High- Level Architecture For Quantum Airlines

Background:

A new cloud-based database python application is required to manage Quantum Airlines flights schedules. The application should be delivered through a muti stage Jenkins cicd pipeline. The pipeline should build a secure Kubernetes cluster then deploy the python application within docker containers.

Design Approach:

The application will be hosted within AWS Elastic Kubernetes Service (EKS). Due to the urgency for the solution, tried and tested secure AWS templates will be used to build-out the Kubernetes platform. For Scalability and Reliability, the application will be managed by EKS Fargate. Docker will be used to build the Kubernetes pods. Cloudformation will be used to build the multi-AZ application database (MySQL).  The CICD pipeline will be used to build all components of the new application

High Level Architecture

The High Level architecture is captured by the stages of the CICD pipeline. The pipeline is scripted through a Jenkins file. The following section describes the major architecture through Jenkins stages:

1) 

stage('Build fargate Kubernetes cluster with eksctl')

The first stage deploys the Fargate EKS stack using the ***eksctl*** utility. This approach was used to speed up the secure environment build. The eksctl utility uses a restrictive stack (EksBuild.json). However, the stack is customised with parameter values for tags. The stack builds a public-facing Kubernetes control plane while the compute resources are in private subnets across availability zones.

The following diagrams show the built architecture:

<https://aws.amazon.com/blogs/compute/task-networking-in-aws-fargate/>

2)

stage('Build database instance, schema and load data with docker')

A multi-AZ MySql database instance is built using a cloudformation template called createDB.json.  After the build, a mysql docker image is used to build a database and table for the flight data.

3)

  stage('Build the docker application image for EKS cluster')

This stage builds the docker image which will be used in a deployment to the K8 Fargate cluster. The dockerfile and application code are included as part of this submission

4)

 stage('Deploy containers to the EKS cluster')

This stage uses a .yml file to deploy the application using kubectl.

5)

stage('Auto scale fargate')

This stage uses a .yml file to autoscale the pods.

This concludes the high level overview of the solution.