HANDS-ON PROJECTS

AWS

● Using AWS Budgets to set custom budgets for tracking costs and usage of resources in an AWS account. Setting up alerts by email when actual or forecasted cost and usage exceed budgeted threshold.

● Using AWS Identity and Access Management (IAM) to manage access to AWS services and resources securely, as well as setting up Multi-Factor Authentication (MFA) for extra protection of the environment.

● Using Amazon’s unlimited simple storage service (S3) to host a static website where individual web pages include static content. Also using this service to store developer codes before hosting them on virtual servers (EC2) for dynamic web apps.

● Used Amazon’s Virtual Private Cloud (Amazon VPC) service to launch AWS resources in a logically isolated virtual network that I designed. Created a public and a private subnet and used the private subnet for databases and web servers to protect the resources from the internet. Elastic Load Balancing was used to automatically distribute incoming application traffic across multiple targets.

● Hosted several web applications (e.g., a PHP-MySQL demo app) on Amazon’s Elastic Compute Cloud (EC2) service to provide secure, resizable compute capacity in the cloud. Also used Amazon’s EC2 Auto Scaling to help maintain application availability and to automatically add or remove EC2 instances according to conditions I defined.

● Bash script to install a LAMP stack on ubuntu

● Used CloudFront, a fast content delivery network (CDN) service to securely deliver my demo application to “customers” globally, thereby enabling low latency and high transfer speeds for the application.

● Used Amazon’s Route 53, a highly available and scalable cloud Domain Name System (DNS) web service to route end users to my demo application using a custom domain name I bought.

● Created an SSL certificate using AWS Certificate Manager, a service that allows easy provisioning, managing, and deploying of public and private SSL/TLS certificates for use with AWS services and internal connected resources. I used this to secure the network communications to my application and to establish the identity of my websites over the Internet i.e., enable HTTPS.

● Used Terraform and cloud formation to create eCommerce website with resources such as VPC with Private and Public Subnets, an Internet Gateway, a Route Table, Security Group, a NAT Gateway, SNS, ALB, ASG, Route 53 and EC2 instances.

● Step-by-step on how to deploy a production-ready WordPress website on AWS.

● Created a Docker file for node.js app and created a ci/cd using Jenkins for the app and then push the app to Docker hub and AWS ECR, use AWS ECS Fargate for the containerized app and push to GitHub.