Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №3
дисциплины «Основы программной инженерии»

Выполнил:
Яблоновский Дмитрий Николаевич
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и сопровождение программного обеспечения», очная форма обучения

_____
(подпись)

Руководитель практики:

Богданов С.С., ассистент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: Основы ветвления Git.

Цель работы: исследование базовых возможностей по работе с локальными и удаленными ветками Git.

Порядок выполнения работы

1. Создал репозиторий GitHub.

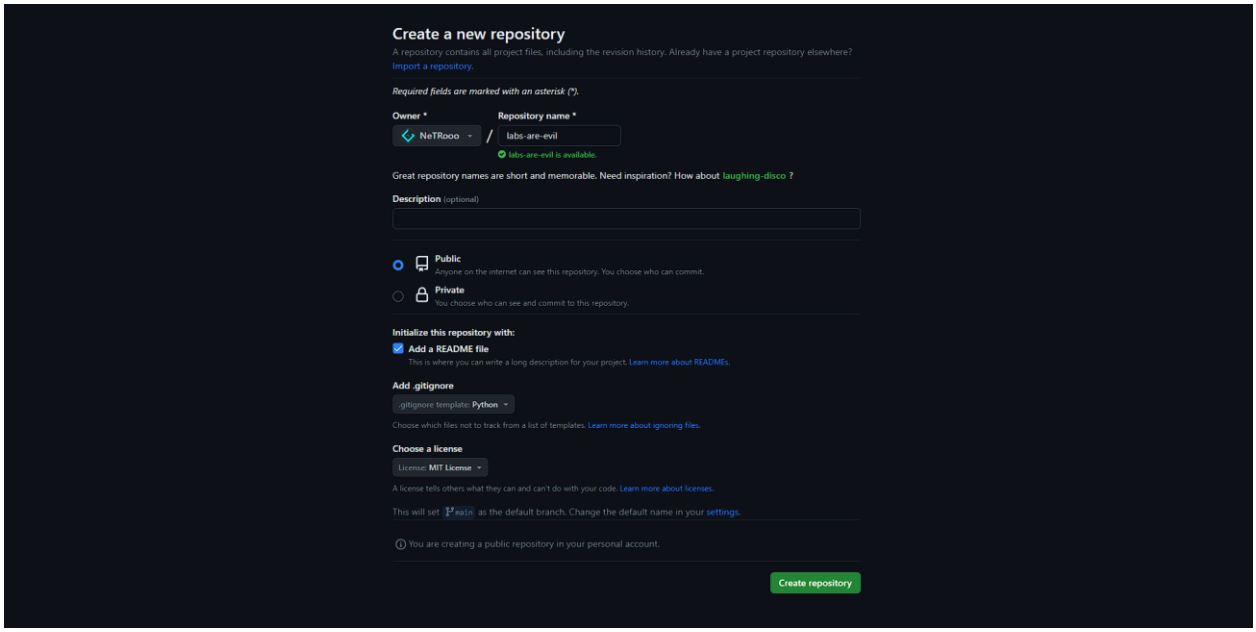

Рисунок 1- Создание репозитория

2. Проработал примеры из лабораторной работы.

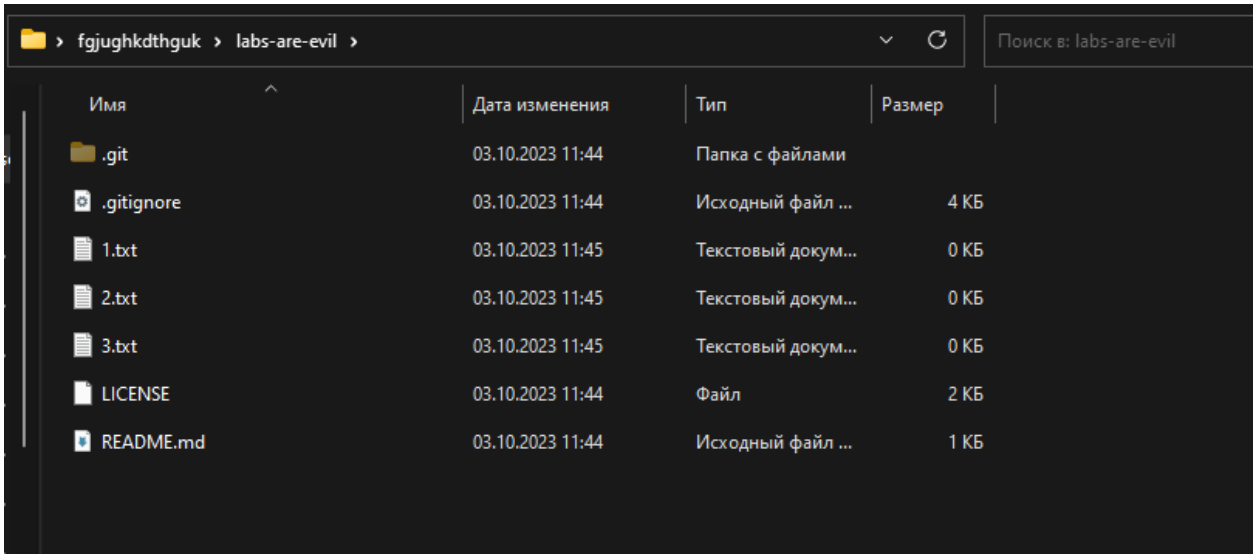

Рисунок 2 – Создал файлы «1.txt», «2.txt», «3.txt»

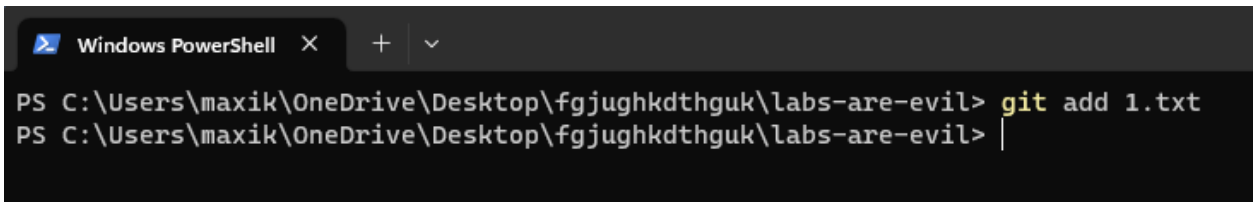

Рисунок 3 – Проиндексировал файл «1.txt»

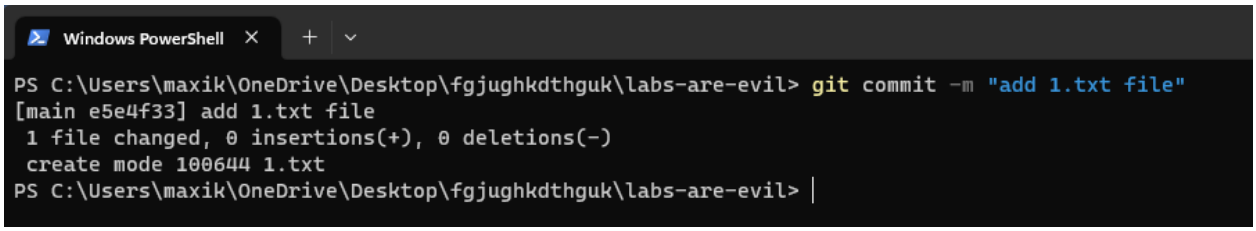

Рисунок 4 – Сделал первый коммит

Рисунок 5 – Проиндексировал файлы «2.txt», «3.txt» и создал коммит

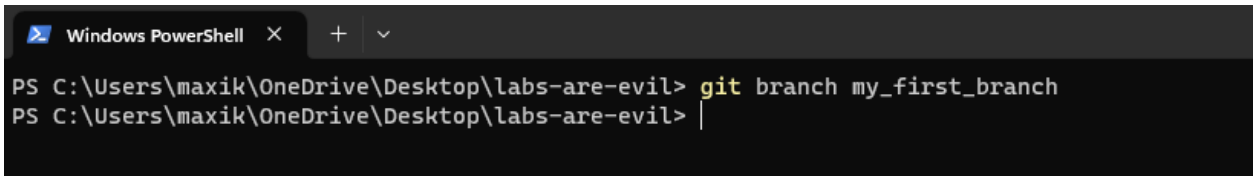

Рисунок 6 – Создал новую ветку «my_first_branch»

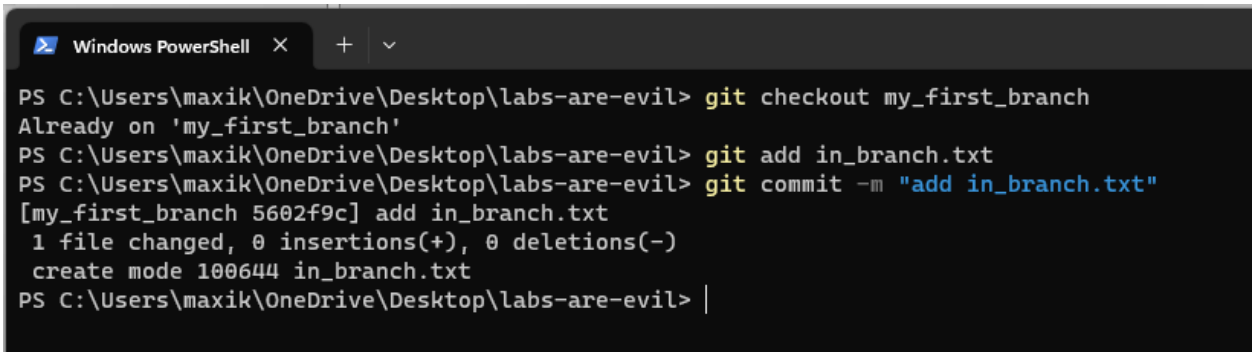

Рисунок 7 – Перешёл в ветку «my_first_branch», создал новый файл и сделал коммит изменений

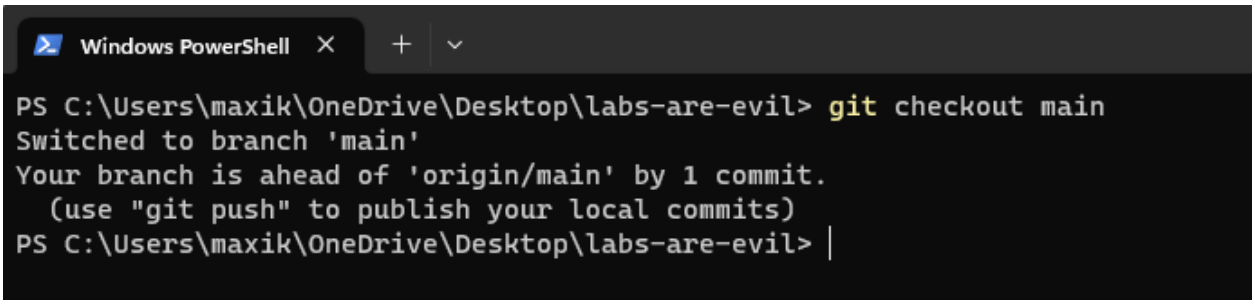

Рисунок 8 – Вернулся в ветку «main»

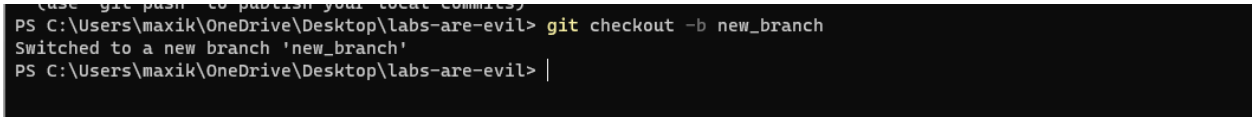

Рисунок 9 – Создал и перешёл в ветку «new_branch»

3. Выполнил клонирование репозитория на рабочий компьютер.

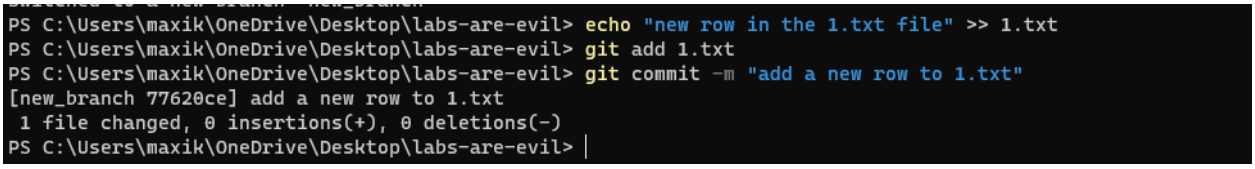

Рисунок 10 – Сделал изменения в файле «1.txt», сделал коммит

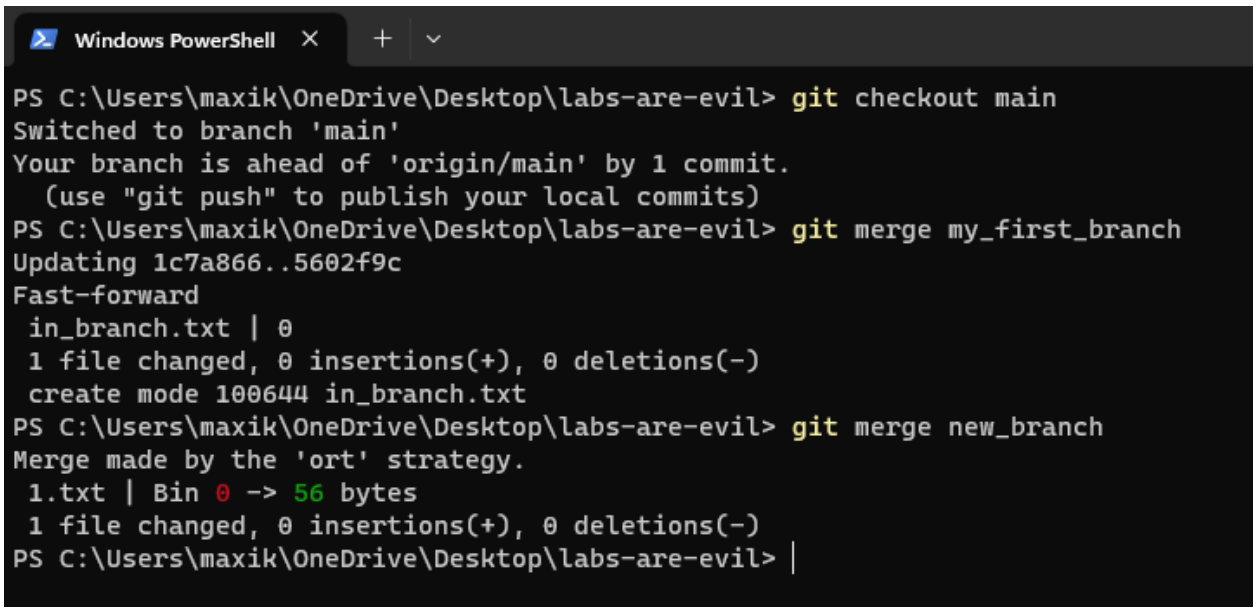

Рисунок 11 – Перешёл в ветку «main», выполнил слияние всех веток

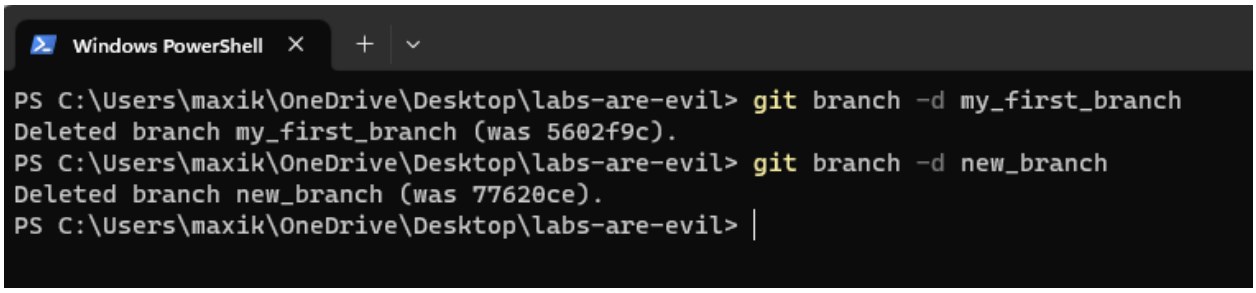

Рисунок 12 – Удалил ветки

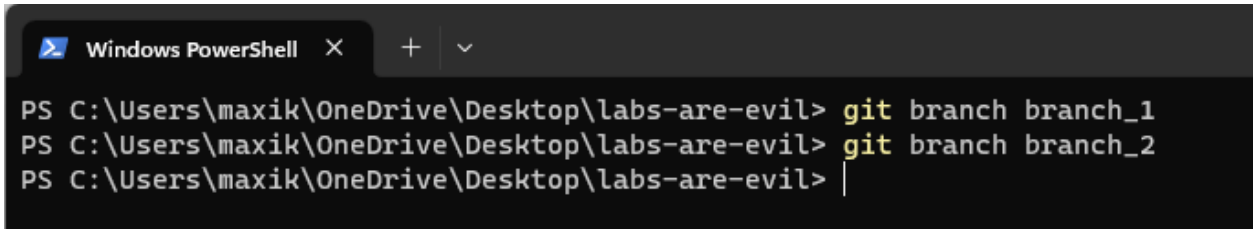

Рисунок 13 – Создал новые ветки

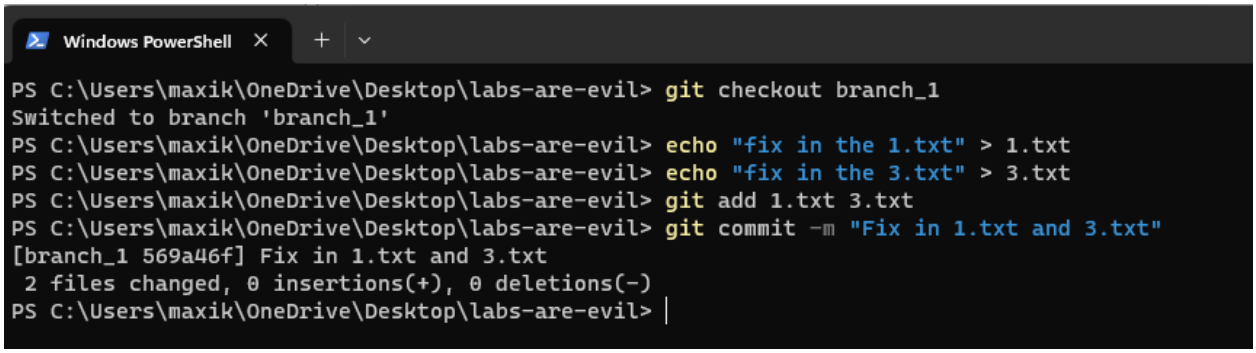

Рисунок 14 – Перешёл в ветку, изменил файлы и сделал коммит

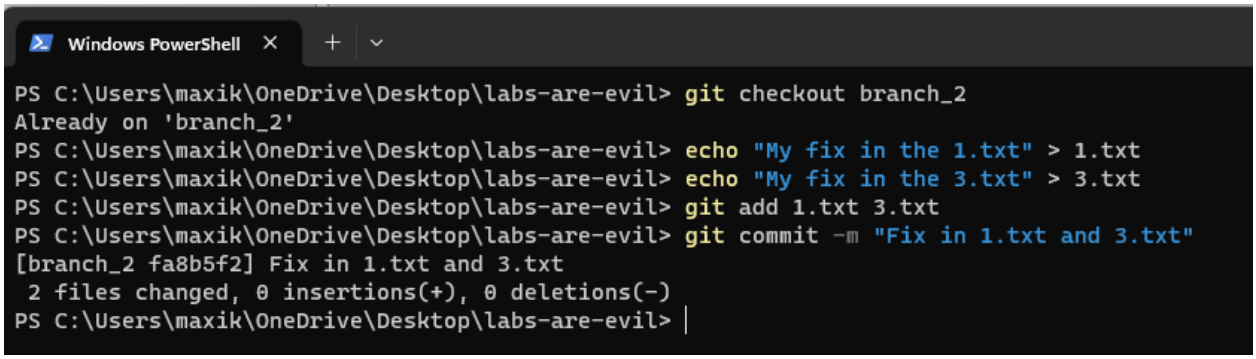

Рисунок 15 – Перешёл в другую ветку, изменил файлы и сделал коммит

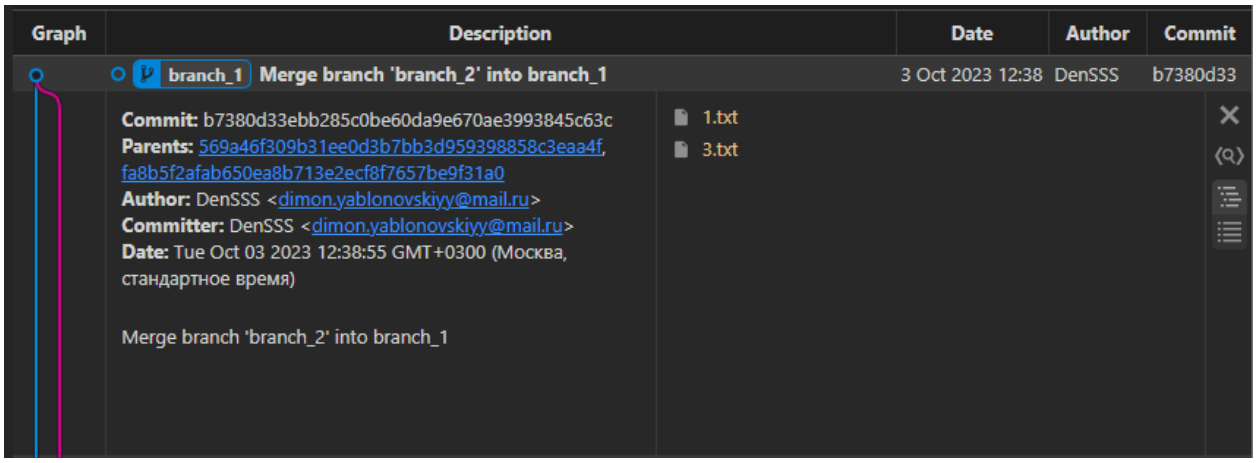

Рисунок 16 – Решил конфликт «1.txt» в ручном режиме

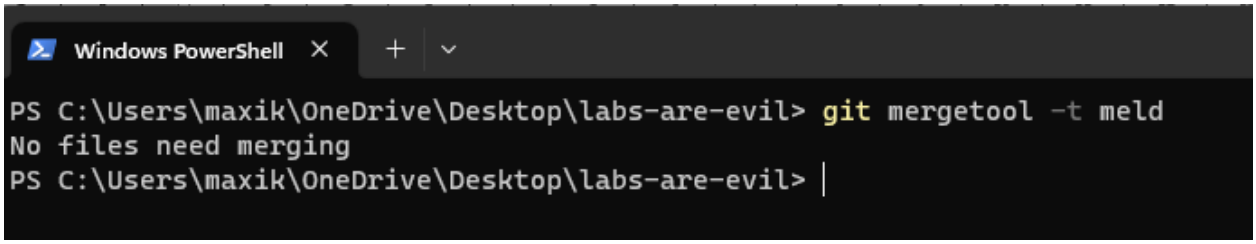

Рисунок 17 – Решил конфликт «3.txt» используя mergetool

```
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil> git push origin branch_1
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 12 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (22/22), 1.94 KiB | 662.00 KiB/s, done.
Total 22 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), done.
remote:
remote: Create a pull request for 'branch_1' on GitHub by visiting:
remote:      https://github.com/NeTRooo/labs-are-evil/pull/new/branch_1
remote:
To https://github.com/NeTRooo/labs-are-evil.git
 * [new branch]      branch_1 -> branch_1
PS C:\Users\maxik\OneDrive\Desktop\labs-are-evil>
```

Рисунок 18 – Отправил ветку на удалённый репозиторий

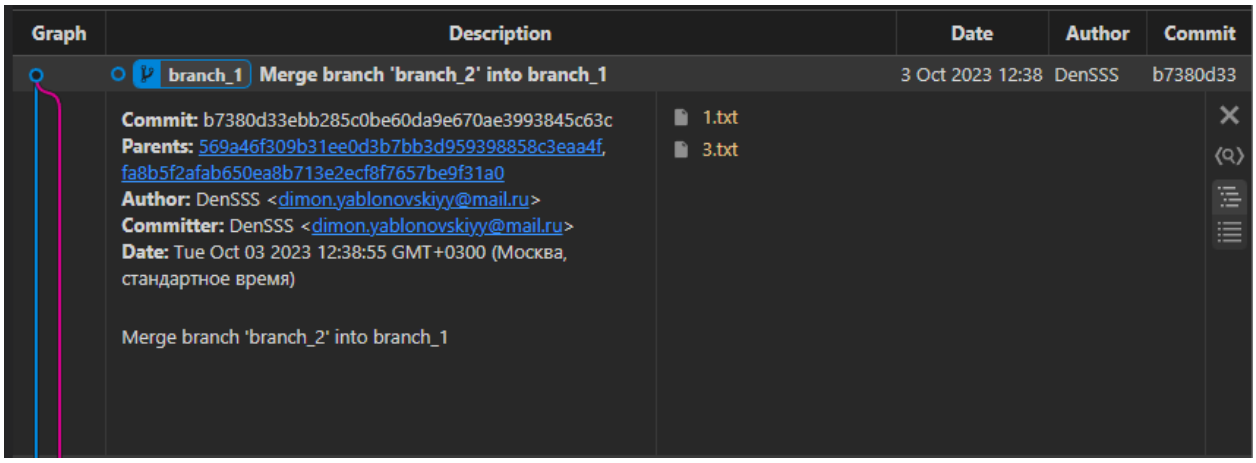

Рисунок 19 – Решил конфликт «1.txt» в ручном режиме

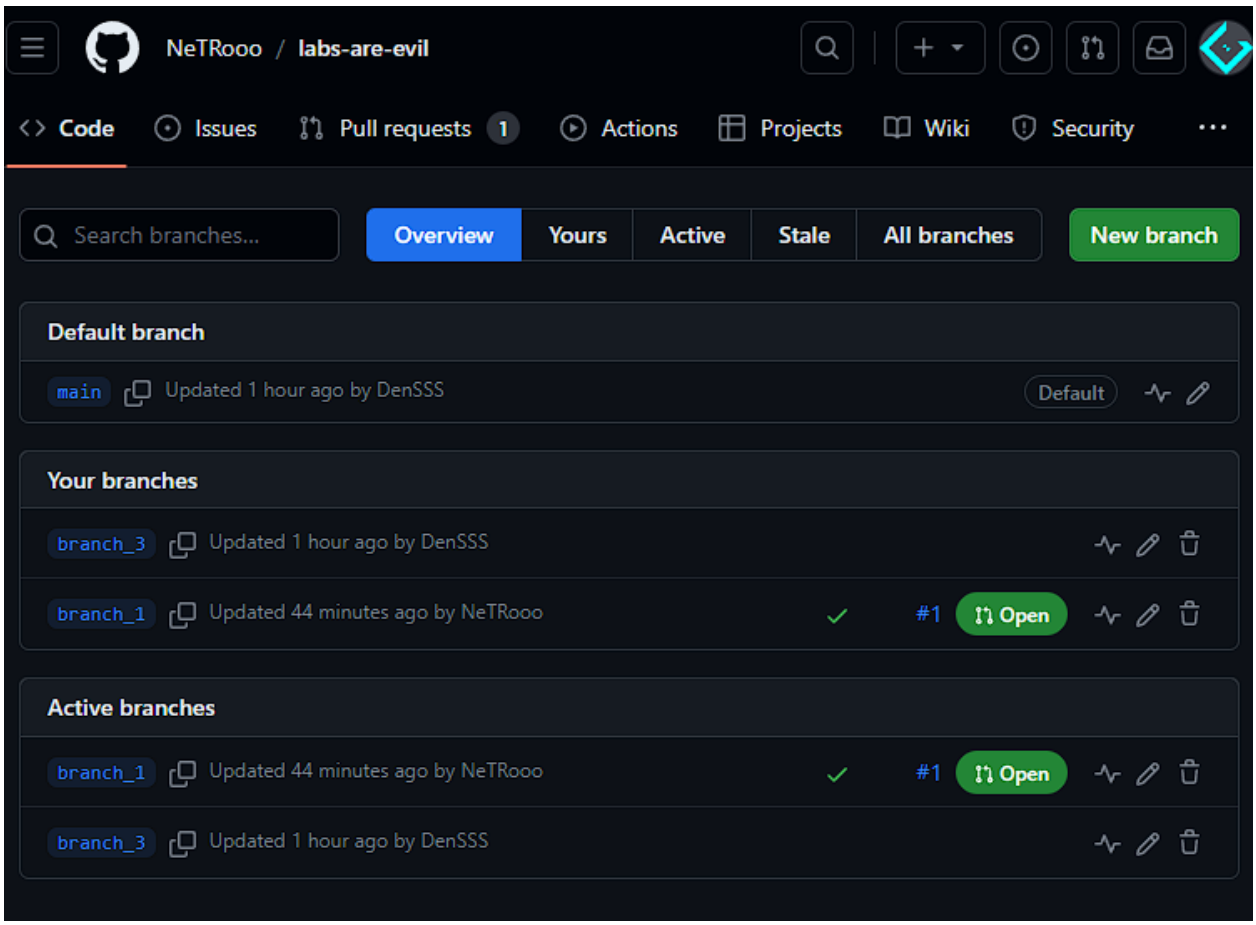

Рисунок 20 – Создал ветку branch_3 на сайте github

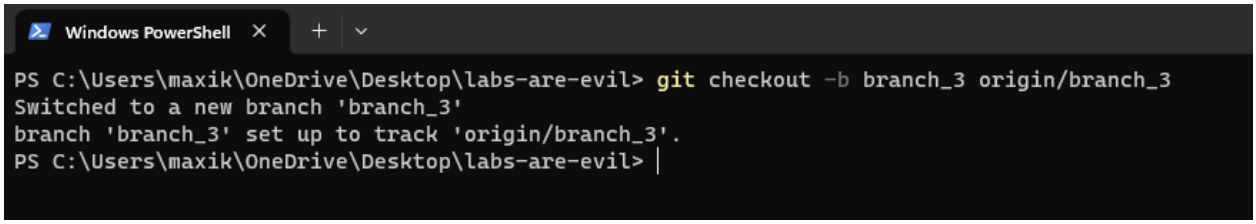

Рисунок 21 – Создал в локальном репозитории ветку отслеживания удалённой ветки branch_3

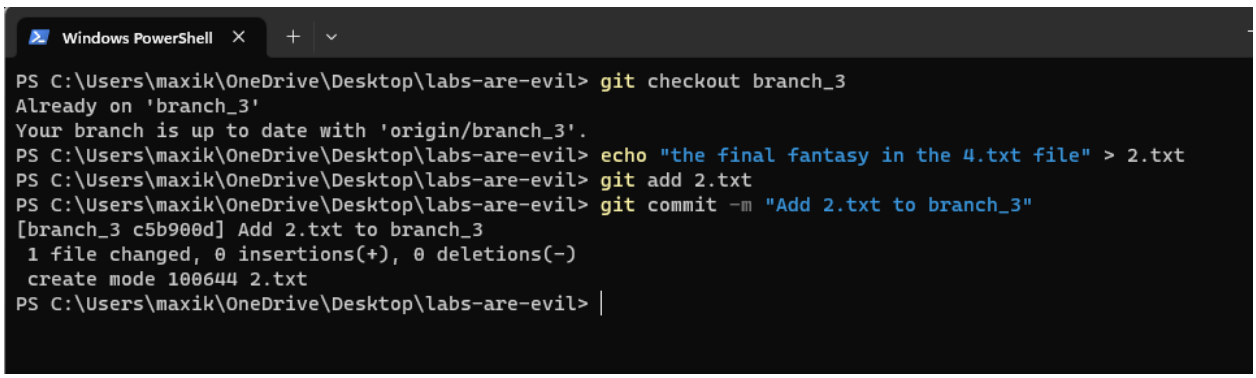

Рисунок 22 – Сменил ветку, изменил файл и сделал коммит

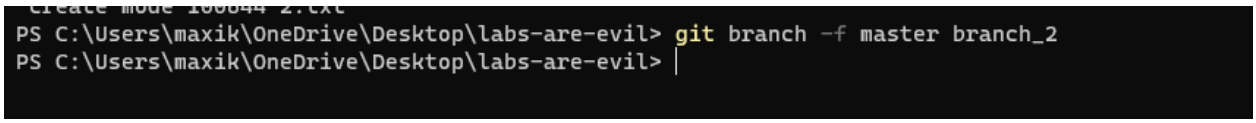

Рисунок 23 – Выполнил перемещение ветки main на ветку branch_2

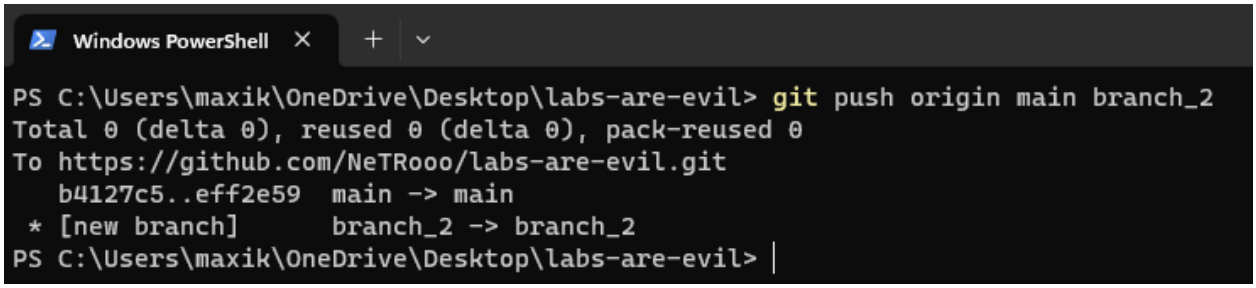

Рисунок 24 – Отправил изменения веток

Ответы на контрольные вопросы

1. Как выполнить историю коммитов в Git? Какие существуют дополнительные опции для просмотра истории коммитов?

С помощью команды git log. Существуют множество дополнительных опций вот некоторые из них:

— -p или -patch – показывает разницу, внесенную в каждый коммит;

— --stat – позволяет увидеть сокращенную статистику;

— --pretty – эта опция меняет формат вывода;

— --since и –until – опции для ограничения вывода по времени

2. Как ограничить вывод при просмотре истории коммитов?

С помощью команды git log и его аргументов, например –since.

3. Как внести изменения в уже сделанный коммит?

С помощью команды git commit –amend.

4. Как отменить индексацию файла в Git?

С помощью команды git reset HEAD.

5. Как отменить изменения в файле?

С помощью команды git checkout -- <file>.

6. Что такое удаленный репозиторий Git?

Удалённые репозитории представляют собой версии вашего проекта, сохранённые в интернете или ещё где-то в сети.

7. Как выполнить просмотр удаленных репозиториев данного локального репозитория?

С помощью команды git remote -v.

8. Как добавить удаленный репозиторий для данного локального репозитория?

С помощью команды git remote add <shortname> <url>.

9. Как выполнить отправку/получение изменений с удаленного репозитория?

Получение изменение – git fetch [remote-name], отправка изменений git push <remote-name> <branch-name>.

10. Как выполнить просмотр удаленного репозитория?
С помощью команды git remote -v.

11. Каково назначение тэгов Git?

Тэги Git - это ссылки на определенные коммиты в истории разработки. Они используются для пометки определенных версий или моментов в вашем проекте. Тэги обычно используются для обозначения релизов или важных этапов в разработке.

12. Как осуществляется работа с тэгами Git?

Для просмотра тэгов – git tag, для создания аннотированного тэга – git tag – a v1.4 -m “сообщение”, для отправки тэга на удаленный сервер – git push origin <tagname>, для удаления тэгов – git tag -d <tagname>.

13. . Самостоятельно изучите назначение флага --prune в командах git fetch и git push . Каково назначение этого флага?

Флаг --prune в командах git fetch и git push используется для удаления удаленных веток или тэгов, которые больше не существуют на удаленном репозитории. Если выполнить команду git fetch --prune, Git удалит локальные ссылки на удаленные ветки и тэги, которые были удалены на удаленном репозитории. Если выполнить команду git push --prune, Git удалит удаленные

ветки и тэги на удаленном репозитории, которых больше нет в вашем локальном репозитории.