**COALFIRE Tech Challenge Doc – E. Akonawe**

[akonawe/coalfire (github.com)](https://github.com/akonawe/coalfire)

The solution provided, per instructions, is to create a fault tolerant, cost efficient, highly available web application. Utilizing dual availability zones within a geo region allows for data and traffic redundancy as well as optimized application performance by routing traffic through an application load balancer. From a cost savings perspective, the use of an autoscaling group for the web application Ec2 instance management, allows for the automatic scaling of needed and removal of underutilized resources. A singular VPC and singular region architecture also allows for cost savings. The s3 bucket segmentation between logging and image data storage, allows for the gathering and storage of application logs and image data independent from the instances (which can be destroyed at anytime based on utilization and demand).

**Notes:**

When running terraform code for environment and resource creation, be sure to edit the **vars.tf** file to update AWS account authentication details such as: secret key, access key, region, and existing key-pair id.

**Issues encountered:**

* Due to some time constraints for personal reasons, the below requirements are incomplete:
  + 2 folders on singular S3 bucket, with 2 policies
  + ALB listener throwing 400 error upon creation that needs to be addressed.
  + Screenshot of EC2 instances is not attached because I could not finish troubleshooting the environment/code in time to successfully connect to an instantiated instance

Resources:

[Terraform Registry](https://registry.terraform.io/)

[Terraform - Security Groups & EC2 instances - Cloud Walker](https://www.cloudwalker.io/2020/09/13/terraform-security-groups-ec2-instances/)

[How to Create S3 Bucket Policy using Terraform - CloudKatha](https://cloudkatha.com/how-to-create-s3-bucket-policy-using-terraform/)