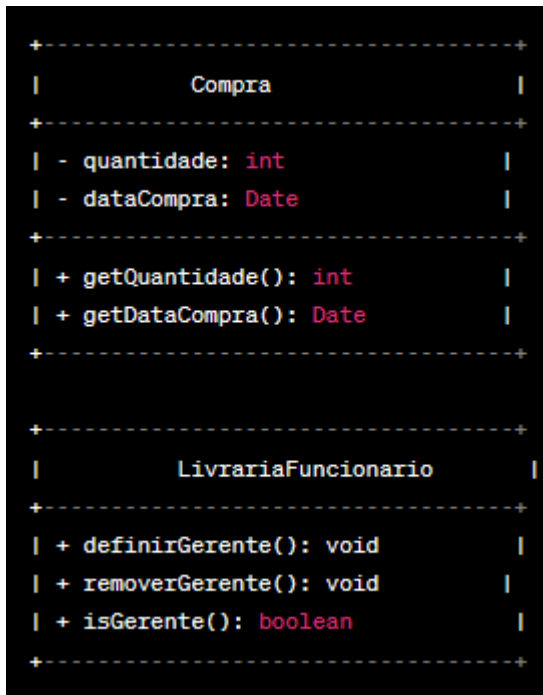
```
+-----+
|          Livraria          |
+-----+
| - nome: String             |
| - morada: String           |
| - contactoTelefonico: String |
| - horarioSemana: String     |
| - horarioFimDeSemana: String |
+-----+
| + getNome(): String        |
| + getMorada(): String      |
| + getContactoTelefonico(): String |
| + getHorarioSemana(): String |
| + getHorarioFimDeSemana(): String |
+-----+

+-----+
|          Pessoa           |
+-----+
| - nome: String             |
| - dataNascimento: Date      |
| - morada: String           |
| - telefone: String          |
| - nif: String               |
+-----+
| + getNome(): String        |
| + getDataNascimento(): Date |
| + getMorada(): String      |
| + getTelefone(): String     |
| + getNif(): String          |
+-----+
```

```
+-----+
|          Funcionario      |
+-----+
| - salario: double          |
| - cargo: String            |
| - anosServico: int          |
+-----+
| + getSalario(): double     |
| + getCargo(): String       |
| + getAnosServico(): int     |
+-----+

+-----+
|          Cliente          |
+-----+
| + comprarLivro(livro: Livro): void |
+-----+

+-----+
|          Livro            |
+-----+
| - idLivro: int             |
| - titulo: String           |
| - numPaginas: int          |
| - preco: double            |
| - dataPublicacao: Date     |
+-----+
| + getIdLivro(): int        |
| + getTitulo(): String      |
| + getNumPaginas(): int     |
| + getPreco(): double       |
| + getDataPublicacao(): Date |
+-----+
```



“Neste diagrama:

Livraria representa a livraria com seus atributos básicos, como nome, morada e horários de atendimento.

Pessoa é uma classe base que contém informações compartilhadas por funcionários e clientes.

Funcionario representa os funcionários da livraria com atributos adicionais, como salário, cargo e anos de serviço.

Cliente é uma classe que herda de Pessoa e possui um método para comprar livros.

Livro representa os livros disponíveis na livraria.

Compra contém informações sobre cada compra, como quantidade e data.

LivrariaFuncionario é uma classe auxiliar que permite definir um funcionário como gerente da livraria e verificar se um funcionário é gerente.

Lembre-se de que você pode adicionar mais detalhes e métodos conforme necessário para atender aos requisitos específicos da sua livraria.”

>>Podemos observar que de facto criou aproximadamente as mesmas classes e atributos que nós para a modelação do contexto fornecido. Este facto revela uma boa compreensão do problema e noção de contexto. É de salientar que não foi gerado um diagrama UML como pedido nem foi modelado como as classes se relacionam nem quais as suas multiplicidades (restrições em falta também). É provável que tal aconteça devido ao ChatBot estar restringido a fornecer respostas numa caixa de texto de tamanho e recursos limitados (sem suporte para esquemas). Vamos então tentar outra abordagem.<<

“Podes por favor descrever como se relacionam estas classes no contexto de um diagrama UML e quais as suas multiplicidades e restrições?”

“Vou descrever como as classes se relacionam no contexto de um diagrama UML, incluindo suas multiplicidades e restrições.

Relacionamento entre Livraria e Funcionario:

Associação: A classe Livraria tem uma associação com a classe Funcionario. Os funcionários trabalham na livraria.

Multiplicidade:

Livraria tem uma multiplicidade de 1 para N (ou seja, uma livraria pode ter vários funcionários).

Funcionario tem uma multiplicidade de 0 para N (um funcionário pode não estar associado a nenhuma livraria ou a várias).

Restrições: Nenhuma restrição específica.

Relacionamento entre Funcionario e Cliente:

Associação: A classe Funcionario e a classe Cliente são ambas subtipos da classe base Pessoa. Isso implica que funcionários e clientes compartilham informações pessoais em comum.

Multiplicidade: Não se aplica, pois é uma associação de herança.

Relacionamento entre Funcionario e LivrariaFuncionario:

Associação: A classe Funcionario tem uma associação com a classe LivrariaFuncionario para determinar se um funcionário é um gerente ou não.

Multiplicidade: A multiplicidade é de 0..1 de LivrariaFuncionario, o que significa que um funcionário pode ou não estar associado a um objeto LivrariaFuncionario (gerente).

Restrições:

Para que um funcionário seja considerado um gerente, ele deve estar associado a um objeto LivrariaFuncionario. Isso é definido pelo método definirGerente().

Relacionamento entre Cliente e Compra:

Associação: A classe Cliente tem uma associação com a classe Compra. Isso indica que um cliente pode realizar várias compras.

Multiplicidade:

Cliente tem uma multiplicidade de 0 para N (um cliente pode não fazer compras ou fazer várias).

Compra tem uma multiplicidade de 0 para N (uma compra pode ser feita por zero ou mais clientes).

Relacionamento entre Compra e Livro:

Associação: A classe Compra tem uma associação com a classe Livro. Isso indica que cada compra está relacionada a um ou mais livros.

Multiplicidade:

Compra tem uma multiplicidade de 1 para N (cada compra envolve pelo menos um livro).

Livro tem uma multiplicidade de 0 para N (um livro pode estar presente em zero ou mais compras).

Relacionamento entre Livro e Livraria:

Associação: A classe Livro tem uma associação com a classe Livraria para representar que a livraria possui uma coleção de livros.

Multiplicidade:

Livro tem uma multiplicidade de 1 para N (um livro pertence a uma ou mais livrarias).

Livraria tem uma multiplicidade de 0 para N (uma livraria pode ter zero ou mais livros).

Estas descrições representam como as classes estão relacionadas no contexto do diagrama UML e ajudam a entender as conexões entre as diferentes partes do sistema da

livraria. Lembre-se de que as restrições específicas podem variar dependendo dos detalhes da implementação do sistema.”

>>Podemos mais uma vez ver uma semelhança entre o output do ChatBot com o UML realizado pelo nosso grupo. Em geral, possui boas noções no que toca às relações entre classes e às suas respectivas multiplicidades, contudo, não concordamos com várias das multiplicidades que utilizou. Tal deve-se ao facto desta ser uma questão influenciada pela interpretação concreta do contexto e objetivo da base de dados. Ao analisar a resposta mais profundamente apercebemo-nos de que tínhamos uma inconsistência no que tocava às multiplicidades e que faltava a ligação de algumas classes, o que nos permitiu a sua correção.

Tome-se como exemplo os seguintes casos com os quais o nosso grupo não concordou: <<

“Funcionario tem uma multiplicidade de 0 para N (um funcionário pode não estar associado a nenhuma livraria ou a várias).”

>>Consideramos que um funcionário tem de estar associado a uma e uma só livraria, já que é aquilo que a nossa base de dados modela.<<

“A multiplicidade é de 0..1 de LivrariaFuncionario, o que significa que um funcionário pode ou não estar associado a um objeto LivrariaFuncionario (gerente).”

>>Na prompt fornecida em relação à existência de um gerente “Os funcionários podem ou não ocupar o cargo de gerência. sendo que para tal aconteça, devem ser cumpridas certas condições. Apenas é possível a ocupação do cargo de gerente por um funcionário de cada vez. É necessário que o funcionário tenha no mínimo 4 anos de serviço.”, o grupo achava que estava expresso que era necessária a existência de pelo menos um gerente contudo numa segunda leitura verificou-se que tal não era verdade e o erro foi corrigido<<

“Para que um funcionário seja considerado um gerente, ele deve estar associado a um objeto LivrariaFuncionario. Isso é definido pelo método definirGerente().”

>>Apesar de ser referido um método que supostamente define um gerente, não é em qualquer lado falada a condição da necessidade de pelo menos 4 anos de serviço por parte de um funcionário para que possa ocupar este cargo”<<

Cliente tem uma multiplicidade de 0 para N (um cliente pode não fazer compras ou fazer várias).

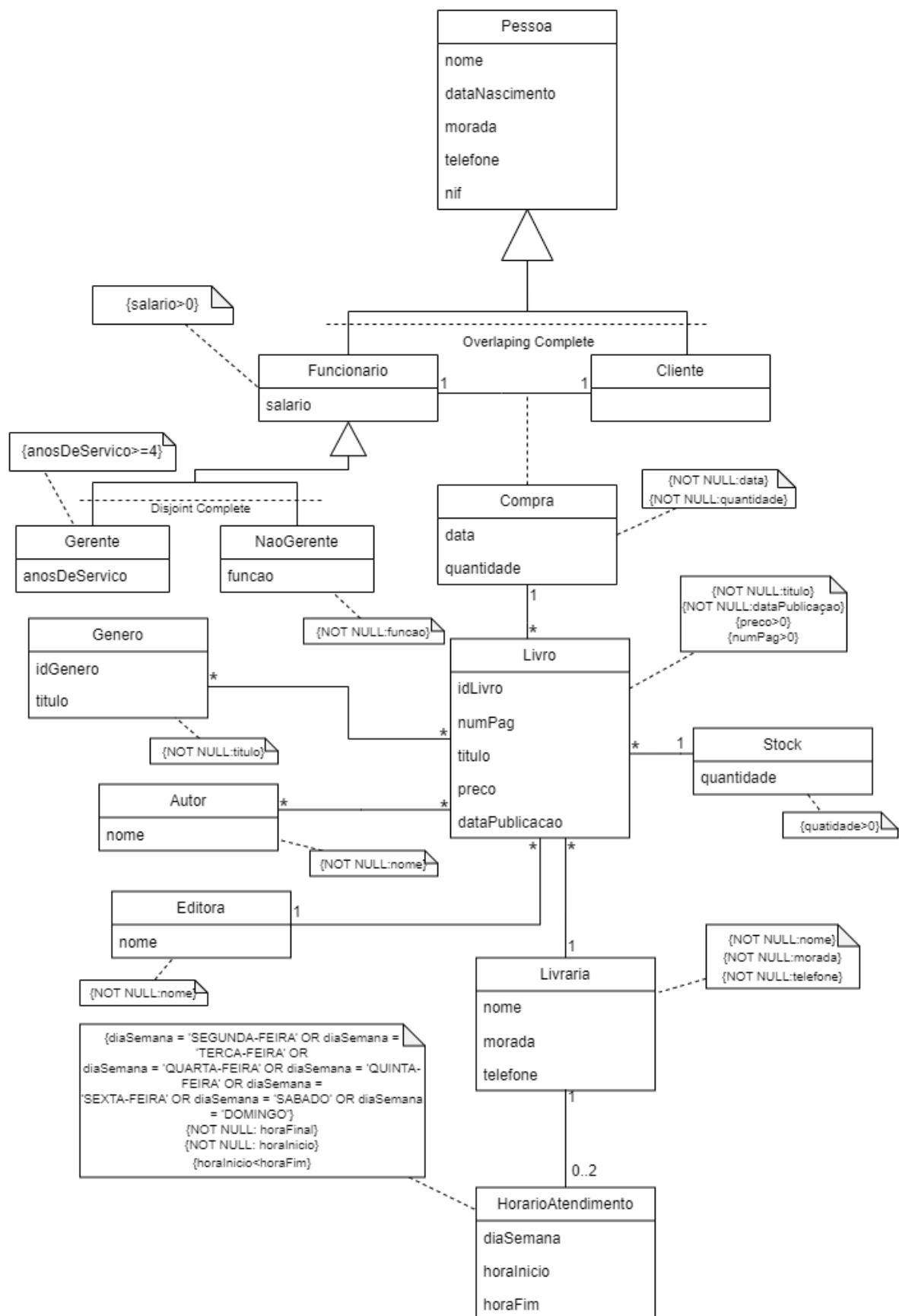
>> No nosso entendimento um cliente tem de fazer pelo menos uma compra para ser cliente.<<

Compra tem uma multiplicidade de 0 para N (uma compra pode ser feita por zero ou mais clientes).

>> Uma compra tem de ser feita por 1 ou mais clientes <<

...

Diagrama UML antes do chat gpt:



Considerações finais:

Na realização do nosso trabalho, se tivéssemos utilizado primeiramente o chatgpt, seria uma ajuda preciosa para o início do projeto como ponto de partida. Foi também muito útil numa fase mais avançada do trabalho como uma “segunda opinião” que nos alertou para erros e faltas de atenção. Especulamos também que se balizássemos melhor o problema, conseguiríamos alcançar resultados mais úteis, no entanto, até que ponto é que proceder desta forma seria assim tão útil? Uma vez que se for para utilizar o resultado que se espera como prompt, mais vale fazê-lo nós próprios.

Acreditamos que o uso das ferramentas de inteligência artificial atualmente ao nosso dispor é uma mais valia mas que ainda têm um longo caminho para andar. Já que possui problemas relativamente à especificidade e taxa de acerto no que toca a pedidos mais complexos feitos pelos utilizadores.