Antino GitHub Guideline

Git Do’s

- Create a Git repository for every new project.
- Always create a new branch for every new feature and bug.
- Regularly commit and push changes to the remote branch to avoid loss of work.
- Include a gitignore file in your project to avoid unwanted files being committed.
- Always commit changes with a concise and useful commit message.
- Keep your branch up to date with development branches.
- Always create a pull request for merging changes from one branch to another.
- Always create one pull request addressing one issue.
- Always review your code once by yourself before creating a pull request.
- Have more than one person review a pull request. It’s not necessary, but is a best practice.
- Enforce standards by using pull request templates and adding continuous integrations.
- Merge changes from the release branch to master after each release.
- Tag the master sources after every release.
- Delete branches if a feature or bug fix is merged to its intended branches and the branch is no longer required.
- Include read/write permission access control to repositories to prevent unauthorized access.
- Add protection for special branches like master and development to safeguard against accidental deletion.

Git Don’ts

- Don’t commit directly to the master or development branches.
- Don’t hold up work by not committing local branch changes to remote branches.
- Never commit application secrets in public repositories.
- Don’t commit large files in the repository. This will increase the size of the repository.
- Don’t create one pull request addressing multiple issues.
- Don’t work on multiple issues in the same branch. If a feature is dropped, it will be difficult to revert changes.
- Don’t reset a branch without committing/stashing your changes. If you do so, your chances will be lost.
- Don’t do a force push until you’re extremely comfortable performing this action.
- Don’t modify or delete public history.

Basic commands

Sr. No	Command	Details
1	git init <directory>	Create an empty Git repo in the specified directory. Run with no arguments to initialize the current directory as a git repository.
2	git clone <repo>	Clone repo located at <repo> into a local machine. Original repo can be located on the local filesystem or on a remote machine via HTTP or SSH
3	git add <directory>	Stage all changes in <directory> for the next commit. Replace <directory> with a <file> to change a specific file.
4	git commit -m "<message>"	Commit the staged snapshot, but instead of launching a text editor, use <message> as the commit message.
5	git status	List which files are staged, unstaged, and untracked.
6	git branch	List all of the branches in your repo. Add a <branch> argument to create a new branch with the name <branch>.
7	git checkout -b <branch>	Create and check out a new branch named <branch>. Drop the -b flag to checkout an existing branch.
8	git merge <branch>	Merge <branch> into the current branch.
9	git remote add <name> <url>	Create a new connection to a remote repo. After adding a remote, you can use <name> as a shortcut for <url> in other commands.
10	git fetch <remote> <branch>	Fetches a specific <branch>, from the repo. Leave off

		<branch> to fetch all remote refs.
11	git pull <remote>	Fetch the specified remote’s copy of the current branch and immediately merge it into the local copy.
12	git push <remote> <branch>	Push the branch to <remote>, along with necessary commits and objects. Creates a named branch in the remote repo if it doesn’t exist.
13	git config --global user.name <name>	Define the author name to be used for all commits by the current user.
14	git config --global user.email <email>	Define the author email to be used for all commits by the current user.

Branch naming conventions

Feature - Major features to be added.
Hotfix - Major bugs to be fixed and will be live soon.
Bugfix - Major and Minor bug fixes.
Release - Stable release version changes.

Syntax- <environment>/<developer-name>/<type>/<issue/feature>

Ex-1- dev/sy/feature/Auth/release1.1 - In dev environment, Sani Yadav was working on Auth feature for release 1.1.

Ex-2- dev/sy/feature/authentication - In dev environment, Sani Yadav was working on Authentication feature.

Ex-3- stage/sy/bugfix/Issue-while-login - In stage environment, Sani Yadav was working on bugfix while login

Ex-4- production/sy/hotfix/login-issues - In production environment, Sani Yadav was working on hotfix while login

Ex-5- dev/sy/feature/login-ui - In dev environment, Sani Yadav was working on Login UI.

Best practices

- Use a unique repository for each project.
- Always use an authorized email to commit the code.
- Always update your branches.
- Don’t add sensitive information to your code.
- Add your .gitignore file to the root folder.
- Delete all your inactive contributors.
- Don’t push .env or any config file.