Metereologia

Un ente che si occupa di meteorologia raccoglie i dati dei fenomeni naturali da varie stazioni sparse per il territorio nazionale e li mette in un repository centralizzato per essere studiati dagli esperti che lavorano nella sede principale. Tu sei un tecnico informatico che prima inserisci i dati giornalieri in un file di testo e fai la commit di essi, ma ti rendi conto che hai sbagliato file e quindi utilizzi i comandi di git per annullare le tue modifiche errate e effettuare la commit del file giusto. Più in dettaglio, i passi da effettuare sono:

1. Creare una *fork* del progetto https://github.com/carmelo-cina-sal/Meteo sul tuo account GitHub. Se l'operazione è stata fatta correttamente, dovrai trovarti il progetto sul tuo repository.
2. Effettuare la *git clone* della nuova repository sul tuo PC locale, sotto *c:\Utenti\studente\PortableGit\repository*.
3. Spostarsi sotto la directory del nuovo progetto appena clonato.
4. Con un editor di testo, aggiungere alla fine del file *Umidita.txt* la seguente riga:
   `94.5 18.74 17.53`
5. Ti rendi conto che in realtà dovevi aggiungere tale riga al file *Precipitazioni.txt* quindi effettua il comando *git reset* sul file *Umidita.txt* per riportarlo a come era prima.
6. Aggiungi la riga alla fine del file corretto, ossia *Precipitazioni.txt*.
7. Per poter aggiungere tali modifiche al repository, sarà necessario utilizzare il comando *git add*.
8. Al fine di allineare il tuo repository remoto, effettua la *commit* sul tuo repository locale e infine effettua la *push* in remoto
9. Su GitHub, dalla copia del tuo repository, effettua la Pull request al progetto principale. ***Nel titolo della Pull request metti il tuo Nome e Cognome, nel commento incolla i comandi inseriti sulla bash di git***. Sei non sei arrivato a fare la pull request, incolla i comandi in un file di testo e invialo a carmelo.cina@iistommasosalvini.edu.it giuseppe.schiavone@iistommasosalvini.edu.it

Di seguito uno schema di riepilogo dei comandi principali di Git.

Configurazione globale
Configurazione dell'utente valida per tutti i repository
 	
`$ git config --global user.name "[name]"`
Imposta il nome che vuoi mostrare sulle tue commit
 	
`$ git config --global user.email "[email address]"`
Imposta l'email che vuoi mostrare sulle tue commit
	Creare repository
	Crea un nuovo repository o clonane uno esistente da un URL
	
	`$ git init [project-name]`
	Crea un nuovo repository locale con il nome specificato
	
	`$ git clone [url]`
	Scarica un progetto esistente e il suo storico di cambiamenti

Effettuare modifiche
Rivedi i cambiamenti al codice e prepara una commit
 	
`$ git status`
Elenca tutti i file nuovi o modificati
 	
`$ git diff`
Mostra le differenze non ancora nell'area di staging
 	
`$ git add [file]`
Crea uno snapshot del file in preparazione al versioning
 	
`$ git diff --staged`
	Rivedere lo storico
	Esplora l'evoluzione dei file del progetto
	
	`$ git log`
	Elenca lo storico di versione per il branch corrente
	
	`$ git log --follow [file]`
	Elenca lo storico di versione per il file specificato, incluse rinominazioni
	
	`$ git diff [first-branch]...[second-branch]`
	Mostra la differenza tra due branch

Mostra le differenze tra staging e ultima modifica

`$ git reset [file]`
Rimuovi un file dall'area di staging, ma mantieni le modifiche

`$ git commit -m "[descriptive message]"`
Salva gli snapshot dei file in maniera permanente nello storico

`$ git show [commit]`
Mostra i metadati e i cambiamenti della commit specificata

Annullare commit
Elimina errori e altera lo storico dei cambiamenti

`$ git reset [commit]`
Annulla tutte le commit effettuate dopo [commit], preservando i cambiamenti locali

`$ git reset --hard [commit]`
Elimina tutto lo storico e i cambiamenti fino alla commit specificata

`$ git restore [file]`
Allinea il file locale al repository

Sincronizzare i cambiamenti
Collegati a un URL remoto e ottieni lo storico dei cambiamenti

`$ git fetch [remote]`
Scarica lo storico dei cambiamenti dal repository remoto

`$ git merge [remote]/[branch]`
Unisci il branch remoto con quello locale

`$ git push --set-upstream [remote] [branch]`
Carica tutti i cambiamenti dal branch locale su GitHub

`$ git pull`
Scarica lo storico e unisci i cambiamenti

`$ git branch [branch-name]`
Crea un nuovo branch