QBurst | A High AI-Q™ Company

m·box

Automating SaaS Replatforming with Agentic AI

Learn how MessageBox used custom AI to complete a multi-month JavaScript to TypeScript migration and Radix UI rebuild in just two weeks.

Overview

- **An end-to-end replatforming** was delivered using Claude Code and custom AI skills to transition a legacy web application from JavaScript to TypeScript and Radix UI.
- **~2 weeks to completion** was achieved, successfully compressing a massive migration process that previously took over 4 months.
- **1 solo developer** executed the entire migration scope with perfect UI consistency, eliminating the need for a multi-developer engineering team.
- **Full TypeScript coverage** replaced a completely JavaScript-based codebase, securing long-term type safety.

m·box MessageBox

Client Profile

Messagebox is a HotelTech SaaS platform built for hospitality teams to manage service requests, employee communications, guest interactions, and housekeeping operations. The core product relies on a primary web application paired with a companion mobile app for status updates.

A Massive Solo Migration Challenge

- **Scale of the Rewrite:** Moving a substantial legacy JavaScript codebase to TypeScript while switching the UI component foundation to Radix UI required massive parallel effort across screens, API integrations, and data models.
- **Consistency Risk:** With many screens to rebuild, manually maintaining a coherent design language across the platform was error-prone and time-consuming.

- **Team Size Constraint:** A solo developer was responsible for delivering the entire migration, making the scope nearly impossible with conventional approaches.
- **Speed Expectations:** The previous legacy migration had taken more than four months; business timelines for next generation major upgrade demanded a significantly faster turnaround.

Claude Code as the Execution Engine

We approached the legacy migration by wiring Claude's capabilities into the entire development workflow from the start, rather than using AI for isolated tasks. Using Claude Code (powered by the Claude Opus 4.8 model) as the core execution engine, we drove the agentic coding work to generate screens, correct output, scaffold new routes, and iterate on TypeScript types.

The developer utilized Claude Code not just to write code, but to review and validate what had already been generated. This created a tight, automated loop between generation and correction within the same workflow, ensuring the output rarely needed significant manual rework.

Key Features & Custom Skills

To provide the AI with grounded domain knowledge, three purpose-built Claude Skills formed the backbone of the approach:

- **Theme and Pattern Extraction:** A custom skill analyzed the existing legacy application to extract its design language, color tokens, spacing, component patterns, and visual conventions, giving Claude a grounded understanding before generating code.
- **API Layer Wiring:** A skill that encoded how the app's API layer was structured, including data models and integration patterns, allowing generated TypeScript components to connect correctly to the backend instantly.
- **Screen Generation:** Guided Claude to generate new screens that perfectly matched the extracted visual system while producing valid TypeScript using Radix UI primitives.

Impact

- **~92% faster request creation for hotel staff:** Cut request creation time from ~1 minute to ~5 seconds using Claude-assisted AI, saving roughly 1.5 hours daily across 100 requests.
- **8x less engineering effort:** Reduced full replatforming effort to ~80 engineering hours (down from 640+) and reached production in 2 weeks instead of 4+ months.
- **No team scale-up required:** Eliminated the need to hire or contract additional developers by delivering the complete migration with a single engineer.
- **560+ hours returned to the product roadmap:** Reinvested saved engineering time directly into building high-value capabilities for hotel customers, such as AI-assisted request creation.
- **Cheaper to extend going forward:** Decreased future feature development effort by implementing full TypeScript coverage and a reusable Radix UI component library.
- **More flexible views for hotel teams:** Unlocked customizable layouts and richer detail displays across rebuilt screens for hotel staff.