



Summer of Azure 2022-2023

The screenshot shows a web application with a dark blue header containing a small green infinity logo. The main content area is white and features the heading "What's on your mind?" followed by the text "You can be honest - This submission is anonymous". Below this is a large, light gray text input box with the placeholder text "We want to keep improving and your honest opinion makes this possible!". A green "SUBMIT" button is centered below the input box. At the bottom, there is a section titled "What others have said:" with a green quote icon and a paragraph of text about REI.

AzureAnonymous

Application Process and Installation Guide

A Real-World Project Created For:



By Chris Chong

A quick 10-minute demo of this project can be found at https://youtu.be/PwTDgw8_Rc8

Table of Contents

Overview	3
Approach	3
Stage 1: Setting up and load balancing webservers in Azure	3
Stage 2: Creating a database	6
Stage 3: Getting index.html form submits to talk to database	7
Stage 4: Installing application on VM	8
Stage 5: Testing for High Availability	9
Stage 6: Making it pretty	11
Technical Specifications & Installation	12
Prerequisites	12
Application Setup	12
For Users with Azure Account Pre-Setup Access	12
Azure Resources	13
Account Access (SSH, private key can be provided on request)	13
Step-by-step guide to setting up AzureAnonymous VMs	14
Create an Azure Virtual Machine	14
Basics	14
Disks	15
Configuring your VM	16
SSH into your VM	16
Login to VM and update/install packages Login with your username (in this case, azureuser), then	17
Clone project, enter directory, create venv	18
Install requirements into venv	18
Amend DB details in runner.py	19
Test run the application	19
Network Architecture Design & Costing	20
Designing for Minimum costs	20
Designing for High Availability	22
Designing for Maximum Availability	23

Note: This document does not cover the configuration of the overall Azure Virtual Network and load balancing solution, and focuses on providing an overview of the application development process as well as in-depth information on how to get the application up and running via the [Github repo](#).

Overview

AzureAnonymous is a simple application created with Cotiss in mind. [Cotiss](#) is built specifically to help organisations be more effective in how they buy & supply goods and services. They have requested a simple website where employees can anonymously submit feedback.

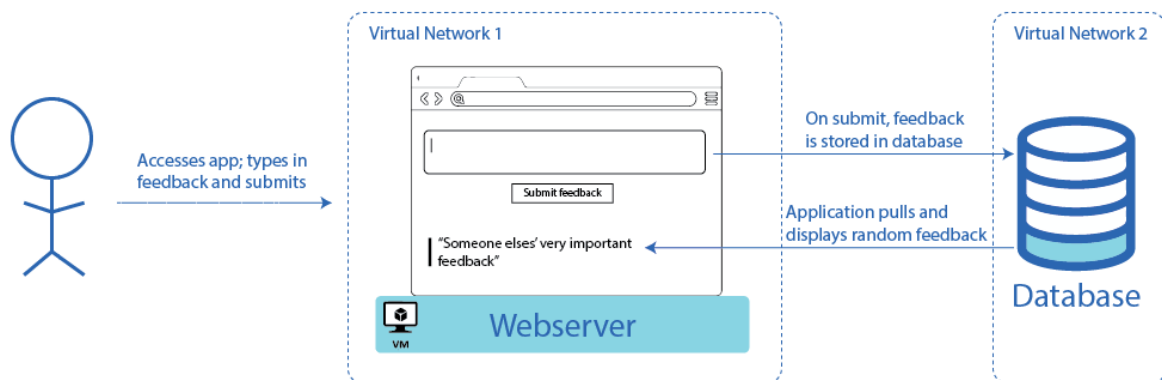
The UX design requested is simple; a random piece of feedback displayed on each page load, and a box at the bottom of the page with a submit button to add new feedback to the feedback bank.

The application itself was not the main goal of this project. The focus lies in the implementation of a high availability application on Azure.

Azure has many PaaS and SaaS solutions that can help eliminate the need to run this application on a virtual machine. However, given that we only had 2 weeks to create this solution, and since this is the first time that I have created a web app from the ground up, I chose to create this using the fundamental basics of web application architecture.

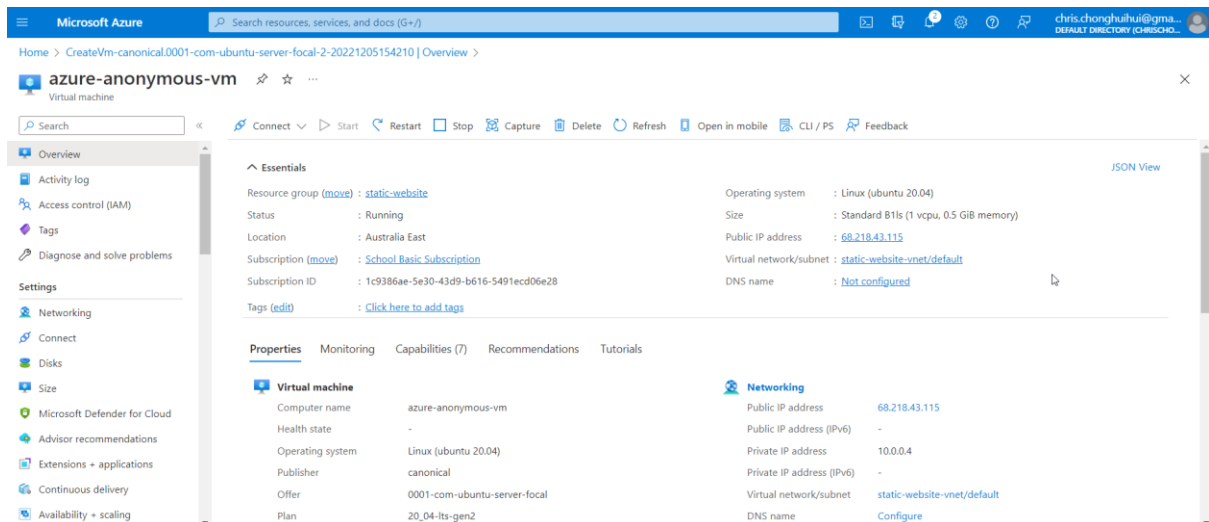
Approach

The most basic setup and interaction that I could identify is as shown:

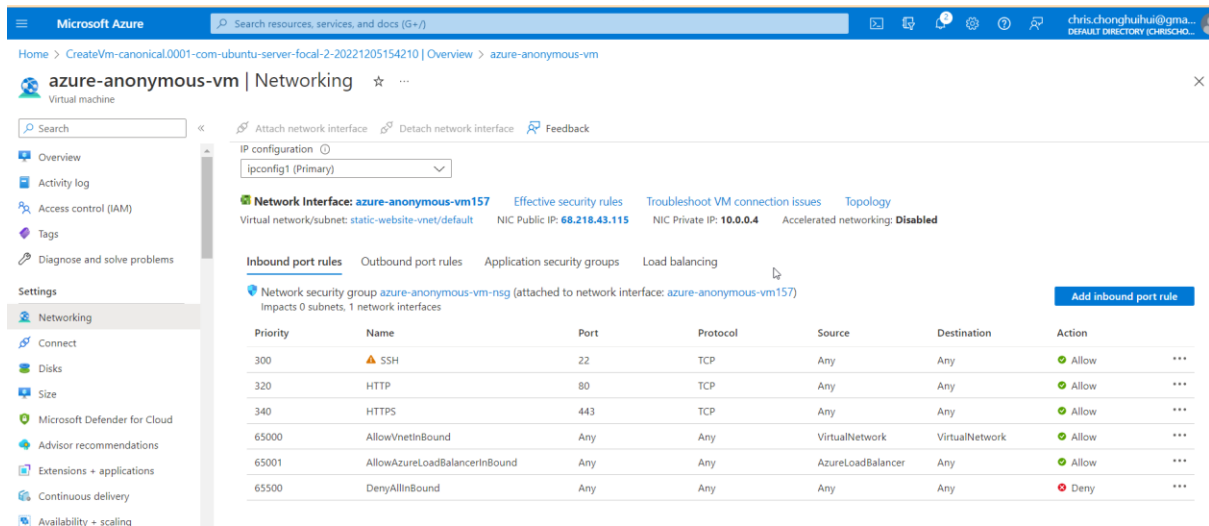


Stage 1: Setting up and load balancing webserver in Azure

My first thought was to learn how to set up a webserver on Azure, learn how to capture and image, and load balance it. I have prior experience in using NGINX as a reverse proxy on AWS and decided to use that for my solution.



Screenshot 1: Creating a tiny Ubuntu VM in Azure



Screenshot 2: Opening ports 80 and 443 (which I can close off later if things change) since website doesn't have any hugely secure data or keep any PIM

```
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
azureuser@azure-anonymous-vm:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
azureuser@azure-anonymous-vm:~$
```



Screenshot 3 & 4: Updating packages, installing Nginx, then checking to make sure that the implementation was successful

FEED BACK FORM

First Name:

Last Name:

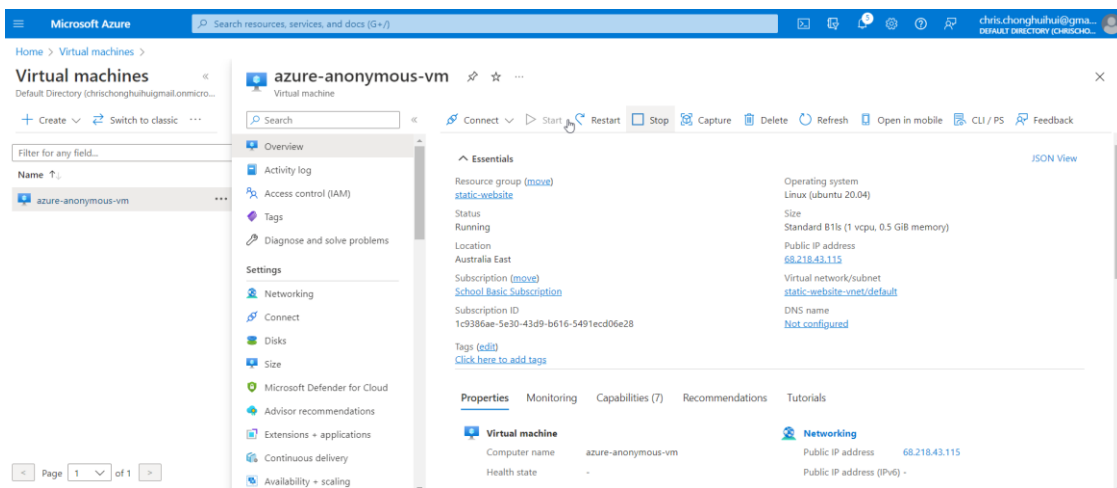
Mail Id:

Country:

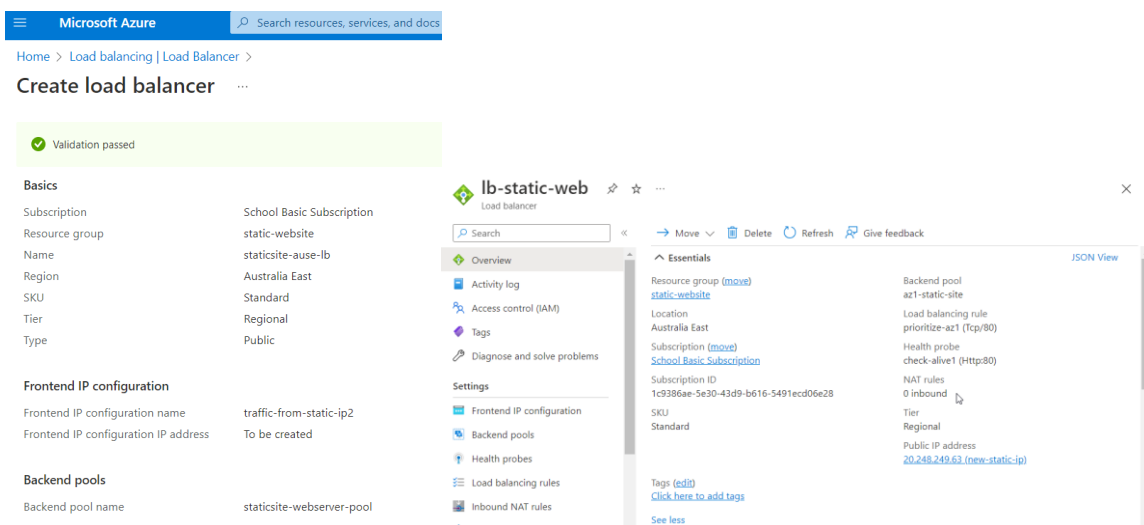
Feedback:

Write something..

Screenshot 5: Creating a test HTML feedback form using code from the following article:
<https://www.c-sharpcorner.com/article/creating-a-feedback-form-using-html/>



Screenshot 6: Capturing VM image for easy replication



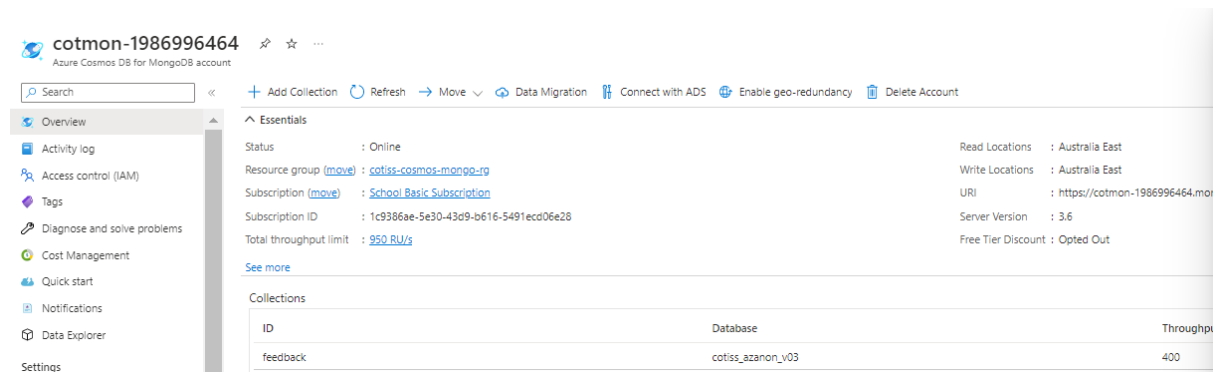
Screenshot 7: Creating a load balancer that evenly directs traffic to VMs in the pool

Stage 2: Creating a database

Once I was sure that the webserver was running as expected, I created a database in Azure using CosmosDB. I have heard of MongoDB being a non-relational database, and chose this because it would allow for flexibility in analysing the feedback, if needed.

The Quickstart Guide for Azure Cosmos DB for MongoDB for Python was really good as a general overview. I installed MongoDB on my local machine and CosmosDB on Azure, then used this guide to help me learn how to connect the two:

<https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/quickstart-python?tabs=azure-powershell%2Cvenv-windows%2Cdotenv>



Screenshot 8: Setting up CosmosDB database

Unfortunately, I forgot to take screenshots of the code here, but it involved something similar to the following:

```
1 import os
2 import sys
3 from random import randint
4 from pprint import pprint
5
6 import pymongo
7
8 import bson
9 from dotenv import load_dotenv
10
11 from flask import Flask, request, render_template, jsonify
12 import random
13
14
15
16 load_dotenv(verbose=True)
17 CONNECTION_STRING = "mongodb://cotmon-1986996464:tE7T1v9V8CMhM7fxoo2lvG9QMkqFH0BmLBuW53i2qLRXv2A0mHcCoal
18
19
20 DB_NAME = "cotiss_azonon_v03"
21 COLLECTION_NAME = "feedback"
22 client = pymongo.MongoClient(CONNECTION_STRING)
23
24 # Create database/collection if it doesn't exist
25 db = client[DB_NAME]
26 collection = db[COLLECTION_NAME]
```

Screenshot 9: Getting Python script to talk to CosmosDB database

When I was sure that the calls were actually creating entries in my Azure database (not locally!), I set the next goal of having my index.html form submissions make calls those calls to the database directly. I have a tiny bit of experience in Python, and remember being told that Flask can be used to develop web apps using Python and decided to use that as my web framework.

Stage 3: Getting index.html form submits to talk to database

```
21 COLLECTION_NAME = "feedback"
22 client = pymongo.MongoClient(CONNECTION_STRING)
23
24 # Create database/collection if it doesn't exist
25 db = client[DB_NAME]
26 collection = db[COLLECTION_NAME]
27
28 # Check the number of documents (records) in database
29 def countdocs():
30     num_docs = collection.count_documents({})
31     print("You have >>>", num_docs, "<<< records in your database")
32     # Set the id for the new feedback
33     return num_docs
34
35 # Gets and returns a piece of random feedback
36 def get_rand_feedback():
37     rand_feedback_index = 0
38     rand_feedback = ""
39     rand_feedback_index = random.randint(1, countdocs())
40     print(rand_feedback_index, "<<<<< This is the random record I'm trying to pull")
41     rand_record = list(collection.find({"_id": rand_feedback_index}))[0]
42     rand_feedback = rand_record['feedback']
43     return rand_feedback
44
45 # For debugging: print("Upserted document with _id {}".format(result.upserted_id))
46 print("Databases available are: ", client.list_database_names())
47 print("Collections available are: ", db.list_collection_names())
48 # for records in collection.find():
49 #     print(records)
50 print("CONNECTION STRING IN RUNNER IS ", CONNECTION_STRING, "<<<<<<<<\n")
51
52
53 # define the flask app
54 app = Flask(__name__)
55 app.config["MONGO_URI"] = CONNECTION_STRING
56
57 # define the home page route
58 @app.route('/')
59 def hello_world():
60     clean_feedback = ""
61     clean_feedback = get_rand_feedback()
62     return render_template("index.html", feedback=clean_feedback)
63
```

Screenshot 10: Getting HTML form to push and pull data from CosmosDB

I went back to the very basics to refresh myself on how HTML forms/queries work. This video was one that I found super helpful: <https://www.youtube.com/watch?v=TCEgdiN0A8s>

These two articles helped heaps in providing me with a general idea of how to write the Flask script to make the calls linking my index.html page to the database:

<https://www.linkedin.com/pulse/integrate-mongodb-flask-creating-simple-student-data-form-phatate>

<https://www.linkedin.com/pulse/integrate-python-flask-mongodb-abhijeet-karmakar>

Stage 4: Installing application on VM

To reduce the complexity of i.e. ports/connectivity etc, Stages 2 and 3 were done on my local machine. The challenge now was to port this over to my Azure VM, so that I could capture an image for easy setup. I had been “checking” my commits into my repository on Github, so this made the process of sharing application information much easier. I found that I went back and forth on my local (Windows) machine and VM (Ubuntu) quite a bit to try and figure out what worked. In the process, I also created a simple readme.txt file to help me remember the varying commands, but also to help make this repo useful to “future me” and other people might find this project interesting/useful.

```
readme.txt - Notepad
File Edit Format View Help
#####
# Instructions to install AzAnonymous
#####
# This assumes you have already created your MongoDB database,
# hosted on CosmosDB
#-----
# Tech stack:
# Ubuntu 20.04.5 LTS and above
# Nginx
# MongoDB (hosted on Azure CosmosDB)
# Python3 (python3-pip, pymongo, python-dotenv)
# Flask
# Gunicorn

# Clone project into a directory
git clone https://github.com/csido/AzAnonMontygo.git

# Install nginx

# If pip is not already in your system
sudo apt-get install python3-pip

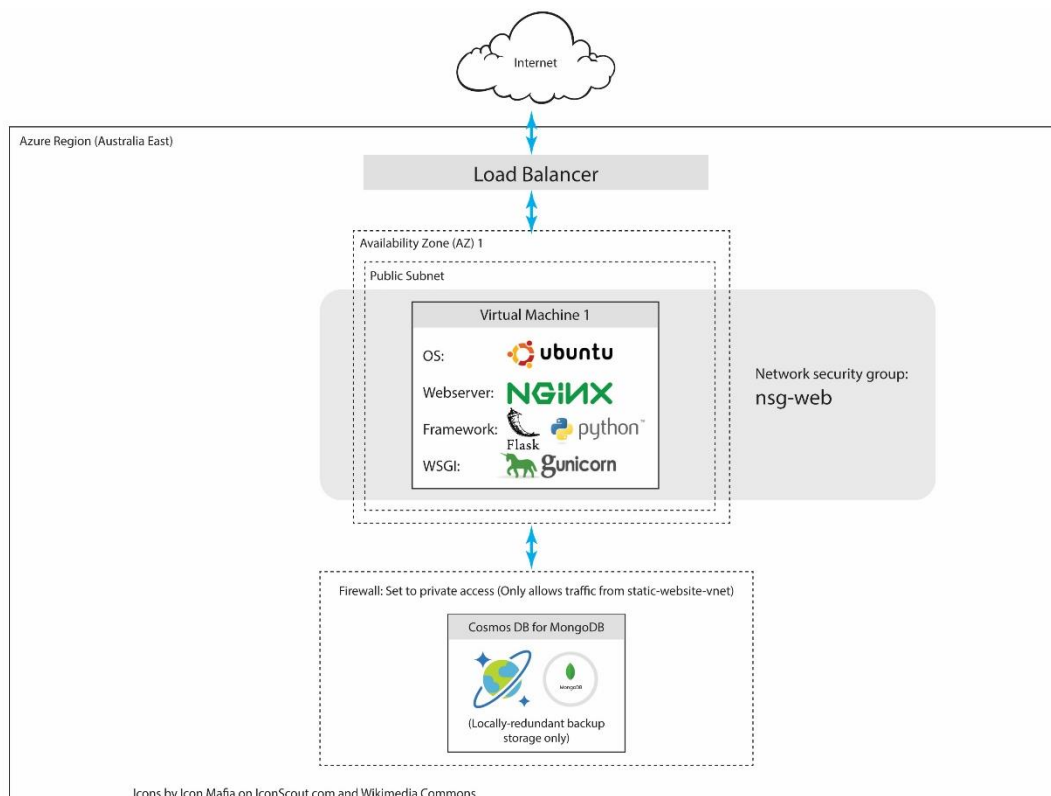
# Change directory to AzAnonMontygo
cd AzAnonMontygo

#####
# Development server deployment instructions
#-----

# Install venv
sudo pip3 install virtualenv
pip install virtualenv (windows local test)
```

Screenshot 11: Documenting commands to install/deploy application

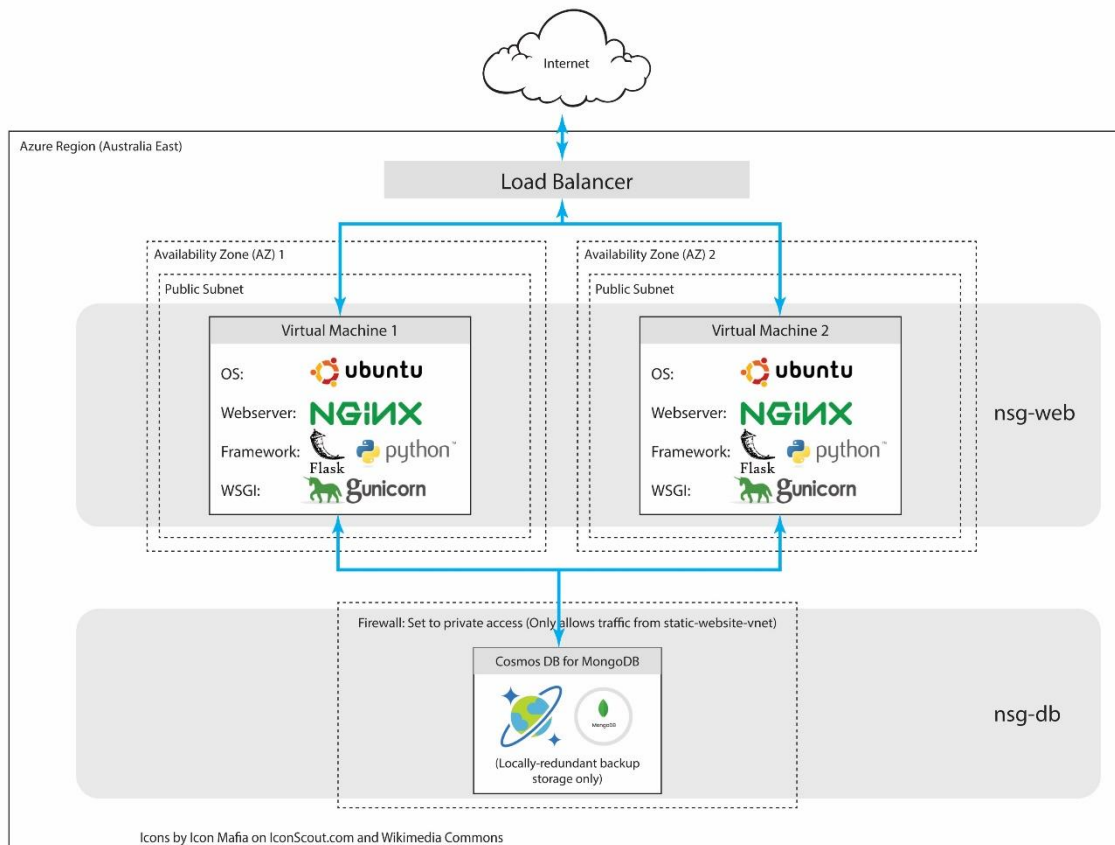
At this stage, I had the following basic setup:



Screenshot 12: Basic application architecture

Stage 5: Testing for High Availability

Once the application was up and running, I captured an image of the VM and looked into the simplest “high availability” solution for this application. My intention was to submit the following network design, with captured VM images that would allow easy deployment of more resilient architectures.



Screenshot 13: Simplest High Availability application architecture

Home > AzureAnonymous-dec2022-web-rg

Resource group

Search

Overview

Activity log

Access control (IAM)

Tags

Resource visualizer

Events

Settings

Deployments

Security

Policies

Properties

Locks

Cost Management

Cost analysis

Cost alerts (preview)

Budgets

Advisor recommendations

Monitoring

Insights (preview)

Essentials

Subscription (mvp) : School Basic Subscription

Subscriptions ID : 1c9386ae-5e30-43d9-b616-5491ecd06e28

Tags (edit) : [Click here to add tags](#)

Deployments : 13 Succeeded

Location : Australia East

Resources

Recommendations (7)

Filter for any field...

Type equals all

Location equals all

Add filter

Showing 1 to 19 of 19 records. ☐ Show hidden types

No grouping

List view

Name	Type	Location
1.0.0 (AzureAnonymousImages_dec22/AzureAnonymous-dec2022submission/1.0.0)	VM image version	Australia East
1.0.1 (AzureAnonymousImages_dec22/AzureAnonymous-dec2022submission/1.0.1)	VM image version	Australia East
1.0.2 (AzureAnonymousImages_dec22/AzureAnonymous-dec2022submission/1.0.2)	VM image version	Australia East
azanon-submissionkey	SSH key	Australia East
azanon-submit-vm1-az1	Virtual machine	Australia East
azanon-submit-vm1-az1-ip	Public IP address	Australia East
azanon-submit-vm1-az1-ni	Network interface	Australia East
azanon-submit-vm1-az1_OsDisk_1_0330728d35e044f5b61ba2644bea208c	Disk	Australia East
azanon-submit-vm2-az2	Virtual machine	Australia East
azanon-submit-vm2-az2-ip	Public IP address	Australia East
azanon-submit-vm2-az2841_z2	Network interface	Australia East
azanon-submit-vm2-az2_OsDisk_1_3ca0f2cd5ab64f90b3a0206fe9bc8814	Disk	Australia East
azanonVM-quickDeploy-17dec22-submissionReady	Template spec	Australia East
AzureAnonymous-dec2022-web-rg-vnet	Virtual network	Australia East
AzureAnonymous-dec2022submission (AzureAnonymousImages_dec22/AzureAnonymous-dec2022submission)	VM image definition	Australia East
AzureAnonymousImages_dec22	Azure compute gallery	Australia East
main-static-ip	Public IP address	Australia East
public-web-loadbalancer	Load balancer	Australia East
web-nsg	Network security group	Australia East

< Previous Page 1 of 1 Next >

Give feedback

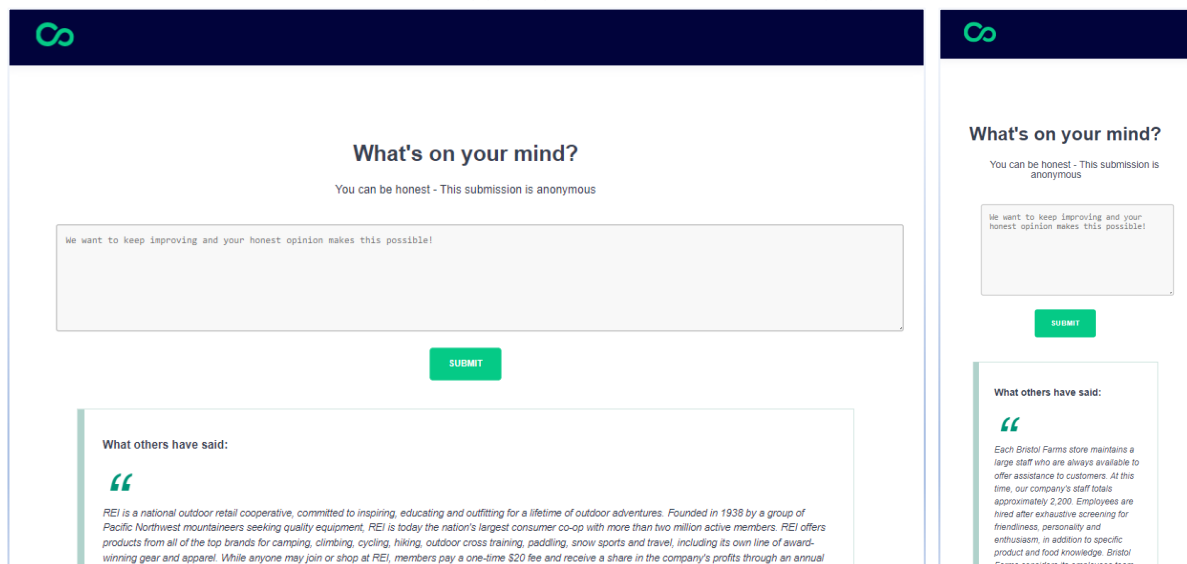
Screenshot 14: Resources required for the implementation of Minimum Cost application architecture (Screenshot 13)

Stage 6: Making it pretty

While this project's focus isn't on the UI/UX of the application, I felt that I needed to (at the very least) make it look like it was an application for Cotiss. With this in mind, I spent about a day or so making it look pretty.

```
runner.py x index.html x
136
137 .quote-text {
138   font-size: 14px;
139   font-style: italic;
140   line-height: 1.5;
141 }
142
143 .quote-text--intro {
144   font-weight: bold;
145   font-style: normal;
146   text-align: left;
147 }
148
149 .quote-text::before {
150   content: "\201c";
151   font-size: 100px;
152   color: #009578;
153   display: block;
154   margin-bottom: -70px;
155   margin-top: -20px;
156 }
157 }
158 </style>
159 </head>
160 <!-- HTML PORTION OF THE FILE -->
161
162 <header class="kl-header">
163   <nav class="kl-navbar content-wrapper">
164     <a href="/?hsLang=en">
165       
166     </a>
167   </nav>
168 </header>
169
170 <body>
171   <div class="container">
172     <center>
173       <h1>What's on your mind?</h1>
174       <label for="feedback">You can be honest - This submission is anonymous </label>
175     </center>
176   </div>
177   <div class="container">
178     <center>
179
```

Screenshot 15: Snippet of HTML + internal CSS styling code



Screenshot 16: Final desktop and mobile UI of the feedback form

The rest of the time was spent on tidying up documentation and report writing.

Technical Specifications & Installation

Prerequisites

This application was built and tested to be run on the following technology stack:

- **VM size:** Standard B1ls (1 vcpu, 0.5 GiB memory, suitable for test loads/env only)
- **OS:** Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-1029-azure x86_64)
- **Webserver:** Nginx/1.18.0 (Ubuntu)
- **Packages** installed on webserver:
 - Python 3.8.10
 - PyMongo==4.3.3
 - python-dotenv==0.21.0
 - Flask 2.2.2
 - Gunicorn (version 20.0.4)
- **Database:** CosmosDB, using MongoDB API
- **Scripts:** Python and HTML/CSS

Application Setup

The Azure Anonymous application can be downloaded at:

<https://github.com/csidon/AzAnonMontygo>

Follow the installation steps in readme.txt

For Users with Azure Account Pre-Setup Access

This account access will only be available for a limited time period, and it is likely that you won't have access to this. If this is not something that you have access to, you will need to sign up for your own Azure Portal access and setup the infrastructure and technology stack detailed in the [Network Architecture Design](#) section. A full step-by-step guide on how to setup an Azure VM and install the application [can be found in the next section](#).

If you have been given access to an Azure account (instructor access until 1 Mar 2023 only):

1. Go to Resource Group **AzureAnonymous-dec2022-web-rg**
2. Start the VMs
3. **SSH** into the VMs (private key will be sent to you via Slack)
(Note that SSH port 22 is exposed to the Internet. This should not be the case in a production environment but has been set as such for easy setup)
4. Login as **azureuser**
5. Change directory into project file
cd AzAnonMontygo
6. Start gunicorn and set it to keep running in the background
(these steps need to be carried out each time the VM is restarted. Ideally this should be a script that runs automatically each time the VM is started up)
gunicorn --workers=3 runner:app --daemon
7. Open <<VM-IP-addresses>> in browser and check if it's working
For the Dec 2022 submission these IP addresses are:
vm1-az1: 40.82.200.169
vm2-az2: 20.70.235.122
main-static-ip: 20.213.168.157. This should receive internet traffic and load balance them amongst the two VMs.

Azure Resources

Our instructor will be given access to an Azure account that has already been setup with the following testing setup:

AzureAnonymous-dec22-web-rg

- 1 x captured Image of the latest version of AzureAnonymous
- 1 x SSH public key (private key can be provided on request)
- 1 x Virtual Network with:
 - a. 1 x Network Security Group configured for web servers
 - b. 1 public subnet (10.0.1.0/24)
 - c. 1 x Load Balancer, connected to:
 - i. 2 x VMs (identical), each with their own static IP addresses

cotiss-cosmos-mongo-rg

- 1 x Azure Cosmos DB for MongoDB account

Account Access (SSH, private key can be provided on request)

Database login as: azureuser

Password: NIL

Step-by-step guide to setting up AzureAnonymous VMs

For Users **without** CSidon Azure Account Access

This section has been created for developers and sysadmins (and future me) that don't have access to the Azure Account with a VM image already set up the application ready to go, and only have access to the Github repository: <https://github.com/csidon/AzAnonMontygo>

Create an Azure Virtual Machine

Create a virtual machine with the following settings:

Basics

Microsoft Azure

Search resources, services, and docs (G+)

Home > Create a resource >

Create a virtual machine

Basics | Disks | Networking | Management | Monitoring | Advanced | Tags | Review + create

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. [Learn more](#)

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *

School Basic Subscription

Resource group *

afterHolsTest

Create new

Create and use your own RG

Instance details

Virtual machine name *

clean-vm1-ause-az1

Enter a memorable VM name and select a region

Region *

(Asia Pacific) Australia East

Availability options

Availability zone

We want redundancy, so select Availability zone and pick a zone

Availability zone *

Zone 1

Security type

Standard

Image *

Ubuntu Server 20.04 LTS - x64 Gen2

See all images | Configure VM generation

This application has been built and tested on Ubuntu 20.04 LTS only

VM architecture

Arm64

☒ x64

Run with Azure Spot discount

☐

Size *

Standard_B1ls - 1 vcpu, 0.5 GiB memory (\$4.82/month)

See all sizes

Cheapest VM size chosen for this application. This might be scaled up depending on traffic loads.

Administrator account

Authentication type

☒ SSH public key

☐ Password

Assumes you already have created/access to an Azure SSH key

Username *

azureuser

SSH public key source

Use existing key stored in Azure

We want redundancy, so select Availability zone and pick a zone

Stored Keys

azanon-submissionkey

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports *

☐ None

☒ Allow selected ports

Select inbound ports *

SSH (22)

Enable for SSH access. Close this to allow only your IP address if in production mode

⚠ This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

Disks

Microsoft Azure

Search resources, services, and docs (G+/J)

Home > Create a resource >

Create a virtual machine

Basics **Disks** Networking Management Monitoring Advanced Tags Review + create

Azure VMs have one operating system disk and a temporary disk for short-term storage. You can attach additional data disks. The size of the VM determines the type of storage you can use and the number of data disks allowed. [Learn more](#)

VM disk encryption

Azure disk storage encryption automatically encrypts your data stored on Azure managed disks (OS and data disks) at rest by default when persisting it to the cloud.

Encryption at host ☐

Encryption at host is not registered for the selected subscription. [Learn more about enabling this feature](#)

OS disk

OS disk type ☐ Standard SSD (locally-redundant storage) ☒ Premium SSD (locally-redundant storage)

The selected VM size supports premium disks. We recommend Premium SSD for high IOPS workloads. Virtual machines with Premium SSD disks qualify for the 99.9% connectivity SLA.

Delete with VM ☒

Key management ☐ Platform-managed key ☐ Customer-managed key

Enable Ultra Disk compatibility ☐

Data disks for clean-vm1-ause-az1

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching	Delete with VM
Create and attach a new disk Attach an existing disk					

Advanced

Review + create

< Previous

Next : Networking >

This application is unlikely to have high IOPS workloads, so Standard SSD has been selected to optimise costs

Networking

Microsoft Azure

Search resources, services, and docs (G+/J)

Home > Create a resource >

Create a virtual machine

Create new

NIC network security group ☐ None ☒ Basic ☐ Advanced

Public inbound ports ☐ None ☒ Allow selected ports

Select inbound ports ☐ SSH (22) ☐ RDP (3389) ☐ Custom

This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

Delete public IP and NIC when VM is deleted ☒

Enable accelerated networking ☐ The selected VM size does not support accelerated networking.

Load balancing

You can place this virtual machine in the backend pool of an existing Azure load balancing solution. [Learn more](#)

Place this virtual machine behind an existing load balancing solution? ☒

Load balancing settings

- Application Gateway** is an HTTP/HTTPS web traffic load balancer with URL-based routing, SSL termination, session persistence, and web application firewall. [Learn more about Application Gateway](#)
- Azure Load Balancer** supports all TCP/UDP network traffic, port-forwarding, and outbound flows. [Learn more about Azure Load Balancer](#)

Load balancing options ☐ Azure load balancer ☐ Application Gateway

Select a load balancer ☐ public-web-loadbalancer ☐ private-web-loadbalancer

Review + create

< Previous

Next : Disks >

Select this option for easy deleting of VMs

This assumes you have set up a load balancing solution. Otherwise, please set one up using a guide such as: <https://learn.microsoft.com/en-us/azure/load-balancer/quickstart-load-balancer-standard-public-portal>

Settings Overview

Microsoft Azure

Search resources, services, and docs (G+I)

Home > Create a resource >

Create a virtual machine ...

Validation passed

Basics

Subscription	School Basic Subscription
Resource group	afterHoiTest
Virtual machine name	clean-vm1-ause-az1
Region	Australia East
Availability options	Availability zone
Availability zone	1
Security type	Standard
Image	Ubuntu Server 20.04 LTS - Gen2
VM architecture	x64
Size	Standard B1s (1 vcpu, 0.5 GiB memory)
Authentication type	SSH public key
Username	azureuser
Key pair name	azanon-submissionkey
Public inbound ports	SSH
Azure Spot	No

VM creation settings at a glance.

Note: This screenshot does not include VMs with a load balancing solution

Disks

OS disk type	Standard SSD LRS
Use managed disks	Yes
Delete OS disk with VM	Enabled
Ephemeral OS disk	No

Networking

Virtual network	(new) afterHoiTest-vnet
Subnet	(new) default (10.1.0.0/24)
Public IP	(new) clean-vm1-ause-az1-ip
Accelerated networking	Off
Place this virtual machine behind an existing load balancing solution?	No
Delete public IP and NIC when VM is deleted	Enabled

!! Make sure that networking settings allow HTTP and HTTPS traffic as well, since traffic from the internet will be hitting this site.

Configuring your VM

SSH into your VM

This assumes that you know how to use Putty to SSH into another machine

Microsoft Azure

Search resources, services, and docs (G+I)

Home > Create a resource > Virtual machines > clean-vm1-ause-az1

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Settings

Networking

Connect

Disks

Size

Microsoft Defender for Cloud

Advisor recommendations

Extensions > applications

Continuous delivery

Availability > scaling

Configuration

Identity

Properties

Locks

Operations

Bastion

Auto-shutdown

Backup

Disaster recovery

Updates

Inventory

Change tracking

Connect

Start

Restart

Stop

Capture

Delete

Refresh

Open in mobile

CU / PS

Feedback

Advisor (1 of 2): All network ports should be restricted on network security groups associated to your virtual machine ->

Essentials

Resource group (linked) > afterHoiTest

Status > Running

Location > Australia East

Subscription ID (linked) > 1c9388a6e-5b

Availability zone > 1

Tags (edit) > Click here to add tags

Properties

Monitoring

Caps

Virtual machine

Computer name

Health state

Operating system

Publisher

Plan

VM generation

VM architecture

Agent status

Agent version

Host group

Host

Proximity placement group

Colocation status

Capacity reservation group

Availability > scaling

Availability state

Putty Configuration

Category

Session

Logging

Terminal

Keyboard

Behaviour

Features

Window

Appearance

Behaviour

Translation

Selection

Colours

Connection

Data

Proxy

SSH

Serial

Telnet

Rlogin

SUPDUP

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address) > 20.11.32.188

Port > 22

Connection type: > SSH

Load, save or delete a stored session

Saved Sessions

Default Settings

CCO-aws

CCO-aws-db

CCO-aws-testwp

CCO-lightsail-depot

CCO-lightsail-depot-temp

CCO-depot-aws-db-test

Close window on exit: > Only on clean exit

About

Help

Open

Cancel

Operating system > Linux (ubuntu 20.04)

Size > Standard B1s

Public IP address > 20.11.32.188

Virtual network/subnet > afterHoiTest-vnet/default

DNS name > Not configured

Networking

Public IP address > 20.11.32.188

Public IP address (IPv6) > -

Private IP address > 10.1.0.4

Private IP address (IPv6) > -

Virtual network/subnet > afterHoiTest-vnet/default

DNS name > Configure

Size

Size > Standard B1s

vCPUs > 1

RAM > 0.5 GiB

Disk

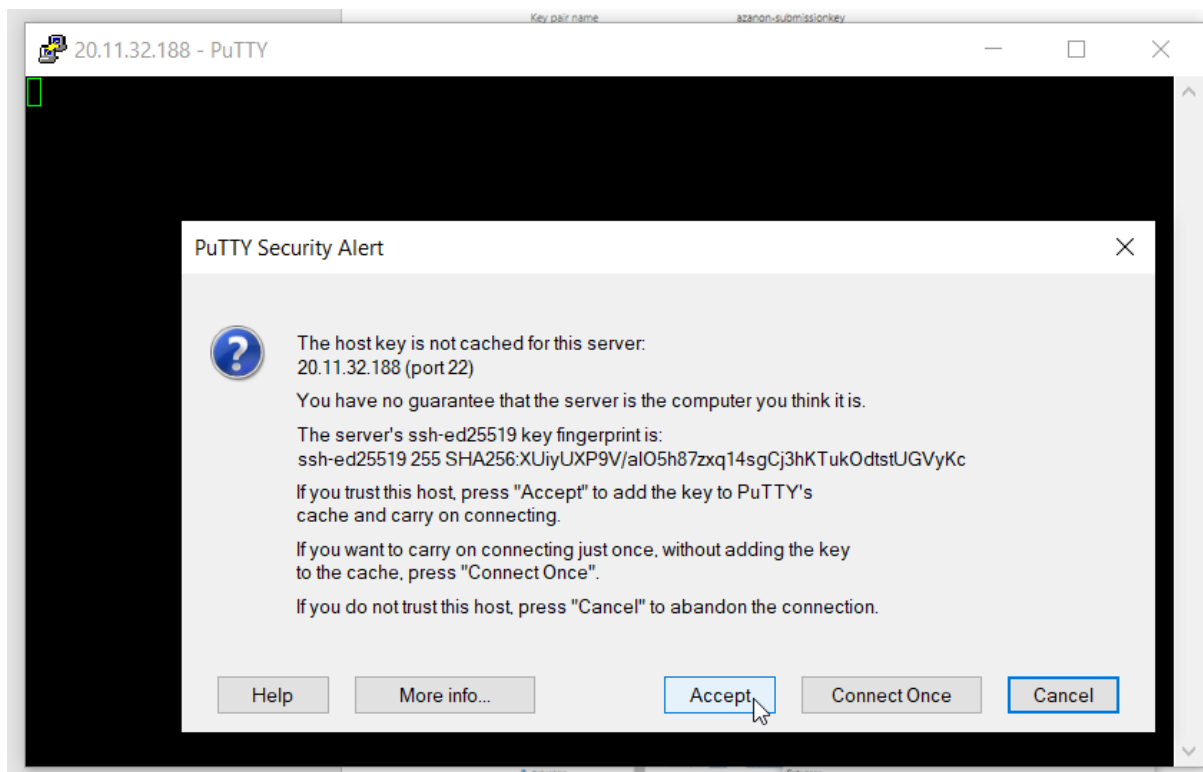
OS disk > clean-vm1-ause-az1_OsDisk_1_1706637652454580943ecc3a4667e4a

Encryption at host > Disabled

Azure disk encryption > Not enabled

Ephemeral OS disk > N/A

Data disks > 0



Login to VM and update/install packages

Login with your username (in this case, azureuser), then

>> sudo apt-get update

Install python, nginx and venv

>> sudo apt-get install python-pip3 -y

>> sudo apt-get install nginx -y

>> sudo pip3 install virtualenv -y

```
azureuser@clean-vm1-ause-az1: ~/AzAnonMontygo
Setting up libjpeg-turbo8:amd64 (2.0.3-0ubuntu1.20.04.3) ...
Setting up libjpeg8:amd64 (8c-2ubuntu8) ...
Setting up libnginx-mod-mail (1.18.0-0ubuntu1.4) ...
Setting up fontconfig-config (2.13.1-2ubuntu3) ...
Setting up libnginx-mod-stream (1.18.0-0ubuntu1.4) ...
Setting up libtiff5:amd64 (4.1.0+git191117-2ubuntu0.20.04.7) ...
Setting up libfontconfig1:amd64 (2.13.1-2ubuntu3) ...
Setting up libgd3:amd64 (2.2.5-5.2ubuntu2.1) ...
Setting up libnginx-mod-http-image-filter (1.18.0-0ubuntu1.4) ...
Setting up nginx-core (1.18.0-0ubuntu1.4) ...
Setting up nginx (1.18.0-0ubuntu1.4) ...
Processing triggers for ufw (0.36-6ubuntu1) ...
Processing triggers for systemd (245.4-4ubuntu3.19) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$ python-pip3 --version
python-pip3: command not found
azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$ pip3 --version
pip 20.0.2 from /usr/lib/python3/dist-packages/pip (python 3.8)
azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$ nginx --version
nginx: invalid option: "-"
azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$
```

Clone project, enter directory, create venv

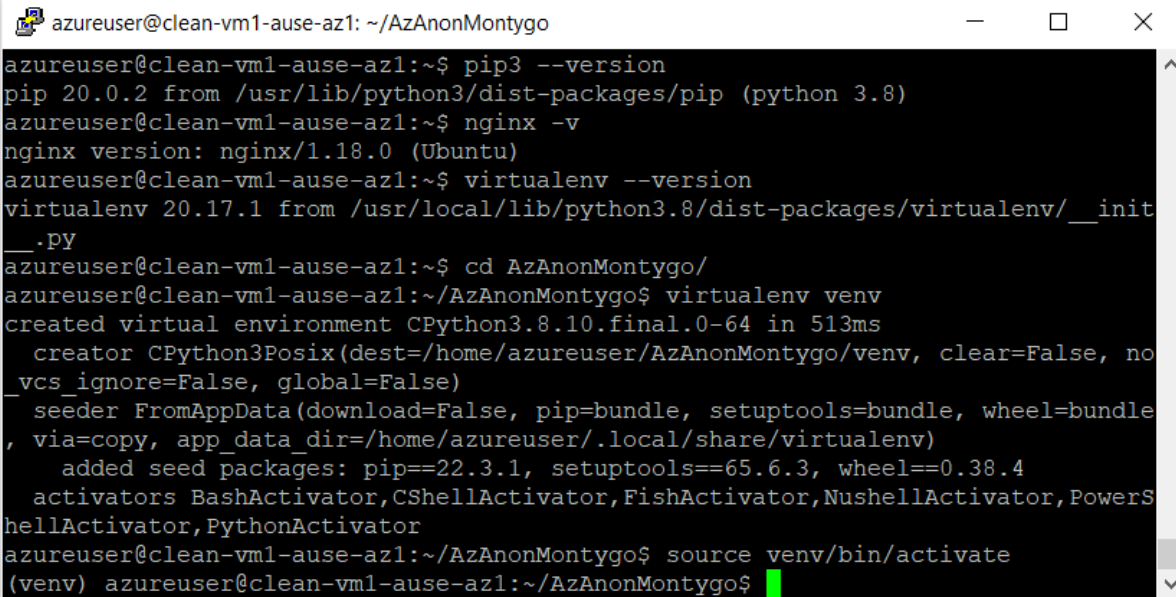
Clone the git project and cd into it, then create a virtual environment and activate it

```
>> git clone https://github.com/csidon/AzAnonMontygo.git
```

```
>> cd AzAnonMontygo
```

```
>> virtualenv venv
```

```
>> source venv/bin/activate
```



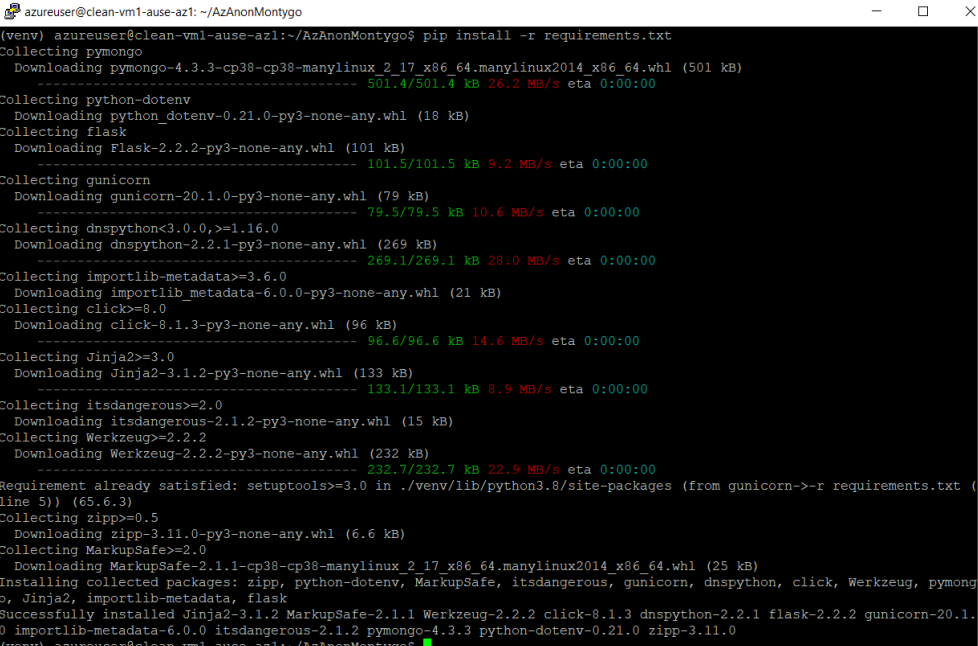
```
azureuser@clean-vm1-ause-az1: ~/AzAnonMontygo
azureuser@clean-vm1-ause-az1:~$ pip3 --version
pip 20.0.2 from /usr/lib/python3/dist-packages/pip (python 3.8)
azureuser@clean-vm1-ause-az1:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
azureuser@clean-vm1-ause-az1:~$ virtualenv --version
virtualenv 20.17.1 from /usr/local/lib/python3.8/dist-packages/virtualenv/___init___.PY
azureuser@clean-vm1-ause-az1:~$ cd AzAnonMontygo/
azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$ virtualenv venv
created virtual environment CPython3.8.10.final.0-64 in 513ms
  creator CPython3Posix(dest=/home/azureuser/AzAnonMontygo/venv, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/home/azureuser/.local/share/virtualenv)
    added seed packages: pip==22.3.1, setuptools==65.6.3, wheel==0.38.4
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$ source venv/bin/activate
(venv) azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$
```

Note: Use this article if you're running into venv installation issues

https://techoverflow.net/2022/02/03/how-to-fix-tox-attributeerror-module-virtualenv-create-via_global_ref-builtin-cpython-mac_os-has-no-attribute-cpython2macosarmframework/

Install requirements into venv

```
>> pip3 install -r requirements.txt
```

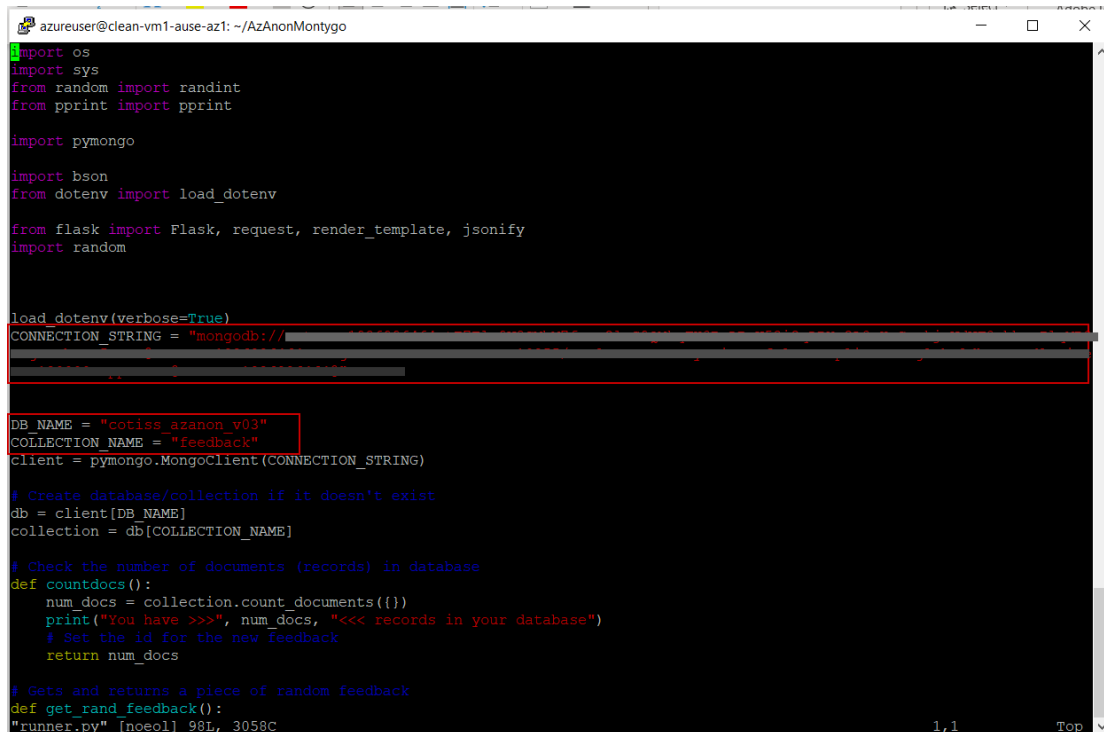


```
(venv) azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$ pip install -r requirements.txt
Collecting pymongo
  Downloading pymongo-4.3.3-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (501 kB)
----- 501.4/501.4 KB 26.2 MB/s eta 0:00:00
Collecting python-dotenv
  Downloading python_dotenv-0.21.0-py3-none-any.whl (18 kB)
Collecting flask
  Downloading Flask-2.2.2-py3-none-any.whl (101 kB)
----- 101.5/101.5 KB 9.2 MB/s eta 0:00:00
Collecting gunicorn
  Downloading gunicorn-20.1.0-py3-none-any.whl (79 kB)
----- 79.5/79.5 KB 10.6 MB/s eta 0:00:00
Collecting dnspython<3.0.0,>=1.16.0
  Downloading dnspython-2.2.1-py3-none-any.whl (269 kB)
----- 269.1/269.1 KB 28.0 MB/s eta 0:00:00
Collecting importlib-metadata>=3.6.0
  Downloading importlib_metadata-6.0.0-py3-none-any.whl (21 kB)
Collecting click>=8.0
  Downloading click-8.1.3-py3-none-any.whl (96 kB)
----- 96.6/96.6 KB 14.6 MB/s eta 0:00:00
Collecting Jinja2>=3.0
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
----- 133.1/133.1 KB 8.9 MB/s eta 0:00:00
Collecting itsdangerous>=2.0
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting Werkzeug>=2.2.2
  Downloading Werkzeug-2.2.2-py3-none-any.whl (232 kB)
----- 232.7/232.7 KB 22.9 MB/s eta 0:00:00
Requirement already satisfied: setuptools>=3.0 in ./venv/lib/python3.8/site-packages (from gunicorn->-r requirements.txt (line 5)) (65.6.3)
Collecting zipp>=0.5
  Downloading zipp-3.11.0-py3-none-any.whl (6.6 kB)
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.1.1-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25 kB)
Installing collected packages: zipp, python-dotenv, MarkupSafe, itsdangerous, gunicorn, dnspython, click, Werkzeug, pymongo, Jinja2, importlib-metadata, flask
Successfully installed Jinja2-3.1.2 MarkupSafe-2.1.1 Werkzeug-2.2.2 click-8.1.3 dnspython-2.2.1 flask-2.2.2 gunicorn-20.1.0 importlib-metadata-6.0.0 itsdangerous-2.1.2 pymongo-4.3.3 python-dotenv-0.21.0 zipp-3.11.0
(venv) azureuser@clean-vm1-ause-az1:~/AzAnonMontygo$
```

Amend DB details in runner.py

Open runner.py and replace CONNECTION_STRING, DB_NAME and COLLECTION_NAME with your database details, then run it

(This assumes that you have already created your MongoDB database, hosted on CosmosDB)



```
azureuser@clean-vm1-ause-az1: ~/AzAnonMontygo
import os
import sys
from random import randint
from pprint import pprint

import pymongo

import bson
from dotenv import load_dotenv

from flask import Flask, request, render_template, jsonify
import random

load_dotenv(verbose=True)
CONNECTION_STRING = "mongodb://[REDACTED]"

DB_NAME = "cotiss_azonon_v03"
COLLECTION_NAME = "feedback"
client = pymongo.MongoClient(CONNECTION_STRING)

# Create database/collection if it doesn't exist
db = client[DB_NAME]
collection = db[COLLECTION_NAME]

# Check the number of documents (records) in database
def countdocs():
    num_docs = collection.count_documents({})
    print("You have >>>", num_docs, "<<< records in your database")
    # Set the id for the new feedback
    return num_docs

# Gets and returns a piece of random feedback
def get_rand_feedback():
    "runner.py" [noeol] 98L, 3058C
1,1 Top
```

Test run the application

```
>> export FLASK_APP=runner.py
```

```
>> flask run -h 0.0.0.0
```

Follow the final steps in readme.txt to deploy for development server or production server.

Note: If you decide to stop/restart the VM, you will need to follow the steps in the [For Users with Azure Account Pre-Setup Access](#) section of this guide to get the application up and running again. Ideally this should be a script that runs automatically each time the VM is started up.

Network Architecture Design & Costing

This documentation does not explore the costs involved in scaling up or scaling out for traffic and focuses largely on the costs and availability of the network design.

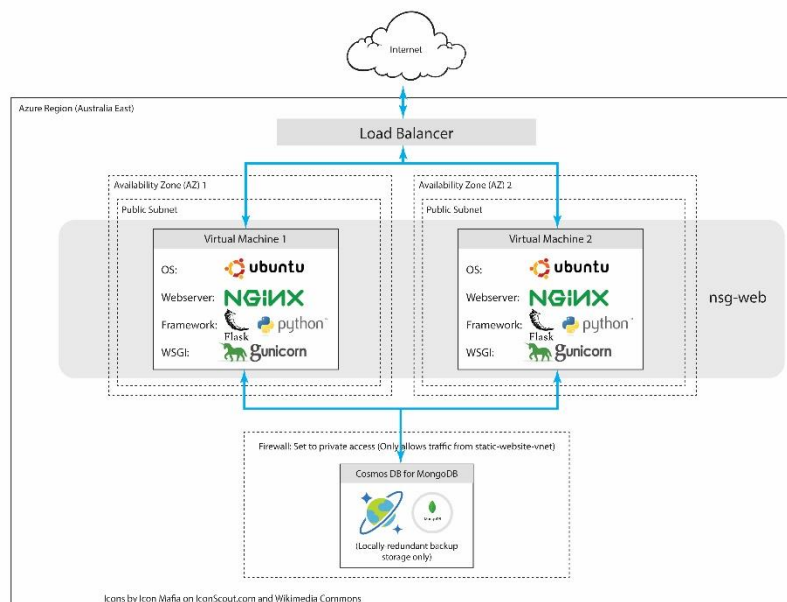
Designing for Minimum costs

Costing Estimate at \$68.14/month pay-as-you-go (PAYG):

<https://azure.com/e/c754f91991e74b8c8c19c453927e0d5c>

Note: If this is a low volume application and is shifted to serverless Database Operations, there is the potential for database costs to drop significantly.

We like networks to be able to self-heal and the brief specifies for us to explore a highly available setup, however, realistically, this application probably doesn't require a high SLA (if any) at the moment. The Azure account that you've been given access to is set up with the most basic of networks for relative high availability, and can be seen as follows:



The term “relative availability” is used because there are 2 VMs in 2 Availability Zones (AZs) that are load balanced, so there should always be at least one webserver up, but with only one database available. The reason for this decision is due to the type of data that is being collected - The data is not highly vital, doesn't contain any Personal Identifiable Data (PID), and the current brief does not specify any need to keep the data for long term evaluation and analysis. Because of this, it is reasonable to use the lowest backup and restore settings for CosmosDB to optimise costs. The database can be queried on a regular basis and cached locally in the VM.

Home > All resources > cotmon-1986996464

cotmon-1986996464 | Backup & Restore
Azure Cosmos DB for MongoDB account

Connection String

Features

Replicate data globally

Default consistency

Backup & Restore

Networking

Data Migration

Advisor Recommendations

Identity

Preview Features

Locks

Integrations

Backup policy mode (change)
Periodic

Backup Interval
How often would you like your backups to be performed?
 ☒ Minute(s)

Backup Retention
How long would you like your backups to be saved?
 ☒ Hours(s)

Copies of data retained 2

Backup storage redundancy * ⓘ
☐ Geo-redundant backup storage
☐ Zone-redundant backup storage
☒ Locally-redundant backup storage

Even though the database doesn't hold any vital data, it is still good practice to protect it from public access since that ensures that i.e. the data is harder to corrupt. Therefore the Public Network Access should be limited only to specific networks.

Microsoft Azure

Home > cotmon-1986996464

cotmon-1986996464 | Networking
Azure Cosmos DB for MongoDB account

Quick start

Notifications

Data Explorer

Settings

Connection String

Features

Replicate data globally

Default consistency

Backup & Restore

Networking

Data Migration

Advisor Recommendations

Identity

Preview Features

Save Discard

Public network access
☐ All networks ☒ Selected networks ☐ Disabled
Configure network security for your Azure Cosmos DB account. [Learn more.](#)

Virtual networks
Secure your Azure Cosmos DB account with virtual networks. [Add existing virtual network](#) [Add new virtual network](#)

Virtual Network	Subnet	Address range	Endpoint Status	Resource Group	Subscription
> static-website-vnet	1	10.1.0.0/16		static-website	School Basic Subscription ***

Firewall
Add IP ranges to allow access from the internet or your on-premises networks. [Add my current IP \(203.211.107.208\)](#) ⓘ

IP (Single IPv4 or CIDR range)

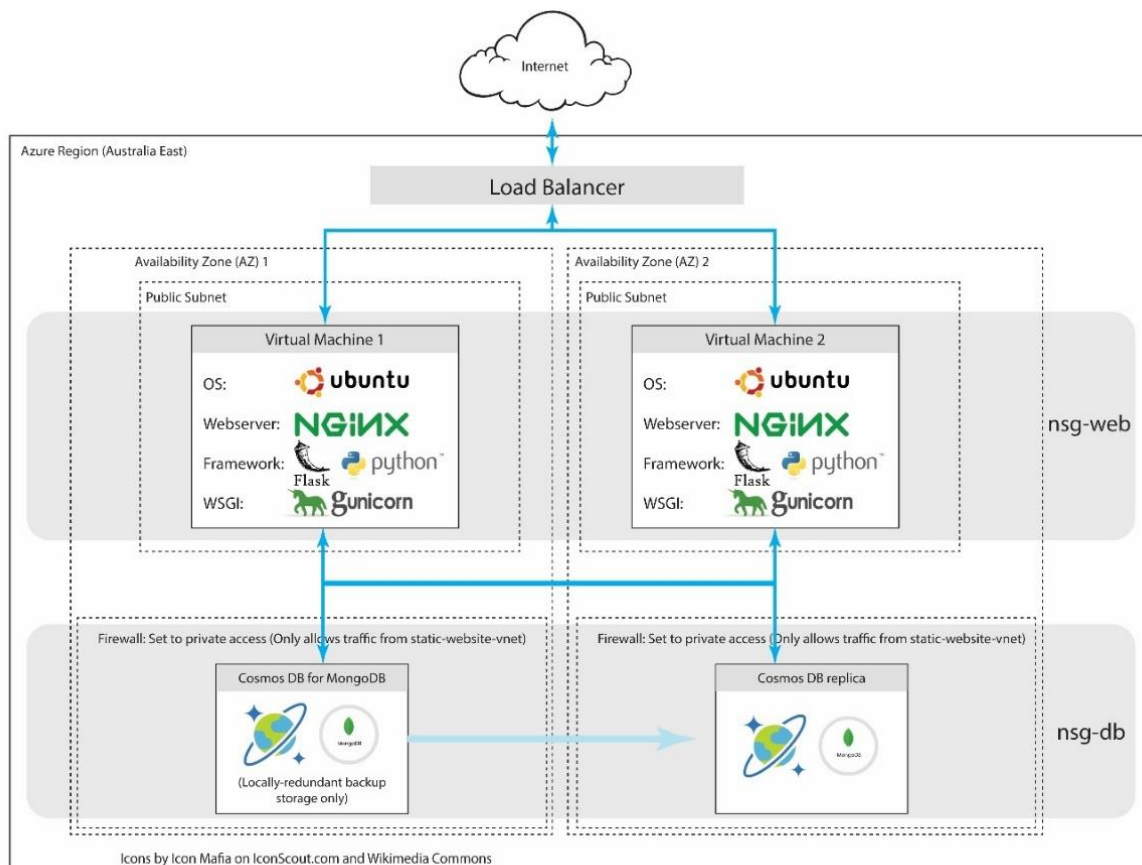
Exceptions
☐ Accept connections from within public Azure datacenters ⓘ
☒ Allow access from Azure Portal ⓘ

Designing for High Availability

Costing Estimate at \$116.04/month pay-as-you-go (PAYG):

<https://azure.com/e/f8132a011f0b44e48a6ef8605de7faee>

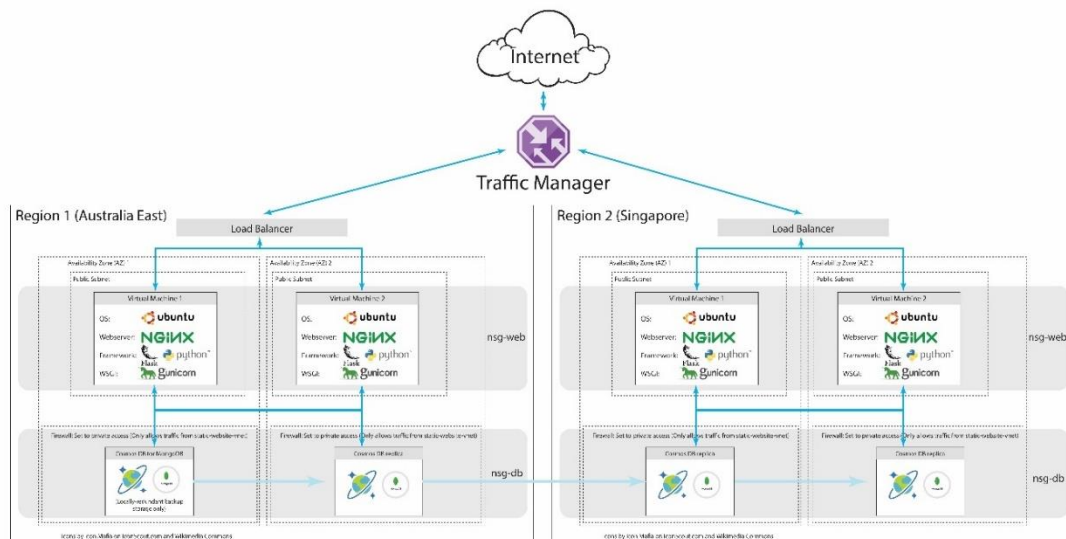
If there is a need for the applications database to be constantly available, then CosmosDB can be set up to allow regular replication within a region. This does almost double the cost though, as can be seen in the costing estimate above.



Designing for Maximum Availability

Costing Estimate at \$216.65/month pay-as-you-go (PAYG):

<https://azure.com/e/ace400f98a01405dacf4c99b06430a3b>



If for some reason a requirement arises over time where there's a need for high availability across regions, the cost again doubles. However, if this application reaches that level of need, it is likely to have far higher traffic and usage. In that case, the number of VMs per AZ would first be increased, so this costing would not be very accurate.