sudo apt-get install git dedik git yükledik

git version dedik versionunu gördük – ha demek yüklenmiş

Önce kullanıcı adı ve email tanıtmak gerekiyor ,

git config - -global user.name “isim” user name atadı

git config - -global user.email “email@asdf.com” email atadı

Tüm git komutların başına git yazılır, - - global sistemdeki kullanıcı ?

git config - -global user.name - görüntüle

pwd - mevcut dizini gösterir

ls – listele , ls -a gizli dosyaları da göster

cd c:\www\ .. \ falan

clear , ctrl+L de aynı , ekranı temizle

dizine girdikten sonra , o dizinin git projesi olması için

git init

yazarız , oraya git dosyaları – gizli olarak - yükler

git deposuna ekleme :

git add . - tüm dosyaları depoya yükle demek tekbaşına yeterli değil , depoda değil henüz staging bölgesinde

git commit -m “bir mesaj , versiyon bilgi vb”

şimdi aktarıldı , git deponun durum bilgisine bakmak için

git log

logları verir

diyelim yeni dosya(lar) eklendi git add dosya1,dosya2 ... diye stage e attık

git commit –m “açıklama: ne ekledik naaptık”

yani önce add ile stage e , commit depoya

peki neden xtra bi alana ihtiyaç duyulmuş.

Git sys 3 bölümden oluşur

Çalışma dizini , geçiş bölgesi – staging , git deposu – repository

Anlık olarak yapılan değişiklikler çalışma dizininde oluşur. Stagin

Diff komutu : git status dedik , bir dosya modified dedi

git diff dersek , neresi modified satır satır gösterir

git diff , working dir ve staging arasındaki farkları verir. Gitt diff - - staged ise staging ile git dir (repository) arasındaki farkları verir

dosya silme ::

1-çalışma dizininden manual olarak silebiliriz 2-git komutu ile silebiliriz (dahakolay)

git checkout dosya.adi - silinmiş dosyayı geri getirir

silinmiş dosyayı stageden de silmek için add ya da rm kullanılabilir . şimdi rep. Commit edilmeye hazır !

elle silmez direkt

git rm dosya.adi yazarsak commit edilmeye hazır !

git rm –r klasor/ bir klasörü içindekilerle silmeye yarar , -r recursive demek

ceh!

Dosya rename :

1-manual 2-git ile

Manual olarak değiştirdikten sonra git add . + ceh

2. git mv a.php b.php , mv 2işlevli isimdeğiştirmek ve başka bir dir e taşımak sonra commit

Bir klasöre taşımak içinde git mv a.php diradi/ , sonra commit

Değişikliği geri alma :

Diyelim yanlış bir iş yaptık / bişi sildik dosyada , hatta dosyayı sildik komple ?

git checkout - - dosya.adi ile bu değişikliği geri alırız

Değişikliği Stage bölgesinden geri alma :

Hadi bide git add dedik , stage e aktardık . commit edersek geçmiş olsun !

git reset HEAD dosya.adi dedik ama yetmez

git checkout dosya.adi diyerek geri aldık

Versiyon Değiştirme

git log ile yapılan değişiklikleri gördük , bu değişikliklerden birine geri dönmek için , istenilen değişikliği - versiyon – hash kodunu kopyaladık , ilk 7 hane de yeter

sonra git checkout Ha$hk0d3 -- . dedik , sonundaki nokta önemli , eğer belli dosya(lar) istenirse sadece onların adı yazılır. Bu şekilde projeyi oradaki versiyonuna döndü ve CEH , en son da commit

GITHUB ::

git ve github aynı şey değildir , git bir versiyon ctrl sys iken github git yazılımı ile entegre olmuş bir depolama alanıdır. Git projelerinizi uzak bilgisayarda saklayıp yönetebileceğiniz bir yerdir , sadece github da yoktur ,gitlab bitbucket ...

önce github da bi rep yarattık ve urlsini kopyaladık : <https://github.com/ulaskorpe/git-h.git> diyelim

sonra ; git remote add depoAdi <https://github.com/ulaskorpe/git-h.git> dedik

localde dep oluşturduktan sonra bunu push etçez

git push jq1 –u master dedik , master olarak up etmek için bizden user/pw isticek girdik ve yükledik

yemedi mi ?? -f , forced ile zorla

dosyalarda bir değişiklik yapıp commit edildikten sonra, yine git push –u depo adı dedik , komple güncelledi

Gitignore ::

Projenizde göstermek istemediğiniz dosyalar varsa bu dosyaların ismini gitignore isimli bir dosyaya yazıyorsunuz , misal vt dosyasını saklamak istiyorsunuz.

cat >> .gitignore diye gitignore dosyamızı oluşturduk . dosya açık geldi

içine saklamak istediğimiz dosyaları yazdık ctrl+c ile save edip çıktık

git status dediğimizde veritabani.txt yani gizlediğimiz dosya gözükmüyor ancak .gitignore gözüküyor

not : saklamak istenilen dosyalar oluştuktan sonra add ya da commit edilmemelidir yoksa gitignore un bir etkisi olmaz

sonra .gitignore dosyasını git deposuna gönderiyoruz

bir dir saklanmak isteniyorsa belgeler/\* satırı eklenmelidir. Belgeler klasörü ignored

başına ! koyarak dir deki bir dosyayı hariç tutabiliriz

!belgeler/dosya1.html

BRANCHES ::

Branch – dallar – branch git projesinin bir kopyasıdır , master tüm projeyi kapsayan anadaldır. Bir projede birden fazla kişinin çalışmasını sağlar , her kişi için bir dal oluşturulur . sonra master dalı ile birleşir , herkesin ayrı geliştirdiği moduller tek bir master projesi olur.

Master proje ile dallar merge edilir

Sonra dizinde git pull komutu ile master dizinler eşitlenir

Git branch diye dallar listelenir

Git branch - - all uzak depodaki dallar da görüntülenir

Git branch ulaskorpe diye yeni branch oluşur

λ git checkout ulaskorpe

Switched to branch 'ulaskorpe' şeklinde branch değişitrilir

Git merge ulaskorpe master diye 2 dal birleştirilir

Watch : her bir commit işlemi yapıldığında takip ettiğin projede haber verilir

Star : beğendim anlamına gelir

Fork : projenin bi kopyasını hesabınıza almış olursunuz

Issues : projeniz ile ilgili bir sorun oldğunda bunu yazarsınız , projedeki diğerleri buna yardım ederler …