STEVEN ONOJA

DevOps Engineer · Cloud Infrastructure · Full-Stack Development
✉ onojasteven13@gmail.com | 🌐 stevenincloud.online | LinkedIn: linkedin.com/in/steven-onoja-95668a225 | 📍 Manchester, UK

PROFESSIONAL SUMMARY

I enjoy building systems that make life easier for both engineers and users. Over the years, I've worked on improving deployment processes, automating repetitive tasks, and fixing the kinds of problems that slow teams down or create unnecessary stress. One of the projects I'm most proud of involves helping reduce deployment time from four hours to under one hour, while another cut delivery lead time by 73% through better tooling and workflow improvements. My background spans DevOps, cloud infrastructure, and full-stack development, but what matters most to me is building reliable systems that people can depend on.

CORE SKILLS

- Problem-solving, Clear communication, Time management, Leadership & mentoring, Adaptability, Organization, Attention to detail, Collaborative working

TECHNICAL SKILLS

- AWS (EC2, VPC, S3, RDS, Lambda, IAM, CloudWatch) · Terraform · Jenkins · GitHub Actions · Docker · Kubernetes · Ansible · Prometheus · Grafana · Python · Bash · JavaScript/TypeScript · React · Next.js · Node.js · REST APIs · MySQL · PostgreSQL · MongoDB · Git · Nginx · JIRA · Confluence

WORK EXPERIENCE

DevOps Engineer · Cloudhight Consulting

Dublin, Ireland | March 2025 – Present

- I joined a 10-person DevOps team supporting large AWS environments and quickly noticed that deployments were taking far longer than they needed to. I rebuilt parts of the Jenkins CI/CD workflow and streamlined release steps, reducing the average deployment time from roughly 4 hours to under 1 hour.

- One of the bigger challenges we faced was a delivery pipeline that had gradually become difficult to maintain. We redesigned parts of the API layer and modernised the tooling around deployments and testing. That work reduced overall delivery lead time by 73% and was estimated to save the client around $800K annually.

- Managing infrastructure manually was creating small but constant operational issues, especially when Auto Scaling introduced new instances. I wrote Bash scripts that generated Ansible inventory files dynamically from scaling events, so new servers registered automatically instead of relying on engineers to update inventories by hand.

- I used Ansible to standardise updates and configuration management across more than 100 servers. Before that, configuration drift kept causing inconsistent behaviour between environments. Once we cleaned that up, customer-reported incidents dropped by around 40%.

- I helped maintain Kubernetes clusters running microservices workloads in production. A lot of the work wasn't glamorous — tuning health checks, improving node reliability, and troubleshooting networking issues — but those changes made deployments far more stable over time.

- Security reviews were often happening too late in the release cycle, so I worked with the team to integrate Trivy, Checkov, and SonarQube directly into the pipeline. Catching vulnerabilities earlier reduced rework and helped us avoid last-minute deployment blockers.

- I introduced New Relic monitoring and alerting for infrastructure and JVM performance after we realised engineers were often reacting to incidents only after users reported them. Better visibility meant we could catch resource spikes and service degradation much earlier.

- I automated AWS provisioning with Terraform modules covering VPCs, EC2, RDS, load balancers, and Auto Scaling resources. I also worked closely with architects to keep infrastructure diagrams and documentation current so newer team members could ramp up faster.

- Outside the technical work, I spent a lot of time keeping runbooks and Confluence documentation updated because I've seen how quickly undocumented systems become difficult to support during incidents.

Applications Developer · RUSUL Ltd

Abuja, Nigeria | Jan 2023 – March 2025

- I rebuilt parts of the company's customer-facing website using Next.js and React, focusing heavily on responsiveness and usability. A big priority was making the platform work properly on slower connections and lower-end mobile devices because a large part of the user base depended on them.

- I integrated REST APIs across the frontend and backend and spent a lot of time improving how authentication, loading states, and errors were handled. Small UX details ended up reducing user confusion and support requests more than expected.

- I developed a React Native streaming application backed by Appwrite, including support for real-time updates and more graceful offline behaviour. The goal wasn't just adding features — it was making the app feel reliable even when connectivity wasn't perfect.

- Alongside development work, I managed day-to-day IT operations and internal support requests. That experience taught me how much engineering work improves when communication and responsiveness are treated seriously, not as secondary tasks.

- I documented internal audit and reporting workflows because teams were processing data inconsistently across departments. Having clearer documentation reduced confusion and made reporting much more predictable.

- I also worked on digital assets and user-facing materials with UX principles in mind, helping improve how the organisation communicated online.

Junior Web Developer · NNPCL

Abuja, Nigeria | April 2021 – Dec 2022

- I built responsive frontend components with React, HTML5, and CSS3 while integrating third-party APIs into existing systems. A lot of the work involved collaborating closely with designers to make sure interfaces behaved consistently across devices.

- Early on, I noticed parts of the application slowing down under heavier usage, so I spent time profiling and debugging performance bottlenecks in the codebase. That experience taught me the value of careful troubleshooting instead of rushing fixes.

- Working in Agile teams helped me understand how delivery planning affects engineering quality. Sprint planning and retrospectives gave me a much better appreciation for communication, prioritisation, and managing technical debt realistically.

- I reviewed and debugged Java applications connected to databases and APIs, extending existing functionality without disrupting systems already in use internally.

- I also contributed algorithmic improvements that reduced processing time in key workflows, which made some routine operations noticeably faster for users.

EDUCATION

MSc Computer Science & Technology with Business Development: Ulster University, Manchester, Aug 2025 – Oct 2026

BEng Computer Engineering: Afe Babalola University Sep 2018 – Jul 2023