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: KEERTHIKA ANANTHAN
Department: AML

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

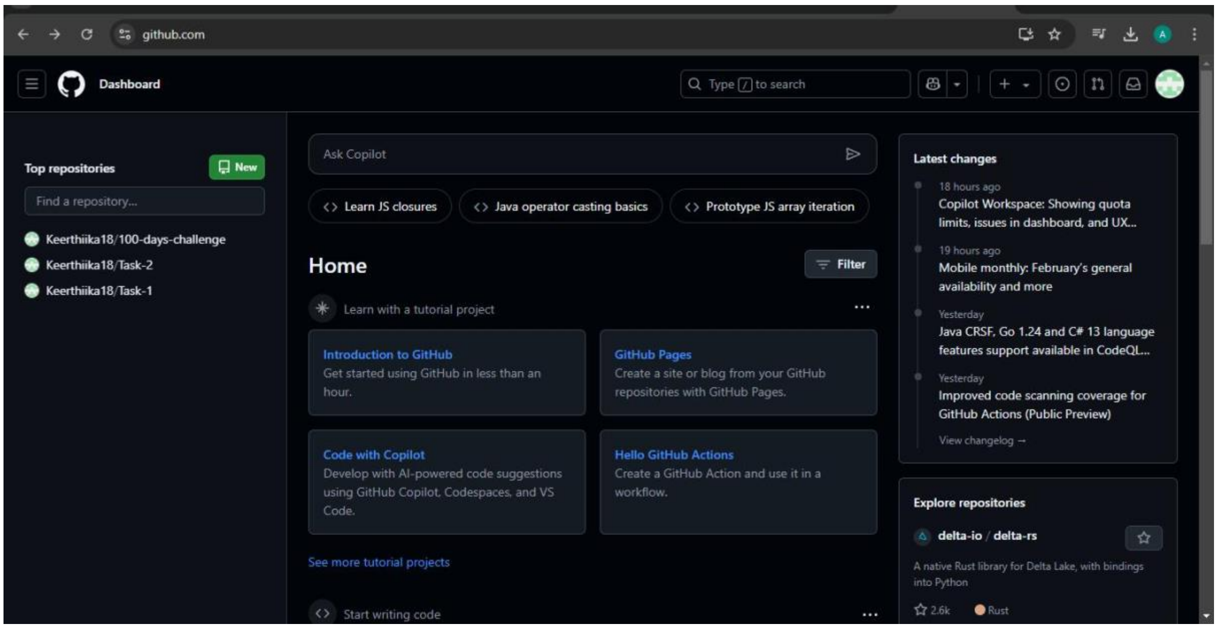
Step 1:

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"**.



Introduction to GitHub
Get started using GitHub in less than an hour.

GitHub Pages
Create a site or blog from your GitHub repositories with GitHub Pages.

Code with Copilot
Develop with AI-powered code suggestions using GitHub Copilot, Codespaces, and VS Code.

Hello GitHub Actions
Create a GitHub Action and use it in a workflow.

18 hours ago
Copilot Workspace: Showing quota limits, issues in dashboard, and UX...

19 hours ago
Mobile monthly: February's general availability and more

Yesterday
Java CRSF, Go 1.24 and C# 13 language features support available in CodeQL...

Yesterday
Improved code scanning coverage for GitHub Actions (Public Preview)

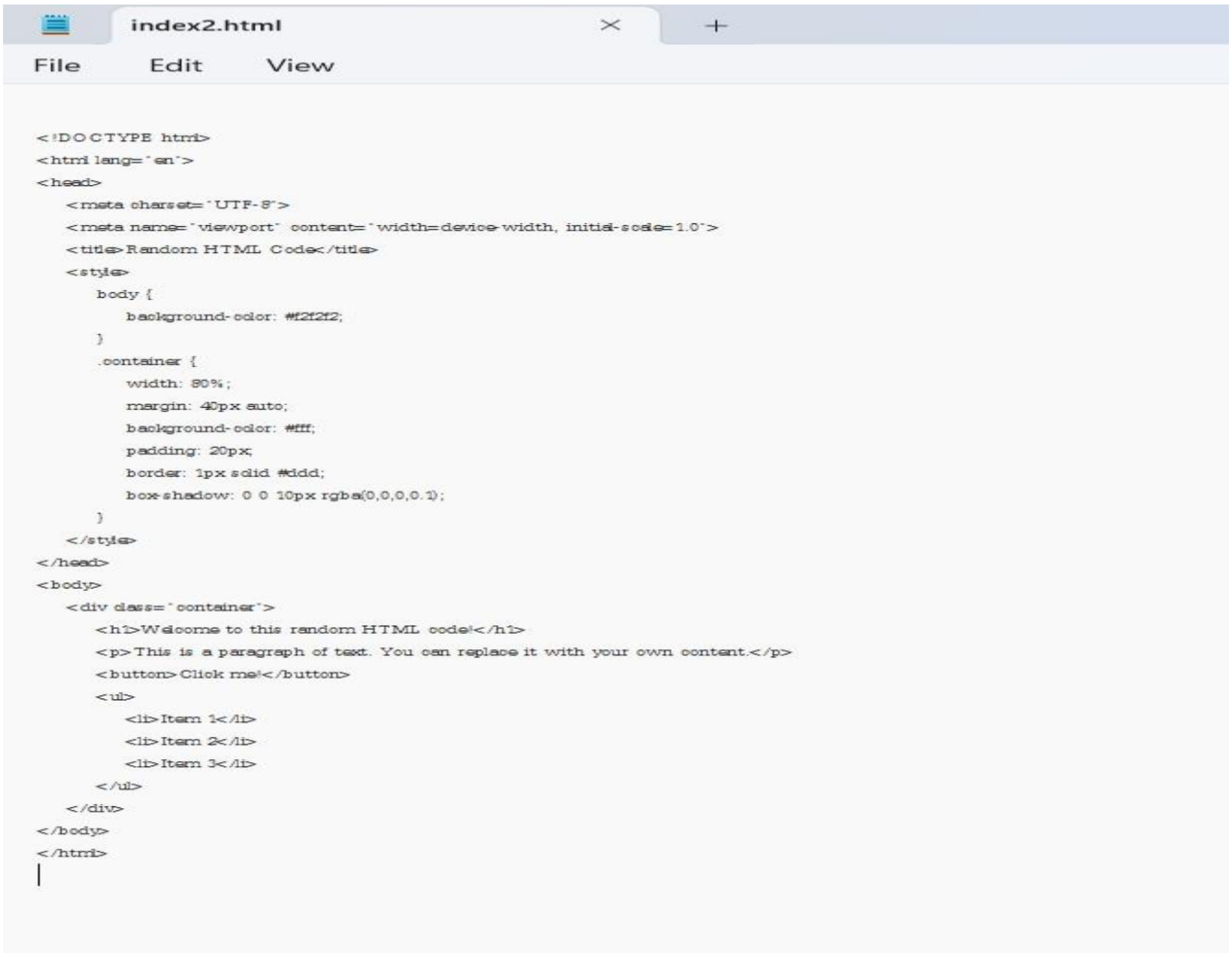
A native Rust library for Delta Lake, with bindings into Python

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:

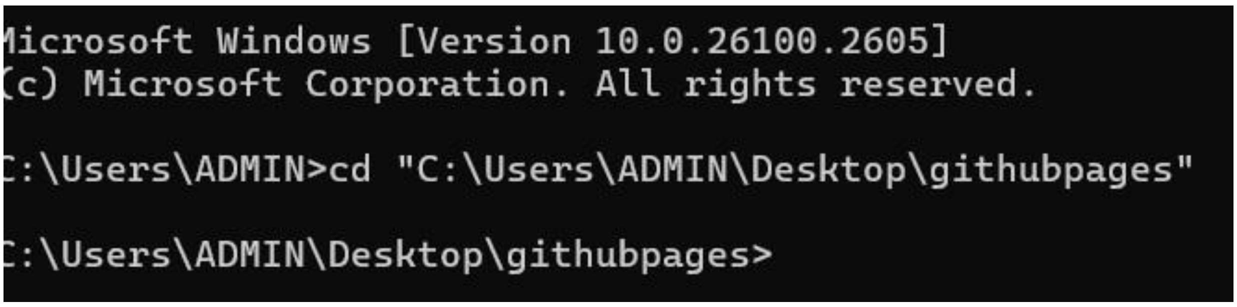


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Random HTML Code</title>
    <style>
        body {
            background-color: #f2f2f2;
        }
        .container {
            width: 80%;
            margin: 40px auto;
            background-color: #fff;
            padding: 20px;
            border: 1px solid #ddd;
            box-shadow: 0 0 10px rgba(0,0,0,0.1);
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Welcome to this random HTML code!</h1>
        <p>This is a paragraph of text. You can replace it with your own content.</p>
        <button>Click me!</button>
        <ul>
            <li>Item 1</li>
            <li>Item 2</li>
            <li>Item 3</li>
        </ul>
    </div>
</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.



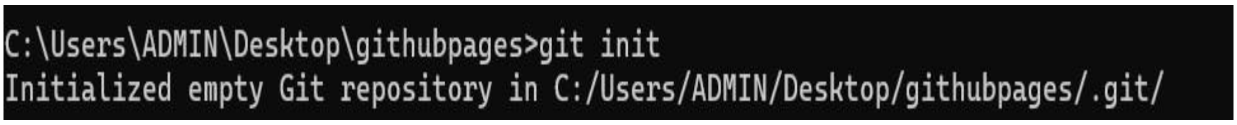
```
licrosoft Windows [Version 10.0.26100.2605]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ADMIN>cd "C:\Users\ADMIN\Desktop\githubpages"

C:\Users\ADMIN\Desktop\githubpages>
```

Step 4:

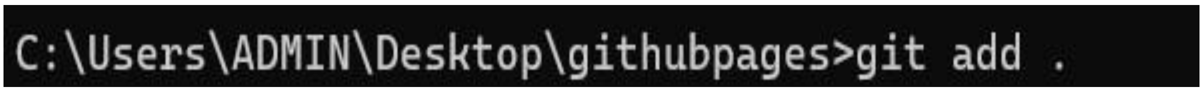
Initialize a Git repository by running:



```
C:\Users\ADMIN\Desktop\githubpages>git init
Initialized empty Git repository in C:/Users/ADMIN/Desktop/githubpages/.git/
```

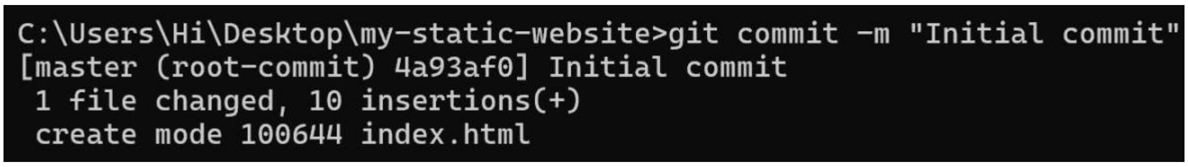
Step 5:

Add your website files to the repository:



```
C:\Users\ADMIN\Desktop\githubpages>git add .
```

Step 6:



```
C:\Users\Hi\Desktop\my-static-website>git commit -m "Initial commit"
[master (root-commit) 4a93af0] Initial commit
 1 file changed, 10 insertions(+)
 create mode 100644 index.html
```

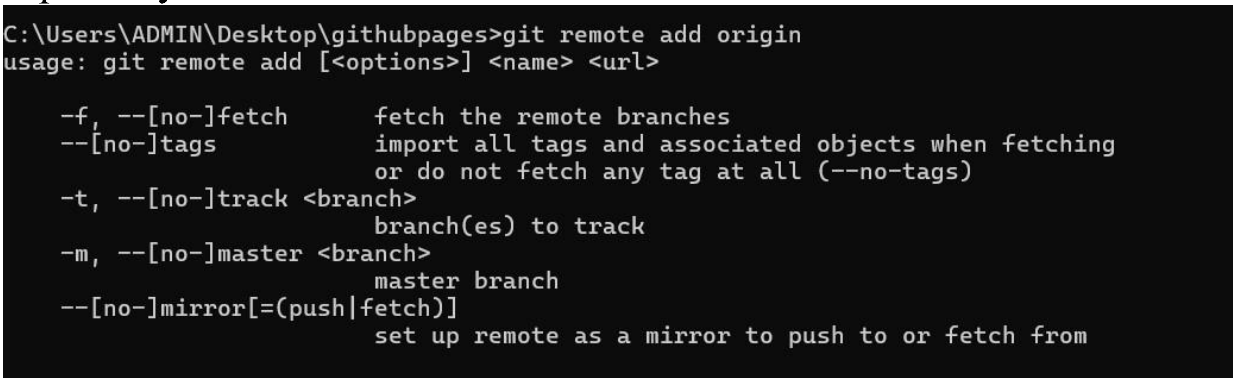
Save the changes in Git with a commit message:

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:



```
C:\Users\ADMIN\Desktop\githubpages>git remote add origin
usage: git remote add [<options>] <name> <url>

    -f, --[no-]fetch      fetch the remote branches
    --[no-]tags           import all tags and associated objects when fetching
                          or do not fetch any tag at all (--no-tags)
    -t, --[no-]track <branch>
                          branch(es) to track
    -m, --[no-]master <branch>
                          master branch
    --[no-]mirror[=(push|fetch)]
                          set up remote as a mirror to push to or fetch from
```

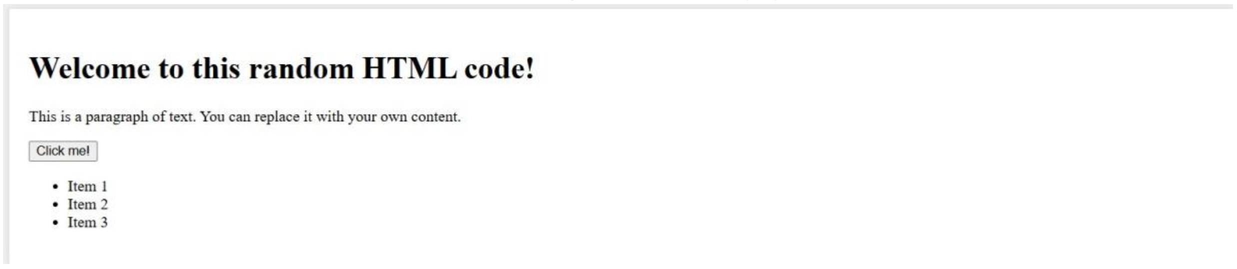
Step 8:

Push your files to GitHub

Step 9:

Access Your Website

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



Welcome to this random HTML code!

This is a paragraph of text. You can replace it with your own content.

- Item 1
- Item 2
- Item 3

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.