AUBURN

# Gitlab DevOps Tool

# Gitlab DevOps Tool

AUBURN

- Gitlab will be used during the course to progressively build our secure software development build pipeline.
- Gitlab provides the follow key capabilities:
  - Git repositories
  - Controlled change management through branch protection and change review
  - Automated build process based on triggers such as code checkins
  - Execution of compile, tests, scanning, packaging, and deployment of applications
  - Repository to manage dependencies
- Provides Continuous Integration/Continuous Deployment (CI/CD) capability
- Goal is to build security testing methods into the CI/CD pipeline

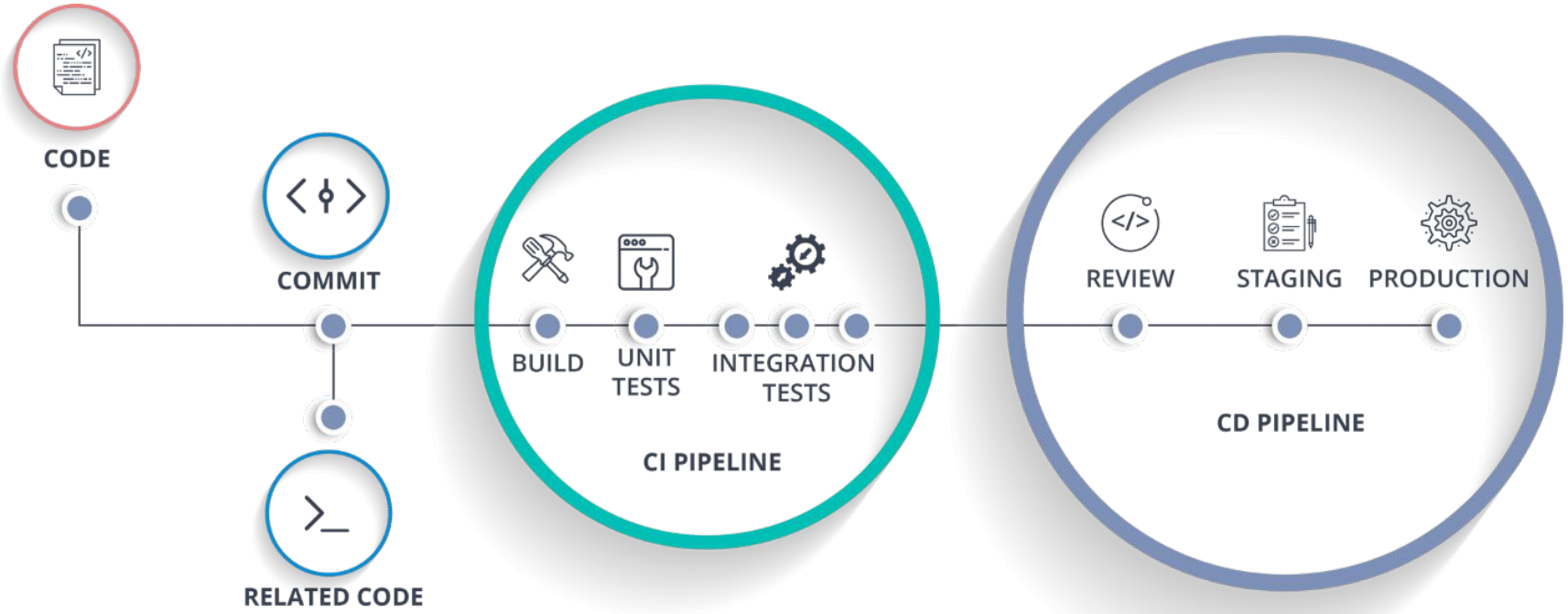
# CI/CD Key Concepts

AUBURN

- **Continuous integration (CI)**
  - Automating the integration of code changes from multiple contributors into a single software project.
  - The central repository is where builds and tests then run.
  - Automated tools are used to assert the new code's correctness before integration.
  - Source code version control system (Git) is the crux of the CI process

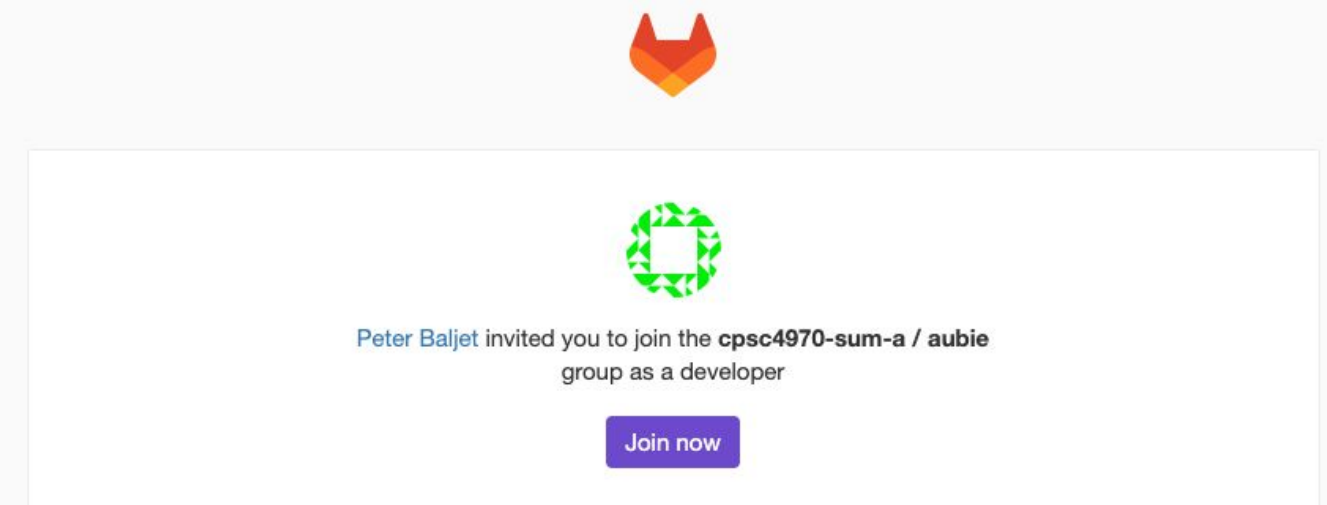
- **Continuous Deployment (CD)**
  - Extension of continuous integration since it automatically deploys all code changes to a testing and/or production environment after the build stage.
  - Further testing can be performed on deployed environments.
  - Software releases can be promoted to a production environment or official release

# What is CI/CD

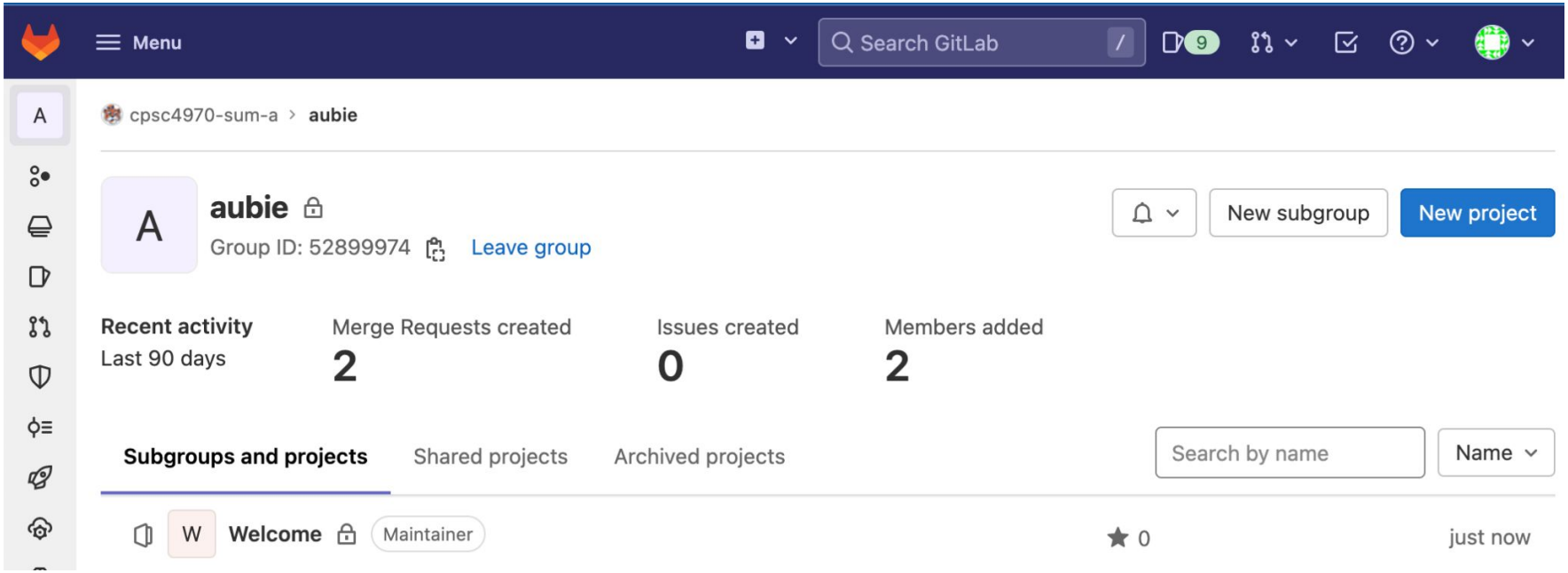

CODE
COMMIT
RELATED CODE
BUILD
UNIT TESTS
INTEGRATION TESTS
CI PIPELINE
REVIEW
STAGING
PRODUCTION
CD PIPELINE

# Gitlab Setup

- You should have received an email providing access your Gitlab group.
  - Group: “CPSC4970-summer-a/<student>”
  - Setup an account with your Auburn email address
- This Gitlab group will be used throughout the course
  - Assignment repositories will be added throughout the course.

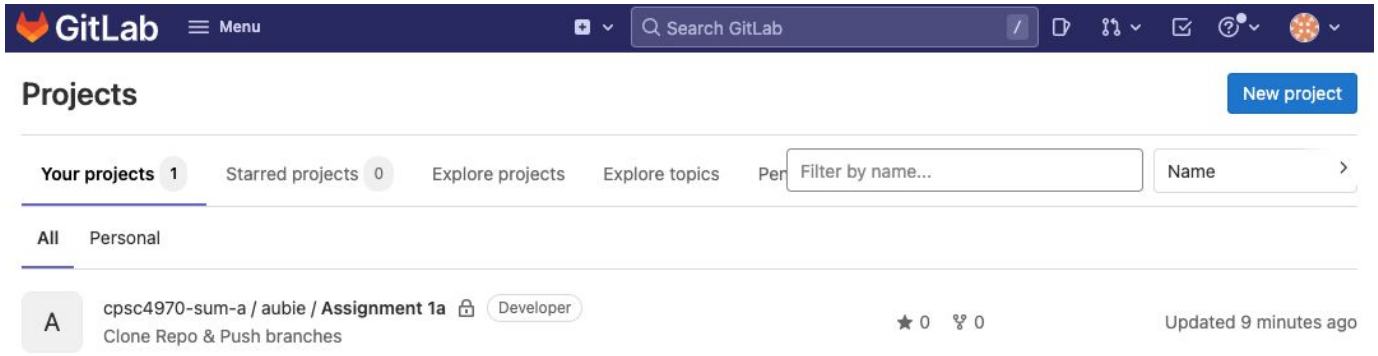


# Gitlab Overview - Login View

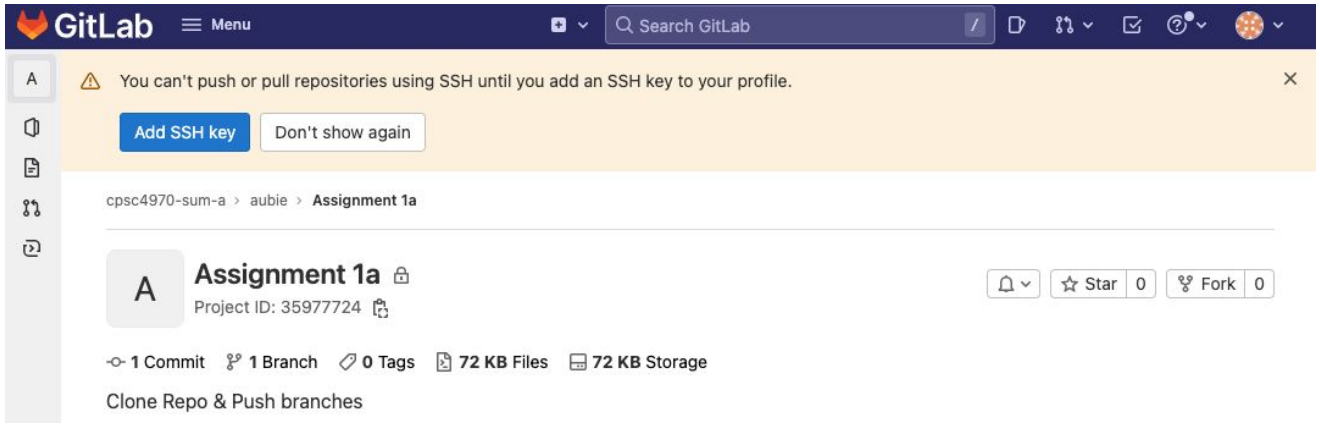


# Gitlab Overview - Setup SSH Key

Select the Assignment 1a Project




Add an SSH key…following instructions.



# Clone Welcome Remote Repository

AUBURN

- **git clone <repository url>**
  - Most common method to clone a remote repository
  - May need to enter credentials or configurable keys to clone

```
 $ git clone https://gitlab.com/cpsc4970-sum-a/aubie/welcome.git
Cloning into 'welcome'...
remote: Enumerating objects: 6, done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 6
Unpacking objects: 100% (6/6), done.
```

- **git remote -v**
  - **Lists remote repositories for push/pull.**
  - **Also known as “upstream”**

```
 (main) $ git remote -v
origin  https://gitlab.com/cpsc4970-sum-a/aubie/welcome.git (fetch)
origin  https://gitlab.com/cpsc4970-sum-a/aubie/welcome.git (push)
```

# Pushing changes to remote repo

AUBURN

- **git push**
  - Defaults to push current branch
  - May be rejected if changes exist on remote branch
  - Good practice to do **git pull** to bring down existing changes first.
  - Will create branch on remote if it does not exist

- **git branch -a**
  - Shows all local and remote branches
  - **git checkout <branchname>** will bring down branch locally

# Gitlab Merge Requests

AUBURN

- Pushing changes to Gitlab will create merge requests.
- Merge requests are a control mechanism for reviewing changes to a branch before they are actually committed.
- Privileges for merging is given to authorized individuals who are knowledgeable and experience to review code
  - Adherence to standards
  - Functional correctness
  - Quality of code
  - Security related issues.

# Assignment 1a

AUBURN

- Practice working with git local and remote repositories
- Graded based on pushing commits and branches to Gitlab repo
- Feel free to practice outside of the assignment.  No points deducted for additional commits and branches present.