

## Day 9: Deep Dive in Git & GitHub

This is [#90DaysofDevops](#) challenge under the guidance of [Shubham Londhe](#) sir.

Day 7 TASK

check this for task:

<https://github.com/LondheShubham153/90DaysOfDevOps/blob/master/2023/day09/tasks.md>



### What is Git and why is it important?

Git is a version control system that is widely used for software development and other version-controlled projects. It is a distributed system, which means that the code and its full history are stored in many different places, so it is very difficult to lose data.

Git is important because it provides a way for multiple people to collaborate on a project and keep track of the changes made to the code over time. This makes it easier to maintain and develop software, especially for large projects. With Git, developers can work independently and submit their changes, which can then be easily merged with the main codebase.

Git also makes it easier to revert to an earlier version of the code if a mistake is made or if you need to return to a previous state for some other reason. Additionally, Git allows for easy experimentation and exploration, as you can switch between different versions of your code to see the impact of different changes.

In summary, Git is a powerful tool that helps software development teams to collaborate and manage changes to their code over time, making it an essential tool for many software projects.

### **What is difference Between Main Branch and Master Branch??**

In Git, the main branch is the default branch name that is usually used to represent the primary line of development. The “main” branch is intended to serve as the long-lived, stable branch, where the source code of a project is held.

The “master” branch, on the other hand, is a commonly used name for the main branch in many Git projects. The name “master” originated from the days when Git was mostly used for version control of software code, and the “master” branch represented the latest version of the code that was considered production-ready.

So to summarize, both “main” and “master” refer to the primary line of development in a Git repository, but “main” is a more recent term that is used to avoid potentially problematic language.

### **Can you explain the difference between Git and GitHub?**

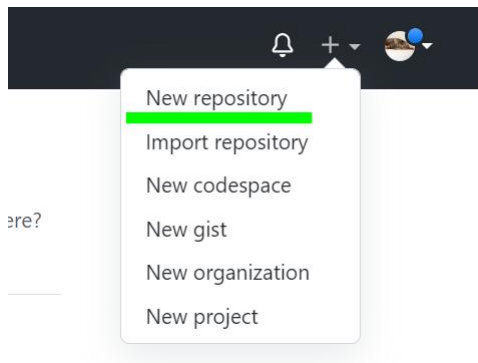
<https://medium.com/@misalPav/about-git-and-git-hub-af1edc4e0615>

### **How do you create a new repository on GitHub?**

To create a new repository on GitHub, follow these steps:

1. Go to the GitHub website and log in to your account.
2. Click the “+” icon in the top right corner of the screen and select “New repository”.
3. Enter a name for your repository in the “Repository name” field.
4. Optionally, you can add a description of your repository in the “Description” field.
5. Select whether you want your repository to be public or private. Public repositories are visible to anyone and can be cloned or forked by anyone, while private repositories are only accessible to you and other people you specifically invite.
6. Choose a license for your repository.
7. Click the “Create repository” button.

Once you have created your repository, you can add files, make changes, and share your code with others. You can also set up a Git repository on your local machine and push your changes to GitHub, so that you can keep track of your code and collaborate with others



## Create a new repository

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

### Repository template

Start your repository with a template repository's contents.

No template ▾

### Owner \*

 rajani103 ▾

### Repository name \*

super-duper-octo-fiesta ✓

Great repository names are short and memorable. Need inspiration? How about [super-duper-octo-fiesta?](#)

### Description (optional)

#### Initialize this repository with:

Skip this step if you're importing an existing repository.

#### ☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

### Add .gitignore

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

.gitignore template: None ▾

### Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▾

This will set  main as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

## What is difference between local & remote repository? How to connect local to remote?

In Git, a local repository is a copy of a project that is stored on your computer. You can make changes to the code in your local repository, commit those changes, and manage the history of your project without an internet connection.

A remote repository, on the other hand, is a version of your project that is stored on a server and is accessible from anywhere with an internet connection. Remote repositories are typically used for collaboration and sharing code with others.

To connect a local repository to a remote repository, you need to use the “git remote” command. Here are the basic steps to connect a local repository to a remote repository:

1. Initialize a Git repository in your local directory by running the “git init” command.
2. Go to GitHub and create a new remote repository for your project.
3. Copy the URL of the remote repository to your clipboard.
4. In your local repository, run the following command to add the remote repository:

```
git remote add origin <remote repository URL>
```

Replace “<remote repository URL>” with the URL of the remote repository.

5. To push your local repository to the remote repository, run the following command:

```
git push -u origin master
```

This command pushes the “master” branch of your local repository to the “origin” remote repository and sets the “origin” remote as the default remote for future pushes.

## Tasks:,

- Create a repository named “Devops” on GitHub

Create a new repository



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

---

**Repository template**  
Start your repository with a template repository's contents.

---

**Owner \*** **Repository name \***

 rajani103 / DevOps 

Great repository names are short and memorable. Need inspiration? How about [super-duper-octo-fiesta?](#)

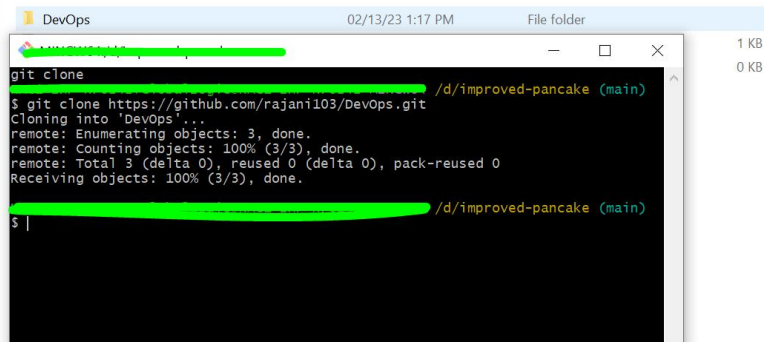
**Description (optional)**

---

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

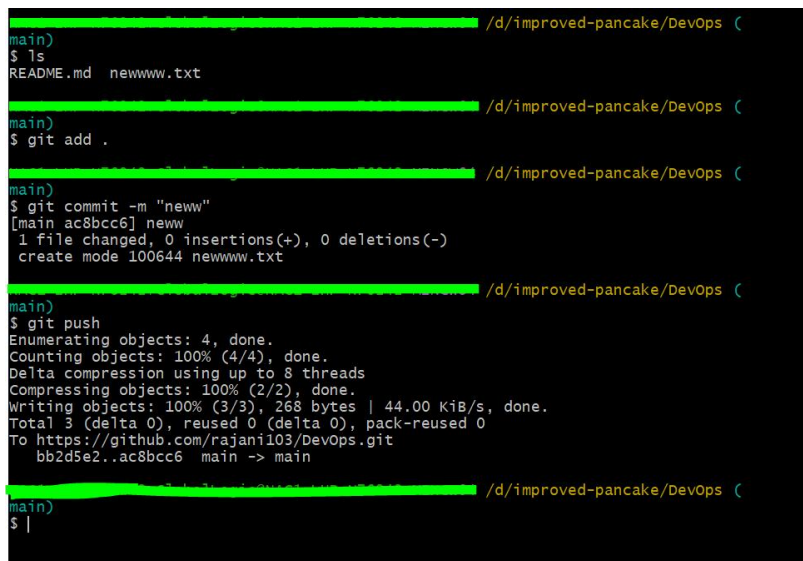
- Connect your local repository to the repository on GitHub.

Cloned the repo first



```
DevOps 02/13/23 1:17 PM File folder
git clone
$ git clone https://github.com/rajanil03/DevOps.git
Cloning into 'DevOps'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
$ |
```

- Create a new file in DevOps/Git/Day-02.txt & add some content to it
- Push your local commits to the repository on GitHub



```
/d/improved-pancake/DevOps (main)
$ ls
README.md newwww.txt

/d/improved-pancake/DevOps (main)
$ git add .

/d/improved-pancake/DevOps (main)
$ git commit -m "neww"
[main ac8bcc6] neww
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 newwww.txt

/d/improved-pancake/DevOps (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 268 bytes | 44.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/rajanil03/DevOps.git
bb2d5e2..ac8bcc6 main -> main

/d/improved-pancake/DevOps (main)
$ |
```

Please, feel free to drop any questions in the comments below. I would be happy to answer them.

If this post was helpful, please do follow and click the clap

\_Thank you for reading

\_Rajani