

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ

CAMPUS MARACANAÚ

CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

**YAN PEDRO FAÇANHA BRASILEIRO
THALES LUCAS LIMA E GOMES
JOÃO VICTOR PEQUENO DAMASCENO
RICARDO MAGALHÃES PARENTE FROTA
DERICK BESSA DE ALMEIDA**

Relatório Técnico - Sistema SIGEJ

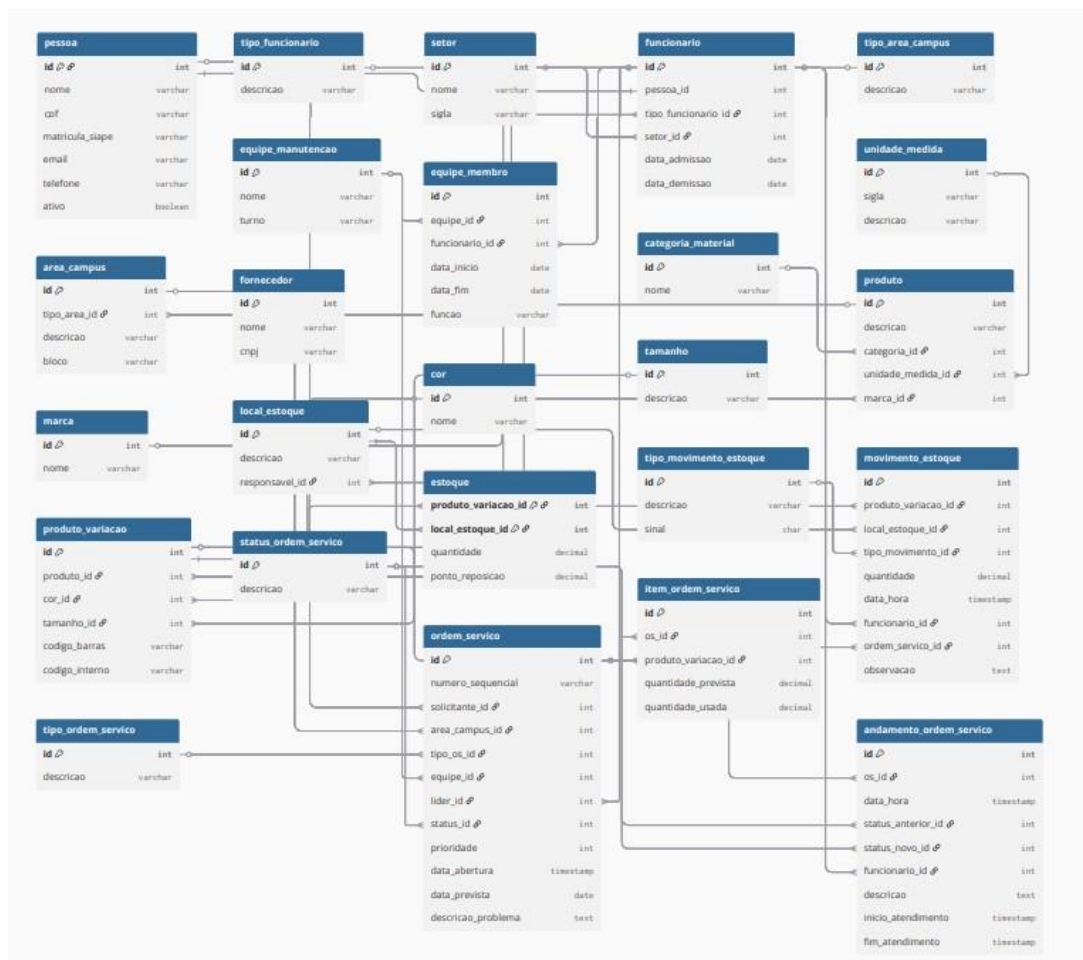
MARACANAÚ, 2025

1 - INTRODUÇÃO

Este projeto implementa o backend do sistema SIGEJ utilizando Java com Spring Boot. O banco de dados utilizado foi o PostgreSQL (via Docker), e a persistência foi feita com JDBC puro (sem ORM), conforme solicitado. O sistema atende aos 4 módulos principais: Gestão de RH (Equipes e Funcionários), Catálogo de Materiais, Controle de Estoque e Gestão de Ordens de Serviço (incluindo Infraestrutura do Campus).

2 - DIAGRAMA DER

A Figura abaixo apresenta o Diagrama Entidade-Relacionamento (DER) do sistema SIGEJ. O modelo é composto por 25 tabelas normalizadas, atendendo integralmente aos requisitos do projeto. As relações foram implementadas no banco de dados PostgreSQL através de chaves estrangeiras (Foreign Keys) e restrições de integridade (Constraints), garantindo a consistência dos dados sem o uso de ORM.



3 - RELATÓRIO DE EXECUÇÃO (testes no postman)

3.1. Módulo de Recursos Humanos (RH)

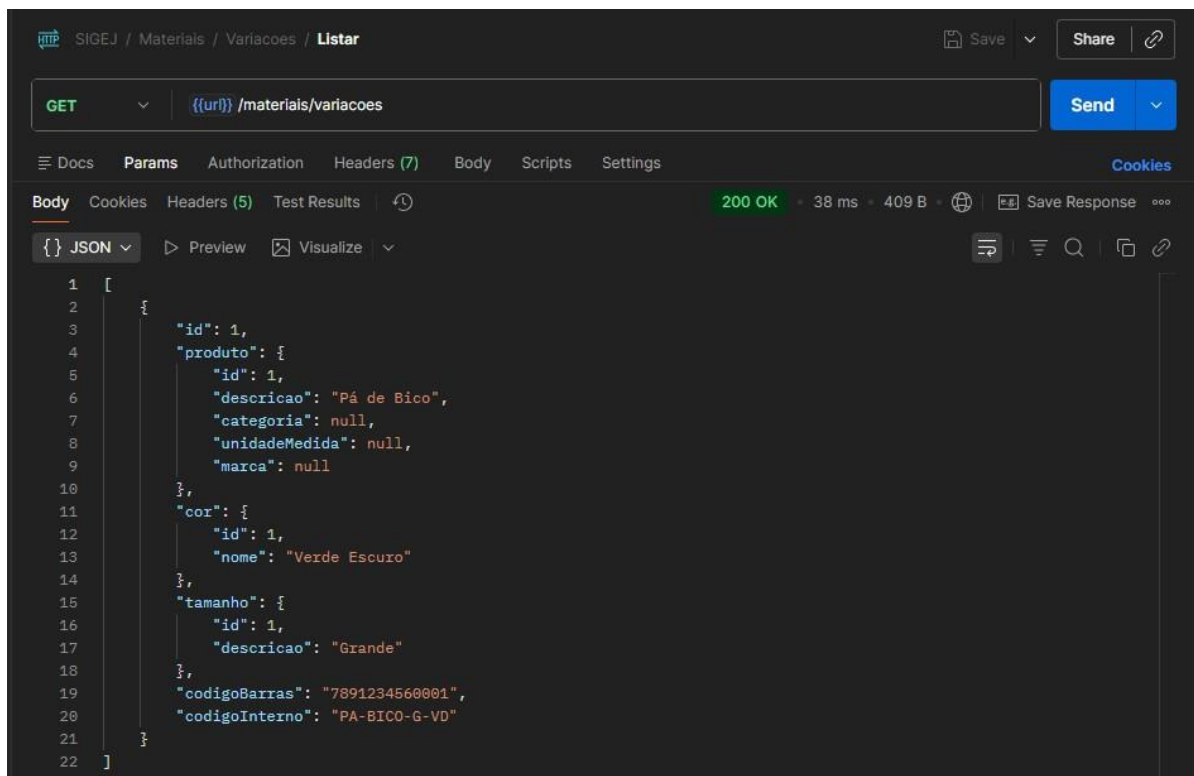
O retorno JSON abaixo exibe os dados do funcionário aninhados com as informações de sua Pessoa, Setor e Cargo, comprovando a integridade das relações.

The screenshot shows a Postman interface for a GET request to the endpoint `/{url}/rh/funcionarios`. The response is a 200 OK status with a response time of 418 ms and a body size of 453 B. The response body is a JSON array containing two employee records. The first record is for Ivan da Silva, and the second record is for a position in the area of Gardening and Landscaping.

```
1 [
2   {
3     "id": 1,
4     "pessoa": {
5       "id": 1,
6       "nome": "Ivan da Silva",
7       "cpf": null,
8       "matriculaSiape": null,
9       "email": null,
10      "telefone": null,
11      "ativo": null
12    },
13    "setor": {
14      "id": 1,
15      "nome": "Jardinagem e Paisagismo",
16      "sigla": null
17    },
18    "tipo": {
19      "id": 1,
20      "descricao": "Tecnico Especializado"
21    },
22    "dataAdmissao": "2025-01-10",
23    "dataDemissao": null
24  }
25 ]
```

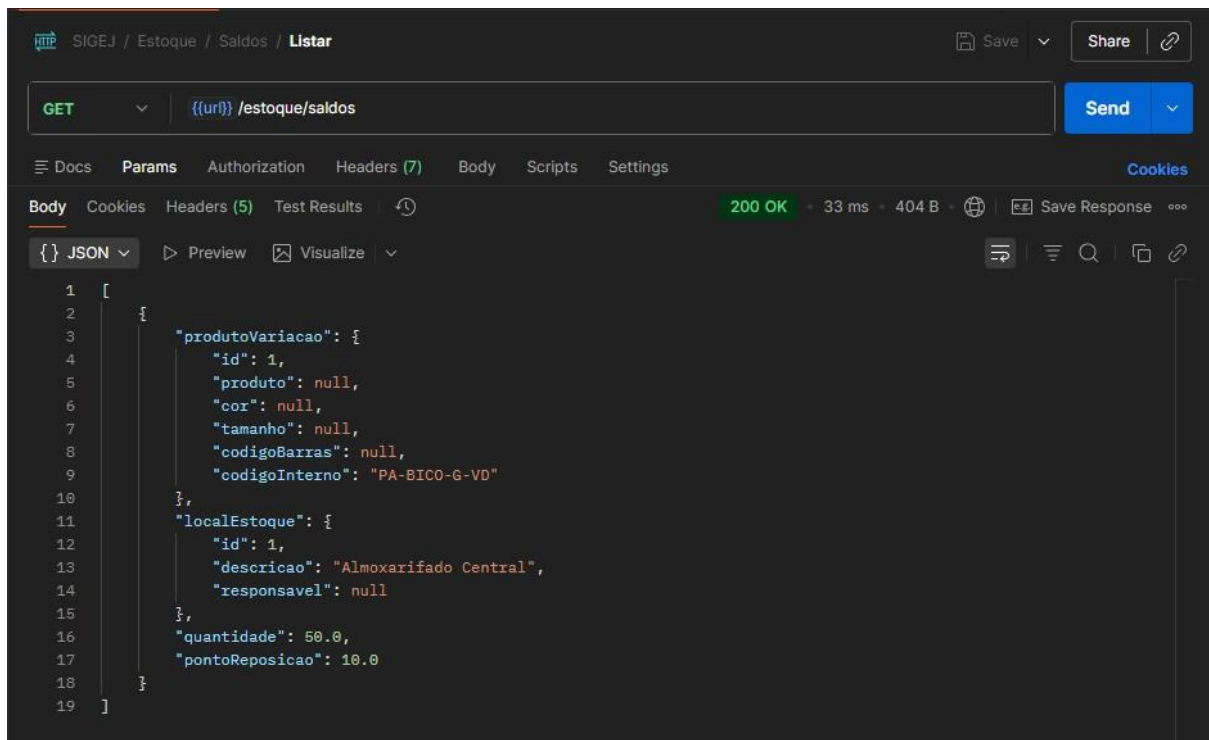
3.2. Módulo de Materiais e Produtos

A imagem abaixo mostra a recuperação de um Produto com suas variações específicas de Cor e Tamanho, essenciais para a gestão precisa do almoxarifado.



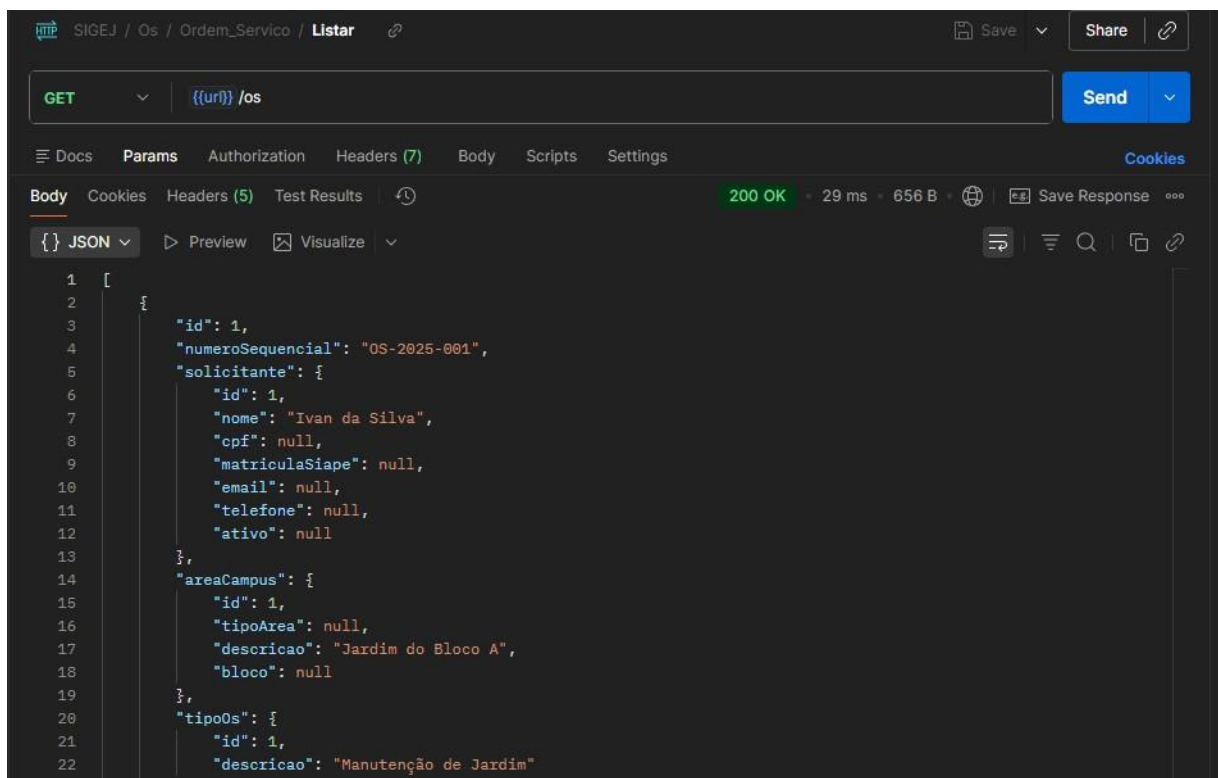
3.3. Controle de Estoque

O módulo de estoque controla o saldo atual de cada item por local de armazenamento. O sistema registra movimentações de entrada e saída e mantém o saldo atualizado, como demonstrado na consulta abaixo que exibe a quantidade disponível no 'Almoxarifado Central'.



3.4. Gestão de Ordens de Serviço (OS)

A Ordem de Serviço consolida as operações do sistema. Os prints abaixo comprovam a abertura de uma OS com prioridade definida, vinculada a um solicitante e a uma área do campus, pronta para execução pelas equipes de manutenção.



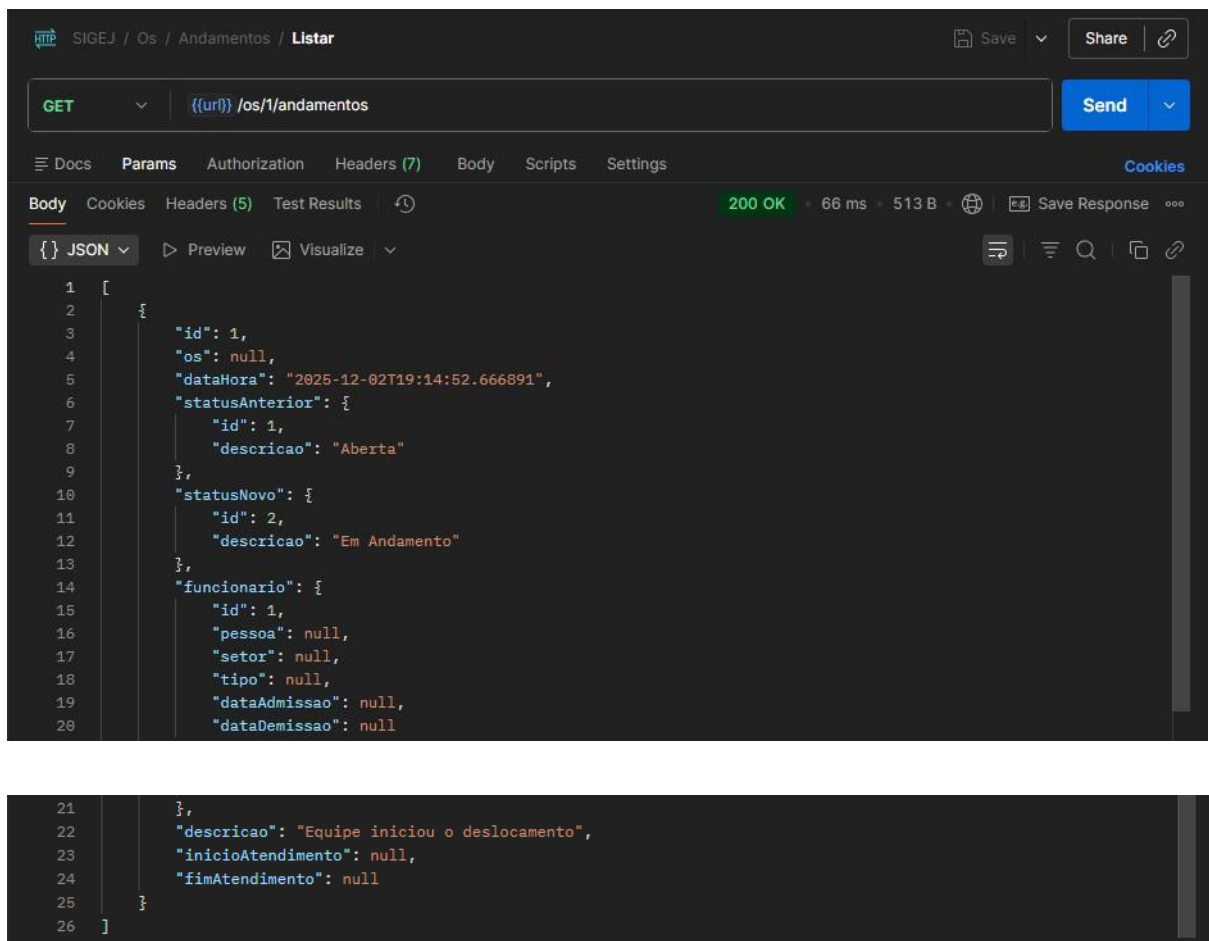
The screenshot shows a REST client interface with a GET request to `{{url}} /os`. The response is a 200 OK status with a 29 ms response time and 656 B of data. The response body is displayed in JSON format, showing a list of service orders. The first object in the list contains the following data:

```
1 [
2   {
3     "id": 1,
4     "numeroSequencial": "OS-2025-001",
5     "solicitante": {
6       "id": 1,
7       "nome": "Ivan da Silva",
8       "cpf": null,
9       "matriculaSiape": null,
10      "email": null,
11      "telefone": null,
12      "ativo": null
13    },
14    "areaCampus": {
15      "id": 1,
16      "tipoArea": null,
17      "descricao": "Jardim do Bloco A",
18      "bloco": null
19    },
20    "tipoOs": {
21      "id": 1,
22      "descricao": "Manutenção de Jardim"
```



The continuation of the JSON response shows the following data for the first object:

```
23    },
24    "equipe": null,
25    "lider": null,
26    "status": {
27      "id": 2,
28      "descricao": "Em Andamento"
29    },
30    "prioridade": 5,
31    "dataAbertura": "2025-12-02T19:14:52.650091",
32    "dataPrevista": null,
33    "descricaoProblema": "Torneira vazando"
34  }
35 ]
```



The screenshot displays a REST client interface with a GET request to the endpoint `/{url}/os/1/andamentos`. The response is a 200 OK status with a 66 ms response time and 513 B of data. The response body is shown in JSON format, containing an array of two objects. The first object represents an open request with details like `id`, `os`, `dataHora`, and `statusAnterior`. The second object represents a request in progress, including `statusNovo` and `funcionario` information.

```
1 [
2   {
3     "id": 1,
4     "os": null,
5     "dataHora": "2025-12-02T19:14:52.666891",
6     "statusAnterior": {
7       "id": 1,
8       "descricao": "Aberta"
9     },
10    "statusNovo": {
11      "id": 2,
12      "descricao": "Em Andamento"
13    },
14    "funcionario": {
15      "id": 1,
16      "pessoa": null,
17      "setor": null,
18      "tipo": null,
19      "dataAdmissao": null,
20      "dataDemissao": null
21    },
22    "descricao": "Equipe iniciou o deslocamento",
23    "inicioAtendimento": null,
24    "fimAtendimento": null
25  }
26 ]
```

4 - CONCLUSÃO

O desenvolvimento do backend do SIGEJ atingiu todos os objetivos propostos. A aplicação foi construída em Java com Spring Boot, utilizando JDBC Template para manipulação direta de SQL, respeitando a restrição de não utilização de frameworks ORM.

O ambiente foi containerizado com Docker (PostgreSQL), garantindo portabilidade, e a automação do banco de dados foi implementada através de scripts SQL (`schema.sql` e `data.sql`) que garantem a recriação limpa e populada do ambiente a cada execução. O sistema atende aos 4 módulos principais (RH, Materiais, Estoque e OS), oferecendo uma API REST completa e testável.