

**Faculty of Computer Science, Dalhousie University**

*12-Feb-2026*

## **DGIN 5201 — Digital Transformation**

### **Lecture 10: Source Version Control**

Location: McCain 2170      Instructor: Vlado Keselj  
Time: 10:05–11:25

#### **Previous Lecture**

- Review of e2 form
  - touch typing
  - text editors, emacs
- Web server Apache
- Available material (e3)
- Concepts and tools
  - Iterative prototyping and rsync
  - Apache options
- Password protection (e4)
- Unix-style customization
  - .htaccess, .htpasswd, .bashrc, .emacs

#### **Announcements: Assignment 2**

- A2 part 1: by tomorrow (Fri 13-Feb-2026) 11:59pm
  - A2 part 1: e1 and e2 on timberlea
- A2 part 2: by Fri 27-Feb-2026
  - A2 part 2: what we finish this week, will be specified
- A2 part 3: by Fri 6-Mar-2026
  - A2 part 3: to be completed after reading week
  - will be specified

#### **Heads-up: Second Half of the Term**

- 1 lab and 1 lecture to finish CS part from 1st half
- Seminars and project meetings
- Team meetings with instructors
  - 20 min formative meetings
  - check schedule on Brightspace Week 7 to 12

#### **Source Version Control**

- Before continuing with e5 etc. let us see how to submit e1, e2, e3, e4 to GitLab
- We will use this as an introduction to git, GitLab, GitHub
- git is a Source Version Control tool
- We already have used making several incremental copies of the code
  - or may use saving a file version as a backup for later
  - saving relatively often, not to lose code accidentally

## What is GitLab?

*Slide notes:*

### 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 the Web

Most software development companies use some system for source version control which enables efficient storage and retrieval of different versions of the source code being developed and a way for different developers to collaboratively work on it. By collaborative work, we mean that they can write and test code independently at the same time, and there is a way to merge their changes easily in the final project. Git is an example of such version control system, which is widely used and very popular. Another example of such system is Subversion, or SVN for short. GitLab is a web-based system which is based on the Git source version control system and provides some additional functionality in a web-app style, similarly to another popular on-line platform — GitHub. Both GitHub and GitLab provide a Web interface to access and manage your Git repository.

GitLab is a Web-based DevOps platform, delivered as a single application, that provides a Git-repository manager providing wiki, issue-tracking and CI/CD pipeline features, using an open-source license, and developed by *GitLab Inc.* Continuous Integration (CI) is an established process to continuously provide integration of written code provided by a team in a shared repository. Developers share the new code in a Merge Request, also called a Pull Request. The request triggers a pipeline to build, test, and validate the new code prior to merging the changes within the repository. Continuous Delivery (CD) ensures the delivery of CI validated code to the app by means of a structured deployment pipeline. In general, CI helps the developers to catch and reduce bugs early in the development cycle, and CD moves verified code to the applications faster.

### Some References

In this tutorial you should learn basic elements of Git and GitLab. For more information and tutorials on GitLab, you can refer to the official documentation:

<https://docs.gitlab.com/ee/README.html>

The following is a great interactive on-line tutorial for basic Git operations:

<https://learngitbranching.js.org>

We will start this hands-on lab of working with the Dal FCS GitLab site, and also using your `timberlea` account to submit your assignment files using Git.

### Step 1. Logging into DalFCS GitLab Website

The Faculty of Computer Science (FCS) at Dalhousie provides an open-source version of GitLab, which we will use in this tutorial.

Open your Web browser and go to the Dalhousie FCS GitLab web site: <https://git.cs.dal.ca> You should be able to see the Login screen, as shown in Figure 1. Login with your CSID login and password.

A user of GitLab can participate in different *projects* and have different roles in them. All participants of a project

### DalFCS Git

Git repos for individual and group use.

Login using your [CSID](#) username & password. You can also check/update your login credentials and check if your account has become locked (i.e. due to repeated password errors) at the [CSID](#) page.

Contact the [DalFCS Helpdesk](#) at [cshelp@cs.dal.ca](mailto:cshelp@cs.dal.ca) for support requests, questions, etc.

If necessary, visit [Email Reconfirm](#) page to confirm your email address.

**CSID**
Standard

---

**Username**

<your csid>

**Password**

●●●●●●●● 👁

Remember me

Sign in

Figure 1: Dal GitLab login screen.

are called *members*. A member's role in a project can be: *Owner*, *Maintainer*, *Developer*, *Reporter*, or *Guest*. We will refer to the Dalhousie FCS GitLab installation as a repository of these projects, but we will also refer sometimes to your project as the repository. Since we use the term *course project* as the part of your coursework, we hope that this will not be confused with referring to the concept of *GitLab project*, which we will call sometimes a *repository* as well. It should always be clear from the context to which concept we are referring to.

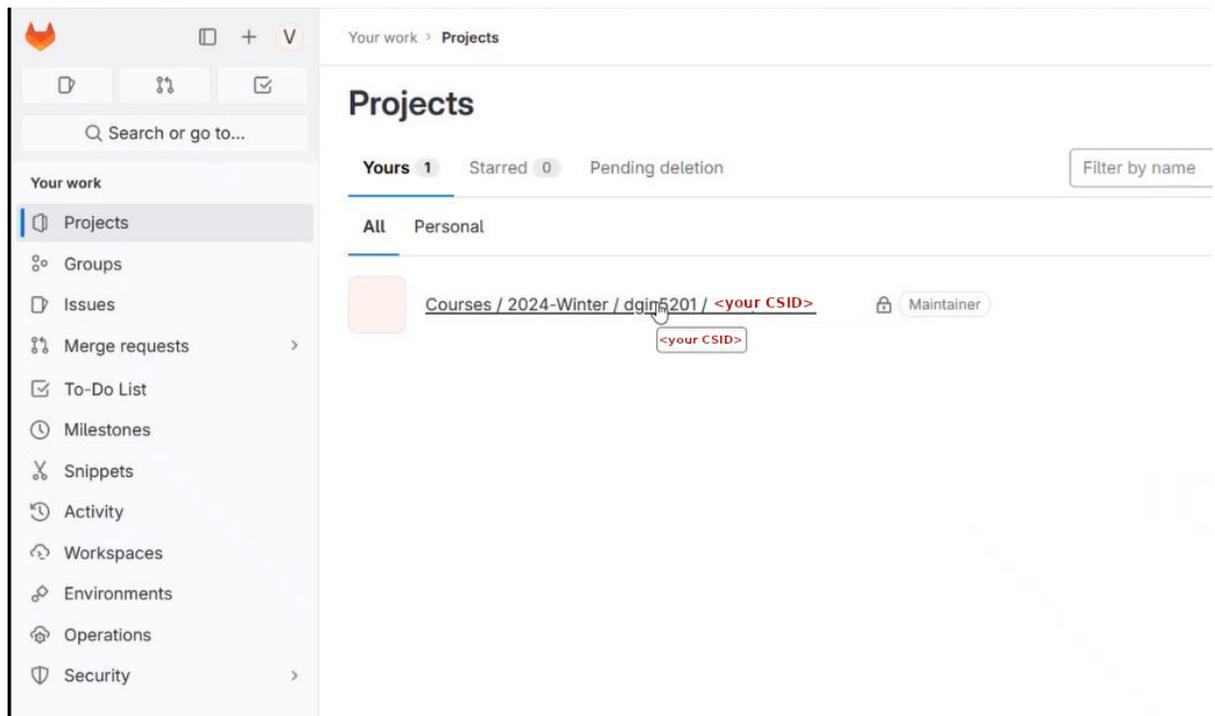
### Step 2. Find your CSID Course Project in DGIN5201 Group

Since you can be a member of different projects in GitLab, you first need to find the project that is assigned to you within this course. This project has the same name as your CSID and it is within the course project group, within this term. In order to find it you can use the "Projects" or "Groups" menu option in the GitLab Web interface, or directly type in the URL of the project.

The following screenshots shows how the page which displays your projects may look like:

### Approximate Page with Your Projects

- an approximate look of the page with your projects



Once you find it, the browser should show the following URL:

`https://git.cs.dal.ca/courses/2026-winter/dgin-5201/<your.csid>` where `<your.csid>` is your CSID. Figure 2 shows how the front page of this project should look like, approximately, with some possible minor differences.

Courses / 2025-Winter / DGIN-5201 / <your\_CSID>

You can't push or pull repositories using SSH until you add an SSH key to your profile.

[Add SSH key](#) [Don't show again](#)

**Auto DevOps**  
Automatically build, test, and deploy your application based on a predefined CI/CD configuration. Learn more in the Auto DevOps documentation.  
[Enable in settings](#)

<your\_CSID> [Code](#)

**The repository for this project is empty**  
To get started, clone the repository or upload some files.

**Command line instructions**  
You can also upload existing files from your computer using the instructions below.

**Configure your Git identity**  
Get started with Git and learn how to configure it.

Local Global

**Git local setup**  
Configure your Git identity locally to use it only for this project:

```
git config --local user.name "<your name>"
git config --local user.email "<your email>"
```

**Add files**  
Push files to this repository using SSH or HTTPS. If you're unsure, we recommend SSH.

SSH HTTPS

[How to use SSH keys?](#)

**Create a new repository**

```
git clone git@git.cs.dal.ca:courses/2025-winter/dgin-5201/<your CSID>.git
cd <your CSID>
git switch --create main
touch README.md
git add README.md
git commit -m "add README"
git push --set-upstream origin main
```

**Push an existing folder**

```
cd existing_folder
git init --initial-branch=main
git remote add origin git@git.cs.dal.ca:courses/2025-winter/dgin-5201/<your CSID>.git
git add .
git commit -m "Initial commit"
git push --set-upstream origin main
```

**Push an existing Git repository**

```
cd existing_repo
git remote rename origin old-origin
git remote add origin git@git.cs.dal.ca:courses/2025-winter/dgin-5201/adm/<your CSID>.git
git push --set-upstream origin --all
git push --set-upstream origin --tags
```

**Project information**

**Invite your team**  
Add members to this project and start collaborating with your team.  
[Invite members](#)

**Upload File**

- + New file
- + Add README
- + Add LICENSE
- + Add CHANGELOG
- + Add CONTRIBUTING
- + Set up CI/CD
- + Add Wiki
- + Configure Integrations

**Created on**  
January 23, 2025

Figure 2: Front page of your GitLab course repository

### Step 3: Uploading your Files from timberlea

- In this step you should upload your lab files from timberlea into the GitLab server
- The instructions are shown in the GitLab page
- You should also open another command-line window for ssh login to timberlea
- **Note:** The following slides will have some incorrect information based on previous years. I will point out differences and correct them after class.

The instructions that we will use to upload the lab files (i.e., the directories e1, ... e4), are shown in the project page:

### Instructions to Upload our Files

The screenshot shows the GitLab interface for a project named 'vladodemo2'. The main content area displays three sections of instructions:

- Create a new repository:**

```
git clone https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vladodemo2.git
cd vladodemo2
git switch --create main
touch README.md
git add README.md
git commit -m "add README"
git push --set-upstream origin main
```
- Push an existing folder:** (This section is circled in red in the image)

```
cd existing_folder
git init --initial-branch=main
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vladodemo2.git
git add .
git commit -m "Initial commit"
git push --set-upstream origin main
```
- Push an existing Git repository:**

```
cd existing_repo
git remote rename origin old-origin
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vladodemo2.git
git push --set-upstream origin --all
git push --set-upstream origin --tags
```

### Login to timberlea Server

- In another window, we login to timberlea server
- It will probably be a command-line window or terminal window in which we type the command:

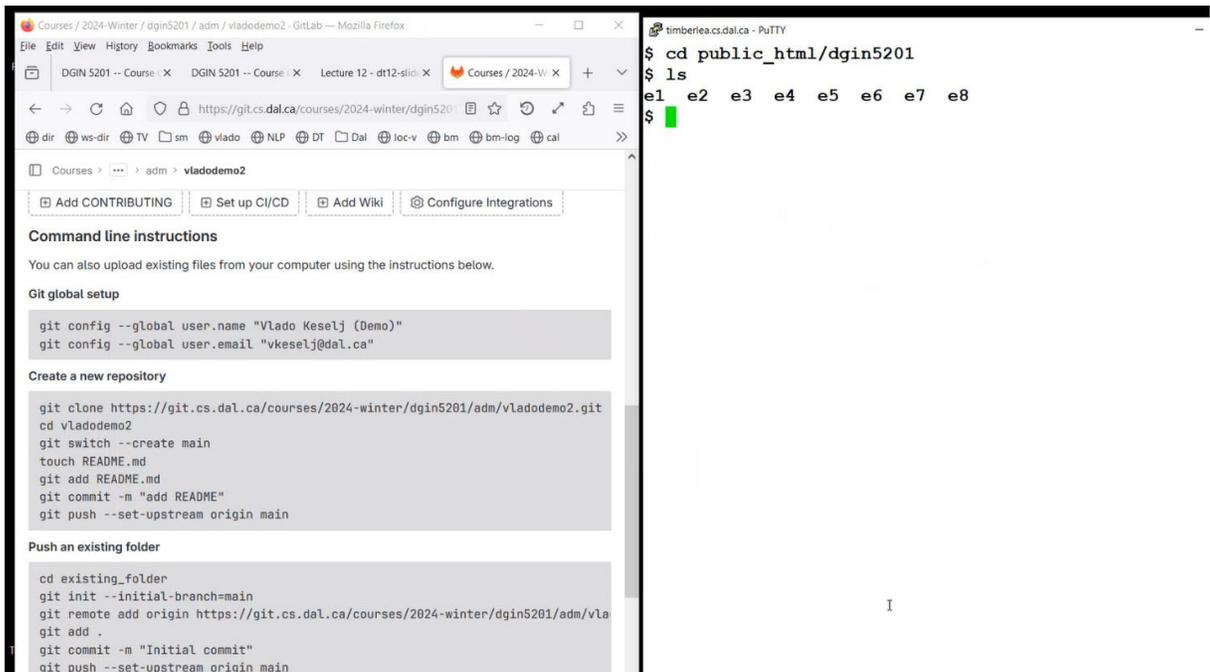
```
ssh <your_csid>@timberlea.cs.dal.ca
```

where instead of <your\_csid> you should use your own CSID

- or maybe you can use PuTTY, mobaxterm, or other SSH application
- You should try to have two windows: the web browser with GitLab, and the command-line window

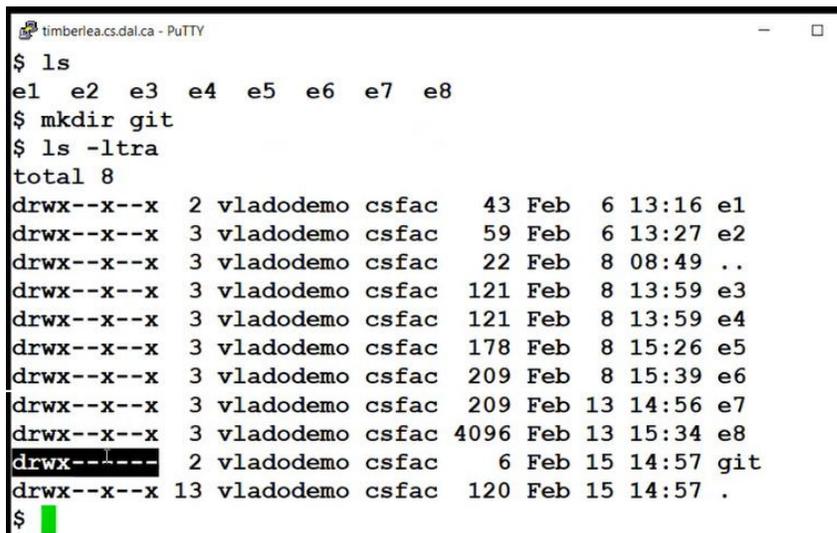
### Going to your Labs Directory

- Change directory to public.html/dgin5201 and list contents
- your screen with two windows could look as follows:



### Create Directory git

- make directory git
- make sure that the directory has permissions rwx for for the user only (rwx-----)
- the next screenshot shows how it would look on my screen



### Copy directories e1...e4 into directory git

- We first `rsync` directory e1:

```

timberlea.cs.dal.ca - PuTTY
$ ls
e1 e2 e3 e4 e5 e6 e7 e8
$ mkdir git
$ ls -ltra
total 8
drwx--x--x  2 vladodemo csfac   43 Feb  6 13:16 e1
drwx--x--x  3 vladodemo csfac   59 Feb  6 13:27 e2
drwx--x--x  3 vladodemo csfac   22 Feb  8 08:49 ..
drwx--x--x  3 vladodemo csfac  121 Feb  8 13:59 e3
drwx--x--x  3 vladodemo csfac  121 Feb  8 13:59 e4
drwx--x--x  3 vladodemo csfac  178 Feb  8 15:26 e5
drwx--x--x  3 vladodemo csfac  209 Feb  8 15:39 e6
drwx--x--x  3 vladodemo csfac  209 Feb 13 14:56 e7
drwx--x--x  3 vladodemo csfac 4096 Feb 13 15:34 e8
drwx-----  2 vladodemo csfac    6 Feb 15 14:57 git
drwx--x--x 13 vladodemo csfac  120 Feb 15 14:57 .
$ rsync -av e1/ git/e1/

```

### Copy other Directories with `rsync`: e2

```

$ rsync -av e2/ git/e2/
sending incremental file list
created directory git/e2
./
index.html

```

```

timberlea.cs.dal.ca - PuTTY
index.html~

sent 255 bytes  received 86 bytes  682.00 bytes/sec
total size is 70  speedup is 0.21
$ rsync -av e2/ git/e2/
sending incremental file list
created directory git/e2
./
index.html
index.html~
material/
material/.htaccess
material/MDI-page.pdf

sent 794,495 bytes  received 136 bytes  1,589,262.00 bytes/
sec
total size is 793,957  speedup is 1.00
$ rsync -av e3/ git/e3/
sending incremental file list
created directory git/e3
./
.htaccess
.htaccess~
.htpasswd
index.html
index.html~
pw
material/
material/.htaccess
material/MDI-page.pdf

```

```

$ rsync -av e4/ git/e4/

```

```
$ rsync -av e8/ git/e8/
sending incremental file list
created directory git/e8
./
.htaccess
5 | htaccess...
```

Use highlighted commands to check your git directory

```
$ ls
e1 e2 e3 e4 e5 e6 e7 e8 git
$ cd git
$ ls
e1 e2 e3 e4 e5 e6 e7 e8
$ ls -l
total 4
drwx--x--x 2 vladodemo csfac 43 Feb 6 13:16 e1
drwx--x--x 3 vladodemo csfac 59 Feb 6 13:27 e2
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e3
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e4
drwx--x--x 3 vladodemo csfac 178 Feb 8 15:26 e5
drwx--x--x 3 vladodemo csfac 209 Feb 8 15:39 e6
drwx--x--x 3 vladodemo csfac 209 Feb 13 14:56 e7
drwx--x--x 3 vladodemo csfac 4096 Feb 13 15:34 e8
$ pwd
<your home directory .....> /public_html/dgin5201/git
```

### Follow git Commands for Upload

We now follow git commands shows on the GitLab page  
 First, let us highlight which ones we will use:

```

git clone https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vladodemo2.git
cd vladodemo2
git switch --create main
touch README.md
git add README.md
git commit -m "add README"
git push --set-upstream origin main

Push an existing folder
cd existing_folder
git init --initial-branch=main
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git add .
git commit -m "Initial commit"
git push --set-upstream origin main

Push an existing Git repository
cd existing_repo
git remote rename origin old-origin
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git push --set-upstream origin --all
git push --set-upstream origin --tags
    
```

```

test.cgi
material/
material/.htaccess
material/MDI-page.pdf

sent 806,456 bytes received 478 bytes 322,773.60 bytes/se
c
total size is 804,714 speedup is 1.00
$ ls
e1 e2 e3 e4 e5 e6 e7 e8 e8a git save
$ cd git
$ ls
e1 e2 e3 e4 e5 e6 e7 e8
$ ls -l
total 4
drwx--x--x 2 vladodemo csfac 43 Feb 6 13:16 e1
drwx--x--x 3 vladodemo csfac 59 Feb 6 13:27 e2
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e3
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e4
drwx--x--x 3 vladodemo csfac 178 Feb 8 15:26 e5
drwx--x--x 3 vladodemo csfac 209 Feb 8 15:39 e6
drwx--x--x 3 vladodemo csfac 209 Feb 13 14:56 e7
drwx--x--x 3 vladodemo csfac 4096 Feb 13 15:34 e8
$ pwd
/users/faculty/vladodemo/public_html/dgin5201/git
    
```

## Type in Commands as Shown in GitLab page

Carefully copy commands from the GitLab page (these screenshots are from the last year):

```

Push an existing folder
cd existing_folder
git init --initial-branch=main
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git add .
git commit -m "Initial commit"
git push --set-upstream origin main

Push an existing Git repository
cd existing_repo
git remote rename origin old-origin
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git push --set-upstream origin --all
git push --set-upstream origin --tags

$ ls -l
total 4
drwx--x--x 2 vladodemo csfac 43 Feb 6 13:16 e1
drwx--x--x 3 vladodemo csfac 59 Feb 6 13:27 e2
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e3
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e4
drwx--x--x 3 vladodemo csfac 178 Feb 8 15:26 e5
drwx--x--x 3 vladodemo csfac 209 Feb 8 15:39 e6
drwx--x--x 3 vladodemo csfac 209 Feb 13 14:56 e7
drwx--x--x 3 vladodemo csfac 4096 Feb 13 15:34 e8
$ pwd
/users/faculty/vladodemo/public_html/dgin5201/git
$ git init --initial-branch=main
Initialized empty Git repository in /users/webhome/vladodem
o/dgin5201/git/.git/
$ git remote add origin https://git.cs.dal.ca/courses/2024-
winter/dgin5201/yourcsid

Push an existing folder
g_folder
--initial-branch=main
> add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vladodemo2.git
t -m "Initial commit"
--set-upstream origin main

Push an existing Git repository
cd existing_repo
git remote rename origin old-origin
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git push --set-upstream origin --all
git push --set-upstream origin --tags

$ ls -l
total 4
drwx--x--x 2 vladodemo csfac 43 Feb 6 13:16 e1
drwx--x--x 3 vladodemo csfac 59 Feb 6 13:27 e2
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e3
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e4
drwx--x--x 3 vladodemo csfac 178 Feb 8 15:26 e5
drwx--x--x 3 vladodemo csfac 209 Feb 8 15:39 e6
drwx--x--x 3 vladodemo csfac 209 Feb 13 14:56 e7
drwx--x--x 3 vladodemo csfac 4096 Feb 13 15:34 e8
$ pwd
/users/faculty/vladodemo/public_html/dgin5201/git
$ git init --initial-branch=main
Initialized empty Git repository in /users/webhome/vladodem
o/dgin5201/git/.git/
$ git remote add origin https://git.cs.dal.ca/courses/2024-
winter/dgin5201/yourcsid.git

Push an existing folder
cd existing_folder
git init --initial-branch=main
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git add .
git commit -m "Initial commit"
git push --set-upstream origin main

Push an existing Git repository
cd existing_repo
git remote rename origin old-origin
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git push --set-upstream origin --all
git push --set-upstream origin --tags

drwx--x--x 3 vladodemo csfac 59 Feb 6 13:27 e2
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e3
drwx--x--x 3 vladodemo csfac 121 Feb 8 13:59 e4
drwx--x--x 3 vladodemo csfac 178 Feb 8 15:26 e5
drwx--x--x 3 vladodemo csfac 209 Feb 8 15:39 e6
drwx--x--x 3 vladodemo csfac 209 Feb 13 14:56 e7
drwx--x--x 3 vladodemo csfac 4096 Feb 13 15:34 e8
$ pwd
/users/faculty/vladodemo/public_html/dgin5201/git
$ git init --initial-branch=main
Initialized empty Git repository in /users/webhome/vladodem
o/dgin5201/git/.git/
$ git remote add origin https://git.cs.dal.ca/courses/2024-
winter/dgin5201/adm/vladodemo2.git
$ git add .
$ git commit -m "Initial commit"

Courses > ... > adm > vladodemo2
git clone https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vladodemo2.git
cd vladodemo2
git switch --create main
touch README.md
git add README.md
git commit -m "add README"
git push --set-upstream origin main

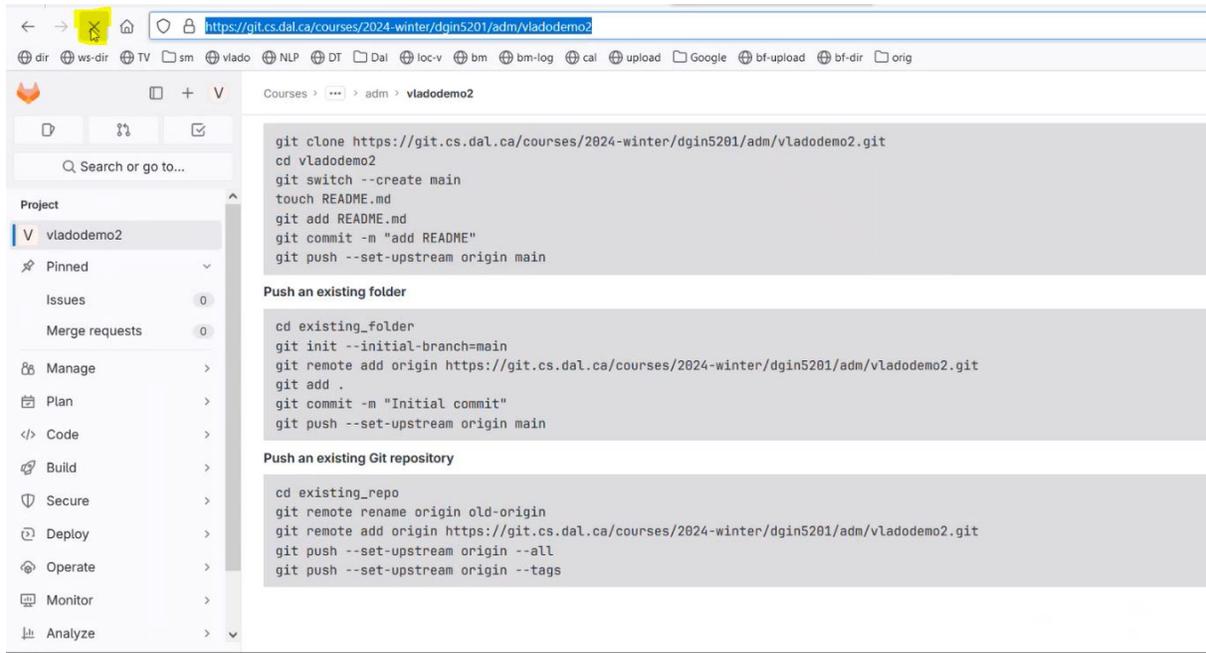
Push an existing folder
cd existing_folder
git init --initial-branch=main
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git add .
git commit -m "Initial commit"
git push --set-upstream origin main

Push an existing Git repository
cd existing_repo
git remote rename origin old-origin
git remote add origin https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/vla
git push --set-upstream origin --all
git push --set-upstream origin --tags

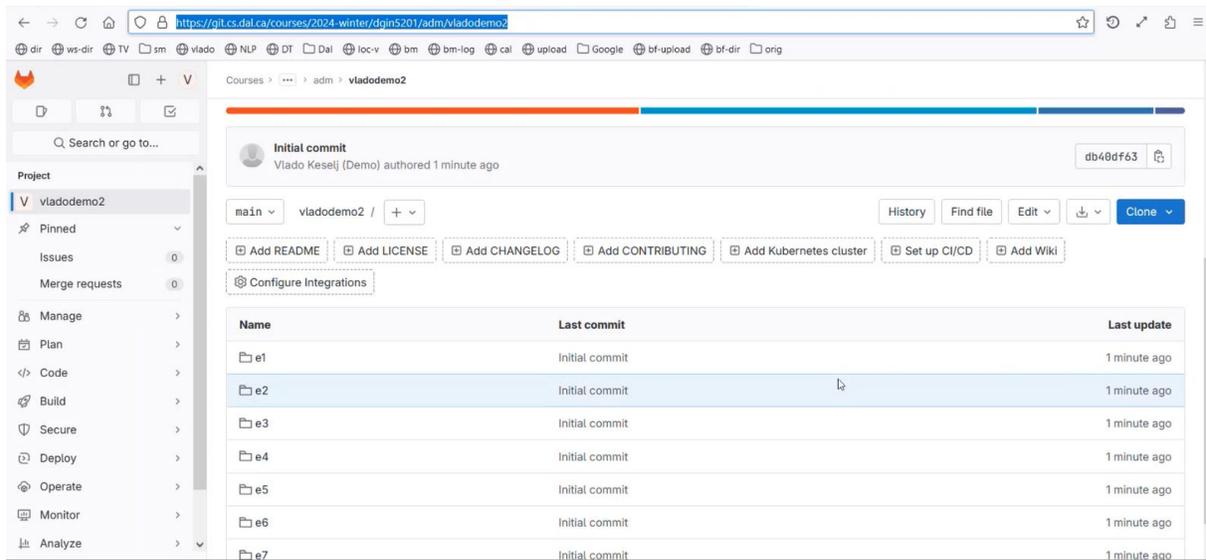
create mode 100644 e8/material/MDI-page.pdf
create mode 100644 e8/pw
create mode 100755 e8/register-py.cgi
create mode 100755 e8/register-py.cgi~
create mode 100755 e8/register.cgi
create mode 100755 e8/register.cgi~
create mode 100755 e8/register.php
create mode 100755 e8/register.py
create mode 100644 e8/registrations-saved.txt
create mode 100755 e8/test.cgi
$ git push --set-upstream origin main
Username for 'https://git.cs.dal.ca': vladodemo
Password for 'https://vladodemo@git.cs.dal.ca':
Enumerating objects: 44, done.
Counting objects: 100% (44/44), done.
Delta compression using up to 32 threads
Compressing objects: 100% (40/40), done.
Writing objects: 100% (44/44), 766.54 KiB | 12.78 MiB/s, do
ne.
Total 44 (delta 21), reused 0 (delta 0), pack-reused 0
To https://git.cs.dal.ca/courses/2024-winter/dgin5201/adm/v
ladodemo2.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'or
igin'.

```

### Refresh GitLab Page to Check Contents



### Expected Contents in your GitLab Repository



The screenshot shows the GitLab web interface for a repository named 'vladodemo2'. The breadcrumb navigation shows 'Courses > ... > adm > vladodemo2'. The current directory is 'main / vladodemo2 /'. There are buttons for 'History', 'Find file', 'Edit', and 'Clone'. Below the navigation, there are several utility buttons: 'Add README', 'Add LICENSE', 'Add CHANGELOG', 'Add CONTRIBUTING', 'Add Kubernetes cluster', 'Set up CI/CD', and 'Add Wiki'. A 'Configure Integrations' button is also present. A table lists the repository's contents:

Name	Last commit	Last update
e1	Initial commit	1 minute ago
e2	Initial commit	1 minute ago
e3	Initial commit	1 minute ago
e4	Initial commit	1 minute ago
e5	Initial commit	1 minute ago
e6	Initial commit	1 minute ago
e7	Initial commit	1 minute ago
e8	Initial commit	1 minute ago

The screenshot shows the GitLab web interface for the 'e8' directory within the 'vladodemo2' repository. The breadcrumb navigation shows 'Courses > ... > adm > vladodemo2'. The current directory is 'main / vladodemo2 / e8 /'. There are buttons for 'Lock', 'History', 'Find file', 'Edit', and 'Clone'. A table lists the contents of the 'e8' directory:

Name	Last commit	Last update
..		
material	Initial commit	1 minute ago
.htaccess	Initial commit	1 minute ago
.htaccess~	Initial commit	1 minute ago
.htpasswd	Initial commit	1 minute ago
index-php.html	Initial commit	1 minute ago
index-php.html~	Initial commit	1 minute ago
index-py.html	Initial commit	1 minute ago
index-py.html~	Initial commit	1 minute ago
index-py2.html	Initial commit	1 minute ago
index-py2.html~	Initial commit	1 minute ago

### GitLab Upload Completed

- With this, we finished uploading e1, e2, e3, e3 to GitLab
- Now, we go back to e5
- Change directory to `~/public_html/dgin5201/`