Arun Teja Reddy Kallam

Tempe, AZ | (480) 937-6420 | akallam04@gmail.com | linkedin.com/in/akallam3 | github.com/akallam04 | arunkallam.com

EDUCATION

B.S. in Computer Science (GPA: 4.0 • Dean's List) **Expected May 2027**
Arizona State University **Tempe, AZ**
Relevant Coursework: Data Structures & Algorithms, Operating Systems, Software Engineering, Object-Oriented Programming, Database Management, Foundations of Machine Learning, Intro to AI, Probability & Statistics.

TECHNICAL SKILLS

Languages: Python, Go, Java, C/C++, JavaScript, TypeScript, SQL, HTML/CSS
AI/LLM: LLM agents, tool calling, LangGraph, MCP, RAG, ChromaDB, LLM evals, guardrails, Claude API, Claude Code
Backend & Cloud: FastAPI, REST APIs, AWS (Lambda, ECR, EventBridge), Docker, Vercel, PostgreSQL, Redis
Testing & DevOps: pytest, Playwright (E2E), concurrency testing, GitHub Actions CI, Git
Data: Pandas, MySQL, web scraping, ETL pipelines, data validation, LLM enrichment

PROJECTS

Shopify AI Support Agent *(Python, LangGraph, FastAPI, Claude API, MCP, ChromaDB, AWS)* **| Live demo | GitHub**

- Built a customer support AI agent for a Shopify store as a LangGraph state machine that answers policy questions with RAG and cancels orders, changes addresses and files returns via tool calling over a custom MCP server.
- Added a code-enforced confirmation guardrail and ran an LLM eval of 50 tasks (200 simulated chats): vs prompt-only rules, chats with an unsafe write fell from **42/200 to 0** and safe resolution rose from **75.5% to 95%**.
- Deployed a FastAPI/Docker backend on AWS Lambda and a Vercel frontend with 600+ pytest and Playwright tests in CI; an eval-gated switch to Claude Haiku 5.5 cut cost per resolved chat by **95%** with zero unsafe writes.

Oversold: Race-Condition Tester for E-commerce Checkouts *(Go, PostgreSQL, Redis, Docker)* **| GitHub | Walkthrough**

- Built a Jepsen-style Go harness that races concurrent checkouts against local copies of Medusa and checks stock and order rules on every request and database row: **3,120 recorded runs**, every verdict rechecked in CI.
- Reproduced two unfixed oversell bugs on three Medusa releases: a multi-line cart oversold in **30/30** runs on each, and two servers sold the last unit twice in **87/90** with the default in-memory lock.
- Verified two unmerged community fixes, built from each PR's source: each took its own bug to **0/30**; neither fixes both.

Shortfall: LLM Agent Benchmark for E-commerce Race Bugs *(Python, Claude API, Docker)* **| GitHub | Walkthrough**

- Designed a benchmark where a Claude agent reads MedusaJS checkout code and writes race-condition tests that Oversold runs; a bug counts only if its community fix clears the failure.
- With no bug report, Claude Haiku 5.5 reproduced each race scenario in **7/10** trials for **under 2 cents per trial**, but covered all 3 bug scenarios together in only 3/10 vs 10/10 for a no-AI test grid.
- Engineered a cost-capped, fault-tolerant agent loop (prompt caching, retries, crash-safe resume, hard $20 cap); in a pre-registered follow-up, Claude Sonnet 5.5 reached **9/10** on all 3, and CI rebuilds every result from **16,994 runs**.

EXPERIENCE

Web Development Intern **Jun 2026 – Aug 2026**
BYLD Commerce **Lehi, UT**

- Developed a Python build and diffing pipeline that turns one HTML/CSS/JavaScript codebase into **14 Squarespace-ready artifacts** (header/footer injections, custom CSS, 11 section blocks), cutting site updates to minutes.
- Architected a framework-free JavaScript front end with a deterministic SVG state machine and scroll-driven UI states.
- Shipped it to production on Squarespace, debugging **4 platform-level conflicts** (CSS cascade overrides, fixed-position overlays, anchor navigation, header-class collisions), and customized client Shopify storefront themes.

Junior Data Analyst **Jun 2025 – Jul 2025**
Food Forest AI **Philadelphia, PA**

- Built web-scraping and ETL pipelines that collect and normalize company data for an **AI-powered B2B search engine**.
- Applied LLM-based extraction and classification to enrich and tag incomplete company profiles, then deduplicated the records and loaded them into **MySQL** for search ingestion.
- Automated Pandas data validation checks across **500+ company profiles** to catch missing fields, anomalies and formatting errors before records reached the AI search engine, cutting manual review.