ENGR 301 *Project Management*

Lecture 7 — GitLab II

James Quilty

School of Engineering and Computer Science
Victoria University of Wellington

Introduction

Today's lecture discusses how GitLab's *Merge Request* feature helps teams and groups follow a consistent branching strategy which is integrated with the issue tracking system

- GitLab documentation

  `https://docs.gitlab.com/ee/user/project/merge_requests/`
- GitLab Flow video (4m48s)

  `https://www.youtube.com/watch?v=InKNIvky2KE`
- Tutorial: It's all connected in GitLab

  `https://about.gitlab.com/blog/2016/03/08/gitlab-tutorial-its-all-connected/`
- How to do GitLab merge request reviews in VS Code

  `https://about.gitlab.com/blog/2021/01/25/mr-reviews-with-vs-code/`

Branching Strategy

A *branching strategy* is necessary to keep the situation with `git` from spiralling out of control. There are *many* branching strategies, the one we will use a simplified version of GitLab Workflow. Branches are:

- created from an *Issue* via a *Merge Request*
- directed toward a *single purpose*
- typically *short-lived* on the time scale of the project

This is sometimes referred to as a “trunk-based work flow”.

Merge Requests

Use the GitLab web interface to simplify the process — that's why it was created:

- Issues to plan work
- **Merge Requests to manage `git` branching and review work**
- CI/CD to check work

Why Merge Requests?

Use the GitLab web interface to simplify the process — that's why it was created:

- Central place of record — *answers the* **why**
- Make suggestions on code — peer review
- Merge in browser — provides multiple safeguards

Merge Requests

GitLab web interface:

- Create MRs from an Issue — automates branch creation
- New Merge Request interface — set options
- Automatic Issue closing — and other neat features
- Merge Request Overview, Commits, Pipelines and Changes