

# Git Lab – Get familiar with basic git commands

**Step 1 Checkout your git repo we created earlier when we built the Jenkins Pipeline job with Jenkinsfile.**

```
bash-3.2$ git clone https://github.com/kevinli-webbertech/gs-spring-boot.git
```

**Step 2 Check your branches in your local. Because you clone the image site of the whole repo from the git server to your local.**

```
bash-3.2$ git branch  
• main
```

## Step 3 you create a local branch, or your development branch

```
bash-3.2$ git checkout -b my_branch
Switched to a new branch 'my_branch'
bash-3.2$ git branch
  main
* my_branch
```

## Step 4 Let us create a new file called Readme.md

```
bash-3.2$ git status
On branch my_branch
nothing to commit, working tree clean

bash-3.2$ touch Readme.md
```

## Step 5 Edit the Readme.md

Assume this is a documentation or source code we need to commit

```
bash-3.2$ git status
On branch my_branch
Untracked files:
  (use "git add <file>..." to include in what will be committed)
```

*Readme.md*

```
nothing added to commit but untracked files present (use "git add"
to track)
```

```
bash-3.2$ vi Readme.md
bash-3.2$ git status
On branch my_branch
Untracked files:
  (use "git add <file>..." to include in what will be committed)
```

*Readme.md*

*nothing added to commit but untracked files present (use "git add" to track)*

```
bash-3.2$ git add Readme.md
bash-3.2$ git status
On branch my_branch
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
```

*new file: Readme.md*

```
bash-3.2$ git commit -m "add a readme file"
[my_branch 864e332] add a readme file
1 file changed, 9 insertions(+)
create mode 100644 Readme.md
```

Now `git status` does not tell you anything beyond this point when you committed all the new files successfully but it will tell anything that is not committed, so called "not tracked".

```
bash-3.2$ git status
On branch my_branch
nothing to commit, working tree clean
```

# Step 6 Git PUSH

Now I try to push the new branch with my new changes to the remote server.

If I do that, I see the following,

```
bash-3.2$ git push  
fatal: The current branch my_branch has no upstream branch.  
To push the current branch and set the remote as upstream, use
```

```
git push --set-upstream origin my_branch
```

Copy the following command from the above message prompt and then execute it, then you will see the new branch created in your local is pushed to the remote server, now the server has it.

```
bash-3.2$ git push --set-upstream origin my_branch
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 366 bytes | 366.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'my_branch' on GitHub by visiting:
remote:      https://github.com/kevinli-webbertech/gs-spring-
boot/pull/new/my_branch
remote:
To https://github.com/kevinli-webbertech/gs-spring-boot.git
 * [new branch]      my_branch -> my_branch
Branch 'my_branch' set up to track remote branch 'my_branch' from
'origin'.
```

## Step 7. Check the new branch in both remote git server and in local.

There are two ways you can check if your branch was successfully pushed to the remote git server.

- You can go to the web page of git and check the new branch there.
- You run a command in your local command line and you do not have to waste to open a browser tab and check the info by clicking a few times.
- Now if I type the following commands,

```
bash-3.2$ git branch
main
• my_branch
```

It just showed my local branches, not the remote branches,

If somehow I removed some local branches, that will not affect the git server repo branches,

unless I delete the remote branches as well.

So now let us delete the local branch after we pushed it.

```
bash-3.2$ git branch
main
• my_branch
```

```
bash-3.2$ git branch -d my_branch
error: Cannot delete branch 'my_branch' checked out at
'/Users/xiaofengli/code/gs-spring-boot'
```

As we notice that above, I am currently on the `my\_branch`, so I can not delete it. But if I really want to delete it, I have to switch to another branch.

```
bash-3.2$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Now we are able to delete the previous branch.

```
bash-3.2$ git branch -d my_branch
warning: deleting branch 'my_branch' that has been merged to
'refs/remotes/origin/my_branch', but not yet merged to
HEAD.
Deleted branch my_branch (was 864e332).
```

Now we check the local branch, and it is not there anymore.

```
bash-3.2$ git branch
• main
```

- Solution 2

How do we check what remote branches we have in the remote git server repo using command?

```
bash-3.2$ git branch -r
origin/HEAD -> origin/main
```

*origin/main*  
*origin/my\_branch*

Now what we can see from the above is that we can see ``origin/my_branch`` in there. That is our remote branch that we just pushed.

Now since we do not have a local branch in our own computer, sometimes at work, I mistakenly got rid of my own personal branch and I really want to get it back. We can checkout remote branch to local as well.

For instance, now I want to checkout my\_branch back to my local from remote server.

```
bash-3.2$ git checkout my_branch  
Branch 'my_branch' set up to track remote branch 'my_branch' from  
'origin'.  
Switched to a new branch 'my_branch'
```

Now I check my local branches and I can see my branch came back.

```
bash-3.2$ git branch  
main  
• my_branch
```

```

bash-3.2$ git branch -h
usage: git branch [<options>] [-r | -a] [--merged | --no-merged]
       or: git branch [<options>] [-l] [-f] <branch-name> [<start-
point>]
       or: git branch [<options>] [-r] (-d | -D) <branch-name>...
       or: git branch [<options>] (-m | -M) [<old-branch>] <new-branch>
       or: git branch [<options>] (-c | -C) [<old-branch>] <new-branch>
       or: git branch [<options>] [-r | -a] [--points-at]
       or: git branch [<options>] [-r | -a] [--format]

```

#### Generic options

```

-v, --verbose          show hash and subject, give twice for
upstream branch
-q, --quiet            suppress informational messages
-t, --track            set up tracking mode (see git-pull(1))
-u, --set-upstream-to <upstream>
                        change the upstream info
--unset-upstream       Unset the upstream info
--color[=<when>]       use colored output
-r, --remotes          act on remote-tracking branches
--contains <commit>    print only branches that contain the
commit
--no-contains <commit>
                        print only branches that don't contain the
commit
--abbrev[=<n>]         use <n> digits to display SHA-1s

```

#### Specific git-branch actions:

```

-a, --all              list both remote-tracking and local
branches
-d, --delete           delete fully merged branch

-D                     delete branch (even if not merged)
-m, --move             move/rename a branch and its reflog
-M                     move/rename a branch, even if target
exists

-c, --copy             copy a branch and its reflog
-C                     copy a branch, even if target exists

-l, --list             list branch names
--create-reflog        create the branch's reflog

--edit-description     edit the description for the branch

```



<code>-f, --force</code>	<i>force creation, move/rename, deletion</i>
<code>--merged &lt;commit&gt;</code>	<i>print only branches that are merged</i>
<code>--no-merged &lt;commit&gt;</code>	<i>print only branches that are not merged</i>
<code>--column[=&lt;style&gt;]</code>	<i>list branches in columns</i>
<code>--sort &lt;key&gt;</code>	<i>field name to sort on</i>
<code>--points-at &lt;object&gt;</code>	<i>print only branches of the object</i>
<code>-i, --ignore-case</code>	<i>sorting and filtering are case insensitive</i>
<code>--format &lt;format&gt;</code>	<i>format to use for the output</i>

In linux/Unix, the -h will help you, and single dash with one letter is similar to double dash with a full words.

For example, `-r`` means `--remotes``

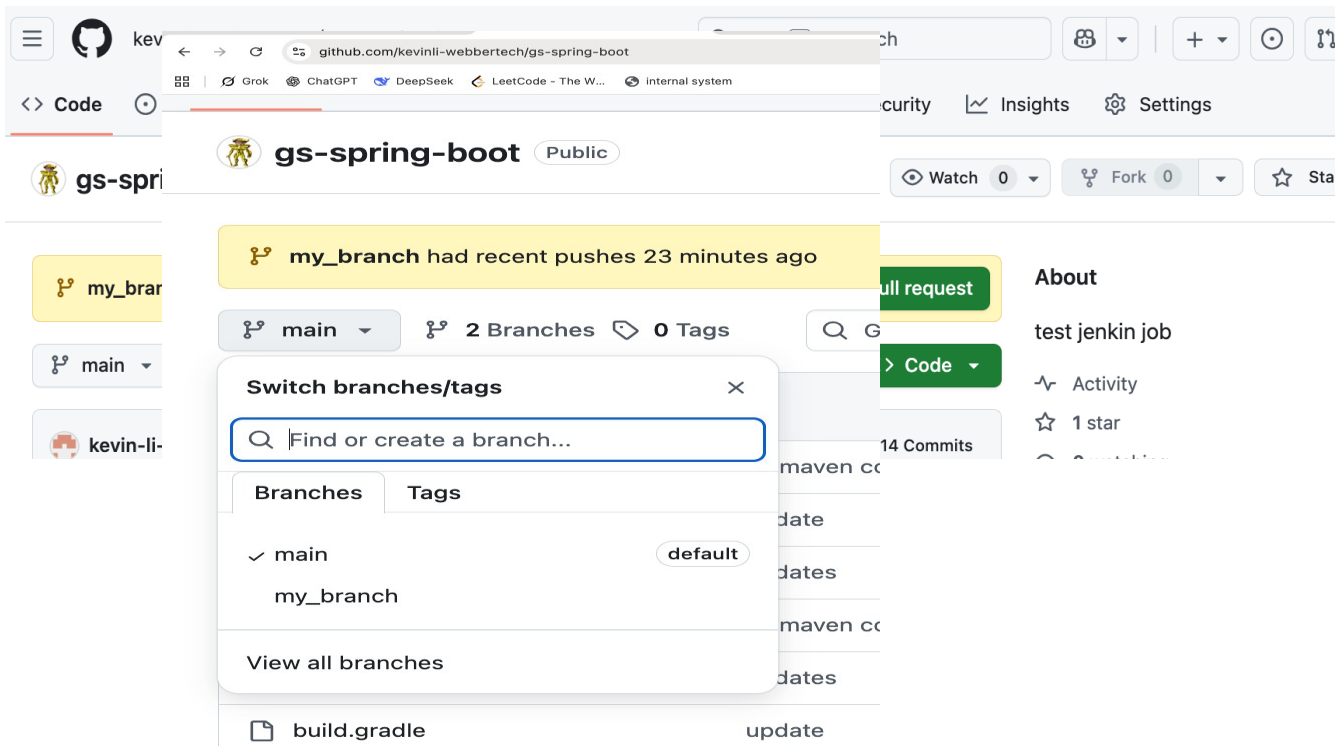
```
bash-3.2$ git branch --remote
origin/HEAD -> origin/main
origin/main
origin/my_branch
```

```
bash-3.2$ git branch --remotes
origin/HEAD -> origin/main
origin/main
origin/my_branch
```

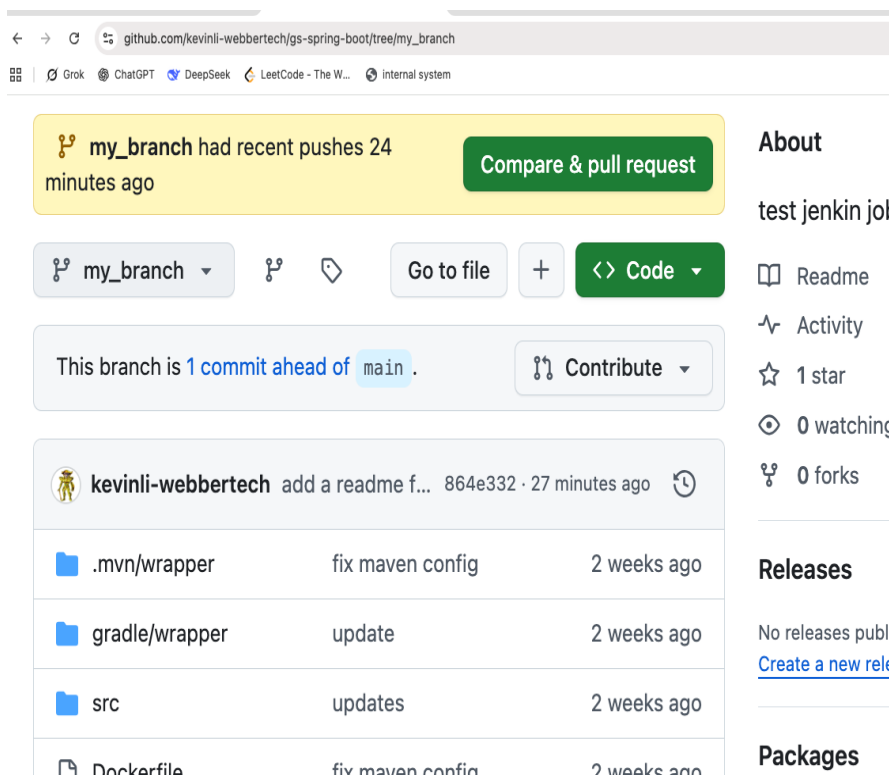
- Solution 2

Let us check the branch on the web UI.











Please see the yellow banner below,



Check the drop-down menu,



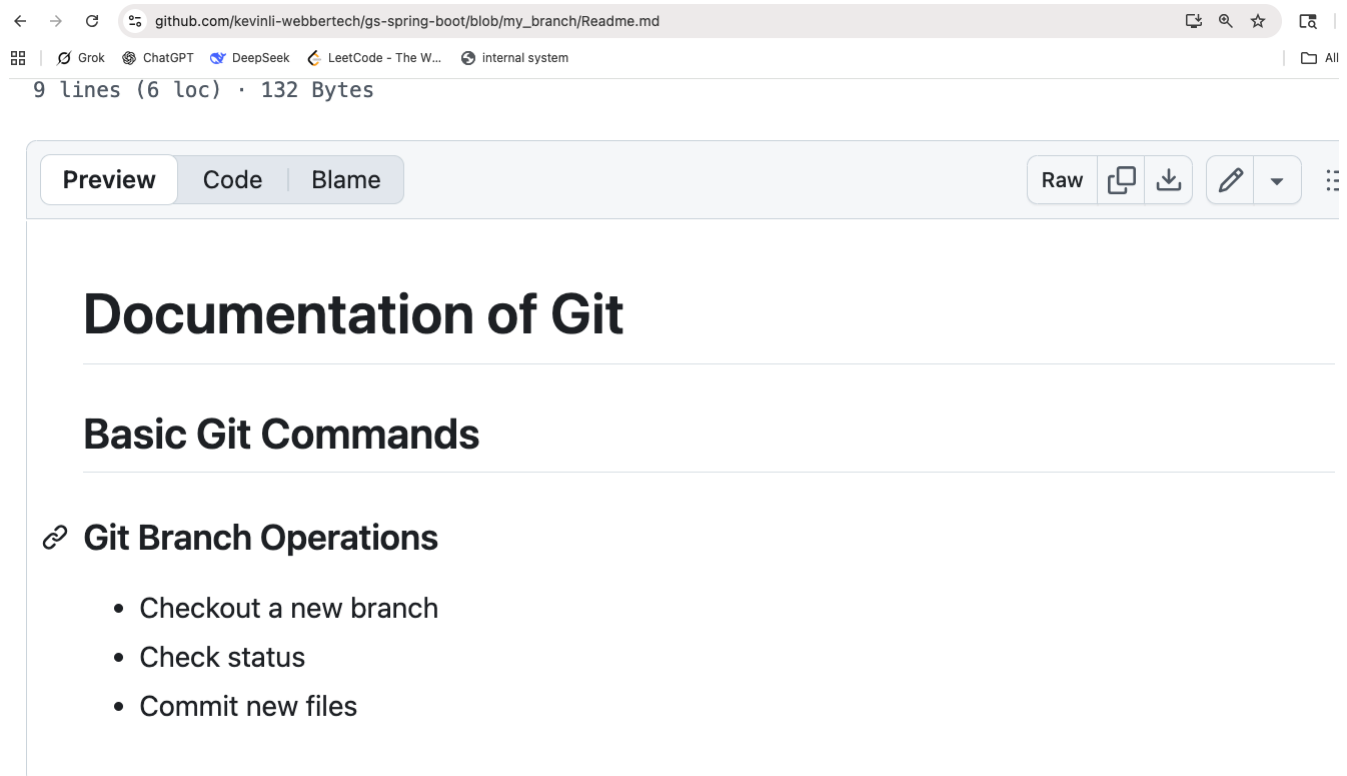
Now let us check the new file we added,

github.com/kevinli-webbertech/gs-spring-boot/tree/my_branch			
	gradle/wrapper	update	2 weeks ago
	src	updates	2 weeks ago
	Dockerfile	fix maven config	2 weeks ago
	Jenkinsfile	updates	2 weeks ago
	<a href="#">Readme.md</a>	add a readme file	27 minutes ago
Readme.md			
	build.gradle	update	2 weeks ago
	gradlew	update	2 weeks ago
	gradlew.bat	update	2 weeks ago
	mvnw	update	2 weeks ago
	.	.	.

Click on the Readme.md, which is a markdown file.

**Markdown** is actually similar to **Markup** Language which is a HTML.

I use markdown in css in my course site so it will convert into html without writing a whole bunch of code and writing markdown is just like a word document but we have a lot of annotations to render and support the writing without html tagging libs.



The screenshot shows a web browser displaying a GitHub README file. The address bar shows the URL: `github.com/kevinli-webbertech/gs-spring-boot/blob/my_branch/Readme.md`. Below the address bar, there are tabs for Grok, ChatGPT, DeepSeek, LeetCode - The W..., and internal system. The file statistics show 9 lines (6 loc) and 132 Bytes. The file is in preview mode, with tabs for Preview, Code, and Blame. The content of the README is as follows:

# Documentation of Git

---

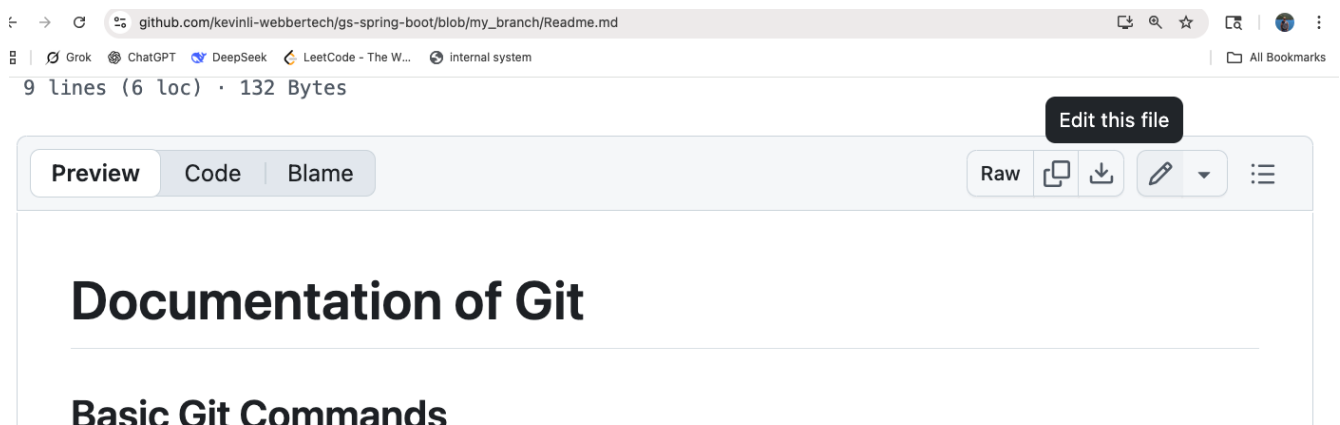
## Basic Git Commands

---

### [Git Branch Operations](#)

- Checkout a new branch
- Check status
- Commit new files

Click on “edit” button,



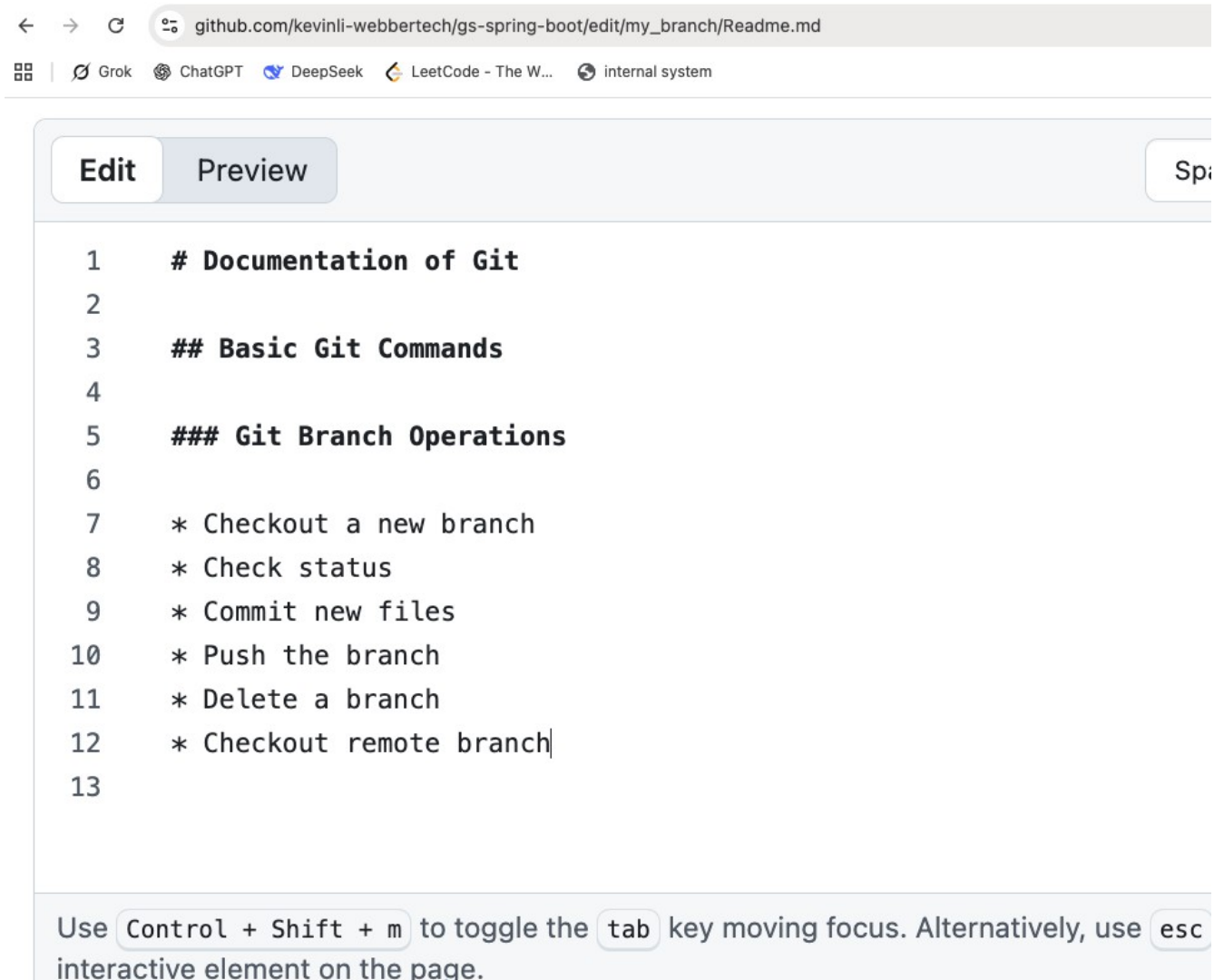
The screenshot shows the same GitHub README file, but with the 'Edit this file' button highlighted in a dark box. The file statistics and content are the same as in the previous screenshot.

# Documentation of Git

---

## Basic Git Commands

Keep Editing and adding new stuff,



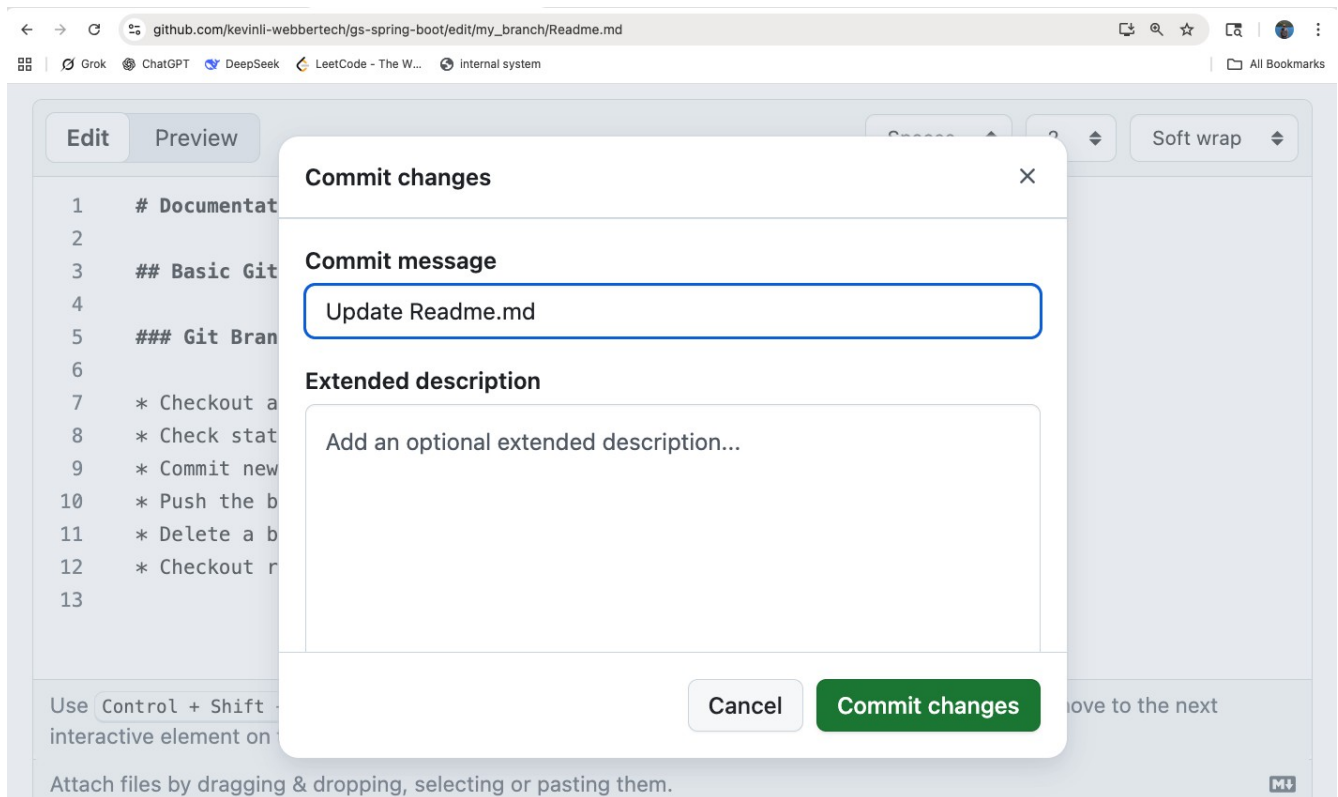
The screenshot shows a web browser window with the address bar displaying `github.com/kevinli-webbertech/gs-spring-boot/edit/my_branch/Readme.md`. Below the address bar, there are several tabs: Grok, ChatGPT, DeepSeek, LeetCode - The W..., and internal system. The main content area has two tabs: "Edit" (active) and "Preview". The "Edit" tab shows a text editor with the following content:

```
1  # Documentation of Git
2
3  ## Basic Git Commands
4
5  ### Git Branch Operations
6
7  * Checkout a new branch
8  * Check status
9  * Commit new files
10 * Push the branch
11 * Delete a branch
12 * Checkout remote branch|
13
```

At the bottom of the editor, there is a footer message: "Use `Control + Shift + m` to toggle the `tab` key moving focus. Alternatively, use `esc` interactive element on the page."

Once you are done, do a ctrl/cmd + s to save the file.

And you will see this,



Hit the green button above.

After you save it, please check the following,


The screenshot shows a GitHub repository page for 'gs-spring-boot' on the 'my\_branch' branch. The page title is 'gs-spring-boot / Readme.md'. Below the title, there are tabs for 'Preview', 'Code', and 'Blame'. The 'Preview' tab is selected, showing the content of the README file. The content includes a section titled 'Basic Git Commands' and a section titled 'Git Branch Operations' with a list of commands: 'Checkout a new branch', 'Check status', 'Commit new files', 'Push the branch', 'Delete a branch', and 'Checkout remote branch'. The 'Push the branch' and 'Delete a branch' items are highlighted in blue.


Next, let us check all the commits we did,

The screenshot shows a GitHub repository page for 'gs-spring-boot' on the 'my\_branch' branch. The page title is 'gs-spring-boot / tree / my\_branch'. Below the title, there are tabs for 'my\_branch', '2 Branches', and '0 Tags'. The 'my\_branch' tab is selected, showing the commit history. The commit history shows a list of commits, with the most recent commit being 'Update Readme.md' by 'kevinli-webbertech' 14f105e · now, which has 16 commits. The commit message 'Update Readme.md' is highlighted in blue. The commit history table has columns for the commit message, the commit hash, and the time ago. The table shows two commits: 'fix maven config' and 'update', both from 2 weeks ago.

Commit Message	Commit Hash	Time Ago
fix maven config	14f105e	2 weeks ago
update	14f105e	2 weeks ago

In the above image, let us click on “16 commits”,

 **kevinli-webbertech** Update Readme.md 14f105e · 1 minute ago 16 Commits

 .mvn/wrapper fix maven config 2 weeks ago

Now we can see the following hashes,

kevinli-webbertech / gs-spring-boot

Type / to search

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

### Commits

my\_branch All users All time

Commits on Oct 28, 2025

Update Readme.md

kevinli-webbertech authored 2 minutes ago

Verified 14f105e

add a readme file

kevinli-webbertech committed 33 minutes ago

864e332

Commits on Oct 16, 2025

updates

kevin-li-klw committed 2 weeks ago

6698552

add maven

kevin-li-klw committed 2 weeks ago

448f071



For instance, click on the hash below,

The screenshot shows the GitHub interface for the repository 'kevinli-webbertech / gs-spring-boot'. The 'Commits' tab is selected, and the branch 'my\_branch' is chosen. The commit history is filtered for 'All users' and 'All time'. A section titled 'Commits on Oct 28, 2025' contains two commits:

- Update Readme.md** by kevinli-webbertech, authored 2 minutes ago. The commit hash is 14f105e, marked as 'Verified'. A 'View commit details' button is visible.
- add a readme file** by kevinli-webbertech, committed 33 minutes ago. The commit hash is 864e332.



Then you click on the hash,

This screenshot is identical to the previous one, showing the same GitHub commit page. The 'Commits' header is highlighted with a blue rectangular box. The commit details for 'Update Readme.md' (hash 14f105e) and 'add a readme file' (hash 864e332) are visible, with the first commit marked as 'Verified'.


In the next view, you can see a diff of previous edits vs the latest edits.

kevinli-webbertech authored 3 minutes ago Verified


Update Readme.md





 1 parent [864e332](#) commit 14f105e 

Filter files...


 Readme.md


1 file changed +3 -0 lines changed

Search within code 

Readme.md   +3  <>  ...

@@ -7,3 +7,6 @@	
7	* Checkout a new branch
8	* Check status
9	* Commit new files
10	+ * Push the branch
11	+ * Delete a branch
12	+ * Checkout remote branch

Comments 0  Lock conversation

 Comment

