Last updated in June 2025

Lukas M. Dutrieux

West Lafayette, IN | ldutrie@purdue.edu | (479)-899-4756 | ldutrieux.dev | LinkedIn | Github

Education

Purdue University *B.S. in Computer Science; 3.1 gpa* *August 2023 – May 2027*
Concentrations in Machine Intelligence & Database Systems

- **Relevant Coursework:** DSA,Systems Programming, Computer Architecture, Programming in C, OOP, Linear Algebra, Discrete Mathematics, Information Systems.
- **Computer Science Department Mentor** - Mentored 3 freshman students through academic and personal challenges, improving their understanding of core CS concepts and study strategies.

Technical Skills

Systems Programming: C/C++, Assembly, Bash, Unix/Linux, GDB, Valgrind, Memory Management

Backend Development: Golang, Java, REST APIs, Distributed Systems, Concurrent Programming

Languages & Technologies: SQL, LaTeX, YAML, Git, Docker, AWS RDS, GitHub Actions

Development Tools: Vim, Helix, GitHub Copilot, Vercel, Cron, Algorithm Design & Implementation

Experience

Backend Developer *West Lafayette, IN*
Boilerexams *May 2025 - Current*

- Architected scalable backend systems handling 500+ concurrent users during finals week, implementing mutual exclusion algorithms to prevent race conditions and ensure data integrity.
- Rebuilt email system architecture using Go.Mail with goroutines for asynchronous processing, improving system reliability and reducing email delivery latency by 40%.
- Led migration from TypeScript to Golang, refactoring codebase for better performance and maintainability while updating Docker Compose configurations for seamless deployment.

Projects

LoL Press Rankings API — Golang, Gin, PostgreSQL, REST API, AWS *Github*

- Engineered high-performance REST API serving League of Legends ranking data with sub-100ms response times, implementing sophisticated caching strategies and database optimization techniques.
- Designed scalable distributed architecture using Golang and PostgreSQL with complex relational schemas, foreign key constraints, and composite indexes ensuring ACID compliance and data integrity.

UNIX Shell Implementation — C++, Lex/Yacc, Systems Programming *Systems Project*

- Implemented full-featured UNIX shell interpreter with advanced process management, I/O redirection, pipe handling, and background process execution using low-level system calls.
- Optimized memory management and process scheduling algorithms, achieving efficient resource utilization and robust error handling in multi-process environments.

LogGPT — Golang, Concurrent Processing, CLI, PostgreSQL *Development Tool*

- Built intelligent log processing system supporting JSON, CLF, and plaintext formats with automatic schema detection and PostgreSQL table generation for developer workflow optimization.
- Implemented concurrent file processing using goroutines and channels, enabling efficient parsing of large log files with real-time progress monitoring and error recovery.

C-to-x86-64 Compiler — C, Assembly, Lex/Yacc, Algorithm Design *Compiler Design Project*

- Designed and implemented complete C-to-x86-64 compiler achieving 100% pass rate on comprehensive test suites, featuring optimized register allocation algorithms and efficient code generation.
- Built comprehensive lexical analysis, parsing, and semantic analysis components with advanced error detection and recovery, supporting multiple data types and complex expressions.