Ministerul Educației al Republicii Moldova Universitatea Tehnică a Moldovei

Catedra: Automatica și Tehnologii Informaționale

RAPORT

Lucrare de laborator Nr.2 *la M.I.D.P.S.*

A efectuat: st. gr. TI-144

I. Stratan

A verificat: lector.univ.

I.Cojanulector.sup.S.Cojocaru

Chişinău 2016

LUCRAREA DE LABORATOR 2

Objective:

- Intelegerea si folosirea CLI (basic level)
- Administrarea remote a masinilor linux machine folosind SSH (remote code editing)
- Version Control Systems (git || mercurial || svn)
- Compileaza codul C/C++/Java/Python prin intermediul CLI, folosind compilatoarele gcc/g++/javac/python

Laboratory Requirements:

- *Basic Level* (nota 5 || 6):
 - o conecteaza-te la server folosind SSH
 - o compileaza cel putin 2 sample programs din setul HelloWolrdPrograms folosind CLI
 - o executa primul commit folosind VCS
- *Normal Level* (nota 7 || 8):
 - 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)
- *Advanced Level* (grade 9 || 10):
 - 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 merge 2 branches
 - o conflict solving between 2 branches
- Bonus Point:
 - o Scrie un script care va compila HelloWolrdPrograms projects.

Taskurile folosite:

```
git config --global user.name "You name"
git config -global user.email "You email"
git config --list
cd ~/. ssh
ssh-keygen
cat ~/.ssh/id_rsa.pub
git init
git status
git clone
git add.
git commit -m ''message ''
git branch name
git checkout nume
git branch –a
git branch -vv
git push origin master
```

Analiza lucrări de laborator :

Linkul meu https://github.com/StratanIon/MIDPSv1

• Am folosit aceste două comnezi pentru a configura github, adica am introdus numele meu de pe github și pe care email a fost înregistrat acest acaunt.

git config --global user.name "You name" git config --global user.email "You email"

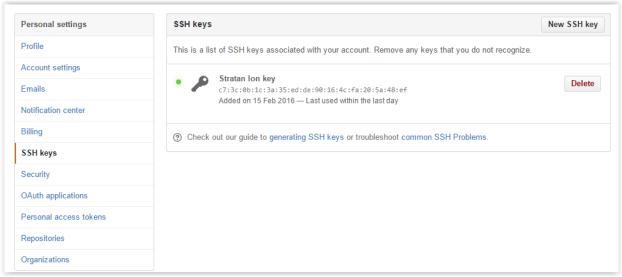
- Am folosit această comandă git config --list pentru a vedea lista de configurări.
- Aceste trei comenzi c d ~/. ssh , ssh-keygen , cat ~/.ssh/id_rsa.pub le-am folosit pentru a genera o cheie ssh și ultima comandă îmi afișează cheia ssh , care am introdus-o pe github . Această cheie ne dă posibilitatea de a lucra cu repozitoriu nostru în oricare timp , fără a introduce de fiecare dată numele și parola la github.
- Comanda git init inițializează mapa noastră ,care crează o mapă nevizibilă cu numele .git , unde în această mapă conține toate resursele pe care le folosim .
- Comanda git status ne reprezinta statutul mapei noastre în momentul de timp dat.
- Comanda git clone, realizează clonarea mapei noastre, sau poate să cloneze alte fișiere de pe github.
- Comanda git add., adauga in local server fişierele noi. Punctul reprezintă adaugarea tuturor fişierilor, dar ca să adăugăm un singur fişier trebuie sa scrim git add numele fisierului.
- Comanda git commit –m "", crează un commit cu mesajul dat, adică ne arată ce sa modificat într-un interval de timp.
- Comanda git branch, reprezinta niște ramuri a proiectului nostru, ramura principală este master, și putem să creem multe ramuri. Fiecare ramură reprezintă niște schimbări ale noastre care nu poate să influențeze direct la proiectul principal. Adica fiecare programist își are misiunea lui și firecare program creat de el asupra proiectului este inclus în branch lui.
- Comanda git checkout nume, aici se realizează interschimbările între branchuri.
- Comanda git branch –a , ne arată totalitatea branchurilor.
- Comanda git branch --v, ne arată totalitatea branchurilor cu comitele lor respective.
- Comanda git push origin master, realizază introducerea fișierilor pe github din mapa ta locală.

Conectare la server folosing SSH:

Folosesc comanda : cd ~/. ssh , apoi ssh-keygen , pentru a genera o cheie.

Ca sa obțin cheia scriu : cat ~/.ssh/id_rsa.pub

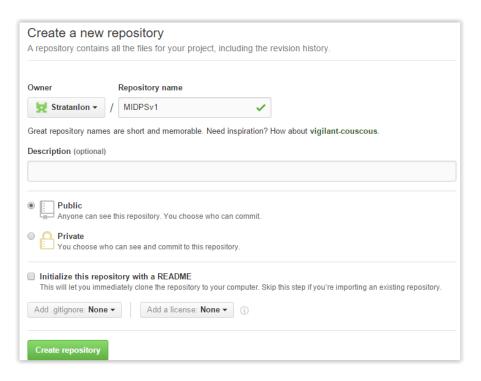
Şi plasez cheia pe Github:

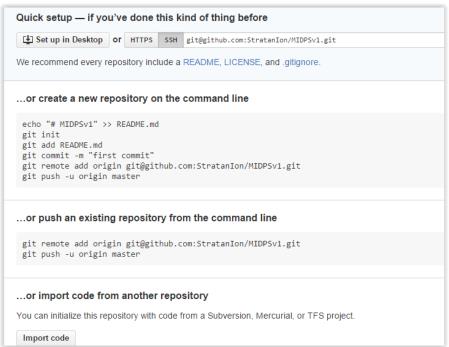


Execut primul comit:

```
ION@Jony MINGW64 /e/MIDPS (master)
$ git commit -m "Primul meu commit"
[master (root-commit) da20651] Primul meu commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 First.txt
```

Inițializez un nou repozitoriu:





Configurez VCS:

```
ION@Jony MINGW64 /e/MIDPS (master)

$ git config --global user.name "StratanIon"

ION@Jony MINGW64 /e/MIDPS (master)

$ git config --global user.email "95stratosfera@mail.ru"

ION@Jony MINGW64 /e/MIDPS (master)

$ git config --list
core.symlinks=false
core.autocrlf=true
color.diff=auto
color.status=auto
color.status=auto
color.branch=auto
color.interactive=true
help.format=html
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
diff.astextplain.textconv=astextplain
```

Creiez două branchuri:

ebase.autosquash=true

user.name=StratanIon

user.email=95stratosfera@mail.ru

core.repositoryformatversion=0 core.filemode=false core.bare=false core.logallrefupdates=true core.symlinks=false

core.ignorecase=true core.hidedotfiles=dotGitOnly

```
ION@Jony MINGW64 /e/MIDPS (master)
$ git branch one-ver

ION@Jony MINGW64 /e/MIDPS (master)
$ git branch two-ver

ION@Jony MINGW64 /e/MIDPS (master)
$ git branch -a
# master
one-ver
two-ver
```

Commit la aceste două branchuri:

```
ION@Jony MINGW64 /e/MIDPS (one-ver)
$ git commit -m "text nou"
[one-ver 2a54312] text nou
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 ab.txt
```

```
ION@Jony MINGW64 /e/MIDPS (two-ver)
$ git commit -m "text in al doilea branch"
[two-ver 3e525a8] text in al doilea branch
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 ba.txt
```

Concluzie

În urma efctuării acestui laborator , am înțeles cum funcționează GITHUB. M-am folosit de unele comenzi pentru a îndeplini sarcinele , comenzile nu sunt grele. Acest mediu este foarte bun, deoarece este folosit la proiecte mari în companii, și e comod de a lucre cu el, pentru că îți aduce folos și îți economisește mult timp.