

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:
Cristina BRADU

lector asistent:
Irina COJANU
lector superior:
Radu MELNIC

Lucrarea de laborator nr.1

1 Scopul lucrarii de laborator

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

2 Obiective

Studierea Version Control Systems (git).

3 Analiza lucrarii de laborator

3.1 Cerintele implementate

Initializarea unui nou repozitoriu.

- Configurarea VCS.

- Commit, Push pe branch.

- Folosirea fisierului .gitignore.

- Revenire la versiunile anterioare.

- Crearea branch-urilor noi.

- Commit pe ambele branch-uri.

- Merge la 2 branchuri.

- Rezolvarea conflictelor.

3.2 Continutul lucrarii

Link-ul repozitoriului:

<https://github.com/BraduCristina/MIDPS>

Pentru a incepe lucrul cu git-ul, am configurat numele utilizatorului pe github.com si e-mail-ul meu, utilizind comenzile date de figure 1.

In mod scris, acestea arata astfel:

```
git config --global user.name "NumeUtilizator"  
git config --global user.email "email@mail.ru"
```

Pentru a configura numele si e-mailul, facem urmatoorii pasi:

Scriem urmatoarele comenzi: (figure 2)

git config global user.name "NUMELE"

git config global user.email EMAIL

git config --list afiseaza e-mail-ul si numele utilizatorului (figure 3).

git help commit ofera ajutor in lucrul cu git.

Analog este **git commit -help** (figure 4).

git status arata statutul

repozitoriului nostru.

Initial, pentru apelarea acestuia, este necesara indicarea mapei in care se afla fisierele noastre.

cd numemapa. In cazul nostru vom folosi instructiunea din figure 5.

cd MIDPS

Pentru a lucra cu fisierul `.gitignore`, initializam fisierele *README.md* si un *.gitignore*. Fisierul *README.md* il completam cu ceva informatie care dorim sa fie afisata, iar in fisierul `.gitignore` adaugam toate fisierele ce trebuie ignorate. (figure 6)

Fisierele noi create le incarcam pe repozitoriul nostru. Pentru aceasta vom avea nevoie de urmatoarele comenzi : (figure 7)

git add . - comanda indexeaza toate fisierele.

git commit -m "some text" - facem snapshot la comenzi.

git push- incarcam fisierele indexate pe git.

Pentru asigurarea corectitudinii pasilor facuti, utilizam comenzile din figure 8 si figure 9.

git status

git show

Una dintre caracteristicile principale ale unui VCS este faptul ca ne permite sa revenim la o versiune mai veche.

Aceasta poate fi efectuata cu ajutorul comenzii (figure 10) **git reset TYPE "codul commitului"**

Exista diferenta intre soft si hard , cind facem soft reset, indexurile ramin neschimbate.

Iar in cazul cind facem hard reset, pierdem indexurile.

Am creat un fisier nou *revert.txt* in versiunea 1. Dupa care l-am sters si am facut commit la versiunea 2, in care am sters fisierul *revert.txt* si dorim sa revenim la versiunea1. La inceput vom lansa comanda

git log

care ne arata logul de commituri si codul pentru fiecare commit. Pentru aceasta preluam primele 8 cifre de la commit-ul anterior (figure 11)

VCS ne permite sa avem mai multe branch-uri(ramuri). Branch-urile sunt utilizate mai des la lucrul in echipa sau cind lucram paralel la un proiect si apoi dorim sa unim toate modificarile. (figure 12)

git branch "name" - creeaza un branch nou cu numele "name".

git branch - vizualizarea branchurilor (* indica branchul curent).

git branch -d "nume" - sterge branchul "nume".

git checkout -b "name" - creeaza un branch nou cu numele "name" si face switch la el.

git checkout "nume" - face switch la branchul "nume".

Comenzi noi:

Operarea cu ele e prezentata in figure 13.

git branch -u upstream/name - face track la branchul indicat din branchul curent.

git branch -u upstream/name "nume" - face track din branchul "nume" la branch-ul indicat.

git branch track "name" upstream/name - creeaza branch-ul "name" si ii face track la branch-ul indicat.

git branch unset-upstream - scoate tracking-ul la branch-ul in care ne aflam.

(figure 13)

Putem avea conflicte in cazul cind dorim sa facem merge la 2 branchuri si unele rinduri sunt diferite. In asa caz ne vine in ajutor un mergetool.

Drept mergetool am ales *kdiff3*. Pentru a seta kdiff3 ca mergetool default, folosim comanda:

git config global merge.tool kdiff3

In continuare vom lucra cu 2 branchuri - "master" si "new".

Vom crea in fiecare branch cite un fisier "tomerge", continutul caruia va fi diferit.(figure 14)

In continuare vom incerca sa facem merge si sa rezolvam acest conflict. (figure 15)

4 Anexa 1 (figures)

```

703KM@Cristina-PC MINGW32 ~
$ git config --global user.name "BraduCristina"

703KM@Cristina-PC MINGW32 ~
$ git config --global user.email "cristina_bradu@mail.ru"
>

```

Figure 1: Configurare nume/email

```

703KM@Cristina-PC MINGW32 ~
$ cd MIDPS

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git config --global user.name "CristinaBradu"

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git config --global user.email "cristinabradu@mail.ru"

703KM@Cristina-PC MINGW32 ~/MIDPS (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=C:/Program Files/Git/mingw32/ssl/certs/ca-bundle.crt
diff.astextplain.textconv=astextplain
rebase.autosquash=true
credential.helper=manager
user.email=cristinabradu@mail.ru
user.name=CristinaBradu
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
remote.origin.url=git@github.com:BraduCristina/MIDPS.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.master.remote=origin
branch.master.merge=refs/heads/master
branch.new.remote=origin
branch.new.merge=refs/heads/master
branch.new_2.remote=origin
branch.new_2.merge=refs/heads/master
merge.tool=vimdiff
merge.conflictstyle=diff3
mergetool.prompt=false

```

Figure 2: Afisarea desfasurata a datelor utilizatorului

```

703KM@Cristina-PC MINGW32 ~
$ 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=C:/Program Files/Git/mingw32/ssl/certs/ca-bundle.crt
diff.astextplain.textconv=astextplain
rebase.autosquash=true
credential.helper=manager
user.email=cristina_bradu@mail.ru

user.name=BraduCristina

```

Figure 3: Afisarea datelor utilizatorului

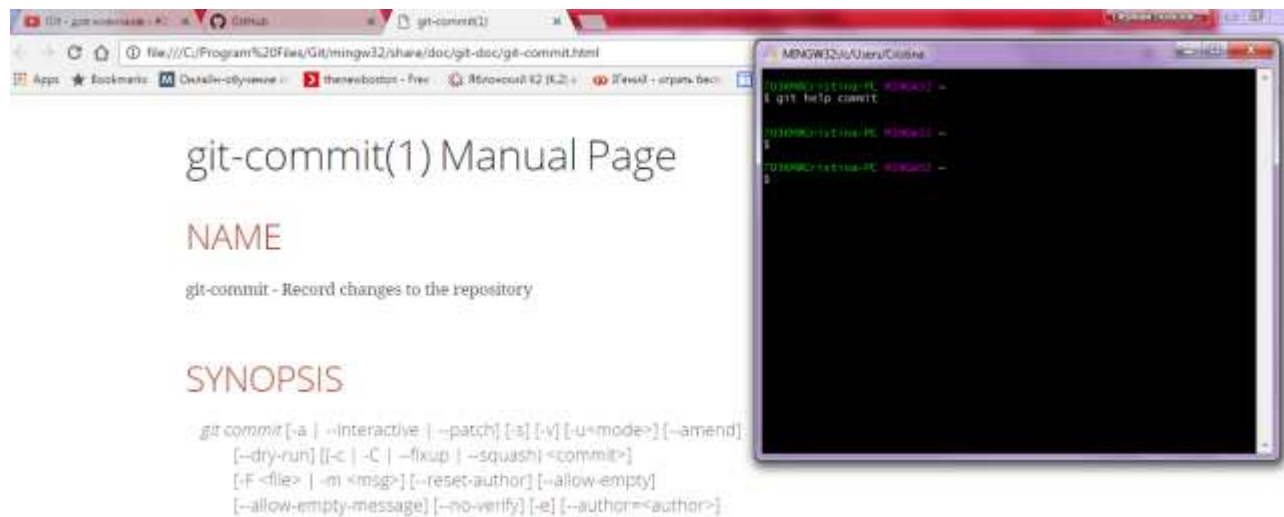


Figure 4: Ajutor in lucrul cu git

```
703KM@Cristina-PC MINGW32 ~  
$ cd MIDPS  
  
703KM@Cristina-PC MINGW32 ~/MIDPS (master)  
$ git status  
On branch master  
Your branch is up-to-date with 'origin/master'.  
nothing to commit, working tree clean
```

Figure 5: Afisarea statutului repozitoriului

```
cd  
703KM@Cristina-PC MINGW32 ~  
$ cd MIDPS  
  
703KM@Cristina-PC MINGW32 ~/MIDPS (master)  
$ vim README.md  
  
703KM@Cristina-PC MINGW32 ~/MIDPS (master)  
$ vim .gitignore
```

Figure 6: Adaugarea fisierelor .gitignore si README.md

```
703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git add *
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory.

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git commit -m "VCS learning"
[master bb8c603] VCS learning
1 file changed, 1 insertion(+), 1 deletion(-)

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git push origin master
To github.com:BraduCristina/MIDPS.git
```

Figure 7: Adaugarea fisierelor pe github.com

```
703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 3 commits.
  (use "git push" to publish your local commits)
nothing to commit, working tree clean
```

Figure 8: Verificarea corectitudinii pasilor


```
703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git show
commit bb8c6030db626db2598ae2fb632c4901b41f49cd
Author: CristinaBradu <crisinabrada@mail.ru>
Date: Tue Feb 21 13:37:27 2017 +0200

    VCS learning

diff --git a/README.md b/README.md
index f56f38f..c64a65f 100644
--- a/README.md
+++ b/README.md
@@ -1,1 @@
-# MIDPS
\ No newline at end of file
+# MIDPS
703KM@Cristina-PC MINGW32 ~/MIDPS (master)
```

Figure 9: Verificarea corectitudinii pasilor implementati

```

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ ls
hello.txt  MIDPS/  README.md  zoo/

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ vim revert.txt

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ ls
hello.txt  MIDPS/  README.md  revert.txt  zoo/

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git add *
warning: LF will be replaced by CRLF in revert.txt.
The file will have its original line endings in your working directory.

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git commit -m "test 1"
[master 2f875fe] test 1
1 file changed, 1 insertion(+)
create mode 100644 revert.txt

```

```

MINGW32:/c/Users/Cristina/MIDPS

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git pull
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 1), reused 2 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), done.
From github.com:BraduCristina/MIDPS
   447a7e3..b00868e  master    -> origin/master
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.

```

```
MINGW32:/c/Users/Cristina/MIDPS

703KM@Cristina-PC MINGW32 ~/MIDPS (master|MERGING)
$ git add .

703KM@Cristina-PC MINGW32 ~/MIDPS (master|MERGING)
$ git commit -m "Salut"
[master b485d13] Salut

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git push
Counting objects: 16, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (16/16), 1.70 KiB | 0 bytes/s, done.
Total 16 (delta 5), reused 0 (delta 0)
remote: Resolving deltas: 100% (5/5), done.
To github.com:BraduCristina/MIDPS.git
   b00868e..b485d13  master -> master

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ ls
hello.txt  Lab1/  Lab2/  Lab3/  Lab4/  Lab5/  MIDPS/  README.md  revert.txt  zoo/

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git rm revert.txt
rm 'revert.txt'

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ ls
hello.txt  Lab1/  Lab2/  Lab3/  Lab4/  Lab5/  MIDPS/  README.md  zoo/

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git add .

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git commit -m "test 2"
[master e342a92] test 2
1 file changed, 1 deletion(-)
```

```
703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git push
Counting objects: 2, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 249 bytes | 0 bytes/s, done.
Total 2 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local objects.
To github.com:BraduCristina/MIDPS.git
   b485d13..e342a92  master -> master
```

Figure 10: Revenirea la o versiune mai veche

```
703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git reset --hard e342a92
HEAD is now at e342a92 test 2

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ ls
hello.txt  Lab1/  Lab2/  Lab3/  Lab4/  Lab5/  MIDPS/  README.md  zoo/

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git reset --soft e342a92

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ ls
hello.txt  Lab1/  Lab2/  Lab3/  Lab4/  Lab5/  MIDPS/  README.md  zoo/
```

Figure 11: Log-ul de commit-uri

```
MINGW32/c/Users/Cristina/MIDPS
703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git branch test

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git branch
checkout
* master
  new
  new_2
  test

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git branch -d test
Deleted branch test (was e342a92).

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git checkout -b new3
Switched to a new branch 'new3'

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git branch
checkout
  master
  new
* new3
  new_2

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ ls
hello.txt  Lab1/  Lab2/  Lab3/  Lab4/  Lab5/  MIDPS/  README.md  zoo/

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git add .

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git add *

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git commit -m "new branch"
On branch new3
nothing to commit, working tree clean
```

```
703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git checkout new3
Already on 'new3'

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git push origin new
Total 0 (delta 0), reused 0 (delta 0)
gTo github.com:BraduCristina/MIDPS.git
dabc9df..83b9782 new -> new
i
703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git push origin new3
Total 0 (delta 0), reused 0 (delta 0)
To github.com:BraduCristina/MIDPS.git
* [new branch] new3 -> new3

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git add .

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git commit -m "Branch nou"
On branch new3
nothing to commit, working tree clean
```

Figure 12: Crearea si lucrul cu branch-urile

```

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git branch
  checkout
  master
  new
= new3
  new_2

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git checkout master
Your branch is up-to-date with 'origin/master'.
Switched to branch 'master'

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git checkout new3
Switched to branch 'new3'

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git branch -u origin/master
Branch new3 set up to track remote branch master from origin.

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git branch -u origin/master new3
Branch new3 set up to track remote branch master from origin.

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git branch --track "new4" origin/master
Branch new4 set up to track remote branch master from origin.

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git branch
  checkout
  master
  new
= new3
  new4
  new_2

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git checkout master
Your branch is up-to-date with 'origin/master'.
Switched to branch 'master'

```



```
703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git checkout master
Your branch is up-to-date with 'origin/master'.
Switched to branch 'master'

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git checkout new3
Your branch is up-to-date with 'origin/master'.
Switched to branch 'new3'

703KM@Cristina-PC MINGW32 ~/MIDPS (new3)
$ git checkout new4
Your branch is up-to-date with 'origin/master'.
Switched to branch 'new4'
```

Figure 13: Crearea si lucrul cu tracking


```

703KM@Cristina-PC MINGW32 ~/MIDPS (new4)
$ git checkout master
Your branch is up-to-date with 'origin/master'.
Switched to branch 'master'

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git branch
  checkout
* master
  new
  new3
  new4
  new_2

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ vim to_merge

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ cat to_merge
Branch merge

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git checkout new
Your branch is behind 'origin/master' by 6 commits, and can be fast-forwarded.
(use "git pull" to update your local branch)
Switched to branch 'new'

703KM@Cristina-PC MINGW32 ~/MIDPS (new)
$ git branch
  checkout
  master
* new
  new3
  new4
  new_2

703KM@Cristina-PC MINGW32 ~/MIDPS (new)
$ vim to_merge

703KM@Cristina-PC MINGW32 ~/MIDPS (new)
$ cat to_merge
Something different

```

Figure 14: Crearea fisierului tomerge

```

703KM@Cristina-PC MINGW32 ~/MIDPS (new)
$ git branch
  checkout
  master
* new
  new3
  new4
  new_2

703KM@Cristina-PC MINGW32 ~/MIDPS (new)
$ git checkout master
Your branch is up-to-date with 'origin/master'.
Switched to branch 'master'

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git branch
  checkout
* master
  new
  new3
  new4
  new_2

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git merge new
Already up-to-date.

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff kdiff3 tkdiff xxdiff meld tortoisemerge gvimdiff diffuze diffmerge ecmerge
rge araxis bc codecompare emerge vimdiff
No files need merging

```

```

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ ls
hello.txt  Lab1/  Lab2/  Lab3/  Lab4/  Lab5/  MIDPS/  README.md  to_merge  zoo/

703KM@Cristina-PC MINGW32 ~/MIDPS (master)
$ cat to_merge
Something different

```

Figure 15: Rezolvarea conflictului

Concluzie

În lucrarea de laborator nr.1 la disciplina MIDPS, am făcut cunostinta cu VCS (Version Control System). VCS reprezintă niste tool-uri software, care usurează lucrul în echipă asupra unui proiect propus. Gratie VCS, echipa dată își poate organiza modificările efectuate în cod în orice moment. Acesta salvează track-urile modificărilor, astfel, sistemul dat devenind destul de eficient și oferă posibilitatea de a lucra în paralel asupra unui proiect într-un timp mult mai scurt. Revenirea la o versiune mai veche la fel este un avantaj al VCS. Pentru a îndeplini lucrarea de laborator, am atins așa obiective, ca crearea fișierelor, editarea lor, crearea ramurilor și rezolvarea conflictelor aparute. Efectuarea commit-urilor la fel a fost un punct care urma să-l implementez, ceea ce am și reușit să fac. Toate comenzile care le-am utilizat, le-am îndeplinit pe Windows 7.