

GITHUB

What is GitHub?

GitHub is one of the most popular resources for developers to share code and work on projects together. It's free, easy to use, and has become central in the movement toward open-source software.

It makes it easy for developers to share code files and collaborate with fellow developers on open-source projects. GitHub also serves as a social networking site where developers can openly network, collaborate, and pitch their work.

How does GitHub work?

GitHub users create accounts, upload files, and create coding projects. But the real work of GitHub happens when users begin to collaborate.

While anyone can code independently, teams of people build most development projects. Sometimes these teams are all in one place at once time, but more often they work asynchronously. There are many challenges to creating collaborative projects with distributed teams. GitHub makes this process much simpler in a few different ways.

First, all the code and documentation are in one place. This limits issues with access for anyone who wants to contribute to a project. Each repository also contains instructions and other details to help outline project goals and rules.

Next, coding is more creative and abstract than most non-technical people think it is. For example, say two devs are working on different pieces of code. These two pieces of code should work together. But sometimes one piece of code can make the other code fail. Or a piece of code can have an unexpected impact on how the other code works.

GitHub solves these problems by showing how both files will change the main branch. It catches these errors before pushing changes, making the coding process more efficient.

GitHub also makes it easier to track changes and go back to previous versions of a project. To explain this, we'll need to understand the technology that GitHub is based on, Git, and talk about version control.

WHAT IS GIT...

Git is open-source version control software, used for managing and tracking file revisions. You can use Git with any file type, but it's most often used for tracking code files.

Git is the most widely used version control system in software development, and GitHub leverages this technology for its service, hence its name. Even though Git's user interface is fairly similar to these other VCSs, Git stores and thinks about information in a very different way, and understanding these differences will help you avoid becoming confused while using it.

The major difference between Git and any other VCS (Subversion and friends included) is the way Git thinks about its data. Conceptually, most other systems store information as a list of file-based changes. These other systems (CVS, Subversion, Perforce, and so on) think of the information they store as a set of files and the changes made to each file over time (this is commonly described as **delta-based** version control).

Git doesn't think of or store its data this way. Instead, Git thinks of its data more like a series of snapshots of a miniature filesystem. With Git, every time you commit, or save the state of your project, Git basically takes a picture of what all your files look like at

that moment and stores a reference to that snapshot. To be efficient, if files have not changed, Git doesn't store the file again, just a link to the previous identical file it has already stored. Git thinks about its data more like a **stream of snapshots**.

Δ Delta storage,	📁 Snapshot storage
saves only the changes made to a file or system rather than the entire state. Each change is stored as a delta or a difference from the previous version, and the system's current state is reconstructed by combining all the deltas. This method is more efficient in terms of storage space, as it only stores the differences between versions but can be more complex to implement and manage.	saves the entire state of a file or system as a single instance, known as a snapshot. Each snapshot represents the system at a given time and serves as a reference point for future changes. This method is simple, efficient, and straightforward, but it can consume a lot of storage space over time if there are frequent changes to the system.



How to push code in GitHub?

Using GitHubCLI...

If your code is stored locally on your computer and is tracked by Git or not tracked by any version control system (VCS), you can import the code to GitHub using GitHub CLI or Git commands.

GitHub CLI is an open source tool for using GitHub from your computer's command line. GitHub CLI can simplify the process of adding an existing project to GitHub using the command line.

HOW TO INSTALL Git on Ubuntu...

1. Git packages are available using apt.
2. It's a good idea to make sure you're running the latest version. To do so, Navigate to your command prompt shell and run the following command to make sure everything is up-to-date: `sudo apt-get update`.
3. To install Git, run the following command: `sudo apt-get install git-all`.
4. Once the command output has been completed, you can verify the installation by typing: `git version`.

HOW TO ADD REPOSITORY FROM LOCAL COMPUTER TO GitHub DESKTOP...

See this...

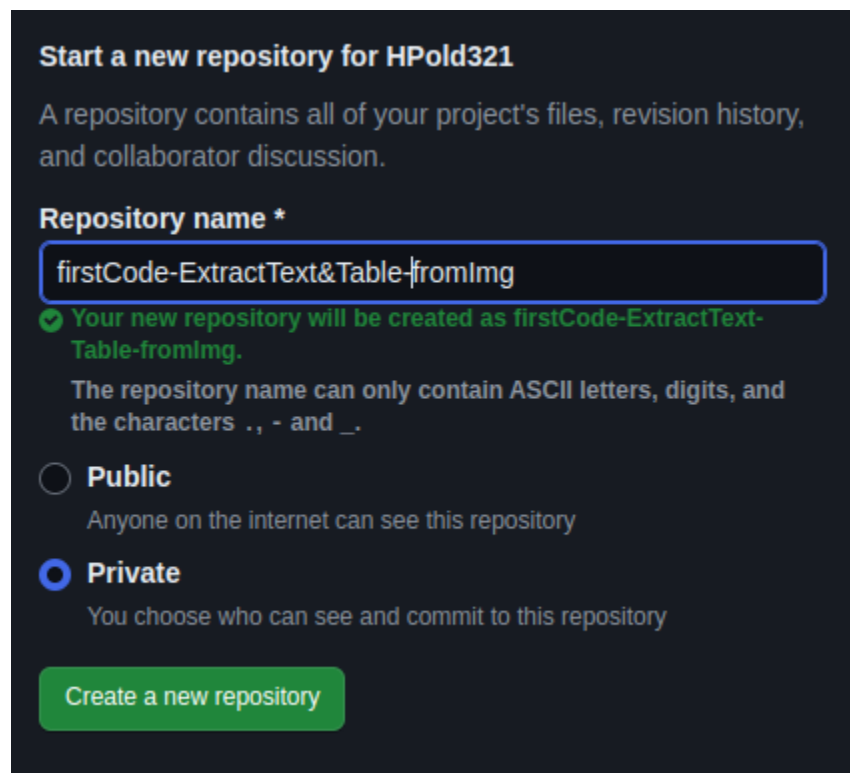
<https://docs.github.com/en/desktop/adding-and-cloning-repositories/adding-a-repository-from-your-local-computer-to-github-desktop>

SCREENSHOTS OF STEP by STEP execution

Installing git

```
ananya@sourav-H610M-H-V2-DDR4:~$ sudo apt-get install git
[sudo] password for ananya:
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.25.1-1ubuntu3.11).
git set to manually installed.
The following packages were automatically installed and are no longer required:
  gir1.2-goa-1.0 libfwupdplugin1 libxmlb1
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 10 not upgraded.
```

Creating a new Repository.



Start a new repository for HPold321

A repository contains all of your project's files, revision history, and collaborator discussion.

Repository name *

firstCode-ExtractText&Table-fromImg

✔ Your new repository will be created as firstCode-ExtractText-Table-fromImg.

The repository name can only contain ASCII letters, digits, and the characters ., - and _.

☐ Public
Anyone on the internet can see this repository

☒ Private
You choose who can see and commit to this repository

Create a new repository

Getting Path to GIT file...

Just go to the folder where your files are stored, right click to 'Open in Terminal'

To create first branch in git...

Always create 'main' branch first

```
>> git checkout -b main
```

To check status of git...

```
ananya@sourav-H610M-H-V2-DDR4:~/Downloads/git_code$ git status
On branch main
Your branch is up to date with 'origin/main'.
```

To add files to repository...

```
>> git add . # Adding '.' adds all files to repository at once.
```

```
ananya@sourav-H610M-H-V2-DDR4:~/Downloads/git_code$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   Screenshot.png
    new file:   Screenshot1.png
    new file:   invoice.jpg
    new file:   invoice.png
    new file:   invoice1.png
    new file:   text.png
```

To publish these, commit...

```

ananya@sourav-H610M-H-V2-DDR4:~/Downloads/git_code$ git commit -am "SecondCommit"
[main 4603432] SecondCommit
6 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Screenshot.png
create mode 100644 Screenshot1.png
create mode 100644 invoice.jpg
create mode 100644 invoice.png
create mode 100644 invoice1.png
create mode 100644 text.png

```

Final...

 HPold321 SecondCommit		4603432 · 3 minutes ago	 2 Commits
 Screenshot.png	SecondCommit	3 minutes ago	
 Screenshot1.png	SecondCommit	3 minutes ago	
 invoice.jpg	SecondCommit	3 minutes ago	
 invoice.png	SecondCommit	3 minutes ago	
 invoice1.png	SecondCommit	3 minutes ago	
 text.png	SecondCommit	3 minutes ago	
 textFromImg.ipynb	first commit	14 minutes ago	