**Gestão de Alojamentos Turísticos**

EST-IPCA Barcelos

LESIPL

Programação Orientada a Objetos

Gilberto Viana Claro

Nº30537

15 de novembro de 2024

Prof. Ernesto Carlos Casanova Ferreira

Ano letivo 2024/2025

Índice

[**Introdução** 3](#_Toc182588331)

[**Objetivos do projeto** 3](#_Toc182588332)

[**Estrutura do Código** 4](#_Toc182588333)

[**Program.cs** 4](#_Toc182588334)

[**Análise dos conceitos de Programação Orientada a Objetos** 5](#_Toc182588335)

[**Conclusão** 6](#_Toc182588336)

# **Introdução**

O objetivo deste trabalho é desenvolver um projeto para a gestão de alojamentos turísticos que permite a gestão de clientes, reservas e alojamentos. Este projeto foi desenvolvido em C#, utilizando conceitos basilares do Paradigma Orientado a Objetos. O projeto foi estruturado em várias classes que representam entidades como Alojamento, Reserva, Pessoas (clientes e administradores) e uma classe de sessão que implementa o padrão Singletonpara gerenciar o estado do utilizador durante a execução do sistema.

# **Objetivos do Projeto**

Os principais objetivos deste projeto foram:

* Criar um projeto em C# para a gestão de alojamentos turísticos.
* Aplicar conceitos fundamentais da Programação Orientada a Objetos (POO): abstração, herança, polimorfismo e encapsulamento (4 pilares).
* Permitir criar e gerir clientes e reservas através de uma interface.
* Implementar o padrão de design Singleton para gerir a sessão do utilizador.

# **Estrutura do Código**

**Classes principais**

Foram criadas várias classes:

* **Alojamento** - Representa um alojamento disponível para reserva, com propriedades como Id, nome, endereço, capacidade e preço por noite.
* **Reserva** - Representa uma reserva de um alojamento feita por um cliente, com propriedades como Id, DataInício, DataFim, Cliente e Alojamento.
* **Pessoas** - Representa os utilizadores do sistema (utilizadores e administradores) com prioridades como Id, Nome, Email, Telefone e Admin (para identificar o administrador).
* **SessãoUtilizador** - Classe responsável pela gestão da sessão do utilizador, implementada com um Singleton, que armazena dados como IdUtilizador, NomeUtilizador, Email e IsAdmin.

**Abstração e Herança**

Foi aplicada a abstração com a classe Pessoas, que define as propriedades comuns a Administrador e Utilizador (nome, email, telefone, etc.). A herança permite que as classes Administrador e Utilizador herdem estas propriedades e comportamentos da classe Pessoas, garantindo a reutilização de código e organização do sistema.

# **Program.cs**

# A execução do programa ocorre no ficheiro Program.cs, onde é criada uma instância de Form1 que exibe a interface de login. O sistema permite que o utilizador insira um email e senha, validando se as credenciais correspondem a um utilizador registado.

# **Análise dos conceitos de Programação Orientada a Objetos**

O projeto utiliza os quatro pilares de POO da seguinte forma:

**Herança**

A herança é aplicada na classe Pessoa, que é a classe base para Utilizador e Administrador. Com isso, o utilizador e administrador compartilham as propriedades Id, nome, email e telefone, sem a necessidade de redefini-las, aumentando a reutilização e a organização do código.

**Encapsulamento**

O encapsulamento é aplicado ao proteger dados como os IDs das reservas e as propriedades dos utilizadores de acessos não controlados. As propriedades da classe Reserva, como o campo ultimoId, são privadas, garantindo que os valores sejam manipulados apenas dentro da classe através de métodos específicos.

**Abstração**

A abstração é aplicada na classe Pessoas, que define uma estrutura comum para Administrador e Utilizador, deixando os detalhes de implementação específicos para essas classes. Isso permite que o código trate as duas entidades de forma genérica e as instâncias sejam diferenciadas com base no seu tipo.

**Singleton**

A implementação do Singleton foi realizada na classe SessaoUtilizador, que garante que apenas uma instância dessa classe seja criada e utilizada ao longo da execução do sistema. A instância única é acessada pelo método ObterInstancia, que garante a consistência dos dados da sessão do utilizador.

**Melhorias futuras**

* **Adicionar Validação de Dados**: Garantir que os dados, como data de início e data do fim, sejam válidos antes de criar uma reserva.
* **Adicionar Funcionalidades**: Incluir funcionalidades para calcular descontos.

# **Conclusão**

Este projeto de gestão de alojamentos turísticos permitiu aplicar os principais conceitos da Programação Orientada a Objetos, como herança, encapsulamento e abstração. Através de uma estrutura modular em C#, foi possível criar um projeto organizado e escalável, com classes bem definidas para utilizadores, administradores, alojamentos e reservas. A utilização do padrão Singleton na gestão da sessão garantiu a consistência dos dados ao longo da aplicação. A implementação demonstra como a POO facilita a manutenção e extensão do código, fornecendo uma base sólida para futuras melhorias e funcionalidades adicionais.