Souvik Mukherjee

Analyst Software Engineer

+91 7330787625 souvikmukherjee150@gmail.com linkedin.com/in/souvik150 github.com/souvik150

About Me

Analyst at Zanskar Research, a proprietary trading firm, where I lead the development of **Sentinel OMS** the firm's order management system. My day-to-day work spans the full order lifecycle: designing state-machine-driven order flows, parent-child order workflows, and execution algorithms, alongside real-time risk (RMS) validation and mark-to-market accounting for low-latency trading across Indian markets. I am SEBI NISM-certified in Equity, Common, and Commodity Derivatives (Series VIII, XIII, and XVI), which grounds the domain modeling behind this infrastructure.

Experience

Analyst Software Engineer — **May 2025 – Present**

Zanskar Research, Bangalore, India

- Led end-to-end design and development of **Sentinel OMS v2** in **Go** for a live securities brokerage — defined architecture, owned technical decisions, and mentored 2 interns on order-lifecycle internals and trading concepts.
- Architected a **two-layer OMS** separating user-facing parent orders from exchange-facing child orders; owned the full lifecycle from **gRPC** ingestion through **COLO** bidirectional streaming, async **NATS JetStream** persistence, and real-time in-memory portfolio updates.
- Built the OMS-side **algo engine bracket** orders, **OCO** multi-exit monitoring, **trailing stops** with live LTP ratcheting, LTP-gated conditional entries, lot-size-aware **iceberg** (LCM) slicing across single and multi-leg baskets, **AMO/GTD** scheduling, and freeze-chunk splitting for exchange quantity limits.
- Designed a dual **request-ID response-ID replay protocol** between OMS and COLO: on restart each side independently replays from its last-seen ID, guaranteeing **exactly-once delivery** and zero missed responses across pod restarts **without a distributed lock**.
- Built a pre-trade compliance chain of **15+ pluggable checks** (margin, MTM, banned/RBI-restricted, gross qty/turnover, EDIS, rate limiting) over deep-cloned portfolio snapshots; instrumented **μs-precision Prometheus** latency tracking and **crash recovery** that replays active orders, rewires SL/TP links, and rebuilds the dual-copy portfolio.

Backend Intern — **Dec 2024 – May 2025**

Zanskar Research, Bangalore, India

- Key contributor in the **OMS team**; developed **limit, market, IOC, iceberg** order types with full lifecycle handling (**new, modify, cancel**) and **exchange response** integration.
- Lead developer of the **RMS (Risk Management System)** module — enforced **bans, restrictions**, and **MTM checks** for robust risk controls.
- Automated **SPAN, VAR, and Refdata** uploads to **AWS S3** using **Apache Airflow** with **calendar-aware scheduling** based on **NSE/BSE holidays**.

Education

Vellore Institute of Technology, Vellore — **Sept 2021 – May 2025**

B. Tech Computer Science and Engineering, CGPA: 8.9/10 — *Tamil Nadu, India*

Projects

Order Matching Engine | *C++20, Custom RB-Tree, Multicast UDP, epoll, Low Latency Programming Link* — **Nov 2025**

- Developed a fully functional **Order Matching Engine** simulating how exchanges (e.g., NSE/BSE) match orders and stream market data to HFTs in microseconds.
- Implemented a **custom Red-Black Tree** for price-sorted bid/ask books (descending for bids, ascending for asks) with strict price–time priority and in-memory trade matching.
- Initially used a **Strategy Pattern** for execution types (*Limit, Market, IOC, FOK, Stop, Iceberg*), but later **removed virtual dispatch and replaced it with branchless, inlined fast-paths** to avoid branch misprediction penalties and vtable stalls critical in sub-100ns hot loops.
- Separated the **order generator** into a dedicated process producing high-frequency order flow and streaming it over **multicast UDP**, mimicking exchange market data dissemination.
- Implemented an **epoll-based event loop** for non-blocking ingestion, batched message reads, and cache-friendly dispatching across instrument-specific matching engines.
- Added real-time **order book visualization and file-based logging** to trace order flow and generate exchange-style Tick By Tick (TBT) feeds.

- Planned **low-latency optimizations** cache-aligned layouts, NUMA-aware core pinning, branchless match loops, custom memory pools, and lock-free SPSC queues.
- **Optimized the matching pipeline from ∼1000 ns to ∼100 ns per order** by eliminating heap allocations, removing pointer indirection (index-based RB-Tree), adding inline hot paths, and restructuring cache-line–aligned price levels.

CoreBlur - Gaussian Blur Engine | *C++, IPC, SHM, Core Pinning Link* **Sept 2025**

- Implemented a high-performance **Gaussian Blur engine** in C++ with support for parallel processing using **IPC and shared memory (SHM)**, enabling multi-process image processing with zero-copy memory access.
- Designed a **tile-based memory layout** for images to maximize cache efficiency, reduce memory bandwidth contention, and allow scalable multi-process execution.
- Implemented **core pinning** to performance (P) and efficiency (E) cores for predictable latency and optimized CPU utilization on hybrid architectures.
- Built a modular architecture separating **image I/O, tile management, IPC management, and Gaussian kernel processing**, enabling clean extensibility for future enhancements like **SIMD acceleration** and advanced scheduling strategies.

MiniGit - Lightweight Version Control System | *C++, STL, Design Patterns, File I/O, Concurrency Link* **Oct 2025**

- Designed and implemented a **Git-like version control system** in modern C++ supporting essential commands such as `init`, `commit`, `branch`, and `merge`, with modular abstractions for repositories and commits.
- Architected the core using **Command, Factory, Strategy, and Singleton** design patterns to achieve clean separation of concerns and extensibility across subsystems.
- Implemented efficient **diff and history tracking** using hash-based content snapshots and custom file serialization for commits and branches.
- Integrated a custom **CLI parser** and multi-threaded commit indexing for fast repository traversal, mimicking Git's object model with lightweight in-memory caching.

NetProbe – High-Performance Socket Framework | *C++, epoll, TCP/UDP, Multicast Link* **Nov 2025**

- Implemented a **low-latency TCP/UDP socket abstraction layer** in C++ with non-blocking I/O, Nagle's algorithm disabling, multicast join/leave management, and optional timestamping — enabling efficient real-time data publishing and subscription for market streams.
- Built an **epoll-based event loop** with connection pooling, asynchronous message dispatch, and adaptive batching to achieve microsecond-level throughput.
- Developed modular utilities for **interface discovery, multicast membership control, and buffer reuse**, supporting both unicast and broadcast transport modes.
- Benchmarked and tuned network performance using **SO_REUSEPORT**, **TCP_NODELAY**, and **recvmsg()/sendmsg()** system-call optimizations across CPU cores.

Leadership / Achievements

- **Won 1st prize (1L) at company-wide Codeathon, awarded by the CEO** — Built an AI trade-journaling system pairing FIFO-matched execution legs with OHLCV context, layering specialized CrewAI multi-agent reasoning (Pydantic-validated, MongoDB-cached) for explainable per-trade and session-level insights. Hybrid design: deterministic quant analytics + agentic qualitative reasoning.
- **Passed NISM Series-XVI: Commodity Derivatives Certification Examination**
- **Passed NISM Series-XIII: Common Derivatives Certification Examination** — Equity, Currency, and Interest-Rate Derivatives.
- **NISM Series-VIII: Equity Derivatives Certification**
- **Published Research Paper in IEEE Explore:** Enhancing Retrieval-Augmented Generation Systems Using AI Models and Graph Databases.
- **AWS Certified Solutions Architect** — Designed multi-AZ, disaster-resilient cloud architectures.