

## - Principi softverskog inženjerstva -

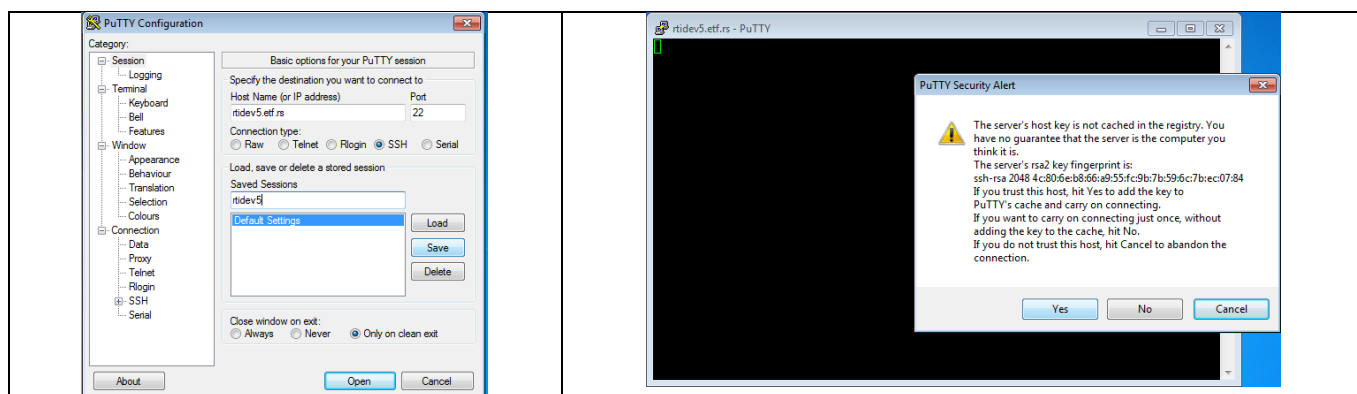
### Lab. vežba br. 1: Sistemi kontrole verzija

Cilj ove laboratorijske vežbe je da demonstrira upotrebu alata za kontrolu verzija (*eng. Version Control System*), kroz GIT i SVN.

#### Vežba 1. Upotreba PUTTY alata, menjanje password-a na Linux-u, pravljenje i listanje direktorijuma

Putty je alat koji se koristi za konekciju i rukovanje Linux računarom sa Windows operativnog Sistema.

1. Pokrenuti Putty
2. Popuniti sve kao na slici:
  - a. Host Name: rtidev5.etf.rs
  - b. Saved Sessions: rtidev5 (ili proizvoljno ime)
3. Odabrati open. Prikazaće se prozor kao na slici 2. Odabrati „yes“ (prikazuje se samo prvi put)
4. Uneti studentski username (piGGBBBBd) i lozinku „si3psi“ (ili koju ste ranije koristili)
5. (OPCIONO) Otkucati komandu **passwd**, namenjenu promeni lozinke. Traži se da unesete tekuću lozinku („password“) a potom željenu novu lozinku
6. Otkucati sledeće komande, jednu po jednu (razdvojene su zaptetama):  
**ls, ls -a, ls -l, ls -la.**
7. Otkucati **mkdir test**. Potom **ls -la**.



#### Vežba 2. Inicijalizacija praznog GIT repozitorijuma

U kreiranom direktorijumu **test** inicijalizovati GIT repozitorijum naredbom **init**:

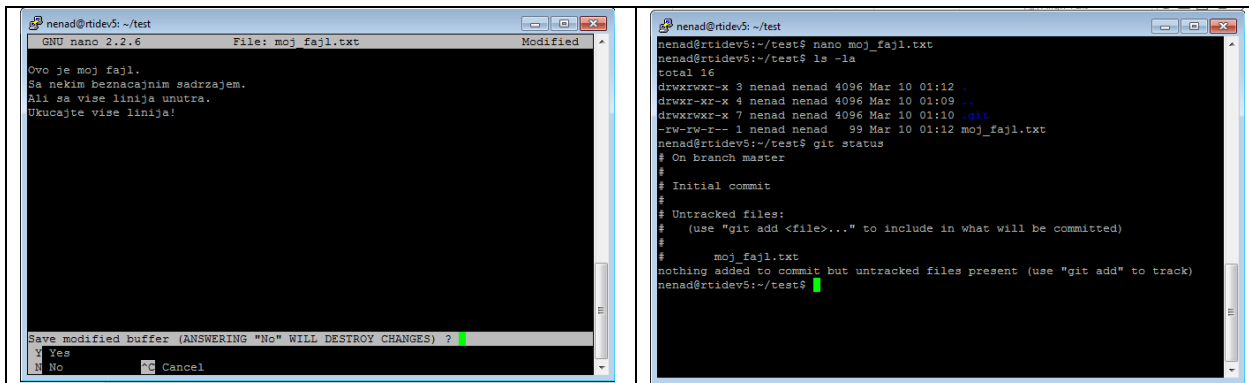
- **cd test # prelazi u direktorijum test**
- **git init # inicijalizuje prazan repozitorijum**
- **ls -la # prikaz svih fajlova u direktorijumu**

Dalji tok rada: ili „dovući“ postojeći repozitorijum, ili krenuti sa ubacivanje sadržaja u ovaj novi repozitorijum.

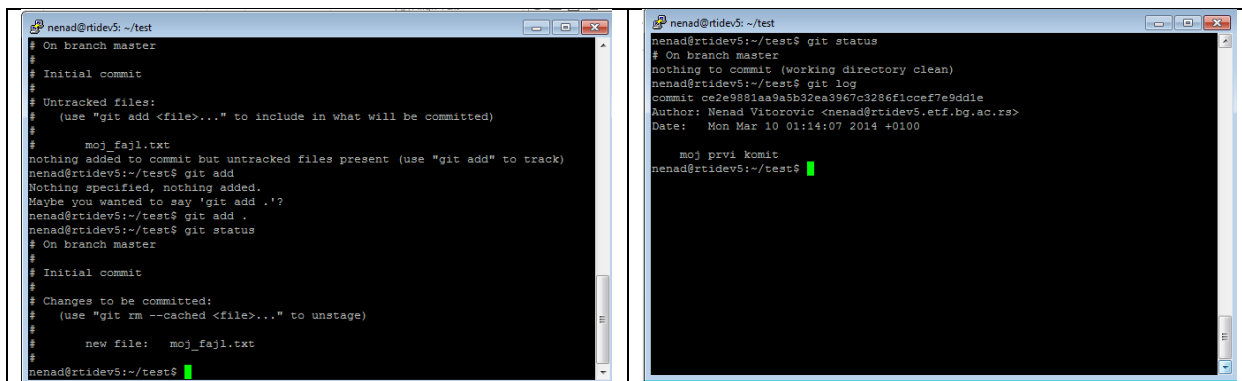
#### Vežba 3. Kreiranje tekstualnog fajla, pregled stanja repozitorijuma i prva naredba commit

- Izvršiti **git status**
  - Nema ničega izmenjenog
- Pokrenuti konzolni editor **nano** (ili neki drugi ako želite): **nano moj\_fajl.txt**. Uneti tekst kao na slici 1.
- Pritisnuti kombinaciju **CTRL+X** (videti da je „exit“ **^+X**). Na pitanje odgovoriti sa **y** ili **yes** a zatim potvrditi da se izmene čuvaju u **moj\_fajl.txt** (priskomom na enter)
- Izvršiti komandu **ls -la**

- Izvršiti komandu `git status`
- Rezultati ovih komandi su na slici 2



- Dodavanje fajlova: priprema za verzionisanje.  
Izvršiti komandu `git add --a` (dve crtice). Izvršiti `git status`. Rezultat je na slici 3.
- Izvršiti komandu `git commit -m "moj prvi komit"`.  
Potom ponovo `git status`.
- Izvršiti komandu `git log`. Rezultat je kao na slici 4



- Ponoviti proces nekoliko puta:  
nano (izmeniti fajl), status, log, add, status, log, commit, status, log. Pratiti šta se od fajlova promenilo. Na kraju, pokrenuti log sa opcijom `--graph`. U čemu je razika?

Pitanja:

- Koja je namena naredbe `git add`? Kada se ona primenjuje i koje su njene posledice?
- Koja je namena naredbe `git status`? Kakve informacije daje?
- Koja je namena naredbe `git commit`? Kakve informacije daje? Kakve su njene posledice?
- Koja je namena naredbe `git log`? Kakve informacije daje? Ima li posledice?

Radni direktorijum je repozitorijum koji se koristi u lokalu. Za sinhronizaciju sa drugim saradnicima na projektu, neophodna je upotreba deljenog repozitorijuma. Deljeni repozitorijum napraviti uz pomoć GitHub ili BitBucket.

Za ostatak vežbe, radiće se na Windows-u, a rad sa repozitorijumom će se odvijati preko nekog od instaliranih alata za Windows.

Priprema:

1. Pokrenuti alat Git Bash (START > All Programs > GIT).
2. Napraviti direktorijum `git-lab` i preći u njega (`mkdir git-lab` `cd git-lab`).

#### **Vežba 4. Rad sa repozitorijumom i udaljenim repozitorijumom. Pojavljivanje i razrešavanje konflikta.**

1. Inicijalizovati repozitorijum komandom  
`git init`
2. Kreirati tekstualni fajl u direktorijumu git-lab pod nazivom readme.txt
3. Proveriti sadržaj repozitorijuma:  
`git status`
4. Dodati fajl u repozitorijum:  
`git add readme.txt`
5. Proveriti sadržaj repozitorijuma:  
`git status`
6. Videti Code stage (šta je to što treba da se commituje?)
7. Izmeniti tekstualni fajl readme.txt , pa ponovo uraditi: `git status`
8. Izvršiti:  
`git add readme.txt`  
`git config --global user.mail vas_email@student.etf.rs`  
`git config --global user.name "Vase ime"`
9. Izvršiti: `git commit`  
Izmeniti prvi red tekstualnog fajla.  
Snimiti fajl:  
:exit ili :wq
10. Proveriti sadržaj:  
`git status`  
Da li ima promena?
11. Izmeniti ponovo tekstualni fajl.
12. `git add readme.txt`
13. `git commit -m 'Ovo je poruka uz commit'`
14. `git log`
15. Ubaciti u direktorijum fajl index.html
16. Izmeniti fajl: readme.txt  
  
Uraditi komandu: `git status`
17. Izvršiti komandu:  
`git add .`  
ili  
`git add *.html`
18. Izvršiti komandu: `touch .gitignore`
19. Upisati u fajl ono što ne želimo da commitujemo: \*.log
20. Izvršiti komandu: `git add .`
21. Izvršiti komandu: `git branch MojaGrana`
22. Izvršiti komandu: `git checkout MojaGrana`
23. Izmeniti fajl readme.txt
24. Izvršiti komandu: `touch index.css`
25. Izvršiti komandu: `git add .`
26. `git commit -m 'Nove promene'`
27. `git checkout master`
28. `git merge MojaGrana`
29. `git commit -a -m 'Neka verzija 6'`
30. `git checkout MojaGrana`
31. `git commit -a -m 'Neka verzija 7'`
32. `git merge master`  
Da li se javlja neki konflikt?

33. `git status`
34. Razlike između <<<HEAD i >>>MASTER, šta pripada izvornoj grani, a šta našoj novoj grani?
35. Izmeniti sadržaj ručnim spajanjem.
36. `git commit -a -m 'Spajanje iz mastera'`
37. `git status`
38. `git mergetool`
39. `touch novifajl.txt`
40. `git checkout master`
41. `git status`  
Da li ima fajlova koji su untracked?
42. `git checkout MojaGrana`
43. `git add .`
44. `git status`
45. `git stash`
46. `git status`
47. `git stash apply`
48. `git remote`
49. `git clone ADRESA`  
**ADRESA** treba da bude udaljeni repozitorijum.  
Potrebno je dodati udaljeni repozitorijum. Izvršiti naredbu:  
`git remote add origin ADRESA`
50. Proveriti da li je remote pod nazivom `origin` dodat: izvršiti naredbu `remote`.
51. Izvršiti komande, ukoliko želite da dobijete sadržaj iz udaljenog repozitorijuma i da ga sinhronizujete koristite sledeće komande:  
`git fetch origin`  
`git pull origin`

#### Rad sa više studenata nad istim repozitorijumom:

1. Svaki student treba koristeći gore navedene komande da podesi putanju udaljenog repozitorijuma i da taj repozitorijum doda.
2. Student1: otvoriti fajl `naslovna.html` i izmeniti sadržaj tag-a `<title>` na proizvoljnu vrednost. Dodati `<h1>Naslov</h1>` neposredno ISPOD `<body>` tag-a. Ukloniti liniju koja sadrži `<div>bezvezni div</div>`.
3. Student1: Izvršiti naredbu `git status`. Komitovati izmene sa porukom "S1: prva izmena" i proveriti status. Pogledati log. Rezultat je prikazan na slici 3.
4. Pitanje: da li se ove promene vide na centralnom repozitorijumu? Da li Student2 može ikako da ih vidi?
5. Student1: Ponovo izmeniti fajl `naslovna.html`: izmeniti sadržaj tag-a `<title>`.
6. Student1: Poslati promene na centralni repozitorijum – izvršiti naredbu `git push`.
7. Pitanje: da li je bilo moguće uraditi `push`, iako postoje izmene u radnom direktorijumu?
8. Student1: U ovom koraku ne raditi `push!!!` Uraditi `commit` promena koje su u lokalnom repozitorijumu, sa porukom "druga izmena, nije poslata". Ne raditi `push!!!`  
Nakon toga, **ponovo** izmeniti fajl `naslovna.html`, ponovo `<title>`.
9. Pitanje: u čemu je razlika između sadržaja centralnog repozitorijuma i radnog direktorijuma Studenta1?
10. Student2: Potrebno je sinhronizovati lokalni repozitorijum sa centralnim. Izvršiti naredbu `git pull`.
11. Izvršiti naredbu `git log`. Šta predstavlja prikazani sadržaj?

## Vežba 5. Rad sa udaljenim repozitorijumom – spajanje promena nad istim fajlom, bez konflikata

Preduslov za ovu vežbu je da je prethodna izvedena uspešno do kraja.

1. Student2: izmeniti sadržaj jedinog `<p>` taga u html stranici, upisati proizvoljan sadržaj.
2. Student2: Komitovati izmene sa porukom "treca izmena, student 2". Izvršiti naredbu `git log`. Šta predstavlja prikazani sadržaj?
3. Student2: Sinhronizovati sa centralnim repozitorijumom: izvršiti naredbu `git push`.
4. Pitanje: u čemu je razlika između verzija koje imate u lokalnim repozitorijumima u odnosu na udaljeni repozitorijum? Šta će se desiti kada Student1 pokuša da sinhronizuje svoj lokalni repozitorijum?
5. Student1: Sinhronizovati sa centralnim repozitorijumom: izvršiti naredbu `git pull origin master`. Problem?  
Ukloniti izmene pomoću `git checkout -- naslovna.htm` (sve se zanemari)<sup>1</sup>. Alternativno, ako biste zadržali zamene, `git add`.  
Ponoviti `pull`.
6. Pitanje: Šta je neophodno da bude zadovoljeno da bi se uspešno izvršio pull?
7. Izvršiti naredbu `git status`. Šta predstavlja prikazani sadržaj?
8. Izvršiti naredbu `git log --graph`.
9. Izvršiti naredbu `git diff naslovna.html`. Šta je prikazano?
10. Budući da je git samostalno spojio ova dva fajla i da nije bilo konflikta (jer smo namerno menjali različite linije!), potrebno je da izvršimo inspekciju merge-a . Nakon što se "sredi" merge, uraditi commit merge-ovanih fajlova, na standardni način.
11. Izvršiti naredbu `git log --graph`. Šta predstavlja prikazani sadržaj? Zašto je nastao problem?

## Vežba 6. Ponoviti prethodnu vežbu, menjajući namerno ISTU liniju koda.

12. Zbog menjanja iste linije koda, git neće umeti da inteligentno uradi merge, kao u prethodnom slučaju. Zbog toga je potrebno pokrenuti neki od merge alata, naredbom `git mergetool`, i ručno sprovesti spajanje.

---

<sup>1</sup> Moguće je skloniti privremeno izmene, dok se završi pull, a potom ih vratiti: naredba `git stash`.