

# SISTEMA UNI4

Nelson Lima – Dez/2025

# OBJETIVOS

## **Sistema para gestão acadêmica**

- Cadastro de gestão de Alunos e Professores
- Acompanhamento das notas
- Acompanhamento dos semestres

# DECISÕES DE ARQUITETURA / TECNOLOGIA

## Banco de dados / Controle de acesso

Banco de dados relacional

PostgreSQL 16.7 – Gratuito e robusto – Armazenar dados

Keycloak 26.4.6 – Legado - Gerenciador de acesso

## Backend

Monolito modular

Orientado a camada – Isolamento do erro, uniformidade funcional

Quarkus 3.30.2 – Otimizado, Flexível

Panache ORM – Simples e funcional

Java 21 – Simples, estável, seguro

Gradle Builder – Integrado, simples, rápido

## Frontend

Página Única(SPA)

a domínio – Clareza, coesão, baixo acoplamento

Angular 20 – Robusto, escalável

PrimeNG – Completo, dinâmico

Template SakaiNG – Gratuito, flexível

## Testes

### Quarkus

- Unitário: JUnit5 e Mockito
- Integração: RestAssured

### Angular

- Unitário: Jest
- Funcional: Cypress

### Infra

- Docker / docker-compose
- GitHub

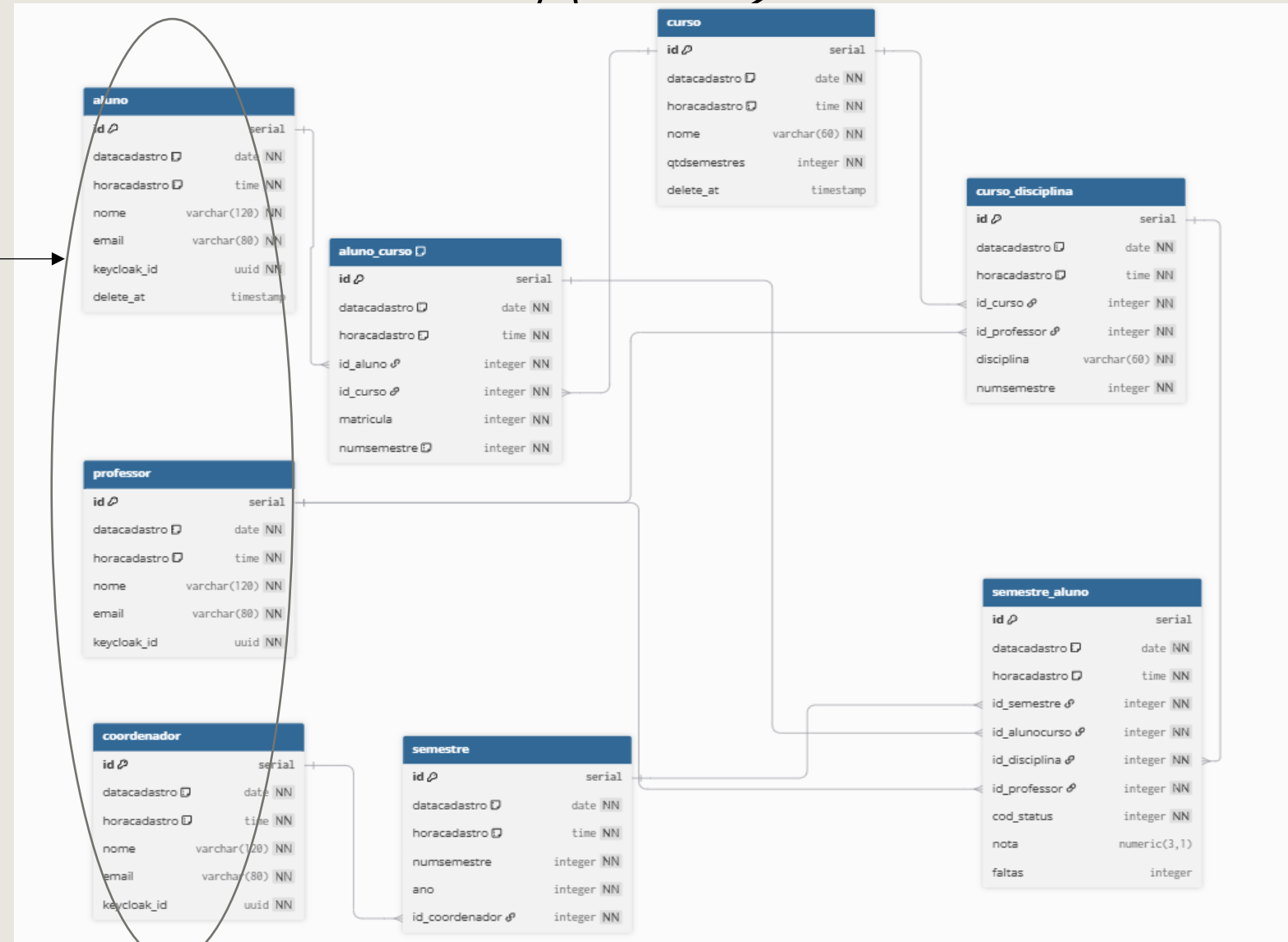


DESENVOLVIMENTO

# BANCO DE DADOS - DIAGRAMA

## Atores:

- Coordenador:
  - Crud(Curso,Professor, Aluno)
  - Gerencia semestre(Novo, Encerra)
- Professor:
  - Lança notas e faltas por semestre para cada disciplina e aluno
- Aluno:
  - Consome o conteúdo, acompanha as notas e falta por semestre por disciplina



# BACKEND

**cliente:** Clientes de outros serviços. Ex.:Keycloak

**logs:** Últimos erros gerados pelas exceptions

**dto:** Objetos de transferência

**entity:** Entidade reflexo da estrutura da tabela do banco para persistência

**mapper:** Conexão entre o DTO e a Entity. Faz a conversão

**repository:** Isola a lógica da persistência. Conexão entre o serviço e dados

**resource:** Recursos que a API vai disponibilizar. A classe que controla os endpoints

**service:** Camada lógica, orquestra as operações, garantir a que as regras sejam seguidas.

**test:** Implementa os testes e garante a qualidade

**exception:** Controle de erros. Temos 4 Exceções:

- **EntityNotFoundExceptionMapper:**  
*Erro na integridade dos dados e da lógica*
- **IllegalArgumentExceptionHandler:**  
*Erro na qualidade do dado(null, overflow, maior, menor, tipo compatível)*
- **ServiceExceptionHandler :**  
*Erro no consumo de serviço(API) externa(Ex. Keycloak)*
- **GlobalExceptionHandler :**  
*Erros globais, gerais não previstos por outras exceptions*

```
✓ QUARKUS-ACADEMICO
  > .gradle
  > bin
  > build
  > gradle
  > logs
  ✓ src
    ✓ main
      > docker
      ✓ java\com\uni4
        > client
        > dto
        > entity
        > exception
        > mapper
        > repository
        > resource
        > service
        J Application.java
        J GreetingResource.java
        J MyEntity.java
      > resources
      > native-test
      > test
```

# FRONTEND

**core:** Contém as subpastas que organizam a lógica da aplicação de serviços compartilhados; Dados gerais, cálculos, acessos e outros mais

**home:** Página de entrada do sistema, ponto comum para redirecionamento

**notfound:** Caso haja erro de rota, links não encontrados erro.

**assets:** Arquivos estáticos necessários pra aparência da aplicação

**environments:** Armazena configurações de acordo com ambiente, produção, teste, virtual ou outro

```
▼ UNI4_ACADEMICO-FRONT
  ▼ angular_uni4-academico
    > .angular
    > .vscode
    > node_modules
    > public
    ▼ src
      ▼ app
        ▼ components
        > core\services
        > layout
        ▼ pages
          > aluno
          > auth
          > coordenador
          > home
          > notfound
          > professor
          > utils
```

**components:** Armazena os elementos de UI da aplicação para reutilização

**layout:** Estrutura geral da aplicação SPA. Esqueleto principal com cores e estilos

**pages:** Contêm as páginas do sistema estrutura Horizontal, Atores(Aluno, Professor, Coordenador) e Gerais

**aluno:** Subdivisão da aplicação da perspectiva da interação com ator Aluno

**auth:** Responsável pelo Login e Guards

**coordenador:** Subdivisão da aplicação para as interações do Coordenador

**professor:** Subdivisão da aplicação para as interações do Professor

**utils:** Armazena componentes no menu Utils da aplicação



## CONSIDERAÇÕES FINAIS

Desafio proposto foi bastante interessante.

A combinação do Angular + PrimeNG com Quarkus + Panache + PostgreSQL são ferramentas bastante robustas.

Não foi levado em consideração a completude de todas as funcionalidades. Foi focado na estruturação e organização (**Scaffolding**) do projeto.

Todos os elementos foram estourados de forma bem organizada, entregues no ponto para sequência de um desenvolvimento profissional.

A integração com Keycloak foi um ponto desafiador, pois não tinha experiência.

Link do projeto:

<https://github.com/NelsonLima07/DesafioUnifor>





# OBRIGADO

Nelson Lima

(85)99615-7534

jnelsonlima3@gmail.com

<https://www.linkedin.com/in/nelson-lima-6250a320/>