

Project Summary

A leading US-based digital health company operating an established Electronic Medical Records (EMR) platform sought to extend its ecosystem with a fully automated Revenue Cycle Management (RCM) solution. The client serves a growing network of physicians, clinics, and care facilities across the United States, and required a purpose-built RCM layer that could integrate tightly with their existing EMR, automate high-volume billing operations, reduce claim rejection rates, and deliver real-time financial visibility — all within a HIPAA-compliant, scalable architecture. smartData engineered a modular, cloud-native RCM platform covering the complete billing lifecycle: from insurance eligibility verification and charge capture to EDI 837 claim submission, ERA 835 auto-posting, denial management, and executive financial dashboards — delivered over 6 months using Agile Scrum under CMMI Level 3 governance.

Problem Statement

The client's clinical workflows were well-supported by its EMR, but the revenue cycle remained largely manual and disconnected from the clinical data layer. Billing staff relied on fragmented tools and manual processes to verify patient insurance, create and submit claims, and post payments — leading to high error rates and delayed reimbursements.

Key pain points included: claim rejection rates exceeding industry benchmarks due to coding errors that went undetected before submission; time-consuming manual charge entry from completed encounters; no visibility into real-time claim status once submissions were dispatched; delayed or missed secondary claims after primary ERA receipt; and an inability to perform bulk denial corrections at scale. Revenue leakage from missed follow-ups, stale aging buckets, and unreconciled patient balances represented a significant financial risk. The client required a purpose-built RCM platform that could integrate tightly with its existing EMR, automate high-volume billing operations, and deliver real-time financial intelligence across providers and payers.

Approach / Solution

Pillar / Pattern

smartData designed and developed a modular, cloud-native RCM platform built on .NET Core and SQL Server, maintaining a fully independent RCM database that synchronizes encounter, patient, provider, and insurance data from the client's EMR through secure REST APIs. This separation ensures clinical and revenue systems can evolve independently while remaining tightly coordinated.

The platform is structured around the complete RCM lifecycle: insurance eligibility verification (real-time and batch via ClaimMD API), charge capture with CPT/ICD code libraries and fee schedule rules, EDI 837P claim generation and clearinghouse submission, real-time claim status tracking (accepted / rejected / denied / paid), automated ERA 835 import and service-line-level payment posting, prior authorization management (EDI 278), patient statement generation, online payments via Stripe integration, and executive-level financial dashboards with drill-down reporting.

Delivery followed Agile Scrum (2-week sprints) under CMMI Level 3 governance, with CI/CD pipelines, DevSecOps practices, and HIPAA-compliant architecture including RBAC, encryption at rest and in transit, and full audit trails. Phase 1 covers core claim workflows and eligibility; Phase 2 extends to advanced analytics, automation rules, and secondary billing enhancements.

Technical Challenges

- Challenges

- ★ EMR-to-RCM data sync without schema coupling. The client's EMR used a proprietary schema, and any direct database dependency would have created tight coupling that would break on EMR upgrades.
- ★ High claim rejection rates due to pre-submission coding errors (CPT/ICD mismatches, bundling violations, invalid modifiers) that only surfaced after payer rejection.
- ★ Complex multi-payer ERA reconciliation. ERA files (EDI 835) arrive with varying formats, partial payments, contractual adjustments, and coordination-of-benefits scenarios that require precise service-line matching.
- ★ Bulk denial correction at scale. Providers routinely needed to correct and resubmit 80–100 claims in a single batch without disrupting active workflows.

- How We Solved It

- ★ Designed a dedicated RCM database with its own local copies of patients, providers, encounters, and insurance data. Background sync jobs and a manual trigger pull data from the EMR via secured REST APIs, with error logging and retry logic ensuring data consistency without schema dependency.
- ★ Implemented a pre-submission claim scrubber applying configurable backend rules for bundling, modifier validation, MUE-like edits, and payer-specific requirements before any claim reaches the clearinghouse — catching errors at the source and significantly improving the clean-claim rate.
- ★ Built automated ERA parsing and posting workflows that match remittance data to submitted claims at the service-line level, apply contractual adjustments, calculate remaining patient responsibility, and automatically trigger secondary claim generation — with a manual ERA screen as fallback for out-of-band payments.
- ★ Delivered a mass fix-and-refile feature supporting batch corrections and resubmission at that scale, alongside aging views and worklist prioritization to ensure high-value denials are addressed before payer deadlines.

Learning

- ★ Clearinghouse integration (ClaimMD) requires deep upfront alignment on payload formats, response parsing, and error codes — plan 2–3 additional sprints for integration hardening in future healthcare RCM projects.
- ★ The EMR data sync architecture (independent RCM DB + API-based sync) proved to be the right call. Future projects should establish the sync layer and audit logging before building any billing workflows on top.
- ★ Client teams with partial internal technical resources (as with this client's internal technical team) benefit from a collaborative scope definition session early — the feature sheet co-creation process here prevented scope creep and aligned expectations on Phase 1 vs Phase 2 boundaries.
- ★ For post-go-live success in RCM projects, plan for 1–2 dedicated support resources for at least 6–8 weeks — clearinghouse-related claim issues often surface only after real-world claim traffic begins and cannot be fully simulated in staging.