1. **Protocols, getting git on server, generating our SSH public key, Setting up the Server, Git Daemon**

Set Up a Git Server (SSH & Git Daemon)

**1. Install Git on the Server**

sudo apt install git -y  # Debian/Ubuntu

**2. Generate & Copy SSH Key (Local Machine)**

ssh-keygen -t rsa -b 4096

ssh-copy-id user@server

**3. Set Up Git User & Repo (Server)**

sudo useradd -m -s /bin/bash git

sudo su - git

mkdir repos && cd repos

git init --bare myrepo.git

**4. Enable Git Daemon (Optional, Read-Only Access)**

git daemon --reuseaddr --base-path=/home/git/repos --export-all --enable=receive-pack

**5. Clone & Push (Client Machine)**

git clone git@server:/home/git/repos/myrepo.git

git add . && git commit -m "Initial commit" && git push origin main

2. **Smart HTTP, Git Web, GitLab, Distributed Workflows, contributing to a project, Maintaining projects**

Git Essentials: Smart HTTP, GitWeb, GitLab & Workflows

**1. Smart HTTP (Git Over HTTPS)**

- Secure Git access via Apache/Nginx using git-http-backend.
- Supports anonymous read & authenticated write.

sudo apt install apache2 -y && sudo a2enmod cgi alias env && sudo systemctl restart apache2

**2. GitWeb (Web-Based Git Viewer)**

- Lightweight interface for browsing repositories.

sudo apt install gitweb -y && sudo systemctl restart apache2

Access: http://server/gitweb

**3. GitLab (Self-Hosted Git Platform)**

- Full DevOps suite like GitHub.

curl -fsSL https://packages.gitlab.com/install/repositories/gitlab/gitlab-ee/script.deb.sh | sudo bash

sudo apt install gitlab-ee -y && sudo gitlab-ctl reconfigure

4. **Distributed Workflows**

- Centralized: Single main repo.
- Feature Branch: New features on separate branches.
- Forking: Contributors fork & submit pull requests.
- GitFlow: Uses main, develop, feature, release, hotfix branches.

**5. Contributing to a Project**

git clone https://github.com/user/project.git && cd project

git checkout -b feature-branch

git add . && git commit -m "New feature"

git push origin feature-branch

Submit a Pull Request.

**6. Maintaining a Git Project**

- Merge PRs, tag releases, manage issues.

git tag -a v1.0 -m "Version 1.0" && git push origin v1.0

### 3. Git diff, Viewing working directory changes, Visualizing diffs with GUI

Git Diff & Viewing Changes

**1. Check Differences**

- Unstaged changes: git diff
- Staged changes: git diff --staged
- Between commits: git diff commit1 commit2

**2. View Working Directory Changes**

- Status of modified files: git status
- Changes in a specific file: git diff filename

**3. Visualizing Diffs (GUI)**

- Use Git's GUI: gitk
- External tools: git difftool (supports Meld, KDiff3, etc.)

### 4. Git Stash & Pop, Checking out old commits, undoing changes exercise

Git Stash, Checkout & Undoing Changes

**1. Stash & Restore**

- Save: git stash
- List: git stash list
- Restore & remove: git stash pop
- Restore only: git stash apply

**2. Checkout Old Commits**

- View history: git log --oneline
- Switch: git checkout <commit-hash>
- Return: git checkout main

**3. Undo Changes**

- Unstaged: git checkout -- filename
- Staged: git reset HEAD filename
- Undo last commit (keep changes): git reset --soft HEAD~1
- Undo last commit (discard changes): git reset --hard HEAD~1

**5. Install Git on your server and configure it for remote access.**

**1. Install Git (Server)**

```
sudo apt install git -y  # Debian/Ubuntu
sudo yum install git -y  # RHEL/CentOS
```

**2. Create Git User & Repo**

```
sudo useradd -m -s /bin/bash git
sudo su - git
mkdir repos && cd repos
git init --bare myrepo.git
```

**3. Set Up SSH Access (Client → Server)**

```
ssh-keygen -t rsa -b 4096
ssh-copy-id git@server
```

**4. Clone & Push (Client)**

```
git clone git@server:/home/git/repos/myrepo.git
cd myrepo && touch README.md
git add . && git commit -m "Init" && git push origin main
```

**6. Generate an SSH key pair (ssh-keygen) and add the public key to your server for secure authentication**

Generate & Add SSH Key for Secure Authentication

**1. Generate SSH Key (Client Machine)**

ssh-keygen -t rsa -b 4096 -C "your_email@example.com"

- Saves key to ~/.ssh/id_rsa (private) & ~/.ssh/id_rsa.pub (public).

**2. Copy Public Key to Server**

ssh-copy-id user@server

**Or manually:**

scp ~/.ssh/id_rsa.pub user@server:~/

ssh user@server "mkdir -p ~/.ssh && cat ~/id_rsa.pub >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"

**3. Test SSH Login**

ssh user@server

7. **Create a Git repository on the server and configure permissions for users**

**1. Create Git User & Repo (Server)**

sudo useradd -m -s /bin/bash git

sudo su - git

mkdir repos && cd repos

git init --bare myrepo.git

**2. Add User SSH Access**

(Client Machine)

ssh-keygen -t rsa -b 4096

ssh-copy-id git@server

(Or manually add ~/.ssh/id_rsa.pub to /home/git/.ssh/authorized_keys on the server.)

**3. Set Repository Permissions**

sudo chown -R git:git /home/git/repos/myrepo.git

sudo chmod -R 755 /home/git/repos/myrepo.git  # Read-only for others

**4. Clone & Push (Client Users)**

git clone git@server:/home/git/repos/myrepo.git

cd myrepo && touch README.md

git add . && git commit -m "Init" && git push origin main

8. **Configure Git with Smart HTTP for easier repository access over HTTP.**

**Set Up Git with Smart HTTP**

**1. Install Apache & Git**

sudo apt install apache2 git -y  # Debian/Ubuntu

sudo yum install httpd git -y  # RHEL/CentOS

**2. Enable Git HTTP Backend**

sudo a2enmod cgi alias env && sudo systemctl restart apache2

**3. Create a Bare Git Repository**

sudo mkdir -p /var/www/git && cd /var/www/git

sudo git init --bare myrepo.git

sudo chown -R www-data:www-data myrepo.git  # (Use `apache` for RHEL)

**4. Configure Apache**

Edit /etc/apache2/sites-available/git.conf:

```
<VirtualHost *:80>
    DocumentRoot /var/www/git
    ScriptAlias /git/ /usr/lib/git-core/git-http-backend/
    SetEnv GIT_PROJECT_ROOT /var/www/git
    SetEnv GIT_HTTP_EXPORT_ALL
</VirtualHost>
```

**Enable & restart Apache:**

sudo a2ensite git && sudo systemctl restart apache2

**5. Clone & Push via HTTP**

git clone http://server/git/myrepo.git

cd myrepo && touch README.md

git add . && git commit -m "Init" && git push origin main

9. **Use git diff to view changes between commits, branches, or files. And Use git status to see changes in the working directory.**

**View Changes with git diff & git status**

**1. git diff (Compare Changes)**

- Unstaged changes:
- git diff
- Staged vs last commit:
- git diff --staged
- Between two commits:
- git diff commit1 commit2

- Between branches:
- git diff branch1 branch2
- For a specific file:
- git diff HEAD filename

**2. git status (Check Working Directory)**

- Show modified, staged, and untracked files:
- git status

**10. Use git stash to temporarily stash changes and git stash pop to apply them back.**

**Git Stash Commands**

- Stash changes: git stash
- View stash list: git stash list
- Apply & remove: git stash pop
- Apply without removing: git stash apply

**11. Use git checkout commit hash to go back to a specific commit and review its state**

**Checkout a Specific Commit**

- View commit history:
- git log --oneline
- Go to a specific commit:
- git checkout <commit-hash>
- Return to the latest commit:
- git checkout main