Yiyuan Li

Singapore, River Vally | yy@eyuan.me | 895 234 47 | eyuan.me | linkedin.com/in/yyuanl

github.com/yuann3

Summary

After rediscovering my obsession with systems programming, I dove deep into low-level projects, building interpreters, servers, and Redis clones from scratch — all while leading peer learning sessions and tackling AI-driven backends.

Skills & Technologies

Programming Languages: Go, Rust, Python, C/C++, Java, SQL, JavaScript (Node.js), C#

Technology: Redis, SQLite, MSSQL, Docker, Flask, React.js, ASP .NET, .NET MAUI, Raylib, Neovim, GDB, Linux

Concepts: Systems Programming, Distributed Systems, RAG, API Design, Multithreading, TDD, CI/CD

Education

The University of Newcastle, Bachelor in Information Technology — Jan 2024 – Sep 2025

- **High Distinction:** Object-Oriented Programming
- **Distinction:** Data Structures & Algorithms, Advanced Database

PSB Academy, Singapore, Diploma in InfoComm Technology — Jan 2023 – Nov 2023

Experience

Peer Assisted Study Sessions (PASS) Leader, The University of Newcastle – Singapore — Feb 2024 – Present

- Mentored a group of 10 peers in Data Structures and OOP using C++ and Java, designing custom practice problems and annotated code snippets to reinforce core concepts.
- Delivered 10+ technical mini-lectures on algorithm design and coding practices such as recursion and sorting, using live code walkthroughs in Java and DSA to boost comprehension.

Cadet, Pisciner, Singapore University of Technology and Design (SUTD), École 42 Programme – Singapore — Sep 2024 – March 2024

- Completed 16 low-level system projects in C, including push_swap, libft, and pipex, mastering memory management, pointer arithmetic, and bash scripting.
- Collaborated in a 150member cohort using Git for version control and peer code reviews, while debugging memory leaks and writing unit tests in projects like getnextline and ft_printf.

Projects

Rego: Redis Server implementation in Go — Personal Project

- Implemented core Redis commands and WAIT replication in Go, passing full Codecrafters tests
- Applied slice optimizations to minimize GC overhead and improved throughput by 30%
- Tools Used: Golang, Database

Ruskey: Custom Programming Language Interpreter — Personal Project

- Built a full-featured interpreter for the Monkey language in Rust, including a lexer, parser, AST, and evaluator, to demonstrate parsing and language runtime implementation.
- Supported language features including booleans, integers, closures, and first-class functions to reflect real-world scripting capabilities.
- Tools Used: Rust, Test-Driven Development, Abstract Syntax Trees, Recursive Descent Parsing

Rust HTTP Server Personal Project

- Developed a multithreaded HTTP/1.1 server in Rust supporting GET/POST requests, file uploads, and gzip compression, with optimized request handling and concurrency.
- Implemented a User-Agent echo endpoint to assist in request debugging and improved response throughput through thread pooling and efficient I/O operations.
- Tools Used: Rust, HTTP/1.1, Multithreading

Pew: Lightweight CLI for Code Dumping Personal Project

- Built a CLI tool in Golang to package entire codebases into a Markdown file for streamlined input into LLM pipelines and documentation workflows.
- Implemented file parsing, Gitignore-style pattern matching, and syntax-highlighted output with tree-style directory visualization.
- Tools Used: Golang, CLI Development, File I/O

Hiraku: AI-Powered Smart Learning Companion The University of Newcastle

- Led backend architecture of an AI assistant using Python (Flask) and SQLite, integrating RAG pipelines for intelligent PDF/TXT document parsing and retrieval.
- Built a secure REST API with JWT-based access control and optimized document vector indexing via ChromaDB for fast semantic retrieval.
- Tool Used: JavaScript/TypeScript (React 19, Next.js), Tailwind CSS, Python (Flask REST API, JWT), Database (SQLite)

Ray Tracer in Rust Personal Project

- A ray tracer was developed from scratch in Rust. It features robust 3D vector mathematics, comprehensive operator overloading, and thorough unit testing. An efficient camera system was designed, incorporating viewport calculations and ray generation capabilities. Rust's type system was utilized to ensure strong compile-time guarantees and memory safety
- Tools Used: Rust, Linear Algebra, Computer Graphics, PPM Image Format

Interests

CLI tools, Computer Graphics, Rock Climbing, Traveling