Deploying a Web Application on AWS EC2 Using Apache Web Server

---

Project Objective

The objective of this project is to gain hands-on experience with Amazon Web Services (AWS) by launching an EC2 instance, configuring secure access, installing and managing the Apache2 web server, and deploying a simple web application that can be accessed through a web browser.

By completing this project, students will develop foundational skills in: - Cloud infrastructure setup using AWS EC2 - Basic Linux server administration - Networking concepts such as ports and security groups - Web server installation and deployment

---

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)

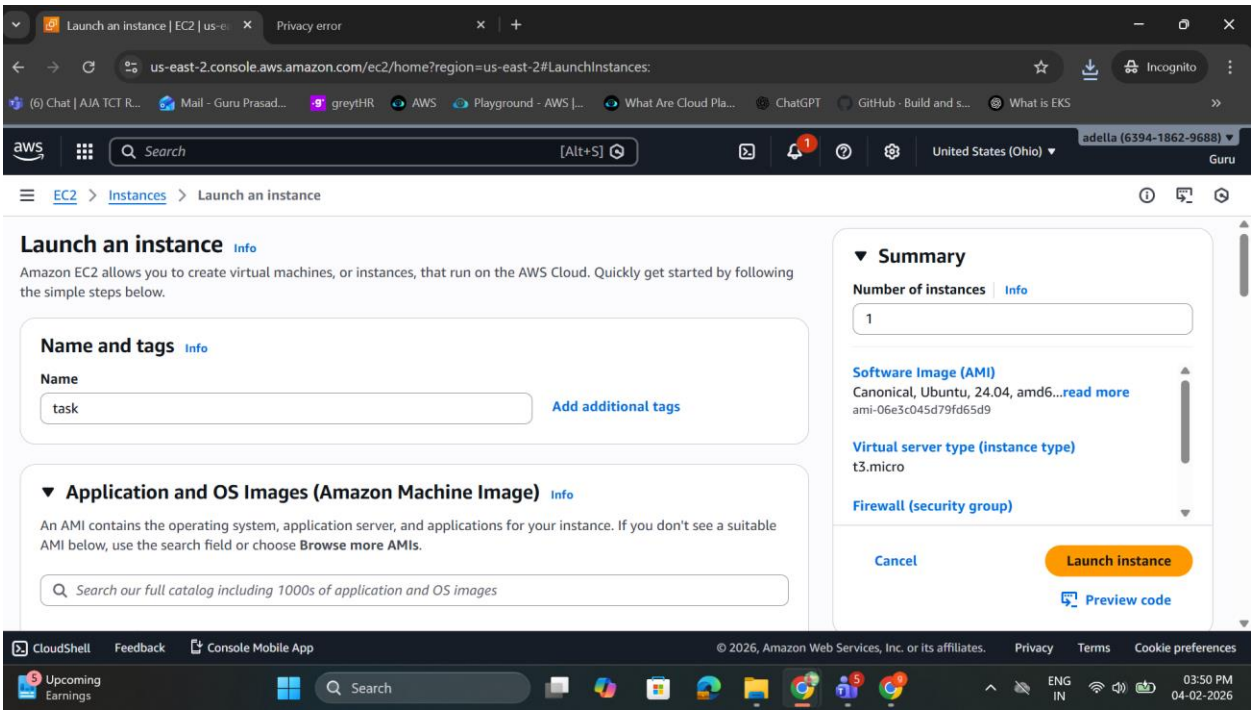
Task 1: Create an EC2 Instance

Steps Performed

1. Logged in to the AWS Management Console.
2. Navigated to **EC2 → Launch Instance**.
3. Selected an Amazon Machine Image (AMI):
    - Ubuntu Server 20.04 LTS or 22.04 LTS
4. Chose the instance type:
    - **t2.micro** (Free Tier eligible)
5. Created or selected an existing key pair for SSH access.
6. Launched the instance and verified that its state changed to **Running**.

Deliverable

- Screenshot showing the EC2 instance in **Running** state.



Task 2: Configure Security Groups (Firewall Rules)

Steps Performed

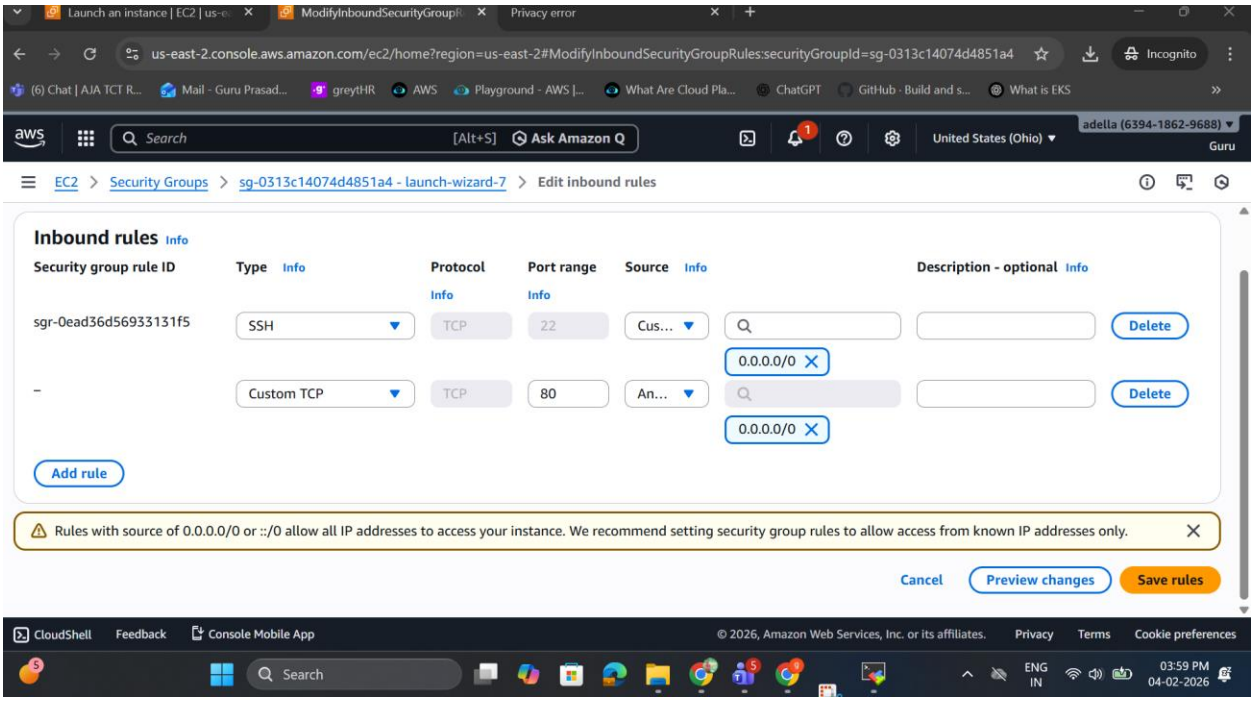
1. Opened the EC2 instance details page.
2. Edited the inbound rules of the associated security group.
3. Added the following inbound rules:

Protocol	Port	Source
SSH	22	My IP
HTTP	80	0.0.0.0/0

4. Saved the security group configuration.

Deliverable

- Screenshot of the security group inbound rules configuration.



Task 3: Connect to EC2 Using SSH

Steps Performed

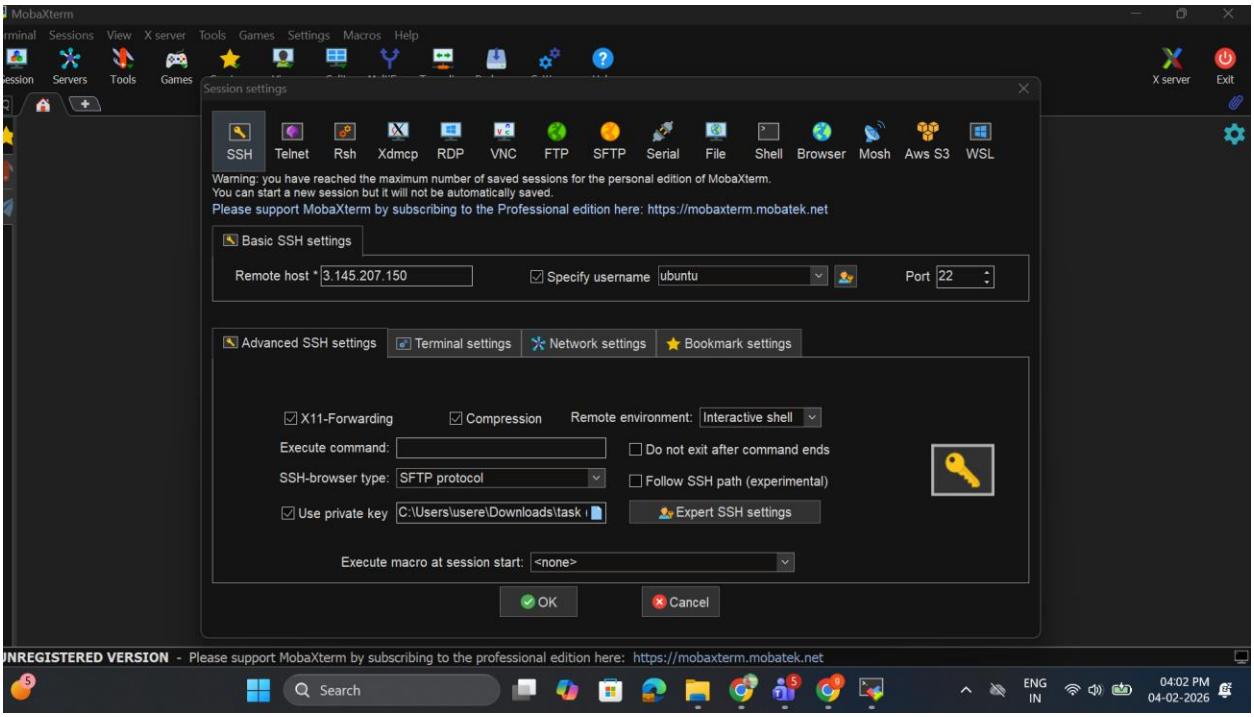
1. Copied the public IPv4 address of the EC2 instance.
2. Opened a terminal (or PuTTY on Windows).
3. Connected to the EC2 instance using the SSH command:

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

4. Verified successful login to the Ubuntu server.

Deliverable

- Screenshot showing successful SSH login to the EC2 instance.



Task 4: Install and Start Apache2 Web Server

Steps Performed

1. Updated the package list:

```
sudo apt update
```

2. Installed Apache2 web server:

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

3. Started and enabled Apache to run on boot:

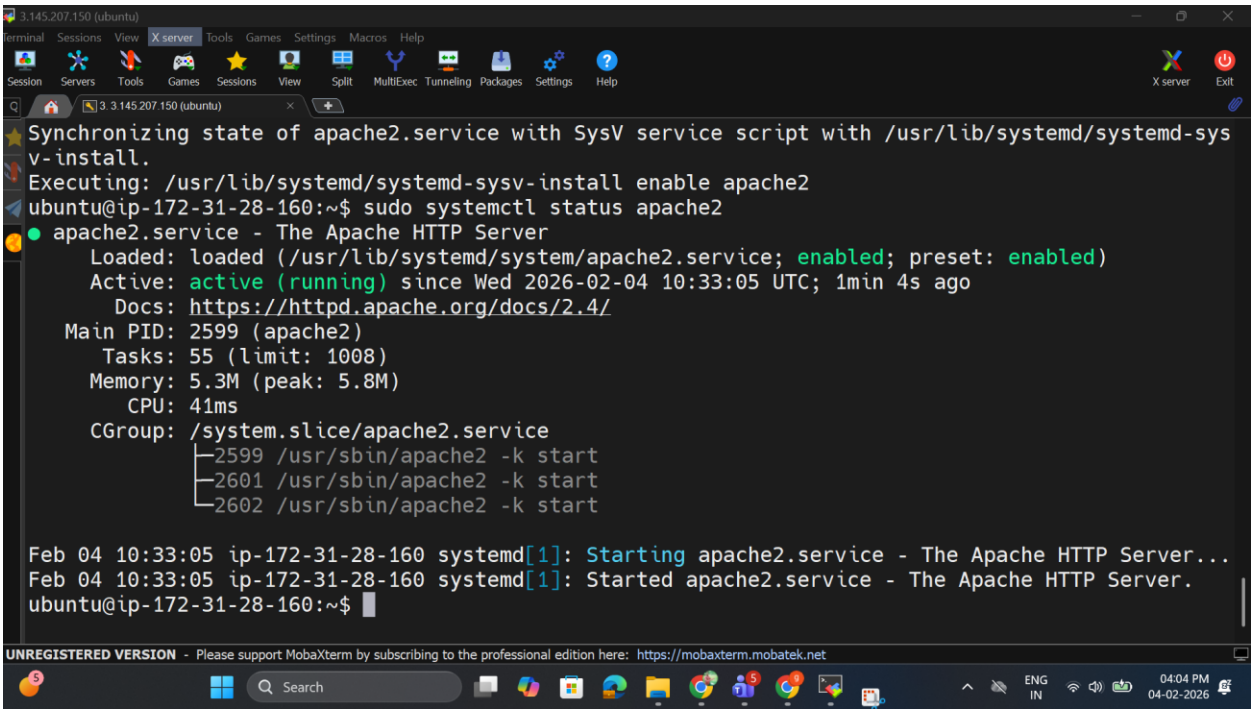
```
sudo systemctl start apache2
sudo systemctl enable apache2
```

4. Verified Apache service status:

```
sudo systemctl status apache2
```

Deliverable

- Screenshot showing Apache2 service running successfully.



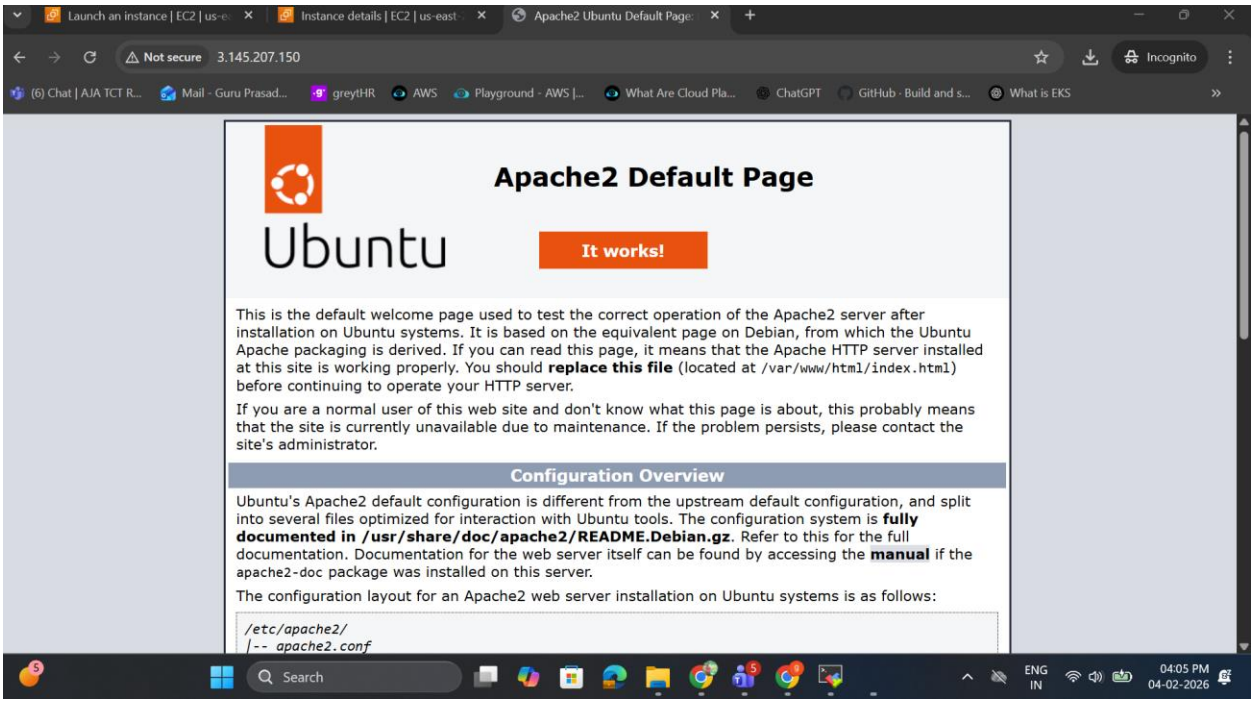
Task 5: Test HTTP Access

Steps Performed

1. Opened a web browser.
2. Entered the EC2 public IPv4 address in the address bar.
3. Confirmed that the Apache2 default welcome page was displayed, indicating successful HTTP access.

Deliverable

- Screenshot of the Apache2 default web page displayed in the browser.



Task 6: Deploy a Sample Web Application

Steps Performed

1. Navigated to the Apache web root directory:

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

2. Created and edited the `index.html` file:

```
sudo nano index.html
```

3. Added the following HTML content:

```
<!DOCTYPE html>
<html>
<head>
<title>Shopping Page</title>
<style>
 body{
    font-family: Arial;
    background: #fef6e4;
 }
 header{
    background: #8bd3dd;
    padding: 15px;
    text-align: center;
    font-size: 24px;
```

```
}
.nav{
    display: flex;
    justify-content: center;
    gap: 20px;
    margin: 20px;
}
.nav a{
    text-decoration: none;
    color: white;
    background: #f582ae;
    padding: 10px 20px;
    border-radius: 20px;
}
.products{
    display: grid;
    grid-template-columns: repeat(4,1fr);
    gap: 20px;
    padding: 20px;
}
.card{
    background: white;
    padding: 15px;
    border-radius: 15px;
    text-align: center;
}
button{
    background: #8bd3dd;
    border: none;
    padding: 8px;
    cursor: pointer;
}
</style>
</head>


<body>
<header> 🌈 Colorful Shopping Store</header>


<div class="nav">
<a href="#men">Men</a>
<a href="#women">Women</a>
<a href="#jewellery">Jewellery</a>
<a href="#kids">Kids</a>
```

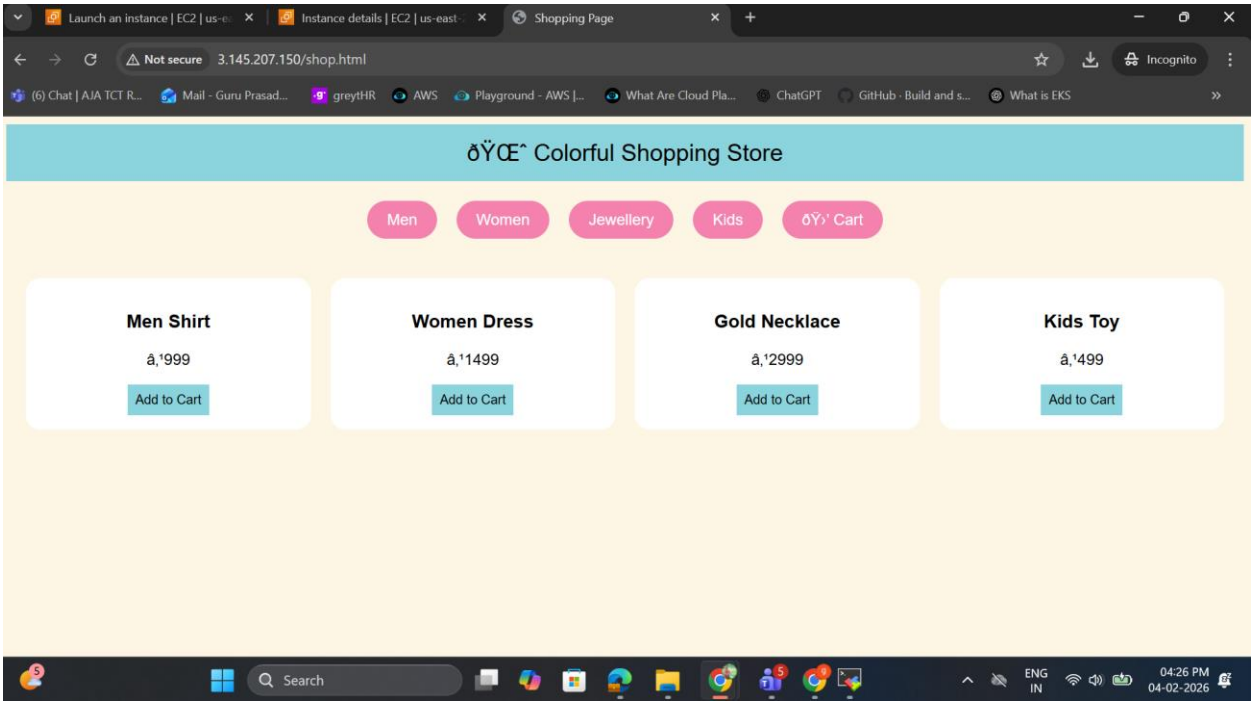
```
<a href="cart.html">🛒 Cart</a>
</div>


<div class="products">
<div class="card">
<h3>Men Shirt</h3>
<p>₹999</p>
<button onclick="addToCart('Men Shirt')">Add to Cart</button>
</div>


   <div class="card">
<h3>Women Dress</h3>
<p>₹1499</p>
<button onclick="addToCart('Women Dress')">Add to Cart</button>
</div>


   <div class="card">
<h3>Gold Necklace</h3>
<p>₹2999</p>
<button onclick="addToCart('Gold Necklace')">Add to Cart</button>
</div>


   <div class="card">
<h3>Kids Toy</h3>
<p>₹499</p>
<button onclick="addToCart('Kids Toy')">Add to Cart</button>
</div>
</div>


<script>
 function addToCart(item){
    let cart = JSON.parse(localStorage.getItem("cart")) || [];
    cart.push(item);
    localStorage.setItem("cart", JSON.stringify(cart));
    alert(item + " added to cart!");
}
</script>
```

```
</body>
</html>
```

4. Saved and exited the file.
5. Refreshed the browser to verify that the custom web application was displayed.

Deliverable

- Screenshot of the custom web application page running on the EC2 instance.



Conclusion

This project successfully demonstrated how to deploy a basic web application on AWS using an EC2 instance and the Apache2 web server. Through this exercise, core concepts of cloud computing, Linux server management, security configuration, and web hosting were applied in a practical environment.

The deployed application confirms that the EC2 instance, security groups, Apache service, and web content are all functioning correctly.