Pontifícia Universidade Católica de Minas Gerais
Instituto de Ciências Exatas e Informática

Nome: Gabriel Henrique Vieira de Oliveira, Matheus Filipe Rocha Viana, Vinicius Cezar Pereira Menezes
Data: 11/09/2025
Matéria: Algorítmos e Estruturas de Dados 3

---

1) Descrição do problema:

O **Sistema de Planejamento de Viagens** tem como objetivo auxiliar usuários na organização de roteiros completos. Cada usuário pode cadastrar suas viagens, informando dados como destino, orçamento e período (data de início e fim). Dentro de cada viagem, é possível planejar atividades (como passeios, eventos ou compromissos), que possuem atributos próprios, como título, descrição, data, custo, status e prioridade.

As atividades são classificadas em categorias, permitindo ao usuário organizar melhor seu planejamento. Dessa forma, o sistema garante uma visão estruturada das viagens, associando atividades a cada roteiro.

O sistema contempla dois perfis principais: **Usuário**, que gerencia suas próprias viagens e atividades, e **Administrador**, responsável pela manutenção do sistema e integridade dos dados.

A arquitetura segue o padrão **MVC + DAO**, assegurando organização do código, persistência em arquivos binários e facilidade de manutenção. Assim, a aplicação oferece uma solução prática e confiável para centralizar e planejar viagens de forma eficiente.

2) Objetivo do trabalho:

• Desenvolver um sistema de planejamento de viagens que permita o CRUD de viagens, usuários e atividades.

• Garantir persistência em arquivos binários com controle de exclusão lógica.

• Fornecer documentação contendo Diagrama de Caso de Uso (DCU), Diagrama Entidade-Relacionamento (DER) e Arquitetura Proposta (MVC).

3) Requisitos Funcionais:

• **RF01:** O sistema deve poder cadastrar viajante (usuário).

• **RF02:** O sistema deve poder visualizar dados de um viajante (usuário).

• **RF03:** O sistema deve poder excluir viajante (usuário).

• **RF04:** O sistema deve poder alterar dados de um viajante (usuário).

• **RF05:** O sistema deve poder criar uma viagem (informando destino, valor, data início e data final).

• **RF06:** O sistema deve poder alterar uma viagem já criada através do id.

• **RF07:** O sistema deve poder excluir uma viagem.

• **RF08:** O sistema deve poder visualizar uma viagem.

• **RF09:** O sistema deve permitir cadastrar uma atividade para uma viagem (informando título, descrição, data, custo, status e prioridade).

• **RF10:** O sistema deve permitir visualizar as atividades associadas a uma viagem.

• **RF11:** O sistema deve permitir alterar uma atividade já cadastrada através do ID.

• **RF12:** O sistema deve permitir excluir uma atividade de uma viagem.

• **RF13:** O sistema deve permitir cadastrar categorias de atividades (informando nome).

• **RF14:** O sistema deve permitir associar uma atividade a uma categoria.

• **RF15:** O sistema deve permitir visualizar categorias cadastradas.

4) Requisitos Não Funcionais:

• **RNF01:** Interface mínima em HTML/CSS (pode ser estática e não obrigatória).

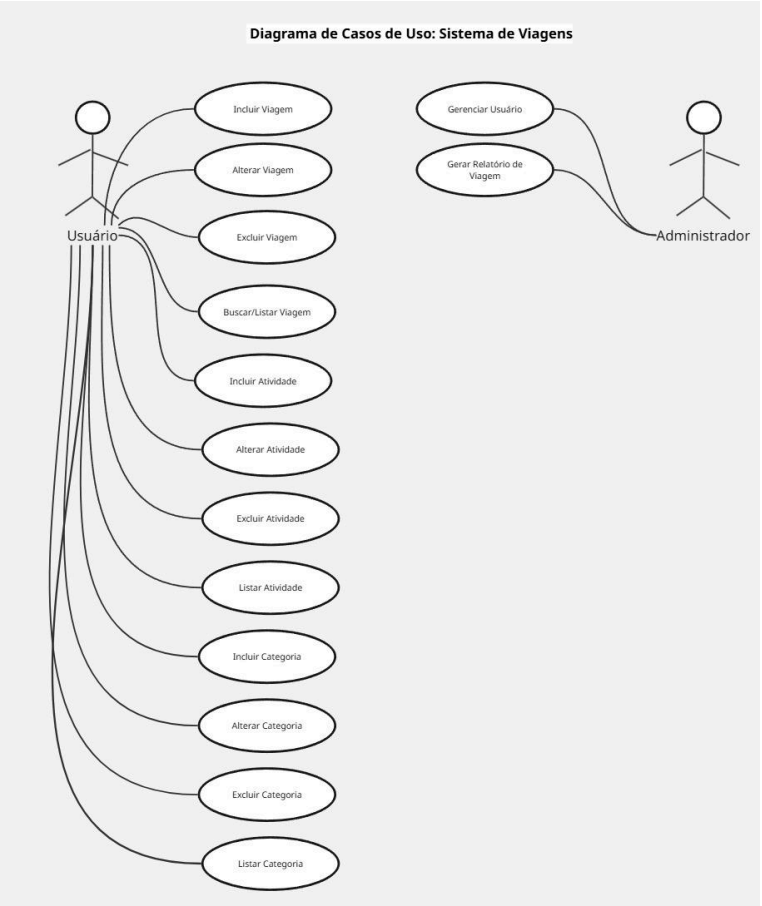
• **RNF02:** Persistência obrigatória em arquivos binários com cabeçalho.

• **RNF03:** Documentação obrigatória (DCU + DER + Arquitetura).

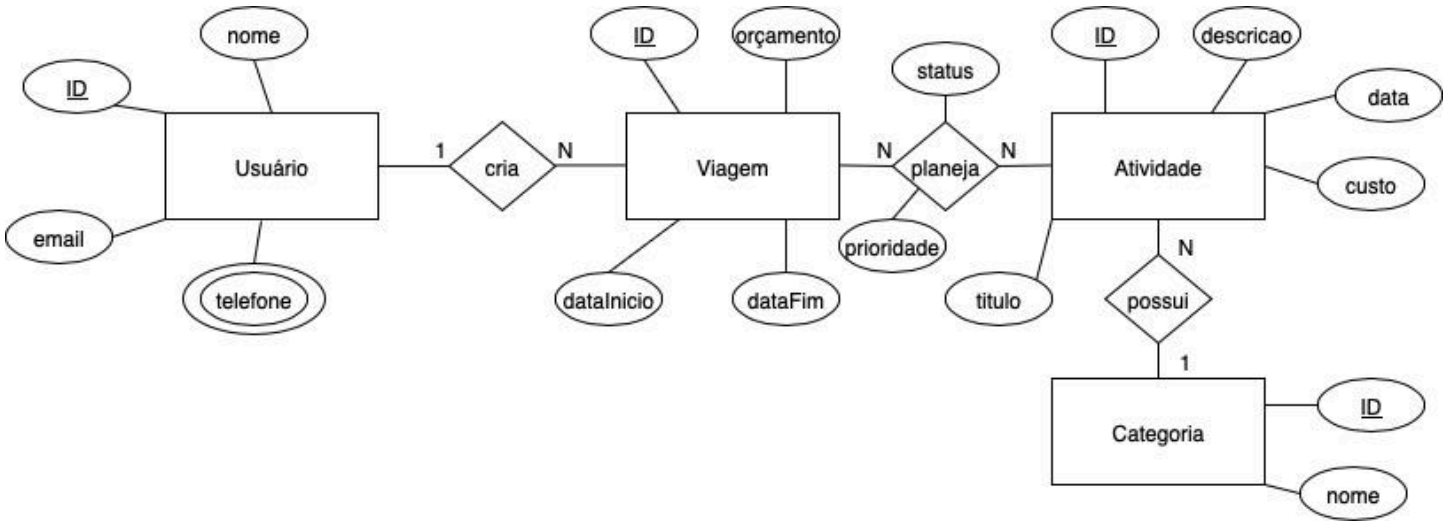
5) Atores:

- **Usuário:** cadastra e gerência suas próprias viagens (incluir, excluir, alterar e buscar);
- **Administrador:** gerência o sistema e os usuários, tendo como objetivo manter a integridade do sistema e realizar manutenções regulares (backup, por exemplo);
- **Atividade:** Entidade associada à viagem, representando compromissos, passeios e eventos programados;
- **Categoria:** classifica as atividades por tipo (ex.: lazer, trabalho, cultura), permitindo melhor organização do planejamento.

6) Diagramas de Caso de Uso:



7) Diagrama Entidade-Relacionamento:



8) Arquitetura proposta:

O sistema seguirá o padrão **MVC + DAO**, distribuído nas seguintes classes:

- **MODEL:**
  Viagem.java - representa a entidade Viagem dos diagramas e seus respectivos atributos;
  Registro.java - encapsula dados auxiliares para manipulação de registros;

- **DAO (Data Access Object):**
  ViagemDAO.java - responsável pelo CRUD de viagens (inserir, buscar, alterar, excluir), operando sobre os arquivos binários;
  Arquivo.java - manipulação dos arquivos;

- **CONTROLLER:**
  BuscarViagem.java - implementa a lógica de busca de viagens conforme critérios do usuário;
  MenuViagem.java - controla o fluxo do CRUD, chamando métodos da DAO e exibindo opções específicas ao usuário;

- **VIEW:**
  Principal.java - ponto de entrada do sistema, exibindo menus iniciais e direcionando para a interação em console;
  MenuViagem.java - atua como meio de vizualização, pois também recebe informações do usuário;

Diagrama de Arquitetura em Camadas: