

Licenciatura em

Engenharia de Sistemas

Informáticos

U.C. Programação Orientada a Objetos (POO)

2024/2025

FASE 1

Docente: Luís Ferreira

Aluno: José António da Cunha Alves

Nº 27967

08/11/2024

# Enunciado

## 1 Motivação

Pretende-se que sejam desenvolvidas soluções em C# para problemas reais de complexidade moderada. Serão identificadas classes, definidas estruturas de dados e implementados os principais processos que permitam suportar essas soluções. Pretende-se ainda contribuir para a boa redação de relatórios.

2 Objetivos

• Consolidar conceitos basilares do Paradigma Orientado a Objetos;

• Analisar problemas reais;

• Desenvolver capacidades de programação em C#;

• Potenciar a experiência no desenvolvimento de software;

• Assimilar o conteúdo da Unidade Curricular.

3 Problema a Explorar

(vii) Comércio eletrónico: sistema que permita a gestão de uma loja online. *keywords*: produtos, categorias, garantias, stocks, clientes, campanhas, vendas, marcas.

# Introdução

Este documento é uma descrição do trabalho realizado na primeira fase do trabalho prático da unidade curricular de Programação Orientada a Objetos.

Neste trabalho, é proposto desenvolver um programa que torne possível a gestão de uma loja online, tendo como termos indispensáveis *produtos, garantias, vendas, clientes, categorias, stocks, campanhas* e *marcas*. Assim sendo, é necessário que o programa contemple funções que permitam todo o tipo de operações de gestão, desde criação de ficha de produto/cliente, bem como a gestão dos stocks dos respetivos produtos, datas de fim de garantias, datas de venda, entre outros.

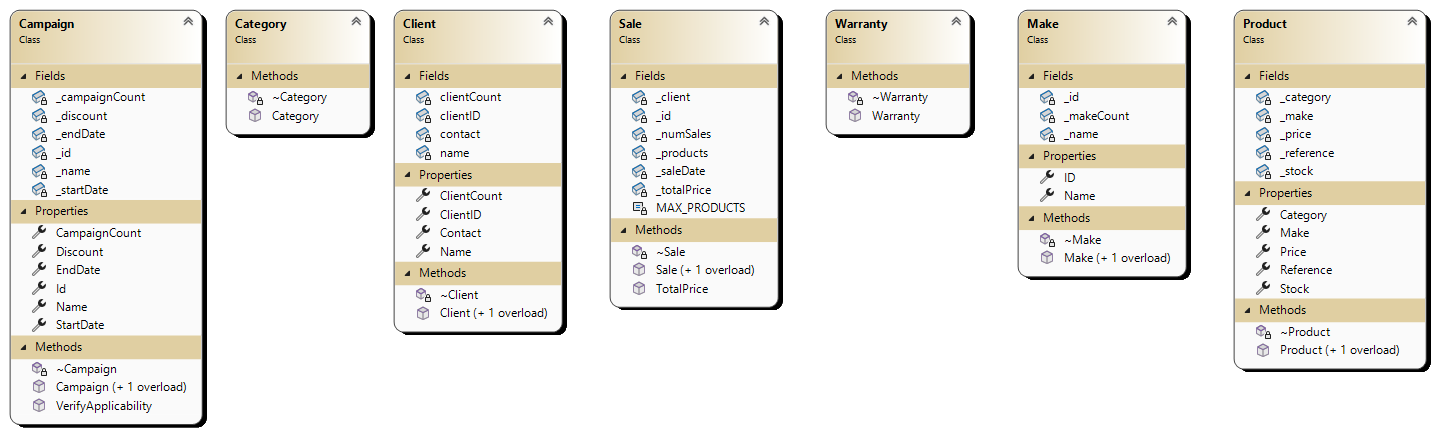
## Revisão de Literatura

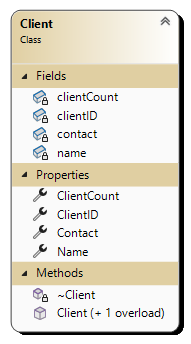
* Programação Orientada a Objetos – Material das aulas;
* Documentação – Doxygen Quick Reference;
* Qualidade do Código – *Clean Code – A Handbook of Agile Software Craftsmanship*; de Robert C. Martin.

Trabalho Desenvolvido

Análise e Modelação

Para permitir a gestão de comércio, foram criadas as seguintes classes:



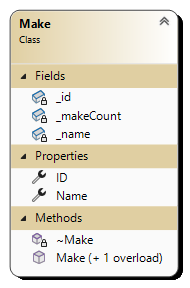
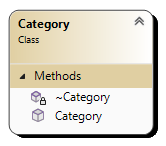
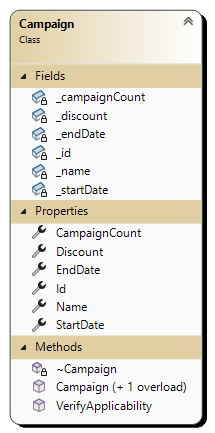


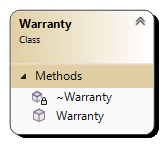
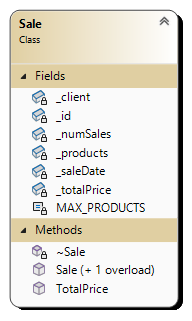
Na classe *Client* temos todos os métodos relacionados com a criação e gestão de clientes. Como atributos temos:

*- clientCount* – é um atributo *static* para guardar o número de clientes da loja.  
*- clientID* – representa o número de cliente.  
*- contact* – contacto do cliente.  
*- name* – nome do cliente.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamenteNa classe *Product* temos os métodos relacionados com a criação e gestão de produtos. Atributos:  
  
- \_*category* – representa a categoria a que o produto pertence.  
- *\_make* – marca do produto.  
- *\_price* – preço do produto.  
- *\_reference* – referência de identificação do produto.  
- \_*stock* – quantidade em armazém.

Na Classe *Make* temos a definição de uma marca e os métodos necessários para a gerir. Atributos:  
  
- *\_id* – número de identificação da marca.  
- *\_makeCount* – contador de marcas registadas no sistema.  
- *\_name* – nome da marca.  
   
  
  
  
  
  
  
  
  
  
Na Classe *Category* temos a definição de categoria de produto e os métodos necessários à sua gestão. Atributos:  
  
- *\_id* – número de identificação da categoria.  
- *\_name* – nome da categoria.  
- *\_catCount* – contador de categorias registadas.  
  
  
  
Na classe *Campaign* temos a definição de campanha/promoção e os métodos necessários para fazer a gestão da mesma. Atributos:  
  
- \_*campaignCount* – contador de campanhas registadas.  
- \_*discount* – valor do desconto, em decimal. (ex: 30% = 0.3)  
- \_*endDate* – data do término da campanha.  
- \_*id* – número da campanha.  
- \_*name* – nome   
- \_*startDate* – data de início da campanha.

  
Na Classe *Sale* podemos encontrar a definição de venda, bem como métodos para gerir as mesmas. Atributos:  
  
- \_*client* – cliente a quem é feita a venda.  
- \_*id* – número da venda.  
- \_*numSales* – contador de vendas.  
- \_*products* – array dos produtos vendidos.  
- \_*saleDate* – data da venda.  
- \_*totalPrice* – preço total da venda.  
- *MAX*\_*PRODUCTS* – constante que define um número máximo de produtos numa venda.  
  
  
  
  
Na Classe *Warranty* podemos encontrar a definição de garantia e os métodos para a sua gestão. Atributos:  
  
- \_*id* – número do processo de garantia.  
- \_*durationMonths* – duração da garantia, em meses.

# Análise de Resultados

O programa resultante é ainda bastante embrionário, visto que há ainda alterações que devem ser feitas. Estas alterações visam a otimização do programa, bem como a tentativa de melhoria da própria implementação, no que à qualidade do código diz respeito.

Exemplos de alterações são:

* remoção da constante *MAX*\_*PRODUCTS* numa venda, que passará pela redefinição de

alguns ciclos.

* Implementação de uma classe própria *Stock*, para facilitar a gestão de mais que um armazém, bem como respeitar o princípio de responsabilidade única.

# Conclusões e Trabalho Futuro