



[martin-leandro.paleico@gwdg.de](mailto:martin-leandro.paleico@gwdg.de)

Martin Paleico

## Git and Gitlab

Collaborative Work, Issue and Feature Tracking

# Table of contents

**1** Version Control and Git

**2** Gitlab

# Today

- Learn a bit about version control and Git (if you haven't already)
- Install Gitlab CE
- Test some of Gitlab's collaborative tools and a bit of Git
- Plenary conclusion

# Outline

**1** Version Control and Git

2 Gitlab

# Version control for system administrators

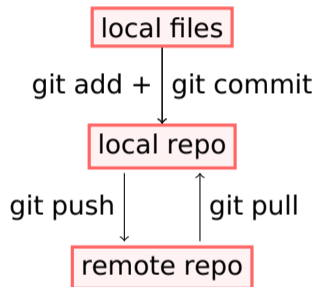
“Framework that allows for keeping track of changes made to files”

- Normally used for code, but relevant for sys. admins: configuration files, documentation, many libraries and services available as repositories
- Versioning
- "Backuping"
- History of changes and reasoning behind them -> Sneaky documentation
- Branching: Work on new features without overwriting base configuration
- Transferability to other systems (e.g. from dev to live system)
- Collaborative and simultaneous work with other admins

# Features of Git

- "Distributed": no unique central repository for files, many local and remote repositories possible with more or less equal rank
- "Non-locking": multiple people can work on the same file (have to deal with it afterward)
- Many other frameworks (mercurial, subversion, etc.) with different philosophies Figure of commit, pull, push
- Many possibilities for remote repository. Here: Gitlab

## Git scheme (reminder)



Reality can be much more complicated, see for example  
<https://blog.osteele.com/2008/05/my-git-workflow/>

## Good practices when working with Git

- Changes to files are stored as diffs, so don't commit binaries, images, PDF's/MS Office formats, etc. -> Store the scripts that generate those when possible
- Don't commit huge files
- Small changes with continuous commits that fix one issue
- Commit functional configurations and code
- Use branches for testing and development
- Mistakes are fixable but sometimes disentangling a repository is hard

# Good practices when working with Git

- Once something enters a repository, it can be very hard to get rid of it: Careful with passwords, API tokens, etc.!

Docker config file

```
1  version: "3.7"
2  services:
3    omeroserver:
4      image: "omero-server-with-figure:5.6.5"
5      user: 'omero-server'
6      environment:
7        CONFIG_omero_db_host: database
8        CONFIG_omero_db_user: omero
9        CONFIG_omero_db_pass: omero
10       CONFIG_omero_db_name: omero
11       ROOTPASS: 0oPsPl41n73x7p4s5w0rd
```

# Outline

1 Version Control and Git

2 Gitlab

# Collaborative Work: Gitlab

The screenshot shows the GitLab web interface for a project named 'myhpcsaproject'. On the left is a sidebar with navigation links: Project information, Repository, Issues (0), Merge requests (0), CI/CD, Security & Compliance, Deployments, Packages and registries, Infrastructure, Monitor, Analytics, Wiki, Snippets, and Settings. The main content area shows the project details. At the top, it says 'test test > myhpcsaproject'. Below this is the project name 'myhpcsaproject' with a lock icon and 'Project ID: 35'. To the right are buttons for notifications, stars (0), forks (0), and a 'Clone' button. Below the project name, it shows '1 Commit', '1 Branch', '0 Tags', and '72 KB Project Storage'. A section for the 'Initial commit' shows the commit hash '69772da4' and the message 'test test authored 22 minutes ago'. Below this is a breadcrumb 'main > myhpcsaproject /' with a '+' button. There are several buttons for adding files: 'README', 'Auto DevOps enabled', 'Add LICENSE', 'Add CHANGELOG', 'Add CONTRIBUTING', 'Add Kubernetes cluster', and 'Configure Integrations'. A table lists the files in the repository:

Name	Last commit	Last update
📄 README.md	Initial commit	22 minutes ago

Below the table is the 'README.md' file content. It starts with the project name 'myhpcsaproject' and a section 'Getting started'. The text says: 'To make it easy for you to get started with GitLab, here's a list of recommended next steps. Already a pro? Just edit this README.md and make it your own. Want to make it easy? [Use the template at the bottom!](#)'. There is a section 'Add your files' with two options: 'Create or upload files' and 'Add files using the command line or push an existing Git repository with the following command:'.

# Collaborative Work: Issues

test test > myhpcproject > Issues > #2

☐ Open ☒ Issue created just now by Administrator

## Need to set up DB for Zyzzyx.

README

⬇️ Drag your designs here or click to upload.

**Tasks**

No tasks are currently assigned. Use tasks to break down this issue into smaller parts.

**Linked items**

Link issues together to show that they're related. [Learn more.](#)

### Activity

- Administrator changed due date to February 20, 2023 just now
- Administrator changed milestone to %Set up the Zyzzyx service just now
- Administrator added [enhancement](#) label just now
- Administrator assigned to [JohnD](#) just now

**Write** **Preview**

Write a comment or drag your files here...

Supports [Markdown](#). For [quick actions](#), type [/](#).

☐ Make this an internal note

**Assignee**

Administrator

**Labels**

[enhancement](#)

**Milestone**

Set up the Zyzzyx service

**Due date**

Feb 20, 2023 - remove due date

**Time tracking**

No estimate or time spent

**Confidentiality**

🔒 Not confidential

**Lock issue**

Unlocked

**Notifications** ☒

**1 participant**

Administrator

Reference: testname2/myhpcsa...

- An issue can be a problem with one of your services
- An issue can also be a task that you need to achieve, such as a new service or feature for an existing service
- This could be used to track and have a record of your own tasks and work

# Collaborative Work: Labels

test test > mytpcsaproject > Labels

All Subscribed Filter Name New label

Labels can be applied to issues and merge requests. Star a label to make it a priority label.

**Prioritized Labels**

Drag to reorder prioritized labels and change their relative priority.

Star labels to start sorting by priority

**Other Labels**

bug	test test / mytpcsaproject	Issues · Merge requests	☆	✎	⋮	Subscribe
confirmed	test test / mytpcsaproject	Issues · Merge requests	☆	✎	⋮	Subscribe
critical	test test / mytpcsaproject	Issues · Merge requests	☆	✎	⋮	Subscribe
discussion	test test / mytpcsaproject	Issues · Merge requests	☆	✎	⋮	Subscribe
documentation	test test / mytpcsaproject	Issues · Merge requests	☆	✎	⋮	Subscribe
enhancement	test test / mytpcsaproject	Issues · Merge requests	☆	✎	⋮	Subscribe
suggestion	test test / mytpcsaproject	Issues · Merge requests	☆	✎	⋮	Subscribe

- Labels help you categorize issues
- You can subscribe to labels and get notifications, or use them to create boards

# Collaborative Work: Milestones

test test &gt; myhpcsaoproject &gt; Milestones &gt; Set up the Zyzzyx service

Open Milestone

Edit

Close milestone

Delete

## Set up the Zyzzyx service

Milestone ID: 1

This instance of gitlab will work correctly.

Issues 2

Merge requests 0

Participants 1

Labels 2

Unstarted Issues (open and unassigned)

1

Document Zyzzyx

#1 documentation

Ongoing Issues (open and assigned)

1

Need to set up DB for Zyzzyx.

#2 enhancement

Completed Issues (closed)

0

0% complete

&gt;&gt;

Start date

No start date

Edit

Due date

No due date

Edit

Issues 2

Open: 2 Closed: 0

New issue

Time tracking

No estimate or time spent

Merge requests 0

Open: 0 Closed: 0 Merged: 0

Releases

None

Reference: testname2/myhpcsa...

- Collect tasks to achieve a specific goal
- Track progress of your goals

# Collaborative Work: Boards

test test &gt; myhpcsaproject &gt; Issue Boards

Development

Search

Q

Edit board

Create list

Open

2

+

Document Zyzzyx

documentation

#1

Need to set up DB for Zyzzyx.

enhancement

#2 Monday

enhancement

1

+

⚙

Title

Create issue

Cancel

Need to set up DB for Zyzzyx.

#2 Monday

Closed

0

■ Visualize issues at a glance

# Plenary Discussion

- Have you used version control before?
- Have you used feature and issue tracking?
- Would you use Gitlab's tools? Any other feature you would need?
- Any other tools that you use for this sort of work?
- Look-back at previous "best practices" presentation