initialize git → git init ← initialize project folder!

git diff .. (for lots of changes with no color) → use git pager → space to navigate pages
f to go forward
b to go backwards
-S to toggle between wrapped + not wrapped text.
q to quit pagemaster

git diff --color-words
↳ helps you zero in on small changes on long lines.

STAGE + COMMIT SHORTCUT
git commit -am "edit suggestion phrase #" ← custom message

git log -n2 to show most recent commit.

COMPARE COMMITS
git diff older commit SHA# .. newer commit SHA#

VIEW A COMMIT
git show SHA# --color-words ← only 6-8 characters needed
↳ will take you to pagemaster

Multiline commit messages
- All you have to do is just commit without the -m tag.
- It will take you to your default text editor and your first line should summarize the commit while the lines underneath it can add specifics

Making atomic commits
- Not making changes in the editor (atom). It's about making them small and compartmentalized
- Have them affect only a single aspect of your project
-

Challenge : Client Edits
- The client requests some prices to change on the website and a few style changes in writing.
- If these prices and style changes exist in different places, you can compartmentalize the commits so that you can categorize each of these distinct change requests from the client

Soltn: Client edits
- Say you make multiple distinct changes in a single file but ideally all of atomic commits about the changes to the file are separated... how do you handle this?
    - Beginner: just do each of the tasks/commits separately
    - "Some advanced techniques" solution?