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: Keshika D Department: ADS

St. JOSEPH'S COLLEGE OF ENGINEERING | St. JOSEPH'S INSTITUTE OF TECHNOLOGY

AUTONOMOUS INSTITUTIONS, AFFILIATED TO ANNA UNIVERSITY

Introduction

GitHub Pages is a static site hosting service designed to publish your projects directly from a GitHub repository. It allows developers to showcase their work, create personal websites, or host documentation in an efficient, free, and straightforward way.

Overview

This project demonstrates how to deploy a static website using GitHub Pages. Starting with the basics of setting up a GitHub repository, we'll explore each step required to host a functional static website. This includes initializing a Git repository, pushing files to GitHub, and configuring GitHub Pages for deployment.

Key Features of GitHub Pages:

Free hosting for public repositories.

Support for static files (HTML, CSS, JavaScript).

Easy integration with version control through Git.

Objectives

1. Learn the fundamentals of GitHub Pages and its deployment process.

2. Understand the importance of static website hosting and its use cases.
3. Gain hands-on experience in using Git and GitHub for project versioning and hosting.

4. Successfully publish a static website and make it publicly accessible.

Importance of Hosting with GitHub Pages

1. Cost-effective: Free for public repositories, making it accessible for students and developers.

2. Version Control: Seamlessly integrates with GitHub, enabling easy updates and collaboration.

3. Visibility: A great way to showcase personal portfolios, projects, or documentation.

4. Ease of Use: Minimal setup required compared to other hosting platforms.

5. Custom Domains: Option to configure custom domains, enhancing the professional appeal of your website.

Step-by-Step Overview

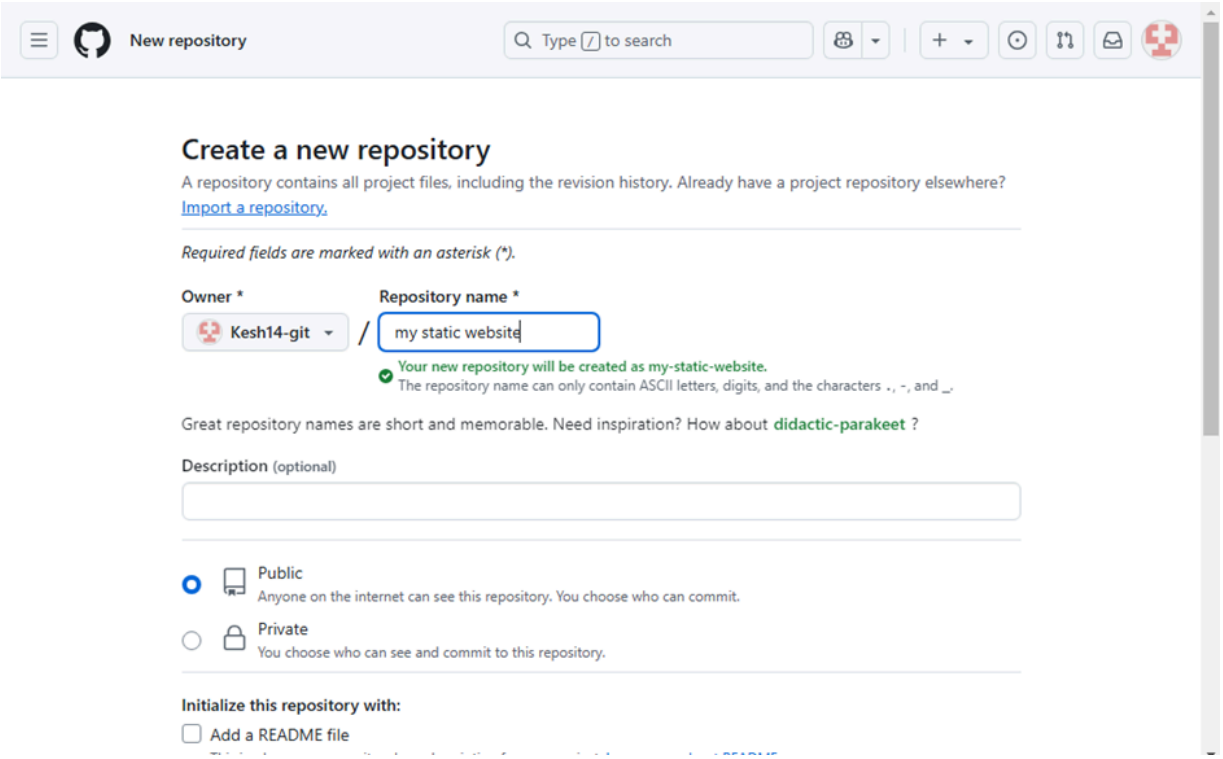
Step 1:

Create a New Repository:

Once you're logged in, click the green **"New"** button on the top right of your GitHub homepage to create a new repository.

Give your repository a name, for example, my-static-website.

Leave the other settings as default, and click **"Create repository"**.



Step 2:

Create a folder (e.g., my-static-website) where you'll keep all your website files.

Inside that folder, create the main file for your website, called **index.html**.

Here's a simple example of what to put in your index.html:

Index - Notepad

File Edit Format View Help

```
<!DOCTYPE html>
<html>
<head>
     <title>My Static Website/title>
</head>
<body>
     <h1>Welcome to My Website!</h1>
     <p>This is a static website hosted on Github Pages.</p>
</body>
</html>
```

Step 3:

Open **Command Prompt** and navigate to the folder where your index.html file is saved.

Use the cd command to navigate.

C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19045.5371]
(c) Microsoft Corporation. All rights reserved.

C:\Users\TEMP.DESKTOP-S0M6S0K.007>E:

E:\>cd E:\Keshika\my static website
```

Step 4:

Initialize a Git repository by running:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.5371]
(c) Microsoft Corporation. All rights reserved.

C:\Users\TEMP.DESKTOP-S0M6S0K.007>E:

E:\>cd E:\Keshika\my static website

E:\Keshika\my static website>git init
Initialized empty Git repository in E:/Keshika/my static website/.git/
```

Step 5:

Add your website files to the repository:

```
E:\Keshika\my static website>git add .
```

Step 6:

Save the changes in Git with a commit message:

```
E:\Keshika\my static website>git add .

E:\Keshika\my static website>git commit -m "Initial commit"
[master (root-commit) a2b59ff] Initial commit
 1 file changed, 10 insertions(+)
 create mode 100644 Index.html

E:\Keshika\my static website>
```

Step 7:

Go to your GitHub repository (the one you created

earlier). Copy the **repository URL**:

In your Command Prompt, link your local repository to the GitHub repository:

```
E:\Keshika\my static website>git remote add origin https://github.com/Kesh14-git/my-static-website
```

Step 8:

Push your files to GitHub:

```
E:\Keshika\my static website>git branch -M main

E:\Keshika\my static website>git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 346 bytes | 173.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
```

Step 9:

Enable GitHub Pages

1. Go to your repository on GitHub.

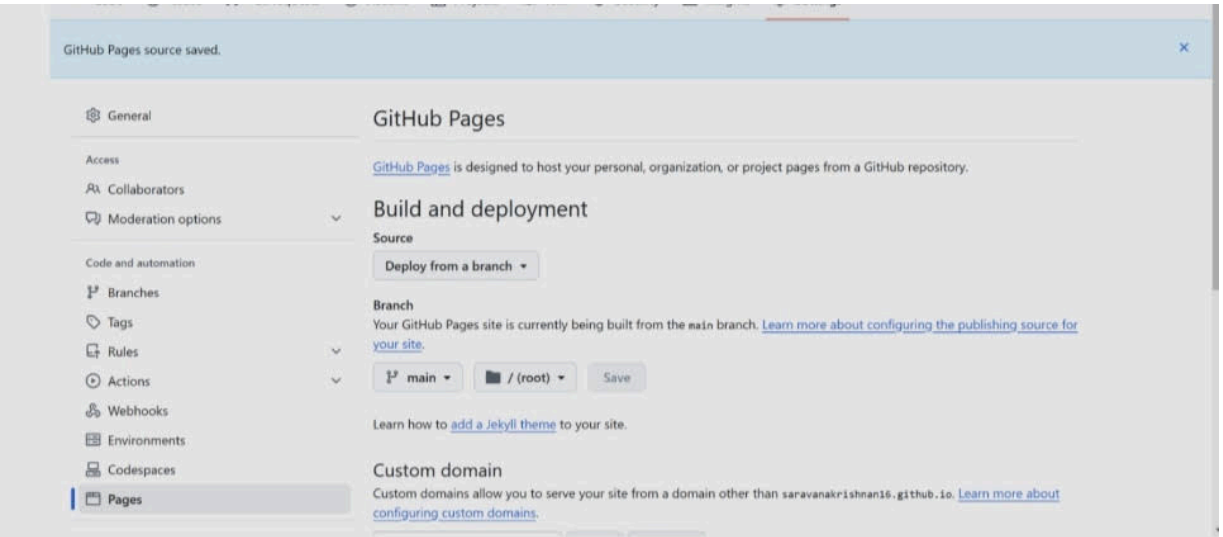
2. Click on the **Settings** tab (it's near the top, next to Code, Issues, etc.).

3. Scroll down to the **Pages** section (on the left menu, under "Code and automation").
4. Under **Source**, select:

   - **Branch**: main

- **Folder**: / (root)

5. Click **Save**.



Step 10:

Access Your Website

Wait a few minutes for GitHub Pages to deploy your site.

Visit your website at:

https://<your-username>.github.io/<your-repository>

Outcome

By completing this PoC of deploying a static website using GitHub Pages, you will:

1. Successfully create and configure a GitHub repository for your project.

2. Initialize a Git repository in your local project folder and link it to GitHub.

3. Upload your static website files (HTML, CSS, JavaScript) to GitHub.

4. Enable GitHub Pages in the repository settings to host your static website.

5. Access your static website live on the web via a GitHub Pages URL.

6. Gain hands-on experience with Git commands like git init, git add, git commit, git remote add, and git push.

7. Understand the process of hosting a static site for free using GitHub Pages.