AYAN JYOTI

✉ jyotiayan47@gmail.com ☎ +91 9062319168 Hooghly, India ayanjyoti portfolio ayanjyoti17

OBJECTIVE

Motivated DevOps and Cloud Engineer with hands-on experience in automating CI/CD pipelines, managing containerized applications, and deploying scalable infrastructure on AWS using Docker, Kubernetes, and Terraform. Proficient in Linux, networking, and scripting with Python and Bash. Eager to contribute to cloud automation, system reliability, and DevOps best practices in a fast-paced engineering environment.

EDUCATION

Academy Of Technology , MAKAUT University 2021–2025
B.Tech in Computer Science Engineering
CGPA ***– 8.05***

SKILLS

- **Cloud Platform**: AWS (EC2, S3, VPC, IAM)
- **Infrastructure as Code (IaC)**: Terraform, Ansible
- **CI/CD Tools**: GitHub Actions, Jenkins
- **Containerization & Orchestration**: Docker, Kubernetes
- **Scripting Languages**: Bash Scripting, Python
- **Version Control**: Git, GitHub
- **Monitoring & Logging**: Prometheus, Grafana
- **Operating Systems**: Linux (Ubuntu), Windows

PROJECTS

Automated DevSecOps Pipeline for a Node.js Application | *GitHub Link*

- Designed and implemented an end-to-end CI/CD pipeline using **GitHub Actions** to automatically build, test, and deploy a containerized **Node.js** application.
- Integrated **SonarQube** to check code quality and **Trivy** to scan **Docker** images for security vulnerabilities, failing the build if critical issues were found.
- Automated deployment to a **Kubernetes** (**Minikube**) cluster using kubectl apply and a **self-hosted GitHub Runner** to manage rolling updates.
- Found and fixed complex, real-world bugs, including Docker network (DNS) failures, Kubernetes pod CrashLoopBackOff errors, and Windows Firewall (ERR_CONNECTION_REFUSED) issues.

Provisioned a 3-Tier Web Architecture on AWS with Terraform | *GitHub Link*

- Automated the provisioning of a scalable and secure 3-tier web architecture on **AWS** from scratch using **Terraform**.
- Designed a custom **VPC** with public subnets for the web tier and private subnets to secure the database tier, controlling all traffic using **Security Groups** and Route Tables.
- Wrote **Terraform** configurations to automatically launch EC2 instances and configure them at boot-time with user_data scripts to install and run an Apache web server.
- Troubleshot and fixed a critical deployment failure by debugging the user_data script, identifying and resolving a package-name conflict (httpd vs. apache2) between different Linux AMIs.

AWS Cost Optimization Script with Python and Lambda | *GitHub Link*

- Developed a **Python** automation script using the **Boto3** library to find and delete orphaned **AWS** EBS snapshots, directly reducing cloud storage costs.
- Engineered the script to run as a serverless **AWS Lambda** function, creating an event-driven solution for automated cloud resource management.
- The script's logic isolates stale snapshots by fetching all snapshots and cross-referencing them against the volumes of all active EC2 instances.