

# Task-1: Apache Web Server Setup

1.Apache install (`sudo apt install apache2`)

2.Firewall configure (`sudo ufw allow 'Apache'`):

## What is a Firewall?

- A firewall is like a **security guard** at your computer's door
- It blocks all incoming connections by default
- Only specifically allowed services can be accessed from outside

## What `sudo ufw allow 'Apache'` does:

With this command, you're telling the firewall:

- Allow **Port 80** (HTTP traffic)
- So that external people can view your website through a browser

## In your case:

Your firewall is **inactive** (`Status: inactive`), so:

- **You won't have any problems**
- The firewall isn't imposing any restrictions
- All connections are automatically allowed

## For the lab:

Since the firewall is inactive, you **can skip this step for now**. But if you want to follow the lab instructions exactly, you can run:

```
bash
● sudo ufw allow 'Apache'
sudo ufw enable
```

**But remember:** If you do `ufw enable`, the firewall will become active and only allowed services will work.

**Recommendation:** Don't enable the firewall right now for the lab. You can do it later when you need external access.

## After this:

Inside File we have to add these after hit into bash (`sudo nano etc/hosts`):

```
127.0.0.1 localhost  
127.0.0.1 webserverlab.com  
127.0.0.1 example.com  
127.0.0.1 anothervhost.com
```

These lines create **fake domain names** that point to your own computer.

## What they do:

- When you type `webserverlab.com` in your browser, it redirects to `127.0.0.1` (your own computer)
- When you type `example.com` in your browser, it redirects to `127.0.0.1` (your own computer)
- Same for `anothervhost.com`

## Why we need this for the lab:

1. **Testing without internet:** You can test websites without buying real domains
2. **Multiple websites on one server:** You can host multiple "different" websites on your single Apache server
3. **Local development:** Everything works on your local machine without needing real web hosting

## Real life example:

It's like having multiple **business cards** for the same person:

- One card says "Web Developer - webserverlab.com"
- Another says "Example Company - example.com"
- But both point to the SAME person (your computer)

## How it works in the lab:

- You'll create separate folders for each domain
- Apache will show different websites for each domain
- Even though all domains point to the same computer, Apache knows which website to show based on the domain name

## Without these lines:

If you type `webserverlab.com` in browser, it would try to find the **real website** on the internet and show "Site not found" error.

## With these lines:

When you type `webserverlab.com`, your computer says "I know this one!" and shows your local Apache website.

Then save(ctrl+x -> y -> enter)

Try in browser:

- `http://localhost`
- `http://webserverlab.com`
- `http://127.0.0.1`

Now in browser we can see the page of apache web server

3. To show the page of apache hit Browser `http://webserverlab.com` or  
`http://localhost`

We can check it by `ping webserverlab.com`

## Task-2: Virtual Host Setup

### Step-1: Single Virtual Host

#### A. Directory Structure Creation:

```
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
```

#### B. HTML Content Creation:

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

#### B. Virtual host configuration file create (sudo nano/etc/apache2/sites-available/[example.com.conf](#))

#### C. Inside the file :

```
<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>
C.Save and exit
D.Enable the file (sudo a2ensite example.com.conf)
E.Default site disable (sudo a2dissite 000-default.conf)
F.Configuration test (Syntax OK)
G.Test in browser (http://example.com)
```

## Step-2: Default Site Behavior Testing

### Commands Executed:

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

### Observations:

- `http://example.com` → Showed custom `example.com` page
- `http://webserverlab.com` → Showed Apache default page
- `http://127.0.0.1` → Showed Apache default page

### Explanation:

- Only `example.com.conf` was enabled
- Other domains (`webserverlab.com`, `127.0.0.1`) had no specific virtual host
- Apache fell back to default virtual host behavior
- Demonstrates Apache's domain-based virtual host routing

আমার case-এ `example.com` virtual host টি first enabled site হওয়ায় সেটা default virtual host হয়ে গেছে। যে কোনো unmatched domain request automatically `example.com`-এ redirect হচ্ছে। এটা fix করতে আমাদের প্রতিটি domain-এর জন্য explicit virtual host configuration করতে হবে।"

## Step-3 - Multiple Virtual Hosts Configuration

---

### Objective

To configure multiple virtual hosts on a single Apache web server, demonstrating the ability to host multiple independent websites with separate content and domains.

---

### Steps Performed

#### 1. Directory Structure Creation

```
bash
```

```
# Created directory structure for second virtual host
```

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

### **# Set proper ownership**

```
sudo chown -R $USER:$USER /var/www/anothervhost.com/html
```

### **# Set appropriate permissions**

```
sudo chmod -R 755 /var/www/anothervhost.com
```

## **2. HTML Content Creation**

**Created `/var/www/anothervhost.com/html/index.html` with unique content:**

### **File Content:**

```
html
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <title>Welcome to AnotherVHost.com!</title>
```

```
  <style>
```

```
    body {
```

```
      font-family: Arial, sans-serif;
```

```
      background-color: #e6f7ff;
```

```
      margin: 50px;
```

```
}

.container {
    background: white;
    padding: 30px;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

h1 {
    color: #ff6600;
}

</style>

</head>

<body>

<div class="container">

<h1> Success! AnotherVHost.com is Working!</h1>

<p>This is the <strong>second virtual host</strong> on our Apache server.</p>

<p>We are successfully hosting multiple websites on the same server!</p>

<hr>

<p><strong>Domain:</strong> anothervhost.com</p>

<p><strong>Server:</strong> Apache Virtual Host</p>

<p><strong>Status:</strong>  Active and Running</p>

</div>
```

```
</body>
```

```
</html>
```

### 3. Virtual Host Configuration

Created `/etc/apache2/sites-available/anothervhost.com.conf`:

Configuration File Content:

```
apache
```

```
<VirtualHost *:80>
```

```
    ServerAdmin admin@anothervhost.com
```

```
    ServerName anothervhost.com
```

```
    ServerAlias www.anothervhost.com
```

```
    DocumentRoot /var/www/anothervhost.com/html
```

```
    ErrorLog ${APACHE_LOG_DIR}/anothervhost_error.log
```

```
    CustomLog ${APACHE_LOG_DIR}/anothervhost_access.log combined
```

```
</VirtualHost>
```

### 4. Virtual Host Activation

**bash**

**# Enable the new virtual host**

```
sudo a2ensite anothervhost.com.conf
```

**# Restart Apache to apply changes**

```
sudo systemctl restart apache2
```

### 5. Domain Mapping

Updated `/etc/hosts` file with:

text

127.0.0.1 localhost

127.0.0.1 webserverlab.com

127.0.0.1 example.com

127.0.0.1 anothervhost.com

---

## Testing & Verification

Test 1: Individual Domain Access

- <http://example.com> ✓ - Displayed custom [example.com](http://example.com) page
- <http://anothervhost.com> ✓ - Displayed custom [anothervhost.com](http://anothervhost.com) page

## Test 2: Simultaneous Operation

Both websites running concurrently on the same Apache instance:

- Different designs and styling
- Separate content and branding
- Independent domain names
- Shared server resources

## Test 3: Configuration Validation

### bash

```
# Verified configuration syntax
```

```
sudo apache2ctl configtest
```

```
# Output: Syntax OK
```

## # Checked enabled sites

```
ls -la /etc/apache2/sites-enabled/
```

```
# Output: example.com.conf & anothervhost.com.conf both present
```

---

## Technical Details

Directory Structure:

text

/var/www/

```
└── example.com/
    ├── html/
    │   └── index.html
    └── anothervhost.com/
        └── html/
            └── index.html
```

Configuration Files:

- `/etc/apache2/sites-available/example.com.conf`
- `/etc/apache2/sites-available/anothervhost.com.conf`

Enabled Sites:

`example.com.conf` → Symbolic link in sites-enabled

`anothervhost.com.conf` → Symbolic link in sites-enabled

## Task-3 - Dynamic Website Deployment with HTML & JavaScript

## Objective

To deploy two dynamic websites using HTML forms and JavaScript for client-side form handling and data processing, demonstrating interactive web applications without server-side programming.

---

## Implementation Details

Website 1: User Registration System ([example.com](http://example.com))

Location:

`/var/www/example.com/html/index.html`

### Features Implemented:

1. Comprehensive Registration Form with 5 input fields:
  - Full Name (Text input)
  - Email Address (Email validation)
  - Age (Number input with range 18-100)
  - Country (Dropdown selection)
  - Phone Number (Telephone input)
2. JavaScript Functionality:
  - Form submission handling with `event.preventDefault()`
  - Input validation
  - Dynamic result display
  - Automatic form reset
  - Alert notifications
3. CSS Styling:
  - Gradient background
  - Modern card layout
  - Responsive design
  - Hover effects and animations

Code Structure:

html

```
<form id="registrationForm">
```

```
    <!-- Form fields -->
```

```
</form>
```

```
<script>

document.getElementById('registrationForm').addEventListener('submit', function(event) {

    // Form handling logic

    // Data processing

    // Result display

});

</script>
```

---

## **Website 2: Advanced Calculator ([anothervhost.com](http://anothervhost.com))**

### **Location:**

/var/www/anothervhost.com/html/index.html

### **Features Implemented:**

#### **1. Mathematical Operations:**

- Addition
- Subtraction
- Multiplication
- Division
- Modulus
- Power Calculation
- Square Root

#### **2. JavaScript Functionality:**

- Multiple calculation functions
- Input validation
- Dynamic result generation
- Error handling (division by zero)
- Keyboard support (Enter key)

#### **3. CSS Styling:**

- Orange gradient theme
- Grid layout for inputs
- Card-based result display

- Interactive animations

#### **Code Structure:**

html

```
<div class="calculator-grid">  
    <!-- Input fields -->  
</div>  
  
<button onclick="performAllCalculations()">Calculate</button>
```

```
<script>
```

```
function performAllCalculations() {  
    // Get input values  
    // Perform calculations  
    // Display results  
}
```

```
</script>
```

---

#### Technical Implementation

HTML Forms Used:

html

```
<!-- Text Input -->  
<input type="text" id="fullName" required>  
  
<!-- Email Input -->
```

```
<input type="email" id="email" required>

<!-- Number Input -->

<input type="number" id="age" min="18" max="100" required>
```

```
<!-- Dropdown Selection -->

<select id="country" required>

  <option value="bd">Bangladesh</option>

</select>
```

```
<!-- Telephone Input -->

<input type="tel" id="phone" required>
```

JavaScript Event Handling:

```
javascript

// Event Listener Approach

document.getElementById('registrationForm').addEventListener('submit',
function(event) {

  event.preventDefault();

  // Process form data

});
```

```
// Inline Event Handler Approach

<button onclick="performAllCalculations()">Calculate</button>
```

**DOM Manipulation:**

```
javascript

// Showing results dynamically

resultDiv.innerHTML = `

<h3>🎉 Registration Successful!</h3>

<p><strong>Name:</strong> ${fullName}</p>

`;

// Display control

resultSection.style.display = 'block';
```

---

## Testing & Verification

Test 1: User Registration System ([example.com](http://example.com))

### Steps:

1. Accessed <http://example.com>
2. Filled all form fields:
  - Name: "John Doe"
  - Email: "john@example.com"
  - Age: 25
  - Country: Bangladesh
  - Phone: "+8801712345678"

### Results:

- JavaScript Alert showed registration details
- Dynamic Result Display on page with all information
- Form Validation worked for empty fields
- Automatic Reset after 5 seconds

Test 2: Advanced Calculator ([anothervhhost.com](http://anothervhhost.com))

### Steps:

1. Accessed <http://anothervhost.com>
2. Entered numbers: 15 and 3
3. Clicked "Calculate All Operations"

Results:

 All Mathematical Operations calculated correctly:

- Addition: 18
- Subtraction: 12
- Multiplication: 45
- Division: 5
- Modulus: 0
- Power: 3375
- Square Root: 4.2426

 Results Displayed in organized layout

 Error Handling for division by zero

 Keyboard Support with Enter key

---

## Key Features Demonstrated

### 1. Client-Side Processing

- No server-side programming required
- All processing done in browser using JavaScript
- Immediate feedback to users

### 2. Form Handling

- Input validation
- Data collection and processing
- User-friendly error messages

### 3. Dynamic Content

- Real-time result updates
- DOM manipulation without page reload
- Interactive user experience

### 4. Cross-Browser Compatibility

- Standard HTML5 form elements
- Vanilla JavaScript (no frameworks)
- Responsive CSS design