📋 Guia Completo de Desenvolvimento - Sistema de Gestão de Materiais Odontológicos

🎯 Visão Geral do Projeto

Sistema para gerenciar materiais odontológicos com controle de estoque, movimentações e relatórios integrado a um sistema de clínica odontológica.

🔧 FASE 1: PREPARAÇÃO E SETUP (1-2 dias)

1.1 Configuração do Ambiente

- [ ] Instalar Java 17+, Maven, Node.js
- [ ] Configurar IDE (IntelliJ/VSCode)
- [ ] Configurar banco de dados (MySQL/PostgreSQL)
- [ ] Criar repositório Git

1.2 Setup do Backend (Spring Boot)

```
# Criar projeto Spring Boot
mvn archetype:generate -DgroupId=com.clinica.odonto \
  -DartifactId=odonto-backend -DarchetypeArtifactId=maven-archetype-quickstart
```

Dependências necessárias (pom.xml):

- spring-boot-starter-web
- spring-boot-starter-data-jpa
- spring-boot-starter-security
- spring-boot-starter-validation
- mysql-connector-java
- jjwt-api, jjwt-impl, jjwt-jackson

1.3 Setup do Frontend (React + Vite)

```
npm create vite@latest frontend -- --template react
cd frontend
npm install @mui/material @emotion/react @emotion/styled
npm install @mui/x-date-pickers date-fns
npm install axios react-router-dom
```

🏗️ FASE 2: BACKEND - ESTRUTURA BASE (3-4 dias)

2.1 Configuração Inicial

- [ ] **application.properties**: Configurar banco, JPA, security
- [ ] **SecurityConfig**: JWT, CORS, endpoints públicos
- [ ] **JwtUtil**: Geração e validação de tokens
- [ ] **DataLoader**: Usuários padrão (admin, dentista, recepcionista)

2.2 Entidades Base (se não existirem)

```
// Ordem de criação:
1. Usuario (base para autenticação)
2. Paciente
3. Dentista
4. Consulta
```

2.3 Sistema de Autenticação

- [ ] **AuthController**: login, refresh token
- [ ] **UserDetailsService**: carregamento de usuários
- [ ] **JwtAuthenticationFilter**: interceptação de requests

FASE 3: BACKEND - MÓDULO DE MATERIAIS (5-7 dias)

3.1 Entidades do Módulo de Materiais

3.1.1 Material.java

```
@Entity
public class Material {
    private Long id;
    private String nome;
    private String codigo;
    private String categoria;
    private String unidadeMedida;
    private Integer estoqueAtual;
    private Integer estoqueMinimo;
    private BigDecimal precoUnitario;
    private Boolean ativo;
    // timestamps
}
```

3.1.2 TipoMovimentacao.java (Enum)

```
public enum TipoMovimentacao {
    ENTRADA, SAIDA, AJUSTE
}
```

3.1.3 MovimentacaoMaterial.java

```
@Entity
public class MovimentacaoMaterial {
    private Long id;
    private Material material;
    private TipoMovimentacao tipo;
    private Integer quantidade;
    private LocalDateTime dataMovimentacao;
    private String observacoes;
    private Usuario usuario;
    private Consulta consulta; // opcional
}
```

3.1.4 MaterialConsulta.java

```
@Entity
public class MaterialConsulta {
    private Long id;
    private Material material;
    private Consulta consulta;
    private Integer quantidadeUtilizada;
    private LocalDateTime dataUtilizacao;
    private Usuario usuarioLancamento;
}
```

3.2 Repositories

```
// MaterialRepository.java
List<Material> findByAtivoTrueOrderByNome();
List<Material> findByEstoqueAtualLessThanEstoqueMinimo();

// MovimentacaoMaterialRepository.java
List<MovimentacaoMaterial> findByDataMovimentacaoBetweenOrderByDataMovimentacaoDesc(
    LocalDateTime inicio, LocalDateTime fim);

// MaterialConsultaRepository.java
List<MaterialConsulta> findByConsultaOrderByDataUtilizacaoDesc(Consulta consulta);
```

3.3 DTOs (Request/Response)

- [ ] **MaterialRequest/Response**

- [ ] **MovimentacaoMaterialRequest/Response**
- [ ] **MaterialConsultaRequest/Response**

3.4 Services

- [ ] **MaterialService**: CRUD, controle de estoque
- [ ] **MovimentacaoMaterialService**: movimentações, relatórios
- [ ] **MaterialConsultaService**: lançamentos em consultas

3.5 Controllers

- [ ] **MaterialController**: endpoints CRUD
- [ ] **MovimentacaoMaterialController**: movimentações e relatórios
- [ ] **MaterialConsultaController**: lançamentos

FASE 4: FRONTEND - ESTRUTURA BASE (2-3 dias)

4.1 Configuração Inicial

- [ ] **AuthContext**: gerenciamento de autenticação
- [ ] **api.js**: configuração do Axios
- [ ] **Router**: rotas protegidas
- [ ] **Layout**: sidebar, header, navegação

4.2 Services

```
// materialService.js
export const materialService = {
  getAll: () => api.get('/api/materiais'),
  create: (data) => api.post('/api/materiais', data),
  update: (id, data) => api.put(`/api/materiais/${id}`, data),
  delete: (id) => api.delete(`/api/materiais/${id}`)
};
```

FASE 5: FRONTEND - PÁGINAS DE MATERIAIS (4-5 dias)

5.1 Página de Gestão de Materiais

Componentes necessários:

- [ ] **Lista de materiais** (DataGrid/Table)
- [ ] **Filtros** (nome, categoria, estoque baixo)
- [ ] **Modal de cadastro/edição**
- [ ] **Confirmação de exclusão**
- [ ] **Indicadores visuais** (estoque baixo)

Funcionalidades:

```
// Estados principais
const [materiais, setMateriais] = useState([]);
const [filtros, setFiltros] = useState({});
const [modalAberto, setModalAberto] = useState(false);
const [materialSelecionado, setMaterialSelecionado] = useState(null);
```

5.2 Página de Movimentações

- [ ] **Formulário de movimentação** (entrada/saída/ajuste)
- [ ] **Histórico de movimentações**
- [ ] **Filtros por período e tipo**

5.3 Integração com Consultas

- [ ] **Modal de lançamento** na página de consultas
- [ ] **Lista de materiais utilizados** por consulta
- [ ] **Atualização automática** do estoque

FASE 6: RELATÓRIOS (3-4 dias)

6.1 Backend - Endpoints de Relatórios

```
// Relatórios necessários
@GetMapping("/relatorio/estoque-baixo")
@GetMapping("/relatorio/movimentacoes/periodo")
@GetMapping("/relatorio/materiais-mais-utilizados")
@GetMapping("/relatorio/custo-por-consulta")
```

6.2 Frontend - Página de Relatórios

- [ ] **Seletor de período**
- [ ] **Gráficos** (Chart.js/Recharts)
- [ ] **Tabelas resumo**
- [ ] **Exportação** (PDF/Excel)

FASE 7: TESTES E VALIDAÇÕES (2-3 dias)

7.1 Testes Backend

```
// Testes unitários
@Test
void deveAtualizarEstoqueAposSaida() {
    // implementar teste
}

@Test
void deveValidarEstoqueInsuficiente() {
    // implementar teste
}
```

7.2 Testes Frontend

- [ ] **Testes de componentes** (Jest/React Testing Library)
- [ ] **Testes de integração** com API
- [ ] **Testes E2E** (Cypress - opcional)

7.3 Validações

- [ ] **Regras de negócio**: estoque não pode ficar negativo
- [ ] **Validações de formulário**: campos obrigatórios
- [ ] **Tratamento de erros**: mensagens amigáveis

FASE 8: DEPLOY E DOCUMENTAÇÃO (1-2 dias)

8.1 Preparação para Deploy

- [ ] **Profiles** de ambiente (dev, prod)
- [ ] **Variáveis de ambiente**
- [ ] **Build de produção**

8.2 Documentação

- [ ] **README.md**: instruções de instalação
- [ ] **API Documentation**: Swagger/OpenAPI
- [ ] **Manual do usuário**: funcionalidades principais

CRONOGRAMA SUGERIDO (20-25 dias)

Semana	Fases	Foco
Semana 1	Fases 1-2	Setup + Backend Base
Semana 2	Fase 3	Módulo de Materiais (Backend)
Semana 3	Fases 4-5	Frontend + Páginas

Semana 4 Fases 6-8 Relatórios + Testes + Deploy

🛠️ FERRAMENTAS RECOMENDADAS

Desenvolvimento

- **Backend**: IntelliJ IDEA, Postman
- **Frontend**: VSCode, React DevTools
- **Banco**: MySQL Workbench, DBeaver

Produtividade

- **Versionamento**: Git + GitHub
- **Gerenciamento**: Trello, Notion
- **Comunicação**: Slack, Discord

🎯 DICAS IMPORTANTES

Ordem de Desenvolvimento

1. **Sempre backend primeiro**: APIs funcionais antes do frontend
2. **Testes incrementais**: testar cada funcionalidade isoladamente
3. **Commits frequentes**: pequenas alterações, commits descritivos
4. **Documentação contínua**: documentar durante o desenvolvimento

Boas Práticas

- **Validação dupla**: frontend + backend
- **Tratamento de erros**: sempre prever cenários de falha
- **Performance**: paginação, lazy loading
- **Segurança**: validação de permissões, sanitização

Pontos de Atenção

- **Controle de estoque**: transações atômicas
- **Concorrência**: múltiplos usuários alterando estoque
- **Auditoria**: log de todas as movimentações
- **Backup**: estratégia de backup dos dados

📋 CHECKLIST FINAL

Backend Completo

- [ ] Todas as entidades criadas e mapeadas
- [ ] Repositories com queries necessárias
- [ ] Services com regras de negócio
- [ ] Controllers com endpoints REST
- [ ] Validações e tratamento de erros
- [ ] Testes unitários implementados

Frontend Completo

- [ ] Páginas de gestão implementadas
- [ ] Integração com APIs funcionando
- [ ] Validações de formulário
- [ ] Tratamento de erros e loading
- [ ] Interface responsiva
- [ ] Testes de componentes

Sistema Integrado

- [ ] Autenticação e autorização
- [ ] Controle de estoque funcionando
- [ ] Relatórios gerando dados corretos
- [ ] Performance adequada

- [ ] Deploy realizado com sucesso

Este guia fornece uma base sólida para desenvolver o sistema completo de forma organizada e eficiente! 🚀

Documento gerado em: Janeiro 2025
Versão: 1.0
Projeto: Sistema de Gestão de Materiais Odontológicos