SGMP

Alunos: Otávio Saturnino,

Murilo Coletta,

Daniel Charlo

Sumário

DOCUMENTAÇÃO TÉCNICA – PredialFix API (PHP Puro)

Projeto: PredialFix – Plataforma de Gestão de Manutenção Predial
Contexto: Sistema interno para controle de chamados de manutenção no SENAI
Tecnologia: PHP Puro
Arquitetura: API RESTful

1. Introdução

Este documento descreve de forma detalhada a documentação técnica da API PredialFix. O foco está exclusivamente na lógica de funcionamento, arquitetura, fluxo de dados, segurança e organização do back-end, sem abordar a implementação de banco de dados, que será definida em etapa posterior do projeto.

2. Objetivo da API

A API PredialFix tem como objetivo centralizar e organizar o fluxo de chamados de manutenção predial, garantindo rastreabilidade, transparência e controle total sobre cada solicitação, desde sua abertura até a conclusão.

3. Escopo Técnico

- Receber e processar requisições HTTP
- Validar dados de entrada
- Controlar autenticação e autorização
- Gerenciar fluxo de status dos chamados
- Retornar respostas padronizadas em JSON
- Garantir segurança e integridade das operações

4. Arquitetura da Aplicação

A aplicação é estruturada em camadas bem definidas, mesmo utilizando PHP puro, seguindo boas práticas de separação de responsabilidades:

• Entry Point: arquivo index.php responsável por receber todas as requisições.
• Router: responsável por identificar a rota e método HTTP.
• Controllers: camada responsável por coordenar ações.
• Services: implementação das regras de negócio.
• Middlewares: validações globais como autenticação.
• Utils: funções auxiliares reutilizáveis.

5. Comunicação RESTful

A API segue os princípios REST:
- Uso correto dos métodos HTTP (GET, POST, PUT)
- Recursos identificados por URLs
- Comunicação stateless

- Respostas sempre em formato JSON

6. Autenticação e Autorização

A autenticação é baseada em token de acesso enviado no header Authorization (Bearer Token). Cada requisição protegida passa obrigatoriamente por um middleware de autenticação.

A autorização é feita com base no nível do usuário:
- Usuário comum: abertura e visualização de chamados
- Responsável técnico: atualização de status e acompanhamento

7. Validação de Dados

Toda entrada de dados é validada manualmente antes do processamento, garantindo:
- Campos obrigatórios
- Tipos de dados corretos
- Limites de tamanho
- Sanitização para evitar dados maliciosos

Requisições inválidas retornam erro HTTP 400.

8. Workflow de Chamados

O fluxo de atendimento segue obrigatoriamente a sequência:

Aberto → Em Análise → Em Execução → Concluído

Não é permitido pular etapas. Tentativas de transição inválida retornam erro.

9. Notificações de Progresso

As notificações são simuladas e retornadas como mensagens informativas nas respostas da API, refletindo o estágio atual do chamado, como:
- Chamado em análise
- Técnico a caminho
- Serviço finalizado

10. Endpoints da API

AUTENTICAÇÃO:
POST /api/login
POST /api/logout

CHAMADOS:
POST /api/chamados
GET /api/chamados
GET /api/chamados/{id}

PUT /api/chamados/{id}/status
GET /api/unidade/{local}/historico

Todas as respostas seguem o padrão:

```
{
  "success": true,
  "message": "Descrição da operação",
  "data": {}
}
```

11. Tratamento de Erros

A API utiliza códigos HTTP apropriados:
- 200: Sucesso
- 201: Recurso criado
- 400: Requisição inválida
- 401: Não autenticado
- 403: Não autorizado
- 404: Recurso não encontrado
- 500: Erro interno

12. Estrutura de Pastas

```
/api
 ├── index.php
 ├── router.php
 ├── controllers/
 │    ├── AuthController.php
 │    ├── ChamadoController.php
 ├── services/
 │    ├── ChamadoService.php
 ├── middlewares/
 │    ├── AuthMiddleware.php
 ├── utils/
 │    ├── Response.php
 │    ├── Validator.php
```

13. Boas Práticas Adotadas

- Código organizado e modular
- Padronização de respostas
- Separação clara de responsabilidades
- Facilidade de manutenção e expansão
- Preparado para integração futura com banco de dados

14. Versionamento e Entrega

O projeto deve ser versionado em repositório GitHub.
Commits descritivos e organização clara de pastas.
README contendo instruções de uso da API.