Caleb Rogers

Software Engineer

Rocky Top, Tennessee (Remote)

Portfolio: https://getrelayworks.com | GitHub: https://github.com/DevCalebR | LinkedIn: https://www.linkedin.com/in/caleb-r-rogers

Professional Summary

Software engineer building AI applications, workflow automation, API integrations, document-processing systems, native macOS tools, and full-stack web applications. Works across Python, FastAPI, Next.js, React, TypeScript, Swift, and SwiftUI, with experience implementing REST APIs, background processing, authentication, webhooks, and reviewable operational workflows.

Technical Skills

Languages: Python, Swift, TypeScript, JavaScript, HTML, CSS

Frameworks: FastAPI, Next.js, React, SwiftUI

Backend and integrations: REST APIs, background processing, authentication, webhooks, Prisma, PostgreSQL, Twilio, Stripe

AI and workflow: AI applications, document processing, structured outputs, OpenAI API integration, Google Docs delivery architecture

Tools and platforms: Git, GitHub, Xcode, Vercel, Cloudflare

Professional Experience

RelayWorks | Founder and Software Engineer | 2026-Present

- Build AI applications, developer tools, workflow automation, and full-stack software, including a commercial source-code product and public engineering case studies.
- Designed and shipped the RelayWorks AI Document Processing Kit, a PDF-first FastAPI and Next.js source-code product with local background processing and reviewable output packaging.
- Publish authentic screenshots, sample outputs, architecture documentation, setup requirements, and explicit limitations so technical work can be evaluated without unsupported claims.

Selected Engineering Work

RelayWorks AI Document Processing Kit

Commercial source-code product | Python, FastAPI, Next.js, React, TypeScript

Built a self-hosted, PDF-first workflow that invokes a separately installed Marker CLI through local background processing and produces Markdown, HTML, structured JSON, plain text, extracted-image review, processing reports, and ZIP packages. The public repository is an evaluation showcase; it does not expose the commercial source code or claim hosted SaaS capabilities.

Product: https://getrelayworks.com/document-processing-kit/

CallbackCloser

Repository and local simulator; live deployment unavailable | Next.js, React, TypeScript, Prisma, PostgreSQL, Twilio, Clerk, Stripe

Implemented missed-call detection, persisted SMS qualification, owner notifications, protected lead workflows, provider webhooks, tenant boundaries, and subscription gating. The repository supports local evaluation and an isolated simulator; it makes no customer, conversion, or production-scale claims.

Repository: https://github.com/DevCalebR/callbackcloser

Selected Engineering Work (continued)

Webhook Dashboard & Alerts

Repository and local demonstration | Next.js, React, TypeScript, Prisma, PostgreSQL, Clerk

Built an operations dashboard with raw-body HMAC verification, idempotent event persistence, searchable payload details, alert rules, recorded outcomes, and controlled replay. Current alert evaluation is synchronous and request limiting is process-local; the project does not claim production scale or guaranteed delivery.

Repository: https://github.com/DevCalebR/webhook-dashboard-alerts

AI Content Automation with Google Docs Delivery

Code and architecture case study; live deployment unavailable | Next.js, React, TypeScript, Prisma, PostgreSQL, OpenAI API, Google APIs

Developed a codebase demonstrating structured content briefs, persisted AI-assisted runs, human review, deterministic Markdown, text, DOCX, and PDF exports, and Google Docs delivery architecture. Provider-backed generation and delivery require separate configuration and manual verification and are not presented as verified live outcomes.

Case study: https://github.com/DevCalebR/ai-content-automation-app-google-docs-delivery

Portfolio Dashboard

Frontend demonstration with synthetic data | React, TypeScript, Vite, TanStack Table, React Hook Form, Zod, Recharts

Built a responsive frontend for browsing synthetic backtest runs, configuring validated mock runs, synchronizing filters with the URL, and reviewing lazy-loaded charts. The project has no live trading system, broker integration, analytics backend, authentication, persistent database, or financial performance claims.

Repository: https://github.com/DevCalebR/portfolio-dashboard

Security Workflow IDE for macOS

Native macOS project; public repository unavailable | Swift, SwiftUI, Xcode, Python and shell execution

Built a native multi-pane developer workspace with an integrated editor, interactive terminal, plugin toolbox, workflow dashboard, persistent project sessions, runtime arguments and execution metrics, and Markdown report generation.