 

**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

Deploy Your Static Website Using GitHub Pages

Host your local Git repository’s static website directly using GitHub Pages.

Name: Nanthini R Department: ADS



**Introduction**

Hosting a static website can sometimes be challenging, requiring a web server and deployment setup. However, **GitHub Pages** provides a simple, free, and reliable way to host static websites directly from a Git repository. It is an excellent solution for personal projects, documentation sites, or portfolios.

In this guide, we will walk you through deploying a static website using GitHub Pages from a local Git repository on **Windows**. This method ensures that any updates you make locally can be pushed directly to GitHub and automatically reflected on your live website.

### ****Overview****

### GitHub Pages allows you to publish static websites directly from a repository. It supports HTML, CSS, JavaScript, and even static site generators like Jekyll. The process involves:

### Creating a GitHub repository to host the website.

### Preparing and pushing your local static website files to GitHub.

### Enabling GitHub Pages for hosting.

### Accessing your website using the provided GitHub Pages URL.

**Objectives**

**Understand GitHub Pages** – Learn what GitHub Pages is and how it can be used to host static websites for free.

**Set Up a GitHub Repository** – Create a repository to store and manage your website files efficiently.

**Initialize a Local Git Repository** – Configure Git on a Windows system to track and push website changes to GitHub.

**Push Static Website Files to GitHub** – Learn how to commit and push HTML, CSS, and JavaScript files to a remote repository.

**Enable and Configure GitHub Pages** – Activate GitHub Pages to host the website and generate a public URL.

**Access and Test the Live Website** – Verify that the website is correctly hosted on GitHub Pages and accessible through a browser.

**Manage Updates and Changes** – Understand how to update the website by pushing changes to GitHub and ensuring automatic deployment.

**Optimize for Future Enhancements** – Explore additional features such as custom domain integration, Jekyll for static site generation, and automation with GitHub Actions.

**Step-by-Step Overview**

**Step 1: Create a GitHub Repository**

1. Log in to GitHub and click on New Repository.
2. Give your repository a name (e.g., my-static-site).
3. Choose Public so GitHub Pages can host it.
4. Check the box "Initialize this repository with a README" (optional).
5. Click Create Repository.

**Step 2: Set Up Your Local Git Repository**

1. Open Git Bash on Windows.
2. Navigate to the directory where your website files are stored:

bash

cd C:/path/to/your/static-site

1. Initialize Git in your project folder:

bash

git init

1. Connect your local repository to the GitHub repository:

bash

git remote add origin https://github.com/your-username/my-static-site.git

**Step 3: Add and Push Your Website Files to GitHub**

1. Add all files to the repository:

bash

git add .

1. Commit the files with a meaningful message:

bash

git commit -m "Initial commit - Adding website files"

1. Push your files to GitHub:

bash

git branch -M main

git push -u origin main

**Step 4: Enable GitHub Pages**

1. Go to your GitHub repository on the GitHub website.
2. Click on Settings > Pages (found in the left sidebar).
3. Under Branch, select main (or the branch you used).
4. Click Save.
5. GitHub will generate a URL for your website (e.g., https://your-username.github.io/my-static-site/).

**Step 5: Test Your Live Website**

1. Open a web browser and visit the provided GitHub Pages URL.
2. If the website is not visible immediately, wait a few minutes as GitHub processes the deployment.
3. If needed, push further updates using:

bash

git add .

git commit -m "Updated website content"

git push origin main

**Outcome**

Deploying a static website using GitHub Pages on Windows provides a simple, efficient, and cost-effective way to host web projects without the need for complex server configurations. By following this process, you have successfully set up a GitHub repository, pushed your static website files, and enabled GitHub Pages for public hosting.

With this setup, any updates you make to your local website can be easily pushed to GitHub, ensuring that your site remains up to date with minimal effort. Additionally, GitHub Pages supports further enhancements such as custom domains, Jekyll integration, and automation with GitHub Actions, allowing you to expand your site's capabilities.