GitLab

Lara Ianov, Ph.D.
Neurodevelopmental Bioinformatics Initiative (NBI)
Civitan International Research Center
University of Alabama at Birmingham

UAB
CIVITAN INTERNATIONAL
RESEARCH CENTER
Neurodevelopmental
Bioinformatics Initiative

Git

- Version control software
  - Tracks changes among all members
  - Maintains a history of all changes in an organized way and allows one to revert changes
  - A higher order method over the standard ”copy and change name method”: copy a file with changes to it and name it differently or place it in time-stamped directories - usually these scripts are filled with unnecessary comments documenting the changes
  - A git repository contains the full history of the project. Each clone will always mirror the full repository. Thus, git is a Distributed Version Control System

GitLab, GitHub, Bitbucket etc.

- Web-based services which can be used to host your git repositories in a streamlined manner. Here are some differences among them:

Free Plans	Public Repos	Private Repos	Collaborators	Storage Space	Hosting	Support
GitHub Public	Unlimited	* 0	Unlimited	N / A	Cloud	Email / Forum
Bitbucket Small teams	Unlimited	Unlimited (1Gb /project)	5	N / A	Cloud	Email / Forum
GitLab Cloud Hosted	Unlimited	Unlimited (10Gb / Project)	Unlimited	Unlimited	Cloud	Forum
GitLab Community Edition	Unlimited	Unlimited	Unlimited	N / A	Self-hosted	Forum

UAB RC hosts

* As a student (with .edu account), you can have private repos for free with access to "student developer plan". Details at: https://help.github.com/articles/applying-for-a-student-developer-pack/

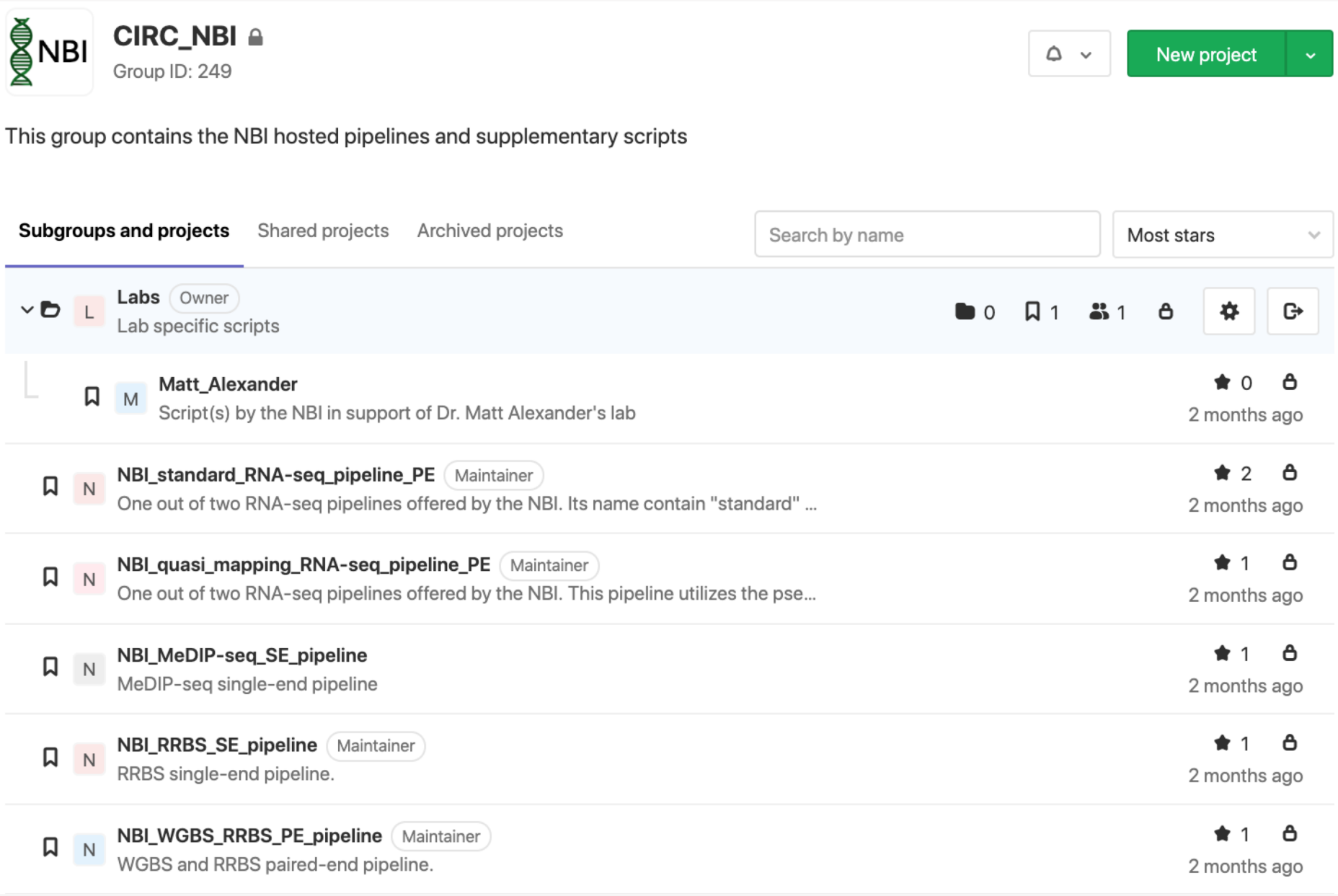
Table source:
https://medium.com/flow-ci/github-vs-bitbucket-vs-gitlab-vs-coding-7cf2b43888a1

GitLab

- Sign-in with Blazer ID and password: https://gitlab.rc.uab.edu/users/sign_in
- Learning sources:
  - Git for beginners from UAB RC: https://docs.uabgrid.uab.edu/wiki/Git_For_Beginners
  - GitLab tutorial: https://www.tutorialspoint.com/gitlab/index.htm
  - GitLab docs: https://docs.gitlab.com/
  - Atlassian's tutorial: https://www.atlassian.com/git/tutorials/what-is-version-control
  - Google ☺
- **Today's focus**: GitLab's graphical interface (with some emphasis on important commands)
- Basic git terminology:
  - `Clone`: copy a repository (repo)
  - `Commit`: record changes to the repo
  - `Push`: updates remote git repo with committed changes (if working outside of GitLab environment)
  - `Branch`: independent area of development from the primary set/branch
  - … and more

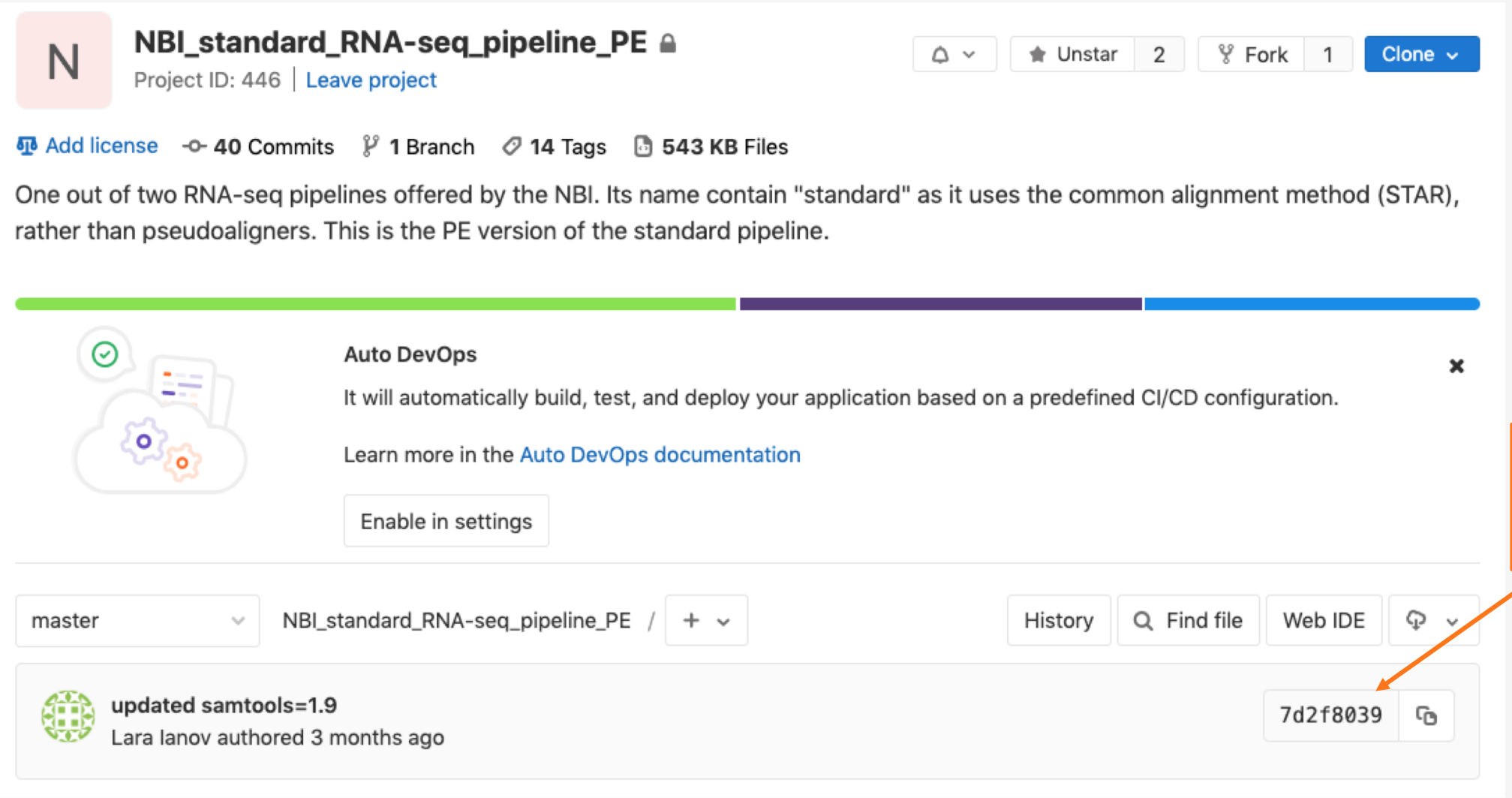
UAB
CIVITAN INTERNATIONAL
RESEARCH CENTER
Neurodevelopmental
Bioinformatics Initiative

GitLab groups



GitLab "groups" are ideal for individual labs/cores

- Unlimited number of projects (there is a limit in your own profile)
- Easy to collaborate/share



SHA-1 hash is essential to git's method of version control

Access can be given to:

1. Specific user(s)
2. Everyone at UAB
3. Globally

They may also open "issues" with requests to add specific features etc.

UAB CIVITAN INTERNATIONAL RESEARCH CENTER
Neurodevelopmental Bioinformatics Initiative

Project members

You can invite a new member to **NBI_standard_RNA-seq_pipeline_PE** or invite another group.

Invite member | Invite group

Select members to invite

Search for members to update or invite

Choose a role permission

Guest

Read more about role permissions

Access expiration date

Expiration date

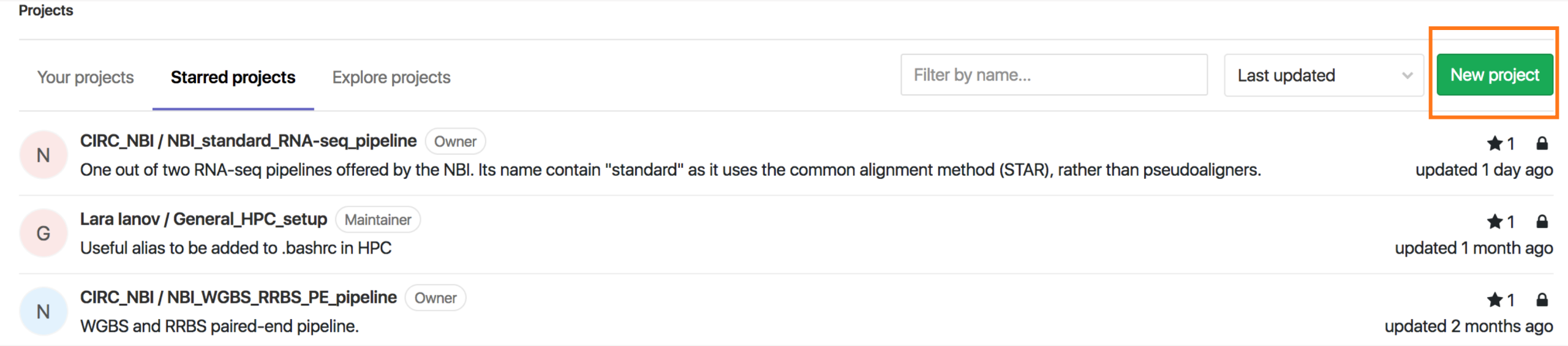
Add to project | Import

See more member permissions at: https://docs.gitlab.com/ee/user/permissions.html

Action	Guest	Reporter	Developer	Maintainer	Owner
Create new issue	✓ [1]	✓	✓	✓	✓
Create confidential issue	✓ [1]	✓	✓	✓	✓
View confidential issues	(✓) [2]	✓	✓	✓	✓
Leave comments	✓ [1]	✓	✓	✓	✓
See related issues	✓	✓	✓	✓	✓
See a list of jobs	✓ [3]	✓	✓	✓	✓
See a job log	✓ [3]	✓	✓	✓	✓
Download and browse job artifacts	✓ [3]	✓	✓	✓	✓
View wiki pages	✓ [1]	✓	✓	✓	✓
View license management reports	✓ [1]	✓	✓	✓	✓
View Security reports	✓ [1]	✓	✓	✓	✓
Pull project code	[1]	✓	✓	✓	✓
Download project	[1]	✓	✓	✓	✓
Assign issues		✓	✓	✓	✓
Assign merge requests			✓	✓	✓
Label issues		✓	✓	✓	✓
Label merge requests			✓	✓	✓
Create code snippets		✓	✓	✓	✓
Manage issue tracker		✓	✓	✓	✓
Manage labels		✓	✓	✓	✓

GitLab – Starting a repository

- From GitLab:



UAB
CIVITAN INTERNATIONAL
RESEARCH CENTER
Neurodevelopmental
Bioinformatics Initiative

Blank project | Create from template | Import project

Project name

My awesome project

Project URL

https://gitlab.rc.uab.edu/ lianov

Project slug

my-awesome-project

Want to house several dependent projects under the same namespace? Create a group.

Project description (optional)

Description format

Path may be user profile or specific groups (e.g.: “CIRC_NBI”)

Visibility Level

- Private
  Project access must be granted explicitly to each user.
- Internal
  The project can be accessed by any logged in user.
- Public
  The project can be accessed without any authentication.

Initialize repository with a README

Allows you to immediately clone this project’s repository. Skip this if you plan to push up an existing repository.

Create project

Cancel

- From the command line:
  - Very nice when uploading a directory of existing files

Command line instructions

Git global setup

```
git config --global user.name "Lara Ianov"
git config --global user.email "lianov@uab.edu"
```

Create a new repository

```
git clone git@gitlab.rc.uab.edu:lianov/gitlab_demo.git
cd gitlab_demo
touch README.md
git add README.md
git commit -m "add README"
git push -u origin master
```

Or type a one-liner to stage and commit all **tracked** files:

```
git commit -a -m "message goes here"
```

Existing folder

```
cd existing_folder
git init
git remote add origin git@gitlab.rc.uab.edu:lianov/gitlab_demo.git
git add .
git commit -m "Initial commit"
git push -u origin master
```

Commands to upload files when initial project (in this case named "gitlab_demo") was created in GitLab

Existing Git repository

```
cd existing_repo
git remote rename origin old-origin
git remote add origin git@gitlab.rc.uab.edu:lianov/gitlab_demo.git
git push -u origin --all
git push -u origin --tags
```

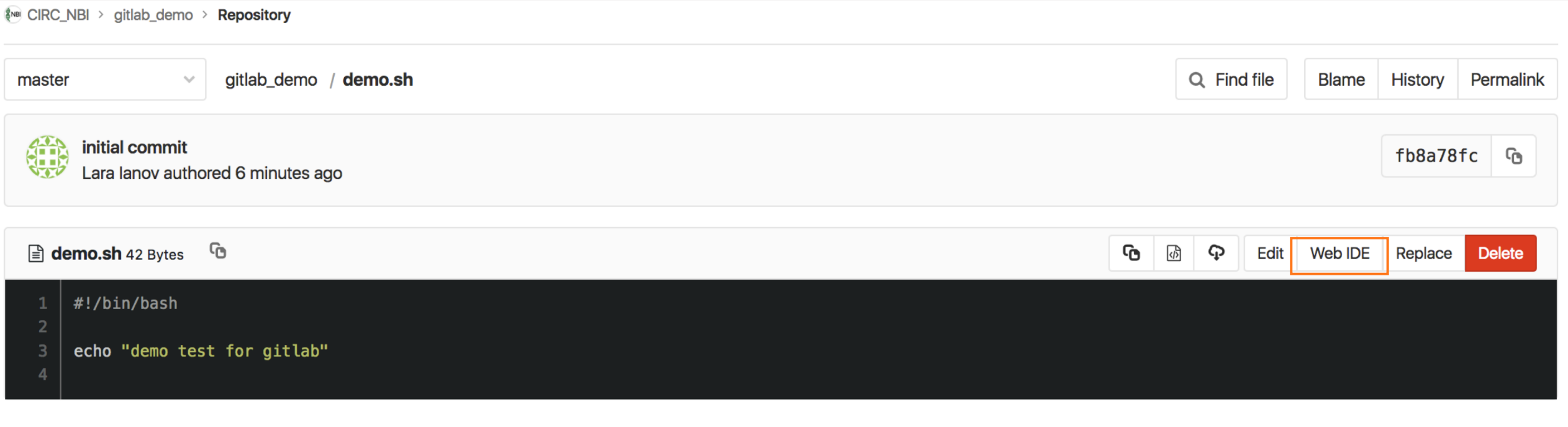
- From the command line:
  - Or create the project directly from the command line and push all files to the project

```
cd existing_folder
git init
git add .
git commit –m “initial commit message”

# With SSH
git push -u git@gitlab.rc.uab.edu:<namespace>/<project_name>.git master
git remote add origin git@gitlab.rc.uab.edu:<namespace>/<project_name>.git

# OR with HTTPS
git push -u https://gitlab.rc.uab.edu/<namespace>/<project_name>.git master
git remote add origin https://gitlab.rc.uab.edu/<namespace>/<project_name>.git
```

FYI on setting up SSH for GitLab: https://docs.gitlab.com/ee/ssh/README.html

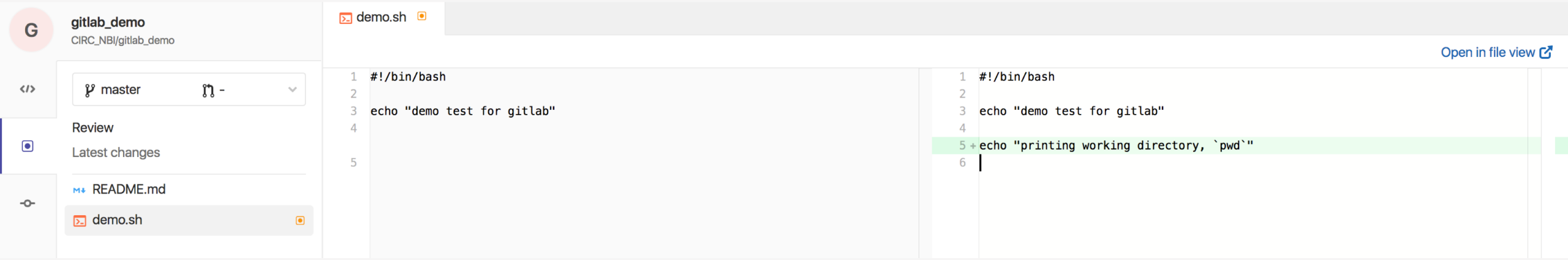
GitLab – Edit and commit

- GitLab contains an integrated development environment (IDE) – OK for small edits, but I do not use it for large projects/changes.



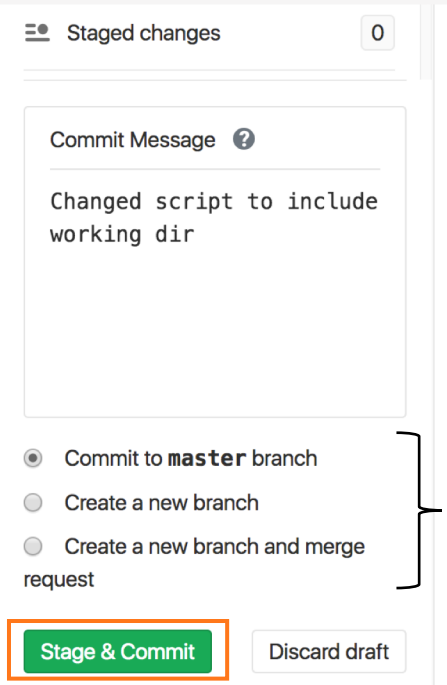
UAB
CIVITAN INTERNATIONAL RESEARCH CENTER
Neurodevelopmental Bioinformatics Initiative

GitLab's IDE



Hit "Stage and Commit" and include a message. Same as `git add . && git commit -m "message"`

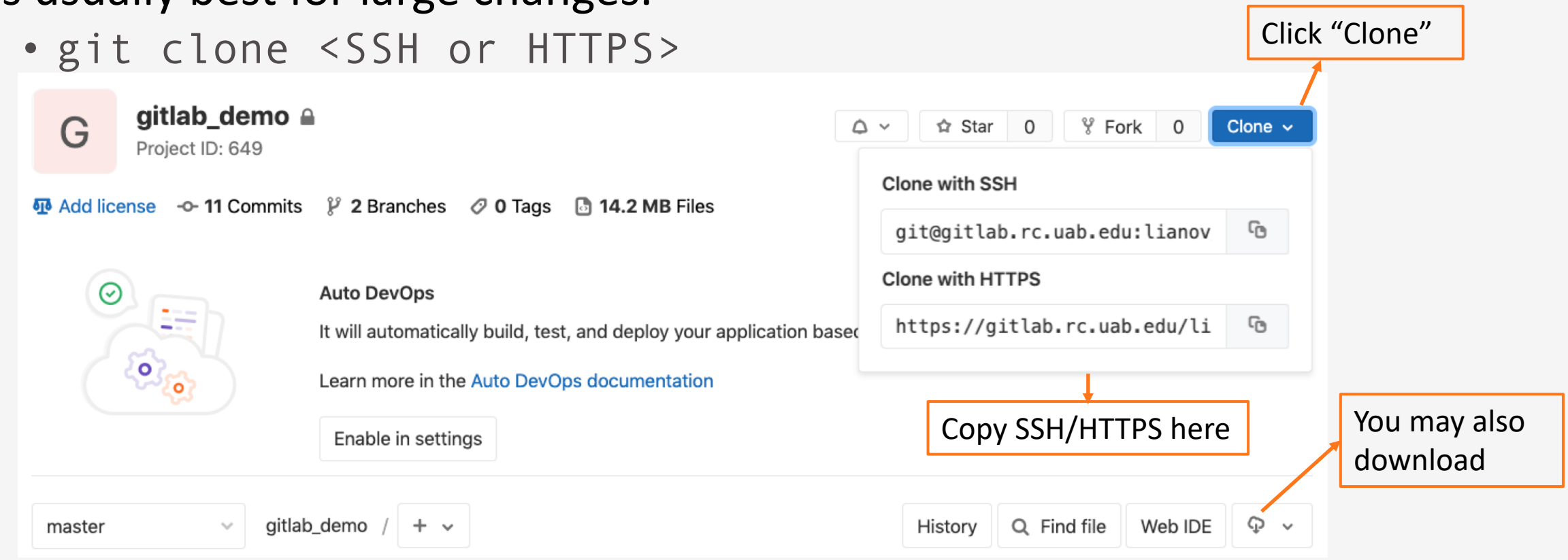
No need to `push` as this is being done directly in GitLab.



Note, the option of committing into the master branch or a new branch. A new branch would be wise if you are still testing the code or to work among collaborators where you would send ("create a merge request") the reviewed code to the project's maintainer.

GitLab – Edit and commit

- ✓GitLab contains an integrated development environment (IDE) – OK for small edits, but I do not use it for large projects/changes.
- Cloning the repo to your local computer and using your favorite editor is usually best for large changes.
  - `git clone <SSH or HTTPS>`



GitLab – Edit and commit

- ✓GitLab contains an integrated development environment (IDE) – OK for small edits, but I do not use it for large projects/changes.
- Cloning the repo to your local computer and using your favorite editor is usually best for large changes.
  - `git clone <SSH or HTTPS>`

demo.sh

```
1  #!/bin/bash
2
3  echo "demo test for gitlab"
4
5  echo "printing working directory, `pwd`"
6
7  echo "line added locally after cloning, and then committed and pushed to GitLab"
8
```

Line added in my editor of choice in my computer and then:

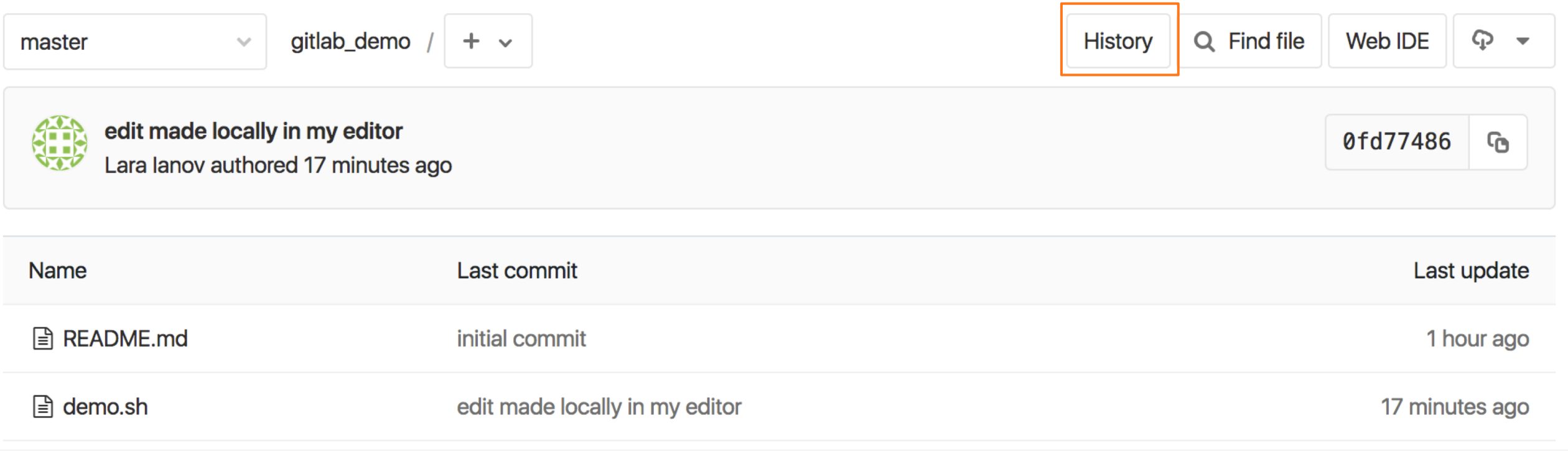
```
git add .
git commit -m “edit made locally in my editor”
git push -u origin master
```

GitLab – Atom integration

- Atom is a great editor for both GitHub and GitLab as it integrates the git commands, allowing you to perform tasks such as `commit` and `push` directly from the editor. Explore Atom's packages such as this to learn more

GitLab – History

- Regardless of where changes were made (in GitLab IDE or locally and then pushed), all changes are easily tracked in GitLab:



FYI command line version: `git log`

master | gitlab_demo | Filter by commit message

28 Nov, 2018 3 commits

edit made locally in my editor
Lara Ianov authored 25 minutes ago — `0fd77486`

Changed script to include working dir
Lara Ianov authored 1 hour ago — `0917c098`

initial commit
Lara Ianov authored 2 hours ago — `fb8a78fc`

Note the changes in SHA-1

edit made locally in my editor

FYI command line version: `git show <SHA-1>`

parent `0917c098` `master`

No related merge requests found

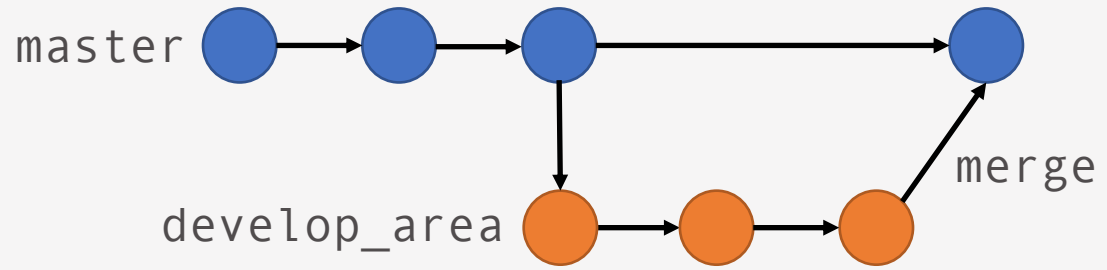
Showing 1 changed file with 2 additions and 0 deletions | Hide whitespace changes | Inline | Side-by-side

demo.sh 100644 → 100755 | View file @ `0fd77486`

```
...  @@ -3,3 +3,5 @@
3    echo "demo test for gitlab"
4
5    echo "printing working directory, `pwd`"
```

```
...  @@ -3,3 +3,5 @@
3    echo "demo test for gitlab"
4
5    echo "printing working directory, `pwd`"
6  +
7  + echo "line added locally after cloning, and then committed and pushed to GitLab"
```

GitLab - Branch



- By default, all projects start with one branch, called `master`. All commits I have done in this demo so far have been made to `master`.
- But what if you would like to work on the repo without committing to the main branch? Maybe testing code without overwriting the `master` branch?
  - This is where `branching` is very useful, especially when more than one person is working on the same code.
  - In the case of more than one person, a collaborator can branch, edit independently at the isolated branch and when finalized, send a `merge` request to the maintainer who can approve to merge all changes into the `master` branch.

UAB
CIVITAN INTERNATIONAL
RESEARCH CENTER
Neurodevelopmental
Bioinformatics Initiative

G

gitlab_demo

Project ID: 936

Star 0 | Fork 0 | Clone

Add license | 1 Commit | 1 Branch | 0 Tags | 2.1 MB Files

Lara Ianov > gitlab_demo > Branches

Overview | Active | Stale | All

Filter by branch name | Delete merged branches | New branch

Protected branches can be managed in project settings.

Active branches

master default protected

25e27b03 · Changes made in Atom and committed etc. directly · 23 hours ago

Lara Ianov > gitlab_demo > **New Branch**

New Branch

Branch name: develop_area

Create from: master

Existing branch name, tag, or commit SHA

Create branch

Cancel

Lara Ianov > gitlab_demo > **Details**

G

gitlab_demo

Project ID: 649

Star 0 Fork 0 Clone

Add license | **11** Commits | **2** Branches | **0** Tags | **14.2 MB** Files

You can now edit the code of the second branch and click on “branches” to compare and add a merge request

Lara Ianov > gitlab_demo > **Branches**

Overview Active Stale All

Filter by branch name | Delete merged branches | New branch

Protected branches can be managed in project settings.

Active branches

develop_area
caeed48f · Update demo.sh · 2 minutes ago
0 1 | Merge request | Compare

master default protected
25e27b03 · Changes made in Atom and committed etc. directly · 23 hours ago

Lara Ianov > gitlab_demo > Compare Revisions > **master...develop_area**

Source `develop_area` ... Target `master` Compare | Create merge request

Commits (1)

Update demo.sh
Lara Ianov authored 3 minutes ago

caeed48f

Showing 1 changed file with 3 additions and 0 deletions

Hide whitespace changes | Inline | Side-by-side

demo.sh

View file @ caeed48f

```
...  @@ -10,3 +10,5 @@ echo "line added locally after cloning, and then committed and pushed to GitLab"
10   # changes below were made in Atom, staged, committed and pushed direct ly from Atom
11   echo "printing current date and time"
12   date
```

```
...  @@ -10,3 +10,5 @@ echo "line added locally after cloning, and then committed and pushed to GitLab"
10   # changes below were made in Atom, staged, committed and pushed direct ly from Atom
11   echo "printing current date and time"
12   date
13  +
14  + #----- This represents finalized code ready to be merged --------
     \ No newline at end of file
```

GitLab - Branch

- Again, using GitLab's IDE may be OK for small changes but I prefer local for larger changes:

```
git clone <SSH or HTTPS> # or git pull
cd <cloned_dir> # only needed if cloned
git branch -a
git checkout <branch_name> (switches the branch)
### modify code ###
git commit -am "message goes here" (adding in one-liner)
git push -u origin <branch_name>
```

- Atom's packages offer good features to do this as well

UAB
CIVITAN INTERNATIONAL
RESEARCH CENTER
Neurodevelopmental
Bioinformatics Initiative

Thanks!

A big thank you to UAB Research Computing for hosting GitLab and for taking the time to provide additional documentation and training

My contact info:

lianov@uab.edu

CIRC 252C | P: 205-996-5871

NBI

UAB CIVITAN INTERNATIONAL RESEARCH CENTER
Neurodevelopmental Bioinformatics Initiative