

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ  
АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО  
ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчёт о лабораторной работе №3 по дисциплине основы программной  
инженерии**

Выполнила:

Нестеренко Тамара Антоновна,  
2 курс, группа ПИЖ-б-о-20-1,

Проверил:

Доцент кафедры инфокоммуникаций,  
Воронкин Р.А.

Ставрополь, 2021 г.

## ВЫПОЛНЕНИЕ

### 1. Практическая часть




 1.txt	23.09.2021 21:29	Текстовый докум...	0 КБ
 2.txt	23.09.2021 21:29	Текстовый докум...	0 КБ
 3.txt	23.09.2021 21:29	Текстовый докум...	0 КБ

Рисунок 1 – Результат создания трёх файлов

```
C:\Users\тома нестеренко\helloworld>git status
On branch main
Your branch is up to date with 'origin/main'.

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

nothing added to commit but untracked files present (use "git add" to track)
C:\Users\тома нестеренко\helloworld>git add 1.txt
C:\Users\тома нестеренко\helloworld>git commit -m"add 1.txt file@"
[main f0ac527] add 1.txt file@
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 1.txt
C:\Users\тома нестеренко\helloworld>git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

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

nothing added to commit but untracked files present (use "git add" to track)
```

Рисунок 2 – Результат индексации файла 1.txt и создания коммита

```
C:\Users\тома нестеренко\helloworld>git add 2.txt
C:\Users\тома нестеренко\helloworld>git add 3.txt
C:\Users\тома нестеренко\helloworld>git commit -m"add new file@
[main 36147ea] add new file@
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 2.txt
 create mode 100644 3.txt
C:\Users\тома нестеренко\helloworld>git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Рисунок 3 – Результат индексации файла 2.txt, 3.txt и создания коммита

```
C:\Users\тома нестеренко\helloworld>git commit --amend -m "add 2.txt and 3.txt"
[main a4c5a21] add 2.txt and 3.txt
Date: Thu Sep 23 21:34:55 2021 +0300
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 2.txt
 create mode 100644 3.txt
```

Рисунок 4 – Результат изменения последнего коммита

```
C:\Users\тома нестеренко\helloworld>git branch my_first_branch
```

Рисунок 5 – Результат добавления новой ветки

```
C:\Users\тома нестеренко\helloworld>git branch
* main
  my_first_branch
C:\Users\тома нестеренко\helloworld>git checkout my_first_branch
Switched to branch 'my_first_branch'
C:\Users\тома нестеренко\helloworld>git branch
  main
* my_first_branch
```

Рисунок 6 – Результат перехода на новую ветку

```
C:\Users\тома нестеренко\helloworld>git add in_branch.txt

C:\Users\тома нестеренко\helloworld>git commit -m "add new file in_branch"
[my_first_branch c1137fb] add new file in_branch
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 in_branch.txt

C:\Users\тома нестеренко\helloworld>git status
on branch my_first_branch
nothing to commit, working tree clean
```

Рисунок 7 – Результат добавления файла на ветку my\_first\_branch

```
C:\Users\тома нестеренко\helloworld>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

C:\Users\тома нестеренко\helloworld>git branch
* main
  my_first_branch
```

Рисунок 8 – Результат перехода на ветку main

```
C:\Users\тома нестеренко\helloworld>git checkout -b new_branch
Switched to a new branch 'new_branch'
```

Рисунок 9 – Результат перехода на ветку main



Рисунок 10 – Результат перехода на ветку main

```
C:\Users\тома нестеренко\helloworld>git add 1.txt  
C:\Users\тома нестеренко\helloworld>git commit -m"Edited file 1.txt  
[new_branch 9aadafb] Edited file 1.txt@  
1 file changed, 1 insertion(+)
```

Рисунок 11 – Результат добавления файла и создания коммита на ветке new\_branch

```
C:\Users\тома нестеренко\helloworld>git checkout main  
Switched to branch 'main'  
Your branch is ahead of 'origin/main' by 2 commits.  
(use "git push" to publish your local commits)  
  
C:\Users\тома нестеренко\helloworld>git merge my_first_branch  
Updating a4c5a21..c1137fb  
Fast-forward  
 in_branch.txt | 0  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 in_branch.txt  
  
C:\Users\тома нестеренко\helloworld>git merge new_branch  
Merge made by the 'recursive' strategy.  
1.txt | 1 +  
1 file changed, 1 insertion(+)
```

Рисунок 12 – Результат слияния веток

```
C:\Users\тома нестеренко\helloworld>git branch -d my_first_branch  
Deleted branch my_first_branch (was c1137fb).  
  
C:\Users\тома нестеренко\helloworld>git branch -d new_branch  
Deleted branch new_branch (was a7733aa).
```

Рисунок 13 – Результат удаления веток

```
C:\Users\тома нестеренко\helloworld>git branch branch_1  
C:\Users\тома нестеренко\helloworld>git branch branch_2  
C:\Users\тома нестеренко\helloworld>git branch  
branch_1  
branch_2  
* main
```

Рисунок 14 – Результат создания двух веток

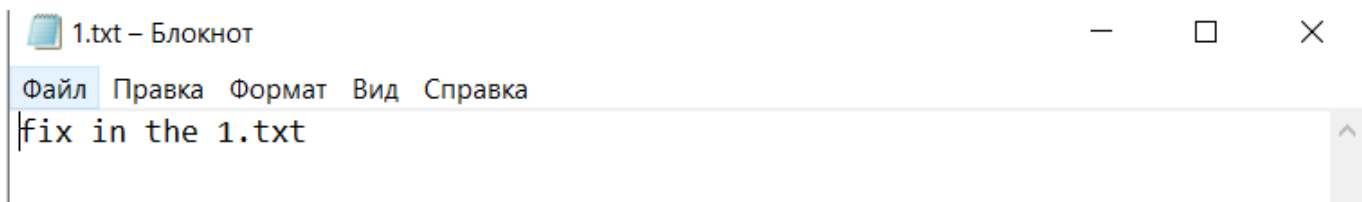


Рисунок 15 – Результат изменения файла 1.txt



Рисунок 16 – Результат изменения файла 3.txt

```
C:\Users\тома нестеренко\helloworld>git add 1.txt
C:\Users\тома нестеренко\helloworld>git add 2.txt
C:\Users\тома нестеренко\helloworld>git commit -m "Edited fike 1.txt and 2.txt"
[branch_1 79c7d31] Edited fike 1.txt and 2.txt
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
C:\Users\тома нестеренко\helloworld>git add 3.txt
C:\Users\тома нестеренко\helloworld>git commit --amend -m"Edited file 1.txt"
[branch_1 700a384] Edited file 1.txt
Date: Thu Sep 23 22:16:07 2021 +0300
2 files changed, 2 insertions(+), 1 deletion(-)
C:\Users\тома нестеренко\helloworld>git commit -m "Edited file 3.txt"
On branch branch_1
nothing to commit, working tree clean
C:\Users\тома нестеренко\helloworld>git status
On branch branch_1
nothing to commit, working tree clean
```

Рисунок 17 – Результат создания коммита для изменений

```
C:\Users\тома нестеренко\helloworld>git checkout branch_2
Switched to branch 'branch_2'
```

Рисунок 18 – Результат перехода на ветку branch\_2



Рисунок 19 – Результат изменения файла 1.txt

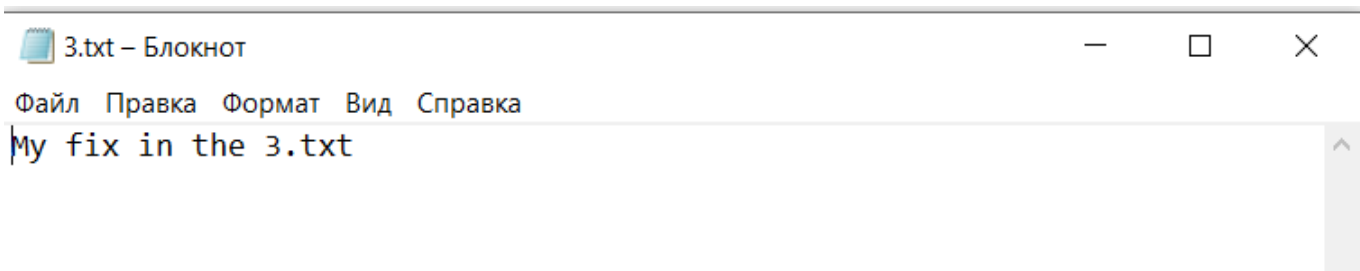


Рисунок 20 – Результат изменения файла 3.txt

```
C:\Users\тома нестеренко\helloworld>git add 1.txt
C:\Users\тома нестеренко\helloworld>git add 3.txt
C:\Users\тома нестеренко\helloworld>git commit -m "Edited file 1.txt and 3.txt"
[branch_2 642d057] Edited file 1.txt and 3.txt
 2 files changed, 2 insertions(+), 1 deletion(-)
C:\Users\тома нестеренко\helloworld>git status
on branch branch_2
nothing to commit, working tree clean
```

Рисунок 21 – Результат создания изменений в файлах 1.txt и 3.txt, создание коммита

```
C:\Users\тома нестеренко\helloworld>git merge branch_1
Auto-merging 3.txt
CONFLICT (content): Merge conflict in 3.txt
Auto-merging 1.txt
CONFLICT (content): Merge conflict in 1.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Рисунок 22 – Результат слияния веток branch\_2 и branch\_1

```

C:\Users\тома нестеренко\helloworld>git merge branch_1
Auto-merging 3.txt
CONFLICT (content): Merge conflict in 3.txt
Auto-merging 1.txt
CONFLICT (content): Merge conflict in 1.txt
Automatic merge failed; fix conflicts and then commit the result.

C:\Users\тома нестеренко\helloworld>git status
On branch branch_2
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   1.txt
        both modified:   3.txt

no changes added to commit (use "git add" and/or "git commit -a")

C:\Users\тома нестеренко\helloworld>git add 1.txt
C:\Users\тома нестеренко\helloworld>git add 3.txt
C:\Users\тома нестеренко\helloworld>git status
On branch branch_2
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
        modified:   1.txt
        modified:   3.txt

```

Рисунок 23 – Результат слияния веток branch\_2 и branch\_1, решение конфликтов

```

C:\Users\тома нестеренко\helloworld>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:
tortoisemerge emerge vimdiff nvimdiff
No files need merging

```

Рисунок 24 – Результат проверки конфликтов с помощью команды «git mergetool»

```

C:\Users\тома нестеренко\helloworld>git commit -m "Add file 1.txt and 3.txt"
[branch_2 24d46a4] Add file 1.txt and 3.txt

C:\Users\тома нестеренко\helloworld>git merge branch_1
Already up to date.

```

Рисунок 25 – Результат создания коммита о новых изменениях и слияние двух веток: branch\_1, branch\_2.



```

C:\Users\тома нестеренко\helloworld>git push origin branch_1
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 4 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (16/16), 1.40 KiB | 477.00 KiB/s, done.
Total 16 (delta 6), reused 1 (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/tamaranesterenko/helloworld/pull/new/branch_1
remote:
To https://github.com/tamaranesterenko/helloworld.git
 * [new branch]      branch_1 -> branch_1

```

Рисунок 26 – Результат отправки ветки на удалённый сервер

The screenshot displays the GitHub interface for the repository 'tamaranesterenko/helloworld'. The 'branch\_1' branch is currently selected. A dropdown menu for switching branches is open, showing 'branch\_3' as the selected option. The repository statistics indicate 2 branches and 3 tags. The commit history shows a recent commit 'efb5ba3' 6 days ago, with a total of 19 commits. The README.md file is visible, containing the text 'Работу выполнила Нестеренко Тамара, студентка 2 курса группы ПИЖ-6-о-20-1' and a code snippet starting with '#include <iostream>'. The right sidebar shows the repository's About section, including a description, README, MIT License, Releases, and Packages.

Рисунок 27 – Результат создание ветки branch\_3

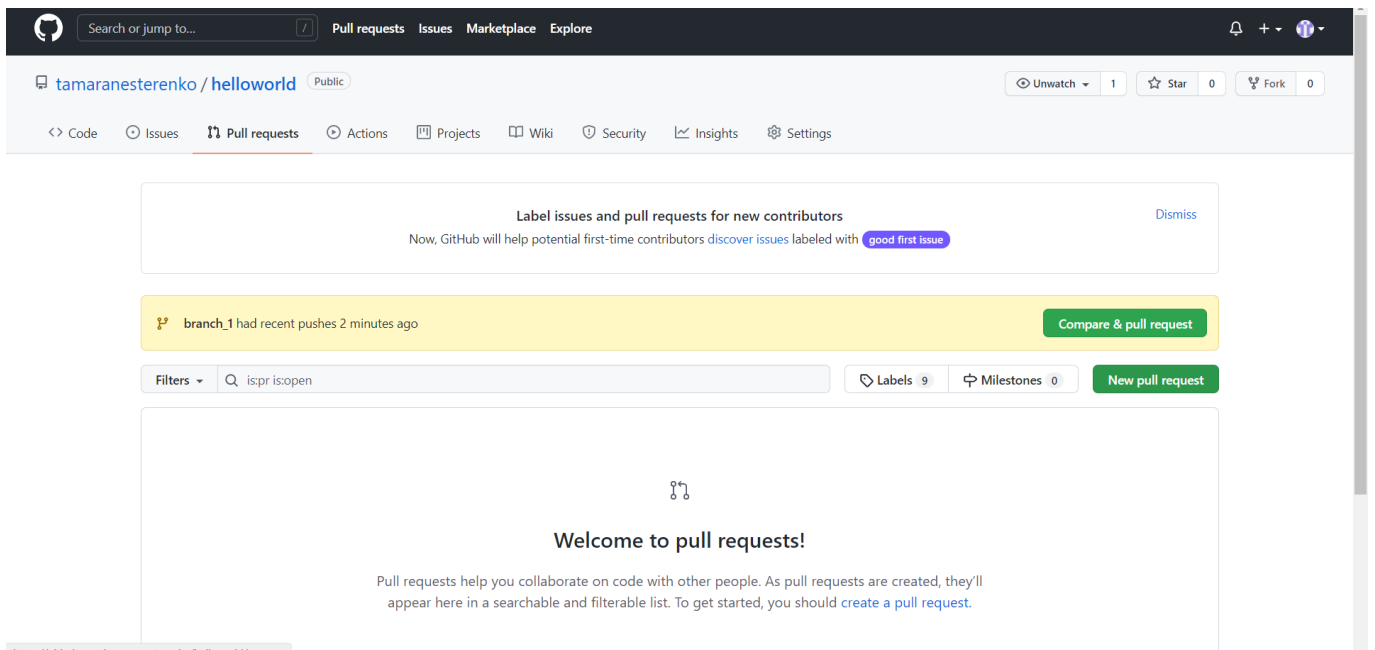


Рисунок 28 – Результат слияния веток

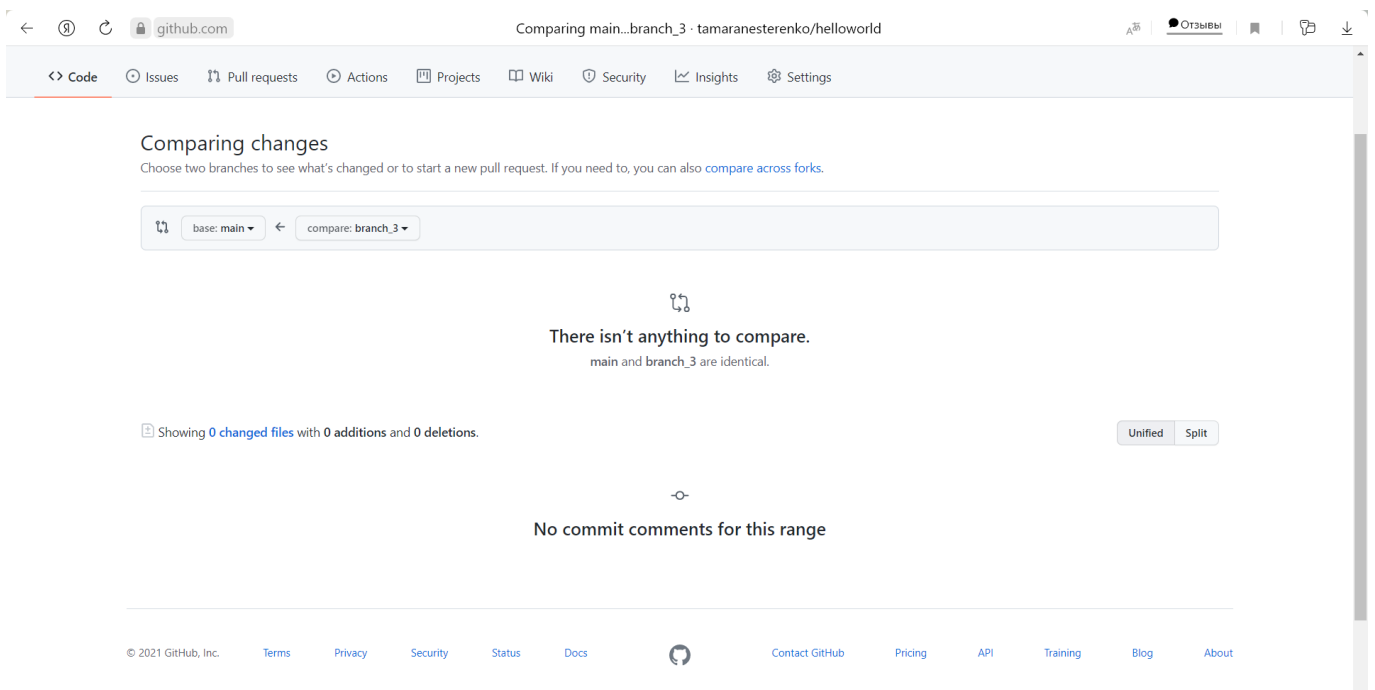


Рисунок 29 – Результат слияния веток


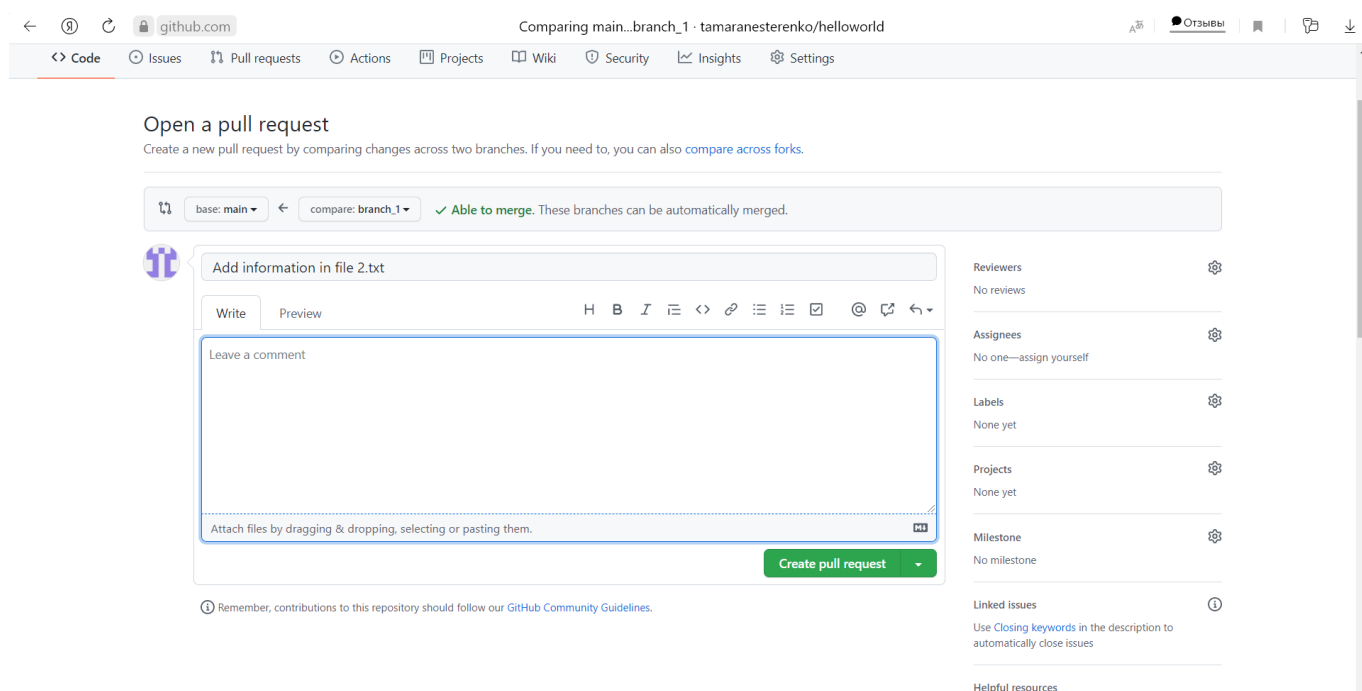
	tamaranesterenko Create 2.txt	7e0e841 now	🕒 20 commits
📁	DOC	Add files via upload	6 days ago
📄	.gitignore	Initial commit	20 days ago
📄	2.txt	Create 2.txt	now
📄	Hello.txt	Changed file Hello.txt	7 days ago
📄	LICENSE	Initial commit	20 days ago
📄	README.md	Changed file README.md	7 days ago

Рисунок 30 – Результат создания файла 2.txt



The screenshot shows the GitHub web interface for creating a pull request. The browser address bar shows 'github.com' and the page title is 'Comparing main...branch\_1 · tamaranesterenko/helloworld'. The navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The main heading is 'Open a pull request' with a subtext: 'Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).' Below this, a comparison bar shows 'base: main' and 'compare: branch\_1' with a green checkmark and the text 'Able to merge. These branches can be automatically merged.' The central area is titled 'Add information in file 2.txt' and contains a 'Write' tab, a rich text editor with a 'Leave a comment' placeholder, and a 'Create pull request' button. The right sidebar contains sections for 'Reviewers' (No reviews), 'Assignees' (No one—assign yourself), 'Labels' (None yet), 'Projects' (None yet), 'Milestone' (No milestone), 'Linked issues' (Use Closing keywords), and 'Helpful resources'.

Рисунок 31 – Результат слияния веток и создания коммита

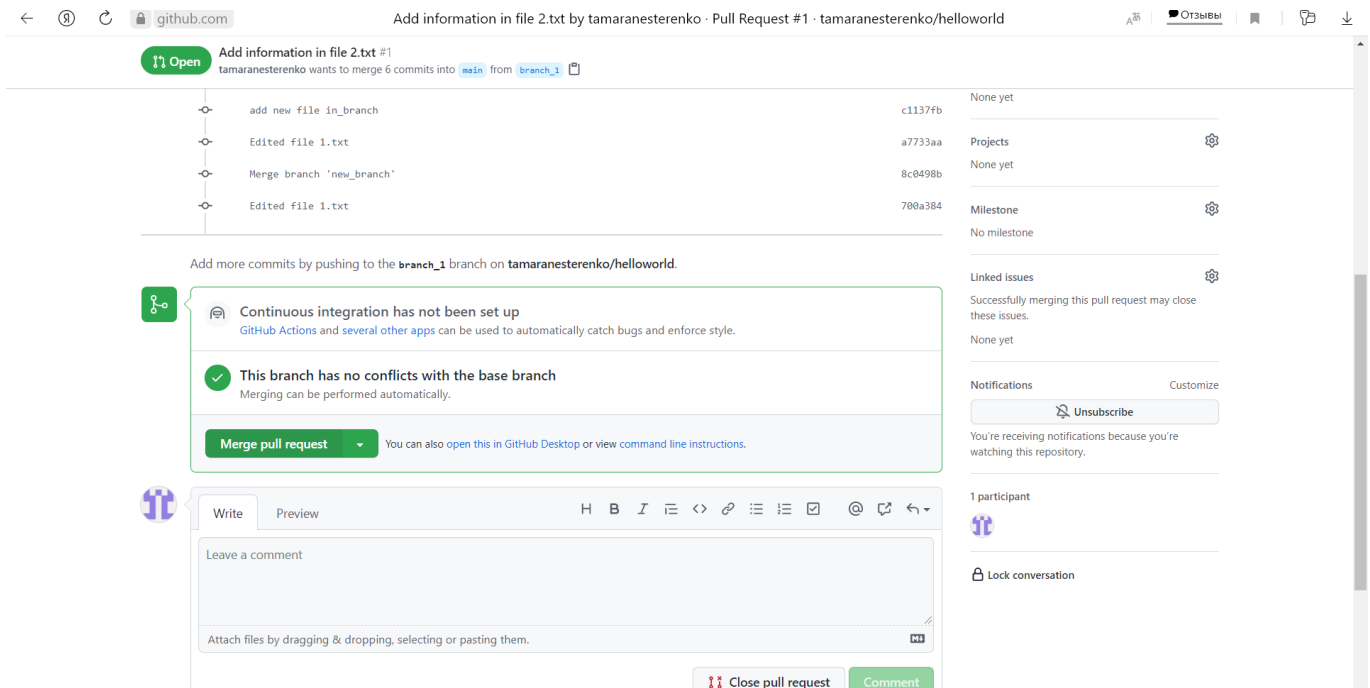


Рисунок 32 – Результат слияния веток

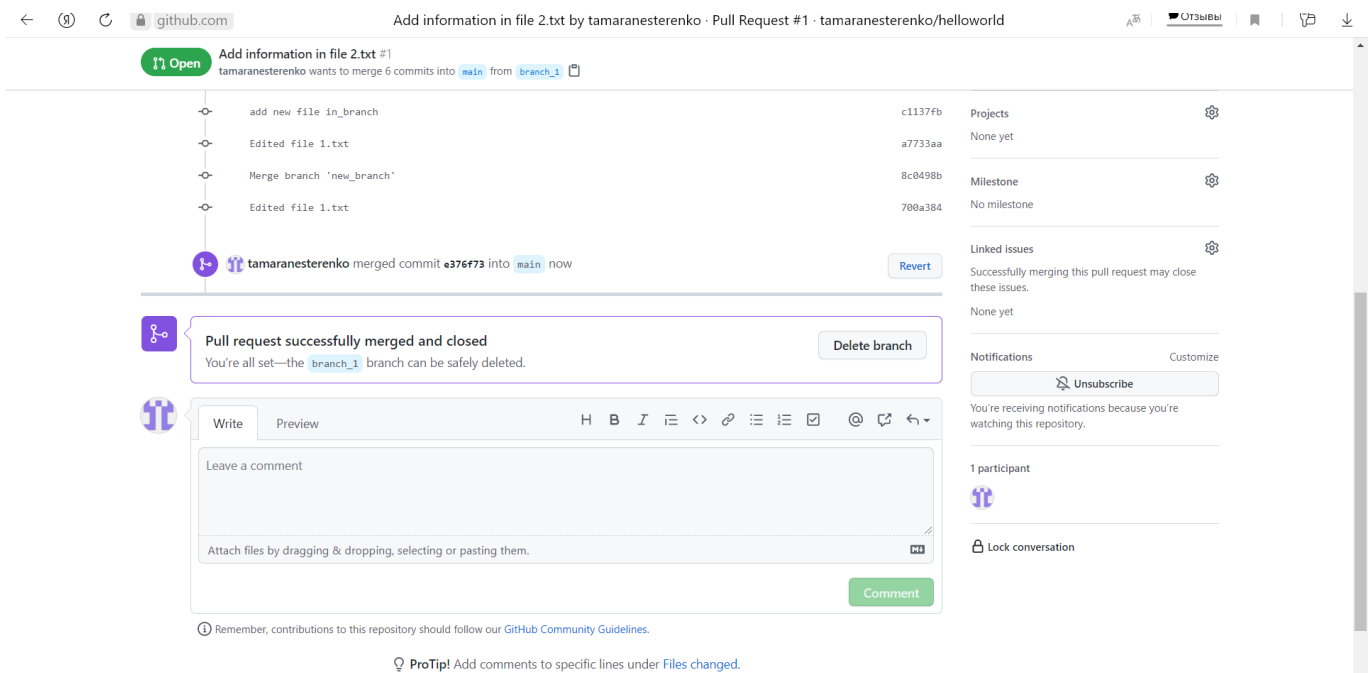


Рисунок 33 – Результат слияния веток и создание коммита

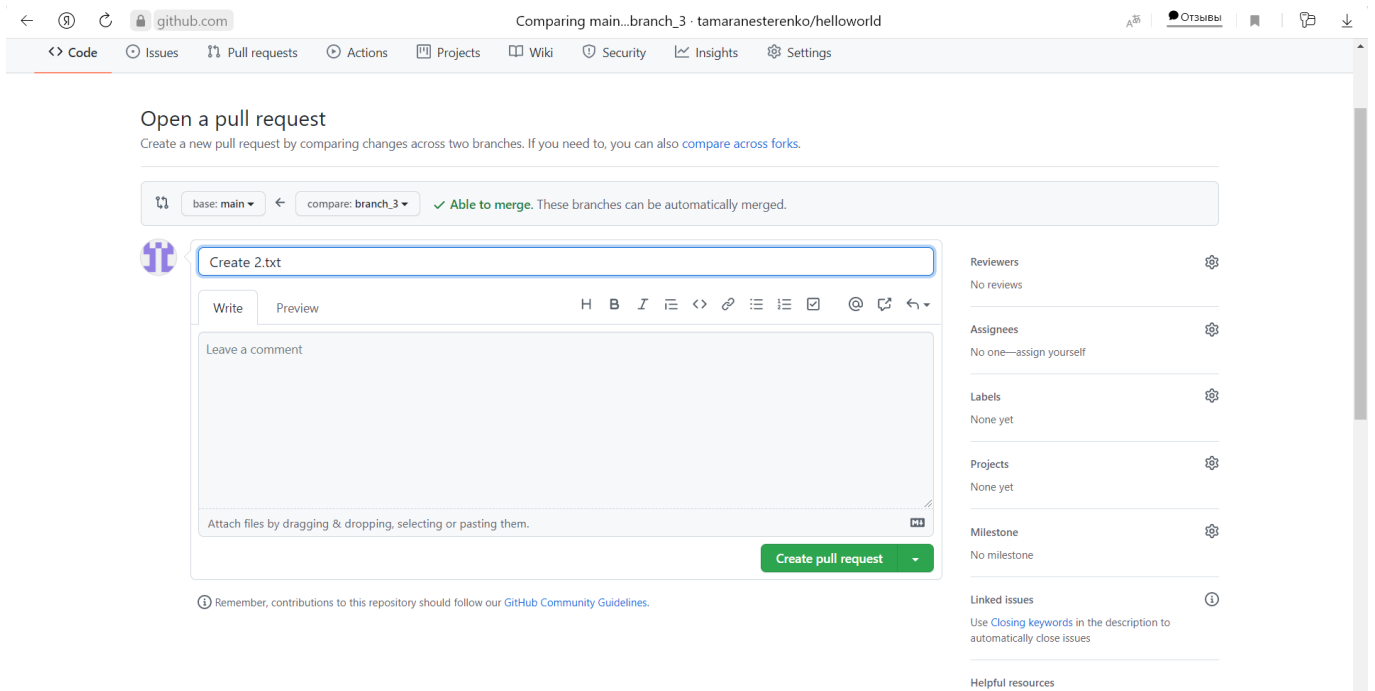


Рисунок 34 – Результат слияния веток и создание коммита

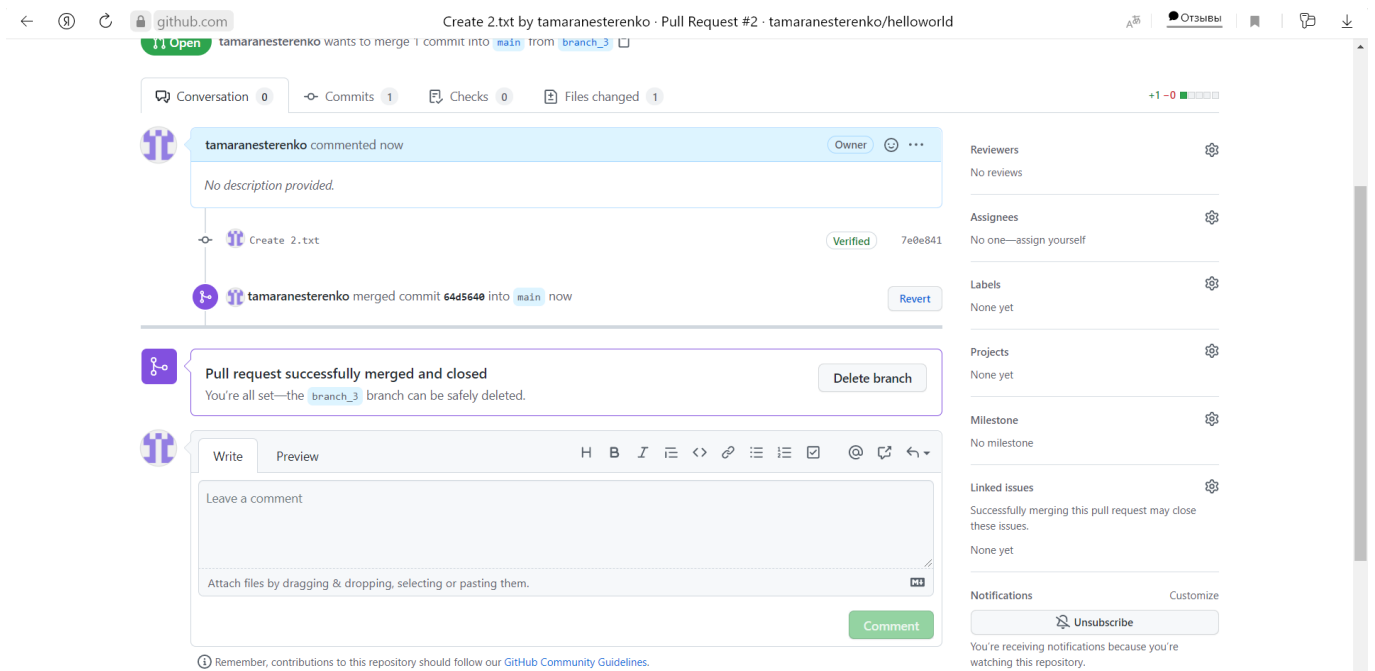


Рисунок 35 – Результат слияния веток и создание коммита

```
C:\Users\тома нестеренко\helloworld>git push origin branch_2
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (8/8), 717 bytes | 358.00 KiB/s, done.
Total 8 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
remote:
remote: Create a pull request for 'branch_2' on GitHub by visiting:
remote:   https://github.com/tamaranesterenko/helloworld/pull/new/branch_2
remote:
To https://github.com/tamaranesterenko/helloworld.git
 * [new branch]      branch_2 -> branch_2
```

Рисунок 36 – Результат отправки ветвей на удалённый сервер

## 2. Ответы на вопросы

### 1. Что такое ветка?

Ветка— это простой перемещаемый указатель на один из таких коммитов.

### 2. Что такое HEAD?

HEAD – это указатель, задача которого ссылаться на определённый коммит в репозитории. Суть данного указателя можно попытаться объяснить с разных сторон.

Во-первых, HEAD – это указатель на коммит в вашем репозитории, который станет родителем следующего коммита. Во-вторых, HEAD указывает на коммит, относительно которого будет создана рабочая копия во время операции checkout.

Вы можете это сделать командой git branch. Команда git branch только создаёт новую ветку, но не переключает на неё.

### 3. Как узнать текущую ветку?

Вы можете легко это увидеть при помощи простой команды git log, которая покажет вам куда указывают указатели веток. Эта опция называется --decorate.

### 4. Как переключаться между ветками?

Для переключения на существующую ветку выполните команду git checkout.

### 5. Что такое удалённая ветка?

Удалённые ссылки — это ссылки (указатели) в ваших удалённых репозиториях, включая ветки, теги и так далее.

### 6. Что такое ветка отслеживания?

Ветки слежения — это ссылки на определённое состояние удалённых веток. Это локальные ветки, которые нельзя перемещать; Git перемещает их автоматически при любой коммуникации с удалённым депозитарием, чтобы гарантировать точное соответствие с ним. Представляйте их как закладки для напоминания о том, где ветки в удалённых репозиториях находились во время последнего подключения к ним.

## **7. Как создать ветку отслеживания?**

Имена веток слежения имеют вид `<remote>/<branch>`. Например, если вы хотите посмотреть, как выглядела ветка `master` на сервере `origin` во время последнего соединения с ним, используйте ветку `origin/master`. Если вы с коллегой работали над одной задачей, и он отправил на сервер ветку `iss53`, при том что у вас может быть своя локальная ветка `iss53`, удалённая ветка будет представлена веткой слежения с именем `origin/iss53`.

## **8. Как отправить изменения из локальной ветки в удалённую ветку?**

Если у вас есть ветка `serverfix`, над которой вы хотите работать с кем-то ещё, вы можете отправить её точно так же, как вы отправляли вашу первую ветку. Выполните команду `git push <remote> <branch>`.

## **9. В чем отличие команд `git fetch` и `git pull`?**

Команда `git fetch` получает с сервера все изменения, которых у вас ещё нет, но не будет изменять состояние вашей рабочей директории. Эта команда просто получает данные и позволяет вам самостоятельно сделать слияние. Тем не менее, существует команда `git pull`, которая в большинстве случаев является командой `git fetch`, за которой непосредственно следует команда `git merge`. Если у вас настроена ветка слежения как показано в предыдущем разделе, или она явно установлена, или она была создана автоматически командами `clone` или `checkout`, `git pull` определит сервер и ветку, за которыми следит ваша текущая ветка, получит данные с этого сервера и затем попытается слить удалённую ветку.

## **10. Как удалить локальную и удалённую ветки?**

Вы можете удалить ветку на удалённом сервере используя параметр `--delete` для команды `git push`. Для удаления ветки `serverfix` на сервере, выполните следующую команду: `git push origin --delete serverfix`

**11. Изучить модель ветвления `git-flow` (использовать материалы статей <https://www.atlassian.com/ru/git/tutorials/comparing-workflows/gitflow-workflow>, <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. В этом рабочем процессе для регистрации истории проекта вместо одной ветки `main` используются две ветки. В главной ветке `main` хранится официальная история релиза, а ветка разработки `develop` предназначена для объединения всех функций. Кроме того, для удобства рекомендуется присваивать всем коммитам в ветке `main` номер версии.

**12. 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.**

GUI для работы с Git. Необходимо в рамках этого вопроса привести описание инструментов для работы с ветками Git, предоставляемых этим средством.