DevOps

Caltech | Center for Technology & Management Education

Post Graduate Program in DevOps

simplilearn

Remote Repositories

Learning Objectives

By the end of the lesson, you will be able to:

- Explain GitHub repository using HTTPS and SSH
- Define fork, pull requests, and pulling commits
- Implement multiple commits, merge file changes, and track issues
- Define upstream, downstream, and tags

Creating a Repository in GitHub using HTTPS and SSH

GitHub: Using HTTPS

The steps to create a repository in GitHub using HTTPS are:

1. Log In to Github
2. Create a new repository
3. Follow GitHub instructions
   - Create a symbolic link to GitHub
   - Push from local repository to GitHub

Assisted Practice

Create a Repository in GitHub Using HTTPS

Problem Statement: Create public repositories for an open-source project. When creating your public repository, make sure to use a credential helper so Git will remember your GitHub username and password every time it talks to GitHub

Steps to Perform:

1. Creating a repository on the local machine
2. Creating a GitHub repository
3. Adding a remote repository using the HTTPS URL
4. Pushing the changes in the local repository to GitHub
5. Checking the status of the local and remote repository

GitHub: Using SSH

The steps to create a repository in GitHub using SSH are:

1. Create a local repository
2. Create SSH key (ssh-keygen)
3. Configure GitHub with SSH public key
4. Create a local repository
5. Create a local repository

NOTE

The SSH key helps you create a repository without a username and password.

Assisted Practice

Create a Repository in GitHub Using SSH

Problem Statement: Create public repositories for an open-source project. When creating your public repository, make sure to use a secure connection without using username and password.

Steps to Perform:

1. Generating a new SSH key
2. Adding the SSH key to the GitHub account
3. Creating a repository on the local machine
4. Creating a GitHub repository

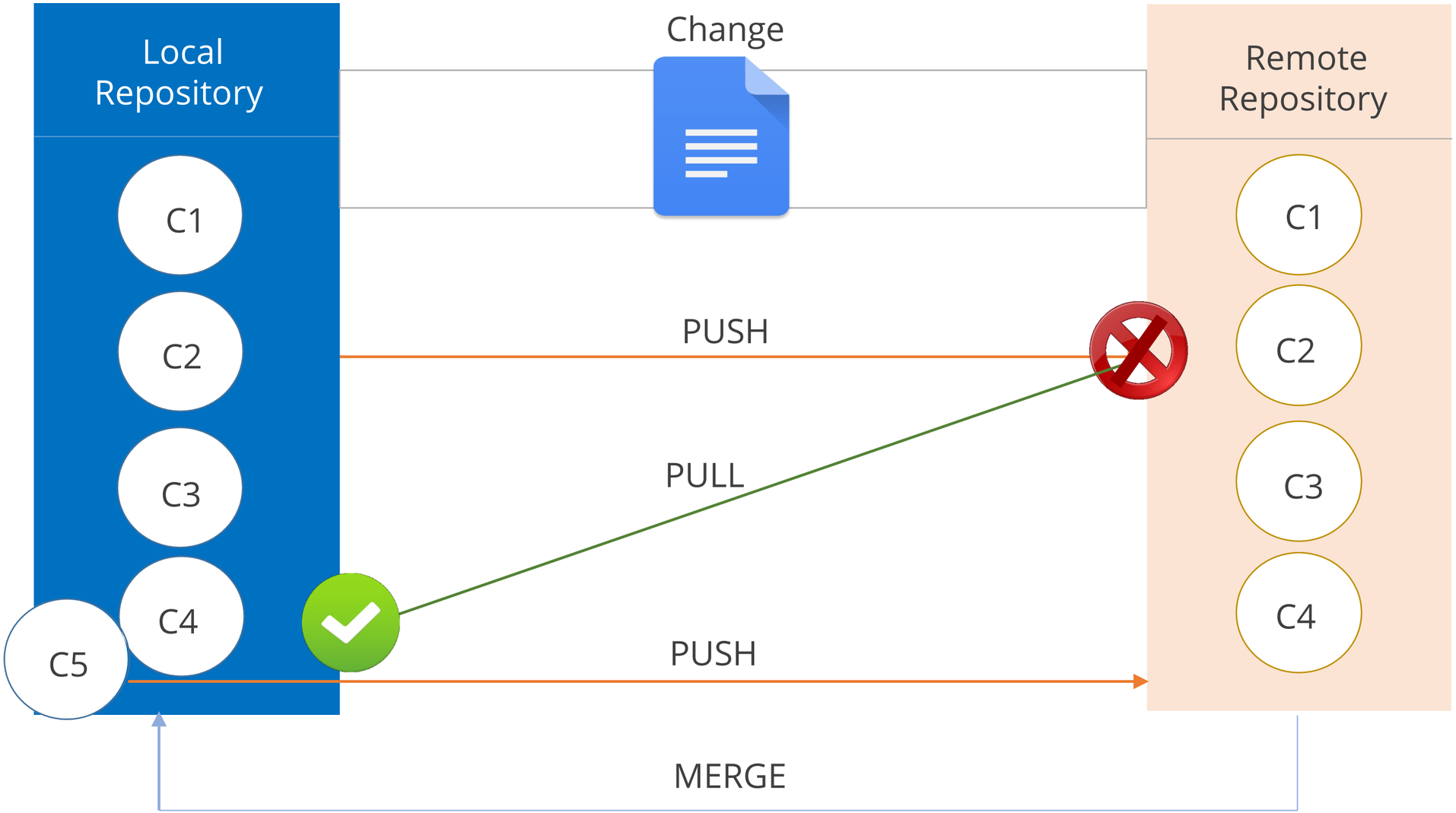
Assisted Practice

Create a Repository in GitHub Using SSH

5. Adding a remote repository using the SSH URL
6. Pushing the changes in the local repository to GitHub
7. Checking the status of the local and remote repository

Managing Multiple Commits in Git

How does Git manage changes?



Assisted Practice
Managing Multiple Commits in Git

Problem Statement: The web designing team wants to create a page about writing Git commit messages for the team practices website.

Steps to Perform:

1. Checking the status of repository
2. Checking the logs for commits history
3. Editing the index.html file
4. Adding an extra header in the index.html file
5. Checking the repository status for any updates

Assisted Practice
Managing Multiple Commits in Git

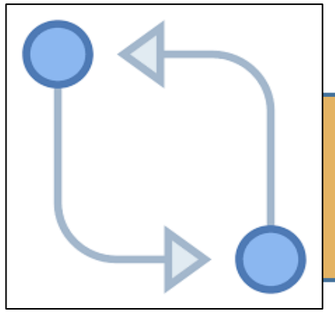
6. Adding the updated file in the staging area and committing the changes
7. Checking the logs for the latest commits
8. Pushing the changes to the remote repository
9. Editing file on the remote repository
10. Fetching the changes from remote repository
11. Checking the logs for all the commits history

Fork and pull Requests in GitHub

GitHub: Fork and Pull

Fork

Creates a new repo in your GitHub account



Pull

Updates the repo in your local project by running the commands

GitHub: Fork and Pull

The steps to create a repository in GitHub using fork and pull requests are:

1. Create a fork
2. Clone your fork
3. Modify the code
4. Push your changes
5. Create a pull request

Assisted Practice

Create a Fork and Pull Request

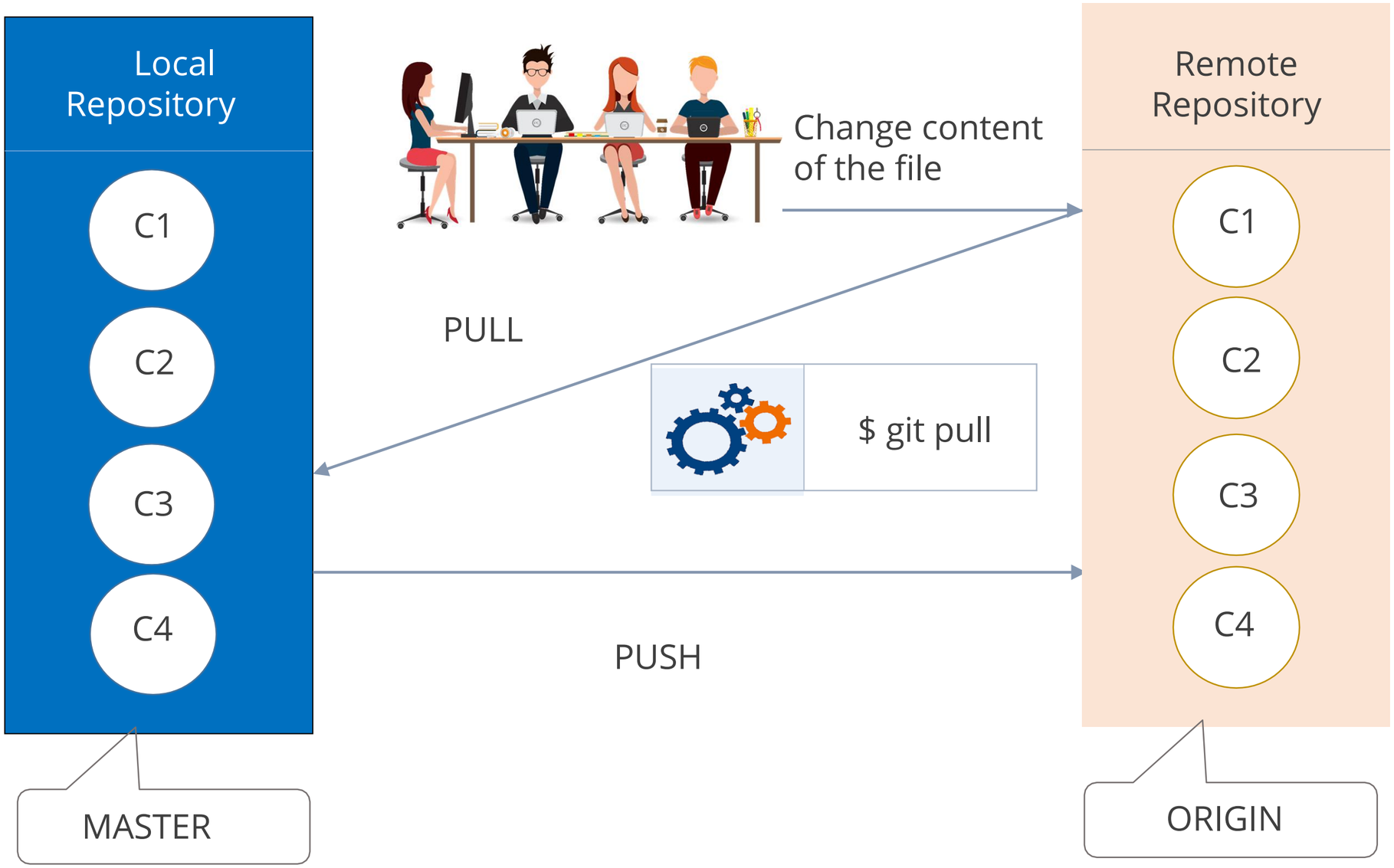
Problem Statement: The team has been assigned a task as per the client request where they have to create a pull request but don't have to work on multiple pull requests to the same repository at once.

Steps to Perform:

1. Create a Fork
2. Clone your Fork
3. Modify the Code
4. Push your Changes
5. Create a Pull Request

Pulling Commits from GitHub

GitHub: Pulling Commits



Assisted Practice

Pulling Commits from GitHub

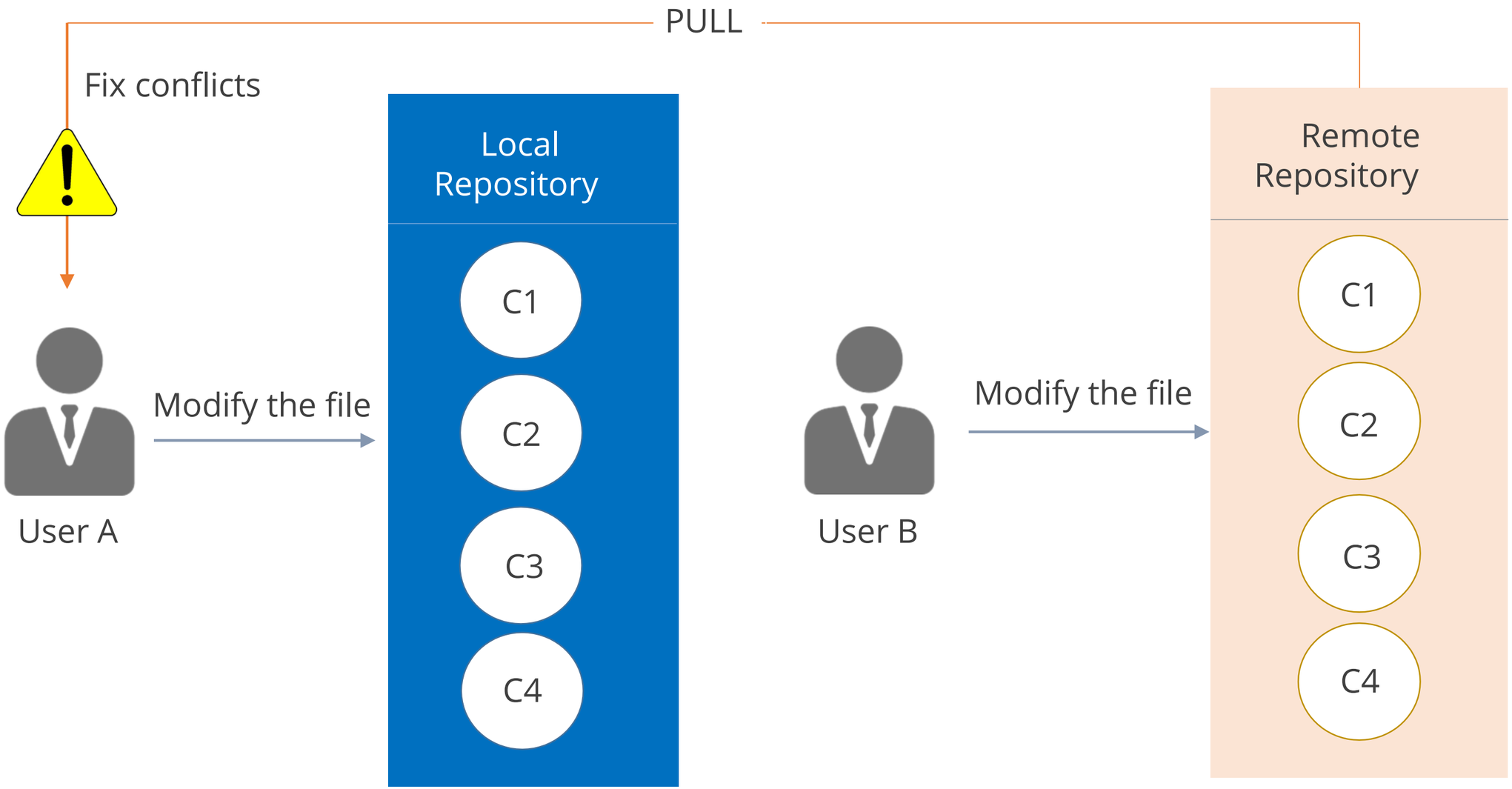
Problem Statement: During scrum the manager has advised the associate of the firm to create a local tracking branch that's associated with a remote branch.

Steps to Perform:

1. Updating the remote repository
2. Checking the commits
3. Checking the status of the local repository
4. Checking the logs for commits history
5. Pulling the main branch from the remote repository
6. Checking the logs for the latest commits

Merging File Changes in Git

Scenario of Merging File Changes in Git



Assisted Practice
Merging File Changes in Git

Problem Statement: You have assigned a task to combine separate changes to an original in Git.

Steps to Perform:

1. Verifying the setup of the remote repository
2. Pulling files from remote repository
3. Editing a file in remote repository
4. Merging the changes in edited file in the local and remote repository

Issue Tracking in GitHub

Creating and Tracking Issues

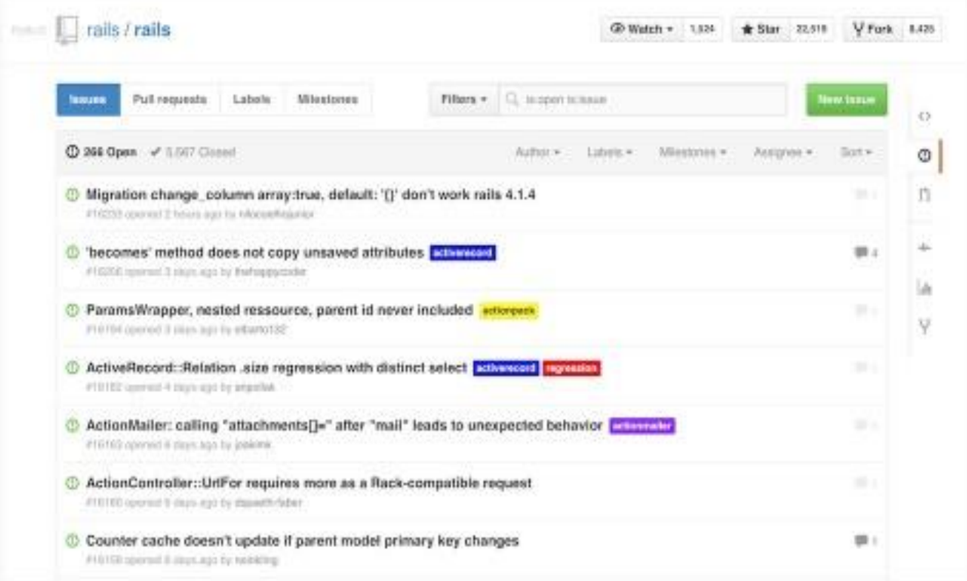
Integrated issue tracking.

A flexible issue tracker lets you stay on top of bugs and focus on features.

Issue listings | Milestones & labels | Commit keywords

Your project's issues page can be as simple or as sophisticated as you like. Filter by open and closed issues, assignees, labels, and milestones. Sort by issue age, number of comments, and update time.

- Keyboard shortcuts make issue assignment and labeling fast.
- Only **teammates** and **collaborators** can create and view issues on private repositories.
- **Anyone** may create and view issues on public repositories.



View the issues page for rails/rails.

Assisted Practice
Creating and Tracking Issues

Problem Statement: You have been assigned a task to do a walkthrough of creating and tracking issues.

Steps to Perform:

1. Selecting issues tab
2. Creating a new label
3. Creating a milestone
4. Creating a new issue
5. Commenting on the issue created

Understand Upstream and Downstream

Downstream and Upstream

Local repository

Downstream

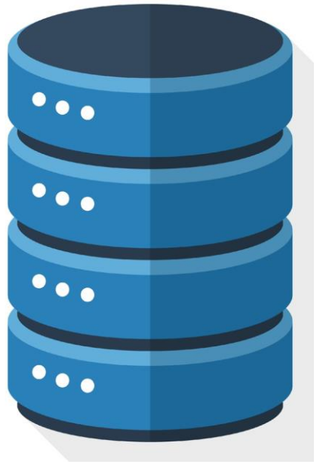
Direct access through the file system

\\shared\path

Remote access through listener

query

Remote repository



Upstream

Merge contributions on the official history

Merge contributions into the official history

Assisted Practice
Getting Started with Git Upstream

Problem Statement: Your team were forking projects but missed out sending it back to the parent repository which means you're at risk for losing track of them. You have to find a solution to make sure contributors are drawing from the same place.

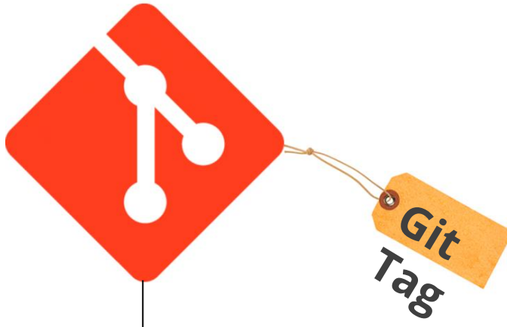
Steps to Perform:

1. Verifying the setup of remote repository
2. Adding upstream to the remote repository
3. Verifying if the remote repository is added correctly
4. Fetching upstream
5. Updating local branch with respect to the upstream branch

Creating and Verifying Tags

Introduction to Tags

- Used to label and mark a specific commit in the history
- Indicate release versions with the release name



Lightweight tag

- Points directly to a specific commit in the history
- Can only add tagger's name

Annotated tag

- Points directly to a specific commit
- Can add comments, signature, date, tagger's name, and email ID

Different Commands in Tags

Command	Explanation
-a/--annotate	Create an unsigned and annotated tag object
-s/--sign	Create a GPG-signed tag using the default email address key
--no-sign	Override **tag.gpgSign** configuration variable that is set to force each and every tag to be signed
-u <keyid>/--local-user=<keyid>	Create a GPG-signed tag, using the given key
-f/--force	Replace an existing tag with the given name (instead of failing)
-d/--delete	Delete existing tags with the given names
-v/--verify	Verify the GPG signature of the given tag names

Assisted Practice
Create and Delete Tags

Problem Statement: Write a command to create and delete a tag.

Steps to Perform:

1. Creating a tag
2. Listing all the tags
3. Adding a description to your tag
4. Deleting a tag

Key Takeaways

- Pull updates the repo on local system and Fork helps to create repo on a GitHub account.
- A local repository can be connected to one or more remote repositories.
- Files can be committed, merged, and used to track issues.
- Tag is used to label and mark a specific commit in the history.

Lesson-End Project

Remote Repositories

Problem Statement:

Fork upstream repo which will cover concepts like creation, issue, pull, and merge.