Comprehensive Linux Operations

Project Overview

This project spans various aspects of Linux system administration, including file management, user and group management, service control, process handling, and more. You will be completing tasks that simulate real-world scenarios, providing hands-on experience with Linux commands and configurations.

Project Breakdown

Part 1: Creating and Editing Text Files (20 minutes)

Scenario: You are tasked with documenting the configurations and settings for a new server. You'll use different text editors to create and update these documents.

1. **Using Nano**

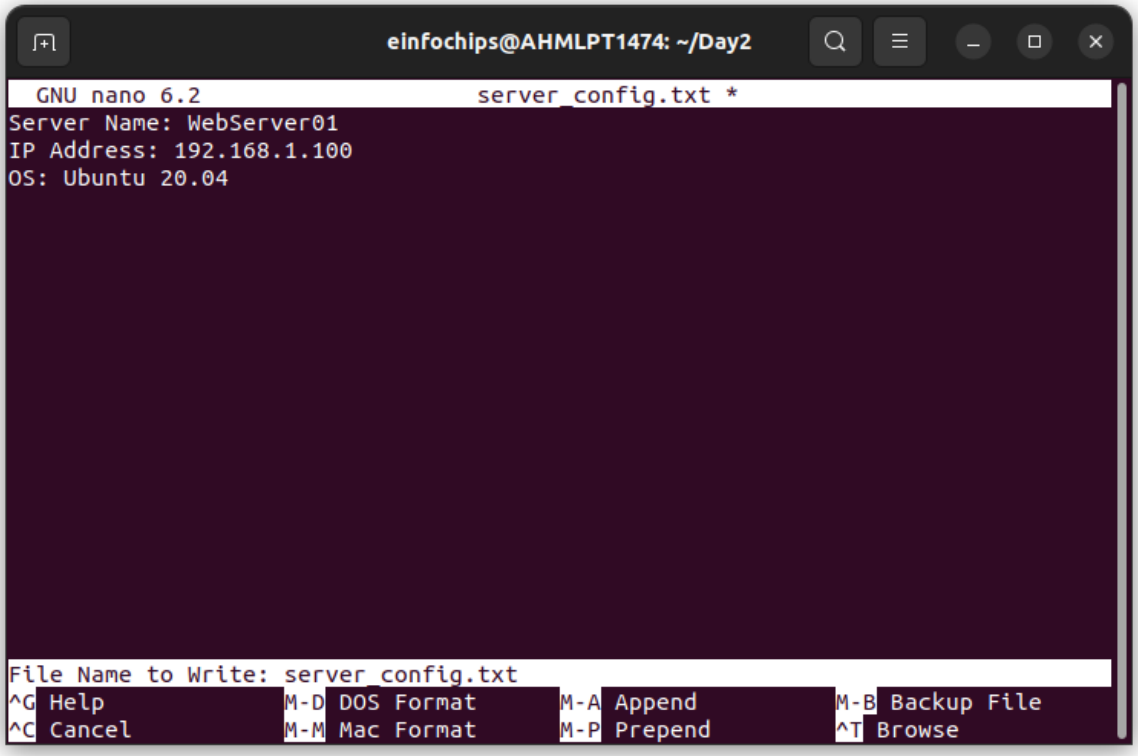
Create a file `server_config.txt` using Nano:

```
nano server_config.txt
```

Add the following content:

```
Server Name: WebServer01
IP Address: 192.168.1.100
OS: Ubuntu 20.04
```

- Save and exit (Ctrl+O, Enter, Ctrl+X).



2. **Using Vi**

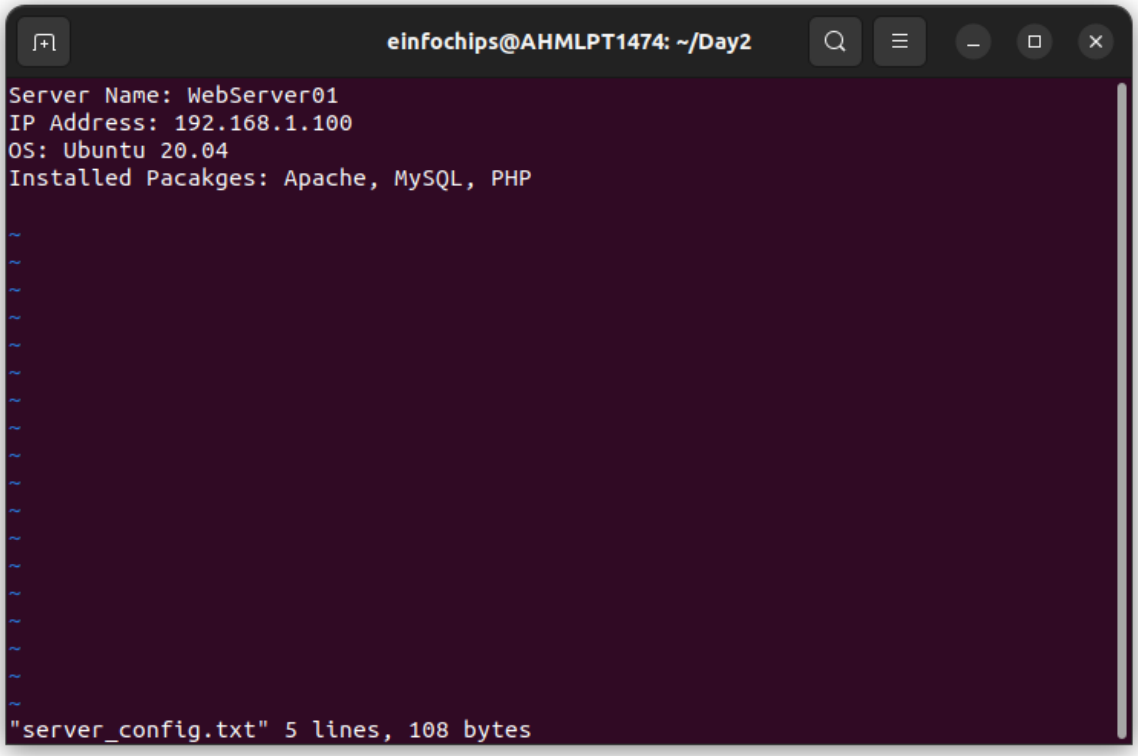
Edit the same file with Vi:

vi server_config.txt

Append the following text:

Installed Packages: Apache, MySQL, PHP

- Save and exit (Esc, :wq).



3. **Using Vim**

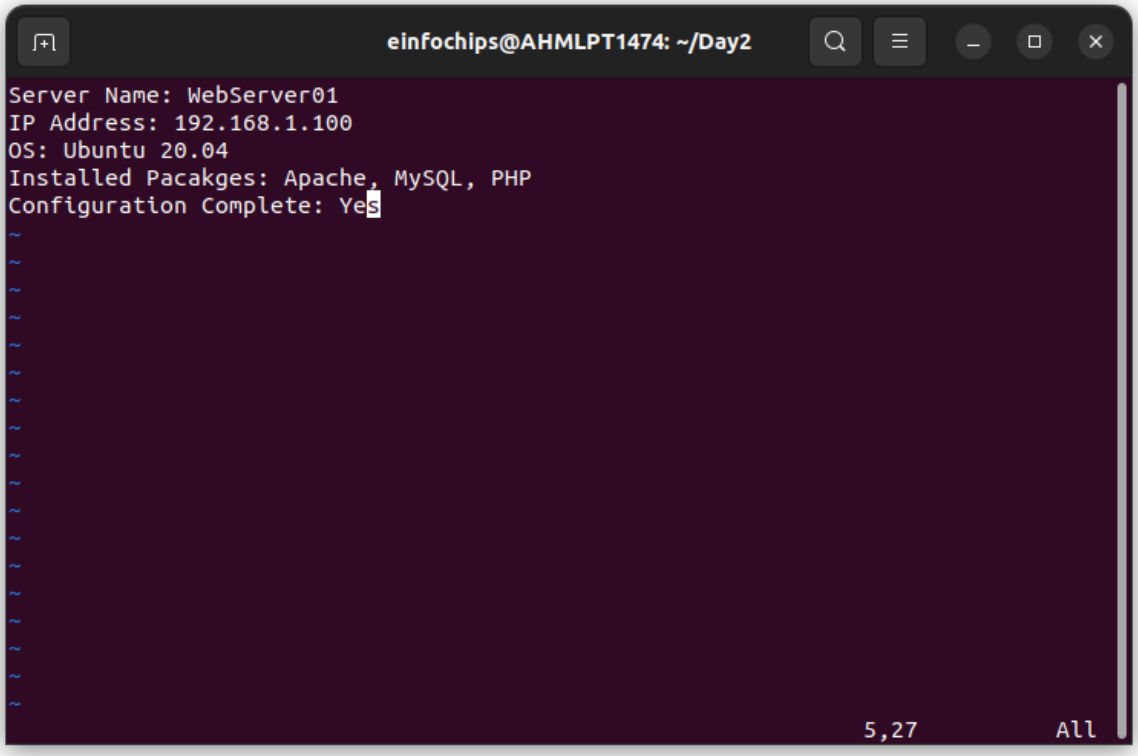
Further edit the file with Vim:

vim server_config.txt

Add the following text:

Configuration Complete: Yes

- Save and exit (Esc, :wq).



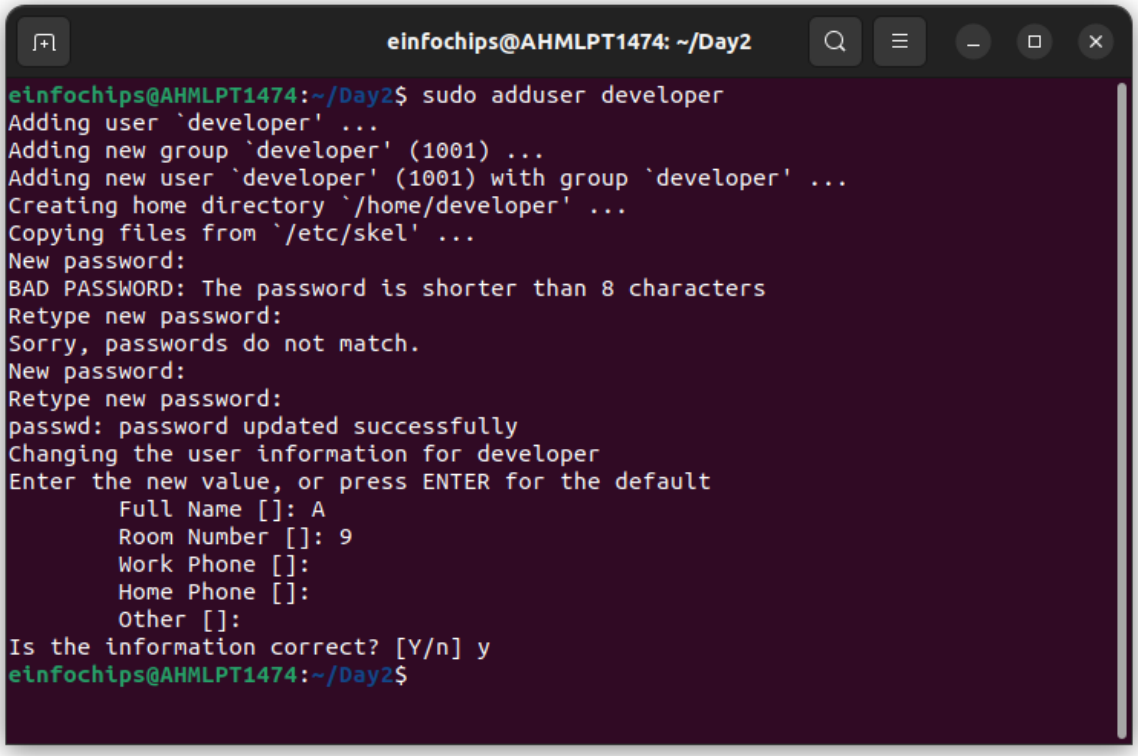
Part 2: User & Group Management (20 minutes)

Scenario: You need to set up user accounts and groups for a new team joining the project.

1. **Adding/Removing Users**

Add a new user developer:

sudo adduser developer



Remove the user developer:

sudo deluser developer

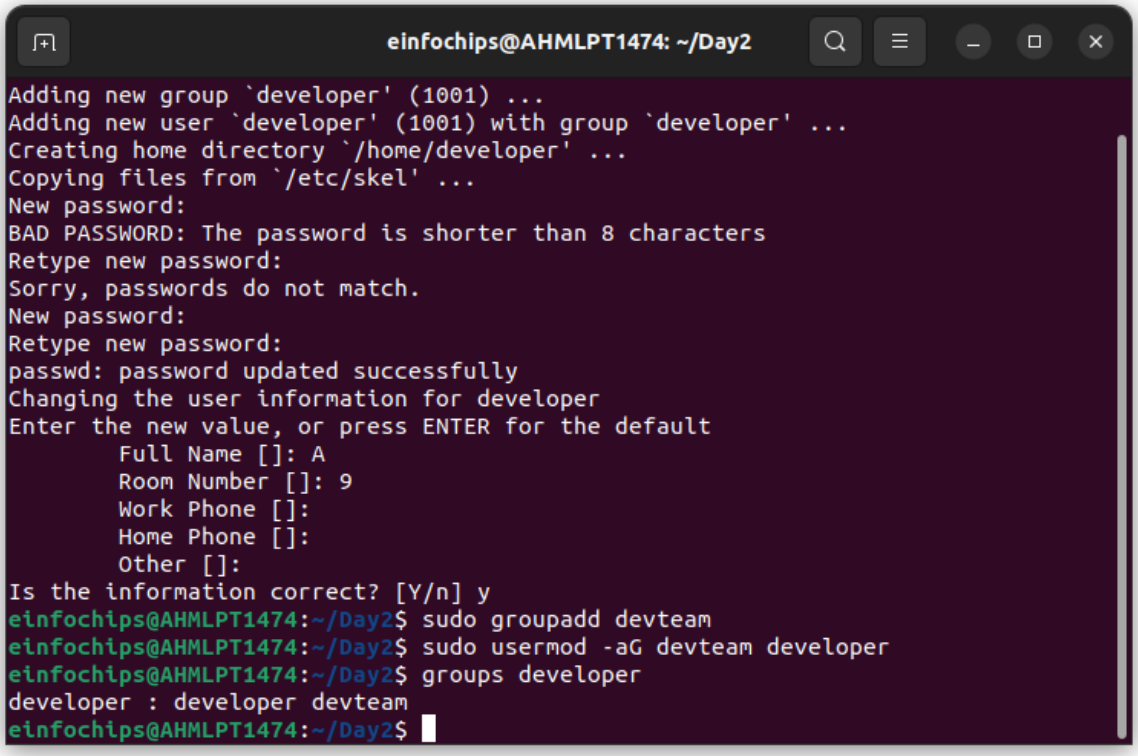
2. **Managing Groups**

Create a group devteam:

sudo groupadd devteam

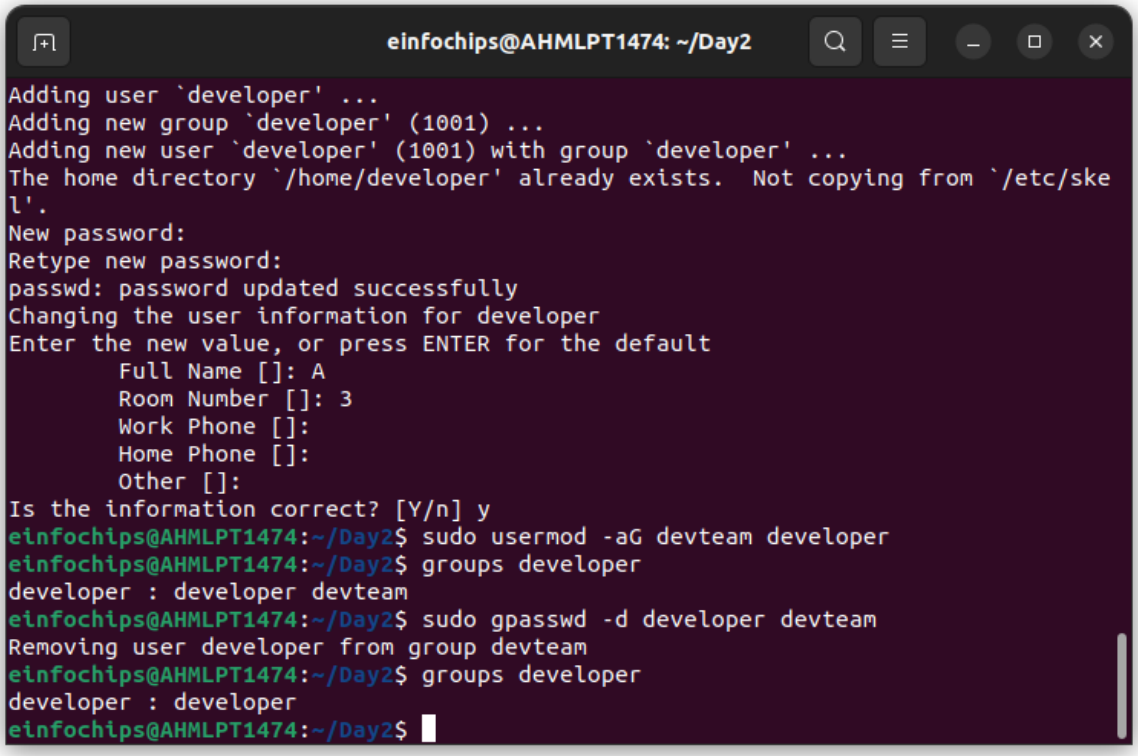
Add the user developer to the devteam group:

sudo usermod -aG devteam developer

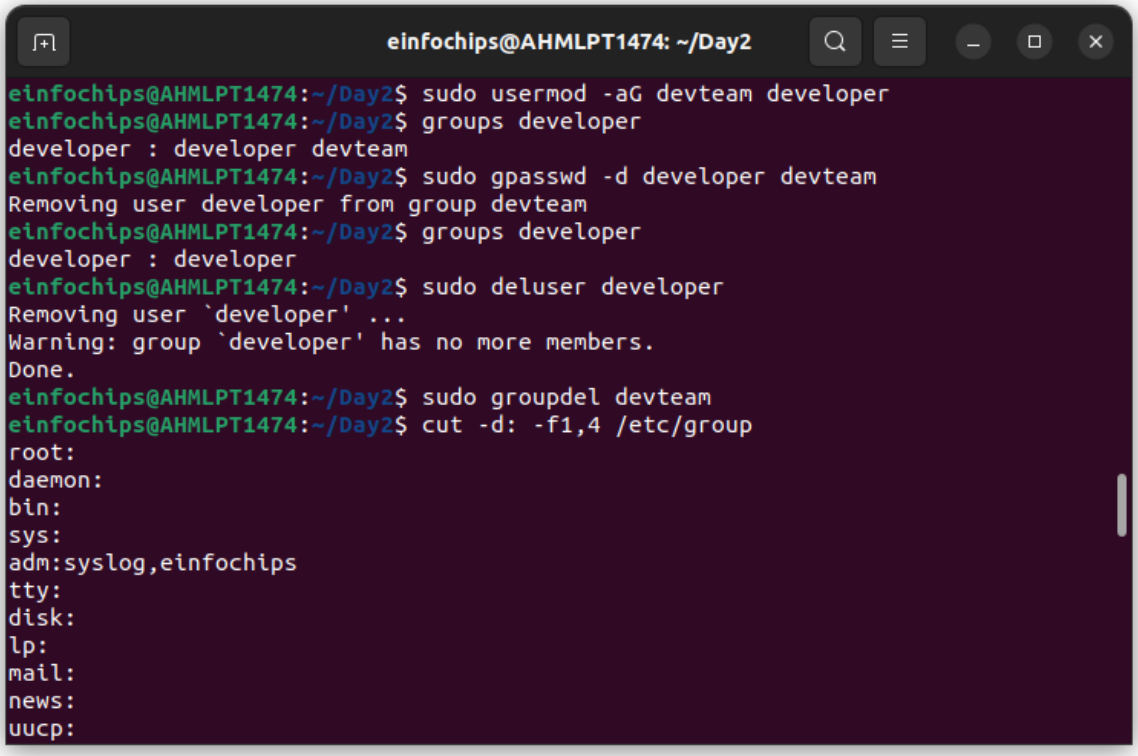


Remove the user developer from the devteam group:

sudo gpasswd -d developer devteam



Delete group:



Part 3: File Permissions Management (20 minutes)

Scenario: Ensure that only the appropriate users have access to specific files and directories.

1. **Understanding File Permissions**

View permissions for server_config.txt:

ls -l server_config.txt

   - Discuss the output (e.g., -rw-r--r--).

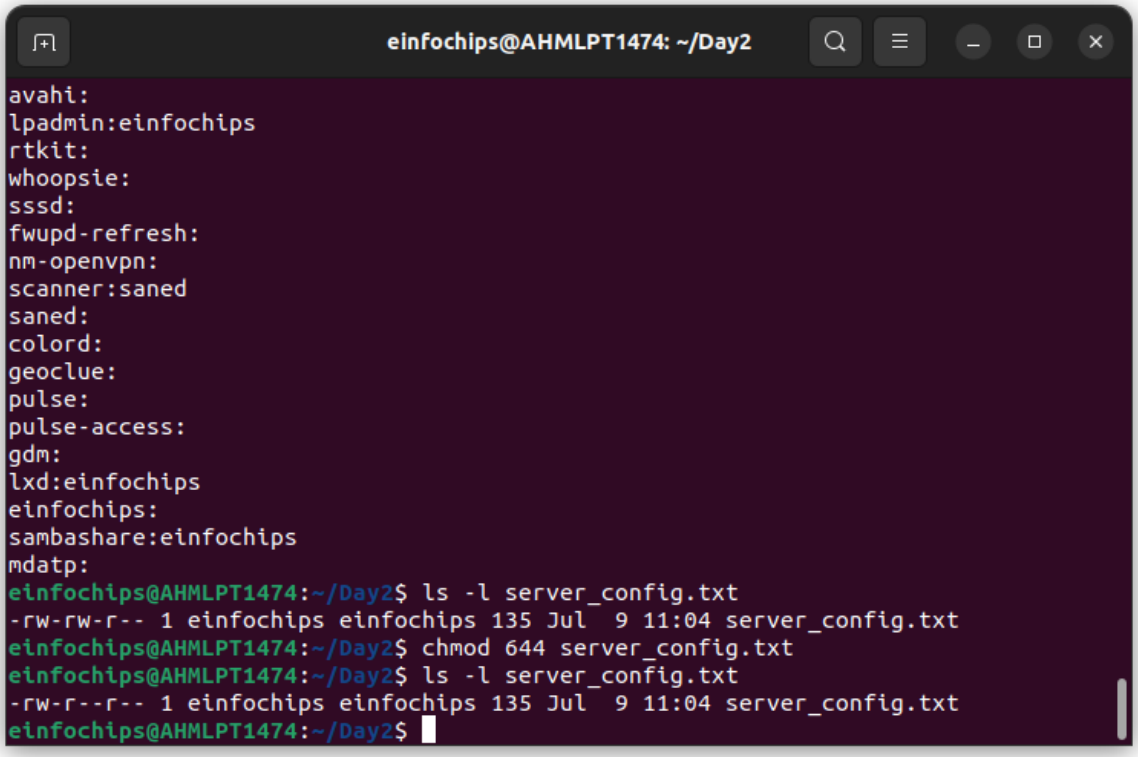
2. **Changing Permissions and Ownership**

Change permissions to read/write for the owner and read-only for others:

chmod 644 server_config.txt

Verify the change:

ls -l server_config.txt



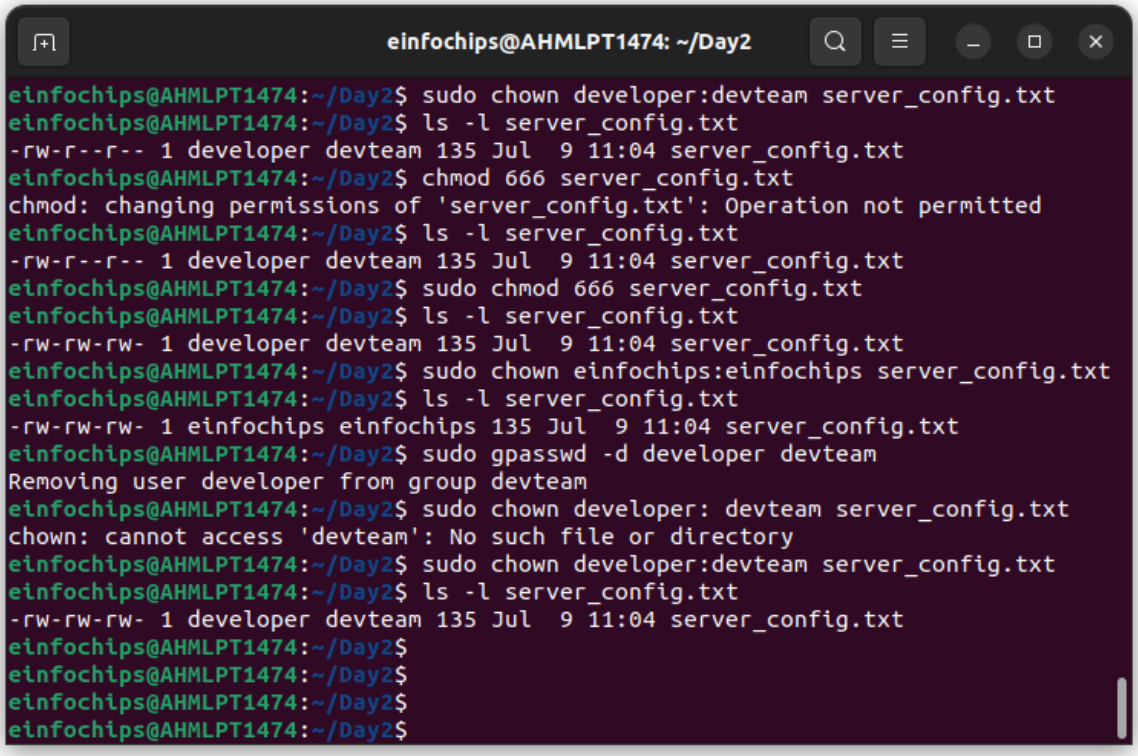
Change the owner to developer and the group to devteam:

sudo chown developer:devteam server_config.txt

○

Verify the change:

ls -l server_config.txt



Part 4: Controlling Services and Daemons (20 minutes)

Scenario: Manage the web server service to ensure it is running correctly and starts on boot.

1. **Managing Services with systemctl**

Start the Apache service:

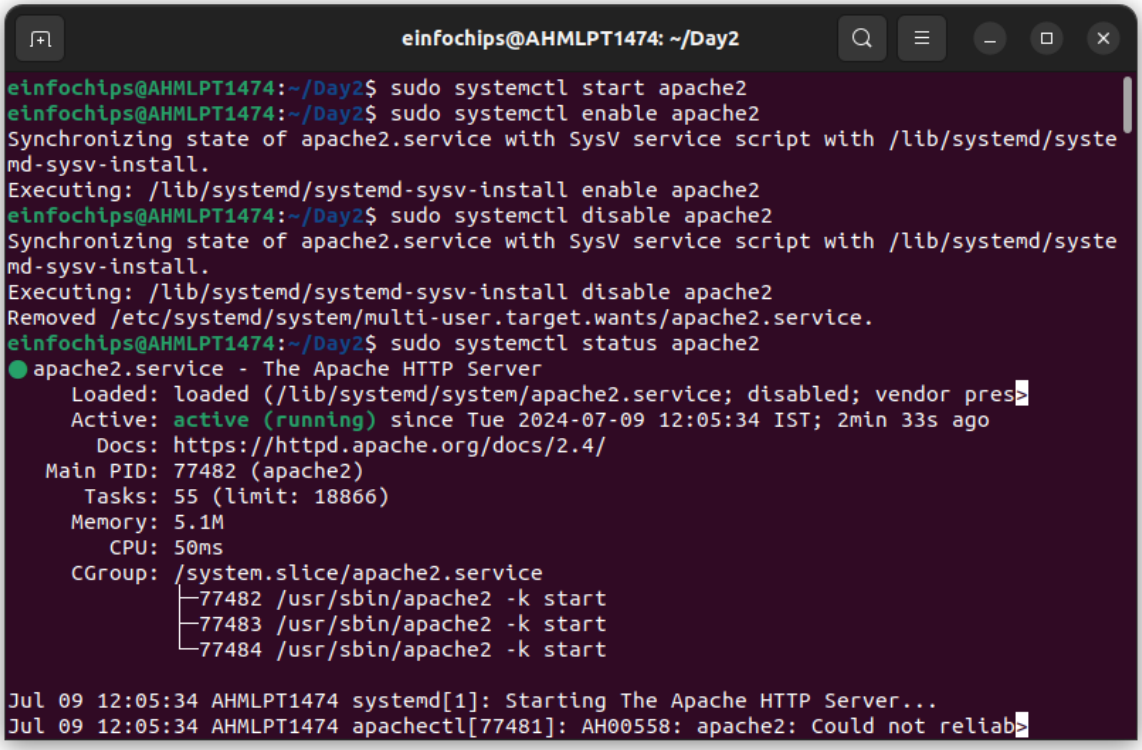
sudo systemctl start apache2

Enable the Apache service to start on boot:

sudo systemctl enable apache2

Disable the Apache service:

sudo systemctl disable apache2

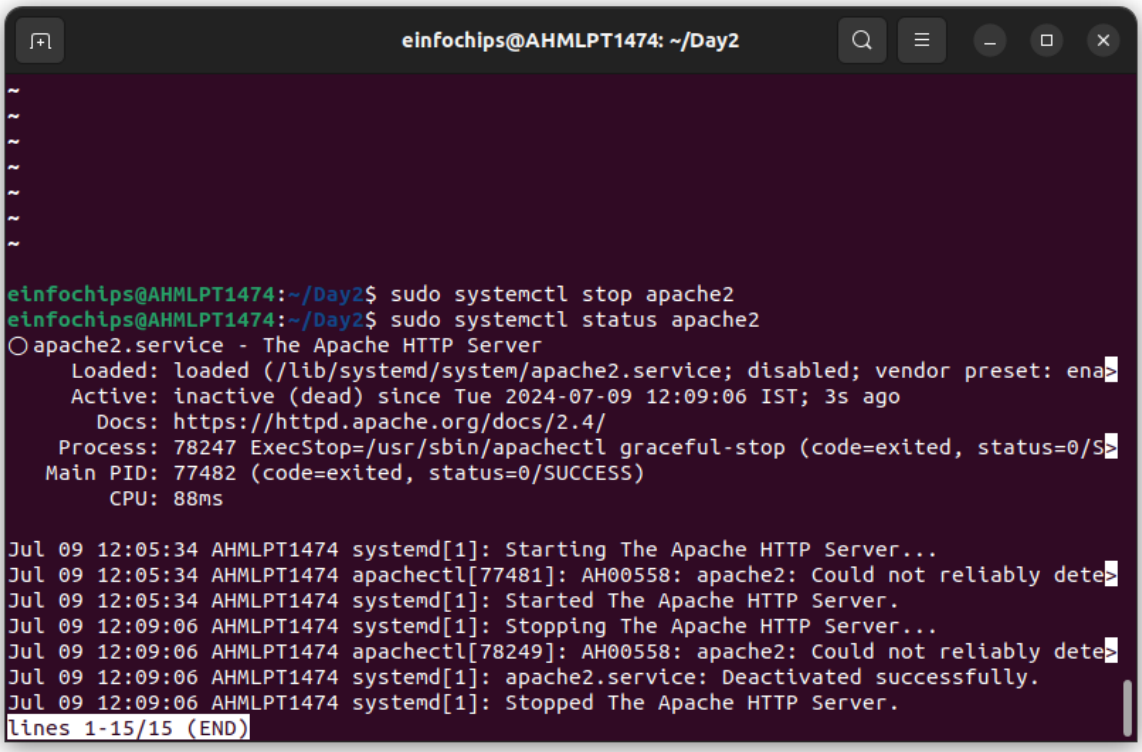


Stop the Apache service:

sudo systemctl stop apache2

Check the status of the Apache service:

sudo systemctl status apache2



2. **Understanding Daemons**
    - Discuss the role of the sshd daemon in providing SSH access to the server.

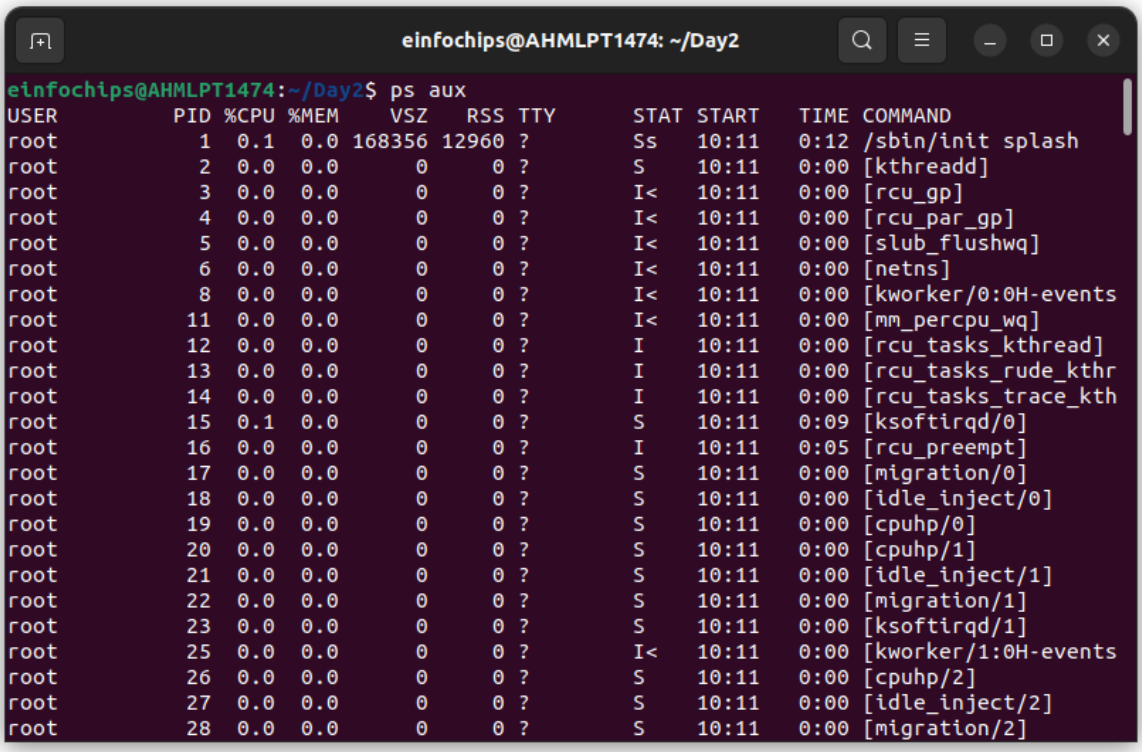
Part 5: Process Handling (20 minutes)

Scenario: Monitor and manage processes to ensure the server is performing optimally.

1. **Viewing Processes**

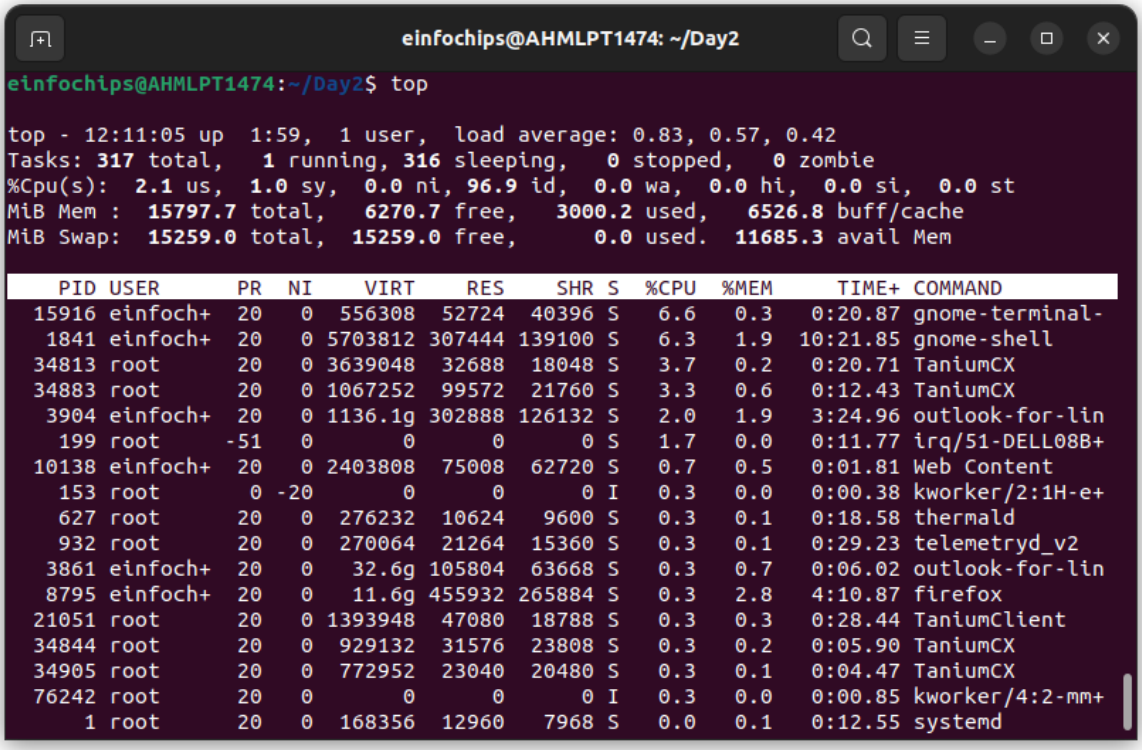
List all running processes:

ps aux



Use top to view processes in real-time:

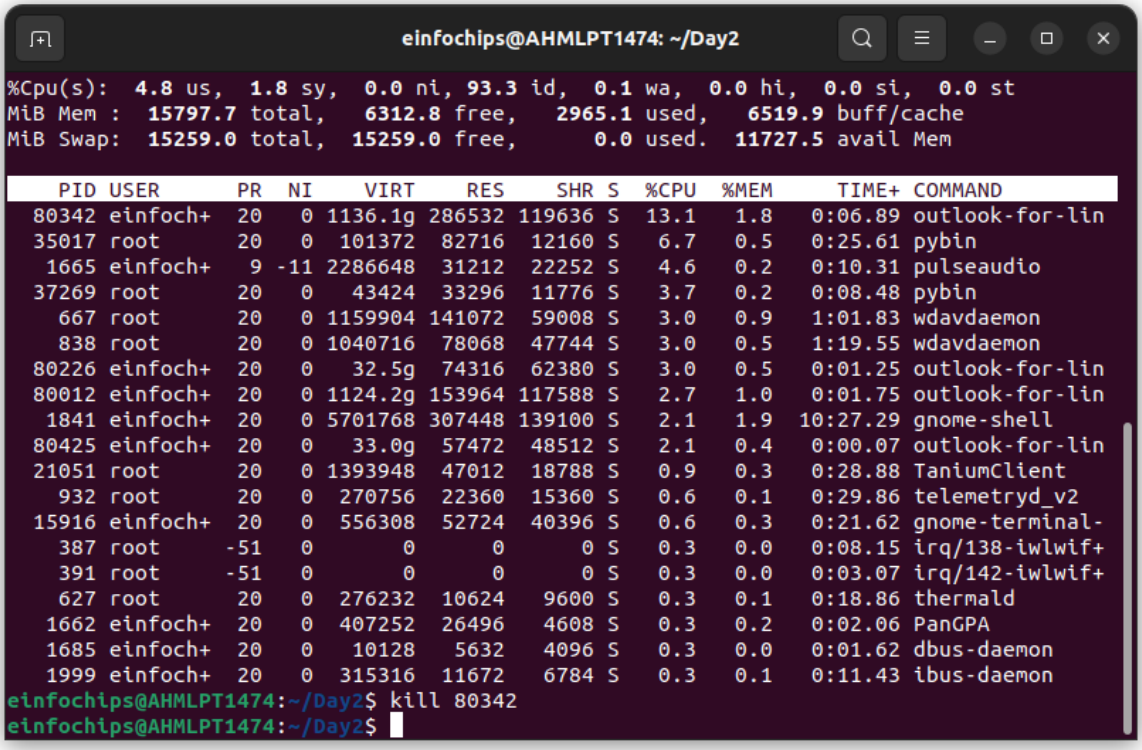
```
top
```



2. **Managing Processes**

Identify a process to kill using ps or top, then kill it:

kill <PID>



Change the priority of a process (e.g., running sleep with a lower priority):

nice -n 10 sleep 100 &

-

Change the priority of the process using renice:

renice +10 <PID>

Creating and Deploying a Static Website with Apache2

Preparation (5 minutes)

- Ensure you have access to a Linux environment (e.g., virtual machines, EC2 instances, or local installations) with sudo privileges.

Activity Breakdown

Part 1: Installing Apache2 (5 minutes)

1. **Update Package Lists**

Open the terminal and run:

```
sudo apt update
```

2. **Install Apache2**

Install Apache2 by running:

```
sudo apt install apache2
```

3. **Start and Enable Apache2**

Start the Apache2 service:

```
sudo systemctl start apache2
```

Enable Apache2 to start on boot:

```
sudo systemctl enable apache2
```

4. **Verify Installation**
   - Open a web browser and navigate to http://your_server_ip. You should see the Apache2 default page.

Part 2: Creating the Website (10 minutes)

1. **Navigate to the Web Directory**

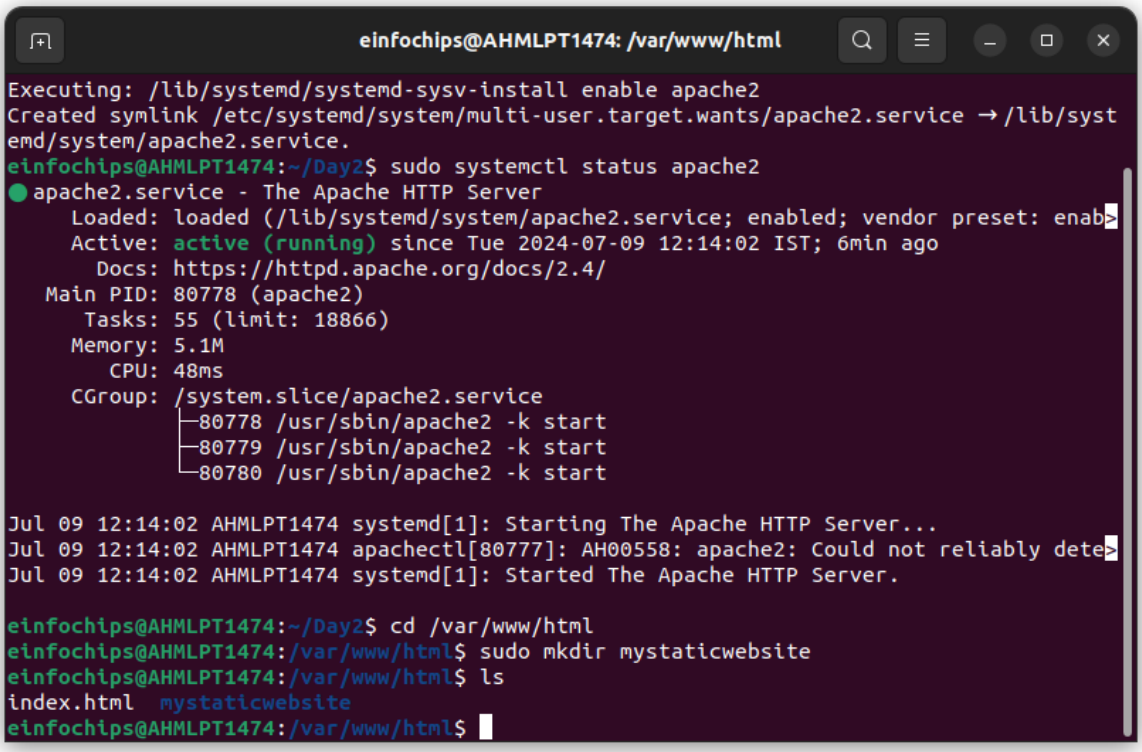
Change to the web root directory:

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

2. **Create a New Directory for the Website**

Create a directory named mystaticwebsite:

```
sudo mkdir mystaticwebsite
```



Change ownership of the directory:

sudo chown -R $USER:$USER /var/www/html/mystaticwebsite

-

3. **Create HTML File**

Create and edit the index.html file:

nano /var/www/html/mystaticwebsite/index.html

-

Add the following content:

```
<!DOCTYPE html>
<html>
<head>
  <title>My Static Website</title>
  <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
```

```
  <h1>Welcome to My Static Website</h1>
  <p>This is a simple static website using Apache2.</p>
  <script src="script.js"></script>
</body>
</html>
```

-
- Save and exit (Ctrl+O, Enter, Ctrl+X).

4. **Create CSS File**

Create and edit the styles.css file:

```
nano /var/www/html/mystaticwebsite/styles.css
```

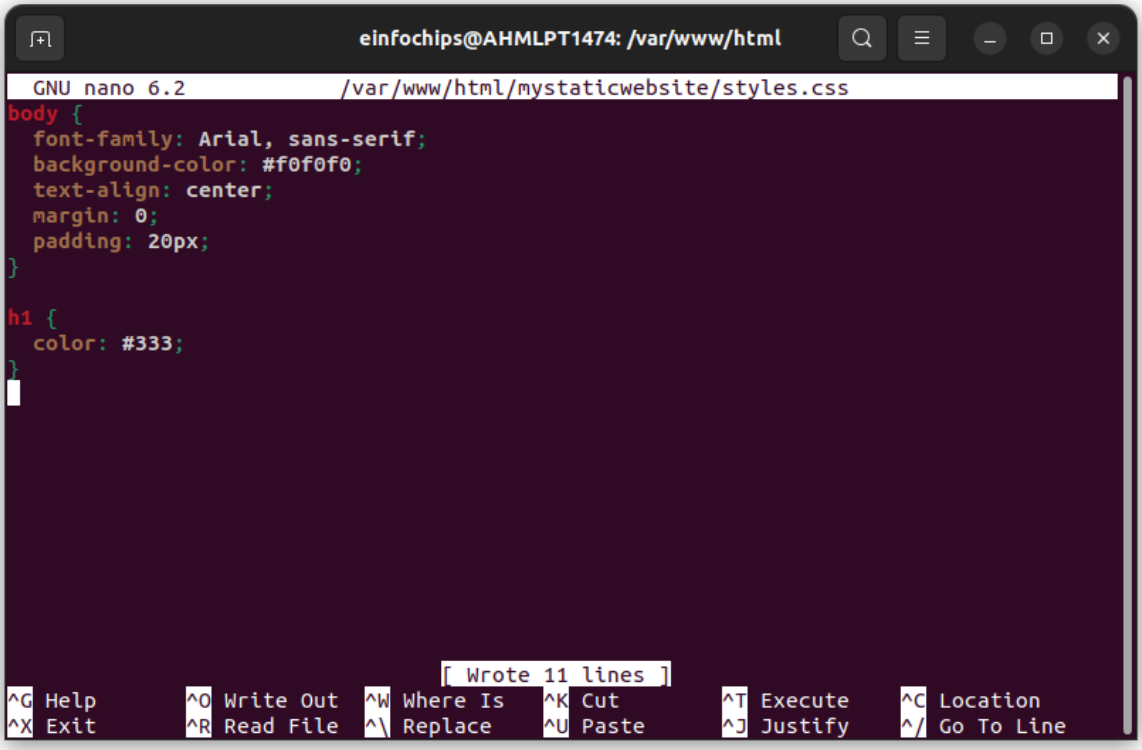
-

Add the following content:

```
body {
  font-family: Arial, sans-serif;
  background-color: #f0f0f0;
  text-align: center;
  margin: 0;
  padding: 20px;
}

h1 {
  color: #333;
}
```

-
- Save and exit (Ctrl+O, Enter, Ctrl+X).



5. **Create JavaScript File**

Create and edit the script.js file:

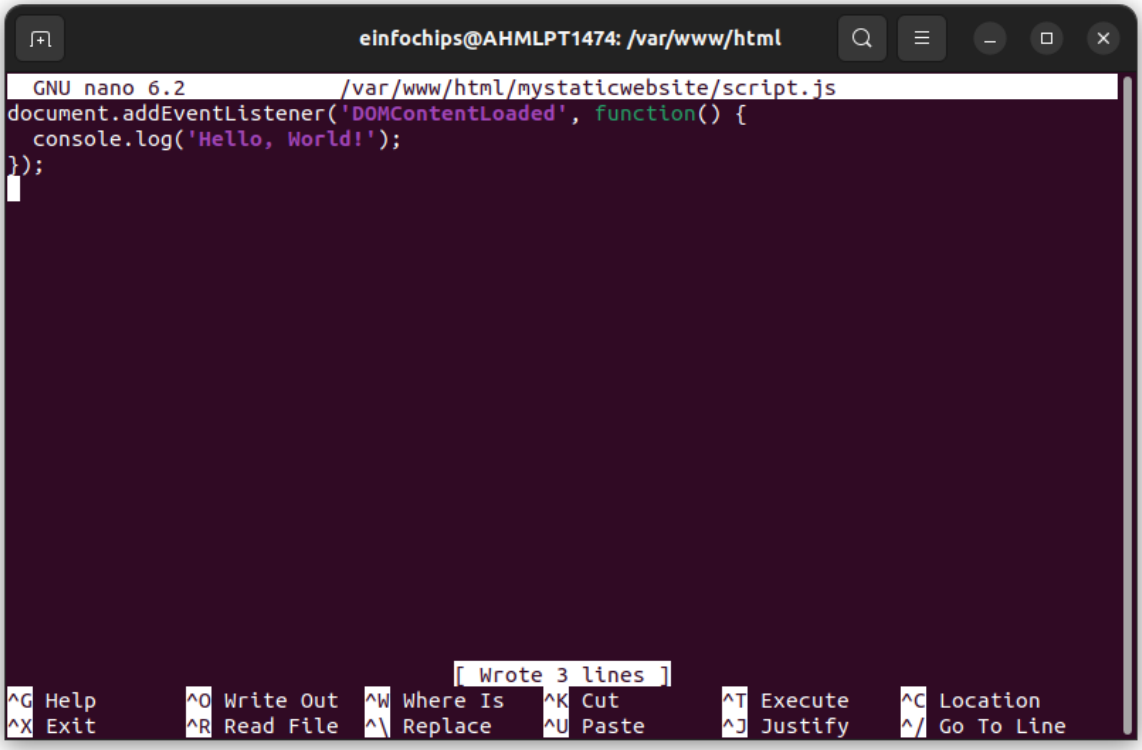
nano /var/www/html/mystaticwebsite/script.js

-

Add the following content:

```
document.addEventListener('DOMContentLoaded', function() {
  console.log('Hello, World!');
});
```

-
- Save and exit (Ctrl+O, Enter, Ctrl+X).



6. **Add an Image**

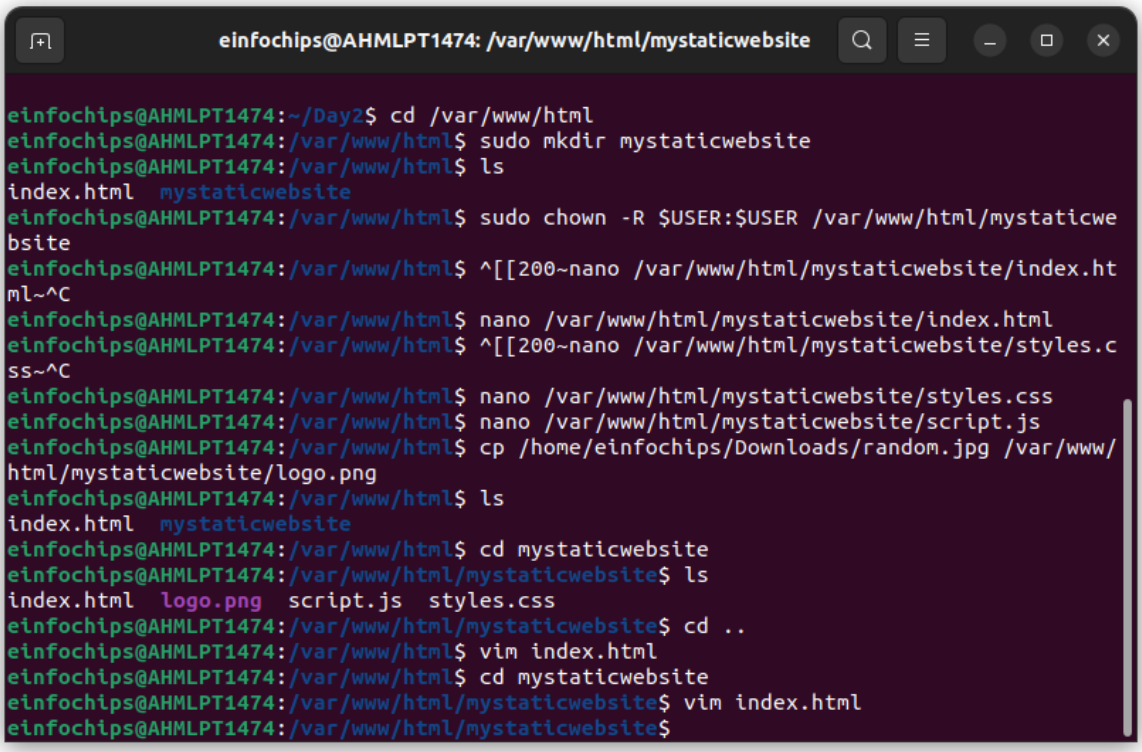
Download or copy an image file (e.g., logo.png) to the website directory:

```
cp /path/to/your/logo.png /var/www/html/mystaticwebsite/logo.png
```

-

Update index.html to include the image:

```
<body>
  <h1>Welcome to My Static Website</h1>
  <img src="logo.png" alt="Logo">
  <p>This is a simple static website using Apache2.</p>
  <script src="script.js"></script>
</body>
```



Part 3: Configuring Apache2 to Serve the Website (10 minutes)

1. **Create a Virtual Host File**

Create and edit the virtual host configuration file:

sudo nano /etc/apache2/sites-available/mystaticwebsite.conf

-

Add the following content:

<VirtualHost *:80>

ServerAdmin webmaster@localhost

DocumentRoot /var/www/html/mystaticwebsite

ErrorLog ${APACHE_LOG_DIR}/error.log

CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>

-
- Save and exit (Ctrl+O, Enter, Ctrl+X).

2. **Enable the New Virtual Host**

Enable the virtual host configuration:

sudo a2ensite mystaticwebsite.conf

3. **Disable the Default Site**

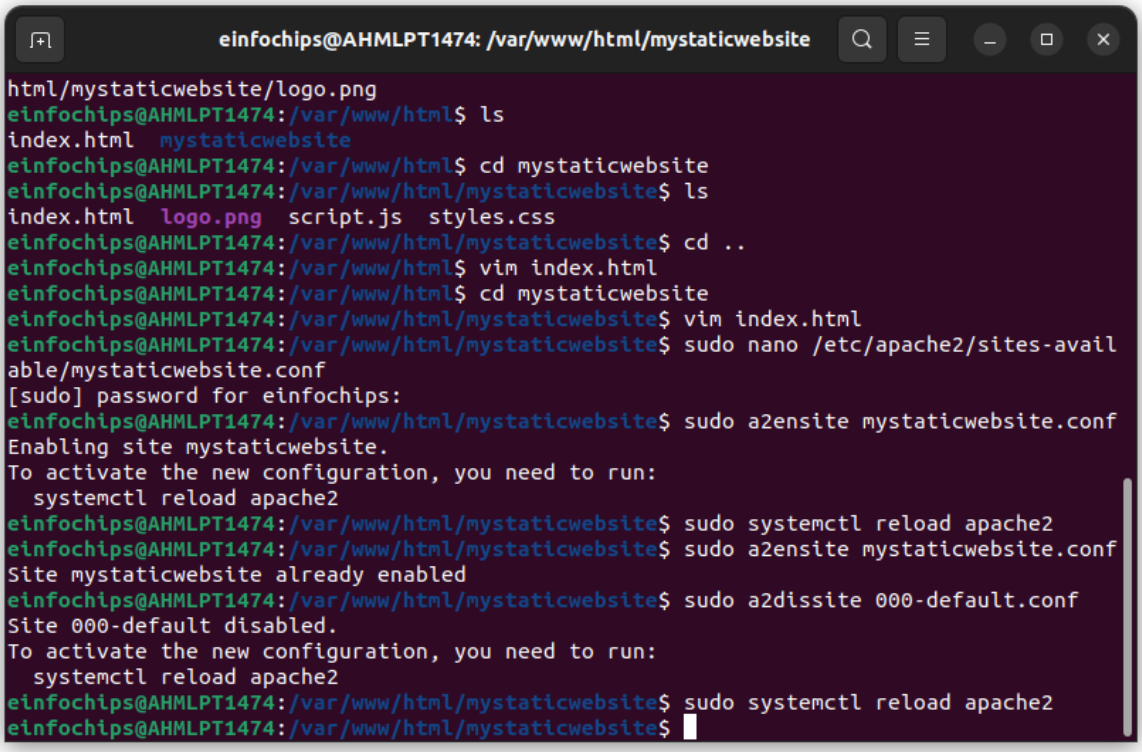
Disable the default site configuration:

sudo a2dissite 000-default.conf

4. **Reload Apache2**

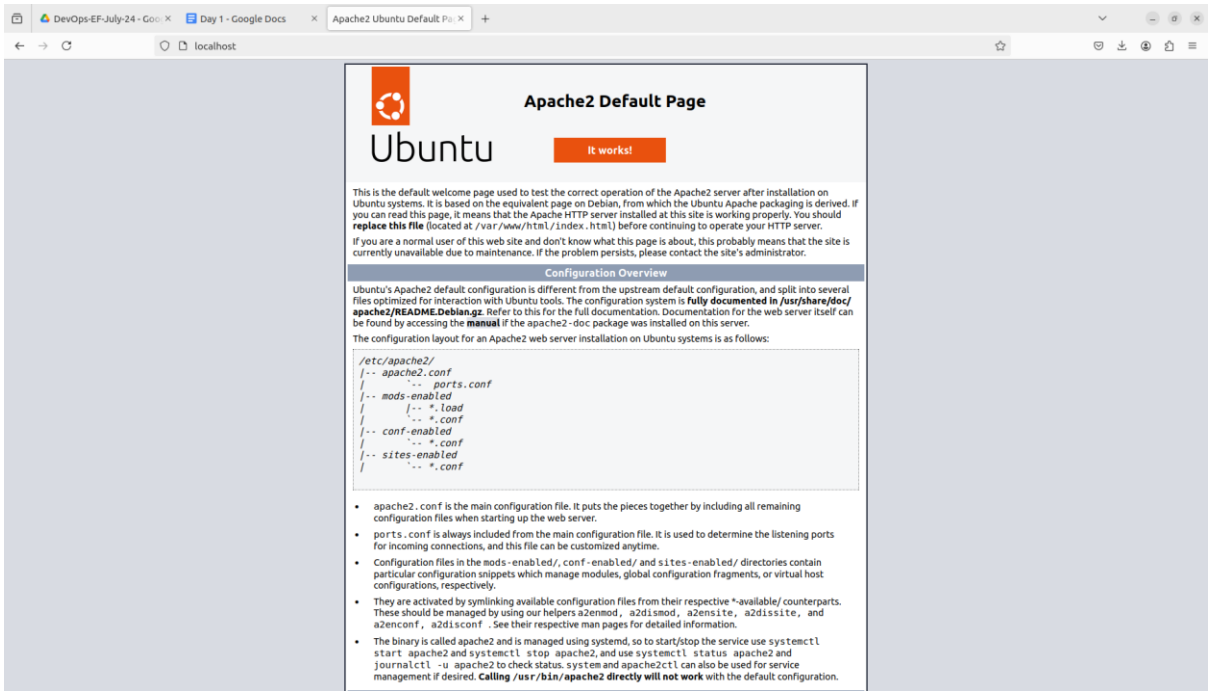
Reload the Apache2 service to apply the changes:

sudo systemctl reload apache2

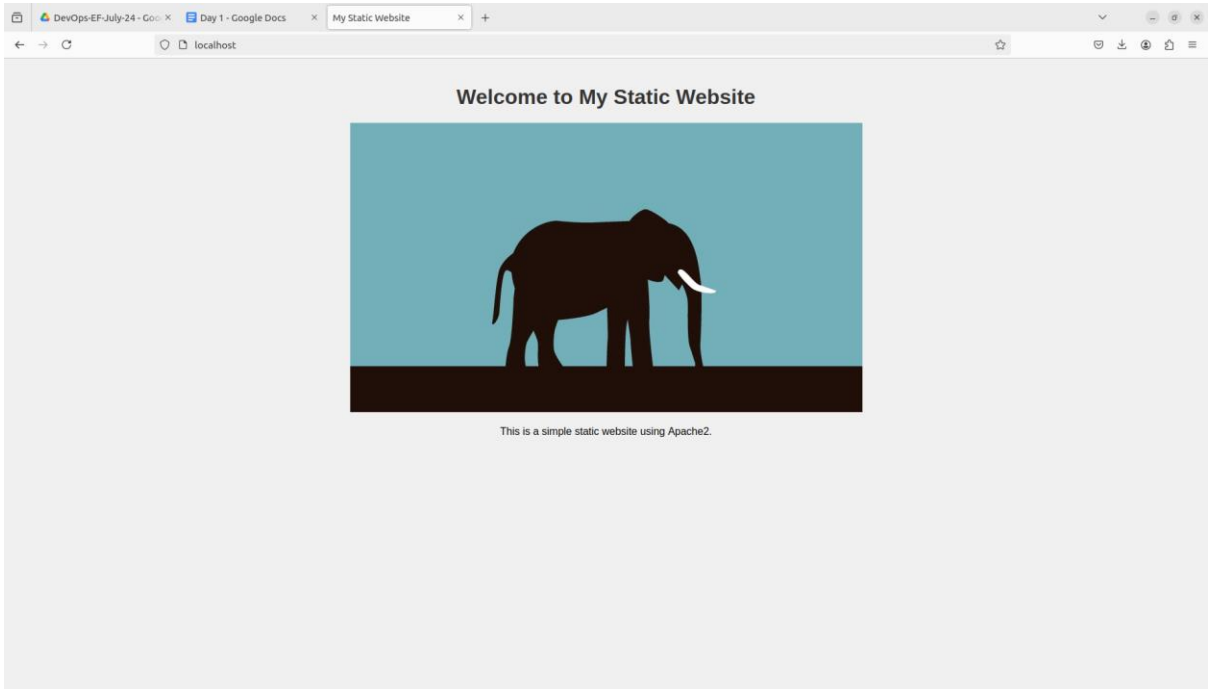


5. **Test the Configuration**
   - Open a web browser and navigate to http://your_server_ip. You should see the static website with the HTML, CSS, JS, and image.

Default:


DevOps-EF-July-24 - Goo
Day 1 - Google Docs
Apache2 Ubuntu Default Pa
localhost
Apache2 Default Page
Ubuntu
It works!
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server.
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.
Configuration Overview
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is fully documented in /usr/share/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the manual if the apache2-doc package was installed on this server.
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:
/etc/apache2/
|-- apache2.conf
| `-- ports.conf
|-- mods-enabled
| |-- *.load
| `-- *.conf
|-- conf-enabled
| `-- *.conf
|-- sites-enabled
| `-- *.conf
apache2.conf is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
ports.conf is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
Configuration files in the mods-enabled/, conf-enabled/ and sites-enabled/ directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers a2enmod, a2dismod, a2ensite, a2dissite, and a2enconf, a2disconf . See their respective man pages for detailed information.
The binary is called apache2 and is managed using systemd, so to start/stop the service use systemctl start apache2 and systemctl stop apache2, and use systemctl status apache2 and journalctl -u apache2 to check status. system and apache2ctl can also be used for service management if desired. Calling /usr/bin/apache2 directly will not work with the default configuration.

New:


DevOps-EF-July-24 - Goo
Day 1 - Google Docs
My Static Website
localhost
Welcome to My Static Website
This is a simple static website using Apache2.