

เรื่องที่ 10 Version Control

ENGCE174 การเขียนโปรแกรมเชิงวัตถุ (Object-oriented programming)

อ.กิตตินันท์ น้อยมณี (mr.kittinan@rmutl.ac.th)

What is GitHub?

GitHub is an immense platform for code hosting. It supports version controlling and collaboration and allows developers to work together on projects. It offers both distributed version control and source code management (SCM) functionality of Git. It also facilitates collaboration features such as bug tracking, feature requests, task management for every project.

Essential components of the GitHub are:

- Repositories
- Branches
- Commits
- Pull Requests
- Git (the version control tool GitHub is built on)

Advantages of GitHub

GitHub can be separated as the Git and the Hub. GitHub service includes access controls as well as collaboration features like task management, repository hosting, and team management.

- The key benefits of GitHub are as follows.
- It is easy to contribute to open source projects via GitHub.
- It helps to create an excellent document.
- You can attract the recruiter by showing off your work. If you have a profile on GitHub, you will have a higher chance of being recruited.
- It allows your work to get out there in front of the public.
- You can track changes in your code across versions.

Features of GitHub

GitHub is a place where programmers and designers work together. They collaborate, contribute, and fix bugs together. It hosts plenty of open source projects and codes of various programming languages.

Some of its significant features are as follows.

- Collaboration
- Integrated issue and bug tracking
- Graphical representation of branches
- Git repositories hosting
- Project management
- Team management
- Code hosting
- Track and assign tasks
- Conversations

GitHub vs. Git

Git is an open-source distributed version control system that is available for everyone at zero cost. It is designed to handle minor to major projects with speed and efficiency. It is developed to co-ordinate the work among programmers. The version control allows you to track and work together with your team members at the same workspace.

While GitHub is an immense platform for code hosting, it supports version controlling and collaboration. It allows developers to work together on projects.

It offers both distributed version control and source code management (SCM) functionality of Git. It also facilitates collaboration features such as bug tracking, feature requests, task management for every project.

GitHub vs. Git

GitHub	Git
It is a cloud-based tool developed around the Git tool.	It is a distributed version control tool that is used to manage the programmer's source code history.
It is an online service that is used to store code and push from the computer running Git.	Git tool is installed on our local machine for version controlling and interacting with online Git service.
It is dedicated to centralize source code hosting.	It is dedicated to version control and code sharing.
It is managed through the web.	It is a command-line utility tool.
It provides a desktop interface called GitHub desktop GUI.	The desktop interface of Git is called Git GUI.
It has a built-in user management feature.	It does not provide any user management feature
It has a market place for tool configuration.	It has a minimal tool configuration feature.

Bitbucket Vs. GitHub

Bitbucket is also a **web-based version control system** owned by **Atlassian**. It contains the source code and allows us to share it among developers. It offers both free accounts and commercial accounts (paid). It provides an unlimited number of repositories for free accounts. But it has some limitations for the free accounts like a private repository can have a maximum of five users.

Bitbucket was launched in 2008 to support Mercurial Projects. Atlassian acquired it in 2010, and from 2011 it also started to support Git.

It is much similar to GitHub. However, no one can be considered the best. Every service has a different feel to them, and it targets different demographics no matter where you're going to get excellent service and get your work done.

Let's see the similarities and differences between Bitbucket and GitHub.

Bitbucket Vs. GitHub

GitHub	Bitbucket
It has a user-friendly and fast interface.	It has a slick and clean interface, which provides a professional view.
It is limited to the Git repository only.	It is not limited to just Git, it also supports other version control systems like Mercurial , but it does not support svn .
It facilitates with graphs: pulse, contributors, commits, code frequency, members of it.	It assists with REST APIs to build third-party applications which can be used in any development language.
It is free for public repositories and paid for private repositories.	It is free for both private and public repositories. But we can have a maximum of five members for a private repository.
GitHub comes with a lot of features and allows you to create your workflows.	Bitbucket provides a more built-in option for flexibility.
We cannot make a private repository on free accounts.	We can create an unlimited private repository for up to five users.

How to Use GitHub?

This question is prevalent for the developers who have never used GitHub. This tutorial will assist you in overcoming this question. There is nothing to worry about, the necessary steps for the using GitHub are as follows:

- Create a GitHub account
- GitHub login
- GitHub Repository
- Create a repository
- Create a file
- Create Branches

Create a GitHub Account

The first step to explore the benefits of GitHub is to create a GitHub account. GitHub provides both the free and pro membership to its user. We can explore many exciting and useful things in its pro account. We can explore unlimited private repository and can control the user access.

To create a GitHub account, visit [GitHub](#).

Click on the **Signup** option at the upper right corner.





Fill the necessary details under sign up like your name, email address, and password. Then click on the **Next: Select a plan** option.



Create a GitHub Account

After selecting a plan, a confirmation link will send to your email address. Activate your account by clicking on the received link, and you are ready to go with GitHub.

	
Free \$0 USD	Pro \$7 USD
	Per month
The basics of GitHub for every developer	Pro tools for developers with advanced requirements
Choose Free	Choose Pro

GitHub Login

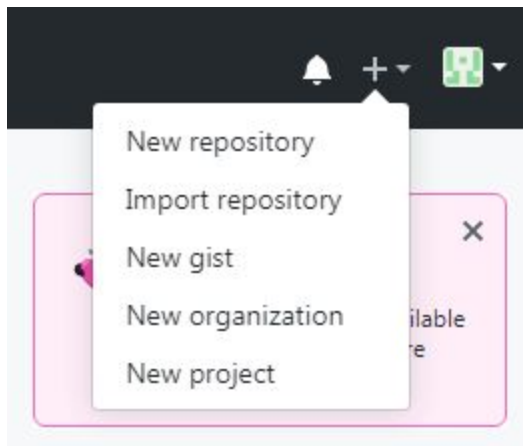
Log in to your GitHub account to use the GitHub service. To login to your account, click on the **Sign-in** option on the upper right corner. It will ask you for your email id and password. You can log in by entering your credentials. At your first login, the homepage will ask you to create your first repository and some other options like exploring the repository.

GitHub Repository

The repositories are the data structures used by GitHub to store metadata for files and directories. It encloses the collection of the files as well as the history of changes made to those files. Generally, the repository is considered a project folder. A single project can have more than one repository.

Create a repository

We can create an unlimited public repository and unlimited private repository (For the pro user) on GitHub. To create a repository on GitHub, click on the '+' symbol on the upper right corner on the login screen.



There are some other options available like import repository, gist, organization, and new project. To create a repository, choose **New repository** option from the given list. When you first log in to your account, you will see the UI as follows:

Create a repository

GitHub asks you to learn Git and GitHub without any code. It will ask you to read the hello world guide for the first uses. Also, you can create a repository (Project) from here.


Click on the new repository option and then fill the required details like repository name, description, and select the access of this repository. You can also initialize the repository with a README file. After filling all the details, click on the **Create Repository** option. It will create a repository for you. Consider the below image:

Hence, we have created a public repository.

Create a new repository

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

Owner

 pune2016 ▾

Repository name *

/

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

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

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



Initialize this repository with a README

This will let you immediately clone the repository to your computer.

Add .gitignore: None ▾

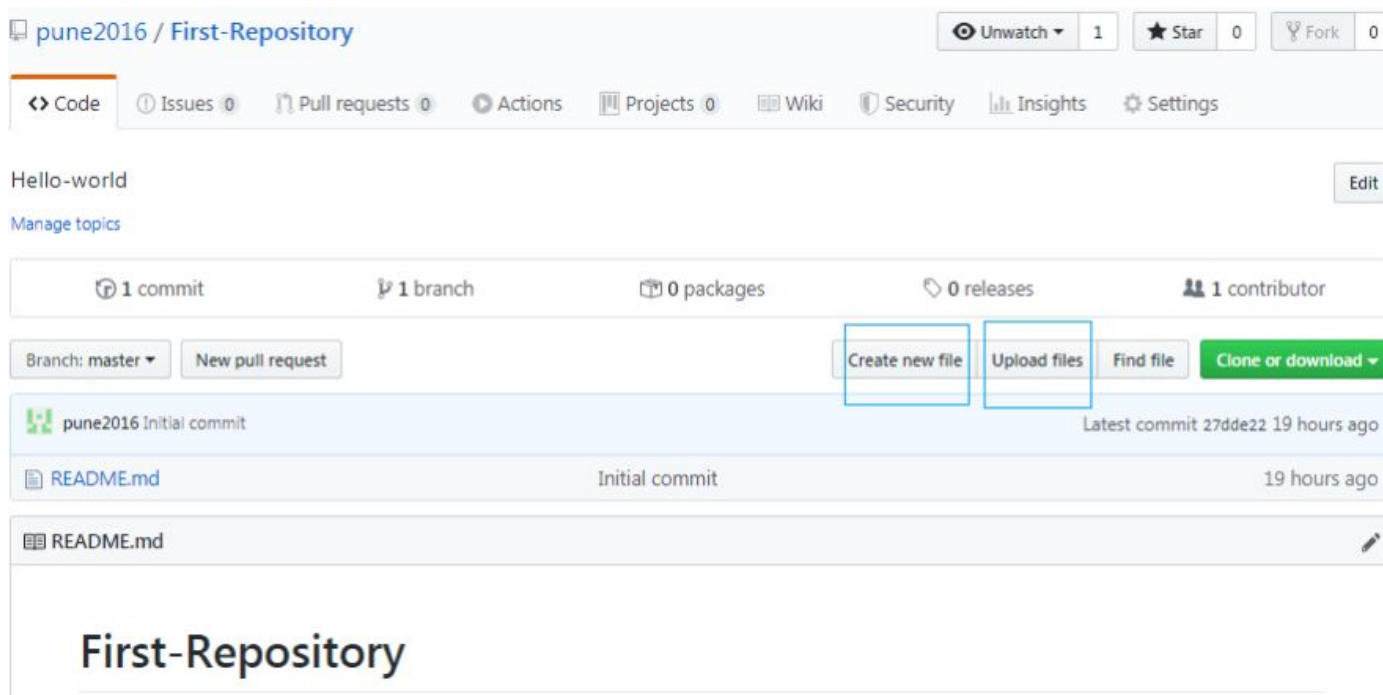
Add a license: None ▾



Create repository

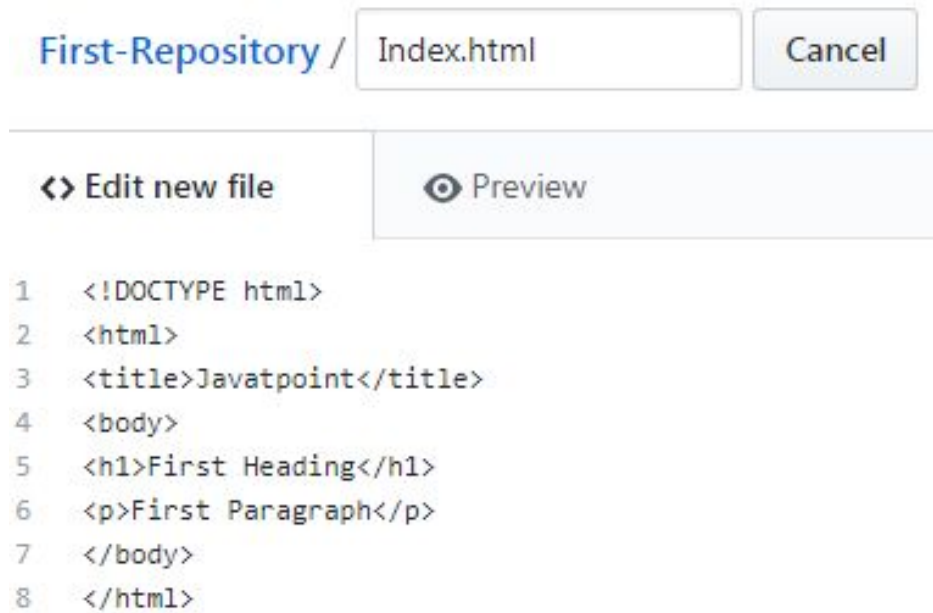
Create a file

In GitHub, creating a file is a straight forward process. Let's create a file in our newly created repository. Consider the below snap of our repository:



Create a file

There are distinct options available to add files to the repository. GitHub allows us to design and upload files. To create a file, click on the '**Create new file**' option. It will open a file structure, and it will look like as follows:



Create a file


Enter the file name on the box and type the code on the editor area.


At the bottom of the page, the commit options are available. Consider the below snap:

Commit new file

Index file

I have added first file to new project

☒  Commit directly to the `master` branch.

☐  Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

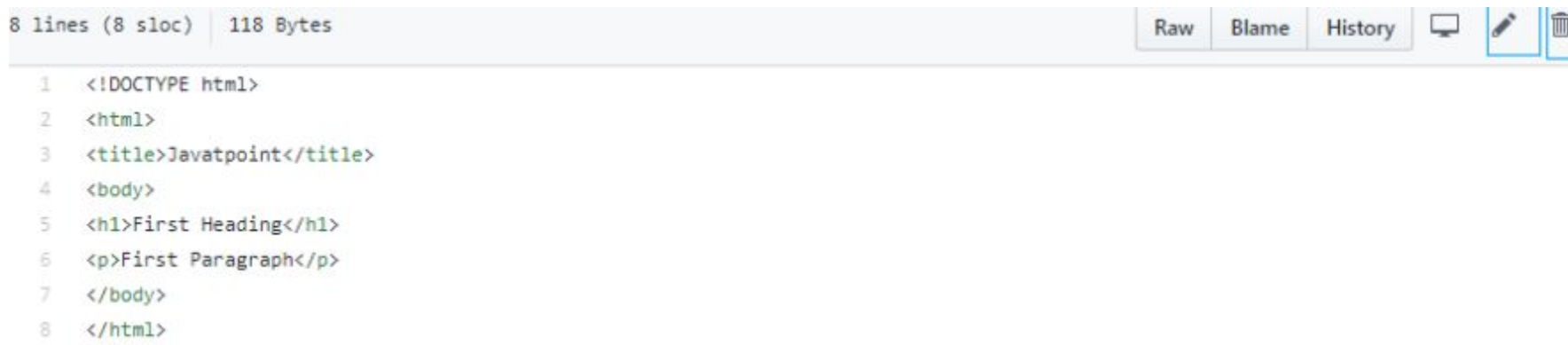
Commit new file

Create a file

In the above image, we can give the commit message in the first text area and the description in the second text area. Also, we can specify whether we want to commit it to the master branch or want to create a new branch.

Click on the '**commit new file**' option. We have successfully added and committed a new file to our repository.

We can edit and delete this file from our project. There are many options available, like edit, delete, Raw, Blame, and history. Consider the below snap of the file.



```
8 lines (8 sloc) 118 Bytes
1 <!DOCTYPE html>
2 <html>
3 <title>Javatpoint</title>
4 <body>
5 <h1>First Heading</h1>
6 <p>First Paragraph</p>
7 </body>
8 </html>
```

Hence we have learned how to create a file and commit changes. Now we will see how to create a new branch.

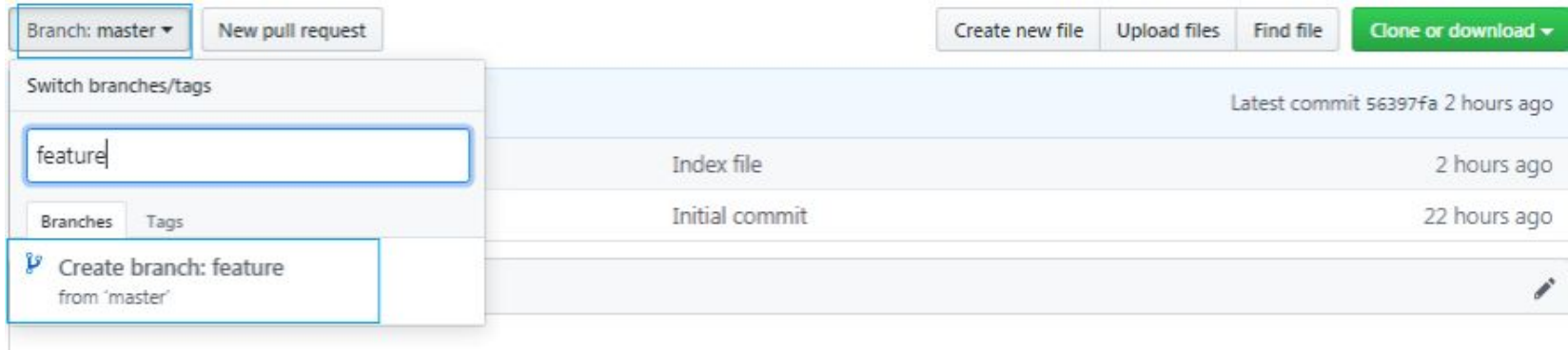
How to create a new branch?

Branches are the pointer to snapshots of changes. Branches are created for a particular purpose like fixing a bug, testing, release, and more. To understand the types of branches, visit [Git Flow](#).

It is complex to merge the unstable code with the main code base and also facilitates you to clean up your future history before merging with the main branch.

The **master branch** is the **default** branch of the repository.

Let's understand how to create a branch in GitHub. To create a new '**feature**' branch, drag the branch option under the repository. This option will list the available branches. A search option is available under the branch. It will search for the requested branch if it is not in the repository, then it will create a new branch by the given name. Consider the below image:



Connect GitHub with your computer

GitHub Download

We can connect the GitHub with our computer. GitHub allows downloading its desktop application. Also, we can connect the GitHub repository with our computer by **Git**.

There are different kinds of audiences, some people love Git commands, and some love the attractive user interface for their work. The people who love the user interface, the GitHub desktop application is one of the best Git clients for them.

Let's see some of its features.

Features of GitHub Desktop

The desktop application of GitHub has incredible features that make collaboration easy for the contributor. Some of its attractive features are as follows:

- Attributing commits with collaborators easily.
- Checkout branches and create a PR(pull request)
- Broad editor and shell integration
- Open-source

Let's understand how to install it on your system.

Connect GitHub with your computer

GitHub Desktop for Windows

To setup GitHub Desktop, we must already have a GitHub account. It is a fast and straight forward way to contribute to GitHub. It is developed to make straight-forward all the processes of GitHub.

GitHub Desktop is an open-source that can be downloaded. If we talk about its technical specification, it is written in Typescript and uses react. It is available for Microsoft Windows or macOS operating systems.

Below are the steps to install the GitHub desktop:

Connect GitHub with your computer

Step1: Visit <https://desktop.github.com> for Github desktop. To download the setup, click on the '**Download for Windows (64bit)**' option. Consider the below image:



Connect GitHub with your computer

Step2: Installation:

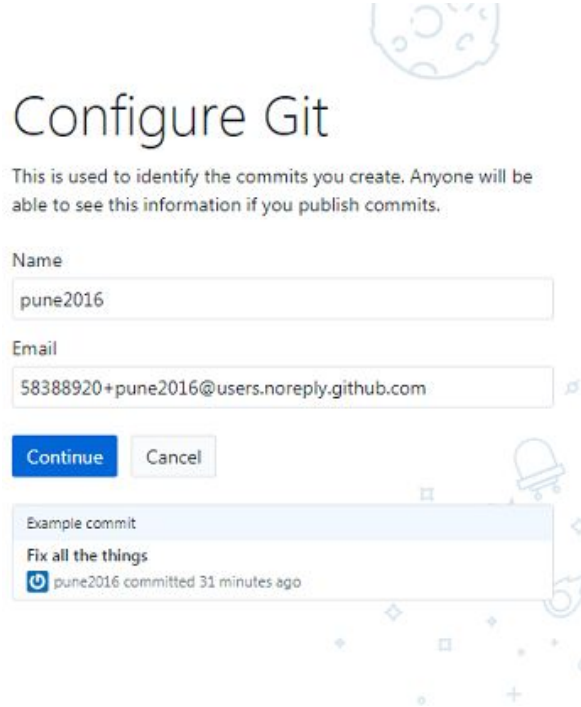
Install the GitHub desktop application by running the installer file. There we can see options like create an account, sign in. However, we can skip this step. It looks like as follows:



Connect GitHub with your computer

Step3: Setting up the desktop application:

The next step after installing the GitHub is to customize it. However, we can skip this step. Sign in to your GitHub account. The configuration will look like as follows:



The image shows the 'Configure Git' web interface. At the top, there's a title 'Configure Git' and a subtitle 'This is used to identify the commits you create. Anyone will be able to see this information if you publish commits.' Below this are two input fields: 'Name' with the value 'pune2016' and 'Email' with the value '58388920+pune2016@users.noreply.github.com'. There are 'Continue' and 'Cancel' buttons. At the bottom, there's a section for 'Example commit' showing a commit message 'Fix all the things' and a commit hash 'pune2016 committed 31 minutes ago'.

Configure Git

This is used to identify the commits you create. Anyone will be able to see this information if you publish commits.

Name
pune2016

Email
58388920+pune2016@users.noreply.github.com

Continue Cancel

Example commit
Fix all the things
pune2016 committed 31 minutes ago



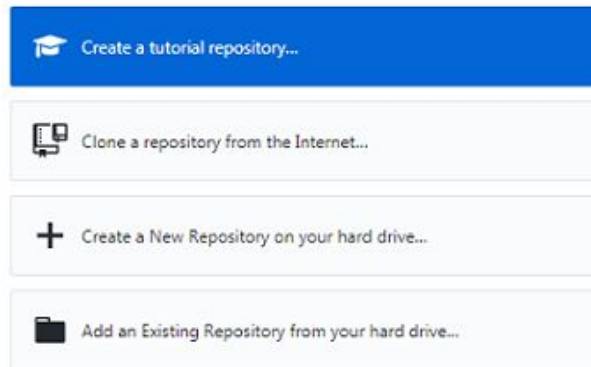
Connect GitHub with your computer

Now we are all set to go with Desktop application of GitHub. The UI of the GitHub desktop will look like as follows:

We can create, clone, or upload a repository to our project with GitHub desktop. It also lists your existing GitHub repositories.

Let's get started!

Add a repository to GitHub Desktop to start collaborating



ProTip! You can drag & drop an existing repository folder here to add it to Desktop



Filter your repositories

Your repositories

pune2016/First-Repository

How to clone (copy) GitHub repository to our PC

There are many ways to copy a GitHub project. We can make a copy of a GitHub project on our local machine. To do so either, we can use the GitHub desktop application or Git Bash. Since here we are talking about GitHub so let's see how to copy by GitHub desktop application.

To copy from GitHub desktop application, follow the below steps:

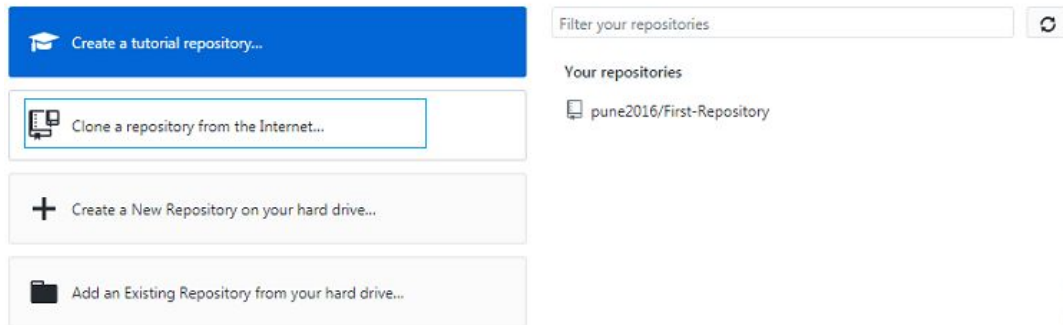
How to clone (copy) GitHub repository to our PC

Step1: Open the GitHub desktop

Open the application, if you have not logged in yet, log in to the application by using your account credential. Select the option '**clone a repository from the internet.**' Consider the below image:

Let's get started!

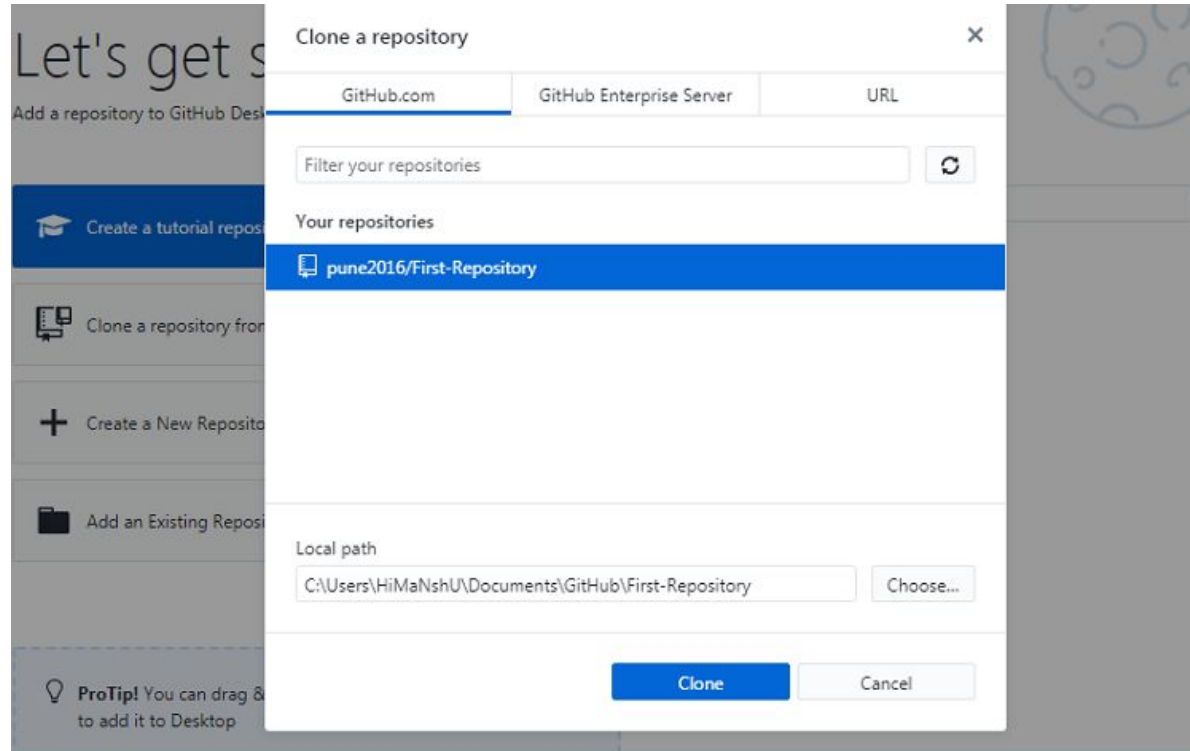
Add a repository to GitHub Desktop to start collaborating



How to clone (copy) GitHub repository to our PC

Step2: Select the repository

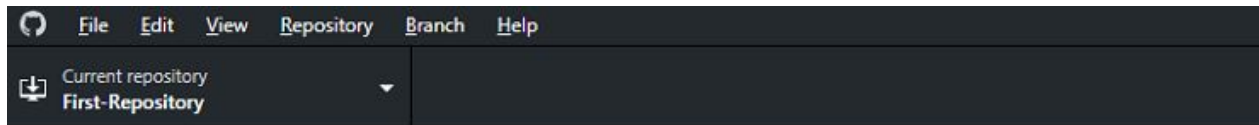
After selecting the clone option, it will list all the available repository on your GitHub account. Consider the below output:



How to clone (copy) GitHub repository to our PC

Step3: Clone the Repository

Select your desired directory which you want to clone and click on the **clone** option. It will start copying the project. Consider the below image:



How to clone (copy) GitHub repository to our PC

It will take a while to copy the project.

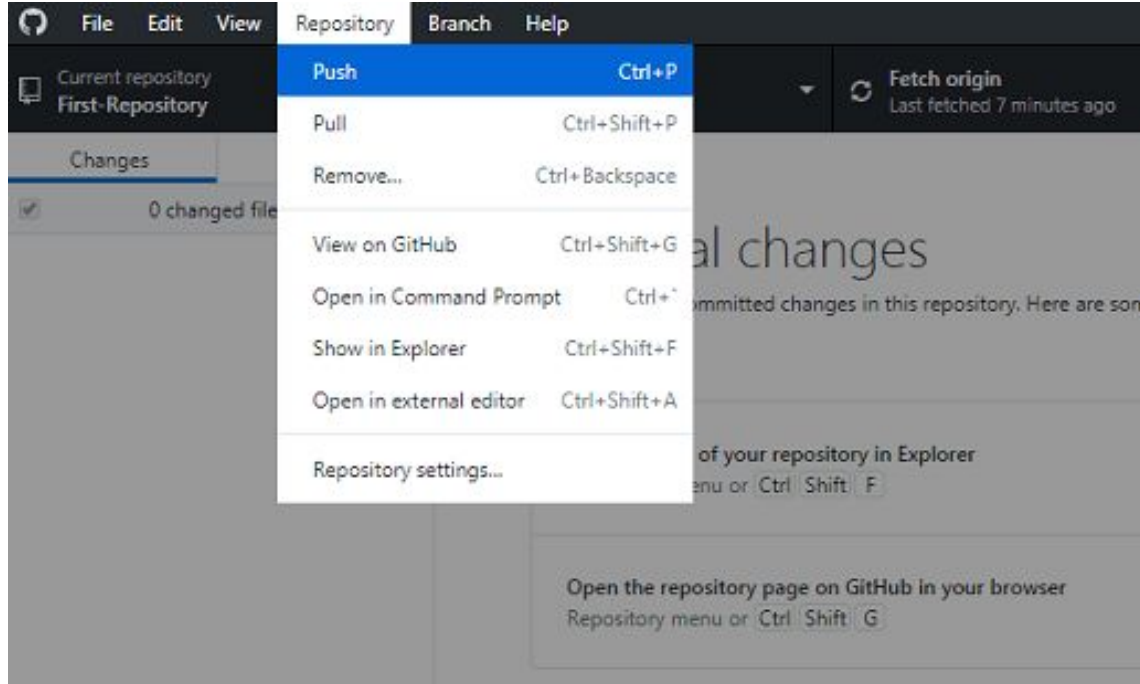
To clone a repository by Git Bash, run the clone command as follows:

```
$ git clone <repository URL>
```

How to pull from GitHub

We need to pull the data from GitHub to keep the local repository updated with the GitHub repository. Suppose any team member made or propose changes for our project. If you want to merge with your local directory, make a pull.

We can pull the updates from the GitHub project by GitHub desktop and Git Bash. To pull the changes by GitHub desktop, navigate to the **repository menu**, and select the pull option.



How to pull from GitHub

It will pull the GitHub repository.

To pull the repository by the Git Bash, run the below command:

```
$ git pull origin master
```

What is GitHub pull request?

Pull request is a process for a developer to notify team members that they have completed a feature. Once their feature branch is ready, the developer files a pull request via their remote server account. Pull request announces all the team members that they need to review the code and merge it into the master branch.

How to commit and push to GitHub

Pushing is the act of transferring the local changes to GitHub. Suppose we made some changes to your local repository and share it on GitHub. To do so, we can push the changes.

We can commit changes from our GitHub desktop application as well as Git Bash. To commit the changes from the GitHub desktop application, follow the below steps:

How to commit and push to GitHub

Step1: Open the file explorer

Open the file explorer from GitHub desktop. To open the file explorer, either press the '**Ctrl +Shift +F**' keys together, or we can select it from the menu.

Step2: Made the changes

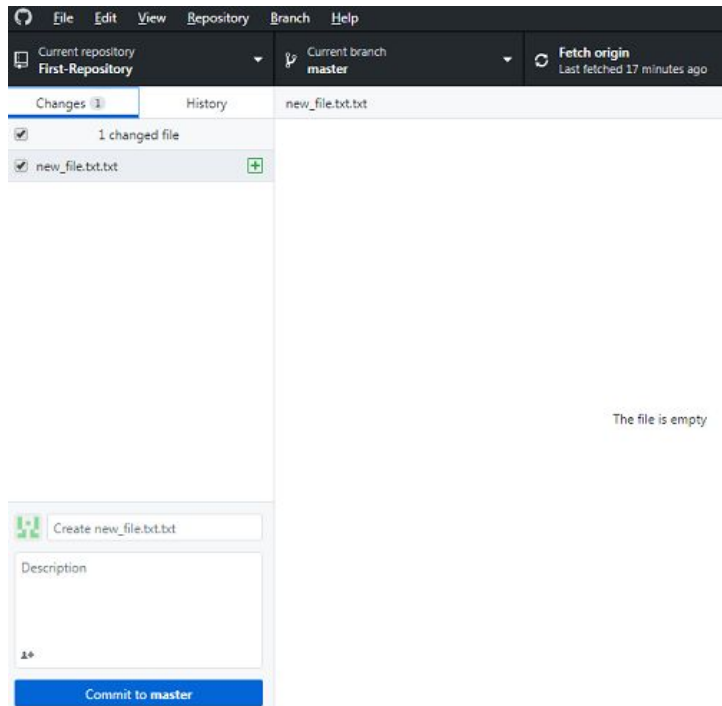
Now, you are in the file explorer made the desired changes. In our case, we have created a file **new_file.txt**.

How to commit and push to GitHub

Step3: Commit the changes

It is necessary to commit the changes to share it on GitHub. To commit the changes, open GitHub desktop; here, we can see the changes that we made. Consider the below image:

To commit the changes, type commit message, and description. After that, click on the commit option as displayed on the above image. Now, you have successfully made a commit. The next step is to push it to GitHub account.



How to commit and push to GitHub

Step4: Push the changes to GitHub account

The change that we have made is now ready to be pushed on GitHub account. The GitHub desktop application starts displaying a notification like a commit is ready to be pushed. Consider the below image:

We can use the '**Ctrl + P**' keys or '**push origin**' option to push the changes to the GitHub repository.

Now, we have these changes in our GitHub repository.

To push the changes by Git Bash, run the below command:

```
$ git push origin master
```

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.



Push 1 commit to the origin remote

You have one local commit waiting to be pushed to GitHub.

Always available in the toolbar when there are local commits waiting to be pushed or

Ctrl P

Push origin

View the files of your repository in Explorer

Repository menu or Ctrl Shift F

Show in Explorer

Open the repository page on GitHub in your browser

Repository menu or Ctrl Shift G

View on GitHub

GitHub fork

A fork is a rough copy of a repository. Forking a repository allows you to freely test and debug the changes without affecting the original project. One of the excessive use of forking is to propose changes for bug fixing. To resolve an issue for a bug that you found, you can:

- Fork the repository.
- Make the fix.
- Forward a pull request to the project owner.

When to Use Git Fork

Generally, forking a repository allows us to experiment on the project without affecting the original project. Following are the reasons for forking the repository:

- Propose changes to someone else's project.
- Use an existing project as a starting point.

How to fork a repository?

The forking and branching are excellent ways to contribute to an open-source project. These two features of Git allows the enhanced collaboration on the projects.

Forking is a safe way to contribute. It allows us to make a rough copy of the project. We can freely experiment on the project. After the final version of the project, we can create a pull request for merging.

It is a straight-forward process. Steps for forking the repository are as follows:

- Login to the GitHub account.
- Find the GitHub repository which you want to fork.
- Click the Fork button on the upper right side of the repository's page.

We can't fork our own repository. Only shared repositories can be a fork.