

# Git Save Points Take Home Notes

*Intro to Vibe Coding / Class 7 / Keep with your Project Studio files*

Git gives your project a history. Today's goal is to understand how a saved file becomes part of a named checkpoint and how that checkpoint reaches a GitHub repository.

## Four places to recognize

**Working files.** The saved files in your project folder are the versions you can open and edit now.

**Staging area.** Git holds the file versions selected for the next commit. Staging leaves the original files in your folder.

**Local repository.** Git records commits on your laptop. A commit captures the staged project snapshot, including unchanged tracked files.

**GitHub repository.** A remote repository can hold the commits you push. A reviewer needs access to a private repository.

## The repeatable cycle

Save and test your change. Inspect what changed. Stage the intended files. Make a commit with a useful message. Push it. Finally, check the remote repository in your browser.

If you stage a file and then edit it again, your next commit uses the staged version. Stage the newer edit too when you want it included.

## Commands you may see

| Command                              | What it helps you see or do                     |
|--------------------------------------|-------------------------------------------------|
| <code>git status</code>              | Shows staged, unstaged, and untracked changes.  |
| <code>git diff</code>                | Shows tracked changes you have not staged yet.  |
| <code>git diff --staged</code>       | Shows changes selected for the next commit.     |
| <code>git add &lt;file&gt;</code>    | Stages the named file's current saved version.  |
| <code>git commit -m "message"</code> | Records the staged snapshot locally.            |
| <code>git push</code>                | Sends local commits to the configured upstream. |
| <code>git log --oneline -5</code>    | Shows the five latest local commits.            |

Replace <file> with an actual filename. The first push may also set an upstream. Let your agent inspect the current branch instead of assuming its name.

## What belongs in the repository

Keep the source code, migrations, package.json, package-lock.json, README, and project configuration that contains no secrets. The lockfile helps reinstall the same dependency versions.

Use .gitignore for node\_modules/, .wrangler/, build output, .env files, and .dev.vars files. Adding a name to .gitignore does not remove a file that Git already tracks. Ask Drew if sensitive files are already tracked.

Local D1 records live under .wrangler/state in this starter. They stay separate from your source history. A migration describes a database structure change; it does not automatically include the records you entered.

## Three different kinds of saving

Save in your editor: updates the file on disk. Commit with Git: records a local checkpoint. Push to GitHub: sends commits to a remote repository. Publishing a working website requires a separate hosting or deployment step.

## When something gets stuck

**The agent cannot find my files.** Open the actual Project Studio folder in the local coding workspace. Ask it to read package.json and show the path. A cloud chat cannot inspect your laptop's folder without an appropriate connection.

**A command is blocked.** Show Drew the command and exact error. Use the tool's normal approval flow when appropriate. Keep sandbox controls in place; do not erase the project or its data to solve a permission issue.

**Git asks who I am.** Use your chosen commit display name and GitHub-provided no-reply email. Set these for this repository. Git author settings and GitHub sign-in are separate.

**The push fails.** Check your account, remote URL, branch, and error with Drew. Avoid force pushes or deleting history. If you still need an account, you can continue making local commits.

**Drew cannot open my repository.** Check the URL and whether the repository is private. Invite the instructor account he gives you and have him verify access.

## A useful checkpoint message

"Add idea search" or "Improve empty project message" tells future you what changed. Pick a message that matches the actual diff. A commit message does not change when you edit the next-message box later.

## Keep exploring

Try one improvement, explain the diff, and repeat the cycle. Update ChatGPT or Claude weekly as usual. Capstone development homework begins after winter break in January 2027.

Lesson, prompts, and simulator: <https://bachkidscode.app/lessons/git-save-points/>

References: [git-scm.com/docs/git-add](https://git-scm.com/docs/git-add), [git-scm.com/docs/git-commit](https://git-scm.com/docs/git-commit), [git-scm.com/docs/git-push](https://git-scm.com/docs/git-push), [git-scm.com/docs/gitignore](https://git-scm.com/docs/gitignore); [docs.github.com](https://docs.github.com) (commit email address); [cli.github.com/manual/gh\\_repo\\_create](https://cli.github.com/manual/gh_repo_create).