МИНЕСТЕРСТВО НАУКИ ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО «СЕВЕРО - КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЁТ

по лабораторной работе №1

Дисциплина: «Языки программирования»

Выполнил: студентка 2 курса
группы ИТС-б-о-20-1
Игнатова Елизавета Сергеевна
Проверил:
к.ф.-м.н., доцент
кафедры инфокоммуникаций
Воронкин Роман Александрович

Работа защищена с оценкой: _____

Ставрополь, 2021

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

Порядок выполнения работы:

Ссылка на репозиторий: https://github.com/Nebula139/Sky

1) Создание 3-х файлов: 1.txt, 2.txt, 3.txt, индексация первого файла;

```
C:\Users\Лиза\Лр1\Sky>git add 1.txt
```

Рисунок 1 – индексация 1-ого файла

2) Коммит 1-ого файла с комментарием “add 1.txt file”;

```
C:\Users\Лиза\Лр1\Sky>git commit -m "add 1.txt file"
[main aff721d] add 1.txt file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
```

Рисунок 2 – коммит 1-ого файла

3) Индексация 2-ого и 3-его файла;

```
C:\Users\Лиза\Лр1\Sky>git add 2.txt

C:\Users\Лиза\Лр1\Sky>git add 3.txt
```

Рисунок 3 – индексация файлов

4) Коммит файлов с комментарием “add 2.txt and 3.txt”;

```
C:\Users\Лиза\Лр1\Sky>git commit -m "add 2.txt and 3.txt"
[main 73a5bd0] add 2.txt and 3.txt
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 2.txt
 create mode 100644 3.txt
```

Рисунок 4 – коммит файлов

5) Создание новой ветки my_first_branch;

```
C:\Users\Лиза\Лр1\Sky>git branch my_first_branch
```

Рисунок 5 – создание ветки

6) Переход на новую ветку, создание файла in_branch.txt и коммит изменений;

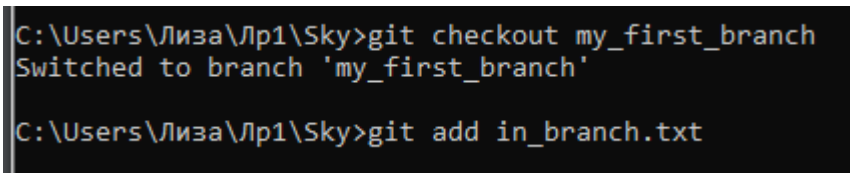

Рисунок 6 – коммит изменений

7) Создание и переход на новую ветку new_branch;

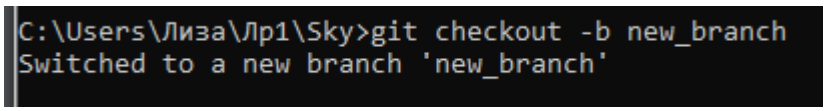

Рисунок 7 – создание новой ветки

8) Внесение изменений в файл 1.txt, добавление строчки “ new row in the 1.txt file”, коммит изменений;

9) Переход на ветку main и выполнение слияния с ветками my_first_branch и new_branch;

10) Удаление веток my_first_branch и new_branch;

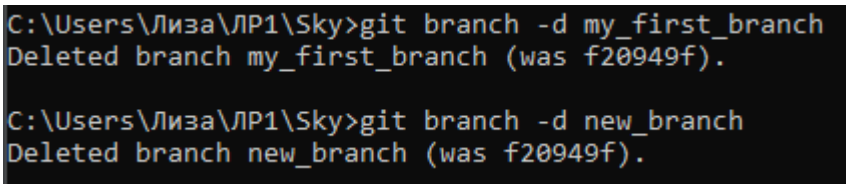

Рисунок 8 – удаление веток

11) Создание веток branch_1 и branch_2;

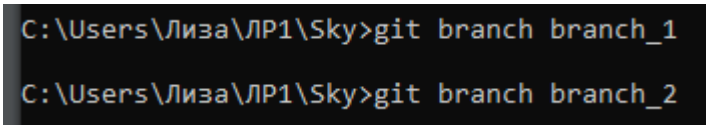

Рисунок 9 – создание новых веток

12) Переход на ветку branch_1 и внесение изменений в файл 1.txt, добавление строчки “ fix in the 1.txt”, внесение изменений в файл 3.txt, добавление строчки “ fix in the 3.txt”, коммит изменений;

13) Переход на ветку branch_2 и внесение изменений в файл 1.txt, добавление строчки “ My fix in the 1.txt”, внесение изменений в файл 3.txt, добавление строчки “My fix in the 3.txt”, коммит изменений;

14) Слияние веток и получение конфликта файлов;

```
C:\Users\Лиза\ЛР1\Sky>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 10 – конфликт файлов

15) Решение конфликтов файлов;
16) Отправка ветки branch_1 на GitHub;

```
C:\Users\Лиза\ЛР1\Sky>git push origin branch_1
info: please complete authentication in your browser...
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 4 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (16/16), 1.35 KiB | 126.00 KiB/s, done.
Total 16 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
remote:
remote: Create a pull request for 'branch_1' on GitHub by visiting:
remote:      https://github.com/Nebula139/Sky/pull/new/branch_1
remote:
To https://github.com/Nebula139/Sky.git
 * [new branch]      branch_1 -> branch_1
```

Рисунок 11 – отправка ветки на GitHub

17) Создание удалённой ветки branch_3;
18) Создание в локальном репозитории ветку отслеживания удалённой ветки branch_3;

```
C:\Users\Лиза\ЛР1\Sky>git checkout -b branch_3 origin/branch_3
Switched to a new branch 'branch_3'
Branch 'branch_3' set up to track remote branch 'branch_3' from 'origin'.
```

Рисунок 12 – создание ветки отслеживания

19) Переход на ветку branch_3, внесение изменений в файл 2.txt, добавление строчки “the final fantasy in the 4.txt file”, отправка веток на GitHub.

```
C:\Users\Лиза\ЛР1\Sky>git push
Everything up-to-date

C:\Users\Лиза\ЛР1\Sky>git push origin branch_3
Everything up-to-date
```

Рисунок 13 – отправление веток на GitHub

Контрольные вопросы:

1.Что такое ветка?

Ветка - простой перемещаемый указатель на один из нескольких родительских коммитов.

2. Что такое HEAD?

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

3. Способы создания веток.

Командами git branch или git checkout.

4. Как узнать текущую ветку?

Командой git log или командой git branch без указания каких-либо дополнительных параметров.

5. Как переключаться между ветками?

Командой git checkout.

6. Что такое удаленная ветка?

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

7. 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 удаленным репозиторием, чтобы гарантировать точное соответствие с ним.

8. Как создать ветку отслеживания?

Ветка отслеживания создаётся с помощью команды git checkout –b *branch_name* origin/*branch_name*

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

Изменения отправляются с помощью команды git push *branch_name*

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

Команда git fetch только получает данные с удалённого сервера, когда же команда git pull получает данные и выполняет слияние с веткой на рабочем месте.

11. Как удалить локальную и удаленную ветки?

Используя параметр --delete для команды git push.

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/106912/). Какие основные типы веток присуствуют в модели git-flow? Как организована работа с ветками в модели git-flow? В чем недостатки git-flow?

Основные типы веток в модели git-flow:

- ветка разработки (develop);
- функциональная ветка (feature);
- ветка выпуска (release);
- ветка исправления (hotfix);

Работа с ветками организована следующим образом:

- из ветки main создается ветка develop;
- из ветки develop создается ветка release;
- из ветки develop создаются ветки feature;
- когда работа над веткой feature завершается, она сливается в ветку develop;
- когда работа над веткой release завершается, она сливается с ветками develop и main;
- если в ветке main обнаруживается проблема, из main создается ветка hotfix;
- когда работа над веткой hotfix завершается, она сливается с ветками develop и main.

Недостатками модели являются замедление работы, сложности с частотой релизов и трата времени в случае конфликтов.

Вывод: проведено исследование базовых возможностей по работе с локальными и удаленными ветками Git.