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: 170.64.255.51

Domain Name: https://bijayan.site

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

Video Explainer: https://youtu.be/piE0l5LBDTQ

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.

Create Droplets

Learn

Droplets are virtual machines that anyone can setup in seconds. You can use droplets, either standalone or as part of a larger, cloud based infrastructure.

Choose Region

New York	San Francisco	Amsterdam
Singapore	London	Frankfurt
Toronto	Bangalore	Sydney

Datacenter

New York - Datacenter 1 - NYC1

Tip: Select the datacenter closest to you or your users Dismiss

Avoid any potential latency by selecting a region closest to you - a region is a geographic area where we have one or more datacenters.

VPC Network - default-nyc1 DEFAULT

All resources created in this datacenter will be members of the same VPC network. They can communicate securely over their Private IP addresses.

Choose an image

OS Marketplace (285) Custom images

Ubuntu	Fedora	Debian	CentOS	AlmaLinux	Rocky Linux

Version

24.10 x64

Choose Size

Need help picking a plan? Help me choose

Droplet Type

SHARED CPU	DEDICATED CPU			
Basic (Plan selected)	General Purpose	CPU-Optimized	Memory-Optimized	Storage-Optimized

Basic virtual machines with a mix of memory and compute resources. Best for small projects that can handle variable levels of CPU performance, like blogs, web apps and dev/test environments.

CPU options

Regular	Premium Intel	Premium AMD
Disk type: SSD	Disk: NVMe SSD	Disk: NVMe SSD

$6/mo	$12/mo	$18/mo	$24/mo	$48/mo	$96/mo
$0.009/hour	$0.018/hour	$0.027/hour	$0.036/hour	$0.071/hour	$0.143/hour
1 GB / 1 CPU	2 GB / 1 CPU	2 GB / 2 CPUs	4 GB / 2 CPUs	8 GB / 4 CPUs	16 GB / 8 CPUs
25 GB SSD Disk	50 GB SSD Disk	60 GB SSD Disk	80 GB SSD Disk	160 GB SSD Disk	320 GB SSD Disk
1000 GB transfer	2 TB transfer	3 TB transfer	4 TB transfer	5 TB transfer	6 TB transfer

Show all plans

Additional Storage

Add Volume

Need more disk space? Add a volume with no manual setup.
Block storage volumes add extra disk space. We automatically format and mount your volume so it's available as soon as your Droplet is, and you can move volumes seamlessly between Droplets at any time. Think of it like a flash drive for your VM.

Backups

Enable automated backup plan
Automatically take backups at the time you specify

Choose Authentication Method

SSH Key	Password
Connect to your Droplet with an SSH key pair	Connect to your Droplet as the "root" user via password

Create root password *

Type your password...

PASSWORD REQUIREMENTS

- Must be at least 8 characters long
- Must contain 1 uppercase letter (cannot be first or last character)
- Must contain 1 number
- Cannot end in a number or special character

⚠ Please store your password securely. You will not be sent an email containing the Droplet's details or password.

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

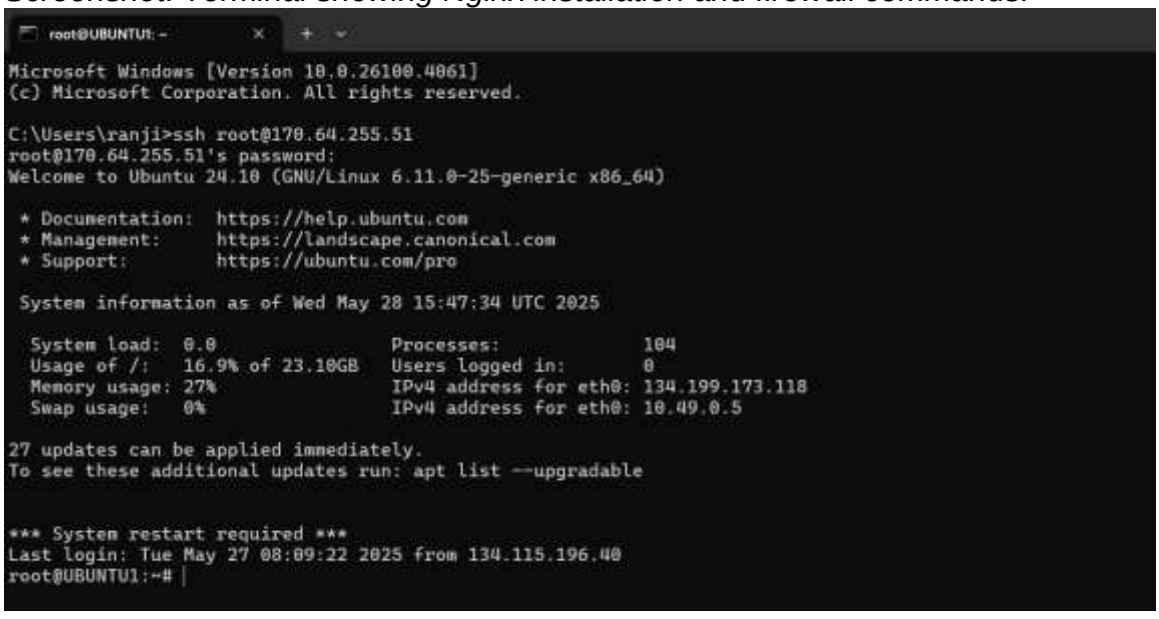
Brief: Nginx will serve your HTML website files.

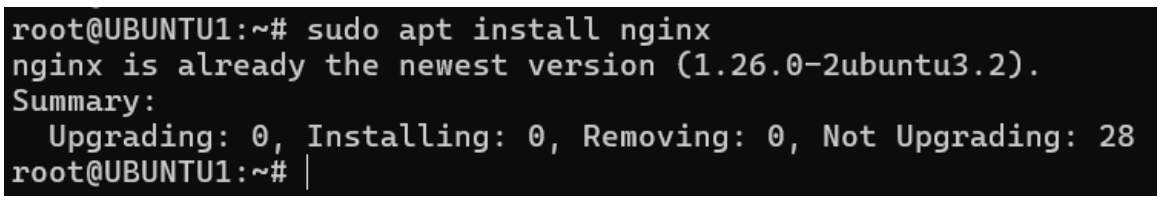
Step 4: Enable Nginx and Allow Firewall

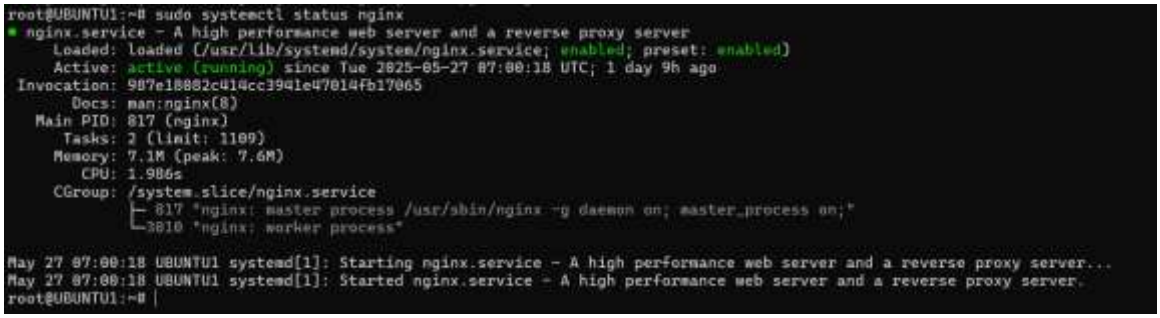
sudo ufw allow 'Nginx Full'
sudo systemctl enable nginx
sudo systemctl start nginx

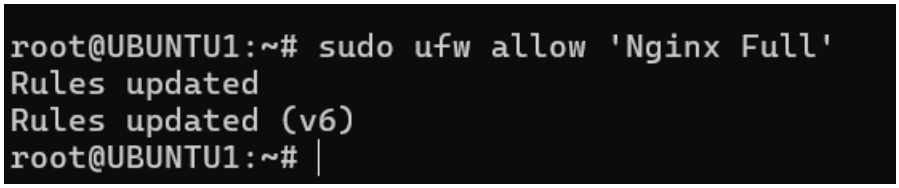
function: Opens required ports and ensures Nginx runs on boot.

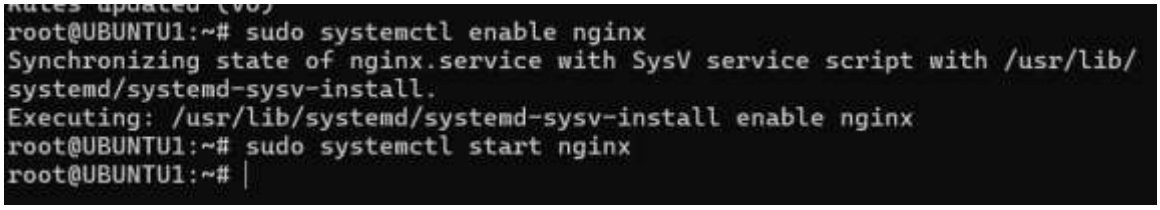
Screenshot: Terminal showing Nginx installation and firewall commands.











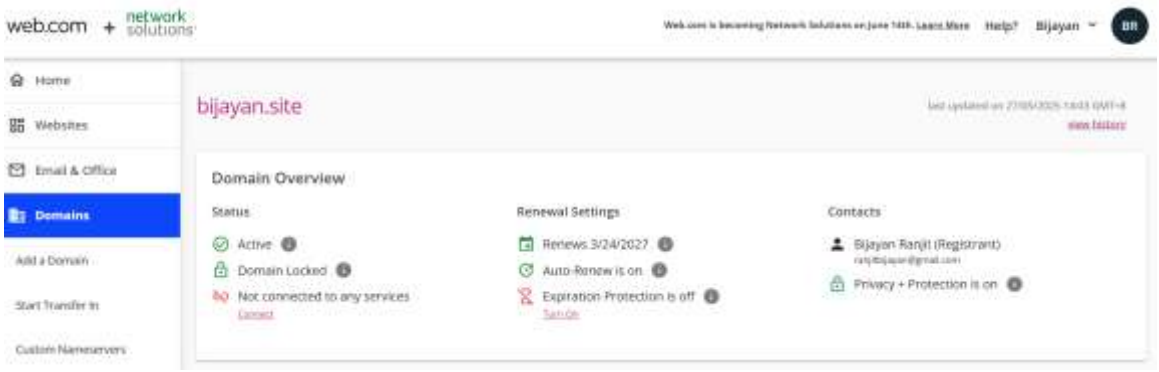
Domain Name and DNS Setup

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.

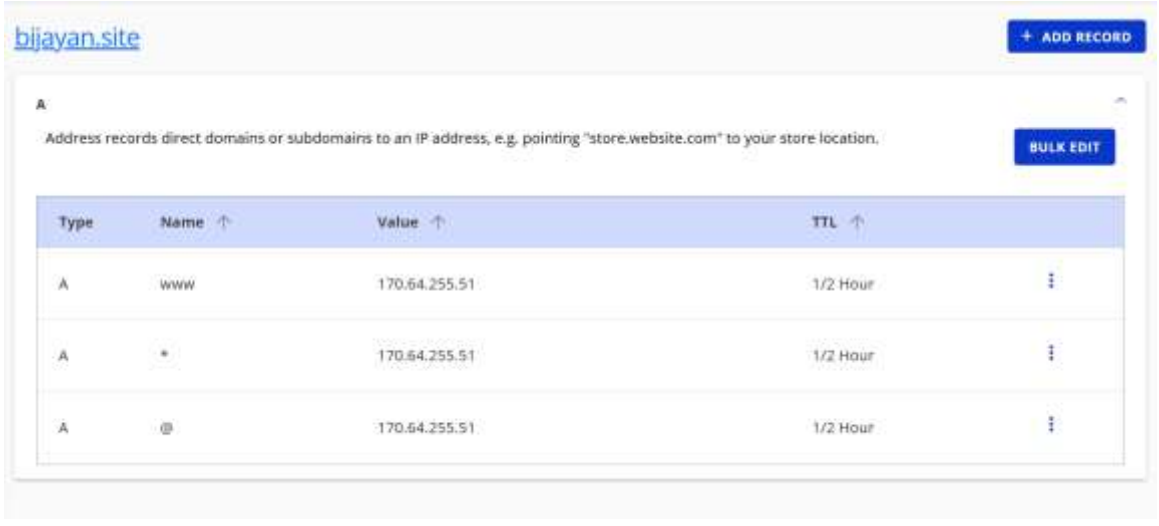
Brief: This links your domain to the server so it can be accessed online.

Screenshot: DNS records configuration.



Advanced DNS Records MANAGE

Edits on **A(3)**



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

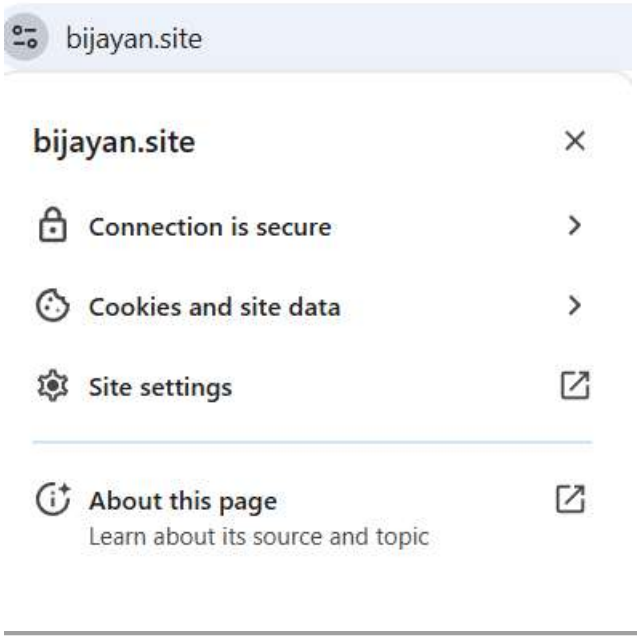
Brief: Automatically secures your site with HTTPS using Let's Encrypt.

Screenshot: Certbot SSL certificate installation.

```
root@UBUNTU1:~# sudo apt install certbot python3-certbot-nginx -y
certbot is already the newest version (2.9.0-1.1).
python3-certbot-nginx is already the newest version (2.9.0-1).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 28
root@UBUNTU1:~#
```

```
root@UBUNTU1:~# sudo certbot --niginx -d bijayan.site www.bijayan.site
```

```
sudo certbot renew --dry-run
```



5. Deploy the Website

Step 7: Upload Files to Server

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

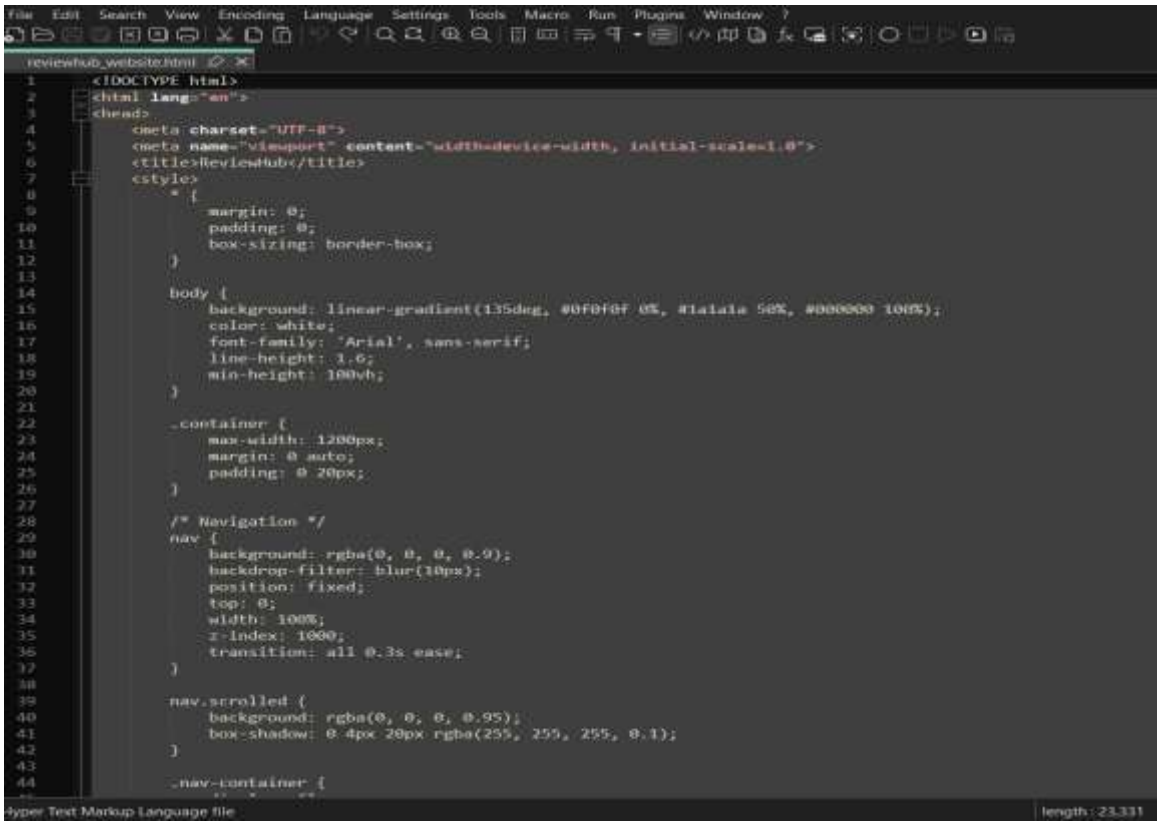
Brief: Copies your HTML/CSS/JS files to the server's web directory.

Step 8: Set Permissions

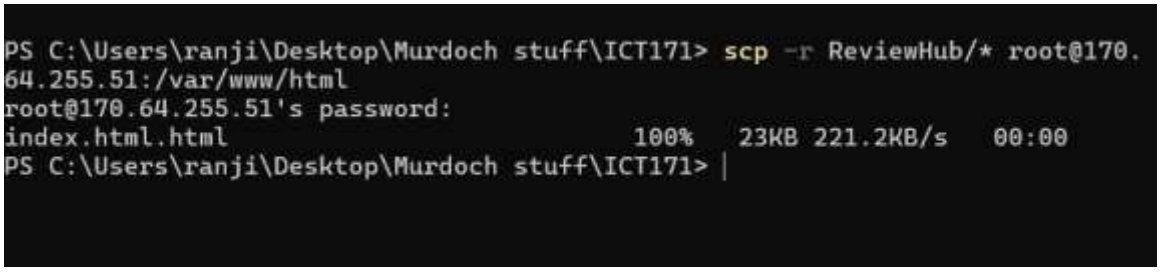
sudo chown -R www-data:www-data /var/www/html

Brief: Ensures Nginx can read the website files.

Screenshot: File upload and permission setting in terminal.

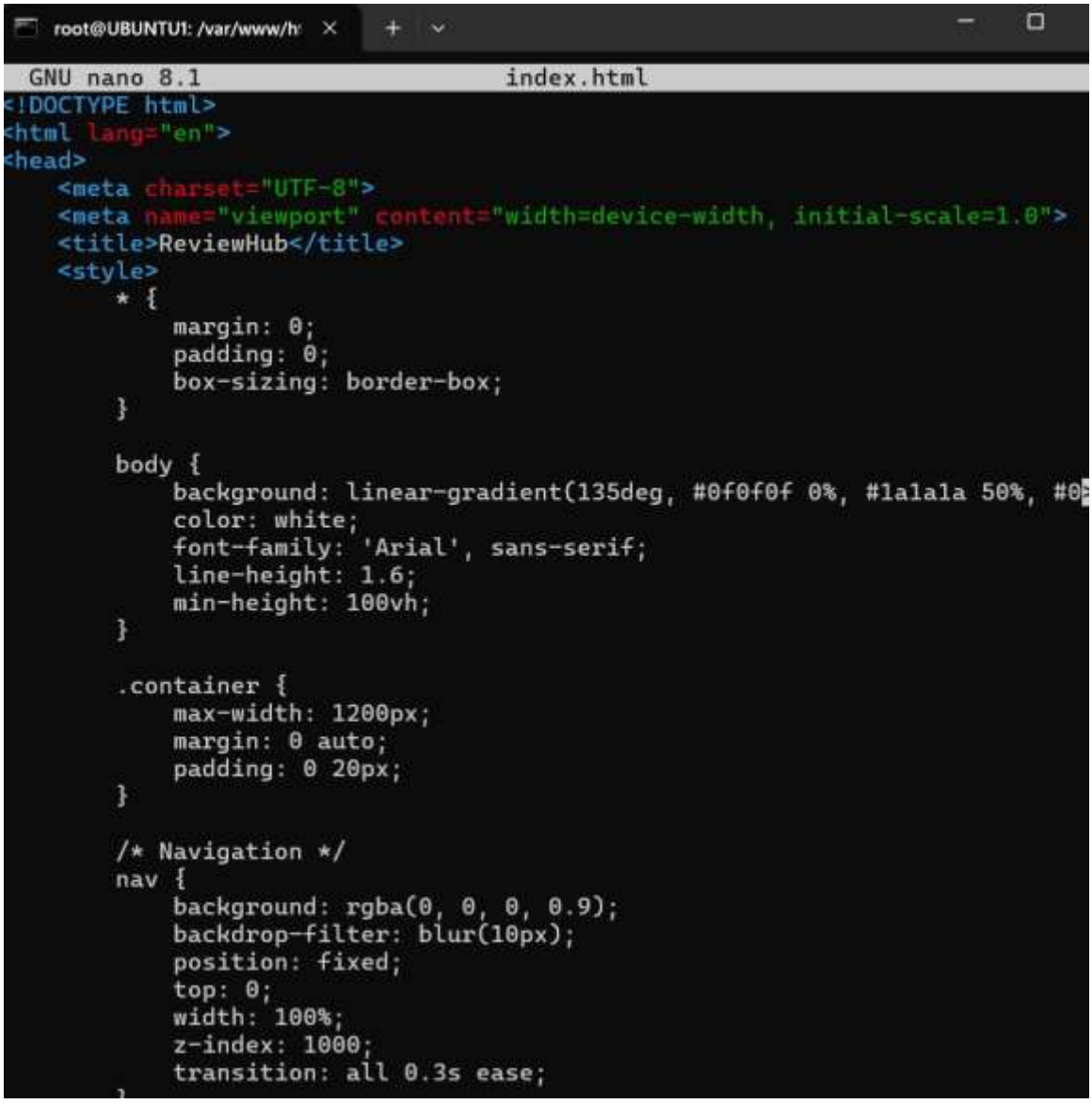


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ReviewHub</title>
    <style>
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            background: linear-gradient(135deg, #0f0f0f 0%, #1a1a1a 50%, #000000 100%);
            color: white;
            font-family: 'Arial', sans-serif;
            line-height: 1.6;
            min-height: 100vh;
        }

        .container {
            max-width: 1200px;
            margin: 0 auto;
            padding: 0 20px;
        }

        /* Navigation */
        nav {
            background: rgba(0, 0, 0, 0.9);
            backdrop-filter: blur(10px);
            position: fixed;
            top: 0;
            width: 100%;
            z-index: 1000;
            transition: all 0.3s ease;
        }

        nav.scrolled {
            background: rgba(0, 0, 0, 0.95);
            box-shadow: 0 4px 20px rgba(255, 255, 255, 0.1);
        }

        .nav-container {
```



```
PS C:\Users\ranji\Desktop\Murdoch stuff\ICT171> scp -r ReviewHub/* root@170.
64.255.51:/var/www/html
root@170.64.255.51's password:
index.html.html                               100%   23KB 221.2KB/s   00:00
PS C:\Users\ranji\Desktop\Murdoch stuff\ICT171>
```

```
root@UBUNTU1:~# cd /var/www/html
root@UBUNTU1:/var/www/html# cd
root@UBUNTU1:~# sudo chown -R www-data:www-data /var/www/html
root@UBUNTU1:~# cd /var/www/html
root@UBUNTU1:/var/www/html# sudo nano index.html
```



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	✅