* E foarte important ca atunci cand cream un soft sau lucram pe o masina, sa avem o documetatie care arata ce a fost modificat si cand, ca sa putem rezolva posibilele probleme care apar.
* **Version Control** – Ne ajuta sa ducem o evidenta a modificarilor facute in proiect/cod.Totodata, Creaza mai multe versiuni ale unui program/proiect si ne permite usor sa revenim la el, sa vedem cine si ce a facut atunci si permite mai multor oameni sa lucreze la acelasi proiect.
* **Version control** ne permite sa vedem cum inflentam noi proiectul si sa stabilim ce si cum vor trebui altii sa faca la el.
* Mereu cand lucram la un cod, e posibil ca sa facem anumite schimbari ce vor deteriora codul si sa avem nevoie sa revenim la codul de dinainte de a face schimbarile.
* Diff file / patch file – fisiere care contin informatii despre ce a fost modificat in fisiere.
* Diff file1 file2 – arata diferentele dintre 2 fisiere. Poate fi aplicat si unui folder cu mai multe fisiere, si va pastra date despre toate. Arata cotinutul complet al celui de al doilea fisier, si ne arata ce a adaugat sau sters fata de vechiul.

“<” – arata ca linia a fost stearsa din fisier

“>” – arata ca linia data a fost adaugata

6c6 – prima cifra, pana la c, arata unde se afla linia care a fost modificata in primul fisier , si a 2 cifra dupa c care linie din al 2 fisier a fost pusa in locul ei. C – de la changed

5c5,6 – linia 5 a fost stearsa si a fost inlocuita cu 2 linii din al 2 fisier - 5 si 6

11a12,15 – la linia11 din primul fisier, au fost adaugate liniile 12,13,14,15 din al 2 fisier. A – added

!!! **,** reprezinta un interval

* Diff -u file1 file2 – format mai bun si mai clar
* Linie stearsa

+ Linie adaugata

* Wdiff file1 file2 – sublinieaza cuvintele noi
* Diff [-u] old\_file new\_file …. > change.diff – creaza un fisier .diff cu toate diferentele , ca sa poata fi vazute si de altii si aplicat apoi. El e numit diff file sau patch .Poate fi aplicat si unui folder cu mai multe fisiere, si va pastra date despre toate.
* Patch file < fisier.diff – face ca diferentele din fisierul .diff sa fie applicate la fisierul specificat.

**Mai multe despre Version Control System (VCS )**

**VCS –** pastreaza urme ale modificarilor facute de noi in fisiere. Putem afla ce s-a modificat, unde, cand si de cine si punem reveni la niste fisiere anterioare. VCS ne ajuta sa lucram cu diferite surse. VCS pastreaza toate versiunile fisierelor!

* VCS ne permite si sa facem descrieri ale fiecarei versiuni.
* VCS permite de a crea colectii la care au acces mai multe persoane si a caror modificari se fac in timp real, evitand astfel redutanta(repetarea) datelor pe fiecare calculator.
* Cel mai mare avantaj e ca putem edita multiple fisiere si sa tratam aceasta colectie de modificari ca un tot intreg, colectie ce e numita **commit**.Commit e o colectie unde se pastreaza anume modificarile facute pe un local repository. Commit nu e un fisier, dar salveaza fisiere.
* Folderul in care se gaseste .git e **Repozitoriu. Aceasta e zona unde se afla deci fisierele noastre, modificarile lor etc.**

**Despre GIT**

* GIT este un VCS
* GIT are o arhitectura distribuita, adica fiecare persoana care contribuie la un repozitoriu are o copie completa a acestuia pe propria masina. Fiecare poate utiliza codul pe care altii l-au facut.
* GIT poate functiona atat pentru proiecte personale, adica fara a impartasi proiectul cu altii, cat si ca un server, desi el nu e considerat chiar server.In schimb, el poate comunica cu serverele GIT utilizand HTTP
* SCM(source control manager) = VCS
* **Commit** – colectie ce pastreaza modificari din unul sau mai multe fisiere.

**Zone in git**

* **Working tree –** zona inafara directorului git, adica directorul unde se afla si .git. **Ea contine versiunea curenta a proiectului, cu care noi lucram**. Adica aici se afla pur si simplu toate fisirele noastre. E ca o masa de lucru unde se fac diferite modificari la obiecte, ca la tamplarie. Aceasta zona contine toate fisierele ce sunt urmarite de git si toate fisierele care inca nu sunt urmarite de git. Git salveaza informatii despre orice modificare facuta in fisiere,orice creare si stergere a unui fisier, dar nu va face nimic cu schimbarile detectate, pana noi nu-I spunem sa atraga atentia la ele si sa le salveze undeva. Fisierele cu rosu fie sunt in working tree, si sunt untracked, adica git nu are salvat nimic despre modificarile din ele, sau nu e luata nicio poza a modificarilor lor, fie au fost modificate si nu s-a folsit git add(sunt modified). Daca dam git add file la un fisier,se va crea o “poza” a lui,adica un file cu modifciarile din el si el va fi verde,asta insemnand ca e pastrata un file cu modificarile lui, insa daca modificam iar ceva in fisier, el din nou va fi rosu, caci modificarile noi nu au fost salvate inca, de aceea e iar rosu. Rosu arata ca s-a facut ceva in fisier, dar git nu a salvat. In asa caz fisierul iese din staging area, si ramane doar in working tree, insa poza facuta lui ramane in staging area.

**Daca modificam un fisier si il adaugam in staging area, se creaza un file cu modificarile facute in el, apoi daca modificam ceva in fisier, el trece din starea de staged in modified, si el nu mai e in staging area, desi in staging area continua sa existe un fisier cu modificarile ce s-au facut in el pana la adaugarea lui in zona asta. Daca adaugam fisierul modificat in staging area iar, vechiul fisier ce pastra modificatile facute in fisier prima data, e inlocuit cu unul nou ce pastreaza noile modificari.**

* **Staging area –** fisier mentinut de git care contine toate informatiile despre ce schimbari trebuie sa se duca in commit-ul nostru urmator. In zona asta, modificarile noastre nu sunt puse in directorul .git, dar sunt abia culese si urmeaza sa fie puse. **Asa ii spunem lui git sa acorde o atentie particulara la fisiere** **si sa ia o “poza” a lor.** Staging area contine schimbarile ce au fost marcate sa fie adaugate in urmatorul commit, adica git atrage atentie la ce s-a modificat pentru a salva mai apoi. Totusi, el salveaza aici doar ce a inregistrat pana la **git add file**. Face ca o poza. Daca facem vreo modificare la fisier, va trebui iar sa se scrie **git add file**, caci el salveaza in stagging area doar ce s-a facut pana la scrierea comenzii. Daca modificam fisierul si nu-I dam git add iar, el e in working tree, nu staging area, caci salvarile lui vechi sunt acolo, nu si noi.In stagging area sunt fisierele cu verde, adica e luata o poza cu ele deja.
* **Git Directory –** contine istoria tuturor fisierelor si modificarilor. E directorul .git; Aici se adauga modificarile salvate de stagging area prin comanda **git commit**.

Working tree e masa de lucru unde se modifica fisierele. Staging area e postasul care livreaza informatiile despre schimbari la git ca el sa le “pozeze” si sa le salveze.

* **Tracked file –** fisier despre care .git aduna istorii de modificare,despre care are “poza” cu toate modificarile lui la momentul actual. Devine tracked cand se foloseste git add file, si desi il modificam, el nu mai devine untracked, dar iese din stanging area si devine modified.
* **Untracked file –** fisier despre care git nu aduna nimic inca. E cazul, deobicei, pentru fisiere noi sau fisiere neadaugate cu git add. Despre ele nu are inca nicio “poza”, deci nu are nimic despre el.

Orice tracked file poate avea una din 3 stari:

* **Modificat** – fisier care a fost in stagging area, dar in care s-au facut modificari si nu au fost puse din nou in stagging area. In staging area continua sa existe o versiune anterioara a lui, cu modificarile vechi, nu cele actuale.
* **Staged**(in scena) – Git noteaza oricand noi modificam vreun fisier, dar nu va stoca nimic despre modificari pana nu le punem in staging area. Fisierul e staged cand ii spunem lui git ca vrem ca el nu doar sa adune informatiile despre modificari, dar si sa le salveze, sa faca o poza ,si deci trece in staged cu git add si **staged inseamna ca git are o poza cu cele mai recente modificari din fisier**. Git anume va acorda o atentie deosebita fisierelor staged, caci modificarile ce se fac el nu doar le va gasi, dar le va si salva undeva,temporar inca. Deci, un fisier staged e un fisier despre care git a coelctat date legate de modificari cele mai recente cu git add. Daca modificam fisierul si nu il punem iar cu git add, el nu mai e staged, caci git nu are poze despre noile sale modificari, ci despre vechile, si fisierul e modified.

Modificarile Fisierelor ce sunt staged sunt gata sa fie committed in

proiectul Git, dar nu sunt inca salvate permanent.

* **Commited**(savarsit) – cand fisierul e committed, modificarile facute la el sunt adaugate in GIT repository.

**Commit message**

Cand scriem un mesaj la commit, e important sa fie clar ce am facut anume, ce am modificat. Un commit message e format din mai multe sectii:

1. Primia linia e un sumar scurt despre comit, apoi urmeaza o liie goala
2. Apoi se scrie o descriere concreta despre commit, de ce am facut modificarile,ce buguri am rezolvat si orice info ce ar putea fi utila.

Randurile ce incep cu # nu vor fi puse in mesajul commitului.