

# Git, GitHub və GitLab: Ətraflı Məlumat

## 1. Git nədir və niyə istifadə olunur?

Git, dünyada ən populyar versiya idarəetmə sistemlərindən (VCS) biridir. Bu sistem proqramçıların kod üzərində birgə işləməsini, dəyişiklikləri izləməsini və əvvəlki versiyalara qayıtmasını təmin edir.

### Git-i fərqləndirən əsas xüsusiyyətlər:

- ✓ **Paylanmış sistem:** Hər bir istifadəçi Git deposunun (repository) tam nüsxəsini lokalda saxlayır.
- ✓ **Tez və effektiv:** Git milyonlarla sətir kodu olan layihələrdə belə sürətli işləyir.
- ✓ **Dəyişiklik tarixçəsi:** Hər commit (dəyişiklik) tarixçədə saxlanır və lazım olduqda əvvəlki versiyalara qayıtmaq mümkündür.
- ✓ **Kodun təhlükəsizliyi:** Git, SHA-1 (Secure Hash Algorithm 1) kriptografik alqoritmindən istifadə edərək məlumatları qoruyur.

---

## 2. Git-in əsas anlayışları

### 2.1 Repository (Depo) nədir?

Git deposu (repository) layihənin bütün tarixçəsini saxlayan yerdir. İki növ depo mövcuddur:

- **Lokal repo** – Kompüterdə saxlanılır.
- **Remote repo** – GitHub, GitLab və ya başqa serverdə saxlanılır.

### 2.2 Commit nədir?

**Commit** – Dəyişiklikləri qeydiyyatla almaq üçün istifadə olunan əməliyyatdır. Hər commit unikaldir və geri qayıtmağa imkan yaradır.

#### Əsas əmrlər:

```
sh
Kodu kopyala
git add . git commit -m "Dəyişiklik mesajı" git push origin main
```

### 2.3 Branch (Budaq) nədir?

**Branch** – Kodun fərqli versiyalarını yaratmağa və eyni anda bir neçə dəyişiklik üzərində işləməyə imkan verir.

- **main (master) branch** – Əsas kod budağıdır.
- **feature branches** – Yeni funksiyalar və ya düzəlişlər üçün istifadə olunur.

**Branch yaratmaq və keçid etmək:**

```
sh
Kodu kopyala
git branch yeni-budag git checkout yeni-budag
```

**Branch-ları birləşdirmək:**

```
sh
Kodu kopyala
git checkout main git merge yeni-budag
```

---

## 3. Git Əmrləri: Praktik İstifadə

### 3.1 Git deposu yaratmaq və ilkin ayarlar

```
sh
Kodu kopyala
git init # Yeni git deposu yarat git remote add origin <repo_URL> # Uzaq
(remote) depo əlavə et git add . # Bütün dəyişiklikləri izləməyə əlavə et git
commit -m "İlk commit" # Dəyişiklikləri təsdiqlə git push -u origin main #
Uzaq depoya göndər
```

### 3.2 Mövcud depodan kod çəkmək

```
sh
Kodu kopyala
git clone <repo_URL> # GitHub/GitLab-dan kodun surətini çıxar git pull origin
main # Ən son dəyişiklikləri çək
```

### 3.3 Git status və dəyişiklikləri izləmək

```
sh
Kodu kopyala
git status # Hansı faylların dəyişdiyini göstərir git log # Tarixçəni göstərir
```

### 3.4 Git conflict həlli

Kodun müxtəlif versiyaları üst-üstə düşəndə **merge conflict** yaranır. Həll etmək üçün:

```
sh
Kodu kopyala
git merge <branch_adi>
```

Daha sonra kodda konflikt olan hissələri düzəldib:

```
sh
Kodu kopyala
git add . git commit -m "Konflikt həll edildi"
```

---

## 4. GitHub və GitLab: Fərqlər və İstifadə Qaydaları

### 4.1 GitHub nədir?

**GitHub** Git repositoriyalarını idarə etmək üçün bulud əsaslı bir platformadır.

**Əsas funksiyalar:**

- ✓ **Kod hostingi** – GitHub açıq və özəl depolar üçün istifadə edilir.
- ✓ **Pull request-lər** – Kod dəyişiklərini əsas budağa əlavə etməyə imkan yaradır.
- ✓ **Issue Tracking** – Layihədə tapşırıqları və problemləri izləməyə kömək edir.
- ✓ **CI/CD** – GitHub Actions avtomatlaşdırmanı dəstəkləyir.

### 4.2 GitLab nədir?

**GitLab** da Git əsaslı kod hosting platformasıdır, lakin fərqli üstünlükləri var:

- ✓ **Öz serverində host etmək imkanı** – Şirkətlər üçün daha təhlükəsizdir.
- ✓ **Daxili CI/CD sistemi** – Test və deploy proseslərini avtomatlaşdırır.
- ✓ **Daha çox pullu xüsusiyyətlər təqdim edir.**

**Əsas fərqlər:**

Xüsusiyyət	GitHub	GitLab
Pull Requests	✓	✓ (Merge Request adlanır)
Öz serverində host etmək	✗	✓
CI/CD	GitHub Actions	Daxili CI/CD var
Pulsuz versiya	✓	✓

---

## 5. GitHub və GitLab-da Praktiki İş

## 5.1 GitHub-da yeni repo yaratmaq

1. **GitHub hesabına daxil ol.**
2. "New Repository" düyməsini sıx.
3. Repo adını yaz və "Create Repository" seç.

### Lokal repodan GitHub-a göndərmək:

sh

Kodu kopyala

```
git remote add origin <repo_URL> git push -u origin main
```

## 5.2 GitHub-da Fork və Pull Request nədir?

- **Fork** – Başqasının reposunu öz hesabına kopyalamaq.
- **Pull Request** – Yeni kodları əsas branch-a birləşdirmək üçün istifadə olunur.

### Pull Request yaratmaq:

1. Kodu dəyiş və commit et.
2. GitHub-da "Pull Request" aç.
3. Kod incələnilib təsdiqləndikdən sonra "Merge" et.

---

## 6. GitLab ilə İş Prosesi

### 6.1 GitLab-da yeni layihə yaratmaq

1. **GitLab hesabına daxil ol.**
2. "New Project" → "Create Blank Project" seç.
3. **Öz serverində host etmək mümkündür.**

### 6.2 Merge Request və CI/CD işlətmək

GitLab Merge Request GitHub-dakı Pull Request-ə bənzəyir.

CI/CD üçün `.gitlab-ci.yml` faylı yaradılır:

yml

Kodu kopyala

```
stages: - build - test - deploy test_job: stage: test script: - echo "Testlər icra edilir..."
```

Bu konfigurasiya kodun avtomatik test olunmasını təmin edir.

---

## 7. Git və GitHub/GitLab üzrə Ən Yaxşı Təcrübələr


- ✓ **Tez-tez commit edin** – Böyük dəyişikliklər yerinə, kiçik addımlarla işləmək daha yaxşıdır.
- ✓ **Branch sistemindən istifadə edin** – Yeni funksiyalar üçün ayrıca branch yaradın.
- ✓ **README.md və sənədləşdirmə əlavə edin** – Digər proqramçılar üçün layihəni anlaşıqlı edin.
- ✓ **Kod incələmə (Code Review) tətbiq edin** – Pull/Merge Request-lər vasitəsilə kod keyfiyyətini artırın.
- ✓ **Git Ignore istifadə edin** – Lazımsız faylları `git ignore` ilə izləməyə daxil etməyin.

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)



Required fields are marked with an asterisk (\*).

Owner \*      Repository name \*

 sabirsumqayit /

Great repository names are short and memorable. Need inspiration? How about **silver-computing-machine** ?

Description (optional)

- ☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**  
You choose who can see and commit to this repository.

Initialize this repository with:

- ☐ Add a README file  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: **None** ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **None** ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.

Create repository