

MCVC – Sistema de Companhia Aérea

Projeto – MC322

Gabriela M. Jacob – RA: 186087

Igor E. Batista – RA: 260511

Letícia Lopes M. da Silva – RA: 184423

Renan Matheus S. Florencio – RA: 244882

1. INTRODUÇÃO

O principal objetivo do nosso projeto foi criar um sistema que pode ser usado por uma companhia aérea, de forma a concentrar as principais informações necessárias para realizar o gerenciamento, como por exemplo: voos, aeroportos, passagens, passageiros e clientes, entre outros.

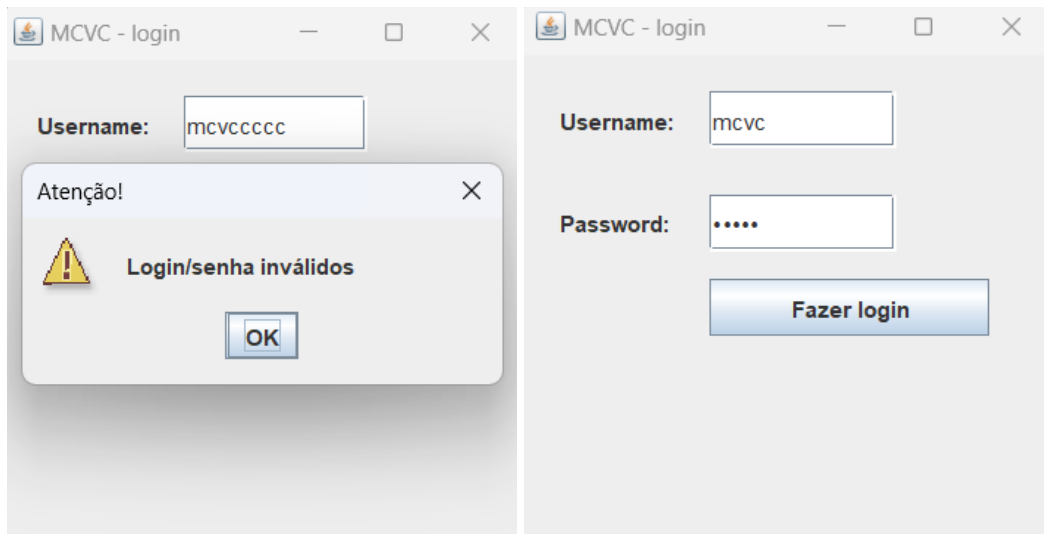
2. CLASSES

As classes utilizadas para implementar o sistema foram as seguintes:

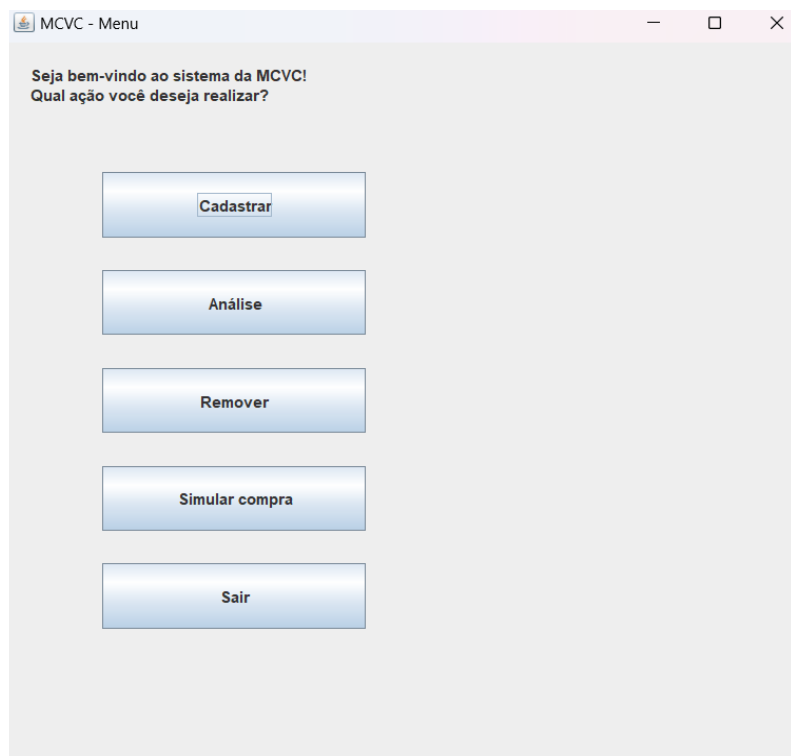
- **Aeroporto:** Aqui são armazenadas informações como a largura da pista e a lista de voos que o aeroporto em questão pode fazer. Cada aeroporto possui coordenadas únicas que usamos para obter as distâncias entre eles. A classe Trajeto, explicada mais à frente, é responsável por armazenar essas rotas.
- **Avião:** Classe que possibilita a existência de objetos do tipo Aviao, os quais possuem um número de série (que é único, e possibilita a identificação dos aviões), além de todos os métodos e atributos de acordo com a UML. A classe Aviao é abstrata, e é classe mãe de: AviaoCargueiro e AviaoComercial.
- **Avião Cargueiro:** Herdeira de Aviao, a classe AviaoCargueiro possibilita a existência de aviões que levam apenas carga, e cujas passagens são vendidas para ClientePJ.
- **Avião Comercial:** Já a classe AviaoComercial cria Aviao que pode transportar pessoas, vendendo Passagem para ClientePF.
- **Carga:** A classe Carga cria uma carga, a identifica e caracteriza, atrelando-a a uma Passagem.
- **Cliente:** A classe Cliente também é abstrata, tendo como herdeiras: ClientePF e ClientePJ.
- **ClientePJ:** No caso da classe ClientePJ, ela herda da classe Cliente, de forma que o que se adiciona é o CNPJ, único e identificador de cada objeto que seja iteração da classe.
- **ClientePF:** Já a classe ClientePF possui CPF (identificador único) e data de nascimento, tratado com o Calendar.

- **Companhia:** Classe principal do sistema, concentra (de forma direta ou indireta) todas as informações relevantes ao bom funcionamento do sistema. Diretamente, armazena a lista de Aeroportos atendidos, de Aviões adquiridos, bem como das conexões existentes entre os Aeroportos (Voos) e o histórico de Clientes. Essa classe possui o método `verificaTodosOsCaminhos`, que usamos para percorrer todas as conexões de aeroportos por meio de voos para encontrar todos os caminhos possíveis dentro de um limite de escalas.
- **Coordenada:** Classe que reúne, de forma estruturada as informações e métodos relacionados à localização dos aeroportos, ou seja, as informações de distância entre eles.
- **Arquivos:** Utilizamos uma estrutura de arquivo txt e criamos um leitor para ele para que seja possível criar diversos aeroportos, clientes e conexões ao rodar o código pela primeira vez. Também utilizamos o tipo serializable para armazenar a companhia quando o programa é fechado.
- **Passageiro:** Classe criada para identificar cada um dos passageiros no momento. Essa classe se difere da classe cliente no sentido de que um mesmo cliente pode comprar diversas passagens para diferentes passageiros. Cada passageiro possui sua passagem.
- **Passagem:** Uma passagem com informações de compra e trajeto é gerada sempre que um cliente (PF ou PJ) escolhe um trajeto para viajar. É dentro dessa classe que calculamos o valor de cada viagem.
- **Trajeto:** Classe responsável por armazenar uma lista de aeroportos que a compõem. A classe é usada sempre que o método de obter rotas para a companhia de um aeroporto a outro é chamado. A partir dela, também é possível calcular a distância total da rota e, dessa forma, calcular o preço da passagem para determinada viagem.
- **Validação:** A classe `Validacao` é a responsável por todos os exceptions dos inputs do sistema. Nela, foram implementados os tratamentos de erros de nome, CPF, CNPJ e data.
- **Voo:** A classe voo é a conexão do grafo que liga cada um dos aeroportos. Cada aeroporto possui uma lista de voos que será verificada quando for necessário calcular uma rota de um aeroporto a outro. Como cada um deles possui sua própria lista, é possível buscar recursivamente até que o aeroporto de destino seja encontrado.
- **Swing:** Essa classe foi utilizada para implementar a interface gráfica do sistema. A biblioteca Swing foi escolhida para realizar tal implementação, e foram feitas funções para cada uma das telas (JFrames) exibidas na interface. Em todas as telas, exceto a tela de login e a tela inicial (que possui o botão sair), foi implementado um botão “Voltar”, o qual fecha a janela atual para que o usuário possa voltar para a tela anterior.

- **Tela de login:** A primeira tela do sistema é uma tela de login, a qual por padrão possui usuário “mcvc” e senha “senha”. Também foi implementado um warning para o caso em que o login ou a senha são digitados incorretamente.

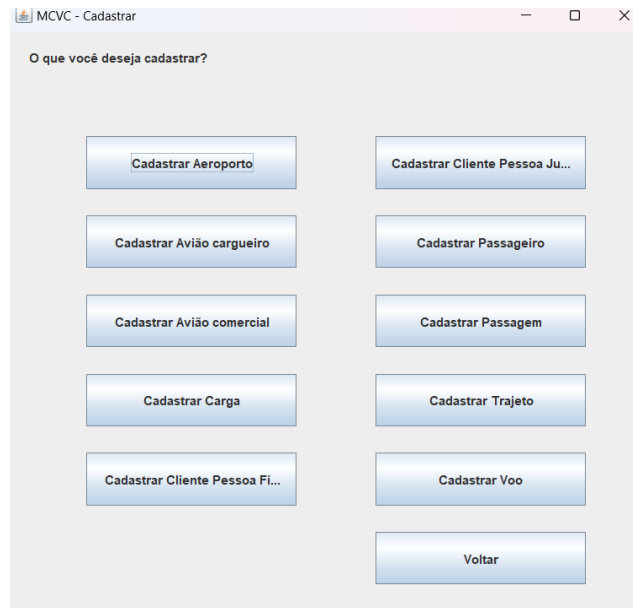


- **Tela inicial:** Já a tela inicial possui todas as opções possíveis de ação do sistema, sendo elas: cadastrar, remover, análise, simular compra e sair. Para as opções de cadastro e remoção, foram implementadas todas as funções no código, enquanto na interface gráfica foram implementados apenas cadastro e remoção de aeroporto, como um exemplo de funcionamento do sistema.



- **Tela de cadastro:** A tela de cadastro é aberta ao selecionar o JButton “Cadastrar”, e nela há a possibilidade de selecionar qual objeto deseja-se cadastrar no sistema.

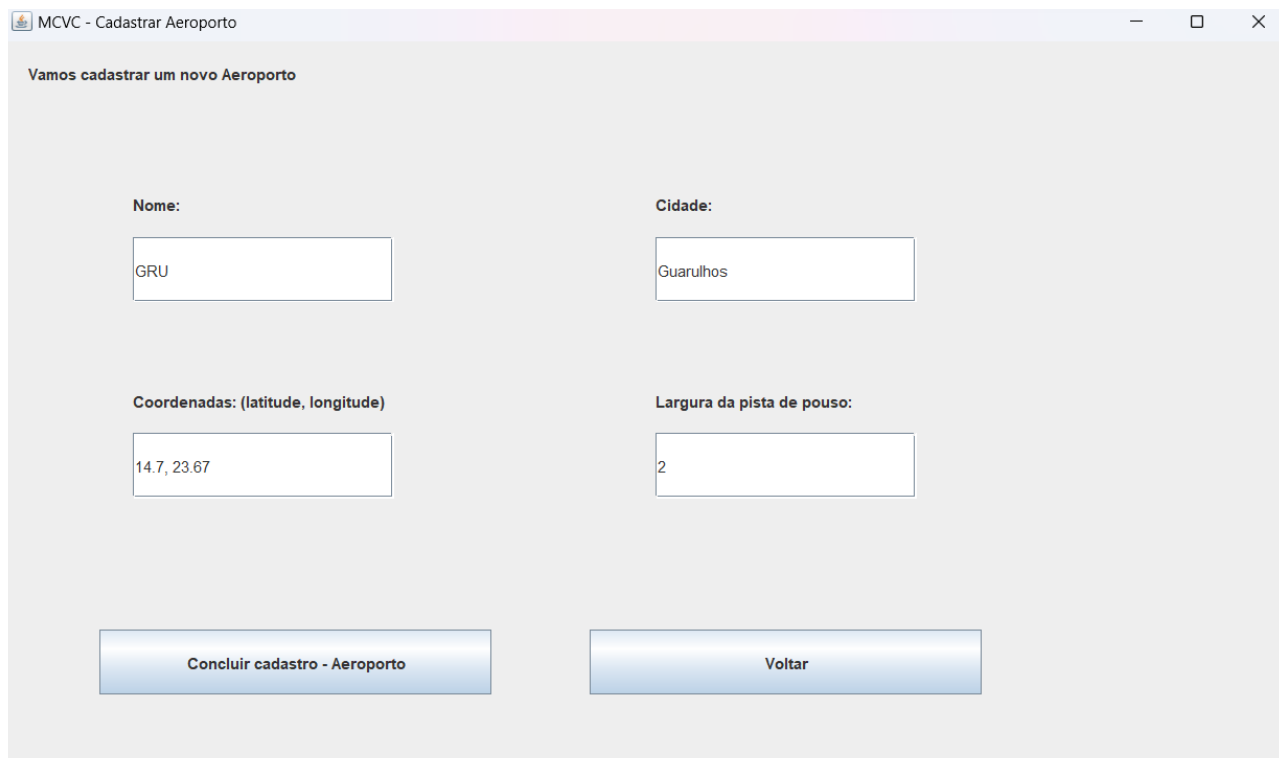
Selecionando “Aeroporto”, uma tela de cadastro é aberta, na qual é possível escrever os dados de um novo aeroporto e, ao selecionar o botão “Concluir cadastro”, todas as informações digitadas pelo usuário nas caixas de texto são salvas e analisadas pela classe Validação. Caso todos os dados estejam de acordo, o aeroporto é cadastrado no sistema, salvo na lista de aeroportos e aparece um popup informando que o aeroporto foi cadastrado com sucesso. Caso contrário, a classe validação retorna um “false”, o que aciona um popup de warning pedindo que o usuário preencha os campos com informações válidas. Além disso, a tela inicial possui um botão “Sair”, o qual, quando clicado, fecha as telas abertas e salva os dados inseridos naquela iteração.



MCVC - Cadastrar

O que você deseja cadastrar?

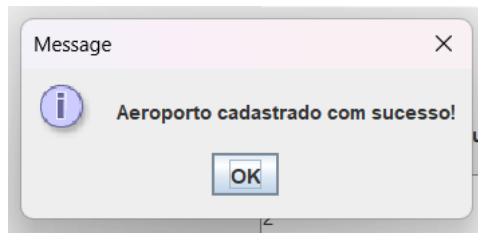
Cadastrar Aeroporto	Cadastrar Cliente Pessoa Ju...
Cadastrar Avião cargueiro	Cadastrar Passageiro
Cadastrar Avião comercial	Cadastrar Passagem
Cadastrar Carga	Cadastrar Trajeto
Cadastrar Cliente Pessoa Fi...	Cadastrar Voo
Voltar	



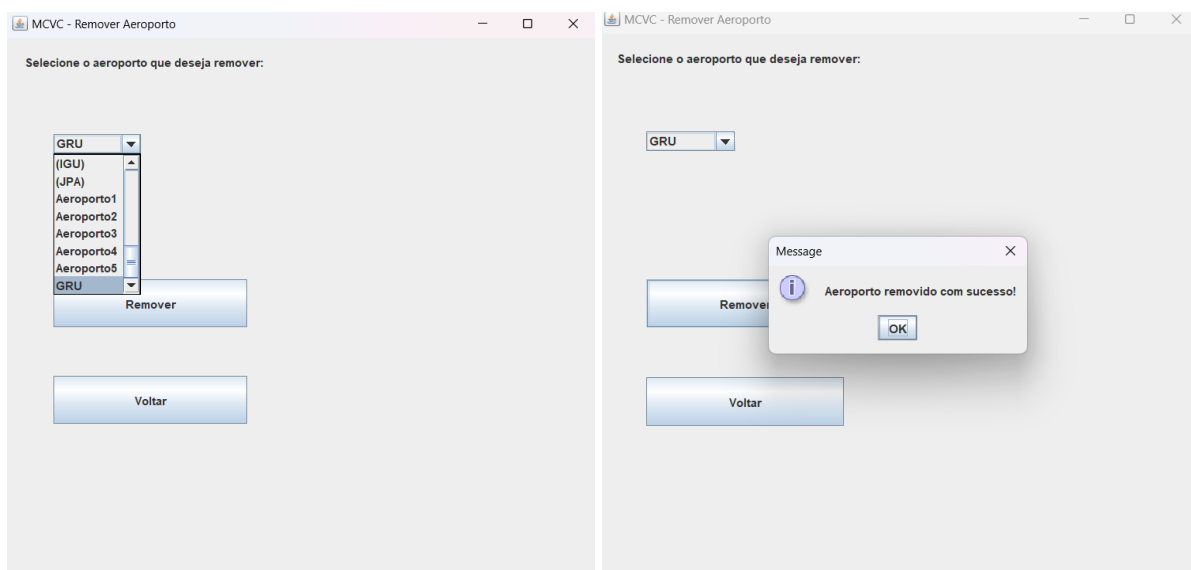
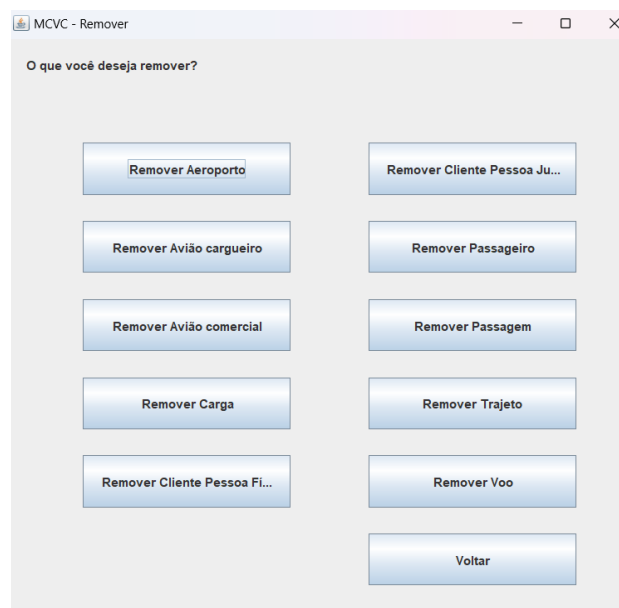
MCVC - Cadastrar Aeroporto

Vamos cadastrar um novo Aeroporto

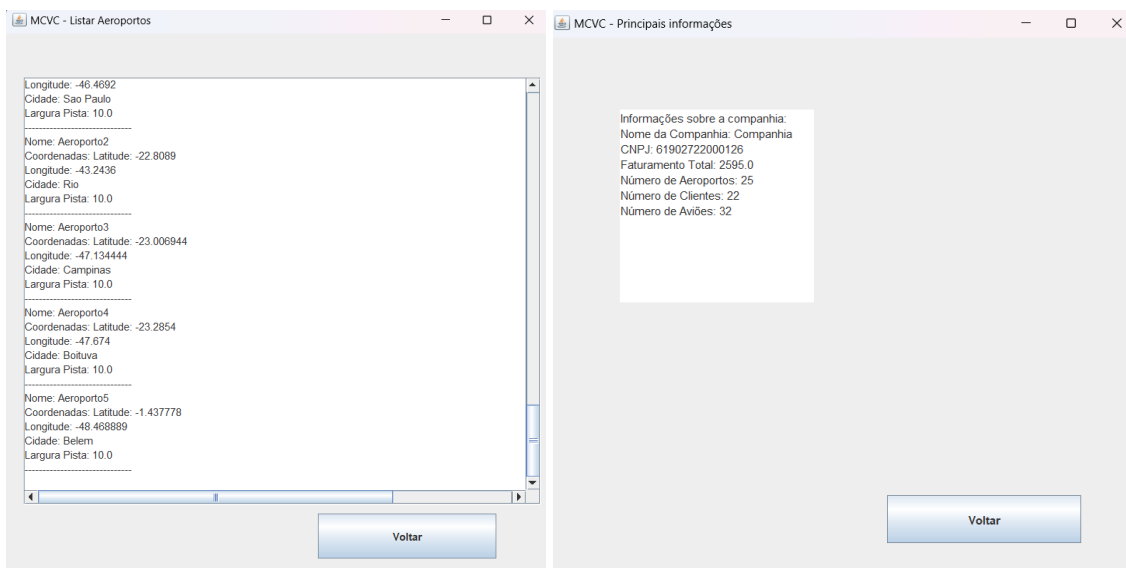
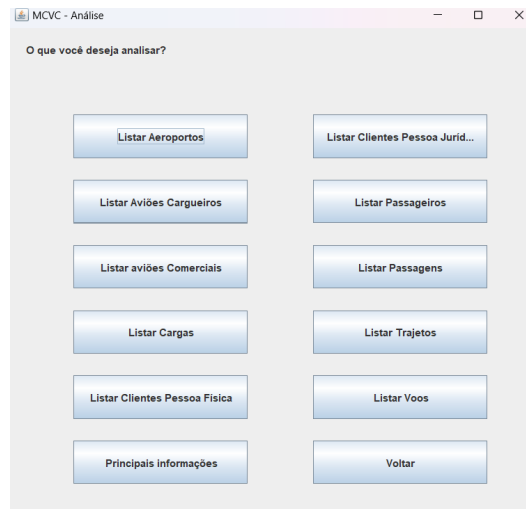
Nome: GRU	Cidade: Guarulhos
Coordenadas: (latitude, longitude) 14.7, 23.67	Largura da pista de pouso: 2
Concluir cadastro - Aeroporto	Voltar



- **Tela de remoção:** A tela de remoção também possui todos os botões de objetos que se pode remover. Uma vez selecionado o botão “Aeroporto”, uma outra tela abre, na qual é possível selecionar o aeroporto que se quer remover da lista de aeroportos já cadastrados, por meio de um JComboBox. Quando o botão de concluir remoção é clicado, o aeroporto é removido do sistema.



- **Tela de análise:** A tela de Análise consiste em gerar listas dos objetos cadastrados no sistema, além da opção de gerar as principais informações sobre a companhia.



- **Tela de simulação de compra:** Por fim, o botão “Simular Compra” da tela inicial leva o usuário para uma tela com duas JComboBox, sendo a primeira o aeroporto de origem e a segunda, o de destino. Assim, o usuário pode selecionar suas preferências e, ao clicar no botão “Calcular Trajetos”, o sistema gera em um JTextArea a lista de possíveis trajetos, o que inclui: as escalas, a distância total a ser percorrida, e o preço da passagem, o qual é calculado de acordo com uma função da classe Companhia.

MCVC - Simular compra

Vamos simular uma compra de passagem.

Origem:

(GRU)

Destino:

(SSA)

Calcular trajetos

Voltar

Guarulhos (GRU) -> Salvador (SSA)
Distancia: 1454 km
Preço: 654.0
Guarulhos (GRU) -> Campinas (VCP)
Distancia: 1543 km
Preço: 521.0
Guarulhos (GRU) -> Recife (REC) ->
Distancia: 2755 km
Preço: 930.0
Guarulhos (GRU) -> Recife (REC) ->
Distancia: 3751 km
Preço: 1013.0
Guarulhos (GRU) -> Recife (REC) ->
Distancia: 6674 km
Preço: 1802.0