Documentação Técnica - Sistema PLASA

Marinha do Brasil - PAPEM

---

1. Visão Geral

1.1 Descrição

Sistema para visualização automatizada de documentos PLASA (Plano de Serviço) e Escalas de Serviço da Marinha do Brasil, com gerenciamento centralizado de avisos importantes.

1.2 Objetivos

- Exibir PLASA com scroll automático contínuo
- Alternar entre escalas de Oficiais e Praças
- Gerenciar avisos com prioridades e períodos de validade
- Converter PDFs para imagens para máxima compatibilidade
- Interface administrativa para gestão completa

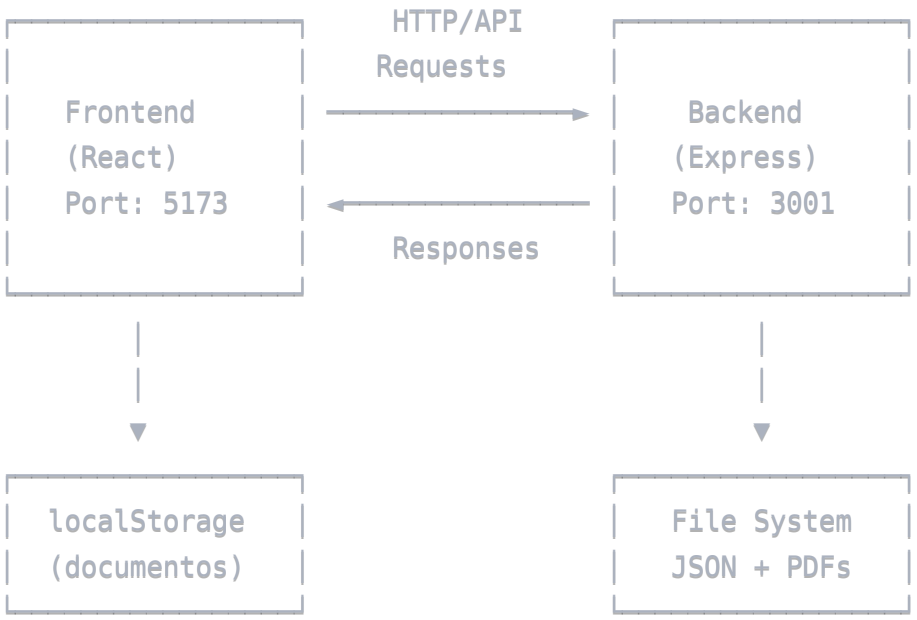
1.3 Características Principais

- **Frontend**: React 18 + TypeScript + Tailwind CSS
- **Backend**: Node.js + Express + Multer
- **Dados**: JSON files (avisos) + arquivos físicos (documentos)
- **Conversão**: PDF.js para renderização de páginas
- **Cache**: Sistema de páginas convertidas para performance

---

2. Arquitetura do Sistema

2.1 Visão Geral da Arquitetura



2.2 Componentes Principais

Frontend (React)

- **PDFViewer**: Visualização e scroll automático
- **NoticeDisplay**: Exibição rotativa de avisos
- **Admin**: Painel administrativo
- **DisplayContext**: Estado global da aplicação

Backend (Express)

- **Upload Handler**: Gerenciamento de arquivos
- **Notice API**: CRUD de avisos
- **PDF Converter**: Conversão páginas para imagem
- **Static Server**: Servir arquivos convertidos

---

3. Estrutura de Arquivos

```
sistema-plasa/
├── frontend/
│   ├── src/
│   │   ├── components/
│   │   │   ├── PDFViewer.tsx          # Visualizador principal
│   │   │   ├── NoticeDisplay.tsx      # Exibição de avisos
│   │   │   └── ui/                    # Componentes shadcn/ui
│   │   ├── context/
│   │   │   └── DisplayContext.tsx     # Estado global
│   │   ├── pages/
│   │   │   ├── Index.tsx              # Página de visualização
│   │   │   ├── Admin.tsx              # Painel administrativo
│   │   │   └── NotFound.tsx           # Página 404
│   │   ├── hooks/
│   │   │   └── use-toast.ts           # Hook para notificações
│   │   └── App.tsx                    # Componente raiz
│   ├── .env                           # Variáveis de ambiente
│   ├── package.json
│   ├── tailwind.config.js
│   └── vite.config.ts
└── backend/
    ├── server.js                      # Servidor principal
    ├── public/
    │   ├── uploads/                   # PDFs enviados
    │   │   ├── plasa-*.pdf
    │   │   └── escala-*.pdf
    │   └── plasa-pages/               # Páginas convertidas
    │       ├── plasa-page-1.jpg
    │       └── plasa-page-*.jpg
    ├── data/
    │   ├── notices.json               # Avisos persistidos
    │   └── config.json                # Configurações
    ├── package.json
    └── README.md
```

---

4. API Endpoints

4.1 Avisos (Notices)

GET /api/notices

Listar todos os avisos cadastrados.

Resposta:

```json
{
  "success": true,
  "notices": [
    {
      "id": "notice-1234567890-abc123",
      "title": "Reunião Mensal",
      "content": "Reunião mensal marcada para sexta-feira às 14h",
      "priority": "high",
      "startDate": "2025-06-01T00:00:00.000Z",
      "endDate": "2025-06-05T23:59:59.000Z",
      "active": true,
      "createdAt": "2025-06-01T10:00:00.000Z",
      "updatedAt": "2025-06-01T10:00:00.000Z"
    }
  ],
  "total": 1,
  "active": 1
}
```

POST /api/notices

Criar novo aviso.

Payload:

```json
{
  "title": "Título do Aviso",
  "content": "Conteúdo detalhado do aviso",
  "priority": "high|medium|low",
  "startDate": "2025-06-01T00:00:00.000Z",
  "endDate": "2025-06-05T23:59:59.000Z",
  "active": true
}
```

PUT /api/notices/:id

Atualizar aviso existente.

DELETE /api/notices/:id

Remover aviso.

4.2 Documentos

POST /api/upload-pdf

Upload de documento PDF ou imagem.

FormData:

```
pdf: <file>
documentType: "plasa|escala"
title: "Título do documento"
category: "oficial|praca" (apenas para escalas)
```

Resposta:

```
json
{
  "success": true,
  "message": "Documento enviado com sucesso",
  "data": {
    "url": "/uploads/plasa-1234567890-documento.pdf",
    "fullUrl": "http://localhost:3001/uploads/plasa-1234567890-documento.pdf",
    "fileName": "plasa-1234567890-documento.pdf",
    "originalName": "documento.pdf",
    "documentType": "plasa",
    "title": "PLASA Junho 2025",
    "size": 2048576,
    "mimeType": "application/pdf",
    "uploadedAt": "2025-06-01T10:30:00.000Z"
  }
}
```

GET /api/list-pdfs

Listar todos os documentos enviados.

DELETE /api/delete-pdf/:filename

Remover documento específico.

4.3 Páginas PLASA

POST /api/upload-plasa-page

Upload de página convertida (uso interno).

POST /api/check-plasa-pages

Verificar se páginas convertidas existem.

Payload:

```json
{
  "totalPages": 8
}
```

Resposta:

```json
{
  "success": true,
  "allPagesExist": true,
  "pageUrls": [
    "/plasa-pages/plasa-page-1.jpg",
    "/plasa-pages/plasa-page-2.jpg"
  ],
  "totalFound": 8,
  "totalExpected": 8
}
```

DELETE /api/clear-plasa-pages

Limpar cache de páginas convertidas.

4.4 Sistema

GET /api/status

Status básico do servidor.

GET /api/system-info

Informações detalhadas do sistema.

---

5. Modelos de Dados

5.1 Notice (Aviso)

typescript

```typescript
interface Notice {
  id: string;                    // ID único gerado automaticamente
  title: string;                 // Título do aviso
  content: string;               // Conteúdo detalhado
  priority: "high" | "medium" | "low";  // Prioridade de exibição
  startDate: Date;               // Data de início da validade
  endDate: Date;                 // Data de fim da validade
  active: boolean;               // Se está ativo para exibição
  createdAt?: Date;              // Data de criação
  updatedAt?: Date;              // Data da última atualização
}
```

5.2 PDFDocument

typescript

```typescript
interface PDFDocument {
  id: string;                    // ID único local
  title: string;                 // Título descritivo
  url: string;                   // URL do arquivo
  type: "plasa" | "escala";      // Tipo do documento
  category?: "oficial" | "praca"; // Categoria (apenas escalas)
  uploadDate: Date;              // Data do upload
  active: boolean;               // Se está ativo para exibição
}
```

5.3 Display Context State

typescript

```typescript
interface DisplayContextType {
  notices: Notice[];                    // Lista de avisos
  plasaDocuments: PDFDocument[];        // Documentos PLASA
  escalaDocuments: PDFDocument[];       // Documentos de Escala
  activePlasaDoc: PDFDocument | null;   // PLASA ativo atual
  activeEscalaDoc: PDFDocument | null;  // Escala ativa atual
  currentEscalaIndex: number;           // Índice de alternância
  documentAlternateInterval: number;    // Intervalo em ms
  scrollSpeed: "slow" | "normal" | "fast"; // Velocidade do scroll
  autoRestartDelay: number;             // Delay para reinício
  isLoading: boolean;                   // Estado de carregamento

  // Métodos
  addNotice: (notice) => Promise<boolean>;
  updateNotice: (notice) => Promise<boolean>;
  deleteNotice: (id) => Promise<boolean>;
  addDocument: (document) => void;
  updateDocument: (document) => void;
  deleteDocument: (id) => void;
  setDocumentAlternateInterval: (interval) => void;
  setScrollSpeed: (speed) => void;
  setAutoRestartDelay: (delay) => void;
  refreshNotices: () => Promise<void>;
}
```

---

6. Fluxos de Processo

6.1 Upload e Exibição de PLASA

1. Admin seleciona PDF → Admin.tsx
2. handleDocumentSubmit() → FormData
3. POST /api/upload-pdf → server.js
4. Multer salva em /public/uploads/ → filesystem
5. addDocument() → DisplayContext
6. localStorage persiste estado → browser
7. PDFViewer detecta novo PLASA → React
8. convertPDFToImages() → PDF.js
9. Canvas renderiza páginas → browser
10. POST /api/upload-plasa-page → server.js
11. Salva JPGs em /public/plasa-pages/ → filesystem
12. Inicia scroll automático → ContinuousAutoScroller

6.2 Gerenciamento de Avisos

```
1. Admin preenche formulário → Admin.tsx
2. handleNoticeSubmit() → validation
3. convertLocalNoticeToServer() → format
4. POST /api/notices → server.js
5. Validação servidor → validation
6. Salva em /data/notices.json → filesystem
7. convertServerNoticeToLocal() → format
8. setNotices() atualiza estado → React
9. NoticeDisplay exibe avisos → render
```

6.3 Alternância de Escalas

```
1. Timer inicia → useEffect
2. setInterval() → JavaScript
3. currentEscalaIndex++ → state
4. activeEscalaDoc recalculado → computed
5. PDFViewer re-renderiza → React
6. Nova escala exibida → DOM
```

---

7. Configuração e Instalação

7.1 Pré-requisitos

- Node.js 18+
- npm 8+
- Navegador moderno (Chrome, Firefox, Edge)

7.2 Variáveis de Ambiente

Frontend (.env)

env

```
VITE_BACKEND_HOST=localhost
VITE_BACKEND_PORT=3001
```

Backend (opcional)

env

```
PORT=3001
NODE_ENV=development
```

7.3 Instalação Backend

bash

```
cd backend
npm install
npm start
```

Saída esperada:

```
🚀 SERVIDOR PLASA v2.3 INICIADO (CORS TOTAL + AVISOS)
🌐 Porta: 3001
📢 Avisos: /path/to/data/notices.json
✅ Servidor com CORS e gerenciamento de avisos!
```

7.4 Instalação Frontend

bash

```
cd frontend
npm install
npm run dev
```

URLs:

- **Frontend**: http://localhost:5173
- **Admin**: http://localhost:5173/admin
- **Backend**: http://localhost:3001

7.5 Dependências Principais

Backend

json

```
{
  "express": "^4.21.2",
  "cors": "^2.8.5",
  "multer": "^1.4.5-lts.1"
}
```

Frontend

```json
{
  "react": "^18.0.0",
  "@tanstack/react-query": "latest",
  "react-router-dom": "latest",
  "tailwindcss": "latest"
}
```

---

8. Recursos e Funcionalidades

8.1 Visualização Principal (Index.tsx)

Layout Responsivo

- **60%**: PLASA com scroll automático
- **40%**: Dividido entre Escala (65%) e Avisos (35%)

PLASA Features

- Conversão automática PDF → Imagem
- Scroll contínuo configurável (lento/normal/rápido)
- Reinício automático após pausa configurável
- Cache de páginas para performance
- Controles de pausa/retomar (overlay)

Escalas Features

- Alternância automática entre categorias
- Suporte a Oficiais e Praças
- Intervalo configurável (10-300 segundos)
- Indicador visual de alternância

Avisos Features

- Rotação baseada em prioridade
- Destaque visual por prioridade (cores)
- Período de validade automático
- Sincronização em tempo real

8.2 Painel Administrativo (Admin.tsx)

Aba Avisos

- Formulário de criação com validação
- Lista com edição inline
- Ativação/desativação rápida
- Status de sincronização (servidor/local)

Aba Documentos

- Upload drag-and-drop
- Preview de documentos
- Categorização automática
- Gestão de documentos ativos

Aba Configurações

- Velocidade de scroll do PLASA
- Intervalo de alternância de escalas
- Delay de reinício automático
- Ferramentas de manutenção

Aba Debug

- Informações técnicas do sistema
- Testes de conectividade
- Estatísticas de uso
- Logs e troubleshooting

---

9. Performance e Cache

9.1 Sistema de Cache

Cache de Páginas PLASA

```javascript
// Verificação antes de converter
const checkExistingPages = async (totalPages) => {
  const response = await fetch('/api/check-plasa-pages', {
    method: 'POST',
    body: JSON.stringify({ totalPages })
  });

  const result = await response.json();
  if (result.allPagesExist) {
    return result.pageUrls; // Usar cache
  }

  return null; // Converter novamente
};
```

Cache Local (localStorage)

- Documentos e configurações persistidos
- Avisos são sempre do servidor
- Sincronização automática na inicialização

9.2 Otimizações

Conversão PDF

- Renderização em background
- Scale otimizado para qualidade/tamanho
- Compressão JPEG automática
- Timeout para PDFs problemáticos

Rede

- CORS totalmente configurado
- Gzip compression
- Cache headers apropriados
- Retry automático para falhas

---

10. Segurança

10.1 Upload de Arquivos

Validações Implementadas

```javascript
// Tipos MIME permitidos
const allowedTypes = [
  'application/pdf',
  'image/jpeg',
  'image/png',
  'image/gif',
  'image/webp'
];

// Limites de tamanho
const limits = {
  fileSize: 50 * 1024 * 1024,    // 50MB documentos
  fieldSize: 10 * 1024 * 1024    // 10MB páginas
};

// Sanitização de nomes
const sanitizedName = filename.replace(/[^a-zA-Z0-9.-]/g, '_');
```

Proteções

- Validação dupla (cliente + servidor)
- Sanitização de nomes de arquivo
- Verificação de extensão e MIME type
- Limite de tamanho rigoroso
- Diretórios isolados

10.2 CORS e Headers

Configuração CORS

```javascript
app.use(cors({
  origin: function (origin, callback) {
    callback(null, true); // Desenvolvimento
  },
  credentials: true,
  methods: ['GET', 'POST', 'PUT', 'DELETE', 'OPTIONS'],
  allowedHeaders: ['Content-Type', 'Authorization', ...]
}));
```

10.3 Dados Sensíveis

- Nenhum dado pessoal armazenado
- Logs sem informações sensíveis
- Validação de entrada em todas APIs
- Error handling sem exposição de stack traces

---

11. Troubleshooting

11.1 Problemas Comuns

Avisos não salvam

Sintomas: Avisos criados não aparecem após refresh **Diagnóstico:**

```bash
# Verificar backend
curl http://localhost:3001/api/status

# Verificar endpoint avisos
curl http://localhost:3001/api/notices
```

Soluções:

1. Verificar se backend está rodando (porta 3001)
2. Conferir variáveis de ambiente no `.env`
3. Verificar logs no console do navegador (F12)
4. Testar conectividade: Admin → Debug → Testar Conexão

PDFs não carregam

Sintomas: Documentos não aparecem ou erro de conversão **Diagnóstico:**

```bash
# Verificar arquivos
ls backend/public/uploads/

# Verificar permissões
ls -la backend/public/uploads/

# Testar URL direta
curl http://localhost:3001/uploads/arquivo.pdf
```

Soluções:

1. Verificar CORS no servidor
2. Confirmar arquivos salvos em `/public/uploads/`
3. Testar conversão manual
4. Limpar cache: Admin → Configurações → Limpar Cache

Performance lenta

Sintomas: Interface travando ou lenta **Diagnóstico:**

```bash
# Verificar uso de memória
curl http://localhost:3001/api/system-info
```

Soluções:

1. Limpar cache de páginas PLASA
2. Verificar tamanho dos PDFs (máximo 50MB)
3. Reduzir velocidade de scroll
4. Reiniciar servidor

11.2 Logs e Debug

Console Frontend (F12)

```javascript
// Logs importantes para debug
console.log("📢 Estado atual dos avisos:", notices);
console.log("📄 Documentos PLASA:", plasaDocuments);
console.log("🌐 Backend URL:", getBackendUrl('/'));
```

Logs Backend

```bash
# Logs do servidor mostram:
📢 Criando aviso no servidor: Título
📄 Upload de documento: arquivo.pdf
🔍 Verificando 8 páginas PLASA...
✅ Sistema funcionando normalmente
```

11.3 Comandos de Manutenção

Limpeza Completa

bash

```bash
# Backend - limpar todos os arquivos
rm -rf backend/public/uploads/*
rm -rf backend/public/plasa-pages/*
rm -f backend/data/notices.json

# Frontend - limpar cache
localStorage.clear()
sessionStorage.clear()
```

Reset do Sistema

bash

```bash
# Parar serviços
pkill -f "node.*server.js"
pkill -f "vite"

# Reinstalar dependências
cd backend && npm install
cd frontend && npm install

# Reiniciar
cd backend && npm start
cd frontend && npm run dev
```

---

12. Deploy e Produção

12.1 Build de Produção

Frontend

bash

```bash
cd frontend
npm run build
# Gera pasta /dist para servir estaticamente
```

Backend

bash

```bash
cd backend
NODE_ENV=production npm start
```

12.2 Configurações de Produção

Environment Variables

```bash
# Backend
PORT=3001
NODE_ENV=production

# Frontend build
VITE_BACKEND_HOST=seu-servidor.com
VITE_BACKEND_PORT=3001
```

CORS Produção

```javascript
// Restringir origins em produção
app.use(cors({
  origin: ['https://seu-dominio.com'],
  credentials: true
}));
```

12.3 Docker (Opcional)

Dockerfile Backend

```dockerfile
FROM node:18-alpine
WORKDIR /app
COPY package*.json ./
RUN npm ci --only=production
COPY . .
EXPOSE 3001
CMD ["npm", "start"]
```

Docker Compose

yaml

```yaml
version: '3.8'
services:
  backend:
    build: ./backend
    ports:
      - "3001:3001"
    volumes:
      - ./backend/data:/app/data
      - ./backend/public:/app/public

  frontend:
    build: ./frontend
    ports:
      - "80:80"
    depends_on:
      - backend
```

---

13. Manutenção

13.1 Backup

bash

```bash
# Backup essencial
tar -czf backup-$(date +%Y%m%d).tar.gz \
  backend/data/ \
  backend/public/uploads/ \
  frontend/.env
```

13.2 Monitoramento

bash

```bash
# Verificar saúde do sistema
curl http://localhost:3001/api/status
curl http://localhost:3001/api/system-info

# Verificar logs
tail -f backend/logs/app.log
```

13.3 Atualizações

bash

```bash
# Atualizar dependências
cd backend && npm update
cd frontend && npm update

# Testar após atualização
npm run test
```

---

14. Glossário

- **PLASA**: Plano de Serviço Semanal da Marinha
- **PAPEM**: Pagadoria de Pessoal da Marinha
- **Escala**: Documento com escalação de serviços
- **PDF.js**: Biblioteca JavaScript para renderização de PDFs
- **Canvas**: Elemento HTML5 para desenho e renderização
- **CORS**: Cross-Origin Resource Sharing
- **Multer**: Middleware Node.js para upload de arquivos
- **shadcn/ui**: Biblioteca de componentes React
- **Context API**: Sistema de estado global do React

Versão: 1.0
Data: Junho 2025
Autor: Sistema Marinha Digital Display
Contato: Setor de TI - PAPEM