

# **COMPX341-22A Assignment 4**

## **Quality Assurance via CI/CD on the cloud**

**Prepared by: Kevin Han 1521885**

### **Table of Content**

## **Contents**

1. Overview.....	2
2. Task 1: Preparing your Cloud9 IDE.....	2
3. Task 2: Initializing AWS Serverless Architecture Model .....	6
4. Task 3: Deploy Locally .....	9
5. Task 4: Configuring your repository.....	11
6. Task 5: Implementing a CI/CD pipeline .....	13
7. Task 6: Implementing Functionality and Testing.....	16
8. Conclusion .....	21

# 1. Overview

The purpose of the report is to document the progress of the steps to understand the importance of AWS and DevOps. The outcome of is to be able to utilise DevOps services on the cloud and apply DevOps and testing processes to deploy applications on the cloud and automate a continuous deployment pipeline. Also show that errors will not be push to repository stored on the cloud/system.

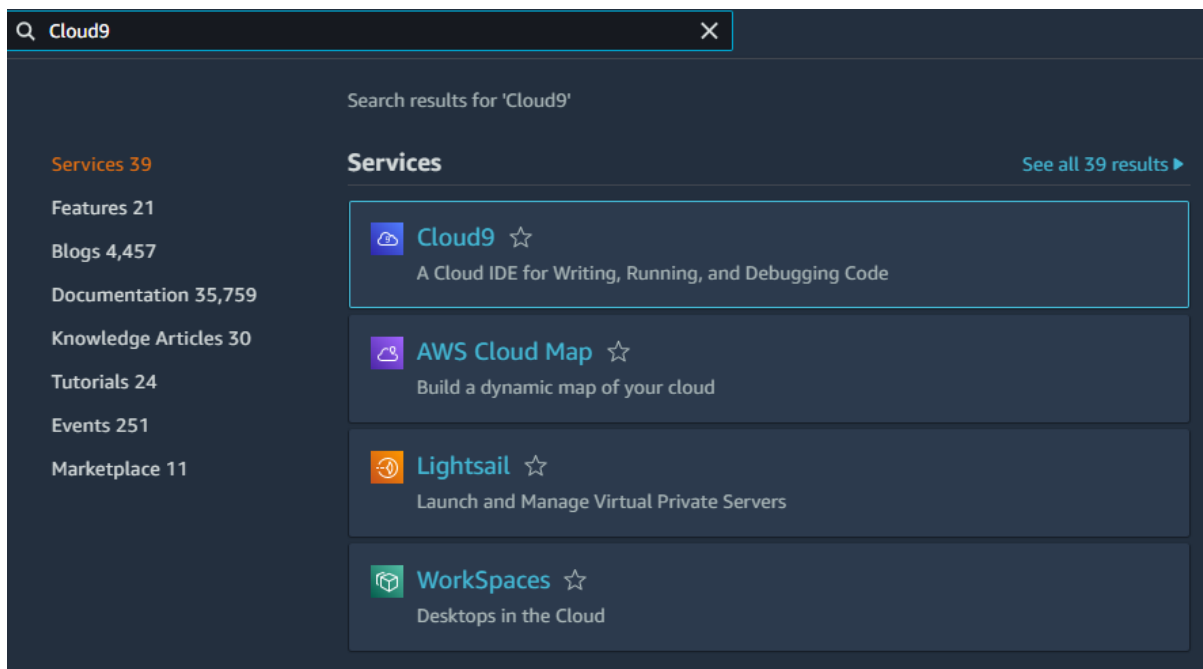
We will be using the AWS (<https://awsacademy.instructure.com/>) to demonstrate my understanding of the task provided.

## 2. Task 1: Preparing your Cloud9 IDE

The task is to initialise the environment for the Cloud9 IDE, to have access the IDE from any logged-in AWS console.

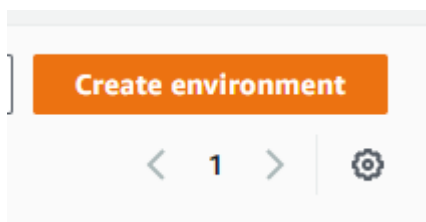
Step 1: Search for Cloud9 in the AWS Console as shown in Fig 1.

Fig 1



Step 2: Click on the create environment as shown in Fig 2.

Fig 2



Step 3: Set-up the environment name based on the assignment specification. The name should be like “your\_name-studentID-assignment4”. It should look like Fig 3.

Fig 3

**Environment name and description**

**Name**  
The name needs to be unique per user. You can update it at any time in your environment settings.

KevinHan-1521885-assignment4

Limit: 60 characters

**Description - Optional**  
This will appear on your environment's card in your dashboard. You can update it at any time in your environment settings.

Write a short description for your environment

Limit: 200 characters

Cancel **Next step**

Step 4: The application will not be using any special services, so the setting can be left as default. It should look like Fig 4-5.

Fig 4

**Environment settings**

**Environment type** [Info](#)  
Run your environment in a new EC2 instance or an existing server. With EC2 instances, you can connect directly through Secure Shell (SSH) or connect via AWS Systems Manager (without opening inbound ports).

- ☒ **Create a new EC2 instance for environment (direct access)**  
Launch a new instance in this region that your environment can access directly via SSH.
- ☐ **Create a new no-ingress EC2 instance for environment (access via Systems Manager)**  
Launch a new instance in this region that your environment can access through Systems Manager.
- ☐ **Create and run in remote server (SSH connection)**  
Configure the secure connection to the remote server for your environment.

**Instance type**

- ☒ **t2.micro (1 GiB RAM + 1 vCPU)**  
Free-tier eligible. Ideal for educational users and exploration.
- ☐ **t3.small (2 GiB RAM + 2 vCPU)**  
Recommended for small-sized web projects.
- ☐ **m5.large (8 GiB RAM + 2 vCPU)**  
Recommended for production and general-purpose development.
- ☐ **Other instance type**  
Select an instance type.

t3.nano ▼

Fig 5

**Platform**

☒ Amazon Linux 2 (recommended)

☐ Amazon Linux AMI

☐ Ubuntu Server 18.04 LTS

**Cost-saving setting**

Choose a predetermined amount of time to auto-hibernate your environment and prevent unnecessary charges. We recommend a hibernation settings of half an hour of no activity to maximize savings.

After 30 minutes (default) ▼

**IAM role**

AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. [Learn more](#)

AWSServiceRoleForAWSCloud9

► **Network settings (advanced)**

No tags associated with the resource.

Add new tag

You can add 50 more tags.

Cancel Previous step Next step

Step 5: Click on the create environment button to create the environment.

Fig 6

**Environment name and settings**

Name

KevinHan-1521885-assignment4

Description

No description provided

Environment type

EC2

Instance type

t2.micro

Subnet

Platform

Amazon Linux 2 (recommended)

Cost-saving settings

After 30 minutes (default)

IAM role

AWSServiceRoleForAWSCloud9 (generated)

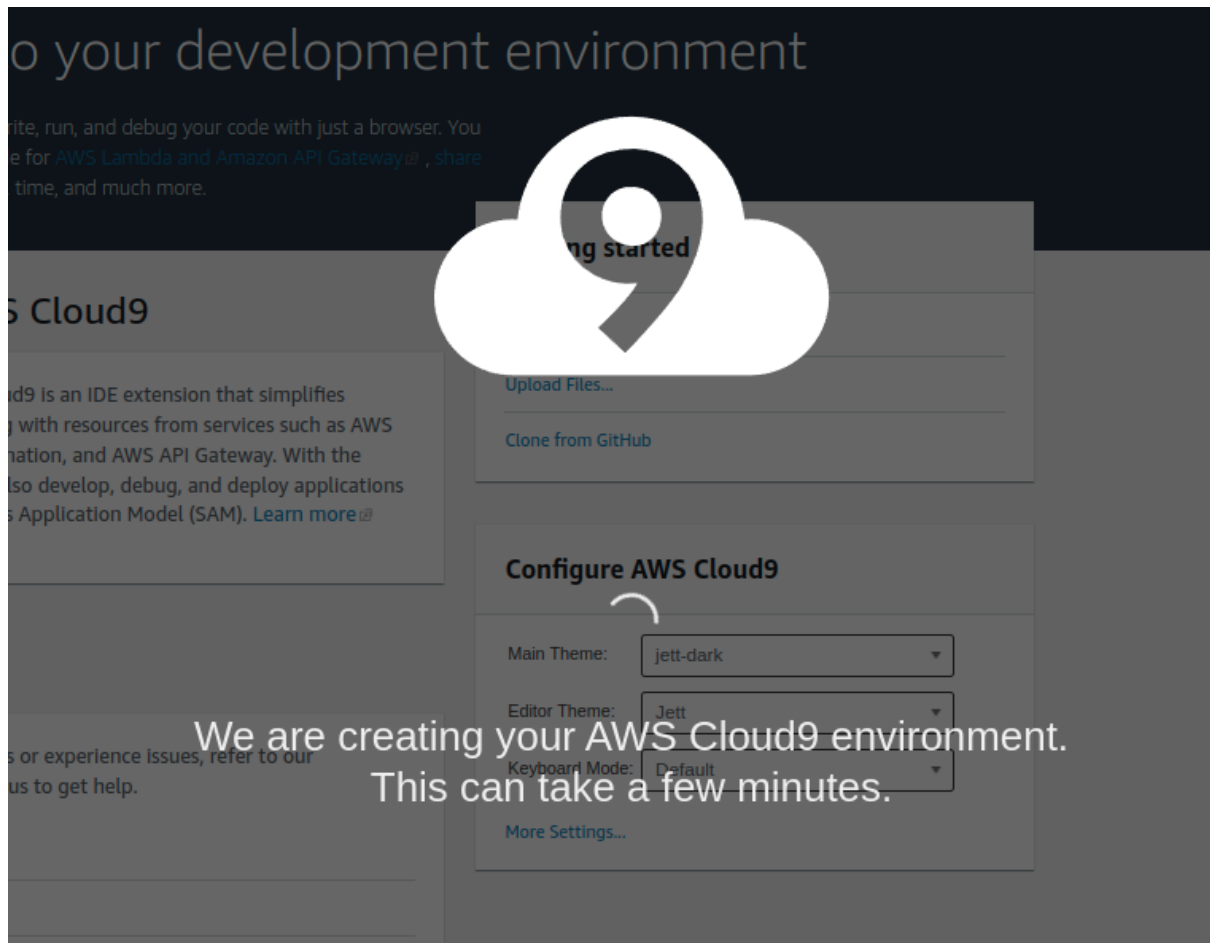
**We recommend the following best practices for using your AWS Cloud9 environment**

- Use **source control and backup** your environment frequently. AWS Cloud9 does not perform automatic backups.
- Perform regular **updates of software** on your environment. AWS Cloud9 does not perform automatic updates on your behalf.
- **Turn on AWS CloudTrail in your AWS account** to track activity in your environment. [Learn more](#)
- Only share your environment with **trusted users**. Sharing your environment may put your AWS access credentials at risk. [Learn more](#)

Cancel Previous step Create environment

After clicking the button, it will show the screen based on the Fig 7. This shows that the AWS environment is currently being created.

Fig 7



Open the environment and type “aws sts get-caller-identity” in the console. This is to ensure that your account is signed in. An example of is shown in Fig 8.

Fig 8

```
voclabs:~/environment $ aws sts get-caller-identity
{
  "Account": "505253522687",
  "UserId": "AROAXLI3UST7TAT2I5ETL:user1997276=Kevin_Han",
  "Arn": "arn:aws:sts::505253522687:assumed-role/voclabs/user1997276=Kevin_Han"
}
```

### 3. Task 2: Initializing AWS Serverless Architecture Model

The task is to initialise the AWS Serverless Architecture Model to the latest version within the Cloud9 IDE. We will be using a preset template provided to initialise the “AWS Hello World Example”.

Step 1: Run the following commands.

```
wget https://cicd.serverlessworkshops.io/assets/bootstrap.sh

chmod +x bootstrap.sh

./bootstrap.sh
```

The following command is to get the latest version, then allowing the program to be executable, then run the bootstrap script. It should look like Fig 9-10 when you run the command.

Fig 9

```
voclabs:~/environment $ wget https://cicd.serverlessworkshops.io/assets/bootstrap.sh
--2022-05-31 03:19:32-- https://cicd.serverlessworkshops.io/assets/bootstrap.sh
Resolving cicd.serverlessworkshops.io (cicd.serverlessworkshops.io)... 13.32.153.110, 13.32.153.126, 13.32.153.77, ...
Connecting to cicd.serverlessworkshops.io (cicd.serverlessworkshops.io)[13.32.153.110]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2143 (2.1K) [application/x-sh]
Saving to: 'bootstrap.sh'

100%[=====]
2022-05-31 03:19:32 (63.2 MB/s) - 'bootstrap.sh' saved [2143/2143]
```

Fig 10

```
+ _logger '[+] Updating SAM...'
++ date
+ echo -e 'Tue May 31 03:21:26 UTC 2022 \033[1;33m[+] [+] Updating SAM... \033[0m'
Tue May 31 03:21:26 UTC 2022 [*] [+] Updating SAM...
+ sudo ./sam-installation/install --update
You can now run: /usr/local/bin/sam --version
+ _logger '[+] Updating Cloud9 SAM binary'
++ date
+ echo -e 'Tue May 31 03:21:28 UTC 2022 \033[1;33m[+] [+] Updating Cloud9 SAM binary \033[0m'
Tue May 31 03:21:28 UTC 2022 [*] [+] Updating Cloud9 SAM binary
++ which sam
+ ln -sf /usr/local/bin/sam /home/ec2-user/.c9/bin/sam
+ cleanup
+ [[ -d sam-installation ]]
+ rm -rf sam-installation
+ echo -e '\033[0;31m [!!!!!!] To be safe, I suggest closing this terminal and opening a new one! \033[0m'
[!!!!!!] To be safe, I suggest closing this terminal and opening a new one!
+ _logger '[+] Restarting Shell to reflect changes'
++ date
+ echo -e 'Tue May 31 03:21:28 UTC 2022 \033[1;33m[+] [+] Restarting Shell to reflect changes \033[0m'
Tue May 31 03:21:28 UTC 2022 [*] [+] Restarting Shell to reflect changes
+ exec /bin/bash
```

The following command “sam --version” shows the current version of the Serverless Architecture Model. At this current moment, the latest version is show in the Fig 11.

Fig 11

```
voclabs:~/environment $ sam --version
SAM CLI, version 1.50.0
```

Step 2: Initialize the Serverless Architecture Model by running the command “sam init”.

This is to initialize the Serverless Architecture Model and we want to use the pre-set hello world template.

Follow the choice as shown in the Fig 12-13.

Fig 12

```
voclabs:~/environment $ sam init

SAM CLI now collects telemetry to better understand customer needs.

You can OPT OUT and disable telemetry collection by setting the
environment variable SAM_CLI_TELEMETRY=0 in your shell.
Thanks for your help!

Learn More: https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/serverless-sam-telemetry.html

You can preselect a particular runtime or package type when using the 'sam init' experience.
Call 'sam init --help' to learn more.

Which template source would you like to use?
  1 - AWS Quick Start Templates
  2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
  1 - Hello World Example
  2 - Multi-step workflow
  3 - Serverless API
  4 - Scheduled task
  5 - Standalone function
  6 - Data processing
  7 - Infrastructure event management
  8 - Machine Learning
Template: 1

Use the most popular runtime and package type? (Python and zip) [y/N]: n

Which runtime would you like to use?
  1 - dotnet6
  2 - dotnet5.0
  3 - dotnetcore3.1
  4 - go1.x
  5 - java11
  6 - java8.al2
  7 - java8
  8 - nodejs16.x
  9 - nodejs14.x
 10 - nodejs12.x
 11 - python3.9
 12 - python3.8
 13 - python3.7
 14 - python3.6
 15 - ruby2.7
 16 - rust (provided.al2)
```

Fig 13

```
9 - nodejs14.x
10 - nodejs12.x
11 - python3.9
12 - python3.8
13 - python3.7
14 - python3.6
15 - ruby2.7
16 - rust (provided.al2)
Runtime: 7

What package type would you like to use?
1 - Zip
2 - Image
Package type: 1

Which dependency manager would you like to use?
1 - gradle
2 - maven
Dependency manager: 2

Would you like to enable X-Ray tracing on the function(s) in your application? [y/N]: n

Project name [sam-app]: KevinHan-1521885-sam-app

Cloning from https://github.com/aws/aws-sam-cli-app-templates (process may take a moment)

-----
Generating application:
-----
Name: KevinHan-1521885-sam-app
Runtime: java8
Architectures: x86_64
Dependency Manager: maven
Application Template: hello-world
Output Directory: .

Next steps can be found in the README file at ./KevinHan-1521885-sam-app/README.md

Commands you can use next
=====
[*] Create pipeline: cd KevinHan-1521885-sam-app && sam pipeline init --bootstrap
[*] Validate SAM template: sam validate
[*] Test Function in the Cloud: sam sync --stack-name {stack-name} --watch
```

After initializing the Serverless Architecture Model, you should see the directory of the project on the menu at the left side of the screen as shown in Fig 14.

Fig 14

```
▼ KevinHan-1521885-assignment4 - /home/ec2-user/environment
  ► KevinHan-1521885-sam-app
    [📄] aws-sam-cli-linux-x86_64.zip
    [📄] bootstrap.sh
    [📄] README.md
```



## 4. Task 3: Deploy Locally

The task is to be able to run the project locally. To do this we need to install some dependency.

Step 1: Install the prebuilt OpenJDK packages, update the configs for the java and javac.

Run the following commands:

```
sudo yum -y install java-1.8.0-openjdk-devel
sudo update-alternatives --config java
sudo update-alternatives --config javac
```

The Fig 15-16 shows the outcome as you enter the commands.

Fig 15

```
Installed:
java-1.8.0-openjdk-devel.x86_64 1:1.8.0.312.b07-1.amzn2.0.2

Dependency Installed:
alsa-lib.x86_64 0:1.1.4-1.2.amzn2
copy-jdk-configs.noarch 0:3.3-10.amzn2
gdk-pixbuf2.x86_64 0:2.36.12-3.amzn2
gtk-update-icon-cache.x86_64 0:3.22.30-3.amzn2
hicolor-icon-theme.noarch 0:0.12-7.amzn2
java-1.8.0-openjdk-headless.x86_64 1:1.8.0.312.b07-1.amzn2.0.2
libX11.x86_64 0:1.6.7-3.amzn2.0.2
libXcomposite.x86_64 0:0.4.4-4.1.amzn2.0.2
libXext.x86_64 0:1.3.3-3.amzn2.0.2
libXi.x86_64 0:1.7.9-1.amzn2.0.2
libXrender.x86_64 0:0.9.10-1.amzn2.0.2
libfontenc.x86_64 0:1.1.3-3.amzn2.0.2
libglvnd-glx.x86_64 0:1.0.1-0.1.git5baae5.amzn2.0.1
libwayland-server.x86_64 0:1.17.0-1.amzn2
lksctp-tools.x86_64 0:1.0.17-2.amzn2.0.2
mesa-libgbm.x86_64 0:18.3.4-5.amzn2.0.1
pccs-lite-libs.x86_64 0:1.8.8-7.amzn2
tzdata-java.noarch 0:2022a-1.amzn2
atk.x86_64 0:2.22.0-3.amzn2.0.2
cups-libs.x86_64 1:1.6.3-51.amzn2
glib.x86_64 0:4.1.6-9.amzn2.0.2
gtk2.x86_64 0:2.24.31-1.amzn2.0.2
jasper-libs.x86_64 0:1.900.1-33.amzn2
libICE.x86_64 0:1.0.9-9.amzn2.0.2
libX11-common.noarch 0:1.6.7-3.amzn2.0.2
libXcursor.x86_64 0:1.1.15-1.amzn2
libXfixes.x86_64 0:5.0.3-1.amzn2.0.2
libXinerama.x86_64 0:1.1.3-2.1.amzn2.0.2
libXtst.x86_64 0:1.2.3-1.amzn2.0.2
libglvnd.x86_64 1:1.0.1-0.1.git5baae5.amzn2.0.1
libthai.x86_64 0:0.1.14-9.amzn2.0.2
libxcb.x86_64 0:1.12-1.amzn2.0.2
mesa-libEGL.x86_64 0:18.3.4-5.amzn2.0.1
mesa-libglapi.x86_64 0:18.3.4-5.amzn2.0.1
pixman.x86_64 0:0.34.0-1.amzn2.0.2
xorg-x11-font-utils.x86_64 1:7.5-21.amzn2
cairo.x86_64 0:1.15.12-4.amzn2
fribidi.x86_64 0:1.0.2-1.amzn2.1
graphite2.x86_64 0:1.3.10-1.amzn2.0.2
harfbuzz.x86_64 0:1.7.5-2.amzn2
java-1.8.0-openjdk.x86_64 1:1.8.0.312.b07-1.amzn2.0.2
libSM.x86_64 0:1.2.2-2.amzn2.0.2
libXau.x86_64 0:1.0.8-2.1.amzn2.0.2
libXdamage.x86_64 0:1.1.4-4.1.amzn2.0.2
libXft.x86_64 0:2.3.2-2.amzn2.0.2
libXrandr.x86_64 0:1.5.1-2.amzn2.0.3
libXxf86vm.x86_64 0:1.1.4-1.amzn2.0.2
libglvnd-egl.x86_64 1:1.0.1-0.1.git5baae5.amzn2.0.1
libwayland-client.x86_64 0:1.17.0-1.amzn2
libxshmfence.x86_64 0:1.2-1.amzn2.0.2
mesa-libGL.x86_64 0:18.3.4-5.amzn2.0.1
pango.x86_64 0:1.42.4-4.amzn2
ttknfdir.x86_64 0:3.0.0-42.amzn2.0.2
xorg-x11-fonts-Type1.noarch 0:7.5-9.amzn2

Complete!
```

Fig 16

```
voclabs:~/environment $ sudo update-alternatives --config java

There are 2 programs which provide 'java'.

   Selection    Command
-----
        1      /usr/lib/jvm/java-11-amazon-corretto.x86_64/bin/java
*+ 2          java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64/jre/bin/java)

Enter to keep the current selection[+], or type selection number: 2
voclabs:~/environment $ sudo update-alternatives --config javac

There are 2 programs which provide 'javac'.

   Selection    Command
-----
        1      /usr/lib/jvm/java-11-amazon-corretto.x86_64/bin/javac
*+ 2          java-1.8.0-openjdk.x86_64 (/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.amzn2.0.2.x86_64/bin/javac)

Enter to keep the current selection[+], or type selection number: 2
voclabs:~/environment $
```

Step 2: Install maven, it is used to by the project to build.

Run the following commands:

```
sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo
sudo sed -i s/$releasever/7/g /etc/yum.repos.d/epel-apache-maven.repo
```

```
sudo yum install -y apache-maven
```

The Fig 17-18 shows what it is expected when you run the commands to install maven.

Fig 17

```
voclabs:~/environment $ sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo
--2022-05-31 03:34:47-- http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo
Resolving repos.fedorapeople.org (repos.fedorapeople.org)... 152.19.134.199, 2600:2701:4000:5211::dead:beef:a7:9474
Connecting to repos.fedorapeople.org (repos.fedorapeople.org)|152.19.134.199|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo [following]
--2022-05-31 03:34:47-- https://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo
Connecting to repos.fedorapeople.org (repos.fedorapeople.org)|152.19.134.199|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 445
Saving to: '/etc/yum.repos.d/epel-apache-maven.repo'

100%[=====]

2022-05-31 03:34:48 (26.8 MB/s) - '/etc/yum.repos.d/epel-apache-maven.repo' saved [445/445]
```

Fig 18

```
voclabs:~/environment $ sudo sed -i s/$(releasever)/g /etc/yum.repos.d/epel-apache-maven.repo
voclabs:~/environment $ sudo yum install -y apache-maven
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                               | 3.7 kB  00:00:00
amzn2extra-docker                        | 3.0 kB  00:00:00
amzn2extra-epel                         | 3.0 kB  00:00:00
amzn2extra-lamp-mariadb10.2-php7.2     | 3.0 kB  00:00:00
epel2AdG.64/metalink                   | 19.5 kB  00:00:00
epel-apache-maven                      | 3.3 kB  00:00:00
hashicorp                              | 1.4 kB  00:00:00
epel-apache-maven/80_64/primary.db     | 7.7 kB  00:00:00
236 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
--> Package apache-maven.noarch 0:3.5.2-1.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

Package Arch Version Repository Size
-----
Installing:
apache-maven noarch 3.5.2-1.el7 epel-apache-maven 8.0 M

Transaction Summary
Install 1 Package

Total download size: 8.0 M
Installed size: 9.6 M
Downloading packages:
apache-maven-3.5.2-1.el7.noarch.rpm | 8.0 MB 00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
Installing : apache-maven-3.5.2-1.el7.noarch 1/1
Verifying : apache-maven-3.5.2-1.el7.noarch 1/1

Installed:
  apache-maven.noarch 0:3.5.2-1.el7

Complete!
voclabs:~/environment $
```

Step 3: We can now build the project from the template file.

Ensure that you are in the right directory, run the command “sam build”. If done correctly, you should see that the console will output “Build Succeeded” as shown in the Fig 19. This shows the project has been successfully built.

Fig 19

```
voclabs:~/environment/KevinHan-1521885-sam-app $ sam build
Your template contains a resource with logical ID "ServerlessRestApi", which is a reserved logical ID in AWS SAM. It could result in unexpected behaviors and is not recommended.
Test the latest build changes for Java runtime 'SAM_CLI_BETA_MAVEN_SCOPE_AND_LAYER=1 sam build'. These changes will replace the existing flow on 1st of April 2022. Check https://github.com/aws/aws-sam-cli/issues/3639 for more information.
Building codeuri: /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction runtime: java8 metadata: {} architecture: x86_64 functions: ['HelloWorldFunction']
Running JavaMavenWorkflow:CopySource
Running JavaMavenWorkflow:MavenBuild
Running JavaMavenWorkflow:MavenCopyDependency
Running JavaMavenWorkflow:MavenCopyArtifacts

Build Succeeded

Built Artifacts : .aws-sam/build
Built Template : .aws-sam/build/template.yaml

Commands you can use next
=====
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {stack-name} --watch
[*] Deploy: sam deploy --guided
```

Step 4: To deploy the project locally after the successful build.

To deploy the project, ensure that you are currently in yourname-studentid-sam-app folder. Then run the command “sam local start-api --port 8080” to deploy the project.

The Fig 20 shows the project has deploy.

Fig 20

```
voclabs:~/environment/KevinHan-1521885-sam-app $ sam local start-api --port 8080
Mounting HelloWorldFunction at http://127.0.0.1:8080/hello [GET]
You can now browse to the above endpoints to invoke your functions. You do not need to restart/reload SAM CLI while working on your functions, changes will be reflected instantly/automatically. You only need to restart SAM CLI if you update your AWS SAM template
2022-05-31 09:22:15 * Running on http://127.0.0.1:8080/ (Press CTRL+C to quit)
```

To test whether the project is working locally, we can send “curl http://localhost:8080/hello” in another terminal. The output should be the same as the Fig 21.

Fig 21

```
voclabs:~/environment $ curl http://localhost:8080/hello
{ "message": "hello world", "location": "184.72.143.166" }voclabs:~/environment $
```

The Fig 22 shows the request made by the curl command above.

Fig 22

```
Mounting /home/ec2-user/environment/KevinHan-1521885-sam-app/.aws-sam/build/HelloWorldFunction as /var/task:ro,delegated inside runtime container
END RequestId: 9731497a-29b7-4414-b3f3-b65e31b27a1a
REPORT RequestId: 9731497a-29b7-4414-b3f3-b65e31b27a1a Init Duration: 1.59 ms Duration: 1054.19 ms Billed Duration: 1055 ms Memory Size: 512 MB Max Memory Used: 512 MB
2022-05-31 09:23:59 127.0.0.1 - - [31/May/2022 09:23:59] "GET /hello HTTP/1.1" 200 -
```

## 5. Task 4: Configuring your repository

The task is to have our latest version of deployed project to be stored in a cloud repository. For this assignment, we are storing the repository in aws’s git service.

Step 1: Creating the git repository

Run the command ‘aws codecommit create-repository --repository-name “yournamesamapp-repo”’ The output of the command is shown in Fig 23.

Fig 23

```
voclabs:~/environment/KevinHan-1521885-sam-app $ aws codecommit create-repository --repository-name "KevinHan-samapp-repo"
{
  "repositoryMetadata": {
    "accountId": "505253522687",
    "repositoryId": "8017a9a5-d655-4083-a9e1-4ef9c7057fd8",
    "repositoryName": "KevinHan-samapp-repo",
    "lastModifiedDate": 1653989413.693,
    "creationDate": 1653989413.693,
    "cloneUrlHttp": "https://git-codecommit.us-east-1.amazonaws.com/v1/repos/KevinHan-samapp-repo",
    "cloneUrlSsh": "ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/KevinHan-samapp-repo",
    "arn": "arn:aws:codecommit:us-east-1:505253522687:KevinHan-samapp-repo"
  }
}
```

Step 2: After successfully creating the repo, we need to configure the git repo with your details. To allow us to know who the commits are made by. Run the command shown in Fig 24.

Fig 24

```
voclabs:~/environment/KevinHan-1521885-sam-app $ git config --global user.name && --global user.email
```

Step 3: We need to config git to ignore some files. Create a .gitignore file, if it is not showing, enable hidden files. We do not want to store the built applications, so add the following as show in the fig 25.

Fig 25

```
1 .aws-sam/
2 packaged.yaml
3 pipeline.sh
```

Step 4: We can now population our repository with the project files. To do this we need to run the following commands “git init”, “git add .” and “git commit -m “Initial commit””.

Git init – to initialise a new local repository which store all the object and ref used in the project and creates a project’s history.

Git add . – add the updates made to the project

Git commit – it captures the snapshot of the current stage changes of the project.

When you run the commands, it should look similar to the Fig 26-27.

Fig 26

```
voclabs:~/environment/KevinHan-1521885-sam-app $ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/ec2-user/environment/KevinHan-1521885-sam-app/.git/
```

Fig 27

```
voclabs:~/environment/KevinHan-1521885-sam-app (master) $ git commit -m "Initial commit"
[master (root-commit) 7b096f2] Initial commit
Committer: EC2 Default User <ec2-user@ip-172-31-30-42.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

7 files changed, 358 insertions(+)
create mode 100644 .gitignore
create mode 100644 HelloWorldFunction/pom.xml
create mode 100644 HelloWorldFunction/src/main/java/helloworld/App.java
create mode 100644 HelloWorldFunction/src/test/java/helloworld/AppTest.java
create mode 100644 README.md
create mode 100644 events/event.json
create mode 100644 template.yaml
```

Step 5: Now we can add the remote origin that we created in step 1 and push the repository to the “cloneUrlHttp”. Follow the commands show in Fig 28-29.

Fig 28

```
voclabs:~/environment/KevinHan-1521885-sam-app (master) $ git remote add origin https://git-codecommit.us-east-1.amazonaws.com/v1/repos/KevinHan-samapp-repo
```

Fig 29

```
voclabs:~/environment/KevinHan-1521885-sam-app (master) $ git push --set-upstream origin master
Enumerating objects: 18, done.
Counting objects: 100% (18/18), done.
Compressing objects: 100% (10/10), done.
Writing objects: 100% (18/18), 6.91 KiB | 1011.00 KiB/s, done.
Total 18 (delta 0), reused 0 (delta 0), pack-reused 0
To https://git-codecommit.us-east-1.amazonaws.com/v1/repos/KevinHan-samapp-repo
* [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

After pushing the repository, you should be able to see your populated repository in the AWS codecommit console. Your populated repository should look like Fig 30.

Fig 30

The screenshot shows the AWS CodeCommit console interface for a repository named 'KevinHan-samapp-repo'. At the top, there's a breadcrumb trail: 'Developer Tools > CodeCommit > Repositories > KevinHan-samapp-repo'. Below this, the repository name 'KevinHan-samapp-repo' is displayed with an 'info' link. To the right of the name are buttons for 'Notify', a dropdown menu currently showing 'master', 'Create pull request', and 'Clone URL'. Below the repository name is a section titled 'KevinHan-samapp-repo info' with an 'Add file' button. This section contains a table of files and folders: 'events' (folder), 'HelloWorldFunction' (folder), '.gitignore' (file), 'README.md' (file), and 'template.yaml' (file). Below this table is a section for the 'README.md' file, which includes a 'View source' button and an 'Edit' button. The README content describes the project as a serverless application deployable with the SAM CLI, listing files like 'HelloWorldFunction/src/main', 'events', 'HelloWorldFunction/src/test', and 'template.yaml'. It also mentions the use of AWS resources and the AWS Toolkit for development.

## 6. Task 5: Implementing a CI/CD pipeline

The task is to create a bash script which automates the testing, building, committing, and pushing. If the tests are successful, it will attempt to build. If the build is successful, it will commit and push to the AWS Code Commit and ask the user if they want to deploy the project. If the build fails, it should not commit and push to AWS Code Commit. If the tests fails, it should not build, commit, and push to AWS Code Commit.

Fig 31

```
echo "(1) Test"
cd HelloWorldFunction
if mvn test ; then
    echo "Test Successful"
    cd ..
    echo "(2) Build"
    if sam build ; then
        read -p "Commit Message : " message
        git add .
        git commit -m "$message"
        git push
        while true; do
            read -p "Deploy Locally(Y/N): " deploy
            case $deploy in
                [yY] ) echo "Deploy locally";
                        sam local start-api --port 8080
                        break;;
                [nN] ) echo "Not deploying locally";
                        exit;;
                * ) echo "Please Enter Y or N";;
            esac
        done
    else
        echo "Build failed"
    fi
else
    echo "Test failed"
fi
```

I have setup the bash script as shown in Fig 31. The bash script first check if it can run the test successfully, then check if can build successfully. If it is build successful, it will ask the user for a commit message before committing and pushing the AWS Code Commit. Then ask the user whether they want to deploy locally. The user are given 2 choices Y or N. If other input are given, it will give an error and ask the user again. Error outputs are shown if the test or build are to fail. I decided to developed this way based on the assignment specification. The following Fig 32-36 shows my pipeline script running.

Fig 32

```

voclabs:~/environment/KevinHan-1521885-sam-app (master) $ bash pipeline.sh
(1) Test
[INFO] Scanning for projects...
[INFO] -----
[INFO] Building A sample Hello World created for SAM CLI. 1.0
[INFO] -----
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ HelloWorld ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/src/main/resources
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ HelloWorld ---
[INFO] Nothing to compile - all classes are up to date
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ HelloWorld ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/src/test/resources
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ HelloWorld ---
[INFO] Nothing to compile - all classes are up to date
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ HelloWorld ---
[INFO] Surefire report directory: /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/target/surefire-reports

-----
T E S T S
-----
Running helloworld.AppTest
Downloading quotes.txt from S3 bucket kevinhan-bucket...
Done!
Successfully wrote to the file.
99
Success is walking from failure to failure with no loss of enthusiasm. -Winston Churchill
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.258 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 6.919 s
[INFO] Finished at: 2022-06-15T09:55:16Z

```

Fig 33

```

bash -lp-172-31-30-42 ex
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 6.919 s
[INFO] Finished at: 2022-06-15T09:55:16Z
[INFO] Final Memory: 10M/24M
[INFO] -----
Test Successful
(2) Build
Your template contains a resource with logical ID "ServerlessTestApp", which is a reserved logical ID in AWS SAM. It could result in unexpected behaviors and is not recommended.
Test the latest build changes for java runtime "SAM_CLI_BETA_NATIVE_SCOPE_AND_LAYER1 sam build". These changes will replace the existing flow on 1st of April 2022. Check https://github.com/aws/aws-sam-cli/issues/3639 for more information.
Building codeuri: /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction runtime: java8 metadata: {} architecture: x86_64 functions: ["HelloWorldFunction"]
Running JavaMavenWorkflow:CopySource
Running JavaMavenWorkflow:MavenBuild
Running JavaMavenWorkflow:MavenCopyDependency
Running JavaMavenWorkflow:MavenCopyArtifacts
Build Succeeded
Built Artifacts : .aws-sam/build
Built Template : .aws-sam/build/template.yaml
Commands you can use next
=====
(*) Validate SAM template: sam validate
(*) Invoke Function: sam local invoke
(*) Test Function in the Cloud: sam sync --stack-name (stack-name) --watch
(*) Deploy: sam deploy --guided
Commit Message : Implemented Task 6
(aster: #765503) Implemented Task 6
Committer: EC2 Default User <ec2-user@ip-172-31-30-42.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

14 files changed, 292 insertions(+), 15 deletions(-)
create mode 100644 .attach_pid14172
create mode 100644 .attach_pid20746

```

Fig 34

```
bash - ip-172-31-30-42.ec2
Commands you can use next
=====
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name [stack-name] --watch
[*] Deploy: sam deploy --guided

Commit Message : Implemented Task 6
[master e765583] Implemented Task 6
Committer: EC2 Default User <ec2-user@ip-172-31-30-42.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname, please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

14 files changed, 292 insertions(+), 15 deletions(-)
create mode 100644 .attach_pid14172
create mode 100644 .attach_pid20746
create mode 100644 HelloWorldFunction/dependency-reduced-pom.xml
create mode 100644 HelloWorldFunction/quotes.txt
create mode 100644 HelloWorldFunction/target/HelloWorld-1.0.jar
rewrite HelloWorldFunction/target/classes/HelloWorld/App.class (80%)
create mode 100644 HelloWorldFunction/target/maven-archiver/pom.properties
create mode 100644 HelloWorldFunction/target/original-HelloWorld-1.0.jar
rewrite HelloWorldFunction/target/test-classes/HelloWorld/AppTest.class (91%)
Enumerating objects: 58, done.
Counting objects: 100% (58/58), done.
Compressing objects: 100% (18/18), done.
Writing objects: 100% (30/30), 8.83 MiB | 9.88 MiB/s, done.
Total 30 (delta 3), reused 0 (delta 0), pack-reused 0
To https://git-codecommit.us-east-1.amazonaws.com/v1/repos/KevinHan-samapp-repo
bed9ad5..e765583 master -> master
Deploy Locally(Y/N): Y
Deploy locally
Mounting HelloWorldFunction at http://127.0.0.1:8080/hello [GET]
You can now browse to the above endpoints to invoke your functions. You do not need to restart/reload SAM CLI while working on your functions, changes will be reflected instantly/automatically. You only need to restart SAM CLI if you update you
ur AWS SAM template
2022-06-15 09:56:41 * Running on http://127.0.0.1:8080/ (Press CTRL+C to quit)
```

Fig 35

Commit ID	Commit message	Commit date	Author	Actions
e765583e	Implemented Task 6	Just now	EC2 Default User	<a href="#">Copy ID</a> <a href="#">Browse</a>
bed9ad5f	Testing and Build Successful	14 days ago	EC2 Default User	<a href="#">Copy ID</a> <a href="#">Browse</a>
67f7d8b3	\$message	14 days ago	EC2 Default User	<a href="#">Copy ID</a> <a href="#">Browse</a>
7b096f20	Initial commit	15 days ago	EC2 Default User	<a href="#">Copy ID</a> <a href="#">Browse</a>

Fig 36

```
Deploy Locally(Y/N): N
Not deploying locally
voclabs:~/environment/KevinHan-1521885-sam-app (master) $
```

## 7. Task 6: Implementing Functionality and Testing

The task is to make some changes to the java application and testing the pipeline and application. The changes are to allow the program to access the S3 bucket and retrieve the quotes stored and randomly display a quote back when a curl is made. Testing must ensure the quote exists in the quotes.txt retrieve from the bucket.

Fig 37 shows the test failed when we made a change to the hello world message in the App.java.



Fig 37

```
Results :

Failed tests:  successfulResponse(helloworld.AppTest)

Tests run: 1, Failures: 1, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD FAILURE
[INFO] -----
[INFO] Total time: 3.857 s
[INFO] Finished at: 2022-06-01T01:35:40Z
[INFO] Final Memory: 8M/20M
[INFO] -----
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-surefire-plugin:2.12.4:test (default-test) on project HelloWorld: There are test failures.
[ERROR]
[ERROR] Please refer to /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/target/surefire-reports for the individual test results.
[ERROR] -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MojoFailureException
Test failed
```

The following Fig 38-39 show the bucket that has been created and the storing the quotes.txt file in the bucket.

Fig 38

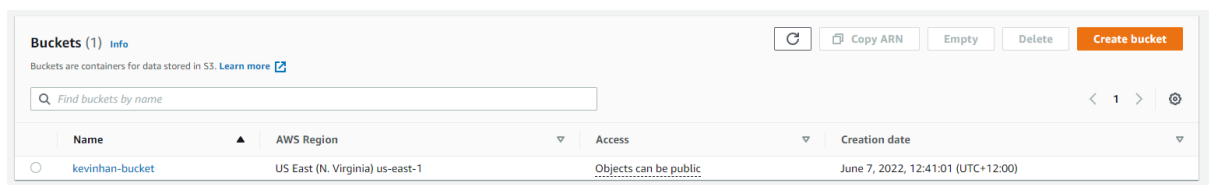


Fig 39

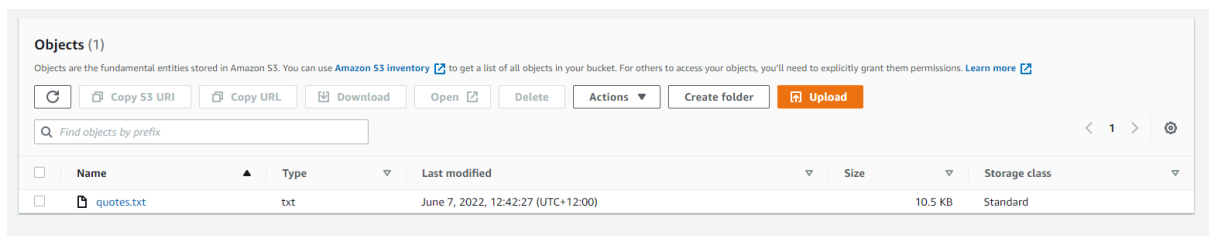


Fig 40 show the imports of the java and aws service that I will be using for the changes made in this task.

Fig 41-42 shows the dependency needed for the App.java to retrieve the contents of the buckets. We are getting the particular dependency instead of the whole Maven SDK Modules, so it doesn't take a long time building the application. The dependency are added to the pom.xml.

Fig 40

```
package helloworld;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.FileReader;
import java.io.FileWriter;
import java.net.URL;
import java.util.concurrent.ThreadLocalRandom;
import java.util.HashMap;
import java.util.Map;
import java.util.stream.Collectors;
import java.util.ArrayList;
import java.nio.charset.StandardCharsets;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.APIGatewayProxyRequestEvent;
import com.amazonaws.services.lambda.runtime.events.APIGatewayProxyResponseEvent;
import com.amazonaws.AmazonServiceException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.S3Object;
import com.amazonaws.services.s3.model.S3ObjectInputStream;
import com.amazonaws.services.s3.model.ListObjectsV2Result;
import com.amazonaws.services.s3.model.S3ObjectSummary;
```

Fig 41

```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>aws-java-sdk-bom</artifactId>
      <version>1.11.1000</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
```

Fig 42

```
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-s3</artifactId>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-dynamodb</artifactId>
</dependency>
</dependencies>
```

Fig 43

```
public class AppTest {
    @Test
    public void successfulResponse() {
        App app = new App();
        APIGatewayProxyResponseEvent result = app.handleRequest(null, null);
        assertEquals(200, result.getStatusCode().intValue());
        assertEquals("application/json", result.getHeaders().get("Content-Type"));
        ArrayList<String> quotelist = new ArrayList<String>();
        BufferedReader reader;
        try {
            reader = new BufferedReader(new FileReader("/home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/quotes.txt"));
            String line = reader.readLine();
            while (line != null) {
                // System.out.println(line);
                // read next line
                line = reader.readLine();
                quotelist.add(line);
            }
            reader.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        String content = result.getBody();
        // System.out.println(content);
        String quote = result.getBody();
        quote = quote.substring(14);
        quote = quote.split("\"")[0];
        // System.out.println(quote);
        assertNotNull(content);
        assertTrue(content.contains("\"message\""));
        assertTrue(quotelist.contains(quote));
        assertTrue(content.contains("\"location\""));
    }
}
```

Fig 43 show the AppTest.java that has been modified to check whether the quote parse contains within the quotes.txt file.

Fig 44

```
// Code to parse bucket data
String key_name = "quotes.txt";
String bucket_name = "kevinhan-bucket";
System.out.format("Downloading %s from S3 bucket %s...\n", key_name, bucket_name);
String result = "";
final AmazonS3 s3 = AmazonS3ClientBuilder.standard().withRegion(Regions.US_EAST_1).build();
try {
    S3Object o = s3.getObject(bucket_name, key_name);
    S3ObjectInputStream s3is = o.getObjectContent();
    result = new BufferedReader(new InputStreamReader(s3is)).lines().parallel().collect(Collectors.joining("\n"));
    // System.out.println(result);
    // FileOutputStream fos = new FileOutputStream(new File(key_name));
    // byte[] read_buf = new byte[1024];
    // int read_len = 0;
    // while ((read_len = s3is.read(read_buf)) > 0) {
    //     fos.write(read_buf, 0, read_len);
    // }
    s3is.close();
    // fos.close();
} catch (AmazonServiceException e) {
    System.err.println(e.getErrorMessage());
    System.exit(1);
} catch (IOException e) {
    System.err.println(e.getMessage());
    System.exit(1);
}
System.out.println("Done!");

try {
    FileWriter myWriter = new FileWriter(key_name);
    myWriter.write(result);
    myWriter.close();
    System.out.println("Successfully wrote to the file.");
} catch (IOException e) {
    System.out.println("An error occurred.");
    e.printStackTrace();
}
```

Fig 44 show the code that parse in from the bucket. Make sure that the region of the bucket is based on the region you created your bucket. Since the SAM runs on the cloud, it is not able to read and write 2 files. So, I decided to convert s3is input stream to a string as shown

in the Fig 44. I added a file write which will write the content of the string to a file which is used by the AppTest.java to check whether the quote exists.

Fig 45

```
//Code to setup quote
// System.out.println(System.getProperty("user.dir"));
ArrayList<String> quotelist = new ArrayList<String>();
BufferedReader reader;
try {
    reader = new BufferedReader(new FileReader("/home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/quotes.txt"));
    String line = reader.readLine();
    while (line != null) {
        // System.out.println(line);
        // read next line
        line = reader.readLine();
        quotelist.add(line);
    }
    reader.close();
} catch (IOException e) {
    e.printStackTrace();
}
String[] tempList = result.split(System.lineSeparator());
for(int i = 0; i < tempList.length; i++) {
    quotelist.add(tempList[i]);
}
System.out.println(quotelist.size());
```

Fig 45 shows the string receive from the bucket being separated by the line separator and stored in an array list called quoteList.

Fig 46, the output has been modified to output the quote instead of the original hello message.

Fig 46

```
final String pageContents = this.getPageContents("https://checkip.amazonaws.com");
String output = String.format("{ \"message\": \"%"+quote+"\", \"location\": \"%s\" }", pageContents);
```

Fig 47

```
voclabs:~/environment/KevinHan-1521885-sam-app/HelloWorldFunction (master) $ mvn test
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building A sample Hello World created for SAM CLI. 1.0
[INFO] -----
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ HelloWorld ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/src/main/resources
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ HelloWorld ---
[INFO] Nothing to compile - all classes are up to date
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ HelloWorld ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/src/test/resources
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ HelloWorld ---
[INFO] Nothing to compile - all classes are up to date
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ HelloWorld ---
[INFO] Surefire report directory: /home/ec2-user/environment/KevinHan-1521885-sam-app/HelloWorldFunction/target/surefire-reports

-----
T E S T S
-----
Running helloworld.AppTest
Downloading quotes.txt from S3 bucket kevinhan-bucket...
Done!
Successfully wrote to the file.
99
Knowledge is being aware of what you can do. Wisdom is knowing when not to do it. ~Anonymous
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.143 sec

Results :
```

Fig 48

```
Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 6.306 s
[INFO] Finished at: 2022-06-15T10:40:18Z
[INFO] Final Memory: 9M/23M
[INFO] -----
voclabs:~/environment/KevinHan-1521885-sam-app/HelloWorldFunction (master) $
```

Fig 49

```
voclabs:~/environment $ curl http://localhost:8080/hello
{"message": "The reason most people never reach their goals is that they don't define them, or ever seriously consider them as believable or achievable. Winners can tell you where they are going, what they plan to do along the way, and who will be sharing the adventure with them. ~Denis Waitley", "location": "3.88.103.44" }voclabs:~/environment $
```

Fig 50

```
voclabs:~/environment/KevinHan-1521885-sam-app (master) $ sam local start-api --port 8080
Mounting HelloWorldFunction at http://127.0.0.1:8080/hello [GET]
You can now browse to the above endpoints to invoke your functions. You do not need to restart/reload SAM CLI while working on your functions, changes will be reflected instantly/automatically. You only need to restart SAM CLI if you update your AWS SAM template
2022-06-15 10:47:54 * Running on http://127.0.0.1:8080/ (Press CTRL+C to quit)
Invoking helloworld.App::handleRequest (java8)
Skip pulling image and use local one: public.ecr.aws/sam/emulation-java8:rapid-1.50.0-x86_64.

Mounting /home/ec2-user/environment/KevinHan-1521885-sam-app/.aws-sam/build/HelloWorldFunction as /var/task:ro,delegated inside runtime container
Picked up JAVA_TOOL_OPTIONS: -XX:+TieredCompilation -XX:TieredStopAtLevel=1
Downloading quotes.txt from S3 bucket kevinhan-bucket...
Done!
99
Thinking should become your capital asset, no matter whatever ups and downs you come across in your life. ~Dr. APJ Kalam
END RequestId: 9020548c-31e8-4c6c-8e59-c42fc76ec2ad
REPORT RequestId: 9020548c-31e8-4c6c-8e59-c42fc76ec2ad Init Duration: 4.32 ms Duration: 4290.71 ms Billed Duration: 4291 ms Memory Size: 512 MB Max Memory Used: 512 MB
2022-06-15 10:48:07 127.0.0.1 - - [15/Jun/2022 10:48:07] "GET /hello HTTP/1.1" 200 -
```

Fig 47-50 shows the test being conducted after the changes has been made to the assignment brief. We can see the message response with a quote from the quotes.txt from the bucket list. For the mvn test, we are testing whether the quote exist within the quote.txt, the http status code is 200.

## 8. Conclusion

The purpose of the project is to show our understanding of implementation of functionality and testing of the changes made. Allowing automate scripts allow us as software engineers to efficiently test and push the latest working deploy project to the repository.