CSCI 4152/6509
Natural Language Processing

Lab 4:

Git and GitLab Tutorial

Lab Instructor: Sigma Jahan and Tymon Wranik-Lohrenz

Faculty of Computer Science

Dalhousie University

Lab Overview

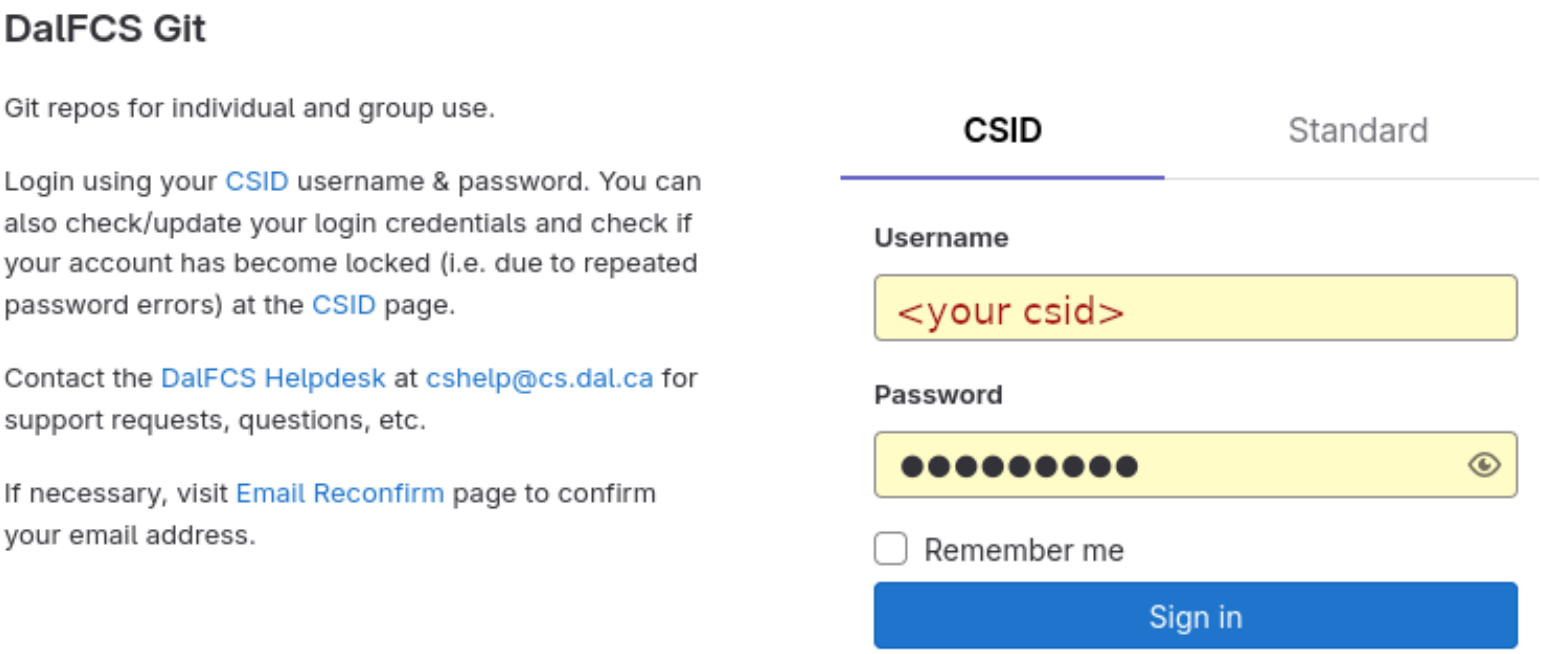
- GitLab Web interface
- How to checkout projects in Git
- Adding and deleting files and directories to Git and GitLab
- Committing and pushing your changes
- Checking out previous commits
- Elements of collaborative work: creating branches
- Merging branches and resolving conflicts

What is GitLab?

- It is based on Git, a source version control system
- A source version control system is used
  - to store and manage different versions of code
  - to provide collaborative platform for software developers
- GitLab is based on Git and provides a web interface
- Similar to GitHub in this sense
- Provides Continuous Integration (CI) and Continuous Delivery (CD) of code
- A lot of material on Git and GitLab can be found on Web

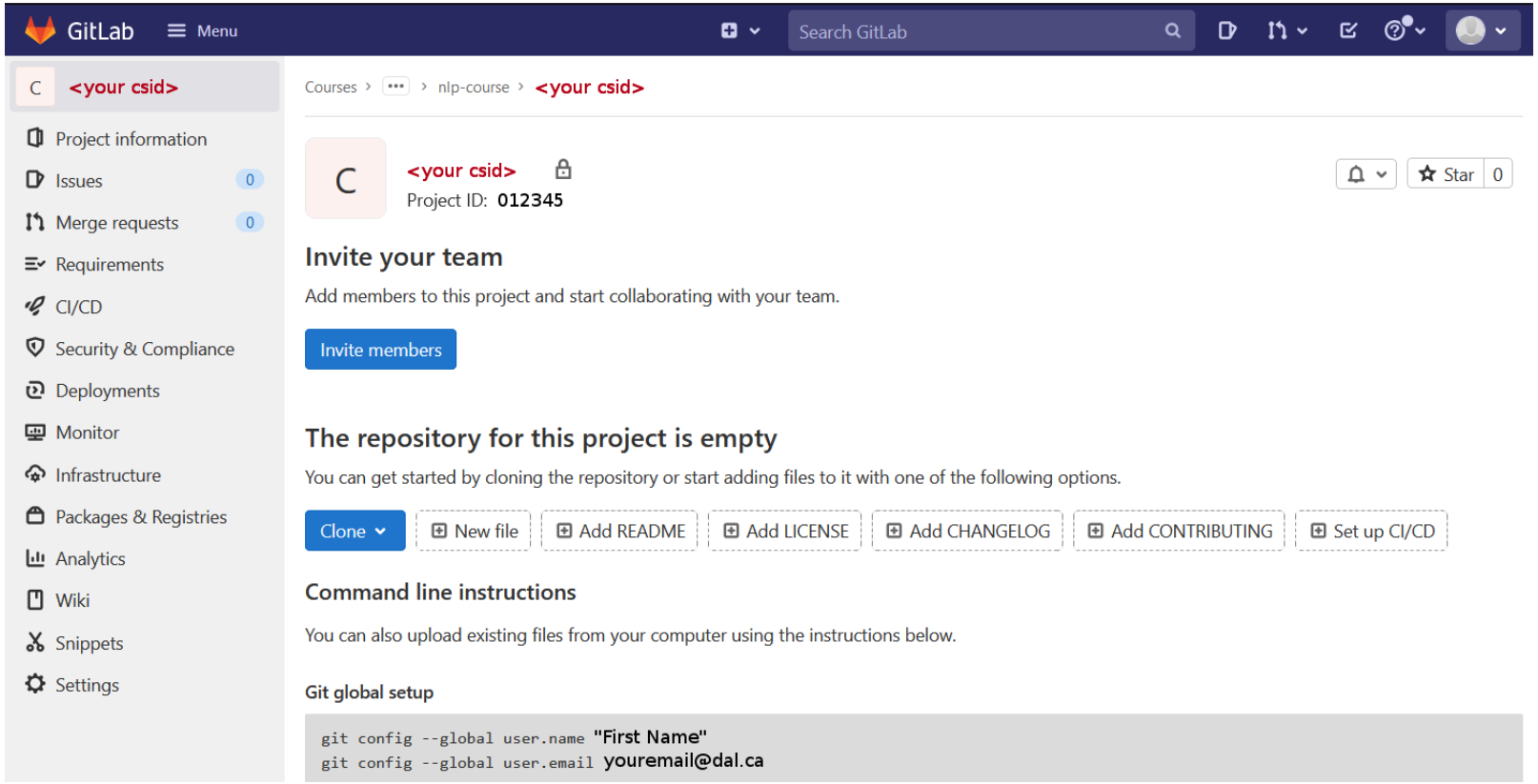
Step 1. Logging into DalFCS GitLab Website

- Open your Web browser and go to: `https://git.cs.dal.ca`



Find Your Course GitLab Project (Repository)

- It is named as your CSID and in NLP course group (`2024-fall/nlp-course/<your_csid>`)
- URL: `https://git.cs.dal.ca/courses/2024-fall/nlp-course/<your_csid>`



Step 2: Creating a README File in GitLab

- Click button “Add README”
- Enter the content given in the notes, starting with

```
# CSCI 4152/6509 Natural Language Processing
                                        (Fall 2024)
## GitLab Course Repository

### Student Details
...
```

- and so on. Enter your name, CSID, and so one where requested
- Commit the changes and observe how README file looks like in the browser
- `README.md` must be in the GitLab repository

Step 3: Logging in to server

`timberlea`

- Login to the server `timberlea`
- Change directory to `csci4152` or `csci6509`
- `mkdir lab4` and `cd lab4`
- Check your current directory with `pwd`
- This is the directory where you should keep files from this lab.

Step 4: Using HTTPS Address in Git

Step 4-a: Find GitLab repository address

Step 4-b: Clone Repository via HTTPS

- Verify your cloned repository
- Change to directory `<your_csid>` and use `ls` to verify that the file `README.md` is there

Step 5: Prepare and Submit Public Key

Step 5-a: Create Directory and Check Keys

```
mkdir lab4g
cd lab4g
ls ~/.ssh/
```

Step 5-b: Generate Keys If Needed

- Before generating keys, **read the security note in the lab notes**
- If we need to generate the keys:

```
ssh-keygen -t rsa
```

- Press Enter to all questions
- Check that keys are generated:

```
ls ~/.ssh/
```

- Copy the public key to `lab4g` directory

```
cp ~/.ssh/id_rsa.pub .
```

Step 5-c: Adding Files in Git

- Commit the key file locally

```
git add id_rsa.pub
git commit -m'Commit id_rsa.pub'
```

- 'Push' (save) the file to GitLab repository

```
git push -u origin main
```

- Use Web browser to find your key file in the GitLab
- Directory `lab4g` and file `id_rsa.pub` must be in the GitLab repository
- Step 5 is completed

Step 6: Setting up SSH Key in GitLab

- In Web browser GitLab page open `id_rsa.pub`
- Click on double-square in corner ("Copy file contents")
- Open "SSH Keys", paste your key, edit, save
- Ready to clone the repository using SSH without password

Step 7: Clone with SSH

- Go to the `timberlea` terminal and check your working directory; you need to be in `lab4` directory
- Rename your previously cloned directory to `<your_csid>-https`
- Clone the repository using SSH
- Remove your previously cloned directory

Step 8: Preparing Files `explore.pl` and Shakespeare

- Check your directory
- Change your directory to `lab4g`
- Prepare `explore.pl`
- Copy the part of Shakespeare's Hamlet to your current directory
- Test the program

Step 9: Commit Files explore.pl and Shakespeare

- Add files `explore.pl` and Shakespeare to Git
- Check with `git status`
- Commit the files version 1.0
- Edit the file `explore.pl`
- Add and commit `explore.pl` version 1.1

Step 10: Explore Previous Commits

- Run and examine `git log`
- Checkout previous commits by using SHA-1 checksums
- Checkout the latest version

Step 11: Push Changes to GitLab

- Check GitLab repository: no files yet
- Push files
- Check GitLab repository in Web browser: files are there
- GitLab should contain `explore.pl` and the Hamlet part by now

Step 12: Creating Branches: Directories Preparation

- Prepare directory `ada`
- Clone and prepare directory `bob`
- Check with `ls` contents of the directory

Step 13: Creating Ada's Branch

- Create and checkout branch `ada-main-program`
- Modify `explore.pl` version 1.2
- Test the program
- Add, commit, and push the changes

Step 14: Creating Bob's Branch

- Go to Bob's directory
- Create and checkout branch `bob-function-explore`
- Modify `explore.pl` to be version `1.2 bob`
- Test the program
- Edit and save `report.txt`
- Add, commit, and push Bob's branch
- Check changes in GitLab browser page

Step 15: Ada Merges Her Branch

- Go to Ada's directory and pull any changes
- Merge Ada's branch
- Push changes

Step 16: Bob Merges His Branch

- Go to Bob's directory and pull any changes
- Merge Bob's branch into `main`
- Resolve the conflict in `explore.pl` manually
- Add and commit changes, and check
- Push changes to GitLab (`origin`)
- Go to Ada's directory and pull the changes
- GitLab should contain all three branches up to date

This is the end of Lab 4.