

Tema: Sistema de Gestão de Estudantes

Este trabalho prático servirá como exame e tem como finalidade consolidar os conhecimentos de Programação Orientada a Objetos em Java, modelagem UML e persistência de dados utilizando ORM (Object–Relational Mapping) com JPA/Hibernate.

O projeto será desenvolvido de forma individual e estará sujeito a defesa oral obrigatória, na qual o estudante deverá demonstrar domínio técnico e conceptual sobre a solução apresentada.

1. Objetivo Geral

Desenvolver uma aplicação em Java que implemente um Sistema Simples de Gestão de Estudantes, aplicando corretamente os princípios de Programação Orientada a Objetos, utilizando ORM (JPA com Hibernate) para realizar operações de CRUD numa base de dados relacional.

2. Objetivos Específicos

- Aplicar os princípios da Programação Orientada a Objetos (encapsulamento, herança, polimorfismo e abstração).
- Modelar o sistema utilizando diagramas UML.
- Configurar e utilizar corretamente um ORM (JPA/Hibernate).
- Implementar operações completas de CRUD.
- Separar responsabilidades por camadas (modelo, persistência, aplicação).
- Defender tecnicamente a solução desenvolvida.

3. Descrição Geral do Sistema

A aplicação deverá permitir o cadastro, consulta, atualização e remoção de estudantes numa base de dados relacional.

A aplicação pode ser:

- Java SE em modo console, ou
- Interface gráfica simples (Swing ou JavaFX).

O foco principal é a arquitetura do sistema, o uso correto de JPA e a aplicação de conceitos de POO.

Bonzo F. Calisto

4. Requisitos Funcionais

O sistema deverá permitir:

1. Cadastrar estudantes com os seguintes dados:
 - Nome
 - Idade
 - Curso
 - E outros
2. Listar todos os estudantes cadastrados.
3. Atualizar os dados de um estudante existente.
4. Remover um estudante pelo identificador.

5. Requisitos Técnicos Obrigatórios

5.1 Linguagem e Tecnologias

- **Linguagem:** Java
- **Paradigma:** Programação Orientada a Objetos (obrigatório)
- **ORM:** JPA com Hibernate
- **Banco de Dados:** MySQL ou H2
- **Configuração ORM:** `persistence.xml`
- **Gestão do Projeto:** Maven (opcional, mas recomendado)

5.2 Programação Orientada a Objetos (Obrigatório)

O projeto deverá obrigatoriamente demonstrar:

- Encapsulamento (atributos privados e métodos de acesso).
- Herança (ex.: classe base `Pessoa`).
- Polimorfismo (uso de métodos sobrescritos e/ou interfaces).
- Abstração (classes abstratas ou interfaces).

Exemplo mínimo exigido:

- Classe abstrata `Pessoa`
- Classe `Estudante` que herda de `Pessoa`
- Interface `Repositorio<T>` com métodos genéricos de CRUD
- Implementação concreta `EstudanteDAO` utilizando JPA

Bonzo F. Calisto

6. Modelagem UML (Fase Obrigatória)

O estudante deverá entregar e defender pelo menos 5 diagramas UML, devidamente identificados:

1. Diagrama de Casos de Uso
2. Diagrama de Classes
3. Diagrama de Sequência (para uma operação CRUD)
4. Diagrama de Atividades
5. Diagrama de Pacotes

Os diagramas devem refletir fielmente a implementação final.

7. Fases do Projeto

Fase 1 – Análise e Modelagem (UML)

- Identificação dos requisitos
- Elaboração dos diagramas UML (mínimo 5)

Fase 2 – Implementação

- Implementação das classes do domínio
- Aplicação de herança e polimorfismo
- Implementação do CRUD com JPA

Fase 3 – Persistência de Dados

- Configuração correta do `persistence.xml`
- Mapeamento da entidade `Estudante`
- Utilização do `EntityManager`

Fase 4 – Testes e Validação

- Testes das operações CRUD
- Demonstração do funcionamento da aplicação

Fase 5 – Documentação e Defesa

- Elaboração do relatório
- Defesa oral individual

Bonzo F. Calisto

8. Entidade Principal

Estudante

Atributo Tipo

id	Long
nome	String
idade	Integer
curso	String

9. Exemplo de Configuração ORM (Obrigatório Anexar)

Exemplo de `persistence.xml`

```
<persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence"
    version="2.2">
  <persistence-unit name="gestaoEstudantesPU">
    <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>

    <class>model.Estudante</class>

    <properties>
      <property name="javax.persistence.jdbc.driver" value="org.h2.Driver"/>
      <property name="javax.persistence.jdbc.url" value="jdbc:h2:mem:estudantes"/>
      <property name="javax.persistence.jdbc.user" value="sa"/>
      <property name="javax.persistence.jdbc.password" value=""/>

      <property name="hibernate.dialect" value="org.hibernate.dialect.H2Dialect"/>
      <property name="hibernate.hbm2ddl.auto" value="update"/>
      <property name="hibernate.show_sql" value="true"/>
    </properties>
  </persistence-unit>
</persistence>
```

10. Critérios de Avaliação

Critério	Peso
Modelagem UML (5 diagramas)	10%
Uso correto de POO e Polimorfismo	30%
Configuração e uso do ORM (JPA/Hibernate)	15%
Funcionamento do CRUD	25%
Organização e qualidade do código	10%
Defesa individual	10%

Bonzo F. Calisto

11. Critérios de Defesa Individual (Obrigatório)

Na defesa, o estudante deverá ser capaz de:

- Explicar os diagramas UML apresentados.
- Justificar as decisões de design orientado a objetos.
- Demonstrar onde e como o polimorfismo é aplicado.
- Explicar o funcionamento do ORM e do EntityManager.
- Executar e explicar uma operação CRUD em tempo real.
- Responder a questões técnicas sobre o código desenvolvido.

A não comparência ou reprovação na defesa implica reprovação no trabalho.

12. Entrega

O estudante deverá submeter via **classroom** dentro do prazo:

1. Código-fonte completo do projeto
2. Relatório em PDF
3. Diagramas UML
4. Certificado de curso de JPA realizado

Bonzo F. Calisto