

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:*  
Ernest BITCA

*lector asistent:*  
Irina COJANU  
*lector superior:*  
Radu MELNIC

# **Lucrarea de laborator #1**

## **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

Initializarea unui nou repositoryu.  
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 Analiza Lucrării de laborator

Linkul la repository <https://github.com/Ernest96/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**.

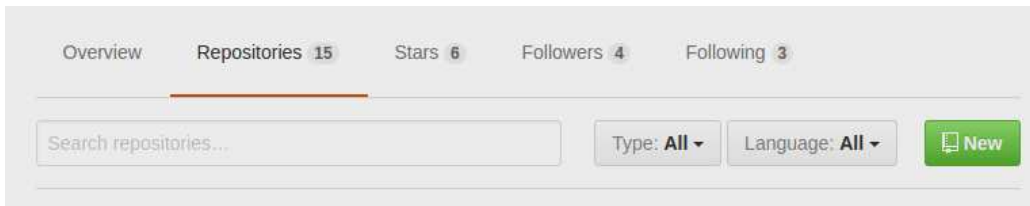
Urmatorul pas este crearea insusi a noului repository pe care il vom crea utilizind urmatoarea comanda **curl -u 'USER' https://api.github.com/user/repos -d '{"name":"NUME"}'**. Unde cuvintele scrise cu CAPS se vor inlocui cu numele utilizatorului si numele repositoryului.

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



```
ernest@Lenovo: ~/testam
ernest@Lenovo:~$ cd testam/
ernest@Lenovo:~/testam$ git init
Initialized empty Git repository in /home/ernest/testam/.git/
ernest@Lenovo:~/testam$ curl -u 'Ernest96' https://api.github.com/user/repos -d '{"name":"test"}'
ernest@Lenovo:~/testam$ git remote add origin https://github.com/Ernest96/test.git
ernest@Lenovo:~/testam$
```

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 const in mai multe etape. La inceput vom configura numele si emailul. Scriem urmatoarele comenzi:

```
git config --global user.name "NUMELE"
```

```
git config --global user.email EMAIL
```

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ git config --global user.name "Ernest96"
ernest@Lenovo:~/MIDPS/LAB1$ git config --global user.email "bitca@inbox.ru"
ernest@Lenovo:~/MIDPS/LAB1$ git config --list
user.name=Ernest96
user.email=bitca@inbox.ru
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
remote.origin.url=https://github.com/Ernest96/MIDPS.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
```

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 repositoryul nostru cu un fisier **README.md** si un **.gitignore**. In fisierul README.md vom adauga niste informatie pentru cei care se vor folosi de repository iar in fisierul .gitignore vom adauga toate fisierele ce trebuiesc ignorate (adica sa nu fie incarcate).

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ vim README.md
ernest@Lenovo:~/MIDPS/LAB1$ vim .gitignore
ernest@Lenovo:~/MIDPS/LAB1$ cat README.md
# MIDPS
Ernest Bitca TI-153
ernest@Lenovo:~/MIDPS/LAB1$ cat .gitignore
#fisierele ce vor fi ignorate :
ignore.txt
#mapa ce va fi ignorata :
ignore/
ernest@Lenovo:~/MIDPS/LAB1$ |
```

Vom adauga fisierele noi create pe repositoryul nostru. Pentru aceasta vom avea nevoie de 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.

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ git add *
ernest@Lenovo:~/MIDPS/LAB1$ git commit -m "I'm kinda slow with LaTeX"
[master 8568023] I'm kinda slow with LaTeX
 6 files changed, 29 insertions(+), 262 deletions(-)
rewrite .gitignore (100%)
create mode 100644 LAB1/TeX_SRC/Commit.tex
ernest@Lenovo:~/MIDPS/LAB1$ git push origin master
Username for 'https://github.com': Ernest96
Password for 'https://Ernest96@github.com':
Counting objects: 10, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (10/10), done.
Writing objects: 100% (10/10), 13.07 KiB | 0 bytes/s, done.
Total 10 (delta 5), reused 0 (delta 0)
remote: Resolving deltas: 100% (5/5), completed with 5 local objects.
To https://github.com/Ernest96/MIDPS.git
 855cc10..8568023 master -> master
ernest@Lenovo:~/MIDPS/LAB1$ |
```

Pentru a ne asigura ca am facut totul bine si nu avem probleme vom utiliza **git status** si **git show**

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ git status
On branch master
nothing to commit, working directory clean
ernest@Lenovo:~/MIDPS/LAB1$ git show
commit 856802310ff3e15d70c62532ce40ab816ed4e242
Author: Ernest96 <bitca@inbox.ru>
Date: Tue Feb 7 22:19:07 2017 +0200

    I'm kinda slow with LaTeX

diff --git a/.gitignore b/.gitignore
index fle3d20..9579a0f 100644
--- a/.gitignore
+++ b/.gitignore
@@ -1,252 +1,4 @@
```

Una dintre caracteristicile principale a unui VCS este faptul ca ne permite sa revenim la o versiune mai veche. Aceasta poate fi efectuata cu ajutorul comenzii **git reset -TYPE "codul comitului"**. Exista diferenta intre **-soft** si **-hard** , cind facem soft reset indexurile ramn neschimbate. Iar in cazul cind facem hard reset , pierdem indexurile.

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ ls
ignore  ignore.txt  LAB1  LAB2  LAB3  LAB4  LAB5  README.md
ernest@Lenovo:~/MIDPS/LAB1$ vim revert.txt
ernest@Lenovo:~/MIDPS/LAB1$ ls
ignore  ignore.txt  LAB1  LAB2  LAB3  LAB4  LAB5  README.md  revert.txt
ernest@Lenovo:~/MIDPS/LAB1$ git add *
The following paths are ignored by one of your .gitignore files:
ignore
ignore.txt
Use -f if you really want to add them.
ernest@Lenovo:~/MIDPS/LAB1$ git commit -m "version 1"
[copy 758067c] version 1
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 revert.txt
ernest@Lenovo:~/MIDPS/LAB1$ git push origin master
Username for 'https://github.com': Ernest96
Password for 'https://Ernest96@github.com':
Everything up-to-date
ernest@Lenovo:~/MIDPS/LAB1$ ls
ignore  ignore.txt  LAB1  LAB2  LAB3  LAB4  LAB5  README.md  revert.txt
ernest@Lenovo:~/MIDPS/LAB1$ git rm revert.txt
rm 'revert.txt'
ernest@Lenovo:~/MIDPS/LAB1$ ls
ignore  ignore.txt  LAB1  LAB2  LAB3  LAB4  LAB5  README.md
ernest@Lenovo:~/MIDPS/LAB1$ git add *
The following paths are ignored by one of your .gitignore files:
ignore
ignore.txt
Use -f if you really want to add them.
ernest@Lenovo:~/MIDPS/LAB1$ git commit -m "version 2"
[copy 05e4b55] version 2
1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100644 revert.txt
ernest@Lenovo:~/MIDPS/LAB1$ git push origin master
```

Dupa cum observam cind am facut commit, s-au ignorat fisierele incluse in .gitignore.

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 dorim sa revenim la versiunea1. La inceput vom lansa comanda **git --log** care ne arata logul de commituri si codul pentru fiecare commit. Vom avea nevoie de primele 7 cifre la commitul anterior.

```
ernest@Lenovo: ~/MIDPS/LAB1
commit 05e4b55cbe61834412dc653fc0e2d4be538b4888
Author: Ernest96 <bitca@inbox.ru>
Date: Wed Feb 8 10:39:05 2017 +0200

    version 2

commit 758067c05055045980360f4d880b6f4909c66ca2
Author: Ernest96 <bitca@inbox.ru>
Date: Wed Feb 8 10:38:18 2017 +0200

    version 1
```

Acum vom folosi comenzile descrise anterior.

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ git reset --hard 758067c0
HEAD is now at 758067c version 1
ernest@Lenovo:~/MIDPS/LAB1$ ls
ignore  ignore.txt  LAB1  LAB2  LAB3  LAB4  LAB5  README.md  revert.txt
ernest@Lenovo:~/MIDPS/LAB1$ git reset --soft 758067c0
ernest@Lenovo:~/MIDPS/LAB1$ ls
ignore  ignore.txt  LAB1  LAB2  LAB3  LAB4  LAB5  README.md  revert.txt
```

Daca am facut niste schimbari in repozitoriu si nu ne satisfac, putem usor reveni la ultima versiune care era pe git utilizind comanda **git pull origin "branch"** Comanda **git pull** va face update la repozitoriul nostru pina la ultima versiune.

VCS ne permite sa avem mai multe **branchuri**. Din traducere branch semnifica "creanga". Branchurile sunt foarte 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 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.

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ git branch copie
ernest@Lenovo:~/MIDPS/LAB1$ git branch
copie
* master
ernest@Lenovo:~/MIDPS/LAB1$ git branch -d copie
Deleted branch copie (was 45b29f0).
ernest@Lenovo:~/MIDPS/LAB1$ git checkout -b nou
D      revert.txt
Switched to a new branch 'nou'
ernest@Lenovo:~/MIDPS/LAB1$ git branch
master
* nou
ernest@Lenovo:~/MIDPS/LAB1$ ls
ignore ignore.txt LAB1 LAB2 LAB3 LAB4 LAB5 README.md
ernest@Lenovo:~/MIDPS/LAB1$ git add *
The following paths are ignored by one of your .gitignore files:
ignore
ignore.txt
Use -f if you really want to add them.
ernest@Lenovo:~/MIDPS/LAB1$ git commit -m "branch nou"
[nou b398b51] branch nou
1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100755 revert.txt
ernest@Lenovo:~/MIDPS/LAB1$ git push origin nou
Username for 'https://github.com': Ernest96
Password for 'https://Ernest96@github.com':
Counting objects: 2, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 213 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 https://github.com/Ernest96/MIDPS.git
 * [new branch]      nou -> nou
ernest@Lenovo:~/MIDPS/LAB1$
```



**git checkout "nume"** - face switch la branchul "nume".  
**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 branchul indicat.  
**git branch --track "name" upstream/name** - creeaza branchul "name" si ii face track la branchul indicat.  
**git branch --unset-upstream** - scoate trackingul la branchul in care ne aflam.

```
ernest@Lenovo: ~/MIDPS/LAB1
ernest@Lenovo:~/MIDPS/LAB1$ git branch
  master
* nou
ernest@Lenovo:~/MIDPS/LAB1$ git checkout master
Switched to branch 'master'
ernest@Lenovo:~/MIDPS/LAB1$ git checkout nou
Switched to branch 'nou'
ernest@Lenovo:~/MIDPS/LAB1$ git branch -u origin/master
Branch nou set up to track remote branch master from origin.
ernest@Lenovo:~/MIDPS/LAB1$ git branch -u origin/master nou
Branch nou set up to track remote branch master from origin.
ernest@Lenovo:~/MIDPS/LAB1$ git branch --track "nou_2" origin/master
Branch nou_2 set up to track remote branch master from origin.
ernest@Lenovo:~/MIDPS/LAB1$ git branch
  master
* nou
  nou_2
ernest@Lenovo:~/MIDPS/LAB1$ git checkout master
Switched to branch 'master'
ernest@Lenovo:~/MIDPS/LAB1$ git checkout nou
Switched to branch 'nou'
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)
ernest@Lenovo:~/MIDPS/LAB1$ git checkout nou_2
Switched to branch 'nou_2'
Your branch is up-to-date with 'origin/master'.
ernest@Lenovo:~/MIDPS/LAB1$ |
```