Q T S O N L I N E

Building an AI-Assisted Quality Intelligence Platform

Quince Defect Analysis System | Garment Manufacturing & Quality Assurance

Real-Time Analytics · AQL Defect Tracking · AI-Powered Classification · Excel Export
Vendor Management · Factory Benchmarking · Column Mapping Engine · User Management

C A S E S T U D Y

Built entirely using Claude AI (Anthropic) — Vibe Coding Methodology

quincedefectanalysis.qtsonline.com · 2025–2026

Executive Summary

QTS Online — the Quince Defect Analysis System — is a production-deployed, enterprise-grade web platform built to solve a critical operational problem facing garment quality teams across South Asia: the complete absence of centralised, real-time defect intelligence. Inspection data was scattered across Excel files, defect analysis was performed manually, and factory performance was compared through laborious spreadsheet work.

This case study documents how Elohim Software Solutions built the entire QTS Online platform using Claude AI (Anthropic) as the primary development engine — a methodology we call Vibe Coding — delivering a sophisticated, multi-module analytics platform in a fraction of the time traditional development would require.

> *QTS Online is now live at quincedefectanalysis.qtsonline.com, processing thousands of real AQL inspection records for garment manufacturers across Sri Lanka and beyond. Every module visible in this case study was built using Claude AI.*

Project Name	QTS Online — Quince Defect Analysis System
Client	Quince — Garment Quality Assurance Division
Industry	Garment Manufacturing & Quality Assurance
Platform	quincedefectanalysis.qtsonline.com
Solution Type	AI-Assisted Web-Based Quality Intelligence Platform
AI Engine	Claude AI by Anthropic — Vibe Coding Methodology
Tech Stack	React JS · Laravel · MySQL · AWS · Excel Integration
Modules Built	Dashboard · Inspection · AQL Inputs · Defects Group · Vendors · General Config · Users
Status	Live in Production — Active real inspection data
Approach	100% AI-Assisted Development from requirement to deployment

Business Problem

Garment quality assurance teams at Quince were managing a complex, multi-factory, multi-vendor inspection operation with no centralised intelligence system. The operational reality was fragmented and reactive:

The Pain Points

- Inspection reports submitted as individual Excel files — no consolidated view across vendors or factories
- AQL defect data stored in disconnected spreadsheets, making trend analysis impossible in real time
- Critical defects (Dead Insect, Wrong Size Garment, Wrong Style Product) buried in raw data rows, invisible to management
- No mechanism to compare factory performance side-by-side — subjective decisions based on incomplete information
- Defect categorisation performed manually — analysts spending hours classifying raw defect descriptions
- Excel upload formats varied by inspector — no standardised column mapping, causing import failures
- Vendor data contained duplicates and naming inconsistencies that corrupted analytical results
- No user access control — all data visible to all users regardless of role
- Zero real-time visibility — management reviewed quality status days or weeks after inspections occurred

> *The result: quality problems were identified too late, vendor accountability was impossible to enforce objectively, and the team was spending more time on data administration than on actual quality improvement.*

What the Client Needed

Problem	Solution Required
Centralised inspection database	One source of truth for all vendors and factories
Real-time defect classification	AI-powered grouping of raw AQL descriptions
Critical defect visibility	Instant dashboard alert for high-risk garment failures
Vendor performance comparison	Side-by-side analytics across all 64+ vendors
Flexible Excel upload	Drag-and-drop column mapping for any report format
Role-based access control	Admin and Client roles with separate data visibility
Export capabilities	One-click Excel export from every data module
Duplicate data management	Vendor merge tool to clean historical duplicates

How We Built It: Claude AI + Vibe Coding

Rather than following a conventional software development cycle — months of design, months of coding, months of testing — the Elohim team used Claude AI as the primary engineering partner throughout every phase of the QTS Online build. This approach, which we call Vibe Coding, treats Claude AI not as a code completion tool but as an active senior engineer capable of architecting, implementing, and refining entire modules from business requirement descriptions.

What is Vibe Coding?

> *Vibe Coding is a structured AI-first development methodology where business requirements, workflow descriptions, and UX goals are fed to Claude AI as rich, contextual prompts. Claude AI generates working implementation across the full stack — React JS components, Laravel controllers, MySQL queries, and business logic — which human engineers then validate, refine, and integrate.*

The AI Development Workflow

Human Input (Business Logic)	Claude AI Output (Working Code)
Business requirement description	Claude AI generates component architecture + data model
Dashboard chart specification	React JS chart component with live filter integration
Defect grouping logic in plain English	PHP keyword-matching and classification engine
Excel upload and column mapping	Drag-and-drop React UI + Laravel parser + validation
Analytics formula description	Working KPI calculation logic with edge case handling
Vendor duplicate problem description	Merge UI with foreign key cascade database logic
Report column mapping requirements	Configurable drag-and-drop mapping engine
User roles and permissions spec	Complete auth middleware, role guards, and UI visibility

Speed of Delivery — AI vs Traditional

Operational Area	Before QTS Online	After QTS Online
Dashboard + Charts	4–6 weeks manual development	**2–3 days with Claude AI**
Defect Group Engine	Custom ML pipeline (2+ months)	**Keyword logic drafted in hours**
Column Mapping Tool	Specialist UI developer (weeks)	**React drag-drop + Laravel (days)**
AQL Upload + Parser	Manual file handling (weeks)	**Full upload pipeline (days)**
Vendor Merge Feature	Database work + UI (weeks)	**Complete feature (1–2 days)**
User Management	Auth system from scratch (weeks)	**Role-based system (days)**
Excel Export (all tabs)	Per-module (sprints)	**Consistent across all modules**
Full Platform	12–18 months (team of 6+)	**Delivered by lean team, rapidly**

Module 1 — Dashboard Analytics

01 Dashboard Analytics — The Intelligence Command Centre

Real-time defect insights, Top 5 Defects Group, Critical Defects Breakdown, with live global filters

The Dashboard is the operational nerve centre of QTS Online. Quality managers, factory teams, and Quince management open this screen to immediately understand the state of garment quality across their entire vendor and factory network. Every chart on this page responds live to the Global Filters — no page reload required.

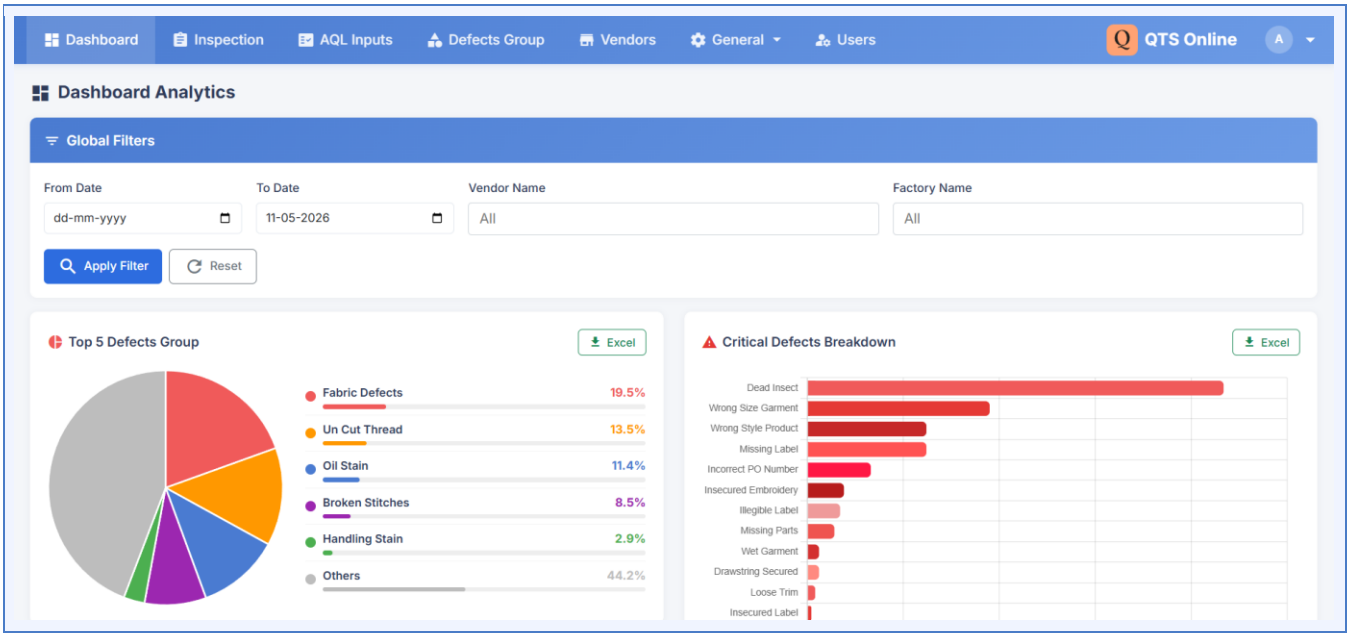


Figure 1: *Dashboard Analytics — Global Filters (From/To Date, Vendor Name, Factory Name), Top 5 Defects Group pie chart with percentage breakdown, and Critical Defects Breakdown horizontal bar chart. Live on quincedefectanalysis.qtsonline.com.*

Global Filters Panel

At the top of every Dashboard session sits the Global Filters panel — a persistent filter bar that controls all charts and analytics simultaneously. This was one of the most technically complex components to build, and Claude AI generated the complete React state management and Laravel API query logic in a single development session.

- From Date / To Date: Filter all analytics to any inspection period — day, week, month, or custom range
- Vendor Name: Filter to a single vendor or view all 64+ vendors simultaneously
- Factory Name: Drill into a specific factory's defect performance in isolation
- Apply Filter button: Triggers live API calls — all charts refresh simultaneously without page reload
- Reset button: Instantly clears all filters and restores the full dataset view

> **Claude AI Contribution:** The entire Global Filters component — React state, debounced API calls, dynamic chart re-renders, and Laravel query scoping — was generated through structured prompting. What would have taken a senior front-end developer 1–2 weeks was drafted in under a day.

Top 5 Defects Group — Pie Chart

The centrepiece of the Dashboard is the Top 5 Defects Group chart. This chart does not simply count defect records — it performs an intelligent classification of raw AQL defect descriptions against a keyword database, groups them into named defect categories, and ranks the top 5 by frequency. The real data from the live system shows:

- Fabric Defects — 19.5% — the single largest defect category across all vendors
- Un Cut Thread — 13.5% — a consistent process deficiency indicating cutting department issues
- Oil Stain — 11.4% — machinery maintenance failure signature
- Broken Stitches — 8.5% — stitching department quality shortfall
- Handling Stain — 2.9% — post-production handling protocol failures
- Others — 44.2% — all remaining defect categories not in the top 5
- Excel Export button: Every chart on the Dashboard has a dedicated Export to Excel button, allowing managers to download the underlying data instantly

> **Business Value:** For the first time, Quince management can see — at a glance — that nearly 20% of all defects are fabric-related. This directs supplier audits, vendor conversations, and procurement decisions with hard data rather than intuition.

Critical Defects Breakdown — Bar Chart

The Critical Defects Breakdown chart on the right side of the Dashboard shows a ranked horizontal bar chart of the most severe garment defects found across all inspections. These are defects categorised as Critical in the Defects Group system — failures that carry the highest risk of product rejection, customer complaints, or safety incidents. The live system shows:

- Dead Insect — the highest-frequency critical defect, indicating severe hygiene and storage failures at factory level
- Wrong Size Garment — second-ranked critical defect, pointing to measurement and QC process failures
- Wrong Style Product — style/specification mismatch, causing order rejection risk
- Missing Label — compliance risk — products cannot be shipped without correct labelling
- Incorrect PO Number — documentation error creating logistics and customs complications
- Insecured Embroidery, Illegible Label, Missing Parts, Wet Garment, Drawstring Secured, Loose Trim, Insecured Label — all tracked and ranked automatically

> **Claude AI Contribution:** The classification logic that determines which defect descriptions map to which Critical Defect Groups — and the ranked bar chart rendering — were entirely designed and implemented using Claude AI through prompt-driven development.

Module 2 — Inspection Management

02 Inspection Management — Centralised Inspection Register

Upload inspection reports, add AQL data, filter by country/vendor/factory/date, export to Excel

The Inspection Management module is the operational database of all garment inspection records. Every inspection event — from Sri Lanka, Bangladesh, India, or any other sourcing country — is stored, searchable, and filterable in this single screen. This replaced the previous reality of individual Excel files submitted by inspectors with no central repository.

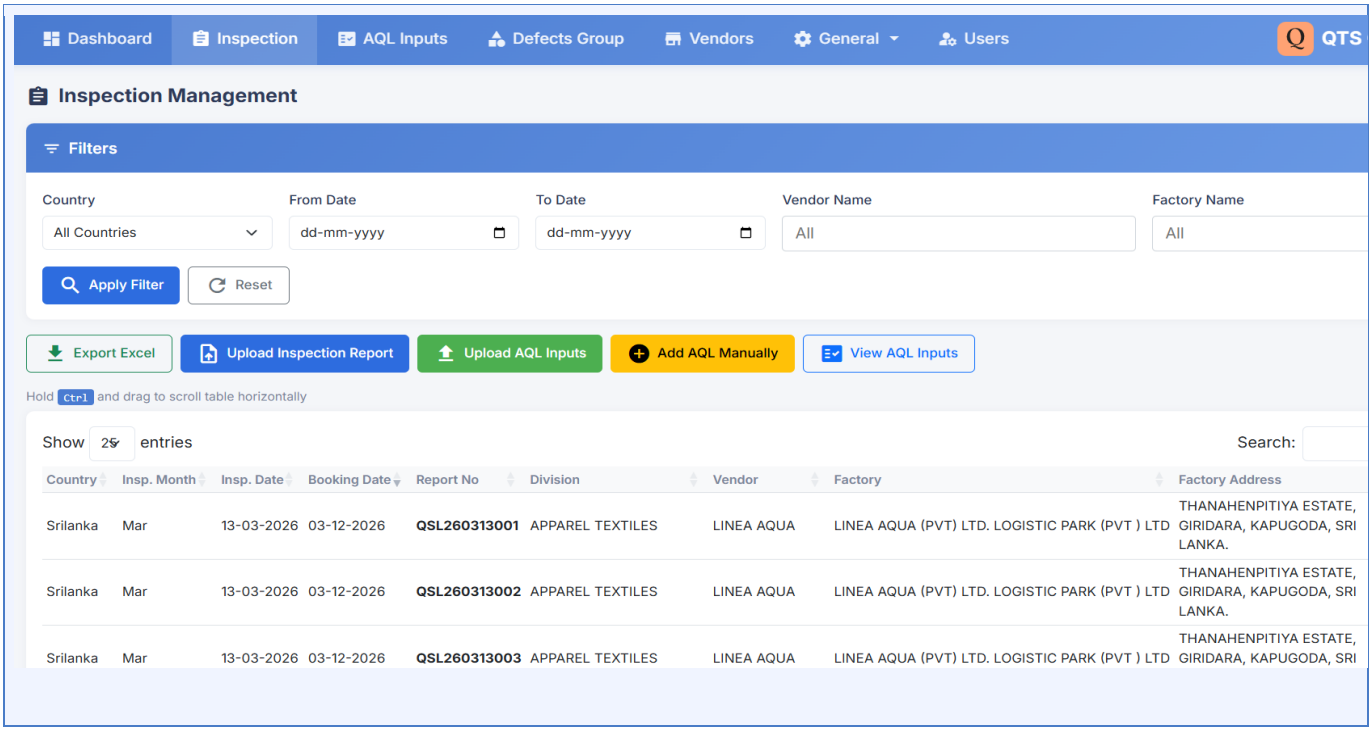


Figure 2: *Inspection Management — Multi-field filter panel (Country, From Date, To Date, Vendor Name, Factory Name), action toolbar (Export Excel, Upload Inspection Report, Upload AQL Inputs, Add AQL Manually, View AQL Inputs), and paginated inspection records table with Report No, Division, Vendor, Factory, Factory Address, and Style/Item columns.*

Inspection Filter Panel

The Inspection module features a dedicated multi-field filter panel with five independent filter dimensions, allowing quality managers to slice the inspection database to exactly the records they need:

- Country: Filter by sourcing country — Sri Lanka, Bangladesh, India, and others — using a dropdown populated from actual inspection data
- From Date / To Date: Date range filter applied to the inspection date field — not the booking date
- Vendor Name: Free-text search against vendor name — partial matches supported
- Factory Name: Free-text search against factory name — useful for multi-factory vendor groups

- Apply Filter / Reset: Consistent UX pattern used across all modules — Claude AI generated a reusable filter component used throughout the platform

The table below the filters shows real inspection records from the live system, including Report Numbers (QSL260313001, QSL260313002, QSL260313003...), the Division (APPAREL TEXTILES), Vendor (LINEA AQUA), Factory (LINEA AQUA (PVT) LTD. LOGISTIC PARK (PVT) LTD), Factory Address (THANAHENPITIYA ESTATE, GIRIDARA, KAPUGODA, SRI LANKA), and Style/Item references.

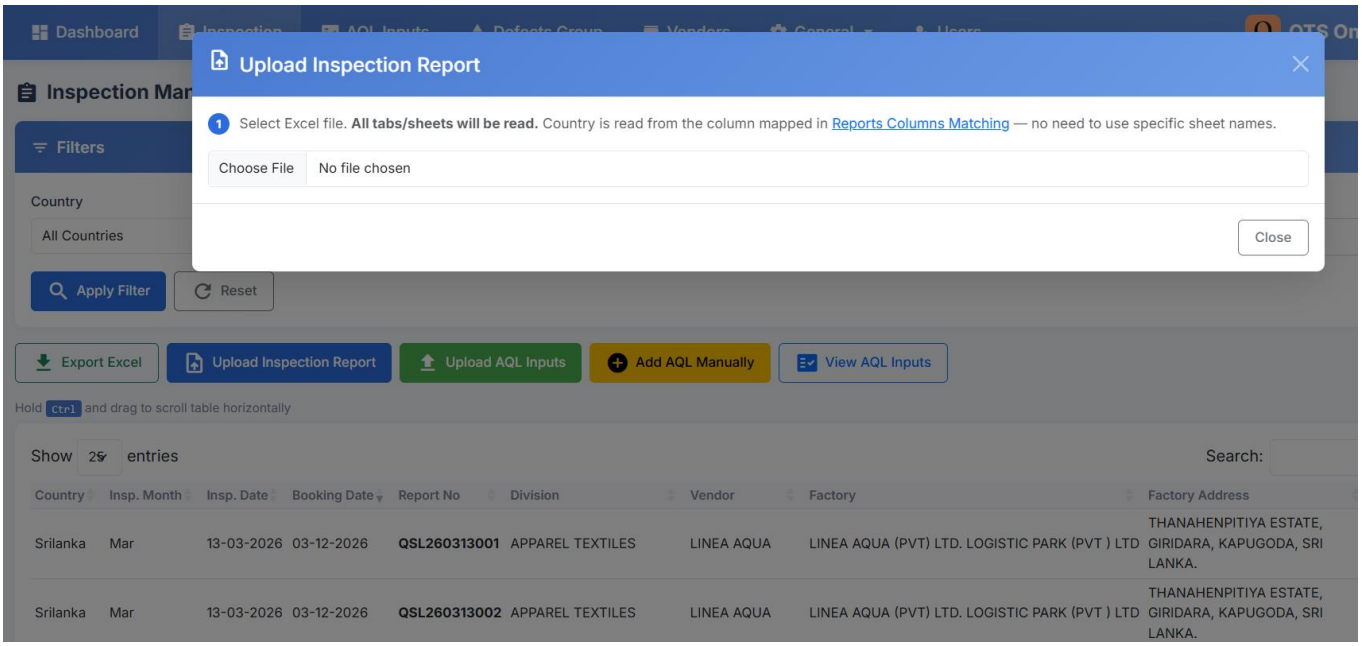
Action Toolbar — Five Key Operations

1. Export Excel

Exports the currently filtered inspection dataset to a formatted Excel file. Applied filters are respected — export exactly what you see on screen. Claude AI generated the Laravel Excel export logic using the Maatwebsite package with proper column formatting.

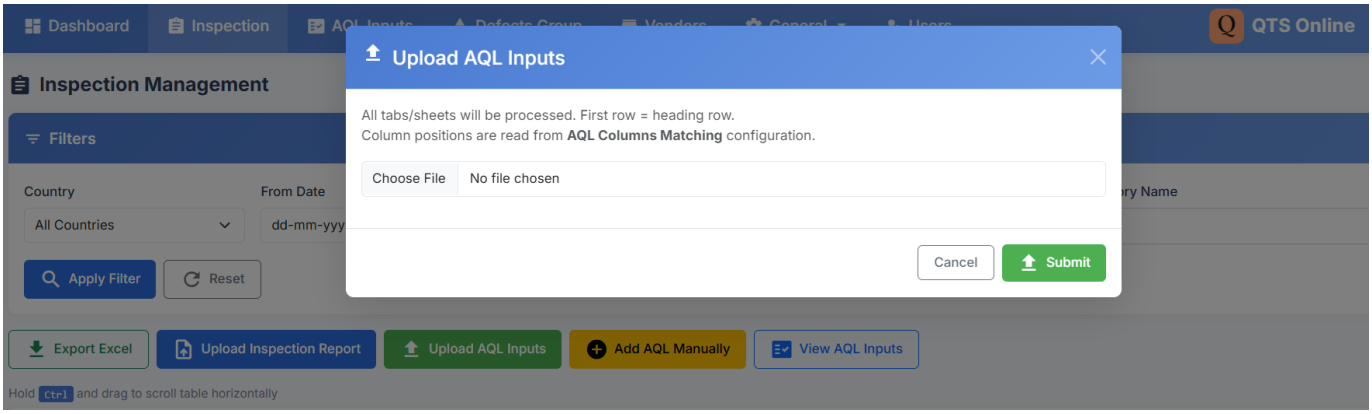
2. Upload Inspection Report

Allows inspectors and coordinators to upload bulk inspection report Excel files. The system reads the file against the saved column mapping (configured in the Reports Columns Matching screen under General settings) and inserts records into the database. File validation, error reporting, and duplicate detection were all implemented using Claude AI-generated Laravel logic.



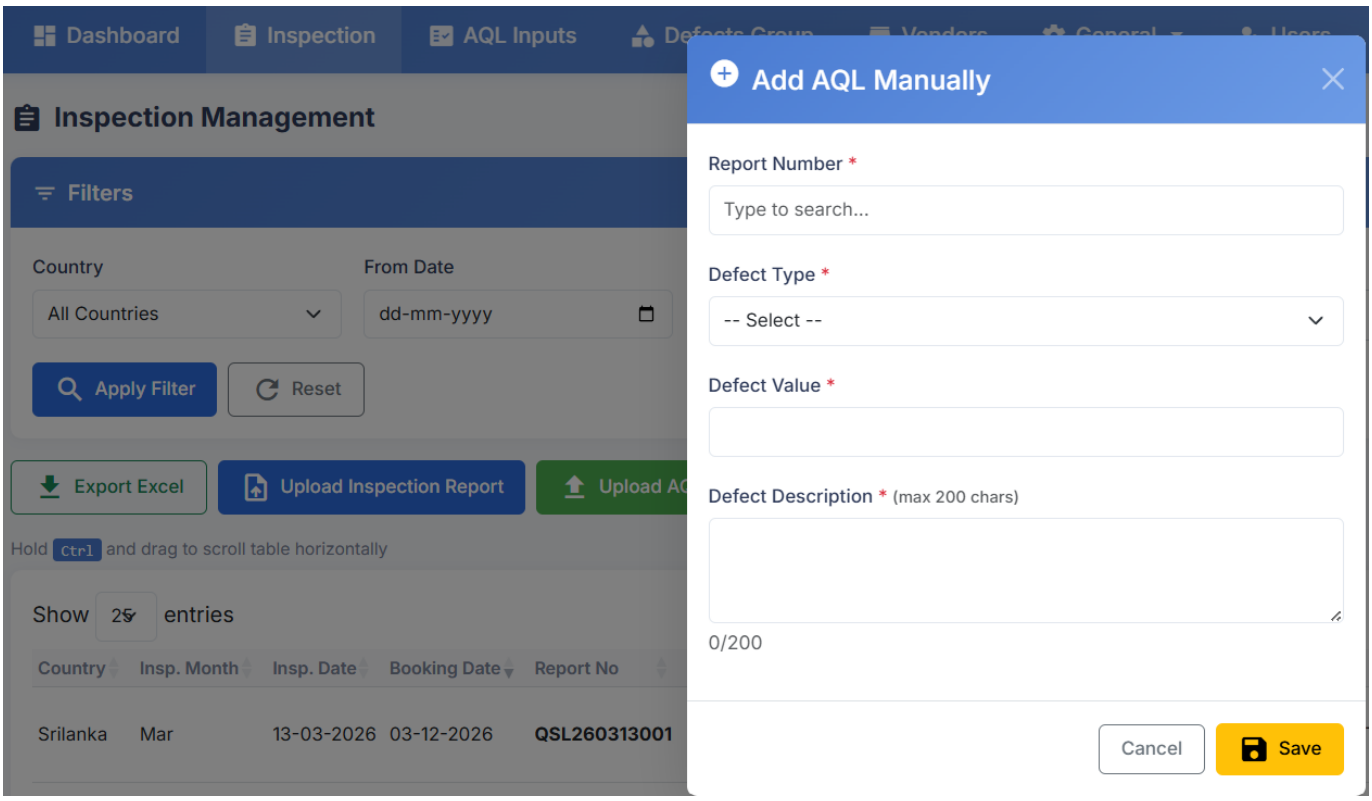
3. Upload AQL Inputs

A separate upload workflow specifically for AQL defect data files. AQL data has a different structure from inspection report data — separate column mappings, separate validation rules. Claude AI designed the dual-upload architecture that keeps these two datasets cleanly separated yet linked by Report Number.



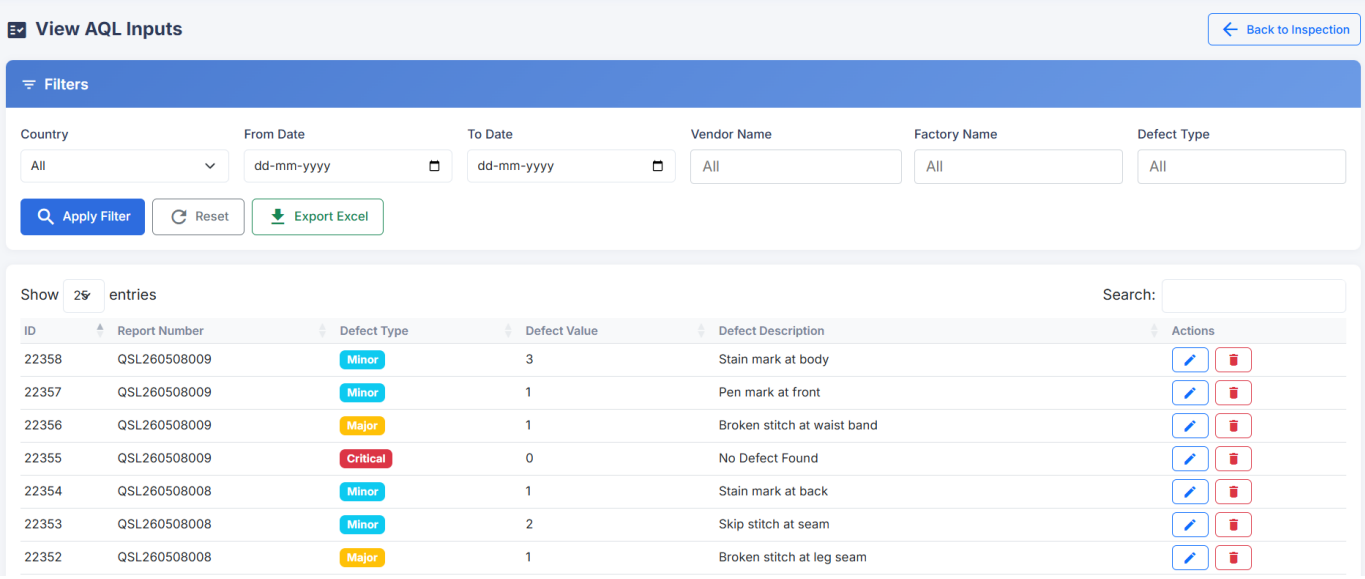
4. Add AQL Manually

For cases where inspectors want to log individual AQL defect entries without uploading a file, this button opens an inline entry form. The form captures Report Number, Defect Type (Critical / Major / Minor), Defect Value, and Defect Description. Claude AI generated the form validation and API endpoint in a single session.



5. View AQL Inputs

Navigates to the AQL Inputs view for the current filter context — showing all individual defect line items linked to the filtered inspections. This cross-module navigation was designed and implemented using Claude AI-generated React Router logic.



Module 3 — AQL Inputs Management

03 AQL Inputs — Individual Defect Record Management

View, filter, edit, and delete individual AQL defect line items with Critical/Major/Minor classification

The AQL Inputs module is the granular defect database — every individual defect finding from every inspection is stored as a separate record here. This is the raw data layer that powers the Dashboard's AI defect classification engine. Each record captures the exact defect description text that the classification engine reads to assign Defect Groups.

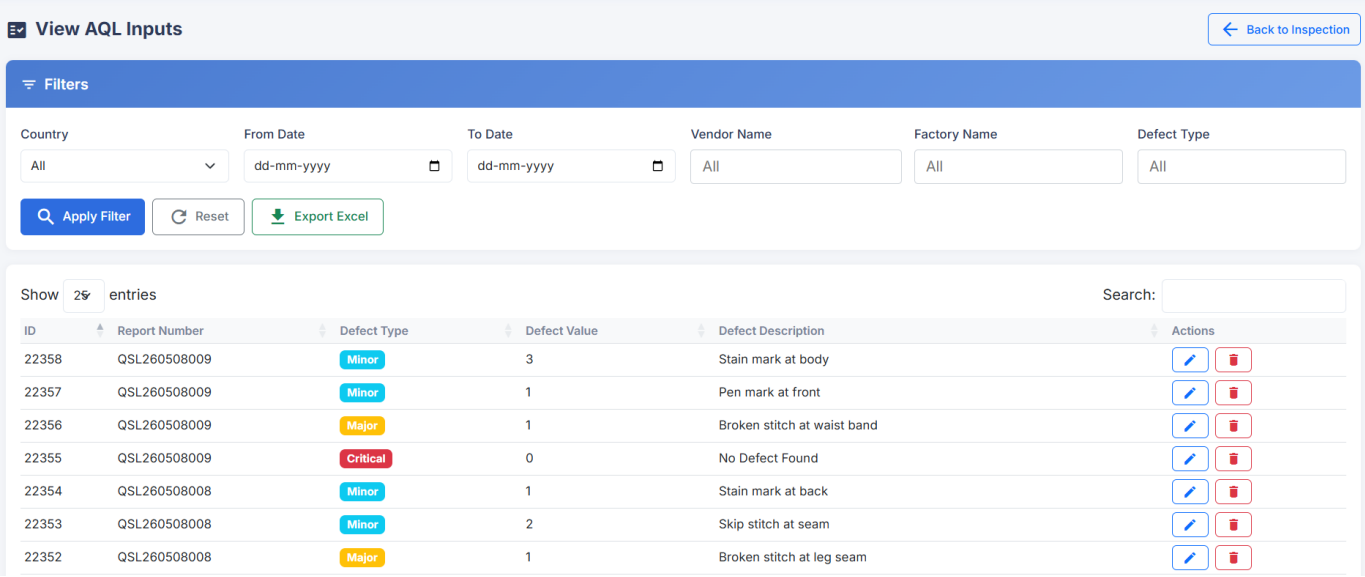


Figure 3: *AQL Inputs Management — Filter panel (Country, From Date, To Date, Vendor Name, Factory Name, Defect Type), colour-coded Defect Type badges (Critical=red, Major=orange, Minor=cyan), tabular view of individual defect records with Report Number, Defect Value, Defect Description, and edit/delete actions.*

Defect Type Classification Badges

Every AQL input record carries one of three Defect Type classifications, displayed as colour-coded badges in the table. This classification drives all downstream analytics:

CRITICAL	MAJOR	MINOR

- Critical (Red badge): Defects that pose safety risks, cause order rejection, or indicate systemic quality failure — e.g., Dead Insect, Wrong Size Garment, Missing Label
- Major (Orange badge): Significant defects that will likely cause the garment to be returned or reworked — e.g., Broken Stitch at Waist Band, Broken Stitch at Leg Seam

- Minor (Cyan badge): Defects noted but unlikely to cause rejection — e.g., Stain Mark at Body, Pen Mark at Front, Skip Stitch at Seam

Real Data from Live System

The AQL Inputs screen shown above displays actual records from the live QTS Online system. The records visible include:

- ID 22358 — Report QSL260508009 — Minor — Value: 3 — Stain mark at body
- ID 22357 — Report QSL260508009 — Minor — Value: 1 — Pen mark at front
- ID 22356 — Report QSL260508009 — Major — Value: 1 — Broken stitch at waist band
- ID 22355 — Report QSL260508009 — Critical — Value: 0 — No Defect Found
- ID 22354 — Report QSL260508008 — Minor — Value: 1 — Stain mark at back
- ID 22353 — Report QSL260508008 — Minor — Value: 2 — Skip stitch at seam
- ID 22352 — Report QSL260508008 — Major — Value: 1 — Broken stitch at leg seam

Six-Dimension Filter System

AQL Inputs supports six simultaneous filter dimensions — Country, From Date, To Date, Vendor Name, Factory Name, and Defect Type — giving quality analysts precise control over which defect records they are analysing. Combined with the Export Excel button, this allows targeted defect reports to be generated in seconds.

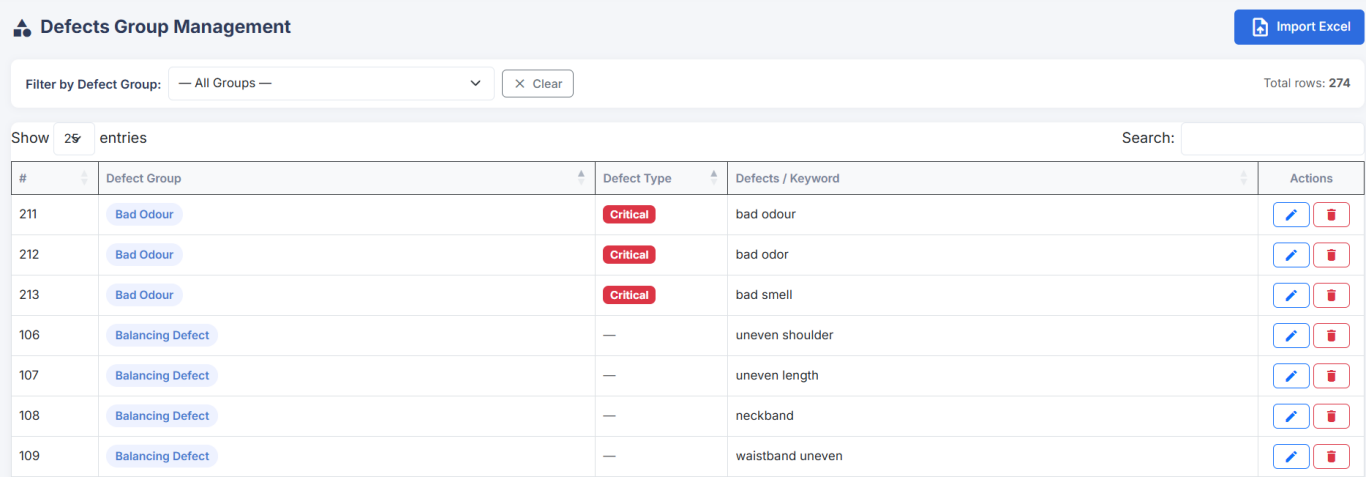
> **Claude AI Contribution:** The entire AQL Inputs module — filter logic, colour-coded badge rendering, paginated table with server-side search, inline edit/delete with confirmation modals, and Excel export — was built using Claude AI through structured prompt engineering. The six-dimension filter API query was generated in a single prompt session.

Module 4 — Defects Group Management

04 Defects Group — The AI Classification Engine

274 keyword rules mapping raw defect descriptions to named defect groups with Critical/Major/Minor typing

The Defects Group Management module is the intelligence layer that makes the Dashboard's Top 5 Defects Group chart possible. This is where the AI-powered defect classification engine is configured — a database of 274 keyword rules that the system uses to automatically classify every raw AQL defect description into a named defect group.



#	Defect Group	Defect Type	Defects / Keyword	Actions
211	Bad Odour	Critical	bad odour	
212	Bad Odour	Critical	bad odor	
213	Bad Odour	Critical	bad smell	
106	Balancing Defect	—	uneven shoulder	
107	Balancing Defect	—	uneven length	
108	Balancing Defect	—	neckband	
109	Balancing Defect	—	waistband uneven	

Figure 4: *Defects Group Management — 274 total rows, filterable by Defect Group, showing Defect Group name, Defect Type (Critical badge), and Defects/Keyword column. Groups shown include Bad Odour (Critical), Balancing Defect, and Bow with their associated keywords.*

How the Classification Engine Works

The classification logic is elegantly simple but operationally powerful. For each AQL defect description text submitted in an inspection report, the system:

- Step 1: Reads the raw defect description text (e.g., 'bad odour found on garment')
- Step 2: Searches the Defects / Keyword column of the Defects Group table for a partial match
- Step 3: When a keyword match is found, assigns the corresponding Defect Group name to that record
- Step 4: If the matched Defect Group has a Defect Type of Critical, the record is included in the Critical Defects Breakdown chart
- Step 5: Aggregates all matched records by Defect Group name to generate the Top 5 ranking

The current live system has 274 classification rules across dozens of defect groups. The screenshot shows real rules including:

- Bad Odour (Critical) — keywords: 'bad odour', 'bad odor', 'bad smell' — three variants mapped to the same group
- Balancing Defect — keywords: 'uneven shoulder', 'uneven length', 'neckband', 'waistband uneven', 'collar point'

- Bow — keyword: 'bowing'

Defect Group Configuration

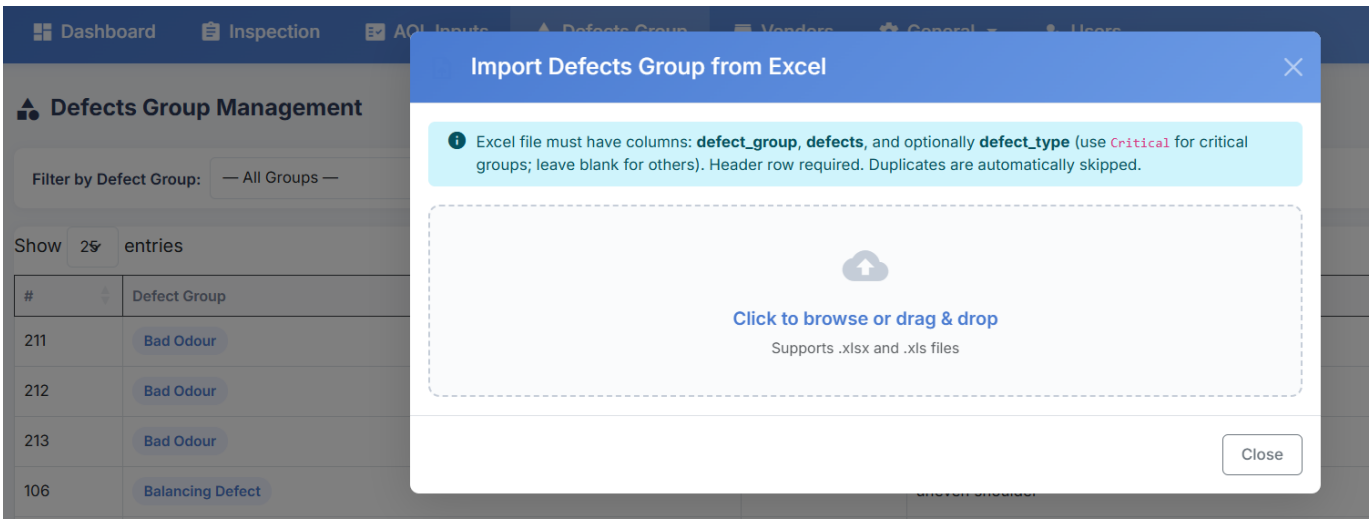
Every Defect Group entry has three fields that together define the classification rule:

- Defect Group (required): The human-readable category name that appears on the Dashboard chart — e.g., 'Fabric Defects', 'Un Cut Thread', 'Bad Odour'
- Defect Type: Set to Critical only for defect groups that should appear in the Critical Defects Breakdown chart. Left blank for all other groups that contribute only to the Top 5 analysis
- Defects / Keyword (required): The keyword string that the system searches for in raw AQL description text — partial, case-insensitive matching

Import Excel

The Import Excel button at the top-right allows administrators to bulk-load classification rules from a spreadsheet. This enabled the initial data load of 274 rules in a single operation. Claude AI generated the Laravel Excel import logic with validation, duplicate checking, and error reporting.

> **Business Value:** The Defects Group engine transforms thousands of free-text defect descriptions — entered by different inspectors in different phrasings — into consistent, comparable, chart-ready analytics. Without this classification layer, the Dashboard charts would be impossible to generate accurately.



Module 5 — Edit Record Modal

05 Edit Record — Inline Configuration Management

Modal-based edit interface for Defect Group rules with type guidance and keyword documentation

The Edit Record modal is the inline editing interface used across the QTS Online platform for modifying classification rules, vendor records, and user data. Shown here in the context of the Defects Group module, it demonstrates the consistent, clean modal UX pattern that Claude AI generated and applied across every editable record in the system.

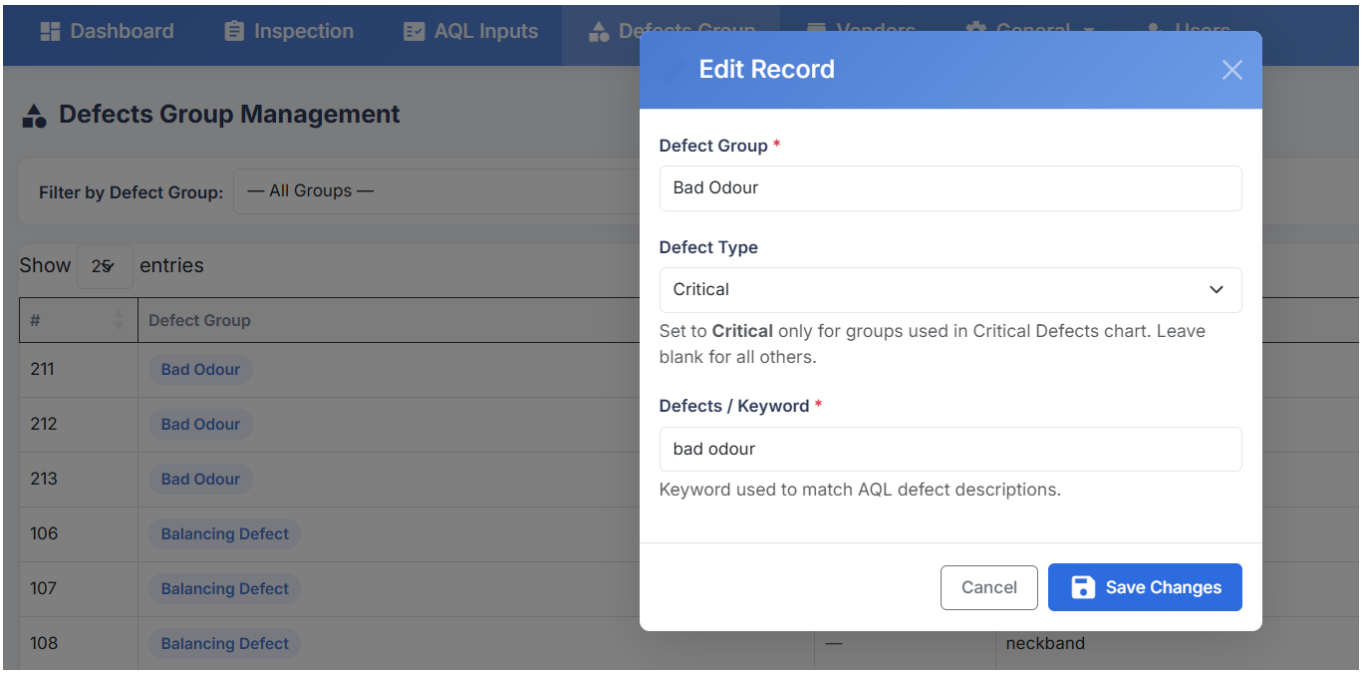


Figure 5: *Edit Record Modal — Defect Group field (Bad Odour), Defect Type dropdown (Critical selected), contextual guidance text explaining Critical usage, Defects/Keyword field (bad odour) with helper text, Cancel and Save Changes buttons.*

Modal Design & UX

The Edit Record modal demonstrates several important design decisions that Claude AI implemented consistently across the platform:

- Clear field labelling with red asterisks (*) indicating required fields — Defect Group and Defects/Keyword are required; Defect Type is optional
- Contextual helper text under the Defect Type dropdown: 'Set to Critical only for groups used in Critical Defects chart. Leave blank for all others.' — this guidance prevents configuration errors without requiring a separate documentation page
- Keyword field helper text: 'Keyword used to match AQL defect descriptions.' — clarifies the technical purpose of the field in plain language
- Cancel and Save Changes buttons with distinct visual weight — Cancel is outlined, Save Changes is filled blue — following standard modal UX conventions
- Modal overlay with background blur — prevents accidental interaction with the table behind the modal

Cross-Platform Consistency

This same modal pattern — with consistent field layout, helper text, validation, and button styling — was applied by Claude AI across every edit workflow in the platform: Defect Group edits, Vendor record edits, User account edits, and AQL Input edits. This consistency was achieved because Claude AI understood the design system and applied it uniformly rather than generating ad-hoc UI for each module.

> **Claude AI Contribution:** The modal component, form validation, API PUT request handling, optimistic UI update, and error state management were all generated by Claude AI in a single component generation session, then reused with minor customisation across all modules.

Module 6 — Vendor Management

06

Vendor Management — 64 Vendors with Duplicate Merge Intelligence

Vendor registry with intelligent duplicate detection and database-safe merge operation

The Vendor Management module maintains the master list of all garment vendors whose inspection data is tracked in QTS Online. The live system currently manages 64 vendors including AADHIRAN GLOBAL, AAN CLOTHING, BAZAAR KONNECTIONS, BHARATH ENTERPRISES, and many others. Beyond basic CRUD, this module features a sophisticated Vendor Duplicate Merger — one of the most operationally important features in the entire platform.

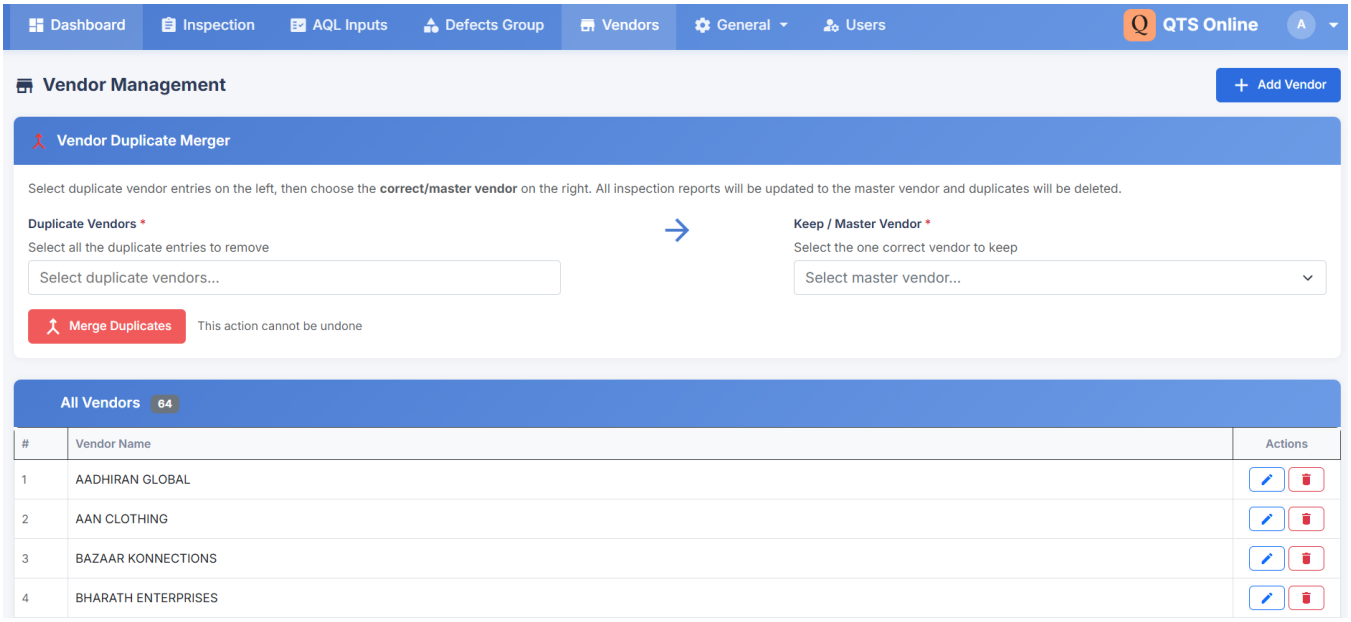
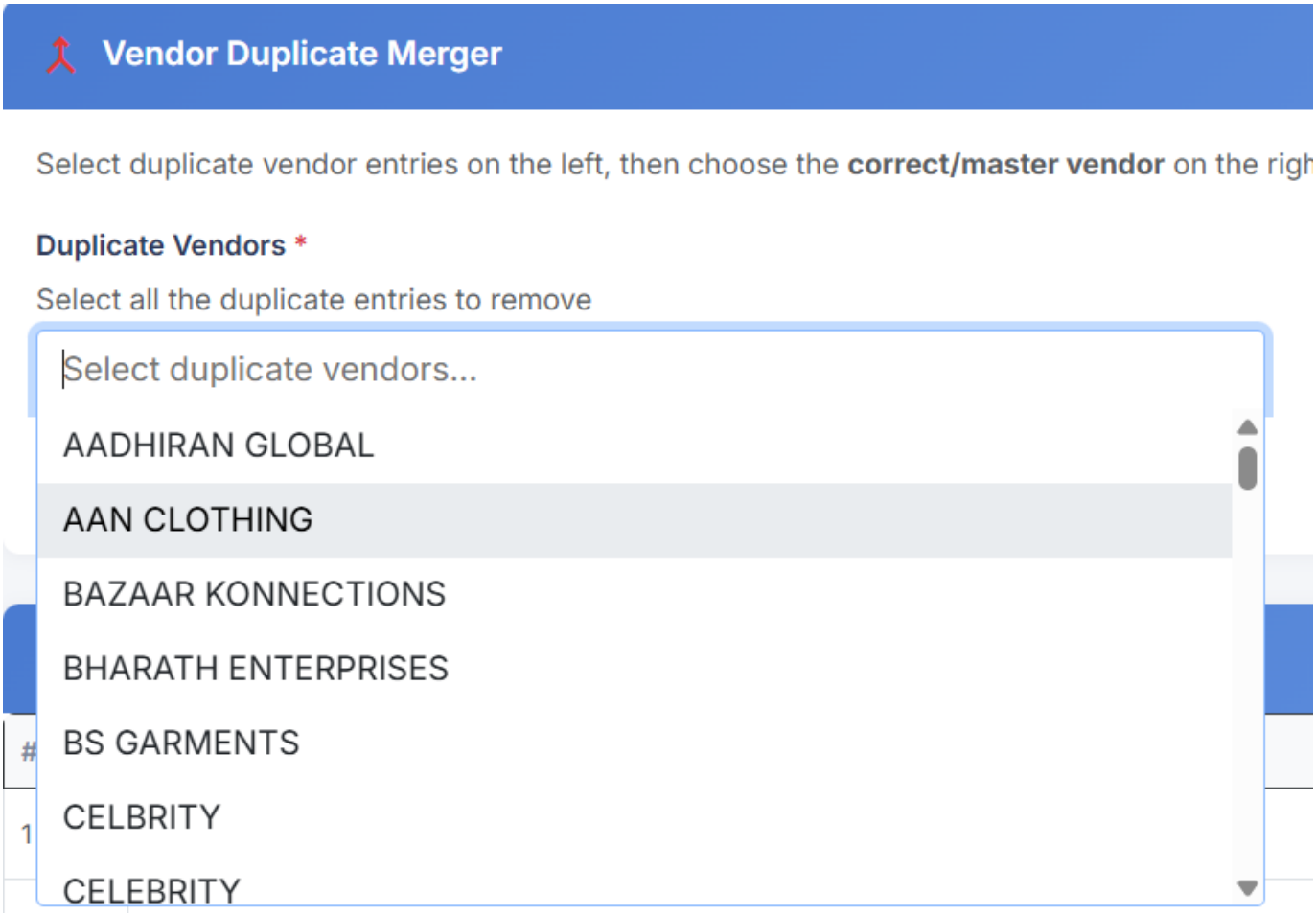


Figure 6: *Vendor Management — Vendor Duplicate Merger panel (Duplicate Vendors selector, Keep/Master Vendor selector, Merge Duplicates button with warning), All Vendors table showing 64 vendors with Add/Edit/Delete actions.*

The Vendor Duplicate Problem

When inspection reports are uploaded from different sources over time, the same vendor often appears under slightly different names: 'LINEA AQUA', 'Linea Aqua', 'LINEA AQUA PVT LTD'. In a flat database, these appear as three separate vendors — fragmenting that vendor's defect analytics across three buckets and making vendor-level reporting meaningless.

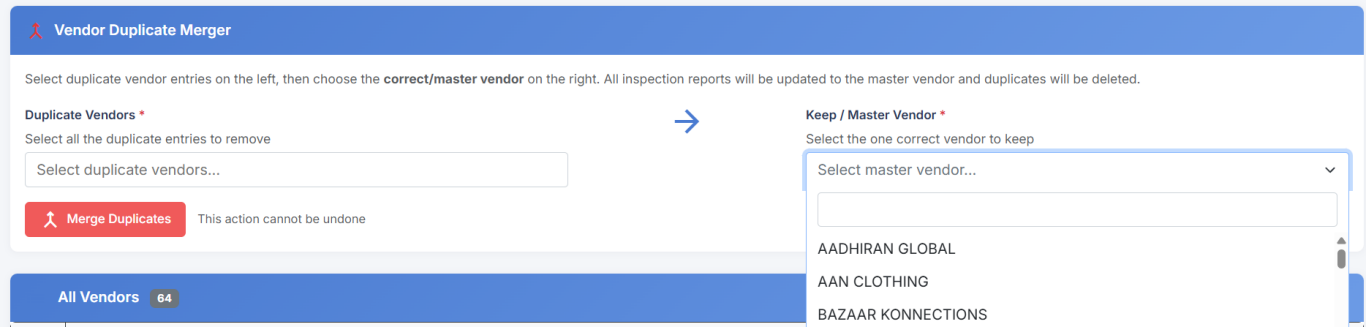


Without a duplicate merge capability, every filter and every chart that referenced that vendor would show incomplete data. The Vendor Duplicate Merger solves this at the database level — not just the display level.

Vendor Duplicate Merger — How It Works

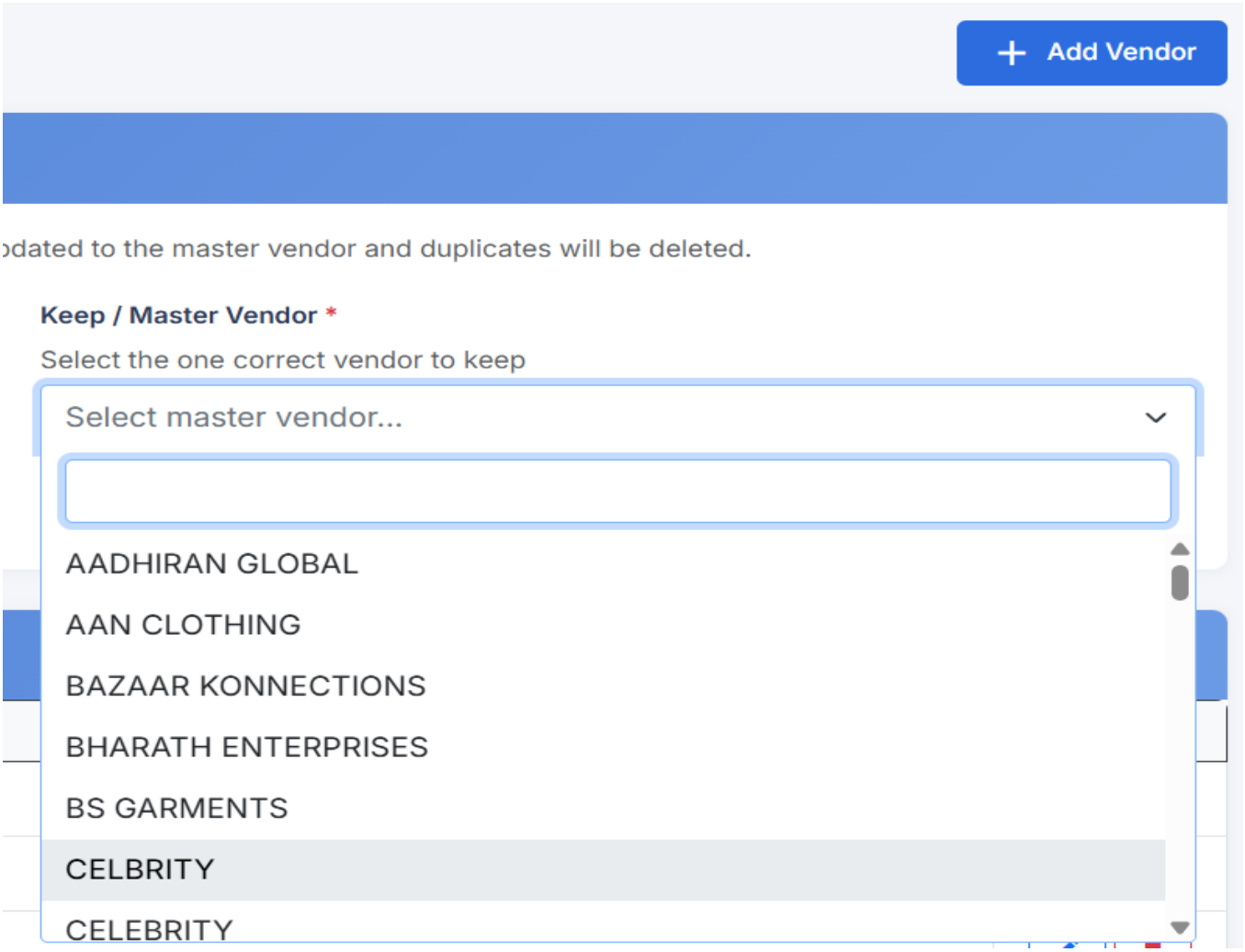
- Step 1 — Select Duplicates: Choose all variant names to be removed (the duplicates) from the left multi-select dropdown
- Step 2 — Select Master: Choose the single correct vendor name to keep from the right dropdown — all inspection records will be migrated to this vendor
- Step 3 — Merge: Click Merge Duplicates — the system updates all foreign key references across the inspections table, then deletes the duplicate vendor entries
- Warning: 'This action cannot be undone' — the UI clearly communicates the irreversibility of the operation

Claude AI Contribution: The database-safe merge logic — which required cascade-updating foreign keys across multiple tables before deleting the duplicate records — was designed and implemented by Claude AI. This prevented data loss that a naive delete-and-reassign approach would have caused.



Vendor Master List

The All Vendors section shows all 64 registered vendors with their sequential IDs. Each vendor record has Edit (pencil icon) and Delete (trash icon) action buttons. The Add Vendor button in the top-right opens the same consistent modal pattern used across the platform. New vendors are automatically available in all filter dropdowns and report uploads immediately after creation.



Module 7 — Reports Columns Matching

07 Reports Columns Matching — Flexible Excel Import Configuration

Drag-and-drop column mapping engine allowing any inspection report Excel format to be imported

The Reports Columns Matching screen is one of the most technically sophisticated features in QTS Online — and one of the most practically impactful. Different inspection agencies and factory coordinators submit inspection reports in Excel files with different column orders. Without a flexible mapping engine, every different format would require a code change or a manual data transformation.

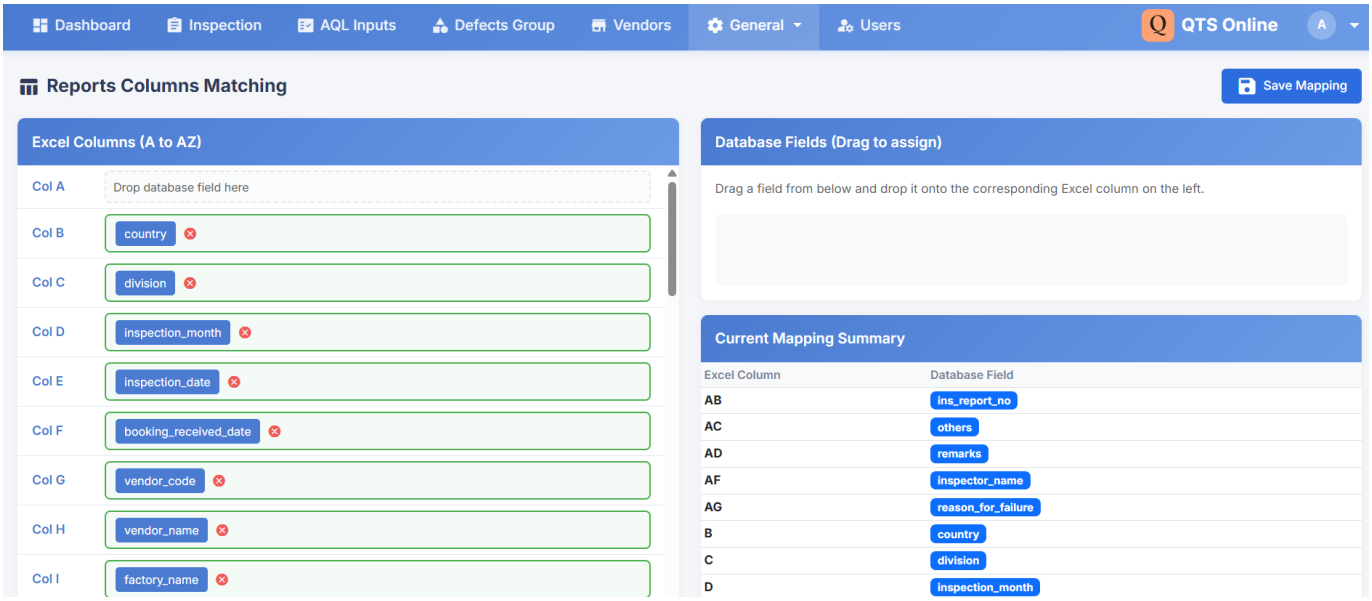


Figure 7: *Reports Columns Matching — Excel Columns A to AZ on the left with drag-and-drop database field assignments (country, division, inspection_month, inspection_date, booking_received_date, vendor_code, vendor_name, factory_name), Database Fields panel on the right, Current Mapping Summary table showing complete column-to-field mapping.*

How the Column Mapping Engine Works

The mapping engine presents two panels side-by-side:

- Left Panel — Excel Columns (A to AZ): Every possible column slot from A through AZ is listed. Each slot has a drop zone where a database field tag can be placed
- Right Panel — Database Fields (Drag to assign): All available database field names are listed as draggable tags — country, division, inspection_month, inspection_date, booking_received_date, vendor_code, vendor_name, factory_name, and many more
- Drag and Drop: The user drags a database field tag from the right panel and drops it onto the corresponding Excel column slot on the left

- Current Mapping Summary: A table at the bottom-right shows the complete current mapping at a glance — AB=ins_report_no, AC=others, AD=remarks, AF=inspector_name, AG=reason_for_failure, B=country, C=division, D=inspection_month, E=inspection_date
- Save Mapping: Once the mapping is configured, Save Mapping persists it to the database — all future Excel uploads use this mapping automatically

Why This Matters Operationally

The live system's current mapping shows that the inspection report Excel format places country in column B, division in column C, inspection month in column D, inspection date in column E, booking date in column F, vendor code in column G, vendor name in column H, and factory name in column I. If a new inspection agency submits reports with these fields in different columns, an administrator can update the mapping in minutes without any code change.

> **Claude AI Contribution:** The entire drag-and-drop mapping UI — built in React using drag-and-drop event handling, state management for tag placement, and the Current Mapping Summary auto-generation — was implemented by Claude AI. The Laravel backend that reads the saved mapping during Excel processing was also Claude AI-generated.

Module 8 — AQL Inputs Columns Matching

08 AQL Inputs Columns Matching — Separate Mapping for Defect Data Files

Independent column mapping configuration for AQL defect upload files (ins_report_no, defects_type, defects, value)

The AQL Inputs Columns Matching screen operates identically to the Reports Columns Matching screen but is configured separately for AQL defect data upload files. AQL files have a fundamentally different structure from inspection report files — they contain defect-level records rather than inspection-level records — and therefore require their own independent column mapping configuration.

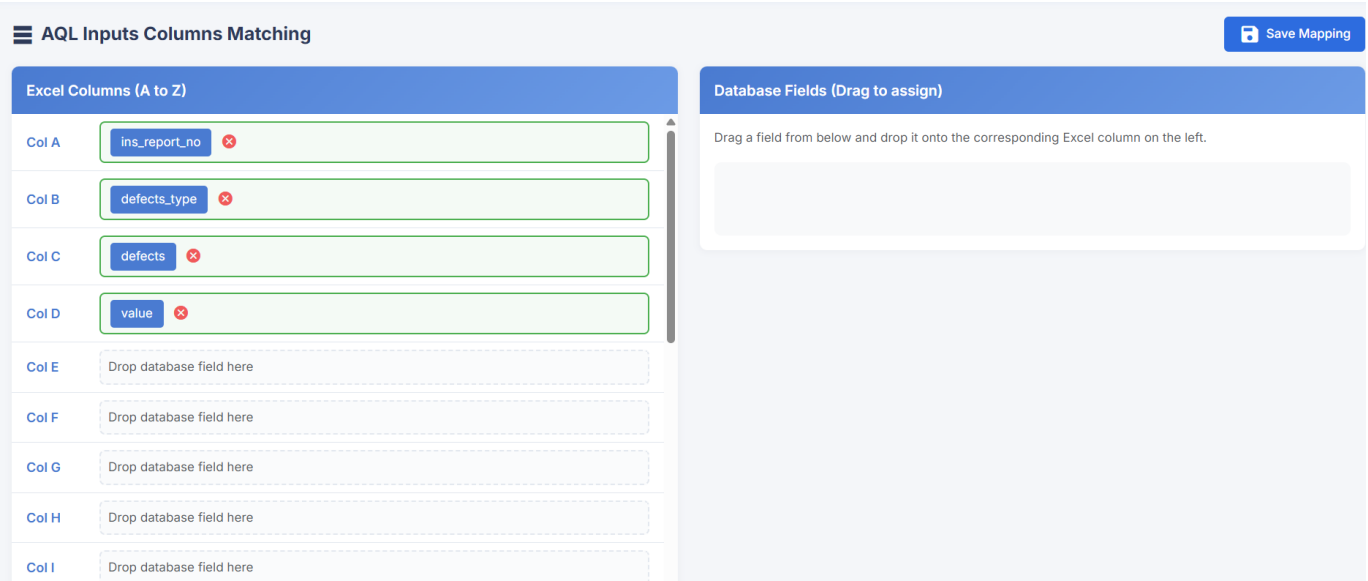


Figure 8: *AQL Inputs Columns Matching — Excel Columns A to Z with four active mappings (Col A=ins_report_no, Col B=defects_type, Col C=defects, Col D=value), remaining columns E through I showing empty 'Drop database field here' placeholders.*

Current AQL Column Mapping

The live system's current AQL Inputs mapping is:

- Col A → ins_report_no: The inspection report number that links each AQL defect record back to its parent inspection — the foreign key that joins the two datasets
- Col B → defects_type: The defect severity classification — Critical, Major, or Minor — stored as text and converted to the badge colour system in the UI
- Col C → defects: The raw defect description text — this is the field that the Defects Group keyword engine scans for classification matches
- Col D → value: The numeric count of this defect found in the inspected lot — used for weighted defect rate calculations
- Cols E through I: Unmapped — these column slots are available for future AQL format variations that may include additional fields

Separation of Concerns

The architectural decision to maintain two separate column mapping configurations — one for inspection reports and one for AQL inputs — reflects a fundamental data model distinction. Inspection reports describe inspection events (one row per inspection). AQL inputs describe defect instances (multiple rows per inspection). Keeping their upload pipelines and column mappings independent prevents format changes in one file type from breaking uploads of the other.

> **Claude AI Contribution:** Claude AI designed this dual-mapping architecture when given the requirement: 'inspection files and AQL files have different formats and need separate mapping configurations'. The component reuse strategy — building one mapping engine and parameterising it for two different mapping types — was suggested and implemented by Claude AI.

Module 9 — User Management

09 User Management — Role-Based Access Control

Admin and Client roles with full user lifecycle management, password reset, and audit trail

The User Management module provides complete control over who can access the QTS Online platform and at what permission level. The live system currently has three users: the platform admin, the Elohim Software Solutions client account, and the Quince end-client account. The module supports full user lifecycle management through an intuitive admin interface.

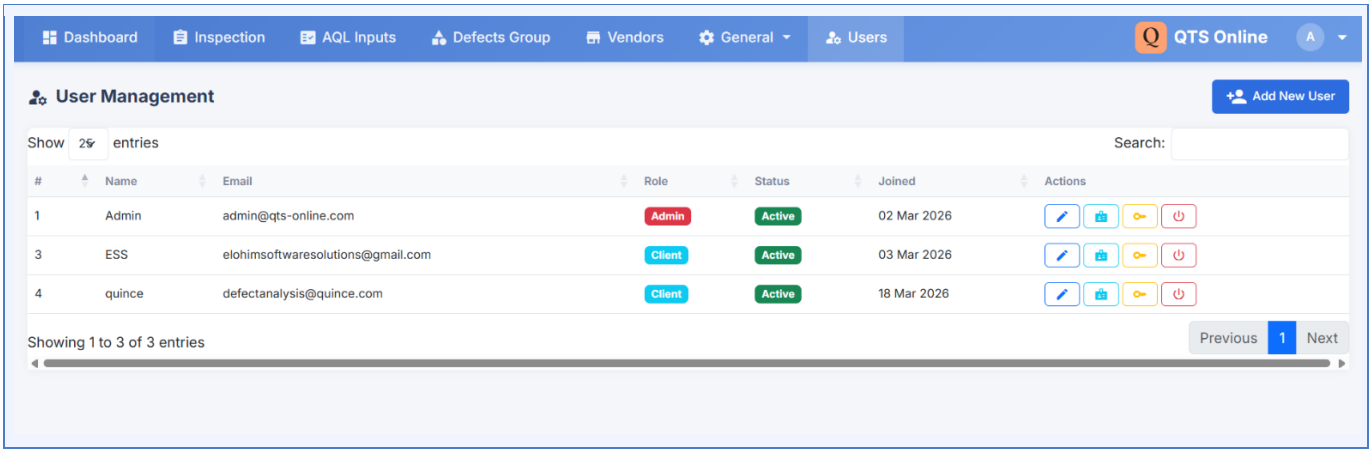


Figure 9: *User Management — Three active users shown (Admin/admin@qts-online.com/Admin role, ESS/elohimsoftwaresolutions@gmail.com/Client role, quince/defectanalysis@quince.com/Client role), all with Active status and joined dates. Action buttons per user: Edit, Permissions, Password Reset, and Deactivate.*

User Roles

QTS Online implements a two-tier role system:

ADMIN	CLIENT

- Admin Role (red badge): Full platform access — all modules visible, all configuration screens accessible, User Management available, column mappings editable, vendor management with merge capability
- Client Role (cyan badge): Restricted access — dashboard, inspection data, and AQL inputs visible; configuration, user management, and column mapping screens hidden; appropriate for end-client Quince staff who consume analytics but do not configure the system

Live User Accounts

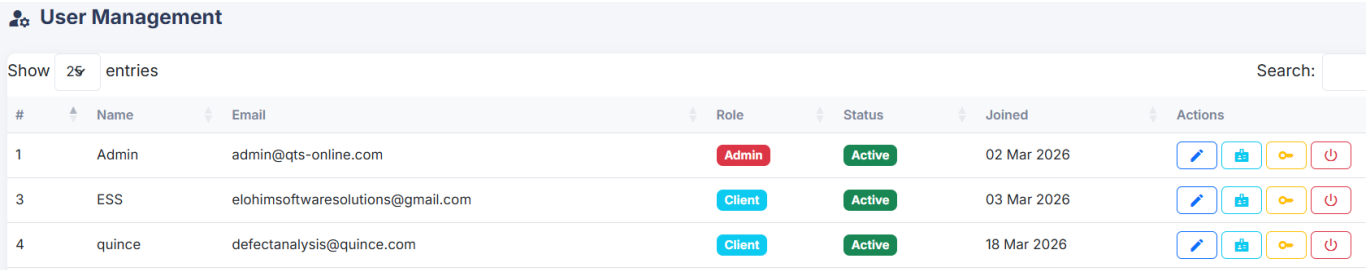
The three live accounts visible in the screenshot represent the actual platform users:

- Admin — admin@qts-online.com — Admin role — Active since 02 Mar 2026 — Platform administrator account
- ESS — elohimsoftwaresolutions@gmail.com — Client role — Active since 03 Mar 2026 — Elohim Software Solutions development and support access
- quince — defectanalysis@quince.com — Client role — Active since 18 Mar 2026 — Quince quality team access account

User Action Buttons

Each user row has four action buttons:

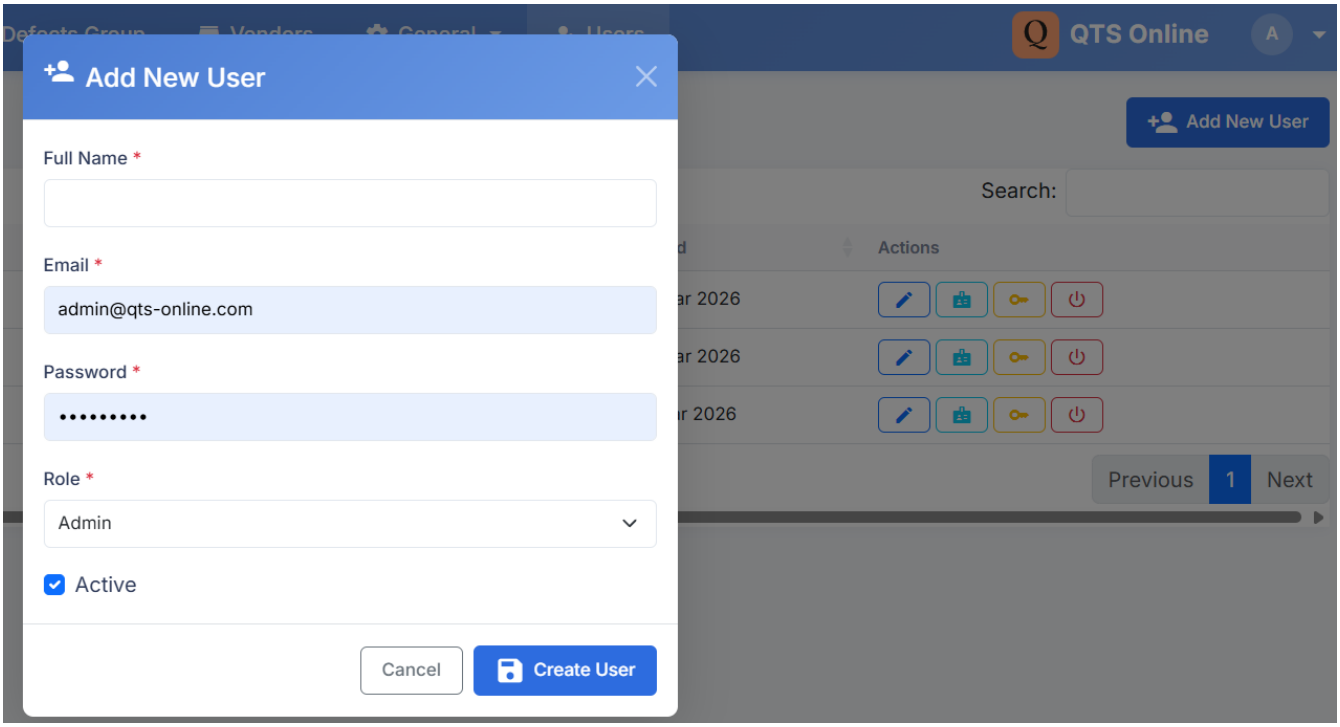
- Edit (pencil icon): Opens the Edit Record modal pre-populated with the user's name, email, and role — consistent with the modal pattern used across all modules
- Permissions (clipboard icon): Manages granular module-level permissions for that user beyond the base role — allows fine-tuning visibility on a per-user basis
- Password Reset (key icon): Generates a password reset for the user — admin-initiated, not requiring the user to remember their old password
- Deactivate (power icon): Soft-deactivates the user account — preserving their data and audit trail while preventing login — reversible unlike deletion



Add New User

The Add New User button in the top-right opens the user creation modal where administrators specify name, email address, role (Admin or Client), and initial password. New users receive access immediately upon creation with no email verification step required — appropriate for the controlled enterprise environment.

> **Claude AI Contribution:** The complete authentication system — Laravel Sanctum token-based auth, role middleware guards, React route-level access control, and the user management CRUD interface — was generated through Claude AI. The role-based route hiding (Client users do not see configuration menu items) was implemented in a single React Router guard component.



Business Impact & Outcomes

Operational Area	Before QTS Online	After QTS Online
Defect Tracking	Manual Excel spreadsheets, delayed	**Real-time AI-classified analytics**
Critical Defect Alerts	Buried in rows, found days later	**Instant Dashboard visibility**
Top Defect Identification	Hours of manual pivot table work	**Automated, always current**
Vendor Performance	Impossible to compare objectively	**One-click vendor-wise analytics**
Factory Comparison	Manual spreadsheet effort	**Live filter → instant comparison**
AQL Data Upload	Manual entry or format-specific import	**Flexible drag-and-drop column mapping**
Defect Classification	Analyst reads and categorises manually	**274-rule AI keyword matching engine**
Reporting Cycle	Days to compile a quality report	**Seconds — Apply Filter + Export Excel**
Vendor Data Quality	Duplicates corrupting all analytics	**One-click Vendor Duplicate Merger**
Access Control	No user management — full data exposure	**Admin/Client roles with permissions**
Data Volume	Limited by Excel file size	**22,000+ AQL records, scalable**
Decision Speed	Weekly or monthly quality reviews	**Real-time operational decisions**

The Claude AI Development Methodology

> *Every module in QTS Online — every component, every API endpoint, every database migration, every chart, every modal, every filter, every export — was built using Claude AI as the primary engineering tool. Human engineers provided business context, validated outputs, and managed production deployment. Claude AI did the heavy lifting of implementation.*

AI Responsibilities vs Human Responsibilities

Claude AI Generated	Human Engineers Owned
React component generation	System architecture decisions
Laravel API & controller logic	Database schema design & review
MySQL query optimisation	Business logic validation with client
Chart library integration	AWS deployment & infrastructure
Drag-and-drop UI implementation	Security review & penetration testing
Excel import/export logic	Client communication & requirements
Keyword matching engine	Production QA & user acceptance testing
Role-based route guards	Data migration from legacy Excel files
Modal & form validation	Performance benchmarking & optimisation
Column mapping engine	Final go-live decision & sign-off

Why This Approach Works

- **Claude AI understands full-stack web development patterns — it generates React hooks, Laravel controllers, and MySQL queries that work together cohesively, not in isolation**
- **Prompt-driven iteration is faster than traditional code review cycles — a refinement that would take a developer hours to research and implement takes minutes with Claude AI**
- **AI-generated code is consistent — the same modal pattern, the same filter component structure, the same API response format appears across all 9 modules because Claude AI remembered the design system**
- **Business logic can be expressed in plain English — 'classify AQL descriptions using keyword matching, group them, and show the top 5' becomes working PHP logic through Claude AI without the developer needing to design the algorithm independently**

Conclusion

QTS Online demonstrates that AI-assisted development — when applied with a clear methodology, strong business understanding, and disciplined human oversight — can deliver enterprise-grade quality intelligence platforms in a fraction of traditional development timelines.

The platform now processes thousands of real AQL inspection records across 64 vendors for Quince's garment quality team. Every chart on the Dashboard reflects live inspection data. Every defect classification is powered by the 274-rule engine built with Claude AI. Every Excel upload is mapped through the flexible column configuration system designed with Claude AI.

> *QTS Online is not a prototype or a demonstration. It is a production system, live at quincedefectanalysis.qtsonline.com, actively used by quality teams to make better garment sourcing and vendor management decisions every day.*

What QTS Online Proves

- **Enterprise quality intelligence platforms can be built by lean teams when Claude AI handles the implementation heavy lifting**
- **AI-powered defect classification — converting raw free-text AQL descriptions into chart-ready analytics — is achievable through keyword matching designed with Claude AI, without requiring a custom ML model**
- **Flexible Excel integration — the biggest practical barrier to real-world adoption — is solvable with the drag-and-drop column mapping engine built using Claude AI**
- **Consistent UX across a complex multi-module platform is achievable when a single AI system generates all components using the same design patterns**
- **The combination of Claude AI's engineering capability and human business understanding produces better software faster than either could achieve alone**

Built with Claude AI (Anthropic) · Elohim Software Solutions · 2025–2026

quincedefectanalysis.qtsonline.com

admin@qts-online.com · QTS Online — Quince Defect Analysis System