J. Seth Daniel

job@sethdaniel.org

https://www.sethdaniel.dev/

https://github.com/WhoIsSethDaniel

214-886-3770

**SUMMARY**

 Graduate of Texas A & M University, with over two decades of experience in

 software engineering, seeking full-time software engineering position. Strong

 background with many variants of UNIX, problem solving, software development,

 and troubleshooting. Experience with web services and APIs, microservices,

 networked applications, database driven applications, multi-process

 applications, web development, and tool development.

**SKILLS**

 *Operating Systems*: Linux (Centos, Redhat, Debian, Ubuntu)

 *Programming*: Go, web-based APIs, web services, micro services, Perl, Python

 *Databases*: MySQL, PostgreSQL, Riak, ElasticSearch

 *SCM*: Git (GitHub and GitLab)

 *Observability*: Grafana, Graphite, Prometheus, Icinga, Icinga 2, Collectd

 *Other*: Docker, GoCD, Jenkins, AWS, Agile (sprints, planning, etc...)

**EXPERIENCE**

*ConnectWise*

 March 2021 – September 2021

 Senior Software Developer

 Working with a small team on the ConnectWise client and two services that the

 client talks to. The client runs on many different types of hosts (Windows,

 Mac, Linux), and records events such as the installation of software or a

 change in the amount of available memory. This information is sent to upstream

 services and presented to users. Another service manages profiles for clients

 that allows users to install / uninstall software, and other management tasks,

 across a large number of clients, remotely.

 The primary mechanism of communication between all the services is Kafka with

 some direct messaging via HTTP. Data storage is provided by Cassandra. The

 storage of the client is done in Amazon S3. Access to the client is done via+

 a set of AWS Gateway endpoints. All services run in AWS. The client and all

 services are written in Go.

*Silicon Shuttlecock*

 Senior Software Developer

 October 2019 - April 2020 (company folded)

 Worked on services, written in Go, that were meant to be used for the tracking

 and management of silicon wafers during the assembly of semiconductors. The

 services provided an API to query and manipulate the metadata for individual

 wafers as well as groups of wafers known as 'lots'. The backend datastore was

 SQLite.

 Unfortunately the software did not sell and the company folded in April 2020.

 *Xome*

Senior Software Developer

 October 2018 - July 2019

 Joined a small team of developers at Xome. Worked on breaking apart a small

 monolithic application into smaller services. Each service had a Postgres backend

 to store (mostly) JSON data using the JSONB column type in Postgres. Each

 service exposed a REST interface and all intra-service communication was done via

 REST. All code ran within Docker containers on locally managed hardware. All new

 services were written in Go.

 Maintained an accounting service of primary interest to the accounting group

 within Xome. Worked closely with the accountants when adding new features, and

 made all technical decisions for the team. Was able to convert multiple by-hand

 projects into automated projects to the great relief of the accountants.

 Was a member of a team that worked on rapid deployment of each of our new services.

 We used Jenkins as our CD platform and the end result was deployment to multiple

 hosts running docker containers. We reached our goal of under 10 seconds per

 deployment allowing very quick iteration for all developers.

 *OpenX*

 Senior Software Developer

 February 2010 - September 2018

 Was the founding member and team lead of the Automation Engineering and

 Observability team at OpenX. This team was directly responsible for a number

 of things including a system that tracked all of the hardware in each

 datacenter and what role each piece of hardware was meant to fulfill, along

 with monitoring, alerting, graphing, imaging, and central tooling.

 Wrote and maintained the system that helped OpenX organize and administer its

 data centers (several data centers worldwide with nearly twenty thousand

 servers). This system was used to track machines, network switches, IPs, PDUs,

 and many other items in each data center. Among its many functions this system

 wrote configuration files for various monitoring systems, controlled which

 users could access which hosts, and re-provisioned hosts.

 At the core of the data center management system was a web-based, RESTful API

 that allowed for things such as creating virtual machines, allocating physical

 machines, rebooting hosts, and creating new images for our machines. This API

 was backed by a dozen different services each using its own backend (primarily

 Postgres). Long running, asynchronous jobs were delegated to a job queuing

 system.

 Designed and performed initial development on a monitoring system that tied

 together Grafana, Graphite, Icinga 2, the data center management system, and

 an in house metric collection system. This monitoring system simplified and

 improved the way monitoring and alerting of metrics occurred at OpenX.

 All of the services my team was responsible for were tested and deployed via

 our CI/CD system (GoCD at first, followed by Jenkins). We were the first team

 at OpenX to use continuous deployment and we helped other teams that wished to

 move toward CD (be it continuous delivery or continuous deployment).

 Testing (unit, functional, operational) is very important. The codebase had

 well over 13,000 tests. Most of the code had well over 90% test coverage.

 *Oversee.net*

 Senior Software Developer

 June 2008 - February 2010

 Primary developer on next generation domain ad serving platform. This included

 a custom web server frontend and many backend services. The entire system was

 asynchronous. The backend services were all built from a common server

 framework. I wrote the web server frontend, the common server framework, and a

 number of the backend services.

 Wrote a packaging and release system built on top of Jenkins. Using

 Jenkins we could package up our code (using RPM) and release it to QA.

 After validation by QA the code could be released to production using the

 same mechanism.

**OPEN SOURCE**

 *Goldsmith* (https://github.com/WhoIsSethDaniel/goldsmith.nvim)

 This is a Neovim plugin I have developed. I use it every day when I am working

 on Go code. It takes advantage of many new features that Neovim provides as well

 as the tools provided by the Go distribution itself.

 *toggle-lsp-diagnostics* (https://github.com/WhoIsSethDaniel/toggle-lsp-diagnostics.nvim)

 Another Neovim plugin I have developed. This one is much simpler than Goldsmith

 and it does exactly one thing: toggle the display of diagnostics from external

 LSP servers.

**EDUCATION**

 Texas A&M University

 Bachelor of Computer Science, 1996