**МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Северо-Кавказский федеральный университет» Кафедра инфокоммуникаций**

**Отчет по лабораторной работе №2**

**по дисциплине «Основы программной инженерии»**

|  |
| --- |
| Выполнил студент группы ПИЖ-б-о-20-1 |
| Ваньянц И.М. « » 20 г. |
| Подпись студента |
| Работа защищена « » \_20 г. |
| Проверил Воронкин Р.А.  (подпись) |

Ставрополь 2021

**ХОД РАБОТЫ**

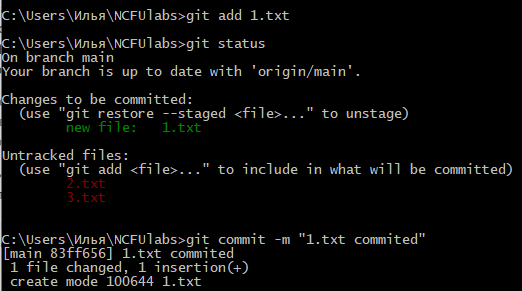


Рисунок 1 – коммит первого файла

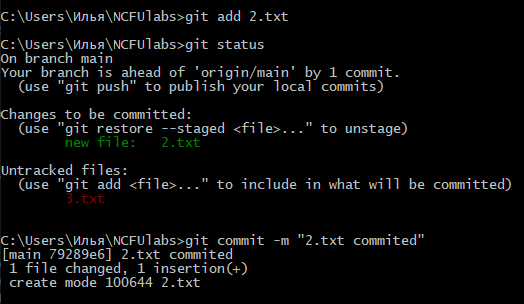


Рисунок 2 – коммит второго файла

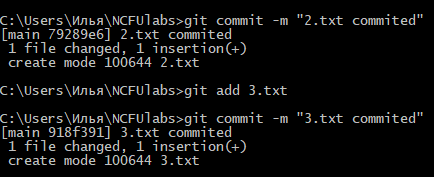


Рисунок 3 – коммит третьего файла

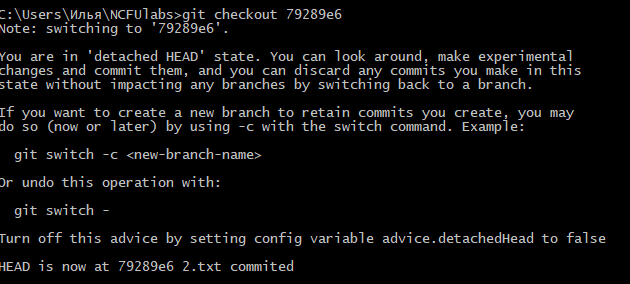


Рисунок 4 – сдвиг HEAD на два коммита назад

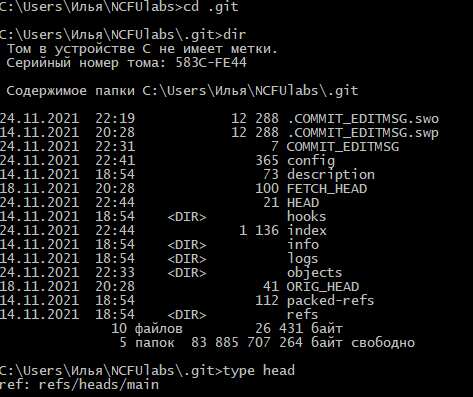


Рисунок 5 – просмотр содержимого репозитория

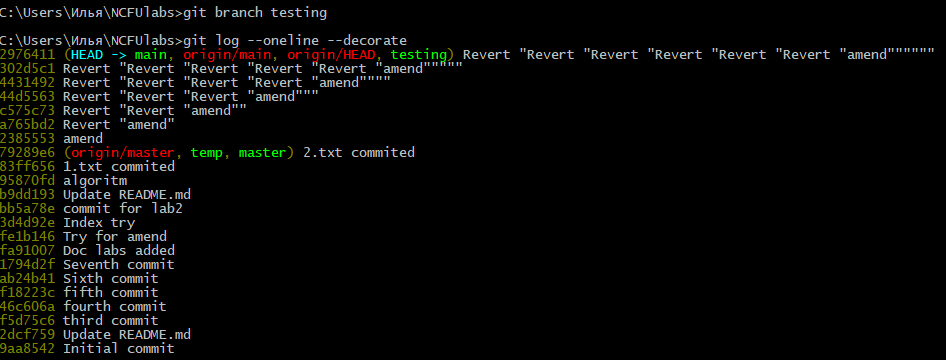


Рисунок 6 – создание ветви testing

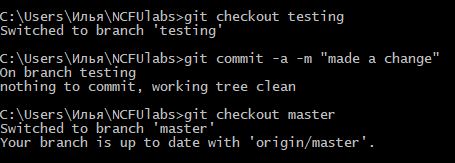


Рисунок 7 – перемещение по ветвям

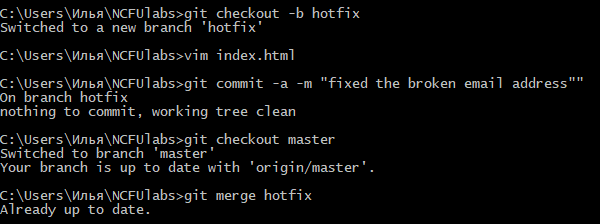
Рисунок 8 – переход на checkout и его слияние с master



Рисунок 9 – удаление ветви hotfix

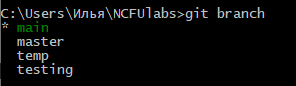


Рисунок 10 – список ветвей

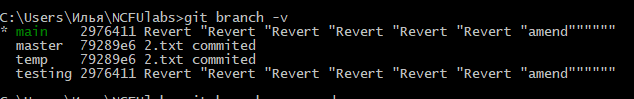


Рисунок 11 – просмотр коммитов в ветвях



Рисунок 12 – отправка изменений



Рисунок 13 – слежение за удаленной веткой

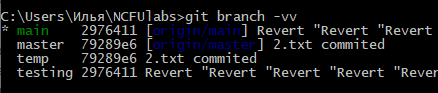


Рисунок 14 – просмотр веток слежения

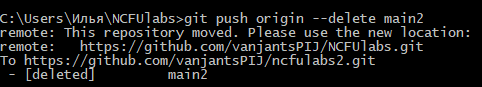


Рисунок 15 – удаление ветви на удаленном репозитории

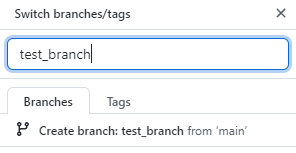


Рисунок 16 – создание новой ветки на github

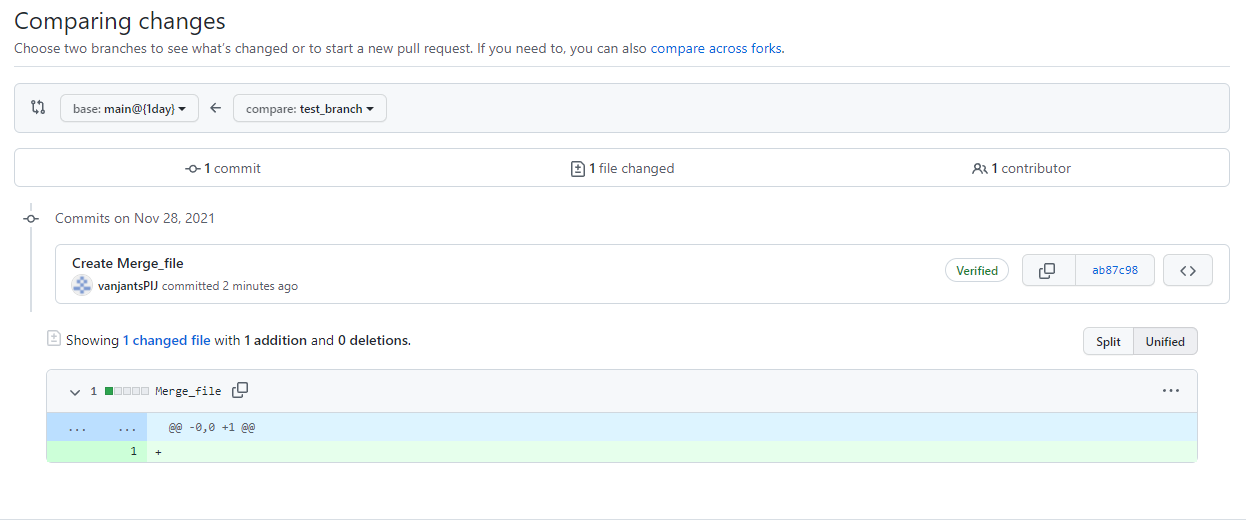


Рисунок 17 – создание pull request-а

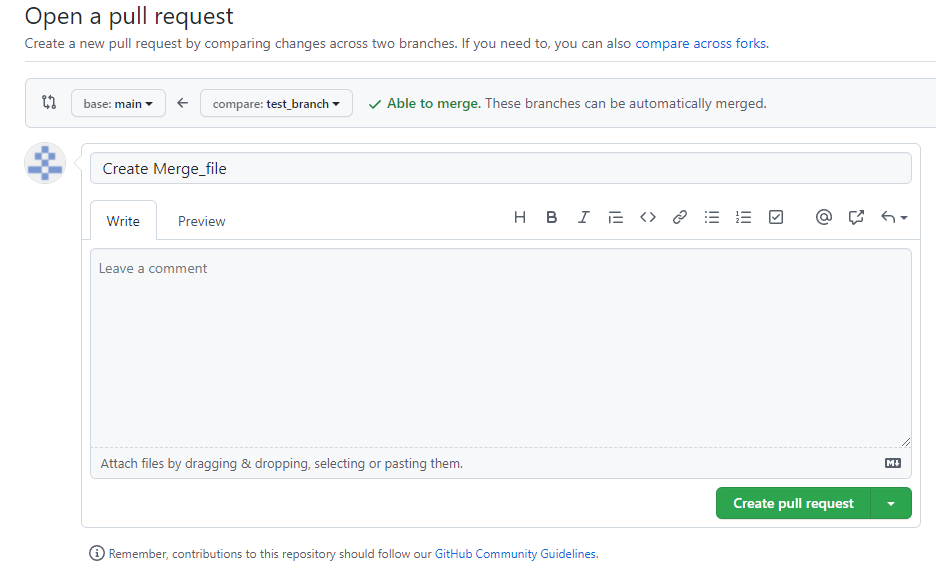


Рисунок 18 – отправка pull request-а

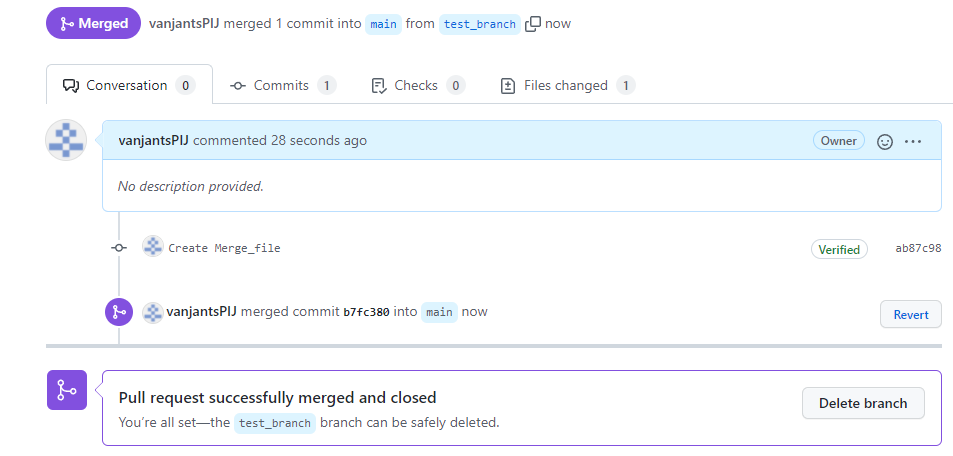


Рисунок 19 – слияние веток через pull request

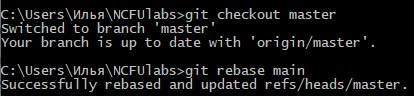


Рисунок 20 – пример перебазирования

**КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Ветка – указатель, который перемещается по коммитам.
2. Head – указатель на коммит в вашем репозитории, который станет родителем следующего коммита.
3. Ветки в git могут создаваться: при помощи git init – создается ветка master; при помощи команды git branch; используя команду git checkout –b – сразу переключает на созданную ветку.
4. При помощи команды git status можно узнать, на какой ветке вы находитесь.
5. Переключение между ветками происходит при помощи команды git checkout
6. Ветка, которая находится на удалённом репозитории.
7. Ветка, которая отслеживает изменения другой ветки, и копирует эти изменения.
8. Отслеживать можно при помощи команды git checkout –track.
9. Отправить изменения на удалённую ветку можно при помощи команды – git push origin имя\_ветки.
10. Git fetch получает с сервера все изменения, которых ещё нет, но не будет изменять состояние рабочей директории. Git pull – это команда git fetch только идущая в связке с командой git merge.
11. Чтобы удалить ветку на удалённом сервере используется команда git push origin –delete имя\_ветки. Локальная ветка удаляется с командой git push –d.
12. В git-flow есть следующие типы веток: feature branches, release branches, hotfix branches. Модель работы с ветками следующая: 1. Из master создается ветка develop, из develop создаются ветки feature, когда разработка новой функциональности завершена, она объединяется с веткой develop, из develop создается ветка release, и когда ветка релиза готова, она объединяется с develop и master. Минусы git-flow: может замедлять работу, когда приходится отправлять большие пулл реквесты, когда итерацию нужно выполнить быстро, релизы сложно делать чаще раза в неделю, история проекта имеет кучу merge коммитов и затрудняет просмотр реальной работы.
13. Gitk показывает ветку коммитов на экране, что помогает более точно работать с коммитами.