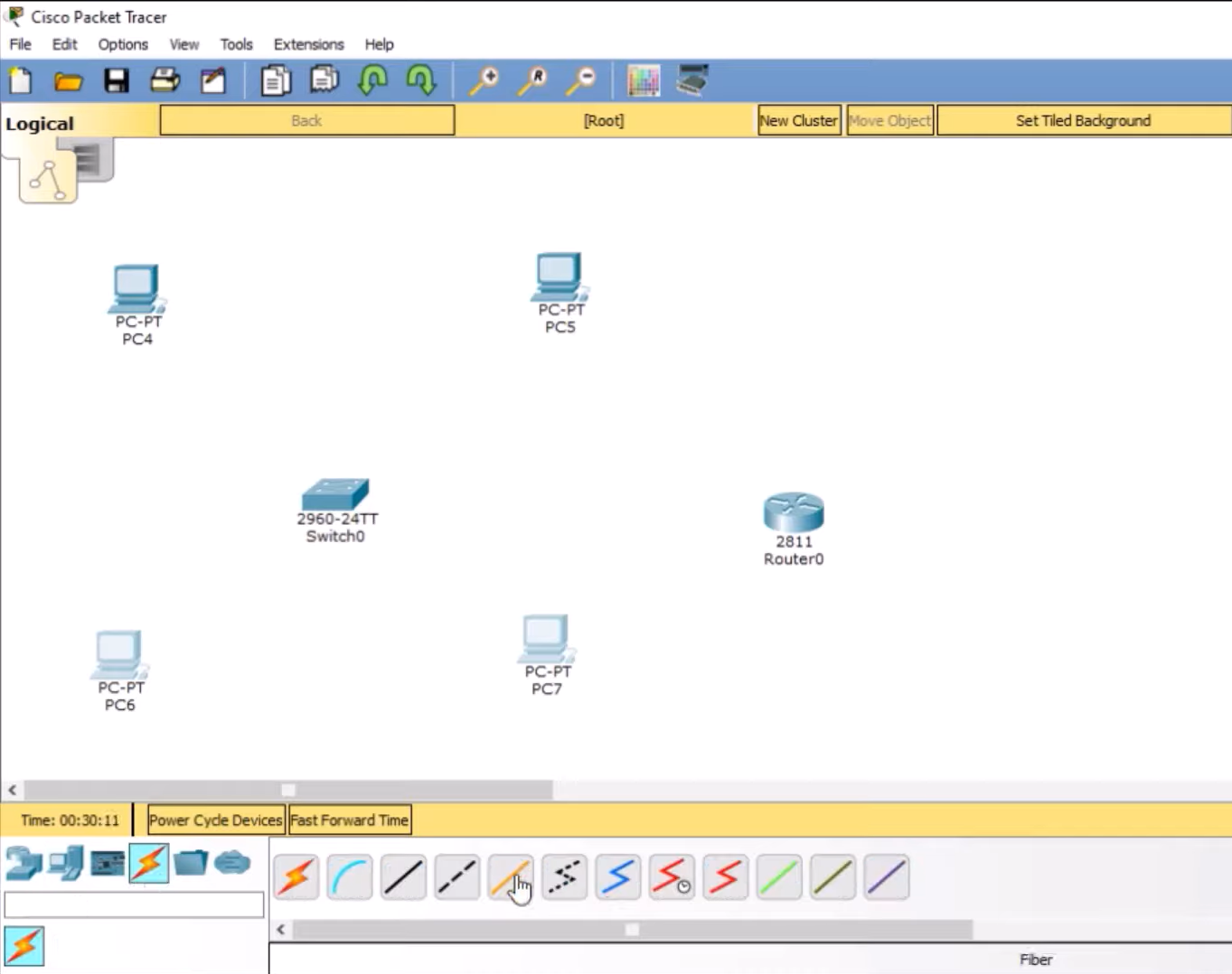
* **Cloud computing**

[[1]](#footnote-1) **Cisco simulator - VLAN design, Routing, Subnetting, Gateway configuration**

ans.

1. Add Devices:
   * Drag and drop a switch (e.g., 2960) and several PCs onto the workspace.



2. Create VLANs:

* Enter the following commands to create VLANs:

#enable

configure terminal

vlan 10

name VLAN10

exit

vlan 20

name VLAN20

exit#

3. Assign Ports to VLANs:

* Assign specific ports to each VLAN.

#interface range fa0/1 - 2

switchport mode access

switchport access vlan 10

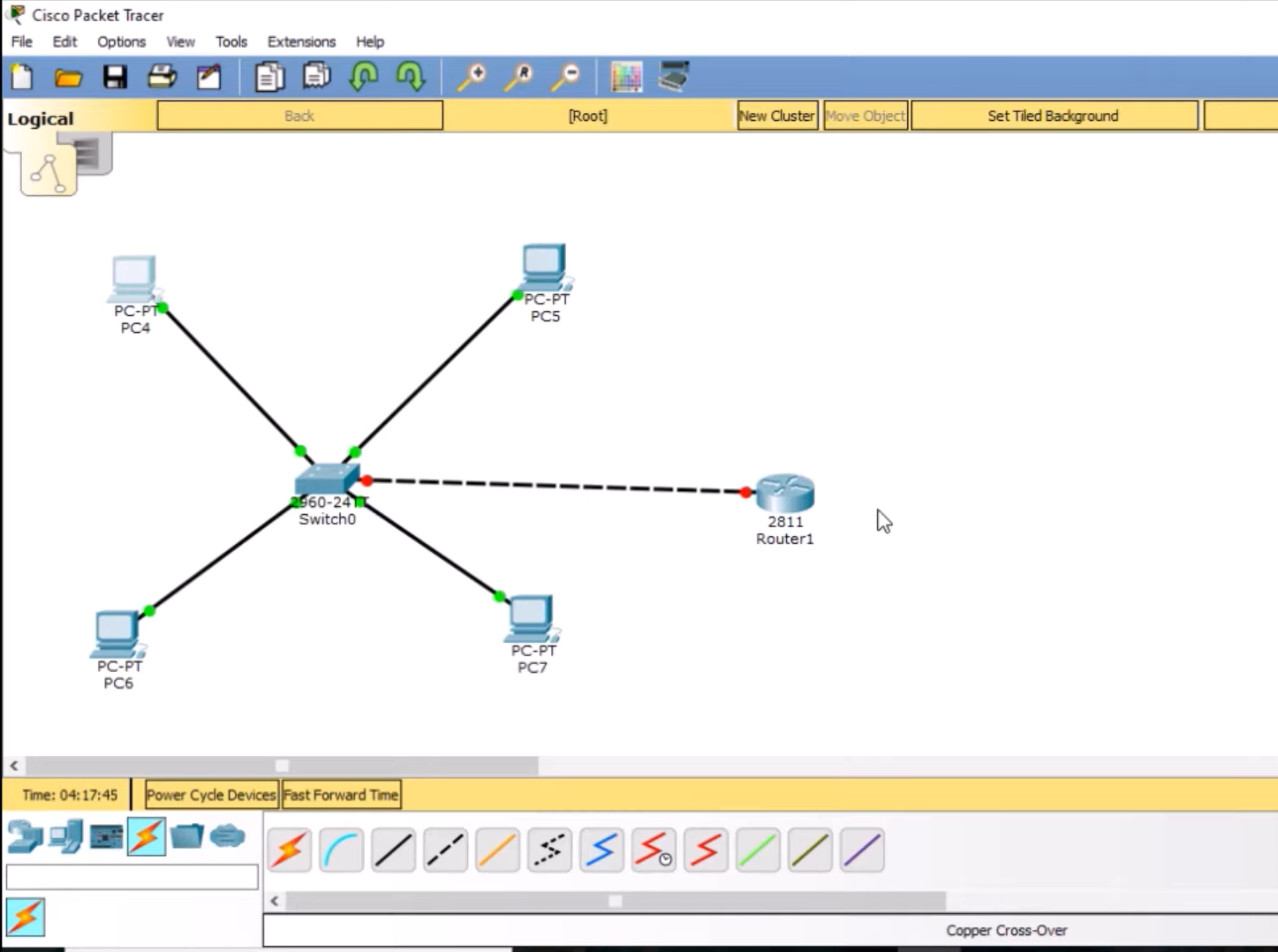
exit

interface range fa0/3 - 4

switchport mode access

switchport access vlan 20

exit#



4. Configure Router Interfaces:

* + Access the router's CLI and configure sub-interfaces for each VLAN:

enable

configure terminal

interface gig0/0.10

encapsulation dot1Q 10

ip address 192.168.10.1 255.255.255.0

exit

interface gig0/0.20

encapsulation dot1Q 20

ip address 192.168.20.1 255.255.255.0

exit

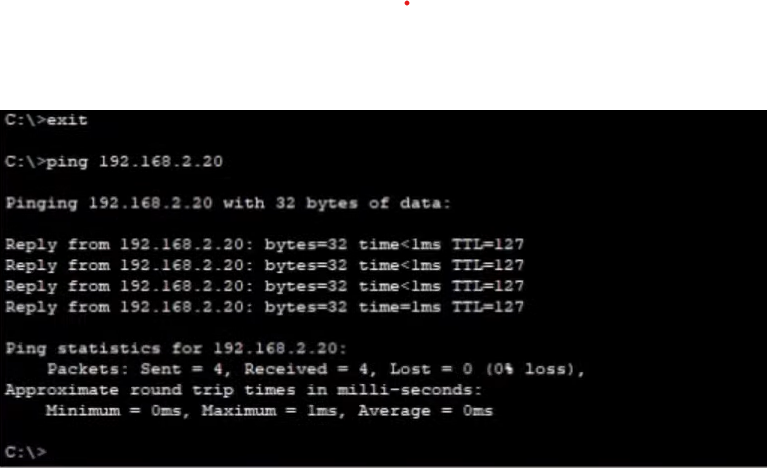
interface gig0/0

no shutdown

1. Assign IP Addresses:
   * For PCs in VLAN 10 (e.g., PC1 and PC2):PC1: IP Address: 192.168.10.2 Subnet Mask: 255.255.255.0
   * Default Gateway: 192.168.10.1

2. For PCs in VLAN 20 (e.g., PC3 and PC4):PC3: IP Address: 192.168.10.3

* + Subnet Mask: 255.255.255.0
  + Default Gateway: 192.168.10.1



**2. Virtual box-based Web Server creation, Images/Snapshots access webpage from 2nd VM on another subnet work**

**Ans.**

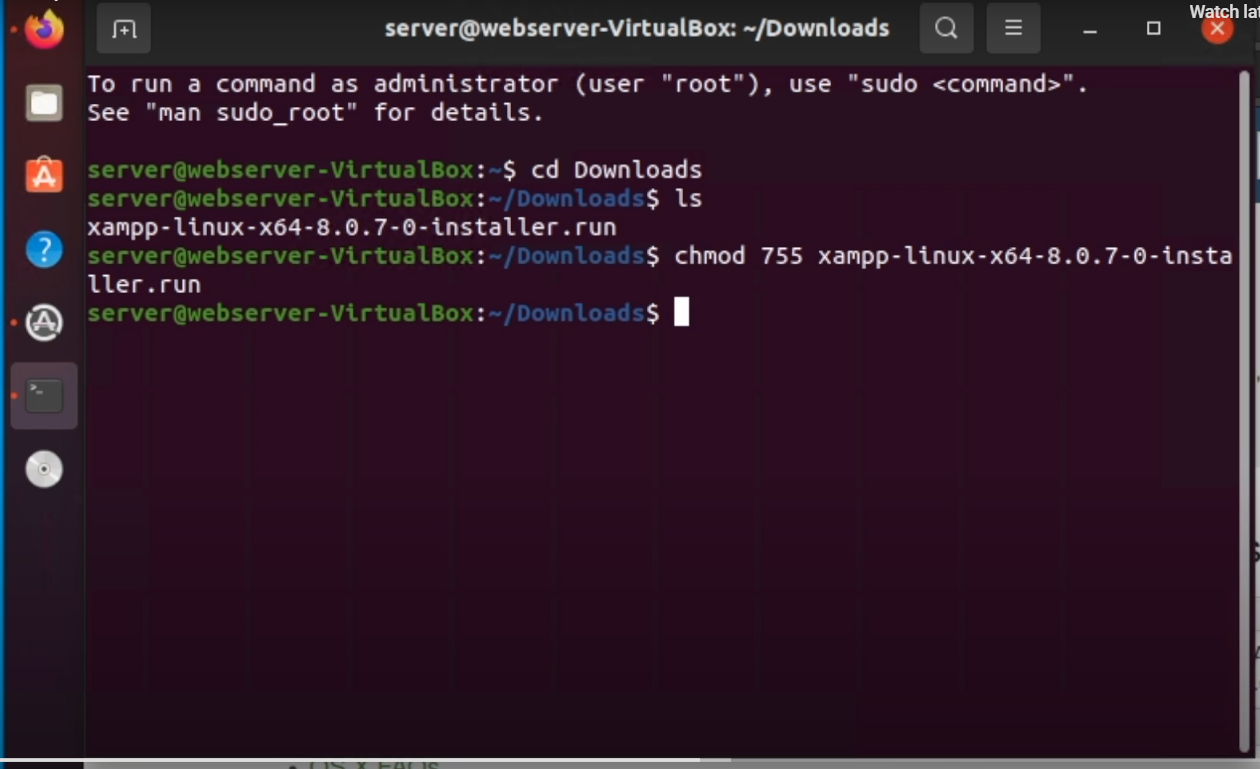
## Create a New Virtual Machine

1. Select your new VM, then click on Settings.
2. Go to the Storage tab:
   * Click on the empty CD/DVD icon under Storage Devices.
   * On the right side, click on the CD icon and select "Choose a disk file".
   * Browse to your downloaded Ubuntu Server ISO and select it.

## Install Ubuntu Server

## Access Your Web Server

1. After installation, reboot your VM.
2. Log in using your credentials.
3. Update your package list:

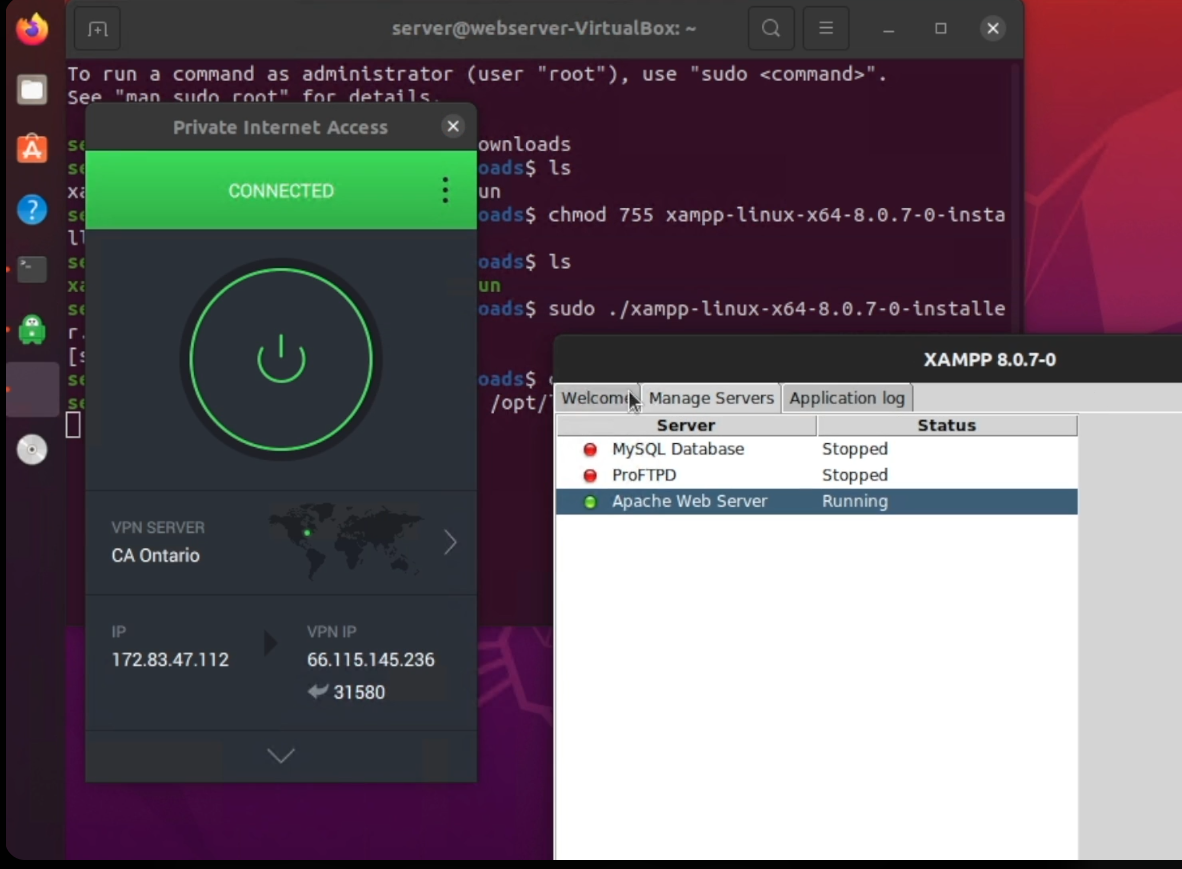
#sudo apt update

Install a web server package (e.g., Apache):

#sudo apt install apache2

#sudo systemctl start apache2

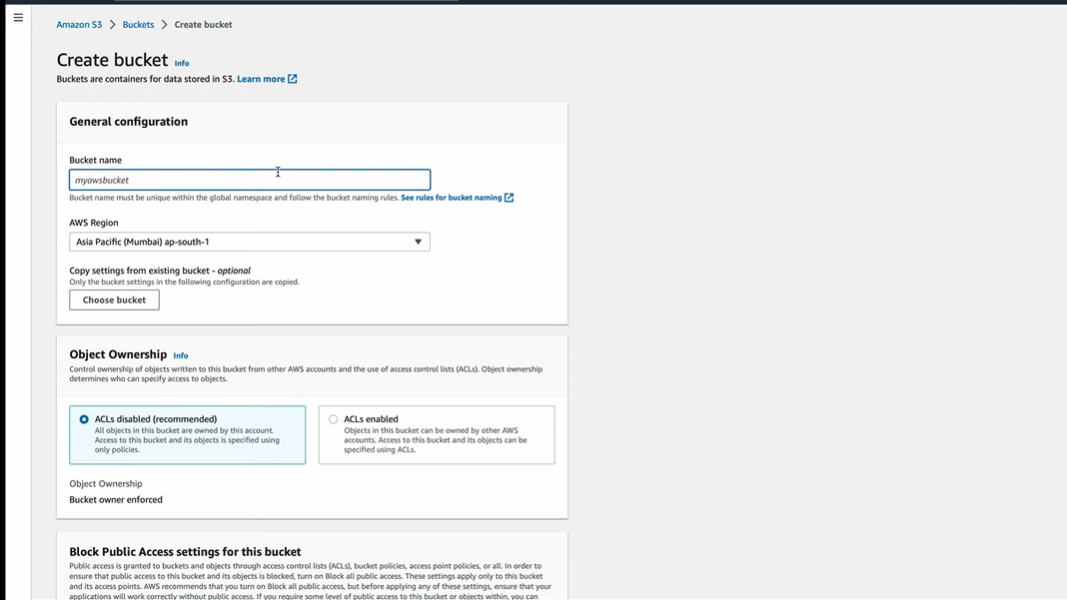
#sudo systemctl enable apache2



**3.EC2 AWS-S3 bucket based static web pages**

Create a Bucket:

* + Click on Create bucket.
  + Enter a unique bucket name
  + Choose a region close to your target audience to minimize latency.
  + Keep the default settings for Block Public Access unless you need to allow public access for your website.
  + Click Create bucket to finalize.



## Configure Bucket Policy

## Set Permissions:

* + Go to the Permissions tab.
  + Click on Bucket Policy and add a policy that allows public access:

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "PublicReadGetObject",

"Effect": "Allow",

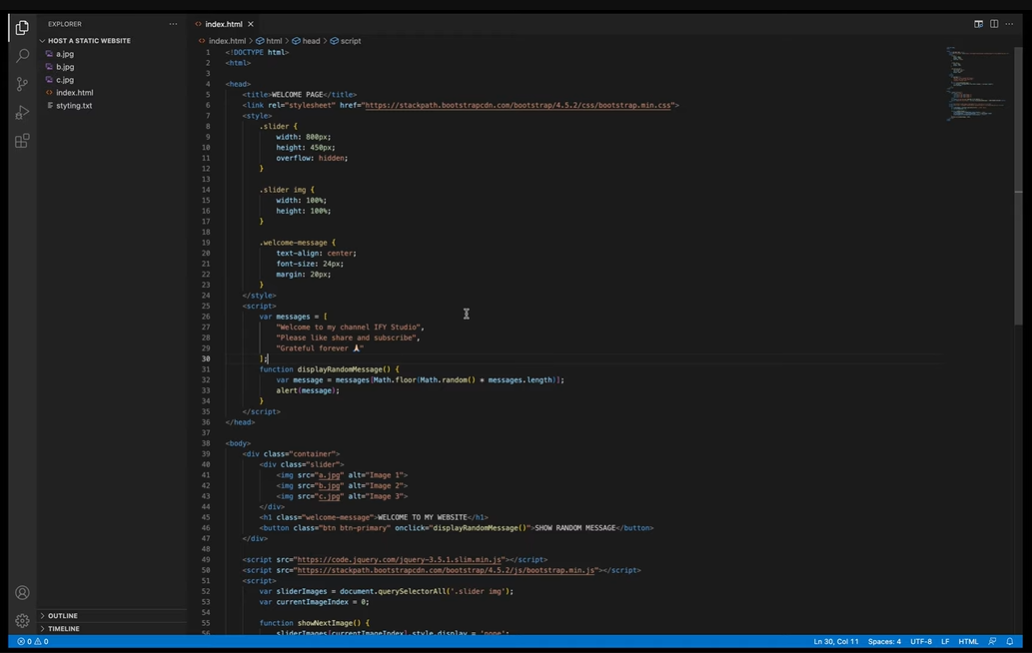
"Principal": "\*",

"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::example.com/\*"

}

Upload Files:



]

}

* After uploading, your static website can be accessed using the endpoint provided in the static website hosting section (http://example.com.s3-website-us-east-1.amazonaws.com).

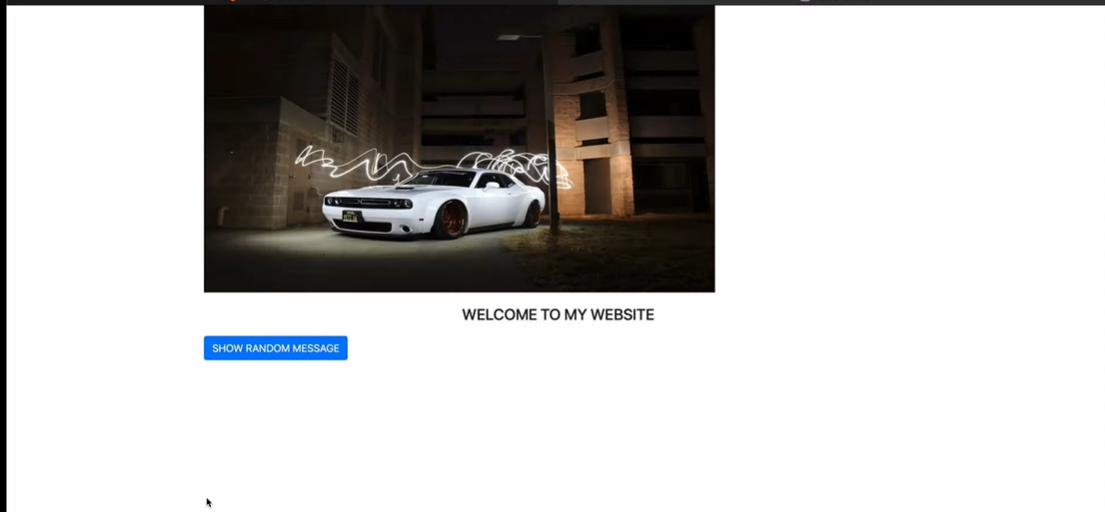
## Set Up an EC2 Instance

If you need server-side processing or other functionalities not supported by S3 alone:

1. Launch an EC2 Instance:
   * Navigate to EC2 in the AWS Management Console.
   * Click on Launch Instance, choose an Amazon Machine Image (AMI), select instance type, configure security groups, and launch
2. Install a Web Server  
   If you need dynamic content or server capabilities:

sudo apt update

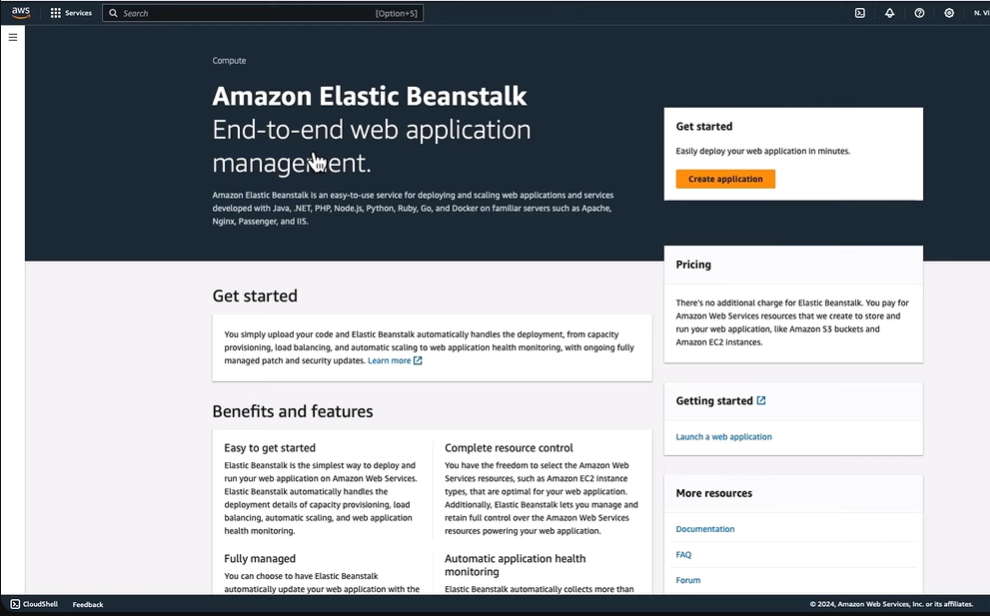
sudo apt install apache2



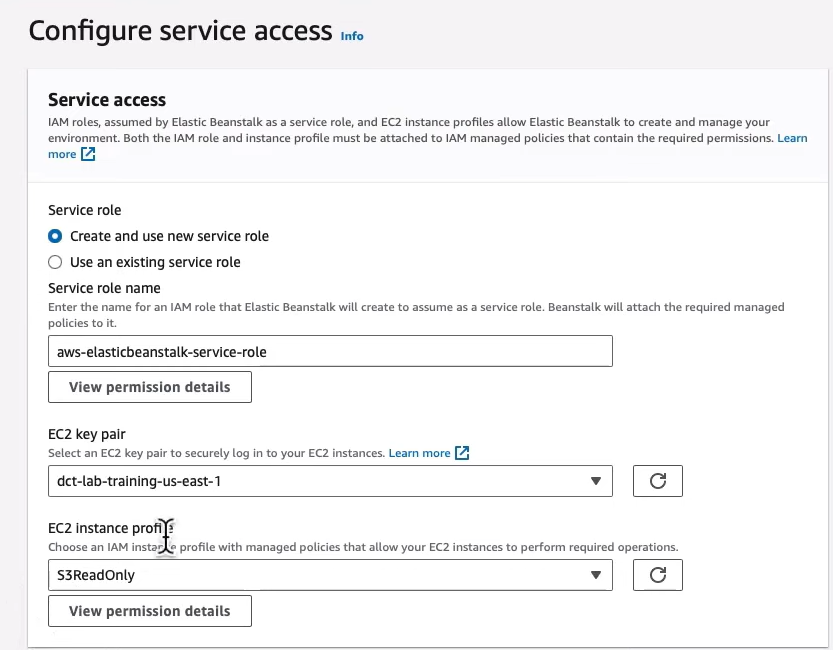
**4. EC2 AWS - Web application using Beanstalk**

## Configure Environment

1. Choose Environment Type:
   * Select Web server environment as the environment tier.
2. Select Platform:
   * Choose the platform for your application (e.g., Node.js, Python, Java, etc.).
3. Configure Service Role:
   * Choose an existing service role or create a new one if necessary.
   * If creating a new role, navigate to the IAM console to set permissions like AWSElasticBeanstalkWebTier, AWSElasticBeanstalkWorkerTier, and others as needed [1](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/GettingStarted.CreateApp.html)[2](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.environments.html).

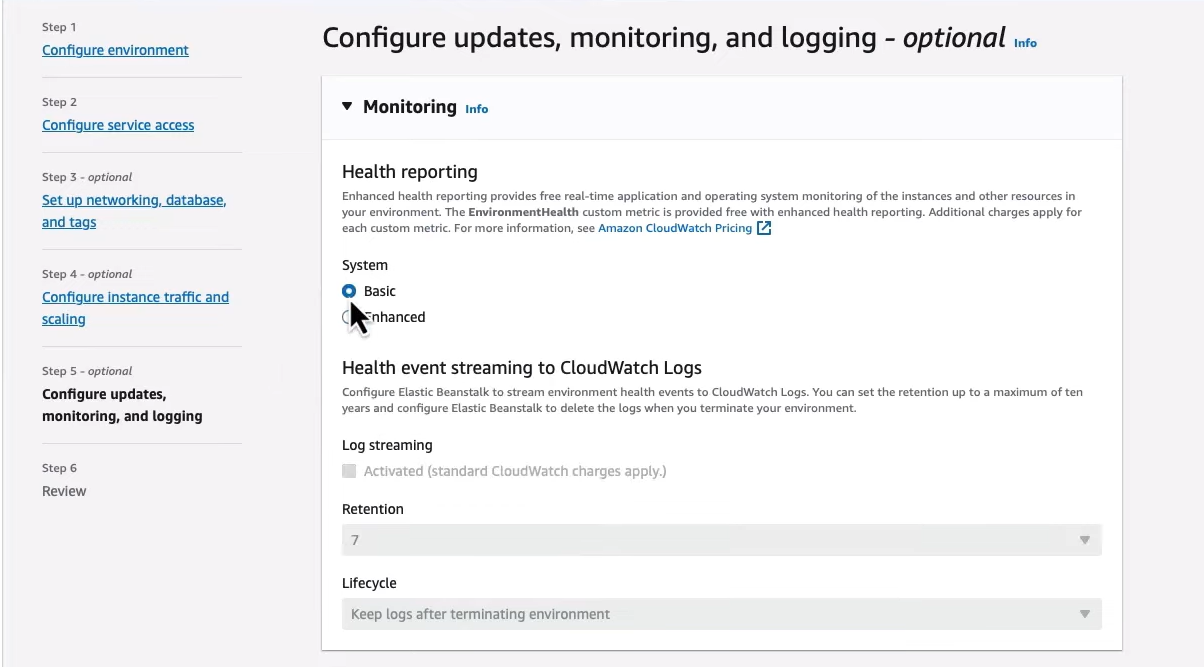


## Cnfigure Application Settings



Create Environment:

* Click on Create Environment.
* AWS will begin provisioning resources, including EC2 instances, security groups, and S3 buckets for storage



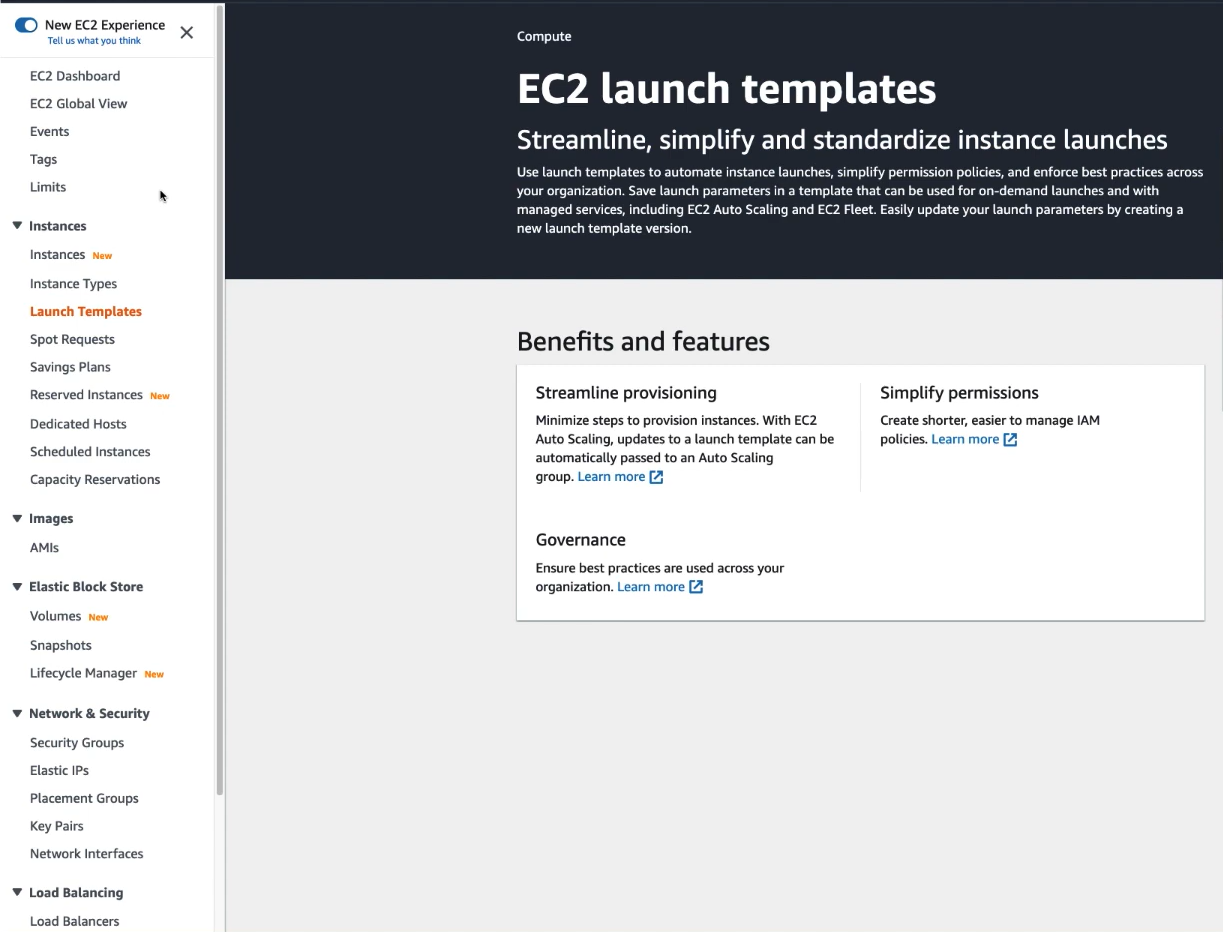
## Deploy Your Application



**5. AWS Local balancing and auto scaling**

## Create a Launch Template

* Click on Launch Templates in the left-hand menu.
* Click on Create launch template.
* Fill in the required details:
  + Name: Provide a name for your template (e.g., my-launch-template).
  + Amazon Machine Image (AMI): Select an appropriate AMI (e.g., Amazon Linux 2).
  + Instance Type: Choose an instance type (e.g., t2.micro for free tier).
  + Key Pair: Select or create a key pair for SSH access.
  + Security Group: Define security group settings to control inbound/outbound traffic.



## Create an Auto Scaling Group

Navigate to Auto Scaling Groups:

* + In the EC2 Dashboard, click on Auto Scaling Groups in the left-hand menu.

Create Auto Scaling Group:

* + Click on the Create Auto Scaling group button.
  + Enter a name for your Auto Scaling group (e.g., my-auto-scaling-group).
  + Select the launch template you created earlier.

Configure Network Settings:

* + Choose the VPC and subnets where instances will be launched. Select multiple subnets for high availability.
  + Click on Next to proceed.

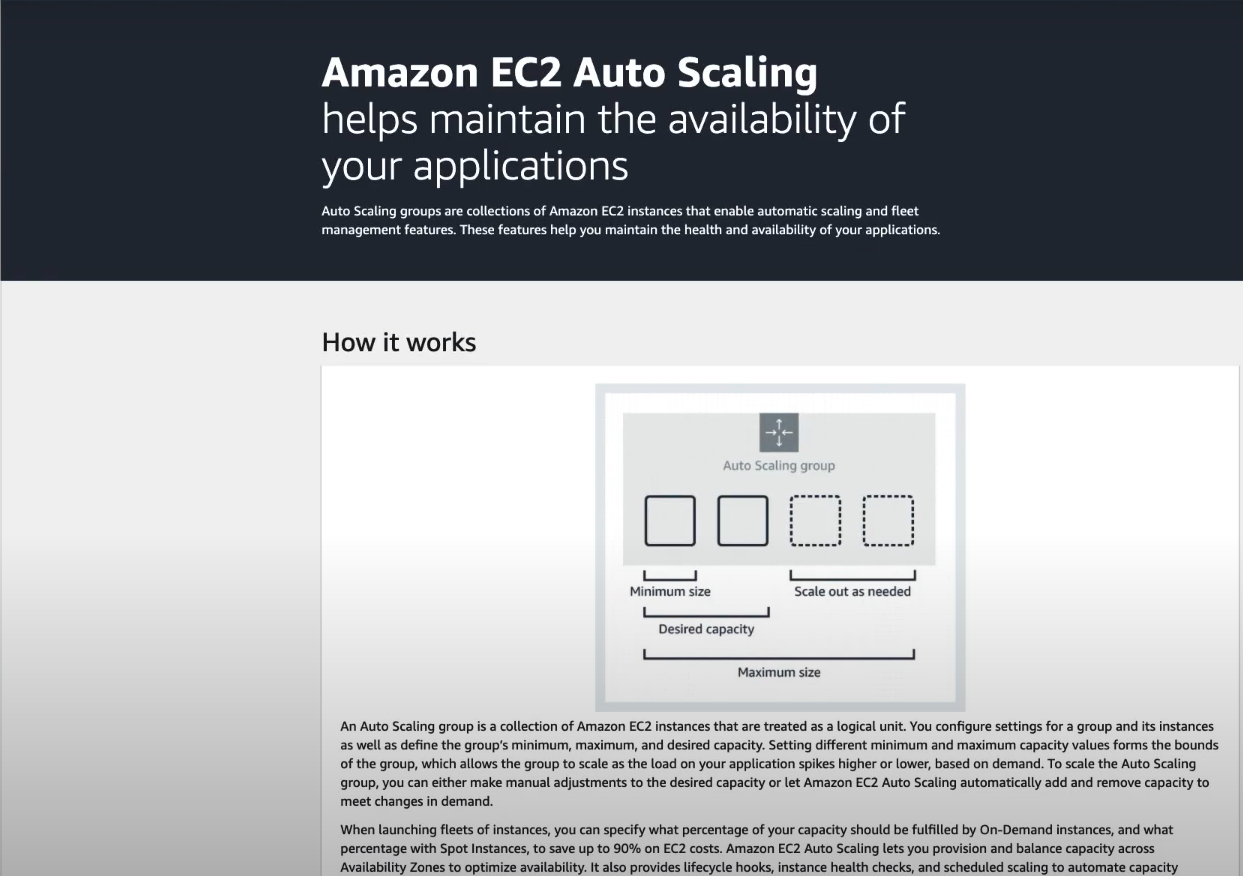
## Set Scaling Policies

Define Capacity Settings:

* + Set minimum, maximum, and desired capacity for your instances.
  + For example:Minimum: 1

Configure Scaling Policies:

* + You can choose between different scaling policies:Target Tracking Scaling
  + Set up notifications if desired to alert you of scaling events.



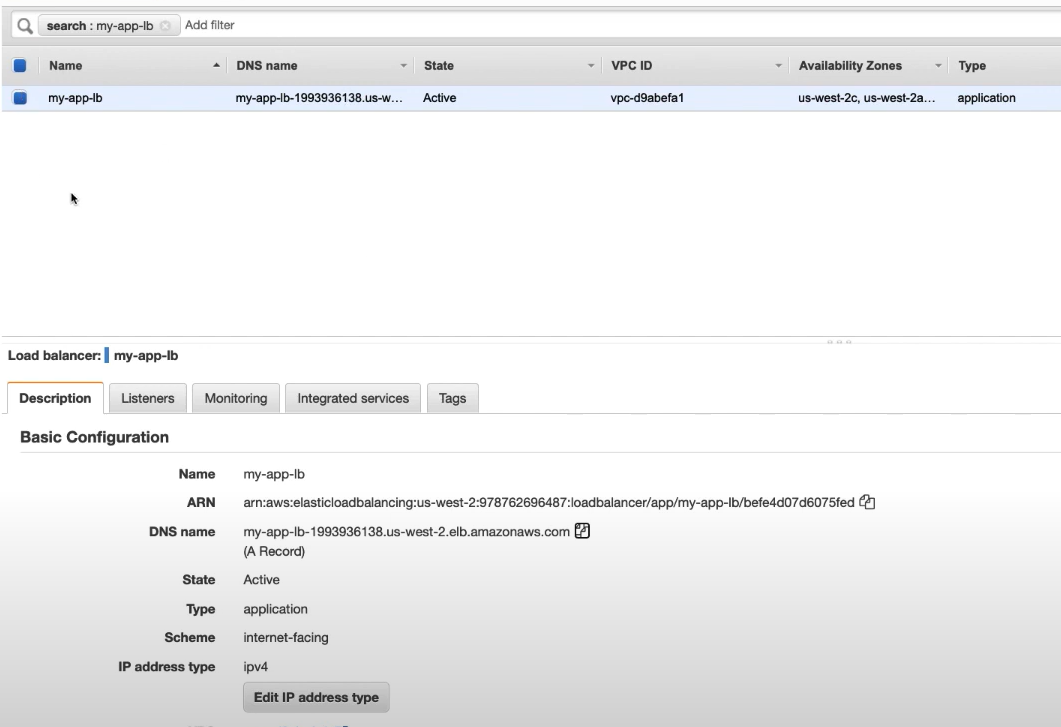
## Review



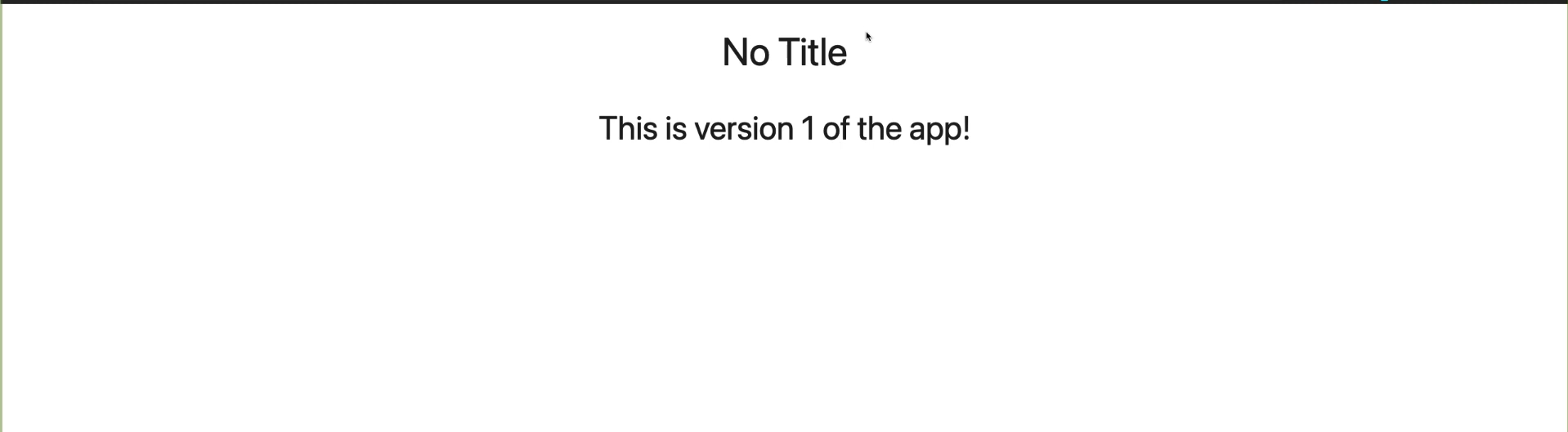
## Configure Load Balancer

## Attach Load Balancer:

* + During the Auto Scaling group setup, you can optionally create or attach an Elastic Load Balancer (ELB).
  + If creating a new load balancer, follow the prompts to configure it, ensuring it distributes traffic across your EC2 instances effectively.



Test Load Balancing (if configured)



1. [↑](#footnote-ref-1)