Отчет по лабораторной работе №2

Операционные системы

Подхалюзина Виолетта Михайловна

Содержание

# 1 Цель работы

Цель данной лабораторной работы – изучение идеологии и применения средств контроля версий, освоение умения по работе с git.

# 2 Задание

1. Создать базовую конфигурацию для работы с git
2. Создать ключ SSH
3. Создать ключ GPG
4. Настроить подписи Git
5. Заргеистрироваться на GitHub
6. Создать локальный каталог для выполнения заданий по предмету.

# 3 Выполнение лабораторной работы

## 3.1 Установка программного обеспечения

Устанавливаю необходимое программное обеспечение git и gh через терминал с помощью команд: dnf install git и dnf install gh (рис. [1](#fig:001)).



Figure 1: Установка git и gh

## 3.2 Базовая настройка git

Задаю в качестве имени и email владельца репозитория свои имя, фамилию и электронную почту (рис. [2](#fig:002)).



Figure 2: Задаю имя и email владельца репозитория

Настраиваю utf-8 в выводе сообщений git для их корректного отображения (рис. [3](#fig:003)).



Figure 3: Настройка utf-8 в выводе сообщений git

Начальной ветке задаю имя master (рис. [4](#fig:004)).



Figure 4: Задаю имя начальной ветки

Задаю параметры autocrlf и safecrlf для корректного отображения конца строки (рис. [5](#fig:005)).

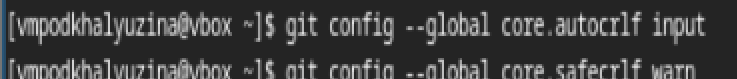


Figure 5: Задаю параметры autocrlf и safecrlf

## 3.3 Создание ключа SSH

Создаю ключ ssh размером 4096 бит по алгоритму rsa (рис. [6](#fig:006)).

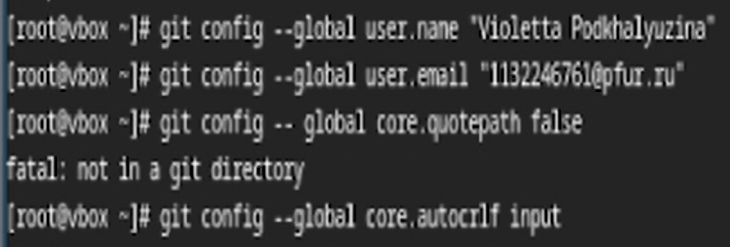


Figure 6: Генерация ssh ключа по алгоритму rsa

Создаю ключ ssh по алгоритму ed25519 (рис. [7](#fig:007)).

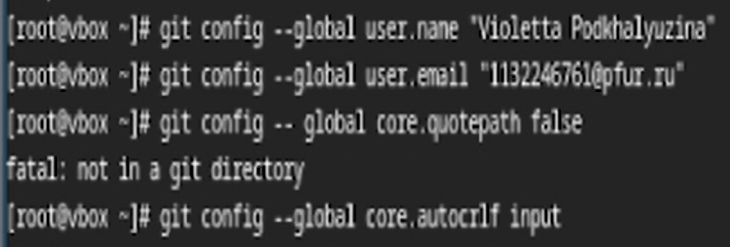


Figure 7: Генерация ssh ключа по алгоритму ed25519

## 3.4 Создание ключа GPG

Генерирую ключ GPG, затем выбираю тип ключа RSA and RSA, задаю максиммальную длину ключа: 4096, оставляю неограниченный срок действия ключа. Далее отвечаю на вопросы программы о личной информации (рис. [8](#fig:008)).

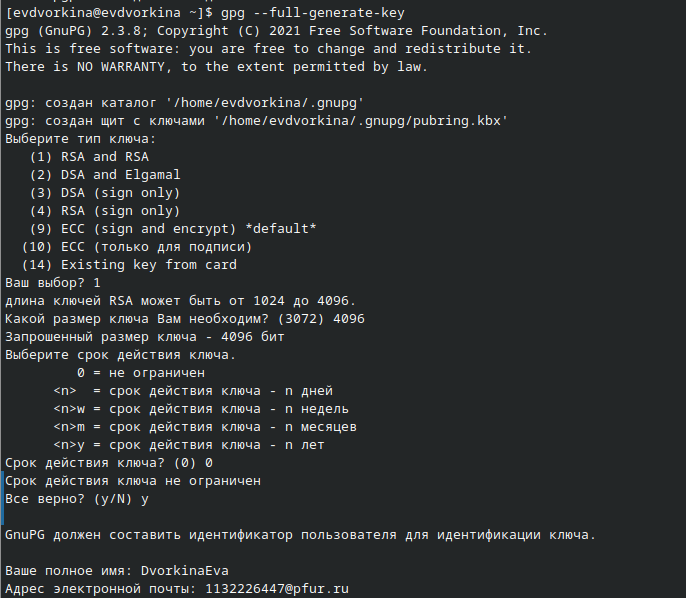


Figure 8: Генерация ключа

Ввожу фразу-пароль для защиты нового ключа (рис. [9](#fig:009)).

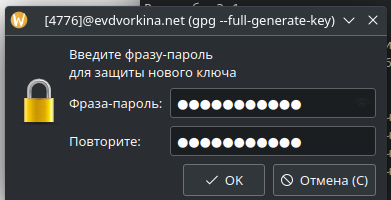


Figure 9: Защита ключа GPG

## 3.5 Регистрация на Github

У меня уже был создан аккаунт на Github, соответственно, основные данные аккаунта я так же заполняла и проводила его настройку, поэтому просто вхожу в свой аккаунт (рис. [10](#fig:010)).

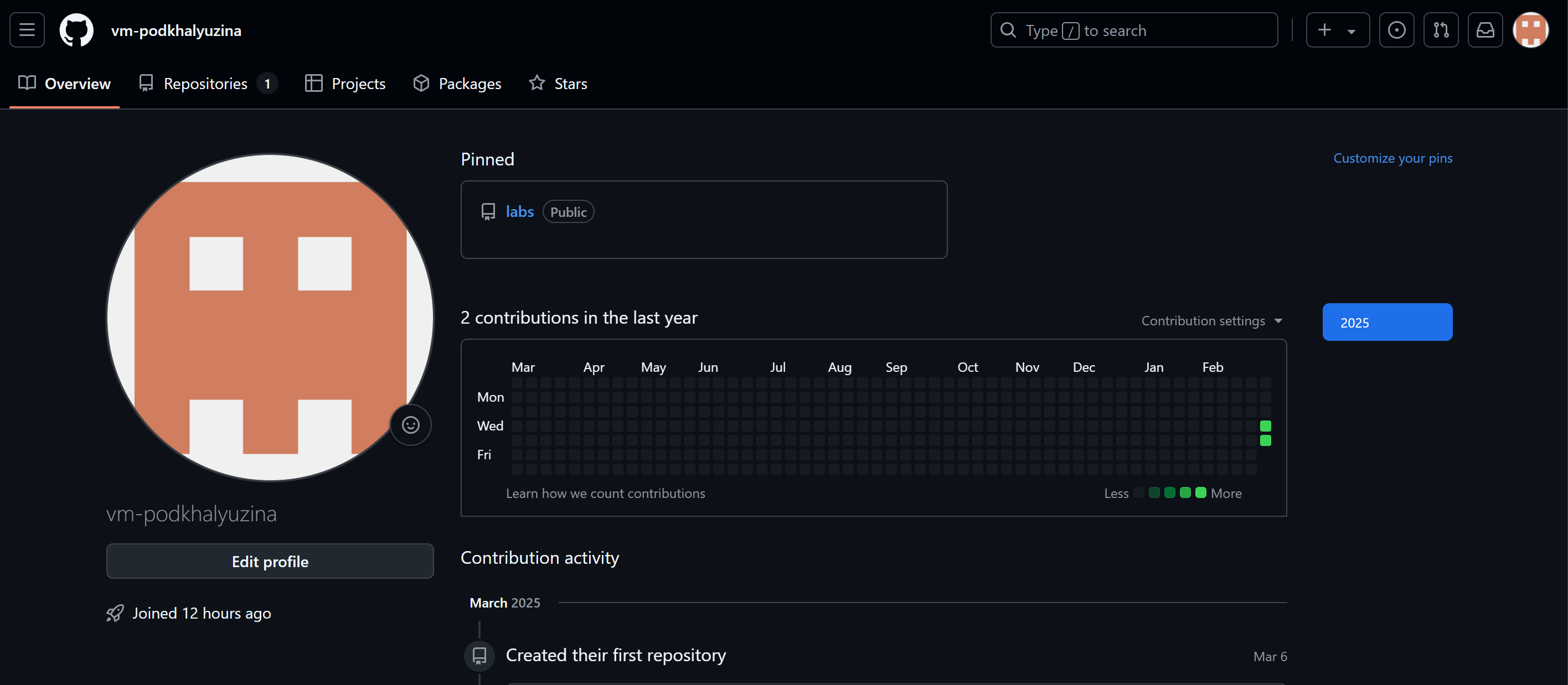


Figure 10: Аккаунт на Github

## 3.6 Добавление ключа GPG в Github

Вывожу список созданных ключей в терминал, ищу в результате запроса отпечаток ключа (последовательность байтов для идентификации более длинного, по сравнению с самим отпечатком, ключа), он стоит после знака слеша, копирую его в буфер обмена (рис. [11](#fig:011)).



Figure 11: Вывод списка ключей

Ввожу в терминале команду, с помощью которой копирую сам ключ GPG в буфер обмена, за это отвечает утилита xclip (рис. [12](#fig:012)).

Figure 12: Копирование ключа в буфер обмена

Figure 12: Копирование ключа в буфер обмена

Открываю настройки GirHub, ищу среди них добавление GPG ключа (рис. [13](#fig:013)).

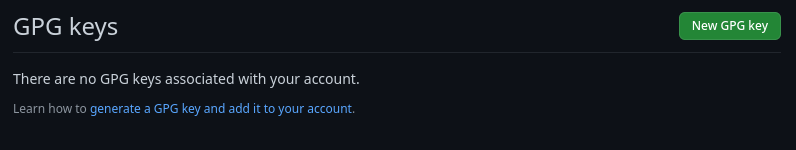


Figure 13: Настройки GitHub

Нажимаю на “New GPG key” и вставляю в поле ключ из буфера обмена (рис. [14](#fig:014)).



Figure 14: Добавление нового PGP ключа

Я добавила ключ GPG на GitHub (рис. [15](#fig:015)).

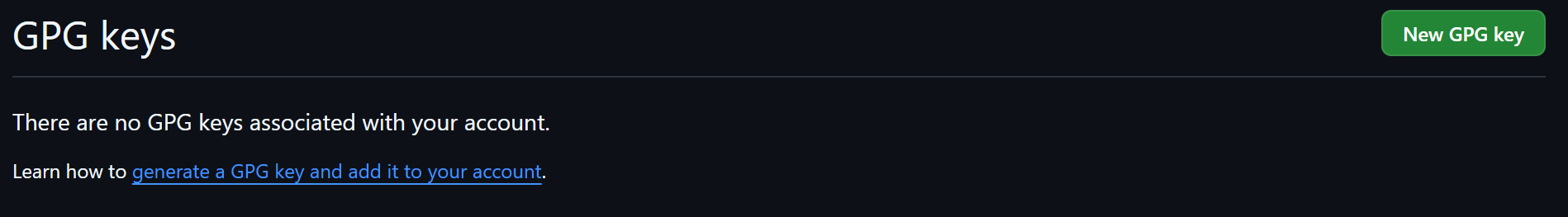


Figure 15: Добавленный ключ GPG

## 3.7 Настроить подписи Git

Настраиваю автоматические подписи коммитов git: используя введенный ранее email, указываю git использовать его при создании подписей коммитов (рис. [16](#fig:016)).

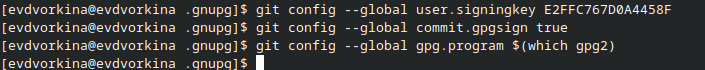


Figure 16: Настройка подписей Git

## 3.8 Настройка gh

Начинаю авторизацию в gh, отвечаю на наводящие вопросы от утилиты, в конце выбираю авторизоваться через браузер (рис. [17](#fig:017)).

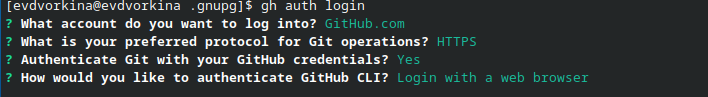


Figure 17: Авторизация в gh

Завершаю авторизацию на сайте (рис. [18](#fig:018)).

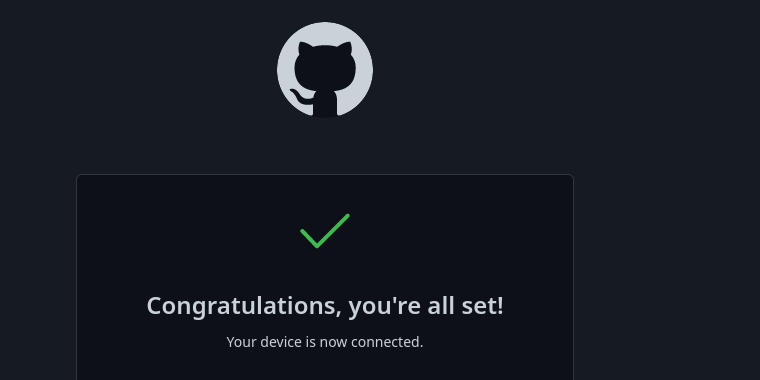


Figure 18: Завершение авторизации через браузер

Вижу сообщение о завершении авторизации под именем vmpodkhalyuzina (рис. [19](#fig:019)).

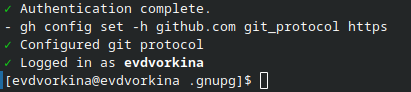


Figure 19: Завершение авторизации

## 3.9 Создание репозитория курса на основе шаблона

Сначала создаю директорию с помощью утилиты mkdir и флага -p, который позволяет установить каталоги на всем указанном пути. После этого с помощью утилиты cd перехожу в только что созданную директорию “Операционные системы”. Далее в терминале ввожу команду gh repo create study\_2022-2023\_os-intro –template yamadharma/course-directory-student-trmplate –public, чтобы создать репозиторий на основе шаблона репозитория. После этого клонирую репозиторий к себе в директорию, я указываю ссылку с протоколом https, а не ssh, потому что при авторизации в gh выбрала протокол https (рис. [20](#fig:020)).

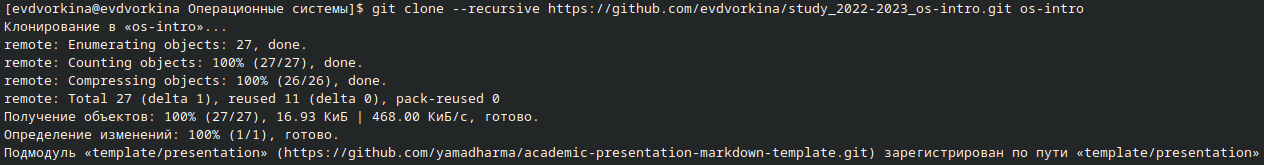


Figure 20: Создание репозитория

Перехожу в каталог курса с помощью утилиты cd, проверяю содержание каталога с помощью утилиты ls (рис. [21](#fig:021)).





Figure 21: Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm, далее создаю необходимые каталоги используя makefile (рис. [22](#fig:022)).

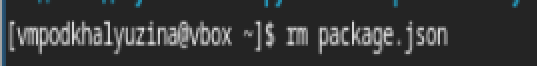






Figure 22: Удаление файлов и создание каталогов

Добавляю все новые файлы для отправки на сервер (сохраняю добавленные изменения) с помощью команды git add и комментирую их с помощью git commit (рис. [23](#fig:023)).





Figure 23: Отправка файлов на сервер

Отправляю файлы на сервер с помощью git push (рис. [24](#fig:024)).



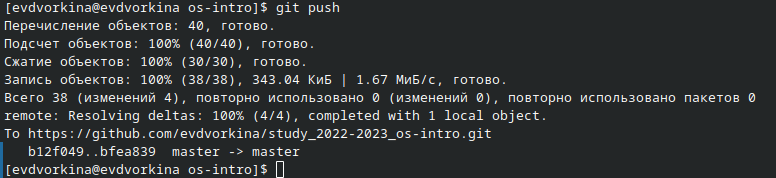


Figure 24: Отправка файлов на сервер

# 4 Выводы

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий, освоила умение по работе с git.

# 5 Ответы на контрольные вопросы.

Системы контроля версий (VCS) — это программы, упрощающие работу с изменяющейся информацией. Они позволяют хранить несколько версий одного документа, предоставляя доступ к более ранним версиям. Такие системы удобны для коллективной работы, так как фиксируют, кто и когда вносил изменения. Основные задачи VCS — сохранение истории изменений, учет причин внесенных правок, поиск авторов изменений и организация совместной работы.

Репозиторий — это хранилище версий, включающее все документы, историю их изменений и служебные данные. Коммит фиксирует изменения, сохраняя разницу между версиями. История содержит все изменения, позволяя вернуться к любой версии. Рабочая копия — это актуальная версия проекта, загруженная из репозитория.

Централизованные VCS (например, CVS, TFS, AccuRev) используют единое хранилище проекта, из которого пользователи берут файлы, изменяют их и возвращают обратно. В децентрализованных VCS (например, Git, Bazaar) у каждого пользователя есть свой репозиторий, позволяющий обмениваться изменениями без обязательного центрального хранилища.

Сначала создается и подключается удаленный репозиторий, затем в процессе работы изменения отправляются на сервер. Перед началом работы участник проекта загружает нужную версию, вносит правки и загружает обновленную версию обратно. Все предыдущие версии сохраняются и доступны в любой момент.

Основные команды Git:

* git init — создание репозитория
* git pull — получение изменений из центрального хранилища
* git push — отправка локальных изменений
* git status — просмотр измененных файлов
* git diff — просмотр изменений
* git add . — добавление всех измененных файлов
* git add [файл] — добавление конкретного файла
* git rm [файл] — удаление файла из индекса (в локальной директории остается)
* git commit -am "[описание]" — сохранение изменений с комментарием
* git commit — сохранение изменений с вводом комментария через редактор
* git checkout -b [ветка] — создание новой ветки
* git checkout [ветка] — переключение между ветками
* git push origin [ветка] — отправка изменений ветки
* git merge --no-ff [ветка] — слияние ветки
* git branch -d [ветка] — удаление локальной слитой ветки
* git branch -D [ветка] — принудительное удаление локальной ветки
* git push origin :[ветка] — удаление ветки из центрального репозитория
* git push --all — отправка всех изменений

Ветвление — это создание параллельных участков разработки, исходящих от одной версии. Основная ветка остается стабильной, а дополнительные используются для разработки новых функций, после чего изменения сливаются.

Для исключения временных файлов из репозитория используют .gitignore, где указывают шаблоны игнорируемых файлов.

# Список литературы

1. Лабораторная работа № 2 [Электронный ресурс] URL: https://esystem.rudn.ru/mod/page/view.php?id=970819