

GİTLAB REHBERİ

HAZIRLAYAN:
KAAN
KALAYCI

GitLab Nedir?

GitLab' a kayıt olma

Öncelikle [GitLab Kayıt](#) sayfasından kullanıcı kaydınızı yapabilirsiniz.

- İsterseniz 1. bölgeden Emailiniz ile kayıt olabilirsiniz. İsterseniz de 2. bölgeden Google, GitHub gibi platformlarda hesabınız var ise oradaki hesaplarınız ile GitLab üzerinden kullanıcı hesabı açabilirsiniz.



GitLab.com

First name

Last name

Username

1

Email

We recommend a work email address.

Password



Minimum length is 8 characters.

Register

By clicking Register or registering through a third party you accept the GitLab [Terms of Use](#) and [acknowledge the Privacy Policy and Cookie Policy](#)

Register with:

2

 Google

 GitHub

 Twitter

 Bitbucket

 Salesforce

Already have an account? [Sign in](#)

GitLab kullanımı

Proje oluřturma

- Hesabınıza giriř yaptıktan sonra anasayfanıza ıkan "Create a Project" seeneėine basın ardından tek bařınıza alıřacak iseniz "create Blank Project" seeneėine basınız ve proje oluřturma ekranına gidiniz.



Create a project

Projects are where you store your code, access issues, wiki and other features of GitLab.



Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

- Create blank project ekranı:
 - Bu ekranda 1. blgede projenize istediėiniz řekilde bir proje adı girebilirsiniz.
 - 2. blgede isteėe baėlı olarak projenizi kendinize ve sizin setiėiniz kiřilere grnr yapacak řekilde private yada herkese grnecek řekilde public yapabilirsiniz. aynı zamanda projenize bir read.me texti oluřturabilirsiniz. Yada aık kaynak kodunuzun gvenlik aıkklarını analiz etmek iin SAST oluřturabilirsiniz.
 - 3. blgeye basarak projenizi oluřturabilirsiniz.



Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

Project name

MyFirstProject

1

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL

https://gitlab.com/Kaanklyc11/

Project slug

myfirstproject

Want to organize several dependent projects under the same namespace? [Create a group](#).

Project deployment target (optional)

Select the deployment target

Visibility Level

☒ Private

Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

☐ Public

The project can be accessed without any authentication.

2

Project Configuration

☒ Initialize repository with a README

Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

☐ Enable Static Application Security Testing (SAST)

Analyze your source code for known security vulnerabilities. [Learn more](#).

Create project

Cancel

3

Git kurulumu

- [sit-scm](#) adresine giderek "Click here to download" seçeneğine bastığımızda site otomatik olarak bilgisayarımıza en uygun güncel sürümü bizim için .exe formatında indirecektir. Exe yi çalıştırdıktan sonra isterseniz karşınıza çıkan seçeneklerden istediğiniz ayarları kendinize göre ayarlayabilirsiniz isterseniz de uygulamanın size önerdiği ayarlardan devam ederek sadece "Next" butonuna basarak kurulumu kolayca gerçekleştirebilirsiniz.
- Git kurulumunuzun gerçekleşip gerçekleşmediğini kontrol etmek için bilgisayarınızda komut istemi(cmd) ye girin ve "git --version" komutunu çalıştırın size kullandığınız git versiyonunu gösterecektir.

```
C:\Users\Kaank>git --version
git version 2.41.0.windows.2
```

- Bazı komutlar gireceğimiz öncelikle yazacağımız komut ile gitlab hesabımızı git e bağlayacağız böylelikle push,pull gibi istemleri gerçekleştirebileceğiz. Bu komutu bir kere yazmak yeterlidir her seferinde kullanıcı adınızı ve mailinizi register yapmaya gerek yoktur. Bu komutlarda kullanacağınız user.name ve user.gmail bilgileri GitLab hesabı oluşturduğunuz kullanıcı adı ve mailiniz olmalıdır. Komutun çalışıp çalışmadığını örnekte gördüğümüz gibi kullanıcı adı veya maili belirtmeden komutu çalıştırırsanız size register yaptığınız bilgileri gösterecektir.

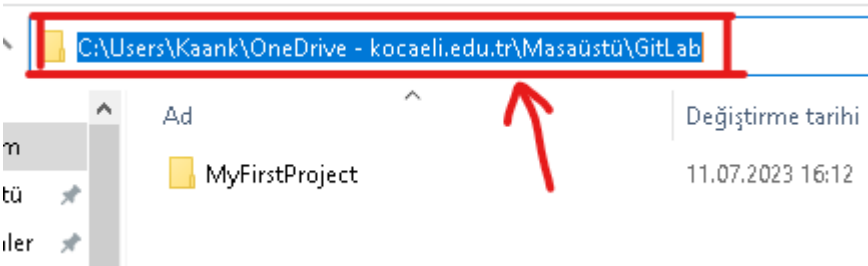
```
C:\Users\Kaank>git config --global user.name "Kaanklyc11"
C:\Users\Kaank>git config --global user.name
Kaanklyc11
C:\Users\Kaank>git config --global user.gmail "kaan.klyc1@hotmail.com"
C:\Users\Kaank>git config --global user.gmail
kaan.klyc1@hotmail.com
```

GitLab'a dosya ve commit ekleme

- Git'e eklemek için öncelikle bir demo proje yada dosya oluşturacağız bunu komut isteminden şu şekilde gerçekleştirebilirsiniz.

```
C:\Users\Kaank>cd desktop  
C:\Users\Kaank\Desktop>mkdir GitLab\MyFirstProject
```

- Oluşturmuş olduğunuz klasöre Git'i başlatmak için klasörümüzün uzantısına sahip cmd komutunda "git init" yazarak git'i başlatıyoruz. Ufak bir bilgi olarak işinizi hızlandıracak bir kısa yol kırmızı bölge ile gösterilen dosyanın url kısmına tıklayıp dosya yolunu cmd olarak değiştirseniz enter tuşuna bastığınızda dosya yolunuza ait bir cmd komut ekranı karşınıza çıkacaktır.



```
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git init  
Initialized empty Git repository in C:/Users/Kaank/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/.git/
```

- "git status" komutu ile git'inizin hangi branchleri olduğunu, herhangi bir commit yapılmadığına dair bilgileri görebilirsiniz.

```
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git status  
On branch master  
  
No commits yet  
  
nothing to commit (create/copy files and use "git add" to track)
```

- Text dosyası oluşturma ve git'e ekleme:
 - 1. bölgede yazdığımız "cd>ReadMe.txt" komutu ile dosyamızın içine bir text dosyası oluşturabiliriz.
 - 2. bölgede git'in durumunu kontrol ettiğimizde untracked files gördüğünü bunu hiçbir şey yapılmadığını ama git add komutu ile git'e ekleyebileceğimizi söylediğini görüyoruz.
 - 3. bölgede "git add ReadMe.txt" yazarak sadece ReadMe.txt dosyasını yada isterseniz "git add ." komutu ile untracked olan bütün dosyaları tek seferde git'e ekleyebilirsiniz.
 - 4. bölgede text dosyasının eklenip eklenmediğini status komutu ile kontrol edebilirsiniz.

```
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>cd>ReadMe.txt 1
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    ReadMe.txt 2
nothing added to commit but untracked files present (use "git add" to track)
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git add . 3
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git status
On branch master

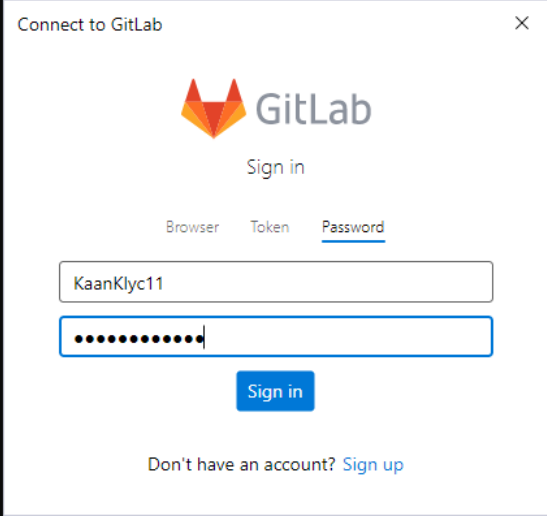
No commits yet 4
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   ReadMe.txt
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>
```

- GitLab e commit ekleme:

- 1. bölgede "git commit -m "....." komutu ile Git'e ekleyeceğimiz commit i giriyoruz. Lakin bu komut tek başına yeterli değil eğer GitLab a girip kontrol ederseniz commit in GitLab a gelmediğini göreceksiniz bu yüzden.
- 2. bölgedeki "git push -u "GitLab proje URL'si" master" komutunu veya "git push -u origin master" komutunu çalıştırarak commit i GitLab a pushluyoruz.
- 3. bölgede push işlemini gerçekleştirmek için bizden GitLab hesabımıza bağlanmamızı istiyor bilgilerimizi yazarak bağlanıyoruz ve commit girdimiz GitLab ımıza pushlanmış oluyor. Commit bilgisi GitLab projemizde master branch ına düşmektedir oradan görebilirsiniz.

```
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git commit -m "Benim ilk commit girdim"
[master (root-commit) 6e7bda9] Benim ilk commit girdim
1 file changed, 1 insertion(+)
create mode 100644 ReadMe.txt

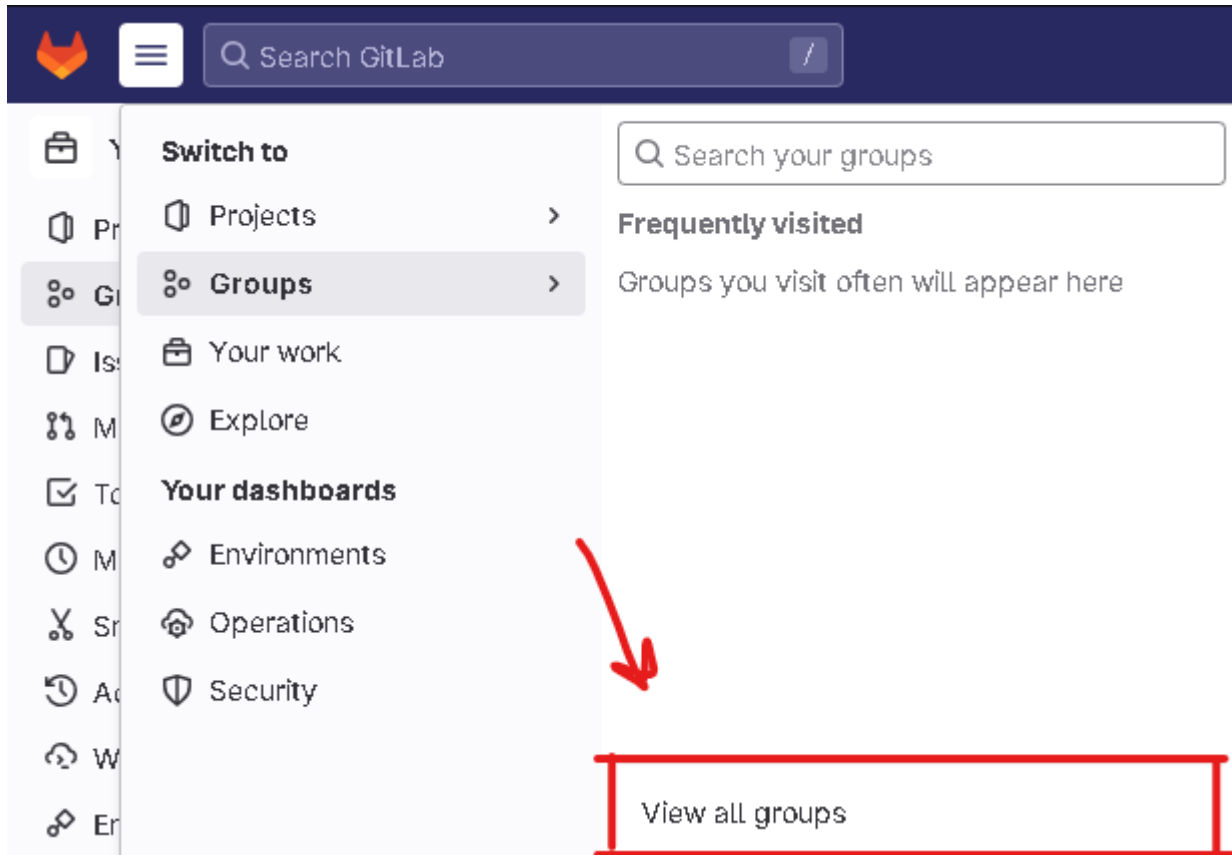
C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git push -u "https://gitlab.com/Kaanklyc11/myfirstproject.git" master
```

A dialog box titled "Connect to GitLab" with a close button (X) in the top right corner. It features the GitLab logo and a "Sign in" button. Below the logo, there are three tabs: "Browser", "Token", and "Password". The "Password" tab is selected. Under the "Password" tab, there is a text input field containing "Kaanklyc11" and a password field with masked characters. A "Sign in" button is located below the password field. At the bottom, there is a link that says "Don't have an account? Sign up".

Başkasının projesini kullanma(Fork işlemleri)

Fork işlemi bir projeyi veya repository'i kopylama işlemidir. Bunlar üzerinde orjinal dosyalara zarar vermeden değişiklikler yapabiliriz.

- Ana sayfada seçenekler menüsünden group kısmında "View All Groups" seçeneğine basınız.



- Ardından çıkan ekranın sağ üst köşesinde "New Group" seçeneğine basarak yeni grup oluşturma ekranına gidiniz.

- 1. bölgede istediğiniz şekilde bir grup adı giriniz.
- 2. bölgede grubun özel mi halka açık mı olduğunu seçiniz.
- 3. bölgede gruptaki rolünüzü ve grubu ne için kullanacağınızı seçiniz.
- 4. bölgede create group seçeneğine basıp grubu oluşturunuz.

Group name

MyFirstGroup

Must start with letter, digit, emoji, or underscore. Can also contain periods, dashes, spaces, and parentheses.

Group URL

https://gitlab.com/ myfirstgroup4860128

Visibility level

Who will be able to see this group? [View the documentation](#)

- ☐ Private
The group and its projects can only be viewed by members.
- ☒ Public
The group and any public projects can be viewed without any authentication.

Now, personalize your GitLab experience

We'll use this to help surface the right features and information to you.

Role

Software Developer

Who will be using this group?

- ☐ My company or team ☒ Just me

What will you use this group for?

I want to learn the basics of Git

Create group

Cancel

- Şimdi ilk projenize geri dönün ve projenin sağ üst kısmındaki fork seçeneğine basınız.



- Fork oluşturma

- 1. bölgede istediğiniz şekilde bir fork ismi giriniz.
- 2. bölgede çalışacağınız grubu seçiniz.
- 3. bölgede forkun kimlere gözükeceğini seçiniz. Ardından forku oluşturunuz.

Project name

MyFirstProject

1

Project URL

https://gitlab.com/

Select a namespace

2

Project slug

myfirstproject

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses,

Want to organize several dependent projects under the same namespace? [Create a group](#)

Project description (optional)

Visibility level ?

☐ Private

Project access must be granted explicitly to each user. If this project is part of a group, access will be

☐ Internal

The project can be accessed by any logged in user.

☒ Public

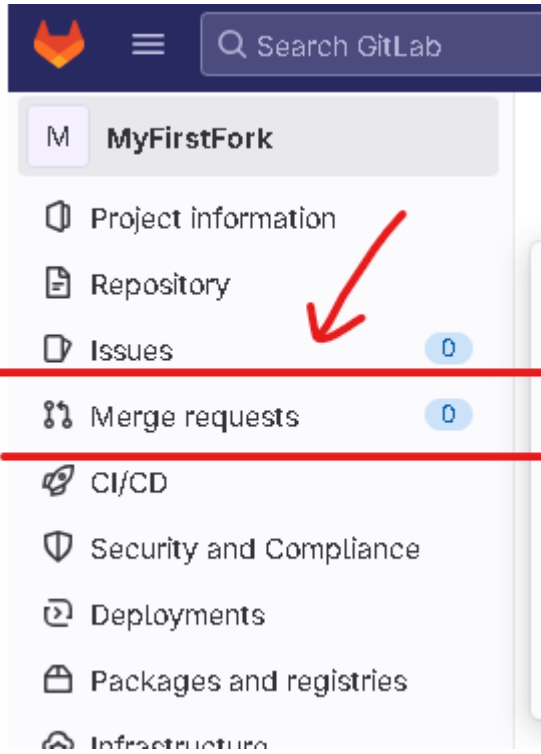
The project can be accessed without any authentication.

3

Fork project

Fork yaptığımız dosyada merge işlemi oluşturma

- Ekranın sağ tarafında "merge requests" bölümüne tıklayınız. Ardından karşınıza çıkan ekranda New merge request seçeneğine tıklayınız.



- Karşımıza çıkan ekranda 1. bölgedeki kısımda nerede değişiklik yaptıysak o branch ı seçiyoruz örneğin master branch ında değişiklik yaptık source branch(kaynak dal) da master branch ını seçiyoruz. Ardından Target branch(hedef dal) da ise yaptığımız değişikliğin nerede uygulanacağını seçiyoruz. Son olarak "compare branches and continue" seçeneğine basıp "New merge request" bölümüne geçiyoruz.

New merge request

Source branch: myfirstgroup4860128/my-first-fork (dropdown: master)

Target branch: KaanKlyc11/myfirstproject (dropdown: master)

Compare branches and continue

- Burada eğer bir değişiklik var ise "description" bölümünün üstünde değişiklikler gözükecektir biz bir değişiklik yapmadığımız için sadece description bölümü mevcut buraya yaptığımız değişiklikleri not olarak ekleyebiliriz. Ve grupta tek olduğumuz için işaretli bölgede kendimizi seçerek merge request i oluşturuyoruz.

New merge request

From myfirstgroup4860128/my-first-fork:master into KaanKlyc11/myfirstproject:master [Change branches](#)

Title (required)

Draft: Master

- ☒ Mark as draft
Drafts cannot be merged until marked ready.

Description

Preview **B** *I* S ~~I~~ `</>` [🔗](#)

- ☰
- ☷
- ☰

📄	📊
---	---

Describe the goal of the changes and what reviewers should be aware of.

Add [description templates](#) to help your contributors to communicate effectively!

Assignee

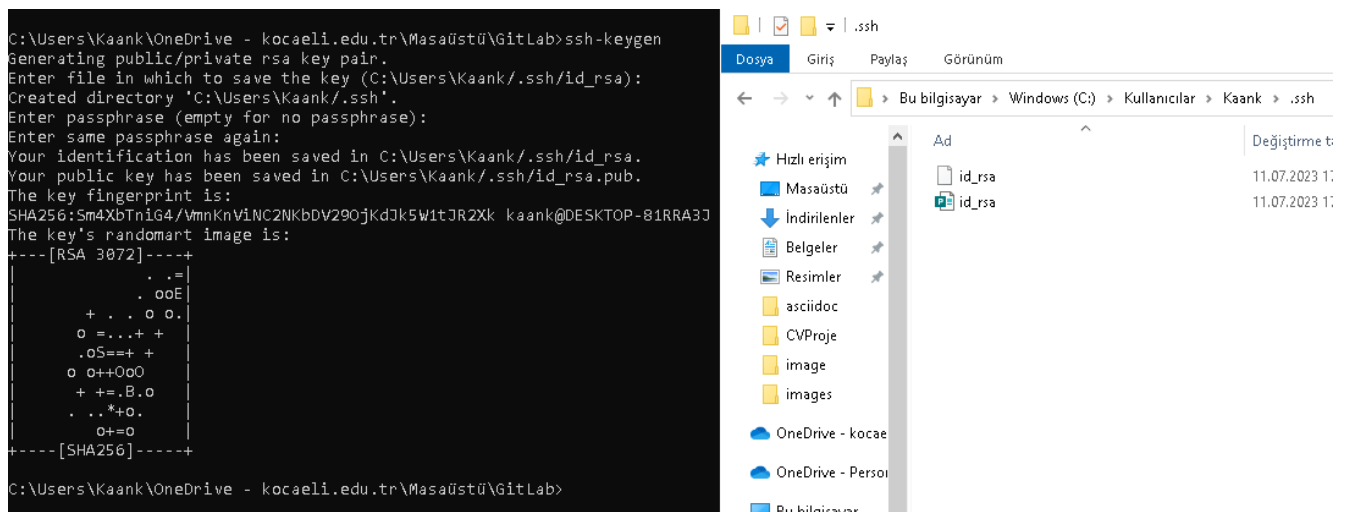
Unassigned

Assign to me

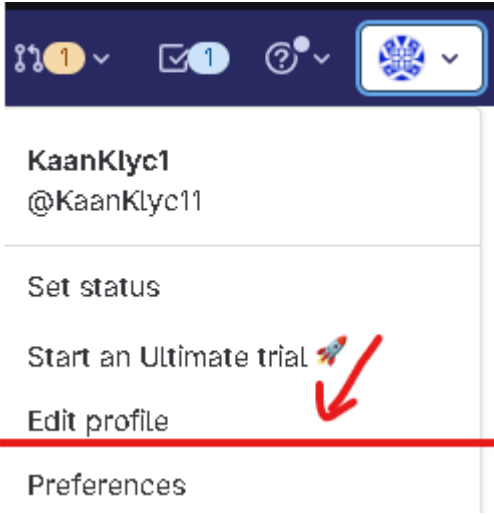
Penjualan

SSH(Secured Shell) Key oluşturma

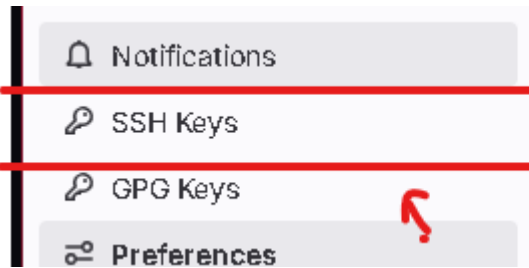
- SSH kimlik doğrulama için kullanılır. Ve SSH key ile yaptığımız bağlantılarda artık GitLab server bizden her seferinde kullanıcı adı ve şifre istemez.
- cmd ye "ssh key-gen" komutunu giriniz ve ardından ayarlara dokunmadan enter tuşuna bir kaç kere basınız bu size ssh key oluşturacaktır. Görselde göreceğiniz gibi .ssh isimli bir dosya oluşmaktadır.



- GitLab a gidin ve ekranın sağ üst köşesinden "preferences" seçeneğini seçin.



- Ekranın sol tarafından SSH Keys seçeneğini seçin.



- 1. bölgeye daha önceden oluşturduğumuz .ssh dosyasına girin. "id_rsa" dosyasını herhangi text görüntüleyebileceğiniz not defteri vb. uygulama ile açınız ve içeriğindeki key'i 1. bölgedeki alana yapıştırınız.
 - 2. bölgede ise isteğinize bağlı anahtarı tanımlayabileceğiniz bir isim giriniz. Ardından "add key" seçeneği ile key'i oluşturunuz.

Key

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQgQDCbBOKLoU05a-
5yleI9TfsY7/5erOQGsb0uVQX4ft7pRvxivYFm+ozBu032Ojr
BYbSeSpy+FGFH6LE9sl3mh9OXC/DvnyuxLynGFbUfMb2A9
3ex4o5ers4XexbhN36w4w3hxqFvHNPZ9TezIU51MK8d3A
0GDwa2UBJ7b1xJpQ1OHBaTeXMOtQ3m+7pZjl18GVmHc+N
+n1hHjudiDGIF0ZDz5Y5q8ANVINMZoEVXDwVB3rCDk5gwl
1AEU= kaank@DESKTOP-81RRA3J

Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nis
sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh

Title

MyKey

Key titles are publicly visible.

GitLab Runner

GitLab Runner, GitLab CI/CD sistemine ait bir bileşendir ve projelerin otomatik olarak derlenmesi, test edilmesi, dağıtılması gibi işlemleri gerçekleştirmek için kullanılır. Bu araç, projenin içinde veya ayrı bir sunucuda çalışabilir ve projenin CI/CD süreçlerini otomatikleştirmeye yardımcı olur. GitLab Runner, .gitlab-ci.yml dosyasında tanımlanan işlemleri yürüterek CI/CD sürecini gerçekleştirir. Bu şekilde, projelerin sürekli entegrasyon ve dağıtımını kolaylaştırır ve otomatik hale getirir.

GitLab Runner kurulumu


- GitLab Runner için sisteminizde bir klasör oluşturun. Ardından [GitLab Runner](#) sayfasından binaries bölümünden işletim sisteminize uygun indirme seçeneğine gidin ve installation bölümünden kendinize uygun bit değerini seçerek .exe dosyasını oluşturduğunuz klasöre indirin.



NOT: Installation bölümünde anlatılan adımları takip ederek kurulumu daha rahat gerçekleştirebilirsiniz.

Binaries

- [Install on GNU/Linux](#)
- [Install on macOS](#)
- [Install on Windows](#)
- [Install on FreeBSD](#)
- [Install nightly builds](#)

Installation

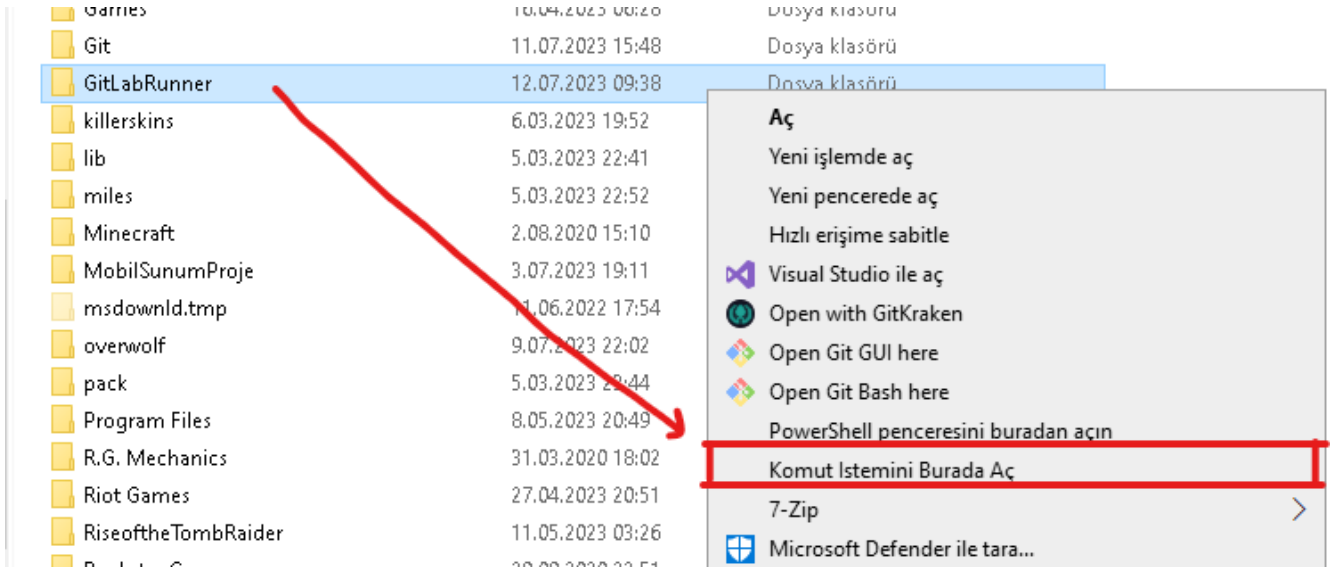
 With GitLab Runner 10, the executable was renamed to

1. Create a folder somewhere in your system, ex.: `C:\GitLa`
2. Download the binary for [64-bit](#)  or [32-bit](#)  and put it in the folder. You can also download a binary for [other tagged release](#).

- Daha önceden text işlemlerinde yaptığımız gibi GitLab-Runner ı indirdiğiniz klasörün dosya yoluna cmd yazarak klasör için cmd bloğunu açınız.

NOT: bu indirme işlemi için cmd bloğu admin olarak çalıştırılmalıdır. Eğer admin olarak

çalıştırmadıysanız indirme düzgün gerçekleşmeyecektir bunun için klasöre "Shift+sağ tık" yaparak klasörü "komut istemini burada aç" seçeneğiyle açın. Bu klasör üzerinde admin cmd si açacaktır. Bu ayar sizde gözüküyor ise "ec menu" gibi ücretsiz programlar ile bu ayarı bilgisayarınıza ekleyebilirsiniz.



- Görseldeki komutları sırası ile çalıştırarak:
 - 1. komut bize gitlab-runner i indirecektir.
 - 2. komut ise indirmenin doğru gerçekleştiğini sağlamak için yazılmış versiyon bilgisi alma komutudur.

```
D:\GitLabRunner>gitlab-runner install
Runtime platform arch=amd64 os=windows pid=21960 revision=b72e108d version=16.1.0

D:\GitLabRunner>gitlab-runner --version
Version: 16.1.0
Git revision: b72e108d
Git branch: 16-1-stable
GO version: go1.19.9
Built: 2023-06-21T21:52:32+0000
OS/Arch: windows/amd64
```

GitLab Runner Register yapma

- [GitLab Runner Register](#) sitesinden işletim sisteminize uygun register yöntemini seçin ve işleme devam edin. Rehberde windows işletim sistemi için devam edecektir.
- 1. bölgede "gitlab-runner register" komutu çalıştırılarak register işlemine başlıyoruz.
- 2. bölgede gitlab a bağlayacağımız için örnek içinde belirtildiği gibi gitlab uzantısını yazıyoruz entere basıp devam ediyoruz.
- 3. bölgede projemize ait register token isteniyor bunu gitlab projemizden temin ediyoruz(register işleminin altında token almayı göreceksiniz)
- 4. bölgede istediğiniz şekilde runner a bir isim veriniz.
- 5. bölgede runner için "ssh, ci" tag larını veriyoruz.
- 6. bölgede isteğe bağlı runner için bir not bırakabilirsiniz. Ve register işleminin başarılı olduğu bilgisini görebilirsiniz.
- 7. bölgede bir executor seçiyoruz burada executor için "shell" yazacağız. Ve register işlemi

başarılı şekilde gerçekleşti bilgisi ekrana gelecektir.

```
D:\GitLabRunner>gitlab-runner register
Runtime platform
Created missing volume system 10
Created missing volume system 10
Enter the gitlab instance URL (for example, https://gitlab.com/):
Enter the registration token:
Enter a description for the runner:
[DESKTOP-8188A33]: my-runner1
Enter tags for the runner (comma-separated):
ssh, ci
Enter optional maintenance note for the runner:
WARNING: Support for registration tokens and runner parameters in the 'register' command has been deprecated in GitLab Runner 13.6 and will be replaced with support for authentication tokens. For more information, see https://gitlab.com/
gitlab-org/gitlab/-/issues/788872
Registering runner... succeeded
Enter an executor: instance, custom, docker, parallels, ssh, docker-autoscaler, docker-machine, docker-windows, shell, virtualbox, kubernetes:
shell
Runner registered successfully. Feel free to start it, but if it's running already the config should be automatically reloaded!
Configuration (with the authentication token) was saved in "D:\GitLabRunner\config.toml"
```

• REGISTER TOKEN NASIL ALINIR?

- GitLab profilimize giriyoruz ve kullanacağımız projemizi seçiyoruz.
- 1. bölgede seçtiğimiz projemiz gözüküyor.
- 2. bölgede projemizin settings bölümüne tıklıyoruz.
- 3. bölgede settings bölümüne tıkladığımızda karşımıza çıkan CI/CD bölümüne tıklıyoruz.
- 4. bölgede CI/CD bölümünün içinde bulunan Runners kısmına geliyoruz ve üzerine tıklıyoruz.
- 5. bölgede Runner içindeki Project runners bölümüne geliyoruz.
- 6. bölgede 3 noktalı ayrıntılar butonuna basıyoruz.
- 7. bölgede karşımıza çıkan registration token kısmını kopyalayıp alıyoruz. Artık kullanabiliriz.

The screenshot shows the GitLab web interface for a project named 'MyFirstProject'. The left sidebar contains a navigation menu with the following items: Project information, Repository, Issues (0), Merge requests (1), CI/CD, Security and Compliance, Deployments, Packages and registries, Infrastructure, Monitor, Analytics, Wiki, Snippets, Settings, General, Integrations, Webhooks, Access Tokens, Repository, Merge requests, CI/CD, Packages and registries, Monitor, and Please Outage. The 'CI/CD' item is highlighted with a red box and labeled '3'. The 'Settings' item is highlighted with a red box and labeled '2'. The 'Runners' section is highlighted with a red box and labeled '4'. The 'Project runners' section is highlighted with a red box and labeled '5'. The 'New project runner' button is highlighted with a red box and labeled '6'. The 'Registration token' field is highlighted with a red box and labeled '7'. The 'Registration token' field contains a masked token '*****' and a warning message 'Support for registration tokens is deprecated'. The 'Registration token' field is also highlighted with a red box and labeled '7'. The 'Registration token' field is also highlighted with a red box and labeled '7'.

1. MyFirstProject

2. Settings

3. CI/CD

4. Runners

5. Project runners

6. New project runner

7. Registration token

Registration token

Support for registration tokens is deprecated

Show runner installation and registration instructions

Reset registration token

- GitLab Runner 1 cmd bloğundan start komutu ile başlatıyoruz. Başlayış başlamadığını register token alırken girdiğimiz runners bölümünden register token bölümünün altında görebilirsiniz.

```
D:\GitLabRunner>gitlab-runner start
Runtime platform arch=amd64 os=windows pid=10348 revision=b72e108d version=16.1.0
```

Project runners

These runners are assigned to this project.

New project runner



Assigned project runners

#25136790 {41NhAbRkV} 🔒

my-runner1

ci

ssh



Remove runner

GitLab CI/CD

CI/CD, istediğiniz zaman sürdürülebilir bir şekilde yayınlayabileceğiniz yazılım geliştirme yoludur.

“CI/CD”, Sürekli Entegrasyon (CI) ve Sürekli Teslimat (CD) uygulamalarının birleşik uygulamalarını ifade eder.

- GitLab CI CI/CD oluştururken öncelikle bize projemizin içine ekleyeceğimiz ".gitlab-ci.yml" isimli .yml uzantılı dosya gerekli bunun için yml uzantılı dosya oluşturabileceğiniz herhangi bir IDE kullanabilirsiniz biz rehberde VS Code kullanacağız.
- VS Code içinde .yml uzantılı görseldeki dosyayı oluşturun. Ve kullanacağınız proje klasörünün içine atın. Örn. açtığımız ilk proje olan "MyFirstProject".

```
.gitlab-ci.yml X
C:\> Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab> .gitlab-ci.yml
1 demo_job_1:
2   tags:
3     - ci
4   script:
5     - echo Hello World
6
```

- Önceden yaptığımız gibi dosya yolundan cmd bloğu açın. İlk projeyi oluştururken kullandığımız "git status", "git add.", "git commit -m" ve "git push -u" komutlarını tekrar kullanarak .yml uzantılı dosyanızı ilk GitLab üzerindeki ilk projenize commit ekleyerek pushlamış oldunuz. GitLab a giderek push ettiğiniz .yml dosyasını görüntüleyebilirsiniz.

```

Microsoft Windows [Version 10.0.19045.3086]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitlab-ci.yml

nothing added to commit but untracked files present (use "git add" to track)

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git add .

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .gitlab-ci.yml

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git commit -m "add .gitlab-ci.yml dosyasi"
[master 21239fe] add .gitlab-ci.yml dosyasi
 1 file changed, 5 insertions(+)
 create mode 100644 .gitlab-ci.yml

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git push -u "https://gitlab.com/Kaanklyc11/myfirstproject.git" master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 348 bytes | 348.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for master, visit:
remote:   https://gitlab.com/Kaanklyc11/myfirstproject/-/merge_requests/new?merge_request%5Bsource_branch%5D=master
remote:
To https://gitlab.com/Kaanklyc11/myfirstproject.git
   6e7bd90..21239fe master -> master
branch 'master' set up to track 'https://gitlab.com/Kaanklyc11/myfirstproject.git/master'.

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>

```

- Şimdi proje dosyanızın içinde ufak bir değişiklik yapacağız ve bunu CI/CD ekranında görüntüleyeceğiz.
 - Proje dosyasının içerisine bir test dosyası oluşturun. Örn. adı "test1" olan txt dosyası oluşturun ve içerisinde bu bir test dosyasıdır yazın. Ve bu dosyayı daha önceki işlemlerde uyguladığımız gibi cmd bloğuyla projenin içerisine pushlayın.

```

Microsoft Windows [Version 10.0.19045.3086]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        test2.txt

nothing added to commit but untracked files present (use "git add" to track)

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git add .

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   test2.txt

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git commit -m "Test2 dosyasi commit edildi"
[master fb7cdd1] Test2 dosyasi commit edildi
 1 file changed, 1 insertion(+)
 create mode 100644 test2.txt

C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab>git push -u "https://gitlab.com/Kaanklyc11/myfirstproject.git" master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 309 bytes | 309.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for master, visit:
remote:   https://gitlab.com/Kaanklyc11/myfirstproject/-/merge_requests/new?merge_request%5Bsource_branch%5D=master
remote:
To https://gitlab.com/Kaanklyc11/myfirstproject.git
   71fb23e..fb7cdd1 master -> master
branch 'master' set up to track 'https://gitlab.com/Kaanklyc11/myfirstproject.git/master'.

```

- GitLab projenizin içerisindeki CI/CD bölümüne tıklayarak yaptığınız commit, push işleminin burada görüntülendiğini göreceksiniz.

The screenshot shows the GitLab CI/CD interface for a project named 'MyFirstProject'. The left sidebar contains a navigation menu with 'CI/CD' highlighted. The main area displays a list of pipelines. The first pipeline, 'Test2 dosyasi commit edildi!', is highlighted with a red box and shows a 'passed' status. The second pipeline, 'add test1.txt', also shows a 'passed' status. The third pipeline, 'add .gitlab-ci.yml dosyasi', shows a 'failed' status.

Status	Pipeline	Triggerer	Stages
passed	Test2 dosyasi commit edildi! #928681602 master • fb7cdd1f latest		
passed	add test1.txt #9286805676 master • 71fb23e9		
failed	add .gitlab-ci.yml dosyasi #928681484 master • 21239fec		

Branch ve Merge işlemleri

Branch işlemleri

- Git Bash uygulamasına girin.
 - 1. bölgede cd komutu ile projemize bağlanıyoruz. Ardından "git branch" komutu ile "MyNewBranch" branch ini oluşturuyoruz.
 - 2. bölgede git bash çalışma ortamını (master) dan (MyNewBranch) e değiştiriyoruz. Böylelikle devamında yaptığımız işlemler MyNewBranch branch inde çalışacaktır.
 - 3. bölgede test3.txt oluşturuyoruz.
 - 4. bölgede her zamanki status kontrolü yeni dosyayı ekleme commit ekleme işlemlerini yapıyoruz.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~
$ cd "C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab\MyFirstProject"

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git branch MyNewBranch

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git checkout MyNewBranch
Switched to branch 'MyNewBranch'

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (MyNewBranch)
$ touch test3.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (MyNewBranch)
$ git status
On branch MyNewBranch
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  ./

nothing added to commit but untracked files present (use "git add" to track)

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (MyNewBranch)
$ git add .

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (MyNewBranch)
$ git commit -m "test3.txt eklendi"
[MyNewBranch 4179782] test3.txt eklendi
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 MyFirstProject/test3.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (MyNewBranch)
$ git status
On branch MyNewBranch
nothing to commit, working tree clean

```

- Bu işlemlerle yeni bir branch oluşturuldu ve içerisine yeni dosya eklendi ama push edilmediği için GitLab üzerinde yeni branch i görüntüleyemiyoruz bu yüzden push etmelisiniz. Ardından GitLab projenizi kontrol ettiğinizde yeni branch ın geldiğini göreceksiniz.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (MyNewBranch)
$ git push origin MyNewBranch
The authenticity of host 'gitlab.com (172.65.251.78)' can't be established.
ED25519 key fingerprint is SHA256:eUXGgm1YGsMAS7vkcX6J0Jd0GHPem5gQp4taiCfCLB8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'gitlab.com' (ED25519) to the list of known hosts.
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 328 bytes | 328.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0

```

Merge işlemleri

- Projenizin içerisini kontrol ederseniz oluşturulan yeni test3.txt in master branch ında olmadığını göreceksiniz çünkü bu text dosyası MyNewBranch içerisine eklendi şimdi master branch a geçerek bu yeni dosyayı merge edip master branch içerisine alacağız.
- 1. bölgede çalıştığımız MyNewBranch ı master branch a geri değiştiriyoruz.
- 2. bölgede "git merge" komutu ile master branch tan MyNewbranch a merge yapıyoruz. Merge işlemi gerçekleşiyor. Ama push etmediğimiz için GitLab ekranında göremiyoruz bu yüzden:
- 3. bölgede yapılan merge işlemini görünür yapmak için "git push" ve "git push origin master" komutları ile push işlemini gerçekleştiriyoruz. Merge işlemi gerçekleşti GitLab projenize giderek kontrol edebilirsiniz.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (MyNewBranch)
$ git checkout master
Switched to branch 'master'

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git merge MyNewBranch
Updating fb7cdd1..4179782
Fast-forward
 MyFirstProject/test3.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 MyFirstProject/test3.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git push
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for master, visit:
remote:   https://gitlab.com/Kaanklyc11/myfirstproject/-/merge_requests/new?merge_request%5Bsource_branch%5D=master
remote:
To https://gitlab.com/Kaanklyc11/myfirstproject.git
 fb7cdd1..4179782 master -> master

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git push origin master
Everything up-to-date

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$

```

Branch silme

- 1. bölgede "git branch -d" komutu ile artık işimize yaramayan MyNewBranch branch ini siliyoruz.
- 2. bölgede bunu işlemi GitLab ortamına taşımak için "git push --delete" ile silme işlemi GitLab a push ediyoruz. Ve işe yaramayan branch ı silmiş oluyoruz.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git branch -d MyNewBranch
Deleted branch MyNewBranch (was 4179782).

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git push origin --delete MyNewBranch
To gitlab.com:Kaanklyc11/myfirstproject.git
 - [deleted]          MyNewBranch

```

GitLab Tags

GitLab'da "tags" (etiketler), belirli bir git repo üzerindeki belirli bir commit'i işaretlemek için kullanılan bir işlemdir. Etiketler, bir sürümün veya bir projenin belirli bir noktasını işaretlemek, yayınlamak veya referans olarak kullanmak için kullanılır.

Tags ekleme

- 1. bölgede "git tag" komutu ile v1.0 tagini ekleyebiliyoruz.
- 2. bölgede "git tag -a v1.1 -m" komutu ile hem tag hemde tag ile birlikte bir commit ekleyebiliyoruz.
- 3. bölgede "git show" komutu ile taglerin hangi işlemi etiketlediğini görüyoruz.
- Son olarak "git push --tags" komutu ile bütün tagleri projeye push ederek ekliyoruz.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git tag v1.0

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git tag
v1.0

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git tag -a v1.1 -m "version 1.1 i yayinlamak icin tag"

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git tag
v1.0
v1.1

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ git show
commit 417978203e05883780d1d0c7a1e5fa93297eb4d7 (HEAD -> master, tag: v1.1, tag: v1.0, origin/master)
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date:   Wed Jul 12 13:42:11 2023 +0300

    test3.txt eklendi

diff --git a/MyFirstProject/test3.txt b/MyFirstProject/test3.txt
new file mode 100644
index 0000000..e69de29

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab/MyFirstProject (master)
$ cd "C:\Users\Kaank\OneDrive - kocaeli.edu.tr\Masaüstü\GitLab"

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git tag
v1.0
v1.1

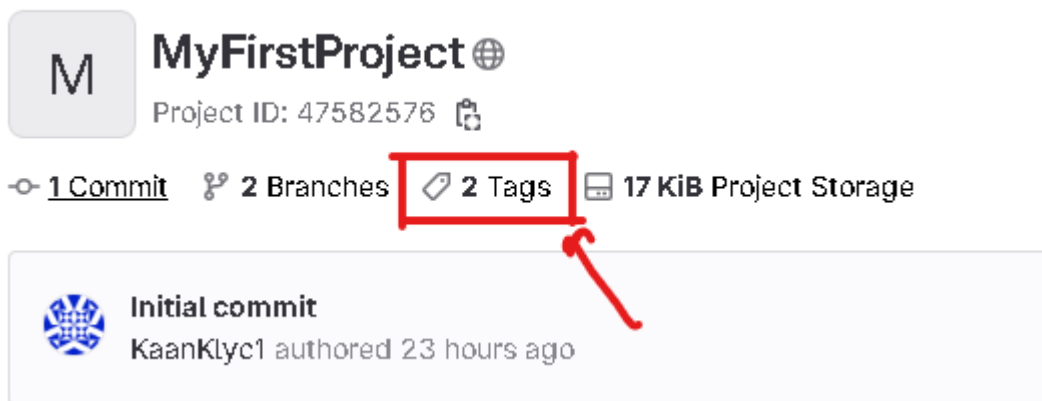
```

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git push --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 175 bytes | 175.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://gitlab.com/KaanKlyc11/myfirstproject.git
* [new tag]          v1.0 -> v1.0
* [new tag]          v1.1 -> v1.1

```

- Ekteki görselde projemizdeki tag sayısını görebiliyoruz.



Tag silme

- 1. bölgede "git tag -d" komutu ile v1.0 tagini siliyoruz.
- 2. bölgede aynı komut ve işlemi v1.1 için gerçekleştiriyoruz.
- 3. bölgede "git push origin -d" komutu ile sildiğimiz tagleri push ederek GitLab proje üzerinden siliyoruz.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git tag -d v1.0
Deleted tag 'v1.0' (was 4179782)

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git tag
v1.1


Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git tag -d v1.1
Deleted tag 'v1.1' (was ac4f277)

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git tag

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git push origin -d v1.0 v1.1
To gitlab.com:KaanKlyc11/myfirstproject.git
- [deleted]          v1.0
- [deleted]          v1.1

```


- Proje üzerinde tag kalmadı.



MyFirstProject

Project ID: 47582576

1 Commit
2 Branches
0 Tags
19 KiB Project Storage



Initial commit
KaankKlyc1 authored 1 day ago

Git Fetch ve Git Pull

git fetch: Uzak depodaki güncelleme bilgilerini yerel depoya indirir, ancak mevcut çalışma kopyasını güncellemez. Yani, dosyalarınızı güncellemez, sadece güncelleme bilgilerini alır ve yereldeki referanslarla eşleştirir.

git pull: git fetch ile git merge işlemini birleştirir. Önce uzaktaki güncelleme bilgilerini git fetch ile alır, ardından yerel çalışma kopyasını otomatik olarak güncellemek için git merge işlemini yapar. Yani, uzak depodan en son güncellemeleri alır ve yerel çalışma kopyasını otomatik olarak günceller.

- Git Fetch İşlemi
 - GitLab proje dosyamızdan örnek olarak test2.txt dosyasının içeriğini güncelleyelim ve bir kaç kelime ekleyelim.
 - "git fetch" komutu kullanarak GitLab üzerinden hangi dosyada değişiklik yaptığımızı görebiliyorsunuz ve isterseniz "git merge" komutu ile merge işlemi gerçekleştirebiliyorsunuz.


```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 298 bytes | 14.00 KiB/s, done.
From https://gitlab.com/Kaanklyc11/myfirstproject
* branch          master      -> FETCH_HEAD
```

test2 - Not Defteri

Dosya Düzen Biçim Görünüm Yardım

Bu Bir Deneme Dosyasıdır

- Git Pull İşlemi

- "git pull" komutu ile GitLab üzerinden içeriğini değişmiş olduğumuz test2.txt dosyasını tek bir komut ile hem fetch işlemini hemde merge işlemini gerçekleştirerek test2.txt dosyamızın içeriğini güncelleyebilirsiniz.

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git pull
From https://gitlab.com/Kaanklyc11/myfirstproject
* branch          master      -> FETCH_HEAD
Updating 4179782..a1a5fcb
Fast-forward
 test2.txt | 3 ++-
1 file changed, 2 insertions(+), 1 deletion(-)
```

test2 - Not Defteri

Dosya Düzen Biçim Görünüm Yardım

Bu Bir Deneme Dosyasıdır

Bu dosyanın içeriğini degisiyoruz.

Git Cherry-Picking

GitLab'da "cherry-picking" (kiraz toplama), belirli bir commit'i seçip başka bir dalda uygulamak için kullanılan bir işlemdir. Bu işlem, belirli bir commitin değişikliklerini veya düzeltmelerini diğer bir dalda hızlıca uygulamak için kullanılır.

- 1. bölgede cherry1.txt dosyası oluşturuluyor.
- 2. bölgede cherry2.txt dosyası oluşturuluyor.
- 3. bölgede bu txt dosyaları GitLab üzerindeki CherryDeneme branch ına push ediliyor. Bu 2 txt ögesi sadece CherryDeneme branch ında mevcut durumdadır. Master branch ına henüz eklenmediler.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ touch cherry1.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git status
On branch cherryDeneme
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        cherry1.txt

nothing added to commit but untracked files present (use "git add" to track)

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git add .

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git commit -m "cherry1.txt eklendi"
[cherryDeneme 1a82afc] cherry1.txt eklendi
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 cherry1.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ touch cherry2.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git add .

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git commit -m "cherry2.txt eklendi"
[cherryDeneme 7bf8a8d] cherry2.txt eklendi
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 cherry2.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git push origin cherryDeneme
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 464 bytes | 464.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for cherryDeneme, visit:
remote:   https://gitlab.com/KaankKlyc11/myfirstproject/-/merge_requests/new?merge_request%5Bsource_branch%5D=cherryDeneme
remote:
To gitlab.com:KaankKlyc11/myfirstproject.git
   a1a5fcb..7bf8a8d  cherryDeneme -> cherryDeneme

```

- CherryDeneme branch ının içeriği:



cherry2.txt eklendi

KaanKlyc1 authored 2 minutes ago

cherryDeneme ▾

myfirstproject /

+ ▾

Name

MyFirstProject

.gitlab-ci.yml

ReadMe.txt

cherry1.txt

cherry2.txt

test1.txt

test2.txt

- Master branch ının içeriği:



Update test2.txt

KaanKlyc1 authored 29 minutes ago

master ▾

myfirstproject /

+ ▾

Name

MyFirstProject

.gitlab-ci.yml

ReadMe.txt

test1.txt

test2.txt

- Cherry-Pick İşlemi:

- 1. bölgede "git log" komutu giriyoruz bu komut bize daha önceden yapılan işlemleri commit

leri ile birlikte listeliyor. Burada kullanacağımız cherry1.txt dosyası olduğundan cherry1.txt dosyasının commit bilgisini kopyalıyoruz.

- 2. bölgede işlemi master branch ında gerçekleştireceğimiz için CherryDeneme branch ını master branch ile değiştiriyoruz artık komutlar master branch için çalışacak.
- 3. bölgede "git cherry-pick" komutu ile önceden kopyaladığımız cherry1.txt dosyasının commit bilgisini kullanarak cherry1.txt dosyasını master branch içerisine çektik.
- 4. bölgede işlemin GitLab üzerinde de görünür halde gerçekleşmesi için "git push origin master" komutunu çalıştırarak yaptığımız cherry-pick işlem sonucunu GitLab üzerinde master branch ına push ettik.

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git log
commit 7bf8a8d8838d0623535ec33efbdcd60c8ca44ce3 (HEAD -> cherryDeneme, origin/cherryDeneme)
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date: Wed Jul 12 15:39:44 2023 +0300

    cherry2.txt eklendi

commit 1a82afc82d2d5c3cd67d8cf08cd36dc874e1dc29
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date: Wed Jul 12 15:38:56 2023 +0300

    cherry1.txt eklendi

commit a1a5fcb3c9375607db3bfb7b0be1e3925a6feced (master)
Author: KaanKlyc1 <kaan.klyc1@hotmail.com>
Date: Wed Jul 12 12:14:47 2023 +0000

    Update test2.txt

commit 417978203e05883780d1d0c7a1e5fa93297eb4d7 (origin/master)
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date: Wed Jul 12 13:42:11 2023 +0300

    test3.txt eklendi

commit fb7cdd1f5a8b4fe9b88c409612865d13917cec07
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date: Wed Jul 12 12:59:27 2023 +0300

    Test2 dosyasi commit edildi

commit 71fb23e90879437f99a6fa105519529856f18877
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date: Wed Jul 12 11:59:09 2023 +0300

    add test1.txt

commit 21239fec2d870938c75b9988f5dd7bbcb9f3234
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date: Wed Jul 12 11:36:51 2023 +0300

    add .gitlab-ci.yml dosyasi

commit 6e7bda9b290bb2f7e444b7c88163985bc43cc9ef
Author: KaanKlyc11 <Kaan.klyc1@hotmail.com>
Date: Tue Jul 11 16:36:30 2023 +0300

    Benim ilk commit girdim

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (cherryDeneme)
$ git checkout master
Switched to branch 'master'

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git cherry-pick 1a82afc82d2d5c3cd67d8cf08cd36dc874e1dc29
[master c8b524e] cherry1.txt eklendi
Date: Wed Jul 12 15:38:56 2023 +0300
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 cherry1.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 281 bytes | 281.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for master, visit:
remote: https://gitlab.com/KaanKlyc11/myfirstproject/-/merge_requests/new?merge_request%5Bsource_branch%5D=master
```

- cherry-pick yapılmış master branch içeriği (cherry1.txt dosyası eklendi)



cherry1.txt eklendi

Kaanklyc1 authored 18 minutes ago

master ▾

myfirstproject /

+ ▾

Name

MyFirstProject

.gitlab-ci.yml

ReadMe.txt

cherry1.txt

test1.txt

test2.txt

Git Checkout, Git Reset, Git Revert

Git Checkout

- Henüz yerel dizinden gönderilmemiş değişiklikleri geri almak için kullanılır.
 - 1. bölgede test2.txt dosyasının local(cihazdaki yerel dosyalarından) içeriğini değiştik ve "git status" kullanarak bunu teyit ettik. Dosyanın içeriği değiştirilmiş.
 - 2. bölgede "checkout test2.txt" komutunu kullanarak test2.txt dosyasının içeriğindeki yapılan son değişikliği kaldırarak, değiştirilmeden önceki haline getirdik.
 - 3. bölgede ise tekrar "git status" komutunu kullanarak test2.txt dosyasının eski haline geri getirildiğini teyit ettik.

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test2.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

1

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git checkout test2.txt
Updated 1 path from the index
```

2

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git status
On branch master
nothing to commit, working tree clean
```

3

Git Reset

- Henüz commit edilmemiş, hafızada commit edilmeyi bekleyen değişiklikleri kaldırmak için kullanılır.
 - 1. bölgede local dosyalarda test2.txt nin içeriğini değiştirmizi teyit ettik.
 - 2. bölgede bu değişikliği staging area ya (push edilmek için verilerin beklemede durduğu bellek-alan) "git add ." komutu ile ekliyoruz.
 - 3. bölgede "git status" ile verideki değişikliğin staging area ya eklendiğini teyit ediyoruz.
 - 4. bölgede "git reset test2.txt" komutu ile staging area ya atılmış test2.txt dosyasını staging area dan geri çıkarıyoruz.
 - 5. bölgede "git status" komutunu tekrar kullanıyoruz ve 1. bölgede olduğu gibi test2.txt dosyasının içeriğinin değiştiğini staging area ya atılması gerektiğini söyleyen uyarıyla karşılaşılıyor. Yani reset komutu ile staging area ya atılmış dosyayı staging area dan çıkartabiliyoruz.

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test2.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git add .
```

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   test2.txt
```

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git reset test2.txt
Unstaged changes after reset:
M       test2.txt
```

```
Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test2.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Git Reset Head~1/2/3

- Yalnızca son eklenen commiti kaldırabilirsin.
- Commit id kullanarak belirli bir commit i kaldıramazsın.
- Yalnızca local den commit kaldırabilirsin.

- 3 tip modu vardır:
 - **git reset soft:**
 - Yalnızca son commiti kaldırabilir.
 - Staging area daki değişiklikleri kaldıramaz.
 - **git reset mixed:**
 - Son local commiti ve staging area daki değişiklikleri kaldırabilir.
 - Varsayılan Reset modudur.
 - **git reset hard:**
 - Son local commiti ve staging area daki değişiklikleri kaldırabilir aynı zamanda değişiklikleri projenin güncel halinden kaldırabilir.
- 1. bölgede test2.txt dosyasının içeriğini localde değiştirdik onu teyit ediyoruz ve değişikliği staging area ya "git add ." komutuyla ekliyoruz.
- 2. bölgede yaptığımız değişikliği belirten "En son commit" commit ini ekliyoruz.
- 3. bölgede "git log --oneline" komutu ile yapılan işlemlerin log bilgilerini görüntülüyoruz ve en son yaptığımız commit işleminin "head" ile tutulduğunu görüyoruz.
- 4. bölgede "git reset head~1" komutu ile head ile tutulan "En son commit" commit ini ve staging area daki değişiklikleri kaldırıyoruz.
- 5. bölgede ise tekrar "git log --oneline" komutunu çalıştırarak komutun çalıştığını kaldırma işleminin gerçekleştiğini teyit ediyoruz.

```

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test2.txt

no changes added to commit (use "git add" and/or "git commit -a")

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git add .

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git commit -m "En son commit"
[master fcdcea9] En son commit
1 file changed, 1 insertion(+), 2 deletions(-)

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git log --oneline
fcdcea9 (HEAD -> master) En son commit
c8b524e (origin/master) cherry1.txt eklendi
a1a5fcb Update test2.txt
4179782 test3.txt eklendi
fb7cdd1 Test2 dosyasi commit edildi
71fb23e add test1.txt
21239fe add .gitlab-ci.yml dosyasi
6e7bda9 Benim ilk commit girdim

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git reset head~1
Unstaged changes after reset:
M   test2.txt

Kaank@DESKTOP-81RRA3J MINGW64 ~/OneDrive - kocaeli.edu.tr/Masaüstü/GitLab (master)
$ git log --oneline
c8b524e (HEAD -> master, origin/master) cherry1.txt eklendi
a1a5fcb Update test2.txt
4179782 test3.txt eklendi
fb7cdd1 Test2 dosyasi commit edildi
71fb23e add test1.txt
21239fe add .gitlab-ci.yml dosyasi
6e7bda9 Benim ilk commit girdim

```

Git Revert

- Belirli bir commit teki değişiklikleri geri almak için kullanılır.
 - 1. bölgede normal "git status" ve "git add ." ve "git commit -m" işlemlerini gerçekleştiriyoruz.
 - 2. bölgede ekstra olarak test2.txt değişikliğini GitLab projemize push ediyoruz.
 - "git log—oneline" komutu ile yapılan son commitin id sini kopyalayarak revert işlemine geçiyoruz.
 - "git revert [commit id]" komutu çalıştırıldığında master branch ı içerisindeki pushlanmış test2.txt verimizi revert ederek test2.txt dosyasını başladığımız noktaya yani değiştirilmeden önceki haline dönüştürüyoruz.
 - "git log --oneline" komutu ile kontrol ettiğimizde de son commit işlemi için Revert komutunun çalıştırıldığını teyit ediyoruz.

