

Міністерство освіти і науки, молоді та спорту України

Національний Технічний Університет України

«Київський Політехнічний Інститут»

Навчально-науковий комплекс

«Інститут прикладного системного аналізу»

Кафедра системного проектування

**«ПРОЕКТУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ»**

Лабораторна робота № 1

“Системи контролю версій SVN, GIT."

Виконала:

студентка 4 курсу,

Група ДА-61

Полозюк К.О.

Київ – 2019

**Мета роботи**: за допомогою системи контролю версій завантажити коди програми у

репозіторій. Відтворити типовий цикл розробки програмного забезпечення з

використанням системи контролю версій.

**Задача**:

1. Вивчити основні команди роботи з репозіторіями.

2. Завантажити код програми у репозіторій.

3. Показати основний цикл роботи з програмним кодом за допомогою системи

контролю версій.

**Короткі теоретичні відомості**

Система керування версіями (від англ Version Control System, VCS або Revision Control

System.) – це спеціальне програмне забезпечення для полегшення роботи з

інформацією, яка часто змінюється. Система керування версіями дозволяє зберігати

декілька версій одного і того ж документа, при необхідності повертатися до більш

ранніх версій, визначати, хто і коли зробив ту чи іншу зміну, керувати гілками

різноманітних версій програми.

**Завдання**

1. Обрати безкоштовну систему репозиторія для системи контролю версіями,

наприклад projectlocker, або інш.

2. Встановити клієнтське безкоштовне програмне забезпечення для роботи с системою

контролю версій (GIT, SVN clients).

3. Протягом роботи над лабораторними роботами 2-6 використовувати систему

контролю версіями.

4. Описати цикл розробки програмного забезпечення з використанням системи

контрою версій.

**Зміст звіту**

1. Мета роботи.

2. Завдання роботи.

3. Оформлення результатів роботи:

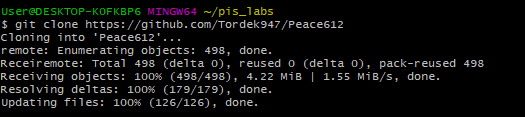
3.1. Опис команд, які використовувалися протягом виконання роботи з системою

контролю версіями.

3.2. Лістинг каталогів у репозиторію для програмних кодів для л.р. 2-6.

4. Висновки.

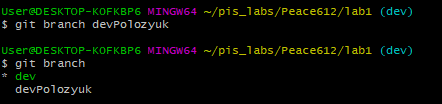
1) **git clone <url>** – створює локальну копію репозиторію



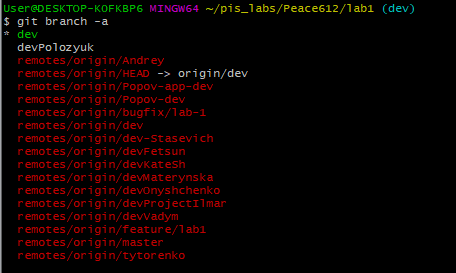
2) **git branch –** показує список гілок у локальному репозиторію



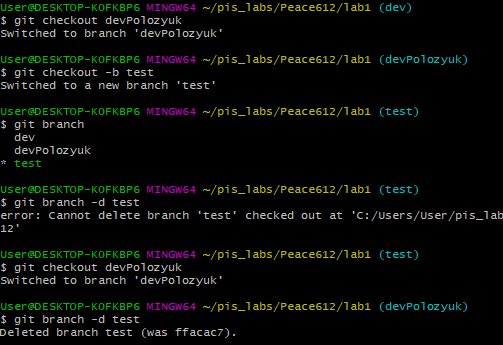
3) **git branch <branch\_name>** - створює нову гілку на локальному репозиторію



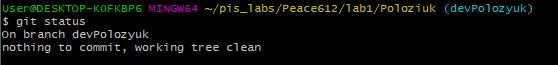
4) **git branch –a** – показує список гілок на локальному та віддаленому репозиторіях



5) **git checkout <branch\_name>** - переходить на іншу гілку, **git checkout –b <branch\_name>** - створює нову гілку та переходить на неї. **git branch –d <branch\_name>**  - видаляє гілку (окрім поточної)

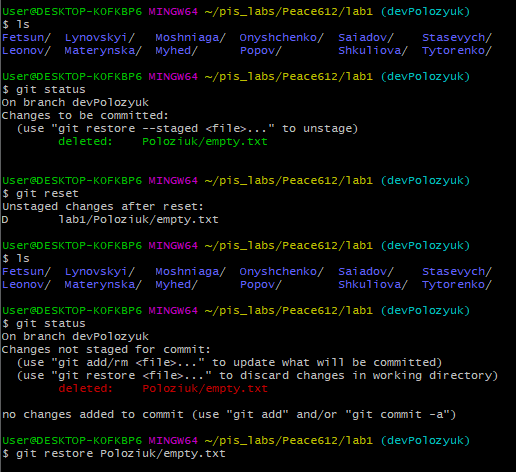


6) **git log** – показує історію репозиторію. **git status** – показує список змін у локальному репозиторію.

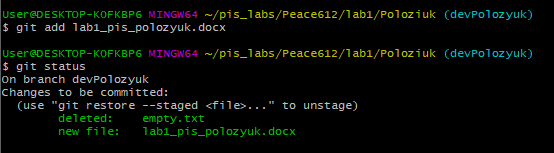


7) **git rm <file\_name>** - видаляє файл з локального репозиторію. **git restore <file\_name>** - повертає файл до початкового стану

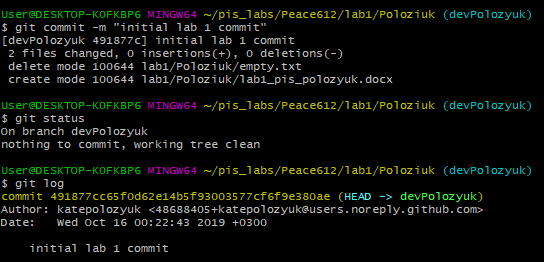




8) **git add <file\_name>** - додає файл у список змін (stash state)



9) **git commit –m “commit\_message”** – коммітить зміни з повідомленням у **commit\_message**



10) **git push** – відправляє зміни на віддалений репозиторій. **git push origin <branch\_name>** - відправляє гілку та всі зміни в ній на віддалений репозиторій.

11) **git merge** – зливає дві гілки в одну

12) **git fetch** – оновлення історії локального репозиторія данними з віддаленого

13) **git pull** – синхронізувати локальний репозиторій з віддаленим (без втрати змін на локальному, з можливими конфліктами). **git pull = git fetch + git merge**