Chaicode

Welcome to Git and GitHub at ChaiCode Cohort

Introduction

Welcome to the ChaiCode development team! Git and GitHub are vital tools in our workflow, enabling efficient collaboration and seamless code management. This guide will help you get started and outline the best practices we follow at ChaiCode.

Basics of Git and GitHub

Git is a distributed version control system that allows developers to track changes, collaborate, and manage multiple project versions.

What is GitHub?

GitHub is a cloud-based platform for hosting Git repositories, providing tools for version control, pull requests, issue tracking, and more.

Installation and Setup

Windows: Download Git from git-scm.com, run the installer, and verify using git --version.
macOS: Use Homebrew: brew install git.
Linux: Use your package manager: sudo apt install git (Ubuntu).

Configuring Git

```
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com
```

Cloning the ChaiCode Repository

To clone a repository, run:

```
git clone https://github.com/ChaiCode/example-repo.git
```

Navigate to the repository folder:

```
cd example-repo
```

Basic Git Commands

git status - Check the status of your repository.
git add - Stage changes.
git commit -m "message" - Commit changes.
git push - Push changes to GitHub.
git pull - Pull updates from GitHub.
git log - View commit history.

Configuring Git

Use present tense (e.g., "Fix bug" not "Fixed bug").
Capitalize the first word.
Keep the message concise (50 characters or less).
Use prefixes like feat:, fix:, docs:.

Branching Workflow

Create and switch branches:

```
git branch feature/tea-menu
git checkout feature/tea-menu
```

Merge branches:

```
git checkout main
git merge feature/tea-menu
```

Pull Requests (PR)

To create a pull request:

1. Push your branch to GitHub: git push -u origin feature/tea-menu.
2. Go to the repository on GitHub and click New Pull Request.
3. Add a meaningful title and description.
4. Request a review from your teammates.

Best Practices

Commit small, frequent changes.
Write descriptive commit messages.
Pull updates regularly to avoid conflicts.
Review code before merging pull requests.