



techcareer

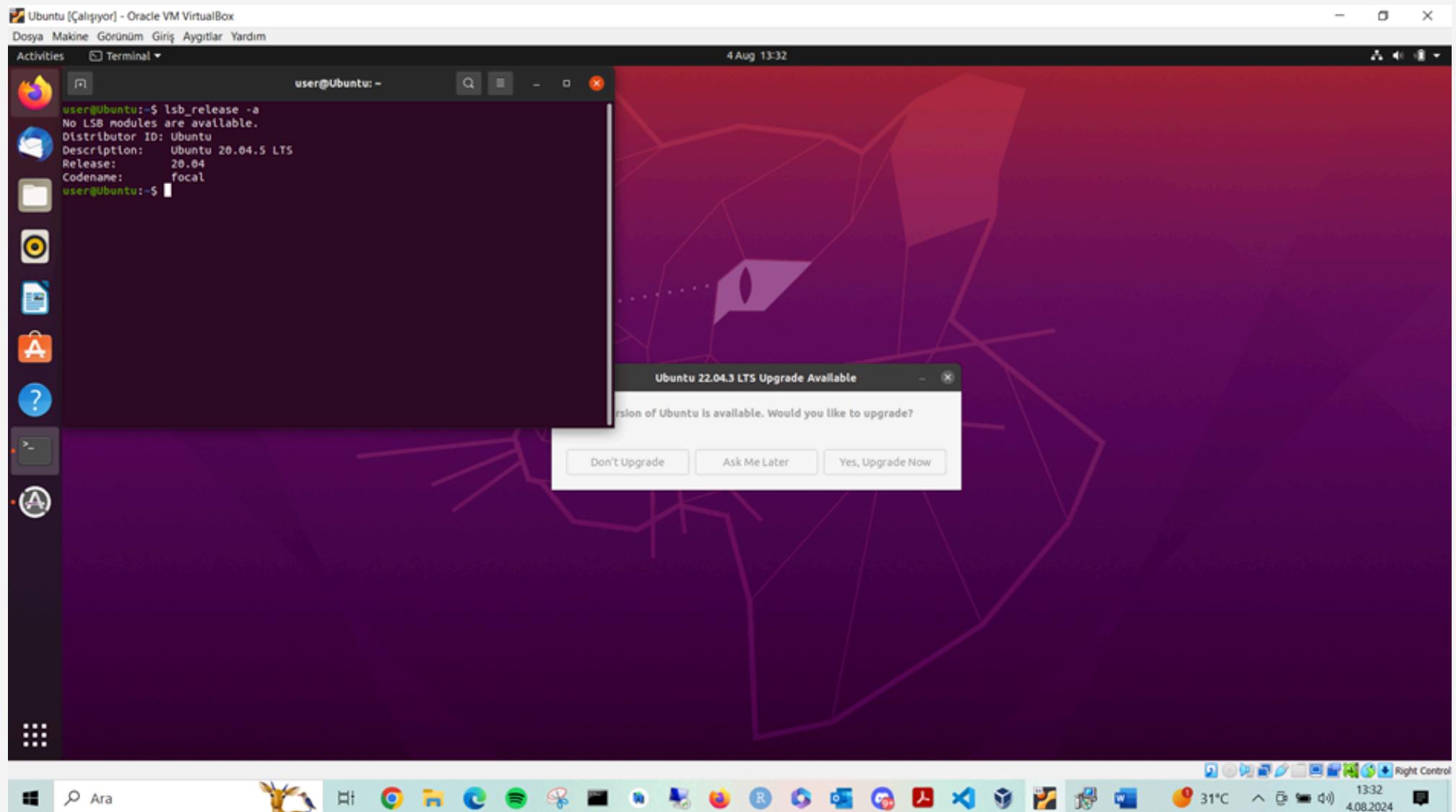
# Senior DevOps Excellence Bootcamp

Berkin Bilgic

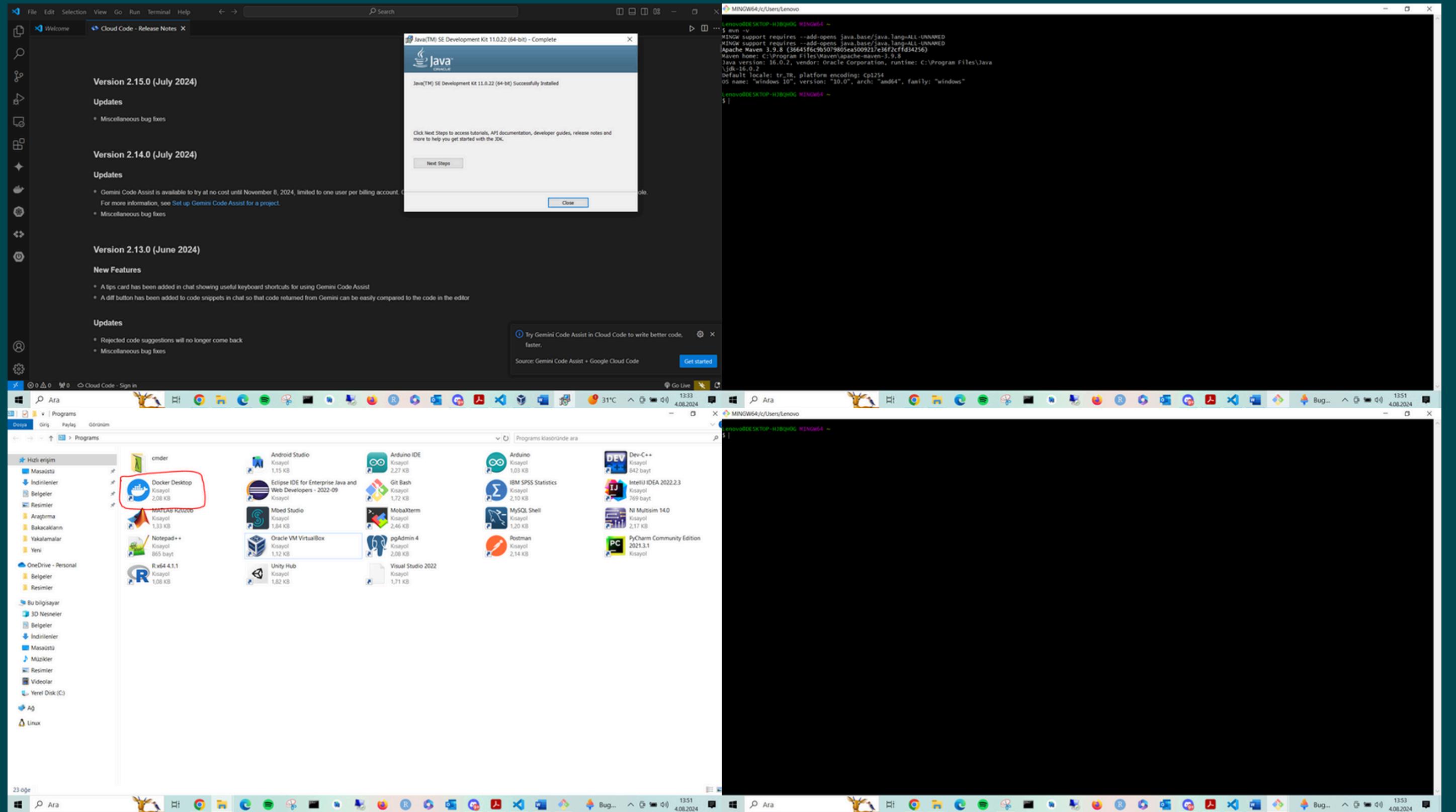


# DevOps Bitirme Projesi

1.adım: Sanal Makine üzerinde (VMBOX) Linux Ubuntu 22.0.4 LTS kuralım (Eğer Linux ve/veya Mac varsa Sanal makine kurmanız gereklidir)



## 2.adım: kurulumları yapalım.



[Back to Agenda Page](#)

**3.adım: port ayarları her bir uygulama için farklı portta olduğunda emin olalım →Linux komut terminalinden derste yaptınız**

```
MINGW64:/c/Users/Lenovo/Desktop/DevOps
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
Compressing objects: 100% (20/20), done.
writing objects: 100% (25/25), 28.10 KiB | 4.68 MiB/s, done.
Total 25 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/BerkinBilge/Senior_DevOp_Tech.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ netstat -n|ptu

Displays protocol statistics and current TCP/IP network connections.

NETSTAT [-a] [-b] [-e] [-f] [-n] [-o] [-p proto] [-r] [-s] [-t] [-x] [-y] [interval]

-a          Displays all connections and listening ports.
-b          Displays the executable involved in creating each connection or
           listening port. In some cases well-known executables host
           multiple independent components, and in these cases the
           sequence of components involved in creating the connection
           or listening port is displayed. In this case the executable
           name is in [] at the bottom, on top is the component it called,
           and so forth until TCP/IP was reached. Note that this option
           can be time-consuming and will fail unless you have sufficient
           permissions.
-e          Displays Ethernet statistics. This may be combined with the -s
           option.
-f          Displays Fully Qualified Domain Names (FQDN) for foreign
           addresses.
-n          Displays addresses and port numbers in numerical form.
-o          Displays the owning process ID associated with each connection.
-p proto    Shows connections for the protocol specified by proto; proto
           may be any of: TCP, UDP, TCPv6, or UDPv6. If used with the -s
           option to display per-protocol statistics, proto may be any of:
           IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, or UDPv6.
-q          Displays all connections, listening ports, and bound
           nonlistening TCP ports. Bound nonlistening ports may or may not
           be associated with an active connection.
-r          Displays the routing table.
-s          Displays per-protocol statistics. By default, statistics are
           shown for IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, and UDPv6;
           the -p option may be used to specify a subset of the default.
-t          Displays the current connection offload state.
-x          Displays NetworkDirect connections, listeners, and shared
           endpoints.
-y          Displays the TCP connection template for all connections.
           Cannot be combined with the other options.
interval   Redisplays selected statistics, pausing interval seconds
           between each display. Press CTRL+C to stop redisplaying
           statistics. If omitted, netstat will print the current
           configuration information once.

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ |
```

**4.adım: GitHub repository üzerinden devops\_project adında repository açalım.  
GitHub repositoryReadMe.md adında dosya oluşturulun uygun formatta yazalım sonrasında local bilgisayarımızda pull yapalım**

The screenshot shows a Microsoft Edge browser window with multiple tabs open at the top. The main content is a GitHub repository page for 'Senior\_DevOp\_Tech'. The repository details are as follows:

- Code**: main branch, 1 Branch, 0 Tags
- Commits**: BerkinBilgc first commit (1855b7f, 13 minutes ago), 1 Commit. Sub-folders: 1\_Windows (first commit, 13 minutes ago) and 2\_Linux Ubuntu (first commit, 13 minutes ago).
- About**: No description, website, or topics provided.
- Activity**: 0 stars, 1 watching, 0 forks.
- Releases**: No releases published. Create a new release.
- Packages**: No packages published. Publish your first package.
- Languages**: Shell 100.0%

A large green 'Add a README' button is prominently displayed on the left side of the repository page. The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray indicating it's 15:03 on August 4, 2024, with a temperature of 31°C.

## 5. Adım.

Git nedir ? VCS açılımı nedir ? iyi bir commit özellikleri nelerdir ?

.adım: git ayarlarını user.name, user.password yapalım.

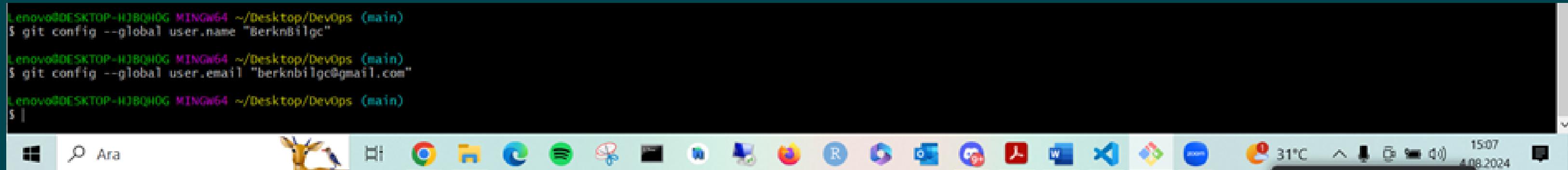
**Git Nedir?** Git, yazılım projelerinde kodu yönetmek için kullanılan dağıtık bir versiyon kontrol sistemidir.

**VCS Nedir?** VCS, "Version Control System" (Versiyon Kontrol Sistemi) demektir. Kod değişikliklerini izler ve yönetir.

**İyi Bir Commit Özellikleri:**

1. **Anlamlı Mesaj:** Kısa ve net.
2. **Kapsamlı Değişiklikler:** Tek bir mantıksal değişiklik.
3. **Atomik:** Tam veya hiç değişiklik yapmaz.
4. **Test Edilebilir:** Değişiklikler test edilmelidir.
5. **Düzenli ve Tutarlı:** Düzenli olarak yapılmalıdır.
6. **Dokümantasyon:** Ek açıklamalar faydalıdır.

Bu özellikler, kod yönetimini kolaylaştırır ve projeyi sürdürülebilir kılar.

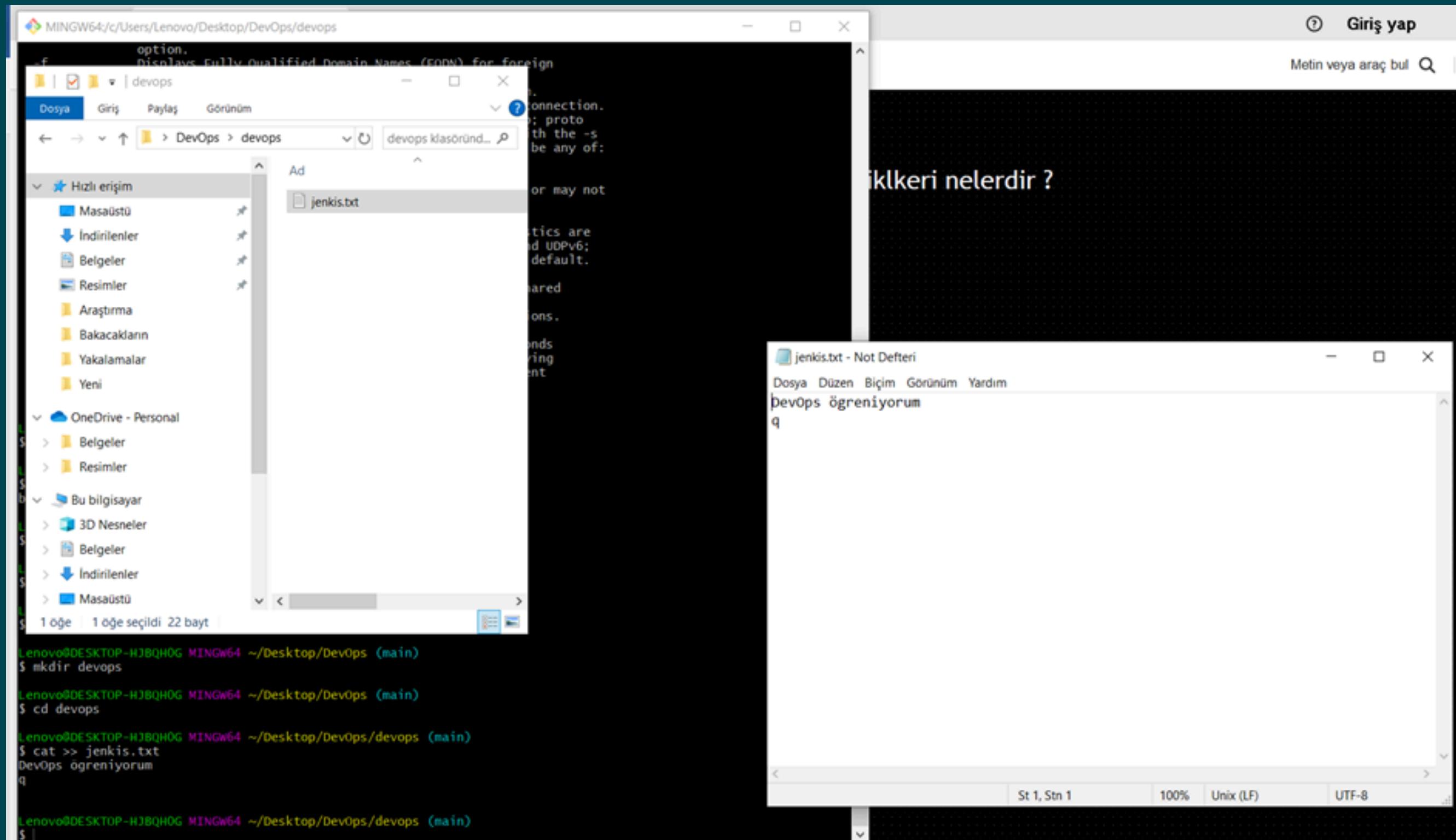


```
Lenovo@DESKTOP-HJ8QHOG MINGW64 ~/Desktop/DevOps (main)
$ git config --global user.name "BerknBilge"
Lenovo@DESKTOP-HJ8QHOG MINGW64 ~/Desktop/DevOps (main)
$ git config --global user.email "berknbilge@gmail.com"
Lenovo@DESKTOP-HJ8QHOG MINGW64 ~/Desktop/DevOps (main)
$ |
```

The screenshot shows a Windows desktop environment. At the bottom is a taskbar with several pinned icons, including File Explorer, Microsoft Edge, and various application icons. Above the taskbar is a system tray showing the date (4.08.2024), time (15:07), battery level (31°C), and other system status indicators. The main focus is a terminal window in the center, which is a dark-themed terminal application. It displays three commands entered at the prompt: 'git config --global user.name "BerknBilge"', 'git config --global user.email "berknbilge@gmail.com"', and a final command starting with '\$ |'. The terminal window has a black background with white text.

6.

adım: Linux komutlarıları dizin adı "devops" ve dosya adı "jenkins.txt" oluşturalım ve "DevOps öğreniyorum" yazalım.



## 7.adım: Git staged area ve unstaged area nedir ?

### Staged Area (Sahneleme Alanı):

- Staged area, commit edilmek üzere hazırlanmış dosyaların bulunduğu alandır.
- git add komutuyla dosyalar bu alana eklenir.
- Bu alanda bulunan dosyalar, bir sonraki commit işlemine dahil edilecektir.

### Unstaged Area (Sahneleme Alanı Dışı):

- Unstaged area, çalışma dizininde yapılan fakat henüz staged area'ya eklenmemiş değişikliklerin bulunduğu alandır.
- Bu değişiklikler, git add komutuyla staged area'ya taşınana kadar commit işlemine dahil edilmez.

### Özetle:

- Staged Area: Commit edilmeye hazır değişiklikler.
- Unstaged Area: Henüz commit edilmeye hazır olmayan değişiklikler.

**8.adım: bu dizindeki dosyaları oluşturduğumuz github repository gönderelim. NOT: main adında branch oluşturalım.**

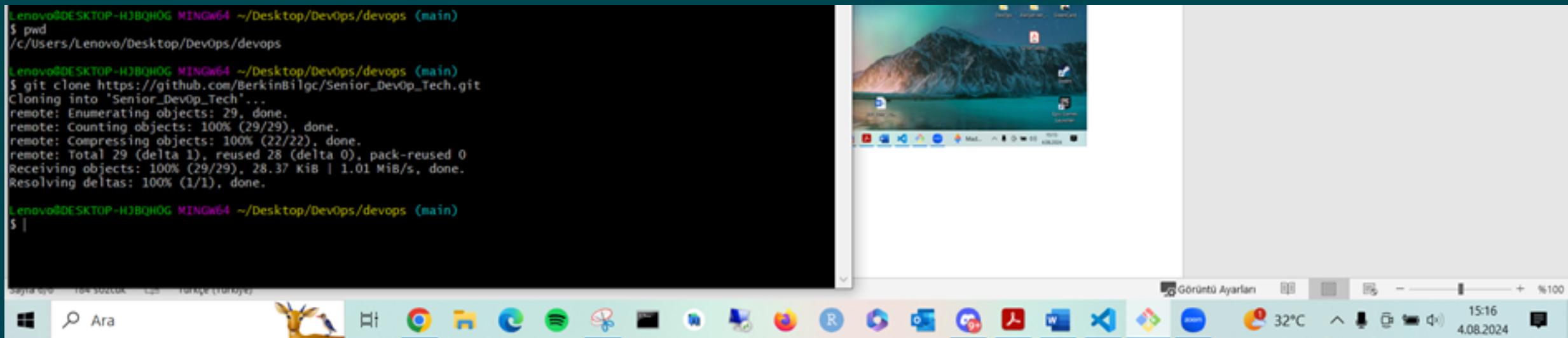
The image shows a Windows desktop environment. On the left, a terminal window titled 'MINGW64:/c/Users/Lenovo/Desktop/DevOps/devops' displays a series of git commands being run:

```
Lenovo@DESKTOP-HJBQHOG MINGW64 ~/Desktop/DevOps/devops (main)
$ git add .
warning: LF will be replaced by CRLF in devops/jenkins.txt.
The file will have its original line endings in your working directory
Lenovo@DESKTOP-HJBQHOG MINGW64 ~/Desktop/DevOps/devops (main)
$ git commit -m "second commit"
[main 9f0da6c] second commit
 1 file changed, 2 insertions(+)
 create mode 100644 devops/jenkins.txt
Lenovo@DESKTOP-HJBQHOG MINGW64 ~/Desktop/DevOps/devops (main)
$ git branch -M "main"
Lenovo@DESKTOP-HJBQHOG MINGW64 ~/Desktop/DevOps/devops (main)
$ git remote add origin https://github.com/BerkinBilgc/Senior_DevOp_Tech.git
error: remote origin already exists.
Lenovo@DESKTOP-HJBQHOG MINGW64 ~/Desktop/DevOps/devops (main)
$ git push -u origin master
error: src refspec master does not match any
error: failed to push some refs to 'https://github.com/BerkinBilgc/Senior_DevOp_Tech.git'
Lenovo@DESKTOP-HJBQHOG MINGW64 ~/Desktop/DevOps/devops (main)
$ git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 381 bytes | 381.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/BerkinBilgc/Senior_DevOp_Tech.git
 1855b7f..9f0da6c main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
Lenovo@DESKTOP-HJBQHOG MINGW64 ~/Desktop/DevOps/devops (main)
$
```

On the right, a file explorer window shows the contents of the 'devops' folder on the desktop. The folder contains several files and subfolders:

- Yeni Metin Belgesi.txt
- Notlar
- Github
- Business
- Programs
- Yüklemeler
- ExamplesCo...
- CV
- Bu fotoğraf hakkında...
- Harç Ücreti Odendi.pdf
- Delter Ücreti Odendi.pdf
- Borda Academy
- 978-3-319... Motivation...
- Data Engineerin...
- Informatics...
- Pictures
- Araştırma
- Zübre
- Senior Queue Systems Pro...
- ENG
- Kariyer.net DevOps...
- Askerlik
- Senior DevOps...
- Ders Dogr...
- DevOps
- Kariyer.net...
- GreenCard
- 1216f7d89b...
- Steam
- Epic Games Launcher
- ABJDM.ID...

## 9.adım: oluşturduğumuz github repository local bilgisarımızda bir yere clone yapalım

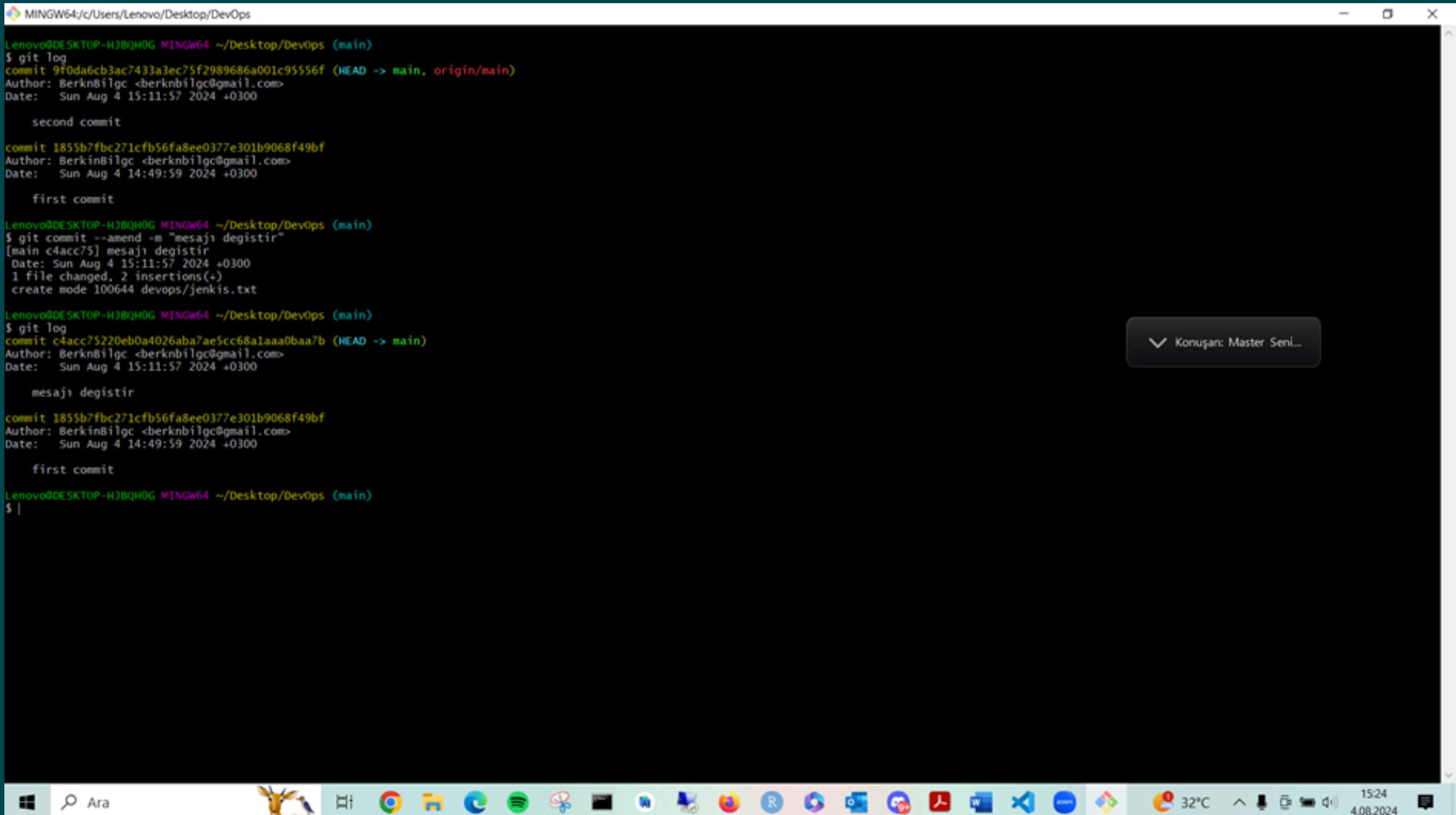


```
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops (main)
$ pwd
/c/Users/Lenovo/Desktop/DevOps/devops

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops (main)
$ git clone https://github.com/BerkinBilgc/Senior_Develop_Tech.git
Cloning into 'Senior_Develop_Tech'...
remote: Enumerating objects: 29, done.
remote: Counting objects: 100% (29/29), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 29 (delta 1), reused 28 (delta 0), pack-reused 0
Receiving objects: 100% (29/29), 28.37 KiB | 1.01 MiB/s, done.
Resolving deltas: 100% (1/1), done.

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops (main)
$ |
```

## 10.adım: en son commitimizin commit içeriğini değiştirilelim (tips: --amend)



```
MINGW64:/c/Users/Lenovo/Desktop/DevOps

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git log
commit 9f0da6cb3ac7433a3ec75f2989686a001c95556f (HEAD -> main, origin/main)
Author: BerknBilgc <berknbilgc@gmail.com>
Date: Sun Aug 4 15:11:57 2024 +0300

    second commit

commit 1855b7fb271cfb56fa8ee0377e301b9068f49bf
Author: BerknBilgc <berknbilgc@gmail.com>
Date: Sun Aug 4 14:49:59 2024 +0300

    first commit

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git commit --amend -m "mesajı degistir"
[main c4acc75] mesajı degistir
Date: Sun Aug 4 15:11:57 2024 +0300
1 file changed, 2 insertions(+)
create mode 100644 devops/jenkins.txt

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git log
commit c4acc75220eb0a4026aba7ae5cc68a1aaa0baa7b (HEAD -> main)
Author: BerknBilgc <berknbilgc@gmail.com>
Date: Sun Aug 4 15:11:57 2024 +0300

    mesajı degistir

commit 1855b7fb271cfb56fa8ee0377e301b9068f49bf
Author: BerknBilgc <berknbilgc@gmail.com>
Date: Sun Aug 4 14:49:59 2024 +0300

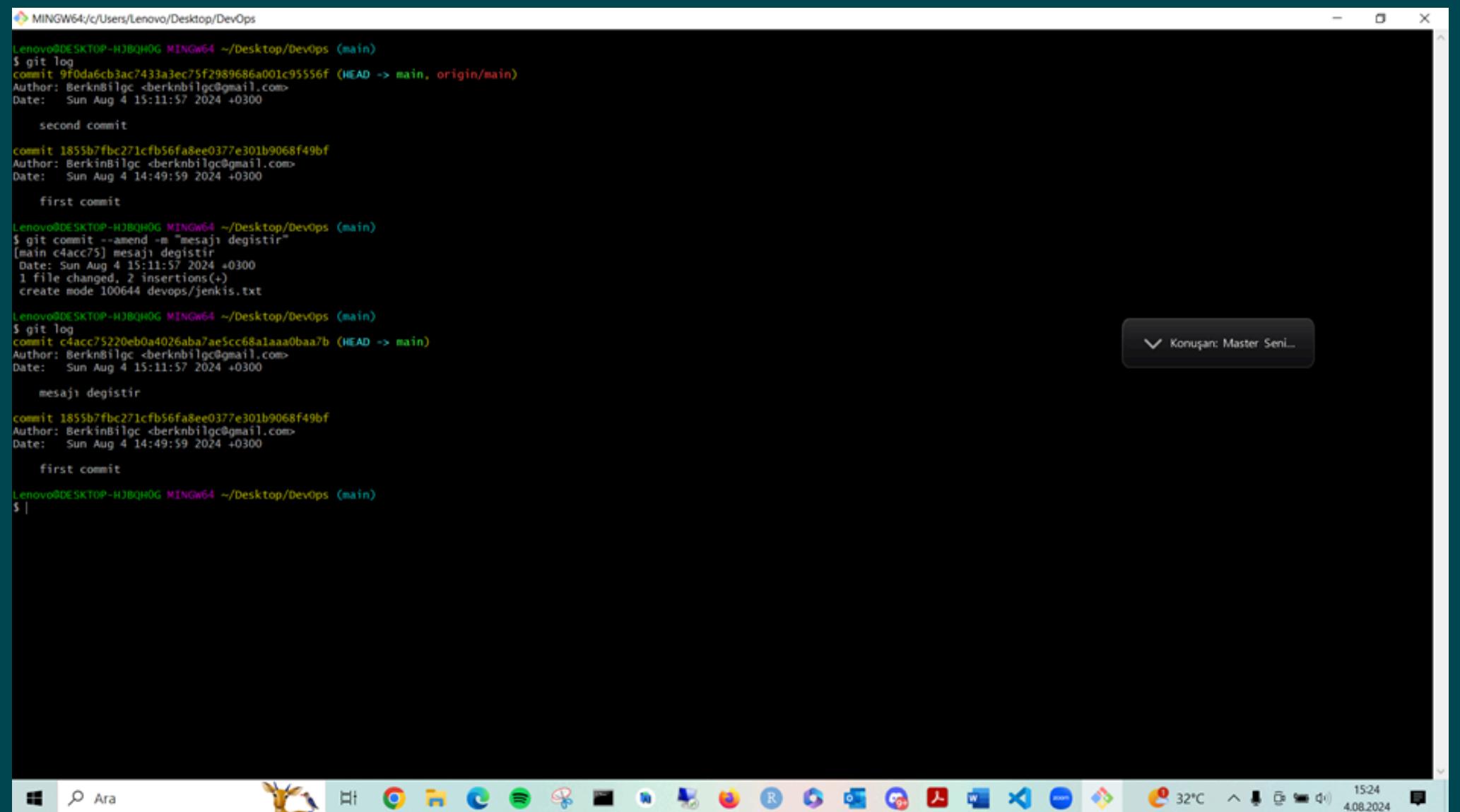
    first commit

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ |
```

## 11.adım: git log ve git status ne iş yapıyordu ?

**git log:** Commit geçmişini gösterir. Commit hash'i, yazar, tarih ve mesajı içerir.

**git status:** Çalışma dizinindeki ve staged area'daki dosyaların durumunu gösterir. Hangi dosyaların değiştiğini ve commit için hazır olduğunu belirtir.



```
MINGW64:/c/Users/Lenovo/Desktop/DevOps (main)
$ git log
commit 9f0da6cb3ac7433a3ec75f2989686a001c95556f (HEAD -> main, origin/main)
Author: BerkınBilgç <berknbilgç@gmail.com>
Date:   Sun Aug 4 15:11:57 2024 +0300

    second commit

commit 1855b7fbc271cfb56fa8ee0377e301b9068f49bf
Author: BerkınBilgç <berknbilgç@gmail.com>
Date:   Sun Aug 4 14:49:59 2024 +0300

    first commit

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git commit --amend -m "mesajı degistir"
[main c4acc75] mesajı degistir
Date: Sun Aug 4 15:11:57 2024 +0300
1 file changed, 2 insertions(+)
create mode 100644 devops/jenkis.txt

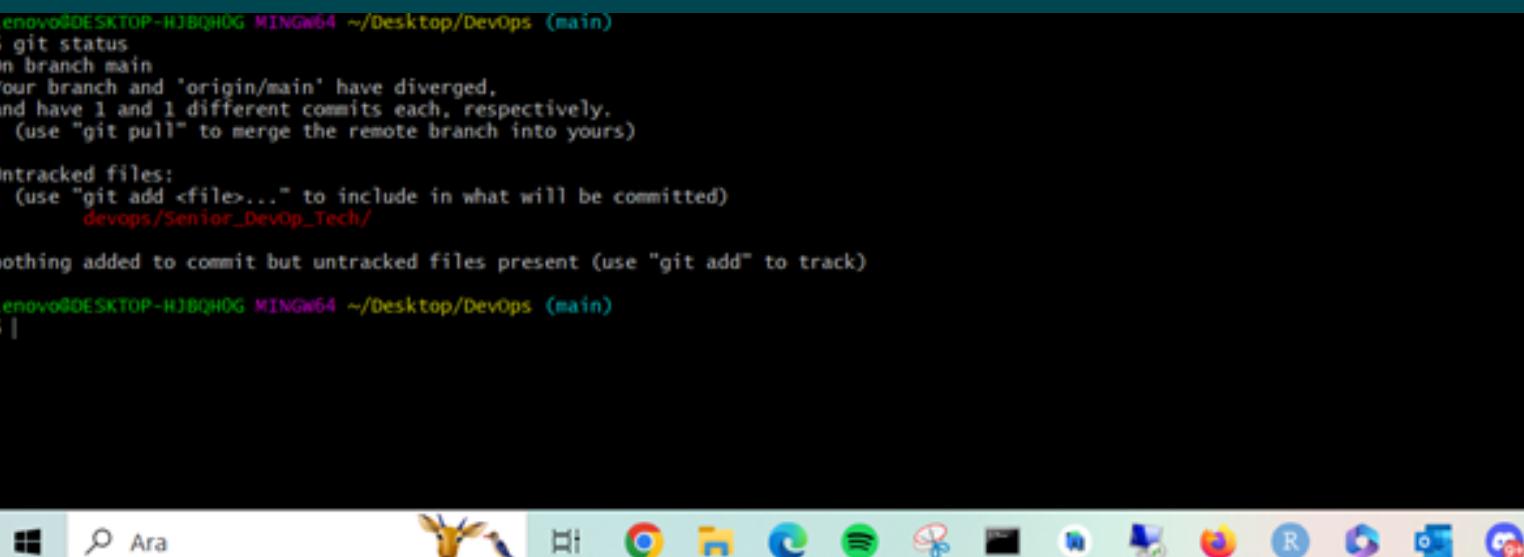
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git log
commit c4acc75220eb0a4026aba7ae5cc68a1aa0baa7b (HEAD -> main)
Author: BerkınBilgç <berknbilgç@gmail.com>
Date:   Sun Aug 4 15:11:57 2024 +0300

    mesajı degistir

commit 1855b7fbc271cfb56fa8ee0377e301b9068f49bf
Author: BerkınBilgç <berknbilgç@gmail.com>
Date:   Sun Aug 4 14:49:59 2024 +0300

    first commit

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ |
```



```
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git status
On branch main
Your branch and 'origin/main' have diverged,
and have 1 and 1 different commits each, respectively.
  (use "git pull" to merge the remote branch into yours)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    devops/Senior_Devo_Tech/

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

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ |
```

## 12.adım: backend adında branch oluşturalım. bu branche bir takım dizinler ekleyelim. commit yapalım. merge işleminde fast-forward yapalım

```
MINGW64:/c/Users/Lenovo/Desktop/DevOps
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git add .
git: 'add' is not a git command. See 'git --help'.
The most similar commands are
  add
  am

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git commit -m "merge oncesinde commit"
On branch main
Your branch and 'origin/main' have diverged,
and have 1 and 1 different commits each, respectively.
(use "git pull" to merge the remote branch into yours)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    devops/Senior_DevOp_Tech/
nothing added to commit but untracked files present (use "git add" to track)

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git branch backend
Switched to branch 'backend'

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git checkout backend
Switched to branch 'backend'

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (backend)
$ git add .
warning: adding embedded git repository: devops/Senior_DevOp_Tech
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:   git submodule add <url> devops/Senior_DevOp_Tech
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:   git rm --cached devops/Senior_DevOp_Tech
hint: See "git help submodule" for more information.

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (backend)
$ git commit -m "backend"
[backend 3693348] backend
 1 file changed, 1 insertion(+)
 create mode 160000 devops/Senior_DevOp_Tech

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (backend)
$ |
```

```
MINGW64:/c/Users/Lenovo/Desktop/DevOps
Your branch and 'origin/main' have diverged,
and have 1 and 1 different commits each, respectively.
(use "git pull" to merge the remote branch into yours)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    devops/Senior_DevOp_Tech/
nothing added to commit but untracked files present (use "git add" to track)

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git branch backend
Switched to branch 'backend'

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git checkout backend
Switched to branch 'backend'

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (backend)
$ git add .
warning: adding embedded git repository: devops/Senior_DevOp_Tech
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:   git submodule add <url> devops/Senior_DevOp_Tech
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:   git rm --cached devops/Senior_DevOp_Tech
hint: See "git help submodule" for more information.

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (backend)
$ git commit -m "backend"
[backend 3693348] backend
 1 file changed, 1 insertion(+)
 create mode 160000 devops/Senior_DevOp_Tech

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (backend)
$ git checkout main
Warning: unable to rmdir 'devops/Senior_DevOp_Tech': Directory not empty
Switched to branch 'main'
Your branch and 'origin/main' have diverged,
and have 1 and 1 different commits each, respectively.
(use "git pull" to merge the remote branch into yours)

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git merge backend
Updating c4acc75..3693348
Fast-forward
  devops/Senior_DevOp_Tech | 1 +
  1 file changed, 1 insertion(+)
 create mode 160000 devops/Senior_DevOp_Tech

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ |
```

### 13.adım: Git GUI ve Git CLI nedir ?

**Git GUI:** Grafiksel kullanıcı arayüzü kullanarak Git işlemlerini yapmanızı sağlar. Butonlar ve menüler ile görsel olarak yönetim kolaylığı sunar.

**Git CLI:** Komut satırı arayüzü kullanarak Git komutlarını yazılı olarak girmenizi sağlar. Daha esnek ve güçlü komutlar sunar.

### 14.adım: frontend adında branch oluşturalım. bu branche bir takım dizinler ekleyelim. commit yapalım.

merge işleminde no-fast-forward yapalım

### 13.adım: Git GUI ve Git CLI nedir ?

```
MINGW64:/c/Users/Lenovo/Desktop/DevOps
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git add .

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git commit -m "merge öncesi commit 2"
[main 801d8aa] merge öncesi commit 2
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Backend/Deneme2.txt
 create mode 100644 Frontend/Deneme2.txt

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git branch frontend
bash: git: command not found

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git branch frontend
bash: git: command not found

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git checkout frontend
Switched to branch 'frontend'

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (frontend)
$ git add .
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (frontend)
$ git commit -m "frontend"
On branch frontend
nothing to commit, working tree clean

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (frontend)
$ git checkout main
Switched to branch 'main'
Your branch and 'origin/main' have diverged,
and have 3 and 1 different commits each, respectively.
 (use "git pull" to merge the remote branch into yours)

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git merge frontend --no-ff
Already up to date.

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ |
```

```
MINGW64:/c/Users/Lenovo/Desktop/DevOps
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git branch frontend
bash: git: command not found

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git branch frontend
bash: git: command not found

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git checkout frontend
Switched to branch 'frontend'

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (frontend)
$ git add .

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (frontend)
$ git commit -m "frontend"
On branch frontend
nothing to commit, working tree clean

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (frontend)
$ git checkout main
Switched to branch 'main'
Your branch and 'origin/main' have diverged,
and have 3 and 1 different commits each, respectively.
 (use "git pull" to merge the remote branch into yours)

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git merge frontend --no-ff
Already up to date.

Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ git log
commit 801d8aa372e5ca4ee1ee6488d6cf5e7c69de490c (HEAD -> main, frontend)
Author: Berknbilg <berknbilg@gmail.com>
Date:   Sun Aug 4 15:36:33 2024 +0300

    merge öncesi commit 2

commit 3693348267b5375082b8abd6f625506d792b3959 (backend)
Author: Berknbilg <berknbilg@gmail.com>
Date:   Sun Aug 4 15:30:59 2024 +0300

    backend

commit c4acc75220eb0a4026aba7ae5ccf68a1aaa0baa7b
Author: Berknbilg <berknbilg@gmail.com>
Date:   Sun Aug 4 15:11:57 2024 +0300

    mesajı degistir

commit 1855b7fbc271cfb56fa8ee0377e301b9068f49bf
Author: BerkinBilg <berkinbilg@gmail.com>
Date:   Sun Aug 4 14:49:59 2024 +0300

    first commit

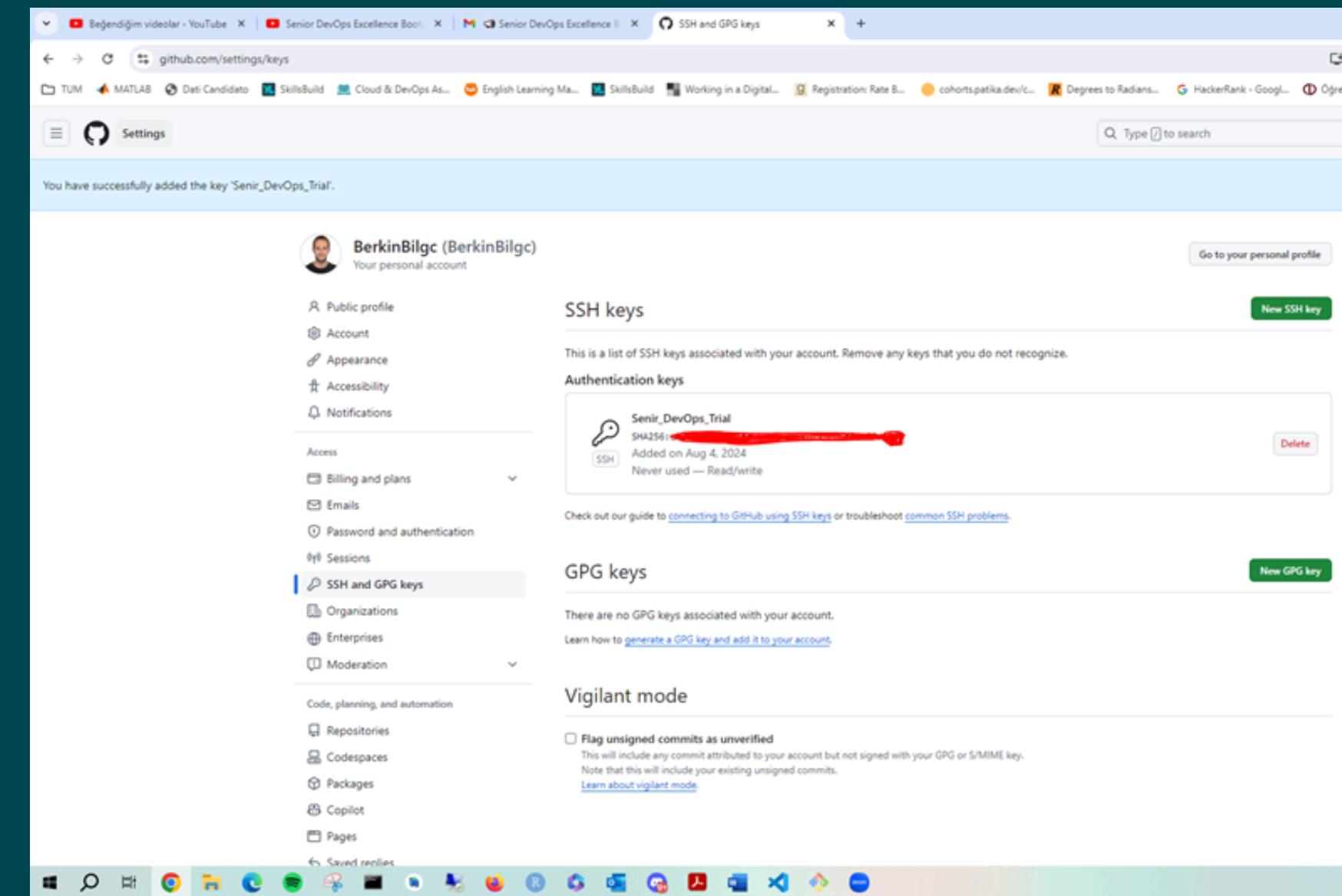
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps (main)
$ |
```

**5.adım: Başka bir github repository açalım ve bu sefer derste öğrendiğimiz SSH-KEY ile github veri gönderme yapalım**  
**Linux komutları dizin adı "devops" ve dosya adı "jenkins.txt" oluşturalım ve "DevOps öğreniyorum" yazalım.**

```
MINGW64/c/Users/Lenovo/Desktop/DevOps/devops2
Lenovo@DESKTOP-HJ8QH0G MINGW64 ~/Desktop/DevOps (main)
$ mkdir devops2
Lenovo@DESKTOP-HJ8QH0G MINGW64 ~/Desktop/DevOps (main)
$ cd devops2
Lenovo@DESKTOP-HJ8QH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ cat >> jenkins.txt
DevOps Öğreniyorum

Lenovo@DESKTOP-HJ8QH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ |
```

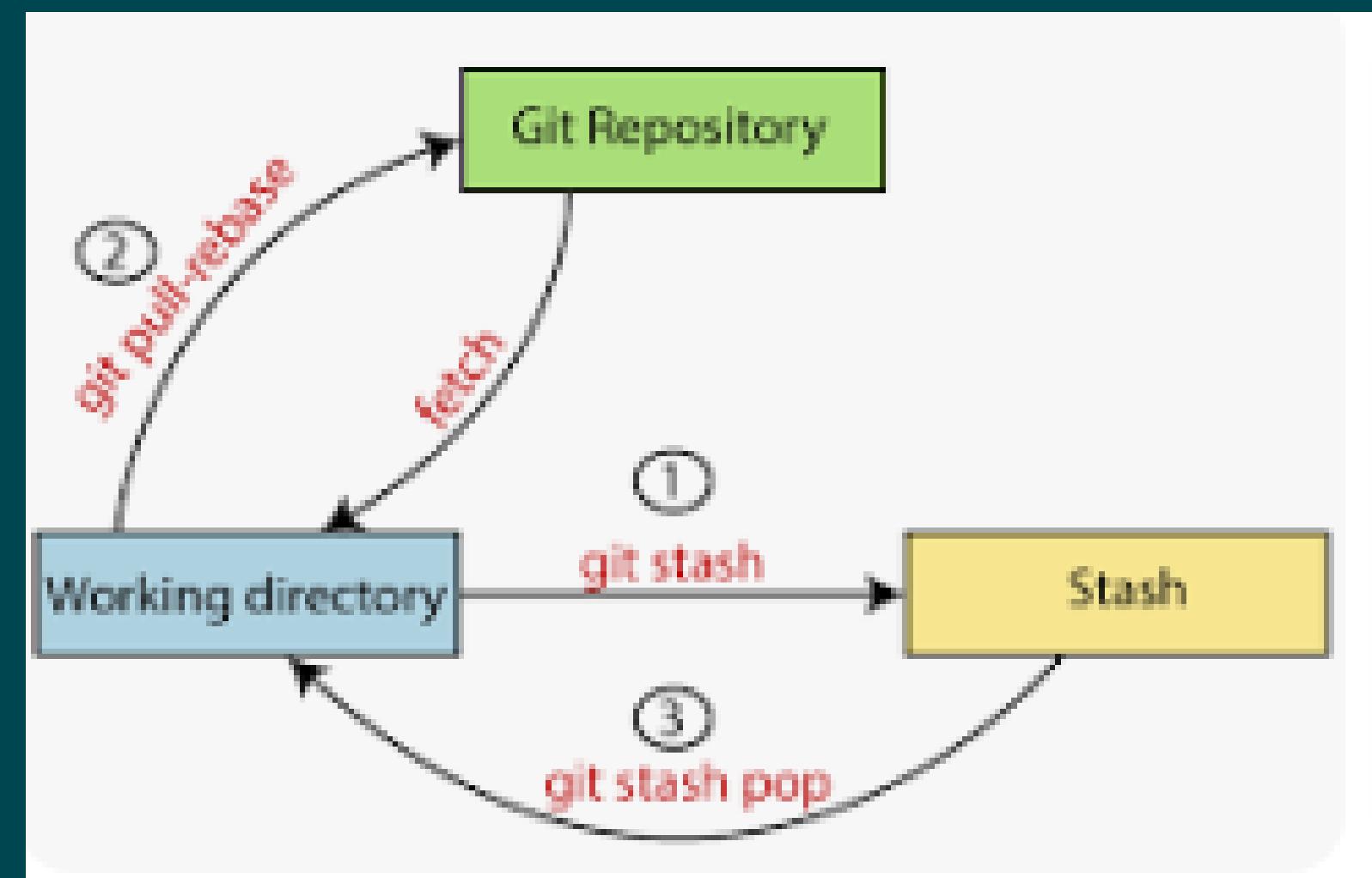
```
MINGW64/c/Users/Lenovo/Desktop/DevOps/devops2
Lenovo@DESKTOP-HJ8QH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ ssh-keygen -t rsa -b 4096 -C "berknbilgc@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Lenovo/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Lenovo/.ssh/id_rsa.
Your public key has been saved in /c/Users/Lenovo/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:VrJF61JEU-Ram1Sqqko3z96mB/VMUUSPYUIbLMZj9I berknbilgc@gmail.com
The key's randomart image is:
+---[RSA 4096]---+
|          o .#o. |
|         . E+++=+ |
|        o+B=.o.   |
|       .o# B     |
|      . S O     |
|     . . . o    |
|    = +o o     |
|   .O+.o.     |
+---[SHA256]----+
```



The screenshot shows a web browser window with the URL `github.com/settings/keys`. The user profile is **BerkinBilgc (BerkinBilgc)**. On the left, there is a sidebar with links like Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and plans, Emails, Password and authentication, Sessions, SSH and GPG keys (which is selected), Organizations, Enterprises, Moderation, Code, planning, and automation, Repositories, Codespaces, Packages, Copilot, Pages, and Saved replies. The main content area is titled "SSH keys" and contains a list with one item: "Senir\_DevOps\_Trial" (SHA256: VrJF61JEU-Ram1Sqqko3z96mB/VMUUSPYUIbLMZj9I). Below the list, it says "Never used — Read/write". There is also a "Delete" button next to the key entry. A note at the bottom says "Check out our guide to [connecting to GitHub using SSH keys](#) or troubleshoot [common SSH problems](#)".

## 16.adım: git stash nedir ?

Git stash, çalışma dizinindeki değişiklikleri (izlenmeyen dosyalar hariç) saklayarak çalışma dizinini temiz hale getirir ve bu değişikliklerin daha sonra geri yüklenmesine olanak tanır. Bu, üzerinde çalıştığınız değişiklikleri kaydetmeden başka bir dal veya görevde geçmeniz gerekiğinde yararlıdır.



17.adım: projelerimizi pushlama yaparken acil.txt adında bir iş geldi ve bu iş öncelik olduğu söylendi var olan add yapılmış dosyalarımızı commitleme yapmadan özel bir alanda saklama yapalım tabi bunu git stash ile yapalım.  
stash adı araf olsun  
acil.txt isimizi bitirdik bunu pushladık  
stash araf adındaki stash çağrııp işleyip ve sonrasında silelim.

```
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git add .
warning: LF will be replaced by CRLF in devops2/jenkins.txt.
The file will have its original line endings in your working directory
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git commit -- save
bash: $: command not found
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ 
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ $ git commit -- save "araf"
bash: $: command not found
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash --save "araf"
error: unknown option 'save'
usage: git stash [push [-p|-patch] [-k|--no=keep-index] [-q|--quiet]
                 [-u|--include-untracked] [-a|--all] [-m|--message <message>
                 [--] [<pathspec>...]]
-k, --keep-index      keep index
-p, --patch           stash in patch mode
-q, --quiet           quiet mode
-u, --include-untracked
-a, --all             include untracked files in stash
-m, --message <message>
--pathspec-from-file <file>
--pathspec-file-nul   read pathspec from file
--pathspec-elements  with --pathspec-from-file, pathspec elements are separated with NUL character
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash list
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash apply araf
error: araf is not a valid reference
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash apply "araf"
error: araf is not a valid reference
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git add .
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git push -u origin main
To https://github.com/BerkinBilg/Senior_Develop_Tech.git
 ! [rejected]      main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/BerkinBilg/Senior_Develop_Tech.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash drop araf
error: araf is not a valid reference
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ |
```

```
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git add .
warning: LF will be replaced by CRLF in devops2/jenkins.txt.
The file will have its original line endings in your working directory
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git commit -- save
bash: $: command not found
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ 
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ $ git commit -- save "araf"
bash: $: command not found
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash --save "araf"
error: unknown option 'save'
usage: git stash [push [-p|-patch] [-k|--no=keep-index] [-q|--quiet]
                 [-u|--include-untracked] [-a|--all] [-m|--message <message>
                 [--] [<pathspec>...]]
-k, --keep-index      keep index
-p, --patch           stash in patch mode
-q, --quiet           quiet mode
-u, --include-untracked
-a, --all             include untracked files in stash
-m, --message <message>
--pathspec-from-file <file>
--pathspec-file-nul   read pathspec from file
--pathspec-elements  with --pathspec-from-file, pathspec elements are separated with NUL character
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash list
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash apply araf
error: araf is not a valid reference
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git stash apply "araf"
error: araf is not a valid reference
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git add .
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git push -u origin main
To https://github.com/BerkinBilg/Senior_Develop_Tech.git
 ! [rejected]      main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/BerkinBilg/Senior_Develop_Tech.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
Lenovo@DESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ |
```

18.adım: git log --all --oneline --decorate --graph komutunu graph adından alias kullanarak kısaltalım ve

A screenshot of a Windows desktop environment. In the center, there is a terminal window titled 'MINGW64/c/Users/Lenovo/Desktop/DevOps/devops2' showing the command 'git log --all --oneline --decorate --graph'. The output lists several commits, including a stash commit, merge commits between 'frontend' and 'backend', and a second commit from 'origin/main'. Below the terminal is a system tray with various icons. A notification bubble from 'Konusan: Master Seni...' is visible in the top right corner.

```
MINGW64/c/Users/Lenovo/Desktop/DevOps/devops2
LenovoDESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git config --global alias.graph "log --all --graph --decorate --oneline"
LenovoDESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ git graph
* 1bbf8d6 (HEAD -> main) stash sonrası commit
* 801ddaa (frontend) merge öncesi commit 2
* 3693348 (backend) backend
* c4acc75 mesajı değiştir
| * 9f0da6c (origin/main) second commit
|/
* 1855b7f first commit
LenovoDESKTOP-HJBQH0G MINGW64 ~/Desktop/DevOps/devops2 (main)
$ | Konusan: Master Seni...
```

19.adım: Rebesa ile merge arasındaki fark nedir ?

Rebase, bir dalın tabanını başka bir dalın sonuna taşıyarak tarihçeyi yeniden yazarken, merge, iki dalı birleştirip tüm değişiklikleri koruyarak yeni bir commit oluşturur. Rebase daha temiz bir tarihçe sağlar, merge ise tüm değişikliklerin kaydını tutar.

## 20.adım: Git Conflict nedir ? Bir conflict yediğimizde ne yapmamız gerekiyor ?

Git conflict (çatışma), aynı dosyanın aynı satırlarında farklı değişiklikler yapıldığında ortaya çıkar ve Git bu değişiklikleri otomatik olarak birleştiremez. Bir conflict ile karşılaşığınızda, dosyaları manuel olarak düzenleyerek hangi değişikliklerin korunacağını belirtmeli ve ardından çatışmaları çözdüğünüzü belirten bir commit yapmalısınız.

## 21.adım: git ignore nedir ?

Git tarafından takip edilmesini istemediğiniz dosya ve dizinleri belirtmek için kullanılan bir dosyadır. Bu dosya, belirli desenlere uyan dosyaların ve dizinlerin Git'in sürüm kontrolünden hariç tutulmasını sağlar.

## 22.adım: git tag v1.1 ? Bu komu ne iş yapar ?

git tag v1.1 komutu, mevcut commit'e v1.1 adında bir etiket (tag) oluşturur. Etiketler genellikle belirli bir sürümü veya önemli bir aşamayı işaretlemek için kullanılır, böylece bu commit'e kolayca referans verebilirsiniz.

## 23.adım: git diff 3b2f0ab 5a2b8de bu komu ne iş yapar ? NOT: 3b2f0ab 5a2b8de commit numarası

git diff 3b2f0ab 5a2b8de komutu ise iki commit arasındaki farkları gösterir. Bu komut, belirtilen commit'ler arasında yapılan değişiklikleri, hangi dosyaların değiştigini ve detayları görüntüler.

## 24.adım: DevOps kültür felsefesi nedir ?

### 2.adım: DevOps açılımı ?

#### DevOps kültür felsefesi nedir ?

DevOps kültür felsefesi, yazılım geliştirme (Dev) ve bilgi teknolojileri operasyonları (Ops) arasındaki işbirliğini ve iletişimini artırarak daha hızlı ve güvenilir yazılım teslimi sağlamayı hedefler. Bu felsefe, sürekli entegrasyon, sürekli teslim (CI/CD), otomasyon ve geri bildirim döngüleri gibi uygulamalarla süreçleri iyileştirir ve organizasyonlar arasında daha güçlü bir işbirliği oluşturur.

- DevOps, "Development" (Geliştirme) ve "Operations" (İşletme) kelimelerinin birleşimidir.

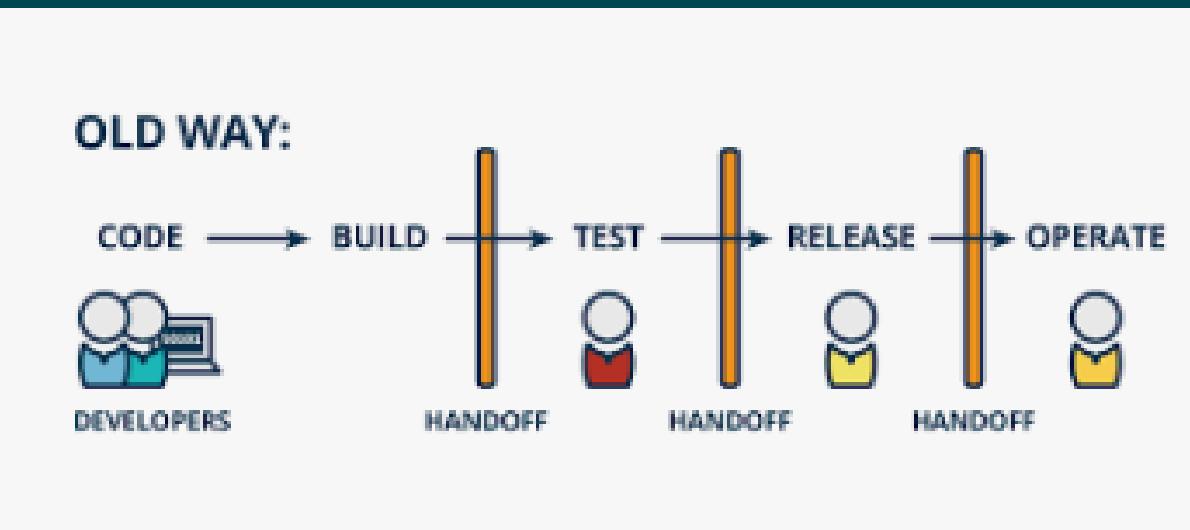
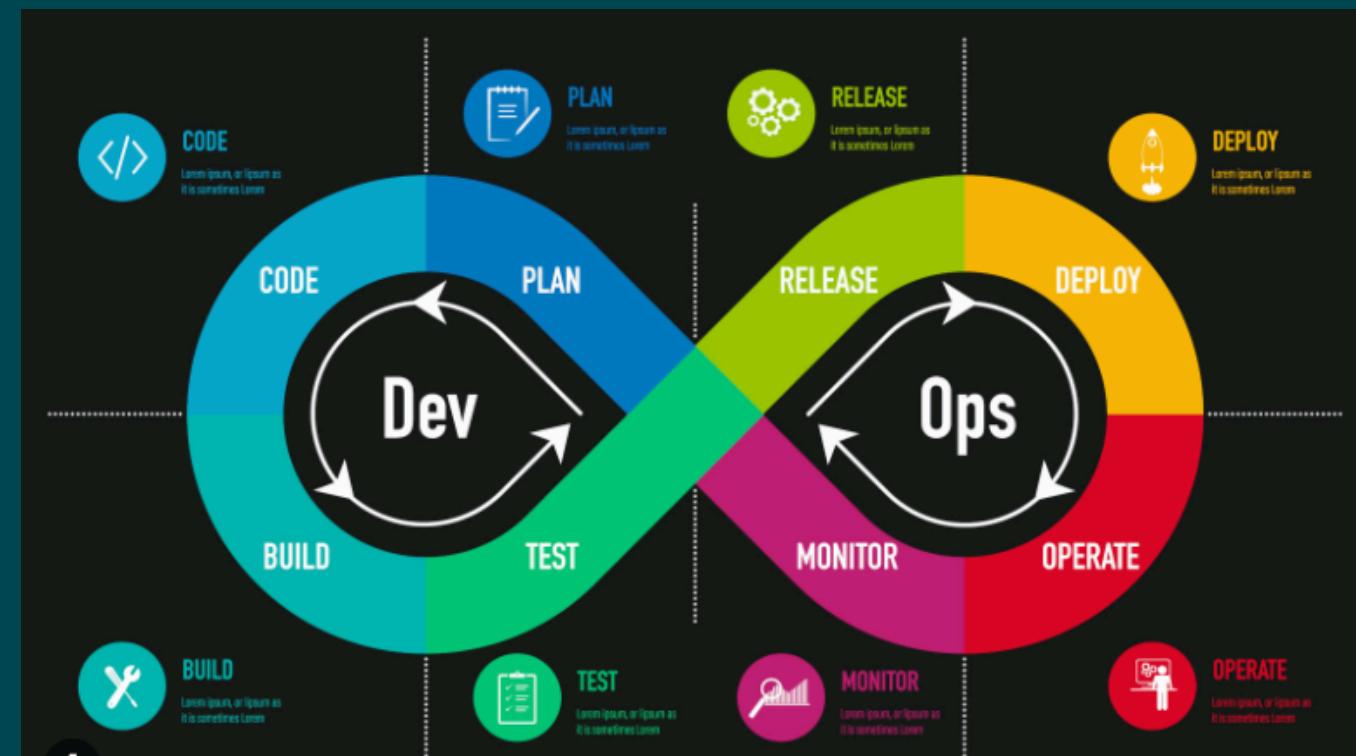
## 26.adım: DevOps Yöntemlerinden Continous /Continous Delivery-Deployment süreçler hakkında bildilerimizi yazalım

**Continuous Delivery (CD):** Yazılımın her değişikliğini otomatik olarak test eder ve üretim ortamına sorunsuz bir şekilde taşıyacak şekilde hazırlar, ancak üretime geçiş manuel onay gerektirebilir.

**Continuous Deployment (CD):** Yazılım değişikliklerini otomatik olarak test eder ve doğrudan üretim ortamına uygular, yani her değişiklik otomatik olarak canlıya alınır.

## 27.adım: DevOps Yöntemlerinden git nedir?

Git, dağıtık bir versiyon kontrol sistemidir. Yazılım geliştirme projelerindeki dosya değişikliklerini takip eder, farklı sürümler arasında geçiş yapmayı sağlar ve ekip üyeleri arasında işbirliği yapmayı kolaylaştırır. Git, her geliştiricinin kendi yerel deposunda tüm proje geçmişine erişim sağlar ve değişiklikleri merkezi bir depoya senkronize eder.



## 28.adım: DevOps Yöntemlerinden Agile nedir?

Agile, yazılım geliştirme sürecinde esneklik, işbirliği ve sürekli iyileştirmeye odaklanan bir yaklaşımdır. Küçük, işlevsel parçalar halinde yazılım geliştirir, sık sık geri bildirim alır ve müşteri ihtiyaçlarına hızlı bir şekilde adapte olur. Agile, genellikle Scrum veya Kanban gibi çerçevelerle uygulanır.

## 29.adım: DevOps Continues monitoring nedir ?

**Continuous Monitoring (Sürekli İzleme),** yazılım uygulamalarının ve sistemlerin sürekli olarak izlenmesi ve performans, güvenlik ve sağlık durumlarının anlık olarak takip edilmesini sağlar. Bu yöntem, sorunları hızla tespit etmeyi ve proaktif önlemler almayı mümkün kılar, böylece sistemlerin güvenilirliği ve kullanıcı deneyimi artırılır.

### 30.adım:Aşağıdaki içeriklerin açıklımları ve ne olduğunu yazalım.?

**XML nedir?    JSON nedir?    Yaml nedir?    http nedir?    server nedir?    JavaJDK    JavaJRE    JVM nedir?**

**XML (eXtensible Markup Language):** Verileri yapısal bir biçimde temsil eden, insan tarafından okunabilir ve makine tarafından işlenebilir bir işaretleme dilidir. XML, veri tanımlama ve taşıma amacıyla kullanılır ve hiyerarşik veri yapıları oluşturur.

**JSON (JavaScript Object Notation):** Verileri hafif bir biçimde temsil eden, insan tarafından okunabilir ve makine tarafından işlenebilir bir veri formatıdır. JSON, anahtar-değer çiftleri ve diziler kullanarak veri yapılarını tanımlar ve genellikle API'lar ve veri alışverişi için tercih edilir.

**YAML (YAML Ain't Markup Language):** Verileri okunabilir bir biçimde temsil eden, genellikle yapılandırma dosyaları için kullanılan bir veri serileştirme formatıdır. YAML, hiyerarşik veri yapıları ve veri türleri ile çalışmak için kullanılır ve JSON'a benzer, ancak daha okunabilir bir sözdizimine sahiptir.

**HTTP (Hypertext Transfer Protocol):** Web üzerinde veri iletimi için kullanılan bir protokoldür. HTTP, istemciler (tarayıcılar) ile sunucular arasında veri iletimi sağlar ve web sayfalarının yüklenmesi ve etkileşimli uygulamaların çalışması için temel bir protokoldür.

**Server (Sunucu):** Ağ üzerinden hizmet sağlayan bir bilgisayar veya yazılım uygulamasıdır. Sunucular, veri depolama, uygulama yönetimi veya web içeriği gibi çeşitli hizmetleri kullanıcılar veya diğer bilgisayarlara sunar.

**Java JDK (Java Development Kit):** Java programlama dili kullanarak uygulama geliştirmek için gerekli araçları ve kütüphaneleri sağlayan bir yazılım geliştirme kitidir. JDK, Java derleyici, JVM ve diğer araçları içerir.

**Java JRE (Java Runtime Environment):** Java uygulamalarını çalıştırmak için gerekli olan çalışma ortamıdır. JRE, JVM'yi (Java Virtual Machine) ve gerekli kütüphaneleri içerir, ancak geliştirme araçları sağlamaz.

**Java JVM (Java Virtual Machine):** Java bytecode'larını çalıştırın sanal bir makinedir. JVM, Java uygulamalarını bağımsız bir ortamda çalıştırır ve farklı işletim sistemlerinde aynı Java programının çalışmasını sağlar.

31.adım:Maven nedir?

mavende→cleaninstall görevi nedir?

Maven, Java projeleri için bir yapı ve proje yönetim aracıdır. Maven, proje yapılandırmasını ve bağımlılıkları yönetmeyi kolaylaştırır, standart bir yapı ve yapılandırma sunar, ve projelerin derlenmesi, test edilmesi ve paketlenmesi gibi işlemleri otomatikleştirir.

`mvn clean install` komutu Maven'de iki temel görevi yerine getirir:

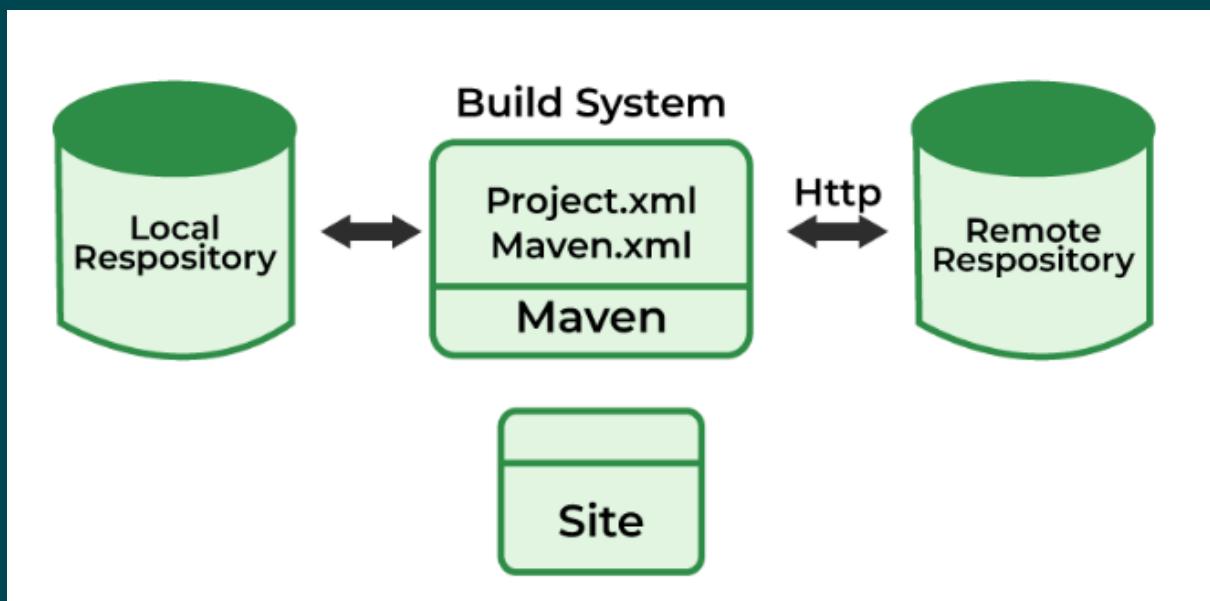
- clean: Projenin önceki derleme çıktıları ve geçici dosyaları temizler.
- install: Projeyi derler, test eder ve sonuçları yerel Maven deposuna (local repository) kurar, böylece diğer projelerde kullanılabilir hale gelir.

Bu komut, genellikle proje üzerinde yapılacak değişiklıkların tüm bileşenlerini sıfırdan oluşturmak ve test etmek için kullanılır.

`./mvncleanpackage-DskipTest=>` Bu komut nedir?

`./mvn clean package -DskipTests` komutu Maven'de iki ana işlevi yerine getirir:

- clean: Projeyi önceki derleme çıktılarından ve geçici dosyalardan temizler.
- package: Projeyi derler ve sonucunda bir JAR veya WAR dosyası gibi paketlenmiş bir dağıtım birimi oluşturur.
- -DskipTests: Test aşamasını atlar, yani testler çalıştırılmaz.
- Bu komut, testleri atlayarak sadece projeyi derleyip paketlemek için kullanılır.

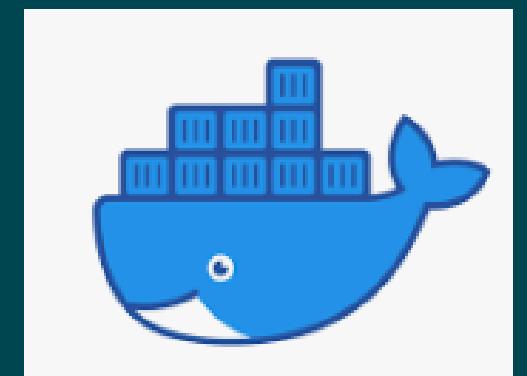
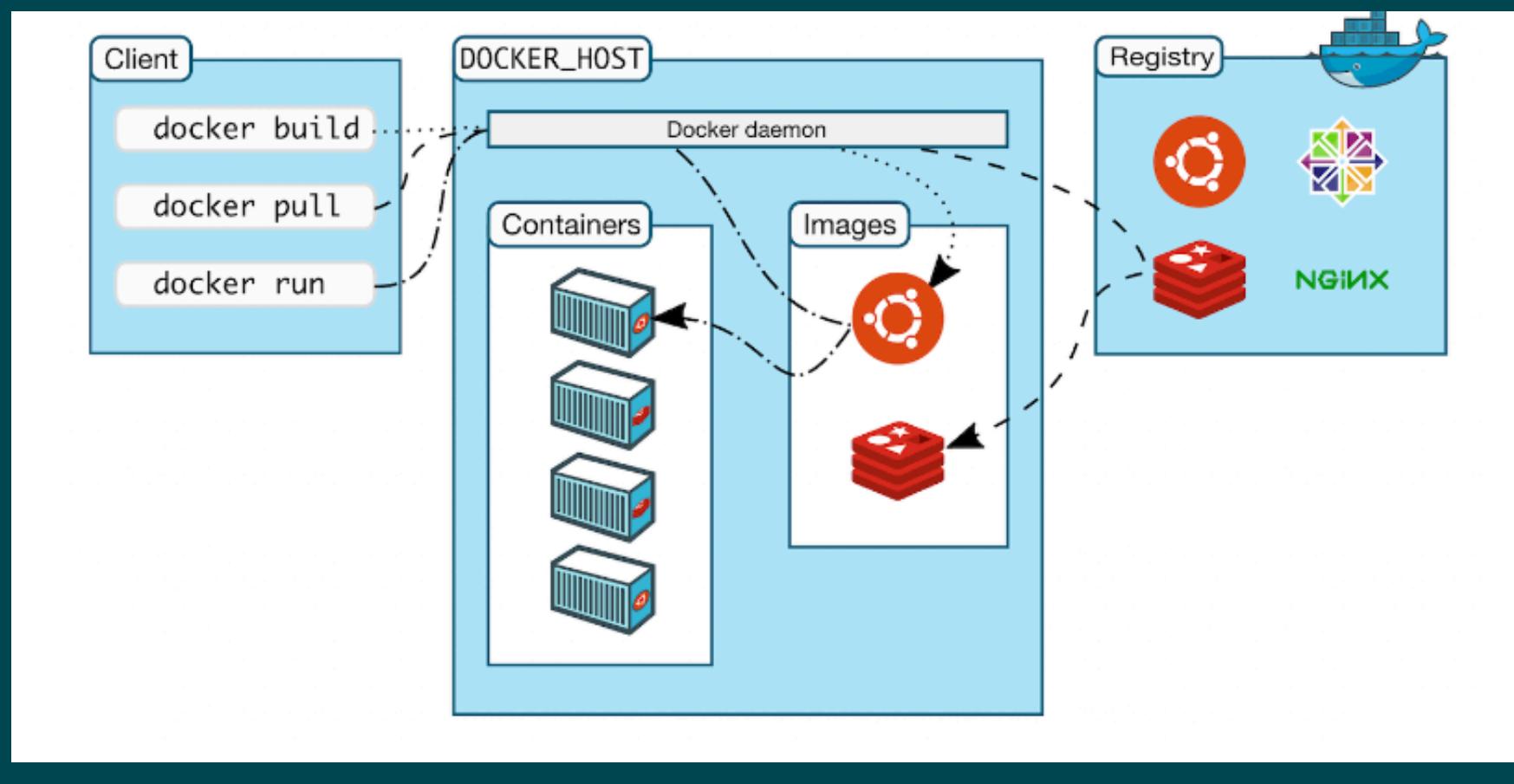


## 32.adım: Docker nedir ? Docker Daemon, Docker CLI ne iş yapıyor ?

Docker, uygulamaları ve bağımlılıklarını izole bir şekilde konteynerlerde paketleyip dağıtmayı sağlayan bir platformdur. Konteynerler, uygulamaların ve tüm gerekli bileşenlerin bağımsız bir ortamda çalışmasını sağlar, bu da taşınabilirlik ve tutarlılık sunar.

**Docker Daemon (dockerd)**, Docker konteynerlerinin oluşturulması, çalıştırılması ve yönetilmesinden sorumlu arka planda çalışan bir sunucudur. Daemon, Docker komutlarını alır, bu komutları işler ve konteynerleri yönetir.

**Docker CLI (Command Line Interface)**, Docker ile etkileşimde bulunmak için kullanılan komut satırı aracıdır. CLI, Docker Daemon'a komutlar gönderir ve konteyner oluşturma, başlatma, durdurma gibi işlemleri kullanıcıya sağlar.



33.adım: Aşağıdaki adımları teker teker yazalım?

34.adım: -docker search nginx

```
Komut İstemi
Microsoft Windows [Version 10.0.19045.4651]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Lenovo>docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
application.jar  latest   f0a34a36c2eb  19 months ago  498MB

C:\Users\Lenovo>docker search nginx
NAME                  DESCRIPTION          STARS      OFFICIAL      AUTOMATED
nginx                Official build of Nginx.    20057     [OK]           []
bitnami/nginx         Bitnami container image for NGINX       193       [OK]
nginxinc/nginx-unprivileged  Unprivileged NGINX Dockerfiles    155
nginx/nginx-ingress   NGINX and NGINX Plus Ingress Controllers for...  92
bitnami/wordpress-nginx  Bitnami container image for WordPress with NL...  91       [OK]
nginx/unit             This repository is retired, use the Docker o...  63
bitnami/nginx-ingress-controller  Bitnami container image for NGINX Ingress Co...  34       [OK]
unit                  Official build of NGINX Unit: Universal Web ...  33       [OK]
rancher/nginx-ingress-controller  13
bitnami/drupal-nginx   DEPRECATED Bitnami Docker Image for Drupa...  9       [OK]
kasmweb/nginx          An Nginx image based off nginx:alpine and in...  8
dynatrace/easytravel-nginx  The Reverse-proxy component (NGINX) of the D...  6
nginxinc/nginx-s3-gateway  Authenticating and caching gateway based on ...  6
bitnami/nginx-exporter  Bitnami container image for NGINX Exporter    5
nginx/nginx-ingress-operator  NGINX Ingress Operator for NGINX and NGINX P...  2
rancher/nginx          2
vmware/nginx-photon   1
rancher/k3d-proxy      NGINX configured via confd to proxy requests...  1
paketobuildpacks/nginx  0
bitnamicharts/nginx    0
rancher/library-nginx   0
mirantis/dtr-nginx     0
docker/dtr-nginx       0
rancher/mirrored-library-nginx  0
droidwiki/nginx         0

C:\Users\Lenovo>
```

35.adım: -docker pull nginx

36.adım: -nginx yandaki özelliklere sahip containerlar özelliklerine göre yazalım.

```
C:\Users\Lenovo>docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
efc2b5ad9eec: Pull complete
8fe9a5eb80f: Pull complete
045037a63be8: Pull complete
7111b42b4bfa: Pull complete
3dfc528a4df9: Pull complete
9e891cdb453b: Pull complete
0f11e17345c5: Pull complete
Digest: sha256:6af79ae5de407283dcea8b00d5c37ace95441fd58a8b1d2aa1ed93f5511bb18c
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest

C:\Users\Lenovo>
```

```
C:\Users\Lenovo>docker run --name web_2 -p 6666:80 --rm -d nginx:latest
e1658e02570427aab7d07592b5d82085ec198b359e5f59cdbcbc582187796117

C:\Users\Lenovo>
```

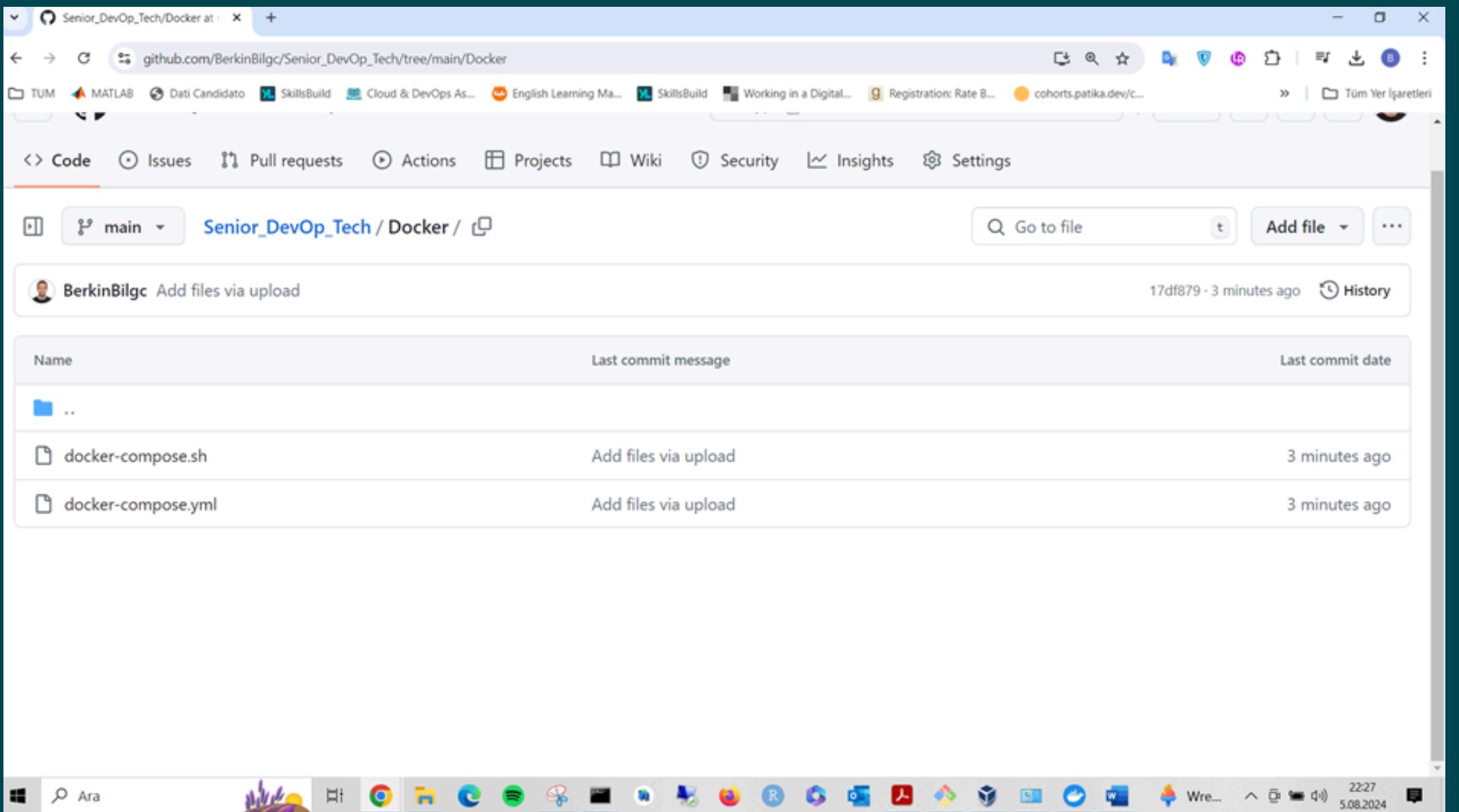
37.adım: bu github adresindeki veriyi github ile clone yapalım. ve yapılacakları aşağıda yazılmıştır.

-git clone URL\_ADDRESS

-bu JAR dosyasının Dockerfile ve docker-compose.yml yazarak image oluşturalım.

-Bu oluşturduğunuz image kendi dockerHub'ta repository'a pushlayalım.

-Kendi repository gönderdiğiniz bu image docker pull ... diyerek tekrardan local bilgisayarına veriyi alalım



#### 38. adım SonarQube nedir ?

SonarQube, yazılım projelerinde kod kalitesini ve güvenliğini analiz eden bir araçtır. Bu platform, statik kod analizi yaparak hataları, güvenlik açıklarını ve kod kokularını tespit eder. Geliştiricilere, kod kalitesini iyileştirmeleri ve yazılım projelerini daha sürdürülebilir hale getirmeleri için ayrıntılı raporlar sunar.

#### 41.adım: -GitLab nedir ?

GitLab, yazılım geliştirme projelerini yönetmek için kullanılan, Git tabanlı bir sürüm kontrol platformudur. GitLab, proje yönetimi, sürekli entegrasyon (CI) ve sürekli teslim (CD) gibi özellikleri tek bir platformda sunarak yazılım geliştirme sürecini hızlandırır ve kolaylaştırır. Kullanıcılar, depolarını barındırabilir, kod incelemeleri yapabilir ve iş birliği içinde çalışabilirler.

#### 42.adım: -GitLab nedir CI/CD nedir ?

GitLab, yazılım geliştirme projelerini yönetmek için kullanılan, Git tabanlı bir sürüm kontrol platformudur. Kullanıcılar, projelerini barındırabilir, kod incelemeleri yapabilir ve iş birliği yapabilirler. GitLab, proje yönetimi, sürekli entegrasyon (CI) ve sürekli teslim (CD) gibi özellikleri tek bir platformda sunarak yazılım geliştirme sürecini hızlandırır ve kolaylaştırır. Hem açık kaynaklı bir sürümü hem de kurumsal ihtiyaçlara yönelik ücretli sürümleri bulunmaktadır.

**Sürekli Entegrasyon (CI):** Yazılım geliştiricilerin kodlarını sık sık, genellikle günde birkaç kez, merkezi bir depoya entegre etmelerini sağlayan bir uygulamadır. Her entegrasyon, otomatik olarak derlenir ve test edilir, böylece hatalar erken tespit edilir ve düzeltilebilir.

**Sürekli Teslim (CD):** Koddaki değişikliklerin otomatik olarak test edilip, kullanıcıların kullanımına sunulacak şekilde dağıtılmasını sağlayan bir uygulamadır. Bu süreç, yazılımın her an dağıtıma hazır olmasını sağlar ve yazılım güncellemlerini daha hızlı ve güvenilir hale getirir.

CI/CD, yazılım geliştirme sürecini otomatikleştirerek, hataları erken tespit etmeyi ve yazılımın kalitesini artırmayı hedefler. GitLab, bu süreçleri entegre bir şekilde yönetmenize yardımcı olan araçlar ve özellikler sunar.

#### 43.adım: -GitLab Runner nedir ?

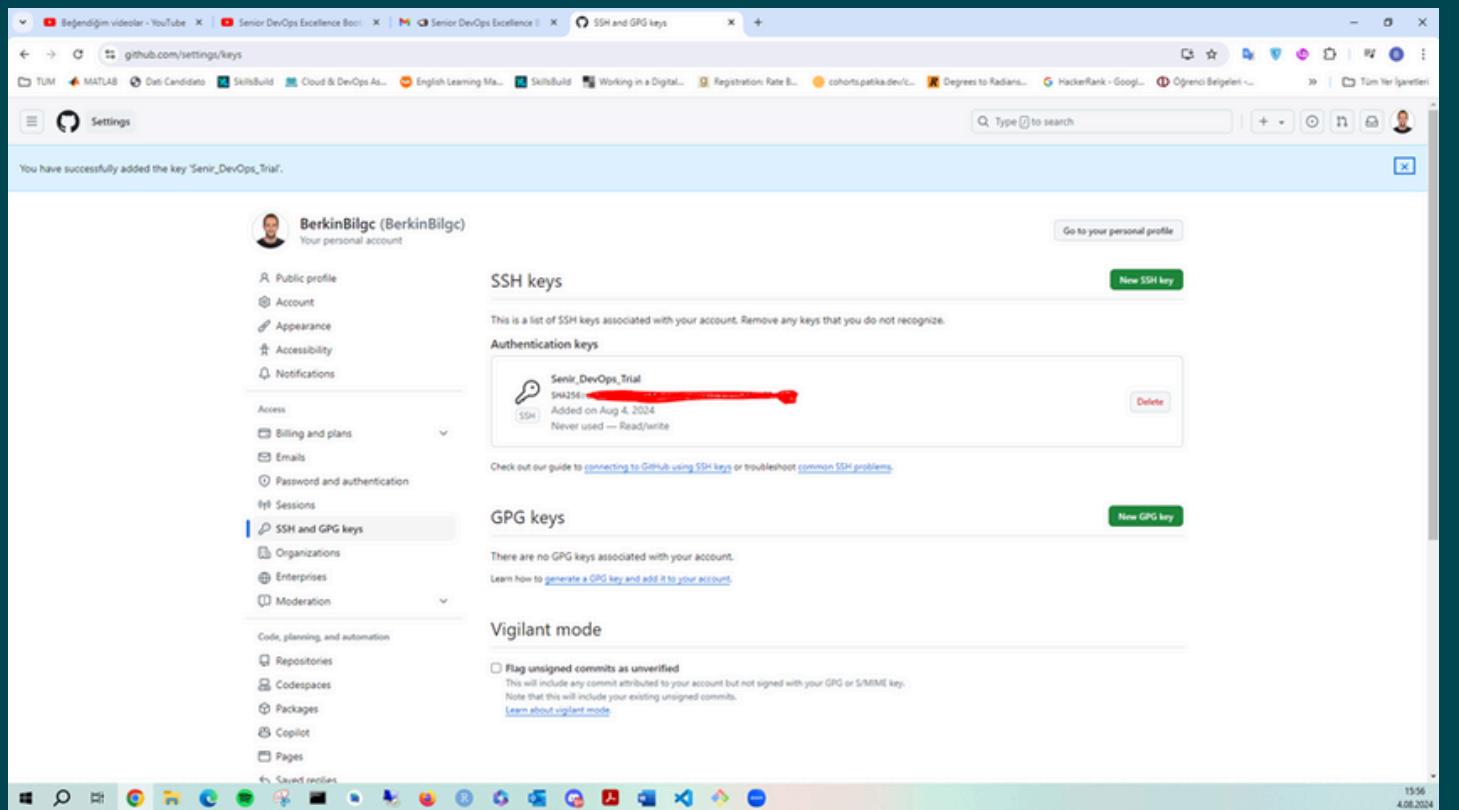
GitLab Runner, GitLab CI/CD için kullanılan ve GitLab sunucusundan gelen iş (job) taleplerini alıp çalıştırın bir uygulamadır. Bu araç, belirli bir ortamda (örneğin, bir sanal makine, konteyner veya fiziksel makine) kodunuzu derleyip test eder ve sonuçları GitLab'a geri gönderir. CI/CD süreçlerini otomatikleştirerek iş akışlarını hızlandırır ve verimliliği artırır.

#### 44.adım: -SSH nedir ?

SSH (Secure Shell), ağ üzerinden güvenli iletişim sağlamak için kullanılan bir protokoldür. Genellikle uzaktaki bir sunucuya güvenli bir şekilde bağlanmak ve komutlar çalıştırmak için kullanılır. SSH, veri şifreleme, kimlik doğrulama ve veri bütünlüğü sağlar.

#### 45.adım: -GitLab ve GitHub SSH-keygen oluşturarak bilgisayarına bağlayın?

GitLab, yazılım geliştirme projelerini yönetmek için kullanılan, Git tabanlı bir sürüm kontrol platformudur. GitLab, proje yönetimi, sürekli entegrasyon (CI) ve sürekli teslim (CD) gibi özellikleri tek bir platformda sunarak yazılım geliştirme sürecini hızlandırır ve kolaylaştırır. Kullanıcılar, depolarını barındırabilir, kod incelemeleri yapabilir ve iş birliği içinde çalışabilirler.



#### 46.adım: -Prometheus nedir, Jenkins nedir ?

Prometheus, açık kaynaklı bir izleme ve uyarı sistemidir. Zaman serisi verilerini toplar, saklar ve sorgular, bu verileri görselleştirir ve uyarılar oluşturur. Özellikle bulut yerel uygulamaların izlenmesinde yaygın olarak kullanılır.

Jenkins, açık kaynaklı bir otomasyon sunucusudur. Sürekli entegrasyon ve sürekli teslim (CI/CD) süreçlerini otomatikleştirmek için kullanılır. Jenkins, eklentiler aracılığıyla çeşitli yazılım projelerini derleyebilir, test edebilir ve dağıtabilir.

#### 47.adım: -Ansible nedir ?

Ansible, açık kaynaklı bir otomasyon aracıdır ve IT altyapısını yönetmek için kullanılır. Sunucuların yapılandırmasını, yazılım kurulumlarını ve uygulama dağıtımlarını otomatikleştirmek amacıyla kullanılır. Kod olarak tanımlı (infrastructure as code) yapılandırma ve yönetim sağlar ve genellikle YAML dilinde yazılmış "playbook" adı verilen betikler kullanır.

#### 48.adım: -Kubernetes nedir ?

Kubernetes, açık kaynaklı bir konteyner orkestrasyon platformudur. Konteyner tabanlı uygulamaları otomatik olarak dağıtmak, ölçeklemek ve yönetmek için kullanılır. Yük dengeleme, otomatik ölçekleme ve hata toleransı gibi özellikler sunar. Kubernetes, konteynerlerin yönetimini merkezi bir kontrol noktası üzerinden sağlar.

#### 49.adım: -Datadog Monitoring nedir ?

Datadog Monitoring, bulut altyapıları, uygulamalar ve hizmetler için entegre bir izleme ve analitik platformudur. Performans metriklerini, logları ve izleme verilerini toplar ve analiz eder, böylece sistemlerin ve uygulamaların sağlığını ve performansını gerçek zamanlı olarak takip eder. Anomali tespiti, uyarı yönetimi ve performans optimizasyonu sağlar.

#### 50.adım: -Kanban nedir ?

Kanban, iş süreçlerini ve projeleri görselleştirmek ve yönetmek için kullanılan bir yöntemdir. Görsel kartlar ve panolar kullanarak iş akışını, görevlerin durumunu ve ilerlemesini takip eder. Bu yöntem, iş süreçlerini optimize etmeye, verimliliği artırmaya ve darboğazları tespit etmeye yardımcı olur.

#### 51.adım: -Scrum, Kanban ve waterfall nedir ?

Scrum, yazılım geliştirme ve proje yönetimi için kullanılan bir çevik (agile) yöntemdir. Kapsamlı bir proje, kısa süreli döngüler (sprintler) içinde yönetilir ve her döngü sonunda işleyen bir ürün parçası teslim edilir. Scrum, belirli roller (Scrum Master, Product Owner, Development Team) ve ritüeller (Daily Stand-up, Sprint Review, Sprint Retrospective) içerir.

Kanban, iş süreçlerini ve projeleri görselleştirmek ve yönetmek için kullanılan bir yöntemdir. İş akışını görselleştiren panolar ve kartlar kullanarak görevlerin durumunu takip eder ve işin akışını optimize eder. Kanban, iş miktarını dengelemeyi ve sürecin verimliliğini artırmayı hedefler.

Waterfall (Şelale) modeli, yazılım geliştirme projelerinde kullanılan geleneksel bir yönetim yöntemidir. Proje, belirli aşamalarda (analiz, tasarım, geliştirme, test, dağıtım) sırasıyla ilerler ve her aşama tamamlanmadan bir sonraki aşamaya geçmez. Bu model, aşamaların birbirini takip ettiği lineer bir yaklaşımı benimser.

# Berkin Bilgiç



berknbilgc@gmail.com



<https://github.com/BerkinBilgc>



[BerkinBilgic](#)