

## **Sistema de Gestão de Jardinagem e Manutenção do Campus Maracanaú – SIGEJ**

### **Ciência da Computação – IFCE Campus Maracanaú**

**Data de entrega: 12 de dezembro de 2025**

Olá, pessoal!

Essa é a atividade final da disciplina que vai fechar o semestre com chave de ouro. Vamos construir um sistema real que a Coordenação de Infraestrutura do campus Maracanaú precisa urgentemente para controlar ordens de serviço, equipes e estoque de materiais de jardinagem e manutenção.

Nome do sistema: SIGEJ – Sistema de Informação de Gestão de Jardinagem e Manutenção

Liberdade total de tecnologia (mas sem ORM!)

- Qualquer linguagem: Java, Python, PHP, Node.js, C#, Go, etc.
- Frontend opcional (pode ser só console)
- Backend obrigatório com SQL escrito na mão (INSERT, SELECT, UPDATE, DELETE explícitos)
- Proibido usar qualquer ORM (Hibernate, Eloquent, Sequelize, Prisma, TypeORM, Django ORM...)
- Testes podem ser feitos com uma classe Main/Testes que gera dados mockados e executa todas as operações.

### **Modelo de Dados Completo (25 tabelas)**

```sql

-- 1 a 25 tabelas (script PostgreSQL completo – já testado)

```
CREATE TABLE pessoa (id SERIAL PRIMARY KEY, nome VARCHAR(100) NOT NULL, cpf VARCHAR(11) UNIQUE, matricula_siape VARCHAR(20), email VARCHAR(100), telefone VARCHAR(20), ativo BOOLEAN DEFAULT true);
```

```
CREATE TABLE tipo_funcionario (id SERIAL PRIMARY KEY, descricao VARCHAR(50) NOT NULL);
```

```
CREATE TABLE funcionario (id SERIAL PRIMARY KEY, pessoa_id INT REFERENCES pessoa(id), tipo_funcionario_id INT REFERENCES tipo_funcionario(id), setor_id INT REFERENCES setor(id), data_admissao DATE, data_demissao DATE);
```

```
CREATE TABLE setor (id SERIAL PRIMARY KEY, nome VARCHAR(80) NOT NULL, sigla VARCHAR(10));
```

```
CREATE TABLE tipo_area_campus (id SERIAL PRIMARY KEY, descricao VARCHAR(50));
```

```
CREATE TABLE area_campus (id SERIAL PRIMARY KEY, tipo_area_id INT REFERENCES tipo_area_campus(id), descricao VARCHAR(100) NOT NULL, bloco VARCHAR(10));
```

```
CREATE TABLE equipe_manutencao (id SERIAL PRIMARY KEY, nome VARCHAR(80) NOT NULL, turno VARCHAR(20));
```

```
CREATE TABLE equipe_membro (id SERIAL PRIMARY KEY, equipe_id INT REFERENCES equipe_manutencao(id), funcionario_id INT REFERENCES funcionario(id), data_inicio DATE NOT NULL, data_fim DATE, funcao VARCHAR(30));
```

```
CREATE TABLE categoria_material (id SERIAL PRIMARY KEY, nome VARCHAR(60) NOT NULL);
```

```
CREATE TABLE unidade_medida (id SERIAL PRIMARY KEY, sigla VARCHAR(10) NOT NULL, descricao VARCHAR(50));
```

```
CREATE TABLE fornecedor (id SERIAL PRIMARY KEY, nome VARCHAR(100), cnpj VARCHAR(18));
```

```
CREATE TABLE marca (id SERIAL PRIMARY KEY, nome VARCHAR(80));
```

```
CREATE TABLE produto (id SERIAL PRIMARY KEY, descricao TEXT NOT NULL, categoria_id INT REFERENCES categoria_material(id), unidade_medida_id INT REFERENCES unidade_medida(id), marca_id INT REFERENCES marca(id));
```

```
CREATE TABLE cor (id SERIAL PRIMARY KEY, nome VARCHAR(30));
```

```
CREATE TABLE tamanho (id SERIAL PRIMARY KEY, descricao VARCHAR(30));
```

```
CREATE TABLE produto_variacao (id SERIAL PRIMARY KEY, produto_id INT REFERENCES produto(id), cor_id INT REFERENCES cor(id), tamanho_id INT REFERENCES tamanho(id), codigo_barras VARCHAR(50) UNIQUE, codigo_interno VARCHAR(30));
```

```
CREATE TABLE local_estoque (id SERIAL PRIMARY KEY, descricao VARCHAR(100), responsavel_id INT REFERENCES funcionario(id));
```

```
CREATE TABLE estoque (produto_variacao_id INT REFERENCES produto_variacao(id),  
local_estoque_id INT REFERENCES local_estoque(id), quantidade DECIMAL(10,3)  
DEFAULT 0, ponto_reposicao DECIMAL(10,3) DEFAULT 0, PRIMARY KEY  
(produto_variacao_id, local_estoque_id));
```

```
CREATE TABLE tipo_movimento_estoque (id SERIAL PRIMARY KEY, descricao  
VARCHAR(50), sinal CHAR(1) CHECK (sinal IN ('+', '-')));
```

```
CREATE TABLE movimento_estoque (id SERIAL PRIMARY KEY, produto_variacao_id INT  
REFERENCES produto_variacao(id), local_estoque_id INT REFERENCES  
local_estoque(id), tipo_movimento_id INT REFERENCES tipo_movimento_estoque(id),  
quantidade DECIMAL(10,3) NOT NULL, data_hora TIMESTAMP DEFAULT NOW(),  
funcionario_id INT REFERENCES funcionario(id), ordem_servico_id INT, observacao  
TEXT);
```

```
CREATE TABLE tipo_ordem_servico (id SERIAL PRIMARY KEY, descricao VARCHAR(80));
```

```
CREATE TABLE status_ordem_servico (id SERIAL PRIMARY KEY, descricao  
VARCHAR(50));
```

```
CREATE TABLE ordem_servico (id SERIAL PRIMARY KEY, numero_sequencial  
VARCHAR(20) UNIQUE, solicitante_id INT REFERENCES pessoa(id), area_campus_id INT  
REFERENCES area_campus(id), tipo_os_id INT REFERENCES tipo_ordem_servico(id),  
equipe_id INT REFERENCES equipe_manutencao(id), lider_id INT REFERENCES  
funcionario(id), status_id INT REFERENCES status_ordem_servico(id) DEFAULT 1,  
prioridade INT CHECK (prioridade IN (1,2,3,4,5)), data_abertura TIMESTAMP DEFAULT  
NOW(), data_prevista DATE, descricao_problema TEXT);
```

```
CREATE TABLE item_ordem_servico (id SERIAL PRIMARY KEY, os_id INT REFERENCES  
ordem_servico(id) ON DELETE CASCADE, produto_variacao_id INT REFERENCES  
produto_variacao(id), quantidade_prevista DECIMAL(10,3), quantidade_usada  
DECIMAL(10,3));
```

```
CREATE TABLE andamento_ordem_servico (id SERIAL PRIMARY KEY, os_id INT  
REFERENCES ordem_servico(id) ON DELETE CASCADE, data_hora TIMESTAMP  
DEFAULT NOW(), status_anterior_id INT REFERENCES status_ordem_servico(id),  
status_novo_id INT REFERENCES status_ordem_servico(id), funcionario_id INT  
REFERENCES funcionario(id), descricao TEXT, inicio_atendimento TIMESTAMP,  
fim_atendimento TIMESTAMP);
```

...

## Consultas SQL Obrigatórias (prontas para copiar)

```
```sql
```

```
-- 1. Ordens de serviço em aberto por prioridade e área
```

```
SELECT os.id, os.numero_sequencial, os.prioridade, ac.descricao AS area, tos.descricao  
AS tipo_servico, p.nome AS solicitante, os.data_abertura
```

```
FROM ordem_servico os
```

```
JOIN area_campus ac ON os.area_campus_id = ac.id
```

```
JOIN tipo_ordem_servico tos ON os.tipo_os_id = tos.id
```

```
JOIN status_ordem_servico sts ON os.status_id = sts.id
```

```
JOIN pessoa p ON os.solicitante_id = p.id
```

```
WHERE sts.descricao IN ('aberta', 'em_atendimento', 'aguardando_material')
```

```
ORDER BY os.prioridade ASC, os.data_abertura ASC;
```

```
```
```

```
```sql
```

```
-- 2. Materiais abaixo do ponto de reposição
```

```
SELECT p.descricao, pv.codigo_interno, le.descricao AS local, e.quantidade,  
e.ponto_reposicao
```

```
FROM estoque e
```

```
JOIN produto_variacao pv ON e.produto_variacao_id = pv.id
```

```
JOIN produto p ON pv.produto_id = p.id
```

```
JOIN local_estoque le ON e.local_estoque_id = le.id
```

```
WHERE e.quantidade < e.ponto_reposicao;
```

```
```
```

```
```sql
```

```
-- 3. Timeline de uma OS (exemplo OS = 1)
```

```
SELECT a.data_hora, pes.nome AS funcionario, sts_novo.descricao AS status_atual,  
a.descricao
```

```
FROM andamento_ordem_servico a
```

```
JOIN funcionario f ON a.funcionario_id = f.id
```

```
JOIN pessoa pes ON f.pessoa_id = pes.id
```

```
JOIN status_ordem_servico sts_novo ON a.status_novo_id = sts_novo.id
```

```
WHERE a.os_id = 1
```

```
ORDER BY a.data_hora;
```

```
```
```

```
```sql
```

```
-- 4. Consumo por equipe em um período
```

```
SELECT eq.nome AS equipe, p.descricao, SUM(me.quantidade) AS total
```

```
FROM movimento_estoque me
```

```
JOIN produto_variacao pv ON me.produto_variacao_id = pv.id
```

```
JOIN produto p ON pv.produto_id = p.id
```

```
JOIN ordem_servico os ON me.ordem_servico_id = os.id
```

```
JOIN equipe_manutencao eq ON os.equipe_id = eq.id
```

```
WHERE me.data_hora BETWEEN '2025-10-01' AND '2025-10-31'
```

```
GROUP BY eq.nome, p.descricao;
```

```
```
```

```
```sql
```

```
-- 5. OS concluídas por tipo no ano
```

```
SELECT tos.descricao, COUNT(*) AS total
```

```
FROM ordem_servico os
```

```
JOIN tipo_ordem_servico tos ON os.tipo_os_id = tos.id
```

```
JOIN status_ordem_servico sts ON os.status_id = sts.id
```

```
WHERE sts.descricao = 'concluída' AND EXTRACT(YEAR FROM os.data_abertura) = 2025
```

```
GROUP BY tos.descricao;
```

```
```
```

#### ### 4. Entrega

- Um ZIP com: código-fonte, diagrama DER, PDF com capa + prints + relatórios
- Quem entregar tudo funcionando e bonitinho leva 10 + elogios!

Dúvidas? Grupo da disciplina ou chama no wpp.

Bons códigos e sucesso!

IFCE Campus Maracanaú – Novembro/2025