**GIT & GITHUB**

[ alias 2](#_Toc101569627)

[ pwd 2](#_Toc101569628)

[ ls 2](#_Toc101569629)

[ cd 2](#_Toc101569630)

[ mkdir <nama\_folder> 2](#_Toc101569631)

[ git init 2](#_Toc101569632)

[ git add <file(s)> 2](#_Toc101569633)

[ git rm –cached <file> 2](#_Toc101569634)

[ git status 2](#_Toc101569635)

[ git commit -m “<pesan>” 2](#_Toc101569636)

[ git config 2](#_Toc101569637)

[ git log 2](#_Toc101569638)

[ git checkout 2](#_Toc101569639)

[ git branch 3](#_Toc101569640)

[ git merge <branch\_tujuan> 3](#_Toc101569641)

[ git help 3](#_Toc101569642)

[ git clone <link> 3](#_Toc101569643)

[ git remote –v 3](#_Toc101569644)

[ git push 3](#_Toc101569645)

[ git fetch 3](#_Toc101569646)

[ git pull 3](#_Toc101569647)

[ tahapan repo biasa 4](#_Toc101569648)

[ tahapan mengambil repo dari github (di local tidak, github ada) 4](#_Toc101569649)

[ tahapan menyimpan repo ke github (di local ada, github tidak) 4](#_Toc101569650)

[ tahapan menyimpan repo konflik ke github (di local ada, github ada) 4](#_Toc101569651)

# alias

membuat snippets : alias <nama>=”<kode>”

contoh :

* pembuatan : alias graph=”git log --all --decorate --oneline –graph”
* pemanggilan : $ graph

# pwd

cek lokasi direktori

# ls

cek ada folder apa saja

# cd

pindah direktori

# mkdir <nama\_folder>

membuat folder

# git init

Meng’inisialisasi repo git di komputer

# git add <file(s)>

untuk menambahkan file ke stagging area

# git rm –cached <file>

menghapus file dari stagging area

# git status

untuk mengetahui status repo

# git commit -m “<pesan>”

save perubahan

* git commit –m “<pesan>” : save perubahan
* git commit –am “<pesan>” : save perubahan tanpa git add

jika ada repo berstatus modified (tidak berlaku jika ada repo baru)

# git config

untuk konfigurasi

* git config --global user.name “<username>”
* git config --global user.email <email>

# git log

melihat riwayat commit

* git log -3 : menampilkan sesuai jumlah yg diinginkan (3)
* git log -- style.css : menampilkan sesuai nama folder yg diinginkan (style.css)
* git log --all --decorate --oneline --graph : tampilan seperti github

# git checkout

mengembalikan perubahan file

* git checkout <5kodeTerdepan> : mengembalikan file ke file yg dipilih
* git checkout <nama\_branch> : pindah branch
* git checkout <7kodeTerdepan> : pindah ke commit yg dituju

# git branch

mememeriksa sedang di branch mana.

* git branch <nama\_branch> : membuat branch
* git branch –d <nama\_branch> : menghapus branch
* git branch –D <nama\_branch> : menghapus branch yg blm di merge
* git branch --merged : untuk melihat branch mana yg sudah di merge

# git merge <branch\_tujuan>

menggabungkan branch

# git help

membuka bantuan

# git clone <link>

untuk menyalin repo github ke local

# git remote –v

periksa daftar remote

# git push

mengirim/menyimpan repo dari local ke github

# git fetch

memeriksa, repo yg ada di remote (github) sudah sejauh mana commitnya

# git pull

menyalin repo dari github ke local

# tahapan repo biasa

* git init
* git add
* git commit -m “pesan”
* git commit –am “pesan” :

jika ada repo berstatus modified (tidak berlaku jika ada repo baru)

# tahapan mengambil repo dari github (di local tidak, github ada)

* git clone <link> : untuk menyalin repo dari github ke local
* git push : mengirim/menyimpan repo dari local ke github

catatan : sebelum push, add dan commit terlebih dahulu

# tahapan menyimpan repo ke github (di local ada, github tidak)

* git remote add origin <link> : menambahkan remote
* git push –u origin master : push pertama kali
* git push : push kedua dan seterusnya

catatan : sebelum push, add dan commit terlebih dahulu

# tahapan menyimpan repo konflik ke github (di local ada, github ada)

contoh kasus : A menambah style.css, B juga menambah style.css

* git fetch
* git graph
* git pull : menyalin repo dari github ke local