

Stack para Healthtech Solo

Stack tecnico completo para MVP de healthtech
construido por uma pessoa so.

FRONTEND

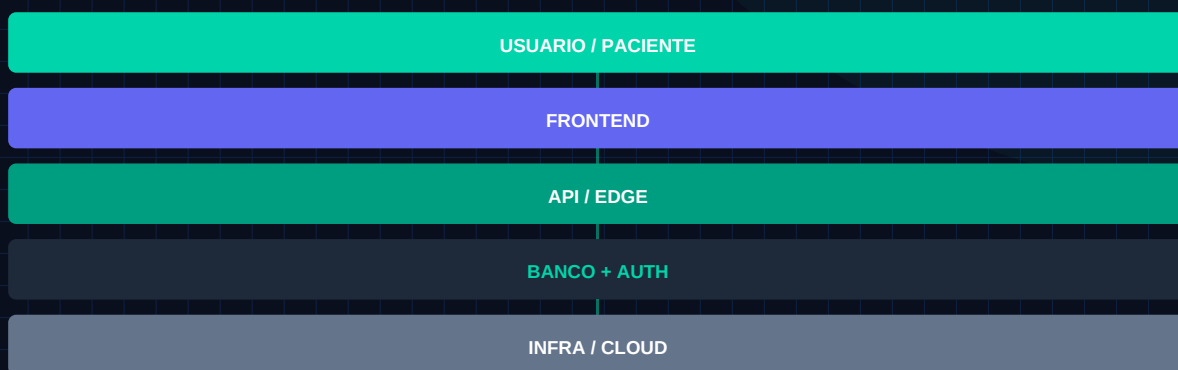
Next.js + Tailwind

BACKEND

Supabase + Edge Fn

INFRA

Vercel + Railway



Arquitetura em camadas — MVP solo

\$0–40/mes

custo inicial de infra

4–8 semanas

para MVP funcional

1 dev

equipe necessaria

LGPD ready

com as escolhas certas

Por que healthtech solo e diferente?

Construir um MVP de healthtech sozinho nao e so uma questao tecnica — e uma questao de sobrevivencia. Voce tem que ser rapido o suficiente para validar antes do dinheiro acabar, seguro o suficiente para nao ter problema com LGPD/CFM, e simples o suficiente para manter sozinho. A stack errada te mata nos tres frentes. Este guia foi montado para o cenario real: 1 desenvolvedor, orcamento proximo de zero, produto em saude digital, prazo de 4 a 8 semanas para MVP funcional com usuarios reais.

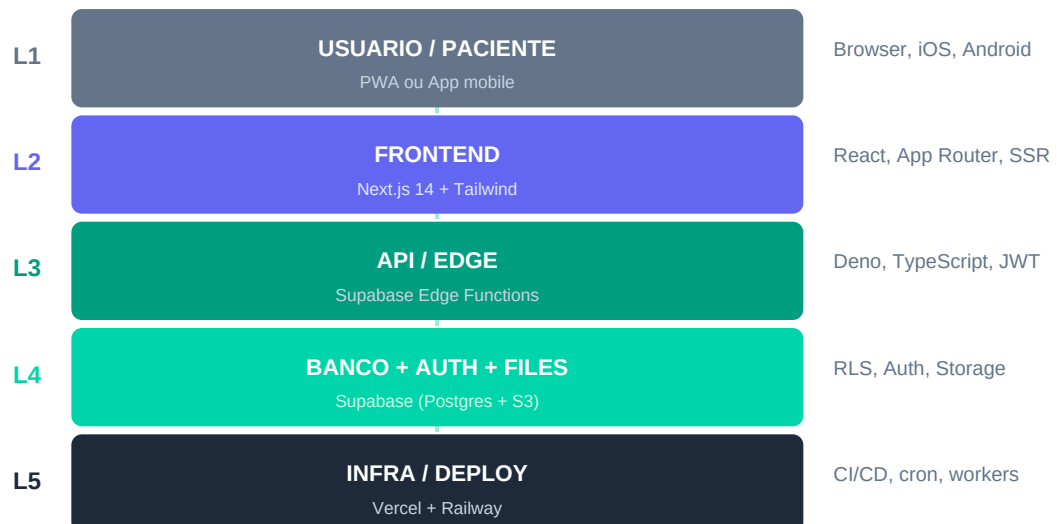
Zero a baixo custo inicial	Planos gratuitos agressivos ate Product-Market Fit
LGPD como arquitetura	Nao como afterthought — construir certo desde o inicio
Um dev consegue manter	Sem ops complexa, sem clusters, sem infra manual
Troca de pecas possivel	Cada camada pode ser substituida sem reescrever tudo

01

Arquitetura recomendada

Cinco camadas. Cada uma com uma escolha principal e o motivo.

A arquitetura abaixo prioriza velocidade de desenvolvimento, custo zero inicial e compliance com dados sensiveis de saude. Nao e a mais elegante — e a mais pratica para quem esta sozinho com um prazo real.



POR QUE SUPABASE NO CENTRO:

Supabase resolve em uma ferramenta: banco Postgres com RLS (Row Level Security — essencial para LGPD), autenticao com MFA, storage de arquivos (exames, laudos), realtime para notificacoes e Edge Functions para logica de backend. Para um dev solo, e o maior multiplicador de velocidade disponivel em saude digital.

Decisao por necessidade — tabela rapida

NECESSIDADE	ESCOLHA RECOMENDADA	ALTERNATIVA	EVITAR
-------------	---------------------	-------------	--------

Banco de dados	Supabase (Postgres)	PlanetScale, Neon	MongoDB (sem RLS nativo)
Autenticacao	Supabase Auth	Clerk, Auth.js	Implementar do zero
Frontend	Next.js 14	Remix, Astro	CRA ou Vue (ecossistema menor)
Estilo / UI	Tailwind + shadcn/ui	Chakra, Mantine	Material UI (pesado, opinionado)
Deploy frontend	Vercel	Netlify, Cloudflare Pages	EC2 manual
Workers / cron	Railway + cron-job.org	Render, Fly.io	Lambda (complexo solo)
Emails	Resend	SendGrid, Postmark	SMTP proprio
Monitoramento	Sentry (free tier)	Datadog (caro), LogRocket	Sem monitoramento
Analytics	PostHog (self-host)	Mixpanel, Amplitude	Google Analytics em saude
Pagamentos	Stripe	Pagar.me, Iugu	Boleto manual
PDF / Laudos	React-pdf + S3	Puppeteer, PDFKit	Word/Excel como solucao
Integracao TISS	lib-tiss (npm)	Implementar parser proprio	Ignorar ate depois

02

Frontend — Next.js + Tailwind + shadcn

O trio que permite velocidade sem sacrificar qualidade de produto.

Next.js 14

Framework

Gratis / open source

POR QUE:

App Router, SSR nativo, API routes, middleware — tudo em um. Server Components reduzem bundle e melhoram performance (caso comum em saúde pública).

TRADEOFF:

Curva de aprendizado do App Router. RSC pode confundir no início.

shadcn/ui

Componentes

Gratis / open source

POR QUE:

Componentes copiados para o repo (não é dependência externa). Acessibilidade out-of-the-box via Radix UI. Economiza semanas básicas.

TRADEOFF:

Não é uma biblioteca convencional — exige entender o pattern de copy-paste de

Tailwind CSS

Estilo

Gratis / open source

POR QUE:

Utility-first: escreve menos CSS, mantém menos CSS. Design system implícito de configuração. Funciona muito bem com shadcn/ui — zero conflito de escopo.

TRADEOFF:

HTML pode ficar verboso. Requer disciplina com componentes para não virar sopa de letras de classes.

React Hook Form +

Gratis / open source

POR QUE:

Formulários em saúde são críticos: dados de paciente, anamnese, prescrição. re-render + Zod validation tipada = formulários performáticos e seguros sem esforço.

TRADEOFF:

Verboso para formulários muito simples. Pode ser overkill para forms de 2 campos.

TanStack Query

Data

Gratis / open source

POR QUE:

Cache, refetch, invalidation e loading states resolvidos. Essencial para múltiplas fontes de dados (prontuário, agenda, faturamento) que atualizam.

TRADEOFF:

Adiciona complexidade em apps simples. Para apenas 1-2 endpoints.

Recharts

Gráficos

Gratis / open source

POR QUE:

Gráficos de evolução de paciente, ocupação, indicadores clínicos. Recharts é simples de integrar com React e tem os tipos de gráfico necessários para healthtech básico.

TRADEOFF:

Não é o mais customizável. Para dashboards complexos, considere D3 direto.

Setup inicial — comandos

Terminal — criando o projeto

```
# Criar projeto Next.js
npx create-next-app@latest meu-healthtech \
  --typescript --tailwind --eslint --app --src-dir

# Instalar shadcn/ui
npx shadcn-ui@latest init

# Dependências essenciais
npm install @supabase/supabase-js @supabase/ssr \
  react-hook-form zod @hookform/resolvers \
  @tanstack/react-query recharts

# Supabase CLI (para migrations locais)
npm install -D supabase
```

03

Backend — Supabase como hub central

Banco + Auth + Storage + Realtime + Edge Functions. Um servico, cinco problemas resolvidos.

Row Level Security (RLS) — o core de LGPD no banco

RLS é o mecanismo mais importante para compliance em healthtech no Supabase. Ele garante, no nível do banco de dados, que cada usuário só acessa os dados que tem permissão — mesmo que o código da aplicação tenha um bug.

SQL — Políticas de RLS para tabela de pacientes

```
-- Habilitar RLS na tabela
ALTER TABLE pacientes ENABLE ROW LEVEL SECURITY;

-- Medico ve apenas seus proprios pacientes
CREATE POLICY 'medico_ve_seus_pacientes'
  ON pacientes FOR SELECT
  USING (medico_id = auth.uid());

-- Paciente ve apenas seus proprios dados
CREATE POLICY 'paciente_ve_seus_dados'
  ON pacientes FOR SELECT
  USING (user_id = auth.uid());

-- Admin ve todos (apenas role admin)
CREATE POLICY 'admin_ve_todos'
  ON pacientes FOR ALL
  USING (auth.jwt() ->> 'role' = 'admin');
```

Schema basico para healthtech

SQL — Tabelas fundamentais

```
-- Perfis de usuario (estende auth.users)
CREATE TABLE profiles (
  id UUID REFERENCES auth.users PRIMARY KEY,
  role TEXT CHECK (role IN ('paciente', 'medico', 'admin')),
  crm TEXT, -- apenas para medicos
  cpf TEXT, -- armazenar hash, nao plaintext
  created_at TIMESTAMPTZ DEFAULT NOW()
);

-- Consultas / agendamentos
CREATE TABLE consultas (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  paciente_id UUID REFERENCES profiles(id),
  medico_id UUID REFERENCES profiles(id),
  status TEXT CHECK (status IN ('agendada', 'realizada', 'cancelada')),
  data_hora TIMESTAMPTZ NOT NULL,
  notas TEXT -- campo clinico - acesso restrito
);
```

Supabase Auth

Autenticação **Grátis até 50k MAU**

POR QUE:

Email/senha, magic link, OAuth (Google, Apple), MFA por TOTP, claims de role — essencial para RLS. No healthtech: MFA dev e clínicos.

TRADEOFF:

Lock-in no ecossistema Supabase. Migração futura exige trabalho.

Supabase Storage

Grátis 1GB / depois \$0.021/GB

POR QUE:

Armazenamento de exames, laudos, imagens clínicas com políticas de acesso RLS do banco. Transformação de imagem nativa. URLs assinadas com expiração, arquivos clínicos com URLs públicas permanentes.

TRADEOFF:

Sem virus scan nativo. Implemente validação de tipo MIME no upload.

Supabase Realtime

Gratis ate 200 conexoes

POR QUE:

Notificacoes de consulta, atualizacao de status de exame, chat. Postgres Changes — qualquer UPDATE/INSERT pode disparar polling.

TRADEOFF:

Nao e WebSocket full-duplex. Para chat complexo, avalie Ably ou Pusher.

Edge Functions

Gratis 500k invocacoes/mes

POR QUE:

Deno + TypeScript rodando na edge. Use para: webhooks de pagamento, v geracao de PDF de laudo, integracao com CFM, envio de emails via Resend. logica critica no frontend.

TRADEOFF:

Cold start de ~400ms. Nao use para operacoes que exigem latencia < 100ms.

04

LGPD e compliance em saude

Dado de saude e dado sensivel. A multa pode chegar a R\$50M. Nao e opcional.

ATENCAO CRITICA:

Dados de saude sao classificados como dados sensiveis pela LGPD (Art. 11). Isso significa base legal mais restrita (consentimento expreso ou protecao da vida), DPO obrigatorio dependendo do volume, e obrigacao de notificar a ANPD em caso de vazamento em ate 72 horas. Implemente desde o primeiro commit — refatorar compliance depois custa 10x mais.

Checklist de compliance por camada

REQUISITO	O QUE FAZER	FERRAMENTA	STATUS
Consentimento LGPD	Termo de uso + politica de privacidade explicitos antes do cadastro. Guardar log do aceite com timestamp e versao.	<i>Supabase DB + timestamp</i>	✓ OK
Criptografia em repouso	Supabase criptografa disco por padrao. Para campos extra-sensiveis (CPF, CRM), usar pgcrypto.	<i>pgcrypto extension</i>	✓ OK
Criptografia em transito	HTTPS obrigatorio em todo o stack. Nunca HTTP, nem em dev — use localhost com SSL.	<i>Vercel SSL / Supabase SSL</i>	✓ OK
RLS no banco	Toda tabela com dado de paciente deve ter RLS ativo. Testar com usuario sem privilegio antes de ir a prod.	<i>Supabase RLS policies</i>	✓ OK
Logs de acesso	Registrar quem acessou qual dado de paciente e quando. Exigido pelo CFM para prontuario eletronico.	<i>Supabase audit_log table</i>	■ Atenção
Direito ao esquecimento	Implementar endpoint de exclusao completa de dados do usuario (Art. 18 LGPD). Incluir dados em backups.	<i>API route + script DB</i>	■ Atenção
Anonimizacao de dados	Dados usados em analytics devem ser anonimizados ou pseudonimizados antes de sair do banco.	<i>View anonimizada + PostHog</i>	■ Atenção
Backup criptografado	Supabase faz backup automatico. Para exportacoes manuais, criptografar antes de sair do ambiente.	<i>Supabase backups + gpg</i>	✓ OK
DPO nomeado	Se processar dados de muitos pacientes, nomear Data Protection Officer e comunicar a ANPD.	<i>Documento + ANPD portal</i>	■ Atenção
TISS / CFM	Para emissao de guias e prontuario eletronico, seguir padroes TISS 3.x e resolucao CFM 1821/2007.	<i>lib-tiss + validacao</i>	■ Atenção

Variaveis de ambiente — o que nunca vai pro repositorio

.env.local — nunca commitar este arquivo

env

```
# Supabase
NEXT_PUBLIC_SUPABASE_URL=https://xxx.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=eyJ...      # publica, mas nao hardcode
SUPABASE_SERVICE_ROLE_KEY=eyJ...          # NUNCA expor no frontend

# Pagamentos
STRIPE_SECRET_KEY=sk_live...               # apenas no servidor
STRIPE_WEBHOOK_SECRET=whsec...

# Email
RESEND_API_KEY=re...

# Monitoramento
SENTRY_DSN=https://...@sentry.io/...

# Criptografia de campos sensiveis
ENCRYPTION_KEY=...                         # 256-bit, gerado com openssl rand
```

05

Infra, deploy e custo

Tudo automatizado. Zero servidor pra configurar. Custo proximo de zero ate PMF.

Vercel

Gratis (hobby) / \$20/mes (pro)

POR QUE:

Deploy automatico a cada push no GitHub. Preview URLs por jobs via Vercel Cron. Para healthtech: plano Pro tem SLA de 9

TRADEOFF:

Plano hobby nao tem SLA. Para producao real, Pro e necessario (

Sentry

Monitoramen
Gratis ate 5k erros/mes

POR QUE:

Captura de erros em frontend e backend com stack trace completo. IMPOR data scrubbing para nao enviar dados de paciente nos reports. Sentry nao e por padrao — configure PII scrubbing antes de ativar.

TRADEOFF:

Dados de sessao podem conter PII. Exige configuracao cuidadosa em healthtech

GitHub Actions

CI/CD
Gratis 2000 min/mes

POR QUE:

Pipeline de teste + lint + deploy automatizado. Para healthtech secrets (git-secrets), teste de RLS policies e audit de depende pipeline.

TRADEOFF:

Minutos esgotam em repos privados com muitos PRs. Otimize con

Resend

Email transacion
Gratis 3k emails/mes

POR QUE:

Emails de confirmacao de consulta, reset de senha, alertas cli Email para templates. Para documentos clinicos por email: ser

TRADEOFF:

Sem email marketing — apenas transacional. Para campanhas, us

Custo mensal estimado por fase

COMPONENTE	CUSTO / MES	PLANO
Supabase	\$0	Free tier
Vercel	\$0-20	Hobby/Pro
Railway	\$0-15	Pay-as-go
Sentry	\$0	Free tier
Resend	\$0	Free tier
dominio + SSL	\$1/mes	Cloudflare
PostHog	\$0	Cloud free
Total MVP	\$1-36	Pre-PMF

06

Roadmap de 8 semanas para MVP

Sprint a sprint — do zero ao primeiro usuario pagante.

S1–S2 Fundacao e Auth

- | | |
|---|---|
| ■ Setup do projeto Next.js + Tailwind + shadcn | ■ Supabase: projeto, schema inicial, RLS basico |
| ■ Autenticacao: cadastro, login, recuperacao de senha | ■ Middleware de autenticacao no Next.js |
| ■ Deploy inicial na Vercel + dominio configurado | ■ Variaveis de ambiente e secrets configurados |

S3–S4 Core do produto

- | | |
|--|--|
| ■ Fluxo principal do usuario (ex: agendamento de consulta) | ■ Dashboard do profissional de saude |
| ■ Upload de arquivos (exames) com Supabase Storage | ■ Notificacoes por email via Resend |
| ■ Validacao de formularios com RHF + Zod | ■ Testes de RLS: usuario A nao acessa dados de usuario B |

S5–S6 Integracao e pagamento

- | | |
|---|---|
| ■ Checkout com Stripe (assinatura ou consulta avulsa) | ■ Webhook de pagamento via Edge Function |
| ■ PDF de laudo/recibo com React-pdf | ■ Logs de auditoria de acesso a dados de paciente |
| ■ Sentry configurado com PII scrubbing ativo | ■ PostHog com eventos anonimizados |

S7–S8 Polimento e lancamento

- | | |
|---|---|
| ■ Testes com 5+ usuarios reais do publico-alvo | ■ Ajustes de UX baseados nos testes |
| ■ Pagina de politica de privacidade e termos de uso | ■ Revisao de seguranca: npm audit, secrets scan |
| ■ Monitoramento de uptime (Better Uptime free tier) | ■ Lancamento: Product Hunt, grupos medicos, indicacao |

07

Integracoes especificas de healthtech

O que nao tem no tutorial padrao do Next.js mas voce vai precisar.

TISS — Troca de Informacoes na Saude Suplementar

- TISS e o padrao XML obrigatorio para troca com operadoras de plano de saude.
- Biblioteca npm: lib-tiss ou implementar parser proprio para TISS 3.x.
- Para MVP: implemente apenas Guia de Consulta (tipo 1) — e o mais comum.
- Validacao de XML contra XSD antes de enviar para operadora.
- Nunca armazene RNO (registro nacional de operadora) em texto plano.

CFM — Conselho Federal de Medicina

- Prontuario eletronico: Resolucao CFM 1821/2007 exige registro de autoria e data/hora.
- Assinatura digital de documentos clinicos: considere ITI/ICP-Brasil para validade legal.
- Tempo de retencao: prontuario deve ser guardado por minimo 20 anos.
- Cada entrada no prontuario deve ser imutavel — use append-only no banco.
- Telemedicina: Resolucao CFM 2314/2022 — verificar requisitos atuais antes de implementar.

Integracao com laboratorios e HIS

- HL7 FHIR R4 e o padrao emergente para interoperabilidade — prefira sobre integracao proprietary.
- Para MVP: comece com integracao manual (upload de PDF) e automatize depois.
- APIs de laboratorios (DASA, Fleury, Hermes Pardini) tem documentacao proprietaria — solicite acesso.
- HIS (sistemas hospitalares): MV, Tasy, Philips — integracao via HL7 v2 ainda muito comum.
- RNDS (Rede Nacional de Dados em Saude) — obrigatorio para alguns tipos de servico de saude.

Teleconsulta — infraestrutura de video

- Daily.co: \$0 ate 10k minutos/mes, SDK React nativo, HIPAA-compliant (verificar adequacao LGPD).
- Whereby Embedded: alternativa com UI pronta — bom para MVP rapido.
- Jitsi self-hosted: gratis mas exige infra propria — nao recomendado para solo.
- Sala de espera virtual: use Supabase Realtime para notificar paciente quando medico esta pronto.
- Gravacao de consulta: exige consentimento expreso e politica clara de retencao.

LEMBRE-SE

A stack e o meio.

O produto e o que importa.

Lance feio. Lance rapido. Aprenda com usuarios reais.

A melhor stack e aquela que voce consegue manter sozinho enquanto valida se alguem paga pelo produto.