Лабораторная работа №2

Система контроля версий Git

Калашникова Дарья Викторовна

Содержание

[1 Цель работы 1](#_Toc179582121)

[2 Выполнение лабораторной работы 1](#_Toc179582122)

[3 Задание для самостоятельной работы 8](#_Toc179582123)

[4 Выводы 11](#_Toc179582124)

# 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 его необходимо предварительно настроить. Для этого введем в терминал следующие команды: (рис. 1).

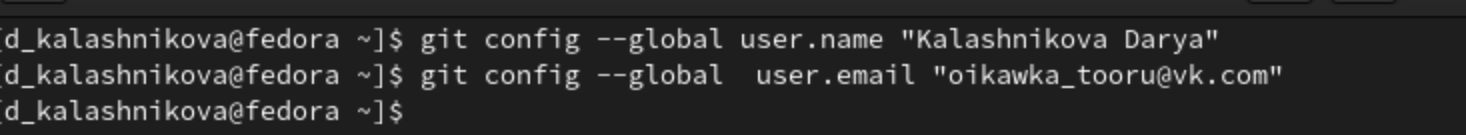


Рис. 1: Настройка имени и адреса эл. Почты.

Введя их, мы задали имя и электронный адрес почты пользователя. Теперь введем следующую команду: (рис. 2).

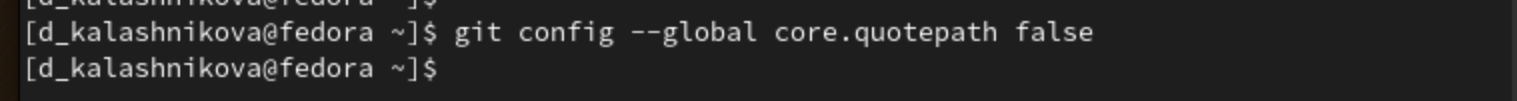


Рис. 2: Настройка UTF8 в выводе сообщений Git.

Благодаря ней, мы настроили вывод сообщений git в кодировке utf8. Теперь мы должны задать имя для (рис. 3). начальной ветки. Мы назовем ее master:

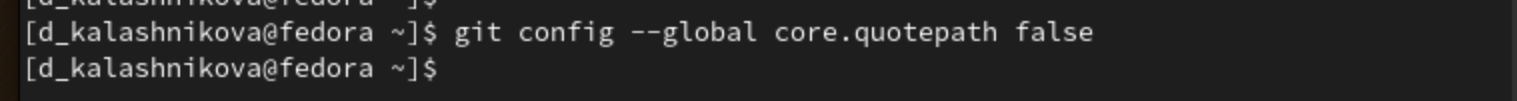


Рис. 3: Конфигурация имени начальной ветки.

Теперь введем следующую команду: (рис. 4).

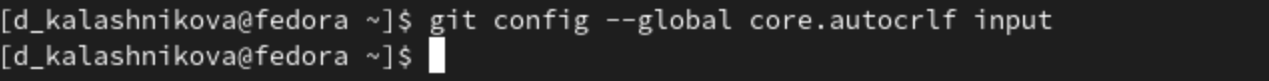


Рис. 4: Настройка автоконвертации окончаний строк.

Таким образом мы указываем git автоматически конвертировать CRLF окончания строк в LF во время commit’а. Теперь выполним вот эту команду (рис. 5). Она будет отвечать за то, чтобы печатать предупреждение в случае, если преобразования из CRLF будут необратимыми. (рис. 5).



Рис. 5: Настройка вывода предупреждений о необратимых преобразованиях из CRLF.

Для того, чтобы сервер мог идентифицировать пользователя, необходимо сгенерировать несколько ssh ключей. Начнем с генерации открытого ключа. Для этого мы введем следующую команду, указав имя, фамилию пользователя и его адрес электронной почты в качестве аргумента: (рис. 6).

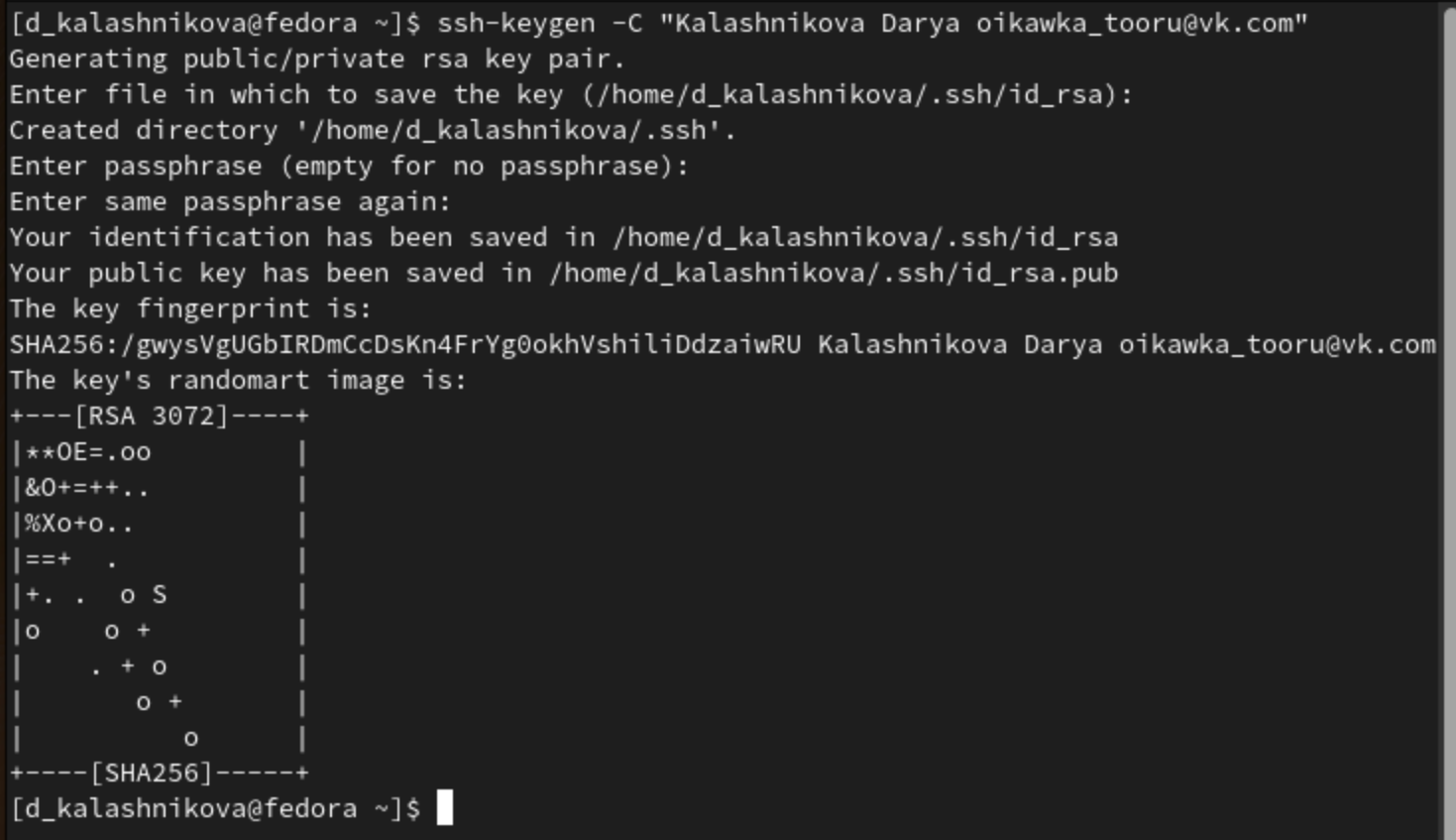


Рис. 6: Генерация ssh ключа.

Зайдем на сайт GitHub: (рис. 7).

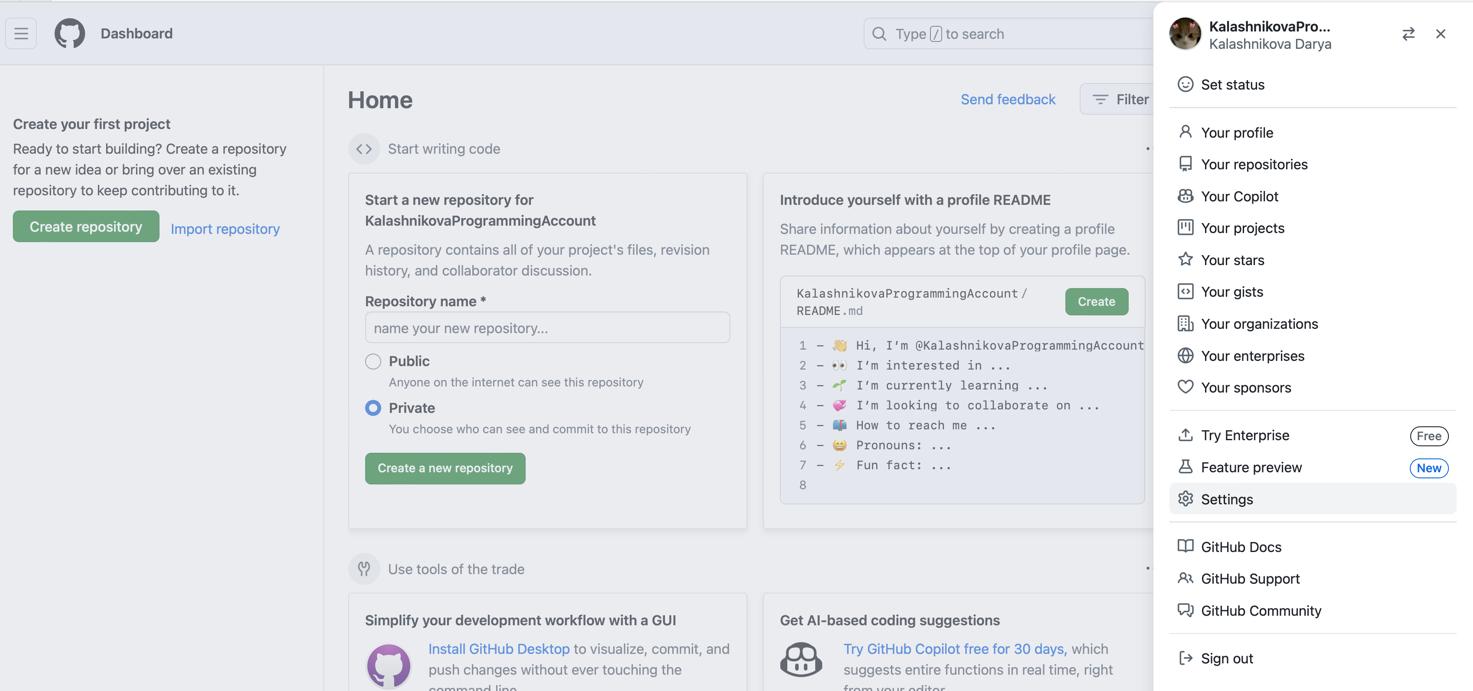


Рис. 7: Перейдём в пункт settings.

Находим раздел SSH and GPG keys и нажимаем New SSH key: (рис. 8).

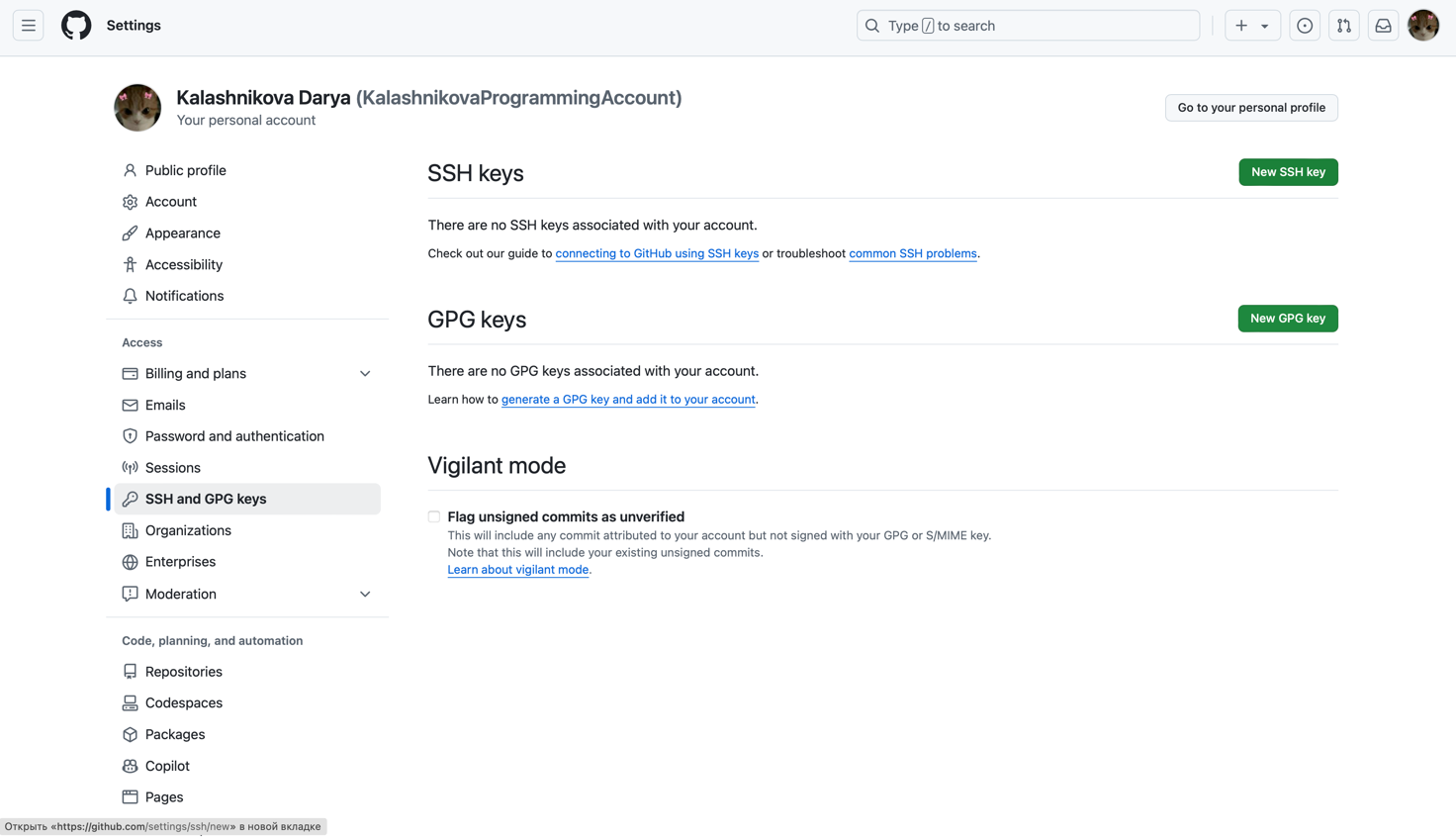


Рис. 8: Раздел настроек “SSH ang GPG keys”.

В предложенное поле “Key” нам необходимо вставить ключ, который мы только что сгенерировали: (рис. 9).

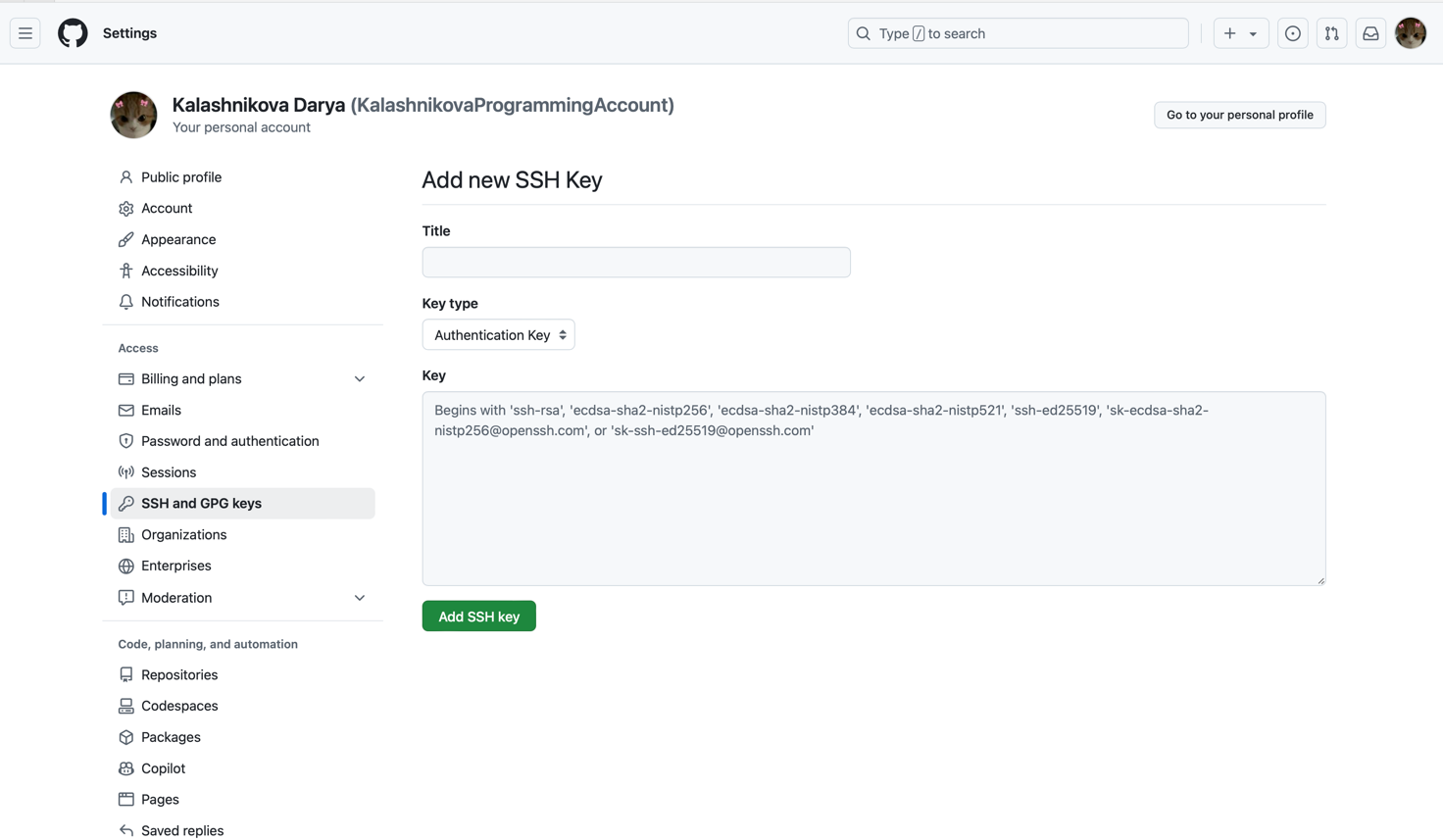


Рис. 9: Поле ввода ключа в окне добавления нового SSH ключа.

Для того, чтобы скопировать ключ для последующей вставки, нам необходимо ввести следующую команду: (рис. 10).(рис. 11).

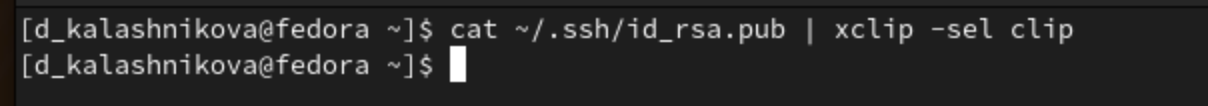


Рис. 10: Считывание и копирование публичного ключа. Команда “cat” прочитает данные из файла id\_rsa.pub, а команда xclip выгрузит их в буфер обмена. Остается лишь вставить содержимое буфера обмена, то есть наш ключ, в предложенное поле “Key” и указать имя ключа в поле “Title”.

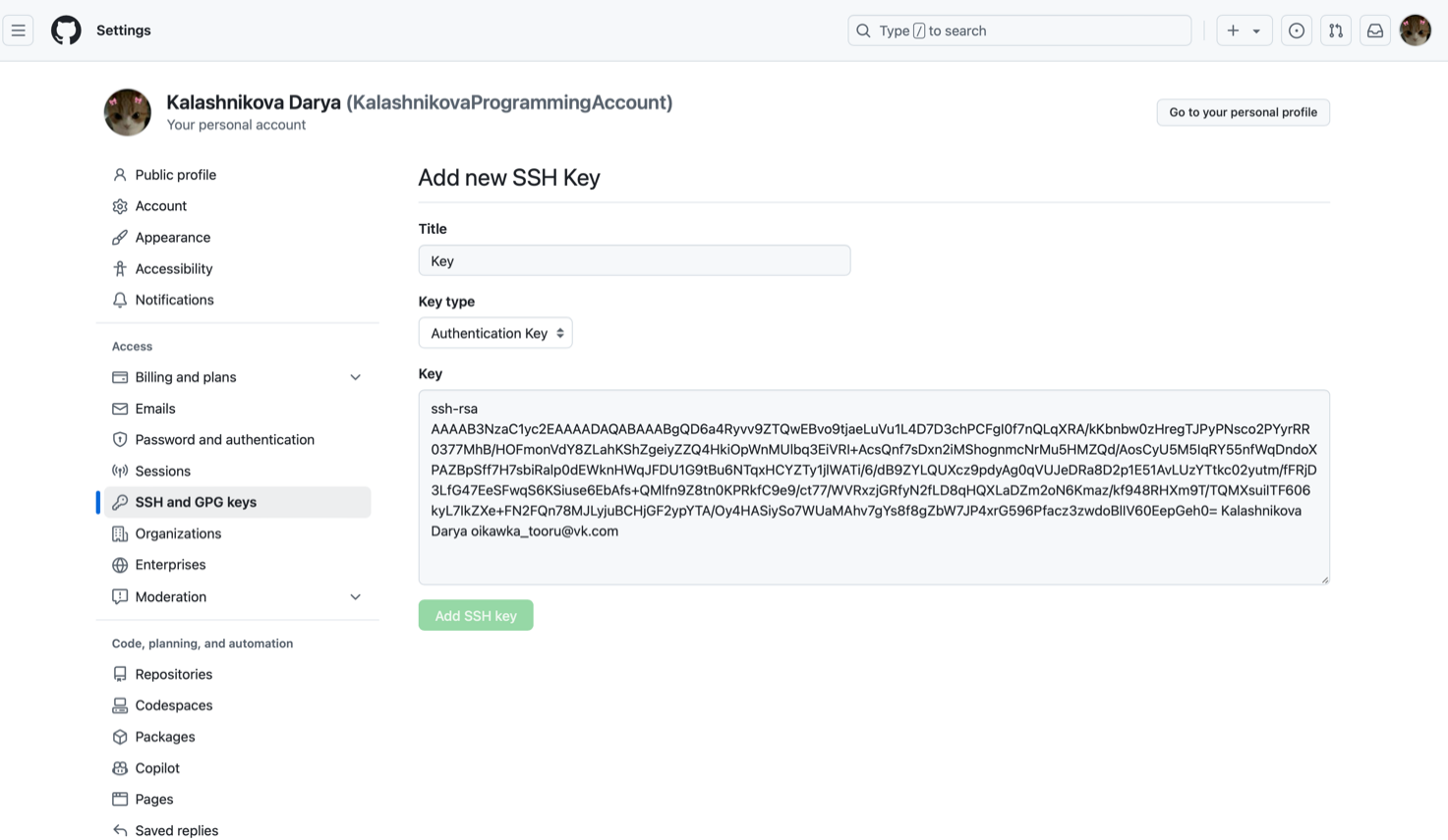


Рис. 11: Заполнение всех необходимых полей для добавления ключа.

Теперь нам необходимо организовать наше рабочее пространство. Для этого создадим каталог “Архитектура компьютера” по следующему адресу: (рис. 12).

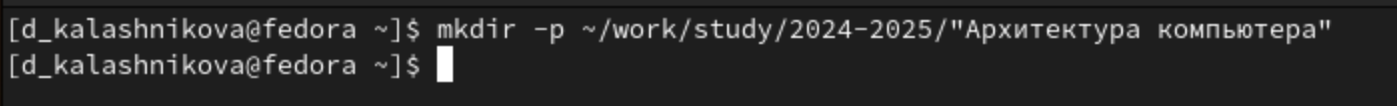


Рис. 12: Создание каталога “Архитектура компьютера”.

После этого нам нужно будет создать репозиторий. Мы будем его создавать на основе шаблона, который находится по следующему адресу: https://github.com/yamadharma/cour se-directory-student-template. (рис. 13).



Рис. 13: Страница шаблона на GitHub.

Нажимаем на кнопку “Use this template”, и в предложенных опциях выбираем “Create new repository”. Далее нас переносит на следующую страницу (рис. 15). Здесь задаём имя нашего репозитория. Он будет называться так: study\_2024-2025\_arh-pc: (рис. 14).

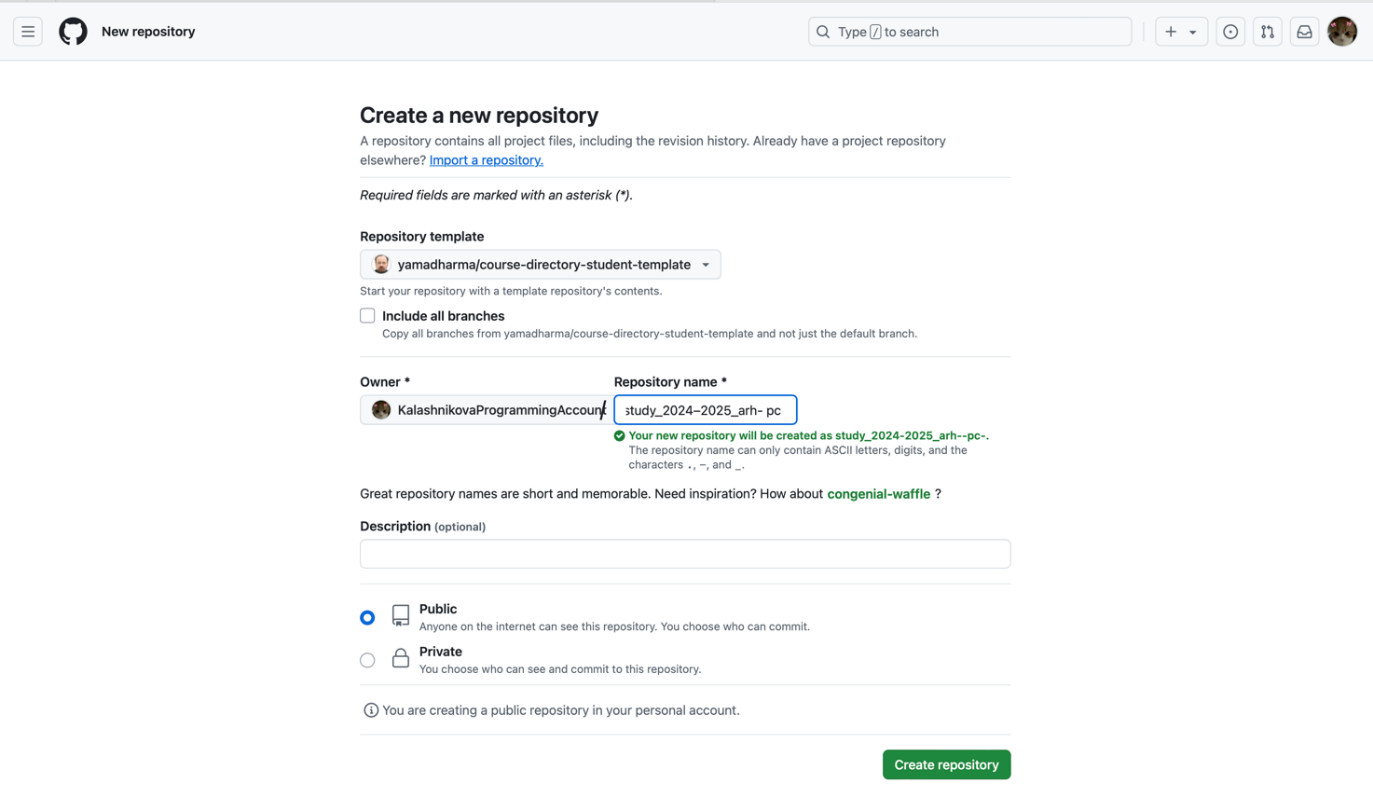


Рис. 14: Создание репозитория на основе шаблона.

Теперь нам нужно клонировать репозиторий на наш компьютер. Для этого перейдем в папку, в которую мы хотим скопировать репозиторий. В нашем случае это заранее созданная папка:(рис. 15).

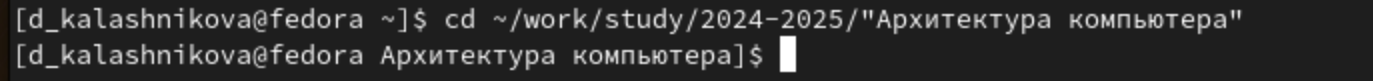


Рис. 15: Перемещение в папку, куда скачается репозиторий.

Теперь перейдем непосредственно к клонированию. Для этого воспользуемся командой git clone, в аргументе указав ссылку на репозиторий. Ссылку можно найти при нажатии на кнопку код на странице нашего репозитория. (рис. 16).(рис. 17).

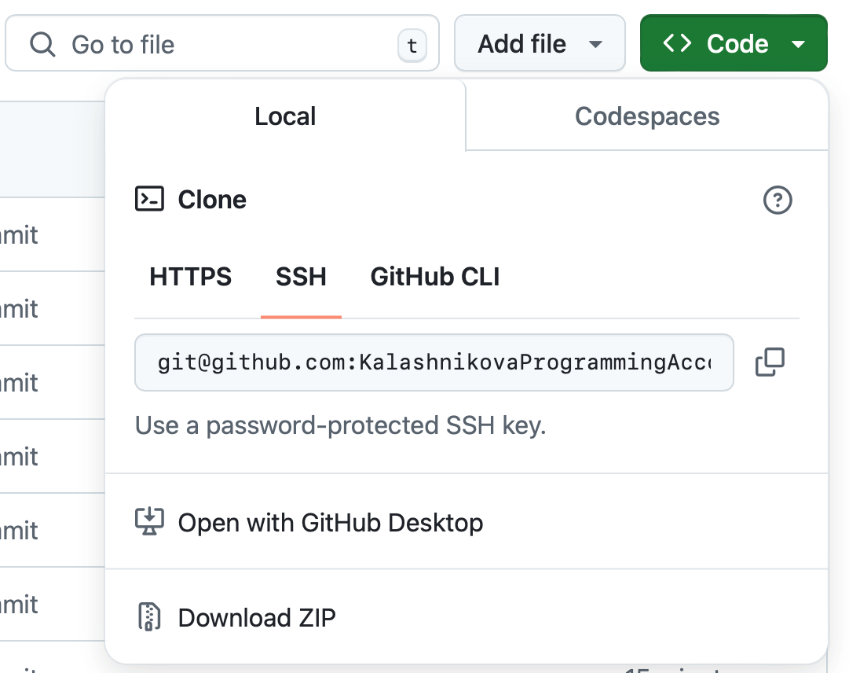


Рис. 16: Ссылка на наш репозиторий.

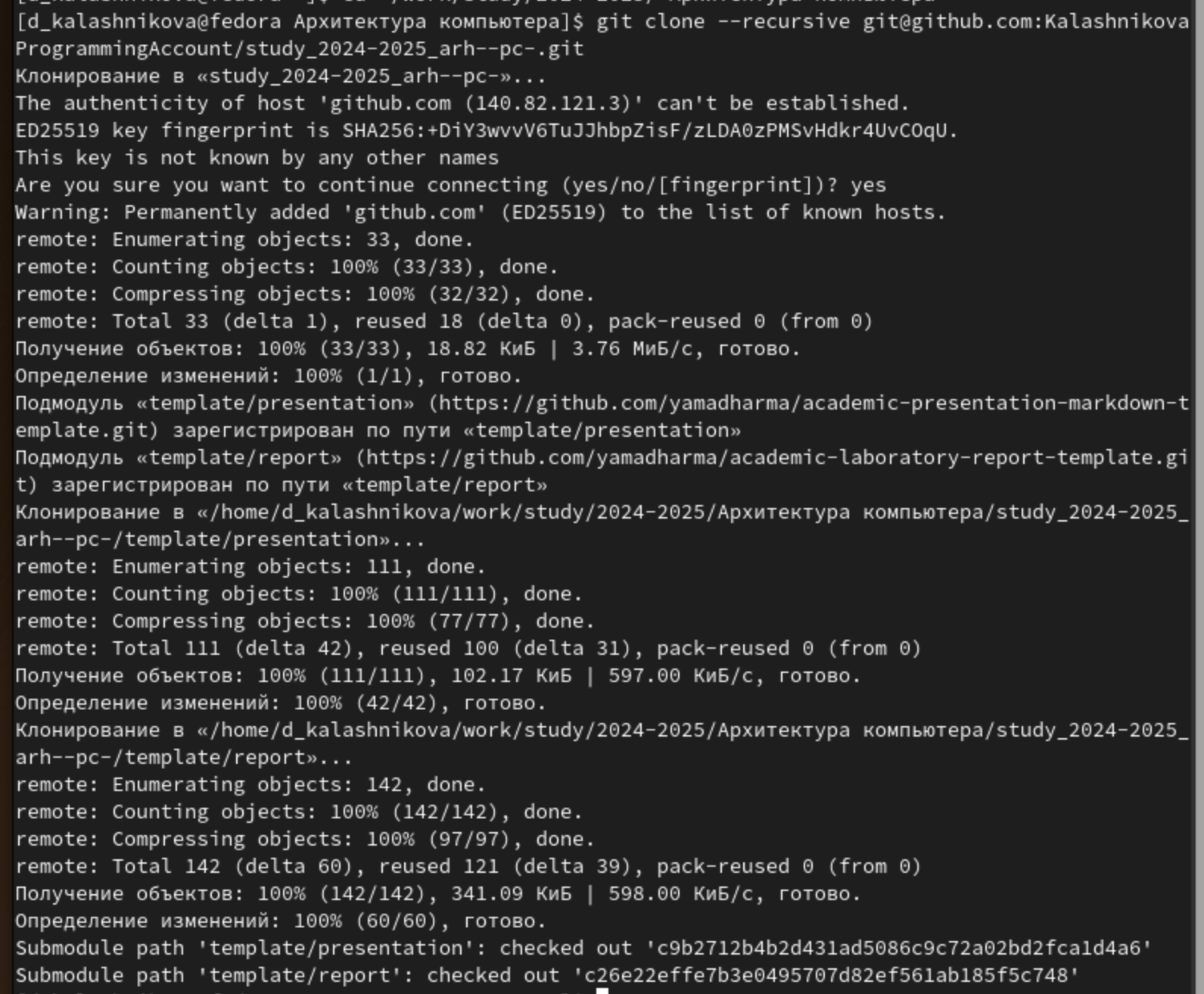


Рис. 17: Скачивание репозитория командой “git clone”.

Теперь перейдем к настройке клонированного каталога. Для начала перейдем в него с помощью команды “cd”: (рис. 18).



Рис. 18: Переход в клонированный каталог.

Удалим с помощью команды “rm” лишний файл: (рис. 19).

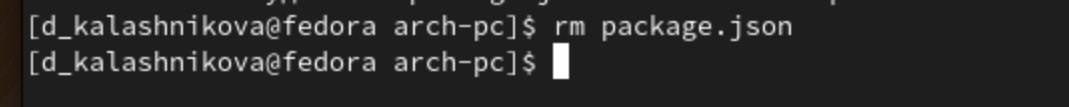


Рис. 19: Удаление файла.

Теперь создадим необходимые файлы. В нашем случае, это будет файл COURSE. Чтобы его создать, мы воспользуемся командой echo, которая запишет в файл строку “acrh-pc” и автоматически создаст его, т.к. этого файла раньше не существовало: (рис. 20).

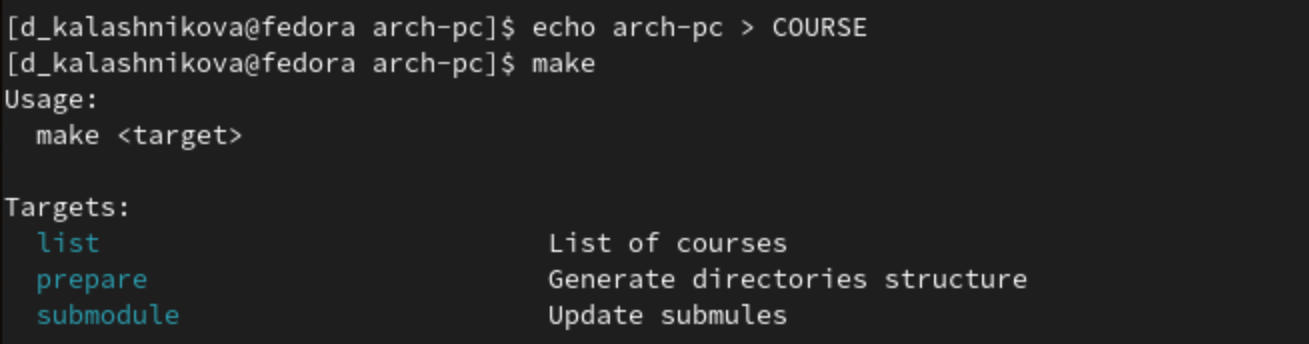


Рис. 20: Создание файла и запись в него строки.

Теперь нам остается лишь отправить файлы на сервер. Для этого с помощью команды git add мы добавим каталоги, которые должны отправляться на сервер. В качестве аргумента мы возьмем точку, которая укажет на то, что мы должны отправить на сервер все файлы и каталоги, которые по иерархии находятся ниже нашего текущего расположения: (рис. 21).

Добавление каталога для отправки на сервер.

Рис. 21: Добавление каталога для отправки на сервер.

Теперь с помощью команды git commit мы сохраним изменения и укажем комментарий, в котором будет поясняться, какие изменения мы сделали. В данном случае в комментарии мы напишем, что создали структуру курса: (рис. 22).

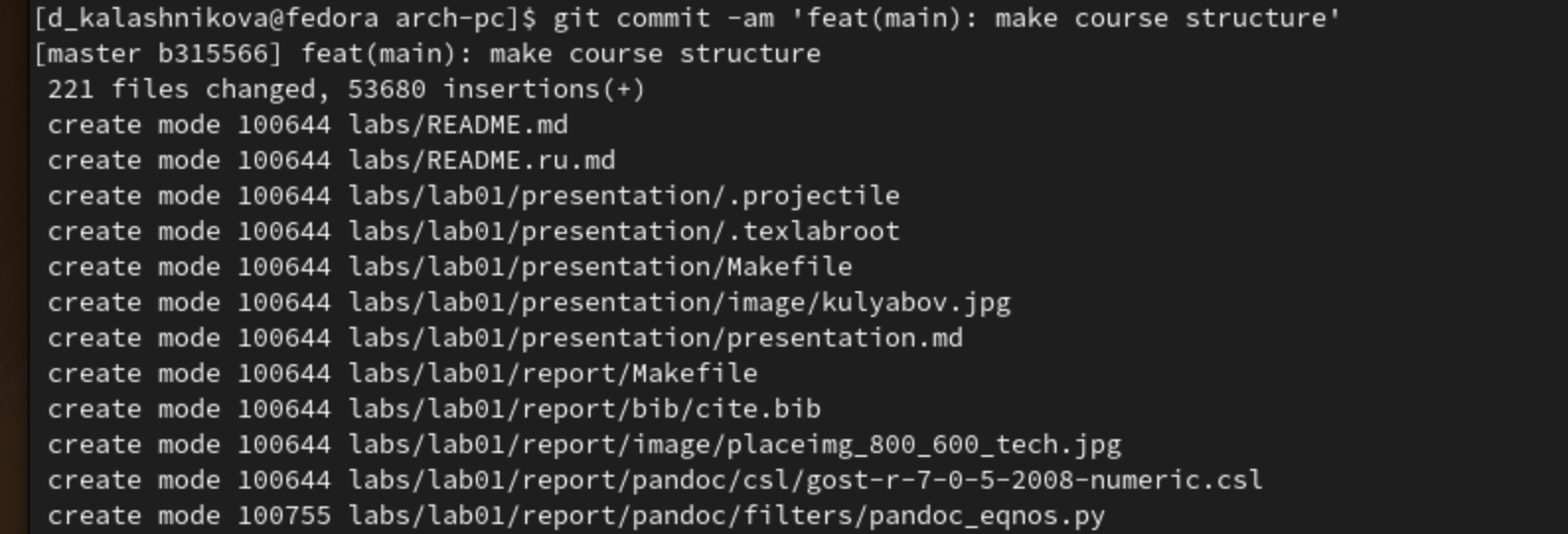


Рис. 22: Сохранение изменений и указание комментария.

Теперь нам осталось окончательно загрузить изменения на сервер. Для этого мы воспользуемся командой git push: (рис. 23).

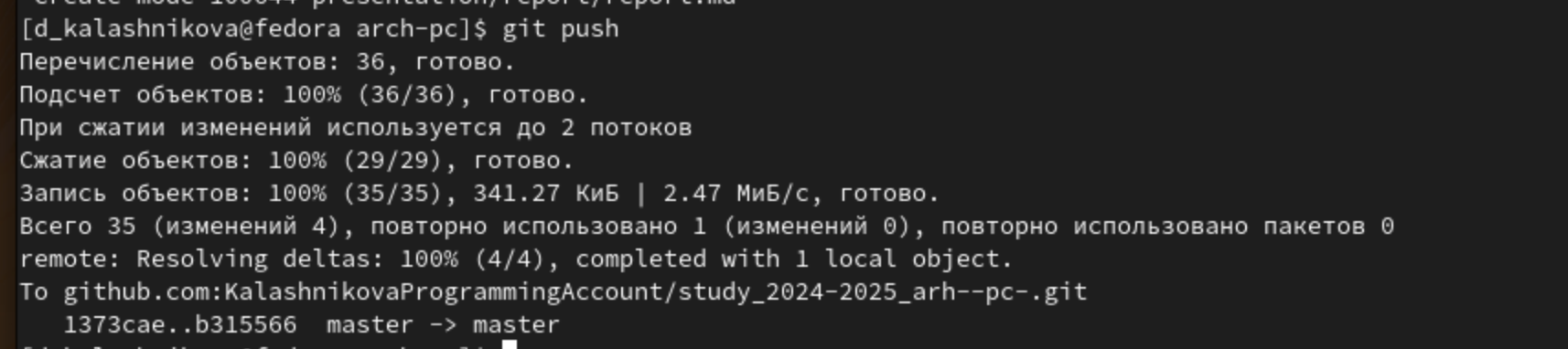


Рис. 23: Отправка репозитория на сервер GitHub.

Остается лишь проверить, сохранились ли файлы на сервере. Сравним файлы, которые находятся на GitHub с теми файлами, что находятся на нашем компьютере: (рис. 24).(рис. 25).

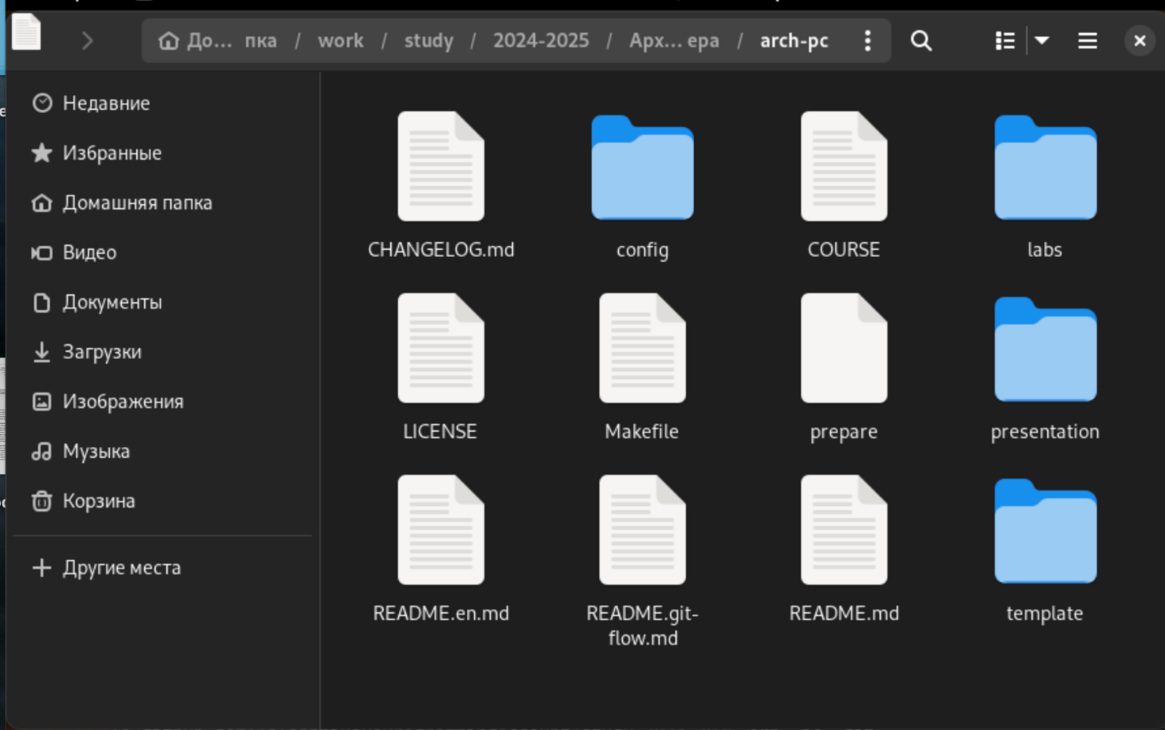


Рис. 24: Файлы на компьютере.

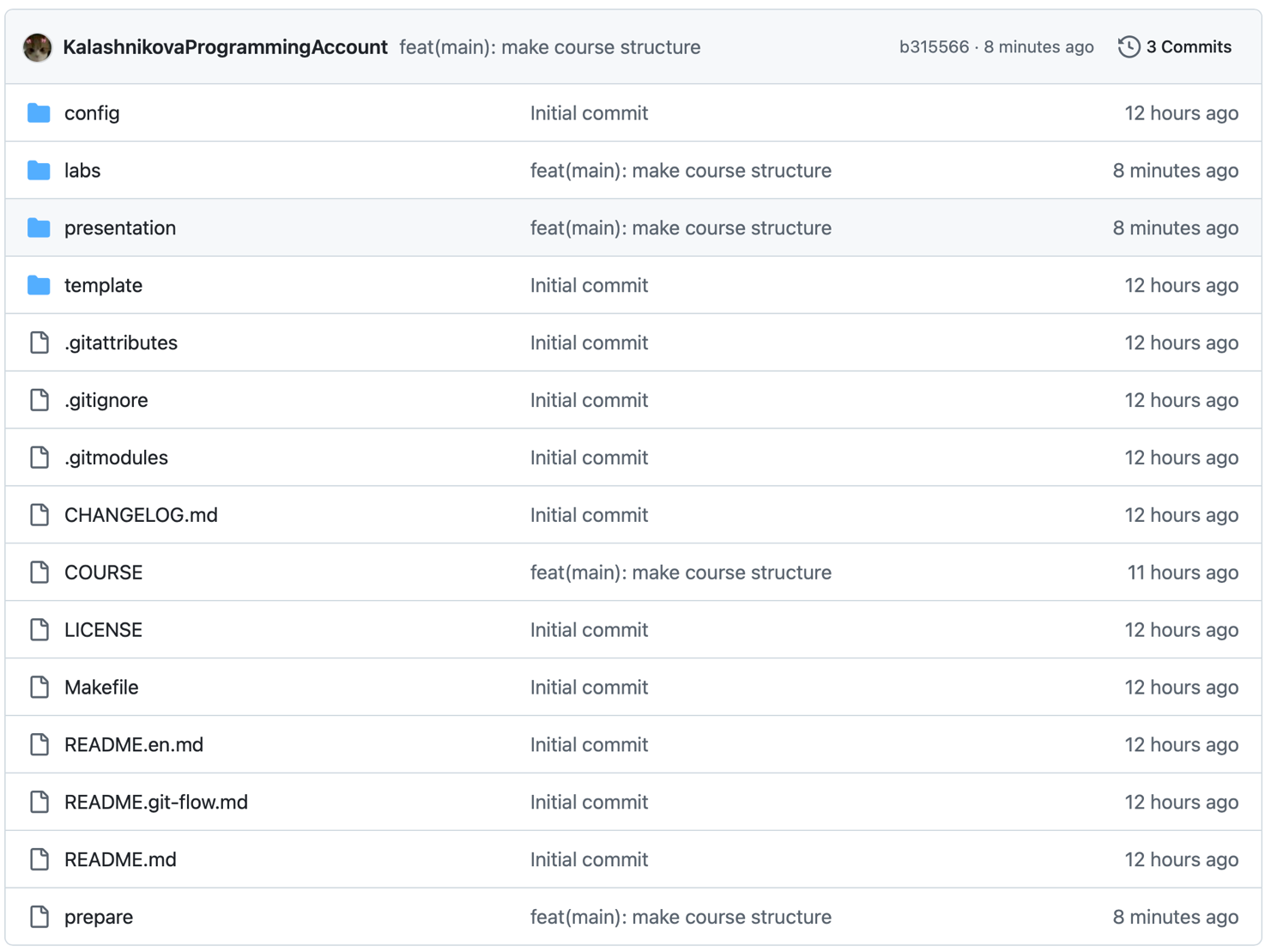


Рис. 25: Файлы на сервере.

Всё совпало.

# 3 Задание для самостоятельной работы

Задание №1. Теперь приступим к выполнению самостоятельной работы. Для начала мы создадим файл отчета для нашей лабораторной работы в папке labs/lab02/report с помощью LibreOffice. (рис. 26).

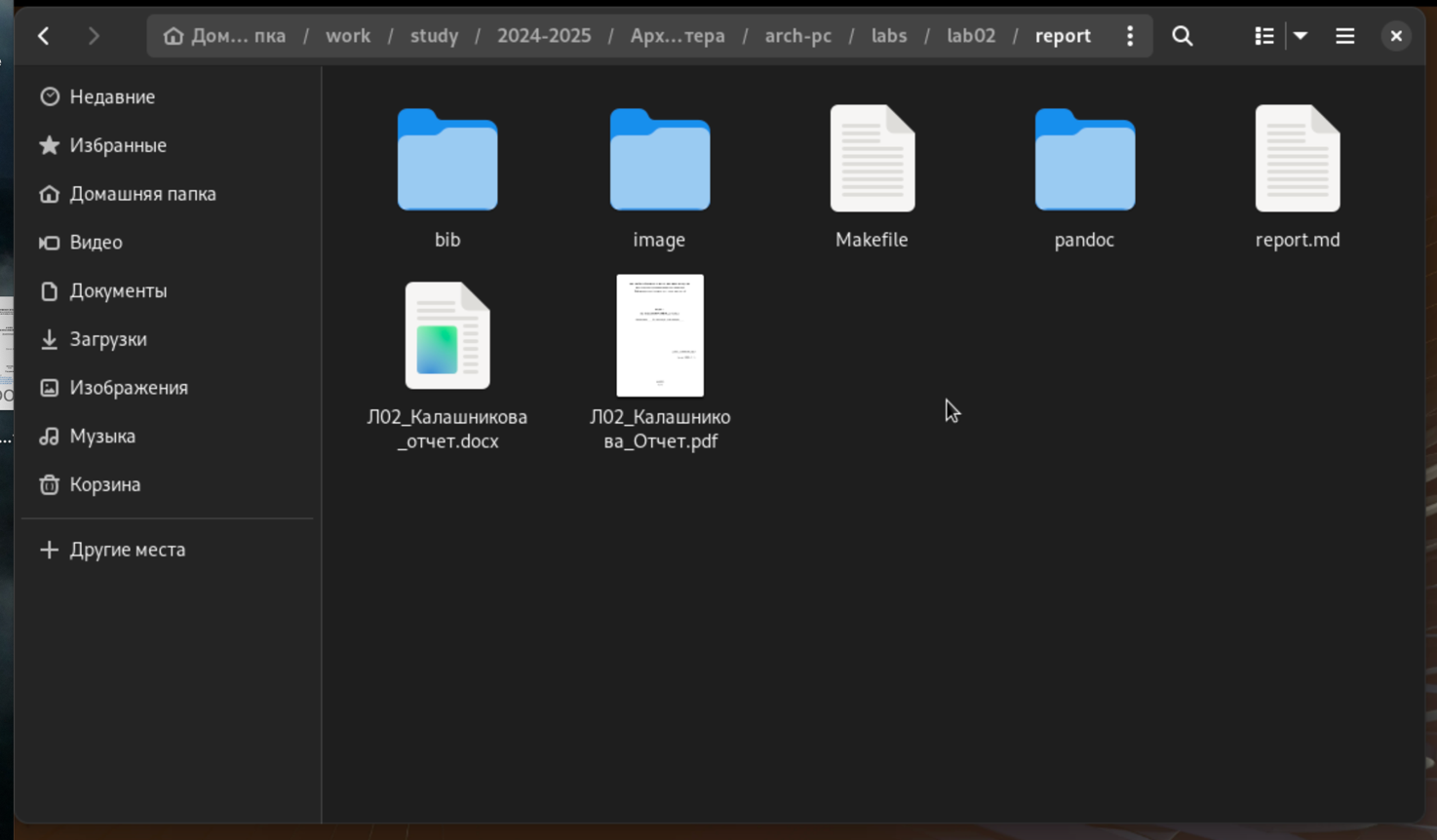


Рис. 26: Наличие отчёта в папке.

Задание №2. Копируем отчет по нашей предыдущей лабораторной работе в соответствующую папку созданного нами рабочего пространства, то есть в папку labs/lab01/report. Для копирования воспользуемся командой “cp”: (рис. 27).

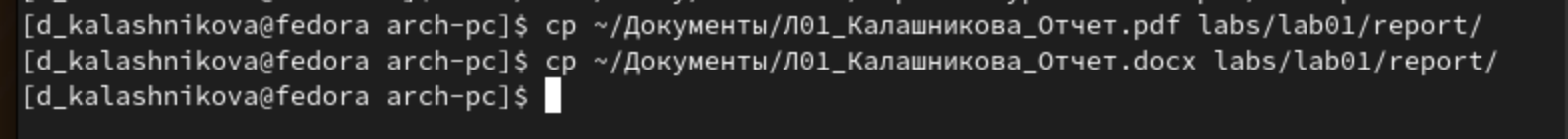


Рис. 27: Копирование отчёта по первой лабораторной работе в папку lab01.

Задание №3. Теперь нам осталось лишь загрузить изменения на GitHub. Для этого нам нужно повторить те шаги, которые мы выполнили в предыдущих пунктах лабораторной работы, а именно, воспользоваться командой “git add .” для того, чтобы указать, что мы хотим сохранить изменения во всех файлах, находящихся в нашем каталоге. (рис. 28).



Рис. 28: Добавление каталога для отправки на сервер.

После этого с помощью команды “git commit” мы укажем комментарий и сохраним изменения. В комментарии мы укажем, что мы загрузили первую и вторую лабораторные работы. (рис. 29).

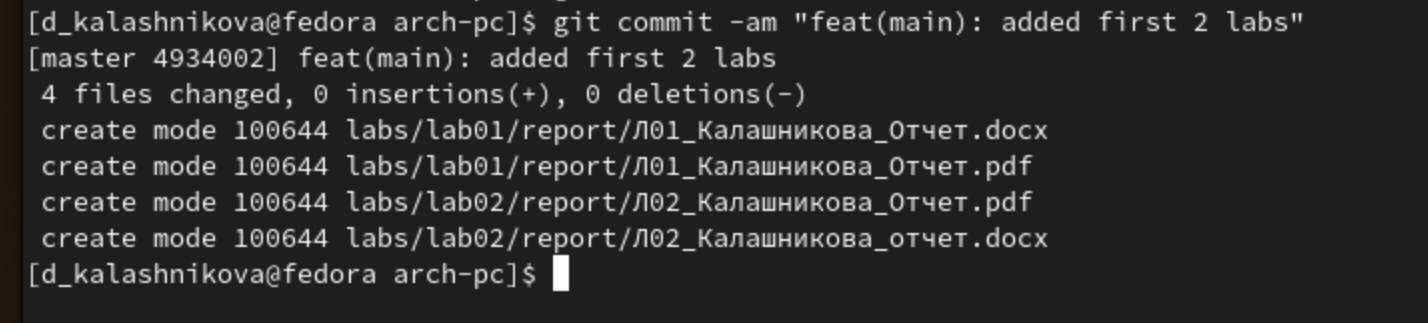


Рис. 29: Сохранение изменений и указание комментария.

После этого вводим команду “git push” для того, чтобы загрузить наши файлы на GitHub. (рис. 30).

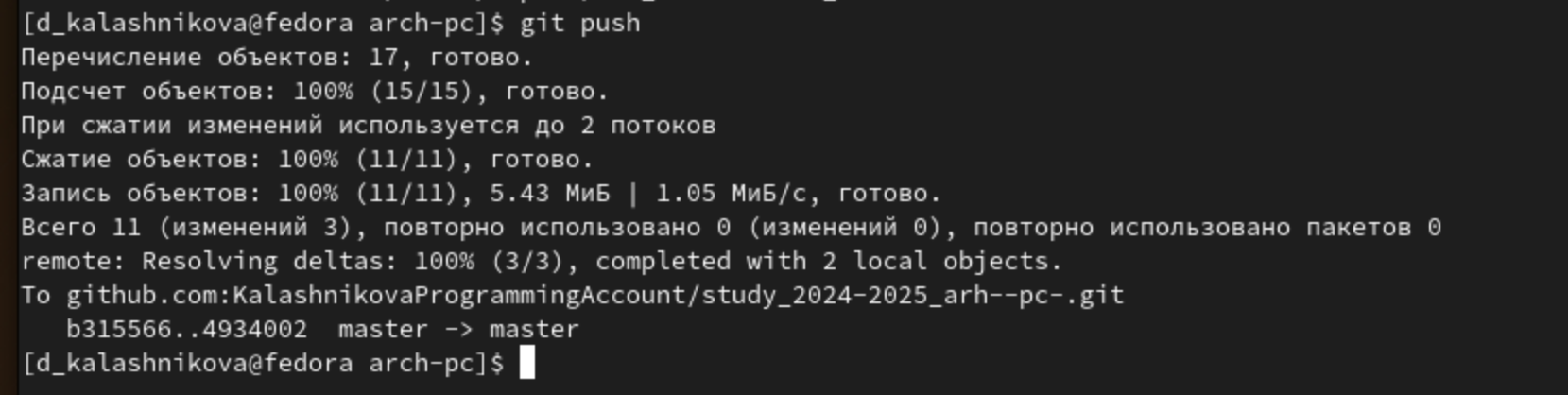


Рис. 30: Загрузка новых файлов на GitHub.

Остается лишь проверить, правильно ли мы все загрузили. Для этого посмотрим время обновления файлов в папке labs в GitHub: (рис. 31).(рис. 32).(рис. 33).

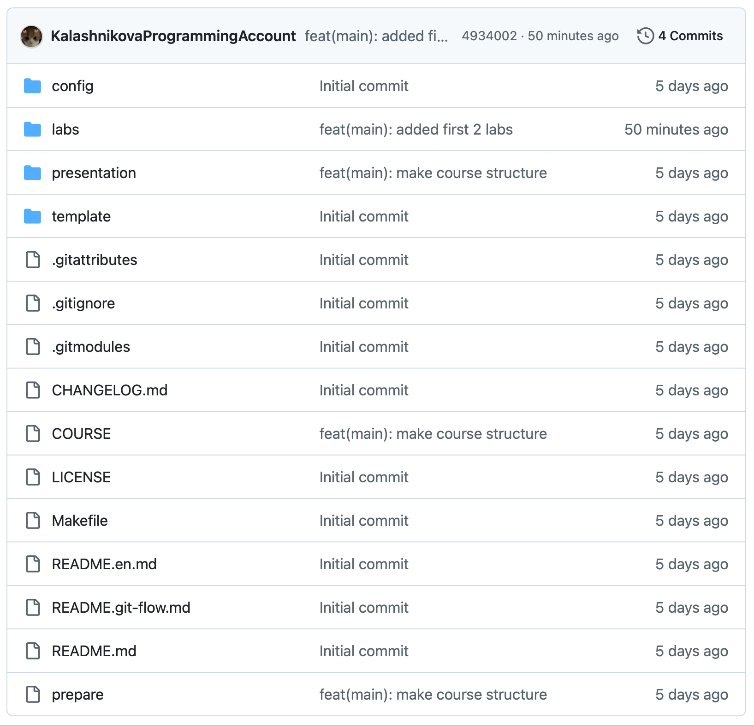


Рис. 31: Вид обновлённого репозитория на GitHub.

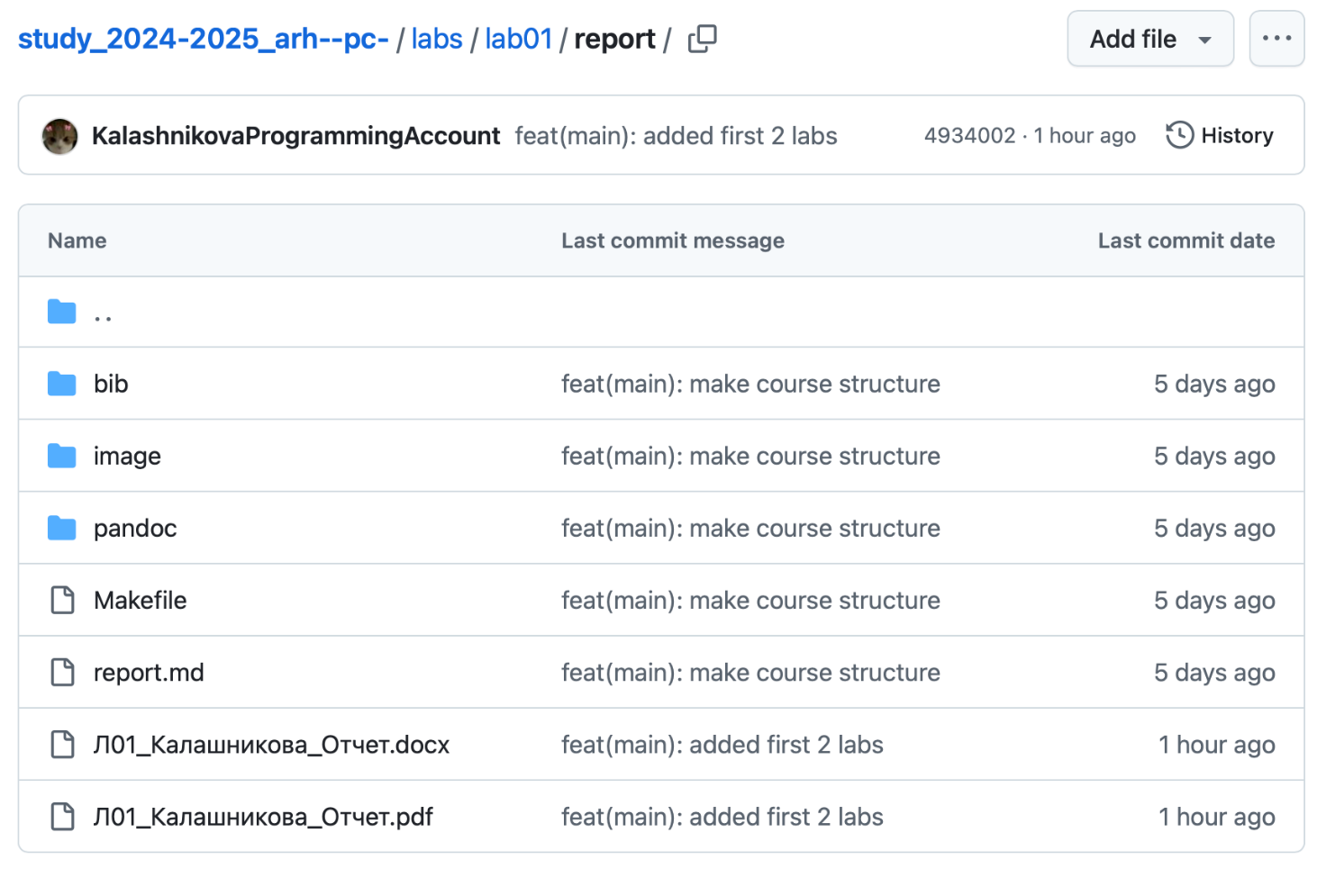


Рис. 32: Наличие отчетов на GitHub.

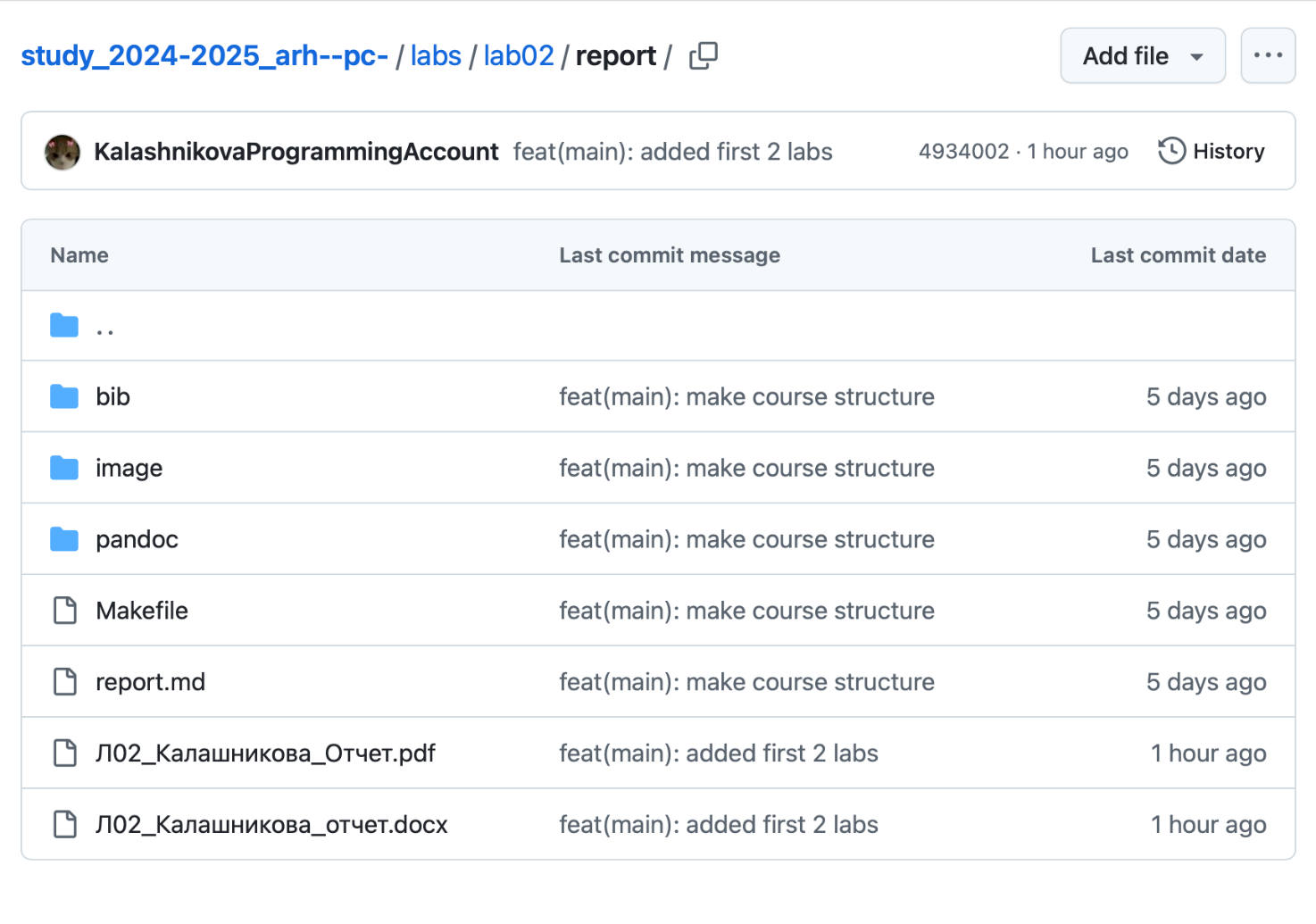


Рис. 33: Наличие отчетов на GitHub.

# 4 Выводы

В результате выполнения лабораторной работы появились практические навыки работы с системой контроля версий Git, была произведена её первоначальная настройка в linux. Было изучено, как создавать репозитории, сохранять изменения и добавлять к ним комментарии, а также как выгружать файлы на сервер. Были приобретены навыки работы с платформой GitHub.