



Parte 02

Professor: Edwar Saliba Júnior

Valor: 15 pontos

Contextualização:

A empresa aérea Tabajara Airlines (TA), já está utilizando a primeira versão do sistema que sua empresa disponibilizou para ela. Em princípio, o sistema até estava atendendo as necessidades da empresa, porém, com o passar do tempo e com o aumento da versatilidade no atendimento aos clientes, a empresa viu-se numa posição favorável para ampliar a gama de serviços disponibilizados aos seus clientes e resolveu ampliar também as funcionalidades do sistema.

Paralelamente às necessidades de inovação da empresa, a ANAC (Agência Nacional de Aviação Civil) solicitou que todos os aeroportos do país estejam preparados para receber qualquer tipo de aeronave, inclusive as mais modernas. Para tanto, solicitou que os registros destas aeronaves sejam feitos de forma que o cadastro esteja bem detalhado, pois, haverá futuramente, a necessidade de modernização dos aeroportos e, para isto, serão feitas estatísticas com os tipos de aeronaves que mais utilizaram os aeroportos. Assim sendo, e para atender as necessidades atuais, a ANAC juntamente com a TA sugeriram a seguinte estrutura de software:

1) Estrutura de classes¹ (Figura 1):

- Cliente,
- Piloto,
- Cidade,
- Aeroporto,
- Voo,
- Aeronave (Carro, Avião e Helicóptero) e
- Passagem.

2) Atributos das Classes:

- Cliente
 1. identificação
 2. nome
 3. logradouro
 4. número
 5. bairro
 6. município
 7. estado
 8. cep
 9. telefone
 10. cpf
- Aeronave
 1. identificação
 2. modelo
 3. capacidade de passageiros
 4. capacidade de carga
- Carro
 1. Piloto
 2. Cidade de origem
 3. Cidade de destino
 4. autonomia de voo (em horas)
- Helicóptero

¹ Neste diagrama não estão representadas as classes de gerenciamento.

- Avião
 1. quantidade de hélices
 1. quantidade de turbinas
 2. capacidade do tanque de cada turbina
- Piloto
 1. identidade
 2. cpf
 3. número do brevê
 4. logradouro
 5. número
 6. bairro
 7. Cidade
 8. telefone
- Cidade
 1. país
 2. estado
- Aeroporto
 1. identificação
 2. nome
 3. Cidade
- Voo
 1. Avião
 2. Aeroporto de partida
 3. data de partida
 4. horário de partida
 5. Aeroporto de chegada
 6. data de chegada
 7. horário de chegada
 8. lotação (quantos passageiros compraram passagem para este voo)
 9. peso da carga embarcada
 10. valor da viagem
- Passagem
 1. Voo
 2. Cliente
 3. data da venda
 4. hora da venda
 5. preço final da viagem

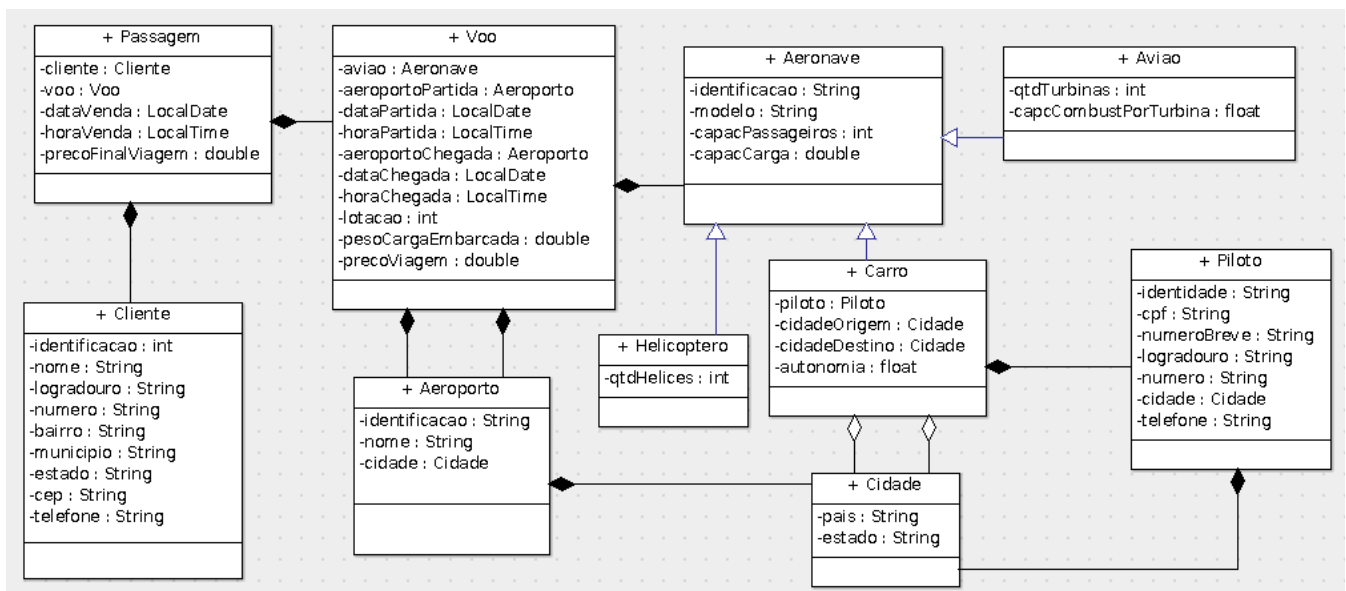


Figura 1: Diagrama de Classes do Sistema

- 3) Caso seja necessário, os grupos poderão criar mais atributos nas classes. Os métodos deverão ser criados de acordo com a necessidade de cada *software*, por este motivo não foram especificados no diagrama UML (Figura 1);
- 4) Sua empresa deverá seguir fielmente os nomes das classes, os nomes dos atributos e tipos que foram propostos pela TA e pela ANAC na Figura 1. Caso isto não aconteça, haverá uma grande penalização por parte dos avaliadores.
- 5) Para esta nova versão, a TA e a ANAC solicitaram que todos os erros encontrados na primeira versão e todas as propostas de melhoria sejam implementadas, pois, caso contrário, haverá penalizações dobradas para cada erro persistido ou melhoria não efetuada.
- 6) As regras de desconto, *overbooking*² e todas as demais regras que foram impostas na versão anterior, não deverão ser alteradas e devem estar funcionando perfeitamente para esta nova versão.
- 7) O *software* deverá fazer uso de um SGBD (Sistema Gerenciador de Banco de Dados) para armazenamento e recuperação dos dados cadastrados. Após uma extensa análise, chegou-se a conclusão de que o SGBD mais adequado em termos de segurança, desempenho e confiabilidade é o PostgreSQL. Assim sendo, você deverá fazer uso do mesmo para implementação do software.
- 8) Para facilitar a manipulação do banco de dados, sugere-se que se criem classes DAO (*Data Access Object*). E que a estrutura do *software* passe a contemplar a arquitetura MVC (*Model, View, Controller*). É tarefa das equipes de desenvolvimento, estudarem e implementarem o padrão DAO e o padrão MVC.
- 9) Para manipulação de objetos em memória principal, **se for o caso**, exige-se que seja utilizada uma única estrutura de ArrayList para armazenar Carro, Avião e Helicóptero. Para tanto, faça uso do recurso conhecido como Polimorfismo.
- 10) A estrutura de *menu* deverá continuar semelhante a da primeira versão do *software*.

Como o seu *software* deverá funcionar

- A estrutura de *menus*, deverá proporcionar ao usuário:
 - a possibilidade de navegar entre o *menu* principal e seus *submenus* sem efetuar qualquer operação;
 - o usuário só poderá sair do programa através do *menu* principal, ou seja, acessando a opção “Sair”;
- Um *software* deve ter uma boa aparência e ser de fácil utilização, para agradar e facilitar a vida de quem o utilizará;
- O *software* que será entregue não deverá fazer uso de vetores e tampouco matrizes. No lugar destes use qualquer objeto das *Collections* de Java.

Regras para a entrega do trabalho

- Deverá ser apresentado e entregue, o projeto (compactado) do código-fonte.
- **O código-fonte que será entregue e apresentado não deverá possuir nenhum tipo de comentário.**
- Deverá ser enviado para o e-mail: eddiesaliba2@yahoo.com (de acordo com as regras a seguir).

2 Excesso de passageiros, ou seja, venda de passagens acima da capacidade que o avião suporta.

- **Não serão recebidos trabalhos após a data marcada para entrega.**
- Para a apresentação no laboratório deverá ser levado pelo grupo, em *pendrive*, uma cópia do arquivo que foi enviado por *e-mail*. Caso o grupo possua alguma restrição ou dificuldade no cumprimento desta regra, então, deverá avisar ao professor com antecedência mínima de 24 horas da data de apresentação.

Regras para envio do *e-mail* com o trabalho

- No assunto do *e-mail* deve constar apenas o título: **IFTM - POO - Paracatu - ADS3PA**
- No corpo do *e-mail* deverá conter, **única e exclusivamente**, o nome completo de todos os integrantes do grupo (**um em cada linha**).
- Só será aceito **UM** *e-mail* por grupo. Portanto, verifique se está tudo certo com seu *e-mail* e trabalho antes de enviá-lo.
- **O *e-mail* deverá ser enviado, no máximo, até UM dia antes da data marcada para apresentação.**

Obs.: O desrespeito a qualquer das regras acima implicará na perda de créditos para o grupo.

Critérios de Avaliação no Laboratório:

- Conformidade do *software* em relação ao solicitado;
- Legibilidade do código (**organização, endentação e etc.**);
- Usabilidade das interfaces de interação com o usuário;
- **Entendimento individual a respeito do código-fonte apresentado.**

BÔNUS: o grupo que conseguir implementar um relógio que funcione, ou seja, não estático e que mostre HH:MM:SS no *menu* principal do *software*, ganhará 1 ponto extra.

BÔNUS 2: o grupo que conseguir implementar um relógio que funcione, ou seja, não estático e que mostre HH:MM:SS em todas as telas do *software*, ganhará 3 pontos extra.

Obs.: os bônus não são cumulativos.