Jonathan Frias

Fremont, CA · (707) 779-2019 · jonathanfrias.1797@gmail.com · linkedin.com/in/jonathan-frias-59452a162 · github.com/jfrias-CS · jfrias-cs.github.io

EDUCATION

California State University, East Bay — B.S. Computer Science Graduated May 2026

Cumulative GPA: 3.4 · CSU East Bay GPA: 3.98 / 4.0 (transferred in at 3.229; raised across 4 semesters)

Coursework: Software Engineering, Artificial Intelligence, Web Development, Data Structures, Algorithms, Statistics, Linear Algebra, Computer Architecture, Programming Concepts, Software Responsibility, Computer Networks.

SKILLS

Languages: TypeScript, JavaScript, Python, Java, Kotlin, C / C++, Rust, Assembly

Frontend: React, Next.js, Tailwind, Vite, HTML / CSS

Backend / Web: FastAPI, Actix-web (Rust), REST APIs, socket programming

Mobile: Android (Kotlin + Jetpack Compose)

AI / ML: OpenAI Fine-tuning, TensorFlow Lite, Claude API + prompt caching, NLG evaluation (BLEURT / COMET / BERTScore / SBert / USE / METEOR), computer vision pipelines

Infra / Tools: Linux, Docker, Tailscale, Git, SciCat, LabView APIs

EXPERIENCE

Automation & Software Development Engineer Intern — Lawrence Berkeley National Laboratory Jun 2025 – Aug 2025 · Berkeley, CA

Advanced Light Source · ASPIRES program

- Built a React + TypeScript interface (forked from `als-computing/sample_metadata_app-frontend`) enabling ALS beamline scientists to configure X-ray spectroscopy samples; **authored multi-row selection (shift+click, click-drag, shift+arrow), bulk duplicate/delete, and an Undo system that reverts an entire selection in one action**.
- Wrote **~1,000 lines of Python automation** (`ScanFileGenerator`) that authenticates to SciCat via `pyscicat`, pulls bars and samples for a proposal, and generates LabView-format scan files for **XAS / XLD / XMCD across 4 beamlines (402, 631, 4.0.2, 6.3.1)** with element-specific edge energies for **11 elements**, organizing output by proposal → bar → sample directories.
- Replaced **hours of per-file manual editing** (a typical proposal requires ~25–300 hand-written scan files — 5–20 samples × 5–15 scan types — at ~1–2 min each) with a **sub-minute batch run**, eliminating per-file format errors during high-stakes beamline operations.
- Selected for the **ASPIRES** (Advancing STEM Pioneers in Research in Energy Sciences) program; delivered a 1-minute lightning pitch and open poster session.
- **Tech:** React 18, TypeScript, Vite, Python, pyscicat, LabView API, Docker, Bulma · Public repos: ScanFileGenerator, sample_metadata_app-frontend

Undergraduate Research Assistant — CAHSI / Walkaware (PI: Prof. Lynne Grewe) Jan 2025 – Oct 2025 · CSU East Bay

- Built **SafeStep**, an Android accessibility app (~970 LOC Kotlin) running a 3-stage ML pipeline per captured frame: **on-device TFLite object detection** locates sidewalk hazards and produces bounding boxes, **on-device TFLite severity regression** classifies each into Low / Moderate / High / Critical, and a **fine-tuned LLM** generates a natural-language guidance narrative spoken aloud via Android TextToSpeech for visually impaired users.
- **Trained both TFLite models from scratch** — hazard detector (224×224, 6-float-per-box output) and severity regressor — deployed fully on-device for offline operation and sub-second latency.
- Curated a **~2,800-row training dataset** (~933 photos × 3 narrative variants as data augmentation) and **iterated through fine-tunes of GPT-3.5-turbo-0125 and GPT-4o Mini** for the narrative head (epoch = 2, temperature = 0.3).
- Built a **six-metric NLG evaluation harness** (BLEURT, COMET, BERTScore, Sentence-BERT, USE similarity, METEOR); achieved **median BLEURT 0.65 / BERTScore F1 0.60 / COMET 0.75 / SBERT cosine 0.80** (strong semantic alignment), and **diagnosed three main error classes — severity mismatches (~25%), directional conflicts (~20%), over-optimism (~15%)** — driving concrete next-iteration priorities (bounding-box-aware prompting, severity-keyword mapping, augmentation of rare cases).
- Selected for **ASPIRES Summer 2025** (1-minute lightning pitch + open poster session).
- **Tech:** Kotlin, Jetpack Compose, TensorFlow Lite 2.13, OpenAI Fine-tuning API, Python / Colab, Android TextToSpeech · Repo: SafeStep2

Field Service Engineer — Murray Window & Door Aug 2024 – May 2025 · Campbell, CA

- Developed a preventive maintenance schedule that contributed to a **15% reduction in product failures**.
- Documented equipment performance metrics in logs and databases to support analysis and reporting.
- Participated in equipment installation, trial runs, investigative tests, repairs, and overhauls.
- Collaborated cross-functionally to resolve technical challenges in service processes.

PROJECTS

FIRST DAY — AI Codebase Onboarding Tutor In progress · ship Jun 2026

- AI agent that ingests any public GitHub repo at a pinned commit and turns it into an interactive tutor: **chat-over-code with citations**, **auto-generated Mermaid architecture diagrams**, and **AI-graded comprehension assessments** that gate progress.
- **Stack:** Next.js 14, TypeScript, Tailwind, shadcn/ui, FastAPI (Python), Claude API (Sonnet 4.6 + Opus 4.7) with prompt caching, tree-sitter, SQLite + sqlite-vec, Mermaid, Vercel + Fly.io.

MotoCrashApp — Hackathon Emergency Response System MESA-U 2024 · Best Research Award (1 of ~100)

- Built a Python (Tkinter) desktop app that monitors for a configurable post-incident countdown; if the rider doesn't dismiss it, the system automatically calls the **Google Geolocation API** for GPS coordinates and **dispatches a WhatsApp message** (via PyWhatKit) to the rider's emergency contact containing their name, allergies, and live latitude/longitude.
- **Stack:** Python, Tkinter, threading (non-blocking countdown), requests, Google Geolocation API, PyWhatKit · ~180 LOC main module + 14 Python files, built in ~48 hours.

Self-Hosted Home Security System — Home-lab / infrastructure project Personal

- Deployed a self-hosted security camera + NVR system on a Linux home server using **Frigate** (open-source NVR with real-time object detection); integrated a network camera and exposed the system remotely via **Tailscale** mesh VPN for off-network access.
- **Stack:** Linux, Docker (Frigate container), Frigate NVR, Tailscale VPN, MQTT. Demonstrates Linux sysadmin, containerization, networking / VPN, integration of open-source tooling.