Лабораторная работа №2

Система контроля версий Git

Коровкин Никита Михайлович

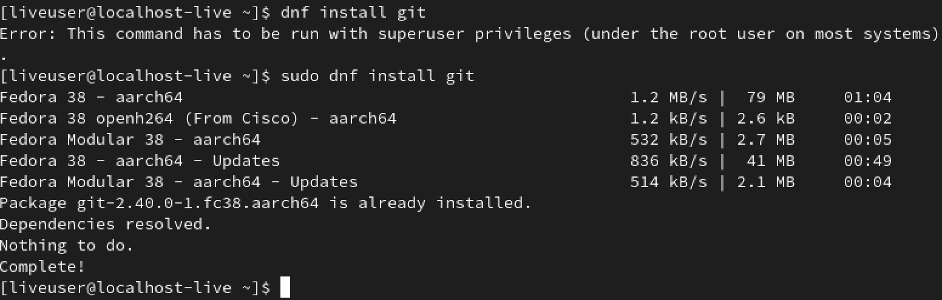
Содержание

# 1 Цель работы

Целью работы является изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой git.

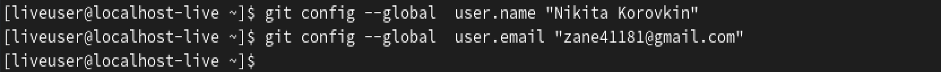
# 2 Выполнение лабораторной работы

Перед началом работы с git нам необходимо сперва установить его, а затем сделать предварительную конфигурацию (см. рис. 1):



Установка git

Откроем терминал и введём первые две команды. В них укажем имя и email владельца репозитория (см. рис. 2).



Настройка имени и адреса email

После того, как мы задали имя пользователя и адрес электронной почты, введём команду, чтобы настроить utf-8 в выводе сообщений git (см. рис. 3).

Настройка utf-8

Настройка utf-8

Теперь мы зададим имя начальной ветки. Она будет называться master (см. рис. 4).

Добавление названия начальной ветке

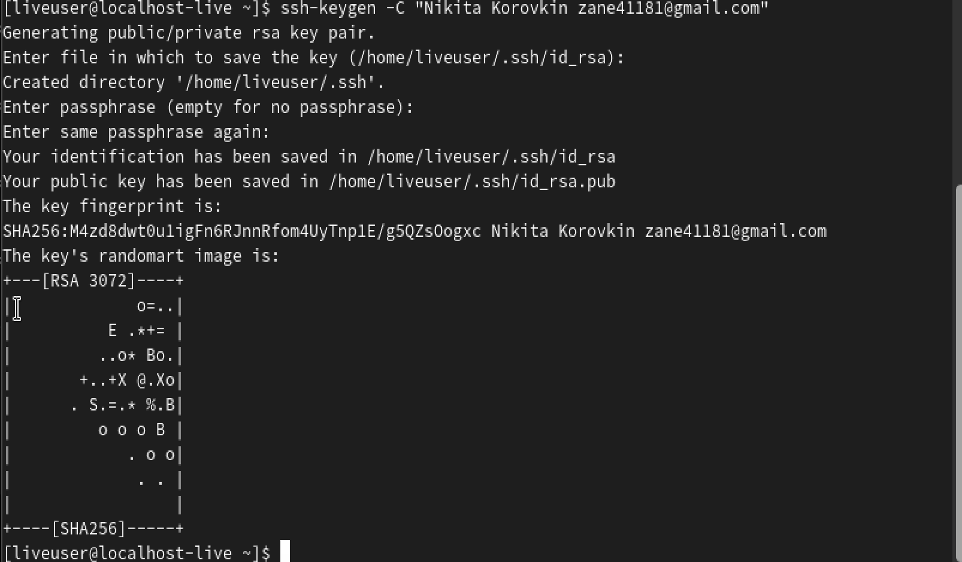
Добавление названия начальной ветке

Затем зададим следующие два параметра: autocrlf и safecrlf (см. рис. 5). Параметр autocrlf нужен для того, чтобы в главном репозитории все переводы строк в текстовых файлах были одинаковы. А команда safecrlf проверяет обратимость преобразования для текущей настройки.

Параметры autocrlf и safecrlf

Параметры autocrlf и safecrlf

Чтобы продолжить работу, нам необходимо сперва сгенерировать открытый SSH ключ. Он необходим для индентификации пользователя на сервере репозиториев. В качествем аргументов необходимо указать свое имя и электронную почту(см.рис. 6).

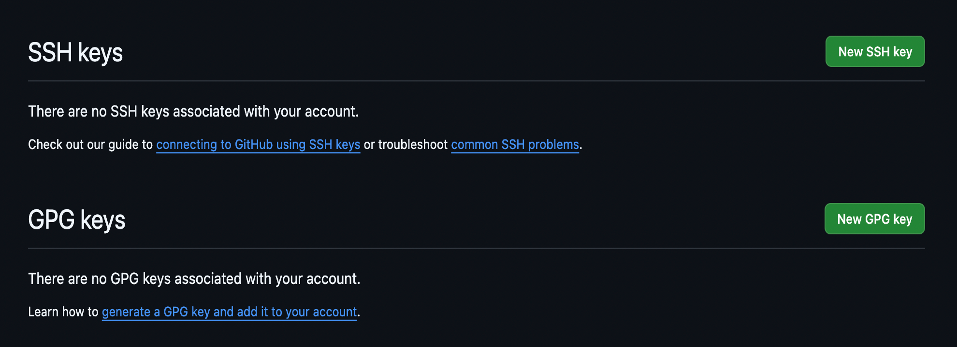


Генерация SSH ключа

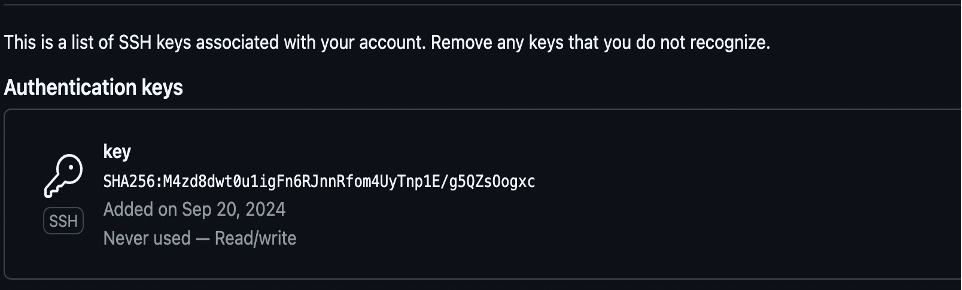
Теперь вставим этот ключ в специальное поле на сайте GitHub. Оно находится в разделе settings. Перейдя в settings, найдём пункт SSH и GPS keys. Нажимаем на SSH key, вставляем ключ в нужное поле(см.рис.7-9).



Меню GitHub и раздел settings



Раздел для создания SSH ключа



SSH ключ

Сгенерированный SSH ключ мы копируем в буфер обмена при помощи команды cat ~/.ssh/id\_rsa.pub | xclip -sel clip (см.рис.10).

Использование команды для копирования в буфер обмена

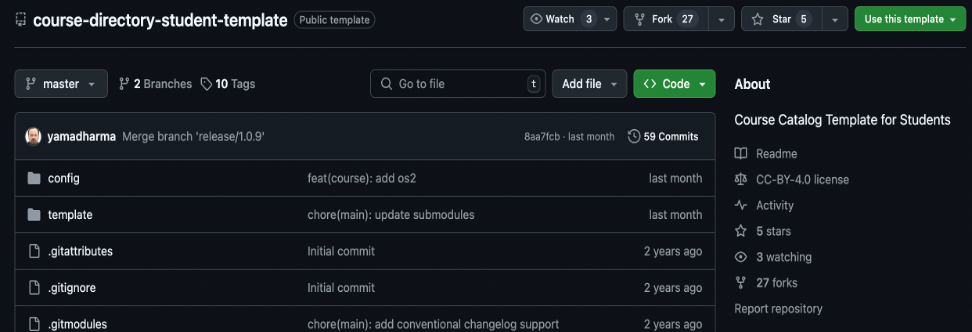
Использование команды для копирования в буфер обмена

После того, как мы добавили ключ нам необходимо при помощи терминала создать каталог «Архитектура компьютера» (см.рис.11).

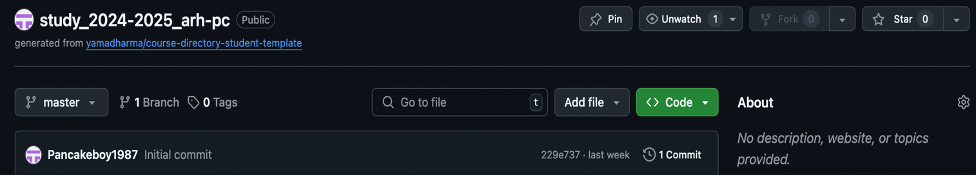
Создание каталога “Архитектура компьютера”

Создание каталога “Архитектура компьютера”

Теперь на сайте GitHub нам нужно будет создать репозиторий курса на основе шаблона. Шаблон находится по данному адресу: https://github.com/yamadharma/cour se-directory-student-template. Перейдя по ссылке, нажимаем «use this template» и создаём свой репозиторий(см.рис.12-13).

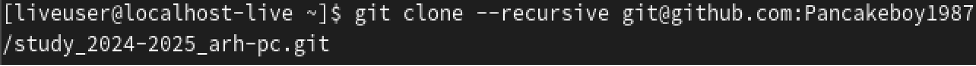


Используем шаблон для создания своего репозитория

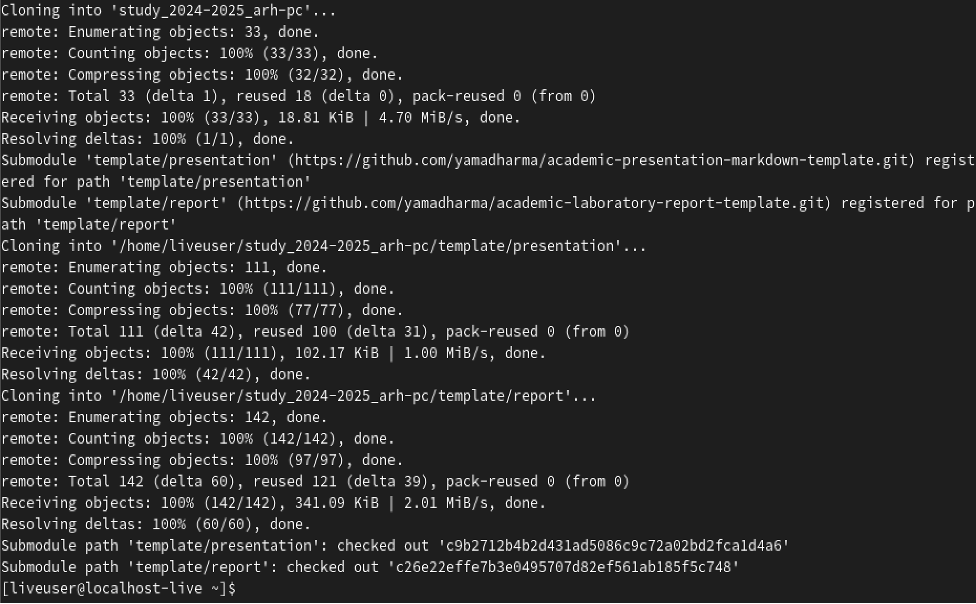


Созданный репозиторий

Теперь, когда репозиторий создан на сайте, нам необходимо клонировать его. Для этого воспользуемся приведенной ниже командой(см.рис.14).



Клонирование репозитория



Весь процесс клонирования репозитория

На рисунке выше(см.рис.15) изображен весь процесс клонирования. После того, как процесс завершен, можно перейти к следующему этапу – настройке каталога курса.

Для этого сперва перейдем в нужный каталог с помощью cd ~/work/study/2024-2025/“Архитектура компьютера”/arch-pc (см.рис.16)

Переход в нужный каталог.

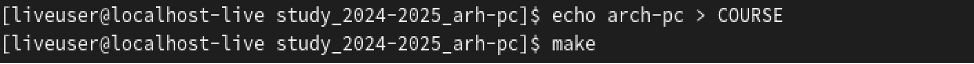
Переход в нужный каталог.

Теперь удалим ненужные файлы при помощи команды rm(см.рис.17)

Удаление лишних файлов

Удаление лишних файлов

После того, как все лишние файлы удалены, создадим новые при помощи команд echo arch-pc > COURSE и make(см.рис.18).



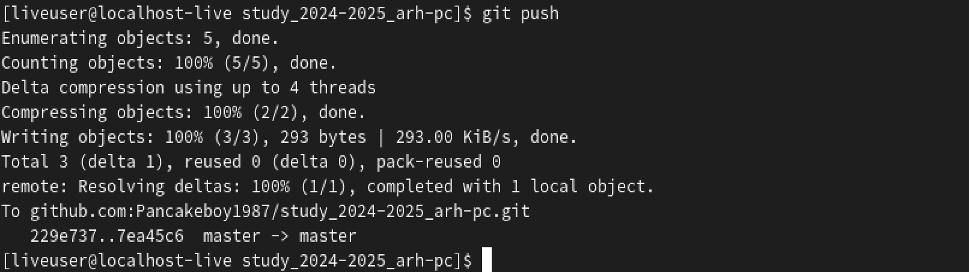
Создание новых файлов

Следующий этап – отправка файлов на сервер. Для этого мы воспользуемся командой git add . , которая предназначена для добавления на сервер сразу всех файлов и каталогов. После этого пропишем команду git commit -am ‘feat(main): make course structure’. Она сохраняет внесённые изменения и добавляет комментарий (см.рис.19).

Добавление каталогов на сервер, сохранение изменений

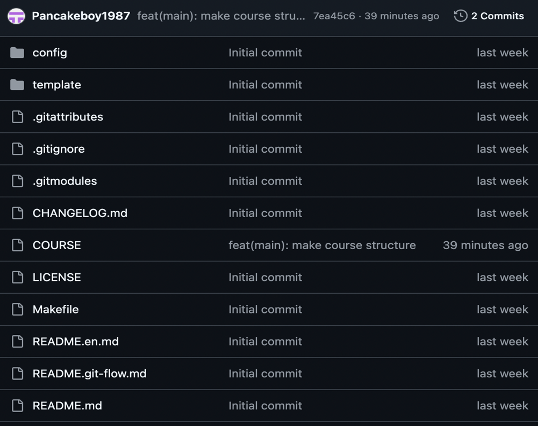
Добавление каталогов на сервер, сохранение изменений

Теперь воспользуемся командой push, чтобы загрузить все изменения на сервер(см.рис.20).

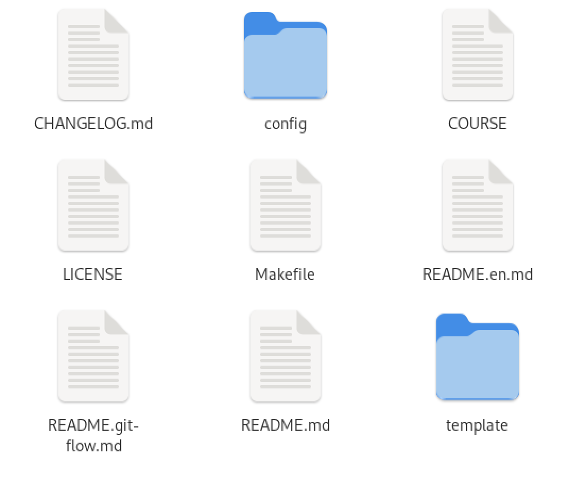


Загрузка изменений на сервер

Остаётся только сравнить файлы GitHub с файлами на компьютере (см.рис.21-22).



Файлы GitHub

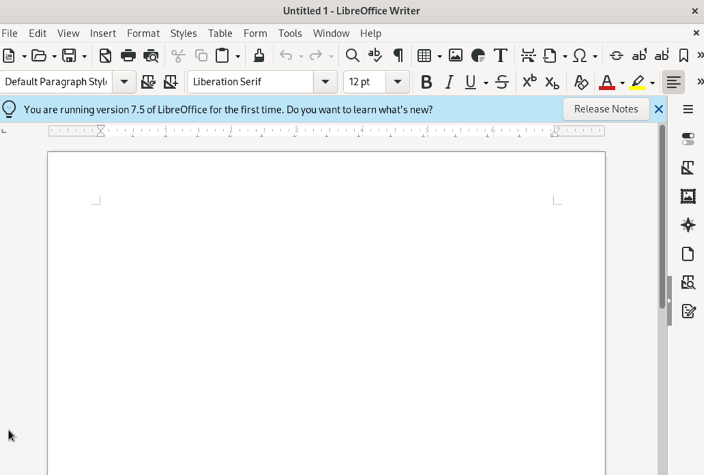


Файлы на компьютере

Файлы совпадают, значит работа выполнена правильно.

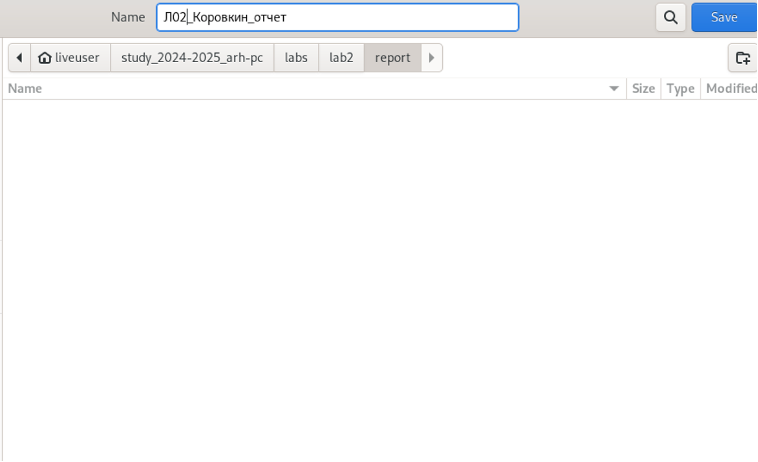
# 3 Выполнение заданий для самостоятельной работы

Приступим к выполнению самостоятельной работы. Для этого сперва создадим файл отчёта. Для этого воспользуемся LibreOffice(см.рис.23).



Создание отчёта

Сохраним файл отчёта в нужном каталоге(см.рис.24)



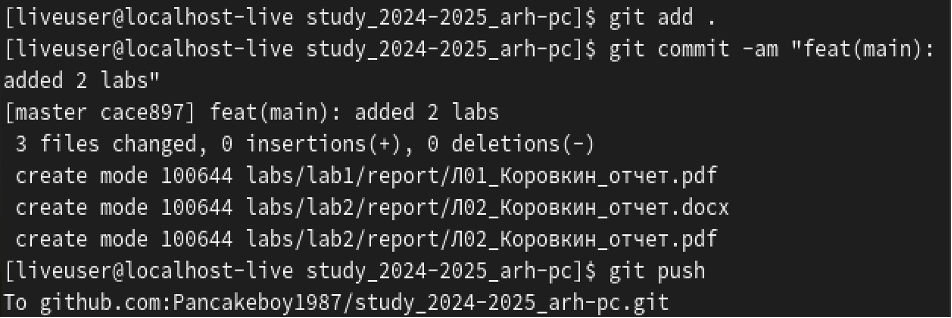
Сохранение файла в нужном каталоге

Теперь, когда наш файл сохранен, скопируем отчёт о предыдущей лабораторной работе в каталог labs/lab01/report. Для этого воспользуемся командой cp(см.рис.25).

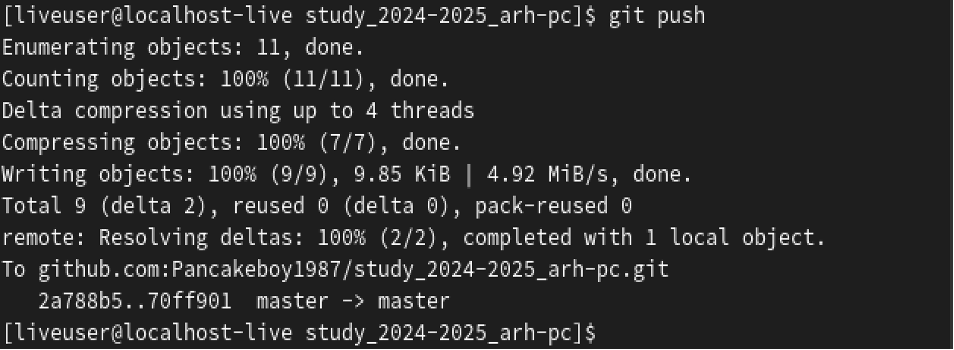
Копируем файл в нужную директорию

Копируем файл в нужную директорию

Остаётся только отправить наши файлы и каталоги на сервер. Для этого проделаем те же самые действия, что и в основной части лабораторной работы. Сперва добавляем файлы и каталог, используем commit для утверждения изменений и отправляем на сервер при помощи push(см.рис.26-27).

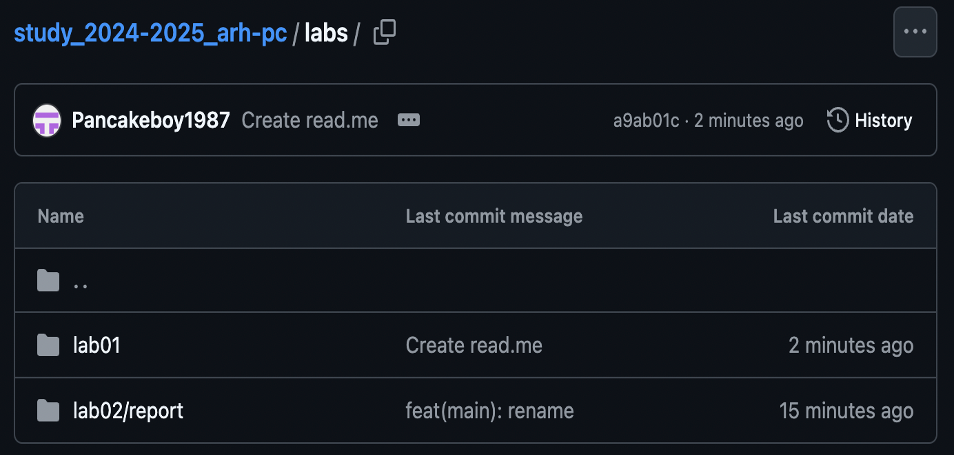


Использование комманд add и commit



Загрузка файлов на сервер

Проверим, получилось ли правильно загрузить файлы на GitHub(см.рис.28). Как мы видим, нужные каталоги на месте. Работа выполнена верно.



Загруженные каталоги с файлами

# 4 Выводы

В ходе выполнения данной лабораторной работы были получены навыки работы с системой контроля версий Git, такие как: первоначальная настройка системы, создание репозитория, изменение, сохранение и загрузка файлов на сервер.