

PIERRE MVITA

Junior Cloud Engineer

Toronto, Ontario, Canada | +1 437 575 8178 | PierreMvita@yahoo.ca | linkedin.com/in/pierre-mvita | github.com/pmvita | pierremvita.dev

CERTIFICATIONS

AWS Certified Solutions Architect – Associate (SAA-C03) | CompTIA Security+ | Docker Foundations Professional Certificate | Google IT Support Professional Certificate

TECHNICAL SKILLS

AWS: Lambda, API Gateway, Step Functions, ECS Fargate, ECR, EC2, S3, DynamoDB, RDS, SQS, SNS, CloudWatch, IAM, VPC, SSM Parameter Store

Infrastructure as code: AWS SAM, CloudFormation, Terraform, Docker, GitHub Actions

Security & networking: Least-privilege IAM, role separation, VPC and subnet design, security groups, VPC endpoints, secrets in SSM Parameter Store, TCP/IP, DNS

Reliability & observability: CloudWatch logs and alarms, structured logging, dead-letter queues, retries with backoff, autoscaling, RTO/RPO planning, rollback procedures

Languages: TypeScript, JavaScript, Node.js, Python, Bash, SQL, PostgreSQL

Ways of working: Well-Architected Framework, cost optimization, Agile, runbooks and technical documentation

HIGHLIGHTING PROJECTS

NimbusFlow — Serverless Event-Driven Data Pipeline on AWS

2026

Technologies: AWS SAM, Lambda, API Gateway, Step Functions, DynamoDB, SQS, SNS, CloudWatch, IAM, TypeScript

- Built a serverless event pipeline — API Gateway into Lambda, onto SQS, through a Step Functions workflow, into DynamoDB, out to SNS — with the whole stack, IAM and alarms included, living in one AWS SAM template.
- Wrote seven TypeScript Lambdas for the pipeline stages, validating payloads at the API edge so a malformed request gets a 400 instead of clogging the queue.
- Put SQS and a dead-letter queue between ingest and processing, and gave every Step Functions state retries and a catch, so a failure gets recorded and emailed rather than quietly disappearing.
- Gave each Lambda its own IAM role with real resource ARNs instead of wildcards, and added JSON logging plus CloudWatch alarms on function errors and dead-letter queue depth.

CloudLedger — Containerized REST API on AWS (Architecture & IaC Design)

2026

Technologies: Terraform, ECS Fargate, Application Load Balancer, RDS PostgreSQL, S3, ECR, SSM Parameter Store, Docker

- Designed a stateless containerized API on ECS Fargate behind a load balancer, with Postgres on RDS kept in private subnets and uploads in a locked-down S3 bucket.
- Mapped out the Terraform for the whole stack (VPC, ECS, ALB, RDS, S3, IAM, SSM, budget alarms) so the environment can be stood up and torn down on demand.
- Split the ECS execution role from the task role and scoped every policy to a specific ARN or prefix, with database and JWT secrets in SSM rather than baked into the image.

EXPERIENCE

Cloud Engineering Projects (Independent)

2024 — Present

Self-Directed AWS Portfolio Work, Toronto, ON

- Build AWS workloads end to end and define them in SAM, CloudFormation, or Terraform instead of the console, so the whole environment can be rebuilt from the repo.
- Scope IAM tightly by default: one role per component, real resource ARNs, and no long-lived access keys sitting in application environments.
- Plan the failure path alongside the happy path, using dead-letter queues, retries with backoff, health-checked deploys, and a written rollback.
- Wire up CloudWatch logs, metrics, and alarms so problems arrive as an email rather than as a surprise later.
- Watch the bill closely with budget alarms, right-sized resources, on-demand billing, and tearing down anything left idle.

Technical Consultant (Sub-Contracts)

Sep 2024 — Apr 2025

Independent Contract Engagements, Toronto, ON

- Cut monthly cloud spend for a client by about 25% by right-sizing resources and moving cold data to cheaper storage tiers.
- Moved over 10TB of business data into S3 with nothing lost, using lifecycle rules and a documented recovery process.
- Set up VPC segmentation and scoped IAM policies to keep sensitive client data isolated, all under NDA.
- Configured cloud workspaces and wrote the runbooks and user guides so clients could take it from there.