Lesson:

Github

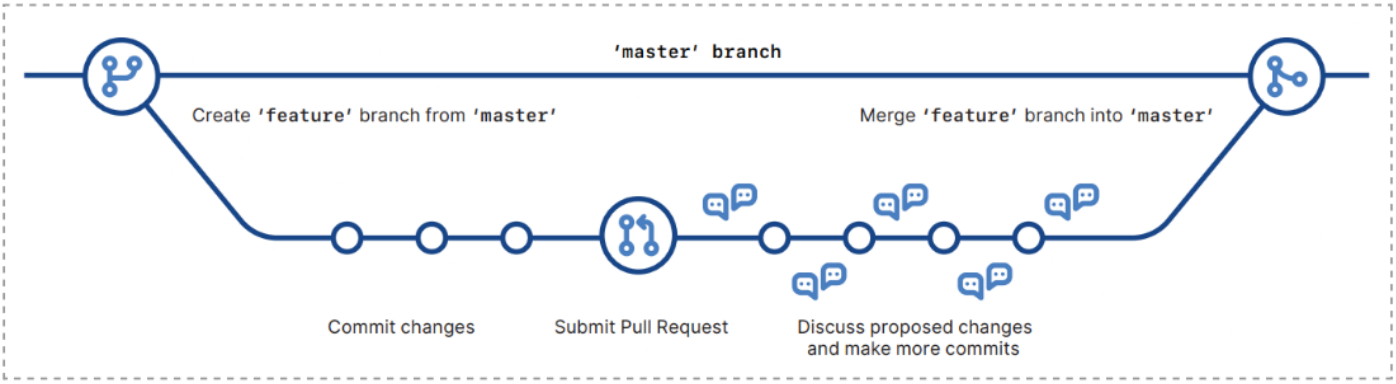
PW SKILLS

Topics

- Introduction to GitHub
- Why use Github?
- Setting up a GitHub account
- Linking the GitHub account VS code
- Creating the repo
- Pushing code to GitHub
- Forking an existing repo
- Generating pull requests and merge requests
- Handling merge conflict
- Hosting a website on GitHub pages
- Alternative of GitHub

Introduction to GitHub

GitHub is a platform where you can upload a copy of your Git repository (often shortened as repo), hosted either on github.com. More than just uploading your Git repositories it allows you to collaborate much more easily with other people on your projects. It does that by providing a centralized location to share the repository, a web-based interface to view it, and features like forking, Pull requests, Issues, Projects, and GitHub Wikis that allow you to specify, discuss, and review changes with your team more effectively.
GitHub Flow -



Why use Github?

GitHub is much more than just a place to store your Git repositories. It provides a number of additional benefits, including the ability to the following as shown below -

1. Version Control - GitHub is built on top of Git, a powerful distributed version control system. It allows developers to track changes, collaborate with others, and manage project versions effectively.

2. Document requirements - Using issues, you can either document bugs or specify new features that you'd like to have your team develop.

3. Collaborate on independent streams of history - Using branches and pull requests, you can collaborate on different branches or features

4. Review work in progress - By looking at the list of pull requests, you can see all of the different features that are currently being worked on, by clicking any given pull request you see the latest changes and all of the discussions about the changes, check the status of integration like a Continuous Integration (CI) server, or even add your own review to approve changes before they are accepted

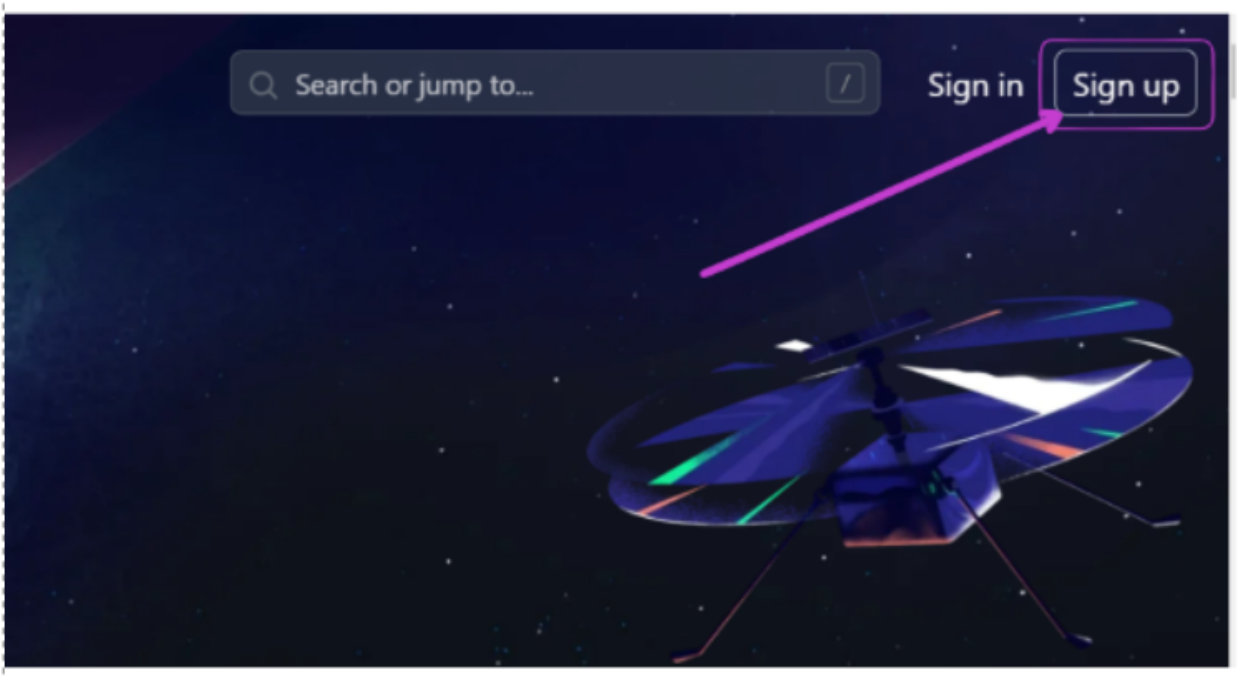
5. See team progress - Looking through the commit history allows you to see what the team has been working on.

6. Public and Private Repositories - GitHub supports both public and private repositories. Public repositories encourage open collaboration and sharing, while private repositories offer secure spaces for proprietary or sensitive projects.

Setting up a GitHub account

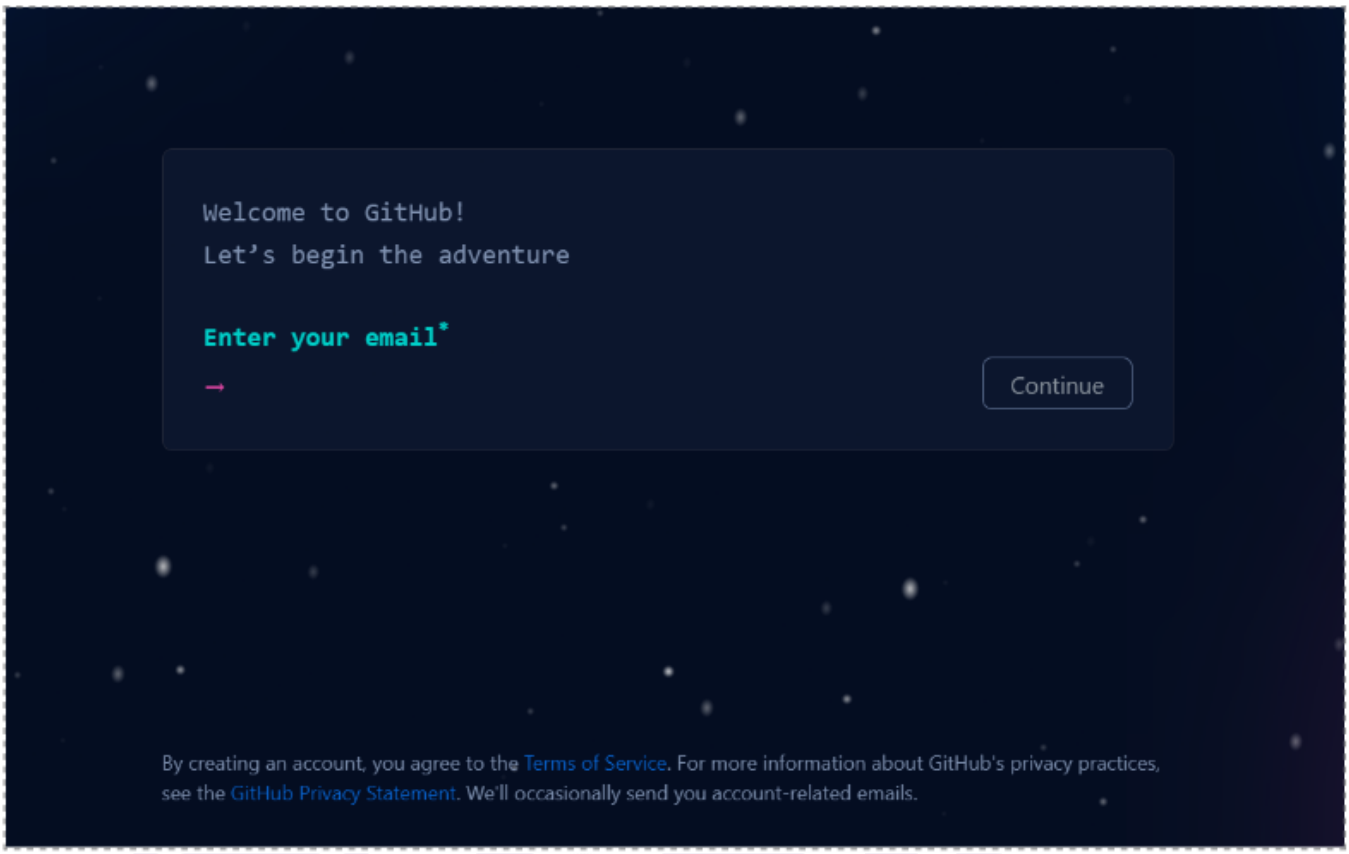
To set up a GitHub account is a simple process, Here's a step-by-step guide to help you create your GitHub account

1. Visit the official GitHub website
Open your web browser and go to the official website i.e. **https://github.com/**



2. Signup
After opening the official GitHub website, click on the signup button in the top right corner.

3. Complete the sign-up Form
Fill in the required information details in the signup form and click “Create an account



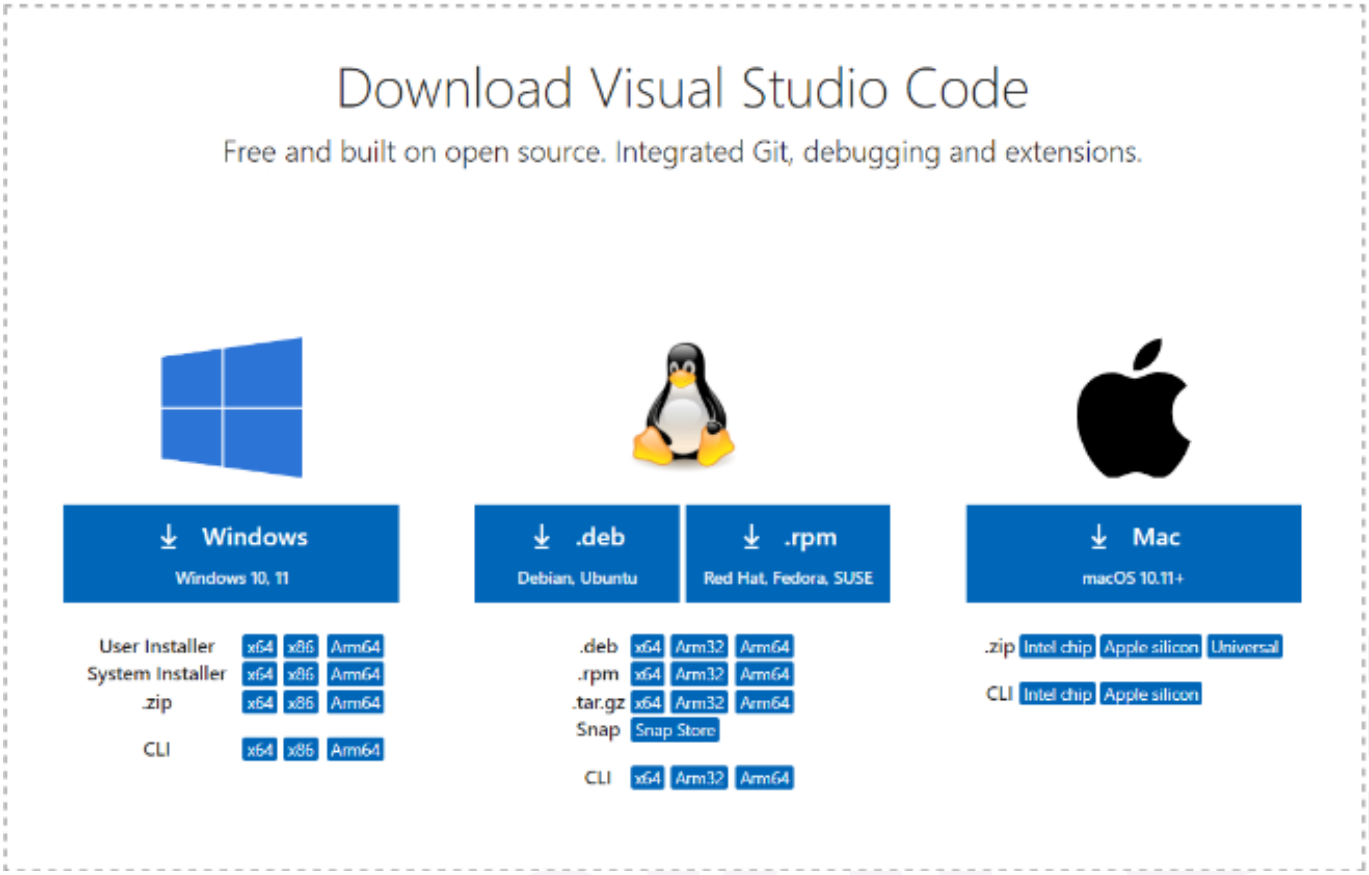
4. Verify Your email address
GitHub will send a verification email to the email address you provided. Open the email and click on the verification link to confirm your email.

5. Set up your Account
GitHub will guide you through a few onboarding steps to personalize your experience. You can choose your preferences, select your areas of interest, and set up a few introductory options.

6. Welcome to GitHub
After completing the onboarding process, you'll be taken to your GitHub dashboard, where you can start using GitHub

Linking the GitHub account VS code

1. Download and install the Visual Studio code from **https://code.visualstudio.com/Download,** click on download, and choose the right version for your operating system.

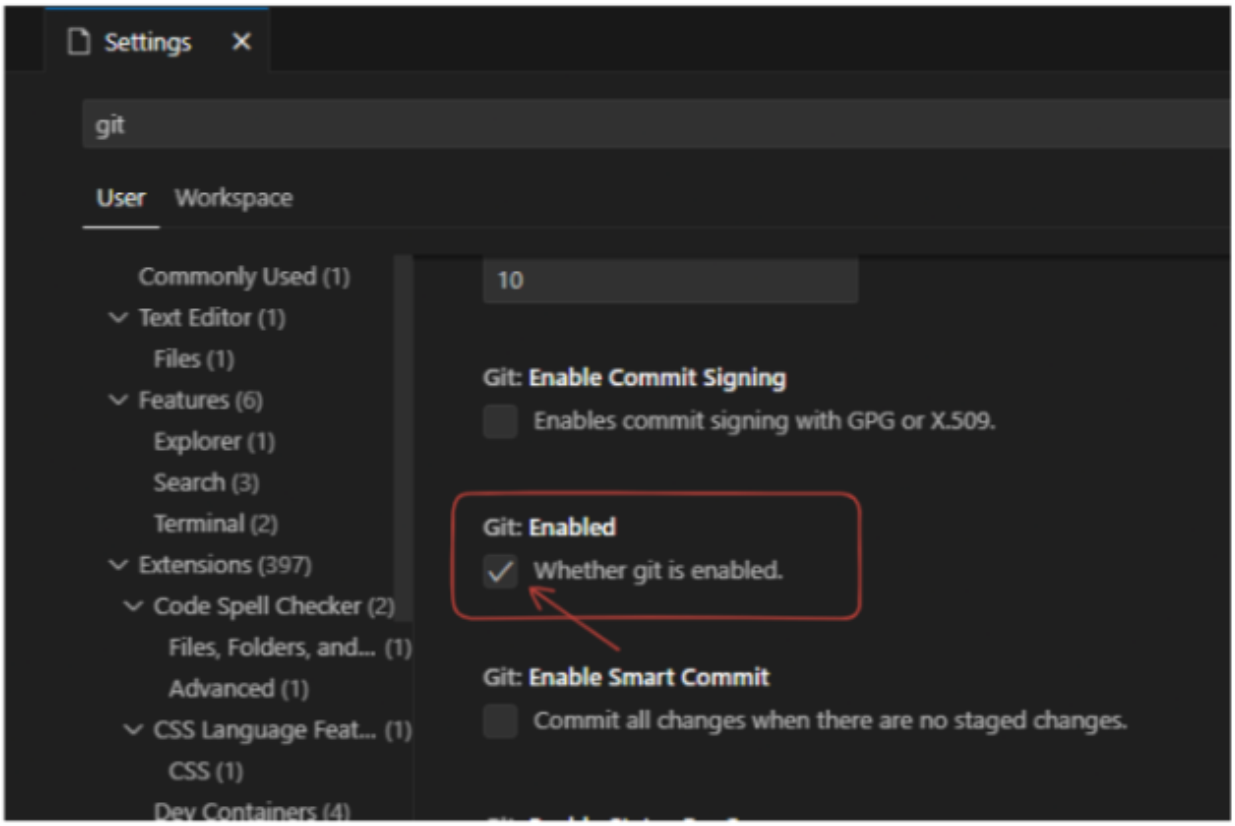


2. Install Git
Make sure Git is installed in your system.
To see if Git is already installed on your system, open your terminal and run the command given below -

```
Unset
git --version
git version 2.40.0.windows.1
```

If it is not installed, however, you can install it by downloading the executable file for your machine from Git's official website i.e**. https://git-scm.com/downloads/**

3. Enable Git in Vscode
To enable Git in VS Code on Windows or Mac
Go to settings > search for “Git” in the search bar > make sure Git:Enabled is ticked.



4. Create and Configure a GitHub account and sign in
Before proceeding, You have to make sure that you have a GitHub account, However, if you do not have a GitHub account, Click on the link - **https://github.com/signup** to sign up

Once your GitHub account is created, configure it with VS Code -

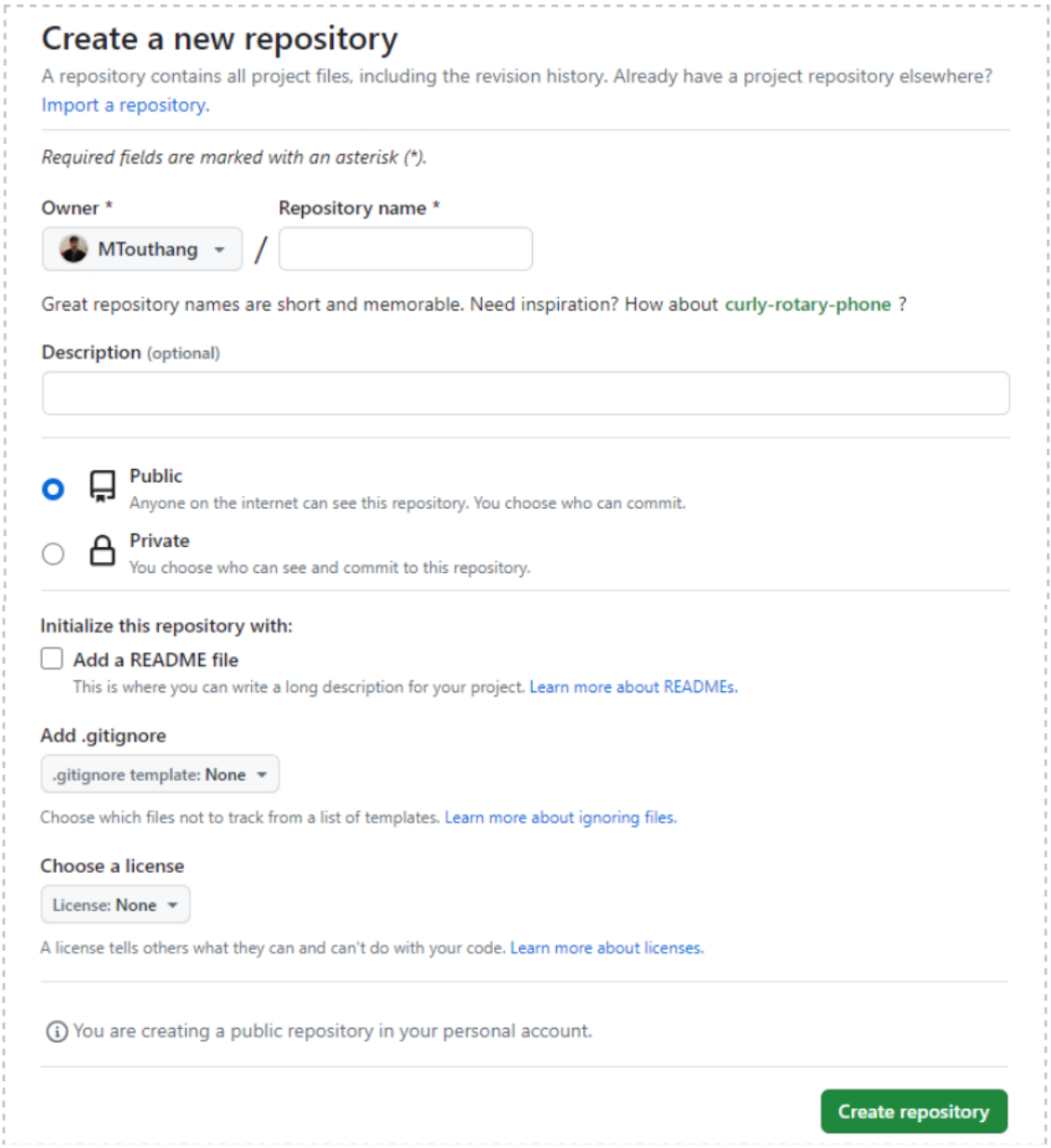
```
Unset
# Set username in Git with the GitHub username
git config --global user.name "your_user_name"

# Set an email address in Git, the email used in GitHub
git config --global user.email "youremail@gmail.com
```

5. Verify that Git is Linked to your GitHub
After configuring your details, you can use the following command in the terminal to view the email and username that was set up.

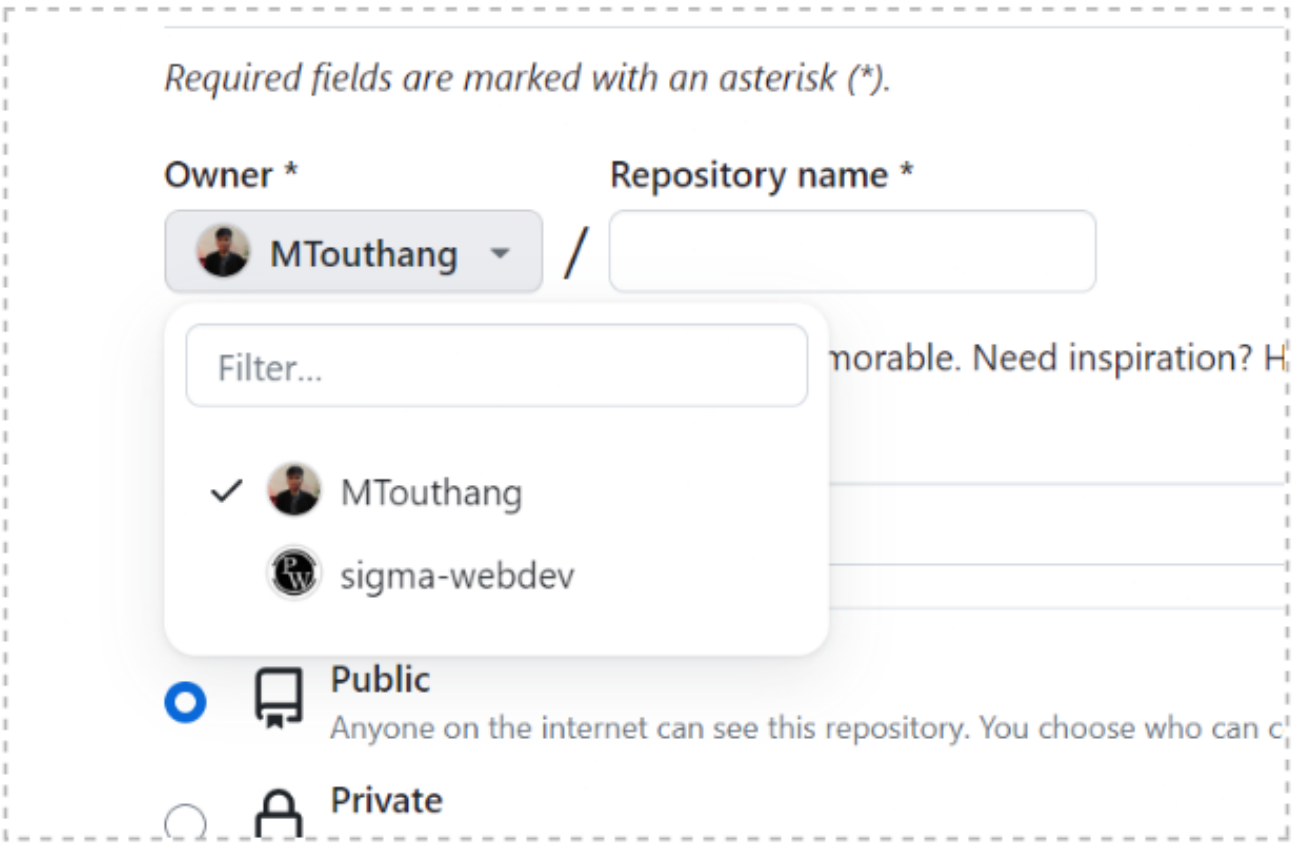
```
Unset
git config --global --list

#output -
user.email=youremail@gmail.com
user.name=yourusername
```

Creating the repo

To create a new repo(repository) on GitHub, click the + sign to the right of your username at the top right of the page. Then click the “New repository” option in the drop-down list. You’ll see the new repository form as shown below -



Steps to create a new repository -
1. Choose project owner -
The first thing to do is decide whether you want to create the repository under your username or under an organization, if you do not have access to any organization you'll just leave this defaulted to your username.

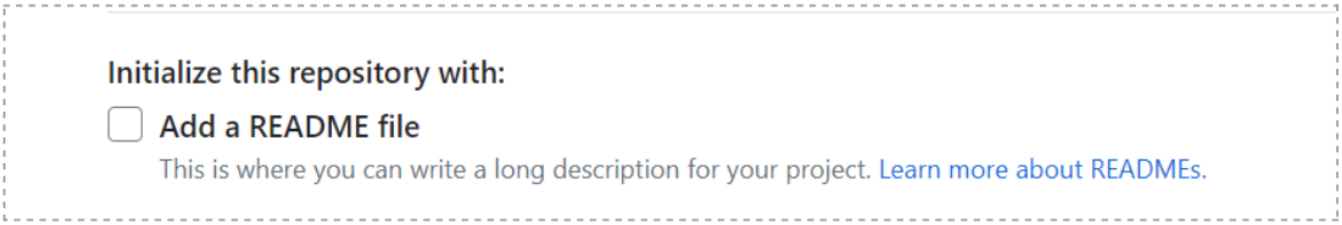


2. Repository name -
The next step is to give the repository a name, Nam should be comprised of letters, numbers, hyphens, and or underscores. Any other characters will be replaced with a hyphen.

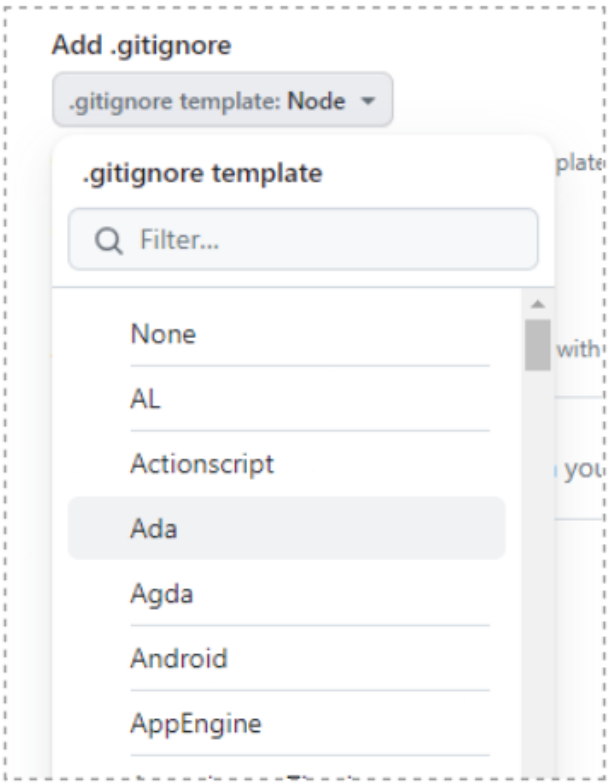
3. Repository description (Optional)
This is an optional step, It is mainly a description or some words about the repository.

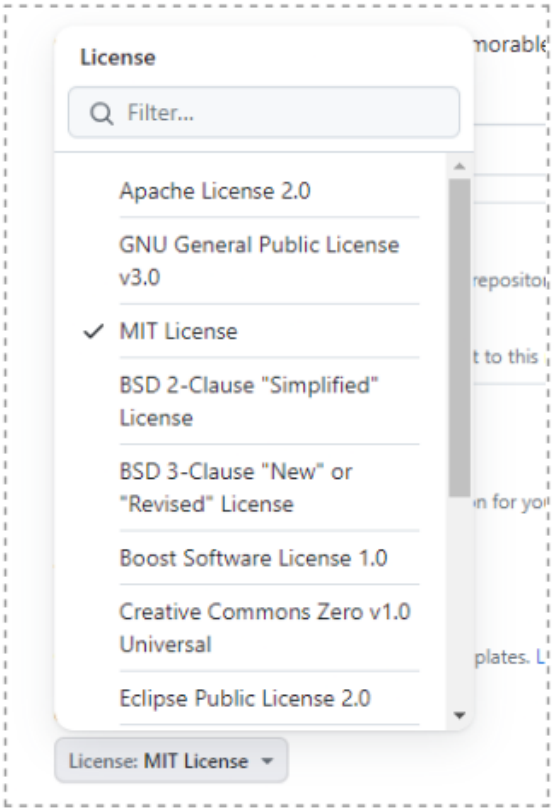
4. Private and Public Repo
After entering the name, you need to decide whether to make the repository private or not. Public repositories can be viewed by anyone. Private repositories can be viewed by only people that you specifically invite as collaborators.

5. Add README.md
The next decision you'll need to make when creating a new repository is whether or not to initialize it with a README file by checking the checkbox as shown below.



6. The .gitignore and license file -
The final decision you'll need to make when creating a new repository is whether to include and choose the suitable .gitignore and license file for your project as shown below -





Pushing code to GitHub

After the Repository has been created, You can push the code and files to the created repository in two ways -

1. Create a new project with Git and push it to GitHub(remote) repository

```
Unset

#1 create a folder "project_html"
mkdir project_html

#2 navigate to the project folder
cd project_html
```

```
#3 initializing git
git init

#4 add code files and stage and commit
touch index.html #create html files and add some html codes
git add .
git commit -m “initial commit”

#5. Get GitHub repo link and perform push
# set up a remote repository link named "origin" for your local git repository.
git remote add origin <github_repo_link>

#  rename the default branch from "master" (the previous default) to "main."
git branch -M main

# push the local branch (in this case, "main") to the remote repository named
"origin."
git push -u origin main
```

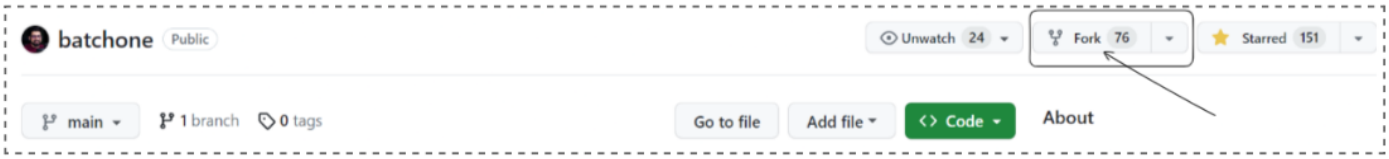
2. Push an existing Git project to the GitHub(remote) repository.
Assume you have a stage and a committed project and want to push your code files to the GitHub repository, follow the instructions below -

```
Unset
# inside the root directory of existing git project -
# add and setup a remote repository link name origin for your local Git repo
git remote add origin <github_repo_link>

#rename default branch master to main
git branch -M main

# push the local branch to the remote repository named origin
git push -u origin main
```

Forking an existing repo

Forking on GitHub is a fundamental process that allows you to create a copy of someone else's repository (the "upstream" repository) under your own GitHub account. This copy, called a "fork," exists in your account and operates as an independent repository. You can modify, add to, and experiment with the code in your fork without affecting the original repository.

To fork a GitHub repository, First, navigate to the GitHub repository to be forked, and click the Fork button in the top right corner of the repository page, as shown below -



After a successful fork, a copy of the GitHub repository will be available in your repository section.

Generating pull requests and merge requests

Generating pull requests and merge requests is a crucial part of the collaborative development process, especially in version control systems like Git.
After you successfully forked and cloned the desired GitHub repository(project)
Here's a step-by-step guide on how to generate a pull request(PR) and merge request -

1. Create a Branch - start by creating a new branch from the main repository, where you'll make your changes for the new features, bugs, fixes, or improvements

```
Unset
# create branch
git branch <branch_name>

#switched branch
git checkout <branch_name>
```

2. Make Changes- Make the necessary code changes in the branch you created. Ensure your changes address the issue or feature you're working on.

3. Stage and commit changes

```
Unset
# stage all changes
git add .

# commit all changes
git commit -m "initial commit"
```

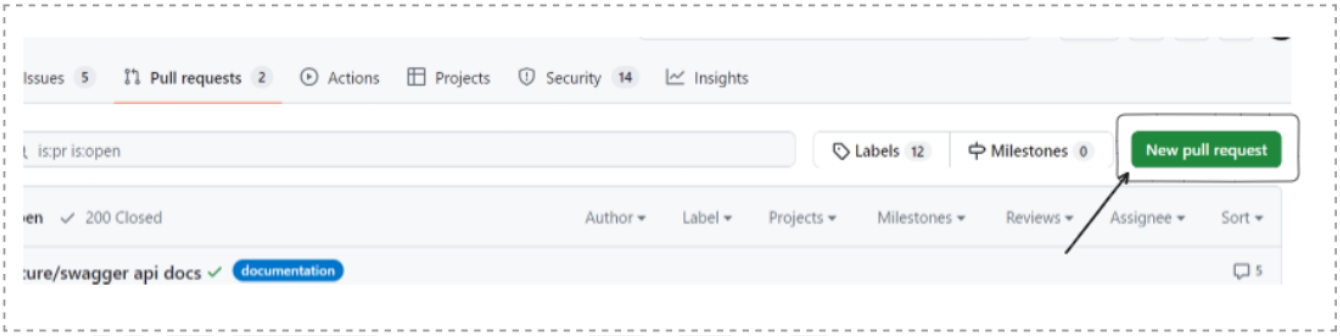
4. Sync with the original repository
Use the git pull command to fetch and merge the changes from the remote repository into your local repository:

```
Unset
git pull
Use the git pull command to fetch and merge the changes from the remote
repository into your local repository:
```

5. Push Branch
Push the branch containing your changes to the remote repository

```
Unset
# from the created branch run the below command
git push
```

6. Open a Pull Request
Once the code files are pushed, go to your forked repository on the platform (GitHub), click on "New Pull Request" and select the base branch (e.g., main) and the branch with your changes as the compare branch. Provide a descriptive title and detailed description of the changes.



7. Review and Discuss
Invite relevant team members or repository maintainers to review your pull request, engage in discussions, and address any feedback or concerns.

8. Make Necessary changes
If requested, make any required changes based on the feedback received during the review.

9. Merge the pull request
Once the pull request is approved and all discussions are resolved, merge the changes into the original repository.

10. Delete the branch
After merging, delete the branch you created for the pull request.

Handling merge conflicts

Merge conflicts are conflicts in files that occur when there are conflicting changes in the code or content of a file between different branches that Git is trying to merge. This typically happens during a merge operation, where Git attempts to integrate changes from one branch to another.

Handling merge conflicts in Git involves resolving differences between branches that occur when Git cannot automatically merge the change.
Here are the steps to handle a merge conflict -

1. Identify the conflict files - When you attempt to merge branches using git pull or git merge, Git may display a message indicating that there is a conflict. You'll see something like:

2. Open the conflict files - Open the files that have conflicts in a text editor. Git will mark the conflicting sections within the file.

3. Resolve the conflicts - Inside the conflicted file, you'll see something like this:

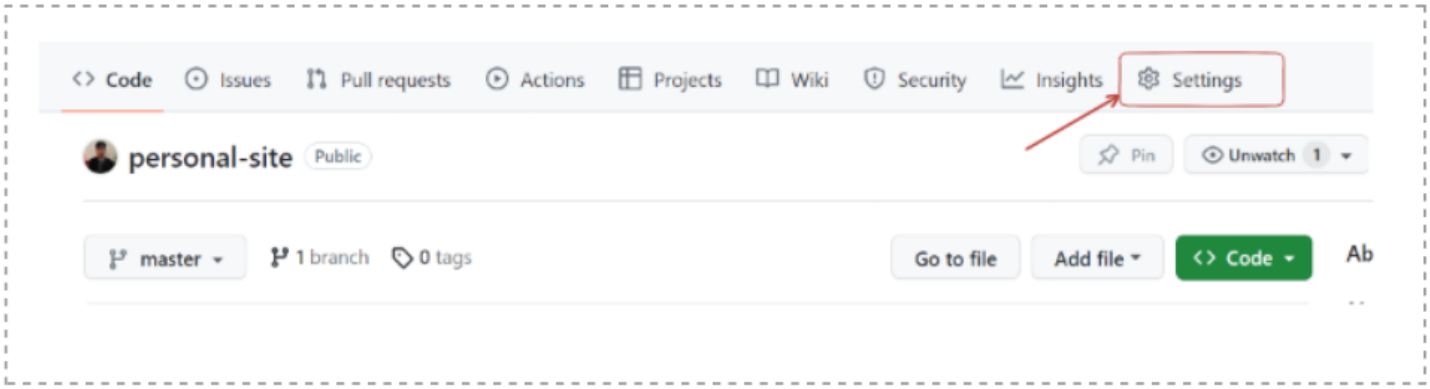
```
Unset
<<<<<<< HEAD
This is the content from the current branch
=======
This is the content from the incoming branch
>>>>>>> incoming-branch
```

4. Now, after resolving the conflict stage and committing your changes push is needed.

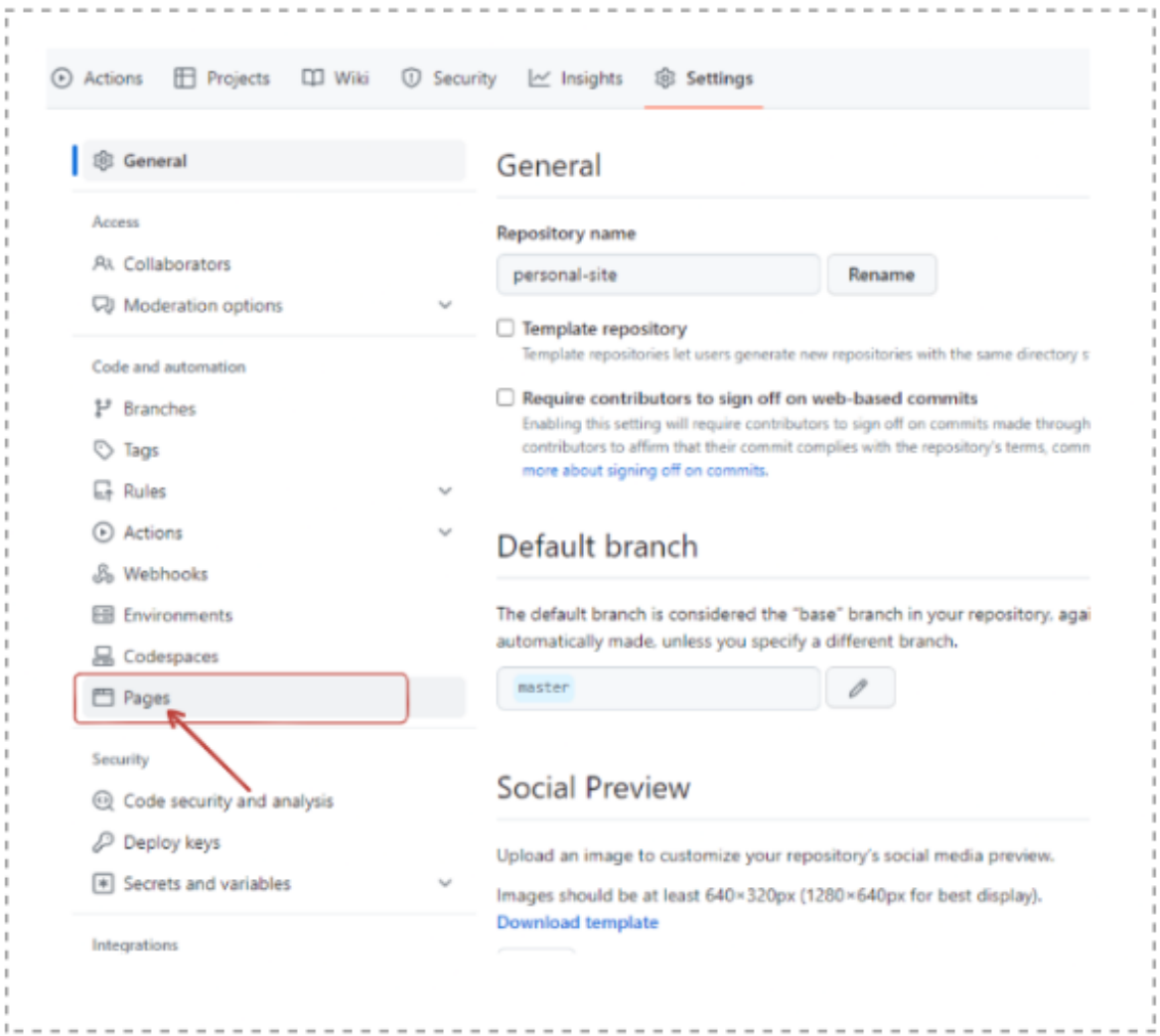
Hosting a website on GitHub Pages

Hosting a website on GitHub Pages is a straightforward process. GitHub Pages allows you to host static websites directly from a GitHub repository.
Here are the steps to host a website using GitHub Pages:

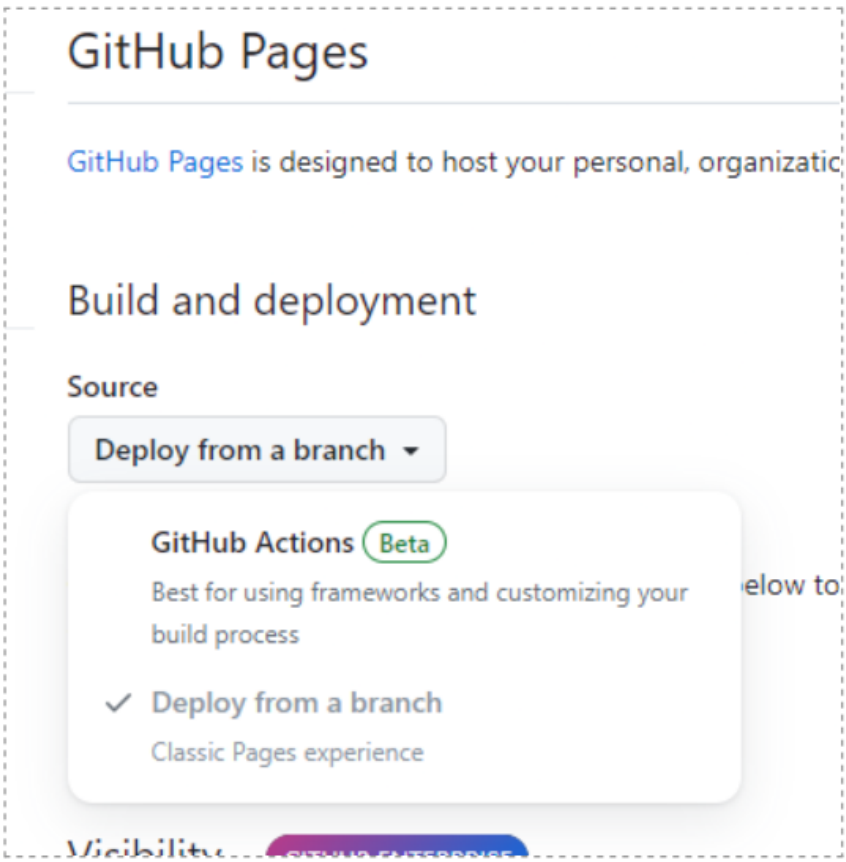
1. Create a GitHub Repository - make sure you have a GitHub repository that is public.
2. Push Your Website Files to the Repository - Create or add all the necessary files like HTML, CSS, and Javascript files for your website.
3. Setup and enable the GitHub page - now that all the necessary files are pushed, navigate the repository and follow the steps given below
   a. Click on the setting icons inside the desired GitHub repository that you want to host.



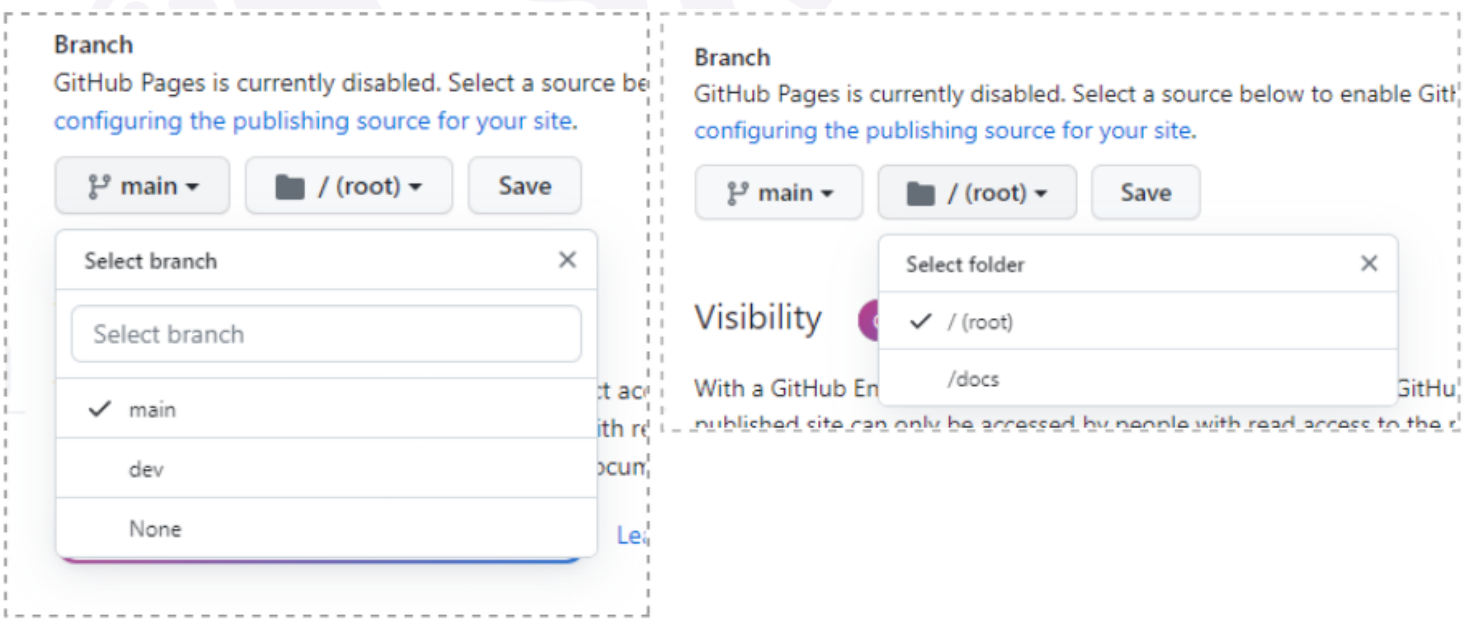
b. Select **pages** on the right sidebar



C. Select the source of deployment - select the “deploy from a branch”

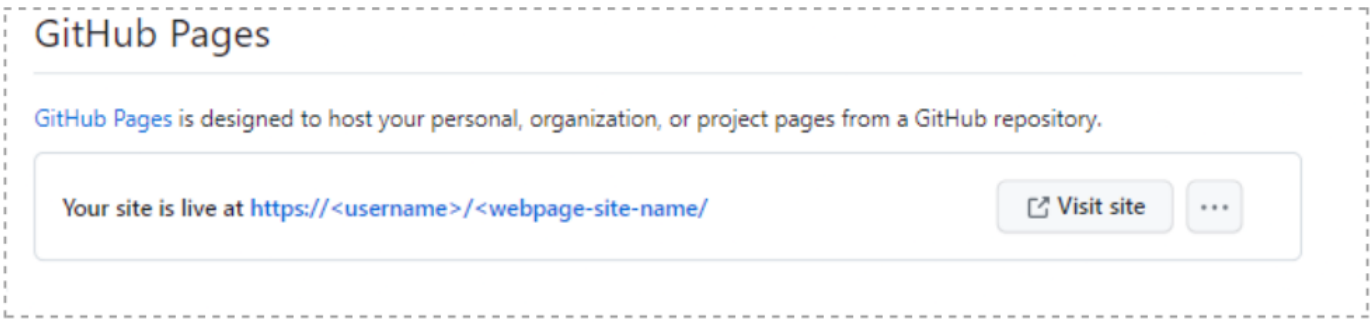


D. Select the branch that needs to be hosted - select the branch and specify the root folder.



Click on save to host the selected branch, now wait for some minutes for your GitHub page to be hosted.

4. Access your web page - Once you've completed all the steps, your webpage will be live and accessible through a link displayed in the "Pages" section, as demonstrated below:



Alternative of GitHub

Some of the platforms that provide the same functionality similar to GitHub can be -

1. GitLab - GitLab provides a comparable suite of services to GitHub, encompassing Git repository hosting, code collaboration, and continuous integration. It also offers GitLab Pages for website hosting.

2. BitBucket - Bitbucket is a robust alternative offering Git repository hosting, code collaboration, and issue tracking. Similar to GitHub, Bitbucket supports continuous integration through features like Bitbucket Pipelines.

3. SourceForge: SourceForge is a long-standing open-source software development platform that offers a variety of features, including code hosting, issue tracking, and project management. It has a free plan for open-source projects, and it also offers paid plans for private projects.

4. Launchpad: Launchpad is a software collaboration platform by Canonical, the parent company of Ubuntu. It offers code hosting, issue tracking, and other features for open-source projects. Launchpad is free to use.

5. Gitea: Gitea is a self-hosted Git repository hosting service. It is free and open source, and it offers many of the same features as GitHub. Gitea is a good option for teams who want to host their own code repositories.