Assignment-7

Uploading a Static Website on an Amazon EC2 Server

Overview

This guide explains:

1. **Creating an EC2 instance**
2. **Connecting to the server using Bitvise SSH Client**
3. **Installing and configuring the NGINX web server**
4. **Uploading and hosting a static website on EC2**

Remark: This method allows you to run a static website on a cloud-based server without needing external hosting services.

Step 1: Create an EC2 Instance

1. Sign in to the **AWS Management Console** and open the **EC2 console**.
2. Click **Instances (Running) → Launch Instance**.
3. In the **Launch an Instance** page:
    - Under **Name and Tags**, enter a descriptive name (e.g., `Snehaec2WebServer`).
    - Under **Application and OS Images (Amazon Machine Image)**:
        - Choose **Ubuntu** from the **Quick Start** options (Free Tier Eligible).
    - Under **Instance Type**, select **t2.micro** (Free Tier Eligible).
4. **Configure Key Pair (Login)**:
    - Select an existing key pair or create a new one.
    - If creating a new key pair:
        - Enter a **Key Pair Name** (e.g., `snehaa1234`).
        - Select **RSA** key type and **.pem** format.
        - Click **Create Key Pair**, and the `.pem` file will be downloaded automatically. **Save this file securely.**
5. **Configure Security Settings**:
    - Under **Firewall (Security Groups)**, check the following:
      ✅ **Allow SSH traffic** (To connect to the instance)
      ✅ **Allow HTTPS traffic** (For secure browsing)
      ✅ **Allow HTTP traffic** (To host the website)
6. Review the instance configuration and click **Launch Instance**.
7. After a few minutes, go to **View All Instances** to see the running instance.

Remark: Ensure you have the **.pem key file** downloaded; it is required for SSH access.

Step 2: Connect to EC2 Instance Using Bitvise SSH Client

1. Select your running instance and copy the **Public IPv4 Address**.
2. Download and install **Bitvise SSH Client** from your browser.
3. Open **Bitvise SSH Client** and enter:
    - **Host**: Paste the **Public IPv4 Address**.
    - **Username**: `ubuntu`
    - **Initial Authentication Method**: Select **public key**.
4. Click on **Client Key Manager → Import**.
    - Choose the **.pem key file** downloaded earlier.
    - Click **Open → Import**.
    - The key appears as **Global 1**.
5. Click **Login → Accept & Save**.
6. Open a **New Terminal Console** in Bitvise and run the following command to update packages:
7. `sudo apt-get update && sudo apt-get upgrade`
    - When prompted, type `y` and press **Enter**.

Step 3: Install and Configure NGINX Web Server

1. In the **New Terminal Console**, install NGINX:

   ```
   sudo apt-get install nginx
   ```

   When prompted, type `y` and press **Enter**.

2. Verify the installation:

   ```
   nginx -v
   ```

    - You should see an output like `nginx version: nginx/1.18.0 (Ubuntu).`

Step 4: Upload Website Files

1. In **Bitvise SSH Client**, open a **New SFTP Window**.
2. The window has two sections:
    - **Local Files** (Your computer)
    - **Remote Files** (EC2 instance)
3. Navigate to your local folder containing the static website files (`index.html`, `about.html`, `next.html`).
4. On the **Remote Files** side, follow these steps:
    - Click the folder icon **until you reach the root directory (`/`)**.
    - Navigate to:
    - `/var/www/html`
5. **Adjust File Permissions** (If needed):

```
sudo chmod 777 /var/www/html
```

    - This allows file uploads.
6. **Drag and Drop the Files** (`index.html`, `about.html`, `next.html`) from **Local Files** to **Remote Files** inside `/var/www/html`.

Remark: If permission errors occur, re-run the `chmod` command and try uploading again.

Step 5: Access the Website

1. Open a web browser.
2. Paste the **Public IPv4 Address** in the address bar.
3. Press **Enter**.
4. If configured correctly, your **index.html** page should load, confirming the website is live.

Final Notes

- If the website does not load, ensure:
    - The **NGINX server** is installed and running (`sudo systemctl status nginx`).
    - The **firewall settings** allow HTTP and HTTPS traffic.
    - The **file permissions** are correctly set for `/var/www/html`.
- The **bucket name must be unique** across AWS.
- For enhanced security, use **IAM roles and access policies** to restrict access.
