LAB 1: WEB HOSTING

Link: https://www.youtube.com/watch?v=YNL22VWgm9Q

Step 1: Creating EC2 instance

1. Create an EC2 instance by clicking on launch instance
2. Select Linux AMI and free tier (t2.micro) instance type
3. Configure security group
    a. Add SSH
    b. Add HTTP rule and to make it publicly accessible, remove ::/0 from source
    c. Add HTTPS rule and to make it publicly accessible, remove ::/0 from source
4. Review and launch
5. Add a key pair (existing / new) and click acknowledge
    a. If a new kay pair is added give the name and download the key pair
    b. The key pair is of pem extension
6. Launch Instance
7. Rename the created instance

Step 2: Creating S3 bucket for web application file

1. Go to S3 console
2. Click create bucket
3. Give bucket name then set region to Mumbai to avoid latency
4. Click on create and bucket is created
5. To make it public
    a. Click on created bucket
    b. Edit public access setting
    c. Uncheck block all public access and ensure all the checkboxes are unchecked
    d. Click save and confirm by typing confirm
6. Open the bucket by clicking on it
7. Click upload -> Add files -> select **zip file** -> click upload

8. To set object as public
    a. click on the uploaded zip file
    b. click actions -> search for make public in the drop down and click
9. go back to ec2 instance
10. Convert the downloaded pem file to ppk file using puttygen
    - Open puttygen and click on load
    - Select the downloaded pem file
    - Click on save private key
    - The downloaded file will be ppk extension
    - To access the EC2 instance and open SSH, ppk is mandatory
    - Close file
11. Open putty software
    a. Copy paste public IPv4 address from the ec2 instance
    b. Paste it in host name in the putty software in the format **"ec2-user@IPv4 address"**
    c. Ensure SSH radio button is enabled
    d. Go to Category on the left side and click SSH
    e. Click on Connection -> SSH -> Auth
    f. Click browse and upload ppk file -> click open
    g. Click yes on the pop up window of security alert
    h. Terminal opens
    i. Type the following commands in the cmd

Step 1: Set to root user

sudo su

Step 2: Update package for EC2 instance

yum update –y

Step 3: Install Apache to run the website

yum install httpd –y

Step 4: Check the path using pwd

Step 5: Change directory to html

cd /var/www/html

Step 6: List the directory using ls command -> no files

Step 7: Get the files from S3

a. Click on the zip file in s3 bucket in AWS
b. Copy paste **Object URL** under Overview
c. wget s3url
d. use ls command to get the list of files -> uploaded zip file can be seen

Step 8: To unzip the uploaded zip file

a. Unzip filename.zip (the same file name in the previous output along with the extension)
b. Type ls -> zip file(red color) and extracted file(blue color) name is printed

Step 9: To move all the files to EC2

a. **mv filename/* .** (filename is in blue color)
b. type **ls** -> list out all the files in EC2
c. to ensure if we are in the right path type **pwd**
d. the path **"/var/www/html"** is printed

Step 10: To run Apache server

service httpd start

Step 11: To get the output

a. go to EC2 instance
b. copy IPv4 address
c. paste the copied address in a new tab to see the website

LAB2: JAVA COMPILER

Link:https://drive.google.com/file/d/1jmx3lPfmEUBQHk3J6M2xUK-Y2sFsetLb/view

Note: IOPS is Input Output Operations per second – CPU burst rate – default value is 100/3000

Step 1: Create EC2 instance

1. Create an EC2 instance by clicking on launch instance
2. Select Linux AMI and free tier (t2.micro) instance type
3. Configure security group
    a. Add SSH
    b. Add Custom TCP rule with port range 8080 and remove ::/0
    c. Review and launch -> add key-pair -> acknowledge -> launch
    d. Rename the created instance

Step 2: Go to putty software

1. Copy paste IPv4 address from EC2 instance
2. Select SSH -> Auth -> Load ppk file -> Click yes to security alert -> Terminal Opens
3. Login as **ec2-user**
4. Put the following commands in the putty terminal

Step 1: Update and check java version

sudo yum update

java –version

Step 2: To make major changes

a. sudo su #to login as root user

b. mkdir java #to install java create a directory
c. cd java #change directory to java

Step 3: If any older version of java is found

sudo yum remove java-**VersionNumber**-openjdk

Step4: If no version of java is found

a. sudo yum install java-1.8.0-openjdk #displays all the installed packages
b. give consent by entering y to download the packages
c. java –version #check the installed java version

Step 5: Download Apache Tomcat

a. google search for tomcat
b. click on the first link
c. click on download and select any version (3 versions available)
d. scroll down go to Binary Distributions -> Core -> **tar.gz** file
e. right click on the file and copy the link
f. go back to putty window

Step 6: To pull the data

a. wget copied_link

i.e. wget https://dlcdn.apache.org/tomcat/tomcat-8/v8.5.70/bin/apache-tomcat-8.5.70.tar.gz

b. to verify enter **ls command** which will print the files
c. Now we have the apache file inside the java folder

Step 7: Extract the file

a. tar xvfz apache-tomcat-8.5.58.tar.gz
b. use ls command to view

Step 8: Change directory to the apache file as we have to work with 3 main files – bin, conf and webapps

cd apache-tomcat-8.5.58/

Step 9: To start the Tomcat server

#find command file inside the bin

a. cd bin
b. ./startup.sh #to start the server
c. ps -ef | grep tomcat
   #to identify the keyword “tomcat” to verify if tomcat is installed and running

Step 10: To pull data from local host web browser

wget http://localhost:8080

Note: index.html is always the home page

Step 11: To get the Tomcat running

a. Copy IPv4 DNS address from EC2 instance
b. Paste the address to a new tab and add **:8080** at the end of the url
c. Now Tomcat is installed and running
d. Click on Manager App button in the right side of the web browser
e. Manager App requires username and password. If entered it does not allow you because it is with default attributes.
f. **To change the attributes modify few files**
   Go back to terminal window

Step 12: Currently it is in bin folder and we should move to subfolder named **“webapps”**

a. cd ..
b. Is webapps/manager/META-INF/
c. vi webapps/manager/META-INF/context.xm1

#to edit the context file

d. Comment <Valve className> (2 lines)
e. Press escape :wq #to close the file

Step 13: Edit the conf file

a. vi conf/tomcat-users
b. Insert 2 lines at the end of the conf file before the close tag of </tomcat-users>
c. <role rolename="manager-gui/>
<user username="tomcat" password="tomcat" roles="tomcat, manager-gui"/>
d. Press escape :wq

Step 14: Change the directory to the bin as it holds the startup and shutdown options

a. cd bin
b. shutdown.sh #to restart shutdown and start again
c. ./startup.sh #to start

Note: It is running successfully. Now we should be able to get into the Tomcat server.

Step 14: Check if you can enter Tomcat Manager App by giving the login credentials

a. Username = "tomcat"
b. Password = "tomcat"
c. It directs to Tomcat Web Application Manager

Step 15: To deploy the war file

a. Scroll down -> **War file to display** section -> click on choose file and upload -> deploy
b. The war file name will be visible in the table after deploying
c. Click on the war file name to see the output
d. You can verify the IPv4 DNS address in both the browser as well as EC2 instance console

Note: We have edited 2 files i.e. context.xml and tomcat-users.xml

LAB 3: ELASTIC BLOCK STORE

Link:https://drive.google.com/file/d/1woSG7JWQ84jnBzJcfWiV5cLa4xnKv5HB/view

Steps:

1. Create EC2 instance with no changes
2. Click on EBS and add volume by clicking on CREATE VOLUME
3. Make changes to the configuration
   a. Size = 1GiB
   b. set the availability zone as per the EC2 instance and
   c. add a tag with key = “name” and value=”My Volume”
4. Attach the volume to the instance created by clicking on Actions -> Attach Volume -> insert instance id -> Attach
5. Status will turn to **in-use**
6. Open putty and type the commands to configure the new volume

Step 1: Creating directory and file system and mount in PUTTY

a. df -h #list out file systems attached in server
b. sudo mkfs -t ext3 /dev/sdf #create a file system in virtual machine
c. sudo mkdir /mnt/data-store #create a directory to mount storage
d. sudo mount /dev/sdf /mnt/data-store #to mount new volume
e. echo "/dev/sdf /mnt/data-store ext3 defaults,noatime 12" | sudo tee -a /etc/fsta
   #mount volume whenever instance get started
f. sudo mkfs -t ext3 /dev/sdf #create file system of type ext3
g. cat /etc/fstab #view the file
h. df –h

Step 2: Create a text file in PUTTY

a. sudo sh -c "echo Hello my name is Asha Thampi. I am creating new Volume > /mnt/data-store/asha.txt"
b. cat /mnt/data-store/asha.txt

Step 3: Create Snapshot in AWS

a. Make a snapshot of the volume by clicking on the volume name in EBS console.
b. Actions -> Create Snapshot
c. Add tag and give key = “name” and value=”My Snapshot”

Step 4: Delete the created text file in PUTTY

a. sudo rm /mnt/data-store/asha.txt
b. sudo cat /mnt/data-store/asha.txt #to confirm deletion
c. sudo ls /mnt/data-store/ #to confirm deletion

Step 5: Restore file from snapshot created in AWS

a. Go to snapshots under EBS click on it
b. Actions -> Create Volume with no modifications except availability zone if different
c. Create tag with key = “name” and value = “Restored Volume”
d. Go to EBS volume and click on Restored Volume and link it to the instance
e. Actions -> Attach Volume -> Select Instance

Step 6: Restore file in PUTTY

a. sudo mkdir /mnt/data-restore #new directory for restored volume
b. sudo mount /dev/sdg /mnt/data-restore
c. ls /mnt/data-restore #confirm file restore
d. cat /mnt/data-restore/asha.txt #confirm file content restore

LAB 4: SCALING & LOAD BALANCING

Link: Module 10 Lab 6

https://drive.google.com/file/d/1JPZWX5RuZelt445LsN4AJhXeO0go_lV7/view?hl=en

Steps:

1. Go to modules lab (Lab 6)
2. EC2 dashboard -> running instances (2) – Bastion host(default instance)
3. Create another instance named **Web Server 1** (Linux instance, t2.micro)

4 MAIN STEPS

1. **Create image and template for this instance**

a. Click on web server 1 -> actions -> image and templates -> create image

b. Give image name as WebServerAMI

c. Do not change any other configuration

d. Click create image (this image will be used for auto-scaling)

e. In instances there will be only 2 instances running

2. **Create a Load Balancer**

   a. Click on load balancer -> create load balancer

   b. There will be 3 types of load balancers

   c. Click on application load balancer

   d. Give the name as WebServerELB

   e. Scroll down to Availability Zone, in VPC click dropdown and select Lab VPC

   f. HTTP will be 8080

   g. Enable both the availability zones. In the first dropdown select public subnet 1 and in the second select public subnet 2

   h. Click next

   i. Ignore the warning in configure security groups(step 2) and click next

j. In Configure security group (step 3), by default it will be default
k. Select Web Security Group and deselect default
l. Click next
m. In configure routing set name as WebServerGroup
n. Do not change any other configurations and click next
o. In register targets there will be 2 instances. Do not select any instance and click next as it gets automatically selected in auto scaling configuration
p. Click review -> create
q. There will be 3 ticks – created load balancer, groups and security group
r. Click close -> WebServerELB is created and will be in provisioning state. No need to wait

3. Configure Auto-scaling

a. Scroll down to auto scaling on left side and click launch configurations
b. Click create launch configuration
c. Set name as WebServerConfig
d. Scroll down to AMI and select WebServerAMI
e. In order to select instance type first go to ec2 instance -> note down instance type(t2.micro) and availability zone(us east 1a)
f. Go back to configuration -> select instance type as t2.micro
g. Scroll down and enable monitoring with Cloudwatch by selecting the checkbox
h. In security group click web security group and scroll down
i. Choose existing key pair (vockey) -> launch configuration
j. Select the created configuration -> click actions -> create auto scaling group
k. Set name as WebServerAutoScalingGroup and scroll down -> click next
l. Select VPC as Lab VPC
m. In subnets, select both the **private subnets** -> click next
n. Click on attach to existing load balancer -> scroll down
o. Select the WebServerGroup in existing load balancer target groups
p. Enable monitoring -> click next
q. Set group size: desired capacity=2, min capacity = 2, maximum capacity=6

r. In scaling policy select target tracking (will enable auto scaling group to monitor if CPU utilization rate is hitting to 60%
s. Rename scaling policy name as WebServerScalingPolicy
t. Set target value = 60 [once it reaches 60% the next server starts and the 40% is the buffer time for the next server to start]
u. Click next -> next -> add tag -> key = name and value = WebServerInstance -> next -> create auto scaling group -> refresh
   [When it reaches 60 the new instance is created with the name WebServerInstance]
v. Go to ec2 -> refresh -> 2 new instances are created (totally 4)
w. Go to target groups -> click on the group -> check if the state is healthy
x. Go to load balancer -> status is active -> copy dns name -> paste in browser -> cpu load is 0% [load can go max upto 100%]
y. Click service -> management and governance -> Cloudwatch -> alarms (on left side) -> all alarms -> there will be 2 alarms and its states keep on changing like ok, in alarm, insufficient data [will take some time to change]
z. Go to the browser displaying cpu load -> click on load test -> cpu load = 100%. [the page refreshed every 5 second]
aa. At the end there will be 6 instances totally
bb. Click on an alarm-> it shows graph of cpu utilization rate
cc. Terminate webserver 1 and all other instances will terminate

LAB 5: CREATE AND DEPLOY VM USING IAAS IN AZURE PLATFORM

Link:

https://drive.google.com/file/d/17827qhvaY-Yimbwjxkg606Ki-iax8JUz/view

Steps:

1. Go to Azure for students platform -> create a VM by clicking on Virtual Machine
2. In resource group click new and create a resource group [no special characters]
3. Give a name -> Region - east us -> Image – windows gen 2 -> VM size – DS1_v2
4. Create a username and password
5. Inbound ports – RDP (by default)
6. Click next -> select standard SSD -> select default encryption -> click next -> next
7. Click review and create -> click create
8. Inside resource group all the resource will be added (vm)
9. **To run vm – 3 ways:**

First Way

a. Click the virtual machine (5 MCA) -> click connect -> select RDP (for windows) / SSH(for linux) -> download RDP
b. Open the downloaded file -> enter username and password for connecting
c. Open server manager in vm

Second Way

a. Download and open Remote Desktop Connection Manger

Third Way

a. Click on cloud shell in the browser – 1st icon on the top right side
b. Click create storage -> bash terminal will open
c. Enter the following commands

AzureVM shell commands cloud computing

```
az group create \
--resource-group vmdemoCLI \
--location eastus

az vm create \
--name mylinux \
--resource-group vmdemoCLI \
--admin-username adminuser \
--generate-ssh-keys \
--image ubuntuLTS \
--location eastus

<!-- Put public address of newly created virtual machine after @-->
ssh adminuser@PbIP
Yes
logout
cd /home
ls
cd /home/asha_thampi/.ssh/
ls
```

d. In the browser click on the created vm -> click connect -> click SSH

e. Click on mylinux instance created -> copy ip address -> paste in putty -> download the key

10. Stop the vm and delete all the resource group

LAB 6: BUILD YOUR VPC AND LAUNCH A WEB SERVER

Link: Module 5 Lab 2

Task 1: Create your VPC

1. In the AWS Management Console, on the Services menu, click VPC.
2. Click Launch VPC Wizard
3. In the left navigation pane, click VPC with Public and Private Subnets (the second option).
4. Click select then configure:
    - VPC name: `Lab VPC`
    - Availability Zone: Select the *first* Availability Zone
    - Public subnet name: `Public Subnet 1`
    - Availability Zone: Select the *first* Availability Zone (the same as used above)
    - Private subnet name: `Private Subnet 1`
    - Elastic IP Allocation ID: Click in the box and select the displayed IP address
5. Click Create VPC -> click Ok

Task 2: Create Additional Subnets

1. In the left navigation pane, click Subnets. [First, you will create a second Public Subnet.]
2. Click Create subnet then configure:

    - VPC ID: `Lab VPC`
    - Subnet name: `Public Subnet 2`
    - Availability Zone: Select the *second* Availability Zone
    - IPv4 CIDR block: `10.0.2.0/24`

[The subnet will have all IP addresses starting with 10.0.2.x]

3. Click create subnet [second private subnet]
4. Click create subnet then configure:

    - VPC ID: `Lab VPC`
    - Subnet name: `Private Subnet 2`
    - Availability Zone: Select the *second* Availability Zone
    - CIDR block: `10.0.3.0/24`

    [The subnet will have all IP addresses starting with 10.0.3.x]
5. Click create subnet
6. In the left navigation pane, click Route Tables.
7. Select the route table with Main = Yes and VPC = Lab VPC. (Expand the *VPC ID* column if necessary to view the VPC name.)
8. In the lower pane, click the Routes tab. In the Name column for this route table, click the pencil then type `Private Route Table` and click save
9. In the lower pane, click the Subnet Associations tab.
10. Click edit subnet associations
11. Select both Private Subnet 1 and Private Subnet 2.
12. Click save associations
13. Select the route table with Main = No and VPC = Lab VPC (and deselect any other subnets).
14. In the Name column for this route table, click the pencil then type `Public Route Table`, and click save
15. In the lower pane, click the Routes tab.
16. Click the Subnet Associations tab.
17. Click edit subnet associations
18. Select both public subnet 1 and public subnet 2
19. Click save associations

Task 3: Create a VPC Security Group

1. In the left navigation pane, click Security Groups
2. Click create security group and then configure:

    - Security group name: `Web Security Group`
    - Description: `Enable HTTP access`
    - VPC: *Lab VPC*
3. In the Inbound rules pane, choose Add rule
4. Configure:
    - Type: *HTTP*
    - Source: *Anywhere*
    - Description: `Permit web requests`
5. Scroll to the bottom of the page and choose create security group

Task 4: Launch A Web Server Instance

1. On services menu click ec2
2. Click launch instance
3. Select amazon linux 2 -> t2.micro -> click next
4. Configure:
    - Network: *Lab VPC*
    - Subnet: *Public Subnet 2* (*not* Private!)
    - Auto-assign Public IP: *Enable*

5. Expand the Advanced Details section (at the bottom of the page).
6. Copy and paste this code into the User data box:

```
#!/bin/bash
# Install Apache Web Server and PHP
yum install -y httpd mysql php
# Download Lab files
```

```
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-ACCLFO-2/2-lab2-vpc/s3/lab-app.zip
unzip lab-app.zip -d /var/www/html/
# Turn on web server
chkconfig httpd on
service httpd start
```

7. Click add storage -> add tags -> click add tag
   Key = Name, value = web Server 1
8. Click next
9. Select existing security group
10. Select web security group
11. Click review and launch -> click continue -> click launch
12. Select existing key pair -> acknowledge -> launch instance -> view instance
13. Wait until Web Server 1 shows *2/2 checks passed* in the Status Checks column.
14. Copy the Public DNS (IPv4) value shown in the Description tab at the bottom of the page.
15. Open a new web browser tab, paste the Public DNS value and press Enter.

LAB 7: IAM USING MICROSOFT AZURE SERVICES

Link:

https://drive.google.com/file/d/1MpzmbCMQCwBAO9IJYW_Y24L-UaZj0OSH/view

To give different users different types of access i.e. if there are 2 users and need to give 1 the contributor role and other the reader role.

1. Go to azure
2. Sign in [Personal id] -> click on user image right top -> click azure portal
3. Click on Virtual Machine and create a vm
4. To add users click on Users on the left panel -> add user -> select create a user -> enter all the user details -> select auto generate password -> click show and copy the password ->click create. Add another User
5. Once the users are created -> click on each of them and note down their id -> click login with another account -> use the credentials of the users created -> 1st is TonyRoy -> login and update the password -> skip the protect account option ->there are no resource groups in it. Create a domain using the same account.
6. Go to azure active directory -> click create a resource -> in the search bar enter azure active directory -> click create -> next ->
   a. give an organization name [CUBLR],
   b. initial domain name [StarkT]
   c. location – India
7. sign into Tony's account again and click on the user icon -> click switch directory -> select the newly created directory
8. go to azure active directory – the role is global administrator
9. click manage tenants -> select CUBLR ->click delete -> all the validations will pass except one

10. click on the given link -> scroll down and click yes for azure access management -> click save and close
11. Refresh and all the validations will pass and all the tenants are deleted. [CUBLR is now deleted]
12. Re-login to Tony and switch to default directory
13. go to main account -> skip delete tenants
14. go to groups -> create a new group -> select security -> give a name -> click create
15. click the created group -> go to members -> click add members
16. search tony -> select
17. go to resource group -> click access control -> add -> select the role -> select members -> submit [this is role assignment for both the users]
18. Login to the users account and the resource group is available to both the users
19. Go to reader account and try to delete the resource group, will receive an error message
20. Go to contributor account and delete the resource group, it is successfully deleted
21. Go back to the original account and the resource group is not found [successfully deleted!!]
22. Delete the users and the groups created from the main account