Lightkeeper DevOps Code Project Guide

Table of Contents

Introduction

Congratulations on making it to the code project interview stage! We're excited you have decided to continue to move forward with the coding assignment and technical interview session! We look forward chatting with you soon about your work and yourself. If you have any questions about the code project, then please contact us at ops@lightkeeper.com.

Project Summary

The code project is is designed to learn how you solve the technical challenge of deploying a web app to the cloud. You will be using Amazon Web Services (AWS), Linux, Git, and Python as a take-home project.

In this project, you will create a (or use an existing) personal AWS account to provision an EC2 micro-sized Linux instance, install dependencies on the virtual machine, host the web app's source code using your git repository, and deploy the web app using a Python script.
Before running the deployment Python script, you must finish writing the code in specific functions. The empty functions include TODO statements in their bodies. Finishing the code in

the script should provision an EC2 instance in an AWS and setup the web app on the server pulled from your git repository. We will have a scheduled review session to discuss your solution and technical challenges.

Use the rest of the guide to help you get started on the project. Good luck!

Requirements

1. AWS account
2. SSH keys
3. Git repository
4. Python installed on your local machine.
5. The *charty* folder.
6. The deployment script, *create_ec2_instance_with_charty_problem.py*

AWS Account Setup

If you are new to AWS or don't have an account, you can create one at https://aws.amazon.com/. Follow AWS's setup to complete your user account. Once you have completed your AWS account, You will need to have an *AWS Access Key ID* and *Secret Key* created in the IAM Console. Keep your AWS credentials private and in a secure location for your use only. The keys will be used to access your AWS account to provision the EC2 instance to host the web app. Configuring your AWS keys is illustrated more in the *AWS CLI Setup* section.

Amazon's guide on creating your AWS Access Key:
https://aws.amazon.com/premiumsupport/knowledge-center/create-access-key/

AWS does require a credit card when EC2 instances are running. Charges are hourly and vary in rate depending on the size and type of EC2 instance. To avoid being charged, you will be provisioning a T2.micro free tier EC2 instance. The instance will be using an Ubuntu Server 16.04 LTS image. The Amazon Machine Image (AMI) is already provided for you in the deployment script.

```
# AMI ID of Ubuntu 16.04 LTS
AMI = 'ami-95a977ea'
```

If you need to terminate an instance, then you can use the AWS CLI or AWS Console. Amazon provides a guide for terminating EC2 instances.

Here is the link for Amazon's Getting Started Guide if you like a better understanding EC2 works with Linux virtual machines:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html

AWS CLI Setup

On your machine, ensure AWS CLI is installed. You will use the command-line tool to configure your AWS credentials. In your machine's terminal using the AWS CLI tools, run the following command to configure your AWS Access Key ID and Secret Key.

```
$ aws configure
AWS Access Key ID [None]: {YOUR_KEY_ID}
AWS Secret Access Key [None]: {YOUR_ACCESS_KEY}
Default region name [None]: us-east-1
Default output format [None]: {PRESS ENTER}
```

Your AWS credentials and config will be stored in ~/.aws.

For more information about AWS CLI:
https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-welcome.html

Setup a Git Repository

We'll use Git to manage the source code of the web app, *charty*. You can use GitHub, GitLab, BitBucket, or your own server as a git hosting provider. SSH keys will be used to access the git repository hosting *charty* from the Ubuntu server.

If you don't have an existing SSH key pair to use, then you will need to generate the SSH public and private key. GitHub has a setup guide for creating SSH keys.

Add the charty directory to your remote git hosting provider, and ensure you have SSH access to the repository using your key pair. On your selected git hosting provider, add your public SSH key. The private SSH key should remain on your local machine and placed on to the EC2 Linux instance. There is a function the deployment script for you to finish writing Python code to copy over the private SSH key in a secure location to clone the repository.

Official guide on how to use Git:
https://git-scm.com/docs/gittutorial

Setup Project Dependencies

Before running the deployment script, *create_ec2_instance_with_charty_problem.py*, you will need to install a few Python packages. The modules are dependent of the deployment script. Ensure you have the listed Python packages:

- Pip - Used for installing Python packages.
- Boto3 - AWS SDK in Python
- Paramiko - Python interface for SSH.

Boto3 and Paramkio are included in *create_ec2_instance_with_charty_problem.py.*

```
import boto3
import paramiko
```

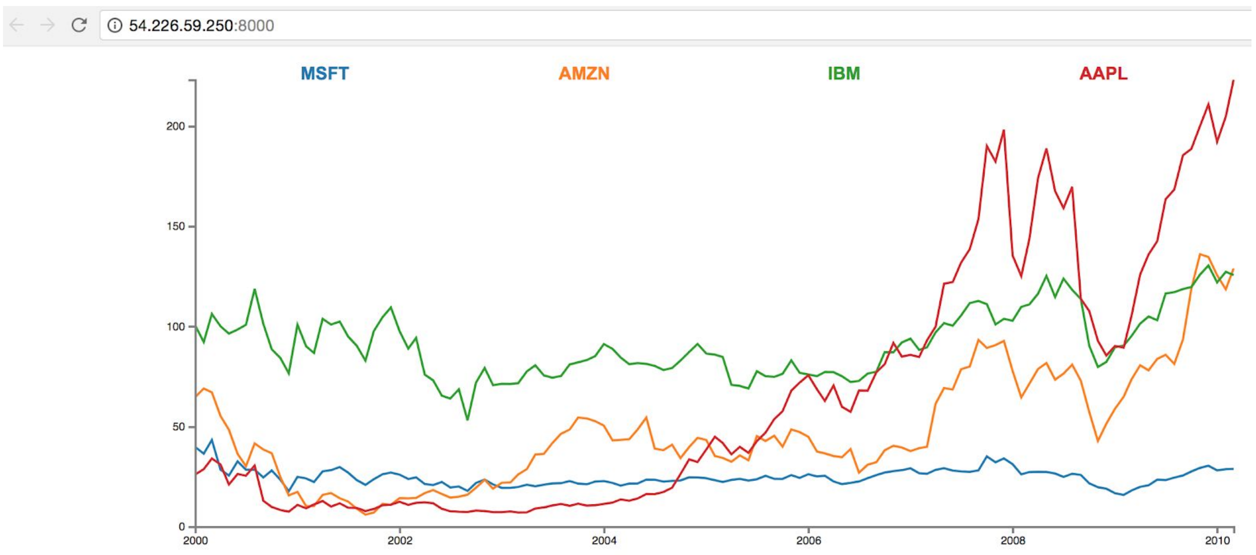
You will be referencing their APIs when writing your code.

About the Web App

The web app is called *charty*. It is a simple web app used to visualize a multi-line graph of popular tech stock tickers of public data for years, 2000 - 2010. The only input is toggling the display of each line in the graph. The stock data is rendered from *charty/static/stocks.csv*.

The app is built using Python Flask, Node.js,and D3.js. Yarn handles setting up the JavaScript components and systemd manages the Flask server as a daemon on the Ubuntu Server.

In the **charty** folder, all the necessary files are provided for you. You do not have to make any changes to the app's code. The README file provides more information on installing and running the web app.



Screenshot of charty.

Expected Results

On the scheduled date to review the code and demo of your solution, the project should produce the following results:

1. Running *create_ec2_instance_with_charty_problem.py* should
    a. Provision an EC2 t2.micro instance using the Ubuntu Server 16.04 LTS AMI.
    b. Clone the web app, *charty*, on the server.
    c. Configure and run the web app.
2. Be able to view the web app in the browser, *http://<your_ec2_instance_ip_address>:8000*
3. Be able to discuss how you approached the problem.
4. Be able to walkthrough the code.