**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Компьютерные и информационные науки**

**ОТЧЕТ**

**по лабораторной работе № 2**

*дисциплина: Архитектура компьютера*

Студент: Нечаева Кира

Группа: НКАбд-04-23

**МОСКВА**

2023 г.

**Содержание**

1. **Цель работы………………………………………………………………………..3**
2. **Задание……………………………………………………………………………...4**
3. **Теоретическое введение…………………………………………………………..5**
4. **Выполнение лабораторной работы……………………………………………...7**
5. **Выводы…………………………………………………………………………….15**
6. **Вопросы для самопроверки..……………………………………………………16**
7. **Источники…………………………………………………………………………17**

**1 Цель работы**

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

**2 Задание**

1. Настройка github
2. Базовая настройка git
3. Создание SSH ключа
4. Создание рабочего пространства на основе шаблона
5. Создание репозитория курса на основе шаблона
6. Настройка каталога курса
7. Задание для самостоятельной работы

**3 Теоретическое введение**

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется.

В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных.

Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зависимости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом.

Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд.

**4 Выолнение лабораторной работы**

1. **Настройка github**

Поскольку у мееня уже есть аккаунт на Github, я пропускаю этот шаг. (рис. 1)

Рис. 1. Аккаунт на github

**2. Базовая настройка git**

Сначала сделаю предварительную конфигурацию git. Открываю терминал и ввожу команды:

git config --global user.name "<Kira Nechaeva>”

git config --global user.email "<kirusya1234@gmail.com>”

указав свои имя и email.

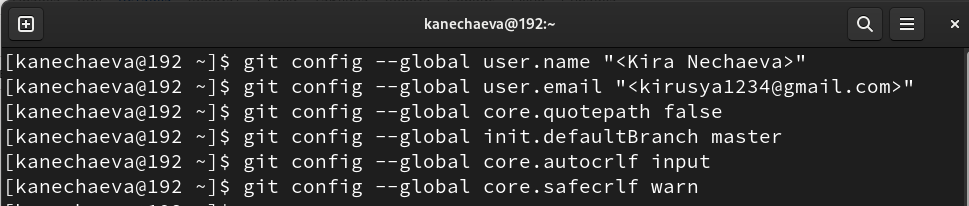
Настраиваю utf-8 в выводе сообщений git для верного отображения символов.

Задаю имя начальной ветки “master”.

Задаю параметр autocrlf со значением input.

Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость.

Все действия отображены в рис.2 на следующей странице.

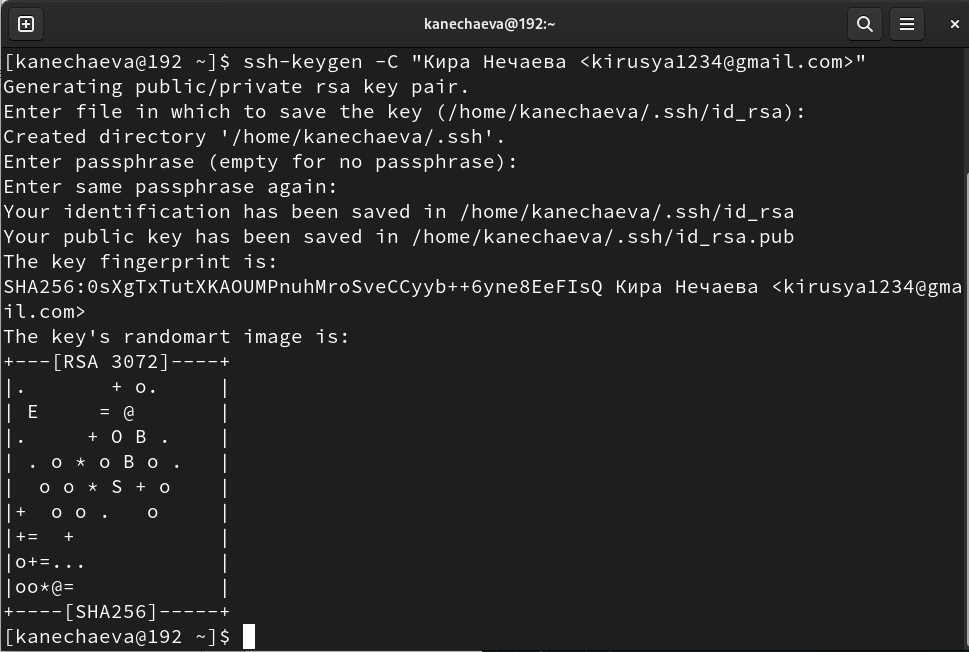
Рис.2. Базовые настройки Git.

**3. Создание SSH ключа**

Для последующей идентификации пользователя на сервере репозиториев генерирую пару ключей (приватный и открытый) с помощью команды

ssh-keygen -C "Кира Нечаева <[kirusya1234@gmail.com](mailto:kirusya1234@gmail.com)>”

Ключи сохраняются в каталоге ~/.ssh/ автоматически. (рис.3)

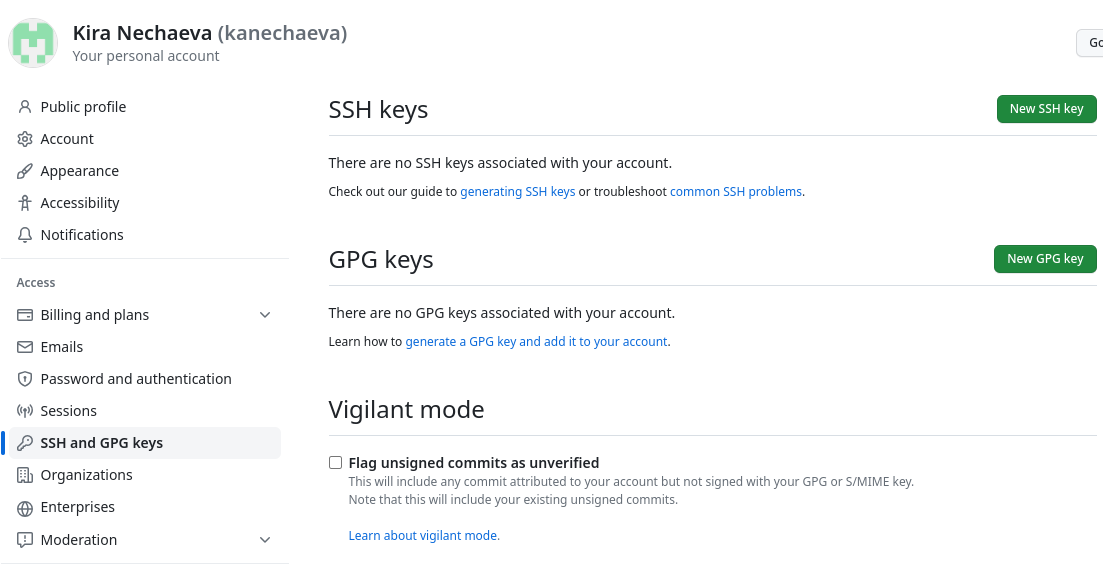
Рис.3. Создание SSH ключа.

Копирую открытый ключ из директории, в которой он был сохранен, с помо-

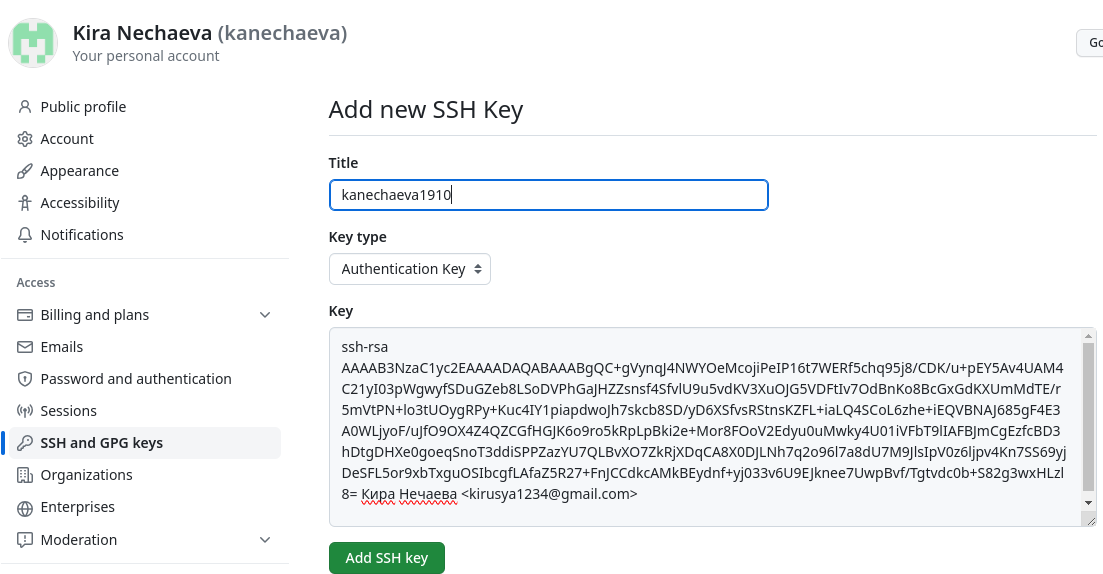
щью утилиты xclip. (рис. 4)

Рис. 4. Копирование содержимого файла

Захожу на сайт GitHub. Выбираю в меню “Настройки”, затем “SSH and GPG keys”. Нажимаю кнопку «New SSH key». (рис. 5)

Рис. 5. Окно SSH and GPG keys.

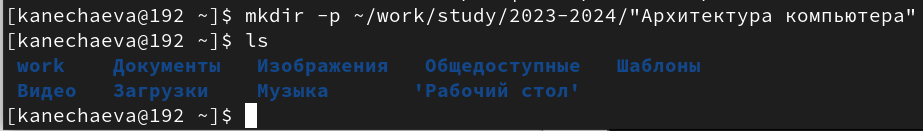
Вставляю скопированный ключ в поле «Key». Называю его kanechaeva1910. Нажимаю «Add SSH-key», чтобы завершить добавление ключа. (рис. 6)

Рис. 6. Добавление ключа

**4. Создание рабочего пространства на основе шаблона**

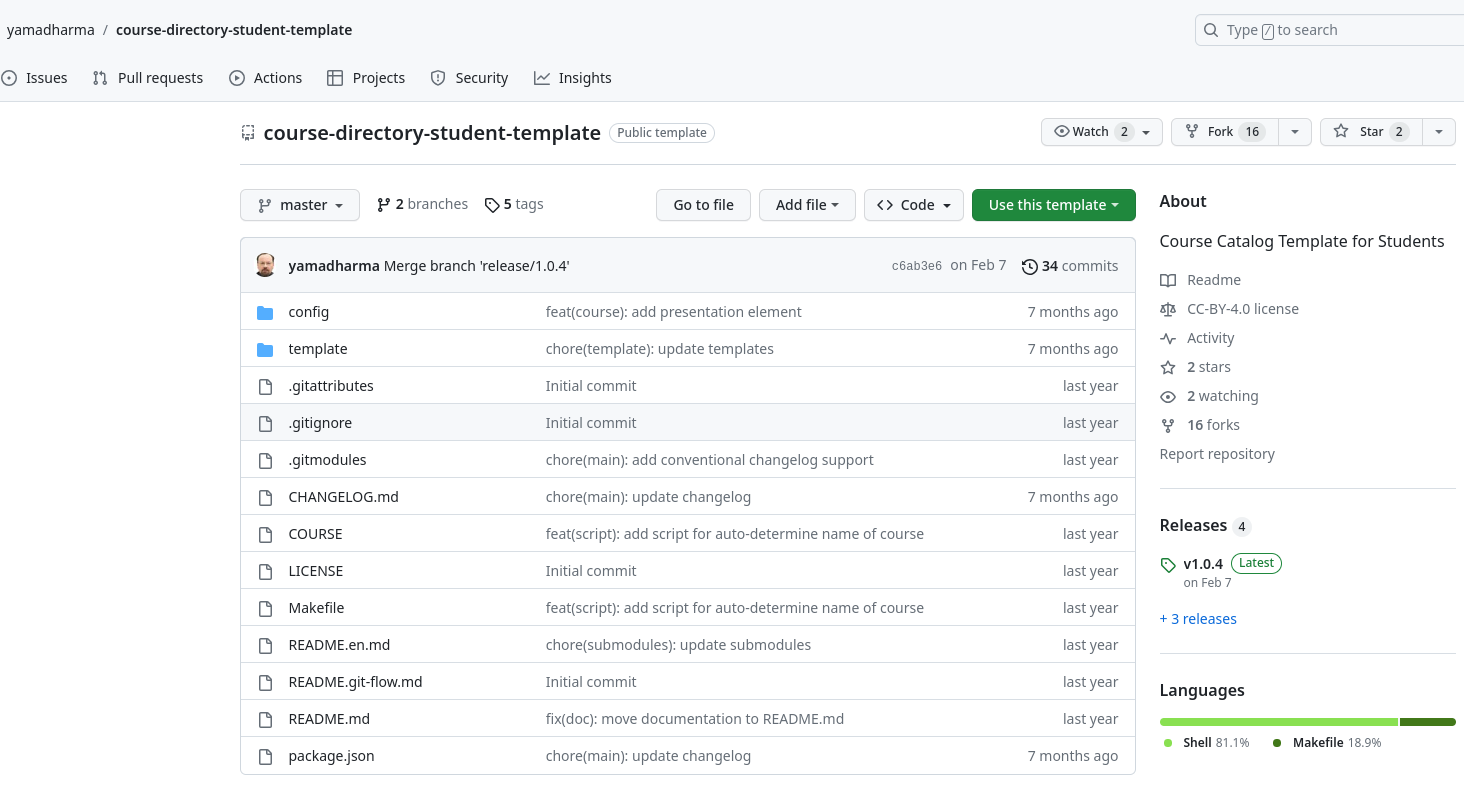
Захожу в терминал. Создаю директорию, рабочее пространство, с помощью утилиты mkdir, блягодаря ключу -p создаю все директории после домашней ~/work/study/2023-2024/"Архитектура компьютера" рекурсивно.

Далее проверяю с помощью ls корректность выполнение своих действий. (рис. 7).

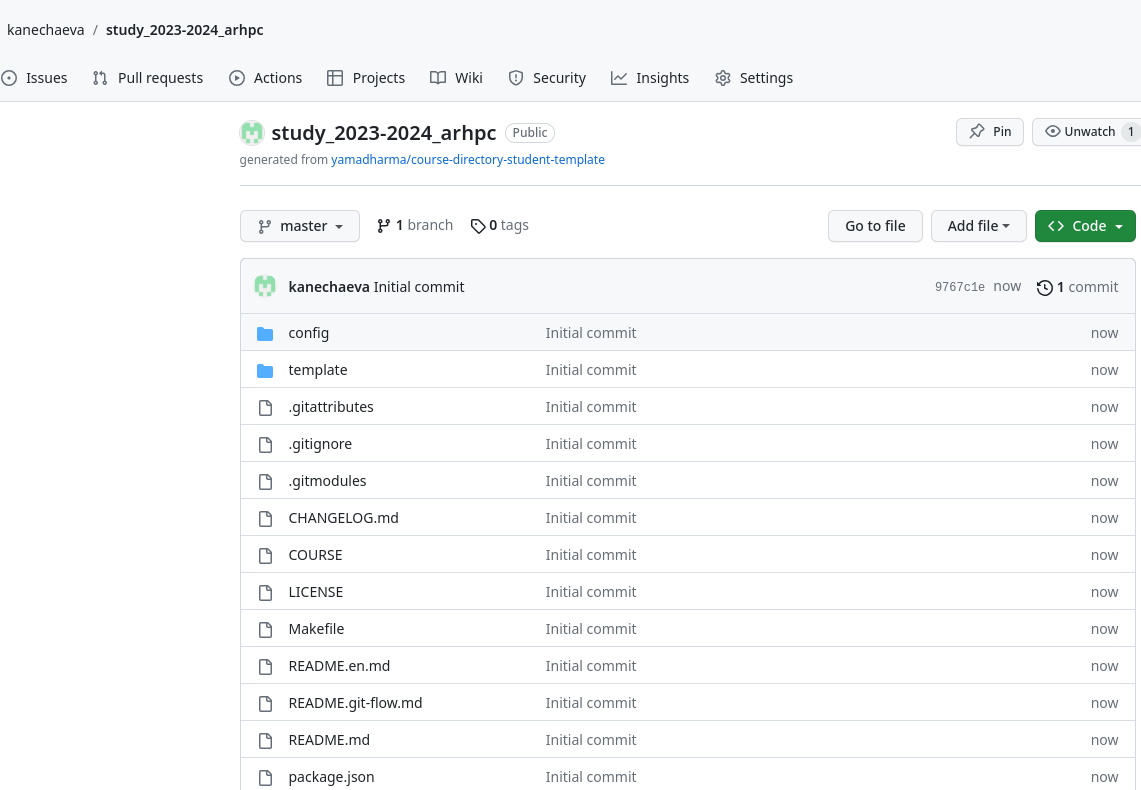
Рис. 7. Создание рабочего пространства.

**5. Создание репозитория курса на основе шаблона**

Репозиторий на основе шаблона создаю через web-интерфейс github. Перехожу на станицу репозитория с шаблоном курса https://github.com/yamadharma/cour se-directory-student-template. Далее выбераю Use this template. (рис. 8)

Рис. 8. Шаблон для репозитория.

В открывшемся окне задаю имя репозитория study\_2023–2024\_arhpc и создаю репозиторий, нажав на кнопку Create repository from template. К сожалению, я не успела сделать скриншот создания репозитория, так что прикрепляю фотографию с созданным репозиторием. (рис. 9)

****Рис. 9. Созданный репозиторий.

Открывю терминал и перехожу в каталог курса с помощью команды cd (рис. 10).

Рис. 10. Перемещение в каталог курса.

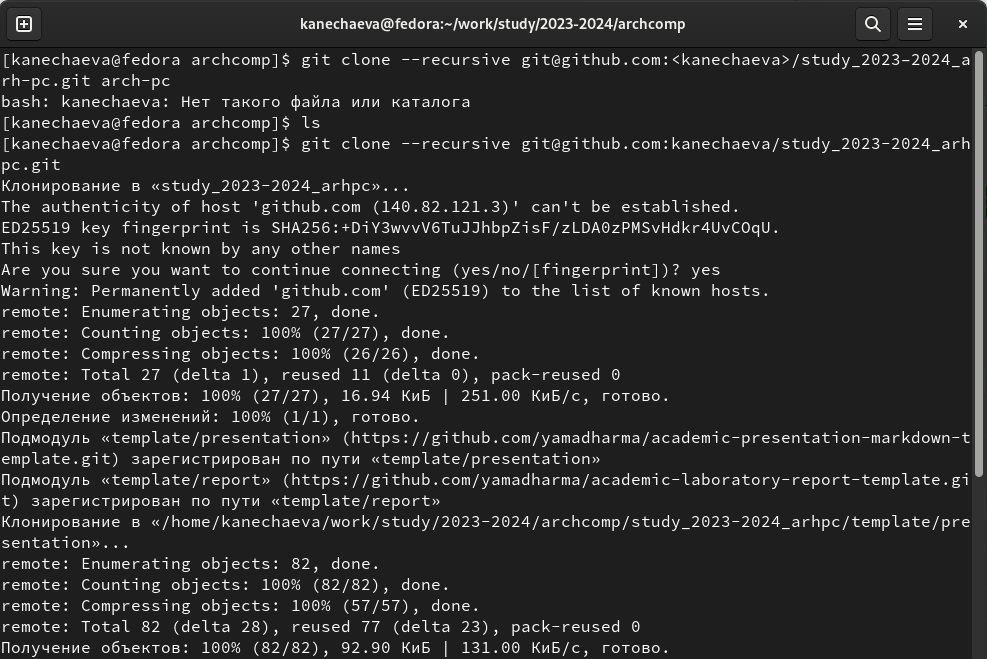
Затем клонирую созданный репозиторий. (рис. 11)

Рис. 11. Клонирование репозитория

Затем проверяю корректность своих действий с помощью утилиты ls (рис.12).

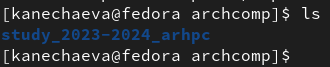


Рис. 12. Наличие папки в директории archcomp

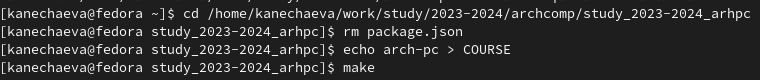
**6. Настройка каталога курса**

Перехожу в каталог курса с помощью утилиты cd.

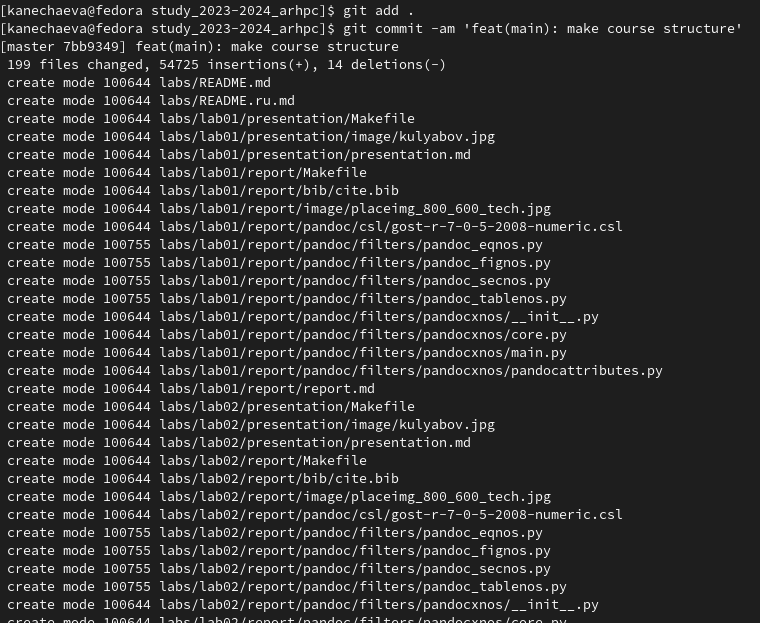
Затем удаляю лишний файл package.json, используя команду rm.

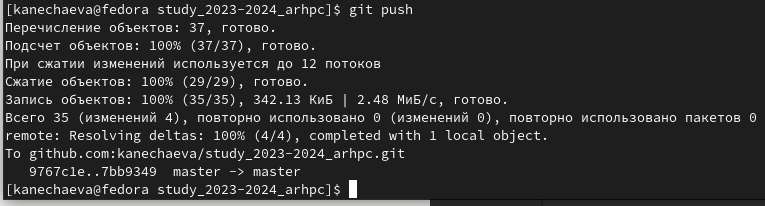
После чего создаю необходимые каталоги.

Все действия отображены в рис. 13.

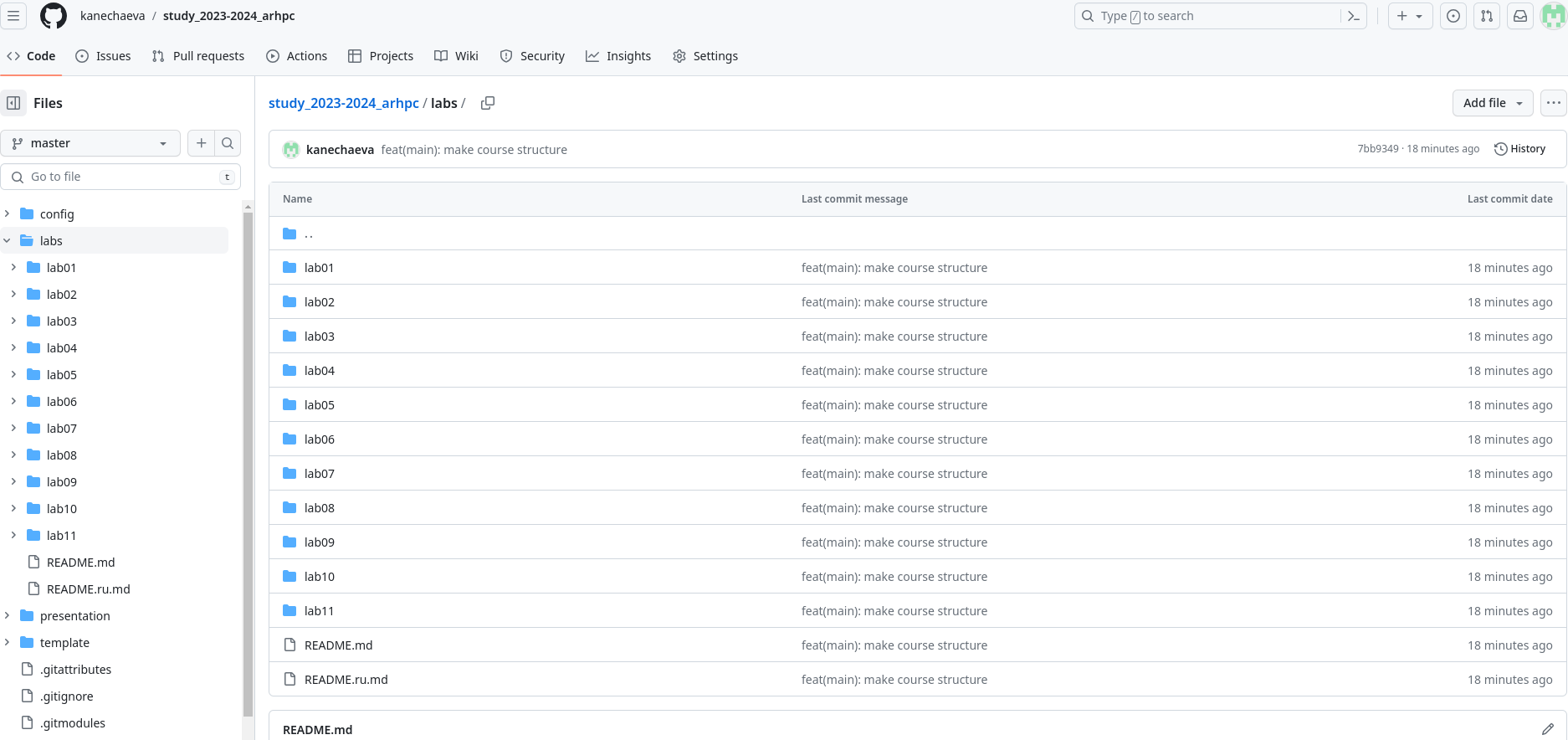
Рис. 13. Удаление и создание необходимых файлов и каталогов.

Далее отправляю файлы на сервер (рис. 14 и рис. 15).

Рис. 14. Добавление и сохранение изменений на сервере

Рис. 15. Выгрузка изменений.

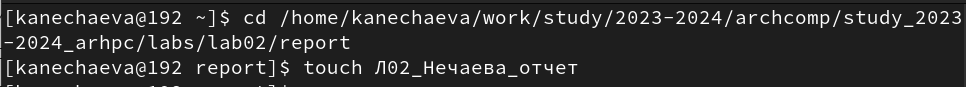
Проверяю правильность создания иерархии рабочего пространства в локальном репозитории и на странице github. (рис. 16)

Рис. 16. Проверка проделанных действий.

**7. Задание для самостоятельной работы**

№1

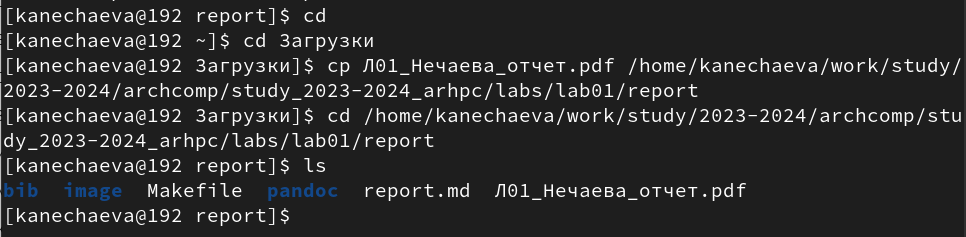
Перехожу в директорию labs/lab02/report. Создаю в каталоге файл для отчета по второй лабораторной работе.(рис. 17)

Рис. 17. Создание файла для отчета по лабораторной работе

Оформляю отчет я в текстовом процессоре LibreOffice Writer.

№2

Далее я перехожу в каталог “Загрузки”, чтоб скопировать из него отчет по первой лабораторной работе в директорию labs/lab1/report. (рис. 18)

Рис. 18. копирование файла Л01\_Нечаева\_отчет в другую директорию

№3

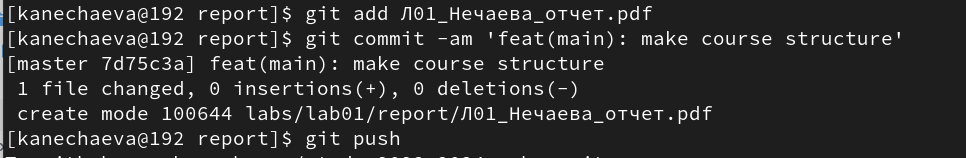
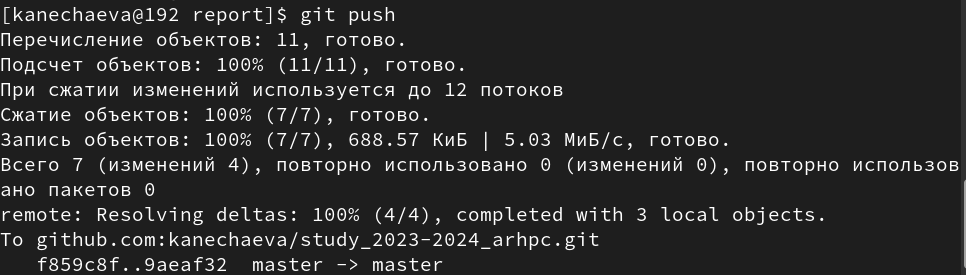
Загружаю файл Л01\_Нечаева\_отчет с помощью таких команд, как:

git add – добавление файла

git commit -am 'feat(main): make course structure' – сохранение изменений

git push – отправка изменений в репозиторий.

Все действия отображены на рис. 19.

Рис. 19. Загрузка файлов на github

Так как сейчас я взаимодействую с отчетом по второй лабороторной работе, его загркузку на githab я осуществлю позднее по тому же алгоритму.

**5 Вывод**

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий и приобрела практические навыки по работе с системой git.

**7 Источники**

1. ТУИС – Архитектура ЭВМ – [Электронный ресурс] - <https://esystem.rudn.ru/mod/resource/view.php?id=1030548>