June Kim

Vancouver, Canada | june@june.kim | june.kim | github.com/kimjune01

Summary

Evaluation and agent-reliability engineer who turns ambiguous claims about coding-agent behavior into experiments, graders, and replayable evidence. 10+ years shipping production software at Google, Loom, and startups. Strongest where software engineering meets measurement: defining what a system must prove, building the cheapest test that can break it, and revising the conclusion when the instrument fails.

Work Experience

Independent | Evaluation & Applied AI Engineer *2025 – present*

- Moved from client AI systems into coding-agent evaluation, carrying production disciplines such as tests, controls, observability, and replay into benchmark research.
- Audited all 728 SWE-bench Pro tasks and established a pointer-checkable 15% underdetermination floor; three reference patches fail the benchmark's own verifier. Evidence: june.kim/a-determinacy-audit-of-swebench-pro
- Showed that 83 of 83 officially passing Terminal-Bench runs still passed after off-task assets were destroyed; implemented frame checks for Harbor and Inspect AI. Evidence: june.kim/terminal-bench-frame
- Applied every DeepSWE reference answer to its verifier and found four failures in 113 tasks; exactly those four improved in the later v1.1 regrade while the pooled rate stayed flat. Evidence: june.kim/auditing-deepswe
- For startup clients, shipped LLM classification, deduplication, OAuth integrations, CI quality gates, E2E infrastructure, and an accessibility-data ingestion pipeline.

Little Bird Software | Applied AI Engineer 2024 – 2025

- Built agentic ingestion pipelines using LLM condensation and deduplication to cut noise 90%; developed Python orchestration services and zero-downtime migrations.
- Architected macOS and Chrome integrations in Swift, Tauri, and Rust over Accessibility APIs for real-time context injection.
- Built the end-to-end RAG path from unstructured desktop data through filtering, extraction, parsing, condensation, and vector retrieval.

Loom | Senior Software Engineer 2022 – 2023

- Raised core video reliability from 97% to 99.7% through multiresolution UI and Shaka Player integration; led a TypeScript refactor of the Electron desktop app.

YouTube / Google | Software Engineer 2019 – 2022

- Turned product requirements into design documents and production implementations while re-architecting the YouTube iOS rendering layer in C++ and TypeScript, reducing UI deployment cycles from months to days.
- Directed the launch of a Premium sign-up framework that increased conversion 2% across 50M+ users; conducted dozens of L3/L4 technical interviews.

Earlier Product Engineering | Firework, Lipsi & Startups 2013 – 2019

- Shipped mobile and web products in Swift, Objective-C, React Native, Rails, and AWS; helped grow Lipsi from launch to 2.3M users and #1 in the US Lifestyle category.
- Trained multiple Lighthouse Labs iOS cohorts from first principles to junior-developer readiness; graduates joined teams including Nike and Hootsuite.

Research Mechanisms & Open Source

- Assurance at the Boundary: synthesized eight public-side benchmark audits into a cost-ordered protocol spanning claim, specification, oracle, frame, gold, score, and decay failures.
- Replayable Claims: a protocol that transmits a claim with its deterministic re-derivation, separating testimonial uncertainty from specification uncertainty. DOI-archived preprint with claim-level receipts.
- The Hypothesis Graph: testable claim nodes and refutation-condition edges that make an agent's inquiry inspectable and reusable; used as the reasoning substrate behind the contribution and audit workflows.
- Experimental judgment: rejected four broken intent-extraction instruments, added a missing control, and withdrew an apparent 13.8% history effect when the control revealed a 14.1% run-to-run noise floor; published the narrower result and worklog.
- Built an issue-discovery, patch, test, and review loop that produced 111 merged pull requests across 89 external repositories in Rust, Go, C++, and Python. Maintainer decisions provide an external outcome measure.
- Representative upstream work: reproduced two Enzyme compiler defects from a structural soundness gate and separately landed two autodiff fixes; merged fixes also include godot, hyper, envoy, servo, TiDB, Flux, and wild.

Education

Bachelor of Science in Computing (second degree) | Simon Fraser University 2015 – 2017

- Coursework included machine learning, systems, algorithms, databases, security, computer vision, programming languages, and numerical analysis.

Bachelor of Business Administration | Simon Fraser University 2008 – 2012

- Product, operations, strategy, finance, accounting, management science, and entrepreneurship.

Skills

- Evaluation: construct validity, benchmark auditing, grader and harness design, agent behavior analysis, red-teaming, contamination, label and ground-truth audits, experimental controls, preregistration, reproducible artifacts
- Engineering: Python, Rust, TypeScript, C/C++, Go, Swift; FastAPI, CI/CD, Git/GitHub; sandboxed and tool-using agents, RAG, embeddings, MCP