

Aakarsh Kashyap

souls.syntax@gmail.com | [linkedin.com/in/souls-syntax](https://www.linkedin.com/in/souls-syntax) | github.com/souls-syntax

EDUCATION

GLA University

B.Tech in Computer Science Engineering, CGPA: 8.51

Mathura, UP

Aug. 2024 – May 2028

Indian Institute of Technology (IIT)

B.S in Data Science and Applications, CGPA: 7.45

Madras, Tamil Nadu

May 2024 – Aug 2028

EXPERIENCE

Technical Contributor, GigaVector (Open Source)

Apr 2026 – Present

GigaVector

Virtual

- Diagnosed and fixed a non-portable compile flag causing SIGILL crashes on WSL2 targets in a vector database with 500+ active users; traced the failure to ABI assumptions that did not hold across toolchain configurations and submitted a patch accepted upstream.
- Resolved a daemon thread lifecycle bug paired with an `is_running` liveness race that allowed silent data corruption on shutdown; the fix required reasoning about memory visibility across thread boundaries and correct teardown ordering under concurrent access.
- Debugged cross-platform DLL dependency failures on Windows involving libcurl linkage, MinGW runtime mismatches, and CFFI symbol resolution errors, restoring build reproducibility across Linux and Windows targets for all downstream contributors.

PROJECTS

sauceOS | C, x86-64 ASM, Make, Limine

Jan 2026 – Present

- Built a 64-bit OS from scratch booting via the Limine bootloader; implemented the Global Descriptor Table, Interrupt Descriptor Table, ISR and IRQ dispatch routines, and Task State Segment entirely in C and x86-64 assembly.
- Developed a buddy memory allocator for kernel heap management in the higher-half virtual address space, correctly handling block splitting, coalescing, and alignment constraints for kernel allocations.
- Implemented 4-level paging with independent per-process virtual address space mapping.
- Implemented a PS/2 keyboard driver with interrupt-driven input handling wired through the IDT; wrote a minimal libc and syscall interface establishing the kernel-user boundary for future userspace programs.

soft-cuda | C++, CUDA, cuBLAS, cuDNN

Mar 2026 – Present

- Built a hybrid CPU/GPU tensor execution engine from scratch featuring a bump allocator memory pool, a lazy evaluation DAG, and a three-pass AOT compiler with profile-guided backend assignment between CPU and GPU operators.
- Implemented a double-buffered SGEMM kernel overlapping memory transfers with compute using CUDA streams; benchmarked hybrid scheduling at 0.94 ms/step against GPU-only dispatch at 1.44 ms/step, a 35% throughput improvement, with cuBLAS as the reference backend.
- Implemented full forward and backward pass with MSE, cross-entropy loss, and SGD; trained XOR to convergence at loss $\approx 3.5 \times 10^{-14}$ and a GPU performance prediction model, validating the autograd pipeline end-to-end on the custom library.

sush | C++, POSIX API

Feb 2026 – Mar 2026

- Implemented a Unix shell from scratch in C++ supporting piping chains, command history, and variable expansion without relying on any system shell utility; used `fork`, `exec`, `pipe`, and `dup2` directly for process creation, file descriptor wiring, and IPC.
- Built a custom lexer and token expander for command parsing, tokenization, and variable resolution, handling edge cases in quoting and substitution that standard shells abstract away.

TECHNICAL SKILLS

Languages: C/C++, Go, Python, x86-64 ASM, CUDA, SQL

Systems: OS internals, memory allocators, POSIX APIs, IPC, syscall interface

GPU / HPC: CUDA kernel programming, CUDA streams, cuBLAS, cuDNN, SGEMM, double buffering, memory hierarchy optimization

Tooling: Git, Docker, CMake, Make, GDB, NeoVim, Arch Linux