# Placement Empowerment Program

***Cloud Computing and DevOps Centre***

Host a Static Website on a Cloud VM Install Apache on your cloud VM and host a simple HTML website.

Name: SREEVATSAN V P Department: AIML



# Introduction

# A static website consists of pre-designed HTML, CSS, and JavaScript files delivered directly to users without server-side processing. Deploying such websites on a cloud-based Virtual Machine (VM) is an efficient, scalable, and budget-friendly method, ensuring seamless global reach with minimal setup.

# Overview

# Hosting a static website on a cloud VM involves:

# Setting Up a VM – Creating a virtual machine on platforms like AWS, Azure, or GCP.

# Configuring a Web Server – Setting up Apache or Nginx to handle static content.

# Uploading Site Files – Storing HTML, CSS, and JavaScript in the server's main directory.

# Enabling Network Access – Allowing HTTP traffic (port 80) for public visibility.

# Testing & Deployment – Verifying functionality before making the site live.

# Objectives

# Develop hands-on knowledge of cloud computing and web hosting.

# Successfully publish a static website online.

# Understand networking essentials like security rules and HTTP settings.

# Explore cost-efficient web hosting options.

# Significance

# Practical Cloud Exposure – A stepping stone to understanding cloud infrastructure and VM operations.

# Scalability & Accessibility – Easily adjust resources and ensure worldwide availability.

# Customization & Control – Full authority over hosting configurations and optimizations.

# Career Advancement – Strengthens web deployment skills for real-world applications.

# 

# Step-by-Step Overview

## Step 1:

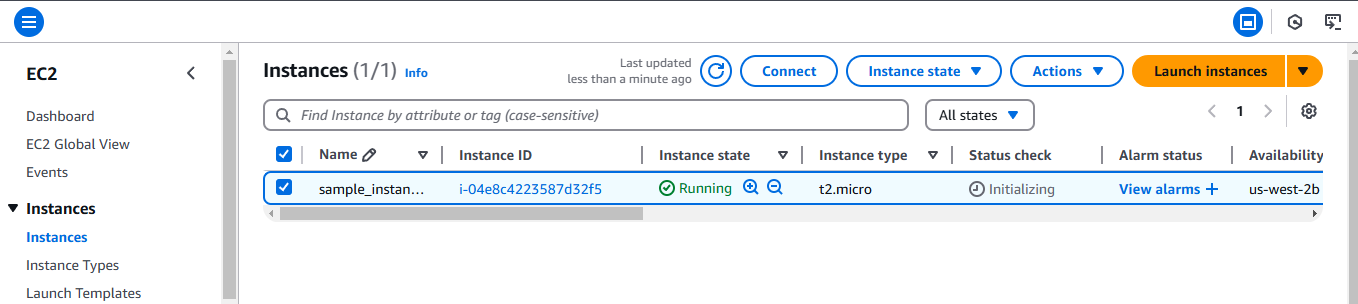
Have an HTML file (with any related assets like CSS/JavaScript) that you want to host in your GitHub repository

A screenshot of a computer

Description automatically generated

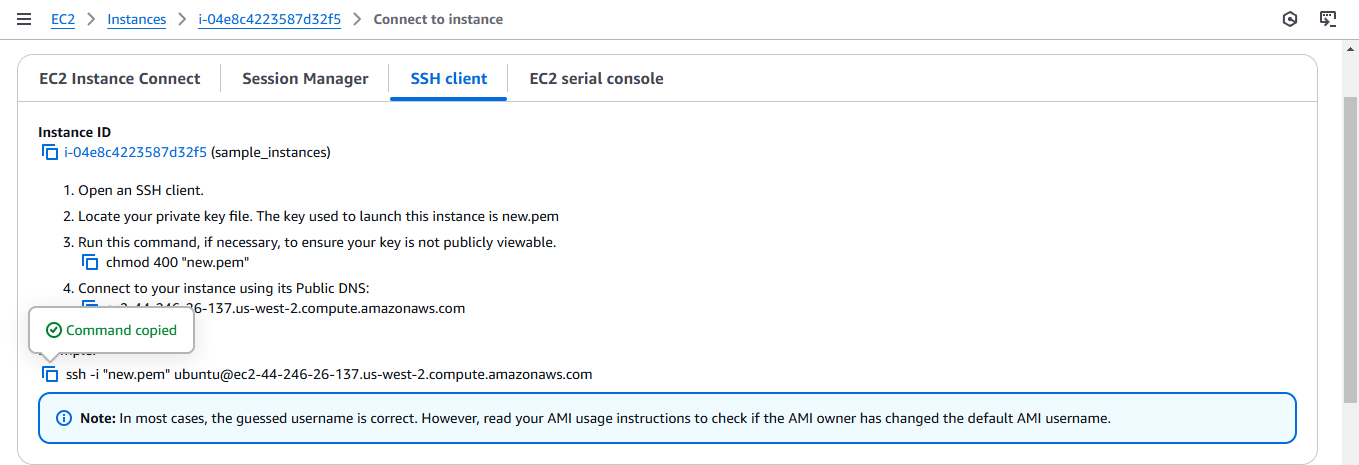
## Step 2:

Launch an EC2 instance, select Ubuntu as the OS, configure security groups to allow all network traffic, create a key pair (e.g., new.pem), and download it for SSH access



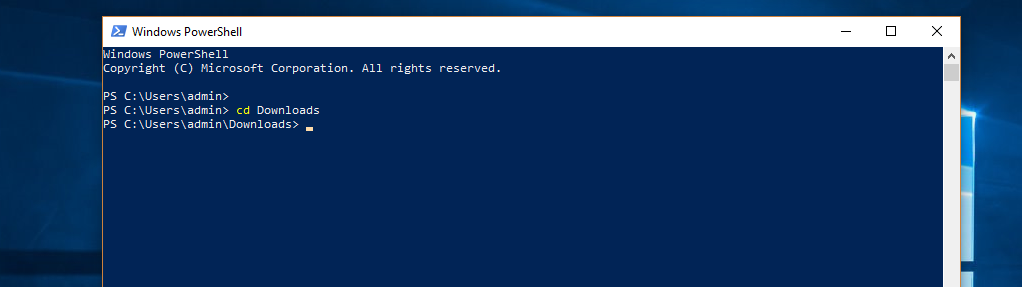
## Step 3:

Click the 'Connect' option on your launched instance, go to the SSH client section, and copy the command provided under the 'Example' section.



## Step 4:

Open PowerShell, navigate to the 'Downloads' directory where the downloaded key pair is located using the **cd Downloads** command



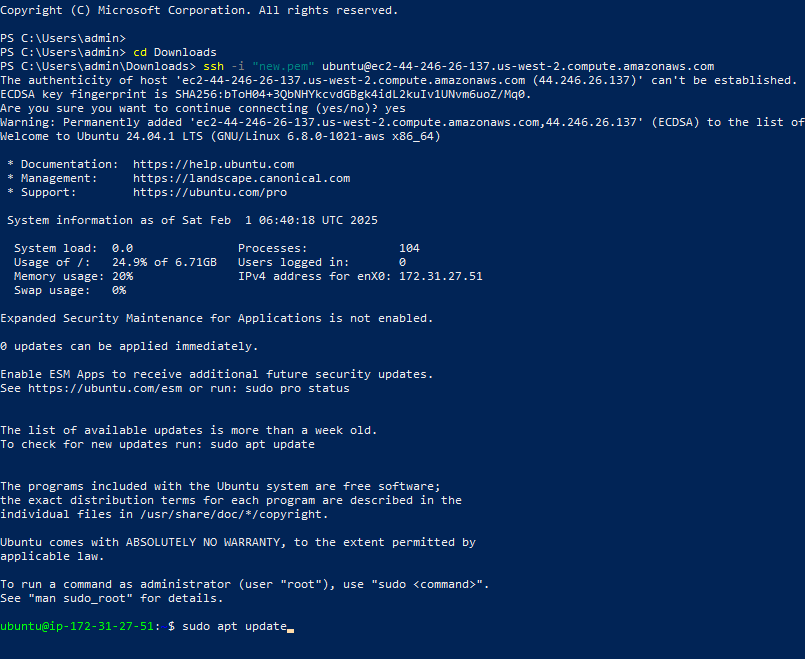
## Step 5:

Paste the command copied from the EC2 Connect's SSH client section, replace the key pair name with your downloaded key (e.g., new.pem), press Enter, and type 'yes' when prompted.

## 

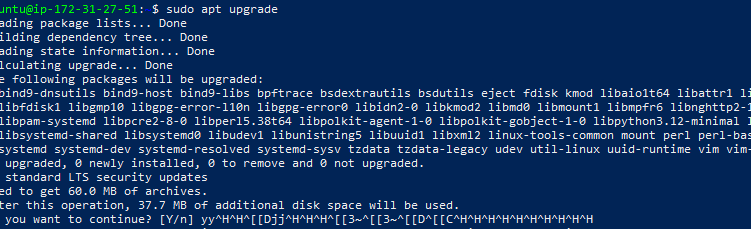
## Step 6:

Run the command **sudo apt update** to update the package list.



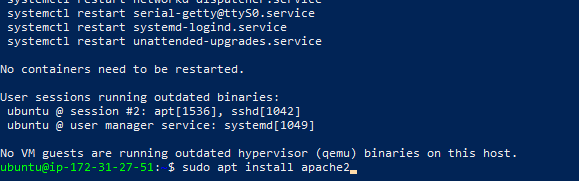
## Step 7:

Run the command **sudo apt upgrade**, and press 'Y' to confirm and continue the upgrade process.



## Step 8:

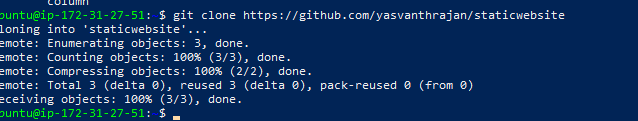
Install the Apache server by running the command **sudo apt install apache2**, and press 'Y' to confirm the installation



## Step 9:

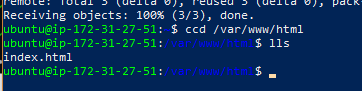
Insert your files by running the command **git clone**

**<repository\_link>** to clone your repository containing the website files



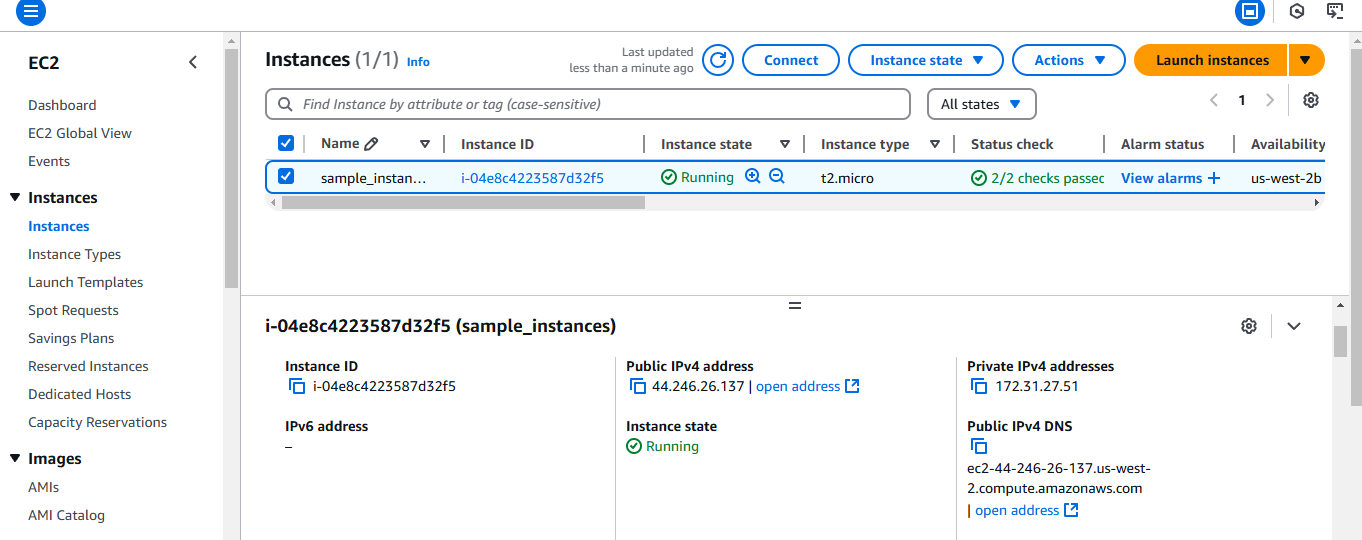
## Step 10:

Run the command **cd /var/www/html** to navigate to the web server's root directory, then type ls to verify that your HTML files from the GitHub repository are present.



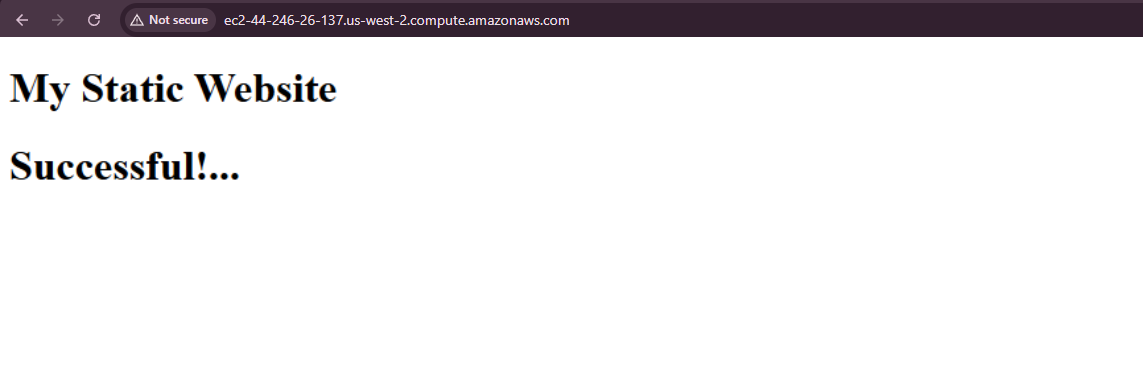
## Step 11:

Copy the Public IPv4 DNS from the instance details page in the EC2 console, as shown in the image below.



## Step 12:

Open Chrome and paste the copied Public IPv4 DNS in the address bar to view the content of your index.html file.



# Outcome

By completing this PoC of deploying a static website using an EC2 instance, you will:

1. Launch and configure an EC2 instance with Ubuntu as the OS.
2. Install and configure Apache web server to serve your static website.
3. Clone your GitHub repository containing your static website files (HTML, CSS, JavaScript) onto your EC2 instance.
4. Upload and place the website files in the Apache root directory (/var/www/html).
5. Access your static website live on the web using the EC2 instance's Public IPv4 DNS.