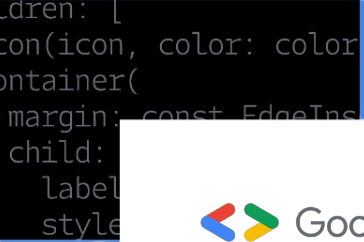


```
dren: [
con(icon, color: color
ontainer(
 margin: const EdgeIns
 child:
   label
   style
```

Google Developer Student Clubs
Institut Teknologi Telkom Surabaya

# Git Study Jam

Leveraging Git as Your Version Control System

Ahmad Mu’min Faisal
Core Team, GDSC ITTS

Ahmad Mu'min Faisal

Core Team at Google Developer Student Clubs Institut Teknologi Telkom Surabaya

"A dedicated Flutter development enthusiast with a profound love for Linux and Free/Libre and Open-Source Software (FLOSS)"

- Division Head of PKM Center – Punggawa Inspiratif
- Lecturer Assistant of DSA, Comp. System Programming, and OOPs
- Head of Public Relation – Kelompok Linux Arek Suroboyo

- 2nd Place – IT Convert 2023
- 1st Place – PLAYBOX Season 4
- 3x Funded – PKM 2022-2023

in ahmadmfaisal
fzl-22
contact@ahmadfaisal.online
fzl-22

Google Developer Student Clubs
Institut Teknologi Telkom Surabaya

bit.ly/discord-gdsc-itts

```
lookup.KeyValue
f.constant(['en
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

Google Developer Student Clubs
Institut Teknologi Telkom Surabaya

## What is Git?

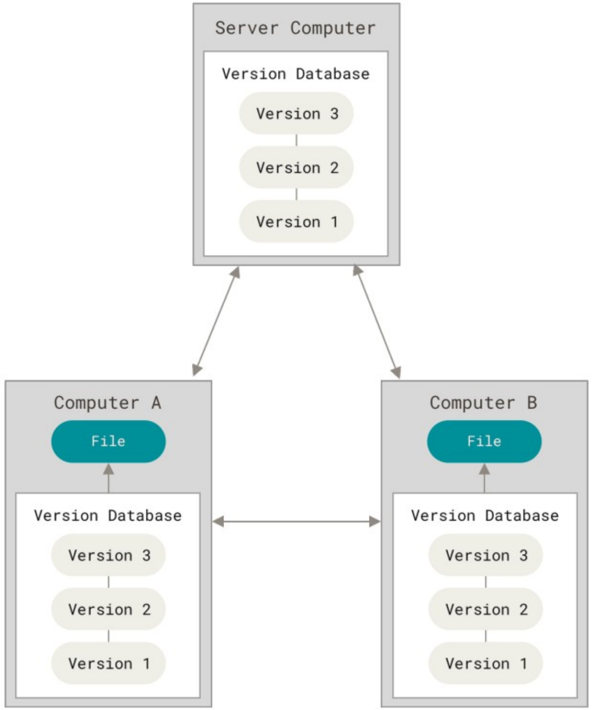
```
lookup.KeyValue
f.constant(['em
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

## What is Git?

Git is a **version control system**, a software to **record changes** to a files or a set of files overtime.

Git also makes it easier for developers to collaborate because it is a **Distributed Version Control System**.

## Distributed Version Control System

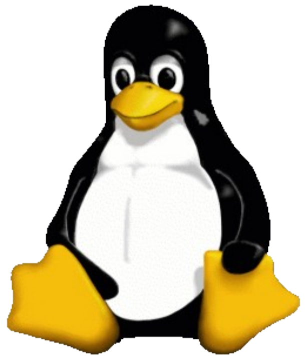


## Brief History of Git

Git is originally created by **Linus Torvalds** (the creator of Linux kernel) in 2005, because:

1. BitKeeper revoked its *free-of-charge* status.
2. Linux developer community is getting bigger, so they need more sophisticated version control system.

Currently, Git is maintained by **Junio Hamano** (software engineer at Google).

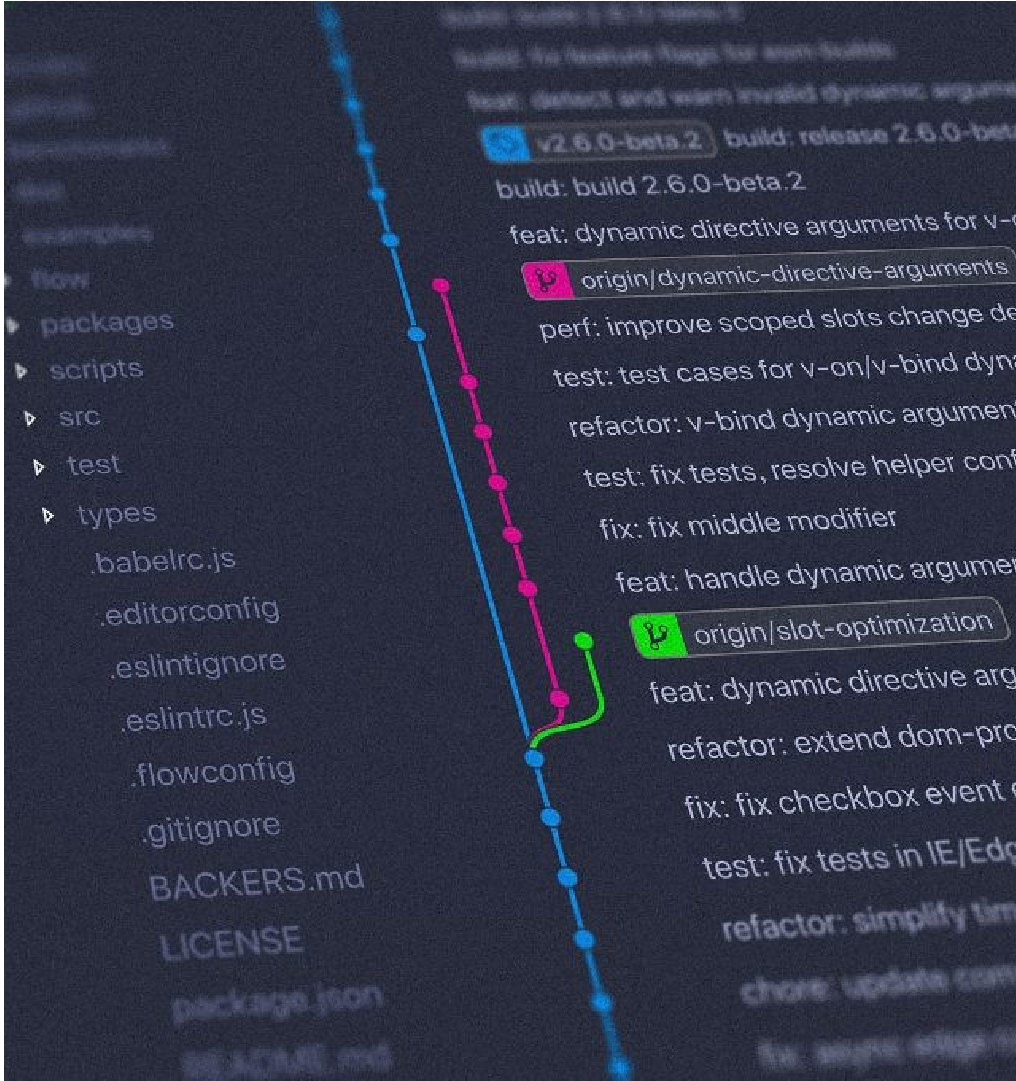


## Why We Use Git?

Some of the reasons why we should learn Git are:

1. **Keep track of changes** to source code.
2. Facilitate **collaboration** between developers.
3. Facilitate contributions to **open source projects**.
4. The **most widely used** Version Control System.

Google Developer Student Clubs



v2.6.0-beta.2 build: release 2.6.0-beta
build: build 2.6.0-beta.2
feat: dynamic directive arguments for v-
origin/dynamic-directive-arguments
perf: improve scoped slots change de
test: test cases for v-on/v-bind dyn
refactor: v-bind dynamic argumen
test: fix tests, resolve helper conf
fix: fix middle modifier
feat: handle dynamic argume
origin/slot-optimization
feat: dynamic directive arg
refactor: extend dom-pro
fix: fix checkbox event
test: fix tests in IE/Edg
refactor: simplify tim

Google Developer Student Clubs
Institut Teknologi Telkom Surabaya

# Everything You Need to Know…

```
lookup.KeyValue
f.constant(['em
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

## How Git Works

Git thinks of its data like a **series of snapshots**.

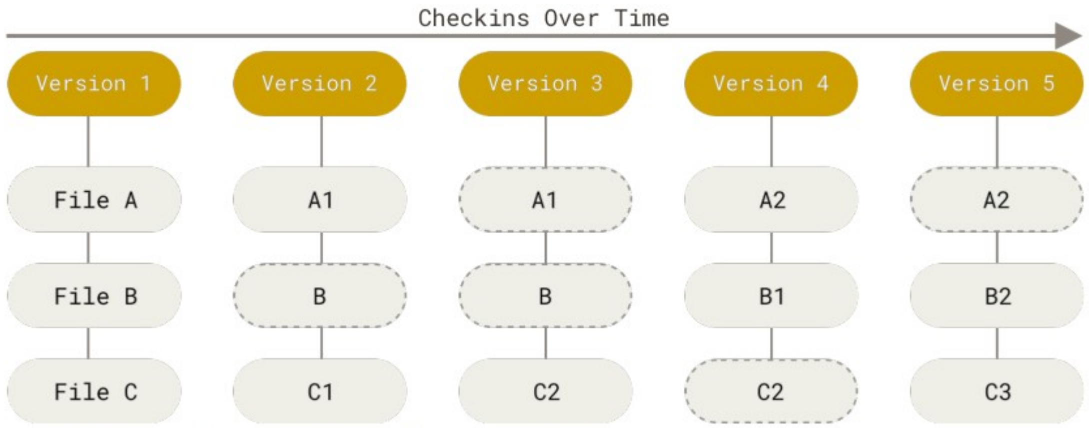


*Figure 5. Storing data as snapshots of the project over time*

## The 3 Main Sections

1. **Working directory**,project version that is ready to use or modify
2. **Staging Area**, stores information about what will go into the next commit.
3. **Git Directory**, stores metadata and committed snapshot

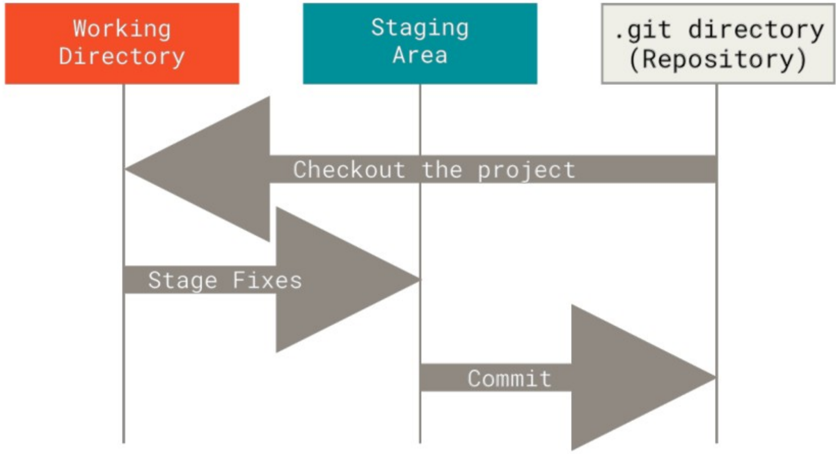


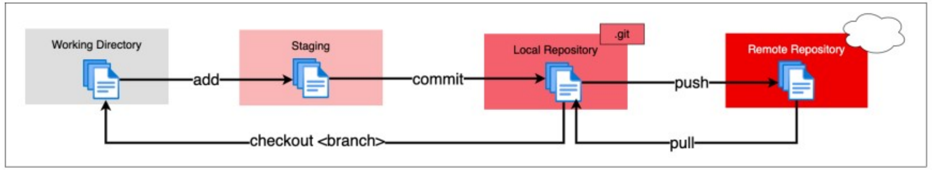
*Figure 6. Working tree, staging area, and Git directory*

Google Developer Student Clubs

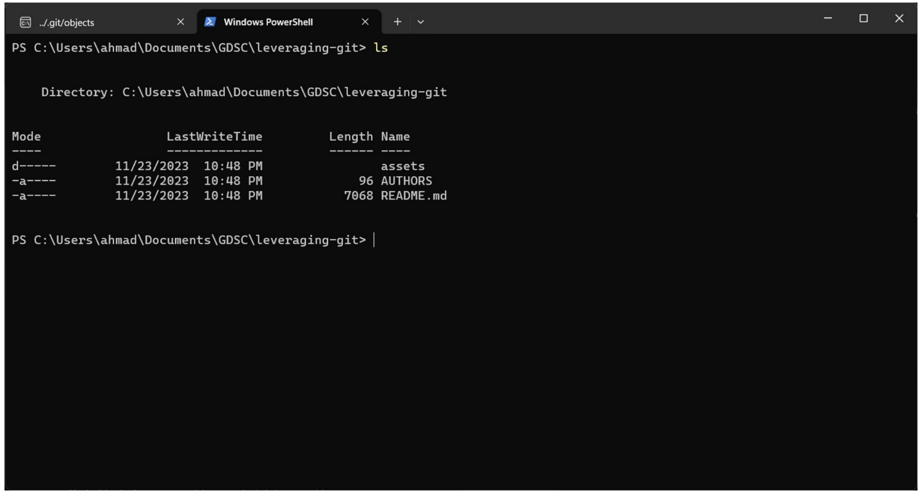
## Git Basic Workflow

Those previous Git sections lead to this basic Git workflow:

1. **Modify file** in the working directory
2. Selectively **stage changes** to the staging area for the next commit
3. Do a **commit**, store the changes added to the staging area as a snapshot to the Git directory.



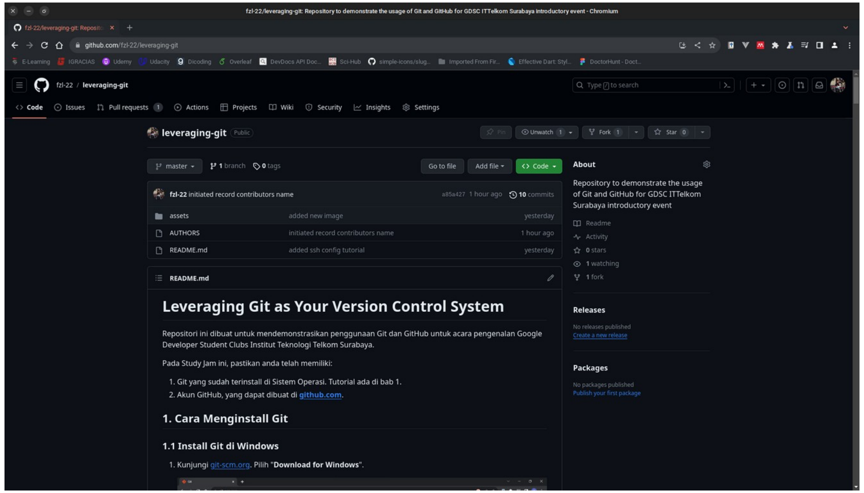
## Local Repository



```
PS C:\Users\ahmad\Documents\GDSC\leveraging-git> ls


    Directory: C:\Users\ahmad\Documents\GDSC\leveraging-git


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        11/23/2023  10:48 PM                assets
-a----        11/23/2023  10:48 PM             96 AUTHORS
-a----        11/23/2023  10:48 PM           7068 README.md


PS C:\Users\ahmad\Documents\GDSC\leveraging-git> |
```

Your Git repository stored on your local machine

## Remote Repository



leveraging-git

fzl-22 initiated record contributors name a85a427 1 hour ago 10 commits

| | | |
|---|---|---|
| assets | added new image | yesterday |
| AUTHORS | initiated record contributors name | 1 hour ago |
| README.md | added ssh config tutorial | yesterday |

About

Repository to demonstrate the usage of Git and GitHub for GDSC ITTelkom Surabaya introductory event

Releases

No releases published

Packages

No packages published

README.md

Leveraging Git as Your Version Control System

Repositori ini dibuat untuk mendemonstrasikan penggunaan Git dan GitHub untuk acara pengenalan Google Developer Student Clubs Institut Teknologi Telkom Surabaya.

Pada Study Jam ini, pastikan anda telah memiliki:

1. Git yang sudah terinstall di Sistem Operasi. Tutorial ada di bab 1.
2. Akun GitHub, yang dapat dibuat di github.com.

1. Cara Menginstall Git

1.1 Install Git di Windows

1. Kunjungi git-scm.org. Pilih "**Download for Windows**".

Your Git repository hosted on Git Server such as Github, Gitlab, and etc.

Google Developer Student Clubs

Google Developer Student Clubs

Institut Teknologi Telkom Surabaya

# Let’s Publish!

**By implementing Git basic workflow**

```
lookup.KeyValue
f.constant(['em
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

## Create Directory and Initiate Git Repository

First, create a directory named **‘git-demo’**. Then populate this directory with some files, for example:

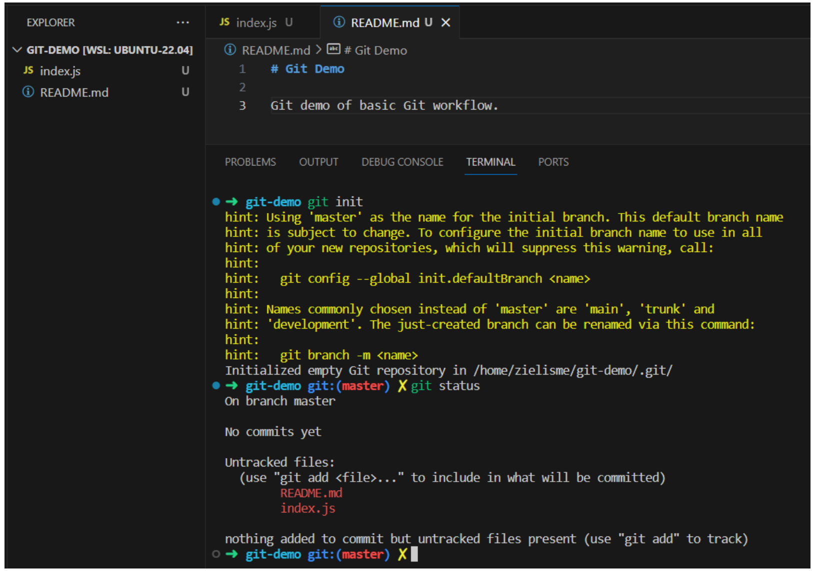
- **index.js** -> console.log(“Hello, World”);
- **README.md** -> # Git Demo

Then, initiate Git repository with this command on terminal:

**git init**

This command will initialize current directory as Git repository with initial branch name as **master**

Let’s, check the output of **git status**



```
# Git Demo

Git demo of basic Git workflow.
```

```
➜ git-demo git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/zielisme/git-demo/.git/
➜ git-demo git:(master) ✗ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        README.md
        index.js

nothing added to commit but untracked files present (use "git add" to track)
➜ git-demo git:(master) ✗
```

Google Developer Student Clubs

# Add Changes to Staging Area

The output of **git status** said that there is an untracked file.

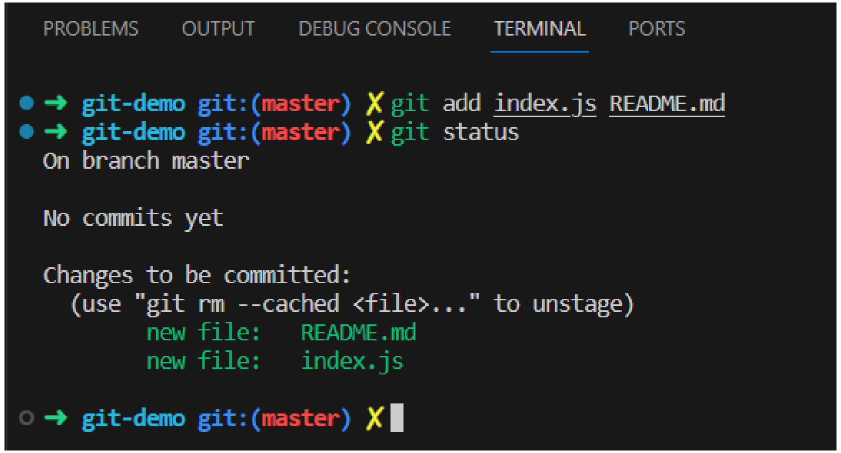
Untracked files mean that Git found a changes from a file that is **not tracked in the previous commit**.

So, **add** those files to the **staging area** for the next commit!

**git add index.js README.md**

Voila! Your changes is **ready to be committed**.

Let's, check again the **git status**!



```
● ➜ git-demo git:(master) ✗ git add index.js README.md
● ➜ git-demo git:(master) ✗ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   README.md
        new file:   index.js

○ ➜ git-demo git:(master) ✗
```

Google Developer Student Clubs

## Commit Your Staged Changes!

The output of **git status** said that our changes is **ready to be committed**.

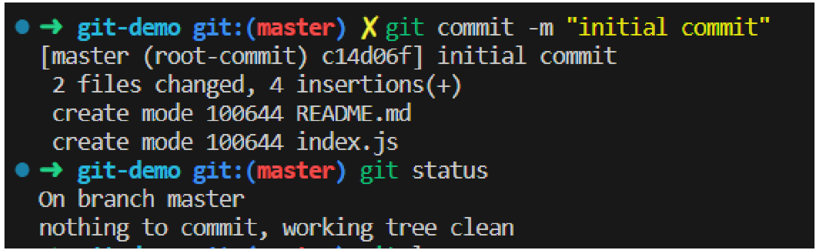
So, just run this command:

**git commit –m “initial commit”**

The string after –m flag is called “**commit message**”.

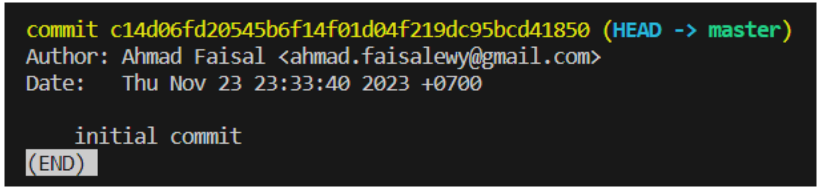
Make sure you gives **proper commit messages** according to the **context of your changes**.

Let’s, check the **git status** again!



```
● ➜ git-demo git:(master) ✗ git commit -m "initial commit"
  [master (root-commit) c14d06f] initial commit
   2 files changed, 4 insertions(+)
   create mode 100644 README.md
   create mode 100644 index.js
● ➜ git-demo git:(master) git status
  On branch master
  nothing to commit, working tree clean
```

Let’s see the **git log**...



```
commit c14d06fd20545b6f14f01d04f219dc95bcd41850 (HEAD -> master)
Author: Ahmad Faisal <ahmad.faisalewy@gmail.com>
Date:   Thu Nov 23 23:33:40 2023 +0700

    initial commit
(END)
```

Google Developer Student Clubs

# Create Remote Repository

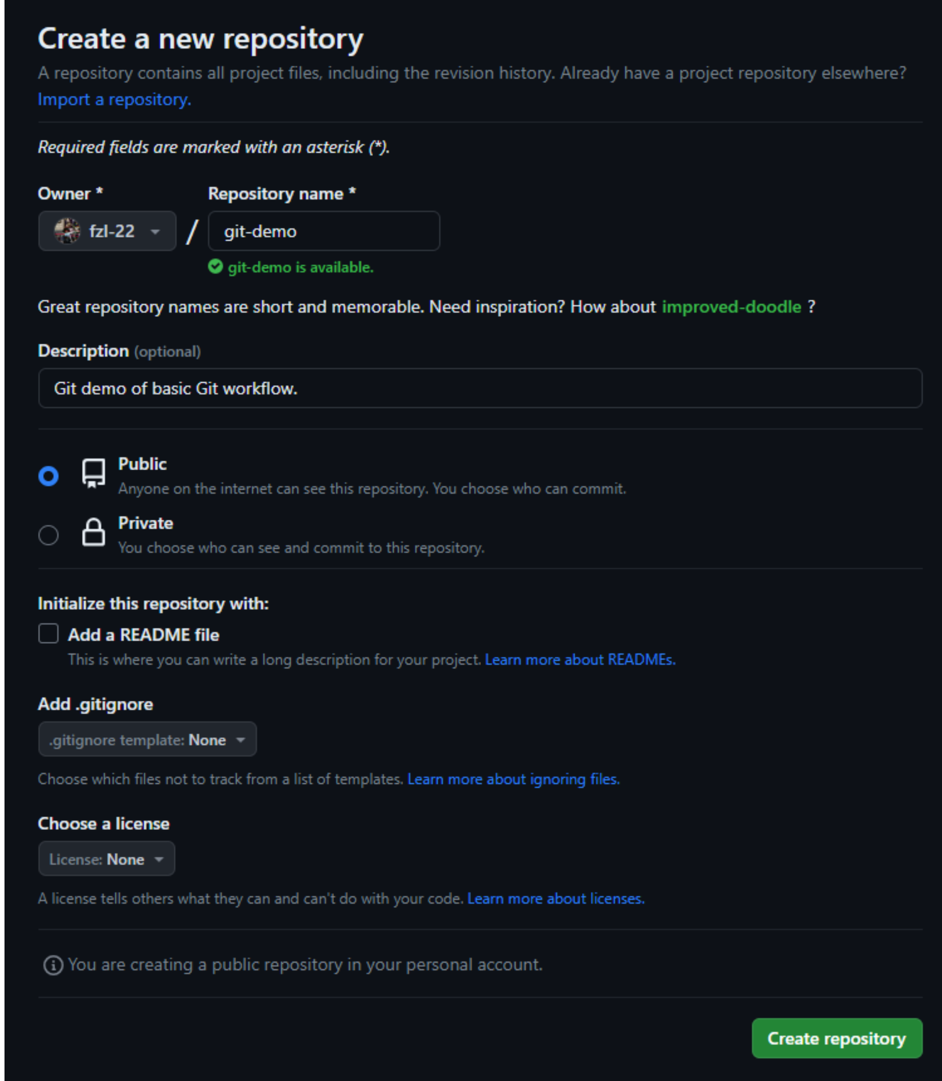
The output of **git status** in the previous section said that our **working tree is clean**. So, **we can push our repository** to the remote repository.

Before that, **create the remote repository** first.

Visit github.com, click "**New**" in the left sidebar to create new Github repository.

**Don't add README file** because it can cause default branch name conflict.

Google Developer Student Clubs


Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

*Required fields are marked with an asterisk (*).*

Owner * fzl-22 / Repository name * git-demo

git-demo is available.

Great repository names are short and memorable. Need inspiration? How about improved-doodle ?

Description (optional): Git demo of basic Git workflow.

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private — You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file — This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore — .gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license — License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

You are creating a public repository in your personal account.

# Connect Local to Remote Repository

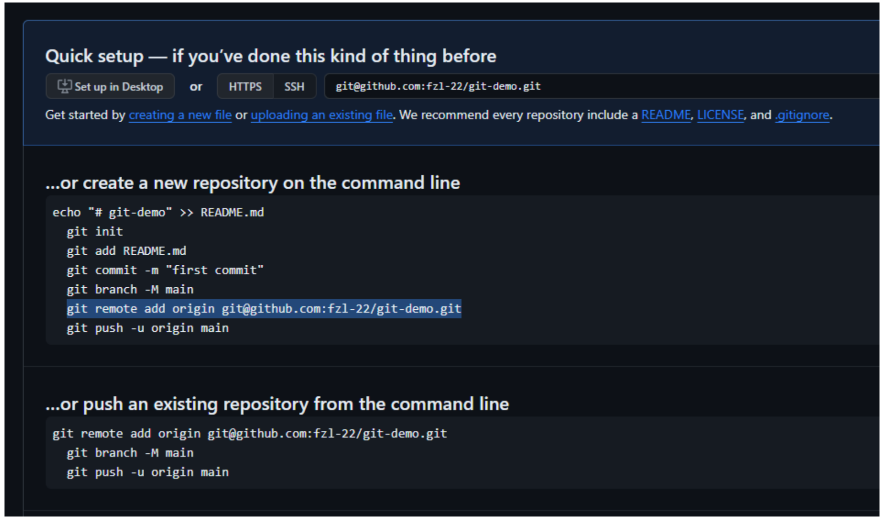
After the Github repository is created, you will be prompted to do some command-line things...

Just don't do it..., because we **already did most** of the given commands previously.

Just run this one and only command (that I have highlighted):

**git remote add origin [git@github.com:fzl-22/git-demo.git]()**

This command adds remote repository named **origin** for your local repository.



Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH git@github.com:fzl-22/git-demo.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# git-demo" >> README.md
  git init
  git add README.md
  git commit -m "first commit"
  git branch -M main
  git remote add origin git@github.com:fzl-22/git-demo.git
  git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:fzl-22/git-demo.git
  git branch -M main
  git push -u origin main
```

To verify it, please run **git remote -v** or **git remote --verbose**.

## Push Local Commits to Remote Repository

To push your local commit, just run this following command:

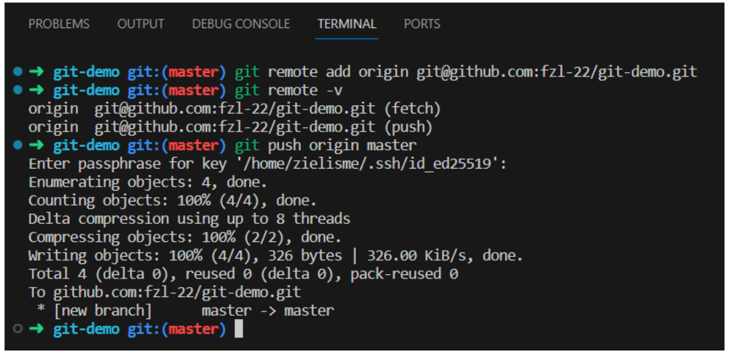
**git push origin master**

If you 'by-any-circumstances' have more remote repository or local branches, You can change **origin** with another remotes or **master** with another local branch.

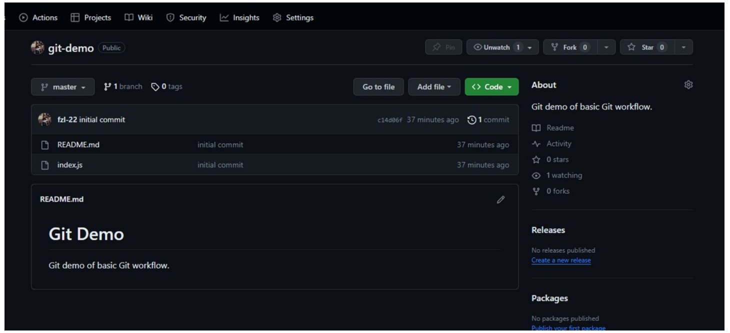
After that, reload your browser and you will find your remote repo is **in sync** with your local repo!

Note that you will be prompted to enter your SSH passphrases.

Google Developer Student Clubs



```
● ➜ git-demo git:(master) git remote add origin git@github.com:fzl-22/git-demo.git
● ➜ git-demo git:(master) git remote -v
  origin  git@github.com:fzl-22/git-demo.git (fetch)
  origin  git@github.com:fzl-22/git-demo.git (push)
● ➜ git-demo git:(master) git push origin master
  Enter passphrase for key '/home/zielisme/.ssh/id_ed25519':
  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% (4/4), 326 bytes | 326.00 KiB/s, done.
  Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
  To github.com:fzl-22/git-demo.git
   * [new branch]      master -> master
○ ➜ git-demo git:(master) █
```



fzl-22 initial commit c14d06f 37 minutes ago 1 commit

| | | |
|---|---|---|
| README.md | initial commit | 37 minutes ago |
| index.js | initial commit | 37 minutes ago |

README.md

Git Demo

Git demo of basic Git workflow.

## Let’s do **some changes**!

Google Developer Student Clubs
Institut Teknologi Telkom Surabaya

## Let’s Do Some Contributions, Shall We?

```
lookup.KeyValue
f.constant(['em
=tf.constant([0
.lookup.StaticV

_buckets=5)
```

## Project Setup

Let’s fork my Github repository at:

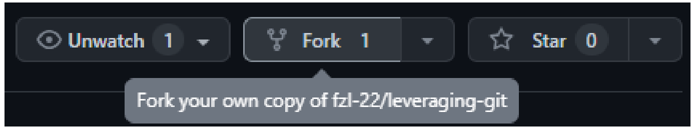
**https://github.com/fzl-22/leveraging-git**

Then, clone your forked repository (**not my original repo**) to your local machine using **SSH** with this command:

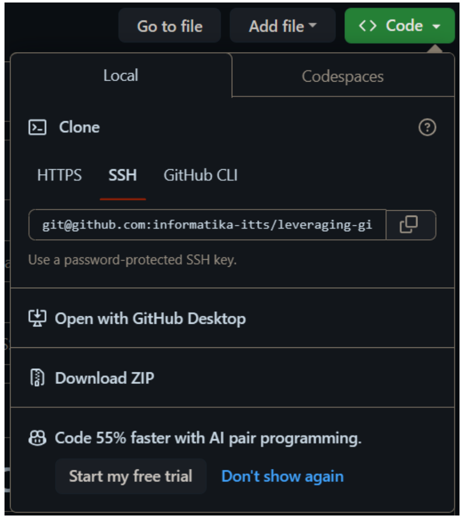
**git clone** git@github.com:{username}/leveraging-git.git

After that, you will have the **copy** of your forked remote repository on your **local machine**.

By doing this, any changes you committed and pushed will **affect only to your remote repository**, not my original/upstream repository.



Forking: A process to copy other people’s repository (upstream/original repo) as your Github repository.



```
git@github.com:informatika-itts/leveraging-gi
```

Use a password-protected SSH key.

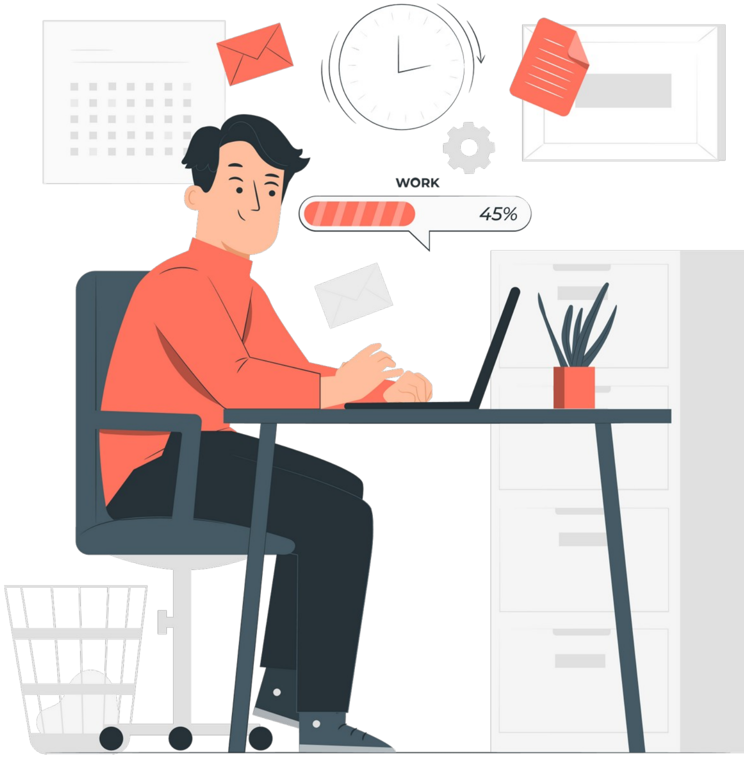
Code 55% faster with AI pair programming.

Cloning: A process to copy a repository into your local machine.

## Task ⏪

1. Locate a file named “CREDITS”, write your full name in the last line of the file.
2. Stage your changes, then do commit.
3. Push your commit to the branch master.

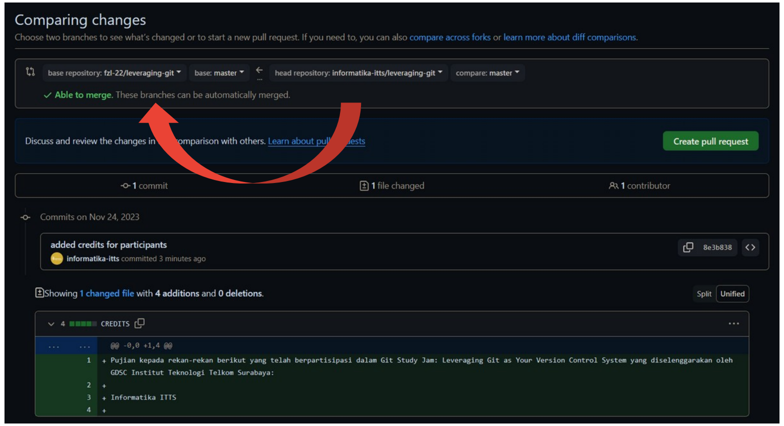
If you have completed these three tasks, you can send “**Pull Request**” to my upstream repository.



# Pull Request to Upstream Repository

**Pull Request** is opened to propose a changes to a repository. The requested PR can be commented, approved, or rejected (by requesting another changes).

If your opened Pull Request is **approved**, your changes is **ready to be merged** in the upstream repository (and/or branch). But, it depends on how the repository branch rule is managed.



Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks or learn more about diff comparisons.

base repository: fzl-22/leveraging-git base: master ← head repository: informatika-itts/leveraging-git compare: master

✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in comparison with others. Learn about pull requests

Create pull request

1 commit 1 file changed 1 contributor

Commits on Nov 24, 2023

added credits for participants
informatika-itts committed 3 minutes ago 8e3b838

Showing 1 changed file with 4 additions and 0 deletions.

Split Unified

4 CREDITS

```
@@ -0,0 +1,4 @@
+ Pujian kepada rekan-rekan berikut yang telah berpartisipasi dalam Git Study Jam: Leveraging Git as Your Version Control System yang diselenggarakan oleh GDSC Institut Teknologi Telkom Surabaya:
+
+ Informatika ITTS
+
```

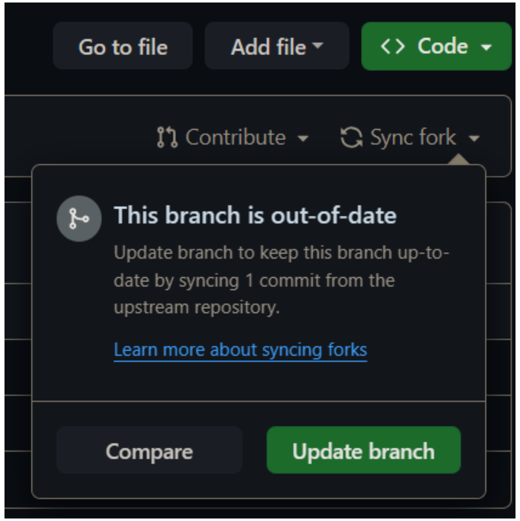
Create pull request to my upstream repository on branch master. But, you can also do a PR to different branch in the same repository.

Google Developer Student Clubs

## Sync Forked Repo to Upstream Repo

If your PR is approved and merged into the upstream repository (or any changes is happened in upstream repo), you can sync forked repository to the upstream repository to keep up to date.

After that, you will have your forked repository is in sync with the upstream repository.



**This branch is out-of-date**

Update branch to keep this branch up-to-date by syncing 1 commit from the upstream repository.

Learn more about syncing forks

Sync forked repository to upstream repository.

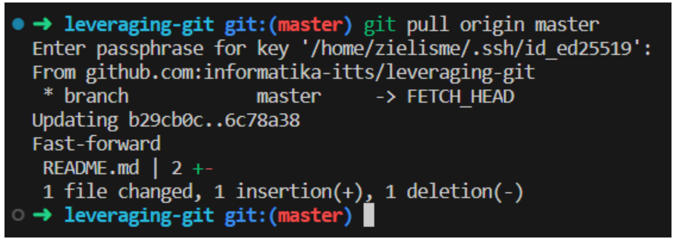
# Sync Local Repo with Git Pull

Lastly, you can **get the changes** from the remote repository to local by requesting Git to pull the changes from remote repository.

**`git pull origin master`**

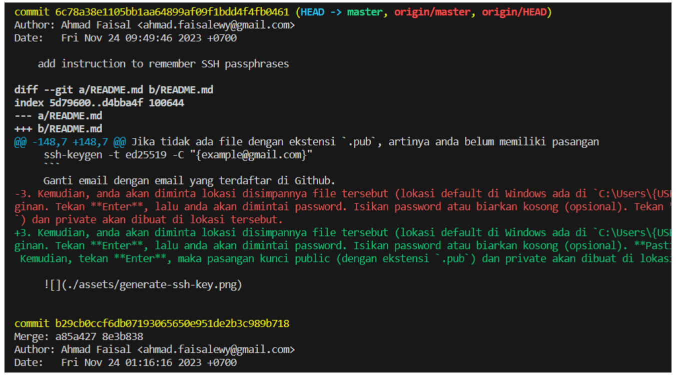
And, voila! You have your **local repository is in sync with both remotes**, your forked repo and upstream repo.

You can do this **workflow** to **contribute with your team mates**, or to **open source project**.



```
● ➜ leveraging-git git:(master) git pull origin master
  Enter passphrase for key '/home/zielisme/.ssh/id_ed25519':
  From github.com:informatika-itts/leveraging-git
   * branch            master     -> FETCH_HEAD
  Updating b29cb0c..6c78a38
  Fast-forward
   README.md | 2 +-
   1 file changed, 1 insertion(+), 1 deletion(-)
○ ➜ leveraging-git git:(master) ▌
```

Pull changes from remote to local



```
commit 6c78a38e1105bb1aa64899af09f1bdd4f4fb0461 (HEAD -> master, origin/master, origin/HEAD)
Author: Ahmad Faisal <ahmad.faisalewy@gmail.com>
Date:   Fri Nov 24 09:49:46 2023 +0700

    add instruction to remember SSH passphrases

diff --git a/README.md b/README.md
index 5d79600..d4bba4f 100644
--- a/README.md
+++ b/README.md
@@ -148,7 +148,7 @@ Jika tidak ada file dengan ekstensi `.pub`, artinya anda belum memiliki pasangan
     ssh-keygen -t ed25519 -C "{example@gmail.com}"
     ```

     Ganti email dengan email yang terdaftar di Github.
-3. Kemudian, anda akan diminta lokasi disimpannya file tersebut (lokasi default di Windows ada di `C:\Users\{US
ginan. Tekan **Enter**, lalu anda akan dimintai password. Isikan password atau biarkan kosong (opsional). Tekan *
`) dan private akan dibuat di lokasi tersebut.
+3. Kemudian, anda akan diminta lokasi disimpannya file tersebut (lokasi default di Windows ada di `C:\Users\{US
ginan. Tekan **Enter**, lalu anda akan dimintai password. Isikan password atau biarkan kosong (opsional). **Past
 Kemudian, tekan **Enter**, maka pasangan kunci public (dengan ekstensi `.pub`) dan private akan dibuat di lokas

     ![](./assets/generate-ssh-key.png)

commit b29cb0ccf6db07193065650e951de2b3c989b718
Merge: a85a427 8e3b838
Author: Ahmad Faisal <ahmad.faisalewy@gmail.com>
Date:   Fri Nov 24 01:16:16 2023 +0700
```

Show the differences by running **`git log --patch`**

Google Developer Student Clubs
```

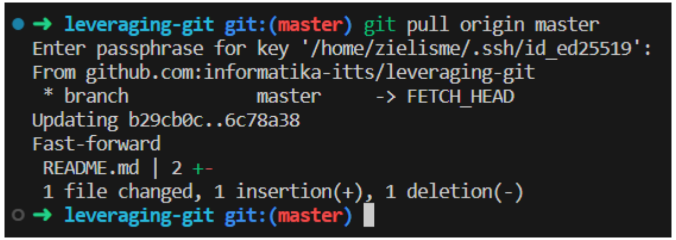
# Sync Local Repo with Git Pull

Lastly, you can **get the changes** from the remote repository to local by requesting Git to pull the changes from remote repository.

**`git pull origin master`**

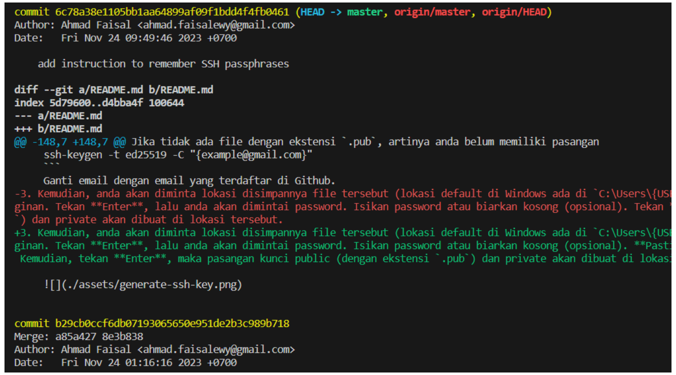
And, voila! You have your **local repository is in sync with both remotes**, your forked repo and upstream repo.

You can do this **workflow** to **contribute with your team mates**, or to **open source project**.



```
● ➜ leveraging-git git:(master) git pull origin master
  Enter passphrase for key '/home/zielisme/.ssh/id_ed25519':
  From github.com:informatika-itts/leveraging-git
   * branch            master     -> FETCH_HEAD
  Updating b29cb0c..6c78a38
  Fast-forward
   README.md | 2 +-
   1 file changed, 1 insertion(+), 1 deletion(-)
○ ➜ leveraging-git git:(master) ▌
```

Pull changes from remote to local



```
commit 6c78a38e1105bb1aa64899af09f1bdd4f4fb0461 (HEAD -> master, origin/master, origin/HEAD)
Author: Ahmad Faisal <ahmad.faisalewy@gmail.com>
Date:   Fri Nov 24 09:49:46 2023 +0700

    add instruction to remember SSH passphrases

diff --git a/README.md b/README.md
index 5d79600..d4bba4f 100644
--- a/README.md
+++ b/README.md
@@ -148,7 +148,7 @@ Jika tidak ada file dengan ekstensi `.pub`, artinya anda belum memiliki pasangan
     ssh-keygen -t ed25519 -C "{example@gmail.com}"
     ```

     Ganti email dengan email yang terdaftar di Github.
-3. Kemudian, anda akan diminta lokasi disimpannya file tersebut (lokasi default di Windows ada di `C:\Users\{US
ginan. Tekan **Enter**, lalu anda akan dimintai password. Isikan password atau biarkan kosong (opsional). Tekan 
`) dan private akan dibuat di lokasi tersebut.
+3. Kemudian, anda akan diminta lokasi disimpannya file tersebut (lokasi default di Windows ada di `C:\Users\{US
ginan. Tekan **Enter**, lalu anda akan dimintai password. Isikan password atau biarkan kosong (opsional). **Past
 Kemudian, tekan **Enter**, maka pasangan kunci public (dengan ekstensi `.pub`) dan private akan dibuat di lokas

     ![](./assets/generate-ssh-key.png)

commit b29cb0ccf6db07193065650e951de2b3c989b718
Merge: a85a427 8e3b838
Author: Ahmad Faisal <ahmad.faisalewy@gmail.com>
Date:   Fri Nov 24 01:16:16 2023 +0700
```

Show the differences by running **`git log --patch`**

Google Developer Student Clubs
```

## Referensi

| Sumber | URL |
|---|---|
| Git Cheat Sheet, by Red Hat Developer | https://developers.redhat.com/cheat-sheets/git-cheat-sheet |
| Pro Git: Everything You Need to Know about Git, by Scott Chacon and Ben Straub | https://git-scm.com/book/en/v2 |

Google Developer Student Clubs

*“You don’t have to be great to start, but you have to start to be great”*

Zig Ziglar

```
1*/
ild: Column(
crossAxisAlignment: CrossAxis
children: [
  /*2*/
  Conta
    pad
    chi
      '
      s

      )
    ),
  ),
  Text(
    'Ka
    sty
      c
    ),
  ),
]
```

## Please submit the feedback form

SCAN ME

Google Developer Student Clubs

## Thank you!

Any Questions?

Google Developer Student Clubs