

**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

Automate static website deployment locally create a script that updates your server whenever changes are pushed.

Name: MADHUMETHRA AP Department: IT



**Introduction**

Automating static website deployment streamlines the process of updating your website. Locally, you'll use a static site generator (like Jekyll or Hugo) to build your site. A deployment script will then copy these built files to your server. Finally, to trigger this automatically, you'll set up a Git hook (on your Git server like GitHub or GitLab) that runs the deployment script whenever you push changes to your repository. This ensures your live website stays up-to-date with minimal effort

**Overview**

. \*Local Development & Build:\*

- Use a Static Site Generator (SSG) like Jekyll, Hugo, or Eleventy to create your website. SSGs convert Markdown, templates, and assets into HTML, CSS, and JavaScript.

- Develop your site locally, previewing changes with a local development server.

- Create a build script (e.g., build.sh) that runs the SSG's build command, generating the final website files.

2. \*Local Testing (Optional but Recommended):\*

- Before deploying to your live server, test the built website locally using a simple web server like python -m http.server or npx serve.

3. \*\*Deployment Script (deploy.sh):\*\*

- This script, run locally, automates the deployment process:

- Executes the build script to generate the website files.

- Uses rsync (or scp) to efficiently copy the built files to your web server via SSH. rsync is preferred for its efficiency in only transferring changed files.

- Optionally, SSHes into the server and restarts the web server (Nginx, Apache, etc.) if necessary.

4. \*Git Hooks (for Automatic Updates):\*

- On your Git server (GitHub, GitLab, etc.), configure a post-receive hook. This hook is triggered after a successful git push.

- The post-receive hook script on the Git server will:

- SSH into your deployment server.

- Execute the deploy.sh script on the deployment server.

5. \*Server Configuration:\*

- Ensure your web server is configured to serve the static website files from the directory you're deploying to.

**Objectives**

To automate the deployment of a static website locally, updating the server whenever changes are pushed.

**Step-by-Step Overview**

Step 1:

Initialize a Git Repository:

\* Create a new directory for your website project.

\* Open your terminal, navigate to the project directory, and initialize a Git repository:

git init

\* Create a Sample Website:

\* Add some basic HTML, CSS, and JavaScript files to the project directory.

\* For example:

\* index.html

\* style.css

\* script.js

Step 2:

Create a Deployment Script (e.g., deploy.sh)

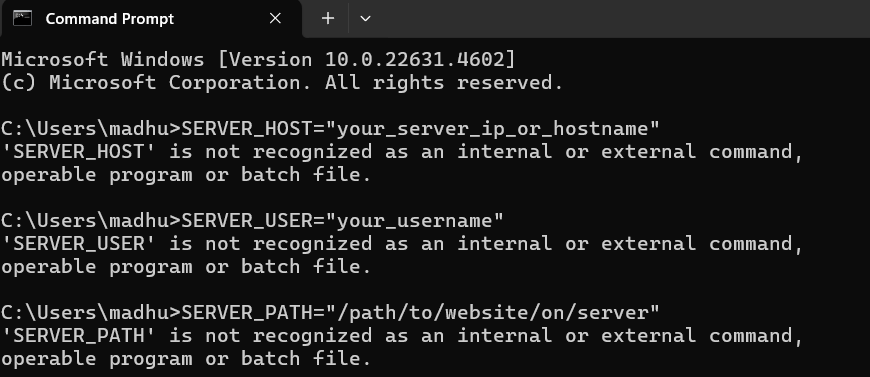
#!/bin/bash

# Replace with your server details

SERVER\_HOST="your\_server\_ip\_or\_hostname"

SERVER\_USER="your\_username"

SERVER\_PATH="/path/to/website/on/server"

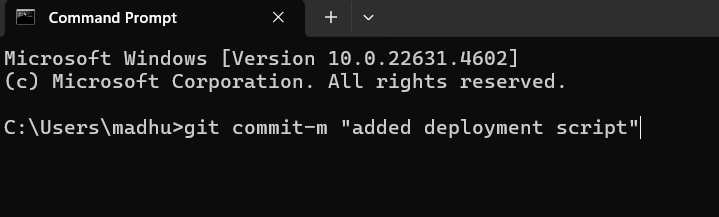


Step 3:

Add the Script to Your Git Repository

git add deploy.sh

git commit -m "Added deployment script"



Step 4:

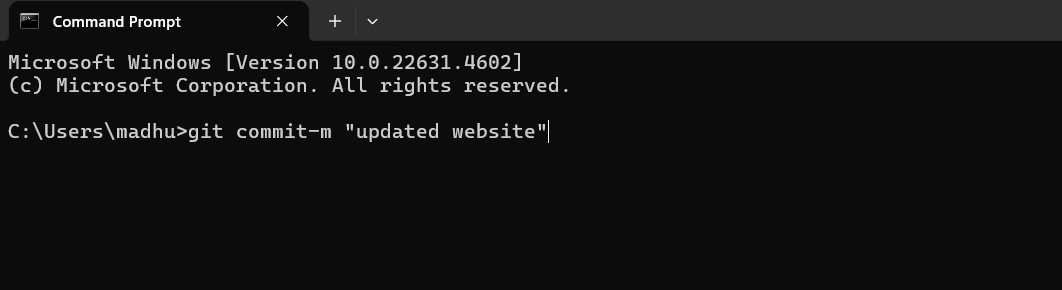
\* Make some changes to your website files.

\* Commit and push the changes to your Git repository:

git add .

git commit -m "Updated website"

git push origin main



**Output**

- A script (deploy.sh) automates the deployment of a static website.

- The script updates the server whenever changes are pushed.

- Streamlined deployment process

- Reduced manual effort and errors

- Faster time-to-market for updates and changes

- Integrate the script with a CI/CD pipeline for further automation.

- Explore options for deploying to a cloud platform or CDN.