NutriScope v2 — Roadmap & Milestones

Goals

- Lightweight, deterministic-first backend (age + sex ranges).
- Clean upload → analyze → save to Reports flow.
- Minimal LLM usage (summary only), low latency/cost.
- Clear disclaimers; no ambiguous flags.
- Foundation for growth (pregnancy/conditions later) and measurable quality improvements.

Phase 0 — Prep

- Create `docs/` folder with: `V2-Roadmap.md`, `Architecture.md`, `API.md`.
- Open a `v2` branch; set code owners; define coding style (Black/ruff for Python, Prettier/ESLint for FE).
- Add `.env.example` (Groq key optional).

Phase 1 — Backend Core (FastAPI)

File tree

```
backend/
  api/routes.py
  core/units.py
  core/resolver.py
  core/evaluator.py
  ingest/parser.py
  kb/loader.py
  summarize/llm.py   # optional
  models/schemas.py
```

Milestones

- `tests_kb_v2.json` (basic CBC/BMP/LFT/Lipid/TSH/VitD/B12/Iron/Ferritin/eGFR).
- `kb/loader.py`: load KB to dict by `test_name`.
- `core/units.py`: unit map + conversions; canonical units per test.
- `core/resolver.py`: deterministic range selection (age+sex, most-specific match).
- `core/evaluator.py`: normalize units → compare with ±2–3% tolerance → `normal / borderline_(low|high) / low / high`.
- `ingest/parser.py`: native PDF text via PyMuPDF; fallback OCR stub (wire later).
- `api/routes.py`:
    - `POST /analyze` — input: `{age, sex, reportName, file}` → output structured results + (optional) summary + disclaimer.
    - `GET /kb/tests` — list supported tests.
    - `GET /reports`, `GET /reports/{id}`, `DELETE /reports/{id}` — simple in-memory/SQLite metadata store.
- `summarize/llm.py` (optional): Groq call that formats plain-English summary from structured results; skip when everything is normal.

Acceptance criteria

- Applies correct age/sex range per test.
- Prefers lab-reported reference ranges when present.
- Returns status + applied range source (“LAB” or “KB”) per row.
- Handles unknown units safely (`needs_review`), no diet advice in that case.

Phase 2 — Frontend (React + TS + Vite)

Views

- Upload
- Reports List
- Report Detail

Milestones

- **Upload form** fields: `Report Name`, `Age`, `Sex (Male/Female)`, `File`.
- **Disclaimers**
  - Compact: below upload button.
  - Longer: above summary on Report Detail.
- **Analyze flow**: on submit → call `/analyze` → save metadata → redirect to Report Detail.
- **Reports list**: Name | Uploaded | Age/Sex | Status | Actions (View/Re-run/Delete).
- **Report detail**:
  - Context badge (Name • Age • Sex)

    - Results table: Test | Value | Unit | Applied Range | Status chip
    - “Why?” tooltip: shows range source + rule matched.
    - Summary (if present) and disclaimer block.

Acceptance criteria

- Reproducible UX; no blocking on LLM for normal cases.
- Clear status chips and applied range visibility.
- File retained and accessible via metadata link (local or object store).

Phase 3 — Quality & Safety Rails

Milestones

- **Plausibility bounds** per test (to catch OCR glitches).
- **OCR confidence gate** (banner if low; suppress diet advice when uncertain).
- **Unit normalization** with safe fallbacks.
- **Caching**: content hash of (file, age, sex) → reuse analysis.

Acceptance criteria

- No “wild” flags due to OCR noise.
- Borderline values labeled accurately (not hard abnormal).
- Repeat analyses consistent for same inputs.

Phase 4 — Monitoring & Improvement Tracking (Lightweight)

Backend

- Log per analysis: {latency_ms, token_cost (if used), ocr_confidence, #flags}.
- Optional SQLite tables: `reports`, `analyses`, `metrics`.

Admin (optional)

- Minimal “Quality Dashboard”: p50/p95 latency, % reports with OCR-warning, flag distribution, LLM usage rate.

Acceptance criteria

- Can demonstrate improvement trends (accuracy proxies, latency, cost) over time.

Research Paper Workstream (in parallel, light)

Sections to draft early

- Problem & Motivation
- Method (KB schema, resolver, evaluator, OCR, optional LLM)
- Evaluation protocol (flagging accuracy vs. lab ranges; OCR extraction accuracy; latency/cost)
- Related Work (with “similarities vs differences”)
- Limitations & Ethics (not medical advice; lab variation; privacy)
- Future Work (pregnancy, chronic conditions, fasting, personalization; improvement monitoring)

Artifacts

- Dataset sheet (anonymized reports, #tests, sources).
- Repro scripts (run analysis batch, compute precision/recall against lab ranges).
- Result tables and small plots.

Checklists

API Contract (final)

- `POST /analyze` returns:


```
{
  "context": {"age": 34, "sex": "male", "report_name": "CBC"},
  "results": [
    {
      "test": "Hemoglobin (Hgb)",
      "value": 12.8, "unit": "g/dL",
      "applied_range": {"low": 13.0, "high": 18.0, "source": "KB", "note": "male ≥18y"},
      "status": "borderline_low"
    }
  ],
  "summary_text": "optional",
  "disclaimer": "…",
  "meta": {"ocr_confidence": 0.92, "analyzer_version": "v2.0.0"}
}
```


KB Commit Rules

- Every test has `canonical_unit`.
- Ranges cover “any” + male/female adults; teen entries optional.
- Advice only shown when flagged (not for `normal` or `needs_review`).

UI Copy (ready)

- Compact disclaimer (Upload):

  ⚠️ NutriScope offers educational guidance and is not a substitute for professional medical advice. Reference ranges may differ slightly from your lab's report.

- Longer disclaimer (Summary):

  ⚠️ Important: NutriScope is an AI-powered tool designed to help you understand your lab reports. While we use standard reference ranges for children, adults, and elderly patients, these values may differ slightly from your testing laboratory's ranges. Information and diet suggestions are educational only and should not replace consultation with a qualified healthcare professional.

Risks & Mitigations

- **OCR variability:** start with native PDFs; add OCR fallback later; confidence gating + plausibility bounds.
- **Unit mismatches:** strict unit map; refuse to flag when uncertain.
- **User trust:** always show applied range + source; clear disclaimers.
- **Scope creep:** feature flags for future contexts (pregnancy/diabetes/CKD).

Done = ✅ (Exit Criteria)

- Backend returns consistent, correct flags with applied ranges for the basic test set.
- Frontend supports Upload → Analyze → Saved to Reports → View Detail.
- Disclaimers visible in both places; advice shown only when flagged.
- Monitoring captures latency, OCR confidence, and LLM usage.
- Draft paper sections exist with initial results from a small anonymized set.