**Documentation for ICT171: Static Personal Portfolio Deployment (IP Version)**

**Author:** Sonam Yoesel Dorji

**Student Number**: 35418679

**Github:** <https://github.com/raiden421/portfolio-server>

Video Explanation link on Youtube: <https://youtu.be/xSmDe7kuCGU>

**Date:** June 9, 2025

**Abstract**

This paper gives an in-depth, step-by-step plan to deploy a static personal portfolio website (HTML, CSS, JavaScript) on an AWS EC2 Ubuntu instance with the public IP [**http://13.210.81.35/**](http://13.210.81.35/). It outlines the provisioning of the cloud infrastructure, enforced security features with SSH key authentication, installation and tweaking of nginx as a static content server, automated continuous deployment using GitHub, detailed cost analysis, and suggests ideas for future improvements for scalability, reliability, and maintainability.

**1. Introduction**

In a time where having an online presence is so significant, self-hosting an online portfolio shows competencies in development and operations. All the steps in the complete life cycle of a web deployment:

• Development and debugging from a local environment

• Provisioning cloud resources

• Secure remote access and hardening the systems

• Web server configuration and deployment of static assets

• Connecting with source control

• Automated updates and continuous delivery pipelines

Conducting this project demonstrates competency and evidence for proficiency in key DevOps practices and gain hands-on experience with technologies widely used today such as AWS, SSH, nginx, and GitHub. The live site can be found at [**http://13.210.81.35/**](http://13.210.81.35/).

**2. Infrastructure Setup**

**2.1 AWS EC2 Instance Provisioning**

**Platform Selection:** AWS was chosen for its free-tier t2.micro instances and comprehensive documentation.  
**Instance Details:**

* AMI: Ubuntu Server 22.04 LTS (HVM), SSD Volume Type
* Type: t2.micro (1 vCPU, 1 GB RAM)
* Storage: 8 GB gp2 SSD
* Network: Default VPC with a security group permitting ports 22 (SSH) and 80 (HTTP)
* SSH Key Pair: Created and named portfolio-key
* Region: us-east-1 (Northern Virginia)

**Procedure:**

1. Sign into the AWS Management Console
2. Go to EC2 → Instances → Launch Instances.
3. Select the given AMI and instance type.
4. Under Key pair (login) section, select Create new key pair and download portfolio-key.pem.
5. Configure the security group to allow inbound ports 22 & 80.
6. Check review and launch instance.
7. Document the instance's Public IPv4 address: 13.210.81.35.

**2.2 Networking (IP Only)**

Since there was no custom domain used, the server accepts HTTP requests directly at http://13.210.81.35/. When a domain is added in future, the DNS records could be configured to associate a domain name with the IP.

**3. SSH Key Management and Security Hardening**

Secure Shell (SSH) access improves security and convenience; the following measures were taken:

1. **Local Key Generation** (Git Bash on Windows):
2. ssh-keygen -t rsa -b 4096 -f ~/.ssh/portfolio-key -C "your\_email@example.com"
3. **Key Upload to AWS**: It is assumed that the public key was imported at the instance launch.
4. **Server-Side Configuration**:
   * Disabled password authentication by editing /etc/ssh/sshd\_config:
   * PasswordAuthentication no
   * PermitRootLogin no
   * Restarted SSH: sudo systemctl restart sshd
5. **Local File Permissions:** Set private key to chmod 400 ~/.ssh/portfolio-key.
6. **Firewall Enforcement:** Enabled UFW:
7. sudo ufw allow OpenSSH
8. sudo ufw allow "Nginx HTTP"
9. sudo ufw enable
10. **User Management:** Created a non-root user deployer with sudo privileges:
11. sudo adduser deployer
12. sudo usermod -aG sudo deployer

Combined, these steps minimize attack surface and enforce strict access control.

**4. nginx Installation and Static Site Deployment**

**4.1 Installing nginx**

sudo apt update && sudo apt install -y nginx

sudo systemctl enable nginx --now

Service status: active.

**4.2 Directory Structure**

sudo mkdir -p /var/www/portfolio

sudo chown -R deployer:deployer /var/www/portfolio

Place your static files under /var/www/portfolio.

**4.3 Configuring nginx Server Block**

Create /etc/nginx/sites-available/portfolio:

server {

listen 80;

server\_name \_;

root /var/www/portfolio;

index index.html;

location / {

try\_files $uri $uri/ =404;

}

access\_log /var/log/nginx/portfolio\_access.log;

error\_log /var/log/nginx/portfolio\_error.log;

}

Enable and reload:

sudo ln -s /etc/nginx/sites-available/portfolio /etc/nginx/sites-enabled/

sudo nginx -t

sudo systemctl reload nginx

Validate in browser: http://13.210.81.35/.

**5. Version Control and Continuous Deployment**

**5.1 GitHub Integration**

Install and configure Git:

sudo apt install git -y

Add server SSH key to GitHub and clone:

git clone git@github.com:yourusername/portfolio.git /var/www/portfolio

**5.2 Deployment Automation**

Create /var/www/portfolio/deploy.sh:

#!/bin/bash

cd /var/www/portfolio

git pull origin main

Make executable and schedule via cron:

chmod +x /var/www/portfolio/deploy.sh

(crontab -l ; echo "\*/5 \* \* \* \* /var/www/portfolio/deploy.sh >> /var/log/deploy.log 2>&1") | crontab -

Check logs: /var/log/deploy.log.

**6. Cost Analysis**

| **Resource** | **Configuration** | **Monthly Cost Estimate** |
| --- | --- | --- |
| EC2 t2.micro | 1 vCPU, 1 GB RAM, Free Tier | $0 |
| EBS (8 GB SSD) | gp2 | $0.80 |
| Data Transfer | 15 GB free per month | $0 |
| **Total** |  | **$0.80** |

**7. Future Enhancements**

* **SSL/TLS Integration:** Register a free domain and install certificates for https://13.210.81.35/ (subject to domain binding).
* **Multi-Region Deployment:** Deploy identical stacks in other AWS regions.
* **Containerization:** Package with Docker and orchestrate via Kubernetes.
* **CI/CD Pipeline:** Use GitHub Actions for automated testing and deployment.
* **Monitoring and Alerts:** Integrate AWS CloudWatch or Prometheus/Grafana.

**8. Acknowledgements**

I gratefully acknowledge the guidance and debugging assistance provided by generative AI tools, particularly ChatGPT (OpenAI), as well as documentation from AWS, nginx, and GitHub.

**9. References**

Amazon Web Services. (2025). *Launching an EC2 Instance*. <https://docs.aws.amazon.com/ec2/>  
Chacon, S., & Straub, B. (2024). *Pro Git* (3rd ed.). Apress.  
Falkner, T. (2023). *Mastering nginx* (2nd ed.). Packt Publishing.  
GitHub Docs. (2025). *Managing SSH Keys*. <https://docs.github.com/en/authentication/connecting-to-github-with-ssh>  
Let’s Encrypt. (2025). *Certbot Documentation*. <https://certbot.eff.org/>  
Ubuntu Documentation. (2025). *Ubuntu Server Guide: Security*. <https://ubuntu.com/server/docs/security-overview>

**10. Figures and Images**

1. **SSH Connection:** Git Bash session showing successful SSH login to ubuntu@13.210.81.35.
2. **Directory Structure:** File tree of /var/www/portfolio with HTML, CSS, JS files.
3. **nginx Server Block:** Screenshot of Nano editor with /etc/nginx/sites-available/portfolio.
4. **Live Site:** Browser showing the homepage at [**http://13.210.81.35/**](http://13.210.81.35/).
5. **Cron Deployment Log:** Sample entries from /var/log/deploy.log.

*(Figures within the final PDF.)*

