PRODUCT ENGINEERING CASE STUDY

Engineering a domain-specific hiring platform.

A full-stack case study of the data systems, career intelligence, AI workflows, application architecture, employer operations, and engineering logic behind Rack & Role.

Rack & Role | Product Engineering Case Study
November 2025 - August 2026

Public technical edition. Metrics are aggregate; personal data, credentials, private identifiers, and sensitive production records are excluded.

EXECUTIVE OVERVIEW

02

A production platform built as a connected system.

Rack & Role evolved from a specialized technical-jobs concept into a production hiring and career-intelligence platform. Its differentiation came from treating job discovery, candidate evidence, employer workflow, and platform operations as one connected system.

73,409	69,640	13,827	212	93
AGGREGATED JOB ROWS	ACTIVE QUALITY-CLEARED JOBS	DISTINCT COMPANIES	PUBLIC TABLES / VIEWS	EDGE FUNCTION SOURCE UNITS

DATA INTELLIGENCE

Acquire, normalize, reason.

Multi-source ingestion, adaptive keyword rotation, raw/canonical separation, taxonomy, dedupe, quarantine, enrichment, salary estimation, geocoding, URL resolution and source-health telemetry.

CAREER INTELLIGENCE

Translate evidence into action.

0-100 matching, structured profiles, resume parsing/building, fact-grounded tailoring, cover letters, interview prep, skills-gap analysis, learning, certifications, alerts and application tracking.

HIRING OPERATIONS

Support the full lifecycle.

Employer verification, job posting, candidate search, applications, screening, scorecards, interviews, offers, hire packets, onboarding, document sets, analytics, billing and entitlements.

Engineering thesis: the platform became valuable when every visible feature was backed by a data contract, quality rule, state model, permission boundary, or operational control. The work was not simply building pages - it was designing how the system should behave when source data was incomplete, conflicting, stale, expensive, or ambiguous.

Primary stack: React 19 · TypeScript 6 · Vite 8 · Tailwind CSS 4 · React Router 8 · TanStack Query · Supabase/PostgreSQL · Edge Functions · Zod · Stripe · Vitest · Playwright

Model the work, not just the title.

Technical hiring is unusually sensitive to terminology. The same work can appear under many titles, while two similar titles can describe completely different technical responsibilities. Rack & Role was designed around the work itself.

DOMAIN PROBLEM

Job titles are weak identifiers.

Employer vocabulary varies across hyperscalers, colocation operators, contractors, OEMs, skilled trades, software teams and AI-infrastructure organizations. Matching only on title creates both false positives and missed opportunities.

Example: Critical Environment Technician and Data Center Technician roles can coexist in the same facility while requiring materially different mechanical/electrical versus server/network evidence.

SYSTEM RESPONSE

Represent fit as structured evidence.

Role family, skills, certifications, clearances, shifts, systems, seniority, location, work context and evidence quality became first-class data used across search, enrichment, profiles and matching.

This same structured model then powered downstream resume, skills-gap, learning and employer-search experiences.

CANDIDATE EVIDENCE

Experience, skills, certifications, preferences, availability, documents and application state.

JOB EVIDENCE

Canonical roles, requirements, compensation, geography, ATS destination and source provenance.

EMPLOYER EVIDENCE

Verified companies, postings, screening rules, pipeline state, offers and onboarding.

OPERATIONAL EVIDENCE

Source yield, health, run events, quarantine, telemetry, QA and audit signals.

The product model was built to answer a harder question than “does this title match?” - does the evidence describe the same work?

A modular product surface over data, platform and operations planes.

The final web product was a React application over a PostgreSQL-centered platform plane. Public surfaces, candidate tools, employer workflows, ingestion, AI, billing, storage and operations were separated into domain modules with explicit contracts.

EXTERNAL INPUTS
Aggregators · ATS feeds · employer jobs · candidate profiles · documents · payment events · editorial inputs

PLATFORM PLANE
PostgreSQL + RLS · Auth · Storage · Edge Functions · RPCs · feature flags · entitlements · usage/event records · webhooks · analytics

CANDIDATE DOMAINS
Jobs · profile · resume · career tools · applications · learning · certifications · messaging · privacy

PRODUCT SURFACES
Public job/company pages · Candidate workspace · Career Studio · Employer console · Admin console

DATA INTELLIGENCE PLANE
Ingestion orchestration · raw/canonical modeling · validation · dedupe · quarantine · taxonomy · enrichment · salary · geocoding · matching

EMPLOYER DOMAINS
Companies · postings · applicants · screening · interviews · offers · onboarding · search · analytics · billing

OPERATIONS PLANE
Source configuration · keyword telemetry · health/self-heal · moderation · QA · security checks · KPI snapshots · release contracts

Domain-driven frontend

- Feature modules owned components, hooks, API functions and query keys.
- Shared primitives stayed separate from business logic.
- Cross-feature access used public module boundaries.
- Automated cycle checks enforced layering.

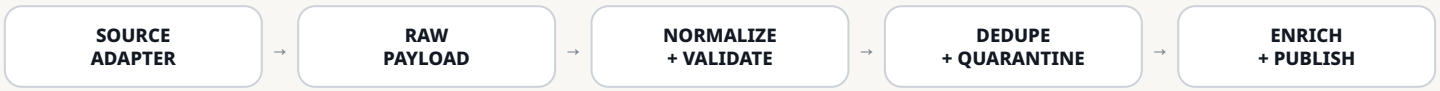
Typed backend contracts

- Zod validation and typed database access.
- RLS and role-aware authorization in PostgreSQL.
- Edge Functions for server-side workflows and integrations.
- RPCs for atomic counters, summaries and domain operations.

05

A resilient acquisition pipeline built for heterogeneous sources.

Job ingestion was engineered as a replaceable-adapter pipeline rather than a collection of scrapers. Each source had its own request shape and operating constraints, but all output converged into the same validation and persistence contract.



Source adapters

- Aggregator, government and ATS-specific connectors.
- Keyword-scoped and company-scoped source strategies.
- Bounded retries, request budgets and runtime ceilings.
- Per-source geography, completeness and yield policies.

Canonicalization

- Raw source evidence preserved separately.
- Normalized title, company, location, salary and employment fields.
- Canonical keys and source identifiers for persistence.
- Application destination resolution and description cleanup.

Operability

- Run records, heartbeats, progress logs and event history.
- Stuck-run detection and recovery controls.
- Quarantine queues and corrective operator actions.
- Health, daily summaries, maintenance and self-heal tooling.

Final source paths included: Adzuna · JSearch · Greenhouse · Lever · Ashby · Workable · Firecrawl-assisted application/source resolution. Earlier generations also included additional source families as the acquisition strategy evolved.

Design decision: source-specific behavior was isolated behind adapters so providers could change without forcing every downstream search, matching, salary or candidate feature to understand the original payload format.

06

Keyword rotation became a measured feedback loop.

Keyword-based sources created an optimization problem: run too few queries and coverage collapses; run everything and runtime, API cost and duplication explode. Rack & Role solved this with a stateful, performance-aware rotation system.

01
Interleave role groups
Master keywords were grouped by occupational family, then interleaved so sequential windows retained category diversity.

→

02
Lock source state
Each source stored its own rotation offset. An ingestion lock prevented concurrent runs from racing the same state.

→

03
Run bounded window
The scheduler selected only the next safe number of keywords for that source and advanced the offset after use.

04
Measure true yield
Each persisted row retained its originating keyword. Returned and inserted counts were recorded per source/keyword.

→

05
Learn from history
Low-yield terms could be suppressed only after enough observations; high-yield terms could receive bounded priority.

→

06
Preserve exploration
Performance optimization was capped so successful terms did not eliminate broad category coverage or trap the system in local maxima.

Concrete guardrails

- Keyword performance used real inserted counts, not placeholder metrics.
- Performance filtering required repeated observations before acting.
- High-yield injection was limited to a fraction of each run.
- Filtering automatically fell back if it would remove too much of the pool.

Runtime-aware scheduling

- Keyword sources carried explicit per-run budgets.
- Edge-function execution targets stayed below runtime ceilings.
- Adzuna generations were tuned down to four keywords per cron cycle when larger windows exceeded wall-clock budgets.
- Company-scoped ATS sources did not use irrelevant keyword-rotation flags.

Why it mattered: acquisition became a feedback-control system: *observe* → *measure* → *adjust* → *continue exploring*. That is materially different from periodically replaying a static search list.

Normalization, deduplication and provenance were product features.

Every upstream source carried different assumptions about location, salary, descriptions, expiration and application URLs. Quality logic was layered so a weak field did not silently become authoritative simply because it arrived from an API.

NORMALIZE

Canonical representation

- Title/company/location cleanup.
- Employment-type normalization.
- US geography inference and territory handling.
- Description sanitation and bounded storage.
- Canonical keys built from company, title, geography and destination context.

PROTECT

Dedupe + quarantine

- Exact source/external identifiers.
- Canonical-key strategies across sources.
- Persisted rejection/dedupe decisions.
- Quarantine instead of silently publishing suspicious records.
- Admin review and release workflows.

MAINTAIN

Freshness + destination

- Source expiry checks.
- Pipeline age rules and display freshness rules.
- Application URL validation.
- Resolved employer destinations preferred over aggregator redirects.
- Historical heuristics could be rolled back when production evidence contradicted them.

SALARY EXTRACTION PATH

SOURCE FIELDS → TEXT PARSING → NORMALIZED ANNUAL RANGE

Supported explicit annual ranges, “k” ranges and hourly compensation converted to annual equivalents under bounded sanity rules.

LOCATION PATH

RAW LOCATION → CITY / STATE INFERENCE → SEARCHABLE GEOGRAPHY

Location normalization supported city/state inference, remote patterns, US territories and source-specific confidence.

Data-engineering principle: raw evidence remained available beside canonical fields. That preserved provenance and made it possible to improve normalizers or classifiers without rewriting history.

Convert inconsistent hiring language into structured domain evidence.

Rack & Role added a domain-specific semantic layer over the job corpus so search and matching could reason about roles, skills, certifications and evidence quality rather than raw employer vocabulary.

Canonical role taxonomy

- 108 canonical roles in the mature taxonomy snapshot.
- 280 role aliases mapped inconsistent employer wording.
- 1,646 anti-merge rules protected distinctions that should never collapse.
- Role families spanned critical environments, electrical, mechanical, MEP, networking, construction, software, AI/ML, hardware, operations and adjacent technical work.

Enrichment v2

- Structured skills, certifications and experience evidence.
- Explicit, extracted and inferred signals kept distinguishable.
- Mismatch flags and guardrail monitoring.
- Shadow orchestration before scoring/read activation.
- Resumable backfills for large-corpus changes.
- Match-diff and scoring QA before rollout.

GUARDRAIL EXAMPLE

CET ≠ DCT. Similar location context did not override materially different occupational evidence.

EVIDENCE MODEL

Signals could be explicit in text, extracted from context, or inferred with lower confidence rather than flattened into one boolean.

ROLLOUT MODEL

New enrichment/scoring behavior could run in shadow mode, compare outputs, then activate behind feature flags.

Taxonomy was not a tag list. It was a set of rules about what may be considered equivalent - and what must remain distinct.

09

Turn structured evidence into explainable career decisions.

Once candidate and job evidence were normalized, the platform could calculate a 0-100 match score using multiple dimensions instead of a single text similarity signal.

CANDIDATE EVIDENCE + **CANONICAL JOB EVIDENCE** → **RULES + WEIGHTS** → **GUARDRAILS + EXPLANATIONS** → **0-100 MATCH**

Hard evidence

- Role-family compatibility.
- Required vs preferred certifications.
- Clearance requirements and compatibility.
- Experience/seniority signals.
- Location and work-context requirements.

Skill evidence

- Aliases and canonical skill concepts.
- Explicit/extracted/inferred provenance.
- System/tool overlap.
- Transferable technical evidence.
- Penalty/cap logic for genuinely missing requirements.

Candidate preference

- Shift preferences.
- Availability/work context.
- Location/remote preferences.
- Target roles.
- Saved-search criteria and alert continuity.

Scoring integrity

Matching logic was synchronized across application paths and supported dedicated fixtures, title-family guardrail tests, sync tests and match-diff monitoring. This reduced the risk that two surfaces would silently assign different scores to the same evidence.

From score to action

The same evidence model supported search filters, recommendations, saved searches, alerts, skills-gap analysis, learning pathways, certification tracking and explainable candidate/job comparisons.

Product result: matching became a reusable career-intelligence primitive rather than an isolated badge on a job card.

Build a compensation signal without hiding evidence quality.

Many technical listings omit compensation or express it inconsistently. Rack & Role built a layered salary-intelligence path that preferred direct evidence and only moved to estimates when source data was incomplete.

DIRECT SOURCE → TEXT EXTRACTION → ROLE / STATE PERCENTILES → NATIONAL + COL FALLBACK → SENIORITY + SANITY GUARDS

Evidence ladder

- Parse explicit min/max fields when supplied.
- Extract annual ranges from description text.
- Extract “$Xk-$Yk” patterns.
- Convert bounded hourly rates into annual equivalents.
- Use empirical state/role distributions only when minimum sample thresholds were met.

Calibration

- National baseline rows supported sparse local markets.
- Cost-of-living multipliers localized fallback ranges.
- Technician and engineering seniority paths were calibrated separately.
- Floor, ceiling and ratio checks rejected implausible ranges.
- Salary source/confidence remained part of the record.

CANDIDATE UX

Salary insight could be shown even when a listing lacked a clean employer-provided range.

DATA INTEGRITY

Estimated compensation was distinguishable from directly extracted compensation rather than presented as equivalent truth.

OPERATIONAL VALUE

The same salary logic supported search, market comparison and career-planning surfaces.

Ground AI-assisted career documents in a verifiable fact model.

The resume stack grew from parsing and visual templates into a structured document system designed to improve a candidate's presentation without inventing qualifications.

Parsing + document model

- PDF validation before processing.
- Text-first extraction when the source contained sufficient text.
- AI-assisted document parsing path for difficult/low-text files.
- Structured JSON output instead of free-form prose.
- Resume builder populated from profile/work/education/skills/certification evidence.
- Multiple visual and ATS-oriented templates.

Two-pass grounded tailoring

- Source resume hashed into a reusable cache key.
- Pass 1 extracted an immutable structured fact ledger.
- The ledger prompt prohibited inference, title inflation, invented metrics and unsupported skills.
- Pass 2 tailored only against ledger facts and target-job context.
- Validation checked unsupported claims, numerical inconsistencies and identity/certification mismatches.

Key design decision: generation was downstream of evidence. The system first established what was verifiably true, then allowed AI to improve framing and relevance inside that boundary.

Resume/Career Studio capabilities: parsing · structured editing · visual templates · job-specific tailoring · per-section regeneration · streaming output · cover letters · inline rewrite · profile-to-resume synchronization · document vault concepts

Use AI inside governed product workflows, not as the workflow itself.

AI was integrated as a workflow component across candidate, content and data systems. The recurring pattern was to give models a narrow responsibility, demand structured output, and surround generation with deterministic state and validation.

CANDIDATE AI

Career Studio

- Resume parsing and tailoring.
- Cover-letter generation and streaming.
- Inline resume/letter rewrite.
- Interview preparation.
- Skills-gap analysis.
- Career-tool workflows.

DATA AI

Enrichment

- Structured skill/certification evidence.
- Role/taxonomy assistance.
- Job enrichment and signal extraction.
- Scoring support in selected generations.
- Batch/backfill orchestration.
- Shadow comparison before activation.

CONTENT AI

Distribution + Signal

- Editorial processing.
- Social/newsletter distribution tooling.
- Structured generation contracts.
- Usage and rate controls.
- Operational spend awareness.
- Human review where appropriate.

Structured-output discipline

Schemas, typed parsers and narrow prompts reduced the number of places where generated prose could become application state without validation.

Deterministic boundaries

Facts, permissions, usage counters, match rules, application states and persistence remained conventional software logic rather than model decisions.

Fallback behavior

When an AI-assisted step could not produce safe/usable output, workflows retained explicit retry, cleanup or manual paths instead of treating model completion as success.

Cost-aware operation

Usage accounting, rate/entitlement controls and operator visibility kept AI behavior tied to the economics and reliability of a production platform.

The most important AI work was not adding a model. It was deciding where the model was allowed to make decisions.

Treat job application as a capability-and-evidence problem.

Application automation became one of Rack & Role's deepest engineering programs because every ATS presented a different combination of URLs, authentication, form semantics, document handling, dynamic UI and confirmation evidence.

Automation generations

- ATS/platform detection and destination resolution.
- API-first submission experiments where provider contracts allowed it.
- Browserbase-managed browser sessions.
- Stagehand and Playwright browser automation.
- Session persistence, prewarming and lease coordination.
- Secure live-session visibility and human takeover concepts.
- Status reconciliation and terminal-state recovery.

Transport architecture

- CandidateApplicationProfile with fact, provenance and privacy concepts.
- Normalized application-form representation.
- Immutable application snapshots for reproducibility.
- Capability verdicts before an adapter could act.
- Idempotent application runs.
- Append-only application-run events as an evidence trail.
- Evidence policies tied to what each rail could actually prove.

DETECT DESTINATION → **RESOLVE CAPABILITIES** → **SNAPSHOT FACTS + FORM** → **EXECUTE PERMITTED RAIL** → **RECORD EVIDENCE**

Engineering evolution: as reliability, privacy and platform constraints became clearer, the final design shifted toward explicit Manual Assist and External Handoff rails. That change preserved candidate control while retaining the strongest architectural work: normalized forms, provenance, idempotency, capability checks and evidence.

ATS families handled across the project lifecycle: Lever · Greenhouse · Ashby · Workday · SmartRecruiters · iCIMS · BambooHR · Taleo · SuccessFactors · Jobvite · BrassRing · generic/external handoff patterns

A connected career workspace built on shared evidence.

The candidate product expanded into a career workspace that connected discovery, identity, evidence, documents, applications and development instead of ending at the job listing.

IDENTITY
Profile · onboarding wizard · military / neurodivergent / vibe-coding pathways · availability · certifications

DISCOVERY
Job search · 0-100 matching · saved jobs · saved searches · alerts · recommendations · job hubs

CAREER TOOLS
Career Studio · resume builder · cover letters · interview prep · skills gap · salary insights · learning

WORKFLOW
Application pipeline · messaging · notifications · background-check requests · privacy/export/deletion

Talent Cards + Talent Wall

Public-facing candidate projections translated a richer internal profile into a controlled talent-discovery surface. Candidate verification, profile moderation and privacy rules supported employer search without exposing unrestricted raw account data.

- Structured skills and certifications.
- Role targets and experience evidence.
- Public/private projection concepts.
- QR/share and presentation tooling.

Candidate media + documents

The platform included visual profile tooling, document storage concepts and client-side image processing. Browser-side background removal/image tooling reduced the need to send every media transformation through a server workflow.

- Photo Studio/background removal.
- Resume/document vault concepts.
- Structured candidate documents.
- Data export and account deletion controls.

Product integration: a saved job could feed matching, resume tailoring, application state, interview preparation, salary context and learning recommendations because those experiences shared the same candidate/job evidence model.

From job posting to an end-to-end hiring operating system.

The employer side grew into a permissions-heavy hiring operating system: company identity, job creation, applicant review, evaluation, interviews, offers, onboarding, talent discovery and commercial entitlements.

Company + jobs

- Company profiles and verification.
- Company ownership/role controls.
- Job posting and moderation.
- Employer dashboards.
- Plan-aware feature access.

Applicant pipeline

- Applications and status workflows.
- Screening questions.
- Scorecards/reviewer workflows.
- Interview scheduling/requests.
- Messaging and notifications.

Hire + onboard

- Offer workflows and offer letters.
- E-sign/audit concepts.
- Hire packets.
- Onboarding checklists/tasks.
- Employer document sets.

TALENT DISCOVERY

Candidate Search / Talent Search

Structured candidate filtering, Talent Card projections, saved-search concepts and employer access gates extended the platform from inbound applicants into proactive talent discovery.

COMMERCIAL INFRASTRUCTURE

Stripe + entitlements

Candidate and employer billing evolved through multiple generations of checkout, portal/webhook handling, plans, promotions, usage counters and entitlement resolution. Feature access was enforced as application state rather than only hidden in the UI.

BACKGROUND-CHECK ARCHITECTURE

Provider adapters, request lifecycle, normalized webhooks, signature/idempotency patterns, consent/document concepts and employer/candidate workflow surfaces.

ENTERPRISE + ANALYTICS

Enterprise dashboards and employer analytics combined jobs, applicants, candidate activity and operational information into management surfaces.

Systems implication: once hiring moves beyond job posting, every action carries ownership, permission, audit and lifecycle requirements. Employer functionality therefore drove significant backend and state-model complexity.

Build the tools required to operate the tools.

A large portion of Rack & Role existed for the operator rather than the public user. Internal tooling made ingestion, quality, spend, moderation, security and system health observable enough to manage.

Ingestion control plane

- Source configuration and source health.
- Current/historical runs and event logs.
- Keyword performance and rotation state.
- Quarantine and auto-review.
- Apply URL resolver history.
- Manual source runs and corrective actions.

Reliability + self-heal

- Heartbeats and progress telemetry.
- Stuck-run detection/recovery.
- Failure/daily reports.
- Database maintenance.
- Description hydration and enrichment batches.
- Health checks and operator alerts.

Admin operations

- Identity/role management.
- Candidate/company verification.
- Talent Card moderation.
- KPI snapshots.
- Audit logging.
- Training-data export governance.

OPERATIONAL FEEDBACK LOOPS

The admin console was effectively a second product: an instrument panel for the platform itself.

Production engineering required permission boundaries and repeatable verification.

Rack & Role combined public job data with authenticated candidate and employer workflows, private documents, billing, AI usage and administrative capabilities. Security and QA therefore had to be built into the architecture rather than added at the end.

Security + privacy engineering

- PostgreSQL row-level security across public tables.
- Role-aware backend contracts and ownership checks.
- Private storage patterns for user documents.
- Authentication, abuse/rate-limit controls and security headers.
- Audit logs for administrative actions.
- Candidate data export and permanent deletion workflows.
- Repeated security audits and remediation passes.

Quality engineering

- TypeScript typechecking.
- Vitest and Node test suites.
- Playwright browser QA.
- Feature-cycle checks.
- Matching synchronization checks.
- Type/contract synchronization.
- Edge Function registry checks.
- Bundle-budget and full-build checks.

462
PASSING TESTS AT FINAL VERIFICATION

0
PUBLIC TABLES WITHOUT RLS AT FINAL SECURITY CHECK

CI
TYPE, LINT, CYCLES, CONTRACTS, TESTS, BUNDLE CHECKS

QA
BROWSER, INTEGRATION, FIXTURES, SYNC AND SMOKE COVERAGE

Engineering practice: audits were used as design inputs. Authorization mismatches, CI gaps, persistence inconsistencies, unsafe workflow patterns and browser-policy conflicts were traced back into code and platform changes rather than treated as one-time compliance work.

Engineering the public surface for searchability, speed and clarity.

Rack & Role also treated public discovery as an engineering system. Job data, company data and editorial content were transformed into indexable surfaces with route-specific metadata and distribution tooling.

Programmatic discovery

- Job detail and company pages.
- Role/location/state/region job hubs.
- Core and dynamic sitemap generation.
- Route-specific canonical URLs.
- Open Graph and structured JSON-LD metadata.
- Indexing submission tooling.

Signal editorial system

- Technical-industry content feed.
- Editorial processing and categorization.
- Bookmarks/feed interactions.
- Social/newsletter distribution pipeline.
- Admin/operator content controls.

Performance + UX

- Lazy-loaded route/section patterns.
- TanStack Query caching and query-key discipline.
- Explicit image dimensions and font fallbacks for layout stability.
- Bundle-budget checks.
- Responsive/mobile-safe UI utilities.
- Shared loading/error/empty-state primitives.

Design system: Sora + IBM Plex typography · mission-critical blue · signal orange · infrastructure teal · hardware-grade dark surfaces · thermal gradient language · semantic chart/status tokens · light/dark themes · Radix/shadcn primitives

Full-stack detail: even public discovery depended on build scripts, sitemap generation, metadata builders, job/company data projections, route architecture and performance budgets - not just page copy.

Nine months of continuous architecture, product and operational iteration.

The project's depth came from iteration. The architecture did not stay fixed: systems were decomposed into feature domains, pipelines were versioned, application automation changed strategy, enrichment was rebuilt, billing evolved, and operational controls expanded as scale exposed new requirements.

NOV 2025 · INITIAL PRODUCT

Candidate, employer and admin surfaces with authentication, resume and early AI workflows.

JAN-FEB · DOMAIN ARCHITECTURE

Feature modules, jobs as a first-class domain, scheduler/worker generations and early billing.

MAR-APR · AUTOMATION + HIRING

Browser automation, application domain separation, offers, recommendations, background-check architecture and job hubs.

MAY-JUN · PLATFORM PHASE

Career Studio, enterprise, hire packets, matching, privacy, QA, readiness, taxonomy, learning and database consolidation.

JUL-AUG · OPERATIONAL MATURITY

Enrichment v2, ingestion quality, adaptive acquisition, telemetry, audits, SEO/distribution, application transport redesign and final platform hardening.

TECHNOLOGY FOOTPRINT

React 19 · TypeScript 6 · Vite 8 · Tailwind 4 · React Router 8 · TanStack Query · Supabase · PostgreSQL · Edge Functions · Zod · Stripe · Vitest · Playwright · Radix UI · Recharts · Browserbase · Stagehand · Firecrawl · AI Gateway · ONNX / image tooling

Development pattern: the strongest systems were the ones that survived repeated contact with real data and production constraints. Decisions increasingly moved from "can this be built?" to "what contract, telemetry, fallback and guardrail make it reliable?"

What Rack & Role ultimately demonstrates.

Rack & Role became a practical demonstration of end-to-end product engineering: defining a domain problem, building the customer surfaces, creating the data platform underneath them, integrating external services, operating the system, measuring its behavior, and repeatedly changing the architecture when evidence demanded it.

Product + UX

Problem framing · candidate/employer journeys · specialized onboarding · career tools · employer workflows · design system · responsive UX · feature rollout and lifecycle decisions.

Software + systems

React/TypeScript architecture · APIs · Edge Functions · auth · storage · PostgreSQL · RLS · RPCs · integrations · billing · application state machines · modular boundaries.

Data + intelligence

Ingestion · adaptive acquisition · normalization · dedupe · quarantine · taxonomy · enrichment · salary · geocoding · matching · structured AI · evidence provenance.

Automation + operations

Schedulers · workers · browser automation · application transports · retries/recovery · self-heal · operator consoles · telemetry · KPI reporting.

Security + quality

Authorization · RLS · privacy · audits · type systems · tests · Playwright · contract checks · CI · build budgets · release hardening.

Engineering judgment

Versioning systems, measuring yield, preserving provenance, introducing guardrails, rolling back harmful heuristics and selecting safer architectures as constraints changed.

The result was not one feature or one technology stack. It was the ability to reason across product, data, software, infrastructure and operations as one system.

73K+

JOBS PRESERVED IN THE FINAL CORPUS

Lifecycle note. In August 2026, Rack & Role was preserved as a read-only public archive so the product and engineering record would remain accessible. The technical case study focuses on the systems, capabilities and decisions that defined the platform.

Evidence basis: preserved repository history, architecture/runbook documentation, final aggregate platform snapshot and code-level verification. Public-safe edition.