МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ

Протокол

Лабораторна робота №2

На тему: “ Основи роботи в Git. Розгалуження”

По предмету: “Інженерія програмного забезпечення”

Виконав:

студент групи АМ-182

Борщов М. I.

Перевірила:

Шапоріна О.Л.

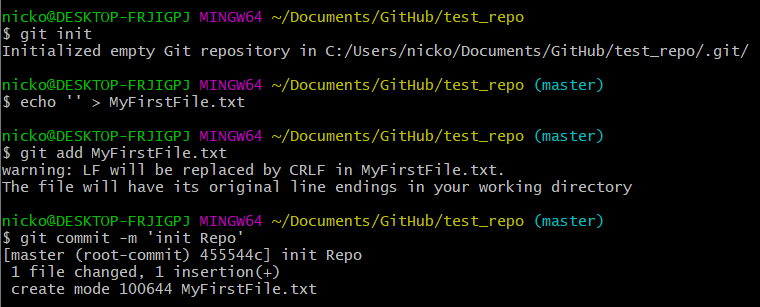
Одеса 2021

**Перелік завдань до лабораторної роботи**

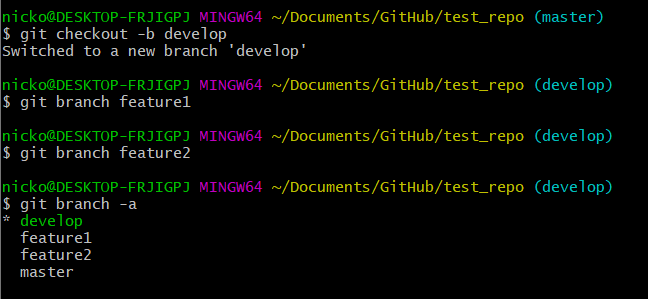
1. Изучить теоретический материал.
2. Используя GitBash и GitGUI, выполнить следующие задачи:
3. Создать пустой репозиторий. Добавить в него файл "MyFirstFile.txt". Зафиксировать изменения.
4. Создать ветку "develop" от ветки "master". Создать ветку "feature1" от ветки "develop". Создать ветку "feature2" от ветки "develop".
5. Переключится на ветку "feature1". Добавить строку "Hello" в файл "MyFirstFile.txt". Зафиксировать изменения. Переключится на ветку "feature2". Добавить строки "H", "e", "l", "l", "o" в файл "MyFirstFile.txt". Зафиксировать изменения.
6. Создать из ветки "master" ветку "hotfix" и переключится на нее. Создать файл "HotFixFile" со строкой "HotFix. Зафиксировать изменения. Добавить в файл "HotFixFile.txt" новую строку "HotFix2". Зафиксировать изменения.
7. Сделать перемещение (rebase) ветки "master" на ветку "hotfix".
8. Сделать перемещение (rebase) ветки "develop" на ветку "master". Сделать перемещение (rebase) ветки "feature1" на ветку "develop". Сделать перемещение (rebase) ветки "feature2" на ветку "develop".
9. Переключиться на ветку "feature1". Добавить файл Feature1.txt в репозиторий. Зафиксировать изменения. Переключиться на ветку "feature2". Добавить файл Feature2.txt в репозиторий. Зафиксировать изменения.
10. Разработка функции "feature1" окончена. Необходимо сделать перемещение (rebase) всех изменений из ветки "develop" на ветку "feature1".
11. Сделать перемещение (rebase) ветки "feature2" на ветку "develop". Решить возникший конфликт объединения.
12. Удалить ветку "hotfix".
13. Создать ветку "release01" от ветки "develop". В ней будет готовится релиз. Переключится на ветку "release01". Создать файл "ReleaseFix.txt" и поместить его в репозиторий. Зафиксировать изменения.
14. Предположим, что из ветки "release01" был произведен релиз. Сделать перемещение (rebase) ветки "master" на ветку "release01". Сделать перемещение (rebase) ветки "develop" на ветку "master". Сделать перемещение (rebase) ветки "feature2" на ветку "develop". Удалить ветку "feature1".
15. Предположим, что последние изменение в ветке "feature2" не нужны. Необходимо откатить их назад сделав revert commit. Создать ветку "feature3" из ветки "develop". Добавить в репозиторий 3 файла: "Feature3.txt", "Feature3.1.txt" и "Feature3.2.txt". Зафиксировать изменения.
16. Создать ветку "feature4" от ветки "develop" и переключится на нее. Добавить в репозиторий папку "f4" с четырьмя файлами внутри: "Feature1.txt", "HotFix.txt", "GettingStarted.txt", "ReleaseFix.txt". Зафиксировать изменения.
17. Сделать перемещение (rebase) ветки "develop" на ветку "feature3". Удалить ветку "feature3".
18. Создать ветку "hotfix2" от ветки "master" и переключится на нее. Добавить в репозиторий папку "hotfix" с файлом "HotFix2.txt" внутри. Зафиксировать изменения. Переместить (rebase) ветку "develop" на ветку "feature2".
19. Сделать перемещение (rebase) ветки "master" на ветку "hotfix2". Сделать перемещение (rebase) ветки "develop" на ветку "master". Сделать перемещение (rebase) ветки "feature2" на ветку "master". Сделать перемещение (rebase) ветки "feature4" на ветку "develop".
20. Сделать перемещение (rebase) ветки "develop" на ветку "feature4". Удалить ветку "feature2".
21. Удалить ветку "feature4".Сделать перемещение (rebase) ветки "master" на ветку "release-candidate2". Сделать перемещение (rebase) ветки "develop" на ветку "master". Переименовать ветку "release-candidate2" в "release2".

**Хід роботи**

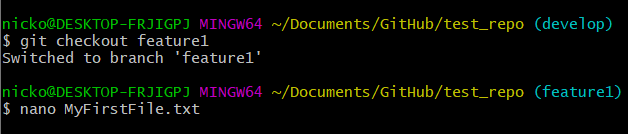
1. Використовуючи GitBash і GitGUI, виконати наступні завдання:
2. Створити порожній репозиторій. Додати в нього файл "MyFirstFile.txt". Зафіксувати зміни.

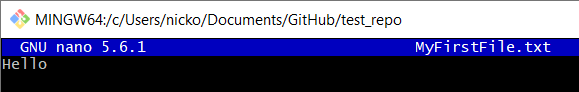


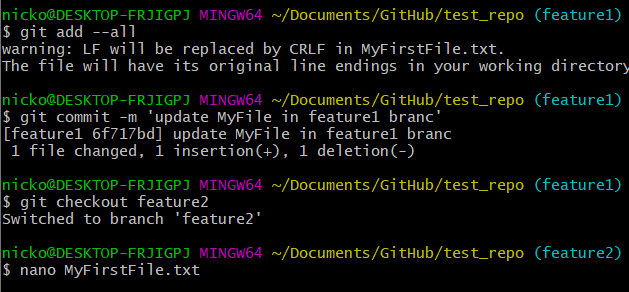
1. Створити гілку "develop" від гілки "master". Створити гілку "feature1" від гілки "develop". Створити гілку "feature2" від гілки "develop".

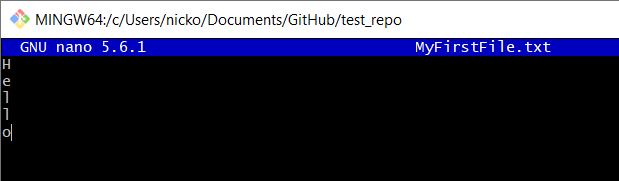


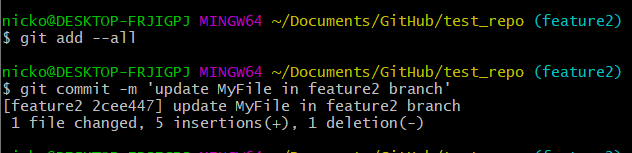
1. Переключитись на гілку "feature1". Додати рядок "Hello" в файл "MyFirstFile.txt". Зафіксувати зміни. Переключиться на гілку "feature2". Додати рядки "H", "e", "l", "l", "o" в файл "MyFirstFile.txt". Зафіксувати зміни.



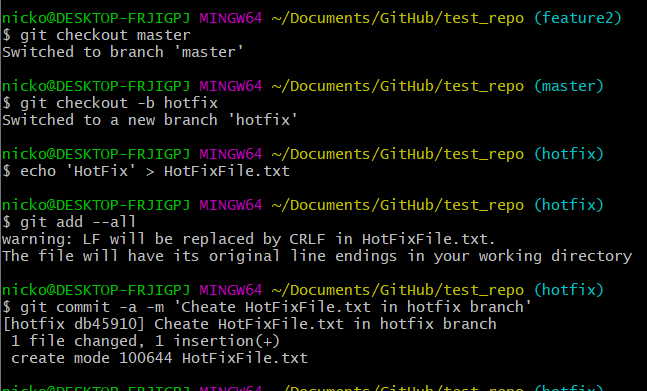




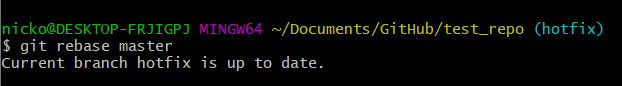




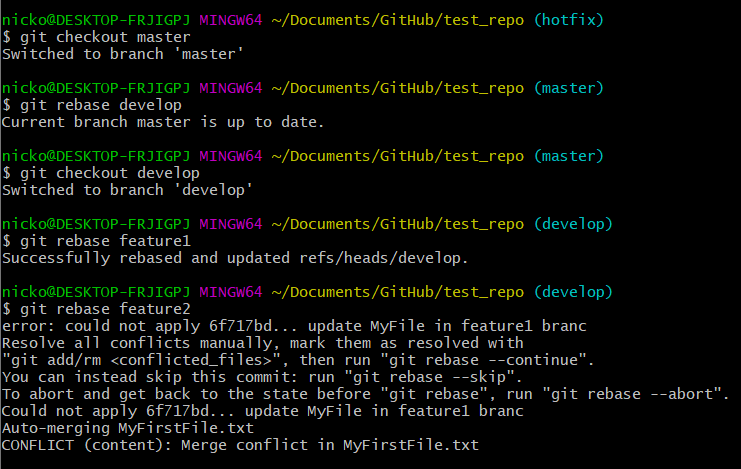
1. Створити з гілки "master" гілку "hotfix" і переключиться на неї. Створити файл "HotFixFile" з рядком "HotFix. Зафіксувати зміни. Додати в файл" HotFixFile.txt "новий рядок" HotFix2 ". Зафіксувати зміни.



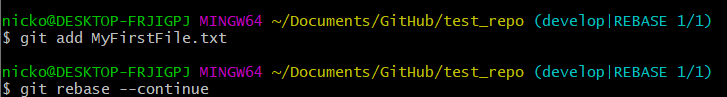
1. Зробити переміщення (rebase) гілки "master" на гілку "hotfix".



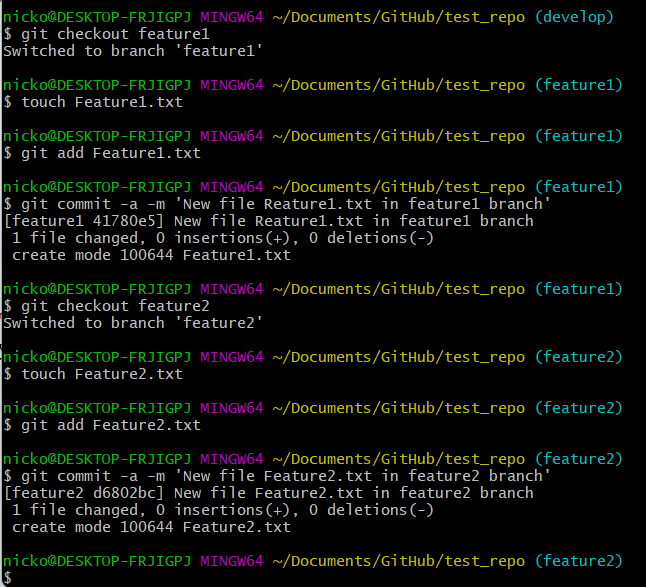
1. Зробити переміщення (rebase) гілки "develop" на гілку "master". Зробити переміщення (rebase) гілки "feature1" на гілку "develop". Зробити переміщення (rebase) гілки "feature2" на гілку "develop".



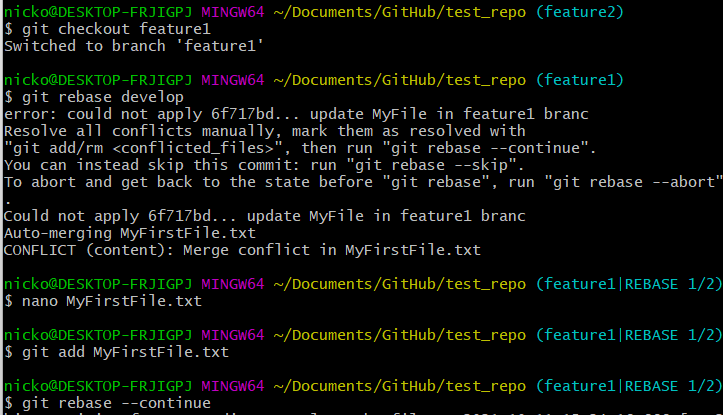
Під час переміщеня гілок виник конфлікт файлів. Це сталося бо зміст файлу MyFirstFile.txt різний у гілках feature1 та feature2.



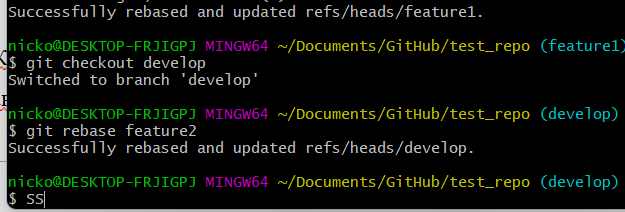
1. Переключитися на гілку "feature1". Додати файл Feature1.txt в репозиторій. Зафіксувати зміни. Переключитися на гілку "feature2". Додати файл Feature2.txt в репозиторій. Зафіксувати зміни.



1. Розробка функції "feature1" закінчена. Необхідно зробити переміщення (rebase) всіх змін з гілки "develop" на гілку "feature1".

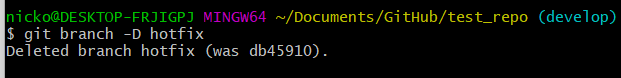


1. Зробити переміщення (rebase) гілки "feature2" на гілку "develop". Вирішити конфлікт, що виник у результаті об'єднання.

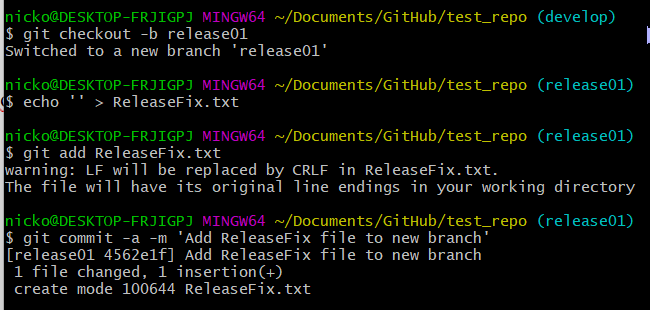


Конфлікт переміщення був вирішений в попередніх завданнях

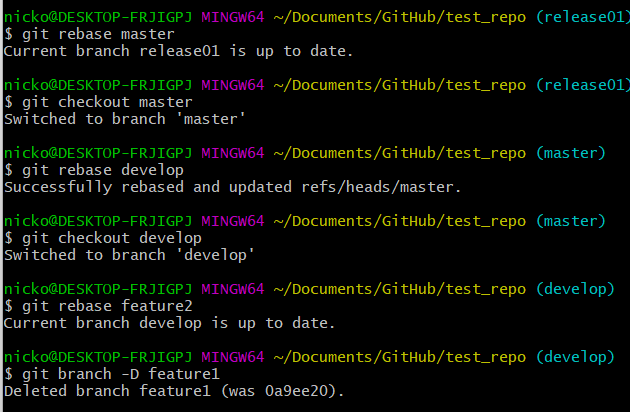
1. Видалити гілку "hotfix".



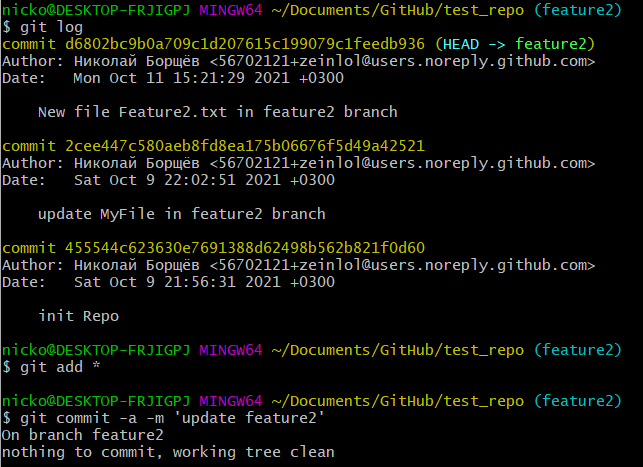
1. Створити гілку "release01" від гілки "develop". У ній буде готується реліз. Переключиться на гілку "release01". Створити файл "ReleaseFix.txt" і помістити його в репозиторій. Зафіксувати зміни.



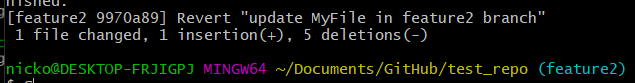
1. Припустимо, що з гілки "release01" був проведений реліз. Зробити переміщення (rebase) гілки "master" на гілку "release01". Зробити переміщення (rebase) гілки "develop" на гілку "master". Зробити переміщення (rebase) гілки "feature2" на гілку "develop". Видалити гілку "feature1".

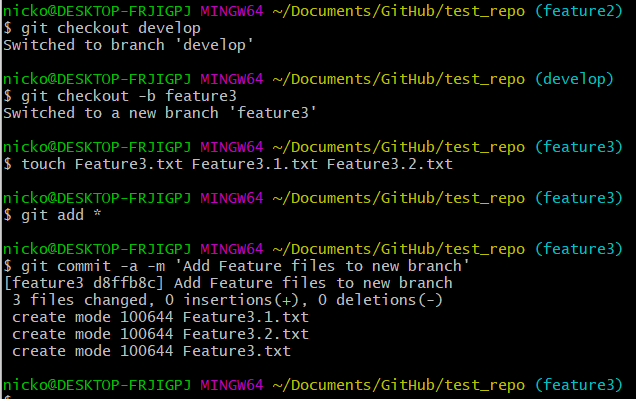


1. Припустимо, що останні зміни в гілці "feature2" не потрібні. Необхідно відкинути їх назад зробивши revert commit. Створити гілку "feature3" з гілки "develop". Додати в репозиторій 3 файлу: "Feature3.txt", "Feature3.1.txt" і "Feature3.2.txt". Зафіксувати зміни.

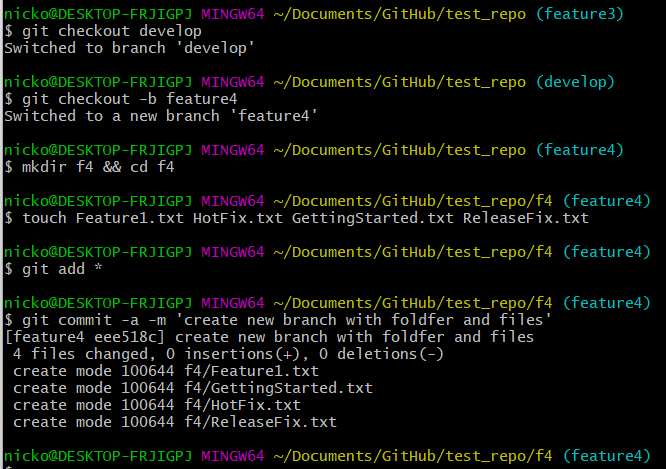




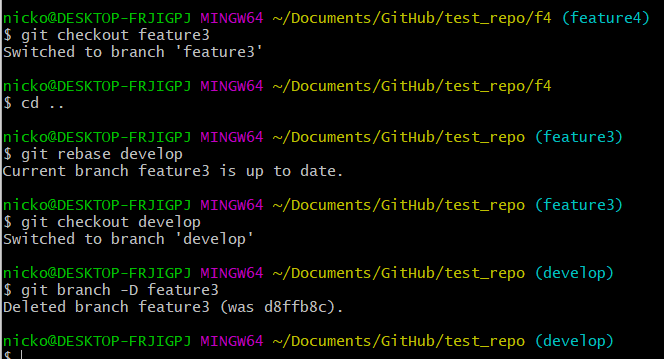




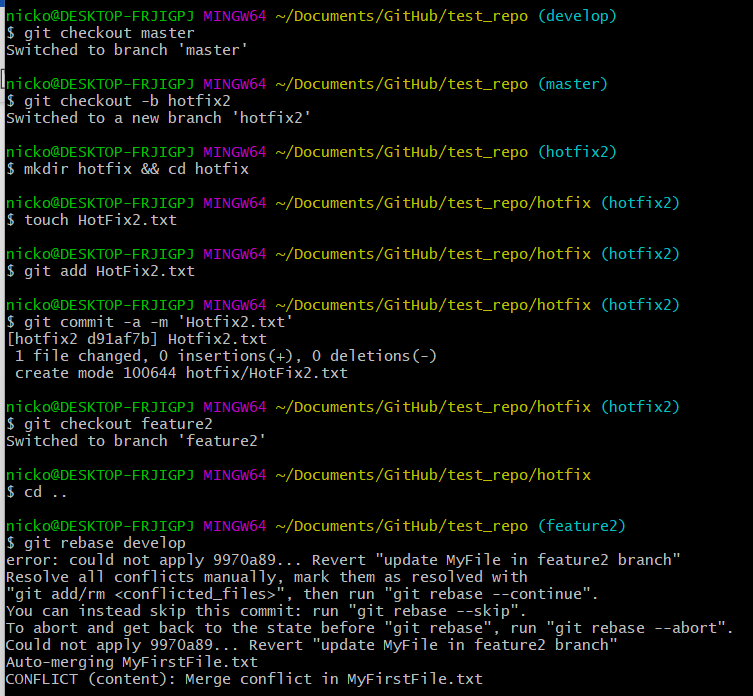
1. Створити гілку "feature4" від гілки "develop" і переключиться на неї. Додати в репозиторій папку "f4" з чотирма файлами всередині: "Feature1.txt", "HotFix.txt", "GettingStarted.txt", "ReleaseFix.txt". Зафіксувати зміни.

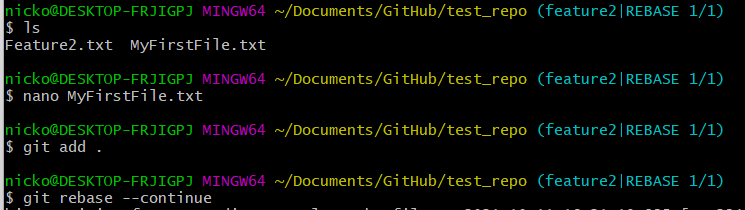


1. Зробити переміщення (rebase) гілки "develop" на гілку "feature3". Видалити гілку "feature3".



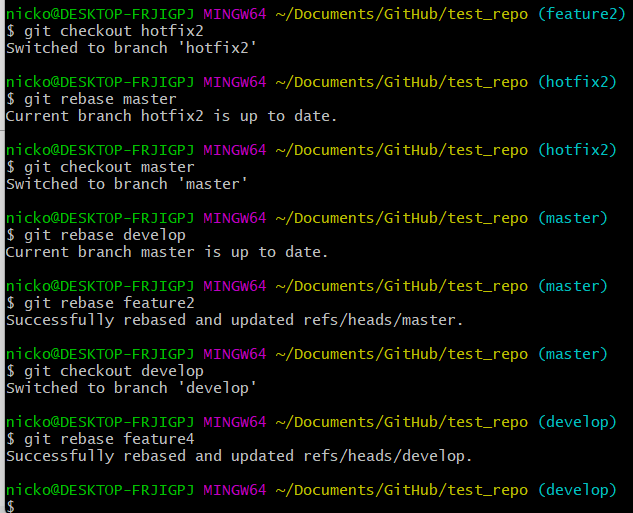
1. Створити гілку "hotfix2" від гілки "master" і переключиться на неї. Додати в репозиторій папку "hotfix" з файлом "HotFix2.txt" всередині. Зафіксувати зміни. Перемістити (rebase) гілку "develop" на гілку "feature2".



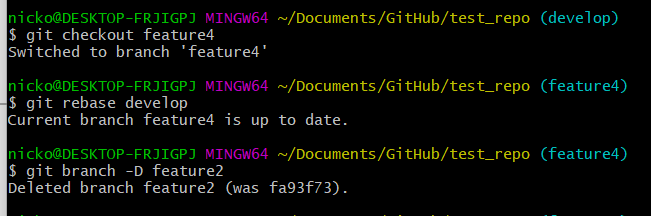


Вирішення merge кофлікту

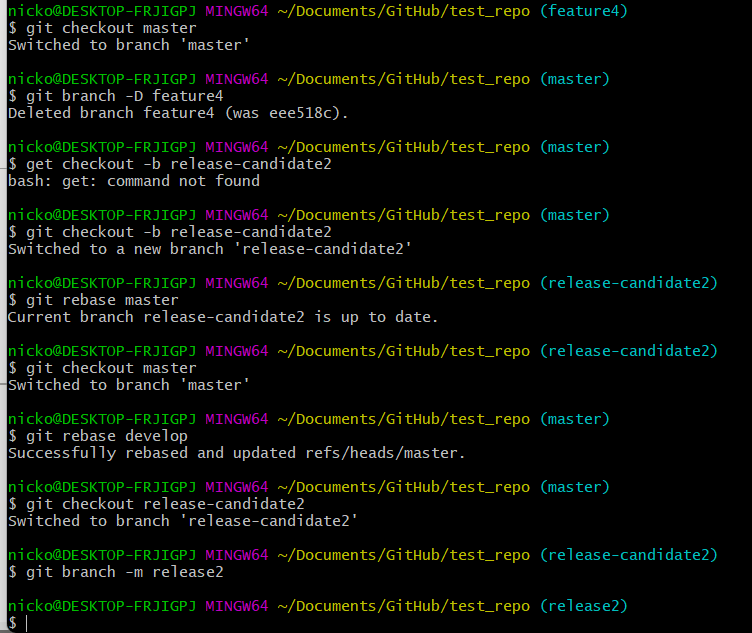
1. Зробити переміщення (rebase) гілки "master" на гілку "hotfix2". Зробити переміщення (rebase) гілки "develop" на гілку "master". Зробити переміщення (rebase) гілки "feature2" на гілку "master". Зробити переміщення (rebase) гілки "feature4" на гілку "develop".



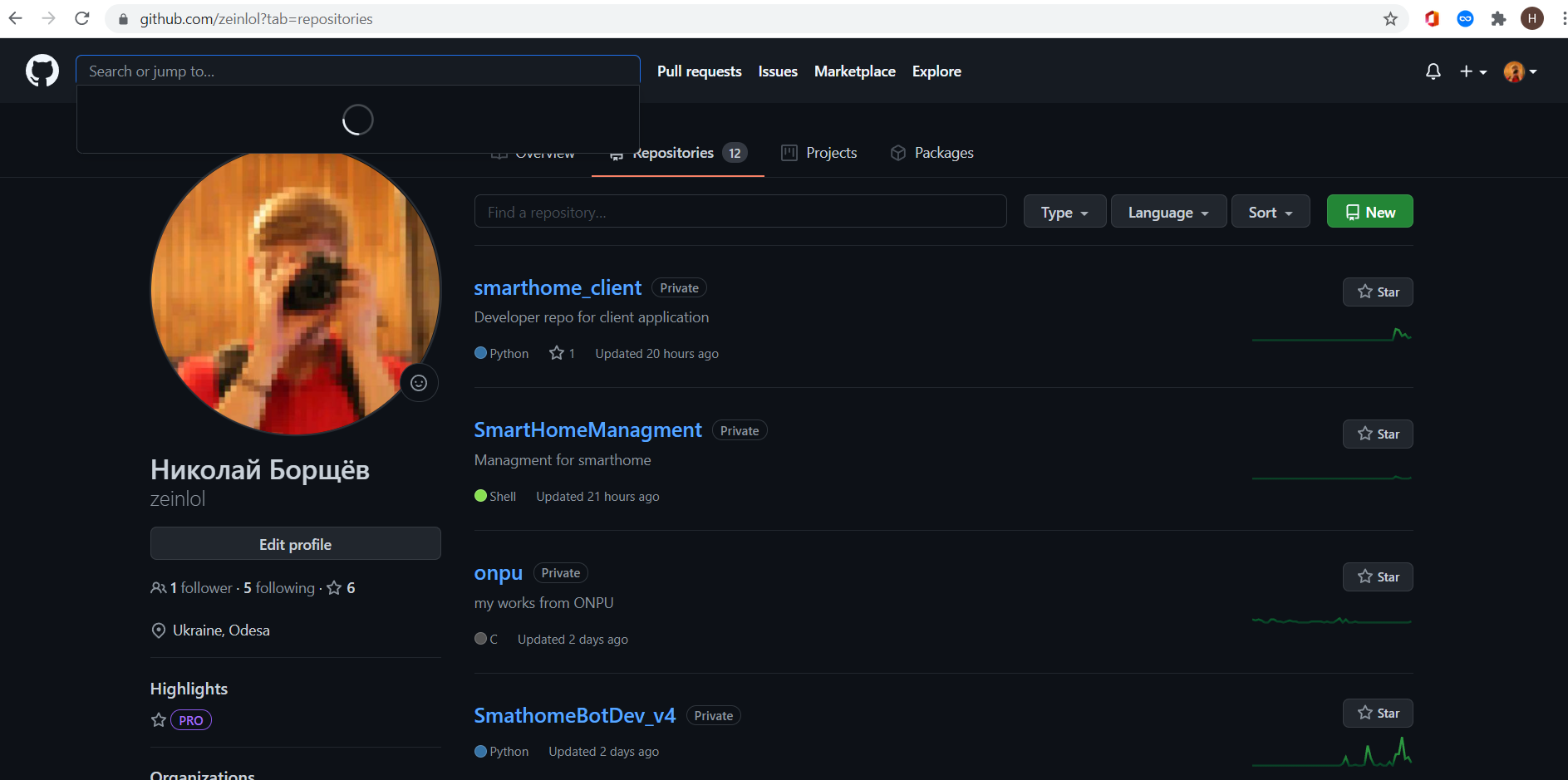
1. Зробити переміщення (rebase) гілки "develop" на гілку "feature4". Видалити гілку "feature2".

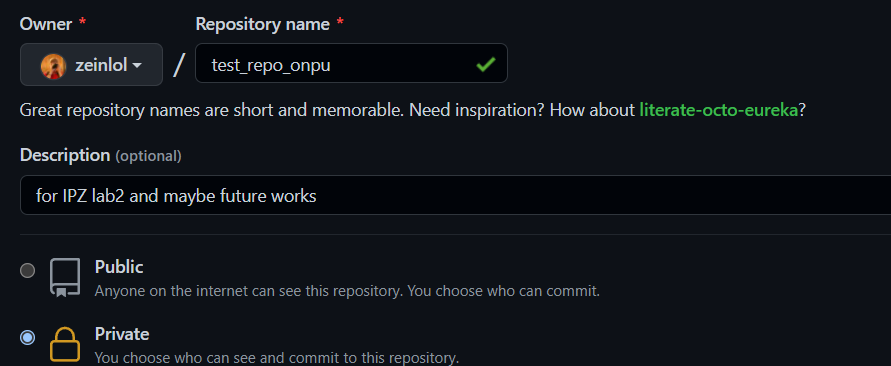


1. Видалити гілку "feature4". Зробити переміщення (rebase) гілки "master" на гілку "release-candidate2". Зробити переміщення (rebase) гілки "develop" на гілку "master". Перейменувати гілку "release-candidate2" в "release2".

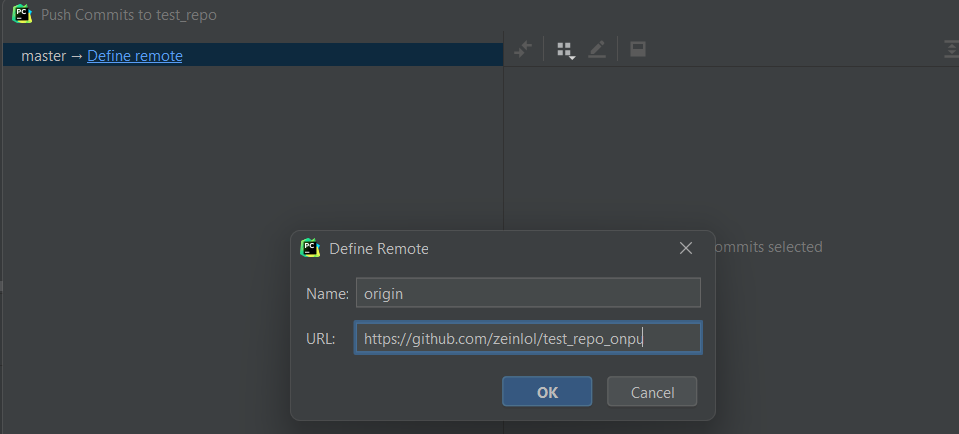


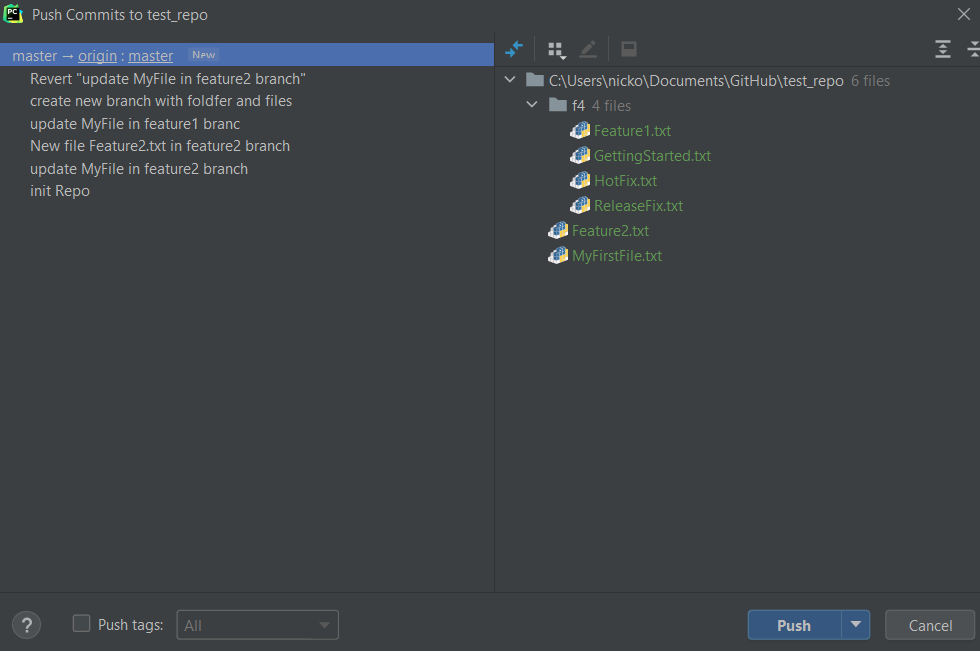
Далі на github.com зроблено репозиторій, на який було завантажено усі файли та історію змін.



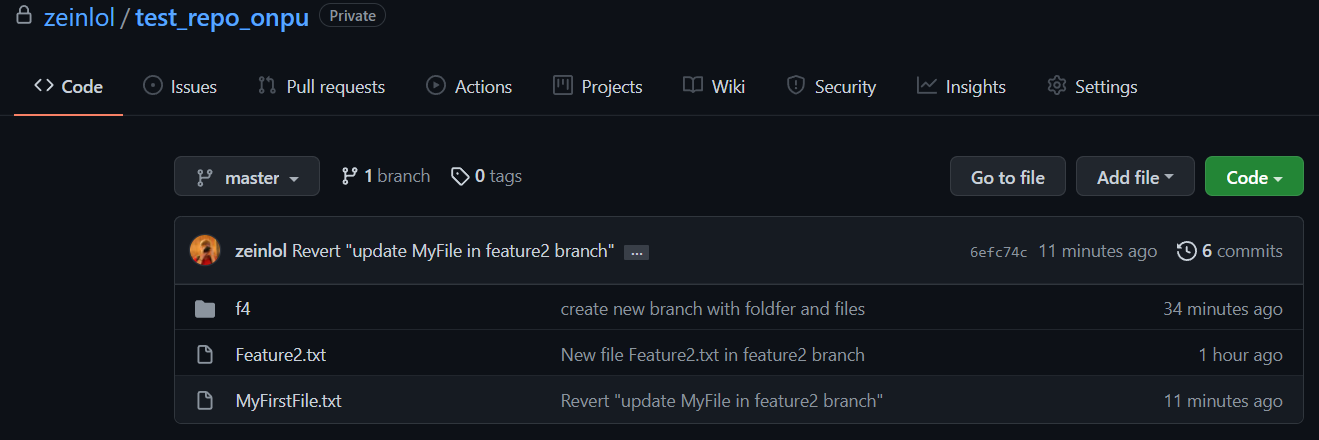


Для завантаження та історії змін було використано IDE PyCharm

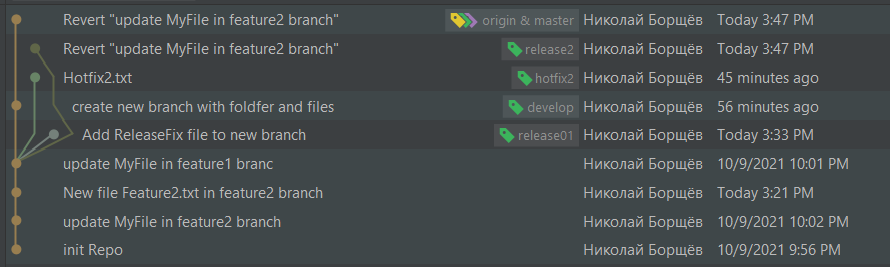




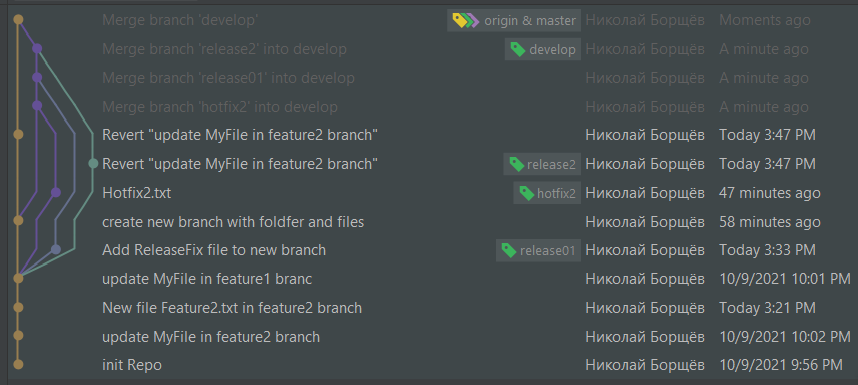
Результат



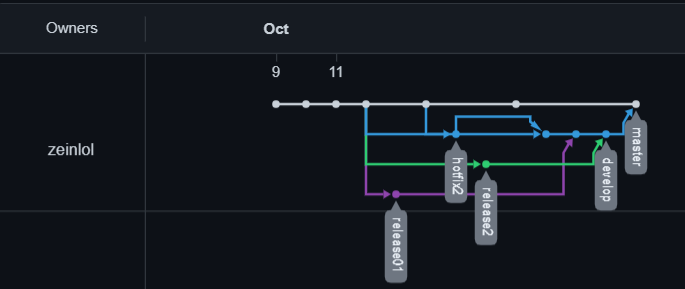
Історія змін



Розгалуження гілок не є ідеальним, бо не були закоммітчені рішення megre-кофліктів та частині гілок з конфліктами була видалена. У IDE PyCharm було виконано злиття усіх “робочих” гілок з гілкою develop, а потім вона була приєднана до гілки master.



Історія гілок на сайті. Гілки можуть бути локальними або завандаженими у мережу, для того щоб їх могли використовувати інші розробники



**Висновок:**

В результаті виконання даної лабораторної роботи я ознайомився з основними командами для роботи в Git Bash, навчився механізмам розгалужень, злиття, переміщення гілок, вирішенню merge-конфліктів та створенню файлів. Створив власний тестовий репозиторій де виконував усі завдання, а потім завантажив його та історію змін на сайт github.com.