

Documentation

Deploy a basic HTML/CSS Website on AWS EC2 Using Nginx

To deploy a basic HTML/CSS website on AWS EC2 using Nginx, follow the steps below:

Prerequisites:

- An AWS account.
- Basic knowledge of EC2, SSH, Nginx, and HTML/CSS.
- EC2 instance running a Linux distribution (e.g., Amazon Linux, Ubuntu).
- Security group configured for HTTP (port 80) and SSH (port 22).

Step 1: Launch an EC2 Instance

1. Log in to AWS Console:

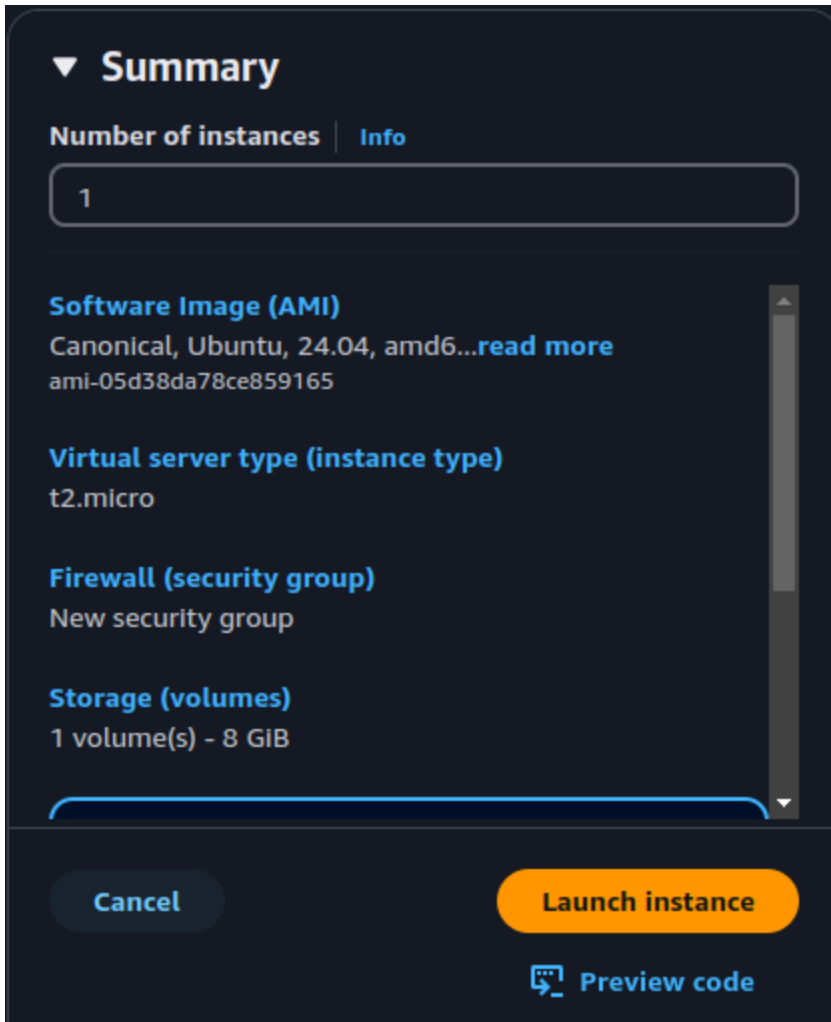
- Go to the [AWS EC2 Dashboard](#).

2. Launch an EC2 Instance:

- Choose an **Amazon Machine Image (AMI)**: Select a Linux-based AMI (e.g., Amazon Linux 2 or Ubuntu).
- **Instance Type**: Choose a basic instance type (e.g., `t2.micro`).
- **Configure Instance**: Leave the default settings, but ensure you select a **public subnet**.
- **Add Storage**: Leave as default.
- **Configure Security Group**: Create a new security group allowing:
 - **SSH (port 22)** from your IP.
 - **HTTP (port 80)** from anywhere (0.0.0.0/0).
- **Key Pair**: Create and download a key pair (`.pem` file) to access the instance via SSH.

3. Launch the Instance:

- Click **Launch** and note the **Public IP Address** of the instance once it's running.



The screenshot shows the 'Summary' tab of the AWS 'Launch instance' wizard. It features a dark blue header with a white 'Summary' title and a downward arrow. Below the header, there are four sections: 'Number of instances' with a value of '1' and an 'Info' link; 'Software Image (AMI)' showing 'Canonical, Ubuntu, 24.04, amd64...read more' and the ID 'ami-05d38da78ce859165'; 'Virtual server type (instance type)' set to 't2.micro'; and 'Firewall (security group)' set to 'New security group'. A 'Storage (volumes)' section shows '1 volume(s) - 8 GiB'. At the bottom, there are two buttons: a grey 'Cancel' button and an orange 'Launch instance' button. Below the 'Launch instance' button is a 'Preview code' link with a terminal icon.

▼ **Summary**

Number of instances | Info

1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd64...read more
ami-05d38da78ce859165

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Cancel Launch instance

Preview code

Step 2: Connect to Your EC2 Instance

1. SSH into Your EC2 Instance:

- Open a terminal on your local machine.
- Use the command below, replacing the path to your `.pem` file and EC2 public IP address:


```
sudo ssh -i /path/to/your-key.pem ec2-user@<EC2-PUBLIC-IP>
```



- If using Ubuntu, replace `ec2-user` with `ubuntu`.


Connect to instance Info


Connect to your Instance `i-0b0bf82057f950086` (My server) using any of these options

EC2 Instance Connect Session Manager **SSH client** EC2 serial console

Instance ID
 `i-0b0bf82057f950086` (My server)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this Instance is `project 4 key.pem`.
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 `chmod 400 "project 4 key.pem"`
4. Connect to your Instance using its Public DNS:
 `ec2-54-71-111-249.us-west-2.compute.amazonaws.com`

Example:
 `ssh -i "project 4 key.pem" ubuntu@ec2-54-71-111-249.us-west-2.compute.amazonaws.com`

 **Note:** In most cases, the guessed username is correct. However, read your AMI usage Instructions to check if the AMI owner has changed the default AMI username.

```

/Downloads$ ssh -i "project 4 key.pem" ubuntu@ec2-54-71-111-249.us-west-2.compute.amazonaws.com

```

```

System load:  0.02          Processes:           106
Usage of /:   24.7% of 6.71GB Users logged in:       0
Memory usage: 22%          IPv4 address for enX0: 172.31.17.231
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-17-231:~$

```

Step 3: Install Nginx

sudo

1. Install Nginx on EC2:

- Run the following command to install Nginx (for Amazon Linux 2, use `yum`; for Ubuntu, use `apt`):
 - For **Amazon Linux 2**:

```
sudo yum update -y
sudo yum install nginx -y
```

- For **Ubuntu**:

```
sudo apt update
sudo apt install nginx -y
```

```
ubuntu@ip-172-31-17-231:~$ sudo apt update
sudo apt install nginx -y
Hit:1 http://us-west-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-west-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-west-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
```

2. Start Nginx:

- Start and enable Nginx to run on boot:
 - For **Amazon Linux 2**:

```
sudo systemctl start nginx
sudo systemctl enable nginx
```

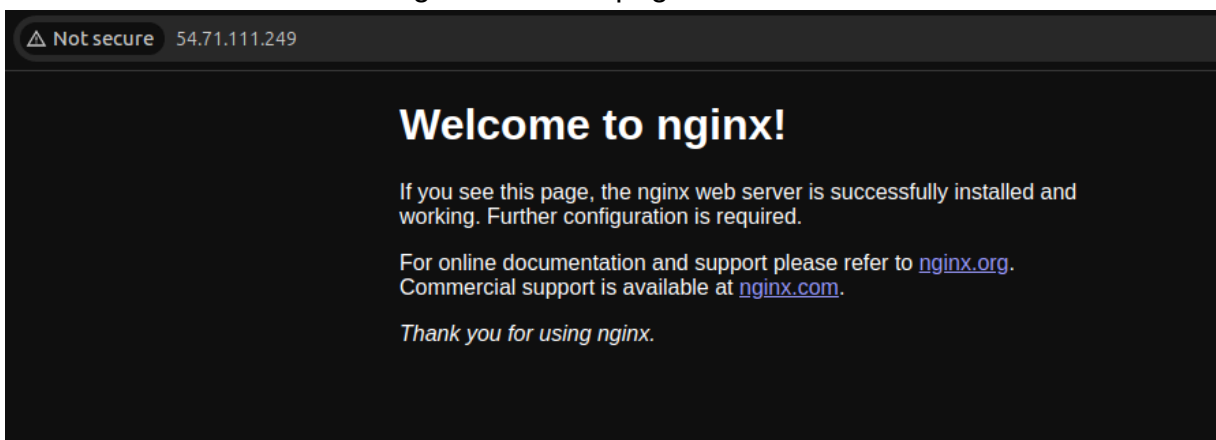
- For **Ubuntu**:

```
sudo systemctl start nginx
sudo systemctl enable nginx
```

```
ubuntu@ip-172-31-17-231:~$ sudo systemctl start nginx
ubuntu@ip-172-31-17-231:~$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-install
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
ubuntu@ip-172-31-17-231:~$
```

3. Verify Nginx is Running:

- Open a web browser and enter the EC2 public IP address.
- You should see the default Nginx welcome page.



Step 4: Set Up Your Website

1. Create Your HTML/CSS Website:

- On your EC2 instance, create a directory for your website files:

```
sudo mkdir -p /var/www/html/mywebsite
```

```
ubuntu@ip-172-31-17-231:~$ sudo mkdir -p /var/www/html/mywebsite
ubuntu@ip-172-31-17-231:~$
```

2. Upload HTML/CSS Files:

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

Create index.html file and paste html code in it.

```
sudo vi index.html
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My Simple Website</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header>
    <h1>Welcome to My Simple Website</h1>
    <nav>
      <ul>
        <li><a href="#">Home</a></li>
        <li><a href="#">About</a></li>
        <li><a href="#">Contact</a></li>
      </ul>
    </nav>
  </header>

  <section>
    <h2>About Us</h2>
    <p>This is a simple website built with HTML and CSS.</p>
  </section>
```

```
    <footer>
      <p>&copy; 2024 My Simple Website</p>
    </footer>
  </body>
</html>
```

Now create styles.css file and paste CSS code in it.

```
/* Basic reset */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

/* Body styling */
body {
  font-family: Arial, sans-serif;
  background-color: #f4f4f4;
  color: #333;
  line-height: 1.6;
}

/* Header styling */
header {
  background-color: #333;
  color: #fff;
  padding: 20px;
  text-align: center;
}

header h1 {
  font-size: 2.5em;
}

nav ul {
  list-style: none;
  margin-top: 10px;
}

nav ul li {
  display: inline;
```

```
        margin-right: 15px;
    }

    nav ul li a {
        color: #fff;
        text-decoration: none;
        font-size: 1.2em;
    }

    nav ul li a:hover {
        text-decoration: underline;
    }

    /* Section styling */
    section {
        padding: 20px;
        background-color: #fff;
        margin: 20px auto;
        width: 80%;
        max-width: 800px;
    }

    section h2 {
        font-size: 2em;
        margin-bottom: 10px;
    }

    section p {
        font-size: 1.2em;
    }

    /* Footer styling */
    footer {
        text-align: center;
        padding: 10px;
        background-color: #333;
        color: #fff;
        position: fixed;
        width: 100%;
        bottom: 0;
    }
```

3. Update Nginx Configuration:

- Edit the Nginx default server block to point to your new website directory:

```
sudo nano /etc/nginx/sites-available
```

```
ubuntu@ip-172-31-17-231:/etc/nginx/sites-available$ sudo vi default
```

- Find the following block and modify it to reflect your website directory:

```
server {  
    listen      80;  
    server_name localhost;  
  
    location / {  
        root    /var/www/html/mywebsite;  
        index   index.html;  
    }  
  
    # Other configuration (log files, etc.)  
}
```

```
#  
# Self signed certs generated by the ssl-cert package  
# Don't use them in a production server!  
#  
# include snippets/snakeoil.conf;  
  
root /var/www/html/mywebsite;  
  
# Add index.php to the list if you are using PHP  
index index.html index.htm index.nginx-debian.html;  
  
server_name _;  
  
location / {
```

4. Test Nginx Configuration:

- Before restarting Nginx, make sure the configuration file is valid:

```
sudo nginx -t
```

```
ubuntu@ip-172-31-17-231:/etc/nginx/sites-available$ sudo nginx -t  
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

- If everything is okay, restart Nginx:

```
sudo systemctl restart nginx
```

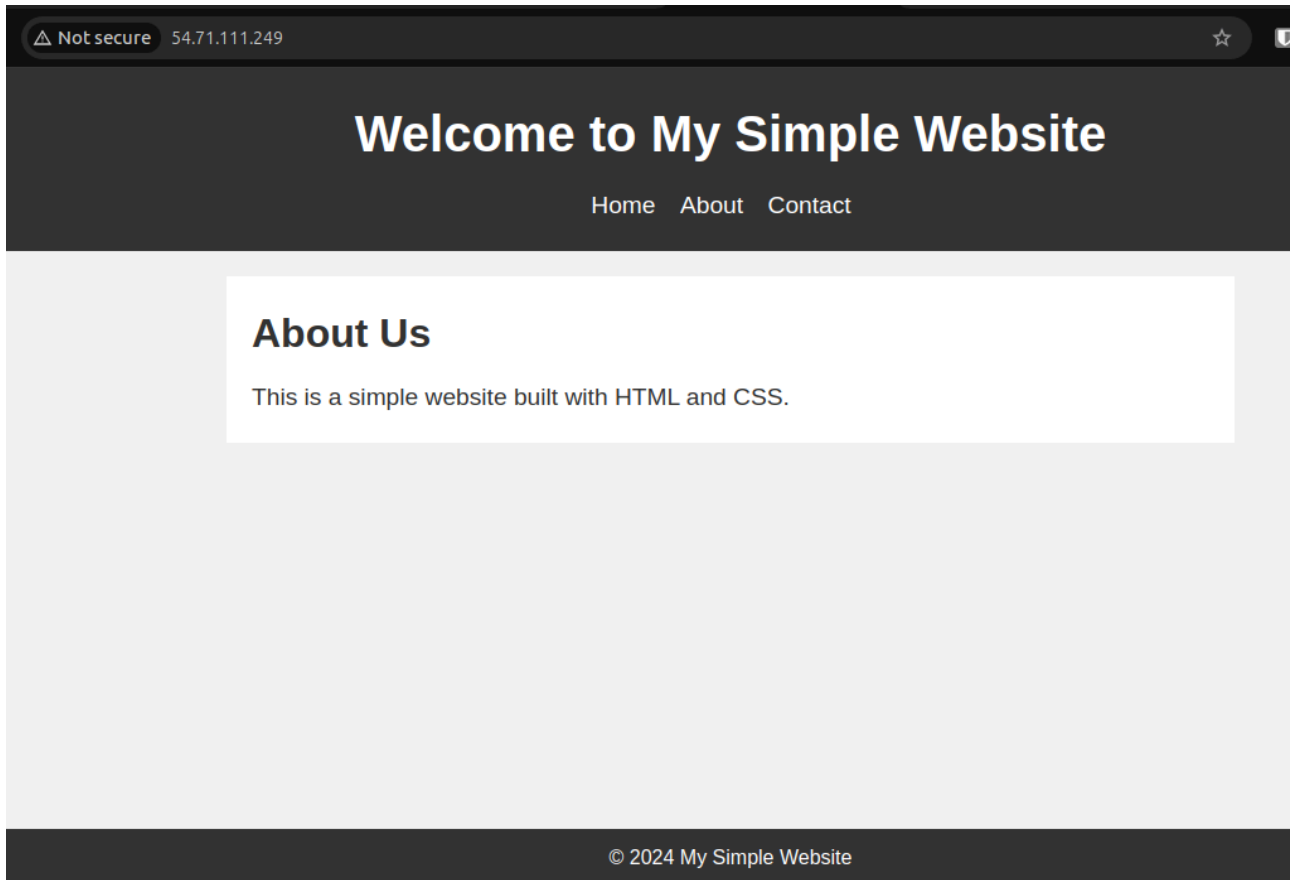
```
ubuntu@ip-172-31-17-231:/etc/nginx/sites-enabled$ sudo systemctl restart nginx
```


Step 5: Access Your Website

- Open a web browser and enter your EC2 instance's public IP address:

```
http://<EC2-PUBLIC-IP>
```

- You should now see your HTML/CSS website.



Step 6: (Optional) Set Up a Domain Name

To make your site more professional, you can configure a domain name to point to your EC2 instance. This involves:

1. Purchasing a domain name from a registrar (e.g., GoDaddy, Route 53).
2. Updating the DNS records to point to the EC2 instance's public IP address.

Step 7: Secure Your Website with HTTPS (Optional)

To enable HTTPS, you can set up SSL certificates using **Let's Encrypt**:

1. Install Certbot (the tool for obtaining SSL certificates):
 - For Amazon Linux 2:

```
sudo yum install -y certbot
```

- For Ubuntu:

```
sudo apt install certbot python3-certbot-nginx
```

2. Obtain an SSL certificate:

```
sudo certbot --nginx -d yourdomain.com
```

3. Certbot will automatically configure Nginx to use HTTPS and redirect HTTP to HTTPS.

Conclusion

Your basic HTML/CSS website is now deployed on AWS EC2 using Nginx. You can expand the website with more pages, JavaScript, and interactivity as needed. Make sure to monitor your instance, apply security patches, and consider further optimizations for scalability and performance.