Chapter 1
Git tutorial

This is a *sample* book written in Quarto. **Quarto** is an advanced version of Rmarkdown, which can combine LATEX with the simplicity of taking notes and integrating codes and calculations in the same document. Quarto is based on a text language markdown enriched with some functionalities of the Pandoc's Markdown package that allows for example to integrate a math equation $a^2 + b^2 = c^2$. A summary of the instructions available in **Quarto** are in the Cheat sheet.

1.1 Overview

In the very beginning, we introduce you the tools to be used during the semester, including:

- Virtual Machine
- Gitlab
- Quarto
- Visual studio code

From **Section 1.2 to 1.5** is a step-by-step tutorial for you to follow to start the project. **Section 1.6** is a brief introduction how you can use Gitlab with VS code. **Section C** introduces the common Quarto syntax.

1.2 Step 1 - Log into your Virtual Machine

To avoid setup issues, we created a virtual machine (**VM**) for you where all the necessary material is already installed. To access the VM, you have to install VMware Horizon and to log on **STI-FM-cours-2023** with your EPFL Gaspar account.

Important here!!!!! Everything that is stored on the VM will be erased once you log off. Therefore, store your project files in your personal document folder . However, python is not able to access your document folder due to path issues on Windows. While you are working on the project, move your project folder from your documents to the desktop to by-pass this issue. Then before log-off, move it back to your document folder or send your changes online (`commit` + `push`). We will try to find another way to avoid this copy-pasting step.

1.3 Step 2 - Git Clone the project

1.3.1 Set up your Gitlab account

What is git?

Since you are from EPFL, you already have a gitlab account. Login with your gaspar credentials and you should already have the project listed on your main page. If that's not the case, send a message to the TA with your username. To clone and make changes you'll nee a personal access token (PAT) and to create in click on the user profile image (top-left) and select "Preferences" as in the picture bellow and then select "Access Tokens".

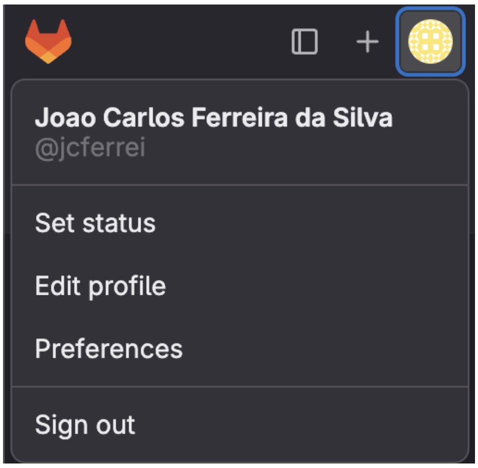
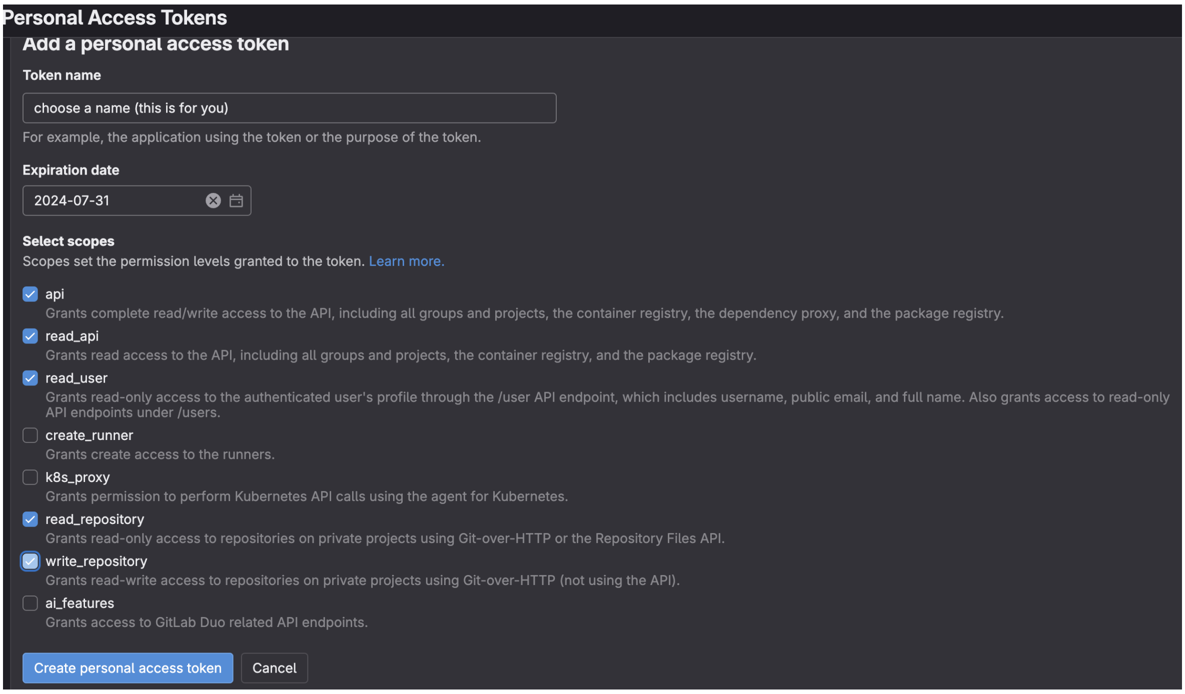


Figure 1.1—Go to Preferences

Here you'll be able to create new Token By clicking on "Add new token" and filling the information like in the picture bellow and click on create:



Personal Access Tokens

Add a personal access token

Token name

choose a name (this is for you)

For example, the application using the token or the purpose of the token.

Expiration date

2024-07-31

Select scopes

Scopes set the permission levels granted to the token. Learn more.

☑ api
Grants complete read/write access to the API, including all groups and projects, the container registry, the dependency proxy, and the package registry.

☑ read_api
Grants read 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.

□ create_runner
Grants create access to the runners.

□ k8s_proxy
Grants permission to perform Kubernetes API calls using the agent for Kubernetes.

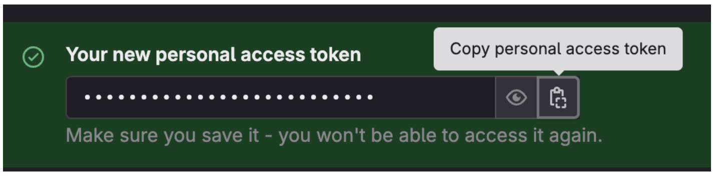
☑ read_repository
Grants read-only access to repositories on private projects using Git-over-HTTP or the Repository Files API.

☑ write_repository
Grants read-write access to repositories on private projects using Git-over-HTTP (not using the API).

□ ai_features
Grants access to GitLab Duo related API endpoints.

Figure 1.2—create new PAT

You should now have a new token that you can copy and save it somewhere as you'll need it for the next steps and you won't be able to see or copy it again.



Your new personal access token

••••••••••••••••••••••••••

Make sure you save it - you won't be able to access it again.

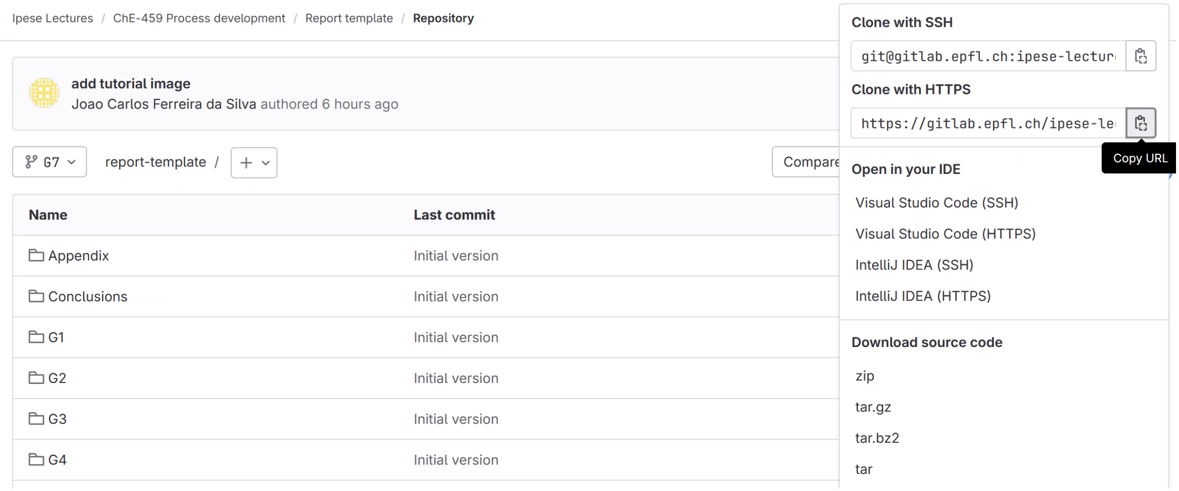
Figure 1.3—copy the new PAT

1.3.2 Clone the project repository to your VM and local machine

For this course, we prepared a git repository with a folder for each group. Any changes will be connected to your account so make sure you only change files in your groups folder. The project is saved online. To have it locally on your VM/computer do the following:

1. Open your git repository in gitlab and copy the HTTPS url of your repository (Figure 1.4).
2. Open your *terminal* or *command prompt* and go to the folder where you would like to clone your repository using the `cd` command. In Figure 1.5, you can see that I chose the `Desktop` folder.

3. Clone the repo using the command `git clone` (Figure 1.5). While cloning the repo, it could be that you are asked to enter Gaspar credentials in the Password section (Figure 1.6). You need your access token here instead of a password otherwise, you can not pass the authentification here.



Ipese Lectures / ChE-459 Process development / Report template / **Repository**

add tutorial image
Joao Carlos Ferreira da Silva authored 6 hours ago

Name	Last commit
Appendix	Initial version
Conclusions	Initial version
G1	Initial version
G2	Initial version
G3	Initial version
G4	Initial version

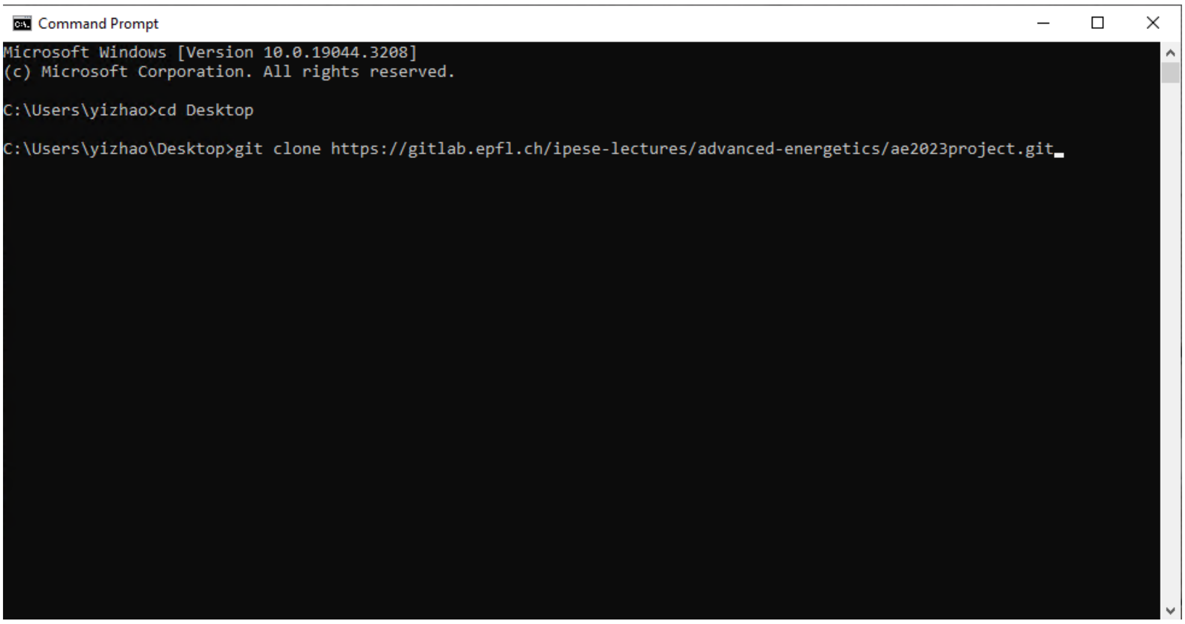
Clone with SSH

`git@gitlab.epfl.ch:ipese-lectur`

Clone with HTTPS

`https://gitlab.epfl.ch/ipese-le`

Figure 1.4—Copy the URL

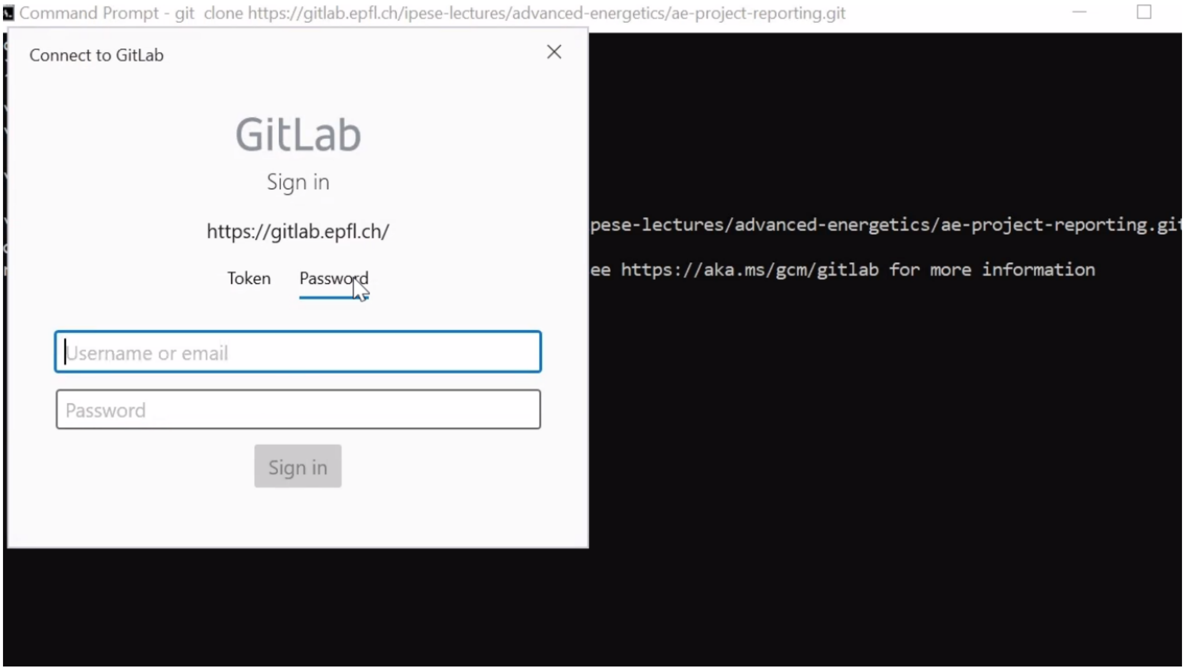


```
Microsoft Windows [Version 10.0.19044.3208]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yizhao>cd Desktop

C:\Users\yizhao\Desktop>git clone https://gitlab.epfl.ch/ipese-lectures/advanced-energetics/ae2023project.git_
```

Figure 1.5—Clone the repo



Command Prompt - git clone https://gitlab.epfl.ch/ipese-lectures/advanced-energetics/ae-project-reporting.git
Connect to GitLab
GitLab
Sign in
https://gitlab.epfl.ch/
Token
Password
Username or email
Password
Sign in

```
pese-lectures/advanced-energetics/ae-project-reporting.gi

ee https://aka.ms/gcm/gitlab for more information
```

Figure 1.6—Login to git

Your repo is now in your computer/VM. Last thing you need to do is changing your brach to not be affected by the othere groups work. In the same terminal you did the git clone, now move inside the folder with `cd report-template` and do `git checkout <group-name>`. (You'll also won't have the right to push changes to the default branch so you really need to change branches). The group/branch names are : Group_1, Group_2, Group_3, Group_4, Group_5, Group_6, Group_7, Group_8, Group_9.

You can start to work on it! In the following two sections, we will briefly introduce visual studio code.

Chapter C
Quarto basics

Quarto is **an open-source scientific and technical publishing system** (according to the official website).

ℹ Note

Using less "catchy" words, Quarto allows you to section C.1 (such as projects' reports) while using data from files section C.2 to display them as table or **??**.

C.1 Write document

C.1.1 Markdown syntax

If you're not familiar with Markdown syntax, please read Markdown basics from Quarto's documentation which provides a good summary of the most frequently used elements.

Alternatively, you can open this document's source in VS Code or any other editor to see how it has been written (Appendix/example.qmd).

C.1.2 Figures

```
![Example of image integration](ipeseImage-1536x864.jpg)
```

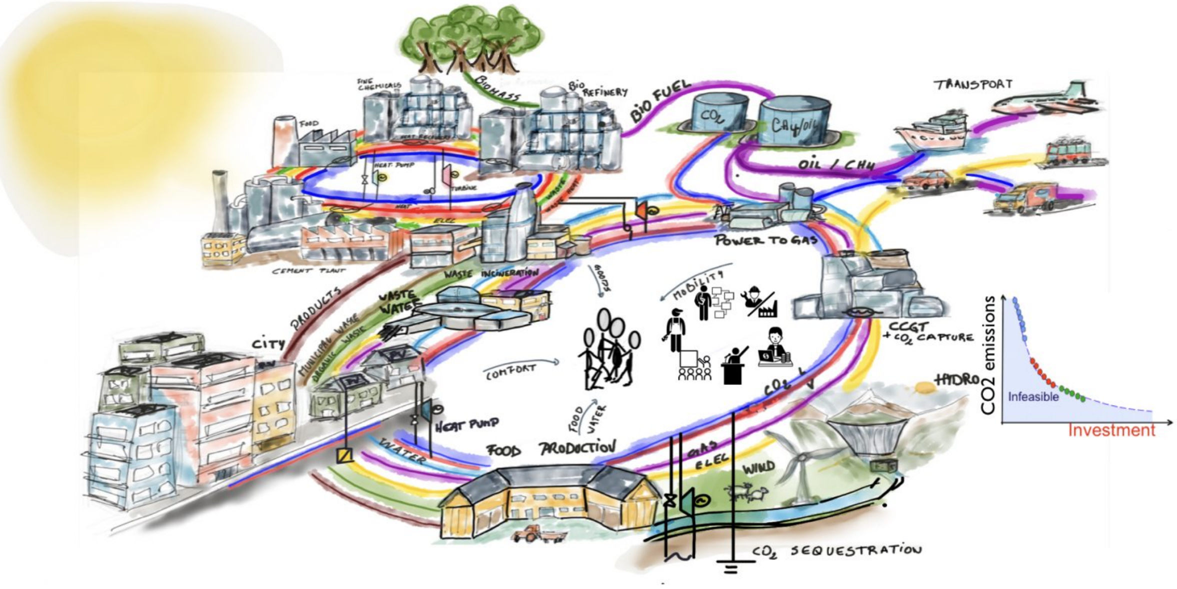


Figure C.1—Example of image integration

C.1.3 Tables

Multiple syntaxes can be used to display tables, you'll find the simpler below and more informations in the official documentation.

```
| Right | Left | Default | Center |
|------:|:-----|---------|:------:|
|   12  |  12  |    12   |    12  |
|  123  |  123 |   123   |   123  |
|    1  |    1 |     1   |     1  |

: Demonstration of pipe table syntax
```

Table C.1—Demonstration of pipe table syntax

Right	Left	Default	Center
12	12	12	12
123	123	123	123
1	1	1	1

ℹ Note

The line starting with `:` is the title and can be omitted. However, only the tables with titles will be added to the list of tables.

C.2 Avoid copy/pasting aka use directly csv

C.2.1 Read csv

```
library(knitr)
rings <- read.csv('example.csv')
kable(rings, caption = "Rings to gather")
```

Table C.2—Rings to gather

owner	number
Elven-kings under the sky	3
Dwarf-lords in their halls of stone	7
Mortal Men doomed to die	9
Dark Lord on his dark throne	1

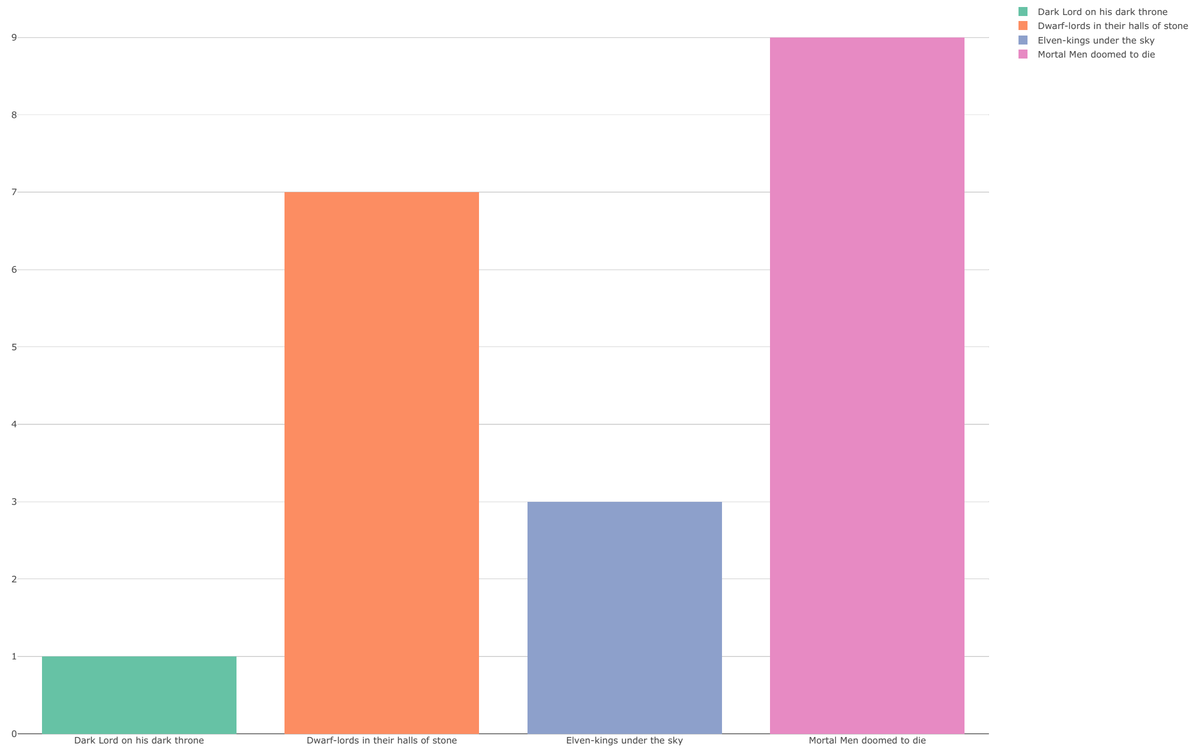
C.2.2 Manipulate

```
`{r} sum(rings$number)` rings were forged.
```

20 rings were forged.

C.3 Render

```
plotly::plot_ly(
        x = rings$owner,
        y = rings$number,
        color = rings$owner,
        type = "bar"
      )
```


🟩 Dark Lord on his dark throne
🟧 Dwarf-lords in their halls of stone
🟦 Elven-kings under the sky
🟪 Mortal Men doomed to die
9
8
7
6
5
4
3
2
1
0
Dark Lord on his dark throne
Dwarf-lords in their halls of stone
Elven-kings under the sky
Mortal Men doomed to die