

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО «МГТУ «СТАНКИН»)**

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | Институт цифровых интеллектуальных систем | Кафедра  компьютерных систем управления | |

Дисциплина «Основы системного программного обеспечения»

**Отчет по лабораторной работе №1**

|  |  |  |
| --- | --- | --- |
| **Выполнил**  **студент гр. АДБ-21-07:** | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  *(дата) (подпись)* | **Ермачков А.Е.** |
| **Проверил**  **к.т.н., доцент** | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  *(дата) (подпись)* | **Ковалев И.А.** |

**Москва 2024 г.**

**Содержание**

[Ход работы 3](#_Toc168046328)

[Вывод 13](#_Toc168046329)

**Тема работы:** Работа с системами контроля версий на примере Git Hub.

**Цель работы:** Получить навыки работы с системой контроля версий на примере Git Hub.

**Задачи работы:**

1. Изучить теоретический материал к лабораторной работе.
2. Проанализировать ход выполнения лабораторной работы.
3. Выполнить лабораторную работу в соответствии с методическими указаниями.

# Ход работы

Создадим репозиторий на GitHub:

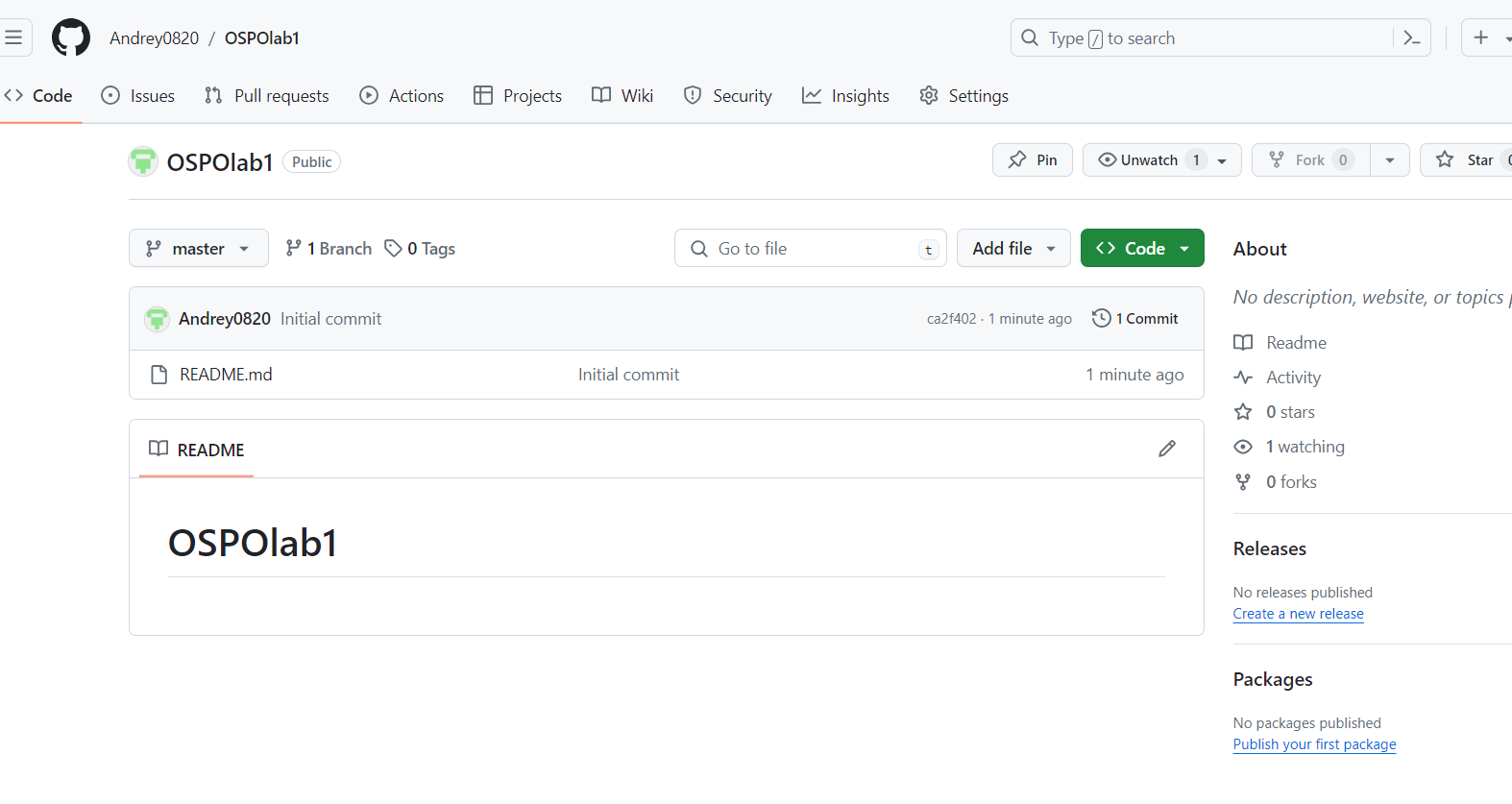


Рис. 1. Созданный репозиторий для выполнения лабораторной работы.

Создадим локальный репозиторий на компьютере:

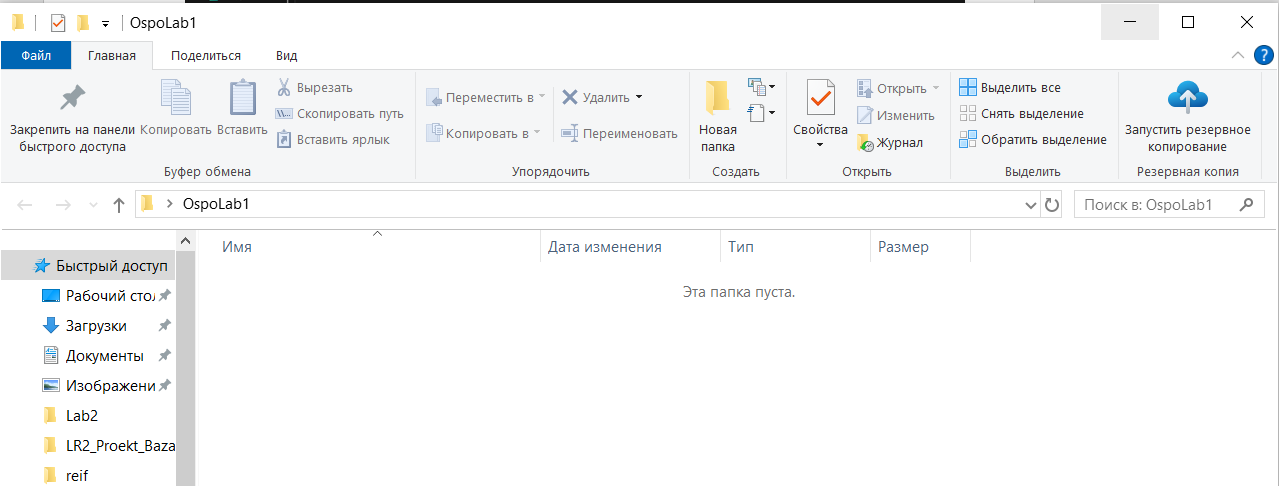


Рис. 2. Создание репозитория.

Теперь, когда мы создали локальный репозиторий на компьютере, настроим имя пользователя и адрес электронной почты. Также инициируем git в локальном репозитории:

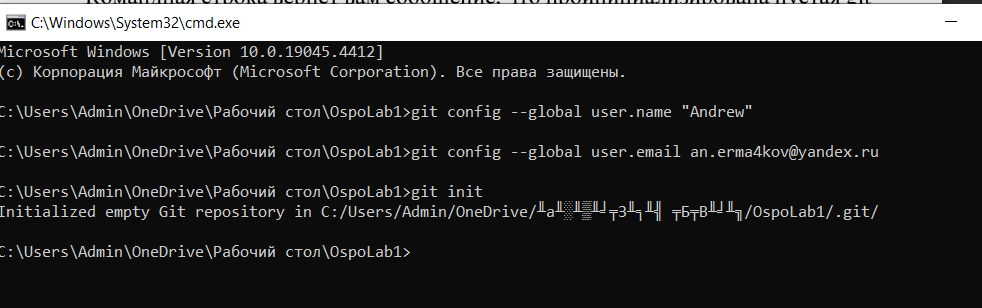


Рис. 3. Настройка имени пользователя и адреса электронной почты; инициализация git в локальном репозитории.

Создадим файл в локальном репозитории и выполним команду git status:

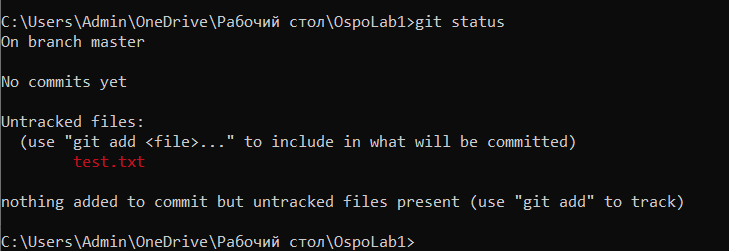


Рис. 4. Результат выполнения команды git status после создания файла в локальном репозитории.

Добавим файл для отслеживания изменений с помощью команды git add:

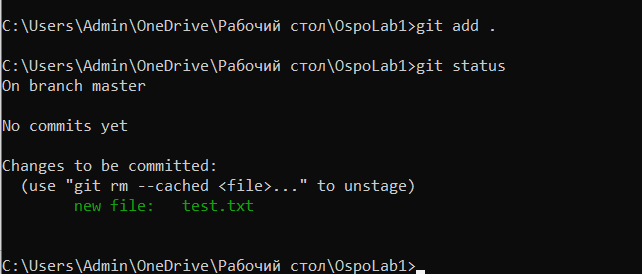


Рис. 5. Результат выполнения команды git add .

Создадим коммит с помощью команды git commit:

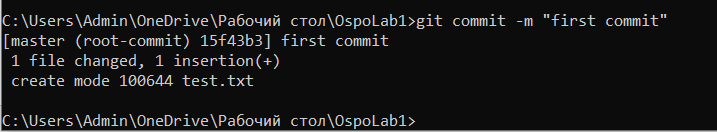


Рис. 6. Результат выполнения команды git commit -m “first commit”.

Чтобы отправить коммит на сервер, нужно связать локальный репозиторий с репозиторием на GitHub. Для этого воспользуемся командой git remote add origin <ссылка на репозиторий>. Также воспользуемся командой git remote -v для проверки к какому проекту мы подключены:

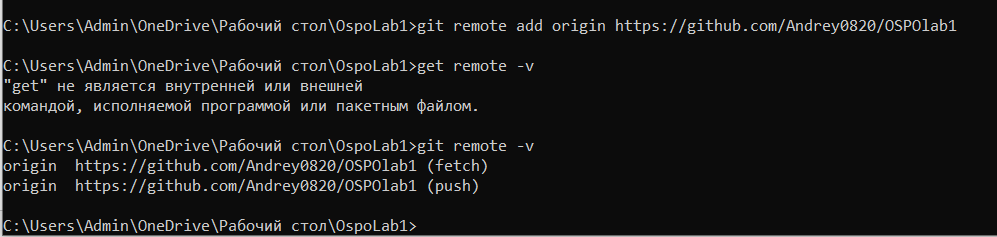


Рис. 7. Результат выполнения команд git remote add origin и git remote -v.

Запросим изменения с сервера с помощью команды git pull origin master:

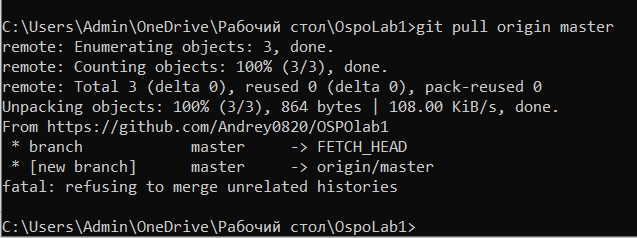


Рис. 8. Результат выполнения команды git pull origin master.

Отправим локальный коммит на сервер с помощью команды git push origin master:

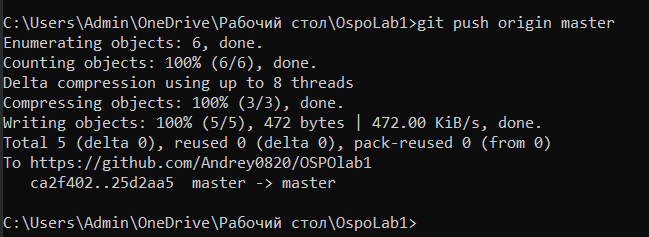


Рис. 9. Результат выполнения команды git push origin master.

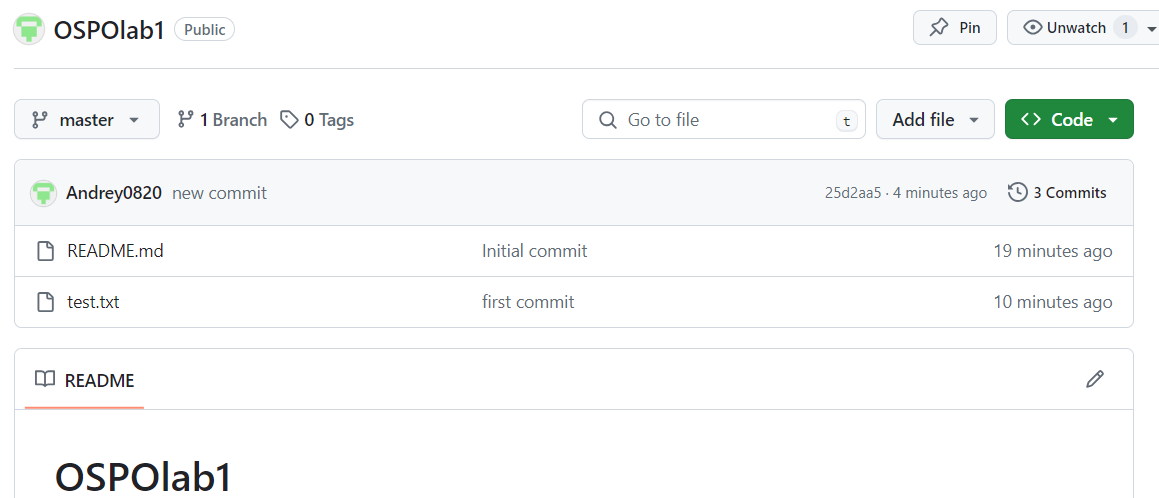


Рис. 10. Изменения на github.com.

Создадим новую ветку с помощью команды git branch и переключимся на нее, используя команду git checkout second:

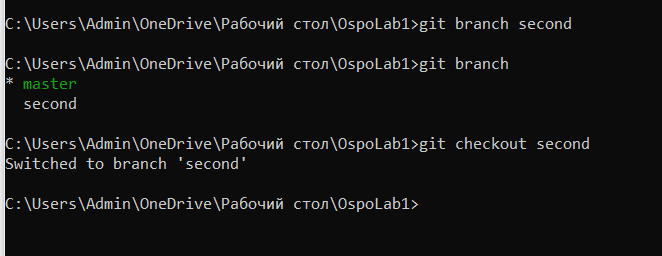


Рис. 11. Результат выполнения команды git branch second и git checkout second.

Создаём новый файл в нашем локальном репозитории с фамилией, добавим в область подготовленных файлов, закоммитим и отправим на сервер:

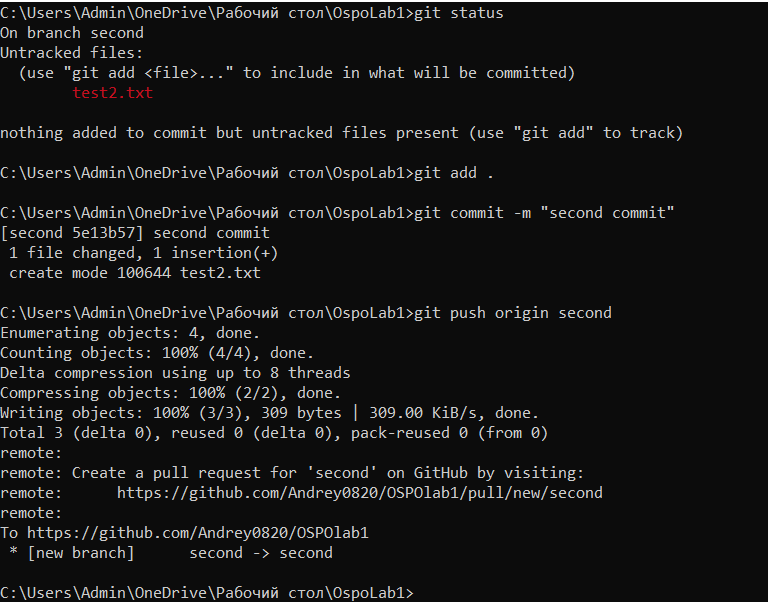


Рис. 12. Отправка нового коммита на сервер на ветку second.

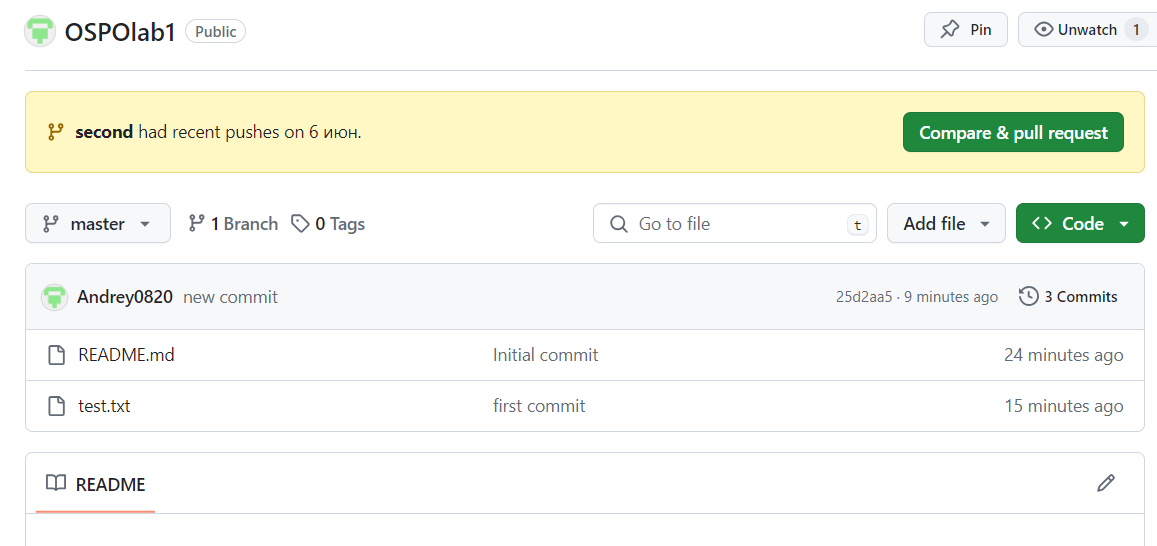


Рис. 13. Проверка отсутствия файла test2.txt в ветке master.

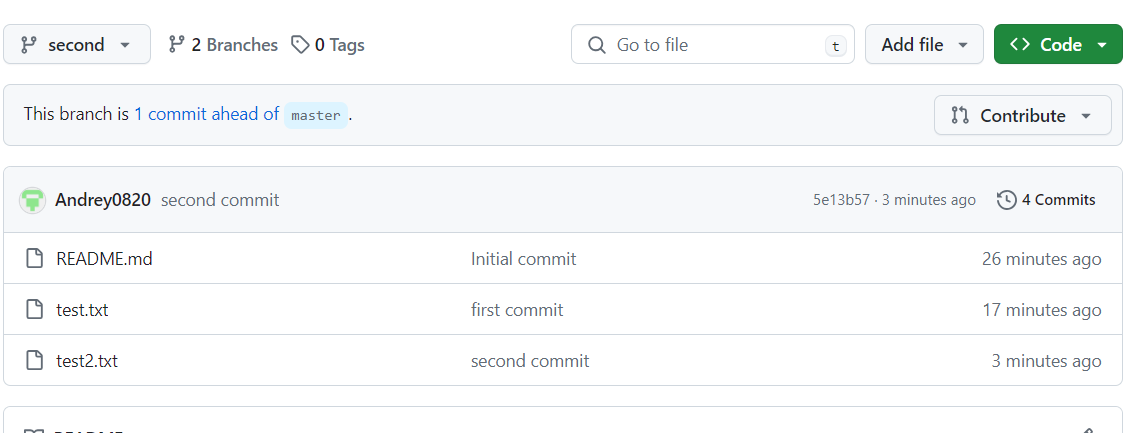


Рис. 14. Проверка наличия файла test2.txt в ветке second.

Теперь сделаем слияние веток master и second. Для этого вначале переключимся на ветку master и выполним команду git merge second. Также после успешного слияния удалим ветку second с помощью команды git branch -d second:

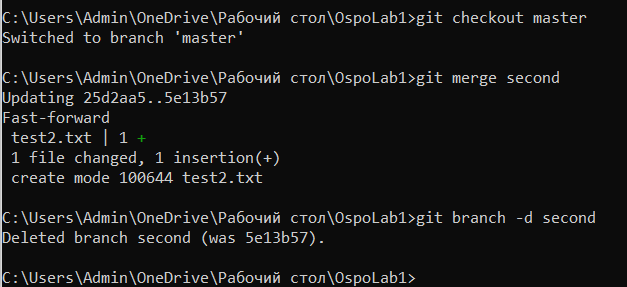


Рис. 15. Результат выполнения команд git merge second и git branch -d second.

Усложним задачу и создадим ситуацию, когда в двух ветках есть одинаковые файлы, над которыми работают разные разработчики. Создадим новую ветку с названием newdev и переключимся на нее:

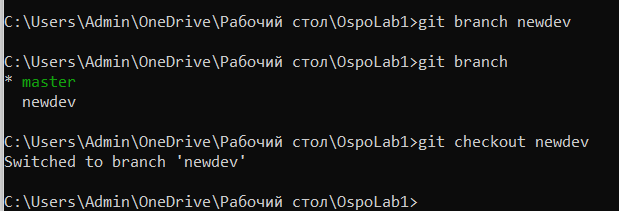


Рис. 16. Успешное создание и переключение на новую ветку newdev.

Добавим в файл text1.txt отчество, зафиксируем изменения и закоммитим их:

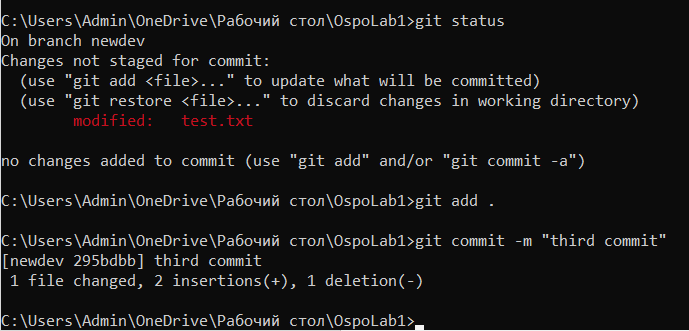


Рис. 17. Успешное изменение файла test.txt и фиксирование изменений в git.

Теперь переключимся на ветку master и проверим файл test.txt и убедимся, что изменений в нем нет:

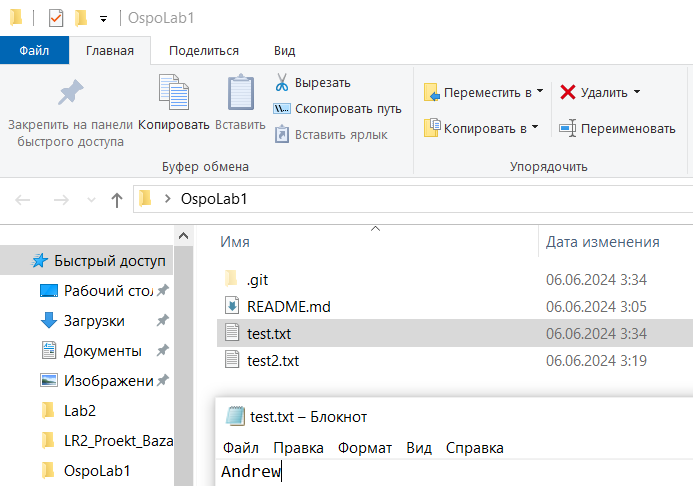


Рис. 18. Проверка файла test.txt на ветку master на изменения.

Проведем объединение ветки master с newdev и посмотрим на результат:

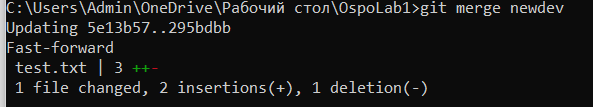


Рис. 19. Успешное слияние ветки master с newdev.

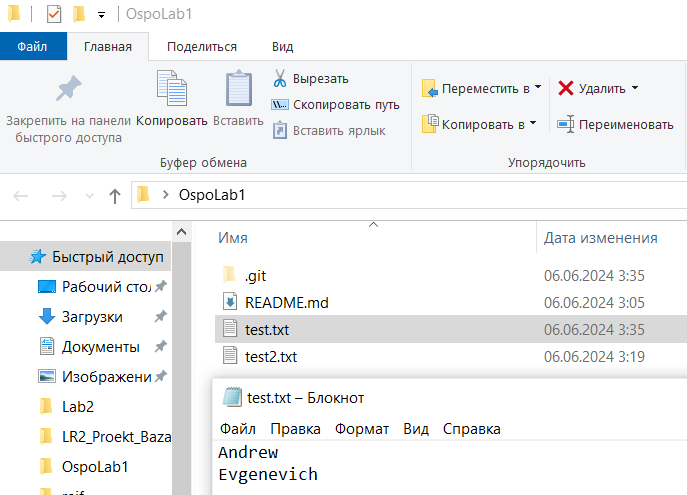
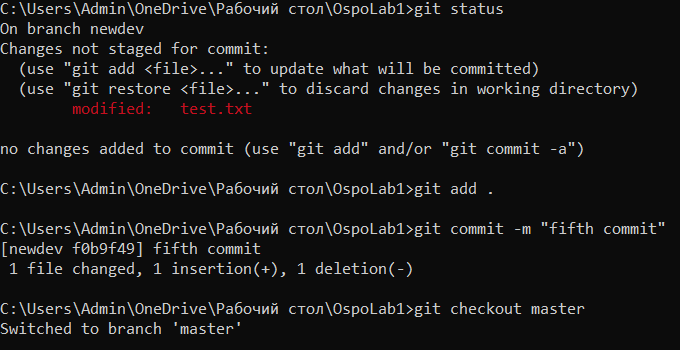
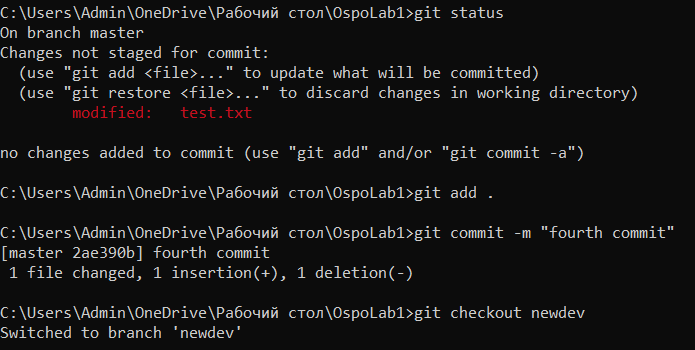


Рис. 20. Проверка наличия изменений в файле test.txt после слияния ветки master c newdev.

Сделаем изменения в файле test.txt в двух ветках и после объединим их:



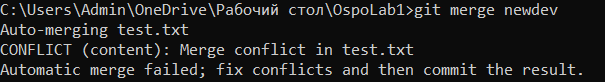


Рис. 21. Создание изменений в двух ветках master и newdev, а также их слияние.

Просмотрим изменения, используя команду git diff:

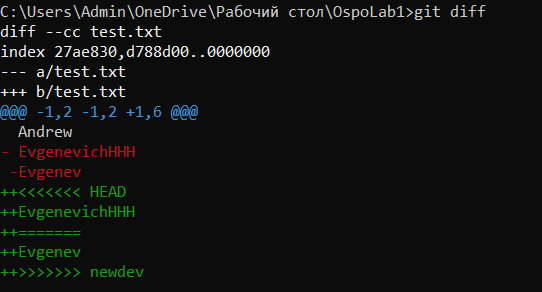


Рис. 22. Результат выполнения команды git diff.

Исправим конфликт в файле test.txt и отправим изменения на сервер:

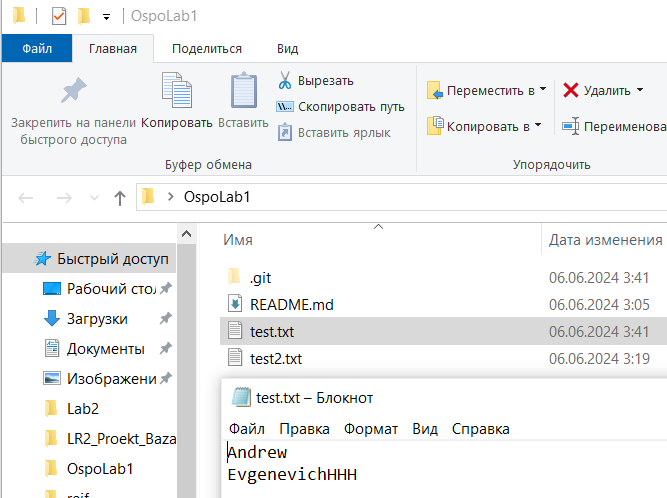


Рис. 23. Исправление конфликта в файле test.txt.

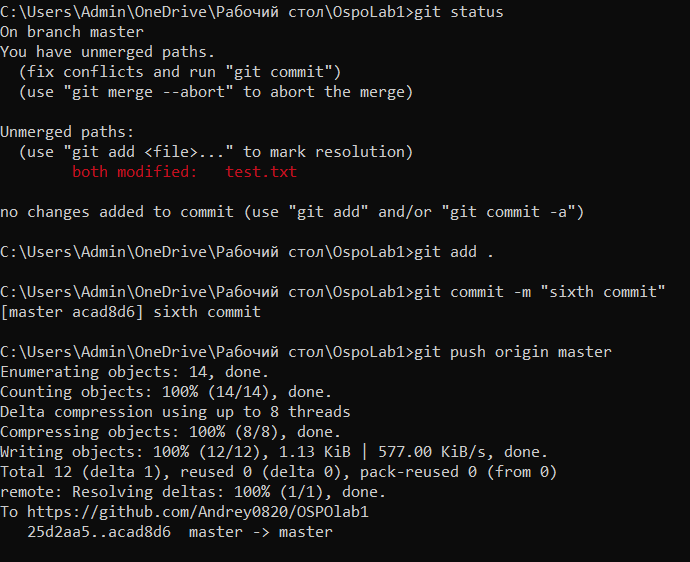


Рис. 24. Фиксирование изменения и отправка не сервер.

Теперь удалим ветки из github:

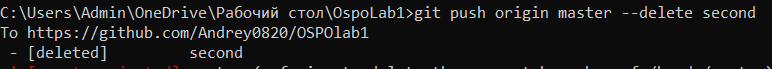


Рис. 25. Удаление веток newdev и second на github.com.

Ссылка на репозиторий, использованный при составлении отчета:

<https://github.com/Andrey0820/OSPOlab1>

# Вывод

В ходе выполнения лабораторной работы были получены навыки по работе с системой контроля версий на примете Git Hub: фиксация изменений файлов; создание и отправка коммитов на сервер; создание новых веток и их слияние; разрешение конфликтов после слияния двух веток, где изменялись одинаковые файлы разными разработчиками.