Github Kullanımı

Github İçeriği

- Sürüm Kontrolü Nedir? (Version Control)
- Git Nedir?
- Hub nedir?
- GitHub Nedir ve Neden Bu Kadar Popüler?
- GitHub Nasıl Kullanılır?
- GitHub Deposu Oluşturun
- GitHub Dalları/Şubeleri Oluşturun

Sürüm Kontrolü Nedir? (Version Control)

- Sürüm kontrolü, bir dosyada veya bir dizi dosyada yapılan değişiklikleri izlemeye ve yönetmeye yardımcı olan bir sistemdir. Ağırlıklı olarak yazılım mühendisleri tarafından kaynak kodunda yapılan değişiklikleri izlemek için kullanılan sürüm kontrol sistemi, tüm değişiklikleri analiz etmelerini ve bir hata yapıldığında herhangi bir sonuç olmadan geri almalarını sağlar.

- Başka bir deyişle, sürüm kontrolü, geliştiricilerin aynı anda projeler üzerinde çalışmasına olanak tanır. Meslektaşlarının çalışmalarını ihlal etmeden veya geciktirmeden ihtiyaç duydukları kadar değişiklik yapmalarını sağlar.
- Kaynak kodunda söz konusu değişiklikler, dağıtıldıklarında projeyi mahvederse, GitHub bunları birkaç tıklamayla tersine çevirmeyi kolaylaştırır ve projenin önceki sürümü geri getirilir.

- Özetlemek gerekirse, sürüm kontrolü riskleri ve çok fazla hata yapma korkusunu ortadan kaldırır. Bunun yerine, çok fazla endişe duymadan işbirliği yapma ve geliştirme özgürlüğü sağlar.

Git Nedir?

- Git, 2005 yılında başlatılan ve piyasadaki en popüler VCS'lerden biri haline gelen açık kaynaklı bir projedir. Geliştiricilerin %87'sinden fazlası projeleri için Git'i kullanır.

- Dağıtılmış bir sürüm kontrol sistemidir. Yani, erişim izni verilen ekipteki herhangi bir geliştirici, Git komut satırı araçlarını (Git command line tools) kullanarak kaynak kodunu ve değişiklik geçmişini yönetebilir.

- Git, merkezi sürüm kontrol sistemlerinden farklı olarak özellik dalları (feature branches) sunar. Bu, ekipteki her yazılım mühendisinin kodda değişiklik yapmak için yalıtılmış bir yerel depo sağlayan bir özellik dalını ayırabileceği anlamına gelir.
- Özellik dalları, orijinal proje kodunun bulunduğu ana dalı etkilemez. Değişiklikler yapıldıktan ve güncellenen kod hazır olduğunda, özellik dalı ana dal ile birleştirilebilir, bu şekilde projede yapılan değişiklikler etkin hale gelir.

Hub nedir?

- Eğer Git, GitHub'ın kalbiyse, Hub da onun ruhudur. GitHub'daki hub ise komut satırını, Git gibi, geliştiriciler için en büyük sosyal ağa çevirendir.

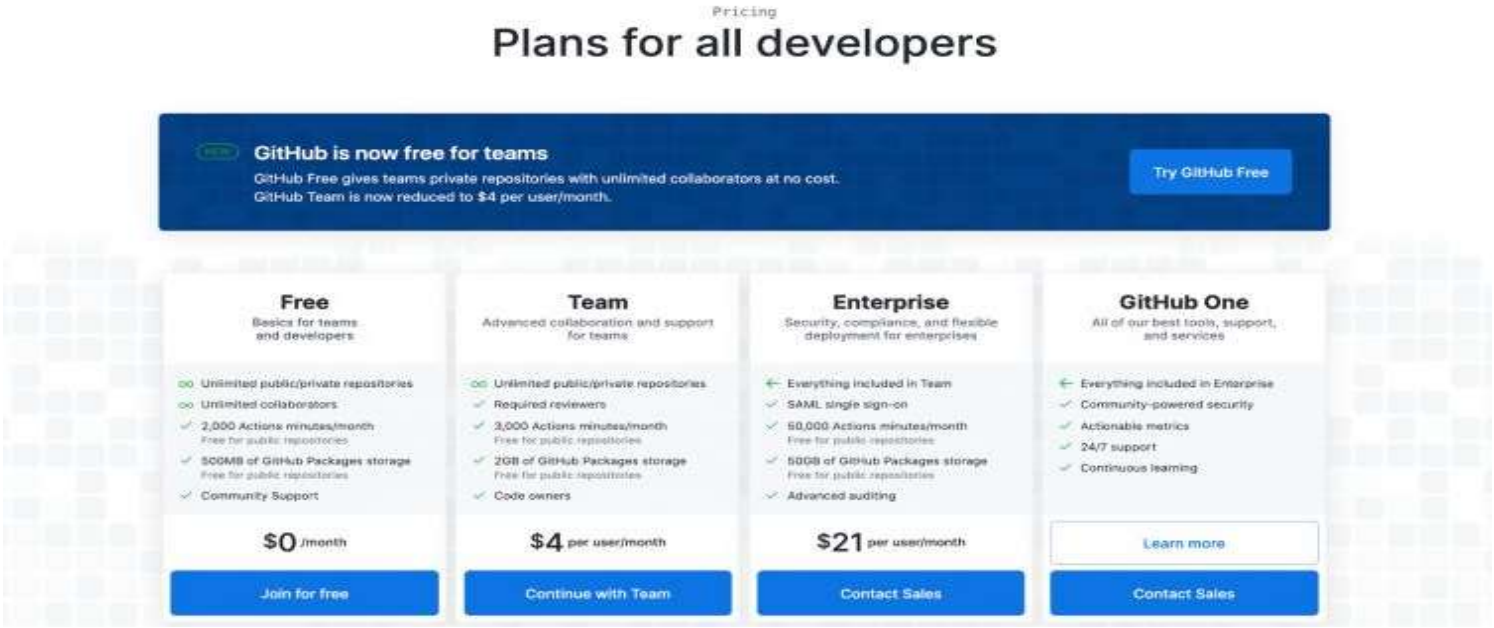
- Belirli bir projeye katkı sağlamanın dışında GitHub kullanıcılarına kendileri gibi hemfikir insanlarla sosyalleşme olanağı sağlar. İnsanları takip edebilir ve ne yaptıklarını veya kimle bağlantı kurduklarını izleyebilirsiniz.

GitHub Nedir ve Neden Bu Kadar Popüler?

- GitHub, çoğunluğu açık kaynaklı projeler olan 100 milyondan fazla fazla depoya ev sahipliği yapıyor.
- Bu istatistik, GitHub'ın en popüler Git GUI istemcileri arasında olduğunu ve çeşitli profesyoneller ve Hostinger gibi büyük işletmeler tarafından kullanıldığını gösteriyor.

GitHub Nasıl Kullanılır?

- GitHub’ı ekibinizle ücretsiz olarak deneyebilirsiniz. Sınırsız depo ve ortak çalışan ancak yalnızca 500 MB depolama alanı içeren temel bir plan mevcuttur.
- GitHub’ın birçok özelliğine daha kapsamlı bir bakış için ücretli planlarından birini seçebilirsiniz.

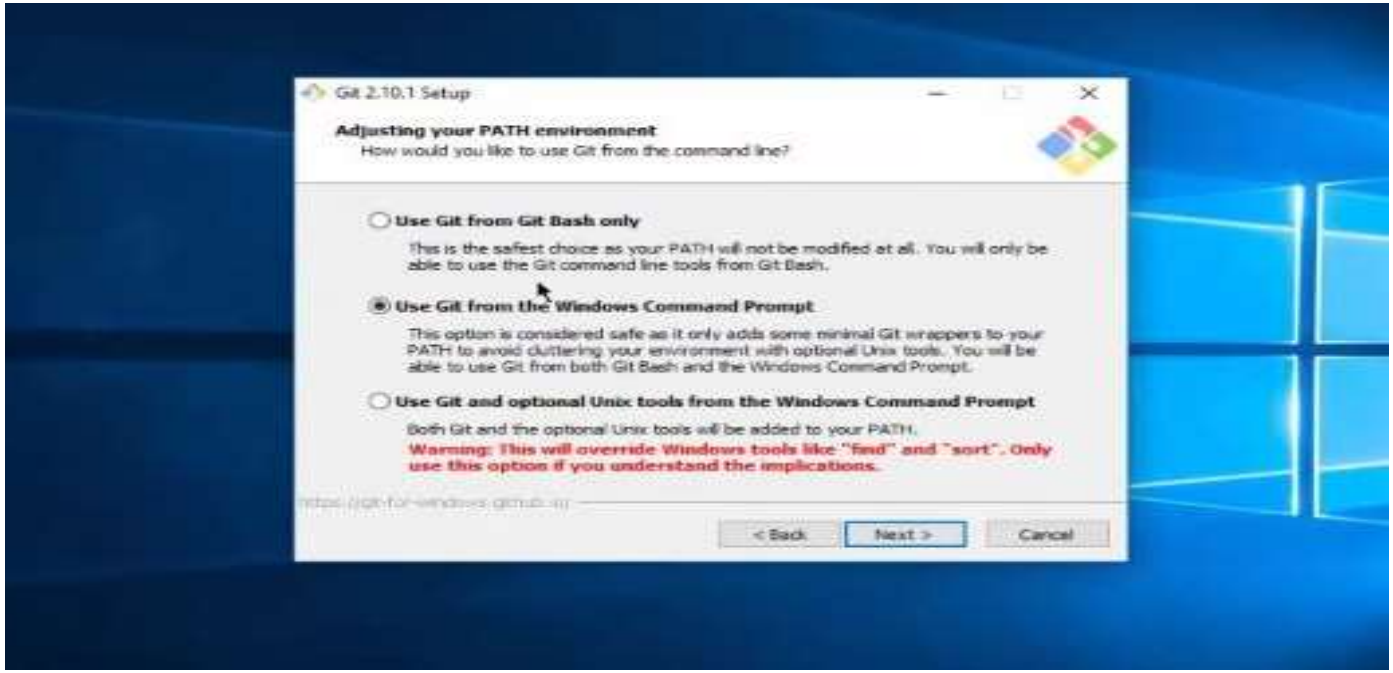


- Bir plan seçtikten, gerekli bilgileri doldurduktan ve kayıt işlemini bitirdikten sonra GitHub'ın neler sunabileceğini keşfetmeye başlayabilirsiniz. Git'ten farklı olarak GitHub, komut satırının kodlanmasını veya kullanılmasını gerektirmez.

Git kurulumu

- Öncelikle Git programını bilgisayarımıza yüklemeliyiz. Bu linki kullanarak direkt indirme sayfasına gidebilirsiniz: https://git-scm.com/downloads
- Linux kullanıyorsanız Linux için, Windows kullanıyorsanız windows için uygulamayı indiriniz.

- Git programını bilgisayarınıza indirdikten sonra kurulum aşamasında diğer programlar içerisinde kullanımına izin ver seçeneğine tıklayarak yükleme işlemini bitirin. Görseldeki en alttaki seçenek.



Git yapılandırması

- Öncelikle indirmiş olduğumuz Git Bash terminalini çalıştıralım.

- Açtığımız terminalde Git hesabımızı Git Bash programı ile bağlamamız gerekiyor.
- Bunun için bir önceki sayfadaki görselde yazan komutları sırasıyla girmemiz gerekir. Bu kod kullanıcı adımızı yani user name i kayıt etmek için kullanılır.
- Her Kod arasına boşluk koymayı unutmayalım.
- Daha sonra aynı user name için yaptığımız işlemleri birde user email için yapalım.


baris@ubuntu: ~
baris@ubuntu:~$ git config --global user.name "Barış Aslan"
baris@ubuntu:~$ git config --global user.email "aslannbaris@gmail.com"

- Girdiğimiz değerlerin geçerli olup olmadığını kontrol etmek için aşağıdaki komutu girip güncel durumuma bakabiliriz.

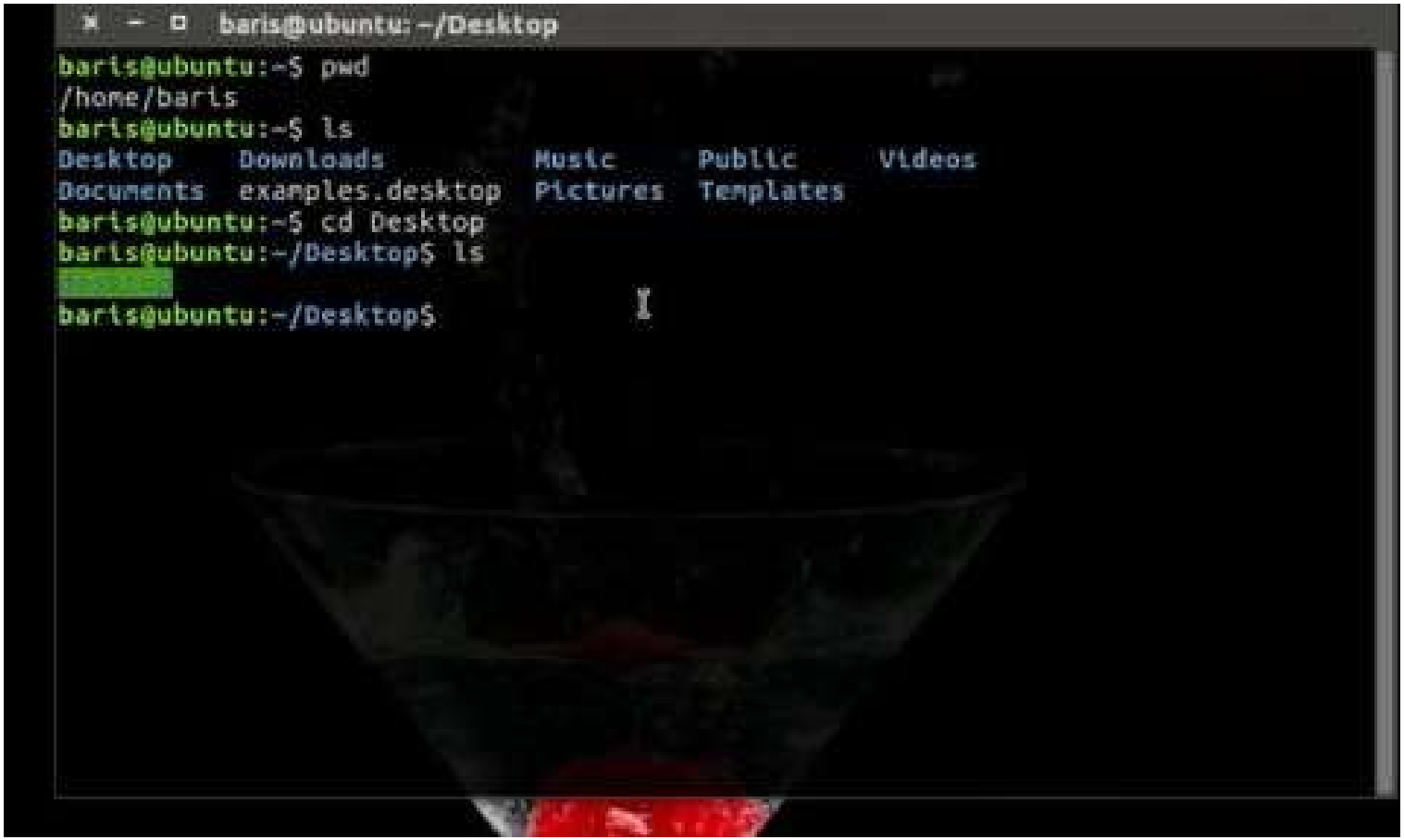
Git bash içerisinde kullanılabilen komutlar

- Pwd: pwd komutu mevcut terminal üzerinde bulunduğumuz tam adresi bize iletir.

- Ls: ls komutu bulunduğumuz dosya içerisindeki dosyaları listelememizi sağlar.

- Cd: cd komutu change directory ‘nin kısaltmasıdır. Yön değiştirme anlamına gelir. Mevcut bulunduğumuz dosyadan istediğimiz dosyaya geçmemizi sağlar.

Komutların Terminal Görseli üzerindeki Gösterimi



- cd projeismi komutunu yazarsak istediğimiz dosyanın içerisine girebiliriz.
- Girdiğimiz herhangi bir dosyanın dışına çıkmak istiyorsak ls komutuyla tekrar dışarı çıkabiliriz.

- Terminali temizlemek istiyorsak **clear** komutunu yazarak terminal ekranınındaki yazılmış geçmiş komutları silebiliriz.
- Aynı şekilde **control + L** kısayolu ile bu işlemi yapabiliriz.

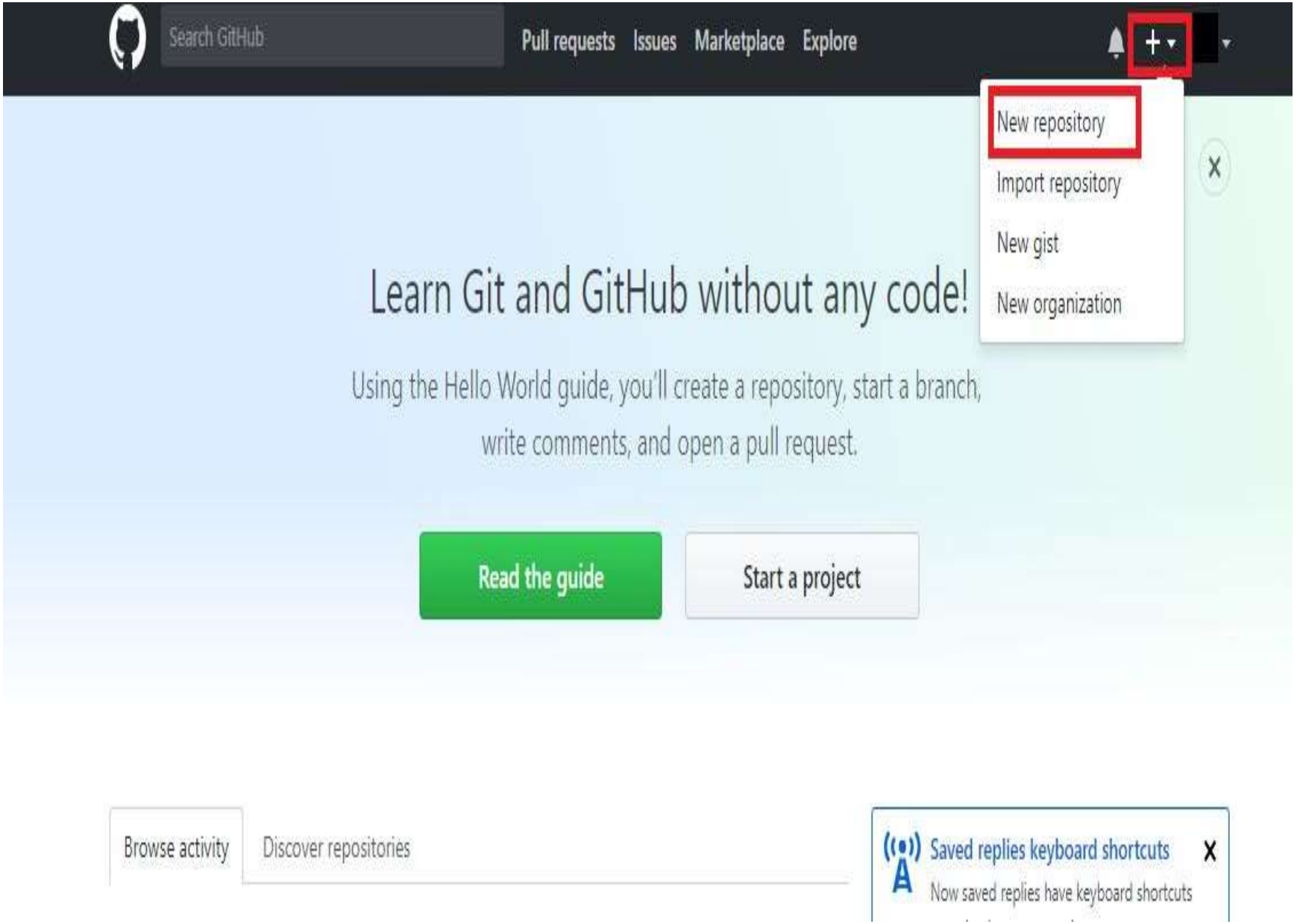
Dosyaları git dosyası haline getirme

- Oluşturduğumuz bir dosyayı github adresine yüklemek istiyorsak bu dosyanın içine git yüklemelerini yapmamız gerekir.
- Öncelikle masaüstünde boş bir dosya oluşturun.
- Oluşturduğunuz bu dosyanın ismini Git Bash programında **cd projeismi** olarak yazın.
- Proje dosyanızın içine girdikten sonra **git init** komutunu yazın. Bu sayede dosyanız artık bir git dosyası haline gelecektir.

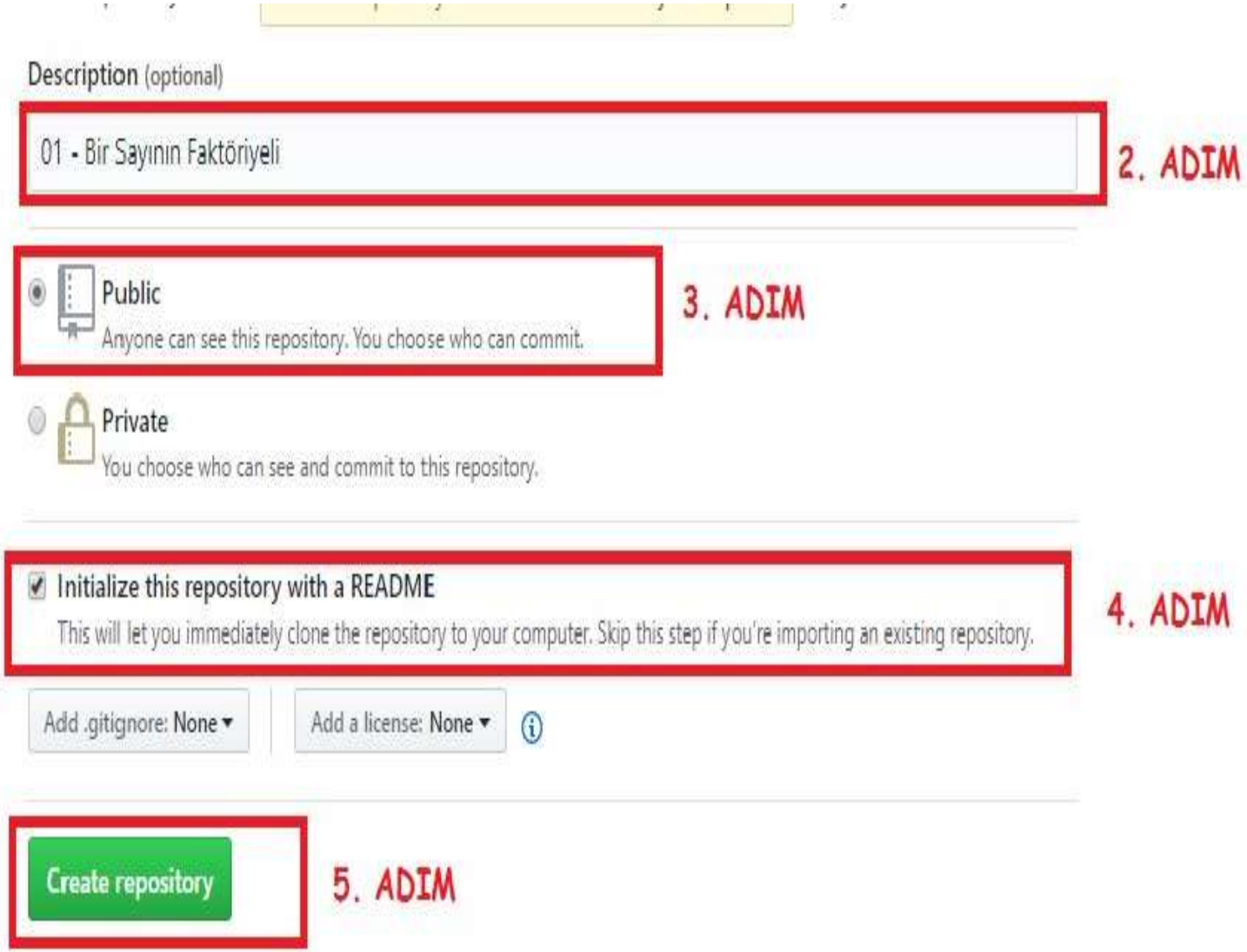
Projeyi Github’a yüklemek

- GitHub’ı kullanabilmek için önce kayıt olmamız gerekiyor. Kayıt için sitemiz:
- https://github.com/

- Kayıt olduktan sonra aşağıdaki sayfa karşımıza çıkacak.


Search GitHub
Pull requests
Issues
Marketplace
Explore
New repository
Import repository
New gist
New organization
Learn Git and GitHub without any code!
Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.
Read the guide
Start a project
Browse activity
Discover repositories
Saved replies keyboard shortcuts
Now saved replies have keyboard shortcuts

Buradan “New Repository” seçeneğine tıklıyoruz.


Description (optional)
01 - Bir Sayının Faktöriyeli
2. ADIM
Public
Anyone can see this repository. You choose who can commit.
3. ADIM
Private
You choose who can see and commit to this repository.
Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.
4. ADIM
Add .gitignore: None
Add a license: None
Create repository
5. ADIM

- **Repository Name”** kısmına projemize vermek istediğimiz ismi yazıyoruz.
- **“Description”** bölümüne proje açıklamamızı yazıyoruz.
- **“Public”** kısmını seçiyoruz, böylece yüklediğimiz projemize herkes ulaşabilecek.
- **“Readme”** ekledikten sonra **“Create Repository”** kısmına tıklıyoruz.

- Bu işlemleri tamamladıktan sonra git Repository ‘mizi oluşturmuş oluyoruz.

- Oluşturduğumuz bu repository’ ye aşağıdaki komutları sırasıyla Git Bash Terminalinde girmeliyiz.
- Komutlar sırasıyla:
- 1. git init
- 2. git add . (burada nokta kullanmamızın sebebi oluşturduğumuz projedeki bütün dosyaları repository’mize aktarmaktır. Lütfen aralarında boşluk olmasına dikkat edin.)
- 3. git commit –m “first commit”

- 4. git branch –M main
- 5. git remote add origin https://github.com/furkankcoder/aa.git (buradaki link örnektir. Repository sayfamızdaki URL bağlantısıdır.)
- 6. git push –u origin main

Tebrikler ilk projenizi repository olarak github’a yüklediniz.

1. GitHub Deposu Oluşturun

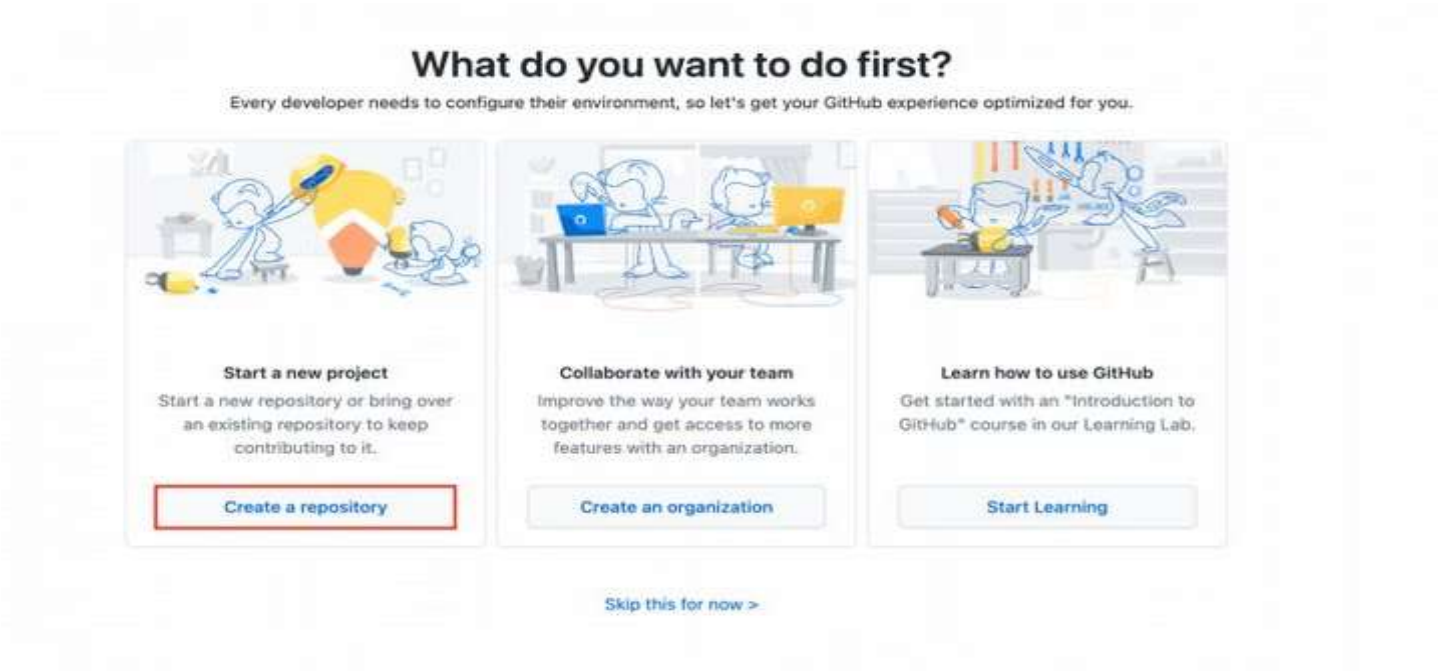
- Bir depo veya repo, projenizin merkezi hub'ı olacaktır. Tek bir dosya veya kod, resim, metin veya başka herhangi bir şey içeren bir dosya koleksiyonu olabilir.

Repository Nedir? (Depo)

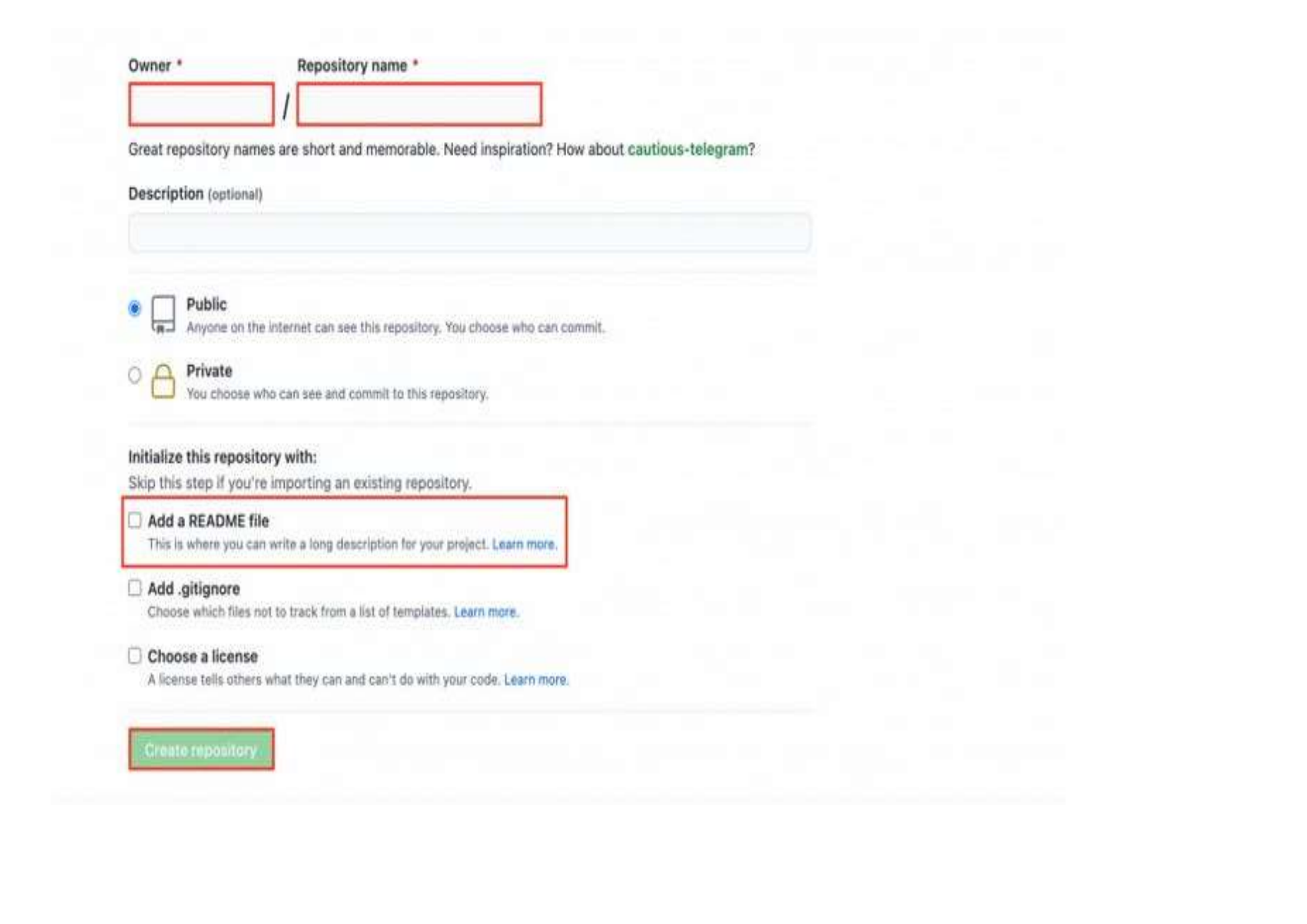
- Repository veya repo, projelerinizin dosyalarının depolandığı bir dizindir. GitHub'ın alanında veya bilgisayarınızdaki yerel bir depoda bulunabilir. Dosyalar, fotoğraflar, sesler veya projenize alakalı her şeyi repository'inizde depolayabilirsiniz.

Repository Oluşturma

- İşlemi başlatmak için şu adımları izleyin:
- Yeni bir proje başlatmak için **Create a repository** ‘a tıklayın.



- Owner (Sahip) bölümü hesap adınız olacak. Bir Repository Name (Depo Adı) oluşturun.
- Açık kaynak yapmak için Public olarak ayarlanıp ayarlanmadığını kontrol edin ve ardından Add a README file dosyası ekle kutusunu işaretleyin.
- Son olarak, Create repository‘ye tıklayın.


Owner *
Repository name *
/
Great repository names are short and memorable. Need inspiration? How about cautious-telegram?
Description (optional)
Public
Anyone on the internet can see this repository. You choose who can commit.
Private
You choose who can see and commit to this repository.
Initialize this repository with:
Skip this step if you're importing an existing repository.
Add a README file
This is where you can write a long description for your project. Learn more.
Add .gitignore
Choose which files not to track from a list of templates. Learn more.
Choose a license
A license tells others what they can and can't do with your code. Learn more.
Create repository

NOT !!!

- Deponuzu açık kaynak (Public) olarak ayarlamanız gerekmediğini unutmayın. Kimin göreceğini ve değişiklik yapacağını yönetmek için **Private** (Özel) olarak ayarlayabilirsiniz.

2. GitHub Dalları/Şubeleri Oluşturun

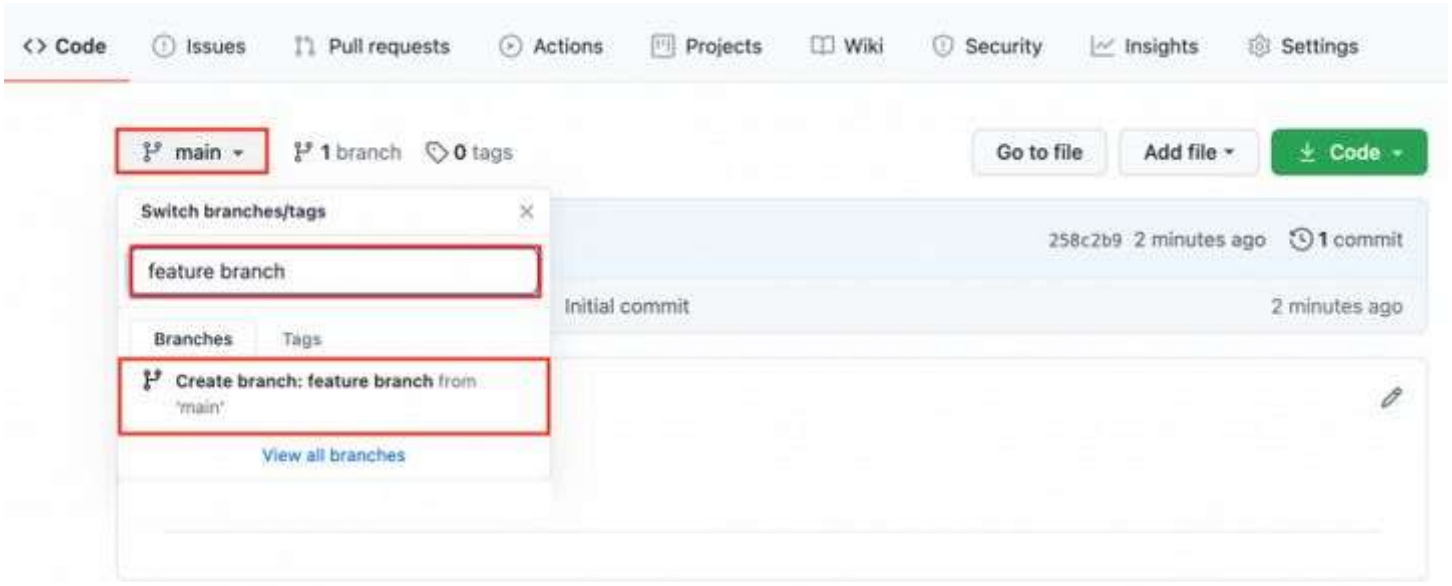
- Dallar oluşturarak bir havuzun farklı versiyonlarını oluşturursunuz. Bir geliştirici, özellik dalında proje değişiklikleri yaparak, entegre edildiğinde ana projeyi nasıl etkileyeceğini görebilir.

Branch nedir? (Dal)

- Branch deponuzun bir kopyasıdır. Branch'i diğerlerinden ayrı bir geliştirme yapmak istediğiniz zaman kullanabilirsiniz.
- Branch üzerinde çalışmak merkezi repository'yi veya DİĞER branchleri etkilemez. İşiniz bittiğinde, pull request'i kullanarak branch'inizi öbür branch'lerle ve merkezi repo ile birleştirebilirsiniz.

Bir özellik dalı şu şekilde oluşturabilirsiniz:

- Yeni deponuza gidin. **Main** butonuna basın ve yeni özellik dalınızın adını girin.**Create branch**‘a (Dal oluştur) tıklayın.



Git Branch Nasıl Kullanılır?

- Başlamadan önce sunucunuza SSH ile erişmeyi unutmayın.
- Git branchlerinin kullanımları ilk başta Git branch komutlarından da fark edebileceğiniz gibi basittir.
- Ancak tıpkı her şeyde olduğu gibi ne kadar branch'iniz varsa onları yönetmek o kadar zor olabilir.

git branch

- Herhangi bir Git projesinde aşağıdaki komut satırını girerek bütün branchleri görüntüleyebilirsiniz.
- **git branch [yeni_branch]**
- Eğer bir branch oluşturulmazsa terminalde bir çıktı olmayacaktır. Bir branch oluşturmak oldukça basittir.

git checkout [yeni_branch]

- Daha sonra ise yeni oluşturulmuş branch'e geçmeniz gerekiyor. Bunu yapmak içinse bu komutu kullanın.
- **Switched to branch 'test'**
- Bu çıktı yeni branch'e geçtiğiniz hakkında bizi bilgilendirecektir. Biz, bu örnekte branch'i test olarak adlandırdık, böylece aldığımız çıktı böyle oldu.

git branch

- Şimdiyse, bu yeni geliştirme branch'inde ana branch'te hiçbir şey değiştirmeden istediğimiz kadar kod düzenlemesi yaratabiliriz. Görebileceğiniz gibi yeni kod eklemeleri için programı organize düzenli tutar.
- Eğer branchleri listelemek için olan kodu çalıştırırsanız yeni bir branch eklendiğini ve sizin de içinde olduğunuzu göreceksiniz.

git branch -d [branch_adi]

- Yeni bir geliştirme branchi oluşturmak istiyorsanız aklınızda bulundurmanız gereken bir şey var. İlk olarak Git'in ana branch'i anlaması için ona odaklanmanız gerekir. Eğer bunu yapmazsanız hata alırsınız.
- Eğer Git'ten bir branch silmek istiyorsanız yukarıdaki komutla yapabilirsiniz:

git checkout master
git branch -d test

- Ancak bunu yapmak için silmek istediğiniz branch’te olmamalısınız. Bu durumda ana branch’e gidin ve daha sonra ise oradan oluşturduğunuz branch’i silin.

git merge [branch]

- Son olarak bir geliştirme branch'ine birçok düzenleme yaptığınız zaman olacaktır. Yeterince stabil olduğunda bu branch'i başka bir geliştirme branch'ine bağlamak isteyebnilirsiniz. Bunun için **merge** komutu bulunmaktadır.

GitHub kullanımı hakkında yardımcı kaynaklar

- Türkçe Kaynak: https://www.youtube.com/watch?v=rWG70T7fePg&list=PLPrHLaayVkhnNstGIzQcxxnj6VYvsHBHy&index=1

- İngilizce Kaynak:
- https://www.youtube.com/watch?v=RGOj5yH7evk