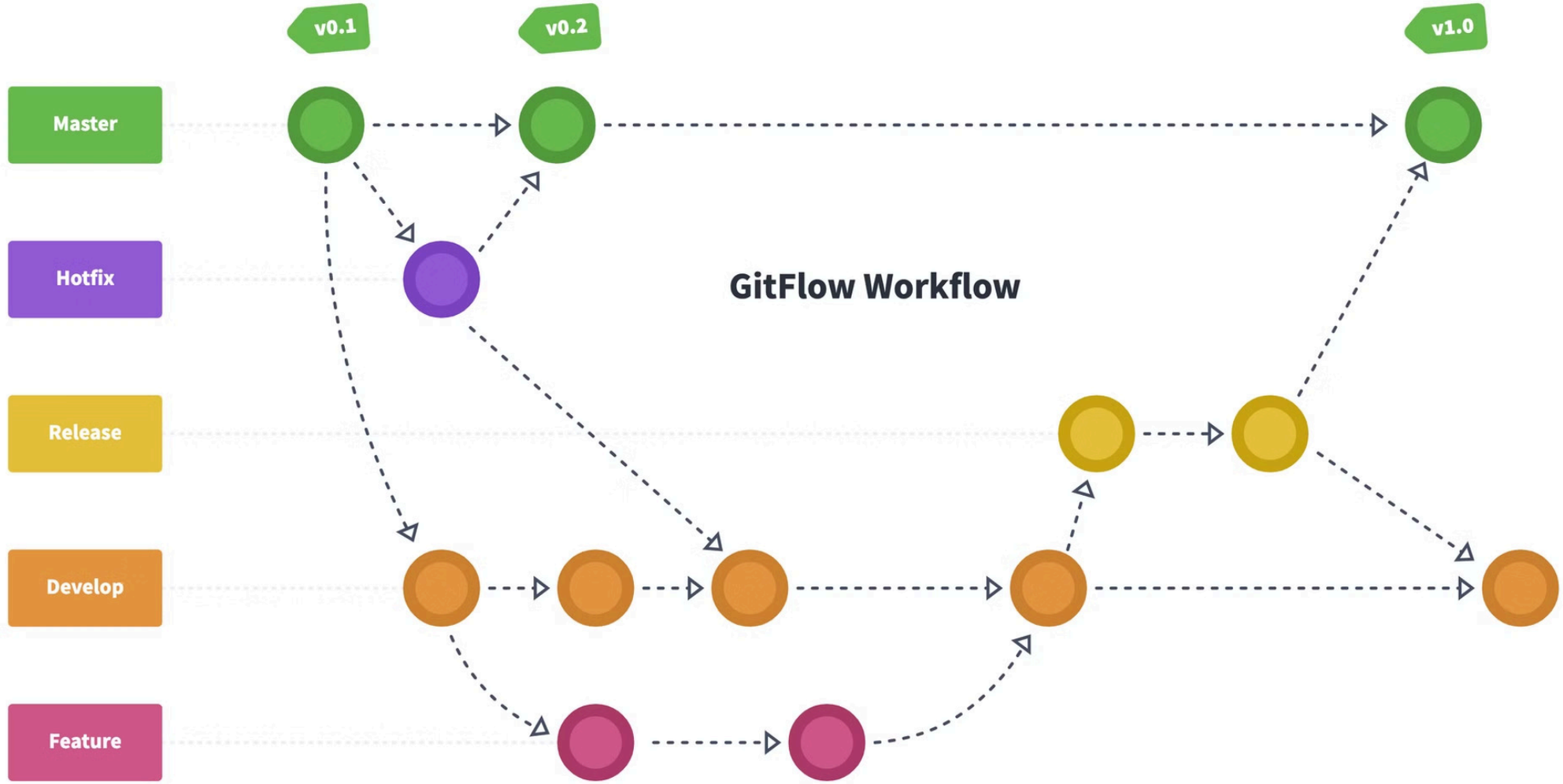
Introduction to Git and GitHub

Git is a powerful version control system, allowing you to track changes in your code and collaborate with others. GitHub is a platform for hosting and managing Git repositories, providing a collaborative environment for developers.

by farheen khan

GitHub process diagram



1. Git install link:-

https://git-scm.com/downloads

Git - Downloads

The entire Pro Git book written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

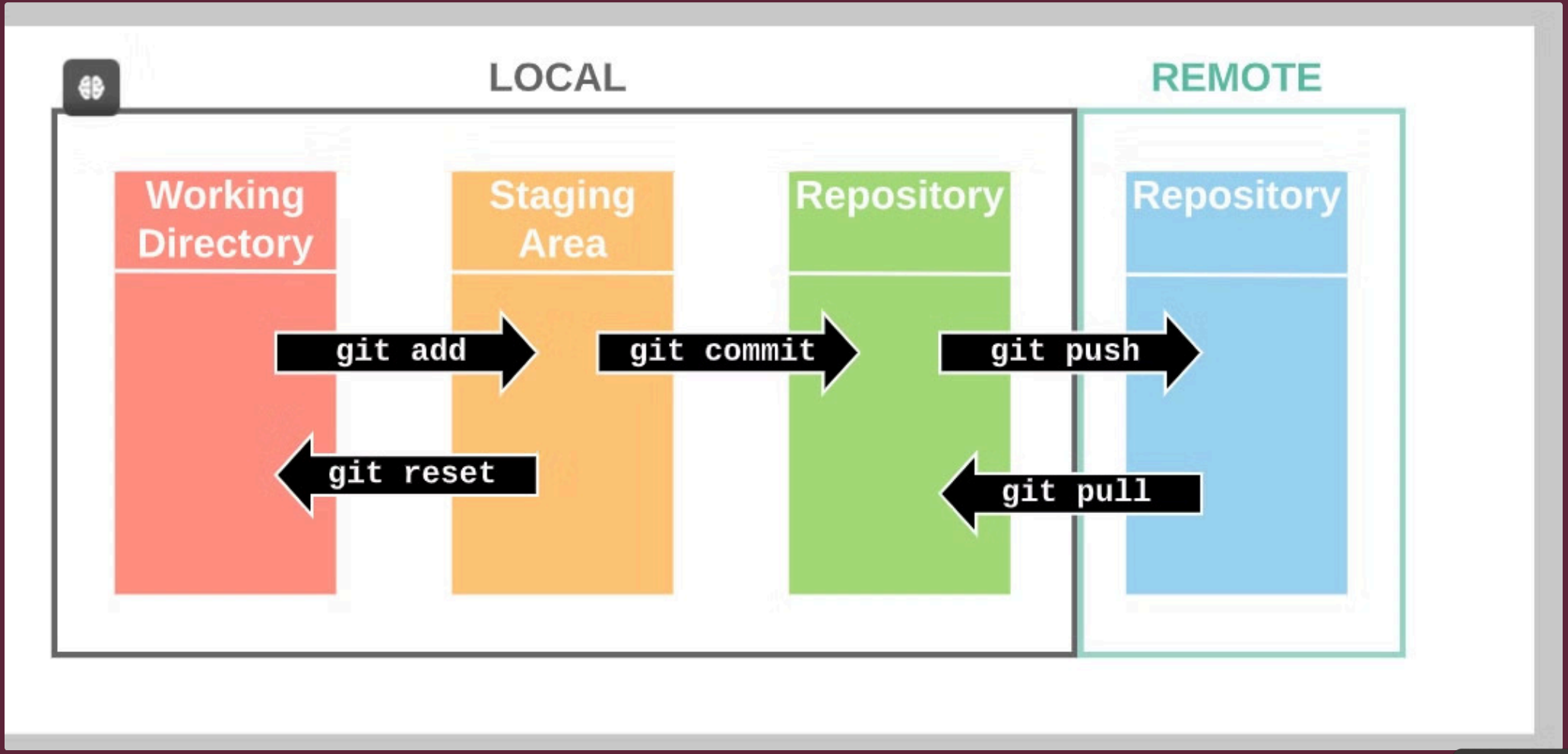
2. Github link:-

https://github.com/

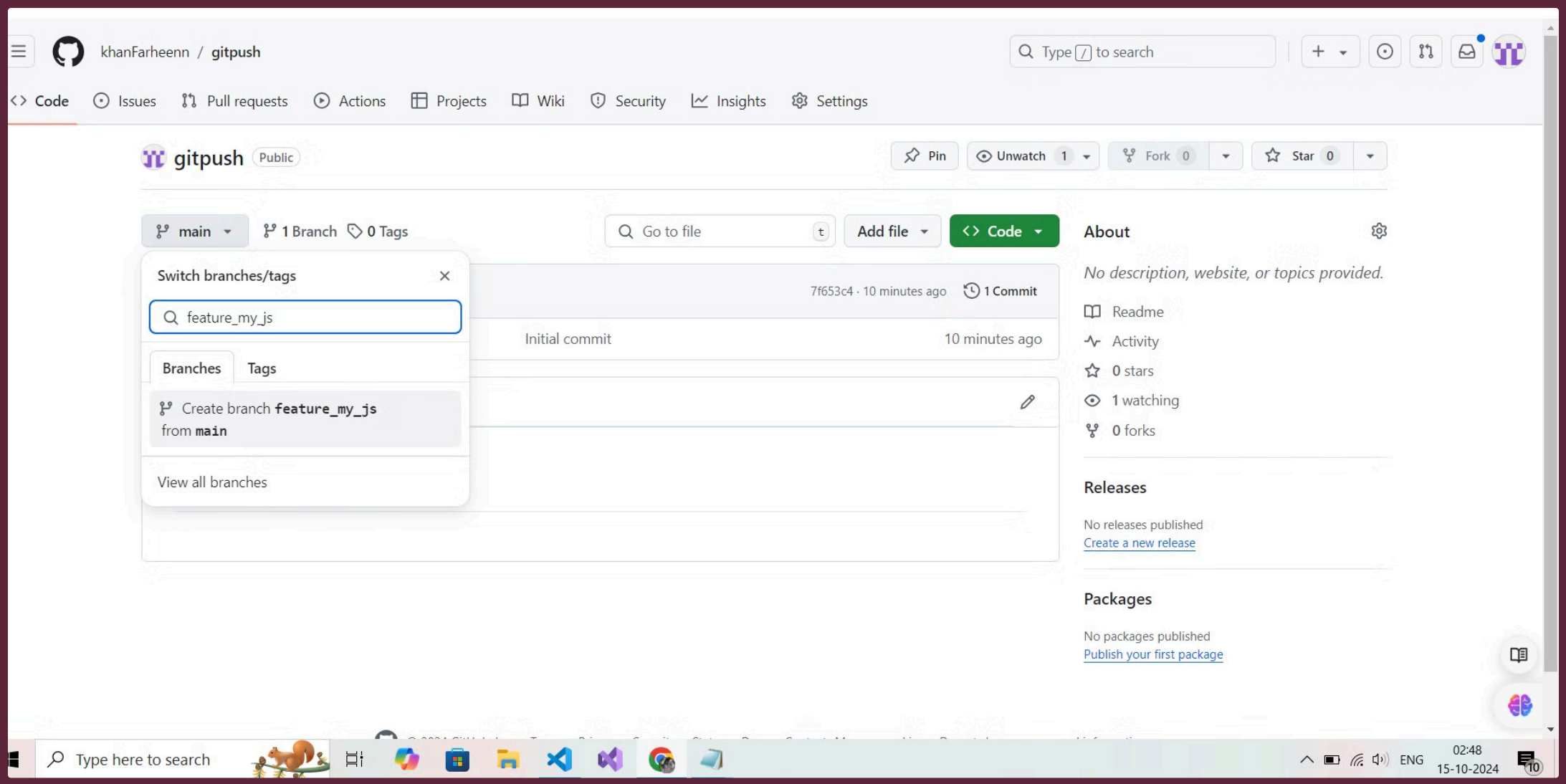
GitHub: Let’s build from here

GitHub is where over 100 million developers shape the future of software, together. Contribute to the open source community, manage your Git repositories, review code like a pro, track bugs and fea...

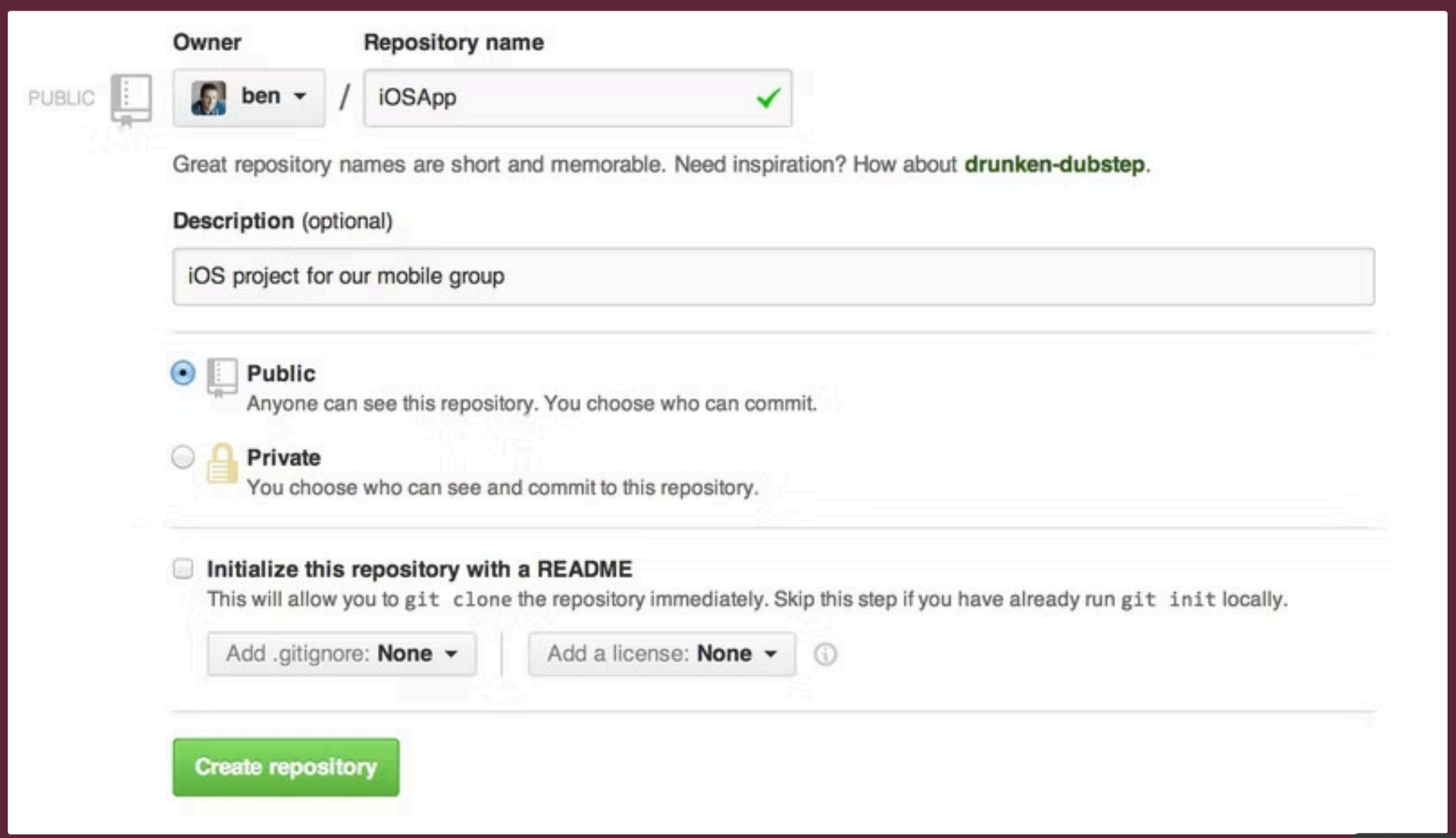
Wprking Directory in github



How to create branch in github



How to create Repository in github



Command :-

1. echo "# gitteach" >> README.md
2. git init
3. git add README.md
4. git commit -m "first commit"
5. git branch -M main
6. git remote add origin **https://github.com/khanFarheenn/gitteach.git**
7. git push -u origin main
8. git remote add origin **https://github.com/khanFarheenn/gitteach.git**
9. git branch -M main
10. git push -u origin main

What is Git?

Git helps track changes in your code over time, like a history of your project.

1 Version Control

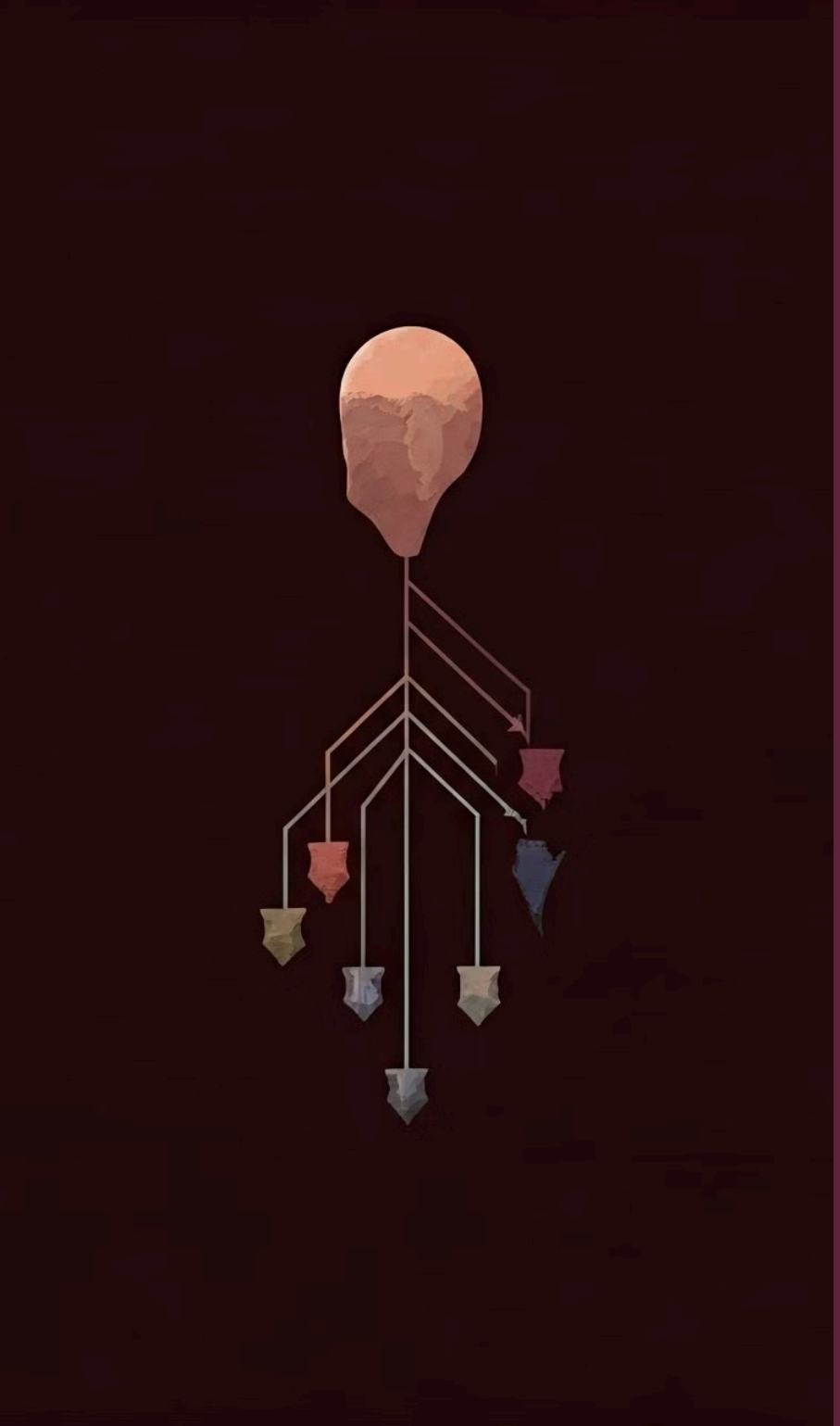
Git lets you go back to older versions of your code, so you always have access to previous project states.

2 Branching

You can create branches to work on different parts of your project at the same time, without messing up the main code. Git lets you combine these branches later.

3 Collaboration

Git makes working together easy because it lets multiple people work on the same project, merge their changes, and fix any problems that might come up.



Git Repositories and Branches

A Git repository is a directory containing all your project files and the Git metadata that tracks their changes. Branches are independent lines of development within a repository, allowing you to work on different features or versions simultaneously.

1

Repository Creation

Start by creating a new Git repository for your project, initializing it with the `git init` command.

2

Branching

Use the `git branch` command to create new branches. Switch between branches using `git checkout`.

3

Merging

Once your work on a branch is complete, merge it back into the main branch using `git merge`.

Committing Changes with Git

Committing changes is the process of saving your work and creating a snapshot of the state of your project. Git allows you to create a detailed history of your work.

Staging Changes

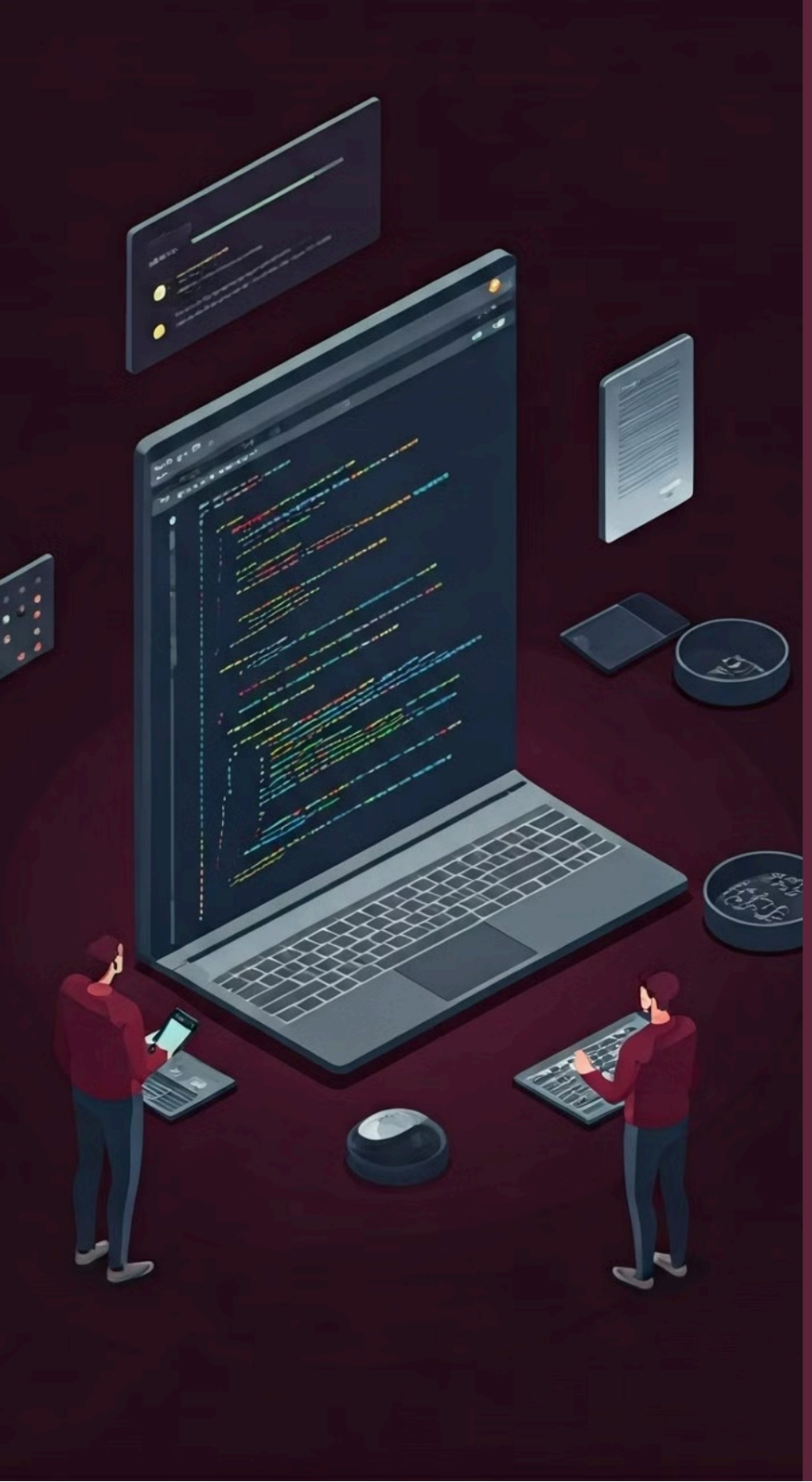
Use the `git add` command to stage the files you want to include in your next commit. This indicates which changes will be part of the commit.

Committing

Use the `git commit` command to create a commit, which captures the staged changes and saves them in your repository.

Commit Messages

Write descriptive commit messages to explain the changes made in each commit, making it easier to understand the history of your project.



Collaborating on GitHub

GitHub provides a platform for hosting and managing Git repositories, enabling seamless collaboration between developers.

Forking Repositories

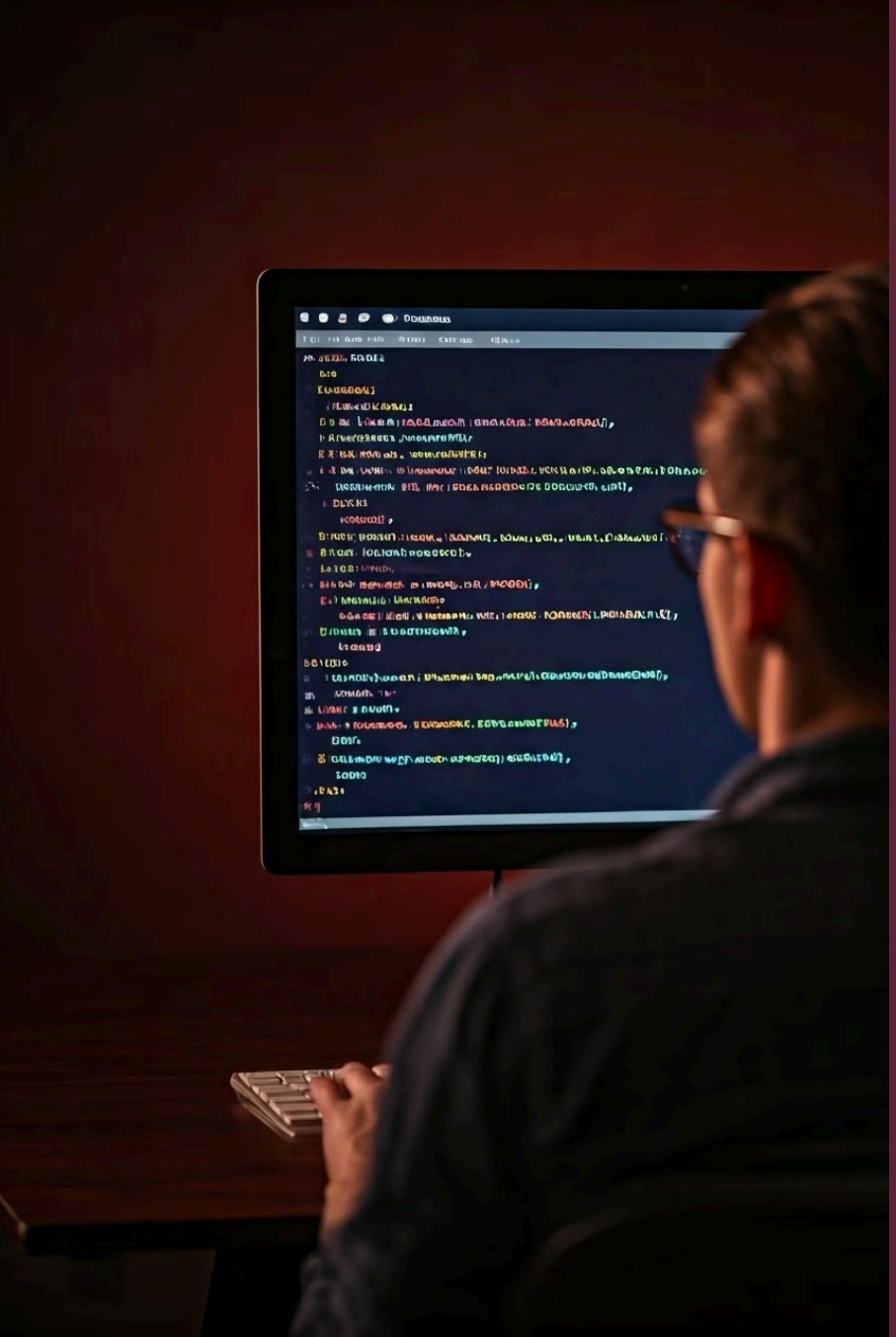
Create a copy of a repository on your own GitHub account to contribute independently.

Pull Requests

Propose changes to the original repository by submitting a pull request, allowing the original project owner to review your changes before merging.

Issues and Discussions

Use issues to track tasks, bugs, and feature requests, and use discussions to hold conversations and brainstorm ideas.



Resolving Merge Conflicts

Merge conflicts occur when changes made in different branches overlap. Resolving merge conflicts requires manually reviewing the code and choosing which changes to keep.

Step	Action
1	Identify Conflicting Files
2	Review the Code
3	Resolve Conflicts
4	Stage and Commit Changes

Pushing a Folder to GitHub

1

Initialize Git

Open the terminal in VS Code and navigate to your folder. Use the command `git init` to create a new Git repository.

2

Add Files

Use `git add .` to add all files in the current directory to the staging area. If you need to add specific files, use `git add `.

3

Commit Changes

Use `git commit -m "Your commit message"` to create a snapshot of your changes and add a descriptive message.

4

Connect to GitHub

Create a new repository on GitHub or use an existing one. In VS Code, click "Source Control", then click on the three dots and select "Push to GitHub".

5

Push to Remote

Enter your GitHub username and password or token. Select the repository and click "Push" to upload your changes to GitHub.

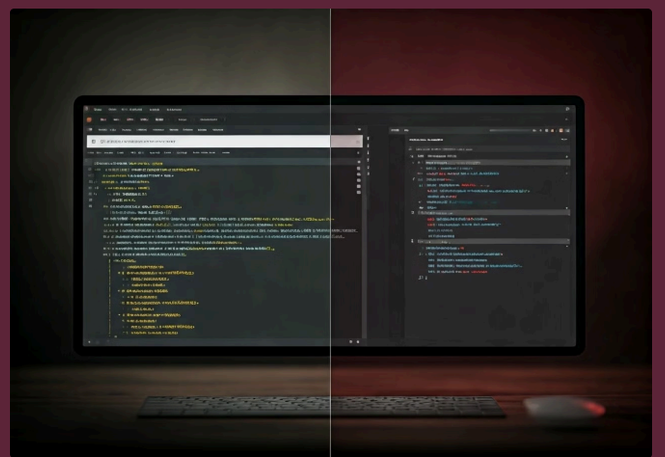
Pushing a Folder to GitHub: Visual Guide

VS Code Steps

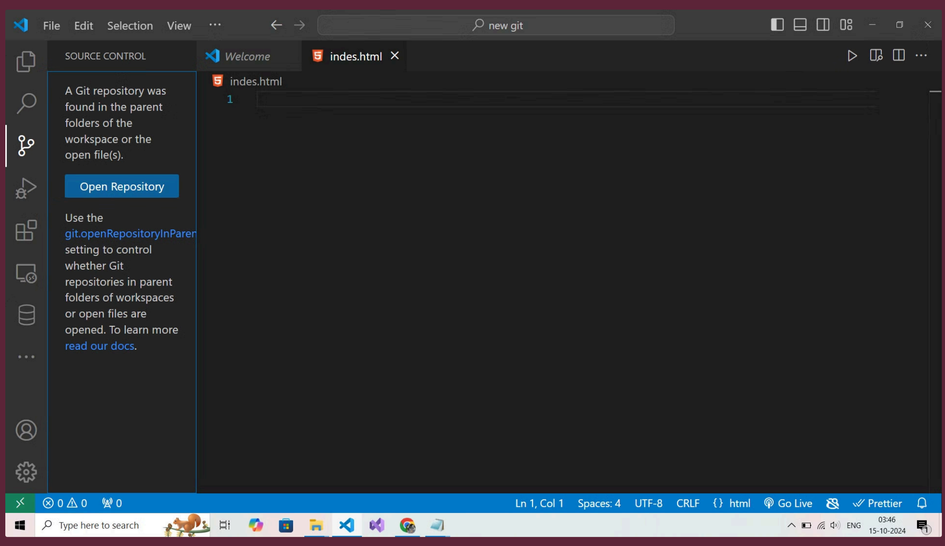
1. Open your VS Code project folder.
2. Click the "Source Control" icon in the left sidebar.
3. Click the "New Branch" icon.
4. Type a branch name and click "Create Branch".
5. Make your changes to files.
6. Click the "Source Control" icon again.
7. Enter a commit message in the text box.
8. Click "Commit" to save your changes.
9. Click the "Source Control" icon again.
10. Click the three dots icon and select "Push to GitHub".
11. Click "Push" to upload changes to GitHub.

GitHub Steps

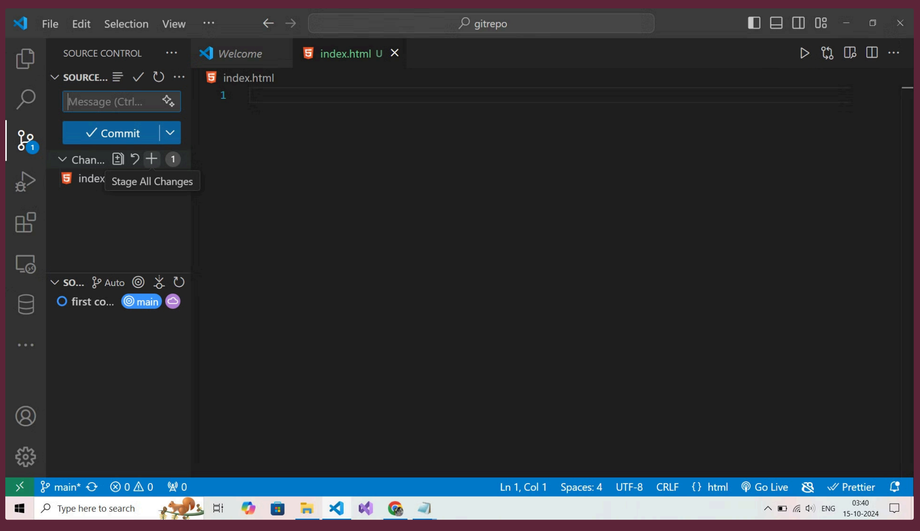
1. Log in to your GitHub account.
2. Create a new repository.
3. Copy the repository URL.
4. Open your VS Code project folder.
5. Click the "Source Control" icon.
6. Click the three dots icon and select "Push to GitHub".
7. Paste the repository URL into the "Remote" field.
8. Click "Push" to upload changes to GitHub.



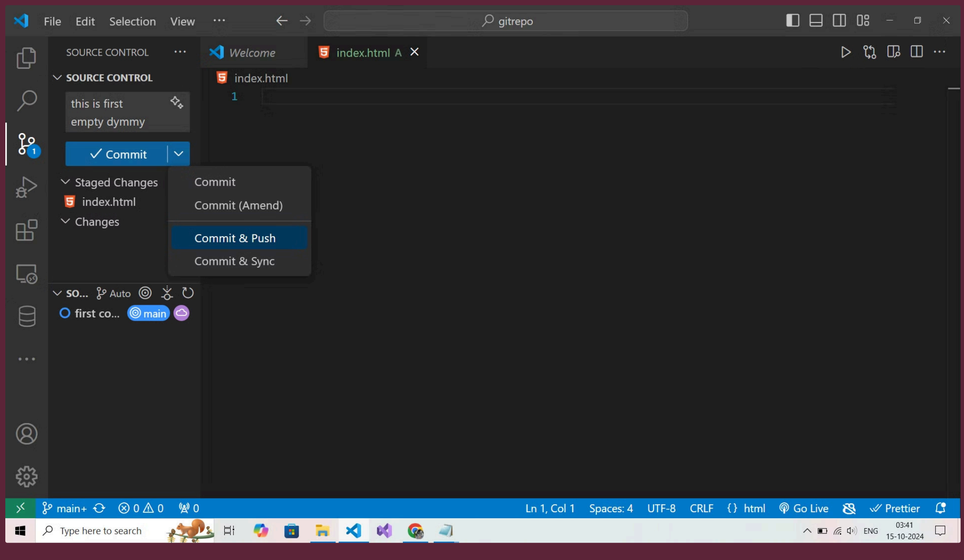
1. Create Repo in vs code



2. Stage All change :-



3. How to commit in vs code :-



4. How to Create Branch in vs code :-