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

_____
(подпись)

Проверил Воронкин Роман Александрович

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: основы ветвления Git.

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

Порядок выполнения работы

1. Создал репозиторий Git.

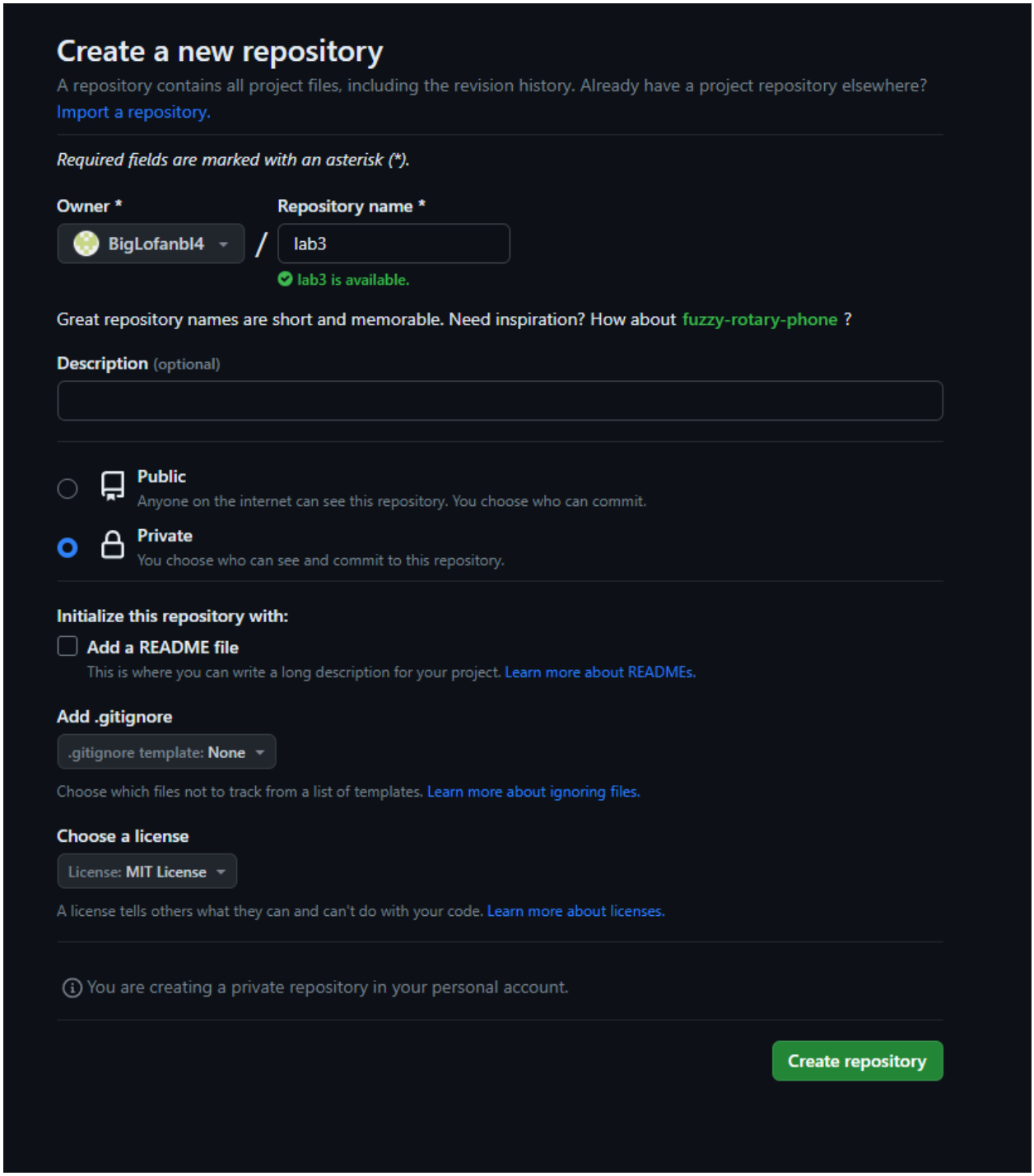

Рисунок 1 – Создание репозитория

2. Создал три файла 1.txt, 2.txt, 3.txt.

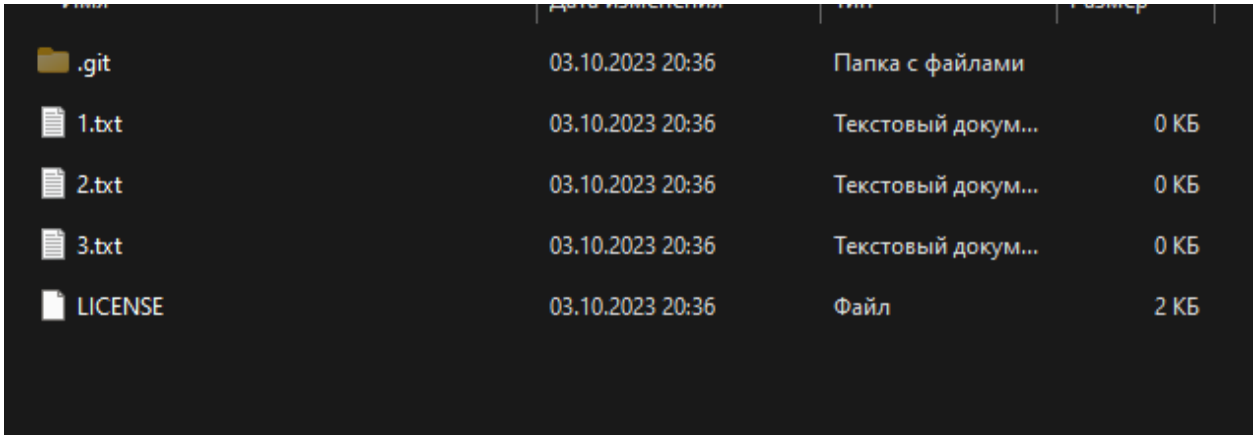

Рисунок 2.1 – Создание требуемых файлов

3. Проиндексировал первый файл и сделал коммит с требуемым комментарием.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git add 1.txt

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git commit -m "add 1.txt file"
[main 12e282e] add 1.txt file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
```

Рисунок 2.2 – Индексация и фиксация 1 файла

4. Проиндексировал второй и третий файлы.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git add 2.txt

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git add 3.txt
```

Рисунок 2.3 – Индексация второго и третьего файла

5. Перезаписал коммит с новым комментарием.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git commit --amend -m "add 2.txt and 3.txt"
[main 0e94eb6] add 2.txt and 3.txt
 Date: Tue Oct 3 20:38:54 2023 +0300
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
 create mode 100644 2.txt
 create mode 100644 3.txt
```

Рисунок 2.4 – Изменение коммита

6. Создал новую ветку «my_first_branch».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git branch my_first_branch
```

Рисунок 2.5 – Создание новой ветки

7. Перешел на созданную ветку, создал новый файл и закоммитил изменения.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git checkout my_first_branch
Switched to branch 'my_first_branch'

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git add in_branch.txt

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git commit -m "add in_branch.txt in new branch"
[my_first_branch 6cdc16d] add in_branch.txt in new branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt
```

Рисунок 3 – Коммит файла в новой ветке

8. Вернулся в ветку «main».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>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)
```

Рисунок 3.1 – Возвращение на ветку «main»

9. Создал и сразу перешел на новую ветку «new_branch».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git checkout -b new_branch
Switched to a new branch 'new_branch'
```

Рисунок 3.2 – Создание и переход на новую ветку

10. Сделал изменения в файле 1.txt и закоммитил изменения.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git add 1.txt

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git commit -m "add changes in 1.txt"
[new_branch b71c2ac] add changes in 1.txt
 1 file changed, 1 insertion(+)
```

Рисунок 3.3 – Коммит изменений

11. Перешел на ветку «main», слил ветки «main» и «my_first_branch», после слил ветки «main» и «new_branch».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>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)

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git merge my_first_branch
Updating 0e94eb6..6cdc16d
Fast-forward
 in_branch.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git merge new_branch
Merge made by the 'ort' strategy.
 1.txt | 1 +
 1 file changed, 1 insertion(+)
```

Рисунок 3.4 – Слияние веток «my_first_branch» и «new_branch» с «main»

12. Удалил ветки «my_first_branch» и «new_branch».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git branch -d my_first_branch
Deleted branch my_first_branch (was 6cdc16d).

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git branch -d new_branch
Deleted branch new_branch (was b71c2ac).
```

Рисунок 3.5 – Удаление веток «my_first_branch» и «new_branch»

13. Создал ветки «branch_1» и «branch_2».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git branch branch_1

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git branch branch_2
```

Рисунок 4 – Создание веток «branch_1» и «branch_2»

14. Перешел на ветку «branch_1» и изменил файлы 1.txt и 3.txt.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git checkout branch_1
Switched to branch 'branch_1'

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git add .

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git commit -m "add fixes in 1.txt and 3.txt"
[branch_1 bfbd4ec] add fixes in 1.txt and 3.txt
 2 files changed, 2 insertions(+), 1 deletion(-)
```

Рисунок 4.1 – Выполнение пункта 15

15. Перешел в ветку «branch_2», изменил файлы 1.txt и 3.txt.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git checkout branch_2
Switched to branch 'branch_2'

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git add .

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git commit -m "add myFixes in 1.txt and 3.txt"
[branch_2 db5fc6f] add myFixes in 1.txt and 3.txt
 2 files changed, 2 insertions(+), 1 deletion(-)
```

Рисунок 4.2 – Выполнение пункта 16

16. Слил изменения ветки «branch_2» в ветку «branch_1».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git checkout branch_1
Switched to branch 'branch_1'

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git merge branch_2
Auto-merging 1.txt
CONFLICT (content): Merge conflict in 1.txt
Auto-merging 3.txt
CONFLICT (content): Merge conflict in 3.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Рисунок 4.3 – Слияние «branch_1» с «branch_2»

17. Решил конфликты: для файла 1.txt в ручном режиме, для файла 3.txt использую команду «git mergetool».

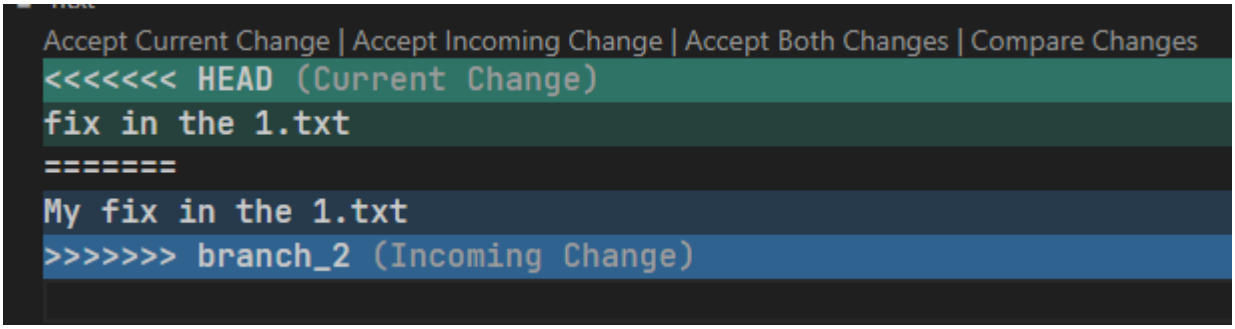

Рисунок 4.4 – Решение конфликта в файле 1.txt

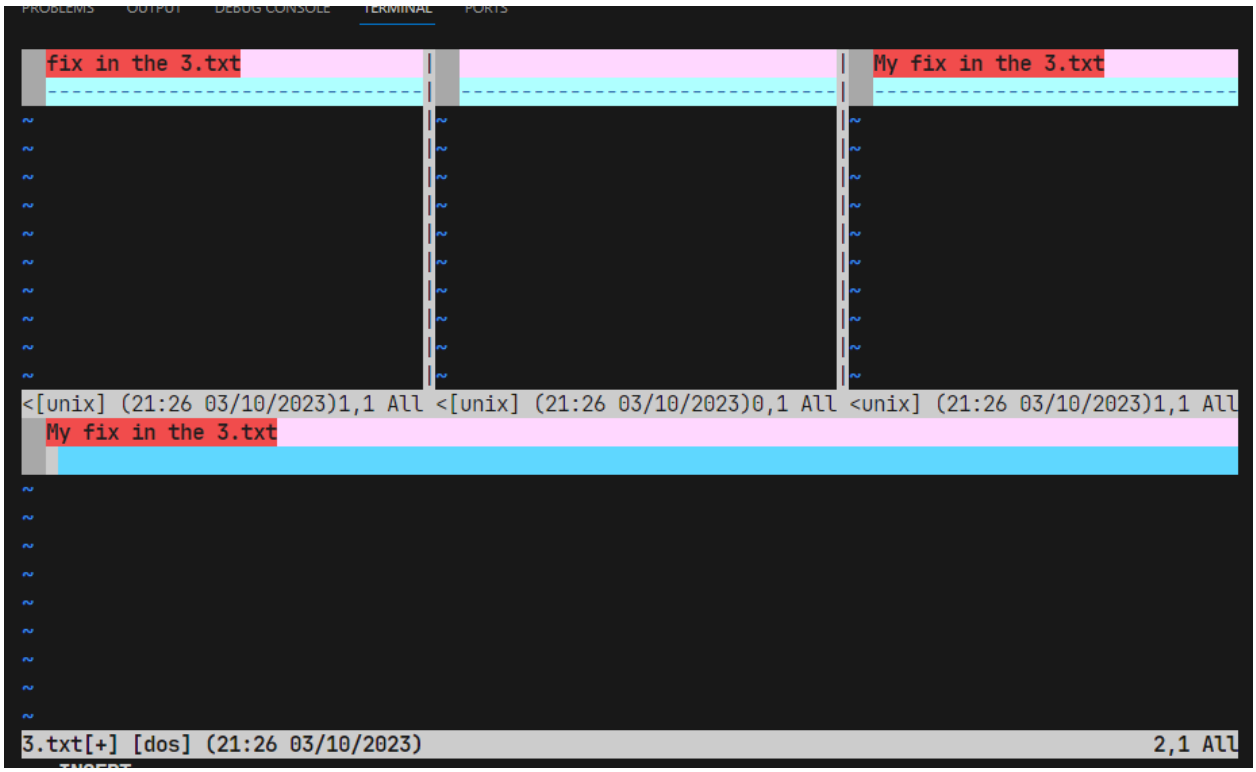

Рисунок 4.5 – Решение конфликта в файле 3.txt

18. Отправил ветку «branch_1» на GitHub.

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>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% (14/14), done.
Writing objects: 100% (22/22), 1.85 KiB | 949.00 KiB/s, done.
Total 22 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (6/6), done.
remote:
remote: Create a pull request for 'branch_1' on GitHub by visiting:
remote:      https://github.com/BigLofanbl4/lab3/pull/new/branch_1
remote:
To https://github.com/BigLofanbl4/lab3.git
 * [new branch]      branch_1 -> branch_1
```

Рисунок 5 – Отправка ветки «branch_1» на удаленный репозиторий

19. Создал средствами GitHub удаленную ветку «branch_3».

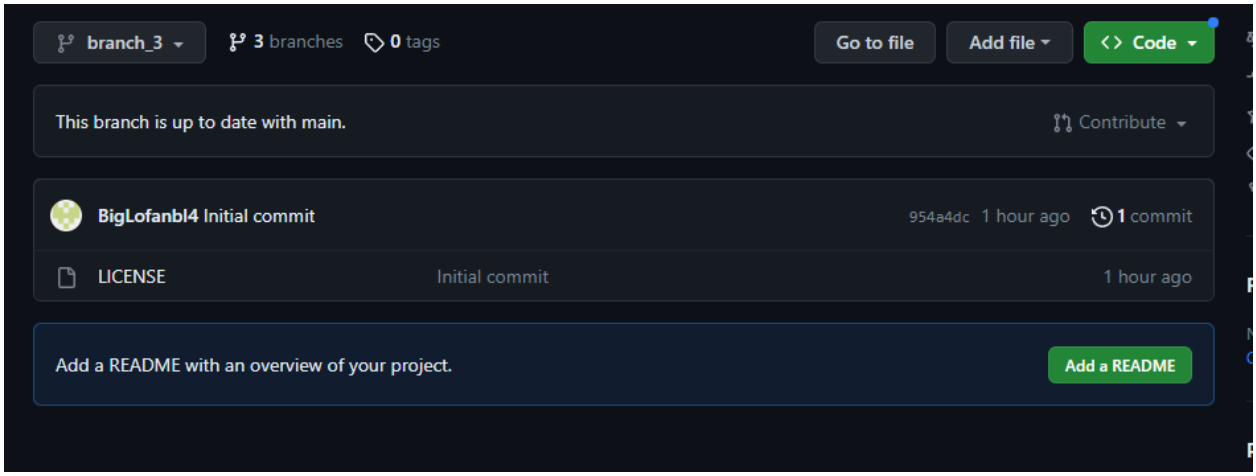

Рисунок 5.1 – Ветка «branch_3»

20. Создал в локальном репозитории ветку отслеживания удаленной ветки «branch_3».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git checkout -b branch_3 origin/branch_3
Switched to a new branch 'branch_3'
branch 'branch_3' set up to track 'origin/branch_3'.
```

Рисунок 5.2 – Создание ветки отслеживания

21. Выполнил перемещение ветки «main» на ветку «branch_2».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 4 commits.
  (use "git push" to publish your local commits)

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git merge branch_2
Updating aa6b106..db5fc6f
Fast-forward
 1.txt | 2 +-
 3.txt | 1 +
 2 files changed, 2 insertions(+), 1 deletion(-)
```

Рисунок 5.3 – Перемещение ветки «main» на ветку «branch_2»

22. Отправил изменения веток «main» и «branch_2».

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>git push origin main
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/BigLofanbl4/lab3.git
   954a4dc..db5fc6f  main -> main

C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\3\lab3>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/BigLofanbl4/lab3/pull/new/branch_2
remote:
To https://github.com/BigLofanbl4/lab3.git
 * [new branch]      branch_2 -> branch_2
```

Рисунок 5.3 – Отправка изменений

Ответы на контрольные вопросы

1. По сути ветка в Git - это простой перемещаемый указатель, который указывает на определенный коммит. Каждая ветка представляет собой отдельную ветвь разработки, которая может изменяться независимо от других веток.

2. HEAD – это указатель на определенный коммит в репозитории.

3. Создать новую ветку можно с помощью команды «git branch <branchName>» или «git checkout -b <branchName>».

4. Узнать текущую ветку в Git можно, использовав команду «git branch», которая предоставит весь список веток в репозитории, текущая ветка будет отмечена символом *.

5. Чтобы переключаться между ветками необходимо воспользоваться командной «git checkout <branchName>».

6. Удаленные ветки - это ветки, которые существуют на удаленном репозитории, а не локально на компьютере.

7. Ветки отслеживания - это ссылки на определенное состояние удаленных веток. Их следует представлять как закладки для напоминания, где находились ветки в удаленных репозиториях во время последнего подключения к ним.

8. С помощью команды «git checkout <branchName> <remote>/<branch>».

9. С помощью команды «git push <remote> <branchName>».

10. Команда «git fetch» получает с сервера все изменения, но не будет делать автоматическое слияние. Git pull же наоборот делает автоматическое слияние.

11. Для удаления локальной ветки необходимо воспользоваться командой «git branch -d <branchName>», для удаления удаленной ветки необходимо воспользоваться командой «git push <remote> --delete <branchName».

12. Модель ветвления git-flow предлагает стратегию организации работы с ветками в Git. Основные типы веток, предлагаемые в этой модели, включают:

— Master (главная ветка): Эта ветка представляет стабильную версию кода, которая находится в производстве.

— Develop (ветка разработки):

— Feature (ветки функциональностей): Каждая новая функциональность или изменение разрабатывается в отдельной ветке.

— Release (ветки релизов): Когда достигается стабильное состояние разработки, создаётся ветка release для подготовки к новому выпуску. На этой ветке могут выполняться последние исправления ошибок, обновления документации и другие действия, связанные с подготовкой релиза. По завершении работы над релизом ветка сливается обратно в Master и Develop, а также помечается тегом с номером версии.

— Hotfix (ветки исправлений): Если в производстве обнаруживается критическая ошибка, требующая немедленного исправления, создаётся ветка hotfix. На этой ветке производится исправление проблемы, а затем она быстро вливается обратно в Master и Develop.

В работе с ветками по модели git-flow важно придерживаться следующих правил:

— Вся разработка ведется в отдельных ветках, которые в последствии вливаются в главные ветки.

— Выделение отдельных веток для каждой функциональности позволяет параллельно разрабатывать несколько фичей.

— Релизы подготавливаются в отдельных ветках, что облегчает процесс их тестирования и исправления ошибок.

— Исправление критических проблем выполняется в отдельных ветках hotfix.

Недостатки модели git-flow включают следующие аспекты:

— Ветвление по модели git-flow может вести к большому количеству веток, особенно при разработке нескольких функциональностей одновременно, что может стать сложным для управления и поддержания.

— Увеличение сложности и объема работы с ветками может стать проблемой для небольших команд или проектов.

— Некоторые ветки, такие как release и hotfix, могут иметь короткий срок существования, что может быть слишком ограничивающим, особенно если необходимо внести несколько изменений внутри них.

— Git-flow предполагает линейное развитие проекта и не всегда хорошо подходит для проектов, требующих более гибкого подхода к разработке.

13. В GitHub Desktop есть несколько инструментов, которые вы можете использовать для работы с ветками:

— Переключение между ветками - в верхней части окна GitHub Desktop есть выпадающий список, в котором отображаются доступные ветки. Выберите ветку, на которую хотите переключиться, чтобы обновить свою локальную рабочую копию репозитория.

— Создание новой ветки - на панели инструментов в GitHub Desktop есть кнопка "New Branch" или "Create Branch". Нажмите на нее, введите название новой ветки и выберите ветку, от которой вы хотите создать новую

ветку. После этого ветка будет создана и вы будете автоматически переключены на нее.

— Слияние веток - чтобы слить одну ветку с другой, щелкните правой кнопкой мыши на ветке, в которую вы хотите слить другую ветку, и выберите "Merge Into Current Branch" или "Merge Branch". Выберите ветку, которую вы хотите слить, и выполните слияние.

— Удаление ветки - щелкните правой кнопкой мыши на ветке, которую вы хотите удалить, и выберите "Delete Branch". Обратите внимание, что вы можете удалить только ветки, которые не являются текущей рабочей веткой или веткой, от которой она ветвилась.

— Отслеживание удаленных веток - GitHub Desktop автоматически отслеживает удаленные ветки и отображает их в списке веток. Если вы хотите отслеживать удаленную ветку локально, нажмите на соответствующую кнопку "Fetch origin" или "Pull origin", чтобы получить обновления удаленных веток.