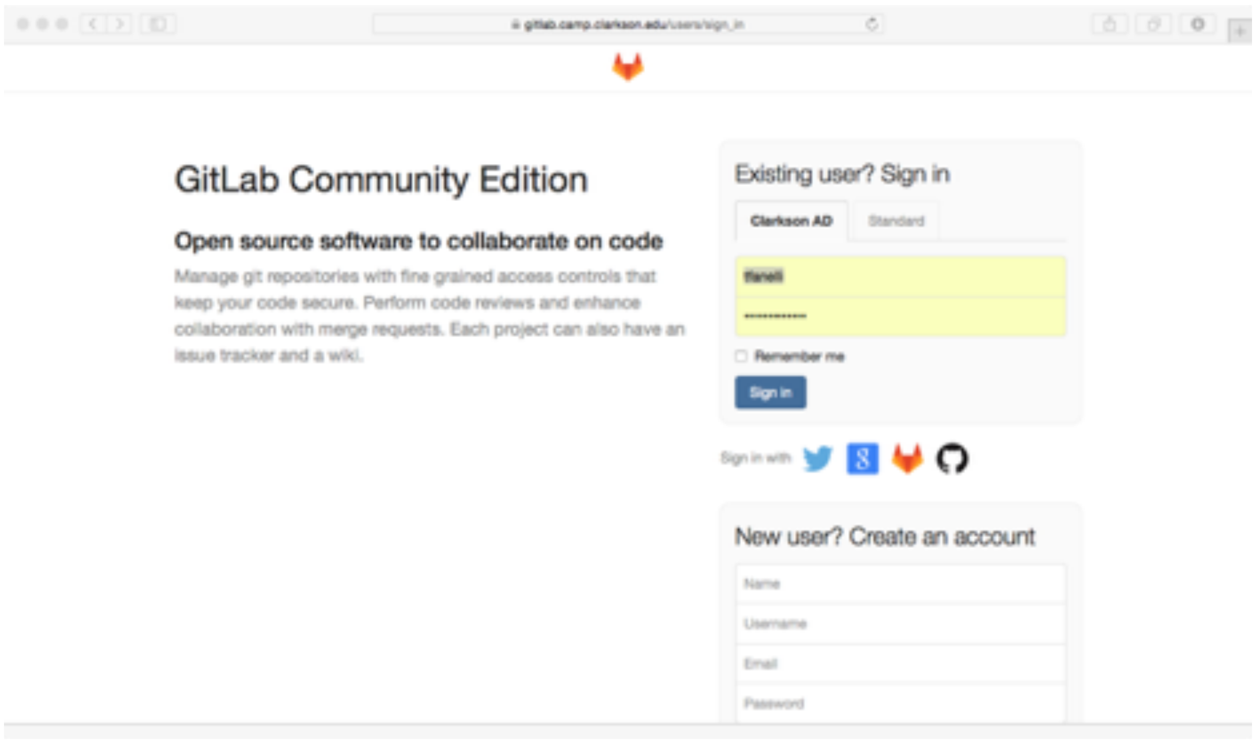
Getting Started with GitLab

Logging In

To log in, go to:

http://gitlab.camp.clarkson.edu

You will be prompted with a certificate warning, because the server uses a self-signed SSL certificate. Please ignore this warning.



GitLab Community Edition

Open source software to collaborate on code

Manage git repositories with fine grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

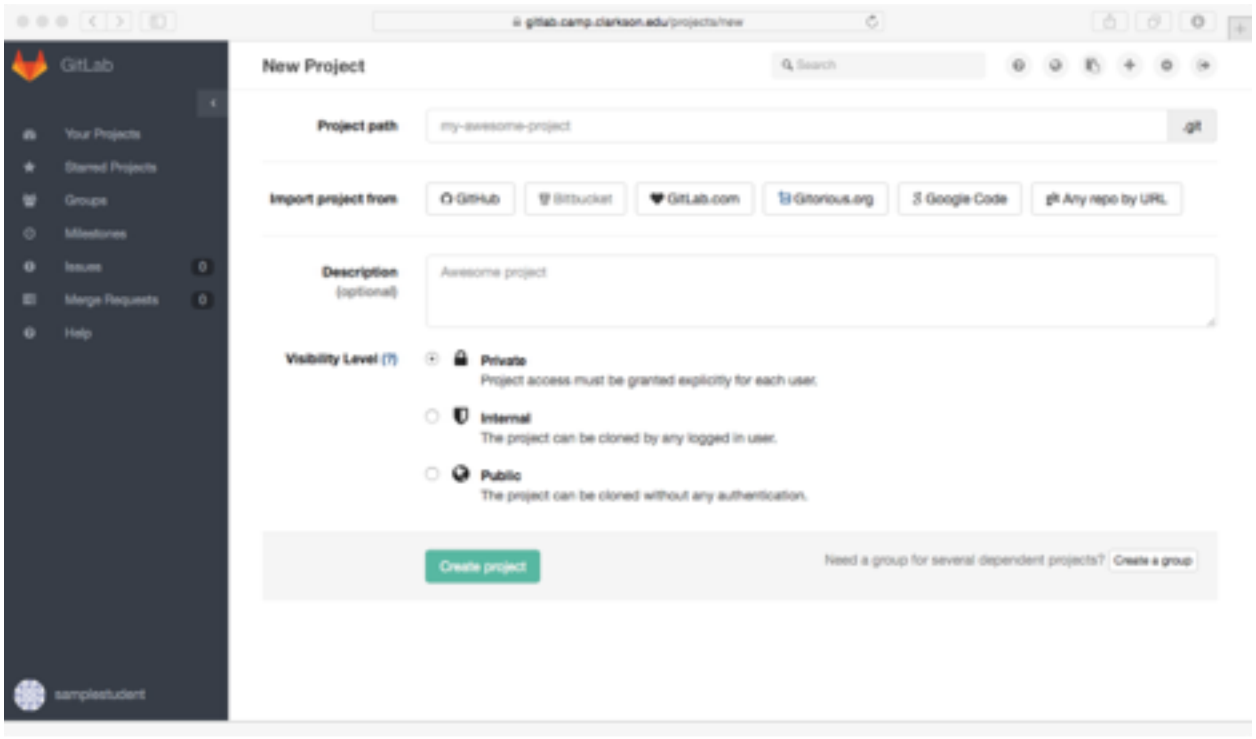
Log in using your Clarkson AD username and password. Once you've logged in, you can go to your profile settings, and link your Twitter, Google, GitLab, or GitHub accounts so that you can log-in later using them.

Once you've logged in, you can start creating projects or groups, or browsing other public projects.

Creating A Project for Your Class

Let's begin by creating a project to store work for a class. This guide will use EE363 as an example, but any class will do!

On the "Your Projects" page, click the "New Project" button.



Your "Project path" can be anything you like - but should be descriptive of the content that will be kept inside of it. Your instructor may have specific guidelines on naming convention that should be used for their class, so pay careful attention to any instructions given.

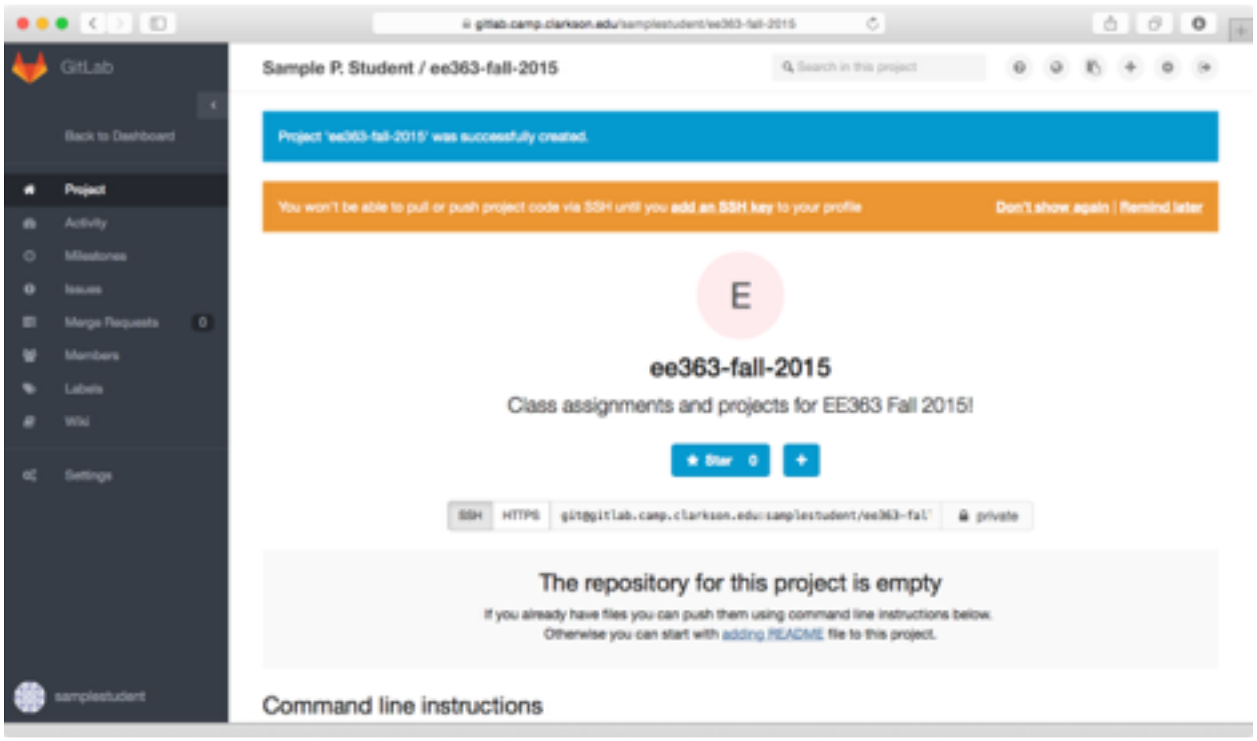
In this tutorial, we'll just use the class number, semester, and year, like so:

ee363-fall-2015

Your Project path can not contain spaces, and should be all lowercase. If you like, go ahead and enter a description.

Be sure to leave the Visibility Level as *Private.* If you make it Internal or Public, than anyone else at Clarkson will be able to see or use your work. Internal and Public repositories are great for individual, non-course related projects that you want to share with the community; but your class work MUST be private (we'll talk about teams next)

When you're done, press "Create Project"



gitlab.camp.clarkson.edu/samplestudent/ee363-fall-2015

Sample P. Student / ee363-fall-2015

Project 'ee363-fall-2015' was successfully created.

You won't be able to pull or push project code via SSH until you add an SSH key to your profile

ee363-fall-2015

Class assignments and projects for EE363 Fall 2015!

```
git@gitlab.camp.clarkson.edu:samplestudent/ee363-fal
```

The repository for this project is empty

If you already have files you can push them using command line instructions below.
Otherwise you can start with adding README file to this project.

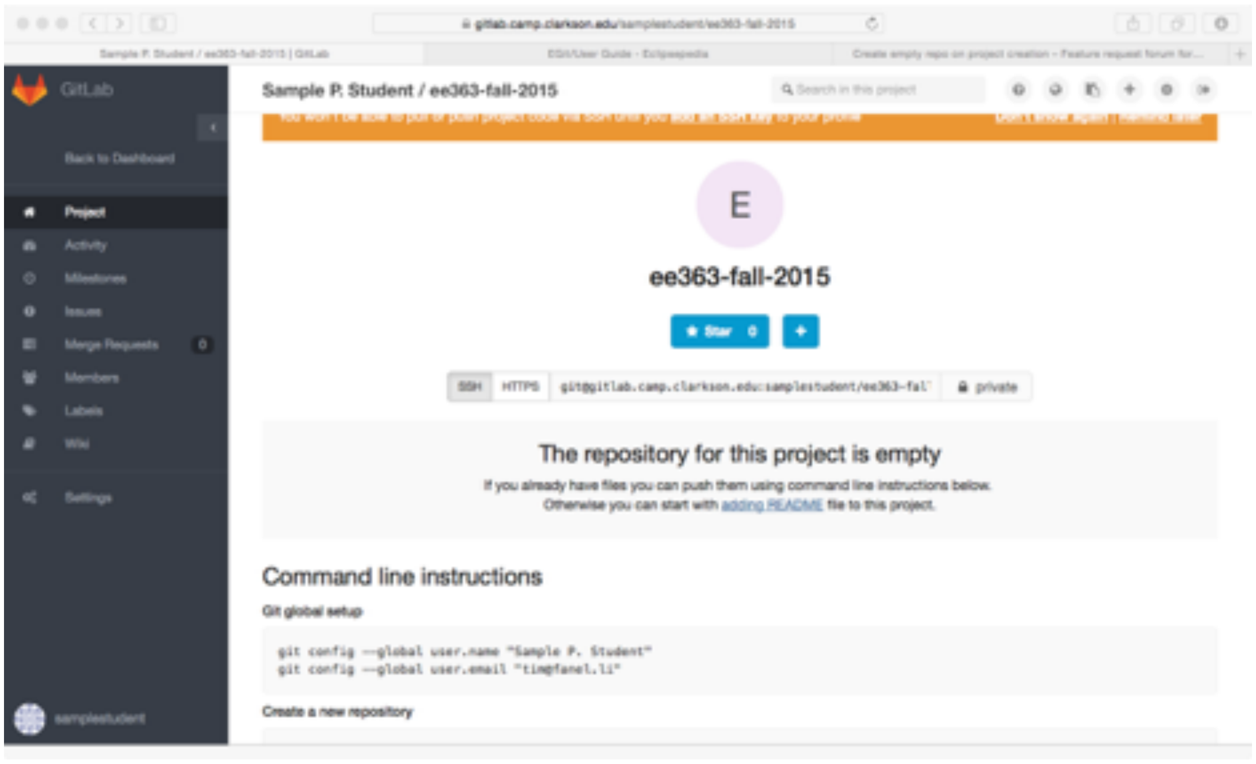
Command line instructions

One last step - be sure to add your instructor as a Member of your new project. This will allow your instructor to see your project, and make changes, leave feedback, etc.

On the left side of the screen, click “Members”, then click the green “Add members” button. Search for your instructor by name or email address. Set their “Project Access” to at least Guest. If you would like your professor to be able to make comments or edits to your project during office hours, then set Project Access to “Developer”.

Click the “Add users to project” button.

Initializing Your Empty Project



Sample P. Student / ee363-fall-2015

ee363-fall-2015

git@gitlab.camp.clarkson.edu:samplestudent/ee363-fal

The repository for this project is empty

If you already have files you can push them using command line instructions below.
Otherwise you can start with adding README file to this project.

Command line instructions

Git global setup

```
git config --global user.name "Sample P. Student"
git config --global user.email "tim@fanel.li"
```

Create a new repository

When the project is first created, the repository is in an *empty, uninitialized* state. To initialize the project repository, simply click the link that says “adding README”.

This will bring up the editor, where you can enter some content into your README. Just a quick one-liner is sufficient. Below the editor, enter a “Commit message”, and then click Commit.

Your project is now initialized, shared with your instructor, and ready for you to push your projects!

Sharing Projects from Eclipse

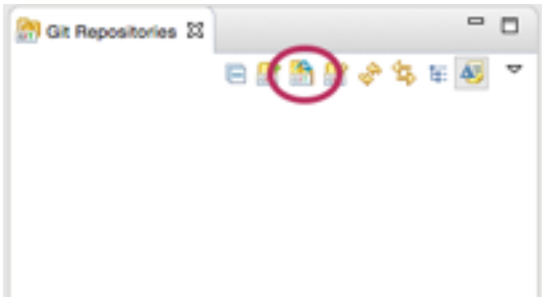
To share a project from Eclipse to your GitLab project, you must first find your project's URL. This is on your project's page, and there are two forms: SSH and HTTPS.

To use SSH, you must add an SSH key to your GitLab profile. We're not going to cover this. So instead, click on HTTPS to get your HTTPS URL, and copy it into your clipboard.

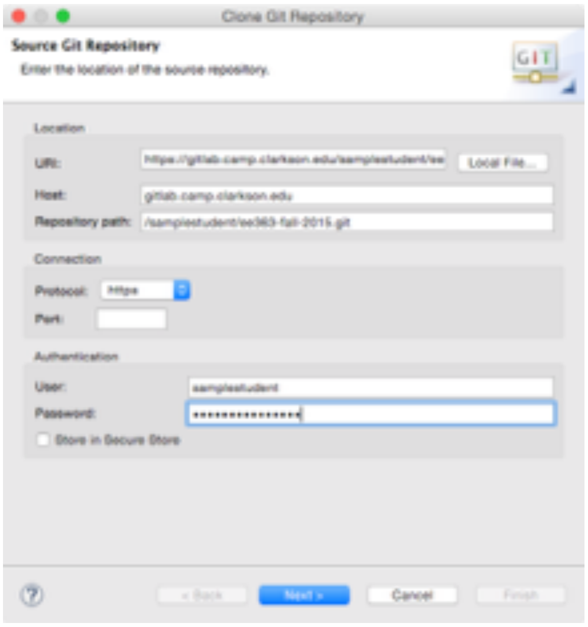
Clone the Repository

The first step is to create a clone of your server repository. The clone is a copy of the repository that resides on your local computer. Changes you make to the clone are only visible to you, until you push them back to the "remote" repository on the server.

In Eclipse, open the "Git" perspective. Then, click the "Clone a Git Repository" button under the Git Repositories view



Paste your project URL into the "URI" field, and enter your AD username and password, then click Next, click Next again, then click Finish.


Clone Git Repository

Source Git Repository

Enter the location of the source repository.

Location

URI: https://gitlab.camp.clarkson.edu/samplestudent/ee

Host: gitlab.camp.clarkson.edu

Repository path: /samplestudent/ee363-fall-2015.git

Connection

Protocol: https

Port:

Authentication

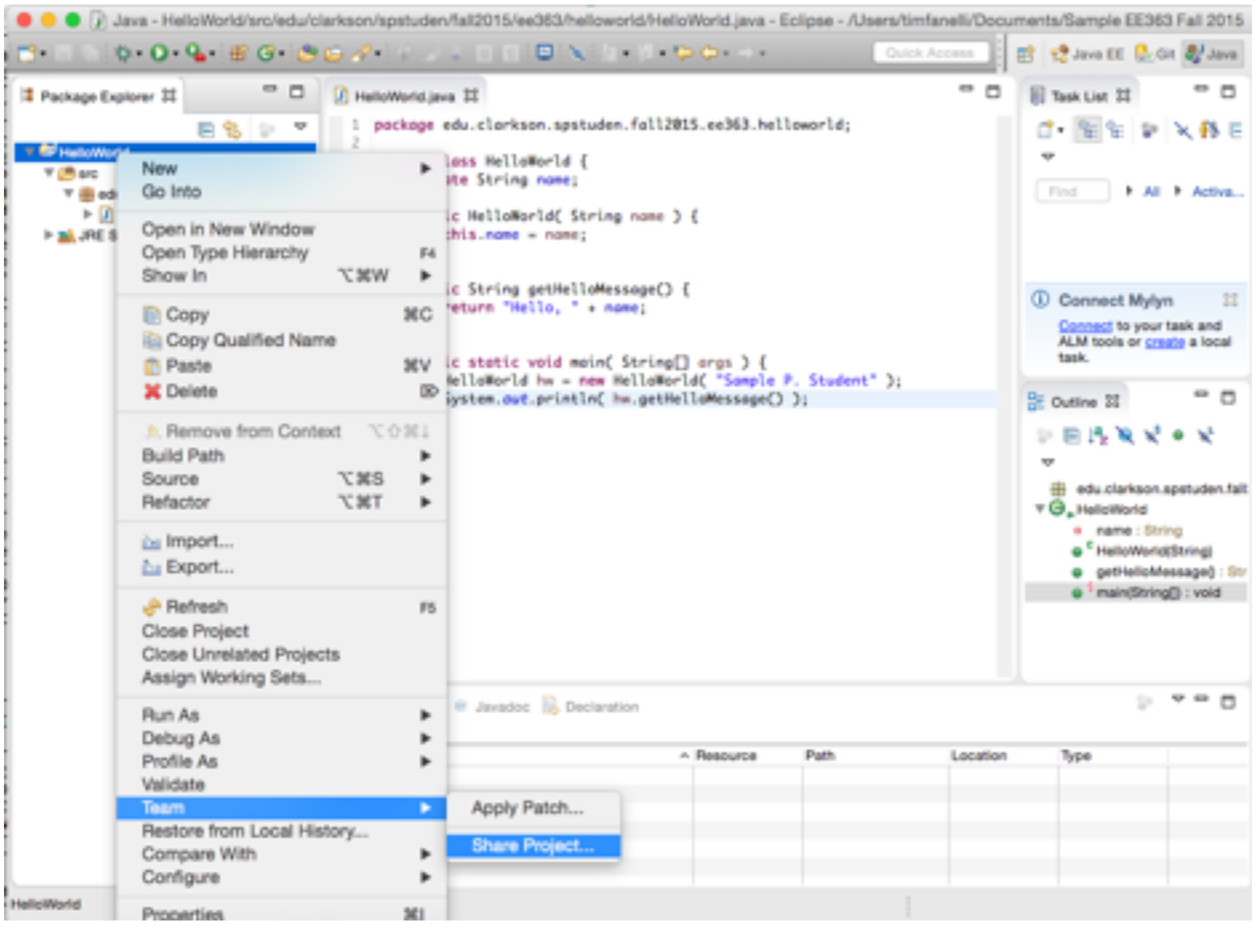
User: samplestudent

Password: ***************

☐ Store in Secure Store

Share Your Project

Now that you have a local clone, you can share your project to it. In Eclipse, switch back to the Java perspective, and then right-click your project and choose Team->Share Project.



```
package edu.clarkson.spstuden.fall2015.ee363.helloworld;

lass HelloWorld {
te String name;

c HelloWorld( String name ) {
his.name = name;

c String getHelloMessage() {
eturn "Hello, " + name;

c static void main( String[] args ) {
elloWorld hw = new HelloWorld( "Sample P. Student" );
ystem.out.println( hw.getHelloMessage() );
```

In the “Configure Git Repository” dialog, choose the clone you just created, then click Finish.

Adding Files to the Index

The “Index” is the list of files in your project that Git will store in the repository. Anytime you add or modify a file in your project, you’ll want to add it to the index so that it gets acted on next time your “commit” your changes.

We’ll add everything in our project to the index.

Right-click the project, and choose Team->Add to index. Notice how the icons on the files changed.

Committing Your Changes

When you are satisfied with your changes, you are ready to “Commit” them to your repository. To do this, right click your project, and choose Team->Commit.

In the dialog that appears, enter a commit message. It’s important to enter a meaningful message that represents what has changed since the *last* commit.

Then, click “Commit”, and notice again how the icons on the files changed.

Pushing Your Changes Upstream

At this point, you have shared and committed your project to your *local clone* only. If you log into the GitLab server, you will not see any of your changes in your project there.

To get your changes to GitLab, you must “push” your changes from your local repository to the remote repository.

You can do this by right clicking your project, and choosing Team -> Push to upstream. You should get a confirmation back indicating the push was successful. You can now view your project in GitLab, again.

NOTE: You can make as many commits as you like, or need, between pushes. Each individual commit is still pushed to the server, so you have a full history of all changes that were made in your project.