 

**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

**Deploy Your Static Website Using GitHub Pages**

**Name: Monika K Department:ADS**



**Introduction and Overview**

In this POC, we will learn how to deploy a static website using GitHub Pages. This process involves setting up a GitHub repository, pushing website files, and enabling GitHub Pages for hosting. By following these steps, you’ll gain hands-on experience in deploying websites using a simple and free hosting solution.

**Objective**

**The goal of this project is to:**

1. Set up a GitHub repository for website files.
2. Push static website files to the repository.
3. Enable GitHub Pages for hosting and deployment.

**Importance of Using GitHub Pages**

**Deploying with GitHub Pages ensures:**

* Free Hosting: No need for external hosting services.
* Version Control: Keeps track of changes through Git.
* Easy Updates: Deploy changes instantly by pushing updates to the repository.

Step-by-Step Overview

Step 1: Create a GitHub Repository

1. Go to [GitHub](https://github.com/) and log in.
2. Click on “New” to create a repository.
3. Name the repository (e.g., my-static-site).
4. Select “Public” and check “Add a README” if needed.
5. Click “Create repository.”

Step 2: Add and Push Website Files

1. Clone the repository to your local machine:

git clone https://github.com/your-username/my-static-site.git

1. Copy your static website files (HTML, CSS, JS) into the repository folder.
2. Add and commit the changes:

git add .

git commit -m "Initial commit for static website"

1. Push the files to GitHub:

git push origin main

Step 3: Enable GitHub Pages

1. Go to the repository on GitHub.
2. Click on “Settings.”
3. Scroll down to the “Pages” section.
4. Under “Branch,” select “main” (or your default branch) and click “Save.”
5. GitHub will generate a live URL for your website (e.g., https://your-username.github.io/my-static-site/).

Step 4: Access Your Deployed Website After enabling GitHub Pages, wait a few minutes and visit the generated URL to see your website live.

Step 5: Automate Deployment with GitHub Actions (Optional) To automate deployments when you push updates, create a .github/workflows/deploy.yml file with the following content:

name: Deploy Website

on:

push:

branches:

- main

jobs:

deploy:

runs-on: ubuntu-latest

steps:

- name: Checkout repository

uses: actions/checkout@v2

- name: Deploy to GitHub Pages

uses: peaceiris/actions-gh-pages@v3

with:

github\_token: ${{ secrets.GITHUB\_TOKEN }}

publish\_dir: ./

This ensures your website is automatically deployed whenever you push updates