

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ**

**ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное**

**учреждение высшего образования**

**«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций «Основы  
ветвления GIT»**

**Отчет по лабораторной работе № 1.3**

**по дисциплине «Основы программной инженерии»**

Выполнил студент группы ПИЖ-б-о-21-1

Кучеренко С. Ю. « » 2022г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 2022г.

Проверил Воронкин Р.А. \_\_\_\_\_

(подпись)

Ставрополь 2022

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

### Ход работы:

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT.

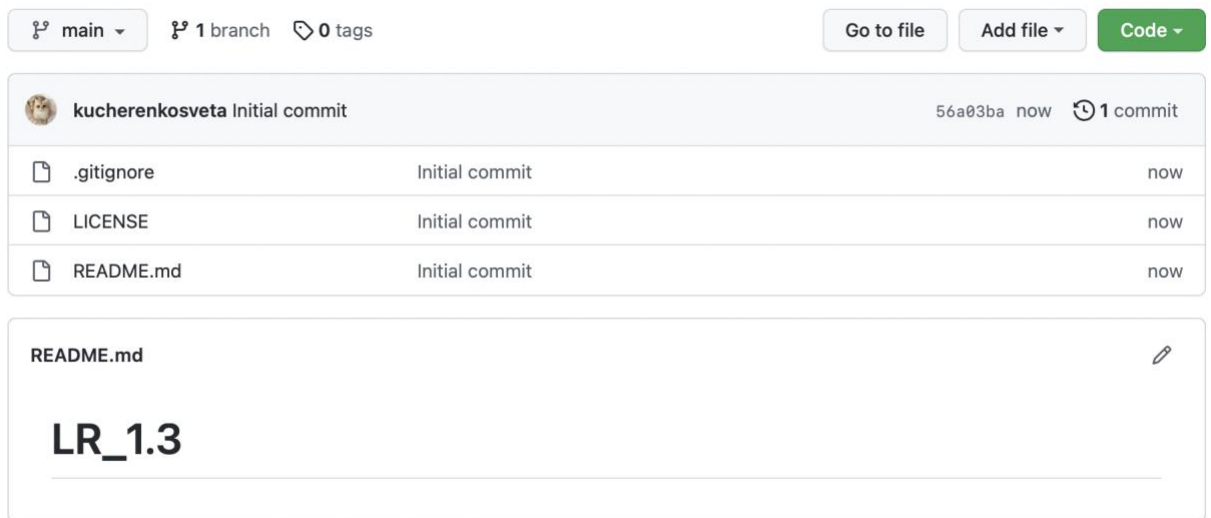


Рисунок 1 – Создание общедоступного репозитория

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

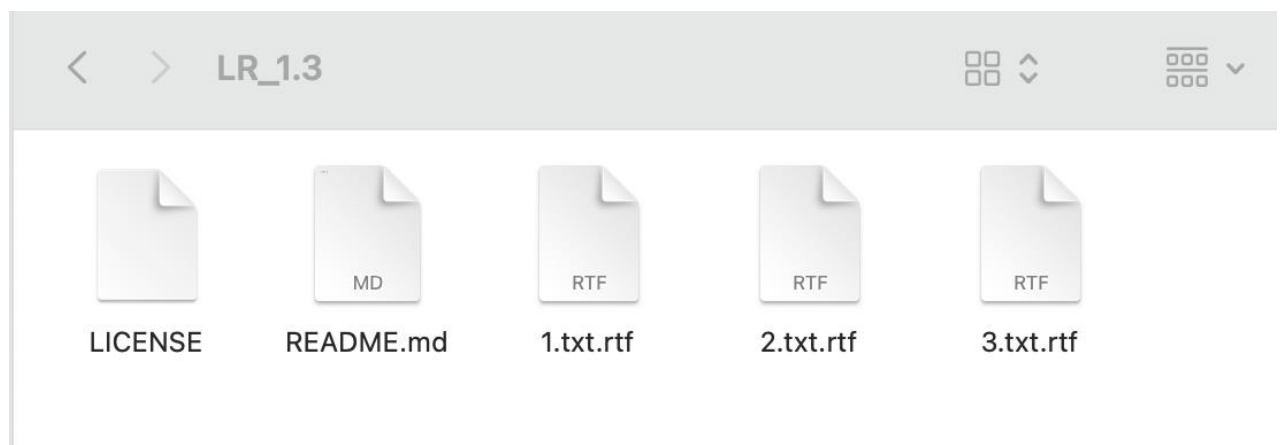


Рисунок 2 – Создание файлов

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

```
[svetik@MacBook-Air-Svetik LR_1.3 % git add .  
[svetik@MacBook-Air-Svetik LR_1.3 % git status  
On branch main  
Your branch is up to date with 'origin/main'.  
  
Changes to be committed:  
  (use "git restore --staged <file>..." to unstage)  
    new file:   1.txt.rtf  
  
[svetik@MacBook-Air-Svetik LR_1.3 % git commit -m "add 1.txt file"  
[main 86669d4] add 1.txt file  
1 file changed, 6 insertions(+)  
create mode 100644 1.txt.rtf  
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 3 – Индексация первого файла и создание коммита

5. Проиндексировать второй и третий файлы.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git add .  
[svetik@MacBook-Air-Svetik LR_1.3 % git add .  
[svetik@MacBook-Air-Svetik LR_1.3 % git status  
On branch main  
Your branch is ahead of 'origin/main' by 1 commit.  
  (use "git push" to publish your local commits)  
  
Changes to be committed:  
  (use "git restore --staged <file>..." to unstage)  
    new file:   .DS_Store  
    new file:   2.txt.rtf  
    new file:   3.txt.rtf  
  
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 4 – Добавление файлов 2 и 3 в индекс

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

```
[svetik@MacBook-Air-Svetik LR_1.3 % git add 2.txt.rtf
[svetik@MacBook-Air-Svetik LR_1.3 % git add 3.txt.rtf
[svetik@MacBook-Air-Svetik LR_1.3 % git commit -m "add 2.txt and 3.txt"
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
      .DS_Store

nothing added to commit but untracked files present (use "git add" to track)
svetik@MacBook-Air-Svetik LR_1.3 %
```

Рисунок 5 – Перезапись коммита

## 7. Создать новую ветку my\_first\_branch.

```
nothing added to commit but untracked files present (use "git add" to track)
[svetik@MacBook-Air-Svetik LR_1.3 % git branch my_first_branch
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
* main
  my_first_branch
[svetik@MacBook-Air-Svetik LR_1.3 %
```

Рисунок 6 – Создание новой ветки

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

```
Last login: Fri Oct 28 14:54:24 on ttys000
[svetik@MacBook-Air-Svetik LR_1.3 % git checkout my_first_branch
A      .DS_Store
Already on 'my_first_branch'
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
  main
* my_first_branch
[svetik@MacBook-Air-Svetik LR_1.3 % git add .
[svetik@MacBook-Air-Svetik LR_1.3 % git status
On branch my_first_branch
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
      new file:   .DS_Store

[svetik@MacBook-Air-Svetik LR_1.3 % git commit
hint: Waiting for your editor to close the file...
```

Рисунок 7 – Создание коммита в новой ветке

9. Вернуться на ветку master.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
* main
  my_first_branch
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 10 – Возвращение на ветку «main»

10. Создать и сразу перейти на ветку new\_branch.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git checkout -b new_branch
Switched to a new branch 'new_branch'
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
main
my_first_branch
* new_branch
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 11 – Одновременное создание и переход на ветку

11. Сделать изменения в файле 1.txt, добавить строчку “new row in the 1.txt file”, закоммитить изменения.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git add .
[svetik@MacBook-Air-Svetik LR_1.3 % git commit "Изменения в 1.txt"
error: pathspec 'Изменения в 1.txt' did not match any file(s) known to git
[svetik@MacBook-Air-Svetik LR_1.3 % git commit -m "Изменения в 1.txt"
[new_branch e53166a] Изменения в 1.txt
1 file changed, 6 deletions(-)
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 12 – Создание изменений в файле 1.txt и коммита

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

```
[svetik@MacBook-Air-Svetik LR_1.3 % git merge my_first_branch
Updating 86669d4..8fcc89f
Fast-forward
 .DS_Store          | Bin 0 -> 6148 bytes
 2.txt.rtf          | 6 ++++++
 3.txt.rtf          | 6 ++++++
 in_branch.txt.rtf | 6 ++++++
 4 files changed, 18 insertions(+)
 create mode 100644 .DS_Store
 create mode 100644 2.txt.rtf
 create mode 100644 3.txt.rtf
 create mode 100644 in_branch.txt.rtf
[svetik@MacBook-Air-Svetik LR_1.3 % git merge new_branch
Already up to date.
svetik@MacBook-Air-Svetik LR_1.3 %
```

Рисунок 13 – Слияние веток main, my\_first\_branch и new\_branch

### 13. Удалить ветки my\_first\_branch и new\_branch.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git branch -d my_first_branch
Deleted branch my_first_branch (was 8fcc89f).
[svetik@MacBook-Air-Svetik LR_1.3 % git branch -d new_branch
Deleted branch new_branch (was e53166a).
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
* main
svetik@MacBook-Air-Svetik LR_1.3 %
```

Рисунок 14 – Удаление веток

### 14. Создать ветки branch\_1 и branch\_2.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git branch branch_1
[svetik@MacBook-Air-Svetik LR_1.3 % git branch branch_2
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
branch_1
branch_2
* main
svetik@MacBook-Air-Svetik LR_1.3 %
```

Рисунок 15 – Создание новых веток



15. 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.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git checkout branch_1
Switched to branch 'branch_1'
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
* branch_1
  branch_2
  main
[svetik@MacBook-Air-Svetik LR_1.3 % git add .
[svetik@MacBook-Air-Svetik LR_1.3 % git status
On branch branch_1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   1.txt.rtf
        modified:   3.txt.rtf

[svetik@MacBook-Air-Svetik LR_1.3 % git commit -m "modified 1.txt and 3.txt"
[branch_1 406d96f] modified 1.txt and 3.txt
 2 files changed, 2 insertions(+), 8 deletions(-)
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 16 – Изменение файлов `1.txt` и `3.txt`, внесение их в коммит на ветке `branch_1`

16. 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.

```

[svetik@MacBook-Air-Svetik LR_1.3 % git checkout branch_2
Already on 'branch_2'
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
  branch_1
* branch_2
  main
[svetik@MacBook-Air-Svetik LR_1.3 % git add .
[svetik@MacBook-Air-Svetik LR_1.3 % git status
On branch branch_2
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   1.txt.rtf
        modified:   3.txt.rtf

[svetik@MacBook-Air-Svetik LR_1.3 % git commit -m "2"
[branch_2 be670fd] 2
 2 files changed, 2 insertions(+), 8 deletions(-)
svetik@MacBook-Air-Svetik LR_1.3 % █

```

Рисунок 17 – Изменение файлов 1.txt и 3.txt, внесение их в коммит на ветке  
branch\_2

#### 17. Слить изменения ветки branch\_2 в ветку branch\_1.

```

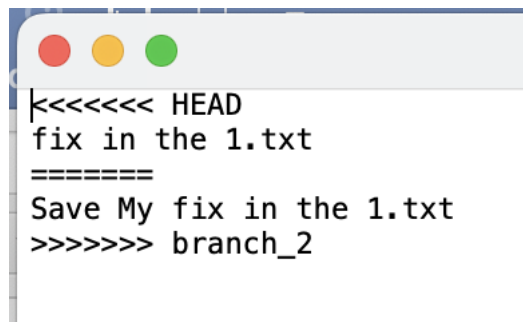
[svetik@MacBook-Air-Svetik LR_1.3 % git checkout branch_1
Switched to branch 'branch_1'
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
* branch_1
  branch_2
  main
[svetik@MacBook-Air-Svetik LR_1.3 % git merge branch_2
Auto-merging 3.txt.rtf
CONFLICT (content): Merge conflict in 3.txt.rtf
Auto-merging 1.txt.rtf
CONFLICT (content): Merge conflict in 1.txt.rtf
Automatic merge failed; fix conflicts and then commit the result.
svetik@MacBook-Air-Svetik LR_1.3 % █

```

Рисунок 18 – Слияние веток branch\_1 и branch\_2

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





```
[svetik@MacBook-Air-Svetik LR_1.3 % git add 1.txt.rtf
[svetik@MacBook-Air-Svetik LR_1.3 % git status
On branch branch_1
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Changes to be committed:
  modified:   1.txt.rtf

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   3.txt.rtf

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  .DS_Store
```

Рисунок 19 – Решение конфликта вручную

```
[svetik@MacBook-Air-Svetik LR_1.3 % git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff tortoisemerge emerge vimdiff nvimdiff
Merging:
3.txt.rtf

Normal merge conflict for '3.txt.rtf':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (opendiff): █
```

Рисунок 19 – Решение конфликта с помощью mergetool

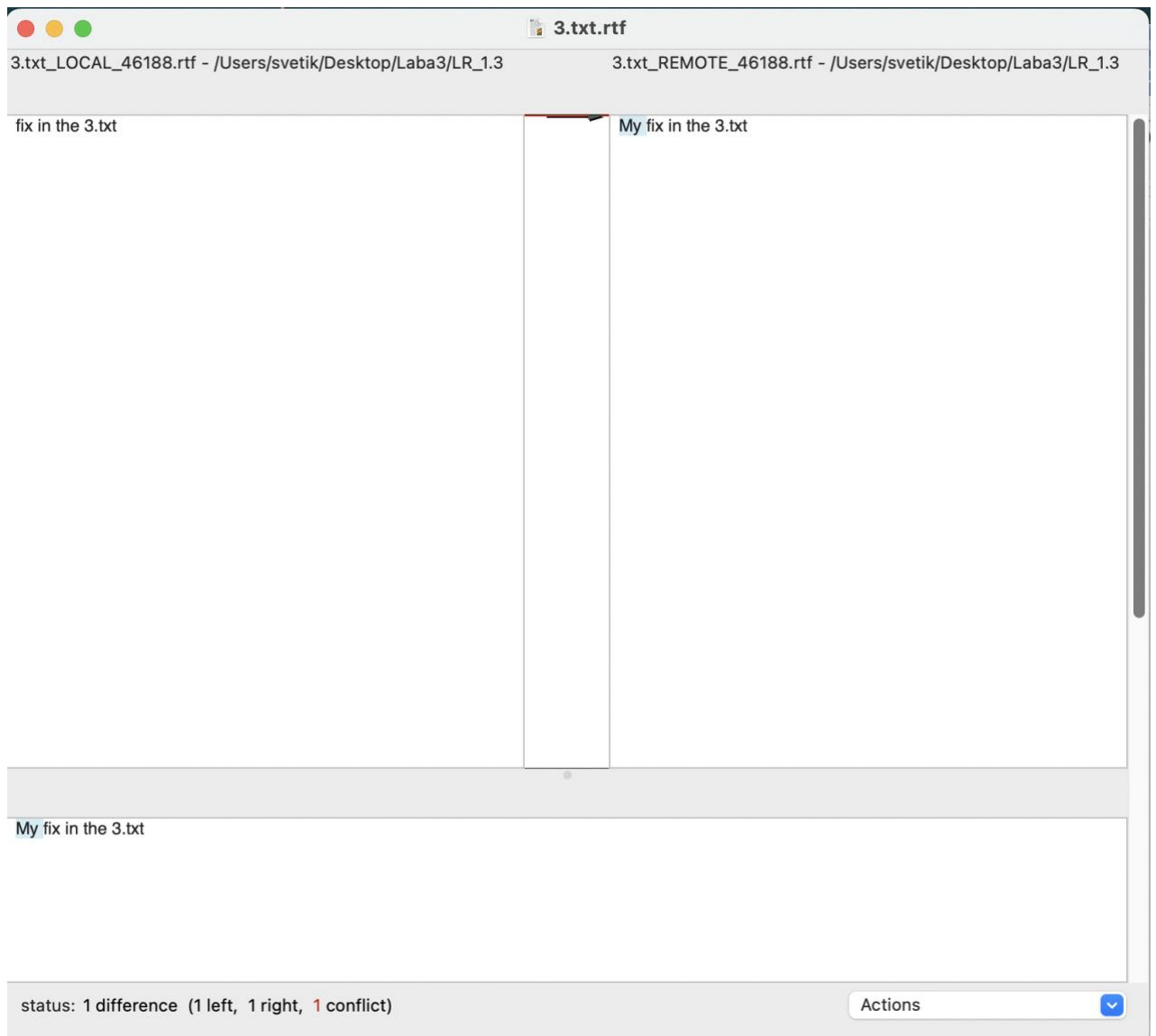


Рисунок 20 – Меню vimdiff

## 19. Отправить ветку branch\_1 на GitHub.

```
[svetik@MacBook-Air-Svetik LR_1.3 % git push --set-upstream origin branch_1
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 8 threads
Compressing objects: 100% (23/23), done.
Writing objects: 100% (28/28), 3.09 KiB | 3.09 MiB/s, done.
Total 28 (delta 12), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (12/12), done.
remote:
remote: Create a pull request for 'branch_1' on GitHub by visiting:
remote:   https://github.com/kucherenkosveta/LR_1.3/pull/new/branch_1
remote:
To https://github.com/kucherenkosveta/LR_1.3.git
 * [new branch]      branch_1 -> branch_1
Branch 'branch_1' set up to track remote branch 'branch_1' from 'origin'.
svetik@MacBook-Air-Svetik LR_1.3 %
```

Рисунок 21 – Отправка изменений на сервер

20. Создать средствами GitHub удаленную ветку branch\_3.

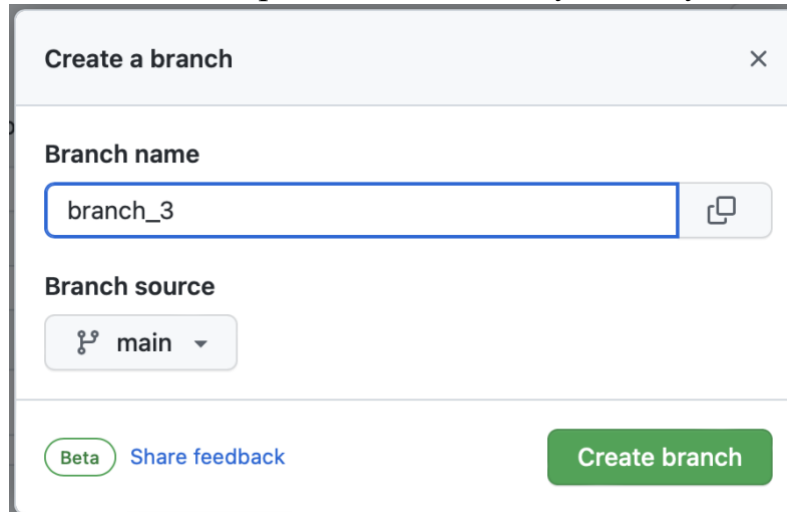


Рисунок 22 – Создание ветки на удаленном репозитории

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

```
[svetik@MacBook-Air-Svetik LR_1.3 % git fetch --all
Fetching origin
From https://github.com/kucherenkosveta/LR_1.3
* [new branch]      branch_3    -> origin/branch_3
[svetik@MacBook-Air-Svetik LR_1.3 % git checkout --track origin/branch_3
Branch 'branch_3' set up to track remote branch 'branch_3' from 'origin'.
Switched to a new branch 'branch_3'
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 23 – Создание ветки отслеживания branch\_3

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

```

[svetik@MacBook-Air-Svetik LR_1.3 % git branch
  branch_1
  branch_2
* branch_3
  main
[svetik@MacBook-Air-Svetik LR_1.3 % git add .
[svetik@MacBook-Air-Svetik LR_1.3 % git status
On branch branch_3
Your branch is up to date with 'origin/branch_3'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   2.txt.rtf

[svetik@MacBook-Air-Svetik LR_1.3 % git add .
[svetik@MacBook-Air-Svetik LR_1.3 % git status
On branch branch_3
Your branch is up to date with 'origin/branch_3'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   2.txt.rtf

[svetik@MacBook-Air-Svetik LR_1.3 % git commit -m "4"
[branch_3 d67017e] 4
 1 file changed, 1 insertion(+)
 create mode 100644 2.txt.rtf
svetik@MacBook-Air-Svetik LR_1.3 % █

```

Рисунок 24 – Переход на ветку branch\_3 и изменение файла

## 23. Выполнить перемещение ветки master на ветку branch\_2.

```

[svetik@MacBook-Air-Svetik LR_1.3 % git checkout branch_2
Switched to branch 'branch_2'
[svetik@MacBook-Air-Svetik LR_1.3 % git branch
  branch_1
* branch_2
  branch_3
  main
[svetik@MacBook-Air-Svetik LR_1.3 % git rebase main
Current branch branch_2 is up to date.
[svetik@MacBook-Air-Svetik LR_1.3 % git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 5 commits.
  (use "git push" to publish your local commits)
[svetik@MacBook-Air-Svetik LR_1.3 % git merge branch_2
Updating 57e6341..be670fd
Fast-forward
 1.txt.rtf | 3 +---
 3.txt.rtf | 7 +-----
 2 files changed, 2 insertions(+), 8 deletions(-)
svetik@MacBook-Air-Svetik LR_1.3 % 4█

```

Рисунок 25 – Перемещение и слияние веток master и branch\_2

## 24. Отправить изменения веток master и branch\_2 на GitHub.

```
svetik@MacBook-Air-Svetik LR_1.3 % git push origin main
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kucherenkosveta/LR_1.3.git
   3e38a7e..be670fd  main -> main
svetik@MacBook-Air-Svetik LR_1.3 % git push origin branch_2
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'branch_2' on GitHub by visiting:
remote:   https://github.com/kucherenkosveta/LR_1.3/pull/new/branch_2
remote:
To https://github.com/kucherenkosveta/LR_1.3.git
 * [new branch]      branch_2 -> branch_2
svetik@MacBook-Air-Svetik LR_1.3 % █
```

Рисунок 26 – Отправка изменений на GitHub.

## Вопросы для защиты работы

### 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.

### 3. Способы создания веток.

С помощью команды `git branch` или `git checkout -b`

### 4. Как узнать текущую ветку?

С помощью команды `git branch`, напротив будет знак «\*»

### 5. Как переключаться между ветками?

С помощью команды `git checkout <имя ветки>`

6. Что такое удаленная ветка?

Удалённые ветки — это ссылки на состояние веток в удаленных репозиториях.

7. Что такое ветка отслеживания?

Ветка отслеживания — это ссылка, расположенная локально, на определённое состояние удалённых веток.

8. Как создать ветку отслеживания?

Для синхронизации `git fetch origin`, а затем `git checkout --track origin/<название_ветки>`.

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

С помощью команды `git push origin <branch >`

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

`Git pull` – это сочетание команд `git fetch` (получение изменений с удаленного репозитория) и `git merge` (объединение веток).

11. Как удалить локальную и удаленную ветки?

Удаление удаленной ветки: `git push origin --delete <branch >`

Удаление локальной: `git branch -d <branch >`



12. Изучить модель ветвления git-flow (использовать материалы статей <https://www.atlassian.com/ru/git/tutorials/comparing-workflows/gitflowworkflow>, <https://habr.com/ru/post/106912/>). Какие основные типы веток присутствуют 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 завершается, она сливается с ветками develop и main.

Первая проблема: авторам приходится использовать ветку develop вместо master, поскольку master зарезервирован для кода, который отправляется в продакшен. Существует сложившийся обычай называть рабочую ветвь по умолчанию master, и делать ответвления и слияния с ней.

Большинство инструментов по умолчанию используют это название для основной ветки и по умолчанию выводят именно ее, 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 развертывание (continuous delivery), что подразумевает, что основная ветвь по умолчанию может быть задеплоена (deploy). А значит, можно избежать использования веток для релиза и патчей, и всех связанных с ними хлопот, например, обратного слияния из веток релизов.