Step 0: Obtain EC2 credentials.
For this step I followed the instruction provided to create an AWS account, choosing the appropriate region N. Virginia (us-east-1), and created a key pair p0KeyPair which is mentioned in the following steps below.

Step 1: install EC2 tools
On this step I faced some trouble with ec2 tools, and I used AWS CLI tools instead to go through the rest of the project.
To set up AWS CLI tools I created an Access Key and a Secret Key from the AWS website and used the command aws configure to set up the key, region, and output format
Console output for the command after setup:

```
aws configure
AWS Access Key ID [****************WRP3]:
AWS Secret Access Key [****************LH/j]:
Default region name [us-east-1]:
Default output format [json]:
```

I set up the environmental variables as described in the project instructions by running the commands:

```
export JAVA_HOME=/usr/lib/jvm/default-java
export EC2_HOME=Location of ec2-api-tools
export AWS_ACCESS_KEY=your-aws-access-key-id
export AWS_SECRET_KEY=your-aws-secret-key
export PATH=$PATH:$EC2_HOME/bin
```

then I tested my environmental by running aws ec2 describe-regions which showed all the regions used in AWS EC2
console output for the command:

```
aws ec2 describe-regions
{
    "Regions": [
        {
            "Endpoint": "ec2.ap-south-1.amazonaws.com",
            "RegionName": "ap-south-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-north-1.amazonaws.com",
            "RegionName": "eu-north-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-3.amazonaws.com",
            "RegionName": "eu-west-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-2.amazonaws.com",
            "RegionName": "eu-west-2",
            "OptInStatus": "opt-in-not-required"
        },
```

```
        {
```

Step 2: Create an instance and record its approximate starting time.
For this step I created an instance using the AMI provided by Amazon and the EC2 Key Pair I had created earlier. I noticed that when using the HVM Instance Store 64-bit AMI for us-east-1 region the CLI command fails, however, when using HVM (SSD) EBS-Backed 64-bit AMI instead the command works perfectly and starts an instance within roughly 2.73 seconds.
AMI used: ami-0ff8a91507f77f867.
Instance id: i-0d50af0fd2479c425.
Command and console output:

```
aws ec2 run-instances --image-id ami-0ff8a91507f77f867 --count 1 --
instance-type t2.micro --key-name p0KeyPair
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-0ff8a91507f77f867",
            "InstanceId": "i-0d50af0fd2479c425",
            "InstanceType": "t2.micro",
            "KeyName": "p0KeyPair",
            "LaunchTime": "2023-02-28T20:21:37+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-1e",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-31-58-144.ec2.internal",
            "PrivateIpAddress": "172.31.58.144",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
```

To obtain information about the instance I ran the command describe-instances which took 1.31 seconds to run
Command and console output:

```
aws ec2 describe-instances --instance-ids i-0d50af0fd2479c425
{
    "Reservations": [
        {
            "Groups": [],
```

```
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-0ff8a91507f77f867",
                    "InstanceId": "i-0d50af0fd2479c425",
                    "InstanceType": "t2.micro",
                    "KeyName": "p0KeyPair",
                    "LaunchTime": "2023-02-28T20:21:37+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1e",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-172-31-58-144.ec2.internal",
                    "PrivateIpAddress": "172.31.58.144",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-100-25-119-25.compute-
1.amazonaws.com",
                    "PublicIpAddress": "100.25.119.25",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "StateTransitionReason": "",
                    "SubnetId": "subnet-0a838b983ce7938ff",
                    "VpcId": "vpc-01ba5664b6a9e2fcb",
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2023-02-
28T20:21:38+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-0ea713968cbf88810"
                            }
                        }
                    ],
```


Step 3: SSH into newly created instance

For this step to obtain the file permissions needed for my key pair I use the command:
Chmod 400 p0KeyPair.pem before I sshed to my instance
Command and console output:

```
ssh -i p0KeyPair.pem ec2-user@100.25.119.25
The authenticity of host '100.25.119.25 (100.25.119.25)' can't be
established.
ED25519 key fingerprint is
SHA256:3DRfdKaish9jWj1sXnhzHQ6Lbaq7BM0m7TnlIki0uwU.
This key is not known by any other names
Are you sure you want to continue connecting
(yes/no/[fingerprint])? yes
Warning: Permanently added '100.25.119.25' (ED25519) to the list
of known hosts.

  __|  __|_  )
  _|  (     /   Amazon Linux AMI
 ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
37 package(s) needed for security, out of 60 available
Run "sudo yum update" to apply all updates.
```

I did face some difficulty trying to run the command multiple times because I was getting timed out even though I changed the settings in the security group. I solved this problem by deleting the default security group and creating a new group with the same settings.

Step 4: Install an application.
for this step I needed to add ‘sudo’ to the beginning of the command line to be guaranteed access.
This operation did not take a lot of time, it was instantly complete!
command and console output:

```
sudo yum install -y perl emacs
Loaded plugins: priorities, update-motd, upgrade-helper
Package 4:perl-5.16.3-294.43.amzn1.x86_64 already installed and latest
version
Resolving Dependencies
--> Running transaction check
---> Package emacs.x86_64 1:24.3-20.22.amzn1 will be installed
--> Processing Dependency: emacs-common = 1:24.3-20.22.amzn1 for
package: 1:emacs-24.3-20.22.amzn1.x86_64
--> Processing Dependency: libgnutls.so.26(GNUTLS_1_4)(64bit) for
package: 1:emacs-24.3-20.22.amzn1.x86_64
--> Processing Dependency: dejavu-sans-mono-fonts for package:
1:emacs-24.3-20.22.amzn1.x86_64
--> Processing Dependency: libgnutls.so.26()(64bit) for package:
1:emacs-24.3-20.22.amzn1.x86_64
--> Running transaction check
---> Package dejavu-sans-mono-fonts.noarch 0:2.33-6.6.amzn1 will be
installed
```

```
---> Package emacs-common.x86_64 1:24.3-20.22.amzn1 will be installed
---> Package gnutls.x86_64 0:2.12.23-21.18.amzn1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
==========
 Package                   Arch      Version                 Repositor
y    Size
================================================================================
==========
Installing:
 emacs                     x86_64    1:24.3-20.22.amzn1      amzn-
main    2.8 M
Installing for dependencies:
 dejavu-sans-mono-fonts    noarch    2.33-6.6.amzn1          amzn-
main    678 k
 emacs-common              x86_64    1:24.3-20.22.amzn1      amzn-
main     24 M
 gnutls                    x86_64    2.12.23-21.18.amzn1     amzn-
main    450 k

Transaction Summary
================================================================================
==========
Install  1 Package (+3 Dependent packages)

Total download size: 28 M
Installed size: 83 M
Downloading packages:
(1/4): dejavu-sans-mono-fonts-2.33-6.6.amzn1.noarch.rpm    | 678
kB   00:00
(2/4): emacs-24.3-20.22.amzn1.x86_64.rpm                   | 2.8
MB   00:00
(3/4): gnutls-2.12.23-21.18.amzn1.x86_64.rpm               | 450
kB   00:00
(4/4): emacs-common-24.3-20.22.amzn1.x86_64.rpm            |  24
MB   00:00
--------------------------------------------------------------------------------
----------
Total                                              49 MB/s |  28
MB  00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : dejavu-sans-mono-fonts-2.33-
6.6.amzn1.noarch                1/4
```

```
  Installing : 1:emacs-common-24.3-
20.22.amzn1.x86_64                          2/4
  Installing : gnutls-2.12.23-
21.18.amzn1.x86_64                               3/4
  Installing : 1:emacs-24.3-
20.22.amzn1.x86_64                                 4/4
  Verifying  : gnutls-2.12.23-
21.18.amzn1.x86_64                               1/4
  Verifying  : 1:emacs-common-24.3-
20.22.amzn1.x86_64                          2/4
  Verifying  : 1:emacs-24.3-
20.22.amzn1.x86_64                                 3/4
  Verifying  : dejavu-sans-mono-fonts-2.33-
6.6.amzn1.noarch                   4/4

Installed:
  emacs.x86_64 1:24.3-
20.22.amzn1

Dependency Installed:
  dejavu-sans-mono-fonts.noarch 0:2.33-
6.6.amzn1
  emacs-common.x86_64 1:24.3-
20.22.amzn1
  gnutls.x86_64 0:2.12.23-
21.18.amzn1

Complete!
[ec2-user@ip-172-31-58-144 ~]$
```

Step 5: Create a new Image from running instance.
Creating an image of the instance took about 1.56 seconds.
Instance id: i-0d50af0fd2479c425.
Image id: ami-0d689db484563c94d.
Command and console output:

```
aws ec2 create-image --instance-id i-0d50af0fd2479c425 --name
677_instimg
{
    "ImageId": "ami-0d689db484563c94d"
}
```

To obtain information about the new image I used describe-images command in combination with the image id. This operation took about 1.07.13 minutes.
Command and console output:

```
 aws ec2 describe-images | grep ami-0d689db484563c94d
            "ImageId": "ami-0d689db484563c94d",
```

Step 6: Create an instance of this new image and terminate all your instances.
Destroying the original instance took about 1.49 seconds.
Command and console output:

```
aws ec2 terminate-instances --instance-ids i-0d50af0fd2479c425
{
    "TerminatingInstances": [
        {
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "InstanceId": "i-0d50af0fd2479c425",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

This caused the status of the instance to change from running to terminated on the AWS EC2 website.
Creating a new instance from the customized ami took about 2.63 seconds
New instance id:
Command and console output:

```
aws ec2 run-instances --image-id ami-0d689db484563c94d --key-name
p0KeyPair --instance-type t2.micro
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-0d689db484563c94d",
            "InstanceId": "i-03b1e997d968d9a69",
            "InstanceType": "t2.micro",
            "KeyName": "p0KeyPair",
            "LaunchTime": "2023-02-28T20:51:21+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-1e",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-31-51-16.ec2.internal",
            "PrivateIpAddress": "172.31.51.16",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
```

```
                "Code": 0,
                "Name": "pending"
            },
```

To terminate this new instance I repeated the same termination command but with the new instance id. This took about 2.26 seconds.
Command and console output:

```
aws ec2 terminate-instances --instance-ids i-03b1e997d968d9a69
{
    "TerminatingInstances": [
        {
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "InstanceId": "i-03b1e997d968d9a69",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

This completely terminated the instance that was created.

Step 7: Compute the cost of this entire operation.

The price listed on the AWS pricing website for the t2.micro instance type is $0.0116 USD which is 0.000193 USD a minute and 0.0000032 USD a second. The S3 price per 1000 requests is $0.0004 USD which is 0.0000004 per instruction.
Given this information the price for the operation above is:
Creating an instance: 0.0000032 x 2.73 = 0.000008736 USD
Describing an instance: 0.0000032 x 1.31 = 0.000004192 USD
Creating an image: 0.0000032 x 1.56 = 0.000004992 USD
Describing an image: 0.000193 x 1.7 = 0.003281 USD
Terminating instance: 0.0000032 x 1.49 = 0.000004768 USD
Starting another instance from customized ami: 0.0000032 x 2.63 = 0.000008416 USD
Terminating new instance: 0.0000032 x 2.26 = 0.000007232 USD
Terminating instance

Overall, it took me an hour to complete all the previous steps and terminate the instance completely. which means the total charge would be **0.0116 + (0.0000004 x 8) ~= 0.116 USD** more or less depending on the exact number of minutes and seconds it took between the creation and termination of the instance.

Step 8: Cleaning up.
To clean up after I followed the instructions provided and using the appropriate CLI commands

Command for deregistering the image:

```
aws ec2 deregister-image --image-id ami-0d689db484563c94d
```

to completely remove the image, I needed to delete the snapshot that belongs to it. I ran describe-snapshots command and image id to obtain the snapshot id, however, the console only described the image and did not return the snapshot id.
Command and console output:

```
aws ec2 describe-snapshots | grep  ami-0d689db484563c94d
            "Description": "Created by CreateImage(i-
0d50af0fd2479c425) for ami-0d689db484563c94d",
```

I obtained the snapshot ID from the Snapshots tab on the EC2 website to delete the snapshot and the image completely.
Command and console output:

```
aws ec2 delete-snapshot --snapshot-id snap-062c48e8330551aaa
```

There was no output for this command.

Alia Mohamed

References

AWS CLI COMMAND Reference Documentation at https://docs.aws.amazon.com/cli/latest/index.html

Amazon EC2 on-Demand Pricing at https://docs.aws.amazon.com/cli/latest/index.html

Amazon S3 Pricing at https://aws.amazon.com/s3/pricing/

Amazon Linux AMI at https://aws.amazon.com/amazon-linux-ami/