

HEALTH ID · OPESCARE.COM

The Record Is the Product

OpesCare's Patient-Owned Health ID

Jude Nshome on how OpesCare turns a patient's medical record into portable, consent-governed infrastructure for Cameroon — and where its limits are.



BY

Jude Nshome

Published 19 October 2026

CONTENTS

- 01** The spine: a portable Health ID
- 02** The marketplace layer: finding care where it actually is
- 03** The governance layer: consent as a first-class protocol
- 04** The family layer: acting for dependents
- 05** One app, five clinics
- 06** Under the hood (for engineers)
- 07** Where the honest limits are
- 08** Why this matters beyond Cameroon
 - Frequently asked questions
 - Sources
 - About the author

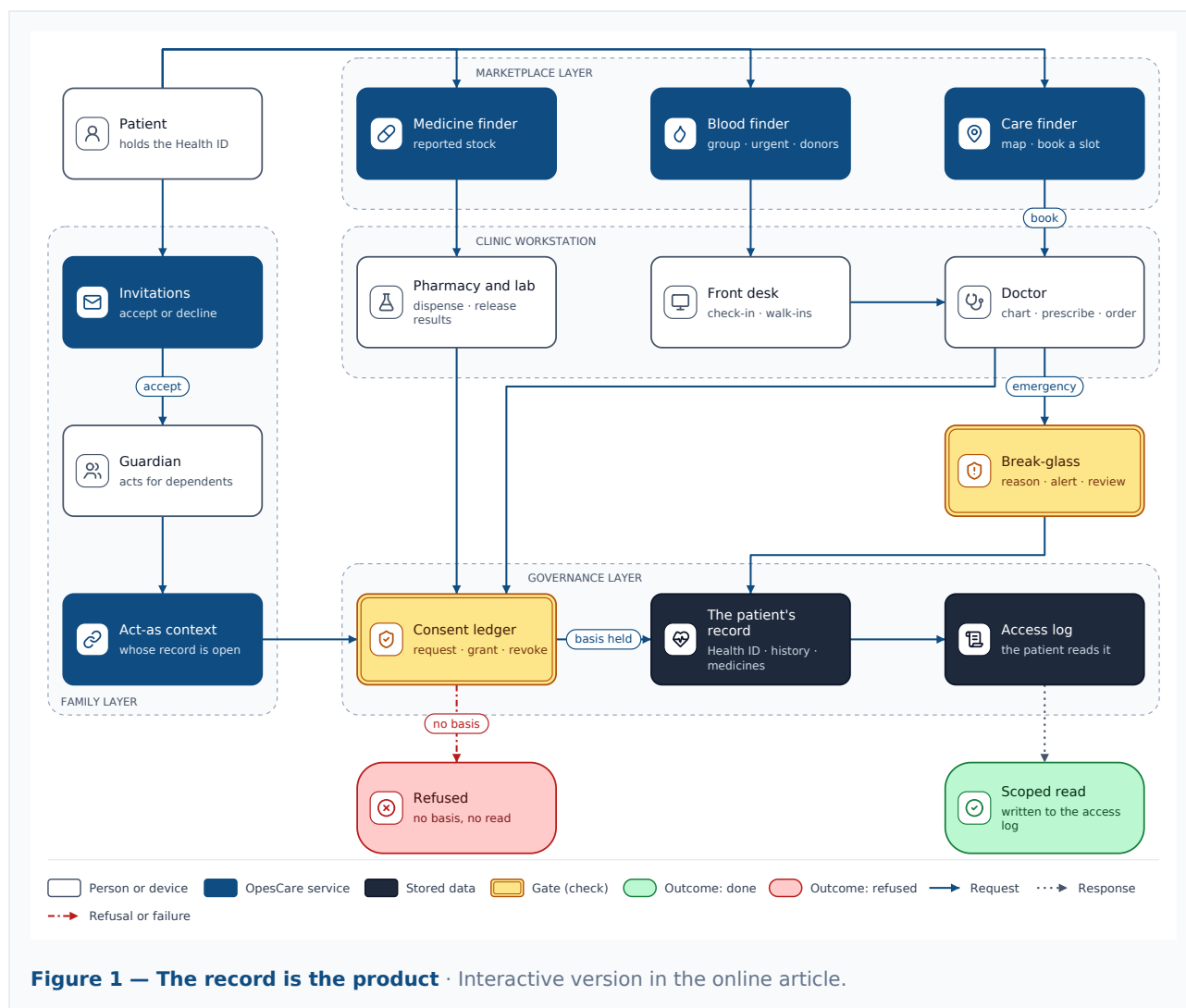
KEY TAKEAWAYS

- OpesCare treats the patient's record — identity, history, medicines, allergies, labs, consent settings and access trail — as the product itself, not a by-product of a visit.
- Cameroon's first Health ID is a verifiable credential with a QR code; a new clinic can verify it and request access, but cannot read the record without a care relationship or the patient's consent.
- Consent is a protocol: requests, a consent ledger, revocation, a patient-visible access log, correction requests and an audited break-glass path for emergencies.
- Guardians act for children and dependents through an explicit act-as context; adults are only added after they accept an invitation.
- The same app serves the patient, the doctor, the front desk, the pharmacist and the lab technician, each working on the one record the patient owns.
- The limits are stated plainly: mobile-money payments wait on merchant accounts, push alerts wait on store credentials, TLS pinning is Android-only, and care is never gated on payment.

Not a health-transaction app: a health-relationship app. The record travels between patient, family, clinic and caregiver — and every handoff is consented, logged and revocable.

In most health systems in sub-Saharan Africa — and, candidly, in most health systems everywhere — the medical record belongs to the institution that created it. Test results sit in a file at one hospital. A prescription history lives with one pharmacy. When a patient moves, migrates, books elsewhere or loses their paper booklet, the record does not come with them. The defaults are set the wrong way round: the institution is durable and the patient is friction.

OpesCare inverts that. It is a bilingual (English and French) health platform for Cameroon in which the patient's record — identity, history, medicines, blood group, allergies, laboratory results, consent settings and access trail — is *the product itself*. Around that record sits a marketplace layer for locating care in the real world: medicine stock reported by nearby pharmacies, blood availability, clinicians and the national facility directory. And the same app, on the same phone, is also the clinic's workstation: doctor, front desk, pharmacist and laboratory technician each get their own experience, each working on the record the patient owns.



This article walks through what that means, feature by feature: what a patient can do on day one, what the staff behind the counter can do, and the governance machinery that makes the whole thing trustworthy enough to hold medical data.

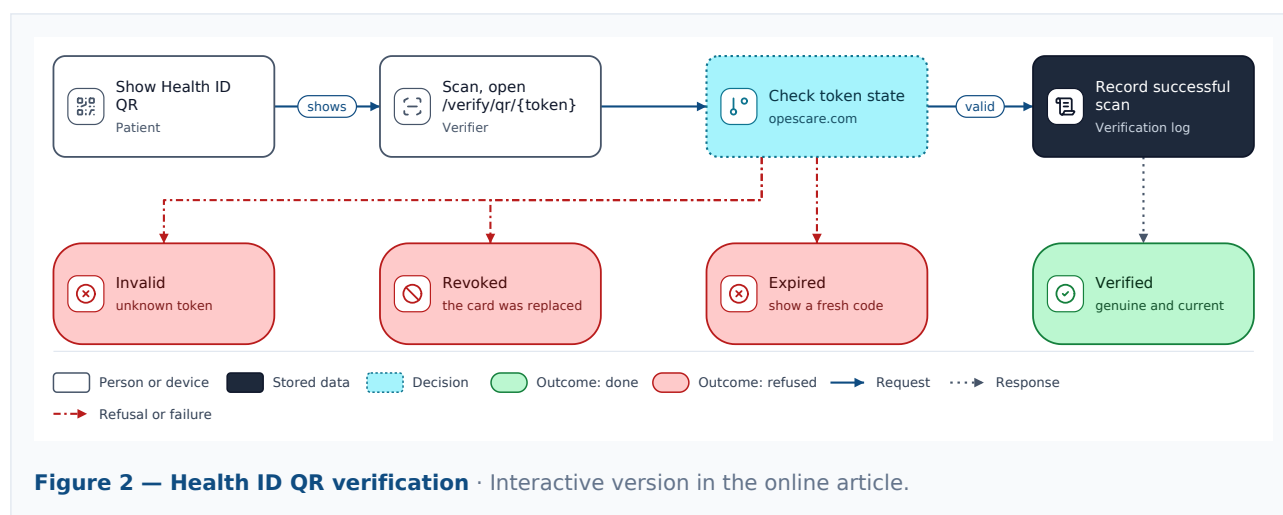
1. The spine: a portable Health ID

Every patient on OpesCare receives **Cameroon's first Health ID** — a verifiable credential, presented like the document it is, with a QR code and a verification link a clinician can scan at the point of care. It extends into a **Health Passport**: a long-form, living presentation of the record (identity and medical summary, medicines, allergies and alerts, encounters, procedures, vaccinations, maternal and child care) laid out like a passport rather than an

administrative dashboard, because that is the format people in rural and low-resource settings still instinctively trust: a document you hold. A page with no records stays empty; nothing is invented to fill it.

Two design decisions matter more than they might appear to.

1. **The ID works across institutions.** The patient shows the card at a new clinic, and that clinic can verify it and request access. No patient is tied to "their" clinic.
2. **The card is built to be shown safely.** It has a dedicated presentation mode; screens that carry health data block screenshots and screen recording; and the app can be locked behind the phone's biometrics, so a handset left at a phone-charging stall cannot open straight onto someone's HIV status or pregnancy history.



The sceptic's question is obvious, and it is the right one: in Cameroon a phone is often shared, and the strongest encryption is worth nothing if whoever picks up the handset inherits everything the owner's session could see. OpesCare treats that shared-phone reality as a design constraint, not an edge case. Tokens live in the operating system's secure storage; signing out or an expired session wipes everything — cached queries, the offline cache, the pending outbox and any exported PDFs — and privacy protection is applied on every screen that renders patient data.

2. The marketplace layer: finding care where it actually is

A directory that only lists facilities is a phone book. OpesCare treats health supply as *inventory that exists right now*, and exposes it through three services.

Medicine finder. Patients search for a medicine, see stock reported by pharmacies on the network, sorted by distance, and can reserve a quantity at the pharmacy that actually has it. The reservation lands in that pharmacy's desk queue, where the pharmacist confirms and dispenses it. Availability comes only from live stock records whose source is on an allow-list: if no pharmacy has reported the medicine, the search says so rather than showing placeholder data.

Blood finder. Blood is the most perishable inventory a health system manages. OpesCare shows availability by blood group and product on a map, supports **urgent requests** and follows each request through to its outcome. Then it does something a supply tool rarely does: it invites the patient to **become a donor**, and includes a donor directory and donor education — because more donors is the only strategy that has ever solved a blood shortage at scale.

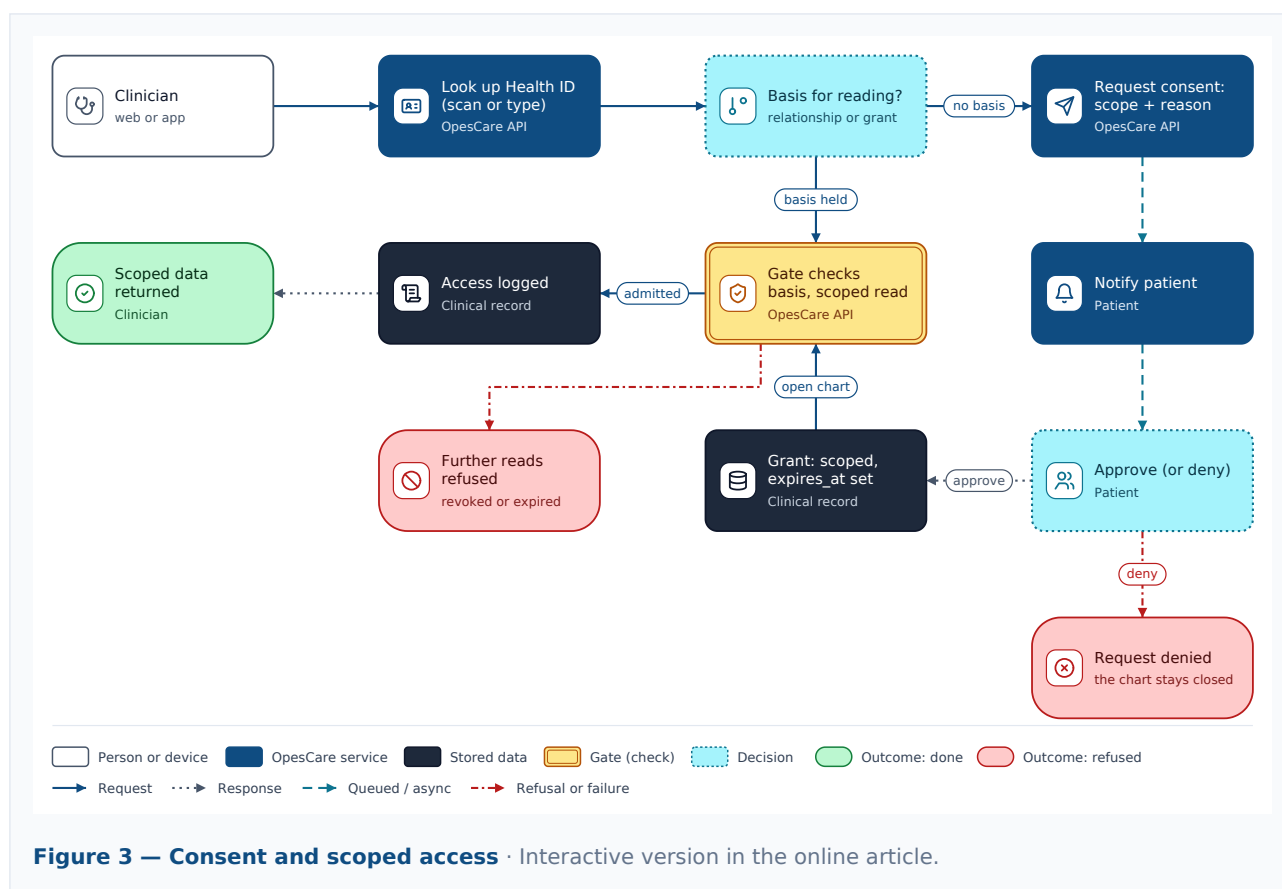
Care finder. Clinicians, facilities and the national directory share one map-driven surface: you find a clinician who is available and near you, book the slot and walk in. The map is not decoration; it shows a country's networked care along a different axis from appointments — geography rather than calendars.

This is a marketplace because it only works when every side of the transaction is instrumented, not just the search box. Phone-book listings fail precisely when stock, urgency and trust do not line up.

3. The governance layer: consent as a first-class protocol

This is where OpesCare stops looking like a well-made app and starts looking like health infrastructure.

A clinician who wants to see a record that has not been released to them does not get a bypass; they make a **request**. The patient's phone shows it on a consent sheet — who is asking, why, which data and for how long — and the patient approves or declines. Grants are kept in a **consent ledger**. Every access to the record is written to an **access log the patient can read**: the person who owns the data can see who opened it and when. Consent can be **revoked**, which is what makes "consent" a word rather than a checkbox. And a **correction request** acknowledges that records contain errors: the patient files the request from the app and it is reviewed at OpesCare's patient-rights desk.



There is an emergency path too. **Break-glass access** for a patient who is unconscious and cannot consent requires a stated reason, opens only a limited emergency profile, alerts the patient and is reviewed afterwards — because a governance system with no emergency access is a governance system that costs lives.

On export, the record has two doors: **PDF** for people and **FHIR R4** for systems, so a secondary provider receives a standards-based format rather than a screenshot. Records can

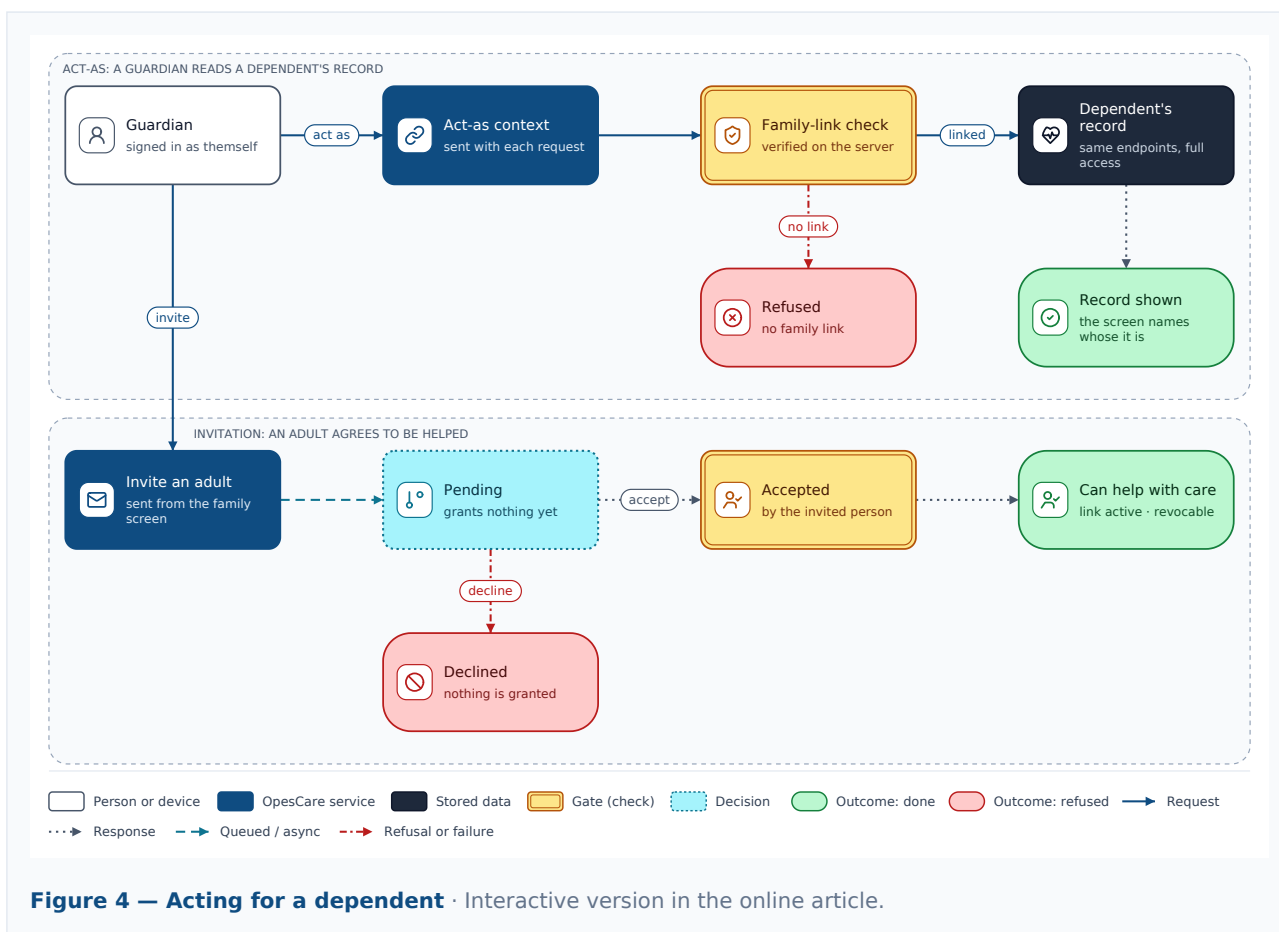
also be **imported** — a FHIR bundle or scanned documents — with an honesty rarely seen in health apps: imported records are stored and *labelled as patient-provided and unverified*. Provenance matters too much to launder.

The same ethos runs throughout: the audit trail is **aimed at the patient**, not at a server log nobody reads. In most electronic records, "who accessed this chart" lives in a compliance database. In OpesCare, it is a screen the patient can question. When governance is legible to the governed, it stops being theatre.

4. The family layer: acting for dependents

A grandmother managing her diabetes does not have the same digital skills, the same literacy or the same phone as her grandchildren. Behind many records sits a family reality: a caregiver who handles the phone, or a relative who arrives in an emergency when nobody can consent.

OpesCare has a real **family architecture**, not a share-my-record button. A guardian can act for children and dependents through an explicit **act-as context**, and the dependent's screens name whose record is open, preventing the identity confusion that harms patients. A dependent's records are read through the same endpoints as the guardian's own, so a parent caring for a child has real access to the record, not a cosmetic linked account with the useful parts removed.



Delegation itself is governed. An adult can be invited to help manage someone's care, and the invitation can be accepted or declined — until it is accepted, it grants nothing. There is no social graph, and nobody is added to anyone's care without saying yes.

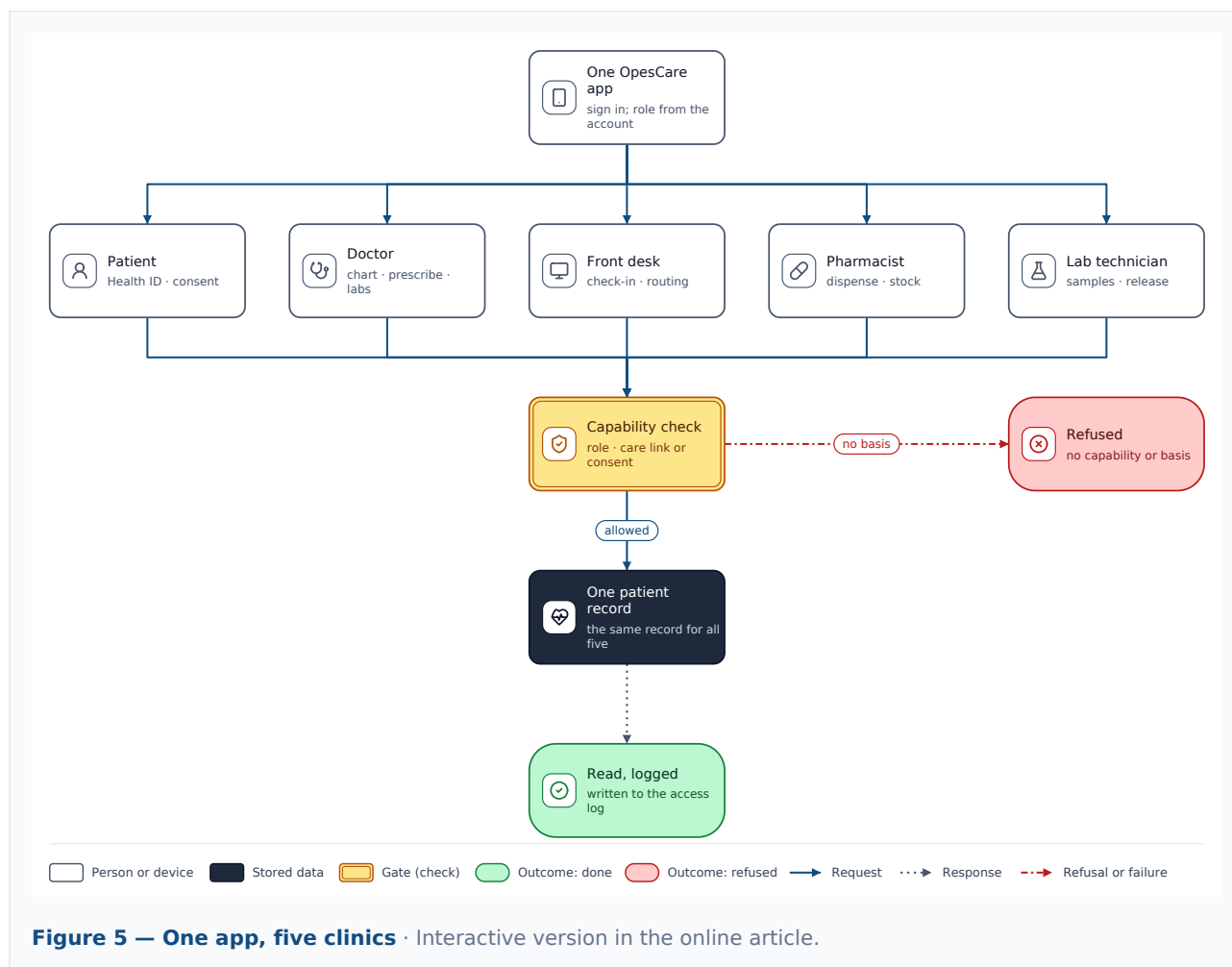
For product people, this is the layer to study. Many patient apps are built for the individual account holder, and the grandmother is left without an answer. OpesCare was built for the household from the start.

5. One app, five clinics

The least obvious product decision in OpesCare — and arguably the boldest — is that **the same app serves the whole clinic**.

When a staff member signs in, the app resolves into a different experience governed by the backend's **role capabilities**. A doctor's day is a schedule, an inbox of appointment requests,

a patient list and a sectioned chart. A **front-desk receptionist** gets the facility's appointment book: check-in, walk-in registration and routing to a clinician. A **pharmacist** gets prescriptions in, dispensing out, reservations, and stock adjustments with an audit trail. A **laboratory technician** gets sample collection, result entry and result release — and releasing a result notifies both the patient and the ordering clinician.



Why is that bold? Because the record is one shared object across those five people. When the same app runs at the front desk, the doctor's desk, the pharmacy and the lab, the record does not have to be carried around the clinic — it is already there. Handoffs that break in paper workflows (a form that does not follow the blood sample, a prescription that does not follow the consultation) become the default.

On top of the roles come the clinical capabilities the record demands: **prescribing**, **laboratory orders**, **follow-up requests**, a **results inbox**, **scanning the Health ID QR code** at the point of care, and the **break-glass** path described above. The clinician's app is a genuine place to write the record — not a billing dashboard pasted onto it.

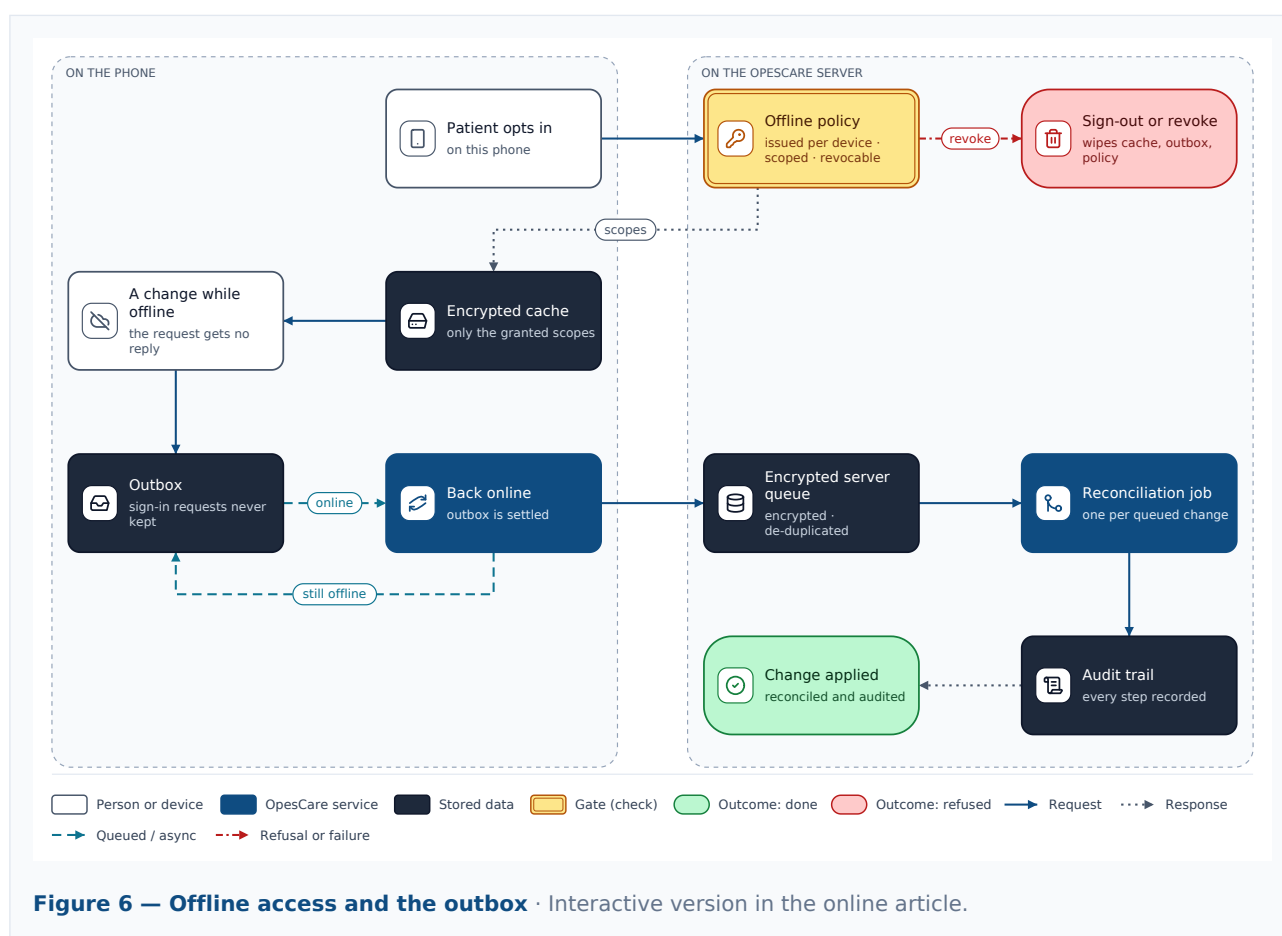
6. Under the hood (for engineers)

For a technical audience, the architecture is clean and the engineering choices are conservative in the right ways.

- **Expo SDK 57, React Native 0.86, React 19 and strict TypeScript:** a single TypeScript codebase for the Android app, with an iOS build profile prepared and a web version.
- **Route-first:** file-based routing across more than ninety screens, with an in-repository navigation audit that flags any screen nothing links to.
- **Server state and client state kept apart:** TanStack Query and Zustand, with one HTTP client and one place where bearer tokens are attached and refreshed.
- **An explicit session model:** one token slot per device with a session-kind marker, so a clinician session and a patient session cannot leak into each other.
- **Offline with a settlement pipeline:** an opt-in, **encrypted** local cache scoped by a server-issued, revocable per-device policy, and an **outbox** that captures writes which failed for lack of a network and settles them into an encrypted server-side queue — de-duplicated, audited, and opened as a reconciliation job. Authentication requests are never written to the device.
- **Transport security:** on Android, a network security configuration pins TLS to certificate-authority root keys rather than to the site certificate, so routine renewals do not break installed apps; no cleartext traffic except a local development profile, app backups disabled, and a short permission list that explicitly blocks storage, microphone and screen-overlay access the app does not need.
- **Quality gate:** TypeScript, ESLint and a Jest suite aimed at the security-critical paths — session wipe, token storage, refreshing an expired token exactly once, the app lock, safe

handling of external links and offline storage — all run in continuous integration on every push and pull request.

- **Internationalisation is systemic**, not skin-deep: English and French at parity across the whole app, and currency in XAF (FCFA).
- **Payments built for the market**: paying an invoice by MTN Mobile Money or Orange Money in XAF is built in — and the product's rule is written into the code and its tests: care is never gated on payment.



7. Where the honest limits are

A serious publication does not pretend a young product is finished. Here is what a technical reviewer will ask about, and where things stand.

- **Mobile-money payments are not yet available.** The MTN Mobile Money and Orange Money flows are built, but real payments need OpesCare's merchant accounts to be connected first. Until then the app says payment is unavailable — and no appointment, record or consultation waits on it.
- **Push notifications do not yet reach the store build.** The machinery exists end to end, and the app is honest about what a given build can and cannot do rather than promising alerts and failing silently, but the push credentials must ship with a release before reminders and result alerts arrive.
- **TLS pinning is Android-only.** The iOS build is otherwise prepared; on iPhone, connections rely on standard certificate validation, and the store release is still on the roadmap.
- **The web version is the smaller sibling, by design.** Browsers have no operating-system secure store, so the web build declines offline caching and keeps tokens in memory, backed only by storage scoped to the browser tab.

One limitation from earlier drafts no longer applies: since 19 October 2026 the app talks to the **production backend at opescare.com**. A production build must name its API explicitly, and the build refuses to ship pointed at a demonstration server by accident.

We would rather these be visible and fixed in flight than glossed over.

8. Why this matters beyond Cameroon

There is a pattern worth naming. Health technology built in high-resource settings tends to optimise the *transaction* — book an appointment, pay a bill, message a doctor. What it misses is the *relationship* underneath: whose data it is, who can see it, who acts for you when you cannot, and what happens when the network drops at the exact moment you need care.

OpesCare builds that foundation first — ownership, consent, delegation, offline access, identity — and then hangs the transactions off it. The order reflects a specific bet: in emerging health markets, trust infrastructure is the scarce good, and booking flows are commodities anyone can buy or rebuild.

Cameroon is where it is being proven, but the pattern is not specific to Cameroon. Whatever label you put on it — *digital health infrastructure, portable medical identity, consent-first health record, a first-of-its-kind patient-owned Health ID network* — the claim underneath is simple:

The record belongs to the patient. Everything else follows from the protocol.

OpesCare is built by OPES Health Systems for care in Cameroon. It is available in English and French, for patients and for staff, on Android and the web today, with iOS on the roadmap. Read more about [how consent works for patients](#), [the Health ID](#), [FHIR](#) and [the Master Patient Index](#), or the [developer documentation](#).

Companion essay: [The Platform Behind the App](#) — how the pharmacy, laboratory, blood bank and front desk work on the same record.

Frequently asked questions

What does "the record is the product" mean?

It means OpesCare is built around the patient's longitudinal record rather than around a single transaction such as a booking or a payment. Finding care, consent, family access and the clinic's own tools all hang off that one record, which the patient carries with them.

Is the OpesCare Health ID the national ID?

No. It is a health identifier issued by OpesCare. It is a first-of-its-kind patient-owned Health ID network in Cameroon, but it is not a government identity document and makes no claim of government endorsement.

Can a clinic read my record just because it scanned my Health ID?

No. Scanning verifies who you are. To read your record a clinician needs a care relationship with you — an appointment, a place on your care team or an open episode at their facility — or a consent grant you approved. Every read is written to an access log you can open.

What happens in an emergency when I cannot consent?

A clinician can use break-glass access. It requires a stated reason, opens only a limited emergency profile, alerts you, and is reviewed afterwards.

Can I manage my child's or my parent's health record?

Yes. A guardian can act for a child or dependent through an explicit act-as context. An adult is never added silently: they receive an invitation and can accept or decline it.

Does OpesCare work without a network connection?

Partly, and only if you opt in. Offline access keeps an encrypted copy of the parts of your record a server-issued, revocable policy allows. Changes made while offline are held in an outbox and settled with the server when you reconnect.

Do I have to pay to receive care through OpesCare?

No. Care is never gated on payment. Paying an invoice by MTN Mobile Money or Orange Money is built into the app but is unavailable until the merchant accounts are connected.

Is OpesCare available on iPhone?

Android comes first. An iOS build is prepared, and the web version runs in any browser, but the iPhone store release is still on the roadmap.

Sources

1. Consent flow — OpesCare documentation — OpesCare
<https://opescare.com/docs/flows/consent> (accessed 19 October 2026)
2. QR verification flow — OpesCare documentation — OpesCare
<https://opescare.com/docs/flows/qr> (accessed 19 October 2026)
3. FHIR R4 ingest flow — OpesCare documentation — OpesCare
<https://opescare.com/docs/flows/fhir> (accessed 19 October 2026)
4. Platform architecture — OpesCare documentation — OpesCare
<https://opescare.com/docs/architecture> (accessed 19 October 2026)
5. HL7 FHIR Release 4 specification — HL7 International
<https://hl7.org/fhir/R4/> (accessed 19 October 2026)

6. Law No. 2010/012 of 21 December 2010 on Cybersecurity and Cybercriminality in Cameroon — Digital Watch Observatory
<https://dig.watch/resource/the-law-no-2010-012-of-21-december-2010-on-cybersecurity-and-cybercriminality-in-cameroon> (accessed 19 October 2026)
7. Network security configuration (certificate pinning) — Android Developers
<https://developer.android.com/privacy-and-security/security-config> (accessed 19 October 2026)



ABOUT THE AUTHOR

Jude Nshome

Jude Nshome is the founder of OpesCare, the bilingual health platform built by OPES Health Systems in Cameroon. He writes about patient-owned health records, consent and building digital health infrastructure for low-resource settings.

Read online: <https://opescare.com/blog/the-record-is-the-product>