Github'ta bir repo oluşturdukran sonra **aşağı komutları**, github tavsiye eder. Örnek olarak GithubTest şeklinde bir repos oluşturduk.

echo "# GithubTest" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin <https://github.com/emreakturk1125/GithubTest.git> => origin aslında isimlendirme olarak düşünebilirsiniz. Sürekli bu url yazmamak için, farklı bir tanımda olur

git push -u origin main => main branch'ini origine push et demektir.

git remote add origin <https://github.com/emreakturk1125/GithubTest.git>

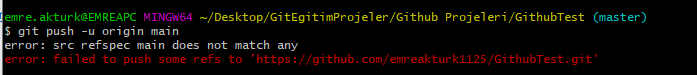
git branch -M main

git push -u origin main

Sonrasında localimizdeki mevcut projemizi Github'a atmak için yapacağımız işlemler sırası ile

Git Bash kullanılarak, masaüstünde ya da herhangibir yerde ;

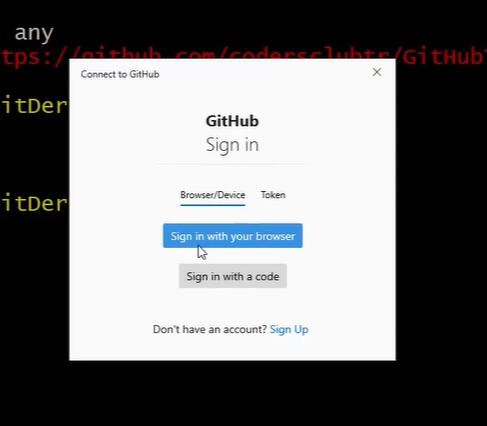
* **mkdir GithubTest =>** klasör oluşturduk
* **git init =>** klasör içine girerek, bu komutu ile git reposu haline getirdik
* **touch test.txt** => text dosyasını oluşturduk
* **git commit -m "first commit"** => ilk commitimizi attık
* **git remote add origin** [**https://github.com/emreakturk1125/GithubTest.git**](https://github.com/emreakturk1125/GithubTest.git) => komutu ile Github'ta oluşturduğumuz repo url'ini origin olarak tanımladık. (1 SEFERLİK ÇALIŞTIRILACAK)
* **git push -u origin main** => komutu çalıştırıldığında, aşağıdaki gibi bir hata alınması durumunda



* **git branch** komutunu çalıştır ve mevcut branch ne ise ona göre



* **git push -u origin master =>** olarak çalıştırılması gerekiyor. Bu şekilde login olmamız istenebilir. Github bilgilerimizle giriş yapabiliriz.



* Login olduktan sonra, bu şekilde localimizeki mevcut projemizi Github'a atmış olduk.
* Farklı bir branch ide origin olarak tanımladığımız [**https://github.com/emreakturk1125/GithubTest.git**](https://github.com/emreakturk1125/GithubTest.git) remote ortamına atabiliriz.

Örnek;

* Master branchindeyken **git branch feat =>** komutu ile farklı bir branch oluştur
* **touch ikinci.txt** => komutu ile yeni bir txt dosyası oluştur
* Değişikliklerden sonra **git commit -m "feat branchindeki ilk commit"**  => commit at
* **git push -u origin feat =>** bu komut ile de ikini branchimiz yani feat branchini origine atmıştık olduk. Böylece iki farklı branci github'a taşımış olduk

1. **git push origin main**:
   * **Bu komut, yerel main dalını uzak origin depoya iter.**
   * **Herhangi bir takip ayarı yapmaz, bu yüzden her git push komutunda dal ismi belirtilmelidir.**
2. **git push -u origin main**:
   * **Bu komut da yerel main dalını uzak origin depoya iter.**
   * **Bu, gelecekte git push komutunu tek başına çalıştırdığınızda, dal ismi belirtmeden doğrudan main dalını origin deposuna itebileceğiniz anlamına gelir.**

Özetle, **git push -u origin main** komutunu bir kez çalıştırdıktan sonra, sadece git push yazarak aynı işlemi yapabilirsiniz. Bu, takip dalları kurmanızı sağlar ve gelecekteki push işlemlerini basitleştirir.

* "**git fetch origin master**" => değişiklikleri al demek, local branchimize değişiklikleri getirir ama dosyalarda bir değişiklik yapmaz. Değişiklik olup olmadığını kontrol etmek içindir
* "**git pull**" => tüm değişiklikleri al ve benim branch'ime ekle demek

* "**git branch pilot**" master üzerinden yeni branch oluşturur
* "**git branch**" branch leri listeler
* "**git remote**" remote taki branchleri listeler
* "**git branch -r**" remote taki yani origindeki branchleri de gösterir
* "**git switch pilot**" ya da "**git checkout pilot**" pilot branch ine geçer
* "**git switch**" ile localdeki branchlerde geçiş yaparken,
* "**git checkout**" ile origindeki branchlerde geçiş yapabiliriz. Fakat "**HEAD detach**" şeklinde master branch ile aynı seviyede olmayan bir branch oluşturulur
* "**git branch -d branchName**" braanch adına göre o branch I siler