Nadeem Ashraf

Forward Deployed Engineer · Full-Stack · JavaScript · TypeScript · React · Node.js · Python
Lucknow, India · Open to relocation (UAE, Saudi Arabia, Singapore, Malaysia) · Visa sponsorship required
+91 79052 17447 · official.nadeemashraf@gmail.com · github.com/devnadeemashraf · linkedin.com/in/nadeemashraf06

Summary

Forward deployed engineer, five years with 30+ US freight-broker customers. I turn ambiguous requirements from ops teams, tech leads, and VPs into scalable solutions, AI or not, built in TypeScript (React, Node.js), Python, and Go.

Experience

AVRL — Freight-tech automation for US freight brokers — Remote (US HQ)
Client Solutions Engineer (Forward Deployed) — *Sep 2024 – Present*

- Own technical intake for 30+ broker customers after onboarding: **1,000+ calls** (3–4 a day) with ops teams, tech leads, and VPs; I turn requests into specs and approve or reject every requirement before engineering builds it.
- Designed a **three-tier equipment classifier** (keyword mapping, type resets, comment overrides) in logic calls with a broker whose shipper buried equipment needs in remarks, so flatbeds stopped pricing as hotshots.
- Replaced exponential 3D bin packing with a **2D pallet-fit check** with stacking rules, so loads of 40–50 pallets could be checked inside live bidding; agreed the trade-off with the customer and leadership.
- Turned the rule-script errors I kept seeing on broker calls into **Cachet**, a no-code rule builder I built solo; its constrained blocks made the syntax errors that broke rules and cost revenue impossible.
- Prototyped an **LLM assistant** for Cachet (OpenAI API): mined Zendesk tickets for how brokers phrase requests, then mapped ambiguous intent to Cachet's condition, join, and precedence blocks.
- Cut render time for 1,000+-condition configs from **9 s to 350 ms** and bundle size by about 55% with delegated listeners, summary-first rendering, virtualization, and code-splitting.
- Designed **Generations UI**, AVRL's design system (Figma to Storybook to shadcn-style package), replacing 25+ duplicated components; mentored 4 engineers and reviewed PRs.

Forward Deployed Engineer — *Sep 2021 – Sep 2024*

- Automated spot-market bidding across **15+ TMS platforms** (Uber Freight, MercuryGate, Emerge, E2open), pricing loads from DAT and Greenscreens rates and margins, eliminating **5–10 min** of manual work per load.
- Kept bid automation running through frequent TMS UI changes with layered selector fallbacks (class, ID, CSS, XPath) across thousands of concurrent loads.
- Built the **decision-tree execution service** broker systems call **1M+ times a day** (peaks near 4M) with Python, FastAPI, Celery, and Redis on Cloud Run; cycle detection blocks looping trees and caching serves 85% of calls.
- Built Chrome extensions, used by all 30+ broker customers, that bring the decision-tree API into the TMS screens their teams already use, saving **100+ hours** of manual bid processing a week.
- Open-sourced schemafied (Python schema validator, on PyPI) and depman (Go dependency checker) from internal services.

Skills

Solutions	Technical discovery, requirements scoping, solution design, stakeholder management, customer training
Languages	TypeScript, JavaScript, Python, SQL; familiar with Go
Frontend	React, Next.js, Redux Toolkit, Tailwind CSS, Storybook, design systems, web performance
Backend	Node.js, Express, FastAPI, Celery, BullMQ, REST APIs, WebSockets, MedusaJS
Data & Cloud	PostgreSQL, Redis, GCP (Cloud Run, Cloud Storage), Docker, GitHub Actions, Linux
Testing	Jest, Vitest; unit, integration, and end-to-end suites
AI	LLM integration (OpenAI API), Claude Code, Cursor, Antigravity, spec-driven development, Graphify

Projects

H&H — Direct-to-consumer fashion store — *Co-founder and sole engineer · Pre-launch*
TypeScript, Next.js 16, MedusaJS v2, PostgreSQL, Redis, Docker

- Wrote a Razorpay payment provider that creates exactly one order per payment (HMAC-verified webhooks, unique-index event claims, server-recomputed prices): 50 concurrent replays of one webhook produced one order.
- Prevented overselling with per-item Redis stock locks (0 oversells in 25 rounds of 50 parallel buyers); ISR plus a shared in-flight cache served 211K page requests with 20 backend hits in a local load test.
- Modelled product types as data (no deploy to add one) with Latin, Arabic, and Devanagari validation; 2,100+ unit and integration tests and OpenAPI 3.1 generated from Zod contracts.

Education

University of Lucknow, Faculty of Engineering and Technology — Lucknow, India
B.Tech, Electronics and Communication Engineering — *Aug 2017 – Aug 2021*