Facultatea Calculatoare, Informatica si Microelectronică

Universitatea Tehnică a Moldovei

Medii Interactive de Dezvoltare a Produselor Soft Lucrarea de laborator#1

**Version** **Control** **Systems** **si** **modul** **de** **setare** **a** **unui** **server**

A efectuat: studentul gr. TI-151 Iordachi Cristian

A verificat: Gojin V.

Chișinău 2017

**Obiective:**

 Version Control Systems (git)

**Cerințele** **laboratorului:**

 *Basic* *Level* :

o initializeaza un nou repositoriu

o configureaza-ti VCS

o crearea branch-urilor (creeaza cel putin 2 branches)

o commit pe ambele branch-uri (cel putin 1 commit per branch)  *Normal* *Level* :

o seteaza un branch to track a remote origin pe care vei putea sa faci push (ex. Github, Bitbucket or custom server)

o reseteaza un branch la commit-ul anterior

o salvarea temporara a schimbarilor care nu se vor face commit imediat. o folosirea fisierului .gitignore

 *Advanced* *Level* :

o merge 2 branches

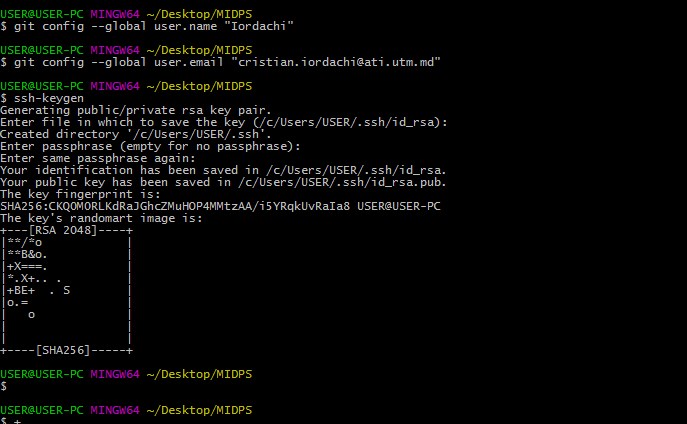
o rezolvarea conflictelor a 2 branches o [comezile git](https://www.siteground.com/tutorials/git/commands.htm) care trebuie cunoscute

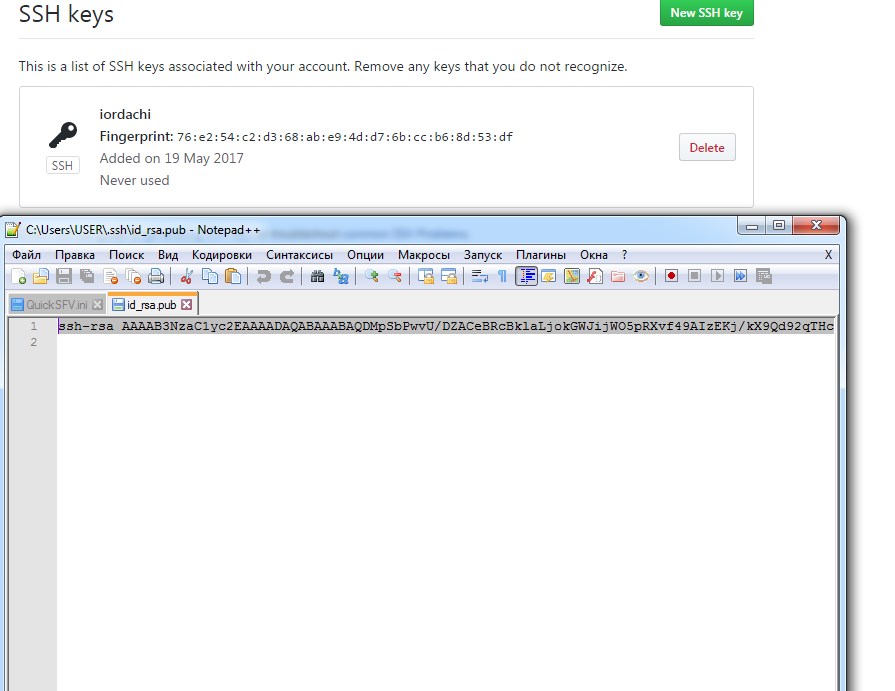
**Analiza** **Lucrării** **de** **laborator:**

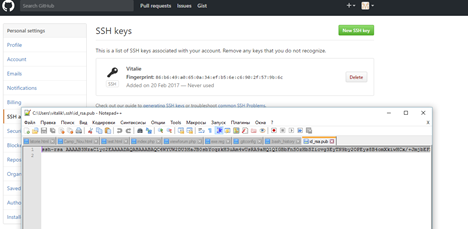
Link-ul la repozitoriu https://github.com/Pastuh2/MIDPS

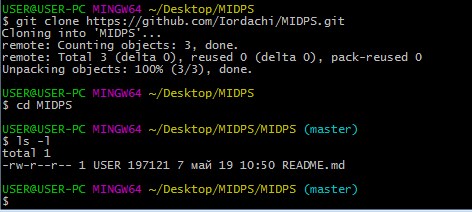
Am creat repozitoriul prin metoda online.Am deschis pagina mea pe github.com,click pe Repositories și apoi pe butonul New.Atunci când am creat repozitoriul MIDPS,l-am inițializat cu un fișier README.

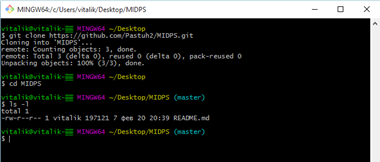
Următorul pas constă în configurarea git-ului.Configurăm numele și email-ul prin comenzile **git** **config** **–global** **user.name** **”NUMELE”** **git** **config** **–global** **user.email** **“EMAIL”** sigenerarea cheii SSH pe care o vom copia în setările de pe github.



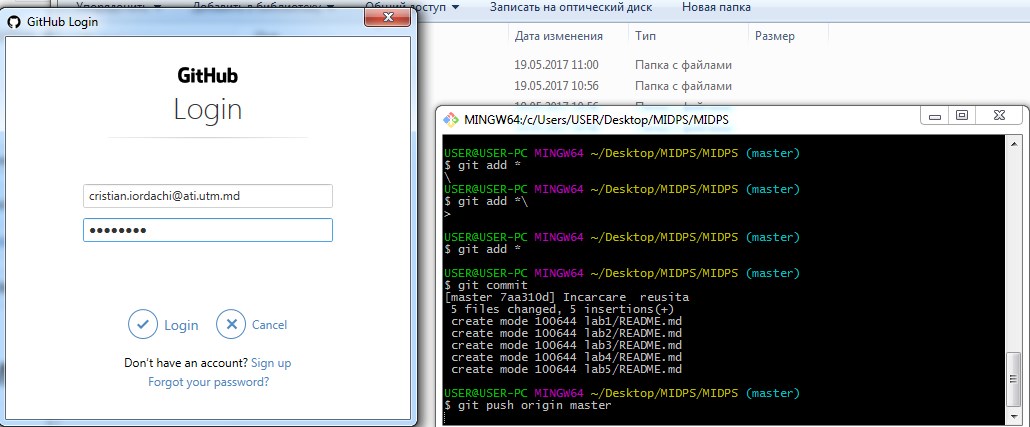


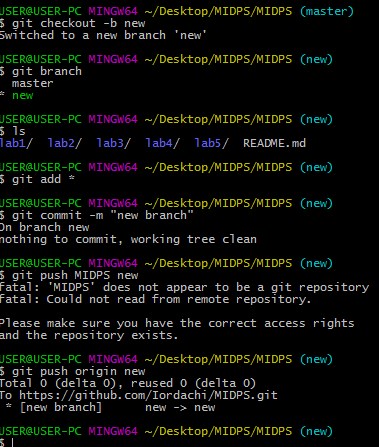
După ce am generat keygen-ul,clonăm repozitoriul pe mașina locală.



Pentru a adăuga fișiere pe repozitoriu,vom folosi următoarele comenzi: **git** **add** **\*** -comanda indexează toate fișierele. **git** **commit** **-m** - comanda face un snapshot la toate schimbările noastre.

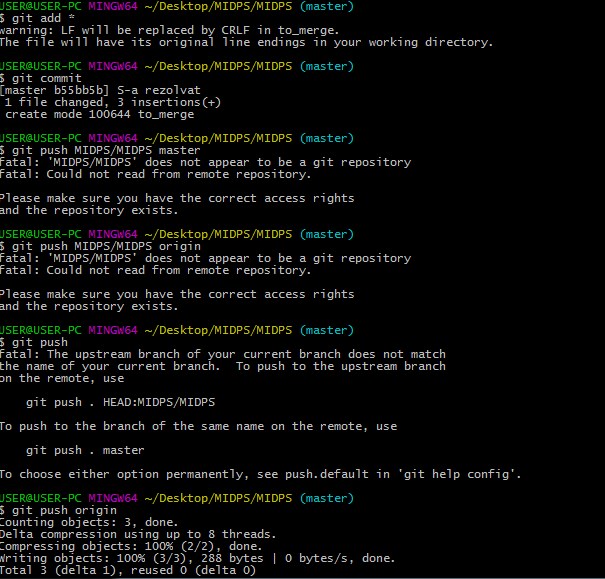
**git** **push** **origin** **master** - comanda încarcă toate fișierele indexate pe git.Totodată vom folosi **git** **status** și **git** **show** pentru a ne asigura că fișierele au fost adăugate în repozitoriu.





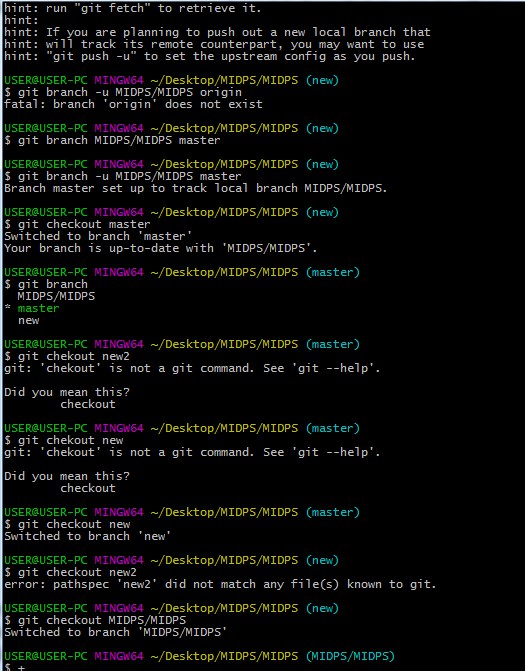
Revenirea la o versiune mai veche poate fi efectuată cu ajutorul comenzii **git** **reset** **–** **TYPE** **”codul** **comitului”**. Există diferența între **–soft** și **–hard** , când facem soft reset indexurile rămân neschimbate. Iar în cazul în care facemhard reset , pierdem indexurile.

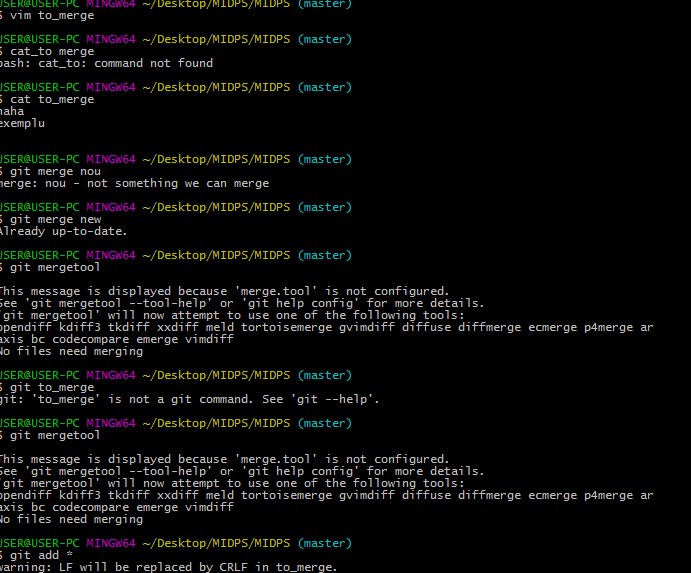
Am creat un fișier nou text.txt în versiunea 1. După care l-am șters și am făcut commit la versiunea 2 în care am sters fișierul test.txt.Dorim să revenim la versiunea1. La început vom lansa comanda **git** **log** care ne arată logul de commituri și codul pentru fiecare commit. Vom avea nevoie de primele 7 cifre la commitul anterior.

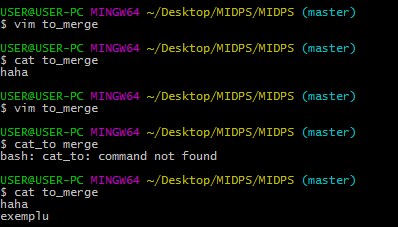


VCS ne permite să avem mai multe **branch-uri**. Branch-urile sunt comod de folosit când dorim să lucrăm paralel la un proiect și apoidorim să unim toate modificarile.

**git** **branch** **”name”** - creează un branch nou cu numele ”name”. **git** **branch** - vizualizarea branch-urilor (\* indică branch-ul curent). **git** **branch** **-d** **”nume”** - șterge branch-ul ”nume”. **git** **checkout** **-b** **”name”** - creează un branch nou cu numele ”name” și face switch la el







**Concluzie:** Am studiat VCS.Mi-am aprofundat cunoștiințele în GitHub.Am învățat cum se creează mai multe branch-uri,cum se mută de la unul la altul,să fac operațiile de resetare la commit-ul anterior.Am aplicat comenzile fundamentale.Consider că fiecare programator trebuie să cunoască GitHub,să lucreze cu VCS. Chiar daca am avut problem cu conexiunea ssh am rezolvat problema cu ajutorului forumului Github.com(Redactind config din mapa .git inlocuiid Http cu SSH problema cu Git push origin master s-a rezolvat ce mi-a permis de a incarca fisierele pe repozitoriul meu fara probleme.

**Am** **lucrat** **cu** **comenzile** **de** **linie** **cum** **ar** **fi** **:’’** **current** **state**

git statuslist which (unstaged) files have changed git difflist (unstaged) changes to files

git loglist recent commits git add fnstage file

git commit -m 'message'commit file

git commit -am 'message'add/commit all changes from all tracked files (no untracked files) in one go

git status

git fetch origin

etc.