3 - Management Interfaces

## Learning Outcomes

After completing the exercises in this lab you should be able to

* Use the Azure Cloud management CLI to perform automated tasks in Azure
* Use the Python language and Azure Python library to carry out similar tasks

## Organisation

Please attempt this lab individually as you will need the skills acquired in this lab in order to complete later labs in this module.

## Grading

This worksheet is worth up to 10% of your overall module grade. You must attend and sign in at a minimum of 10 labs in the semester in order to obtain CA credit.

You may work on this worksheet during lab 3 and lab 4 with instructor assistance. You may also be requested to demonstrate your submission to the lab instructor in order to receive credit.

## Submission

The deadline for submission is Sunday Nov 04, 2018 @23:59 through Webcourses.

## Requirements

For this lab you will need to sign up for the following services if you’ve not already done so:

* An installation of the Python 3 language in your environment
* An MS Azure subscription

## Resources

You are free to research whatever you need to solve the problems in this lab. Some recommended resources include:

* <https://docs.microsoft.com/en-us/cli/azure/?view=azure-cli-latest>
* <https://docs.microsoft.com/en-us/python/api/overview/azure/?view=azure-python>

## Problem Sets

|  |  |  |
| --- | --- | --- |
| 1. | Follow the tutorial steps detailed here: <https://docs.microsoft.com/en-us/cli/azure/azure-cli-vm-tutorial?view=azure-cli-latest>  By the end of this you should have completed the following steps just using the CLI (no GUI):   * Installed the Azure CLI locally      * Connected your subscription to it and logged in      * Created a resource group      * Created one or more VMs in your cloud      * Be able to connect to that VM over SSH      * Be able to query VM properties using the Azure CLI          * Be able to set local env variables from Azure CLI output (e.g. network addresses)          * Be able to cleanup your cloud resources.     Snap screenshots of your progress at the milestone markers through the tutorial | 20  Marks |
| 2. | Now it’s time to put it altogether. In the following exercise you will make use of the skills from worksheets 1 and 2 and the Azure CLI. Take screenshots of your work as you reach each milestone   1. Create a simple Python Flask server which accepts connections on port 8080 to the URI “/status” and outputs the following JSON document with the computed values in red below.  |  | | --- | | {  “hostname”: “the machine’s host name here”,  “ip\_address”: “the machine’s IP address here”,  “cpus”: “the number of host CPUs here”,  “memory”: “the machine’s memory (in GBs) here”,  } |   Connect the cloudcomputing vm via port 80.  Check if python is installed. Install Flask     1. Test this server locally to make sure that it works 2. Commit your server code to Github (or Bitbucket) and take note of the repository name for later      1. Create a bash script to carry out the following tasks:    1. Create a Linux VM on Azure    2. Install the Python 3 environment on your VM    3. Install your Flask server on this VM (from Github)    4. Start up the remote flask server    5. Connect to the remote flask server (port 8080) and read the JSON document from the “/status” URI    6. Parse that output and print the results from your bash script      1. Create a second bash script to do the following    1. Query your created cloud resources (from this worksheet)    2. Tear them all down   HINT: You will need the Azure CLI (obviously), bash, ssh, curl, jq, awk, etc when putting this solution together | 45  Marks |
| 3 | Follow the tutorial steps details here:  <https://docs.microsoft.com/en-us/python/azure/python-sdk-azure-get-started?view=azure-python>  By the end of this you should have completed the following steps just using the Python SDK and the CLI (no GUI). Take screenshots of your work as you reach each milestone   * Install the Python SDK * Set up authentication        * Created a resource group (CLI)      * Created a VM (Python)   Create a virtual network and subnet    Create a public address    Create a network interface client    Linux\_VM.py  Check     * Created a Web App (Python)     Create free plan for the web    Python\_web.py   * Created a Database (Python)   Create a resource group    Create SQL server    Add firewall rule    Create SQL db    SQL.py   * Created a Storage Blob (Python)   Create resource group    Create a storage account    AccountStorage.py | 25  Marks |
| 4 | What are your reflections on using the CLI and Python management interfaces and how they compare?  Using python to virtual machine, web application and storage account you have pip install a few libraries to be able to actually run the code that is given. Using azure cli can be a little confusing at the beginning but once you get the hang out of it, it is not that complicated. Personally, I find cli easier to do stuff such as create a VM. | 10  Marks |

## 