INSTITUTO POLITÉCNICO DO CÁVADO E DO AVE
ESCOLA SUPERIOR DE TECNOLOGIA

IPCA INSTITUTO POLITÉCNICO DO CÁVADO E DO AVE

Gestão de Alojamentos Turísticos

Aluno
João Pedro Júnior Barbosa 27964
Professor
Luís Gonzaga Martins Ferreira
Projeto apresentado
ao Instituto Politécnico do Cávado e do Ave

Novembro,2024

Índice

Introdução

A gestão eficiente de reservas e check-ins é essencial para o sucesso de qualquer estabelecimento de alojamento, como hotéis, pousadas ou plataformas de aluguel de temporada. Nesse contexto, o presente projeto foi concebido com o objetivo de desenvolver um sistema robusto e automatizado para atender a essas necessidades, utilizando os princípios da Programação Orientada a Objetos (POO) em C#.

O sistema implementado integra funcionalidades essenciais, como o registo e a gestão de reservas, verificação de disponibilidade, execução de check-ins, e armazenamento de informações de forma segura. Baseado em uma arquitetura modular e escalável, o projeto segue boas práticas de desenvolvimento, como a separação por camadas (N-tier architecture), a documentação detalhada do código, e a aplicação de interfaces e abstrações para garantir flexibilidade e manutenção.

Outro ponto fundamental foi a inclusão de funcionalidades como persistência de dados em ficheiros binários e o uso de listas em substituição a arrays, promovendo maior eficiência na manipulação de elementos. Adicionalmente, o sistema implementa validações rigorosas, assegurando que operações como reservas e check-ins sejam realizadas apenas sob condições adequadas, refletindo práticas comuns no setor hoteleiro.

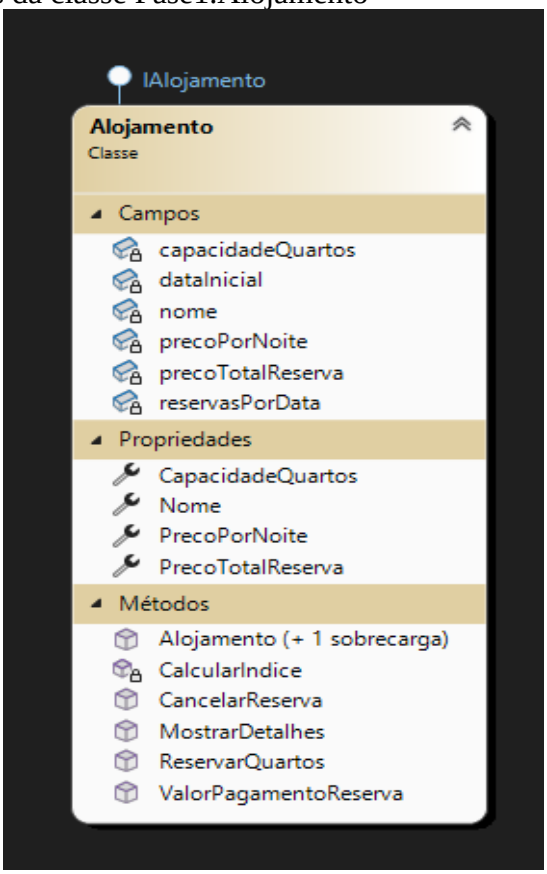
Por fim, o desenvolvimento deste sistema teve como foco a criação de uma solução que não apenas responda às necessidades atuais de gestão de alojamentos, mas que também seja adaptável a futuros desafios e expansões, como a integração com plataformas externas ou a inclusão de novas funcionalidades analíticas.

Documentação da classe

Referência à classe Fase1.Alojamento

Purpose: Created by: João Barbosa Created on: 06/11/2024 13:49:24.

Diagrama de heranças da classe Fase1.Alojamento



Membros públicos

- **Alojamento** (string nome, decimal precoPorNoite, int capacidadeQuartos, DateTime dataInicial)
  Inicializa uma nova instância da classe Alojamento.

- **Alojamento** (string nome, decimal precoPorNoite, int capacidadeQuartos)
- void **MostrarDetalhes** (DateTime dataEntrada, DateTime dataSaida)
  Exibe detalhes da disponibilidade de quartos no alojamento na data de inicio.

- void **ValorPagamentoReserva** (DateTime dataEntrada, DateTime dataSaida)
  Exibe o preço a pagar pela estadia.

- short **ReservarQuartos** (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  Reserva uma quantidade de quartos para o intervalo de datas especificado e calcula o total a pagar.

- bool **CancelarReserva** (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)

Cancela uma quantidade de quartos reservados para o intervalo de datas especificado.

Propriedades

- string **Nome** `[get, set]`
  Obtém ou define o nome do alojamento.

- decimal **PrecoPorNoite** `[get, set]`
  Obtém ou define o preço por noite.

- int **CapacidadeQuartos** `[get, set]`
  Obtém ou define a capacidade de quartos do alojamento.

- decimal? **PrecoTotalReserva** `[get, set]`
  Obtém ou define o preço total da reserva.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 06/11/2024 13:49:24.

Representa um alojamento com capacidade para reserva de quartos.

Documentação dos Construtores & Destrutor

Fase1.Alojamento.Alojamento (string *nome*, decimal *precoPorNoite*, int *capacidadeQuartos*, DateTime *dataInicial*)

Inicializa uma nova instância da classe Alojamento.

Parâmetros

nome	Nome do alojamento.
precoPorNoite	Preço por noite do alojamento.
capacidadeQuartos	Capacidade total de quartos.
dataInicial	Data inicial de controle das reservas.

Fase1.Alojamento.Alojamento (string *nome*, decimal *precoPorNoite*, int *capacidadeQuartos*)

Documentação das funções

bool Fase1.Alojamento.CancelarReserva (DateTime *dataEntrada*, DateTime *dataSaida*, int *quantidadeQuartos*)

Cancela uma quantidade de quartos reservados para o intervalo de datas especificado.

Parâmetros

dataEntrada	Data de início do cancelamento.
dataSaida	Data de término do cancelamento.
quantidadeQuartos	Quantidade de quartos a cancelar.

Retorna

Retorna `true` se o cancelamento foi bem-sucedido; caso contrário, `false` .

Implementa **Fase1.IAlojamento** (*p. 16*).

void Fase1.Alojamento.MostrarDetalhes (DateTime *dataEntrada*, DateTime *dataSaida*)

Exibe detalhes da disponibilidade de quartos no alojamento na data de inicio.

Parâmetros

dataEntrada	Data de início.
dataSaida	Data de término.

Implementa **Fase1.IAlojamento** (*p. 17*).

short Fase1.Alojamento.ReservarQuartos (DateTime *dataEntrada*, DateTime *dataSaida*, int *quantidadeQuartos*)

Reserva uma quantidade de quartos para o intervalo de datas especificado e calcula o total a pagar.

Parâmetros

dataEntrada	Data de início da reserva.
dataSaida	Data de término da reserva.
quantidadeQuartos	Quantidade de quartos a reservar.

Retorna

Retorna `true` se a reserva foi bem-sucedida; caso contrário, `false` .

Implementa **Fase1.IAlojamento** (*p. 17*).

void Fase1.Alojamento.ValorPagamentoReserva (DateTime *dataEntrada*, DateTime *dataSaida*)

Exibe o preço a pagar pela estadia.

Parâmetros

dataEntrada	Data de início.
dataSaida	Data de término.

Documentação das propriedades

int Fase1.Alojamento.CapacidadeQuartos `[get], [set]`

Obtém ou define a capacidade de quartos do alojamento.

string Fase1.Alojamento.Nome `[get]` , `[set]`

Obtém ou define o nome do alojamento.

decimal Fase1.Alojamento.PrecoPorNoite `[get]` , `[set]`

Obtém ou define o preço por noite.

decimal? Fase1.Alojamento.PrecoTotalReserva `[get]` , `[set]`

Obtém ou define o preço total da reserva.

O preço total da reserva como um valor decimal. Pode ser `null` se o preço não for especificado.

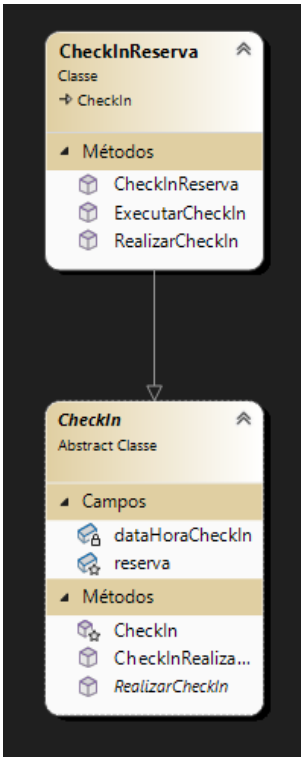
A documentação para esta classe foi gerada a partir do seguinte ficheiro:

Alojamento.cs

Referência à classe Fase1.CheckIn

Purpose: Created by: João Barbosa Created on: 12/11/2024 15:20:05.

Diagrama de heranças da classe Fase1.CheckIn



Membros públicos

- bool **RealizarCheckIn** ()
  Realiza o check-in para a reserva. Este método deve ser implementado pelas classes derivadas.

- bool **CheckInRealizado** ()
  Verifica se o check-in foi realizado.

Membros protegidos

- **CheckIn** (**Reserva reserva**)
  Inicializa uma nova instância da classe CheckIn com a reserva especificada.

Atributos Protegidos

- **Reserva reserva**
  Reserva associada ao check-in.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 12/11/2024 15:20:05.

Classe abstrata que representa o processo de check-in para uma reserva.

Documentação dos Construtores & Destrutor

Fase1.CheckIn.CheckIn (Reserva *reserva*) [protected]

Inicializa uma nova instância da classe CheckIn com a reserva especificada.

Parâmetros

reserva	A reserva associada ao check-in.

Documentação das funções

bool Fase1.CheckIn.CheckInRealizado ()

Verifica se o check-in foi realizado.

Retorna

Retorna `true` se o check-in foi realizado; caso contrário, `false` .

bool Fase1.CheckIn.RealizarCheckIn () [abstract]

Realiza o check-in para a reserva. Este método deve ser implementado pelas classes derivadas.

Documentação dos dados membro

Reserva Fase1.CheckIn.reserva [protected]

Reserva associada ao check-in.

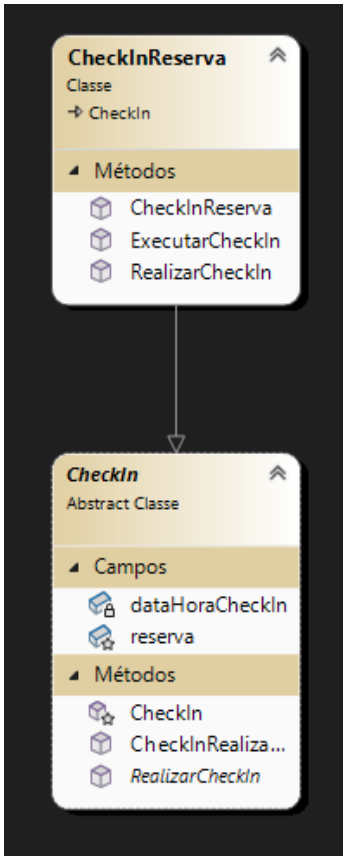
A documentação para esta classe foi gerada a partir do seguinte ficheiro:

CheckIn.cs

Referência à classe Fase1.CheckInReserva

Purpose: Created by: João Barbosa Created on: 12/11/2024 15:39:15.

Diagrama de heranças da classe Fase1.CheckInReserva



Membros públicos

- **CheckInReserva** (**Reserva reserva**)
  Inicializa uma nova instância da classe CheckInReserva com a reserva especificada.

- override bool **RealizarCheckIn** ()
  Tenta realizar o check-in para a reserva e retorna um valor indicando o sucesso.

- void **ExecutarCheckIn** ()
  Executa o check-in e exibe mensagens apropriadas com base no resultado do check-in.

Membros públicos herdados de Fase1.CheckIn

- bool **RealizarCheckIn** ()
  Realiza o check-in para a reserva. Este método deve ser implementado pelas classes derivadas.

- bool **CheckInRealizado** ()
  Verifica se o check-in foi realizado.

Outros membros herdados

Membros protegidos herdados de Fase1.CheckIn

- **CheckIn** (**Reserva reserva**)
  Inicializa uma nova instância da classe CheckIn com a reserva especificada.

Atributos Protegidos herdados de Fase1.CheckIn

- **Reserva reserva**
  Reserva associada ao check-in.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 12/11/2024 15:39:15.

Classe concreta que representa o check-in de uma reserva específica. Herda da classe abstrata CheckIn.

Documentação dos Construtores & Destrutor

Fase1.CheckInReserva.CheckInReserva (Reserva *reserva*)

Inicializa uma nova instância da classe CheckInReserva com a reserva especificada.

Parâmetros

reserva	A reserva associada ao check-in.

Documentação das funções

void Fase1.CheckInReserva.ExecutarCheckIn ()

Executa o check-in e exibe mensagens apropriadas com base no resultado do check-in.

override bool Fase1.CheckInReserva.RealizarCheckIn ()

Tenta realizar o check-in para a reserva e retorna um valor indicando o sucesso.

Retorna

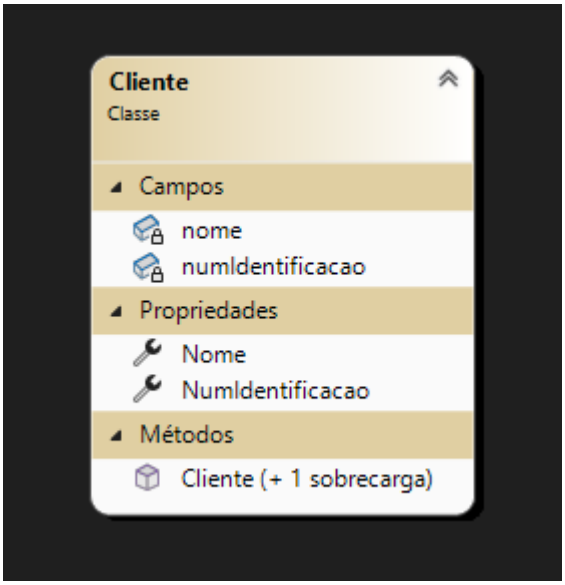
Retorna `true` se o check-in for realizado com sucesso; caso contrário, `false` .

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

CheckInReserva.cs

Referência à classe Fase1.Cliente

Purpose: Created by: João Barbosa Created on: 06/11/2024 14:36:00.



Membros públicos

- **Cliente** ()
  Construtor padrão da classe Cliente. Inicializa o nome como uma string vazia e o número de identificação como "0".

- **Cliente** (string nome, string numIdentificacao)
  Construtor com parâmetros da classe Cliente.

Propriedades

- string **Nome** `[get, set]`
  Obtém ou define o nome do cliente.

- string **NumIdentificacao** `[get, set]`
  Obtém ou define o número de identificação do cliente.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 06/11/2024 14:36:00.

Classe que representa um cliente.

Documentação dos Construtores & Destrutor

Fase1.Cliente.Cliente ()

Construtor padrão da classe Cliente. Inicializa o nome como uma string vazia e o número de identificação como "0".

Fase1.Cliente.Cliente (string *nome*, string *numIdentificacao*)

Construtor com parâmetros da classe Cliente.

Parâmetros

nome	Nome do cliente.
numIdentificacao	Número de identificação do cliente.

Documentação das propriedades

string Fase1.Cliente.Nome `[get], [set]`

Obtém ou define o nome do cliente.

string Fase1.Cliente.NumIdentificacao `[get], [set]`

Obtém ou define o número de identificação do cliente.

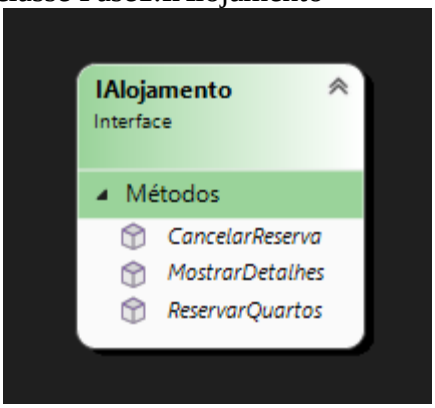
A documentação para esta classe foi gerada a partir do seguinte ficheiro:

Cliente.cs

Referência ao interface Fase1.IAlojamento

Purpose: Created by: João Barbosa Created on: 06/11/2024 14:19:30.

Diagrama de heranças da classe Fase1.IAlojamento



Membros públicos

- void **MostrarDetalhes** (DateTime dataEntrada, DateTime dataSaida)
  Exibe detalhes da disponibilidade de quartos no alojamento entre as datas especificadas.

- short **ReservarQuartos** (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  Realiza a reserva de uma quantidade de quartos no alojamento para o intervalo de datas especificado.

- bool **CancelarReserva** (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  Cancela uma quantidade de quartos reservados no alojamento para o intervalo de datas especificado.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 06/11/2024 14:19:30.

Interface que define as operações básicas para gerenciamento de um alojamento.

Documentação das funções

bool Fase1.IAlojamento.CancelarReserva (DateTime *dataEntrada*, DateTime *dataSaida*, int *quantidadeQuartos*)

Cancela uma quantidade de quartos reservados no alojamento para o intervalo de datas especificado.

Parâmetros

dataEntrada	Data de início do cancelamento.
dataSaida	Data de término do cancelamento.
quantidadeQuartos	Número de quartos a serem cancelados.

Retorna

Retorna `true` se o cancelamento foi realizado com sucesso; caso contrário, `false` .

Implementado em **Fase1.Alojamento** (*p. 7*).

void Fase1.IAlojamento.MostrarDetalhes (DateTime *dataEntrada*, DateTime *dataSaida*)

Exibe detalhes da disponibilidade de quartos no alojamento entre as datas especificadas.

Parâmetros

dataEntrada	Data de início para verificar disponibilidade.
dataSaida	Data de término para verificar disponibilidade.

Implementado em **Fase1.Alojamento** (*p. 8*).

short Fase1.IAlojamento.ReservarQuartos (DateTime *dataEntrada*, DateTime *dataSaida*, int *quantidadeQuartos*)

Realiza a reserva de uma quantidade de quartos no alojamento para o intervalo de datas especificado.

Parâmetros

dataEntrada	Data de início da reserva.
dataSaida	Data de término da reserva.
quantidadeQuartos	Número de quartos a serem reservados.

Retorna

Retorna `true` se a reserva foi realizada com sucesso; caso contrário, `false` .

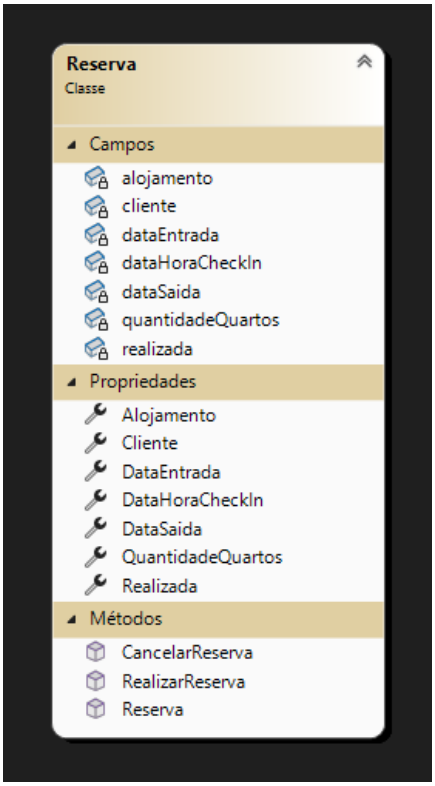
Implementado em **Fase1.Alojamento** (*p. 8*).

A documentação para este interface foi gerada a partir do seguinte ficheiro:

IAlojamento.cs

Referência à classe Fase1.Reserva

Purpose: Created by: João Barbosa Created on: 06/11/2024 16:00:13.



Membros públicos

- **Reserva** (**Cliente** cliente, **Alojamento** alojamento, DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  Inicializa uma nova instância da classe Reserva.

- int **RealizarReserva** ()
  Realiza a reserva no alojamento e marca como realizada.

- void **CancelarReserva** ()
  Cancela a reserva no alojamento.

Propriedades

- **Cliente Cliente** `[get, set]`
  Obtém ou define o cliente que fez a reserva.

- **Alojamento Alojamento** `[get, set]`
  Obtém ou define o alojamento da reserva.

- DateTime **DataEntrada** `[get, set]`
  Obtém ou define a data de entrada na reserva.

- DateTime **DataSaida** `[get, set]`

Obtém ou define a data de saída na reserva.

- int **QuantidadeQuartos** `[get, set]`
  Obtém ou define a quantidade de quartos reservados.

- bool **Realizada** `[get, set]`
  Obtém ou define um valor indicando se a reserva foi realizada.

- DateTime? **DataHoraCheckIn** `[get, set]`
  Obtém ou define a data e hora do check-in, se realizado.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 06/11/2024 16:00:13.

Representa uma reserva de quartos em um alojamento feita por um cliente.

Documentação dos Construtores & Destrutor

Fase1.Reserva.Reserva (Cliente *cliente*, Alojamento *alojamento*, DateTime *dataEntrada*, DateTime *dataSaida*, int *quantidadeQuartos*)

Inicializa uma nova instância da classe Reserva.

Parâmetros

cliente	Cliente que fez a reserva.
alojamento	Alojamento onde será feita a reserva.
dataEntrada	Data de entrada na reserva.
dataSaida	Data de saída na reserva.
quantidadeQuartos	Quantidade de quartos a serem reservados.

Documentação das funções

void Fase1.Reserva.CancelarReserva ()

Cancela a reserva no alojamento.

int Fase1.Reserva.RealizarReserva ()

Realiza a reserva no alojamento e marca como realizada.

Retorna

Retorna `true` se a reserva foi bem-sucedida; caso contrário, `false` .

Documentação das propriedades

Alojamento Fase1.Reserva.Alojamento `[get]`, `[set]`

Obtém ou define o alojamento da reserva.

Cliente Fase1.Reserva.Cliente `[get]`, `[set]`

Obtém ou define o cliente que fez a reserva.

DateTime Fase1.Reserva.DataEntrada `[get]`, `[set]`

Obtém ou define a data de entrada na reserva.

DateTime? Fase1.Reserva.DataHoraCheckIn `[get]`, `[set]`

Obtém ou define a data e hora do check-in, se realizado.

DateTime Fase1.Reserva.DataSaida `[get]`, `[set]`

Obtém ou define a data de saída na reserva.

int Fase1.Reserva.QuantidadeQuartos `[get]`, `[set]`

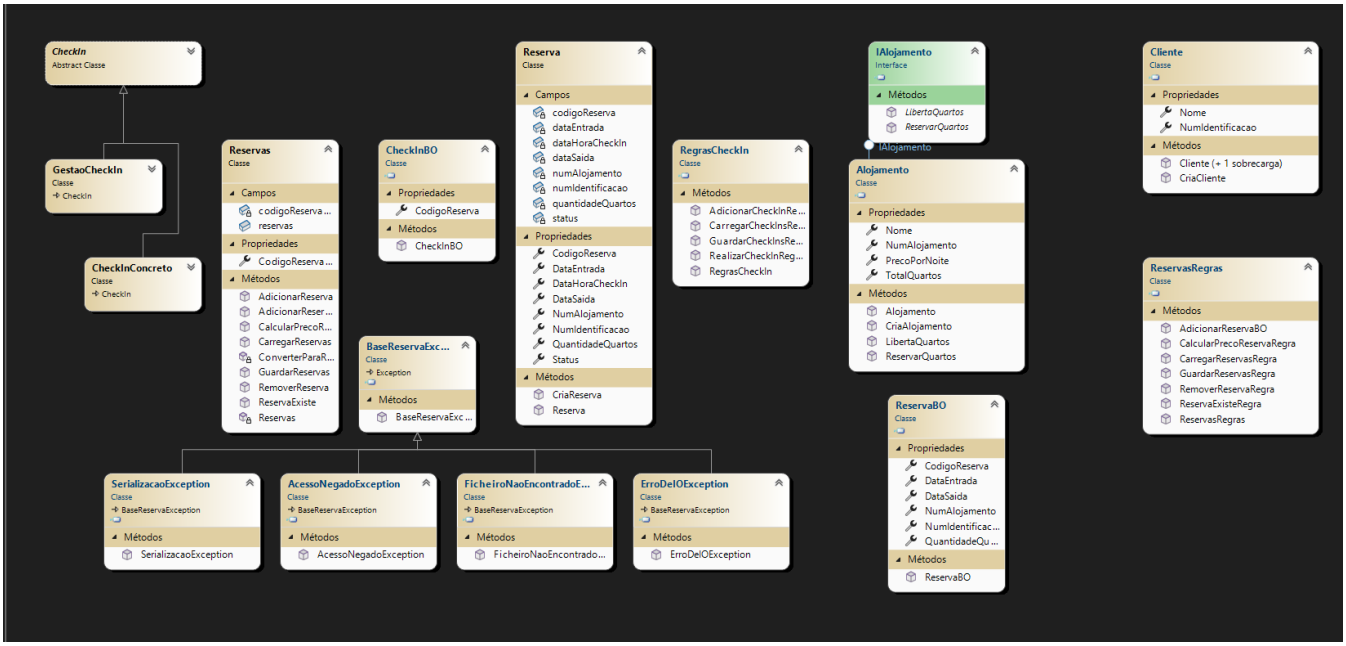
Obtém ou define a quantidade de quartos reservados.

bool Fase1.Reserva.Realizada `[get]`, `[set]`

Obtém ou define um valor indicando se a reserva foi realizada.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:
Reserva.c

2ª Fase


CheckIn
Abstract Classe
GestaoCheckIn
Classe
CheckIn
CheckInConcreto
Classe
CheckIn
Reservas
Classe
Campos
codigoReserva...
reservas
Propriedades
CodigoReserva...
Métodos
AdicionarReserva
AdicionarReser...
CalcularPrecoR...
CarregarReservas
ConverterParaR...
GuardarReservas
RemoverReserva
ReservaExiste
Reservas
CheckInBO
Classe
Propriedades
CodigoReserva
Métodos
CheckInBO
BaseReservaExc...
Classe
Exception
Métodos
BaseReservaExc...
Reserva
Classe
Campos
codigoReserva
dataEntrada
dataHoraCheckIn
dataSaida
numAlojamento
numIdentificacao
quantidadeQuartos
status
Propriedades
CodigoReserva
DataEntrada
DataHoraCheckIn
DataSaida
NumAlojamento
NumIdentificacao
QuantidadeQuartos
Status
Métodos
CriaReserva
Reserva
RegrasCheckIn
Classe
Métodos
AdicionarCheckInRe...
CarregarCheckInsRe...
GuardarCheckInsRe...
RealizarCheckInReg...
RegrasCheckIn
IAlojamento
Interface
Métodos
LibertaQuartos
ReservarQuartos
IAlojamento
Alojamento
Classe
Propriedades
Nome
NumAlojamento
PrecoPorNoite
TotalQuartos
Métodos
Alojamento
CriaAlojamento
LibertaQuartos
ReservarQuartos
Cliente
Classe
Propriedades
Nome
NumIdentificacao
Métodos
Cliente (+ 1 sobrecarga)
CriaCliente
ReservasRegras
Classe
Métodos
AdicionarReservaBO
CalcularPrecoReservaRegra
CarregarReservasRegra
GuardarReservasRegra
RemoverReservaRegra
ReservaExisteRegra
ReservasRegras
ReservaBO
Classe
Propriedades
CodigoReserva
DataEntrada
DataSaida
NumAlojamento
NumIdentificac...
QuantidadeQu...
Métodos
ReservaBO
SerializacaoException
Classe
BaseReservaException
Métodos
SerializacaoException
AcessoNegadoException
Classe
BaseReservaException
Métodos
AcessoNegadoException
FicheiroNaoEncontradoE...
Classe
BaseReservaException
Métodos
FicheiroNaoEncontrado...
ErroDeIOException
Classe
BaseReservaException
Métodos
ErroDeIOException

Documentação da classe

Referência à classe trataProblemas.AcessoNegadoException

Membros públicos

- **AcessoNegadoException** ()
  *Inicializa uma nova instância da classe **AcessoNegadoException** com um código de erro padrão (-1).*

Membros públicos herdados de trataProblemas.BaseReservaException

- **BaseReservaException** (string codigoErro)
  *Inicializa uma nova instância da classe **BaseReservaException** com o código de erro especificado.*

Descrição detalhada

Exceção lançada quando o acesso é negado.

Documentação dos Construtores & Destrutor

trataProblemas.AcessoNegadoException.AcessoNegadoException ()

Inicializa uma nova instância da classe **AcessoNegadoException** com um código de erro padrão (-1).

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- trataProblemas/**trata.cs**

Referência à classe BO.Alojamento

Membros públicos

- **Alojamento** (string nome, decimal precoPorNoite, int totalQuartos, int numAlojamento)
  Construtor para inicializar um alojamento com o nome, preço por noite e capacidade de quartos.

- short **ReservarQuartos** (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  Realiza a reserva de uma quantidade de quartos no alojamento para o intervalo de datas especificado.

- bool **LibertaQuartos** (int quantidade)
  Libera uma quantidade especificada de quartos.

Membros públicos herdados de BO.IAlojamento

Membros públicos estáticos

- static **Alojamento CriaAlojamento** (string nome, decimal precoPorNoite, int totalQuartos, int numAlojamento)
  Cria uma nova instância de um alojamento com as informações especificadas.

Propriedades

- string **Nome** `[get, set]`
  Obtém ou define o nome do alojamento.

- decimal **PrecoPorNoite** `[get, set]`
  Obtém ou define o preço por noite.

- int **TotalQuartos** `[get, set]`
  Obtém ou define a capacidade de quartos do alojamento.

- int **NumAlojamento** `[get, set]`
  *Número de identificação do **Alojamento**.*

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 17/12/2024 17:32:51.

Documentação dos Construtores & Destrutor

BO.Alojamento.Alojamento (string nome, decimal precoPorNoite, int totalQuartos, int numAlojamento)

Construtor para inicializar um alojamento com o nome, preço por noite e capacidade de quartos.

Parâmetros

nome	O nome do alojamento.
precoPorNoite	O preço por noite do alojamento.
capacidadeQuartos	O número máximo de quartos disponíveis no alojamento.

Documentação das funções

static Alojamento BO.Alojamento.CriaAlojamento (string nome, decimal precoPorNoite, int totalQuartos, int numAlojamento) `[static]`

Cria uma nova instância de um alojamento com as informações especificadas.

Parâmetros

nome	Nome do alojamento.
precoPorNoite	Preço por noite do alojamento.
totalQuartos	Quantidade total de quartos disponíveis no alojamento.
numAlojamento	Número de identificação do alojamento.

Retorna

Uma nova instância da classe **Alojamento**.

bool BO.Alojamento.LibertaQuartos (int quantidade)

Liberta uma quantidade especificada de quartos.

Parâmetros

quantidade	Quantidade de quartos a liberar.

Retorna

Retorna `true` se a operação for bem-sucedida; caso contrário, `false` .

Implementa **BO.IAlojamento** (*p. 14*).

short BO.Alojamento.ReservarQuartos (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)

Realiza a reserva de uma quantidade de quartos no alojamento para o intervalo de datas especificado.

Parâmetros

dataEntrada	Data de início da reserva.
dataSaida	Data de término da reserva.

quantidadeQuartos	Número de quartos a serem reservados.

Retorna

Retorna `true` se a reserva foi realizada com sucesso; caso contrário, `false` .

Implementa **BO.IAlojamento** (*p. 15*).

Documentação das propriedades

string BO.Alojamento.Nome `[get], [set]`

Obtém ou define o nome do alojamento.

int BO.Alojamento.NumAlojamento `[get], [set]`

Número de identificação do **Alojamento**.

decimal BO.Alojamento.PrecoPorNoite `[get], [set]`

Obtém ou define o preço por noite.

int BO.Alojamento.TotalQuartos `[get], [set]`

Obtém ou define a capacidade de quartos do alojamento.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- BO/**AlojamentoBO.cs**

Referência à classe trataProblemas.BaseReservaException

Membros públicos

- **BaseReservaException** (string codigoErro)
  *Inicializa uma nova instância da classe **BaseReservaException** com o código de erro especificado.*

Descrição detalhada

Classe base para as exceções do sistema de reservas.

Documentação dos Construtores & Destrutor

trataProblemas.BaseReservaException.BaseReservaException (string codigoErro)

Inicializa uma nova instância da classe **BaseReservaException** com o código de erro especificado.

Parâmetros

codigoErro	Código de erro associado à exceção.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- trataProblemas/**trata.cs**

Referência à classe dados.CheckIn

Membros públicos

- **CheckIn** (int codigoReserva)
  *Construtor da classe **CheckIn**. Inicializa a propriedade CodigoReserva com o valor fornecido e define a propriedade DataHoraCheckIn como null, indicando que o check-in ainda não foi realizado.*

- bool **AdicionarCheckIn** (**CheckIn** checkIn)
  Método abstrato que deve ser implementado para adicionar um check-in.

- bool **GuardarCheckIns** (string caminhoFicheiro)
  Método abstrato que deve ser implementado para guardar os check-ins em um ficheiro.

- bool **CarregarCheckIns** (string caminhoFicheiro)
  Método abstrato que deve ser implementado para carregar os check-ins de um ficheiro.

- bool **CheckInRealizado** ()
  Método abstrato que deve ser implementado para verificar se o check-in foi realizado.

- void **RegistarCheckIn** (DateTime dataHora)
  Método responsável por registrar a data e hora do check-in.

Propriedades

- int **CodigoReserva** `[get, set]`
  Propriedade que representa o código da reserva associada ao check-in.

- DateTime? **DataHoraCheckIn** `[get, set]`
  Propriedade que representa a data e hora em que o check-in foi realizado. Se não houver check-in, o valor será null.

Documentação dos Construtores & Destrutor

dados.CheckIn.CheckIn (int codigoReserva)

Construtor da classe **CheckIn**. Inicializa a propriedade CodigoReserva com o valor fornecido e define a propriedade DataHoraCheckIn como null, indicando que o check-in ainda não foi realizado.

Parâmetros

codigoReserva	Código da reserva associada ao check-in.

Documentação das funções

bool dados.CheckIn.AdicionarCheckIn (CheckIn checkIn) [abstract]

Método abstrato que deve ser implementado para adicionar um check-in.

Parâmetros

checkIn	Objeto do tipo **CheckIn** que será adicionado.

Retorna

Retorna true se o check-in for adicionado com sucesso, caso contrário, retorna false.

bool dados.CheckIn.CarregarCheckIns (string caminhoFicheiro) [abstract]

Método abstrato que deve ser implementado para carregar os check-ins de um ficheiro.

Parâmetros

caminhoFicheiro	Caminho do ficheiro de onde os check-ins serão carregados.

Retorna

Retorna true se os check-ins forem carregados com sucesso, caso contrário, retorna false.

bool dados.CheckIn.CheckInRealizado () [abstract]

Método abstrato que deve ser implementado para verificar se o check-in foi realizado.

Retorna

Retorna true se o check-in foi realizado, caso contrário, retorna false.

bool dados.CheckIn.GuardarCheckIns (string caminhoFicheiro) [abstract]

Método abstrato que deve ser implementado para guardar os check-ins em um ficheiro.

Parâmetros

caminhoFicheiro	Caminho do ficheiro onde os check-ins serão guardados.

Retorna

Retorna true se os check-ins forem guardados com sucesso, caso contrário, retorna false.

void dados.CheckIn.RegistarCheckIn (DateTime dataHora)

Método responsável por registrar a data e hora do check-in.

Parâmetros

dataHora	A data e hora do check-in a ser registrado.

Documentação das propriedades

int dados.CheckIn.CodigoReserva `[get], [set]`

Propriedade que representa o código da reserva associada ao check-in.

DateTime? dados.CheckIn.DataHoraCheckIn `[get], [set]`

Propriedade que representa a data e hora em que o check-in foi realizado. Se não houver check-in, o valor será null.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- dados/**CheckIn.cs**

Referência à classe BO.CheckInBO

Purpose: Created by: João Barbosa Created on: 18/12/2024 16:30:50.

Membros públicos

- **CheckInBO** (int codigoReserva)
  *Construtor que inicializa uma nova instância da classe **CheckInBO** com um código de reserva.*

Propriedades

- int **CodigoReserva** `[get, set]`
  Propriedade que obtém ou define o código da reserva.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 18/12/2024 16:30:50.

Documentação dos Construtores & Destrutor

BO.CheckInBO.CheckInBO (int codigoReserva)

Construtor que inicializa uma nova instância da classe **CheckInBO** com um código de reserva.

Parâmetros

codigoReserva	Código da reserva associada ao check-in.

Documentação das propriedades

int BO.CheckInBO.CodigoReserva`[get], [set]`

Propriedade que obtém ou define o código da reserva.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- BO/**CheckInBO.cs**

Referência à classe dados.CheckInConcreto

Membros públicos

- **CheckInConcreto** (int codigoReserva)
  *Construtor da classe **CheckInConcreto** que inicializa um novo objeto com o código da reserva.*

- **CheckInConcreto** (int codigoReserva, DateTime dataHoraCheckIn)
  *Construtor da classe **CheckInConcreto** que inicializa um novo objeto com o código da reserva e a data/hora do check-in.*

- override bool **AdicionarCheckIn** (**CheckIn** checkIn)
  Adiciona um check-in à lista de check-ins.

- override bool **GuardarCheckIns** (string caminhoFicheiro)
  guarda os check-ins em um ficheiro.

- override bool **CarregarCheckIns** (string caminhoFicheiro)
  Carrega os check-ins a partir de um ficheiro.

- override bool **CheckInRealizado** ()
  Verifica se o check-in foi realizado com base na data e hora do check-in.

Membros públicos herdados de dados.CheckIn

- **CheckIn** (int codigoReserva)
  *Construtor da classe **CheckIn**. Inicializa a propriedade CodigoReserva com o valor fornecido e define a propriedade DataHoraCheckIn como null, indicando que o check-in ainda não foi realizado.*

- bool **AdicionarCheckIn** (**CheckIn** checkIn)
  Método abstrato que deve ser implementado para adicionar um check-in.

- bool **GuardarCheckIns** (string caminhoFicheiro)
  Método abstrato que deve ser implementado para guardar os check-ins em um ficheiro.

- bool **CarregarCheckIns** (string caminhoFicheiro)
  Método abstrato que deve ser implementado para carregar os check-ins de um ficheiro.

- bool **CheckInRealizado** ()
  Método abstrato que deve ser implementado para verificar se o check-in foi realizado.

- void **RegistarCheckIn** (DateTime dataHora)
  Método responsável por registrar a data e hora do check-in.

Outros membros herdados

Propriedades herdados de dados.CheckIn

- int **CodigoReserva** `[get, set]`
  Propriedade que representa o código da reserva associada ao check-in.

- DateTime? **DataHoraCheckIn** `[get, set]`
  Propriedade que representa a data e hora em que o check-in foi realizado. Se não houver check-in, o valor será null.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 18/12/2024 16:11:51.

Documentação dos Construtores & Destrutor

dados.CheckInConcreto.CheckInConcreto (int codigoReserva)

Construtor da classe **CheckInConcreto** que inicializa um novo objeto com o código da reserva.

Parâmetros

codigoReserva	Código da reserva que será associado ao check-in concreto.

dados.CheckInConcreto.CheckInConcreto (int codigoReserva, DateTime dataHoraCheckIn)

Construtor da classe **CheckInConcreto** que inicializa um novo objeto com o código da reserva e a data/hora do check-in.

Parâmetros

codigoReserva	Código da reserva que será associado ao check-in concreto.
dataHoraCheckIn	Data e hora do check-in que será registrado.

Documentação das funções

override bool dados.CheckInConcreto.AdicionarCheckIn (CheckIn checkIn)

Adiciona um check-in à lista de check-ins.

Parâmetros

checkIn	Objeto do tipo **CheckIn** que representa o check-in a ser adicionado.

Retorna

Retorna true, indicando que o check-in foi adicionado com sucesso. A lógica de adição pode ser ajustada conforme necessário.

override bool dados.CheckInConcreto.CarregarCheckIns (string caminhoFicheiro)

Carrega os check-ins a partir de um ficheiro.

Parâmetros

caminhoFicheiro	Caminho do ficheiro de onde os check-ins serão carregados.

Retorna

Retorna true, indicando que os check-ins foram carregados com sucesso. A lógica de carregamento pode ser ajustada conforme necessário.

override bool dados.CheckInConcreto.CheckInRealizado ()

Verifica se o check-in foi realizado com base na data e hora do check-in.

Retorna

Retorna true se o check-in foi realizado (ou seja, se a data/hora do check-in foi definida), caso contrário retorna false.

override bool dados.CheckInConcreto.GuardarCheckIns (string caminhoFicheiro)

guarda os check-ins em um ficheiro.

Parâmetros

caminhoFicheiro	Caminho do ficheiro onde os check-ins serão guadados.

Retorna

Retorna true, indicando que os check-ins foram guardados com sucesso. A lógica de guardar pode ser ajustada conforme necessário.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- dados/**CheckInConcreto.cs**

Referência à classe BO.Cliente

Purpose: Created by: João Barbosa Created on: 17/12/2024 14:32:04.

Membros públicos

- **Cliente** ()
  *Construtor padrão da classe **Cliente**. Inicializa o nome como uma string vazia e o número de identificação como "0".*

- **Cliente** (string nome, int numIdentificacao)
  *Construtor com parâmetros da classe **Cliente**.*

Membros públicos estáticos

- static **Cliente CriaCliente** (string nome, int numIdentificacao)

Propriedades

- string **Nome** `[get, set]`
  Obtém ou define o nome do cliente.

- int **NumIdentificacao** `[get, set]`
  Obtém ou define o número de identificação do cliente.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 17/12/2024 14:32:04.

Documentação dos Construtores & Destrutor

BO.Cliente.Cliente ()

Construtor padrão da classe **Cliente**. Inicializa o nome como uma string vazia e o número de identificação como "0".

BO.Cliente.Cliente (string nome, int numIdentificacao)

Construtor com parâmetros da classe **Cliente**.

Parâmetros

nome	Nome do cliente.
numIdentificacao	Número de identificação do cliente.

Documentação das funções

static Cliente BO.Cliente.CriaCliente (string nome, int numIdentificacao) `[static]`

Documentação das propriedades

string BO.Cliente.Nome `[get], [set]`

Obtém ou define o nome do cliente.

int BO.Cliente.NumIdentificacao `[get], [set]`

Obtém ou define o número de identificação do cliente.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- BO/**ClienteBO.cs**

Referência à classe trataProblemas.ErroDeIOException

Membros públicos

- **ErroDeIOException** ()
  *Inicializa uma nova instância da classe **ErroDeIOException** com um código de erro padrão (-2).*

Membros públicos herdados de trataProblemas.BaseReservaException

- **BaseReservaException** (string codigoErro)
  *Inicializa uma nova instância da classe **BaseReservaException** com o código de erro especificado.*

Descrição detalhada

Exceção lançada devido a um erro de entrada/saída.

Documentação dos Construtores & Destrutor

trataProblemas.ErroDeIOException.ErroDeIOException ()

Inicializa uma nova instância da classe **ErroDeIOException** com um código de erro padrão (-2).

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- trataProblemas/**trata.cs**

Referência à classe trataProblemas.FicheiroNaoEncontradoException

Membros públicos

- **FicheiroNaoEncontradoException** ()
  *Inicializa uma nova instância da classe **FicheiroNaoEncontradoException** com um código de erro padrão (-5).*

Membros públicos herdados de trataProblemas.BaseReservaException

- **BaseReservaException** (string codigoErro)
  *Inicializa uma nova instância da classe **BaseReservaException** com o código de erro especificado.*

Descrição detalhada

Exceção lançada quando um ficheiro não é encontrado.

Documentação dos Construtores & Destrutor

trataProblemas.FicheiroNaoEncontradoException.FicheiroNaoEncontradoException ()

Inicializa uma nova instância da classe **FicheiroNaoEncontradoException** com um código de erro padrão (-5).

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- trataProblemas/**trata.cs**

Referência à classe dados.GestaoCheckIn

Membros públicos

- **GestaoCheckIn** (int codigoReserva)
  *Inicializa uma nova instância da classe **GestaoCheckIn**.*

- bool **AdicionarCheckIn** (**CheckInBO** checkInBO)
  *Adiciona um novo check-in à lista de check-ins após convertê-lo de CheckInBO para **CheckInConcreto**.*

- bool **RealizarCheckIn** (int codigoReserva)
  *Realiza o check-in para a reserva especificada, criando um novo objeto **CheckInConcreto** e adicionando-o à lista de check-ins.*

- override bool **AdicionarCheckIn** (**CheckIn** checkIn)
  Adiciona um check-in à lista de check-ins caso ele já tenha sido realizado.

- override bool **GuardarCheckIns** (string caminhoFicheiro)
  Guarda a lista de check-ins no ficheiro especificado.

- override bool **CarregarCheckIns** (string caminhoFicheiro)
  Carrega a lista de check-ins a partir de um ficheiro.

- override bool **CheckInRealizado** ()
  Verifica se o check-in foi realizado.

Membros públicos herdados de dados.CheckIn

- **CheckIn** (int codigoReserva)
  *Construtor da classe **CheckIn**. Inicializa a propriedade CodigoReserva com o valor fornecido e define a propriedade DataHoraCheckIn como null, indicando que o check-in ainda não foi realizado.*

- bool **AdicionarCheckIn** (**CheckIn** checkIn)
  Método abstrato que deve ser implementado para adicionar um check-in.

- bool **GuardarCheckIns** (string caminhoFicheiro)
  Método abstrato que deve ser implementado para guardar os check-ins em um ficheiro.

- bool **CarregarCheckIns** (string caminhoFicheiro)
  Método abstrato que deve ser implementado para carregar os check-ins de um ficheiro.

- bool **CheckInRealizado** ()
  Método abstrato que deve ser implementado para verificar se o check-in foi realizado.

- void **RegistarCheckIn** (DateTime dataHora)
  Método responsável por registrar a data e hora do check-in.

Atributos Públicos Estáticos

- static List< **CheckIn** > **listaCheckIns** = new List<**CheckIn**>()

Outros membros herdados

Propriedades herdados de dados.CheckIn

- int **CodigoReserva** `[get, set]`
  Propriedade que representa o código da reserva associada ao check-in.

- DateTime? **DataHoraCheckIn** `[get, set]`
  Propriedade que representa a data e hora em que o check-in foi realizado. Se não houver check-in, o valor será null.

Documentação dos Construtores & Destrutor

dados.GestaoCheckIn.GestaoCheckIn (int codigoReserva)

Inicializa uma nova instância da classe **GestaoCheckIn**.

Parâmetros

codigoReserva	O código da reserva associada ao check-in.

Documentação das funções

override bool dados.GestaoCheckIn.AdicionarCheckIn (CheckIn checkIn)

Adiciona um check-in à lista de check-ins caso ele já tenha sido realizado.

Parâmetros

checkIn	Objeto do tipo **CheckIn** a ser adicionado.

Retorna

Retorna `true` se o check-in foi adicionado com sucesso; retorna `false` caso contrário.

bool dados.GestaoCheckIn.AdicionarCheckIn (CheckInBO checkInBO)

Adiciona um novo check-in à lista de check-ins após convertê-lo de CheckInBO para **CheckInConcreto**.

Parâmetros

checkInBO	O objeto CheckInBO que contém os dados do check-in a ser adicionado.

Retorna

Retorna `true` após adicionar o check-in à lista.

override bool dados.GestaoCheckIn.CarregarCheckIns (string caminhoFicheiro)

Carrega a lista de check-ins a partir de um ficheiro.

Parâmetros

caminhoFicheiro	Caminho do ficheiro a ser carregado.

Retorna

Retorna `true` se o carregamento for bem-sucedido.

Excepções

FicheiroNaoEncontradoException	Lançada se o ficheiro não existir.
AcessoNegadoException	Sem permissão para acessar o ficheiro.
ErroDeIOException	Erro ao carregar do ficheiro.
SerializacaoException	Erro durante a desserialização dos dados.
BaseReservaException	Erro genérico com código "-4".

override bool dados.GestaoCheckIn.CheckInRealizado ()

Verifica se o check-in foi realizado.

Retorna

Retorna `true` se o check-in foi realizado; caso contrário, `false` .

override bool dados.GestaoCheckIn.GuardarCheckIns (string caminhoFicheiro)

Guarda a lista de check-ins no ficheiro especificado.

Parâmetros

caminhoFicheiro	Caminho do ficheiro onde os check-ins serão guardados.

Retorna

Retorna `true` se guardar for bem-sucedido.

Excepções

AcessoNegadoException	Sem permissão para acessar o ficheiro.
ErroDeIOException	Erro ao guardar no ficheiro.
SerializacaoException	Erro durante a serialização dos dados.
BaseReservaException	Erro genérico com código "-4".

bool dados.GestaoCheckIn.RealizarCheckIn (int codigoReserva)

Realiza o check-in para a reserva especificada, criando um novo objeto **CheckInConcreto** e adicionando-o à lista de check-ins.

Parâmetros

codigoReserva	Código da reserva para a qual o check-in será realizado.

Retorna

Retorna `true` se o check-in for realizado com sucesso, ou `false` caso contrário.

Documentação dos dados membro

List<CheckIn> dados.GestaoCheckIn.listaCheckIns = new List<CheckIn>() `[static]`

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- dados/**CheckInReserva.cs**

Referência ao interface BO.IAlojamento

Membros públicos

- short **ReservarQuartos** (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  Realiza a reserva de uma quantidade de quartos no alojamento para o intervalo de datas especificado.

- bool **LibertaQuartos** (int quantidade)
  Liberta uma quantidade especificada de quartos.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 17/12/2024 14:32:45.

Documentação das funções

bool BO.IAlojamento.LibertaQuartos (int quantidade)

Liberta uma quantidade especificada de quartos.

Parâmetros

quantidade	Quantidade de quartos a libetar.

Retorna

Retorna `true` se a operação for bem-sucedida; caso contrário, `false` .

Implementado em **BO.Alojamento** (*p. 13*).

short BO.IAlojamento.ReservarQuartos (DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)

Realiza a reserva de uma quantidade de quartos no alojamento para o intervalo de datas especificado.

Parâmetros

dataEntrada	Data de início da reserva.
dataSaida	Data de término da reserva.
quantidadeQuartos	Número de quartos a serem reservados.

Retorna

Retorna `true` se a reserva foi realizada com sucesso; caso contrário, `false` .

Implementado em **BO.Alojamento** (*p. 14*).

A documentação para este interface foi gerada a partir do seguinte ficheiro:

- BO/**IAlojamento.cs**

Referência à classe Regras.RegrasCheckIn

Purpose: Created by: João Barbosa Created on: 18/12/2024 11:23:58.

Membros públicos

- **RegrasCheckIn** ()
  *Construtor da classe **RegrasCheckIn**. Inicializa uma nova instância de GestaoCheckIn.*

- bool **AdicionarCheckInRegra** (**CheckInBO** checkIn)
  Adiciona um check-in à gestão de check-ins.

- bool **GuardarCheckInsRegra** (string caminhoFicheiro)
  Guarda os check-ins em um ficheiro.

- bool **CarregarCheckInsRegra** (string caminhoFicheiro)
  Carrega os check-ins a partir de um ficheiro.

- bool **RealizarCheckInRegras** (int codigoReserva)
  Realiza o check-in para a reserva especificada.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 18/12/2024 11:23:58.

Documentação dos Construtores & Destrutor

Regras.RegrasCheckIn.RegrasCheckIn ()

Construtor da classe **RegrasCheckIn**. Inicializa uma nova instância de GestaoCheckIn.

Documentação das funções

bool Regras.RegrasCheckIn.AdicionarCheckInRegra (CheckInBO checkIn)

Adiciona um check-in à gestão de check-ins.

Parâmetros

checkIn	O check-in a ser adicionado.

Retorna

Retorna `true` se o check-in foi adicionado com sucesso, caso contrário, retorna `false` .

bool Regras.RegrasCheckIn.CarregarCheckInsRegra (string caminhoFicheiro)

Carrega os check-ins a partir de um ficheiro.

Parâmetros

caminhoFicheiro	Caminho do ficheiro de onde os check-ins serão carregados.

Retorna

Retorna `true` se os check-ins foram carregados com sucesso, caso contrário, retorna `false` .

bool Regras.RegrasCheckIn.GuardarCheckInsRegra (string caminhoFicheiro)

Guarda os check-ins em um ficheiro.

Parâmetros

caminhoFicheiro	Caminho do ficheiro onde os check-ins serão guardados.

Retorna

Retorna `true` se os check-ins foram guardados com sucesso, caso contrário, retorna `false` .

bool Regras.RegrasCheckIn.RealizarCheckInRegras (int codigoReserva)

Realiza o check-in para a reserva especificada.

Parâmetros

codigoReserva	Código da reserva para a qual o check-in será realizado.

Retorna

Retorna `true` se o check-in foi realizado com sucesso, caso contrário, retorna `false` .

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- Regras/**RegrasCheckIn.cs**

Referência à classe dados.Reserva

Purpose: Created by: João Barbosa Created on: 17/12/2024 14:36:54.

Membros públicos

- **Reserva** (int numIdentificacao, int numAlojamento, DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  *Inicializa uma nova instância da classe **Reserva**.*

Membros públicos estáticos

- static **Reserva CriaReserva** (int numIdentificacao, int numAlojamento, DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  *Inicializa uma nova instância da classe **Reserva**.*

Propriedades

- int **NumIdentificacao** `[get, set]`
  Obtém ou define o número de identificação do Cliente.

- int **NumAlojamento** `[get, set]`
  Obtém ou define o número de Identificação do Alojamento.

- DateTime **DataEntrada** `[get, set]`
  Obtém ou define a data de entrada na reserva.

- DateTime **DataSaida** `[get, set]`
  Obtém ou define a data de saída na reserva.

- int **QuantidadeQuartos** `[get, set]`
  Obtém ou define a quantidade de quartos reservados.

- DateTime? **DataHoraCheckIn** `[get, set]`
  Obtém ou define a data e hora do check-in, se realizado.

- int **CodigoReserva** `[get, set]`
  Obtém ou define o código da reserva.

- string **Status** `[get, set]`
  Obtém ou define o estado da reserva (ativo, cancelado, expirado)

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 17/12/2024 14:36:54.

Documentação dos Construtores & Destrutor

dados.Reserva.Reserva (int numIdentificacao, int numAlojamento, DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)

Inicializa uma nova instância da classe **Reserva**.

Parâmetros

cliente	Cliente que fez a reserva.
alojamento	Alojamento onde será feita a reserva.
dataEntrada	Data de entrada na reserva.
dataSaida	Data de saída na reserva.
quantidadeQuartos	Quantidade de quartos a serem reservados.

Documentação das funções

static Reserva dados.Reserva.CriaReserva (int numIdentificacao, int numAlojamento, DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos) `[static]`

Inicializa uma nova instância da classe **Reserva**.

Parâmetros

cliente	Pessoa cliente que realizou a reserva.
alojamento	Alojamento onde será realizada a reserva.
dataEntrada	Data prevista para entrada na reserva.
dataSaida	Data prevista para saída da reserva.
quantidadeQuartos	Quantidade de quartos incluídos na reserva.

Documentação das propriedades

int dados.Reserva.CodigoReserva `[get]`, `[set]`

Obtém ou define o código da reserva.

DateTime dados.Reserva.DataEntrada `[get]`, `[set]`

Obtém ou define a data de entrada na reserva.

DateTime? dados.Reserva.DataHoraCheckIn `[get]`, `[set]`

Obtém ou define a data e hora do check-in, se realizado.

DateTime dados.Reserva.DataSaida `[get], [set]`

Obtém ou define a data de saída na reserva.

int dados.Reserva.NumAlojamento `[get], [set]`

Obtém ou define o número de Identificação do Alojamento.

int dados.Reserva.NumIdentificacao `[get], [set]`

Obtém ou define o número de identificação do Cliente.

int dados.Reserva.QuantidadeQuartos `[get], [set]`

Obtém ou define a quantidade de quartos reservados.

string dados.Reserva.Status `[get], [set]`

Obtém ou define o estado da reserva (ativo, cancelado, expirado)

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- dados/**Reserva.cs**

Referência à classe BO.ReservaBO

Membros públicos

- **ReservaBO** (int numIdentificacao, int numAlojamento, DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)
  Inicializa uma nova instância da classe Reserva.

Propriedades

- int **NumIdentificacao** `[get, set]`
  Obtém ou define o número de identificação do cliente.

- int **NumAlojamento** `[get, set]`
  *Obtém ou define o número de identificação do **Alojamento**.*

- DateTime **DataEntrada** `[get, set]`
  Obtém ou define a data de entrada na reserva.

- DateTime **DataSaida** `[get, set]`
  Obtém ou define a data de saída na reserva.

- int **QuantidadeQuartos** `[get, set]`
  Obtém ou define a quantidade de quartos reservados.

- int **CodigoReserva** `[get, set]`
  Obtém ou define o código da reserva.

Documentação dos Construtores & Destrutor

BO.ReservaBO.ReservaBO (int numIdentificacao, int numAlojamento, DateTime dataEntrada, DateTime dataSaida, int quantidadeQuartos)

Inicializa uma nova instância da classe Reserva.

Parâmetros

cliente	**Cliente** que fez a reserva.
alojamento	**Alojamento** onde será feita a reserva.
dataEntrada	Data de entrada na reserva.
dataSaida	Data de saída na reserva.
quantidadeQuartos	Quantidade de quartos a serem reservados.

Documentação das propriedades

int BO.ReservaBO.CodigoReserva `[get]`, `[set]`

Obtém ou define o código da reserva.

DateTime BO.ReservaBO.DataEntrada `[get]`, `[set]`

Obtém ou define a data de entrada na reserva.

DateTime BO.ReservaBO.DataSaida `[get]`, `[set]`

Obtém ou define a data de saída na reserva.

int BO.ReservaBO.NumAlojamento `[get]`, `[set]`

Obtém ou define o número de identificação do **Alojamento**.

int BO.ReservaBO.NumIdentificacao `[get]`, `[set]`

Obtém ou define o número de identificação do cliente.

int BO.ReservaBO.QuantidadeQuartos `[get]`, `[set]`

Obtém ou define a quantidade de quartos reservados.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- BO/**ReservasBO.cs**

Referência à classe dados.Reservas

Membros públicos

- int **AdicionarReservaBO** (**ReservaBO** reservaBO)
  *Adiciona uma nova reserva **BO** à lista, convertendo-a para **dados.Reserva**.*

- int **AdicionarReserva** (**Reserva** reserva)
  Adiciona uma nova reserva à lista.

- bool **RemoverReserva** (int codigoReserva)
  Remove uma reserva pelo código.

- decimal **CalcularPrecoReserva** (int codigoReserva, decimal precoPorNoite)
  Calcula o preço total da reserva.

- bool **ReservaExiste** (int codigoReserva)
  Verifica se uma reserva existe na lista de reservas.

- bool **GuardarReservas** (string caminhoFicheiro)
  Guarda a lista de reservas em um ficheiro binário especificado.

- bool **CarregarReservas** (string caminhoFicheiro)
  Carrega a lista de reservas a partir de um ficheiro binário especificado.

Atributos Públicos Estáticos

- static List< **Reserva** > **reservas** = new List<**Reserva**>()

Propriedades

- int **CodigoReservaAtual** `[get, set]`
  Obtém ou define o código da reserva atual.

Documentação das funções

int dados.Reservas.AdicionarReserva (Reserva reserva)

Adiciona uma nova reserva à lista.

Parâmetros

reserva	**Reserva** a ser adicionada

Retorna

Código da reserva ou -1 se inválida

int dados.Reservas.AdicionarReservaBO (ReservaBO reservaBO)

Adiciona uma nova reserva **BO** à lista, convertendo-a para **dados.Reserva**.

Parâmetros

reservaBO	ReservaBO a ser adicionada.

Retorna

Código da reserva gerado ou -1 se a reserva for inválida.

decimal dados.Reservas.CalcularPrecoReserva (int codigoReserva, decimal precoPorNoite)

Calcula o preço total da reserva.

Parâmetros

codigoReserva	Código da reserva
precoPorNoite	Preço por noite

Retorna

Preço total ou -1 se reserva não for encontrada

bool dados.Reservas.CarregarReservas (string caminhoFicheiro)

Carrega a lista de reservas a partir de um ficheiro binário especificado.

Parâmetros

caminhoFicheiro	O caminho completo do ficheiro de onde as reservas serão carregadas.

Retorna

Retorna `true` se o processo de carregamento for bem-sucedido.

Excepções

FicheiroNaoEncontradoException	Lançada quando o ficheiro especificado não é encontrado.
AcessoNegadoException	Lançada quando não há permissão para acessar o ficheiro.
ErroDeIOException	Lançada quando ocorre um erro de entrada/saída ao acessar o ficheiro.
SerializacaoException	Lançada quando ocorre um erro ao desserializar os dados.
BaseReservaException	Lançada para erros genéricos não específicos (-4).

bool dados.Reservas.GuardarReservas (string caminhoFicheiro)

Guarda a lista de reservas em um ficheiro binário especificado.

Parâmetros

caminhoFicheiro	O caminho completo do ficheiro onde as reservas serão guardadas.

Retorna

Retorna `true` se o processo de guardar for bem-sucedido.

Excepções

AcessoNegadoException	Lançada quando não há permissão para acessar o ficheiro.
ErroDeIOException	Lançada quando ocorre um erro de entrada/saída ao acessar o ficheiro.
SerializacaoException	Lançada quando ocorre um erro ao serializar os dados.
BaseReservaException	Lançada para erros genéricos não específicos (-4).

bool dados.Reservas.RemoverReserva (int codigoReserva)

Remove uma reserva pelo código.

Parâmetros

codigoReserva	Código da reserva

Retorna

True se removida com sucesso, False caso contrário

bool dados.Reservas.ReservaExiste (int codigoReserva)

Verifica se uma reserva existe na lista de reservas.

Parâmetros

codigoReserva	Código da reserva a ser verificada.

Retorna

Retorna `true` se a reserva existir, caso contrário, `false` .

Documentação dos dados membro

List<Reserva> dados.Reservas.reservas = new List<Reserva>() `[static]`

Documentação das propriedades

int dados.Reservas.CodigoReservaAtual `[get], [set]`

Obtém ou define o código da reserva atual.

O código da reserva atual.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- dados/**Reservas.cs**

Referência à classe Regras.ReservasRegras

Purpose: Created by: João Barbosa Created on: 17/12/2024 16:27:39.

Membros públicos

- int **AdicionarReservaBO** (**ReservaBO** reservaBO)
  Adiciona uma nova reserva à lista de reservas, realizando as devidas validações.

- bool **RemoverReservaRegra** (int codigoReserva)
  Remove uma reserva com base no código fornecido.

- decimal **CalcularPrecoReservaRegra** (int codigoReserva, decimal precoPorNoite)
  Calcula o preço total da reserva com base no código da reserva e no preço por noite.

- bool **ReservaExisteRegra** (int codigoReserva)
  Verifica se uma reserva existe com base no código fornecido.

- bool **GuardarReservasRegra** (string caminhoFicheiro)
  Tenta guardar as reservas em um ficheiro especificado.

- bool **CarregarReservasRegra** (string caminhoFicheiro)
  Tenta carregar as reservas de um ficheiro especificado.

Descrição detalhada

Purpose: Created by: João Barbosa Created on: 17/12/2024 16:27:39.

Documentação das funções

int Regras.ReservasRegras.AdicionarReservaBO (ReservaBO reservaBO)

Adiciona uma nova reserva à lista de reservas, realizando as devidas validações.

Parâmetros

reservaBO	Objeto que contém os dados da reserva a ser adicionada.

Retorna

Retorna o código da reserva se adicionada com sucesso, ou um código de erro específico caso ocorra uma falha.

Os seguintes códigos de erro são retornados:

- -1: Reserva inválida (objeto nulo)
- -2: Data de entrada inválida (antes da data atual)
- -3: Intervalo de datas inválido (data de saída anterior à de entrada)
- -4: Quantidade de quartos inválida (menor ou igual a 0)
- -5: Falha ao adicionar a reserva

decimal Regras.ReservasRegras.CalcularPrecoReservaRegra (int codigoReserva, decimal precoPorNoite)

Calcula o preço total da reserva com base no código da reserva e no preço por noite.

Parâmetros

codigoReserva	Código da reserva para a qual o preço será calculado.
precoPorNoite	Preço por noite de hospedagem.

Retorna

Preço total da reserva ou 0 se a reserva não for encontrada.

bool Regras.ReservasRegras.CarregarReservasRegra (string caminhoFicheiro)

Tenta carregar as reservas de um ficheiro especificado.

Parâmetros

caminhoFicheiro	Caminho do ficheiro de onde as reservas serão carregadas.

Retorna

Retorna true se as reservas forem carregadas com sucesso, caso contrário, retorna false.

bool Regras.ReservasRegras.GuardarReservasRegra (string caminhoFicheiro)

Tenta guardar as reservas em um ficheiro especificado.

Parâmetros

caminhoFicheiro	Caminho do ficheiro onde as reservas serão guardadas.

Retorna

Retorna true se as reservas forem guardadas com sucesso, caso contrário, retorna false.

bool Regras.ReservasRegras.RemoverReservaRegra (int codigoReserva)

Remove uma reserva com base no código fornecido.

Parâmetros

codigoReserva	Código da reserva a ser removida.

Retorna

`true` se a reserva for removida com sucesso, caso contrário, `false` .

bool Regras.ReservasRegras.ReservaExisteRegra (int codigoReserva)

Verifica se uma reserva existe com base no código fornecido.

Parâmetros

codigoReserva	Código da reserva a ser verificada.

Retorna

Retorna true se a reserva existir, caso contrário, retorna false.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- Regras/**ReservasRegras.cs**

Referência à classe ReservasTests.ReservasTests

Membros públicos

- void **Setup** ()
  Inicializa uma nova instância da classe Reservas antes de cada teste.

- void **AdicionarReservaValida** ()
  *Testa o método **Reservas.AdicionarReserva** com uma reserva válida. Verifica se o código da reserva retornado é diferente de -1 e corresponde ao valor esperado.*

- void **AdicionarReservaInvalida** ()
  *Testa o método **Reservas.AdicionarReserva** com uma reserva inválida (data de saída anterior à data de entrada). Verifica se o código retornado é -1.*

- void **AdicionarReservaQuartosNegativos** ()
  *Testa o método **Reservas.AdicionarReserva** com uma reserva contendo uma quantidade negativa de quartos. Verifica se o código retornado é -1.*

Documentação das funções

void ReservasTests.ReservasTests.AdicionarReservaInvalida ()

Testa o método **Reservas.AdicionarReserva** com uma reserva inválida (data de saída anterior à data de entrada). Verifica se o código retornado é -1.

void ReservasTests.ReservasTests.AdicionarReservaQuartosNegativos ()

Testa o método **Reservas.AdicionarReserva** com uma reserva contendo uma quantidade negativa de quartos. Verifica se o código retornado é -1.

void ReservasTests.ReservasTests.AdicionarReservaValida ()

Testa o método **Reservas.AdicionarReserva** com uma reserva válida. Verifica se o código da reserva retornado é diferente de -1 e corresponde ao valor esperado.

void ReservasTests.ReservasTests.Setup ()

Inicializa uma nova instância da classe Reservas antes de cada teste.

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- TestesReserva/**testesAdicionarReserva.cs**

Referência à classe trataProblemas.SerializacaoException

Membros públicos

- **SerializacaoException** ()
  *Inicializa uma nova instância da classe **SerializacaoException** com um código de erro padrão (-3).*

Membros públicos herdados de trataProblemas.BaseReservaException

- **BaseReservaException** (string codigoErro)
  *Inicializa uma nova instância da classe **BaseReservaException** com o código de erro especificado.*

Descrição detalhada

Exceção lançada quando ocorre um erro de serialização.

Documentação dos Construtores & Destrutor

trataProblemas.SerializacaoException.SerializacaoException ()

Inicializa uma nova instância da classe **SerializacaoException** com um código de erro padrão (-3).

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

- trataProblemas/**trata.cs**

Alterações realizadas

Ao longo da evolução do projeto, foram realizadas alterações significativas que melhoraram a estrutura, a funcionalidade e a eficiência do sistema. Uma das principais mudanças foi a utilização de bibliotecas, o que permitiu ampliar as funcionalidades do código de forma prática e padronizada, eliminando a necessidade de reimplementar soluções já existentes. Além disso, a adoção da programação por camadas, seguindo a arquitetura N-tier, trouxe uma separação clara entre as responsabilidades do sistema, organizando-o em camadas como apresentação, regras e acesso a dados, além destas ainda existe uma camada BO com as classes simplificadas. Isso facilitou não apenas a manutenção, mas também a escalabilidade e a evolução do projeto.

Outro avanço importante foi a introdução de testes unitários, que permitiram validar partes isoladas do sistema, garantindo maior confiabilidade no código e reduzindo significativamente os erros antes de novas implementações, decidi implementar os testes na função AdicionarReserva. Junto a isso, o tratamento de exceções tornando o programa mais robusto e capaz de lidar com situações inesperadas de forma controlada, além de fornecer mensagens claras para facilitar a resolução de problemas. As exceções foram utilizadas nas funções de guardar e carregar em ficheiros binários, já que foi as funções que achei mais prudente a utilização.

A funcionalidade de guardar e carregar dados em ficheiros binários foi implementada, assegurando a preservação das informações entre diferentes execuções do programa. Essa abordagem proporcionou eficiência e segurança no armazenamento de dados. Por fim, a substituição dos arrays por listas trouxe maior flexibilidade para a manipulação dos elementos, uma vez que as listas permitem operações mais dinâmicas, como inserções e remoções sem limitações de tamanho fixo.

Para além disso, ainda foram alteradas a lógicas de algumas classes que de acordo com o ponto de vista da eficiência não estavam a trabalhar como deviam. Uma alteração foi, por exemplo, deixei de passar objetos nos atributos e passei apenas a passar códigos identificáveis dos objetos. O mesmo aconteceu nas funções que deixaram de receber objetos e passaram a receber códigos como parâmetros como acontece por exemplo com a função de remover reserva.

Essas alterações não apenas melhoram o desempenho geral do sistema, mas também estabeleceram uma base sólida para futuras melhorias, garantindo um código mais organizado, confiável e alinhado às melhores práticas de desenvolvimento.

Principais dificuldades

A minha principal dificuldade, nesta segunda fase, foi sem dúvida a utilização de programação por camadas, uma vez que tive dificuldades em identificar o que deveria ficar em cada camada e posteriormente tive problemas na comunicação entre elas.

Conclusão

A implementação do sistema de gestão de reservas e check-ins para alojamentos superou as expectativas ao atingir os objetivos propostos, resultando em um programa robusto, funcional e alinhado às melhores práticas de desenvolvimento.

A adoção de uma arquitetura orientada a objetos foi essencial para garantir uma separação clara de responsabilidades entre as diferentes partes do sistema, favorecendo a modularidade, a manutenção e a escalabilidade.
Entre os avanços mais notáveis, destacam-se a introdução da programação em camadas, que organiza o sistema de maneira lógica e eficiente, e o uso de testes unitários para validar funcionalidades críticas, como a adição de reservas. Essas práticas aumentaram significativamente a confiabilidade do código, permitindo uma identificação precoce de erros. Adicionalmente, a integração de tratamento de exceções tornou o sistema mais resiliente, capaz de lidar com problemas inesperados de forma controlada, especialmente em operações de entrada e saída de dados.

A substituição de arrays por listas dinamizou a manipulação de dados, enquanto a transição de atributos e parâmetros baseados em objetos para identificadores simplificou a lógica e melhorou a eficiência do sistema. As funcionalidades de persistência de dados em ficheiros binários consolidaram a capacidade de armazenar informações de forma segura e acessível entre diferentes execuções.

Apesar dos desafios encontrados, especialmente relacionados à aplicação da programação em camadas e à comunicação entre elas, essas dificuldades foram superadas com aprendizado e ajustes contínuos. Como resultado, o sistema apresenta uma base sólida e escalável, pronta para evoluir e incorporar novas funcionalidades, como histórico de estadias, integração com sistemas externos e relatórios analíticos.

Em síntese, este trabalho demonstra a importância de um planeamento cuidadoso, do uso de boas práticas de programação orientada a objetos e da aplicação de padrões modernos de desenvolvimento. A solução desenvolvida cumpre o seu propósito de forma eficaz, automatizando operações de reserva e check-in com segurança e precisão, e oferece uma plataforma versátil e confiável para o futuro da gestão de alojamentos.