ReviewHub – Cloud Server Project Documentation

Student Name: Bijayan Ranjit

Student ID: 35442024

Unit: ICT171 – Introduction to Server Environments and Architectures

Project: ReviewHub – Personal Book Review Website

Global IP Address: 134.199.160.1

Domain Name: https://bijayan.site

GitHub Repository: https://github.com/BJN10/ReviewHub

Video Explainer: https://youtu.be/your-video-link

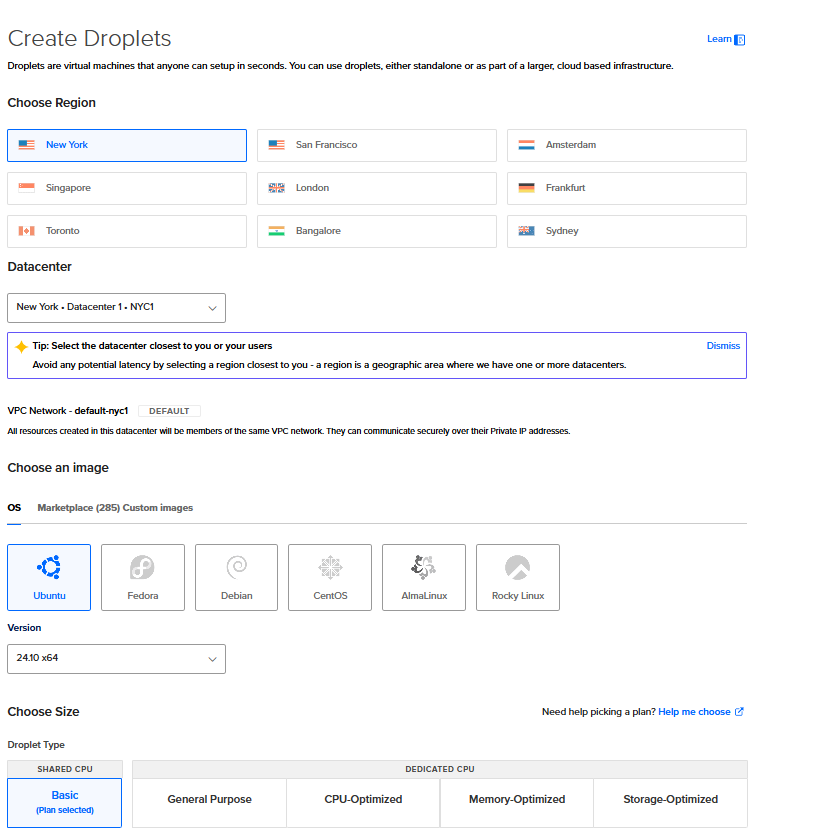
# Project Overview

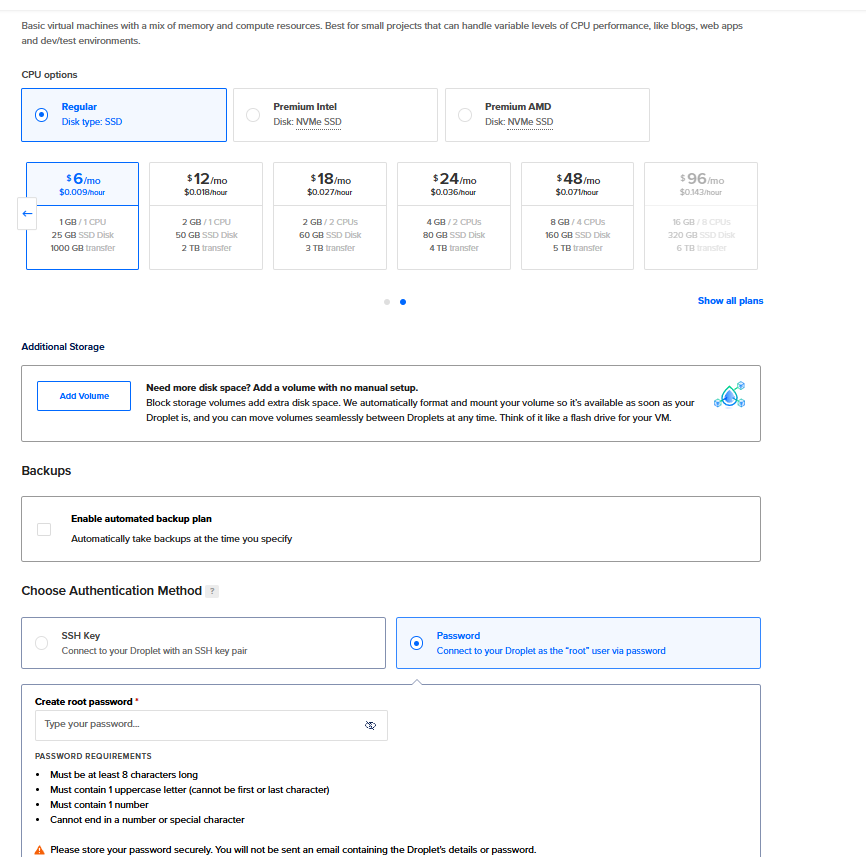
ReviewHub is a personal book review website hosted on a cloud-based Ubuntu server. This project demonstrates my ability to configure, deploy, secure, and maintain a web server manually using Infrastructure as a Service (IaaS). The website features a clean, minimalist layout using HTML, CSS, and JavaScript, with a black background and white text for accessibility.  
  
The site includes a homepage with popular reviews, a search bar for filtering content, an about page, and a review request submission page. Book reviews cover genres such as fiction, non-fiction, cybersecurity, and self-help.  
  
The server is deployed on DigitalOcean, with Nginx as the web server and secured using Certbot SSL/TLS certificates. All DNS settings were configured to point to a custom domain purchased separately. A Bash script was also written to monitor the server's uptime and status. All files, including the website, documentation, and script, are version-controlled and publicly available on GitHub.  
  
This project not only meets the technical goals of ICT171 but also connects a personal passion—reading—with practical cloud skills. It has been developed in a way that it can be maintained or extended well beyond this unit.

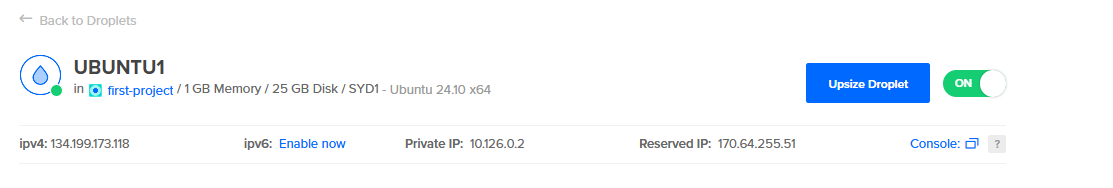
### Server Setup Steps (Ubuntu + Nginx)

**Step 1: Create a Droplet on DigitalOcean**  
Choose Ubuntu 22.04 LTS, allocate minimum 1GB RAM, and generate SSH keys for secure access.

Screenshot: DigitalOcean Droplet setup interface.







**Step 2: Update the System**

sudo apt update && sudo apt upgrade -y

Brief: Ensures the system has the latest security patches.

**Step 3: Install Nginx**

sudo apt install nginx -y

Brief: Nginx will serve your HTML website files.

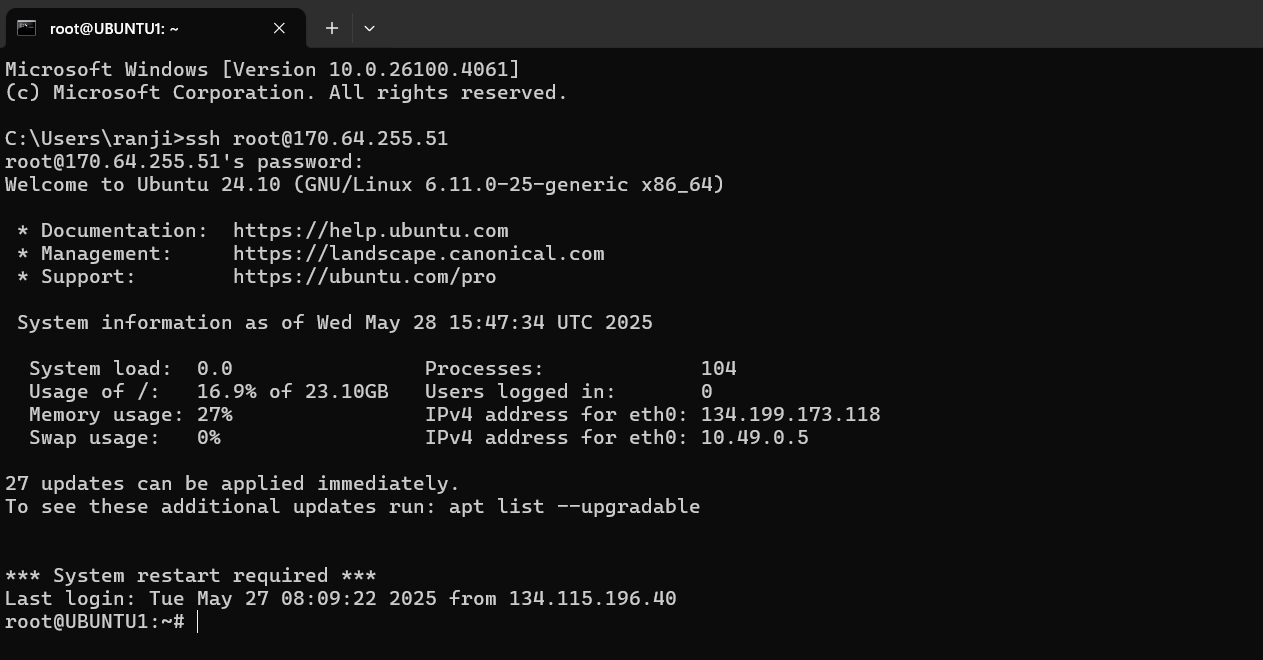
**Step 4: Enable Nginx and Allow Firewall**

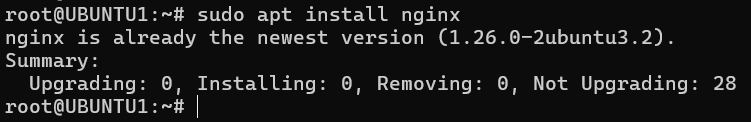
sudo ufw allow 'Nginx Full'

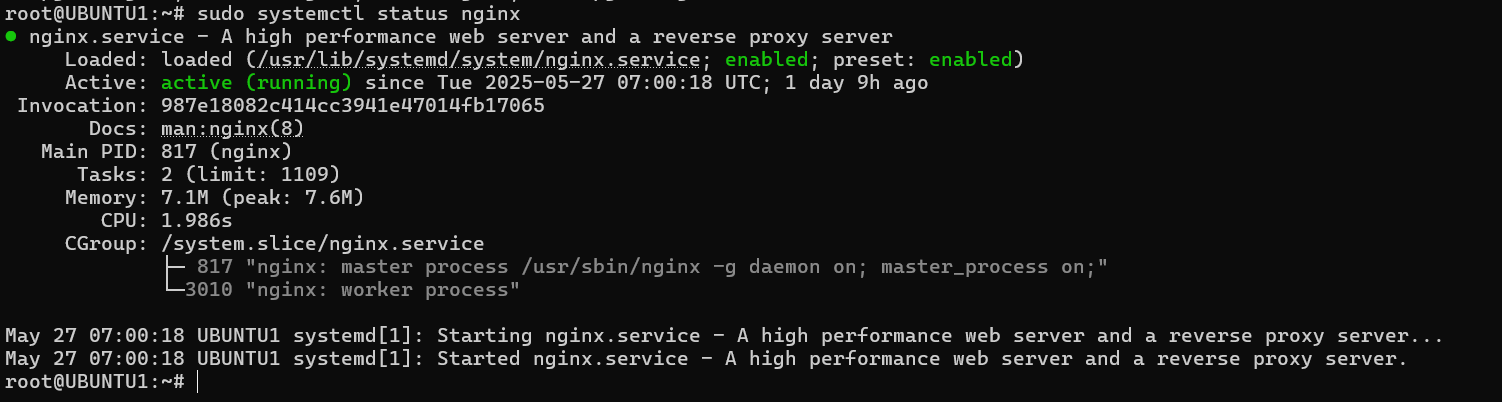
sudo systemctl enable nginx

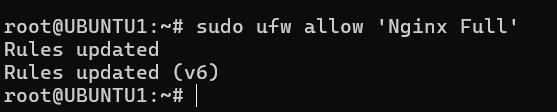
sudo systemctl start nginx

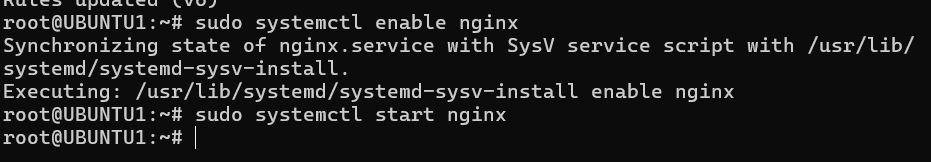
function: Opens required ports and ensures Nginx runs on boot.

*Screenshot: Terminal showing Nginx installation and firewall commands.* 









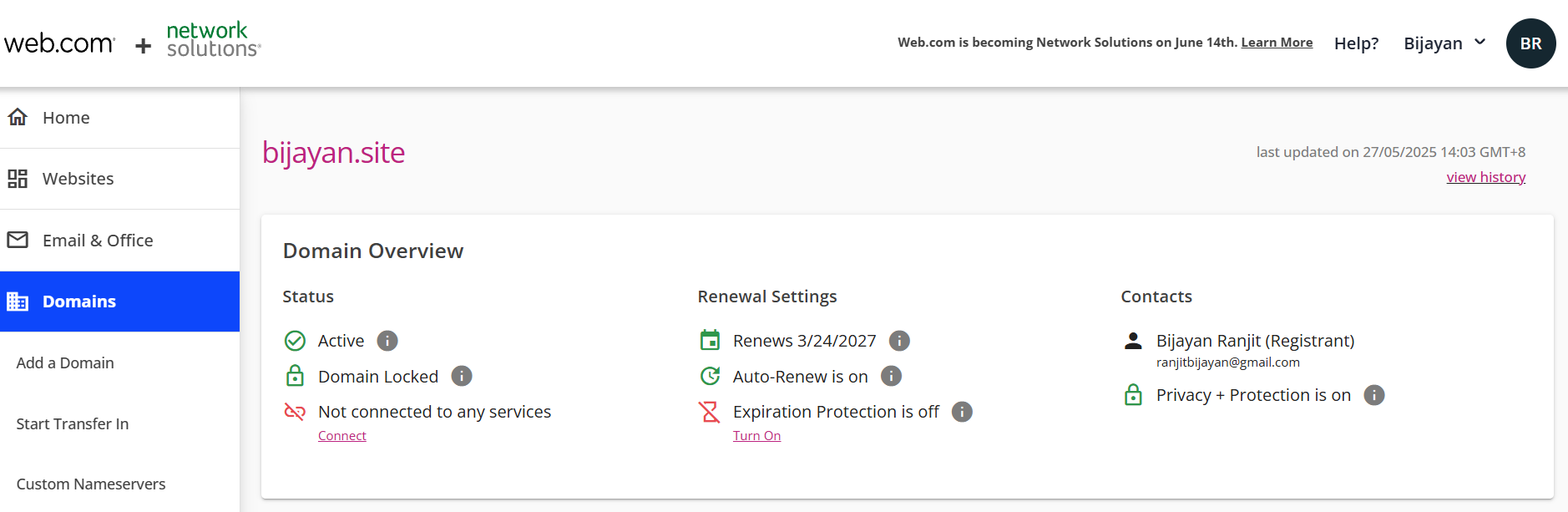
### Domain Name and DNS Setup

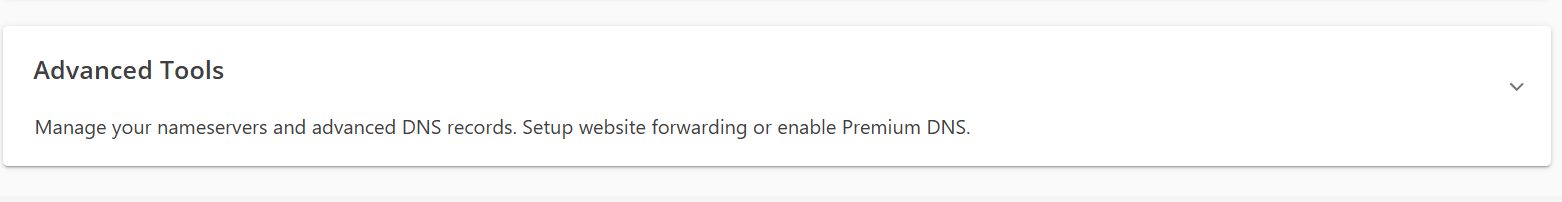
**Step 5: Point Domain to Server IP**

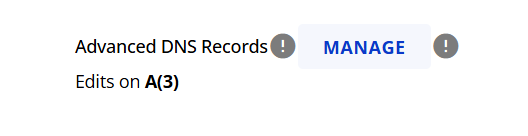
* Login to your domain registrar.
* Add an A record for @ and www pointing to your server's IP.

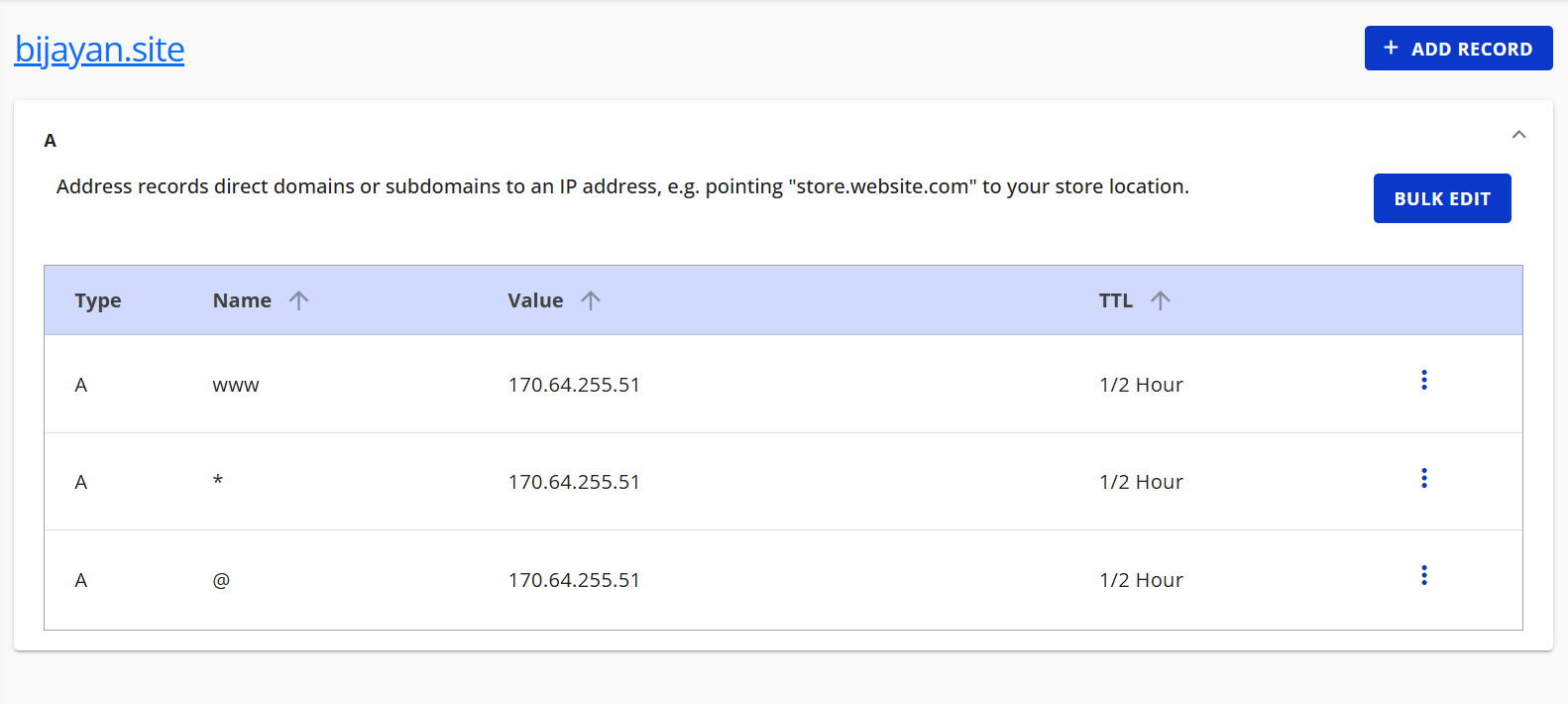
Brief: This links your domain to the server so it can be accessed online.

Screenshot: DNS records configuration.









### 4. SSL/TLS with Certbot

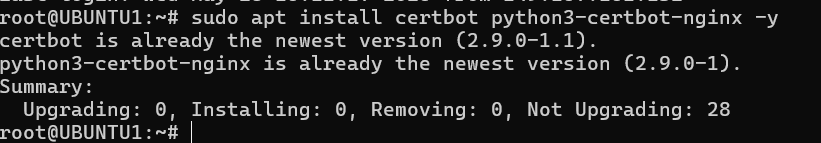
**Step 6: Install Certbot and SSL**

sudo apt install certbot python3-certbot-nginx -y

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

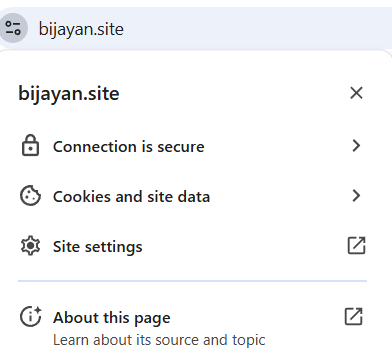
Brief: Automatically secures your site with HTTPS using Let's Encrypt.

Screenshot: Certbot SSL certificate installation.









### 5. Deploy the Website

**Step 7: Upload Files to Server**

scp -r \* root@170.64.255.51:/var/www/html/

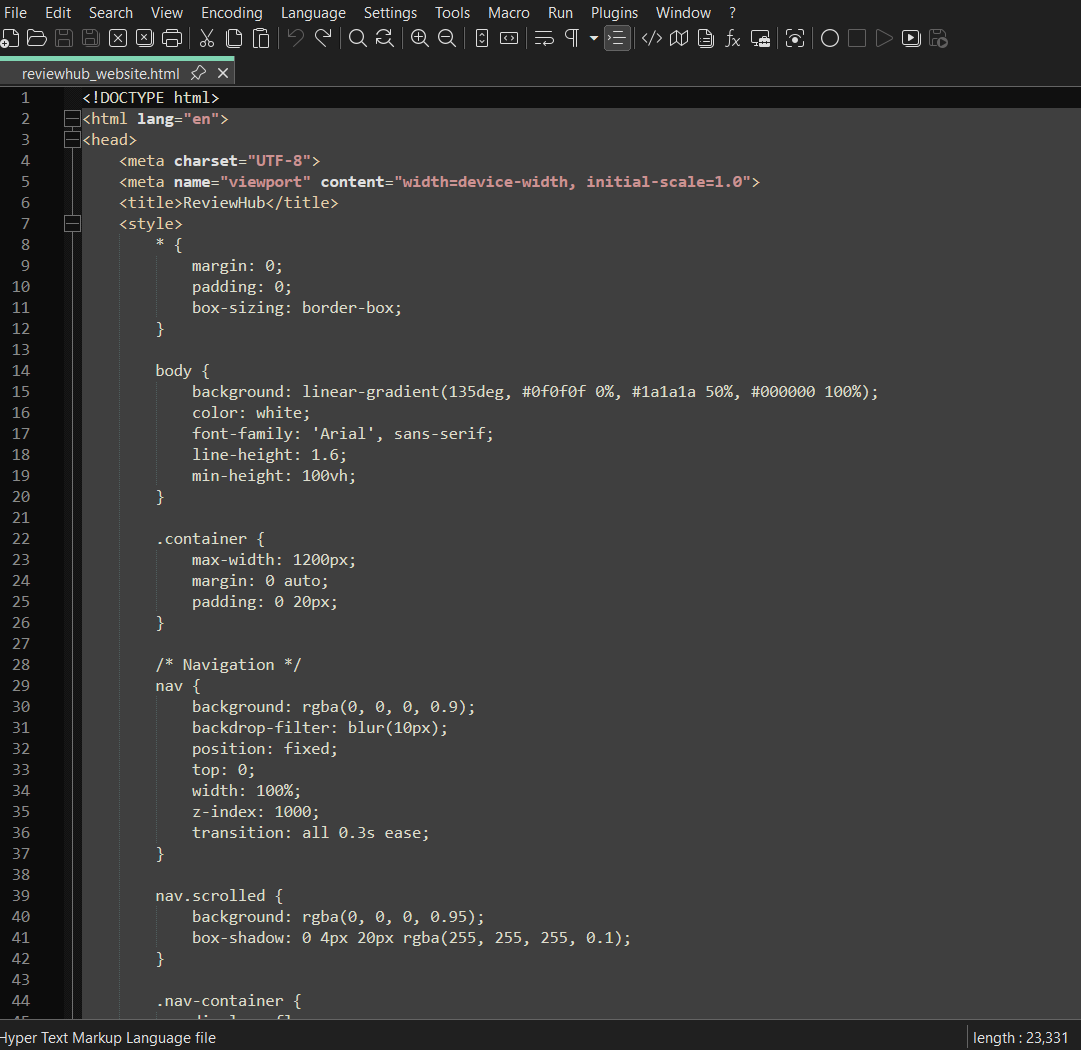
Brief: Copies your HTML/CSS/JS files to the server's web directory.

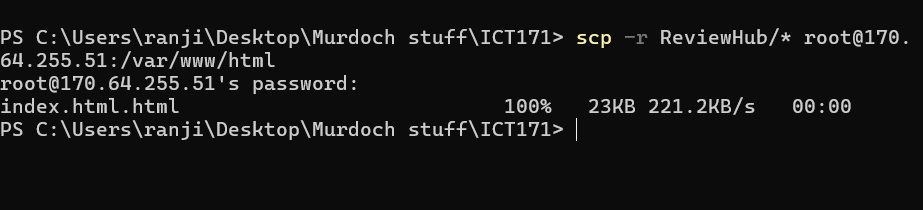
**Step 8: Set Permissions**

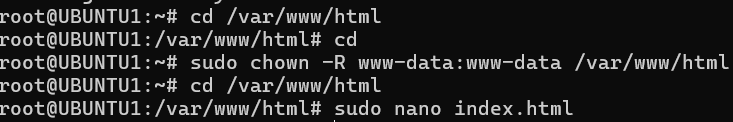
sudo chown -R www-data:www-data /var/www/html

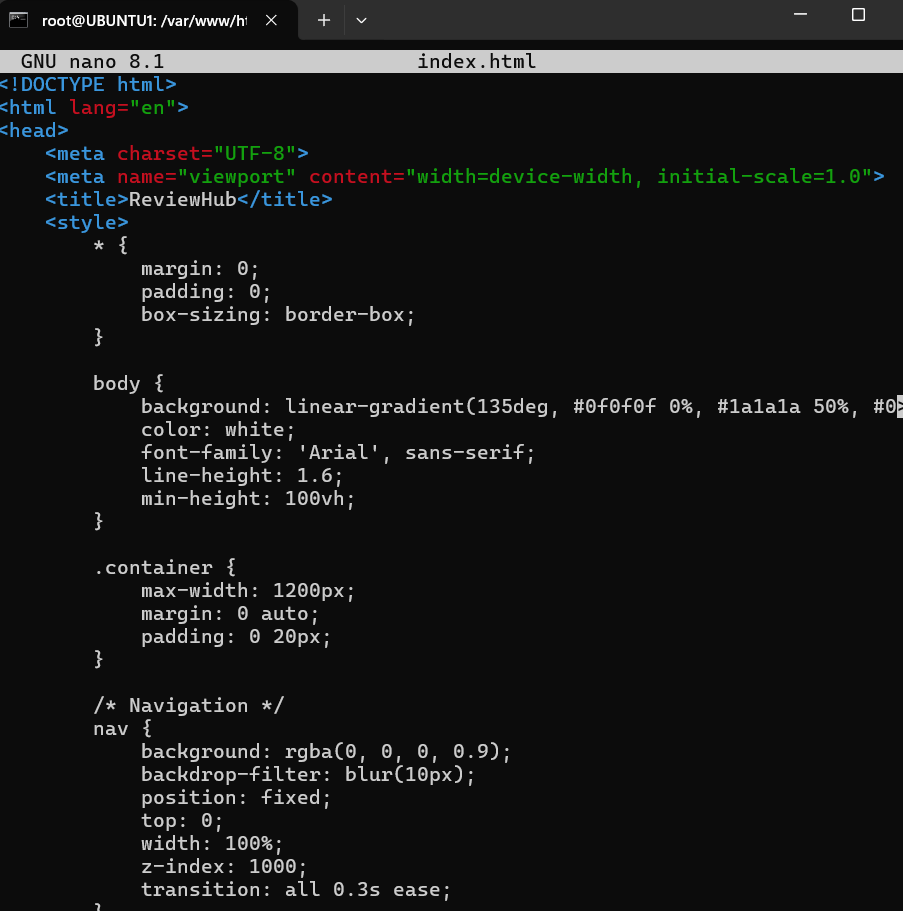
Brief: Ensures Nginx can read the website files.

Screenshot: File upload and permission setting in terminal.









### Bash Script: Server Monitor

**Script Purpose:** Checks if Nginx is running and logs uptime.

🡪#!/bin/bash

🡪date >> /var/log/server\_status.log

🡪systemctl status nginx >> /var/log/server\_status.log

Brief: Simple log script to track if Nginx is active.

**Run with cron:**

🡪 crontab –e

🡪 \*/30 \* \* \* \* /bin/bash /path/to/script.sh

Brief: Runs every 30 mins and appends output to log file.

### GitHub and Version Control

**Key Steps:**

* Initialize repository: git init
* Commit changes: git add . && git commit -m "Initial commit"
* Push to GitHub: git remote add origin <repo-url> and git push -u origin main

Brief: Allows version tracking and easy project sharing.

### Video Explainer (To be recorded)

Show:

* Server dashboard
* File uploads
* Nginx test
* SSL status
* Website demo

Brief: Demonstrates key technical steps visually.

### Final Checklist

| **Task** | **Status** |
| --- | --- |
| Real IP address is included in the documentation | ✅ |
| Domain name is included and resolves correctly | ✅ |
| Website loads successfully over HTTPS (SSL/TLS active) | ✅ |
| Nginx service is running and serving the site | ✅ |
| Bash script logs server status to /var/log/server\_status.log | ✅ |
| Cron job runs the script every 30 minutes | ✅ |
| GitHub repository includes all relevant files and scripts | ✅ |
| README.md explains the project purpose and structure | ✅ |
| Video explainer is uploaded and link is added to doc | ✅ |
| Screenshots of setup are embedded in documentation | ✅ |
| Final DOCX and/or PDF exported and ready for LMS submission | ✅ |
| Server is left running and publicly accessible | ✅ |