

# Primi passi su GitHub

*Introduzione e Descrizione delle  
principali funzioni*

*A cura di : Matteo Ballocco & Marco Ena*

*07/02/18*

# **Argomenti trattati:**

- i. Definizione e presentazione dei software di controllo versione distribuito esistenti**
- ii. Cenni storici e introduzione a Git e GitHub**
- iii. Descrizione struttura e funzionamento di GitHub da Interfaccia web e Terminale**

# Distributed Version Control System

## Controllo versione distribuito (o DVCS) :

Permette di **tenere traccia** delle **modifiche** e delle versioni apportate al codice sorgente del software, **senza** la necessità di dover utilizzare un **server centrale**.

Nel tempo sono stati creati vari *software* che implementavano tale sistema di controllo.

Di seguito ne sono riportati alcuni tra i più utilizzati:

- **Concurrent Versions System (CVS) :**

è un sistema software che implementa un sistema di controllo versione. CVS è divenuto popolare nel mondo del **software libero** ed è distribuito sotto la **GNU General Public License**.

# Distributed Version Control System

- **Subversion (SVN) :**

è un sistema di controllo versione per software, progettato dalla CollabNet Inc. con lo scopo di essere il **naturale successore di CVS**, oramai considerato superato.

- **Bazaar (bzd) :**

è un **software libero** per il controllo versione distribuito, ideato da Canonical Ltd, è **scritto in Python** e fa parte del progetto GNU.

- **BitKeeper :**

è un software di controllo di versione distribuito, prodotto da BitMover Inc, originariamente era un **software proprietario**, ma dal 2016 è diventato **open source**.

# Distributed Version Control System

- **Mercurial :**

è un **software multiplatforma** di controllo di versione distribuito creato da Matt Mackall e disponibile sotto GNU General Public License 2.0.

- **Git :**

è un software di controllo versione distribuito utilizzabile da interfaccia a riga di comando, creato da Linus Torvalds.

# Da dove nasce GitHub...

Nel 2005 Linus Torvalds creò **Git**, che nello slang americano significa “*idiota*”

Git nacque per essere un **semplice strumento** per facilitare lo sviluppo del kernel Linux ed è diventato uno degli strumenti di controllo versione più diffusi.

Lo **sviluppo** di Git è iniziato dopo che molti sviluppatori del kernel di Linux sono stati costretti ad abbandonare il sistema proprietario **BitKeeper** poiché **non più disponibile gratuitamente**.

**Torvalds** voleva un sistema distribuito che potesse usare come BitKeeper, ma nessuno dei sistemi disponibili gratuitamente soddisfaceva i suoi bisogni, in particolare **il suo bisogno di velocità**.

# Da dove nasce GitHub...

## Caratteristiche di Git :

- Forte supporto allo sviluppo non lineare :  
Supporta **diramazioni** e **fusioni** (branching and merging) rapide e comode, comprende strumenti specifici per la visualizzazione e navigazione della **cronologia di sviluppo** non lineare.
- Sviluppo distribuito :  
Ogni sviluppatore può creare una **copia locale** dell'intera cronologia di sviluppo e le modifiche vengono copiate da una repository a un'altra. Queste **modifiche** vengono **importate** come diramazioni aggiuntive di sviluppo, e possono essere **fuse** allo stesso modo di una diramazione sviluppata localmente.

# Da dove nasce GitHub...

- *I repository possono essere pubblicati facilmente :*  
tramite HTTP, FTP, ssh, rsync, ha anche un'emulazione del server CVS, che consente di usare gli esistenti client CVS e plugin per IDE per accedere ai repository Git.
- *Gestione efficiente di grandi progetti :*  
Git è molto **veloce e scalabile**. È tipicamente un ordine di grandezza più veloce degli altri sistemi di controllo versione, e due ordini di grandezza più veloce per alcune operazioni.
- *Autenticazione crittografica della cronologia :*  
La **cronologia** viene memorizzata in modo tale che il nome di una revisione particolare ("*commit*") **dipende dalla completa cronologia di sviluppo** che conduce a tale commit. Una volta pubblicata non è più possibile cambiare le vecchie versioni senza che ciò venga notato.



# Da dove nasce GitHub...

**GitHub** (nato nel 2008) è un **servizio di hosting** per progetti software.

Il nome "GitHub" deriva dal fatto che GitHub è una **implementazione** dello strumento di controllo versione distribuito **Git**.

Il sito è principalmente **utilizzato dagli sviluppatori** che caricano il **codice sorgente** dei loro programmi e lo rendono scaricabile dagli utenti.

Questi ultimi possono **interagire** con lo sviluppatore **tramite** un sistema di **issue tracking, pull request** e commenti che permette di **migliorare il codice** della repository risolvendo bug o aggiungendo funzionalità.

Inoltre Github elabora dettagliate **pagine che riassumono** come gli sviluppatori lavorino sulle **varie versioni** dei repository.

# Comandi Git da Terminale

I seguenti comandi hanno la potenzialità di fare **interagire git** con il nostro profilo **github**:

1. `git config --global user.name "Severus91"`  
(configura il nome su cui congiungere le transazioni)
2. `git config --global user.email "severus.91@gmail.com"`  
(configura il nome su cui congiungere le transazioni)
3. `git config --global color.ui auto`  
(abilita la colorazione d'aiuto della linea di comando in output)

# Comandi Git da Terminale

Esploriamo i comandi d'aiuto:

- **~\$ git** : mostra la lista di comandi base che interagiscono dal terminale con git.
- **~\$ git help -a** : mostra una lista di tutti i comandi utilizzabili
- **~\$ git help *nome comando*** : mostra una guida dedicata al comando specificato
- **~\$ git help -g** : mostra una lista delle guide concettuali di git
- **~\$ git help *nome guida concettuale*** : mostra la documentazione relativa alla guida concettuale indicata

# Che cosa è una Repository

Una repository viene solitamente utilizzata per **organizzare** un singolo progetto.

Le repository possono contenere cartelle e file, immagini, video, fogli di calcolo e set di dati, ovvero, **tutto ciò di cui ha bisogno** il tuo progetto.

Si consiglia di includere un **README** o un file con **informazioni sul tuo progetto**. GitHub ti consente di aggiungerne uno nello stesso momento in cui crei la tua nuova repository.

La tua repository può essere un luogo in cui **archiviare** idee, risorse o anche condividere e discutere le cose con gli altri.

Offre anche altre opzioni comuni come un **file di licenza**.

# Permessi di visualizzazione e Licenze

GitHub nel momento in cui crei una repository ti dà la facoltà di renderla :

- Pubblica :  
Visibile e scaricabile in locale da chiunque, con la possibilità di proporre delle modifiche al progetto originario.
- Privata :  
Dove tu decidi a chi sarà visibile tale repository e chi di conseguenza potrà proporre dei commit. (Questo servizio è a pagamento)

# Permessi di visualizzazione e Licenze

Una volta decisa la modalità di pubblicazione GitHub ti dà anche la possibilità di scegliere quale Licenza applicare alla tua repository:

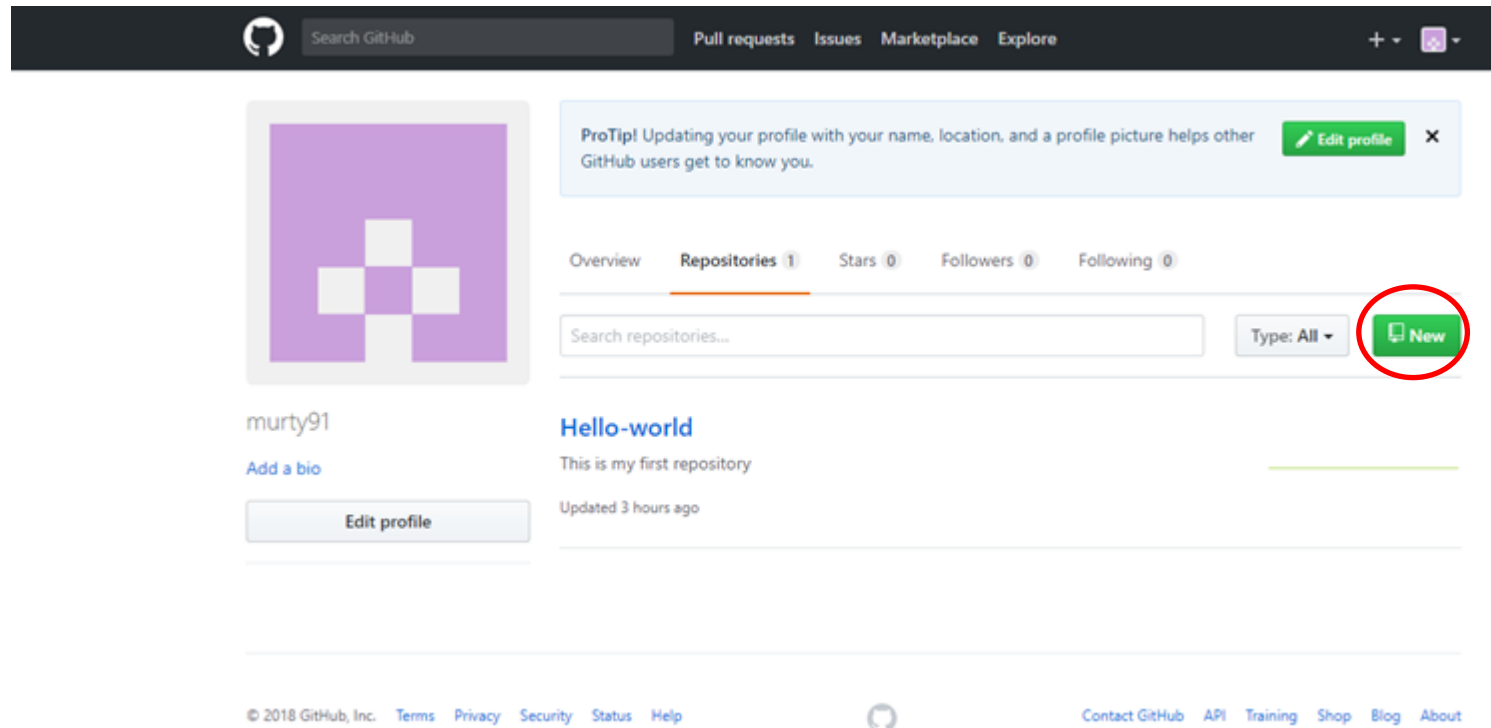
- *Apache License 2.0*
- *GNU General Public License v3.0*
- *MIT License*
- *Licenza BSD Semplificata/ Licenza FreeBSD (2 clausole)*
- *Licenza BSD modificata/Nuova licenza BSD (3 clausole)*
- *Eclipse Public License*
- *GNU Affero General Public License v3.0*

# Permessi di visualizzazione e Licenze

Una volta decisa la modalità di pubblicazione GitHub ti dà anche la possibilità di scegliere quale Licenza applicare alla tua repository:

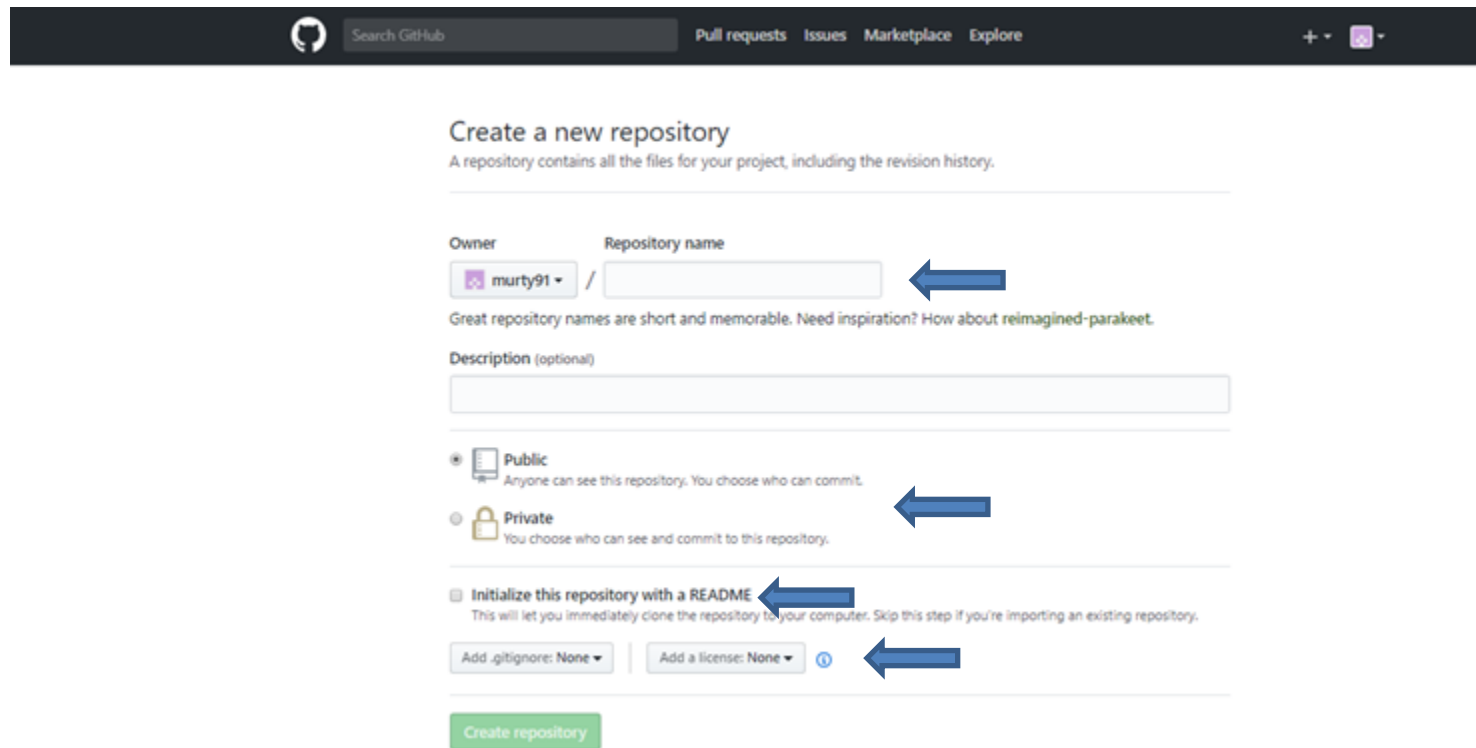
- *GNU General Public License v3.0*
- *GNU Lesser General Public License v3.0*
- *Mozilla Public License 2.0*
- *The Unlicense*

# Come creare una Repository dal Browser





# Come creare una Repository dal Browser



The screenshot shows the GitHub 'Create a new repository' page. At the top is a dark navigation bar with the GitHub logo, a search bar, and links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below this, the heading 'Create a new repository' is followed by a subtext: 'A repository contains all the files for your project, including the revision history.' The form contains several sections: 'Owner' with a dropdown menu showing 'murty91', 'Repository name' with an empty text box, 'Description (optional)' with a text area, and two radio button options for 'Public' (selected) and 'Private'. Below these are checkboxes for 'Initialize this repository with a README' and 'Add a license: None'. At the bottom is a green 'Create repository' button. Four blue arrows point to the 'Repository name' field, the 'Public' radio button, the 'Initialize this repository with a README' checkbox, and the 'Add a license: None' dropdown.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: murty91 /

Great repository names are short and memorable. Need inspiration? How about [reimagined-parakeet](#).

Description (optional):

☒ **Public**  
Anyone can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

☒ **Initialize this repository with a README**  
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None ⓘ

Create repository

# Git Ignore

È un file dove puoi inserire i **nomi** dei file che non vuoi vengano caricati e questi saranno **automaticamente ignorati**.

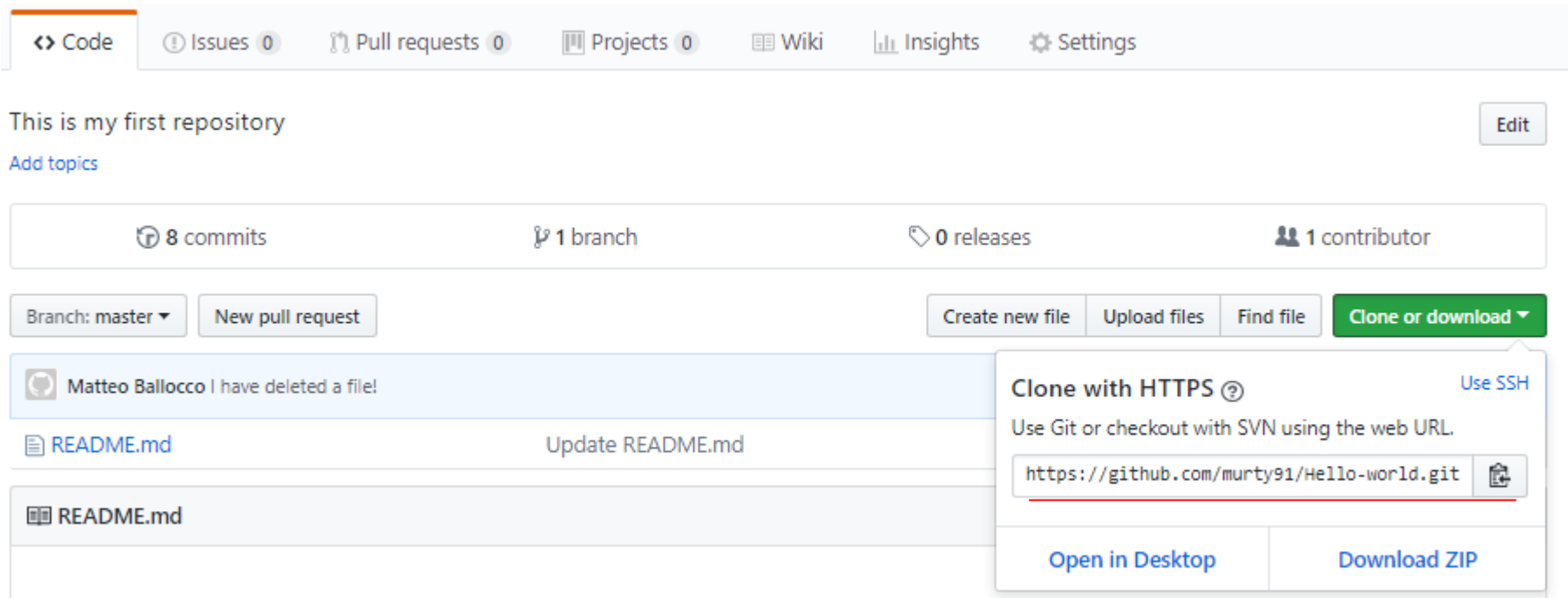
All'interno del file *.gitignore* si possono, ad esempio, ignorare tutti i file che finiscono con ~ (solitamente *file temporanei* di testo) con il comando : \* ~

Oppure si possono anche includere le directory **log**, **tmp** o **pid**, documenti anch'essi generati automaticamente.

L'impostazione di un file con il nome di *.gitignore* ignorerà i file in quella directory e nelle directory più profonde.

# Comandi Git da Locale

- Per importare la nostra repository da remoto a locale utilizziamo il seguente comando:
- **~\$ git clone *web URL***  
URL necessario lo troverete cliccando sul bottone verde “Clone or download” presente nella repository stessa.



The screenshot shows the GitHub interface for a repository named "This is my first repository". The top navigation bar includes links for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. Below the repository name, there are statistics: 8 commits, 1 branch, 0 releases, and 1 contributor. A dropdown menu for the "Clone or download" button is open, showing the "Clone with HTTPS" option selected. The dropdown also includes a link to "Use SSH", a description "Use Git or checkout with SVN using the web URL.", the repository URL "https://github.com/murty91/Hello-world.git", and buttons for "Open in Desktop" and "Download ZIP".

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

This is my first repository [Add topics](#) [Edit](#)

8 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file **Clone or download**

Matteo Ballocco I have deleted a file!

README.md Update README.md

README.md

**Clone with HTTPS** [Use SSH](#)

Use Git or checkout with SVN using the web URL.

<https://github.com/murty91/Hello-world.git>

[Open in Desktop](#) [Download ZIP](#)

# Comandi Git da Locale

Per inizializzare una cartella da locale possiamo usare il comando:

- **~\$ git init** *Nome\_cartella*

Per inizializzarla su una repository già esistente dobbiamo prima di tutto entrarci con il comando:

- **~\$ cd** *Nome\_repository*

Una volta eseguito procediamo con:

- **~/*Nome\_repository*\$ git init** *Nome\_cartella*

Eseguito il comando ci verrà restituito il seguente output:

*Initialized empty Git repository in  
/home/Utente/Nome\_repository/nome\_cartella/.git/*

# Che cosa è una Branch

Branching è il modo di lavorare su **diverse versioni** di un repository **contemporaneamente**.

Di default il tuo repository ha un ramo chiamato **master** che è considerato il ramo definitivo.

Usiamo i branch per sperimentare e **apportare modifiche** prima di affidarli al master.

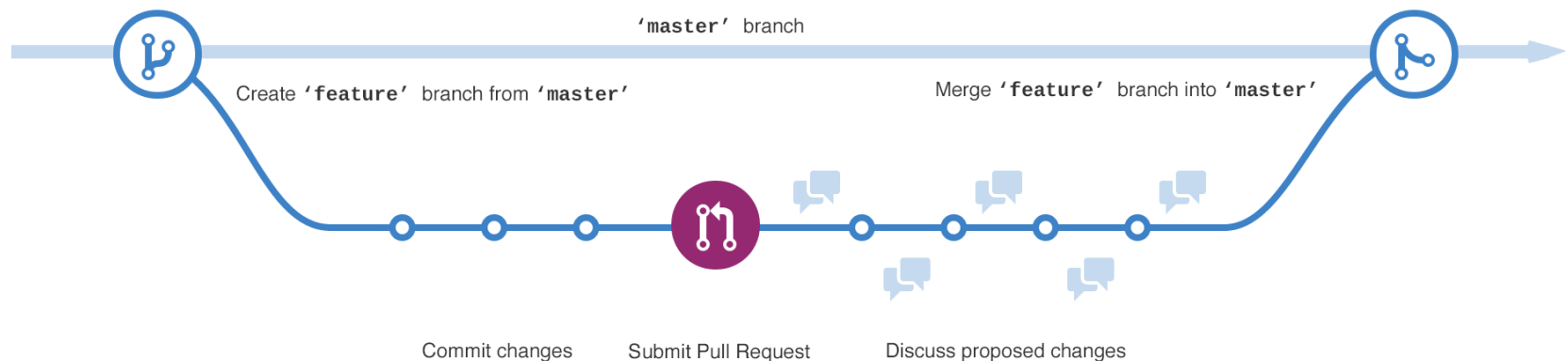
Quando crei un ramo fuori dal ramo principale, stai **creando una copia** del master come era **in quel momento**.

Se qualcun altro ha apportato **modifiche** al ramo principale **mentre stavi lavorando** sul tuo ramo, è possibile **inserire tali aggiornamenti**.

# Che cosa è una Branch

Questo diagramma mostra:

- Il ramo **master**
- Un nuovo ramo(**branch**) chiamato feature (perché stiamo facendo 'feature work' su questo ramo)
- Il **viaggio** che questa branch fa **prima di essere fuso** nel master



# Che cosa è una Branch

Immagina di salvare **diverse versioni** di un file.

Qualcosa di simile a:

- storia.txt
- storia-modificata.txt
- storia-modificata-rivista.txt

I **rami** raggiungono **obiettivi simili** nei repository.

Su GitHub i rami si utilizzano per mantenere le **correzioni di bug** e di funzioni separate dal ramo principale di produzione.

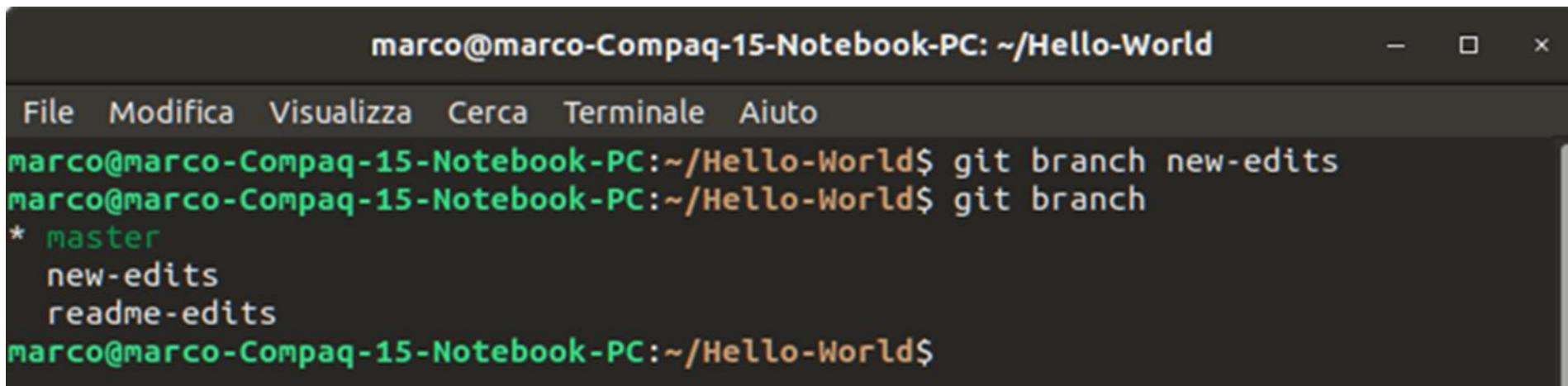
Quando un cambiamento è **pronto**, si può **unire** il ramo nel master.

# Come creare una Branch da Locale

Procediamo con la creazione di un nuovo branch:

1. **Entriamo** nella nostra repository ed inseriamo il comando:
  - `~/Hello-World$ git branch nome-branch`
2. Questo comando ci mostra i branch presenti sulla nostra repository :
  - `~/Hello-World$ git branch`

Evidenziata, in questo caso, in verde su quale branch ci troviamo, ora siamo sul master.

A screenshot of a terminal window titled "marco@marco-Compaq-15-Notebook-PC: ~/Hello-World". The terminal shows the execution of two git commands. The first command, "git branch new-edits", creates a new branch. The second command, "git branch", lists the current branches. The output shows three branches: "master" (highlighted with a green asterisk), "new-edits", and "readme-edits".

```
marco@marco-Compaq-15-Notebook-PC: ~/Hello-World
File Modifica Visualizza Cerca Terminale Aiuto
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git branch new-edits
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git branch
* master
  new-edits
  readme-edits
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$
```



# Operazioni sul Branch da Locale

Inoltriamoci nella comprensione delle potenzialità di git :

1. **Spostiamoci** sul branch appena creato digitando:
  - `~/Hello-World$ git checkout nome-branch`
2. **Apriamo** un file con un editor di testo per esempio “nano” in questo caso “*README.md*” che avevamo creato in precedenza e **modifichiamolo**  
La modifica del file non è mai definitiva finché non la **salviamo**.  
Il file modificato si trova infatti sulla nostra “*HEAD*” una sorta di memoria temporanea.
3. Mostriamo tutto quello che si trova sulla nostra HEAD eseguendo il comando:
  - `~/Hello-World$ git status`

# Operazioni sul Branch da Locale

```
marco@marco-Compaq-15-Notebook-PC: ~/Hello-World
File Modifica Visualizza Cerca Terminale Aiuto
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git checkout new-edits
M      README.md
Si è passati al branch 'new-edits'
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ nano README.md
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git status
Sul branch new-edits
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

# Operazioni sul Branch da Locale

4. Andiamo a **salvare le modifiche** del file con il comando:

- `~/Hello-World$ git add README.md`

Il comando “**git add .**” salva tutti file modificati.

5. Le modifiche ora sono **salvate** ma per renderle effettive bisogna fare una “commit” con il comando:

- `~/Hello-World$ git commit -m "spiegazione delle modifiche nel commit"`

6. Per **ogni primo commit** su di una branch Git ci chiederà di impostare il branch corrente da locale (sorgente) a remoto (destinazione) :

- `~/Hello-World$ git push --set-upstream origin "nome_branch«`

Una volta eseguito il comando per sincronizzare le commit su questa branch basterà un “**git push**”

# Operazioni sul Branch da Locale

```
marco@marco-Compaq-15-Notebook-PC: ~/Hello-World
File Modifica Visualizza Cerca Terminale Aiuto
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git add README.md
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git commit -m "I've made some example changes"
[new-edits 2a5abfa] I've made some example changes
1 file changed, 1 insertion(+), 1 deletion(-)
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git push
fatal: Il branch corrente new-edits non ha alcun branch upstream.
Per eseguire il push del branch corrente ed impostare remote come upstream, usa

git push --set-upstream origin new-edits

marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git push --set-upstream origin new-edits
Username for 'https://github.com': MarcoEna
Password for 'https://MarcoEna@github.com':
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 318 bytes | 318.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/MarcoEna/Hello-World.git
* [new branch]      new-edits -> new-edits
Branch new-edits set up to track remote branch new-edits from origin.
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$
```

# Come creare una Branch dal Browser

1. Vai nel tuo nuovo **repository**.
2. Fare clic sul menu a discesa nella parte superiore dell'elenco dei file che dice **branch**: master.
3. Digitare un nome di ramo, ad es. **Modifica**, nella nuova casella di testo del ramo.
4. Seleziona la casella blu **Crea ramo** o premi "Invio" sulla tastiera.

# Come creare una Branch da Browser

The screenshot shows the GitHub interface for a repository named "Esempio-" by user "murty91". The repository has 1 commit, 1 branch, 0 releases, and 1 contributor. The "Code" tab is selected, and a "Switch branches/tags" modal is open. The modal has a search bar with the text "Find or create a branch..." and a list of branches with "master" selected. A blue arrow points from the search bar to the "Initial commit" entry in the commit history. The commit history shows the "Initial commit" 2 days ago. The repository description is "Questo è un esempio".

murty91 / Esempio-

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Questo è un esempio [Add topics](#) [Edit](#)

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request

Create new file Upload files Find file Clone or download

Switch branches/tags

Find or create a branch...

Branches Tags

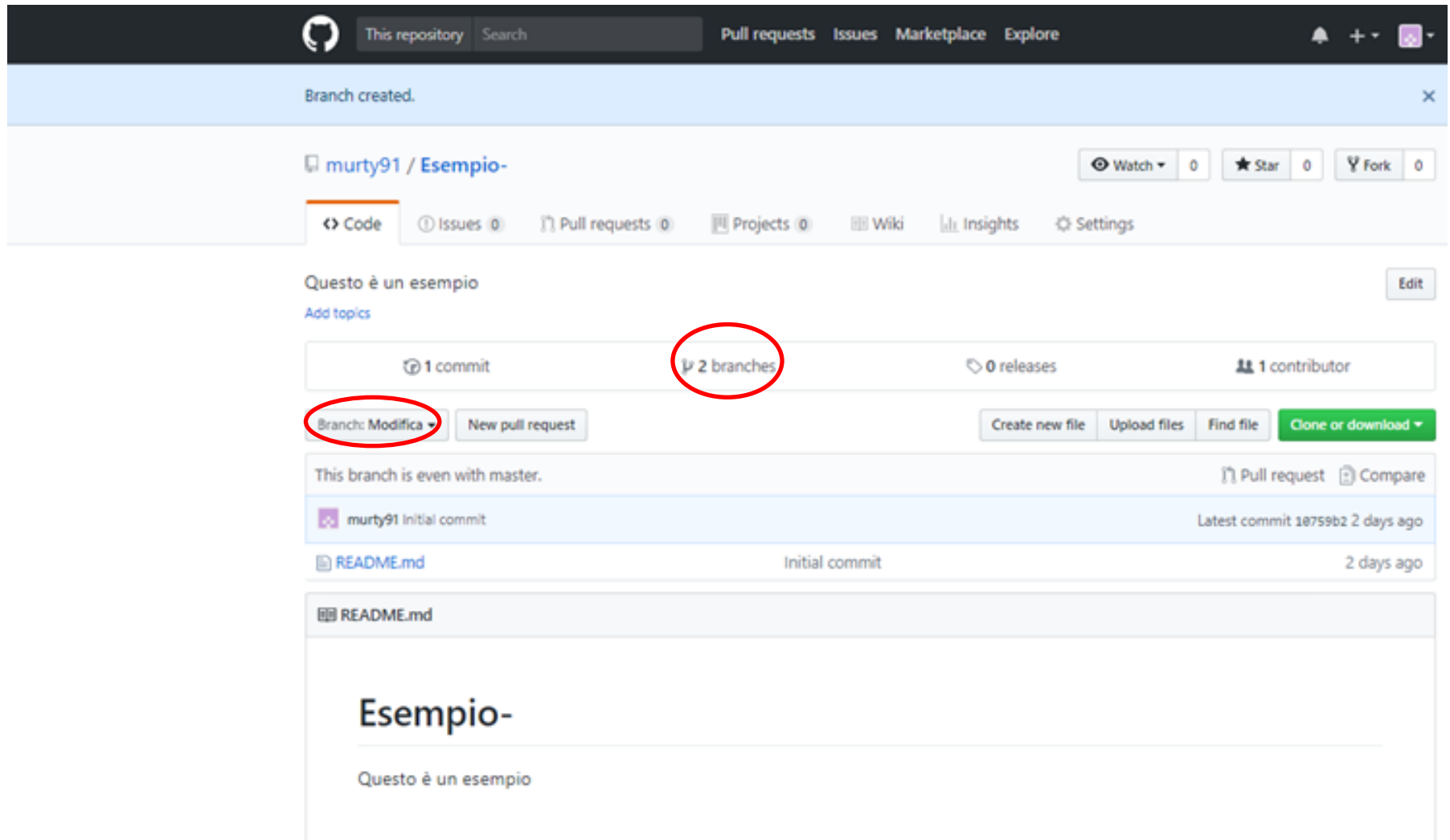
✓ master

Initial commit 2 days ago

Esempio-

Questo è un esempio

# Come creare una Branch dal Browser



The screenshot shows the GitHub interface for a repository named "Esempio-" by user "murty91". At the top, a blue banner indicates "Branch created." Below the repository name, there are tabs for "Code", "Issues", "Pull requests", "Projects", "Wiki", "Insights", and "Settings". The "Code" tab is selected. The repository statistics show "1 commit", "2 branches" (circled in red), "0 releases", and "1 contributor". Below the statistics, there is a dropdown menu labeled "Branch: Modifica" (circled in red) and a "New pull request" button. To the right of these are buttons for "Create new file", "Upload files", "Find file", and a green "Clone or download" button. The main content area shows the commit history, with the "Initial commit" by "murty91" listed. Below the commit history, the "README.md" file is displayed, containing the text "Esempio-" and "Questo è un esempio".

Branch created.

murty91 / Esempio-

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Questo è un esempio [Edit](#)

[Add topics](#)

1 commit 2 branches 0 releases 1 contributor

Branch: Modifica New pull request Create new file Upload files Find file Clone or download

This branch is even with master. [Pull request](#) [Compare](#)

murty91 Initial commit Latest commit 18759b2 2 days ago

README.md Initial commit 2 days ago

README.md

## Esempio-

Questo è un esempio

# Come modificare un file dal Browser

Ora sei nella schermata code nella tua **Branch** “Modifica”, che è una **copia del master**.

Qui si possono fare alcune **modifiche**.

Su GitHub, le **modifiche salvate** sono chiamate **commit**.

Ogni commit ha un **messaggio** di commit associato, che è una **descrizione** che spiega perché è stata apportata una particolare **modifica**.

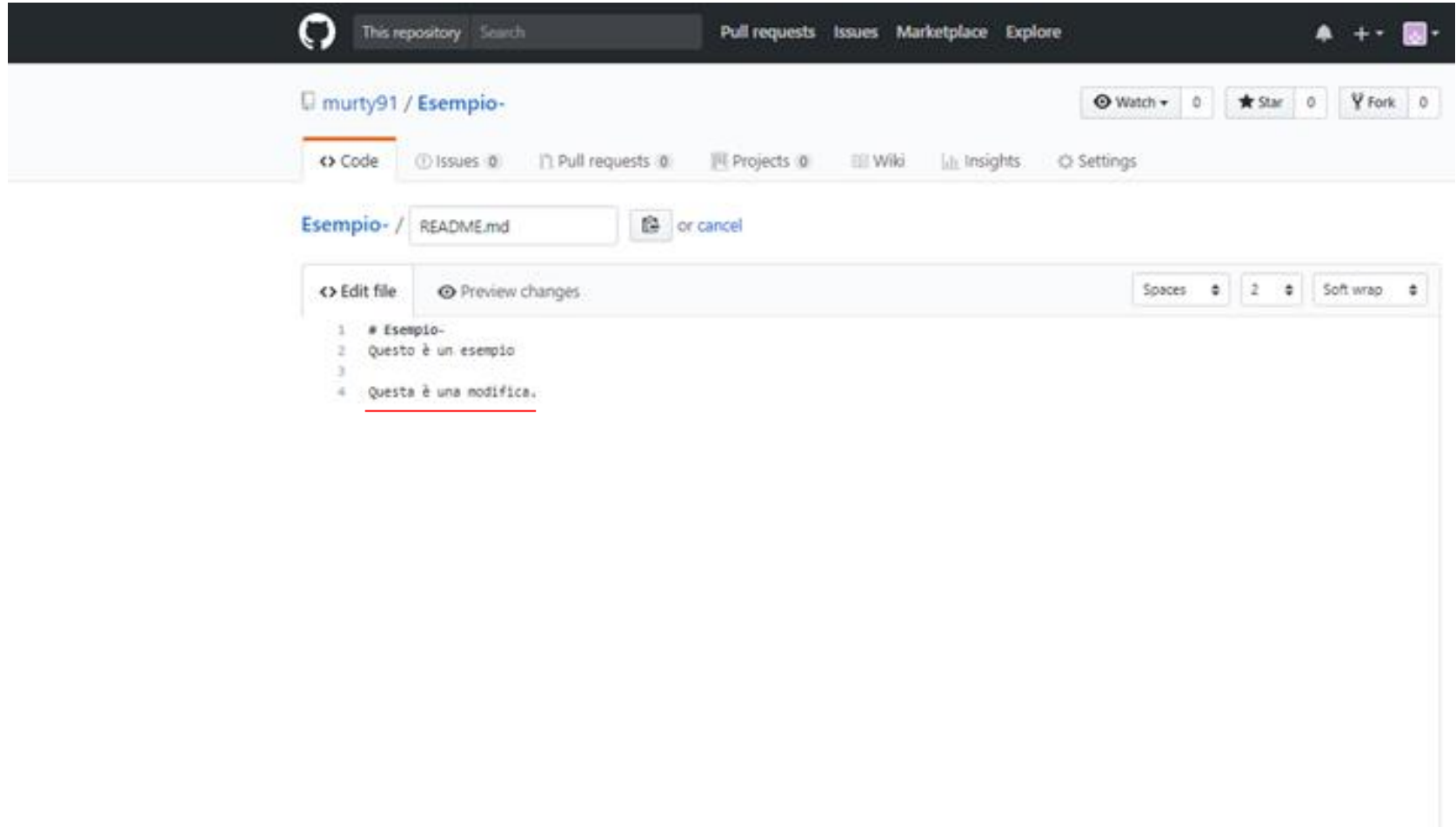
I messaggi di commit **catturano la cronologia** delle tue modifiche, così gli altri contributori possono capire **cosa hai fatto e perché**.



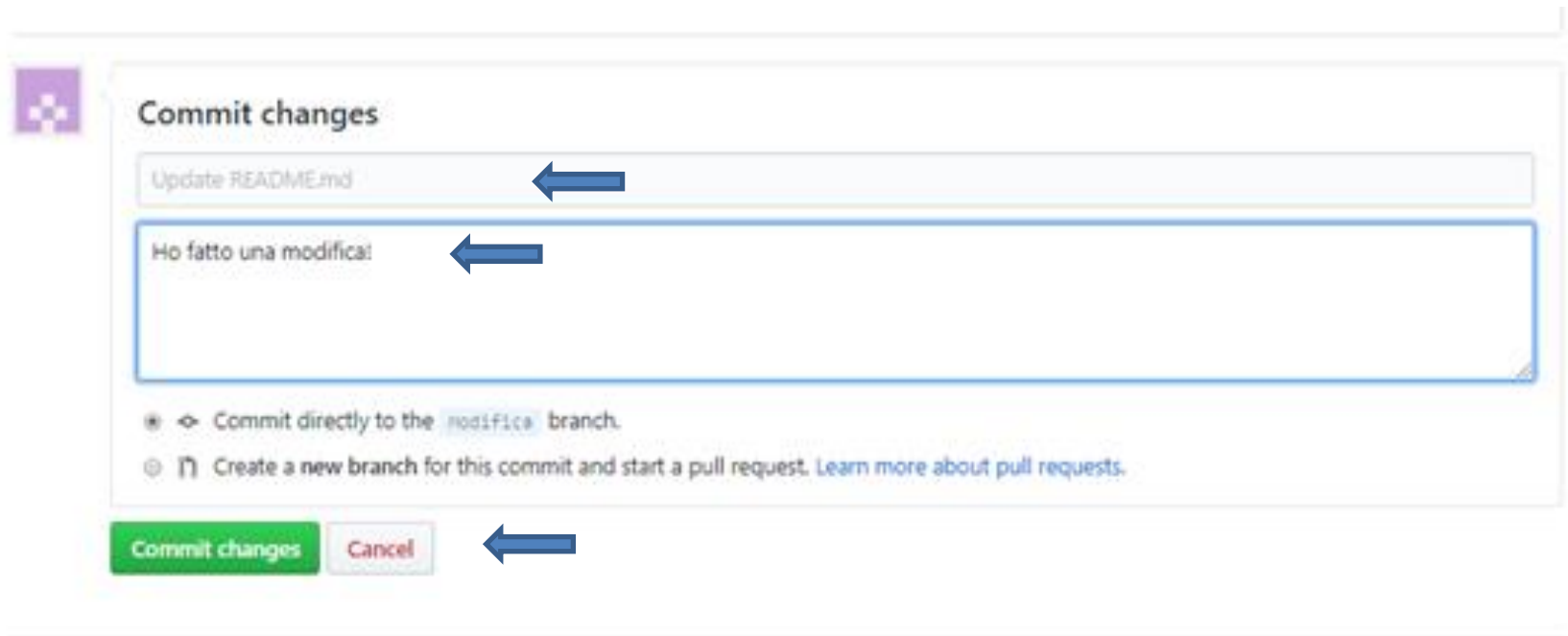
# Come modificare un file dal Browser


1. Fare clic sul file **README.md**.
2. Fai clic sull'icona a forma di **matita** nell'angolo in alto a destra della visualizzazione del file per modificarla.
3. Nell'editor, scrivi o **fai qualche modifica**.
4. Scrivi un **messaggio di commit** che descriva le tue modifiche.
5. Fai clic sul pulsante **Cambia modifiche**.

# Come modificare un file dal Browser



# Come modificare un file dal Browser



 **Commit changes**

Update README.md

Ho fatto una modifica!

☒ Commit directly to the `modifica` branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

**Commit changes** Cancel

# Come modificare un file dal Browser

Queste **modifiche** verranno apportate **solo al file README** sulla Branch **“Modifica”**.

Quindi ora questo ramo contiene **contenuti diversi dal master**.

# Cosa è una Pull Request

Ora che hai delle modifiche in un ramo fuori dal master, puoi aprire una **Pull Request**.

La pull request è il cuore della **collaborazione** su GitHub.

Quando apri una pull request , stai **proponendo le tue modifiche** e richiedi che qualcuno riveda e aggiunga il tuo contributo e lo unisca nel loro ramo.

Le pull request **mostrano le differenze tra i contenuti** di entrambi i rami.

Le modifiche, le aggiunte e le sottrazioni sono mostrate **in verde e rosso**.

# Cosa è una Pull Request

Non appena esegui un **commit**, puoi aprire una **pull request** e avviare una **discussione**, anche prima che il codice sia finito.

Usando il sistema **@mention** nel tuo messaggio di richiesta di pull, puoi chiedere feedback a persone o team specifici.

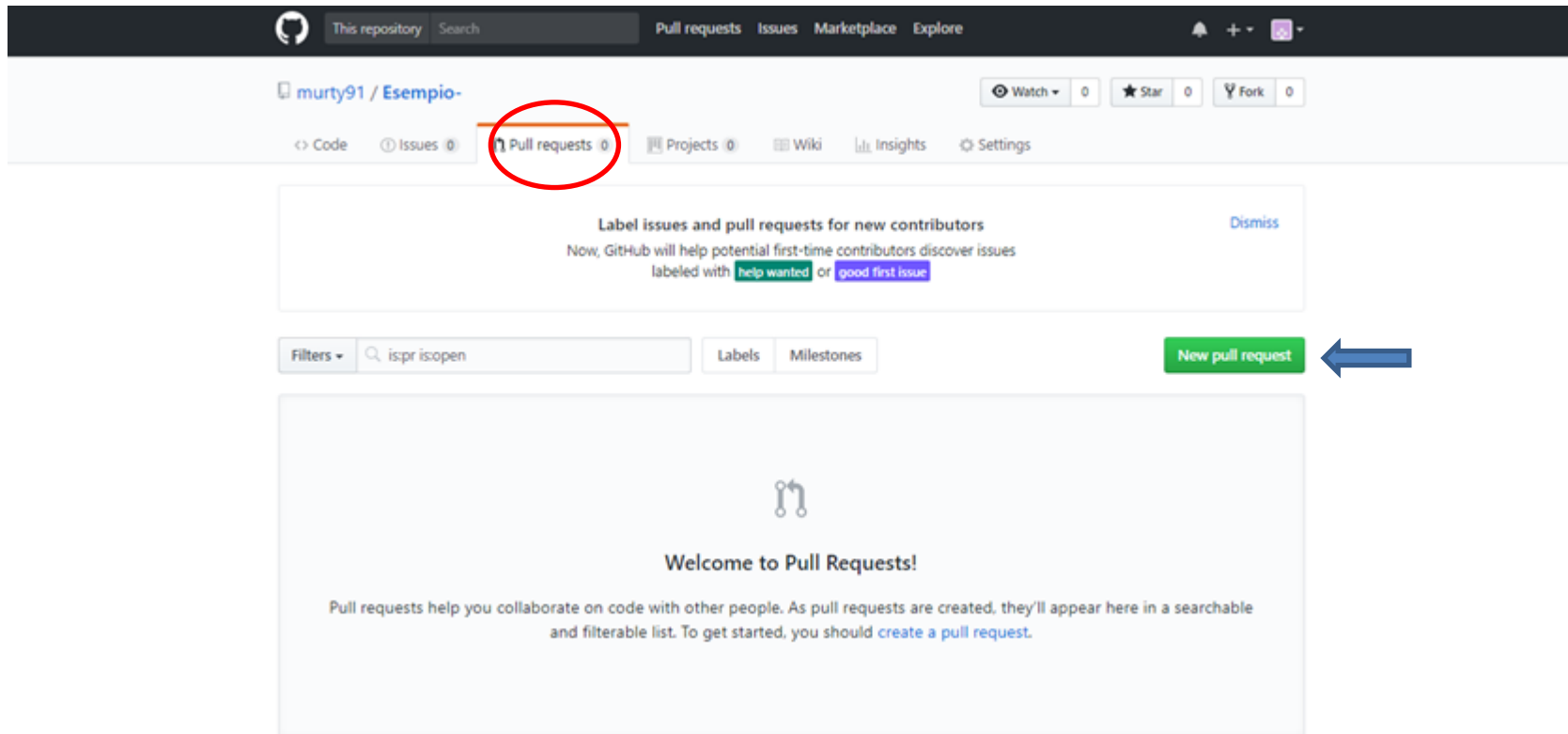
Puoi persino aprire le richieste di pull nel tuo repository e **unirle tu stesso**.

È un **ottimo** modo **per imparare** GitHub Flow prima di lavorare su progetti più grandi.

# Come creare una Pull Request dal Browser

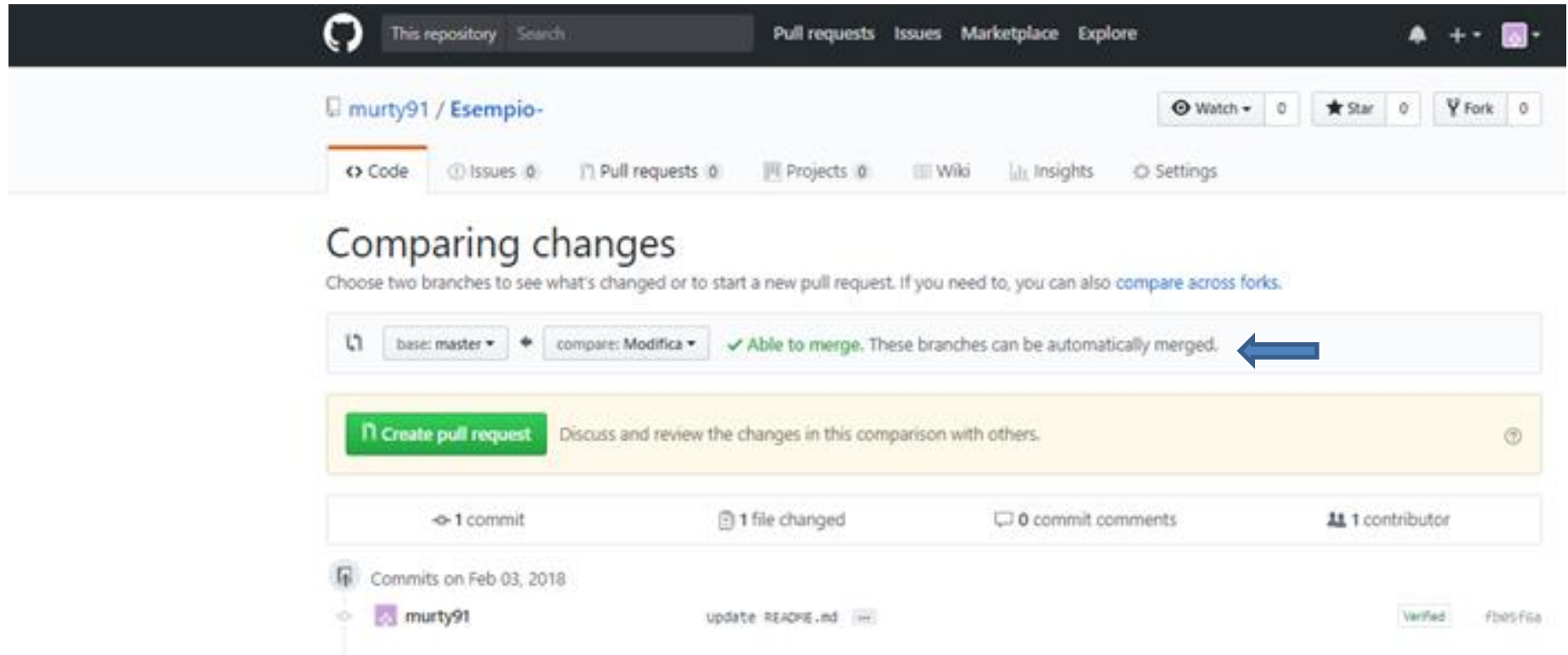
1. Fare clic sulla scheda **Pull Requests**, quindi dalla pagina Richiesta di pull, fare clic sul pulsante verde New pull request.
2. Nella casella **Comparing changes**, seleziona il ramo che hai creato, “Modifica”, e confrontalo con il master (l'originale).
3. **Controlla le modifiche** al fondo della pagina , assicurati che siano ciò che vuoi inviare.
4. Quando sei soddisfatto del fatto che queste sono le modifiche che desideri inviare, fai clic sul pulsante verde **Crea Pull Request**.
5. Dai alla tua Request Pull un **titolo** e scrivi una **breve descrizione** delle tue modifiche
6. Quando hai finito con il tuo messaggio, fai clic su **Crea pull request**.

# Come creare una Pull Request dal Browser





# Come creare una Pull Request dal Browser



The screenshot shows the GitHub interface for a repository named 'murty91 / Esempio-'. The top navigation bar includes links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the repository name, there are tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The 'Pull requests' tab is selected, and the page title is 'Comparing changes'. A message states: 'Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).' Below this, a comparison bar shows 'base: master' and 'compare: Modifica'. To the right of the comparison bar, a green checkmark and the text 'Able to merge. These branches can be automatically merged.' are displayed, with a blue arrow pointing to this text. Below the comparison bar, there is a green button labeled 'Create pull request' and a text prompt: 'Discuss and review the changes in this comparison with others.' At the bottom, a summary bar indicates '1 commit', '1 file changed', '0 commit comments', and '1 contributor'. The commit history shows a commit by 'murty91' on Feb 03, 2018, with the message 'update README.md'.

murty91 / Esempio-

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

## Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base: master + compare: Modifica ✓ Able to merge. These branches can be automatically merged.

Create pull request Discuss and review the changes in this comparison with others.

1 commit 1 file changed 0 commit comments 1 contributor

Commits on Feb 03, 2018

murty91 update README.md Verified fb05f5a

# Come creare una Pull Request dal Browser

Showing 1 changed file with 2 additions and 0 deletions.

Unified Split

2 README.md



View



... @@ -1,2 +1,4 @@

1 1 # Esempio-

2 2 questo è un esempio

3 +  
4 +Questa è una modifica.



No commit comments for this range

# Come creare una Pull Request dal Browser

The screenshot shows the GitHub interface for creating a pull request. At the top, the repository name is 'murty91 / Esempio-'. Below the repository name, there are tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The 'Pull requests' tab is selected. The main heading is 'Open a pull request', followed by a subtext: 'Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).' Below this, there is a comparison bar showing 'base: master' and 'compare: Modifica', with a green checkmark and the text 'Able to merge. These branches can be automatically merged.' The main content area is titled 'Update README.md' and has a 'Write' tab selected. The text input area contains the text 'Ho fatto una modifica'. Below the text input, there is a note: 'Attach files by dragging & dropping, selecting them, or pasting from the clipboard.' At the bottom right of the main content area, there is a green button labeled 'Create pull request', which is circled in red. On the right side of the interface, there are sections for 'Reviewers', 'Assignees', 'Labels', 'Projects', and 'Milestone', each with a dropdown menu and a gear icon.

murty91 / Esempio-

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

## Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: master + compare: Modifica ✓ Able to merge. These branches can be automatically merged.

Update README.md

Write Preview

Ho fatto una modifica

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Create pull request

Reviewers  
No reviews—request one

Assignees  
No one—assign yourself

Labels  
None yet

Projects  
None yet

Milestone  
No milestone

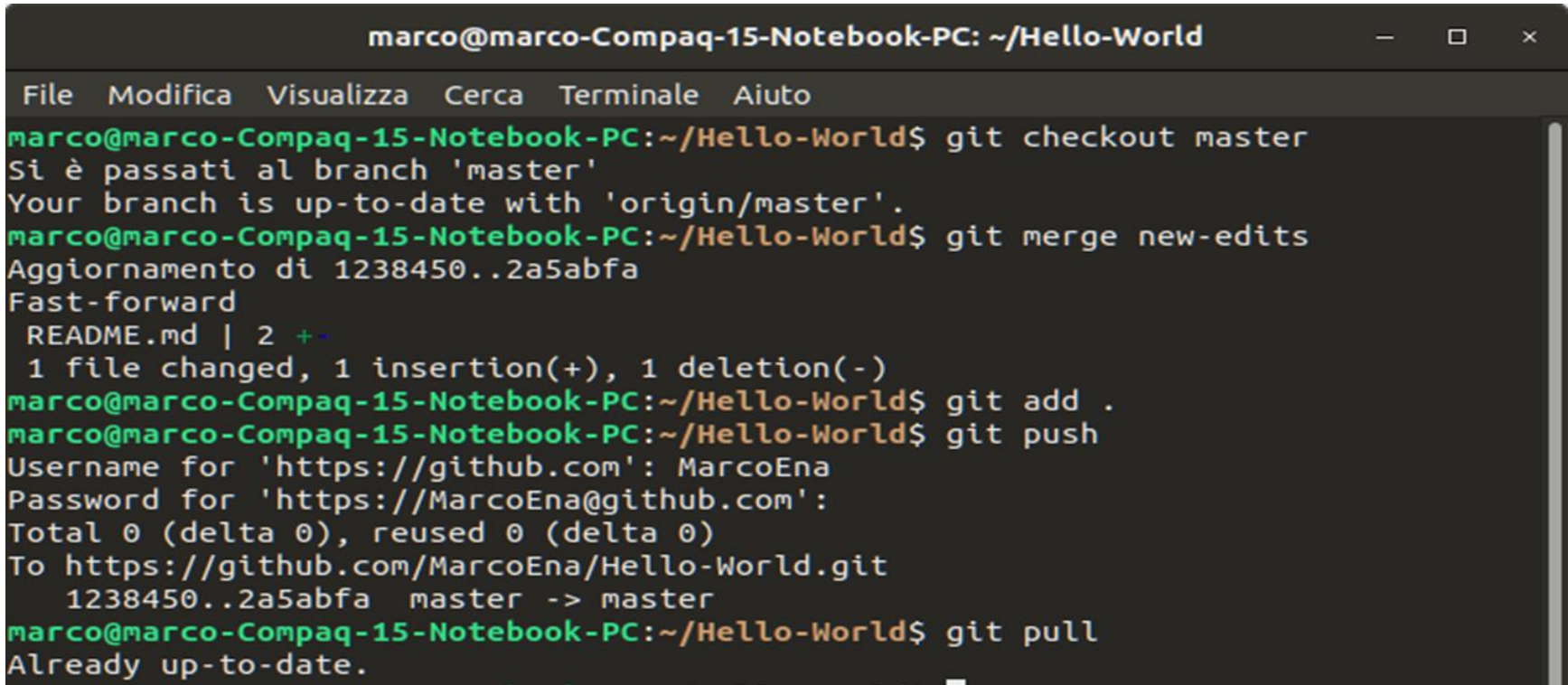
# Come effettuare una Merge da Locale

Congiungiamo le commit della nostra branch con il master:

1. Ritorniamo sul master con “***git checkout master***”
2. Utilizziamo la funzione di “**merge**” (fusione) con il comando:
  - ***~/Hello-World\$ git merge nome\_branch***
3. Il seguente comando sarà un “ git add . ”
4. Sincronizziamo la merge con un “git push”

Così facendo abbiamo **sincronizzato** e **fuso** i due branch se volessimo poi riportare dei cambiamenti effettuati da remoto ci basterebbe usare il comando “***git pull***”

# Come effettuare una Merge da Locale

A screenshot of a terminal window titled "marco@marco-Compaq-15-Notebook-PC: ~/Hello-World". The window has a menu bar with "File", "Modifica", "Visualizza", "Cerca", "Terminale", and "Aiuto". The terminal shows the following sequence of commands and output:

```
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git checkout master
Si è passati al branch 'master'
Your branch is up-to-date with 'origin/master'.
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git merge new-edits
Aggiornamento di 1238450..2a5abfa
Fast-forward
 README.md | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git add .
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git push
Username for 'https://github.com': MarcoEna
Password for 'https://MarcoEna@github.com':
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/MarcoEna/Hello-World.git
 1238450..2a5abfa  master -> master
marco@marco-Compaq-15-Notebook-PC:~/Hello-World$ git pull
Already up-to-date.
```

# Come fare una Merge dal Browser

In questo ultimo passaggio, è il momento di mettere insieme le tue modifiche - **unendo** il tuo ramo “Modifica” al ramo principale.

1. Fai clic sul pulsante verde **Merge Pull Request** per unire le modifiche in master.
2. Fai clic su **Confirm Merge**.
3. Vai avanti ed **elimina il ramo**, poiché le sue modifiche sono state incorporate, con il pulsante Delete Branch nella casella viola.

# Come fare una Merge dal Browser

The screenshot shows a GitHub pull request interface for the repository 'murty91 / Esempio-'. The pull request is titled 'Update README.md #1' and is from the 'Modifica' branch to the 'master' branch. The pull request is open and has 1 commit. The interface includes a navigation bar with links to 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. A comment from 'murty91' is visible, stating 'Ho fatto una modifica!'. Below the comment, a green box indicates that the branch has no conflicts with the base branch, allowing for an automatic merge. The 'Merge pull request' button is highlighted with a red circle. The right sidebar shows settings for reviewers, assignees, labels, projects, and milestones.

murty91 / Esempio-

Update README.md #1

Open murty91 wants to merge 1 commit into master from Modifica

Conversation 0 Commits 1 Files changed 1 +2 -0

murty91 commented 3 days ago

Ho fatto una modifica!

Update README.md Verified fb05f6a

Add more commits by pushing to the Modifica branch on murty91/Esempio-.

This branch has no conflicts with the base branch  
Merging can be performed automatically.

Merge pull request You can also open this in GitHub Desktop or view command line instructions.

Reviewers: No reviews—request one

Assignees: No one—assign yourself

Labels: None yet

Projects: None yet

Milestone: No milestone

# Come fare una Merge dal Browser

The screenshot displays a GitHub pull request for the repository 'murty91 / Esempio-'. The title of the pull request is 'Update README.md #1'. A comment from the user 'murty91' states 'Ho fatto una modifica!' (I made a modification!). The pull request details show 'Update README.md' as the commit. At the bottom, the 'Merge pull request #1 from murty91/Modifica' dialog is open, with the 'Confirm merge' button highlighted by a red circle. A blue arrow points to this button. The right sidebar contains configuration options for reviewers, assignees, labels, projects, and milestones.

This repository Search Pull requests Issues Marketplace Explore

murty91 / Esempio- Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 1 Projects 0 Wiki Insights Settings

## Update README.md #1

Open murty91 wants to merge 1 commit into master from Modifica

Conversation 0 Commits 1 Files changed 1 +2 -0

murty91 commented 3 days ago

Owner + 👤 ✎

Ho fatto una modifica! ←

Update README.md Verified fbef5fa

Add more commits by pushing to the Modifica branch on murty91/Esempio-.

Merge pull request #1 from murty91/Modifica

Update README.md ←

Confirm merge Cancel

Reviewers  
No reviews—request one

Assignees  
No one—assign yourself

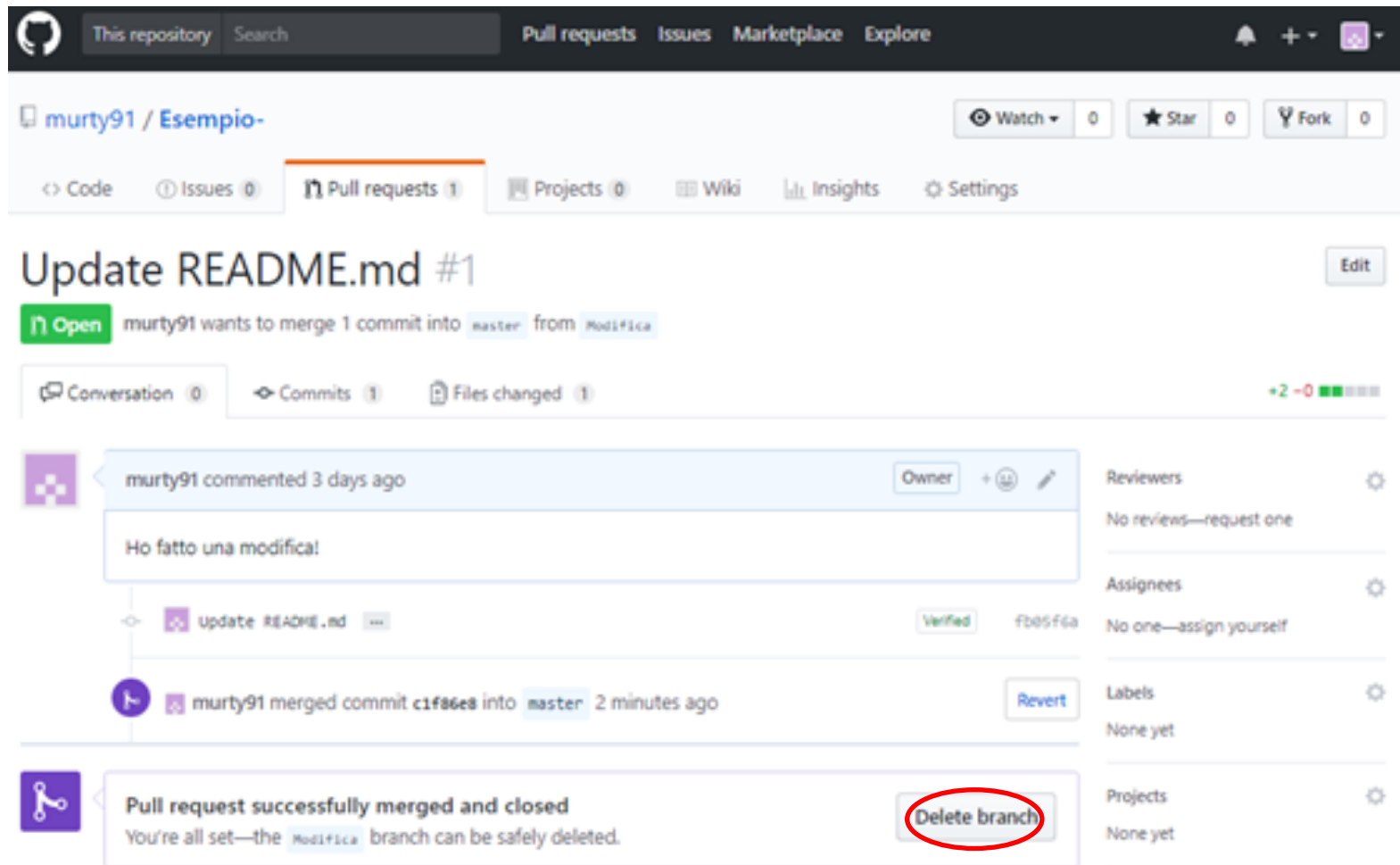
Labels  
None yet

Projects  
None yet

Milestone  
No milestone



# Come fare una Merge dal Browser



The screenshot shows a GitHub interface for a repository named "murty91 / Esempio-". The "Pull requests" tab is selected, displaying a pull request titled "Update README.md #1". The pull request was created by "murty91" and is targeting the "master" branch from a branch named "Modifica". The status is "Closed", and it shows "1 commit" and "1 file changed".

Below the pull request details, there is a list of comments and actions. The most recent comment from "murty91" states: "Ho fatto una modifica!". Below this, a commit "Update README.md" is shown, followed by a message: "murty91 merged commit c1f86e8 into master 2 minutes ago".

At the bottom of the pull request, a message indicates: "Pull request successfully merged and closed. You're all set—the Modifica branch can be safely deleted." A button labeled "Delete branch" is circled in red, indicating the next step in the process.

On the right side of the pull request, there are sections for "Reviewers", "Assignees", "Labels", and "Projects", each with a gear icon for settings.

# Problema nel Merge: Conflitto

## Gestione dei conflitti :

Durante l'operazione di merge si possono verificare dei conflitti nel caso in cui siano stati **modificati** gli **stessi file** e le **stesse righe di codice** su branch differenti.

Ovviamente Git non può decidere da solo quale modifica deve applicare, quindi bisognerà risolvere il conflitto in **maniera manuale** analizzando le differenze.

Facciamo un' esempio semplice...

# Problema nel Merge: Conflitto

1. **Creiamo** nel nostro precedente repository **due nuove branch** ( “Prova conflitto 1” e “Prova conflitto 2”).
2. Modifichiamo in entrambe le branch la **medesima riga** con frasi differenti.
3. Facciamo una **Comparing changes** tra uno dei due rami ed il master e poi tra le 2 branch
4. Proviamo a fare una **Pull Request** per una Merge tra le **2 branch**.
5. Clicchiamo su **Resolve conflicts** e cerchiamo di risolvere il conflitto.

# Problema nel Merge: Conflitto

The screenshot shows a GitHub repository page for 'murty91 / Esempio-'. The repository has 3 commits, 3 branches, 0 releases, and 1 contributor. The 'Code' tab is selected, showing a file named 'modifica'. The file has two commits: 'Latest commit c1f86e8 an hour ago' and 'Update README.md 3 days ago'. A 'Switch branches/tags' dialog box is open, showing a list of branches: 'Prova-conflitto-1' (selected), 'Prova-conflitto-2', and 'master' (checked). The dialog box also has a search bar and tabs for 'Branches' and 'Tags'. Below the dialog box, the text 'Questo è un esempio' and 'Questa è una modifica.' is visible.

This repository

Search

Pull requests Issues Marketplace Explore

murty91 / Esempio-

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Questo è un esempio [Add topics](#) [Edit](#)

3 commits 3 branches 0 releases 1 contributor

Branch: master New pull request

Create new file Upload files Find file Clone or download

modifica Latest commit c1f86e8 an hour ago

Update README.md 3 days ago

Switch branches/tags

Find or create a branch...

Branches Tags

Prova-conflitto-1

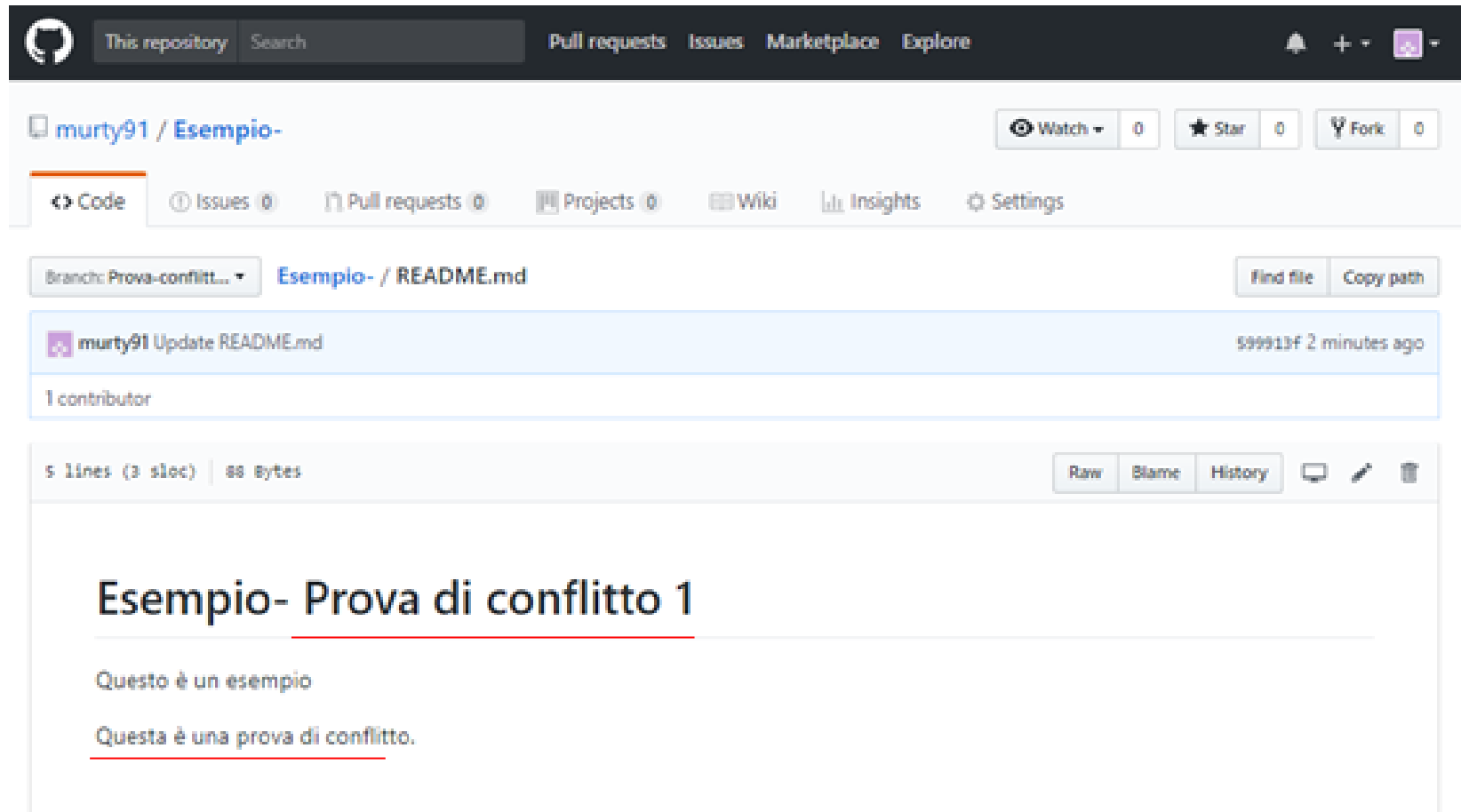
Prova-conflitto-2

✓ master

Questo è un esempio

Questa è una modifica.

# Problema nel Merge: Conflitto



The screenshot shows a GitHub repository page for 'murty91 / Eempio-'. The repository has 0 Watchers, 0 Stars, and 0 Forks. The 'Code' tab is selected, showing the 'Eempio- / README.md' file. The file is 5 lines (3 sloc) and 88 Bytes. The commit history shows a recent update by 'murty91' titled 'Update README.md' with commit hash '599913f' 2 minutes ago. The file content displays a merge conflict in the README.md file. The conflict is indicated by red underlines and the text 'Questo è una prova di conflitto.' is repeated. The conflict is located in the README.md file, specifically in the section titled 'Eempio- Prova di conflitto 1'.

murty91 / Eempio-

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Branch: Prova-conflict... Eempio- / README.md Find file Copy path

murty91 Update README.md 599913f 2 minutes ago

1 contributor

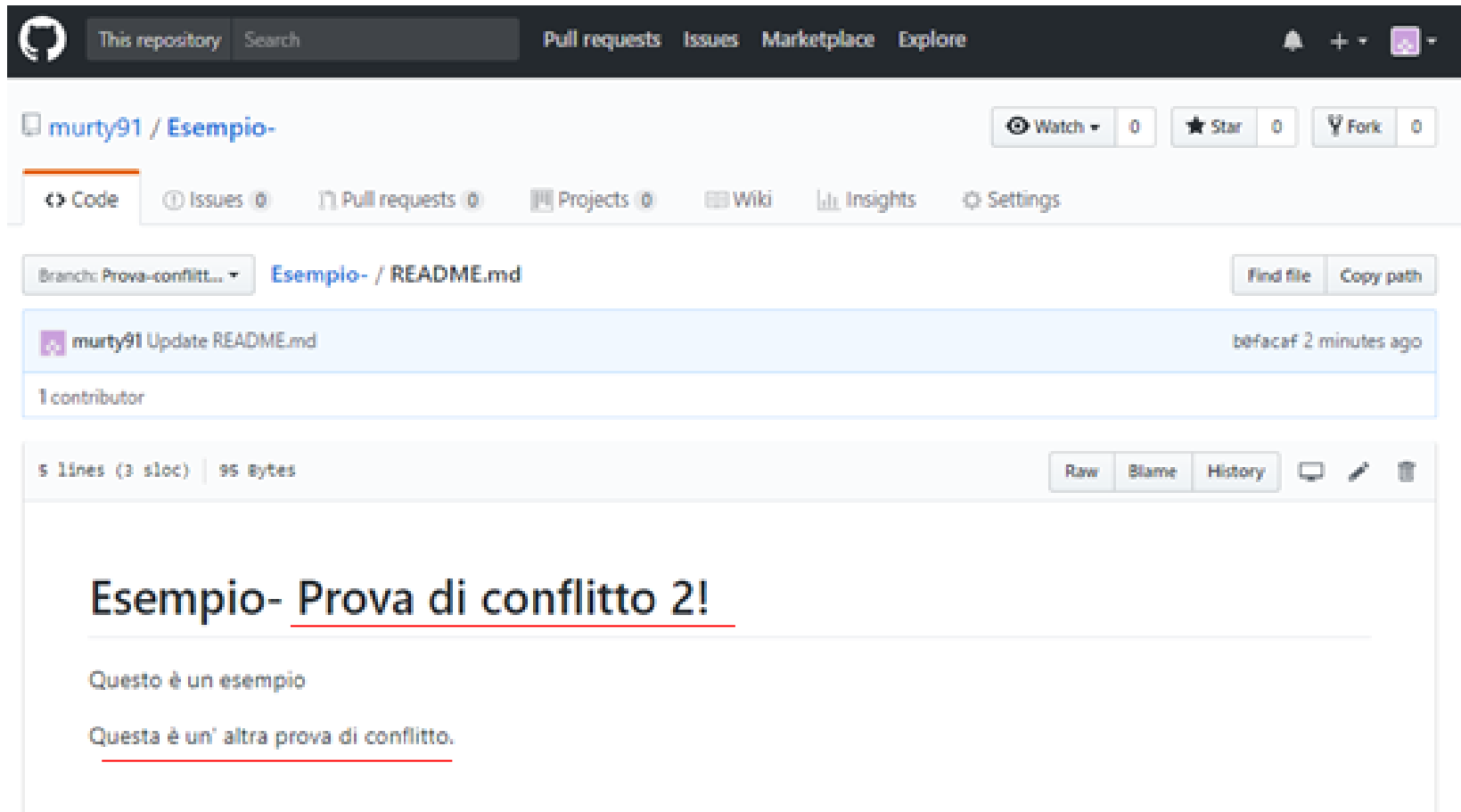
5 lines (3 sloc) 88 Bytes Raw Blame History

## Eempio- Prova di conflitto 1

Questo è un esempio

Questa è una prova di conflitto.

# Problema nel Merge: Conflitto



The screenshot shows the GitHub interface for a repository named "Esempio-" by user "murty91". The repository has 0 watches, 0 stars, and 0 forks. The "Code" tab is selected, showing the file "README.md" from the "Prova-conflict..." branch. The file was updated by "murty91" 2 minutes ago. The file content is as follows:

```
5 lines (3 sloc) | 95 bytes
```

**Esempio- Prova di conflitto 2!**

Questo è un esempio

Questa è un' altra prova di conflitto.

The file content shows a merge conflict. The first line is "Esempio- Prova di conflitto 2!". The second line is "Questo è un esempio". The third line is "Questa è un' altra prova di conflitto.". The underline on the third line indicates a conflict with a previous version of the file.

# Problema nel Merge: Conflitto

The screenshot shows a GitHub pull request interface. At the top, there's a navigation bar with links for Code, Issues, Pull requests, Projects, Wiki, Insights, and Settings. Below this, the title "Comparing changes" is displayed, followed by a subtitle: "Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#)." The main comparison area shows "base: master" and "compare: Prova-conflitto-2". A green checkmark indicates "Able to merge. These branches can be automatically merged." A blue arrow points to this status. Below the comparison area, there's a yellow box with a green "Create pull request" button and the text "Discuss and review the changes in this comparison with others." The pull request summary shows "1 commit", "1 file changed", "0 commit comments", and "1 contributor". The commit history shows a commit by "murty91" on Feb 06, 2018, titled "Update README.md". The file diff for README.md is shown, indicating 2 additions and 2 deletions. The diff content is as follows:

```
4 README.md
... @@ -1,4 +1,4 @@
1 -# Esempio-
1 +# Esempio- Prova di conflitto 2!
2   Questo è un esempio
3
4 -Questa è una modifica.
4 +Questa è un' altra prova di conflitto.
```

A blue arrow points to the new line in the diff: "+# Esempio- Prova di conflitto 2!".

# Problema nel Merge: Conflitto

The screenshot shows a GitHub interface for comparing changes between two branches. At the top, navigation tabs include Code, Issues, Pull requests, Projects, Wiki, Insights, and Settings. The main heading is "Comparing changes", with a subtext explaining the purpose: "Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#)." Below this, a comparison bar shows "base: Prova-conflitto-1" and "compare: Prova-conflitto-2". A red error message states: "X Can't automatically merge. Don't worry, you can still create the pull request." A blue arrow points to this message. Below the comparison bar is a yellow box with a green "Create pull request" button and the text "Discuss and review the changes in this comparison with others." Further down, a summary bar indicates "1 commit", "1 file changed", "0 commit comments", and "1 contributor". The commit history shows a commit by "murty91" on Feb 06, 2018, titled "Update README.md". Below the commit, it says "Showing 1 changed file with 2 additions and 2 deletions." and provides "Unified" and "Split" view options. The file diff for "README.md" is shown in a code block. It highlights four lines of changes: a deletion of a comment, an addition of a new comment, a deletion of a line of text, and an addition of a new line of text. A blue arrow points to the diff area.

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

## Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base: Prova-conflitto-1 compare: Prova-conflitto-2 X Can't automatically merge. Don't worry, you can still create the pull request.

Create pull request Discuss and review the changes in this comparison with others.

1 commit 1 file changed 0 commit comments 1 contributor

Commits on Feb 06, 2018

Update README.md Verified befacaf

Showing 1 changed file with 2 additions and 2 deletions. Unified Split

```
4 README.md
... -1,4 +1,4 @@
1 -# Esempio-
1 +# Esempio- Prova di conflitto 2!
2 2 Questo è un esempio
3 3
4 -Questa è una modifica.
4 +Questa è un' altra prova di conflitto.
```



# Problema nel Merge: Conflitto

The screenshot shows a GitHub Pull Request titled "Update README.md #5". The PR is from the "Prova-conflicto-1" branch to the "Prova-conflicto-2" branch, submitted by user "murty91". The PR status is "Open".

Below the title, there are tabs for "Conversation", "Commits", and "Files changed". The "Files changed" tab is selected, showing a diff for "README.md".

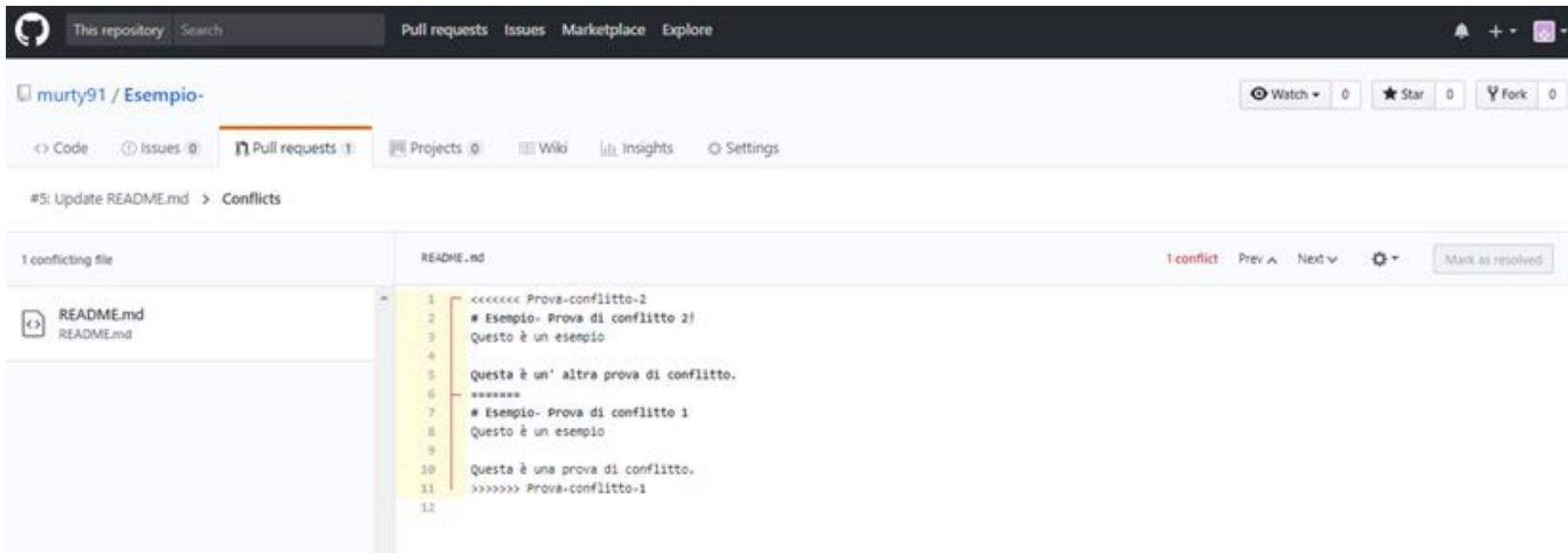
A comment from "murty91" is visible, stating "Prova di conflitto 2!".

Below the diff, a message indicates: "This branch has conflicts that must be resolved". A button labeled "Resolve conflicts" is present. Below this, a section titled "Conflicting files" lists "README.md".

At the bottom, there is a "Merge pull request" button and a note: "You can also open this in GitHub Desktop or view command line instructions."

On the right side, there are sections for "Reviewers", "Assignees", "Labels", "Milestone", and "Notifications". A blue arrow points to the "Resolve conflicts" button.

# Problema nel Merge: Conflitto



The screenshot shows a GitHub interface for a repository named 'murty91 / Esempio-'. The top navigation bar includes links for 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The repository name is displayed on the left, with 'Watch', 'Star', and 'Fork' buttons on the right. Below the repository name, there are tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The 'Pull requests' tab is active, showing a pull request titled '#5: Update README.md' with a 'Conflicts' section. The conflict is located in the file 'README.md'. The interface shows a diff view of the file, with a red bracket on the left indicating the conflict. The text in the diff is as follows:

```
1 <<<<<< Prova-conflitto-2
2 # Esempio- Prova di conflitto 2!
3 Questo è un esempio
4
5 Questa è un' altra prova di conflitto.
6
7 # Esempio- Prova di conflitto 1
8 Questo è un esempio
9
10 Questa è una prova di conflitto.
11 >>>>>> Prova-conflitto-1
12
```

At the top of the diff view, it says '1 conflict'. On the right side of the diff view, there are buttons for 'Prev', 'Next', and 'Mark as resolved'.

# Problema nel Push: Conflitto

## Gestione dei conflitti :

Un altro conflitto tipico in cui possiamo imbatterci questa volta  
Non riguarda il **merge** ma il **push** del nostro file.

- Git infatti non permette di effettuare da due utenti diversi  
Un **push** sulla stessa **branch**.
- Questo avviene per evitare modifiche indesiderate e perdita di controllo della nostra versione.
- Questo problema lo incontreremo nella gran parte dei casi nel momento in cui creeremo un team (**organization** su github).

# Cosa sono le Fork su GitHub



## Fork :

- Sono un'**alternativa** alla branch
- Il forking non è un elemento di git ( non esiste infatti un comando git fork), ma di github e sistemi simili (per esempio bitbucket o gitlab).
- Serve nella pratica a **creare una coppia** server del repository di partenza e quindi portare avanti un progetto in una **direzione diversa** da quella originale.

# Cosa sono le Fork su GitHub

- Spesso utilizzato nei progetti open source per **creare versioni personalizzate** o adattamenti da testare prima di essere ricongiunti con la versione originale.
- Per essere **ricongiunta** una fork non avendo i necessari privilegi di partenza non può avvalersi del comando git push, ma si può solo creare una **Pull Request**.

# Ringraziamenti e Contatti

- Ringraziamo tutti per l'interesse dimostrato e vi invitiamo a contattarci per eventuali dubbi o domande sui nostri profili GitHub
-  murty91
-  MarcoEna