UNIVABIO 2026 · AI FOR HUMAN HEALTH

SOURCE CODE

VitalLens Source Code

Complete source of the VitalLens prototype — an AI health companion that turns lab reports and symptoms into plain-language guidance.

21	2,446	5	4
SOURCE FILES	LINES OF CODE	API ROUTES	APP MODULES

Stack: Next.js 16 (App Router) · TypeScript · Tailwind CSS 4 · shadcn/ui · Prisma + SQLite · Recharts · Framer Motion

AI: GLM-4V vision model for lab report extraction · GLM clinical LLM for briefings, triage & visit prep

Architecture: Client components → Next.js API routes → z-ai-web-dev-sdk (server-side only) → SQLite via Prisma

Contents

All application source files, in architecture order: data layer → AI layer → UI modules → API routes. Generated code (shadcn/ui components, Prisma client) and configuration boilerplate are excluded for readability.

01	prisma/schema.prisma	Database schema — LabReport & SymptomSession models	30
02	src/lib/medical.ts	Medical core — 40+ reference ranges, critical thresholds, status & trend engine	176
03	src/lib/ai.ts	AI service layer — VLM lab extraction, LLM briefing, triage prompt, doctor prep	245
04	src/lib/store.ts	Client state — Zustand store for cross-module sync	17
05	src/lib/db.ts	Prisma client singleton	13
06	src/app/layout.tsx	Root layout & metadata	45
07	src/app/page.tsx	Home page composition	21
08	src/components/vitallens/hero.tsx	Hero section — value proposition & animated demo card	159
09	src/components/vitallens/app-shell.tsx	App shell — 4-module tab navigation	40
10	src/components/vitallens/lab-decoder.tsx	Module 1: Lab Decoder — upload, VLM pipeline UI, markers table	460
11	src/components/vitallens/timeline.tsx	Module 2: Health Timeline — trend UI & reference-band charts	247
12	src/components/vitallens/symptom-checker.tsx	Module 3: Nurse V — conversational triage chat	222
13	src/components/vitallens/doctor-prep.tsx	Module 4: Doctor Visit Prep — sheet generation, print & download	170
14	src/components/vitallens/markdown.tsx	Markdown renderer with GFM tables	41
15	src/components/vitallens/sections.tsx	How-it-works, safety principles & footer	117
16	src/app/api/lab-reports/route.ts	API: upload & analyze lab report (VLM + LLM pipeline), list, delete	103
17	src/app/api/lab-reports/trends/route.ts	API: cross-report trend computation	17
18	src/app/api/symptom-checker/route.ts	API: symptom triage conversation	34
19	src/app/api/doctor-prep/route.ts	API: visit prep sheet generation	33
20	src/app/api/demo-seed/route.ts	API: demo timeline seeder (Sarah Chen case study)	160
21	package.json	Dependencies & project manifest	96

prisma/schema.prisma

Database schema — LabReport & SymptomSession models

30 lines

```
generator client {
  provider = "prisma-client-js"
}

datasource db {
  provider = "sqlite"
  url      = env("DATABASE_URL")
}

model LabReport {
  id          String   @id @default(uuid())
  title       String
  patientName String   @default("Guest")
  reportDate  DateTime
  labName     String?
  sourceImage String?
  markers     String // JSON: Marker[]
  summary     String? // AI plain-language summary (markdown)
  redFlags    String? // JSON: RedFlag[]
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt
}

model SymptomSession {
  id         String   @id @default(uuid())
  messages   String // JSON: ChatMessage[]
  assessment String? // JSON: TriageAssessment
  createdAt  DateTime @default(now())
  updatedAt  DateTime @updatedAt
}
```

src/lib/medical.ts

Medical core — 40+ reference ranges, critical thresholds, status & trend engine

176 lines

```
// VitalLens 医学核心库：类型、参考范围、红旗逻辑

export type MarkerStatus = 'normal' | 'low' | 'high' | 'critical-low' | 'critical-high' | 'unknown';

export interface LabMarker {
  name: string;
  value: number;
  unit: string;
  refLow?: number;
  refHigh?: number;
  category?: string; // CBC, Metabolic, Lipids, Thyroid, Iron ...
  status: MarkerStatus;
  note?: string;
}

export interface RedFlag {
  marker: string;
  value: string;
  severity: 'critical' | 'warning';
  message: string;
}

export interface TrendPoint {
  date: string;
  value: number;
}

export interface MarkerTrend {
  name: string;
  unit: string;
  points: TrendPoint[];
  changePct?: number;
```

```
  direction: 'up' | 'down' | 'flat';
  status: MarkerStatus;
}

// 成人通用参考范围(实际以报告标注为准; 此表用于报告未标注时 + 趋势评估)
export const FALLBACK_RANGES: Record<string, { low: number; high: number; unit: string; category: string }> = {
  'hemoglobin': { low: 13.5, high: 17.5, unit: 'g/dL', category: 'CBC' },
  'hematocrit': { low: 41, high: 53, unit: '%', category: 'CBC' },
  'rbc': { low: 4.5, high: 5.9, unit: '10^6/uL', category: 'CBC' },
  'wbc': { low: 4.5, high: 11.0, unit: '10^3/uL', category: 'CBC' },
  'platelets': { low: 150, high: 400, unit: '10^3/uL', category: 'CBC' },
  'mcv': { low: 80, high: 100, unit: 'fL', category: 'CBC' },
  'mch': { low: 27, high: 33, unit: 'pg', category: 'CBC' },
  'mchc': { low: 32, high: 36, unit: 'g/dL', category: 'CBC' },
  'ferritin': { low: 30, high: 400, unit: 'ng/mL', category: 'Iron Studies' },
  'iron': { low: 60, high: 170, unit: 'ug/dL', category: 'Iron Studies' },
  'tibc': { low: 240, high: 450, unit: 'ug/dL', category: 'Iron Studies' },
  'transferrin_saturation': { low: 20, high: 50, unit: '%', category: 'Iron Studies' },
  'glucose': { low: 70, high: 99, unit: 'mg/dL', category: 'Metabolic' },
  'hba1c': { low: 4, high: 5.6, unit: '%', category: 'Metabolic' },
  'alt': { low: 7, high: 56, unit: 'U/L', category: 'Liver' },
  'ast': { low: 10, high: 40, unit: 'U/L', category: 'Liver' },
  'alp': { low: 44, high: 147, unit: 'U/L', category: 'Liver' },
  'bilirubin_total': { low: 0.1, high: 1.2, unit: 'mg/dL', category: 'Liver' },
  'albumin': { low: 3.5, high: 5.2, unit: 'g/dL', category: 'Liver' },
  'creatinine': { low: 0.6, high: 1.3, unit: 'mg/dL', category: 'Kidney' },
  'egfr': { low: 90, high: 120, unit: 'mL/min/1.73', category: 'Kidney' },
  'bun': { low: 7, high: 20, unit: 'mg/dL', category: 'Kidney' },
  'sodium': { low: 136, high: 145, unit: 'mmol/L', category: 'Electrolytes' },
  'potassium': { low: 3.5, high: 5.1, unit: 'mmol/L', category: 'Electrolytes' },
  'chloride': { low: 98, high: 107, unit: 'mmol/L', category: 'Electrolytes' },
  'calcium': { low: 8.5, high: 10.5, unit: 'mg/dL', category: 'Electrolytes' },
  'tsh': { low: 0.4, high: 4.0, unit: 'mIU/L', category: 'Thyroid' },
  't4_free': { low: 0.8, high: 1.8, unit: 'ng/dL', category: 'Thyroid' },
  't3_total': { low: 80, high: 200, unit: 'ng/dL', category: 'Thyroid' },
  'vitamin_d': { low: 30, high: 100, unit: 'ng/mL', category: 'Vitamins' },
  'vitamin_b12': { low: 200, high: 900, unit: 'pg/mL', category: 'Vitamins' },
  'total_cholesterol': { low: 0, high: 200, unit: 'mg/dL', category: 'Lipids' },
  'ldl': { low: 0, high: 100, unit: 'mg/dL', category: 'Lipids' },
  'hdl': { low: 40, high: 80, unit: 'mg/dL', category: 'Lipids' },
  'triglycerides': { low: 0, high: 150, unit: 'mg/dL', category: 'Lipids' },
  'crp': { low: 0, high: 1.0, unit: 'mg/dL', category: 'Inflammation' },
  'esr': { low: 0, high: 20, unit: 'mm/hr', category: 'Inflammation' },
  'psa': { low: 0, high: 4.0, unit: 'ng/mL', category: 'Other' },
  'uric_acid': { low: 3.5, high: 7.2, unit: 'mg/dL', category: 'Other' },
};

// 危急值阈值(达到即触发红色警告)
export const CRITICAL_THRESHOLDS: Record<string, { low?: number; high?: number }> = {
  'hemoglobin': { low: 7.0, high: 20 },
  'platelets': { low: 50, high: 1000 },
  'glucose': { low: 50, high: 400 },
  'potassium': { low: 2.5, high: 6.0 },
  'sodium': { low: 120, high: 160 },
  'calcium': { low: 6.5, high: 12.5 },
  'creatinine': { high: 4.0 },
  'alt': { high: 500 },
  'ast': { high: 500 },
  'bilirubin_total': { high: 5.0 },
  'wbc': { low: 1.5, high: 30 },
};

export function normalizeMarkerName(name: string): string {
  return name.toLowerCase().trim()
    .replace(/[^a-z0-9\s]/g, ' ')
    .replace(/\s+/g, '_')
    .replace(/haemoglobin|hgb|hb$/, 'hemoglobin')
    .replace(/haematocrit|hct$/, 'hematocrit')
    .replace(/platelet_count|plt$/, 'platelets')
    .replace(/white_blood_cell|wbc$/, 'wbc')
    .replace(/red_blood_cell|rbc$/, 'rbc')
    .replace(/fasting_glucose|blood_glucose|glucose_fasting$/, 'glucose')
    .replace(/ferritin_serum$/, 'ferritin')
    .replace(/tsh_serum$/, 'tsh')
    .replace(/total_cholesterol_$/, 'total_cholesterol');
}

export function evaluateStatus(value: number, refLow?: number, refHigh?: number, markerName?: string): MarkerStat
us {
  const norm = markerName ? normalizeMarkerName(markerName) : '';
  const crit = CRITICAL_THRESHOLDS[norm];
```

```
  if (crit) {
    if (crit.low !== undefined && value <= crit.low) return 'critical-low';
    if (crit.high !== undefined && value >= crit.high) return 'critical-high';
  }
  const low = refLow;
  const high = refHigh;
  if (low !== undefined && value < low) return 'low';
  if (high !== undefined && value > high) return 'high';
  if (low !== undefined && high !== undefined) return 'normal';
  return 'unknown';
}

export function isAbnormal(status: MarkerStatus): boolean {
  return status !== 'normal' && status !== 'unknown';
}

export function statusLabel(status: MarkerStatus): string {
  switch (status) {
    case 'normal': return 'Normal';
    case 'low': return 'Below range';
    case 'high': return 'Above range';
    case 'critical-low': return 'CRITICAL LOW';
    case 'critical-high': return 'CRITICAL HIGH';
    default: return 'No reference';
  }
}

// 从多份报告计算趋势
export function computeTrends(reports: { reportDate: Date; markers: LabMarker[] }[]): MarkerTrend[] {
  const sorted = [...reports].sort((a, b) => a.reportDate.getTime() - b.reportDate.getTime());
  const byMarker = new Map<string, { unit: string; points: TrendPoint[]; refLow?: number; refHigh?: number; lates
tStatus: MarkerStatus }>();
  for (const r of sorted) {
    for (const m of r.markers) {
      const key = m.name;
      if (!byMarker.has(key)) {
        byMarker.set(key, { unit: m.unit, points: [], refLow: m.refLow, refHigh: m.refHigh, latestStatus: m.statu
s });
      }
      const entry = byMarker.get(key)!;
      entry.points.push({ date: r.reportDate.toISOString().slice(0, 10), value: m.value });
      entry.latestStatus = m.status;
      if (m.refLow !== undefined) entry.refLow = m.refLow;
      if (m.refHigh !== undefined) entry.refHigh = m.refHigh;
    }
  }
  const trends: MarkerTrend[] = [];
  for (const [name, e] of byMarker) {
    if (e.points.length < 1) continue;
    let direction: 'up' | 'down' | 'flat' = 'flat';
    let changePct: number | undefined;
    if (e.points.length >= 2) {
      const first = e.points[0].value;
      const last = e.points[e.points.length - 1].value;
      if (first !== 0) changePct = Math.round(((last - first) / Math.abs(first)) * 1000) / 10;
      direction = changePct === undefined ? 'flat' : changePct > 2 ? 'up' : changePct < -2 ? 'down' : 'flat';
    }
    trends.push({ name, unit: e.unit, points: e.points, changePct, direction, status: e.latestStatus });
  }
  return trends.sort((a, b) => {
    const aAb = isAbnormal(a.status) ? 0 : 1;
    const bAb = isAbnormal(b.status) ? 0 : 1;
    if (aAb !== bAb) return aAb - bAb;
    return (b.changePct ? Math.abs(b.changePct) : 0) - (a.changePct ? Math.abs(a.changePct) : 0);
  });
}
```

src/lib/ai.ts — AI service layer — VLM lab extraction, LLM briefing, triage prompt, doctor prep

245 lines

```
// VitalLens AI 服务层（仅服务端）
import ZAI from 'z-ai-web-dev-sdk';
import { LabMarker, RedFlag, evaluateStatus, normalizeMarkerName, FALLBACK_RANGES, CRITICAL_THRESHOLDS } from './
medical';

export interface ExtractedReport {
  patientName: string;
```

```
  reportDate: string; // ISO date
  labName?: string;
  markers: LabMarker[];
}

// 第一步：VLM 从化验单图片提取结构化数据
export async function extractLabReport(imageBase64: string): Promise<ExtractedReport> {
  const zai = await ZAI.create();
  const response = await zai.chat.completions.createVision({
    messages: [
      {
        role: 'user',
        content: [
          {
            type: 'text',
            text: `You are a precise medical lab report parser. Analyze this lab report image and extract structu
red data.

Return STRICT JSON only (no markdown, no explanation) with this exact shape:
{
  "patientName": "string or 'Unknown'",
  "reportDate": "YYYY-MM-DD (best guess from document; if none visible use 2026-01-01)",
  "labName": "string or null",
  "markers": [
    {
      "name": "short standard marker name, e.g. Hemoglobin, Ferritin, TSH, Glucose",
      "value": number,
      "unit": "string e.g. g/dL",
      "refLow": number or null (reference range lower bound printed on the report),
      "refHigh": number or null (reference range upper bound printed on the report),
      "category": "one of: CBC, Iron Studies, Metabolic, Liver, Kidney, Electrolytes, Thyroid, Lipids, Vitamins,
Inflammation, Other"
    }
  ]
}

Rules:
- Extract EVERY numeric test marker visible. Include all panels (CBC, chemistry, lipids, etc).
- Use the reference ranges PRINTED ON THE REPORT when available; otherwise null.
- Values must be plain numbers (no commas, no symbols).
- If a value has '<' or '>' prefix, use the number and add it to name e.g. "Glucose (>400)".
- Do NOT invent markers that are not visible.`,
          },
          {
            type: 'image_url',
            image_url: { url: imageBase64.startsWith('data:') ? imageBase64 : `${imageBase
64}` },
          },
        ],
      },
    ],
    thinking: { type: 'disabled' },
  });

  const raw = response.choices[0]?.message?.content || '';
  return parseExtracted(raw);
}

function parseExtracted(raw: string): ExtractedReport {
  // 提取 JSON(容错：去掉可能的 markdown 代码块)
  let text = raw.trim();
  const fence = text.match(/```(?:json)?\s*([\s\S]*?)```/);
  if (fence) text = fence[1].trim();
  const start = text.indexOf('{');
  const end = text.lastIndexOf('}');
  if (start === -1 || end === -1) throw new Error('AI response did not contain JSON');
  let parsed: any;
  try {
    parsed = JSON.parse(text.slice(start, end + 1));
  } catch {
    throw new Error('Failed to parse extracted lab data');
  }
  const markers: LabMarker[] = (parsed.markers || [])
    .filter((m: any) => typeof m.value === 'number' && !Number.isNaN(m.value) && m.name)
    .map((m: any) => {
      const refLow = typeof m.refLow === 'number' ? m.refLow : undefined;
      const refHigh = typeof m.refHigh === 'number' ? m.refHigh : undefined;
      // 报告未标注范围时使用内置参考范围
      const norm = normalizeMarkerName(String(m.name));
      const fallback = FALLBACK_RANGES[norm];
      const effLow = refLow ?? fallback?.low;
```

```
      const effHigh = refHigh ?? fallback?.high;
      return {
        name: String(m.name),
        value: m.value,
        unit: String(m.unit || fallback?.unit || ''),
        refLow: effLow,
        refHigh: effHigh,
        category: String(m.category || fallback?.category || 'Other'),
        status: evaluateStatus(m.value, effLow, effHigh, String(m.name)),
      } as LabMarker;
    });
  return {
    patientName: String(parsed.patientName || 'Unknown'),
    reportDate: String(parsed.reportDate || new Date().toISOString().slice(0, 10)),
    labName: parsed.labName ? String(parsed.labName) : undefined,
    markers,
  };
}

// 第二步: LLM 生成通俗解读
export async function analyzeLabReport(extracted: ExtractedReport, history: { date: string; markers: LabMarker[]
}[] = []): Promise<{ summary: string; redFlags: RedFlag[] }> {
  const zai = await ZAI.create();

  // 确定性红旗检测(不依赖 LLM, 保证安全兜底)
  const redFlags: RedFlag[] = [];
  for (const m of extracted.markers) {
    const norm = normalizeMarkerName(m.name);
    const crit = CRITICAL_THRESHOLDS[norm];
    if (crit?.low !== undefined && m.value <= crit.low) {
      redFlags.push({
        marker: m.name,
        value: `${m.value} ${m.unit}`,
        severity: 'critical',
        message: `${m.name} is at a critically low level (${m.value} ${m.unit}). This can be a medical emergency
- seek urgent medical attention now.`,
      });
    } else if (crit?.high !== undefined && m.value >= crit.high) {
      redFlags.push({
        marker: m.name,
        value: `${m.value} ${m.unit}`,
        severity: 'critical',
        message: `${m.name} is at a critically high level (${m.value} ${m.unit}). This can be a medical emergency
- seek urgent medical attention now.`,
      });
    }
  }

  const markersJson = JSON.stringify(extracted.markers.map(m => ({ name: m.name, value: m.value, unit: m.unit, re
f: m.refLow !== undefined && m.refHigh !== undefined ? `${m.refLow}-${m.refHigh}` : 'n/a', status: m.status })));
  const historyJson = history.length
    ? JSON.stringify(history.map(h => ({ date: h.date, markers: h.markers.map(m => ({ name: m.name, value: m.valu
e })) })))
    : 'none';

  const prompt = `You are Dr. V, the clinical explainer inside VitalLens — an app that helps ordinary people unde
rstand their lab results. A user just uploaded a lab report.

EXTRACTED MARKERS (current report):
${markersJson}

PATIENT NAME: ${extracted.patientName}
REPORT DATE: ${extracted.reportDate}

PREVIOUS REPORTS HISTORY (for trend context):
${historyJson}

Write a plain-language health briefing in markdown with EXACTLY these sections:

## What this report shows
2-3 sentences overview in friendly, non-alarming language (reading level: general public).

## Key findings
A markdown list. For each notable marker (abnormal ones first): **MarkerName (value unit)** — what it means in on
e sentence, plus "→" what it could relate to. Keep each item to max 2 sentences. If everything is normal, say so
clearly.

## Trends over time
ONLY if history exists: compare current vs previous values, note direction and % change, and what the trend sugge
sts. If no history, write exactly: "No previous reports yet — upload past reports to unlock trend analysis."
```

```
## What you can do next
3-4 practical, safe, actionable bullets (lifestyle, monitoring, when to see a doctor). Never diagnose. Always fra
me as "discuss with a healthcare professional" where appropriate.

## Questions to ask your doctor
2-3 smart questions the user could bring to their next appointment.

RULES:
- NEVER state a diagnosis. Use "can be associated with", "often relates to".
- Be warm, clear, and calm. No medical jargon without a plain-language gloss.
- Do NOT repeat the JSON. Write for a human.
- Total length: 250-400 words.`;

  const response = await zai.chat.completions.create({
    messages: [{ role: 'user', content: prompt }],
    thinking: { type: 'disabled' },
  });

  return { summary: response.choices[0]?.message?.content || '', redFlags };
}

// 症状分诊对话
export const TRIAGE_SYSTEM_PROMPT = `You are Nurse V, the AI symptom guide inside VitalLens. You help users figur
e out what level of care they need. You are NOT a doctor and never diagnose.

YOUR PROCESS (follow strictly):
1. First message: warmly invite the user to describe what they're feeling. Ask 1-2 focused follow-up questions at
a time (onset, severity 1-10, associated symptoms, relevant history).
2. After you have enough context (usually 2-3 exchanges), give a structured assessment:

CARE LEVEL — one of:
• 🚨 EMERGENCY — call emergency services / go to ER now (red-flag symptoms: chest pain, difficulty breathing, str
oke signs like facial droop/one-sided weakness/slurred speech, severe bleeding, suicidal thoughts, fainting, seve
re allergic reaction)
• 🩺 SEE A DOCTOR — book an appointment within a few days
• 💊 SELF-CARE — manage at home, monitor, see a doctor if it worsens

Then: WHY (2-3 sentences reasoning), WHAT TO DO NOW (2-3 bullets), WATCH FOR (warning signs that would escalate t
he care level).

RULES:
- If ANY red-flag symptom appears at ANY point, lead with 🚨 EMERGENCY immediately and repeat the emergency advic
e.
- Never name specific diseases as conclusions; you may mention what symptoms "can be associated with" when explai
ning urgency.
- Be warm, brief, and structured. Use short paragraphs and bullets. Max 200 words per message.
- End conversational messages with your next question. End assessments with a supportive closing line.
- If asked something off-topic, gently steer back to their symptoms.`;

// 医生就诊准备
export async function generateDoctorPrep(patientName: string, reports: { date: string; markers: LabMarker[]; summ
ary?: string }[], symptomTranscript: string): Promise<string> {
  const zai = await ZAI.create();
  const reportsJson = JSON.stringify(reports.map(r => ({
    date: r.date,
    abnormal: r.markers.filter(m => m.status !== 'normal' && m.status !== 'unknown').map(m => `${m.name}: ${m.val
ue} ${m.unit} (ref ${m.refLow ?? '?'}-${m.refHigh ?? '?'})`),
    allCount: r.markers.length,
  })));

  const prompt = `Create a "Doctor Visit Prep Sheet" for ${patientName || 'the patient'} as markdown. This will b
e printed and brought to a doctor's appointment.

DATA:
Lab reports summary:
${reportsJson}

Symptom conversation transcript (may be empty):
${symptomTranscript || '(none recorded)'}

Format EXACTLY as:

## Visit Prep Sheet — [current date]

### 1. Why I'm coming
2-3 sentences in first person ("I"), summarizing the situation using the lab data and symptoms.

### 2. My recent lab results
Compact markdown table: | Marker | Value | Reference | Status | for abnormal + most relevant markers (max 10 row
s).
```

```
### 3. Changes over time
Brief trend notes (which markers moved and by how much).

### 4. My symptoms & timeline
Bullet list from the transcript, or "No symptom session recorded."

### 5. Questions I want to ask
4-5 specific, prioritized questions.

### 6. Medications & supplements I take
Write exactly: "(Space to fill in before the visit)"

RULES: First person voice. Concise. Factual. No diagnoses. 200-300 words.`;

  const response = await zai.chat.completions.create({
    messages: [{ role: 'user', content: prompt }],
    thinking: { type: 'disabled' },
  });
  return response.choices[0]?.message?.content || '';
}
```

src/lib/store.ts

Client state — Zustand store for cross-module sync

17 lines

```
'use client';

import { create } from 'zustand';

interface AppState {
  reportsVersion: number;
  bumpReports: () => void;
  symptomTranscript: string;
  setSymptomTranscript: (s: string) => void;
}

export const useAppStore = create<AppState>((set) => ({
  reportsVersion: 0,
  bumpReports: () => set((s) => ({ reportsVersion: s.reportsVersion + 1 })),
  symptomTranscript: '',
  setSymptomTranscript: (s) => set({ symptomTranscript: s }),
}));
```

src/lib/db.ts

Prisma client singleton

13 lines

```
import { PrismaClient } from '@prisma/client'

const globalForPrisma = globalThis as unknown as {
  prisma: PrismaClient | undefined
}

export const db =
  globalForPrisma.prisma ??
  new PrismaClient({
    log: ['query'],
  })

if (process.env.NODE_ENV !== 'production') globalForPrisma.prisma = db
```

src/app/layout.tsx

Root layout & metadata

45 lines

```
import type { Metadata } from "next";
import { Geist, Geist_Mono } from "next/font/google";
import "./globals.css";
import { Toaster } from "@/components/ui/toaster";

const geistSans = Geist({
  variable: "--font-geist-sans",
  subsets: ["latin"],
```

```
});

const geistMono = Geist_Mono({
  variable: "--font-geist-mono",
  subsets: ["latin"],
});

export const metadata: Metadata = {
  title: "VitalLens — Understand your health, earlier",
  description:
    "VitalLens turns lab reports and symptoms into clear, calm guidance. Vision AI reads your blood tests, trend
intelligence detects changes early, and Dr. V explains everything in plain language.",
  keywords: ["health AI", "lab report decoder", "symptom checker", "health trends", "UnivaBio"],
  authors: [{ name: "VitalLens Team" }],
  openGraph: {
    title: "VitalLens — Your blood tests, finally speaking your language",
    description: "Detect illness earlier. Understand it better. Live with it more easily.",
    siteName: "VitalLens",
    type: "website",
  },
};

export default function RootLayout({
  children,
}: Readonly<{
  children: React.ReactNode;
}>) {
  return (
    <html lang="en" suppressHydrationWarning>
      <body
        className={`${geistSans.variable} ${geistMono.variable} antialiased bg-background text-foreground`}
      >
        {children}
        <Toaster />
      </body>
    </html>
  );
}
```

src/app/page.tsx

Home page composition
21 lines

```
import { AppShell } from '@/components/vitallens/app-shell';
import { Hero } from '@/components/vitallens/hero';
import { HowItWorks, Safety, SiteFooter } from '@/components/vitallens/sections';

export default function Home() {
  return (
    <div className="flex min-h-screen flex-col bg-background">
      <Hero />
      <main className="flex-1 pb-20">
        <AppShell />
        <div className="mt-24">
          <HowItWorks />
        </div>
        <div className="mt-24">
          <Safety />
        </div>
      </main>
      <SiteFooter />
    </div>
  );
}
```

src/components/vitallens/hero.tsx

Hero section — value proposition & animated demo card
159 lines

```
'use client';

import { motion } from 'framer-motion';
import { Button } from '@/components/ui/button';
import { Badge } from '@/components/ui/badge';
import { ScanHeart, ArrowDown, Eye, MessageCircleHeart, ClipboardList, ShieldCheck } from 'lucide-react';
```

```tsx
const stats = [
  { value: '40+', label: 'blood markers recognized' },
  { value: '3', label: 'levels of care guidance' },
  { value: '< 30s', label: 'from photo to plain-language briefing' },
];

export function Hero() {
  return (
    <header className="relative overflow-hidden">
      {/* 背景装饰 */}
      <div className="pointer-events-none absolute inset-0 -z-10">
        <div className="absolute -top-32 left-1/2 h-[480px] w-[820px] -translate-x-1/2 rounded-full bg-teal-100/50 blur-3xl" />
        <div className="absolute right-[10%] top-40 h-56 w-56 rounded-full bg-emerald-100/60 blur-2xl" />
        <svg className="absolute inset-x-0 bottom-0 h-40 w-full text-teal-600/5" xmlns="http://www.w3.org/2000/svg">
          <defs>
            <pattern id="grid" width="32" height="32" patternUnits="userSpaceOnUse">
              <path d="M 32 0 L 0 0 0 32" fill="none" stroke="currentColor" strokeWidth="0.5" />
            </pattern>
          </defs>
          <rect width="100%" height="100%" fill="url(#grid)" />
        </svg>
      </div>

      {/* 导航 */}
      <nav className="mx-auto flex max-w-6xl items-center justify-between px-4 py-5 sm:px-6">
        <a href="#" className="flex items-center gap-2.5">
          <div className="rounded-xl bg-gradient-to-br from-teal-600 to-emerald-600 p-2 shadow-md shadow-teal-600/20">
            <ScanHeart className="h-5 w-5 text-white" />
          </div>
          <span className="text-lg font-bold tracking-tight">Vital<span className="text-teal-700">Lens</span></span>
        </a>
        <div className="flex items-center gap-3">
          <a href="#how" className="hidden text-sm font-medium text-muted-foreground transition-colors hover:text-foreground sm:block">How it works</a>
          <a href="#safety" className="hidden text-sm font-medium text-muted-foreground transition-colors hover:text-foreground sm:block">Safety</a>
          <Button asChild className="rounded-full bg-teal-700 px-5 hover:bg-teal-800">
            <a href="#app">Open the app</a>
          </Button>
        </div>
      </nav>

      <div className="mx-auto max-w-6xl px-4 pb-14 pt-10 sm:px-6 sm:pt-16">
        <div className="grid items-center gap-10 lg:grid-cols-[1.1fr_0.9fr]">
          <motion.div
            initial={{ opacity: 0, y: 24 }}
            animate={{ opacity: 1, y: 0 }}
            transition={{ duration: 0.6, ease: 'easeOut' }}
          >
            <Badge variant="outline" className="mb-5 gap-1.5 rounded-full border-teal-200 bg-teal-50 px-3 py-1 text-teal-800">
              <span className="relative flex h-2 w-2"><span className="absolute inline-flex h-full w-full animate-ping rounded-full bg-teal-500 opacity-60" /><span className="relative inline-flex h-2 w-2 rounded-full bg-teal-600" /></span>
              AI for Human Health · built for UnivaBio 2026
            </Badge>
            <h1 className="text-4xl font-extrabold leading-[1.08] tracking-tight sm:text-5xl lg:text-[3.4rem]">
              Your blood tests,
              <br />
              <span className="bg-gradient-to-r from-teal-600 to-emerald-600 bg-clip-text text-transparent">finally speaking your language.</span>
            </h1>
            <p className="mt-5 max-w-xl text-lg leading-relaxed text-muted-foreground">
              VitalLens turns lab reports and symptoms into clear, calm guidance — helping you
              <strong className="font-semibold text-foreground"> detect illness earlier</strong>,
              <strong className="font-semibold text-foreground"> understand it better</strong>, and
              <strong className="font-semibold text-foreground"> live with it more easily</strong>.
            </p>
            <div className="mt-8 flex flex-wrap gap-3">
              <Button asChild size="lg" className="gap-2 rounded-full bg-teal-700 px-7 text-base hover:bg-teal-800">
                <a href="#app"><Eye className="h-5 w-5" /> Decode my first report</a>
              </Button>
              <Button asChild size="lg" variant="outline" className="gap-2 rounded-full px-7 text-base">
                <a href="#how">How it works <ArrowDown className="h-4 w-4" /></a>
              </Button>
            </div>
```

```
            <div className="mt-10 grid max-w-lg grid-cols-3 gap-4">
              {stats.map((s, i) => (
                <motion.div
                  key={s.label}
                  initial={{ opacity: 0, y: 12 }}
                  animate={{ opacity: 1, y: 0 }}
                  transition={{ delay: 0.3 + i * 0.12, duration: 0.5 }}
                >
                  <p className="text-2xl font-bold text-teal-800 sm:text-3xl">{s.value}</p>
                  <p className="mt-1 text-xs leading-snug text-muted-foreground">{s.label}</p>
                </motion.div>
              ))}
            </div>
          </motion.div>

          {/* 右侧演示卡 */}
          <motion.div
            initial={{ opacity: 0, scale: 0.95, y: 20 }}
            animate={{ opacity: 1, scale: 1, y: 0 }}
            transition={{ duration: 0.7, delay: 0.15, ease: 'easeOut' }}
            className="relative hidden lg:block"
          >
            <div className="relative rotate-1 rounded-2xl border bg-white p-5 shadow-xl shadow-teal-900/10 transi
tion-transform hover:rotate-0">
              <div className="mb-3 flex items-center justify-between">
                <p className="text-xs font-semibold uppercase tracking-wider text-teal-700">Latest briefing</p>
                <Badge variant="outline" className="border-amber-200 bg-amber-50 text-[11px] text-amber-700">2 ma
rkers off</Badge>
              </div>
              <div className="space-y-2.5">
                {[
                  { n: 'Hemoglobin', v: '10.8 g/dL', r: '12.0–15.5', off: true },
                  { n: 'Ferritin', v: '14 ng/mL', r: '30–400', off: true },
                  { n: 'Glucose', v: '90 mg/dL', r: '70–99', off: false },
                  { n: 'TSH', v: '2.0 mIU/L', r: '0.4–4.0', off: false },
                ].map((m) => (
                  <div key={m.n} className={`flex items-center justify-between rounded-lg border px-3 py-2 ${m.of
f ? 'border-amber-200 bg-amber-50/70' : 'border-border bg-muted/30'}`}>
                    <span className="text-sm font-medium">{m.n}</span>
                    <span className="flex items-center gap-2 text-sm">
                      <span className="font-semibold">{m.v}</span>
                      <span className={`h-2 w-2 rounded-full ${m.off ? 'bg-amber-500' : 'bg-emerald-500'}`} />
                    </span>
                  </div>
                ))}
              </div>
              <div className="mt-4 rounded-lg bg-gradient-to-br from-teal-50 to-emerald-50 p-3.5">
                <p className="flex items-start gap-2 text-[13px] leading-relaxed text-teal-900">
                  <MessageCircleHeart className="mt-0.5 h-4 w-4 shrink-0 text-teal-700" />
                  <span><strong>Dr. V:</strong> Your iron stores have fallen 69% since March. This pattern usuall
y responds well to treatment — book a doctor&apos;s visit and ask about an iron workup.</span>
                </p>
              </div>
            </div>
            <motion.div
              animate={{ y: [0, -10, 0] }}
              transition={{ repeat: Infinity, duration: 3.5, ease: 'easeInOut' }}
              className="absolute -bottom-6 -left-8 -rotate-2 rounded-xl border bg-white p-3.5 shadow-lg"
            >
              <div className="flex items-center gap-2.5">
                <div className="rounded-lg bg-emerald-50 p-2"><ClipboardList className="h-4 w-4 text-emerald-600"
/></div>
                <div>
                  <p className="text-xs font-semibold">Visit prep ready</p>
                  <p className="text-[11px] text-muted-foreground">5 questions for your doctor</p>
                </div>
              </div>
            </motion.div>
            <motion.div
              animate={{ y: [0, 8, 0] }}
              transition={{ repeat: Infinity, duration: 4, ease: 'easeInOut', delay: 0.6 }}
              className="absolute -right-6 -top-6 rotate-3 rounded-xl border bg-white p-3.5 shadow-lg"
            >
              <div className="flex items-center gap-2.5">
                <div className="rounded-lg bg-teal-50 p-2"><ShieldCheck className="h-4 w-4 text-teal-600" /></div
>
                <div>
                  <p className="text-xs font-semibold">No diagnosis, ever</p>
                  <p className="text-[11px] text-muted-foreground">Guidance & education only</p>
                </div>
              </div>
```

```
            </motion.div>
          </motion.div>
        </div>
      </div>
    </header>
  );
}
```

src/components/vitallens/app-shell.tsx

App shell — 4-module tab navigation
40 lines

```tsx
'use client';

import { useState } from 'react';
import { Tabs, TabsContent, TabsList, TabsTrigger } from '@/components/ui/tabs';
import { FileImage, TrendingUp, HeartPulse, ClipboardList } from 'lucide-react';
import { LabDecoder } from './lab-decoder';
import { HealthTimeline } from './timeline';
import { SymptomChecker } from './symptom-checker';
import { DoctorPrep } from './doctor-prep';

export function AppShell() {
  const [tab, setTab] = useState('decoder');

  return (
    <div id="app" className="mx-auto w-full max-w-6xl px-4 sm:px-6">
      <Tabs value={tab} onValueChange={setTab} className="w-full">
        <div className="mb-6 flex justify-center">
          <TabsList className="grid h-auto w-full max-w-2xl grid-cols-2 rounded-xl p-1.5 sm:grid-cols-4">
            <TabsTrigger value="decoder" className="gap-2 rounded-lg py-2.5 data-[state=active]:shadow-sm">
              <FileImage className="h-4 w-4" /><span className="hidden sm:inline">Lab Decoder</span><span classNa
me="sm:hidden">Labs</span>
            </TabsTrigger>
            <TabsTrigger value="timeline" className="gap-2 rounded-lg py-2.5 data-[state=active]:shadow-sm">
              <TrendingUp className="h-4 w-4" /><span className="hidden sm:inline">Timeline</span><span className
="sm:hidden">Trends</span>
            </TabsTrigger>
            <TabsTrigger value="symptoms" className="gap-2 rounded-lg py-2.5 data-[state=active]:shadow-sm">
              <HeartPulse className="h-4 w-4" /><span className="hidden sm:inline">Symptom Guide</span><span clas
sName="sm:hidden">Nurse V</span>
            </TabsTrigger>
            <TabsTrigger value="doctor" className="gap-2 rounded-lg py-2.5 data-[state=active]:shadow-sm">
              <ClipboardList className="h-4 w-4" /><span className="hidden sm:inline">Doctor Prep</span><span cla
ssName="sm:hidden">Prep</span>
            </TabsTrigger>
          </TabsList>
        </div>
        <TabsContent value="decoder"><LabDecoder onGoTimeline={() => setTab('timeline')} /></TabsContent>
        <TabsContent value="timeline"><HealthTimeline /></TabsContent>
        <TabsContent value="symptoms"><SymptomChecker /></TabsContent>
        <TabsContent value="doctor"><DoctorPrep /></TabsContent>
      </Tabs>
    </div>
  );
}
```

src/components/vitallens/lab-decoder.tsx

Module 1: Lab Decoder — upload, VLM pipeline UI, markers table
460 lines

```tsx
'use client';

import { useCallback, useEffect, useRef, useState } from 'react';
import { Button } from '@/components/ui/button';
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';
import { Badge } from '@/components/ui/badge';
import { Progress } from '@/components/ui/progress';
import { Alert, AlertDescription, AlertTitle } from '@/components/ui/alert';
import {
  Dialog, DialogContent, DialogHeader, DialogTitle,
} from '@/components/ui/dialog';
import { Tabs, TabsContent, TabsList, TabsTrigger } from '@/components/ui/tabs';
import {
  UploadCloud, FileImage, Stethoscope, Activity, AlertTriangle, Sparkles,
  Loader2, CheckCircle2, TrendingUp, RotateCcw, Trash2, Eye, ArrowRight,
```

```tsx
} from 'lucide-react';
import { LabMarker, MarkerStatus, RedFlag, statusLabel } from '@/lib/medical';
import { Markdown } from './markdown';
import { useAppStore } from '@/lib/store';
import { toast } from '@/hooks/use-toast';

export interface Report {
  id: string;
  title: string;
  patientName: string;
  reportDate: string;
  labName?: string;
  markers: LabMarker[];
  summary?: string;
  redFlags: RedFlag[];
  sourceImage?: string | null;
}

const STATUS_STYLES: Record<MarkerStatus, { badge: string; row: string; dot: string }> = {
  normal: { badge: 'bg-emerald-50 text-emerald-700 border-emerald-200', row: '', dot: 'bg-emerald-500' },
  low: { badge: 'bg-amber-50 text-amber-700 border-amber-200', row: '', dot: 'bg-amber-500' },
  high: { badge: 'bg-amber-50 text-amber-700 border-amber-200', row: '', dot: 'bg-amber-500' },
  'critical-low': { badge: 'bg-red-50 text-red-700 border-red-200 font-semibold', row: 'bg-red-50/50', dot: 'bg-r
ed-600 animate-pulse' },
  'critical-high': { badge: 'bg-red-50 text-red-700 border-red-200 font-semibold', row: 'bg-red-50/50', dot: 'bg-
red-600 animate-pulse' },
  unknown: { badge: 'bg-muted text-muted-foreground border-border', row: '', dot: 'bg-gray-400' },
};

const PIPELINE_STEPS = [
  { key: 'read', label: 'Reading the report (vision AI)' },
  { key: 'extract', label: 'Extracting markers & reference ranges' },
  { key: 'analyze', label: 'Analyzing patterns & context' },
  { key: 'write', label: 'Writing your plain-language briefing' },
];

export function LabDecoder({ onGoTimeline }: { onGoTimeline: () => void }) {
  const [reports, setReports] = useState<Report[]>([]);
  const [selected, setSelected] = useState<Report | null>(null);
  const [uploading, setUploading] = useState(false);
  const [step, setStep] = useState(0);
  const [dragOver, setDragOver] = useState(false);
  const [previewImg, setPreviewImg] = useState<string | null>(null);
  const [loading, setLoading] = useState(true);
  const fileRef = useRef<HTMLInputElement>(null);
  const { reportsVersion, bumpReports } = useAppStore();

  const loadReports = useCallback(async () => {
    try {
      const res = await fetch('/api/lab-reports');
      const data = await res.json();
      const list: Report[] = data.reports || [];
      setReports(list);
      setSelected((prev) => prev ?? list[0] ?? null);
    } catch {
      toast({ title: 'Could not load reports', variant: 'destructive' });
    } finally {
      setLoading(false);
    }
  }, []);

  useEffect(() => { loadReports(); }, [loadReports, reportsVersion]);

  // 选择最新报告
  useEffect(() => {
    if (reports.length && !selected) setSelected(reports[0]);
  }, [reports, selected]);

  const analyze = useCallback(async (dataUrl: string) => {
    setUploading(true);
    setStep(0);
    setPreviewImg(dataUrl);
    const timers = [
      setTimeout(() => setStep(1), 1200),
      setTimeout(() => setStep(2), 3600),
      setTimeout(() => setStep(3), 6800),
    ];
    try {
      const res = await fetch('/api/lab-reports', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
```

```
        body: JSON.stringify({ image: dataUrl }),
      });
      const data = await res.json();
      timers.forEach(clearTimeout);
      if (!res.ok) throw new Error(data.error || 'Analysis failed');
      toast({ title: 'Report analyzed ✓', description: `${data.report.markers.length} markers extracted and interpreted.` });
      bumpReports();
      setSelected(data.report);
      setPreviewImg(null);
    } catch (e: any) {
      timers.forEach(clearTimeout);
      toast({ title: 'Analysis failed', description: e.message, variant: 'destructive' });
      setPreviewImg(null);
    } finally {
      setUploading(false);
      setStep(0);
    }
  }, [bumpReports]);

  const readFile = useCallback((file: File) => {
    if (!file.type.startsWith('image/')) {
      toast({ title: 'Please upload an image file', description: 'JPG, PNG or WebP photos of your lab report.', variant: 'destructive' });
      return;
    }
    const reader = new FileReader();
    reader.onload = () => analyze(reader.result as string);
    reader.readAsDataURL(file);
  }, [analyze]);

  const trySample = useCallback(async () => {
    try {
      const res = await fetch('/samples/sample-lab-report.png');
      const blob = await res.blob();
      const reader = new FileReader();
      reader.onload = () => analyze(reader.result as string);
      reader.readAsDataURL(blob);
    } catch {
      toast({ title: 'Sample unavailable', variant: 'destructive' });
    }
  }, [analyze]);

  const loadDemo = useCallback(async () => {
    try {
      const res = await fetch('/api/demo-seed', { method: 'POST' });
      if (!res.ok) throw new Error('seed failed');
      toast({ title: 'Demo timeline loaded', description: "Sarah Chen's 3 reports (Mar / Jun / Oct 2026) are ready." });
      bumpReports();
    } catch {
      toast({ title: 'Could not load demo data', variant: 'destructive' });
    }
  }, [bumpReports]);

  const deleteReport = useCallback(async (id: string) => {
    try {
      await fetch(`/api/lab-reports?id=${id}`, { method: 'DELETE' });
      setSelected(null);
      bumpReports();
      toast({ title: 'Report deleted' });
    } catch {
      toast({ title: 'Delete failed', variant: 'destructive' });
    }
  }, [bumpReports]);

  const abnormal = selected?.markers.filter((m) => m.status !== 'normal' && m.status !== 'unknown') || [];
  const critical = selected?.markers.filter((m) => m.status.startsWith('critical')) || [];
  const hasRedFlags = critical.length > 0 || (selected?.redFlags?.length || 0) > 0;

  return (
    <div className="grid gap-6 lg:grid-cols-[380px_1fr]">
      {/* 左侧: 上传区 + 报告列表 */}
      <div className="space-y-4">
        <Card className="overflow-hidden border-2 border-dashed border-teal-200 bg-gradient-to-b from-teal-50/60 to-white">
          <CardContent className="p-6">
            <div
              role="button"
              tabIndex={0}
              aria-label="Upload lab report image"
```

```
            onClick={() => fileRef.current?.click()}
            onKeyDown={(e) => e.key === 'Enter' && fileRef.current?.click()}
            onDragOver={(e) => { e.preventDefault(); setDragOver(true); }}
            onDragLeave={() => setDragOver(false)}
            onDrop={(e) => { e.preventDefault(); setDragOver(false); const f = e.dataTransfer.files?.[0]; if
(f) readFile(f); }}
            className={`flex cursor-pointer flex-col items-center justify-center rounded-xl border-2 border-das
hed px-4 py-8 text-center transition-all hover:border-teal-400 hover:bg-teal-50/50 ${dragOver ? 'border-teal-500
bg-teal-50 scale-[1.01]' : 'border-teal-200'}`}
          >
            <div className="mb-3 rounded-full bg-teal-600/10 p-3.5">
              <UploadCloud className="h-8 w-8 text-teal-700" />
            </div>
            <p className="font-semibold text-foreground">Upload a lab report</p>
            <p className="mt-1 text-sm text-muted-foreground">Drag & drop a photo, or click to browse.<br />PDF
exports of blood tests work great too (as images).</p>
          </div>
          <input
            ref={fileRef} type="file" accept="image/*" className="hidden"
            onChange={(e) => { const f = e.target.files?.[0]; if (f) readFile(f); e.target.value = ''; }}
          />
          <div className="mt-4 flex flex-col gap-2">
            <Button onClick={trySample} disabled={uploading} variant="outline" className="w-full justify-start
gap-2">
              <FileImage className="h-4 w-4 text-teal-700" /> Try a sample report image
            </Button>
            <Button onClick={loadDemo} disabled={uploading} variant="outline" className="w-full justify-start g
ap-2">
              <TrendingUp className="h-4 w-4 text-teal-700" /> Load demo timeline (3 reports)
            </Button>
          </div>
        </CardContent>
      </Card>

      {/* 报告历史 */}
      <Card>
        <CardHeader className="pb-3">
          <CardTitle className="flex items-center justify-between text-base">
            <span className="flex items-center gap-2"><Stethoscope className="h-4 w-4 text-teal-700" /> Your re
ports</span>
            <span className="text-xs font-normal text-muted-foreground">{reports.length} saved</span>
          </CardTitle>
        </CardHeader>
        <CardContent className="p-0">
          {loading ? (
            <div className="flex items-center justify-center py-10 text-muted-foreground"><Loader2 className="m
r-2 h-4 w-4 animate-spin" /> Loading…</div>
          ) : reports.length === 0 ? (
            <div className="px-6 py-10 text-center text-sm text-muted-foreground">
              <Activity className="mx-auto mb-2 h-8 w-8 opacity-30" />
              No reports yet. Upload one or load the demo timeline to explore.
            </div>
          ) : (
            <div className="max-h-96 overflow-y-auto vl-scroll">
              {reports.map((r) => {
                const abn = r.markers.filter((m) => m.status !== 'normal' && m.status !== 'unknown').length;
                return (
                  <button
                    key={r.id}
                    onClick={() => setSelected(r)}
                    className={`flex w-full items-start gap-3 border-b px-4 py-3 text-left transition-colors la
st:border-0 hover:bg-muted/60 ${selected?.id === r.id ? 'bg-teal-50/70 border-l-2 border-l-teal-600' : ''}`}
                  >
                    <div className="mt-0.5 rounded-md bg-teal-600/10 p-1.5"><FileImage className="h-4 w-4 text-
teal-700" /></div>
                    <div className="min-w-0 flex-1">
                      <p className="truncate text-sm font-medium">{new Date(r.reportDate).toLocaleDateString('e
n-US', { month: 'short', day: 'numeric', year: 'numeric' })} — {r.patientName}</p>
                      <p className="truncate text-xs text-muted-foreground">{r.markers.length} markers · {r.lab
Name || 'Lab report'}</p>
                    </div>
                    {abn > 0 && <Badge variant="outline" className="shrink-0 border-amber-200 bg-amber-50 text-
amber-700">{abn} off</Badge>}
                  </button>
                );
              })}
            </div>
          )}
        </CardContent>
      </Card>
    </div>
```

```
      {/* 右侧: 分析结果 */}
      <div className="space-y-4">
        {uploading && <AnalyzingCard step={step} preview={previewImg} />}

        {!uploading && !selected && !loading && (
          <EmptyState onTrySample={trySample} onLoadDemo={loadDemo} />
        )}

        {!uploading && selected && (
          <>
            {/* 红旗警告 */}
            {hasRedFlags && (
              <Alert variant="destructive" className="border-red-300 bg-red-50">
                <AlertTriangle className="h-5 w-5 text-red-600" />
                <AlertTitle className="text-red-800">Urgent values detected — please read</AlertTitle>
                <AlertDescription className="space-y-1 text-red-700">
                  {(selected.redFlags?.length ? selected.redFlags : critical.map((m) => ({
                    marker: m.name, value: `${m.value} ${m.unit}`, severity: 'critical' as const,
                    message: `${m.name} is at a critical level (${m.value} ${m.unit}). Please seek medical care p
romptly.`,
                  }))).map((f: RedFlag, i: number) => (
                    <p key={i}>• <strong>{f.marker} = {f.value}</strong> — {f.message}</p>
                  ))}
                </AlertDescription>
              </Alert>
            )}

            {/* 摘要统计 */}
            <div className="grid grid-cols-2 gap-3 sm:grid-cols-4">
              <StatCard icon={<FileImage className="h-4 w-4" />} label="Markers read" value={String(selected.mark
ers.length)} tone="teal" />
              <StatCard icon={<CheckCircle2 className="h-4 w-4" />} label="In range" value={String(selected.marke
rs.length - abnormal.length)} tone="emerald" />
              <StatCard icon={<Activity className="h-4 w-4" />} label="Out of range" value={String(abnormal.lengt
h)} tone={abnormal.length ? 'amber' : 'slate'} />
              <StatCard icon={<AlertTriangle className="h-4 w-4" />} label="Critical" value={String(critical.leng
th)} tone={critical.length ? 'red' : 'slate'} />
            </div>

            <Tabs defaultValue="briefing" className="w-full">
              <TabsList className="grid w-full grid-cols-2">
                <TabsTrigger value="briefing" className="gap-1.5"><Sparkles className="h-3.5 w-3.5" /> AI briefin
g</TabsTrigger>
                <TabsTrigger value="markers" className="gap-1.5"><Activity className="h-3.5 w-3.5" /> All markers
</TabsTrigger>
              </TabsList>
              <TabsContent value="briefing" className="mt-4">
                <Card>
                  <CardHeader className="flex-row items-center justify-between space-y-0 pb-3">
                    <div>
                      <CardTitle className="text-base">Your plain-language briefing</CardTitle>
                      <p className="mt-1 text-xs text-muted-foreground">
                        {new Date(selected.reportDate).toLocaleDateString('en-US', { month: 'long', day: 'numeri
c', year: 'numeric' })} · {selected.labName || 'Lab report'} · {selected.patientName}
                      </p>
                    </div>
                    <div className="flex gap-1">
                      {selected.sourceImage && (
                        <Button size="icon" variant="ghost" onClick={() => setPreviewImg(selected.sourceImage!)}
title="View original">
                          <Eye className="h-4 w-4" />
                        </Button>
                      )}
                      <Button size="icon" variant="ghost" onClick={() => deleteReport(selected.id)} title="Delete
report" className="text-muted-foreground hover:text-red-600">
                        <Trash2 className="h-4 w-4" />
                      </Button>
                    </div>
                  </CardHeader>
                  <CardContent>
                    {selected.summary ? <Markdown content={selected.summary} /> : <p className="text-sm text-mute
d-foreground">No briefing stored for this report.</p>}
                    <div className="mt-5 flex flex-wrap gap-2 border-t pt-4">
                      <Button onClick={onGoTimeline} variant="outline" size="sm" className="gap-1.5">
                        <TrendingUp className="h-3.5 w-3.5 text-teal-700" /> See trends across reports <ArrowRigh
t className="h-3.5 w-3.5" />
                      </Button>
                    </div>
                  </CardContent>
                </Card>
```

```
              </TabsContent>
              <TabsContent value="markers" className="mt-4">
                <Card>
                  <CardHeader className="pb-3"><CardTitle className="text-base">Extracted markers</CardTitle></CardHeader>
                  <CardContent className="p-0">
                    <div className="max-h-[560px] overflow-y-auto vl-scroll">
                      <table className="w-full text-sm">
                        <thead className="sticky top-0 bg-muted/80 backdrop-blur">
                          <tr className="text-left text-xs uppercase tracking-wide text-muted-foreground">
                            <th className="px-4 py-2.5 font-medium">Marker</th>
                            <th className="px-4 py-2.5 font-medium">Result</th>
                            <th className="hidden px-4 py-2.5 font-medium sm:table-cell">Reference</th>
                            <th className="px-4 py-2.5 text-right font-medium">Status</th>
                          </tr>
                        </thead>
                        <tbody>
                          {groupMarkers(selected.markers).map(([cat, ms]) => (
                            <>
                              <tr key={cat} className="bg-teal-50/50">
                                <td colSpan={4} className="px-4 py-1.5 text-xs font-semibold uppercase tracking-wide text-teal-800">{cat}</td>
                              </tr>
                              {ms.map((m) => {
                                const st = STATUS_STYLES[m.status];
                                return (
                                  <tr key={m.name} className={`border-b last:border-0 ${st.row}`}>
                                    <td className="px-4 py-2.5 font-medium">{m.name}</td>
                                    <td className="px-4 py-2.5">{m.value} <span className="text-xs text-muted-foreground">{m.unit}</span></td>
                                    <td className="hidden px-4 py-2.5 text-muted-foreground sm:table-cell">
                                      {m.refLow !== undefined && m.refHigh !== undefined ? `${m.refLow} – ${m.refHigh}` : '—'}
                                    </td>
                                    <td className="px-4 py-2.5 text-right"><span className={`inline-flex items-center gap-1.5 rounded-full border px-2 py-0.5 text-xs ${st.badge}`}><span className={`h-1.5 w-1.5 rounded-full ${st.dot}`} />{statusLabel(m.status)}</span></td>
                                  </tr>
                                );
                              })}
                            </>
                          ))}
                        </tbody>
                      </table>
                    </div>
                  </CardContent>
                </Card>
              </TabsContent>
            </Tabs>
          </>
        )}
      </div>

      <Dialog open={!!previewImg && !uploading} onOpenChange={(o) => !o && setPreviewImg(null)}>
        <DialogContent className="max-w-2xl">
          <DialogHeader><DialogTitle>Original report</DialogTitle></DialogHeader>
          {previewImg && <img src={previewImg} alt="Uploaded lab report" className="max-h-[70vh] w-full rounded-lg object-contain" />}
        </DialogContent>
      </Dialog>
    </div>
  );
}

function groupMarkers(markers: LabMarker[]): [string, LabMarker[]][] {
  const map = new Map<string, LabMarker[]>();
  for (const m of markers) {
    const cat = m.category || 'Other';
    if (!map.has(cat)) map.set(cat, []);
    map.get(cat)!.push(m);
  }
  return Array.from(map.entries());
}

function AnalyzingCard({ step, preview }: { step: number; preview: string | null }) {
  return (
    <Card className="border-teal-200 bg-gradient-to-b from-teal-50/50 to-white">
      <CardContent className="p-6">
        <div className="mb-4 flex items-center gap-3">
          <div className="relative">
            <div className="h-10 w-10 rounded-full bg-teal-600/10 p-2"><Sparkles className="h-full w-full animate
```

```
-pulse text-teal-700" /></div>
          </div>
          <div>
            <p className="font-semibold">Analyzing your report…</p>
            <p className="text-xs text-muted-foreground">Vision AI + clinical language model working together</p>
          </div>
        </div>
        <div className="space-y-3">
          {PIPELINE_STEPS.map((s, i) => (
            <div key={s.key} className="flex items-center gap-3">
              {i < step ? <CheckCircle2 className="h-4 w-4 shrink-0 text-emerald-600" />
                : i === step ? <Loader2 className="h-4 w-4 shrink-0 animate-spin text-teal-700" />
                : <div className="h-4 w-4 shrink-0 rounded-full border-2 border-muted" />}
              <span className={`text-sm ${i <= step ? 'text-foreground' : 'text-muted-foreground'}`}>{s.label}</s
pan>
            </div>
          ))}
        </div>
        <Progress value={((step + 1) / PIPELINE_STEPS.length) * 100} className="mt-4 h-1.5" />
        {preview && (
          <div className="mt-5 flex justify-center rounded-lg border bg-white p-2">
            <img src={preview} alt="Report being analyzed" className="max-h-48 rounded object-contain opacity-80"
/>
          </div>
        )}
      </CardContent>
    </Card>
  );
}

function EmptyState({ onTrySample, onLoadDemo }: { onTrySample: () => void; onLoadDemo: () => void }) {
  return (
    <Card className="flex min-h-[420px] items-center justify-center border-dashed">
      <CardContent className="p-10 text-center">
        <div className="mx-auto mb-4 w-fit rounded-2xl bg-gradient-to-br from-teal-500 to-emerald-600 p-4 shadow-
lg shadow-teal-600/20">
          <ScanHeart className="h-10 w-10 text-white" />
        </div>
        <h3 className="text-xl font-semibold">Turn a lab report into understanding</h3>
        <p className="mx-auto mt-2 max-w-md text-sm text-muted-foreground">
          Snap a photo of any blood test result. VitalLens reads every marker with vision AI,
          flags anything out of range, and explains what it all means — in plain language.
        </p>
        <div className="mt-6 flex flex-wrap justify-center gap-3">
          <Button onClick={onTrySample} className="gap-2 bg-teal-700 hover:bg-teal-800"><FileImage className="h-4
w-4" /> Try a sample report</Button>
          <Button onClick={onLoadDemo} variant="outline" className="gap-2"><RotateCcw className="h-4 w-4" /> Load
demo timeline</Button>
        </div>
      </CardContent>
    </Card>
  );
}

function ScanHeart(props: React.SVGProps<SVGSVGElement>) {
  return (
    <svg viewBox="0 0 24 24" fill="none" stroke="currentColor" strokeWidth="2" strokeLinecap="round" strokeLinejo
in="round" {...props}>
      <path d="M11.6 3.5 4 8.5v7l7.6 5 7.6-5v-7Z" />
      <path d="M12 11.5h.01" />
      <path d="M8.5 15a3.5 3.5 0 0 1 7 0" />
    </svg>
  );
}

function StatCard({ icon, label, value, tone }: { icon: React.ReactNode; label: string; value: string; tone: 'tea
l' | 'emerald' | 'amber' | 'red' | 'slate' }) {
  const tones: Record<string, string> = {
    teal: 'text-teal-700 bg-teal-50 border-teal-100',
    emerald: 'text-emerald-700 bg-emerald-50 border-emerald-100',
    amber: 'text-amber-700 bg-amber-50 border-amber-100',
    red: 'text-red-700 bg-red-50 border-red-100',
    slate: 'text-muted-foreground bg-muted/50 border-border',
  };
  return (
    <Card className={`border ${tones[tone]}`}>
      <CardContent className="flex items-center gap-3 p-4">
        <div className="rounded-lg bg-white/70 p-2 shadow-sm">{icon}</div>
        <div>
          <p className="text-2xl font-bold leading-none">{value}</p>
          <p className="mt-1 text-xs opacity-80">{label}</p>
```

```
        </div>
      </CardContent>
    </Card>
  );
}
```

src/components/vitallens/timeline.tsx

Module 2: Health Timeline — trend UI & reference-band charts
247 lines

```tsx
'use client';

import { useEffect, useState } from 'react';
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';
import { Badge } from '@/components/ui/badge';
import { Button } from '@/components/ui/button';
import { Skeleton } from '@/components/ui/skeleton';
import {
  LineChart, Line, XAxis, YAxis, CartesianGrid, Tooltip, ResponsiveContainer,
  ReferenceArea,
} from 'recharts';
import { TrendingUp, TrendingDown, Minus, AlertCircle, Loader2, Sparkles } from 'lucide-react';
import { MarkerTrend, isAbnormal, FALLBACK_RANGES, normalizeMarkerName } from '@/lib/medical';
import { useAppStore } from '@/lib/store';
import { Markdown } from './markdown';

import { toast } from '@/hooks/use-toast';

export function HealthTimeline() {
  const [trends, setTrends] = useState<MarkerTrend[]>([]);
  const [loading, setLoading] = useState(true);
  const [selectedTrend, setSelectedTrend] = useState<MarkerTrend | null>(null);
  const [aiTrendNote, setAiTrendNote] = useState<string | null>(null);
  const [noteLoading, setNoteLoading] = useState(false);
  const { reportsVersion } = useAppStore();

  useEffect(() => {
    (async () => {
      setLoading(true);
      try {
        const res = await fetch('/api/lab-reports/trends');
        const data = await res.json();
        const t: MarkerTrend[] = data.trends || [];
        setTrends(t);
        const interesting = t.find((x) => isAbnormal(x.status) && x.points.length >= 2) || t.find((x) => x.point
s.length >= 2) || null;
        setSelectedTrend(interesting);
      } finally {
        setLoading(false);
      }
    })();
  }, [reportsVersion]);

  // 生成 AI 全史解读
  const askAiTrend = async () => {
    if (!selectedTrend) return;
    setNoteLoading(true);
    setAiTrendNote(null);
    try {
      const res = await fetch('/api/doctor-prep', { method: 'POST', headers: { 'Content-Type': 'application/json'
}, body: '{}' });
      const sd = await res.json();
      if (!res.ok) throw new Error(sd.error || 'failed');
      setAiTrendNote(sd.prep || null);
    } catch {
      setAiTrendNote(null);
      toast({ title: 'Could not generate review', variant: 'destructive' });
    } finally {
      setNoteLoading(false);
    }
  };

  if (loading) {
    return (
      <div className="space-y-4">
        <Skeleton className="h-8 w-64" />
        <div className="grid gap-6 lg:grid-cols-[320px_1fr]">
          <div className="space-y-2">{[1, 2, 3, 4, 5].map((i) => <Skeleton key={i} className="h-14 w-full" />)}</
```

```
div>
          <Skeleton className="h-80 w-full" />
        </div>
      </div>
    );
  }

  const trackedTrends = trends.filter((t) => t.points.length >= 2);
  const singleTrends = trends.filter((t) => t.points.length === 1);

  if (!trends.length) {
    return (
      <Card className="flex min-h-[320px] items-center justify-center border-dashed">
        <CardContent className="p-8 text-center">
          <TrendingUp className="mx-auto mb-3 h-10 w-10 text-teal-700/50" />
          <h3 className="font-semibold">No history yet</h3>
          <p className="mx-auto mt-1 max-w-sm text-sm text-muted-foreground">
            Upload at least two lab reports from different dates to unlock trend intelligence —
            VitalLens watches how your markers move over time.
          </p>
        </CardContent>
      </Card>
    );
  }

  return (
    <div className="space-y-5">
      {/* 概览条 */}
      <div className="flex flex-wrap items-center gap-3 rounded-xl border bg-gradient-to-r from-teal-50/80 to-tra
nsparent p-4">
        <div className="rounded-lg bg-teal-600/10 p-2"><TrendingUp className="h-5 w-5 text-teal-700" /></div>
        <div className="flex-1">
          <p className="text-sm font-semibold">{trackedTrends.length} markers tracked across reports</p>
          <p className="text-xs text-muted-foreground">VitalLens compares every new report with your history to c
atch changes early.</p>
        </div>
        {trackedTrends.filter((t) => isAbnormal(t.status)).length > 0 && (
          <Badge variant="outline" className="border-amber-200 bg-amber-50 text-amber-700">
            <AlertCircle className="mr-1 h-3 w-3" /> {trackedTrends.filter((t) => isAbnormal(t.status)).length} n
eed attention
          </Badge>
        )}
      </div>

      <div className="grid gap-6 lg:grid-cols-[320px_1fr]">
        {/* 左: 趋势列表 */}
        <Card className="h-fit">
          <CardHeader className="pb-3">
            <CardTitle className="text-base">Marker trends</CardTitle>
          </CardHeader>
          <CardContent className="p-0">
            <div className="max-h-[520px] overflow-y-auto vl-scroll">
              {trackedTrends.map((t) => {
                const active = selectedTrend?.name === t.name;
                const icon = t.direction === 'up' ? <TrendingUp className="h-4 w-4" />
                  : t.direction === 'down' ? <TrendingDown className="h-4 w-4" />
                  : <Minus className="h-4 w-4" />;
                return (
                  <button
                    key={t.name}
                    onClick={() => { setSelectedTrend(t); setAiTrendNote(null); }}
                    className={`flex w-full items-center gap-3 border-b px-4 py-3 text-left transition-colors las
t:border-0 hover:bg-muted/60 ${active ? 'bg-teal-50/70 border-l-2 border-l-teal-600' : ''}`}
                  >
                    <span className={`rounded-md p-1.5 ${
                      isAbnormal(t.status) ? 'bg-amber-50 text-amber-600' : 'bg-emerald-50 text-emerald-600'
                    }`}>{icon}</span>
                    <div className="min-w-0 flex-1">
                      <p className="truncate text-sm font-medium">{t.name}</p>
                      <p className="text-xs text-muted-foreground">
                        {t.points[t.points.length - 1].value} {t.unit}
                        {t.changePct !== undefined && ` · ${t.changePct > 0 ? '+' : ''}${t.changePct}%`}
                      </p>
                    </div>
                    {isAbnormal(t.status) && <span className="h-2 w-2 rounded-full bg-amber-500" />}
                  </button>
                );
              })}
              {singleTrends.length > 0 && (
                <div className="border-t bg-muted/30 px-4 py-2">
                  <p className="text-xs font-medium text-muted-foreground">Only one data point (needs 2+ for tren
```

```
ds):</p>
                  <div className="mt-1.5 flex flex-wrap gap-1">
                    {singleTrends.slice(0, 10).map((t) => (
                      <Badge key={t.name} variant="outline" className="text-[11px] font-normal text-muted-foregro
und">{t.name}</Badge>
                    ))}
                  </div>
                </div>
              )}
            </div>
          </CardContent>
        </Card>

        {/* 右: 图表 */}
        <Card>
          <CardHeader className="pb-2">
            <CardTitle className="flex items-center justify-between text-base">
              <span>{selectedTrend?.name || 'Select a marker'}</span>
              {selectedTrend?.changePct !== undefined && (
                <Badge variant="outline" className={
                  selectedTrend.changePct > 0 ? 'border-amber-200 bg-amber-50 text-amber-700'
                  : selectedTrend.changePct < 0 ? 'border-teal-200 bg-teal-50 text-teal-700'
                  : 'border-border bg-muted text-muted-foreground'
                }>
                  {selectedTrend.changePct > 0 ? '▲' : selectedTrend.changePct < 0 ? '▼' : '–'} {Math.abs(selecte
dTrend.changePct)}% over {selectedTrend?.points.length} reports
                </Badge>
              )}
            </CardTitle>
          </CardHeader>
          <CardContent>
            {selectedTrend ? (
              <div className="h-72 w-full">
                <ResponsiveContainer width="100%" height="100%">
                  <LineChart data={selectedTrend.points} margin={{ top: 8, right: 16, bottom: 0, left: -8 }}>
                    <CartesianGrid strokeDasharray="3 3" stroke="#e5e7eb" />
                    <XAxis dataKey="date" tick={{ fontSize: 12, fill: '#6b7280' }} tickLine={false} />
                    <YAxis tick={{ fontSize: 12, fill: '#6b7280' }} tickLine={false} domain={yDomain(selectedTren
d)} />
                    <Tooltip
                      contentStyle={{ borderRadius: 10, border: '1px solid #e5e7eb', fontSize: 13 }}
                      formatter={(v: number) => [`${v} ${selectedTrend.unit}`, selectedTrend.name]}
                    />
                    {(() => {
                      const range = reportsRefForRange(selectedTrend.name);
                      if (!range) return null;
                      const dom = yDomain(selectedTrend);
                      const dMin = typeof dom[0] === 'number' ? dom[0] : range.low;
                      const dMax = typeof dom[1] === 'number' ? dom[1] : range.high;
                      // 裁剪参考带到可见域，避免 recharts 丢弃越界的 ReferenceArea
                      const y1 = Math.max(range.low, dMin);
                      const y2 = Math.min(range.high, dMax);
                      if (y2 <= y1) return null;
                      return <ReferenceArea y1={y1} y2={y2} fill="#10b981" fillOpacity={0.08} />;
                    })()}
                    <Line
                      type="monotone" dataKey="value" stroke="#0f766e" strokeWidth={2.5}
                      dot={{ r: 4, fill: '#0f766e' }} activeDot={{ r: 6 }}
                    />
                  </LineChart>
                </ResponsiveContainer>
                <p className="mt-2 text-center text-xs text-muted-foreground">
                  Green band = normal reference range. Falling outside the band doesn&apos;t mean a diagnosis — a
lways discuss results with a clinician.
                </p>
              </div>
            ) : (
              <div className="flex h-72 items-center justify-center text-sm text-muted-foreground">Select a marke
r to see its trend</div>
            )}
            <div className="mt-4 border-t pt-4">
              {noteLoading ? (
                <div className="flex items-center gap-2 text-sm text-muted-foreground"><Loader2 className="h-4 w-
4 animate-spin text-teal-700" /> Dr. V is reviewing your full history…</div>
              ) : aiTrendNote ? (
                <div className="rounded-lg border border-teal-100 bg-teal-50/40 p-4">
                  <p className="mb-2 flex items-center gap-1.5 text-sm font-medium text-teal-800"><Sparkles class
Name="h-4 w-4" /> Full-history review by Dr. V</p>
                  <Markdown content={aiTrendNote} className="text-sm" />
                </div>
              ) : (
```

```
                <Button onClick={askAiTrend} variant="outline" size="sm" className="gap-2">
                  <Sparkles className="h-3.5 w-3.5 text-teal-700" /> Ask Dr. V to review my full history
                </Button>
              )}
            </div>
          </CardContent>
        </Card>
      </div>
    </div>
  );
}

// 从内置参考范围取图表用正常区间
function reportsRefForRange(name: string): { low: number; high: number } | null {
  const fb = FALLBACK_RANGES[normalizeMarkerName(name)];
  return fb ? { low: fb.low, high: fb.high } : null;
}

// Y 轴范围: 确保数据与参考带都可见,同时避免超大参考范围压扁曲线
function yDomain(t: MarkerTrend): [number | string, number | string] {
  const values = t.points.map((p) => p.value);
  const dataMin = Math.min(...values);
  const dataMax = Math.max(...values);
  const range = reportsRefForRange(t.name);
  if (!range) return ['auto', 'auto'];
  const span = dataMax - dataMin || Math.abs(dataMax) * 0.2 || 1;
  // 参考带上限若远超数据,限制在数据最大值 + 1个数据跨度,避免压扁
  const effHigh = Math.min(range.high, dataMax + span);
  const effLow = Math.min(range.low, dataMin);
  const pad = (effHigh - effLow) * 0.12 || 1;
  return [Math.max(0, effLow - pad), effHigh + pad];
}
```

src/components/vitallens/symptom-checker.tsx

Module 3: Nurse V — conversational triage chat

222 lines

```
'use client';

import { useEffect, useRef, useState } from 'react';
import { Button } from '@/components/ui/button';
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';
import { Alert, AlertDescription, AlertTitle } from '@/components/ui/alert';
import { Textarea } from '@/components/ui/textarea';
import { Badge } from '@/components/ui/badge';
import { HeartPulse, Send, Loader2, ShieldAlert, RotateCcw, User, Stethoscope } from 'lucide-react';
import { useAppStore } from '@/lib/store';
import { toast } from '@/hooks/use-toast';

interface Msg { role: 'user' | 'assistant'; content: string; ts: number }

const SUGGESTIONS = [
  'I have a headache and mild fever since yesterday',
  'My chest feels tight when I climb stairs',
  "I've been unusually tired for two weeks",
  'My child has a rash and mild fever',
];

const WELCOME: Msg = {
  role: 'assistant',
  ts: Date.now(),
  content: `Hi, I'm **Nurse V** 👋 I'm here to help you figure out **what level of care you might need** — whether that's self-care at home, a doctor's visit, or urgent emergency care.

Tell me: **what are you feeling right now?** (For example: where it hurts, when it started, how bad it is on a 1-10 scale.)`,
};

export function SymptomChecker() {
  const [messages, setMessages] = useState<Msg[]>([WELCOME]);
  const [input, setInput] = useState('');
  const [thinking, setThinking] = useState(false);
  const scrollRef = useRef<HTMLDivElement>(null);
  const { setSymptomTranscript } = useAppStore();

  useEffect(() => {
    scrollRef.current?.scrollTo({ top: scrollRef.current.scrollHeight, behavior: 'smooth' });
  }, [messages, thinking]);
```

```
  // 保存转录(供 Doctor Prep 使用, 持久化以跨页面)
  useEffect(() => {
    const transcript = messages
      .filter((m) => m !== WELCOME)
      .map((m) => `${m.role === 'user' ? 'Patient' : 'Nurse V'}: ${m.content}`)
      .join('\n');
    setSymptomTranscript(transcript);
    try { localStorage.setItem('vl_symptoms', transcript); } catch { /* ignore */ }
  }, [messages, setSymptomTranscript]);

  const send = async (text?: string) => {
    const content = (text ?? input).trim();
    if (!content || thinking) return;
    const next: Msg[] = [...messages, { role: 'user', content, ts: Date.now() }];
    setMessages(next);
    setInput('');
    setThinking(true);
    try {
      const res = await fetch('/api/symptom-checker', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ messages: next.filter((m) => m !== WELCOME).map(({ role, content }) => ({ role, co
ntent })) }),
      });
      const data = await res.json();
      if (!res.ok) throw new Error(data.error);
      setMessages([...next, { role: 'assistant', content: data.reply, ts: Date.now() }]);
    } catch (e: any) {
      toast({ title: 'Message failed', description: 'Please try again.', variant: 'destructive' });
    } finally {
      setThinking(false);
    }
  };

  const reset = () => {
    setMessages([WELCOME]);
    setInput('');
  };

  const hasEmergency = messages.some((m) => m.content.includes('🚨'));

  return (
    <div className="grid gap-5 lg:grid-cols-[1fr_300px]">
      <Card className="flex h-[620px] flex-col">
        <CardHeader className="flex-row items-center justify-between space-y-0 border-b pb-4">
          <div className="flex items-center gap-3">
            <div className="rounded-full bg-gradient-to-br from-teal-500 to-emerald-600 p-2.5 shadow-md shadow-te
al-600/20">
              <HeartPulse className="h-5 w-5 text-white" />
            </div>
            <div>
              <CardTitle className="text-base">Nurse V — Symptom Guide</CardTitle>
              <p className="text-xs text-muted-foreground">Triage guidance · never a diagnosis</p>
            </div>
          </div>
          <Button variant="ghost" size="icon" onClick={reset} title="Start over"><RotateCcw className="h-4 w-4" /
></Button>
        </CardHeader>

        <div ref={scrollRef} className="vl-scroll flex-1 space-y-4 overflow-y-auto p-4 sm:p-5">
          {messages.map((m, i) => (
            <MessageBubble key={i} msg={m} />
          ))}
          {thinking && (
            <div className="flex items-center gap-2.5 text-sm text-muted-foreground">
              <div className="rounded-2xl rounded-bl-sm border bg-white p-3 shadow-sm">
                <div className="flex gap-1">
                  <span className="h-2 w-2 animate-bounce rounded-full bg-teal-500 [animation-delay:0ms]" />
                  <span className="h-2 w-2 animate-bounce rounded-full bg-teal-500 [animation-delay:150ms]" />
                  <span className="h-2 w-2 animate-bounce rounded-full bg-teal-500 [animation-delay:300ms]" />
                </div>
              </div>
              Nurse V is thinking…
            </div>
          )}
        </div>

        <div className="border-t p-3 sm:p-4">
          {messages.length <= 1 && (
            <div className="mb-3 flex flex-wrap gap-2">
```

```
              {SUGGESTIONS.map((s) => (
                <button
                  key={s}
                  onClick={() => send(s)}
                  className="rounded-full border bg-muted/40 px-3 py-1.5 text-xs text-muted-foreground transition
-colors hover:border-teal-300 hover:bg-teal-50 hover:text-teal-800"
                >
                  {s}
                </button>
              ))}
            </div>
          )}
          <div className="flex gap-2">
            <Textarea
              value={input}
              onChange={(e) => setInput(e.target.value)}
              onKeyDown={(e) => {
                if (e.key === 'Enter' && !e.shiftKey) { e.preventDefault(); send(); }
              }}
              placeholder="Describe your symptoms… (Enter to send)"
              className="min-h-[52px] resize-none border-teal-200 focus-visible:ring-teal-500"
              aria-label="Describe your symptoms"
              maxRows={3}
            />
            <Button onClick={() => send()} disabled={thinking || !input.trim()} className="aspect-square h-[52px]
w-[52px] shrink-0 bg-teal-700 p-0 hover:bg-teal-800" aria-label="Send">
              <Send className="h-5 w-5" />
            </Button>
          </div>
        </div>
      </Card>

      {/* 侧栏 */}
      <div className="space-y-4">
        {hasEmergency ? (
          <Alert variant="destructive" className="border-red-300 bg-red-50">
            <ShieldAlert className="h-5 w-5 text-red-600" />
            <AlertTitle className="text-red-800">Emergency guidance active</AlertTitle>
            <AlertDescription className="text-red-700">
              If you or someone near you has emergency symptoms — call your local emergency number now. Do not wa
it.
            </AlertDescription>
          </Alert>
        ) : (
          <Alert className="border-teal-200 bg-teal-50/60">
            <Stethoscope className="h-5 w-5 text-teal-700" />
            <AlertTitle className="text-teal-900">How triage works</AlertTitle>
            <AlertDescription className="text-muted-foreground">
              Nurse V asks focused questions, then suggests one of three care levels:
            </AlertDescription>
            <div className="mt-2 space-y-1.5 text-xs">
              <Badge variant="outline" className="w-full justify-start border-red-200 bg-red-50 text-red-700">🚨
Emergency — get help now</Badge>
              <Badge variant="outline" className="w-full justify-start border-amber-200 bg-amber-50 text-amber-70
0">🩺 See a doctor — book soon</Badge>
              <Badge variant="outline" className="w-full justify-start border-emerald-200 bg-emerald-50 text-emer
ald-700">💊 Self-care — monitor at home</Badge>
            </div>
          </Alert>
        )}
        <Card>
          <CardContent className="p-4 text-xs leading-relaxed text-muted-foreground">
            <p className="mb-2 font-semibold text-foreground">Privacy first</p>
            Conversations stay on your device&apos;s session. Nothing is shared with insurance companies or emplo
yers. VitalLens is an educational tool and does not replace professional medical advice, diagnosis, or treatment.
          </CardContent>
        </Card>
      </div>
    </div>
  );
}

function MessageBubble({ msg }: { msg: Msg }) {
  const isUser = msg.role === 'user';
  const isEmergency = msg.content.includes('🚨');
  return (
    <div className={`flex gap-2.5 ${isUser ? 'flex-row-reverse' : ''}`}>
      <div className={`mt-0.5 shrink-0 rounded-full p-1.5 ${isUser ? 'bg-muted' : 'bg-gradient-to-br from-teal-50
0 to-emerald-600'}`}>
        {isUser ? <User className="h-3.5 w-3.5 text-muted-foreground" /> : <Stethoscope className="h-3.5 w-3.5 te
xt-white" />}
```

```
      </div>
      <div className={`max-w-[85%] rounded-2xl px-4 py-3 text-sm leading-relaxed shadow-sm ${
        isUser
          ? 'rounded-br-sm bg-teal-700 text-white'
          : isEmergency
            ? 'rounded-bl-sm border-2 border-red-200 bg-red-50'
            : 'rounded-bl-sm border bg-white'
      }`}>
        {isUser ? <p className="whitespace-pre-wrap">{msg.content}</p> : <MarkdownLite content={msg.content} emer
gency={isEmergency} />}
      </div>
    </div>
  );
}

import ReactMarkdown from 'react-markdown';
import remarkGfm from 'remark-gfm';
function MarkdownLite({ content, emergency }: { content: string; emergency?: boolean }) {
  return (
    <div className={emergency ? 'text-red-800' : 'text-foreground/90'}>
      <ReactMarkdown
        remarkPlugins={[remarkGfm]}
        components={{
          p: ({ children }) => <p className="mb-2 last:mb-0">{children}</p>,
          ul: ({ children }) => <ul className="mb-2 ml-4 list-disc space-y-1">{children}</ul>,
          ol: ({ children }) => <ol className="mb-2 ml-4 list-decimal space-y-1">{children}</ol>,
          strong: ({ children }) => <strong className={emergency ? 'text-red-900' : 'font-semibold text-foregroun
d'}>{children}</strong>,
        }}
      >
        {content}
      </ReactMarkdown>
    </div>
  );
}
```

src/components/vitallens/doctor-prep.tsx — Module 4: Doctor Visit Prep — sheet generation, print & download — 170 lines

```
'use client';

import { useState } from 'react';
import { Button } from '@/components/ui/button';
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';
import { Alert, AlertDescription, AlertTitle } from '@/components/ui/alert';
import { ClipboardList, Loader2, Printer, Sparkles, FileDown, CalendarCheck } from 'lucide-react';
import { useAppStore } from '@/lib/store';
import { Markdown } from './markdown';
import { toast } from '@/hooks/use-toast';

export function DoctorPrep() {
  const [prep, setPrep] = useState<string | null>(null);
  const [loading, setLoading] = useState(false);
  const [meta, setMeta] = useState<{ patientName: string; reportCount: number; withSymptoms: boolean } | null>(nu
ll);
  const { symptomTranscript } = useAppStore();

  const generate = async () => {
    setLoading(true);
    // 读取持久化的症状转录(跨页面/会话)
    let transcript = symptomTranscript;
    try {
      const stored = localStorage.getItem('vl_symptoms');
      if (stored) transcript = stored;
    } catch { /* ignore */ }
    try {
      const res = await fetch('/api/doctor-prep', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ symptomTranscript: transcript }),
      });
      const data = await res.json();
      if (!res.ok) throw new Error(data.error || 'Generation failed');
      setPrep(data.prep);
      setMeta({ patientName: data.patientName, reportCount: data.reportCount, withSymptoms: !!transcript });
    } catch (e: any) {
      toast({ title: 'Could not generate prep sheet', description: e.message, variant: 'destructive' });
```

```
    } finally {
      setLoading(false);
    }
  };

  const download = () => {
    if (!prep) return;
    const html = `<!DOCTYPE html><html><head><meta charset="utf-8"><title>VitalLens Visit Prep</title>
<style>
body{font-family:Georgia,serif;max-width:720px;margin:40px auto;padding:0 24px;color:#1a2b2b;line-height:1.6}
h1{color:#0f766e;border-bottom:2px solid #0f766e;padding-bottom:8px;font-size:22px}
h2{color:#0f766e;font-size:16px;margin-top:24px}
h3{color:#334155;font-size:14px;margin-top:18px}
table{border-collapse:collapse;width:100%;font-size:13px}
th,td{border:1px solid #cbd5d5;padding:6px 10px;text-align:left}
th{background:#f0fdfa}
ul{padding-left:20px}
.footer{margin-top:40px;font-size:11px;color:#888;border-top:1px solid #ddd;padding-top:8px}
</style></head><body>
<h1>🩺 VitalLens — Visit Prep Sheet</h1>
${markdownToHtml(prep)}
<p class="footer">Generated by VitalLens · Educational tool — not a medical record. Verify all values against off
icial lab reports.</p>
</body></html>`;
    const blob = new Blob([html], { type: 'text/html' });
    const url = URL.createObjectURL(blob);
    const a = document.createElement('a');
    a.href = url;
    a.download = `vitallens-visit-prep-${new Date().toISOString().slice(0, 10)}.html`;
    a.click();
    URL.revokeObjectURL(url);
    toast({ title: 'Prep sheet downloaded' });
  };

  return (
    <div className="mx-auto max-w-3xl space-y-5">
      <Card className="overflow-hidden">
        <CardContent className="relative p-6 sm:p-8">
          <div className="pointer-events-none absolute -right-10 -top-10 h-40 w-40 rounded-full bg-teal-100/60 bl
ur-2xl" />
          <div className="flex flex-col items-start gap-4 sm:flex-row sm:items-center">
            <div className="rounded-2xl bg-gradient-to-br from-teal-500 to-emerald-600 p-4 shadow-lg shadow-teal-
600/20">
              <ClipboardList className="h-8 w-8 text-white" />
            </div>
            <div className="flex-1">
              <h3 className="text-xl font-semibold">Walk in prepared, walk out with answers</h3>
              <p className="mt-1 text-sm text-muted-foreground">
                VitalLens combines your lab history{symptomTranscript ? ', symptom conversation' : ''} into a one
-page
                visit prep sheet: what to say, what to show, and exactly what to ask.
              </p>
            </div>
          </div>
          <div className="mt-6 flex flex-wrap gap-3">
            <Button onClick={generate} disabled={loading} className="gap-2 bg-teal-700 hover:bg-teal-800">
              {loading ? <Loader2 className="h-4 w-4 animate-spin" /> : <Sparkles className="h-4 w-4" />}
              {loading ? 'Building your sheet…' : 'Generate visit prep sheet'}
            </Button>
            {prep && (
              <>
                <Button variant="outline" onClick={() => window.print()} className="gap-2"><Printer className="h-
4 w-4" /> Print</Button>
                <Button variant="outline" onClick={download} className="gap-2"><FileDown className="h-4 w-4" /> D
ownload</Button>
              </>
            )}
          </div>
          {meta && !loading && (
            <p className="mt-3 flex items-center gap-1.5 text-xs text-muted-foreground">
              <CalendarCheck className="h-3.5 w-3.5 text-teal-700" />
              Built from {meta.reportCount} lab report{meta.reportCount > 1 ? 's' : ''}{meta.withSymptoms ? ' + y
our symptom session' : ''} for {meta.patientName}.
            </p>
          )}
        </CardContent>
      </Card>

      {loading && (
        <Card>
          <CardContent className="space-y-3 p-6">
```

```
            <div className="flex items-center gap-2 text-sm text-muted-foreground"><Loader2 className="h-4 w-4 an
imate-spin text-teal-700" /> Dr. V is compiling your history…</div>
            <div className="h-3 w-3/4 animate-pulse rounded bg-muted" />
            <div className="h-3 w-full animate-pulse rounded bg-muted" />
            <div className="h-3 w-5/6 animate-pulse rounded bg-muted" />
            <div className="h-3 w-2/3 animate-pulse rounded bg-muted" />
          </CardContent>
        </Card>
      )}

      {prep && !loading && (
        <Card className="print:border-0 print:shadow-none">
          <CardHeader className="pb-3">
            <CardTitle className="text-base">Your visit prep sheet</CardTitle>
          </CardHeader>
          <CardContent>
            <Markdown content={prep} />
            <Alert className="mt-5 border-teal-200 bg-teal-50/50 print:hidden">
              <AlertTitle className="text-sm text-teal-900">Before your visit</AlertTitle>
              <AlertDescription className="text-xs text-muted-foreground">
                Bring your original lab reports too. This sheet summarizes values but the official documents rema
in the source of truth.
              </AlertDescription>
            </Alert>
          </CardContent>
        </Card>
      )}
    </div>
  );
}

// 简易 markdown → html(用于下载)
function markdownToHtml(md: string): string {
  const esc = (s: string) => s.replace(/&/g, '&amp;').replace(/</g, '&lt;').replace(/>/g, '&gt;');
  const lines = md.split('\n');
  let html = '';
  let inList = false;
  let inTable = false;
  for (let i = 0; i < lines.length; i++) {
    const line = lines[i];
    if (line.startsWith('### ')) { if (inList) { html += '</ul>'; inList = false; } if (inTable) { html += '</tab
le>'; inTable = false; } html += `<h3>${inline(esc(line.slice(4)))}</h3>`; continue; }
    if (line.startsWith('## ')) { if (inList) { html += '</ul>'; inList = false; } if (inTable) { html += '</tabl
e>'; inTable = false; } html += `<h2>${inline(esc(line.slice(3)))}</h2>`; continue; }
    if (line.startsWith('| ')) {
      const cells = line.split('|').slice(1, -1).map((c) => c.trim());
      if (cells.every((c) => /^-+$/.test(c.replace(/\s/g, '')) || c === '')) continue;
      if (!inTable) { if (inList) { html += '</ul>'; inList = false; } html += '<table>'; inTable = true; html +=
'<tr>' + cells.map((c) => `<th>${inline(esc(c))}</th>`).join('') + '</tr>'; continue; }
      html += '<tr>' + cells.map((c) => `<td>${inline(esc(c))}</td>`).join('') + '</tr>';
      continue;
    } else if (inTable) { html += '</table>'; inTable = false; }
    if (/^[-*•] /.test(line)) { if (!inList) { html += '<ul>'; inList = true; } html += `<li>${inline(esc(line.sl
ice(2)))}</li>`; continue; }
    if (/^\d+\. /.test(line)) { if (!inList) { html += '<ul>'; inList = true; } html += `<li>${inline(esc(line.re
place(/^\d+\. /, '')))}</li>`; continue; }
    if (inList && line.trim() === '') { html += '</ul>'; inList = false; continue; }
    if (line.trim() !== '') { if (inList) { html += '</ul>'; inList = false; } html += `<p>${inline(esc(line))}</
p>`; }
  }
  if (inList) html += '</ul>';
  if (inTable) html += '</table>';
  return html;
}
function inline(s: string): string {
  return s.replace(/\*\*(.+?)\*\*/g, '<strong>$1</strong>').replace(/\*(.+?)\*/g, '<em>$1</em>');
}
```

src/components/vitallens/markdown.tsx — Markdown renderer with GFM tables — 41 lines

```
'use client';

import ReactMarkdown from 'react-markdown';
import remarkGfm from 'remark-gfm';
import { cn } from '@/lib/utils';
```

```
export function Markdown({ content, className }: { content: string; className?: string }) {
  return (
    <div className={cn('vl-markdown text-[15px] leading-relaxed text-foreground/90', className)}>
      <ReactMarkdown
        remarkPlugins={[remarkGfm]}
        components={{
          h2: ({ children }) => <h3 className="mt-5 mb-2 text-[15px] font-semibold text-foreground flex items-cen
ter gap-2 first:mt-0">{children}</h3>,
          h3: ({ children }) => <h4 className="mt-4 mb-2 text-sm font-semibold text-foreground">{children}</h4>,
          p: ({ children }) => <p className="mb-3 last:mb-0">{children}</p>,
          ul: ({ children }) => <ul className="mb-3 ml-4 list-disc space-y-1.5 marker:text-teal-600">{children}</
ul>,
          ol: ({ children }) => <ol className="mb-3 ml-4 list-decimal space-y-1.5 marker:text-teal-600 marker:fon
t-semibold">{children}</ol>,
          li: ({ children }) => <li className="pl-1">{children}</li>,
          strong: ({ children }) => <strong className="font-semibold text-foreground">{children}</strong>,
          a: ({ children, href }) => (
            <a href={href} className="text-teal-700 underline underline-offset-2 hover:text-teal-900" target="_bl
ank" rel="noreferrer">
              {children}
            </a>
          ),
          code: ({ children }) => <code className="rounded bg-muted px-1.5 py-0.5 font-mono text-[13px]">{childre
n}</code>,
          blockquote: ({ children }) => <blockquote className="border-l-2 border-teal-500 pl-3 text-muted-foregro
und italic">{children}</blockquote>,
          table: ({ children }) => (
            <div className="mb-3 overflow-x-auto rounded-lg border">
              <table className="w-full text-sm">{children}</table>
            </div>
          ),
          th: ({ children }) => <th className="border-b bg-muted/60 px-3 py-2 text-left font-semibold">{children}
</th>,
          td: ({ children }) => <td className="border-b px-3 py-2 align-top">{children}</td>,
          hr: () => <hr className="my-4 border-border" />,
        }}
      >
        {content}
      </ReactMarkdown>
    </div>
  );
}
```

src/components/vitallens/sections.tsx — How-it-works, safety principles & footer — 117 lines

```
'use client';

import { motion } from 'framer-motion';
import { Card, CardContent } from '@/components/ui/card';
import { Camera, LineChart, HeartPulse, ClipboardList, ShieldCheck, Lock, Stethoscope, AlertTriangle } from 'luci
de-react';

export function HowItWorks() {
  const steps = [
    {
      icon: <Camera className="h-6 w-6" />,
      title: '1 · Snap or upload',
      desc: 'Photograph any blood test report — CBC, metabolic panel, lipids, thyroid. Vision AI reads every mark
er and its reference range, just like a lab technician would.',
    },
    {
      icon: <LineChart className="h-6 w-6" />,
      title: '2 · Track over time',
      desc: 'Each report joins your health timeline. VitalLens watches how markers move between dates and surface
s changes that a single report can miss.',
    },
    {
      icon: <HeartPulse className="h-6 w-6" />,
      title: '3 · Understand & act',
      desc: 'Dr. V explains findings in plain language, Nurse V guides symptom triage across three care levels, a
nd Doctor Prep turns it all into a one-page visit sheet.',
    },
  ];
  return (
    <section id="how" className="mx-auto max-w-6xl px-4 sm:px-6">
      <div className="mb-10 text-center">
```

```
        <h2 className="text-3xl font-bold tracking-tight sm:text-4xl">How VitalLens works</h2>
        <p className="mx-auto mt-3 max-w-2xl text-muted-foreground">
          Three steps from a paper report to genuine understanding — powered by vision AI and a clinical language model working together.
        </p>
      </div>
      <div className="grid gap-6 md:grid-cols-3">
        {steps.map((s, i) => (
          <motion.div
            key={s.title}
            initial={{ opacity: 0, y: 24 }}
            whileInView={{ opacity: 1, y: 0 }}
            viewport={{ once: true, margin: '-60px' }}
            transition={{ delay: i * 0.12, duration: 0.5 }}
          >
            <Card className="h-full border-t-4 border-t-teal-600/70 transition-shadow hover:shadow-lg">
              <CardContent className="p-6">
                <div className="mb-4 w-fit rounded-xl bg-teal-600/10 p-3 text-teal-700">{s.icon}</div>
                <h3 className="mb-2 text-lg font-semibold">{s.title}</h3>
                <p className="text-sm leading-relaxed text-muted-foreground">{s.desc}</p>
              </CardContent>
            </Card>
          </motion.div>
        ))}
      </div>
    </section>
  );
}

export function Safety() {
  const items = [
    {
      icon: <ShieldCheck className="h-5 w-5" />,
      title: 'Never a diagnosis',
      desc: 'VitalLens explains and educates; it never names a condition as your diagnosis. Every briefing points you back to a healthcare professional.',
    },
    {
      icon: <AlertTriangle className="h-5 w-5" />,
      title: 'Safety-first red flags',
      desc: 'A deterministic safety layer (not just AI) checks every report for critical values and symptom conversations for emergency signs — escalation always wins.',
    },
    {
      icon: <Lock className="h-5 w-5" />,
      title: 'Private by design',
      desc: 'Your data stays in your own session database. No accounts, no sharing with insurers or employers, and you can delete any report in one click.',
    },
    {
      icon: <Stethoscope className="h-5 w-5" />,
      title: 'Honest about limits',
      desc: 'Reference ranges shown are adult defaults; your report\u2019s printed ranges always take priority. VitalLens is best used alongside — never instead of — your doctor.',
    },
  ];
  return (
    <section id="safety" className="mx-auto max-w-6xl px-4 sm:px-6">
      <div className="rounded-3xl bg-gradient-to-br from-teal-900 via-teal-800 to-emerald-900 p-8 sm:p-12">
        <div className="mb-8 flex flex-col items-start gap-3">
          <span className="rounded-full bg-white/10 px-3 py-1 text-xs font-medium text-teal-100">Our safety principles</span>
          <h2 className="text-2xl font-bold text-white sm:text-3xl">Built like a good nurse: honest, careful, on your side</h2>
          <p className="max-w-2xl text-sm leading-relaxed text-teal-100/80">
            Health tools earn trust by being careful, not by pretending certainty. Here is exactly how VitalLens keeps you safe while making health data understandable.
          </p>
        </div>
        <div className="grid gap-5 sm:grid-cols-2">
          {items.map((it) => (
            <div key={it.title} className="rounded-2xl border border-white/10 bg-white/5 p-5 backdrop-blur transition-colors hover:bg-white/10">
              <div className="mb-3 flex items-center gap-2.5">
                <div className="rounded-lg bg-teal-400/20 p-2 text-teal-200">{it.icon}</div>
                <h3 className="font-semibold text-white">{it.title}</h3>
              </div>
              <p className="text-[13px] leading-relaxed text-teal-100/75">{it.desc}</p>
            </div>
          ))}
        </div>
```

```
        <p className="mt-8 border-t border-white/10 pt-5 text-xs leading-relaxed text-teal-200/60">
          VitalLens is an educational tool and does not provide medical advice, diagnosis, or treatment. Always s
eek the advice of a physician or other qualified health provider with any questions regarding a medical conditio
n. Never disregard professional medical advice or delay seeking it because of something you have read here. If yo
u think you may have a medical emergency, call your doctor or emergency services immediately.
        </p>
      </div>
    </section>
  );
}

export function SiteFooter() {
  return (
    <footer className="mt-auto border-t bg-muted/30">
      <div className="mx-auto flex max-w-6xl flex-col items-center justify-between gap-3 px-4 py-6 text-xs text-m
uted-foreground sm:flex-row sm:px-6">
        <p>© 2026 VitalLens · Built for the UnivaBio Hackathon — AI for Human Health</p>
        <p>Made with vision AI, clinical language models & care</p>
      </div>
    </footer>
  );
}
```

src/app/api/lab-reports/route.ts

API: upload & analyze lab report (VLM + LLM pipeline), list, delete

103 lines

```
import { NextRequest, NextResponse } from 'next/server';
import { db } from '@/lib/db';
import { extractLabReport, analyzeLabReport } from '@/lib/ai';
import { LabMarker, RedFlag } from '@/lib/medical';

// GET /api/lab-reports — 列出所有报告
export async function GET() {
  try {
    const reports = await db.labReport.findMany({ orderBy: { reportDate: 'desc' } });
    return NextResponse.json({
      reports: reports.map((r) => ({
        id: r.id,
        title: r.title,
        patientName: r.patientName,
        reportDate: r.reportDate,
        labName: r.labName,
        markers: JSON.parse(r.markers) as LabMarker[],
        summary: r.summary,
        redFlags: r.redFlags ? (JSON.parse(r.redFlags) as RedFlag[]) : [],
        createdAt: r.createdAt,
      })),
    });
  } catch (e) {
    console.error('GET lab-reports error:', e);
    return NextResponse.json({ error: 'Failed to load reports' }, { status: 500 });
  }
}

// POST /api/lab-reports — 上传图片 → VLM 提取 → LLM 解读 → 保存
export async function POST(req: NextRequest) {
  try {
    const body = await req.json();
    const image = body.image as string; // data URL
    if (!image || !image.startsWith('data:image/')) {
      return NextResponse.json({ error: 'Please provide a valid lab report image (data URL).' }, { status: 400
});
    }
    if (image.length > 9 * 1024 * 1024) {
      return NextResponse.json({ error: 'Image too large. Please use an image under 6MB.' }, { status: 400 });
    }

    // 1. VLM 提取
    const extracted = await extractLabReport(image);
    if (!extracted.markers.length) {
      return NextResponse.json({ error: 'No lab markers could be read from this image. Try a clearer, well-lit ph
oto of the report.' }, { status: 422 });
    }

    // 2. 获取历史用于趋势上下文
    const previous = await db.labReport.findMany({
      orderBy: { reportDate: 'desc' },
```

```
      take: 3,
    });

    // 3. LLM 解读
    const { summary, redFlags } = await analyzeLabReport(
      extracted,
      previous.map((p) => ({ date: p.reportDate.toISOString().slice(0, 10), markers: JSON.parse(p.markers) as Lab
Marker[] })),
    );

    // 4. 保存
    const title = `${extracted.labName || 'Lab Report'} — ${extracted.reportDate}`;
    const saved = await db.labReport.create({
      data: {
        title,
        patientName: extracted.patientName,
        reportDate: new Date(extracted.reportDate + 'T12:00:00Z'),
        labName: extracted.labName,
        sourceImage: image.length < 700_000 ? image : null,
        markers: JSON.stringify(extracted.markers),
        summary,
        redFlags: JSON.stringify(redFlags),
      },
    });

    return NextResponse.json({
      report: {
        id: saved.id,
        title: saved.title,
        patientName: saved.patientName,
        reportDate: saved.reportDate,
        labName: saved.labName,
        markers: extracted.markers,
        summary,
        redFlags,
        createdAt: saved.createdAt,
      },
    });
  } catch (e: any) {
    console.error('POST lab-reports error:', e);
    return NextResponse.json({ error: e.message || 'Analysis failed. Please try again.' }, { status: 500 });
  }
}

// DELETE /api/lab-reports?id=xxx
export async function DELETE(req: NextRequest) {
  try {
    const id = req.nextUrl.searchParams.get('id');
    if (!id) return NextResponse.json({ error: 'Missing id' }, { status: 400 });
    await db.labReport.delete({ where: { id } });
    return NextResponse.json({ ok: true });
  } catch (e) {
    return NextResponse.json({ error: 'Failed to delete' }, { status: 500 });
  }
}
```

src/app/api/lab-reports/trends/route.ts

API: cross-report trend computation

17 lines

```
import { NextResponse } from 'next/server';
import { db } from '@/lib/db';
import { computeTrends, LabMarker } from '@/lib/medical';

// GET /api/lab-reports/trends — 跨报告趋势
export async function GET() {
  try {
    const reports = await db.labReport.findMany({ orderBy: { reportDate: 'asc' } });
    const trends = computeTrends(
      reports.map((r) => ({ reportDate: new Date(r.reportDate), markers: JSON.parse(r.markers) as LabMarker[]
})),
    );
    return NextResponse.json({ trends, reportCount: reports.length });
  } catch (e) {
    console.error('trends error:', e);
    return NextResponse.json({ error: 'Failed to compute trends' }, { status: 500 });
```

```
  }
}
```

src/app/api/symptom-checker/route.ts

API: symptom triage conversation
34 lines

```ts
import { NextRequest, NextResponse } from 'next/server';
import ZAI from 'z-ai-web-dev-sdk';
import { TRIAGE_SYSTEM_PROMPT } from '@/lib/ai';

interface ChatMessage {
  role: 'user' | 'assistant';
  content: string;
}

// POST /api/symptom-checker — 症状分诊对话
export async function POST(req: NextRequest) {
  try {
    const body = await req.json();
    const messages: ChatMessage[] = Array.isArray(body.messages) ? body.messages.slice(-20) : [];
    if (!messages.length || messages[messages.length - 1].role !== 'user') {
      return NextResponse.json({ error: 'Invalid conversation' }, { status: 400 });
    }

    const zai = await ZAI.create();
    const response = await zai.chat.completions.create({
      messages: [
        { role: 'system', content: TRIAGE_SYSTEM_PROMPT },
        ...messages.map((m) => ({ role: m.role, content: m.content.slice(0, 4000) })),
      ],
      thinking: { type: 'disabled' },
    });

    const reply = response.choices[0]?.message?.content || "I'm sorry, I couldn't process that. Could you rephras
e?";
    return NextResponse.json({ reply });
  } catch (e: any) {
    console.error('symptom-checker error:', e);
    return NextResponse.json({ error: 'The nurse assistant is unavailable right now. Please try again.' }, { stat
us: 500 });
  }
}
```

src/app/api/doctor-prep/route.ts

API: visit prep sheet generation
33 lines

```ts
import { NextRequest, NextResponse } from 'next/server';
import { db } from '@/lib/db';
import { generateDoctorPrep } from '@/lib/ai';
import { LabMarker } from '@/lib/medical';

// POST /api/doctor-prep — 生成就诊准备清单
export async function POST(req: NextRequest) {
  try {
    const body = await req.json().catch(() => ({}));
    const symptomTranscript: string = typeof body.symptomTranscript === 'string' ? body.symptomTranscript : '';

    const reports = await db.labReport.findMany({ orderBy: { reportDate: 'desc' }, take: 6 });
    if (!reports.length) {
      return NextResponse.json({ error: 'Upload at least one lab report first — your prep sheet is built from you
r real data.' }, { status: 400 });
    }
    const patientName = reports[0].patientName || 'Guest';
    const asc = [...reports].sort((a, b) => new Date(a.reportDate).getTime() - new Date(b.reportDate).getTime());

    const prep = await generateDoctorPrep(
      patientName,
      asc.map((r) => ({
        date: new Date(r.reportDate).toISOString().slice(0, 10),
        markers: JSON.parse(r.markers) as LabMarker[],
      })),
      symptomTranscript,
    );
```

```
    return NextResponse.json({ prep, patientName, reportCount: reports.length });
  } catch (e: any) {
    console.error('doctor-prep error:', e);
    return NextResponse.json({ error: e.message || 'Failed to generate prep sheet' }, { status: 500 });
  }
}
```

src/app/api/demo-seed/route.ts

API: demo timeline seeder (Sarah Chen case study)
160 lines

```
import { NextResponse } from 'next/server';
import { db } from '@/lib/db';
import { LabMarker, RedFlag, evaluateStatus } from '@/lib/medical';

// 演示数据：Sarah Chen 的 3 份化验单，讲述"缺铁性贫血逐渐发展、被趋势分析及早发现"的故事
function mk(name: string, value: number, unit: string, refLow: number, refHigh: number, category: string): LabMar
ker {
  return { name, value, unit, refLow, refHigh, category, status: evaluateStatus(value, refLow, refHigh, name) };
}

const REPORTS: {
  date: string;
  title: string;
  markers: LabMarker[];
  summary: string;
  redFlags: RedFlag[];
}[] = [
  {
    date: '2026-03-12',
    title: 'CityMed Diagnostics — 2026-03-12',
    markers: [
      mk('Hemoglobin', 13.9, 'g/dL', 12.0, 15.5, 'CBC'),
      mk('Hematocrit', 41.2, '%', 36, 46, 'CBC'),
      mk('MCV', 89.4, 'fL', 80, 100, 'CBC'),
      mk('Ferritin', 45, 'ng/mL', 30, 400, 'Iron Studies'),
      mk('Transferrin Saturation', 24, '%', 20, 50, 'Iron Studies'),
      mk('Glucose', 88, 'mg/dL', 70, 99, 'Metabolic'),
      mk('TSH', 2.1, 'mIU/L', 0.4, 4.0, 'Thyroid'),
      mk('Total Cholesterol', 182, 'mg/dL', 0, 200, 'Lipids'),
      mk('LDL', 104, 'mg/dL', 0, 100, 'Lipids'),
      mk('HDL', 62, 'mg/dL', 40, 80, 'Lipids'),
    ],
    summary: `## What this report shows
This is a routine blood panel from March, and overall it looks healthy. All major organ-function markers and bloo
d counts sit comfortably inside their reference ranges.

## Key findings
- **Hemoglobin (13.9 g/dL)** — normal. Your blood is carrying oxygen well.
- **Ferritin (45 ng/mL)** — technically normal, but in the lower third of the range. Ferritin reflects your iron
reserves, so this is worth watching over time.
- **LDL cholesterol (104 mg/dL)** — just above the ideal cutoff; a minor note rather than a concern.

## Trends over time
No previous reports yet — upload past reports to unlock trend analysis.

## What you can do next
- Keep eating iron-rich foods (lean red meat, lentils, spinach) alongside vitamin C, which boosts absorption.
- Recheck iron studies in a few months, especially if you feel unusually tired.
- Discuss your LDL level with a healthcare professional at your next routine visit.

## Questions to ask your doctor
- Is my ferritin level something we should track regularly?
- Based on my cholesterol, do I need any lifestyle changes now?`,
    redFlags: [],
  },
  {
    date: '2026-06-18',
    title: 'CityMed Diagnostics — 2026-06-18',
    markers: [
      mk('Hemoglobin', 12.1, 'g/dL', 12.0, 15.5, 'CBC'),
      mk('Hematocrit', 36.8, '%', 36, 46, 'CBC'),
      mk('MCV', 84.1, 'fL', 80, 100, 'CBC'),
      mk('Ferritin', 26, 'ng/mL', 30, 400, 'Iron Studies'),
      mk('Transferrin Saturation', 17, '%', 20, 50, 'Iron Studies'),
      mk('Glucose', 91, 'mg/dL', 70, 99, 'Metabolic'),
      mk('TSH', 2.3, 'mIU/L', 0.4, 4.0, 'Thyroid'),
      mk('Total Cholesterol', 179, 'mg/dL', 0, 200, 'Lipids'),
```

```
      mk('LDL', 101, 'mg/dL', 0, 100, 'Lipids'),
      mk('HDL', 60, 'mg/dL', 40, 80, 'Lipids'),
    ],
    summary: `## What this report shows
Compared with March, this panel shows early changes worth paying attention to — nothing alarming yet, but a clear direction.

## Key findings
- **Ferritin (26 ng/mL)** — has slipped below the normal floor. Your iron reserves are running low.
- **Transferrin saturation (17%)** — below range, which also points toward low available iron.
- **Hemoglobin (12.1 g/dL)** — still normal, but only just above the lower limit; it has dropped from 13.9 in March.
- **MCV (84.1 fL)** — normal but drifting down; red cells are starting to shrink slightly.

## Trends over time
Since March: hemoglobin fell about 13% (13.9 → 12.1) and ferritin fell about 42% (45 → 26). This pattern — iron stores dropping first, hemoglobin following — is the classic early sequence of iron deficiency. Catching it at this stage is exactly when simple fixes work best.

## What you can do next
- Talk to a healthcare professional about iron supplementation — doses and duration should be decided with them.
- Pair iron-rich meals with vitamin C (citrus, peppers) and avoid tea/coffee right next to iron-rich meals, as they reduce absorption.
- If you have heavy periods, are vegetarian, or donate blood often, mention it — these are common reasons for low iron.
- Recheck in 8–12 weeks.

## Questions to ask your doctor
- Should I start iron supplements, and which type do you recommend?
- Do we need to check for any source of blood loss?`,
    redFlags: [],
  },
  {
    date: '2026-10-05',
    title: 'CityMed Diagnostics — 2026-10-05',
    markers: [
      mk('Hemoglobin', 10.8, 'g/dL', 12.0, 15.5, 'CBC'),
      mk('Hematocrit', 33.5, '%', 36, 46, 'CBC'),
      mk('MCV', 77.8, 'fL', 80, 100, 'CBC'),
      mk('MCH', 25.1, 'pg', 27, 33, 'CBC'),
      mk('Ferritin', 14, 'ng/mL', 30, 400, 'Iron Studies'),
      mk('Transferrin Saturation', 12, '%', 20, 50, 'Iron Studies'),
      mk('Red Blood Cells', 4.42, '10^6/uL', 4.5, 5.9, 'CBC'),
      mk('Glucose', 90, 'mg/dL', 70, 99, 'Metabolic'),
      mk('TSH', 2.0, 'mIU/L', 0.4, 4.0, 'Thyroid'),
      mk('Total Cholesterol', 176, 'mg/dL', 0, 200, 'Lipids'),
      mk('LDL', 99, 'mg/dL', 0, 100, 'Lipids'),
      mk('HDL', 61, 'mg/dL', 40, 80, 'Lipids'),
    ],
    summary: `## What this report shows
This panel confirms that the trend spotted in June has continued. Several red-blood-cell markers are now outside their reference ranges, which is consistent with established iron-deficiency anemia. This is very treatable — the important thing is to act now and find out why it developed.

## Key findings
- **Hemoglobin (10.8 g/dL)** — below range (down from 12.1 in June and 13.9 in March). This level often goes hand-in-hand with tiredness, breathlessness on exertion, or difficulty concentrating.
- **Ferritin (14 ng/mL)** — well below range; iron reserves are nearly depleted.
- **MCV (77.8 fL)** — below range. Red cells are now smaller than normal, a pattern seen when the body makes them without enough iron.
- **MCH (25.1 pg)** — below range; each red cell carries less hemoglobin than usual.
- **Hematocrit (33.5%)** and **RBC count (4.42)** — both just under their floors.

## Trends over time
Over 7 months, hemoglobin fell 22% (13.9 → 12.1 → 10.8) and ferritin fell 69% (45 → 26 → 14). The step-by-step pattern — stores falling first, then cell size, then hemoglobin — is a textbook progression of worsening iron deficiency. The good news: it was caught, it is treatable, and recovery is usually straightforward.

## What you can do next
- Book a doctor's visit soon — this level of anemia deserves proper evaluation and usually prescription-strength iron.
- Ask about testing for the cause (e.g., celiac screening, gastrointestinal evaluation, menstrual factors) — iron deficiency always has a reason.
- Eat iron-rich foods daily, but treat diet as a support, not the fix, at this stage.
- Watch for chest pain, resting breathlessness, fainting, or a racing heart — seek urgent care if any appear.

## Questions to ask your doctor
- What workup do you recommend to find the cause of my iron deficiency?
- What iron regimen do you suggest, and when should we recheck my levels?
- Are there any symptoms that should make me seek care sooner?`,
    redFlags: [],
```

```
  },
];

// POST /api/demo-seed — 载入演示数据
export async function POST() {
  try {
    // 清除旧演示数据
    const existing = await db.labReport.findMany({ where: { patientName: 'Sarah Chen' } });
    for (const e of existing) await db.labReport.delete({ where: { id: e.id } });

    for (const r of REPORTS) {
      await db.labReport.create({
        data: {
          title: r.title,
          patientName: 'Sarah Chen',
          reportDate: new Date(r.date + 'T12:00:00Z'),
          labName: 'CityMed Diagnostics',
          markers: JSON.stringify(r.markers),
          summary: r.summary,
          redFlags: JSON.stringify(r.redFlags),
        },
      });
    }
    return NextResponse.json({ ok: true, count: REPORTS.length });
  } catch (e: any) {
    console.error('demo-seed error:', e);
    return NextResponse.json({ error: e.message || 'Seed failed' }, { status: 500 });
  }
}
```

package.json

Dependencies & project manifest

96 lines

```
{
  "name": "nextjs_tailwind_shadcn_ts",
  "version": "0.2.1",
  "private": true,
  "scripts": {
    "dev": "next dev -p 3000 2>&1 | tee dev.log",
    "build": "next build && cp -r .next/static .next/standalone/.next/ && cp -r public .next/standalone/",
    "start": "NODE_ENV=production bun .next/standalone/server.js 2>&1 | tee server.log",
    "lint": "eslint .",
    "db:push": "prisma db push --accept-data-loss",
    "db:generate": "prisma generate",
    "db:migrate": "prisma migrate dev",
    "db:reset": "prisma migrate reset"
  },
  "dependencies": {
    "@dnd-kit/core": "^6.3.1",
    "@dnd-kit/sortable": "^10.0.0",
    "@dnd-kit/utilities": "^3.2.2",
    "@hookform/resolvers": "^5.1.1",
    "@mdxeditor/editor": "^3.39.1",
    "@prisma/client": "^6.11.1",
    "@radix-ui/react-accordion": "^1.2.11",
    "@radix-ui/react-alert-dialog": "^1.1.14",
    "@radix-ui/react-aspect-ratio": "^1.1.7",
    "@radix-ui/react-avatar": "^1.1.10",
    "@radix-ui/react-checkbox": "^1.3.2",
    "@radix-ui/react-collapsible": "^1.1.11",
    "@radix-ui/react-context-menu": "^2.2.15",
    "@radix-ui/react-dialog": "^1.1.14",
    "@radix-ui/react-dropdown-menu": "^2.1.15",
    "@radix-ui/react-hover-card": "^1.1.14",
    "@radix-ui/react-label": "^2.1.7",
    "@radix-ui/react-menubar": "^1.1.15",
    "@radix-ui/react-navigation-menu": "^1.2.13",
    "@radix-ui/react-popover": "^1.1.14",
    "@radix-ui/react-progress": "^1.1.7",
    "@radix-ui/react-radio-group": "^1.3.7",
    "@radix-ui/react-scroll-area": "^1.2.9",
    "@radix-ui/react-select": "^2.2.5",
    "@radix-ui/react-separator": "^1.1.7",
    "@radix-ui/react-slider": "^1.3.5",
    "@radix-ui/react-slot": "^1.2.3",
    "@radix-ui/react-switch": "^1.2.5",
    "@radix-ui/react-tabs": "^1.1.12",
```

```
    "@radix-ui/react-toast": "^1.2.14",
    "@radix-ui/react-toggle": "^1.1.9",
    "@radix-ui/react-toggle-group": "^1.1.10",
    "@radix-ui/react-tooltip": "^1.2.7",
    "@reactuses/core": "^6.0.5",
    "@tanstack/react-query": "^5.82.0",
    "@tanstack/react-table": "^8.21.3",
    "class-variance-authority": "^0.7.1",
    "clsx": "^2.1.1",
    "cmdk": "^1.1.1",
    "date-fns": "^4.1.0",
    "embla-carousel-react": "^8.6.0",
    "framer-motion": "^12.23.2",
    "input-otp": "^1.4.2",
    "lucide-react": "^0.525.0",
    "next": "^16.1.1",
    "next-auth": "^4.24.11",
    "next-intl": "^4.3.4",
    "next-themes": "^0.4.6",
    "pagedjs": "^0.4.3",
    "prisma": "^6.11.1",
    "react": "^19.0.0",
    "react-day-picker": "^9.8.0",
    "react-dom": "^19.0.0",
    "react-hook-form": "^7.60.0",
    "react-markdown": "^10.1.0",
    "react-resizable-panels": "^3.0.3",
    "react-syntax-highlighter": "^15.6.1",
    "recharts": "^2.15.4",
    "remark-gfm": "^4.0.1",
    "sharp": "^0.34.3",
    "sonner": "^2.0.6",
    "tailwind-merge": "^3.3.1",
    "tailwindcss-animate": "^1.0.7",
    "uuid": "^11.1.0",
    "vaul": "^1.1.2",
    "z-ai-web-dev-sdk": "^0.0.18",
    "zod": "^4.0.2",
    "zustand": "^5.0.6"
  },
  "devDependencies": {
    "@tailwindcss/postcss": "^4",
    "@types/react": "^19",
    "@types/react-dom": "^19",
    "bun-types": "^1.3.4",
    "eslint": "^9",
    "eslint-config-next": "^16.1.1",
    "tailwindcss": "^4",
    "tw-animate-css": "^1.3.5",
    "typescript": "^5"
  }
}
```