Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційні систем та технологій

Лабораторна робота №1

із дисципліни «*Технології розробки програмного забезпечення*»

Тема: «Системи контроля версій. Розподілена система контролю версій Git»

Виконав:
Студент групи ІА-31
Клим’юк В.Л.

Перевірив:
Мягкий М.Ю.

Київ-2025

Тема: Системи контроля версій. Розподілена система контролю версій «Git».

Мета: Навчитися виконувати основні операції в роботі з децентралізованими системами контролю версій на прикладі роботи з сучасною системою Git.

Теоретичні відомості:

Система управління версій - це ПЗ, що призначене допомагати команді розробників керувати змінами у вихідному коді під час роботи. Система керування версіями дозволяє додавати зміни в файлах в репозиторій і таким чином після кожної фіксації змін мати нову ревізію файлів. Це дозволяє повертатися до попередніх версій коду для аналізу внесених змін або пошуку, які зміни призвели до появи помилки. Таким чином можна знайти хто, коли і які зміни зробив в коді, а також чому ці зміни були зроблені.

Git - це розподілена система контролю версій, створена Лінусом Торвальдсом. Така система дозволяє кожному розробнику мати свою локальну версію репозиторію, яка синхронізується з основною.

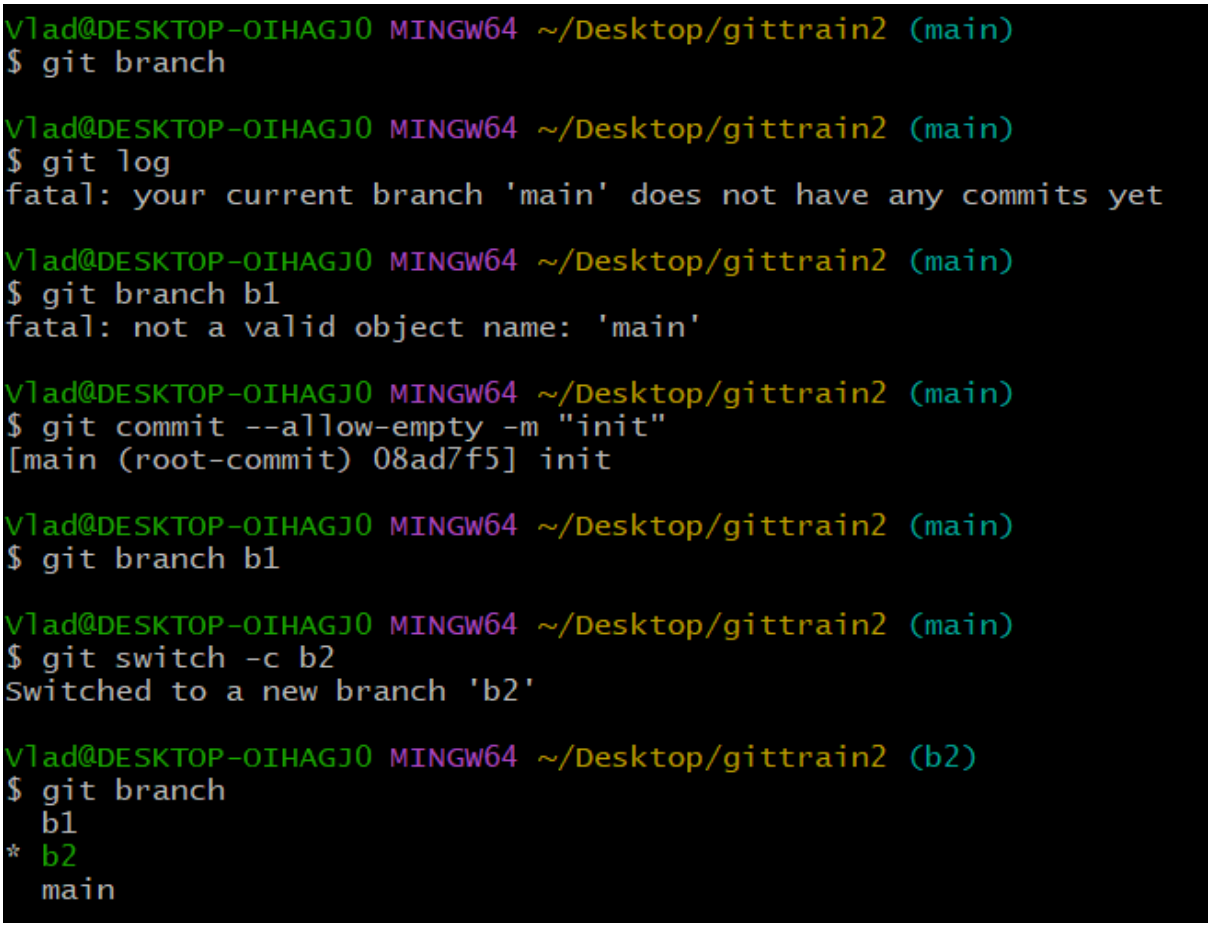
З Git можна працювати через візуальну оболонку або командний рядок. Git підтримує операції для фіксації змін, створення гілок, злиття гілок, синхронізації з віддаленим репозиторієм тощо. Найпоширенішими командами є: додавання в стадію підготовки до фіксації (git add), фіксація змін (git commit), створення гілок або перехід між ними (git branch), роботи з гілками (git merge, git rebase, git cherry-pick), відправка змін у віддалений репозиторій (git push), синхронізація локального репозиторію з віддаленим (git fetch, git pull) та інші.

Хід роботи:

1. Створення репозиторію

```
Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2
$ git init
Initialized empty Git repository in C:/Users/Vlad/Desktop/gittrain2/.git/
```

2. Створення гілок



3. Створення файлів f1 та f2 та додавання в різні гілки b1 та b2

```
Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ echo "1" > f1.txt

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ echo "2" > f2.txt

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ git status
On branch b2
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        f1.txt
        f2.txt

nothing added to commit but untracked files present (use "git add" to track)

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ git add f1.txt
warning: in the working copy of 'f1.txt', LF will be replaced by CRLF the next t
ime Git touches it

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ git commit -m "f1"
[b2 535c86f] f1
 1 file changed, 1 insertion(+)
 create mode 100644 f1.txt

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ git status
On branch b2
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        f2.txt

nothing added to commit but untracked files present (use "git add" to track)

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ git add f2.txt
warning: in the working copy of 'f2.txt', LF will be replaced by CRLF the next t
ime Git touches it

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b2)
$ git switch b1
Switched to branch 'b1'
A       f2.txt

Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b1)
$ git status
On branch b1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   f2.txt


Vlad@DESKTOP-OIHAGJO MINGW64 ~/Desktop/gittrain2 (b1)
$ git commit -m "f2"
[b1 196a75c] f2
 1 file changed, 1 insertion(+)
 create mode 100644 f2.txt
```

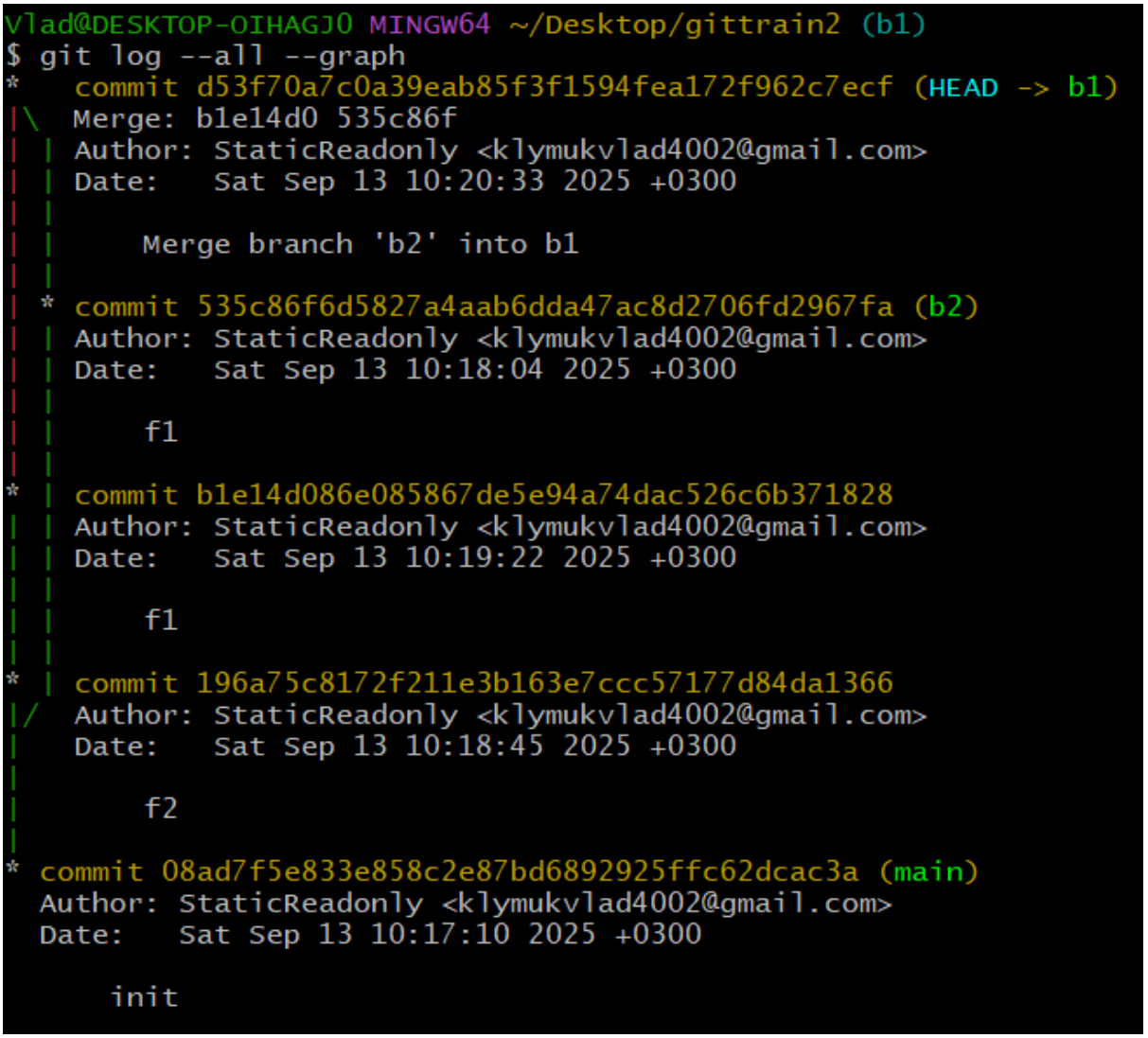
4. Створення ще одного файлу f1 на гілці b1

```
Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2 (b1)
$ echo "A" > f1.txt

Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2 (b1)
$ git add .
warning: in the working copy of 'f1.txt', LF will be replaced by CRLF the next t
ime Git touches it

Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2 (b1)
$ git commit -m "f1"
[b1 b1e14d0] f1
 1 file changed, 1 insertion(+)
 create mode 100644 f1.txt
```

5. Злиття гілок b1 b2 і вирішення конфлікту

```
Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2 (b1)
$ git merge b2
Auto-merging f1.txt
CONFLICT (add/add): Merge conflict in f1.txt
Automatic merge failed; fix conflicts and then commit the result.

Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2 (b1|MERGING)
$ git status
On branch b1
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both added:      f1.txt

no changes added to commit (use "git add" and/or "git commit -a")

Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2 (b1|MERGING)
$ git add .

Vlad@DESKTOP-OIHAGJ0 MINGW64 ~/Desktop/gittrain2 (b1|MERGING)
$ git merge --continue
[b1 d53f70a] Merge branch 'b2' into b1
```

6. Вивід історії комітів



Висновки:

При виконанні цієї лабораторної роботи ми ознайомились історією розвитку систем контролю версій і навчились працювати з Git. Було розглянуто найнеобхідніші та базові команди для фіксації змін, переходу між гілками і їхнього злиття, а також виводу історії змін.