Curtis Fenner

curtisfenner.com/resume
github.com/CurtisFenner
curtiswilliamfenner@gmail.com

Education

University of Michigan College of Engineering

Ann Arbor, MI. 2014 – 2018

- 4.0 GPA. Computer Science B.S.E. with minor in Mathematics.
- **Selected Coursework**: Distributed Systems (W2017), Grad. Programming Languages (F2017)
- Teaching assistant for Distributed Systems (F2017)
- World Finalist in 2017 ACM International Collegiate Programming Contest (ICPC)

Experience

Square — Software Engineer, Orders API Team

Atlanta, GA. Aug. 2018–present

- Backend engineer on the Orders API team, which operates both a public REST API, as well as internal systems that serve Square Point-of-Sale mobile apps
- Responsible for designing, reviewing, and implementing features that integrate with many other microservices including for payments, catalog, customers, and fulfillments
- Work emphasizes designs that are maintainable, scalable, strongly consistent, and highly available
- **Technologies**:
  - Java – for API server implementation
  - Protocol Buffers – for RPCs, modeling the API schema, and database serialization
  - TypeScript & JavaScript – for building internal web interfaces
  - MySQL – for durable, distributed storage enabling a strongly consistent API experience
- Redesigned complex request validation logic to significantly improve test coverage and code quality
- Created library functions to support old API versions with minimal ongoing maintanenace burden
- Drafted new documentation & a "mini-lecture" presentation to document complex cart-calculation logic

Qualtrics — Software Engineer Intern, Data Platform Team

Seattle, WA. Summer 2017

- Wrote code in a data-aggregation service as a member of the data platform team, and set up Elasticsearch + Logstash + Kibana for internal log analysis
- **Technologies**:
  - Scala – for backend implementation
  - Elasticsearch – for aggregating metrics to produce custom reports, and for log analysis
- Redesigned a data aggregation feature to get correct weighting across different displays
- Prototype the use of Elasticsearch for log management and indexing additional response information

Qumulo — Software Engineer Intern, Filesystem Performance Team

Seattle, WA. Summer 2016

- Wrote code and tests as a member of the performance Scrum team for a distributed filesystem server
- **Technologies**:
  - C – for filesystem implementation
  - Python – for integration-test automation and code generation
- Developed sharding of deleted file space reclamation to double free-space reclaimation rate
- Eliminated lock contention in a multithreaded cache to reduce file operation latency
- Implemented disk block allocation changes to ensure significantly faster metadata operations

Square — Software Engineer Intern, Public API Team

San Francisco, CA. Summer 2015

- Wrote Go and JavaScript (Node.JS) as member of public API team
- **Technologies**:
  - Node.JS – for implementation of microservice serving public API
  - Go – for implementation of a new microservice to eventually replace the Node.JS server
- Optimized and refactored public API server to halve average query time
- Ported a significant amount of the Node.JS implementation to Go in anticipation of Square's V2 APIs

Updated 8 November 2020.

Skills

- **Programming Languages**:
  - Java
  - Python
  - Scala
  - C
  - C++
  - JavaScript & TypeScript
  - HTML & CSS
  - Lua
  - Go
- **Technologies**:
  - MySQL
  - Elasticsearch
  - Git
  - React
  - Protocol Buffers

Projects

WeBWorK Proof Checker

- Designed and built a homework interface for writing and checking simple natural-deduction based proofs for students learning logic.
- Built a prototype implementation in JavaScript
- Translated the prototype into a Perl library that can be instantiated as a WeBWorK problem type
- Work included designing and creating
  - a logical formula parser
  - a tree-based symbolic pattern matcher
  - built-in logical deduction rules
  - a wrapping library to enable professors with limited programming experience to set up problems
  - a simple table-based user interface that could be rendered in WeBWorK
- https://curtisfenner.com/prove

Smol Programming Language & Compiler

- Designed and implemented a toy programming language and compiler that uses a rudimentary SMT solver to check assertions at compile time
- Work included
  - a PEG parser library
  - a type-checker that supports constrained type parameters
  - a code-generator that produces C99 code
  - a rudimentary CDCL based SMT solver, which can "verify" whether simple mathematical statements are true or false (Smol supported uninterpreted functions with quantifiers)
- https://github.com/CurtisFenner/smol-builder

Updated 8 November 2020.