



UNIVERSITY
OF CRETE

CS-100 – Git Tutorial

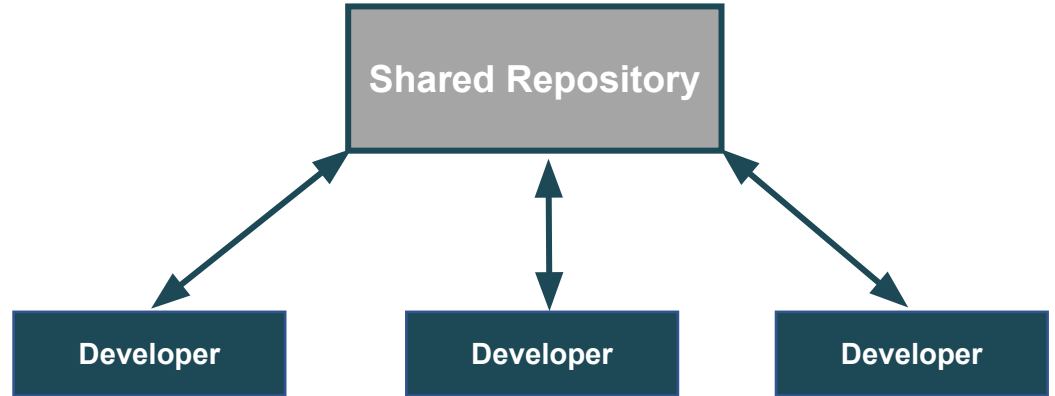
Iacovos G. Kolokasis
University of Crete
kolokasis@csd.uoc.gr

Outline

- What is git
- How to join CS100 group
- What are ssh keys and how to use them
- How to fork a project
- Make a project private and add TAs as members
- How to use private.py
- How to use git

What is Git?

- Version control system
- Track file history
- Shared among multiple users



How to Login to CSD Git?



- Go to csd gitlab: <https://gitlab-csd.datacenter.uoc.gr>

Computer Science Department, University of Crete



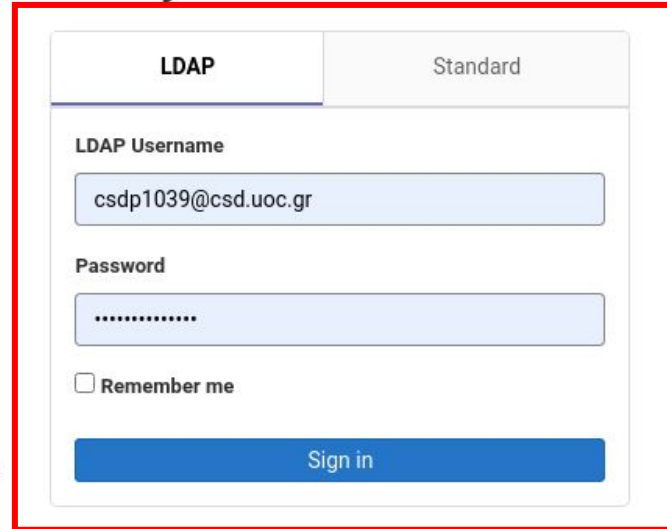
Pilot installation of GitLab, a web-based Git-repository manager.

GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

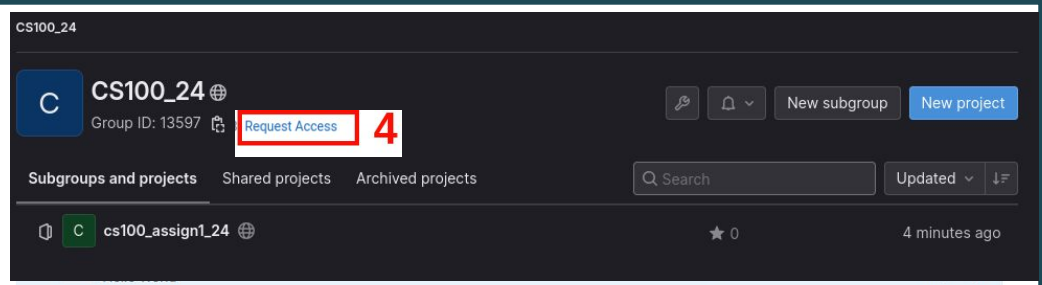
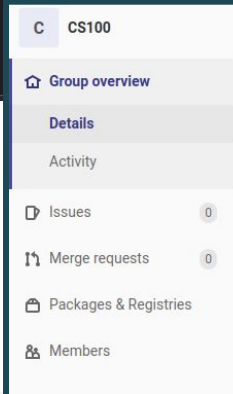
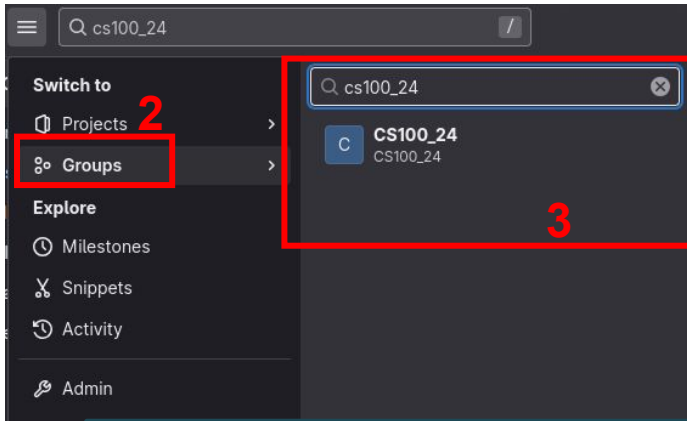
1



The screenshot shows the GitLab login interface. The 'LDAP' tab is selected, and the 'Standard' tab is visible. The form contains the following fields and options:

- LDAP Username:** A text input field containing the value 'csdp1039@csd.uoc.gr'.
- Password:** A password input field with masked characters '.....'.
- Remember me:** An unchecked checkbox.
- Sign in:** A blue button at the bottom of the form.

How to Join CS100 Group?



Generating ssh-keys

- Run: `ssh-keygen -t rsa -b 2048 -C "kolokasis@ics.forth.gr"`
- Press enter in the next three options
- Run: `cat ~/.ssh/id_rsa.pub`
- Copy the printed text

```
[root@desktop ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:vvDbwVF2YHAoSXCxnjhNojc4oEenLJCYgGcsGNyI2Gyw root@desktop.example.com
The key's randormart image is:
+---[RSA 2048]-----+
|B0o+ o.. .o+ |
|E=B.+ = . .o . |
|*X B . . o . |
|*+o o . . o . |
|B . . S . |
| o . . . |
| . . o |
| o o . |
| +.. |
+-----[SHA256]-----+
```

The diagram includes three callout boxes:

- A callout pointing to the two empty input lines: "Enter passphrase here, if required"
- A callout pointing to the file path `/root/.ssh/id_rsa.pub`: "Location of Public Key"
- A callout pointing to the file path `/root/.ssh/id_rsa`: "Location of Private Key"

Add ssh-keys in your Account

The screenshot shows the GitLab interface with the User Settings page open. A red box labeled '1' highlights the user profile icon in the top navigation bar. A second red box labeled '2' highlights the 'Preferences' option in the user settings dropdown menu. A third red box labeled '3' highlights the 'SSH Keys' option in the left-hand navigation sidebar.

User Settings > Preferences

Search settings

Navigation theme

Customize the appearance of the application header and navigation sidebar.

- Indigo
- Light Indigo
- Blue
- Light Blue
- Green
- Light Green
- Red
- Light Red
- Dark
- Light
- Dark Mode (alpha)

Syntax highlighting theme

This setting allows you to customize the appearance of the syntax. [Learn more.](#)

- White
- Dark
- Solarized Light

Add ssh-keys in your Account

User Settings > SSH Keys

Q Search settings

SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab.

Add an SSH key

To add an SSH key you need to [generate one](#) or use an [existing key](#).

Key

Paste your public SSH key, which is usually contained in the file '~/.ssh/id_ed25519.pub' or '~/.ssh/id_rsa.pub' and begins with 'ssh-ed25519' or 'ssh-rsa'. Do not paste your private SSH key, as that can compromise your identity.

Typically starts with "ssh-ed25519 ..." or "ssh-rsa ..."

Title

e.g. My MacBook key

Give your individual key a title. This will be publicly visible.

Expires at

mm/dd/yyyy

Key can still be used after expiration.

Add key

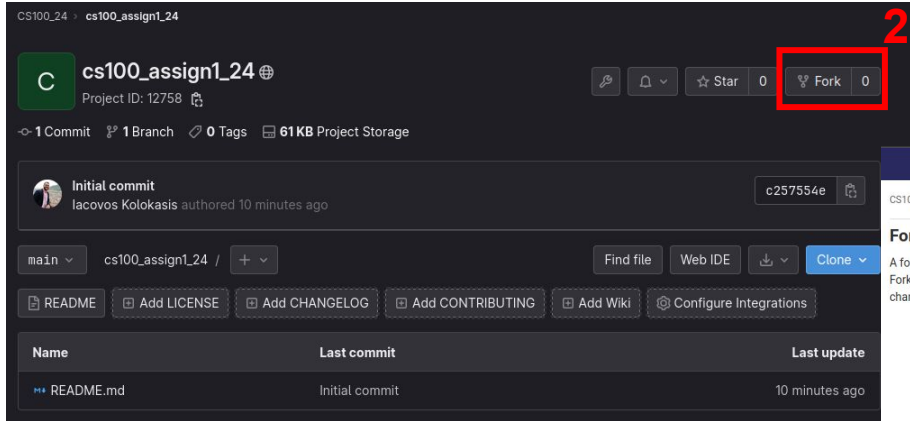
4

5

How to Fork a Repository

The screenshot displays the GitLab interface for the 'CS100' group. The left sidebar shows navigation options: Group overview, Details, Activity, Issues (0), Merge requests (0), Kubernetes, Packages & Registries, Members, and Settings. The main content area shows the 'CS100_23' group (Group ID: 1661) with buttons for 'New subgroup' and 'New project'. Below this, the 'CS100 - Introduction to Computer Science' section is visible. Under the 'Subgroups and projects' tab, a project named 'assign0' is listed, highlighted by a red box. A red number '1' is positioned to the right of the box. Search filters for 'Search by name' and 'Last updated' are also present.

How to Fork a Repository



CS100_24 cs100_assign1_24

cs100_assign1_24 Project ID: 12758

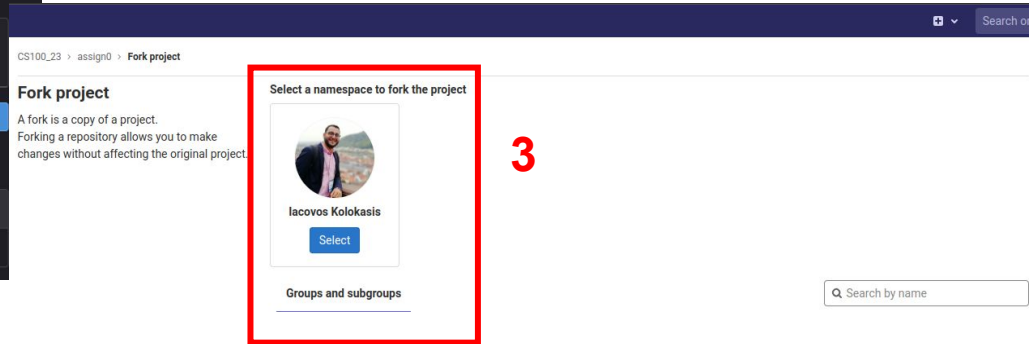
1 Commit 1 Branch 0 Tags 61 KB Project Storage

Initial commit
Iacovos Kolokasis authored 10 minutes ago

main cs100_assign1_24 / Find file Web IDE Clone

README Add LICENSE Add CHANGELOG Add CONTRIBUTING Add Wiki Configure Integrations

Name	Last commit	Last update
README.md	Initial commit	10 minutes ago




CS100_23 assign0 Fork project

Fork project

A fork is a copy of a project.
Forking a repository allows you to make changes without affecting the original project.

Select a namespace to fork the project


Iacovos Kolokasis
Select

Groups and subgroups

Search by name

Info About Fork

- You need to fork each assignment **only once!**
- If you want to download your project in different devices (e.g., laptop, desktop PC) you need to clone it

The screenshot shows a GitHub repository page for 'assign0' by user 'Iacovos Kolokasis'. A blue notification banner at the top states 'The project was successfully forked.' Below the repository name, there are statistics: 2 Commits, 1 Branch, 0 Tags, 133 KB Files, and 246 KB Storage. A 'Clone' button is highlighted with a red box. Below the clone button, there is a commit history table and a README section.

Name	Last commit	Last update
README.md	Update README.md	29 minutes ago
main.c	Initial commit	31 minutes ago
private.py	Initial commit	31 minutes ago

Assignment0: Hello World

Make Repository Private and Add TAs as Members

You can make the repo private and add TAs as members by hand or using the private.py script

By Hand

1. Go to Project Settings -> General
2. Visibility
3. Change Project Visibility to Private

Also add the TA in your project

1. Go to ProjectSettings -> Members

Make your Project Private

- To make your project private
- Go to Project Settings -> General
- Visibility
- Change Project Visibility to Private

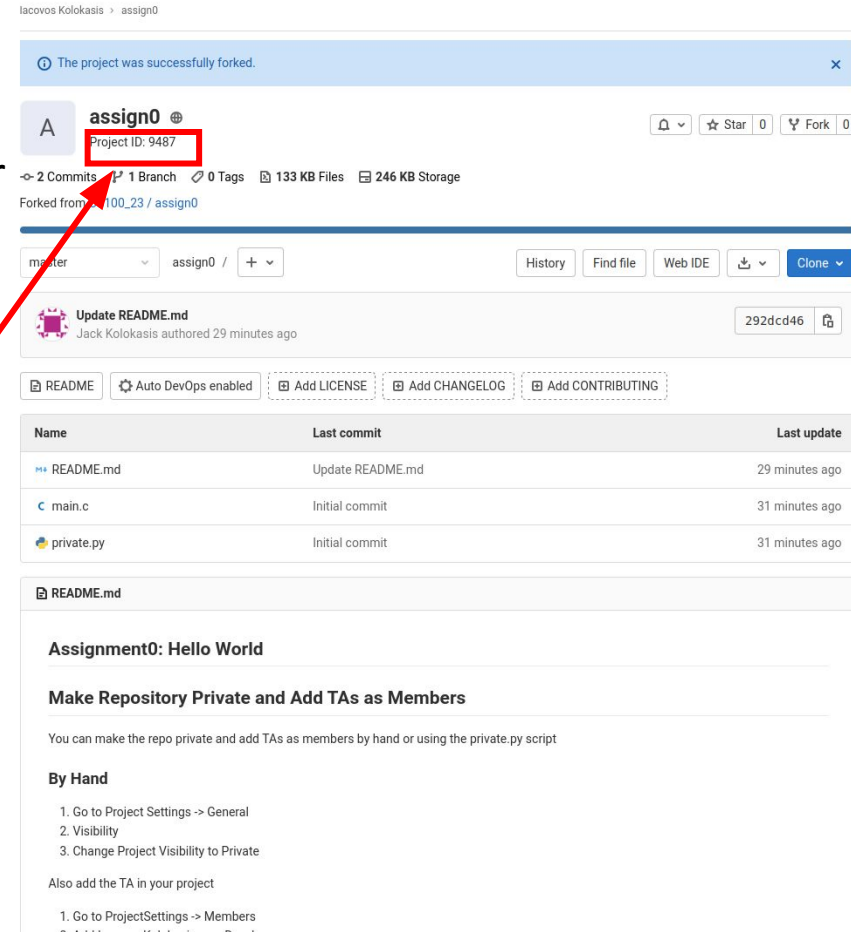
- To add the TAs in your project
- Go to Project Settings -> Members
- Add each TA as a Developer

- But we can avoid to do these steps on every assignment

Using private.py

- Run `pip3 install --upgrade python-gitlab --user`
- Copy your access token and the assignment project id

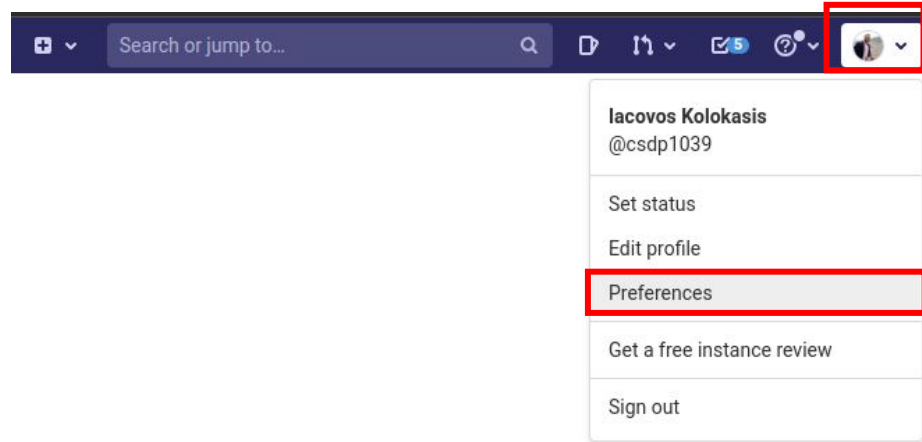
`python3 private.py -t <accessToken> -p <projectID>`



The screenshot shows a GitLab repository page for a project named 'assign0'. A red box highlights the 'Project ID: 9487' in the repository header. A red arrow points from this ID to the command 'python3 private.py -t <accessToken> -p <projectID>' in the text block to the left. The repository page includes a notification 'The project was successfully forked.', repository statistics (2 Commits, 1 Branch, 0 Tags, 133 KB Files, 246 KB Storage), a commit history table, and a README section titled 'Assignment0: Hello World' with instructions on how to make the repository private and add TAs as members.

Name	Last commit	Last update
README.md	Update README.md	29 minutes ago
main.c	Initial commit	31 minutes ago
private.py	Initial commit	31 minutes ago

Generating an Access Token



Generating an Access Token

The screenshot shows the GitLab user interface. At the top, the navigation bar includes the GitLab logo, 'Projects', 'Groups', 'More', and a search bar. The left sidebar contains 'User Settings' with a list of options: Profile, Account, Applications, Chat, Access Tokens (highlighted with a red box), Emails, Notifications, SSH Keys, GPG Keys, Preferences, Active Sessions, and Authentication log. The main content area is titled 'User Settings > Preferences' and features a search bar for settings. Under the 'Navigation theme' section, there is a description: 'Customize the appearance of the application header and navigation sidebar.' Below this, there are 12 color swatches arranged in a 3x4 grid. The first row contains Indigo (selected), Light Indigo, Blue, and Light Blue. The second row contains Green, Light Green, Red, and Light Red. The third row contains Dark, Light, and Dark Mode (alpha). Under the 'Syntax highlighting theme' section, there is a description: 'This setting allows you to customize the appearance of the syntax. [Learn more.](#)' Below this, there are three code snippets showing the same Ruby code with different syntax highlighting styles. The first snippet is 'White' (selected), the second is 'Dark', and the third is 'Solarized Light'.

User Settings > Preferences

Search settings

Navigation theme

Customize the appearance of the application header and navigation sidebar.

- Indigo
- Light Indigo
- Blue
- Light Blue
- Green
- Light Green
- Red
- Light Red
- Dark
- Light
- Dark Mode (alpha)

Syntax highlighting theme

This setting allows you to customize the appearance of the syntax. [Learn more.](#)

- White
- Dark
- Solarized Light

Generating an Access Token

User Settings > Access Tokens

Search settings

Personal Access Tokens

You can generate a personal access token for each application you use that needs access to the GitLab API.

You can also use personal access tokens to authenticate against Git over HTTP. They are the only accepted password when you have Two-Factor Authentication (2FA) enabled.

Add a personal access token

Enter the name of your application, and we'll return a unique personal access token.

Name

Add your name e.g., Iacovos Kolokasis

Expires at

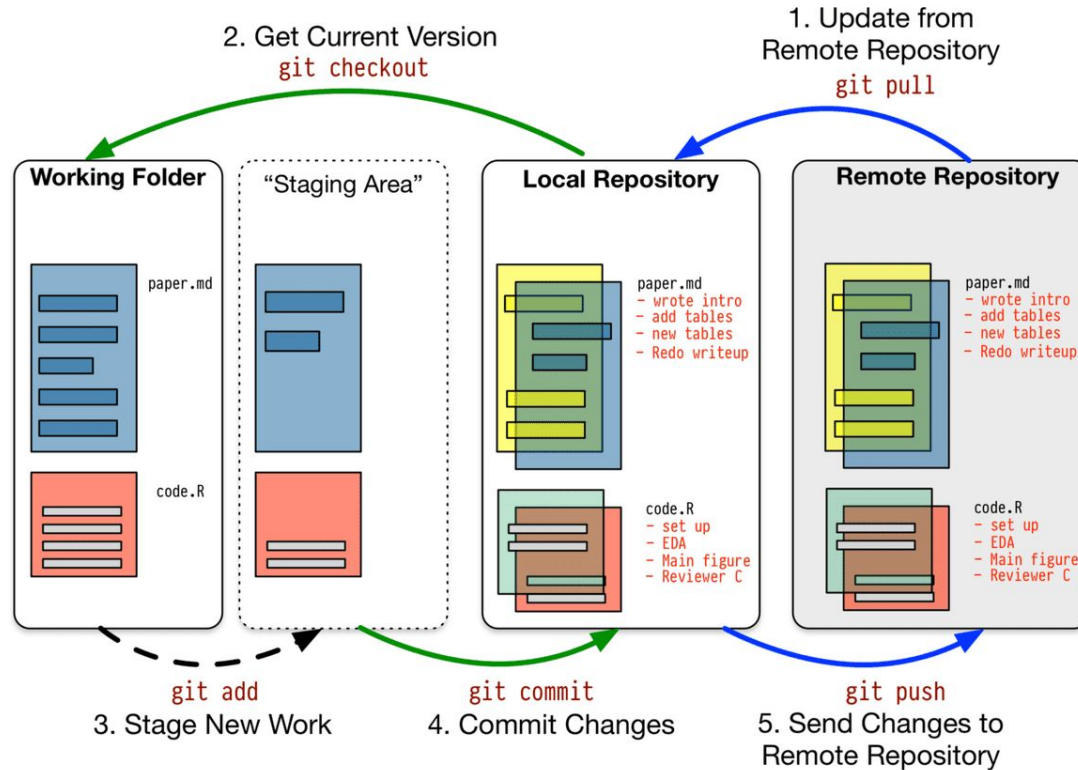
Scopes

- api**
Grants complete read/write access to the API, including all groups and projects, the container registry, and the package registry.
- read_user**
Grants read-only access to the authenticated user's profile through the /user API endpoint, which includes username, public email, and full name. Also grants access to read-only API endpoints under /users.
- read_api**
Grants read access to the API, including all groups and projects, the container registry, and the package registry.
- read_repository**
Grants read-only access to repositories on private projects using Git-over-HTTP or the Repository Files API.
- write_repository**
Grants read-write access to repositories on private projects using Git-over-HTTP (not using the API).
- read_registry**
Grants read-only access to container registry images on private projects.
- write_registry**
Grants write access to container registry images on private projects.
- sudo**
Grants permission to perform API actions as any user in the system, when authenticated as an admin user.

Select all the options

Create personal access token

How does Git Work?



How to Use Git?

- To download a git repo run
- To stage a file run
- To commit staged files run
- To save your code in the server run
- To turnin your code run "\$make turnin"
- To delete a previous turnin run "\$make undoTurnin"

Guidelines

- Commit often – at least one commit for each step
- Pay attention to your commit messages
- Describe precisely the purpose of the commit
- Commit only relevant files and modifications!!
- Push frequently
- In case something goes wrong with your PC/laptop

Thank you for your attention

Iacovos G. Kolokasis

University of Crete & ICS – FORTH

www.csd.uoc.gr/~kolokasis

kolokasis@ics.forth.gr

