Git

Let’s learn Git through a practical project.

\*Project:\* Simple Website

\*Goal:\* Create a simple website using HTML, CSS, and Git.

\*Step-by-Step Guide:\*

\*Step 1: Initialize Git Repository\*

1. Create a new directory for your project: `mkdir simple-website`

2. Navigate into the directory: `cd simple-website`

3. Initialize a Git repository: `git init`

\*Git Command:\* `git init`

\*What it does:\* Creates a new Git repository in the current directory.

\*Step 2: Create Files\*

1. Create an HTML file: `touch index.html`

2. Create a CSS file: `touch styles.css`

3. Add content to `index.html` and `styles.css`

\*File Contents:\*

`index.html`:

```

<!DOCTYPE html>

<html>

<head>

<title>Simple Website</title>

<link rel=”stylesheet” href=”styles.css”>

</head>

<body>

<h1>Welcome to my website!</h1>

</body>

</html>

```

`styles.css`:

```

Body {

Background-color: #f2f2f2;

}

```

\*Step 3: Add Files to Git\*

1. Stage files: `git add .` (adds all files)

2. Commit changes: `git commit -m “Initial commit”`

\*Git Commands:\*

- `git add .` (stages all files)

- `git add <file>` (stages specific file)

- `git commit -m “<message>”` (commits changes with message)

\*What it does:\* Adds files to Git tracking and commits changes.

\*Step 4: Create Branch\*

1. Create a new branch: `git branch feature/header`

2. Switch to new branch: `git checkout feature/header`

\*Git Commands:\*

- `git branch <branch-name>` (creates new branch)

- `git checkout <branch-name>` (switches to branch)

\*What it does:\* Creates a new branch for feature development.

\*Step 5: Make Changes\*

1. Edit `index.html` and add a header:

```

<!DOCTYPE html>

<html>

<head>

<title>Simple Website</title>

<link rel=”stylesheet” href=”styles.css”>

</head>

<body>

<header>

<h1>Welcome to my website!</h1>

</header>

</body>

</html>

```

1. Commit changes: `git commit -am “Added header”`

\*Git Command:\* `git commit -am “<message>”` (commits changes with message and automatically stages)

\*What it does:\* Commits changes to the feature branch.

\*Step 6: Merge Branch\*

1. Switch to master branch: `git checkout master`

2. Merge feature branch: `git merge feature/header`

\*Git Commands:\*

- `git checkout master` (switches to master branch)

- `git merge <branch-name>` (merges branch into current branch)

\*What it does:\* Merges feature branch into master branch.

\*Step 7: Push to Remote Repository\*

1. Create a GitHub repository: (link unavailable)

2. Link local repository to remote: `git remote add origin <repository-url>`

3. Push changes: `git push -u origin master`

\*Git Commands:\*

- `git remote add origin <repository-url>` (links local repository to remote)

- `git push -u origin <branch-name>` (pushes changes and sets upstream tracking)

\*What it does:\* Pushes changes to remote repository.

\*Congratulations!\* You’ve completed the Git project.

Practice and explore more Git commands:

- `git log` (commit history)

- `git status` (repository status)

- `git diff` (file differences)

- `git reset` (reset changes)

\*Additional Resources:\*

- GitHub Guides: (link unavailable)

- Git Documentation: (link unavailable)

\*Next Steps:\*

1. Experiment with Git commands.

2. Create a new project using Git.

3. Explore Git tools and integrations.

How would you like to proceed?

1. Practice Git commands.

2. Create a new project.

3. Explore Git tools.