|  |  |  |
| --- | --- | --- |
| SCHOOL OF INFORMATION AND TECHNOLOGY | | |
| NAME: MERCADO, CASSIE KYM M. | DATE PERFORMED: NOVEMBER 21, 2024 |  |
| Section: BSIT – IDC2 | DATE SUBMITTED: NOVEMBER 26, 2024 |

# SYSADM1 – Git Basics

Answer the following research questions about Git, GitLab desktop and GitHub.

1. What is Git, and why is it important in software development?

Git is a distribution version control system or dVCS that was created to monitor source code modifications during software development. It is important in software development as it facilitates effective teamwork while giving developers the ability to control and manage over different versions of a project. Git allows several software developers to work on the same project at once, minimizing disputes and guaranteeing that their modifications are seamlessly incorporated. This makes Git an essential tool in software development as it supports version control, teamwork, and an organized management of code contributions.

1. How does Git track changes in a project?

Git tracks changes and modifications made to a project by taking snapshots of the project files and keeping them in a repository. As the user edits their files, Git detects them as modified since the last commit. Once committed, Git keeps a record of the differences between the current version and the previous version by keeping a thorough and detailed history of the project's development and documenting the changes made.

1. What is the difference between a local repository and a remote repository in Git?

A local repository contains all of the project files and version history that are kept on a developer's computer. On the other hand, a remote repository, which is hosted on a server or platform like GitHub and GitLab, is shared by multiple team members. Setting up a remote repository facilitates cooperation by enabling developers to exchange information and synchronize changes across regions. Since they are hosted on a local computer for a single user, this is not visible in a local repository.

1. What are the basic Git commands?

|  |  |
| --- | --- |
| **Basic Git Commands** | |
| **Command** | **Description** |
| git init | Used to create an empty Git repository. |
| git add | Used after checking the status of the files, to add those files to the staging area. Before running the commit command, "git add" is used to add any new or modified files. |
| git commit | Makes sure that the changes are saved to the local repository. |
| git status | Tells the current state of the repository. |
| git config | Used initially to configure the user.name and user.email. This specifies what email id and username will be used from a local repository. |
| git branch | Used to determine what branch the local repository is on. Also enables adding and deleting a branch. |
| git merge | Used to integrate the branches together. The command combines the changes from one branch to another branch. It is used to merge the changes in the staging branch to the stable branch. |
| git clone | Used to create a local working copy of an existing remote repository |
| git pull | Used to fetch and merge changes from the remote repository to the local repository. |
| git push | Used to transfer the commits or pushing the content from the local repository to the remote repository. |

1. How do you check the status of a Git repository?

To check the status of a Git repository, use the ***git status*** command. This command allows us to see which files are not tracked by Git, which changes have been staged, and which have not.

1. What is the purpose of branches in Git, and how do you create and switch between them?

The purpose of branches in Git is to separate our work for each task. This allows developers to work on different features as it helps us isolate and work on one part of the project without directly impacting the main codebase.

To create a branch, use ***git branch <branch-name>.*** To switch between branches, use ***git checkout <branch-name>*** or ***git switch <branch-name>.***

1. What are GitLab Desktop and GitHub, and how are they different from Git?

Git is a distributed version control system that manages project versions and tracks code changes locally on a developer's device. On the other hand, web-based platforms GitHub and GitLab host repositories and provide Git-based version control, project management, and collaboration tools. In contrast to Git, which is based on a Command Line Interface (CLI), GitHub and GitLab include built-in Graphical User Interface (GUI) support in addition to CLI support, which simplifies development processes and makes collaboration simpler.

1. How do you connect a local Git repository to a GitLab or GitHub repository?

To connect a local Git repository to a GitLab or GitHub remote repository, use the ***git remote add origin <repository-URL> command***. This sets up the link between the local repository and the remote one hosted on GitLab or GitHub.

1. What are the steps to collaborate with others using GitLab or GitHub?

To collaborate and invite others using GitLab or GitHub, follow these steps:

* Ask for the username of the person you're inviting as a collaborator.
* Navigate to the main page of the repository.
* Under your repository name, click Settings. If you cannot see the "Settings" tab, select the ... dropdown menu, then click Settings.
* In the "Access" section of the sidebar, click Collaborators.
* Click Add People.
* In the search field, start typing the name of person you want to invite, then click a name in the list of matches.
* Click Add NAME to REPOSITORY.
* The user will receive an email inviting them to the repository. Once they accept your invitation, they will have collaborator access to your repository.

1. How do you resolve merge conflicts in Git?

Merge conflicts happen when changes in two branches overlap. To resolve this issue, follow these steps:

* Identify conflicts using git status.
* Open conflicting files and manually edit them to resolve the differences.
* Stage the resolved files using git add.
* Commit the changes using git commit to finalize the resolution.

1. What is a pull request, and why is it used in GitHub?

A pull request is a proposal to combine a collection of changes from one branch to another. Before integrating the suggested modifications in to the man codebase, contributors can examine and discuss the changes in a pull request.

1. What are some best practices for writing commit messages?

Best practices for writing commit messages are:

* Write clear and concise summaries.
* Use the imperative mood (e.g., ‘Add feature’ instead of ‘Added feature’).
* Provide additional details in the body if necessary.
* Include relevant context such as issue numbers, feature requests, or discussions related to the changes.
* Mention bug fixes.
* Keep messages under 50 characters for the subject line and wrap text at 72 characters for the body.

**REFERENCES**

Afreen, S. (2024). *All the Git commands you need to know about.* Retrieved from: https://www.simplilearn.com/tutorials/git-tutorial/git-commands

Git-scm. (n.d.). *2.2 Git basics – Recording changes to the repository.* Retrieved from: https://git-scm.com/book/ms/v2/Git-Basics-Recording-Changes-to-the-Repository#:~:text=As%20you%20edit%20files%2C%20Git,changes%2C%20and%20the%20cycle%20repeats.

GitHub Docs. (n.d.). *About pull requests.* Retrieved from: https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-pull-requests

GitHub Docs. (n.d.). *Inviting collaborators to a personal repository.* Retrieved from: https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal-account-on-github/managing-access-to-your-personal-repositories/inviting-collaborators-to-a-personal-repository

Holocombe, J. (n.d.). *How to inspect a Git repository on GitHub.* Retrieved from: https://www.greengeeks.com/tutorials/inspect-a-git-repository/#:~:text=You%20can%20inspect%20a%20Git,t%20being%20tracked%20by%20Git.

Jackson-Barnes, S. (2024). *Git vs. GitHub vs. GitLab: Benefits, differences, and how to choose the right one.* Retrieved from: https://www.orientsoftware.com/blog/git-vs-github-vs-gitlab/#:~:text=For%20example%2C%20both%20GitHub%20and,the%20differences%20between%20these%20services.

Kirinyet, B. (2023). *9 Best practices for writing good Git commit messages.* Retrieved from: https://www.linkedin.com/pulse/7-best-practices-writing-good-git-commitmessages-kirinyet-brian

Kurata, D. (2023). *How to work with branches in Git.* Retrieved from: https://www.freecodecamp.org/news/how-to-work-with-branches-in-git/

Nulab. (n.d.). *Repositories.* Retrieved from: https://nulab.com/learn/software-development/git-tutorial/git-basics/repositories/#:~:text=A%20remote%20repository%20is%20hosted,machine%20for%20an%20individual%20user.

Worsley, S. (2023). *What is Git? – The complete guide to Git.* Retrieved from: https://www.datacamp.com/blog/all-about-git