TECNICHE DI RAPPRESENTAZIONE E MODELLIZZAZIONE DEI DATI

— Part 1 —

(2 CFU out of 6 total CFU)

Link moodle: https://moodle2.units.it/course/view.php?id=11703

Teams code: 0ftoqj8

The Integrated Development Environment

Setup a working environment:

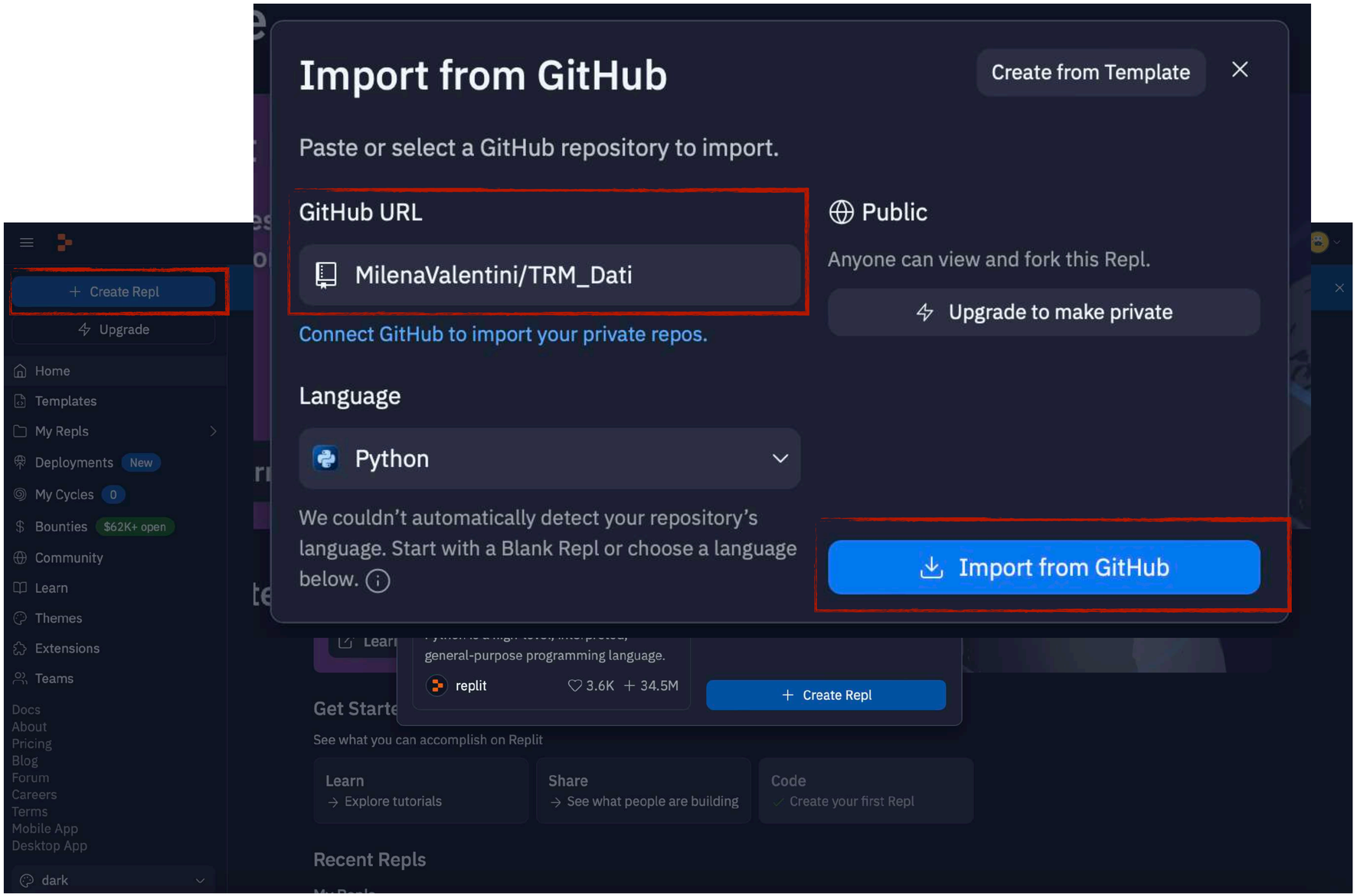
Register on Git

Create a public repository on Git

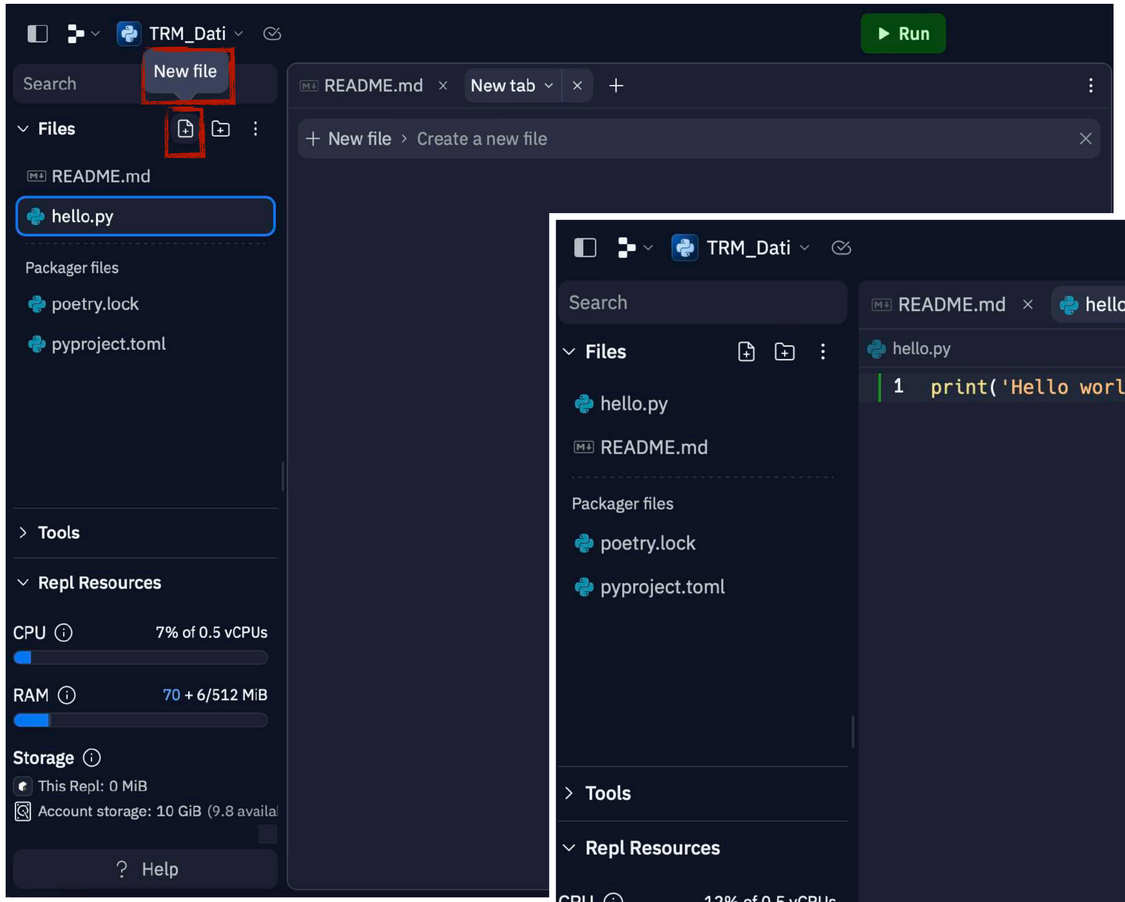
Register on repl.it (replit.com)

Create a new repl project by importing from Git the repository you have just created there

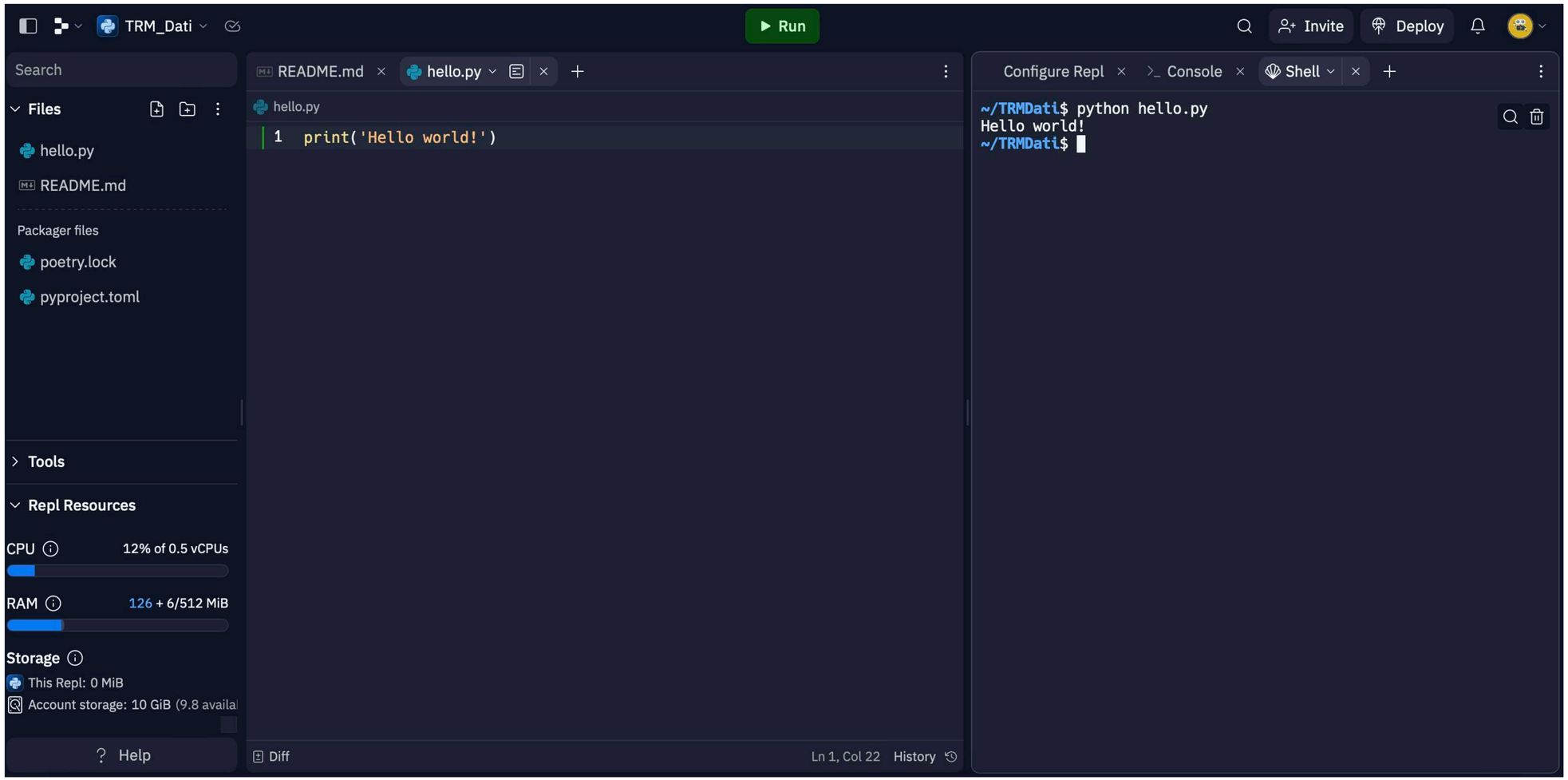
Set Python as default language



The Integrated Development Environment

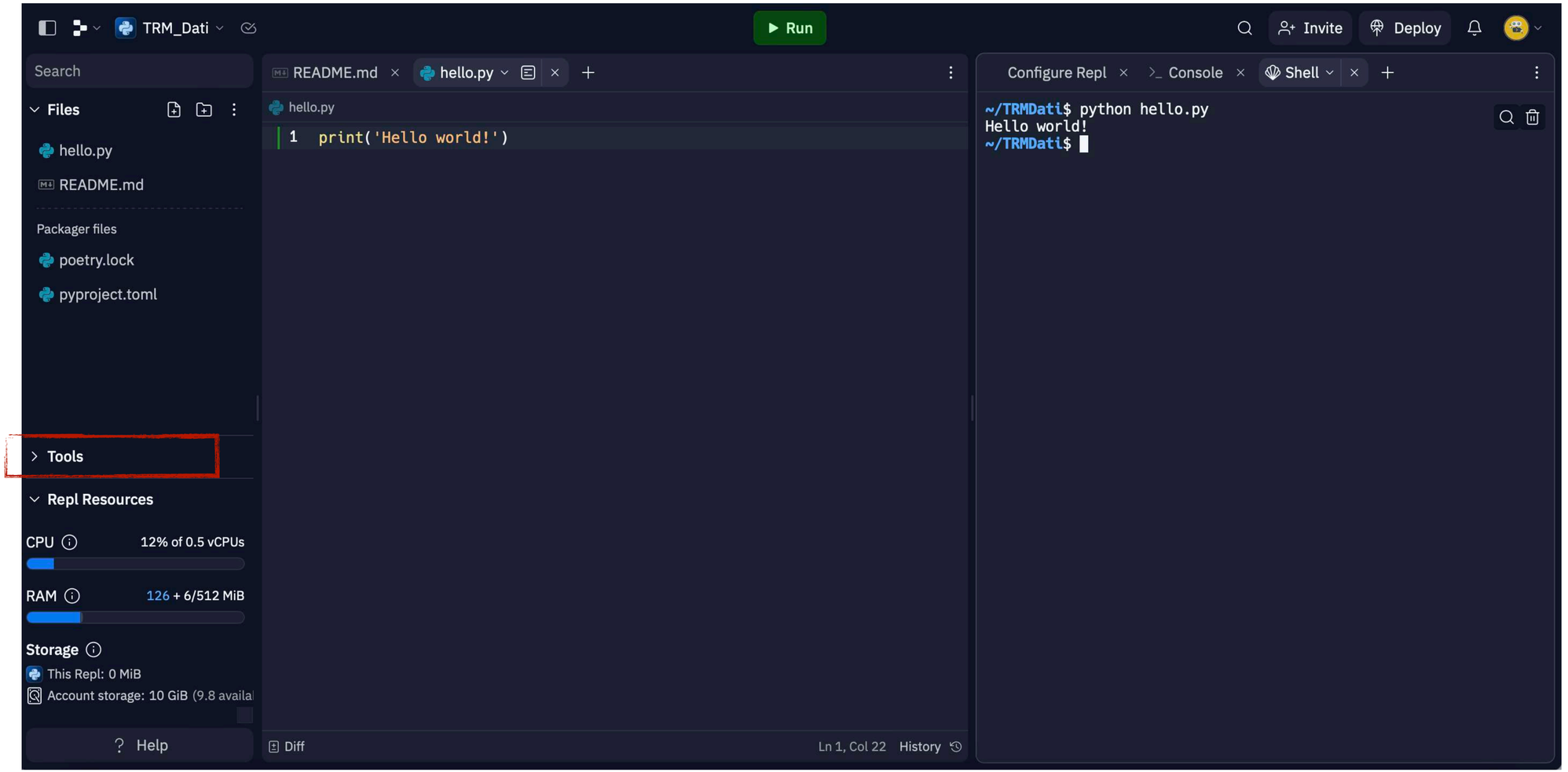


Create a new file to produce the first working script



The Integrated Development Environment

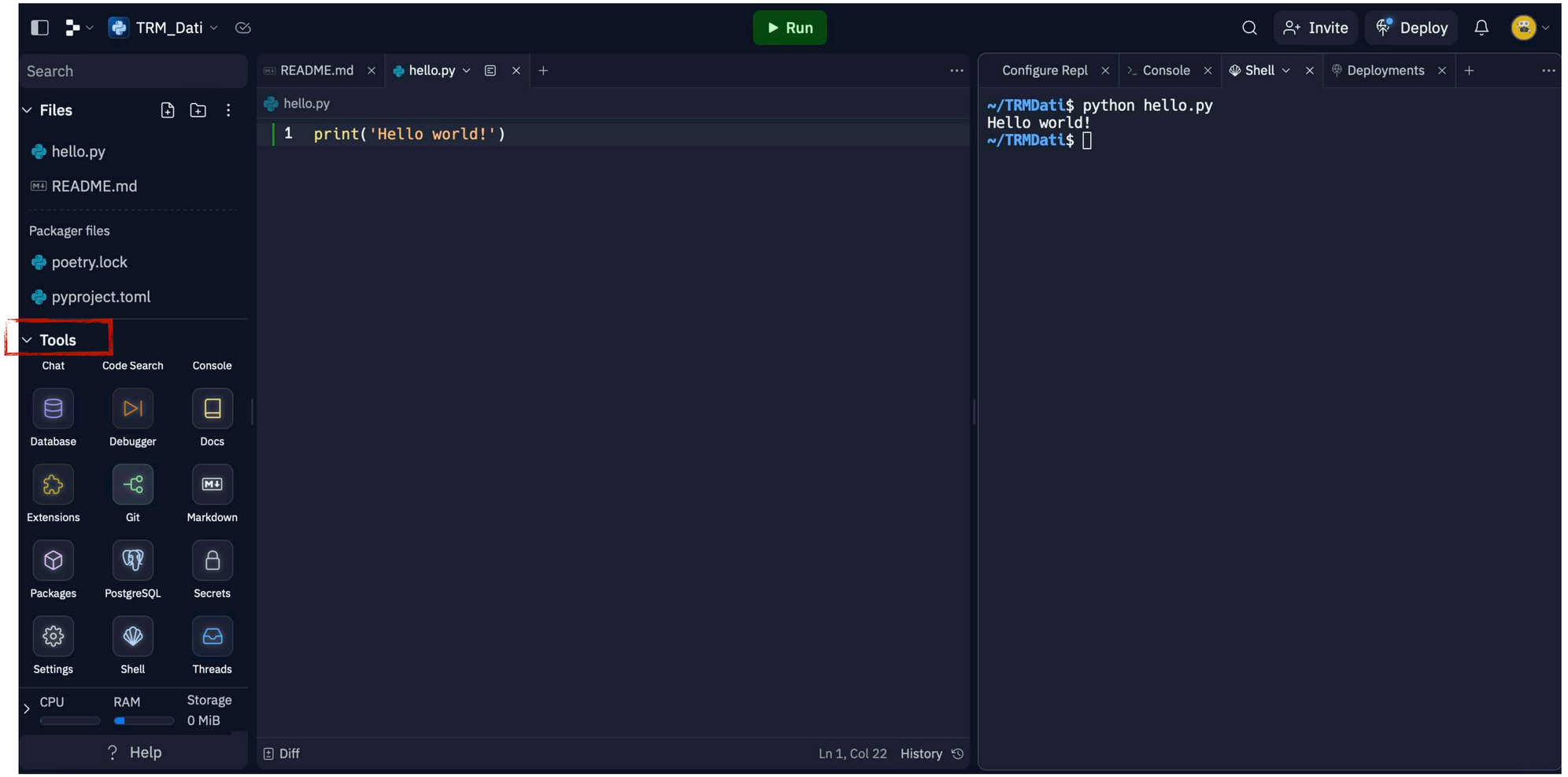
Connecting replit to Git



Other useful material: https://replit.com/talk/learn/Replit-Git-Tutorial/23331

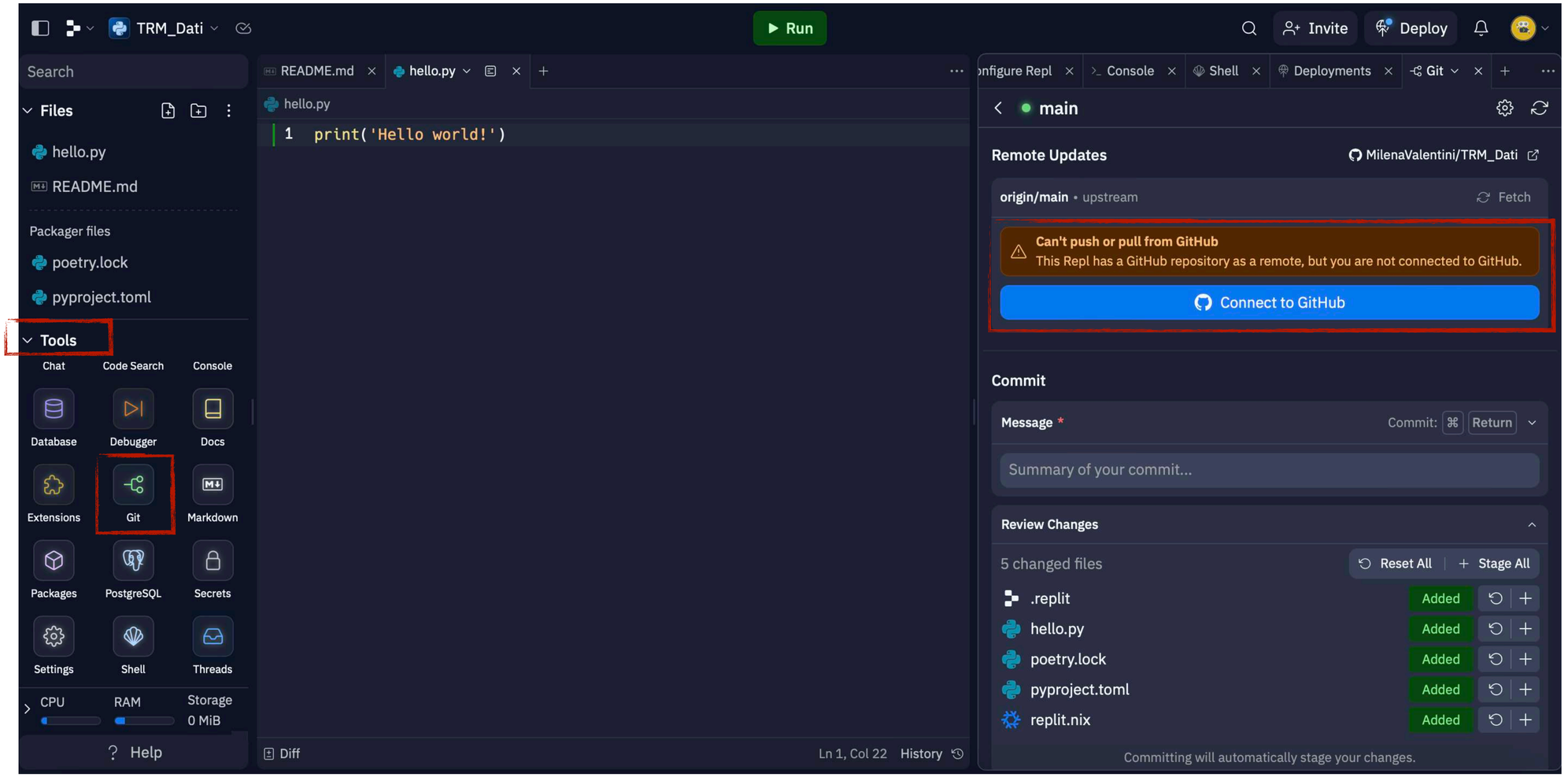
The Integrated Development Environment

Connecting replit to Git

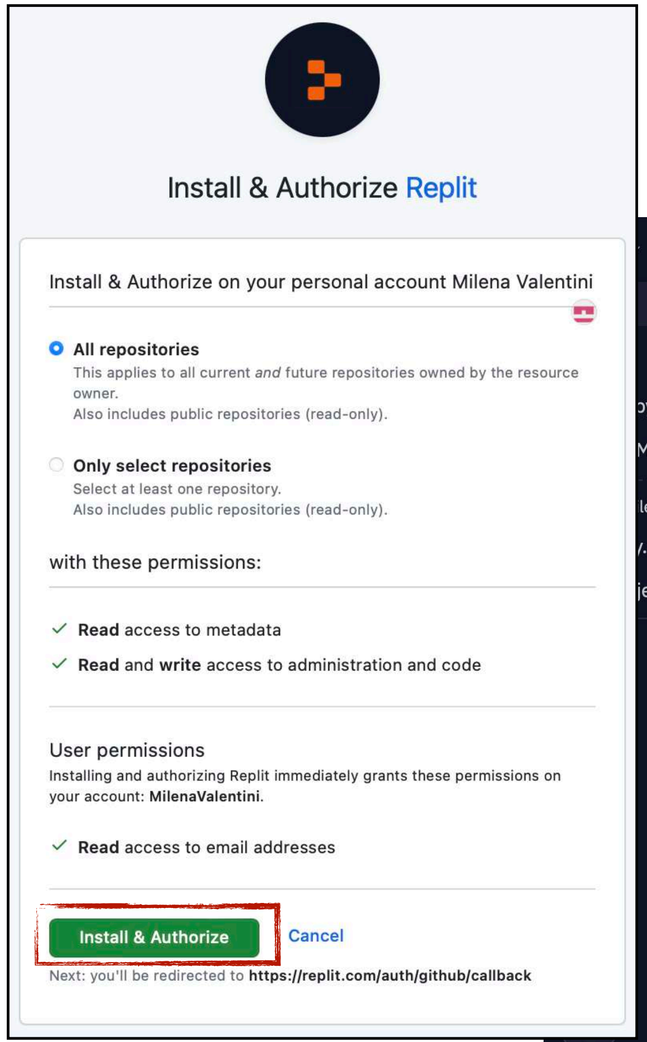


The Integrated Development Environment

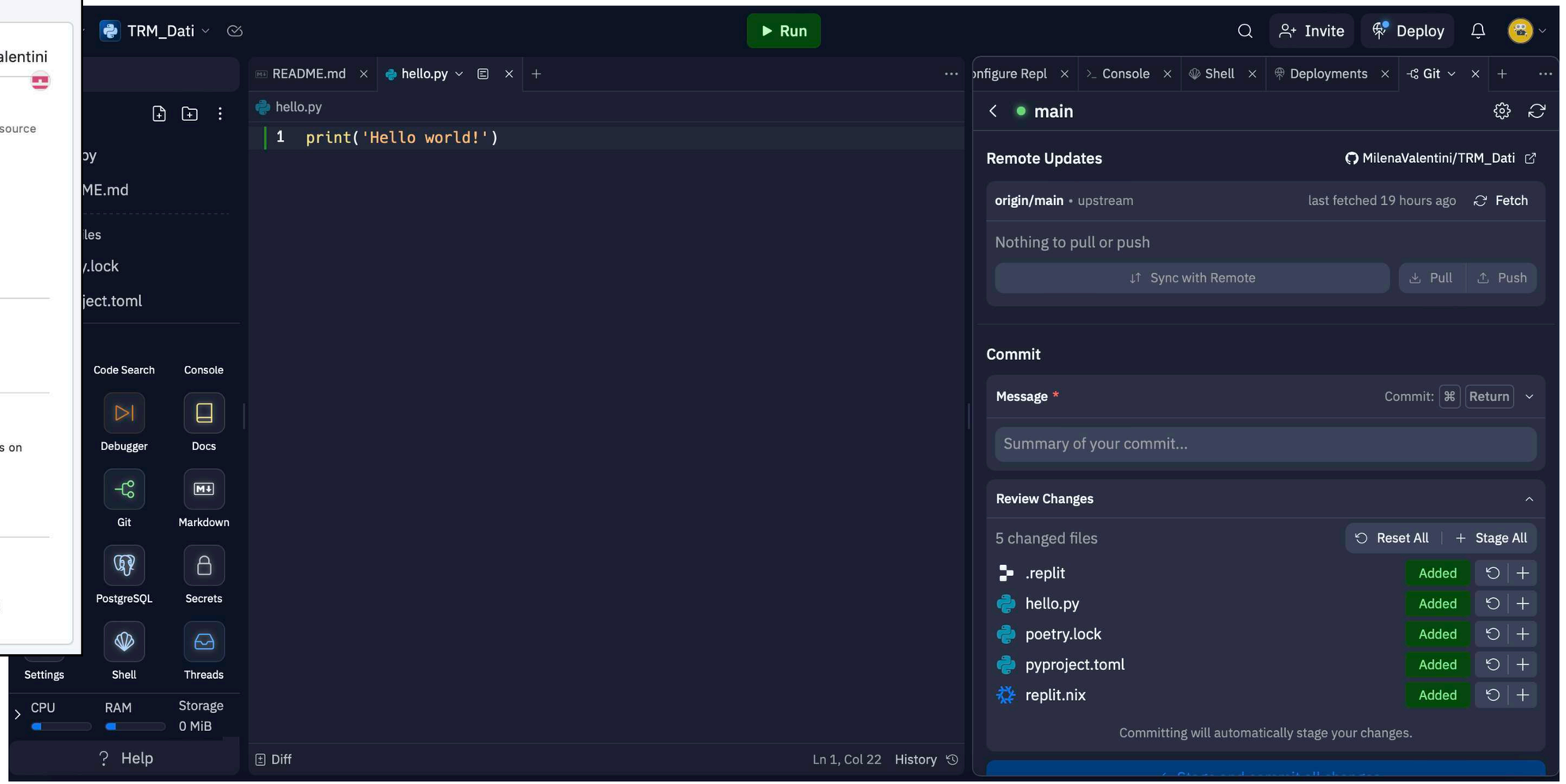
Connecting replit to Git



The Integrated Development Environment

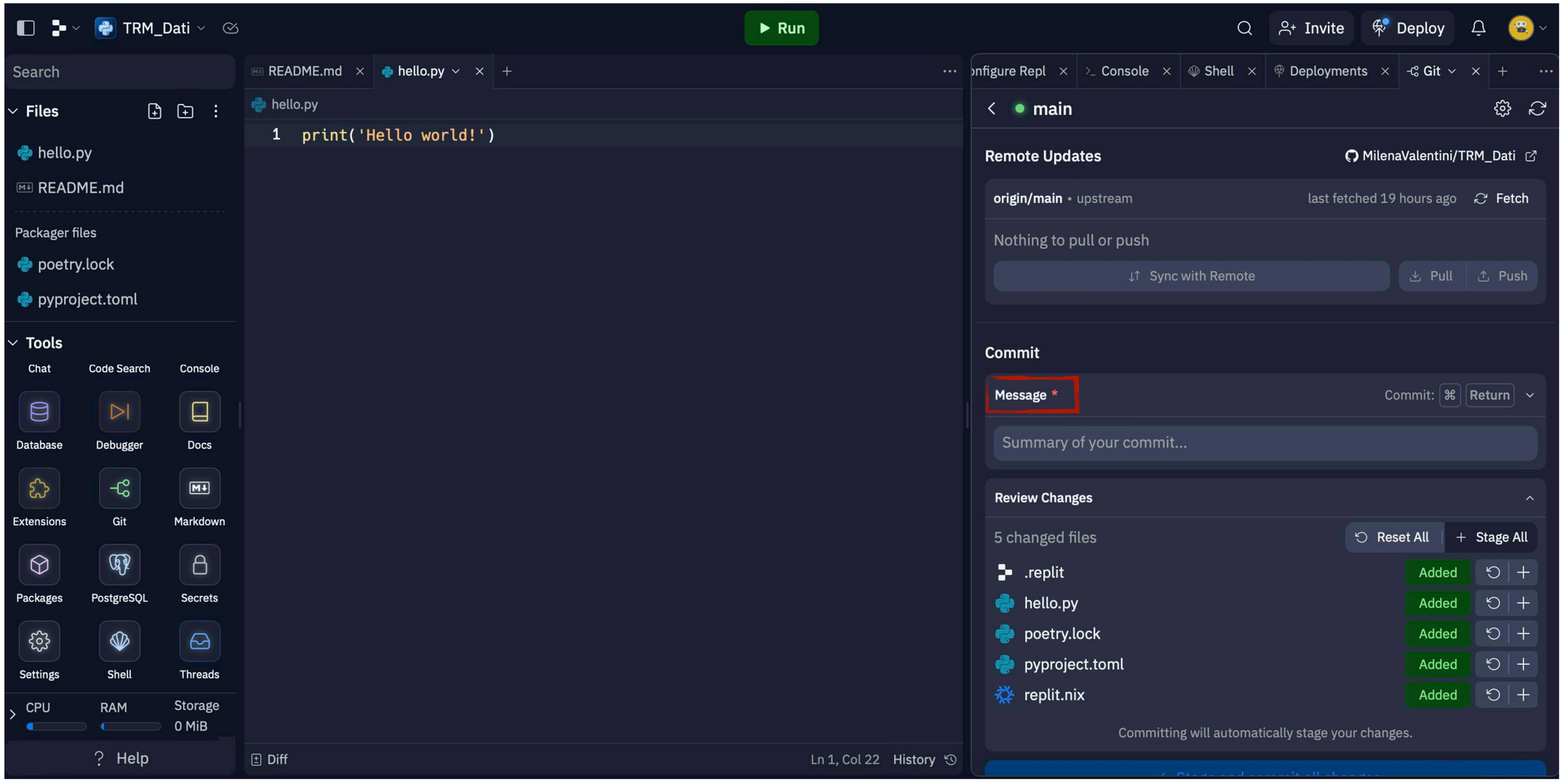


Authorize Replit on Git



The Integrated Development Environment

Add message to describe your commit

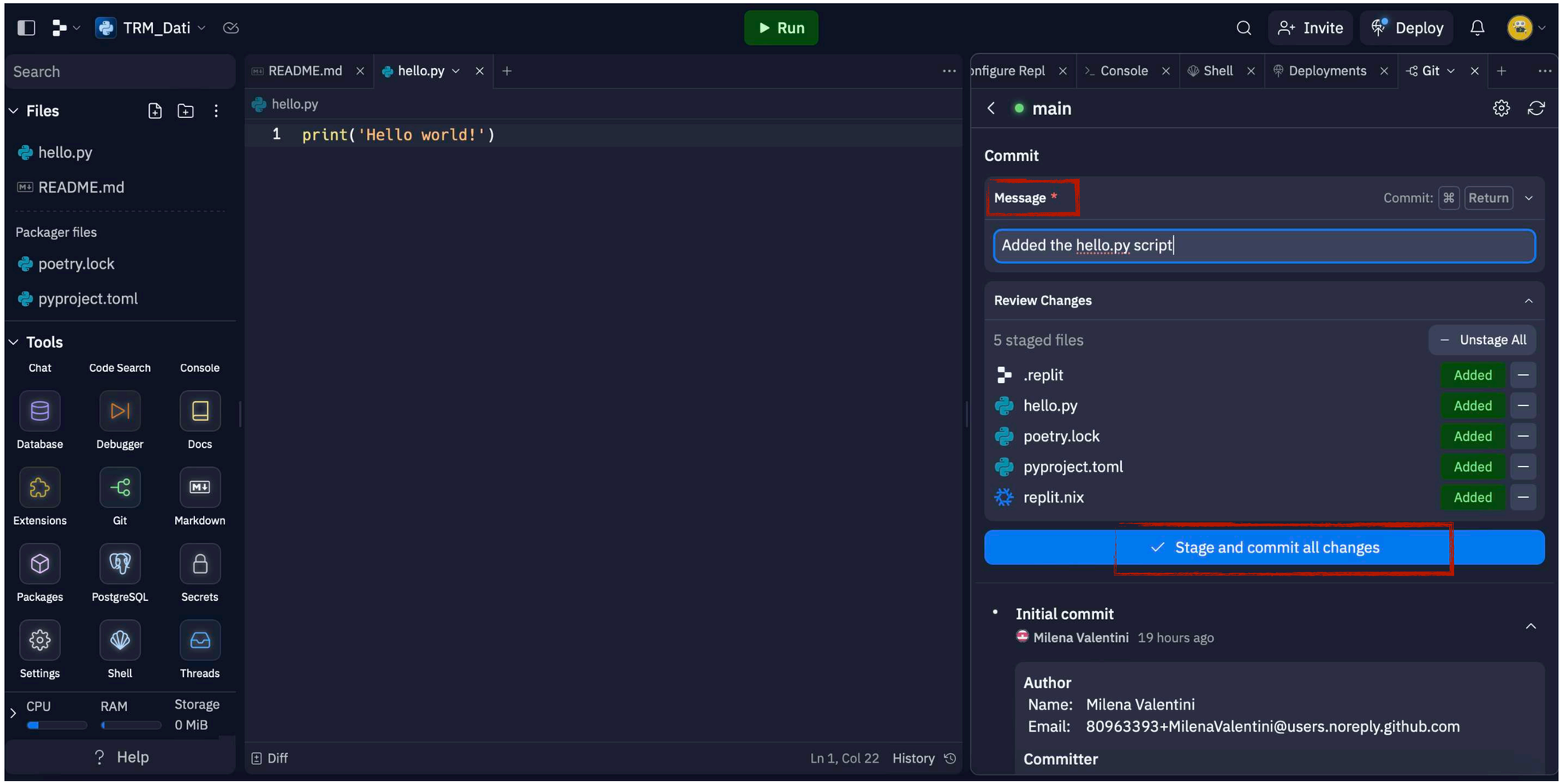


The Integrated Development Environment

Add message to describe your commit

Stage files to commit (the hello.py script, and additional files internal to replit plus related to libraries)

Commit



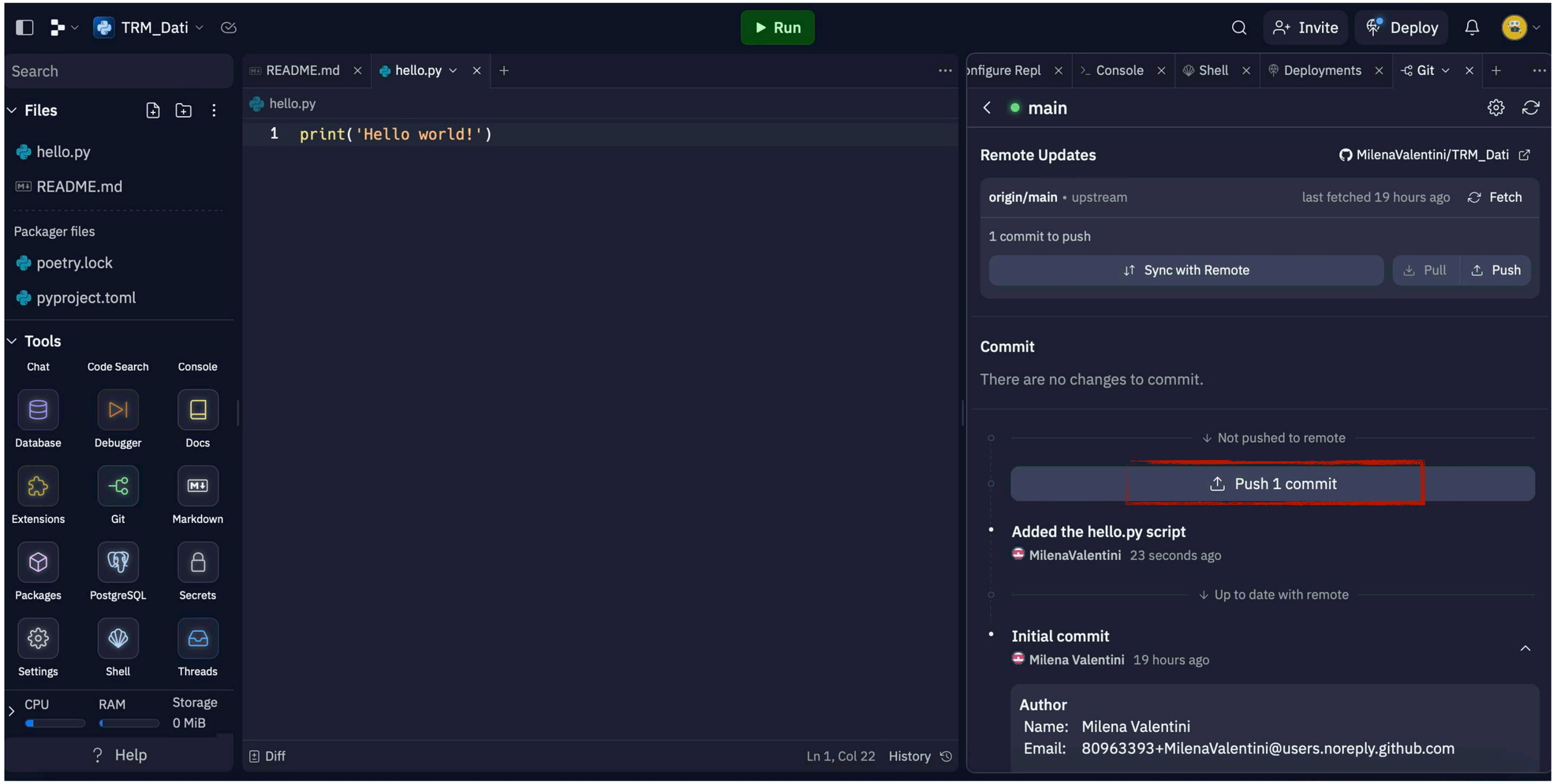
The Integrated Development Environment

Add message to describe your commit

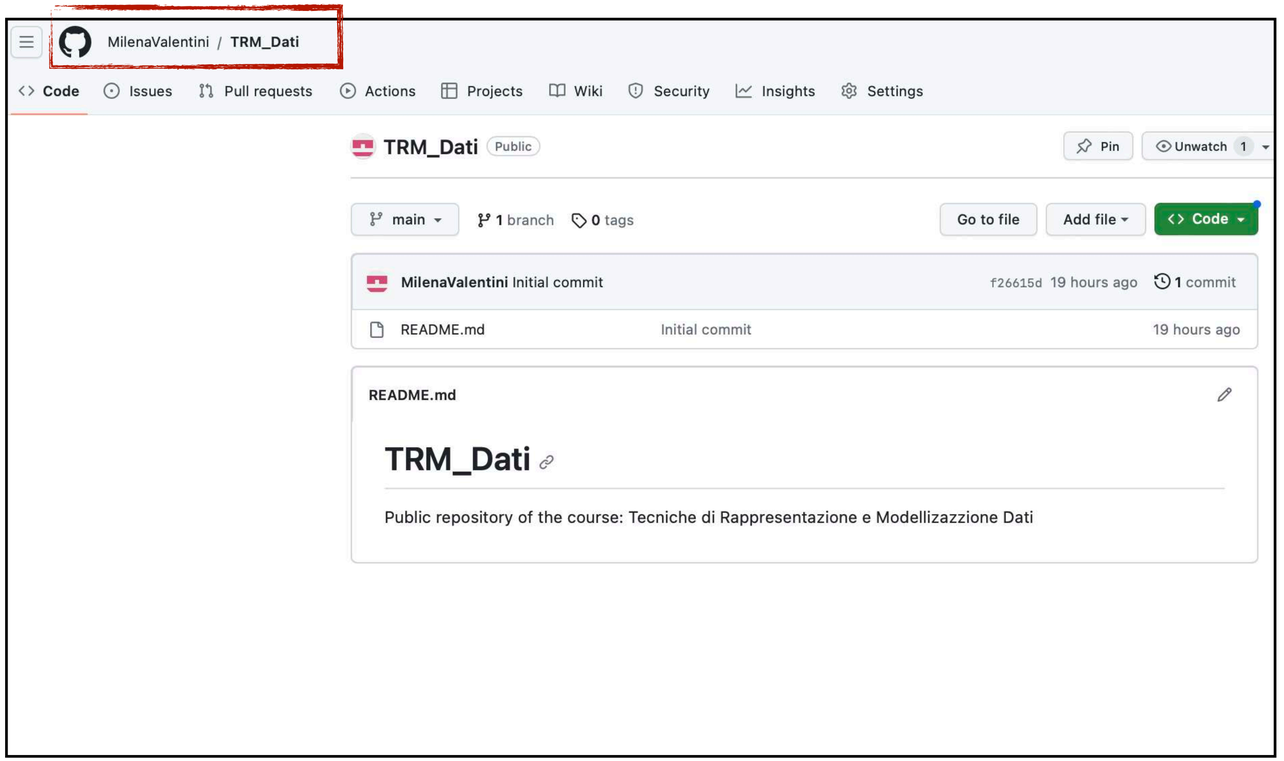
Stage files to commit (the hello.py script, and additional files internal to replit plus related to libraries)

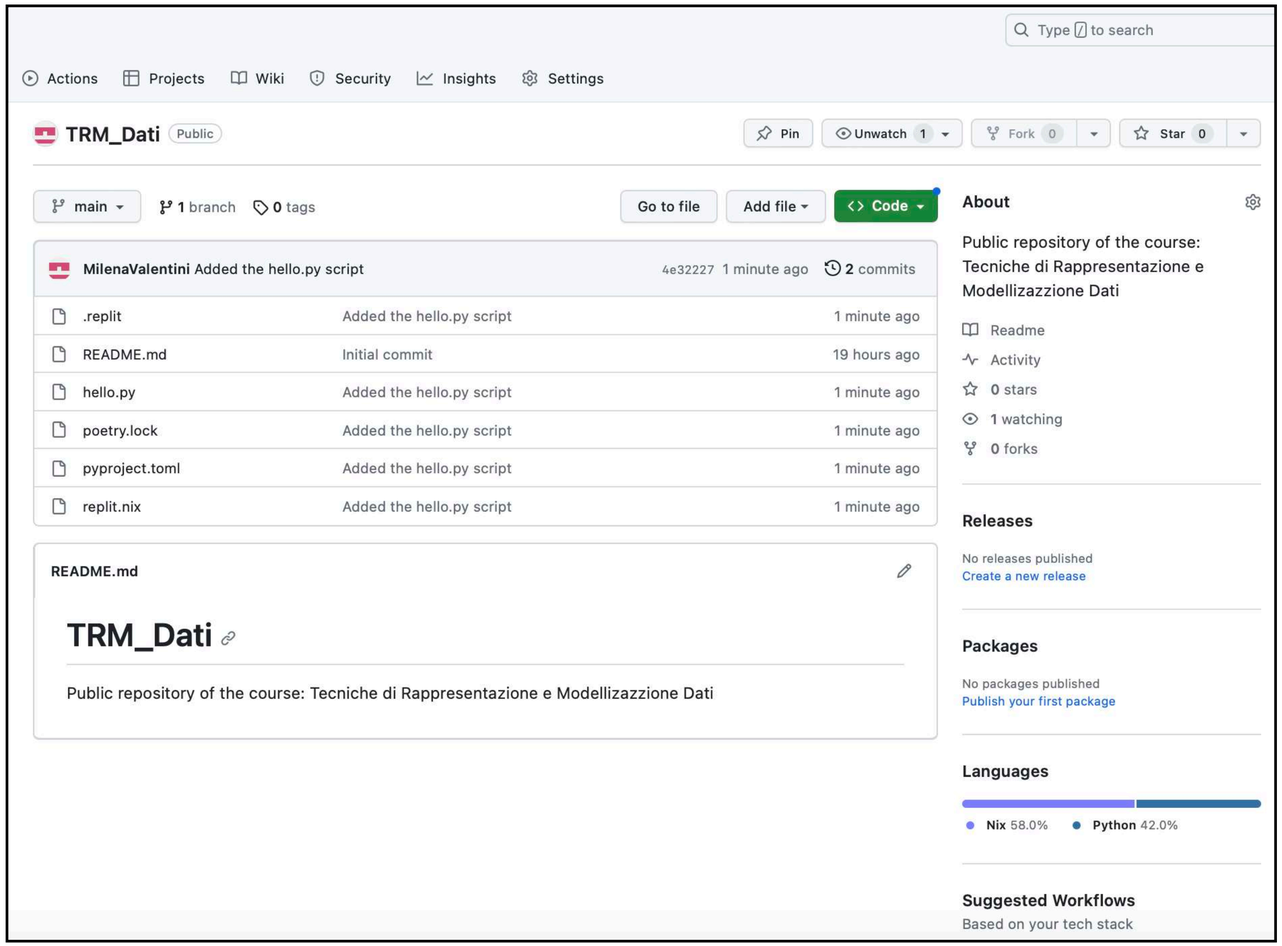
Commit

Push to Git



The Integrated Development Environment





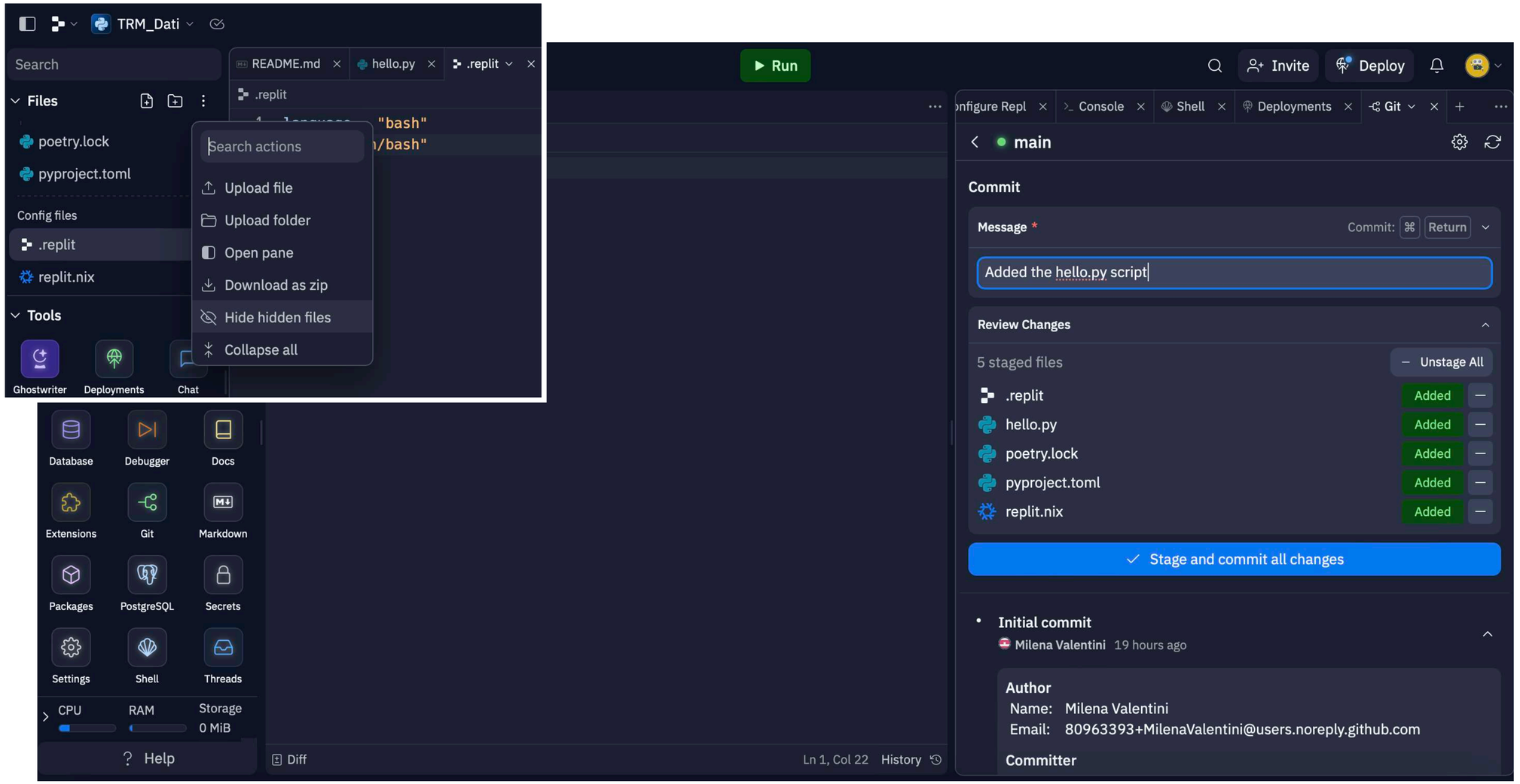
The Git repository has been successfully updated

The Integrated Development Environment

Make sure to configure
the hidden file .replit as follows:

language = “bash"
run = “/bin/bash”

Stage, commit
and push it to Git



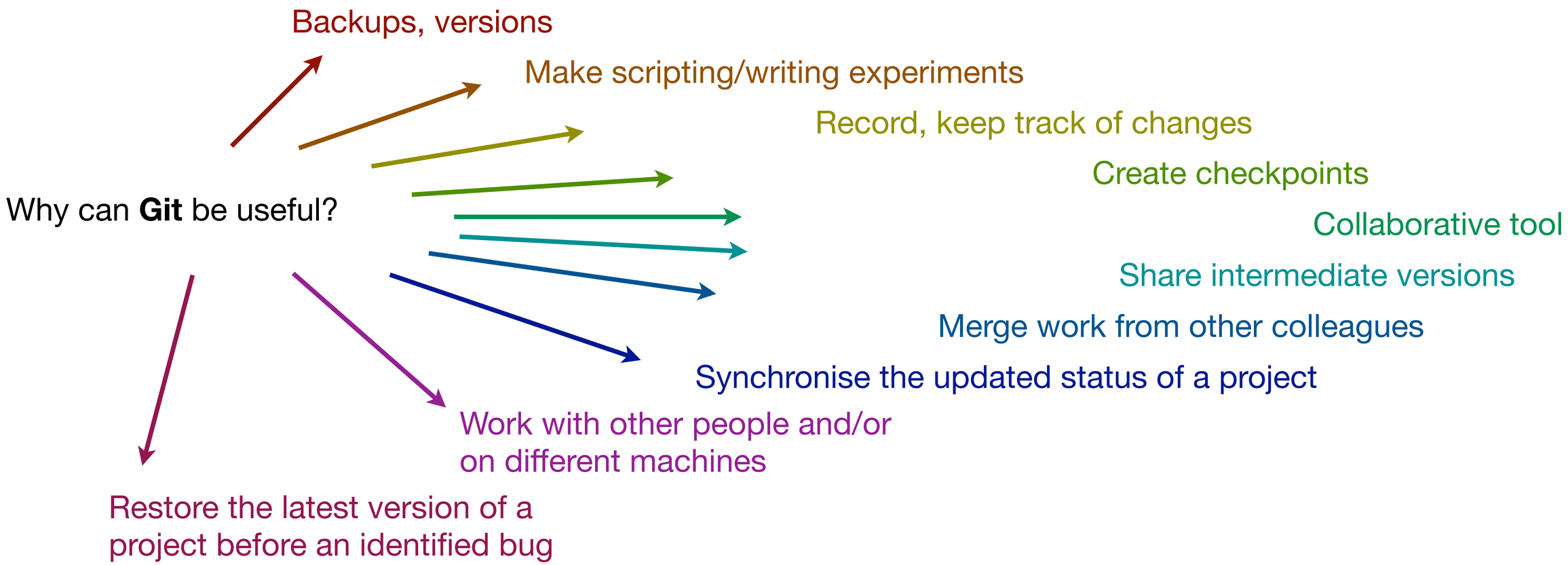
Git: Intro

Git is the most used version control system and it’s under continuous development

Git clients: https://git-scm.com/downloads/guis/

You can work from the terminal

Git: essentials and how-to

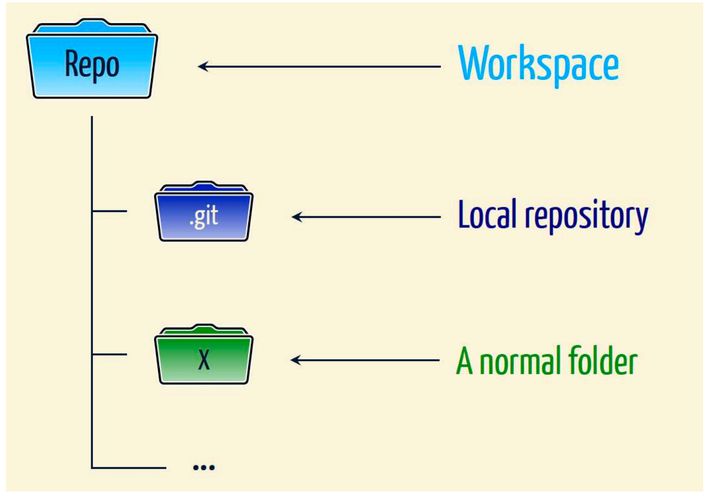


How would that be without Git?

Useful resources:
https://git-scm.com/book/en/v2
https://github.com/AxelKrypton/Git-crash-course

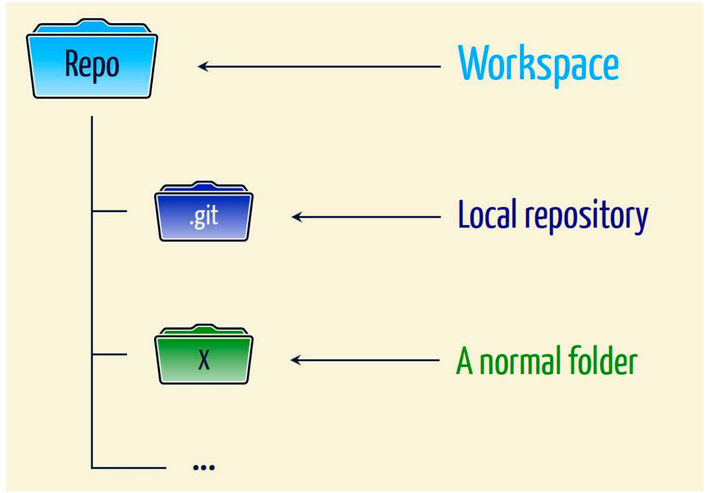
Git: Intro

A repository is a directory containing all the versions of the files

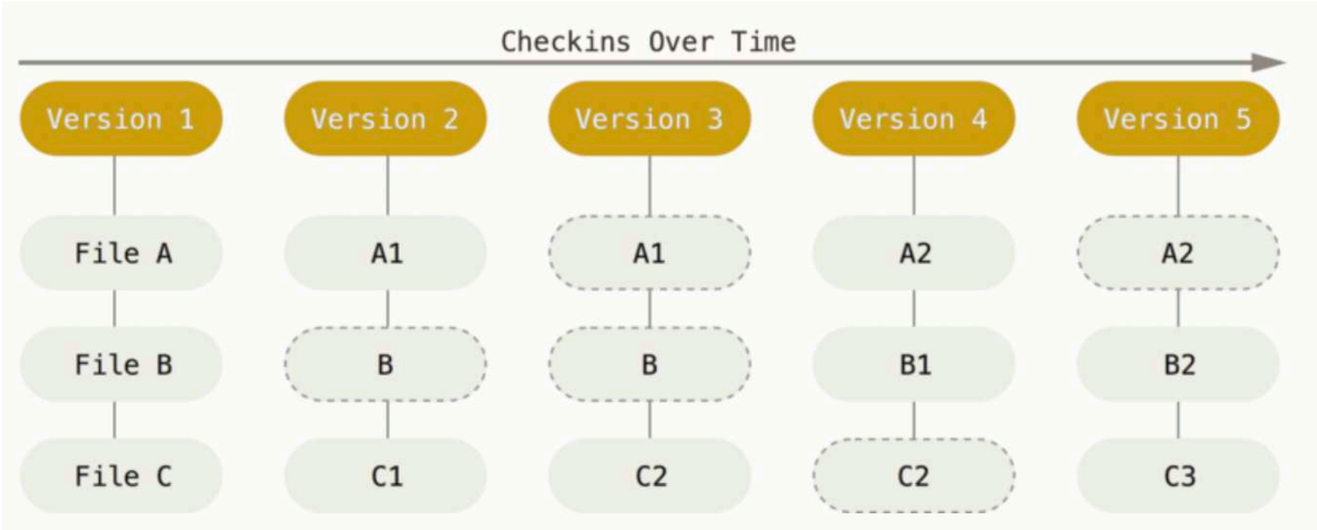


Git: Intro

A repository is a directory containing all the versions of the files



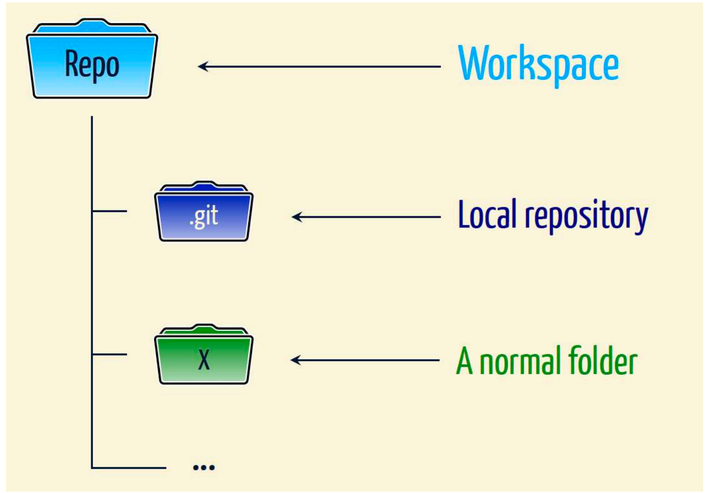
Records are based on snapshots taken over time



Snapshots are called commit and are referenced by a checksum

Git: Intro

A repository is a directory containing all the versions of the files



Git: essentials and how-to

```
(base) MacBook-Pro-2:TRM_Dati_Lezioni milenavalentini$ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone     Clone a repository into a new directory
   init      Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add       Add file contents to the index
   mv        Move or rename a file, a directory, or a symlink
   restore   Restore working tree files
   rm        Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect    Use binary search to find the commit that introduced a bug
   diff      Show changes between commits, commit and working tree, etc
   grep      Print lines matching a pattern
   log       Show commit logs
   show      Show various types of objects
   status    Show the working tree status

grow, mark and tweak your common history
   branch    List, create, or delete branches
   commit    Record changes to the repository
   merge     Join two or more development histories together
   rebase    Reapply commits on top of another base tip
```

apt-get update
apt-get install (--user) git

Git: essentials and how-to

```
(base) MacBook-Pro-2:~ milenavalentini$
(base) MacBook-Pro-2:~ milenavalentini$ cd Desktop/
(base) MacBook-Pro-2:Desktop milenavalentini$ mkdir Git_test_folder
(base) MacBook-Pro-2:Desktop milenavalentini$ cd Git_test_folder/
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ touch file_1.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ touch file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ touch file_3.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls
file_1.txt      file_2.txt      file_3.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls -la
total 0
drwxr-xr-x   5 milenavalentini  staff   160 Oct  1 22:00 .
drwx------@ 50 milenavalentini  staff  1600 Oct  1 21:59 ..
-rw-r--r--   1 milenavalentini  staff     0 Oct  1 21:59 file_1.txt
-rw-r--r--   1 milenavalentini  staff     0 Oct  1 21:59 file_2.txt
-rw-r--r--   1 milenavalentini  staff     0 Oct  1 22:00 file_3.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ cd ..
(base) MacBook-Pro-2:Desktop milenavalentini$ git init Git_test_folder/
Initialized empty Git repository in /Users/milenavalentini/Desktop/Git_test_folder/.git/
(base) MacBook-Pro-2:Desktop milenavalentini$
(base) MacBook-Pro-2:Desktop milenavalentini$ cd Git_test_folder/
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls -la
total 0
drwxr-xr-x   6 milenavalentini  staff   192 Oct  1 22:00 .
drwx------@ 50 milenavalentini  staff  1600 Oct  1 21:59 ..
drwxr-xr-x   9 milenavalentini  staff   288 Oct  1 22:00 .git
-rw-r--r--   1 milenavalentini  staff     0 Oct  1 21:59 file_1.txt
-rw-r--r--   1 milenavalentini  staff     0 Oct  1 21:59 file_2.txt
-rw-r--r--   1 milenavalentini  staff     0 Oct  1 22:00 file_3.txt
```

Git: essentials and how-to

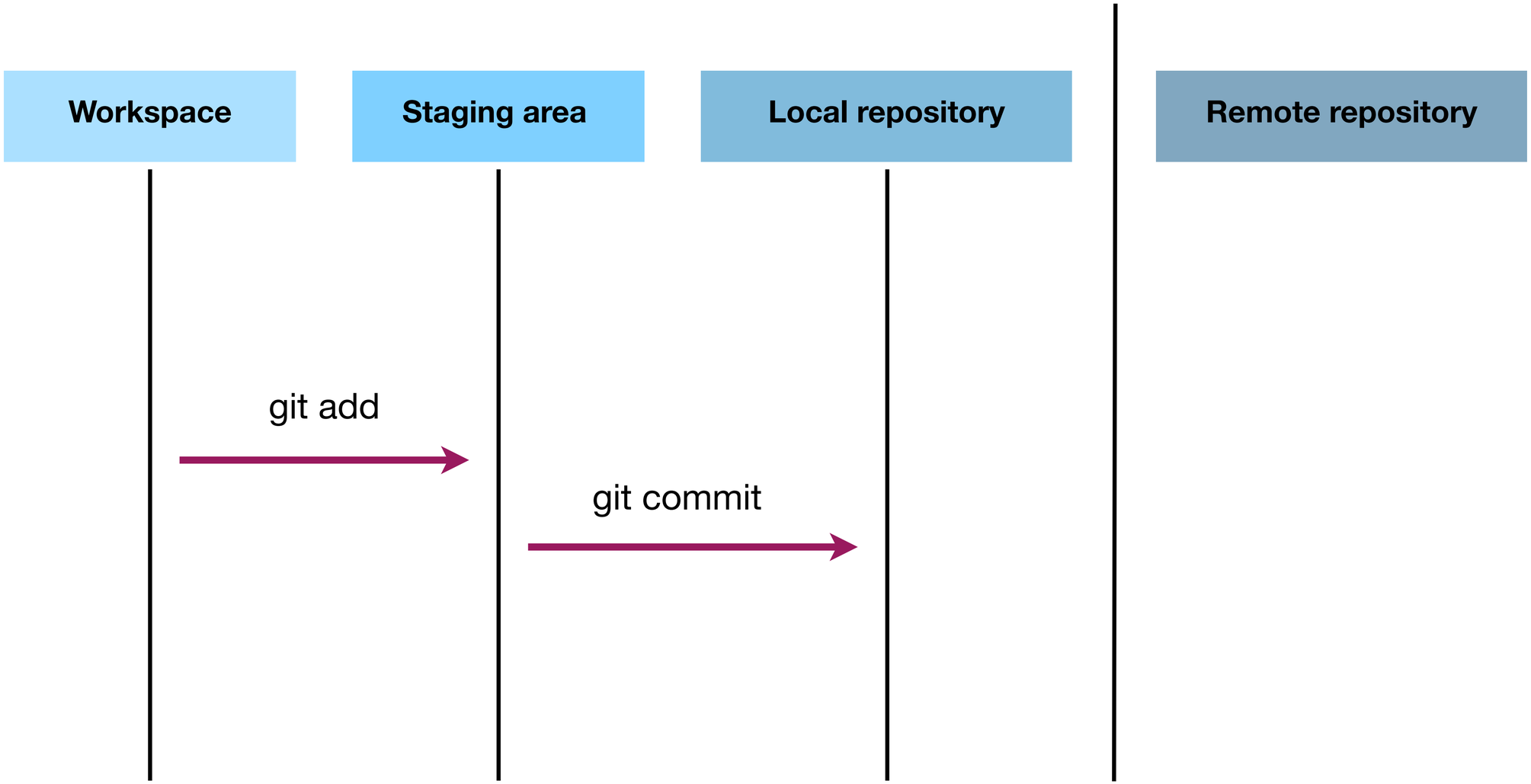
Initial configuration

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git config --global user.name "Milena Valentini"
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git config --global user.email valentini@usm.lmu.de
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

Git: essentials and how-to

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file_1.txt
        file_2.txt
        file_3.txt

nothing added to commit but untracked files present (use "git add" to track)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ cd ..
(base) MacBook-Pro-2:Desktop milenavalentini$ mkdir Git_test_folder_noInit
(base) MacBook-Pro-2:Desktop milenavalentini$ cd Git_test_folder_noInit/
(base) MacBook-Pro-2:Git_test_folder_noInit milenavalentini$ git status
fatal: not a git repository (or any of the parent directories): .git
(base) MacBook-Pro-2:Git_test_folder_noInit milenavalentini$
```

Git: essentials and how-to



List of Git commands: https://git-scm.com/docs

Git: essentials and how-to

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file_1.txt
        file_2.txt
        file_3.txt

nothing added to commit but untracked files present (use "git add" to track)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

git status displays
an overview of the workspace
and
the current snapshot (only)
of the staging area

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add .
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file_1.txt
        new file:   file_2.txt
        new file:   file_3.txt

(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

git add to “copy”
all the files in the folder
from the workspace
to the staging area
(otherwise, specify
the filename to be added)

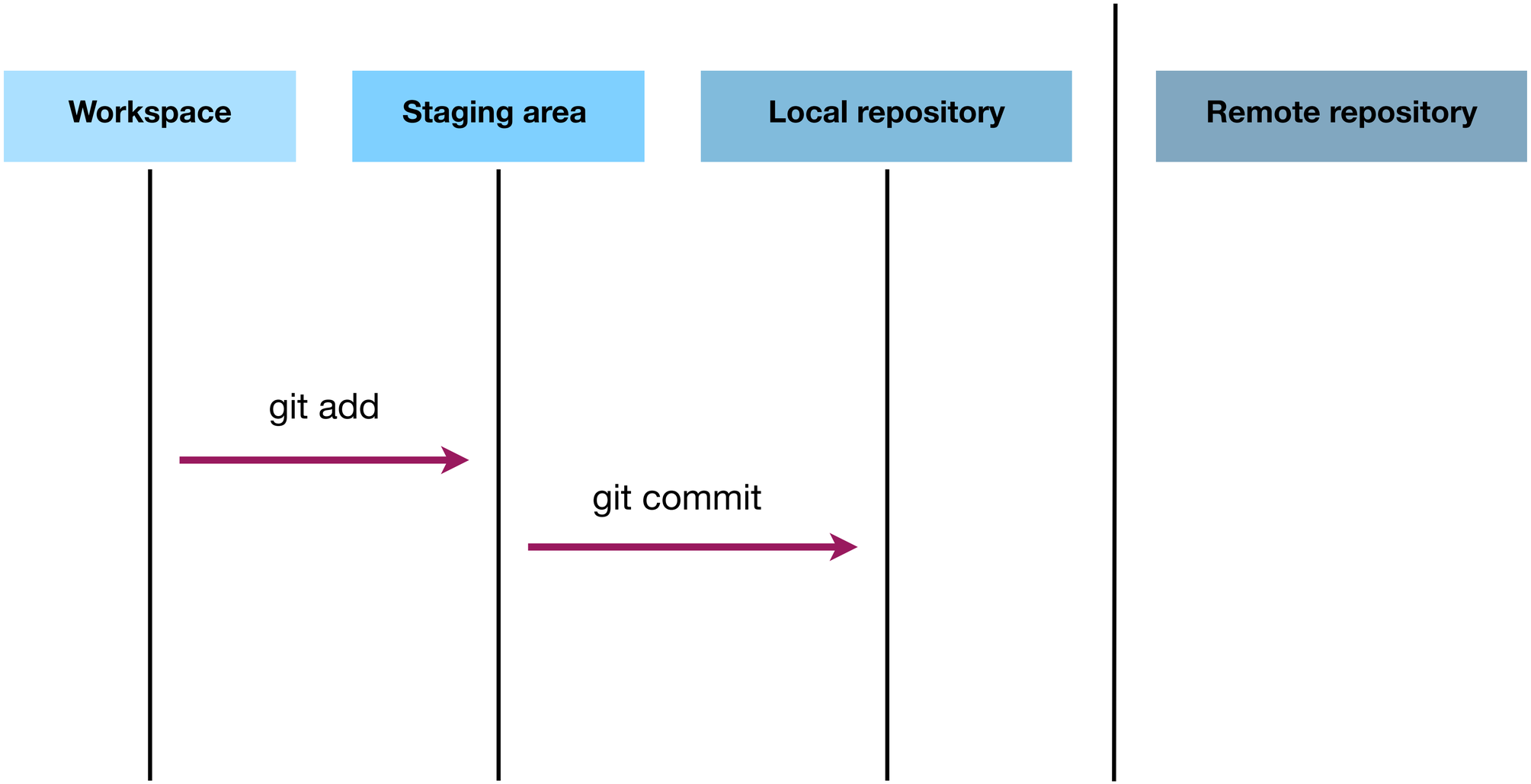
Git: essentials and how-to

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ touch file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls
file_1.txt      file_2.txt      file_3.txt      file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file_1.txt
        new file:   file_2.txt
        new file:   file_3.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file_4.txt

(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file_1.txt
        new file:   file_2.txt
        new file:   file_3.txt
        new file:   file_4.txt

(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

git status displays
an overview of the workspace
and
the current snapshot (only)
of the staging area

git add to “copy”
all the files in the folder
from the workspace
to the staging area
(otherwise, specify
the filename to be added)

Git: essentials and how-to



List of Git commands: https://git-scm.com/docs

Git: essentials and how-to

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'new files created'
[main (root-commit) 7d88be2] new files created
 4 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file_1.txt
 create mode 100644 file_2.txt
 create mode 100644 file_3.txt
 create mode 100644 file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'edited content of file_4.txt'
[main 544b8c4] edited content of file_4.txt
 1 file changed, 1 insertion(+)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'inserted text in file_2.txt'
[main a179a5d] inserted text in file_2.txt
 1 file changed, 1 insertion(+)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

git commit
copies files
from
the staging area
to
the local repository

Workflow:
edit
add
commit
(and repeat)

Git: essentials and how-to

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git log
commit a179a5d8af8d99fea993a1c5c485c68067132f9b (HEAD -> main)
Author: Milena Valentini <valentini@usm.lmu.de>
Date:   Sun Oct 1 23:20:48 2023 +0200

    inserted text in file_2.txt

commit 544b8c428fdc6ed8e280dd965c38b585a495bce1
Author: Milena Valentini <milenavalentini@MacBook-Pro-2.local>
Date:   Sun Oct 1 23:19:02 2023 +0200

    edited content of file_4.txt

commit 7d88be2f81fd4d8a2705d1b88d131e770d4e7fa2
Author: Milena Valentini <milenavalentini@MacBook-Pro-2.local>
Date:   Sun Oct 1 23:15:15 2023 +0200

    new files created
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

git log
to output
changes with time
(most recent files on top)

Git: essentials and how-to

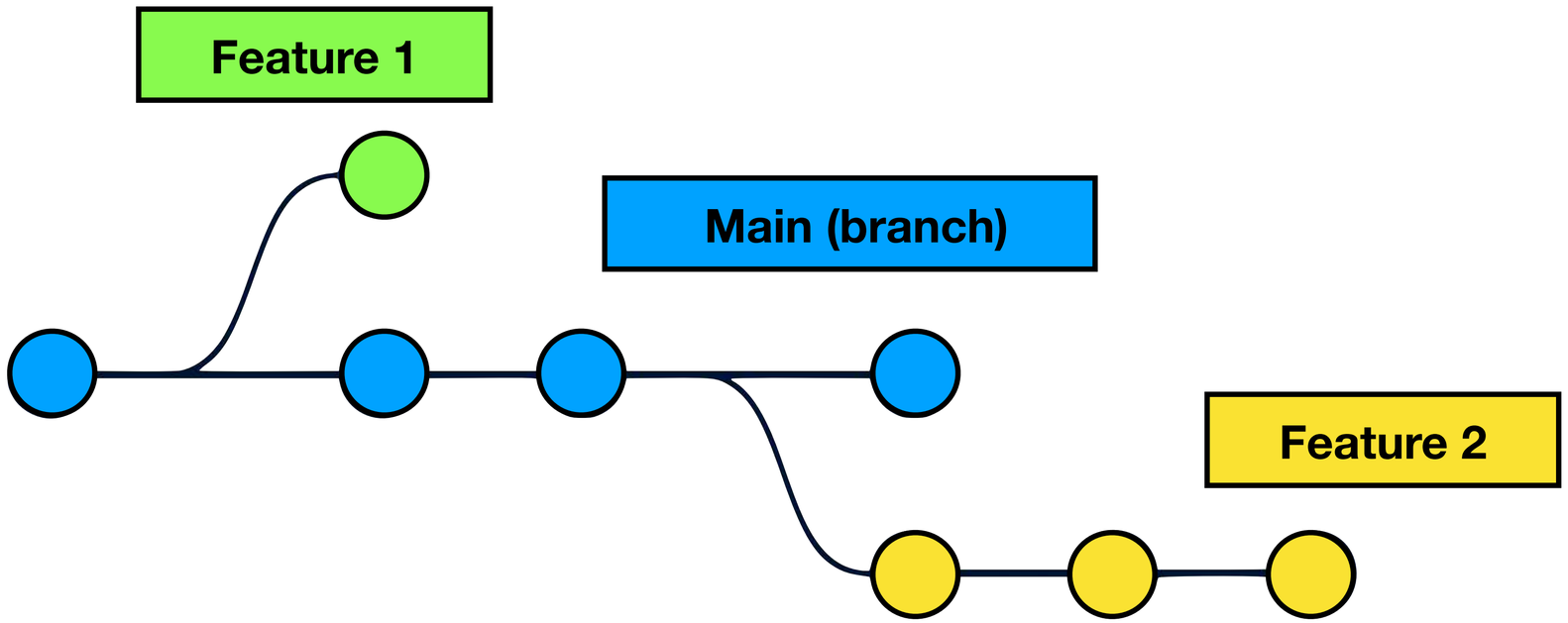
```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git log
commit a179a5d8af8d99fea993a1c5c485c68067132f9b (HEAD -> main)
Author: Milena Valentini <valentini@usm.lmu.de>
Date:   Sun Oct 1 23:20:48 2023 +0200

    inserted text in file_2.txt

commit 544b8c428fdc6ed8e280dd965c38b585a495bce1
Author: Milena Valentini <milenavalentini@MacBook-Pro-2.local>
Date:   Sun Oct 1 23:19:02 2023 +0200

    edited content of file_4.txt

commit 7d88be2f81fd4d8a2705d1b88d131e770d4e7fa2
Author: Milena Valentini <milenavalentini@MacBook-Pro-2.local>
Date:   Sun Oct 1 23:15:15 2023 +0200

    new files created
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

git log
to output
changes with time
(most recent files on top)

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git status
On branch main
nothing to commit, working tree clean
```

git status

Git: branches



Branches store different versions of your project

They are pointers to a commit

The main (or master) branch always exists and it is created during the initialization

Key for parallel development

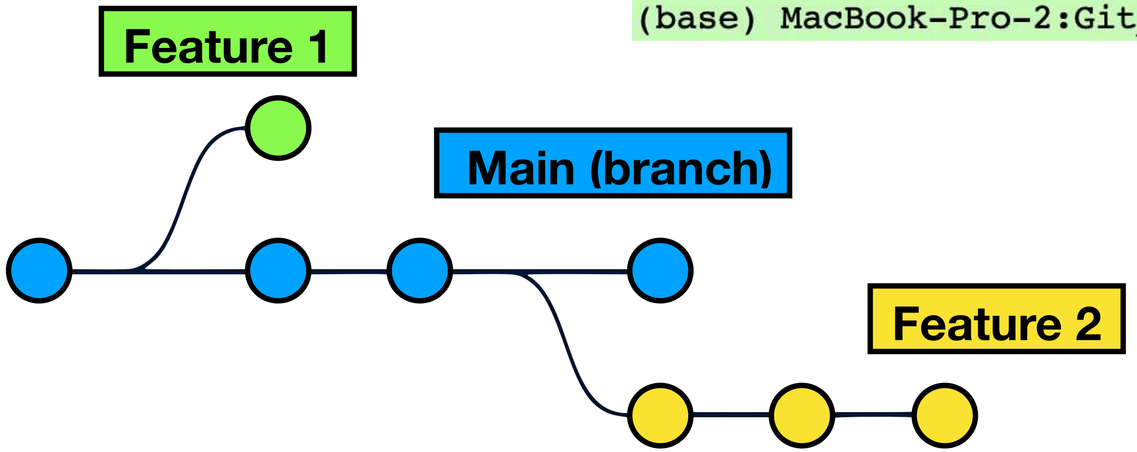
Git: branches

List all existing local branches

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch feature_A
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch feature_B
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
  feature_B
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch -d feature_B
Deleted branch feature_B (was a179a5d).
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

Create a new branch

Delete a branch

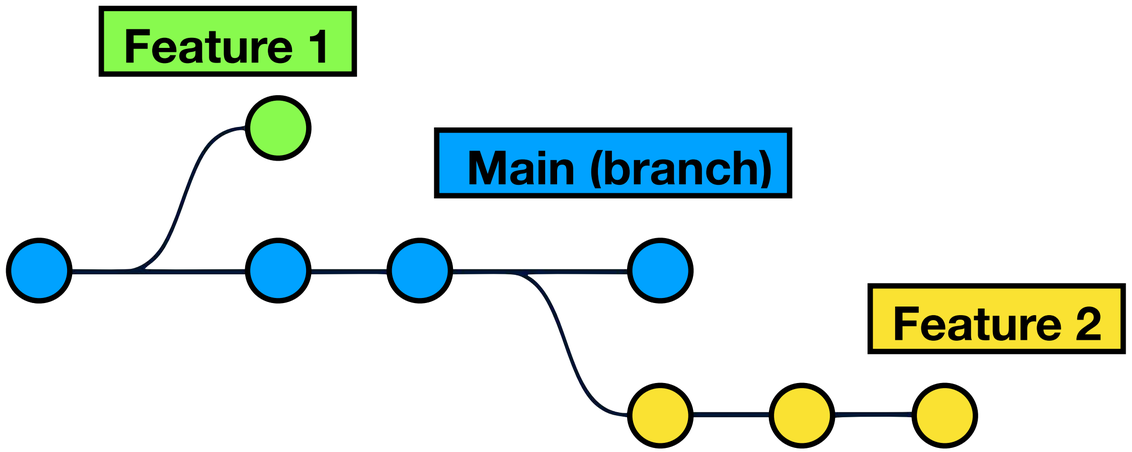


Git: branches

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch feature_A
Switched to branch 'feature_A'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
* feature_A
  main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ _
```

Switch to another branch

git branch operates on the local repository

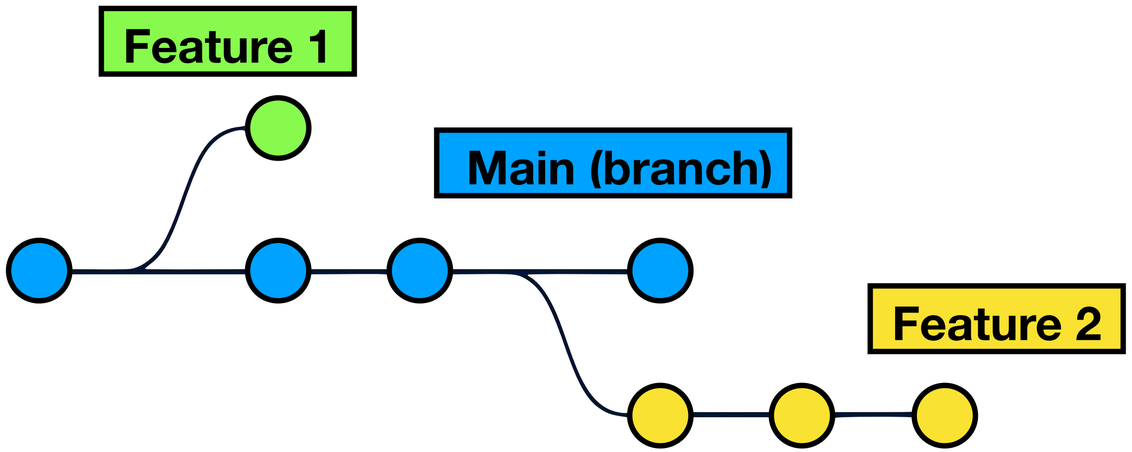


Git: branches

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch feature_A
Switched to branch 'feature_A'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls
file_1.txt      file_2.txt      file_3.txt      file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified the content of file_2.txt'
[feature_A a7d509c] modified the content of file_2.txt
 1 file changed, 1 insertion(+)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ _
```

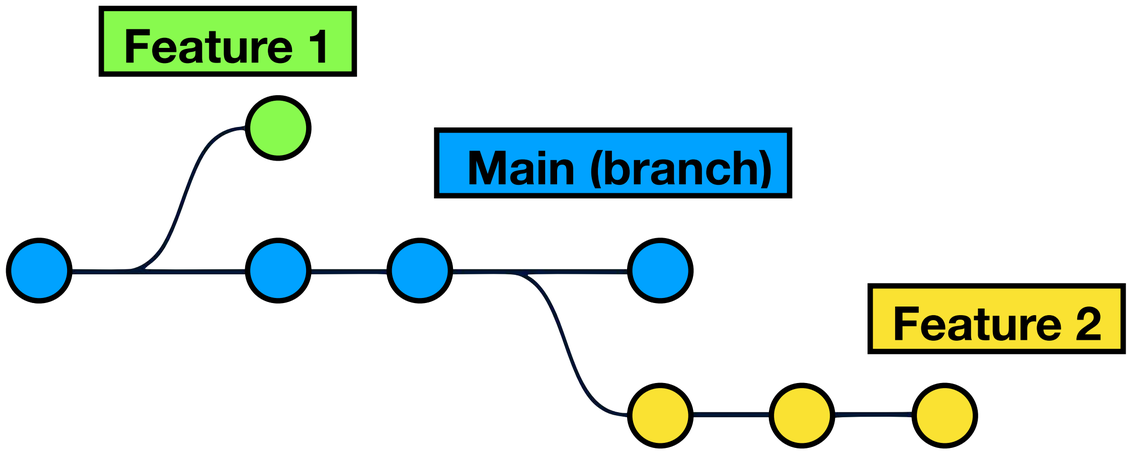
Edit a file in a branch

commit to that branch



Git: branches

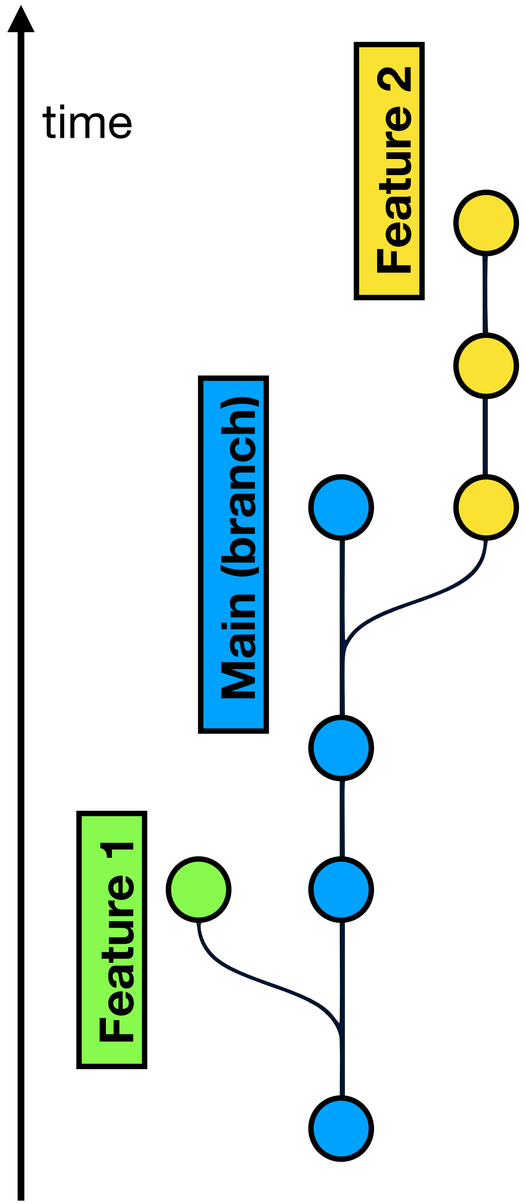
```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
* feature_A
  main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch main
Switched to branch 'main'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git merge feature_A
Updating 331b46f..a7d509c
Fast-forward
 file_2.txt | 1 +
 1 file changed, 1 insertion(+)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```



Copies the content
of the specified branch (i.e. feature_A)
in the current branch (i.e. main)

Crucial to understand in which
branch I am before making a merge

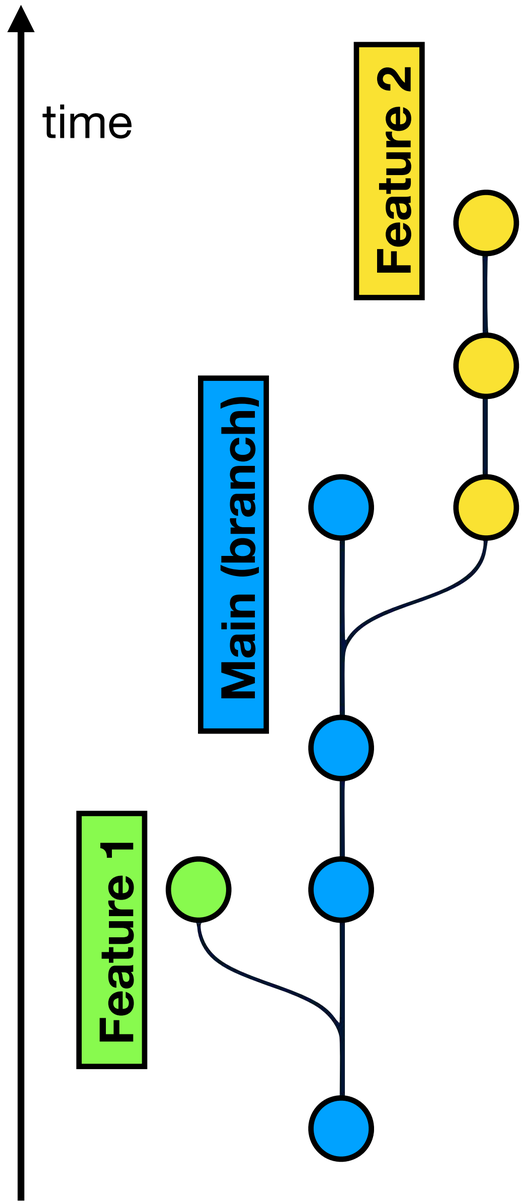
Git: branches




```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch feature_C
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
  feature_C
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch feature_C
Switched to branch 'feature_C'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
* feature_C
  main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls
file_1.txt      file_2.txt      file_3.txt      file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on feature_C'
[feature_C a96367a] modified file_2 on feature_C
 1 file changed, 1 insertion(+)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch main
Switched to branch 'main'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on main'
[main 6cd7523] modified file_2 on main
 1 file changed, 1 deletion(-)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
  feature_C
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git merge feature_C
Auto-merging file_2.txt
CONFLICT (content): Merge conflict in file_2.txt
Automatic merge failed; fix conflicts and then commit the result.
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on main to resolve conflicts
with file_2 on feature_C'
[main 2a2d1ca] modified file_2 on main to resolve conflicts with file_2 on feature_C
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

Edit file_2 in feature_C

Git: branches

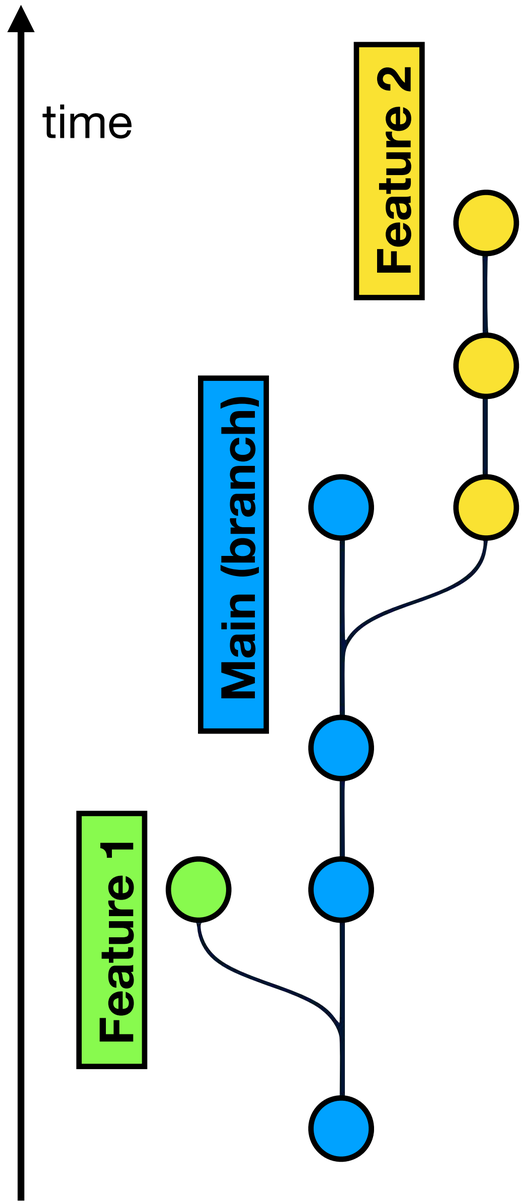



```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch feature_C
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
  feature_C
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch feature_C
Switched to branch 'feature_C'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
* feature_C
  main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls
file_1.txt      file_2.txt      file_3.txt      file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on feature_C'
[feature_C a96367a] modified file_2 on feature_C
 1 file changed, 1 insertion(+)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch main
Switched to branch 'main'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on main'
[main 6cd7523] modified file_2 on main
 1 file changed, 1 deletion(-)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
  feature_C
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git merge feature_C
Auto-merging file_2.txt
CONFLICT (content): Merge conflict in file_2.txt
Automatic merge failed; fix conflicts and then commit the result.
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on main to resolve conflicts
with file_2 on feature_C'
[main 2a2d1ca] modified file_2 on main to resolve conflicts with file_2 on feature_C
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

Edit file_2 in feature_C

Edit the same in main

Git: branches



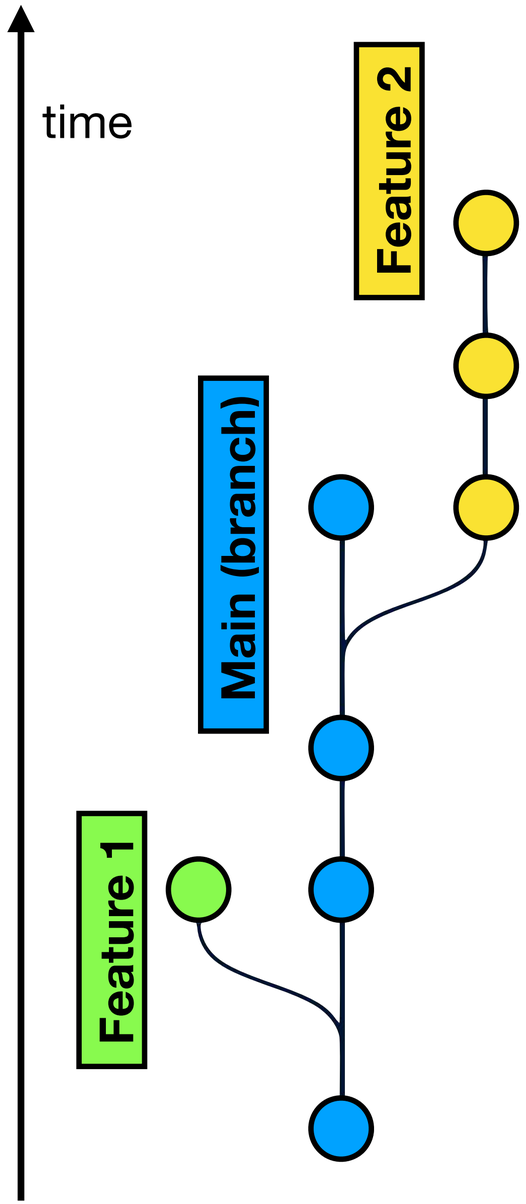

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch feature_C
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
  feature_C
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch feature_C
Switched to branch 'feature_C'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
* feature_C
  main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ ls
file_1.txt      file_2.txt      file_3.txt      file_4.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on feature_C'
[feature_C a96367a] modified file_2 on feature_C
 1 file changed, 1 insertion(+)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git switch main
Switched to branch 'main'
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on main'
[main 6cd7523] modified file_2 on main
 1 file changed, 1 deletion(-)
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch
  feature_A
  feature_C
* main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git merge feature_C
Auto-merging file_2.txt
CONFLICT (content): Merge conflict in file_2.txt
Automatic merge failed; fix conflicts and then commit the result.
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ vi file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git add file_2.txt
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git commit -m 'modified file_2 on main to resolve conflicts
with file_2 on feature_C'
[main 2a2d1ca] modified file_2 on main to resolve conflicts with file_2 on feature_C
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

Edit file_2 in feature_C

Edit the same in main

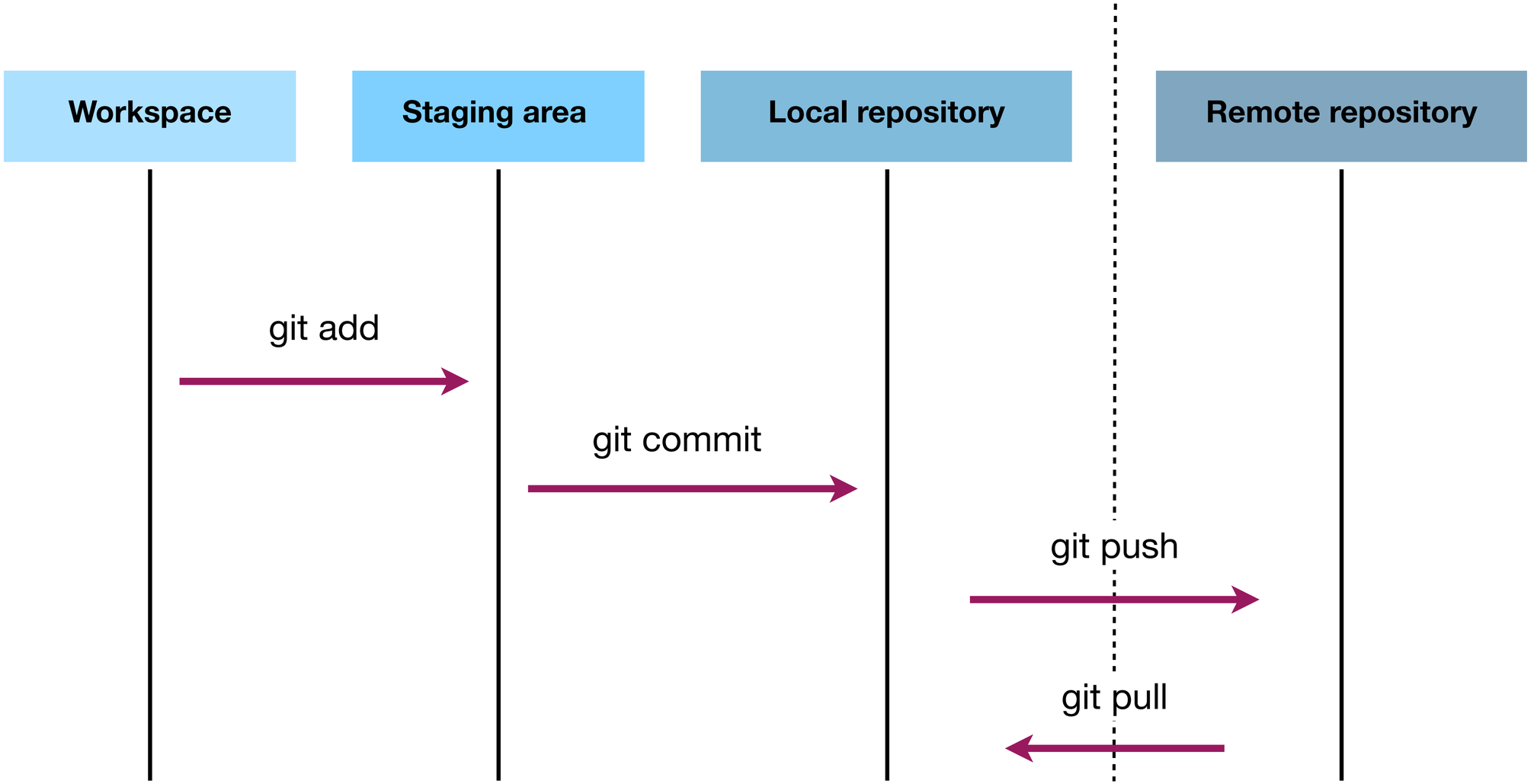
```
42
84
<<<<<<< HEAD
=======
168
336
>>>>>>> feature_C
```

Git: branches

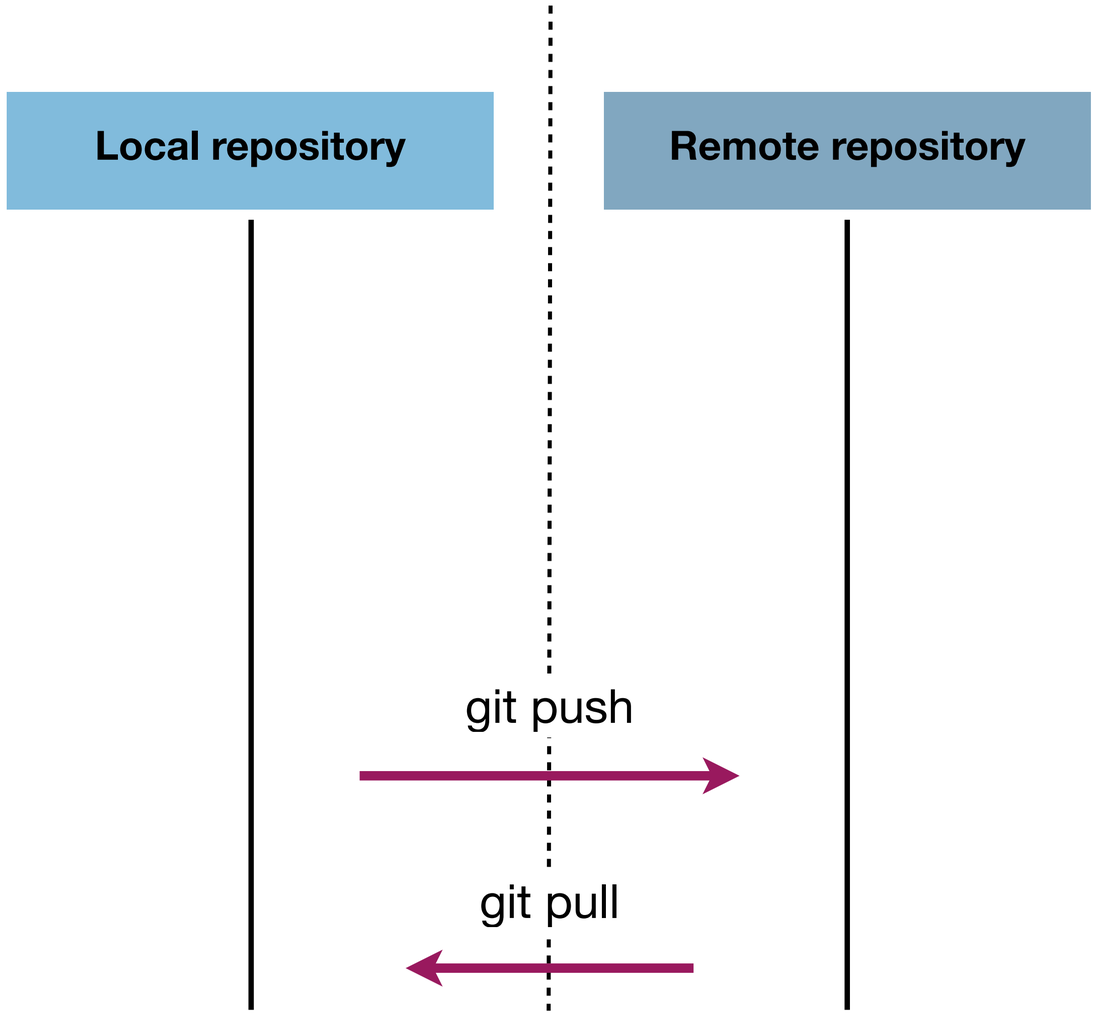



```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git log
commit 2a2d1ca55771afba839a9e544f6f7286760f470a (HEAD -> main)
Merge: 6cd7523 a96367a
Author: Milena Valentini <valentini@usm.lmu.de>
Date:   Mon Oct 2 02:39:50 2023 +0200

    modified file_2 on main to resolve conflicts with file_2 on feature_C

commit 6cd7523db0a8c997f01641acc8057cb76d97cc3d
Author: Milena Valentini <valentini@usm.lmu.de>
Date:   Mon Oct 2 02:37:48 2023 +0200

    modified file_2 on main

commit a96367ace141b3030cea96da6cd1fba293e65e4c (feature_C)
Author: Milena Valentini <valentini@usm.lmu.de>
Date:   Mon Oct 2 02:37:10 2023 +0200

    modified file_2 on feature_C

commit de0da59bea5d023e78f6fcdf59273ce982f36677
Author: Milena Valentini <valentini@usm.lmu.de>
Date:   Mon Oct 2 02:26:19 2023 +0200

    modified file_4.txt by adding a second line

commit a7d509c24a356ebc3fe5996d2f8745843bb8b60a (feature_A)
Author: Milena Valentini <valentini@usm.lmu.de>
Date:   Mon Oct 2 02:15:28 2023 +0200

    modified the content of file_2.txt
```

Git: from local to remote repository

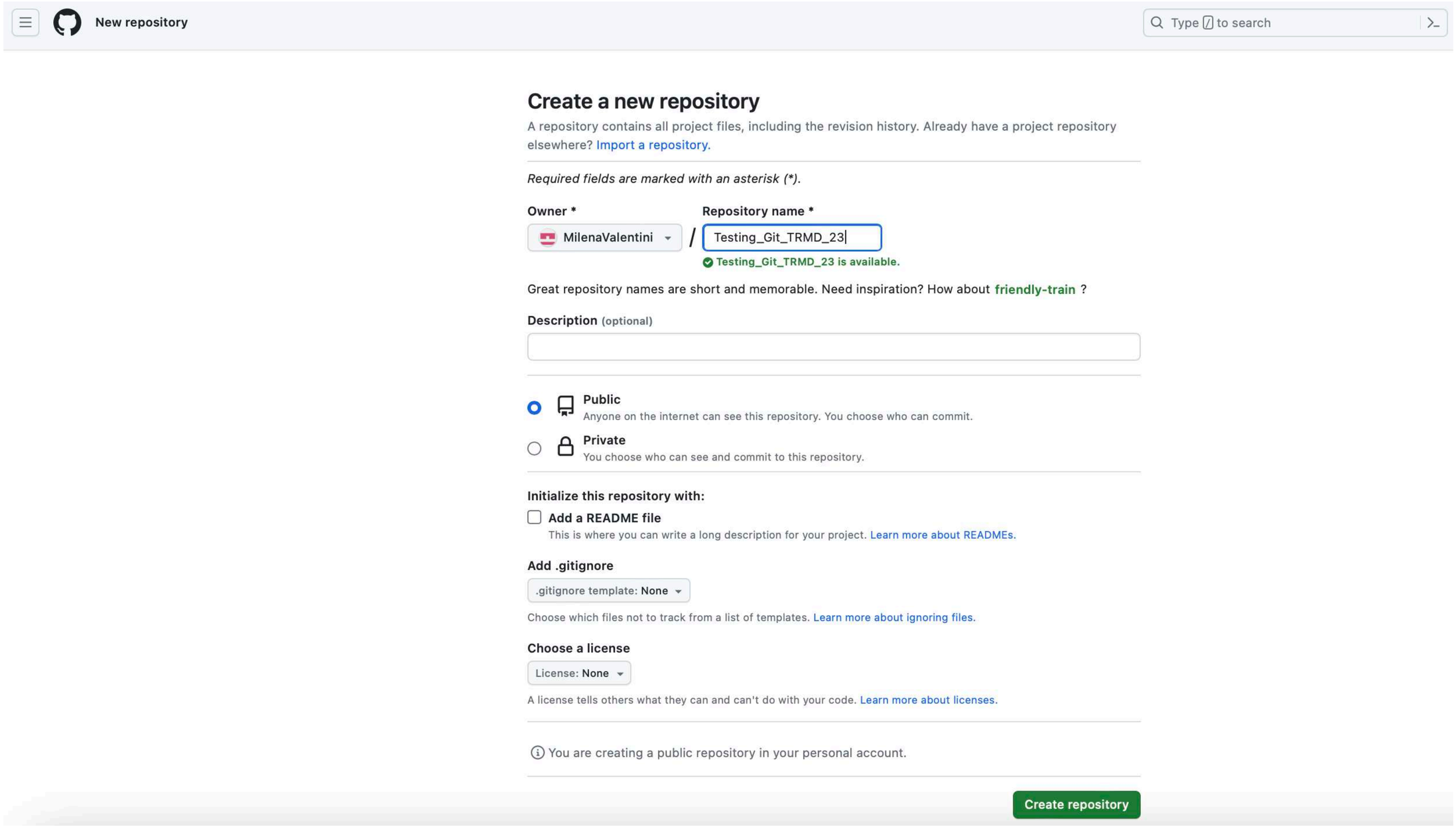


Git: from local to remote repository

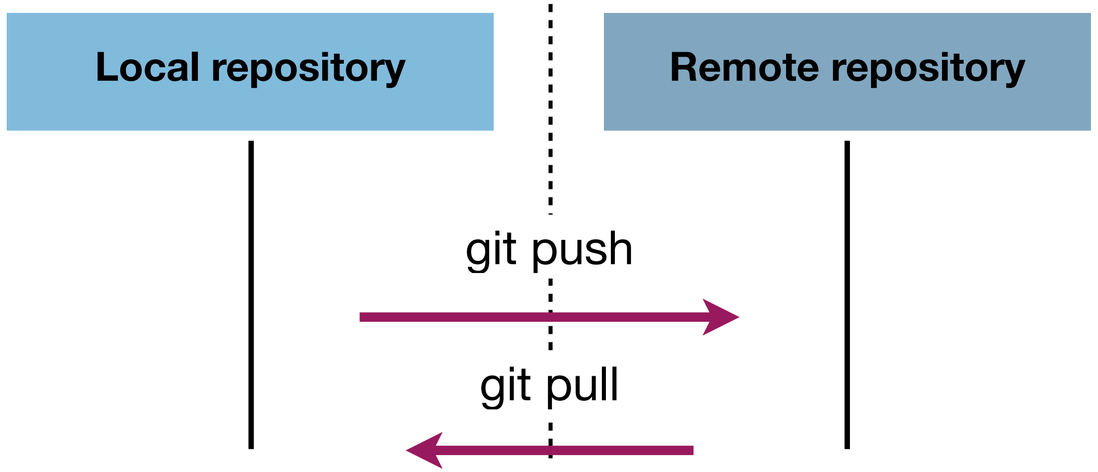


Create a new, empty repository on Git

Git: from local to remote repository



Git: from local to remote repository

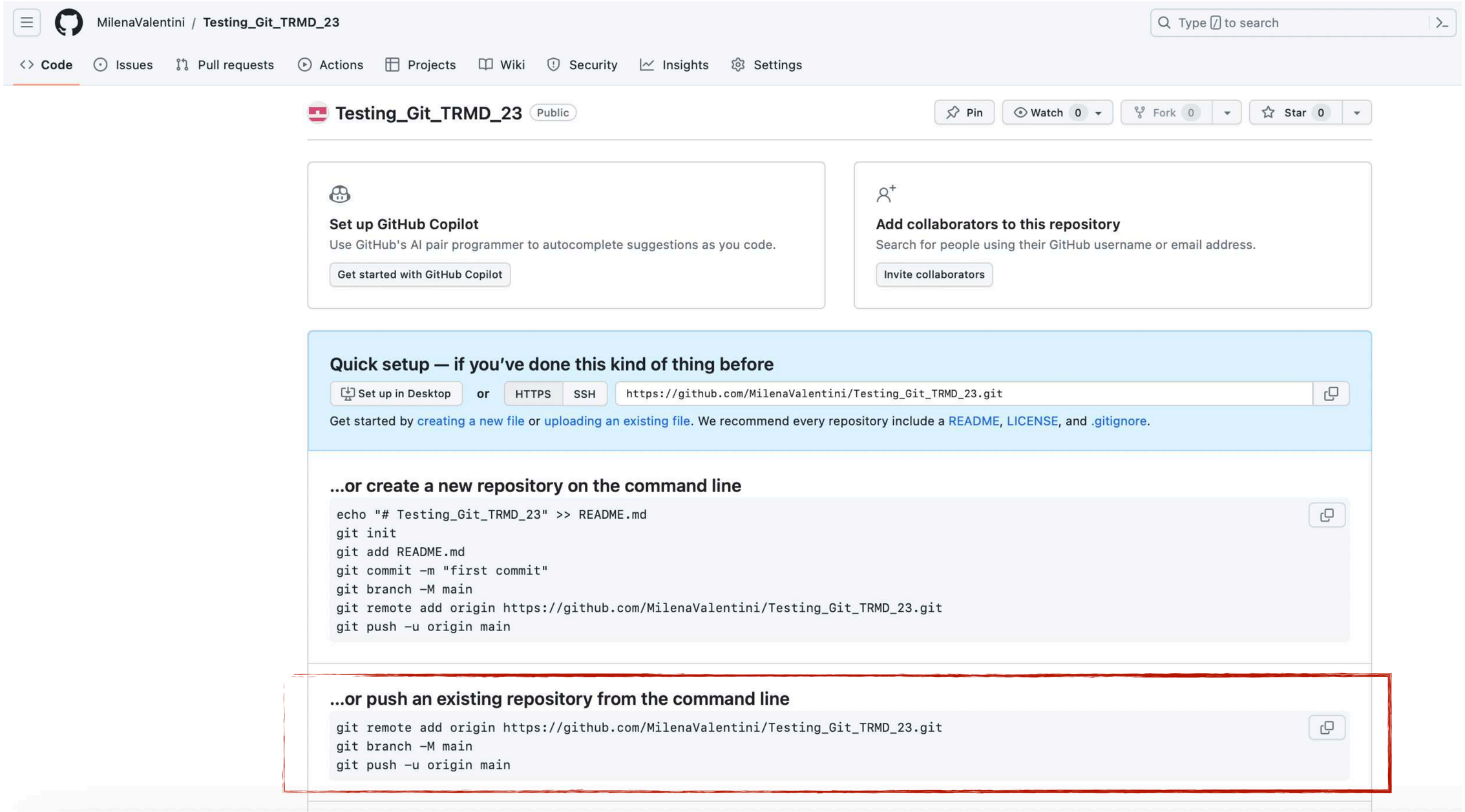


Create a new, empty repository on Git

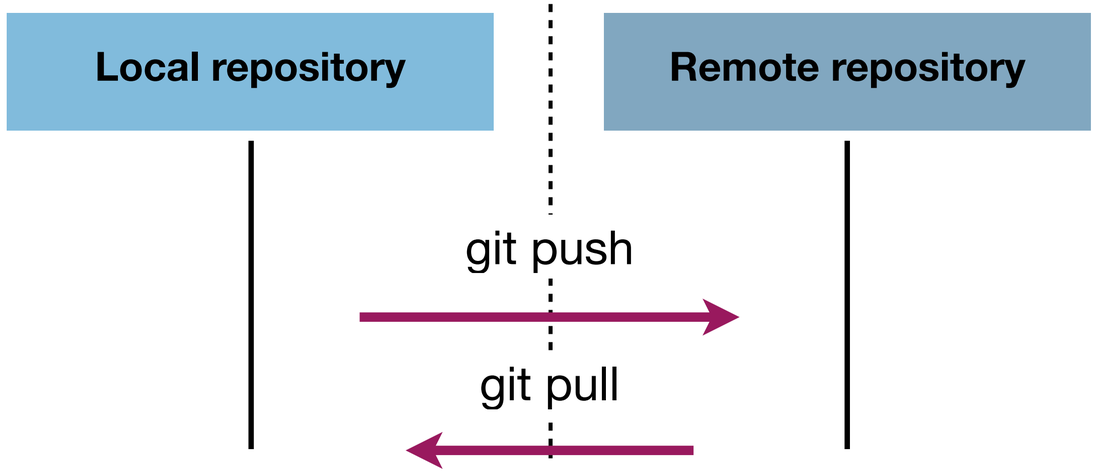
In the main branch in the local repository:

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git remote add origin https://github.com/MilenaValentini/Testing_Git_TRMD_23
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch -M main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git push -u origin main
Username for 'https://github.com': valentini
Password for 'https://valentini@github.com':
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (24/24), 1.90 KiB | 1.90 MiB/s, done.
Total 24 (delta 10), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (10/10), done.
To https://github.com/MilenaValentini/Testing_Git_TRMD_23
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ _
```

Git: from local to remote repository



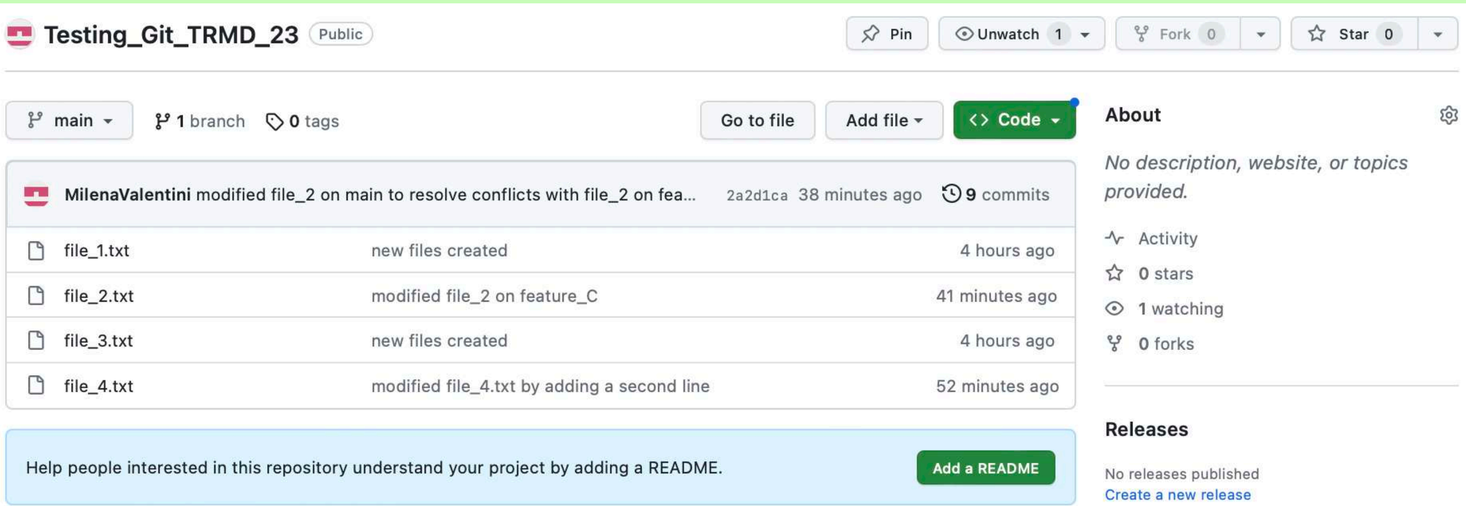
Git: from local to remote repository



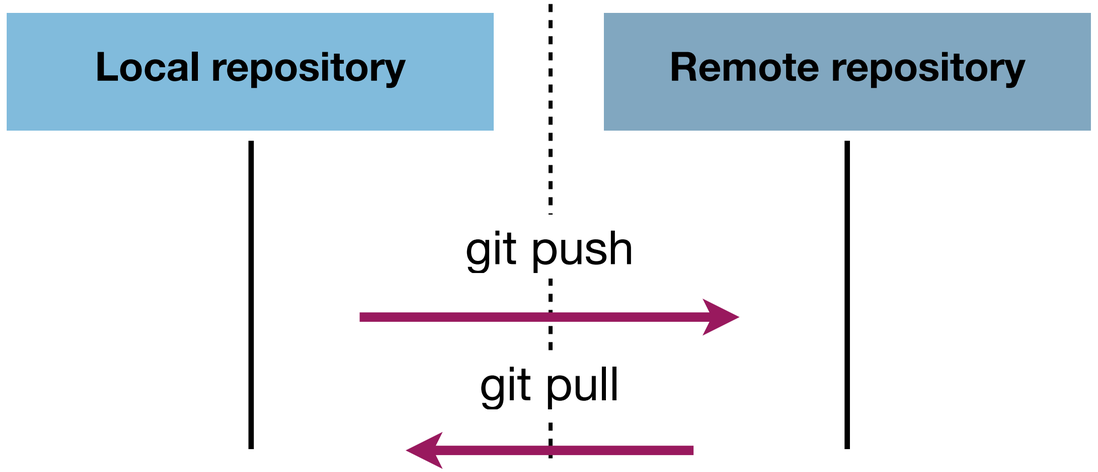
Create a new, empty repository on Git

In the main branch in the local repository:

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git remote add origin https://github.com/MilenaValentini/Testing_Git_TRMD_23
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch -M main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git push -u origin main
Username for 'https://github.com': valentini
Password for 'https://valentini@github.com':
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (24/24), 1.90 KiB | 1.90 M
Total 24 (delta 10), reused 0 (delta 0), pack-re
remote: Resolving deltas: 100% (10/10), done.
To https://github.com/MilenaValentini/Testing_Gi
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
(base) MacBook-Pro-2:Git_test_folder milenavalen
```



Git: from local to remote repository

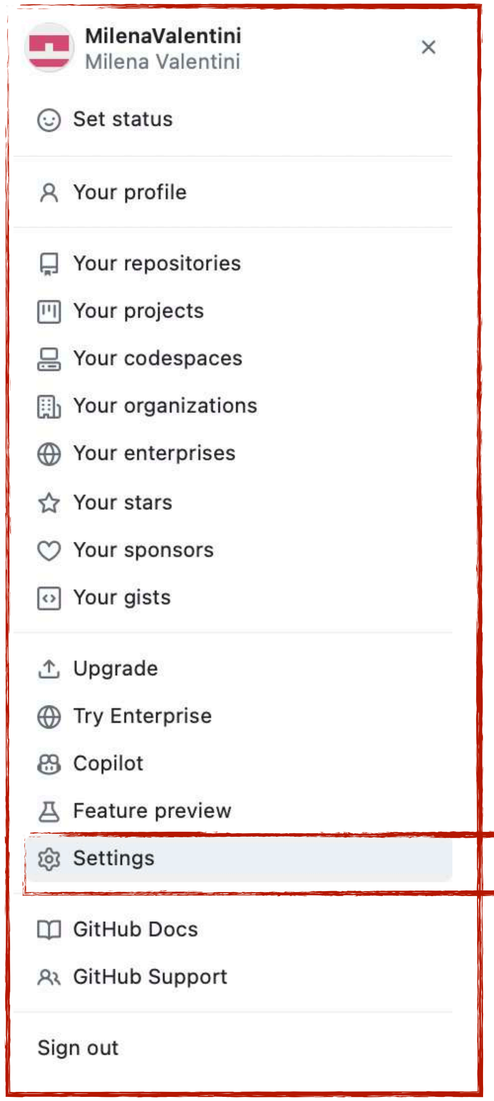


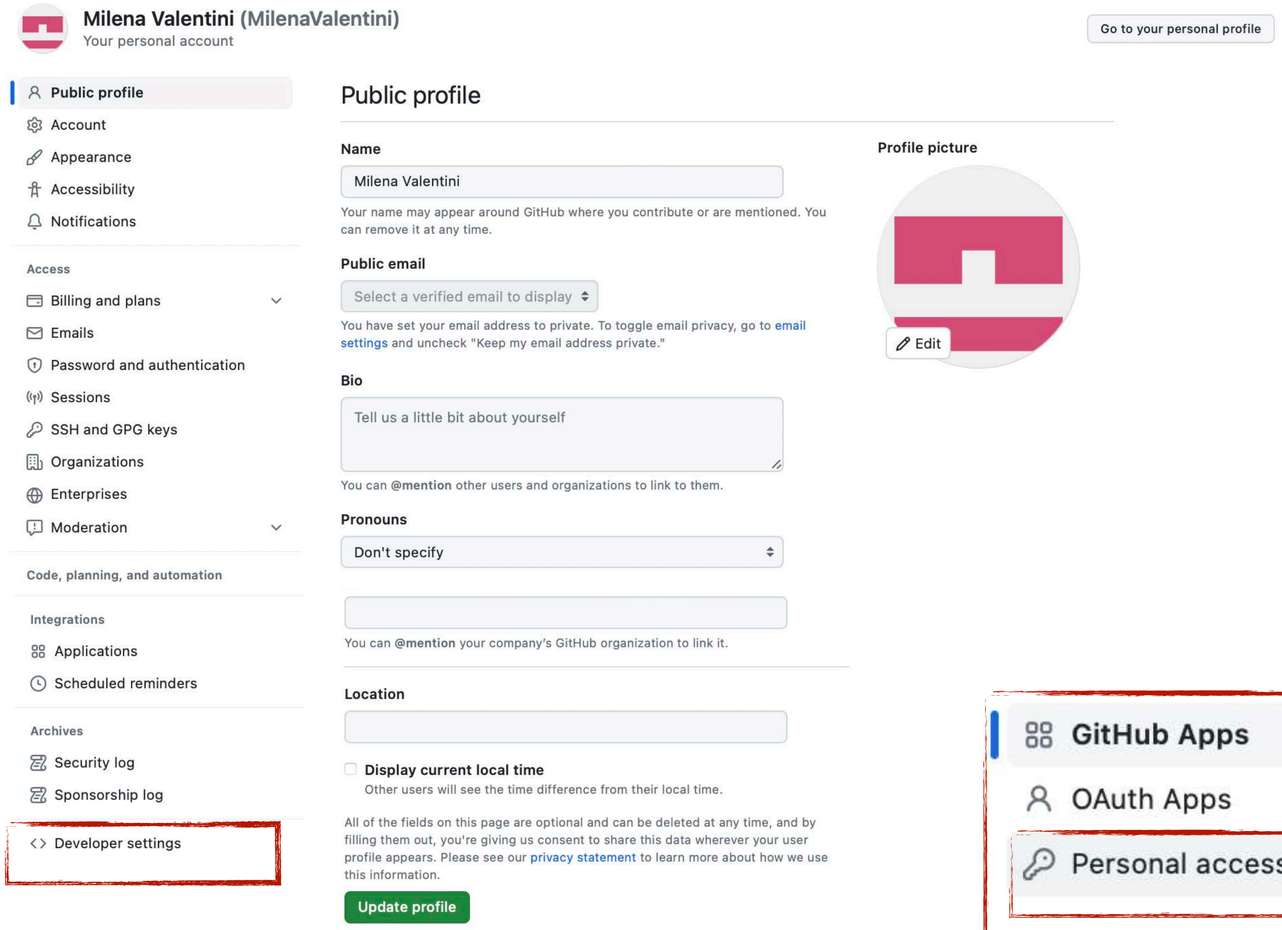
Create a new, empty repository on Git

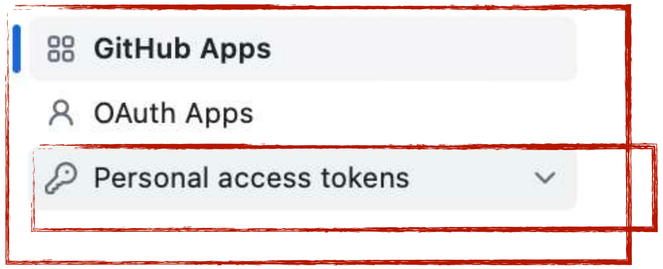
In the main branch in the local repository:

```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git remote add origin https://github.com/MilenaValentini/Testing_Git_TRMD_23
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch -M main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git push -u origin main
Username for 'https://github.com': valentini
Password for 'https://valentini@github.com':
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (24/24), 1.90 KiB | 1.90 MiB/s, done.
Total 24 (delta 10), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (10/10), done.
To https://github.com/MilenaValentini/Testing_Git_TRMD_23
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ _
```

Git: from local to remote repository

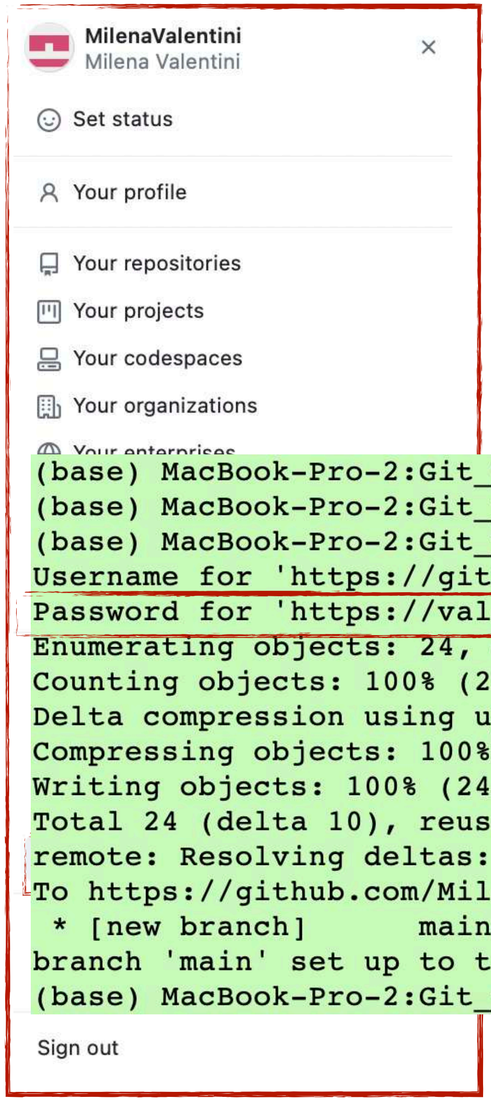


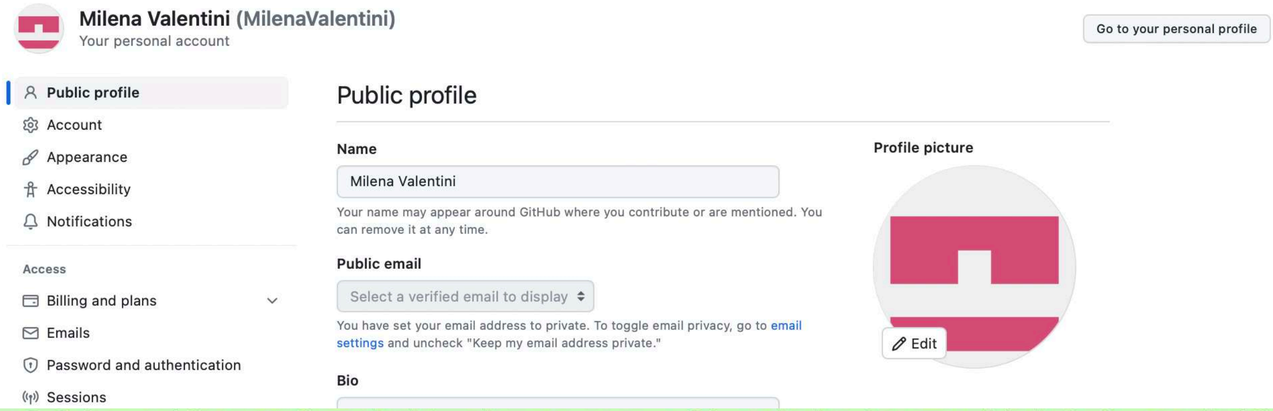




Settings —> Developer settings —> Personal access tokens —> classic —>
Generate new token —> set expiration date and allow everything —> Generate token

Git: from local to remote repository






```
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git remote add origin https://github.com/MilenaValentini/Testing_Git_TRMD_23
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git branch -M main
(base) MacBook-Pro-2:Git_test_folder milenavalentini$ git push -u origin main
Username for 'https://github.com': valentini
Password for 'https://valentini@github.com':
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (24/24), 1.90 KiB | 1.90 MiB/s, done.
Total 24 (delta 10), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (10/10), done.
To https://github.com/MilenaValentini/Testing_Git_TRMD_23
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
(base) MacBook-Pro-2:Git_test_folder milenavalentini$
```

Settings —> Developer settings —> Personal access tokens —> classic —>
Generate new token —> set expiration date and allow everything —> Generate token

Git: from local to remote repository