TASK1

Basic Concept of Git and Terminologies:

Git is a free, distributed version control system which allows programmers to keep track of code changes, via "snapshots" (commits), in its current state. Utilizing commits allows programmers to test, debug, and create new features collaboratively. All commits are kept in what is known as a "Git Repository" that can be hosted on your computer, private servers, or open source websites,such as Github.
Git has 3 different "areas" for code:
Working directory: The area that you will be doing all of your work in (creating, editing, deleting, and organizing files).
Staging area: The area where you will list the changes that you have made to the working directory.
Repository: Where Git permanently stores the changes you have made as different versions of the project.
Advantages of Git: Free and open source
Fast and small
Implicit backup
Security
Easier branching

1. Git Repository:
A Git repository is a directory that contains a Git working directory. The working directory contains all the files that are tracked by Git. The Git repository is the place where the files are stored.
2. Git Working Directory:
A Git working directory is a directory that contains all the files that are tracked by Git.
3. Branch:
A branch is a version of the repository that diverges from the main working project. A Git project can have more than one branch.
4. Checkout:
The git checkout command is used to switch between branches in a repository.
5. Merge:
Merge is a command that is used to merge the changes from one branch to another branch.
6. Push:

Push operation copies changes from a local repository instance to a remote one.This is used to store the changes permanently into the Git repository. This is the same as the commit operation in Subversion.
7. Pull:
Pull operation copies the changes from a remote repository instance to a local one. The pull operation is used for synchronization between two repository instances. This is the same as the update operation in Subversion.
8. Fetch:
It is used to fetch branches and tags from one or more other repositories, along with the objects necessary to complete their histories. It updates the remote-tracking branches.
9. Clone:
The git clone is a Git command-line utility. It is used to make a copy of the target repository or clone it.
10. Master:
Master is a naming convention for the Git branch. It's a default branch of Git.
11. HEAD:
HEAD is a pointer, which always points to the latest commit in the branch.
Whenever you make a commit, HEAD is updated with the latest commit. The heads of the branches are stored in .git/refs/heads/ directory.
12. Tags:
Tag assigns a meaningful name with a specific version in the repository. Tags are very similar to branches, but the difference is that tags are immutable.
13. Upstream And Downstream:
In Git terms, think of Upstream as the changes that you have made and now you want to send(Upstream) those changes to get merged into the project. And When you download or clone a repository and information is flowed (Downstream) from the repository to you.

Basic Commands Of Git:

1.SETUP: Configuring user information used across all local repositories

- git config --global user.name “[firstname lastname]”
  set a name that is identifiable for credit when review version history
- git config --global user.email “[valid-email]”
  set an email address that will be associated with each history marker
- git config --global color.ui auto
  set automatic command line coloring for Git for easy reviewing

2. SETUP & INIT: Configuring user information, initializing and cloning repositories

- git init
  initialize an existing directory as a Git repository
- git clone [url]
  retrieve an entire repository from a hosted location via URL

3. STAGE & SNAPSHOT: Working with snapshots and the Git staging area

- git status
  show modified files in working directory, staged for your next commit
- git add [file]
  add a file as it looks now to your next commit (stage)
- git add*
  Add all files of a repo to staging (Index) area
- git diff
  diff of what is changed but not staged
- git diff --staged
  diff of what is staged but not yet committed
- git commit -m “[descriptive message]”
  commit your staged content as a new commit snapshot

4. BRANCH & MERGE: Isolating work in branches, changing context, and integrating changes

- git branch
  list your branches. a * will appear next to the currently active branch
- git branch [branch-name]
  create a new branch at the current commit
- git checkout
  switch to another branch and check it out into your working directory
- git merge [branch]
  merge the specified branch’s history into the current one

5. COMMIT HISTORY:

- git log
  Display the most recent commits and the status of the head
- git reflog
  List operations (e.g. checkouts or commits) made on local repositories.

6. TEMPORARY COMMITS:Temporarily store modified, tracked files in order to change branches

- git stash
  Save modified and staged changes
- git stash list
  list stack-order of stashed file changes
- git stash pop
  write working from top of stash stack
- git stash drop
  discard the changes from top of stash stack

7. REVERTING CHANGES:

- git reset –hard
  Reset the change.When --hard is used,all changes are discarded.
- git revert
  Undo the changes

8. SHARE & UPDATE: Retrieving updates from another repository and updating local repos

- git remote add [alias] [url]
  add a git URL as an alias
- git fetch [alias]
  fetch down all the branches from that Git remote
- git merge [alias]/[branch]
  merge a remote branch into your current branch to bring it up to date
- git push [alias] [branch]
  Transmit local branch commits to the remote repository branch
- git pull
  fetch and merge any commits from the tracking remote branch

9. REMOVING FILES:

- git rm <file Name>
  Remove the files from the working tree and from the index.

Concepts of GITHUB,GitLab,BitBucket

GitHub:

It is a git-based repository hosting platform, currently owned by Microsoft.
It only hosts projects that use the Git VCS
It is free for public repositories and for private ones it is paid.
It comes with its own Wiki and issues tracking system.
We can decide if someone gets read or write access to a repository.
It's the largest repository host with more than 38+ million projects.
It provides support for an online web-based VS code editor.
It has size limitations. The file size can't be more than 100 Mb while the repository can host 2GB of information.
Some remarkable features of GitHub are:

Commit History can be seen
Graphs: pulse, contributors, commits, code frequency, members of it.
Pull requests with code review and comments
Issue Tracking
Email notifications

GitLab:

It has free private repositories that Github doesn't.
GitLab Community Edition is free and open-sourced.
We can set and modify people's permissions according to their roles.
In this, you can attach any file to any issue which can't be done inside GitHub.
The source code of GitLab Community Edition is available on their website.
It has a relatively slow interface.
It supports Git import.
Some remarkable features of GitHub are:

Issue Tracker
Commit graph and reporting tools
Create new issues from the Issue Board
Ease of migration from other providers

Bitbucket:

It supports the Mercurial VCS(version control system) in addition to Git.
It is not open source but by buying the self-hosted version the full source code is provided.
Bitbucket is written in Python and uses the Django web framework.

We get free private repositories on Bitbucket.
It offers both commercial plans and free accounts. It offers free accounts with an unlimited number of private repositories.
Imports existing Git projects from Excel, Github, etc.
Some remarkable features of Bitbucket are

Issue tracking
REST APIs to build third-party applications which can use any development language
Code search is possible
Pull requests with code review and comments
Snippets that allow developers to share code segments or files

Industrial Practices Of Using Git:

- Version Control: Git is utilized in industrial settings for version control, enabling teams to track changes in code over time, facilitating collaboration among developers.
- Branching Strategies: Teams adopt branching strategies like GitFlow or GitHub Flow to manage development workflows effectively, allowing for the isolation of features or bug fixes.
- Main Branch Management: Maintaining a stable main branch is crucial, with periodic merges of feature branches and regular releases to ensure a reliable and functional codebase.
- Continuous Integration (CI): CI tools integrate with Git to automate testing processes, ensuring that changes don't break the existing codebase and maintaining a high level of code quality.
- Automation: Git is often integrated into automated processes, including deployment pipelines, allowing for the seamless integration of new code changes into production environments.
- Configuration Management: Configuration files are versioned with Git, ensuring consistency and traceability of changes to settings and parameters across environments.
- History and Auditing: Git's comprehensive commit history provides a valuable resource for tracking changes, understanding development timelines, and aiding in debugging and auditing processes.
- Collaboration and Distributed Development: Git's distributed nature allows teams to work collaboratively, even in geographically dispersed environments, with each developer having a local copy of the entire repository.

Cloning a repository to local:

Cloning in Git refers to the process of creating a local copy of a remote repository. When you clone a Git repository, you are essentially copying all the files, commit history, and branches from a remote server (such as GitHub, GitLab, or Bitbucket) to your own machine. It accesses the repository through a remote URL.

Step 1: Open GitHub and navigate to the main page of the repository.

Step 2: Click “Code” and copy the given URL.

Step 3: Open “Git Bash” and change the current working directory to the location where you want the cloned directory.

Step 4: Type git clone in the terminal, paste the URL you copied earlier, and press “enter” to create your local clone.

```
git clone {repository URL}
```

Resources Used:

- Chatgpt
- GitHub Docs
- GeeksforGeeks
- javaTpoint