Sistema de reserva de voos

Docente: Alysson Filgueira Milanez
Discente: Bruno Victor Paiva Da Silva

1. Definição do sistema:

O sistema de reserva de voos é basicamente uma forma de simular como seria um administrador no sistema com permissões perante as suas funcionalidades, onde foi utilizado a linguagem de programação *C#* com suas especificações formais auxiliadas pelo uso do *CodeContracts* que ajuda na criação de contatos formais no código e também houve a utilização da *IDE: Visual Studio* da *Microsoft*, ele o administrador poderia reservar esse voo selecionando a opção que o cliente irá realizar embarque e o nome da pessoa que ele está cadastrando para viajar, o sistema também não permite que usuários com o mesmo nome para o mesmo voo sejam cadastrados pelo administrador.

Além disso é possível também que haja um cancelamento desta reserva indo na opção de cancelamento e fornecendo as credenciais que foram também realizadas para reserva que é o número do voo e o nome do passageiro. Existe a opção de cadastrar um voo, adicionando ele no sistema e a possibilidade também de visualizar os passageiros que estão com reserva e de fazer logout no sistema, onde caso selecione a opção sair o programa encerra de forma instantânea.

O sistema tem o objetivo de facilitar e apresentar um esboço de como funcionaria uma reserva de voos no olhar de quem está organizando ou neste caso gerenciando as informações que serão amplamente modificadas.

2. Requisitos do sistema:

2.1 Requisitos Funcionais:

[RF001] Reservar um voo.

O sistema deve possibilitar que o administrador possa selecionar um voo que esteja disponível, onde é adicionado o nome do usuário para que seja possível realizar a reserva.

PRIORIDADE: (x) Essencial; ( ) Importante; ( ) Desejável.

```
Unset
Contract.Requires(!string.IsNullOrEmpty(passengerName));
Contract.Ensures(passengers.Count == Contract.OldValue(passengers.Count) + 1);
```

Pré-condição: O método *'BookSeat'* exige que o nome do passageiro não seja nulo ou vazio

Pós-condição: Após a execução do método, o número de passageiros no voo aumentará em 1 se houver assento disponível.

[RF002] Cancelar reserva.

O sistema deve permitir o cancelamento da reserva desde que ela já exista no sistema, onde é solicitado o nome do passageiro, se o nome fornecido estiver na lista o voo é cancelado.

PRIORIDADE: (x) Essencial; ( ) Importante; ( ) Desejável.

```
Unset
Contract.Requires(!string.IsNullOrEmpty(passengerName));
Contract.Ensures(passengers.Count == Contract.OldValue(passengers.Count) - 1);
```

Pré-condição: O método *'CancelSeat'* exige que o nome do passageiro não seja nulo ou vazio
Pós-condição: Após a execução do método, o número de passageiros no voo diminui em 1 se se caso o passageiro estiver na lista de passageiros.

[RF003] Adicionar voo.

O sistema deve permitir que os administradores possam adicionar novos voos ao sistema, inserindo as credenciais necessárias para isso, permitindo que seja aceita apenas informações que sejam diferentes das já existentes no sistema.

PRIORIDADE: (x) Essencial; ( ) Importante; ( ) Desejável.

```
Unset
Contract.Requires(!string.IsNullOrEmpty(flightNumber));
Contract.Requires(!string.IsNullOrEmpty(destination));
Contract.Requires(capacity > 0);
Contract.Ensures(flights.Any(f => f.FlightNumber == flightNumber));
```

Pré-condição: O método *'AddFlight'* exige que o número do voo, o destino e a capacidade sejam válidos neste caso não nulos ou vazios, com capacidade > 0.
Pós-condição: Após a execução do método, a lista de voos conterá um novo voo se o número de voo não estiver em uso.

[RF004] Visualizar detalhes de voo.

O sistema deve permitir que as informações do voo possam ser visualizadas com um maior grau de detalhe possível.

PRIORIDADE: ( ) Essencial; (x) Importante; ( ) Desejável.

```
Unset
Contract.Requires(flights != null);
Contract.Requires(flightIndex >= 0 && flightIndex < flights.Count);
Contract.Ensures(Contract.Result<bool>() == true);
```

Pré-condição: O método *'DisplayDetails'* exige que a lista de voos não seja nula e que o índice do voo seja válido

Pós-condição: Será exibido os detalhes do voo após a execução.

[RF005] Visualizar lista de passageiros de um voo.

O sistema deve permitir que seja possível visualizar a lista de todos os passageiros que estão em aguardo para aquele voo.

PRIORIDADE: ( ) Essencial; (x) Importante; ( ) Desejável.

```
Unset
Contract.Requires(flights != null);
Contract.Requires(flightIndex >= 0 && flightIndex < flights.Count);
Contract.Ensures(Contract.Result<bool>() == true);
```

Pré-condição: O método *'ViewPassengers'* exige que a lista de voos não seja nula e que o índice do voo seja válido

Pós-condição: Será exibido os detalhes do voo após a execução..

[RF006] Sair.

O sistema deve permitir a saída do sistema após ser selecionado essa opção.

PRIORIDADE: ( ) Essencial; (x) Importante; ( ) Desejável.

```
Unset
//Não há pré-condição
//Pós-condição: O sistema foi enserrado
```

2.2 Requisitos não Funcionais:

[RNF001] Ambiguidade de usuários.

O sistema não deve permitir que tenham usuários com o mesmo nome.

PRIORIDADE: (X) Essencial; ( ) Importante; ( ) Desejável.

[RNF002] Invalidade das opções.

O sistema não deve aceitar opções que não estejam disponíveis para reserva.

PRIORIDADE: (X) Essencial; ( ) Importante; ( ) Desejável.

3. Cenário de uso:

Usuário: Santos Dumont, administrador de um sistema de reserva de voos, deseja utilizar o sistema a fins de trabalho.

[RF001] Reservar um voo.

Dumont abre o sistema e seleciona a opção de reservar voos e em seguida é listado os voos disponíveis, depois disso selecione o número do voo que deseja ir, que neste caso foi a opção 1- Nova York, e informou o nome do tripulante Anakin Skywalker.

[RF002] Cancelar reserva.

Em seguida após cadastrar o voo Dumont recebe uma ligação do senhor Anakin Skywalker a quem ele tinha acabado de reservar um voo pedindo para cancelar esse voo, ciente disso Dumont seleciona a opção de cancelamento informando número do voo e o nome do tripulante que deseja cancelar, o que no sistema irá a ser novamente incrementado o valor do sistema.

[RF003] Adicionar voo.

A fim de cadastrar mais voos que foram repassados para o seu trabalho, Dumont cadastrar um voo para Nice na França onde após selecionar essa opção ele informa o destino, o número do voo e a quantidade de assentos disponíveis.

[RF004] Visualizar detalhes de voo.

Com o intuito de verificar se seu voo foi realmente cadastrado, Dumont seleciona a opção de visualizar detalhes de voo, e visualiza se todos os detalhes em que ele cadastrou estão corretos com o que precisa ser.

[RF005] Visualizar lista de passageiros de um voo.

Para realmente ter certeza se foi removido ou neste caso cancelado a reserva de voo do senhor Skywalker, Dumont utiliza de outra funcionalidade que tem disponível a ele onde ele verifica a lista de passageiros para Nova York, onde ela está vazia.

[RF006] Sair.

Após finalizar seu trabalho Dumont resolve sair do sistema, seleciona a opção de sair, onde o sistema encerra instantaneamente.