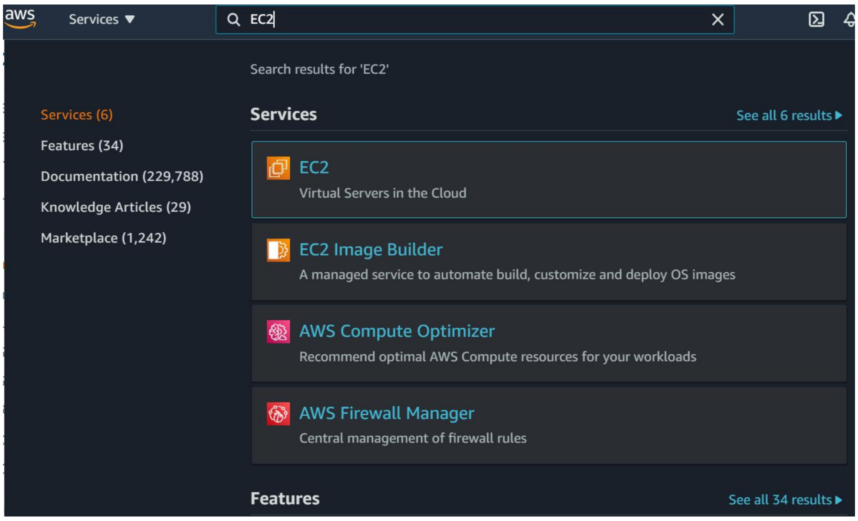
EC2 in action: creating an EC2 instance with Linux AMI

EC2 in action: Creating an EC2 instance with Linux AMI

by: **Rohit Sah**

First go to the EC2 section



Search results for 'EC2'

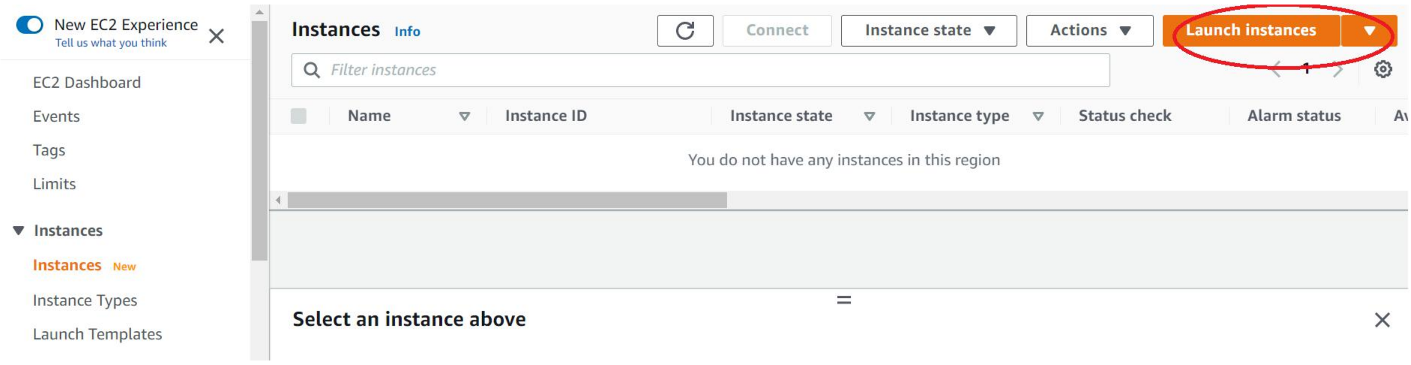
EC2 – Virtual Servers in the Cloud

EC2 Image Builder – A managed service to automate build, customize and deploy OS images

AWS Compute Optimizer – Recommend optimal AWS Compute resources for your workloads

AWS Firewall Manager – Central management of firewall rules

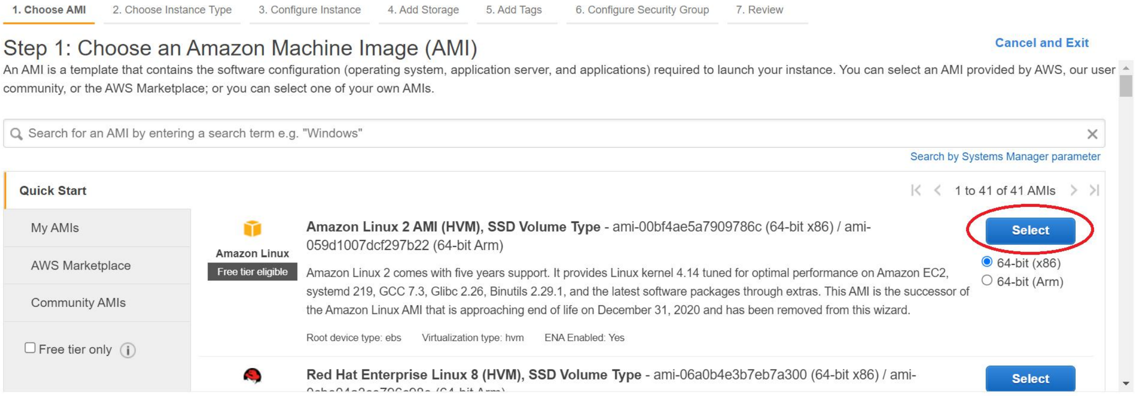
Then Click on Launch Instance



You do not have any instances in this region

Select an instance above

Select the AMI(Amazon Machine Image) that you want



Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Search for an AMI by entering a search term e.g. "Windows"

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-00bf4ae5a7909786c (64-bit x86) / ami-059d1007dcf297b22 (64-bit Arm)

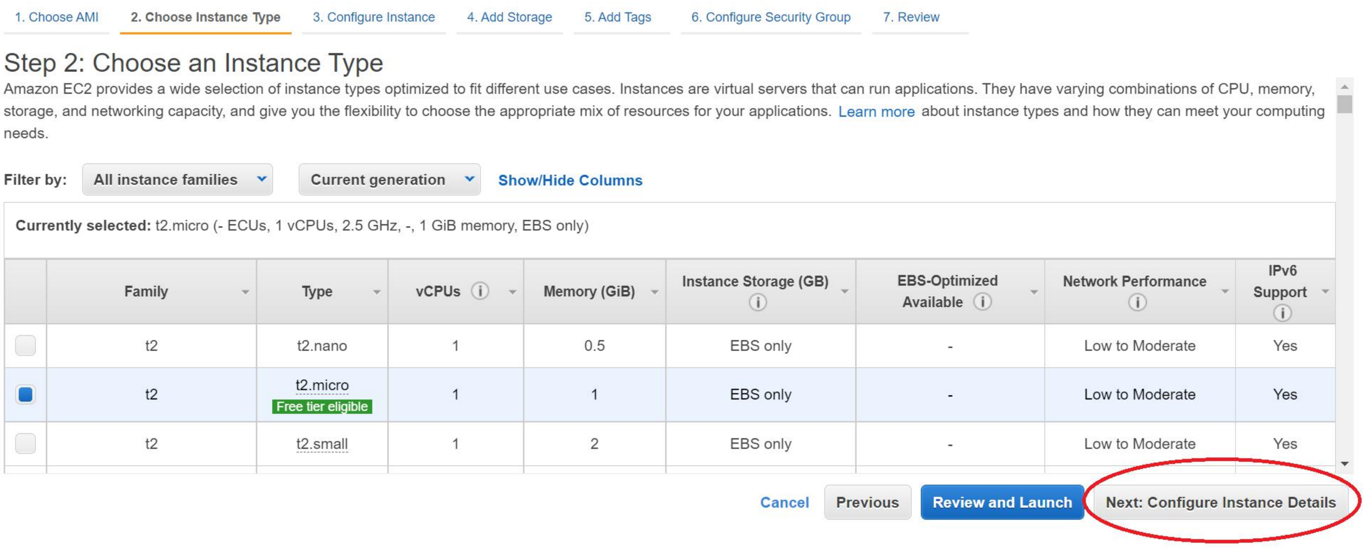
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is approaching end of life on December 31, 2020 and has been removed from this wizard.

Root device type: ebs   Virtualization type: hvm   ENA Enabled: Yes

● 64-bit (x86)
○ 64-bit (Arm)

Red Hat Enterprise Linux 8 (HVM), SSD Volume Type - ami-06a0b4e3b7eb7a300 (64-bit x86) / ami-

Select the Instance type or the machine that you want



Step 2: Choose an Instance Type

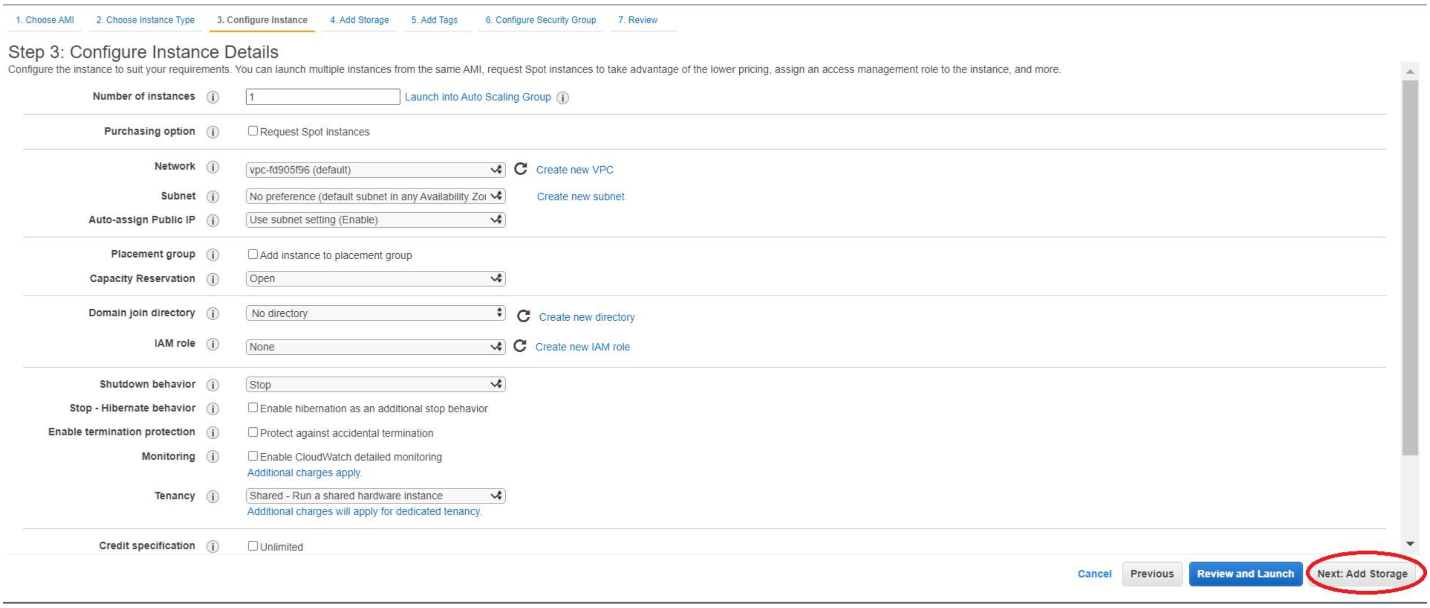
Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

Filter by: All instance families Current generation Show/Hide Columns

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
□	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
☑	t2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
□	t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes

Then configure the instance details



Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances: 1 Launch into Auto Scaling Group

Purchasing option: □ Request Spot instances

Network: vpc-fd905f96 (default) Create new VPC

Subnet: No preference (default subnet in any Availability Zoı Create new subnet

Auto-assign Public IP: Use subnet setting (Enable)

Placement group: □ Add instance to placement group

Capacity Reservation: Open

Domain join directory: No directory Create new directory

IAM role: None Create new IAM role

Shutdown behavior: Stop

Stop - Hibernate behavior: □ Enable hibernation as an additional stop behavior

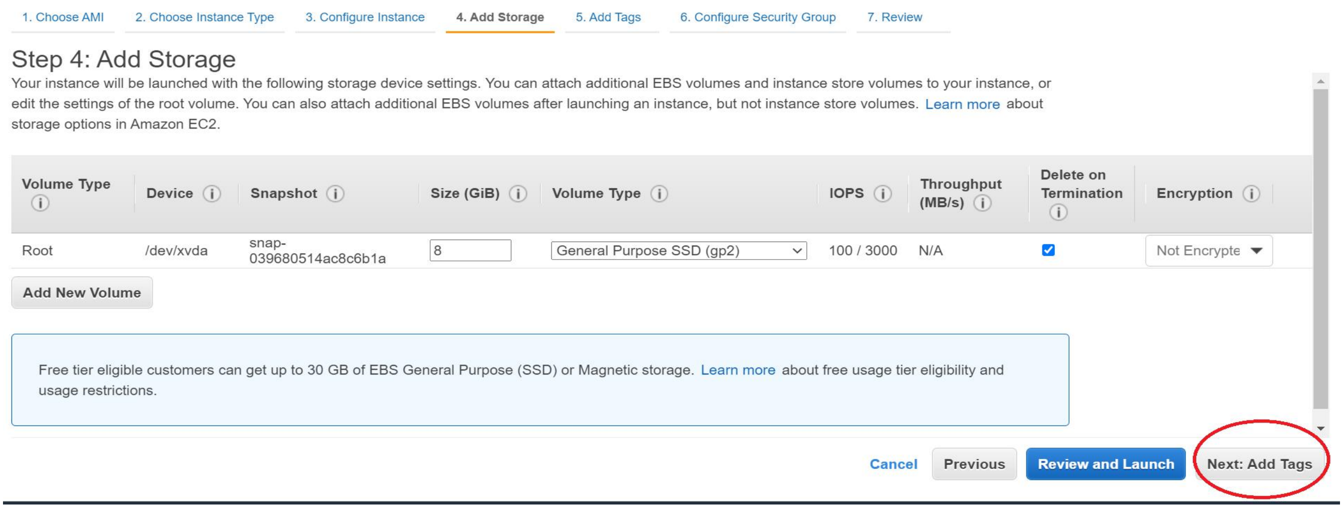
Enable termination protection: □ Protect against accidental termination

Monitoring: □ Enable CloudWatch detailed monitoring
Additional charges apply.

Tenancy: Shared - Run a shared hardware instance
Additional charges will apply for dedicated tenancy.

Credit specification: □ Unlimited

Select the storage details



Step 4: Add Storage

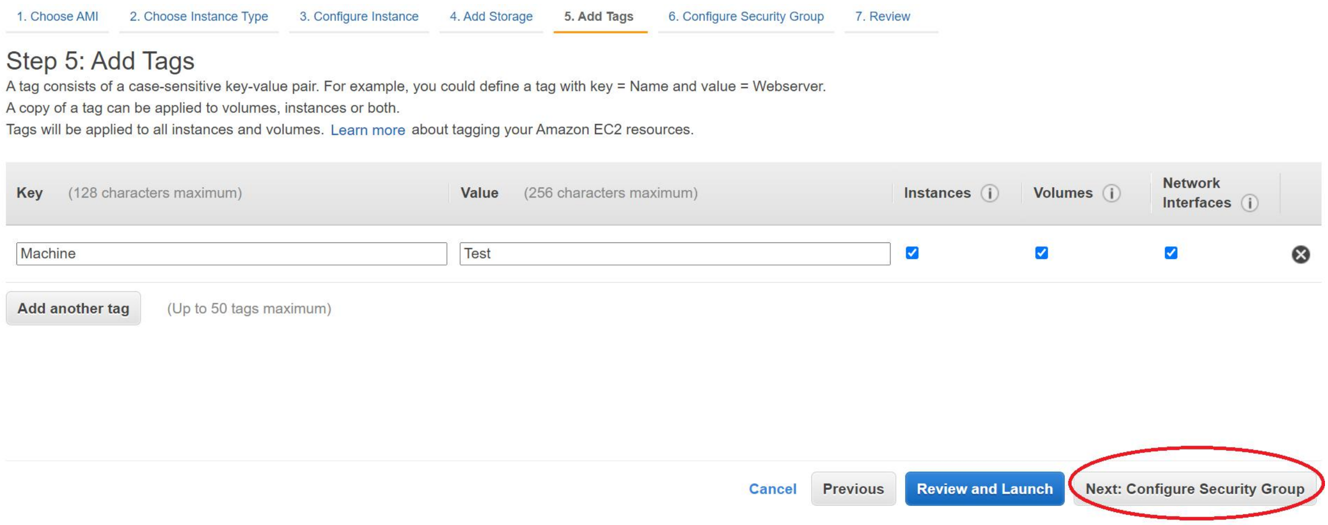
Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/xvda	snap-039680514ac8c6b1a	8	General Purpose SSD (gp2)	100 / 3000	N/A	☑	Not Encrypte

Add New Volume

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.

Add a tag to the EC2 machine but it is optional

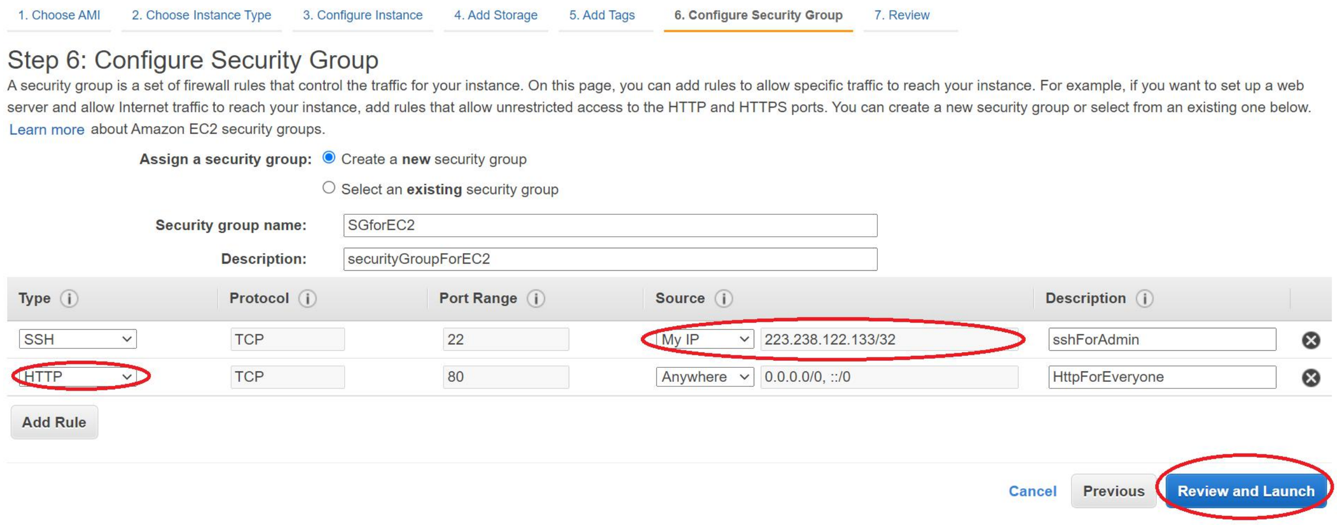


Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. Learn more about tagging your Amazon EC2 resources.

Key (128 characters maximum)	Value (256 characters maximum)	Instances	Volumes	Network Interfaces	
Machine	Test	☑	☑	☑	⊗

Now the most important part Is to configure the security groups,here we are setting an SSH from our own IP and HTTP from everywhere so everyone can access the machine through the IP but only we can SSH



Step 6: Configure Security Group

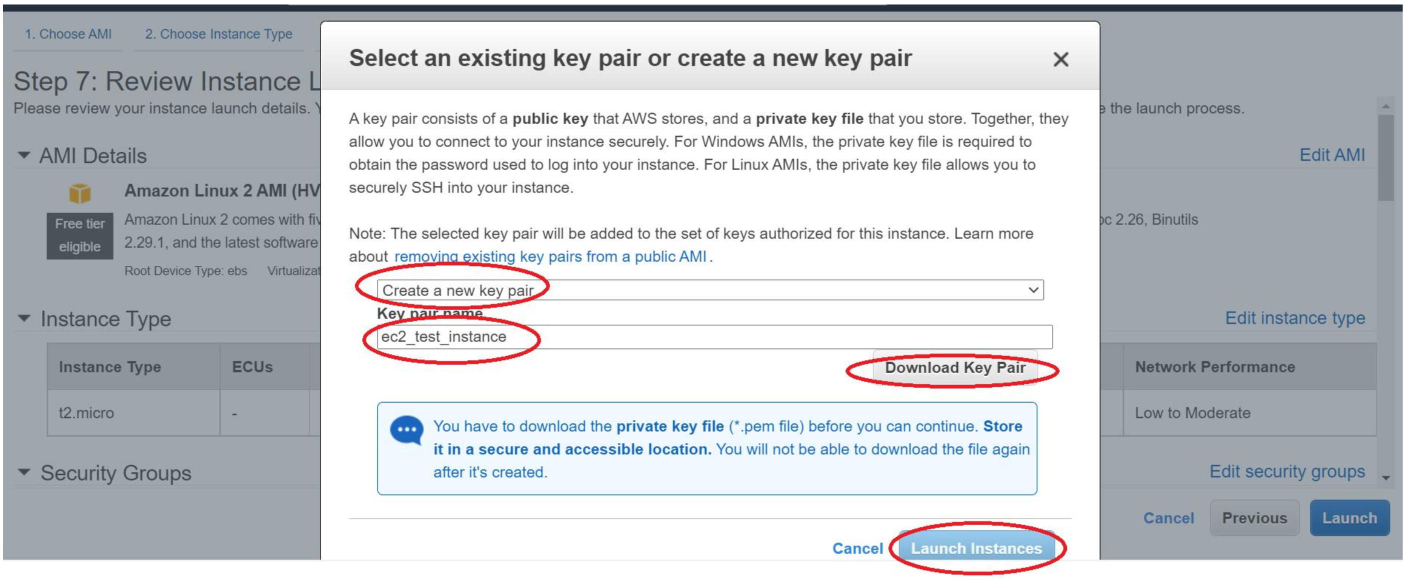
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: ● Create a **new** security group
○ Select an **existing** security group

Security group name: SGforEC2
Description: securityGroupForEC2

Type	Protocol	Port Range	Source		Description	
SSH	TCP	22	My IP	223.238.122.133/32	sshForAdmin	⊗
HTTP	TCP	80	Anywhere	0.0.0.0/0, ::/0	HttpForEveryone	⊗

Now we set a key pair and download it for ssh into our instance from our terminal



Step 7: Review Instance L
Please review your instance launch details. Y … the launch process.

AMI Details — Edit AMI

Amazon Linux 2 AMI (HV
Free tier eligible
Amazon Linux 2 comes with fiv … c 2.26, Binutils
2.29.1, and the latest software
Root Device Type: ebs Virtualizat

Instance Type — Edit instance type

Instance Type	ECUs	Network Performance
t2.micro	-	Low to Moderate

Security Groups — Edit security groups

Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

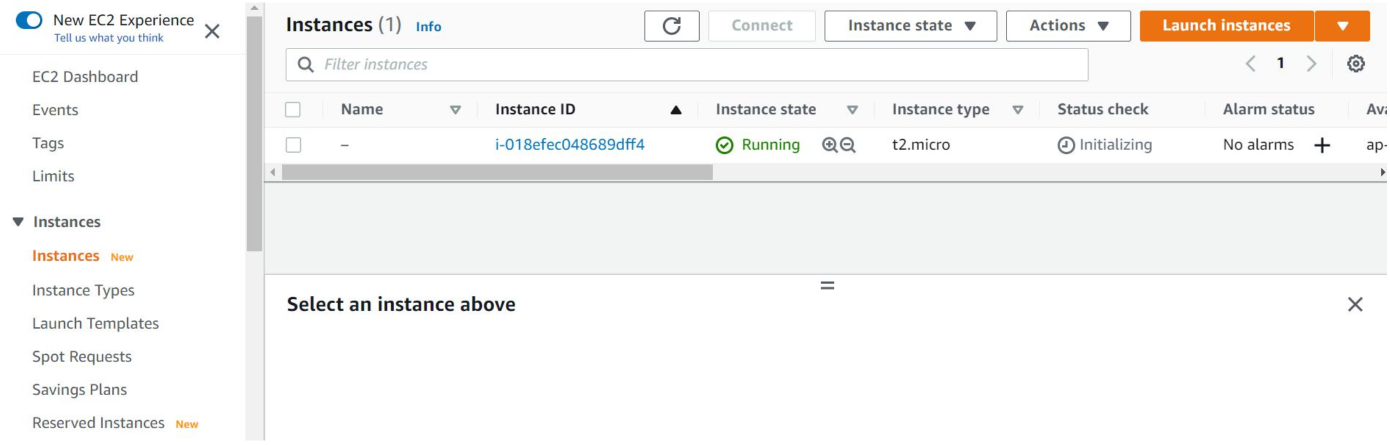
Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

Create a new key pair

Key pair name
ec2_test_instance

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

We can see the listed instance that we just created



New EC2 Experience
Tell us what you think

EC2 Dashboard
Events
Tags
Limits

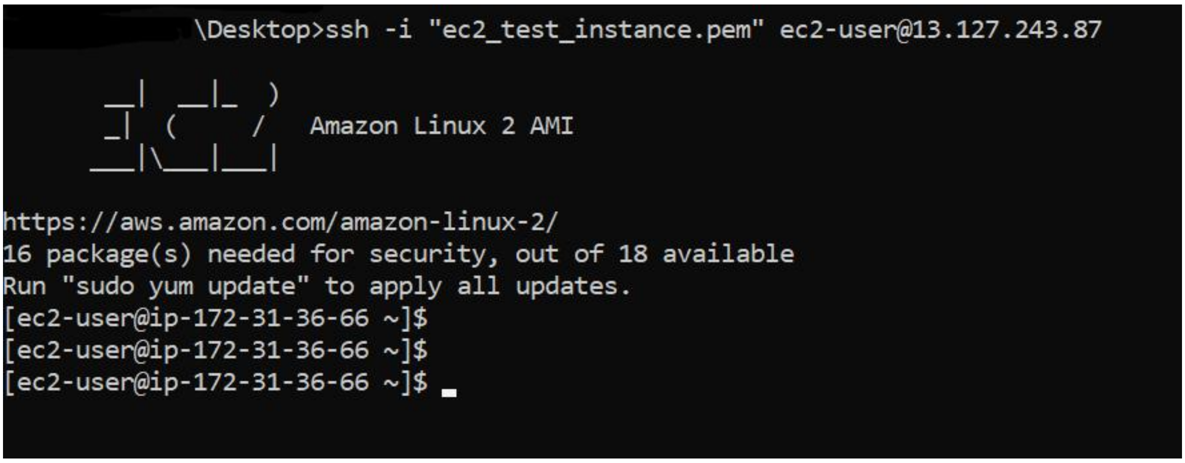
▼ Instances
Instances New
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances New

Instances (1) Info

□	Name	Instance ID ▲	Instance state	Instance type	Status check	Alarm status	Ava
□	–	i-018efec048689dff4	✓ Running	t2.micro	Initializing	No alarms	ap-

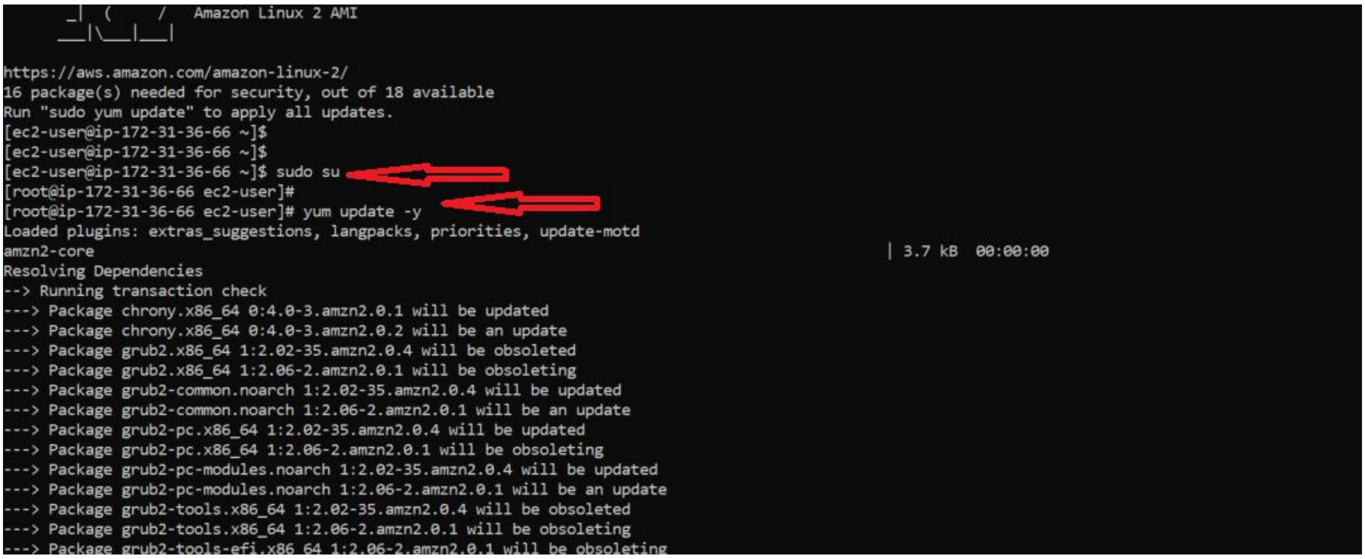
Select an instance above

Now to ssh from our local machine to the EC2 instance we got the terminal and enter the command as shown below with the user as ec2@user and at the public IP of the instance

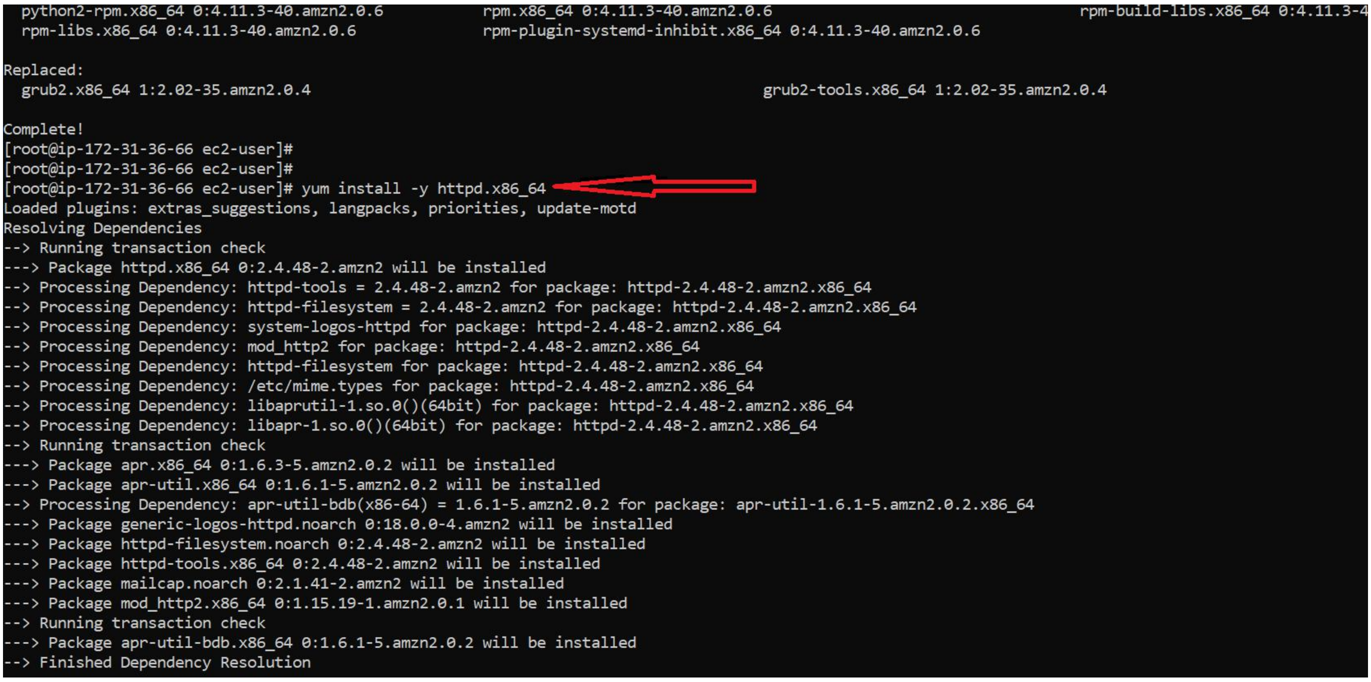


```
[redacted]\Desktop>ssh -i "ec2_test_instance.pem" ec2-user@13.127.243.87

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
16 package(s) needed for security, out of 18 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-36-66 ~]$
[ec2-user@ip-172-31-36-66 ~]$
[ec2-user@ip-172-31-36-66 ~]$ _
```

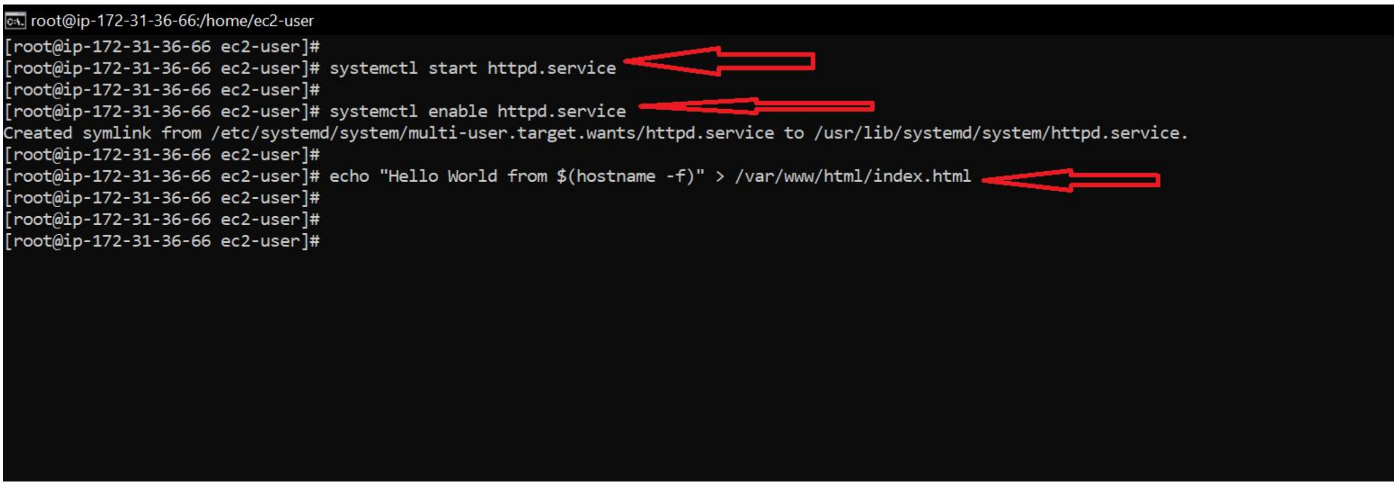
Now we will launch a web server from our instance so first we install all the dependencies



```
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
16 package(s) needed for security, out of 18 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-36-66 ~]$
[ec2-user@ip-172-31-36-66 ~]$
[ec2-user@ip-172-31-36-66 ~]$ sudo su
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]# yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                                     | 3.7 kB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package chrony.x86_64 0:4.0-3.amzn2.0.1 will be updated
---> Package chrony.x86_64 0:4.0-3.amzn2.0.2 will be an update
---> Package grub2.x86_64 1:2.02-35.amzn2.0.4 will be obsoleted
---> Package grub2.x86_64 1:2.06-2.amzn2.0.1 will be obsoleting
---> Package grub2-common.noarch 1:2.02-35.amzn2.0.4 will be updated
---> Package grub2-common.noarch 1:2.06-2.amzn2.0.1 will be an update
---> Package grub2-pc.x86_64 1:2.02-35.amzn2.0.4 will be updated
---> Package grub2-pc.x86_64 1:2.06-2.amzn2.0.1 will be obsoleting
---> Package grub2-pc-modules.noarch 1:2.02-35.amzn2.0.4 will be updated
---> Package grub2-pc-modules.noarch 1:2.06-2.amzn2.0.1 will be an update
---> Package grub2-tools.x86_64 1:2.02-35.amzn2.0.4 will be obsoleted
---> Package grub2-tools.x86_64 1:2.06-2.amzn2.0.1 will be obsoleting
---> Package grub2-tools-efi.x86_64 1:2.06-2.amzn2.0.1 will be obsoleting
```

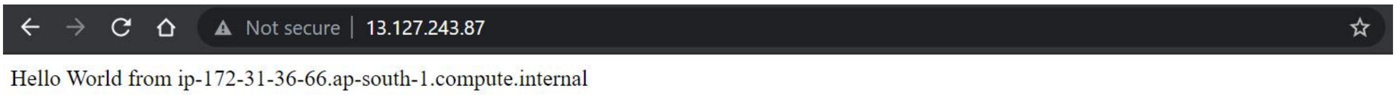


```
  python2-rpm.x86_64 0:4.11.3-40.amzn2.0.6           rpm.x86_64 0:4.11.3-40.amzn2.0.6                                 rpm-build-libs.x86_64 0:4.11.3-4
  rpm-libs.x86_64 0:4.11.3-40.amzn2.0.6              rpm-plugin-systemd-inhibit.x86_64 0:4.11.3-40.amzn2.0.6

Replaced:
  grub2.x86_64 1:2.02-35.amzn2.0.4                                                  grub2-tools.x86_64 1:2.02-35.amzn2.0.4

Complete!
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]# yum install -y httpd.x86_64
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.48-2.amzn2 will be installed
--> Processing Dependency: httpd-tools = 2.4.48-2.amzn2 for package: httpd-2.4.48-2.amzn2.x86_64
--> Processing Dependency: httpd-filesystem = 2.4.48-2.amzn2 for package: httpd-2.4.48-2.amzn2.x86_64
--> Processing Dependency: system-logos-httpd for package: httpd-2.4.48-2.amzn2.x86_64
--> Processing Dependency: mod_http2 for package: httpd-2.4.48-2.amzn2.x86_64
--> Processing Dependency: httpd-filesystem for package: httpd-2.4.48-2.amzn2.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.4.48-2.amzn2.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.4.48-2.amzn2.x86_64
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.4.48-2.amzn2.x86_64
--> Running transaction check
---> Package apr.x86_64 0:1.6.3-5.amzn2.0.2 will be installed
---> Package apr-util.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
--> Processing Dependency: apr-util-bdb(x86-64) = 1.6.1-5.amzn2.0.2 for package: apr-util-1.6.1-5.amzn2.0.2.x86_64
---> Package generic-logos-httpd.noarch 0:18.0.0-4.amzn2 will be installed
---> Package httpd-filesystem.noarch 0:2.4.48-2.amzn2 will be installed
---> Package httpd-tools.x86_64 0:2.4.48-2.amzn2 will be installed
---> Package mailcap.noarch 0:2.1.41-2.amzn2 will be installed
---> Package mod_http2.x86_64 0:1.15.19-1.amzn2.0.1 will be installed
--> Running transaction check
---> Package apr-util-bdb.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
--> Finished Dependency Resolution
```



```
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]# systemctl start httpd.service
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]# systemctl enable httpd.service
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]# echo "Hello World from $(hostname -f)" > /var/www/html/index.html
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]#
[root@ip-172-31-36-66 ec2-user]#
```

Now as we have installed and configured our web server we can view it on the HTTP rule from the browser



Hello World from ip-172-31-36-66.ap-south-1.compute.internal

So we have successfully deployed a httpd web-server on our Linux machine and can reach to our site through the IP or the DNS