Web Lab 1 Report: Apache Web Server Installation & Maintenance

Task 1: Setting up an Apache Web Server

Installation and Configuration

The Apache web server was installed on Ubuntu using the following commands:

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

After installation, the firewall was configured to allow HTTP traffic:

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

The Apache service was verified to be running:

```
sudo systemctl status apache2
```

Domain Configuration

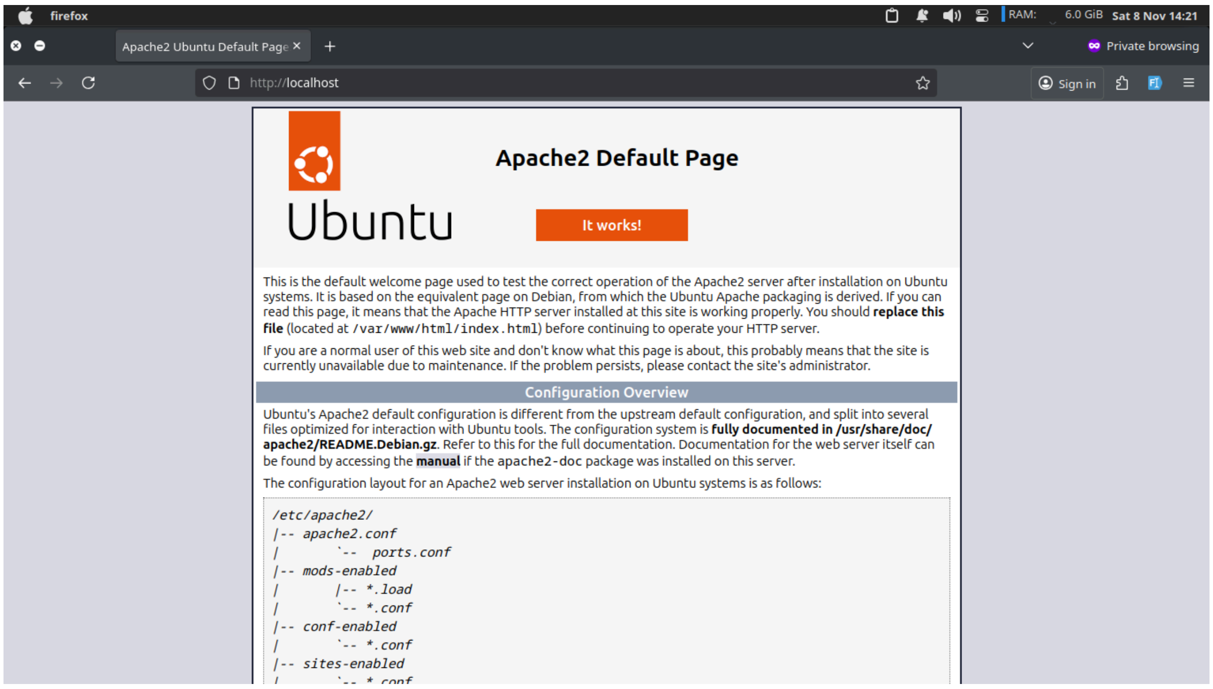
To facilitate testing, the domain name `webserverlab.com` was mapped to localhost by adding the following entry to `/etc/hosts`:

```
127.0.0.1 webserverlab.com
```

Checkpoint 1: Default Apache Landing Page (2 Marks)

The Apache web server was successfully installed and configured. The default Apache landing page was accessed using both `localhost` and `webserverlab.com` to verify the installation.

Screenshot 1: Accessing via localhost



Apache2 Default Page

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.

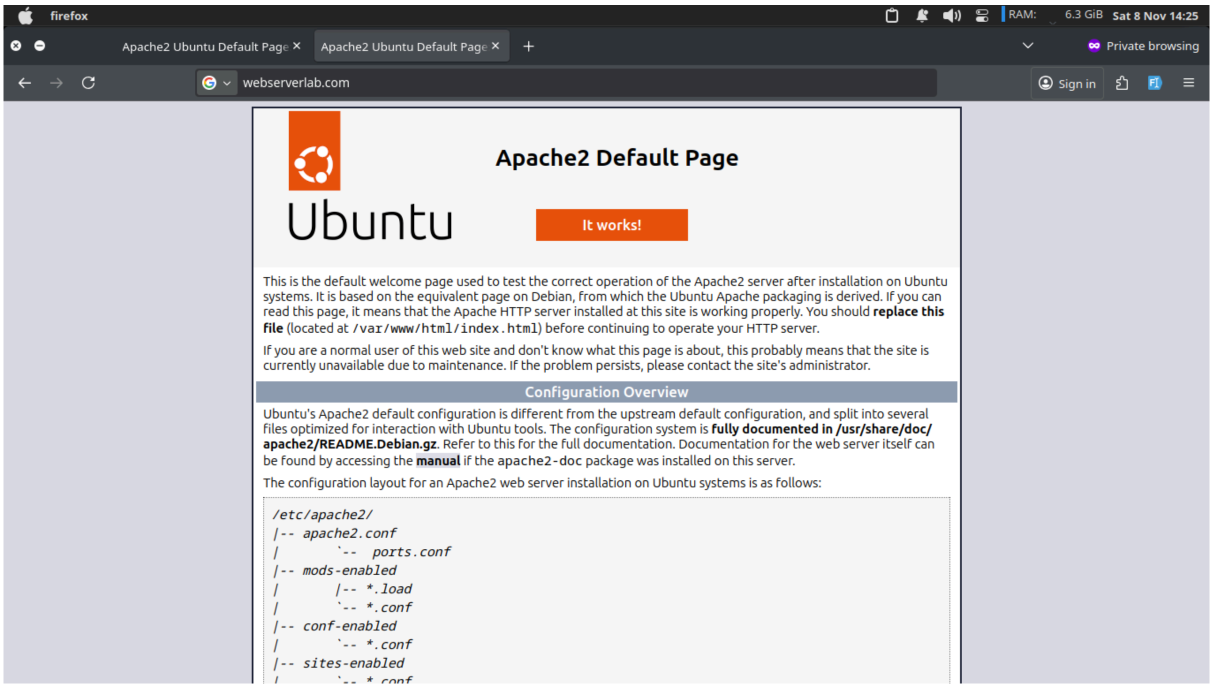
Configuration Overview

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
```

Screenshot 2: Accessing via webserverlab.com



Apache2 Default Page

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
```

Both methods successfully displayed the default Apache web page, confirming that Apache is working correctly.

Task 2: Setting up Virtual Hosts

Step 1: Managing Apache Process

The following commands were used to manage the Apache service:

- **Stop**: `sudo systemctl stop apache2`
- **Start**: `sudo systemctl start apache2`
- **Restart**: `sudo systemctl restart apache2`
- **Reload**: `sudo systemctl reload apache2`
- **Enable at boot**: `sudo systemctl enable apache2`

Step 2: Setting up a Single Virtual Host (example.com)

A virtual host was created for `example.com` following these steps:

1. **Created directory structure:**

```
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
```

2. **Created index.html file** at `/var/www/example.com/html/index.html`:

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

3. **Created virtual host configuration** at `/etc/apache2/sites-available/example.com.conf`:

```
<VirtualHost *:80>ServerAdmin admin@example.comServerName example.comServerAlias www.example.comDocumentRoot /var/www/example.com/htmlErrorLog ${APACHE_LOG_DIR}/error.logCustomLog ${APACHE_LOG_DIR}/access.log combined</VirtualHost>
```

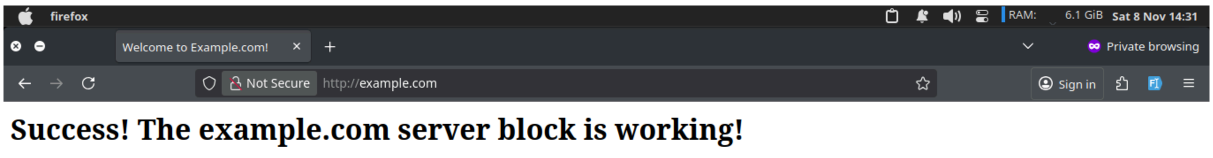
4. **Enabled the site and disabled default:**

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

Checkpoint 2: example.com Virtual Host (5 Marks)

The virtual host for `example.com` was successfully configured and tested. The site displays the custom content as expected.

Screenshot: example.com virtual host



Success! The example.com server block is working!

The virtual host is working correctly, serving content from `/var/www/example.com/html` instead of the default Apache directory.

Checkpoint 3: Understanding Virtual Host Behavior (4 Marks)

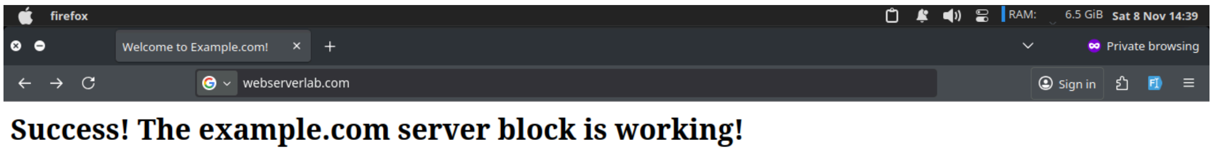
After enabling the `example.com` virtual host and disabling the default configuration, the following observations were made:

Behavior Analysis:

1. **http://webserverlab.com**: When accessing `webserverlab.com`, Apache serves the `example.com` virtual host because:
   - The default site (000-default.conf) was disabled
   - Apache uses the first available virtual host as the default when no matching ServerName or ServerAlias is found
   - Since `example.com` was the only enabled virtual host, it became the default

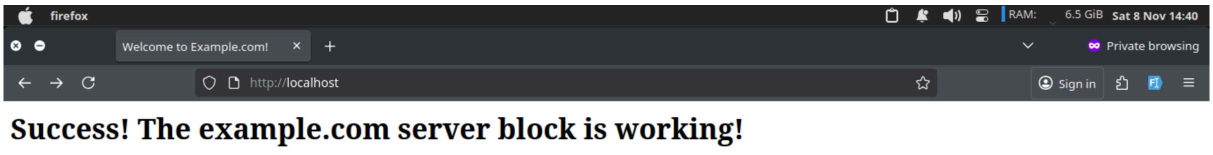
2. **http://127.0.0.1**: Accessing via IP address also serves the `example.com` virtual host for the same reason - it becomes the default when no specific match is found.

Screenshot 1: webserverlab.com behavior



Success! The example.com server block is working!

Screenshot 2: localhost behavior



Success! The example.com server block is working!

Explanation:
Apache's virtual host matching works by comparing the `Host` header in the HTTP request against the `ServerName` and `ServerAlias` directives. When no match is found, Apache uses the first virtual host in alphabetical order (or the first enabled one) as the default. This is why both `webserverlab.com` and `127.0.0.1` resolve to the `example.com` virtual host when the default site is disabled.

Step 3: Setting up Multiple Virtual Hosts

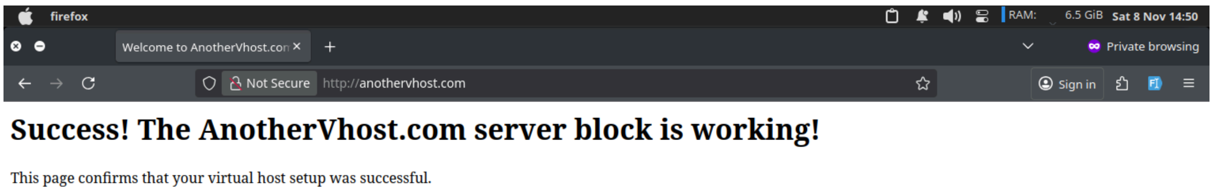
A second virtual host was created for `anothervhost.com` following the same procedure:

1. Created directory structure at `/var/www/anothervhost.com/html`
2. Created a custom `index.html` with different content
3. Created virtual host configuration at `/etc/apache2/sites-available/anothervhost.com.conf`
4. Enabled the site and restarted Apache

Checkpoint 4: Multiple Virtual Hosts (4 Marks)

The second virtual host `anothervhost.com` was successfully configured and tested. Both virtual hosts (`example.com` and `anothervhost.com`) are now working independently.

Screenshot: anothervhost.com virtual host


firefox
RAM: 6.5 GiB Sat 8 Nov 14:50
Welcome to AnotherVhost.com
Private browsing
Not Secure http://anothervhost.com
Sign in

Success! The AnotherVhost.com server block is working!

This page confirms that your virtual host setup was successful.

Both virtual hosts are functioning correctly, demonstrating the ability to host multiple websites on a single Apache server.

Task 3: Hosting Dynamic Websites using HTML and JavaScript

Implementation

Two dynamic websites were created using HTML and JavaScript, each hosted on separate virtual hosts. Each website includes HTML forms that accept user input and use JavaScript to handle form submissions and process the data dynamically.

Dynamic Website Features

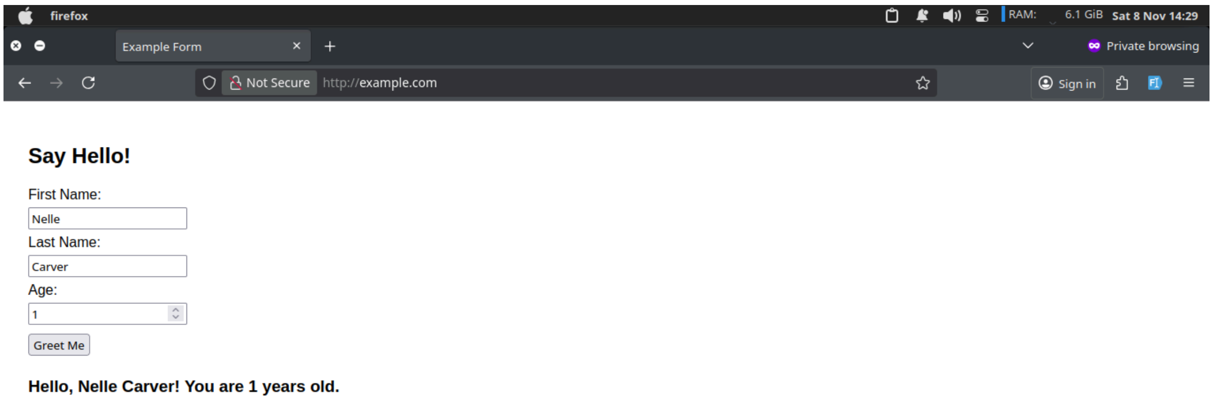
Both websites include:

- HTML forms with input fields (2-3 values)
- JavaScript event handlers for form submission
- Client-side processing and validation
- Dynamic content display based on user input

Checkpoint 5: Dynamic Websites Deployment (5 Marks)

Two dynamic websites were successfully deployed on separate virtual hosts, demonstrating the integration of HTML forms with JavaScript for interactive web functionality.

Screenshot 1: Dynamic website on example.com virtual host



Say Hello!

First Name: Nelle

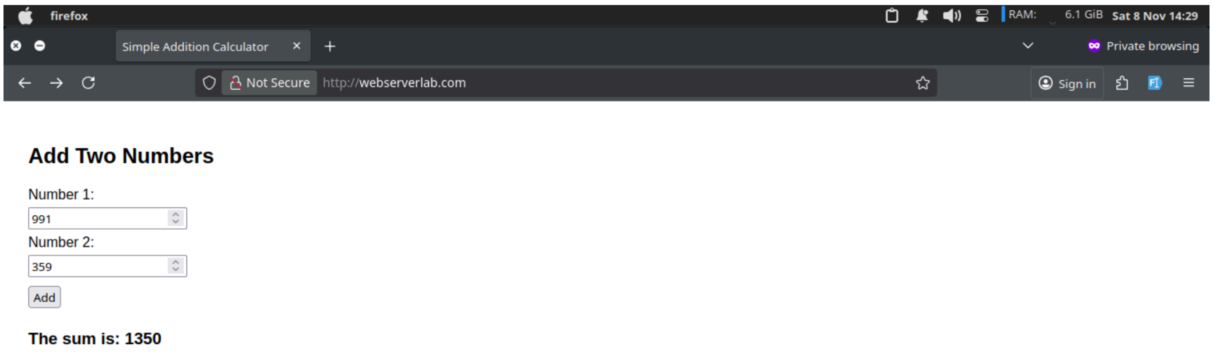
Last Name: Carver

Age: 1

Greet Me

Hello, Nelle Carver! You are 1 years old.

Screenshot 2: Dynamic website on webserverlab.com virtual host



Add Two Numbers

Number 1: 991

Number 2: 359

Add

The sum is: 1350

Both dynamic websites are fully functional, accepting user input through forms and processing the data using JavaScript to provide interactive user experiences.