

Pulkit Chauhan

+91-8445825150 | chauhanpulkit1708@gmail.com | [Linkedin](#) | [Github](#) | [Codeforces](#) | [Website](#)

Achievements

- **Codeforces Candidate Master** (rating **1953**, peak **1985**, handle `pulkit`) — top finishes of **96th**, **183rd**, **367th**, and **369th** in Codeforces Rated Div. 2 contests, consistently solving 4–6 problems against a global field.

Low-Latency Systems

PitchFork - NASDAQ ITCH 5.0 Order Book Engine | *C++17, io_uring, WASM* | [Code](#) | [Live](#) 2025 – Present

- Built an event-driven ITCH 5.0 binary feed handler and full-depth limit order book in **C++17** sustaining **~250 MB/s** on full-day NASDAQ data; every optimization verified against a deterministic **FNV-1a book-state fingerprint**, making refactors provably behavior-preserving.
- Replaced sorted-vector price levels with a **directly-indexed 2048-slot windowed price ladder** ($O(n)$ insert/erase $\rightarrow O(1)$): **3.2× improvement in p99.9 per-message latency**, measured with a custom `rdtsc+lfence` harness; diagnosed and fixed window-thrashing on low-priced symbols.
- Unified two per-order hash maps into one order registry, halving hot-path lookups for a **1.9× throughput gain**; profiled PGO and kept it *out* after data showed no benefit on this memory-latency-bound workload.
- Added an **io_uring** ingestion path (registered fixed buffers, QD=4 read-ahead) behind a `ByteSource` abstraction guaranteeing framing across chunk boundaries; warm/cold-cache benchmarks vs. `mmap` isolated parsing, not I/O, as the bottleneck.
- Extracted an I/O-free **Engine/Strategy** backtester and ran a multi-symbol **order-flow-imbalance (OFI) study** across all **65,536 NASDAQ locates**, reporting per-symbol hit-rate, t-statistic, profit factor, and spread-cost-adjusted bps P&L; on 2021-07-13 data **AAPL showed a 59.48% hit-rate at 32σ significance**.
- Compiled the I/O-free core to a **46 KB WebAssembly module** with fingerprint-identical output — **WASM (17.3M msg/s) edged native (16.4M msg/s)** on a 22.8M-message slice — powering a **live in-browser backtester** (watchlist grid, equity panel, adjustable replay interval); a lock-free SPSC queue (`alignas(64)` atomics, acquire/release ordering) streams snapshots over UDP to a Next.js dashboard, and a synthetic ITCH generator drives CI correctness tests.

Open Source

Google Summer of Code 2026 Contributor

June 2026 – Present

CCExtractor Development — sample-platform (CI/testing infrastructure)

Remote

- Architected a security-hardened REST API (documented via a 2,800+ line OpenAPI v3 spec) featuring **argon2**-hashed tokens, scoped RBAC, and tiered rate-limiting; merged across 7 PRs with a 100% SonarQube pass rate.
- Audited and patched critical infrastructure vulnerabilities: resolved `before_request` hook race conditions, worker-level brute-force multipliers, and a fail-open token expiry mechanism.
- Engineered a cross-platform CI test harness (Linux/Windows) supporting idempotent execution and multi-backend storage resolution (GCS/Local), fixing latent file-sharing violations in production.

Projects

Band-Former — Audio-to-Guitar-Tab Pipeline | *C++17, Python, PyTorch, CMake*

2025 – Present

- Engineered an end-to-end transcription pipeline using BS-Roformer (16.5 SDR separation) and OpenCV/HMM video-to-tab fusion; orchestrated the C++/Python boundary via CMake, deploying the PyTorch stack to Blackwell (sm_120) GPUs.
- Designed a C++17 beat-synchronous Viterbi chord classifier with CQT lateral inhibition, feeding a shortest-path fingering solver that models fretboard playability via Viterbi DP / ILP and biomechanical cost functions.

Education

University of Petroleum and Energy Studies (UPES)

Dehradun, India

B.Tech in Computer Science (DevOps Specialization) | CGPA: **8.48/10.0**

Aug. 2023 – May 2027

Technical Skills

Languages: C++17, Python, TypeScript, Go, SQL, Bash

Low-Latency: lock-free data structures, cache-line alignment / false sharing, memory ordering, `io_uring`, `mmap`, CPU affinity, `rdtsc` microbenchmarking, `perf`, WebAssembly, binary wire protocols (ITCH 5.0)

Infra & Backend: Linux, CMake, Flask, PostgreSQL, Redis, Docker, GitHub Actions CI