Sve naredbe odnose se na **GIT BASH** cmd enviroment.

Za inicijaliziranje lokalnog repositority :

**git init** --inicijalizira repository u trenutnom folderu

**git status** --za pregled stanja foldera i datoteka unutar foldera (untracked, modified, new file...)

**git add .** --spremanje svih promjena u svim datotekama i priprema za commitanje (staging)

**git add file.txt** –spremanje promijena samo jedne datoteke (file.txt) (staging)

**git add \*.html** – spremanje promijena svih datoteka ekstenzije .html (staging)

**git commit**

Kod prvog pokrenja commit naredbe git će tražiti pristupne podatke (email, password)

Za unos pristupnih podataka unijeti:

**git config --global user.email** **"mail na githubu"**

(git config --global user.name "ime na githubu")

Nakon toga opet: *git commit*.

U dobijenom VIM screen ekranu može se unijeti komentar kod commitanja a da bi se ušlo u insert mode treba pritisnuti tipku **I**

Nakon unosa komentara izaći iz insert moda tipkom **ESC**

Nakon toga utipkati malim slovima

**:wq**

Naredba će prepisati i commitati fajlove. Ovime su sada sve promjene fajlova ažurirane u lokalnom repositoriju

Ako se utipka:

**git commit -m** **'Komentar komitanja'**

onda git ne ide u vim screen.

Ako se utipka :

**git commit –a –m** **'Komentar komitanja'**

onda se u jednoj naredbi izvršava staging (*add*) i commitanje. Ovo vrijedi samo za fajlove koji se već prate. Za sve nove fajlove koji se još ne prate (*untracked*) mora se posebno izvršiti naredba *add.*

Ako se želi poništiti spremljena ažuriranja jednostavno je dovoljno obrisati hidden **.git** folder u folderu u kojem se izvršila naredba **init**

**git log** –prikazati će povijest commitanja i izmijena zabilježenih u repositoriju

Za izbjegavanje pojedinih fajlova potrebno je kreirati fajl **.gitignore** i u njemu popisati listu fajlova koje ne želimo syncati. Na windowsima nije moguće kreirati fajl bez imena pa se tako može upotrijebiti naredba:

**touch .gitignore**

Vrijedi ista sintaksa (npr *\*.log* za isključivanje svih fajlova sa ekstenzijom .log)

.gitignore fajl se mora syncati.

Jedan folder odnosno repository u folderu može imati nekoliko **brancheva** (grana). Ovo je korisno za višekorisnički rad gdje svatko može raditi u svom brenchu.

Novi branch se kreira naredbom:

**git branch** **ImeMogBrancha**

Da bi se prebacili na novi branch potrebno je utipkati:

**git checkout** **ImeMogBrancha**

Ime odabranog Brancha će tada pisati u glavnoj naredbi koja prikazuje **ime stroja i putanju foldera.**

Jednom kada su promjene napravljene u odvojenom Branchu (add, commit) potrebno se vratiti u master Branch i mergati izmijene između naše Branche i master Branche. Znači u master branchi izvršiti naredbu:

**git merge ImeMogBrancha**

Kod merganja mogući su konflikti. Ako dvije grane rade izmijene na istom fajlu, jedna (master) commita izmijene a druga (MojaBrancha) napravi izmijene (bez add, staging) i commita i svoje kao rezultat dobićemo CONFLICT.

CONFLICT sadržaj istaknut je u navedenoj datoteci.

Kada se otvori datoteka HEAD dio je izmijena iz trenutne branshe (MojaBrancha) i prikazana je iznad izmijene koja je provedena u branshi master.

Odlučuje se što će i kojim redom ostati, uklanjaju se oznake HEAD i druge grane, crte, sprema se datoteka te se okida naredba:

**git commit –a –m** **'merge from master'**

Postoje i alati koji olakšavaju rješavanje konflikata kod merganja.

Neki od njih su: tortisemerge, **winmerge**.

Preporučljivo je instaliranje jednog od njih za pravo razvojno okruženje

TODO – OBRADITI NAREDBU STASH

Sve promjene moraju se ručno dohvatiti (retreive) ili gurnuti (push) u remote repository

**git remote** –daje popis postojećih remote repositorija

Bilo koji **remote repository** koji dodajemo odnosi se samo na trenutni **lokalni repositorij** u kojem radimo (lokalni repositorij iz trenutnog foldera)

Drugim rječima ako smo neki remote repositorij dodali u jedan folder ne možemo taj repositorij koristiti u drugom folderu.

Postupak stvaranja remote repositorija:

* Otići na github u repositorij koji smo kreirali za ovu potrebu i kopirati url (copy to clipboard)
* Izvršiti naredbu **git remote add origin UrlizClipboarda.**
* Naredba **git remote** u ovom folderu trebala bi sada prikazati remote repositorij (defaultni je origin)

Nakon kreiranja remote repositorija moguće su sljedeće naredbe za download ili upload podataka:

**git fetch origin** –dohvatiti će sve promijene sa servera od zadnjeg cloniranja u lokalni folder ali neće mergati sa podacima. Mergati se mora manualno

**git pull origin** –automatski dohvaća i merga promijene iz remote brancha u trenutnu branchu

**git push origin master** –govorimo gitu da pošalje (upload) naše izmijene u remote repositorij poznat nama kao origin te da ih commita u master branchu. U ovom trenutku upitati će nas se za credentiale (username i password sa githuba)

Za dodavanje dodatnog remote repositorija izvršiti naredbu:

**git remote add** **ImeNovogRepo** **Urlputanja**