TAI TRAN

Brisbane, Australia
0479 049 287 | Email: taitranz@ymail.com LinkedIn: linkedin.com/in/taitranz Portfolio: taitranz.com
GitHub: github.com/Taitranz Blog: blog.taitranz.com

PROFILE

Software engineer focused on outcomes: turning ambiguous requirements into shipped, maintainable software, owning delivery end-to-end, and balancing timelines, engineering capacity, and technical trade-offs. Primarily Python, but language-agnostic and happy to use the right tool for the job.

TECHNICAL SKILLS

- Languages: Python (primary), Java, C#, SQL, HTML, CSS, JavaScript.
- Frameworks & Libraries: FastAPI, PySide6 (Qt6), JavaFX, VisPy (OpenGL), NumPy, CuPy, Pandas, Flask, SQLAlchemy, Alembic.
- Databases: PostgreSQL, SQLite (WAL mode), SQLAlchemy ORM.
- Architecture & Practices: Clean Architecture, Domain-Driven Design, CQRS, Event-Driven Architecture, repository pattern, dependency injection, MVVM, type safety (MyPy), linting (Ruff), testing (pytest, pytest-qt, unit testing), CI with coverage gating, Logging & Observability, Information Security, REST APIs, Software Architecture Design, OOP.
- Tools & Platforms: Git/GitHub, Stripe, Cloudflare, OANDA API, Figma, MkDocs, Parquet.
- Domains: Real-time trading & charting, AI-native workflows, Cybersecurity, GPU Computing (CUDA), High-Performance Computing, Financial Data Visualisation, Machine Learning, UI/UX.

PROFESSIONAL EXPERIENCE

Lead Software Engineer (Contract) | Lewy Security (Agentic Security Hardening)

Nov 2025 - Present | Sydney, New South Wales, Australia (Remote)

Major Duties:

- **End-to-End Architecture Ownership:** Architected a modular developer-facing security platform (45k+ LOC) in Python using Clean Architecture. Designed a plugin system for 12+ security agents, enabling extensibility with strict type safety.
- **Security Analysis Orchestration:** Built a workflow that normalised results from 5+ static analysis tools into a unified model. Engineered a parallel pipeline to ingest and deduplicate 500+ findings/sec locally.
- **Hotspot Identification:** Implemented risk-density modelling to identify the top ~10% of security-sensitive hotspots, reducing time-to-signal by ~60% via targeted prioritisation.
- **AI-Assisted Triage:** Integrated LLM triage to cluster thousands of alerts into semantic groups. Reduced false-positive noise by ~75% using multi-stage verification against allowlists and known-safe patterns.
- **Controlled Remediation:** Implemented an assisted-fix workflow with patches scoped to <8k-token windows. Achieved ~65% first-pass acceptance by constraining LLM output to verified syntax and repository context.
- **Auditability & Tooling:** Developed a sub-50ms-startup CLI for local and CI usage. Built a SQLite-backed tracking system handling 50,000+ audit events with fast queries and reproducible runs.

Successes:

- **Review Efficiency:** Reduced review time from hours to minutes by grouping findings, enabling developers to triage 50+ issues per batch.
- **Context Optimisation:** Reduced LLM token usage by ~40% via AST-based slicing while maintaining remediation accuracy and lowering cost.
- **Modular Scalability:** Added 3 new analysis domains (Secrets, IaC, Config) in under 2 weeks without refactoring core orchestration logic.
- **Reliability:** Maintained ~90%+ test coverage across the orchestration engine, avoiding regressions during rapid iteration.

Lead Software Engineer | Hooper Music Studio (Contract)

Nov 2025 - Jan 2026 | Brisbane, Queensland, Australia (On-site)

Major Duties:

- **System Architecture:** Architected a scalable platform using FastAPI/PostgreSQL, modelling 20+ domain entities. Managed schema evolution through 40+ zero-downtime Alembic migrations, tying together RBAC, temporal scheduling, and double-entry financial ledgers.
- **Advanced Scheduling Engine:** Engineered a conflict-free booking kernel evaluating 150+ weekly time slots per teacher against constraints (holidays, recurring patterns, 24-hour lockouts). Implemented temporal rules to regenerate lesson series on reschedule while preserving referential integrity and schedule correctness.
- **Proprietary Financial Core:** Built a custom double-entry accounting system to replace external tools. Developed a heuristic CSV parser to normalise multi-format bank feeds and automate reconciliation across thousands of transaction rows, producing near real-time P&L; reporting.
- **Payments & Commerce:** Integrated Stripe webhooks with a digital asset store, linking point-of-sale events directly into the accounting ledger. Automated dynamic PDF invoice generation using ReportLab, eliminating manual invoicing for standard flows.
- **Security & Ops:** Implemented zero-trust principles with OAuth2, 2FA, and granular RBAC. Established CI/CD with strict mypy, ruff linting, and automated tests to protect complex business logic (e.g., exam management) from regressions.

Successes:

- **Digital Transformation:** Reduced administrative overhead by ~90% by replacing manual spreadsheets with an event-driven workflow for payroll, invoicing, and lesson tracking.
- **Front-end Architecture:** Delivered a responsive UI by refactoring into 30+ modular ES6 components, including custom calendar controls and dark mode without a heavyweight framework.
- **Rapid Feature Delivery:** Used a modular service layer to deliver complex features (e.g., Lesson Escalations, Reschedule Credits) in sub-2-week sprint cycles while maintaining production stability.

Lead Software Engineer (Contract) | Valgo Trading

Jun 2025 - Dec 2025 | Brisbane, Queensland, Australia (Hybrid)

Major Duties:

- **System Architecture:** Architected a modular desktop platform (200k+ LOC) using Python and Clean Architecture. Applied CQRS to strictly separate deterministic trading logic from statistical analysis, improving testability and enabling modular packaging.
- **AI Agent Integration:** Engineered an autonomous decision layer using fine-tuned LLMs. Reduced hallucination risk by grounding agent reasoning in structured “Market State” objects rather than raw price noise.
- **GPU Rendering Engine:** Built a GPU-accelerated VisPy/OpenGL pipeline to render 10,000+ interactive candlesticks in real time. Used CuPy for GPU-backed data preparation and texture packing, and wrote custom shaders to sustain ~160 FPS on high-density datasets, avoiding bottlenecks in standard Python plotting libraries.
- **Concurrency & Threading:** Developed a thread-safe event bus to manage high-throughput data ingestion. Resolved GIL-related UI freezing during heavy AI inference by implementing a non-blocking immutable “snapshot” pattern for state synchronisation.
- **Execution & Data:** Integrated real-time OANDA feeds into a unified state engine. Designed the system to use identical logic paths for both backtesting and live execution, ensuring simulation validity.

Successes:

- **Performance Optimisation:** Outperformed standard Matplotlib implementations by ~5×, sustaining ~160 FPS for high-density datasets via GPU texture packing.
- **Reliable Autonomy:** Delivered a stable human-in-the-loop system where LLM decisions are transparently explained and visually correlated with market structure in real time.
- **Stability:** Eliminated race conditions between the AI and UI threads by enforcing immutable state snapshots, ensuring data consistency during high-volatility events.

KEY PROJECTS

PySide6 Date Range Popover - Reusable Date/Range Picker Component - Python, PySide6, Qt, pytest, pytest-qt, MkDocs

- Built a reusable date/date-range picker widget with clean Qt signals to decouple UI from business logic.
- Packaged as an installable library with semantic versioning, documentation (MkDocs), and CI-tested coverage (pytest + pytest-qt).

EDUCATION

Bachelor of Information Technology - Major: Computer Science; Second Major: Computational and Simulation Science

Queensland University of Technology (QUT), Brisbane

WRITING & TECHNICAL COMMUNICATION

- “Valgo: An Agentic Trading System for EUR/USD” - deep-dive on an agentic EUR/USD trading platform combining Clean Architecture, event-driven design, and an LLM-backed agent to go from raw market data to autonomous execution.
- “Your AI Assistant Doesn’t Need Better Inference. It Needs Better Documentation” - described a two-layer documentation architecture (AGENTS.md entry point + .agent tree) to improve AI-assisted development by giving tools structured, high-signal context.
- “NOV11 Valgo Progress Update” - documented architectural decisions, performance characteristics, and lessons learned from scaling a Clean Architecture trading terminal.

REFERENCES

References available upon request; reference page will be provided.