

Jiemin (Jenny) Zhang

Backend and AI engineer

London, UK | jenny.tuetoo@gmail.com | [LinkedIn](#) | [GitHub](#) | jieminzhang.me

Profile

Software engineer who moved into tech after twelve years as a corporate accountant and an MSc in Finance and Investment. Completed the 42 London Common Core (100%) in 12 months, with every project passed at full marks. I build FastAPI services, event-driven backends and LLM agents that are measured with benchmarks, and I understand the layer underneath: threads, queues, signals and containers. Looking for a backend or AI engineering role.

Skills

Daily use: Python, Git, Linux command line, Docker

Built in projects: FastAPI, SQLAlchemy 2, PostgreSQL, Redis (cache and Streams), Qdrant, JWT and OAuth, pytest, GitHub Actions

AI engineering: LLM agents, agent memory, RAG, BM25, Reciprocal Rank Fusion, constrained decoding, benchmarking

Worked alongside: code sandbox and MCP tool servers (Agent Smith), Nginx, Prometheus and Grafana (AlphaMind), built by teammates; I studied and integrated them

Working knowledge: C, C++ (POSIX threads, message queues, signals), Bash

Learning next: running services in production, distributed systems, SQL in depth, Rust

Projects

AlphaMind (ft_transcendence)

Jun – Oct 2026 | team of three

Multi-agent AI research platform for retail investors. Python, FastAPI, PostgreSQL, Redis, Qdrant, LangGraph, Docker.

- Product owner and backend lead: product scope, the three services (auth, core, AI), the database layer, tests, CI/CD and the cloud deployment, and review of all changes. One teammate led the infrastructure and one the frontend.
- Designed three independent FastAPI services with separate database schemas and no cross-service foreign keys. Services talk through REST and events; key events use Redis Streams with consumer groups and retries, with an outbox in the core service, published only after the database commit. 17 containers start with one command.
- Account security: bcrypt, short-lived JWT access tokens, rotating refresh tokens stored as hashes with replay detection, TOTP two-factor login, OAuth with state and PKCE, role-based permissions and rate limits. Redis holds live user state so a ban or role change applies at once.
- AI service: a LangGraph pipeline with up to nine specialist agents (market, news, sentiment, filings, fundamentals, technicals, risk, macro, plus portfolio when holdings exist), ReAct loops, a bull and bear debate and a reflection step that can send a result back for another try. Developed with AI coding tools; I set the requirements, reviewed the changes, ran the evaluations and decided each change.
- Retrieval and memory: hybrid search (BM25 plus vectors in Qdrant, merged with Reciprocal Rank Fusion), a knowledge graph, and layered agent memory (track record per ticker, tool-use patterns per role, archived reports) with quality gates on writes.
- Quality: batch-run evaluations of the AI service with fixes for the systematic errors found; a pytest suite against real PostgreSQL and Redis (written with AI assistance), plus flake8 and mypy --strict gates in CI.

Agent Smith

May – Jun 2026 | team of two

Autonomous coding agent, benchmarked on MBPP and SWE-bench. Python, Docker, MCP, pytest.

- Built the agent loop (Thought, Code, Observation) with step, token and time limits and per-step metrics, a parser that accepts several answer formats, and an LLM client that rotates API keys and retries on rate limits.
- Docker lifecycle for SWE-bench: one container per task, the agent works on the repository inside it and the result is collected as a git patch. My teammate built the code sandbox and the MCP tool servers; we integrated and tested the two halves together and each studied the other's part.
- Team benchmark of 11 models on free-tier quotas, using a harness that stores every run. On all 257 MBPP tasks the best model scored about 90%; on SWE-bench the best solved 8 of 8 tasks in its best run. Models tuned for code did not beat general models.

RAG over the vLLM codebase

Mar 2026 | individual

Question answering over a large code repository. Python, BM25, TF-IDF, Reciprocal Rank Fusion, Pydantic.

- Hybrid retrieval (BM25 and TF-IDF with a code-aware tokenizer, merged with Reciprocal Rank Fusion), AST-based chunking at function and class boundaries, and multi-query expansion for vocabulary mismatch.
- Evaluation framework and a 100-query benchmark: Recall@5 of about 0.76 on documentation and 0.66 on code, above the project thresholds of 0.55 and 0.45. Measured that multi-query expansion costs 3 to 4.5 times more latency and helps only on code.
- Latency work with BM25S and hash-based caching; separate modules for chunking, indexing, retrieval, generation and evaluation so each can be tested and swapped alone.

Kafka-style message system in C++ over POSIX message queues.

- Built the server and concurrency: one main queue and one reply queue per client, a reader and a worker thread per topic, and an append-only log with offsets so a reconnecting consumer resumes where it stopped.
- Graceful shutdown: SIGINT and SIGTERM are blocked and handled with ppoll and the original mask, which closes the race before a blocking call; topics stop with a poison pill. Wrote the hash map (FNV-1a, chained buckets, rehash at 0.75). A teammate built the client, codec, prefix index and tests.
- One worker thread per topic keeps consumer state free of locks and messages from one producer in order; a full consumer queue is waited on for at most 200 ms, then the message is dropped for that consumer.

Codexion

Jan 2026 | individual

Multi-threaded scheduler in C with POSIX threads.

- FIFO and EDF scheduling over a priority queue (binary heap); both dongles are taken inside one scheduler critical section, so a coder never holds one while waiting for the other (no hold-and-wait, no deadlock).
- One scheduler lock keeps decisions deterministic, trading some throughput for correctness.
- Monitor thread that wakes just before the earliest deadline and detects burnout within 10 ms; separate mutexes for scheduler, state and time data. A 200-coder stress test finishes in about a second with both policies.

Call Me Maybe

Feb 2026 | individual

Constrained decoding for function calling with Qwen3-0.6B. Python, Pydantic.

- Logit masking: tokens that cannot continue a valid function name get a score of minus infinity, so the function name is always valid; argument values are extracted by rules. Prefix matching on cached token tables; Pydantic validation.

Other 42 projects (code on GitHub)

2025 – 2026

- **Inception:** Docker Compose stack from custom Dockerfiles: NGINX with TLS, WordPress, MariaDB, Redis, Adminer, Portainer, FTP; Docker secrets and volumes.
- **Fly-in:** drone routing on a graph with zone and link capacity limits; my own Dijkstra-based planner and turn-by-turn simulator, no graph libraries.
- **The Answer Protocol:** real-time multiplayer text game over a line-based TCP protocol; async Rust server and command-line clients (team project).
- **A-Maze-ing and Pac-Man:** maze generation (DFS, Prim, Kruskal) and BFS solving as a reusable Python module; Pac-Man game with ghost modes and a state machine.
- **NetPractice, Born2beroot, libft, push_swap and 11 Python modules:** networking, a hardened Debian VM, C fundamentals, algorithms, OOP and Pydantic.

Education

42 London, Software Engineering

Oct 2025 – present

- Piscine Sep 2025; Common Core 100% in Oct 2026 (usually 1 to 2 years); every project passed at full marks, including the bonus part wherever a project has one. Exams Rank 02 to 06 passed. Now in the 42 Advanced part.

MSc Finance and Investment (Merit), Durham University

Sep 2023 – Jan 2025

- Thesis (mark 72, distinction level): options trading and the stock market. Three years of S&P 500 panel data, R and Python, fixed- and random-effects models. Options volume raises stock volume and helps predict price moves.

BSc Financial Management, University of International Business and Economics, Beijing

2007 – 2011

Experience

Corporate Accountant, Gallup (Beijing)

Aug 2013 – Sep 2023

- Ran end-to-end accounting for Gallup China and Hong Kong (payables, receivables, billing, payroll, tax, cost control), taking on more areas over ten years.
- Built annual budgets, rolling cash-flow forecasts and monthly variance reports that senior management used for planning and decisions.
- Acted as financial business partner: built pricing and contract models with the sales and operations teams.
- Improved payroll, payables and onboarding processes; passed every annual financial, tax and internal audit with zero discrepancies.

Accountant, Schneider Electric (Beijing)

Jul 2011 – Jul 2013

- Top performer in the shared services team, consistently above targets. Standardised reconciliation methods that the whole team adopted; wrote process manuals and trained new hires.

Hackathon and qualifications

BearingPoint AI Hackathon at 42 London, 1st place (Dec 2025). Designed and prototyped a multi-agent sales intelligence copilot with a winning team of six.

Qualifications: Microsoft Azure AI Fundamentals (2024), CFA Program Level I (2023), CICPA Certified Public Accountant (2019).