## 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 Onta Vladisla
A verificat:	Cojanu Irina

### **Objective:**

• 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:
  - 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 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.

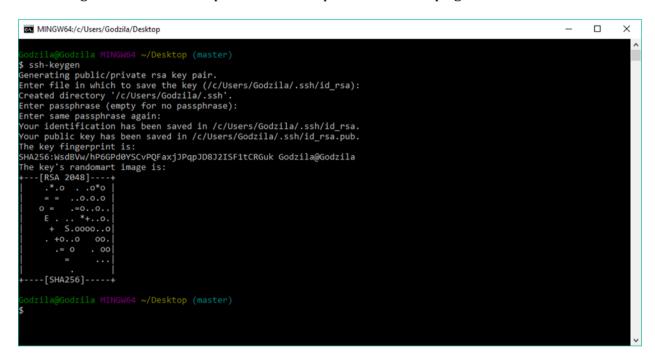
Următorul pas constă în configurarea git-ului.Configurăm numele și email-ul prin comenzile git config –global user.name "NUMELE" git config –global user.email "EMAIL".

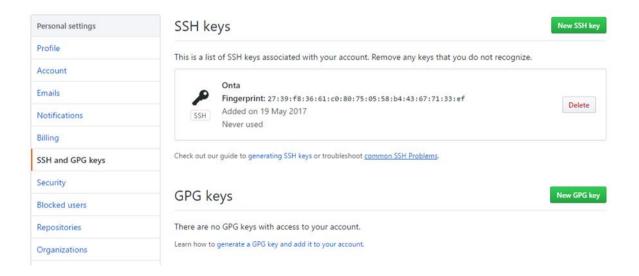
```
Godzila@Godzila MINGW64 ~/Desktop (master)
$ git config --global user.name "Onta"

Godzila@Godzila MINGW64 ~/Desktop (master)
$ git config --global user.email "vladislav.onta@gmail.com"

Godzila@Godzila MINGW64 ~/Desktop (master)
$ git config --global user.email "vladislav.onta@gmail.com"
```

Urmează generarea 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.

```
MINGW64/c/Users/Godzila/Desktop/MIDPS (master)

$ git add *

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)

$ git commit
[master b9a0a87] Incarcare reusita!

$ files changed, $ insertions(+)
create mode 100644 lab 1/README.md
create mode 100644 lab 2/README.md
create mode 100644 lab 3/README.md
create mode 100644 lab 5/README.md
create mode 100644 lab 5/README.md

create mode 100644 lab 5/README.md

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)

$ git push origin master

Counting objects: 2, done.

Delta compression using up to 2 threads.

Compression using up to 2 threads.

Compression objects: 100% (2/2), 284 bytes | 0 bytes/s, done.

Writing objects: 100% (2/2), 284 bytes | 0 bytes/s, done.

Total 2 (delta 0), reused 0 (delta 0)

To https://github.com/VladislavOnta/MIDPS.git

4205522..b9a0a87 master -> master

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)

$ ______

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)
```

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 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.

```
    ™ MINGW64/c/Users/Godzila/Desktop/MIDPS
    wodzila@Godzila MINGW64 ~/Desktop/MIDPS (master)
    $ git branch
    * master
    versiune1

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)
    $ git branch - d "versiune1"
    Deleted branch versiune1 (was b9a@a87).

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)
    $ git branch "origin"

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)
    $ git branch
    * master
    origin

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)
    $ git checkout - b 'new'
    Switched to a new branch 'new'

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (new)
    $ git branch
    master
    * new
    origin

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (new)

$ git branch
    master
    * new
    origin

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (new)

$ codzila@Godzila MINGW64 ~/Desktop/MIDPS (new)
```

```
MINGW64:/c/Users/Godzila/Desktop/MIDPS (new)
$ git branch
master
* new
origin

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (new)
$ git checkout master
Your branch is up-to-date with 'origin/master'.
Switched to branch 'master'
$ git checkout new
$ switched to branch 'moster'
$ git checkout new
$ Switched to branch 'new'

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (new)
$

$ git checkout new
$ Switched to branch 'new'

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (new)
$
```

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

```
Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)

$ git push origin
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 298 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/VladsilavOnta/MIDPS.git
b9a0a87..d8b884a master -> master

Godzila@Godzila MINGW64 ~/Desktop/MIDPS (master)

$
```

```
EW BuGpam MINGWG&/c/Users/Godila/Desktop/MIDPS (master)

Sgit checkout new
Switched to branch 'new'

Godzila@Godzila MINGWG& ~/Desktop/MIDPS (new)

Sgit commit -m "hello"

On branch new
nothing to commit, working tree clean

Godzila@Godzila MINGWG& ~/Desktop/MIDPS (new)

Sgit checkout master

Your branch is up-to-date with 'origin/master'.

Switched to branch 'master'

Sgit commit -m "buna seera"

On branch master

Your branch is up-to-date with 'origin/master'.

nothing to commit, working tree clean

Godzila@Godzila MINGWG& ~/Desktop/MIDPS (master)

Sgit push origin

Everything up-to-date

Godzila@Godzila MINGWG& ~/Desktop/MIDPS (master)

Sgit push origin

Everything NINGWG& ~/Desktop/MIDPS (master)
```

**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 status list which (unstaged) files have changed git diff list (unstaged) changes to files git log list recent commits git add fn stage 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.
```