D2C OPs

Enterprise Service Operations Platform

Technical Documentation

Platform Version: 1.0 | Date: 9 May 2026

Prepared by: Amos Matimba, Software Developer — D2C Telcare

CONFIDENTIAL

1. System Overview

D2C OPs is a full-stack enterprise service operations platform built for D2C Telcare to manage IT operations across 8 client business processes. It replaces fragmented spreadsheets and email chains with a unified, role-controlled operations hub.

Core Capabilities

- Real-time incident management with severity tracking and email escalation
- Downtime logging with automated resolution notifications
- Shift handover management between operations teams
- Ticket tracking across incident, change, access, and report types
- Approval workflows for change and access requests
- Automation rule engine with scheduled and threshold-based triggers
- On-call schedule management
- AI-powered incident root cause analysis (OpenAI GPT-4o)
- Audit logging of all user actions
- Reporting dashboards
- Client-facing notification system

Supported Client Processes

Process	Description
Ignite	Client business process — Ignite
Sunking	Client business process — Sunking
RDG	Client business process — RDG
MobiHive	Client business process — MobiHive
Momo	Client business process — Momo
Sanlam	Client business process — Sanlam
MBA	Client business process — MBA
Muzanu	Client business process — Muzanu

Platform Overview — Dashboard

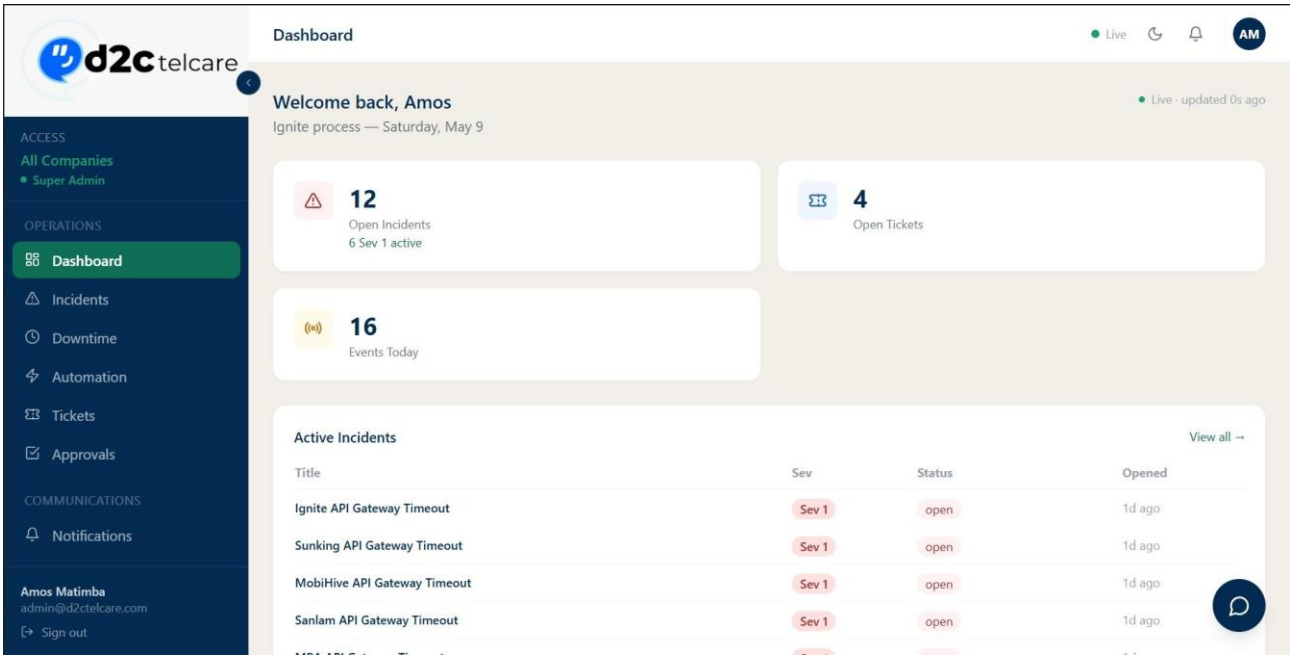

Figure 1.1 — Dashboard showing stat cards (Open Incidents, Open Tickets, Events Today) and Active Incidents table

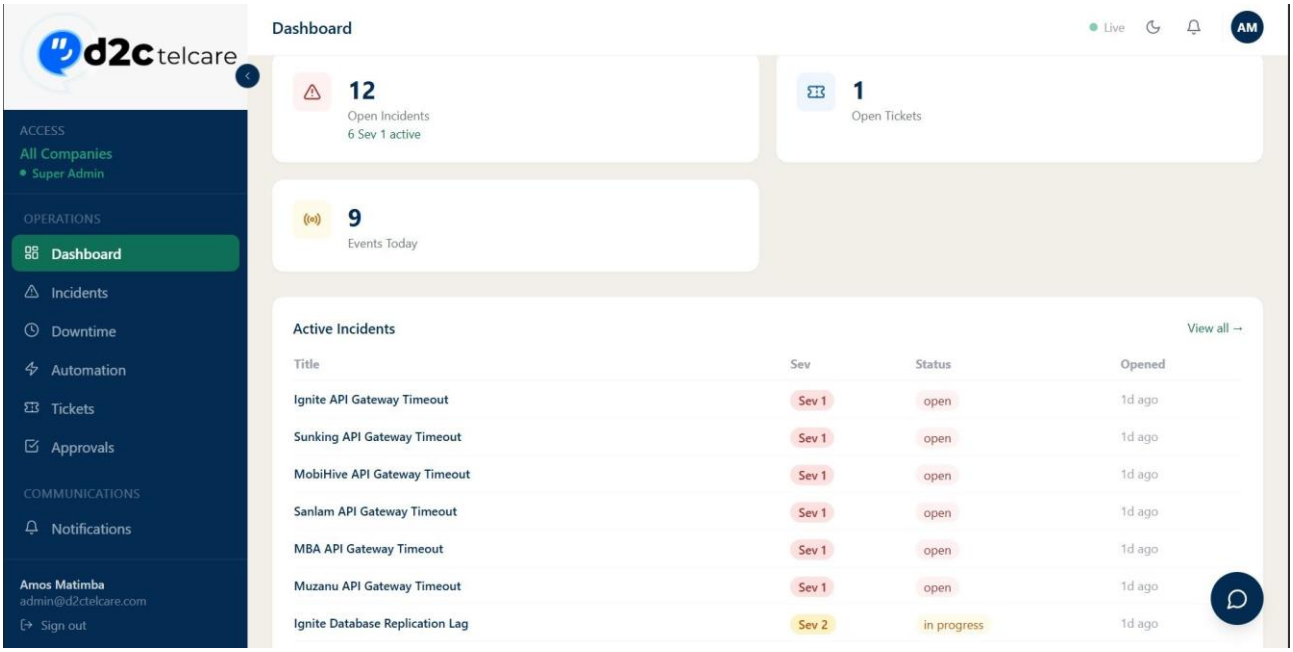

Figure 1.2 — Dashboard scrolled view showing the full Active Incidents table with severity and status badges

2. Architecture

The platform follows a standard three-tier architecture:

```
React Frontend (Port 3000) <--> Express/Node.js API (Port 4000) <--> MySQL Database
```

Frontend

A single-page React application. All routing is client-side. Private routes are protected by JWT token presence in localStorage. A Zustand auth store persists user session and role.

Backend

A RESTful Express.js API written in TypeScript. Every route group is isolated in its own router file. Authentication middleware validates the JWT on every protected request. Process isolation is enforced at the query level — all data reads are scoped to the authenticated user's process_id.

Database

MySQL managed through Knex.js migrations. Eight migrations bring the schema from baseline to production state. A seed file populates processes, users, and test data.

Background Worker

A polling worker runs every 60 seconds inside the server process. It evaluates escalation rules, fires threshold-based automation actions, and sends unattended-incident reminder emails (30-minute cutoff, fires once per incident using a reminder_sent_at flag).

Email

Nodemailer with Gmail SMTP using a lazy singleton transporter (connection pool, maxConnections: 5). All bulk notifications use Promise.all() for parallel delivery.

3. Technology Stack

Layer	Technology
Frontend Framework	React 18 + TypeScript
Build Tool	Vite
Styling	Tailwind CSS (custom design tokens)
State Management	Zustand (persisted to localStorage)
Routing	React Router v6
Backend Framework	Express.js + TypeScript
Database ORM	Knex.js
Database	MySQL 8
Authentication	JWT (jsonwebtoken) — HS256
Password Hashing	bcryptjs

Email	Nodemailer + Gmail SMTP (STARTTLS port 587)
AI Insights	OpenAI GPT-4o (falls back to local regex analysis)

Layer	Technology
Runtime	Node.js 18+

Design Tokens (Tailwind Custom Colours)

Token	Hex	Usage
bg-navy	#042C53	Sidebar, primary text, buttons
bg-teal	#0F6E56	Primary action buttons
bg-teal-light	#1D9E75	Hover states
bg-amber	#BA7517	Medium severity, warnings
bg-danger	#A32D2D	High severity, errors

4. Database Schema & Migrations

The database is named d2c_ops. Knex manages versioned migrations.

Migration History

Migration File	Change
001_initial_schema.js	Base schema — processes, users, incidents, downtime_events, tickets, notifications, approvals, automation_rules, escalation chains, handovers, on_call_schedules, audit_log, activity_log
002_notifications_direction.js	Adds direction column to notifications (ops_to_client)
003_super_admin_role.js	Adds super_admin role enum value to users
004_verification_codes.js	Creates verification_codes table for OTP flows
005_escalation_executions.js	Creates escalation_executions table to track tier firings
006_handover_time.js	Adds handover_time column to handovers
007_ticket_description.js	Adds description column to tickets
008_incident_reminder.js	Adds reminder_sent_at (TIMESTAMP NULL) to incidents for 30-min unattended email

Core Tables

processes
One row per client business process. All data is scoped to a process_id.

users
Platform users. Columns: id, email, name, password_hash, role, process_id, is_active.

incidents
id, process_id, title, description, severity (1–4), status (open / in_progress / resolved), assigned_to, opened_at, resolved_at, rca, reminder_sent_at.

downtime_events
id, process_id, incident_id (nullable), started_at, ended_at, services_affected (JSON array), impact_level (low / medium / high).

tickets
id, process_id, title, description, type (incident / change / access / report), status (open / in_progress / closed), assigned_to, created_at.

approvals
id, process_id, type, title, submitted_by, reviewed_by, status (pending / approved / rejected), submitted_at, reviewed_at.

handovers
id, process_id, submitted_by, recipient_id, shift, handover_time, open_incidents, actions_taken, notes, acknowledged_at.

audit_log
Immutable record of every create/update/delete action: user_id, process_id, action, entity_type, entity_id, created_at.

verification_codes
OTP codes for registration and password reset: email, code, purpose, expires_at, used.

5. Role-Based Access Control

Six roles are defined. Each role inherits the permissions of all roles below it plus its own additions.

Role	Key Permissions
viewer	Read-only access to all module data within their process
operator	+ Create & update incidents, submit handovers, create tickets
manager	+ Approve requests, manage automation rules, export data
admin	+ Manage users, delete records

super_admin	+ View all processes (cross-process access, no process filter applied)
client	View data, receive and send notifications to the ops team

User Management

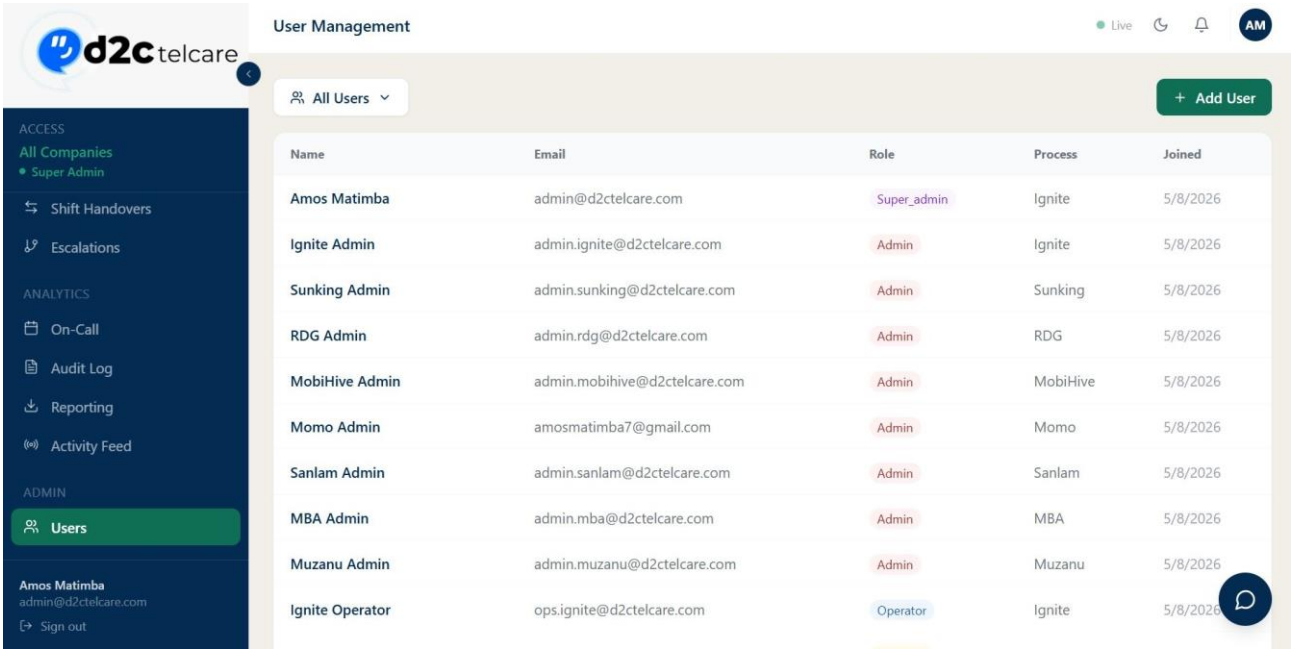

Figure 5.1 — User Management list showing all users with name, email, role badge, process, and join date

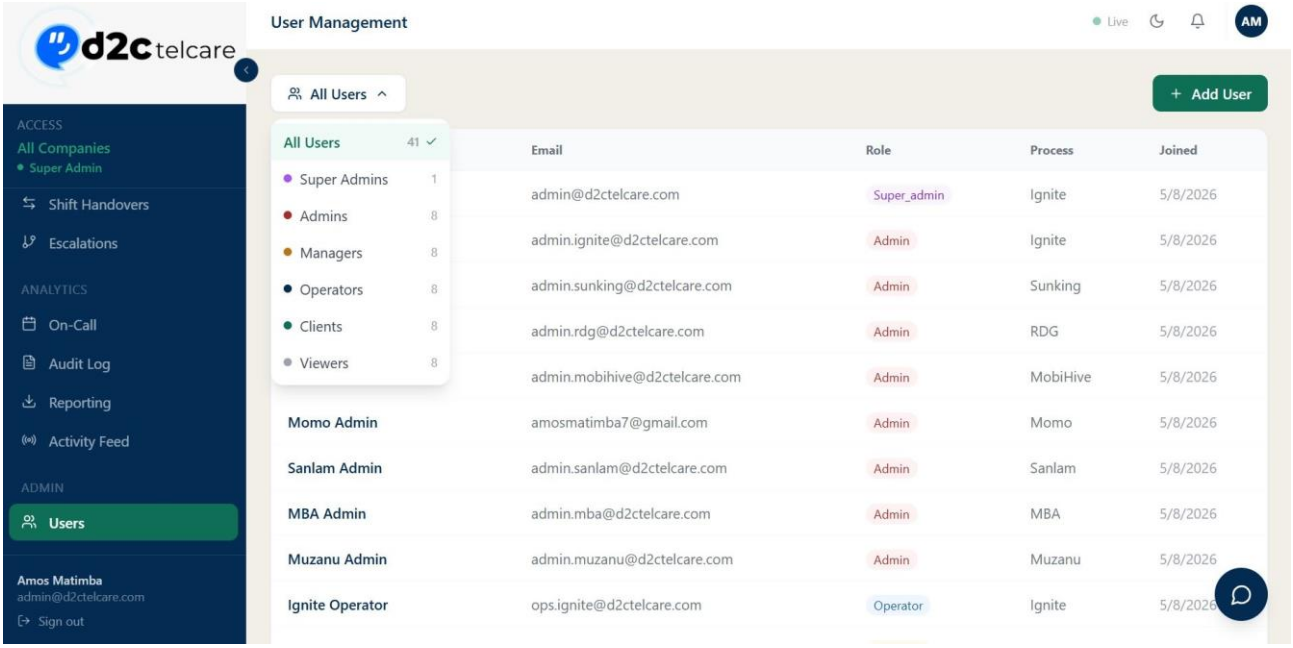

Figure 5.2 — User Management with role filter dropdown showing all role categories and counts

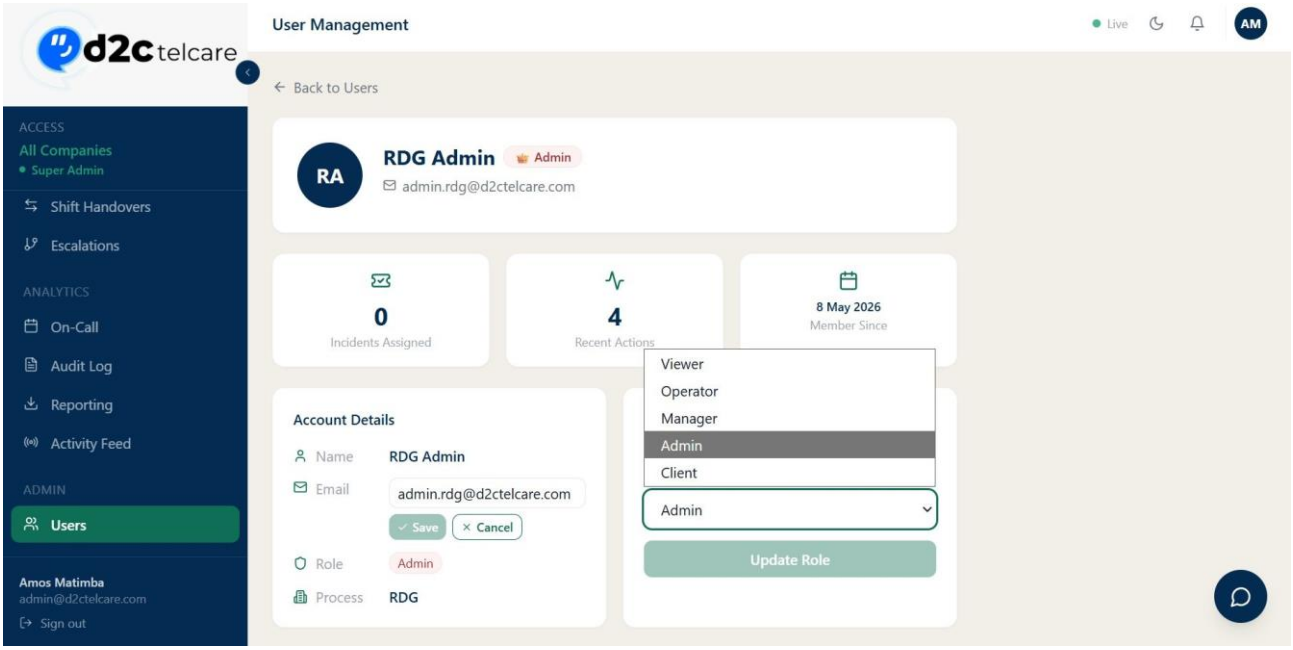

Figure 5.3 — User profile page with role edit dropdown (Viewer, Operator, Manager, Admin, Client)

6. Module Reference

D2C OPs consists of 14 modules accessible from the navigation sidebar.

6.1 Dashboard — Route: / — Access: All roles

The landing page after login. Displays stat cards at the top: Open Incidents, Open Tickets, and Events Today. Below the stats is a live table of active (non-resolved) incidents with severity badges, status, assignee, and time-open. Admins and super_admins see data aggregated across all processes.

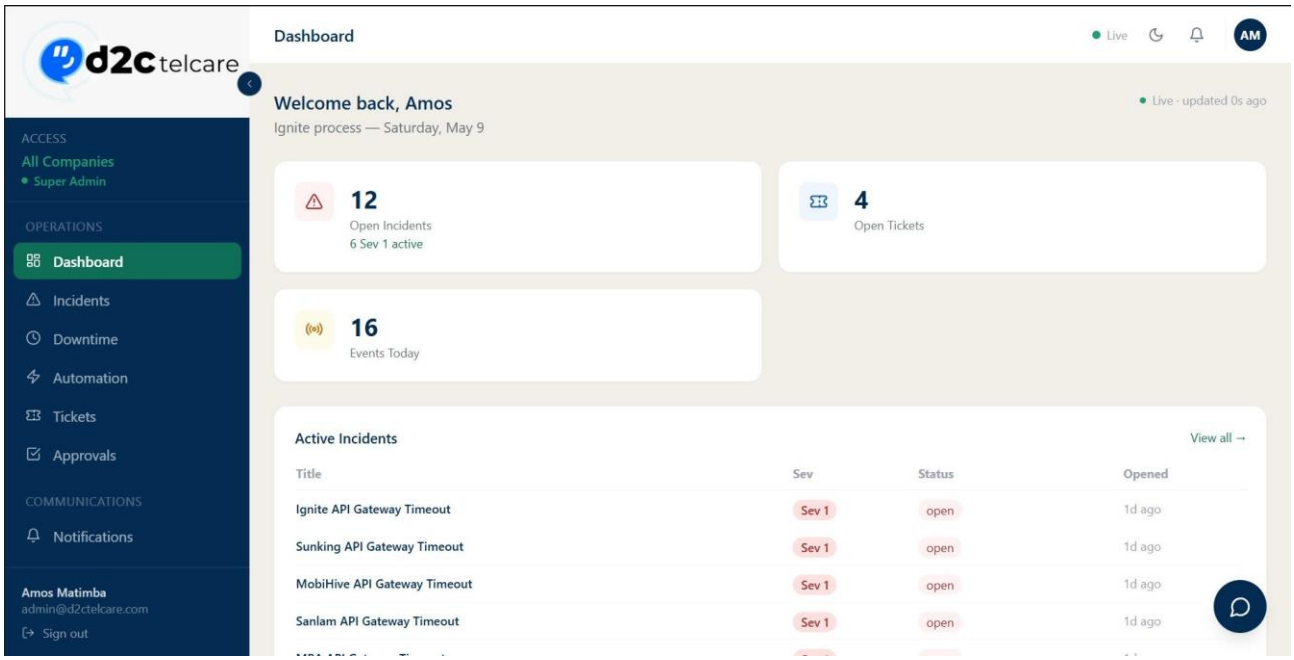

Figure 6.1a — Dashboard showing Open Incidents (12, 6 Sev 1 active), Open Tickets (4), Events Today (16) and Active Incidents table

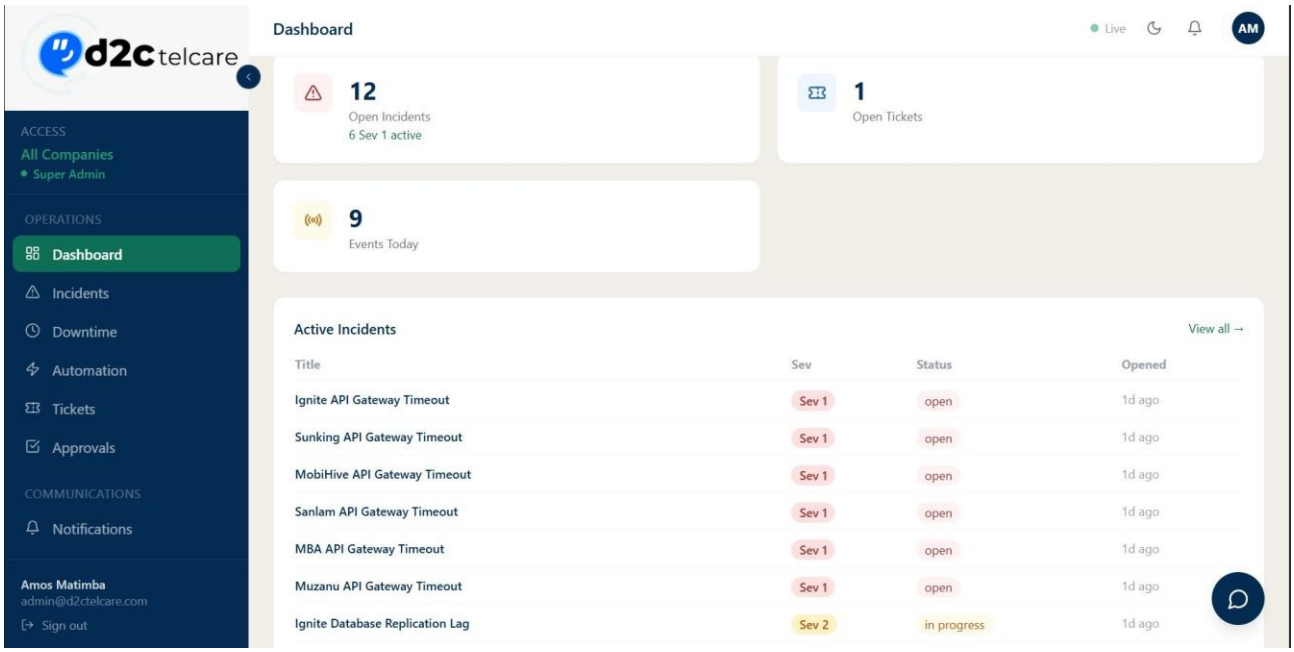

Figure 6.1b — Dashboard scrolled view showing the full Active Incidents table with Sev 1 and Sev 2 badges

6.2 Incidents — Route: /incidents — Access: operator, manager, admin

The core module. Incidents have three statuses: open, in_progress, resolved. Severity runs from Sev 1 (Critical) to Sev 4 (Low).

Incident Workflow

- Operator or manager creates an incident — all process users receive a New Incident email.
- A manager assigns the incident — the assignee receives a personal Assigned to You email.
- The incident is marked In Progress — all process users receive an In Progress email.
- If the incident stays open and unassigned for 30 minutes, the background worker sends a singleUnattended reminder email.
- The incident is resolved — all process users receive a Resolved email with duration.

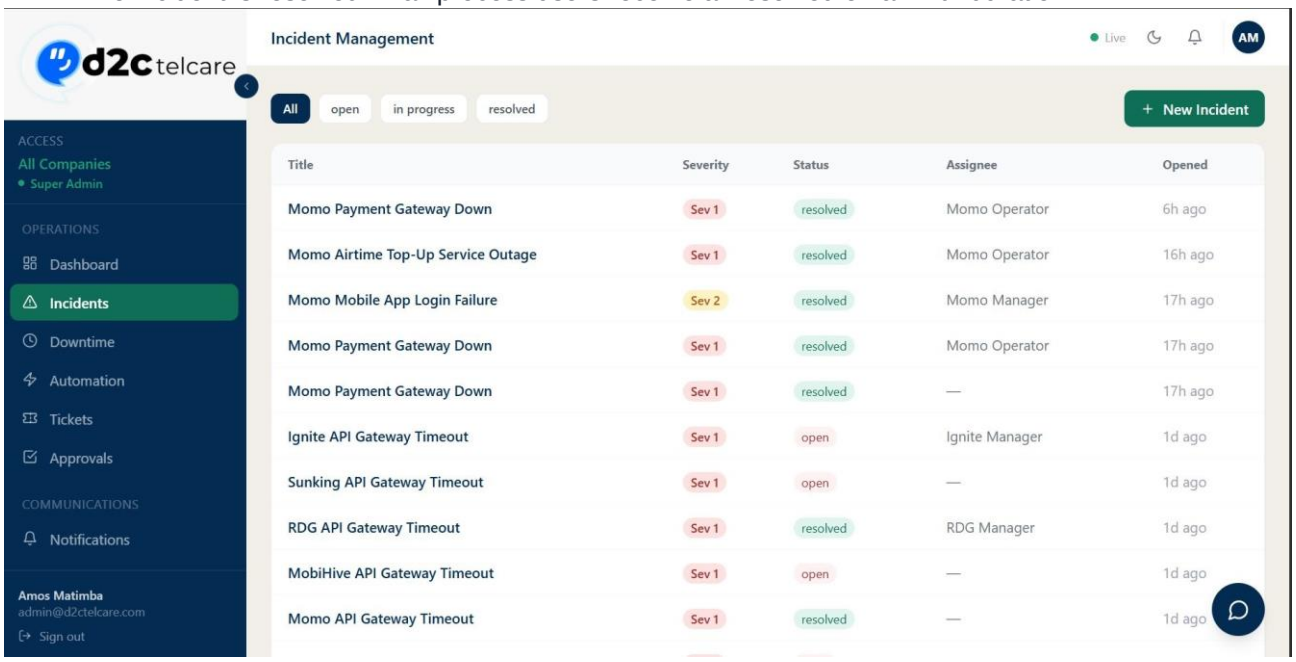

Figure 6.2a — Incident Management list: Sev 1 (red) and Sev 2 (amber) badges, resolved (green) and open status

Figure 6.2b — Incidents list continued: Sev 2 in progress and Sev 4 resolved entries

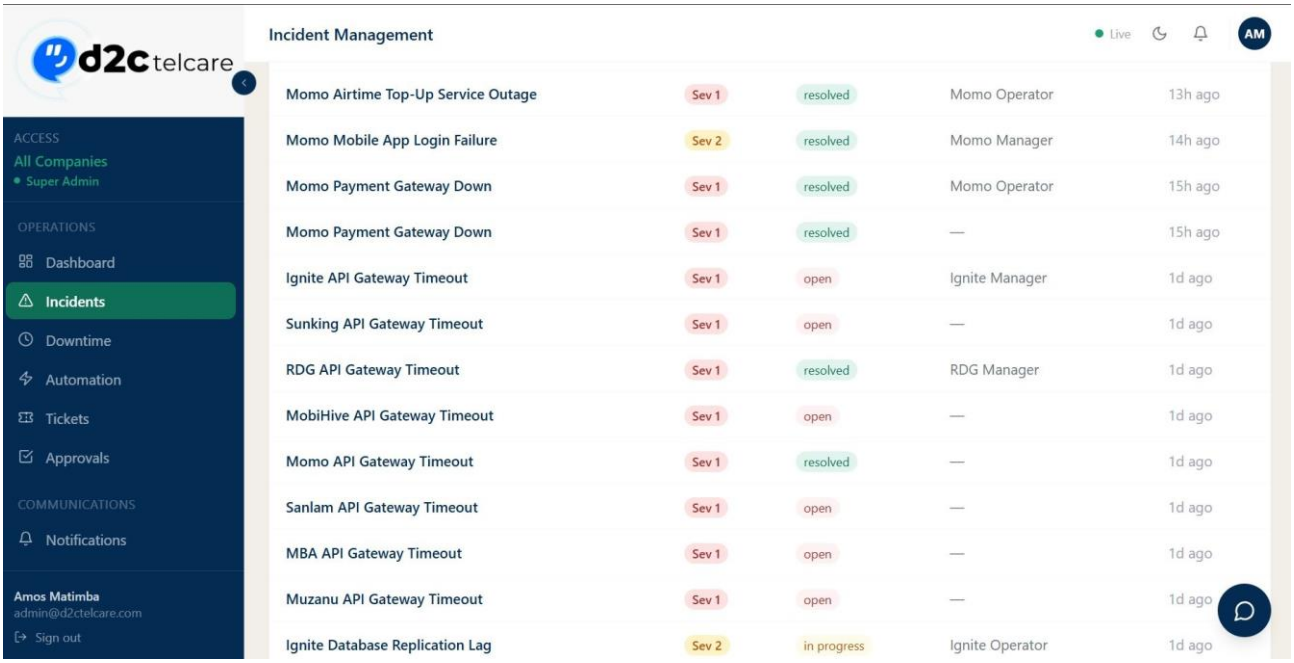

Figure 6.2c — Incidents list compact sidebar mode showing cross-process incidents

Incident Detail Modal

Each incident card is clickable. The detail modal allows editing the description inline, changing the assignee, adding Root Cause Analysis (RCA), and transitioning the status.

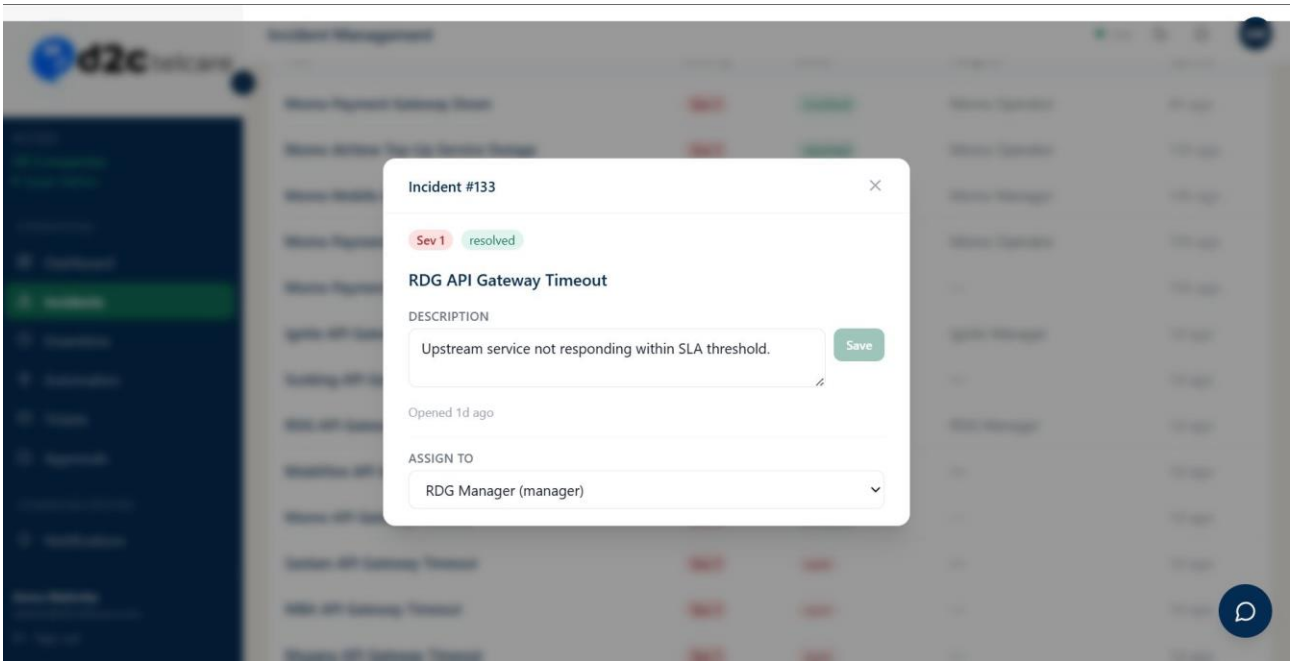

Figure 6.2d — Incident detail modal (Incident #133 — RDG API Gateway Timeout, resolved)

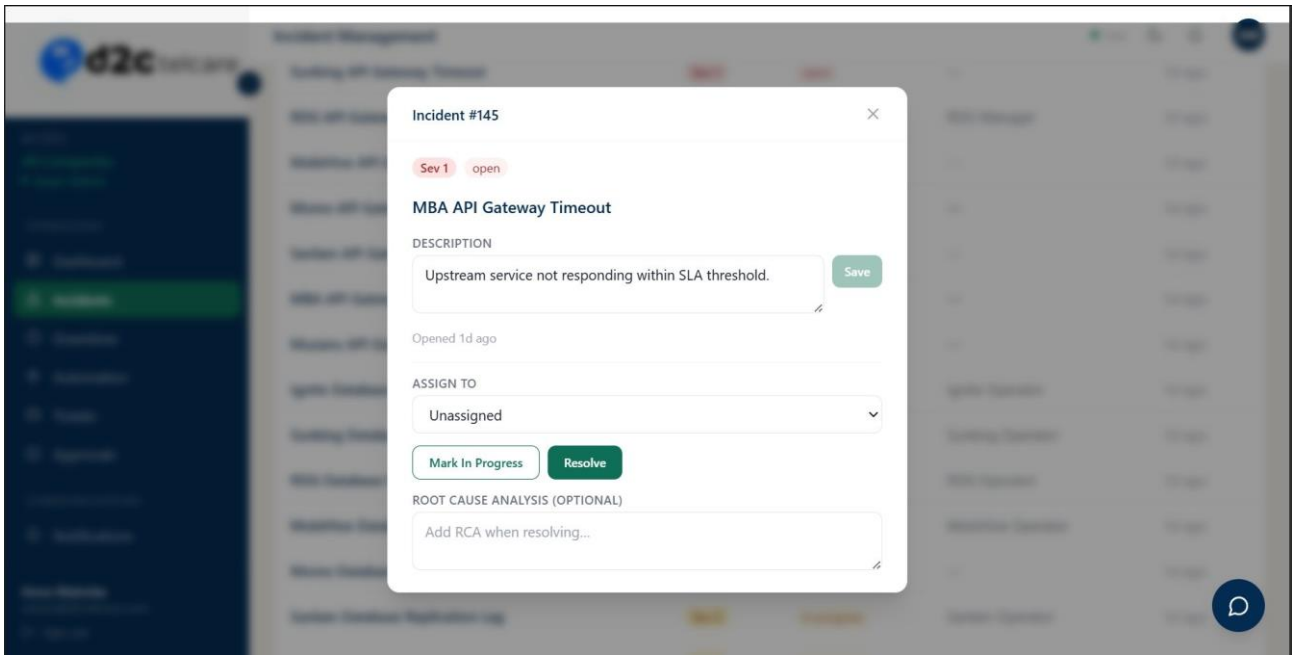

Figure 6.2e — Incident detail modal (Incident #145 — MBA API Gateway Timeout, open) showing Mark In Progress, Resolve, and RCA textarea

6.3 Downtime — Route: /downtime — Access: operator, manager, admin

Logs service outage events. Fields: Start time, Services Affected (JSON array), Impact Level (low / medium / high). When logged, managers and admins receive a Downtime Reported email. When ended, they receive a Downtime Resolved email with duration.

Figure 6.3 — Downtime Tracking page showing impact level badges (low, medium, high), Ongoing rows with End button, and Ended rows

6.4 Ticketing — Route: /tickets — Access: operator and above

Four ticket types: Incident, Change, Access, Report. Three statuses: open, in_progress, closed. Supports status filter tabs and assignee selection.

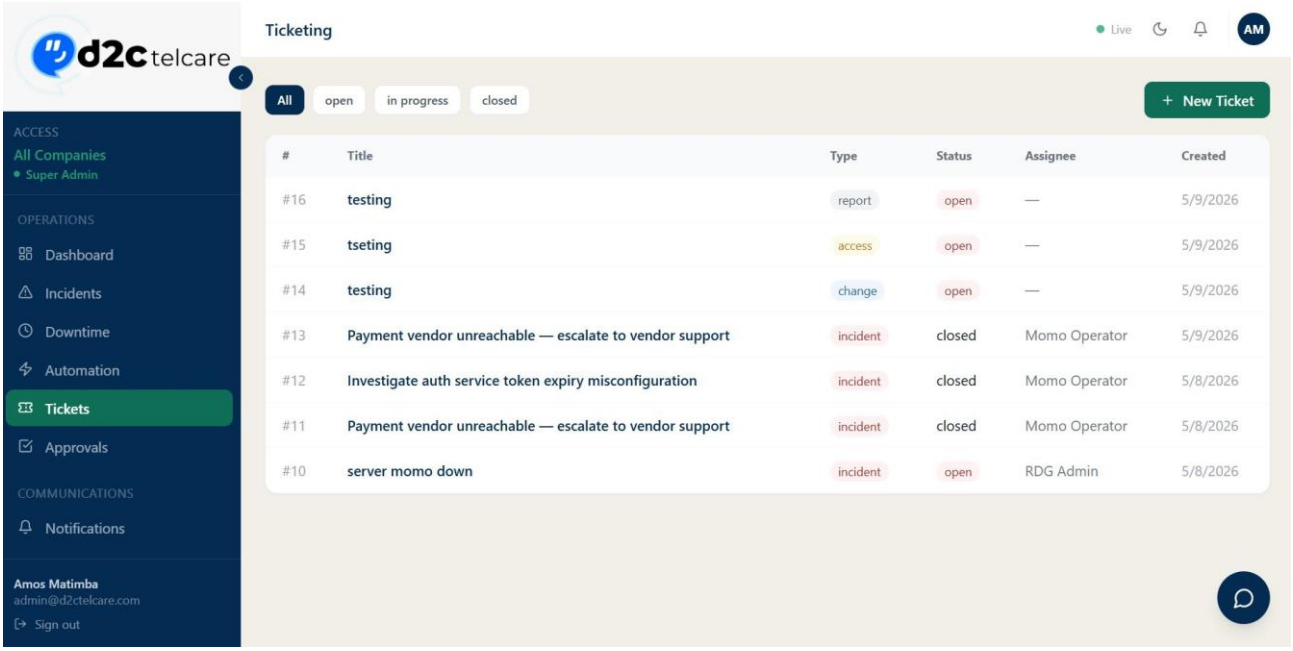

Figure 6.4 — Ticketing page showing all ticket types (incident, access, change, report) with status badges (open, closed) and filter tabs

6.5 Approvals — Route: /approvals — Access: manager, admin

Approval requests for change or access actions. Statuses: pending, approved, rejected. Submitters receive an email when decided; managers/admins receive email when a new request is pending.

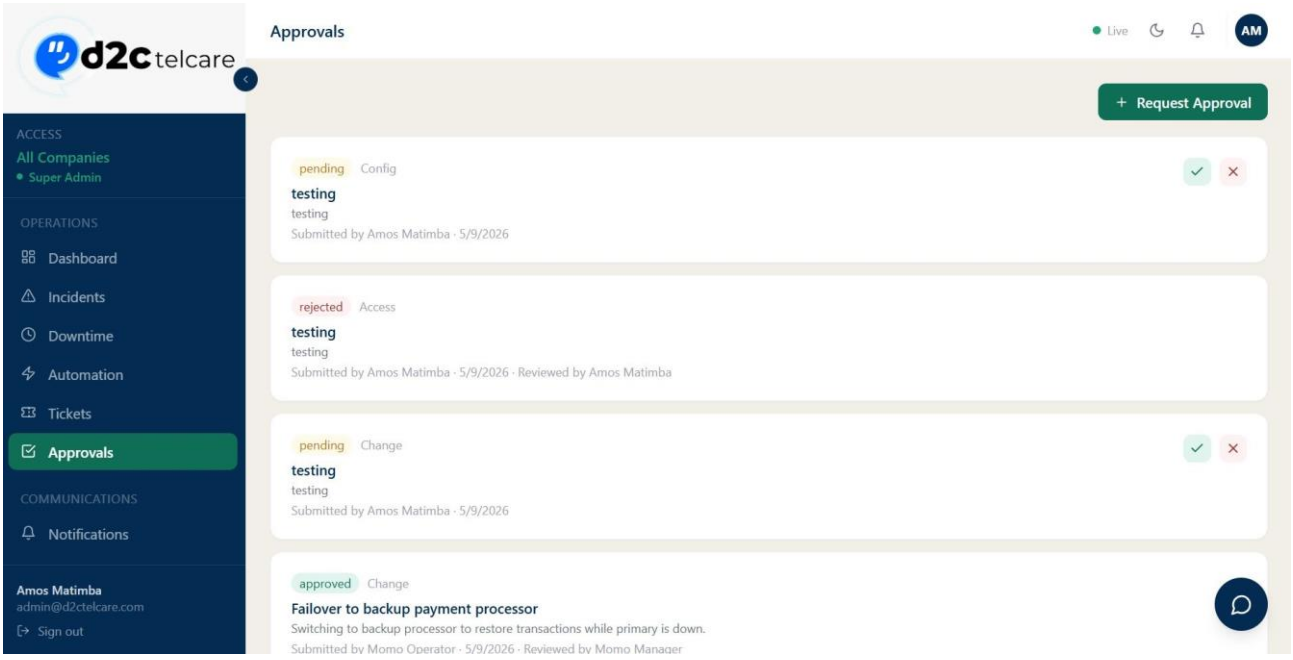

Figure 6.5 — Approvals page showing pending requests with approve/reject buttons, and rejected/approved entries below

6.6 Shift Handovers — Route: /handovers — Access: operator and above

Shift handover records between team members. When submitted, the recipient receives a handover email.

When acknowledged, the original submitter receives a confirmation.

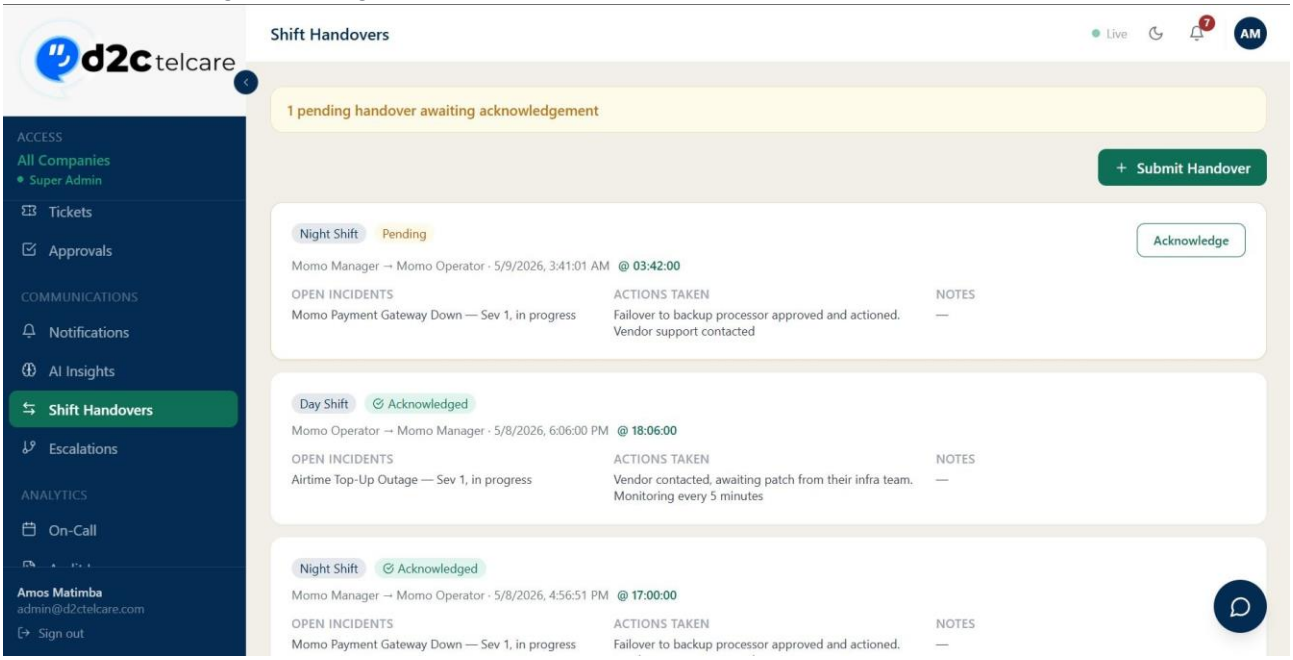

Figure 6.6a — Shift Handovers list: Night Shift (Pending) and Day/Night Shift (Acknowledged) entries

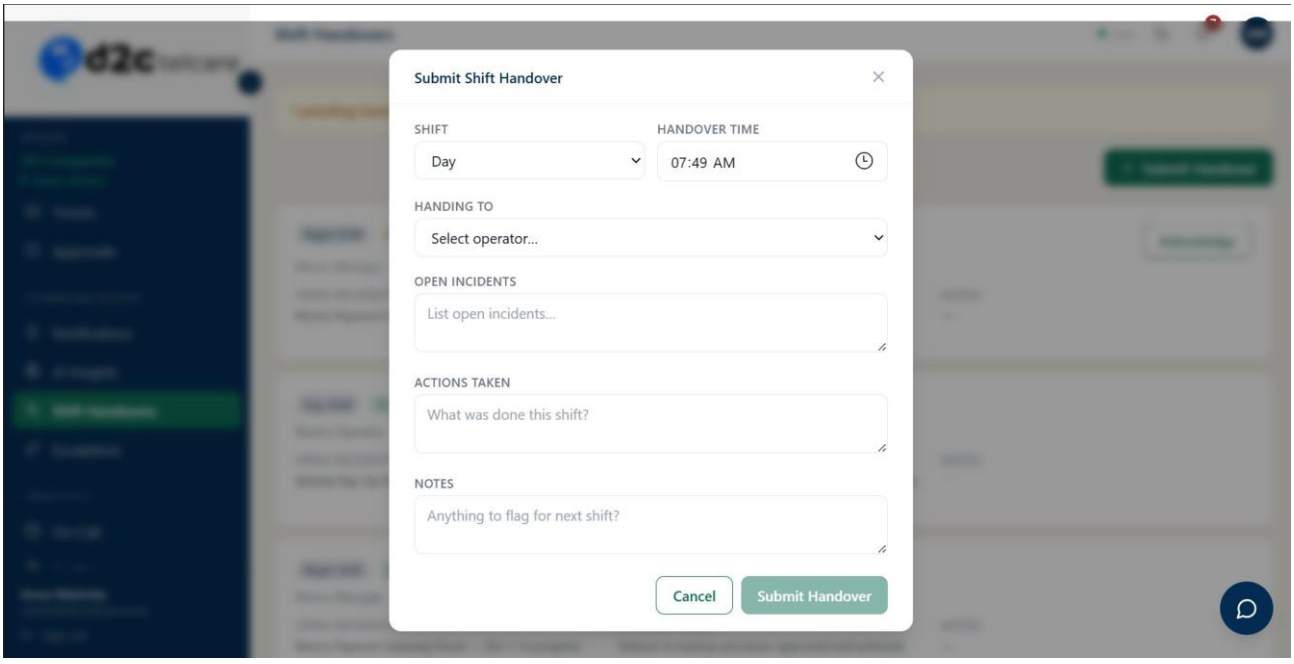

Figure 6.6b — Submit Shift Handover modal with Shift, Handover Time, Handing To, Open Incidents, Actions Taken, and Notes fields

6.7 Escalations — Route: /escalations — Access: manager, admin

Manages escalation chains — named sequences of tiers that define who gets notified and when during an unresolved incident. Each chain has a name, an optional trigger severity filter, and one or more numbered tiers. Each tier specifies: delay in minutes, notification channel (e.g., email), and an assigned user (with on-call fallback if no user is set).

When a new chain is created, tiers can be added with the "+ Add Tier" option. The background worker polls every 60 seconds and fires tiers that are due. Execution history prevents duplicate firings.

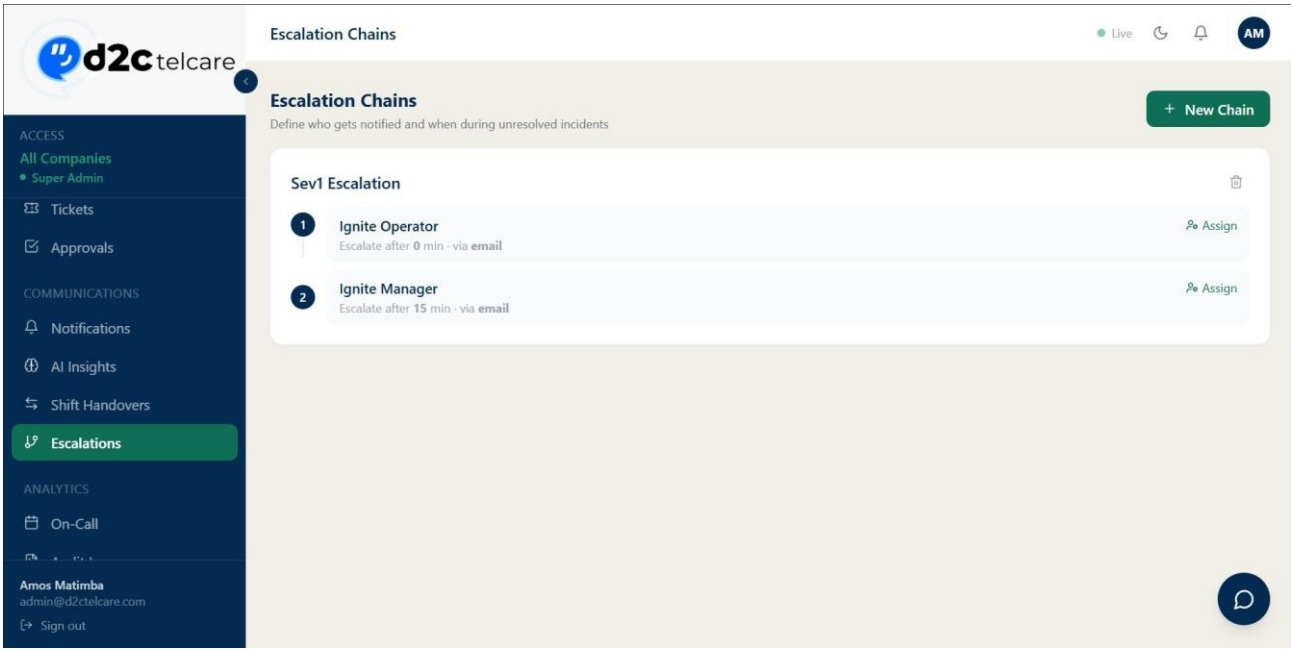

Figure 6.7a — Escalation Chains list showing the Sev1 Escalation chain: Tier 1 (Ignite Operator, 0 min via email) and Tier 2 (Ignite Manager, 15 min via email)

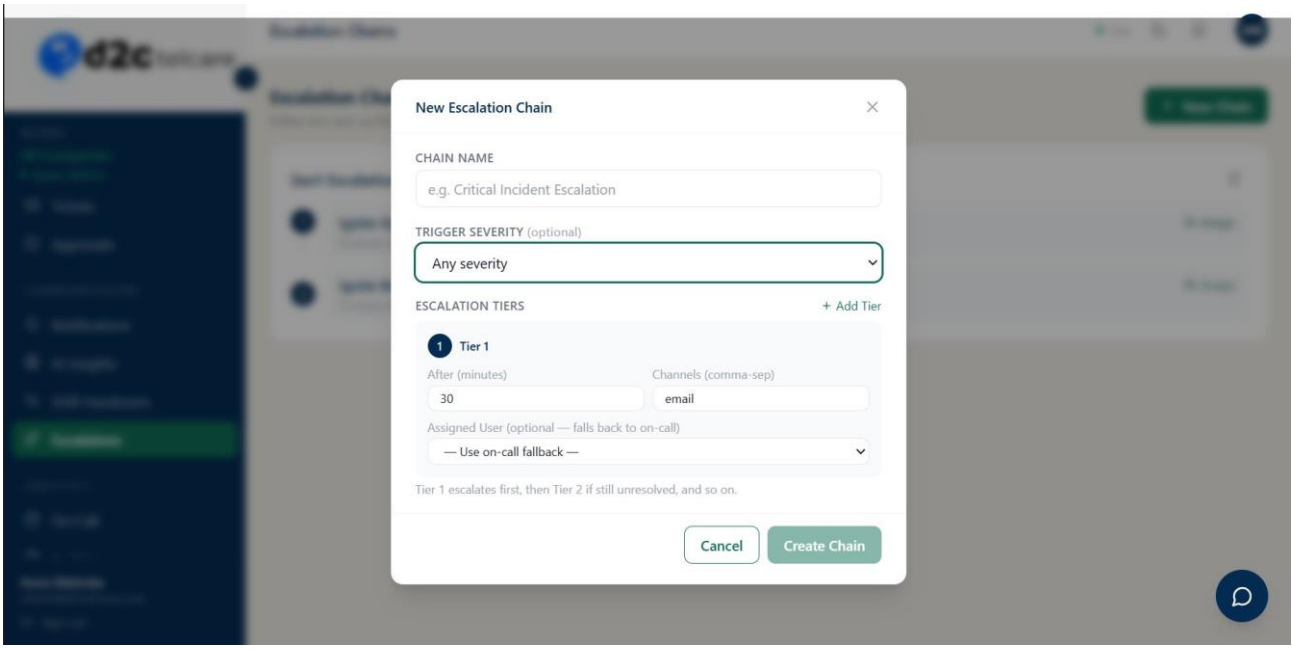

Figure 6.7b — New Escalation Chain modal showing Chain Name, Trigger Severity dropdown, and Tier 1 configuration with After (minutes), Channels, and Assigned User fields

6.8 Automation — Route: /automation — Access: manager, admin

Rule-based automation engine. Each rule has a name, a trigger, and an action. Rules are listed with their execution count and an Active status badge.

Trigger types

- Severity unresolved after N minutes — fires when an incident of a specified severity remains openbeyond a configured time threshold
- Custom trigger — configurable for other conditions

Actions

- Escalate via chain — links to an existing escalation chain which then fires its tiers

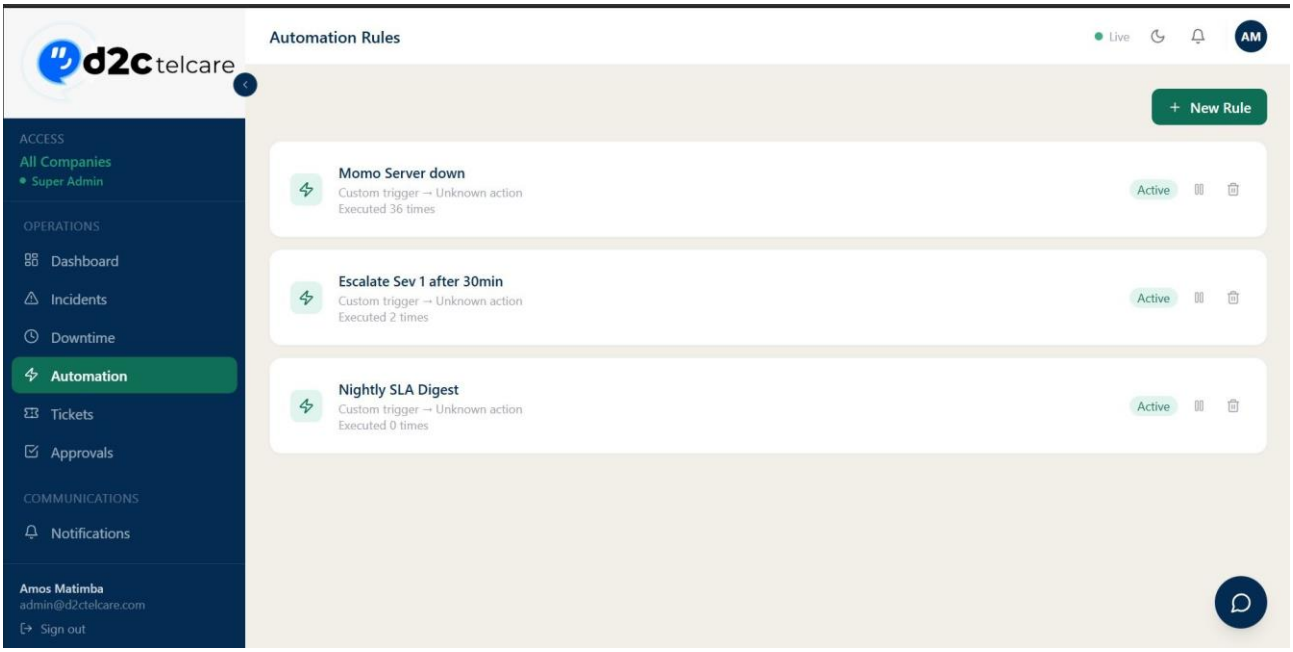

Figure 6.8a — Automation Rules page: Momo Server down (36 executions), Escalate Sev 1 after 30min (2 executions), Nightly SLA Digest (0 executions) — all Active

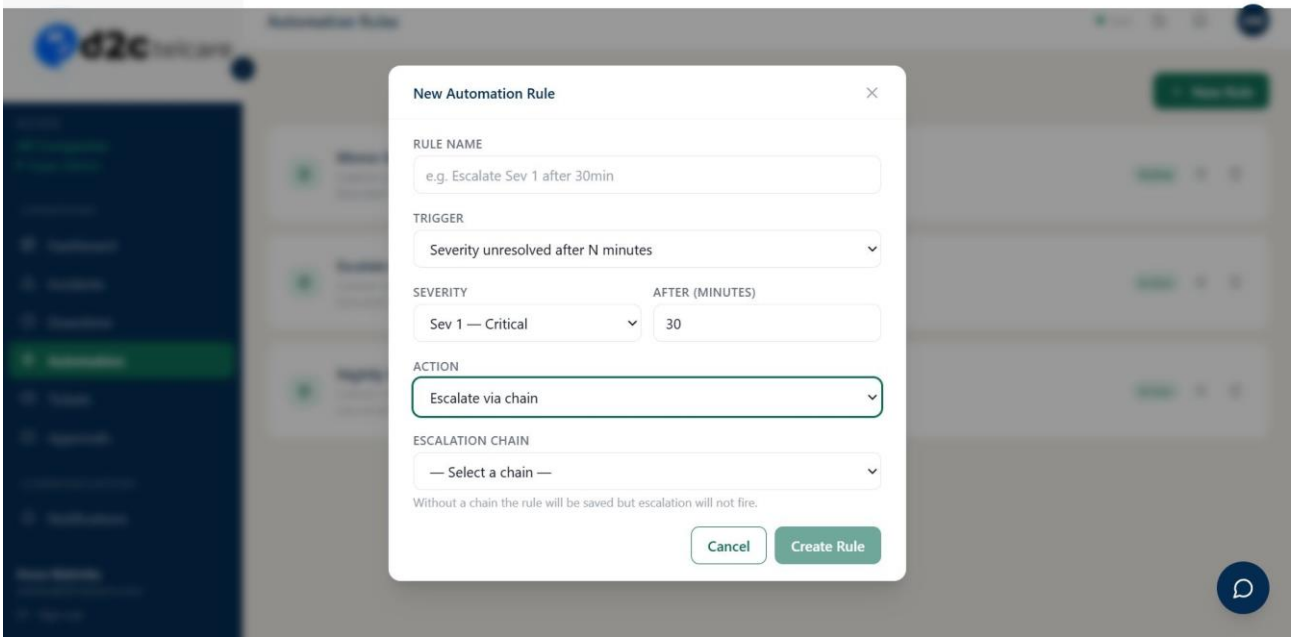

Figure 6.8b — New Automation Rule modal: Trigger set to "Severity unresolved after N minutes", Severity = Sev 1 Critical, After = 30 min, Action = Escalate via chain

6.9 Notifications — Route: /notifications — Access: All roles

Client Notifications is the ops-to-client broadcast channel. The list displays: Direction badge (To Client), Subject, Sent By, number of Recipients, Bounce count, and Sent At timestamp. Bounce tracking shows how many recipients did not receive the message.

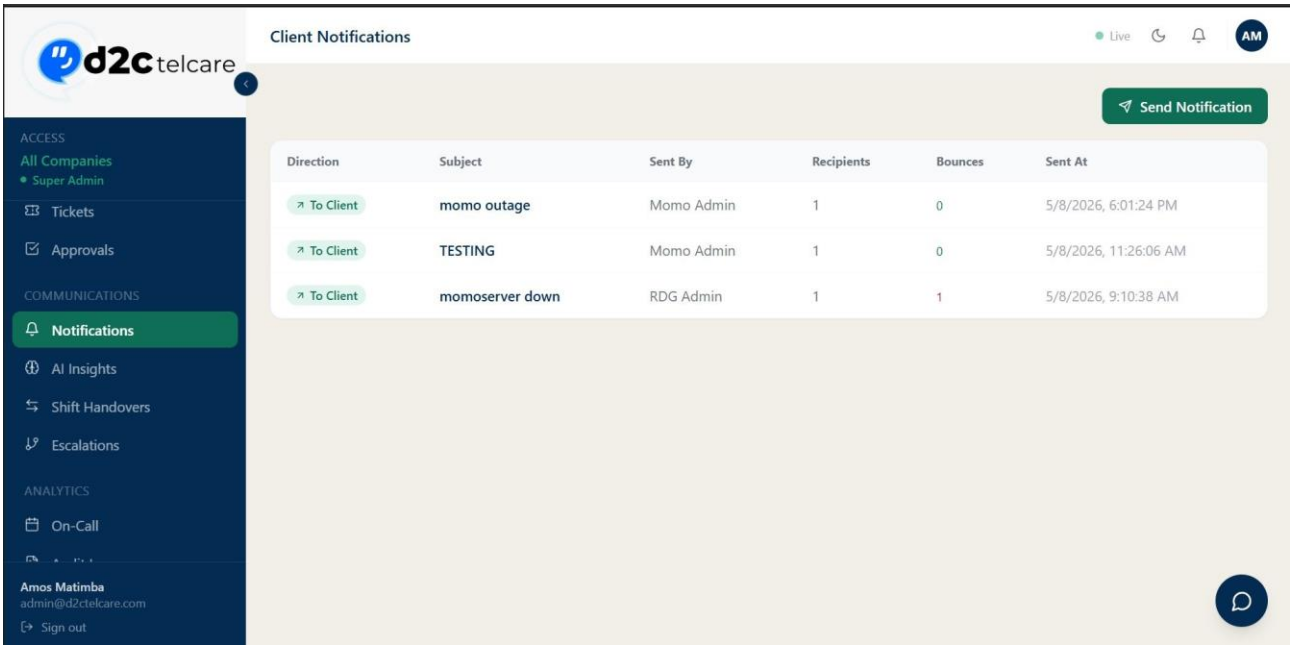

Figure 6.9a — Client Notifications page showing the table with Direction, Subject, Sent By, Recipients, Bounces, and Sent At columns

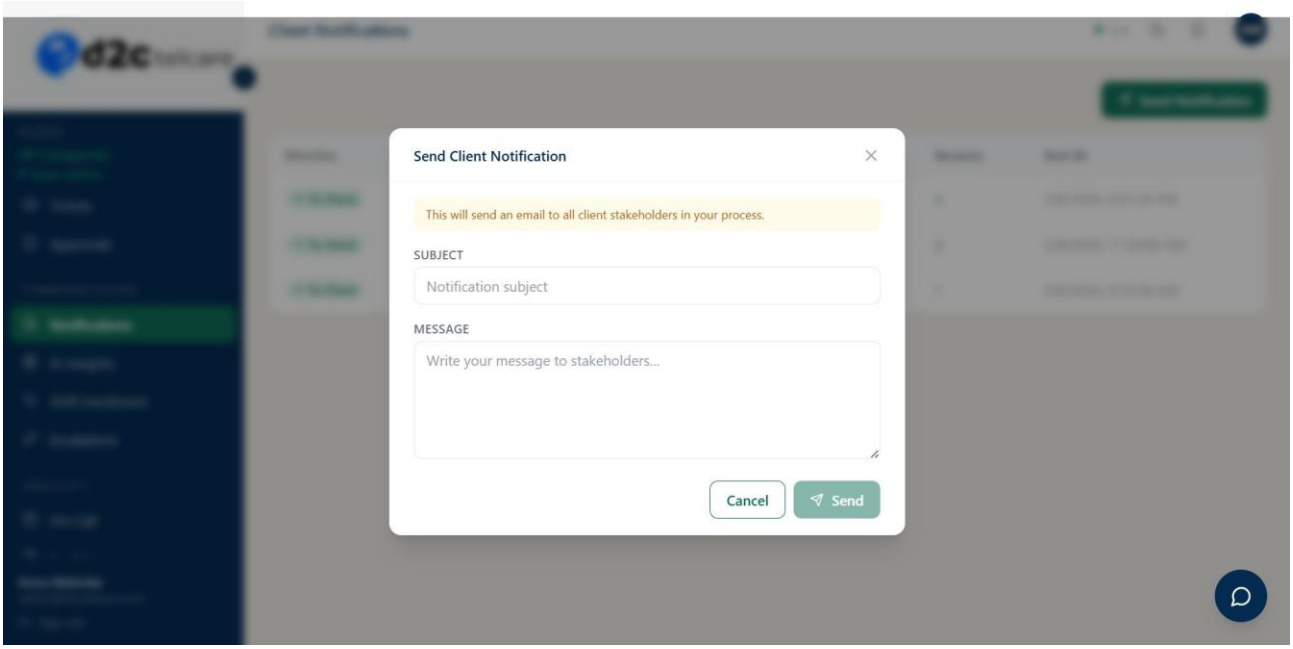

Figure 6.9b — Send Client Notification modal with Subject and Message fields, and warning: "This will send an email to all client stakeholders in your process"

6.10 AI Insights — Route: /ai-insights — Access: All roles

Submits incident data to OpenAI GPT-4o for root cause analysis, recommended actions, similar past incidents, and prevention tips. Also includes an AI Assistant chat tab for general platform questions.

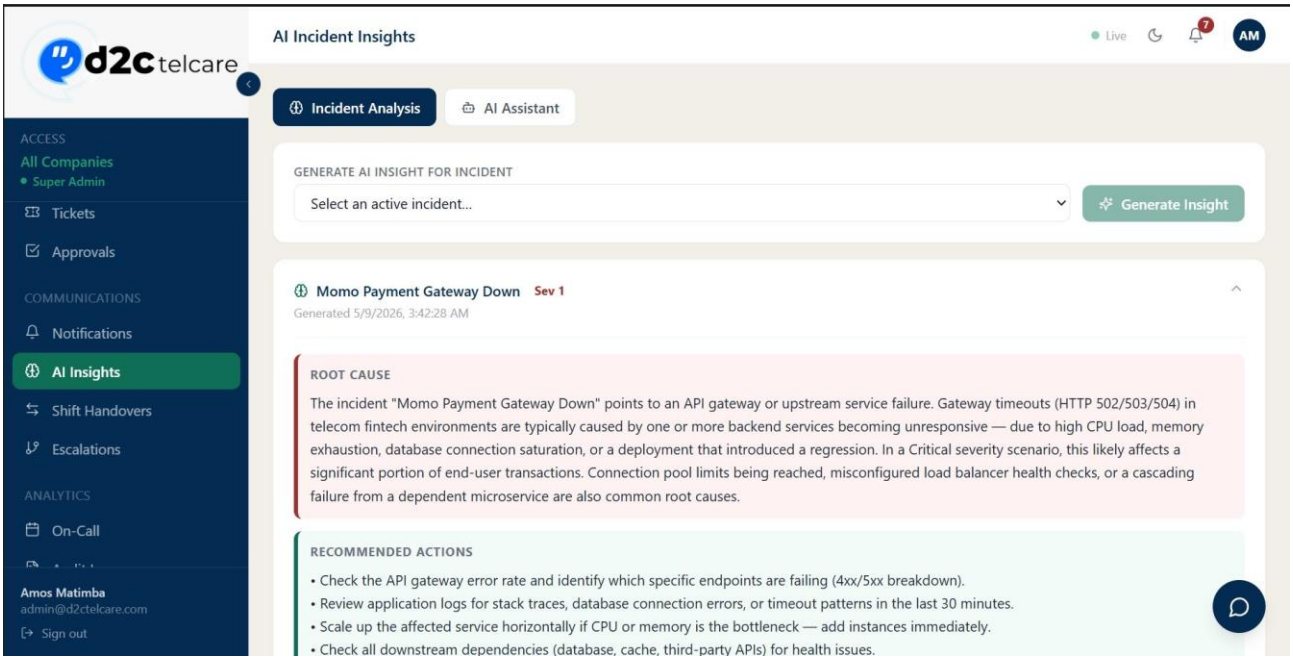

Figure 6.10a — AI Insights: Root Cause and Recommended Actions panels for Momo Payment Gateway Down (Sev 1)

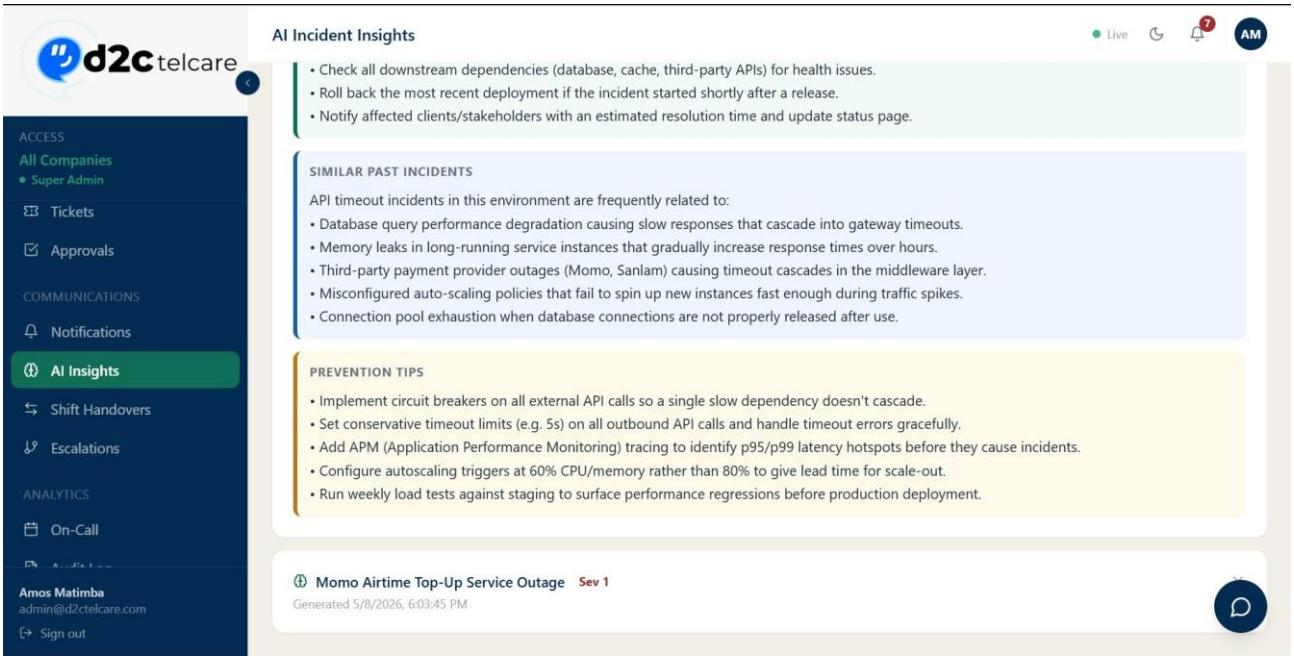

Figure 6.10b — AI Insights: Similar Past Incidents and Prevention Tips panels

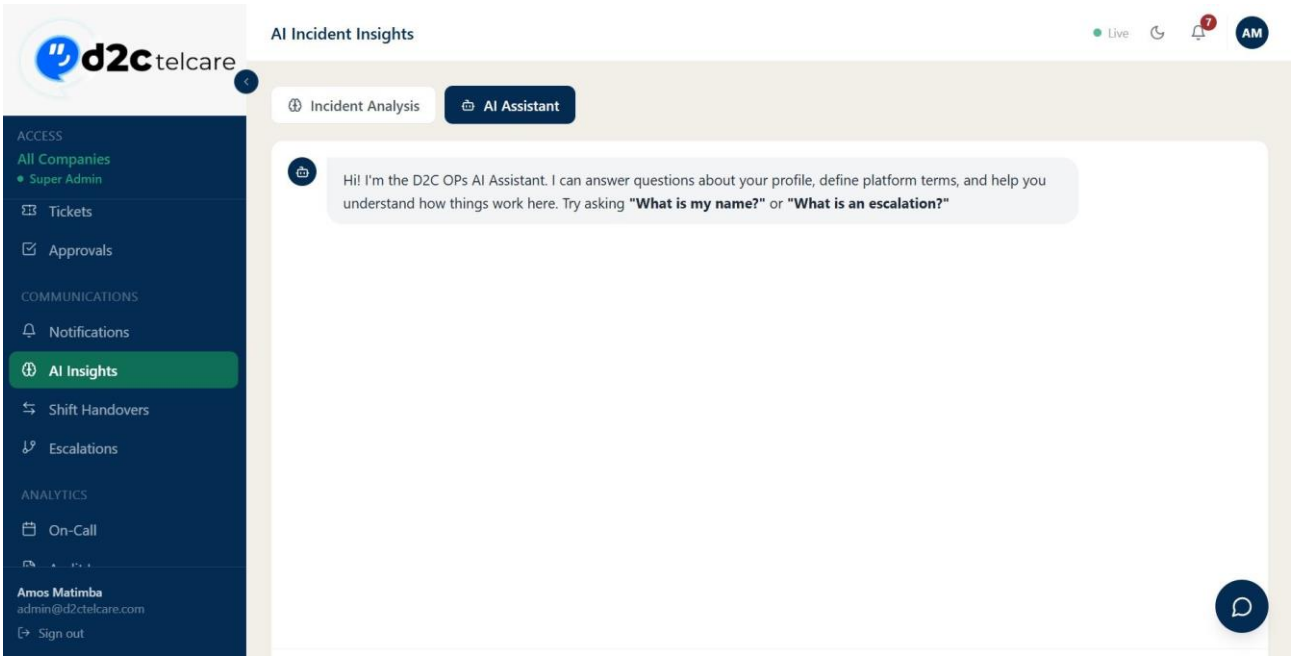

Figure 6.10c — AI Assistant tab with the D2C OPs AI Assistant chat interface

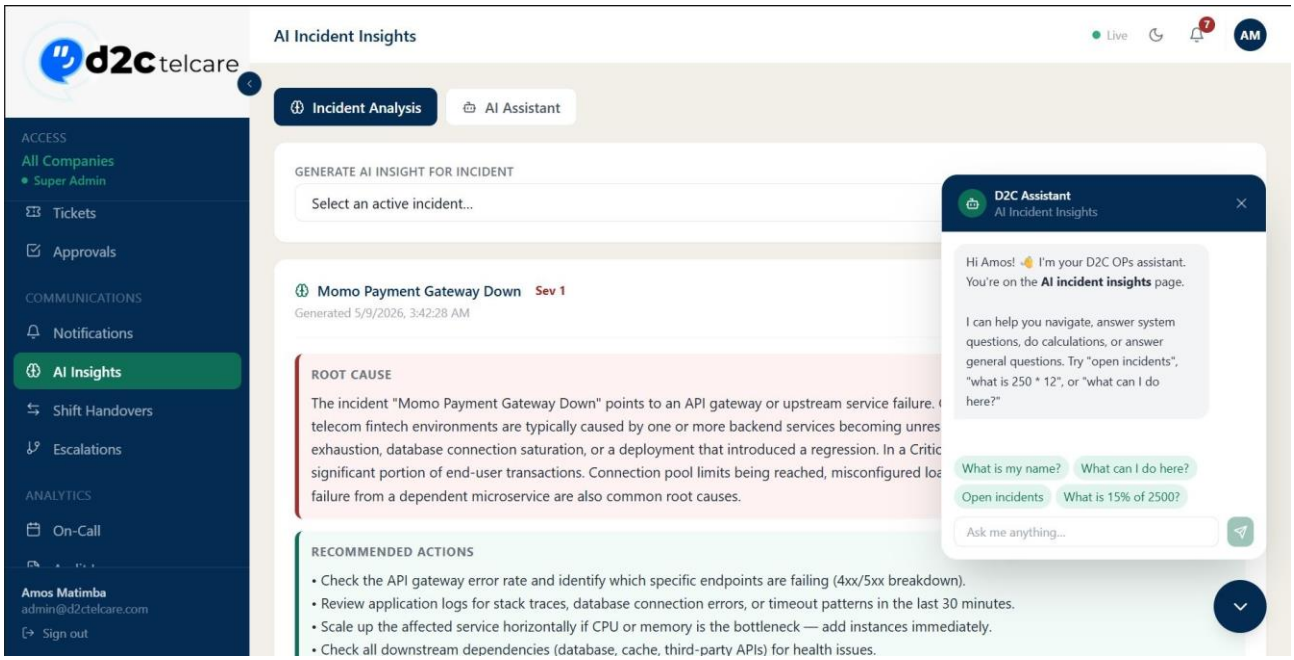

Figure 6.10d — AI Assistant chat popup showing context-aware suggestions and quick-action prompts

6.11 On-Call — Route: /oncall — Access: manager/admin to manage; all toview

On-Call Scheduling assigns team members to on-call duties by date. The list shows: Date, Person, Role badge (primary / backup / escalation), and Process. The on-call schedule integrates with the escalation engine — when a chain tier has no assigned user, the system falls back to the on-call primary for that process on the current date.

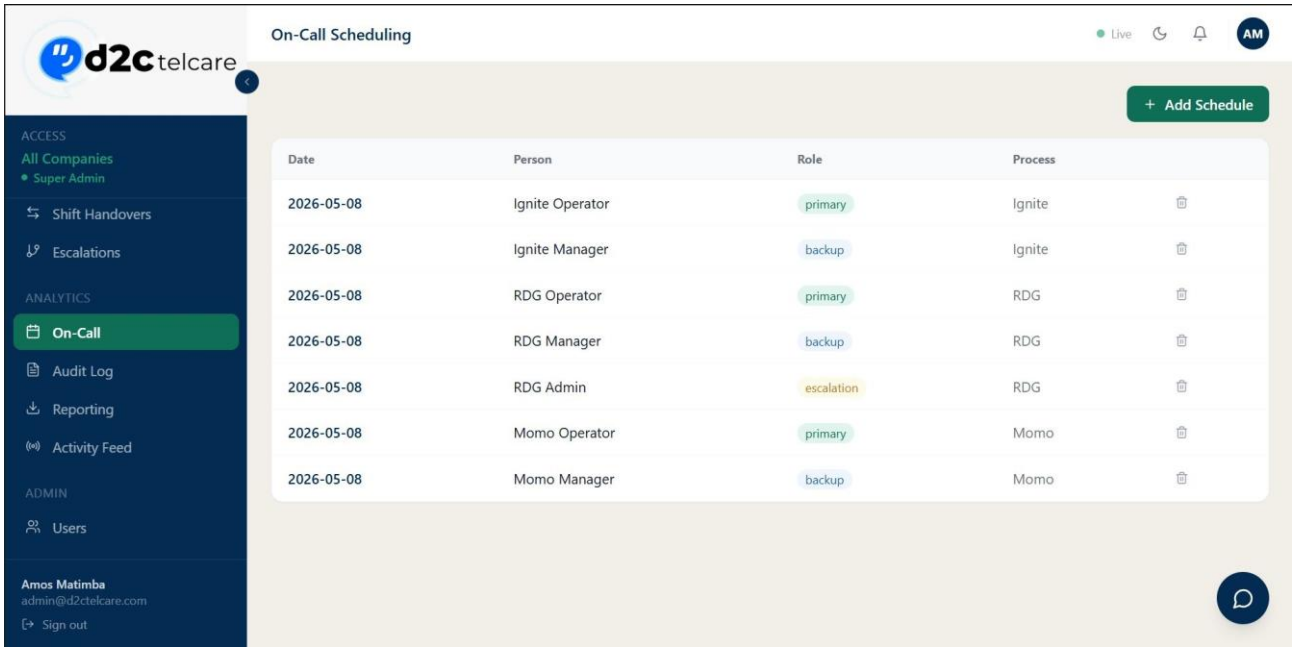

Figure 6.11a — On-Call Scheduling list showing Date, Person, Role (primary/backup/escalation badges), and Process across multiple processes

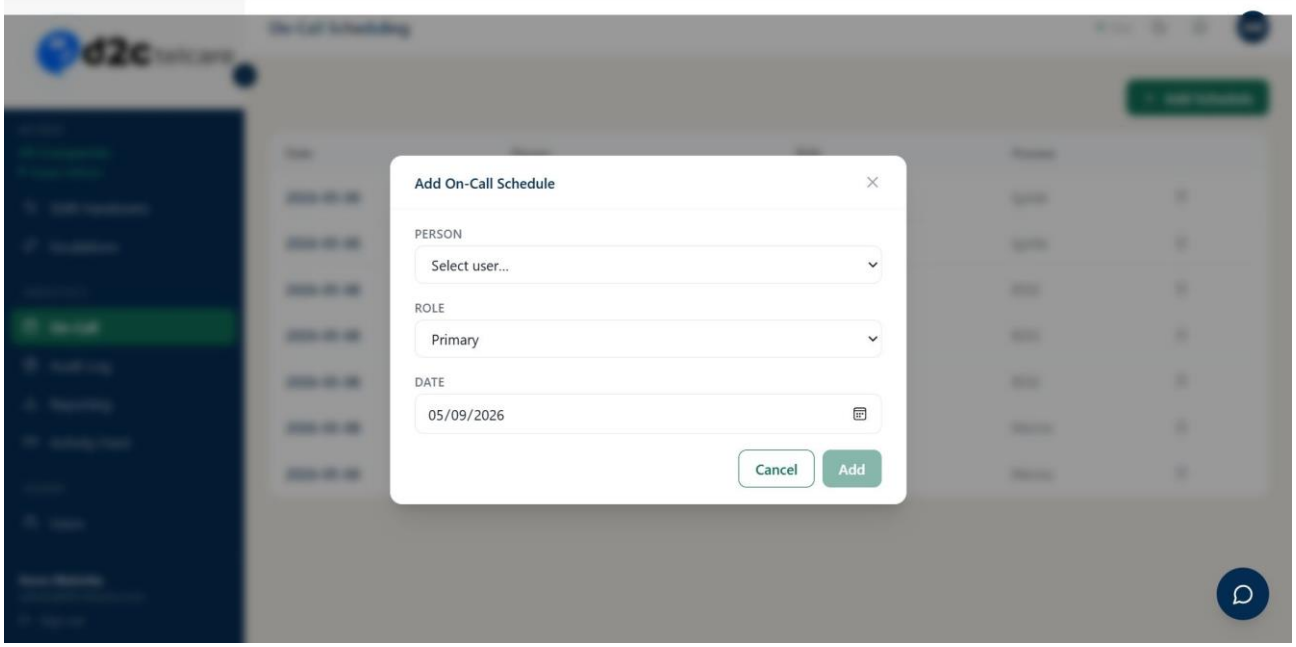

Figure 6.11b — Add On-Call Schedule modal showing Person dropdown, Role (Primary), and Date picker

6.12 Reporting — Route: /reporting — Access: manager, admin

Reporting & Export provides three report types — Incident Report, Downtime Summary, and Audit Log Export — each with its own date range filter and independent CSV and PDF export buttons. The PDF output is a branded D2C OPs report with a navy header bar and paginated data table.

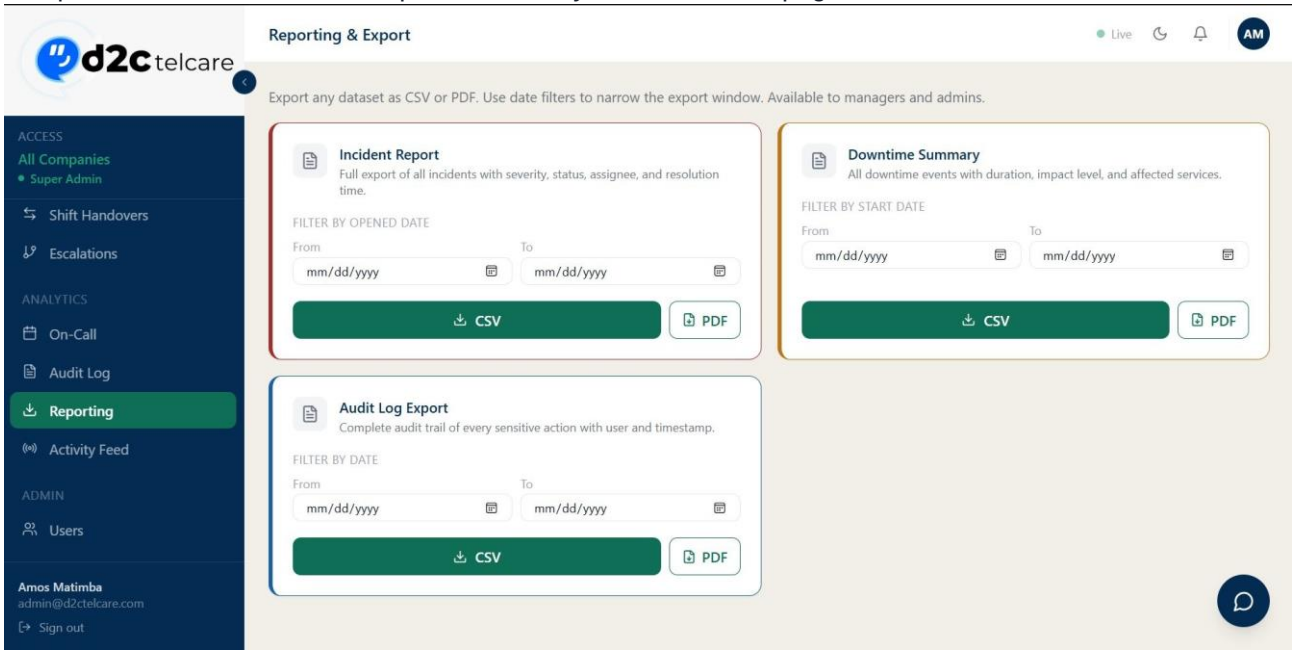

Figure 6.12a — Reporting & Export page: three report cards (Incident Report, Downtime Summary, Audit Log Export) with date filters and CSV/PDF buttons

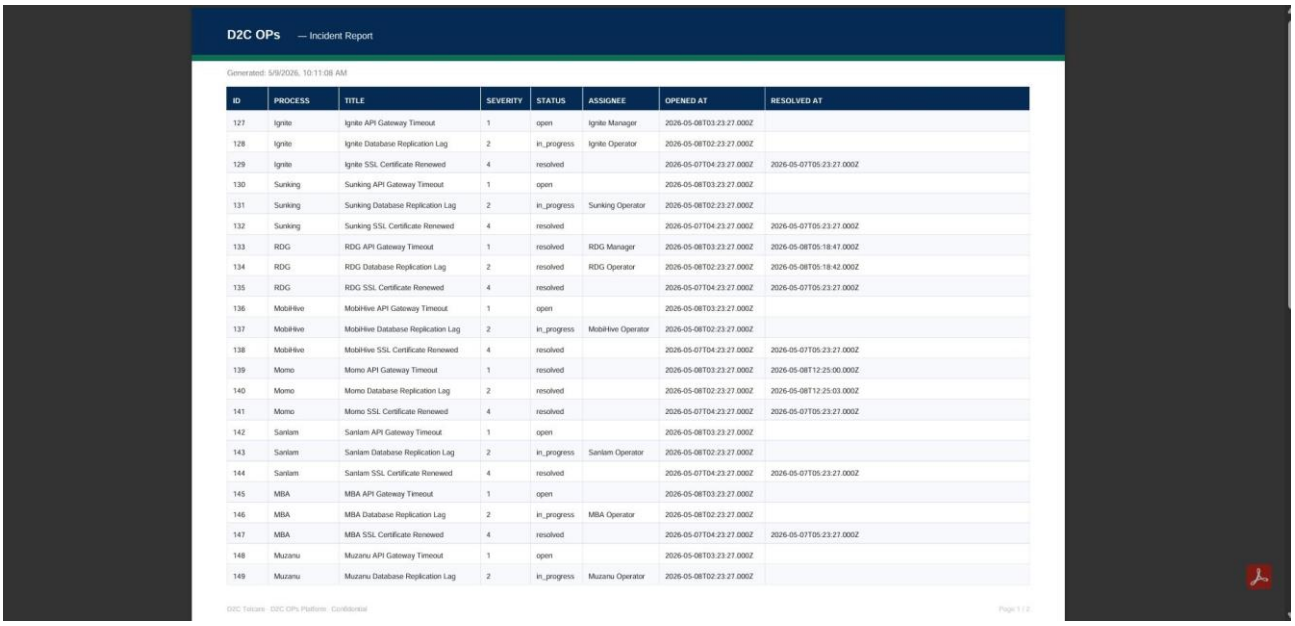

Figure 6.12b — Generated Incident Report PDF showing ID, Process, Title, Severity, Status, Assignee, Opened At, Resolved At columns with data from all 8 processes

6.13 Audit Log — Route: /audit-log — Access: admin, super_admin

Immutable chronological log of every action on the platform. Records who did what, to which record, and when. Supports filtering by action type and date range.

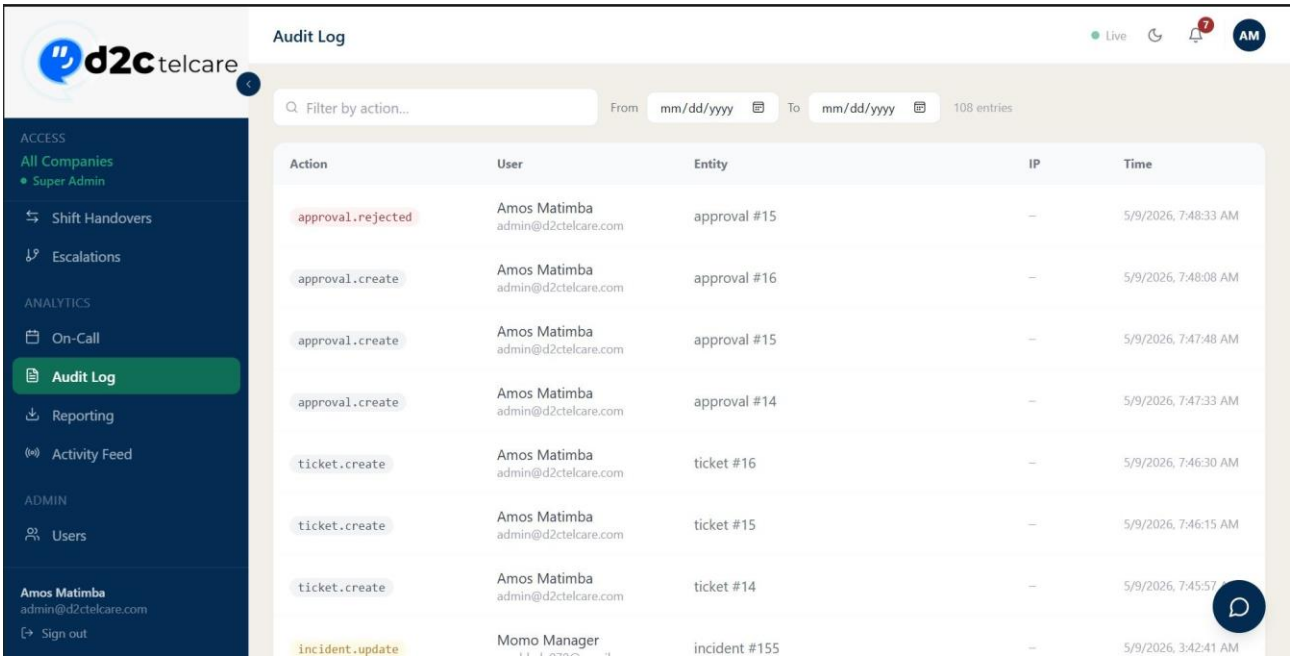

Figure 6.13a — Audit Log showing 108 entries with colour-coded action badges (approval.rejected, ticket.create, etc.), user, entity, and timestamp

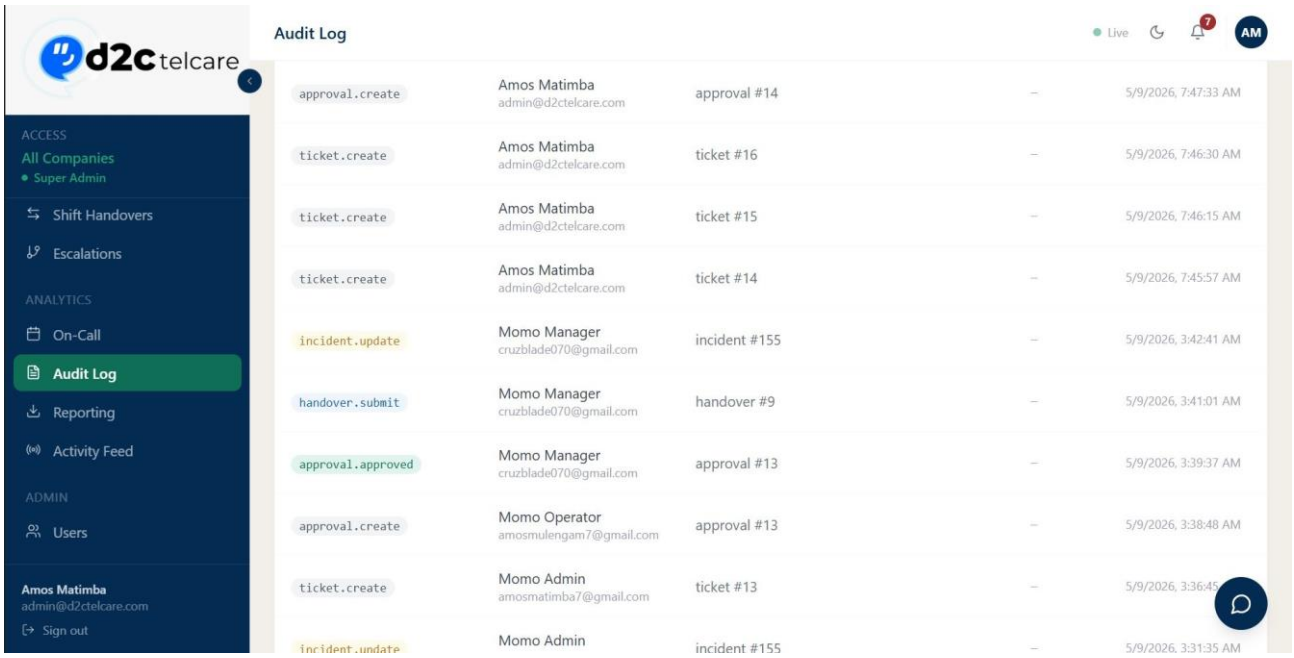

Figure 6.13b — Audit Log continued: incident.update, handover.submit, and approval.approved entries

6.14 Activity Feed — Route: /activity — Access: All roles

The Live Activity Feed is a real-time event stream that auto-refreshes every 15 seconds. Events are displayed in the format entity → action with the acting user's name, record number, process, and a relative timestamp. Controls include an event type filter, Pause button, and manual Refresh button.

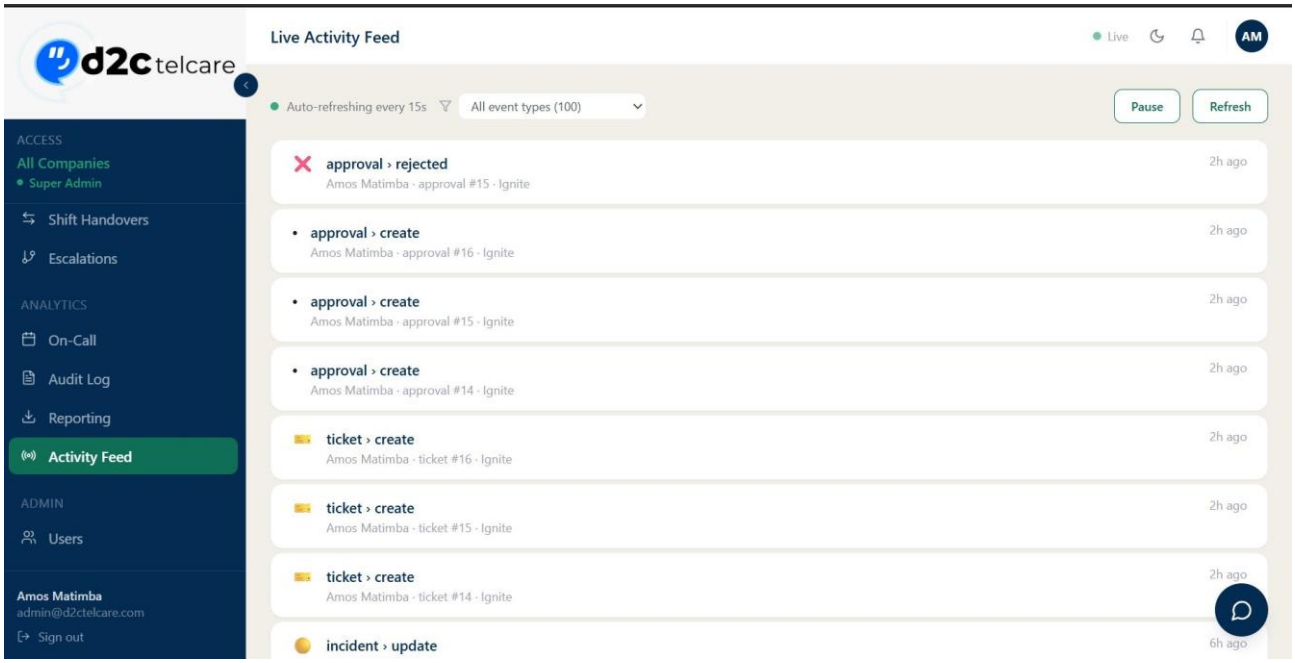

Figure 6.14 — Live Activity Feed showing auto-refresh indicator, event type filter (100 events), Pause/Refresh buttons, and events: approval-rejected, approval-create, ticket-create, incident-update

7. Email Notification System

All emails are sent from "D2C OPs Platform" using a shared Nodemailer singleton transporter with Gmail SMTP.

Performance Design

- Lazy singleton — the transporter is created once on first send, not per email.
- Connection pool — pool: true, maxConnections: 5 keeps SMTP connections alive.
- Parallel bulk sends — Promise.all() fires all emails in a batch simultaneously.
- TLS auto-detect — port 587 uses STARTTLS, port 465 uses direct TLS.

Email Event Matrix

Event	Recipient	Trigger
Incident Created	All process users	POST /incidents
Incident Assigned	Assignee only	PATCH /incidents/:id — assigned_to changes
Incident Unattended	All process users	Background worker — open 30+ min, no reminder sent
Incident In Progress	All process users	PATCH — status → in_progress
Incident Resolved	All process users	PATCH — status → resolved
Event	**Recipient**	**Trigger**
Downtime Reported	Process managers & admins	POST /downtime
Downtime Resolved	Process managers & admins	PATCH /downtime/:id — ended_at set
Handover Submitted	Recipient	POST /handovers
Handover Acknowledged	Submitter	PATCH /handovers/:id
Approval Submitted	Managers & admins	POST /approvals
Approval Decided	Submitter	PATCH /approvals/:id
Welcome Email	New user	POST /users
Password Reset OTP	Requesting user	POST /auth/forgot-password
Email Changed	Old & new address	PATCH /users/:id/email

Email Template Example

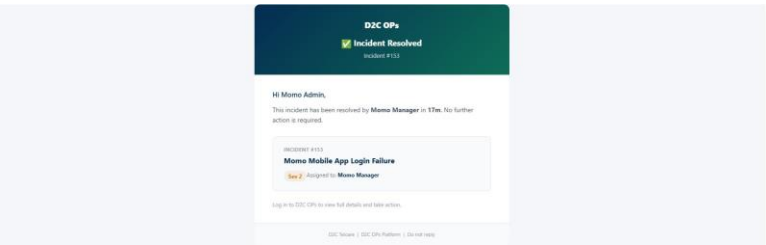

Figure 7.1 — Incident Resolved email: D2C OPs header, incident title, Sev 2 badge, assignee (Momo Manager), resolved in 17m

8. Authentication & Security

Registration & Login Flow

- User submits email — server generates a 6-digit OTP, stores in verification_codes, sends via email.
- User submits OTP + password — server validates OTP (15-min expiry), hashes password withbcrypt, creates user record.
- Login — server validates credentials, issues signed JWT containing userId, processId, role,processSlug.
- Frontend stores JWT in localStorage under the Zustand auth store key (d2c-ops-auth).
- All API requests include Authorization: Bearer header.
- authenticate middleware validates and decodes the token on every protected route.

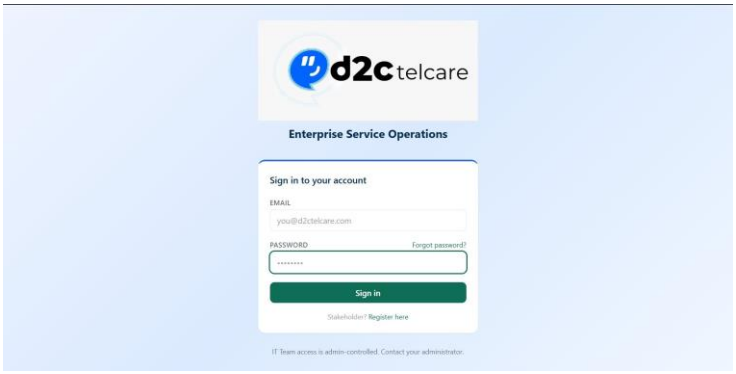

Figure 8.1 — Login page: D2C Telcare branding, Enterprise Service Operations subtitle, email/password fields, and Sign In button

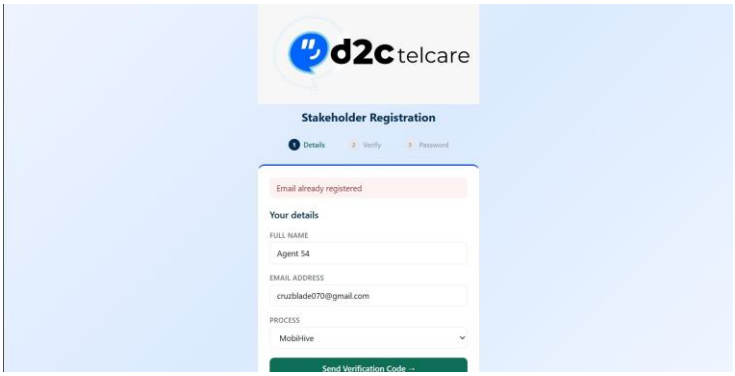

Figure 8.2 — Stakeholder Registration (Step 1 — Details): 3-step flow (Details → Verify → Password)

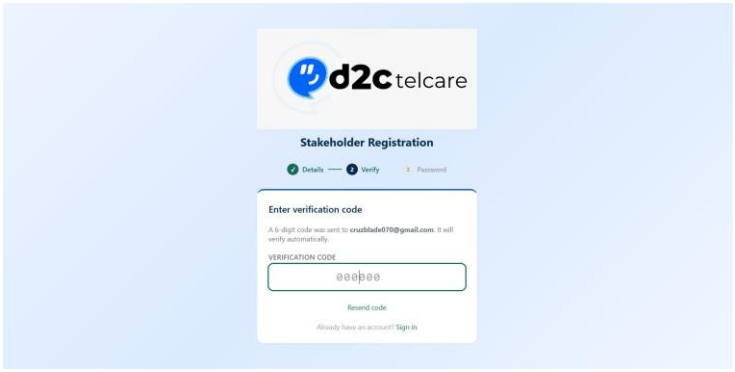

Figure 8.3 — Registration Step 2 (Verify): 6-digit verification code entry with Resend code option

Security Controls

Control	Detail
Password Hashing	bcrypt, salt rounds: 10
JWT Algorithm	HS256, configurable via JWT_SECRET env var
Public Routes	/auth/login, /auth/register, /auth/verify-code, /auth/forgot-password
Control	**Detail**
Role Enforcement	requireRole(...allowedRoles) middleware — returns 403 if role not in allowed list
Process Isolation	All queries include process_id filter; super_admin bypasses the filter
OTP Codes	Single-use (used flag) and expire after 15 minutes

9. Multi-Process (Multi-Tenant) Design

Each client business process (Ignite, Sunking, RDG, etc.) is a separate row in the processes table. Every data table has a process_id foreign key.

Data Isolation Rule

Every API query is filtered by process_id derived from the authenticated user's JWT. No user can read or modify another process's data unless they hold the super_admin role.

Super Admin View

The super admin (admin@d2ctelcare.com) bypasses the process filter on all list endpoints, sees a Process column in relevant tables, and can manage users across all processes.

Process Slug

Each process has a URL-friendly slug (e.g., momo, ignite) stored in the JWT and used for process-specific routing and display.

10. Deployment & Setup

Prerequisites

- Node.js 18+
- MySQL 8+
- Gmail account with an App Password (for SMTP)
- OpenAI API key (optional, for AI insights)

Environment Variables (server/.env)

```
DATABASE_URL=mysql://root:password@localhost:3306/d2c_ops
JWT_SECRET=your-secret-key
SMTP_HOST=smtp.gmail.com
SMTP_PORT=587
SMTP_USER=your-gmail@gmail.com
SMTP_PASS=your-app-password OPENAI_API_KEY=sk-
...
```

Local Setup Steps

Step	Command	Description
1	npm run install:all	Install all dependencies
2	mysql -u root -p -e "CREATE DATABASE d2c_ops;"	Create the database
3	cd server && npm run migrate	Run all migrations
4	npm run seed	Seed processes, users, and test data

5	npm run dev	Start backend on port 4000
6	cd ../client && npm run dev	Start frontend on port 3000

Production Build

Step	Command
Build frontend	cd client && npm run build
Build backend	cd server && npm run build
Start server	node dist/server/src/index.js

11. Seed Data & Default Credentials

The seed file creates 8 processes and the following default accounts. Change all passwords before any production use.

Role	Email	Default Password
Super Admin	admin@d2ctelcare.com	Admin@D2C2026
Ignite Operator	ops.ignite@d2ctelcare.com	Operator@123
Ignite Manager	mgr.ignite@d2ctelcare.com	Operator@123
Sunking Operator	ops.sunking@d2ctelcare.com	Operator@123
Client Stakeholder	stakeholder@ignite.com	Client@123

```
Security notice: Change all default passwords before any production or demonstration deployment.
```