

# Derrick Lee

derrick@dlee.dev // (408) 823-7288  
dlee.dev // github.com/drkle3

## EDUCATION

---

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

**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.
- › Collaborated with a distributed team, locally in San Francisco and remotely in Berlin.
- › 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.

## RELEVANT SKILLS

---

### Languages

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

### Related Technologies

- › Git, Docker, React, React Native, Redux, Node.js, GraphQL, PostgreSQL

## PROJECTS

---

**picatch** (2,000+ lines of code)

**December 2019 – Present**

- › Self hosted photo gallery based on existing directory file structure.
- › Uses a Rust backend API server with actix-web and a frontend written in TypeScript with React.
- › Deployed with GitHub Actions and Docker.

**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, and GraphQL with Apollo.

**Operating System Simulations** (5,000+ lines of code)

**April 2018 – June 2018**

- › Performance tests to determine differences between sequential and random disk access with C, based on both physical and OS aspects. Programs and set up executed with Bash scripts.
- › Multi threaded simulation written in Rust to test different memory page replacement algorithms and memory sizes with given sequence of page requests. Data visualized with R.
- › Benchmarks in Rust to determine the overhead of synchronization primitives and lock contention.