Zakaria EL MOUMNAOUI

Casablanca, Morocco | zakariaelmoumnaoui96@gmail.com | +212 622906171
LinkedIn: zakaria el moumnaoui | GitHub: zakaria-statistics

DevOps and Cloud Engineer

Summary

DevOps engineer passionate about automation, orchestration, and continuous integration. Experienced in infrastructure management and CI/CD pipeline implementation.

Technical Skills

CI/CD and Automation: GitLab CI/CD, Jenkins, GitHub Actions, ArgoCD, Ansible, Terraform, Vagrant
Containerization and Orchestration: Docker, Containerd, Kubernetes, Docker Swarm
Cloud and Infrastructure: AWS, Azure, VMware ESXi, VirtualBox
Systems and Security: Linux, WSL, SSL/TLS, VPN, Firewalls, Nginx
Monitoring and Logging: Prometheus, Grafana, Loki
Languages and Scripting: Bash, PowerShell, Python, Java, TypeScript
Databases: MySQL, PostgreSQL, Firebase
AI and LLMOps: Ollama, LangChain, llms, Agents/Tools, Prompt engineering
Frameworks and Libraries: Spring Boot, Angular, React

Experience

Cloud DevOps Engineer at We Are Beebay (2024 - Present)

Python Automation & Reporting – Multi-Tool Security Aggregation 2025

Development of a full Python module to centralize and analyze CI security reports:

- Advanced parsing of **XML/JSON reports**: SpotBugs, PMD, Checkstyle, Dependency-Check, Gitleaks.
- Dynamic construction of **XML schemas** (streaming, `iterparse`, namespaces) to handle very large files.
- Normalization of findings (file, line, severity, rule, CVSS, CWE) and **global KPI computation** per tool.
- Full SonarQube API integration: `/api/issues/search`, `/api/hotspots/search`, `/api/qualitygates/project_status`, `/api/measures/component`.
- Multi-source aggregation: **merging vulnerabilities, hotspots, code smells, secrets, SCA results** into a uniform structure.
- Generation of an **automated executive report**: summary, status (HEALTHY / ATTENTION / BLOCKER), recommendations.
- LLM API integration (OpenAI-compatible) to enrich the report with insights and recommendations.
- Creation and email delivery of a consolidated **PDF report** (PDF + text content).
- Structured logging, modular architecture (*utils*, *details*, *sonar*, *mailer*, *llm_client*).

Self-Managed Kubernetes Platform on Azure secured with Vault (Terraform + Ansible) 2025

Step-by-step implementation of a solid foundation for web applications with isolated databases:

- **Networking**: Single VNet with 5 dedicated subnets — Sub1 (kube bastion), Sub2 (control-plane), Sub3 (workers), Sub4 (DB), Sub5 (DB bastion).
- **Access security**: Minimal NSG rules (SSH allowed only from our public IP to the bastions; minimal internal paths open only between required VMs).
- **Provisioning**: Infrastructure deployed via Terraform (network, public IPs for bastions, kube/DB VMs). Outputs validated (public/private IPs).
- **Secrets management**: Operational Vault integration (AppRole) to retrieve Azure secrets and drive Terraform through a wrapper (bash script).
- **Base hardening**: cloud-init applied (users/SSH keys, updates, utilities) on bastions, kube nodes and DB VMs.

- **K8s cluster bootstrap**: prerequisites via **cloud-init**, orchestration via **Ansible** (control-plane init, worker join, Calico CNI); `kubectl` access from the bastion.
- **Ansible — Caching**: shared facts cached (kubeconfig, `join_cmd`) via `jsonfile` backend, reusable across plays/runs.
- **Ansible — PostgreSQL (Sub4)**: install + listening, CIDR ACLs (`pg_hba.conf`), DB/user creation, logical backup (script+cron), smoke test (`SELECT 1`) — fully idempotent.
- **DB/K8s integration**: secrets delivered by Vault (AppRole), headless Services + Endpoints with private IPs, environment variables for apps.
- **Vault & AppRole (secure delivery)**: implementation of a secure SecretID delivery flow using response-wrapping and Vault Agent (init/sidecar) to generate DB credentials.
- **Dynamic DB credentials**: issuance of temporary credentials via `database/creds/<role>` (lease management, rotation/renewal, DB smoke tests).

Azure Backup & Recovery with Terraform & PostgreSQL 2025

Designed and executed a full-stack backup and recovery strategy on **Azure**, combining infrastructure-level snapshots with database-level Point-in-Time Recovery (PITR):

- Provisioned **Azure Linux VMs** and **managed disks** using **Terraform**, with explicit disk attachments (LUN0) for persistent storage.
- Implemented **disk snapshot lifecycle**: created incremental snapshots, restored them as new managed disks, and re-attached them to VMs for disaster recovery testing.
- Installed **PostgreSQL 17 (PGDG)** and prepared the database cluster on the persistent disk /**mnt**/**appdata**.
- Configured **WAL archiving** and performed **base backups** to enable precise database recovery.
- Simulated failure by wiping the data directory and performed **Point-in-Time Recovery (PITR)** using archived WAL segments and base backups.
- Validated recovery at both layers: infrastructure (disk replacement) and application (Postgres PITR), demonstrating end-to-end resiliency.
- Applied best practices for **Infrastructure as Code (IaC)**, snapshot cost-awareness, and clean resource lifecycle management with **Terraform**.

Azure Infrastructure Provisioning with Terraform 2025

Built and managed a modular cloud environment on **Azure** using **Terraform**:

- Designed and implemented two reusable modules: **network** (VNet, Subnet, NSG, NIC, Public IP) and **compute** (Linux VM).
- Provisioned **Linux VMs** with static Public IPs and cloud-init scripts to automatically install and configure **NGINX** at boot time.
- Scoped all resources within dedicated **Resource Group** for isolation and lifecycle management.
- Applied **environment-specific configurations** (**dev.tfvars**, **prod.tfvars**) to parameterize VM size, network CIDRs, disk size, and firewall rules.
- Implemented **Network Security Groups** with environment-aware rules (SSH-only in dev, SSH + HTTP/HTTPS in prod).
- Enforced **Infrastructure as Code (IaC)** practices for consistent, repeatable deployments and simplified environment switching.
- Introduced **cost awareness** by optimizing VM types, monitoring disk/IP charges, and enforcing budget limits.
- Secured sensitive credentials through **environment variables** and **tfvars** files.

CI/CD Pipelines with GitLab CI/CD, Spring Boot, Next.js, and PostgreSQL 2025

Designed and implemented an automated CI/CD pipeline using **GitLab CI/CD**:

- Automated backend build and test phases (**Spring Boot**) with **Maven**.
- Built and optimized Docker images for backend and frontend (**Next.js**).
- Automated deployment of applications and **PostgreSQL** database on a dedicated VM using Docker Compose.

Kubernetes & GitHub Actions: Local Deployment of an LLM Agent using Ollama 2025

Design and orchestration of a production-like AI agent platform:

- Stack: **FastAPI** backend with **SSE** streaming, **React** UI, **LangChain** + **Ollama (Mistral)** for local inference.

- **Kubernetes**: Deployments and Services (backend & frontend) via cluster & resource limits. *Result: stable, observable runtime.*
- **CI/CD to Kubernetes**: Automated GitHub workflow triggered on changes that builds and packages the application, publishes it, updates the running environment, and verifies the rollout. *Result: predictable, low-risk releases.*
- **GPU acceleration**: GPU-enabled inference using the **NVIDIA device plugin** on Kubernetes.

CI/CD and Deployment Pipelines with Jenkins and Kubernetes 2024

Designed and implemented two automated pipelines with **Jenkins** and **Kubernetes**:

- Set up dedicated VMs for each ecosystem tool (**Jenkins**, **SonarQube**, **ArgoCD**).
- Configured **NGINX** as a reverse proxy to secure and centralize access.
- Implemented **SSL/TLS encryption** to secure web traffic and protect sensitive data in transit.
- CI/CD pipeline: Automated builds and tests with **Maven**, code analysis with **SonarQube**, and Docker image generation.
- Deployment pipeline: Kubernetes manifest updates and deployment via **ArgoCD**.

R&D Projects

LLM Infrastructure on Azure with Terraform 2025

Design and operation of an LLM application on **Azure** with **Terraform** and GPU-enabled VMs:

- **RBAC**: creation of a **service principal** with least-privilege roles (`Contributor` at RG scope, `Reader` at subscription level for images); secure storage of credentials (tenant, subscription, client, secret) for the provider.
- **Networking & compute** via modules: **VNet**, subnet, **NSG**, public IP, NIC, **Key Pair** and **Linux VM**; all resources grouped in a dedicated **Resource Group** for isolation and lifecycle management.
- **cloud-init**: installation of **Docker** and deployment of **Open WebUI** (container) on first boot, with **dynamic variable injection** via `templatefile`; `systemd` service for automatic restarts.
- **DNS & TLS/HTTPS**: **A record** (e.g., `llm.infra-ia.com`) pointing to the **static public IP**; NSG rules opened for **80/443**; **Nginx** as reverse proxy (app on `127.0.0.1:8080`), `HTTP→HTTPS` redirection, **Let's Encrypt** certificates (webroot) with **auto-renewal** and Nginx reload.
- **GPU**: enablement on supported SKUs (e.g., `Standard_NC4as_T4_v3`), installation of NVIDIA drivers and `nvidia-container-toolkit`.
- **GPU quotas**: querying and increasing limits per family (e.g., *Standard NCASv3_T4 Family*) via `az rest` and **Support ticket**.
- **Multi-environment setup** with **tfvars** (`dev.tfvars`, `prod.tfvars`) for VM sizes, CIDR, disks, and NSG rules; full **IaC compliance** for reproducible deployments.
- **OpenAI integration** via `OPENAI_API_KEY` and `OPENAI_BASE_URL`; **local LLM** (**Ollama**) as fallback or to reduce external API costs.
- **Cost & security**: SKU optimization, monitoring public IPs/disks, budgets/alerts; hardened NSG rules (restricted SSH in dev, HTTP/HTTPS in prod).

LLM Infrastructure on AWS with Terraform 2025

Deployment of an LLM application on **AWS** with **Terraform**, supporting GPU-based options:

- **IAM**: creation of a **Terraform user/role** with minimal privileges; environment variables for authentication (no hardcoded keys).
- **Networking & compute** via **modules**: **VPC**, public subnet, **Internet Gateway**, **Route Table**, **Security Group** (SSH/HTTP), **Key Pair**, **EC2** and additional **EBS volume**; **Terraform HTTP probe** for availability checks.
- **cloud-init**: installation of **Docker** and deployment of **Open WebUI** at first boot, with **dynamic variable injection** via `templatefile`; `systemd` service for orchestration and restarts.
- **DNS & TLS/HTTPS**: **A record** (e.g., `llm.infra-ia.com`) pointing to an **Elastic IP**; SG rules opened for **80/443**; **Nginx** as reverse proxy (app on `127.0.0.1:8080`), `HTTP→HTTPS` redirection, **Let's Encrypt** certificates (webroot) with **auto-renewal** and Nginx reload.
- **Multi-environment setup**: **tfvars** for instance types, CIDR, disks, and rules; reproducible **IaC** deployments.
- **GPU (optional)**: instance families `g4dn.xlarge`/`g5.xlarge` as required; verification of **Service Quotas** and regional availability; installation of NVIDIA drivers and `nvidia-container-toolkit`.
- **Quotas**: vCPU/GPU increase requests through **Service Quotas** and Support tickets; ability to start

with **CPU** for pipeline validation, then upgrade to GPU after approval.
- **OpenAI integration** via `OPENAI_API_KEY` and `OPENAI_BASE_URL`; **local LLM** (**Ollama**) as fallback to reduce external API costs.
- **Cost & governance**: budget sizing, proper **destroy procedures** documented and enforced.

vSphere Infrastructure Automation with Terraform 2024
Full automation of the **VMware vSphere** environment, including the installation of **vCenter** and **ESXi** hosts, automated VM provisioning with **Terraform**, and centralized, secure administration of virtual resources.

Automated Oracle Cloud Deployment 2024
Infrastructure automation on **Oracle Cloud** using **Terraform**, covering VM and VCN provisioning with dynamic AD management, automatic AD rotation in case of failure, and secure resource governance.

Kubernetes Cluster Automation with Bash 2023
Development of **Bash** scripts for efficient Kubernetes cluster management, including resource optimization, health monitoring, automated initialization and cleanup of obsolete configurations, and detailed logging system setup.

Ansible and Vagrant Automation 2023
Multi-platform deployment (**Hyper-V**, **VirtualBox**, **VMware**) with **Ansible**-driven automated configuration, network management, and full validation of connectivity and configurations.

Certifications

AWS Fundamentals - AWS In Progress
Understanding of cloud services and resources, deployment models, and security on **AWS**.

Kubernetes - KodeKloud In Progress
Preparation for CKAD, CKA, and CKS certifications.

Kubernetes and OpenShift - IBM 2024
Mastery of Kubernetes concepts: objects, networking, security, deployment, and an introduction to the OpenShift ecosystem.

Docker Essentials - IBM 2023
Creating and managing Docker images, service orchestration with Docker Swarm, scalability, and image backup to Docker Hub.

NDG Linux Unhatched - Cisco 2023
Introduction to shell commands, user management, file system manipulation, process management, and Bash scripting.

Networking Basics - Cisco 2023
Fundamentals of networking: connectivity, IP addressing, subnetting, routing, and communication protocols.

SQL and Databases - IBM 2023
Designing and querying relational databases.

Education

Master's in Computer Science: Big Data and Cloud Computing 2022-2024
École Normale Supérieure de l'Enseignement Technique Mohammadia.

Bachelor's in Mathematics and Statistics 2014-2020
Faculty of Sciences Semlalia Marrakech.

Languages

French: Fluent **English**: Fluent