 

**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: Nanthini R Department: ADS



**Introduction**

Manually updating a static website every time you make changes can be inefficient and prone to errors. Automating the deployment process ensures that the latest version of your website is always available on your local or remote server without requiring manual intervention.

This guide will walk you through creating a PowerShell script that:

1. Pulls the latest changes from a Git repository.
2. Copies the updated files to the web server directory.
3. Sets proper file permissions if required.
4. Restarts the web server if necessary.
5. Triggers the script automatically using a Git hook whenever changes are pushed.

By implementing this automation, you will save time and ensure that your website remains up to date with minimal effort.

**Overview**

Deploying a static website manually every time changes are made can be time-consuming and prone to mistakes. Automating the deployment process ensures that any updates pushed to a Git repository are immediately reflected on the local web server without requiring manual intervention.

This guide provides a step-by-step approach to setting up an automated deployment system using **PowerShell and Git hooks** on Windows. The process involves:

1. **Setting up the environment**, including Git for Windows and a web server like IIS or XAMPP.
2. **Creating a PowerShell script** to fetch the latest changes, copy files to the web directory, and restart the server if needed.
3. **Configuring a Git hook** that automatically triggers the deployment script whenever changes are pushed.
4. **Testing the setup** to ensure that updates are applied seamlessly.

By implementing this automation, developers can maintain an up-to-date static website with minimal effort, ensuring consistency and reducing downtime

**Objectives**

By the end of this POC, you will:

**1. Understand the Basics of Version Control**: Gain insight into the importance of Git for managing and tracking changes in your projects.

**2. Set Up a Git Repository**: Learn how to initialize a Git repository to version control your static website locally.

**3. Track Changes Effectively**: Understand how to stage and commit files to ensure every change is logged.

4. **Organize Your Project**: Maintain a clean and structured workflow for your static website, with the ability to roll back changes when needed.

5. **Prepare for Collaboration**: Lay the groundwork to share your repository and collaborate with others using Git when required

**Importance of Setting Up a Local Git Repository**

**Track Changes**: Git records all modifications, ensuring a clear history of your project.

**Rollback**: Easily revert to previous versions to recover from mistakes.

**Collaboration**: Prepares your project for team work, enabling smooth integration of changes.

**Step-by-Step Overview**

**Step 1: Set Up Your Environment**

Before we begin, make sure you have the following prerequisites installed and configured:

* **Git for Windows** installed and properly set up.
* **A local web server**, such as IIS (Internet Information Services) or XAMPP (Apache).
* **A Git repository** containing your static website files.
* **PowerShell enabled** for scripting.

Once you have these prerequisites ready, we can proceed with creating the deployment script.

**Step 2: Create the Deployment Script**

We will create a PowerShell script that automates the website deployment process. This script will pull the latest changes from Git, copy the files to the web server directory, and restart the web server if required.

**Step 3: Automate Deployment with a Git Hook**

To make this process fully automatic, we can configure Git to trigger the deployment script whenever you push new changes to the repository.

**Create a Post-Receive Hook**

1. **Navigate to the hooks directory** in your Git repository:

cd C:\path\to\your\repo\.git\hooks

1. **Create a new post-receive hook**:

New-Item -ItemType File -Path post-receive -Force

1. **Edit the file** with Notepad or a code editor:

notepad post-receive

1. **Add the following lines to the file**:

#!/bin/sh

powershell -ExecutionPolicy Bypass -File "C:/path/to/deploy.ps1"

1. **Save the file and grant execution permissions**:

icacls post-receive /grant Everyone:F

Now, every time you push changes to your repository, the deployment script will execute automatically.

**Step 4: Testing the Deployment**

To ensure that the automation works as expected, follow these steps:

1. **Make some changes** to your website files.
2. **Commit the changes** to your local Git repository:

git add .

git commit -m "Updated website content"

1. **Push the changes** to trigger the deployment script:

git push origin main

1. **Check the website directory** to confirm that the files have been updated.
2. **Review the deployment log** (C:\logs\deploy.log) to verify if any errors occurred.

If everything works correctly, your website should be updated automatically after each push.

**Expected Outcome**

Automating the deployment of a static website on Windows ensures that your latest changes are always live without requiring manual intervention. By setting up a PowerShell script and integrating it with a Git hook, you create a streamlined workflow where every push to the repository automatically updates the website on your server.

This process not only saves time but also reduces human errors, making website management more efficient. You can further enhance this automation by adding notifications, backup procedures, or additional security measures.