**Exercise 3**

### Branching and Merging

**Branching** is the process of creating a separate line of development from your main project. Think of the main branch as the stable, production-ready code. When you want to work on a new feature or fix a bug, you create a new **branch** off the main branch. This allows you to make changes without affecting the main codebase. When you're done, you can bring all your changes back into the main branch.

**Merging** is the process of combining two or more branches together. When your work on a feature branch is complete, you can merge it back into the main branch. Git will take the changes from your feature branch and integrate them into the main branch, creating a unified codebase.

### Creating a Branch in GitLab

To create a new branch in GitLab, navigate to your repository's main page.

1. Click the "**+**" button next to the branch name dropdown.
2. Select "**New branch**".
3. Enter a name for your new branch (e.g., feature/user-login).
4. Specify the source branch from which you want to create your new branch (usually main).
5. Click "**Create branch**".

### Creating a Merge Request in GitLab

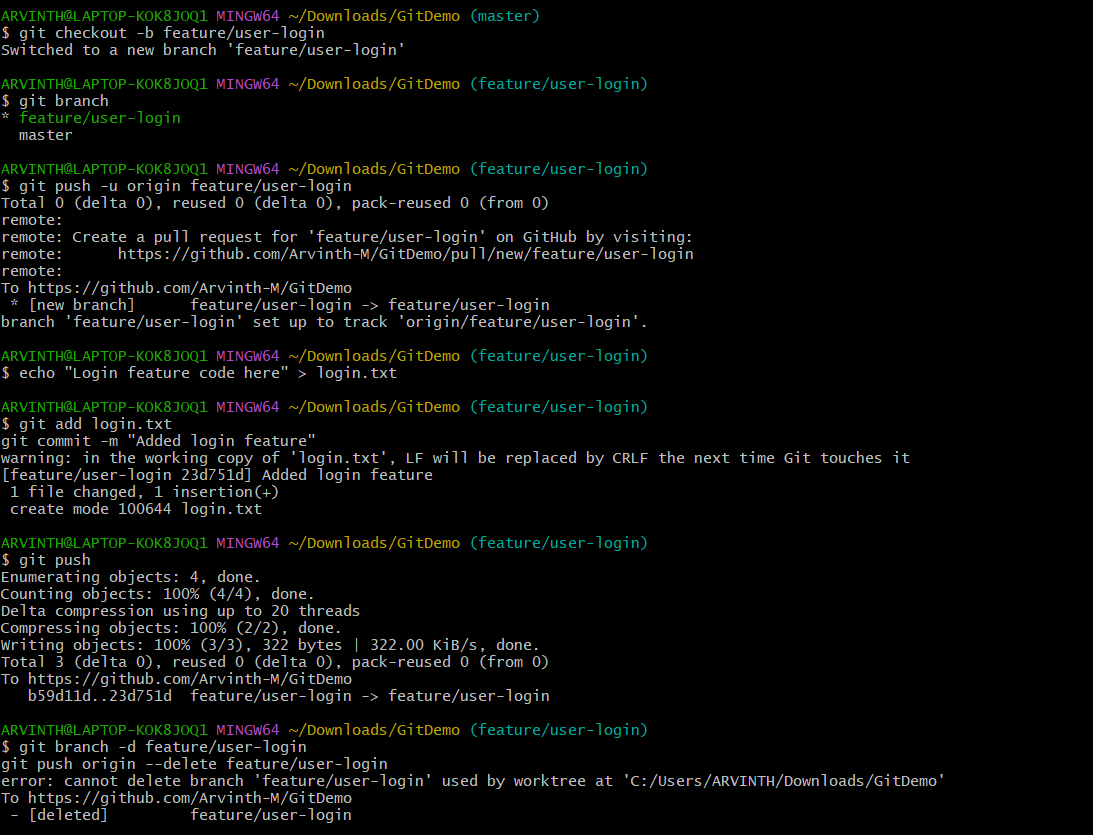
A **Merge Request (MR)** is a formal proposal to merge changes from one branch into another. It's a way to discuss and review changes before they are accepted into the main codebase.

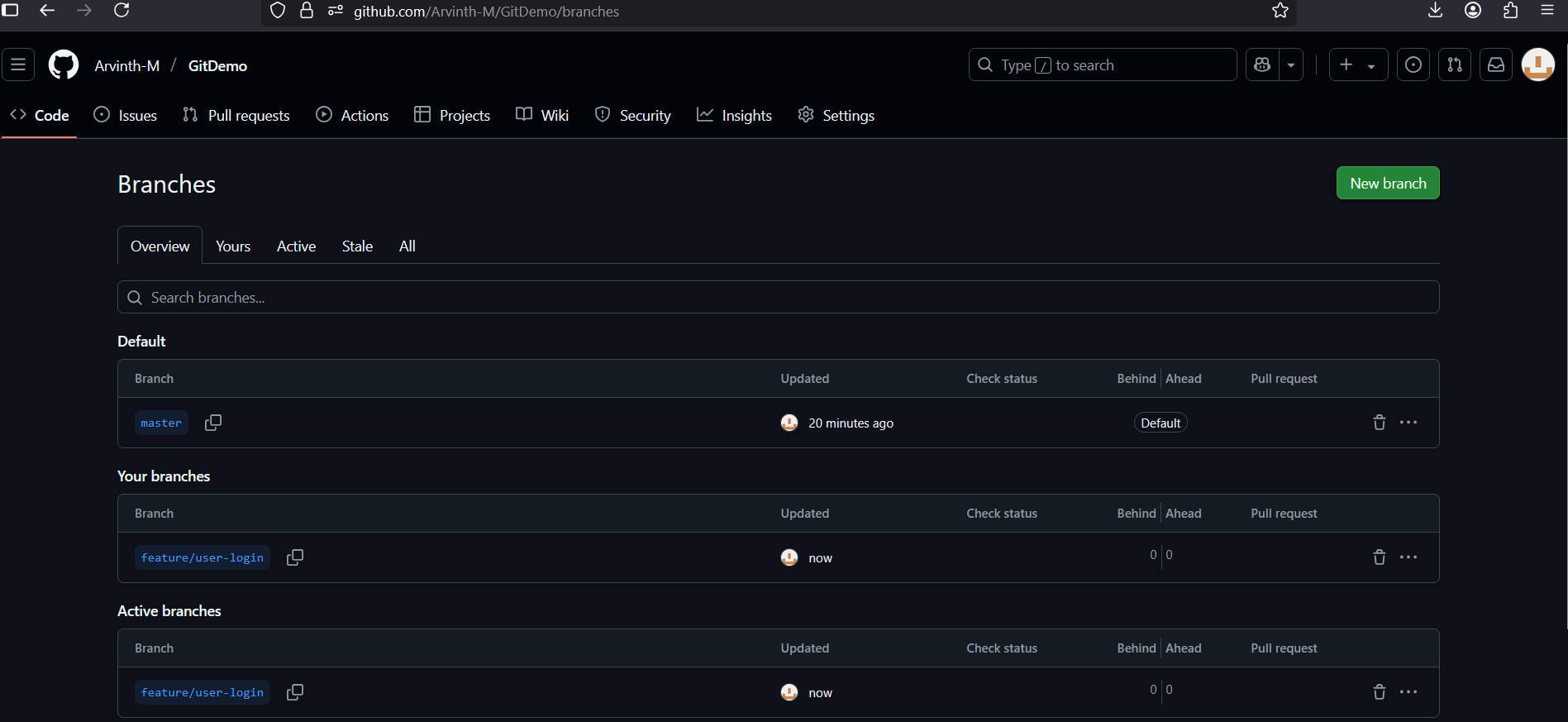
To create a merge request in GitLab:

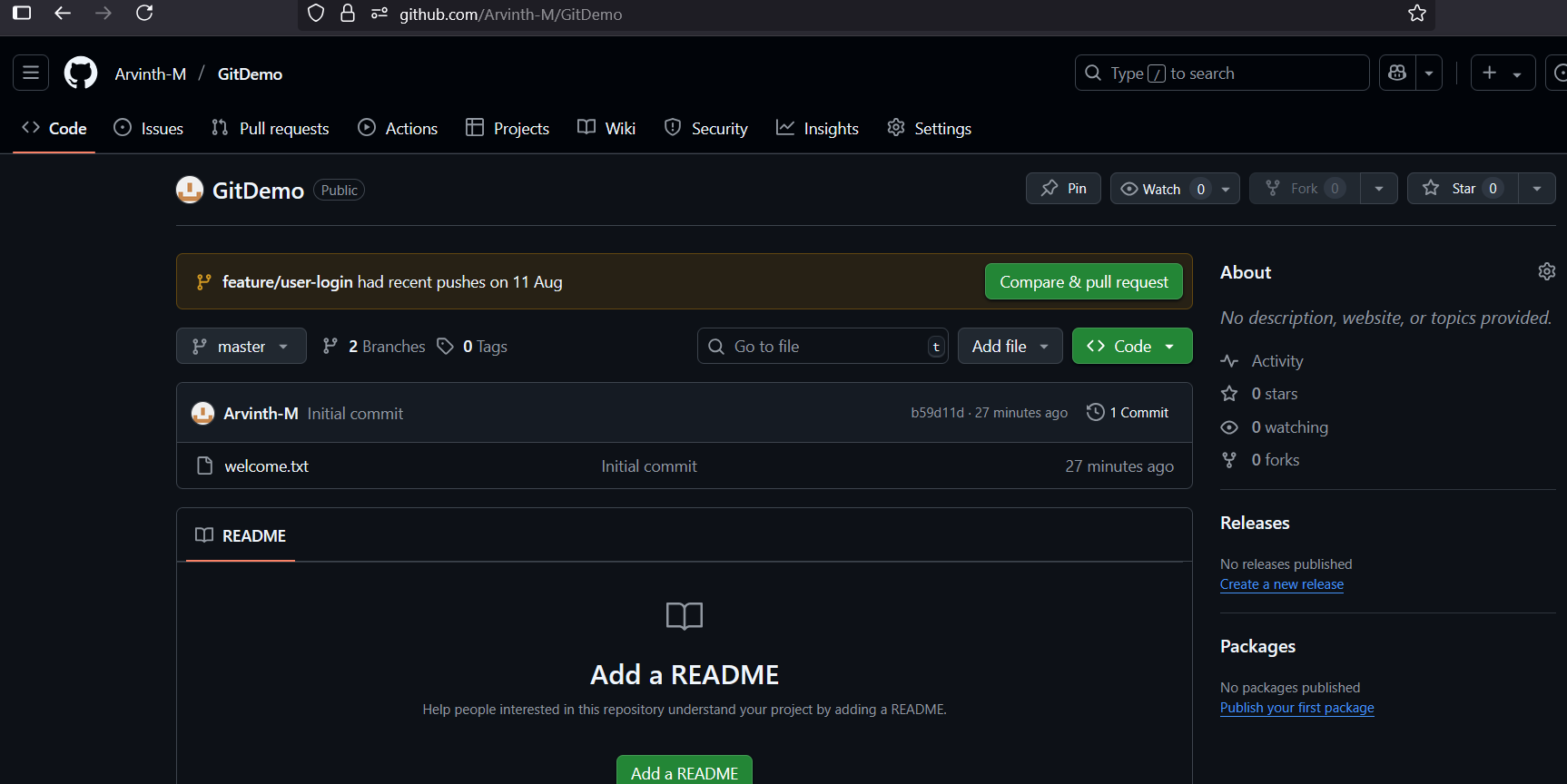
1. After pushing your new branch to the remote repository, a banner will appear in the GitLab UI with a button to "**Create merge request**". Click it.
2. Alternatively, go to the "**Merge requests**" section of your project and click "**New merge request**".
3. Choose the **source branch** (your new branch) and the **target branch** (the branch you want to merge into, like main).
4. Fill out the merge request form. Provide a descriptive title and a detailed description of your changes. You can also assign reviewers and a due date.
5. Click "**Create merge request**".

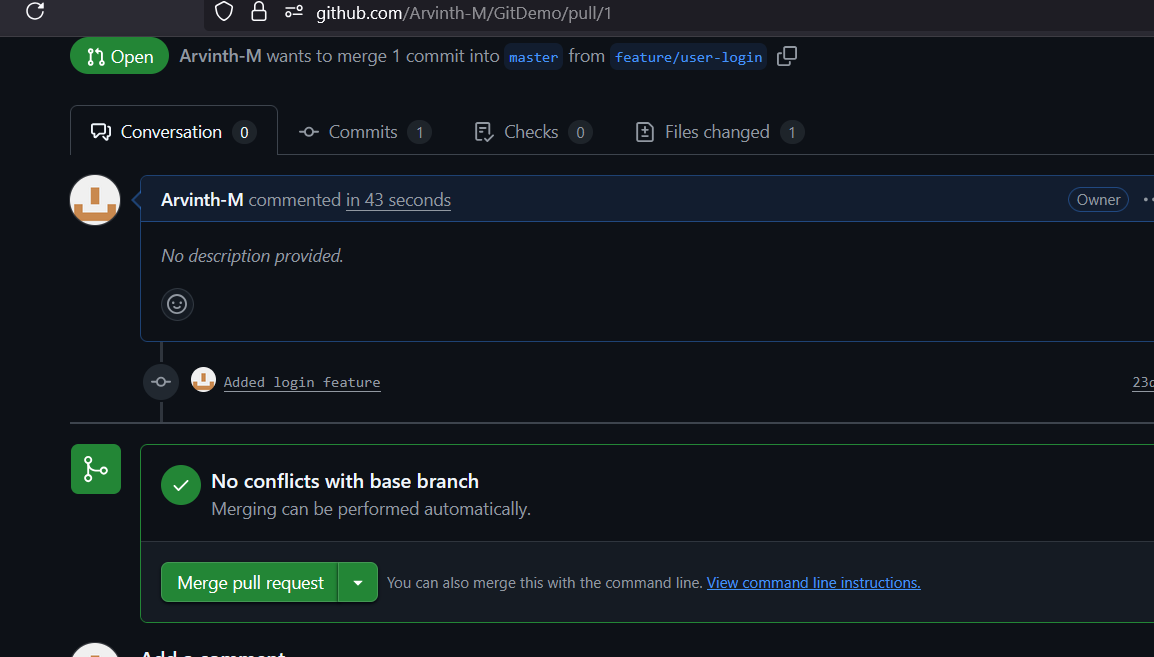
Once the merge request is created, your team members can review your code, leave comments, and suggest changes. After the code is approved, you can merge your changes into the target branch.

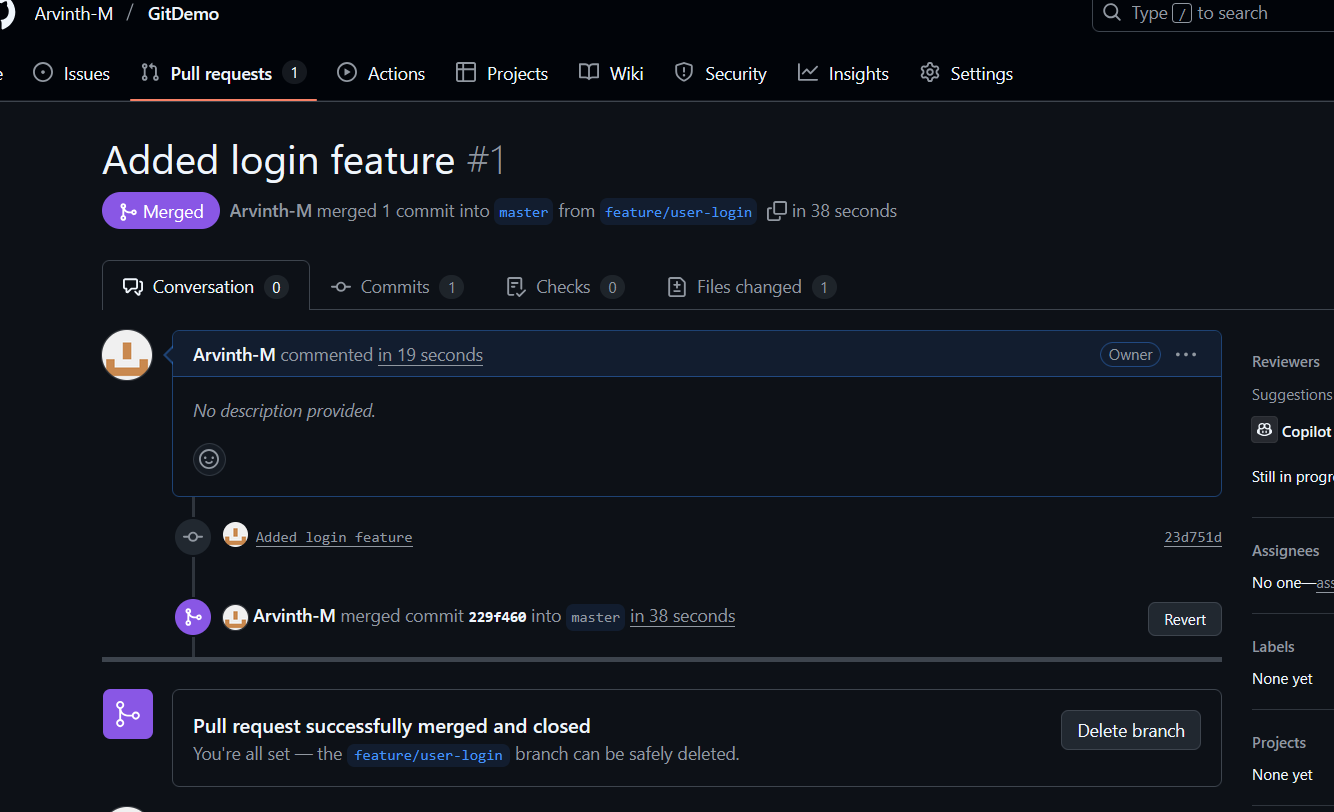
**Output:**

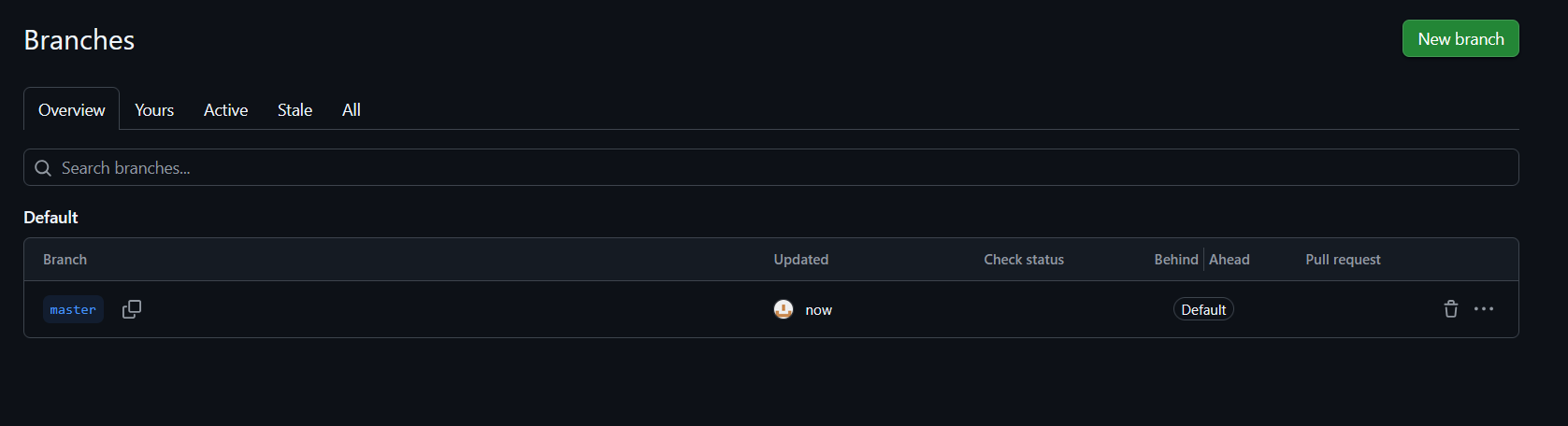












ARVINTH@LAPTOP-KOK8JOQ1 MINGW64 ~/Downloads/GitDemo (feature/user-login)

$ git status

On branch feature/user-login

Your branch is based on 'origin/feature/user-login', but the upstream is gone.

(use "git branch --unset-upstream" to fixup)

Untracked files:

(use "git add <file>..." to include in what will be committed)

.gitignore

nothing added to commit but untracked files present (use "git add" to track)