Lei Sun — Platform Engineer / Developer Productivity Engineer

Full-Stack Foundation | PayPal FinTech | Developer Platform | Environment Automation / Multi-Stack / AI Toolchains

Email: erishen@qq.com | Phone: +86 186-2179-6513 | Shanghai, China
Personal site: https://erishen.cn | GitHub: https://github.com/erishen

Summary

Focused on platform engineering and developer tooling, backed by a solid full-stack foundation. At PayPal, built from scratch a standardized Docker test environment system covering 6 language stacks and 13 e-commerce platforms; a three-layer image caching architecture reduced cold-start times from minutes to under 10 seconds; pushed 30+ multi-arch (arm64/amd64) images to JFrog; packaged a self-service startup tool and Makefile unified interface so AI Agents and engineers can spin up environments without knowing any underlying details. In parallel, built an AI-assisted code analysis toolchain with emphasis on root-cause analysis and pattern documentation, growing a shared team knowledge base rather than individual expertise. Tech stack covers Docker, Makefile, Shell, JFrog, Python, FastAPI, React, Next.js, Node.js, TypeScript, Redis, and gRPC.

Core Competencies

- **Platform thinking** — Abstract high-frequency, repetitive, error-prone engineering processes into reusable infrastructure; hide complexity so developers and Agents focus on business logic
- **Multi-stack environment management** — Independently built a standardized test environment system covering PHP / Java / Ruby / Python / .NET / Node.js across 13 e-commerce platforms
- **Multi-arch compatibility depth** — Systematically resolved arm64 (local Mac) vs. amd64 (CI / cloud) native extension incompatibilities; three-layer image caching architecture is reusable across projects
- **AI toolchain engineering** — Built AI-assisted code analysis tools integrating sub-second codebase sampling with multi-model analysis into developer-facing workflows
- **Documentation discipline** — Every environment issue gets a root-cause analysis and reusable pattern document; outcomes accumulate as team knowledge, not tribal expertise

Work Experience

PayPal

Full-Stack Engineer | Platform & Tooling | 2024.07 — Present

In 2026, primarily built AI payment migration verification infrastructure from scratch, providing standardized multi-stack e-commerce test environments, automated verification pipelines, and AI-assisted toolchains for AI Agents and engineers.

Responsibilities

- Designed and implemented a standardized Docker test environment system covering 6 language stacks (PHP, Java, Ruby, Python, .NET, Node.js) across 13 e-commerce platforms
- Built a three-layer image caching architecture (dependency cache → platform compilation → application image) to solve cross-platform native extension multi-arch compatibility
- Packaged a self-service startup tool handling environment variable injection, architecture detection, volume pre-population, and database initialization internally — exposing only a few env vars externally
- Unified platform operation commands (start / stop / reset / test / build) with a Makefile interface, establishing standard engineering conventions
- Built an AI-assisted code analysis tool integrating MCP with multiple models (Claude / GPT-4 / DeepSeek / Gemini); supports rapid sampling analysis of large codebases and automatic Docker configuration generation
- Maintained docker-compose-patterns.md and issues-found.md knowledge bases, cataloguing environment engineering lessons ranked by learning value

Impact

- Delivered a 13-platform e-commerce payment test system; pushed 30+ JFrog multi-arch (arm64/amd64) images supporting fully offline cold starts
- New platform onboarding time reduced from a half-day of debugging to 1–2 hours
- Ruby gem cold start reduced from 2 minutes to 10 seconds; Java Maven build reduced from 5 minutes to 10 seconds
- AI Agents can independently complete environment startup and payment flow verification without human intervention on environment issues
- Built 60+ Playwright E2E test cases covering the full payment chain: checkout, authorize, capture, cancel, and refund
- Identified and fixed complex platform issues including OrbStack BuildKit isolated network namespace behavior and VirtioFS nested bind mount silent failures; produced reusable solution documents
- Contributed to Next.js 12→14 and Node.js 18→20 stack upgrades; delivered Customer Identity Platform (CIP) end-to-end

Tech Stack: Docker, Makefile, Python, Shell, JFrog, Playwright, Node.js, Ruby, Java, PHP, .NET

Quantex Information Technology (Shanghai)

Full-Stack Engineer | 2024.01 — 2024.06

Led frontend integration for a Disney resort AI digital-human customer service system; initial integration completed in 1 month.

Responsibilities

- Implemented Web-based 3D avatar rendering, motion and expression control using React, Three.js, and react-three-fiber
- Integrated Azure ASR / TTS voice sync; resolved iOS / Android WebGL and audio playback compatibility issues

Impact

- Delivered the first working integration of the AI digital-human system within 1 month, enabling the project to proceed to the demo stage

Tech Stack: React, Next.js, Three.js, react-three-fiber, Azure AI, Socket.io

---

Trip.com Group

Senior Frontend Engineer | 2017.04 — 2024.01

Owned Trip.com international Blog / Destinations content channel development and maintenance; drove team React / Node.js tech migration.

Responsibilities

- Built the department's first Node.js SSR content platform, driving the team's migration from .NET to React / Node.js
- Long-term ownership of Trip.com international multilingual content channel; continuously optimized Core Web Vitals and SEO

Impact

- Drove team tech stack migration; established SSR / SEO / Hybrid multi-channel delivery
- Led 2—4 person frontend sub-teams; established code review standards, engineering norms, and knowledge-sharing culture

Tech Stack: React, Next.js, Node.js, Express, Redis, MySQL, React Native

---

Jiuzhen Network Technology (Shanghai)

Frontend Engineer → Frontend Engineering Manager | 2015.03 — 2017.04

Responsibilities

- Led React Native cross-platform social app development targeting Android and iOS
- Managed a 4—7 person frontend team; set up Jenkins multi-branch automated build pipelines

Tech Stack: React, React Native, Node.js, PHP, MySQL, Jenkins

---

Actiontec (Shanghai)

Software Engineer | 2009.10 — 2015.03

Responsibilities

- Led Hybrid mobile app development for a photo-management product
- Built router embedded Boa WebServer and management interface; collaborated with US-based cross-timezone teams

Tech Stack: Cordova, HTML5, JavaScript, Java, Linux / C / Shell

---

ZTE Corporation

Software Engineer | 2008.10 — 2009.10

Responsibilities

- Built alarm module for a 3G network management platform; implemented alarm task processing and desktop client display

Tech Stack: Java, Swing, EJB, JBoss, Oracle

Gksel Information Technology (Shanghai)

Software Engineer → Technical Manager | 2007.04 — 2008.10

Responsibilities

- Developed Web projects (rental platform, doctor appointment booking); managed a 3-person technical team

Tech Stack: Java, Struts, Spring, PHP, JavaScript, jQuery

Jiangsu Jinswei Information Technology

Software Engineer | 2005.04 — 2007.04

Responsibilities

- Developed and delivered custom enterprise CRM system

Tech Stack: Java, JavaScript, Struts, Tomcat, MS SQL Server

Projects

PayPal — Multi-Stack E-Commerce Test Environment Platform

2026 | Docker, JFrog, Shell, Makefile, Playwright, Multi-Language Stacks

Designed and built unified test environment infrastructure for the AI payment integration verification team. Goal: any AI Agent or engineer picks up the project and has a running environment within 5 minutes, with consistent behavior across arm64 and amd64.

- Covers PHP×6 + Python×1 + Java×2 + .NET×2 + Node.js×1 + Ruby×1 across 13 e-commerce platforms with 20+ independent Docker environments
- Three-layer image caching: taking Ruby as an example — gem-cache (platform-agnostic gem source) → bundle-cache (per-arch native extension build artifacts) → application image; resolves arm64/amd64 native extension incompatibility
- Offline cold start: npm prebuild-cache (@next/swc, better-sqlite3, etc.) and Maven dependencies pre-bundled into JFrog; containers select by architecture automatically without internet access
- Self-service interface: only a few environment variables exposed externally; all initialization details (image pull, volume fill, credential injection, database init) handled internally by the tool
- Makefile unified interface: standardized start / stop / reset / test / build commands across all platforms; AI Agents can invoke directly

---

AI-Assisted Code Analysis Tool

2026 | Python, FastAPI, MCP, Claude / GPT-4 / DeepSeek / Gemini

Rapid analysis tool for large codebases: samples and analyzes 4,000+ file projects in under 1 second; multi-model collaboration generates code structure insights and Docker configurations.

- Integrated MCP toolchain for codebase structure sampling, tech stack identification, dependency analysis, and config file generation
- Supports Claude / GPT-4 / DeepSeek / Gemini multi-model routing by scenario and cost
- Auto-generates Dockerfile and docker-compose.yml drafts; developers adjust env vars and start

Related personal project: ai-analyze — MCP + Serena-based codebase sampling and analysis tool

---

PayPal — Large-Scale Merchant Site Scanning

2026 | Python, Playwright, DNS, HTTP, Data Analysis

Layered scanning of hundreds of thousands of merchant URLs to identify legacy payment integration sites, supporting business team migration decisions.

- Three-layer architecture: DNS pre-check + HTTP static scan + Playwright deep exploration — progressively filters invalid targets
- Identified thousands of legacy WPS payment integration sites; analyzed integration pattern distribution (auto-submit / encrypted / hosted / cart)
- Integrated three-source threat intelligence (URLhaus / Phishing.Army / HaGeZi); auto-filtered malicious sites
- Produced structured multi-category site datasets directly supporting business team migration prioritization

---

PayPal — Customer Identity Platform (CIP)

2025 | Next.js, Node.js, Express, Redis, gRPC, K8s

End-to-end ownership: design, development, testing, security audit, and production release.

- Built project structure, page routing, BFF API, Redis caching, and K8s production deployment configuration
- Implemented privacy policy management and user consent tracking meeting PIPL compliance requirements

---

Quantex Information Technology — Disney AI Digital-Human Customer Service

2024 | React, Next.js, Three.js, react-three-fiber, Azure AI

- Integrated 3D avatar model: Web rendering, Azure ASR / TTS voice sync, and lip-sync; initial integration completed in 1 month

---

Trip.com — International Content Platform

2020–2024 | React, Next.js, Node.js, Express, SSR, Redis, MySQL

Project URL: https://www.trip.com/blog

- Built the department's first Node.js SSR content platform; drove team migration from .NET stack
- Long-term ownership of Trip.com Blog / Destinations multilingual content channels; optimized Core Web Vitals and SEO

Open Source

Ongoing personal experiments in developer tooling, CLI engineering, and platform standardization.

Developer Tools / CLI: lobster, nsgm — OpenAI Function Calling-compatible tool registry + REST API (standardized tool invocation for AI Agents); Next.js + GraphQL + MySQL full-stack code generation scaffold (94+ commits, actively maintained)

Component Registry / Docs: shadcn-registry — self-hosted shadcn/ui component registry + Storybook 9 documentation site; demonstrates platform engineering thinking for internal component distribution and standardization

---

Education

Nanjing University of Information Science & Technology | B.Sc. in Information and Computing Science | 2001–2005

Target Role

Desired Position: Platform Engineer / Developer Productivity Engineer / DevOps / Tooling Engineer
Location: Shanghai preferred; open to remote
Availability: Immediately
Target Industries: Developer platforms, internal tooling, AI toolchains, FinTech infrastructure, DevX engineering

---

> My edge isn't running someone else's CI/CD pipeline — it's identifying engineering pain points from scratch, designing reusable environment abstractions and toolchains, and enabling both teams and AI Agents to work independently and reliably.