Hands-on: Simulate team PR flow with assigned roles

Objective:
Practice the typical collaborative Git workflow involving multiple team members with roles like Developer, Reviewer, and Integrator, using Pull Requests in Bitbucket Enterprise.

Setup

- **Team Members:**
  - Developer 1 (Feature branch author)
  - Developer 2 (Reviewer)
  - Integrator/Maintainer (Approves & merges PR)
- **Prerequisite:**
  All team members have Bitbucket Enterprise access and appropriate permissions (write access for Developers, merge rights for Integrator).

Step 1: Developer 1 Creates Feature Branch & Pushes Code

- Developer 1 clones the repo locally:

```
git clone https://bitbucket.yourcompany.com/scm/project/repo.git
cd repo

```

- Create and switch to feature branch:

```
git checkout -b feature/awesome-feature

```

- Make code changes, stage, and commit:

```
git add .
git commit -m "Add awesome feature"

```

- Push branch to Bitbucket:

```
git push origin feature/awesome-feature

```

Step 2: Developer 1 Creates a Pull Request

- Developer 1 logs into Bitbucket Enterprise web UI.
- Navigates to repository → **Pull Requests** → **Create pull request**.
- Selects source branch `feature/awesome-feature` and target branch `main` (or `develop`).
- Adds descriptive title and summary.
- Assigns **Reviewer** (Developer 2) and **Integrator** if possible.
- Creates the PR.

Step 3: Developer 2 (Reviewer) Reviews the PR

- Developer 2 opens the PR in Bitbucket GUI.
- Checks **Diff** tab for code changes.
- Adds inline comments or general feedback for improvements or questions.
- If issues found, requests changes using the PR interface.
- Developer 1 is notified to update code.

Step 4: Developer 1 Updates Code Based on Feedback

- Developer 1 makes the requested changes locally.
- Commits updates:

```
git add .
git commit -m "Address review comments"

```

- Pushes updates to the same branch:

```
git push origin feature/awesome-feature

```

- Bitbucket automatically updates the existing PR with new commits.

Step 5: Developer 2 Approves the PR

- Once satisfied, Developer 2 clicks **Approve** in the PR interface.

Step 6: Integrator Merges the PR

- Integrator reviews final PR status (approvals, CI build results).
- Clicks **Merge** button on the PR page.
- Selects merge strategy per team standards (merge commit, squash, or rebase).
- Confirms merge.
- Optionally deletes feature branch post-merge.

Step 7: Team Sync

- Everyone pulls latest changes to local repos:

```
git checkout main
git pull origin main

```

Summary:

- Developer 1 writes and pushes feature branch code.
- Developer 1 creates PR and assigns Reviewer.
- Developer 2 reviews and requests changes or approves.
- Developer 1 updates code if needed.

- Integrator merges PR after approval.

This flow simulates a typical collaborative PR process in an enterprise environment using Bitbucket.