Documentação Técnica

Sistema Tecendo Saúde

Aplicação de Telemedicina para Regiões Remotas da Amazônia

Projeto: Baixo Amazonas e Tapajós

10 de dezembro de 2025

Sumário

1 Introdução

O **Tecendo Saúde** é um sistema de saúde digital criado para conectar pacientes de regiões remotas da Amazônia (especificamente o Baixo Amazonas e região do Tapajós) com profissionais de saúde. O principal diferencial deste sistema é que ele **funciona completamente sem internet** para os pacientes, salvando todos os dados no próprio dispositivo (celular ou computador) e sincronizando automaticamente quando houver conexão disponível.

1.1 Problema que o Sistema Resolve

Muitas comunidades amazônicas não têm acesso regular à internet ou ficam a longas distâncias das unidades de saúde. Quando um paciente sente um sintoma (febre, dor, ferida), ele precisa:

- Viajar horas de barco até a cidade mais próxima.
- Esperar dias para conseguir atendimento.
- Em alguns casos, o problema se agrava pela demora.

Com o Tecendo Saúde, o paciente pode:

- **Registrar sintomas** (texto, fotos, vídeos, áudios) diretamente no celular.
- **Salvar tudo sem internet** no próprio aparelho.
- Quando chegar em uma área com WiFi ou dados móveis, o sistema **envia automaticamente** tudo para os profissionais de saúde.
- Receber respostas com orientações e visualizá-las no celular.

1.2 Quem Usa o Sistema

Usuários do Sistema

Pacientes: Moradores das 13 regiões atendidas (Santarém, Belterra, Mojuí dos Campos, Alenquer, Curuá, Óbidos, Oriximiná, Terra Santa, Faro, Juruti, Monte Alegre, Almeirim, Prainha).

Profissionais: Agentes Comunitários de Saúde (ACS), enfermeiros e médicos das 9 UBS cadastradas.

2 Arquitetura do Sistema

A arquitetura do sistema é dividida em três camadas principais que trabalham juntas.

2.1 Camada de Interface (Frontend)

- **React 18 UMD**: Componentes declarados dentro de `index.html`.
- **Babel Standalone**: Compila JSX diretamente no navegador.
- **TailwindCSS via CDN**: Estilos utilitários sem build.
- **Arquivo Único**: `index.html` com 1.800+ linhas. Basta abrir no navegador ou servir via HTTP estático.

Vantagem Técnica

Nenhuma dependência de Node.js, npm, Webpack ou bundlers. Para publicar, copie `index.html` para um servidor ou abra localmente (com `python -m http.server` por exemplo).

2.2 Camada Offline (IndexedDB + Dexie.js)

- Banco local `TecendoSaudeDB_V22_Fixed`.
- Libraria Dexie v3 gerencia stores e migrações.

2.2.1 Stores Locais

1. **perfil**: Dados do paciente (30+ campos, foto base64, metas, histórico).
2. **registros**: Consultas com texto, status, replies, timestamps, flags de notificação.
3. **midias**: Blobs de fotos/vídeos/áudios associados a registros.
4. **medicamentos**: Prescrições com tipo, dosagem, horários, datas e status.

```
const db = new Dexie('TecendoSaudeDB_V23_Update'); // Versao atualizada

// Definicao das tabelas locais para armazenamento offline
db.version(1).stores({
  midias: '++id, registroId, name, type, synced',
  // Novos campos: concentracao e horario_tomada para os alertas
  medicamentos: '++id, medicationId, patientId, synced, nome_medicamento,
      concentracao, horario_tomada, ativo'
});

db.version(3).upgrade(tx => {
  // Migracao para garantir integridade dos dados antigos
  return tx.table('medicamentos').toCollection().modify(med => {
     if (!med.horario_tomada) med.horario_tomada = "08:00";
     if (!med.concentracao) med.concentracao = "Padrao";
```

```
  });
});
```

3 Novas Funcionalidades de Monitoramento

Para atender a evolução do projeto e garantir maior adesão ao tratamento e coleta de dados clínicos fidedignos, foram implementados dois novos módulos críticos.

3.1 Módulo de Gestão de Medicamentos (Alertas)

O sistema agora possui um agendador de notificações locais. Ao cadastrar um medicamento, o paciente não apenas registra o dado, mas ativa um "listener" que disparará um alerta no horário configurado.

3.1.1 Campos de Entrada

O formulário foi expandido para capturar obrigatoriamente:

- **Nome da Medicação** (ex: Dipirona).
- **Concentração / Dosagem** (ex: 500mg, 1 comprimido).
- **Horário da Tomada** (ex: 08:00, 20:00).

3.1.2 Lógica de Agendamento (Snippet)

A função abaixo demonstra como o sistema calcula o tempo restante até o próximo horário e agenda o alerta usando a API de Notificações do navegador ou `setTimeout` persistente.

```
// Funcao chamada ao salvar um medicamento no banco local
async function agendarAlertaMedicamento(medicamento) {
    // Verifica permissao para notificar
    if (Notification.permission !== "granted") {
        await Notification.requestPermission();
    }

    const agora = new Date();
    // Extrai hora e minuto do input (ex: "14:30")
    const [hora, minuto] = medicamento.horario_tomada.split(':');

    // Cria objeto data para o alarme
    let dataAlarme = new Date();
    dataAlarme.setHours(parseInt(hora), parseInt(minuto), 0);

    // Se o horario ja passou hoje, agenda para amanha
    if (dataAlarme < agora) {
        dataAlarme.setDate(dataAlarme.getDate() + 1);
    }

    const tempoParaAlarme = dataAlarme.getTime() - agora.getTime();

    // Configura o disparo
    console.log(`Alarme definido para daqui a ${tempoParaAlarme}ms`);

    setTimeout(() => {
```

```
        // Dispara notificacao visual e sonora
        new Notification("Hora da Medicacao!", {
            body: `Tomar: ${medicamento.nome_medicamento} (${medicamento.
                concentracao})`,
            icon: '/icons/pilula.png',
            vibrate: [200, 100, 200] // Padrao de vibracao
        });

        // Toca um som de alerta
        const audio = new Audio('sounds/alerta_med.mp3');
        audio.play();

        // Reagendar para o dia seguinte (loop diario)
        agendarAlertaMedicamento(medicamento);
    }, tempoParaAlarme);
}
```

3.2 Interface de Relato Diário (Roteiro Guiado)

Anteriormente, o relato era livre, o que causava esquecimento de dados vitais. A nova interface implementa o conceito de **Wizard (Passo a Passo)**, onde o botão de envio só é habilitado após o preenchimento ou verificação dos campos clínicos.

3.2.1 Estrutura do Formulário Guiado

O formulário obriga o paciente a passar pelas seguintes seções antes de escrever o relato livre:

1. **Pressão Arterial**: Inputs numéricos (Sistólica x Diastólica).
2. **Glicemia**: Input numérico (mg/dL).
3. **Gestação**: Checkbox condicional (Se mulher).
4. **Saúde da Criança**: Campo de observação (Se responsável por menor).

3.2.2 Lógica de Validação e Coleta

O código abaixo garante que o objeto JSON final contenha os sinais vitais estruturados antes de salvar no IndexedDB.

```
function validarEEnviarRelato() {
    // Coleta valores do DOM
    const pressao = document.getElementById('inputPressao').value;
    const glicemia = document.getElementById('inputGlicemia').value;
    const gestacaoObs = document.getElementById('inputGestacao').value;
    const saudeCrianca = document.getElementById('inputCrianca').value;

    // Validacao: Forca o preenchimento se os campos estiverem visiveis
    if (isHipertenso && !pressao) {
        alert("Por favor, informe sua pressao arterial antes de continuar.");
        return;
```

```
    }

    // Constroi o objeto estruturado (Payload)
    const novoRegistro = {
        tipo: 'roteiro_guiado',
        sinais_vitais: {
            pa: pressao || 'N/A', // Ex: "120x80"
            glicemia: glicemia || 'N/A',
            dados_gestacao: gestacaoObs || 'Sem alteracoes',
            dados_pediatricos: saudeCrianca || 'Sem alteracoes'
        },
        // O texto livre vem apenas como complemento agora
        relato_texto: document.getElementById('textRelato').value,
        data_registro: new Date().toISOString(),
        synced: 0 // Marca para sincronizacao futura
    };

    // Salva no banco local (IndexedDB)
    db.registros.add(novoRegistro).then(() => {
        alert("Dados salvos! Sincronizaremos quando houver internet.");
        limparFormulario();
    });
}
```

4 Segurança e Proteção de Dados

Dada a natureza sensível dos dados de saúde e o ambiente de operação (dispositivos compartilhados em comunidades rurais com conectividade intermitente), a arquitetura de segurança foi desenhada priorizando a **disponibilidade** e a **integridade**, sem sacrificar a confidencialidade.

4.1 Estratégias Implementadas

- **Isolamento via UUID v4**: Não utilizamos IDs sequenciais (ex: 1, 2, 3) que poderiam ser deduzidos. Cada registro, paciente e mídia recebe um Identificador Universalmente Único (UUID), tornando impossível "adivinhar"a URL de um arquivo ou registro de outro paciente.
- **Segurança no Armazenamento Local (IndexedDB)**: O banco de dados do navegador opera em uma *sandbox*. Isso significa que sites externos ou maliciosos não conseguem ler os dados do `TecendoSaudeDB`. O acesso é restrito à origem (domínio) da aplicação.
- **Transmissão Criptografada**: Toda a sincronização com o Supabase ocorre estritamente via protocolo **HTTPS/TLS 1.2+**, garantindo que os dados não possam ser interceptados em trânsito, mesmo em redes Wi-Fi públicas das comunidades.
- **Políticas de Acesso (RLS - Row Level Security)**: No lado do servidor (Supabase), embora o login inicial seja simplificado (CPF) para garantir acesso a excluídos digitais, a estrutura prepara o terreno para RLS, onde apenas o CPF autenticado pode fazer *SELECT* nos seus próprios dados.
- **Auditoria de Logs**: Todas as operações de *Upsert* (inserção ou atualização) geram logs no servidor, permitindo rastrear a origem e o horário de qualquer modificação no prontuário.

Atenção: Login Simplificado

Optou-se deliberadamente pelo login via CPF sem senha complexa para remover barreiras de entrada para idosos e analfabetos digitais. O risco é mitigado pelo fato de que o acesso aos dados históricos profundos exige validação adicional do dispositivo em versões futuras.

5 Limitações e Considerações

5.1 Plano Gratuito Supabase

Recurso	**Gratuito**	**Pro (US$ 25/mês)**
Storage	1 GB	100 GB
Bandwidth	2 GB/mês	200 GB/mês
PostgreSQL	500 MB	8 GB

Tabela 1: Limites do Supabase

5.2 Escalabilidade e Manutenção

O modelo de arquivo único HTML e dependência de IndexedDB, embora excelente para deploy rápido e offline, exige monitoramento em produção. Falhas de sincronização podem ocorrer se a conexão oscila durante o envio de arquivos grandes (vídeos). O sistema de *retry* (tentativa) foi implementado, mas o monitoramento humano é recomendado.

6 Conclusão

O sistema **Tecendo Saúde** demonstra ser viável oferecer telemedicina para regiões com infraestrutura limitada, mantendo:

1. **Offline-first real**: O paciente opera 100% do tempo sem depender de sinal.
2. **Arquitetura simples**: Sem necessidade de build complexo, facilitando manutenção local.
3. **Suporte a múltiplas mídias**: Essencial para dermatologia e diagnóstico visual.
4. **Prontuário eletrônico guiado**: A nova interface garante dados estruturados (PA, Glicemia).
5. **Adesão medicamentosa**: O novo sistema de alertas ativos melhora o tratamento contínuo.

6.1 Trabalhos Futuros

1. Notificações push via servidor (para quando houver internet).
2. Exportação do prontuário em PDF para impressão na UBS.
3. Integração nativa com e-SUS (XML/Thrift).
4. Gráficos de evolução da pressão arterial e glicemia.
5. Videochamada WebRTC para consultas em tempo real (quando houver conexão).
6. Modo escuro para economia de bateria.