

Kurzanleitung zum Arbeiten mit TU-GitLab

Inhalt

1	Nutzungsberechtigte.....	2
2	Client Software.....	2
3	Bedingung für erfolgreiches Anmelden in GitLab an der RWTH Aachen.....	2
4	Erstanmeldung in GitLab an der RWTH Aachen.....	4
5	Erstes Projekt anlegen.....	5
6	Projekte mit HTTPS verbinden.....	6
7	Projekte mit SSH-Key verbinden.....	7
7.1	SSH-Key erzeugen.....	7
7.2	SSH-Key: im GitLab-Profil hinterlegen.....	8
7.3	SSH-Agenten auf dem lokalen PC starten.....	10
8	Erstes Arbeiten mit Git und GitLab.....	12
8.1	Neues Repository anlegen von der Kommandozeile.....	13
9	Lifecycle-Prozesse der TU-GitLab Instanzen.....	15
10	Archivieren von Projekten.....	16
11	Weiterführende Links und Anleitungen.....	17

1 Nutzungsberechtigte

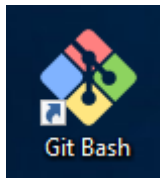
TU-GitLab wird durch die nationale Forschungsdateninfrastruktur, NFDI4ING, über die RWTH Aachen (Rheinisch-Westfälische Technische Hochschule Aachen) für die TU Darmstadt angeboten.

Interne Anwender/innen	Externe
<ul style="list-style-type: none">• Mitarbeitende und Studierende der Konsortialpartner• besitzen einen DFN-Account• Authentifizierung über DFN-AAI	<ul style="list-style-type: none">• nicht dem Konsortium angehörende Personen• können in Projekten mitarbeiten• Authentifizierung über GitHub-Accounts

2 Client Software

Um mit der Versionsverwaltung *Git* und *GitLab* zu arbeiten, benötigen Sie die *Git*-Clientsoftware auf Ihrem lokalen PC. Das Programm kann von der Web-Seite <https://git-scm.com/downloads> heruntergeladen und anschließend installiert werden.

Danach befindet sich die *Git Bash* auf Ihrem PC:



3 Bedingung für erfolgreiches Anmelden in GitLab an der RWTH Aachen

Damit Sie sich erfolgreich in GitLab anmelden können, müssen Sie im IDM-Portal der TU Darmstadt dem *DFN-AAI* zugestimmt haben.

Melden Sie sich dazu im IDM-Portal <https://www.idm.tu-darmstadt.de> mit Ihrer TU-ID und Ihrem Passwort an. Wählen Sie *IDM-Portal* aus der Liste der HRZ Links aus.

Dort klicken Sie in der persönlichen Accountverwaltung auf *Zustimmungen*:

- Persönliche Accountverwaltung
- Adressbuch
- Kontaktdaten verwalten
- Gast TU-ID
- Gruppenmitgliedschaften
- Notfallkontakt
- Passwort neu setzen
- WLAN Besucher
- Zustimmungen**
- Zweckgebundener Datenbrief

Dann wieder auf *Zustimmungen*:

Zustimmungen

An der TU-Darmstadt ist es bei einigen Services, wie bei dem DFN-AAI Shibboleth oder dem Sync&Share-Service Hessenbox-DA, erforderlich, dass Sie den Nutzungsbedingungen zustimmen. Sollten Sie Ihre Zustimmung bereits erteilt haben, besteht hier die Möglichkeit diese zurückzunehmen.

[Zustimmungen](#)

Für das Arbeiten mit Gitlab müssen Sie die Zustimmung zur Nutzung von DFN-AAI erteilen:

Zustimmung zur Nutzung von DFN-AAI

Sie haben der Nutzung von DFN-AAI NICHT zugestimmt. ✘

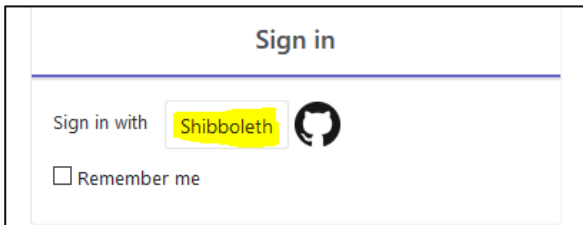
Die DFN-AAI-Föderation ist ein Dienst des DFN-Vereins für wissenschaftliche Einrichtungen (Universitäten, Institute) und Anbieter (kommerziell und nicht kommerziell). Sie schafft das notwendige Vertrauensverhältnis sowie einen organisatorischen und technischen Rahmen für den Austausch von Benutzerinformationen zwischen Einrichtungen und Anbietern.

[Zustimmung erteilen](#)

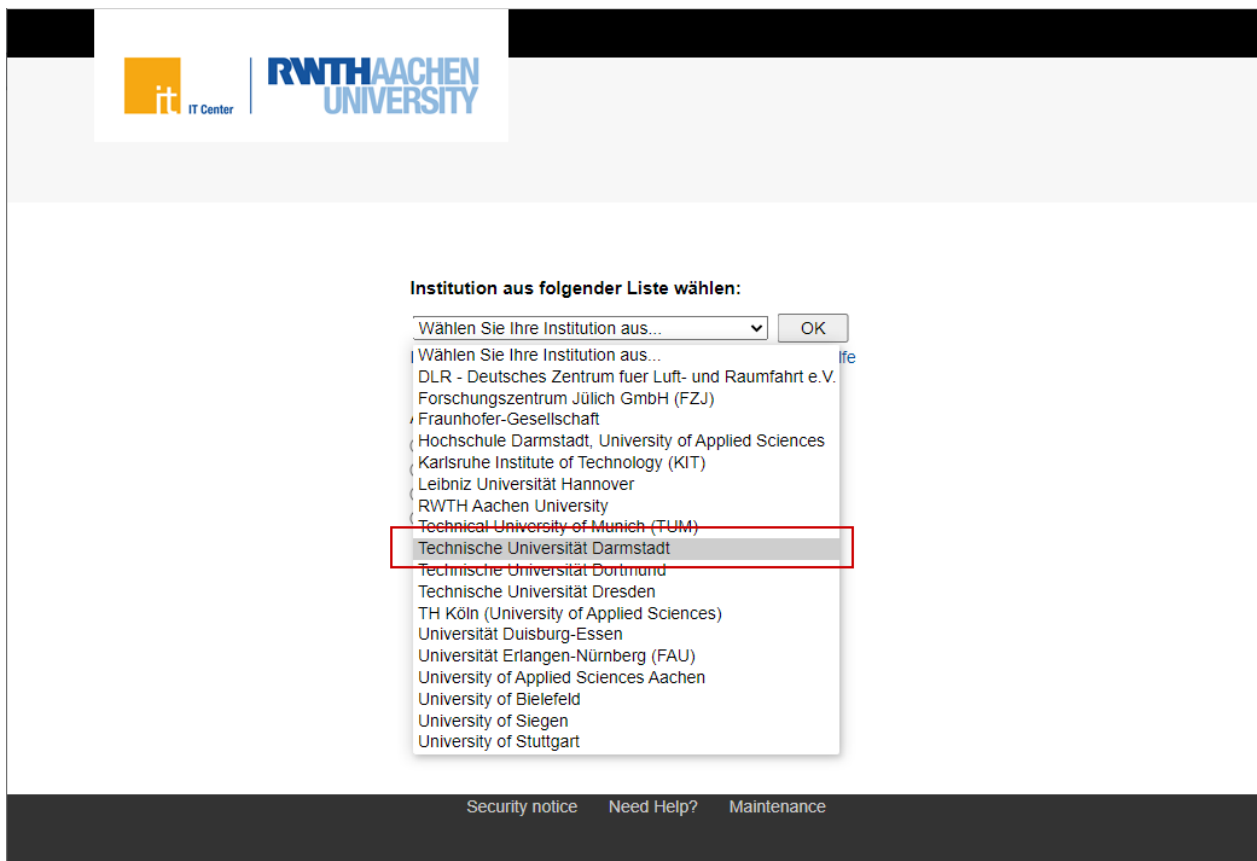
Sie können sich nun wieder aus dem IDM-Portal abmelden.

4 Erstanmeldung in GitLab an der RWTH Aachen

Rufen Sie die Web-Seite <https://git.rwth-aachen.de/> auf und wählen Sie zum Einloggen (Sign In) *Shibboleth* aus:



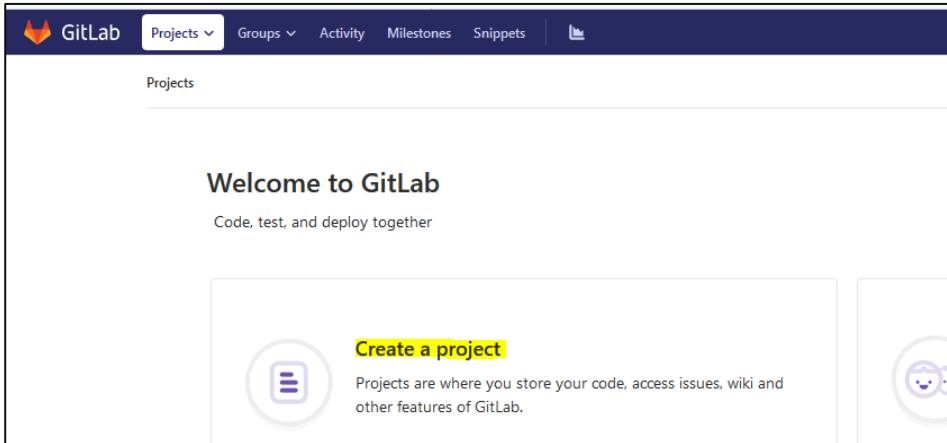
Sie kommen zur *Auswahlseite für die angeschlossenen Hochschulen*. Wählen Sie aus der Dropdown-Liste *Technische Universität Darmstadt* aus:



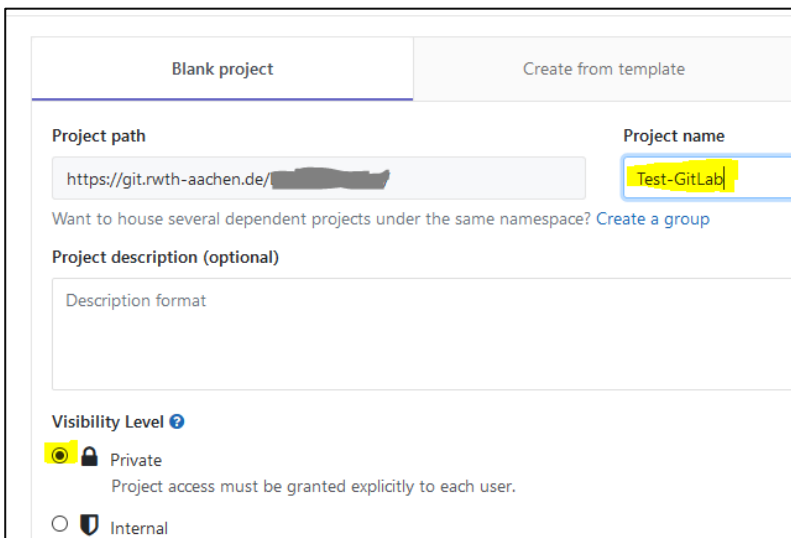
Melden Sie sich mit Ihrer TU-ID und dem dazugehörigen Passwort an. Bestätigen Sie die Nutzungsbedingungen und die Weitergabe der Daten.

5 Erstes Projekt anlegen

Erzeugen Sie ein erstes Projekt, mit dem Sie die Verbindung von Ihrem lokalen PC zum Repository in Aachen testen können.



Geben Sie einen Namen für das Projekt ein und belassen Sie den Projekt-Typ (*Visibility Level*) auf *Private*. Gehen Sie anschließend auf *Create Project*.

The image shows the 'Create Project' form in GitLab. There are two tabs at the top: 'Blank project' (selected) and 'Create from template'. The 'Project path' field contains 'https://git.rwth-aachen.de/'. The 'Project name' field contains 'Test-GitLab' and is highlighted with a blue border. Below the path field, there is a link: 'Want to house several dependent projects under the same namespace? Create a group'. The 'Project description (optional)' field is empty. Under 'Visibility Level', the 'Private' option is selected with a yellow highlight, and the 'Internal' option is unselected. A note below the visibility level options states: 'Project access must be granted explicitly to each user.'

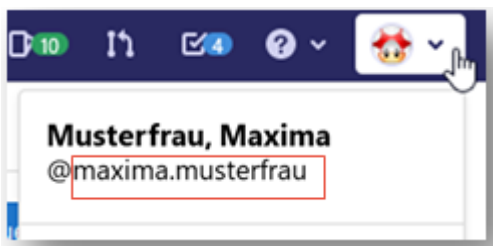
Sobald das Projekt angelegt ist, wird es unter *Projects* -> *Your Projects* aufgelistet. Um Ihr Projekt lokal zu clonen, können Sie zwischen SSH und HTTPS wählen. Nachfolgend sind beide Varianten vorgestellt.



6 Projekte mit HTTPS verbinden

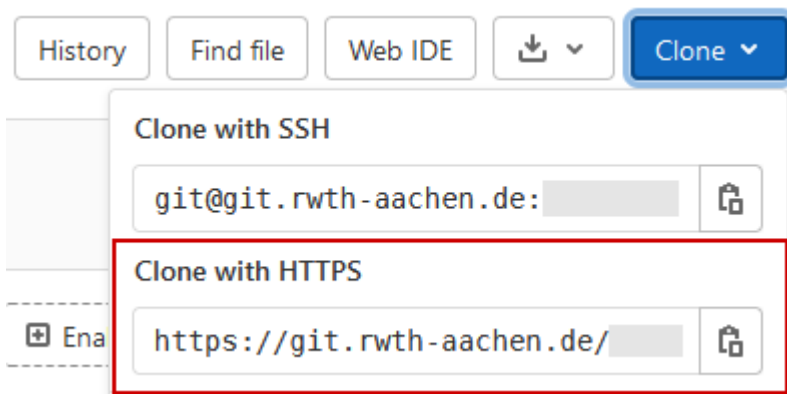
Eine Möglichkeit Ihr Remote Repository lokal mit Ihrem PC zu verbinden ist mittels HTTPS Verbindung. Sie benötigen dafür Ihren GitLab-Account Usernamen, Ihr Passwort sowie den HTTPS-Link zu Ihrem Projekt.

Ihren Usernamen finden Sie rechts oben bei Ihren Account-Informationen. Gemeint ist der Name hinter dem @. Meist hat er die Form „vorname.nachname“.



Als Passwort verwenden Sie das Passwort der TU-ID, mit welcher Sie sich einloggen.

Den HTTPS-Link zu Ihrem Projekt finden Sie beim Button „Clone“. Wählen Sie die Variante „Clone with HTTPS“.



Um ein existierendes Projekt lokal zu klonen, verwenden Sie in Ihrer GitBash folgenden Befehl:

```
git clone https://git.rwth-aachen.de/ihrname/ihrprojekt.git
```

Beim ersten Verbinden werden Sie dann nach Ihrem Usernamen und Passwort gefragt.

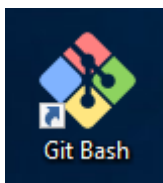
(Siehe auch Kapitel 8)

7 Projekte mit SSH-Key verbinden

Eine weitere Variante Projekte zwischen Ihrem lokalen PC und dem Remote Repository GitLab an der RWTH Aachen zu verbinden funktioniert über eine SSH-Verbindung. So werden die Daten über eine verschlüsselte Netzwerkverbindung übertragen.

7.1 SSH-Key erzeugen

Zum Erzeugen eines SSH-Keys starten Sie die *Git Bash* auf Ihrem lokalen PC:



Sie befinden sich automatisch in Ihrem Home-Laufwerk (Im Windows Explorer ist dieses Verzeichnis auf dem Laufwerksbuchstaben *C* bzw. *H*). In der *Git Bash* wird es als `/c` , `/h` oder `~` (Tilde) dargestellt.

Legen Sie zum Speichern der Schlüssel einen Ordner unterhalb Ihres Home-Laufwerkes an und erzeugen Sie die Schlüssel in dem angelegten Verzeichnis:

```
mkdir SSH-Keys  
/usr/bin/ssh-keygen -t rsa -b 4096 -f /<Home-Laufwerk>/<Ordner>/<Schlüsselname>
```

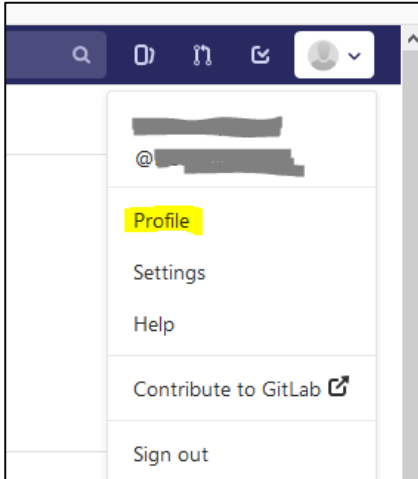
Beispiel: Home-Laufwerk ist `/h`, Ordner ist *SSH-Keys* und der Schlüsselname *key-fuer-gitlab*:

```
/usr/bin/ssh-keygen -t rsa -b 4096 -f /h/SSH-Keys/key-fuer-gitlab
```

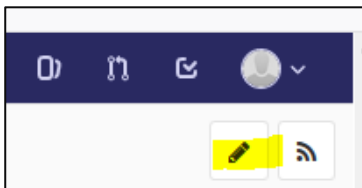
Bei den Fragen nach *passphrase* geben Sie ein Password ein.

7.2 SSH-Key: im GitLab-Profil hinterlegen

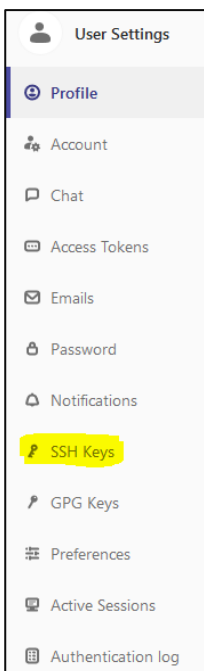
Öffnen Sie auf der GitLab-Seite die Profil-Einstellungen:



Gehen Sie dann auf Profil bearbeiten:



Und wählen Sie im Menü auf der linken Seite *SSH Keys*:



Auf dem lokalen PC öffnen Sie nun den in Kapitel 7.1 abgespeicherten Public Key (Dateiendung: `.pub`) mit einem Editor (z.B. *Notepad++*); kopieren den Inhalt (*Strg-A*, dann *Strg-C*) und fügen diesen dann in das entsprechende Feld im GitLab-Profil ein. Füllen Sie das Feld *Title* aus und fügen Sie den Key über *Add key* hinzu.

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_rsa.pub` and begins with `'ssh-rsa'`. Don't use your private SSH key.

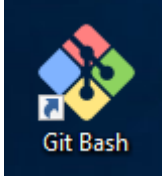
```
ssh-rsa AAAAB3NzaGZjaGki...  
/bZf5xTf6hzd9HZoKvZHAL  
/UXu1n44YHAMjynCN2cGnKMV4y78LB+M3U5soYItuxJ93BzUiaEC6gjRL7taLiE59AgWd6qwYFs  
5n...ap/TMSjNAnTtiffC2b4gRLb2knzDX0f0G  
/aMHPnncg9h2HPReycd3t9NEH1JXkKxm6adYzRNp284fPA5M108ZPNWTm64KV3PXub8  
/QJANrKrzU8n0D05LYk00pYmbVws5nHEHwPc1X2sl  
/ouMj2YjNTuEbSRd83qxNSvOgAY17TvuLuObGfHwx0f2rUCw3+Jyfk9voVzOjbiw8c6Uq+V9jrkWb  
/bBuxhBNEGTPy9bY4RjxE8vga7Et1bBswyCpRj/eh629hPAe3vHq  
/JS2tbx7VPvIz6R4Kfip9DSCABDVGywl
```

Title

Name your individual key via a title

7.3 SSH-Agenten auf dem lokalen PC starten

Starten Sie die *Git Bash* auf Ihrem lokalen PC



Geben Sie folgende Befehle in der Bash ein, um den SSH-Agenten zu starten und den privaten Schlüssel hinzuzufügen.

```
eval "$(ssh-agent -s)"  
ssh-add /<Home-Laufwerk>/<Ordner>/<Schlüsselname>
```

(Geben Sie hier den korrekten Pfad und den Namen (Dateiname **OHNE** die Endung *.pub*) an, die Sie in Kapitel 7.1 gewählt haben)

Hier wird nach dem Passwort gefragt, das Sie in Kapitel 7.1 vergeben haben.

```
user@... MINGW64 ~  
$ eval "$(ssh-agent -s)"  
Agent pid 7420  
  
user@... MINGW64 ~  
$ ssh-add /h/SSH-Keys/key-fuer-gitlab  
Enter passphrase for /h/SSH-Keys/key-fuer-gitlab:  
Identity added: /h/SSH-Keys/key-fuer-gitlab (/h/SSH-Keys/key-fuer-gitlab)  
  
user@... MINGW64 ~  
$ |
```

Damit diese beiden Befehle nicht bei jedem Start der *Git Bash* ausgeführt werden müssen, muss der folgende Inhalt in die Datei *.bashrc* eingetragen werden. Diese Datei steht direkt im Home-Laufwerk (Laufwerksbuchstabe *C* oder *H*).

```
env=~/.ssh/agent.env  
  
agent_load_env () { test -f "$env" && . "$env" >| /dev/null ; }  
  
agent_start () {  
  (umask 077; ssh-agent >| "$env")  
  . "$env" >| /dev/null ; }  
  
agent_load_env  
  
# agent_run_state: 0=agent running w/ key; 1=agent w/o key; 2= agent not running  
agent_run_state=$(ssh-add -l >| /dev/null 2>&1; echo $?)  
  
if [ ! "$SSH_AUTH_SOCK" ] || [ $agent_run_state = 2 ]; then
```

```
agent_start
ssh-add /h/SSH-Keys/key-fuer-gitlab
elif [ "$SSH_AUTH_SOCK" ] && [ $agent_run_state = 1 ]; then
ssh-add
fi

unset env
```

Wird dann die *Git Bash* wieder gestartet, wird einmal nach dem Passwort für den SSH-Key gefragt, das dann gültig ist, solange die Bash offen ist.

8 Erstes Arbeiten mit Git und GitLab

Auf der GitLab Web-Seite wählen Sie das in Kapitel 5 angelegte Projekt aus. Bei einem neuen Projekt werden immer Command Line-Befehle für die ersten Arbeiten aufgelistet.

Bitte beachten, dass das hier beschriebene „Create a new repository“ nur funktioniert, wenn das Projekt schon über die Web-Seite neu angelegt wurde. In Kapitel 8.1 wird beschrieben, wie man ein Repository von der Kommandozeile aus anlegt.

Command line instructions

Git global setup

```
git config --global user.name "[REDACTED]"
git config --global user.email "[REDACTED]@hrz.tu-darmstadt.de"
```

Create a new repository

```
git clone git@git.rwth-aachen.de:[REDACTED]/Test2-GitLab.git
cd Test2-GitLab
touch README.md
git add README.md
git commit -m "add README"
git push -u origin master
```

Existing folder

```
cd existing_folder
git init
git remote add origin git@git.rwth-aachen.de:[REDACTED]/Test2-GitLab.git
git add .
git commit -m "Initial commit"
git push -u origin master
```

Existing Git repository

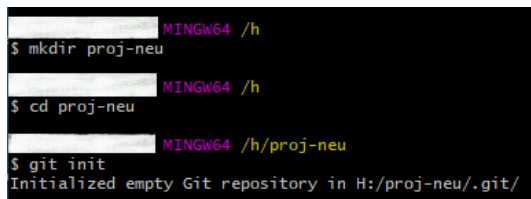
```
cd existing_repo
git remote rename origin old-origin
git remote add origin git@git.rwth-aachen.de:[REDACTED]/Test2-GitLab.git
git push -u origin --all
git push -u origin --tags
```

8.1 Neues Repository anlegen von der Kommandozeile

Wurde das neue Projekt noch nicht über die GitLab Web-Seite angelegt, kann es auch über die Kommandozeilen der *Git Bash* erzeugt werden:

- Neues lokales Verzeichnis anlegen (hier im Beispiel hat es den Namen „proj-neu“):

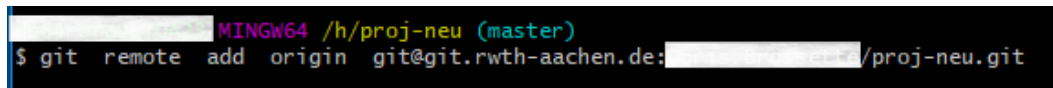
```
mkdir proj-neu
cd proj-neu
git init
```



```
MINGW64 /h
$ mkdir proj-neu
MINGW64 /h
$ cd proj-neu
MINGW64 /h/proj-neu
$ git init
Initialized empty Git repository in H:/proj-neu/.git/
```

- Neues Repository auf dem GitLab Server anlegen. Der Projekt- und Repositoryname sollte gleich sein:

```
git remote add origin git@git.rwth-aachen.de:<GitLab-Schemaname>/<Projektname>.git
```



```
MINGW64 /h/proj-neu (master)
$ git remote add origin git@git.rwth-aachen.de: /proj-neu.git
```

- Nun muss man eine Datei anlegen (hier *README*), ansonsten werden die Git-Daten nicht ins Repository hochgeladen:

```
touch README
git add README
git commit -m „initial commit“
git push -u origin master
```

```
MINGW64 /h/proj-neu (master)
$ touch README

MINGW64 /h/proj-neu (master)
$ git add README

MINGW64 /h/proj-neu (master)
$ git commit -m "initial commit"
[master (root-commit) 5e3dec4] initial commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README

MINGW64 /h/proj-neu (master)
$ git push -u origin master
Warning: Permanently added the ECDSA host key for IP address '134.130.122.52' to the list of known hosts.
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 221 bytes | 44.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: The private project Doris.Brossette/proj-neu was successfully created.
remote:
remote: To configure the remote, run:
remote:   git remote add origin git@git.rwth-aachen.de:Doris.Brossette/proj-neu.git
remote:
remote: To view the project, visit:
remote:   https://git.rwth-aachen.de/[redacted]/proj-neu
remote:
To git.rwth-aachen.de:[redacted]/proj-neu.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

MINGW64 /h/proj-neu (master)
$ |
```

Damit ist das neue Repository auf GitLab angelegt und man kann wie gewohnt damit arbeiten.

9 Lifecycle-Prozesse der TU-GitLab Instanzen

In den [Nutzungsbedingungen der RWTH Aachen](#) sind für GitLab Lifecycle-Prozesse für Projekte und Nutzer_innen verankert.

Projekt-Lifecycle

Wird ein Projekt über den Zeitraum von 24 Monaten nicht genutzt, erfolgt eine erste Nachfrage per E-Mail bei dem/der Projekthinhaber_in, ob das Projekt weiterhin benötigt wird. Nach weiteren 3 Monaten sowie nach 6 Monaten erfolgt eine erneute Nachfrage. Ist nach dreimaliger Nachfrage nutzerseitig keine Rückmeldung erfolgt, wird das Projekt nach einer weiteren Woche gelöscht.

Der Projekt-Lifecycle bezieht sich nur auf ungenutzte Projekte. Solange Änderungen am Projekt vorgenommen werden, z.B. in Form eines Commit, ist das Projekt von der Löschung nicht betroffen.

Wollen Sie ein Projekt unverändert über die genannten 24+6 Monate hinaus erhalten, setzen Sie Ihr Projekt in den „Archiv-Status“. Es unterliegt dann keinen Löschfristen. Eine Beschreibung zur Archivierung finden Sie in Kapitel 10.

Details zum Lifecycle in der Anleitung der RWTH Aachen:

<https://help.itc.rwth-aachen.de/service/ubrf9cmzd17m/article/7bf138d794de4299ac46d954f708e6eb/>

10 Archivieren von Projekten

Projekte, die nicht mehr geändert werden, aber dennoch als Archiv langfristig zur Verfügung stehen sollen, können über die Projekteinstellungen archiviert werden.

Öffnen Sie den Menübereich: Settings -> General. Scrollen Sie dann im Inhaltsbereich bis zum Bereich „Advanced“ und öffnen Sie diesen. Dort finden Sie den Abschnitt „Archive project“. Wird das Projekt archiviert, ist es nur noch lesend verfügbar. Es können keine Änderungen mehr vorgenommen werden.

Archive project

Archiving the project will make it entirely read only. It is hidden from the dashboard and doesn't show up in searches. **The repository cannot be committed to, and no issues, comments, or other entities can be created.** [Learn more.](#)

Archive project

Sollten Sie zu einem späteren Zeitpunkt Ihr Projekt wieder aus dem Archiv in den aktiven Zustand zurückführen wollen, können Sie das an derselben Stelle über den Button „Unarchive project“ tun.

Unarchive project

Unarchiving the project will restore its members' ability to make changes to it. The repository can be committed to, and issues, comments, and other entities can be created. **Once active, this project shows up in the search and on the dashboard.** [Learn more.](#)

Unarchive project

Hier geht's zur Anleitung der RWTH Aachen:

<https://help.itc.rwth-aachen.de/service/ubrf9cmzd17m/article/c64a3b17dfaf457ca0911d57e3a92b2f/>

11 Weiterführende Links und Anleitungen

HRZ-Homepage zu TU-GitLab: <https://www.hrz.tu-darmstadt.de/gitlab>

Anleitungsseiten der RWTH Aachen: <https://help.itc.rwth-aachen.de/service/ubrf9cmzd17m>

GitLab Instanz: <https://git.rwth-aachen.de/>

GitLab CE Instanz: <https://git-ce.rwth-aachen.de>