Git Essentials: Ghidul complet

# 1. Bazele Git

- \*\*Repozitoriu (Repository):\*\* Un depozit pentru codul sursă. Poate fi local (pe calculatorul tău) sau remote (pe un server, de exemplu, GitHub).  
- \*\*Commit:\*\* O înregistrare a modificărilor făcute în proiect. Fiecare commit are un identificator unic (hash).  
- \*\*Branch:\*\* O ramură de dezvoltare. Git permite crearea de ramuri multiple pentru a facilita munca paralelă pe diverse funcționalități.  
- \*\*Merge:\*\* Procesul de îmbinare a modificărilor dintr-o ramură în alta.  
- \*\*Clone:\*\* Copierea unui repo remote pe calculatorul tău.  
- \*\*Pull:\*\* Descărcarea și îmbinarea modificărilor dintr-un repo remote în repo-ul local.  
- \*\*Push:\*\* Încărcarea modificărilor din repo-ul local în repo-ul remote.

# 2. Comenzi de bază

- `git init`: Inițializează un nou repo Git.  
- `git clone [URL]`: Clonează un repo remote pe calculatorul tău.  
- `git status`: Afișează starea curentă a repo-ului, arătând ce fișiere au fost modificate.  
- `git add [fisier]`: Adaugă un fișier în zona de staging pentru următorul commit.  
- `git commit -m "mesaj"`: Creează un commit cu modificările din zona de staging.  
- `git push`: Împinge modificările locale către repo-ul remote.  
- `git pull`: Îmbină modificările din repo-ul remote în repo-ul local.  
- `git branch`: Afișează ramurile curente. `git branch [nume]` creează o nouă ramură.  
- `git checkout [nume-ramura]`: Schimbă ramura curentă.

# 3. Conceptul de Distribuire

- \*\*Distribuit:\*\* Fiecare utilizator are o copie completă a istoricului de versiuni. Acest lucru permite lucrul offline și oferă redundanță.

# 4. Mergers și Conflicte

- \*\*Merge Conflicts:\*\* Apar atunci când două modificări contradictorii sunt combinate. Acestea trebuie rezolvate manual.  
- \*\*Rebase:\*\* O alternativă la merge, care reface istoricul commit-urilor pentru a crea un istoric mai liniar.

# 5. Colaborarea cu Alții

- \*\*Fork:\*\* Crearea unei copii independente a unui repo. Utilizată des pe platforme cum ar fi GitHub pentru a contribui la proiectele altora.  
- \*\*Pull Request:\*\* O cerere de îmbinare a modificărilor dintr-o ramură (sau fork) într-o altă ramură. Este utilizată pentru a discuta și revizui codul înainte de a-l îmbina.

# 6. Alte Comenzi Utile

- `git log`: Afișează istoricul commit-urilor.  
- `git diff`: Afișează diferențele dintre modificări.  
- `git stash`: Salvează modificările curente fără a le comite, permițându-ți să lucrezi pe altceva și să le recuperezi mai târziu.  
- `git reset`: Resetează istoricul commit-urilor. Poate fi periculos dacă nu este folosit corect.

# 7. Best Practices

- \*\*Commit-uri Frecvente:\*\* Realizează commit-uri frecvente și semnificative pentru a facilita urmărirea modificărilor.  
- \*\*Mesaje de Commit Clare:\*\* Scrie mesaje clare și concise pentru commit-uri pentru a înțelege rapid modificările efectuate.  
- \*\*Utilizează Ramuri:\*\* Creează ramuri pentru funcționalități sau bug fix-uri pentru a menține ramura principală stabilă.  
- \*\*Revizuire de Cod:\*\* Efectuează revizuiri de cod prin pull requests pentru a asigura calitatea codului și a colabora eficient cu echipa.

# Indicații Adiționale

- \*\*Configurație inițială:\*\* Configurarea numelui de utilizator și a emailului.  
 ```sh  
 git config --global user.name "Numele Tău"  
 git config --global user.email "emailul@tau.com"  
 ```  
- \*\*Alias-uri Git:\*\* Crearea de comenzi rapide pentru comenzi comune.  
 ```sh  
 git config --global alias.co checkout  
 git config --global alias.br branch  
 git config --global alias.ci commit  
 git config --global alias.st status  
 ```  
- \*\*.gitignore:\*\* Fisier pentru excluderea fișierelor și directoarelor din repo.  
 ```  
 # Exemple de fișiere și directoare de exclus  
 \*.log  
 node\_modules/  
 .env  
 ```  
- \*\*Clonarea unui repo specificând ramura:\*\*  
 ```sh  
 git clone -b [nume-ramura] [URL]  
 ```  
- \*\*Revederea istoricului grafic:\*\*  
 ```sh  
 git log --graph --oneline --all  
 ```