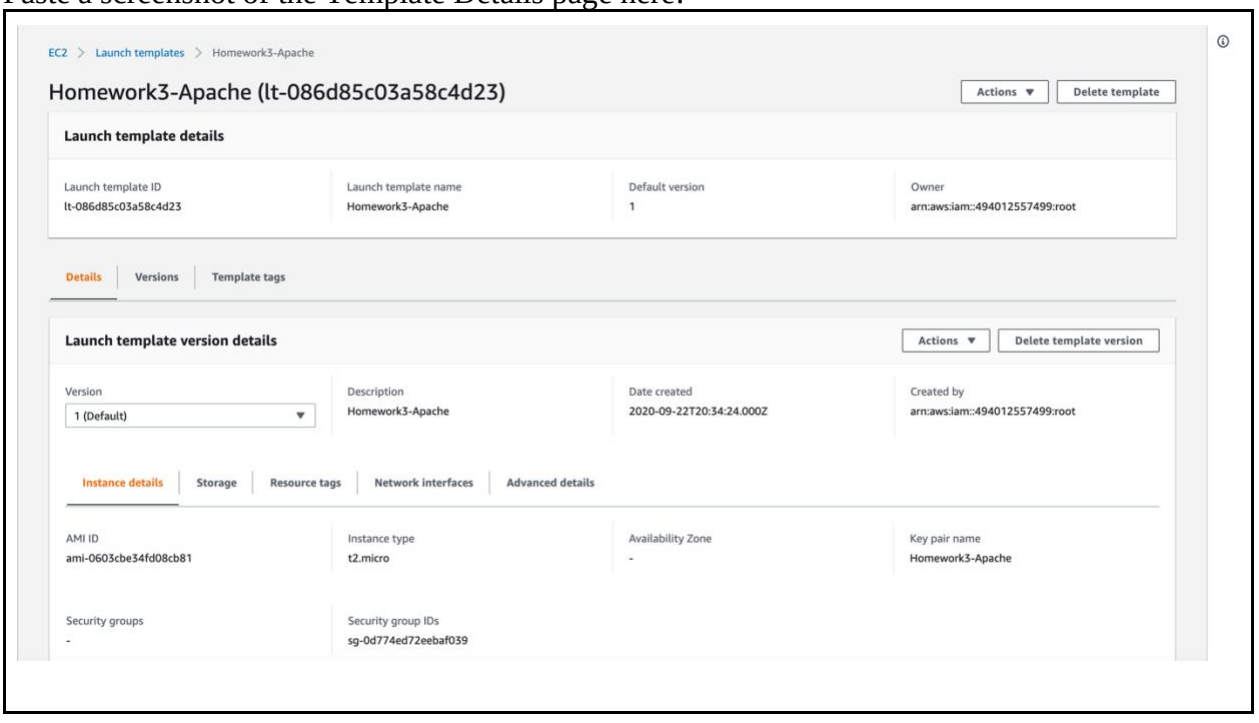
Rahul Ghosh

Assignment 3 - EC2 Templates, AMIs and EBS
Due: Sunday, Sept 27 11:59PM

Problem 1 - Template Configuration (15 Points)

- Create an EC2 template which can be used to automatically build an EC2 instance with the following characteristics:
    - Uses Amazon Linux 2 OS
    - Uses a t2.micro as your instance type
    - Uses a key pair that will allow you access your instance
    - Uses a security group which allows:
        - SSH access from your IP address
        - HTTP/HTTPS open to the world
    - User data which will perform the following operations:
        - Run system updates
        - Install Apache Webserver
        - Start Apache Webserver
        - Adds a very simple 'index.html' file as your default web page that displays a message of <your name>

Paste a screenshot of the Template Details page here:



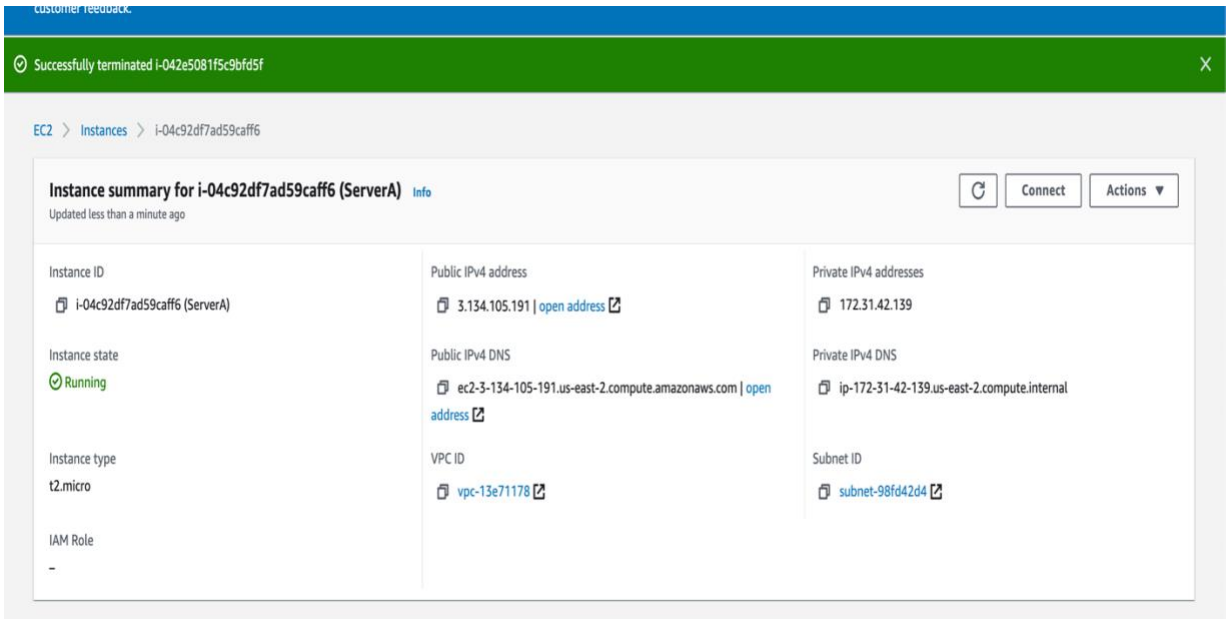
Paste your User Data script here:

```
#! /bin/bash
yum update -y
yum -y install httpd
service httpd start
echo "<h1>Rahul Ghosh's webpage is working YAY!!!!</h1>" >
/var/www/html/index.html
```

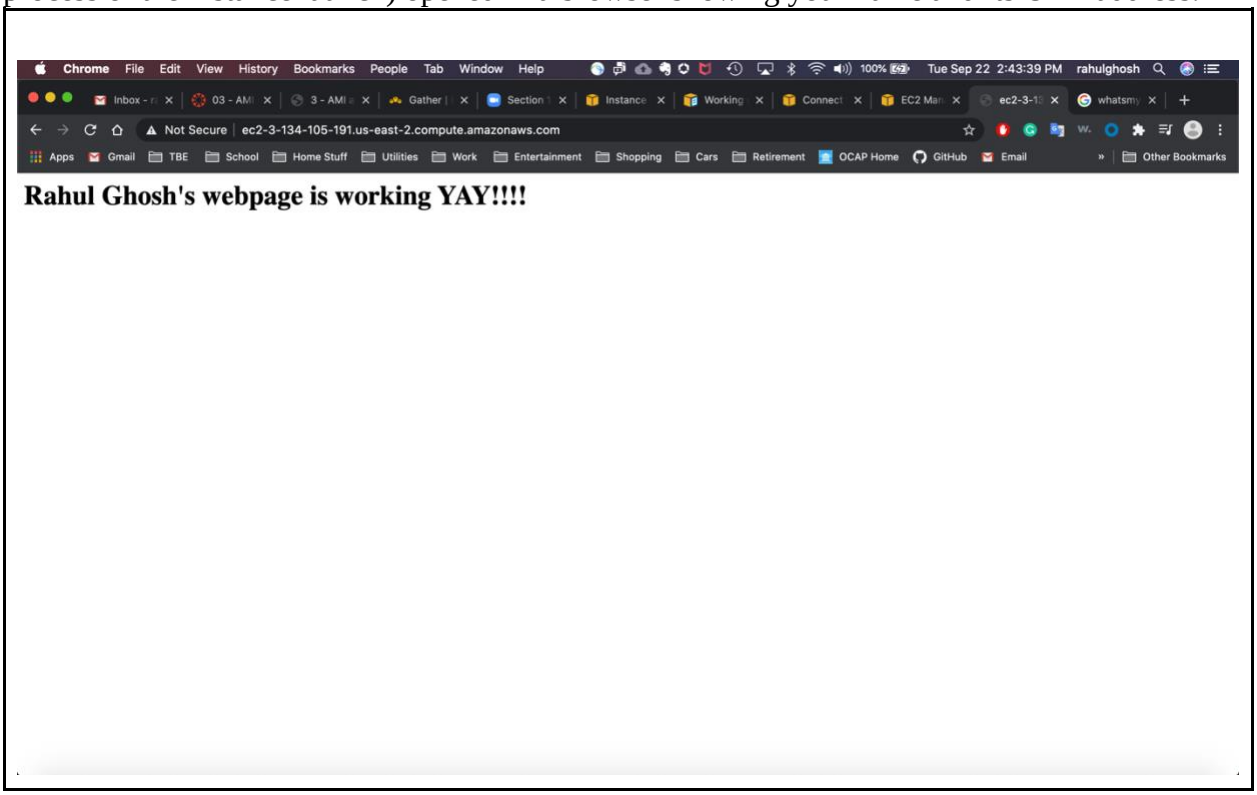
Problem 2 - AMI Creation (15 Points)

- Launch an EC2 instance from your Template and name it **Server A**

Paste a screenshot of the running EC2 Instance's Details of Server A (only the Instance Summary is necessary):

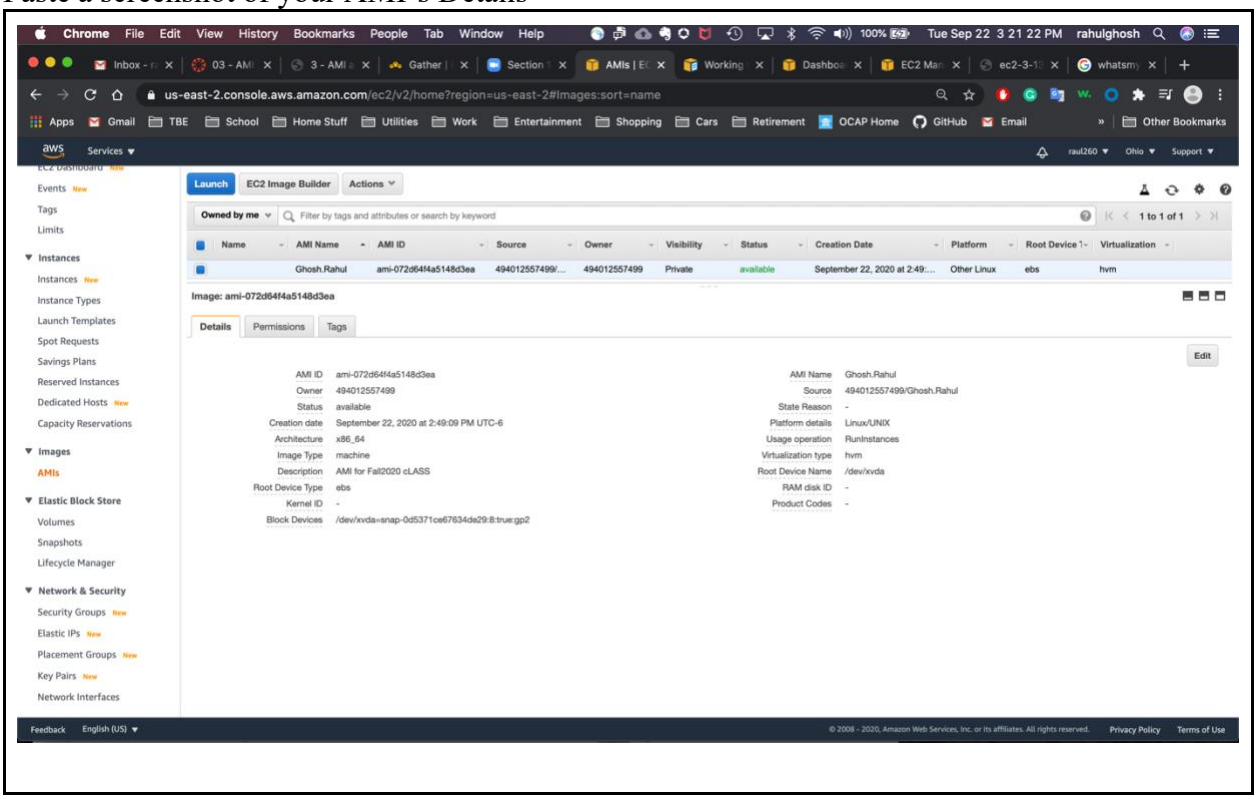


Paste a screenshot of the index.html (it was created by the user data in your template in the process of the instance launch) opened in a browser showing your name and its URL address.



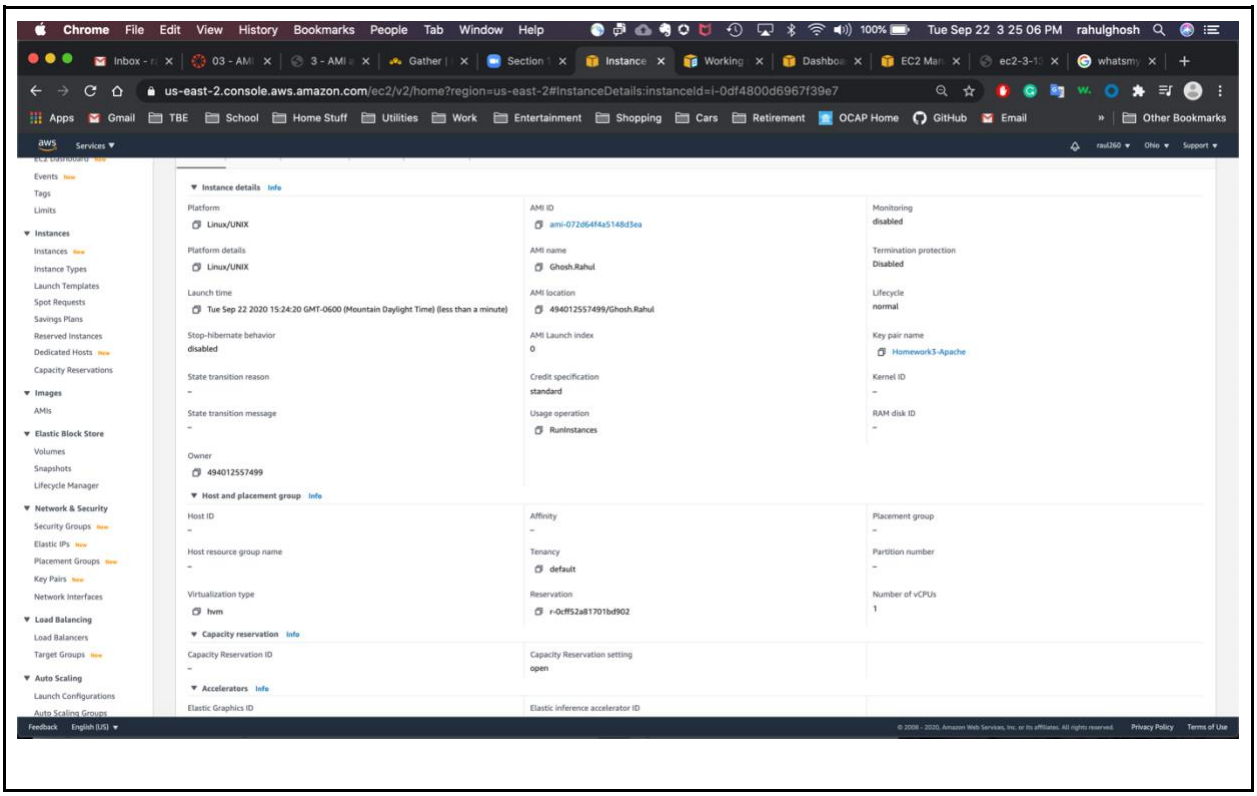
- Create an AMI from that instance. Set the name of your AMI as `<your last name>.<your first name>`. For example, mine would be named: `misicko.greg`

Paste a screenshot of your AMI's Details



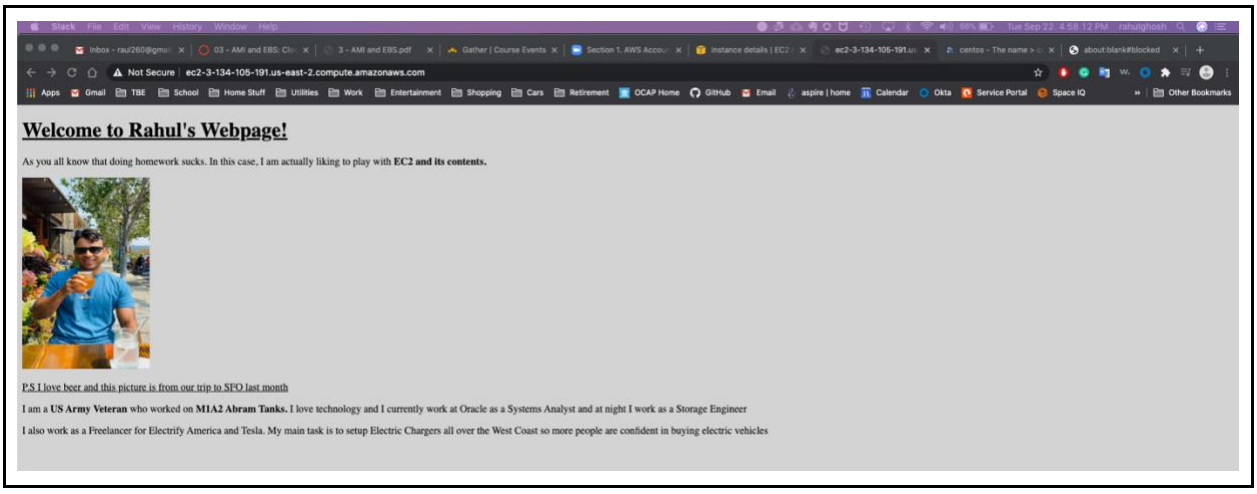
- Spin up a new instance of the machine from that AMI and name it **Server B**

Paste a screenshot of Server B's Instance Details:



- Create a webpage with your biography as its content. It should include
  - A description of yourself (fictitious is fine, we'll never know)
  - A photo of yourself (or a photo of what you'd like to represent your appearance)
- Copy the created webpage to the Server A and check that you can see that webpage on the Internet, using the Public DNS of the Server A.

Paste a screenshot of your Biography Page opened in a browser here, showing the URL bar with it:



- Edit the web page on **Server A** so that the image file is served up by **Server B**. **(*Hint:*** *You need to place your web page on* ***Server A*** *and the image file on* ***Server B*** *in a web-accessible location.***)**

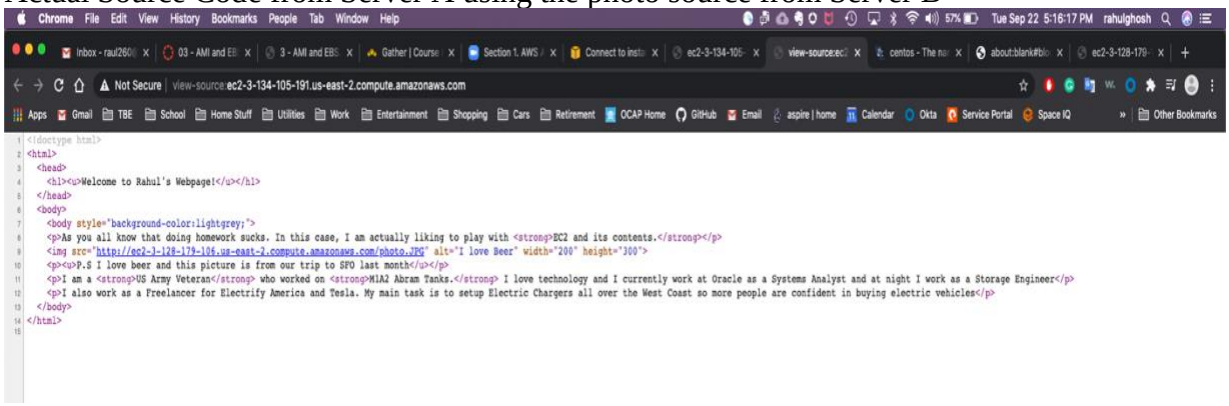
Paste the URL of your page on Server A:

http://ec2-3-134-105-191.us-east-2.compute.amazonaws.com/

Paste the URL of your image served from Server B:

http://ec2-3-128-179-106.us-east-2.compute.amazonaws.com/photo.JPG

Actual Source Code from Server A using the photo source from Server B

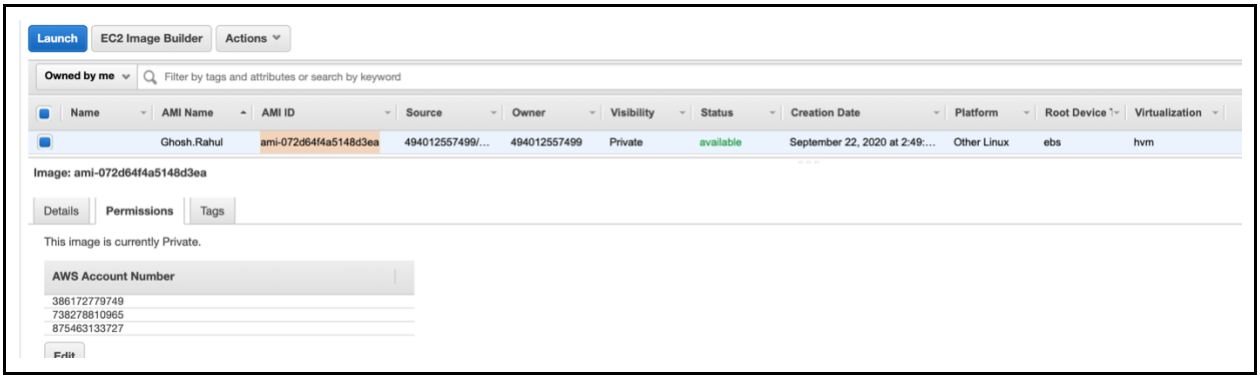


Problem 3: AMI Distribution (10 Points)

- The AMI you created in Problem 2 is private, by default. Keep it that way, but share it privately with the teaching staff. Their account IDs are:
  - Anthony: 738278810965
  - Sergei: 875463133727
  - Hector: 386172779749

Provide the AMI ID of your shared AMI (in case we can't identify you by name for some reason):

AMI ID: ami-072d64f4a5148d3ea

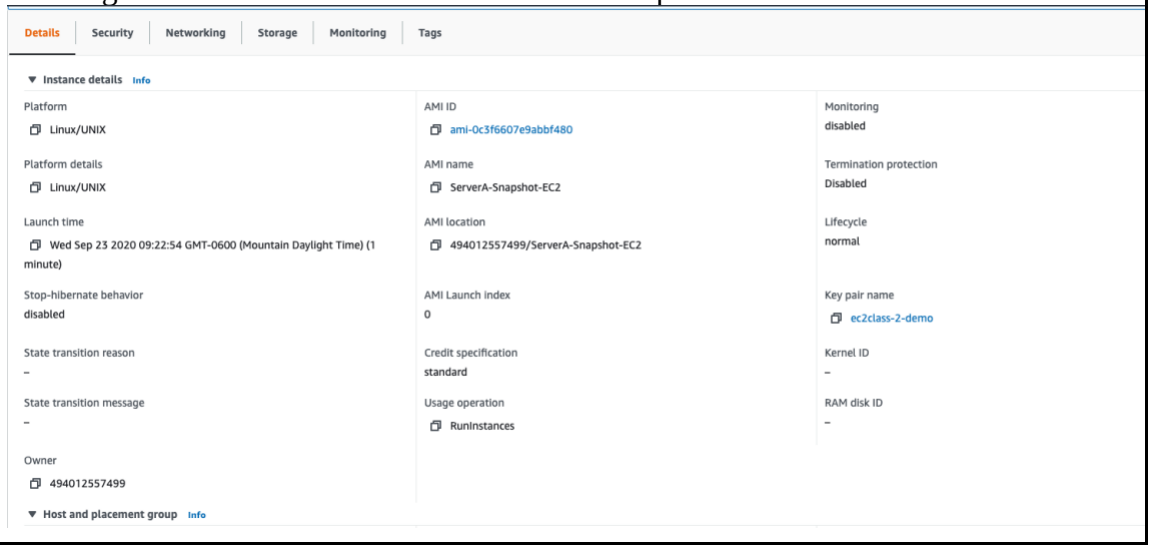


Problem 3: EBS (30 Points)

- Stop your Server A instance (don't Terminate it, just Stop it).
- Look under your Instance Storage Details tab, and you'll see the volume ID for your Root EBS volume. This is the hard drive of your EC2 computer. Click on that Volume ID to see it's details.
- Create a snapshot of this volume, name it "server-a-snapshot"

Explain how you would now launch an EC2 instance from this snapshot. This will be a multi-step process:

1. I will need to create an AMI from the Snapshot – Create Image >> Go to AMI
2. Once the AMI is created I will launch the EC2 instance from that AMI.>> Launch >>EC2
3. Running instance below that was created from Snapshot

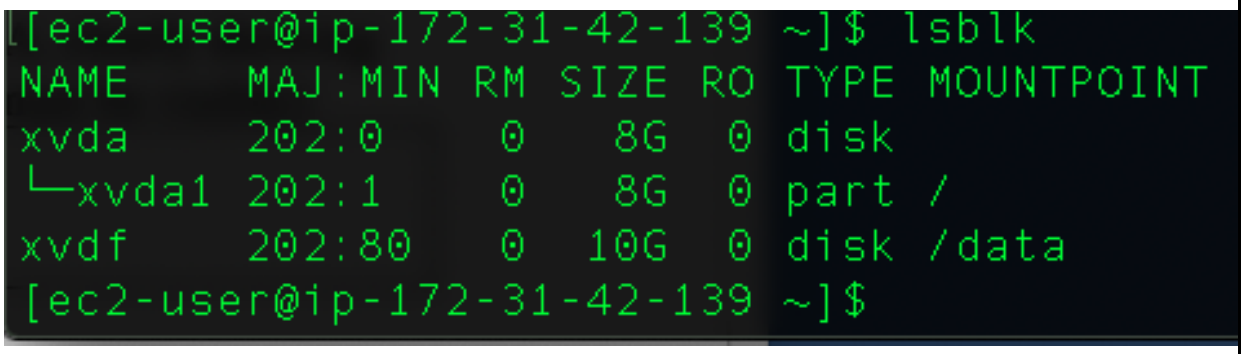


- Restart your Server A instance.
- Create a new EBS volume in the same availability zone as your EC2 instance, and name it “server-a-secondary”. (Hint: you name your Volume with a Tag, where the key is ‘Name’)
- Once your volume is created, attach it to Server A.

Now that you have an EBS Volume created and attached to Server A, you will need to connect to your server and mount the volume.
Once you connect to your EC2 instance, **what are the commands you’ll need to execute in order to mount and use that volume, and what does each command do**? (Hint: the solution to this is well documented at https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-using-volumes.html)

1. Connect to your ec2 instance
2. We will list the volumes by typing **lsblk** which shows the attached volumes (xvdf new volume)
3. Determine the volume and then run **sudo file -s /dev/xvdf (my volume name)**
4. The above command will tell us if the volume has any data or not (my case I didn’t)
5. If there is no data we will need to create one by running **sudo mkfs -t xfs /dev/xvdf**
6. Use this command to create a mount point directory for the volume **sudo mkdir /data**
7. Use this command to mount the volume at the directory **sudo mount /dev/xvdf /data**

Paste a screenshot of the running EC2 Instance's Storage Details for Server A, clearly showing the additional volume that you mounted (Root Devices and Block Devices must be visible):



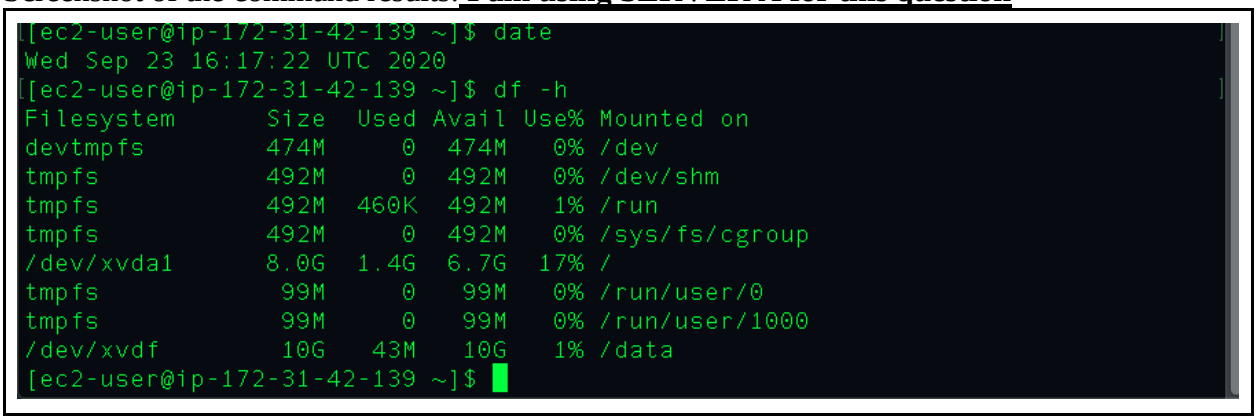
Problem 4: AMI Configuration (30 Points):

In real world scenarios, when running workloads on ec2 instances you might find your instance overloaded **or** under-utilized. In such cases you might want to change the instance type to suit your situation. Let's look at few very simple ways to view storage and memory utilization.

- Create an EC2 linux instance from your Template or use one from the above problems.
  - Identify disk-space allocated for root file system of the instance using the below command

```
df -h  - Root volume allocated 6.7G /dev/xvda1
```

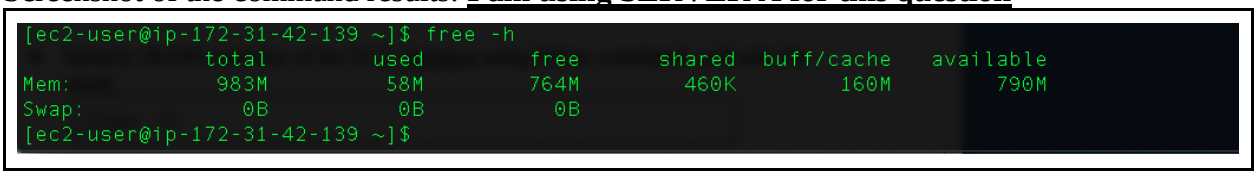
Screenshot of the command results: **I am using SERVER A for this question**



- Identify (RAM) memory of the EC2 instance using below command and note the result:

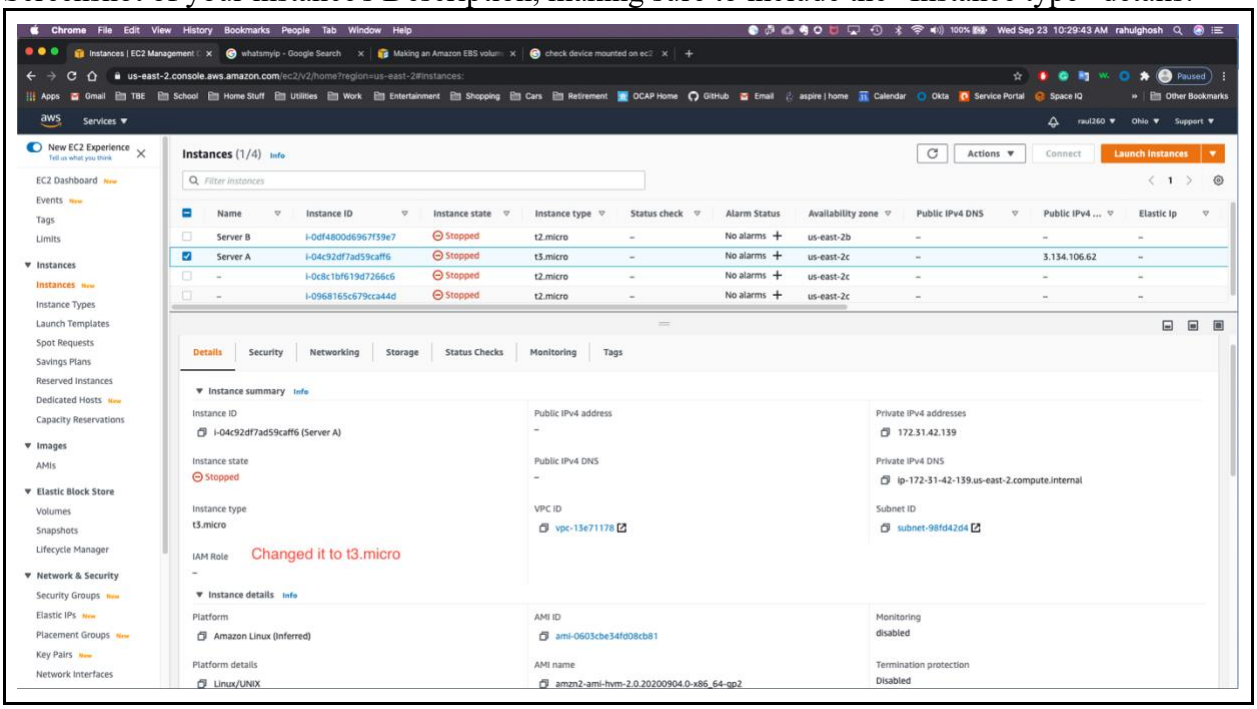
```
free -h
```

Screenshot of the command results: **I am using SERVER A for this question**



- From the AWS Console, go to EC2 dashboard → instances → select the instance
    - Stop the instance from the AWS Console.
    - From the “Actions” tab change the instance type from your t2.micro to a t3.micro; You are effectively upgrading your EC2 instance.

Screenshot of your instance's Description, making sure to include the “Instance type” details:

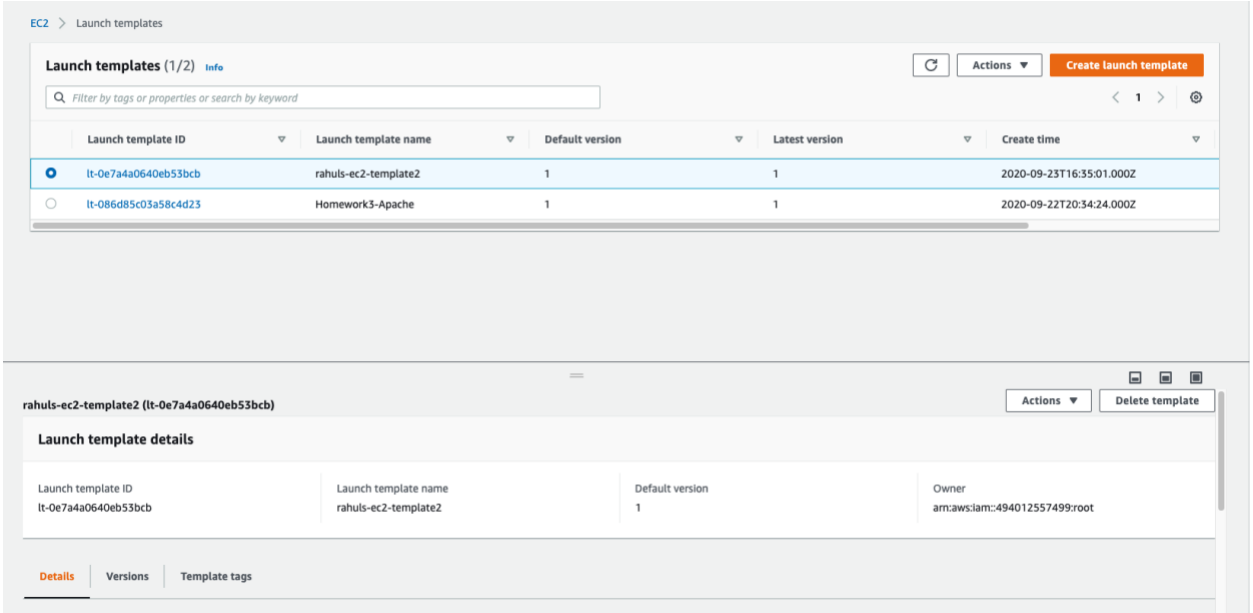


When you are finished:
To clean up your account and prevent costs from accumulating you can clean up (delete) your Instances and Snapshots. Leave your shared AMI available until your homework assignment is graded because the teaching staff will be verifying that you shared access with them.

Bonus

Bonus 1 (3 Points):

Add a second **version** of your Template from problem 1 that is identical to the first, but includes the installation of MySQL server as a step in your user data script. Show that you have multiple versions of your template, and also provide the updated User Data script.



User Data Script

```
#! /bin/bash
sudo yum update -y
sudo yum -y install httpd
sudo yum -y install mysql-server
sudo service mysqld start
sudo service httpd start
```

Bonus 2 (7 Points):
This one is probably pretty challenging, and will be covered a little later in the course: Creating index.html through the use of echo or cat commands isn't a very real-world scenario. Typically you would have a much more substantial website sitting in an S3 bucket, and would fetch the latest version of it when you start up your EC2 instance. How would you prepare a user data script which pulls a file named 'index.html' from an S3 bucket and copies it into /var/www/html ?

Before this script is executed you will need to make sure you have a S3 bucket configured and tested.
Follow the link to create an S3 Bucket:
https://docs.aws.amazon.com/AmazonS3/latest/gsg/CreatingABucket.html

USER DATA SCRIPT

```
sudo yum update -y
sudo yum install httpd -y
aws s3 cp s3://YOURBUCKETNAMEHERE/index.html /var/www/html/
aws s3 cp s3://YOURBUCKETNAMEHERE/healthcheck.html /var/www/html/
sudo service httpd start
chkconfig httpd on
```