# [First Last Name]

[City, State] | [email@example.com] | [+1 (555) 555-5555] | [LinkedIn URL] | [GitHub URL]

## PROFESSIONAL SUMMARY

[DevOps Engineer] with [X+] years of experience designing, automating, and optimizing cloud-native infrastructures and CI/CD pipelines across [AWS/Azure/GCP] environments.

Proven track record implementing **Infrastructure as Code**, observability, and security best practices to improve deployment reliability, scalability, and release velocity.

Skilled in bridging development and operations teams, reducing manual toil through **automation, containerization, and orchestration**, and enabling high-availability, fault-tolerant systems.

## PROFESSIONAL EXPERIENCE

### [Senior DevOps Engineer] | [Tech Company A]

[MM/YYYY] – Present | [City, State]

- Designed and implemented end-to-end **CI/CD pipelines** using [GitHub Actions/Jenkins/GitLab CI], reducing average deployment time by [40–60%] and enabling [multiple] production releases per day.
- Architected scalable, highly available infrastructure on [AWS/Azure/GCP] with **Infrastructure as Code** using [Terraform/CloudFormation], standardizing environment provisioning and cutting configuration drift incidents by [X]%
- Implemented containerization and orchestration with **Docker** and **Kubernetes** ([EKS/AKS/GKE]), optimizing resource utilization and improving application uptime to [99.9%+] through automated health checks and rolling updates.

### [DevOps Engineer] | [Software Company B]

[MM/YYYY] – [MM/YYYY] | [City, State]

- Developed automated build, test, and deployment workflows for microservices using [CI tool] and [artifact repository], increasing deployment success rate by [X]% and reducing rollbacks through standardized release processes.
- Set up centralized logging and monitoring with **Prometheus**, **Grafana**, and [ELK/EFK stack], establishing alerting thresholds that reduced mean time to detection (MTTD) and resolution (MTTR) for production issues.
- Collaborated with development teams to implement **configuration management** using [Ansible/Chef/Puppet], automating server provisioning, patching, and application configuration across [N+] environments.

## EDUCATION

### [Bachelor of Science in Computer Science] | [University Name]

[MM/YYYY] – [MM/YYYY] | [City, State]

[Relevant Coursework: Operating Systems, Distributed Systems, Cloud Computing, Networking, Software Engineering]

## SKILLS

**Cloud Platforms:** [AWS] · [Azure] · [Google Cloud Platform]

**DevOps & Automation:** [CI/CD (GitHub Actions, GitLab CI, Jenkins)] · [Infrastructure as Code (Terraform, CloudFormation)] · [Configuration Management (Ansible, Chef, Puppet)]

**Containers & Orchestration:** [Docker] · [Kubernetes] · [Helm] · [Container Registries (ECR, ACR, GCR)]

**Monitoring & Logging:** [Prometheus] · [Grafana] · [ELK/EFK Stack] · [CloudWatch/Stackdriver/Azure Monitor]

**Scripting & Programming:** [Python] · [Bash] · [PowerShell] · [Go/Other Language]

**Security & Reliability:** [Secrets Management (Vault/SSM)] · [IAM & RBAC] · [Backup/DR Strategies] · [SSL/TLS, Network Security Groups]

**Soft Skills:** Cross-functional collaboration · Incident response · Documentation · Agile/Scrum · Stakeholder communication

## PROJECTS

### [Cloud-Native CI/CD Pipeline Modernization] | [Personal/Company Project]

[MM/YYYY] – [MM/YYYY]

- Built a fully automated **GitOps-style** CI/CD pipeline using [GitHub Actions/GitLab CI] and [Argo CD/Flux], enabling declarative deployments to [Kubernetes] across staging and production clusters.

- Integrated automated unit, integration, and security scans ([SAST/DAST]) into the pipeline, enforcing quality gates and preventing vulnerable builds from reaching production.

**[Infrastructure as Code for Multi-Environment Platform] | [Personal/Company Project]**

[MM/YYYY] – [MM/YYYY]

- Developed reusable **Terraform** modules to provision VPCs, subnets, security groups, managed databases, and Kubernetes clusters across [dev/test/prod] environments.
- Implemented remote state management and environment-specific workspaces, enabling consistent, version-controlled infrastructure changes and peer-reviewed pull-request workflows.