**Web Application Deployment in AKS cluster**

This document outlines the necessary steps for effectively utilizing the Azure CLI, Terraform CLI, and Kubectl CLI for cloud management and deployment. It is essential to install all three command-line interfaces first.

The process begins with authenticating to Azure Cloud using the Azure CLI. Next, we will develop Terraform scripts to create a resource group, set up an Azure Kubernetes Service (AKS) cluster, and establish an Azure Container Registry (ACR).

Once the cluster is configured, we will create a role assignment to allow the AKS cluster to retrieve images from the ACR. After setting up the cluster successfully, we will construct a pipeline to facilitate the deployment of container-based applications.

Access to the cluster will be managed using Kubectl commands.

To log in to the Azure portal, please execute the following commands:

**az login --service-principal -u ${ARM\_CLIENT\_ID} -p ${ARM\_CLIENT\_SECRET} --tenant ${ARM\_TENANT\_ID}**

**az account set --subscription ${ARM\_SUBSCRIPTION\_ID}**

### Terraform Commands for Provisioning AKS

* `terraform validate`
* `terraform init`
* `terraform plan`
* `terraform apply --auto-approve`

If you lack Owner access or User Access Administrator roles, it is necessary to create a service principal:

**az ad sp create-for-RBAC --name <service-principal-name> --role Contributor --scopes /subscriptions/<subscription-id>/resourceGroups/<resource-group-name>**

# Assign the AcrPull role to the service principal for the ACR

**az role assignment create --assignee <service-principal-id> --role AcrPull --scope /subscriptions/<subscription-id>/resourceGroups/<resource-group-name>/providers/Microsoft.ContainerRegistry/registries/<acr-name>**

### Azure Pipeline for AKS Deployment

* Automate the build process
* Automate the creation of Docker images
* Automate the upload of Docker images to the ACR
* Automate the deployment of containers to the AKS cluster

### Monitoring Application and Cluster Information

1. To monitor both the application and the cluster, the following steps should be undertaken:

* Install Helm.
* Install the Kube-Prometheus-Stack:

```bash

**helm repo add prometheus-community https://prometheus-community.github.io/helm-charts**

**helm repo update**

```

1. Deploy the chart into a new namespace named "monitoring":

```bash

**kubectl create ns monitoring**

```

1. Verify the installation with the command:

```bash

**helm install monitoring prometheus-community/kube-Prometheus-stack -n monitoring -f ./custom\_kube\_prometheus\_stack.yml**

```

1. Access the Prometheus user interface:

```bash

**kubectl port-forward service/prometheus-operated -n monitoring 9090:9090**

```

1. Access the Grafana user interface:

```bash

**kubectl port-forward service/monitoring-grafana -n monitoring 8080:80**

```

1. Access the Alertmanager user interface:

```bash

**kubectl port-forward service/alertmanager-operated -n monitoring 9093:9093**

```

This guide provides a comprehensive framework for setting up Azure services and monitoring applications within an AKS cluster, ensuring best practices are adhered to throughout the process.