

CareCova MVP — Simple Deployment System Architecture (Render + Vercel + Mono + Remita)

1) High-level overview

CareCova is a verification-driven medical financing workflow with **human-in-the-loop approval**, where identity/income/employment signals are pulled via **Mono** and **Remita**, and repayment is tracked after provider disbursement.

2) Deployment

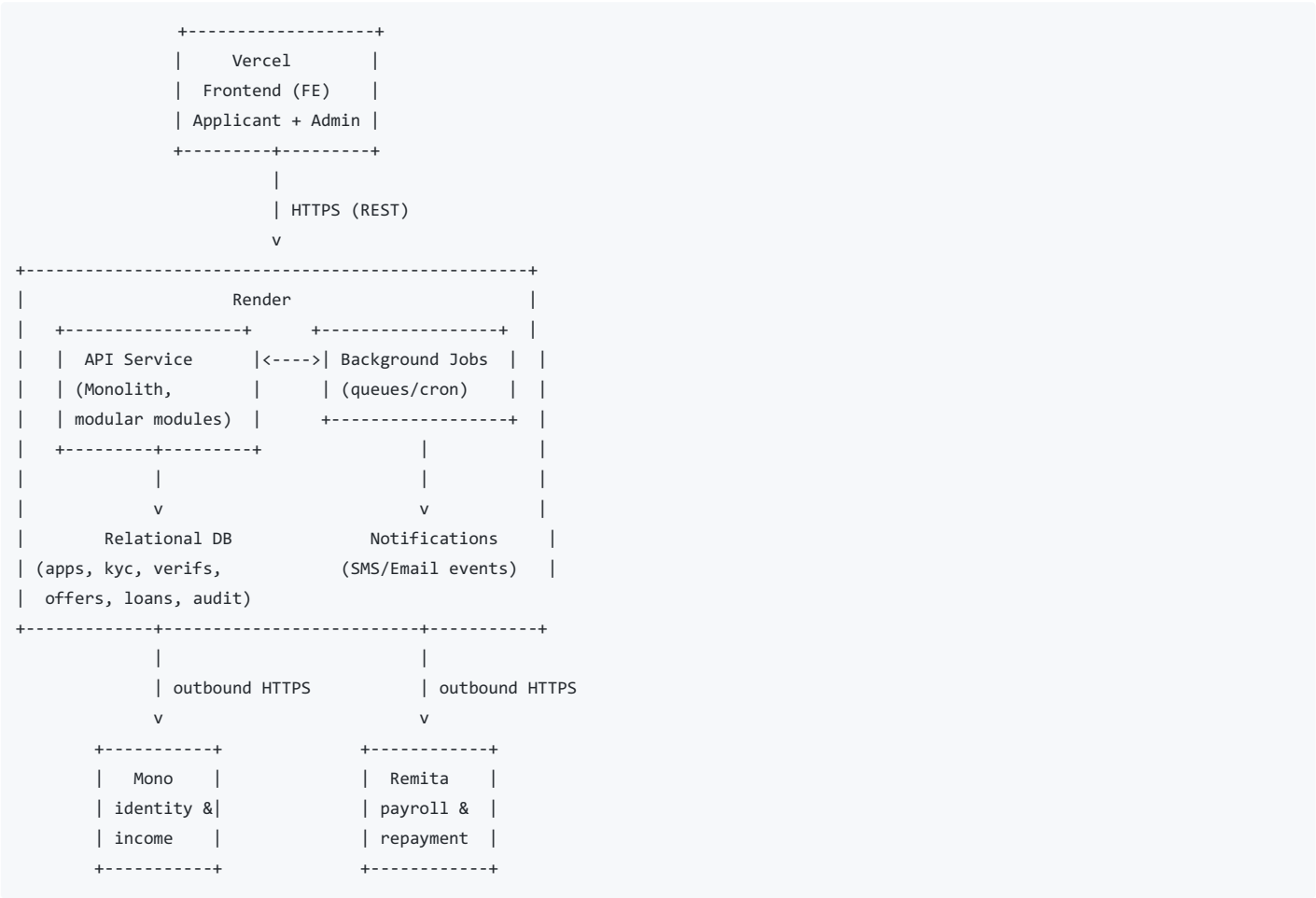
Frontend (Vercel)

- Applicant web app (apply, upload docs, accept offer, repayment view)
- Admin web app (review applications, verification results, risk score, approve/reject, trigger offer)

Backend (Render)

- Monolithic API (modular code) providing REST endpoints and admin APIs
- Background job worker (async verifications, notifications, retries)
- Relational DB (applications, verifications, offers, loans, installments, audit logs)
- Secure integrations layer (Mono + Remita; plus optional credit bureau later)

3) Simple architecture diagram



Core backend modules are aligned to: auth, application mgmt, Mono service, Remita service, risk/offer/loan engines, notifications, audit/compliance.

4) Key components and responsibilities

A) Vercel Frontend

- **Applicant portal**
 - Application form + consent capture
 - KYC uploads
 - Offer acceptance (OTP flow)
 - Repayment status & schedule

- **Admin portal**
 - Queue of applications
 - View verification reports + risk score
 - Approve/reject/modify offer and record reason + timestamp

B) Render Backend (Monolithic, modular)

- **REST API**
 - /auth/*, /applications/*, /offers/*, /loans/*, /admin/*
 - **Background job worker**
 - Run verifications (Mono/Remita calls)
 - Generate schedules, send notifications, retry-safe operations
 - **Database**
 - Core entities: USERS, KYC_PROFILES, APPLICATIONS, VERIFICATIONS, RISK_ASSESSMENTS, OFFERS, LOANS, INSTALLMENTS, PAYMENT_ATTEMPTS, AUDIT_LOGS
 - **Secure integrations**
 - Mono: identity verification + bank statement/income analysis outputs
 - Remita: payroll validation + repayment reference creation + payment tracking
-

5) Data flow (end-to-end, simplified)

1. **Applicant submits application (FE → API)** Backend stores application + KYC + consent; status moves to credit/verification pipeline.
 2. **Verification phase (Jobs → Mono/Remita)**
 - If **Private sector**: Mono verification + income analysis
 - If **Government/public**: Remita payroll validation (if enabled) + income validation
 3. **Risk assessment (API/Jobs)**
 - Compute risk score + flags; status becomes **PENDING_ADMIN_DECISION**
 4. **Admin decision (Admin FE → API)**
 - Approve / Reject / Request docs / Adjust amount
 - Decision is audit-logged (reason + timestamp).
 5. **Offer + OTP (API → FE)**
 - Offer generated with repayment schedule + installment breakdown; user confirms via OTP.
 6. **Disbursement + repayment tracking**
 - Payment sent to provider; loan activated; repayment schedule begins.
 - Repayment priority differs by sector (Remita salary deduction option for government; bank debit/card for private).
-

6) Security + operational notes (kept simple)

- **TLS everywhere**, encryption at rest, RBAC for admin access, immutable audit logs, explicit consent logging
 - Add **idempotency keys** for payment/disbursement calls and retries (especially in background jobs).
 - Rate-limit public endpoints, and use short integration timeouts + retry with backoff.
-