

GITLAB

CS 346: Application
Development

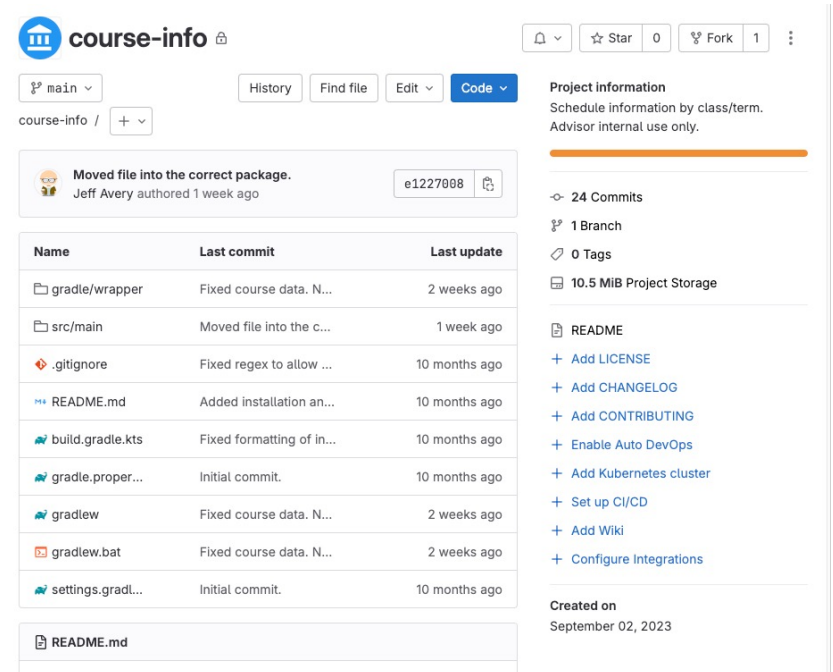
WHAT IS GITLAB?

GitLab is a project tracking system.

- It's like GitHub but targets enterprise customers.
- UW has our own self-hosted instance.
<https://git.uwaterloo.ca/>

Features

- **Project planning, tracking** (issues, milestones).
- Source code management (Git, merging)
- Continuous integration (running tests, deployment).
- Security auditing (out-of-scope for us)
- Wiki for documentation, diagrams.



The screenshot shows a GitLab repository page for 'course-info'. The page includes a navigation bar with 'main' branch selected, 'History', 'Find file', 'Edit', and 'Code' buttons. A commit message 'Moved file into the correct package.' by Jeff Avery is displayed. Below this is a table of files with columns for 'Name', 'Last commit', and 'Last update'. The right sidebar contains project information, commit statistics (24 Commits, 1 Branch, 0 Tags), storage usage (10.5 MiB), and a list of actions like 'Add LICENSE', 'Add CHANGELOG', 'Enable Auto DevOps', 'Add Kubernetes cluster', 'Set up CI/CD', 'Add Wiki', and 'Configure Integrations'. The page was created on September 02, 2023.

Name	Last commit	Last update
gradle/wrapper	Fixed course data. N...	2 weeks ago
src/main	Moved file into the c...	1 week ago
.gitignore	Fixed regex to allow ...	10 months ago
README.md	Added installation an...	10 months ago
build.gradle.kts	Fixed formatting of in...	10 months ago
gradle.proper...	Initial commit.	10 months ago
gradlew	Fixed course data. N...	2 weeks ago
gradlew.bat	Fixed course data. N...	2 weeks ago
settings.gradl...	Initial commit.	10 months ago

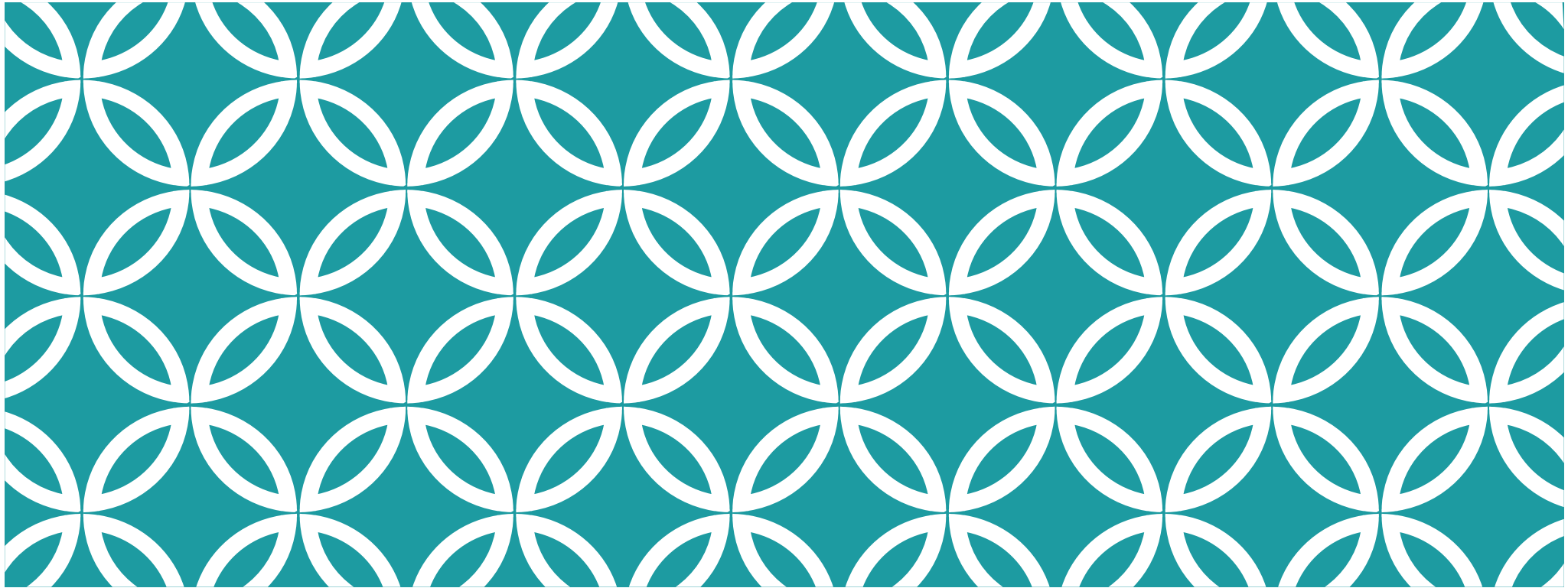
The interface looks familiar.

MAIN FUNCTIONALITY

Here's the main functionality that we'll be using in this course:

- **Project tracking.** We will use project issues and milestones to track the work that you do towards your project.
- **Source control.** Your GitLab repository is a Git repo. You should be cloning it to your personal computers and using the repo for your source code (and any other documents).
- **Software releases.** We'll use the built-in mechanisms to tag and release software for each milestone.
- **Wiki.** Most documents should be stored as pages in your Wiki, written in Markdown. This includes everything that you are asked to submit in the course.

For a small sample project, see <https://git.uwaterloo.ca/cs346/public/mm>



GITLAB SETUP

CS 346: Application
Development

GETTING STARTED

1. Navigate to <https://git.uwaterloo.ca>
2. Click on **+ > New Project > Create blank project.**

Project name
Team-101-5

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL
https://git.uwaterloo.ca/ j2avery

Project slug
team-101-5

Visibility Level ⓘ

Private
Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

Internal
The project can be accessed by any logged in user except external users.

Public
The project can be accessed without any authentication.

Project Configuration

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.

Enable Static Application Security Testing (SAST)
Analyze your source code for known security vulnerabilities. [Learn more.](#)

Create project Cancel

The URL should be under a team-mates name e.g., j2avery

Use your team number (from Learn) when naming your project!

Make sure it's private to your team

SETUP: MEMBERS

It doesn't matter who "owns" the project, since we will add other team members and give them full access.

- Add your team members as Owners.
- Add course staff as Developers.

Manage > Members

Members were successfully added



Project members

You can invite a new member to **notae** or invite another group.

Import from a project | Invite a group | **Invite members**

Members 2

Filter members | Account

Account	Source	Max role	Expiration	Activity
 Caroline Kierstead @ctkierstead	Direct member by Jeff Avery	Owner	Expiration date	User created: Apr 27, 2015 Access granted: Aug 06, 2024 Last activity: Aug 05, 2024
 Jeff Avery It's you @j2avery	Direct member by Jeff Avery	Owner	Expiration date	User created: Aug 20, 2015 Access granted: Jan 05, 2024 Last activity: Aug 05, 2024

SETUP: SOURCE CODE

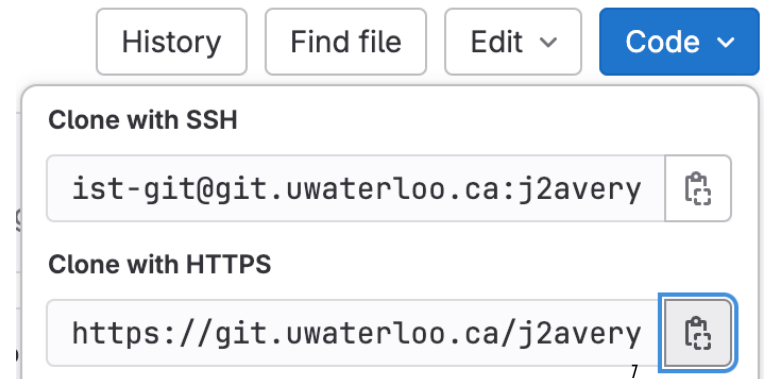
Once the repo is created, everyone that has access should be able to `git clone` it to their local machine.

Code button

- Copy the URL
- Clone the URL.

```
$ git clone https://git.uwaterloo.ca/cs346/public/mm
```

We'll add code when we create your project.



SETUP: STARTING FILES

1. Add a **README.md** file to your repository.

This file determines what is shown when you open the project (i.e., it's the “Landing Page” for your project).

- It's JUST a markdown file! You can edit it in VS code (or whatever) and just add to the repo.

2. Add a **.gitignore**

List files and directories that you do NOT want to add to your repository.

- Just a plain-text file that you can add to the repo at the top-level.

Example files here: <https://git.uwaterloo.ca/cs346/public/mm>

SETUP: MILESTONES




Milestones are important deadlines that you want to highlight in your project. In this course, these are the four software releases that you will product.

Plan > Milestones.

Jeff Avery > cs346-template > Milestones

Open 3 Closed 0 All 3

Filter by milestone name Due soon ▾ New milestone

Sprint 1 Feb 7, 2022–Feb 18, 2022 Expired Jeff Avery / cs346-template	 6 Issues · 0 Merge requests 100% complete	Close Milestone
Sprint 2 Feb 28, 2022–Mar 11, 2022 Expired Jeff Avery / cs346-template	 3 Issues · 0 Merge requests 66% complete	Close Milestone
Sprint 3 Mar 14, 2022–Mar 25, 2022 Expired Jeff Avery / cs346-template	 2 Issues · 0 Merge requests 50% complete	Close Milestone

SETUP: ISSUES

Every feature that you could implement in your project is an **issue** (synonymous with task).

A project management system, at its essence, tracks issues, and the relationship between them.

Issues contain fields, including

- **Title:** short description.
- **Description:** details required to implement it,
- **Priority:** high/medium/low.
- **Status:** new/assigned/closed

Plan > Issues











New Issue

Title (required)

Type [?](#)

Issue ▼

Description

Preview | B I  |  </>  |     |    ↶ ↷

Write a description or drag your files here...

Switch to rich text editing ⌘

Add [description templates](#) to help your contributors to communicate effectively!

This issue is confidential and should only be visible to team members with at least Reporter access.

Assignee

Unassigned ▼

[Assign to me](#)

Milestone

Select milestone ▼

Labels

Select label ▼

Due date

Select due date

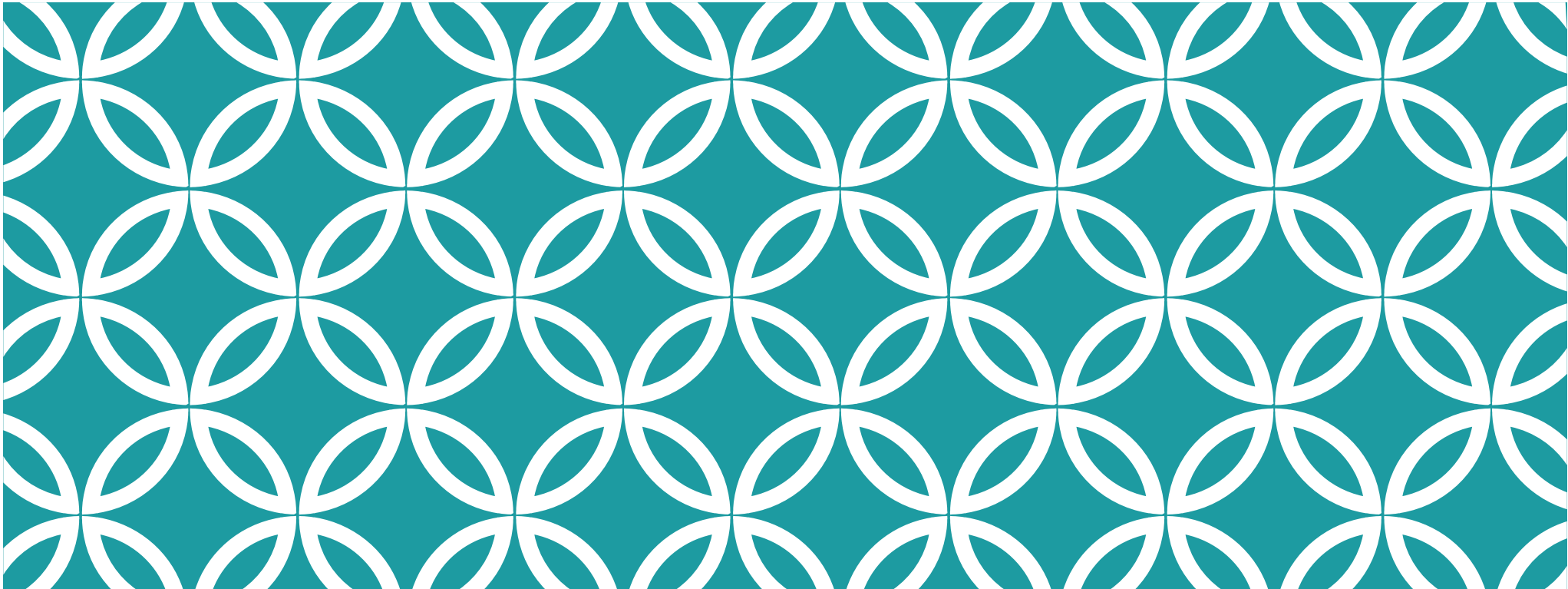
SETUP: LABELS

Create tags for priority that you can use to label and sort issues.

Manage > Labels

Prioritized labels
Drag to reorder prioritized labels and change their relative priority.

high Jeff Avery / notae	Prioritized	Issues	Merge requests	★	Subscribe	⋮
medium Jeff Avery / notae	Prioritized	Issues	Merge requests	★	Subscribe	⋮
low Jeff Avery / notae	Prioritized	Issues	Merge requests	★	Subscribe	⋮



TRACKING ISSUES

CS 346: Application
Development





WHAT DOES AN ISSUE LOOK LIKE?


Work is tracked by issues.

Each issue represents one requirement, or one high-level feature.

- Granularity: Each requirement should represent one complete (and testable) feature.
- Dependencies: You may have dependencies or overlap between requirements. This is ok!

User Story Requirement	Issue Title	Description/Details
Issue 1: Save data to a file	Save our data to a text file.	JSON file in the user's home directory. •Create data class to store data. •Add support for JSON (library). •Write function to convert list and list elements to JSON. • Convert list to JSON and store in a file.
Issue 2: Restore data from a file	Read data from a file to into a suitable data class.	Read data from JSON file in user's home directory (see issue 1). •Create data class to store data. •Add support for JSON (library). •Write function to extract list and list elements from JSON. • Read JSON from a file into a list.

 Open  Issue created 1 minute ago by  Jeff Avery 

Close issue 

Save data to a file

Save our data (stored internally as a list) to a text file in JSON format.

Steps:

1. Create data class to store data.
2. Add support for JSON (library).
3. Write function to convert list and list elements to JSON.
4. Convert list to JSON and store in a file.

 Drag your designs here or [click to upload](#).


Child items  0

Add a task 

No child items are currently assigned. Use child items to prioritize tasks that your team should complete in order to accomplish your goals!

Linked items   1

Add 

 Restore data from a file
#14

 Sprint 1 

An example of an issue, linked to a related issue. Each should have its own independent unit tests.

GOAL: UNASSIGNED ISSUES

Your outcome from requirements and design should be a “pool” of unassigned issues in your project. These unassigned issues constitute your **Product Backlog**.

During your sprint kickoff, your team should

- Establish a major goal for the sprint (e.g., “get the UI working”, “integrate front and backend”)
- Determine which issues need to be addressed to meet those goals.
- Add those issues to your milestone and assign them to team members.

Default values for each issue:

- **Type:** Issue
- **Assignee:** Unassigned
- **Milestone:** No milestone
- **Due date:** Unselected
- **Description:** As much as you have!
- **Dependencies:** Where they make sense.