# DevOps Take Home Assessment

**Problem - 1**

Create IAM Policies for following scenarios

* Allow a user to view S3 buckets and read/write access to a specific bucket and objects of all types within the bucket.
  + Bucket name: xyz-media
* A user requires administrative access to all resources and read only access to IAM. However users should be able to perform following actions on their own IAM user
  + changing password
  + add/modifying MFA
  + add/remove access keys.
* You have configured RDS password rotation via secret manager for two users. Each secret has the username configured as a prefix. Craft a policy that would only allow users to read their own secrets. However this policy should not block the users ability to read any other secrets in the secret manager.
  + Secret naming convention - <username>-rds-credentials
  + Usernames -
    - mark
    - harry

**Problem - 2**

You have been tasked to create a bash script to accomplish the following. Provide a script file and provide steps on how you would implement it on an EC2 instance (Ubuntu 22.04) which would execute this script at instance startup or restart.

* Check if Java is installed, if not install the latest OpenJDK version.
* Check if security updates are configured as unattended updates and if not enable them.
* Install the following package if not already installed -> <https://www.elastic.co/guide/en/fleet/current/install-standalone-elastic-agent.html>

**Problem - 3**

Propose solutions for the following scenarios

**Scenario 1 - Hosting a Website**

Host a static website with 10 pages and 500MB in assets with a SSL certificate. The domain is already configured in Route 53. Propose a solution that would have the least management overhead while being cost effective.

**Scenario 2 - Application Deployment**

A business analyst has requested your input on how to deploy a python based data processing application.

*Application Functionality*

* The application will query data from multiple sources including RDS/Redshift/S3.
* Temporary store the data in the application and do the necessary processing.
* Output the data to a RDS instance.

*Considerations*

* The data processing activity will be automatically carried out every 6 hours.
* Data processing will take an average of 45-60 minutes
* Application must be automatically scalable to handle any data load.
* Application must be easily deployable in multiple AWS environments with minimal downtime between updates.y
* Application will not require a persistent data store as any processed data will be discarded immediately after its been copied over to the RDS instance.

**Scenario 3 - Data Transfer**

What would be the most cost effective solution to migrate two S3 buckets containing 500GB of data in the Sydney region to a new set of buckets in Oregon. Provide a cost breakdown.

**Scenario 4 (Optional) - Data Transfer (Cross Provider)**

You have been tasked with extracting a table with 20 million records in an AWS MySQL RDS instance and transferring them to a BigQuery table in Google Cloud Platform.

**Problem - 4**

Create an nginx container hosting any python/javascript application (any sample application) and deploy the container on an ECS cluster.

***Part 1 -*** Create a Docker file configuring Nginx on a latest Ubuntu base image and bundle the application including a process manager (pm2/gunicorn) for the sample application.

***Part 2 -*** Create a Jenkins based CI pipeline to build the image and push it to the ECR repository.

***Part 3 -*** Provision the Terraform infrastructure necessary to deploy the container to a ECS Cluster. Use the following configuration as the basis for the infrastructure. Create any other resources as required. Utilise a S3 backend with DynamoDB as state locking mechanism.

* VPC – 10.0.0.0/16
* Subnets
  + EC2 Private 1 – 10.0.10.0/24
  + EC2 Private 2 – 10.0.11.0/24
  + ELB Public 1 – 10.0.20.0/24
  + ELB Public 2 – 10.0.21.0/24
* Route Table
  + EC2 Private RT
  + ELB Public RT
* NACL
  + EC2 Private NACL
  + ELB Public NACL
* ALB
  + Listener Rules
    - HTTP redirected to HTTPS.
    - HTTPS pointed to the ECS.
* ECS Cluster
  + Capacity Provider - EC2
* WAF
  + Rule sets
    - AWSManagedRulesCommonRuleSet
    - AWSManagedRulesAmazonIpReputationList
* IAM roles/policies

Note - Provision any other resource that may be required to facilitate a web application.

***Part 4 -*** Create an Infrastructure diagram for the provisioned resources.

***Part 5 -*** Create a Jenkins based CD pipeline to deploy the container to the ECS cluster.

**Problem - 5 - Stretch Goal 01**

Deploy the nginx container created in Problem 5 in a Kubernetes cluster.

***Part 1 -*** Deploy the container as a stateless application with 3 replicas and configure the service and nginx ingress.

***Part 2 -*** Create a CD pipeline to deploy the application to Kubernetes. Use any CI/CD tool as you see fit.

**Problem - 6 - Stretch Goal 02**

Create a lambda function to fulfil the following monitoring requirement.

* Check all the security groups in all regions within an AWS account and identify any rules that allow inbound connections from anywhere from the internet (0.0.0.0/0 and ::/0) other than for HTTP and HTTPS ports.
* Send an email containing the following information for all the identified rules.
  + Region
  + Security Group Name
  + Security Group ID
  + Inbound Rule Detail
* Execute the lambda function on a daily basis at 00:00 UTC