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 РАБОТЕ №1.3
дисциплины «Основы кроссплатформенного программирования»

Выполнил:
Сластёнов Андрей Сергеевич
1 курс, группа ИТС-б-о-21-1,
11.03.02 «Инфокоммуникационные технологии и системы связи»,
направленность (профиль) Сети связи и системы коммутации,
очная форма обучения

_____
(подпись)

Руководитель практики от университета:
Воронкин Р.А., канд. тех. наук, доцент,
доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г

Тема: основы ветвления Git.

Цель работы: исследовать базовые возможности по работе с локальными и удаленными ветками Git.

Порядок выполнения работы:

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

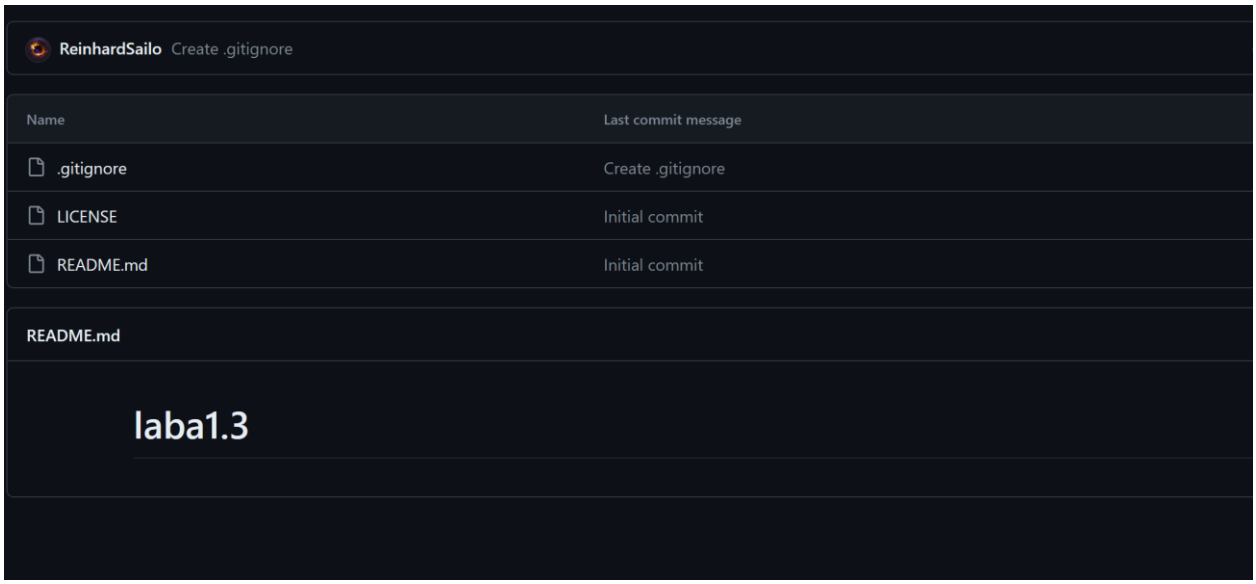


Рис. 1. Новый репозиторий.

2. Проклонировал репозиторий на свой компьютер.

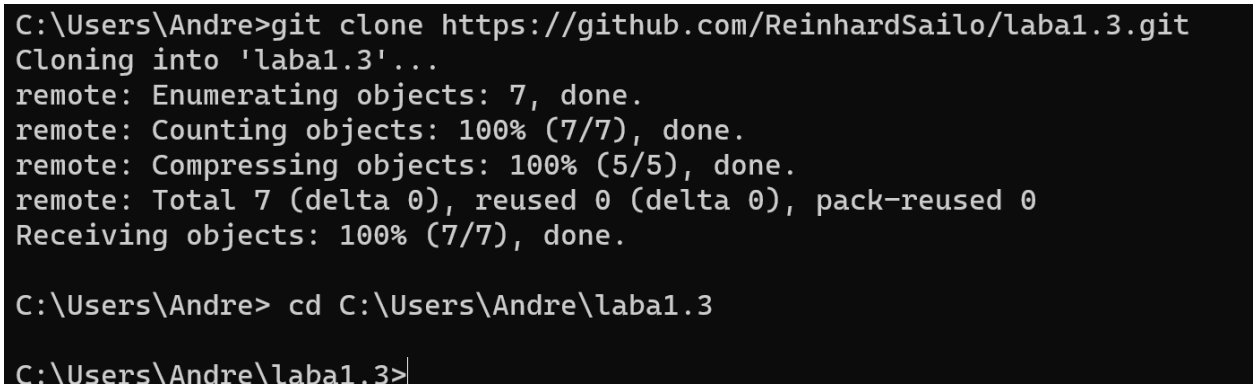


Рис. 2. Клонирование.

3. Добавил 3 новых текстовых файла.

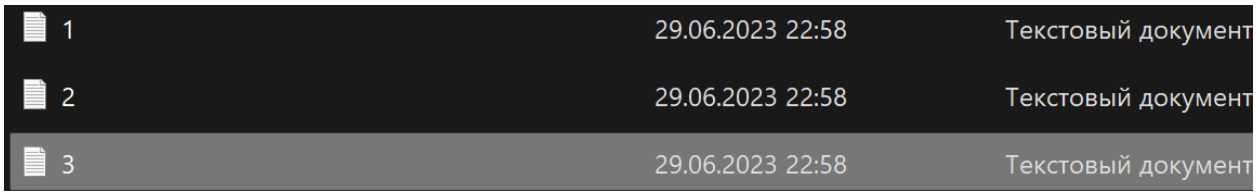


Рис. 3. Новые текстовые файлы.

4. Проиндексировал первый файл и сделать коммит.

```
C:\Users\Andre\laba1.3>git add 1.txt

C:\Users\Andre\laba1.3>git commit -m "add 1.txt file"
[main 52891a0] add 1.txt file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt

C:\Users\Andre\laba1.3>
```

Рис. 4. Добавление изменений и их фиксация.

5. Проиндексировал второй и третий файлы.

```
C:\Users\Andre\laba1.3>git add 2.txt 3.txt

C:\Users\Andre\laba1.3>git commit --amend -m "add 2.txt and 3.txt."
[main e97b33c] add 2.txt and 3.txt.
 Date: Thu Jun 29 23:09:23 2023 +0300
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
 create mode 100644 2.txt
 create mode 100644 3.txt
```

Рис. 5. Добавление изменений.

6. Перезаписал уже сделанный коммит.

```
C:\Users\Andre\laba1.3>git commit --amend -m "add 2.txt and 3.txt."
[main 2748a3a] add 2.txt and 3.txt.
 Date: Thu Jun 29 23:09:23 2023 +0300
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
 create mode 100644 2.txt
 create mode 100644 3.txt

C:\Users\Andre\laba1.3>
```

Рис. 6. Редактирование существующего коммита.

7. Создание новой ветки и переход на нее и создать новый файл in_branch.txt

```
C:\Users\Andre\laba1.3>git branch my_first_branch

C:\Users\Andre\laba1.3>git checkout my_first_branch
Switched to branch 'my_first_branch'

C:\Users\Andre\laba1.3>git status
On branch my_first_branch
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        in_branch.txt

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

C:\Users\Andre\laba1.3>git add .

C:\Users\Andre\laba1.3>git commit -m "in_branch"
[my_first_branch 9806ec7] in_branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt

C:\Users\Andre\laba1.3>
```

Рис. 8. Создание файла и фиксация.

8. Перешел вновь на новую ветку main и создал и сразу перешел на ветку new_branch

```
C:\Users\Andre\laba1.3>git checkout main
Switched to branch 'main'
Your branch and 'origin/main' have diverged,
and have 1 and 1 different commits each, respectively.
  (use "git pull" to merge the remote branch into yours)

C:\Users\Andre\laba1.3>git checkout -b new_branch
Switched to a new branch 'new_branch'

C:\Users\Andre\laba1.3>
```

Рис. 10. Создание и сразу переход на ветку.

9. Сделал изменения в файле 1.txt, добавил строчку “new row in the 1.txt file”, закоммитил изменения.

```
C:\Users\Andre\laba1.3>git add 1.txt

C:\Users\Andre\laba1.3>git commit -m "Add new row in the 1.txt file"
[new_branch 2c7f6fb] Add new row in the 1.txt file
 1 file changed, 1 insertion(+)

C:\Users\Andre\laba1.3>
```

Рис. 11. Изменение в 1.txt фиксация этих изменений.

10. Перешел на ветку main и слил ветки main и my_first_branch, после слил ветки main и new_branch.

```
C:\Users\Andre\laba1.3>git merge my_first_branch
Updating 2748a3a..9806ec7
Fast-forward
 in_branch.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 in_branch.txt

C:\Users\Andre\laba1.3>git merge new_branch
Merge made by the 'ort' strategy.
 1.txt | 1 +
 1 file changed, 1 insertion(+)

C:\Users\Andre\laba1.3>
```

Рис. 12. Слияние веток.

11. Удалил ветки my_first_branch и new_branch.

```
C:\Users\Andre\laba1.3>git branch -d my_first_branch
Deleted branch my_first_branch (was 9806ec7).

C:\Users\Andre\laba1.3>git branch -d new_branch
Deleted branch new_branch (was 2c7f6fb).

C:\Users\Andre\laba1.3>
```

Рис. 13. Удаление веток.

12. Создал ветки branch_1 и branch_2.

```
C:\Users\Andre\laba1.3>git branch branch_1

C:\Users\Andre\laba1.3>git branch branch_2

C:\Users\Andre\laba1.3>git branch
  branch_1
  branch_2
* main

C:\Users\Andre\laba1.3>
```

Рис. 14. Создание новых веток.

13. Перешел на ветку branch_1 и изменил файл 1.txt, удалил все содержимое и добавил текст “fix in the 1.txt”, изменил файл 3.txt, удалил все содержимое и добавил текст “fix in the 3.txt”, закоммитил изменения.

```
C:\Users\Andre\laba1.3>git checkout branch_1
Switched to branch 'branch_1'

C:\Users\Andre\laba1.3>git add 1.txt 3.txt

C:\Users\Andre\laba1.3>git commit -m "Fix in the 1.txt and 3.txt"
[branch_1 771215e] Fix in the 1.txt and 3.txt
 2 files changed, 2 insertions(+), 1 deletion(-)

C:\Users\Andre\laba1.3>
```

Рис. 15. Изменения в первой ветке и во второй ветка и фиксация изменений.

14. Перешел на ветку branch_2 и также изменил файл 1.txt, удалил все содержимое и добавил текст “My fix in the 1.txt”, изменил файл 3.txt, удалил все содержимое и добавил текст “My fix in the 3.txt”, закоммитил изменения.

```
C:\Users\Andre\laba1.3>git checkout branch_2
Switched to branch 'branch_2'

C:\Users\Andre\laba1.3>git add 1.txt 3.txt

C:\Users\Andre\laba1.3>git commit -m "My fix in the 1.txt and 3.txt"
[branch_2 c8654bb] My fix in the 1.txt and 3.txt
 2 files changed, 2 insertions(+), 1 deletion(-)

C:\Users\Andre\laba1.3>
```

Рис. 16. Переход на вторую ветку.

15. Слил изменения ветки branch_2 в ветку branch_1

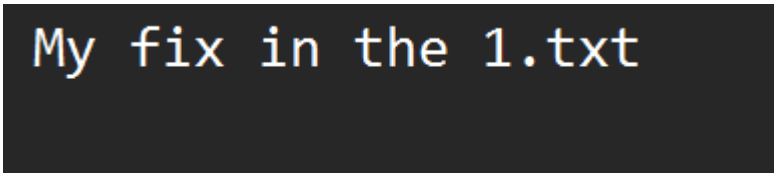
```
C:\Users\Andre\laba1.3>git checkout branch_1
Switched to branch 'branch_1'

C:\Users\Andre\laba1.3>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.

C:\Users\Andre\laba1.3>
```

Рис. 17. Слияние веток.

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



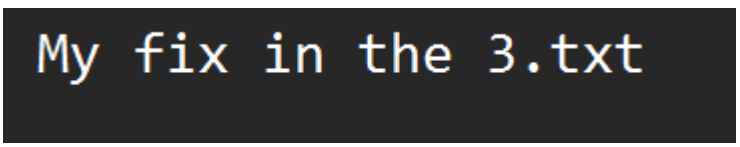


Рис. 18. Решение конфликта в ручную.

17. Отправил branch_1 на удаленный git push origin branch_1.

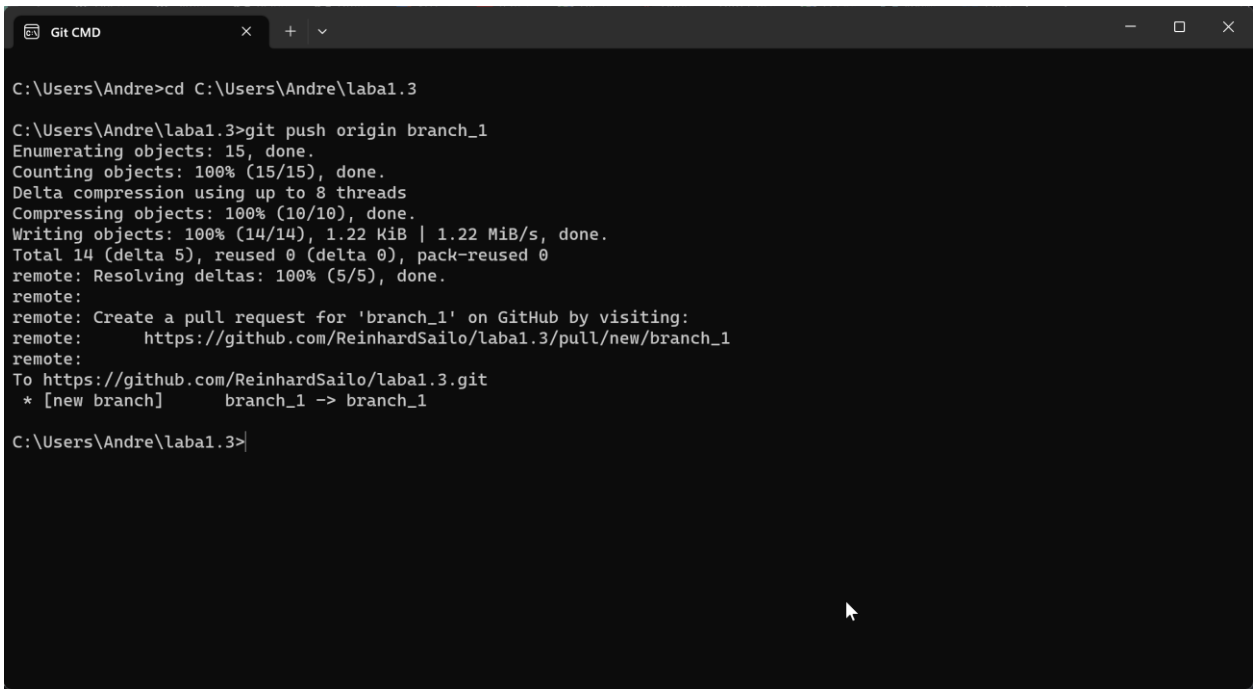
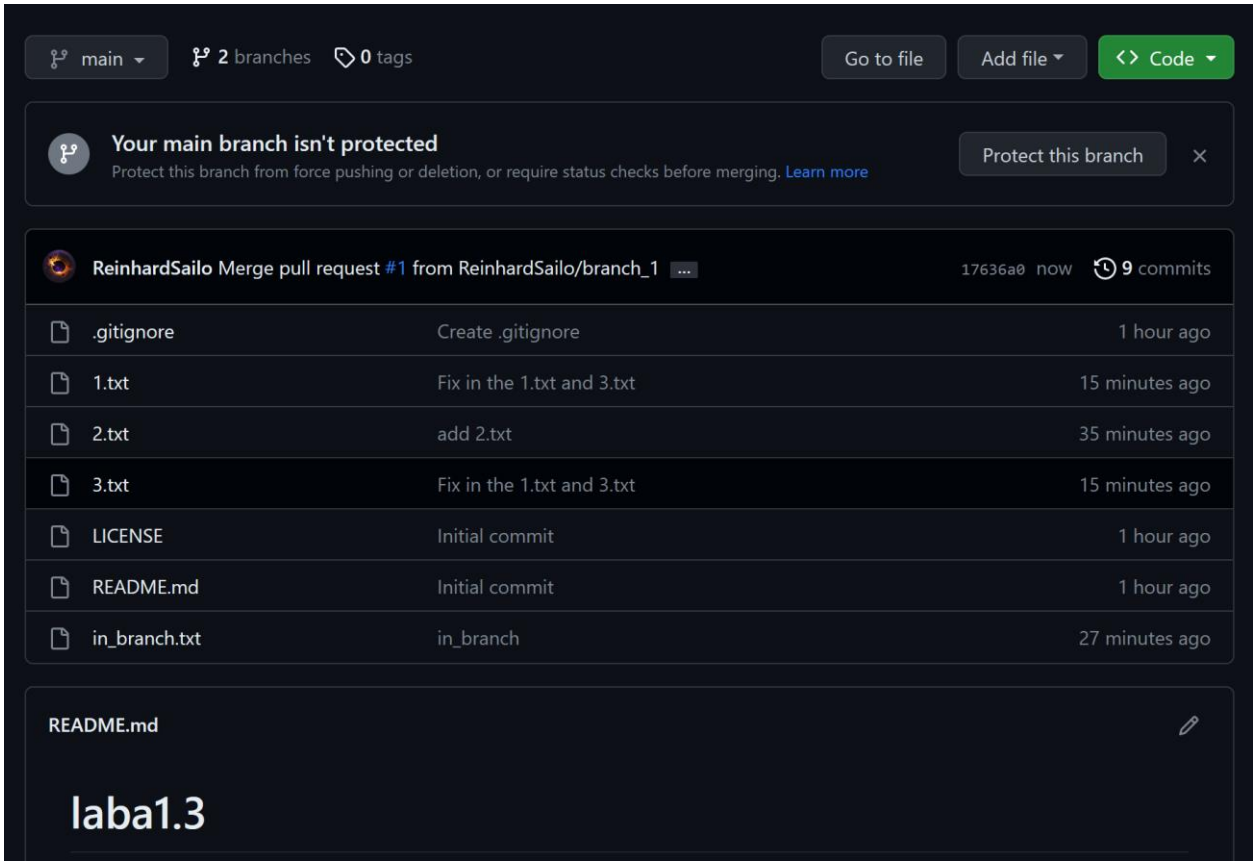


Рис. 20. Отправление ветки на удаленный сервер.



Ссылка: https://github.com/ReinhardSailo/laba1.3

Ответы на контрольные вопросы:

1. Что такое ветка? Почти каждая система контроля версий (СКВ) в какой-то форме поддерживает ветвление. Используя ветвление, Вы отклоняетесь от основной линии разработки и продолжаете работу независимо от неё, не вмешиваясь в основную линию.

2. Что такое HEAD? HEAD в Git – это указатель на текущую ссылку ветви, которая, в свою очередь, является указателем на последний сделанный вами коммит или последний коммит, который был извлечен из вашего рабочего каталога.

3. Способы создания веток. Создать ветку можно с помощью двух команд. Команда, которая просто создает ветку: git branch “name_branch”. Команда, которая создает ветку и сразу же к ней переходит: git checkout –b “name_branch”.

4. Как узнать текущую ветку? Текущую ветку можно узнать с помощью команды: git branch.

5. Как переключаться между ветками? Между ветками можно переключаться с помощью команды: git checkout “name_branch”.

6. Что такое удаленная ветка? Удаленные ветки - это ссылки на определённое состояние удалённых веток. Это локальные ветки, которые нельзя перемещать.

7. Что такое ветка отслеживания? Отслеживаемые ветки — это локальные ветки, которые напрямую связаны с удалённой веткой. Если, находясь на отслеживаемой ветке, вы наберёте git push, Git уже будет знать, на какой сервер и в какую ветку отправлять изменения.

8. Как создать ветку отслеживания? Ветку отслеживания можно создать с помощью команды: git checkout -- track origin/.

9. Как отправить изменения из локальной ветки в удаленную ветку? Отправить изменения из локальной ветки в удаленную можно с помощью команды: git push origin .

10. 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.

11. Как удалить локальную и удаленную ветки? Для удаление локальной ветки используется команда: git branch –d Для удаления удаленной ветки используется команда: git push –delete origin/

12. Какие основные типы веток присутствуют в модели git-flow? Как организованна работа с ветками в модели git-flow? В чем недостатки git-flow? Существуют следующие типы ветвей:

1) ветви функциональностей;

2) ветви релизов;

3) ветви исправлений.

Ветви функциональностей (feature branches), также называемые иногда тематическими ветвями (topic branches), используются для разработки новых функций, которые должны появиться в текущем или будущем релизах.

Ветви релизов (release branches) используются для подготовки к выпуску новых версий продукта. Они позволяют расставить финальные точки над i перед выпуском новой версии.

Ветви для исправлений (hotfix branches) весьма похожи на ветви релизов (release branches), так как они тоже используются для подготовки новых выпусков продукта, разве лишь незапланированных.

Недостатки git flow: авторам приходится использовать ветку develop вместо master, поскольку master зарезервирован для кода.

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 излишяя.