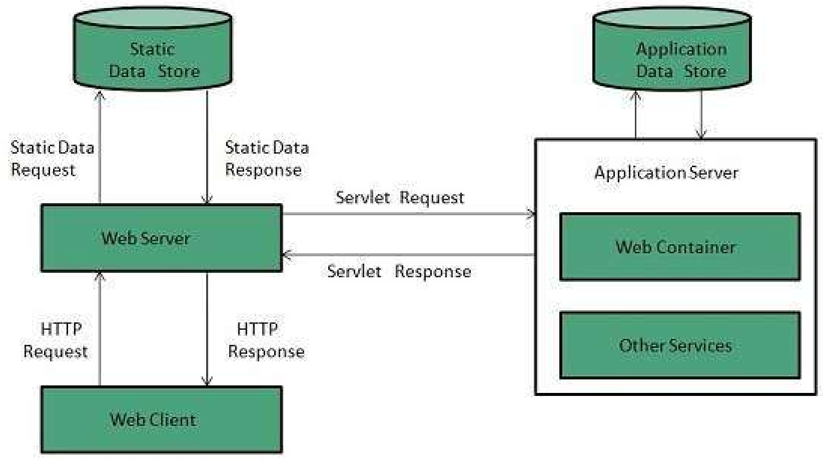
	Experiment No : 8 Date :
Title	**Configuration of Web Server**
Aim	**To install and configure Apache2 webserver**
Hardware Requirement	Personal Computer
Software Requirement	Linux Operating System(Ubuntu 16.04) , Shell-Interpreter
Theory	A web server is software and hardware that uses HTTP (Hypertext Transfer Protocol) and other protocols to respond to client requests made over the World Wide Web. The main job of a web server is to display website content through storing, processing and delivering webpages to users. Besides HTTP, web servers also support SMTP (Simple Mail Transfer Protocol) and FTP (File Transfer Protocol), used for email, file transfer and storage. Web server hardware is connected to the internet and allows data to be exchanged with other connected devices, while web server software controls how a user accesses hosted files. The web server process is an example of the client/server model. All computers that host websites must have web server software. Web servers are used in web hosting, or the hosting of data for websites and web-based applications -- or web applications. **Working of Server** Web server software is accessed through the domain names of websites and ensures the delivery of the site's content to the requesting user. The software side is also comprised of several components, with at least an HTTP server. The HTTP server is able to understand HTTP and URLs. As hardware, a web server is a computer that stores web server software and other files related to a website, such as HTML documents, images and JavaScript files.  Static Data Store Application Data Store Static Data Request Static Data Response Application Server Servlet Request Web Server Web Container Servlet Response HTTP Request HTTP Response Other Services Web Client

When a web browser, like Google Chrome or Firefox, needs a file that's hosted on a web server, the browser will request the file by HTTP. When the request is received by the web server, the HTTP server will accept the request, find the content and send it back to the browser through HTTP.

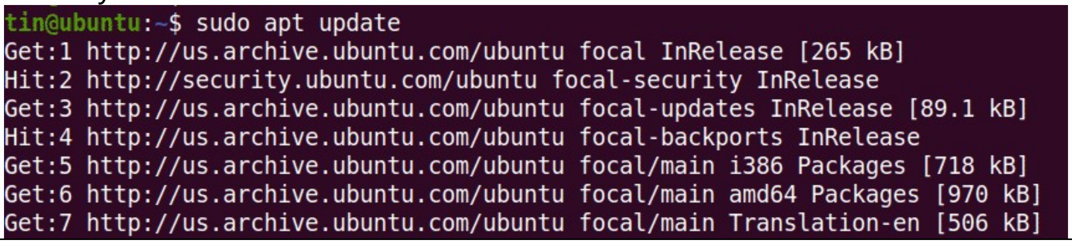
More specifically, when a browser requests a page from a web server, the process will follow a series of steps. First, a person will specify a URL in a web browser's address bar.

The web browser will then obtain the IP address of the domain name -- either translating the URL through DNS (Domain Name System) or by searching in its cache. This will bring the browser to a web server. The browser will then request the specific file from the web server by an HTTP request.
The web server will respond, sending the browser the requested page, again, through HTTP. If the requested page does not exist or if something goes wrong, the web server will respond with an error message. The browser will then be able to display the webpage.
Multiple domains also can be hosted on one web server.

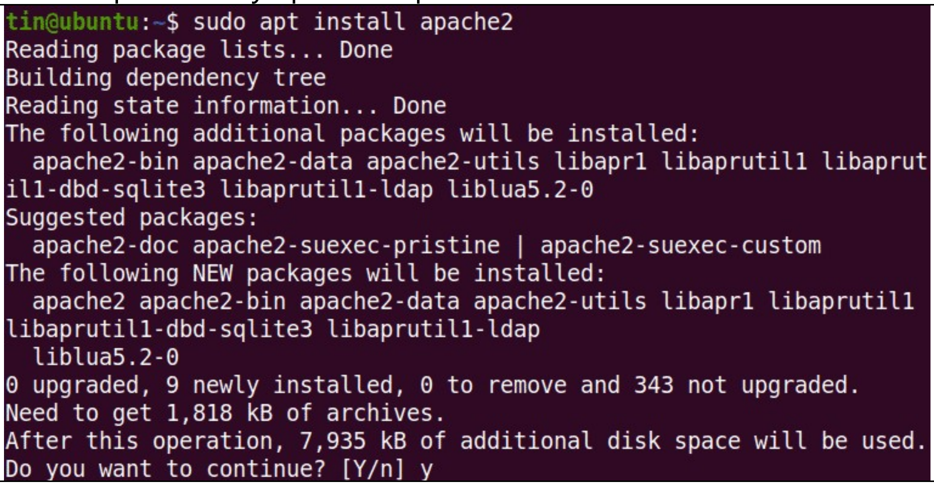
Dynamic and static web servers
A web server may be used as static or dynamic content. The static content is the one that is fixed. The emotional content is something that can be changed and updated. The static web server contains HTTP software and computer. This is static because the server sends hosted files as is present to the browser.

Dynamic web browser will have the webserver and the software like the database and the applications server. This is dynamic because the application server is used to update the files hosted before these are sent to the browser. The web server generates content when the database requests it. The process is flexible but complicated too.

Apache Web Server
Apache web server is the most extensively used open-source web server supported on the majority of the OS including Linux, Windows, MacOS, Solaris, etc. It is highly customizable and can be integrated with other modules. Installing and configuring Apache for basic setup is quite easy. This article will explain how to install and configure the Apache web server on Ubuntu operating system.

Installation Step By Step	
	First, we will need to update the system repository index to install the most recent version of Apache2. To do so, launch the Terminal by using the Ctrl+Alt+T shortcut and execute the following command: 0 seconds of 4 minutes, 33 secondsVolume 0%
Step-1	**Step 1: Update Software Packages** $ sudo apt upgrade $ sudo apt update
	Only an authorized user can install, update, or remove the packages from the Linux system. 

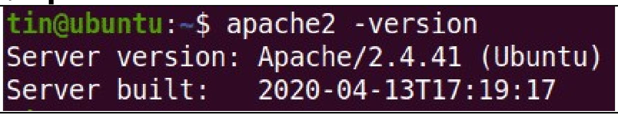
```
tin@ubuntu:~$ sudo apt update
Get:1 http://us.archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Hit:2 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [89.1 kB]
Hit:4 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Get:5 http://us.archive.ubuntu.com/ubuntu focal/main i386 Packages [718 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu focal/main amd64 Packages [970 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu focal/main Translation-en [506 kB]
```

Step-2	**Install Apache2 Webserver**
	To install Apache2 web server. For this, execute the below command in Terminal: $ sudo apt install apache2 or $sudo apt install -y apache2 apache2-utils
	

```
tin@ubuntu:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprut
il1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1
libaprutil1-dbd-sqlite3 libaprutil1-ldap
  liblua5.2-0
0 upgraded, 9 newly installed, 0 to remove and 343 not upgraded.
Need to get 1,818 kB of archives.
After this operation, 7,935 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

	The system might ask for confirmation by providing you with a **Y/n** option. Hit **y** and then Enter to continue. After that, the Apache2 web server and its all dependencies will be installed on your system.
	(Systemctl is **a systemd utility that is responsible for Controlling the systemd system and service manager**. Systemd is a collection of system management daemons, utilities, and libraries which serves as a replacement of System V init daemon.)
	Once installed, verify the version of the Apache server as follows

$ apache2 -version



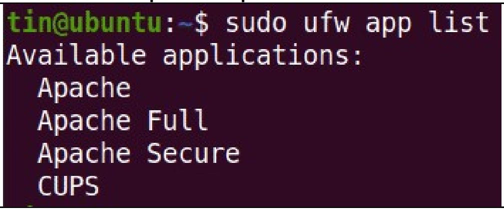
```
tin@ubuntu:~$ apache2 -version
Server version: Apache/2.4.41 (Ubuntu)
Server built:   2020-04-13T17:19:17
```

Step-3 **Firewall configuration**

Now, we will need to open certain ports on our system in order to access Apache from outside. First, let's list the application profiles that we need to give Apache access to. Run the following command to do so:

$ sudo ufw app list

Different apache profiles can be seen.

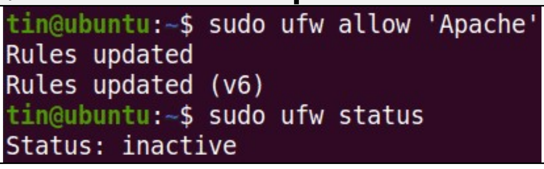


```
tin@ubuntu:~$ sudo ufw app list
Available applications:
  Apache
  Apache Full
  Apache Secure
  CUPS
```

(UFW (**uncomplicated firewall**) is a firewall configuration tool that runs on top of iptables , included by default within Ubuntu distributions. It provides a streamlined interface for configuring common firewall use cases via the command line.)

We will use the highly restrictive profile 'Apache' to enable network activity on port 80.

$ sudo ufw allow 'Apache'



```
tin@ubuntu:~$ sudo ufw allow 'Apache'
Rules updated
Rules updated (v6)
tin@ubuntu:~$ sudo ufw status
Status: inactive
```

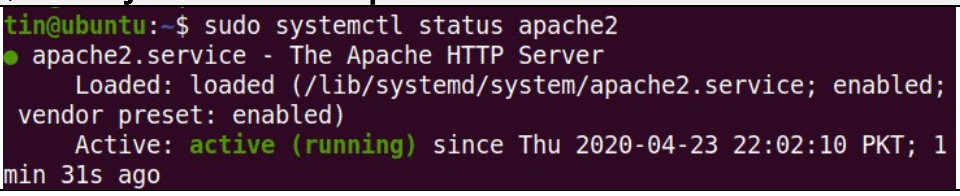
Now check the status which will show Apache allowed in firewall

$ sudo ufw status

Step-4 **Configuring Apache web server; Verifying Apache service**

Before moving towards configuration, first, verify if the Apache service is operational. For this, execute the below command in Terminal:

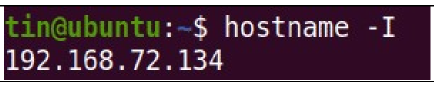
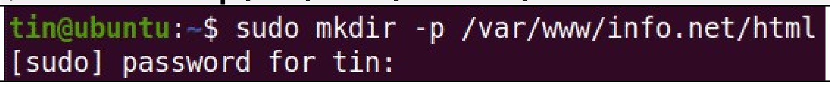
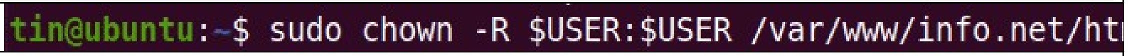
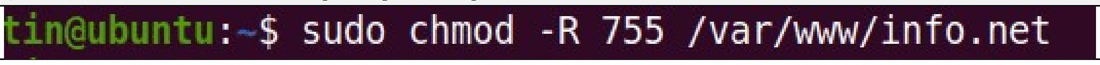
$ sudo systemctl status apache2

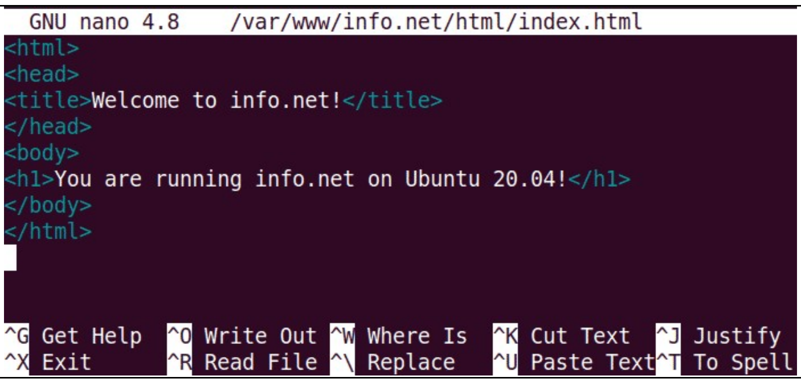


```
tin@ubuntu:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled;
 vendor preset: enabled)
     Active: active (running) since Thu 2020-04-23 22:02:10 PKT; 1
min 31s ago
```

From the above output, you can see the Apache2 service is active and running.
Another approach to verify if Apache is running fine by requesting a web page from the Apache web server. To do so, find your IP address using the following command:

$ hostname –I

	 `tin@ubuntu:~$ hostname -I` `192.168.72.134`
	Then open the web browser and access apache welcome page as follows:
	http://192.168.72.134 Replace the 192.168.72.134 by the IP address of your machine.
	By navigating to the above link in the browser, you see the Apache welcome page which is the indication that the Apache server is working properly.
Setting Up Virtual Hosts in Apache	
	If you have multiple domains that need to be server from the single Apache web server, then you will require to set up virtual hosts. In the following, we will show you how to set up a virtual host in Apache. We will set up the domain name "info.net". Make sure to replace the info.ne with your own domain name.
Step-1	**Create a directory for your domain**
	In this step, we will create a directory for our domain name. This directory will be used for storing the data on our website. Run the following command in Terminal by replacing the info.net with your own domain name:
	$ sudo mkdir -p /var/www/info.net/html
	 `tin@ubuntu:~$ sudo mkdir -p /var/www/info.net/html` `[sudo] password for tin:`
	Change the directory ownership to current user:
	$ sudo chown -R $USER:$USER /var/www/info.net/html
	 `tin@ubuntu:~$ sudo chown -R $USER:$USER /var/www/info.net/ht`
	Assign necessary permissions as follows:
	$ sudo chmod -R 755 /var/www/info.net
	 `tin@ubuntu:~$ sudo chmod -R 755 /var/www/info.net`
Step-2	**Make a sample page for your website**
	We have setup virtual host and assign necessary permission. Now we, will create a sample page for our website. We will create the sample page using Nano editor, however, any text editor can be used for this purpose.
	$ nano /var/www/info.net/html/index.html
	HML code
	`<html>` **`<head>`** **`<title>Welcome to info.net!</title>`** **`</head>`** **`<body>`** **`<h1>You are running info.net on Ubuntu 20.04!</h1>`** **`</body>`** **`</html>`**
	Now use Ctrl+O to save and then Ctrl+X to exit the file.



```
  GNU nano 4.8    /var/www/info.net/html/index.html
<html>
<head>
<title>Welcome to info.net!</title>
</head>
<body>
<h1>You are running info.net on Ubuntu 20.04!</h1>
</body>
</html>


^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell
```

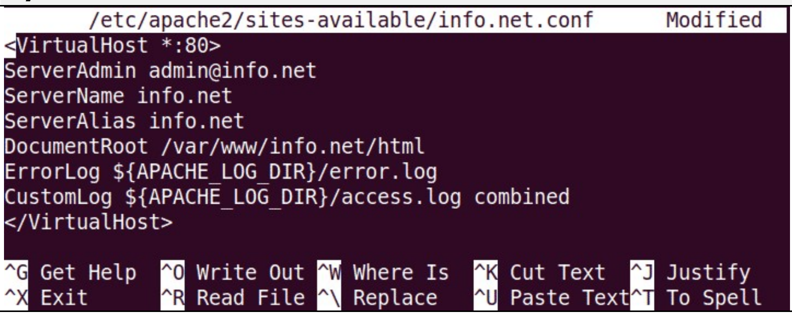
Step-3 **Create a virtual host file**

Apache server comes with virtual host file by default. This file is used to serve the contents of the web server. However, we will generate the new virtual host file with the following command:

$ sudo nano /etc/apache2/sites-available/info.net.conf

Now enter the below lines by replacing the **info.net** by your own domain name.

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



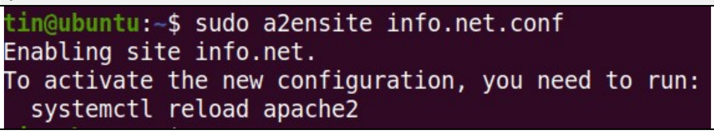
```
       /etc/apache2/sites-available/info.net.conf      Modified
<VirtualHost *:80>
ServerAdmin admin@info.net
ServerName info.net
ServerAlias info.net
DocumentRoot /var/www/info.net/html
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell
```

Now use Ctrl+O to save and then Ctrl+X to exit the file.

Step-4 **Activate virtual host configuration file**

In this step, we will be creating the virtual host configuration file. For this, execute the following command in Terminal

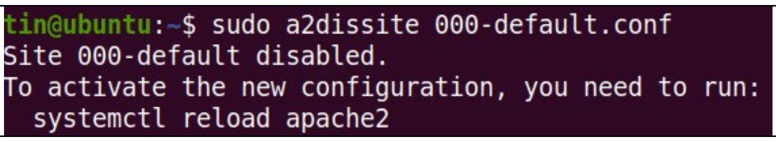
$ sudo a2ensite info.net.conf



```
tin@ubuntu:~$ sudo a2ensite info.net.conf
Enabling site info.net.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

Now disable the "000-default.conf" default virtual configuration file as follows:

$ sudo a2dissite 000-default.conf



```
tin@ubuntu:~$ sudo a2dissite 000-default.conf
Site 000-default disabled.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

Now restart Apache to activate the new configuration as follows

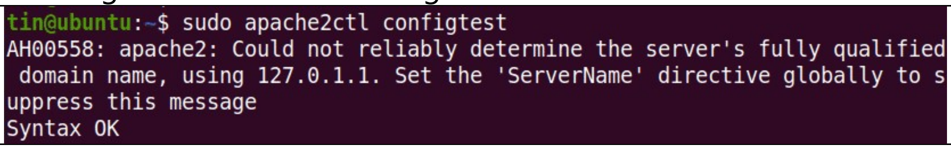
$ sudo systemctl restart apache2

Step-5 **Test for errors**

Once all the configurations are completed, you can test for any configuration errors:

$ sudo apache2ctl configtest

You might receive the following error



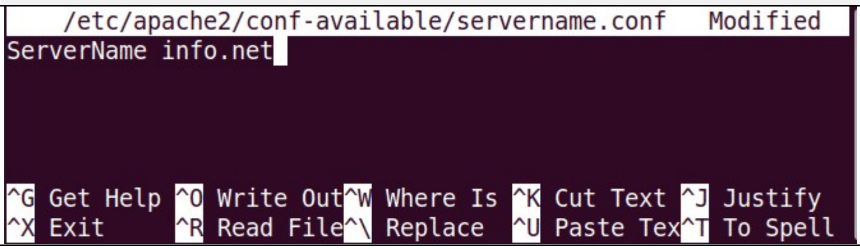
```
tin@ubuntu:~$ sudo apache2ctl configtest
AH00558: apache2: Could not reliably determine the server's fully qualified
 domain name, using 127.0.1.1. Set the 'ServerName' directive globally to s
uppress this message
Syntax OK
```

In order to resolve this error, edit the **servername.conf** file:

$ sudo nano /etc/apache2/conf-available/servername.conf

Then add this line by replacing the info.net with your own domain name

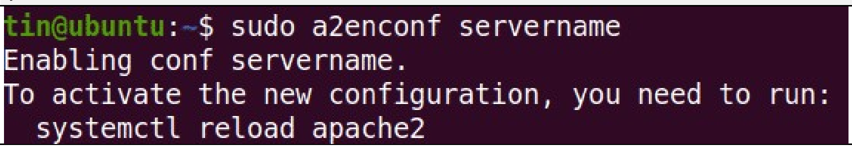
ServerName info.net



```
    /etc/apache2/conf-available/servername.conf    Modified
ServerName info.net




^G Get Help ^O Write Out^W Where Is ^K Cut Text ^J Justify
^X Exit     ^R Read File^\ Replace  ^U Paste Tex^T To Spell
```

Save and exit the **servername.conf** file and run:

$ sudo a2enconf servername

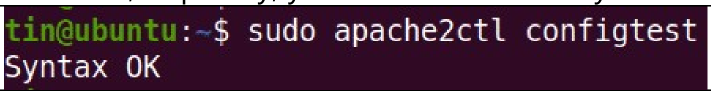


```
tin@ubuntu:~$ sudo a2enconf servername
Enabling conf servername.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

Now again execute

$ sudo apache2ctl configtest

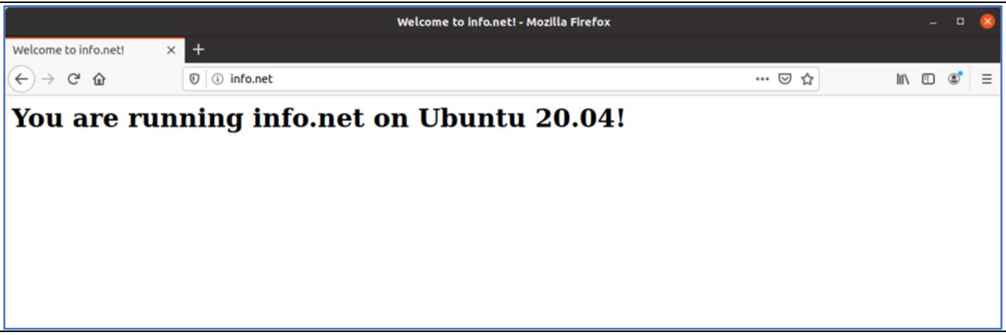
This time, hopefully, you will not receive any error.



```
tin@ubuntu:~$ sudo apache2ctl configtest
Syntax OK
```

Step-6 **Test virtual host**

Now the Apache web server is ready to serve our domain. Let's test this by navigating to the following link in the browser:

	http://info.net
	Replace the **info.net** with your domain name. The following index page shows the Apache server is ready to serve our domain name.
	 Welcome to info.net! - Mozilla Firefox info.net **You are running info.net on Ubuntu 20.04!**
	Managing Apache server
	In order to manage the Apache server, here are some of the useful commands that you can run in Terminal:
	To start the Apache server:
	$ sudo systemctl start apache2
	To stop the Apache server:
	$ sudo systemctl stop apache2
	To reload apache server to update the new configurations:
	$ sudo systemctl reload apache2
	To start Apache at boot:
	$ sudo systemctl enable apache2
Conclusion	This experiment successfully demonstrated the installation and configuration of the Apache web server on an Ubuntu 16.04 Linux system. Through the process, we were able to successfully set up a fully functioning Apache web server capable of serving content for a specific domain. The virtual host configuration allows for hosting multiple websites or web applications on the same physical server, demonstrating the flexibility and scalability of the Apache web server. Overall, this experiment provided hands-on experience with the installation, configuration, and management of a widely-used open-source web server solution.

Signature	
Grade	
Date	