Neil R. Simon

nsimon@solidcode.com | +1 (303) 402-9500 | LinkedIn | GitHub

Available for full-time and contract work in cloud automation and DevOps.

Professional Summary

Experienced DevOps Engineer specializing in Azure, Kubernetes, Docker, and Bash scripting. Adept at designing scalable cloud infrastructure, optimizing workflows, and automating deployments to enhance efficiency and reliability. Passionate about automation, security, and maintainable code in cloud environments.

Technical Skills

- Azure — Key Vaults, Storage Accounts, Blobs, Files, Resource Tags
- Kubernetes — kubectl, Terraform (basic), scripted pod access
- Docker & Containerization — Docker, Azure Container Registry (ACR)
- CI/CD & Version Control — GitHub, GitHub Actions
- Secrets Management — Azure Key Vault, SAS keys, tokens, policies, permissions
- Scripting & Automation — Bash, Python, Azure CLI, GitHub CLI, Kubernetes automation
- Azure PostgreSQL — pg_dump, pg_restore, psql scripting, management, reporting

Professional Experience

Senior Data/Solution Engineer
Integral Consulting Inc. | Nov 2011 – Present
(Cloud automation and infrastructure focus since 2021)

- Automated Azure infrastructure using Bash-based scripts for streamlined cloud operations.
- Migrated 100+ PostgreSQL production databases from legacy servers to newer infrastructure with minimal downtime.
- Managed Azure Kubernetes Service (AKS) deployments, improving efficiency and uptime.
- Optimized PostgreSQL database operations, including automated backup, restore, and reporting.
- Enhanced container workflows by integrating Docker and ACR for seamless image storage and retrieval.

Previous Experience
20+ years in software engineering and Linux server management:

- Provisioned and managed dozens of Linux production cloud servers on Citrix and DigitalOcean over 15 years.
- API development & network management tools.
- Test automation & failure diagnosis.
- Cross-platform development & workflow optimization.
- Team collaboration & technical leadership.

Core Principles

- Naming standards matter – Precision in naming ensures clarity and maintainability.
- Refactoring is key – Making code clearer, more efficient, and future-proof.
- Readability and scalability – Code should be intuitive for both humans and automation.
- First, make it work. Then, make it work better.

Education

B.S. in Computer Science
Stephen F. Austin State University