Automated Cloud Services

Assignment 1

A. Core functionality required

The overall objective of this assignment is to use Python 3 to automate the process of creating, launching and monitoring a public-facing web server in the Amazon cloud. The web server will run on an EC2 instance and display some static content that is stored in S3. The program that does this must be called **newwebserver.py**

More detailed specification:

1. ***Launch EC2 instance***. Firstly, your Python program should create and launch a new Amazon EC2 *nano* instance. You must use the Boto3 API library to launch from an Amazon Linux 2 AMI. Use an up-to-date AMI version. You will need to have your API credentials in a configuration file (`~/.aws/credentials`) and not in the code.
2. ***Configure appropriate instance settings (at launch).*** Ensure that your program launches the instance into an appropriate security group (you can optionally create one programmatically) and that the instance is accessible using your SSH key.
3. ***EC2 start-up script.*** You should provide a *"User Data"* script when creating the instance. This start-up script should apply any required patches to the operating system and then install the web server (e.g. *Apache*).
4. ***Copy image to S3***. Another core requirement is that you write Python 3 code to create an S3 bucket and copy an image up to this bucket. This image is available at ***https://witacsresources.s3-eu-west-1.amazonaws.com/image.jpg***
5. The image at this URL will change from time to time, so your code will need to handle this in a generic manner.
6. ***Configure web server home page***. Your web server home page should be configured to display the following two items:
   - Some attribute(s) of the instance, for example availability zone or subnet. Use SSH remote command execution to use *curl* or equivalent to retrieve this information from *instance metadata.* You will need to use the public IP address or DNS name assigned to your instance to connect to it using SSH.
   - The above image, *loaded from your own S3 URL*. The HTML for your page will need to include something like
     `<img src=https://bucketname.s3-eu-west-1.amazonaws.com/image.jpg>`
7. ***Monitoring.*** A bash script called *monitor.sh* is provided that runs some sample basic monitoring commands. From your Python script, use *scp* (secure copy) to copy this script up onto your newly-created instance and then use SSH remote command execution to set the appropriate permissions and execute this script on the instance.

B. Non-functional issues: readability, robustness, user-friendliness

As well as for core functionality and testing, marks will be awarded for:

- **Robustness**. Your code should do appropriate error handling (using exceptions) and output meaningful messages to the user when errors and unexpected situations occur.
- **Logging**. Display to the console (and/or log to a file) details of what is happening, whether normal or errors.
- **Code readability.** It helps to have good code comments, appropriate code layout/spacing, and good variable and function names.
- **Testing**. Ensure that your code is well tested and debugged. You may optionally wish to use Python's *unittest* module or alternatives.

C. Additional functionality – to your own specification

The above is the core assignment specification. In addition you are expected to explore one or more other tasks. We are leaving it up to your individual initiative to decide what to do, but the following would be reasonable examples:

- Configure other AWS services remotely
- Pass parameters as command line arguments
- Query web server access/error logs, possibly using *grep* or equivalent
- Use boto3 to monitor instance metrics such as CPU utilisation using CloudWatch

N.B. Do not implement an elaborate menu/UI. The program is meant to demonstrate cloud services automation with minimal user interaction. However it is ok to have some user prompts – for example at the start you might wish to prompt for key name, bucket name, etc.

Assignment submission:

You are required to submit a single **zip** archive containing the following:

1. Your Python program(s)
2. Completed *Assignment Review spreadsheet* provided.

Assignment review:

- A Zoom call (approx. 10 mins) will be scheduled for each student to provide a short demonstration of your submission and answer some questions. These calls will take place during Week 7 and a schedule will be published closer to the time.

Marking scheme:

- 60% Core functionality – as specified
- 10% Additional functionality – to your own specification (at least one "extra")
- 15% Non-functional issues including testing, code quality, robustness, etc
- 15% Overall coherence and clarity

Note on working with EC2 for your assignment:

You should not need to save any work on an Amazon EC2 instance while working on this assignment. You should create and maintain your scripts locally and also take backups for yourself. Thus any termination of your EC2 instances should not affect your work.

Warning re originality of code provided:

It is **very** important to provide a proper citation/reference for any code that appears in your submission that is not entirely written by you. Any such code should be used very sparingly. It is much better to write a short program from scratch that meets the specification than to submit something long and elaborate if that means using substantial chunks of code or a design that is not your own.