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 specialising in AI-native workflows, trading automation, and full-stack delivery. Passionate builder focused on robust, real-world systems that drive business impact. Primary strength is Python (FastAPI, PySide6, NumPy), but language-agnostic in choosing the right tool for the job. Committed to shipping high-quality code with real ownership, from architecture to production. Based in Brisbane; open to international relocation.

TECHNICAL SKILLS

- Languages: Python (primary), Java, C#, SQL, HTML, CSS, JavaScript.
- Frameworks & Libraries: FastAPI, PySide6 (Qt6), JavaFX, VisPy (OpenGL), NumPy, CuPy, Flask, SQLAlchemy, Alembic.
- Databases: PostgreSQL, SQLite (WAL mode), SQLAlchemy ORM.
- Architecture & Practices: Clean Architecture, Domain-Driven Design, CQRS, event-driven design, repository pattern, dependency injection, MVVM-style GUI structure, type safety with MyPy, linting with Ruff, testing with pytest/pytest-qt, CI with coverage gating, Logging & Observability, Information Security.
- Tools & Platforms: Git/GitHub, Stripe, Cloudflare, OANDA API, Figma, MkDocs, pytest, Parquet.
- Domains: Real-time trading and charting, language learning apps, ticketing/marketplace systems, AI-assisted development workflows, Cybersecurity.

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:** Designed and built the entire codebase for a developer-facing security automation platform, owning architecture and extensibility from concept through implementation.
- **Security Analysis Orchestration:** Built a unified workflow that coordinates multiple security analysis capabilities, normalising results into a structured findings model for review, prioritisation, and lifecycle tracking.
- **Hotspot Identification & Risk Focus:** Implemented codebase modelling to identify security-sensitive hotspots, enabling targeted inspection instead of broad, low-signal scanning.
- **AI-Assisted Triage Pipeline:** Integrated LLM-based triage to cluster related findings, rank issues by likely impact, and flag probable false positives, with strict boundaries around context exposure.
- **Controlled Remediation Workflow:** Implemented a guided fix process where accepted findings are reviewed in isolation and remediation suggestions are generated from scoped context only.
- **Auditability & Tooling:** Developed persistent tracking for detection, review decisions, dismissals, accepted risks, and remediation outcomes, delivered via a fast CLI suitable for local and CI use.

Successes:

- **Higher Signal, Less Noise:** Reduced review overhead by improving grouping and prioritisation so developers focus on the most meaningful risks.
- **Safe AI Usage:** Kept AI-assisted analysis bounded to relevant context to avoid unnecessary exposure of internal system details.
- **Modular Foundation:** Established an architecture that supports adding new analysis domains and workflows without disrupting core functionality.
- **Maintainable Delivery:** Sustained a clean, traceable codebase that supports rapid iteration while preserving correctness.

Software Engineer | Hooper Music Studio (Contract)

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

Major Duties:

- **System Architecture:** Designed and developed a full web application using FastAPI, PostgreSQL, and Jinja2. Implemented a modular service pattern with SQLAlchemy ORM and Alembic migrations to manage complex relationships across 13+ domain models including users, lessons, invoices, and accounting.
- **Advanced Scheduling Engine:** Built a conflict-free booking system supporting recurring semester slots, teacher availability, and ad-hoc changes. Integrated FullCalendar for interactive scheduling and implemented custom logic for reschedule credits and 24-hour lockout rules.
- **Financial Infrastructure:** Developed a double-entry accounting module with automated bank reconciliation, CSV parsing with regex-based categorisation, synthetic ledger entries for invoices, and real-time financial reporting including profit and loss and balance sheet views.
- **Payments and Invoicing:** Integrated Stripe payments and webhooks for status updates. Created a custom PDF generator using ReportLab for invoices and receipts with dynamic line items, credit handling, and tax calculations.
- **Security and Access Control:** Implemented secure authentication including OAuth2 (Google/Facebook/Apple), 2FA, RBAC (Admin/Teacher/Student/Developer), and Cloudflare Turnstile for bot protection.

Successes:

- **Business Automation:** Reduced administrative workload by automating lesson reminders, invoice delivery through Resend, and teacher payroll calculations.
- **UX/UI Improvements:** Delivered a mobile-responsive frontend with dark mode, custom CSS micro-interactions, and accessibility-focused adjustments to improve parent and student experience.
- **Reliability:** Maintained a stable codebase with strict type checking (mypy), automated linting, and safe database migration patterns supporting rapid feature rollout.
- **Feature Integration:** Added a digital Book Store and Instrument Catalogue fully linked to invoicing, enabling inventory tracking and streamlined point-of-sale operations.

Lead Software Engineer (Contract) | Valgo Trading

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

Major Duties:

- **System Architecture:** Designed a modular desktop application using Python, PySide6, and VisPy. Implemented Clean Architecture with DDD, CQRS, and Event-Driven patterns to support a general-purpose quantitative analysis layer where deterministic and statistical modules publish typed outputs from normalised live data.
- **AI Agent Integration:** Engineered an autonomous decision layer using a fine-tuned Large Language Model. The system consumes structured market snapshots (trend state, volatility, sentiment) rather than raw prices to generate trade decisions with transparent rationales, emulating expert reasoning.
- **GPU Rendering Engine:** Built a custom rendering pipeline using VisPy and CuPy (CUDA) for real-time visualisation of high-frequency data and technical indicators with sub-millisecond latency.
- **Execution & Data:** Integrated real-time OANDA tick data and news streams into a unified queryable state. Developed a dynamic execution engine to handle order routing, position sizing, and risk constraints in both simulation and live modes.
- **Concurrency:** Developed a thread-safe Event Bus with event coalescing to manage high-throughput streams, using mutex protection to synchronise background data ingestion with the UI.

Successes:

- **End-to-End Autonomy:** Achieved a fully autonomous loop that scans the market, interprets structure via the LLM agent, generates explanations, and executes trades with consistent precision.
- **Performance Optimisation:** Achieved 60fps rendering of large datasets by leveraging GPU texture packing and optimising OpenGL draw calls, outperforming standard plotting libraries.
- **Reliability:** Solved integration challenges between the AI layer, UI, and data feeds by implementing a robust thread lifecycle system and 'quiet mode' for event handling.
- **Observability:** Established a logging system with correlation tracking and profiling to monitor the decision-making pipeline from data ingestion to trade execution.

KEY PROJECTS

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

- Developed a reusable date and date-range popover widget exposing clear Qt signals so host applications can connect business logic without touching internal widget structure.
- Centralised configuration through a DatePickerConfig object for picker mode, initial date/range, theming, and bounds.
- Packaged the component as an installable Python project with semantic versioning and documentation hosted via MkDocs.
- Built an automated test suite (pytest + pytest-qt) with offscreen Qt configuration and high coverage, integrated into CI with coverage reports.

Socslingo - Duolingo-Inspired Desktop Language Learning App - Java, JavaFX, SQLite, CSS, SLF4J/Logback

- Implemented a Japanese learning app with registration/login, profile management, flashcards, multi-deck organisation, and interactive mini-games.
- Built an MVC-style architecture with controllers, services, DAOs, and managers, supporting several thousand lines of Java while keeping data flow consistent.
- Designed character-recognition activities with progress tracking, heart-based life system, and animation-heavy UI using JavaFX timelines and transitions.
- Implemented a central SceneManager and image caching system leveraging JavaFX Service/Task and a ConcurrentHashMap-backed cache.

Seek Music - Concert Ticketing Web Platform - Python, Flask, SQLAlchemy, SQLite, HTML, CSS, JavaScript

- Built a full-stack ticketing platform where organisers create/manage events and users browse, search, and book concert tickets.
- Implemented authentication with hashed passwords, user profiles, role-based behaviour, and a booking system with real-time ticket availability and status.
- Added media handling, rich search/filter UI, and separate views for user bookings and organiser listings.
- Produced a detailed post-mortem covering security, database design, performance, and testing improvements.

EDUCATION

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

Queensland University of Technology (QUT), Brisbane

Coursework projects include Seek Music and Socslingo, delivered as major assessment pieces.

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.
- Socslingo and Seek Music post-mortems - long-form reflections analysing trade-offs, technical debt, security issues, and refactoring plans for both projects.

REFERENCES

References available upon request; reference page will be provided.