**ALEX XU**

**1. Basic Setup**

* git config --global user.name "Your Name": Set your name for commits.
* git config --global user.email "your.email@example.com": Set your email for commits.

**2. Creating Repositories**

* git init: Initialize a new Git repository in your project.
* git clone <repository-url>: Clone an existing repository to your local machine.

**3. Tracking Changes**

* git status: Show the status of your working directory (modified files, untracked files, etc.).
* git add <file>: Stage specific files for the next commit.
* git add .: Stage all changed files.
* git commit -m "Commit message": Commit your staged changes with a message.
* git commit -am "Commit message": Commit changes to tracked files with a message (skips git add).

**4. Branching and Merging**

* git branch: List all branches in the repository.
* git branch <branch-name>: Create a new branch.
* git checkout <branch-name>: Switch to another branch.
* git merge <branch-name>: Merge another branch into the current one.
* git checkout -b <branch-name>: Create and switch to a new branch in one command.

**5. Pushing and Pulling**

* git push origin <branch-name>: Push your changes to a remote repository.
* git pull origin <branch-name>: Pull the latest changes from a remote repository.

**6. Viewing Logs**

* git log: View the commit history.
* git log --oneline --graph --all: View a graphical representation of the commit history in one line per commit.

**7. Undoing Changes**

* git reset <file>: Unstage a file.
* git reset --hard: Reset your working directory to the last commit (discard changes).
* git checkout -- <file>: Discard changes in a specific file.

**8. Stashing**

* git stash: Temporarily save your changes (when you need to switch branches or pull updates).
* git stash apply: Re-apply the stashed changes.

**9. Collaboration**

* git remote add origin <repository-url>: Add a remote repository.
* git fetch: Fetch changes from a remote repository without merging.
* git rebase <branch-name>: Reapply commits on top of another base branch (useful for keeping branches up to date).

**10. Tagging**

* git tag <tag-name>: Create a tag for a specific commit (e.g., for releases).
* git push origin <tag-name>: Push the tag to a remote repository.

**Git SSH** (Secure Shell) is a method of securely connecting to remote Git repositories, such as those hosted on platforms like GitHub, GitLab, or Bitbucket. Instead of using HTTPS (which requires entering your username and password each time you push or pull changes), SSH uses a pair of cryptographic keys (public and private) to authenticate and securely communicate with the server.

Here’s how Git SSH works:

**How Git SSH Works:**

1. **SSH Key Pair**: SSH authentication relies on a pair of keys:
   * **Public Key**: This is shared with the Git hosting service (e.g., GitHub).
   * **Private Key**: This stays securely on your local machine and is used to verify your identity when you communicate with the server.
2. **SSH Authentication**: When you push or pull changes using SSH, Git uses your private key to verify your identity. If the public key stored on the remote repository matches your private key, the server allows access without needing a username or passwor