**AO Technical Task Submission – Matthew Marshall**

**Task 1 – NGINX Container & Application Container (Incl. Docker-compose)**

* Dependencies
  + AWS Backend Bucket
  + EC2 Key Pair
* Docker-compose file allows quick and easy local setup for NGINX routed to AO application container
* Use ./ECR to deploy Container Registry
* Push AO and NGINX images to registry for initial deployment
* Updated Repo URI in container-def.json to run code
* Within the repository main directory located is the main infrastructure
* VPC, Subnets, IGW, NGW, Public & Private Route Tables, Load Balancer, Auto Scaling Group, SG’S, ECS Cluster running EC2 type.
* The Load Balancer, forwards traffic onto the NGINX reverse proxy server which then redirects to the application
* Terraform is written in modules

**Task 2 – CICD Experiment in IaC**

* AWS CodeBuild & AWS CodePipeline were chosen for the CI/CD tools.
* Terraform for this is in the CICD folder.
* CodeBuild builds the docker image with docker privileged rights
* The buildspec.yml logs into ECR, builds the AO application then pushes it to registry
* Artifacts then exported in json format to deploy directly to our ECS Cluster.
* IAM roles crated for CodeBuild & CodePipeline
* GitHub credentials used in the source-credentials.tf
* Webhooks set to push

**Task 3 - Recommendations**

* Use HTTPS, SSL certs and redirect HTTP traffic to HTTPS
* Terraform state locks could be introduced
* WAF on the external load balancer.
* Add Terraform deployments to the pipeline.
* Currently in the Terraform no Bastion host is setup to access underlying infra – I would add Bastion.
* Use Parameter Store / Secrets Manager for credentials
* AWS Fargate for Serverless deployment
* IAM Review – granular policies for IAM roles.