

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ**

**CAMPUS MARACANAÚ**

**CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

**YAN PEDRO FAÇANHA BRASILEIRO**

**Relatório Técnico - Sistema SIGEJ**

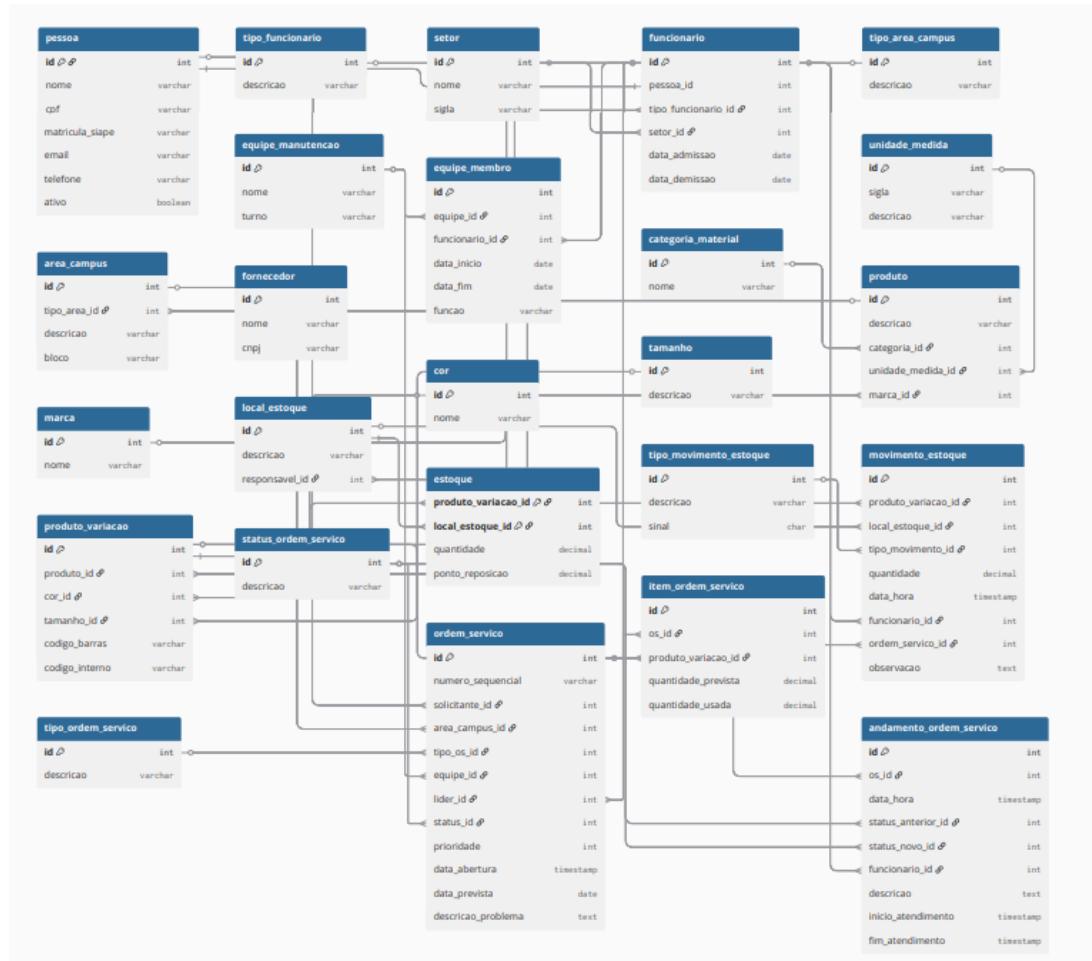
**MARACANAÚ, 2025**

## 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.

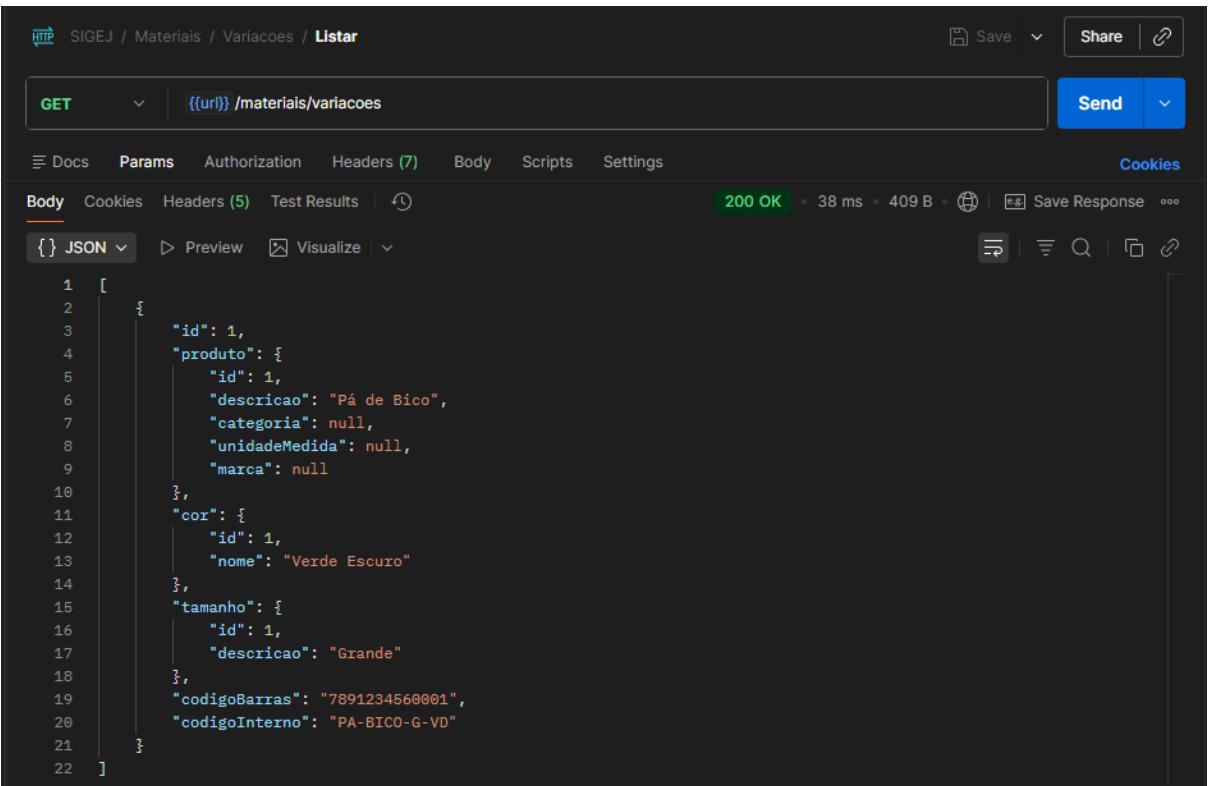
The screenshot shows a Postman interface with the following details:

- URL:** {{url}} /rh/funcionarios
- Method:** GET
- Headers:** Authorization, Headers (7), Body, Scripts, Settings, Cookies
- Query Params:** Key, Value, Description, Bulk Edit
- Body:** JSON, Preview, Visualize
- Response Status:** 200 OK, 418 ms, 453 B, Save Response
- Response Content (JSON):**

```
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  }]
```

### 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.



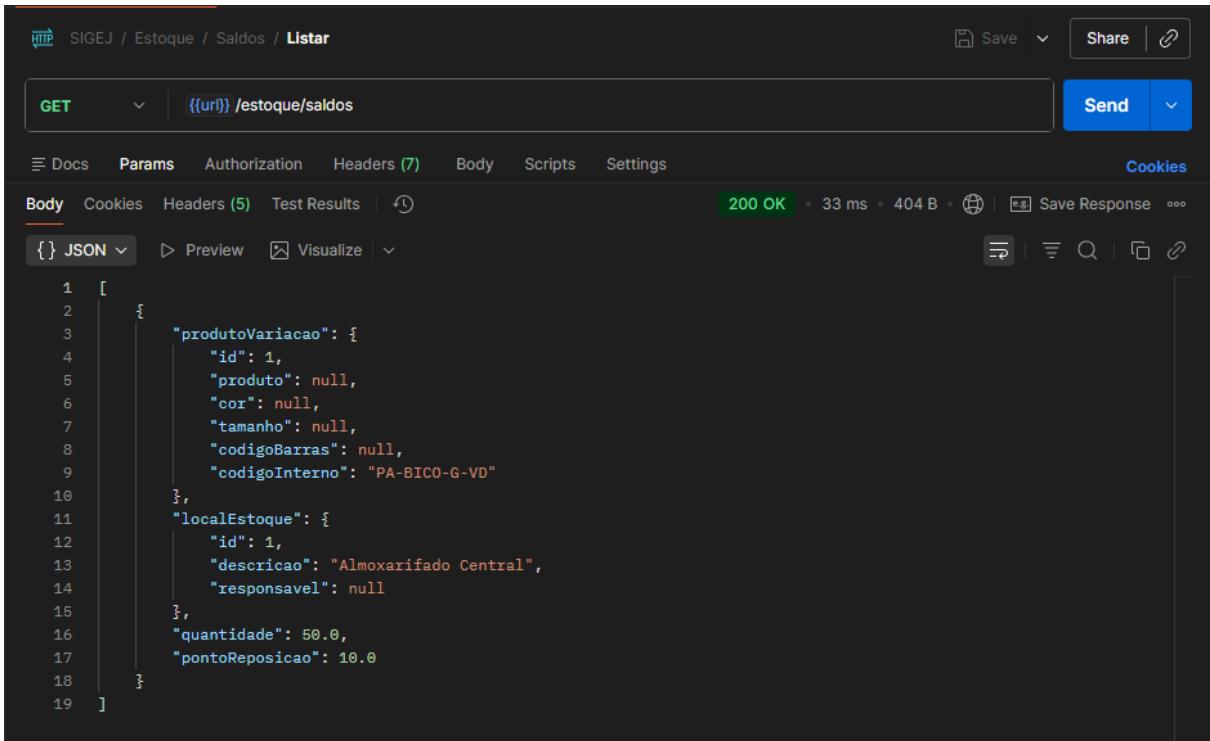
The screenshot shows a REST client interface with the following details:

- URL:** {{url}} /materiais/variacoes
- Method:** GET
- Status:** 200 OK
- Time:** 38 ms
- Size:** 409 B
- Headers:** (5)
- Body:** (JSON)
- Content:** A JSON object representing a product variation, including its product details, color, size, and barcode information.

```
1 [  
2   {  
3     "id": 1,  
4     "produto": {  
5       "id": 1,  
6       "descricao": "P\u00e1 de Bico",  
7       "categoria": null,  
8       "unidadeMedida": null,  
9       "marca": null  
10      },  
11      "cor": {  
12        "id": 1,  
13        "nome": "Verde Escuro"  
14      },  
15      "tamanho": {  
16        "id": 1,  
17        "descricao": "Grande"  
18      },  
19      "codigoBarras": "7891234560001",  
20      "codigoInterno": "PA-BICO-G-VD"  
21    }  
22 ]
```

### 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'.



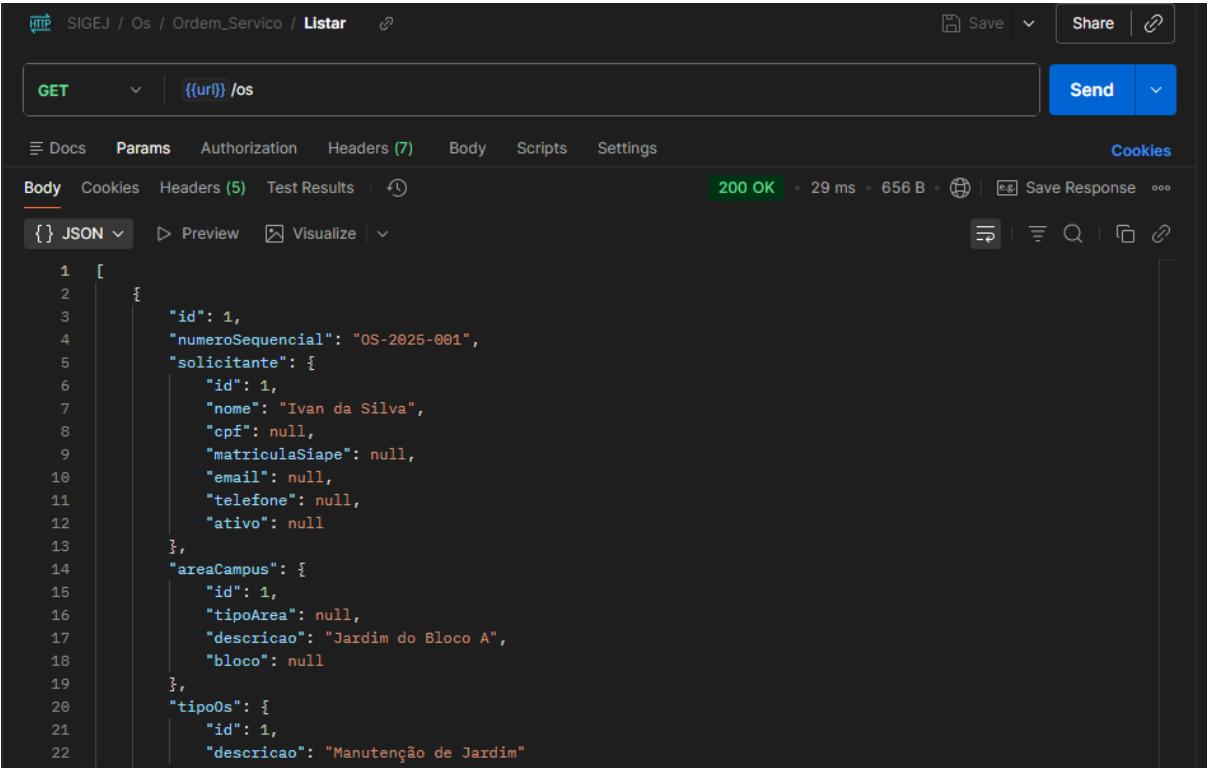
The screenshot shows a REST client interface with the following details:

- HTTP:** SIGEJ / Estoque / Saldos / Listar
- Method:** GET
- URL:** {{url}} /estoque/saldos
- Status:** 200 OK
- Time:** 33 ms
- Size:** 404 B
- Headers:** (7)
- Body:** JSON (Preview and Visualize tabs available)
- Response Content:**

```
1 [  
2   {  
3     "produtoVariacao": {  
4       "id": 1,  
5       "produto": null,  
6       "cor": null,  
7       "tamanho": null,  
8       "codigoBarras": null,  
9       "codigoInterno": "PA-BICO-G-VD"  
10    },  
11    "localEstoque": {  
12      "id": 1,  
13      "descricao": "Almoxarifado Central",  
14      "responsavel": null  
15    },  
16    "quantidade": 50.0,  
17    "pontoReposicao": 10.0  
18  }  
19 ]
```

### 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 API client interface. The URL is `http://SIGEJ/Os/Ordem_Servico>Listar`. The method is `GET`, and the endpoint is `{{url}}/os`. The response status is `200 OK` with a duration of `29 ms` and a size of `656 B`. The response body is a JSON array containing one service order object:

```
1 [  
2   {  
3     "id": 1,  
4     "numeroSeqencial": "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"  
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 ]
```

HTTP SIGEJ / Os / Andamentos / Listar

Save Share

GET {{url}} /os/1/andamentos Send

Docs Params Authorization Headers (7) Body Scripts Settings Cookies

Body Cookies Headers (5) Test Results 200 OK 66 ms 513 B Save Response

{ JSON Preview Visualize

```
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.