## Um projeto para a unidade curricular de aeda, no 2º ano do mieic



Realizado pelo grupo 8 da turma 3:

* Carlos Vieira; Nº de Estudante: up201606868  
  E-mail: up201606868@fe.up.pt
* João Álvaro Ferreira; Nº de Estudante: up201605592;

E-Mail: up201605592@fe.up.pt

* João Carlos Maduro; Nº de Estudante: up201605219;   
  E-Mail: [up201605219@fe.up.pt](mailto:up201605219@fe.up.pt)

**Data: 15 de Novembro de 2017**

Índice

[Um projeto para a unidade curricular de aeda, no 2º ano do mieic 1](#_Toc502858985)

[**Tema do trabalho** 3](#_Toc502858986)

[**Implementação da solução para a proposta dada** 4](#_Toc502858987)

[Relatório UML 5](#_Toc502858988)

[LISTA DE CASOS DE UTILIZAÇÃO 6](#_Toc502858989)

[**DIFICULDADES SENTIDAS** 7](#_Toc502858990)

[**CONTRIBUIÇÃO DE CADA MEMBRO** 8](#_Toc502858991)

# **Tema do trabalho**

A proposta de trabalho que escolhemos implica o desenvolvimento de software para a gestão de uma empresa que funciona como agente de reservas. A empresa, “Porto Rivers”, faz isto criando estruturas de dados tanto para os clientes como para os fornecedores, facilitando a interação entre ambos o máximo possível.  
  
Esta proposta implica, para além de criar estruturas para guardar informação de clientes e de fornecedores, fazer o mesmo para as ofertas dos fornecedores (que a estes estão associadas) e para as reservas efetuadas pelos clientes. Fazemos isto atendendo aos pormenores indicados para cada fornecedor, diferenciando entre clientes registados ou não (registo este que implica acumulação de pontos que podem ser gastos para receber descontos).

Foi nos exigido que desenvolvêssemos métodos para gerir a informação descrevida utilizando classes, servindo-nos de polimorfismo e herança para estas mesmas classes, estruturas lineares (arrays, vetores, etc.), que passássemos a informação guardada para ficheiros e que qualquer erro do utilizador fosse detetado com uma exceção – algo que fizemos, como será descrito mais a fundo na próxima secção.

# **Implementação da solução para a proposta dada**

O primeiro passo no desenvolvimento do programa foi implementar as classes dos fornecedores, das ofertas, das reservas e dos clientes, sendo que esta última tem como subclasse os clientes reservados. Estas classes interagem constantemente, sendo, por exemplo, um dos membros privados da classe fornecedores um vetor com todas as ofertas que tem disponíveis.

Geramos uma classe time, e fizemos overload aos operadors “<” e “==”, de modo a facilitarmos e tornarmos uniforme o uso de datas e horas nas funções que estas requerem (principalmente no que toca a reservas).

De modo tornar mais direta a manipulação de dados, implementamos também uma class Empresa com usos mais generalistas – tem vetores dos fornecedores, dos clientes e das reservas, e métodos que manipulam estes vetores de acordo com o input do utilizador (que é feito em funções dos menus, que também são métodos desta classe).

Estes vetores são ordenados com algoritmos sort e é feita a pesquisa de elementos através de pesquisa binária.

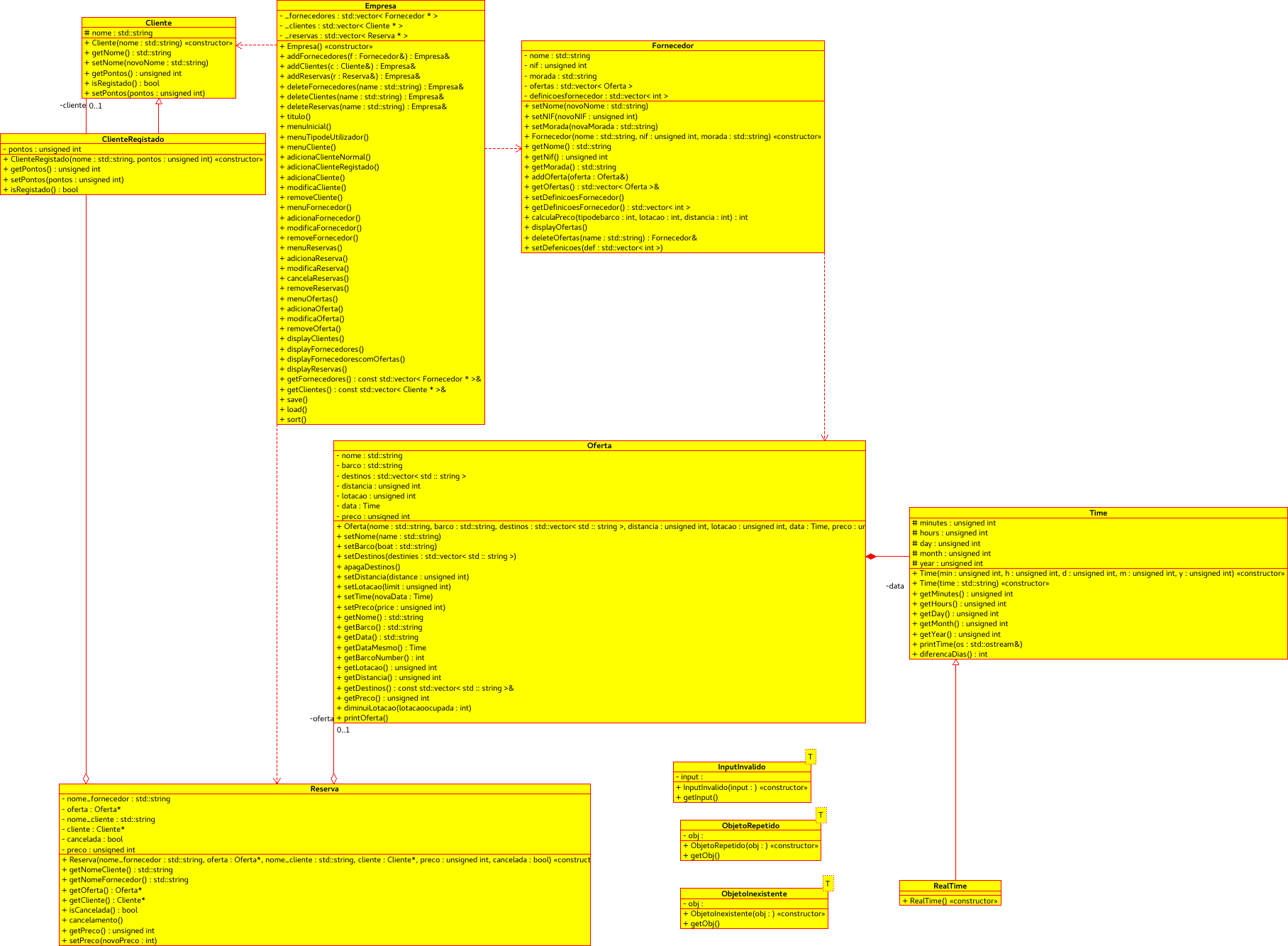
Para além disto, criamos os métodos que passam toda a informação guardada nestas classes para ficheiros de texto individuais, de modo a não perder a informação após o fecho do programa.

Após a estruturação inicial, foram feitos os menus e toda a interface com que o utilizador irá interagir. Foi criado um sub-menu para os métodos de cada classe previamente referida, desde o registo, modificação e eliminação de clientes ou fornecedores, à criação de novas reservas ou adição de ofertas para um fornecedor.

Por fim, implementamos deteção de exceções (com try, throw e catch) nos inputs do utilizador que levariam a uma procura por um objeto com um nome específico, de forma a lidar com a exceção caso este não fosse encontrado pela pesquisa binária.

# Relatório UML

A visualização deste num documento PDF não é de grande qualidade devido ao grande tamanho, logo será também enviado dentro do ZIP em separado.



# LISTA DE CASOS DE UTILIZAÇÃO

Chegando ao menu principal, o utilizador depara-se com quatro opções, a gestão de clientes, fornecedores, ofertas e reservas. Em qualquer uma destas opções, o utilizador poderá criar novos clientes, fornecedores, ofertas ou reservas, apaga-las, modifica-las e, no caso das reservas, cancelá-las.

Para poder criar uma reserva é necessário existirem previamente ofertas e clientes, e para criar uma oferta é necessário que existam previamente fornecedores.

# **DIFICULDADES SENTIDAS**

As principais dificuldades sentidas neste trabalho foram a organização e armazenamento eficiente dos dados que nos competia organizar. Essas dificuldades – principalmente sentidas no início do projeto – levaram mais tarde a algumas complicações desnecessárias em funções que se serviam desses dados. Se tivéssemos feito a estruturação inicial do trabalho de modo mais eficiente, essas complicações não teriam surgido. No entanto, estas dificuldades foram resolvidas e não são sentidas no fim do trabalho.

# **CONTRIBUIÇÃO DE CADA MEMBRO**

Neste trabalho, sentimos que cada membro do grupo investiu aproximadamente o mesmo número de horas de trabalho e de esforço. Apesar de cada um ter tido tarefas mais específicas (que serão enumeradas seguidamente), todos fizeram um pouco de tudo e resolveram erros em todas as partes do trabalho.

A divisão de trabalhos inicial foi:

Carlos Vieira: classes e funções destas, classe Time como um todo

João Carlos Maduro: funções de acessos a ficheiros, desenvolvimento de classes e funções de classes

João Álvaro Ferreira: funções de menu, displays de informação e acesso a dados, criação

No entanto, como já foi dito, após a fase inicial, qualquer um dos membros trabalhou em todo o tipo de funções dentro deste trabalho.