





Jack Schedel

Developing modern and efficient software

-  <https://schedel.io>
-  jack@schedel.io
-  [github/jackschedel](https://github.com/jackschedel)
-  [in/schedel](https://in.linkedin.com/in/schedel)

Education

Bachelor of Science - Exp. Dec 2024

Computer Science
University of Florida
3.49 GPA

Coursework

- Digital Logic and Computer Systems
- Microprocessor Applications
- Real Time Operating Systems
- Programming Language Concepts
- Algorithm Abstraction and Design
- Advanced Systems Programming
- Human-Computer Interaction
- Bioinformatic Algorithms
- Distributed Operating Systems

Skills

- Go
- Typescript / Next.js
- Rust
- C / C++
- React / Tailwind CSS
- Drizzle ORM / Sequelize
- Neovim / Lua
- Python / Jupyter Notebooks
- Embedded STM32 / MSP430
- Postgres / MySQL
- AWS / Azure / Vercel
- C# / .NET
- Java

Work Experience

Strategy Lead, FSGP 2023 Champions, Solar Gators **Spring 2022 - Present**

- Developed real-time telemetry data analysis tools for our team's solar-powered racecar to run live regression calculations in the racetrack pit using TypeScript.
- Created race physics simulation CLI tool using Go, used in conjunction with a Python optimization solver using the mystic library to run a gradient descent optimization to optimize energy efficiency and find the ideal driving strategy for the complex 11-turn track.
- Used the analysis tools to manage and determine team race strategy during the 2023 Formula Sun Grand Prix, in which our team won the overall competition.

Azure Software Engineering Intern, Microsoft **Summer 2024**

- Wrote C# code for a business-to-business Azure .NET service which manages employees with automated workflows which run tasks upon employee change.
- Implemented a feature to facilitate the transfer of owned groups upon employee leave, which allows the manager to select a new owner.

Embedded Systems Intern, Oak Ridge National Laboratory **Summer 2023**

- Wrote C and C++ code deployed to MSP430 chips on custom PCBs for a localized tracking system designed for use with containers for fissile nuclear material.
- Used light propagation latency from ultra-wideband communications between container devices and anchor devices to find one-dimensional distance.
- Implemented multilateration to determine container location using calculated distance information between the container device and multiple anchor devices with known exact placements around a target area.

Projects

AutoCalibr, Generative Autoencoder for 3D Meshes **Summer 2023 - Present**

- Created and trained a custom variational autoencoder model using Keras to generate new meshes using principle component analysis in the latent space.
- Handled comprehensive preprocessing of raw mesh data including face tri conversion, volume/positioning normalization, and face subdivision padding.
- Generated thousands of different random variations of each object during normalization and padding to artificially expand the dataset and enhance the model's comprehension of the intrinsic properties of the mesh data.

KoalaClient, OpenAI API Client Interface **Summer 2023**

- Developed an advanced interface for the OpenAI API using TypeScript and Tailwind CSS to better integrate LLMs into my personal workflow.
- Added OpenAI Whisper speech transcription, and a searchable prompt pallet to insert frequently used prompts from a custom user-defined library.
- Implemented tweakable model configuration overrides, unrestricted conversation history editing, and syncing across devices via Google Drive. Deployed the project to a website (client.koaladev.io) and Electron desktop app.

EndGame2, UCI Chess Engine **Summer 2023**

- Developed a UCI-compatible chess engine written in Rust that uses minimax tree-traversal over all possible board states using a custom board-state evaluation function.
- Implemented alpha-beta pruning, zobrist position hashing, and capture-resolve quiescence searching to improve search times and engine performance.

RTOS Tetris, Custom RTOS for Tiva C Series **Fall 2023**

- Developed a UCI-compatible chess engine written in Rust that uses minimax tree-traversal over all possible board states using a custom board-state evaluation function.
- Heavily optimized the thread context switching and minimizing unnecessary redraws, resulting in a snappy gameplay experience, despite only running on a 16MHz processor