Julius Liu

Full Stack Developer | TypeScript, React, Next.js

Email: CONTACT@JULIUSLIU.DEV Portfolio: JULIUSLIU.DEV

EDUCATION

Computer Programming Diploma, Algonquin College, Ottawa ON — Aug 2026 (Expected)

- Cumulative GPA: 3.99/4.0, Dean’s Honour List
- Relevant coursework: object-oriented programming in Java, system design and analysis, database management and administration, web development with PHP, computer networking

SKILLS

- **Backend**: JavaScript, TypeScript, Node.js, NPM, Express, Next.js, Zod, Postman, Vitest, PHP
- **Frontend**: HTML, CSS, Bootstrap, React, Redux, Formik, Tailwind CSS, Shadcn UI, Framer Motion
- **Database**: MySQL, PostgreSQL, MongoDB, Prisma, Drizzle, GraphQL, SQL Server, Oracle
- **Other languages & tools**: Java, Go, Python, Git, Husky, GitHub Actions, Docker, Linux, macOS
- **Domain knowledge**: data structures and algorithms, operating systems, AWS, test automation

PROJECTS

Morph (AI-powered essay editing) — Next.js, MongoDB, Prisma

- Integrated **Gemini API** with OpenAPI schema and prompt enginnering for **structured output**.
- Developed **accessible, responsive design and consistent UI** with Tailwind CSS and Shadcn UI.
- Incorporated **cache and data preloading** for better user experience, **lowering FCP by 80%**.
- Ensured deployment reliability with a **robust CI/CD pipeline** featuring **end-to-end testing** using Github Actions and Playwright.

Shuffle (note-taking tool with text highlighting) — Next.js, Hono, SQLite, Drizzle

- Designed a **normalized relational database**, implemented **user authentication**, securing **RESTful API endpoints** with **middleware**-based optimistic **authentication and authorization** checks.
- Delivered client-side data management using SWR for fetching, caching, and optimistic updates.
- Achieved **end-to-end type-safety** with **client/server validation** (Zod and React Hook Form) and Hono RPC client integration.
- Created a **custom React hook** and components for reliable and efficient retrieval, processing, and rendering of text selections.

Distributed MapReduce Word Count — Go, RPC, Concurrency

- Implemented a fully **concurrent task management** system using Go RPCs, with thread-safe coordination between workers and a shared coordinator.
- Built a **thread-safe coordinator** with Go’s sync.Mutex to manage concurrent RPC requests from workers, featuring **dynamic task scheduling, state tracking, and job completion detection**.
- Implemented **timeout-based task recovery** via per-task goroutines that reassign map or reduce jobs if not completed, providing **crash resilience and fault tolerance**.