

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: CHELSIAH M

Department: CSE

Introduction

GitHub Pages is a free and straightforward hosting service provided by GitHub for static websites. By leveraging this feature, you can host personal, project, or organization websites directly from a GitHub repository. This guide walks you through the process of deploying your static website using GitHub Pages.

Objectives

By the end of this POC, you will:

1. Understand the basics of GitHub Pages and how it hosts static websites.
2. Learn to configure a GitHub repository for website hosting.
3. Successfully deploy a static website using GitHub Pages.

Importance of deploy your static website using Github pages

Free Hosting – GitHub Pages provides a **cost-free** solution for hosting static websites, eliminating the need for paid hosting services.

Easy Deployment – Websites can be deployed **directly from a GitHub repository** without needing complex configurations or additional tools.

Automatic Updates – Any changes pushed to the repository are **automatically updated** on the live website, ensuring seamless maintenance.

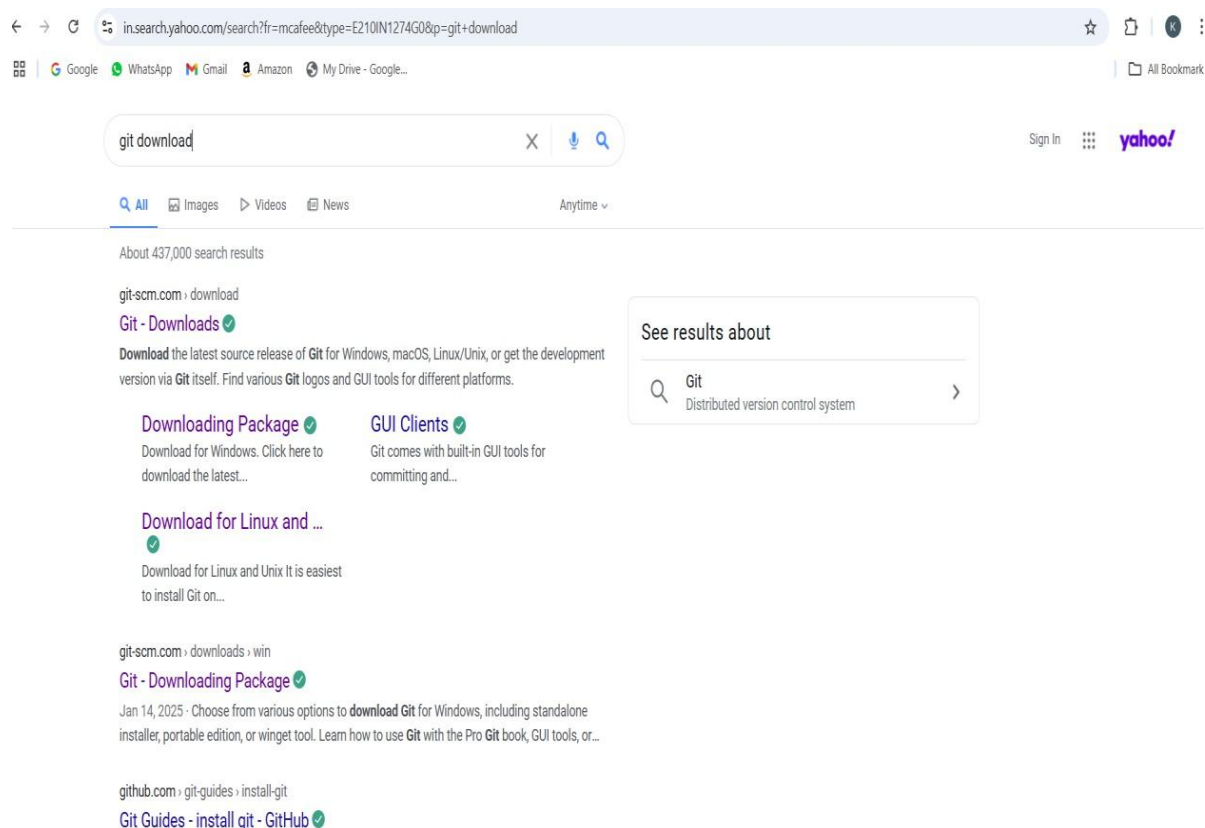
Version Control – Since the website is hosted on GitHub, you get **built-in version control**, allowing easy tracking of changes and rollbacks if needed.

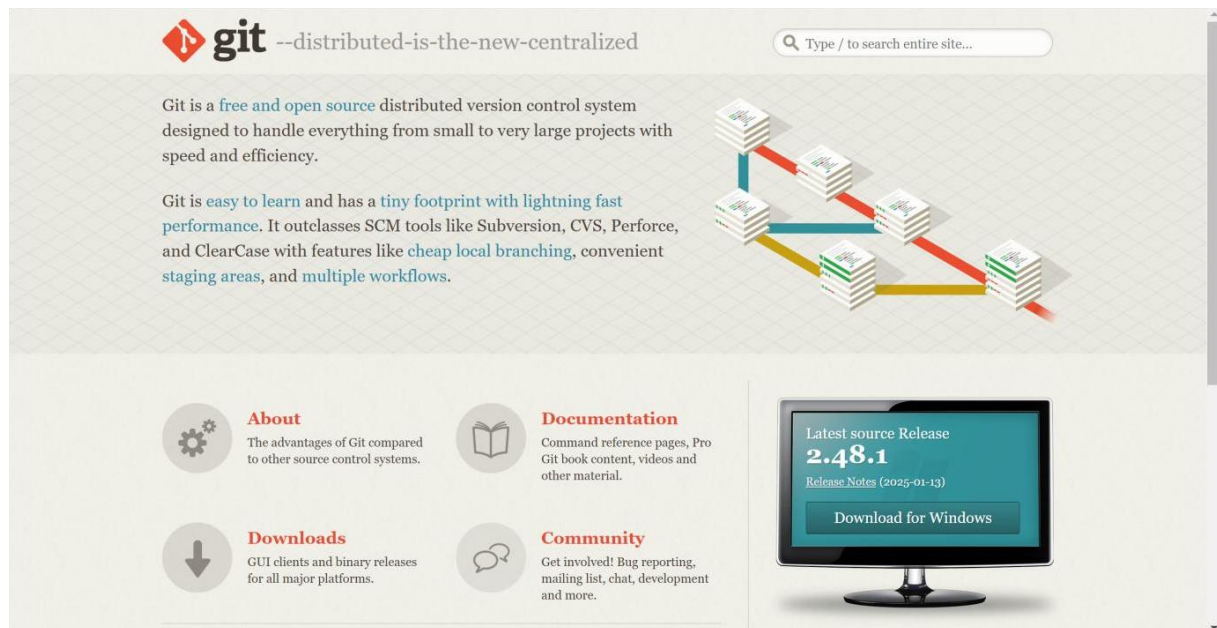
Custom Domain Support – GitHub Pages allows you to configure a **custom domain** for a professional web presence with proper branding.

Step-by-Step Overview

Step 1:

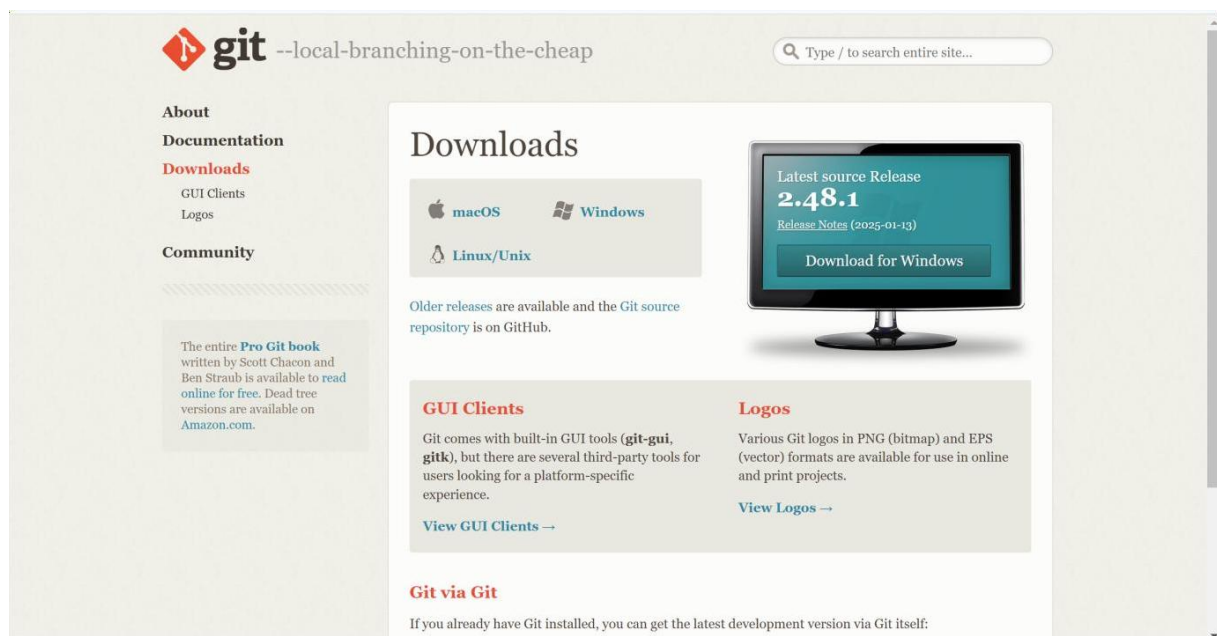
Search for "Git" in Chrome, download it, and click the "Downloads" option on the website.





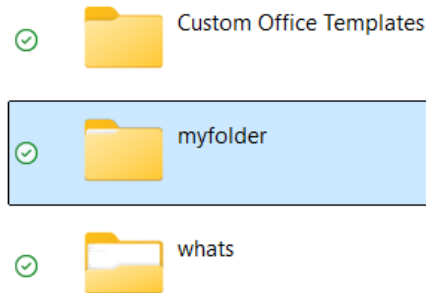
Step 2:

Click the **Windows** option on the download page and follow the installation wizard.



Step 3:

First, open your terminal or command prompt and create a new folder for your project. Let's name it myfolder



Step 4:

Open VS code, create a simple HTML file named **myweb.html**. You can write some basic HTML

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Simple HTML</title>
7
8  </head>
9  <body>
10     <div class="container">
11         <h1>Welcome to My Simple Webpage</h1>
12         <p>This is a basic example of HTML.</p>
13         <button class="btn">Click Me</button>
14     </div>
15 </body>
16 </html>
```

Step 5:

Output for the HTML code



Step 6:

Open the **VS code Terminal** and set the path to the folder named that we created.

Git clone

Basic syntax for git clone is **git clone <repository_url>**

- ☐ Find the repository URL on GitHub (or another Git hosting service).
- ☐ Run the command:

```
PS C:\Users\kirth\OneDrive\ドキュメント\gitweb> git clone https://github.com/kirthika-05/sampletest.git
Cloning into 'sampletest'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```

Step 7:

The command **cd website/** changes the directory to the website folder, ensuring you are in the correct project location for deployment. This allows you to run Git commands like **git init**, **git add .**, and **git push** to deploy your static website using GitHub Pages.

Step 8:

Now, initialize Git by typing this command:

git init

This command will create a .git folder inside your project folder, which tells Git to start tracking your files.

Next, we need to tell Git to start tracking your website files.

To tell Git which files to track, use the git add command. If you want to track all the files in your folder, type

git add .

Basic syntax is **git add newrepo.html** This command adds all the files to Git's tracking system.

git commit

Basic syntax is **git commit -m "Your commit message"**

A **commit** is a snapshot of the project's current state. When you commit, Git records changes to tracked files and saves them in the repository history.

- **git commit**: Saves changes in the local repository.
- **-m "message"**: Adds a message describing what the commit does.

```
PS C:\Users\kirth\OneDrive\トキユメント\myfolder\website> git init
Reinitialized existing Git repository in C:/Users/kirth/OneDrive/トキユメント/myfolder/website/.git/
PS C:\Users\kirth\OneDrive\トキユメント\myfolder\website> git add .
PS C:\Users\kirth\OneDrive\トキユメント\myfolder\website> git commit -m "deploy"
[main (root-commit) 10cc457] deploy
1 file changed, 16 insertions(+)
create mode 100644 myweb.html
```

Step 9:

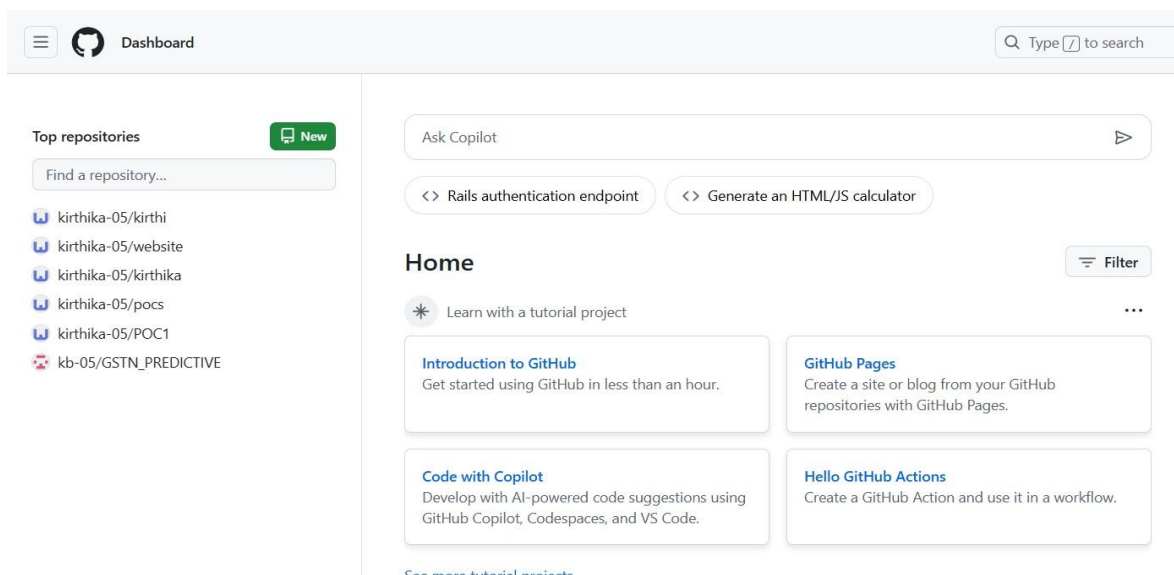
Set Up Your Name and Email Globally Git doesn't know who is making the commit because you haven't configured your name and email yet. Git uses this information to track who made the changes.

```
PS C:\Users\kirth\OneDrive\ドキュメント\mynew> git config --global user.name "kirthikasekar"
PS C:\Users\kirth\OneDrive\ドキュメント\mynew> git config --global user.email "kirthikasekar288@gmail.com"
```

Step 10:

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.




Step 11:

Give your repository a name, for example, test and click "**Create repository**".

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *	Repository name *
 kirthika-05	/ websit
	✔ websit is available.

Great repository names are short and memorable. Need inspiration? How about [solid-funicular](#) ?

Description (optional)

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

- ☐ Add a README file
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

Step 12:

Add the Remote Repository URL to Your Local Repository:

Go back to your Command Line and type the following:

```
git remote add origin https://github.com/yourusername/my-website.git
```

Replace your username with your GitHub username and with the name of your GitHub repository

Step 13:

git branch -V & **git branch -M**

The main command is used to **rename the current branch** to main. Here's what it does:

-M: This flag forces the renaming, even if a branch named main already exists. It will overwrite the existing main branch.

main: This is the new name for the current branch.

Step 14:

git push -u origin main

The command is used to push your local **main** branch to the remote repository (**origin**) and set it as the upstream branch

```
PS C:\Users\kirth\OneDrive\ドキュメント\myfolder\website> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 457 bytes | 457.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/kirthika-05/website.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Step 15:

Verify Your Files on GitHub

Go to your GitHub Repository:

Open your web browser and navigate to your GitHub repository (e.g., <https://github.com/yourusername/my-website>).

Branch

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more about configuring the publishing source for your site.](#)

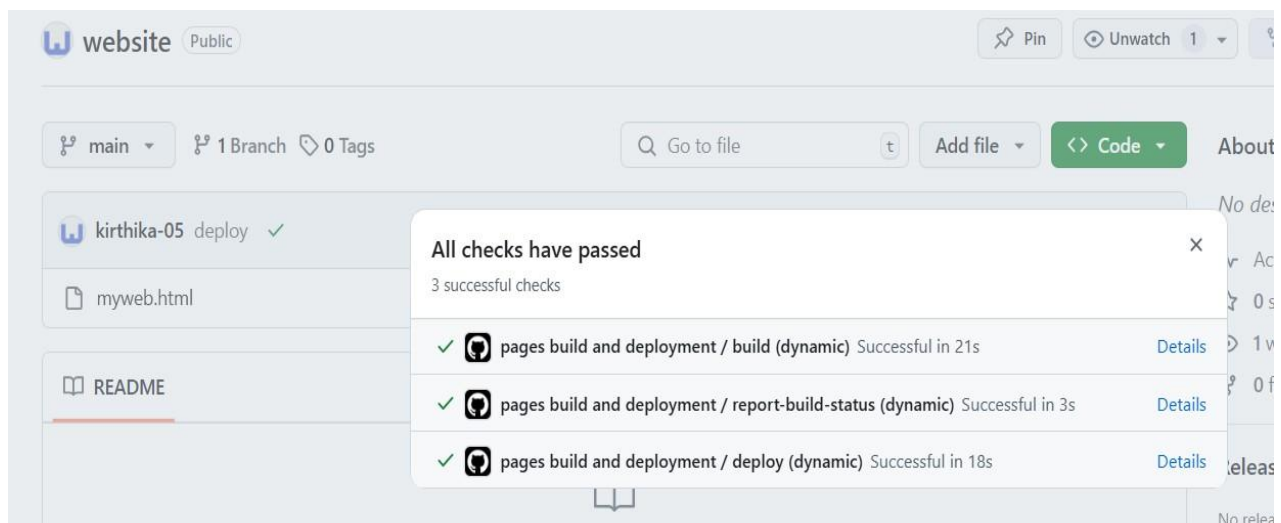
 main ▾

 / (root) ▾

Save

Step 16:

With GitHub Pages, updates are automatic whenever you push changes to the repository, the site is instantly redeployed, ensuring seamless version control. Your Website is **deployed**



Outcome

By completing this PoC, you will have:

Deploying a static website using **GitHub Pages** is an efficient, cost-free, and beginner-friendly method for hosting web projects. It simplifies website management, enables **automatic updates**, and enhances collaboration while providing secure and reliable hosting.