IAM Users, Roles & Policies

Add all IAM roles & policies needed to meet the assignment objectives to your CloudFormation template.

Create Load Balancer Security Group

* Create a security group for the load balancer to access the web application.
* Add ingress rule to allow TCP traffic on ports 80, and 443 from anywhere in the world.
* This security group will be referred to as the load balancer security group.

App Security Group Updates

* Update the EC2 security group for your EC2 instances that will host web applications.
* The ingress rule should allow TCP traffic on ports 22 and port on which your application runs.
* The Source of the traffic should be the load balancer security group.
  + Restrict access to the instance from the internet.

The Auto Scaling Application Stack

So far our web application has been accessible by the IP address of the EC2 instance on HTTP protocol. We will now disable direct access to our web application using the IP address of the EC2 instance. The web application will now only be accessible from the load balancer.

Setup Autoscaling for EC2 Instances

Instead of launching standalone EC2 instances, we are now going to launch them in an auto-scaling group with a minimum of **1**instance and a maximum of **5**. Use the default (or customize them as per your preference) for properties not listed below.

Launch Configuration

| **Key** | **Value** |
| --- | --- |
| ImageId | **Your custom AMI** |
| Instance Type | t2.micro |
| KeyName | *YOUR\_AWS\_KEYNAME* |
| AssociatePublicIpAddress | True |
| UserData | *SAME\_USER\_DATA\_AS\_CURRENT\_EC2\_INSTANCE* |
| IAM Role | *SAME\_AS\_CURRENT\_EC2\_INSTANCE* |
| Resource Name | asg\_launch\_config |
| Security Group | **WebAppSecurityGroup** |

Auto Scaling Group

**Note:** You should add [tags](https://docs.aws.amazon.com/AutoScaling/latest/DeveloperGuide/ASTagging.html)([AutoScalingGroup TagProperty](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-as-tags.html)) to the EC2 instances in your Auto Scaling Group.

| **Parameter** | **Value** |
| --- | --- |
| Cooldown | 60 |
| LaunchConfigurationName | asg\_launch\_config |
| MinSize | 3 |
| MaxSize | 5 |
| DesiredCapacity | 3 |

AutoScaling Policies

Create the following autoscaling policies:

1. Scale up policy when average CPU usage is above 5%. Increment by 1.
2. Scale down policy when average CPU usage is below 3%. Decrement by 1.

Setup Application Load Balancer For Your Web Application

* EC2 instances launched in the auto-scaling group should now be load balanced.
* Add a balancer resource to your CloudFormation template.
* Setup Application load balancer to accept HTTP traffic on port 80 and forward it to your application instances on whatever port it listens on.
* Attach the load balancer security group to the load balacer.

DNS Updates

* Route53 should be updated from the CloudFormation template.
* Route53 resource record for your domain name should now be an alias for your load balancer application.
* The CloudFormation template should configure Route53 so that your domain points to your load balancer and your web application is accessible thru **http://your-domain-name.tld/**.
* Your application must be accessible using root context i.e. [**https://your-domain-name.tld/**](https://your-domain-name.tld/) and not [**https://your-domain-name.tld/app-0.1/**](https://your-domain-name.tld/app-0.1/)

Create JMeter Load Testing Script (optional)

* Using [Apache JMeter](http://jmeter.apache.org/) create tests that can be run against your application APIs.
* JMeter tests need to make 500 concurrent API calls to your application.
* JMeter tests should be stored in your application GitHub repository.

Documentation

Meter

* [JMeter User Manual](https://jmeter.apache.org/usermanual/index.html)
* [How To Use Apache JMeter To Perform Load Testing on a Web Server](https://www.digitalocean.com/community/tutorials/how-to-use-apache-jmeter-to-perform-load-testing-on-a-web-server)
* [How To Use Apache JMeter To Perform Load Testing on a Web Server](https://www.digitalocean.com/community/tutorials/how-to-use-apache-jmeter-to-perform-load-testing-on-a-web-server)
* [Fill Forms and Submit With JMeter](https://dzone.com/articles/fill-forms-and-submit-with-jmeter-made-easy)
* [Using CSV DATA SET with JMeter](https://guide.blazemeter.com/hc/en-us/articles/206733689-Using-CSV-DATA-SET-CONFIG)
* <https://www.blazemeter.com/blog/how-use-http-basic-authentication-jmeter/>

Load Testing & Autoscaling

* [Sample Auto Scaling CloudFormation Template](https://s3-us-east-2.amazonaws.com/cloudformation-templates-us-east-2/AutoScalingMultiAZWithNotifications.template)
* [Auto Scaling AutoScalingGroup TagProperty](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-as-tags.html)
* [AWS::Route53::RecordSet CloudFormation](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-route53-recordset.html)
* [Load Testing Web Servers with Siege](https://linode.com/docs/tools-reference/tools/load-testing-with-siege/)
* <https://docs.aws.amazon.com/autoscaling/ec2/userguide/autoscaling-load-balancer.html>

Submission

The assignment will be considered late if commits are made to the **main** branch after the due date.

1. All work for this assignment must be done on the feature branch in your fork and merged to main when you are dev complete.
2. The feature and main branches must be in sync.
3. Submit your code from **all** repositories in this assignment. **Read the instructions carefully to create your zip file.**
   1. Create a folder with the naming convention **firstname\_lastname\_neuid**
   2. In the **firstname\_lastname\_neuid**clone all of your GitHub (organization) repositories with the **git clone**command. It is important that you clone the repos so that your commit history and branches are preserved.
   3. Once you have cloned all of your repositories, you will create a zip of the **firstname\_lastname\_neuid\_a\_##** directory. The zip file should be **firstname\_lastname\_neuid\_a\_##.zip** where **##** is the assignment number.
   4. Now unzip the zip file in some other directory and confirm the content of the zip files.
   5. Upload the Zip to this assignment.
4. You are allowed to resubmit. If you think there may be an issue with the ZIP file, feel free to submit it again. Only the latest will be used grading.

Grading Guidelines

The following guidelines are for information only. They are subject to change at the discretion of the instructor and TA.

10% penalty will be imposed if the application crashes due to unhandled exceptions/errors.

Previous Assignment Objectives

* TAs must verify that students have resolved all open issues from the previous assignment(s).

Git Repository Content Check (10% Penalty)

* Check the repository for any IDE-specific files. IDE configuration files must not be in the repository.
  + Verify their .gitignore configuration.
* Check the repository for build artifacts such as .class, .jar, .war files and build, node\_modules directory. None of these should be checked into the repository.
* Check for dependencies. Dependencies from the Maven repository or npm should not be committed to the git repository.

Pre-requisite for Demo (5% Penalty)

* Delete all VPCs you may have created in your demo AWS account.

Application Crash (10% Penalty)

* The application should not crash or return *500 Internal Server Errors* during the demo.