ARISH JOE

Product Designer · Fintech & Payments UX · Enterprise SaaS

COMPLETE PORTFOLIO

9+ Years · Enterprise ERP · Fintech · 1,500+ Organisations

CONTENTS

ARISH JOE

Product Designer · Fintech · Enterprise SaaS

I design products that work for the business, feel right for the user, and scale with both.

9 years designing high-impact digital products across fintech, payments, and enterprise platforms. I turn technical complexity into experiences that feel effortless — from the first click to the last transaction.

Open to Product Designer and Lead Designer roles, remote or Melbourne-based.

WHAT I DO

End-to-End Product Design

From research and strategy through to high-fidelity design, design systems, and engineering handoff. I own the full process — not just the pretty screens.

Fintech & Payments UX

Deep experience in payment flows, transaction interfaces, financial dashboards, POS systems, and merchant-facing products across enterprise and SaaS environments.

AI-Assisted Design

I use Claude AI, Figma Make, and Cursor AI as core parts of my workflow — moving independently from concept to testable solution and accelerating delivery cycles.

EXPERIENCE & CLIENTS

Intuit QuickBooks · Pronto Software — Clients include Petstock, Haymes Paint, RSEA Safety, Shimano Oceania, and 1,500+ enterprise organisations across Australia and internationally.

SELECTED WORK

ENTERPRISE · RETAIL & WHOLESALE · UX STRATEGY

Pronto Xi POS — Role-Based Experience Architecture

Replaced a one-size-fits-all POS interface with a role-based experience model serving retail staff, wholesale operators, and store managers — across 1,500+ enterprise organisations.

3 user groups · 1 platform · 14+ integrated capabilities

FINTECH · PAYMENTS · SAAS

Intuit QuickBooks — Merchant Payment Experience

Led product design across QuickBooks' payments and financial management ecosystem — identifying 40+ high-impact experience gaps and delivering evidence-based design adopted directly into the product roadmap.

Invoicing · Payroll · Banking · Expenses · Reporting

ENTERPRISE · HR & PAYROLL · PLATFORM ARCHITECTURE

Pronto Xi NTS — Unified Employee Experience Platform

Unified six disconnected HR systems — Onboarding, Payroll, Identity Verification, Manager Approvals, Timesheets and Employee Administration — into a single connected platform for 1,500+ enterprise organisations.

45% reduction in admin time · 60+ Agile sprints · 9 cross-functional team layers

ENTERPRISE · ERP · INTERNATIONALISATION

Pronto Xi i18n — The 15-Year Problem. Solved.

Led system re-architecture of Pronto Xi's entire internationalisation foundation — commissioned by the CMO, designed from the ground up, and POC-validated across 24 modules to unlock international market expansion.

Dictionary System · Translation Engine · One-Click Automation · 1,500+ Organisations

ENTERPRISE · ERP · DESIGN SYSTEMS

Pronto Xi Data Grid — Enterprise UX Transformation

Redesigned the core data grid interaction system used daily across 24 Pronto Xi modules — turning a rigid, confusing interface into a decision-support system that actually works.

35–45% efficiency improvement · 24 modules adopted

ENTERPRISE · DESIGN SYSTEMS · IBM CARBON · WCAG 2.2 AA

Pronto Xi — Hybrid Enterprise Design System

Built from scratch over 8 years — a hybrid design system combining IBM Carbon foundations with Pronto Xi-native components, standardising UX across all 24 modules and reducing design-to-dev handoff time by 40%.

24 modules adopted · 40% faster handoff · 8 years · WCAG 2.2 AA compliant

ABOUT ME

Great UX doesn't draw attention to itself. It removes friction so people can focus on their work — not the software.

My Story — What shaped me

The experience that shaped how I think about design was leading the UX transformation of Pronto Xi — taking a legacy desktop ERP used by 1,500+ organisations and rebuilding it as a modern, scalable web platform. That project taught me something I carry into every engagement: enterprise UX isn't about making software look modern. It's about simplifying complex workflows without removing the power that experienced users depend on.

Who influenced me

The people who influenced me most weren't famous designers. They were warehouse staff, payroll officers, retail cashiers, and finance managers — real people navigating real systems under real pressure. Their frustrations became my research. Their workarounds became my design briefs. The defining moment came during usability testing when a user completed a redesigned workflow significantly faster than before and said, simply: “This is how it should have worked all along.” That's the standard I design to.

What I'm looking for

A company that treats design as a strategic partner — not a delivery function. I'm most energised by organisations building products with meaningful, complex workflows, where UX has a measurable impact on both customers and the business. I'm less interested in chasing trends and more interested in building products that solve real problems — and proving it with outcomes.

HOW I WORK

I join technical conversations early. Not to observe — to shape what gets built. The best design decisions happen before a single wireframe is drawn.

I research before I design. Every brief I've ever received had a different real problem underneath it. I find that problem first.

I design for the person who uses it most. Not the occasional user. Not the demo scenario. The warehouse manager running 200 transactions a day. The finance officer reconciling accounts under deadline pressure. That's who I design for.

I think in systems, not screens. A well-designed component that scales across 24 modules is worth more than 24 individually beautiful screens.

I use AI as a core tool, not a novelty. Claude AI, Figma Make, and Cursor AI are part of how I work every day — accelerating discovery, generating variations, and moving from concept to testable solution faster than traditional workflows allow.

BEYOND DESIGN

Outside of work, I'm very much a family person. I play badminton, ride BMW motorcycles, and have a deep appreciation for automotive and supercar design — spaces where engineering precision and aesthetic craft meet in exactly the same way they do in great software. I love travelling, discovering food from different cultures, and volunteering in community initiatives — because understanding people is at the heart of good design.

WHAT I BELIEVE

Design is not decoration. It's the difference between a product people tolerate and a product people trust.

ARISH JOE

My Process — How I Design

I don't start with wireframes. I start with the real problem — because the brief is almost never the real problem.

My process isn't a template I apply to every project. It's a way of thinking I've developed across 9 years of designing complex enterprise products — from a 15-year internationalisation problem nobody had cracked, to a POS platform serving 1,500+ organisations, to a data grid used by warehouse managers, finance teams, and payroll officers every single day.

Every project is different. But the thinking follows the same disciplined sequence: discover the real problem, define the right opportunity, design with evidence, validate before committing, deliver with precision.

1. DISCOVER — I FIND THE PROBLEM NOBODY WROTE IN THE BRIEF

Most design briefs describe a symptom, not a cause. Before I open Figma, I go looking for the real problem — stakeholder interviews, user interviews, on-site observation, shadowing, and a systematic audit of the existing product: heuristic evaluation, workflow analysis, support ticket review, analytics.

From my work: *On the i18n programme, discovery revealed the problem wasn't translation — it was a 15-year-old architectural failure where thousands of UI strings were hard-coded directly into the interface. Every previous attempt had failed because it treated a system problem as a content problem.*

2. DEFINE — I REFRAME THE PROBLEM BEFORE DEFINING THE SOLUTION

I build personas from research evidence, not assumptions. I map current-state journeys — the actual path users take today, including every workaround — then map the future state, and define success metrics before designing anything.

From my work: *On the POS programme, this phase revealed three completely distinct user groups — retail cashiers, wholesale operators, and store managers — each with different goals. That insight drove the decision to build one platform with three role-based experience layers, instead of three separate applications.*

3. DESIGN — I DESIGN SYSTEMS, NOT SCREENS

Once the problem is defined, I design architecture first — information architecture, user flows, interaction models, component logic — before visual design. I run design studios with engineering and business stakeholders to surface constraints early, and I build for the design system from the start, not as an afterthought.

From my work: *On the Data Grid redesign, the design phase began with a complete interaction architecture — mapping every possible user action within the grid before touching visual design. That produced the view management system and contextual controls that became the most-used features post-launch.*

4. VALIDATE — I TEST ASSUMPTIONS, NOT JUST DESIGNS

Validation runs continuously from the first wireframe to the final prototype. I run moderated usability testing on real tasks, heuristic evaluations, and A/B testing where genuine uncertainty exists — and I present findings as evidence, not opinion.

From my work: *On the Data Grid programme, usability testing surfaced the finding that became the centrepiece of the redesign: users weren't making mistakes because buttons were in the wrong place — they had no confidence about what they were looking at. That reframed the whole solution from UI polish to decision-support design.*

5. DELIVER — I DELIVER DESIGNS THAT ACTUALLY GET BUILT THE WAY THEY WERE DESIGNED

I write detailed engineering handoff specs, run design QA against implemented builds, and for complex system-level work, I drive proof-of-concept testing directly in staging environments with engineering — not just handing over a file and disappearing.

From my work: *On the i18n programme, delivery included driving the POC in the test server myself — implementing the dictionary system and translation API integration to validate the one-click automation worked end-to-end before any production commitment.*

THE PRINCIPLE THAT CONNECTS EVERYTHING

Are we solving the right problem, in the right way, for the right people? When the answer is yes at every phase — that's when great products get built.

ARISH JOE

Pronto Software — Platform Overview & Body of Work

8 years. 24 modules. 1,500+ organisations. This is what it looks like when design transforms an entire enterprise platform — not just a screen.

8 Years as Product Designer · 24 Modules Redesigned · 1,500+ Organisations Impacted · 5 Major Initiatives Led

ABOUT PRONTO SOFTWARE

Pronto Software is one of Australia's most established enterprise software companies — founded in Melbourne, serving businesses across Australia and internationally for over 40 years.

Their flagship product, Pronto Xi, is a fully integrated enterprise resource planning (ERP) platform used by 1,500+ organisations across retail, wholesale, manufacturing, distribution, hospitality, and professional services. Clients include Petstock, Haymes Paint, RSEA Safety, Shimano Oceania, Wallace Bishop, Ainsworth Game Technology, Bondor Australia, Vic's Meat, and hundreds more across Australia and internationally.

Pronto Xi covers the full breadth of enterprise operations — Financials, Distribution, Inventory Management, Warehouse Management, Purchasing, Sales Order Management, CRM, Manufacturing, Supply Chain, Payroll, HR, Project Management, Asset & Facility Management, Point of Sale, Service Management, Business Intelligence & Reporting, Workflow & Approvals, Document Management, Mobile Applications, eCommerce, and Pronto Assist (AI Assistant).

It is not a simple product. It is one of the most functionally complex enterprise platforms in the Australian market — and designing for it requires a rare combination of systems thinking, deep user research, and the ability to simplify without removing the power that enterprise users depend on.

MY ROLE AT PRONTO SOFTWARE

Product Designer, Sep 2017 – Jul 2025, Melbourne. I joined as a Product Designer and grew into the Lead Product Designer role — taking end-to-end ownership of the most complex, most strategically significant design initiatives across the Pronto Xi platform.

Pronto Software consistently gave me their hardest, longest-standing, and most commercially critical design problems — the ones that had defeated previous attempts, stalled for years, or required rethinking the platform's foundation rather than redesigning individual screens. I didn't just design features. I re-architected experiences.

I led the transformation of Pronto Xi from a legacy desktop ERP into a modern, scalable web platform — rebuilding the design foundation from the ground up while preserving the depth and power that 1,500+ enterprise organisations depended on.

I worked directly with engineering leads, product managers, business analysts, the Chief Marketing Officer, and executive stakeholders — joining technical conversations early, shaping product direction before wireframes were drawn, and presenting design strategy in business language rather than design language.

I built the Hybrid Design System that standardised UX across all 24 modules, reduced design-to-dev handoff time by 40%, and became the foundation for every subsequent product initiative — including Pronto Assist, Pronto Software's AI Assistant module.

SELECTED INITIATIVES

UX STRATEGY · ROLE-BASED EXPERIENCE ARCHITECTURE · RETAIL & WHOLESALE

Point of Sale (POS)

Redesigned the entire experience architecture for retail cashiers, wholesale operators, and store managers — introducing a role-based model on a single shared platform. One codebase. One design system. Three contextual experiences.

Key outcome: Role-based experience architecture adopted platform-wide · AI-powered Wholesale Intelligence Tier system designed and integrated.

ENTERPRISE UX TRANSFORMATION · DESIGN SYSTEMS · DECISION-SUPPORT DESIGN

Data Grid

Redesigned the interaction system for the most-used interface element in the platform — introducing view management, contextual column controls, density modes, and progressive disclosure, adopted across all 24 modules.

Key outcome: 35–45% improvement in task efficiency · 20–30% reduction in user errors · 24 modules adopted the redesign.

SYSTEM RE-ARCHITECTURE · GOOGLE CLOUD TRANSLATION API · GLOBAL MARKET EXPANSION

Internationalisation (i18n)

Commissioned directly by the CMO to solve a 15-year unsolved architecture problem. Designed the Pronto Lexicon system — centralised dictionary, automated module scanner, translation API integration, one-click automation, administrator manager, live preview, and analytics dashboard.

Key outcome: 15-year problem solved · One-click translation automation · International market expansion unblocked · POC validated.

DESIGN SYSTEM ARCHITECTURE · IBM CARBON · COMPONENT LIBRARY · WCAG 2.2 AA

Hybrid Design System

Built from scratch over 8 years — a hybrid system combining IBM Carbon-influenced components with Pronto Xi-native solutions for enterprise ERP requirements Carbon didn't address.

Key outcome: 24 modules adopted · 40% reduction in design-to-dev handoff · WCAG 2.2 AA compliant · Still running after 8 years.

WHY THIS WORK MATTERS

Enterprise software design is the hardest category of product design. The users are experts. The workflows are complex. The stakes — financial data, operational decisions, business-critical processes — are real. There's no room for confusion, no tolerance for inconsistency, and no forgiveness for interfaces that make experienced users feel like beginners.

Everything I built at Pronto Software was designed for those users — warehouse staff navigating high-volume inventory, finance officers running month-end reconciliation, payroll managers processing weekly pays, operations teams making real-time business decisions.

Their frustration was my research. Their workarounds were my design briefs. Their confidence — when the software finally got out of their way — was the outcome I designed for. That's the standard I bring to every project.

ARISH JOE

Pronto Xi POS — Role-Based Experience Architecture

Stakeholders wanted to add more features to fix a usability problem. Research showed the opposite was true. The problem wasn't missing functionality — it was too much functionality, shown to the wrong people, at the wrong time.

3 User Groups Hidden in One Interface · 1 Shared Platform · 14+ Integrated Capabilities · 1,500+ Organisations

OVERVIEW

Pronto Xi's Point of Sale solution is one of the most functionally rich POS platforms in the Australian enterprise market — fully integrated with inventory, finance, CRM, pricing, promotions, warehouse operations, BNPL, eCommerce, offline selling, and BI reporting across retail, wholesale, and trade businesses.

The problem wasn't capability. The platform had too much of it — delivered to everyone, all the time. A retail cashier processing a transaction under queue pressure was navigating the same interface as a wholesale operator managing account credit and a store manager reviewing KPI dashboards. Same menus. Same workflows. Completely different jobs.

My role: Lead UX Designer — owned end-to-end from research strategy through to design system delivery and engineering handoff. Clients impacted: Petstock, Haymes Paint, RSEA Safety, Shimano Oceania, Wallace Bishop, and 1,500+ organisations.

THE PROBLEM — WHAT WAS ACTUALLY BROKEN

The stakeholder brief was clear: users are frustrated, support tickets are high, training times are too long. The proposed solution was equally clear: add more features. I pushed back — before committing to any solution, I needed to understand what was actually causing the friction.

The existing architecture treated every user as though they performed the same job — despite fundamentally different responsibilities, success metrics, and information needs. The result: visual clutter, excessive navigation for simple tasks, critical information buried, high cognitive load, staff working around the system, long onboarding, and rising support tickets.

The insight that reframed the entire project:

Users weren't frustrated because features were missing. They were frustrated because the right features weren't surfaced to the right people at the right moment. Adding more functionality would have made the problem measurably worse.

RESEARCH — HOW I GOT TO THE REAL PROBLEM

A structured, mixed-method discovery programme across multiple enterprise retail and wholesale clients, designed to answer specific strategic questions: How do retail staff perform under pressure? How do wholesale operators fundamentally differ? What do managers need for fast decisions? Which workflows generate the most friction, and why?

- Customer research: stakeholder interviews, user interviews, contextual inquiry, on-site observation, shadowing, journey mapping, persona validation
- Product evaluation: heuristic evaluations, product audits, competitor benchmarking, current-state vs future-state mapping
- Design collaboration: stakeholder workshops, design studios, design critiques, prioritisation workshops
- Validation: wireframe testing, prototype testing, moderated usability testing, A/B testing

What the research found: three completely distinct user groups emerged — each with different goals, different workflows, and different definitions of success.

THE THREE USERS

Retail Staff — Designed for Speed

Wholesale Operators — Designed for Account Management

Store Managers — Designed for Decision Making

THE STRATEGIC DECISION — ONE PLATFORM, THREE EXPERIENCES

This was the most consequential design decision of the entire project, and the one I'm most proud of.

Option A — Three independent POS applications

Clean separation, full control per persona. The problem: three codebases, three design systems, exponential implementation cost — impossible to scale across 1,500+ client organisations.

Option B — One shared framework with role-based experience layers

A single platform and codebase, dynamically adapting navigation, dashboards, permissions, and feature visibility based on user role. One platform. One design system. Infinite contextual experiences.

I recommended Option B. The business approved it. This was a product strategy call, not just a UX call — it required understanding technical feasibility, implementation cost, and scalability across 1,500+ organisations, validated across product, engineering, and business stakeholders before a single screen was designed.

THE DESIGN — WHAT WE BUILT

Retail Experience — Stripped back to essentials

Fast checkout flow, large touch targets, barcode scanning as the primary interaction, clear error states handled with minimal steps.

Wholesale Experience — Account-first navigation

Customer lookup leads to order history, pricing agreements, and credit status before any transaction begins — designed for relationship management, not speed.

Manager Experience — Dashboard-first

KPI visibility above the fold, drill-down from summary to detail — answering operational questions without navigating through transactions.

All three experiences were built from the same component library — consistent interaction patterns, shared tokens, unified accessibility standards. The role determined what was shown, not how it was built.

PROGRESSIVE DISCLOSURE

The unifying interaction principle across all three experiences: surface what's needed for the task at hand, reveal depth only when required. For retail, the checkout shows nothing but the transaction. For wholesale, account summary leads to order detail leads to pricing history. For managers, KPI headline leads to store breakdown leads to individual transaction detail. This reduced cognitive load across all three user types without reducing any capability.

FROM MEMORY TO INTELLIGENCE — AI-POWERED WHOLESALE TIER SYSTEM

During wholesale research, I watched operators switch between multiple screens — payment history, order frequency, credit status — before making a single decision. The data existed. It just wasn't where they needed it.

I designed a Trader Intelligence Panel embedded directly in the wholesale order flow. Four tiers — Bronze, Silver, Gold, Platinum — reflecting payment reliability, order consistency, tenure, and order value, each surfacing a clear recommended action.

I pitched it as a revenue protection tool, not a UX feature — showing scenarios where a Platinum trader received the same friction as a new account, or an operator missed a reorder window with no signal. That reframe got the room aligned immediately. Operator-only, invisible to customers, and built as the foundation for predictive reorder alerts and credit risk scoring feeding into the Pronto Assist AI roadmap.

IMPACT & OUTCOMES

Experience outcomes

- Measurably reduced cognitive load across all three user groups in usability testing
- Faster task completion for retail staff, improved workflow efficiency for wholesale operators

- Stronger KPI visibility for managers without drilling into transactions

Business outcomes

- Reduced support ticket volume and faster staff onboarding
- Greater platform scalability — one codebase serves infinite role configurations across 1,500+ clients
- Increased enterprise licensing opportunities — the role-based model became a differentiating sales feature

Strategic outcome

The role-based experience architecture became the foundation for future Pronto Xi product development — a scalable model for how the platform adapts to user context across all modules, not just POS.

WHAT THIS DEMONSTRATES

- I pushed back on the brief — research proved the stakeholders wrong, and I had the confidence to redirect a major initiative
- I made a product strategy call, not just a design call — the Option A vs B decision required understanding technical architecture and cost at scale
- I designed a system, not screens — decisions that scale long after any individual screen is redesigned

CONCLUSION

Sustainable UX transformation is not achieved by adding functionality. It's achieved by understanding how different users work and designing experiences that match user goals — not system capabilities. One platform. Three experiences. Designed for the people who actually use it.

ARISH JOE

NTS — Unified Employee Experience Platform

Six enterprise systems. Five user groups. Zero connection between them. I designed the platform that unified them all.

45% Admin Time Reduced · 6 Systems Unified · 9 Cross-Functional Team Layers · 1,500+ Organisations

THE CONTEXT

At Pronto Software, employee onboarding wasn't one broken screen. It was six broken systems that had never been designed to work together — HR, Payroll, Employee Onboarding, Manager Approvals, Timesheets, and Employee Administration, each with its own interface, workflows, and data.

A new employee joining an organisation running Pronto Xi triggered a cascade of manual handoffs, email follow-ups, and disconnected tasks across every one of those systems simultaneously. Onboarding typically took 2–3 days to complete. New hires regularly started their first day without full system access, without payroll configured, and without clear guidance on what they were supposed to do next.

The business logic within Pronto Xi was sound. The user experience was completely fragmented. NTS was the platform I proposed, designed, and delivered to solve this — connecting HR, Payroll, Manager Approvals, Identity Verification, Tax Setup, Timesheets, and Employee Administration into a single, coherent workspace for the first time in Pronto Xi's history.

WHO NTS WAS BUILT FOR

Employees

A guided, progressive onboarding journey from offer acceptance to first productive day — clear next steps, visible progress, no anxiety.

HR Administrators

A centralised onboarding dashboard with real-time status across every new hire — no more email follow-ups or chasing approvals.

Payroll Officers

Structured, validated payroll setup data flowing directly from onboarding — eliminating the first-pay-run errors caused by incomplete manual entry.

Hiring Managers

In-platform approval workflows with full context and automatic escalation, replacing untracked email approvals with a transparent, accountable process.

THE STRATEGIC PIVOT — FROM REDESIGN TO PLATFORM

When I presented discovery findings to leadership, the expected response would have been: redesign the onboarding screens. I proposed something different.

The core architectural principle: separate the experience layer from the business logic layer. The underlying business rules — payroll rules, compliance, employment agreements — stayed exactly as they were. What changed was the experience layer entirely: one connected workspace, one source of truth, knowing where every onboarding stood, for every person, at every moment.

Replacing the ERP backend would have taken years and cost millions. Redesigning individual screens would have produced six better screens still connected by broken handoffs. Separating the experience layer preserved everything that worked while completely transforming everything that didn't.

WHAT NTS UNIFIED — SIX FLOWS

- Employment Offer & Welcome Portal — digital offer acceptance with a guided, trackable welcome experience
- Employee Onboarding Dashboard — real-time status visible to employee, manager, and HR simultaneously
- Identity Verification — a guided digital flow replacing physical document exchanges and disorganised email attachments
- Tax & Payroll Setup — a progressive, validated flow replacing confusing forms that caused first-pay-run errors
- Manager Approvals — structured, tracked, escalating approval workflows replacing untracked email chains
- Timesheets & Ongoing Employee Interactions — extending the connected workspace beyond day one

THE DELIVERY — 60 AGILE SPRINTS, 9 TEAM LAYERS, 2+ YEARS

NTS was delivered through nine distinct discipline layers — UX Design, Product Management, Technical Architecture, Data Layer, Service Layer, Front-End Engineering, Infrastructure, QA, and Development — all collaborating across 60+ Agile sprints over two years.

I was the bridge between every other team. I collaborated on API contracts, shaped architecture decisions, and ran design QA. That cross-layer presence — a designer who understood the technical architecture, not just the interface — is what kept NTS coherent across two years and nine team layers.

IMPACT & OUTCOMES

- 45% reduction in onboarding administration time — measured across HR, Payroll, Managers, and Employees
- Payroll errors dramatically reduced through the guided Tax & Payroll Setup flow
- Manager approval delays eliminated — structured in-platform approvals replaced untracked email
- New employee anxiety significantly reduced — clear guidance and visible progress before day one
- New licensing opportunities created for HR and Payroll modules through the NTS platform model
- The experience-layer/business-logic separation principle became Pronto Software's template for future modernisation — subsequently applied to Supply Chain and the i18n programme

CONCLUSION

New employees deserved a better first day. HR teams deserved to stop chasing approvals by email. Payroll officers deserved complete, accurate data on day one. Managers deserved visibility without asking. NTS didn't just fix onboarding — it proved that enterprise ERP could be transformed by separating the experience from the system.

ARISH JOE

Pronto Xi Data Grid — Enterprise UX Transformation

Pronto Xi ERP · Enterprise SaaS · B2B Product Design · Design system powered by IBM Carbon Design principles, adapted for enterprise ERP.

↑ 35–45% Task Efficiency · ↓ 20–30% User Errors · ↓ Training Dependency · ↑ Decision Confidence

OVERVIEW

Redesigned the core data grid experience in Pronto Xi — used daily by enterprise users to manage high-volume, complex data across finance, operations, and inventory workflows. The goal: transform a dense, inefficient, error-prone interface into a scalable, intuitive, high-performance interaction system.

PROBLEM

The existing data grid was a critical but broken experience — this wasn't just a UI issue, it was a system-level usability problem affecting productivity at scale.

- Difficult to scan and interpret large datasets
- High cognitive load due to poor hierarchy and structure
- Inefficient workflows requiring excessive clicks and manual effort
- Inconsistent behaviours across modules
- High dependency on training and support

USERS

A multi-role system with different goals but shared interface constraints: Finance teams (payroll, accounting), Operations teams (inventory, logistics), and Admin and managers (reporting, oversight).

MY ROLE

Lead UX Designer — owned the initiative end-to-end: Discovery → Problem framing → Design → Validation → Delivery. Worked closely with Product Managers, Engineers, and Business stakeholders.

UX PROCESS

1. Discover

Methods: user interviews (finance and operations users), usability testing on the existing grid, support ticket and feedback analysis, workflow observation.

Key insights: users struggled with finding, filtering, and acting on data quickly. Most errors came from misinterpretation, not system failure. Heavy reliance on memory and training instead of UI guidance.

The insight that reframed everything:

The grid wasn't supporting decision-making — it was forcing users to think too much.

2. Define

Reframed the problem: design a data interaction system that supports fast, confident decision-making. Design principles: clarity over density, reduce cognitive load, progressive disclosure, consistency across modules, speed of interaction.

3. Design — System Thinking Approach

- Information Hierarchy & Readability — clear visual hierarchy, improved scanability for large datasets
- Smart Filtering & Sorting — simplified filtering with contextual controls, fewer steps to find relevant data
- Inline Actions & Editing — direct interaction within the grid, reduced navigation and context switching

- Feedback & Error Prevention — inline validation and system feedback, reduced errors and uncertainty
- Scalable Interaction Patterns — reusable grid behaviours across modules, foundation for a design system component

AI-Driven UX — forward-looking layer

Suggested filters based on user behaviour, smart defaults for frequently used actions, contextual assistance such as highlighting anomalies in data. Goal: reduce effort, not just improve UI.

4. Prototype & Validate

Methods: interactive Figma prototypes, task-based usability testing, iterative feedback loops. What we tested: time to complete tasks, error rates, ease of navigation and understanding.

5. Deliver & Scale

Worked with Engineering to ensure feasible, scalable implementation. Defined reusable patterns for design system integration. Rolled out improvements incrementally across modules.

DESIGN APPROACH — WHAT I BROUGHT

Systems Thinking

Designed not just a component, but a scalable interaction system.

Product Thinking

Focused on user outcomes and business impact, not just UI improvements.

AI Mindset

Explored how automation and intelligence can reduce effort.

Practical UX

Balanced user needs, technical feasibility, and business constraints.

CONCLUSION

In complex systems, the problem is rarely the interface — it's how users think, decide, and act within it. By shifting from 'table design' to 'decision-support system design', we significantly improved usability, efficiency, and confidence.

By simplifying complexity, improving usability, and creating a more intuitive interaction model, this redesign enhanced daily productivity for enterprise users and established a stronger UX foundation for broader platform modernisation.

NDA Statement: To respect confidentiality, certain details such as data, visuals, and system specifics have been adapted or abstracted. The focus is on showcasing design approach, thinking, and impact.

ARISH JOE

Pronto Xi i18n — The 15-Year Problem, Solved (Pronto Lexicon)

Pronto Software had been unable to crack internationalisation for 15 years. Every previous attempt had failed — not because of lack of effort, but because the problem had been misdiagnosed. It wasn't a translation problem. It was a system architecture problem.

15 Years Unsolved · 5+ Global Regions · One-Click Automation · Google Cloud Translation API

OVERVIEW

Commissioned directly by the Chief Marketing Officer, I led the redesign of Pronto Xi's internationalisation (i18n) experience — the Pronto Lexicon project — to support multi-language, multi-region users while maintaining usability, consistency, and scalability. The goal: transform i18n from a fragmented, inconsistent implementation into a system-driven, scalable UX approach across the platform.

My role: Lead UX Designer, owning the initiative across Discovery → Problem Definition → UX Framework → Validation → Rollout. Collaborated with Product & Engineering, translation teams, and global stakeholders.

THE PROBLEM

The product faced major challenges expanding globally: hard-coded text across the system, inconsistent translations and terminology, UI breakage due to text expansion (German, French), poor readability in non-English languages, and no standard process for managing translations — causing poor global user experience, increased support issues, and high maintenance effort.

Users: global customers across 5+ regions, non-English speaking users, and internal teams managing translations.

DISCOVER

Methods: a UI audit to identify hard-coded text and inconsistencies, analysis of translation issues across modules, feedback from global users and stakeholders, and a review of localisation workflows.

Key insights: i18n was treated as a technical afterthought, not a UX problem. Translations failed due to lack of context and structure. The UI was never designed for language flexibility.

The insight that reframed everything:

Internationalisation is a system design problem, not just translation.

DEFINE

Reframed the problem: not “fix translations” but “design a scalable i18n system that supports global usability and consistency.” Key goals: ensure the UI works across languages, improve translation quality and consistency, enable scalable localisation, and maintain usability across regions.

DESIGN — THE I18N UX SYSTEM

1. Standardised Content Structure

Eliminated hard-coded text at the dictionary level. Introduced structured, reusable content keys, enabling scalable translation management.

2. UI for Language Flexibility

Designed layouts to handle text expansion and contraction, ensuring responsive behaviour across languages and preventing UI breakage.

3. Consistent Terminology System

Standardised key terms across the platform, ensuring consistent language across every module.

4. UX Guidelines for i18n

Defined best practices for text length, labels, error messages, and date/time formats.

API & INTEGRATION LAYER

To support a scalable, efficient internationalisation system, I worked closely with Engineering to integrate translation APIs and automation workflows — leveraging the Google Cloud Translation API for dynamic translation support, designing workflows for automated translation generation and fallback handling, and enabling seamless integration between UI content and translation services.

The centrepiece: a one-click automated translation process for system administrators — replacing what had previously taken weeks of manual, error-prone effort with a single, auditable, repeatable action.

PROTOTYPE & VALIDATE

Methods: multi-language UI testing, layout testing for text expansion, and feedback from global users — focused on readability, layout stability, and consistency.

DESIGN APPROACH

Systems Thinking

Designed i18n as a cross-product system, not a feature.

Scalability First

Ensured solutions worked across multiple languages and regions.

Product Thinking

Balanced user experience, business expansion, and technical constraints.

AI-Forward Thinking

Explored intelligent localisation and automation.

IMPACT

- Unlocked international market expansion previously blocked by i18n limitations
- Positioned Pronto Xi competitively against SAP, NetSuite, and Microsoft Dynamics on internationalisation capability
- One-click translation automation reduced a multi-week manual process to a single repeatable action
- Established a scalable i18n foundation adopted across 24 Pronto Xi modules

KEY LEARNING

Internationalisation is not about translating words — it's about designing systems that work across cultures, languages, and contexts. By treating i18n as a UX and system design problem, we enabled true global scalability.

NDA Statement: To respect confidentiality, certain details such as data, visuals, and system specifics have been adapted or abstracted. The focus is on showcasing design approach, thinking, and impact.

ARISH JOE

Intuit QuickBooks — Gold & Platinum Tier Merchant Experience

QuickBooks' most valuable customers — Gold and Platinum tier merchants — were paying for features they didn't know existed. I was brought in to fix that.

THE BRIEF

In 2025, I joined Intuit QuickBooks on a 6-month contract as a Product Designer, embedded directly within the payments and financial management product team. My focus was specific: the Gold and Platinum tier merchant experience — QuickBooks' highest-value customers, running complex financial workflows across invoicing, payroll, banking, expenses, reporting, and customer management every single day. They weren't getting a product that matched what they were paying for. My job was to change that.

THE PROBLEM — WHAT I FOUND

Gold and Platinum tier merchants are not casual users. They live inside QuickBooks for hours every day. The product wasn't built for how they actually worked. Three critical problems emerged from my research:

Problem 1 — Features they paid for were invisible

The most commercially damaging finding of the entire engagement. Gold and Platinum customers were describing features already included in their subscription as “something I'd need to pay extra for.” The interface never surfaced them at the right moment. Result: underutilised subscriptions, frustrated merchants, and avoidable support calls.

Problem 2 — Daily tasks took too long

High-frequency workflows were designed for occasional users, not power users. Approving a batch of invoices required opening each one individually. For a finance manager processing 50+ invoices a day, these were hours of lost productivity every week.

Problem 3 — No experience difference between tiers

A Gold tier merchant experienced the same interface as a basic plan user — nothing reflected the value of their investment. If the product doesn't feel different, the tier doesn't feel worth it.

WHO I WAS DESIGNING FOR

The Finance Manager

Runs daily financial operations for a mid-to-large business. Uses QuickBooks 4–6 hours every day. Definition of success: “I got through my entire morning financial workflow without thinking about the software once.”

The Business Owner / Director

Oversees financial performance at a strategic level. Definition of success: “I opened QuickBooks, saw exactly what I needed, made a decision, and moved on.”

THE RESEARCH — HOW I IDENTIFIED THE REAL PROBLEMS

I didn't assume. I investigated.

- Moderated usability testing — observed merchants completing real tasks across all six modules, timing completion, noting hesitation and workarounds
- Stakeholder interviews with account managers and support teams who knew exactly where the friction lived
- Support ticket analysis — the highest-frequency tickets pointed directly to feature discoverability and invoice approval friction
- Heuristic evaluation of all six modules, scored by severity, frequency, and retention risk
- Analytics review — the gap between what merchants had access to and what they actually activated

The finding that changed everything: merchants weren't calling support because things were broken. They were calling because they couldn't find what they needed — and often, what they needed was already there.

THE DESIGN — WHAT I BUILT

The most impactful change was also the most conceptually simple: surface the right features to the right merchants at the right moment. I designed a contextual prompt system — intelligent nudges appearing inline, at the exact moment a merchant would benefit from a feature they hadn't activated yet.

A Gold merchant generating their third manual report this week sees: “You can save this as a template and generate it in one click.” A Platinum merchant approving invoices one by one sees: “You have bulk approval available on your plan.” The feature was always there. Now it's findable — without a support call.

One-Click Invoice Approvals

Before: Select invoice → Open → Review → Approve → Return to list → Repeat × 50. After: Select all → Review summary → Approve all → Done. A task that took 15+ steps now takes 4 — hours returned to a finance manager's week, every week.

Unified Customer Financial Panel

When a customer calls to query a payment, the account manager needs their complete financial picture immediately. I designed a single view surfacing everything relevant to that customer relationship, accessible in one click from the customer record.

Consolidated Tier Dashboard

A command centre for both Gold and Platinum merchants — cash flow, invoices requiring action, upcoming payroll, and one-click access to the most-used reports, all in one place.

DESIGNING FOR INTUIT ASSIST

During my engagement, Intuit was actively rolling out Intuit Assist — a virtual team of specialised AI agents (Accounting, Payments, Customer, Finance, Project Management). Rather than designing around it, I designed with it in mind.

The contextual feature prompts I designed complement Intuit Assist's intelligence layer. The one-click invoice approvals aligned directly with the Payments Agent's automation model. The consolidated dashboard created the visual foundation for the Finance Agent's daily summaries and anomaly flags to surface within.

THE ROADMAP

All 20+ findings were organised into a prioritised roadmap across three horizons and presented to QuickBooks product leadership:

- Now — critical experience failures: feature discoverability, bulk approvals, unified customer view
- Next — workflow efficiency: report templates, consolidated dashboard, cross-module navigation
- Later — tier differentiation and AI layer: advanced forecasting, AI-assisted anomaly detection

Every item came with a corresponding design — wireframes, flows, and high-fidelity mockups. Not abstract recommendations. Actionable designs ready for engineering handoff.

THE IMPACT

- 20+ experience gaps identified, evidenced, and designed against — adopted directly into the QuickBooks product roadmap
- Feature discoverability failure resolved — merchants can now find and activate capabilities they're already paying for
- Daily workflow efficiency significantly improved for high-volume merchants
- Support call reduction built into every design decision
- Tier value made visible — Gold and Platinum merchants experience a product that reflects their investment for the first time

WHAT THIS ENGAGEMENT DEMONSTRATES

I design for the user who uses the product most, not the user who uses it least.

Every design decision I made was evaluated against Gold and Platinum reality — high volume, high frequency, high stakes.

I find the commercially significant insight, not just the usability issue.

The feature discoverability finding isn't a UX finding. It's a revenue finding — the difference between a designer who improves interfaces and one who impacts the business.

I design systems, not screens.

Contextual prompts, tier-aware components, report template architecture — patterns that scale across every module, every tier, every merchant type.

CONCLUSION

QuickBooks' best customers deserved a better product. Not more features — a smarter, faster, more considered experience that recognised who they were and what they needed to do. That's what I designed.

Contract Product Designer · Intuit QuickBooks · 2025–2026 · 6-month engagement

NDA Statement: To respect confidentiality, certain details including data, visuals, and system specifics have been adapted or abstracted. The focus is on showcasing design approach, thinking, and impact.

ARISH JOE

Pronto Xi Dashboard — Role-Based Workspace Redesign

Every user opened the same dashboard, regardless of what their job actually was. I redesigned it as a platform, not a page — one framework, twenty-four modules, a different workspace for every role.

45% Fewer Unnecessary Widgets · 30% Faster Task Access · 4→1 Navigation Steps · 24 Modules, One Framework

THE PROBLEM

When I started looking at the legacy Pronto Xi dashboard, I found a single, static landing page — every user, every role, every module, staring at the same generic layout regardless of what they actually needed to get done that day.

I didn't need to run a research session to find the first problem — it was sitting right there on the screen. A single view carried eighteen portlets simultaneously, all rendered at identical visual weight, with nothing telling the user what needed attention right now versus what was just there for reference. I watched a critical SLA breach alert appear as a modal, then again in a portlet, then a third time in the activity feed — one real event, three uncoordinated warnings. The interface had access to every kind of enterprise data. It had no concept of who was looking at it.

That's when I realised this wasn't a visual design problem. It was an architectural one: Pronto Xi had never designed a dashboard system — only a dashboard page, copied forward as the platform grew to 24 modules.

THE REFRAME

The brief I was handed could easily have stopped at “make the dashboard look modern.” I chose to design around a different question instead: what does this specific person need to see the moment they log in, and how do I build that without redesigning it 24 separate times? That second half of the question is what turned this into a platform problem for me, not just a screen problem.

WHAT I BUILT

Role-based dashboards, not one-size-fits-all

I designed it so a Payroll Officer lands directly on upcoming pay runs and approvals, a Finance user sees AP/AR/GL/P&L immediately, and a Sales Rep sees exactly five focused metrics instead of the ten identically-weighted KPIs the legacy screen threw at everyone. I built the role model as a first-class control in the interface itself — Sales Rep, Sales Manager, and Service Manager switch between their own workspaces from the same screen, labelled “Role-Based Workspace · One CRM Framework.”

A single component framework behind every version of it

I built every role-based dashboard from the same underlying components — KPI cards, widgets, charts, activity lists — configured differently per role rather than rebuilt from scratch each time. One framework, reused, never duplicated.

Responsive by design, not by exception

Desktop carries multi-column analytical layouts; tablet and mobile automatically reflow into prioritised card layouts — same data, restructured for the device.

Personalisation without fragmentation

Five visual themes, including Dark Mode, layered on the same component system. Every theme runs through the same token set, so switching themes changes only colour — never layout, hierarchy, or behaviour.

THE JOURNEY — THREE ROLES, ONE FRAMEWORK

Same login, same framework, three different jobs get done. I compressed each to the moment that matters: land, spot the insight, act on it.

Sales Rep — Marcus Chen

Sees	Dashboard, 5 KPI cards, pipeline promoted	Quota gap ($110k) + AI flags the priority deal	Opens the flagged deal
Thinks	“Where am I at?”	“I didn't have to work that out myself”	Untouched 4 days — now unmissable

Sales Manager

Sees	Team leaderboard — EDB 92%, RR 118%, CJ 47%	AI flags CJ's stalled deals, suggests a review	Books a pipeline review with CJ
Thinks	“How's the team tracking?”	“CJ needs support, not just a number”	One flow, no detour

Service Manager

Sees	Case queue, breach count surfaced first	CSE-1180 breached 16 days — AI flags churn risk	Escalates the case
Thinks	“What needs me first?”	Same breach that once fired three alerts	One clear action, not three duplicates

Same components underneath all three. What changes is entirely what the framework decides to surface.

PLATFORM THINKING

This is the same architectural principle I carried through every major Pronto Xi initiative I led: build the system once, let every module inherit it. I didn't design 24 different dashboards. I designed one configurable dashboard framework — reused across Finance, Payroll, HR, CRM, Inventory, and Manufacturing. Developers configure a new dashboard rather than build one from nothing; QA validates one component once, rather than 24 separate implementations.

The same architecture is also what let me turn the dashboard into a usable surface for Pronto AI. The Quota to Goal card doesn't just display data — it feeds a generated recommendation: which deal to prioritise, and why. That only works because the dashboard already knows who's looking at it and what they're responsible for.

IMPACT

45% fewer unnecessary widgets displayed per user — the dashboard now shows what's relevant to the role logged in, not everything the platform is capable of showing.

30% faster access to daily operational tasks, with key workflows reduced from four navigation steps down to one.

One framework supporting 24 modules — new modules configure into the existing dashboard system rather than requiring a bespoke build.

Consistent behaviour regardless of role, module, or device — standardised components simplified regression testing platform-wide.

CONCLUSION

A dashboard isn't one screen. At platform scale, it's an architecture decision. I didn't design 24 dashboards. I designed the system that lets 24 modules each feel like they were built specifically for the person using them.

ARISH JOE

Pronto Xi — Hybrid Enterprise Design System

Anyone can create a style guide. Building a design system that 24 modules actually adopt — and that engineers trust enough to build from — that's a different problem entirely.

24 Modules Adopted · 40% Faster Dev Handoff · 8 Years Maintained · WCAG 2.2 AA Compliant

THE CONTEXT

When I joined Pronto Software, Pronto Xi had no design system — only informal, undocumented conventions that had accumulated across years of individual module development. Every module looked slightly different. Every designer made slightly different decisions. Every engineering team implemented slightly different components.

At small scale this is manageable. At enterprise scale — 24 integrated modules, 1,500+ client organisations — it becomes a serious business problem: inconsistent components across modules, accessibility compliance that existed in some places and not others, and new features requiring design decisions already made and forgotten elsewhere on the platform.

I built the foundation that made the platform's modernisation coherent, scalable, and maintainable — from scratch, over 8 years.

WHAT A DESIGN SYSTEM ACTUALLY SOLVES

Most people think design systems are about visual consistency. That's the visible output — not the real value. At enterprise scale, a design system solves four business problems:

Speed

Documented, tested, ready-to-use components mean designers and engineers stop reimplementing the same button for the fifteenth time.

Consistency

Users learn an interface once and apply that knowledge everywhere — cognitive load drops, confidence rises.

Scalability

A design decision made at system level — a colour token, a spacing scale — changes everywhere at once when updated, instead of requiring a months-long manual project.

Governance

Documented standards mean new team members onboard faster and the platform stays coherent as it evolves.

WHAT I BUILT

Design Tokens — Three-Tier Architecture

Primitive tokens (raw values), semantic tokens (design decisions), and component tokens (specific applications). Changing a brand colour requires updating one primitive token — everything downstream inherits the change automatically, across every module.

Enterprise Colour System

A framework engineered specifically for data-heavy, workflow-intensive ERP interfaces — categorical and sequential data visualisation scales tested for colour-blindness accessibility, consistent semantic status colours across all 24 modules, light and dark theme support, and full WCAG 2.2 AA validation built in from the start, not retrofitted.

Component Library

Form components, data display components (built for enterprise density, not general-purpose use), navigation, feedback, and action components — every one built using Figma's variant system, Auto Layout, and variables, so the Figma file itself is the specification.

Governance Model

Contribution guidelines, change management process, per-component usage guidelines, and design-to-development handoff standards — the system that keeps a design system coherent as it grows, rather than slowly fragmenting.

THE IBM CARBON DECISION

Pronto Xi partially adopted the IBM Carbon Design System as a reference — but a full adoption was deliberately ruled out. Carbon is built for IBM's product context; applying it wholesale would have broken existing workflows that 1,500+ organisations depended on daily and created a migration burden that would have delayed delivery significantly.

The decision: adopt Carbon selectively where its solutions were proven and applicable — data tables, form components, notification patterns, button hierarchy — and build native components where Pronto Xi's enterprise ERP requirements went beyond what Carbon was designed to handle, particularly around data grid density and complex workflows.

The result is a hybrid system — Carbon-influenced where Carbon was superior, Pronto Xi-native where the platform's requirements demanded it. Not wholesale adoption. Informed, deliberate, context-aware integration.

DRIVING ADOPTION ACROSS 24 MODULES

- Started with the highest-visibility module — the Data Grid redesign became the proof of concept that convinced other teams to follow, backed by a 35–45% efficiency improvement
- Made the system faster to use than designing from scratch — adoption wasn't mandated, it was incentivised
- Maintained it actively — updating components as requirements evolved, deprecating patterns that no longer served their purpose
- Documented at the point of use — usage guidelines live in Figma next to the components they document, not in a separate page nobody reads

IMPACT

- 40% reduction in design-to-dev handoff time, measured across three major module releases
- Design cycles that previously required days of foundational work began with real product problems immediately
- Users navigating between modules — Finance to CRM to Inventory to HR — experience a coherent, predictable interface
- WCAG 2.2 AA compliance became a baseline guarantee every module inherits, not a per-project checklist
- 1,500+ enterprise organisations experience a product built on this shared foundation — still active after 8 years

CONCLUSION

A design system is not a deliverable. It's a living foundation that makes everything built on top of it faster, more consistent, and more scalable. This one has been running for 8 years, across 24 modules, for 1,500+ organisations.

THE THROUGHLINE

Seven projects. I never took the brief at face value.

In every case study in this document, I was handed a stated problem — and in every case, the stated problem wasn't the real one.

On POS, stakeholders wanted more features. Research proved the real problem was too many features shown to the wrong people, at the wrong time.

On NTS, the ask was “improve onboarding.” The real problem was six disconnected systems that had never been designed to work together.

On the Data Grid, the brief was a UI improvement. It was actually a decision-support system that had never been designed to support decisions.

On i18n, the brief was to “fix translation.” It was actually a 15-year system architecture failure that three previous attempts had misdiagnosed.

On Intuit QuickBooks, the brief was to improve the Gold and Platinum tier experience. The real problem was merchants already paying for capabilities they didn't know existed — the interface was failing them, not the feature set.

On the Dashboard, the brief was a visual refresh. The real problem was that every user, regardless of role, was looking at the same generic page — a platform-scale architecture problem wearing a screen-deep brief.

On the Design System, the easy version was a style guide. What actually got built was a governed practice that's still running eight years later.

That's the pattern across all of it: find the real problem before designing the solution, then build the system — not just the screen — that solves it permanently.

What I'm Looking For

A team building something complex enough that the obvious answer is usually wrong — where the value isn't in making things look polished, but in finding the real problem underneath the one everyone's already agreed on.