**Public-Cloud-Recruitment Assessment Sentia**

This document illustrates the design I have proposed and prepared for the technical assessment given to me by Sentia.

Total time spent: 6 hours

**Content:**

1. Introduction
2. Design

* Technologies used

1. Infrastructure
2. Future aspirations for the project

**Introduction**

The purpose of this document is to provide the reader an overview of the design of complete infrastructure that I have proposed to deploy for the give case study.

For IaC I have used ARM template written in Json and a PowerShell script to deploy those templates.

It is vital to use the PowerShell script provided to deploy the resources, as I am using outputs of few deployments as parameters for next deployments.

Total duration for completing the assessment 6 hours

**Design:**

Below is the design overview of the infrastructure. The deployment mimics the design completely.

Graphical user interface, diagram

Description automatically generated

**Technologies used:**

ARM templates JSON

PowerShell

**Assumptions:**

* Instead of mongo DB cluster I have deployed a Single Cosmos DB account with one MongoDB and one collection. Additional collections can be added
* IP address for setting up S2S tunnel from Vnet to on premises office have been assumed
* Node.js, mongo DB versions are assumed
* Autoscaling is only configured for High CPU utilization. Any other matrix can be added later such httprequests, RAM utilization etc.

**Infrastructure**

I have deployed the infrastructure in an internal azure subscription and the deployment is identical to the design above.

**Azure App service (webapp):**

This is deployed from template file deploywebapp.json.

This resource serves the requirement of hosting a node.js application. I have deployed an app service plan which then hosts a webapp with Node.js version 10.14.

I have delegated the frontend subnet of the vnet for this webapp to restrict, filter traffic incoming towards this application.

The webapp currently scale’s out when the CPU utilization is more than 60% and scales down when utilization is below 20% with a cooldown period of 10 mins between the scale operations.

The app also has 3 deployments slots to serve various version of application – acceptance, testing, Production. One can easily deploy the application to any of this slot, test it and then swap it with production slot to achieve continuous delivery without downtime.

**CosmosDB:**

I have created a cosmos db account which is then used to host a mongodb database. Currently I have assumed and deployed only one container with one DB . This can be changed with more clarity on the application requirements.

This serves the replacement of on premises mongoDBs cluster.

The cosmosDB firewall rules have been changed to allow traffic only from backend subnet. I have enabled cosmosdb service endpoint on backend subnet to help achieve the connectivity.

**Storage account – File share:**

I have created a storage account which hosts a file share.

This File share will be the replacement of the FTP server to store documents.

I have created a private endpoint on backend subnet of the vnet for the storage account. This way the storage account is reachable by all resources via backend subnet. This was we achieve connectivity between cosmosdb and fileshare.

**Automation account:**

I have deployed an Atuomation account with purpose to run Python scripts on hourly basis.

In this automation account I have also created a python runbook which can contain python scripts which can be scheduled to run on hourly basis.

**Virtual network:**

This resource serves the connectivity requirements.

I have created one Vnet, with 3 subnets (frontend , backenc, gateway subnet) a gateway and a connection object.

The frontend subnet is completely delegated to app service and I have configured the web app to use this delegation to filter traffic flowing to the webapp.

The backend subnet has a service endpoint enabled for cosmosdb and a private endpoint for storage account.

Frontend & backend subnets can communicate with each other but only on according to the rules set on NSG. I have 2 NSG one on each subnet specifying rules according to which traffic can flow through them. Both the NSG have block all rule at the end to ensure no traffic flow through them other than the rules specified with higher priority.

The gateway subnet is created to host a virtual network gateway. Client expectation is to be able to connect to their on premises resources securely and hence we have the gateway to establish a S2S tunnel with the on premises infrastructure. The connection object has a local gateway whose values are assumed for sake of deployment which helps the gateway to establish connectivity with assumed on premises IP’s. This way client can connect to their application securely in azure.

**Resource group, subscription:**

All the above resources are deployed in single resource group and in same subscription.

**Future aspiration for the project**

To prepare a CI/CD pipeline instead of a PS script to automate the deployment in a much better structured manner

Create cluster of DB, containers in cosmosdb.

Have better autoscaling rules with more metrics such as http requests, memory utilization etc.

Configure backups for DR scenarios