Yeon Lee

+1 (678) 644-3180 — yeonholee50@gmail.com — yeonthelee.tech
linkedin.com/in/yeon-lee — github.com/yeonholee50

Professional Summary

Backend and full-stack engineer focused on building and improving data pipelines, messaging, and observability. Designs typed, transport-agnostic contracts and builds services that are deterministic, modular, testable, and easy to operate.

Experience

Attachments King Feb 2025 – Jul 2025
Software Engineer — AI Schema *San Francisco, CA*

- Standardized AI rule authoring across teams by introducing a centralized cursor-rule router; adoption as the single rules pathway reduced fragmentation and simplified maintenance.
- Replaced ad-hoc rules with a tree-based compatibility engine on AWS Neptune that encodes hydraulic power, interface, and loader-arm constraints and emits valid combos dynamically, cutting test-matrix generation from days to under 1 hour.
- Enabled same-day price updates across about 20k SKUs/week and reduced bad-data incidents by >86%, by normalizing PDF/CSV/HTML sources with drift/orphan detection and auto-publishing clean SKUs.
- Removed manual CSR checks for 12+ vendors and improved availability freshness to p95 < 10 min, by deploying a reusable, real-time scraping pipeline for non-API suppliers.

Techrupt Innovations Dec 2024 – Feb 2025
Software Developer *Remote*

- Improved momentum estimator validation accuracy by 0.5 percentage points by refining feature generation and establishing reproducible training and evaluation runs.
- Shortened research cycles by introducing versioned configurations and disciplined run tracking for consistent experiment comparisons under NDA constraints.

LymphaTech Aug 2023 – May 2024
Backend Developer *Atlanta, GA*

- Enabled reliable multi-device access with fewer synchronization conflicts and lower resource use by adopting a two-stage UI–server communication pattern with flag-based sync.
- Held contour measurement error within four percent of ground truth by tuning PyTorch components and integrating Open3D-based contour line generation.
- Improved delivery predictability across biweekly sprints by coordinating Scrum ceremonies and aligning stakeholder feedback via transparent Jira workflows.

Projects

AmpyFin (Open Source) *OSS AmpyFin*

- **Purpose** — Provide a plug-and-play platform for live trading systems with strict message contracts, deterministic replays, and swappable layers for ingestion, transport, configuration, and observability.
- **Architecture** — Layered design with clear boundaries:
  - Schemas — Canonical Protobufs (`bars.v1`, `ticks.v1`, `fundamentals.v1`) in ampy-proto; explicit decimal & currency semantics; `event_time`, `ingest_time`, `as_of` timestamps.
  - Messaging — Standard envelope and headers (run ID, universe ID, trace context, QoS) with NATS JetStream and Kafka bindings in ampy-bus.
  - Configuration — Typed, layered configs with validation and secret indirection in ampy-config.
  - Observability — Uniform logs, metrics, and traces (OTLP) with SDKs and a ready-to-run stack in ampy-observability.

- **Operations** — Reference Docker Compose, golden samples, and CI smoke tests ensure consistent local bring-up, schema evolution, and deterministic bus replays.
- **Open-source stance** — Core infra is open to foster ecosystem growth; proprietary strategy logic can live out-of-tree behind stable interfaces.

yfinance-go *yfinance-go*

- **Purpose** — Delivered a free-data ingestion path with up to 8 concurrent ticker pulls and backoff/retry controls, matching AmpyFin's proto/bus/config/obs contracts so users can swap it for DataBento without code changes.
- **Architecture** — Standardizes output to `ampy-proto` messages; includes session rotation, backoff, circuit breakers, and both library and CLI interfaces; supports daily/weekly/monthly/intraday bars and quotes with documented limits; falls back to structured modular HTML parsing for views not exposed via API endpoints.

Euler System *Euler System*

- **Purpose** — Serve as a defensive overlay that governs exposure and execution using a continuously updated risk score (0–100) and regime label; discourage fragile behavior through position caps, leverage limits, order throttles, and circuit-breaker halts when conditions deteriorate.
- **Pipeline** — Sequential stages (Fetch, Process, Infer, System) separate acquisition, signal processing, inference, and policy application; hysteresis and decay prevent flip-flopping; configurations are versioned for audits.
- **Integration** — Publish outputs to console, rotating logs, an operator GUI, and LAN/bus so OMS and strategy services adapt automatically; in AmpyFin, Euler messages share the same contracts for replay alongside market data.

Val System *Val System*

- **Purpose** — Generate explainable, consensus fair values from multiple valuation models so operators can compare price vs. value, gate risk, and drive sizing with defensible inputs.
- **Pipeline** — Deterministic stages with a shared `PipelineContext`: *Fetch* metrics via single-purpose adapters (e.g., EPS, revenue TTM, FCF TTM, BVPS, shares outstanding, growth) selected through an adapter registry; *Process* enabled strategies (DCF Gordon, Earnings Power Value, Residual Income, EV/EBIT bridge, FCF yield, Lynch, multiples reversion) with captured assumptions; *Result* produces median consensus, then writes to console and (optionally) UDP broadcast, MongoDB, and a live PyQt5 GUI.
- **Integration** — Runtime is controlled via a simple CLI (`scripts/cli.py`) with flags for run-once/loop, GUI toggle, sleep interval, adapter overrides (e.g., choose `yfinance` for EPS while using another provider for price), and MongoDB storage; also includes a UDP listener for external consumers. Adapters use rotating sessions to avoid rate limiting.

NyxHub *NyxHub*

- **Overview** — Centralized, secure file routing by username with reliable retrieval; end-to-end encryption and one-command startup simplify operations; clear separation between core service and UI aids maintainability.

Jin Slackbot *Jin SlackBot*

- **Overview** — Workspace automation bot for message management, reminders, polls, and data interactions, built on Slack API and MongoDB with a maintainable command design.

Education

Georgia Institute of Technology Aug 2020 – May 2024

B.S. in Computer Science (Threads: Intelligence; Systems & Architecture) *Atlanta, GA*

- Selected coursework: Data Structures and Algorithms; Systems and Networks; Computer Networking; Design and Analysis of Algorithms; Artificial Intelligence; Computer Vision; Automata.

Skills

Languages: Python, Go, C/C++, Java, SQL, Bash, HTML/CSS, JavaScript

Infrastructure: Docker, Render, Linux, Grafana, Prometheus, OpenTelemetry, Kafka, NATS JetStream, Protobuf

Databases and Web: PostgreSQL, MongoDB, AWS Neptune, DuckDB, REST API, FastAPI, React

Tool: GitHub, Git, Linear, Jira, VS Code

Practices: Agile Methodology, SCRUM, CI/CD, TDD, Code Review, Documentation