

# Programmazione Mobile

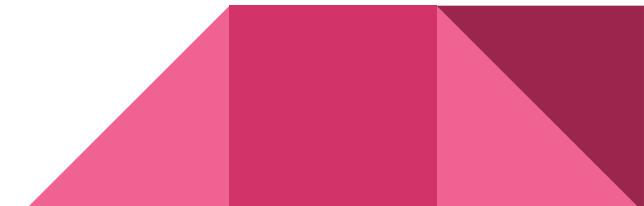
**Nicola Noviello**

[nicola.noviello@unimol.it](mailto:nicola.noviello@unimol.it)

Corso di Laurea in Informatica  
Dipartimento di Bioscienze e Territorio  
Università degli Studi del Molise  
Anno 2023/2024

# Lezione: Introduzione a Git (parte prima)

- Concetti base di Git
- Comandi principali
- Branches
- Pull (Merge) Request
- Risoluzione dei conflitti
- .gitignore
- Cenni introduttivi di tecniche avanzate



# Version Control

# Version Control

- L'utilizzo di Version Control offre un vantaggio concreto agli sviluppatori che lavorano in contemporanea sullo stesso codice (code base)
- Il codice è caricato su internet in maniera centralizzata (**Code Repository**)
- Ogni sviluppatore che lavora a quel codice lavora su una copia locale dell'intero progetto
- Il progetto è acquisito (**fetched**) dalla repo remota e le modifiche, una volta terminate saranno inviate (**pushed**) alla repo

# Version Control

- Git è il più moderno dei Version Control (ma ne esistono altri)
- Git è in grado di unire (**merge**) in maniera automatica i blocchi di codice che arrivano dai diversi sviluppatori che lavorano ad un progetto
- Git **NON** è in grado di gestire automaticamente i conflitti (**merge conflicts**) quando una stessa linea di codice è stata cambiata da due sviluppatori. In quel caso il conflitto deve essere gestito manualmente
- Il progetto è acquisito (**fetched**) dalla repo remota e le modifiche, una volta terminate saranno inviate (**pushed**) alla repo. Per questo motivo una delle migliori pratiche è quella di acquisire (**pull**) codice e di inviare (**push**) anche piccole modifiche e molto spesso. Questa tecnica viene chiamata **Continuous Integration**

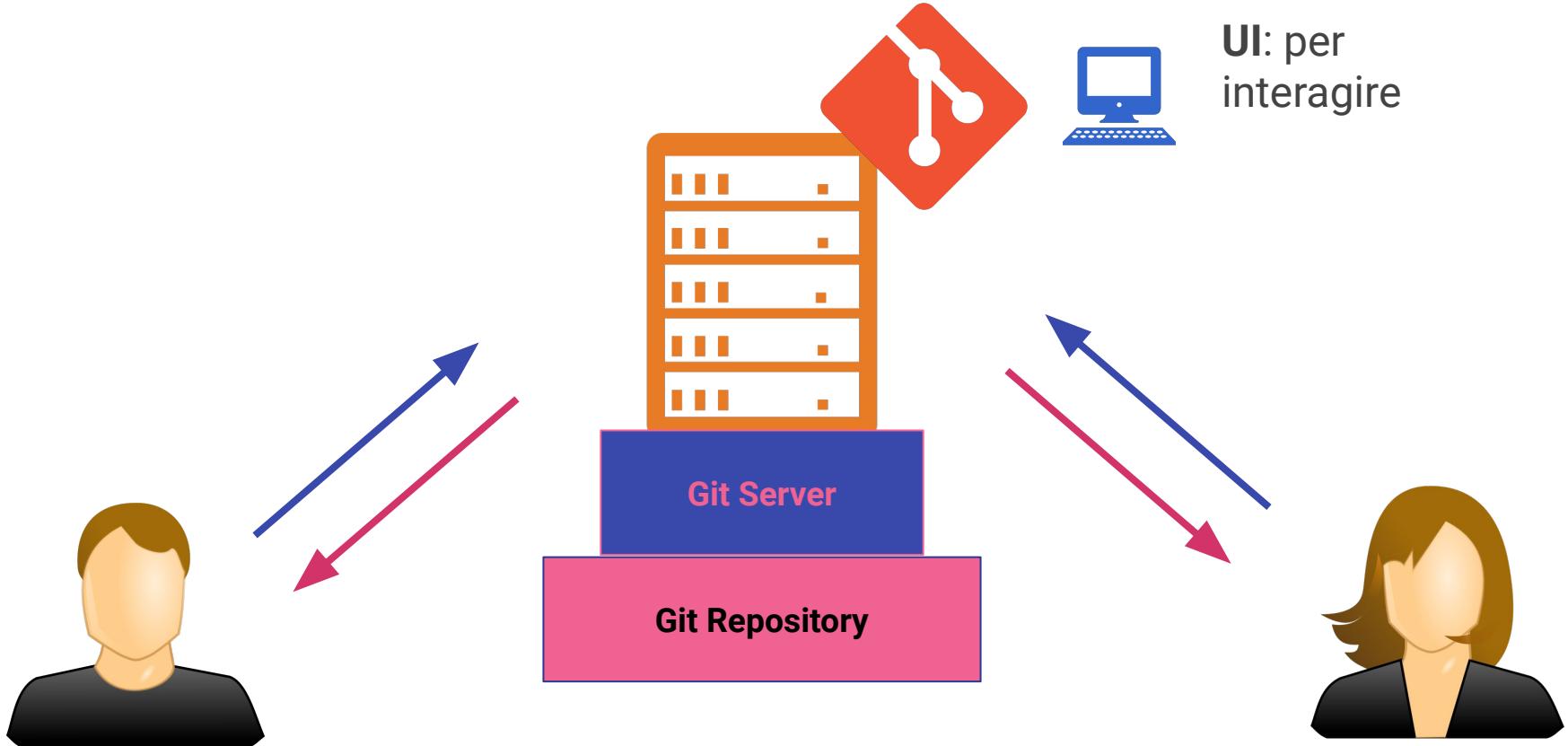
# History

Ma cosa succede se invio del codice che rompe tutto?

Git mantiene la storia (**history of changes**) di tutte le modifiche e i cambiamenti che vengono effettuati nel codice ed in caso di introduzione degli errori è possibile spostarsi ad una versione precedente (**revert**)

# Concetti base di Git e primi passi su GitHub

- **Repository Git**



# Repository Git

- **Remote di un Repository Git:** dove è presente il codice su un server Git
- **Local Git Repository:** una copia locale del codice
- **History:** lista dei cambiamenti sul codice (accessibile anche con il comando `git log`)
- **Staging:** contiene le modifiche che dovranno far parte di una commit
- **Git client:** strumenti che permettono l'esecuzione di comandi git (da terminale, da ide, da web, etc.)

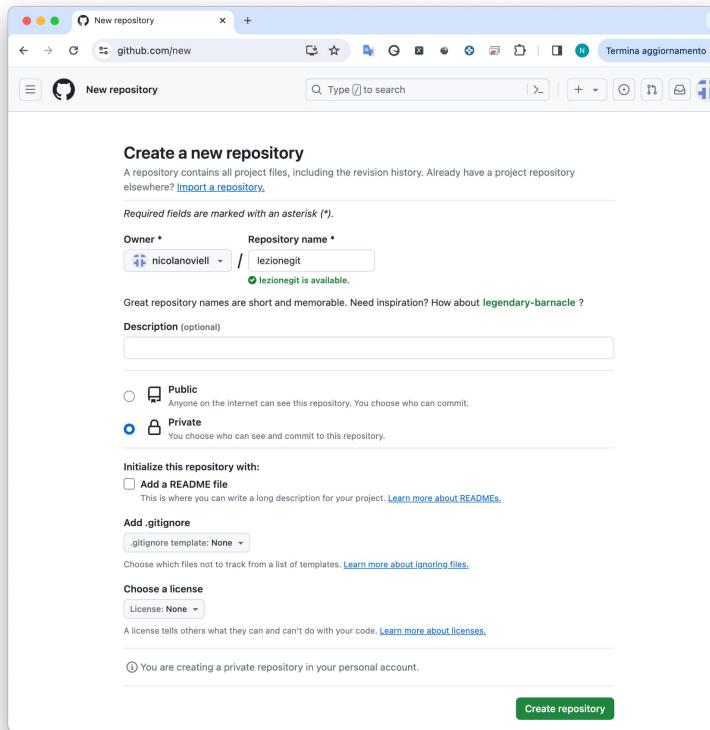
# Repository Git



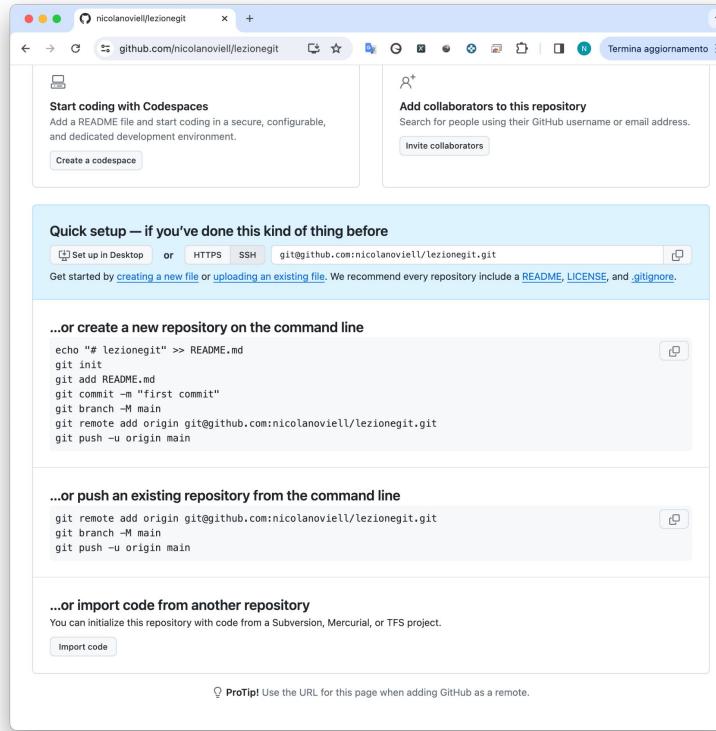
# Repository Git Remoto

- Esistono diversi Git Repository pubblici come **GitHub**, **GitLab**, Bitbucket, ognuno con le proprie caratteristiche ed i propri punti di forza;
- Alcune aziende offrono la possibilità di installare i propri prodotti su Server Git privati;
- I repository possono essere pubblici o privati, l'utilizzo dipende dalla situazione, di solito quelli pubblici hanno una pubblica utilità (es. progetti open source)
- Tramite UI ed API è possibile gestire molti aspetti legati ai progetti in maniera semplificata

# Creazione di un nuovo repository



# Creazione di un nuovo repository

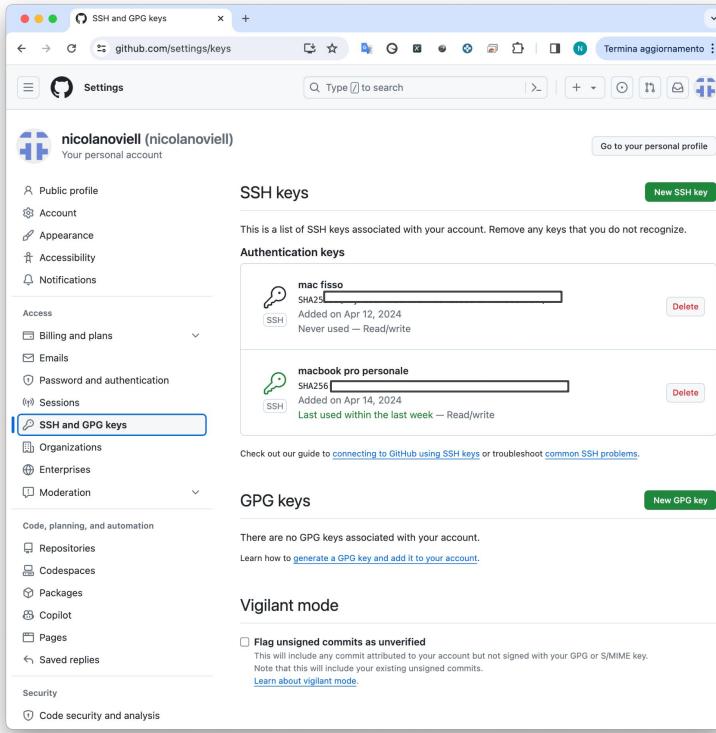


# Creazione di una chiave ssh

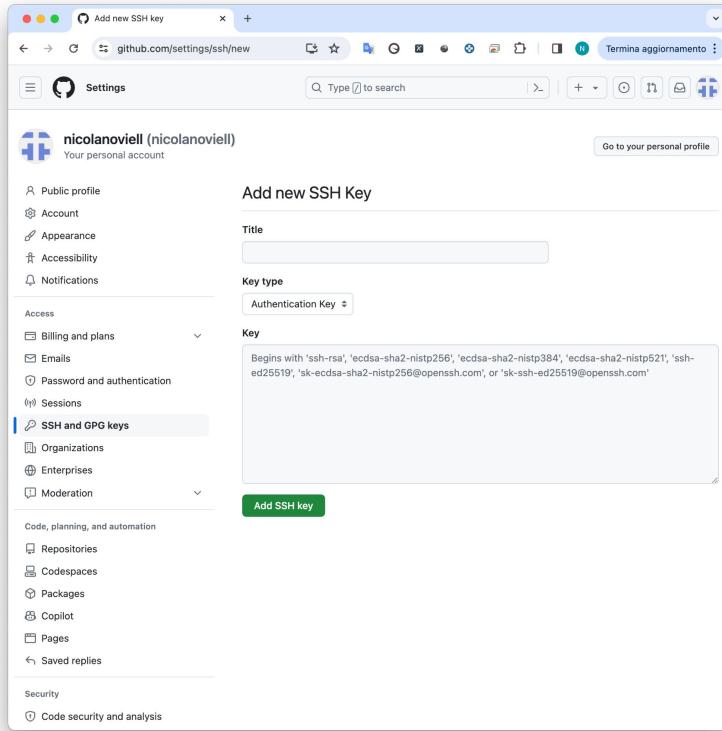
```
flutter-proj -- zsh - 110x27
nico@Mac-mini-di-Nico flutter-proj % ssh-keygen -t ed25519 -C "nicola.noviello@unimol.it"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/Users/nico/.ssh/id_ed25519): /Users/nico/.ssh/githubunimol
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/nico/.ssh/githubunimol
Your public key has been saved in /Users/nico/.ssh/githubunimol.pub
The key fingerprint is:
SHA256: [REDACTED] nicola.noviello@unimol.it
The key's randomart image is:
+--[ED25519 Key]--+
|          .         |
|          ..        |
|          o         |
|          .+        |
|          ..        |
|          .+        |
|          ..        |
|          .+        |
|          ..        |
+----[S]
```

```
nico@Mac-mini-di-Nico flutter-proj % cat /Users/nico/.ssh/githubunimol.pub
```

# Aggiunta della chiave ssh su GitHub



# Aggiunta della chiave ssh su GitHub



# git clone

```
lezionegit -- zsh -- 110x27
nico@Mac-mini-di-Nico flutter-proj % git clone git@github.com:nicolanoviell/lezionegit.git
Cloning into 'lezionegit'...
warning: You appear to have cloned an empty repository.
nico@Mac-mini-di-Nico flutter-proj % cd lezionegit
nico@Mac-mini-di-Nico lezionegit % ls -lha
total 0
drwxr-xr-x  3 nico  staff   96B 28 Apr 11:12 .
drwxr-xr-x 11 nico  staff  352B 28 Apr 11:12 ..
drwxr-xr-x  9 nico  staff  288B 28 Apr 11:12 .git
nico@Mac-mini-di-Nico lezionegit %
```

# git config

```
[nico@Mac-mini-di-Nico lezionegit % git config --global user.name "Nicola Noviello"
nico@Mac-mini-di-Nico lezionegit % git config --global user.email "nicola.noviello@unimol.it"]
```

# Repository Git



# Creazione / modifica di un file locale (working directory)



A screenshot of a macOS terminal window. The window title is "lezionegit — zsh — 94x21". The command line shows "nico@Mac-mini-di-Nico lezionegit % vim README.md". The terminal is otherwise empty, indicating the file has not been edited yet.

Creazione / modifica di un file locale (working directory)

The screenshot shows a terminal window with a light gray background. In the top left corner, there are three small colored circles: red, yellow, and green. The title bar at the top center reads "lezionegit — vim README.md — 94x21". The main area of the terminal contains the following text:

```
Solo una nota nel mio README
```

Below this, there is a vertical column of approximately 30 purple tilde (~) characters, likely representing a cursor or a placeholder for text. At the bottom left, the command ":wq" is visible, indicating the user is about to save and quit the file.

# git status

```
nico@Mac-mini-di-Nico lezionegit % vim README.md
nico@Mac-mini-di-Nico lezionegit % git status
On branch main

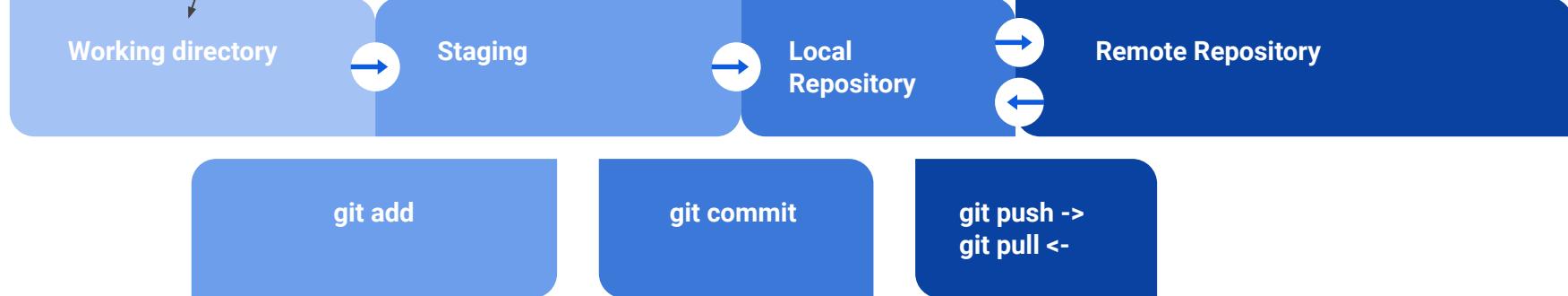
No commits yet

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)
nico@Mac-mini-di-Nico lezionegit %
```

# Repository Git

Prima modifica al  
readme



# git add

```
nico@Mac-mini-di-Nico lezionegit % git status
On branch main

No commits yet

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)
nico@Mac-mini-di-Nico lezionegit % git add README.md
nico@Mac-mini-di-Nico lezionegit % git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   README.md

nico@Mac-mini-di-Nico lezionegit %
```

# Repository Git

Prima modifica al  
readme



# Cosa succede se faccio una modifica dopo il git add?



A screenshot of a macOS terminal window. The window title is "lezionegit — zsh — 94x21". The terminal prompt shows "nico@Mac-mini-di-Nico lezionegit % vim README.md" followed by a blank line where the user has just typed "nico@Mac-mini-di-Nico lezionegit %". The window has the standard OS X title bar with red, yellow, and green buttons.

# Cosa succede se faccio una modifica dopo il git add?

# git status

```
nico@Mac-mini-di-Nico lezionegit % git status
On branch main

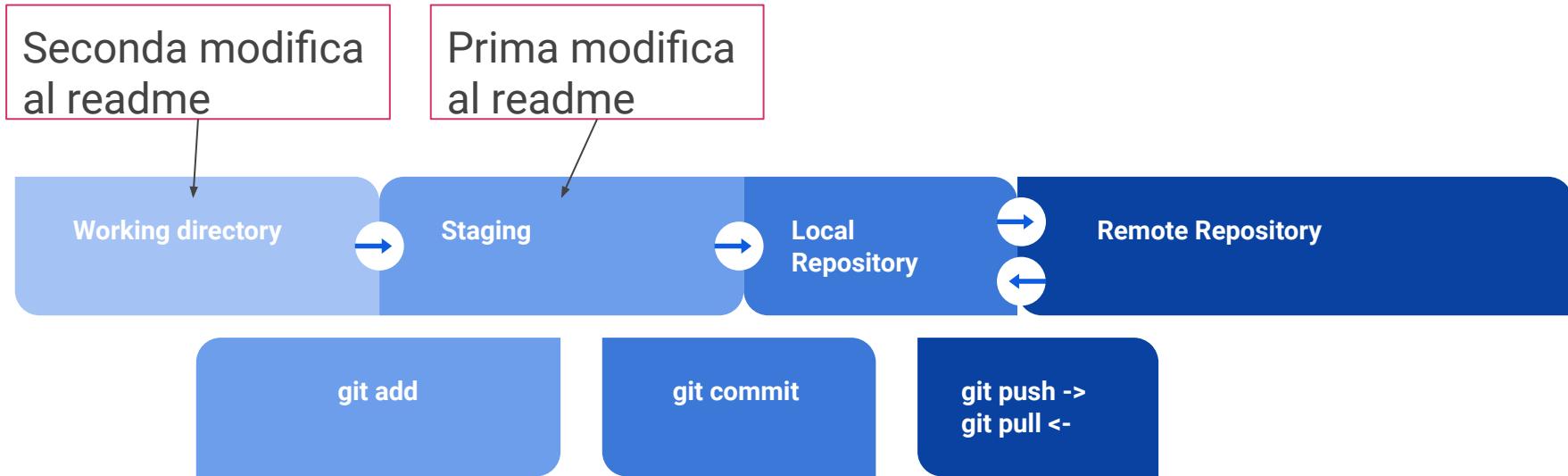
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   README.md

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:  README.md

nico@Mac-mini-di-Nico lezionegit %
```

# Repository Git



# git commit

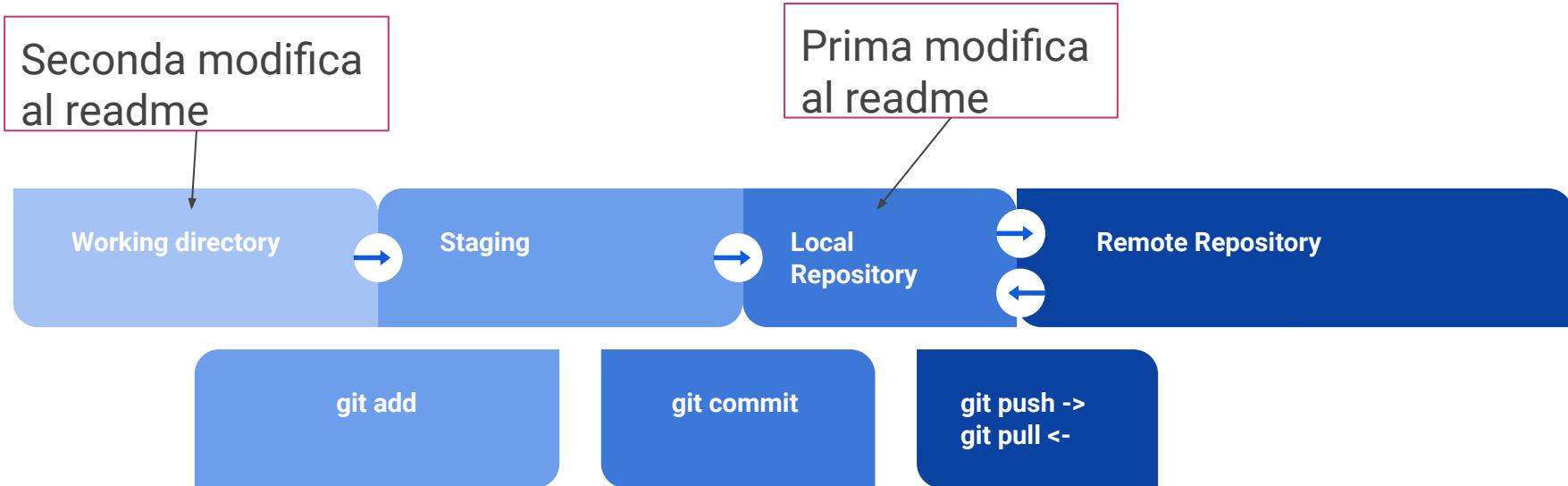


A screenshot of a macOS terminal window. The window title is "lezionegit — zsh — 94x21". The terminal prompt shows "nico@Mac-mini-di-Nico lezionegit % git commit" followed by a cursor. The terminal has the standard OS X look with red, yellow, and green window control buttons.

# git commit

```
Aggiunta la prima versione del readme
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
#
# Initial commit
#
# Changes to be committed:
#       new file:   README.md
#
# Changes not staged for commit:
#       modified:  README.md
#
~
~
~
~
~
~
-- INSERT --
```

# Repository Git



# git status

```
nico@Mac-mini-di-Nico lezionegit % git status
On branch main
Your branch is based on 'origin/main', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

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:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
nico@Mac-mini-di-Nico lezionegit %
```

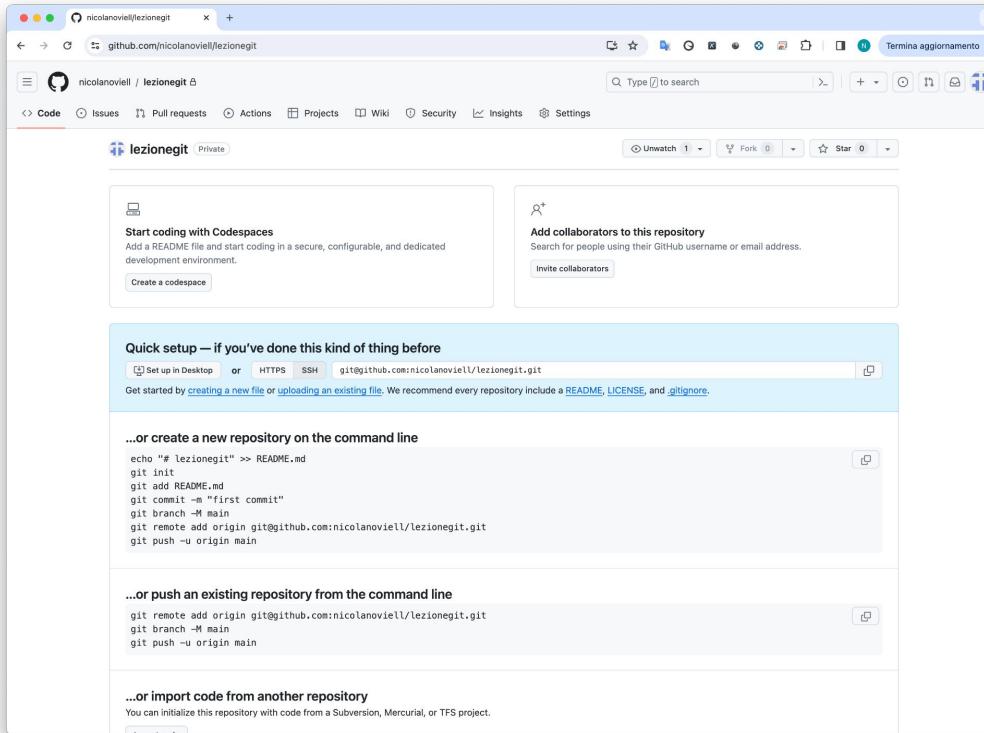
# git log (lista dei commit)

```
nico@Mac-mini-di-Nico lezionegit % git log
commit d61b339ce95284d370bf29bef4b5bccfd513b55d (HEAD -> main)
Author: Nicola Noviello <nicola.noviello@unimol.it>
Date:   Sun Apr 28 12:54:15 2024 +0200

    Aggiunta la prima versione del readme
nico@Mac-mini-di-Nico lezionegit %
```

Questa è la lista dei commit del Local repository o del remote repository?

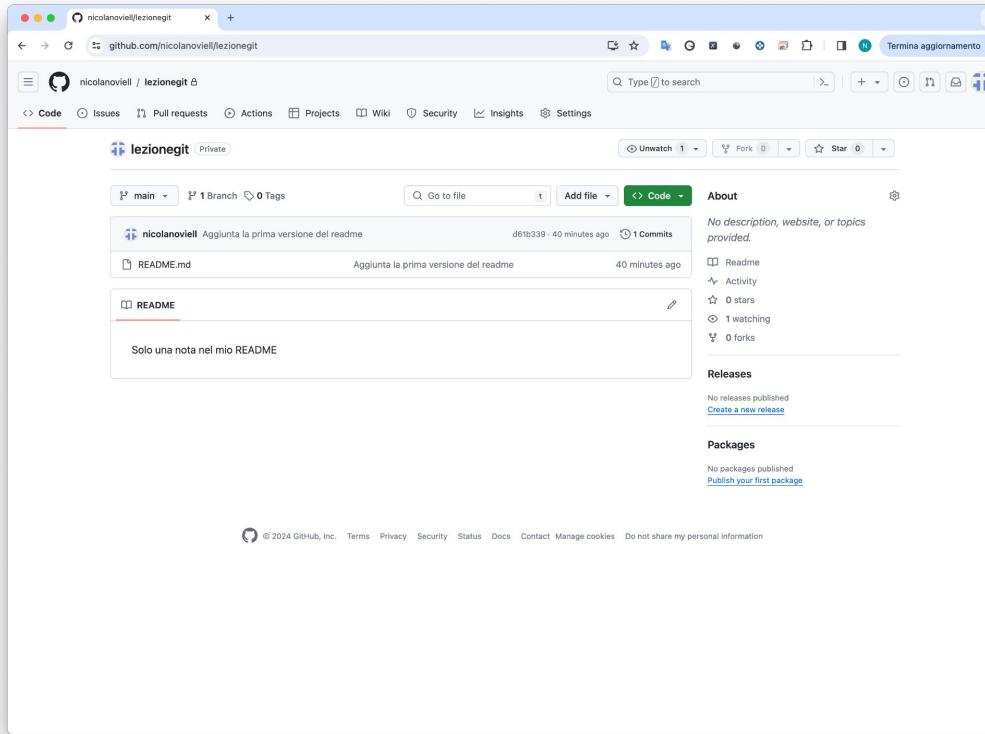
# remote



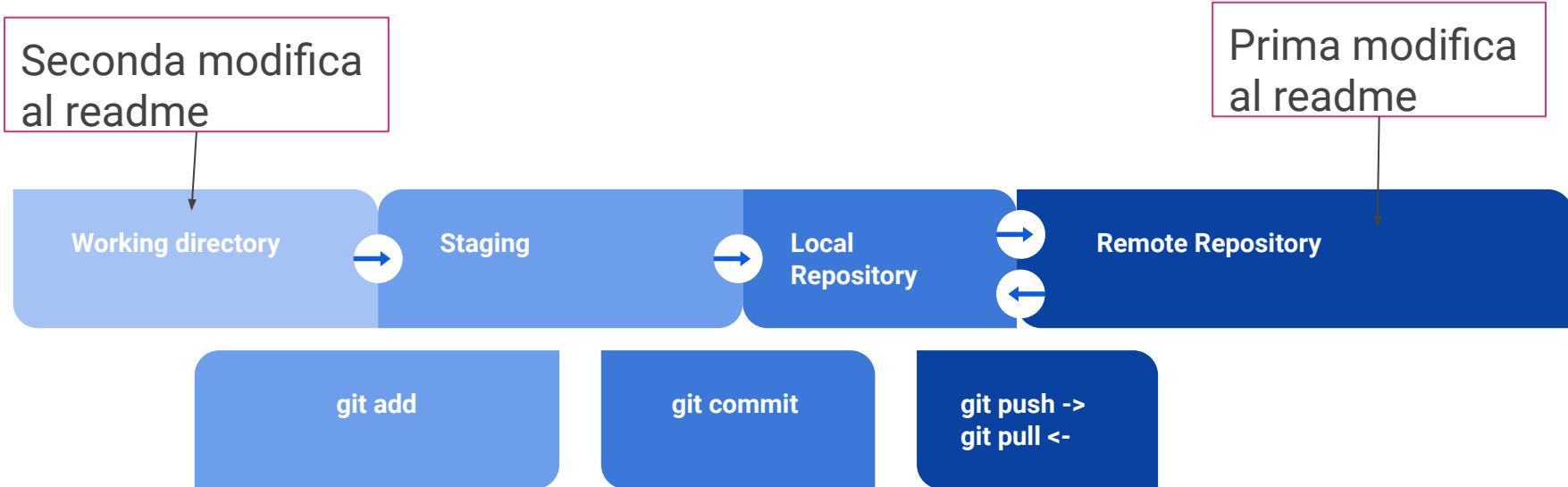
# git push

```
nico@Mac-mini-di-Nico lezionegit % git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 264 bytes | 264.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:nicolanoviell/lezionegit.git
 * [new branch]      main -> main
nico@Mac-mini-di-Nico lezionegit %
```

# git push



# Repository Git



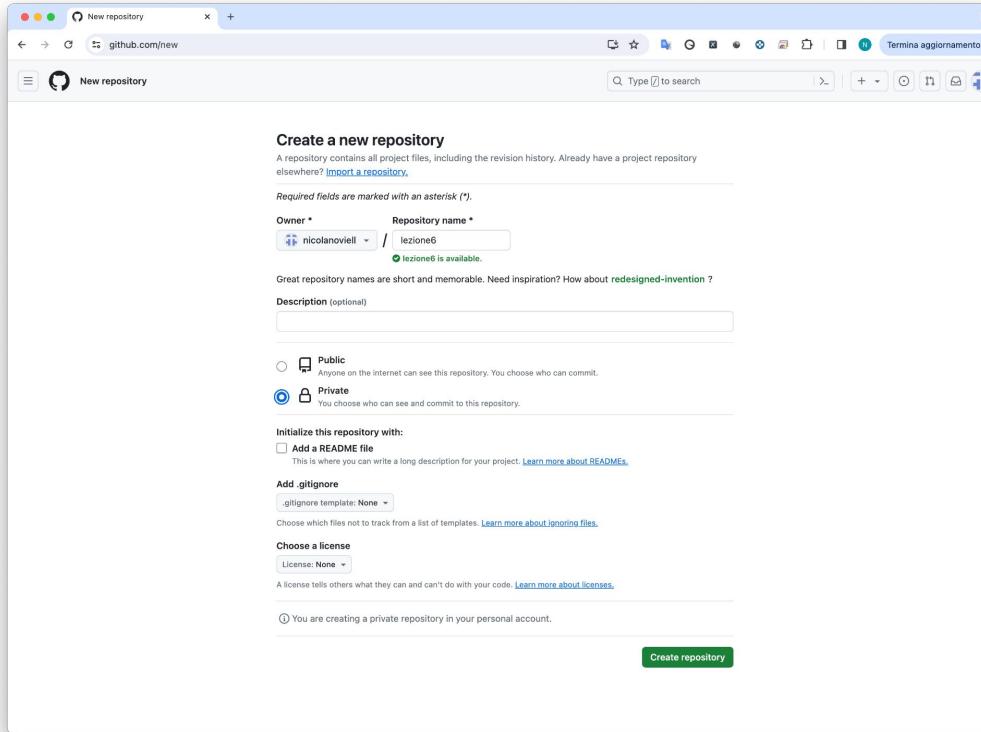


# Portare su GitHub un progetto locale

# Progetto di partenza

```
nico@Mac-mini-di-Nico lezione6 % ls
README.md          lezione6.iml      pubspec.yaml
analysis_options.yaml lib            test
android           linux           web
images             macos          windows
ios                pubspec.lock
nico@Mac-mini-di-Nico lezione6 % git status
fatal: not a git repository (or any of the parent directories): .git
nico@Mac-mini-di-Nico lezione6 %
```

# Creo un nuovo progetto su GitHub



# git init

```
nico@Mac-mini-di-Nico lezione6 % git init
Initialized empty Git repository in /Users/nico/flutter-proj/lezione6/.git/
nico@Mac-mini-di-Nico lezione6 % ls -a
.
..
.DS_Store
.git
.gitignore
.idea
.metadata
 README.md      linux
 analysis_options.yaml  macos
 android        pubspec.lock
 images         pubspec.yaml
 ios            test
 lezione6.iml   web
 lib            windows
nico@Mac-mini-di-Nico lezione6 %
```

# git status

```
nico@Mac-mini-di-Nico lezione6 % git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore
    .metadata
    README.md
    analysis_options.yaml
    android/
    images/
    ios/
    lib/
    linux/
    macos/
    pubspec.lock
    pubspec.yaml
    test/
    web/
```

# git add .

```
Untracked files:
(use "git add <file>..." to include in what will be committed)
.gitignore
.metadata
README.md
analysis_options.yaml
android/
images/
ios/
lib/
linux/
macos/
pubspec.lock
pubspec.yaml
test/
web/
windows/

nothing added to commit but untracked files present (use "git add" to track)
[nico@Mac-mini-di-Nico lezione6 % git add .
nico@Mac-mini-di-Nico lezione6 % ]
```

# git status

```
nico@Mac-mini-di-Nico lezione6 % git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:  .gitignore
    new file:  .metadata
    new file:  README.md
    new file:  analysis_options.yaml
    new file:  android/.gitignore
    new file:  android/app/build.gradle
    new file:  android/app/src/debug/AndroidManifest.xml
    new file:  android/app/src/main/AndroidManifest.xml
    new file:  android/app/src/main/kotlin/com/example/lezione6/MainActivity.kt
    new file:  android/app/src/main/res/drawable-v21/launch_background.xml
    new file:  android/app/src/main/res/drawable/launch_background.xml
    new file:  android/app/src/main/res/mipmap-hdpi/ic_launcher.png
    new file:  android/app/src/main/res/mipmap-mdpi/ic_launcher.png
```

# git commit -m “...”

```
nico@Mac-mini-di-Nico lezione6 % git commit -m "progetto iniziale"
[main (root-commit) 19ef67a] progetto iniziale
 136 files changed, 4983 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 .metadata
 create mode 100644 README.md
 create mode 100644 analysis_options.yaml
 create mode 100644 android/.gitignore
 create mode 100644 android/app/build.gradle
 create mode 100644 android/app/src/debug/AndroidManifest.xml
 create mode 100644 android/app/src/main/AndroidManifest.xml
 create mode 100644 android/app/src/main/kotlin/com/example/lezione6/MainActivity.kt
 create mode 100644 android/app/src/main/res/drawable-v21/launch_background.xml
 create mode 100644 android/app/src/main/res/drawable/launch_background.xml
 create mode 100644 android/app/src/main/res/mipmap-hdpi/ic_launcher.png
 create mode 100644 android/app/src/main/res/mipmap-mdpi/ic_launcher.png
 create mode 100644 android/app/src/main/res/mipmap-xhdpi/ic_launcher.png
 create mode 100644 android/app/src/main/res/mipmap-xxhdpi/ic_launcher.png
 create mode 100644 android/app/src/main/res/mipmap-xxxhdpi/ic_launcher.png
 create mode 100644 android/app/src/main/res/values-night/styles.xml
 create mode 100644 android/app/src/main/res/values/styles.xml
```

# git status

```
nico@Mac-mini-di-Nico lezione6 % git status
On branch main
nothing to commit, working tree clean
nico@Mac-mini-di-Nico lezione6 % git log
commit 19ef67a676c17889cebafe21ca2b40d80ff0b06bb (HEAD -> main)
Author: Nicola Noviello <nicola.noviello@unimol.it>
Date:   Sun Apr 28 13:54:02 2024 +0200

    progetto iniziale
nico@Mac-mini-di-Nico lezione6 %
```

# git push



```
nico@Mac-mini-di-Nico lezione6 % git push
fatal: No configured push destination.
Either specify the URL from the command-line or configure a remote repository using
    git remote add <name> <url>
and then push using the remote name
    git push <name>
nico@Mac-mini-di-Nico lezione6 %
```

# git remote add origin [...]

```
lezione6 -- zsh -- 94x21
fatal: No configured push destination.
Either specify the URL from the command-line or configure a remote repository using

    git remote add <name> <url>

and then push using the remote name

    git push <name>

nico@Mac-mini-di-Nico lezione6 % git remote add origin git@github.com:nicolanoviell/lezione6.git
nico@Mac-mini-di-Nico lezione6 % git push
fatal: The current branch main has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin main

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

nico@Mac-mini-di-Nico lezione6 %
```

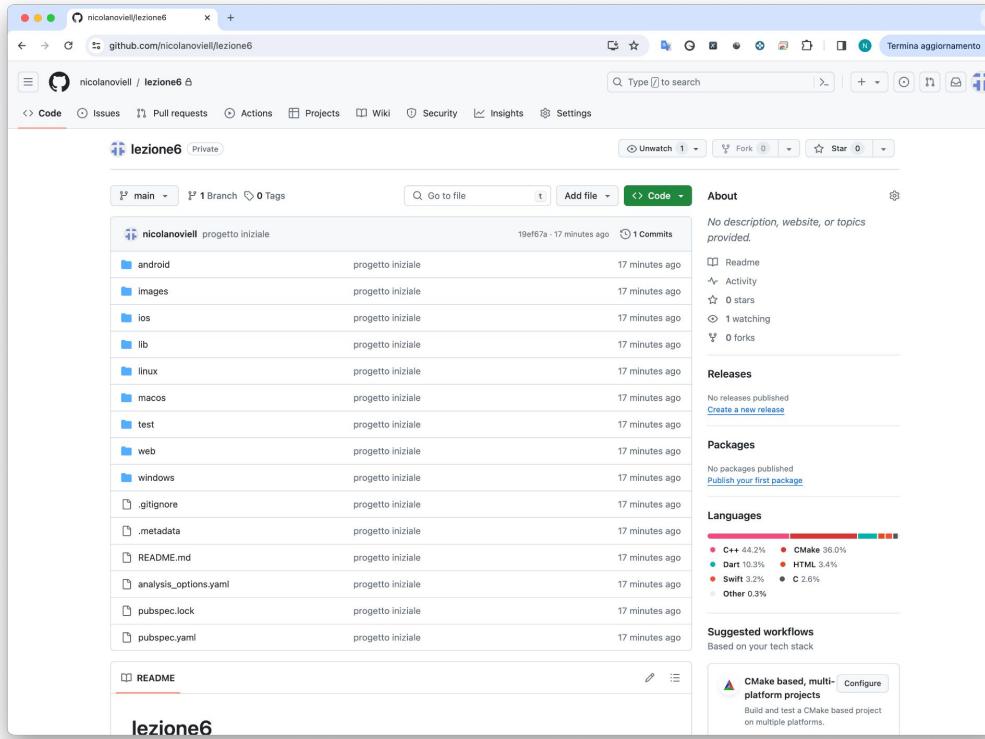
Remote   
Branch 

# git push --set-upstream origin [...]

```
nico@Mac-mini-di-Nico lezione6 % git push --set-upstream origin main
Enumerating objects: 186, done.
Counting objects: 100% (186/186), done.
Delta compression using up to 8 threads
Compressing objects: 100% (156/156), done.
Writing objects: 100% (186/186), 1.08 MiB | 9.65 MiB/s, done.
Total 186 (delta 19), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (19/19), done.
To github.com:nicolanoviell/lezione6.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
nico@Mac-mini-di-Nico lezione6 %
```



# Remote Repository



# E se volessimo “staccare” da git il nostro progetto locale?

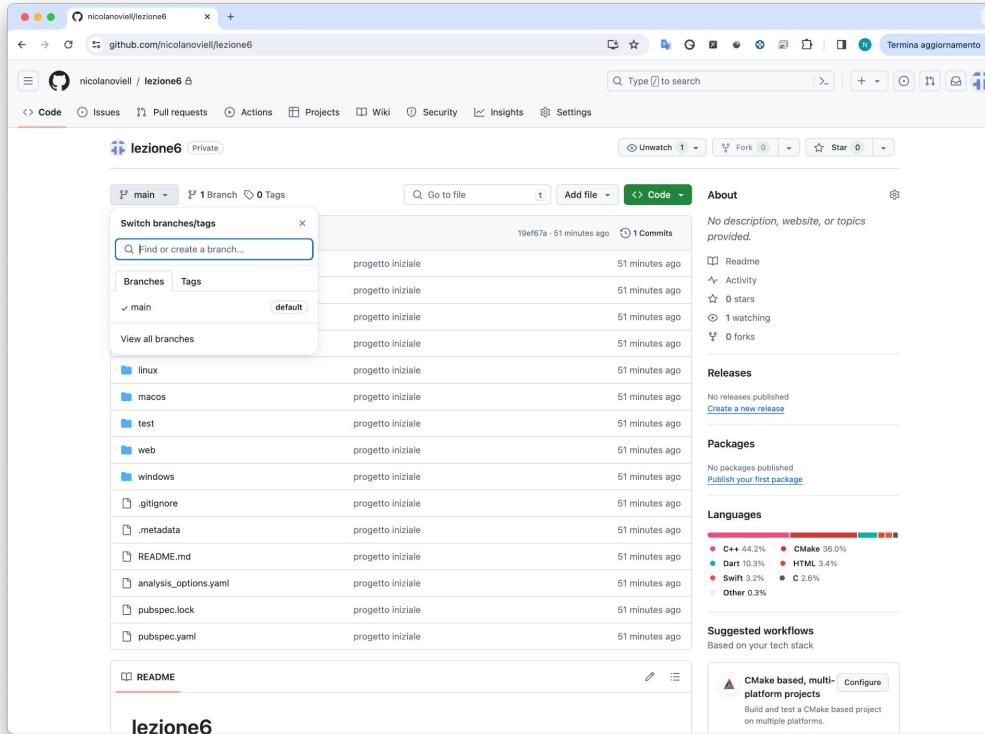
```
nico@Mac-mini-di-Nico lezione6 % ls -a
.
..
.DS_Store
.git
.gitignore
.idea
.metadata
.
README.md
analysis_options.yaml
android
images
ios
lezione6.iml
lib
linux
macos
pubspec.lock
pubspec.yaml
test
web
windows

nico@Mac-mini-di-Nico lezione6 % git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
nico@Mac-mini-di-Nico lezione6 % rm -Rf .git
nico@Mac-mini-di-Nico lezione6 % git status
fatal: not a git repository (or any of the parent directories): .git
nico@Mac-mini-di-Nico lezione6 %
```

# Branches

# Branch: main



# Funzionamento dei branch



Master o Main branch, viene creato di default quando si crea un progetto

# Funzionamento dei branch



**main**  
Progetto iniziale

**main**  
Modifica Readme

**main**  
Feature 1



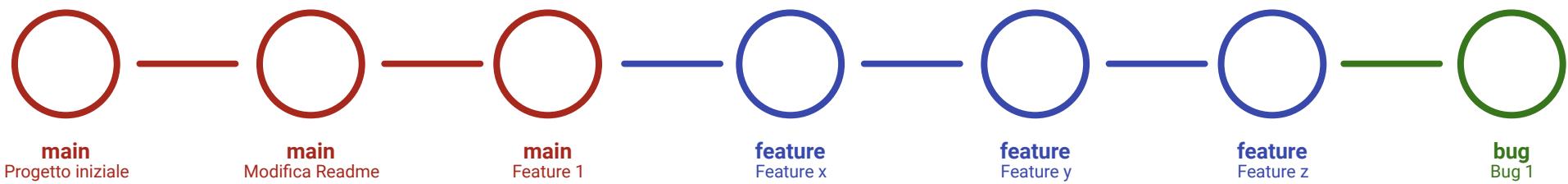
**feature**  
Feature x

**feature**  
Feature y

**feature**  
Feature z

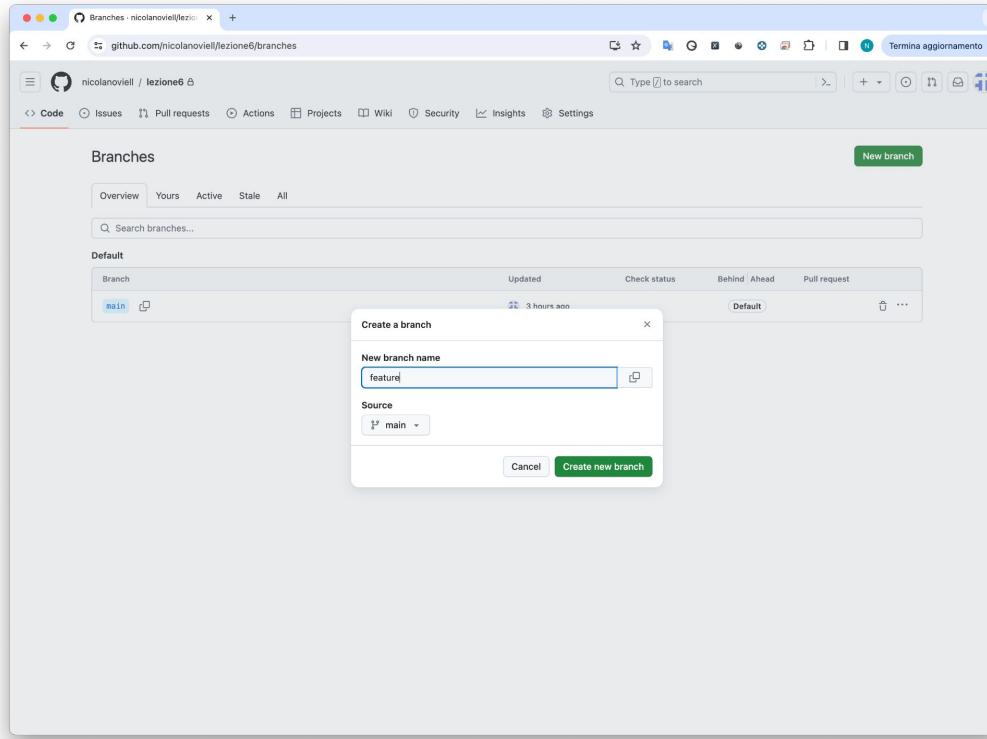
Ne creiamo altri per aggiungere feature o correggere bug

# Funzionamento dei branch



L'obiettivo è quello di avere il branch `main` che abbia tutte le funzionalità ma che sia anche sempre pronto per la messa in produzione

# Creazione di un branch



# Creazione di un branch: caratteristiche

- L'ideale è creare dei branch specifici per feature o bug: bisogna ridurre al minimo la dimensione di un nuovo branch, così da riuscire a mantenere sempre il branch principale funzionante (in caso di errore il revert è molto semplice, sarà difficile romperlo);
- Grandi feature o branch aperti da molto tempo aumentano il rischio di conflitti durante le merge ed aumentano il rischio di introduzione di nuovi bug;
- Quando si crea un nuovo branch è importante partire dalla codebase principale;
- In sintesi: 1 funzionalità, 1 branch

# git checkout

```
nico@Mac-mini-di-Nico lezione6 % git branch
* main
nico@Mac-mini-di-Nico lezione6 % git pull
From github.com:nicolanoviell/lezione6
 * [new branch]      feature    -> origin/feature
```

# git checkout

```
nico@Mac-mini-di-Nico lezione6 % git status
On branch main
nothing to commit, working tree clean
nico@Mac-mini-di-Nico lezione6 % git branch
  feature
* main
nico@Mac-mini-di-Nico lezione6 % git checkout feature
Switched to branch 'feature'
Your branch is up to date with 'origin/feature'.
nico@Mac-mini-di-Nico lezione6 %
```

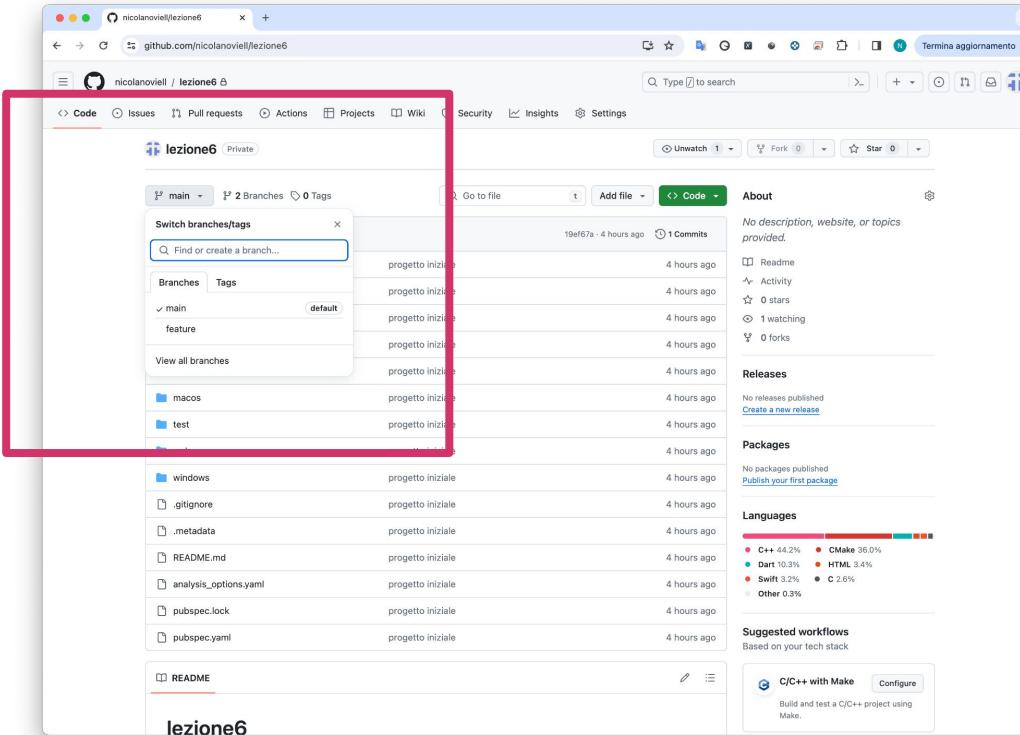
Ora sto lavorando sul branch “feature”

# git checkout -b nuovobranch

```
nico@Mac-mini-di-Nico lezione6 % git status
On branch main
nothing to commit, working tree clean
nico@Mac-mini-di-Nico lezione6 % git branch
  feature
* main
nico@Mac-mini-di-Nico lezione6 % git checkout feature
Switched to branch 'feature'
Your branch is up to date with 'origin/feature'.
nico@Mac-mini-di-Nico lezione6 % git checkout -b bugfix/readmesbagliato
Switched to a new branch 'bugfix/readmesbagliato'
nico@Mac-mini-di-Nico lezione6 %
```

Ho creato un nuovo local branch e sto lavorando su quello

## Nuovo local branch



# Creazione di un nuovo remote branch

```
nico@Mac-mini-di-Nico lezione6 % git status
On branch bugfix/readmesbagliato
nothing to commit, working tree clean
nico@Mac-mini-di-Nico lezione6 % git branch
* bugfix/readmesbagliato
  feature
  main
nico@Mac-mini-di-Nico lezione6 %
```

# Creazione di un nuovo remote branch

```
nico@Mac-mini-di-Nico lezione6 % vim README.md
nico@Mac-mini-di-Nico lezione6 % git add .
nico@Mac-mini-di-Nico lezione6 % git status
On branch bugfix/readmesbagliato
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md

nico@Mac-mini-di-Nico lezione6 % git push
fatal: The current branch bugfix/readmesbagliato has no upstream branch.
To push the current branch and set the remote as upstream, use

  git push --set-upstream origin bugfix/readmesbagliato

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

nico@Mac-mini-di-Nico lezione6 % git push --set-upstream origin bugfix/readmesbagliato
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'bugfix/readmesbagliato' on GitHub by visiting:
```

# Creazione di un nuovo remote branch

```
modified: README.md

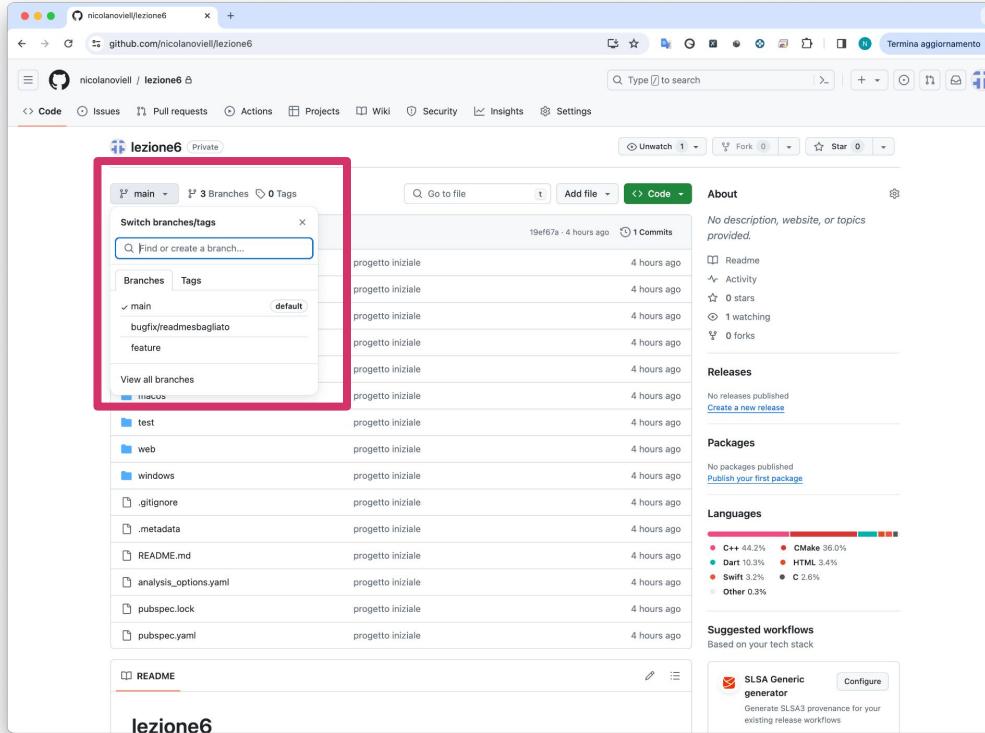
[nico@Mac-mini-di-Nico lezione6 % git push
fatal: The current branch bugfix/readmesbagliato has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin bugfix/readmesbagliato

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

[nico@Mac-mini-di-Nico lezione6 % git push --set-upstream origin bugfix/readmesbagliato
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'bugfix/readmesbagliato' on GitHub by visiting:
remote:     https://github.com/nicolanoviell/lezione6/pull/new/bugfix/readmesbagliato
remote:
To github.com:nicolanoviell/lezione6.git
 * [new branch]      bugfix/readmesbagliato -> bugfix/readmesbagliato
branch 'bugfix/readmesbagliato' set up to track 'origin/bugfix/readmesbagliato'.
nico@Mac-mini-di-Nico lezione6 %
```

# Creazione di un nuovo remote branch



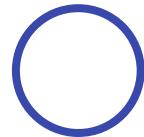
# Situazione attuale dei branch



**feature**  
progetto iniziale



**bugfix/readmesbagliato**  
progetto iniziale



**main**  
progetto iniziale

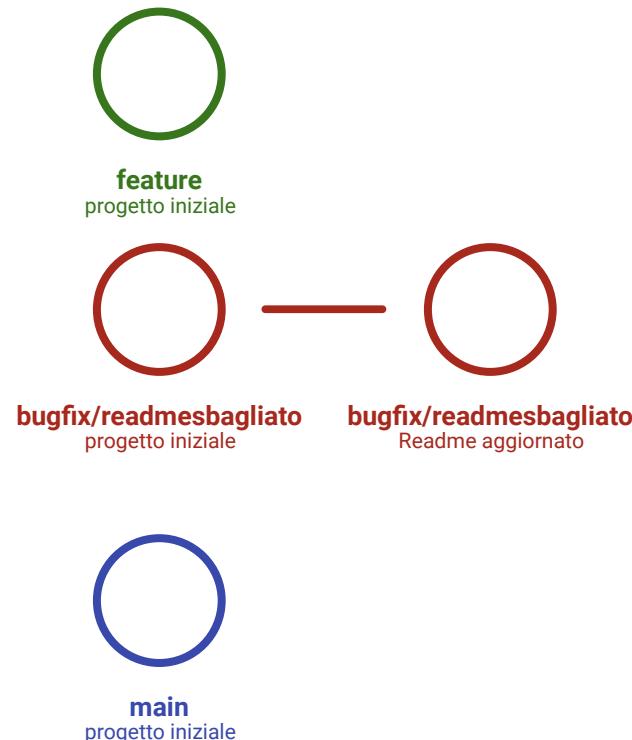
# Commit sul nuovo branch

```
On branch bugfix/readmesbagliato
Your branch is up to date with 'origin/bugfix/readmesbagliato'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md

[nico@Mac-mini-di-Nico lezione6 % git commit -m 'Readme aggiornato'
[bugfix/readmesbagliato d4541a0] Readme aggiornato
  1 file changed, 1 insertion(+), 1 deletion(-)
[nico@Mac-mini-di-Nico lezione6 % git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 327 bytes | 327.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:nicolanoviell/lezione6.git
  19ef67a..d4541a0  bugfix/readmesbagliato -> bugfix/readmesbagliato
nico@Mac-mini-di-Nico lezione6 %
```

# Situazione attuale dei branch



# Continuous Integration

La Continuous Integration (CI) è una pratica fondamentale nello sviluppo software moderno. Consiste nell'integrazione continua dei cambiamenti nel codice sorgente da parte dei diversi membri del team. Questo processo avviene in modo automatico e frequente, generalmente più volte al giorno.

# Continuous Integration

- **Riduzione dei conflitti:** Un branch costantemente aggiornato riduce il rischio di conflitti nel momento in cui si integrano le modifiche. Ciò garantisce una transizione più fluida tra le diverse versioni del codice
- **Test tempestivi:** Gli sviluppatori possono eseguire integration e unit test su un branch sempre aggiornato, individuando e risolvendo prontamente eventuali errori o problemi di compatibilità
- **Feedback immediato:** L'integrazione continua permette di ricevere feedback tempestivo sulle modifiche apportate, consentendo agli sviluppatori di correggere eventuali errori rapidamente

# Esercizio

Create un account GitHub e portate su un repository un progetto realizzato a lezione. Domanda: a cosa serve il file `.gitignore`? Per quale motivo (anche nel caso di un progetto Flutter) è particolarmente importante integrarlo?