Laboratorinis darbas Nr.: 2

1. *Repozitorijų kūrimas:*

Tam, kad būtų galima skelbti duomenis į serverį, reikia sukurti atitinkamus aplankus tiek kliento, tiek serverio pusėje, bei tarp jų sudaryti tinkamą ryšį. Tada galima siųsti duomenis (*git push*) bei gauti duomenis (*git pull*) arba netgi nusikopijuoti visą repozitoriją su visais kada nors atliktais ir dokumentuotais pakeitimais (*git clone*). Esant poreikiui taip pat galima išskirti atšakas (angl.: *branches*). Prieš kuriant repozitorijas reikia užtikrinti, kad git programinė įranga instaliuota. Jeigu ne, tai padaryti galima terminalo lange parašius *sudo apt-get update && sudo apt-get install git-core.*

Serverio pusėje sukurti repozitoriją nesudėtinga, o pavyzdys pateikiamas toliau.

Reikalinga sukurti atskirą vartotoją, kuris bus naudojamas kaip vietinis serveris. Tai gali būti jau esamas vartotojas, tačiau šiuo atveju sukuriamas naujas vartotojas git. Sukūrimui įvedama: *sudo adduser git*. Taip pat būtina pridėti vartotoją *git* prie „*sudoers*“ grupės, tam naudojama komanda: *sudo adduser git sudo*. Po šių veiksmų būtina paleisti virtualią mašiną iš naujo ir prisijungti kaip *git* vartotojas. Užsikrovus virtualiai mašinai suvedama komanda: *sudo* su bei suvedamas slaptažodis.

Vėliau reikia sukurti naują aplanką direktorijoje */home/git/* pavadinimu *„first“*. Tam naudojama komanda: *mkdir first.* Tam, kad šiame aplanke būtų galima inicijuoti *git* direktoriją, reikia įvesti šią komanda: *cd first && git --bare init.* Visi veiksmai su šiuo vartotoju atlikti. Galima atsijungti nuo šio vartotojo ir prisijungti kaip pagrindinis vartotojas „DELL“.

Tam, kad būtų galima patikrinti, ar galima skelbti failus, reikia kažkur sukurti aplanką ir jame sukurti bent vieną failą. Tai atliekama su šia komanda: *mkdir MyProject && cd MyProject && touch README.txt*. Atsiradusiame aplanke taip reikia sukurti *git* aplanką, su *git init* komanda. Sukurtas README failas gali būti pridėtas *git add - A* komanda. Pirmą kartą kreipiantis sistema paprastai paprašo autentifikavimo teikiant failus versijavimui, todėl prieš tai reikėtų nurodyti elektroninį paštą ir vartotojo vardą. Jiems aprašyti atitinkamai reikalingos *git config --global user.email mano@pastas.com* ir *git config --global user.name Vilius* komandos. Po šių komandų galima teikti failus versijavimui su *git commit –m “[pageidaujama žinutė]”*.

Failas pateiktas versijavimui, tačiau dar neatiduotas lokaliam serveriui. Apskritai, tarp dviejų sukurtų aplankų dar kol kas nėra ryšio, jį reikia sukurti. Tam reikalinga *ssh* (angl.: *secure shell*) komunikaciją inicijuojanti programa *OpenSSH* (įrašoma su *sudo apt-get update && sudo apt-get install openssh-server*), po šios komandos sėkmingo įvykdymo reikėtų kviesti *git remote add origin git@localhost:first*, taip nurodant, kad pastarasis aplankas bus vieta, kur serveryje bus saugomi pakeitimai. Galiausiai pakeitimus pateikti serveriui galima su *git push origin master* bei sutinkant su

visais pasirodančiais pasirinkimais. Suvedus *git status* turėtų pasirodyti užrašas, kad visi pakeitimai buvo atlikti ir duomenys yra sinchronizuoti (angl.: *nothing to commit, working directory is clean*). Tai reiškia, kad vietinis serveris yra sukurtas ir veikia tinkamai.

2. *Duomenų iš repozitorijos gavimas ir pakeitimų darymas:*

Tam, kad būtų dirbama su paskutiniais duomenimis, reikalinga pastoviai atnaujinti juos. Tai atlikti padeda *git pull* komanda.

Pirmiausia reikalinga sukurti aplanką *MyProject*. Kadangi žingsniai yra identiški vartotojo aplanko kūrimui pirmojoje dalyje, kūrimo proceso apibūdinimas praleidžiamas.

Sukūrus naują aplanką reikalinga gauti pakeitimus, kadangi kol kas aplankas tuščias. Tai reikėtų atlikti su *git pull origin master*. Sėkmės atveju aplanke turėtų atsirasti README.txt failas. Be jo dar aplanke taip pat gali būti paslėptų failų, pavyzdžiui *.gitignore*, kurio turinyje nurodoma, kurių failų pakeitimų serveriui pateikti nevalia.

3. *Konfliktų sprendimas:*

Jeigu keletas vartotojų dirba su tuo pačiu projektu ir dėl to keičia tuos pačius failus, tikėtina, kad vienas iš jų atliks pakeitimus, kuriuos galbūt jau atliko kitas asmuo, arba ištrins būtent tai, ką pridėjo kitas. Toks nutikimas vadinamas konfliktu. Dėl šios priežasties paskutinis vartotojas negalės pilnai pateikti duomenų, prieš tai turės ištaisyti susidariusį konfliktą – ištrinti, pridėti arba pataisyti atitinkamas eilutes.

Laboratorinio darbo metu konfliktas taip pat nutiko. Antrajam vartotojui įrašius naują eilutę į failą, kurį jau pateikė ir pakeitimus į serverį įrašė pirmasis vartotojas. Antrasis vartotojas prieš atlikdamas pakeitimus nepaprašė gauti naujausių pakeitimų *git pull origin master* komanda, todėl neturi teisės be leidimo ignoruoti naujausių pakeitimų.

Kadangi vartotojas nepaprašė naujausių pakeitimų, *git push origin master* išveda klaidos sąlygą. Vartotojas tada turi suvesti *git pull origin master* ir ekrane pamatęs, kad nutiko konfliktas, šiuos konfliktus išspręsti. Konfliktų sprendimas nėra sudėtingas – *git* programa pažymi, kuriose vietose yra nesutapimai, tad galima ranka peržiūrint failus šiuos konfliktus išspręsti ištrinant nereikalingas eilutes. Apie visus atliktus pakeitimus, kuriuos vartotojai perdavė serveriui, galima sužinoti *git log* komanda.