CLOUDS Assignment 1

This assignment is a series of labs using the Azure cloud. You will get the play with two of the key virtualization technologies you learned about in lecture 2 (VM and containers). Before you do each part, go over all instructions. First, read all the resources and make a list of things to do and commands to run. This way you will get the complete workflow before you start doing things.

Follow instructions in the book. Particularly, follow conventions in naming ResourceGroups and using them effective to group resources. Use a separate RG for each assignment. This way, later, you will be able to clearly see billing per RG this way, and also delete and deallocate resources easily.

1. Create a free Azure account

1. Go to https://azure.microsoft.com/en-us/free/students/. Follow the instructions to create your account. Sign up for the Education account for Students. This will give you a 1 year credit for $100.
2. Get AzureMol book (https://azure.microsoft.com/en-us/resources/learn-azure-in-a-month-of-lunches/). You can also get a copy from Moodle if you want.
3. Read Chapter 1 from AzureMol book

2. Deploy a website on Virtual Machine

1. Read AzureMol book chapter 2.1, 2.2, 2.3
2. Create a Ubuntu virtual machine (basic B1 or similar free tier instance type) and install your favorite webserver(nginx, lamp-stack). You can follow the book's advice.
3. Update the default index.html with your own portfolio website with static HTML and optionally CSS files and images (a simple single page html will also be enough, no fancy web dev is required).

3. Provision disks in VM and measure IOPS

1. Read AzureMol book chapter 4.1, 4.2, 4.3. Use https://learn.microsoft.com/en-us/azure/virtual-machines/linux/tutorial-manage-disks as a reference
2. Resize vm you created earlier to get a DS2_V2 type or another which can allocate Premium SSD storage.

3. Dynamically add 2 disks to the linux VM you created earlier – one of each StandardSSD_LRS, and Premium_LRS types.
4. Format and mount file systems using the two disks (You can use the Microsoft learn resource as a reference)
5. Install FIO (https://installati.one/ubuntu/20.04/fio/) and use it to measure the sequential throughput (for 64KB size) and random read IOPS (4KB size) on the two devices (FIO . Here's an example: https://blog.purestorage.com/purely-technical/io-plumbing-tests-with-fio/. You don't need to use same commands. Feel free to use your own. Also, you don't need to run FIO for a long time. Once the file is created, you can run it for just one minute.
6. Unmount the disks
7. Detach the disks from running VM, delete the disks
8. Resize the VM back to old config.

4. Deploy dockerized static site via Azure ACR

1. Install docker **within ubntu vm you created earlier**
   - Use this as a reference for installation: https://docs.docker.com/engine/install/ubuntu/
   - After installing here's permission setup you need to do to run docker as a non-root user: https://docs.docker.com/engine/install/linux-postinstall/
   - Tip: You need to logout and back in for permission to kick in
2. Create a dockerized static site **from within VM** and verify with CURL
   - Use azuremol chapter 19.2 as reference for the Dockerfile (to get a nginx container that serves your static html)
   - Run docker locally in VM and use CURL or wget to verify your local docker deployment (curl http://localhost:80 should give you the html)
3. Create ACR and get login server **from Azure Shell**
   - Use https://learn.microsoft.com/en-us/azure/container-instances/container-instances-tutorial-prepare-acr as reference
   - Tip: There is no need to login from azure shell, you just need the login server
4. Push to ACR **from within VM**
   - Reference: https://learn.microsoft.com/en-us/azure/container-instances/container-instances-tutorial-prepare-acr
   - Tip: You have to docker login first before you can docker push (https://www.c-

sharpcorner.com/article/push-pull-docker-image-with-azure-container-registry/)

5. Deploy app and verify **from Azure Shell**
   - https://learn.microsoft.com/en-us/azure/container-instances/container-instances-tutorial-deploy-app
   - Tip: get fqdn and verify site is working (make sure you use http and not https)

5. Create Azure webapp static site

1. Read AzureMol book chapter 3. Use https://learn.microsoft.com/en-us/azure/app-service/quickstart-python?tabs=flask%2Cwindows%2Cazure-portal%2Clocal-git-deploy%2Cdeploy-instructions-azportal%2Cterminal-bash%2Cdeploy-instructions-zip-azcli as reference (particularly create web app and use local git parts)
2. Create a webapp on azure (use ASP.NET v3.5 runtime)
3. Via deployment center, create a local git repo.
4. Create a local directory with a static html file. You can use the same file from #1.
5. Configure git (git init, git config… as per book) via azure cloud shell in this directory. Push the index.html site you created to the git repo to deploy the website.

6. Deploy static site on Azure Blob Store

1. Use https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blob-static-website-how-to?tabs=azure-portal as a reference to deploy your index.html page on Azure blob store.

Deliverable: at the end of this assignment, you have to fill and submit on moodle a file called “Azure-Lab1-Deliverable.docx”, which is available in the assignment 1 section on moodle.