Justin Spencer

Ashburn, VA • jspencer.jms@gmail.com • jamsupreme.net

Education

Neumont University - https://neumont.edu
Bachelor of Science in Computer Science, 2009

Certifications

- AWS Solutions Architect (Professional) 2021-Present
- AWS DevOps Engineer (Professional) 2021-Present

Skills

Languages
Ruby - C# - Java - Javascript (JS) / Node - Terraform (HCL) - YAML (k8s/CloudFormation) - Bash - Some Python & R

Web
"Serverless" APIs - Ruby on Rails - OpenAPI - JSON Schema - React - GraphQL - HTML & CSS & JS & AJAX - React - Angular - RESTful APIs - Microservices - NodeJS Express - Java Spring - ASP.Net MVC

Middle Tier
Package management (Nuget, Rubygems, NPM, Gradle/Maven) - ORM Frameworks (.NET EF/ADO, ActiveRecord, Hibernate, Sequelize) - IOC containers (Autofac, Spring) - Pub/Sub (Kafka, SQS/SNS, Sidekiq) - Bouncy Castle (PKI FIPS encryption)

Databases
mySQL - Aurora RDS - MS SQL - Redis - PostgreSQL - SQLite - DynamoDb - MongoDB - Liquibase - DBUp - Flyway

Infrastructure
Jenkins - AWS CodePipeline suite - Github Actions - Bamboo - Octopus Deploy - Cruise Control - NAnt - Continuous Integration (CI) - Continuous Deployment (CD) - Pivotal Cloud Foundry (PCF) - Kubernetes (k8s) & Helm - Powershell - Chef - Ansible - AWS Security Hub

Theory
Fault Tolerance - Metaprogramming (C# and Ruby) - Design patterns - Agile methodologies (Scrum, Kanban) - Localization / Internationalization - Async code - Concurrent code - Domain Specific Languages (DSLs) - Functional Programming (Elixir/Erlang/JS Ramda) - Contract Testing (Pact) - Test Doubles / "Fakes" best practices for APIs - High Availability Strategies & various load balancers - Feature toggles (flags) - Domain-Driven Design - Evolvable APIs

Monitoring / Logging
DataDog - NewRelic - Splunk - AWS X-Ray & CloudWatch - k8s dashboard - Rollbar - Raygun - LeanSentry

Tools / Other
Git - JIRA - Confluence - Twistlock - Section 508 & ARIA accessibility - Nexus Repository - OneArtifactory - Camunda Rules engine - Linting tools (ESLint, TFlint, rubocop) - Github/PR Best Practices - Subversion - TFS - Social APIs (FB, Twitter)

Employment Summary

• ICF	December 2019 - Present	https://www.icf.com/
• Verizon	June 2019 - December 2019	https://www.verizon.com
• Excella Consulting	March 2016 - June 2019	https://www.excella.com/
• Personica	August 2014 - March 2016	https://www.personica.com/
• REI Systems	July 2009 - August 2014	https://www.reisystems.com/

Work Experience

ICF

Full Stack Developer / DevSecOps / Cloud Engineer / Solution Architect
Project: Several
Team Size: Various

MilitaryChildCare.com - 2026 - I was brought onto the project to resolve a number of growing pains the system was experiencing spread across two of its applications: MilitaryChildCare (MCC) and Fee Assistance Management System

Justin Spencer

Ashburn, VA • jspencer.jms@gmail.com • jamsupreme.net

(FAMS). The MCC application made heavy use of non-performant stored procedures that included elaborate business logic spanning several use cases that piled up in the database rather than application code. These were resolved with a mix of rewrites and new indexes. For FAMS, the C# IQueryable was leveraged to prevent unnecessary scans and minimize result sets while valid asset caching was added to prevent excess load on existing noisy IAM middleware. On top of dramatically improving application latency and uptime, I was involved in additional initiatives to add MFA, extract right-sized microservices, and plan for cloud migration.

OASH IDP Cloud migration - 2022-2026 - Implemented a substantial cloud migration involving several tightly coupled OASH services into AWS GovCloud. This involved some traditional "lift and shift" (or rehost) as well as replatform strategies to safely move the entire stack into AWS GovCloud without disruption. Everything in the cloud was provisioned with Infrastructure As Code via Terraform and a dedicated CI/CD pipeline to avoid drift. As some integrations still exist in on-prem networks, site-to-site VPN was established to their Trusted Internet Connection (TIC) to support a hybrid network. A mix of Network Load Balancers (NLBs) and Application Load Balancers (ALBs) were provisioned for different use cases to support API communication with internal APIs without internet access, SFTP file exchanges, and interfacing with legacy applications that remain on-prem. The existing architecture of manually maintained physical servers were migrated to EC2s in Autoscaling Groups with automated configuration management to guarantee resilience and provide high availability wherever possible. Security Hub was set up to consolidate data from several security services such as Config, Inspector, GuardDuty, CloudWatch, and CloudTrail and achieve 100% compliance. This assisted in making for a streamlined ATO process despite the FedRamp Moderate classification. The entire fleet of 50+ servers across all environments was kept hardened by routine AMI updates, custom configuration management, and automated Patch Manager Policies. In addition to traditional servers, API Gateways, lambdas, and containers were provisioned and included in the infrastructure to support gradually replacing the older services with more scalable and cost effective replacements.

National Child Welfare Data Management System (NCWDMS) - 2020-2024 - https://ncwdms.acf.hhs.gov/ - For this system we had a UI in which State/Tribe/Territory (STT) users could log into a React app hosted by Cloudfront and upload XML files containing information according to the relevant AFCARS Technical Bulletins, such as Out of home care, adoption assistance, and family first.
The entire system was "serverless" and made heavy usage of Lambda, SQS, EventBridge, and API Gateway to allow inter-service communication. Terraform and CodeBuild were used to enable CI/CD for both infrastructure resources and the app code itself. The project also required an extensive rules library with an exhaustive test suite to ensure the many different permutations of 180+ XML elements would produce the desired output reports for end users. The comprehensive test suite tested thousands of data permutations to validate that different rule results correctly interacted with one another, resulting in hundreds of thousands of rule executions per full test suite run.

I also created several sandbox projects that could be leveraged in tech challenges or incorporate piecemeal into existing projects. This involved patterns for reusable CloudFormation templates, self-documenting Jenkins libraries, and various bootstrap and teardown utilities for infrastructure.

Verizon

MTS 4 (Architect, Interim Manager)
Project: TPVM (https://www22.verizon.com/wholesale/business/Doing-Business.html)
Team Size: 54

- Custodian / Interim Project Manager of "TPVM" system after mass exodus of critical personnel. It was composed of **90+** microservices and some additional on-premise applications, some of which were in mainframes.
- Auditing for California Consumer Privacy Act (CCPA)
- Managing contractors and job postings, Allocating project funding, Managing sprint priorities, communicating with upper management (directors and executive directors), Coordinating work between on-shore and off-shore.
- Migration of many microservices concurrently from Pivotal Cloud Foundry (on-prem) to Kubernetes (AWS hosted) - This entailed networking (firewall) changes, some code changes for poorly-performing apps, meeting new security requirements (Twistlock), and adding regression suites when possible.
- Introduced best practices for linting code, managing git branches, adding app health checks, and improving documentation by an order of magnitude

Excella Consulting

Lead Consultant (Tech Lead / Supervisor)
Project: https://my.uscis.gov/
Team Size: 4 per "pod", 8 per team, 24 on the web project (excluding non-developers)

- Developing strategies to ensure that the entirety of the USCIS ecosystem (identity providers, user profile storage, adjudication systems, secure file storage, and data governance) could work in concert, such as: pub/sub implementation (Kafka), data integrity (JSON schema), and handling downtime or incremental releases with feature toggles.
- Ruby gem overhaul - The gems were originally file based, which resulted in hidden dependencies, circular dependencies, unexpected behavior, and inaccurate gemspecs. We updated our pipelines to build the gems and correctly version them.
- myUSCIS tech challenge - I was involved in the "tech challenge" phase of the myUSCIS re-compete proposal effort. We had set up a number of tools beforehand and worked together to effectively simulate an agile project.
- Rolling out several "eProcessing" forms - We originally built the N-400 and since evolved the system to support

Justin Spencer

Ashburn, VA • jspencer.jms@gmail.com • jamsupreme.net

several types of other benefits, including some that work in concert with other benefits (e.g. the G-28)
- Evolving "eProcessing" across systems - The ecosystem for eProcessing evolved as we added or removed microservices and swapped different data stores. In some cases this would also involve data that needed to be re-structured and/or duplicated in multiple places while maintaining consistency. This was done seamlessly on many different occasions.
- JSON Schema adoption - Prior to introducing JSON schema, different systems had different copies of excel sheets with differing values. In some cases, the APIs or databases even drifted from their documentation. To establish a consistent source of truth we built a JSON schema API which functioned as a medium for discussing form content, a basis for consistent Kafka messages, and later as a tool to greatly speed up form development.

Personica (Formerly Fishbowl, Inc)

Sr. Software Engineer

My work at Fishbowl was often variegated, changing depending on the needs of clients and the availability of resources that could develop the software necessary to meet those needs. Given my versatility as a full stack developer, my specific responsibilities were generally in flux from one project to the next. I'll attempt to illustrate some significant accomplishments among those projects.

- CI / CD Overhaul - Fishbowl originally had CruiseControl with NAnt to build and deploy products. It was an error-prone multi-hour affair that blocked builds for ALL products. I overhauled everything into Bamboo with Octopus Deploy so everybody could enjoy deploys in minutes.
- SSO Consolidation - Fishbowl acquired another company and eventually needed to consolidate the SSO systems to provide a seamless integration across its product suites.
- API-Based UI re-design - We had to re-design one of the product UIs. To improve testability and allow for other systems to use the data, we built a new API. I also built some IQueryable helpers to enable simple full-text searching.
- Segmentation UI widget & API - We had a new "Segmentation" API that could allow users to appropriately target their campaigns against a desired audience. We built some angular widgets with async (cancellable) calls to the API so that the expected audience could be updated real-time.
- Social Media Publishing - We allowed users to publish mailing content as social media posts across multiple platforms. This worked across multiple mailing system products and also had some workflow support.

REI Systems

Sr. Software Engineer

Much of my time at REI was spent on projects within the Health Resources and Services Administration (HRSA) program. Chief among the HRSA products is the Enterprise Handbooks (EHB) which is the nickname given to the web site in which much of their work is done. Most of my work involved transforming complex business rules into well-designed systems.

Some noteworthy projects:
- Audit Tracking Component - A cross-cutting concern that could apply to any data model and easily audit data changes
- HIV/AIDS Bureau (HAB) Formula - Retrieve data from several data points (excel, FTP files, etc.) and appropriately calculating funding
- HRSA EHB Funding Memo Re-Design - An overhaul of the existing system that built up funding memos. It involved several complex patterns with which funding could be supplied to grantees.
- HRSA EHB Post Award (v1 and v2) - I designed the initial system that monitored grantees post-award. There were in fact two slightly different systems, one for grantees, and another for reviewers, but both shared similar architectures and workflows.
- HRSA EHB Awards Re-Design - This was a team of 10 developers re-designing one of the most critical parts of the EHB system for awarding grantees money. I worked on the "funding information" module which had a somewhat complex interface involving several calculations involving multiple data sources.