GİT KULLANIM NOTLARIM

Pwd : Terminal dizini

ls : Dosya ve dizinleri listeler

cd : Dizin geçişleri veya değiştirmek için

git init : Projede git versiyonlama sistemini başlatmak için kullanılır.

git config --system : Tüm kullanıcı ve projeler içi geçerli olan ayarlardır.

git config –global : Sadece sizin kullanıcınız için geçerli olan ayarlardır.

.git/config : Dosyasında ise proje bazındaki git ayarları yer alır.

git config –global user.name “fahrikaraca” : Projedeki kullanıcı ismi

git config –global user.email “fahrikaraca0@gmail.com” : Projedeki kullanici email adresi

git config --global core.editor notepad++ : Editör eklemek için PATH yolu verilmelidir.

git config –global –l : Tüm ayarları listelemek için

git add . veya git add dosya\_adi : Staging area ya ekleme işlemi yapar.

git add –a –m “ilk commit “ : Bu komut Staging area işlemine gerek kalmadan hem ekleme hemde commit işlemi yapar.

git –commit –m “ilk komit” : Commit işlemini yaparak repositorye aktarır.

git log : Repositorydeki commitleri görüntüler.

git status : Working area ile staging area arasindaki farkları gösterir.

git diff dosya1 dosya2 : Dosyalar arasindaki farklılıkları gösterir.

git diff dosyaadi : Bu komut dosyadaki değişiklikleri gösterir.

git diff –staged : staging area ve repository arasindaki değişiklikleri gösterir.

git rm dosya\_adi : Silme işlemini yapar.

git rm –r klasör\_adi/ : Belirtilen klasörü siler.

git mv dosya\_adi dosya\_adi2 : Dosya adini değiştirir

git checkout – dosya\_adi : Staging area bölgesindeki dosyayı geri getirir.

git reset head dosya\_adi : Repositoryde silinmiş öğeyi staging areaya taşıma işlemi yapar.

git checkout branch\_adi : Branch değiştirme yapar.

git branch yeni\_isim : Yeni bir branch oluşturur.

git branch : Dosyada bulunan branchleri gösterir.

git config –global alias.hist “og –all – graph –decorate –oneline” : Verilen komuta kısaltma sağlar.

git config –global merge.tool p4merge : Eklentiyi kurul merge işlemi sırasında daha faydalı kullanmak için.(PATH BİLGİSİ VERİLMELİ)

git config –global diff.tool p4merge : Yapılan değişiklikeri p4merge eklentisi ile açılması için.

git config –global –e 🡪 Default editörle tüm ayarları açar.

git diff head : Working ile repository arasındaki değişiklik

git diff –staged head : Staging ile Repository arasindaki değişiklik yapar.

git diff –DosyaAdi : Tek bir dosya karşılaştırması yapmak için.

git diff komit\_id komit\_id2 : İki commit arasındaki farklar için .

git branch yeniBranchAdi : Yeni branch oluşturma.

git checkout branchAdi : Branch değiştirmek için .

git branch –a : Tüm branchler için .

git branch –d branchAdi : Bir branchi silmek için .

git checkout –b yeniBranch : Yeni bir branch oluşturup ona geçiş yapmak için.

git merge Branch\_adi : merg işlemi için gereklidir.

git merge branchAdi –-no-ff : Dallanmaları graph üzerindeki göstermek için.

git mergetool : Conflicts çözümü için yapılır.

git log –graph –oneline – decorate : Commitlerde yapılan değişiklikler.

git merge branch\_adi : Master ile branch arasindaki birleştirme işlemidir.

git rebase master : Master ile oluşturulan branch birleştirme işlemi yapılır.

git merge Branch\_adi : Yapılmasi zorunlu Rebase den sonra.

Rebase Conflict oluşumu

git rebase –abort : Yapılan merge işlemi iptal edilir.

git mergtool : Acılmasi gerekiyor.

Git rebase –continue : Conflict tamamlandıktan sonra devam işlemi.

git merge branch\_adi : Burda merge işlemi tamamlanıyor.

DİKKAT !!!

Conflicts oluşmaması durumunda :

1 - > Branch birleştirme işlemi için master a geçilir.

2 -> git merge branch \_Adi yazılır. İşlem yapılır.

3- > Commit ekranı gelırse mesaj yazılır.

Conflict Olusursa !!!

1 - > Branch birleştirme işlemi için master a geçilir.

2 -> git merge branch \_Adi yazılır. İşlem yapılır.

3- > Conflicts Uyarı verir.

4- > git status den bakılabilir.

5 -> git mergetool açılır.

6- > Değişiklik yapılır.

7 - > git status den bakılabilir.

8 -> git commit –m “Conflicts çözüldü “ yapılmasi gerekiyor.

Conflict işlemi sonrasinda commit işlemi yapılmalıdır.

REBASE İŞLEMİ

1 -> git rebase master denilir.

2 -> git checkout master yapılarak geçilir.

3 - > git merge branç\_Adi yazılır.

CONFLİCTS REBASE İŞLEMİ

1 -> git rebase master yapılır conflict oluştuğunu söyler.

2 -> git mergetool ile değişikliğe gidilir.

3 - > git status ile değişikliğe bakılabilir.

4- > git rebase --continue ile devam edilir.

5 -> git checkout master ile branch değiştirilir.

6 -> git commit ile işlem yapılır.

git remote –v : Uzak Repository işlemi .

git clone http : // github link : Projenin kopyasini getirir.

Git push origin master : Origin burda daha önce yapılan bir link varsa isim olarak davranır.

Git remote add origin http://github link