

FACULTATEA CALCULATOARE, INFORMATICA SI
MICROELECTRONICA

UNIVERSITATEA TEHNICA A MOLDOVEI

MEDII INTERACTIVE DE DEZVOLTARE A
PRODUSELOR SOFT

LUCRAREA DE LABORATOR#1

Version Control Systems si modul de setare a unui server

Autor:
Toma Ana

lector asistent:
Irina Cojanu
lector superior:
Radu Melnic

1 Scopul lucrarii de laborator

De a se invata utilizarea unui Version Control System si modul de setare a unui server.

2 Obiective

Studiarea Version Control Systems (git).

3 Mersul lucrării de laborator

3.1 Cerintele

1. Initializarea unui nou repository.
2. Configurarea VCS.
3. Commit, Push pe branch.
4. Folosirea fisierului .gitignore.
5. Revenire la versiunile anterioare.
6. Crearea branch-urilor noi.
7. Commit pe ambele branch-uri.
8. Merge la 2 branchuri.
9. Rezolvarea conflictelor.

3.2 Analiza Lucrării de laborator

Linkul la repository **<https://github.com/TomaAna/MIDPS>**

Sunt mai multe modalitati de a initializa un repository pe github. Putem crea o mapa goala in care vom plasa gitul nostru prin intermediul comenzii **git init**.

Urmatorul pas este crearea insusi a noului repository pe care il vom crea utilizand urmatoarea comanda **curl -u 'USER' https://api.github.com /user/repos -d '{"name": "NUME"}'**. Unde cuvintele scrise cu CAPS se vor inlocui cu numele utilizatorului si numele repositoryului.

Dupa aceasta este necesar sa unim gitul nostru gol cu repositoryul creat. Vom folosi urmatoarea comanda **git remote add origin "Linkul la repository"**

O alta metoda de a crea un repository este cea online. Pentru aceasta este nevoie sa deschidem pagina noastra pe github , sa alegem **repositories** si sa apasam butonul **new**.

Configurarea gitului consta in mai multe etape. La inceput vom configura numele si emailul. Scrim urmatoarele comenzi:

git config --global user.name "NUMELE"

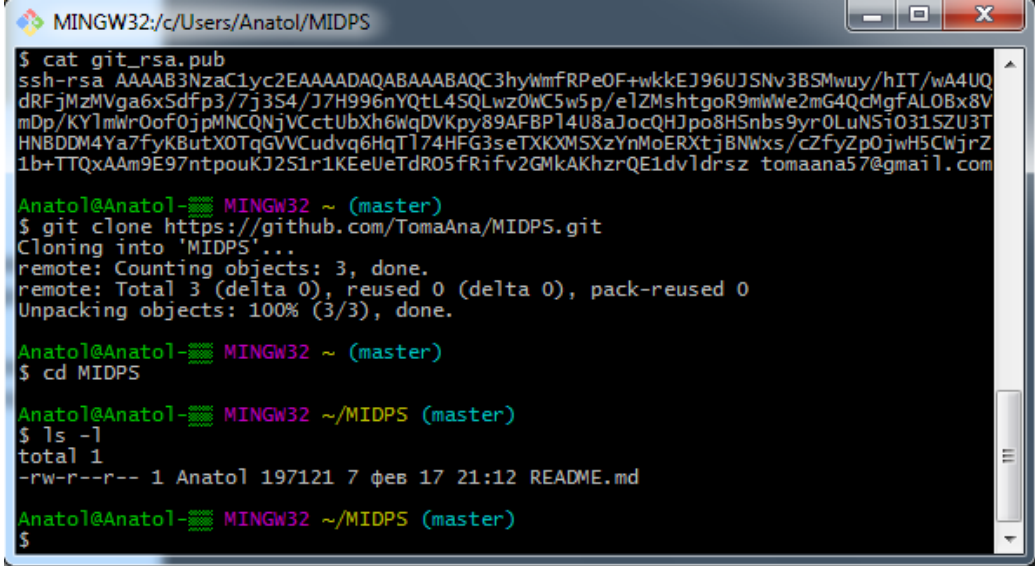
git config --global user.email EMAIL

```
MINGW32/c/Users/Anatol
Anatol@Anatol- MINGW32 ~ (master)
$ git config --global user.name "AnaToma"
Anatol@Anatol- MINGW32 ~ (master)
$ git config --global user.email "tomaana57@gmail.com"
Anatol@Anatol- MINGW32 ~ (master)
$ git config --list
core.symlinks=false
core.autocrlf=true
core.fscache=true
color.diff=auto
color.status=auto
color.branch=auto
color.interactive=true
pack.packsizelimit=2g
help.format=html
http.sslcainfo=D:/C++/Git/mingw32/ssl/certs/ca-bundle.crt
diff.astextplain.textconv=astextplain
rebase.autosquash=true
credential.helper=manager
user.name=AnaToma
user.email=tomaana57@gmail.com
```

Urmatorul pas consta in generarea la cheia **SSH** (Secure Shell). Scriem in CLI **ssh-keygen**, iar cheia obtinuta o copiem in setarile noastre de pe git. Este de dorit sa initializam repozitoriul nostru cu un fisier **README.md** si un **.gitignore**. In fisierul README.md vom adauga niste informatie pentru cei care se vor folosi de repozitoriu iar in fisierul .gitignore vom adauga toate fisierele ce trebuiesc ignorate (adica sa nu fie incarcate).

```
MINGW32/c/Users/Anatol
SHA256:MU3LyaaRGqHS3TVpiwZWDADrE3qWqvZ5IQmA8ZUFSpq tomaana57@gmail.com
The key's randomart image is:
+---[RSA 2048]-----+
|o+o.+==+.+.|
|Eo=.ooo.B+|
|. * o.o.*oB.|
|+ + oo*|
|. B . ..S|
|+ + .|
|. . .|
|.. ..|
|o .o.|
+---[SHA256]-----+
Anatol@Anatol- MINGW32 ~ (master)
$ cat git_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQAC3hyWmFRPeOF+wkkEJ96UJ5Nv3B5Mwuy/hIT/wA4UQ
dRFjMzMVga6xSdfp3/7j3S4/37H996nYQtL4SQLwz0WC5w5p/e1ZMshgtgoR9mWw2mG4QcMqFALOBx8V
mDp/KY1mWr0oF0jpMNCQNjVCctUbXh6WqDVkpy89AFBP14U8aJocQHJpo8HSnbs9yr0LuNSi031SZU3T
HNBDDM4Ya7FyKButX0TqGVVCudvq6HqT174HFG3seTXKXMSXzYnMoERXtjBNWxs/cZfyZp0jwH5Cwjrz
1b+TTQxAAm9E97ntpouKJ2S1r1KEeUeTdR05fRifv2GMkAKhzrQE1dv1drsz tomaana57@gmail.com
Anatol@Anatol- MINGW32 ~ (master)
$ |
```

Dupa ce am generat keygen-ul,clonam repozitoriul pe masina locala.



```
MINGW32:/c/Users/Anatol/MIDPS
$ cat git_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQAC3hyWmfRPeOF+wkEJ96UJ5Nv3BSMwuy/hIT/wA4UQ
dRFjMzMVga6xSdfp3/7j354/J7H996nYQtL45QLwz0WC5w5p/e1ZMshtgoR9mWw2mG4QcMgfAL0Bx8V
mDp/KY1mWrOof0jpMNCQNjVCctUbXh6WqDVkpy89AFBP14U8aJocQHJpo8HSnbs9yr0LuNS1031SZU3T
HNBDDM4Ya7fyKButX0TqGVVCudvq6HqT174HFG3seTXKXMSXzYnMoERXtjBNWxs/cZfyZp0jwH5CwjZ
1b+TTQxAAm9E97ntpouKJ251r1KEeUeTdR05fRifv2GMkAKhzrQE1dvl drsz tomaana57@gmail.com

Anatol@Anatol- MINGW32 ~ (master)
$ git clone https://github.com/TomaAna/MIDPS.git
Cloning into 'MIDPS'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.

Anatol@Anatol- MINGW32 ~ (master)
$ cd MIDPS

Anatol@Anatol- MINGW32 ~/MIDPS (master)
$ ls -l
total 1
-rw-r--r-- 1 Anatol 197121 7 feb 17 21:12 README.md

Anatol@Anatol- MINGW32 ~/MIDPS (master)
$
```

Pentru a adauga fisiere pe repozitoriu,vom folosi urmatoarele comenzi: git add * - comanda indexeaza toate fisierele. git commit -m - comanda face un snapshot la toate schimbarile noastre. git push origin master - comanda incarca toate fisierele indexate pe git.Totodata vom folosi git status si git show pentru a ne asigura ca fisierele au fost adaugate in repozitoriu.

```
MINGW32/c/Users/Anatol/MIDPS/LAB1

Anatol@Anatol- [MINGW32 ~/MIDPS/LAB1 (master)]
$ touch text.txt

Anatol@Anatol- [MINGW32 ~/MIDPS/LAB1 (master)]
$ git add *

Anatol@Anatol- [MINGW32 ~/MIDPS/LAB1 (master)]
$ git commit -m "versiunea 1"
[master d10fb1d] versiunea 1
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 LAB1/text.txt

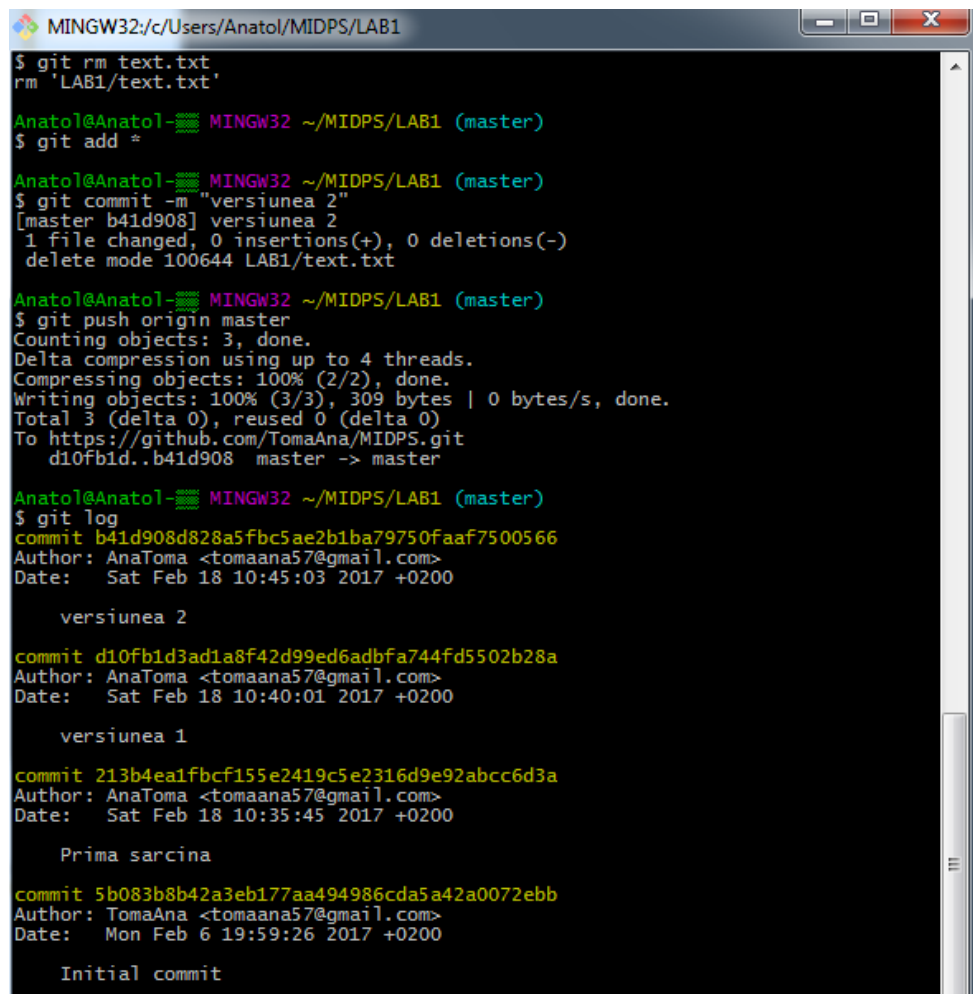
Anatol@Anatol- [MINGW32 ~/MIDPS/LAB1 (master)]
$ git push origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 321 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/TomaAna/MIDPS.git
213b4ea..d10fb1d master -> master

Anatol@Anatol- [MINGW32 ~/MIDPS/LAB1 (master)]
$ git rm test.txt
fatal: pathspec 'test.txt' did not match any files

Anatol@Anatol- [MINGW32 ~/MIDPS/LAB1 (master)]
$ ls
README.md text.txt

Anatol@Anatol- [MINGW32 ~/MIDPS/LAB1 (master)]
$ git rm text.txt
rm 'LAB1/text.txt'
```

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 facem `hard reset`, pierdem indexurile. Am creat un fișier nou `text.txt` în versiunea 1. După care l-am sters și am făcut commit la versiunea 2 în care am sters fișierul `test.txt`. Dorim să revenim la versiunea 1. 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.



```

MINGW32/c/Users/Anatol/MIDPS/LAB1
$ git rm text.txt
rm 'LAB1/text.txt'

Anatol@Anatol-MINGW32 ~/MIDPS/LAB1 (master)
$ git add *

Anatol@Anatol-MINGW32 ~/MIDPS/LAB1 (master)
$ git commit -m "versiunea 2"
[master b41d908] versiunea 2
1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100644 LAB1/text.txt

Anatol@Anatol-MINGW32 ~/MIDPS/LAB1 (master)
$ git push origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 309 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/TomaAna/MIDPS.git
d10fb1d..b41d908 master -> master

Anatol@Anatol-MINGW32 ~/MIDPS/LAB1 (master)
$ git log
commit b41d908d828a5fbc5ae2b1ba79750faaf7500566
Author: AnaToma <tomaana57@gmail.com>
Date: Sat Feb 18 10:45:03 2017 +0200

    versiunea 2

commit d10fb1d3ad1a8f42d99ed6adbfa744fd5502b28a
Author: AnaToma <tomaana57@gmail.com>
Date: Sat Feb 18 10:40:01 2017 +0200

    versiunea 1

commit 213b4ea1fbcf155e2419c5e2316d9e92abcc6d3a
Author: AnaToma <tomaana57@gmail.com>
Date: Sat Feb 18 10:35:45 2017 +0200

    Prima sarcina

commit 5b083b8b42a3eb177aa494986cda5a42a0072ebb
Author: TomaAna <tomaana57@gmail.com>
Date: Mon Feb 6 19:59:26 2017 +0200

    Initial commit

```

Acum folosim comenzile `git reset --hard` și `git reset --soft`.

```
MINGW32:/c/Users/Anatol/MIDPS/LAB1

commit 213b4ea1fbcf155e2419c5e2316d9e92abcc6d3a
Author: AnaToma <tomaana57@gmail.com>
Date: Sat Feb 18 10:35:45 2017 +0200

    Prima sarcina

commit 5b083b8b42a3eb177aa494986cda5a42a0072ebb
Author: TomaAna <tomaana57@gmail.com>
Date: Mon Feb 6 19:59:26 2017 +0200

    Initial commit

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git reset --hard d10fb1d
HEAD is now at d10fb1d versiunea 1

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ ls
README.md text.txt

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git reset --soft d10fb1d

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ ls
README.md text.txt
```

VCS ne permite sa avem mai multe branch-uri. Branch-urile sunt comod de folosit cind dorim sa lucram paralel la un proiect si apoi dorim sa unim toate modificarile. `git branch name` - creeaza un branch nou cu numele name. `git branch` - vizualizarea branch-urilor (* indica branch-ul curent). `git branch -d nume` - sterge branch-ul nume. `git checkout -b name` - creeaza un branch nou cu numele name si face switch la el.


```
MINGW32/c/Users/Anatol/MIDPS/LAB1
$ ls
README.md  text.txt

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git branch copy

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git branch
copy
* master

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git branch -d copy
Deleted branch copy (was d10fb1d).

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git branch nou

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git checkout -b nou
fatal: A branch named 'nou' already exists.

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git checkout nou
Switched to branch 'nou'

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git branch
master
* nou

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ ls
README.md  text.txt

MINGW32/c/Users/Anatol/MIDPS/LAB1

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git add *

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git commit -m "branch nou"
On branch nou
nothing to commit, working tree clean

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git push origin nou
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/TomaAna/MIDPS.git
* [new branch]      nou -> nou
```

```
MINGW32:/c/Users/Anatol/MIDPS/LAB1

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git branch
  master
* nou

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git checkout master
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.
(use "git pull" to update your local branch)
Switched to branch 'master'

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git pull
Updating d10fb1d..b41d908
Fast-forward
 LAB1/text.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 LAB1/text.txt

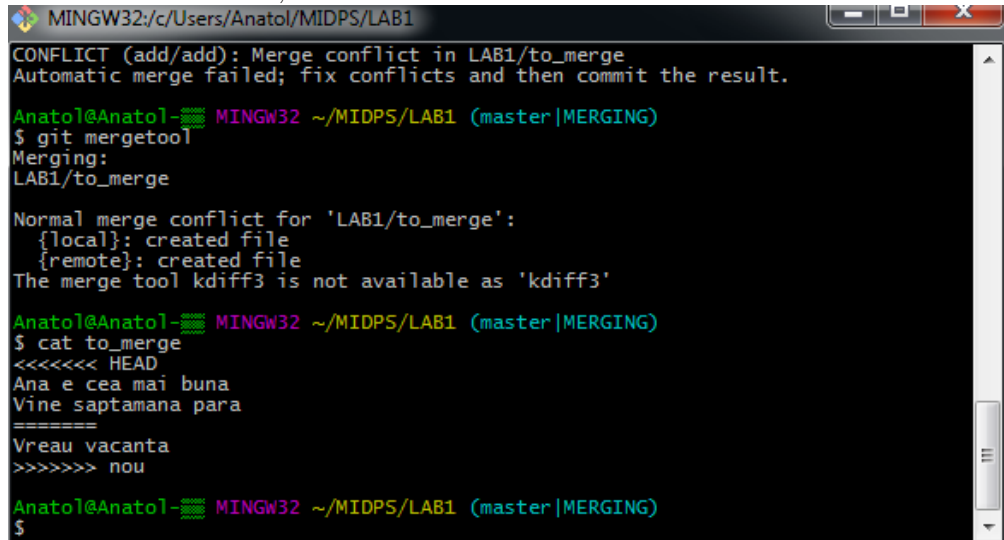
Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (master)
$ git checkout nou
Switched to branch 'nou'

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git branch -u origin/master
Branch nou set up to track remote branch master from origin.

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git branch --track "nou2" origin/master
Branch nou2 set up to track remote branch master from origin.

Anatol@Anatol- MINGW32 ~/MIDPS/LAB1 (nou)
$ git branch
  master
* nou
  nou2
```

Vom lucra cu 2 branch-uri - master si nou. Vom crea in fiecare branch cite un fisier to mer,dar continutul fiecaruia va fi diferit.



```
MINGW32/c:/Users/Anatol/MIDPS/LAB1
CONFLICT (add/add): Merge conflict in LAB1/to_merge
Automatic merge failed; fix conflicts and then commit the result.

Anatol@Anatol- [?] MINGW32 ~/MIDPS/LAB1 (master|MERGING)
$ git mergetool
Merging:
LAB1/to_merge

Normal merge conflict for 'LAB1/to_merge':
  {local}: created file
  {remote}: created file
The merge tool kdiff3 is not available as 'kdiff3'

Anatol@Anatol- [?] MINGW32 ~/MIDPS/LAB1 (master|MERGING)
$ cat to_merge
<<<<<<< HEAD
Ana e cea mai buna
Vine saptamana para
=====
Vreau vacanta
>>>>>>> nou

Anatol@Anatol- [?] MINGW32 ~/MIDPS/LAB1 (master|MERGING)
$
```

4 Concluzie

În urma efectuării lucrării de laborator numărul 1 la MIDP am studiat **VCS**. Mi-am aprofundat cunoștințele în folosirea platformei GitHub. Am analizat initializarea unui nou repository și am învățat cum să creez branch-uri noi. A fost o experiență nouă pentru mine, deoarece anterior nu am folosit git-ul. Astfel am aflat că git-ul este un sistem open-source de control a versiunilor conceput de Linus Torvalds – aceeași persoană care a creat sistemul de operare Linux. Git este în fapt asemănător cu alte astfel de sisteme de control a versiunilor, precum Mercurial, Subversion sau CVS. Git este efectiv o unealtă bazată pe linii de comandă, însă locul în care se centralizează toate datele și în care are loc stocarea proiectelor este efectiv Hub-ul, mai exact GitHub.com. Aici dezvoltatorii pot adăuga și stoca proiecte la care lucrează împreună cu alți pasionați.