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: студент 2 курса
группы ИТС-б-о-20-1
Попов Данила Владимирович
Проверил:
к.ф.-м.н., доцент
кафедры инфокоммуникаций
Воронкин Роман Александрович

Работа защищена с оценкой: _____

Ставрополь, 2021

Основы ветвления Git.

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

https://github.com/12W300/one.git

Ход работы:

1. Создан общедоступный репозиторий на GitHub, в котором использована лицензия MIT.

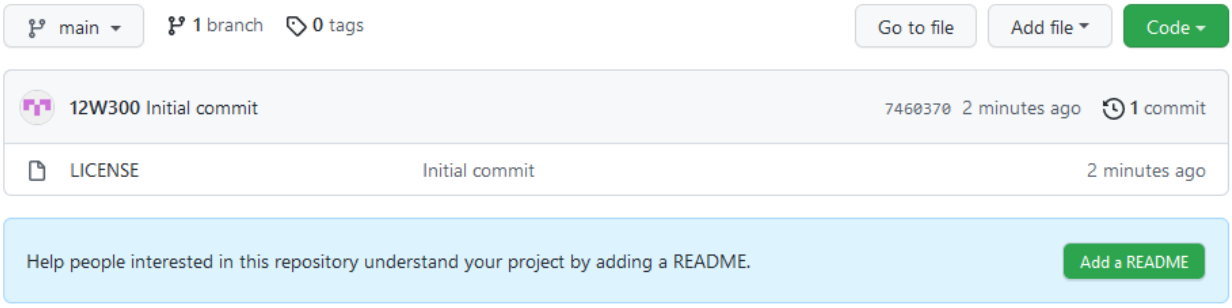

Рисунок 1 – созданный репозиторий

2. Созданы 3 txt файла. Проиндексирован первый файл и выполнен коммит.

```
C:\Users\12W300\Git\one>git add 1.txt
```

Рисунок 2 – индексация файла 1.txt

3. Проиндексированы второй и третий файлы, выполнен коммит.

```
C:\Users\12W300\Git\one>git add .

C:\Users\12W300\Git\one>git commit -m "add 2.txt and 3.txt"
```

Рисунок 3 – индексация файлов 2.txt и 3.txt

4. Создана новая ветка my_first_branch.

```
C:\Users\12W300\Git\one>git branch "my_first_branch"
```

Рисунок 4 – создание ветки

5. Выполнен переход на ветку my_first_branch, добавлен файл in_branch.txt, выполнен коммит.

6. Создана ветка new_branch и одновременно выполнен переход на неё.

```
C:\Users\12W300\Git\one>git checkout -b new_branch
Switched to a new branch 'new_branch'
```

Рисунок 5 – создание ветки и переход на неё в одной команде

7. Выполнены изменения в файле 1.txt, выполнен коммит.

8. Выполнено слияние веток master и my_first_branch, затем master и new_branch.

9. Удалены ветки my_first_branch и new_branch. Созданы ветки branch_1 и branch_2.

```
C:\Users\12W300\Git\one>git branch -d "my_first_branch"
Deleted branch my_first_branch (was 10b86c7).

C:\Users\12W300\Git\one>git branch -d "new_branch"
Deleted branch new_branch (was af17a89).
```

Рисунок 6 – удаление старых веток

10. Выполнен переход на branch_1, изменены файл 1.txt и 3.txt, выполнен коммит.

```
C:\Users\12W300\Git\one>git checkout -b branch_1
Switched to a new branch 'branch_1'

C:\Users\12W300\Git\one>git add 1.txt

C:\Users\12W300\Git\one>git add 3.txt

C:\Users\12W300\Git\one>git commit -m "changed 1.txt and 3.txt"
[branch_1 39271ef] changed 1.txt and 3.txt
 2 files changed, 2 insertions(+), 1 deletion(-)
```

Рисунок 7 – коммит изменений

11. Выполнен переход на ветку branch_2, изменены файл 1.txt и 3.txt, выполнен коммит изменений.

```
C:\Users\12W300\Git\one>git checkout -b branch_2
Switched to a new branch 'branch_2'

C:\Users\12W300\Git\one>git add 1.txt

C:\Users\12W300\Git\one>git add 3.txt

C:\Users\12W300\Git\one>git commit -m "changed 1.txt and 3.txt"
[branch_2 6942870] changed 1.txt and 3.txt
 2 files changed, 2 insertions(+), 2 deletions(-)
```

Рисунок 8 – коммит файлов ветки 2

12. Выполнено слияние веток branch_1 и branch_2.

13. Решён конфликт слияния файлов.

14. Отправлена ветка branch_1 на GitHub.

```
C:\Users\12W300\Git\one>git push origin branch_1
Enumerating objects: 27, done.
Counting objects: 100% (27/27), done.
Delta compression using up to 12 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (26/26), 2.33 KiB | 1.16 MiB/s, done.
Total 26 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), done.
remote:
remote: Create a pull request for 'branch_1' on GitHub by visiting:
remote:      https://github.com/12W300/one/pull/new/branch_1
remote:
To https://github.com/12W300/one.git
 * [new branch]      branch_1 -> branch_1
```

Рисунок 9 – отправление ветки

15. Создана удалённая ветка branch_3.

16. Создана ветка отслеживания branch_3.

```
C:\Users\12W300\Git\one>git fetch origin
From https://github.com/12W300/one
 * [new branch]      branch_3   -> origin/branch_3

C:\Users\12W300\Git\one>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 10 – создание ветки отслеживания

17. В ветке branch_3 изменён файл 2.txt.

18. Выполнено перемещение ветки master на branch_2, отправлены изменения 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 о проделанной работе: проведено исследование базовых возможностей по работе с локальными и удаленными ветками Git.