Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

**«Основы ветвления Git»**

**ОТЧЕТ**

**по лабораторной работе №3**

**дисциплины**

**«Основы программной инженерии»**

|  |  |  |
| --- | --- | --- |
|  | Выполнил:  Гълбачева Доротея Андреева  2 курс, группа ПИЖ-б-о-21-1,  09.03.04 «Программная инженерия», направленность (профиль) «Разработка и сопровождение программного обеспечения», очная форма обучения  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | Проверил:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | |  | |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_ Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2022 г.

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

**Методика и порядок выполнения работы:**

1. Создать три файла: 1.txt, 2.txt, 3.txt.

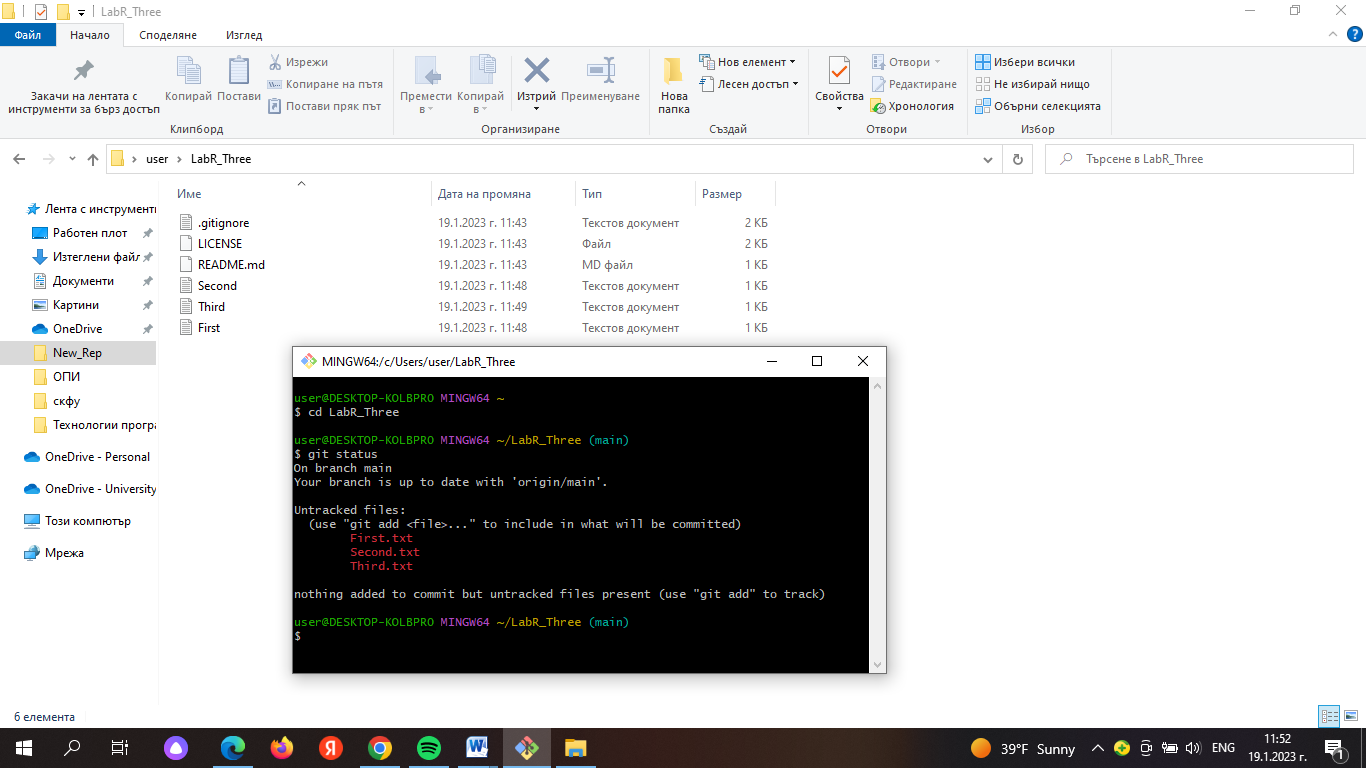


Рисунок 3.1 – Созданые файлов

1. Проиндексировать первый файл и сделать коммит с комментарием "add 1.txt file".

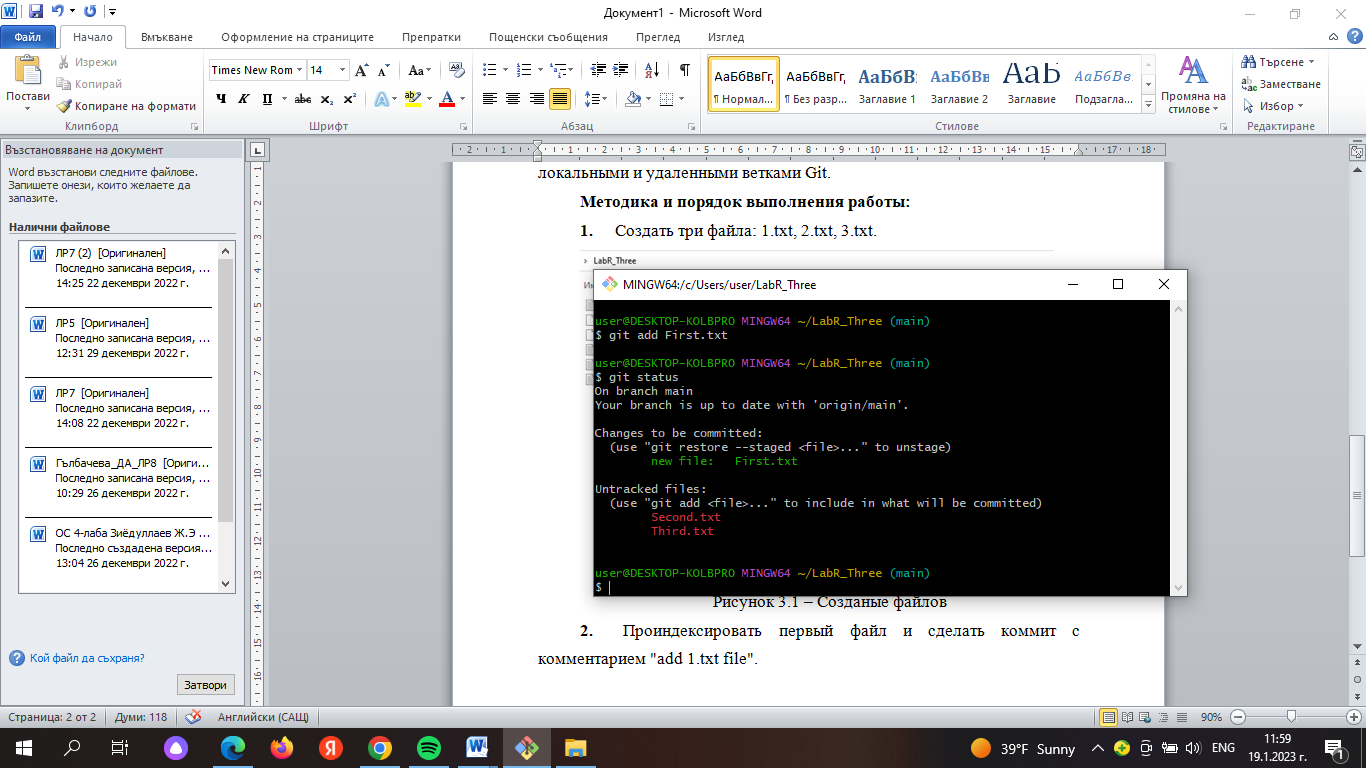


Рисунок 3.2 – Индексация первого файла

1. Проиндексировать второй и третий файлы.

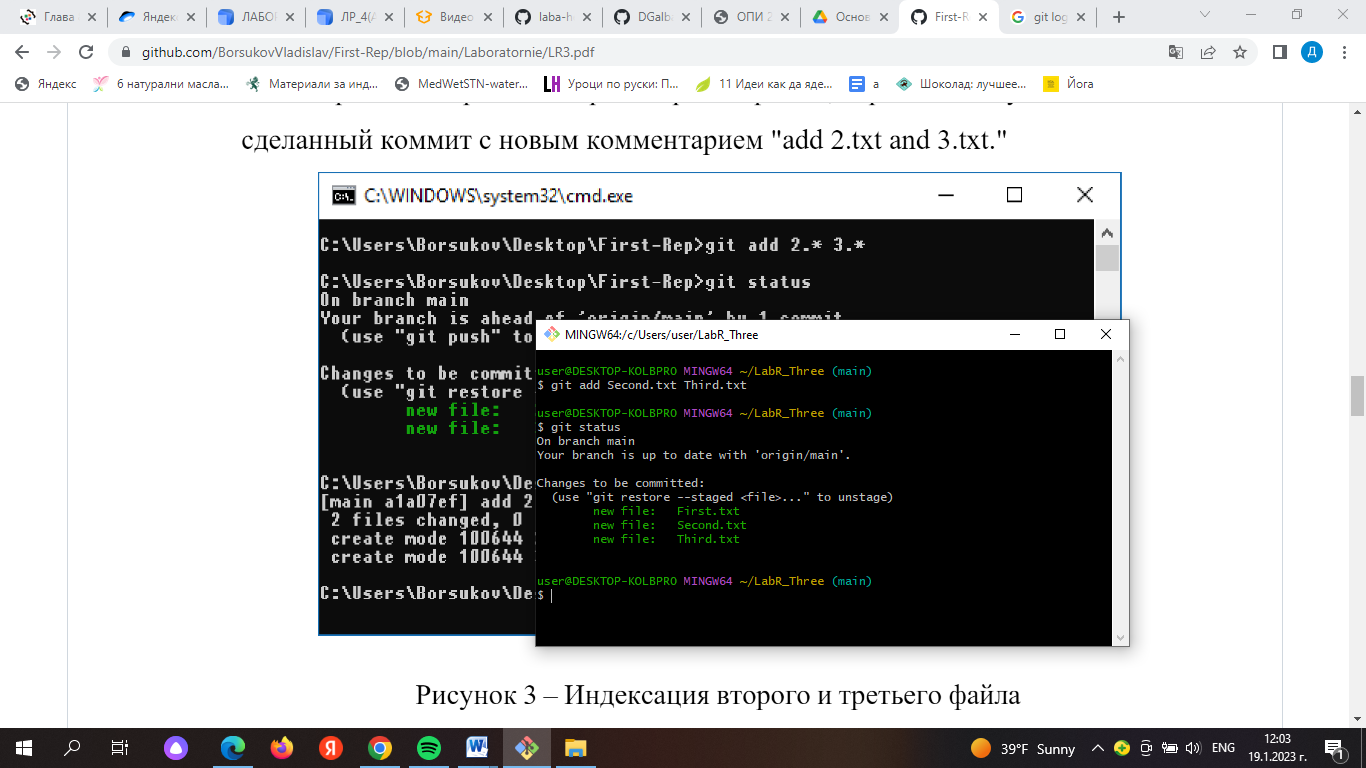


Рисунок 3.3 – Индексация второго и третьего файла

1. Перезаписать уже сделанный коммит с новым комментарием "add 2.txt and 3.txt."

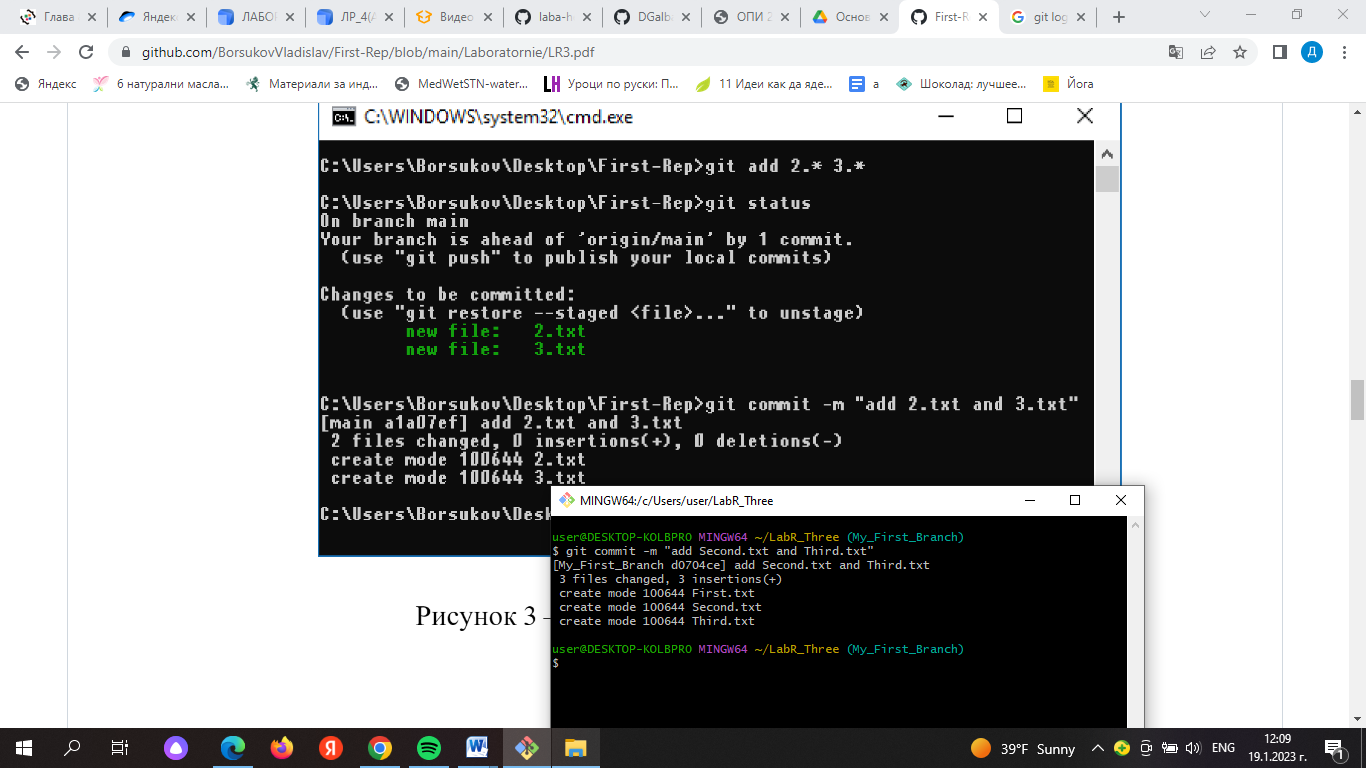


Рисунок 3.4 – Перезапис коммита

1. Создать новую ветку my\_first\_branch.

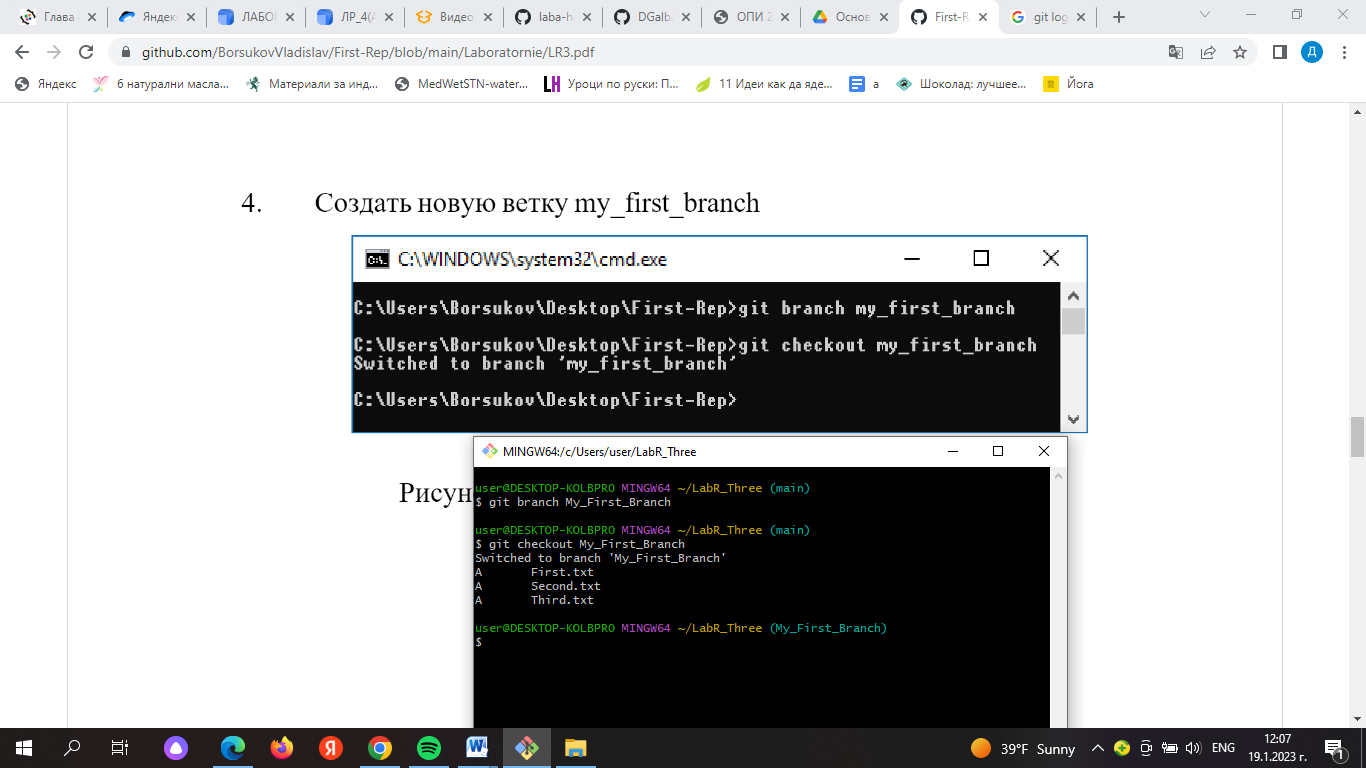


Рисунок 3.4 – Создание новой ветки «My\_First\_Branch»

1. Перейти на ветку и создать новый файл in\_branch.txt, закоммитить изменения.

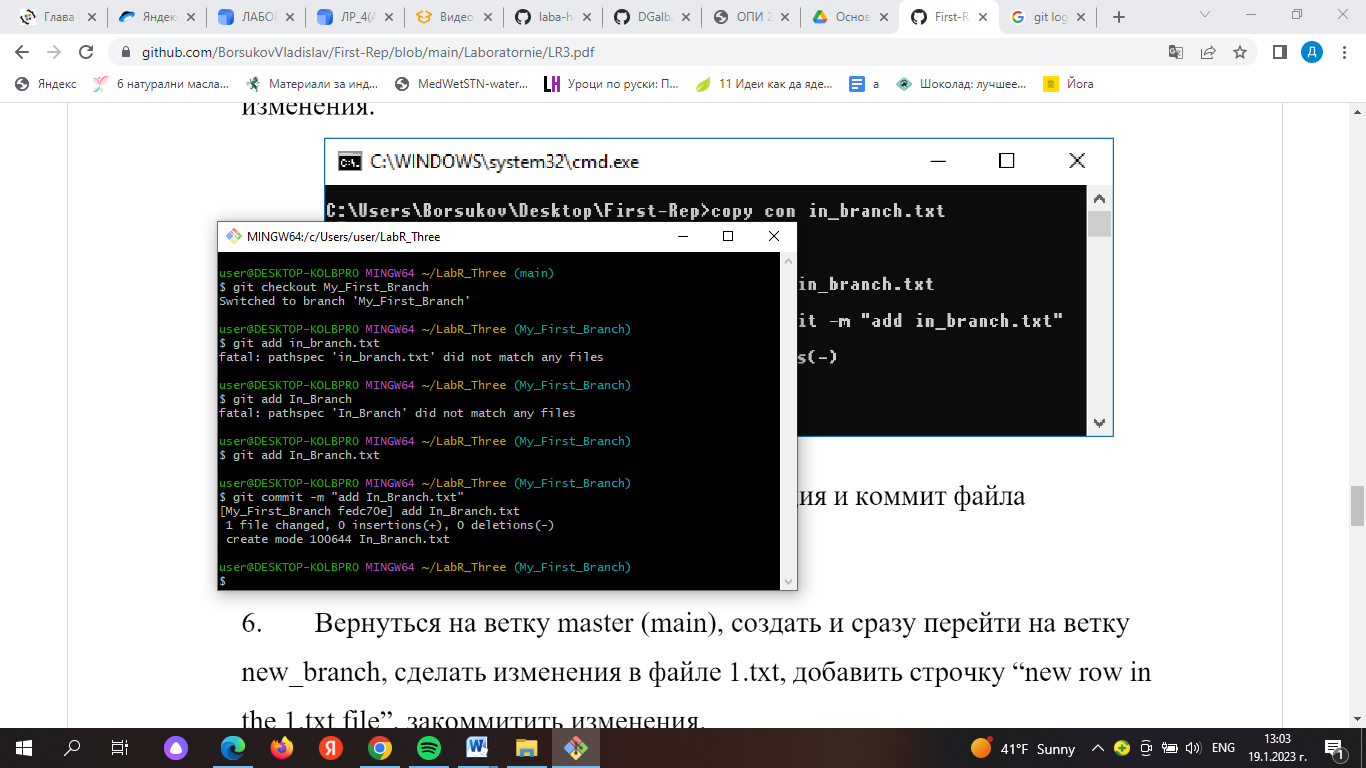


Рисунок 3.6 – Индексация и коммит файла

1. Вернуться на ветку master. Создать и сразу перейти на ветку new\_branch. Сделать изменения в файле 1.txt, добавить строчку “new row in the 1.txt file”, закоммитить изменения.

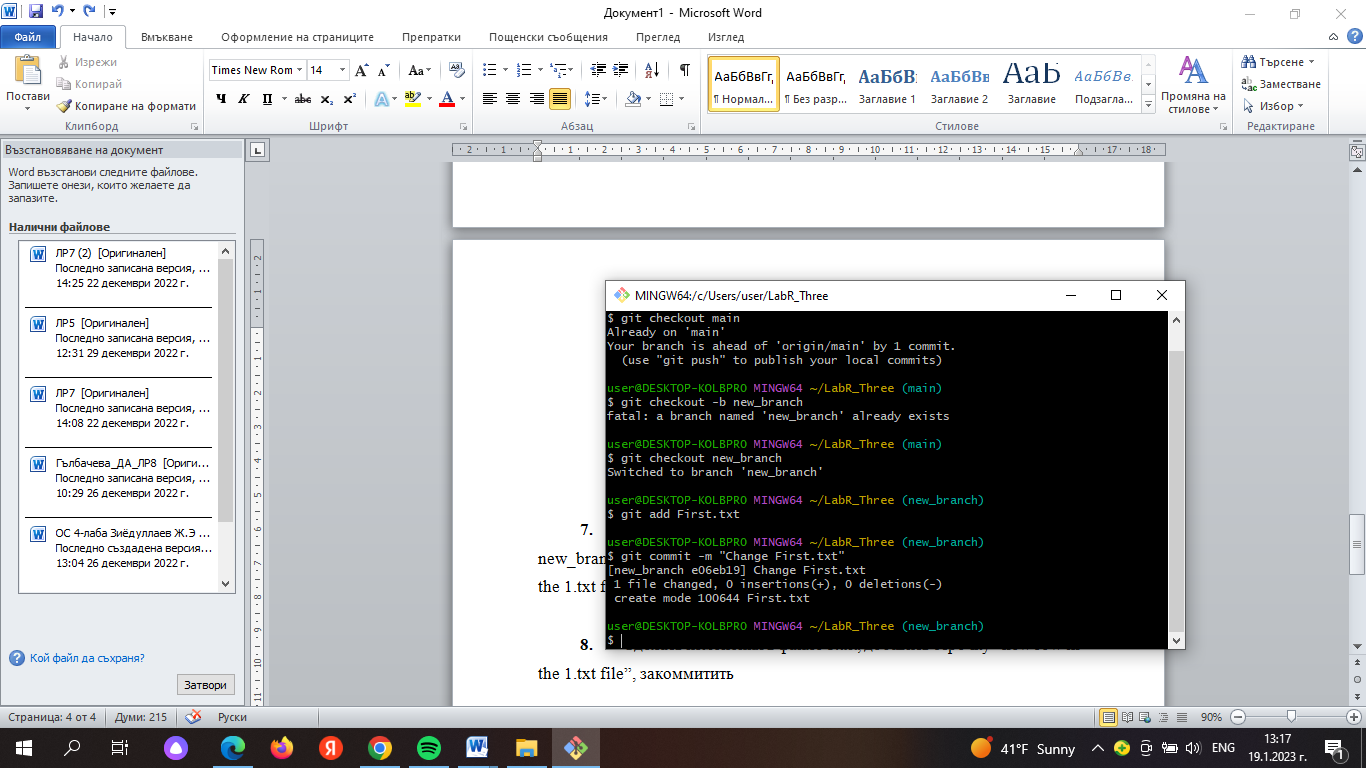


Рисунок 3.7 – Возврат к ветке (main), создание новой ветки и сразу переходом на нее и добавление изменений в файл с последующим коммитом

1. Перейти на ветку master и слить ветки master и my\_first\_branch, после чего слить ветки master и new\_branch.

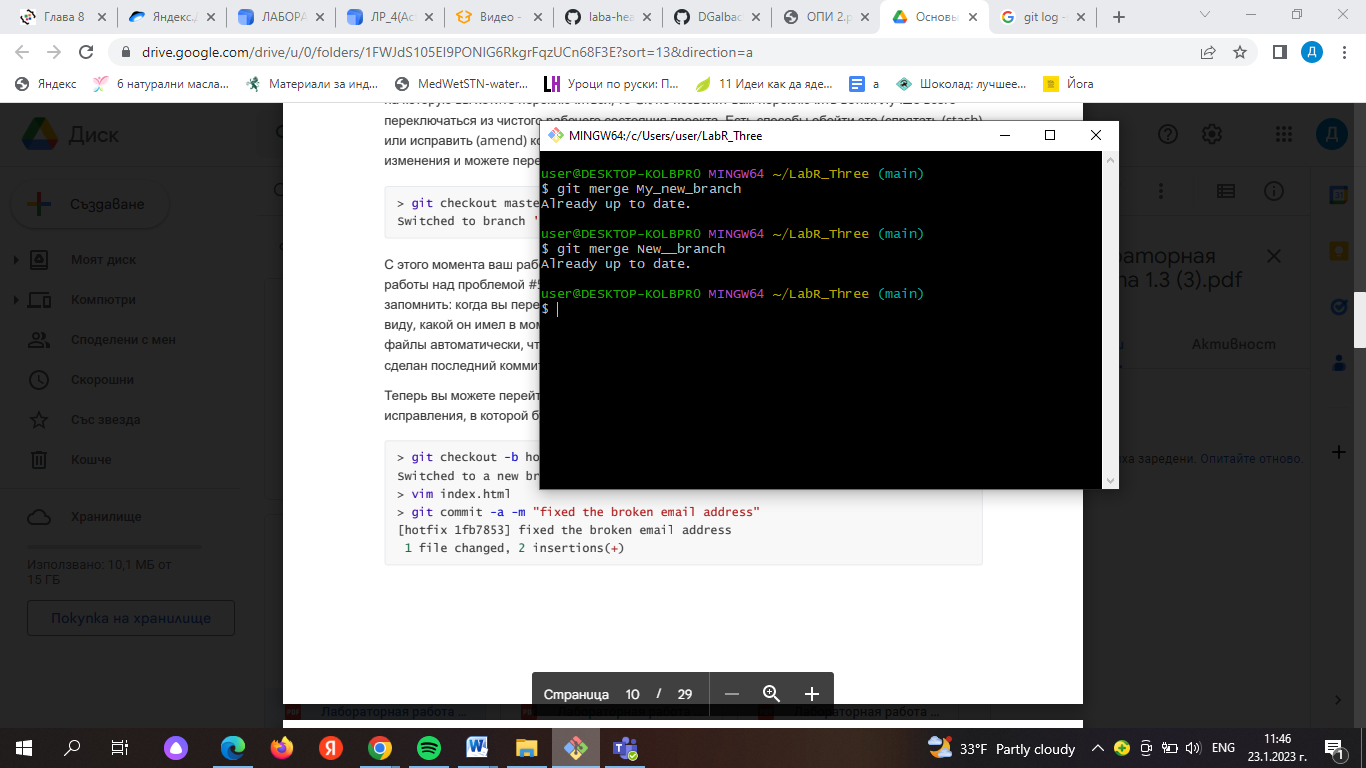


Рисунок 3.8 – Слияние веток

1. Удалить ветки my\_first\_branch и new\_branch.

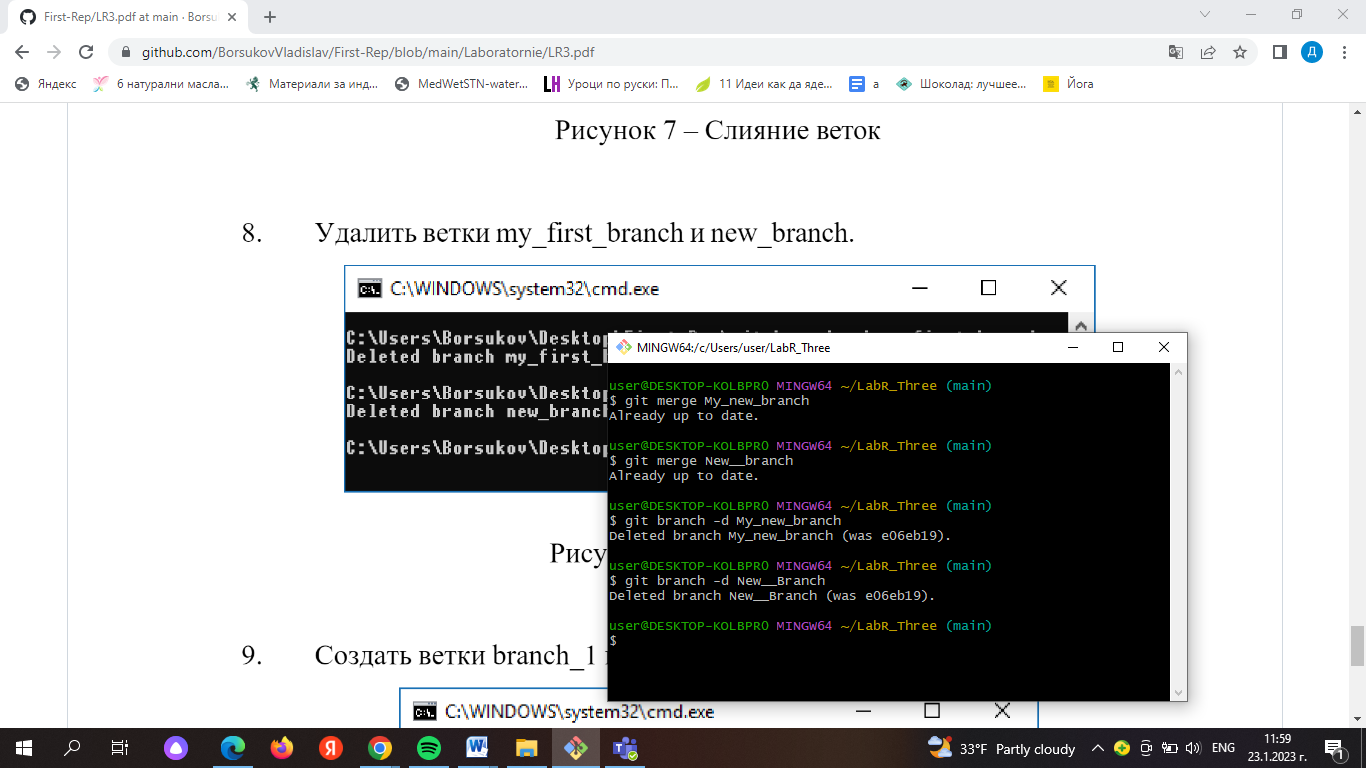


Рисунок 3.9 – Удаление веток

1. Создать ветки branch\_1 и branch\_2.

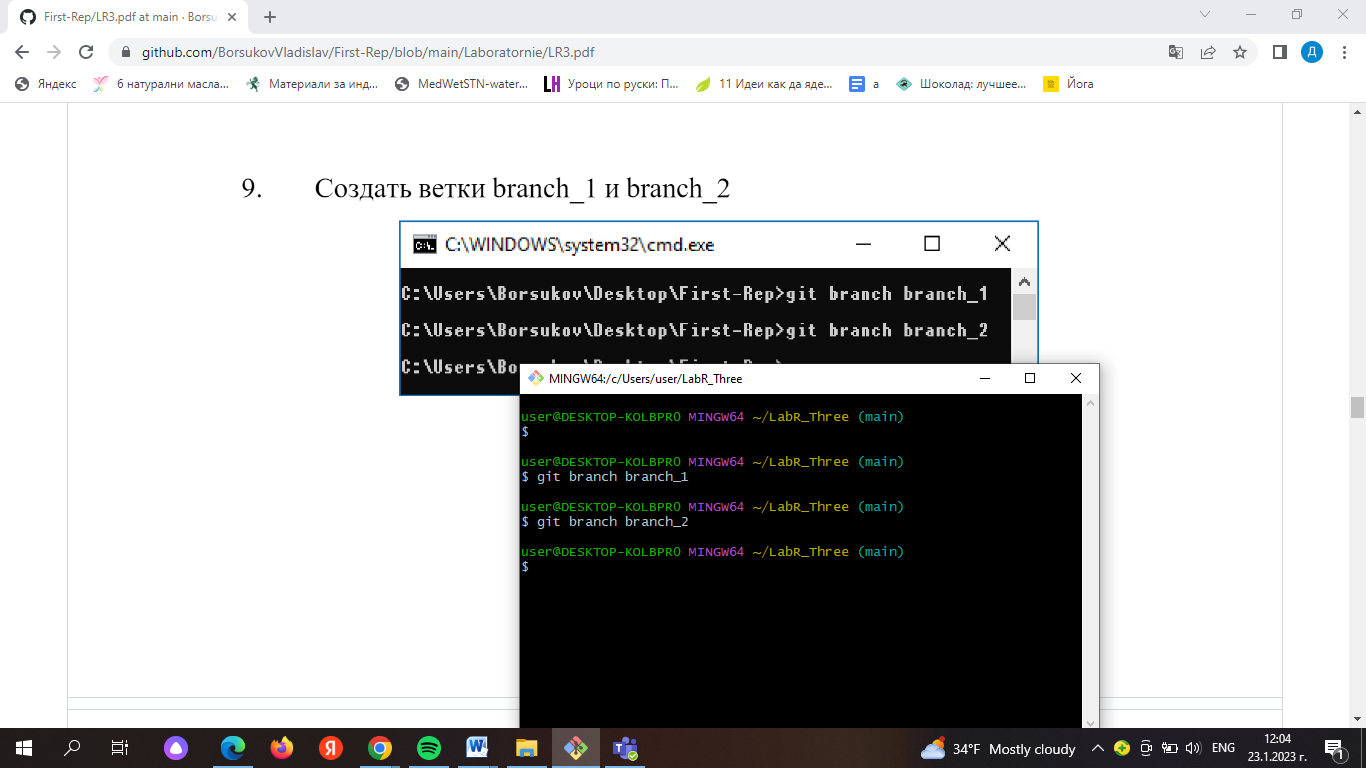


Рисунок 3.10 – Создание веток

1. Перейти на ветку branch\_1 и изменить файл 1.txt, удалить все содержимое и добавить текст “fix in the 1.txt”, изменить файл 3.txt, удалить все содержимое и добавить текст “fix in the 3.txt”, закоммитить изменения.

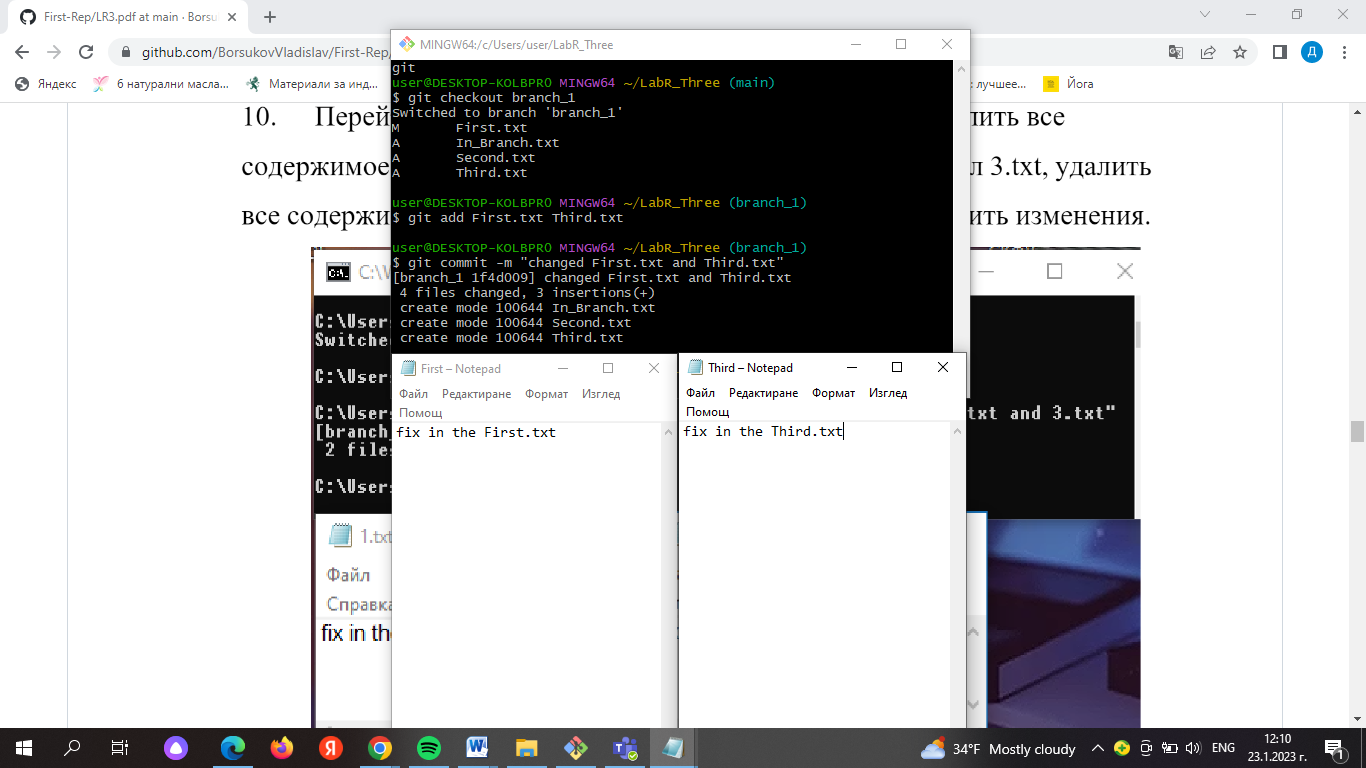


Рисунок 3.11 – Переход на ветку branch\_1, изменение файлов

1. Перейти на ветку branch\_2 и также изменить файл 1.txt, удалить все содержимое и добавить текст “My fix in the 1.txt”, изменить файл 3.txt, удалить все содержимое и добавить текст “My fix in the 3.txt”, закоммитить изменения.

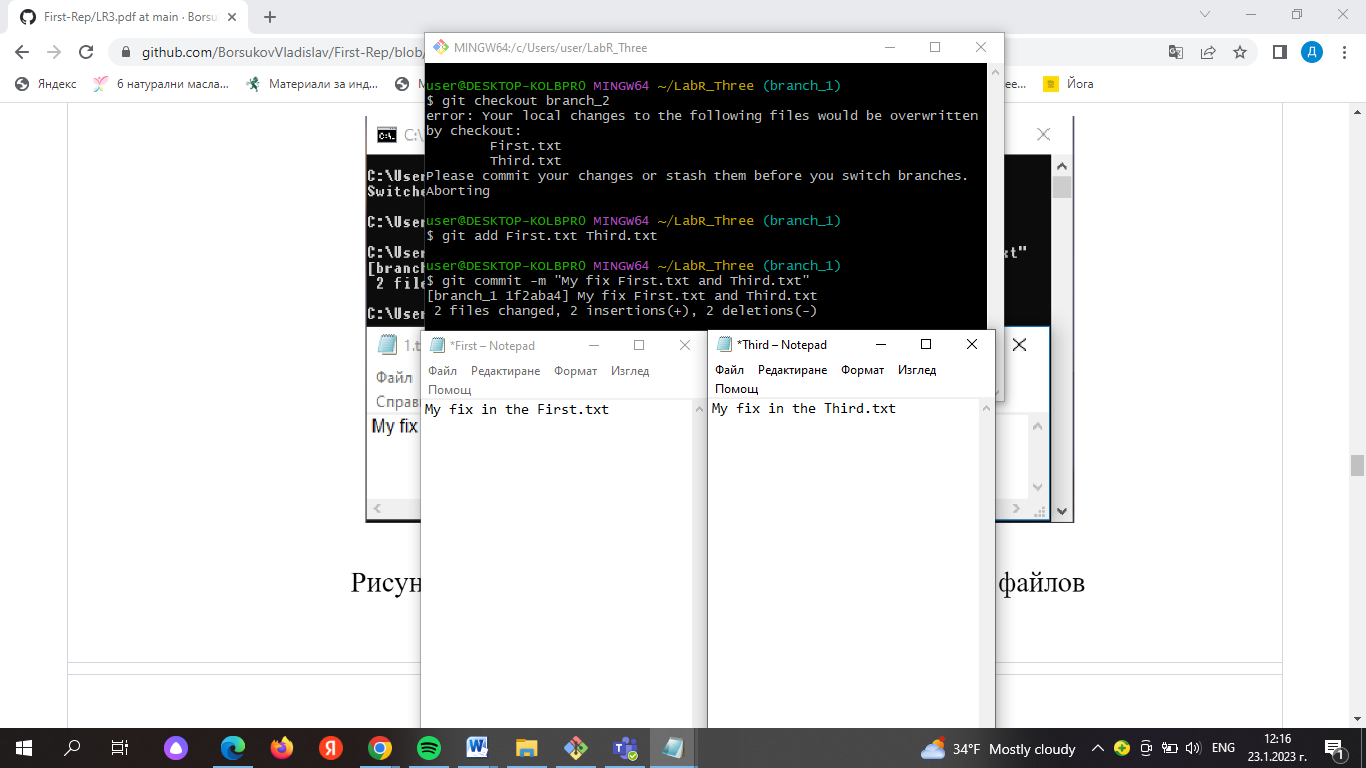


Рисунок 3.12 – Переход на ветку branch\_2 и изменение файлов

1. Слить изменения ветки branch\_2 в ветку branch\_1.

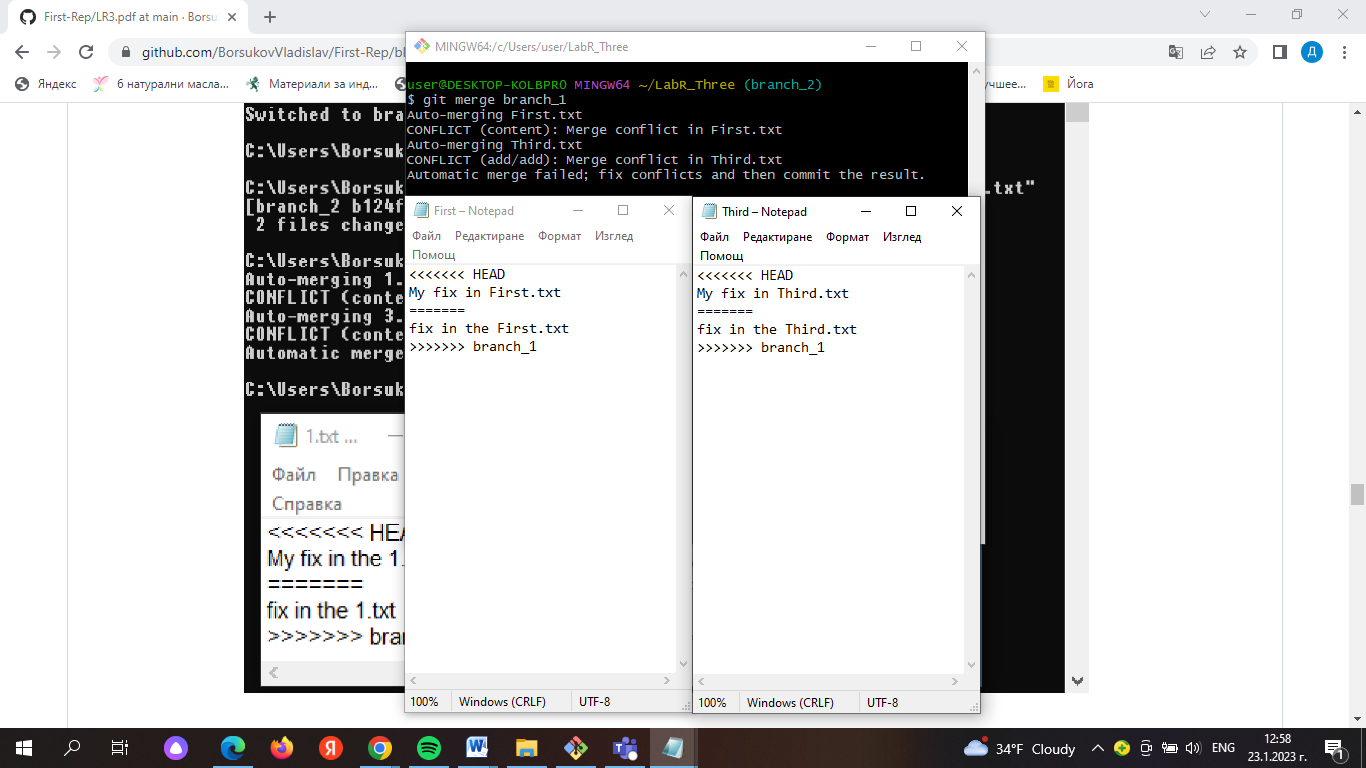


Рисунок 3.13 – Результат слияния веток

1. Решить конфликт файла 1.txt в ручном режиме, а конфликт 3.txt используя команду git mergetool c помощью одной из доступных утилит, например Meld.

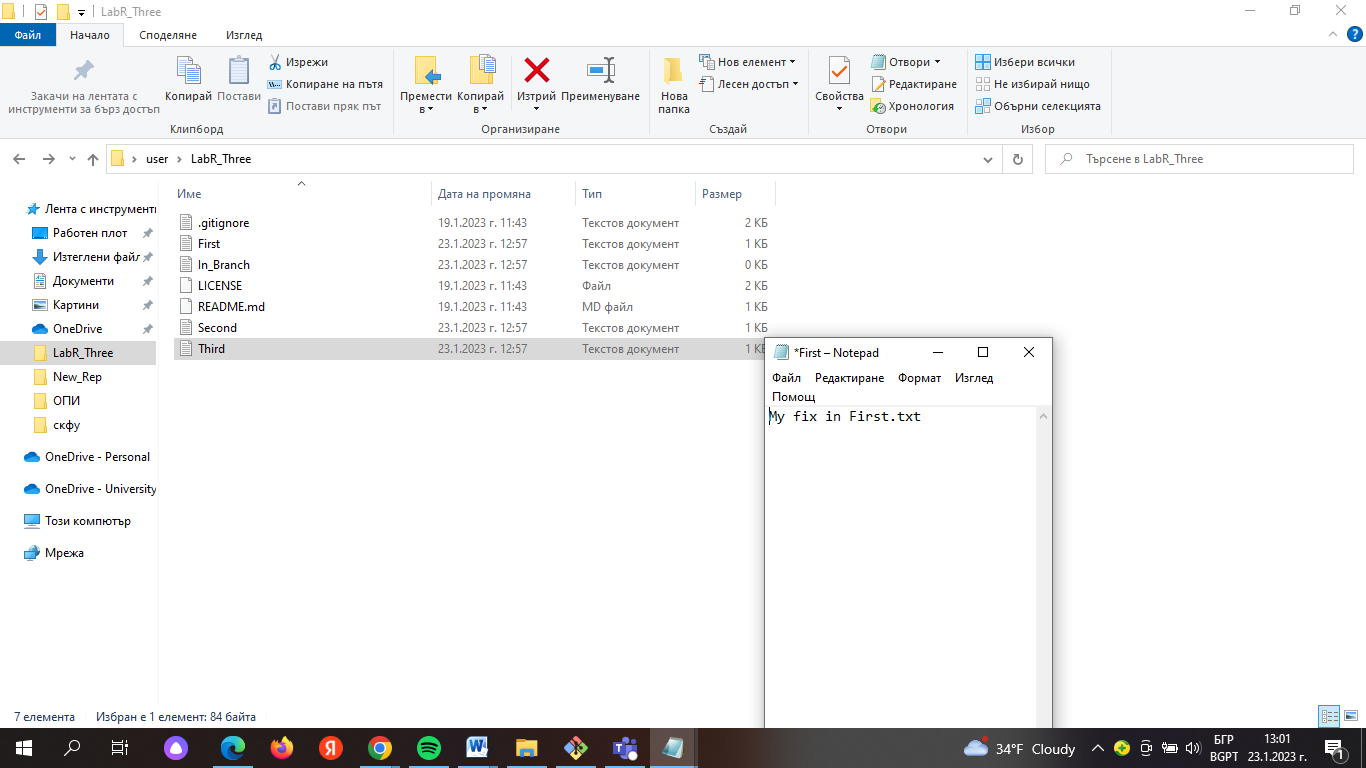


Рисунок 3.14 – Решение конфликта файлов вручную

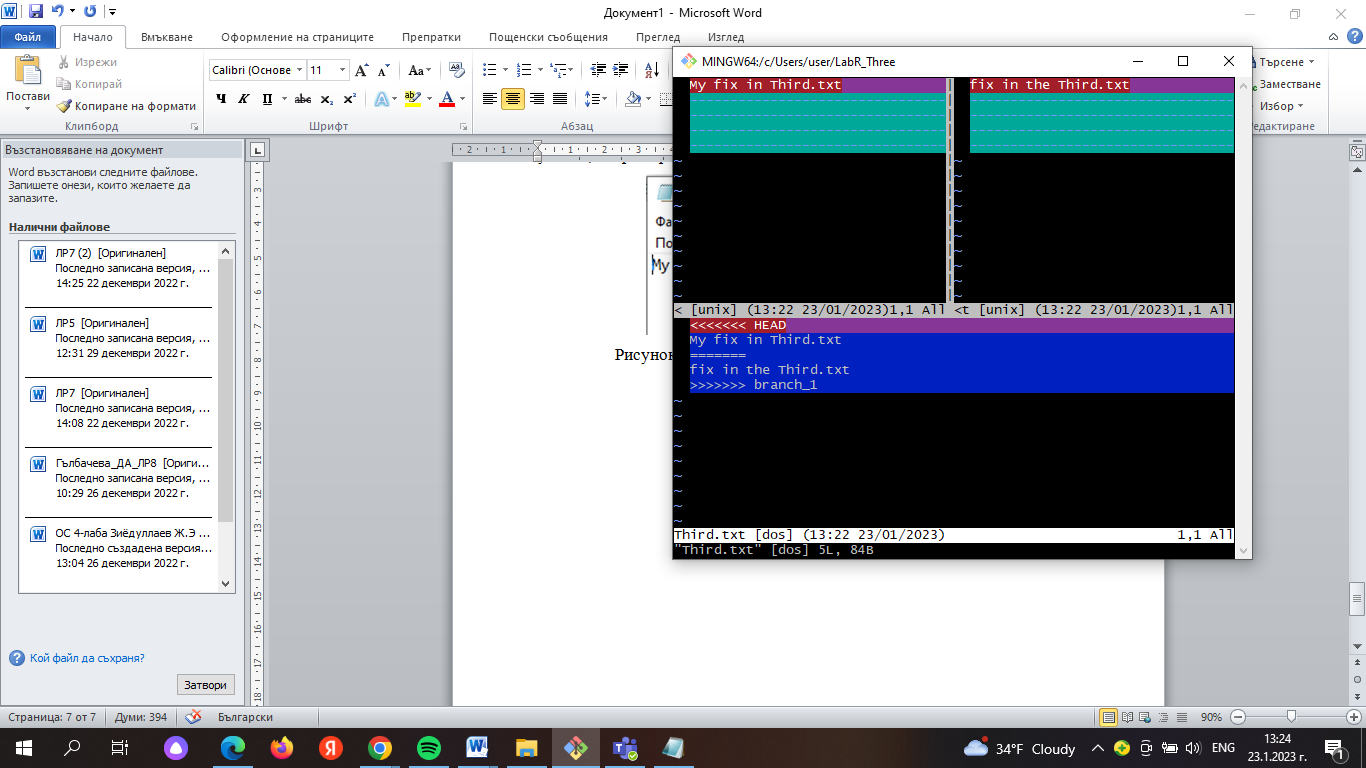


Рисунок 3.15 – Решение конфликта файлов через git mergetool с использованием vimdiff

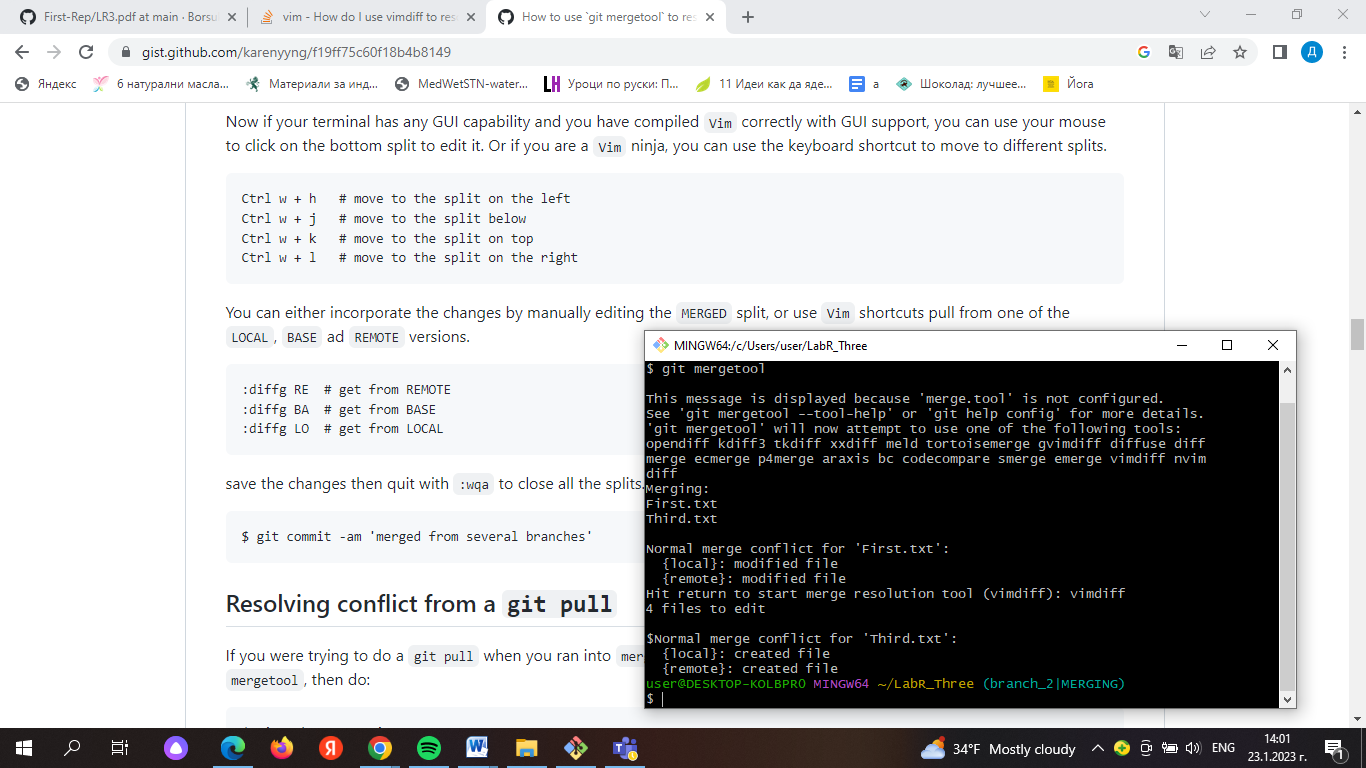


Рисунок 3.16 – Результат работы git mergetool

1. Отправить ветку branch\_1 на GitHub.

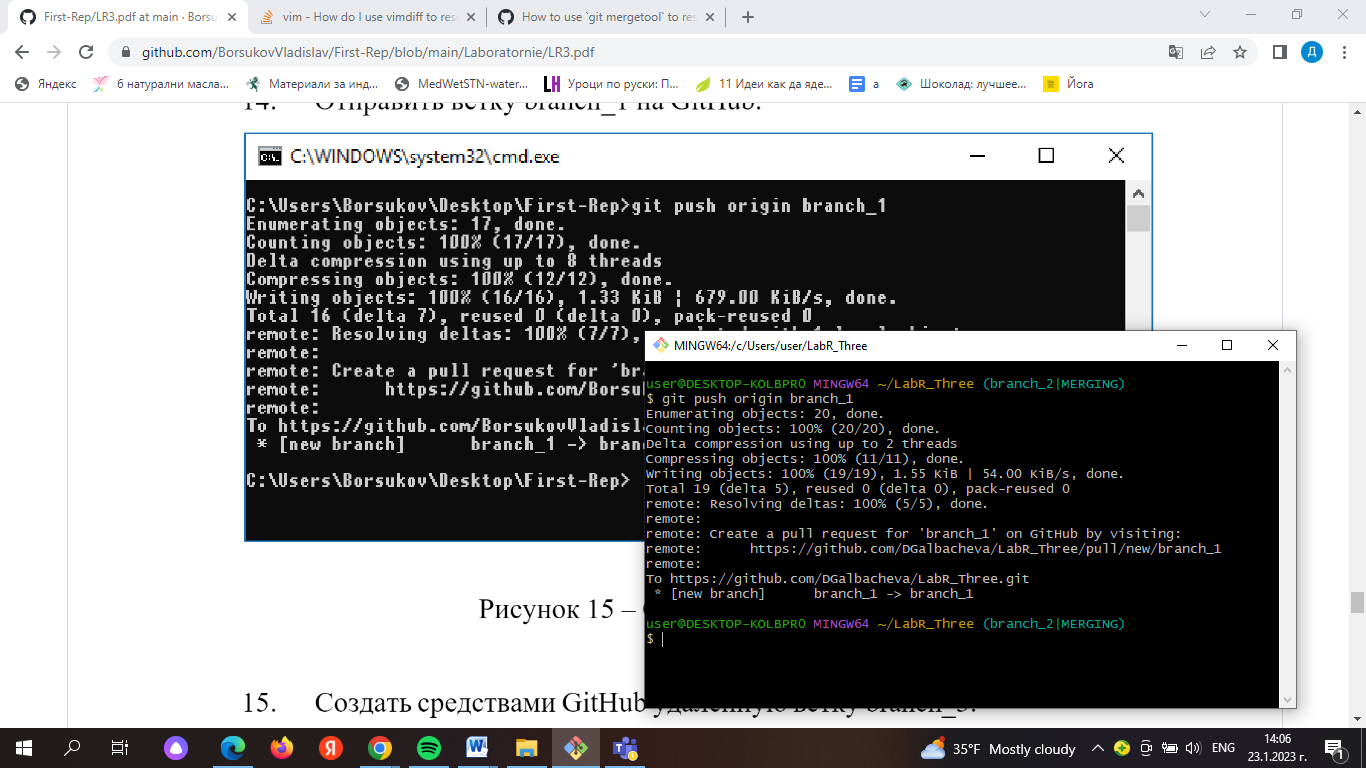


Рисунок 3.17 – Отправка ветки на GitHub

1. Создать средствами GitHub удаленную ветку branch\_3.

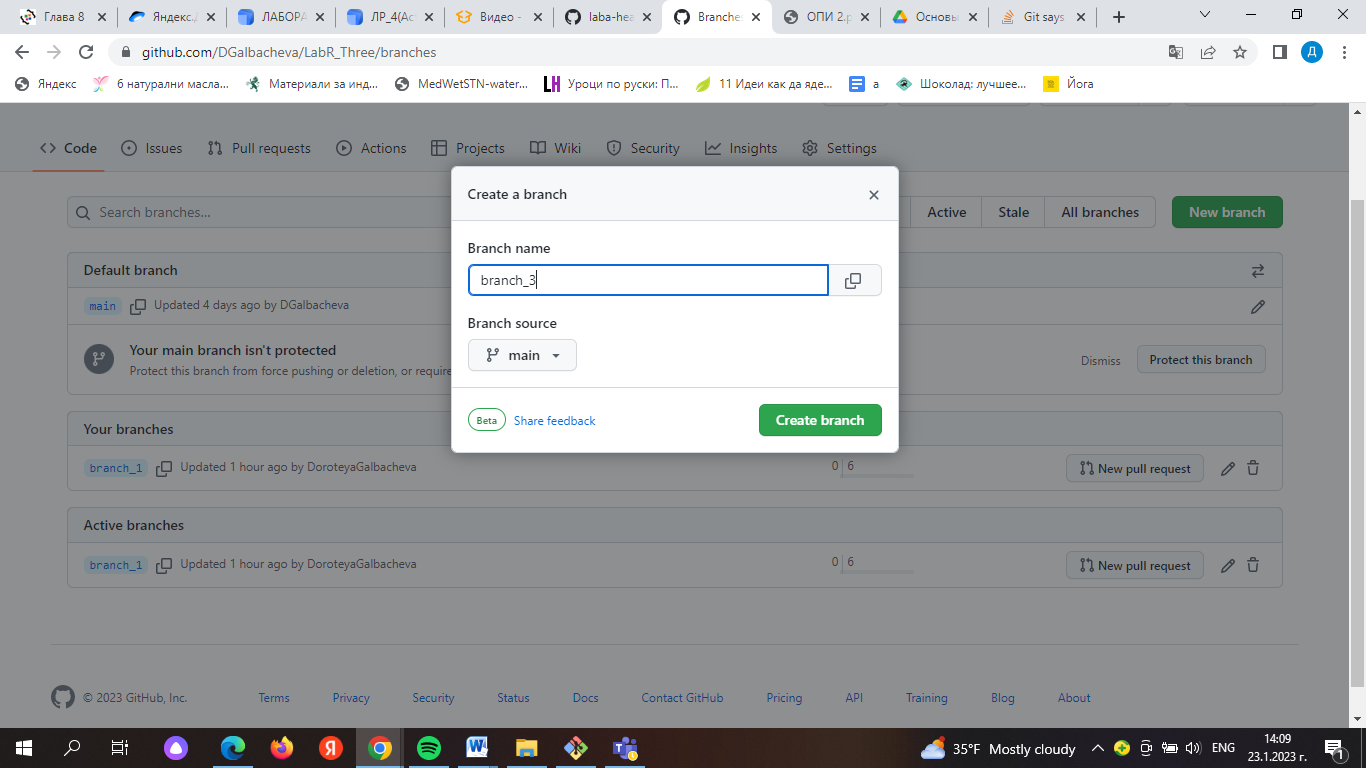


Рисунок 3.18 – Создание удаленной ветки через GitHub

1. Создать в локальном репозитории ветку отслеживания удаленной ветки branch\_3.

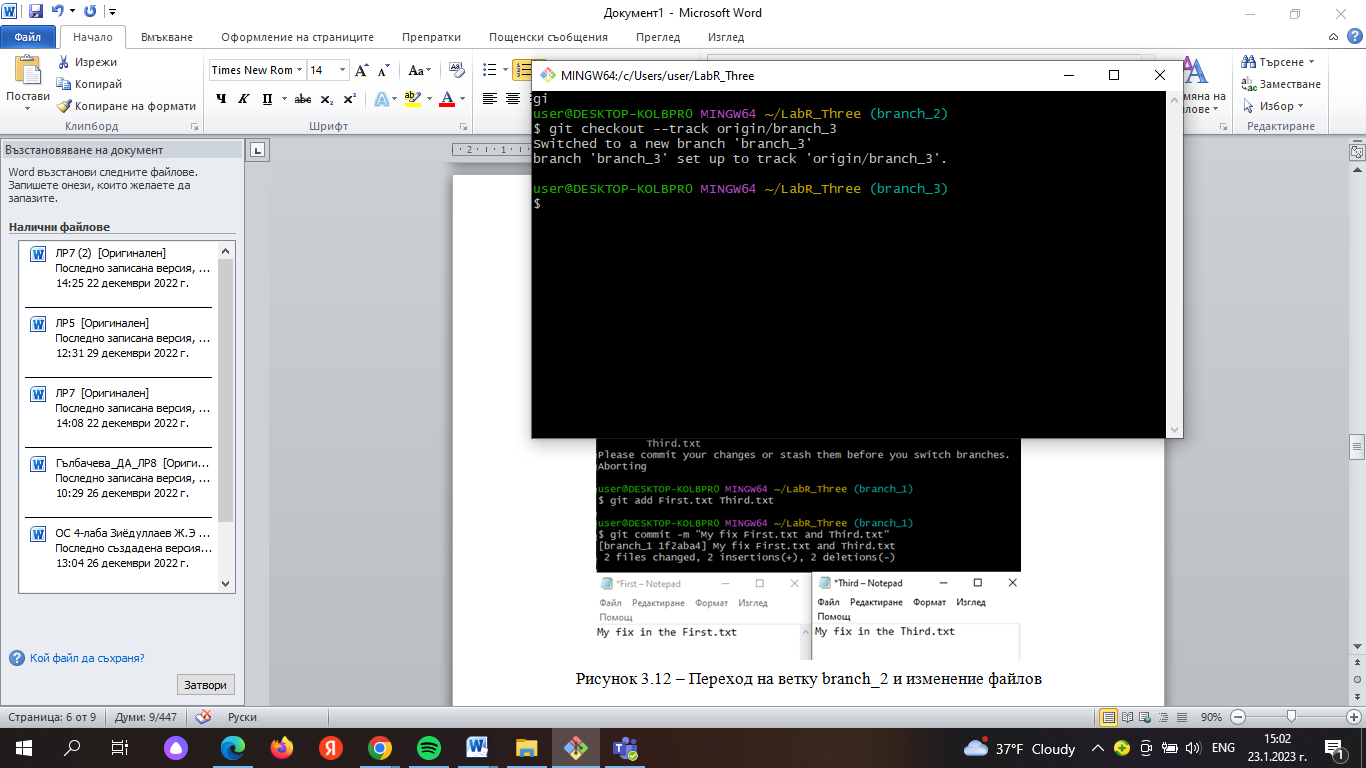


Рисунок 3.19 – Создание веток отслеживания

1. Перейти на ветку branch\_3 и добавить файл файл 2.txt строку "the final fantasy in the 4.txt file".

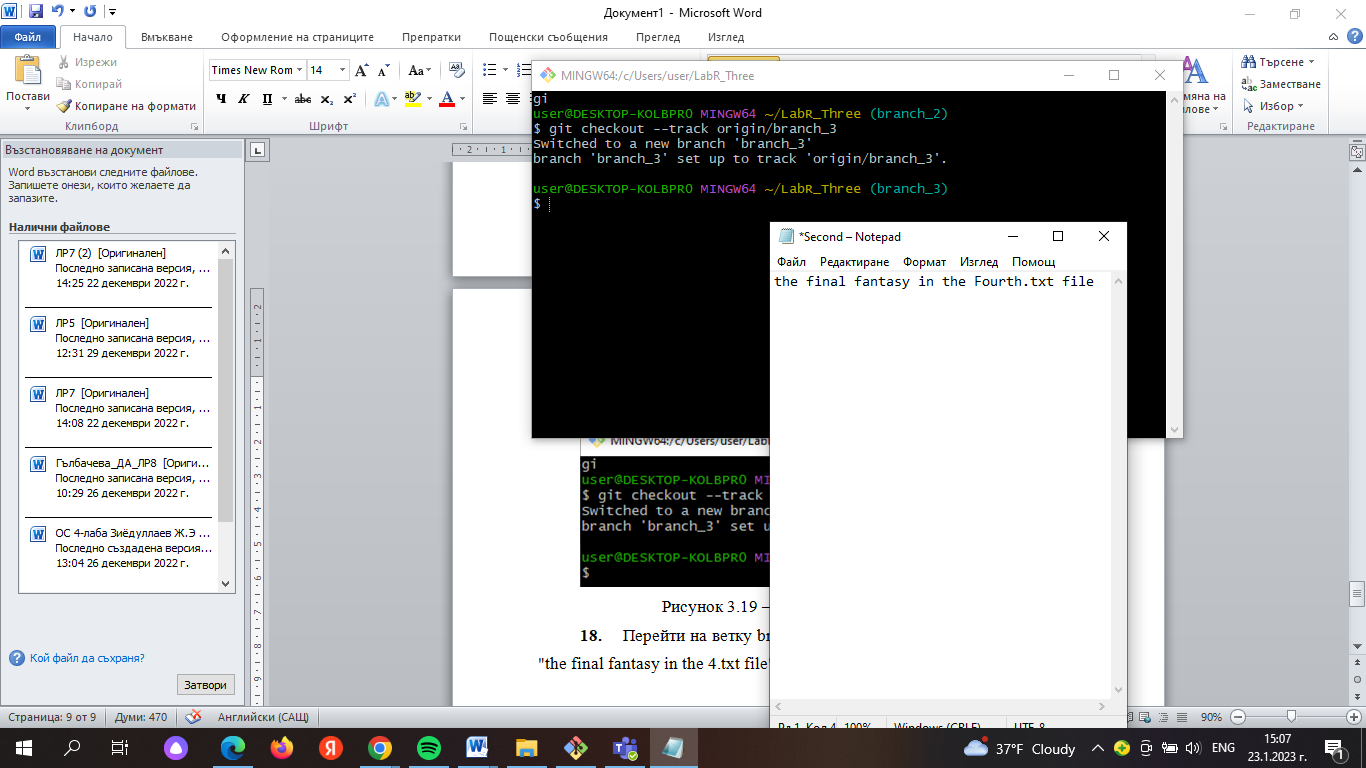


Рисунок 3.20 – Переход на ветку и добавление файла

1. Отправить изменения веток master и branch\_2 на GitHub.

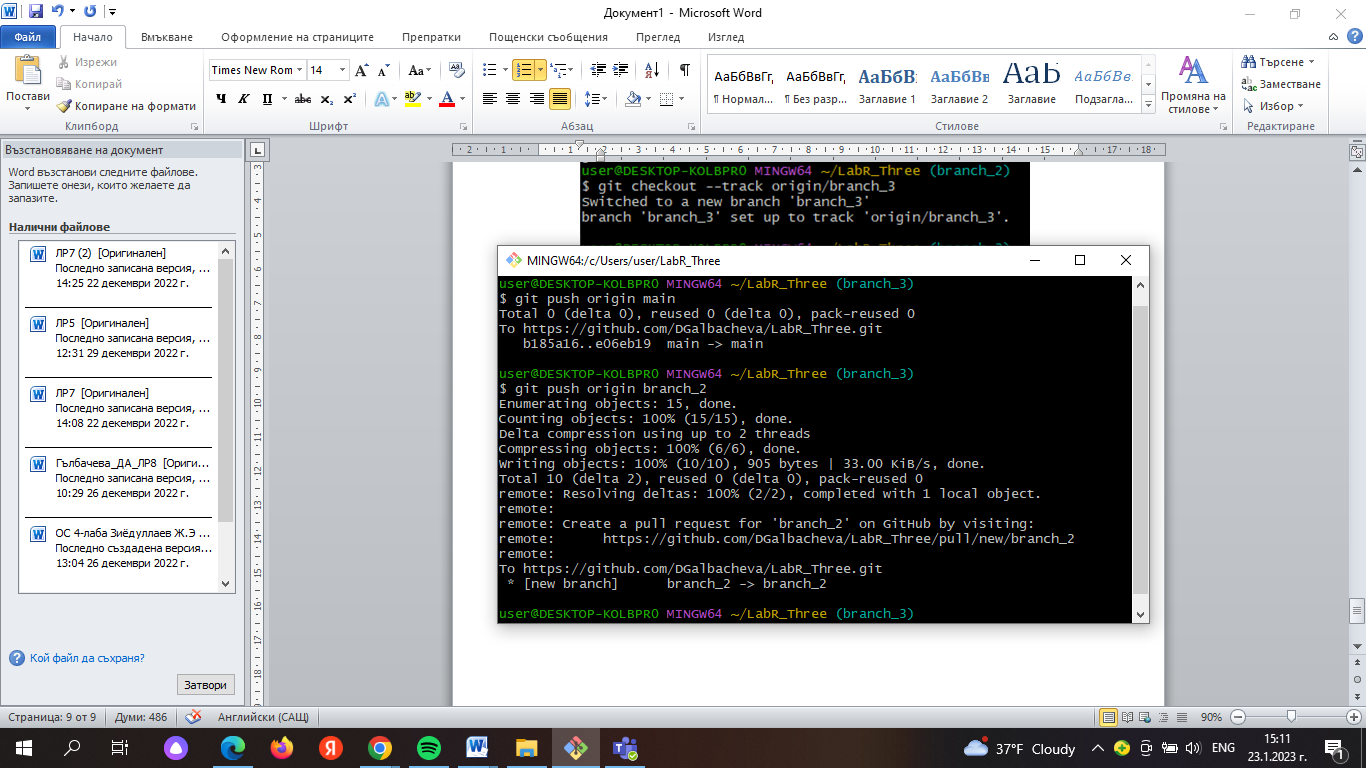


Рисунок 3.21 – Отправление изменения веток на GitHub

**Ответы на вопросы:**

1. Что такое ветка?

Ветка в Git – это просто легковесный подвижный указатель на один из коммитов.

2. Что такое HEAD?

HEAD - это указатель, задача которого ссылаться на определенный коммит в репозитории.

HEAD – это указатель на коммит в вашем репозитории, который станет родителем следующего коммита.

3. Способы создания веток.

Командой git branch

Командой git checkout -b

4. Как узнать текущую ветку?

Командой git branch

5. Как переключаться между ветками?

Git checkout «имя ветки»

6. Что такое удаленная ветка?

Удаленные ветки – это ссылки на состояние веток в удаленных репозиториях.

7. Что такое ветка отслеживания?

Ветки сложения – это ссылки на определенное состояние удаленных веток.

8. Как создать ветку отслеживания?

Для синхронизации ваших изменений с удаленным сервером выполним команду git fetch origin. Далее прописать git checkout –track origin/«имя ветки»

9. Как отправить изменения из локальной ветки в удаленную ветку?

Командой git push «имя ветки»

10. В чем отличие команд git fetch и git pull ?

Git pull – это, по сути, команда git fetch, после которой сразу же следует get merge. Get fetch получает изменения в локальную копию.

11. Как удалить локальную и удаленную ветки?

Удаление удаленной ветки производится при помощи команды: git push origin –delete «имя ветки»

12. Изучить модель ветвления git-flow (использовать материалы статей https://www.atlassian.com/ru/git/tutorials/comparing-workflows/gitflow-workflow, https://habr.com/ru/post/10691 2/). Какие основные типы веток присуствуют в модели git-flow? Как организована работа с ветками в модели git-flow? В чем недостатки git-flow?

Git-flow – альтернативная модель ветвления Git, в которой используются функциональные ветки и несколько основных веток.

Под каждую новую функцию нужно выделить собственную ветку, которую можно отправить в центральный репозиторий для создания резервной копии или совместной работы команды. Ветки feature создаются не на основе main, а на основе develop. Когда работа над функцией завершается, соответствующая ветка сливается с веткой develop. Функции не следует отправлять напрямую в ветку main.

Последовательность действий при работе по модели Gitflow:

1. Из ветки main создается ветка develop.
2. Из ветки develop создается ветка release.
3. Из ветки develop создаются ветки feature.
4. Когда работа над веткой feature завершается, она сливается в ветку develop.
5. Когда работа над веткой release завершается, она сливается с ветками develop и main.
6. Если в ветке main обнаруживается проблема, из main создается ветка hotfix.
7. Когда работа над веткой hotfix заветшается, она сливается с ветками develop и main.

Первая проблема: авторам приходится использовать ветку develop вместо master, поскольку master зарезервирован для кода, который отправляется в продалшен. Существует сложившийся обычай называть рабочую ветвь по умолчанию master, и делать ответвления и слияния с ней. Большинство инстументов по умолчаниз используют это название для основной ветки и по умолцчанию выводят именно еее, и бывает неудобно постоянно переключаться вручную на другую ветку.

Вторая проблема процесса git flow – сложности, возникающие из-за веток для патчей и для релиза. Подобная структура может подойти некоторым организациям, но для абсолютного большинства она просто убийственно излишняя. На сегодняшний день большинство компаний практикуют непрерывное разветывание (continuous delivery), что подразумевает, что основная ветвь по умольчанию может быть задеплоена (deploy). А значит, можно избежать использования веток для релиза и патчей, и всех связанных с нами хлопот, например, обратного слияния из веток релизов.

13. На прошлой лабораторной работе было задание выбрать одно из программных средств с GUI для работы с Git. Необходимо в рамках этого вопроса привести описание инструментов для работы с ветками Git, предоставляемых этим средством.

