RS&C

RECEIVABLE SERVICES & CONSULTING

Portal Platform Overview

Features · Security · Scalability · Communication
Document Handling · Encryption · Licensing

Confidential — Internal Use

August 2026

Platform at a Glance

5 User Tiers	**40+** Admin Modules
130+ API Routes	**65+** DB Tables
3 Demand Stages	**24 hr** Session Lifetime

Platform Architecture

Full-stack SaaS portal built for receivables management at enterprise scale

Client Layer

- React 18 + TypeScript
- Vite build toolchain
- Wouter client-side routing
- TanStack Query v5 (cache + mutations)
- Tailwind CSS + shadcn/ui components
- Role-aware protected routes
- SessionTimeout + live session revalidation

Application Server

- Node.js + Express (TypeScript via tsx)
- Passport.js LocalStrategy auth
- 130+ RESTful API routes (modular routers)
- Drizzle ORM + raw pool.query for migrations
- OpenTelemetry SDK v2 (traces + metrics)
- In-process rate limiting (IP-based)
- Background email queue (5-min cron)

Data & Services

- PostgreSQL via Supabase (transaction pooler)
- connect-pg-simple session store
- Supabase Storage (documents + PDFs)
- Stripe (payments + webhooks)
- Brevo (transactional email + SMTP)
- OTel OTLP exporter (optional endpoint)
- Health monitoring + auto-restart watchdog

Five-Tier Access Model

Role-based access control with tenant isolation at every layer

Main Admin

userType: mainAdmin

Full platform access — user management, billing, system config, all ops modules

User Admin

userType: userAdmin

Elevated ops access — client/user management, CRM, reports, communications

Client Admin

userType: clientAdmin

Manages their own organisation's users, accounts, and documents within their tenant

Client User

userType: client

Portal access — dashboards, documents, demand workflow, billing, assurance center

Prospect

userType: prospect

Limited intake access — pending conversion; no financial or CRM data visible

- *Tenant isolation enforced server-side via clientId scoping on every data-access route*
- *Permission checks backed by server/roles.ts + adminSettings.ts — never client-supplied claims*

Client Portal

What clients and prospects experience in their authenticated workspace

Dashboard

Account summary, placement status, outstanding balances, recent activity feed

Documents

Upload, view, and download firm-approved documents; download audit trail per file

Demand Workflow

Three-stage demand letter portal: Courtesy → Formal → Final. Status tracking per account

Appointments

Schedule phone, Zoom, or Teams consultations; real-time status updates

Assurance Center

Compliance artifact library, disclosure ledger, assessment request and tracking

Billing

Stripe-hosted checkout, payment history, refund requests, policy acceptance records

Security Settings

Password management, security questions (enhanced tier), email verification

Pro App Features

Mobile-field-capture module overview, feature previews, upgrade prompts

Admin Command Surface

40+ modules across CRM, operations, compliance, and platform configuration

User & Prospect Management	Approval pipeline, account conversion, access grants, ESRT sync
CRM Hierarchy	Clients → Customers → Placements → Invoices with full audit trail
Accounts & Workcards	Account-level journals, work events, placement notes, status codes
Demand Module	Template editor, 3-stage lifecycle, deadline tracking, PDF renditions
Communications Hub	Call log, email templates, comm event journal, SMS data model
Calendar & Appointments	Scheduling, status management, client-facing booking
Reports & Revenue	Receivable term profiles, compensation agreements, payment allocation
Ops Dashboard	Command center, live metrics, telemetry spans, deployment records
Incident Management	13-column incident schema, escalation workflow, SLA tracking
Security Scans	Vulnerability scan imports, triage, target control management
Assurance Center (Admin)	Disclosure ledger, artifact library, assessment management
Assessor Workspace	Invitation system, lifecycle cron, evidence/findings/report portal
Vulnerability Disclosure	Public intake, 13 tables, reporter tokens, remediation tracking
Billing Admin	Stripe checkout review, payment history, refund approvals
Settings — Platform	Branding, storage config, email templates, module catalog
Settings — Security	CSP config, rate limit review, session policy
Settings — Integrations	Live integration status (keys detected, Stripe mode, webhook stats)
Audit Requests & Logs	Public audit questionnaire intake, log viewer, document audit

Demand Letter Module

Structured three-stage receivables demand workflow with PDF generation and audit trail

LTR1 — **Stage 1 Courtesy Reminder**

Prior-effort intake & timeline verification required. Courtesy payment reminder sent to debtor.

LTR2 — **Stage 2 Formal Past-Due**

Stage 1 completion gated. Formal past-due demand with legal tone and payment deadline.

LTR3 — **Stage 3 Final Pre-Placement**

Stage 2 completion gated. Final notice — advises imminent placement to collection agency.

PDF Renditions

Electronic-send PDF · Print-ready PDF · Review/preview PDF
Immutable sent snapshots stored in Supabase rsc-demand-pdfs bucket
Content-hashed for tamper detection

Access Control

Templates scoped: standard · licensed (per client) · administrative exception
Only draft templates are mutable — published templates locked
Full version/audit trail on every template change

Delivery & Ledger

Demand sends trigger Brevo email with PDF attachment
Every send recorded in communication_ledger (provider, external ID, timestamp)
Client-visible delivery status exposed in portal

Security Architecture

Defence-in-depth across authentication, transport, data access, and audit

🔑 Authentication

- Passport.js LocalStrategy with DB-backed sessions
- SESSION_SECRET from env — no hardcoded fallback
- Custom cookie name rsc.sid · HttpOnly · SameSite=Lax
- Secure flag enabled in production (NODE_ENV=production)
- 24-hour session lifetime with server-side destroy on logout
- Password reset uses generic response (no user enumeration)

🔐 Password Security

- Node crypto.scrypt with random 16-byte hex salt
- 64-byte derived key — stored as hexHash.hexSalt
- timingSafeEqual for all comparisons (no timing attacks)
- Constant-time API key validation (apiKeyValid())
- Security questions required for enhanced-tier accounts
- Account lockout on inactive/archived status

🛡️ Transport & Headers

- Helmet.js for HTTP security headers
- Content-Security-Policy (report-only → enforcement pending)
- HTTPS enforced at proxy layer; trust proxy: 1 configured
- All /api/ responses: Cache-Control: private, no-store
- CORS restricted to configured origins
- No secrets logged — redacted in OTel span attributes

🚦 Rate Limiting

- Login: 10 req / 15 min per IP
- Registration / password reset: 5 req / hour
- Billing checkout + refund: 10 req / hour
- File uploads: 20 req / hour; imports: 5 req / hour
- Escalating 15-minute block windows up to ~2 hours
- In-process (stateful per-instance); Redis upgrade path noted

🏢 Multi-Tenant IDOR Prevention

- clientId resolved from authenticated session — never client-supplied
- Every data-access route verifies entity ownership before returning data
- Applies to reads AND writes (correction, approval, status change)
- CRM hierarchy (client → customer → placement → invoice) enforced
- Comm events and demand accounts enforce ownership chain
- Cross-tenant ID rejection at query layer

📋 Audit & Compliance

- security_audit_logs — account/password/access events
- ops_audit_log — admin operations journal
- document_audit — upload/download/approve/delete per file
- communication_ledger — every email send with provider + external ID
- consent_ledger — explicit consent records with IP + user-agent
- vuln_report_audit_log — immutable vulnerability disclosure trail

Encryption & Data Protection

Layered controls for credentials, transport, tenant identity, and storage

🔒 Credential Hashing

- Algorithm: Node.js crypto.scrypt (memory-hard KDF)
- Salt: 16 random bytes, unique per password, stored as hex
- Output: 64-byte derived key stored as hex (128 hex chars)
- Storage format: hexDerivedKey.hexSalt in users.password column
- Comparison: timingSafeEqual — constant time regardless of match
- API keys: constant-time check via dedicated apiKeyValid() helper
- No plaintext, no reversible encryption — passwords cannot be recovered

🌐 Transport Security

- All external API calls use HTTPS (Brevo, Stripe, Supabase)
- Inbound TLS terminated at Replit reverse proxy (trust proxy: 1)
- Secure cookie flag active in production environment
- HSTS and other transport headers via Helmet.js
- CSP currently report-only; enforcement is a tracked pending task
- No credentials in URLs; all secrets via environment variables

🔍 OTel Tenant Pseudonymisation

- Tenant IDs are HMAC-pseudonymised before appearing in trace attributes
- SQL query text and parameters are never recorded in spans
- Only operation type (SELECT/INSERT/…) and table name captured
- Allowlist-based attribute sanitisation in server/telemetry/redact.ts
- Error messages sanitised before export — no PII in telemetry
- HMAC key is per-instance (not shared across scaled replicas)

🗄️ Storage & Database

- Database: Supabase PostgreSQL (provider-managed encryption at rest)
- File storage: Supabase Storage buckets (provider-managed AES-256)
- Demand PDFs: uploaded to Supabase rsc-demand-pdfs bucket (content-hashed)
- Session data: stored in sessions table via connect-pg-simple
- No application-layer encryption on top of provider encryption
- Local disk (uploads/) is ephemeral; Supabase is the durable store
- Supabase access uses service-role key (server-only, never in browser)

Communication Infrastructure

Multi-channel, ledger-backed messaging with full delivery audit trail

Brevo — Primary Email Provider

Transactional API (BREVO_API_KEY) + SMTP relay (BREVO_SMTP_USER / BREVO_SMTP_PASSWORD) · API health-checked on startup · File-based retry queue (email_queue/) · Every successful send is recorded in communication_ledger with provider, external message ID, and timestamp

SMS / Twilio — Data Model Ready

channel=sms in comm_events and communication_ledger. Provider field supports Twilio. Live sending requires TWILIO_* credentials — listed as future module.

Events That Trigger Emails

- User registration / welcome message
- Email verification (verification code, HTML template)
- Password reset (token link, generic response to prevent enumeration)
- New public audit request — staff notification + submitter confirmation
- Admin resend welcome (manual trigger from user management)
- Demand letter send (LTR1/LTR2/LTR3 — PDF attached)
- Account status change notifications (approval / conversion)
- Billing confirmation (Stripe checkout success webhook)

Dual-Ledger Design

communication_ledger

Channel, destination, message type, subject, body, delivery_status (queued/sent/delivered/failed), provider, external_id, metadata JSON

comm_events

Operational journal — call, email, sms, note per account. Direction, contact info, purpose/outcome, amounts, visibility, negotiation notes. Tenant-scoped with client-visible filter.

consent_ledger

Explicit consent records with consent_type, status, method, consent text, IP address, user-agent, and source application.

Email retry queue

File-based queue in email_queue/ — processed every 5 minutes. Failed sends remain for retry; successful sends removed from queue.

Document Handling

Structured upload pipeline with CRM linkage, storage tiering, and immutable audit trail

🗂️ Storage Architecture

Demand PDFs & evidence → Supabase Storage (rsc-demand-pdfs bucket · content-hashed · immutable sent snapshots) | General documents → PostgreSQL metadata + file URL (local disk / configured cloud) | Account photos → Supabase Storage (account-files bucket, streamed via authenticated route) | Transient PDF generation → /tmp (never persisted to disk)

📄 Document Record Schema

- id, user_id, title, file_name, file_url, file_type, file_size
- status: pending → approved / rejected
- uploaded_by_name, approved_by, approved_at, rejected_by, rejected_at
- download_count (incremented on each access)
- CRM links: client_id, customer_id, placement_id, invoice_id
- Optional description and display_id for reporting

🔍 Audit Trail — document_audit

- Every file interaction logged: upload · download · view · delete
- approve · reject events with reviewer user_id
- Fields: file_name, file_type, file_size, action, IP, user-agent
- source_app tag (which portal area triggered the action)
- Immutable append-only log — no delete path in schema
- Used for compliance reporting and access investigation

📊 Supported Use Cases

- Client-uploaded supporting documents (invoices, photos, contracts)
- Admin-managed firm documents (policies, agreements, disclosures)
- Demand letter PDFs (LTR1/2/3 renditions + electronic/print/review)
- Assurance artifacts (compliance evidence library)
- Vulnerability disclosure attachments (disabled by default flag)
- Account work evidence (field capture module)

🔒 Access Controls

- Document download requires active authenticated session
- Client documents scoped to tenant clientId — no cross-tenant reads
- Account photo stream route enforces client_id match before serving
- Admin approval/rejection creates admin_update_log entry
- Upload rate limit: 20 requests / hour per IP
- File type validation at route layer before storage write

Scalability & Observability

Production-grade telemetry, connection management, and background processing

OpenTelemetry Tracing

- OpenTelemetry SDK v2 — HTTP request spans + DB child spans on every query
- OTLP exporter to configurable endpoint; noop processor when endpoint absent
- SQL classified by operation type + table — text/params never recorded
- Tenant IDs HMAC-pseudonymised before span attributes are written
- Console exporter available for local development trace inspection
- Request middleware attaches trace context to all downstream calls

Performance & Connection Management

- PostgreSQL connection pool: max 10 connections, 5-sec timeout
- Drizzle ORM + raw pool.query for migrations and complex joins
- Session store: connect-pg-simple (PostgreSQL-backed, not in-memory)
- Email queue processed every 5 minutes (background cron)
- Rate-limiter cleanup cron every 10 minutes (evicts stale IP buckets)
- Scheduled health monitoring with auto-restart watchdog

Resilience & Fault Tolerance

- Server readiness gate — 503 returned until all modules initialised
- Additive schema migrations (ensureColumns/ensureTables) on every boot
- Idempotent billing table type corrections run at startup
- Stripe webhook idempotency via stripe_webhook_events deduplication table
- Email retry queue (file-based) survives process restart
- Database connection retried with backoff at startup
- Known limitation: rate limiter is per-process — Redis adapter needed for multi-instance

Horizontal Scaling Path

- Stateless application server — horizontal scaling ready (Redis rate-limit adapter)
- Supabase PostgreSQL pooler supports high connection counts transparently
- OTel OTLP endpoint configurable for Datadog, Grafana, Jaeger, or Honeycomb
- All secrets via environment variables — deployable to any container host
- Session store is DB-backed — survives process restart / rolling deploys

Licensing, Billing & Module Architecture

Stripe-integrated payments, per-client template licensing, and module capability framework

Stripe Payment Flow

1. **Client initiates checkout**
   POST /api/billing/checkout-session (rate-limited 10/hr)
2. **Stripe-hosted checkout**
   Stripe creates PaymentIntent
   Client redirected to Stripe UI
3. **Webhook received**
   POST /api/stripe/webhook
   Signature verified (STRIPE_WEBHOOK_SECRET)
4. **Idempotency check**
   stripe_webhook_events table
   prevents duplicate processing
5. **Platform record updated**
   platform_payments status → paid
   policy_acceptances created
6. **Refund path**
   Client submits reason → admin reviews
   → Stripe refund via API

📦 Module Catalog

- Mobile Field Capture — Active (included in base)
- Bulk Import — Add-on (activatable)
- Advanced Reporting — Coming soon
- Mechanics Lien Workflow — Coming soon
- Legal Bundle Export — Coming soon
- White Label Portal — Coming soon
- Agency / Firm Edition — Coming soon
- Module enable/disable backed by featureFlagKey registry

📜 Template Licensing

- Demand templates have a scope attribute:

  standard — available to all clients

  licensed — scoped to a specific licensed_client_id

  administrative_exception — one-off approval
- Published templates are immutable (version-locked)
- Full version/audit trail on every template change
- Clone workflow defaults to administrative_exception scope

🗒️ Receivables & Compensation

- rsc_compensation_agreements — per-client fee structures
- rsc_compensation_components — fee type, %, min/max caps
- receivable_term_profiles — late fees, interest, compounding
- receivable_term_overrides — account/invoice-level exceptions
- payment_records + payment_allocations — gross payments
- account_charge_ledger — calculated charges with audit detail
- policy_acceptances — versioned T&C acceptance records + IP

Security Compliance & Vulnerability Disclosure

Coordinated vulnerability disclosure program with structured remediation and audit trail

Vulnerability Disclosure Program (VDP)

Public Intake	POST /api/vulndisclosure/submit — rate-limited, no auth required. Anonymous reporter token issued for status tracking.
Triage	Admin reviews report, sets severity/status. Communications sent via vuln_report_communications.
Remediation	Fix plan in vuln_report_remediations with target date and owner. Retesting logged in vuln_report_retests.
Risk Acceptance	Unresolved findings can be risk-accepted with expiry date. Requires mainAdmin approval.
Disclosure	Public disclosure record created after remediation. Approval workflow enforced; timing controlled.
Audit	vuln_report_audit_log — immutable record of all admin actions. All status changes, comms, and decisions preserved.

13 VDP Tables

- vuln_reports · vuln_report_events · vuln_reporter_tokens
- vuln_report_communications · vuln_report_comm_revisions
- vuln_report_attachments · vuln_report_duplicates
- vuln_report_remediations · vuln_report_risk_acceptances
- vuln_report_disclosures · vuln_report_retests
- vuln_report_audit_log · vuln_disclosure_policy · vuln_security_txt

Compliance Features

- Consent ledger — explicit opt-in/out records with IP + user-agent
- Communication ledger — immutable send history per user/contact
- Policy acceptances — versioned T&C acceptance with timestamp + IP
- Assessor terms acceptance — IP + timestamp logged at acceptance
- Document audit — every file access action logged immutably
- Correction request workflow — approve-then-apply with field allowlist
- Admin update log — all admin data changes journaled with prior/new value
- security.txt endpoint — machine-readable vulnerability contact policy

Platform Summary & Production Readiness

What is live, what is validated, and what is on the tracked roadmap

✅ Production Ready

- Five-tier RBAC with tenant isolation (clientId-scoped at every query)
- Passport auth + scrypt passwords + timingSafeEqual comparisons
- Supabase PostgreSQL — persistent across deploys
- Brevo email with retry queue and communication ledger
- Stripe sandbox validated (45/45 checks — checkout, webhooks, refunds)
- Three-stage demand module with PDF renditions (Supabase-stored)
- Full audit trail: document, communication, consent, ops, vuln logs
- OTel telemetry with HMAC tenant pseudonymisation
- Session cookie fix — domain-independent (no hardcoded host)
- GET /api/session-check liveness endpoint — no false expiry on login
- 13-table vulnerability disclosure program with public intake
- Assessor workspace with lifecycle cron and invitation system
- Assurance Center — disclosure ledger, artifact library, assessments
- Integrations status API — key detection, webhook stats,

🔧 Tracked Roadmap (Pending Tasks)

- Enforce CSP (currently report-only — task #46)
- Rate limiting validation under sustained burst (task #47)
- Command Center resilience when one source is slow (task #48)
- Suspended account session invalidation (task #43)
- Stale client data on back-button after logout (task #45)
- Security scan → incident escalation workflow (task #19)
- Ops Dashboard security posture summary (task #20)
- Browser end-to-end tests for scan/triage/controls (task #18)
- Backfill missing email audit records (task #32)
- Verify demand emails reach recipients (task #33)
- Fix sign-in/register buttons on main website (task #39)
- Nodemailer CVE removal / upgrade (task #44)

🚀 Future Modules

- SMS / Twilio — data model complete; sending not wired
- Bulk Import — add-on module (featureFlagKey defined)
- Advanced Reporting — module catalog entry, not yet built
- Mechanics Lien Workflow — planned Legal Bundle module
- Legal Bundle Export — planned Legal Bundle module
- White Label Portal — planned enterprise tier
- Agency / Firm Edition — planned enterprise tier
- Google Calendar sync — OAuth setup required
- E-signature (DocuSign / HelloSign) — planned
- OCR / Document Parsing — Mobile Field Capture module
- Carrier API integration — enterprise feature
- Multi-instance rate limiting (Redis adapter)

RS&C Portal Platform

Receivable Services & Consulting

Built for enterprise receivables management.
Secure · Scalable · Auditable.

📧 support@receivableservicesconsulting.com

🌐 receivableservicesconsulting.com

🔐 security@receivableservicesconsulting.com (vulnerability reports)

Confidential — For internal and authorised partner use only.