

Lab Report: Apache Web Server Installation & Maintenance

Name: Amit Kumar Sharma

Registration: 2020831009

Platform: Ubuntu on WSL (Windows Subsystem for Linux)

Objectives

1. Install, administer, and maintain an Apache web server.
 2. Configure virtual hosts to host multiple websites on a single server.
 3. Host dynamic websites using HTML and JavaScript.
-

Environment & Setup

- **OS:** Ubuntu 22.04 on Windows 10 via WSL
 - **Web Server:** Apache2
 - **Browser:** Firefox / Chrome (for testing)
 - **Terminal Tools:** `curl` to demonstrate web server responses
-

Task 1: Installing Apache Web Server

Step 1: Installing Apache

```
sudo apt update  
sudo apt install apache2
```

Check



Apache status:

```
sudo systemctl status apache2
```

Observation: Apache service started successfully.

Step 2: Adjusting the Firewall

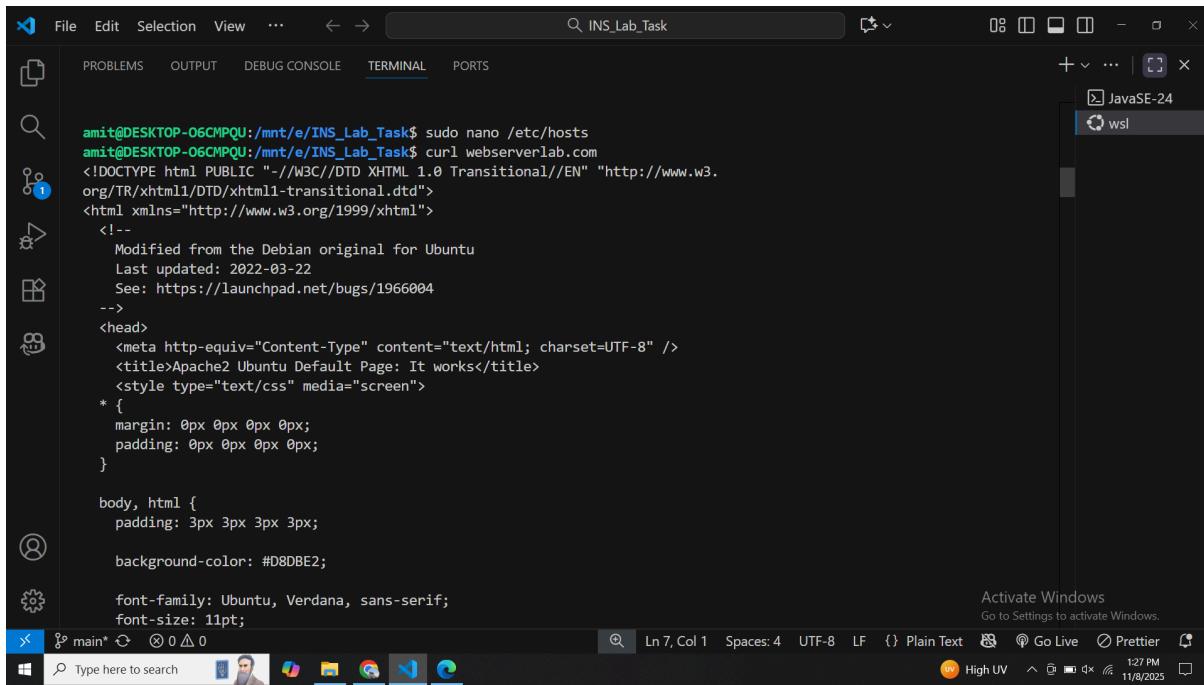
```
sudo ufw allow 'Apache'  
sudo ufw status
```

Observation: HTTP traffic allowed.

Step 3: Checking the Web Server

Test via terminal:

```
curl http://webserverlab.com
```



The screenshot shows a terminal window in VS Code with the title bar "INS_Lab_Task". The "TERMINAL" tab is selected. The terminal content displays the Apache default configuration file (`/etc/apache2/apache2.conf`). The file includes standard Apache directives like `DocumentRoot`, `ServerName`, and `ServerAdmin`. It also contains a `VirtualHost` block for port 80, which maps to the IP address `127.0.0.1` and the document root `/var/www/html`. The configuration is annotated with comments explaining the purpose of each section.

Observation: Default Apache page displayed.

 **Checkpoint 1:** Apache installation verified.

Task 2: Setting Up Virtual Hosts

Step 1: Managing the Apache Process

```
sudo systemctl stop apache2
sudo systemctl start apache2
sudo systemctl restart apache2
sudo systemctl reload apache2
sudo systemctl enable apache2
sudo systemctl disable apache2
```

Observation: Commands to control Apache service demonstrated.

Step 2: Single Virtual Host – `example.com`

Create directory and assign permissions:

```
sudo mkdir -p /var/www/example.com/html
```

```
sudo chown -R $USER:$USER /var/www/example.com/html  
sudo chmod -R 755 /var/www/example.com
```

Create `index.html`:

```
<html>  
<head><title>Welcome to Example.com!</title></head>  
<body><h1>Success! The example.com server block is  
working!</h1></body>  
</html>
```

Create virtual host configuration:

```
sudo nano /etc/apache2/sites-available/example.com.conf
```

Add:

```
<VirtualHost *:80>  
    ServerAdmin admin@example.com  
    ServerName example.com  
    ServerAlias www.example.com  
    DocumentRoot /var/www/example.com/html  
    ErrorLog ${APACHE_LOG_DIR}/error.log  
    CustomLog ${APACHE_LOG_DIR}/access.log combined  
</VirtualHost>
```

Enable site and disable default:

```
sudo a2ensite example.com.conf  
sudo a2dissite 000-default.conf  
sudo apache2ctl configtest  
sudo systemctl restart apache2
```

Test:

```
curl http://example.com
```

The screenshot shows a terminal window in VS Code with the title bar "INS_Lab_Task". The terminal tab is selected, displaying the following command-line session:

```
Nov 08 13:50:47 DESKTOP-06CMPQU systemd[1]: Reloading apache2.service - The Apache HTTP Server...
Nov 08 13:50:47 DESKTOP-06CMPQU systemd[1]: Reloaded apache2.service - The Apache HTTP Server.
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo nano /etc/hosts
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ curl webserverlab.com
<h1>webserverlab.com is working!</h1>

amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo mkdir -p /var/www/example.com/html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo chown -R $USER:$USER /var/www/example.com/html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo chmod -R 755 /var/www/example.com
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ nano /var/www/example.com/html/index.html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo nano /etc/apache2/sites-available/example.com.conf
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo a2ensite example.com.conf

Enabling site example.com.
To activate the new configuration, you need to run:
    systemctl reload apache2
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo systemctl reload apache2
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo nano /etc/hosts
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ curl example.com
<html>
<head>
<title>Welcome to Example.com!</title>
</head>
<body>
<h1>Success! The example.com server block is working!</h1>
</body>
</html>
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$
```

The terminal also shows a message from the operating system: "Activate Windows Go to Settings to activate Windows." The status bar at the bottom right indicates "Humid" and the date/time "11/8/2025 2:03 PM".

✓ **Checkpoint 2:** Virtual host `example.com` is working.

Step 3: Observing Behavior with Multiple Hosts

After enabling `example.com` and restarting Apache:

```
sudo a2ensite example.com.conf
sudo systemctl restart apache2
```

Test:

```
curl http://webserverlab.com
curl http://127.0.0.1
```

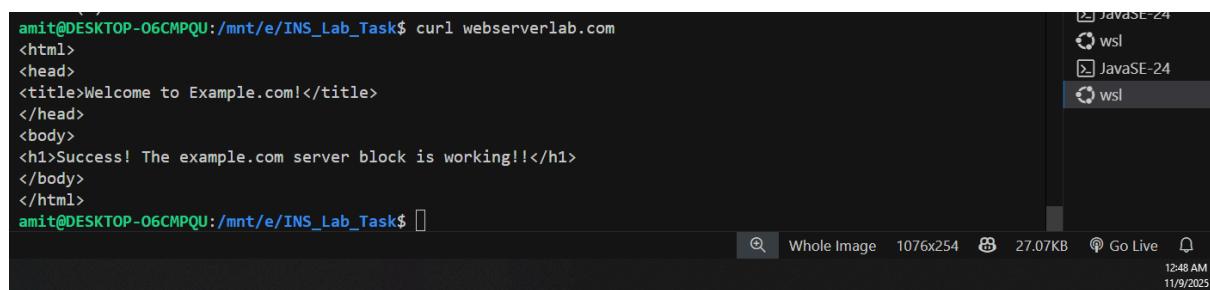
Observation:

- `webserverlab.com` now points to `example.com` content because the default site was disabled.
- Requests to `127.0.0.1` also serve `example.com` content.

Reason: Apache serves the first matching virtual host. Without the default site, requests without a specific `ServerName` match the first enabled virtual host (example.com).

```
</html>
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ curl localhost
<html>
<head>
<title>Welcome to Example.com!</title>
</head>
<body>
<h1>Success! The example.com server block is working!!</h1>
</body>
</html>
```

```
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ curl webserverlab.com
<html>
<head>
<title>Welcome to Example.com!</title>
</head>
<body>
<h1>Success! The example.com server block is working!!</h1>
</body>
</html>
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$
```



✓ **Checkpoint 3:** Virtual host behavior demonstrated and explained.

Step 4: Multiple Virtual Hosts – anothervhhost.com

Repeat steps to create another virtual host with a different HTML file:

Test:

```
curl http://anothervhhost.com
```

The screenshot shows a terminal window titled 'INS_Lab_Task' within a code editor interface. The terminal output displays the configuration of two virtual hosts:

```
<head>
<title>Welcome to Example.com!</title>
</head>
<body>
<h1>Success! The example.com server block is working!</h1>
</body>
</html>
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo mkdir -p /var/www/anothervhost.com/html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo chown -R $USER:$USER /var/www/anothervhost.com/html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo chmod -R 755 /var/www/anothervhost.com
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ nano /var/www/anothervhost.com/html/index.html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo nano /etc/apache2/sites-available/anothervhost.com.conf
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo a2ensite anothervhost.com.conf
Enabling site anothervhost.com.
To activate the new configuration, you need to run:
    systemctl reload apache2
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo systemctl reload apache2
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo nano /etc/hosts
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ curl anothervhost.com
<html>
<head>
<title>Welcome to anothervhost.com!</title>
</head>
<body>
<h1>Success! The anothervhost.com server block is working!</h1>
</body>
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$
```

The terminal also shows the configuration of the hosts file and the curl command to test the deployed host.

✓ **Checkpoint 4:** Multiple virtual hosts successfully deployed.

Task 3: Hosting Dynamic Websites Using HTML & JavaScript

- Two dynamic websites were created using HTML forms and JavaScript.
- Example: `example.com` collects user input and displays it dynamically using JS.

Test via terminal:

```
curl http://example.com/form.html
```

The screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the command `curl example.com` and its output, which is the HTML code for a simple calculator. The code includes a form with fields for two numbers and an operation selection.

```
</body>
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo nano /var/www/example.com/html/index.html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ curl example.com
<!DOCTYPE html>
<html>
<head>
    <title>Simple Calculator</title>
</head>
<body>

<h2>Simple Calculator</h2>

<form id="calcForm">
    <label>Number 1:</label>
    <input type="number" id="num1" required><br><br>

    <label>Number 2:</label>
    <input type="number" id="num2" required><br><br>

    <label>Operation:</label>
    <select id="operation">
        <option value="add">Addition</option>
        <option value="sub">Subtraction</option>
        <option value="mul">Multiplication</option>
        <option value="div">Division</option>
    </select><br><br>
```

curl <http://anothervhhost.com/form.html>

The screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the command `curl anothervhhost.com` and its output, which is the HTML code for a BMI calculator. The code includes a form for height and weight, and a script for handling the submission.

```
</html>
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ sudo nano /var/www/anothervhhost.com/html/index.html
amit@DESKTOP-06CMPQU:/mnt/e/INS_Lab_Task$ curl anothervhhost.com
<!DOCTYPE html>
<html>
<head>
    <title>BMI Calculator</title>
</head>
<body>

<h2>BMI Calculator</h2>

<form id="bmiForm">
    <label>Height (cm):</label>
    <input type="number" id="height" required><br><br>

    <label>Weight (kg):</label>
    <input type="number" id="weight" required><br><br>

    <button type="submit">Compute BMI</button>
</form>

<h3 id="bmiResult"></h3>

<script>
document.getElementById("bmiForm").addEventListener("submit", function(e) {
```

✓ **Checkpoint 5:** Two dynamic websites deployed successfully.

Conclusion

1. Apache was successfully installed and verified.

2. Virtual hosts were configured, including multiple virtual hosts.
3. Dynamic websites using HTML & JavaScript were hosted.
4. Apache service management commands were demonstrated.

All 5 checkpoints completed successfully. ✓