Module three

Project Version Control with Git

Implementing a feature request

A feature request is a requirement to modify or add functionality to an application or website. Suppose that you have received a requirement to add the letter D to the Phonetic Website. To implement it you need to modify the three HTML files located in the working directory of the project.

Adding "Delta", "Detroit" and "David" to pilots.html, cities.html and names.html.

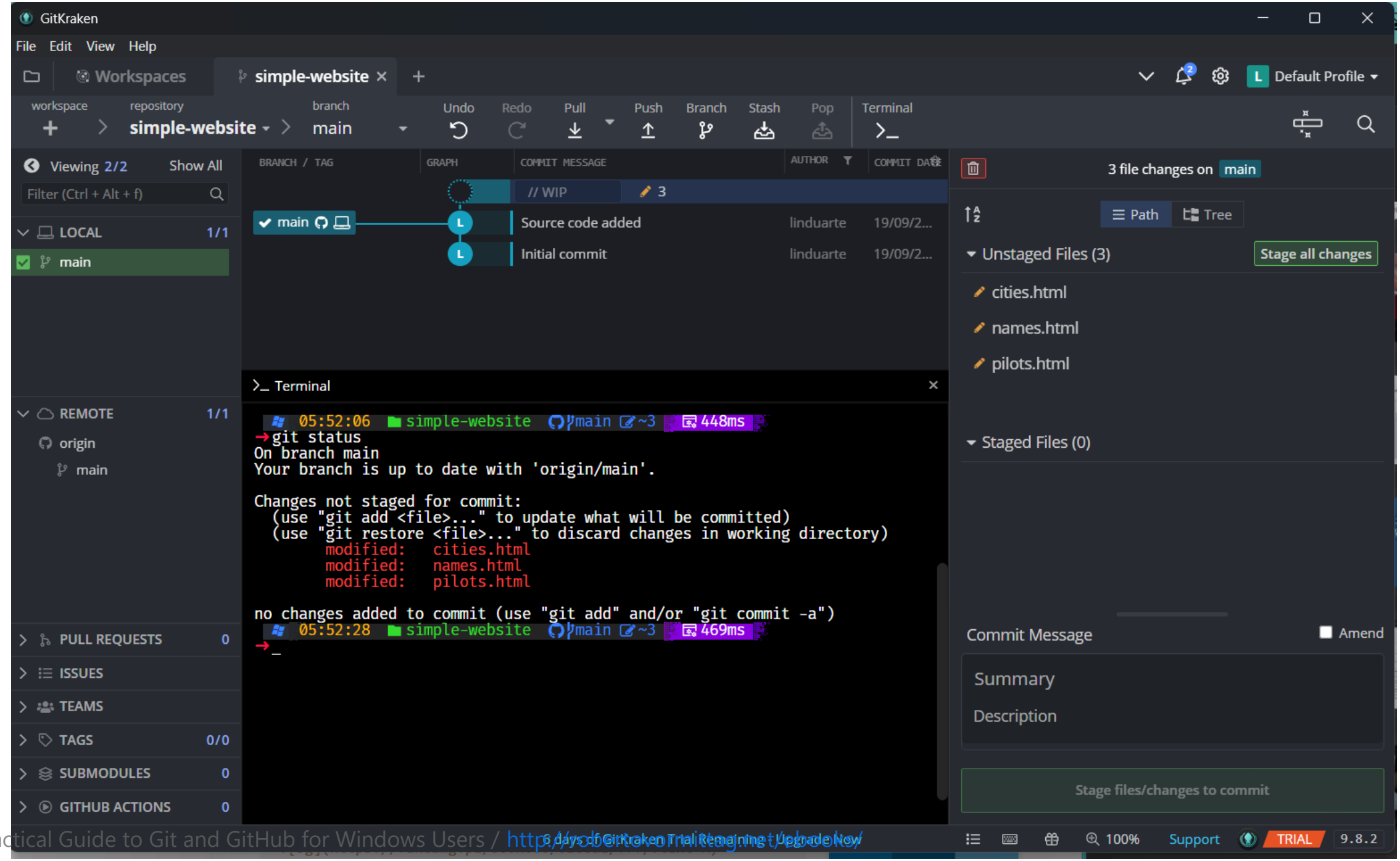
Use an editor to make the changes.

```
<h1>Pilot's Alphabet</h1>
<ul id="pilot">
<!-- ALPHABET START -->
  <li>Alpha</li>
  <li>Bravo</li>
  <li>Charlie</li>
  <li>Delta</li>
<!-- ALPHABET END -->
```

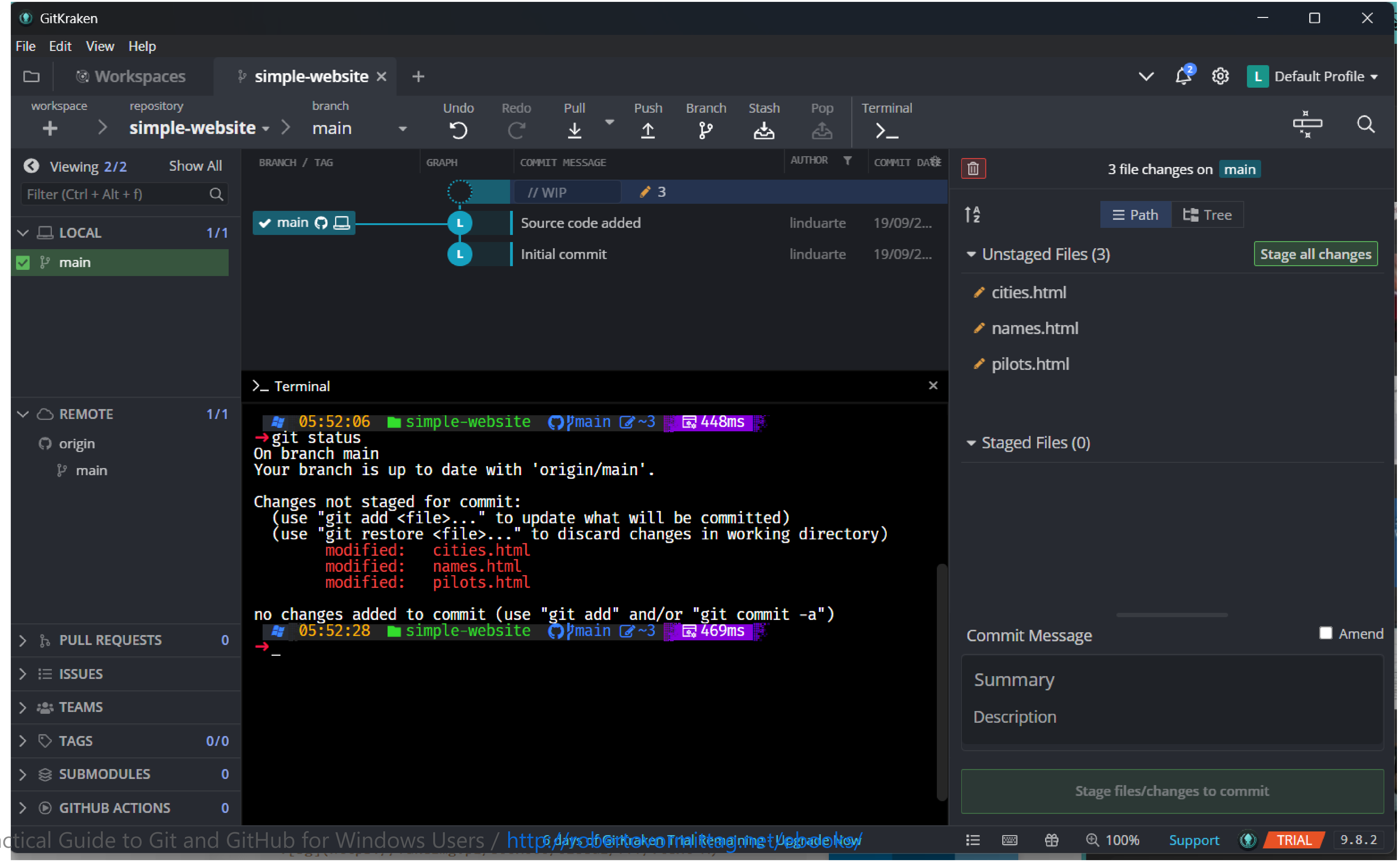
```
<h1>Cities' Alphabet</h1>
<ul id="city">
<!-- ALPHABET START -->
  <li>Atlanta</li>
  <li>Boston</li>
  <li>Chicago</li>
  <li>Detroit</li>
<!-- ALPHABET END -->
```

```
<h1>Names' Alphabet</h1>
<ul id="name">
<!-- ALPHABET START -->
  <li>Andrew</li>
  <li>Brigitte</li>
  <li>Charles</li>
  <li>David</li>
<!-- ALPHABET END -->
```

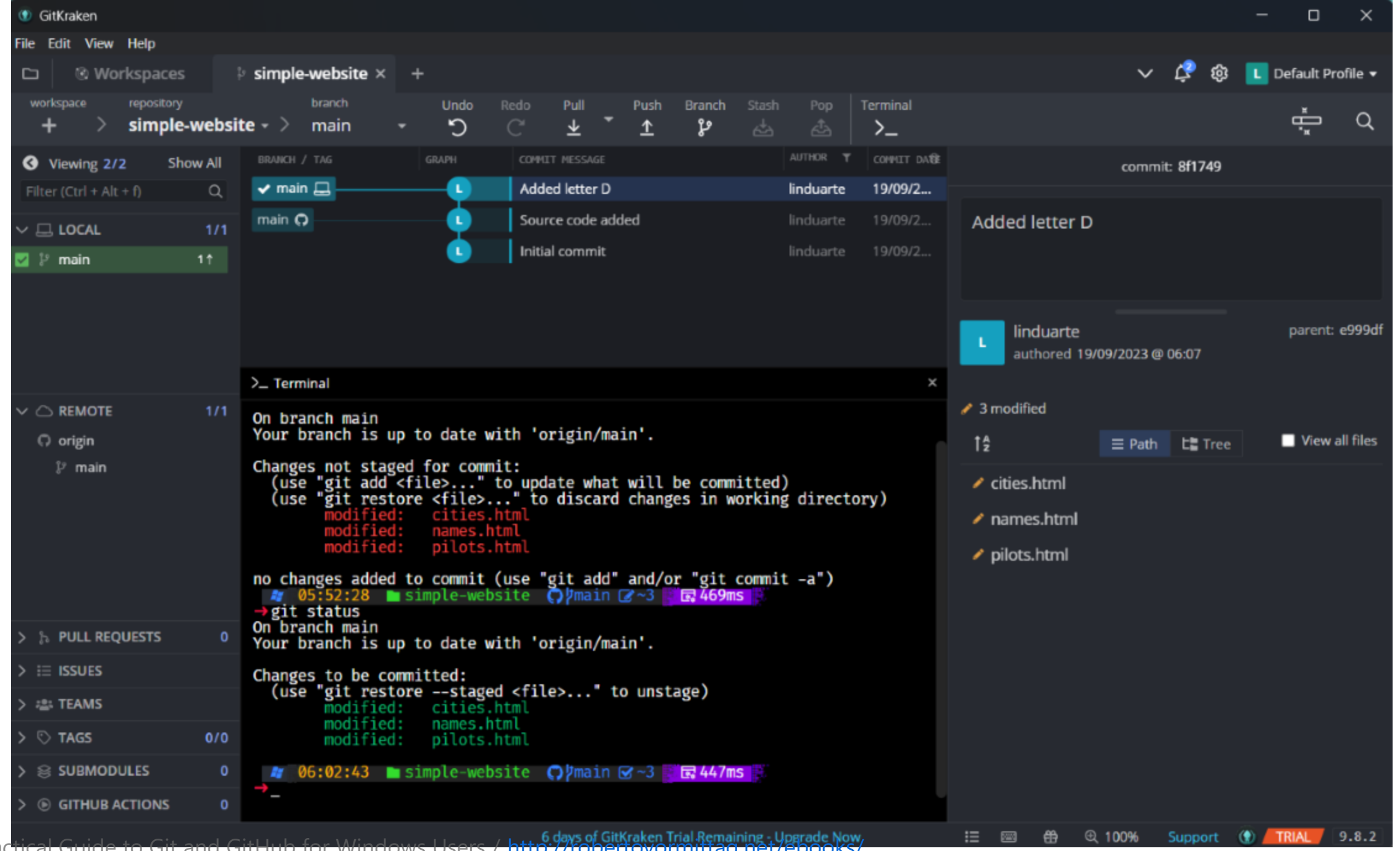
Let´s take a picture using GitKraken to show us the process.


GitKraken
File Edit View Help
Workspaces
simple-website
Default Profile
workspace
repository
branch
Undo
Redo
Pull
Push
Branch
Stash
Pop
Terminal
simple-website
main
Viewing 2/2
Show All
Filter (Ctrl + Alt + f)
LOCAL 1/1
main
REMOTE 1/1
origin
main
PULL REQUESTS 0
ISSUES
TEAMS
TAGS 0/0
SUBMODULES 0
GITHUB ACTIONS 0
BRANCH / TAG
GRAPH
COMMIT MESSAGE
AUTHOR
COMMIT DATE
// WIP 3
main
Source code added linduarte 19/09/2...
Initial commit linduarte 19/09/2...
Terminal
05:52:06 simple-website main ~3 448ms
git status
On branch main
Your branch is up to date with 'origin/main'.
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: cities.html
modified: names.html
modified: pilots.html
no changes added to commit (use "git add" and/or "git commit -a")
05:52:28 simple-website main ~3 469ms
3 file changes on main
Path
Tree
Unstaged Files (3)
Stage all changes
cities.html
names.html
pilots.html
Staged Files (0)
Commit Message
Amend
Summary
Description
Stage files/changes to commit
100%
Support
TRIAL
9.8.2

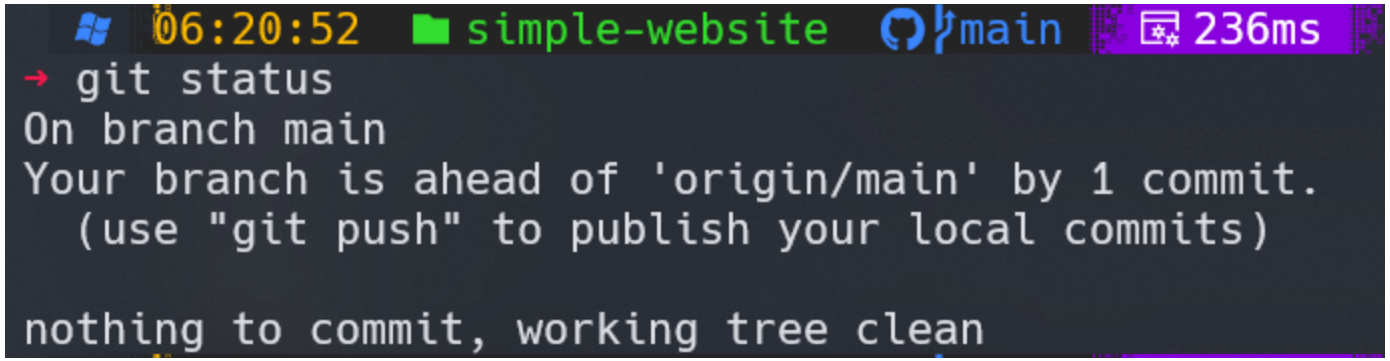
After adding all html files


GitKraken
File Edit View Help
Workspaces
simple-website
Default Profile
workspace
repository
simple-website
branch
main
Undo
Redo
Pull
Push
Branch
Stash
Pop
Terminal
Viewing 2/2
Show All
Filter (Ctrl + Alt + f)
LOCAL 1/1
main
REMOTE 1/1
origin
main
PULL REQUESTS 0
ISSUES
TEAMS
TAGS 0/0
SUBMODULES 0
GITHUB ACTIONS 0
BRANCH / TAG
GRAPH
COMMIT MESSAGE
AUTHOR
COMMIT DATE
// WIP
3
main
Source code added
linduarte
19/09/2...
Initial commit
linduarte
19/09/2...
Terminal
05:52:06 simple-website main ~3 448ms
git status
On branch main
Your branch is up to date with 'origin/main'.
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: cities.html
modified: names.html
modified: pilots.html
no changes added to commit (use "git add" and/or "git commit -a")
05:52:28 simple-website main ~3 469ms
3 file changes on main
Path
Tree
Unstaged Files (3)
Stage all changes
cities.html
names.html
pilots.html
Staged Files (0)
Commit Message
Amend
Summary
Description
Stage files/changes to commit
6 days of GitKraken Trial Remaining - Upgrade Now
100%
Support
TRIAL
9.8.2

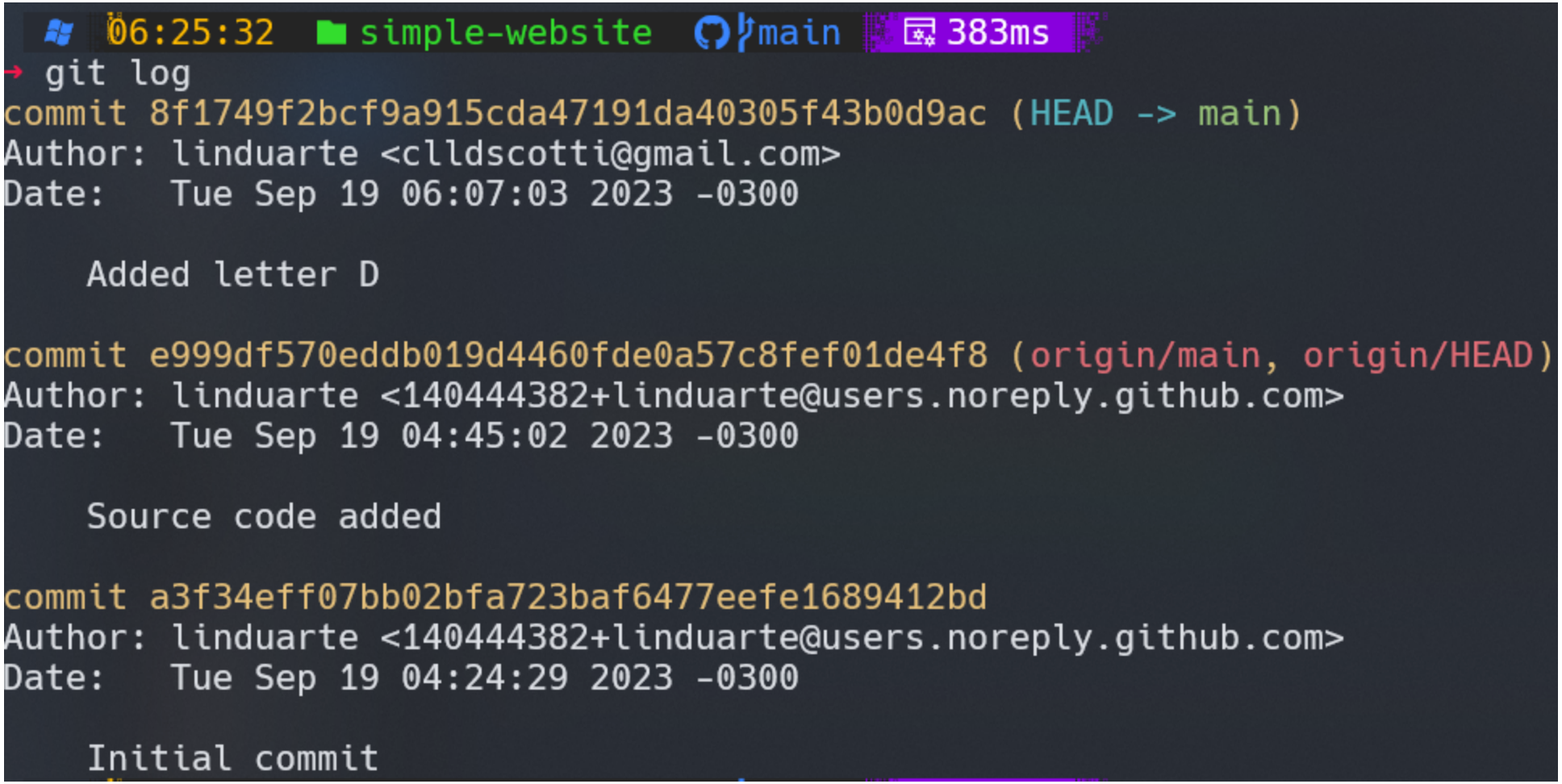
Let´s see the committing of the html files in GitKraken


GitKraken
File Edit View Help
Workspaces
simple-website
Default Profile
workspace
repository
branch
simple-website
main
Undo
Redo
Pull
Push
Branch
Stash
Pop
Terminal
Viewing 2/2
Show All
Filter (Ctrl + Alt + f)
LOCAL 1/1
main 1↑
REMOTE 1/1
origin
main
PULL REQUESTS 0
ISSUES
TEAMS
TAGS 0/0
SUBMODULES 0
GITHUB ACTIONS 0
BRANCH / TAG
GRAPH
COMMIT MESSAGE
AUTHOR
COMMIT DATE
main
main
Added letter D linduarte 19/09/2...
Source code added linduarte 19/09/2...
Initial commit linduarte 19/09/2...
commit: 8f1749
Added letter D
linduarte
authored 19/09/2023 @ 06:07
parent: e999df
3 modified
Path
Tree
View all files
cities.html
names.html
pilots.html
Terminal
On branch main
Your branch is up to date with 'origin/main'.
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: cities.html
modified: names.html
modified: pilots.html
no changes added to commit (use "git add" and/or "git commit -a")
05:52:28 simple-website main ~3 469ms
git status
On branch main
Your branch is up to date with 'origin/main'.
Changes to be committed:
(use "git restore --staged <file>..." to unstage)
modified: cities.html
modified: names.html
modified: pilots.html
06:02:43 simple-website main ~3 447ms
6 days of GitKraken Trial Remaining - Upgrade Now
100%
Support
TRIAL
9.8.2

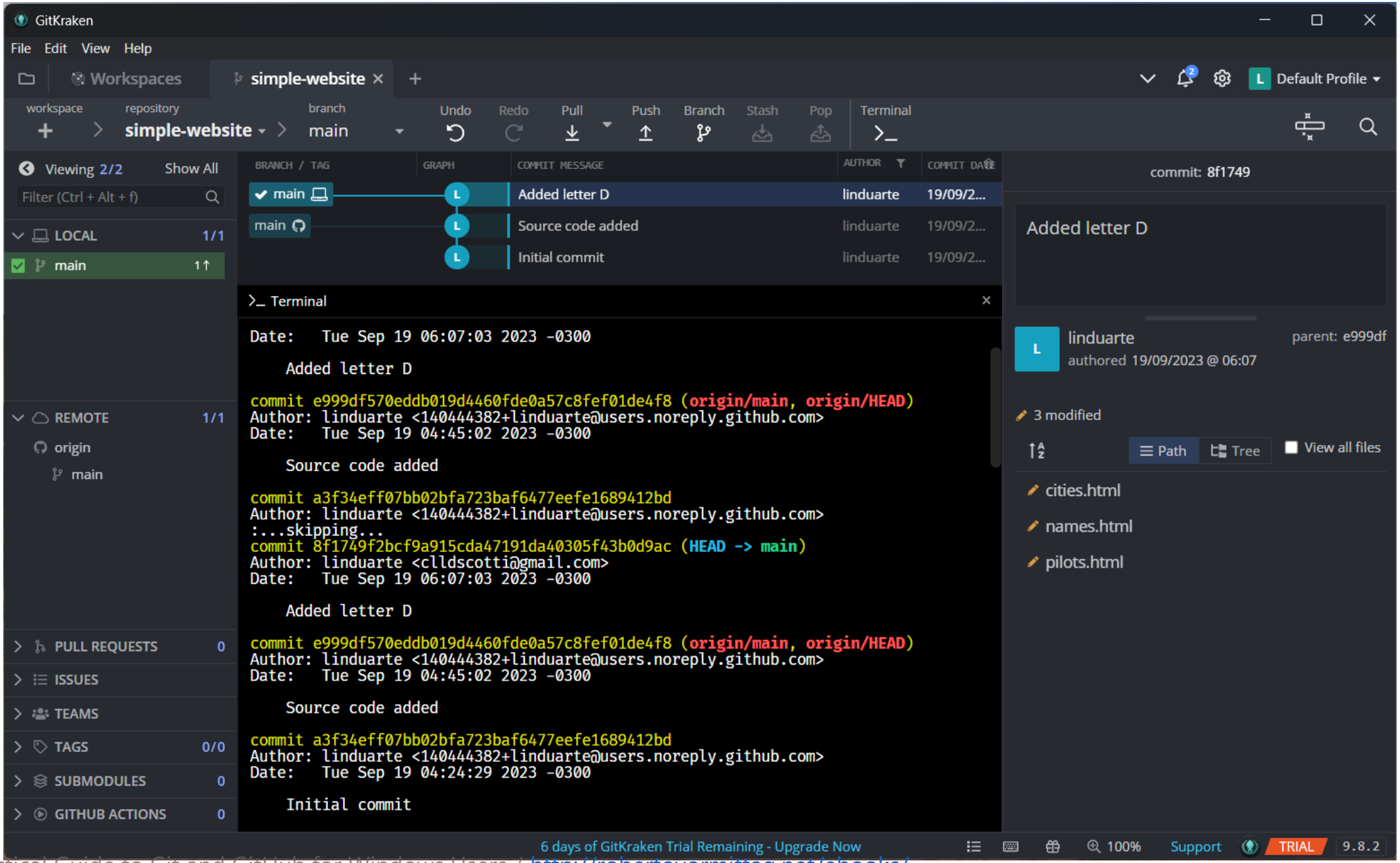
Viewing from Git bash the final status



Viewing Project History

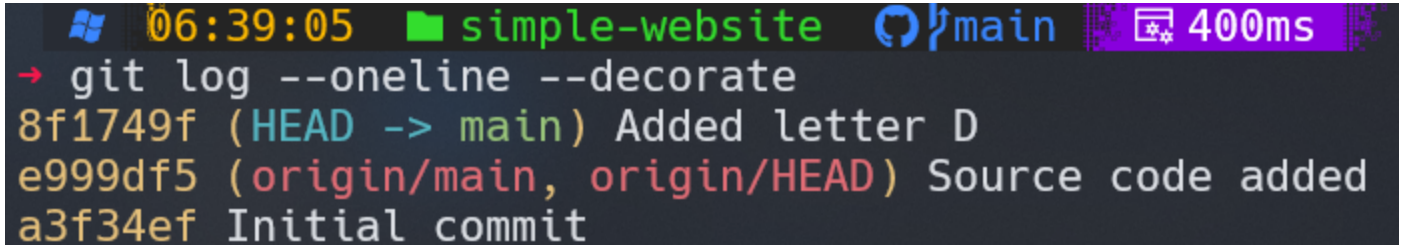


Viewing Git log from GitKraken

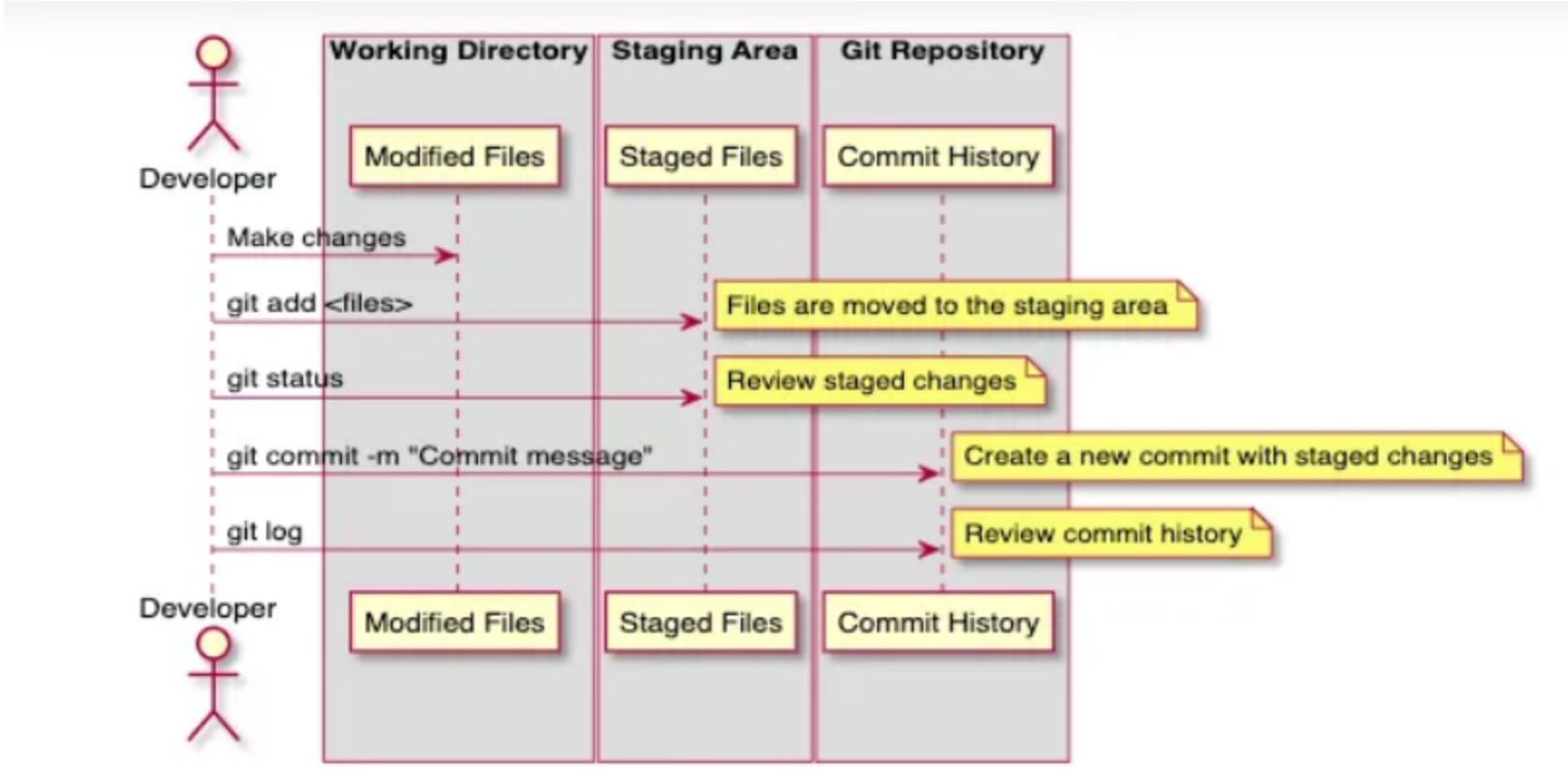

GitKraken
File Edit View Help
Workspaces
simple-website
Default Profile
workspace
repository
branch
simple-website
main
Undo
Redo
Pull
Push
Branch
Stash
Pop
Terminal
Viewing 2/2
Show All
Filter (Ctrl + Alt + f)
LOCAL 1/1
main 1↑
REMOTE 1/1
origin
main
PULL REQUESTS 0
ISSUES
TEAMS
TAGS 0/0
SUBMODULES 0
GITHUB ACTIONS 0
BRANCH / TAG
GRAPH
COMMIT MESSAGE
AUTHOR
COMMIT DATE
main
main
Added letter D
Source code added
Initial commit
linduarte
19/09/2...
Terminal
Date: Tue Sep 19 06:07:03 2023 -0300
Added letter D
commit e999df570eddb019d4460fde0a57c8fef01de4f8 (origin/main, origin/HEAD)
Author: linduarte <140444382+linduarte@users.noreply.github.com>
Date: Tue Sep 19 04:45:02 2023 -0300
Source code added
commit a3f34eff07bb02bfa723baf6477eefe1689412bd
Author: linduarte <140444382+linduarte@users.noreply.github.com>
:...skipping...
commit 8f1749f2bcf9a915cda47191da40305f43b0d9ac (HEAD -> main)
Author: linduarte <clldscotti@gmail.com>
Date: Tue Sep 19 06:07:03 2023 -0300
Added letter D
commit e999df570eddb019d4460fde0a57c8fef01de4f8 (origin/main, origin/HEAD)
Author: linduarte <140444382+linduarte@users.noreply.github.com>
Date: Tue Sep 19 04:45:02 2023 -0300
Source code added
commit a3f34eff07bb02bfa723baf6477eefe1689412bd
Author: linduarte <140444382+linduarte@users.noreply.github.com>
Date: Tue Sep 19 04:24:29 2023 -0300
Initial commit
commit: 8f1749
Added letter D
linduarte
authored 19/09/2023 @ 06:07
parent: e999df
3 modified
Path
Tree
View all files
cities.html
names.html
pilots.html
6 days of GitKraken Trial Remaining - Upgrade Now
100%
Support
TRIAL
9.8.2

A commit is a central concept in Git. Each commit represents a version of your project. You can also think of a commit as a snapshot of your project at a specific point in time. Commits allow users to undo changes in a project and go back to previous versions. We will shortly explore these features.

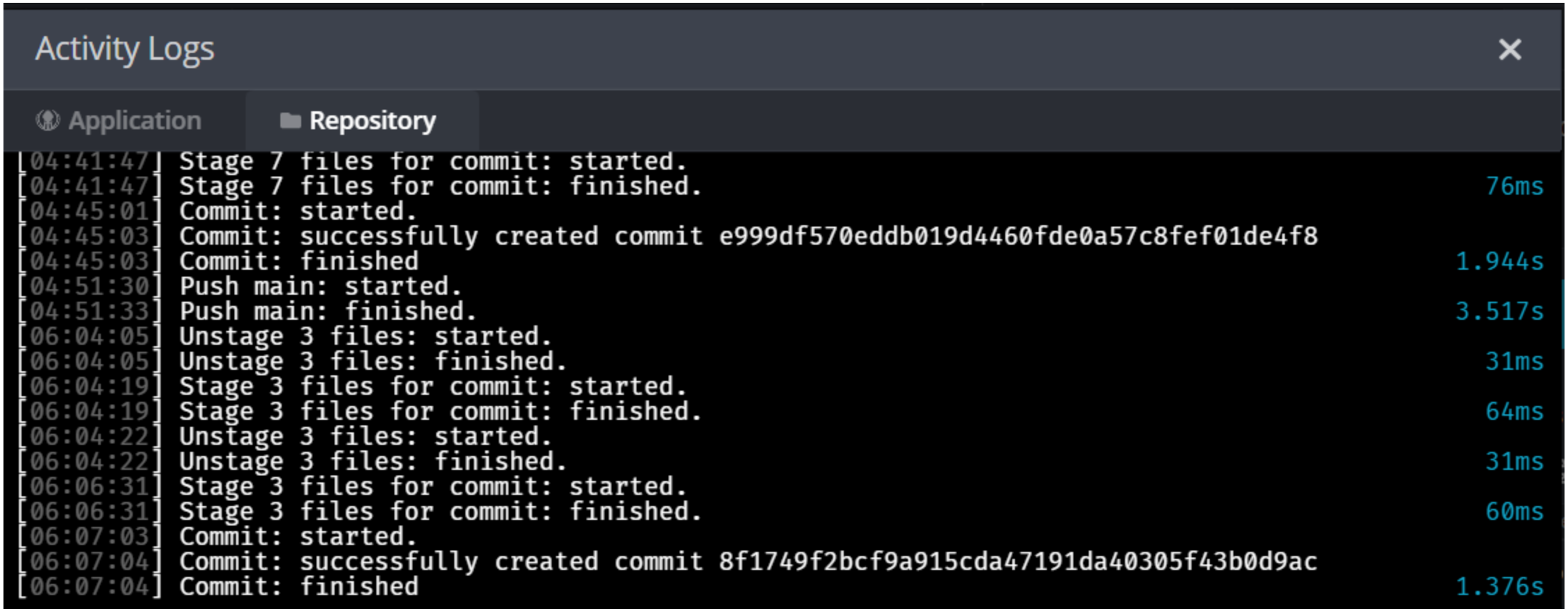
It is also possible to apply switches to the git log command to output information in a more compact format.



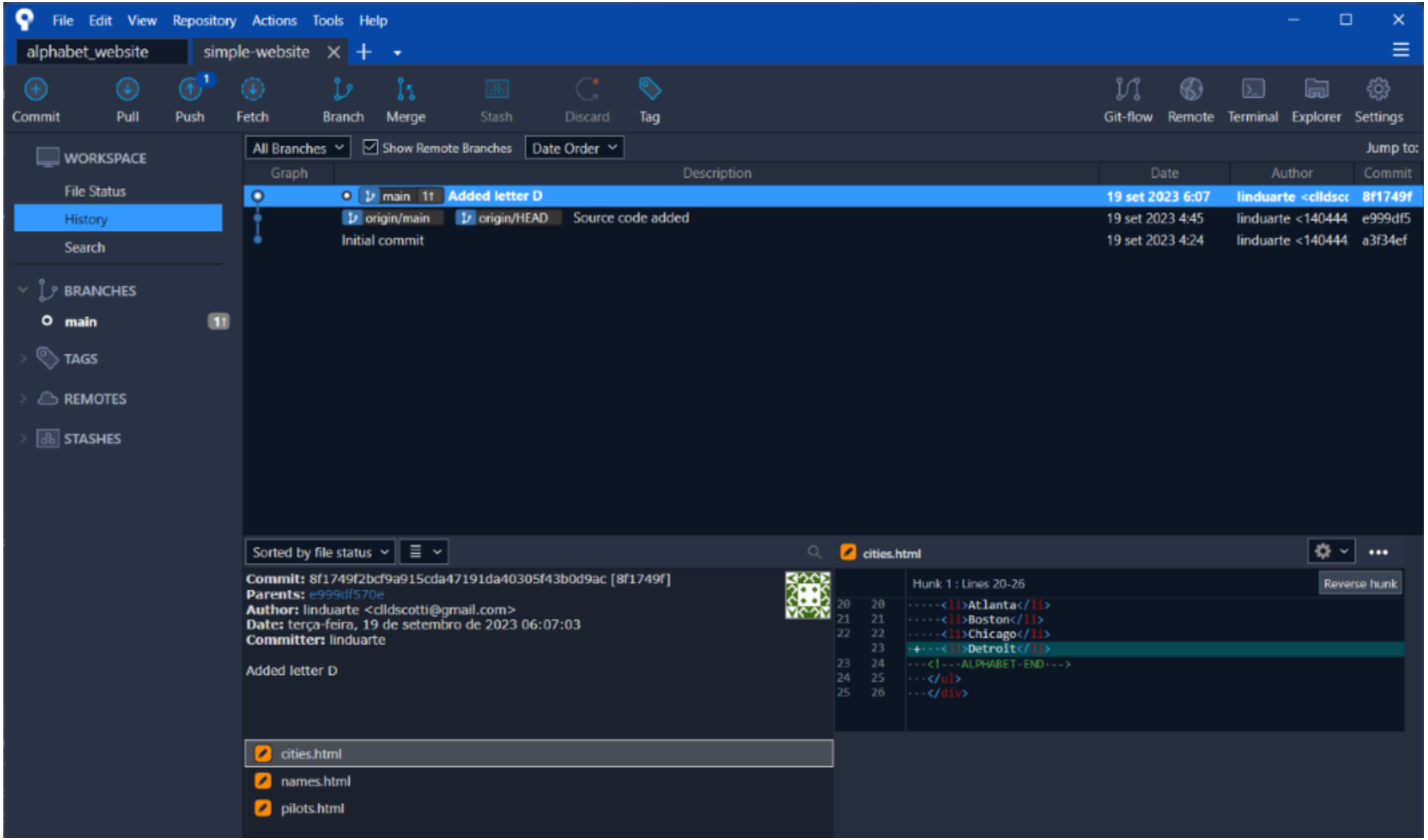
A graphical vision of Git commit



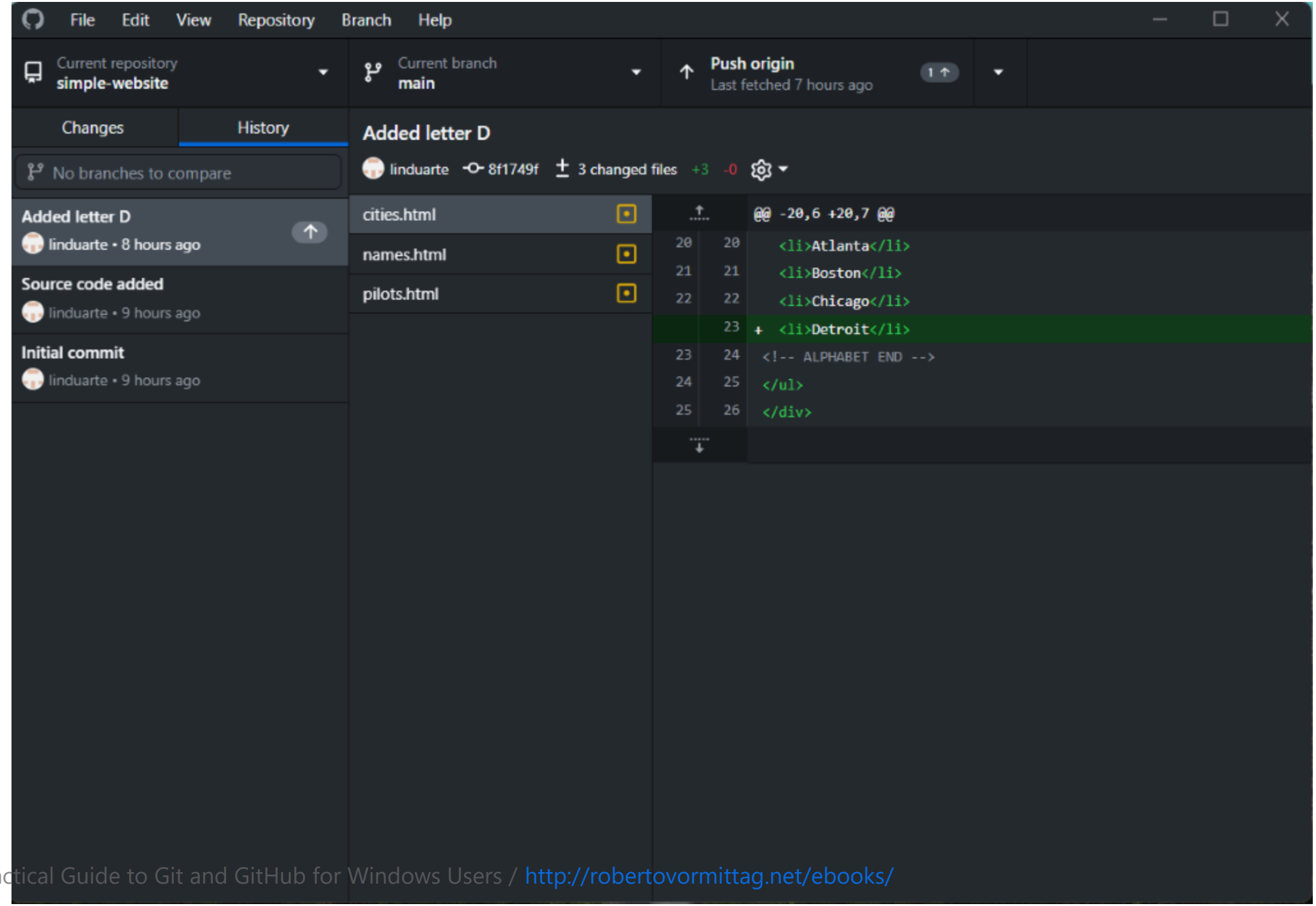
Git Kraken give us a full log up to now



A vision from Sourcetree app


File Edit View Repository Actions Tools Help
alphabet_website
simple-website
Commit
Pull
Push
Fetch
Branch
Merge
Stash
Discard
Tag
Git-flow
Remote
Terminal
Explorer
Settings
WORKSPACE
File Status
History
Search
BRANCHES
main 1↑
TAGS
REMOTES
STASHES
All Branches
Show Remote Branches
Date Order
Jump to:
Graph
Description
Date
Author
Commit
main 1↑ Added letter D
19 set 2023 6:07
linduarte <clldsc
8f1749f
origin/main
origin/HEAD
Source code added
19 set 2023 4:45
linduarte <140444
e999df5
Initial commit
19 set 2023 4:24
linduarte <140444
a3f34ef
Sorted by file status
Commit: 8f1749f2bcf9a915cda47191da40305f43b0d9ac [8f1749f]
Parents: e999df570e
Author: linduarte <clldscotti@gmail.com>
Date: terça-feira, 19 de setembro de 2023 06:07:03
Committer: linduarte
Added letter D
cities.html
names.html
pilots.html
cities.html
Hunk 1 : Lines 20-26
Reverse hunk
20 20 ·····
Atlanta
21 21 ·····
Boston
22 22 ·····
Chicago
23 +···
Detroit
23 24 ···<!---ALPHABET-END--->
24 25 ···
25 26 ···

A vision from GitHub Desktop

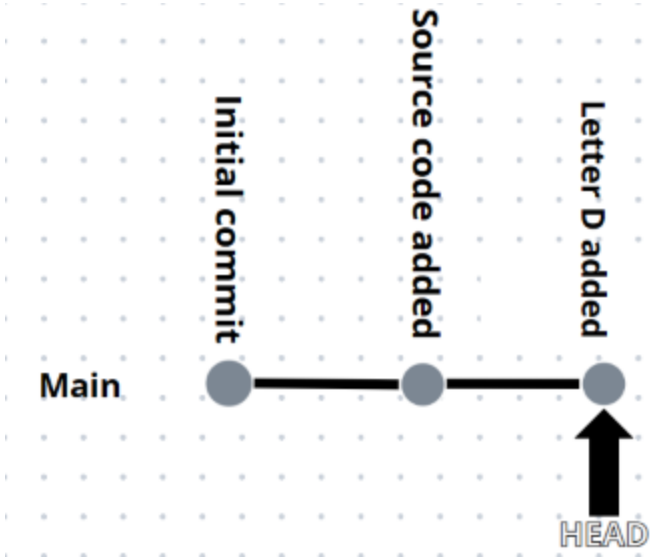

File
Edit
View
Repository
Branch
Help
Current repository
simple-website
Current branch
main
Push origin
Last fetched 7 hours ago
1
Changes
History
No branches to compare
Added letter D
linduarte • 8 hours ago
Source code added
linduarte • 9 hours ago
Initial commit
linduarte • 9 hours ago
Added letter D
linduarte
8f1749f
3 changed files +3 -0
cities.html
names.html
pilots.html
@@ -20,6 +20,7 @@
20 20 Atlanta
21 21 Boston
22 22 Chicago
+ 23 Detroit
23 24 <!-- ALPHABET END -->
24 25
25 26

What is a Branch?

A branch represents an *independent line of development* in a project with its own *separate history of commits*.You can list the branches in the local repository by running the git branch command. The --remote switch shows the local copy of the branches in the remote repository on GitHub.

You can list the branches in the local repository by running the git branch command. The --remote switch shows the local copy of the branches in the remote repository on GitHub.

Right now your repository has only a single branch called main both locally and remotely. When you first setup a repository Git creates the main branch automatically for you. All Git projects have a main branch by default. If your Git project were a tree the main branch would be the trunk. From the trunk it is possible to manually create separate lines of development as independent branches with their own separate commits.



Comparing Versions

It is important at this point to note that there are several versions of a file in a Git project:

1 - The working directory version (the one that you use for editing)
2 - The staged version (after you run git add to add the file to the index for the next commit)

3 - The committed versions (one version for each commit)

The git diff command shows changes between the working directory, index and commit versions.

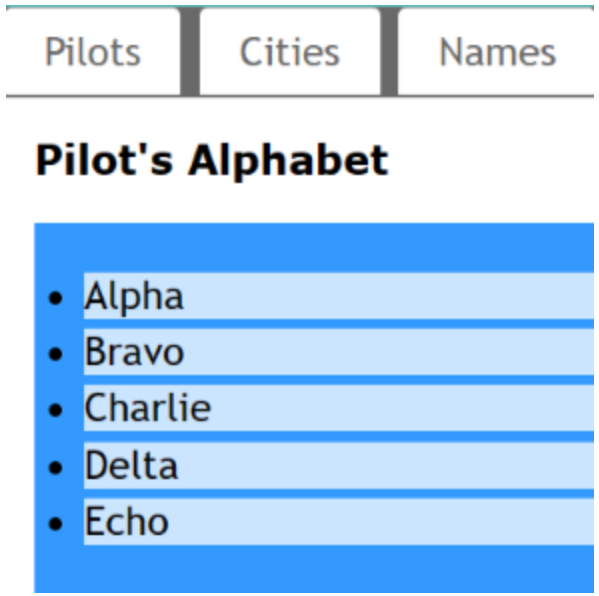
To explore the capabilities of the git diff command we need to create some additional versions.

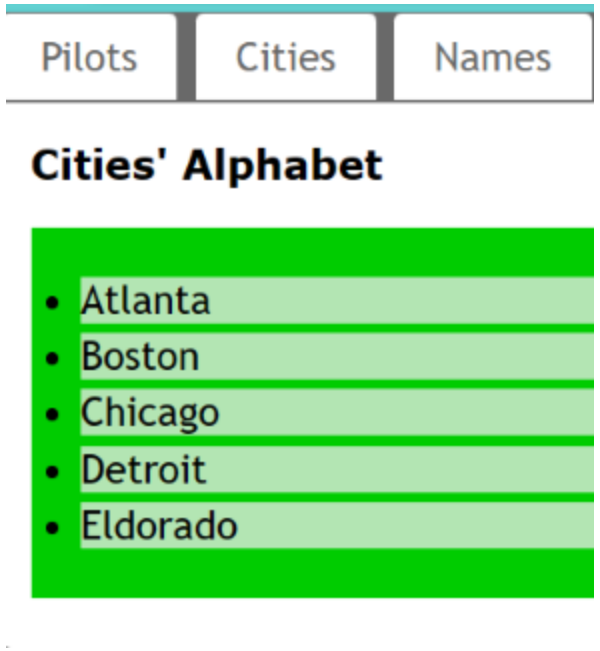
Implement the following feature request: add the letter E to the Phonetic Website.

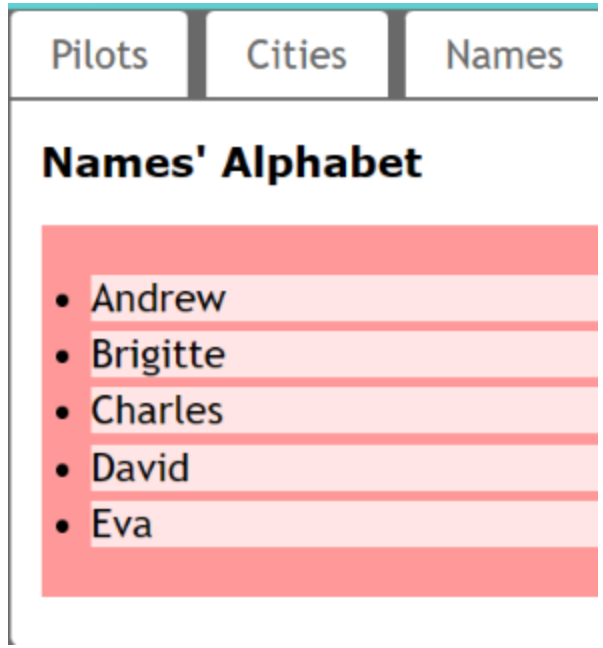
```
<h1>Pilot's Alphabet</h1>
<ul id="pilot">
<!-- ALPHABET START -->
  <li>Alpha</li>
  <li>Bravo</li>
  <li>Charlie</li>
  <li>Delta</li>
  <li>Echo</li>
<!-- ALPHABET END -->
```

```
<h1>Cities' Alphabet</h1>
<ul id="city">
<!-- ALPHABET START -->
  <li>Atlanta</li>
  <li>Boston</li>
  <li>Chicago</li>
  <li>Detroit</li>
  <li>Eldorado</li>
<!-- ALPHABET END -->
```

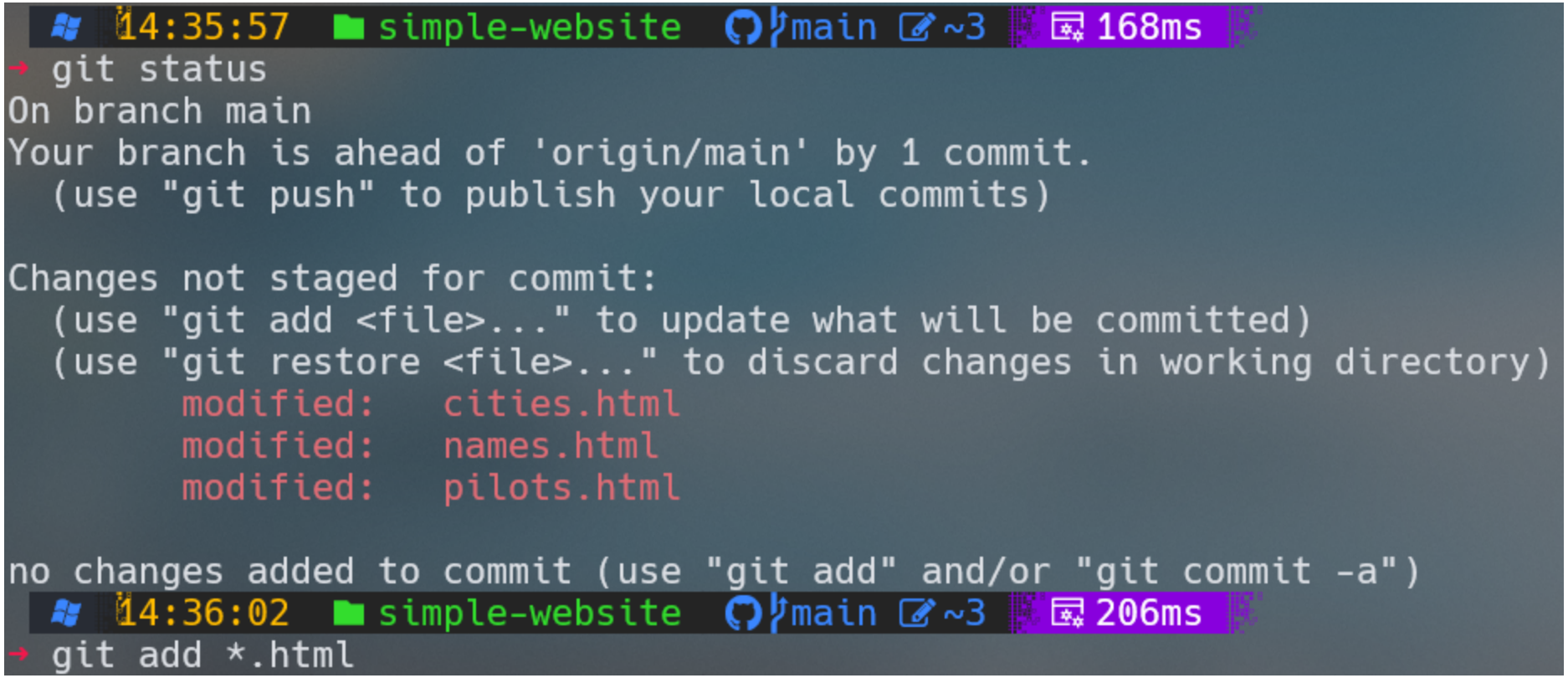
```
<h1>Names' Alphabet</h1>
<ul id="name">
<!-- ALPHABET START -->
  <li>Andrew</li>
  <li>Brigitte</li>
  <li>Charles</li>
  <li>David</li>
  <li>Eva</li>
<!-- ALPHABET END -->
```


Pilots
Cities
Names
Pilot's Alphabet
Alpha
Bravo
Charlie
Delta
Echo


Pilots
Cities
Names
Cities' Alphabet
Atlanta
Boston
Chicago
Detroit
Eldorado


Pilots
Cities
Names
Names' Alphabet
Andrew
Brigitte
Charles
David
Eva

Check status and add files to the index (staging area)



Now implement another feature request: add the letter F to the Phonetic Website. For example you can add the words "Foxtrot" to pilots.html,"Fillmore" to cities.html and "Fred" to names.html.

This time do not stage.

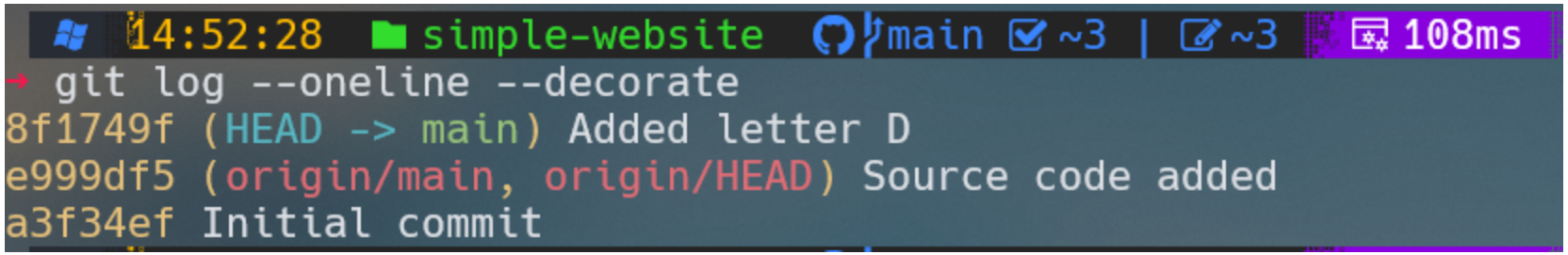
As you can see from the output of git status we have now two different versions of the HTML files. One version contains the staged changes in the first exercise. The second one contains the unstaged changes we did in the second exercise. And of course we also have the committed versions as shown by git log:

```
14:47:47 simple-website main ~3 | ~3 190ms
→ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   cities.html
        modified:   names.html
        modified:   pilots.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   cities.html
        modified:   names.html
        modified:   pilots.html
```

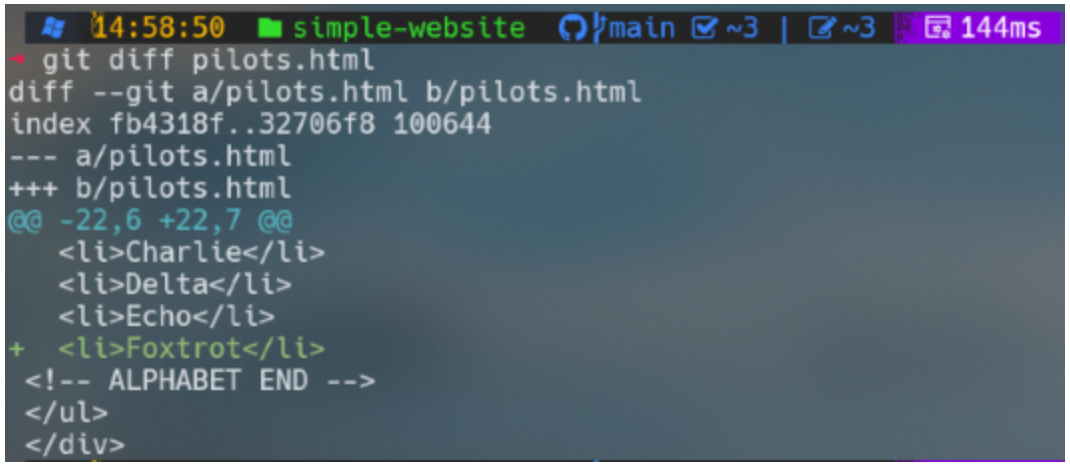
Let´s check with git log the committed versions



We can now use git diff to view the differences between the various versions. We will take the pilots.html file as an example.

Viewing Unstaged Changes

To view the changes in pilots.html that have not been staged type:



The output shows that the line containing the word Foxtrot has been added as indicated by the plus (+) sign. That is expected as we have added the word Foxtrot without staging the change in the second exercise.

Let´s check git log and viewing unstaged changes by git diff

```
  15:00:24  simple_website  main ~3 | ~3  114ms
→ git log --oneline --decorate
aaabd6b (HEAD -> main) Added letter D
7cdb449 (origin/main, origin/HEAD) Source code added
0af44e4 Initial commit
  15:00:50  simple_website  main ~3 | ~3  186ms
→ # Using git diff to view the differences between the varios versions
  15:04:37  simple_website  main ~3 | ~3
→ git diff pilots.html
diff --git a/pilots.html b/pilots.html
index fb4318f..32706f8 100644
--- a/pilots.html
+++ b/pilots.html
@@ -22,6 +22,7 @@
    <li>Charlie</li>
    <li>Delta</li>
    <li>Echo</li>
+   <li>Foxtrot</li>
  <!-- ALPHABET END -->
  </ul>
  </div>
```

Interpreting the output of the `git diff pilots.html` command line by line.

1. `diff --git a/pilots.html b/pilots.html`: This line indicates that the `git diff` command is comparing two versions of the file `pilots.html`. The `a/pilots.html` represents the original file, while `b/pilots.html` represents the modified file.
2. `index fb4318f..32706f8 100644`: This line provides information about the file index. It shows the commit hashes (`fb4318f` and `32706f8`) of the original and modified versions of `pilots.html`. The `100644` represents the file mode, which is typically associated with regular files.

3. `--- a/pilots.html`: This line indicates the start of the section related to the original version of the file (`a/pilots.html`).
4. `+++ b/pilots.html`: This line indicates the start of the section related to the modified version of the file (`b/pilots.html`).
5. `@@ -22,6 +22,7 @@`: This line is the unified diff header. It provides information about the line numbers and context for the changes that follow:
    - `-22,6`: This means that the change begins at line 22 in the original file and affects 6 lines.
    - `+22,7`: This means that the change begins at line 22 in the modified file and affects 7 lines. The `+` indicates that a line has been added.

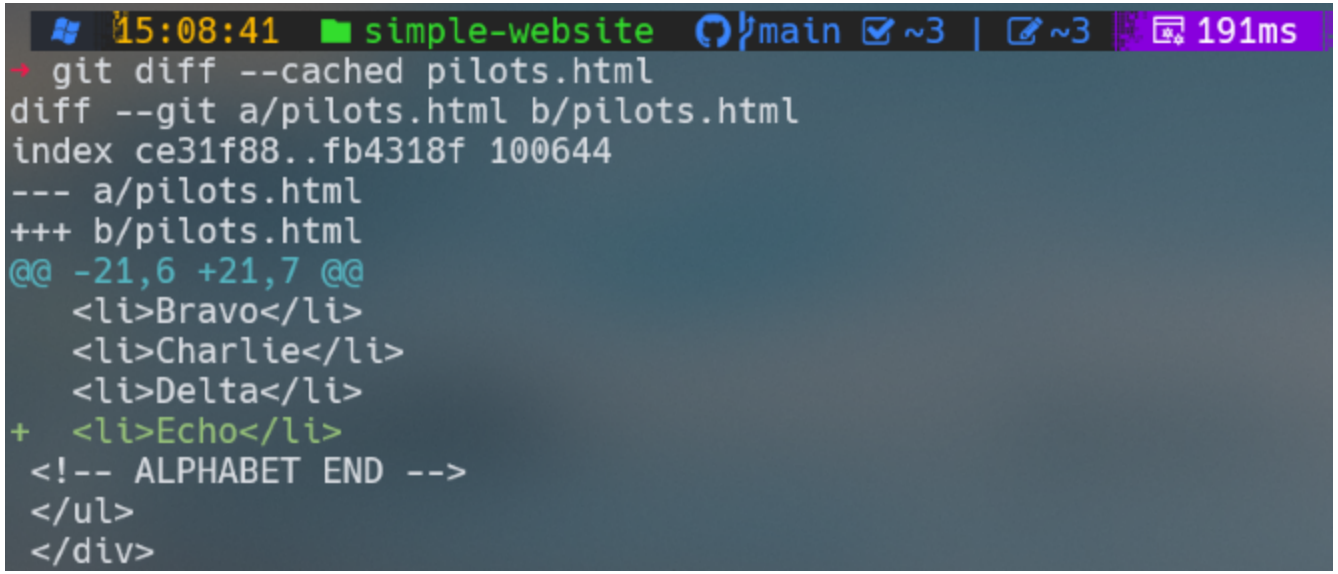
So, in summary, the git diff output indicates that a single line "

- "Foxtrot"

" was added at line 22 in the modified pilots.html file compared to the original version.

Viewing Changes Between the Index and the Last Commit

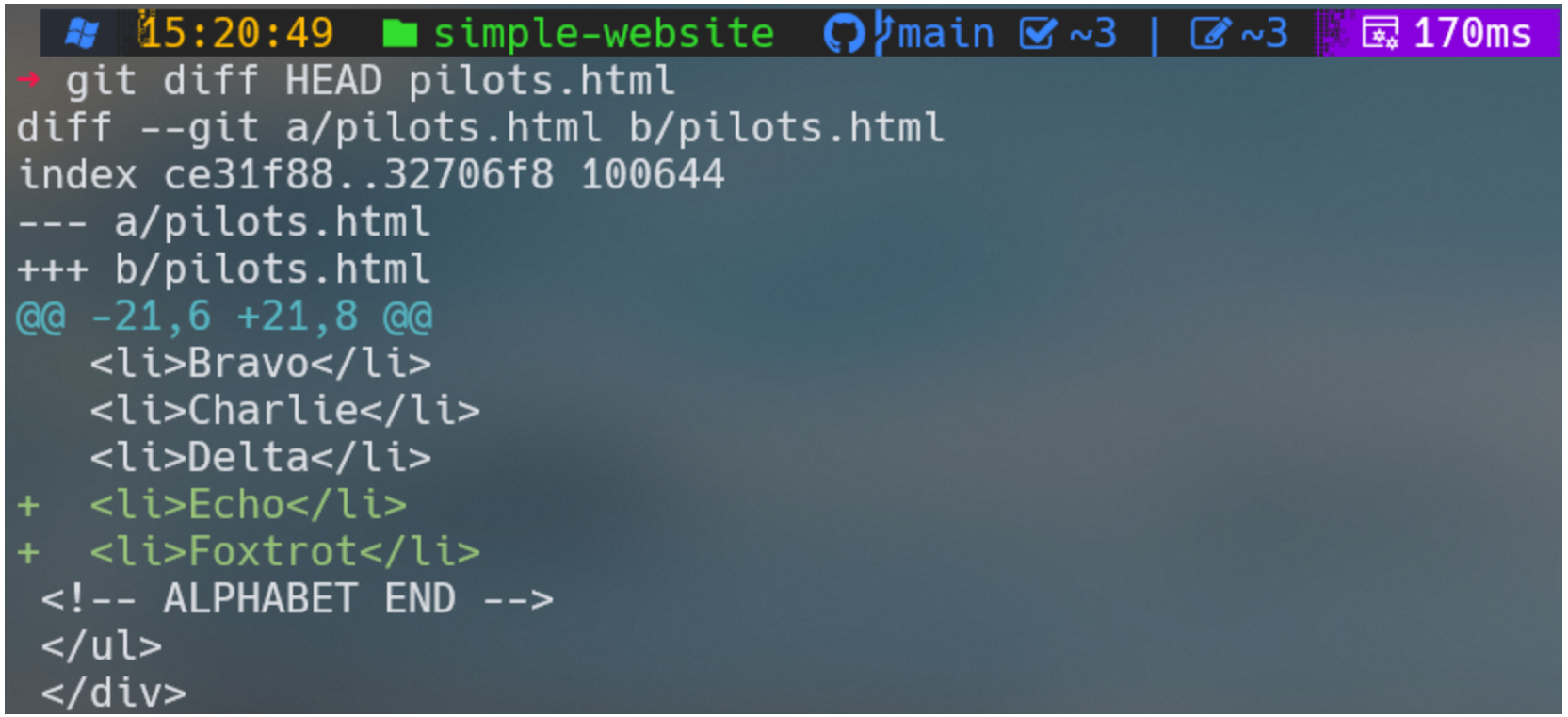
To view the changes in pilots.html between the index (staging area) and the last commit use the --cached switch:



The output shows that the line containing the word Echo has been added as indicated by the plus (+) sign. That is expected since we have staged this change in the first exercise.

Viewing Changes Since the Last Commit

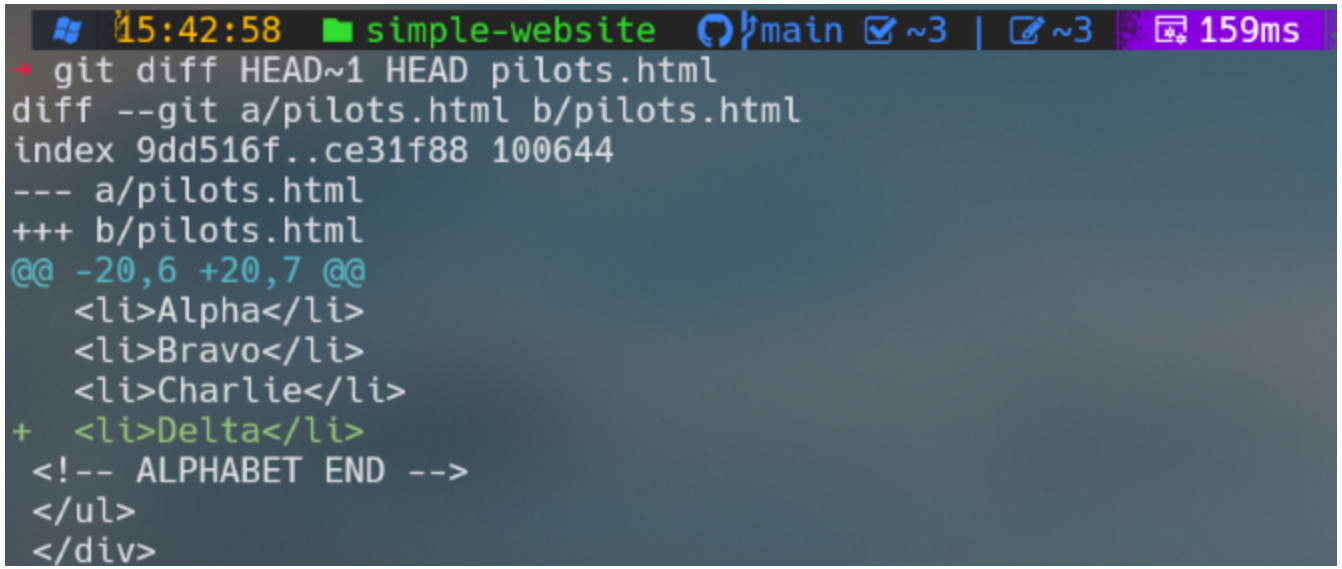
To view the changes in pilots.html since the last commit type:



The output shows that the lines containing the words Echo and Foxtrot have been added as indicated by the plus (+) signs. The git diff command interprets HEAD as the hash of the last commit. The result is expected since the last commit occurred before we made the changes in the previous two exercises.

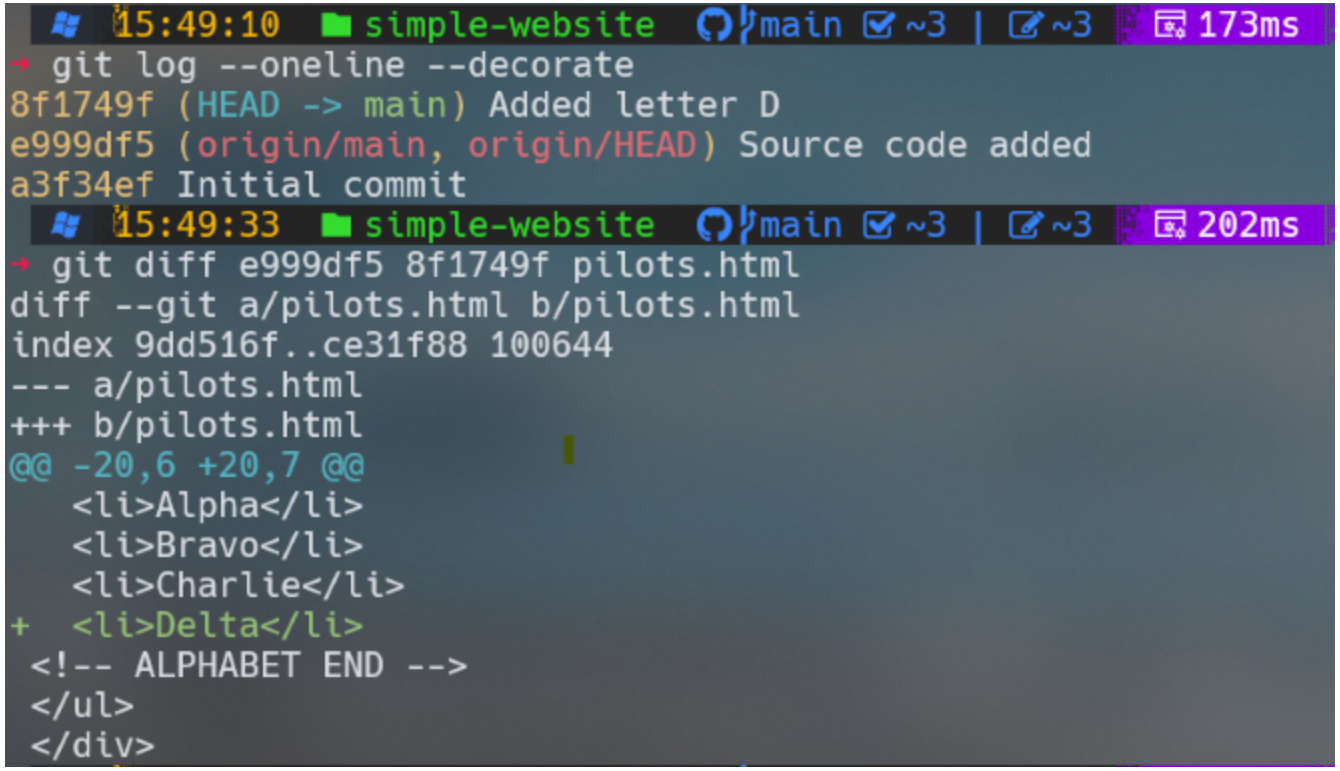
Viewing Changes Between Any Two Commits

To view the changes between the last commit and the commit before last type:

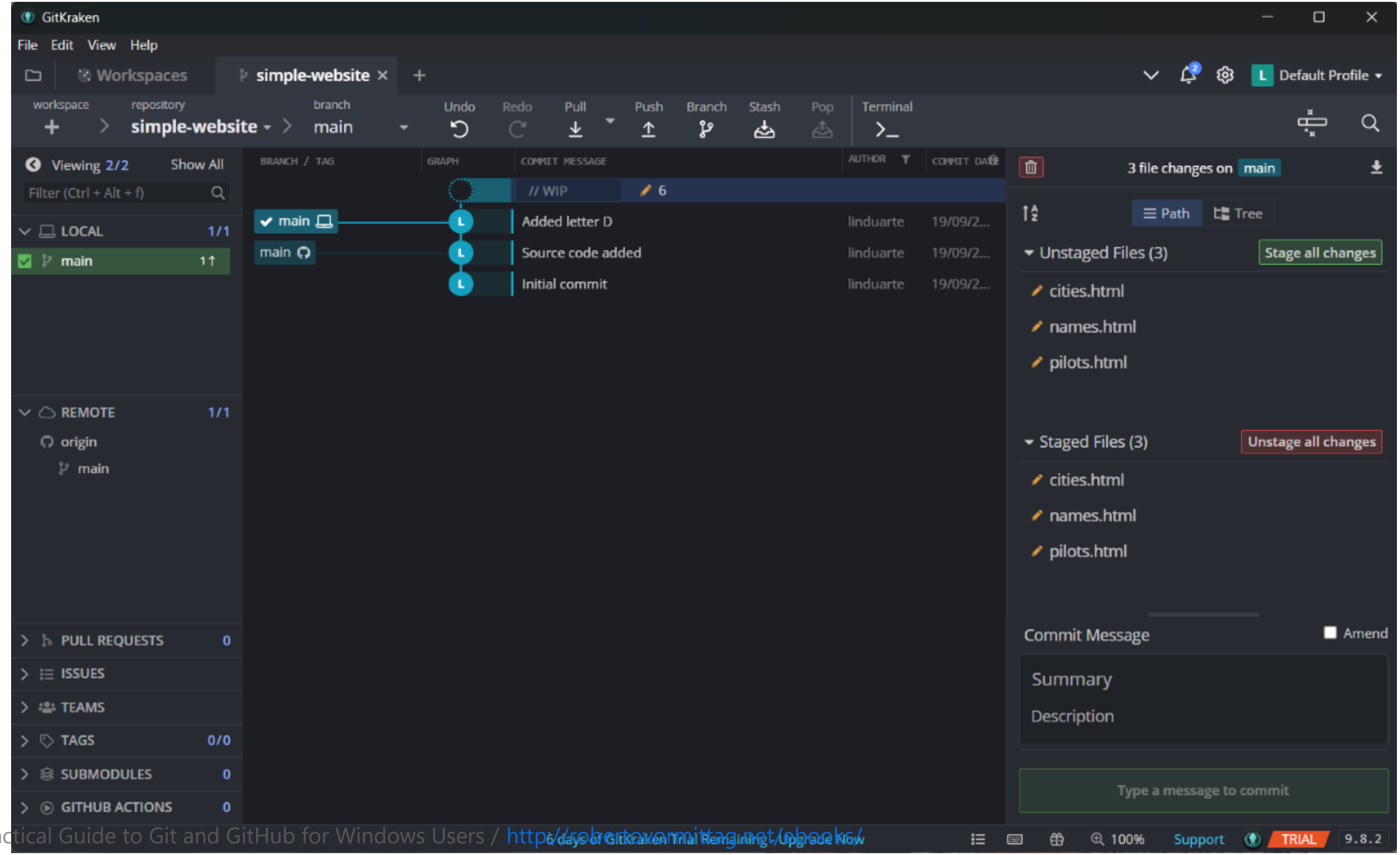


The output shows that the line containing the word Delta has been added as indicated by the plus (+) sign. The git diff command interprets HEAD~1 as the hash of the commit before last.

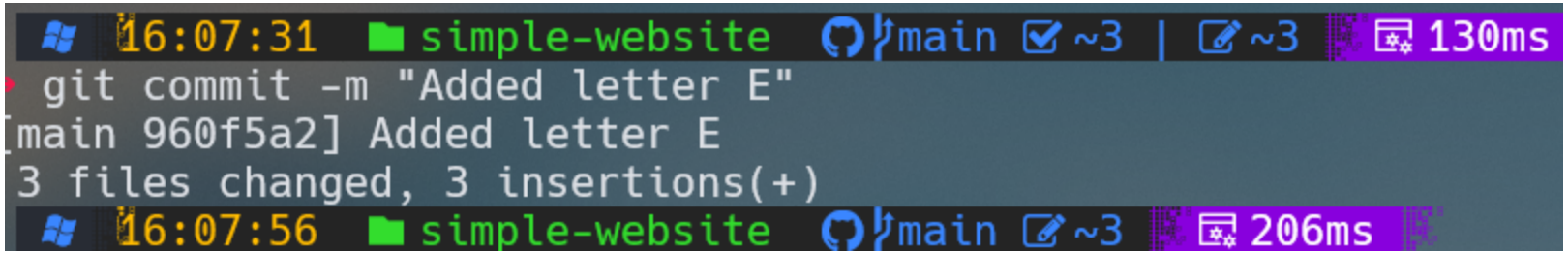
You can also use the short hash obtained from git log to view the difference between any two commits:



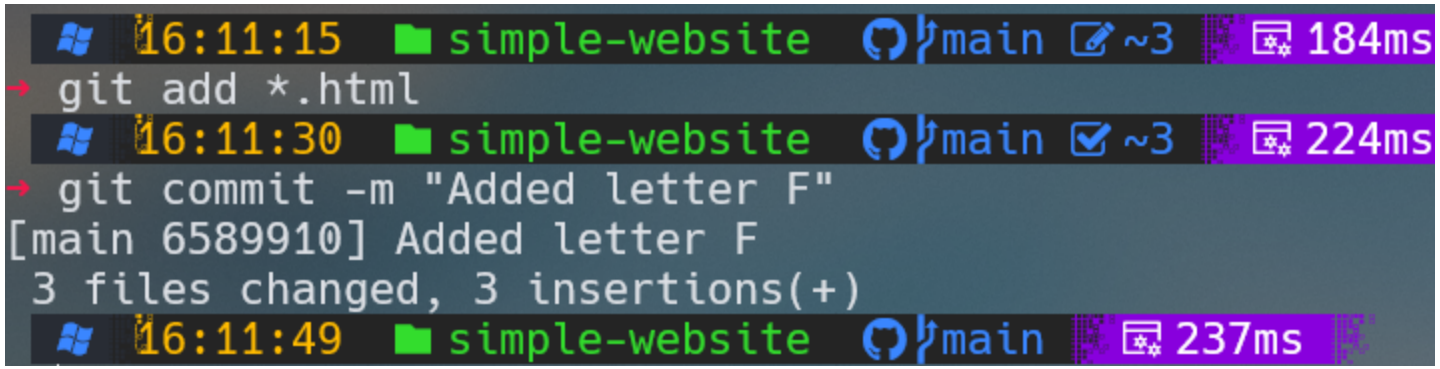
In the example above the hashes e999df5 and 8f1749f identify the before last and last commits.


GitKraken
File Edit View Help
Workspaces
simple-website
Default Profile
workspace
repository
branch
simple-website
main
Undo
Redo
Pull
Push
Branch
Stash
Pop
Terminal
Viewing 2/2
Show All
Filter (Ctrl + Alt + f)
LOCAL 1/1
main 1↑
REMOTE 1/1
origin
main
PULL REQUESTS 0
ISSUES
TEAMS
TAGS 0/0
SUBMODULES 0
GITHUB ACTIONS 0
BRANCH / TAG
GRAPH
COMMIT MESSAGE
AUTHOR
COMMIT DATE
// WIP 6
main
Added letter D linduarte 19/09/2...
main
Source code added linduarte 19/09/2...
Initial commit linduarte 19/09/2...
3 file changes on main
Path
Tree
Unstaged Files (3)
Stage all changes
cities.html
names.html
pilots.html
Staged Files (3)
Unstage all changes
cities.html
names.html
pilots.html
Commit Message
Amend
Summary
Description
Type a message to commit
100%
Support
TRIAL
9.8.2

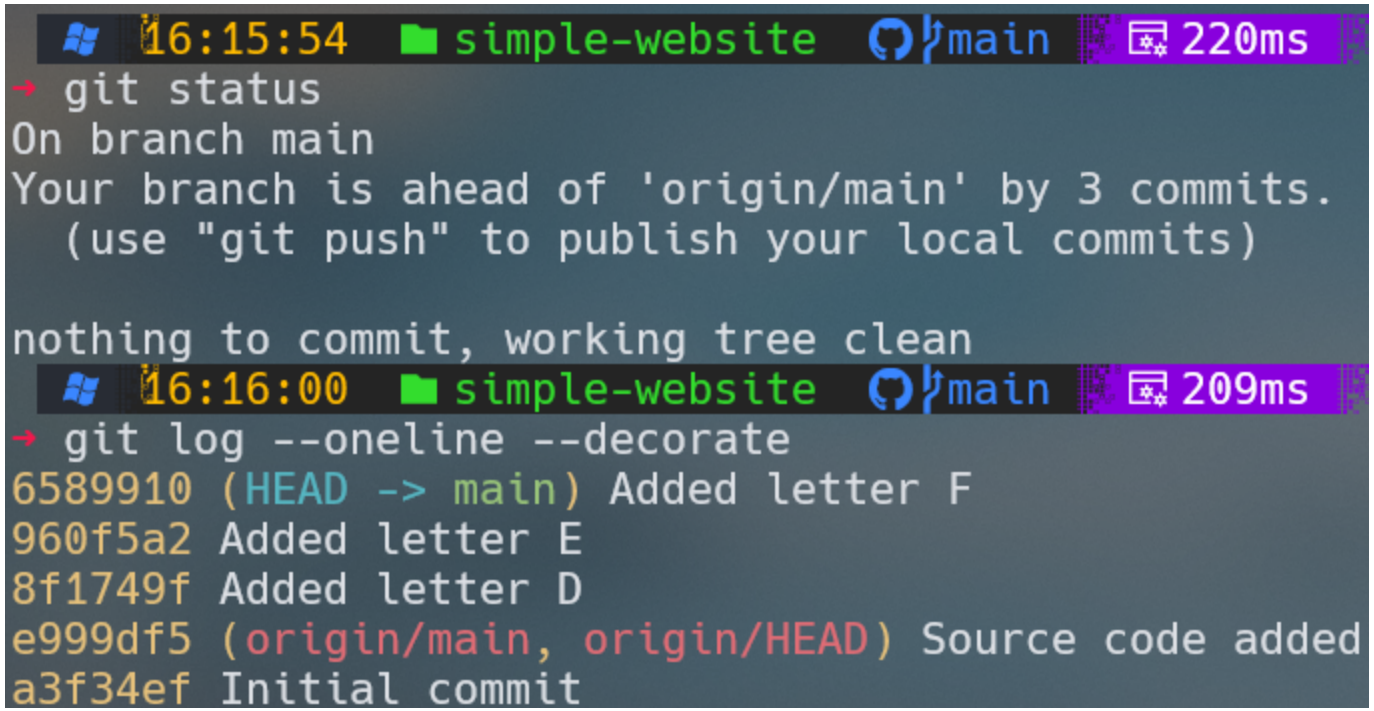
We have now covered all the basic use cases. Before moving to the next section let's commit the pending changes. First commit the letter E change request which is already staged:



Now stage and commit the letter F change request:



You should now have a clean working directory and a longer history log:



Comparing what has changed between different versions of a source code file is very useful, but what if we want to undo a change?

Undoing Changes

One of the reasons to track a project's history is to have the ability to undo changes. It is a common situation in software development: you change something, you test the change and it does not quite work the way you expected. You then decide to revert the code back to the previous version. With Git you can easily accomplish that.

There are three possible undo scenarios in Git:

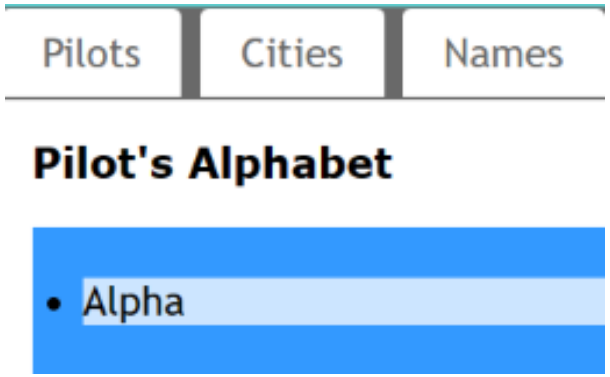
1 - Undoing changes in the working directory before staging

2 - Undoing changes after staging and before committing

3 - Undoing committed changes

Unwanted Change

To demonstrate each of the above scenarios we will need an unwanted change to undo. We will again use the Phonetic Website project to experiment with. If you have followed all the exercises in the previous sections you should now have a clean working directory without any uncommitted changes.



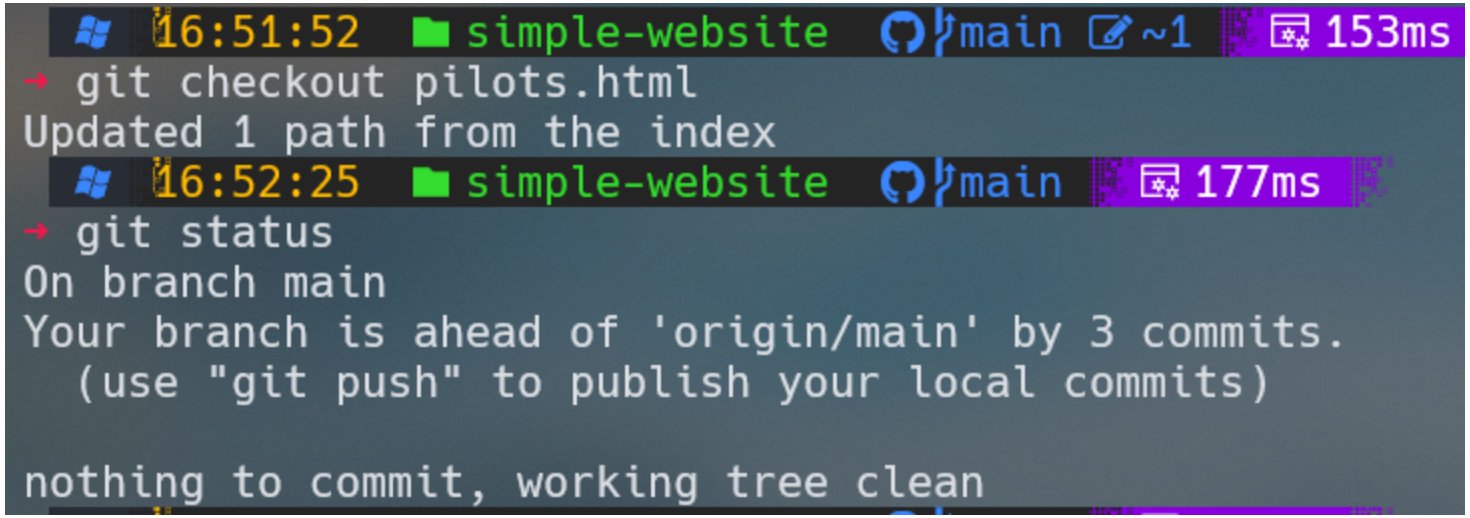
Open pilots.html in a text editor and delete all the words except "Alpha" so that the alphabet ends up looking like this:

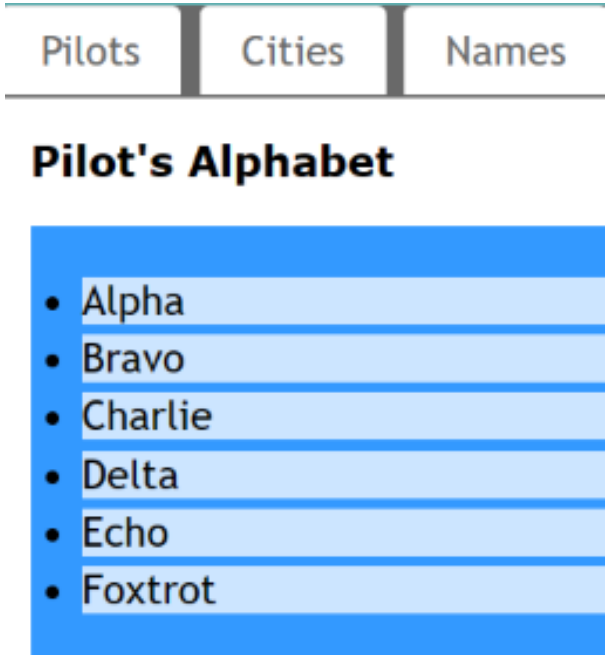
```
<h1>Pilot's Alphabet</h1>
<ul id="pilot">
<!-- ALPHABET START -->
  <li>Alpha</li>
<!-- ALPHABET END -->
```

Save the change and test how the page looks now when browsing the website. You should have only the letter A left in the Pilot's Alphabet. We will learn how to recover the lost words using Git undo features. In this simple case you could just manually add the lost words again to fix the problem. Suppose however the change involved editing dozens of lines of code in various parts of the source file. In that case it would be impossible to correct it manually. In such a situation Git undo features become invaluable.

Undoing Changes Before Staging

To recover the lost words in pilots.html (following the "Unwanted Change Exercise" at the beginning of this section) all you have to do is to revert the file back to its last committed version using the git checkout command as follows:

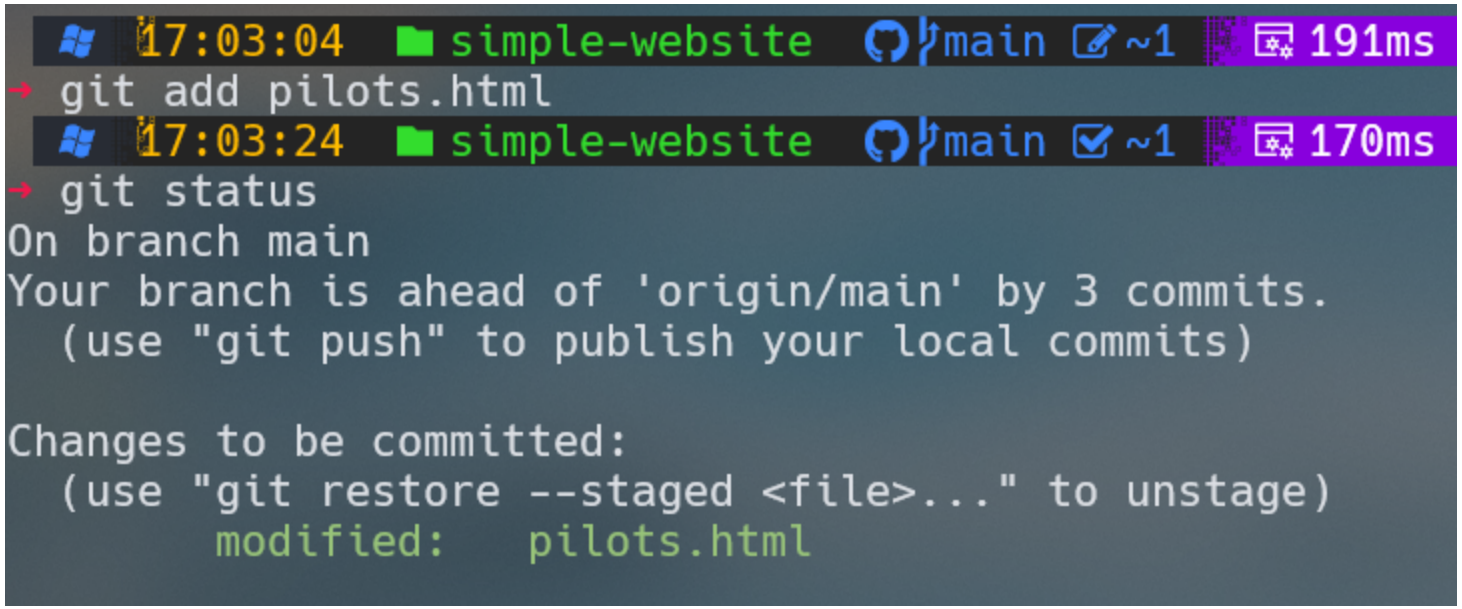




Undoing Changes After Staging

Repeat the "Unwanted Change Exercise" at the beginning of this section.Now stage the change:

```
<h1>Pilot's Alphabet</h1>
<ul id="pilot">
<!-- ALPHABET START -->
  <li>Alpha</li>
<!-- ALPHABET END -->
```



Run git status. It shows that pilots.html is ready to be committed. It also suggests to run git reset HEAD to un-stage the change. And that is what we need to do first:

The git reset command has removed the file from the index (staging area) however the unwanted change is still in the working directory. To recover the lost words, you still need to repeat the process for undoing un-staged changes and run git checkout to restore the last committed version:

Git restore x Git reset

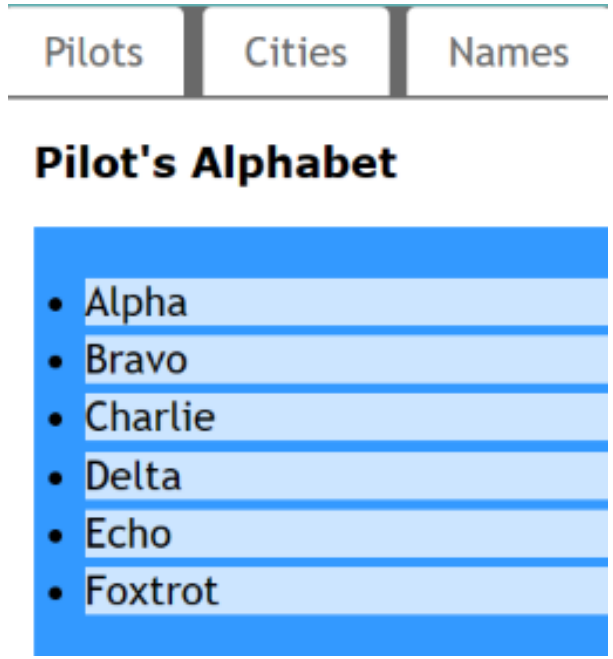
Both `git restore` and `git reset` are used to undo changes in your repository. However, they work differently.

`git restore` is used for restoring files in the working tree from the index or another commit. This doesn't update the branch. On the other hand, `git reset` is used to update your branch.

So, if you want to undo changes that you have made to a file in your working directory but haven't staged yet, you can use `git restore`. If you want to undo changes that you have made to a file and have already staged it, you can use `git reset`.

```
 17:14:27  simple-website  main ~1  175ms
→ git checkout pilots.html
Updated 1 path from the index
 17:14:39  simple-website  main  209ms
→ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```



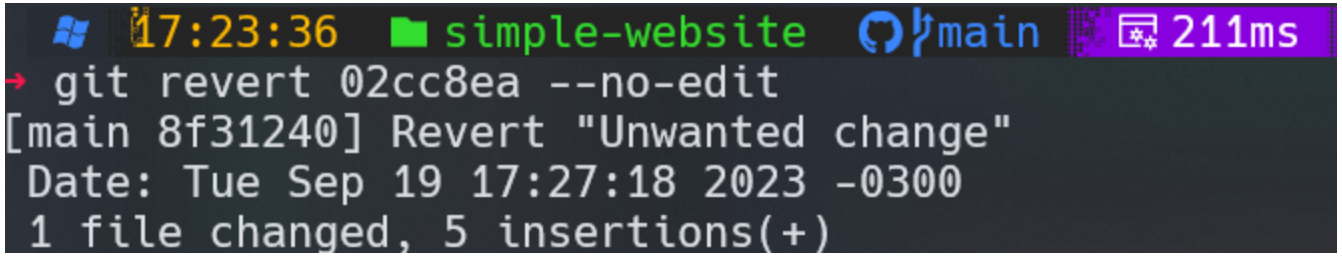
Undoing Committed Changes

Repeat the "Unwanted Change Exercise" at the beginning of this section.This time we are going to commit the unwanted change:

```
<h1>Pilot's Alphabet</h1>
<ul id="pilot">
<!-- ALPHABET START -->
  <li>Alpha</li>
<!-- ALPHABET END -->
```

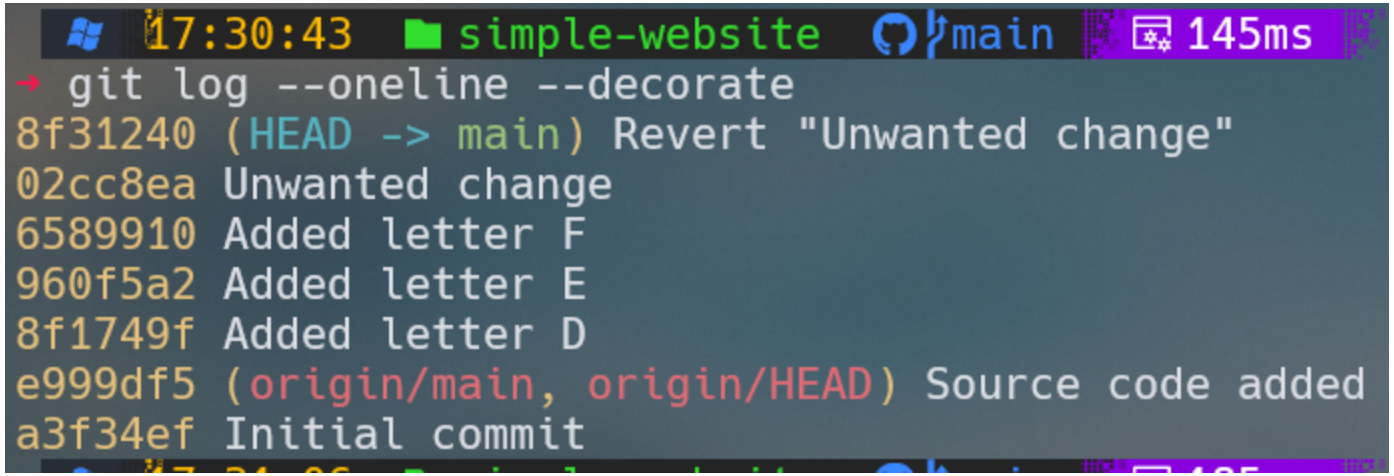
```
 17:22:28  simple-website  main ~1  171ms
→ git add pilots.html
 17:22:42  simple-website  main ~1  191ms
→ git commit -m "Unwanted change"
[main 02cc8ea] Unwanted change
 1 file changed, 5 deletions(-)
 17:23:12  simple-website  main  216ms
→ git log --oneline --decorate
02cc8ea (HEAD -> main) Unwanted change
6589910 Added letter F
960f5a2 Added letter E
8f1749f Added letter D
e999df5 (origin/main, origin/HEAD) Source code added
a3f34ef Initial commit
```

To undo the committed change you need to run the git revert command specifying the hash of the unwanted commit as shown in your git log output.



The --no-edit flag prevents the commit editor to popup.

Let´s check the project history



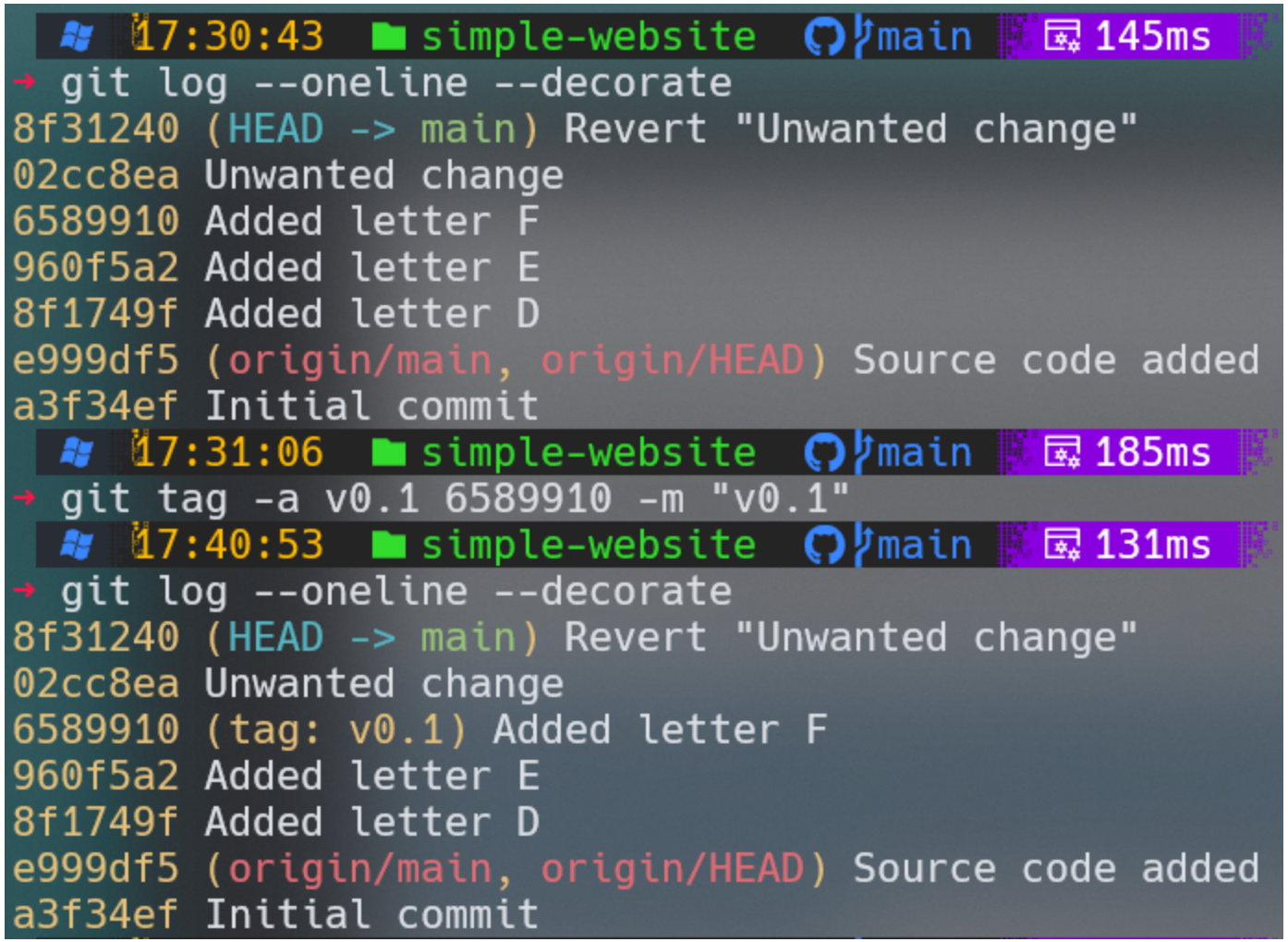
As you can see from the git log output a revert commit was added to cancel the effect of the unwanted change. Git is designed to never loose history so it keeps the commit you want to revert and overrides it with a new one.

We have covered in this section three basic undo change scenarios for a single file. Later in the sequence of slides we will learn how to navigate history and get the whole project back to a previous version. In the next section we will learn how to give a meaningful name to stable versions of a project

Tagging Versions

You can use Git to attach a tag to easily identify a stable version of a project with the git tag command. The tag can be any arbitrary string but traditional versioning schemes assign a number sequence starting with zero.

For instance, suppose you want to assign the tag v0.1 to the version (commit) where you added the letter F to the Phonetic Website project. First you need to find out what the short hash is for the corresponding commit from the history log.


17:30:43 simple-website main 145ms
git log --oneline --decorate
8f31240 (HEAD -> main) Revert "Unwanted change"
02cc8ea Unwanted change
6589910 Added letter F
960f5a2 Added letter E
8f1749f Added letter D
e999df5 (origin/main, origin/HEAD) Source code added
a3f34ef Initial commit
17:31:06 simple-website main 185ms
git tag -a v0.1 6589910 -m "v0.1"
17:40:53 simple-website main 131ms
git log --oneline --decorate
8f31240 (HEAD -> main) Revert "Unwanted change"
02cc8ea Unwanted change
6589910 (tag: v0.1) Added letter F
960f5a2 Added letter E
8f1749f Added letter D
e999df5 (origin/main, origin/HEAD) Source code added
a3f34ef Initial commit

Tagging Versions

You can use Git to attach a tag to easily identify a stable version of a project with **git tag** command.

1- **Create a Tag:**

To create a tag, you can use the git tag command followed by a tag name. For example, if you want to tag a version 1.0 of your Python project, you can do:

```
git tag 1.0
```

2- **List Tags:**

To see a list of tags in your repository, you can use:

```
git tag
```

3- **Tagging Commits:**

You can also tag specific commits by specifying the commit's SHA-1 hash:

```
git tag -a v1.1 <commit-SHA-1>
```

This creates an annotated tag with a message. Annotated tags are recommended for versioning.

Tag - Continue

4 - **Push Tags:**

Tags are not automatically pushed to remote repositories when you push changes. To push tags, you can use:

```
git push origin --tags
```

This sends your tags to the remote repository.

5 - **Checkout Tags:**

You can check out a specific tag to view the code at that version:

```
git checkout v1.0
```

6 - **Delete Tags:**

To delete a tag, use the -d option:

```
git tag -d 1.0
```

7 - **To delete a tag on the remote repository:**

```
git push origin --delete 1.0
```

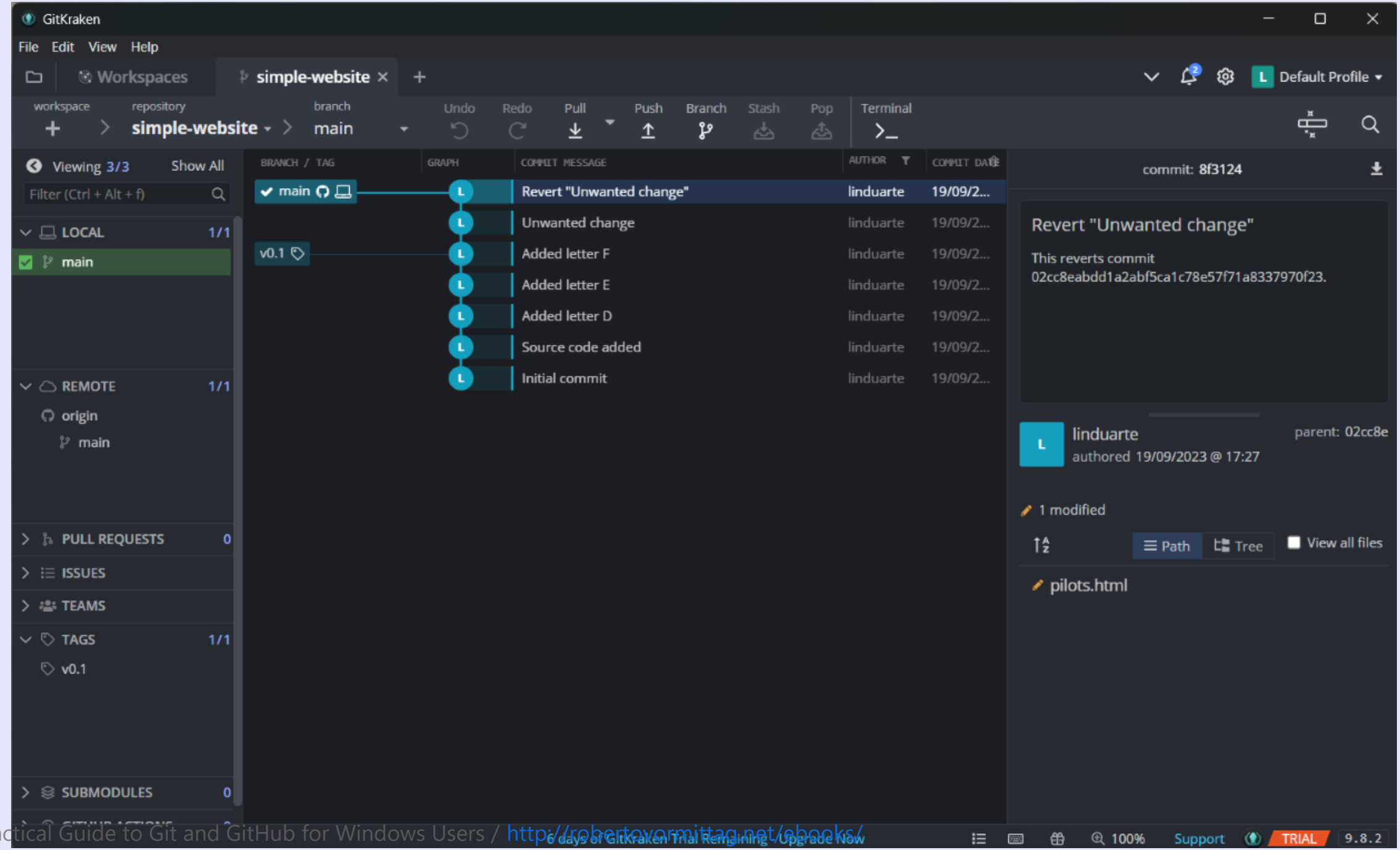
Let´s checking the status of the working tree:

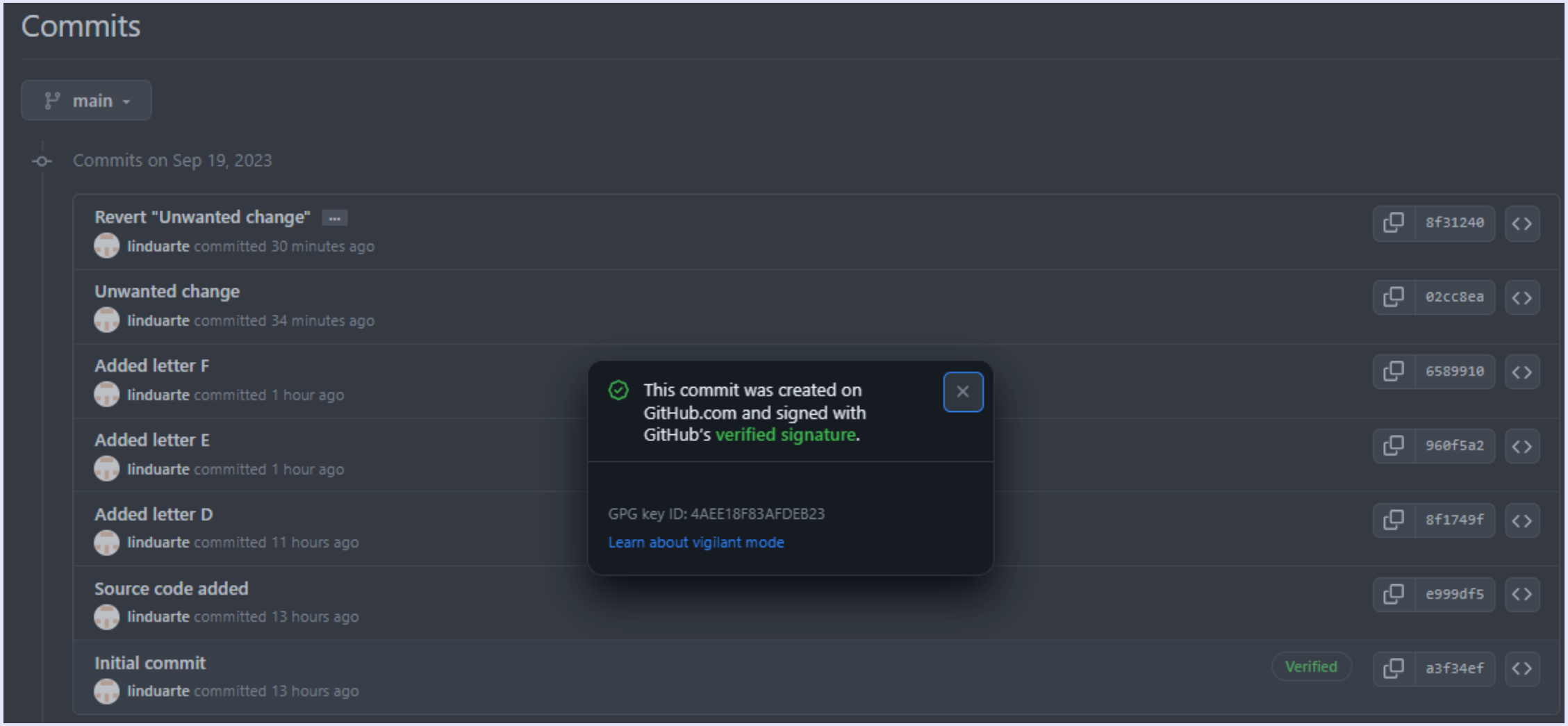
```
17:44:13  simple-website  main  182ms
→ git status
On branch main
Your branch is ahead of 'origin/main' by 5 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

The working directory is clean but the remote repository on GitHub is behind by 5 commits. It is time to synchronize:

```
17:47:09  simple-website  main  159ms
→ git push origin main
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 16 threads
Compressing objects: 100% (19/19), done.
Writing objects: 100% (19/19), 1.71 KiB | 876.00 KiB/s, done.
Total 19 (delta 12), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (12/12), completed with 3 local objects.
To https://github.com/linduarte/simple-website
   e999df5..8f31240  main -> main
```


GitKraken
File Edit View Help
Workspaces
simple-website
Default Profile
workspace
repository
simple-website
branch
main
Undo
Redo
Pull
Push
Branch
Stash
Pop
Terminal
Viewing 3/3
Show All
Filter (Ctrl + Alt + f)
LOCAL 1/1
main
REMOTE 1/1
origin
main
PULL REQUESTS 0
ISSUES
TEAMS
TAGS 1/1
v0.1
SUBMODULES 0
BRANCH / TAG
GRAPH
COMMIT MESSAGE
AUTHOR
COMMIT DATE
main
v0.1
Revert "Unwanted change" linduarte 19/09/2...
Unwanted change linduarte 19/09/2...
Added letter F linduarte 19/09/2...
Added letter E linduarte 19/09/2...
Added letter D linduarte 19/09/2...
Source code added linduarte 19/09/2...
Initial commit linduarte 19/09/2...
commit: 8f3124
Revert "Unwanted change"
This reverts commit 02cc8eabdd1a2abf5ca1c78e57f71a8337970f23.
linduarte
authored 19/09/2023 @ 17:27
parent: 02cc8e
1 modified
Path
Tree
View all files
pilots.html
100%
Support
TRIAL
9.8.2


Commits
main
Commits on Sep 19, 2023
Revert "Unwanted change"
linduarte committed 30 minutes ago
8f31240
Unwanted change
linduarte committed 34 minutes ago
02cc8ea
Added letter F
linduarte committed 1 hour ago
6589910
Added letter E
linduarte committed 1 hour ago
960f5a2
Added letter D
linduarte committed 11 hours ago
8f1749f
Source code added
linduarte committed 13 hours ago
e999df5
Initial commit
linduarte committed 13 hours ago
Verified
a3f34ef
This commit was created on GitHub.com and signed with GitHub's verified signature.
GPG key ID: 4AEE18F83AFDEB23
Learn about vigilant mode