

Eric Gilerson

Westfield, NJ | 908-858-7609 | ericgilerson@gmail.com | github.com/EricGilerson | www.linkedin.com/in/ericgilerson/

Education

University of Maryland

B.S. in Computer Science, Minors in Computational Finance & Mathematics, GPA: 4.0

College Park, MD

Expected Spring 2028

Languages & Skills

Languages: Python, C++, Go, Java, Swift, TypeScript, JavaScript, SQL, R, MATLAB

Frameworks & Tools: Next.js, Node.js, PyTorch, NumPy, FastAPI, Docker, AWS, GCP, Celery, CI/CD

Experience

Software Engineering Intern

New York, NY

Arbis AI

May 2026 – Present

- Shipped a Python SDK that finds running agents (over 99% accuracy) or registers them by hand, ingesting traces in FastAPI for runs, compliance evaluations, workflow graphs, version control, and identity management for every agent and workflow.
- Cut the dashboard from 6.7s to under 100ms and each payload from 82 MB to 160 KB with Postgres indexes and JSON dedup.
- Replaced a 5-second notification poll with event pushes, cutting p99 latency from over 15s to about 20ms.
- Caught over 60% more risk and drift by filling incomplete agent cards with AI, and added failure explanations on all 42 rules.

Founding Software Engineer & Technical Lead

San Francisco, CA

Ajama AI (Palantir Fellowship Cohort 002)

January 2026 – Present

- Built Structora, a Tauri, React, and FastAPI desktop app on AWS (EC2, S3, MongoDB) for site files, deliverables, and team messages in one workspace instead of email, shared drives, and a separate chat.
- Cut first-draft time for briefs, reports, and RFIs by about 75% with parallel research workers over firm knowledge, the project, and the web, orchestrated on Celery and Redis, plus LanceDB hybrid search over prior drafts, firm files, and research.
- Built a plain-language question layer over 3D building models (IFC/BIM), raising useful answers from 50% to over 90%.
- Cached project context in Redis so tool-call loops reuse it, cutting assembly from 78ms to 9ms (9×) and off the database CPU.

Contract Software Engineer

College Park, MD

Voodoo

September 2025 – December 2025

- Built the TikTok-style feed and live voice rooms, where people join to talk about a shared topic, for the Vex iOS app in Swift, mixing and watermarking 10+ hours of video a day with AVFoundation and FFmpeg, with no manual export after each take.
- Built atomic Redis matching into those topic rooms, falling back to a voice AI agent when the timeout window is empty.

Projects

NeurIPS 2026 Conference Paper

April 2026 – September 2026

- Sole author of *Don't Learn What You Can Compute*, accepted to NeurIPS 2026: frozen residual blocks that compute exact integer arithmetic inside a transformer's forward pass, while the model learns only a small interface on top of a frozen base model.
- Hit 99.95% exact answers on 36,000 problems with a frozen SmolLM2-360M, up from 4.56%, and held 99.89% after shrinking the added parameters to about 306K on the same 36,000 problems, with the base model left frozen.
- Joint training from scratch raised math accuracy from 19% to 97.7% at the same text quality, and across five seeds a paired fact-recall task learned faster, peaking 7.5 points ahead at 10k steps, while the math accuracy stayed intact.

Draft (GitHub)

June 2026 – Present

- Built a Go and React desktop app replacing Docker Compose with one-click deploys from 25+ templates, stable hostnames, and variable references so services point at each other by name instead of a pasted host and port.
- Pins each service to a git branch with redeploy on commit, plus pull-request sandboxes on a 4-step TTL lifecycle and rollback.
- Shipped an MCP server so a coding agent can deploy a service, read its logs, and open a shell without leaving the editor.
- Added multiple environments as a local testing layer, each with its own hostnames, volumes, and a branch pin per service.

Rindle (PyPI | GitHub)

July 2025 – November 2025

- Published a C++20 library on PyPI that turns per-ticker price CSVs into sliding-window training tensors, fitting scalars in parallel and saving a manifest so later runs reload without rebuilding the dataset from the CSVs.
- Exposed the windows as NumPy arrays through Python bindings that release the GIL during the build.