Derrick Lee

CONTACT INFO

dlee3@scu.edu — 408-823-7288 — github.com/drklee3 — linkedin.com/in/drklee — dlee.dev

EDUCATION

B.S., Computer Science Santa Clara University **December 2019**

Relevant Coursework:

- Object Oriented Programming (C++)
- Data Structures (C++)
- Operating Systems (C/Rust)
- Theory of Automata and Formal Languages
- Computer Networks (C)
- Database Systems (Oracle SQL/PHP)
- Theory of Algorithms
- Programming Languages (Python/Java/Scala)
- Computer Security
- Design Management of Software

EXPERIENCE

Celo, Software Engineering Intern **June 2019 - September 2019**

- On the applications team primarily working with **TypeScript**, **React Native**, and **Redux**.
- Implemented social backup and recovery in the mobile wallet to keep user mnemonic seed phrases safe with the help of other users. Provides users with an option to split their mnemonic phrase to keep safe with friends.

Open Source Developer **January 2018 - Present**

- Fixed incomplete features, improved and clarified code examples and documentation for **serenity**, a **Rust** library for interacting with the chat and VoIP platform Discord's API.

RELEVANT SKILLS

Languages

- TypeScript, JavaScript, Rust, C, C++, Python, HTML5, CSS, PHP, SQL, Bash

Related Technologies

- Git, React, React Native, Redux, Redux Saga, Node.js, GraphQL, PostgreSQL

PROJECTS

sushii-bot (14,000+ lines of code) **December 2017 – January 2019**

- Chat bot for Discord with a ranking system, activity tracker, moderation tools and more with **over 64,000 total users**.
- Written in **Rust** with a **PostgreSQL** database, **diesel-rs**, and connection pooling with **r2d2-diesel**.
- Uses a **TypeScript** web server with **Koa** and **Puppeteer** to generate images from HTML.
- Paired a website with user leaderboards and statistics made with **Node.js**, **Next.js**, **React**, **Koa**, **Apollo** server and client for **GraphQL** endpoints, and **Join Monster** for batch data fetching.

Operating System Simulations (5,000+ lines of code) **April 2018 – June 2018**

- Runs sequential and random disk reads with **C**, determines time differences and possible causes based on both physical and OS aspects. Programs and set up executed with **Bash** scripts.
- Multi threaded simulation written in **Rust** of different memory page replacement algorithms with given page requests and a range of memory sizes. Data visualized with plots made in **R**.
- Benchmarks in **Rust** to determine the overhead of synchronization primitives (Mutex) and lock contention.