

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:

Alexandru TOLOACA

asistent universitar:

Irina COJANU

Lucrarea de laborator 1

1 Scopul lucrării de laborator

Studierea și utilizarea unui Version Control System și modul de setare a unui server

2 Obiective

Studierea Version Control Systems (git).

Avantajele github

- Salvarea tuturor modificărilor, astfel ca se poate reveni oricând la o versiune mai veche dacă se descoperă introducerea unor defecte în ultima versiune
- Cea mai recentă versiune a codului este mereu disponibilă tuturor dezvoltatorilor, făcând astfel colaborarea și sincronizarea mai ușoară decât în cazul trimiterii de fișiere care conțin cod sursă.

Dezavantajele github

- Pot apărea complicații privind integritatea. (*La aplicarea greșită a schimbărilor*)

3 Implementarea lucrării de laborator

3.1 Cerinte:

- Initializarea unui nou repository.
- Configurarea unui VCS.
- Folosirea fisierului .gitignore
- Revenirea la versiunile anterioare.
- Crearea branch-urilor noi.
- Commit pe ambele branch-uri.
- Merge la 2 branch-uri.
- Rezolvarea conflictelor.

4 Analiza lucrării de laborator

Linkul la repository este : <https://github.com/burduleaFM/MIDPS>

4.1 Initializarea unui nou repository

Initializarea repositoryului pe github poate fi facuta prin mai multe metode:

- a) *Utilizind pagina principala si meniurile de pe saitul <http://github.com>*

Dupa logarea utilizatorului pe github, in coltul sting sus, se gasesc optiunile privind activitatea utilizatorului. In acest meniu gasim submeniul *New repository*.

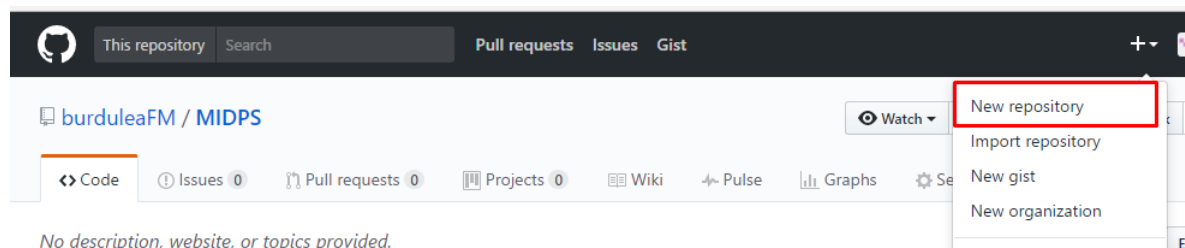
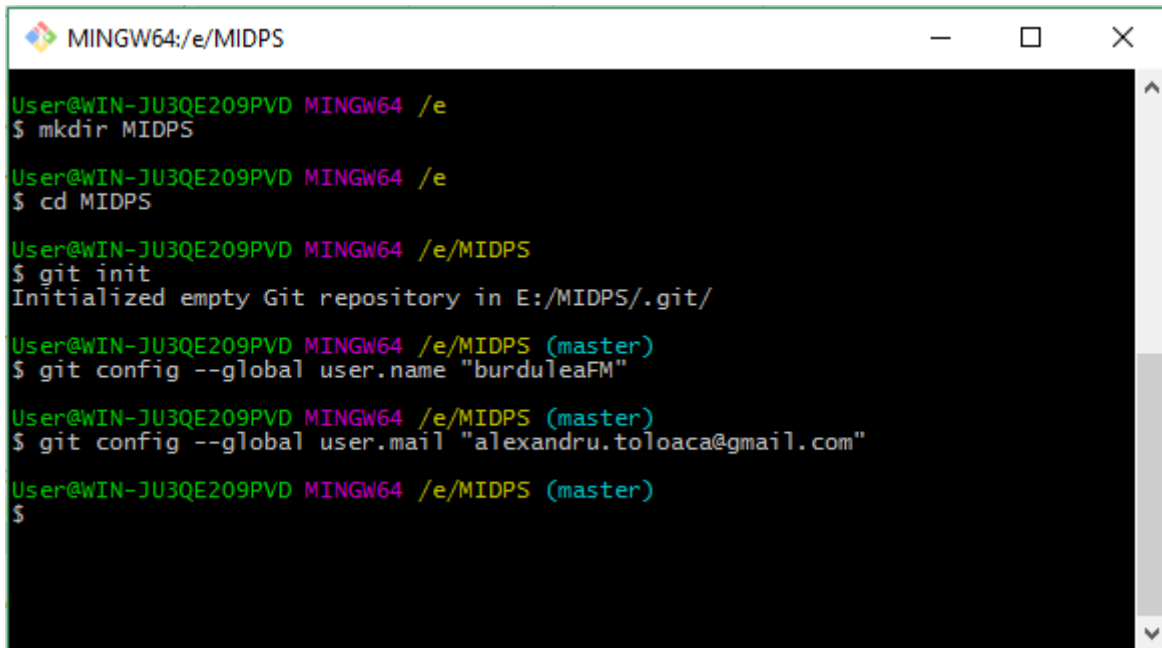


Figure 4.1 – Crearea repositoryului direct de pe github, [2]

- b) Folosind comanda **git init** Se creaza un folder gol in care vom plasa gitul nostru utilizind comanda data, in terminalul bash. Apoi aplicam configurariile necesare



```
MINGW64:/e/MIDPS

User@WIN-JU3QE209PVD MINGW64 /e
$ mkdir MIDPS

User@WIN-JU3QE209PVD MINGW64 /e
$ cd MIDPS

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS
$ git init
Initialized empty Git repository in E:/MIDPS/.git/

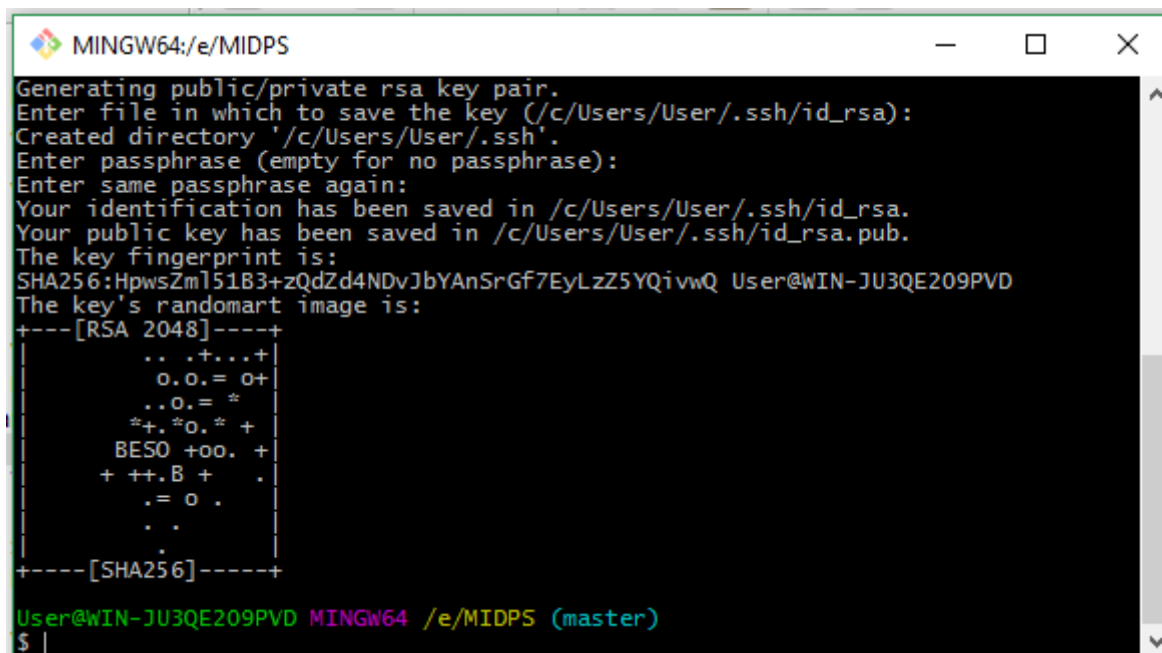
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$ git config --global user.name "burduleaFM"

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$ git config --global user.mail "alexandru.toloaca@gmail.com"

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$
```

Figure 4.2 – Crearea repozitorului utilizind comanda *git init*, [2]

Urmatorul pas este generarea cheii **SSH**. In terminalul bash scriem comanda **ssh-keygen**, iar cheia obtinuta o copiem in setarile noastre de pe github.



```
MINGW64:/e/MIDPS

Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/User/.ssh/id_rsa):
Created directory '/c/Users/User/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/User/.ssh/id_rsa.
Your public key has been saved in /c/Users/User/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Hpw5Zm151B3+zQdZd4NDvJbYAnSrGf7EyLzZ5YQivwQ User@WIN-JU3QE209PVD
The key's randomart image is:
+---[RSA 2048]---+
|  .. .+...+ |
|  o.o.= o+ |
|  ..o.= * |
|  *+.*o.* + |
|  BES0 +oo. + |
|  + ++.B + |
|  . = o . |
|  . . |
|  . |
+---[SHA256]-----+

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$ |
```

Figure 4.3 – Generarea ssh, [2]

Cheia obtinuta :

```
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDBxvPs5HX0PNDerLi3ZYzEQspE8gy5cyXw0ulW0ixYU
XYRTzpoTiaIfEUnJJ2tBUxcHc7bNlr8H45f9N3E5FP5ygC1OJ1Z03saM58P6ITc8KfjM7lwJJrnArmp0
Lc0IYZgUk8h0pBdwjLma1VZgpNKGOLxwmhwf31rKGJIMcXLVmSjz4LBuaD0SnEHiy5w+naz+Rfn/MtM3
3AoWrZNT4lfhZn4CgqHKE19awONb1Zs+aDYrQAONbunMo4cga/9ALHDCJZiOrFkOZFT+eAUeXZ+mzzmv
WD+dKzRyh1zyiF+JN6yv62LbCS0ICKMGPMF1c02A2QKvIjc0/fz2NcCZeSrt User@WIN-JU3QE209PVD
```

Figure 4.4– Codul general, [2]

<> Code ! Issues 0 🔑 Pull requests 0 📁 Projects 0 📖 Wiki 📡 Pulse 📊 Graphs

Options
Collaborators
Branches
Webhooks
Integrations & services
Deploy keys

Deploy keys

There are no deploy keys for this repository

Title

Ssh1

Key

AAAAB3NzaC1yc2EAAAADAQABAAQDBxvPs5HX0PNDerLi3ZYzEQspE8gy5cyXw0ulW0ixYU
nJJ2tBUxcHc7bNlr8H45f9N3E5FP5ygC1OJ1Z03saM58P6ITc8KfjM7lwJJrnArmp0
pNKGOLxwmhwf31rKGJIMcXLVmSjz4LBuaD0SnEHiy5w+naz+Rfn/MtM3
3AoWrZNT4lfhZn4CgqHKE19awONb1Zs+aDYrQAONbunMo4cga/9ALHDCJZiOrFkOZFT+eAUeXZ+mzzmv
WD+dKzRyh1zyiF+JN6yv62LbCS0ICKMGPMF1c02A2QKvIjc0/fz2NcCZeSrt User@WIN-JU3QE209PVD

☒ **Allow write access**
Can this key be used to push to this repository? Deploy keys always have push access.

Add key

Figure 4.5– Aplicarea codul ssh in setari, [2]

4.2 Setare branch to track remote origin

Dupa aceasta este necesar sa ne unim gitul nostru cu repozitoriul creat. Pentru realizarea acestei sarcini este putem folosi comanda **git remote add origin "linkul la repozitoriu"**

```
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$ git remote add origin https://github.com/burduleaFM/MIDPS.git
```

Figure 4.6– Aplicarea comenzii git remote origin, [2]

4.3 Utilizarea fisierului .gitignore

Vom crea preventiv citeva mape si fisiere pe care le vom adauga in repozitoriul nostru, dar inainte de asta vom realiza asa numitul fisier **.gitignore**.

Fisierul **.gitignore** este un fisier ce presupune o lista de fisiere si foldere care nu sunt puse in indexare si care nu vor participa la sharing. *Pentru exemplu vom crea un fisier si un folder pe care le vom include in .gitignore*

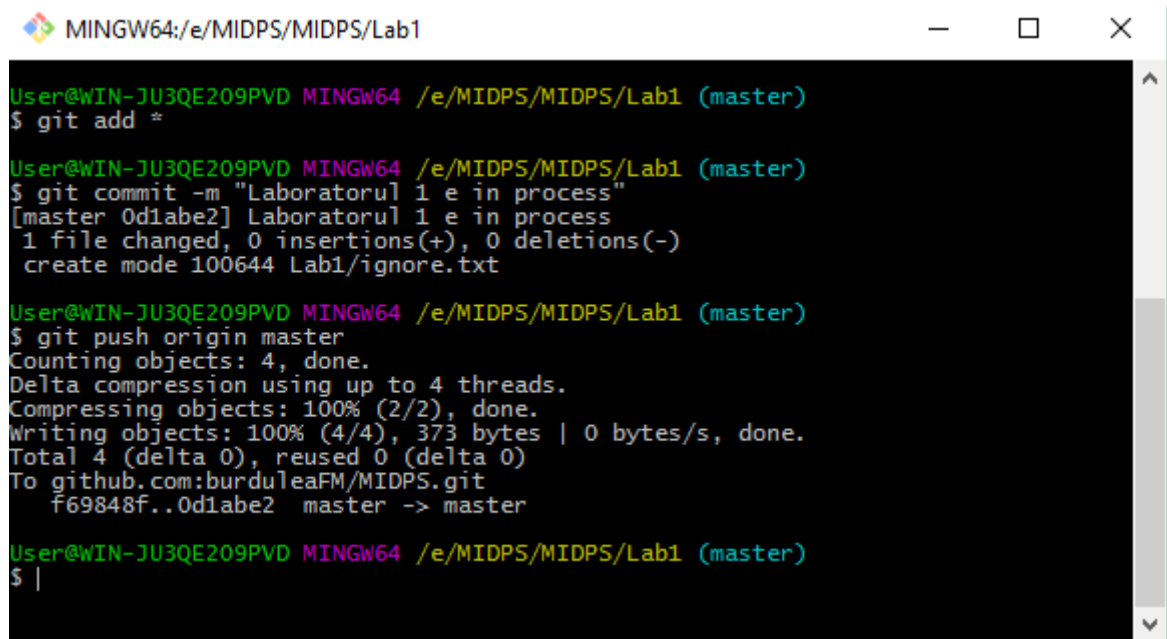
```
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$ start notepad .gitignore
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
$ cat .gitignore
ignore.txt
ignore/
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS (master)
```

Figure 4.7– Folderele si fisierele ce vor fi ignorate, [2]

4.4 Primul commit

Vom mai crea niste foldere si niste fisiere pe care mai tirziu le vom adauga in repository. Vom folosi comenzi cum sunt :

- **git add *** - care indexeaza toate fisierele.
- **git commit -m** - comanda care face imagine virtuala a tuturor schimbarilor din fisiere.
- **git push origin master** - comanda care incarca toate fisierele indexate pe git.



```
MINGW64:/e/MIDPS/MIDPS/Lab1
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git add *

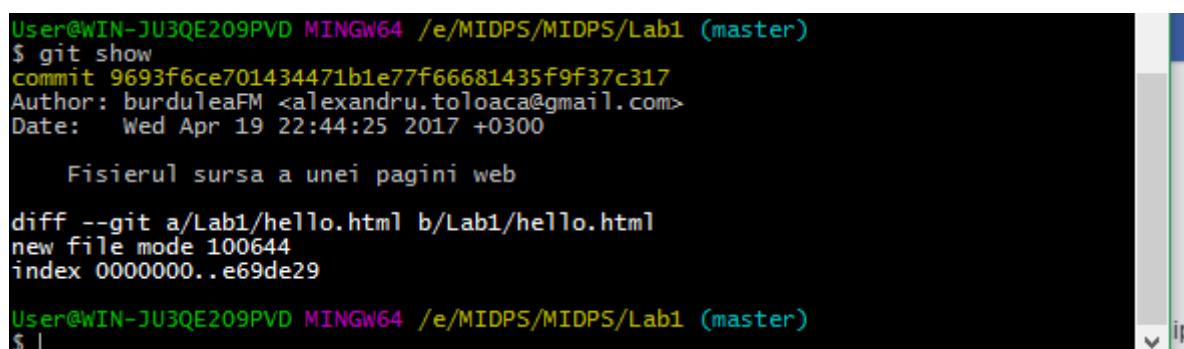
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git commit -m "Laboratorul 1 e in process"
[master 0d1abe2] Laboratorul 1 e in process
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Lab1/ignore.txt

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git push origin master
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 373 bytes | 0 bytes/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To github.com:burduleaFM/MIDPS.git
 f69848f..0d1abe2 master -> master

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ |
```

Figure 4.8– Rezultatele commitului, [2]

Pentru verificarea starii putem folosi comenzile **git status** si **git show**



```
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git show
commit 9693f6ce701434471b1e77f66681435f9f37c317
Author: burduleaFM <alexandru.toiloaca@gmail.com>
Date: Wed Apr 19 22:44:25 2017 +0300

    Fisierul sursa a unei pagini web

diff --git a/Lab1/hello.html b/Lab1/hello.html
new file mode 100644
index 0000000..e69de29

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ |
```

Figure 4.9– Git show, [2]

4.5 Crearea branch-urilor si adaugarea de fisiere pe fiecare

Pentru crearea unui branch este necesara utilizarea comenzii *git branch name*". Comanda *git branch* ne afiseaza branch-urile. Pentru a crea un nou branch si a face switch la el se foloseste comanda *git branch -b "name"*.

```
MINGW64:/e/MIDPS/MIDPS/Lab1

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git branch branch1

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git branch branch2

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git branch
  branch1
  branch2
* master

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ |
```

Figure 4.10 – Crearea branchurilor, [2]

Pentru adaugarea fisierelor pe fiecare dintre branch-urile create trebuie sa selectam pe cel dorit, aceasta o putem face cu comanda *git checkout* apoi adaugam fisierele cu comanda *git add* . si facem commit.

```
MINGW64:/e/MIDPS/MIDPS/Lab1

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (master)
$ git checkout branch1
Switched to branch 'branch1'

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (branch1)
$ git add test_branch1

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (branch1)
$ git commit -m "Commit pe branch1"
[branch1 a673782] Commit pe branch1
1 file changed, 1 insertion(+)
create mode 100644 Lab1/test_branch1

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (branch1)
$ git checkout branch2
Switched to branch 'branch2'

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (branch2)
$ git add test_branch2

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (branch2)
$ git commit -m "Commit pe branch2"
[branch2 758f85f] Commit pe branch2
1 file changed, 1 insertion(+)
create mode 100644 Lab1/test_branch2

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS/Lab1 (branch2)
$
```

Figure 4.11 – Adaugarea fisierelor pe fiecare branch, [2]

4.6 Resetarea la comitul anterior

Pentru resetarea la comitul anterior este necesar sa folosim comanda *git reset --hard HEAD*, si deasemenea si comanda *git log* pentru aflarea codului comitului. Resetarea la comitul anterior.


```

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS (branch1)
$ git log
commit a6737823ec6ea66b33bb78d6e87a39f1f3a71919
Author: burduleaFM <alexandru.toloaca@gmail.com>
Date: Sun May 21 13:43:52 2017 +0300

    Commit pe branch1

```

Figure 4.12– Logul, [2]

```

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS (branch1)
$ git reset --hard HEAD
HEAD is now at a673782 Commit pe branch1

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS (branch1)
$ git log
commit a6737823ec6ea66b33bb78d6e87a39f1f3a71919
Author: burduleaFM <alexandru.toloaca@gmail.com>
Date: Sun May 21 13:43:52 2017 +0300

    Commit pe branch1

commit 9693f6ce701434471b1e77f66681435f9f37c317
Author: burduleaFM <alexandru.toloaca@gmail.com>
Date: Wed Apr 19 22:44:25 2017 +0300

    Fisierul sursa a unei pagini web

commit 0d1abe27f49dfe8cad7b2f109218870d568592be
Author: burduleaFM <alexandru.toloaca@gmail.com>
Date: Wed Apr 19 22:38:53 2017 +0300

    Laboratorul 1 e in process

commit f69848f97a55aae49ebafb4cae77670fe5376308
Author: burduleaFM <alexandru.toloaca@gmail.com>

```

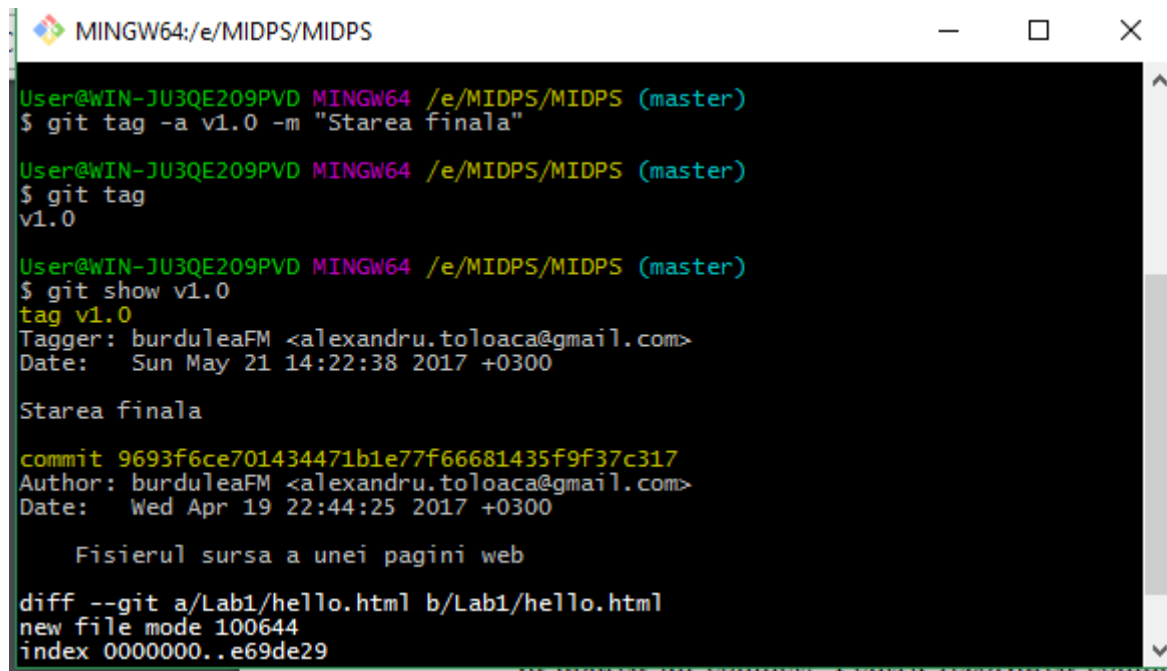
Figure 4.13– Reset, [2]

4.7 Merge 2 branches

Merge-ul este una din cele mai importante functii git care permite copierea tuturor fisierelor dintr-un branch in altul. Acest lucru permite mai multor echipe sa opereze cu acelasi proiect din branch-uri diferite. La realizarea aceste-i sarcini am facut merge intre 2 branch-uri. Toate acestea se pot realiza cu ajutorul comenzii *git merge 'numele branchului cu care se face merge'*

4.8 Folosirea tagurilor

Dupa ce am finalizat lucrul asupra repozitoriului ,am scris comanda `git tag -a v1.0 -m 'Final commit'` pentru a adauga un tag anotat.

A screenshot of a terminal window titled 'MINGW64:/e/MIDPS/MIDPS'. The terminal shows a series of commands and their outputs. The user is in the 'master' branch. They run 'git tag -a v1.0 -m "Starea finala"', then 'git tag', and finally 'git show v1.0'. The output of 'git show v1.0' displays the tag information, including the tag name 'v1.0', the tagger 'burduleaFM', the date 'Sun May 21 14:22:38 2017 +0300', the commit hash '9693f6ce701434471b1e77f66681435f9f37c317', the author 'burduleaFM', the date 'Wed Apr 19 22:44:25 2017 +0300', and the commit message 'Fisierul sursa a unei pagini web'. It also shows a diff indicating a new file 'a/Lab1/hello.html' was added.

```
MINGW64:/e/MIDPS/MIDPS
User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS (master)
$ git tag -a v1.0 -m "Starea finala"

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS (master)
$ git tag
v1.0

User@WIN-JU3QE209PVD MINGW64 /e/MIDPS/MIDPS (master)
$ git show v1.0
tag v1.0
Tagger: burduleaFM <alexandru.toloaca@gmail.com>
Date: Sun May 21 14:22:38 2017 +0300

Starea finala

commit 9693f6ce701434471b1e77f66681435f9f37c317
Author: burduleaFM <alexandru.toloaca@gmail.com>
Date: Wed Apr 19 22:44:25 2017 +0300

    Fisierul sursa a unei pagini web

diff --git a/Lab1/hello.html b/Lab1/hello.html
new file mode 100644
index 0000000..e69de29
```

Figure 4.14– Folosirea tagurilor, [2]

Concluzie

În urma acestui laborator am însușit mai aprofundat utilizarea sistemului de control al versiunilor git. Chiar dacă în majoritatea cazurilor nu lucrez în echipă asupra proiectelor, totuși este util utilizarea git, deoarece așa cu mult mai ușor pot introduce modificări în proiect și reveni la o stare anterioară, dacă am stricat ceva. De asemenea este comod când lucrezi pe mai multe stații diferite asupra aceluiași proiect. Am înțeles că pot folosi mai multe protocoale pentru conectarea la repositoryul remote, și deja mereu voi folosi protocolul ssh, deoarece el asigură o conexiune sigură și criptată, de asemenea și posibilitatea de a trimite schimbări pe repo remote.

References

- 1 Git how to?, *official page*, <https://githowto.com/ru>
- 2 Aldebran Robotics, *official page*, www.aldebaran.com/en