Comprehensive Guide to Hosting Website Publicly.

There are several methods to host a website:

1. **Cloud Virtual Machines (VMs):** Providers such as AWS, Google Cloud, Azure, and Digital Ocean offer the ability to spin up a full VM in the cloud. These options provide full control and scalability but typically come with associated costs.
2. **Free Hosting/VPS for Static Sites:** Platforms like GitHub Pages, Netlify, and Vercel are ideal for static site applications. They often offer generous free tiers.
3. **Local/Personal Systems with Port Forwarding**: You can run a web server on your personal computer and expose it to the internet by configuring your router's port forwarding. This method provides complete control but depends on the reliability of your home network.

Due to limited resources, I opted for hosting on Netlify. I pointed my domain name to Netlify, which provided a smooth and cost-effective way to deploy my site.

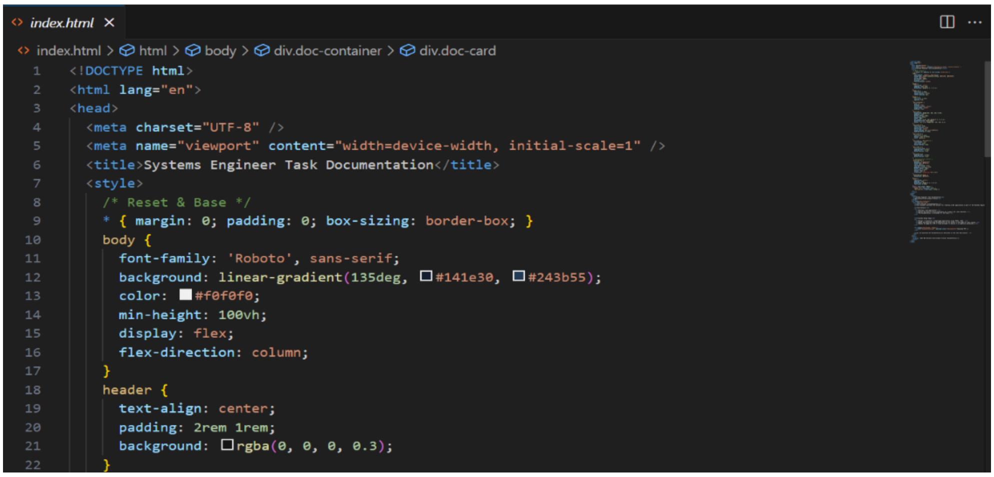
Introduction

As mentioned, there are multiple methods to host your website. You could deploy a full virtual machine (VM) in the cloud or other options of cloud like Docker, Kubernetes, App services and cloud function or Cloud Run (in GCP), or use free VPS or static hosting platforms for sites or even run a server on your local machine with port forwarding. In my project, I needed a solution that was simple, fast, and free—so I chose Netlify. I pointed my domain name to Netlify, allowing me to take advantage of its free hosting for static sites while bypassing the need for a full-fledged cloud VM

Preparing Your Website Code

Before deployment, it's essential to ensure that your website's code is structured, valid, and free from unresolved dependencies. Follow these steps:

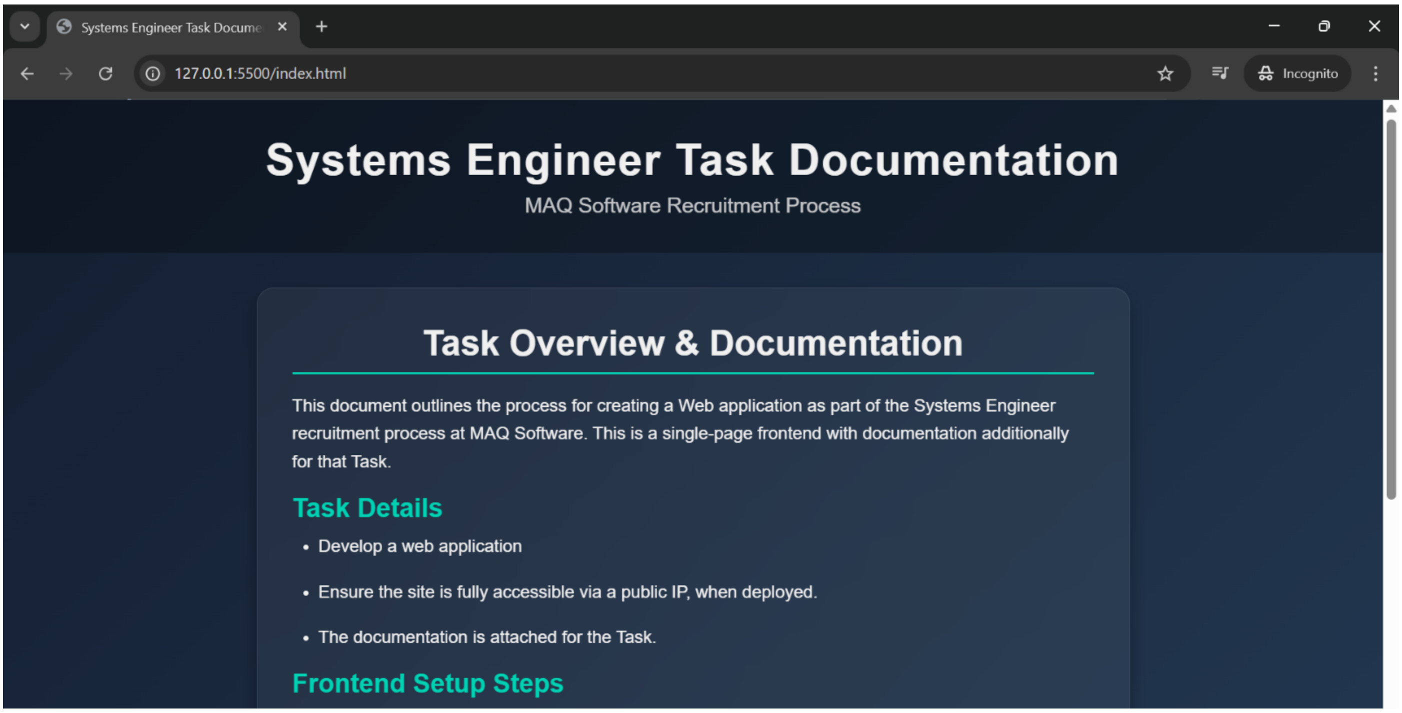
- **Code Structure & Standards:**
  Ensure your HTML, CSS, and JavaScript code follows best practices and web standards. Validate your HTML using the and your CSS.



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Systems Engineer Task Documentation</title>
  <style>
    /* Reset & Base */
    * { margin: 0; padding: 0; box-sizing: border-box; }
    body {
      font-family: 'Roboto', sans-serif;
      background: linear-gradient(135deg, #141e30, #243b55);
      color: #f0f0f0;
      min-height: 100vh;
      display: flex;
      flex-direction: column;
    }
    header {
      text-align: center;
      padding: 2rem 1rem;
      background: rgba(0, 0, 0, 0.3);
    }
```

- **Dependency Check:(If Required)**
  Verify that all dependencies (external libraries, frameworks, images, fonts, etc.) are correctly referenced. For dynamic sites that require a build process (using tools like npm, webpack, or Gulp), run your build command locally to see if any errors occur.

- **Local Testing:**
  Open your site locally in various browsers to check for compatibility issues. This helps you catch problems early before deploying.



Systems Engineer Task Documentation

MAQ Software Recruitment Process

Task Overview & Documentation

This document outlines the process for creating a Web application as part of the Systems Engineer recruitment process at MAQ Software. This is a single-page frontend with documentation additionally for that Task.

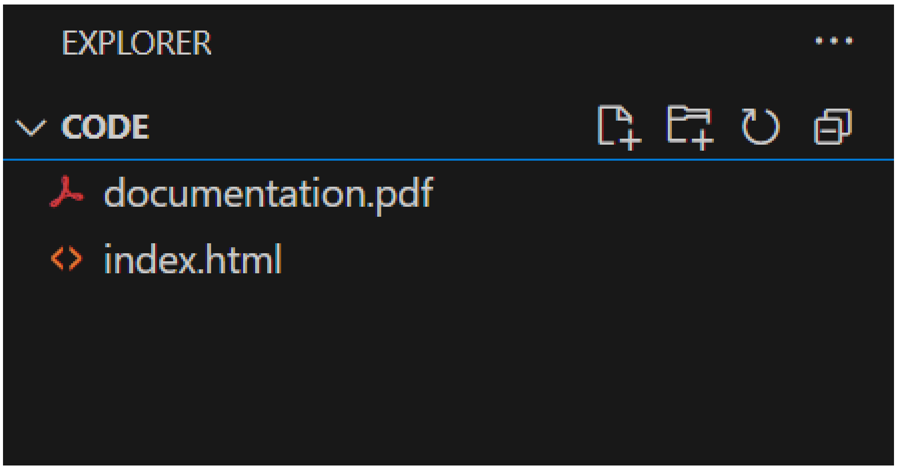
Task Details

- Develop a web application
- Ensure the site is fully accessible via a public IP, when deployed.
- The documentation is attached for the Task.

Frontend Setup Steps

- **Responsive Design:**
  Ensure that your website's design is responsive and looks good on different devices (desktop, tablet, mobile). You can use browser developer tools to simulate various screen sizes.

- **File Structure:**
  Keep a clean file structure with separate directories for HTML, CSS, JavaScript, images, and other assets. This organization simplifies debugging and future updates.



documentation.pdf
index.html

By confirming that your code meets the required standards and works as expected locally, you'll have a solid foundation for a smooth deployment to Netlify.

- **Prerequisites**

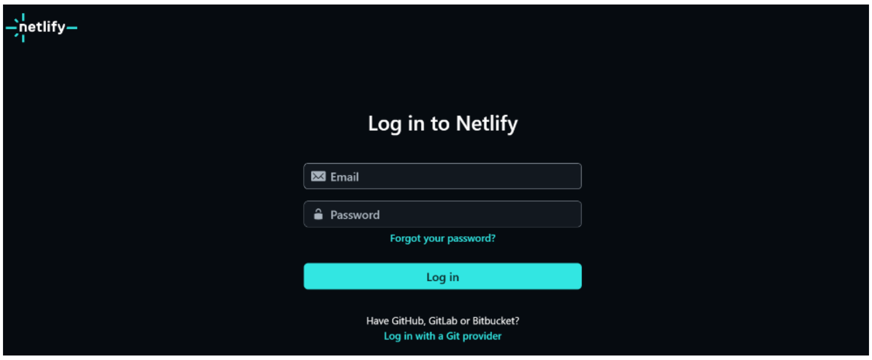
  Before starting, ensure you have:

  - A Netlify account.
  - A Git repository (on GitHub, GitLab) containing your website's code.
  - Your website validated and tested locally (see **Preparing Your Website Code** above).
  - Basic knowledge of Git and your website's build process (if applicable).

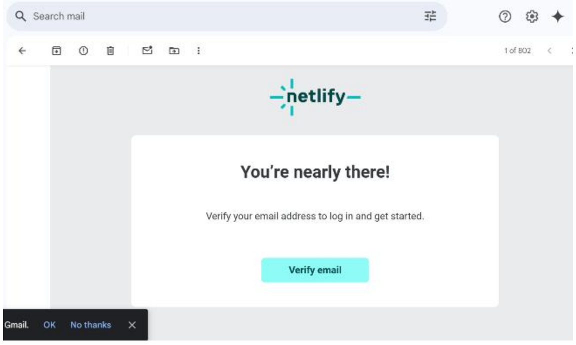
- **Step-by-Step Deployment Guide**

  a. Creating a Netlify Account

  1. **Sign Up:**
     Visit Netlify's signup page and create an account using your email or a third-party login (GitHub, GitLab, or Bitbucket).



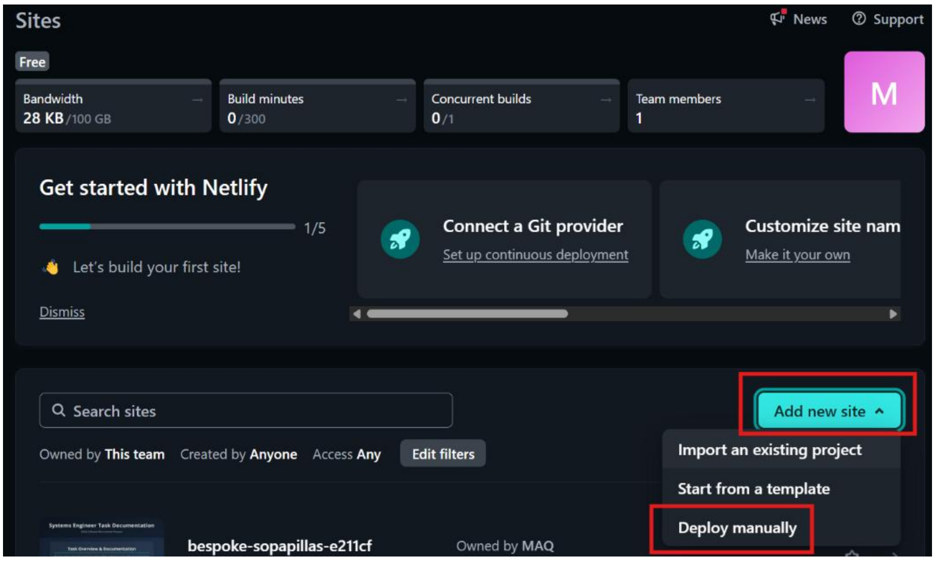
  2. **Verify Your Email:**
     Confirm your email address by clicking the verification link sent to your inbox.


You're nearly there!

Verify your email address to log in and get started.

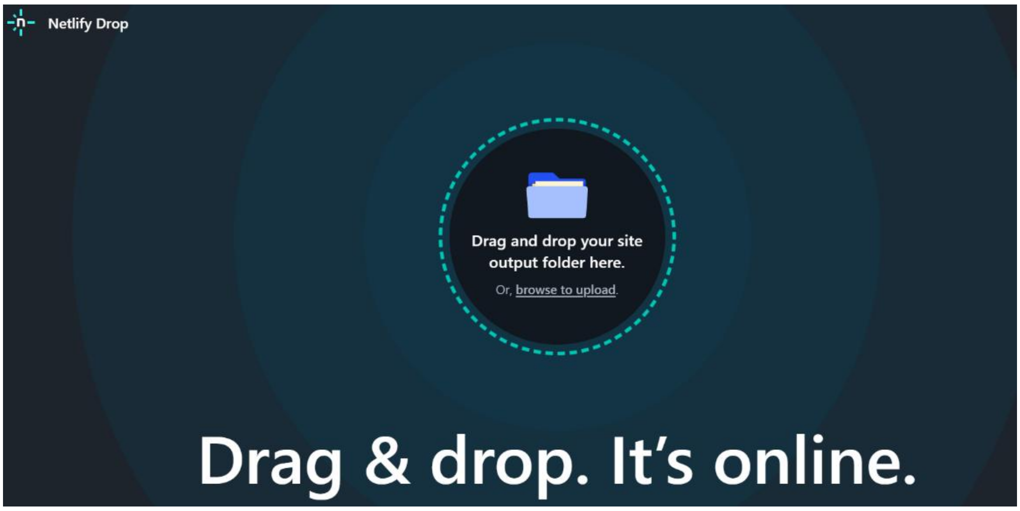
b. Connecting Your Repository

1. **New Site from Git:**
   In your Netlify dashboard, click the **"New site from Git"** button.



2. **Choose Your Git Provider:**
   Select GitHub, GitLab, or Bitbucket. Authorize Netlify to access your repositories if prompted.

3. **Select Repository:**
   Choose the repository that contains your website code.

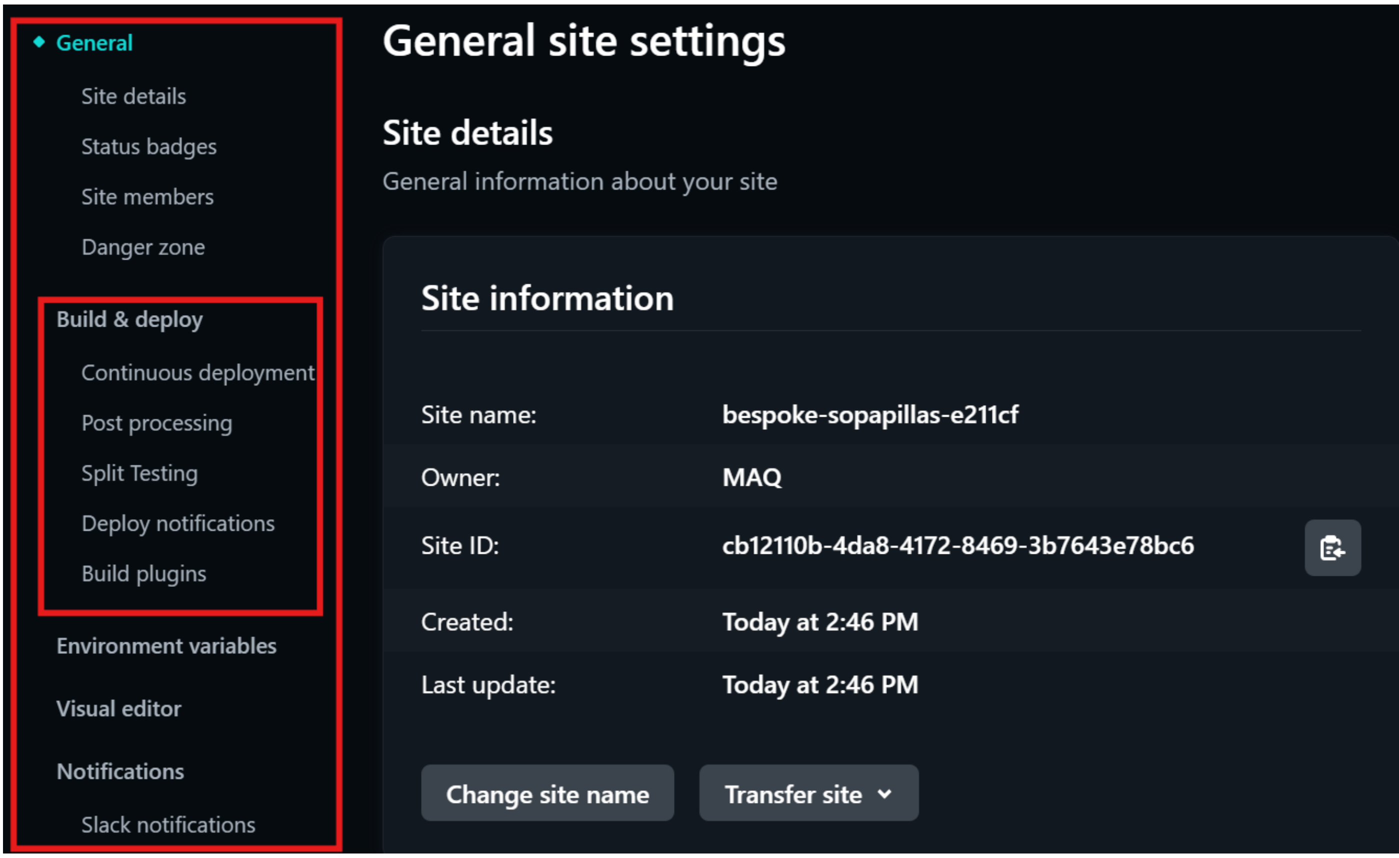

Netlify Drop
Drag and drop your site output folder here.
Or, browse to upload.
Drag & drop. It's online.

c. Configuring Build Settings (optional if required)

1. **Build Command:**
   Enter the build command (e.g., npm run build, gatsby build, or jekyll build) used to compile your website.

2. **Publish Directory:**
   Specify the folder where your built site resides (e.g., build, public, or _site).

3. **Advanced Settings (Optional):**
   Add any required environment variables or custom configurations.



General site settings

Site details

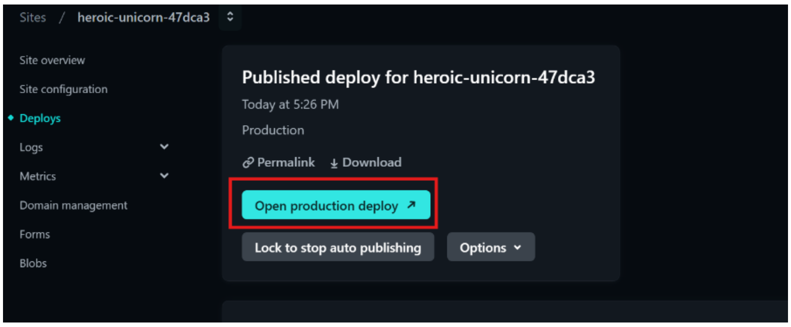
General information about your site

Site information

Site name:	bespoke-sopapillas-e211cf
Owner:	MAQ
Site ID:	cb12110b-4da8-4172-8469-3b7643e78bc6
Created:	Today at 2:46 PM
Last update:	Today at 2:46 PM

d. Deploying Your Site

1. **Deploy Site:**
   Click **"Deploy site"**. Netlify will pull your repository, run the build command, and deploy your site.

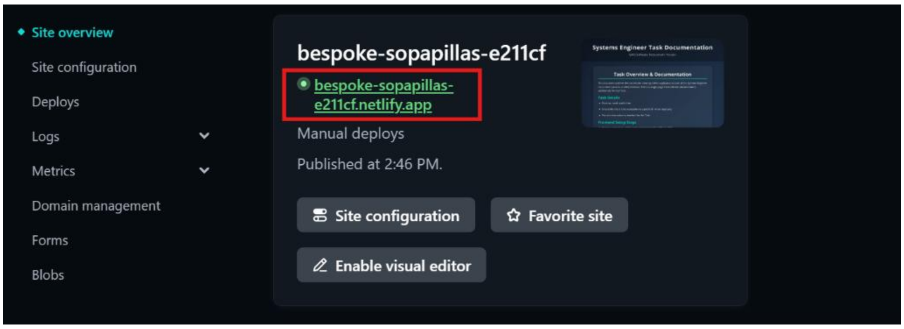


Published deploy for heroic-unicorn-47dca3

Today at 5:26 PM

Production

2. **Live URL:**
   After deployment, Netlify provides a temporary URL (e.g., https://your-site-name.netlify.app). You can later set up a custom domain.



bespoke-sopapillas-e211cf

bespoke-sopapillas-e211cf.netlify.app

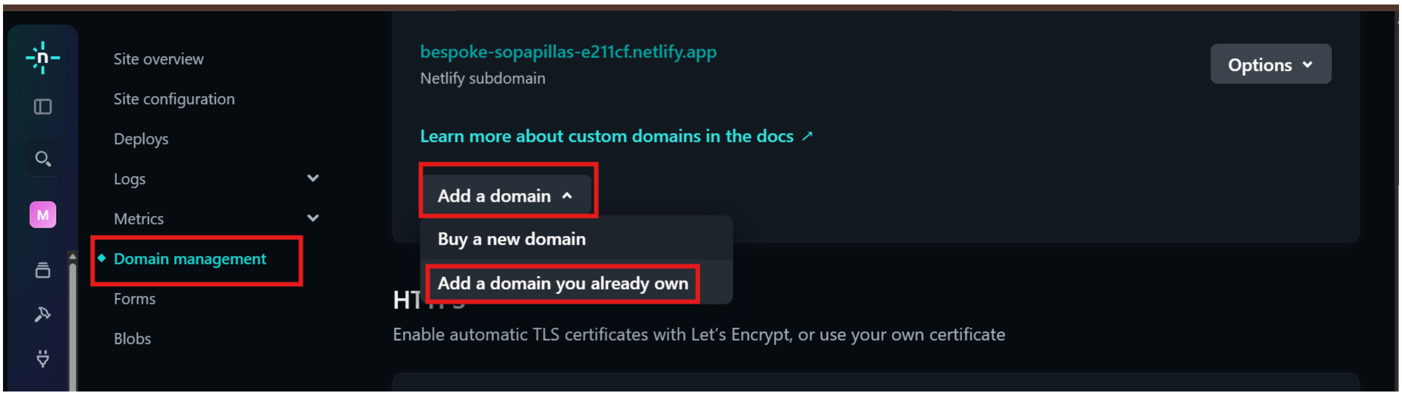
Manual deploys

Published at 2:46 PM.

- **Advanced Configurations**

Custom Domains & HTTPS

- **Add a Custom Domain:**
  In your site’s dashboard, navigate to **"Domain settings"** and add your custom domain. Follow Netlify’s instructions to update your DNS records.



Enable automatic TLS certificates with Let’s Encrypt, or use your own certificate

- **Enable HTTPS:(If Required)**
  Netlify automatically provisions a free SSL certificate via Let’s Encrypt. Simply enable HTTPS with one click.

- **Optimizing Your Site on Netlify**

- **Continuous Integration:**
  Benefit from automated deployments on every commit.
- **Performance Monitoring:**
  Use Netlify Analytics to monitor site traffic, performance, and uptime.
- **Redirects & Rewrites:**
  Use a _redirects file in your publish directory to manage URL routing and handle 404s.
- **Edge Handlers:**
  Optimize content delivery by modifying requests and responses at the network edge.