

Deploying a Web Application on AWS EC2 Using Apache Web Server

Project Objective

The goal of this project is to gain **hands-on experience with Amazon Web Services (AWS)** by:

- Launching an EC2 instance
- Configuring SSH and HTTP access using Security Groups
- Installing and managing the Apache2 web server
- Deploying a sample web application accessible through a browser

By completing this project, students will understand:

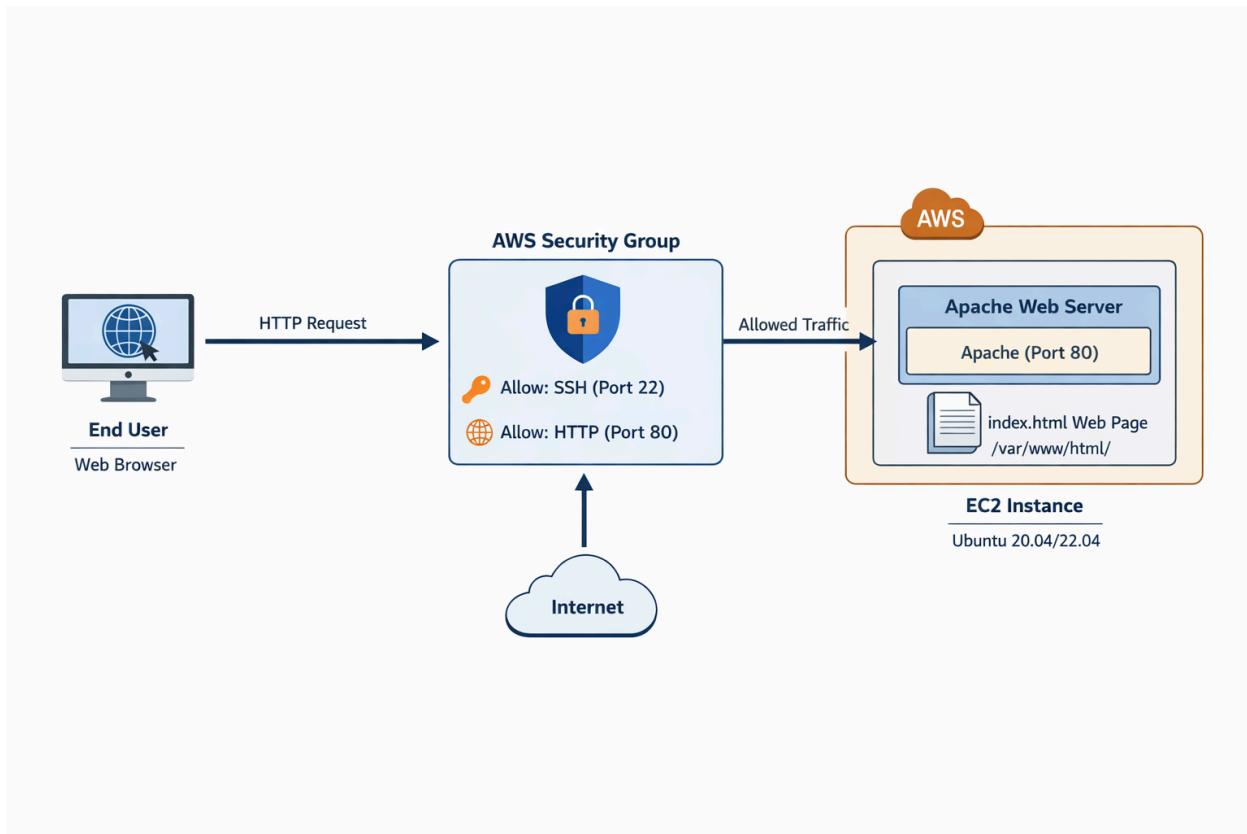
- Basic cloud infrastructure setup
 - Linux server management
 - Web application deployment on AWS
-

Prerequisites

- Basic knowledge of Linux commands
 - An active AWS account
 - Basic understanding of networking concepts (ports, security groups)
 - SSH client:
 - Terminal (Linux/macOS)
 - PuTTY (Windows)
-

Project Architecture Overview

User Browser → EC2 Public IP → Apache2 Web Server → HTML Web Application



Project Tasks & Step-by-Step Implementation

Task 1: Create an EC2 Instance

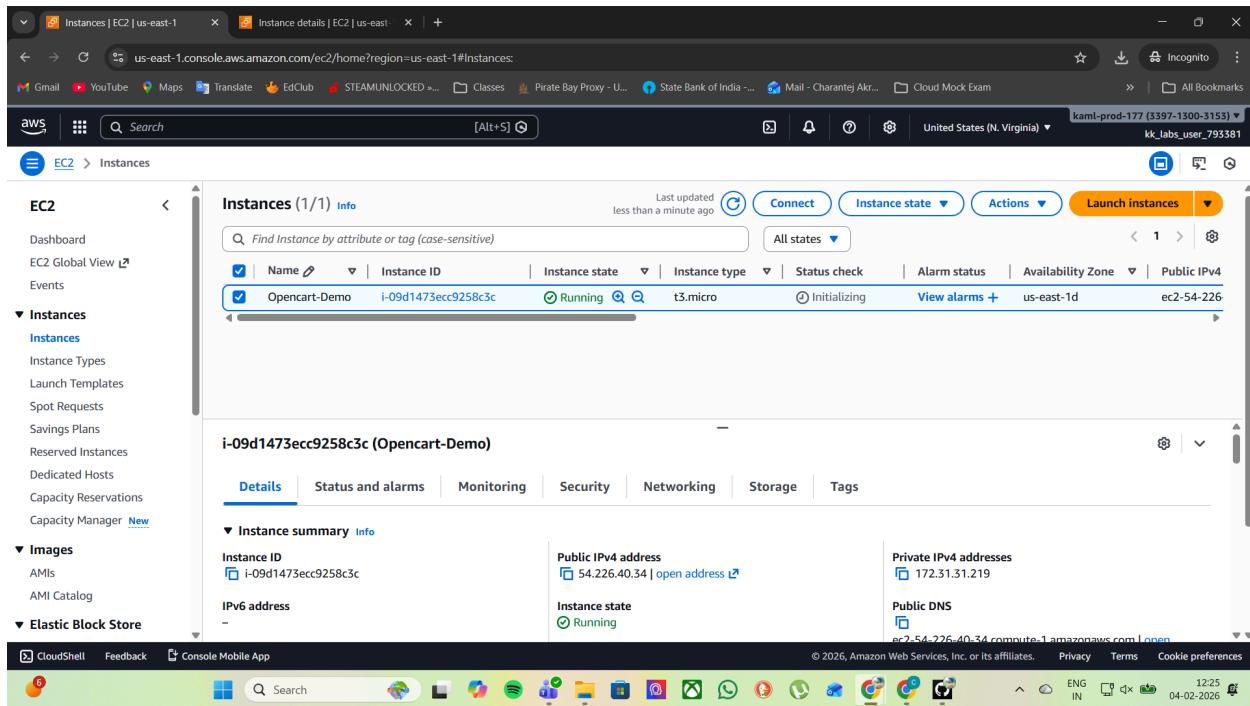
Steps

1. Log in to the **AWS Management Console**
2. Navigate to **EC2 → Launch Instance**
3. Choose an Amazon Machine Image (AMI):
 - Ubuntu Server **20.04 LTS or 22.04 LTS**
4. Select instance type:
 - **t2.micro** (Free Tier eligible)

5. Create or select an existing **key pair** for SSH access
6. Launch the EC2 instance

Deliverable

- EC2 instance in **Running** state



Task 2: Configure Security Groups (Firewall Rules)

Steps

1. Open the EC2 instance details
2. Navigate to the **Security** tab
3. Edit inbound rules for the security group

Inbound Rules Configuration

Protocol	Port	Source
----------	------	--------

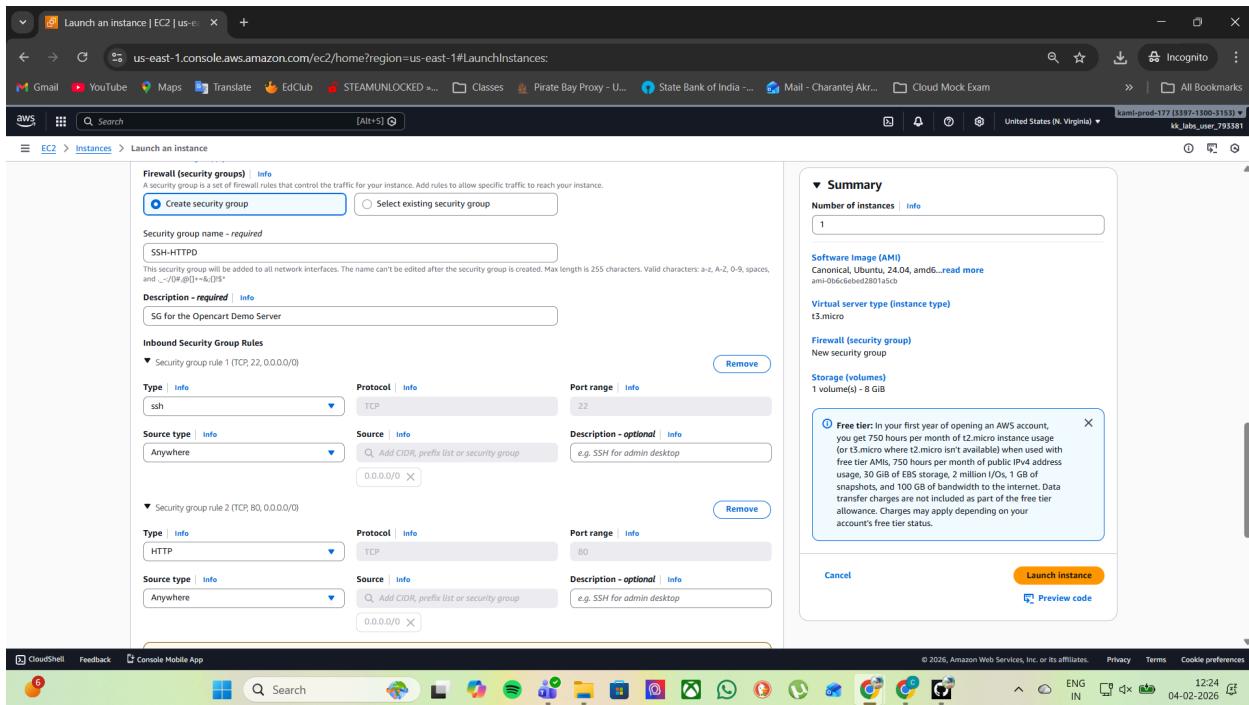
SSH	22	Your IP
-----	----	---------

HTTP 80 Anywhere
(0.0.0.0/0)

📌 SSH access is restricted to your IP for security best practices.

Deliverable

- Properly configured inbound rules



Task 3: Connect to EC2 Using SSH

Steps

- Copy the **Public IPv4 Address** of the EC2 instance
- Open terminal and connect using SSH:

```
ssh -i your-key.pem ubuntu@<public-ip>
```

3. Verify successful login to the EC2 instance

Deliverable

- Successful SSH connection

Connect Info

Connect to an instance using the browser-based client.

EC2 Instance Connect Session Manager **SSH client** EC2 serial console

Instance ID [i-09d1473ecc9258c3c](#) (Opencart-Demo)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is dummy.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "dummy.pem"
4. Connect to your instance using its Public DNS:
<ssh -i "dummy.pem" ubuntu@ec2-54-226-40-34.compute-1.amazonaws.com>

Example:
<ssh -i "dummy.pem" ubuntu@ec2-54-226-40-34.compute-1.amazonaws.com>

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

```
PS C:\Users\MSI 1> cd .\Downloads\  
PS C:\Users\MSI 1\Downloads> ssh -i "dummy.pem" ubuntu@ec2-54-226-40-34.compute-1.amazonaws.com  
The authenticity of host 'ec2-54-226-40-34.compute-1.amazonaws.com (54.226.40.34)' can't be established.  
ED25519 key fingerprint is SHA256:D1Z79jaKXCigMNqlz9U0g/W761GKLfcgczbwUiac.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added 'ec2-54-226-40-34.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
```

Connect Info

Connect to an instance

EC2 Instance Connect Session Manager **SSH client**

Instance ID [i-09d1473ecc9258c3c](#)

1. Open an SSH client.
2. Locate your private key file.
3. Run this command.
chmod 400 "dummy.pem"
4. Connect to your instance using its Public DNS:
<ssh -i "dummy.pem" ubuntu@ec2-54-226-40-34.compute-1.amazonaws.com>

Command copy

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Task 4: Install and Start Apache2 Web Server

Steps

Update the package list:

```
sudo apt update
```

```
ubuntu@ip-172-31-31-219:~$ sudo apt -y update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
```

Install Apache2:

```
sudo apt install apache2 -y
```

```
ubuntu@ip-172-31-31-219:~$ sudo apt install apache2 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```

Start and enable Apache service:

```
sudo systemctl start apache2
```

```
sudo systemctl enable apache2
```

```
ubuntu@ip-172-31-31-219:~$ sudo systemctl start apache2
ubuntu@ip-172-31-31-219:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@ip-172-31-31-219:~$ |
```

Verify Apache status:

```
sudo systemctl status apache2
```

Deliverable

- Apache service running successfully

```
ubuntu@ip-172-31-31-219:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
  Active: active (running) since Wed 2026-02-04 07:01:04 UTC; 23s ago
    Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2266 (Apache2)
      Tasks: 55 (limit: 1008)
     Memory: 5.7M (peak: 6.4M)
        CPU: 38ms
       CGroup: /system.slice/apache2.service
           └─2266 /usr/sbin/apache2 -k start
              ├─2268 /usr/sbin/apache2 -k start
              ├─2269 /usr/sbin/apache2 -k start
              └─2269 /usr/sbin/apache2 -k start

Feb 04 07:01:04 ip-172-31-31-219 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Feb 04 07:01:04 ip-172-31-31-219 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Task 5: Test HTTP Access

Steps

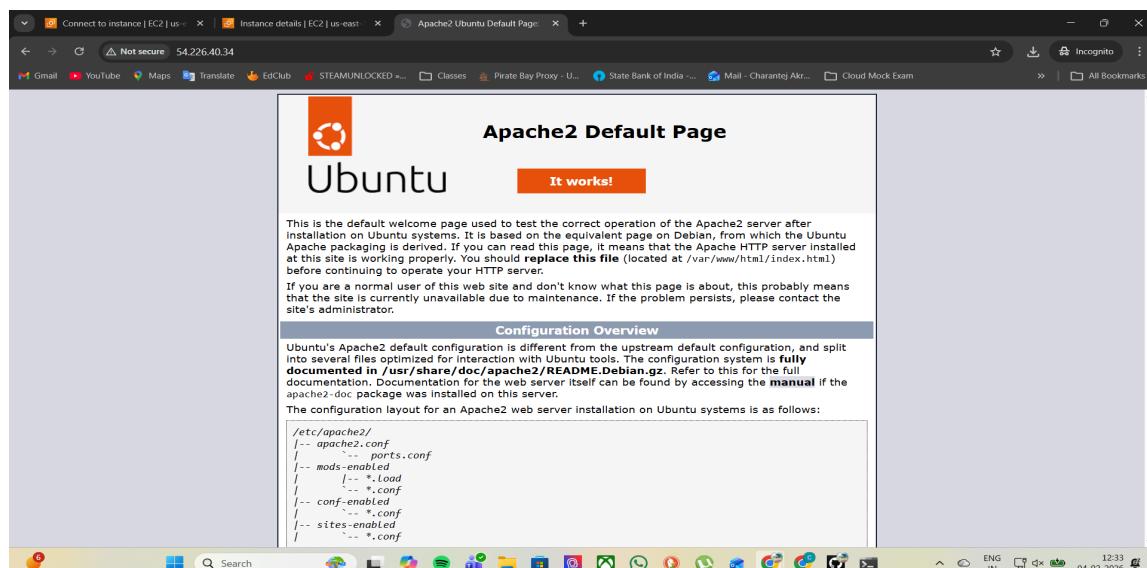
1. Open a web browser
2. Enter the EC2 Public IP Address:

<http://<public-ip>>

3. Confirm the **Apache2 default page** is displayed

Deliverable

- Apache default web page visible in browser



Task 6: Deploy a Sample Web Application

Steps

Navigate to Apache web root:

```
cd /var/www/html
```

```
ubuntu@ip-172-31-31-219:~$ cd /var/www/html
ubuntu@ip-172-31-31-219:/var/www/html$ ls
index.html
ubuntu@ip-172-31-31-219:/var/www/html$ |
```

Delete the default HTML file:

```
sudo rm index.html
```

```
ubuntu@ip-172-31-31-219:/var/www/html$ sudo rm index.html
ubuntu@ip-172-31-31-219:/var/www/html$ ls
ubuntu@ip-172-31-31-219:/var/www/html$ |
```

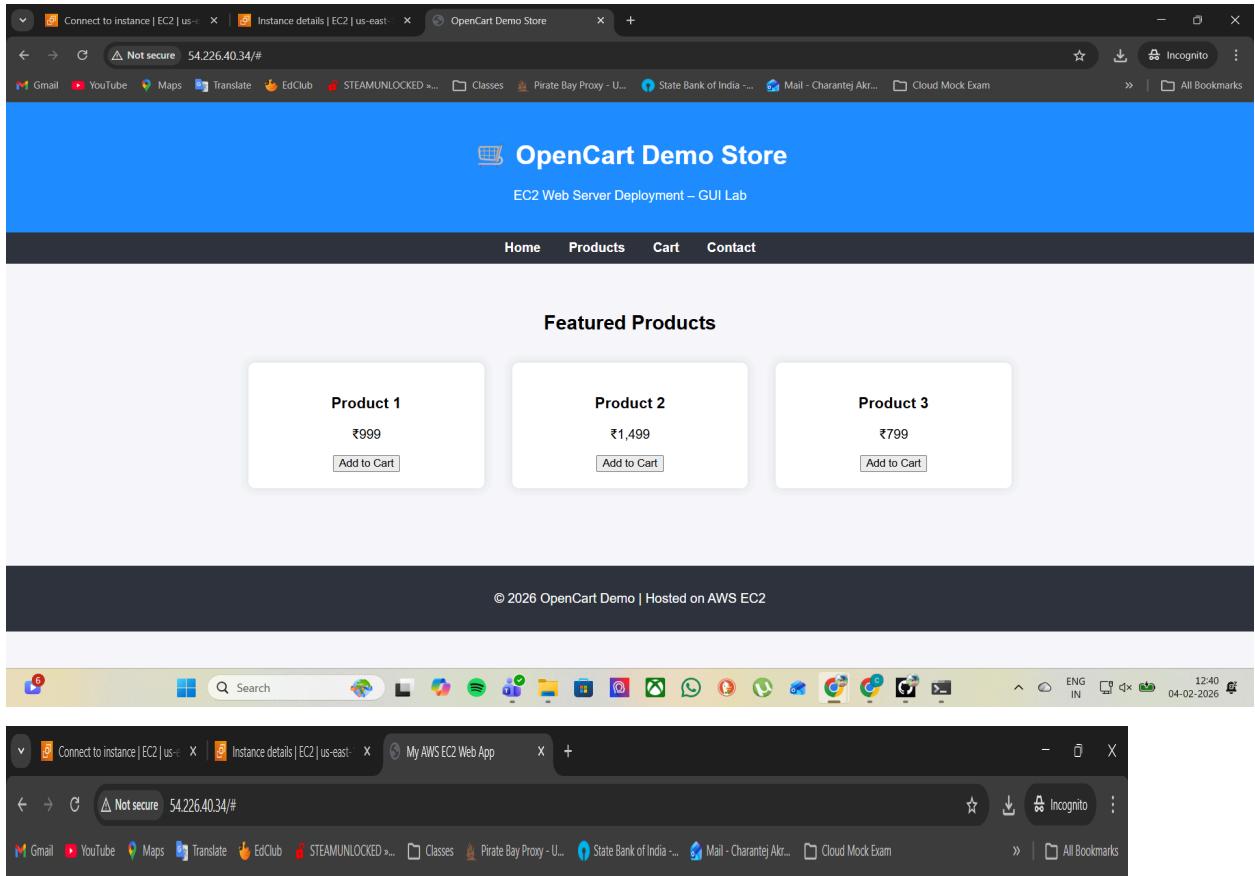
Add the following content or any content you want. I am using code from my repo.

```
<!DOCTYPE html>
<html>
<head>
    <title>My AWS EC2 Web App</title>
</head>
<body>
    <h1>Welcome to My EC2 Web Server</h1>
    <p>Apache2 is running successfully on AWS!</p>
</body>
</html>
```

Save and exit the file, then refresh the browser.

Deliverable

- Custom web application page displayed



Notes

- This project uses **AWS Management Console (GUI)** for infrastructure setup
- Suitable for:
 - Beginners in AWS
 - Students learning cloud fundamentals
 - Entry-level DevOps and Cloud roles