Hands-on: Break → Troubleshoot → Fix Git Issues

Objective:

Simulate common Git mistakes and issues, learn how to identify them, and fix them using appropriate commands and workflows.

✅ Pre-requisites:

- Local Git setup
- A test repository cloned from Bitbucket Enterprise
- Two team members (can be simulated by switching user identity locally)

🔧 Scenario 1: Uncommitted Work Lost During Checkout

Break:

1. Make changes to a file without committing:

   ```
   echo "test change" >> index.html
   ```

2. Attempt to switch branches:

   ```
   git checkout main
   ```

Troubleshoot:

- Git will refuse checkout with warning: “Your local changes to the following files would be overwritten…”

Fix:

- Option 1 (preserve changes):

  ```
  git stash
  git checkout main
  git stash pop
  ```

- Option 2 (discard changes):

  ```
  git checkout -- index.html
  git checkout main
  ```

🔧 Scenario 2: Wrong Commit Message or File

Break:

1. Commit the wrong file or message:

   ```
   git add config.js
   git commit -m "Fix styling"
   ```

Troubleshoot:

- Realize `config.js` wasn’t meant to be committed, and message is wrong.

Fix:

- Amend commit (if not pushed yet):

```
git reset --soft HEAD~1
git reset config.js
git commit -m "Correct: Fix styling for homepage"

```

🔧 Scenario 3: Merge Conflict

Break:

1. On `main`, update `README.md` and commit.
2. On `feature`, update the same line in `README.md` and commit.
3. Try to merge `main` into `feature`:

```
git checkout feature
git merge main

```

Troubleshoot:

- Git shows conflict in `README.md`.

Fix:

1. Open the file and resolve the conflict manually.
2. Mark it as resolved:

```
git add README.md
git commit

```

🔧 Scenario 4: Force Push Gone Wrong

Break:

1. Rebase and force push to remote:

```
git rebase main
git push --force

```

2. Collaborator's local branch is now out of sync.

Troubleshoot:

- Collaborator gets error: “Updates were rejected because the remote contains work...”

Fix (for collaborator):

- Sync with force-updated branch:

```
git fetch
git reset --hard origin/feature-branch

```

🔧 Scenario 5: Detached HEAD State

Break:

1. Checkout a commit hash:

```
git checkout <commit-hash>
```

Troubleshoot:

- Git shows: "You are in 'detached HEAD' state."

Fix:

- Create a new branch from detached state:

```
git checkout -b hotfix-temporary
```